Arm® Cortex®-A720AE (MP170)

Software Developer Errata Notice

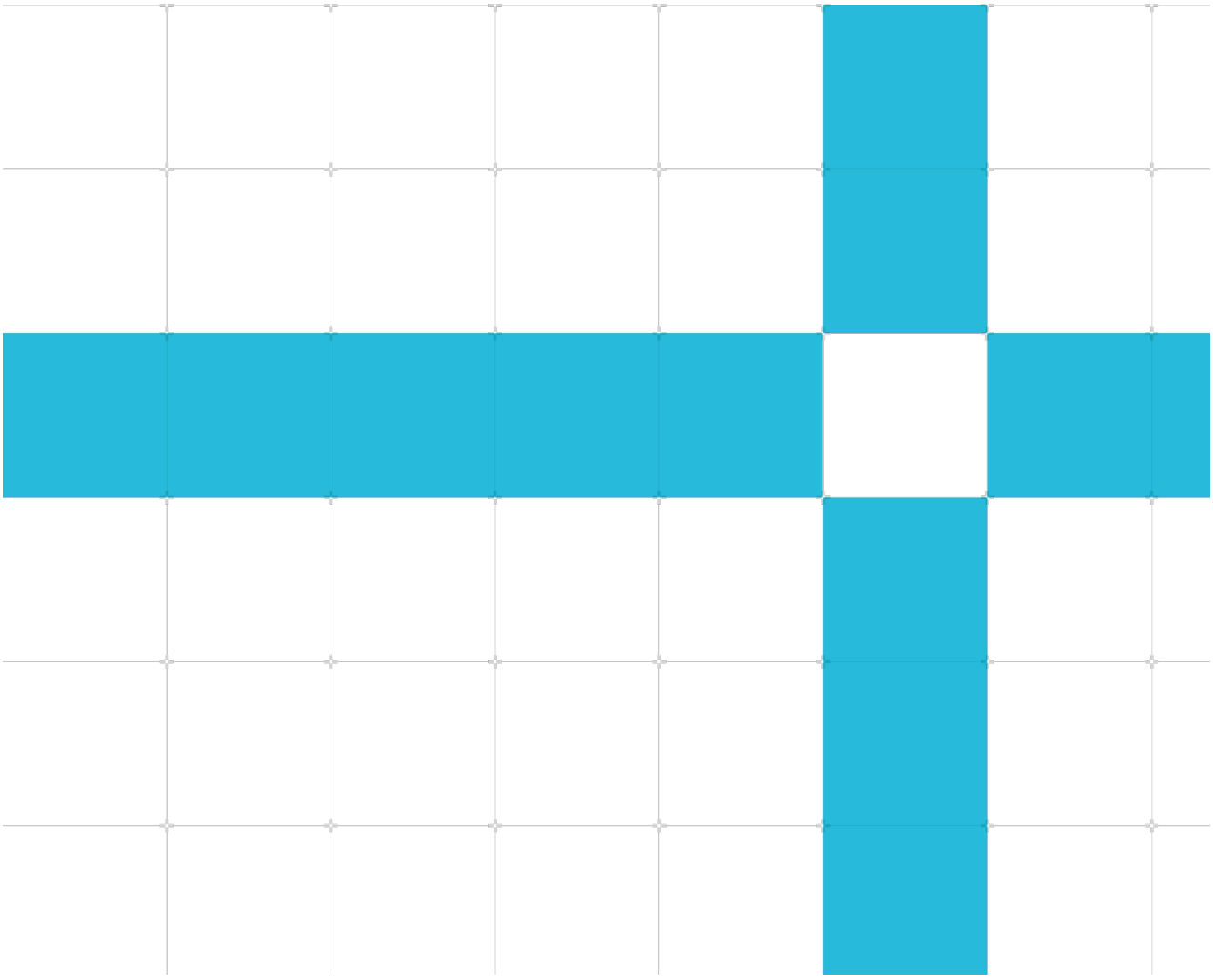
Date of issue: September 06, 2024

Non-Confidential

Document version: 6.0

Document ID: SDEN-3090091

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



This document (SDEN_3090091_6.0_en) was issued on September 06, 2024.

There might be a later issue at **http://developer.arm.com/documentation/SDEN-3090091**

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**.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on Arm® Cortex®-A720AE (MP170), 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.

Contents

Introduction

Scope

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

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

Categorization of errata

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

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

Change Control

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

September 06, 2024: Changes in document version v6.0

ID	Status	Area	Category	Summary
3645545	New	Programmer	Category B	TBRE might write to memory for which it does not have write permissions
3699562	New	Programmer	Category B	Read of ICH_VMCR_EL2.VBPR1 might return incorrect data based on SCR_EL3.NS
3711913	New	Programmer	Category B	DSB ST instructions might not properly order younger loads
3637713	New	Programmer	Category C	EDSCR.STATUS not updated on Halting Step when a Load-Exclusive instruction generates a synchronous exception
3655077	New	Programmer	Category C	PSTATE.{PAN,UAO} synchronization might not be honored while MSR PSTATE is speculative
3655257	New	Programmer	Category C	ESR.IESB can have an incorrect value on SError

April 30, 2024: Changes in document version v5.0

ID	Status	Area	Category	Summary
3456103	New	Programmer	Category B	MSR PSTATE.SSBS to 0 is not fully self-synchronizing
3522596	New	Programmer	Category C	TRBIRQ is incorrectly masked when TRBLIMITR_EL1.E = 0

March 13, 2024: Changes in document version v4.0

No new or updated errata in this document version.

January 19, 2024: Changes in document version v3.0

No new or updated errata in this document version.

December 15, 2023: Changes in document version v2.0

ID	Status	Area	Category	Summary
3120195	New	Programmer	Category C	SVE first faulting load crossing a 64B boundary might silently corrupt data in case of double external abort
3132558	New	Programmer	Category C	A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time
3132559	New	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails

November 16, 2023: 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
3711913	Programmer	Category B	DSB ST instructions might not properly order younger loads	r0p0	Open
3699562	Programmer	Category B	Read of ICH_VMCR_EL2.VBPR1 might return incorrect data based on SCR_EL3.NS	r0p0	Open
3645545	Programmer	Category B	TBRE might write to memory for which it does not have write permissions	r0p0	Open
3456103	Programmer	Category B	MSR PSTATE.SSBS to 0 is not fully self-synchronizing	r0p0	Open
3655257	Programmer	Category C	ESR.IESB can have an incorrect value on SError	r0p0	Open
3655077	Programmer	Category C	PSTATE.{PAN,UAO} synchronization might not be honored while MSR PSTATE is speculative	r0p0	Open
3637713	Programmer	Category C	EDSCR.STATUS not updated on Halting Step when a Load-Exclusive instruction generates a synchronous exception	r0p0	Open
3522596	Programmer	Category C	TRBIRQ is incorrectly masked when TRBLIMITR_EL1.E = 0	r0p0	Open
3132559	Programmer	Category C	Checked stores in precise mode might fail to report tag check fails	r0p0	Open
3132558	Programmer	Category C	A continuous flow of snoops and other instructions might prevent a store from becoming visible in finite time	r0p0	Open
3120195	Programmer	Category C	SVE first faulting load crossing a 64B boundary might silently corrupt data in case of double external abort	r0p0	Open

Errata descriptions

Category A

There are no errata in this category.

Category A (rare)

There are no errata in this category.

Category B

3711913
DSB ST instructions might not properly order younger loads

Status

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

Description

When executing a DSB ST instruction, load instructions following the DSB might not be properly ordered with respect to store instructions preceding the DSB.

Configurations affected

This erratum affects all configurations.

Conditions

This erratum occurs under the following conditions:

- A Write instruction W1 is executed.
- A DSB ST instruction is executed.
- A Read instruction R2 is executed.
- W1 and R2 access different memory locations.
- Some micro-architectural timing conditions occur.

Implications

If the previous conditions are met, R2 might be executed before W1. This only affects the result of R2 and the value of the memory location is not corrupted.

Workaround

This erratum can be avoided by inserting a DMB LD after each DSB ST using the following:

```
MOV x0, #5
MSR s3_6_c15_c8_0, x0
ISB
LDR x0, =0xd503329f
MSR s3_6_c15_c8_2, x0
LDR x0, =0xfffff3ff
MSR s3_6_c15_c8_3, x0
```

```
MOV x1, #0
ORR x1, #1<<0
ORR x1, #3<<4
ORR x1, #0xf<<6
ORR x1, #1<<22
ORR x1, #1<<32
MSR s3_6_c15_c8_1, x1
ISB
```

3699562
Read of ICH_VMCR_EL2.VBPR1 might return incorrect data based on SCR_EL3.NS

Status

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

Description

When ICH_VMCR_EL2.VBPR1 is written in Secure state (SCR_EL3.NS==0) and then subsequently read in Non-secure state (SCR_EL3.NS==1), a wrong value might be returned. The same issue exists in the opposite way: write in Non-secure state and read in Secure state. ICH_VMCR_EL2.VBPR1 is an alias of ICV_BPR1_EL1 which is architecturally defined as NOT banked. The RTL erroneously has this register implemented as two separate registers (secure and non-secure copies) banked by SCR_EL3.NS.

Configurations Affected

This erratum affects all configurations.

Conditions

The erratum occurs if all the following conditions apply:

1. The PE is executing at EL3
2. SCR_EL3.NS == 1 or 0
3. The PE executes an MSR ICH_VMCR_EL2.VBPR1 instruction
4. SCR_EL3.NS == 0 or 1 (the opposite value from when the MSR occurred)
5. The PE executes an MRS <dst>, ICH_VMCR_EL2.VBPR1 instruction

Implications

If the conditions are met, the MRS <dst>, ICH_VMCR_EL2.VBPR1 instruction will erroneously return the value that was last written to this field with the opposite SCR_EL.NS value from which it was read (or the reset value if it was never written in that security state).

Workaround

The workaround is for EL3 software that performs context save/restore on a change of Security state to use a value of SCR_EL3.NS when accessing ICH_VMCR_EL2 that reflects the Security state that owns the data being saved or restored. For example, EL3 software should set SCR_EL3.NS to 1 when saving or restoring the value ICH_VMCR_EL2 for Non-secure (or Realm) state. EL3 software should clear SCR_EL3.NS to 0 when saving or restoring the value ICH_VMCR_EL2 for Secure state.

3645545
TBRE might write to memory for which it does not have write permissions

Status

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

Description

TBRE might write to memory for which it does not have write permissions.

Configurations affected

This erratum affects all configurations.

Conditions

This erratum occurs under the following conditions:

- TRBLIMITR_EL1.E = 1
- TRBE is stopped due to a stage 1 or stage 2 fault that occurred during the translation for TRBPTR_EL1
- An MSR instruction setting TRBLIMITR_EL1.E = 0 is executed
- An MSR clearing TRBSR_EL1.S is executed

Implications

If the previous conditions are met, TRBE might write to the memory at TRBPTR_EL1 despite the translation for TRBPTR_EL1 having caused a stage 1 fault or a stage 2 fault.

Workaround

For non-virtualized operating systems, no workaround is expected to be required since correctly-written software should place an ISB instruction following the MSR TRBLIMITR_EL1 that clears the E bit. For virtualized operating systems, this erratum can be avoided by ensuring the hypervisor does not expose the support for TRBE to the guest operation system. Arm expects this matches existing usages of TRBE under virtualization.

3456103
MSR PSTATE.SSBS to 0 is not fully self-synchronizing

Status

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

Description

When PSTATE.SSBS is written to 0, the Arm Architecture specifies that side-effects are guaranteed to be visible to later instructions in the Execution stream. However, for a window of time during speculative execution of **`MSR PSTATE.SSBS`**, speculative store data bypassing might still occur.

Configurations affected

This erratum affects all configurations.

Conditions

The erratum occurs if the following condition applies:

`MSR PSTATE.SSBS` executes, setting PSTATE.SSBS to 0.

Implications

Security sensitive code executed shortly after **`MSR PSTATE.SSBS`** to 0 might not be fully protected by the *Speculative Store Bypass Safe* (SSBS) feature.

Workaround

Software at EL3, EL2, and EL1 should follow writes to the SSBS register with a *Speculation Barrier* (SB) instruction to ensure that the new value of PSTATE.SSBS affects subsequent instructions in the Execution stream under speculation.

A kernel at EL1 or EL2 should not advertise the presence of MRS/MSR instructions to read/write the SSBS register from EL0. Arm expects that kernels provide system calls for EL0 software to modify PSTATE.SSBS when the SSBS register is not implemented and that EL0 software will use this when the presence of the SSBS register is not advertised.

Category B (rare)

There are no errata in this category.

Category C

3655257
ESR.IESB can have an incorrect value on SError

Status

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

Description

ESR.IESB is supposed to be set only when getting an SError during the IESB phase of an exception entry or exit, and if SError exception is taken after this exception entry/exit. Instead, it can be incorrectly set to 1 or 0.

Configurations affected

This erratum affects all configurations.

Conditions

The erratum occurs if all the following conditions apply, and ESR.IESB for the SError exception might incorrectly be set to 1:

- SCTLR_ELx.IESB is set for any ELx, or SCR_EL3.EA & SCR_EL3.NMEA is set
- An SError arrives

The erratum also occurs if all the following conditions apply, and ESR.IESB for the SError exception, which is taken after exception entry, will incorrectly be set to 0:

- An exception entry other than SVC/HVC/SMC is taken, targeting EL1 or EL2
- SCTLR_ELx.IESB is set, where x is the target EL
- An SError arrives during exception entry

Implications

ESR.IESB on SError exceptions is not correct.

Workaround

There is no workaround.

3655077
PSTATE.{PAN,UAO} synchronization might not be honored while MSR PSTATE is speculative

Status

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

Description

When software directly writes PSTATE.PAN or PSTATE.UAO with an MSR instruction, the Arm Architecture specifies that side-effects are guaranteed to be visible to later instructions in the Execution stream. However, for a window of time prior to the execution of MSR PSTATE.{PAN,UAO}, instructions following the MSR might speculatively execute with the old context, prior to re-executing non-speculatively under the new, expected context.

Configurations affected

This erratum affects all configurations.

Conditions

The erratum occurs if the following condition applies:

- MSR PSTATE.{PAN or UAO} executes

Implications

Speculative execution of instructions using stale PSTATE.{UAO,PAN} context could in theory present a window of opportunity for a security attack. However, Arm security team has evaluated the practical risk to be very low, given the use-cases of the bits in question and the complexity involved in exploiting.

Workaround

A workaround is not expected to be required.

3637713
EDSCR.STATUS not updated on Halting Step when a Load-Exclusive instruction generates a synchronous exception

Status

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

Description

When a Load-Exclusive instruction is executed with Halting Step enabled, EDSCR.STATUS is not updated if the Load-Exclusive instruction causes a synchronous exception.

Configurations affected

This erratum affects all configurations.

Conditions

This erratum occurs under the following conditions:

1. In Debug state, the debugger enables Halting Step
2. Debug state is exited and a Load-Exclusive instruction (LDX*/LDAX*) is stepped
3. The Load-Exclusive generates a synchronous exception while executing

Implications

If the conditions are met, EDSCR.STATUS will not be updated.

Workaround

There is no workaround.

3522596
TRBIRQ is incorrectly masked when TRBLIMITR_EL1.E = 0

Status

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

Description

TRBIRQ is incorrectly masked when TRBLIMITR_EL1.E = 0.

Configurations affected

This erratum affects all configurations.

Conditions

This erratum occurs under the following conditions:

- TRBSR_EL1.IRQ = 1
- TRBLIMITR_EL1.E = 0

Implications

If the previous conditions are met, the trace buffer management interrupt, TRBIRQ, will not be asserted.

Workaround

Arm does not believe a workaround is needed for this erratum.

3132559
Checked stores in precise mode might fail to report tag check fails

Status

Fault Type: Programmer Category C.
Fault Status: Present in r0p0. 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 under the following conditions:

- 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

There is no workaround.

3132558
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. 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.

3120195
SVE first faulting load crossing a 64B boundary might silently corrupt data in case of double external abort

Status

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

Description

SVE first faulting load crossing a 64B boundary might silently corrupt data in case of double external abort.

Configurations Affected

This erratum affects all configurations.

Conditions

This erratum occurs under the following conditions:

- A load instruction is executed
- A load instruction R1 loads at least 32 bytes of data to SIMD&FP registers or SVE registers
- The access is mapped to cacheable memory
- The access crosses a 64 bytes boundary
- An external abort response is sent by the system for both 64B regions
- Very specific micro-architectural timing conditions occur

Implications

If the previous conditions are met, the load instruction might behave as follows:

- the First Fault Register is updated to the first active element after the 64B boundary
- data for all active elements will be set to 0

Workaround

There is no workaround.

Proprietary notice

(PRE-1121-V1.0)

Product and document information

Read the information in these sections to understand the release status of the product and documentation, and the conventions used in the Arm documents.

Product status

All products and Services provided by Arm require deliverables to be prepared and made available at different levels of completeness. The information in this document indicates the appropriate level of completeness for the associated deliverables.

Product completeness status

The information in this document is Final, that is for a developed product.

Product revision status

The rxpy identifier indicates the revision status of the product described in this manual, where:

rx

Identifies the major revision of the product.

py

Identifies the minor revision or modification status of the product.