



## Arm Cortex-A720 (MP151)

### Software Developer Errata Notice

Date of issue: 25-May-2023

Non-Confidential

Document version: 8.0

Copyright © 2022, 2023 Arm® Limited (or its affiliates). All rights reserved.

Document ID: SDEN-2439421

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, 2023 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-A720 (MP151), 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](mailto:terms@arm.com).

# Contents

<b>rOp1 implementation fixes</b>	<b>6</b>
<b>Introduction</b>	<b>7</b>
Scope	7
Categorization of errata	7
<b>Change Control</b>	<b>8</b>
<b>Errata summary table</b>	<b>12</b>
<b>Errata descriptions</b>	<b>15</b>
Category A	15
Category A (rare)	15
Category B	16
2940794 Load crossing a 64B boundary might cause data corruption	16
2926083 Enabling the Statistical Profiling Extension might lead to deadlock	17
2844092 Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault	18
2792132 Crossing 4K SVE Non-fault loads might cause data corruption	20
2729604 An FP/SIMD Data-Processing instruction reading a General-Purpose register might cause the core to deadlock	21
2724969 Read requests generated by the L2 data prefetcher might generate MPAM information using PARTID_I/PMG_I rather than PARTID_D/PMG_D	22
2641656 Data Cache Clean operations by VA might cause the PE to deadlock	23
2621464 Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault	25
2615075 CPU might violate restrictions on instructions fetches when all stages of translation are disabled	27
2561043 Speculatively executing 4KB-crossing loads might lead to data corruption	29
2463007 Reads of CNTVCTSS_ELO might be performed speculatively	30
Category B (rare)	32
2689241 Concurrent modification and execution of instructions in the same PE might not succeed	32
Category C	34
2926087 Checked stores in precise mode might fail to report tag check fails	34
2821794 Checked load crossing 4K boundary might fail to report a tag check fault into TFSR_ELx	36
2802390 Hardware update of the Access flag or Dirty bit might cause corruption of Allocation Tags for translation table descriptors	37
2752414 CFP RCTX might fail to invalidate the specified ASID and VMID	38

2742807	PMU counters might fail to freeze on overflow	40
2740679	Some PMU events might have incorrect values	41
2738461	AMU Event 0x0011, Core frequency cycles might increment incorrectly when the core is in WFE state	43
2711875	Information in some Statistical Profiling Extension packets might be incorrect	44
2700891	An MTE checked store might report a speculative SError in case of poisoned tags	45
2694688	Read accesses to memory-mapped RAS registers might fail to return the expected value	46
2693172	The PE might not respond to APB accesses to some registers in OFF_EMU state	47
2684560	Interrupts might be taken following a Context Synchronization Event when MDSR_EL1.INTdis or EDSCR.INTdis are set	48
2659723	A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time	49
2648296	ESR_ELx for a Breakpoint exception might be incorrect if breakpoints are concurrently disabled through APB	50
2631723	RAMINDEX operations accessing the L1 Data cache might fail to return the correct result	51
2625934	UNDEFINED exceptions due to EDSCR.SDD might be prioritized over EL2 traps caused by HCR_EL2.TIDCP	52
2623527	Setting CPUECTLR2_EL1[11] might cause the PE to deadlock	54
2621369	Some PMU events might have incorrect values	55
2568091	Cross-4KB Neon/SVE load instructions might observe poison that is present on another cache line	57
2561142	Checked First-Fault Neon/SVE load instructions might take a synchronous exception instead of updating FFR upon DBE on L1 Tag and external abort from the system	58
2477399	Incorrect transition to Software-Step Active state after OSLAR_EL1.OSLK write followed by ERET	59
2464850	Information in some Statistical Profiling Extension packets might be incorrect	60
2458921	Checked stores in precise mode might fail to report tag check fails	61
2455243	Crossing 4KB load might report incorrect FFR index	63
2430467	Software-step with no instruction step after debug-exit from ELO with a bad mode in DSPSR	64
2303132	Instruction sampling bias exists in SPE implementation	65

## rOp1 implementation fixes

Note the following errata might be fixed in some implementations of rOp1. This can be determined by reading the REVIDR\_EL1 register where a value or a set bit indicates that the erratum is fixed in this part.

REVIDR_EL1[23:0] = 0x1	2724969	Read requests generated by the L2 data prefetcher might generate MPAM information using PARTID_I/PMG_I rather than PARTID_D/PMG_D
	2729604	An FP/SIMD Data-Processing instruction reading a General-Purpose register might cause the core to deadlock

Note that there is no change to the MIDR\_EL1 which remains at rOp1 but the REVIDR is updated to indicate which errata are corrected. Software will identify this release through the combination of MIDR\_EL1 and REVIDR\_EL1.

# 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:

<b>Category A</b>	A critical error. No workaround is available or workarounds are impactful. The error is likely to be common for many systems and applications.
<b>Category A (Rare)</b>	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.
<b>Category B</b>	A significant error or a critical error with an acceptable workaround. The error is likely to be common for many systems and applications.
<b>Category B (Rare)</b>	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.
<b>Category C</b>	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.

## 25-May-2023: Changes in document version v8.0

ID	Status	Area	Category	Summary
<a href="#">2844092</a>	New	Programmer	Category B	Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault
<a href="#">2926083</a>	New	Programmer	Category B	Enabling the Statistical Profiling Extension might lead to deadlock
<a href="#">2940794</a>	New	Programmer	Category B	Load crossing a 64B boundary might cause data corruption
<a href="#">2689241</a>	New	Programmer	Category B (rare)	Concurrent modification and execution of instructions in the same PE might not succeed
<a href="#">2458921</a>	Updated	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails
<a href="#">2740679</a>	Updated	Programmer	Category C	Some PMU events might have incorrect values

## 03-Mar-2023: Changes in document version v7.0

ID	Status	Area	Category	Summary
<a href="#">2463007</a>	Updated	Programmer	Category B	Reads of CNTVCTSS_ELO might be performed speculatively
<a href="#">2792132</a>	New	Programmer	Category B	Crossing 4K SVE Non-fault loads might cause data corruption
<a href="#">2623527</a>	Updated	Programmer	Category C	Setting CPUECTLR2_EL1[11] might cause the PE to deadlock
<a href="#">2738461</a>	New	Programmer	Category C	AMU Event 0x0011, Core frequency cycles might increment incorrectly when the core is in WFE state
<a href="#">2752414</a>	New	Programmer	Category C	CFP RCTX might fail to invalidate the specified ASID and VMID
<a href="#">2821794</a>	New	Programmer	Category C	Checked load crossing 4K boundary might fail to report a tag check fault into TFSR_ELx

## 13-Jan-2023: Changes in document version v6.0

ID	Status	Area	Category	Summary
<a href="#">2623527</a>	Updated	Programmer	Category C	Setting CPUECTLR2_EL1[11] might cause the PE to deadlock
<a href="#">2740679</a>	New	Programmer	Category C	Some PMU events might have incorrect values
<a href="#">2742807</a>	New	Programmer	Category C	PMU counters might fail to freeze on overflow
<a href="#">2802390</a>	New	Programmer	Category C	Hardware update of the Access flag or Dirty bit might cause corruption of Allocation Tags for translation table descriptors



**08-Sep-2022: Changes in document version v5.0**

ID	Status	Area	Category	Summary
<a href="#">2724969</a>	New	Programmer	Category B	Read requests generated by the L2 data prefetcher might generate MPAM information using PARTID_I/PMG_I rather than PARTID_D/PMG_D
<a href="#">2729604</a>	New	Programmer	Category B	An FP/SIMD Data-Processing instruction reading a General-Purpose register might cause the core to deadlock
<a href="#">2684560</a>	New	Programmer	Category C	Interrupts might be taken following a Context Synchronization Event when MDSR_EL1.INTdis or EDSCR.INTdis are set

**29-Jul-2022: Changes in document version v4.0**

ID	Status	Area	Category	Summary
<a href="#">2463007</a>	Updated	Programmer	Category B	Reads of CNTVCTSS_ELO might be performed speculatively
<a href="#">2561043</a>	Updated	Programmer	Category B	Speculatively executing 4KB-crossing loads might lead to data corruption
<a href="#">2615075</a>	Updated	Programmer	Category B	CPU might violate restrictions on instructions fetches when all stages of translation are disabled
<a href="#">2621464</a>	Updated	Programmer	Category B	Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault
<a href="#">2641656</a>	Updated	Programmer	Category B	Data Cache Clean operations by VA might cause the PE to deadlock
<a href="#">2303132</a>	Updated	Programmer	Category C	Instruction sampling bias exists in SPE implementation
<a href="#">2430467</a>	Updated	Programmer	Category C	Software-step with no instruction step after debug-exit from ELO with a bad mode in DSPSR
<a href="#">2455243</a>	Updated	Programmer	Category C	Crossing 4KB load might report incorrect FFR index
<a href="#">2458921</a>	Updated	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails
<a href="#">2464850</a>	Updated	Programmer	Category C	Information in some Statistical Profiling Extension packets might be incorrect
<a href="#">2477399</a>	Updated	Programmer	Category C	Incorrect transition to Software-Step Active state after OSLAR_EL1.OSLK write followed by ERET
<a href="#">2561142</a>	Updated	Programmer	Category C	Checked First-Fault Neon/SVE load instructions might take a synchronous exception instead of updating FFR upon DBE on L1 Tag and external abort from the system
<a href="#">2568091</a>	Updated	Programmer	Category C	Cross-4KB Neon/SVE load instructions might observe poison that is present on another cache line
<a href="#">2621369</a>	Updated	Programmer	Category C	Some PMU events might have incorrect values
<a href="#">2623527</a>	Updated	Programmer	Category C	Setting CPUECTLR2_EL1[11] might cause the PE to deadlock
<a href="#">2625934</a>	Updated	Programmer	Category C	UNDEFINED exceptions due to EDSCR.SDD might be prioritized over EL2 traps caused by HCR_EL2.TIDCP
<a href="#">2631723</a>	Updated	Programmer	Category C	RAMINDEX operations accessing the L1 Data cache might fail to return the correct result
<a href="#">2648296</a>	Updated	Programmer	Category C	ESR_ELx for a Breakpoint exception might be incorrect if breakpoints are concurrently disabled through APB

ID	Status	Area	Category	Summary
<a href="#">2659723</a>	Updated	Programmer	Category C	A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time
<a href="#">2693172</a>	New	Programmer	Category C	The PE might not respond to APB accesses to some registers in OFF_EMU state
<a href="#">2694688</a>	New	Programmer	Category C	Read accesses to memory-mapped RAS registers might fail to return the expected value
<a href="#">2700891</a>	New	Programmer	Category C	An MTE checked store might report a speculative SError in case of poisoned tags
<a href="#">2711875</a>	New	Programmer	Category C	Information in some Statistical Profiling Extension packets might be incorrect

**24-Jun-2022: Changes in document version v3.0**

ID	Status	Area	Category	Summary
<a href="#">2621464</a>	New	Programmer	Category B	Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault
<a href="#">2648296</a>	New	Programmer	Category C	ESR_ELx for a Breakpoint exception might be incorrect if breakpoints are concurrently disabled through APB
<a href="#">2659723</a>	New	Programmer	Category C	A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time

**25-May-2022: Changes in document version v2.0**

ID	Status	Area	Category	Summary
<a href="#">2615075</a>	New	Programmer	Category B	CPU might violate restrictions on instructions fetches when all stages of translation are disabled
<a href="#">2641656</a>	New	Programmer	Category B	Data Cache Clean operations by VA might cause the PE to deadlock
<a href="#">2430467</a>	New	Programmer	Category C	Software-step with no instruction step after debug-exit from EL0 with a bad mode in DSPSR
<a href="#">2464850</a>	New	Programmer	Category C	Information in some Statistical Profiling Extension packets might be incorrect
<a href="#">2477399</a>	New	Programmer	Category C	Incorrect transition to Software-Step Active state after OSLAR_EL1.OSLK write followed by ERET
<a href="#">2621369</a>	New	Programmer	Category C	Some PMU events might have incorrect values
<a href="#">2623527</a>	New	Programmer	Category C	Setting CPUECTLR2_EL1[11] might cause the PE to deadlock
<a href="#">2625934</a>	New	Programmer	Category C	UNDEFINED exceptions due to EDSCR.SDD might be prioritized over EL2 traps caused by HCR_EL2.TIDCP
<a href="#">2631723</a>	New	Programmer	Category C	RAMINDEX operations accessing the L1 Data cache might fail to return the correct result

## 08-Apr-2022: Changes in document version v1.0

ID	Status	Area	Category	Summary
<a href="#">2463007</a>	New	Programmer	Category B	Reads of CNTVCTSS_ELO might be performed speculatively
<a href="#">2561043</a>	New	Programmer	Category B	Speculatively executing 4KB-crossing loads might lead to data corruption
<a href="#">2303132</a>	New	Programmer	Category C	Instruction sampling bias exists in SPE implementation
<a href="#">2455243</a>	New	Programmer	Category C	Crossing 4KB load might report incorrect FFR index
<a href="#">2458921</a>	New	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails
<a href="#">2561142</a>	New	Programmer	Category C	Checked First-Fault Neon/SVE load instructions might take a synchronous exception instead of updating FFR upon DBE on L1 Tag and external abort from the system
<a href="#">2568091</a>	New	Programmer	Category C	Cross-4KB Neon/SVE load instructions might observe poison that is present on another cache line

# 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
<a href="#">2940794</a>	Programmer	Category B	Load crossing a 64B boundary might cause data corruption	r0p0, r0p1	Open
<a href="#">2926083</a>	Programmer	Category B	Enabling the Statistical Profiling Extension might lead to deadlock	r0p0, r0p1	Open
<a href="#">2844092</a>	Programmer	Category B	Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault	r0p0, r0p1	Open
<a href="#">2792132</a>	Programmer	Category B	Crossing 4K SVE Non-fault loads might cause data corruption	r0p0, r0p1	Open
<a href="#">2729604</a>	Programmer	Category B	An FP/SIMD Data-Processing instruction reading a General-Purpose register might cause the core to deadlock	r0p0, r0p1	Open
<a href="#">2724969</a>	Programmer	Category B	Read requests generated by the L2 data prefetcher might generate MPAM information using PARTID_I/PMG_I rather than PARTID_D/PMG_D	r0p0, r0p1	Open
<a href="#">2641656</a>	Programmer	Category B	Data Cache Clean operations by VA might cause the PE to deadlock	r0p0	r0p1
<a href="#">2621464</a>	Programmer	Category B	Values of FAR_ELx and ESR_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault	r0p0	r0p1
<a href="#">2615075</a>	Programmer	Category B	CPU might violate restrictions on instructions fetches when all stages of translation are disabled	r0p0	r0p1
<a href="#">2561043</a>	Programmer	Category B	Speculatively executing 4KB-crossing loads might lead to data corruption	r0p0	r0p1
<a href="#">2463007</a>	Programmer	Category B	Reads of CNTVCTSS_ELO might be performed speculatively	r0p0	r0p1
<a href="#">2689241</a>	Programmer	Category B (rare)	Concurrent modification and execution of instructions in the same PE might not succeed	r0p0	r0p1
<a href="#">2926087</a>	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails	r0p0, r0p1	Open

ID	Area	Category	Summary	Found in versions	Fixed in version
<a href="#">2821794</a>	Programmer	Category C	Checked load crossing 4K boundary might fail to report a tag check fault into TFSR_ELx	r0p0, r0p1	Open
<a href="#">2802390</a>	Programmer	Category C	Hardware update of the Access flag or Dirty bit might cause corruption of Allocation Tags for translation table descriptors	r0p0, r0p1	Open
<a href="#">2752414</a>	Programmer	Category C	CFP RCTX might fail to invalidate the specified ASID and VMID	r0p0, r0p1	Open
<a href="#">2742807</a>	Programmer	Category C	PMU counters might fail to freeze on overflow	r0p0, r0p1	Open
<a href="#">2740679</a>	Programmer	Category C	Some PMU events might have incorrect values	r0p0, r0p1	Open
<a href="#">2738461</a>	Programmer	Category C	AMU Event 0x0011, Core frequency cycles might increment incorrectly when the core is in WFE state	r0p0, r0p1	Open
<a href="#">2711875</a>	Programmer	Category C	Information in some Statistical Profiling Extension packets might be incorrect	r0p0, r0p1	Open
<a href="#">2700891</a>	Programmer	Category C	An MTE checked store might report a speculative SError in case of poisoned tags	r0p0	r0p1
<a href="#">2694688</a>	Programmer	Category C	Read accesses to memory-mapped RAS registers might fail to return the expected value	r0p0	r0p1
<a href="#">2693172</a>	Programmer	Category C	The PE might not respond to APB accesses to some registers in OFF_EMU state	r0p0	r0p1
<a href="#">2684560</a>	Programmer	Category C	Interrupts might be taken following a Context Synchronization Event when MDSR_EL1.INTdis or EDSCR.INTdis are set	r0p0, r0p1	Open
<a href="#">2659723</a>	Programmer	Category C	A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time	r0p0, r0p1	Open
<a href="#">2648296</a>	Programmer	Category C	ESR_ELx for a Breakpoint exception might be incorrect if breakpoints are concurrently disabled through APB	r0p0	r0p1
<a href="#">2631723</a>	Programmer	Category C	RAMINDEX operations accessing the L1 Data cache might fail to return the correct result	r0p0	r0p1

ID	Area	Category	Summary	Found in versions	Fixed in version
<a href="#">2625934</a>	Programmer	Category C	UNDEFINED exceptions due to EDSCR.SDD might be prioritized over EL2 traps caused by HCR_EL2.TIDCP	r0p0	r0p1
<a href="#">2623527</a>	Programmer	Category C	Setting CPUECTLR2_EL1[11] might cause the PE to deadlock	r0p0	r0p1
<a href="#">2621369</a>	Programmer	Category C	Some PMU events might have incorrect values	r0p0	r0p1
<a href="#">2568091</a>	Programmer	Category C	Cross-4KB Neon/SVE load instructions might observe poison that is present on another cache line	r0p0	r0p1
<a href="#">2561142</a>	Programmer	Category C	Checked First-Fault Neon/SVE load instructions might take a synchronous exception instead of updating FFR upon DBE on L1 Tag and external abort from the system	r0p0	r0p1
<a href="#">2477399</a>	Programmer	Category C	Incorrect transition to Software-Step Active state after OSLAR_EL1.OSLK write followed by ERET	r0p0	r0p1
<a href="#">2464850</a>	Programmer	Category C	Information in some Statistical Profiling Extension packets might be incorrect	r0p0	r0p1
<a href="#">2458921</a>	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails	r0p0, r0p1	Open
<a href="#">2455243</a>	Programmer	Category C	Crossing 4KB load might report incorrect FFR index	r0p0	r0p1
<a href="#">2430467</a>	Programmer	Category C	Software-step with no instruction step after debug-exit from EL0 with a bad mode in DSPSR	r0p0	r0p1
<a href="#">2303132</a>	Programmer	Category C	Instruction sampling bias exists in SPE implementation	r0p0, r0p1	Open

# Errata descriptions

## Category A

There are no errata in this category.

## Category A (rare)

There are no errata in this category.

## Category B

2940794

### Load crossing a 64B boundary might cause data corruption

#### Status

Fault type: Programmer Category B  
Fault status: Present in r0p0, and r0p1. Open.

#### Description

Load crossing a 64 Bytes boundary might cause data corruption.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum can occur when a load instruction is executed and all of the following conditions are met:

- The load instruction reads at least 32 bytes of data to non general-purpose registers
- The access crosses a 64 bytes boundary
- There are one or more older writes that access the same 32 bytes (or more) of data

#### Implications

If the previous conditions are met, and under very specific timing conditions, the load instruction might return incorrect data.

#### Workaround

This erratum can be avoided by setting CPUACTLR2\_EL1[37] = 1:

```
MRS x0, S3_0_C15_C1_1
ORR x0, x0, #1<<37
MSR S3_0_C15_C1_1, x0
```

Setting this bit is expected to have a negligible performance impact.



## 2926083

### Enabling the Statistical Profiling Extension might lead to deadlock

#### Status

Fault type: Programmer Category B.  
Fault status: Present in r0p0 and r0p1. Open.

#### Description

When the *Statistical Profiling Extension* (SPE) is enabled, selecting specific operations for sampling might cause the PE to deadlock.

#### Configurations Affected

This erratum affects configurations with SPE = TRUE.

#### Conditions

This erratum can occur when SPE is enabled and the following condition is met:

- A Store instruction is selected for sampling and it accesses the same cacheline as an older Store Release.

#### Implications

When the previous conditions are met, and under very unlikely timing conditions, the PE might deadlock.

#### Workaround

This erratum can be avoided by setting bits[58:57] = 0b11 in CPUACTLR\_EL1:

```
MRS x0, S3_0_C15_C1_0
ORR x0, x0, #3<<57
MSR S3_0_C15_C1_0, x0
```

## 2844092

### Values of FAR\_ELx and ESR\_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault

#### Status

Fault Type: Programmer Category B

Fault Status: Present in r0p0, and r0p1. Open.

#### Description

If an instruction (F1) is architecturally executed on a PE (PE0) while its mapping is being changed by another PE (PE1), an Instruction Abort generated on PE0 for an instruction F2 that is in program order after F1, might report ESR\_ELx and FAR\_ELx values consistent with a permission fault being taken on F1.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- PE0 architecturally executes an instruction F1.
- PE1 concurrently modifies the mapping for F1 in a way that fetching from this mapping would cause a Permission Fault.
- PE0 takes any Instruction Abort exception from any stage of translation for an instruction F2 in program order after F1.
- There is no Context Synchronization Event between the execution of F1 and F2.

#### Implications

If the previous conditions are met:

- The Prefetch Abort generated for F2 is routed consistently with the cause of the exception.
- The values of ESR\_ELx and FAR\_ELx for this exception are consistent with an Prefetch Abort caused by a Permission Fault generated by fetching from the mapping for F1 installed by P1.
- ELR\_ELx points to F2.

#### Workaround

This erratum can be avoided by setting CPUACTLR4\_EL1[11] = 1

```
MRS x0, S3_0_C15_C1_3
ORR x0, x0, #1 << 11
MSR S3_0_C15_C1_3, x0
```

This is expected to have a negligible performance impact.

## 2792132

### Crossing 4K SVE Non-fault loads might cause data corruption

#### Status

Fault Type: Programmer Category B.

Fault Status: Present in r0p0, and r0p1. Open.

#### Description

Crossing 4K SVE Non-fault loads might cause data corruption.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum can occur when a Cross4k SVE Non-fault load is executed and all of the following conditions are met:

- the access crosses a 4kB boundary
- a fault is observed on the part of the access after the 4kB boundary
- the part of the access before the 4kB boundary generates an MMU fault, and the mapping is concurrently being updated by another PE.

#### Implications

If the previous conditions are met and under very specific timing conditions, it is possible that the index reported inside First Fault Register correspond to the first active element inside the second page. Elements that are before the element reported by FFR will incorrectly be set to 0.

#### Workaround

This erratum can be avoided by setting CPUACTLR2\_EL1[26] = 1:

```
MRS x0, S3_0_C15_C1_1
ORR x0, x0, #1<<26
MSR S3_0_C15_C1_1, x0
```

Setting this bit is expected to have a negligible performance impact.

## 2729604

### An FP/SIMD Data-Processing instruction reading a General-Purpose register might cause the core to deadlock

#### Status

Fault Type: Programmer Category B

Fault Status: Present in r0p0, and r0p1. Open.

#### Description

An FP/SIMD Data-Processing instruction reading a General-Purpose register might cause the core to deadlock when executed with a sequence of floating-point division or square-root operations.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs when all the following conditions are met:

- The PE architecturally executes an FP/SIMD Data-Processing instruction reading a General-Purpose register.
- The PE executes a sequence of at least four floating-point division or square-root operations, even if these are interleaved with other instructions or executed as part of a wrong speculative path.

#### Implications

If the conditions are met, in the presence of a pipeline flush (due to a branch mispredict, an exception being taken or other microarchitectural conditions causing a pipeline flush) and under specific timing conditions, the PE can deadlock.

#### Workaround

This erratum can be avoided by setting bits[61:60] in CPUACTLR\_EL1:

```

MRS      x0, S3_0_C15_C1_0
ORR      x0, x0, #(1<<60)
ORR      x0, x0, #(1<<61)
MSR      S3_0_C15_C1_0, x0

```

Setting these bits might impact the performance of workloads heavily relying on floating-point division or square-root operations.

## 2724969

### Read requests generated by the L2 data prefetcher might generate MPAM information using PARTID\_I/PMG\_I rather than PARTID\_D/PMG\_D

#### Status

Fault Type: Programmer Category B

Fault Status: Present in r0p0 and r0p1. Open.

#### Description

When the L2 data prefetcher is enabled, read requests should generate MPAM information using PARTID\_D/PMG\_D. When the erratum occurs, the generated MPAM information is instead generated using PARTID\_I/PMG\_I.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs when the L2 data prefetcher is enabled and generates accesses in a context where generation of MPAM information is enabled.

#### Implications

When the previous conditions are met, read requests generated by the L2 data prefetcher might generate MPAM information using PARTID\_I/PMG\_I rather than PARTID\_D/PMG\_D.

#### Workaround

This erratum can be avoided by programming PARTID\_I/PMG\_I to the same values as PARTID\_D/PMG\_D.

## 2641656

### Data Cache Clean operations by VA might cause the PE to deadlock

#### Status

Fault Type: Programmer Category B.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Under specific timing conditions, the execution of a Data Cache Clean operation by *Virtual Address* (VA) to the Point of Coherency, Point of Persistence, or Point of Deep Persistence might cause the *Processing Element* (PE) to deadlock.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs under the following conditions:

- The core executes one of the following Data Cache Clean (not Invalidating) operations by VA to the Point of Coherency, Persistence, or Deep Persistence:
  - DC CGDVAC
  - DC CGDVADP
  - DC CGDVAP
  - DC CGVAC
  - DC CGVADP
  - DC CGVAP
  - DC CVAC
  - DC CVADP
  - DC CVAP
- The line is present in Dirty state in the L2 cache.
- A forwarding snoop to the same line is received before the operation completes.

#### Implications

If the previous conditions are met, under specific micro-architectural and timing conditions, the PE can deadlock.

#### Workaround

- For Clean operations to the Point of Coherency, set bit[10] of CPUACTLR2\_EL1, which changes them to Clean and Invalidate operations.
- There is no workaround for Clean operations to the Point of Persistence or Deep Persistence. These are not expected to be used on sample silicon.



## 2621464

### Values of FAR\_ELx and ESR\_ELx on an Instruction Abort might not be consistent with the instruction that generated the fault

#### Status

Fault Type: Programmer Category B

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

If an instruction (F1) is architecturally executed on a PE (PE0) while its mapping is being changed by another PE (PE1), an Instruction Abort generated on PE0 for an instruction F2 that is in program order after F1, might report ESR\_ELx and FAR\_ELx values consistent with a fault being taken on F1.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- PE0 architecturally executes an instruction F1
- PE1 concurrently modifies the mapping for F1 in a way that fetching from this mapping would cause a Permission Fault
- PE0 takes any of the following exceptions for an instruction F2 in program order after F1:
  - an Instruction Abort because of a Permission Fault
  - a Breakpoint exception, and F1 and F2 are within the same 32B-aligned, 32B granule
- There is no Context Synchronization Event between the execution of F1 and F2.

#### Implications

If the previous conditions are met:

- The Instruction Abort or Breakpoint exception generated for F2 is routed consistently with the cause of the exception.
- The values of ESR\_ELx and FAR\_ELx for this exception are consistent with an Instruction Abort caused by a Permission Fault generated by fetching from the mapping for F1 installed by P1.
- ELR\_ELx points to F2

#### Workaround

This erratum can be avoided by ensuring that break-before-make is used when transitioning a page from Executable to Non-Executable.

## 2615075

### CPU might violate restrictions on instructions fetches when all stages of translation are disabled

#### Status

Fault Type: Programmer Category B

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

CPU might violate restrictions on instructions fetches when all stages of translation are disabled.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs when all the following conditions are met:

- All stages of translation are disabled for the current execution context
- For any 32-bit-aligned, 32-bit memory block between the current architectural PC and the end of the next 4KB granule:
  - the memory block contains an AArch64 branch opcode
  - the memory block is not accessed as part of a simple sequential execution of the program
  - This might happen, for example, if there is data placed after instructions in the same 4KB granule of memory.
- PSTATE.SBS=1

#### Implications

When the above conditions are met, the PE might access memory locations as a result of instruction fetches where none of the conditions described in the Arm Architecture Reference Manual for A-profile architecture (issue H.a, section: D5.2.9 The effects of disabling a stage of address translation) are met:

- The memory location is in the same block of memory as, or in the next contiguous block of memory to, an instruction that a simple sequential execution of the program either requires to be fetched now or has required to be fetched since the last reset.
- The memory location is the target of a direct branch that a simple sequential execution of the program would have taken since the most recent of:
  - the last reset
  - the last synchronization of instruction cache maintenance targeting the address of the branch instruction

## Workaround

This erratum can be avoided by setting `PSTATE.SSBS=0` when all stages of translation are disabled.

## 2561043

### Speculatively executing 4KB-crossing loads might lead to data corruption

#### Status

Fault Type: Programmer Category B.  
Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Executing loads crossing a 4KB boundary might cause such loads to return incorrect data, leading to data corruption.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

1. The processing element (PE) speculatively executes a 4KB-crossing load R1.
2. Other rare microarchitectural conditions occur.

#### Implications

If the previous conditions are met and under very specific timing conditions, it is possible for R1 to return corrupted data.

#### Workaround

This erratum can be avoided by setting bit[26] in CPUACTLR2\_EL1:

```
MRS x0, S3_0_C15_C1_1
ORR x0, x0, #1<<26
MSR S3_0_C15_C1_1, x0
```

Setting this bit is not expected to have a significant performance impact.

## 2463007

### Reads of CNTVCTSS\_ELO might be performed speculatively

#### Status

Fault Type: Programmer Category B  
Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Reads of CNTVCTSS\_ELO might be performed speculatively with respect to older instructions other than MRS/MSR instructions.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs if the PE executes an MRS CNTVCTSS\_ELO that does not generate a trap.

#### Implications

If the previous conditions are met, the timer value might be read speculatively. In particular, when using the following sequence, the MRS of CNTVCTSS\_ELO might be performed before the LDR X1 is observed.

```
loop
    LDR X1, [X2]
    CMP X1, #1
    B.NE loop
    MRS X1, CNTVCTSS_ELO
```

#### Workaround

This erratum can be avoided by using the following sequence:

```
MOV x0, #2
MSR s3_6_c15_c8_0, x0
ISB
LDR x0, =0xd53be0c0
MSR s3_6_c15_c8_2, x0
LDR x0, =0xfffffffffe0
MSR s3_6_c15_c8_3, x0
MOV x1, #0
ORR x1, x1, #1<<0
ORR x1, x1, #3<<4
ORR x1, x1, #0xf<<6
ORR x1, x1, #1<<13
```

```
ORR x1, x1, #1<<53
MSR s3_6_c15_c8_1, x1
ISB
```

## Category B (rare)

2689241

### Concurrent modification and execution of instructions in the same PE might not succeed

#### Status

Fault type: Programmer Category B (rare).  
Fault status: Present in r0p0. Fixed in r0p1.

#### Description

When a *Processing Element* (PE) performs a sequence to modify instructions, the IC instruction in that sequence might fail to properly invalidate the instruction cache if the modified instructions were being concurrently executed.

#### Configurations affected

This erratum affects all configurations.

#### Conditions

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

- The PE executes store instructions to address A1 to modify instructions
- The PE might be concurrently, speculatively executing instructions from A1
- A1 is present in the L1 data cache
- Rare micro-architectural timing conditions occur

#### Implications

If the previous conditions are met, the PE on which the instruction modification sequence was performed might observe the previous opcodes for an indefinite amount of time. Other PEs will observe the updated opcodes.

#### Workaround

This erratum can be avoided by executing the following sequence:

```
MOV x0, #0
MSR s3_6_c15_c8_0, x0
ISB
```



```
LDR x0, =0xd503339f
MSR s3_6_c15_c8_2, x0
LDR x0, =0xffffffff3ff
MSR s3_6_c15_c8_3, x0
MOV x0, #0
ORR x0, #1<<0
ORR x0, #3<<4
ORR x0, #0xf<<6
ORR x0, #1<<20
ORR x0, #1<<31
ORR x0, #1<<50
ORR x0, #1<<13
ORR x0, #1<<21
ORR x0, #1<<43
MSR s3_6_c15_c8_1, x1
```

This sequence will cause a trap to EL3 when specific micro-architectural conditions related to the erratum are met.

The trap handler in EL3 must execute the following sequence:

```
MRS x0, ELR_EL3
ADD x0, x0, #4
MSR ELR_EL3, x0
ERET
```

## Category C

2926087

### Checked stores in precise mode might fail to report tag check fails

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0, r0p1. Open.

#### Description

When *Memory Tagging Extension* (MTE) is used in Synchronous mode, the Tag-Check-read performed by a store instruction might fail to be ordered with respect to older instructions with acquire semantics. This ordering violation might lead to the store instruction writing to memory in cases where it should have failed its tag check.

#### Configurations Affected

This erratum affects configurations with BROADCASTMTE=1.

#### Conditions

The erratum occurs when all the following conditions are met:

- MTE is enabled in precise checking mode (SCTLR\_ELx.TCF='b01).
- An instruction R1 with acquire semantics is executed. R1 can be one of:
  - Any LDAR\* or LDAP\* instruction.
  - Any SWP\*, CAS\* or LD\* Atomic instruction with acquire semantics.
- A tag-checked store instruction W3, performing a Tag-Check-read R2, is executed.
- R1 is in program order before W3.

#### Implications

When the above conditions are met, the Tag-Check-read R2 performed by W3 might be observed before R1.

- If the observed tag value does not cause a Tag Check Fault, W3 will update memory even in cases where observation in the correct order should have resulted in a tag Check Fault.
- If the observed tag value causes a Tag Check Fault, there are no implications and the *Processing Element* (PE) behaves as expected.

This will lead to a very small percentage of escapes in the tag checking logic, sometimes causing a tag check pass when it should be a tag check fail.

## Workaround

This erratum has no workaround.

## 2821794

### Checked load crossing 4K boundary might fail to report a tag check fault into TFSR\_ELx

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0, and r0p1. Open.

#### Description

A checked load crossing the 4K boundary might ignore a tag check fail on any bytes before the 4K boundary, leading to TFSR\_ELx not being updated (SCTLR\_ELx.TCF=0b10).

#### Configurations Affected

This erratum affects configurations with BROADCASTMTE=1.

#### Conditions

The *Processing Element* (PE) is running with Memory Tagging enabled in asynchronous mode (SCTLR\_ELx.TCF=0b01).

Additionally, the PE is executing the following instructions:

1. Checked load crossing 4K boundary, observing a tag check fail on any bytes before the 4KB boundary.
2. A younger load cross 4K boundary taking any type of Data Abort.
3. Both loads are speculatively executed with very specific timing conditions.

#### Implications

If the previous conditions are met, checked load (1) might not report the tag check fail result inside TFSR\_ELx, causing a minor loss in term of tag check fail detection coverage.

#### Workaround

This erratum has no workaround.

## 2802390

### Hardware update of the Access flag or Dirty bit might cause corruption of Allocation Tags for translation table descriptors

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0, and r0p1. Open

#### Description

The Allocation Tags for a descriptor might be corrupted in case of hardware update of the Access flag or Dirty bit.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs when all the following conditions are met:

1. BROADCASTMTE=1
2. The PE successfully performs an hardware update of either the Access flag or the Dirty bit
3. Specific microarchitectural conditions occur

#### Implications

If the previous conditions are met, the core will set the Allocation Tags in the 64B-aligned, 64B granule containing the translation table entry that was updated to 0.

#### Workaround

Arm expects that the majority of software will not use tag checked accesses to manipulate translation tables, and hence no workaround is required. Software using tag checked accesses can force an unchecked logical address tag when manipulating translation tables.

## 2752414

### CFP RCTX might fail to invalidate the specified ASID and VMID

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0, and r0p1. Open.

#### Description

When a *Processing Element* (PE) performs a CFP RCTX instruction from EL0 or EL1, the ASID, VMID, Secure State, or Execution Level targeted by the instruction might not be the ones specified by the architecture.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

1. A CFP RCTX might fail to invalidate predictions associated with the required ASID when all of the following conditions are met:
  - A CFP RCTX instruction is executed from EL0.
  - The specified ASID (bits[15:0] of the instruction) is different from the current ASID.
2. A CFP RCTX might fail to invalidate predictions associated with the required VMID when all of the following conditions are met:
  - A CFP RCTX instruction is executed from the EL1&0 translation regime.
  - VTCR\_EL2.VS = 1.
  - Bits[15:8] of the specified VMID differ from bits[15:8] of the current VMID.
3. A CFP RCTX might fail to invalidate predictions associated with the required Secure State and Execution Level when executed from an Exception Level different than EL0.

#### Implications

If the previous conditions are met, the PE might invalidate the Control Flow Prediction information associated with the context specified in the register argument instead of the information associated with the context required to be invalidated.

#### Workaround

Points 1 and 2 of this erratum can be avoided by trapping CFP RCTX instructions to EL2 and have the hypervisor execute the CFP instruction with the correct arguments. Arm expects that most hypervisors already operate this way.

Point 3 of this erratum can be avoided by:

- Setting HCR\_EL2.TGE=0 when executing a CFP RCTX operation meant to invalidate EL1.
- Setting SCR\_EL3.NS to the expected Secure State targeted by the CFP RCTX.
- Setting GASID=1 field when executing a CFP RCTX operation meant to invalidate all ASID.

## 2742807

### PMU counters might fail to freeze on overflow

#### Status

Fault Type: Programmer Category C.  
Fault Status: Present in r0p0, and r0p1. Open.

#### Description

PMU counters might fail to freeze on overflow.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- PMCR\_ELO.FZO=1
- MDCR\_EL2.HPMN is set to the number of implemented PMU counters
- Any of the bits in PMOVSLR\_ELO becomes set

#### Implications

When the above conditions are met, PMCR\_ELO.FZO will fail to take effect, and enabled PMU counters that have not overflowed will continue counting.

#### Workaround

This erratum can be avoided by setting MDCR\_EL2.HPMN to a value smaller than the number of implemented PMU counters.



## 2740679

### Some PMU events might have incorrect values

#### Status

Fault type: Programmer Category C  
Fault status: Present in r0p0, and r0p1. Open.

#### Description

Some *Performance Monitoring Unit* (PMU) events might be counted incorrectly and either be incremented in situations where they should not, or fail to be incremented in situations where they should.

#### Configurations affected

This erratum affects all configurations.

#### Conditions

The erratum occurs if a given PMU counter is configured to count the following event under the listed situation:

- 0x0077, CRYPTO\_SPEC
- In the following situation:
  - Instructions from the SVE2 Crypto Extensions instruction are not counted as CRYPTO\_SPEC.

The erratum occurs if a given PMU counter is configured to count one of the following events, resulting in events that might be significantly undercounted:

- 0x003D, STALL\_SLOT\_BACKEND
- 0x003F, STALL\_SLOT
- 0x002A, L3D\_CACHE\_REFILL
- 0x0058, L2D\_CACHE\_INVALID

The erratum occurs if a given PMU counter is configured to count one of the following events, resulting in events that might be significantly overcounted:

- 0x002B, L3D\_CACHE
- 0x00A0, L3D\_CACHE\_RD

#### Implications

If the previous conditions are met, the affected PMU events will be incorrect.

## Workaround

This erratum has no workaround.

## 2738461

### AMU Event 0x0011, Core frequency cycles might increment incorrectly when the core is in WFE state

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0, r0p1. Open.

#### Description

The core frequency cycles *Activity Monitor Unit* (AMU) event may not count correctly when the core is in *Wait For Event* (WFE) state and the clocks in the core are enabled.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs under the following conditions:

1. The architected activity monitor counter register 0 (AMEVCNTR00) is enabled.
2. The core executes a WFE instruction.
3. The clocks in the core are never disabled, or
4. The clocks in the core are temporarily enabled without causing the core to exit WFE state due to one of the following events:
  - A system snoop request that must be serviced by the core L1 data cache or the L2 cache.
  - A cache or Translation Lookaside Buffer (TLB) maintenance operation that must be serviced by the core L1 instruction cache, L1 data cache, L2 cache, or TLB.
  - An access on the Utility bus interface.
  - A Generic Interrupt Controller (GIC) CPU access or debug access through the Advanced Peripheral Bus (APB) interface.

#### Implications

The core frequency cycles AMU event will continue to increment when clocks are enabled even though the core is in WFE state. Arm expects this to be a minor issue as the resulting discrepancies will likely be negligible from the point of view of consuming these counts in the system firmware at the 1ms level.

#### Workaround

There is no workaround.

## 2711875

### Information in some Statistical Profiling Extension packets might be incorrect

#### Status

Fault Type: Programmer Category C.  
Fault Status: Present in rOp0, rOp1. Open.

#### Description

Some information contained in the *Statistical Profiling Extension* (SPE) packets might be incorrect.

#### Configurations Affected

This erratum affects configurations with SPE=TRUE.

#### Conditions

No specific conditions are required to hit this erratum.

#### Implications

- The Operation Type packet field CLASS[1:0] might be incorrect if the tracked operation is one of:
  - LDRAA or LDRAB instruction. The instructions are incorrectly assigned a type of 'Other', and the information presented in the corresponding SPE packet is consistent with a packet of type 'Other'.

This also affects filtering of the packets whenever filtering is configured to use some of the affected packet contents.

#### Workaround

This erratum has no workaround.

## 2700891

### An MTE checked store might report a speculative SError in case of poisoned tags

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

A Tag-Checked store instruction that observes poison during the tag check operation might report an SError even if the store is not architecturally executed.

#### Configurations Affected

All configurations with CORE\_CACHE\_PROTECTION set to TRUE are affected.

#### Conditions

The erratum occurs under the following conditions:

- The core executes a Tag-Checked store instruction.
- The store observes poison on the allocation tags while performing the tag check operation.
- The store is not architecturally executed and is discarded as a result of an older branch misprediction, interrupt, or other microarchitectural conditions causing a pipeline flush.

#### Implications

If the previous conditions are met, the core might report an SError for an instruction that is not architecturally executed.

This makes the CPU contradict the value of the ID\_AA64MMFR1\_EL1.SpecSEI, behaving as if it were 0b0001.

#### Workaround

This erratum has no workaround.

## 2694688

### Read accesses to memory-mapped RAS registers might fail to return the expected value

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Read accesses to some memory-mapped RAS registers might fail to return the expected value.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs when accessing one of the following registers through the memory-mapped interface to the RAS registers:

- ERRGSR
- ERRDEVAFF
- ERRDEVARCH
- ERRDEVID

#### Implications

When the above conditions are met, reads of the affected registers will always return 0.

#### Workaround

This erratum has no workaround.

## 2693172

### The PE might not respond to APB accesses to some registers in OFF\_EMU state

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

The *Processing Element* (PE) might not respond to APB accesses to some registers in OFF\_EMU.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- The PE is in OFF\_EMU state.
- An APB access to one of the following registers:
  - DBGBVR<n>
  - DBGBCR<n>
  - DBGWVR<n>
  - DBGWCR<n>
  - EDWAR
  - EDITR
  - EDECCR

#### Implications

When the above conditions are met, the PE stop responding to APB traffic while it remains in OFF\_EMU state.

#### Workaround

This erratum has no workaround.

## 2684560

### Interrupts might be taken following a Context Synchronization Event when MDSR\_EL1.INTdis or EDSCR.INTdis are set

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0, and r0p1. Open.

#### Description

When an interrupt is pending when executing a *Context Synchronization Event* (CSE) that synchronizes setting MDSR\_EL1.INTdis or EDSCR.INTdis to 1, the interrupt might be taken immediately after the CSE and before any subsequent instructions, even though interrupts should be disabled.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- ExternalInvasiveDebugEnabled() or ExternalSecureInvasiveDebugEnabled() is TRUE
- One of the following conditions is met:
  - the PE executes an MSR MDSCR\_EL1 setting INTdis to 1
  - an external agent sets EDSCR.INTdis to 1
- An interrupt is pending when the CSE synchronizing the change to INTDis is performed.

#### Implications

The interrupt is taken immediately after the CSE and before executing any subsequent instruction.

#### Workaround

- When INTdis is set using MSR MDSCR\_EL1, no workaround is required.
- When INTdis is set using a write to EDSCR, this erratum can be avoided by executing a CSE before existing Debug State.



## 2659723

### A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0 and r0p1. Open.

#### Description

A continuous flow of snoops from other cores combined with a continuous flow of other instructions might prevent an older store from becoming visible in finite time.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs under all of the following conditions:

- Core0 executes a store to cacheable address A.
- Core1 executes a continuous flow of loads or stores at address A.
- Core0 executes a continuous flow of loads or stores operations to another address B.

#### Implications

If the above conditions are met, the store operation executed on the Core0 might not be visible in finite time. The core itself keeps performing forward progress and stays interruptible.

#### Workaround

No workaround is deemed necessary for this erratum.

## 2648296

### ESR\_ELx for a Breakpoint exception might be incorrect if breakpoints are concurrently disabled through APB

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- The PE architecturally executes an ERET instruction that causes PSTATE.IL to be set to 1
- A breakpoint was set to hit on the target address of the ERET instruction
- An external debugger concurrently disables this breakpoint through the APB interface

#### Implications

If the above conditions are met:

- The PE generates an exception that will be routed consistently with a Breakpoint exception
- The value of ESR\_ELx for this exception is consistent with an Illegal Execution state exception
- ELR\_ELx and FAR\_ELx correctly point to the instruction that generated the exception

#### Workaround

This erratum has no workaround.

## 2631723

### RAMINDEX operations accessing the L1 Data cache might fail to return the correct result

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

RAMINDEX operations accessing the L1 Data cache might fail to return the correct result.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs when all the following conditions are met:

- a RAMINDEX operation targeting the L1 Data cache (Xt[31:24]=0x9)
- the operation targets an address with VA[5:4]='b11

#### Implications

If the previous conditions are met, the RAMINDEX operation will fail to populate the operation result registers (IMP\_DSIDE\_DATA0\_EL3 and IMP\_DSIDE\_DATA1\_EL3) with the value from the targeted RAM location.

#### Workaround

This erratum has no workaround.

## 2625934

### UNDEFINED exceptions due to EDSCR.SDD might be prioritized over EL2 traps caused by HCR\_EL2.TIDCP

#### Status

Fault Type: Programmer Category C

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

For MSR instructions to IMPLEMENTATION DEFINED registers which have access controlled by an EL3 register, UNDEFINED exceptions due to EDSCR.SDD are incorrectly prioritized over EL2 traps caused by HCR\_EL2.TIDCP.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs when all the following conditions are met:

- the PE is in the following state:
  - PSTATE.EL == EL1
  - HCR\_EL2.TIDCP == 1
  - EDSCR.SDD == 1
  - Halted() returns TRUE
- the PE executes an MSR instruction to one of the following registers:
  - IMP\_CPUACTLR2\_EL1 and ACTLR\_EL3.ACTLREN == 0
  - IMP\_CPUACTLR3\_EL1 and ACTLR\_EL3.ACTLREN == 0
  - IMP\_CPUACTLR4\_EL1 and ACTLR\_EL3.ACTLREN == 0
  - IMP\_CPUACTLR\_EL1 and ACTLR\_EL3.ACTLREN == 0
  - IMP\_CPUECTLR2\_EL1 and ACTLR\_EL3.ECTLREN == 0
  - IMP\_CPUECTLR\_EL1 and ACTLR\_EL3.ECTLREN == 0
  - IMP\_CPUPMPDPCR\_EL1 and ACTLR\_EL3.PDPEN == 0
  - IMP\_CPUPWRCTLR\_EL1 and ACTLR\_EL3.PWREN == 0

#### Implications

When the previous conditions are met, the UNDEFINED exception caused by EDSCR.SDD will be prioritized over the EL2 trap caused by HCR\_EL2.TIDCP.

## Workaround

This erratum has no workaround.

## 2623527

### Setting CPUECTLR2\_EL1[11] might cause the PE to deadlock

#### Status

Fault Type: Programmer Category C.  
Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Setting CPUECTLR2\_EL1[11] might cause the *Processing Element* (PE) to deadlock.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs if all the following conditions apply:

- PSTATE.SSBS is set to 0.
- CPUECTLR2\_EL1[11] is set to 1.

#### Implications

If the conditions are met, the PE might deadlock.

#### Workaround

This erratum can be avoided by setting CPUECTLR2\_EL1[11] to 0 using the following sequence

```
MRS x0, S3_0_C15_C1_5
BFC x0, #11, #1
MSR S3_0_C15_C1_5, x0
```

This bit resets to 0, therefore this workaround is only necessary for software that changed its default value.

## 2621369

### Some PMU events might have incorrect values

#### Status

Fault Type: Programmer Category C  
Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Some *Performance Monitoring Unit* (PMU) events might be counted incorrectly and either be incremented in situations where they should not, or fail to be incremented in situations where they should.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs if a given PMU counter is configured to count one of the following events:

- 0x0029, L3D\_CACHE\_ALLOCATE
- 0x0050, L2D\_CACHE\_RD
- 0x8148, L2D\_CACHE\_RW

The erratum occurs if a given PMU counter is configured to count one of the following events under the listed situations:

- 0x0074, ASE\_SPEC
- 0x0075, VFP\_SPEC
- 0x0077, CRYPTO\_SPEC
- 0x80E3, SVE\_INST\_SPEC
- In the following situations:
  - Advanced SIMD data-processing instructions might incorrectly be counted as VFP\_SPEC instead of ASE\_SPEC
  - FCSEL instructions might be counted as ASE\_SPEC instead of VFP\_SPEC
  - PMUL instructions from the 'Advanced SIMD three same' instruction class might be counted as CRYPTO\_SPEC instead of ASE\_SPEC
  - Instructions from the SVE2 Crypto Extensions instruction list might be counted as CRYPTO\_SPEC in addition to SVE\_INST\_SPEC

The erratum occurs if a given PMU counter is configured to count one of the following events under the listed situations:

- 0x80E7, ASE\_SVE\_INT16\_SPEC

- 0x80EB, ASE\_SVE\_INT32\_SPEC
- 0x80EF, ASE\_SVE\_INT64\_SPEC
- In the following situations:
  - FCSEL instructions might be counted
  - PMUL instructions from the 'Advanced SIMD three same' instruction class might fail to be counted

The erratum occurs if a given PMU counter is configured to count one of the following events under the listed situation:

- 0x8014, FP\_HP\_SPEC
- 0x8018, FP\_SP\_SPEC
- 0x801C, FP\_DP\_SPEC
- In the following situation:
  - FCSEL instructions might fail to be counted

## Implications

If the previous conditions are met, the affected PMU events will be incorrect.

## Workaround

This erratum has no workaround.



## 2568091

### Cross-4KB Neon/SVE load instructions might observe poison that is present on another cache line

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

A cross-4KB load might transiently observe poisoned data that is present on another cache line, leading to a spurious synchronous External abort.

#### Configurations Affected

This erratum affects configurations with `CORE_CACHE_PROTECTION=1`.

#### Conditions

The erratum can occur if the following conditions are met:

1. A Neon/SVE load instruction is executed, reading at least 32 bytes of data.
2. The load crosses a 4KB boundary, and its memory attributes are cacheable.
3. Poisoned data is present in another cache line inside of the first 4KB page.
4. Rare micro-architectural conditions occur.

#### Implications

If the above conditions occur, the load might incorrectly observe the poison information and report a Synchronous External Abort.

#### Workaround

There is no workaround.

## 2561142

### Checked First-Fault Neon/SVE load instructions might take a synchronous exception instead of updating FFR upon DBE on L1 Tag and external abort from the system

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

A checked First-Fault load might take a synchronous exception instead of updating FFR in case it observes DBE on L1 TagRam and external abort from the system.

#### Configurations Affected

This erratum affects configurations with CORE\_CACHE\_PROTECTION=1 and BROADCASTMTE=1.

#### Conditions

The erratum can occur if the following conditions are met:

1. MTE is enabled in asymmetric checking mode (SCTLR\_ELx.TCF='b11).
2. PE executes a checked Neon/SVE load observing a DBE on L1 TagRam and an external abort from the system.
3. Rare micro-architectural conditions occur.

#### Implications

If the above conditions occur, the load might incorrectly take a synchronous exception instead of silently reporting the fault inside FFR.

ESR/FAR\_ELx are correctly reported on the synchronous exception.

#### Workaround

There is no workaround.

## 2477399

### Incorrect transition to Software-Step Active state after OSLAR\_EL1.OSLK write followed by ERET

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

When executing an ERET instruction, synchronization of pending changes to OSLAR\_EL1.OSLK does not take effect correctly, resulting in a transition of Software-Step state-machine to Active state.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

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

- The *Processing Element* (PE) is executing at the Exception Level ELx
- The software-step is enabled at a lower exception level, ELy
- The following two instructions are executed:
  - MSR OSLAR\_EL1 that sets the OSLK bit to 1
    - this disables software-step at ELy
  - ERET to ELy

#### Implications

If the above conditions are met, Software-Step will incorrectly transition to Active state instead of staying in an Inactive state. The Software-Step exception can happen immediately after ERET, or after one stepped instruction, depending on the value of SPSR\_ELx.SS before ERET.

#### Workaround

This erratum can be avoided by not relying on ERET to synchronize the change to OSLAR\_EL1.OSLK, and to use an explicit ISB instruction instead. It is expected that most operating systems already do this in practice.

## 2464850

### Information in some Statistical Profiling Extension packets might be incorrect

#### Status

Fault Type: Programmer Category C  
Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Some information contained in the *Statistical Profiling Extension* (SPE) packets might be incorrect.

#### Configurations Affected

This erratum affects configurations with SPE=TRUE.

#### Conditions

No specific conditions are required to hit this erratum.

#### Implications

- The Events packet fields E[9] 'Last level cache miss', and E[8] 'Last level cache access' might be wrong if the tracked operation is one of:
  - an Atomic memory operation, LD<op> or ST<op>
  - a Swap instruction
  - a Compare and Swap instruction

This also affects filtering of the packets whenever filtering is configured to use some of the affected packet contents.

#### Workaround

This erratum has no workaround.

## 2458921

### Checked stores in precise mode might fail to report tag check fails

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0, r0p1. Open.

#### Description

When *Memory Tagging Extension* (MTE) is used in Synchronous mode, the Tag-Check-read performed by a load or store instruction might fail to be ordered with respect to older instructions with acquire semantics. This ordering violation might lead to the store instruction writing to memory in cases where it should have failed its tag check.

#### Configurations Affected

This erratum affects configurations with BROADCASTMTE=1.

#### Conditions

The erratum occurs when all the following conditions are met:

- MTE is enabled in precise checking mode (SCTLR\_ELx.TCF='b01).
- An instruction R1 with acquire semantics is executed. R1 can be one of:
  - Any LDAR\* or LDAP\* instruction.
  - Any SWP\*, CAS\* or LD\* Atomic instruction with acquire semantics.
- A tag-checked load or store instruction W3, performing a Tag-Check-read R2, is executed.
- R1 is in program order before W3.

#### Implications

When the above conditions are met, the Tag-Check-read R2 performed by W3 might be observed before R1.

- If the observed tag value does not cause a Tag Check Fault, W3 will update memory even in cases where observation in the correct order should have resulted in a tag Check Fault.
- If the observed tag value causes a Tag Check Fault, there are no implications and the *Processing Element* (PE) behaves as expected.

This will lead to a very small percentage of escapes in the tag checking logic, sometimes causing a tag check pass when it should be a tag check fail.

#### Workaround

This erratum has no workaround.

## 2455243

### Crossing 4KB load might report incorrect FFR index

#### Status

Fault Type: Programmer Category C.  
Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

Executing a crossing 4KB SVE Non-Fault or First-Fault load might lead to an incorrect value being reported to the FFR register in case on a watchpoint fault and a `tag_check_fail` or an *Error Correcting Code* ( ECC ) fault that are being observed after the 4KB boundary.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

The erratum occurs if all the following conditions are met:

- The *Processing Element* (PE) executes a 4KB-crossing SVE Non-Fault or First-Fault contiguous load.
- The load observes a watchpoint fault on a byte higher than the first one after the 4KB boundary.
- The load observes a MTE `tag_check_fail` or an ECC double bit error on a byte lower than the watchpoint, after the 4KB boundary.

#### Implications

If the above conditions are met, and under specific timing conditions, it is possible for the FFR to be updated with the position of the watchpoint fault instead of the `tag_check_fail` or ECC fault.

#### Workaround

There is no workaround.

## 2430467

### Software-step with no instruction step after debug-exit from ELO with a bad mode in DSPSR

#### Status

Fault Type: Programmer Category C.

Fault Status: Present in r0p0. Fixed in r0p1.

#### Description

On exit from Debug state, PSTATE.SS is set according to DSPSR.SS and DSPSR.M.

If DSPSR.M encodes an illegal value, then PSTATE.SS should be set according to the current Exception level. When the erratum occurs, the *Processing Element* (PE) always writes PSTATE.SS to 0.

#### Configurations Affected

This erratum affects all configurations.

#### Conditions

This erratum occurs when all the following conditions are met:

- DSPSR.M encodes an illegal value.
- DSPSR.SS is set.
- The PE exits Debug State while in ELO with Software Step enabled.

#### Implications

If the previous conditions are met, then, on exit from Debug state the PE will directly take a Software-step Exception, instead of stepping an instruction as would be expected from DSPSR.SS=1.

#### Workaround

This erratum has no workaround.



## 2303132

### Instruction sampling bias exists in SPE implementation

#### Status

Fault Type: Programmer Category C.  
Fault Status: Present in r0p0 and r0p1. Open.

#### Description

A PE that is used to perform instruction sampling using the SPE mechanism might exhibit sampling bias.

#### Configurations Affected

This erratum affects all configurations where SPE=1.

#### Conditions

1. SPE configured and utilized on the PE.

#### Implications

Software utilizing SPE might see unexpectedly high sample counts for some instructions and unexpectedly low sample counts for other instructions.

#### Workaround

This erratum has no workaround.