Arm Cortex-R52 Processor MP040

Software Developer Errata Notice

Date of issue: 08-Nov-2022

Non-Confidential

Document version: v17.0

Document ID: SDEN-857344

Copyright © 2022 Arm® Limited (or its affiliates). All rights reserved.

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

Non-confidential proprietary notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word "partner" in reference to Arm's customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm's trademark usage guidelines at **https://www.arm.com/company/policies/trademarks**.

Copyright © 2022 Arm® Limited (or its affiliates). All rights reserved.

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-R52 Processor MP040, 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

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

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-Nov-2022: Changes in document version v17.0

ID	Status	Area	Category	Summary
2743885	New	Programmer	Category B	GICR_TYPER registers do not reflect affinity configuration inputs

27-Jul-2022: Changes in document version v16.0

No new or updated errata in this document version.

19-Apr-2022: Changes in document version v15.0

ID	Status	Area	Category	Summary
2289294	New	Programmer	Category C	DTR flags not cleared on external debugger access while leaving Debug state

09-Jul-2021: Changes in document version v14.0

ID	Status	Area	Category	Summary
2077568	Updated	Programmer	Category C	ATB flush response may be delayed

20-Apr-2021: Changes in document version v13.0

ID	Status	Area	Category	Summary
2130897	New	Programmer	Category A	VLDM/VSTM may corrupt data under certain conditions

19-Mar-2021: Changes in document version v12.0

ID	Status	Area	Category	Summary
2072382	New	Programmer	Category C	Data ATB flush may not respond
2077568	New	Programmer	Category C	ATB flush response may be delayed

31-Jul-2020: Changes in document version v11.0

No new or updated errata in this document version.

17-Apr-2020: Changes in document version v10.0

No new or updated errata in this document version.

29-Mar-2019: Changes in document version v9.0

ID	Status	Area	Category	Summary
1412115	New	Programmer	Category B	False DCLS errors can be generated in configurations without bus protection

08-Feb-2019: Changes in document version v8.0

ID	Status	Area	Category	Summary
1328481	New	Programmer	Category C	The debugger view of the number of cores might be incorrect when switching between Split and Lock modes

26-Sep-2018: Changes in document version v7.0

No new or updated errata in this document version.

13-Feb-2018: Changes in document version v6.0

ID	Status	Area	Category	Summary
1050715	New	Programmer	Category C	Cross trigger interrupt generation does not work using CTIAPPPULSE register

12-Sep-2017: Changes in document version v5.0

ID	Status	Area	Category	Summary
865776	New	Programmer	Category C	Repeated asynchronous exceptions might generate false livelock indication
888346	New	Programmer	Category C	Implementation defined Data Cache Debug Read operations can only access the first doubleword of a cacheline
904856	New	Programmer	Category C	An MBIST request in the middle of pending AXIS requests might cause the core to deadlock
925639	New	Programmer	Category C	Debugger read transactions to Device Affinity Registers of CoreSight components return incorrect values

31-Mar-2017: Changes in document version v4.0

ID	Status	Area	Category	Summary
786955	New	Programmer	Category B	Wrong DFSR.STATUS or HSR.DFSC is set on asynchronous aborts because of detected errors on BRESPMx signals
827402	New	Programmer	Category B	A stream of data cache maintenance operations can lead to a deadlock
844709	New	Programmer	Category B (rare)	Advanced SIMD integer multiply instructions in IT block might cause data corruption
763109	Updated	Programmer	Category C	Undefined exception might be taken on an instruction cache ECC error
818201	New	Programmer	Category C	Accesses to direct-mapped EL2-controlled MPU registers are not unallocated when no programmable regions are configured
836865	New	Programmer	Category C	An EXOKAY response to a read on the LLPP interface indicates the wrong ADFSR value
848521	New	Programmer	Category C	The ETM can enter a low power state when active, potentially corrupting trace when TRCEVENTCTL1R.LPOVERRIDE is set
857196	New	Programmer	Category C	Performance Monitors count might be inaccurate for event 0x01B (INST_SPEC)
857509	New	Programmer	Category C	Mismatch between EDPRSR.SR and EDPRSR.R
857511	New	Programmer	Category C	Debug not entering Memory Access mode without setting EDSCR.ERR

22-Dec-2016: Changes in document version v3.0

No new or updated errata in this document version.

19-Dec-2016: Changes in document version v2.0

ID	Status	Area	Category	Summary
752456	New	Programmer	Category C	Store-exclusive to Shareable Device memory might cause core to deadlock/livelock
763109	New	Programmer	Category C	Undefined exception might be taken on an instruction cache ECC error
779430	New	Programmer	Category C	Bus timeout combined with online MBIST entry might lead to deadlock

22-Aug-2016: 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
2130897	Programmer	Category A	VLDM/VSTM may corrupt data under certain conditions	r0p0, r1p0, r1p1, r1p2, r1p3	r1p4
786955	Programmer	Category B	Wrong DFSR.STATUS or HSR.DFSC is set on asynchronous aborts because of detected errors on BRESPMx signals	r0p0	r1p0
1412115	Programmer	Category B	False DCLS errors can be generated in configurations without bus protection	r0p0, r1p0, r1p1, r1p2	r1p3
827402	Programmer	Category B	A stream of data cache maintenance operations can lead to a deadlock	r0p0	r1p0
2743885	Programmer	Category B	GICR_TYPER registers do not reflect affinity configuration inputs	r0p0, r1p0, r1p1, r1p2, r1p3, r1p4	Open
844709	Programmer	Category B (rare)	Advanced SIMD integer multiply instructions in IT block might cause data corruption	r0p0	r1p0
752456	Programmer	Category C	Store-exclusive to Shareable Device memory might cause core to deadlock/livelock	r0p0	r1p0
779430	Programmer	Category C	Bus timeout combined with online MBIST entry might lead to deadlock	r0p0	r1p0
763109	Programmer	Category C	Undefined exception might be taken on an instruction cache ECC error	r0p0	r1p0
818201	Programmer	Category C	Accesses to direct-mapped EL2-controlled MPU registers are not unallocated when no programmable regions are configured	r0p0	r1p0
1328481	Programmer	Category C	The debugger view of the number of cores might be incorrect when switching between Split and Lock modes	r1p0, r1p1, r1p2, r1p3, r1p4	Open
857511	Programmer	Category C	Debug not entering Memory Access mode without setting EDSCR.ERR	r0p0	r1p0

ID	Area	Category	Summary	Found in versions	Fixed in version
1050715	Programmer	Category C	Cross trigger interrupt generation does not work using CTIAPPPULSE register	r0p0, r1p0, r1p1	r1p2
904856	Programmer	Category C	An MBIST request in the middle of pending AXIS requests might cause the core to deadlock	r0p0, r1p0	r1p1
857196	Programmer	Category C	Performance Monitors count might be inaccurate for event 0x01B (INST_SPEC)	r0p0	r1p0
925639	Programmer	Category C	Debugger read transactions to Device Affinity Registers of CoreSight components return incorrect values	r0p0, r1p0	r1p1
865776	Programmer	Category C	Repeated asynchronous exceptions might generate false livelock indication	r1p0	r1p1
857509	Programmer	Category C	Mismatch between EDPRSR.SR and EDPRSR.R	r0p0	r1p0
836865	Programmer	Category C	An EXOKAY response to a read on the LLPP interface indicates the wrong ADFSR value	r0p0	r1p0
848521	Programmer	Category C	The ETM can enter a low power state when active, potentially corrupting trace when TRCEVENTCTL1R.LPOVERRIDE is set	r0p0	r1p0
888346	Programmer	Category C	Implementation defined Data Cache Debug Read operations can only access the first doubleword of a cacheline	r1p0	r1p1
2289294	Programmer	Category C	DTR flags not cleared on external debugger access while leaving Debug state	r0p0, r1p0, r1p1, r1p2, r1p3, r1p4	Open
2077568	Programmer	Category C	ATB flush response may be delayed	r0p0, r1p0, r1p1, r1p2, r1p3	r1p4
2072382	Programmer	Category C	Data ATB flush may not respond	r0p0, r1p0, r1p1, r1p2, r1p3, r1p4	Open

Errata descriptions

Category A

2130897
VLDM/VSTM may corrupt data under certain conditions

Status

Affects: Cortex-R52
Fault Type: Programmer, Category A
Fault Status: Present in r0p0, r1p0, r1p1, r1p2, r1p3. Fixed in r1p4.

Description

The Cortex-R52 processor supports fetching of memory data in either big-endian or little-endian modes. Changing between the two can be enabled by executing the **`SETEND`** instruction.

The processor also supports instructions that load (or store) multiple registers to (or from) the Advanced SIMD and floating-point register file. These instructions are the **`VLDM`**, **`VSTM,`** and their aliases **`VPOP`**, **`FLDMDBX`**, **`FLDMIAX`**, **`VPUSH`**, **`FSTMDBX`**, **`FSTMIAX`**.

Under certain and very specific conditions, in a program that contains both a **`SETEND`** instruction that changes the current endianness and a variant of a **`VLDM`** (or **`VSTM`**) instruction, the processor corrupts the loaded (or stored) data.

Configurations affected

The erratum only affects systems where the program contains instructions that change the endianness between big-endian and little-endian.

Conditions

This erratum occurs when all the following conditions are met:

- The program executes a load/store instruction followed by a conditional branch instruction
- The load/store instruction and the conditional branch instruction are dual-issued. This happens when the CPUACTLR.DIDIS field is set to 0.
- The load/store instruction stalls in the pipeline for at least five cycles. A load that misses in the data cache typically stalls for at least five cycles.
- The conditional branch instruction mispredicts at least once in the execution
  - In the fail path of the mispredicted branch, the program contains a **`SETEND`** instruction within

the first five instructions
- In the correct path of the mispredicted branch, a `VLDM.64/VSTM.64` is the first executed instruction and this instruction operates on D registers but accesses a memory location whose base address is not 8-byte aligned.

Implications

If this erratum occurs, the processor corrupts the loaded (or stored) data.

Workaround

To avoid this erratum, insert an `ISB` instruction before the `SETEND` instruction.

Category A (rare)

There are no errata in this category.

Category B

786955
Wrong DFSR.STATUS or HSR.DFSC is set on asynchronous aborts because of detected errors on **BRESPMx** signals

Status

Affects: Cortex-R52
Fault Type: Programmer, Category B
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor can be configured with a bus protection scheme that enhances the signal integrity of *AXI master* (AXIM), *AXI slave* (AXIS), Flash, and *Low Latency Peripheral Port* (LLPP) bus interfaces. Additional *Error Correcting Code* (ECC) bits are transported on the bus in parallel to the normal payload data. When the processor receives bus responses from the system, it decodes the additional ECC bits and detects whether the bus control or data signals are erroneous. Error handling varies and might range from assuming that the bus reply is correct to taking a data abort.

In the case of the AXIM bus port, when the bus protection scheme detects an error on the write response channel (**BRESPMx** signal), the processor takes an asynchronous abort. There are two possibilities:

- If the abort is taken to EL1, DFSR.STATUS is set to 0b010001 ("Slave error interrupt"), DFSR.ExT is set to 0b1 ("External abort marked as slave error"), ADFSR.PORT is set to 0b000 ("AXI"), and ADFSR.TYPE is set to 0b01 ("Error on Response").
- If the abort is taken to EL2, HSR.EC is set to either 0b100100 ("Data Abort from a lower Exception level") or 0b100101 ("Data Abort taken without a change in Exception level"), HSR.DFSC is set to 0b010001 ("Slave error interrupt"), HSR.EA is set to 0b1 ("External abort marked as slave error"), HADFSR.PORT is set to 0b000 ("AXI"), and HADFSR.TYPE is set to 0b01 ("Error on Response").

Configurations affected

The processor must be configured with a signal integrity bus protection scheme.

Conditions

This erratum always occurs when the bus protection scheme detects an error on the write response channel of the AXIM bus port (**BRESPMx** signal).

Implications

If this erratum occurs, the processor takes the asynchronous abort correctly and sets the ADFSR or HADFSR registers correctly. However, the DFSR or HSR registers are set incorrectly. There are two possibilities:

- If the abort is taken to EL1, DSFR.STATUS is set to 0b011001 ("Slave error interrupt from a parity or ECC error on memory access") and DFSR.ExT is set to 0b0 ("External abort marked as decode error").
- If the abort is taken to EL2, HSR.DFSC is set to 0b011001 ("Slave error interrupt from a parity or ECC error on memory access") and HSR.EA is set to 0b0 ("External abort marked as decode error").

Workaround

This erratum can be avoided by the exception handling software, which can clearly identify whether the erratum conditions are met. Specifically:

- All asynchronous data aborts taken to EL1 with DFSR.STATUS set to 0b011001 ("Slave error interrupt from a parity or ECC error on memory access"), ADFSR.PORT set to 0b000 ("AXI") and ADFSR.TYPE set to 0b01 ("Error on Response") should be treated as signal integrity errors on the write response channel of the AXIM bus port.
- All asynchronous data aborts taken to EL2 with HSR.DFSC set to 0b011001 ("Slave error interrupt from a parity or ECC error on memory access"), ADFSR.PORT set to 0b000 ("AXI") and ADFSR.TYPE set to 0b01 ("Error on Response") should be treated as signal integrity errors on the write response channel of the AXIM bus port.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

1412115
False DCLS errors can be generated in configurations without bus protection

Status

Affects: Cortex-R52
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r1p0, r1p1, r1p2. Fixed in r1p3.

Description

The Cortex-R52 processor can be configured with *Dual-Core Lock-Step* (DCLS) or Split/Lock. In these configurations, most of the processor logic is instantiated twice and driven from the same inputs to provide redundancy. The outputs of the two copies of the logic are then compared to detect errors. In the DCLS configuration, all logic is duplicated, except all RAM instances and ETM logic.

Because of this erratum, when Cortex-R52 is configured without bus protection and a write is performed through the *AXI master* (AXIM) interface, the unused byte lanes can be compared causing an error to be signalled.

Configurations affected

This erratum affects DCLS or Split/Lock configurations which do not include bus signal integrity protection. That is, the configuration file has LOCK_STEP != 0, and BUS_PROTECTION == 0.

Conditions

This erratum happens when all of the following conditions occur:

- A store is performed through the AXIM interface.
- The write data uses a data buffer within the store unit for which not all of the bytes have previously been initialized by earlier stores.
- The processor does not include bus protection.
- The uninitialized bytes have differing values in the two copies of the store unit.
- The core to top-level DCLS comparators are enabled for the appropriate core. That is, for core *n*, **DCLSCOMPIN[14*n]** is HIGH.

Implications

When this erratum occurs, false errors are signaled on **DCLSCOMPOUT[(14*n)+1:(14*n)]** for core *n*.

The implications of this depend on how the system handles DCLS errors which the processor signals. The system typically treats the false error that is signaled as an indication of a fault and the response might involve shutting down the system to enter a safe state. The erratum is expected to impact the availability of the processor.

Note: This erratum is not observable in a normal simulation of the processor RTL because of the Verilog semantics of X-propagation.

Workarounds

This erratum can be avoided by ensuring that the AXIM store data buffers are fully initialized before they are used. The following code is suggested for this purpose and it should be executed from EL2 after the register file has been initialized but before any other writes through the AXIM:

```
ldr   r8, =SCRATCH_AXIM_ADDRESS
ldr   r0, =0x5bacce55
mcr   p15, #0x4, r0, c15, c0, #1
stm   r8!, {r0-r7}
stm   r8, {r0-r7}
dsb
mov   r3, #0x10000004
mcr   p15, #0x4, r3, c15, c0, #1
stm   r8, {r0-r7}
mov   r3, #0x10000000
mcr   p15, #0x4, r3, c15, c0, #1
movt  r0, #0x5000
mcr   p15, #0x4, r0, c15, c0, #1
```

Where `SCRATCH_AXIM_ADDRESS` is the address of a 16-byte aligned, Normal memory location accessed via the AXIM which has 64 bytes of space which can be overwritten.

Note:

- If you have **CFGINITREG** tied HIGH, the register file is automatically initialized when the processor leaves reset.

This workaround causes the TESTR1 unlock event, **ERREVENTn[25]** to be asserted when it executes.

827402
A stream of data cache maintenance operations can lead to a deadlock

Status

Affects: Cortex-R52
Fault Type: Programmer, Category B
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor has a data cache controller that is capable of caching accesses for both AXIM and flash memory. The data cache is also capable of executing cache maintenance operations and handling ECC errors. Sometimes, executing a stream of data cache maintenance operations might cause the processor to deadlock.

Configurations affected

To exercise this erratum, all of the following must hold:

1. The processor must be built with a data cache (`L1_DCACHEn` is 1).
2. The processor should be able to access AXIM and flash (`CFGFLASHIMP` is 1).
3. Cache segregation should be setup to have at least one way allocated to both flash and AXIM.

Conditions

This erratum occurs when the following happens in order:

1. The core executes either a load or store to one memory type (AXIM or flash).
2. The core executes a stream of at least four DCIMVAC or DCISW operations to a different memory type (flash or AXIM). Sometimes, under the presence of ECC errors, it might be sufficient to execute a stream of two of these operations instead of the four.

Implications

If the conditions for this erratum are satisfied, the processor might deadlock.

Workaround

This erratum can be worked around by executing a data full barrier before starting data cache maintenance operations. If the processing element is currently executing in EL1 and there is no access to a higher privilege level, then a data synchronization barrier can be used in place of a data full barrier.

2743885
GICR_TYPER registers do not reflect affinity configuration inputs

Status

Affects: Cortex-R52
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r1p0, r1p1, r1p2, r1p3, r1p4. Open

Description

The Cortex-R52 processor has an identifier which is set via the processor top-level configuration inputs **CFGMPIDRAFF1** and **CFGMPIDRAFF2**. These are referred to as the Aff1 and Aff2 identifier fields. The Aff3 identifier field is always zero. These affinity fields are reported through the MPIDR system register for each core and the GICR_TYPER register for each associated Redistributor. Because of this erratum, the GICR_TYPER.Aff1 and GICR_TYPER.Aff2 fields are always zero.

Conditions

This erratum occurs if either of the **CFGMPIDRAFF1** or **CFGMPIDRAFF2** inputs is nonzero, and one of the GICR_TYPER registers are read.

Implications

An operating system running on a particular core might compare the affinity values of the core from the MPIDR with the affinity values found in the GICR_TYPER of the various Redistributors in order to identify which Redistributor is associated with the core. This erratum will cause all such comparisons to mismatch which might lead to a failure or hang during booting.

Workaround

Software which identifies Cortex-R52 Redistributors by matching affinity values should compare the Aff0 value only and ignore the Aff1 and Aff2 values.

Note

The Redistributors' registers within a Cortex-R52 processor cluster are only accessible to the cores within that cluster and all the Redistributors within a cluster are associated with cores within the cluster or the GIC export port for that cluster. Therefore, it is only necessary to compare the Aff0 values to precisely identify the correct Redistributor.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

Category B (rare)

844709
Advanced SIMD integer multiply instructions in IT block might cause data corruption

Status

Affects: Cortex-R52
Fault Type: Programmer, Category B Rare
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor can optionally be configured to support Advanced SIMD instructions. These instructions are available when the processor is either in the A32 or the T32 execution state.

The Cortex-R52 processor also supports conditional execution of instructions. In the T32 execution state, when the processor executes an `IT` instruction, up to four instructions at a time can become conditional.

Because of this erratum, if certain Advanced SIMD integer multiply instructions are executed conditionally, the processor might produce the wrong result for subsequent dependent Advanced SIMD integer multiply-accumulate instructions.

Configurations affected

The processor must be configured with Advanced SIMD support.

Conditions

This erratum occurs when the processor executes a sequence of instructions in the T32 execution state. The sequence must comprise of the following instructions:
1. An `IT` instruction.
2. An optional instruction that belongs to the IT block.
3. An Advanced SIMD producer instruction that belongs to the IT block.
4. An optional instruction that belongs to the IT block.
5. An Advanced SIMD consumer instruction that might or might not belong to the IT block.

Additionally, all the following conditions must hold:

- The producer instruction (3) must be one of the following:
  - VMLA (by scalar).
  - VMLA (integer).

  - VMLAL (by scalar).
  - VMLAL (integer).
  - VMLS (by scalar).
  - VMLS (integer).
  - VMLSL (by scalar).
  - VMLSL (integer).
  - VMUL (by scalar).
  - VMUL (integer).
  - VMULL (by scalar).
  - VMULL (integer).
- The consumer instruction (5) must be one of the following:
  - VMLA (by scalar).
  - VMLA (integer).
  - VMLAL (by scalar).
  - VMLS (by scalar).
  - VMLS (integer).
  - VMLSL (by scalar).
- The <Qd> or <Dd> destination operand of the producer instruction must match the <Qd> or <Dd> source operand of the consumer instruction.
- The <dt> data type of the producer instruction must be compatible with the <dt> data type of the consumer instruction.
- Both the producer and the consumer instruction must be of the integer variant and not of the floating-point or the polynomial variants.
- The producer instruction must fail its condition code check.
- The consumer instruction must either:
  - Not belong in the IT block and therefore, execute unconditionally.
  - Belong in the IT block and pass its condition code check, because either:
    - The consumer has the opposite condition code as the producer.
    - The consumer has the same condition code as the producer, but a flag-setting instruction executes successfully in (4) and changes the condition flags.

Even though all the conditions above might hold, the erratum does not always occur. Specifically, any of the following conditions might affect its occurrence:

- The timing of the instructions in the processor pipeline.
- The arrangement of the instructions in dual-issuing pairs.

Note
The ARM architecture deprecates the conditional execution of any instruction encoding provided by Advanced SIMD that is not also provided by floating-point. The architecture strongly recommends that T32 Advanced SIMD instructions are never included in an IT block.

Implications

If this erratum occurs, the processor incorrectly forwards parts of the resulting value of the <Qd> or <Dd> operand from the producer to the consumer instruction, although the producer instruction has failed its condition code check. Therefore, the result of the consumer instruction is wrong.

Workaround

A workaround is not expected to be required in most cases. This is because the erratum conditions require use of deprecated code.

If a workaround is required, the erratum can be avoided by disabling dual-issue. Dual-issue can be disabled by setting CPUACTLR.DIDIS to 0b1. Disabling dual-issue has a significant performance impact so this workaround is not recommended for most systems. If the erratum is encountered, it is preferable to work around it by avoiding the conditions described.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

Category C

752456
Store-exclusive to Shareable Device memory might cause core to deadlock/livelock

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor has an AXI-slave port which enables system masters (which might include other Cortex-R52 cores) to access the private TCMs belonging to a core. Because of this erratum, a store from one core might be blocked from completing if it is followed by a store exclusive to Device memory. Depending on whether or not Fast Interrupts are enabled, the core is either livelocked or deadlocked.

Configurations affected

The erratum only affects the processor if the Memory Reconstruction Port (MRP) is enabled, that is **CFGMRPEN** is HIGH immediately after reset.

Additionally, the system must be designed so an AXI write transaction generated by a core is blocked from completing until a write transaction (from another master) to the same core's private TCM completes. For example, a processor configured with more than one core in a system which enables one core to access another core's private TCMs through its main AXI-master or LLPP ports and the AXI-slave port. However, the same type of dependency can also be created by the interconnect and apply to a single-core processor.

Conditions

The erratum occurs when the local exclusive monitor of the core is in the locked state, and the following sequence of instructions occurs:

1. An external agent (which might be another Cortex-R52 core) initiates a write to the core's TCMs.
2. The core initiates a store which gets blocked from completion by the write described in [1]. One way in which this might occur is if the store is to a memory location that is mapped to the processor's AXI-slave port, that is, it is a store to the private TCM of another core in the same cluster, and it arrives at the AXI-slave port after [1].
3. The core executes a store to a Device memory location.
4. The core executes a Shareable Device store-exclusive operation.

It is not necessary for these operations to be executed back-to-back, they can be interleaved with other operations, although some types of operations prevent the erratum from occurring.

The erratum can occur if the accesses in [2] and [3] are the same, which is a Device store that gets blocked from completion by the write described in [1].

Implications

If this erratum occurs, the core deadlocks so it does not make any forward progress. If Fast Interrupts are enabled (HCTLR.FI is set), then the core is only livelocked and takes an interrupt if permitted. After handling the interrupt, execution of the same piece of code might make forward progress depending on the presence and timing of accesses to the TCMs from external agents.

Workaround

The erratum can be avoided if Normal memory is used for exclusives. This enables the core to execute the store exclusive [4] without depending on the completion of the earlier Device store [3]. If using the Device attribute is essential, then a data synchronization barrier (`DSB`) can be used before the store exclusive instruction [4]. Using either Normal memory attribute or a data synchronization barrier as a workaround for the conditions of the erratum should normally not affect the performance of executing the semaphore.

779430
Bus timeout combined with online MBIST entry might lead to deadlock

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor includes a bus timeout register (BUSTIMEOUTR), that can be programmed to monitor whether the AXI or Flash master ports are taking too long to respond. If this happens, the corresponding port is considered broken and the system is notified through a top-level timeout pin. Additionally, if a critical access for an instruction fetch is pending on the broken port, the bus timeout register can be programmed to make the core take a synchronous abort. If the software is written so that the handler code depends only on TCM memories, the processor is expected to make forward progress by entering an exception handler because of the abort.

The Cortex-R52 processor also supports online MBIST for its L1 cache memories. For the instruction cache, when the online MBIST controller sends a request to the core to enter online MBIST mode, the core first ensures that it is safe to begin online MBIST and then allows the MBIST controller to continue.

Because of this erratum, if a bus port is about to time out and an online MBIST request for the instruction cache is received before the timeout occurs, the core might deadlock.

Configurations affected

The processor must be configured with an L1 instruction cache.

Conditions

This erratum occurs if all of the following happen in order:

1. BUSTIMEOUTR is programmed to generate aborts for a certain bus port (AXI Master or Flash).
2. The front end of the core fetches instructions from this bus port.
3. This bus port does not respond to some of the outstanding requests.
4. The MBIST controller requests the start of online MBIST in the instruction cache.
5. The front end of the core fetches an instruction from this bus port and depends on it (critical access).

Implications

If this erratum occurs, the processor does not take a synchronous abort for the access in [5]. Also, the core deadlocks and remains waiting for the bus port to respond. The MBIST controller is also unable to enter online MBIST for the instruction cache.

In spite of the implication mentioned, the system will be notified of the bus port timeout through the top-level pins **ERREVENTx[13]** or **ERREVENTx[14]**, for AXI Master or Flash respectively. Also, if the exception handler code relies only on TCM memories, the core is able to break the deadlock and execute the handler if interrupts are enabled and an interrupt with sufficient priority is received.

Workaround

This erratum can be avoided if either online MBIST for the instruction cache is not used, or the BUSTIMEOUTR register is not programmed.

If the workaround mentioned is not suitable, software can ensure that when the request to enter online MBIST for the instruction cache is given, the core does not fetch any instructions from the instruction cache. For example, the core can execute a `WFI` instruction and the interrupt controller can ensure that no interrupt has sufficient priority except for the one which indicates the end of online MBIST.

If both the above workarounds are not suitable, the system designer can connect the **ERREVENTx[13]** and **ERREVENTx[14]** top-level pins to an interrupt source of sufficient priority to break the deadlock if it occurs.

763109
Undefined exception might be taken on an instruction cache ECC error

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor handles instruction cache ECC errors by invalidating the cache line and by re-fetching the instruction from the main memory. The ECC error is logged into the ICERR* system registers and reported on the appropriate top-level signals (**ERREVENTx**, **PRIMEMERR*x**, **SECMEMERR*x**).

Due to this erratum, under certain conditions, an instruction cache ECC error might cause the processor to take an Undefined exception instead. In such cases, the ECC error might not get logged or reported.

Configurations affected

This erratum only affects the processor when it is configured with an L1 instruction cache controller and RAM protection is implemented in the RAM integration layer.

Conditions

This erratum occurs only when all of the following are true:

- The instruction cache is activated (SCTLR.I = 1 or HSCTLR.I = 1).
- RAM protection is enabled (MEMPROTCTLR.EN = 1).
- The `IT` instruction functionality is partially disabled by setting SCTLR.ITD = 1 or HSCTLR.ITD = 1.
- The processor executes a valid T32 `IT` instruction which specifies an IT block of one subsequent instruction.
- The fetch of the instruction within this IT block encounters a single or double-bit error in the instruction cache.

Implications

The ECC error is not going to be logged into the ICERR* registers and it is not going to be reported by any of the **ERREVENTx**, **PRIMEMERR*x**, **SECMEMERR*x** pins. If the instruction within the IT block was legal, that is, it was not meant to cause an Undefined exception because of SCTLR.ITD = 1 or HSCTLR.ITD = 1, then the program flow is be also affected. The processor takes an Undefined exception upon executing the instruction.

Workaround

Software must not set SCTLR.ITD = 1 or HSCTLR.ITD = 1.

818201
Accesses to direct-mapped EL2-controlled MPU registers are not unallocated when no programmable regions are configured

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor can be configured without EL2-controlled MPU regions or with 16 programmable EL2-controlled MPU regions. Direct accesses to the EL2-controlled MPU base and limit system registers should behave in the following way:

- When 16 programmable EL2-controlled MPU regions are configured, the processor provides direct access to HPRBAR0-HPRBAR15 and HPRLAR0-HPRLAR15, and treats direct accesses to HPRBAR16-HPRBAR31 and HPRLAR16-HPRLAR31 as unallocated.
- When no programmable EL2-controlled MPU regions are configured, the processor treats direct accesses to HPRBAR0-HPRBAR31 and HPRLAR0-HPRLAR31 as unallocated.

Because of this erratum, direct accesses to HPRBAR0-HPRBAR15 and HPRLAR0-HPRLAR15 are permitted.

Configurations affected

The processor must be configured without programmable EL2-controlled MPU regions.

Conditions

This erratum occurs when software running at EL2 tries to read or write one of HPRBAR0-HPRBAR15 or HPRLAR0-HPRLAR15.

Implications

If this erratum occurs, the processor provides direct access to HPRBAR0-HPRBAR15 and HPRLAR0-HPRLAR15. The accesses are RAZ/WI. This means that software cannot use the success or failure of such an access to determine whether or not the corresponding region is present. In addition, any bug in software which causes it to erroneously access one of these registers will not be trapped.

Workaround

There is no workaround.

1328481 The debugger view of the number of cores might be incorrect when switching between Split and Lock modes

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r1p0, r1p1, r1p2, r1p3, r1p4. Open.

Description

The Cortex-R52 processor can be configured with Split/Lock, which allows the number of cores that the processor presents to vary, under Cold reset, between Split mode and Lock mode. The debugger uses the entries in the Cortex-R52 debug ROM table to determine the number and offsets of the cores in the processor. The processor ROM table always indicates the current number of cores because of this erratum. If a debugger reads the ROM table and the mode is subsequently switched, the information that it reads is incorrect.

Configurations Affected

This erratum affects the Cortex-R52 processor when configured with Split/Lock.

Conditions

1. The SoC dynamically switches the processor between Split and Lock mode under Cold reset.
2. The SoC does not reset its debug logic when resetting the processor.
3. The debugger performs debug accesses.

Implications

Debug accesses might not work as expected because:

- A debugger connected to a part that boots up in Lock mode and switches to Split mode is not aware of all the accessible cores.
- A debugger connected to a part that boots up in Split mode and switches to Lock mode might attempt to access cores that are now redundant. Such accesses receive an error response.

Workaround

To ensure a debugger has a complete description of the system, the ROM table must be read following each reset of all cores in the Cortex-R52 processor. A reset of a core can be detected by reading EDPRSR.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

857511
Debug not entering Memory Access mode without setting EDSCR.ERR

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor supports entering debug Memory Access (MA) mode through the APB interface. This might fail when an instruction previously executed through the APB interface has not yet completed.

Configurations affected

This erratum affects all processor configurations.

Conditions

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

Implications

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

Workaround

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

1050715
Cross trigger interrupt generation does not work using CTIAPPPULSE register

Status

Affects: Cortex-R52
Fault Type: Category C
Fault Status: Present in r0p0, r1p0, r1p1. Fixed in r1p2.

Description

The Cortex-R52 processor includes a *Cross Trigger Interface* (CTI) for each core which can generate trigger events to the core or its associated *Embedded Trace Macrocell* (ETM). One such event, generated by trigger output 2, is the CTI interrupt which is connected to a *Private Peripheral Interrupt* (PPI) on the *Generic Interrupt Controller* (GIC) for interrupting the execution of the core. The GIC expects this interrupt signal to be held until it is cleared by software but because of this erratum, this does not always occur.

Configurations affected

All configurations are affected.

Conditions

The CTI is enabled, trigger output 2 is enabled for a channel event in CTI Input Channel to Output Trigger Enable Register 2, CTIOUTEN2, and an event is generated on the corresponding channel by one of the following conditions.

1. A write to CTI Application Pulse Register, CTIAPPPULSE, with the bit corresponding to the enabled channel set.
2. The corresponding channel input is asserted HIGH for a small number of cycles.
3. A trigger input which is enabled for the corresponding channel in CTI Input Trigger to Output Channel Enable Registers, CTIINEN0-7, is asserted HIGH for a small number of cycles.

Additionally, the GIC interrupt corresponding to the CTI interrupt (INTID24) must be enabled or subsequently become enabled in the core

Implications

Because the CTI interrupt is asserted for only one or a small number of cycles, the core does not recognize the interrupt and an exception is not taken. For larger numbers of cycles, the interrupt might be recognised and an exception might be taken but this depends on the relative timing of various factors. Therefore, it is not reliable.

Workaround

Work around the erratum by doing either of the following.

1. Using CTI Application Trigger Set Register, CTIAPPSET, to generate the appropriate channel event instead of CTIAPPPULSE. If this workaround is used, the channel event must be cleared after the interrupt has been handled using the CTI Application Trigger Clear Register, CTIAPPCLEAR.
2. Instead of routing the channel event to a GIC PPI through the CTI interrupt trigger output, route it to an enabled channel output (**CTMCHOUT**), connect this to one of the **SPI** GIC inputs and configure the relevant interrupt to be edge-sensitive in the GIC.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

904856
An MBIST request in the middle of pending AXIS requests might cause the core to deadlock

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0 and r1p0. Fixed in r1p1.

Description

The Cortex-R52 processor has an AXIS port that enables cores in the Cortex-R52 processor to access private TCMs that belong to a core. The Cortex-R52 processor also has the ability to serve an MBIST request to one of caches or TCMs. As a result of this erratum, a linefill followed by a *Data Full Barrier* (DFB) from core A might be blocked from completing if it is followed by:

- A write from core B to TCMs that belong to core A.
- An MBIST request to core A.

Configurations affected

The erratum affects all Cortex-R52 configurations. In addtion to the Cortex-R52 configurations, the system must be designed so that an AXI read/write burst generated by a core is presented as single AXI transactions on the AXIS port.

Conditions

The erratum occurs under the following sequence of events:

1. A Cortex-R52 core A initiates a read to TCMs that belong to another core B through the AXIS port. This read must generate a linefill.
2. Core A executes a DFB following the read.
3. Core B initiates a store to the TCMs that belong to core A. Core A must observe the store after the read access that core A initiates in the first step.
4. Core B receives an MBIST request to access any internal memory (caches or TCMs).

It is necessary for these operations to occur in the prescribed order to exercise the erratum.

Implications

If this erratum occurs, the core deadlocks and does not make any progress.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

Workaround

To avoid this erratum:

- Use normal non-cacheable attributes to access private TCMs.
- Enable Fast Interrupts, that is, set HCTLR.FI. This makes it possible to come out of the deadlock and continue with normal execution.

857196
Performance Monitors count might be inaccurate for event 0x01B (INST_SPEC)

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor implements the Performance Monitors Extension. The PMU event, INST_SPEC (0x01B) counts instructions that the processor speculatively executes. The definition of speculatively executed is IMPLEMENTATION DEFINED. Cortex-R52 is an in-order processor and defines the INST_SPEC event to count instructions that are architecturally executed, that is, to count the same number of instructions as the INST_RETIRED event (0x008).

Because of this erratum, the number of instructions the INST_SPEC event counts might be wrong.

Configurations affected

This erratum is present in all configurations.

Conditions

This erratum occurs when the software programs an event counter to count the INST_SPEC event.

Implications

If this erratum occurs, the processor might count the wrong number of instructions for the INST_SPEC event.

Workaround

Software should not use the INST_SPEC event. Instead, the software should replace it with the INST_RETIRED event.

925639
Debugger read transactions to Device Affinity Registers of CoreSight components return incorrect values

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0 and r1p0. Fixed in r1p1.

Description

The Cortex-R52 processor includes the following CoreSight components for each configured core:

- A debug block.
- A *Performance Monitoring Unit* (PMU).
- A *Cross Trigger Interface* (CTI).
- An *Embedded Trace Macrocell* (ETM).

A debugger can access the CoreSight component registers through the debug APB slave interface. This erratum causes the Device Affinity Registers (DEVAFF0 and DEVAFF1) for all CoreSight components in the Cortex-R52 processor to hold incorrect values. These values do not match the Multiprocessor Affinity Register (MPIDR) of the core that the component connects to.

Configurations affected

This erratum affects all configurations.

Conditions

This erratum occurs when a debugger reads one of the Device Affinity Registers through the debug APB slave interface.

Implications

A debugger cannot use the Device Affinity Registers to determine whether two components have an affinity with each other.

Workaround

The debugger can do either of the following:

- Perform topology detection.

- Manual configuration.

Note:
The connection between components inside the Cortex-R52 processor is fixed.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

865776
Repeated asynchronous exceptions might generate false livelock indication

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r1p0. Fixed in r1p1.

Description

The Cortex-R52 processor supports three asynchronous exceptions.

- IRQ exceptions.
- FIQ exception.
- Asynchronous aborts.

When an asynchronous exception is unmasked and taken, the processor interrupts the execution flow, updates the execution state, and jumps to an address that the corresponding entry in the vector table specifies.

The processor also offers some livelock detection that might occur in the presence of multiple hard memory errors. The processor detects whether an exception of the same type is consecutively taken 20 times without any instructions retiring in between. In this case, the processor output, **ERREVENTx[24]**, is pulsed to indicate to the system that the processor is potentially in livelock.

Because of this erratum, the processor might falsely indicate that it is in livelock when handling consecutive asynchronous exceptions.

Configurations affected

This erratum affects all processor configurations.

Conditions

The following conditions must occur 20 times in succession.

1. Execution of either a mispredicted conditional, or mispredicted indirect branch instruction, or both.
2. At the same cycle as the first condition, an asynchronous exception is taken, interrupting the branch instruction.
3. An arbitrary number of instructions are executed without the processor taking any synchronous exceptions.

Implications

Although the processor is making forward progress, the **ERREVENTx[24]** output is pulsed and the system is notified that the processor is potentially in livelock.

Workaround

Reentrant exception handlers can be modified to avoid placing conditional and indirect branches immediately after the `CPS` or `MSR` instructions which unmask subsequent asynchronous exceptions.

This erratum is not expected to affect non-reentrant exception handlers. Asynchronous exception unmasking happens during the `ERET` instruction execution. It is unlikely that the next asynchronous exception interrupts either a mispredicted conditional, or mispredicted indirect branch instruction, or both in non-handler code 20 consecutive times.

857509
Mismatch between EDPRSR.SR and EDPRSR.R

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor provides access to the EDPRSR through the APB interface. If this access is done at the same time as the core leaves a Warm reset, then a subsequent read of the same register will read an incorrect value of the SR field.

Configurations affected

This erratum affects all processor configurations.

Conditions

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

Implications

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

Workaround

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

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

836865
An EXOKAY response to a read on the LLPP interface indicates the wrong ADFSR value

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

The Cortex-R52 processor is capable of detecting and reporting some errors received on the external buses. One possible error is a non-exclusive read transaction on the LLPP master interface receiving an EXOKAY response. The intended behavior is Cortex-R52 taking a synchronous external abort and reporting "No auxiliary error classification" in the *Auxiliary Data Fault Status Register* (ADFSR). However, this erratum means that the ADFSR reports "Error on TCM, Cache or bus data".

Configurations affected

This erratum only affects integrations that have memory or peripherals attached to the LLPP interface and **CFGLLPPIMP** correspondingly tied HIGH to enable its usage.

Processor configurations which include signal integrity bus protection are capable of detecting this type of error if it results from a single or double-bit error in the transmission of the response, so this erratum is not considered to affect such configurations.

Conditions

This erratum occurs when the core performs a read through the LLPP interface and receives an EXOKAY response. This might be a result of a fault somewhere in the system.

Implications

If an EXOKAY response is received on the LLPP master interface, Cortex-R52 incorrectly reports "Error on TCM, Cache or bus data" in the ADFSR. If software accesses the DFSR and ADFSR in order to understand the fault that occurred, then the response is incorrectly classified as a double-bit (Fatal) error on **RDATAPx**.

Workaround

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

If detection of this type of error is required without the use of bus protection, then detection logic could be implemented outside the processor to transform any EXOKAY value on **RRESPPx** into a SLVERR value.

ARM recommends that if general protection of random errors of this type is required, then the processor (and system) should be configured with the bus protection features.

848521
The ETM can enter a low power state when active, potentially corrupting trace when TRCEVENTCTL1R.LPOVERRIDE is set

Status

Affects: Cortex-R52
Fault Type: Programmer, Category C
Fault Status: Present in r0p0. Fixed in r1p0.

Description

If the Cortex-R52 processor executes a WFI or WFE instruction, the core that executed the instruction can be placed into a low power state. This erratum means that the low power state can be entered when the ETM is not idle, potentially corrupting trace if event trace is generated at the same time. This erratum occurs when ETM trace is enabled during the transition into the low power state and TRCEVENTCTL1R.LPOVERRIDE is set.

Configurations affected

This erratum affects all hardware configurations of Cortex-R52.

Conditions

This erratum requires the processor to execute a WFI or WFE instruction while TRCEVENTCTL1R.LPOVERRIDE is set and one of the following conditions to occur in the ETM:

- TRCPRGCTLR.EN is asserted before the low power state is entered.
- TRCOSLSR.OSLK is cleared before the low power state is entered.
- **NIDENx** and/or **DBGENx** is asserted before the low power state is entered.

Implications

The ETM might enter a low power state when TRCEVENTCTL1R.LPOVERRIDE is set. This might result in trace being corrupted if event trace is generated at the same time.

Workaround

TRCEVENTCTL1R.LPOVERRIDE must be clear if event trace can be generated during a WFI/WFE instruction.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

888346
Implementation defined Data Cache Debug Read operations can only access the first doubleword of a cacheline

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r1p0. Fixed in r1p1.

Description

The Cortex-R52 processor includes implementation defined Data Cache Debug Read instructions that allow the contents of a specified doubleword-sized cache location to be placed into a software accessible register. This erratum means that the cache line offset is ignored and all Data Cache Debug Read instructions targeting the *Level 1* (L1) data side data cache data RAMs return the first doubleword of the cacheline. MBIST accesses targeting the *Level 1* (L1) data side data cache data RAMs are not affected by this erratum.

Configurations affected

Cortex-R52 configurations that include a L1 data cache are affected.

Conditions

This erratum occurs when the core performs a Data Cache Debug Read instruction targeting the L1 data cache data RAM, for any non-zero doubleword offset.

Implications

If the Cortex-R52 processor executes a Data Cache Debug Read instruction targeting a non-zero doubleword offset in the L1 data cache data RAM, then the instruction returns the data from the first doubleword in the cache line.

Workaround

For systems that implement an MBIST controller connected to the Cortex-R52 processor, the contents of the L1 data cache data RAM can be read through MBIST read operations.

Copyright © 2022 Arm® Limited or its affiliates. All rights reserved.

Non-Confidential

2289294 DTR flags not cleared on external debugger access while leaving Debug state

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r1p0, r1p1, r1p2, r1p3, r1p4. Open.

Description

The *Data Transfer Registers* (DTRs) provide a mechanism to transfer data between an external debugger and the core. They consist of write-only registers to transmit data (DBGDTRTX_EL0 and DBGDTRTXint), read-only registers to receive data (DBGDTRRX_EL0 and DBGDTRRXint), and associated data control flags.

Due to this erratum, if these registers are accessed by the external debugger while the debug exit procedure is in progress, then the accesses will go ahead but the flow control flags will not be updated correctly.

Configurations affected

This erratum affects all configurations.

Conditions

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

1. The debugger requests a debug exit.
2. The debugger does an external access to the DBGDTRRX/ DBGDTRTX, while the debug exit is ongoing.
3. Certain microarchitectural timing conditions are met.

Implications

For an external read to DBGDTRTX, the EDSCR.TXU and EDSCR.TXfull flags will not be updated.
For an external write to DBGDTRRX, the write will be ignored and the EDSCR.RXO and EDSCR.RXfull flags will not be updated.

Workaround

There is no workaround.

2077568
ATB flush response may be delayed

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r1p0, r1p1, r1p2, r1p3. Fixed in r1p4.

Description:

The *Embedded Trace Macrocell* (ETM) supports an external flush request for each ATB bus. Under certain timing conditions, an AFREADY response from the Cortex-R52 processor may be delayed until a new ATB transfer is generated by the Cortex-R52 processor.

Configurations affected

This erratum affects all processor configurations.

Conditions

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

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

Implications

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

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

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

Workaround

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

2072382
Data ATB flush may not respond

Status

Affects: Cortex-R52
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r1p0, r1p1, r1p2, r1p3, r1p4. Open.

Description:

The *Embedded Trace Macrocell* (ETM) supports an external flush request for each ATB bus. If the ETM is active and configured with no data trace, there will be no flush response on the data ATB.

Configurations affected

This erratum affects all processor configurations.

Conditions

This erratum occurs when all of the following conditions are met:

- The ETM is enabled
- Data trace is disabled (both TRCCONFIGR.DA and TRCCONFIGR.DV are clear)
- The ETM is not in a low-power state due to WFI or WFE
- An ATB flush is requested on the data ATB bus

Implications

External trace infrastructure which is waiting for trace to be captured may stall forever (for example in a "flush and stop" scenario). Instruction trace will flush as expected, but this may not be observed at the trace capture device. Disabling the ETM or entering a low-power state will restore the expected behavior.

Workaround

To avoid this erratum:

- Enable TRCCONFIGR.DA or TRCCONFIGR.DV
- Program all of the following registers to zero if data trace is not required:
  - TRCVDARCCTLR
  - TRCVDCTLR
  - TRCVDSACCTLR

- ◦ TRCEVENTCTL1R.DATAEN

This will slightly increase power consumption, but it will not cause any data trace generation.