



ARM Core  
ARM968E-S (AT410/AT412)  
**Errata Notice**

This document contains all errata known at the date of issue in supported releases up to and including revision r1p0 of ARM968E-S

**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 email to <mailto: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	5
ERRATA SUMMARY TABLE	7
ERRATA - CATEGORY 1	8
There are no Errata in this Category	8
ERRATA - CATEGORY 2	9
<b>407761</b> : Processor AHB bus can deadlock when prefetching Thumb instructions	9
<b>407762</b> : Processor can deadlock when prefetching ARM instructions	11
<b>409661</b> : Real monitor breakpoint missed in Thumb state	13
ERRATA - CATEGORY 3	14
<b>426961</b> : Coprocessor 15 instructions immediately following an MSR instruction may be executed with the wrong privilege level	14
ERRATA - SYSTEM	16
<b>744581</b> : AHB sequential writes to the same TCM memory from the DMA slave port can result in lost transfers for some bus masters	16

## 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.
System	Errata or possible issues that have system implications and therefore should be considered by system designers.

---

## Change Control

### 19 Oct 2010: Changes in Document v4

Page	Status	ID	Cat	Summary
16	New	744581	System	AHB sequential writes to the same TCM memory from the DMA slave port can result in lost transfers for some bus masters

### 07 Mar 2007: Changes in Document v3

Page	Status	ID	Cat	Summary
14	New	426961	Cat 3	Coprocessor 15 instructions immediately following an MSR instruction may be executed with the wrong privilege level

### 09 Nov 2006: Changes in Document v2

Page	Status	ID	Cat	Summary
9	New	407761	Cat 2	Processor AHB bus can deadlock when prefetching Thumb instructions
13	New	409661	Cat 2	Real monitor breakpoint missed in Thumb state
11	New	407762	Cat 2	Processor can deadlock when prefetching ARM instructions

## 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.

Erratum 409661 is a new ID number for a previously included Erratum ID 326713. This original erratum defect was filed against the ARM9E core, and subsequently fixed in revision r2p1 of that ARM9E core. However, this is not included in the ARM968E-S product, so the new Erratum ID 409661 has been opened specifically for the ARM968E-S. It contains the same information as the original Erratum ID 326713.

ID	Cat	Summary of Erratum	r0p0	r0p1
407761	Cat 2	Processor AHB bus can deadlock when prefetching Thumb instructions	X	
407762	Cat 2	Processor can deadlock when prefetching ARM instructions	X	
409661	Cat 2	Real monitor breakpoint missed in Thumb state	X	X
426961	Cat 3	Coprocessor 15 instructions immediately following an MSR instruction may be executed with the wrong privilege level	X	X
744581	System	AHB sequential writes to the same TCM memory from the DMA slave port can result in lost transfers for some bus masters	X	X

## Errata - Category 1

**There are no Errata in this Category**



## Errata - Category 2

### **407761: Processor AHB bus can deadlock when prefetching Thumb instructions**

#### **Status**

Affects: product ARM968E-S.

Fault status: Cat 2, Present in: r0p0, Fixed in r0p1.

#### **Description**

When the processor is prefetching instructions for the Thumb instruction set, it is possible that the processor deadlocks. The deadlock manifests itself when the ARM9 Integer Core instructs the Instruction Prefetch Buffer to perform a buffer flush due to a change of program flow while the Instruction Prefetch Buffer is completing its previous fill. The change of program flow will cause the Instruction Prefetch Buffer to issue a new burst request of two 32-bit elements to the ARM968 BIU. The result is that in the BIU, the AHB master port drops the burst request made by the Instruction Prefetch Buffer – and hence, the processor deadlocks. In this deadlock situation a debugger connection is not possible.

Prefetching Thumb instructions from the Instruction TCM are not affected by this erratum.

#### **Conditions**

The deadlock can occur under the following conditions:

- ARM968 r0p0 processor is prefetching Thumb instructions using the BIU AHB master. The erratum can be generated with perfect memory or memory that requires wait states.
- Any BIU AHB master core-to-bus ratio can produce the errata.
- Any setting of the CFGTHUMB32 primary input can produce the errata.
- The Instruction Prefetch Buffer must be empty, or about to go empty. The buffer requests a refill. This will occur as a result of a non-sequential prefetch request via a change of program flow.

#### **Implications**

For any software that is using the ARM968 processor and executing the Thumb instruction set, the Instruction Prefetch Buffer can deadlock. When the ARM9 Integer Core encounters a program flow altering instruction, it will issue a non-sequential memory request to the memory system. This non-sequential memory request will cause the Instruction Prefetching Buffer to complete its outstanding request, perform a prefetch buffer flush, and issue a new request to the BIU.

When the non-sequential request is issued to the BIU, data may be in-flight to the prefetch buffer for the previous prefetch request, which has been flushed. Due to the timing of the in-flight data and the new prefetch buffer request, the BIU can drop the new prefetch buffer request. Hence, a deadlock of the ARM968 processor results as the Instruction Prefetch Buffer is waiting for data for the new prefetch buffer request and the BIU is in the idle state as the request has been dropped.

---

**Workaround**

A software workaround for this bug can be accommodated by disabling the Instruction Prefetch Buffer, by setting bit [16] in the CP15 Configuration Control Register as follows:

```
MRC p15, 0, rX, c15, c1, 0
```

```
ORR rX, rX, #(1<<16)      ; disable Inst. Pref. Buffer
```

```
MCR p15, 0, rX, c15, c1, 0
```

This will degrade the overall performance of application since all instruction fetches will now be Non-Sequential accesses on the AHB bus. Hence, all AHB bursts will no longer be generated for any instruction AHB transactions. Performance degradation will be system dependent.

**407762: Processor can deadlock when prefetching ARM instructions****Status**

Affects: product ARM968E-S.  
Fault status: Cat 2, Present in: r0p0, Fixed in r0p1.

**Description**

When the processor is prefetching instructions for the ARM instruction set, it is possible that the processor deadlocks. The deadlock manifests itself when the ARM9 Integer Core instructs the Instruction Prefetch Buffer to perform a buffer flush due to a change of program flow while the Instruction Prefetch Buffer is completing its previous fill. The change of program flow, near a 1KB address boundary, will cause the Instruction Prefetch Buffer to issue a new burst request of four 32-bit elements to the ARM968 BIU. The BIU, due to being near the 1KB address boundary, will convert that request to a single 32-bit element request on AHB. The result of the conversion is that the BIU does not acknowledge receipt of the burst request made by the Instruction Prefetch Buffer – and hence, the processor deadlocks. In this deadlock situation, a debugger connection is not possible.

Instruction prefetching of Thumb instructions via the Instruction Prefetch Buffer are not affected by this erratum as only ARM instruction prefetching will generate a burst request of four 32-bit elements. Prefetching ARM instructions from the Instruction TCM are not affected by this erratum.

**Conditions**

The deadlock can occur under the following conditions:

- ARM968 r0p0 processor is prefetching ARM instructions using the BIU AHB master. Any AHB slave device must be capable of forcing HREADY low for the BIU AHB master request, i.e. HREADY==0 due to a slow response from a SRAM, FLASH, arbiter, etc.
- The Instruction Prefetch Buffer is empty and requests a buffer refill. The number of elements will always be equal to 4 as the change in program flow will empty the contents of the prefetch buffer.
- Prefetching must be near a 1KB boundary, such that the address bits [9:4] are set for the change of program flow instruction or instruction fetches immediately following the change of program flow instruction.
- Any BIU AHB master core-to-bus ratio can produce the erratum.

The conditions and timing that cause this erratum are very rare due the ARM9 rate of consumption of instructions from the prefetch buffer, the response time of the AHB memory slave, and being near a 1KB addressable boundary; however, when the conditions do occur, the processor will deadlock.

**Implications**

For any software that is using the ARM968 processor and executing the ARM instruction set, the Instruction Prefetch Buffer can deadlock when prefetching instructions near a 1KB address boundary. When the ARM9 Integer Core encounters a program flow altering instruction, it will issue a non-sequential memory request to the memory system. This non-sequential memory request will cause the Instruction Prefetching Buffer to complete its outstanding request, perform a prefetch buffer flush, and issue a new request to the BIU. When the non-sequential request is issued to the BIU, data may be in-flight to the prefetch buffer for the previous prefetch request, which has been flushed. Due to the timing of the in-flight data and the timing of the new prefetch buffer request, the BIU may not acknowledge the new prefetch buffer request. Hence, a deadlock of the ARM968

processor results as the Instruction Prefetch Buffer is waiting for data for the new prefetch buffer request and the BIU is in the idle state as the request has been not been acknowledged.

### **Workaround**

A software workaround for this bug can be accommodated by disabling the Instruction Prefetch Buffer, by setting bit [16] in the CP15 Configuration Control Register as follows:

```
MRC p15, 0, rX, c15, c1, 0
```

```
ORR rX, rX, #(1<<16)      ; disable Inst. Pref. Buffer
```

```
MCR p15, 0, rX, c15, c1, 0
```

This will degrade the overall performance of application since all instruction fetches will now be Non-Sequential accesses on the AHB bus. Hence, all AHB bursts will no longer be generated for any instruction AHB transactions. Performance degradation will be system dependent.

**409661: Real monitor breakpoint missed in Thumb state****Status**

Affects: product ARM968E-S.  
Fault status: Cat 2, Present in: r0p0,r0p1, Open.

**Description**

Under certain circumstances, a breakpoint on a Thumb instruction is not taken if Monitor mode debug is enabled.

In Monitor mode debug, if the core is configured for 32 bit Thumb instruction fetches, and a breakpoint is set on an odd half word address, the breakpoint is not taken if the instruction at the preceding even half word address is stalled due to an interlock.

**Conditions**

1. Debug is enabled (DBGEN is HIGH and Debug control register bit[5] is LOW)
2. 32-bit fetches in Thumb state are enabled (CFGTHUMB32 is HIGH)
3. Monitor mode debug is enabled (Debug control register bit[4] is HIGH)
4. A breakpoint is set on an odd halfword address (Address bit[1] = 1)
5. The instruction at the preceding even half word address is stalled due to an interlock

An example code sequence that causes this behavior is:

```
ADD R2, #4      ; Address = 0x100
LDR R0, [R1]    ; Address = 0x102
ADD R0, #1      ; Address = 0x104, Interlock due to data dependency on LDR
STR R0, [R2]    ; Address = 0x106, Breakpoint address
```

**Implications**

This erratum only affects Monitor mode debug operations. The normal function of the core is not affected.

**Workaround**

To workaround this erratum, use the BKPT instruction instead of setting breakpoints.

## Errata - Category 3

### **426961: Coprocessor 15 instructions immediately following an MSR instruction may be executed with the wrong privilege level**

#### **Status**

Affects: product ARM968E-S.

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

#### **Description**

It is possible to change from a privileged mode to User mode by executing the MSR instruction. The subsequent 2 instructions will be present in the pipeline when the MSR instruction is executed. The MSR instruction does not cause the pipeline to be flushed. If the instruction immediately following the MSR instruction is a Coprocessor 15 operation it will be executed as if in a privileged mode rather than user mode.

Coprocessor 15 operations will not be executed when changing from User to privileged mode as the pipeline is flushed in these cases.

This erratum does not affect the return from exceptions as this usually involves an operation to change the program counter. Changing the program counter causes the pipeline to be flushed and so subsequent Coprocessor 15 operations will be executed with the correct privileges.

#### **Conditions**

For this behaviour to be exhibited, the following conditions must exist:

1. An MSR instruction is executed that changes from a privileged mode to User mode.
2. The instruction following the MSR instruction is a Coprocessor 15 operation.

#### **Implications**

The ARM Architecture Reference Manual recommends that an Instruction Memory Barrier (IMB) sequence is executed after using an MSR to change from a privileged mode to User mode.

If this is not done the erratum allows User mode access to all Coprocessor 15 operations including reading and writing of the System Control register. The code sequence that generates this behaviour would be very unusual. In most cases changing from User to privileged mode is done on return from a procedure, and this would use an instruction that would cause the pipeline to flush. It is most unusual for User code to immediately follow privileged code, where the last privileged instruction is to change to User mode.

#### **Workaround**

The erratum can be avoided by not placing coprocessor 15 instructions immediately after an MSR instruction.

Placing a NOP instruction immediately after the MSR instruction will ensure that the correct privilege level is used for subsequent Coprocessor 15 operations.

MSR CPSR\_c, Rm

NOP

<next instruction>

## Errata - System

### **744581: AHB sequential writes to the same TCM memory from the DMA slave port can result in lost transfers for some bus masters**

#### **Status**

Affects: product ARM968E-S.

Fault status: System, Present in: r0p0, r0p1, Open.

#### **Description**

The ARM968 has an AHB-Lite slave interface that can be used to DMA data in and out of the TCM memories, of which there are three: D0TCM, D1TCM and ITCM. If multiple sequential write transfers are done across this interface to the same TCM memory and these writes are followed by an IDLE or BUSY transfer, it is possible that the next transfer sent after the BUSY or IDLE cycle will be lost. For this to occur, the TCM must make use of wait states and the AHB master must ignore the HREADYOUT response that is given for IDLE and BUSY transfers.

#### **Conditions**

1. A write transfer is made across the DMA interface to a TCM
2. A second write transfer is made across the DMA interface to the same TCM
3. The next transaction presented from the master on the AHB interface is an IDLE or BUSY transfer. AHB-Lite protocol expects HREADYOUT to always be asserted in response to IDLE and BUSY transfers.
4. HREADYOUT is de-asserted in response to the IDLE or BUSY transfer which is not allowed on the AHB protocol. This incorrect behavior will only occur if TCM wait states are used.
5. If the AHB master is not sampling HREADYOUT in response to the IDLE or BUSY transfer and proceeds to initiate a new transfer, that new transfer will not be captured by the processor and therefore will be lost.

#### **Implications**

The AHB-Lite protocol requires that an AHB slave asserts HREADYOUTD on all IDLE and BUSY transactions. Therefore, deassertion of HREADYOUTD in response to these transaction types is a violation of AHB-protocol which can result in lost transfers from masters that are properly observing the guidelines of the protocol. This can happen in any system which makes use of both TCM memories with wait states as well as a master that does not sample HREADY from the slave in response to IDLE and BUSY transactions.

#### **Workaround**

If the master checks its HREADY input even in IDLE and BUSY transfers and avoids sending transfers whenever HREADY is deasserted, this erratum will not occur.

If the master cannot be changed, another solution is to add a wrapper around the 968 which will



1. Assert a new version of HREADYOUTD in response to IDLE or BUSY transfers
2. Register all AHB control signals immediately following an IDLE or BUSY transaction
3. Send the registered version of the AHB control signals to the processor when the 968 version of HREADYOUTD is asserted.