Arm® Processor
Cortex®-A17 (MP038)
Software Developers Errata Notice

Non-Confidential - Released

This document contains all errata known at the date of issue in releases up to and including revision r0p0.

 Copyright © 2014-2019 Arm. All rights reserved.

Non Confidential

Software Developers Errata Notice

Copyright © 2014-2019 Arm. All rights reserved.

Non-Confidential Proprietary Notice

This document is protected by copyright and the practice or implementation of the information herein may be protected by one or more patents or pending 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.

This document is Non-Confidential but any disclosure by you is subject to you providing the recipient the conditions set out in this notice and procuring the acceptance by the recipient of the conditions set out in this notice.

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

Unless otherwise stated in the terms of the Agreement, this document is provided “as is”. Arm makes no representations or warranties, either express or implied, included but not limited to, warranties of merchantability, fitness for a particular purpose, or non-infringement, that the content of this document is suitable for any particular purpose or that any practice or implementation of the contents of the document will not infringe any third party patents, copyrights, trade secrets, or other rights. Further, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of such 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 LOSS, LOST REVENUE, LOST PROFITS OR DATA, SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF OR RELATED TO ANY FURNISHING, PRACTICING, MODIFYING OR ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Words and logos marked with ® or TM are registered trademarks or trademarks, respectively, of Arm Limited. Other brands and names mentioned herein may be the trademarks of their respective owners. Unless otherwise stated in the terms of the Agreement, you will not use or permit others to use any trademark of Arm Limited.

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.

In this document, where the term Arm is used to refer to the company it means “Arm or any of its subsidiaries as appropriate”.

Copyright © 2014-2019 Arm Limited 110 Fulbourn Road, Cambridge, England CB1 9NJ. All rights reserved.

Web Address

http://www.arm.com

Feedback on content

If you have any comments on content, then send an e-mail to errata@arm.com . Give:

- the document title
- the document number, PJDOC-466751330-10836
- the page numbers to which your comments apply
- a concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Release Information

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 in Chapter 2. Fixed errata are not shown as updated unless the erratum text has changed. The summary table in section 2.2 identifies errata that have been fixed in each product revision.

18 Mar 2014: Changes in Document v1

Page	Status	ID	Cat	Rare	Summary of Erratum
30	New	824719	CatC		Stage 2 fault might not be reported when address translation for a Cache Maintenance Operation uses a NoAccess Domain

22 May 2014: Changes in Document v2

Page	Status	ID	Cat	Rare	Summary of Erratum
10	New	826370	CatA		ICIMVA operation executed concurrently with a broadcast ICI or TLBI operation may cause either operation to be performed
23	New	826369	CatB	Rare	CPU might execute incorrect instructions following a TLBI+DSB sequence
24	New	827821	CatB	Rare	CPU might execute stale instructions following broadcast ICI operations
25	New	828419	CatB	Rare	CPU might execute an Instruction Cache Maintenance Operation by MVA with an incorrect NS descriptor
26	New	828420	CatB	Rare	In a multi-cluster system, Cache Clean by MVA to PoC might not be correctly executed

17 Jul 2014: Changes in Document v3

Page	Status	ID	Cat	Rare	Summary of Erratum
13	New	830075	CatB		Instructions might be fetched from unauthorized memory region when MMU is off
14	New	831171	CatB		A DSB might complete before the end of a broadcasted TLB or Icache invalidation
27	New	832071	CatB	Rare	Read after read issue with Streaming store and concomitant store by another CPU
31	New	831169	CatC		ECC error on L2 DATA RAMs might report an erroneous error index

11 Sep 2014: Changes in Document v4

Page	Status	ID	Cat	Rare	Summary of Erratum
15	New	833219	CatB		In Debug State, transition from PL0 or PL1 into PL2 might not work if secure debug privileges are disabled
16	New	834819	CatB		An access might use an invalid translation after it has been made valid
17	New	834869	CatB		A TLBIMVA(IS) executed in Non-Secure PL1 might fail to invalidate the required TLB entries when Stage2 is enabled
18	New	834871	CatB		A CPU might execute stale non-cached instructions after an ISB
27	Updated	832071	CatB	Rare	Read after read issue with Streaming store and concomitant store by another CPU
32	New	833221	CatC		The Error Location field of the L2MRERRSR register might not be correctly updated
31	Updated	831169	CatC		ECC error on L2 DATA RAMs might report an erroneous error index

27 Feb 2015: Changes in Document v5

Page	Status	ID	Cat	Rare	Summary of Erratum

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

11	New	841920	CatA	Rare	A sequence of two Advanced SIMD or Floating-point instructions might cause a dead-lock or a corruption
28	New	844219	CatB	Rare	DVM Sync might not properly guarantee completion of all accesses using an invalidated TLB entry
33	New	844221	CatC		Stage2 Descriptors might be accessed using an incorrect Cacheability attribute
34	New	844223	CatC		UNDEFINED instructions might not UNDEF

23 Apr 2015: Changes in Document v6

Page	Status	ID	Cat	Rare	Summary of Erratum
12	New	846720	CatA	Rare	Concurrent modification and execution of instructions might fail
29	New	845920	CatB	Rare	Cache Coherency protocol is broken when connecting a Cortex-A17 cluster with some specific ACE interconnects
35	New	846719	CatC		PAR might not be architecturally correct upon completion of an ATS12NS instruction
36	New	847219	CatC		CPU might take a breakpoint instead of a vector catch.

29 Oct 2015: Changes in Document v7

Page	Status	ID	Cat	Rare	Summary of Erratum
19	New	852421	CatB		DMB ST might fail to create order between stores
20	New	852423	CatB		Execution of a sequence of instruction might lead to either a data corruption or a CPU deadlock

12 Mar 2018: Changes in Document v8

Page	Status	ID	Cat	Rare	Summary of Erratum
22	New	857272	CatB		Processor might deadlock under some very rare internal conditions

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

Contents

 Copyright ©2014-2019 Arm. All rights reserved.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

Chapter 1.
Introduction

This chapter introduces the errata notice for the ARM Cortex-A17 MPCore processor.

1.1. Scope of this document

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

- the current status of the defect
- where the implementation deviates from the specification and the conditions under which erroneous behavior occurs
- the implications of the erratum with respect to typical applications
- the application and limitations of a 'work-around' where possible

This document describes errata that may impact anyone who is developing software that will run on implementations of this ARM product.

1.2. Categorization of errata

Errata recorded in this document are split into the following levels of severity:

Table 1 Categorization of errata

Errata Type	Definition
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.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

Chapter 2.
Errata Descriptions

2.1. Product Revision Status

The r*n*p*n* identifier indicates the revision status of the product described in this book, where:

r*n* Identifies the major revision of the product.

p*n* Identifies the minor revision or modification status of the product.

2.2. Revisions Affected

Table 2 below lists the product revisions affected by each erratum. A cell marked with **X** indicates that the erratum affects the revision shown at the top of that column.

This document includes errata that affect revision **Error! Reference source not found.** only.

Refer to the reference material supplied with your product to identify the revision of the IP.

For Cortex-A17 r0p1 errata, please refer to Cortex-A12 Software Developers Errata Notice document (r0).

Table 2 Revisions Affected

ID	Cat	Rare	Summary of Erratum	r0p0
826370	CatA		ICIMVA operation executed concurrently with a broadcast ICI or TLBI operation may cause either operation to be performed	X
846720	CatA	Rare	Concurrent modification and execution of instructions might fail	X
841920	CatA	Rare	A sequence of two Advanced SIMD or Floating-point instructions might cause a dead-lock or a corruption	X
857272	CatB		Processor might deadlock under some very rare internal conditions	X
852423	CatB		Execution of a sequence of instruction might lead to either a data corruption or a CPU deadlock	X
852421	CatB		DMB ST might fail to create order between stores	X
834871	CatB		A CPU might execute stale non-cached instructions after an ISB	X
834869	CatB		A TLBIMVA(IS) executed in Non-Secure PL1 might fail to invalidate the required TLB entries when Stage2 is enabled	X
834819	CatB		An access might use an invalid translation after it has been made valid	X
833219	CatB		In Debug State, transition from PL0 or PL1 into PL2 might not work if secure debug privileges are disabled	X
831171	CatB		A DSB might complete before the end of a broadcasted TLB or Icache invalidation	X

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

ID	Cat	Rare	Summary of Erratum	r0p0
830075	CatB		Instructions might be fetched from unauthorized memory region when MMU is off	X
845920	CatB	Rare	Cache Coherency protocol is broken when connecting a Cortex-A17 cluster with some specific ACE interconnects	X
844219	CatB	Rare	DVM Sync might not properly guarantee completion of all accesses using an invalidated TLB entry	X
832071	CatB	Rare	Read after read issue with Streaming store and concomitant store by another CPU	X
828420	CatB	Rare	In a multi-cluster system, Cache Clean by MVA to PoC might not be correctly executed	X
828419	CatB	Rare	CPU might execute an Instruction Cache Maintenance Operation by MVA with an incorrect NS descriptor	X
827821	CatB	Rare	CPU might execute stale instructions following broadcast ICI operations	X
826369	CatB	Rare	CPU might execute incorrect instructions following a TLBI+DSB sequence	X
847219	CatC		CPU might take a breakpoint instead of a vector catch.	X
846719	CatC		PAR might not be architecturally correct upon completion of an ATS12NS instruction	X
844223	CatC		UNDEFINED instructions might not UNDEF	X
844221	CatC		Stage2 Descriptors might be accessed using an incorrect Cacheability attribute	X
833221	CatC		The Error Location field of the L2MRERRSR register might not be correctly updated	X
831169	CatC		ECC error on L2 DATA RAMs might report an erroneous error index	X
824719	CatC		Stage 2 fault might not be reported when address translation for a Cache Maintenance Operation uses a NoAccess Domain	X

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

2.3. Category A

826370: ICIMVA operation executed concurrently with a broadcast ICI or TLBI operation may cause either operation to be performed

Category A

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

ICIMVA operation executed concurrently with a broadcast ICI or TLBI operation may cause either operation to be performed using incorrect attributes

Configurations affected

This erratum affects all configurations of Cortex-A17 in systems containing more than 1 CPU (MP2/MP3/MP4 cluster or ACE system)

Conditions

The following conditions have to be met in order for the erratum to occur:

- The local ICIMVA operation needs to perform a Page Table Walk

- The result of the Page Table Walk for the ICIMVA operation is arbitrated in the same cycle as a broadcast ICI or TLBI operation

- Both the local ICIMVA operation and the broadcast operation are affected if the broadcast operation is either:

  - an ICIALL operation from a CPU in the same cluster

  - a Virtual Instruction Cache Invalidate received on ACE

  - a Physical Instruction Cache Invalidate received on ACE that does not specify a Virtual Index

- Only the local ICIMVA operation is affected if the broadcast operation is either:

  - an ICIMVA operation from a CPU in the same cluster

  - a Physical Instruction Cache Invalidate received on ACE that also specifies a Virtual Index

Implications

The affected operations will be performed using incorrect attributes, causing the expected operation not to be performed. As a result, instruction fetches may see stale instructions.

Workaround

No workaround.

 Copyright ©2014-2019 Arm. All rights reserved.

2.4. Category A (Rare)

841920: A sequence of two Advanced SIMD or Floating-point instructions might cause a dead-lock or a corruption

Category A Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Under very rare timing conditions, a sequence of two Advanced SIMD or Floating-point instructions might cause a deadlock or a data corruption.

Configurations affected

All Cortex-A12 and all Cortex-A17 configurations with SIMD and Floating-point instruction extension are affected.

Conditions

The errata is triggered by a code sequence of two consecutive instructions, as described below:

One of these Advanced SIMD instructions as the first instruction:

- VZIP to Quad register.
- VUZP to Quad register.
- VCVT.F16.F32 to Double or Quad register.
- VCVT.F32.F16 to Double or Quad register.
- VLD3 to one lane.
- VLD4 to one lane.
- VLD3 to all lanes.
- VLD4 to all lanes.

One of the following Advanced SIMD or floating-point instruction as the second instruction:

- Reading the CPSR flags (conditional instruction).
- Writing the CPSR flags (VMRS apsr_nzcv, FPSR).
- Instruction transferring an Advanced SIMD or floating-point register to a general purpose register (VMOV, VMRS FPSCR).

Additional micro-architectural conditions are required to reach the point of failure. These micro-architectural conditions are not directly controllable in software, but include unexpected patterns of memory accesses and code sequences.

Implications

When the erratum occurs, it might cause a deadlock or a data corruption.

Workaround

There is no workaround. ARM recommends using a Cortex-A17 REL maintenance release for future SoC.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

846720: Concurrent modification and execution of instructions might fail

Category A Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Under some very rare circumstances, when one thread of execution modifies instructions as they are executed by another thread of execution without explicit synchronization, the instructions executed by the second thread might be UNPREDICTABLE, even for the set of instructions where the architecture guarantees that behavior is consistent with execution of either:

- The instructions originally fetched.

- A fetch of the new instructions that results from the modification.

Configurations affected

All configurations of Cortex-A12 and Cortex-A17 are affected.

Conditions

For the erratum to occur, the following conditions must be met:

- This execution thread must branch into the upper 64 bits of a 128-bit aligned chunk of instructions without executing any of the instructions in the lower 64 bits of the 128-bit chunk of instructions. These instructions must be fetched from the instruction cache, as opposed to being fetched from the memory.

- The cache line containing the previously executed instructions must be naturally evicted by cache replacement policy, as opposed to execution of an Instruction Cache Maintenance Operation.

- The second thread of execution must modify at least an instruction within the upper 64 bits of the 128-bits chunk of instructions. These instructions must be made visible to the Point of Unification.

- The first thread must fetch the same 128-bits chunk of instructions into its cache.

- Without having received any Instruction Cache Maintenance Operation, nor any TLB Maintenance Operation, nor having executed any of these, nor executed any Instruction Synchronisation Barrier (ISB), the first execution thread must execute instructions from this chunk of instructions starting with instructions within the 64 lower bits, and continuing over the upper 64 bits.

Implications

The instructions executed in the upper 64 bits of the instruction chunk have an UNPREDICTABLE value, including the logical OR value of the instructions originally fetched and the instructions that result from the modification by the second thread.

ARM expects JiT compilers dynamically patching branch targets to make use of this functionality. However, the hardware constraints required to hit the erratum make it unlikely that these programs are impacted.

Workaround

There is no workaround.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

2.5. Category B

830075: Instructions might be fetched from unauthorized memory region when MMU is off

Category B
Products Affected: Cortex-A17 MPCore.
Present in: r0p0

Description

When disabling the MMU, architecture expects that instructions are only fetched from:

- Addresses already seen by previously executed branches.
- Addresses in the 4k region after the currently executing instruction.

Under specific conditions, a limited number of instructions might be fetched from the target of a branch executed before MMU was stopped, which is incorrect.

Configurations affected

This erratum affects all configurations of Cortex-A12 and Cortex-A17.

Conditions

The processor must be executing a code sequence having the following characteristics:

- A branch is executed, and put in the BTAC.
- The MMU is turned off for the translation regime used by the CPU. This can happen in the following case:
  - Changing the value of the system control register for this translation regime.
  - Changing the translation regime by changing the security state or privilege level.
- One of the four subsequent instructions has the same VA as a previously executed branch.

If these conditions are met, the CPU can issue one access to an unauthorized memory region.

Implications

The affected CPU might fetch instructions from a forbidden unauthorized memory region.

Workaround

A software workaround exists for the cases where the MMU is turned off by changing the system control register, or where an exception returns from a more privileged translation regime with its MMU enabled to another less privileged translation regime. It consists of performing a Branch Predictor Invalidate All operation before doing the ISB (in the case where the system control register is changed) of the exception return instruction (in the case of a change of the translation regime).

This workaround does not apply in the case where the less privileged translation regime has its MMU enabled while it is disabled for a more privileged translation regime.

 Copyright ©2014-2019 Arm. All rights reserved.
Non Confidential

831171: A DSB might complete before the end of a broadcasted TLB or Icache invalidation

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Within rare timing constraints, a DSB following a TLB or Icache invalidation might complete before the TLB or ICache invalidation has effectively completed on other CPUs within the cluster.

Configurations affected

This erratum affects all configurations of Cortex-A17 in a system with three or more CPUs.

Conditions

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

- CPU1 performs a TLB or ICache maintenance operation followed by a DSB.
- CPU2 performs a TLB or ICache maintenance operation followed by a DSB.
- CPU0 is running.

Implications

When executing the sequence of transactions described above and when particular timing conditions occur, CPU1 might complete the DSB before CPU0 has effectively performed the broadcasted TLB or Icache invalidation.

Workaround

- Issue the TLB or ICache maintenance invalidation twice before the DSB solves the problem. If multiple TLB or ICache maintenance invalidations are executed back-to-back, only the last maintenance operation before the DSB needs to be replicated.

or

- Use IPI for TLB and ICache maintenance operations.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

833219: In Debug State, transition from PL0 or PL1 into PL2 might not work if secure debug privileges are disabled

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Switching from PL0 or PL1 into PL2 through the execution of an MSR CPSR_fsxc instruction while in debug mode might not work.

Configurations affected

This erratum affects all configurations of Cortex-A12 and Cortex-A17.

Conditions

This erratum occurs when Secure Debug Privileges are disabled by clamping SPIDEN to 0.

Implications

In debug state, an MSR CPSR_fsxc executed in PL0 or PL1 and targeting entry into PL2 will leave the processor in the Privilege Level it was before executing the MSR instruction.

Workaround

To allow transition from PL0 or PL1 into PL2 in debug state the debugger can force the execution of an HVC instruction present in the code by branching and then stepping on it.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

834819: An access might use an invalid translation after it has been made valid

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

An access (Load/Store or instruction fetch) might reuse an old invalid translation after a new valid translation should have correctly been paged in.

Configurations affected

This erratum affects all Cortex-A17 configurations.

Conditions

- The MMU fetches its descriptors in Non-Cacheable.

- A Non-Cacheable Descriptor at address A makes the translation for virtual address B invalid.

- A PLD or prefetch is performed at address B that launches a PTW using this descriptor.

- The value of this descriptor is changed and the address is paged in. The necessary barrier (DSB + ISB) is performed.

- An access (Load/store or instruction fetch) at address B is performed and requests a translation to the MMU.

- This MMU request merges in the old request launched by the PLD that uses the old descriptor.

Note that ARM recommends to not configure the MMU to access the descriptor in non-cacheable. The errata is not present when the descriptor is accessed in cacheable.

Implications

If all of these conditions are met, an access might receive an unexpected abort.

Workaround

The workaround consists of ensuring the abort handler can cope with spurious faults.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

834869: A TLBIMVA(IS) executed in Non-Secure PL1 might fail to invalidate the required TLB entries when Stage2 is enabled

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

In a system where Stage2 translation is enabled and where an Operating System executing in Non-Secure PL1 replaces a valid translation table descriptor with another valid translation table descriptor without using the "break-before-make" scheme, a TLBIMVA operation executed to ensure that the new mapping is visible might fail to properly invalidate all TLB entries for the previous mapping.

Configurations affected

This erratum affects all configurations of Cortex-A12 and Cortex-A17.

Conditions

This erratum only occurs when Stage1 uses the Short-descriptor translation table format and for these specific combinations of Stage1 page size and Stage2 page size:

- Switching from a 64KB page to another 64KB page, if the old page was backed by a 4KB Stage2 page and the new one by a 2MB Stage2 page.

- Switching from a 16MB page to another 16MB page, if the old page was backed by a 2MB Stage2 page and the new one by a 1GB Stage2 page.

Stage1 translation using the Long-descriptor format and other combinations of Stage1 and Stage2 pages sizes are not affected by this erratum.

Implications

It is possible for a TLBIMVA(IS) operation to fail to invalidate TLB entries for the old mapping.

Workaround

When installing the new mapping, perform a TLB invalidation for the entire range covered by the new page.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

834871: A CPU might execute stale non-cached instructions after an ISB

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

After an ISB, the CPU might execute instructions that have been fetched from non-cached memory prior to the execution of the ISB.

Configurations affected

This erratum affects all configurations of Cortex-A12 and Cortex-A17.

Conditions

The following conditions must be hit to trigger the erratum:

- The CPU performs an instruction fetch request through a non-cached mapping at the address of the following modified code sequence.

- The CPU executes the self-modifying sequence by storing new instructions to the region seen as non-cached by the Instruction Cache, without invalidating the instruction cache by MVA.

- The CPU starts executing the newly-installed code through a non-cached mapping.

- The aforementioned instruction fetch request completes and returns values read before the store of the new instructions.

Implications

The affected CPU might execute stale instructions.

Workaround

Invalidating the relevant addresses in the Instruction Cache though the use of ICIMVAU, despite the memory location being marked as non-cached, solves the issue.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

852421: DMB ST might fail to create order between stores

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Under very rare timing conditions, execution of a DMB ST instruction might fail to properly order stores from GroupA and stores from GroupB.

Configurations affected

All Cortex-A17 configurations are affected.

Conditions

For the erratum to occur, the following conditions must be met:

- The CPU is writing a full cache line using "write streaming" mode without allocating the cache line into the L1.

- DMB ST is executed before the cache line has been written back and can be observed by other agents.

Implications

When the erratum occurs, a DMB ST instruction intended to create order between the full cache line write (in GroupA) and younger stores (in GroupB) might fail to create order properly; any GroupB store might then be observed before the full cache line write from GroupA can be observed.

Workaround

Setting bit[24] of the Feature Register prevents the erratum.

```
MRC p15, 0, r0, c15, c0, 1
ORR r0, r0, #1<<24
MCR p15, 0, r0, c15, c0, 1
```

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

852423: Execution of a sequence of instruction might lead to either a data corruption or a CPU deadlock

Category B

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Under very rare timing conditions, when executing a sequence of two instructions having opposed condition codes, the CPU might hang or corrupt the value of the destination register.

Configurations affected

This defect affects all configurations of the processor.

Conditions

The erratum might be triggered upon execution of two instructions, back to back in execution order, the two instructions having the following characteristics:

- the two instructions must be conditional, with resolution of their condition codes being opposed;

- each of the two instructions must have either only one operand, or two operands in which case the destination register must be identical to one of the two operand registers.

Note that other timing conditions are required to trigger the erratum.

Implications

Under rare timing conditions, either of a deadlock or a data corruption might occur.

Workaround

Setting bit[12] of the Diagnostic Register prevents this erratum from occurring:

```
MRC p15, 0, r0, c15, c0, 1
ORR r0, r0, #1<<12
MCR p15, 0, r0, c15, c0, 1
```

The workaround is expected to have a negligible performance impact.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

857272: Processor might deadlock under some very rare internal conditions

Category B
Products Affected: Cortex-A17 MPCore.
Present in: r0p0

Description

Under very rare timing conditions, the processor might hang.

Configurations affected

This defect affects all configurations of the processor.

Conditions

The erratum might be triggered when certain internal hardware conditions are met.

Implications

Under rare timing conditions, a deadlock might occur.

Workaround

Setting bit[11:10] of the Diagnostic Register prevents this erratum from occurring:

```
MRC p15, 0, r0, c15, c0, 1
ORR r0, r0, #1<<11
ORR r0, r0, #1<<10
MCR p15, 0, r0, c15, c0, 1
```

The workaround is expected to have a negligible impact on performance.

 Copyright ©2014-2019 Arm. All rights reserved.
Non Confidential

2.6. Category B (Rare)

826369: CPU might execute incorrect instructions following a TLBI+DSB sequence

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

When executing the sequence TLBI followed by a DVM SYNC, ARMv7 architecture expects that all external memory requests using previous TLB mapping are finished on the DVM SYNC. This means that any modification on data on pages that are no-longer executable are not visible by any CPU.

It is possible for Cortex-A12 and Cortex-A17 to start, or use the result of, instruction fetches that used an address targeted by the TLBI operation even after the subsequent DSB/DVM Sync completes.

Configurations affected

This erratum affects all configurations of Cortex-A12 and Cortex-A17.

Conditions

The processor must be executing a code sequence having the following characteristics:

- The CPU receives a TLBI request.

- The CPU receives a DVM SYNC request.

- A CPU changes data on a PA that was mapped before the TLBI, but is no longer mapped as executable.

- The Iside receives data from a linefill for this PA, with the new data. The request for this linefill could have been sent at any previous time, independently of TLBI and DVM SYNC requests.

- The Iside sends the incorrect data to the Core.

Implications

The affected CPU might execute incorrect instructions.

Workaround

The software workaround is to force all CPUs to execute a ISB by broadcasting a software synchronisation request.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

827821: CPU might execute stale instructions following broadcast ICI operations

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

A code-modifying sequence might fail to synchronize instruction fetches of other CPUs with store instructions of the CPU that executes the code-modifying sequence, leading to execution of stale instructions.

Configurations affected

This erratum affects configurations of Cortex-A17 with more than one CPU, or that connects to a coherent interconnect.

Conditions

The processor must be executing a code sequence having the following characteristics:

- CPU A starts a linefill caused by an instruction fetch at address C.

- CPU B executes a code-modifying sequence targeting C (Store, Data Cache Clean, Instruction Cache and Branch Prediction invalidation).

- CPU A and CPU B synchronize to ensure the newly-installed code is visible to CPU A.

- CPU A starts executing the newly-installed code.

The linefill on CPU A completes and returns stale data from memory.

In addition, the erratum only occurs if the Instruction Cache Invalidation instruction described above is either:

- an ICIALL operation.

- an ICIMVA operation broadcast from a CPU in another cluster.

Implications

The affected CPU might execute stale instructions.

Workaround

The software workaround is to add an additional ICIMVA operation at the end of each sequence of ICI operations affected by this erratum (as described in the `Conditions` section), before signalling the completion of that sequence to other CPUs. This additional ICIMVA operation can target any MVA for which address translation will not generate a fault. This includes repeating the last ICIMVA operation.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

828419: CPU might execute an Instruction Cache Maintenance Operation by MVA with an incorrect NS descriptor

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Under rare timing conditions, a local Instruction cache maintenance operation by MVA might target the Secure alias instead of the Non-secure alias.

Configurations affected

This erratum affects configurations of Cortex-A17 with more than one CPU, or that are connected to a coherent interconnect.

Conditions

The processor must be executing a code sequence having the following characteristics:

- CPU A starts an ICIMVA operation that targets a Non-secure line and misses both in the L1 and in the L2 TLB and therefore launches a page table walk.

- CPU A receives a broadcasted TLB invalidate operation from another CPU or another cluster.

- CPU A then receives a broadcasted Instruction cache maintenance operation from another CPU or another cluster.

- CPU A completes the translation requested by the ICIMVA but cannot directly send the request to the instruction cache because of the previous broadcasted Instruction cache maintenance.

Implications

The affected CPU might not correctly invalidate stale instructions.

Workaround

A potential workaround is to replace ICIMVAU operations by ICIALLUIS operations.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

828420: In a multi-cluster system, Cache Clean by MVA to PoC might not be correctly executed

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

If a line is present in the dirty state in another cluster, it is possible that a Data Clean operation by MVA is not correctly performed and that the line remains in the dirty state in the multi-cluster system and does not become visible externally to a non-coherent agent.

Configurations affected

This erratum affects all configurations of Cortex-A17 in a multi-cluster system.

Conditions

The multi-cluster system must be executing a code sequence with the following characteristics:

- The line at address A is present in cluster A in the dirty state.

- The line at address A is not present in cluster B.

- One of the CPUs in cluster B executes a Data Cache Clean operation to PoC for the line at address A.

- One of the CPUs in cluster B, or the ACP of cluster B, concurrently requests the same line on a data access that will be sent as a ReadShared of a ReadUnique on the AR channel. This can be because of a Load or a Store operation or to a PLD or prefetch operation targeting the L1.

Implications

The affected cluster might not correctly execute the maintenance operation and might keep the affected line in a dirty state without updating the main memory. Therefore, an agent using a non-coherent cacheable buffer that relies on correct execution of the cache maintenance operation might not see the updated value.

Workaround

The software workaround is to treat every Data Clean by MVA to PoC as Data Clean Invalidate by MVA to PoC.

This can be done by setting bit[30] of the CPU diagnostic register (accessible in the cp15 register space at address cp15,0,c15,c0,1). It has the effect of transforming every Data Clean operation in Data Clean Invalidate operation.

This can also be done by replacing the DCCMVAC instruction by a DCCIMVAC in the case of communication between a device and the CPU using a non-coherent cacheable buffer.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

832071: Read after read issue with Streaming store and concomitant store by another CPU

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Within rare timing constraints, if a CPU is performing a streaming store (write full cache line) and another CPU is performing a store at the same address, a younger load might read the older value of the data (written by the streaming store), while an older load from the same CPU reads the newest value of the data written by the other CPU.

Configurations affected

This erratum affects all configurations of Cortex-A17 and Cortex-A12 with more than one CPU.

Conditions

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

- Core0 is writing a full cache line at address A.
- Core0 issues two load operations out-of-order at this address.
- Core1 is also writing at this address.

Implications

When executing the sequence of transactions described above and when particular timing conditions occur, the two load operations might not received the data in the correct order as shown by the following table.

Core0	Core1
STR FULL @A, a0	STR @A, a1
LDR @A, a1	
LDR @A, a0	

Workaround

This case can be avoided by:

- Issuing a DMB between the write of the full cache line and the subsequent load operations

or

- Issuing a DMB between the two load operations.

or

- Disabling the Streaming Mode. This can be done by setting bit[1] of the CPU diagnostic register (accessible in the cp15 register space at address cp15,0,c15,c0,1). Note that streaming performance of the processor is impacted, resulting in a potential 1 to 2% performance drop on mobile workloads.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

844219: DVM Sync might not properly guarantee completion of all accesses using an invalidated TLB entry

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

A DVM SYNC operation intended to guarantee completion of all accesses using an invalidated TLB entry might not correctly wait for all accesses to complete.

Configurations affected

All Cortex-A17 configurations are affected.

Conditions

For the erratum to occur, the following conditions must be met:

- CPU A is performing a page table walk for VA X.

- CPU B executes a TLB maintenance operation targeting VA X.

- CPU A starts a new page table walk for VA Y.

In addition, VA X and VA Y must have the following properties:

- VA Y and VA X are within the same 1M-aligned region.

- The TLB invalidation targets a page table entry used to translate VA Y and VA X that is not the last-level page table entry.

  - When using the VMSA page table format, the erratum only affects modifications to the Level 1 page tables when using 4k pages

  - When using the LPAE page table format, this erratum only affects modifications to

    - Level 1 page table entries when using 2M pages

    - Level 1 and Level 2 page tables entries when using 4k pages.

This erratum does not affect cases where a page table entry is changed from a pointer to a lower-level table to a last-level entry that covers the same physical range and with the same attributes

Implications

When the erratum occurs, a DVM SYNC operation intended to complete the TLB maintenance from CPU B might not properly guarantee completion of the accesses on CPU A that used the previous translation for VA Y. As a result, accesses using the stale TLB entry might be performed until the associated instructions complete.

Workaround

ARM does not recommend to deploy any workaround for this errata.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

845920: Cache Coherency protocol is broken when connecting a Cortex-A17 cluster with some specific ACE interconnects

Category B Rare

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

In a system including a Cortex-A17 cluster and a CCI-500 interconnect, cache coherency might not be ensured by the Cortex-A17 cluster with regards to external cache coherent agents.

Configurations affected

This erratum affects all systems that include:

- At least one Cortex-A17 cluster with at least two CPUs.

- At least a second cached-coherent agent.

- An interconnect that can send a second command to the Cortex-A17 cluster before it receives data back from a first ReadUnique or CleanInvalid command that has been sent to the same address, where both commands target the same Cortex-A17 cluster.

Note that the CCI-500 exhibits such behavior, but the CCI-400 does not.

Conditions

The scenario leading to the issue is:

- 0 - As a pre-requisite, the Cortex-A17 cluster must have a complete cache line from address A present in the L2 cache in the clean state.

- 1 - CPU0 in the Cortex-A17 cluster writes a cache line at address A in its entirety. An ACE MakeUnique command is sent to the AMBA/ACE bus, and the line is sent for allocation into the L2 cache.

- 2 - CPU1 in the same cluster performs a read request to the cache line at address A. The read might be the result of either a speculative access or an architecturally commited one.

- 3 - Before the MakeUnique command issued at step 1 completes, and therefore before the write executed by CPU0 is committed, an external cache coherent agent performs either:

  - A broadcast Clean and Invalidate Cache Maintenance Operation to address A, or

  - A store or a prefetch for write to address A. The prefetch for write might result of speculative execution.

- 4 - The interconnect sends the ReadUnique command resulting from step 3 to the Cortex-A17 cluster.

- 5 - The Cortex-A17 cluster answers the ACE command from step 4 on the AMBA4/ACE CR channel. The interconnect must accept the answer on the CR channel and not accept the associated data on the CD channel.

- 6 - The interconnect sends the response from the MakeUnique command issued in step 1 back to the Cortex-A17 cluster. The associated cache line gets allocated into the L2 cache.

- 7 - The read request issued by CPU1 at step 2 is serviced from the L2 cache.

- 8 - The interconnect must accept the data resulting from the ACE command in step 4 only after step 7 completes.

After this sequence, the Cortex-A17 cluster might not issue a MakeUnique command for a write targeting the cache line at address A.

Implications

The erratum might cause a data corruption.

Workaround

The workaround consists in disabling prefetch for write by setting bit [4] of the CPU diagnostic register (accessible in the cp15 register space at address cp15,0,c15,c0,1). ARM does not expect any noticeable

 Copyright ©2014-2019 Arm. All rights reserved.

performance degradation to result from this setting. To date, most configurations ARM is aware of are not susceptible to this erratum. Notably, CCI-400 based solutions are immune.

2.7. Category C

824719: Stage 2 fault might not be reported when address translation for a Cache Maintenance Operation uses a NoAccess Domain

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

A Stage2 fault on the final Stage1 translation for the address of a Cache Maintenance Operation might not be reported if the address translation indicates the CMO accesses a region using a NoAccess domain.

Configurations affected

This erratum affects all Cortex-A12 and all Cortex-A17 configurations.

Conditions

The erratum occurs if both:

- The address translation indicates the Domain for the targeted area is NoAccess
- The address translation generates one of the following faults during the final Stage1 translation:
  - Access Flag Fault
  - Translation Fault
  - External Abort

Implications

The Cache Maintenance Operation will be performed using an arbitraty physical address, and no Stage2 abort will be generated

Note that this erratum does not cause a data corruption, as all straight invalidating CMOs are translated into the corresponding clean and invalidate cache maintenance operation.

Workaround

The workaround consists, in a Virtualized system, in not making use of the Domains in V7VMSA.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

831169: ECC error on L2 DATA RAMs might report an erroneous error index

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

When an ECC error is detected on L2 DATA RAM, the "Error location" field in L2MRERRSR might report an erroneous value.

Configurations affected

Cortex-A17 configured with:

- an L2 cache size smaller than 8MB.
- ECC implemented.

Conditions

The errata is triggered if:

- an ECC error is detected on the L2 DATA RAM read access.
- the aforementioned access has been triggered by an incoming snoop request from the ACE port.

Implications

The "Error location" field in the L2MRERRSR reports bits[18:6] of the address used for the lookup, independently of the L2 cache size.

Also, when new ECC errors are detected, the "Other error count" and "Same error count" fields might be corrupted.

Workaround

There is no Software Workaround.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

833221: The Error Location field of the L2MRERRSR register might not be correctly updated

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

The Error Location field of the L2 Memory Error Syndrome Register (L2MRERRSR) might not update correctly for some ECC errors.

Configurations affected

This erratum affects all configurations of Cortex-A17 when ECC is implemented.

Conditions

The erratum occurs when an ECC error is detected on the L2 DATA RAM upon an eviction from the L2 cache to the next memory level.

Implications

The Error Location field of the L2 Memory Error Syndrome Register might contain a stale value that does not reflect the correct Index / Way where the error has been triggered.

Workaround

There is no workaround.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

844221: Stage2 Descriptors might be accessed using an incorrect Cacheability attribute

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

When performing a page table walk for the Non-secure PL0/PL1 translation regime with Stage2 translation enabled, the Stage2 descriptors might be read from memory using a Cacheability attribute different from the one configured in VTCR.IRGN0.

Configurations affected

All Cortex-A12 and all Cortex-A17 configurations are affected.

Conditions

For the erratum to occur, the Stage2 translation and the PL2 translation regime must be configured as follows:

- VTCR.IRGN0 is set to '01' or '11', indicating the memory associated with page table walks using VTTBR is Inner WriteBack.

- HSCTLR.C is set to 0.

Implications

When the erratum occurs, it might cause the descriptors to be accessed as Normal Non-Cacheable memory instead of the memory type described by VTCR.IRGN0.

Workaround

When using page table descriptors marked as residing in cacheable memory, the Hypervisor should always enable the data cache by setting HSCTLR.C to 1. ARM does not expect the affected combination to be used under normal circumstances.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

844223: UNDEFINED instructions might not UNDEF

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Executing some UNDEFINED opcodes might not raise an UNDEF exception; the processor will either execute an SDIV instruction or a NOP.

Configurations affected

This defect affects all configurations of the processor.

Conditions

The following opcodes are affected:

Thumb2 : 1111 1011 1001 Rn (1)(1)(1)(1) Rd 0001 Rm

Arm: 1111 0111 1001 (----) (----) (----) (---1) (----)

Implications

If the opcode is of the form:

Thumb2: 1111 1011 1001 Rn (1)(1)(1)(1) Rd 0001 Rm

then an SDIV instruction will be executed.

Otherwise, a NOP will be executed.

Note that this encoding is not part of the encoding that ARM guarantees will always UNDEF.

Workaround

The software should not rely on taking an UNDEF trap when executing an opcode within that encoding space. Note that this encoding is not part of the encoding that ARM guarantees will always UNDEF.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

846719: PAR might not be architecturally correct upon completion of an ATS12NS instruction

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Upon execution of an ATS12NS instruction, in case the requested translation induces both a Stage 1 permission fault and either of a stage 2 MMU fault other than a permission fault, the PAR might get updated to reflect the stage 2 fault instead of the stage 1 permission fault.

Configurations affected

All configurations of Cortex-A12 and Cortex-A17 are affected.

Conditions

The translation requested by the ATS12NS instruction must report a permission fault at stage 1, and the IPA to PA translation must induce a stage 2 fault.

Implications

The PAR is updated to reflect the stage 2 fault. It is architecturally defined that the stage 1 permission fault should be reported in the PAR.

Workaround

There is no workaround.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential

847219: CPU might take a breakpoint instead of a vector catch.

Category C

Products Affected: Cortex-A17 MPCore.

Present in: r0p0

Description

Under some software and hardware conditions, when vector catch is enabled and a breakpoint is set, a breakpoint exception is thrown instead of a vector catch exception.

Configurations affected

All configurations of Cortex-A12 and Cortex-A17 are affected.

Conditions

The erratum occurs when a breakpoint is programmed on the shadow of a predicted taken branch (that is, a branch present in the prediction structures), the branch being on the first 16-bit part of a 32-bit word and the breakpoint set on the second 16-bit part of the same 32-bit word. If the predicted target of the branch is the location of an exception vector where the vector catch exception has been enabled, a breakpoint exception might be taken instead of a vector catch exception.

Note that a vector catch exception is raised and reported correctly whenever the vector is entered as part of an exception being taken, as opposed to the vector being executed as the result of a branch to the vector code.

Implications

The IFSR/HSR will report a breakpoint exception instead of a vector catch exception.

Workaround

There is no workaround.

 Copyright ©2014-2019 Arm. All rights reserved.

Non Confidential