Arm Cortex-M55 AT633 and Cortex-M55 with FPU AT634

Software Developer Errata Notice

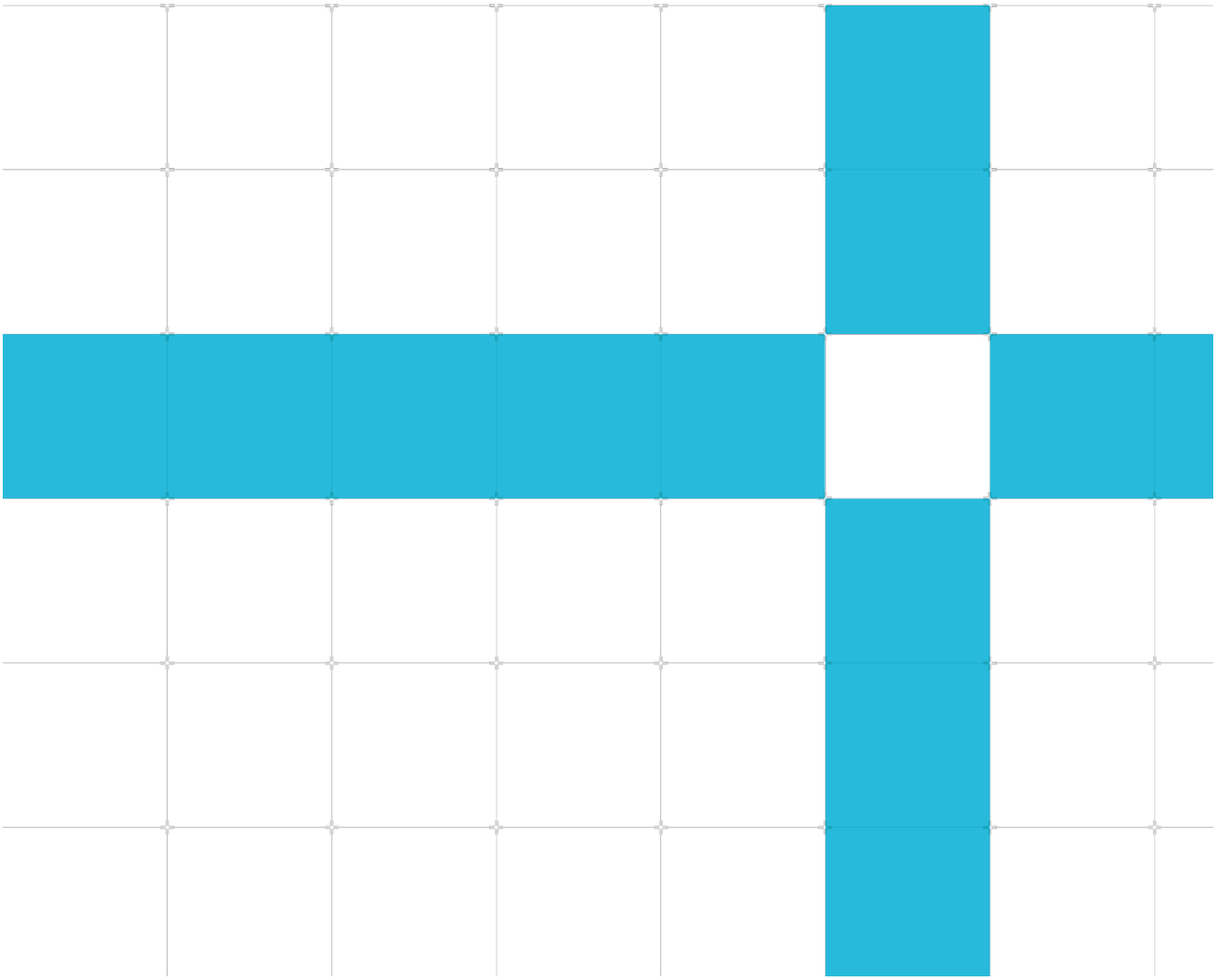
Date of issue: 08-Jun-2023

Non-Confidential

Document version: 13.0

Document ID: SDEN-1679655

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-M55 AT633 and Cortex-M55 with FPU AT634, 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.

08-Jun-2023: Changes in document version v13.0

ID	Status	Area	Category	Summary
2791503	New	Programmer	Category C	ITM can generate Synchronization and timestamp packets when NIDEN=0

18-Aug-2022: Changes in document version v12.0

ID	Status	Area	Category	Summary
2675831	New	Programmer	Category C	After deactivating the instruction cache self-modified code might not be executed correctly
2703268	New	Programmer	Category C	L1D_CACHE_REFILL and L1D_CACHE_MISS_RD PMU events might be inaccurate

17-May-2022: Changes in document version v11.0

ID	Status	Area	Category	Summary
2454643	New	Programmer	Category C	A VSTMDB which specifies unimplemented registers may write to unexpected memory locations
2461408	New	Programmer	Category C	Unprivileged debugger access to DPDLPSTATE register incorrectly modifies register state
2567270	New	Programmer	Category C	A BX using the LR where the t-bit (bit[0]) of the LR is not set, under certain circumstances, might not cause a INVSTATE fault
2619476	New	Programmer	Category C	TCM ECC errors for S-AHB accesses will not be reported in sleep state
2653102	New	Programmer	Category C	Unprivileged debug transactions to the STIR and EVENTSPR registers can pend interrupts

28-Jan-2022: Changes in document version v10.0

ID	Status	Area	Category	Summary
2428419	New	Programmer	Category B	Automatic EWIC drain will not function if the processor is powered down during sleep when the secure extension is not configured
2397983	New	Programmer	Category C	PMU event L1I_CACHE is incorrectly implemented in the instruction cache

22-Nov-2021: Changes in document version v9.0

No new or updated errata in this document version.

24-Sep-2021: Changes in document version v8.0

ID	Status	Area	Category	Summary
2233762	New	Programmer	Category B	An interrupted vector instruction with implicit LE might cause the ETM to generate an Address packet with the wrong PC
2277273	New	Programmer	Category B	Cache maintenance registers and ERRDEVID are not accessible from an unprivileged debugger
2132823	New	Programmer	Category C	TPIU does not generate frame synchronization packets when TPIU_PSCR is programmed with a value greater than 16
2161015	New	Programmer	Category C	When tracing an exception the ETM may incorrectly indicate a serious fault is pending
2219778	New	Programmer	Category C	Conditional instruction tracing might not work due to an interrupted VSTM, when ETM is configured to trace condition codes for data instructions only
2226595	New	Programmer	Category C	The DWT might generate a Data trace PC value packet with the incorrect PC, when ITM stall is enabled
2252094	New	Programmer	Category C	PMU cycle counter and PMU event counter incorrectly stop counting when there is an overflow
2267938	New	Programmer	Category C	The DWT might not trace an instruction when ITM stall is enabled
2285085	New	Programmer	Category C	DWT Performance Profiling Counters might sometimes have an incorrect value
2295637	New	Programmer	Category C	DWT_FUNCTION.MATCHED of the DWT Comparator gets cleared on a Speculative read that gets cancelled
2312624	New	Programmer	Category C	Trace reconstruction might be incorrect, due to invalid exception return address on a Lockup exception

12-Jul-2021: Changes in document version v7.0

ID	Status	Area	Category	Summary
2184678	New	Programmer	Category B	Unprivileged debugger access to some SCB registers incorrectly modifies register state
2217780	New	Programmer	Category B	A partially completed VLLDM might leave Secure floating-point, or vector data unprotected
2173589	New	Programmer	Category C	SBIST AHB interconnect can forward the incorrect value for HSEL
2176014	New	Programmer	Category C	A hardware breakpoint might be ignored after a Secure function call with unaligned SG

23-Apr-2021: Changes in document version v6.0

ID	Status	Area	Category	Summary
2019023	New	Programmer	Category C	The DWT might incorrectly expose the IPSR of a Secure exception when the ITM stall is enabled and Secure non-invasive debug is prohibited
2080562	New	Programmer	Category C	A Secure unprivileged debugger write to DSCSR.CDS may be ignored
2085795	New	Programmer	Category C	The DWT might incorrectly generate a match on the address or data of a memory transaction which is not performed
2117423	New	Programmer	Category C	A WFx Instruction may not be traced by the DWT when ITM stall is enabled
2129008	New	Programmer	Category C	A Secure read of ICSR_NS.VECTPENDING might return the Secure value
2134265	New	Programmer	Category C	A faulting vector load/store instruction might not be traced by the DWT when ITM stall is enabled

23-Oct-2020: Changes in document version v5.0

ID	Status	Area	Category	Summary
1960594	New	Programmer	Category C	Performance Monitor Unit is not disabled on warm reset

31-Jul-2020: Changes in document version v4.0

No new or updated errata in this document version.

03-Jul-2020: Changes in document version v3.0

ID	Status	Area	Category	Summary
1784253	New	Programmer	Category C	Invalid DWT trace on MVE load or store with predication
1784383	New	Programmer	Category C	Execution priority might be incorrect for one cycle for a Debug Monitor event after AIRCR, ITNS, IPR or SHPRn is changed
1794743	New	Programmer	Category C	When the DWT stalls the processor an instruction or data might be traced as part of the following instruction
1803238	New	Programmer	Category C	Some PMU events might be inaccurate
1810002	New	Programmer	Category C	An instruction or exception which causes lockup might not enter debug state when expected
1849620	New	Programmer	Category C	Trace might be incorrect when Unprivileged Debug is enabled
1858750	New	Programmer	Category C	A branch that is taken after an interstating branch that has not been taken might incorrectly fetch as if it is in Non-secure state
1871826	New	Programmer	Category C	The DWT might incorrectly expose the address of a BXNS

03-Apr-2020: Changes in document version v2.0

ID	Status	Area	Category	Summary
1735179	New	Programmer	Category A	Incorrect result written to the scalar register file for an MVE VMOV (general-purpose register to vector lane) instruction
1707936	New	Programmer	Category B	Unprivileged DAP accesses can modify the debug monitor exception enable
1717323	New	Programmer	Category B	Non-cacheable loads might not observe previous Non-cacheable stores
1720713	New	Programmer	Category B	Pending interrupts might get lost or corrupted when entering sleep using EWIC automatic sequence
1721673	New	Programmer	Category B	Deadlock due to Write-through Write-allocate store to location that always gets a bus error
1729416	New	Programmer	Category B	Automatic cache invalidation does not occur on Warm reset
1732413	New	Programmer	Category B	A VLLDM might restore a lazy context when flushed by an asynchronous halt request
1698164	New	Programmer	Category C	Processor might not step correctly after a branch which changes security state
1709730	New	Programmer	Category C	ETM may trace some interrupted instructions incorrectly
1733006	New	Programmer	Category C	ITM stall might not guarantee delivery of hardware trace packets
1736567	New	Programmer	Category C	An INVPC exception on FunctionReturn can target the wrong Security state
1740040	New	Programmer	Category C	The LR can be corrupted when the last instruction of a loop is overlapped and interrupted.
1745058	New	Programmer	Category C	Processor might step two consecutive exception entries when debug halt stepping
1746023	New	Programmer	Category C	PMU event counting and processor halting might not work as expected

20-Dec-2019: 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
1735179	Programmer	Category A	Incorrect result written to the scalar register file for an MVE VMOV (general-purpose register to vector lane) instruction	r0p0	r0p1
1717323	Programmer	Category B	Non-cacheable loads might not observe previous Non-cacheable stores	r0p0	r0p1
1720713	Programmer	Category B	Pending interrupts might get lost or corrupted when entering sleep using EWIC automatic sequence	r0p0	r0p1
1732413	Programmer	Category B	A VLLDM might restore a lazy context when flushed by an asynchronous halt request	r0p0	r0p1
1729416	Programmer	Category B	Automatic cache invalidation does not occur on Warm reset	r0p0	r0p1
1721673	Programmer	Category B	Deadlock due to Write-through Write-allocate store to location that always gets a bus error	r0p0	r0p1
1707936	Programmer	Category B	Unprivileged DAP accesses can modify the debug monitor exception enable	r0p0	r0p1
2277273	Programmer	Category B	Cache maintenance registers and ERRDEVID are not accessible from an unprivileged debugger	r0p0, r0p1, r0p2, r1p0	r1p1
2428419	Programmer	Category B	Automatic EWIC drain will not function if the processor is powered down during sleep when the secure extension is not configured	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2233762	Programmer	Category B	An interrupted vector instruction with implicit LE might cause the ETM to generate an Address packet with the wrong PC	r0p0, r0p1, r0p2, r1p0	r1p1
2184678	Programmer	Category B	Unprivileged debugger access to some SCB registers incorrectly modifies register state	r0p0, r0p1, r0p2, r1p0	r1p1
2217780	Programmer	Category B	A partially completed VLLDM might leave Secure floating-point, or vector data unprotected	r0p0, r0p1, r0p2, r1p0	r1p1

ID	Area	Category	Summary	Found in versions	Fixed in version
1794743	Programmer	Category C	When the DWT stalls the processor an instruction or data might be traced as part of the following instruction	r0p1	r0p2
1784383	Programmer	Category C	Execution priority might be incorrect for one cycle for a Debug Monitor event after AIRCR, ITNS, IPR or SHPRn is changed	r0p0, r0p1	r0p2
1849620	Programmer	Category C	Trace might be incorrect when Unprivileged Debug is enabled	r0p0, r0p1	r0p2
1746023	Programmer	Category C	PMU event counting and processor halting might not work as expected	r0p0	r0p1
1733006	Programmer	Category C	ITM stall might not guarantee delivery of hardware trace packets	r0p0	r0p1
1740040	Programmer	Category C	The LR can be corrupted when the last instruction of a loop is overlapped and interrupted.	r0p0	r0p1
1736567	Programmer	Category C	An INVPC exception on FunctionReturn can target the wrong Security state	r0p0	r0p1
1803238	Programmer	Category C	Some PMU events might be inaccurate	r0p0, r0p1	r0p2
1810002	Programmer	Category C	An instruction or exception which causes lockup might not enter debug state when expected	r0p0, r0p1	r0p2
1745058	Programmer	Category C	Processor might step two consecutive exception entries when debug halt stepping	r0p0	r0p1
1858750	Programmer	Category C	A branch that is taken after an interstating branch that has not been taken might incorrectly fetch as if it is in Non-secure state	r0p0, r0p1	r0p2
1784253	Programmer	Category C	Invalid DWT trace on MVE load or store with predication	r0p0	r0p1
2703268	Programmer	Category C	L1D_CACHE_REFILL and L1D_CACHE_MISS_RD PMU events might be inaccurate	r0p0, r0p1, r0p2, r1p0, r1p1	Open
1698164	Programmer	Category C	Processor might not step correctly after a branch which changes security state	r0p0	r0p1
1871826	Programmer	Category C	The DWT might incorrectly expose the address of a BXNS	r0p0, r0p1	r0p2
1709730	Programmer	Category C	ETM may trace some interrupted instructions incorrectly	r0p0	r0p1

ID	Area	Category	Summary	Found in versions	Fixed in version
2619476	Programmer	Category C	TCM ECC errors for S-AHB accesses will not be reported in sleep state	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2567270	Programmer	Category C	A BX using the LR where the t-bit (bit[0]) of the LR is not set, under certain circumstances, might not cause a INVSTATE fault	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2653102	Programmer	Category C	Unprivileged debug transactions to the STIR and EVENTSPR registers can pend interrupts	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2675831	Programmer	Category C	After deactivating the instruction cache self-modified code might not be executed correctly	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2312624	Programmer	Category C	Trace reconstruction might be incorrect, due to invalid exception return address on a Lockup exception	r0p0, r0p1, r0p2, r1p0	r1p1
2295637	Programmer	Category C	DWT_FUNCTION.MATCHED of the DWT Comparator gets cleared on a Speculative read that gets cancelled	r0p0, r0p1, r0p2, r1p0	r1p1
2285085	Programmer	Category C	DWT Performance Profiling Counters might sometimes have an incorrect value	r0p0, r0p1, r0p2, r1p0	r1p1
2397983	Programmer	Category C	PMU event L1I_CACHE is incorrectly implemented in the instruction cache	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2019023	Programmer	Category C	The DWT might incorrectly expose the IPSR of a Secure exception when the ITM stall is enabled and Secure non-invasive debug is prohibited	r0p1, r0p2	r1p0
2461408	Programmer	Category C	Unprivileged debugger access to DPDLPSTATE register incorrectly modifies register state	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2454643	Programmer	Category C	A VSTMDB which specifies unimplemented registers may write to unexpected memory locations	r0p0, r0p1, r0p2, r1p0, r1p1	Open
2161015	Programmer	Category C	When tracing an exception the ETM may incorrectly indicate a serious fault is pending	r0p0, r0p1, r0p2, r1p0	r1p1
2129008	Programmer	Category C	A Secure read of ICSR_NS.VECTPENDING might return the Secure value	r0p0, r0p1, r0p2	r1p0
2267938	Programmer	Category C	The DWT might not trace an instruction when ITM stall is enabled	r0p1, r0p2, r1p0	r1p1

ID	Area	Category	Summary	Found in versions	Fixed in version
2226595	Programmer	Category C	The DWT might generate a Data trace PC value packet with the incorrect PC, when ITM stall is enabled	r0p1, r0p2, r1p0	r1p1
1960594	Programmer	Category C	Performance Monitor Unit is not disabled on warm reset	r0p0, r0p1, r0p2	r1p0
2252094	Programmer	Category C	PMU cycle counter and PMU event counter incorrectly stop counting when there is an overflow	r0p0, r0p1, r0p2, r1p0	r1p1
2173589	Programmer	Category C	SBIST AHB interconnect can forward the incorrect value for HSEL	r1p0	r1p1
2219778	Programmer	Category C	Conditional instruction tracing might not work due to an interrupted VSTM, when ETM is configured to trace condition codes for data instructions only	r0p0, r0p1, r0p2, r1p0	r1p1
2176014	Programmer	Category C	A hardware breakpoint might be ignored after a Secure function call with unaligned SG	r0p0, r0p1, r0p2, r1p0	r1p1
2132823	Programmer	Category C	TPIU does not generate frame synchronization packets when TPIU_PSCR is programmed with a value greater than 16	r0p0, r0p1, r0p2, r1p0	r1p1
2134265	Programmer	Category C	A faulting vector load/store instruction might not be traced by the DWT when ITM stall is enabled	r0p1, r0p2	r1p0
2117423	Programmer	Category C	A WFx Instruction may not be traced by the DWT when ITM stall is enabled	r0p1, r0p2	r1p0
2085795	Programmer	Category C	The DWT might incorrectly generate a match on the address or data of a memory transaction which is not performed	r0p0, r0p1, r0p2	r1p0
2080562	Programmer	Category C	A Secure unprivileged debugger write to DSCSR.CDS may be ignored	r0p0, r0p1, r0p2	r1p0
2791503	Programmer	Category C	ITM can generate Synchronization and timestamp packets when NIDEN=0	r0p0, r0p1, r0p2, r1p1	Open

Errata descriptions

Category A

1735179
Incorrect result written to the scalar register file for an MVE VMOV (general-purpose register to vector lane) instruction

Status

Affects: Cortex-M55
Fault Type: Programmer Category A
Fault Status: Present in r0p0. Fixed in r0p1

Description

The processor can produce an incorrect result for a VMOV (general-purpose register to vector lane) instruction. This can only occur when a specific sequence of instructions is executed.

Configurations affected

This erratum affects all configurations of the processor configured with the M-profile Vector Extension.

Conditions

One of the following instructions is executed by the processor:

```
VMOV<c>.<dt> Rx, Qx[idx]
VMOV<c> Ry, Rx, Qd[idx], Qd[idx2]
```

Followed by *one* of the following instructions writing Rn:

```
VSHLC<v>.<dt> Qd, Rn, #<imm>
VIDUP<v>.<dt> Qd, Rn, #<imm>
VDDUP<v>.<dt> Qd, Rn, #<imm>
VIWDUP<v>.<dt> Qd, Rn, Rm, #<imm>
VDWDUP<v>.<dt> Qd, Rn, Rm, #<imm>
```

Followed by:

```
VMOV<c>.<dt> Qy[idx], Rx
```

Where:

- Rx != Rn or Rm

- Ry != Rn or Rm
- Rx is written on beat 2 or 3
- The sequence is not in an IT block
- No faults or exceptions occur in the sequence

Implications

The Qy[idx] destination register of a VMOV (general-purpose register to vector lane) instruction will be written with an incorrect result. The result is generated by the processor incorrectly using the Rn or the top 32-bits of the Qd result of the VSHLC, VIDUP, VIWDUP, VDDUP or VDWDUP instruction as the source operand for the VMOV (general-purpose register to vector lane).

Workaround

If using the r0p0 version of the processor, set ACTLR.DISOLAP = 1. If using a subsequent version of the processor, which have resolved this erratum, do not set ACTLR.DISOLAP = 1 as it has significant performance implications.

Category A (rare)

There are no errata in this category.

Category B

1717323
Non-cacheable loads might not observe previous Non-cacheable stores

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0. Fixed in r0p1.

Description

This erratum might cause a Non-cacheable load to not observe a previous Non-cacheable store if the store location has already been allocated into the cache. This can occur only if this location was previously treated as Cacheable.

Configurations Affected

All configurations with a data cache and *Memory Protection Unit* (MPU) are affected.

Conditions

The following sequence of operations are required to hit this erratum:

1. A line must be allocated to the cache for one of the following reasons:
   - An explicit load or store to that cacheline
   - The prefetcher has triggered a linefill
2. A Shareable or Non-cacheable store to the cacheline allocated in step 1
3. A Shareable or Non-cacheable load to the same address as the store

The line can only be allocated to the cache if the memory is marked as Non-shareable and Cacheable. For the same cacheline to be Non-cacheable later, one of the following must occur:

1. The MPU is reprogrammed
2. One access is from Non-secure, one access is from Secure, and the MPUs for each Security level have different attributes for that address
3. The processor core must switch between executing at a negative priority level and a positive priority level, and MPU_CTRL.HFNMIENA == 0b0

The cache must be active (MSCR.DCACTIVE == 0b1) and enabled (CCR.DC == 0b1) for all instructions.

There must be no data cache invalidate or clean-and-invalidate operations to the affected cacheline between the point the line is allocated into the cache and the Non-cacheable store.

This erratum can occur even if there are many other instructions in the sequence required to hit this erratum.

Implications

A load can return incorrect data.

Workaround

If software does an explicit load or store access to a Non-shareable Cacheable cacheline, it must do a data cache clean-and-invalidate operation to that cacheline before it accesses that cacheline with any Shareable or Non-cacheable accesses.

The prefetcher must be disabled (by setting PFCR.EN = 0b0) if any of the following conditions are met:

- The MPU programming is ever changed such that a currently Non-shareable Cacheable address becomes either Shareable or Non-cacheable
- The processor core enters a HardFault or NMI handler and MPU_CTRL.HFNMIENA = 0b0
- There is an address that is accessible from both Secure and Non-secure regions, and the Secure and Non-secure MPUs do not use a consistent set of atributes for that address

1720713

Pending interrupts might get lost or corrupted when entering sleep using EWIC automatic sequence

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0. Fixed in r0p1.

Description

If the processor wants to enter a low-power state where the clock or power has been removed, it must rely on another component to control when it wakes up. The processor has an *Internal Wakeup Interrupt Controller* (IWIC) and *External Wakeup Interrupt Controller* (EWIC) to control processor wakeup. The processor can be configured to include:

- Either the IWIC or EWIC
- Both the IWIC and EWIC
- Neither the IWIC nor EWIC

Before entering sleep, the processor can be configured to automatically copy any currently pending interrupts into the EWIC, so that pending interrupt information can be restored on exiting sleep. This erratum causes the information about pending interrupts to be corrupted by the automatic sequence.

Configurations Affected

All configurations with an EWIC are effected.

Conditions

The following conditions are required to be TRUE to encounter this erratum:

- The processor must be configured to use WIC sleep (WICCONTROL[0] = 0b1)
- The processor must be configured to use the EWIC (WICCONTROL[1] = 0b0)
- The automatic powerdown sequence must be used (WICCONTROL[3] = 0b1)
- The processor must be using deep sleep (SCR.SLEEPDEEP == 0b1)
- The processor must have interrupts pending

The processor must then enter sleep through a WFE or WFI instruction.

SLEEPONEXIT is not affected.

Implications

Depending on the priority number of each pending interrupt, a different interrupt is pended instead, or the pending interrupt information might be lost.

- **IRQ[31:0]** - These interrupts are lost
- **IRQ[N:32], where N>=32** - These interrupts are pended

If an incorrect pending interrupt causes the processor to wakeup, the processor immediately wakes up on entry to sleep.

Workaround

The automatic sequence must be disabled by setting EWIC_ASCR.ASPD = 0b0.

Any pending interrupts, and the wakeup events, must be copied into the EWIC by software before entering sleep. This requires the following sequence:

1. Enable the EWIC by setting EWIC_CR.EN = 0b1
2. Copy NVIC_ISPR0 into EWIC_PEND0
3. Repeat step 2 for each NVIC_ISPR{N} that might contain pending interrupts that need to be saved
4. Copy EVENTMASKA into EWIC_MASKA
5. Copy EVENTMASK0 into EWIC_MASK0
6. Repeat step 5 for each EVENTMASK{N} register that might contain events that should cause the processor to wakeup
7. The processor can now safely enter sleep

1732413
A VLLDM might restore a lazy context when flushed by an asynchronous halt request

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0. Fixed in r0p1

Description

It has been found that a VLLDM which restores a lazy context might do so even when flushed by an asynchronous request to enter halt. The VLLDM instruction re-executes when leaving debug state causing the floating-point state to be corrupted.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the Armv8-M Security Extension and the Floating-point Extension or *M-profile Vector Extension* (MVE).

Conditions

The erratum can occur if:

- There is an outstanding lazy Floating-point or MVE context (FPCCR.LSPACT==1 && CONTROL.FPCA==0). A VLLDM instruction is executed in order to restore this context
- An asynchronous halt request occurs while executing the VLLDM causing it to be flushed and debug state entered.
- The VLLDM might have restored the context without retiring (FPCCR.LSPACT==0 and CONTROL.FPCA==1)

Implications

The VLLDM instruction is flushed because of the halt request but still restores the lazy context. When leaving debug state the VLLDM instruction is re-executed except, this time FPCCR.LSPACT==0. Therefore, an attempt is made to restore *UNKNOWN* state from memory. This corrupts the floating-point context trying to be restored.

The VLLDM instruction can only be executed in Secure state. Therefore, there are no security implications.

Lazy stacking is an optimization and not guaranteed Therefore, the base address used must be pointing to a Secure area of memory preventing Non-secure memory from modifying the values which are restored.

Workaround

If Secure debug is enabled on the device, you can avoid this erratum by disabling lazy floating-point preservation. That is, by setting FPCCR.LSPEN to 0.

Additional information

There is no way for the debugger to detect that this issue has occurred when in debug state. This means software must handle this issue and the debugger is responsible for correcting it.

1729416

Automatic cache invalidation does not occur on Warm reset

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0, Fixed in r0p1

Description

The Cortex-M55 processor provides the ability to automatically invalidate the cache in certain situations. One of these cases is when a Warm reset occurs when the processor is in the WARM_RST power mode (selected using the external P-channel interface), and the caches are enabled. In this case, the automatic invalidation does not occur because of this erratum.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with an instruction cache or data cache.

Conditions

- Automatic cache invalidation is enabled by setting the external input signal INITL1RSTDIS to 0.
- The processor enters the warm reset power mode because of a request on COREPREQ with COREPSTATE set to WARM_RST.
- The system asserts the warm reset signal (nSYSRESET) to perform a functional reset.
- The cache is not invalidated as expected before execution is started.

Implications

The cache is not invalidated automatically.

Workaround

One of the following workarounds can be used:

1. Software can invalidate the caches manually using the cache maintenance registers.
2. Assert the reset when in WFI/WFE sleep rather than the specific P-Channel Warm reset power mode

1721673

Deadlock due to Write-through Write-allocate store to location that always gets a bus error

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0. Fixed in r0p1

Description

If a cacheline always gets a bus error on *Master AXI (_M-AXI) reads, any Non-shareable Write-Through stores with the Write-Allocate hint to the M-AXI are unable to leave the _Store Buffer* (STB). This causes some of the subsequent instructions to deadlock the processor core.

Configurations Affected

All configurations with a data cache and *Memory Protection Unit* (MPU) are affected.

Conditions

The data cache must be present, active and enabled. The MPU must be present and enabled.

There must be a Non-shareable Write-Through store to M-AXI with the Write-Allocate hint. This store must not get a MemManage fault or Secure Fault. The cacheline of this store must always get a bus error on M-AXI reads.

Implications

Any subsequent DSB, DMB or ESB instruction deadlocks the processor.

Any subsequent stores that need to write to M-AXI fill store buffer slots. These stores are also unable to leave causing the processor to deadlock.

When the processor is in deadlock state, it is not be able to enter halt state or service debugger accesses to memory.

Workaround

Software must not use the Write-Allocate hint for any Write-Through memory.

1707936
Unprivileged DAP accesses can modify the debug monitor exception enable

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0. Fixed in r0p1.

Description

The *Unprivileged Debug Extension* (UDE) provides support for unprivileged DAP requests to write to certain registers or fields. This erratum causes the debug monitor enable bits to be incorrectly written to by an unprivileged DAP request.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DAP interface.

Conditions

- Unprivileged DAP requests are allowed (DAUTHCTRL.UIDAPEN==1)
- The DAP makes an unprivileged request to DEMCR.MON_EN or DEMCR.UMON_EN.

Implications

The write occurs and the debug monitor enable is modified. This allows an unprivileged DAP request to disable or enable the DebugMonitor exception.

There is no leak of state as this bit only allows the debug monitor exception to either be taken or not taken. The exception follows all the normal rules to protect privileged or secure information.

Workaround

Software should not set DAUTHCTRL.UIDAPEN to 1 if both of the following are true:

- It does not trust an external debugger to not modify DEMCR in this way
- It does not manage spurious DebugMonitor exceptions or relies on correct generation of Debug Monitor exceptions for functionally correct behaviour

2277273
Cache maintenance registers and ERRDEVID are not accessible from an unprivileged debugger

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The processor includes support for accessing internal registers from an unprivileged debugger. Due to this erratum, the data and instruction cache maintenance registers and the Error Record Device ID Register, ERRDEVID are not accessible and will instead return a fault.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor with Halting debug, DBGLVL > 0.

Conditions

This erratum occurs when all the following conditions are met:

Either DAUTHCTRL_S.UIDAPEN or DAUTHCTRL_NS.UIDAPEN is set.
A write access is made to any of the following cache maintenance registers:

- DCCIMVAC at 0xE000EF70
- DCCISW at 0xE000EF74
- DCCMVAC at 0xE000EF68
- DCCMVAU at 0xE000EF64
- DCCSW at 0xE000EF6C
- ICIALLU at 0xE000EF50
- ICIMVAU at 0xE000EF58

Or a read access is made to the following register:

- ERRDEVID at 0xE0005FC8

Implications

If a debugger inserts an instruction such as a BKPT operation into memory, cache maintenance operations are required to guarantee coherency between the data cache and the instruction cache, such that the new instruction is executed rather than a 'stale' instruction which was previously fetched into the instruction cache. Due to this erratum the cache maintenance operations cannot be applied if the debugger is unprivileged.

The ERRDEVID register cannot be read by an unprivileged debugger to determine the number of Error records supported by the Armv8.1-M RAS Extension.
Note: This register is always 1 for Cortex-M55 when ECC is enabled

Workaround

There is no workaround.

2428419
Automatic EWIC drain will not function if the processor is powered down during sleep when the secure extension is not configured

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

When the Cortex-M55 processor core is in deep sleep mode, the processor can be put in a low-power state where the *Nested Vectored Interrupt Controller* (NVIC) does not control wake up. Instead, either the *Internal Wakeup Interrupt Controller* (IWIC) or the *External Wakeup Interrupt Controller* (EWIC) can be used for capturing pending interrupts and detecting wakeup conditions, and this is called WIC sleep.

When entering WIC sleep, the processor can be configured to automatically transfer any external interrupts that are already pended in the NVIC to the EWIC. This transfer prevents interrupts at a lower priority than the current thread being lost if the processor is powered down. Any interrupts that are pended before or during WIC sleep can be transferred back to the NVIC from the EWIC when a wakeup event occurs.

The Cortex-M55 processor can be configured to support one of two low-power implementation models when using the EWIC, either:

1. retention (processor state save and restore using SW is not required), or
2. power down (processor state save and restore using SW is required)

Due to this erratum, the processor will not automatically transfer the pending interrupt and other information from the EWIC to the NVIC when a wakeup event occurs if the processor Security Extension is not present and the power down low-power implementation model is used.

Please note, automatic transfer of the pending interrupt information will occur from the EWIC to the NVIC when a wakeup event occurs if the processor Security Extension is not present and the retention low-power implementation model is used. This is because the processor will not be reset during the wakeup process.

Configurations affected

This erratum affects configurations of the Cortex-M55 processor where the Security Extension is not present and the EWIC is present. Verilog parameters SECEXT == 0 and EWIC > 0.

Conditions

The processor **WICCONTROL[3:0]** input is set to 0bx101 which sets it to automatically transfer the pending interrupt information from the EWIC to the NVIC on wake-up from WIC sleep.

Implications

When the processor is powered down in WIC sleep and the security extension is not present, pending interrupt information will not be automatically restored to the processor on wakeup. Hence, pending interrupts will not be taken after they are enabled by software as part of the wakeup sequence.

An RTL fix is not planned because the software workaround fully resolves this erratum. The software transparent automatic pending interrupt information EWIC transfer feature when the processor is powered down in WIC sleep, is considered deprecated in Cortex-M55 and will not be present in future processors.

Workaround

This erratum can be worked around by transferring the state information from the EWIC to the NVIC using software. This can be included as part of the processor state restoration software run at wakeup when the processor power down in WIC sleep low-power model is implemented.

The processor can carry out the required state transfer using software to write the processor state via the EPPB and IPPB interfaces:

- Read EWIC_PENDA register and write the result to EVENTSPR register
- Read EWIC_PENDn registers and write the result to NVIC_ISPRn registers for all valid interrupts supported by the EWIC
- Write to EWIC_CR to clear the EN field and disable the EWIC

See the *Cortex-M55 Implementation and Integration Manual* for further details about the automatic sleep entry and wake-up EWIC/NVIC transfer sequences.

2233762
An interrupted vector instruction with implicit LE might cause the ETM to generate an Address packet with the wrong PC

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

This erratum can cause the *Embedded Trace Macrocell* (ETM) to generate an Exception packet with the wrong preferred exception return address, when a vector instruction executing just before an implicit LE, gets interrupted. This can cause a trace analyzer to reconstruct the stream of instructions incorrectly.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with an ETM and the *M-profile Vector Extension* (MVE) extension implemented.

Conditions

This erratum occurs when the following sequence of conditions is met:

1. An MVE vector instruction, which is subject to beat-wise execution is executed
2. An implicit LE, which is not in the last iteration of a low-overhead-loop, is executed with the MVE instruction
3. The MVE instruction completes at least one beat
4. An asynchronous interrupt or fault occurs before the MVE instruction completes

Implications

The ETM might generate an Exception packet with the incorrect preferred exception return address. Specifically, the preferred exception return address traced might be the address of the vector instruction, instead than the target address of the implicit LE. This might cause a trace analyzer to reconstruct the stream of instructions incorrectly.

If the loop has no branches (other than the LE instruction), the trace will imply an extra iteration of the loop, and this is not trivially detectable. If the loop has at least one branch before the MVE instruction, then a trace analyzer will encounter the branch instruction when analyzing the Exception element. This branch instruction would be unexpected and the trace analyzer might be able to detect this erratum has occurred.

Workaround

There is no workaround for this erratum.

2184678
Unprivileged debugger access to some SCB registers incorrectly modifies register state

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The processor contains architectural and **`IMPLEMENTATION DEFINED`** registers in the *System Control Block* (SCB) that are only accessible by privileged code or a privileged debugger. Due to this erratum, some of the registers can be written by an unprivileged debugger.

Writing to the following registers from an unprivileged debugger will return a fault, but the register state will be incorrectly updated:

- MSCR, Memory System Control Register
- PFCR, Prefetcher Control Register
- ITCMCR, ITCM Control register
- DTCMCR, DTCM Control register
- PAHBCR, P-AHB Control Register

Writing to the following registers from an unprivileged debugger will incorrectly update the state without returning a fault:

- ACTLR, Auxiliary Control Register
- CCR, Configuration and Control Register

The following read-only register can be accessed by an unprivileged debugger when it should fault:

- ERRIIDR, Error Implementer ID Register

Configurations affected

This erratum affects all configurations of the processor configured with Halting debug, DBGLVL > 0.

Conditions

This erratum occurs when Unprivileged Invasive DAP Access Enable, DAUTHCTRL.UIDAPEN = 1 (either bank) and:

For the erratum part concerning MSCR, PFCR, ITCMCR, DTCMCR and PAHBCR to occur, the corresponding lock signal or configuration allows software to update these registers, that is:

- For ITCMCR and DTCMCR, the external input signal **LOCKTCM** is not asserted
- For PAHBCR, the external input signal **LOCKPAHB** is not asserted
- For PFCR, Data Cache is configured in the processor

and one of the following conditions applies:

- Security Extensions are included (SECEXT != 0) and there is a Secure write access through unprivileged DAP request to one of the following IMPDEF SCB registers:
- BusFault and NMI are Non-secure and exceptions can target Non-secure HardFault (AIRCR.BFHFNMINS = 1) and there is a write access through unprivileged DAP request to one of the following IMPDEF SCB registers:

```
0xE001E000, MSCR
0xE001E004, PFCR
0xE001E010, ITCMCR
0xE001E014, DTCMCR
0xE001E018, PAHBCR
```

For the erratum part concerning ACTLR and CCR to occur, the following condition applies:

- There is a write access through an unprivileged DAP request to either of the following two registers:

```
0xE000E008, ACTLR
0xE000ED14, CCR
```

For the erratum part concerning ERRIIDR to occur, the following condition applies:

- There is a read access through an unprivileged DAP request to the following register:

```
0xE0005E10, ERRIIDR
```

Implications

If this erratum occurs, an unprivileged debugger write will change the state of these registers which could impact the processor configuration by enabling or disabling features that are configurable by these registers. Furthermore, the unprivileged debugger will not report fault when the access is to the CCR register.

Workaround

There is no workaround for this erratum.

2217780
A partially completed VLLDM might leave Secure floating-point, or vector data unprotected

Status

Affects: Cortex-M55
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The VLLDM instruction allows Secure software to restore a floating-point or vector context from memory. Due to this erratum, if this instruction is interrupted or it faults before it completes then Secure data might be left unprotected in the floating-point and vector register file, including the VPR and FPSCR.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the Armv8-M Security Extension and either the Floating-point Extension or *M-profile Vector Extension* (MVE).

Conditions

This erratum occurs when all the following conditions are met:

- There is no active floating-point context, (CONTROL.FPCA==0)
- Secure lazy floating-point state preservation is not active, (FPCCR_S.LSPACT==0)
- The floating-point and vector registers are treated as Secure (FPCCR_S.TS==1)
- Secure floating-point state needs to be restored, (CONTROL_S.SFPA == 1)
- Non-secure state is permitted to access to the floating-point registers, (NSACR.CP10 == 1)
- A VLLDM instruction has loaded at least one register from memory and does not complete due to an interrupt or fault

Implications

If the floating-point or vector registers contain Secure data, a VLSTM instruction is usually executed before calling a Non-secure function to protect the Secure data. This might cause the data to be transferred to memory (either directly by the VLSTM or indirectly by the triggering of a subsequent lazy state preservation operation). If the data has been transferred to memory, it is restored using VLLDM on return to Secure state.

If the VLLDM is interrupted or it faults before it completes and enters a Non-secure handler, the partial register state which has been loaded will be accessible to Non-secure state.

Workaround

To avoid this erratum, software can ensure a floating-point context is active before executing the VLLDM instruction by performing the following sequence:

1. Read CONTROL_S.SFPA
2. If CONTROL_S.SFPA==1 then execute an instruction which has no functional effect apart from causing context creation (such as `VMOV S0, S0`)

Category B (rare)

There are no errata in this category.

Category C

1794743
When the DWT stalls the processor an instruction or data might be traced as part of the following instruction

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p1. Fixed in r0p2.

Description

The *Data Watchpoint and Trace Unit* (DWT) provides the ability to stall the processor. When doing so, it is possible that that an instruction, associated data transaction, or both might be traced with the instruction address of the following instruction.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the DWT and *Instrumentation Trace Macrocell* (ITM).

Conditions

DWT trace is enabled, ITM_TCR.STALLENA is set to 1, and the following scenario occurs:

1. An instruction is executed or a data transaction is performed, while there is a stall request from the DWT.
2. A dual-issue pair is then executed, while the DWT stall request is still asserted.
3. An asynchronous interrupt or debug halt request occurs before the dual-issue pair completes.

Implications

For the scenario described above, the DWT might trace the instruction and its data with the instruction address of the following instruction.

Workaround

There is no workaround for this erratum.

1784383
Execution priority might be incorrect for one cycle for a Debug Monitor event after AIRCR, ITNS, IPR or SHPRn is changed

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The AIRCR, ITNS, IPR, SHPR0, SHPR1 and SHPR2 registers are used in the *Nested Vectored Interrupt Controller* (NVIC) active tree to calculate the execution priority, which is used to determine whether the debug monitor events causes an exception. As a result of this erratum, when there are high latency interrupts (with corresponding IRQTIER value equal to 1) active, there might be a cycle delay in getting correct active tree output after any of the AIRCR, ITNS, IPR, SHPR0, SHPR1 and SHPR2 registers is updated. This results in the NVIC selecting the wrong execution priority relative to Debug Monitor events for one cycle that is neither based on the old register values nor the new register values.

Configuration affected

This erratum affects all configurations including high latency interrupts, (Verilog parameter IRQTIER != 0).

Conditions

- At least one high latency IRQ is active.
- Debug Monitor is enabled and Halting debug is disabled for the security state
- A write is carried out to one of the following registers, changing the value : AIRCR, ITNS, IPR, SHPR0, SHPR1 or SHPR2

Implications

Instruction stepping, asynchronous debug events, and breakpoints might be incorrectly triggered or missed in the cycle after the write to AIRCR, ITNS, IPR SHPR0, SHPR1, and SHPR2.
This erratum is unlikely to affect a realistic system because the registers associated with exception priority are not expected to be programmed dynamically when interrupts are active.

Workaround

There is no workaround for this erratum.

1849620
Trace might be incorrect when Unprivileged debug is enabled

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

Unprivileged debug does not trace as expected in some scenarios. This can mean that the *Data Watchpoint and Trace* (DWT) or *Instrumentation Trace Macrocell* (ITM) might trace Non-secure information unexpectedly or that the *Embedded Trace Macrocell* (ETM) might not trace some instructions on switching to Secure state.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the Armv8-M Security Extension and Halting Debug.

Conditions

Trace might occur incorrectly in the following scenarios:

- Non-secure information is traced unexpectedly. This occurs in the following cases:
  - Unprivileged debug is only allowed in Secure state, that is, Non-secure state is privileged. The processor is in Non-secure state and writes to ITM_STIM register.
  - Unprivileged debug is only allowed in Non-secure state. Processor is in Secure state and the *Data Watchpoint Trace* (DWT) cycle counter is enabled.
- Secure trace might be missed. This can occur in the following situations:
  - ETM trace enabled and Unprivileged debug is only allowed in Secure state.
  - Branch from Non-secure privileged to Secure unprivileged (Secure Gateway check).

Implications

- When Non-secure information is traced unexpectedly, the DWT or ITM traces Non-secure information when Unprivileged debug is not enabled for the current mode.
  - Note: Cycle counter matches which are programmed to generate debug events do so correctly. This issue only affects trace.
- If Secure trace information is missed, the ETM might miss tracing up to the first three instructions of the Secure Unprivileged function.

Workaround

There is no workaround.

1746023
PMU event counting and processor halting might not work as expected

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1.

Description

The *Performance Monitoring Unit* (PMU) allows various events to be counted. It can also be configured to generate a debug event when a counter overflows. This erratum causes:

- Some events to not be counted frequently enough.
- The PMU generates a debug halt event that does not halt the processor.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with PMU.

Conditions

The PMU is enabled and one of the following conditions occurs:

1. An instruction retired event occurs including but not restricted to BR_RETURN_RETIRED, MVE_LDST_RETIRED and MVE_LDST_MULTI_RETIRED.
2. A PMU counter overflows which should cause a debug halt request.

Implications

For the scenarios above:

1. The counter does not increment for all events of the specified type.
2. The processor does not pend a halt request (setting DHCSR.C_HALT). Instead the overflow is ignored.

The events are not counted frequently enough and there is no leaking of Secure information to Non-secure.

Workaround

There is no workaround.

1733006
ITM stall might not guarantee delivery of hardware trace packets

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1.

Description

The Armv8-M architecture (in rule **RRGCV**) specifies the ability for the ITM to guarantee delivery of hardware trace packets. This erratum can cause some of the hardware trace packets being lost when ITM stalling is enabled.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DWT and ITM (DBGLVL>0).

Conditions

The following conditions cause this erratum to occur:

- ITM stall is enabled (ITM_TCR.STALLENA==1).
- The DWT is configured to generate data trace.

Implications

Some of the data trace packets might be lost.

Workaround

There is no workaround.

1740040
The LR can be corrupted when the last instruction of a loop is overlapped and interrupted

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1

Description

When interrupting a pair of instructions, the first of which is the last instruction of a low overhead loop, the LR might hold the value of the next iteration and not the current one.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the *M-profile Vector Extension* (MVE).

Conditions

This erratum can occur if:

- The last instruction of a low overhead loop is overlapped with a vector load/store whose base is the LR and has base writeback.
- The overlapped pair is interrupted
- The LR may hold the value of the next iteration rather than the current

Implications

Upon returning from the exception, the last instruction in the loop will see the decremented LR value. In addition the number of iterations will be reduced by 1.

Workaround

There is no workaround for this erratum.
The code sequence which causes this erratum has no useful purpose and is not expected to be found in regular software.

1736567
An INVPC exception on FunctionReturn can target the wrong Security state

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1

Description

When there is an outstanding write to the PPB memory region, it is possible for an INVPC UsageFault exception during FunctionReturn() to target the wrong Security state.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the Armv8-M Security Extension.

Conditions

This erratum can occur if:

- A PPB write occurs which is stalled.
- A load instruction with base writeback, loads the PC and triggers a FunctionReturn() operation
- An INVPC UsageFault is detected.
- If the PPB write is still outstanding the a Non-secure INVPC UsageFault might get pended instead of a Secure INVPC UsageFault

Implications

A Non-secure UsageFault handler is pended or taken instead of a Secure handler. On returning from the Non-secure UsageFault handler, the FunctionReturn() re-executes and pends a Secure handler. Secure state information is not revealed to Non-secure state.

Workaround

There is no workaround for this erratum.

1803238
Some PMU events might be inaccurate

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The following PMU events supported by the Cortex-M55 processor might trigger too frequently or infrequently:

- MVE_STALL_BREAK
- MVE_STALL_DEPENDENCY
- MVE_FP_SP_RETIRED
- MVE_PRED
- MEMORY_ERROR
- BR_RETIRED
- ST_RETIRED

Configurations affected

This erratum affects configurations of the Cortex-M55 processor with the Verilog parameter DBGLVL != 0.

Conditions

There are no special conditions associated with this erratum.

Implications

MVE_STALL_BREAK

MVE_STALL_BREAK is incorrectly triggered on stalls associated with multi-cycle scalar floating point instructions. This event can be considered unaffected unless executing code containing scalar floating-point instructions, in which case a slightly higher frequency events might be seen.

MVE_STALL_DEPENDENCY

MVE_STALL_DEPENDENCY is incorrectly triggered on stalls associated with scalar floating point or scalar long shift dependencies. This event can be considered unaffected unless executing code containing scalar floating-point or scalar long shift instructions, in which case a slightly higher frequency might be seen.

MVE_FP_SP_RETIRED

MVE_FP_SP_RETIRED is incorrectly triggered on *M-profile Vector Extension* (MVE0 VCVT instructions that convert to or from single-precision floating-point representation. This event can be considered unaffected unless executing code containing MVE VCVT that converts to or from single-precision floating-point representation, in which case a slightly higher frequency might be seen.

MVE_PRED

MVE_PRED is:

- Incorrectly triggered on MVE instructions that do not have predication applied
- Not triggered as expected on MVE instructions that do have predication applied.

This event is affected in all MVE code and therefore, you must avoid using it.

MEMORY_ERROR

MEMORY_ERROR is not triggered as expected on a data cache or instruction cache *Error Correcting Code* (ECC) error. This event can be considered unaffected unless using the MEMORY_ERROR for ECC errors from data cache or instruction cache, in which case, you must avoid using it.

BR_RETIRED

BR_RETIRED is not triggered as expected on branches which are not taken because of their condition code check. As conditional branches are regularly used in compiled code, you must avoid using this event.

ST_RETIRED

ST_RETIRED is incorrectly triggered on store-exclusive instructions that do not have exclusive access to the memory addressed. This event can be considered unaffected unless executing code containing store-exclusives. For example, semaphores and mutexes, in which case, you must avoid using it.

Workaround

There is no workaround for this erratum.

1810002
An instruction or exception which causes lockup might not enter debug state when expected

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

In some cases a debug event might not cause entry to debug state when expected. This only occurs in a small number of scenarios where the instruction or exception will cause lockup.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with Halting Debug.

Conditions

This erratum occurs in the following cases:

- Vector catch
  - A HardFault exception is being taken
  - The vector fetch of the HardFault handler faults and will cause lockup
  - The HardFault exception is entered and the processor locks up
  - An NMI occurs before the vector catch event is handled. The processor re-enters lockup on returning from the NMI to the HardFault handler.
- Halt stepping an instruction which causes lazy stacking.
  - The lazy stacking process has a fault which causes lockup.
  - AIRCR.IESB is set. The implicit error barrier detects an imprecise error due to the lazy stacking memory accesses.
  - The processor enters lockup as expected but the step event does not occur.

Implications

In both cases the processor enters the lockup state but a debug event does not occur

- The processor does not halt
- DFSR does not indicate the missed event.

A debugger can still enter debug state by setting C_HALT.

Workaround

There is no workaround for this erratum.

1745058
Processor might step two consecutive exception entries when debug halt stepping

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1.

Description

The Armv8-M architecture mandates (in Rule ZVKS) that when halt stepping only one `PushStack()` sequence can be stepped, that is, one exception entry. However, this erratum might result in the processor halt stepping an additional entry sequence before entering debug state.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with halting debug (DBGLVL>0).

Conditions

- Debug halt stepping is active.
- An exception entry occurs which must be stepped.
- A new exception occurs after the first exception has been taken but before debug state is entered.

Implications

There will be an additional exception entry sequence before entering debug state.

Workaround

There is no workaround.

1858750
A branch that is taken after an interstating branch that has not been taken might incorrectly fetch as if it is in Non-secure state

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2

Description

It has been found that an unconditional branch following a condition code failing interstating branch might cause instructions to be fetched as if in Non-secure state. This can lead to Non-secure instructions being executed in Secure state or an unexpected SFSR.INVEP SecureFault being raised for instructions in Secure memory.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the M-profile Security Extension.

Conditions

The erratum might occur if the processor is executing in Secure state and the following code sequence occurs:

1. One of the following flag setting instructions is executed:
   - `MRC`
   - `CLRM`
   - `MULS`
   - An arithmetic operation operating on two registers with one being shifted. For example, `ADD Rd, Rn, Rm <shift> #imm.`
2. An interstating branch (BXNS, BLXNS) which fails its condition code check.
   - The original flags, before the sequence, must indicate that the BXNS instruction will pass its condition code check.
   - The branch must be the last in an IT block.
3. An unconditional branch that is one of `B`, `BX LR` or `BL`.

Implications

The unconditional branch is taken but can result in the *Instruction Fetch Unit* (IFU) incorrectly transitioning to Non-secure. The main processor state does not transition. This can result in unexpected behavior:

- If the target of the unconditional branch is in Non-secure memory, then Non-secure instructions execute in Secure state. This behavior cannot be exploited by Non-secure code and is considered implausible for Secure code.
- If the target of the unconditional branch is in Secure and Non-secure Callable memory, then everything behaves as normal.
- If the target of the unconditional branch is only in Secure memory, then the processor raises an unexpected SFSR.INVEP SecureFault.

The sequence is not expected to be seen in realistic software and must be avoided.

Workaround

No workaround is required.

1784253
Invalid DWT trace on MVE load or store with predication

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1

Description

The *Data Watchpoint and Trace* (DWT) produces an erroneous byte of all zeroes in the trace stream when there is a data address with value match on two bytes of an MVE load or store beat with predicated bytes in between.

Configurations affected

This erratum affects all configurations of the processor configured with parameters MVE != 0 and DBGLVL != 0.

Conditions

The following conditions have to be met for this erratum to occur:

- The DWT is enabled (DEMCR.TRCENA == 1). A DWT comparator n is programmed to match a data address such that the DWT_FUNCTIONn.MATCH field is programmed to one of the following:

```
0b1101, Data Address with Value, only writes
0b1110, Data Address with Value, only reads
0b1100, Data Address with Value, either reads or writes
```

- The DWT comparator n is programmed to generate Data Trace Data Value packet such that DWT_FUNCTIONn.ACTION field is programmed to one of the following.

```
0b10, generate a data value packet on comparator match
0b11, generate a data value and PC packet on comparator match
```

- The comparator that matches is the lowest indexed comparator in the DWT that generate a Data Trace Data Value packet type for that access.
- The data value being traced is an MVE load or store and has one of the following byte enables:

```
B3 B2 B1 B0
0  1  0  1
1  0  0  1
1  0  1  0
```

Where:

- Bx is 0 for every byte that is predicated.
- Bx is 1 for every byte that is not predicated.

Implications

An erroneous byte of all zeroes will be present in the trace stream which could be incorrectly decoded by the downstream trace logic.

Workaround

There is no workaround for this erratum.

2703268
L1D_CACHE_REFILL and L1D_CACHE_MISS_RD PMU events might be inaccurate

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. r0p2, r1p0, r1p1. Open

Description

The following *Performance Monitoring Unit* (PMU) events supported by the Cortex-M55 processor might trigger too frequently:

- L1D_CACHE_REFILL
- L1D_CACHE_MISS_RD

Configurations affected

This erratum affects configurations of the Cortex-M55 processor with the Verilog parameter DBGLVL != 0 and DCACHESZ != 0.

Conditions

There are no special conditions associated with this erratum.

Implications

L1D_CACHE_REFILL

L1D_CACHE_REFILL is incorrectly triggered for data cache refills caused by PLD instructions and the data prefetcher. Hence, this can cause the frequency for this event to be higher than the frequency of the L1D_CACHE event. This can result in an inaccurate value for data cache refill rate ratio calculations.

L1D_CACHE_MISS_RD

L1D_CACHE_MISS_RD is incorrectly triggered for data cache misses due to PLD instructions, store instructions and linefills made by the data prefetcher. Hence, this can cause the frequency for this event to be higher than the frequency of the L1D_CACHE_RD event. This can result in an inaccurate value for data cache read operation miss rate ratio calculations.

Workaround

There is no workaround for this erratum.

1698164
Processor might not step correctly after a branch which changes security state

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1.

Description

When a security state transition occurs as the result of a branch it is possible that the first instruction is not stepped correctly. This scenario only occurs when the target state has Unprivileged debug enabled.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the Security Extension.

Conditions

Unprivileged debug stepping is enabled for the target state but not the current state. One of the following security transitions occurs:

1. An interstating branch (BXNS or BLXNS) causes the processor to enter non-secure state.
2. An SG instruction causes the processor to enter secure state

Implications

The first instruction executed in the target state might not be stepped when expected. The processor correctly steps the second instruction.

Workaround

There is no workaround.

1871826
The DWT might incorrectly expose the address of a BXNS

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2

Description

The *Data Watchpoint and Trace Unit* (DWT) might expose the secure address of an interstating BXNS instruction following an interstating SG instruction, through a periodic PC sample packet which should not be generated.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the DWT and Security Extension.

Conditions

1. DWT trace is configured as follows:
    - Periodic PC packet generation is enabled
    - Secure non-invasive debug is not allowed
2. An SG instruction is executed causing a transition to Secure state.
3. A BXNS instruction is executed immediately after the SG instruction causing a transition back to Non-secure.

Implications

The DWT might incorrectly generate a periodic PC sample packet, which can expose the Secure address of the BXNS instruction, on an interstating SG-BXNS sequence. However, there is no Secure information exposed, because the sequence traced does not include any control flow change in Secure code.

Workaround

There is no workaround.

1709730
ETM might trace some interrupted instructions incorrectly

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r0p1.

Description

It has been found that the *Embedded Trace Macrocell* (ETM-M55) might trace some instructions incorrectly when they are interrupted by an exception or entry into debug state.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the ETM.

Conditions

ETM trace is enabled and one of the following scenarios occur:

1. An LDM starts executing but does not complete. This can be because of an asynchronous interrupt or synchronous fault.
2. AIRCR.IESB is set to1 and a PC modifying instruction which triggers exception return is interrupted while the implicit ESB is being resolved.

Implications

The implications of this erratum are:

1. The LDM is not traced when it should be even though it started execution
2. The instruction is traced when it should not be since it did not complete (the implicit ESB did not finish).

Workaround

There is no workaround.

2619476
TCM ECC errors for S-AHB accesses will not be reported in sleep state

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

Due to this erratum, correctable and uncorrectable *Error Correcting Code* (ECC) errors that occur during *Direct Memory Access* (DMA) accesses to *Tightly Coupled Memories* (TCMs) via the S-AHB interface when the processor is in sleep state will not be reported in the *Reliability*, *Availability*, *and Serviceability* (RAS) registers and on the DME interface. But they will be recorded in the TEBR registers.

This erratum only affects reporting of TCM ECC errors when the processor is in the sleep state and therefore the functional operation of the TCM ECC logic is otherwise unaffected. Hence, for example correctable TCM ECC errors for read transactions via the S-AHB interface are corrected as normal. Also, uncorrectable TCM ECC errors are indicated by an ERROR response on the HRESPS S-AHB interface signal for read transactions. See the Cortex-M55 TRM for further details about the TCM ECC functionality.

Configurations affected

This erratum affects configurations where the ECC parameter is set to 1.

Conditions

The following conditions are required to cause this erratum:

- The processor is in sleep state. Normally this is the result of executing a WFI or WFE instruction.
- A DMA controller accesses a TCM via the S-AHB interface.
- An ECC error is detected for a TCM read initiated from the S-AHB interface. This can be a read transaction or read from a RMW caused by a sub-chunk write transaction.

See the Cortex-M55 TRM for details about S-AHB interface availability in low power states.

Implications

TCM ECC errors may not reported in the RAS registers or on the DME interface. This could result in some TCM ECC errors not being recorded by the system fault monitoring hardware.

Workaround

The TEBR registers could be checked periodically by software and any change should then be notified to the system fault monitoring hardware.

2567270
A BX using the LR where the t-bit (bit[0]) of the LR is not set, under certain circumstances, might not cause a INVSTATE fault

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

A BX LR instruction is executed early in the Cortex-M55 pipeline sometimes speculatively ahead of other branches which may be interlocked. Once the interlocked branches are resolved, the BX LR speculative execution is either allowed to continue or flushed. At this point, any fault that occurred executing the BX LR is recognized and raises an exception. Under certain specific conditions a corrupt LR, where bit[0] is not set, continue executing instead of a INVSTATE fault occurring.

Conditions

This erratum occurs when the processor executes the following code sequence:

```
Bcc <imm> # the condition depending on the flags of a previous instruction, such that it eve
ntually fails and is not taken
BX LR     # bit[0] of LR must be '0'
```

When:

- The BX LR instruction is located in the last three words of the before the boundary of the TCM or cache region
- An implementation dependent stall occurs in the processor pipeline while executing the sequence

Implications

The processor continues to execute from the corrupt address held in the LR. This erratum cannot cause escalation of Security or privilege state in the processor.

Workaround

There is no workaround. It is expected that the compiler will not generate a corrupt LR as a return address.

2653102
Unprivileged debug transactions to the STIR and EVENTSPR registers can pend interrupts

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

Due to this erratum, an unprivileged debugger access to the architectural STIR register can pend an interrupt (IRQ) and an unprivileged debugger access to the `IMPLEMENTATION DEFINED` EVENTSPR can pend a *Non-Maskable Interrupt* (NMI). Writes to these two registers from an unprivileged debugger should be ignored and a fault should be returned to the debugger.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor with Halting debug, DBGLVL > 0.

Conditions

The erratum part concerning STIR register occurs when all the following conditions are met:

- DAUTHCTRL.UIDAPEN is set to 0b1
- The CCR.USERSETMPEND bit is 0b0
- A debugger performs an unprivileged byte or halfword access to the second byte of the STIR register

The erratum part concerning EVENTSPR register occurs when all the following conditions are met:

- DAUTHCTRL.UIDAPEN is set to 0b1
- A debugger performs an unprivileged access to EVENTSPR

Implications

The implications of unprivileged debugger accesses to the STIR register under the conditions listed above are as follows:

- A byte or halfword read/write that is not word-aligned will not fault.
- A byte write to address 0xE000EF01 can pend interrupt 0 or interrupt 256 depending on bit [0] of the written byte.

The implications of unprivileged debugger accesses to the EVENTSPR register under the conditions listed above are as follows:

- A write to the address 0xE001E400 with bit[1] set in the write-data will pend a non-maskable interrupt

Note:

- There is no possibility of leaking Secure or privileged information to an unprivileged debugger.
- This erratum does not apply to software running on the processor and so the behavior is as specified in the Armv8-M Architecture Reference Manual. Therefore, unprivileged software cannot access the STIR register if it is not permitted to do so by the CCR.USERSETMPEND bit or the EVENTSPR register
- The STIR and EVENTSPR register are both write-only registers, so this erratum has no implications for unprivileged debugger reads.
- The implications of this erratum can be mitigated by including interrupt service routines for NMI and interrupt 0 and interrupt 256 (if these maskable interrupts are enabled).

Workaround

There is no workaround for this erratum.

2675831
After deactivating the instruction cache self-modified code might not be executed correctly

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

Cortex-M55 supports the use of self-modifying code but requires the following before the modified code is executed:
1) Cache maintenance operations are performed to flush the old code from the instruction cache.
2) An ISB instruction is executed to flush old code from buffers internal to the processor.

Cortex-M55 also allows the instruction cache to be deactivated using the IACTIVE bit of *Memory System Control Register* (MSCR). The instruction cache is usually deactivated to allow power down of the cache SRAM.

If the instruction cache is deactivated shortly before code is modified the old code might be executed despite the use of cache maintenance operations and the execution of an ISB instruction.

Due to this erratum, an internal instruction buffer is not flushed when the instruction cache is inactive. This means old code can be erroneously executed after the associated address has been modified by software or from a debugger using the appropriate cache maintenance and synchronization operations.

Configurations affected

This erratum affects configurations of the Cortex-M55 processor that includes the instruction cache. Configuration parameter ICACHESZ > 0.

Conditions

1) There is a cache line containing executable code and which has memory attributes indicating that it is cacheable.
2) The cache line is in normal memory and not in *Instruction Tightly Coupled Memory* (ITCM) and *Data Tightly Coupled Memory* (DTCM).
3) The cache line is read from external memory due to instruction fetch.
4) A store is executed that writes to the memory location corresponding to the address which has already been fetched into the instruction cache
5) The write modifies instruction data and the data is cleaned out to L2 memory using the code sequence in the Arm v8-M Architecture Reference Manual (see identifier IMLLC).
6) The instruction cache is active when the cache line is read from external memory.
7) The instruction cache is inactive when the required cache maintenance operations occur.
8) The instruction cache is inactive when the required ISB instruction is executed.
9) Between the write and the execution of the modified instruction no other cache line is read from external memory due to instruction fetch.

Note: A cache line is an area of memory 32 bytes in size and 32 byte-aligned.

Implications

If the conditions listed in this erratum occur the previous code at the instruction location will be executed from the internal buffer until it is replaced with new instructions.
The erratum is unlikely to result in any real issues in a realistic software or debug scenario as executing code from an inactive instruction cache will result in very low performance. When the cache is inactive to minimize power Arm recommends running code from the *Tightly Coupled Memory* (TCM).

Workaround

No workaround necessary as this issue is unlikely to be encountered in a realistic software or debug scenario.

2312624
Trace reconstruction might be incorrect, due to invalid exception return address on a Lockup exception

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

This erratum can cause the *Embedded Trace Macrocell* (ETM) to trace an incorrect value of the preferred exception return address on a Lockup exception. As a result, a trace analyzer might be unable to reconstruct the stream of instructions correctly.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with an ETM.

Conditions

This erratum can occur in one of the following scenarios:

Scenario 1: Lockup is caused by one of the following cases:

- A BusFault on instruction fetch
- An instruction access violation
- A memory fault, an imprecise bus fault, or an ECC fault caused by a memory access of an LDM/STM instruction
- A software BKPT
- A faulting VLDRB.8, VSTRB.8 instruction

Scenario 2: Lockup is caused by an asynchronous exception during tail-chain.

Implications

In the first scenario, lockup is traced with the Lockup address (0xEFFFFFFE) instead of the address of the instruction that caused it. In this case, a trace analyzer might miss the execution of instructions between the last branch target, exception target, or ISB, and the lockup exception.

In the second scenario, the preferred exception return address of the lockup exception is 0xFFFFFFFE instead of the Lockup address (0xEFFFFFFE). In this case, the trace reconstructor might not be able to reconstruct the trace stream correctly.

Workaround

There is no workaround for the first scenario of this erratum.

The second scenario can be avoided if the trace reconstructor assumes that, immediately following an exception entry, a Lockup that traced 0xFFFFFFFE as its preferred exception return address, has the Lockup address as its PER instead.

2295637
DWT_FUNCTION.MATCHED of the DWT Comparator gets cleared on a Speculative read that gets cancelled

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The MATCHED bit field of DWT Comparator Function Register, DWT_FUNCTION, should clear to zero when read. Due to this erratum, the bit field will be cleared on a Speculative read that gets canceled due to an LSU instruction ahead of it which faults or is interrupted, too. This is the case for all implemented *Data Watchpoint and Trace* (DWT) comparators.

Configurations affected

This erratum affects all configurations of the processor configured with Halting debug, DBGLVL > 0.

Conditions

This erratum occurs when all the following conditions are met:

- The DWT Comparator n is implemented, where n = 0-7, and it has a match;
- A Speculative read is made to the DWT_FUNCTIONn register which is canceled due to a load or store instruction ahead of it which faults or is interrupted;
- A Non-speculative read is made to the DWT_FUNCTIONn register. The MATCHED field will be 0b0 as it was incorrectly cleared by the previous Speculative read.

Implications

If this erratum occurs, a DWT comparator match will be missed if DWT_FUNCTION.MATCHED bit field is read following a Speculative read of the bit field that got canceled due to the scenario described above.

Workaround

There is no workaround for this erratum.

2285085
DWT Performance Profiling Counters might sometimes have an incorrect value

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

This erratum can cause the *Data Watchpoint and Trace* (DWT) Profiling Counters to take an incorrect value. Specifically, it can cause the DWT Folded Instruction Count Register (DWT_FOLDCNT) or the DWT LSU Count Register (DWT_LSUCNT) to increment incorrectly.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DWT and *Instrumentation Trace Macrocell* (ITM) (DBGLVL>0).

Conditions

This erratum occurs in one of the following scenarios of DWT trace:

Scenario 1:

DWT_CTRL.FOLDEVTENA==1, ITM_TCR.STALLENA==1 and the following sequence of conditions is met:

1. An instruction or data traced by the DWT causes the ITM to stall the PE
2. A dual-issue pair is executed, while the ITM stall request is asserted

Scenario 2:

DWT_CTRL.LSUEVTENA == 1 and one of the following conditions is met:

- A. ITM_TCR.STALLENA==1, an instruction or data traced by the DWT causes the ITM to stall the PE, and a memory transaction of a load-store instruction is performed, while the ITM stall request is asserted
- B. The *M-profile Vector Extension* (MVE) extensions are implemented and a vector load-store instruction which is subject to beat-wise execution is executed
- C. A CLRM, CLREX, or a PLD instruction is executed, or a memory transaction caused by a debug access is performed

Implications

The DWT Folded Instruction Count Register (DWT_FOLDCNT) or the DWT LSU Count Register (DWT_LSUCNT) might have an incorrect value. Specifically:

- in Scenario 1, DWT_FOLDCNT will increment more than it should
- in Scenarios 2.A and 2.B, DWT_LSUCNT will fail to increment
- in Scenario 2.C, DWT_LSUCNT will increment more than it should

Workaround

There is no workaround for this erratum.

2397983
PMU event L1I_CACHE is incorrectly implemented in the instruction cache

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

The *Performance Monitoring Unit* (PMU) event L1I_CACHE is defined in the Armv8-M architecture and counts each Attributable access at least to the L1 instruction cache. Due to this erratum, the event does not count accesses to the instruction cache where the fetch address is sequential to the previous access and the access address does not cross a 32-byte cache line address boundary.

Configurations affected

This erratum affects configurations of the Cortex-M55 processor that includes the instruction cache. Configuration parameter ICACHESZ > 0.

Conditions

This erratum occurs when all the following conditions are met:

- The event is selected in the PMU Event Type and Filter Register, PMU_EVTYPERn, configuring event counter n
- Event counter n is enabled in PMU_CNTENSET.P[n]

Implications

If the L1I_CACHE PMU event is used to gather statistics on the instructions cache, the number of recorded accesses will be lower than expected.

Workaround

There is no workaround for this erratum.

2019023
The DWT might incorrectly expose the IPSR of a Secure exception when the ITM stall is enabled and Secure non-invasive debug is prohibited

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p1 and r0p2. Fixed in r1p0

Description

The Armv8-M architecture (in rule RRGCV) specifies the ability for the *Instrumentation Trace Macrocell* (ITM) to guarantee delivery of hardware trace packets. This erratum can cause the *Interrupt Program Status Register* (IPSR) value of a Secure exception to be incorrectly exposed through an Exception Trace packet, when ITM stalling is enabled and Secure non-invasive debug is prohibited.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the *Data Watchpoint and Trace Unit* (DWT) and ITM (DBGLVL>0), and the Security Extension implemented.

Conditions

The erratum occurs if all the following conditions are met:

- ITM stall is enabled (ITM_TCR.STALLENA==1)
- DWT trace is configured with Secure non-invasive debug prohibited
- There is an exception entry from Non-secure state to Secure state
- DWT data trace of the last stacking operation causes the ITM to stall the PE

Implications

The DWT might expose the IPSR of a Secure exception to a Non-secure debug or trace system, through an Exception Trace packet. However, there is no significant piece of Secure information to be leaked, as the exception number of the Secure exception can be known to the user.

Workaround

There is no workaround for this erratum.

2461408
Unprivileged debugger access to DPDLPSTATE register incorrectly modifies register state

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, and r1p1. Open

Description

The processor contains an **`IMPLEMENTATION DEFINED`** register DPDLPSTATE in the *Private Peripheral Bus* (PPB) region that specifies the wanted low-power state for the Debug power domain in the processor. This register is only accessible by privileged code or a privileged debugger. Due to this erratum, a write to the DPDLPSTATE register from an unprivileged debugger will return a fault, as expected, but the register state will be incorrectly updated.

Configurations affected

This erratum affects all configurations of the processor configured with Halting debug, DBGLVL > 0.

Conditions

This erratum occurs when Unprivileged Invasive DAP Access Enable, DAUTHCTRL.UIDAPEN = 1 (either bank) and there is a write access through an unprivileged DAP request to the DPDLPSTATE register located at address 0xE001E304.

Implications

If this erratum occurs, an unprivileged debugger write will change the state of the DPLPSTATE register. This can select a lower or higher power mode for the Debug power domain than intended by a privileged debug environment.

Note: Changing the value of the register will not change the functional behavior of the processor. Even if debug domain is powered down it will be powered up again if a subsequent debug access is made.

Workaround

There is no workaround for this erratum.

2454643
A VSTMDB which specifies unimplemented registers may write to unexpected memory locations

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, and r1p1. Open

Description

The VSTMDB and FSTMDBX instructions specify a list of floating-point registers to store to memory. The instruction encoding allows a large number of registers to be specified. It has been found that if the specified registers include those not supported by the processor then data may be written to memory locations above the base address.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the floating-point or *M-profile Vector Extension* (MVE) extensions.

Conditions

This erratum will occur if the *Processing Element* (PE) executes a VSTMDB or FSTMDBX instruction with an immediate value greater than 63.

Note: In almost all cases the specified encoding is UNPREDICTABLE. All cases were UNPREDICTABLE prior to Armv8.1-M.

Implications

The PE will incorrectly write to memory locations immediately above the base address (at most 32 words).

Workaround

There is no workaround. It is expected that the compiler will not generate these encodings.

2161015
When tracing an exception the ETM may incorrectly indicate a serious fault is pending

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The *Embedded Trace Macrocell* (ETM) Exception instruction trace packet contains information about any serious faults pending (field P). It has been found that this field may be incorrectly set when entering the handler of a serious fault.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the ETM.

Conditions

This erratum occurs when the ETM trace is enabled and all the following conditions occur:

- There is only one serious fault pending (one of BusFault, HardFault, MemManage SecureFault)
- The PE is entering the exception handler of the serious fault

Implications

If this erratum occurs, the ETM Exception packet will indicate that a serious fault is pending even though the fault is no longer pending.

Workaround

There is no workaround.

2129008
A Secure read of ICSR_NS.VECTPENDING might return the Secure value

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2. Fixed in r1p0.

Description

The ARMv8-M architecture defines a way for a Secure PE or debugger to access the Non-secure version of a register. In some cases the Secure data should be masked when read by Non-secure. It has been found that the ICSR.VECTPENDING field may return a Secure value when accessed as Non-secure from Secure state.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the Security Extension.

Conditions

A Secure exception is the highest priority pending exception and one of the following occurs:

- The Secure PE attempts to read the Non-secure version of Interrupt Control and State Register (ICSR_NS) located at address (0xE002ED04)
- A Secure debugger (DHCSR.S_SDE==1) attempts to read the ICSR_NS (that is DSCSR.SBRSELEN==0 and PE is Non-secure or DSCSR.SBRSELEN==1 and DSCSR.SBRSEL==0)

Implications

The ICSR_NS read will return the unmasked value of the VECTPENDING field. This is only possible from Secure state. No information is leaked to Non-secure.

Workaround

There is no workaround for this erratum.

2267938
The DWT might not trace an instruction when ITM stall is enabled

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The Armv8-M architecture (in rule **RRGCV**) specifies the ability for the *Instrumentation Trace Macrocell* (ITM) to guarantee delivery of hardware trace packets. This erratum can cause the *Data Watchpoint and Trace* (DWT) not to generate a PC value packet, when ITM stalling is enabled.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DWT and ITM (DBGLVL>0).

Conditions

This erratum occurs if the ITM stall is enabled (ITM_TCR.STALLENA==1) and the following sequence of events occurs:

1. An instruction or data traced by the DWT causes the ITM to stall the PE
2. A dual-issue pair is executed, while the ITM stall request is asserted
3. One of the following scenarios occurs:
    - A fault, an asynchronous interrupt or debug halt request occurs before the dual-issue pair completes, or
    - A UDIV/SDIV, an ADD/SUB SP, or a Long MVE Shift instruction with CP10 not present, is executed with an implicit LE

Implications

The DWT might not generate a PC value packet, when it matches on the instruction address of the dual-issued instruction.

Workaround

There is no workaround for this erratum.

2226595
The DWT might generate a Data trace PC value packet with the incorrect PC, when ITM stall is enabled

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The Armv8-M architecture (in rule **RRGCV**) specifies the ability for the *Instrumentation Trace Macrocell* (ITM) to guarantee delivery of hardware trace packets. This erratum can cause the *Data Watchpoint and Trace* (DWT) to generate a Data trace PC value packet on exception entry stacking operations with the incorrect PC value when ITM stalling is enabled.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DWT and ITM (DBGLVL>0).

Conditions

This erratum occurs if the ITM stall is enabled (ITM_TCR.STALLENA==1) and one of the following scenarios occurs:

Scenario 1 - There is a sequence of events where:

1. An instruction or data traced by the DWT causes the ITM to stall the PE
2. An LDR PC instruction is executed
3. An asynchronous interrupt occurs while the LDR PC is being executed
4. The LDR PC instruction completes

Scenario 2 - There is a sequence of events where:

1. An LDM/STM or a vector load/store instruction is executed
2. The load/store instruction or a data transaction traced by the DWT causes the ITM to stall the PE
3. The load/store instruction completes its execution
4. A dual-issue pair is executed while the ITM stall request is asserted
5. A fault, an asynchronous interrupt, or debug halt request occurs before the dual-issue pair completes

Implications

Scenario 1

The DWT might generate a Data trace PC value packet on the exception entry stacking operations with the incorrect PC. Specifically, the PC value in the Data trace PC value packet might be the instruction address of the LDR PC, instead of the return address of the exception.

Scenario 2

The DWT might generate a Data trace PC value packet on the last memory transaction of the load/store instruction, with the incorrect PC. Specifically, the PC value in the Data trace PC value packet might be the instruction address of the older dual-issued instruction, instead of the address of the load/store instruction.

Workaround

There is no workaround for this erratum.

1960594
Performance Monitor Unit is not disabled on Warm reset

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2. Fixed in r1p0.

Description

The Cortex-M55 processor includes a *Performance Monitor Unit* (PMU) which can be used to count architectural and implementation defined events in the processor. The PMU functionality should be disabled when a Warm reset is applied to the processor. However, as a result of this erratum, PMU functionality is not disabled when Warm reset is applied to the processor, and this can cause unexpected behavior after reset.

Configuration affected

This erratum affects all configurations where the PMU is included in the processor (when the Verilog parameter DBGLVL != 0).

Conditions

This erratum occurs when the following conditions are met:

- The PMU is enabled and configured to count an event
- A warm reset event occurs

Implications

The PMU remains enabled and active after the Warm reset. This can result in the following unexpected behavior in the processor:

- The PMU event counter increments if a configured PMU event occurs. The PMU cycle counter increments.
- The event counter or cycle counter can overflow and raise a debug event if enabled or generate a trace packet from the ITM if trace is authorized after reset

Workaround

The PMU can be disabled at the start of the reset handler by clearing PMU_CTRL.E in software or by halting the processor on reset with vector catch and clearing the register from the debugger.

2252094
PMU cycle counter and PMU event counter incorrectly stop counting when there is an overflow

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1

Description

The *Performance Monitoring Unit* (PMU) has a feature to freeze the event counters on overflow. Due to this erratum, the PMU cycle counter will incorrectly stop counting when there is an overflow of either the PMU cycle counter or any of the PMU event counters.

Furthermore, PMU event counters will stop counting when the PMU cycle counter overflows.

Configurations affected

This erratum affects all configurations of the processor configured with Halting debug, DBGLVL > 0.

Conditions

For the erratum part concerning the PMU cycle counter to occur, the following condition applies:

- PMU cycle counter is enabled and counts, that is:
  - Core is not halted
  - Trace is enabled, DEMCR.TRCENA = 1
  - PMU_CTRL.E = 1 and PMU_CNTENSET.C = 1
  - Core is in Non-secure state or __ PMU cycle counter is enabled in Secure state, PMU_CTRL.DP = 0
- Freeze on overflow is enabled, PMU_CTRL.FZO = 1
- Cycle counter overflows *or* PMU_OVSSET.C is 1 *or* any PMU event counter overflows *or* PMU_OVSSET.Pn is non-zero

For the erratum part concerning the PMU event counter to occur, the following condition applies:

- PMU event counter is enabled and counts, that is:
  - Core is not halted
  - Trace is enabled, DEMCR.TRCENA = 1
  - DWT profiling counters are disabled, that is all the following are met:
    - DWT_CTRL.CPIEVTENA = 0
    - DWT_CTRL.EXCEVTENA = 0
    - DWT_CTRL.SLEEPEVTENA= 0

        - DWT_CTRL.LSUEVTENA = 0
        - DWT_CTRL.FOLDEVTENA= 0
    - PMU_CTRL.E = 1 and PMU_CNTENSET.Pn[n] = 1
- Freeze on overflow is enabled, PMU_CTRL.FZO = 1
- Cycle counter overflows *or* PMU_OVSSET.C is 1

Implications

If this erratum occurs, PMU cycle counter and PMU event counters will read an incorrect value if the PMU cycle counter or any of the PMU event counters overflow.

Workaround

There is no workaround for this erratum.

2173589
SBIST AHB interconnect can forward the incorrect value for HSEL

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r1p0. Fixed in r1p1.

Description

The SBIST AHB interconnect arbitrates AHB transactions on the *Slave-AHB* (S-AHB) interface between the external system master and the internal SBIST AHB Master that is part of the *Software Test Library* (STL) hardware. This erratum can cause a transaction from the external master to be lost if the interface is being held by the SBIST Master.

As a consequence, the SBIST AHB Master will not be used by the STL to cover the processor S-AHB Interface. Any STL test that makes use of the SBIST AHB Master will be disabled.

Configurations affected

This erratum affects configurations that include the STL HW at the MCU Level (SBISTC = 1)

Conditions

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

1. The SBIST AHB Master is performing a transaction that is keeping the internal AHB slave busy (either a burst transaction or a single transaction for which the slave is adding wait states).
2. The external master starts a single transaction in the same cycle as in condition one above. Because the slave is busy, this transaction will be put into an address phase buffer by the interconnect.
3. The external master drops **HSEL** during the data phase for the transaction in condition two. The interconnect will forward this current value of **HSEL** to the slave instead of the value in the address phase buffer, causing the slave to observe a transaction with **HSEL** = 0 (which will be ignored).

Implications

If this erratum occurs, a transaction on the Cortex-M55 processor S-AHB interface from the external system can be lost.

Workaround

No workaround is required because the STL will not enable the SBIST AHB Master.

2219778
Conditional instruction tracing might not work due to an interrupted VSTM, when ETM is configured to trace condition codes for data instructions only

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

This erratum can cause Conditional instruction result packets to be associated to an incorrect instruction due to an interrupted VSTM instruction, when the *Embedded Trace Macrocell* (ETM) is configured to trace condition codes only for load/store instructions.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with an ETM, and the *M-profile Vector Extension* (MVE) or the Floating-point Extension implemented.

Conditions

This erratum can occur if the ETM trace is enabled, TRCCONFIGR.COND has one of the following values:

- 0b001
- 0b010
- 0b011

and the following sequence of events occurs:

1. A VSTM instruction is executed
2. The VSTM completes at least one memory access
3. An external interrupt causes the VSTM instruction to be interrupted before it completes

Implications

Conditional instruction result packets might be associated with an incorrect instruction.

Workaround

To avoid this erratum, set TRCCONFIGR.COND to 0b111 (ETM is configured so that all conditional instructions are traced).

2176014
A hardware breakpoint might be ignored after a Secure function call with unaligned SG

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1.

Description

The *Flash Patch and Breakpoint* (FPB) provides a mechanism to raise a hardware breakpoint on a specific fetch address. Due to this erratum, a breakpoint might be missed when transitioning from Non-secure to Secure state when the *Secure Gateway* (SG) instruction is not word-aligned.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with Halting debug (DBGLVL > 0) and the Security Extension.

Conditions

This erratum occurs when only Secure unprivileged debug is allowed and Non-secure state calls a Secure function where:

- The SG instruction at the target address is not word-aligned
- The FPB is set to match on the instruction following the SG (in other words, the FPB is programmed to match on the first halfword after the SG)

Implications

If this erratum occurs, the breakpoint will be missed and the PE will continue executing.

Workaround

There is no workaround.

2132823
TPIU does not generate frame synchronization packets when TPIU_PSCR is programmed with a value greater than 16

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Fixed in r1p1

Description

The *Trace Port Interface Unit* (TPIU) Periodic Synchronization Counter has a maximum value of 2^{16}, which is smaller than the maximum value programmable in the *TPIU Periodic Synchronization Control Register*, TPIU_PSCR, 2^{31}.

If the programmed reload value is greater than the maximum value of 2^{16}, then the Periodic Synchronization Counter should be reloaded with its maximum value and the TPIU would generate synchronization requests at this interval. Due to this erratum, if the programmed reload value is greater than the maximum value of 2^{16}, then the Periodic Synchronization Counter is disabled.

Configurations affected

This erratum affects all configurations of the processor configured with DBGLVL > 0 and either ITM !=0 or ETM != 0 and the Cortex-M55 TPIU is included in the system.

Conditions

This erratum occurs when a value greater than `0b10000` is programmed in TPIU_PSCR.PSCount.

Implications

If this erratum occurs, the Periodic Synchronization Counter will incorrectly become disabled, and no frame synchronization packets will be generated.

Workaround

There is no workaround for this erratum.

2134265
A faulting vector load/store instruction might not be traced by the DWT when ITM stall is enabled

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p1, r0p2. Fixed in r1p0.

Description

The Armv8-M architecture (in rule **RRGCV**) specifies the ability for the *Instrumentation Trace Macrocell* (ITM) to guarantee delivery of hardware trace packets. This erratum can cause a vector load/store instruction to not be traced by the *Data Watchpoint and Trace* (DWT), when ITM stalling is enabled.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DWT and ITM (DBGLVL>0), and the *M-profile Vector Extension* (MVE).

Conditions

This erratum occurs if the ITM stall is enabled (ITM_TCR.STALLENA==1) and the following sequence of events occurs:

1. An instruction or data traced by the DWT causes the ITM to stall the PE
2. A vector load/store instruction (which is subject to beat-wise execution) is executed
    - Beats 0 and 1 of the vector load/store instruction complete
    - Either beat 2 or beat 3 of the vector load/store instruction raises a fault

Implications

The DWT will not generate a PC value packet when matching on the instruction address of the vector load/store instruction.

Workaround

There is no workaround for this erratum.

2117423
A WFx Instruction may not be traced by the DWT when ITM stall is enabled

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p1, r0p2. Fixed in r1p0.

Description

The Armv8-M architecture (in rule **RRGCV**) specifies the ability for the *Instrumentation Trace Macrocell* (ITM) to guarantee delivery of hardware trace packets. This erratum can cause a WFE or WFI instruction not to be traced by the *Data Watchpoint and Trace* (DWT), when ITM stalling is enabled.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with a DWT and ITM (DBGLVL>0).

Conditions

This erratum occurs if the ITM stall is enabled (ITM_TCR.STALLENA==1) and the following sequence of events occurs:

1. An instruction or data traced by the DWT that causes the ITM to stall the PE
2. Another instruction
3. A WFE or WFI instruction that causes the core to enter a low-power state

Implications

The DWT might not trace the WFE or WFI instruction.

Workaround

There is no workaround for this erratum.

2085795
The DWT might incorrectly generate a match on the address or data of a memory transaction which is not performed

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2. Fixed in r1p0

Description

In the case of an interrupted LDM, STM, LDRD, STRD or Floating-point VLDM, VSTM, VLDR.F64, VSTR.F64 instruction to Device memory, the *Data Watchpoint and Trace Unit* (DWT) might incorrectly generate a match on the address or data, of the memory access which is not performed.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor that is configured with the DWT implemented.

Conditions

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

1. The PE executes an LDM, STM, LDRD, STRD, VLDM, VSTM, VLDR.F64, or VSTR.F64 instruction
2. The instruction targets Device memory
3. The instruction is interrupted before it completes

Implications

The DWT might generate a match on the address or data of a memory transaction which is not performed. Because the memory transaction is not performed, the data matched on will be an unknown value.

Workaround

There is no workaround for this erratum.

2080562
A Secure unprivileged debugger write to DSCSR.CDS may be ignored

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2. Fixed in r1p0.

Description

A debugger is allowed to change the current Security state of the PE by writing to the DSCSR.CDS field. It has been found that in one *Unprivileged Debug Extension* (UDE) scenario this write may not occur.

Configurations affected

This erratum affects all configurations of the Cortex-M55 processor configured with the M-profile Security Extension and support for an external debugger (DBGLVL>0).

Conditions

The PE must enter Debug state under the following conditions:

- The PE is in Secure unprivileged state
- Only Secure unprivileged debug is permitted (DHCSR.S_SDE==1, DHCSR.S_SUIDE==1 and DHCSR.S_NSUIDE==0)
- Non-secure state is in privileged mode due to CONTROL_NS.NPRIV==0

The debugger then writes 0 to DSCSR.CDS to put the PE into Non-secure state.

Implications

The write to DSCSR.CDS is ignored and the PE remains in Secure state.
Note: Direct accesses to Non-secure state (for example via DCRSR) behave as expected.

Workaround

A workaround is not required.

2791503
ITM can generate Synchronization and timestamp packets when NIDEN=0

Status

Affects: Cortex-M55
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0, r1p1. Open

Description

The *Instrumentation Trace Macrocell* (ITM) generates synchronization and timestamp packets based on *Data Watchpoint Trace* (DWT) signaling if trace is enabled, Non-secure Non-invasive debug is allowed, and synchronization packet transmission or timestamp packet generation is enabled respectively. Due to this erratum the ITM will incorrectly generate synchronization and Local and Global Timestamps packets even when Non-secure Non-invasive debug is not allowed if the remaining conditions are met.

Configurations affected

This erratum affects all configurations of the processor configured with DBGLVL > 0 and ITM = 1.

Conditions

This erratum occurs when the following conditions are met:

- Global enable for all DWT, PMU, and ITM features is set, that is, DEMCR.TRCENA=1, AND
- Non-secure Non-invasive debug is not allowed, AND
  - Synchronization packet transmission for a synchronous TPIU is enabled, that is, ITM_TCR.SYNCENA=1, OR
  - Local timestamp generation is enabled, OR
  - Global timestamp generation is enabled.

Implications

If this erratum occurs, the ITM will generate some trace even when trace generation is not allowed due to Non-Invasive Non-Secure debug not being enabled. However, as the packets being generated in this case are synchronization and timestamp packets and these packets do not represent security/functional insight into the software running on the PE in restricted modes, there are no security implications of this erratum.

When Cortex-M55 is used in a safety application there is an *Assumption of Use* (AoU) that ITM trace must be disabled. Although ITM trace is not disabled by setting NIDEN low (because of this erratum), it is still disabled by the DEMCR.TRCENA register which is set to zero at reset. In safety applications it is important to maintain the DEMCR.TRCENA register at zero to ensure ITM trace packets are disabled.

Workaround

There is no workaround for this erratum.