ARM Core
Cortex™-M1 (AT470)
Errata Notice

This document contains all errata known at the date of issue in supported releases up to and including revision r1p0 of Cortex-M1

Proprietary notice

Words and logos marked with ® or ™ are registered trademarks or trademarks of ARM Limited in the EU and other countries, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM Limited in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM Limited shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

Document confidentiality status

This document is Non Confidential.

Web address

http://www.arm.com/

Feedback on the product

If you have any comments or suggestions about this product, contact your supplier giving:

- The product name
- A concise explanation of your comments.

Feedback on this document

If you have any comments on about this document, please send an email to support-cores@arm.com giving:

- The document title
- The documents number
- The page number(s) to which your comments refer
- A concise explanation of your comments

General suggestion for additions and improvements are also welcome.

Contents

Introduction

Scope

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

- a unique defect tracking identifier
- 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

Categorisation of Errata

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

Category 1 Behavior that is impossible to work around and that severely restricts the use of the product in all, or the majority of applications, rendering the device unusable.

Category 2 Behavior that contravenes the specified behavior and that might limit or severely impair the intended use of specified features, but does not render the product unusable in all or the majority of applications.

Category 3 Behavior that was not the originally intended behavior but should not cause any problems in applications.

Change Control

10 Jun 2008: Changes in Document v3

Page	Status	ID	Cat	Summary
14	New	536213	Cat 3	Spurious debug reset acknowledge of debug reset request in SWJ-DP
15	New	536426	Cat 3	AHB protocol violations around system reset

28 Sep 2007: Changes in Document v2

Page	Status	ID	Cat	Summary
10	New	441592	Cat 3	Incorrect priority or stack pointer on LOCKUP state entry
12	New	453368	Cat 3	Spurious debug state entry from the LOCKUP state

26 Mar 2007: Changes in Document v1

First revision of document with no errata listed.

Errata Summary Table

The errata associated with this product affect product versions as below.

A cell shown thus **X** indicates that the defect affects the revision shown at the top of that column.

ID	Cat	Summary of Erratum	r0p0	r0p1	r1p0
441592	Cat 3	Incorrect priority or stack pointer on LOCKUP state entry	X		
453368	Cat 3	Spurious debug state entry from the LOCKUP state	X		
536213	Cat 3	Spurious debug reset acknowledge of debug reset request in SWJ-DP	X	X	
536426	Cat 3	AHB protocol violations around system reset	X	X	

Errata - Category 1

There are no Errata in this Category

Errata - Category 2

There are no Errata in this Category

Errata - Category 3

441592: Incorrect priority or stack pointer on LOCKUP state entry

Status

Affects: product Cortex-M1.

Fault status: Cat 3, Present in: r0p0, Fixed in r0p1.

Description

On exception entry, caused by a Fault in the HardFault handler, the processor should enter the LOCKUP state at a priority of "-1", corresponding to the HardFault priority. On exception entry, caused by a Fault in the NMI handler, the processor should enter the LOCKUP state at a priority of "-2", corresponding to the NMI priority. The stack and registers should correspond to the thread in which the Fault was detected. A Fault when reading the processor state from the stack should be handled as a Fault at the destination priority. If this is unknown due to the Fault, the processor should enter the LOCKUP state at a maximum priority of "-1".

Due to this erratum, the exception entry and exit sequences do not correctly handle imprecise faults that should lead to the processor entering the LOCKUP state.

An imprecise fault that should cause the processor to enter the LOCKUP state, and that is recognized at the same time as an exception entry or exit may either

- Enter the LOCKUP state at the incorrect priority of "-2" corresponding to NMI
- Enter the LOCKUP state at the correct priority, but with the stack pointer and registers restored

A fault on restoring the xPSR from the stack during exit from the NMI handler will

- Enter the LOCKUP state at the incorrect priority of "-2" corresponding to NMI.

Conditions

Under any of the following conditions this erratum can occur:

1. On exception entry, all of the following conditions are required:
   - Execution in the HardFault handler
   - The AHB system returns an ABORT for a write access, recognized as a Fault
   - NMI recognition co-incident with the Fault recognition
2. On exception exit, all of the following conditions are required:
   - Execution in the HardFault or NMI handler
   - The AHB system returns an ABORT for a write access, recognized as a Fault
   - The handler exit instruction is executed before the Fault is recognized
3. On exception exit, all of the following conditions are required:
   - Execution in the NMI handler
   - The AHB system returns an ABORT for the read access corresponding to the xPSR read, recognized as a Fault

Implications

Once the Processor enters LOCKUP state at a priority of "-2", an NMI can no longer cause the processor to exit the LOCKUP state. Entering LOCKUP state at an incorrect priority of "-2" as a result of a Fault in the HardFault handler or during the state restore on return from the NMI handler will prevent further NMI exceptions from being taken.

If the Stack Pointer is incorrect, it is more difficult to debug the cause of LOCKUP state entry. There is no other consequence of the Stack Pointer being incorrect.

Workaround

To facilitate debug of the cause of LOCKUP state entry, a DSB instruction can be added to the exception handlers for NMI and HardFault. The DSB should be the last instruction before the exception return instruction.

The LOCKUP state is visible outside the processor, on the LOCKUP pin. When it is no-longer necessary to debug the causes of LOCKUP state entry, the LOCKUP pin can be used to trigger a watchdog timer to reset the processor. Provided that the NMI input signal remains active until the processor executes the NMI exception handler, then the NMI handler will be called immediately after reset. There will be some delay, but little other consequence of entering LOCKUP at the incorrect priority.

453368: Spurious debug state entry from the LOCKUP state

Status

Affects: product Cortex-M1.

Fault status: Cat 3, Present in: r0p0, Fixed in r0p1.

Description

The processor should not be able to leave the LOCKUP state and enter halting mode debug state via a BKPT instruction or Breakpoint Comparator Unit (BPU) comparator match.

Due to this erratum, the processor may spuriously leave the LOCKUP state and enter halting mode debug state with the Debug Fault Status Register (DFSR) indicating a breakpoint as the reason for debug state entry.

Conditions

This erratum will only occur if all the following conditions apply:

1. A debug configuration of the processor is in use
2. C_DEBUGEN in the Debug Halting Control and Status Register (DHCSR) is set
3. The byte stored at address 0x1 is equal to 0xBE. This is when bits[15:8] of the initial Main Stack Pointer (MSP) value in a Little-endian configuration or bits[23:16] in a BE-8 configuration equal 0xBE. This data value can be decoded as a BKPT instruction.
4. The processor enters the LOCKUP state

Implications

Under the conditions described above, immediately after entering the LOCKUP state, the processor will enter halting mode debug state spuriously with bit[1] of the Debug Fault Status Register (DFSR) set to indicate a breakpoint as the reason for debug state entry.

If the C_HALT bit in the DHCSR is subsequently cleared, the processor will leave halt state and re-enter the LOCKUP state assuming no relevant processor state was changed while it was halted. Another spurious debug state entry will follow and this cycle will continue until one or more of the conditions described above are removed.

If the processor was locked up at a priority of -1 before the spurious halt mode entry, and an MMI is pended, the processor will not service the NMI whilst it is halted. It will however, service the NMI when the C_HALT bit is cleared. The effect of this erratum therefore is a delayed servicing of the NMI.

If reset occurs while the processor has been spuriously halted, the processor will reset correctly. The erratum has no effect in this case.

Workaround

To avoid this erratum completely, the value of the byte at address 0x1 can be changed to ensure that it is no longer equal to 0xBE.

If it is not possible or desirable to change the code image in this way, the following workaround can be used to limit the effect of this erratum:

The debugger can detect the erratum condition by observing that the processor has halted with:

1. Bits[5:0] of the xPSR equal to 0x2 or 0x3 (indicating an execution priority of -1 or -2)

2. DebugReturnAddress() equal to 0xFFFF_FFFE
3. Bit[1] of the DFSR set (indicating a breakpoint as the cause of debug state entry)

If all the above conditions are true, then the debugger can treat the entry to debug state as spurious and simply clear the C_HALT bit to cause the processor to return to the LOCKUP state.

536213: Spurious debug reset acknowledge of debug reset request in SWJ-DP

Status

Affects:	product Cortex-M1.
Fault status:	Cat 3, Present in: r0p0,r0p1, Fixed in r1p0.

Description

This erratum only affects the debug configuration of the processor.

In a Cortex-M1 system, debug reset is a power-on-reset. Initiating a debug reset will cause a debugger to lose connection with the target. For this reason, debugger requests for debug reset are ignored.

Debuggers can request a debug reset by setting bit [26] of the Control/Status register in the SWJ-DP. If the debug reset occurs, bit [27] of the same register should be set. If the debug reset does not occur, bit[27] should be clear. This erratum is that bit[27] will be erroneously set whenever bit[26] is set even though the debug request is ignored and no debug reset occurs.

Conditions

1. A debug configuration of the processor is in use
2. A debugger attempts to request a debug reset by setting bit[26] of the Control/Status register

Implications

This erratum is only relevant for agents that access the debug Control/Status register.

The only implication of this erratum is that any request for a debug reset will erroneously appear to have been serviced.

Workaround

For Cortex-M1 targets, a debugger should ignore bit[27] of the Control/Status register.

536426: AHB protocol violations around system reset

Status

Affects: product Cortex-M1.

Fault status: Cat 3, Present in: r0p0,r0p1, Fixed in r1p0.

Description

This erratum only affects a debug configuration of the processor.

The AHB protocol mandates that a master should not perform any non-IDLE transactions on the bus when the bus is held in reset. In Cortex-M1 systems, the bus system must be reset by SYSRESETn.

Due to this erratum, the processor may:

- Perform non-IDLE transactions during reset that were initiated by software just before reset
- Perform non-IDLE transactions after reset that were initiated by software just before reset

Conditions

This erratum may occur under the following conditions:

1. A debug configuration of the processor is in use
2. SYSRESETn is asserted during a software-initiated memory access to the AHB port
3. Either of the following is true at the time SYSRESETn is asserted:
   - a debugger access to the AHB port is in progress
   - the last AHB port access was a debugger access

Implications

The implications of this erratum are entirely system dependent. It is anticipated that most slaves that implement a reset will ignore transactions that occur during reset.

Workaround

The following workaround ensures that no accesses initiated before reset can occur after reset is de-asserted. It does not prevent accesses occurring during reset.

- The system reset controller should hold SYSRESETn asserted for at least 7 HCLK cycles at a time