Arm® DSU (MP135)

Software Developer Errata Notice

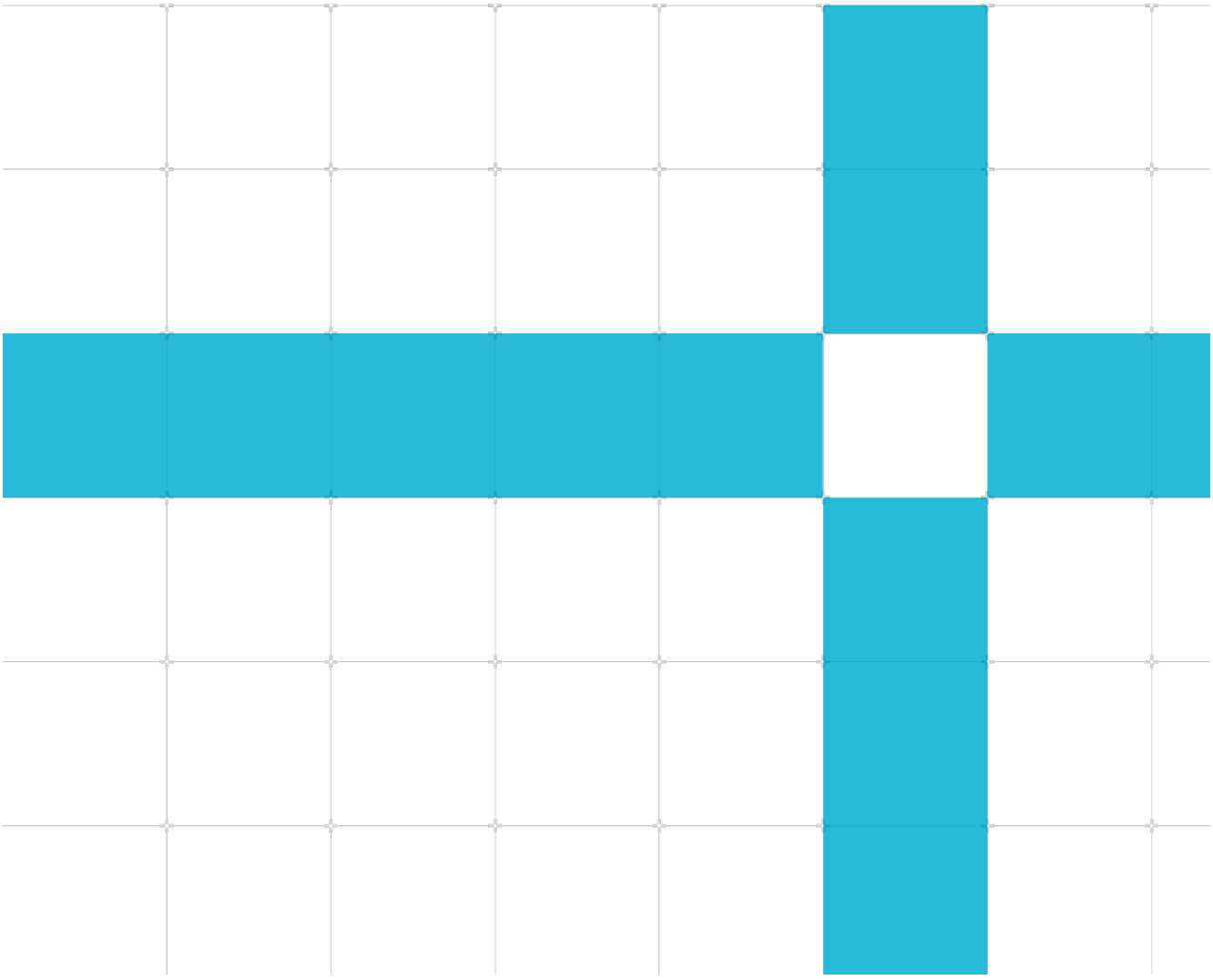
Date of issue: 05-Aug-2022

Non-Confidential

Document version: v2.0

Document ID: SDEN-2001827

This document contains all known errata since the r4p2 release of the product.



Non-confidential proprietary notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality status

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

Unrestricted Access is an Arm internal classification.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on Arm® DSU (MP135), create a ticket on **https://support.developer.arm.com**.

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

Inclusive language commitment

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

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

Contents

Introduction

Scope

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

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

Categorization of errata

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

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

Change Control

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

05-Aug-2022: Changes in document version v2.0

ID	Status	Area	Category	Summary
2714521	New	Programmer	Category C	Interconnect DErr on dirty data not reported in RAS registers

02-Nov-2020: Changes in document version v1.0

ID	Status	Area	Category	Summary
1327550	New	Programmer	Category B	Incorrect ordering of Clean to the Point of Persistence
1758329	New	Programmer	Category C	Incorrect ordering after change in cacheability

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
1327550	Programmer	Category B	Incorrect ordering of Clean to the Point of Persistence	r4p2	Open
2714521	Programmer	Category C	Interconnect DErr on dirty data not reported in RAS registers	r4p2	Open
1758329	Programmer	Category C	Incorrect ordering after change in cacheability	r4p2	Open

Errata descriptions

Category A

There are no errata in this category.

Category A (rare)

There are no errata in this category.

Category B

1327550
Incorrect ordering of Clean to the Point of Persistence

Status

Affects: DSU
Fault Type: Programmer Category B
Fault Status: Present in r4p2. Open.

Description

When using Cortex-A55 cores, Cortex-A75 cores or multithreaded cores with certain uncommon memory types that are not cached in the cluster, the DSU might perform a data cache clean to the Point of Persistence before a store instruction to the same address, even when the Armv8.2 architecture requires that these instructions occur in program order. Other types of maintenance instruction are not affected.

Configurations Affected

This erratum affects configurations where:

- The DSU cluster contains at least one Cortex-A55 core, Cortex-A75 core or multithreaded core.
- The DSU input signal **BROADCASTPERSIST**=1, indicating that the system implements the Point of Persistence.

Conditions

1. A Cortex-A55 core, Cortex-A75 core or a multithreaded core executes a store instruction to memory that is one of the following memory types:
   - Inner Write-Back, Outer Write-Through.
   - Inner Write-Back, Outer Non-cacheable.
   - Inner Write-Through, Outer Write-Back.
   - Inner Write-Through, Outer Write-Through.
   - Inner Write-Through, Outer Non-cacheable.
2. The same core subsequently executes the AArch64 data cache clean to the Point of Persistence instruction (`DC CVAP`) to the same cache line address without a `DMB` or `DSB` instruction in between.

Implications

These memory types are not expected to be common, so most software should not be affected. If these memory types are used, then the Clean to the Point of Persistence might occur before the store instruction, so the cache maintenance operation does not guarantee that the store has reached the Point of Persistence. As a result, some stores might not have reached persistent memory when software believes they have, which might lead to corruption during powerdown or power failure.

The DSU master ports indicate that all these memory types are Non-cacheable, so there is no impact on any system cache.

Workaround

If possible, software should map persistent memory using the Inner Write-Back, Outer Write-Back or Inner Non-Cacheable, Outer Non-Cacheable memory types. If this is not possible, then software can work around this issue by inserting a `DMB` instruction before a `DC CVAP` instruction, which ensures that any previous stores are ordered before it.

Category B (rare)

There are no errata in this category.

Category C

2714521
Interconnect DErr on dirty data not reported in RAS registers

Status

Fault Type: Programmer Category C
Fault Status: Present in r4p2. Open.

Description

Some errors reported by the interconnect on linefills that allocate to L3 will not be reported in the DSU RAS Error Record Registers.

Configurations Affected

This erratum affects configurations of the DSU with at least one CHI master port configured. It does not affect Direct connect configurations.
It also requires the interconnect to use DErr responses rather than the poison support on CHI.

Conditions

1. The DSU requests a linefill that allocates to L3. This could be generated from a PRFM instruction targeting L3 or, on some cores, the hardware prefetcher can generate these requests.
2. The interconnect returns dirty data with a DErr response indicating that there is an error in the data.

Implications

The DSU will not allocate the line to L3 because of the error, and so the dirty data will be discarded. The DSU RAS Error Record Registers are not correctly updated to indicate that the dirty data was lost. The original cause of the error happened outside of the DSU and should have been logged by the component that detected the error, although, this might have been recorded as a deferred error.
For systems that use DErr responses, there might be a negligible increase in overall system failure rate because of this erratum. However, any system where RAS is particularly important would be expected to use the poison field rather than signal a DErr, since the poison allows the line to be allocated into L3 and the error can remain deferred.

Workaround

No workaround is necessary.

1758329
Incorrect ordering after change in cacheability

Status

Affects: DSU
Fault Type: Programmer Category C
Fault Status: Present in r4p2. Open.

Description

If the memory type of an address region is changed from Cacheable to Non-cacheable, and then back again, and rare microarchitectural conditions occur, then stale data might be observed in a cache.

Configurations Affected

This erratum affects all configurations except Direct Connect configurations.

Conditions

1. An address region of memory is marked in the translation tables as Write-Back Cacheable memory.
2. The hardware prefetcher starts a data prefetch to an address within this region. This must generate a StashOnce CHI transaction from the core to the DSU, and in some cases the DSU might pass the StashOnce on to the interconnect if the DSU is configured with a CHI master.
3. The translation tables are updated to change the memory type to Non-cacheable or Device memory. This would involve a break-before-make sequence.
4. A sequence of cache clean and invalidate instructions are executed to ensure that any Cacheable data in the memory region does not remain in the caches.
5. The StashOnce transaction and the clean and invalidate transaction to the same address get reordered within the DSU or externally if the StashOnce was sent to the interconnect. This means that the StashOnce transaction can cause the line to be allocated into the cache after the cache maintenance has completed.
6. A core or other master in the system writes to the region that is now marked Non-cacheable or Device.
7. The translation tables are changed a second time, to mark the memory as Write-Back Cacheable again.
8. A load instruction is executed. The load might observe the stale data that was prefetched into the cache, rather than the Non-cacheable data that was written.

Implications

The above sequence is very specific and would typically take a very long time to execute. It requires that the StashOnce transaction is started before the translation table modification, yet does not complete until after both the translation table modification and the cache maintenance. Additionally, the StashOnce and cache maintenance transactions must be reordered by the DSU or interconnect, and this is an unlikely event, especially if they are not started at a similar time. Therefore the combination of these conditions is going to be extremely rare. Furthermore, the change in memory type implies a change of use of the memory, and many such changes of use will not require preservation of the data between uses.

Workaround

No workaround is necessary.

Note that this erratum is caused by a deficiency in the CHI architecture and will be corrected in CHI Issue D onward.