ARM10200 Rev1 Errata List

CPU Cores Division

Document number:	ARM10200E REV1 -PRDC-000684 3.0
Date of Issue:	9/6/2001
Author	John Hudson
Authorised by:	Ty Garibay

Abstract

This document describes the known errata in the ARM10200 Rev1, ARM1020E Rev1, VFP10 Rev1 and VFPv2. designs

Keywords

Errata, bug, workaround, ARM10200, ARM1020E, VFP10.

This is a working document throughout the product lifecycle and, as such, the content may be modified as new information is uncovered.

Contents

1 ABOUT THIS DOCUMENT

1.1 Change History

Issue	Date	By	Change
0.1	8/1/2001	John Hudson	First Draft for Domino.Doc
1.0	9/4/2001	Kerry McGuire	Approved for Release
2.0	9/4/2001	Kerry McGuire	No change
3.0	9/17/2001	John Hudson / Kerry McGuire	Additional comments on existing errata – Approved for Release

1.2 References

This document refers to the following documents.

Ref.	Document No	Author(s)	Title

1.3 Scope

This document describes the errata discovered in the implementation of the ARM10200/ARM1020T/VFP10 Rev1, categorised by level of severity. Each description includes:

- where the implementation deviates from the specification

- the conditions under which erroneous behaviour occurs

- the implications of the erratum with respect to typical applications

- the application and limitations of a 'work-around' where possible

- the status of corrective action.

1.4 Terms and Abbreviations

This document uses the following terms and abbreviations.

Term	Meaning

2 CATEGORISATION OF ERRATA

Errata recorded in this document are split into three groups:

Category 1   Features which are impossible to work around and severely restricts the use of the device in all or the majority of applications rendering the device unusable.

Category 2   Features which contravene the specified behaviour and may limit or severely impair the intended use of specified features but does not render the device unusable in all or the majority of applications.

Category 3   Features that were not the originally intended behaviour but should not cause any problems in applications.

2.1 Errata Summary

3 CATEGORY 1 ERRATA

There are no errata in this group.

4 CATEGORY 2 ERRATA

4.1 VFP10: N-bit Assertion In Double Precision

Internal database reference: Arm10 Rev0 bugs #207 and Rev1 bugsr1 #302.

4.1.1 Summary

Portions of the VFP10 coprocessor undefined space may cause processor to hang if executed.

4.1.2 Description

The VFPv1 Architecture Specification states that in a double precision operation, the 22nd (D), 7th (N), and 5th (M) bits in the instruction must be set to zero. What is not defined is what will happen if these bits are not set to zero. Whilst testing these "undefined" cases, it was found that the VFP10 will hang when it encounters the N-bit asserted in a double precision instruction.

4.1.3 Conditions

To set up failing condition the N-bit set to 1 within a VFP10 coprocessor instruction for a double precision operation.

4.1.4 Implications

This category of malformed VFP10 coprocessor instruction when used within the undefined instruction space may cause the system to hang.

4.1.5 Workaround

Do not encode and execute this malformed VFP10 coprocessor instruction for undefined instruction trapping.

4.1.6 Corrective Action

Expected to be fixed in future revisions.

4.2 VFP10: Two-Coprocessor Configuration Dropping An Instruction

Internal database reference: Arm10 Rev1 bugsr1 #324.

4.2.1 Summary

Two-VFP10 coprocessor system configurations may drop either one of the coprocessor's instructions.

4.2.2 Description

If two coprocessors are connected to an ARM10 external coprocessor interface, when one coprocessor has an internal stall and the other coprocessor is issued an instruction, there is a chance that the second coprocessor will drop that instruction. It should be noted that coprocessor instructions using CP15/CP14 are excluded from this errata.

4.2.3 Conditions

To setup the fa iling condition, a two-VFP10 coprocessor system must be implemented. The ARM program running on the system then must interleave code to both coprocessors wherein serializing instructions (MCRs to non-writable registers) are NOT placed in between instruction which reference the other coprocessor.

4.2.4 Implications

Expected executed code sequence may be missing some of either VFP10 coprocessor's instructions.

4.2.5 Workaround

Whenever an instruction is to be issued to a different coprocessor than the prior coprocessor instruction, precede that instruction with a serializing MCR to a non-writable register.

For example, replace

```
        …
        GADDD r5, r6            ; unknown coprocessor
        FADDD s9, d4, s12       ; VFP10 instruction
        …
```
With
```
        …
        GADDD r5, r6            ; unknown coprocessor
        FMXR FPSID, r4          ; serializing move to non-writable register
        FADDD s9, d4, s12       ; VFP10 instruction
        …
```

4.2.6 Corrective Action

Expected to be fixed in future revisions.

4.3 VFP10: Coprocessor Interface Failure On 2 Iteration STC Instruction

ARM Bug tracking database entry: Arm10 Rev1 bugs r1 #331.

4.3.1 Summary

The VFP has demonstrated a condition that may result in lost data (silent data corruption) under certain circumstances.

4.3.2 Description

Silent data corruption may spuriously occur on the second transfer of VFP10 STC instruction containing exactly 2 iterations.

4.3.3 Conditions

The conditions under which the failure is exhibited requires at least three instructions.

The first one must cause a stall condition of the second instruction in the core E stage, asserting ASTOPCPE in the first stage of the second instruction.

The second instruction must be a floating-point store of exactly 2 iterations (this could be a FSTMS rx, {sx-sy}, with y-x == 2 or 3, selecting either 3 or 4 single-precision data to be transferred to the ARM, or a FSTMD or FSTMX rx, {dx- dy}, with y-x == 1, selecting exactly 2 double-precision data to be transferred to the ARM.)

The third instruction must be a floating-point store, either STC, MRC, or MRRC.

4.3.4 Implications

The indication is that the data of second transfer of the FSTMS, FSTMD or FSTMX in the second position is lost, i.e., is not written to memory, leaving the prior contents of the memory address of the second transfer unchanged. No exception or other indication of this event is made.

4.3.5 Workaround

1. Do not execute VFP store instructions of 2 iterations.
2. Do not execute VFP store instructions of 2 iterations followed immediately by a VFP store.
3. Do not execute two VFP store instructions in immediate succession.
Note that this only applies to VFP10 Rev1, VFPv2 architecture.

4.3.6 Corrective Action

Expected to be corrected on future releases.

4.4 VFP10: RunFast Mode Limitations When Exceptions Are Enabled

4.4.1 Summary

A problem with preserving the source registers may exist for certain combinations of VFP FPSCR control bits.

4.4.2 Description

The following code is an example of code that could fail if some exceptions are enabled. The `flds` instruction may overwrite the source register, if an exception occurred on the `fadds`. This would prevent support code from recalculating the `fadds` because the source operands have changed.

```
…
fadds s2, s3, s4
flds s3, [r1]
…
```

4.4.3 Conditions

The supported configurations are as follows:

```
FZ    DN    exc enables        Mode
------------------------------------------
 0    X              X         non-RunFast
 X    0              X         non-RunFast
 1    1          all 0         RunFast
 1    1        any set         UNPREDICTABLE on VFP10
```

4.4.4 Implications

Enabling exceptions while FZ==DN==1 will cause an UNPREDICTABLE result if exceptions are taken.

4.4.5 Workaround

Do not set FZ==DN==1 with any exceptions enabled.

4.4.6 Corrective Action

The UNPREDICTABLE configuration space (FZ==DN==1 and some exception bits enabled) will become defined as non-RunFast in future implementations.

5 CATEGORY 3 ERRATA

5.1 ARM10200 Off-Chip AHB Interface

ARM Bug tracking database entry: Arm10 Rev1 bugsr1 #331.

5.1.1 Summary

The ARM10200 Off-Chip AHB Interface has intentionally omitted the SPLIT and RETRY response features from ARM10200 test chip.

5.1.2 Description

During the design of the ARM10200 it was decided to simplify the off-chip AHB interface to resemble the current proposal for an 'AHB Lite' interface. So no support was built into the AHB bridge, which controls the off-chip AHB interface to provide any support for RETRY/SPLIT responses.

Only the OK and ERROR responses are supported.

5.1.3 Conditions

The missing support for RETRY/SPLITs is an integrated part of the AHB bridge design.

5.1.4 Implications

No external (off-chip) AHB slave should ever respond with a RETRY or a SPLIT response. This should have no functional impact, as the ARM10200 is the only AHB master on the off-chip AHB bus.

5.1.5 Workaround

Use the de-assertion of HREADYX to delay reads and writes, if an external AHB slave cannot respond immediately.

5.1.6 Corrective Action

None planned.