



ARM PrimeCell
AHB Memory Controller (PL241)
Errata Notice

This document contains all errata known at the date of issue in releases up to and including revision r0p1-00rel0 of AHB SRAM Controller

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:errata@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
410562: Memory interface locks up when EBIBACKOFF asserted during turnaround time of burst finishing at boundary	8
ERRATA - CATEGORY 2	10
372884: Mode updates not holding off subsequent transactions	10
391189: Write data lost when a read transfer is broken by a write while previous writes are still pending	11
392506: Unlocking AXI transaction missing when a single locked read closely follows a locked write	12
401306: The Bridge can fail to stall the AHB writes after dealing with a broken read	14
410561: tAVH violation on reads in mux-mode operation	16
414339: Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave	17
415054: Error when asynchronous write follows a synchronous write and is to a different chip-select	18
415513: CS remaining asserted between transactions is not supported by CRAM 1.0	19
ERRATA - CATEGORY 3	20
408515: Memory Configuration Register does not indicate mux_mode correctly	20
409123: Setting tWP to 1 in async mux_mode gives a tWP value of 0.	21
418214: Register update mechanism mismatching manager commands	22
ERRATA - DOCUMENTATION	24
381891: Row boundary behaviour not documented	24
400894: Burst align bit behaviour different to that documented	25
408029: Discrepancy in waveform in TRM	26
ERRATA – DRIVER SOFTWARE	27
There are no Errata in this Category	27

Introduction

Scope

This document describes errata categorised 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

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

22 Mar 2007: Changes in Document v6

Page	Status	ID	Cat	Summary
19	New	415513	Cat 2	CS remaining asserted between transactions is not supported by CRAM 1.0
18	New	415054	Cat 2	Error when asynchronous write follows a synchronous write and is to a different chip-select
22	New	418214	Cat 3	Register update mechanism mismatching manager commands
26	New	408029	Doc	Discrepancy in waveform in TRM

04 Dec 2006: Changes in Document v5

Page	Status	ID	Cat	Summary
8	New	410562	Cat 1	Memory interface locks up when EBIBACKOFF asserted during turnaround time of burst finishing at boundary
16	New	414339	Cat 2	Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave
15	New	410561	Cat 2	tAVH violation on reads in mux-mode operation
9	New	372884	Cat 2	Mode updates not holding off subsequent transactions
18	New	409123	Cat 3	Setting tWP to 1 in async mux_mode gives a tWP value of 0.
17	New	408515	Cat 3	Memory Configuration Register does not indicate mux_mode correctly

27 Sep 2006: Changes in Document v4

Page	Status	ID	Cat	Summary
14	New	401306	Cat 2	The Bridge can fail to stall the AHB writes after dealing with a broken read

10 Aug 2006: Changes in Document v2

Page	Status	ID	Cat	Summary
12	New	392506	Cat 2	Unlocking AXI transaction missing when a single locked read closely follows a locked write
11	New	391189	Cat 2	Write data lost when a read transfer is broken by a write while previous writes are still pending
25	New	400894	Doc	Burst align bit behaviour different to that documented
24	New	381891	Doc	Row boundary behaviour not documented

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	r0p0-00rel0	r0p1-00rel0
381891	Doc	Row boundary behaviour not documented	X	
400894	Doc	Burst align bit behaviour different to that documented	X	
408029	Doc	Discrepancy in waveform in TRM	X	
410562	Cat 1	Memory interface locks up when EBIBACKOFF asserted during turnaround time of burst finishing at boundary	X	
372884	Cat 2	Mode updates not holding off subsequent transactions	X	
391189	Cat 2	Write data lost when a read transfer is broken by a write while previous writes are still pending	X	
392506	Cat 2	Unlocking AXI transaction missing when a single locked read closely follows a locked write	X	
401306	Cat 2	The Bridge can fail to stall the AHB writes after dealing with a broken read	X	
410561	Cat 2	tAVH violation on reads in mux-mode operation	X	
414339	Cat 2	Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave	X	
415054	Cat 2	Error when asynchronous write follows a synchronous write and is to a different chip-select	X	
415513	Cat 2	CS remaining asserted between transactions is not supported by CRAM 1.0	X	
408515	Cat 3	Memory Configuration Register does not indicate mux_mode correctly	X	
409123	Cat 3	Setting tWP to 1 in async mux_mode gives a tWP value of 0.	X	
418214	Cat 3	Register update mechanism mismatching manager commands	X	

Errata - Category 1

410562: Memory interface locks up when EBIBACKOFF asserted during turnaround time of burst finishing at boundary

Status

Affects: product AHB SRAM Controller .

Fault status: Cat 1, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

PSRAM devices deassert the wait signal at the end of a burst that finishes at or about to cross a page boundary.

The wait signal is sampled on the feedback clock which is only driven during memory transactions. If the last rising edge of the feedback clock does not sample wait high after the transaction then the wait signal will remain low internally until the next memory access starts. The last value sampled on the wait signal depends on a combination of clock speed, feedback clock delay and the time between chip-select deassertion and the wait signal returning to its high impedance state.

If the sampled wait signal remains low and a turnaround time is required i.e. between a read and write then if EBIBACKOFF is asserted during that turnaround time the memory interface will lock-up.

This problem can occur with all PSRAM devices.

Implications

Under the above described conditions, the memory interface will lock up and will never release the EBI.

Workaround

none

Errata - Category 2

372884: Mode updates not holding off subsequent transactions

Status

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The PL350 provides a mechanism for synchronising the update of a memory configuration register with its own internal configuration registers.

Following an update, subsequent commands should be formatted using the new settings. However, because the command fifo in the memory interface can be populated immediately prior to the update occurring, the commands already in the fifo when the update occurs will be incorrectly formatted for the new settings.

Commands during an update should be throttled in the format block such that the final command that performs the update is the only command active in the memory interface at that time.

Implications

This defect will affect a system which is accessing a chip whilst performing an update which, for example, changes burst length to a smaller value.

Typically it is expected that the change would be from default (smaller) burst lengths to longer, in which case the resulting behaviour should not cause any issues.

Workaround

There is no workaround for this issue.

391189: Write data lost when a read transfer is broken by a write while previous writes are still pending**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The problem occurs when a specific combination of AHB transfers are routed to the AHB Memory Controller. Also the internal memory controllers must respond in a certain way.

The AHB sequence is

- A burst of writes marked as "bufferable" HPROT[2] = 1'b1.
- A read burst occurs.
- A single bufferable write occurs and breaks the read burst.

The internal memory controller to which the initial write burst occurs must be busy. The data from the write burst is held in the AHB interface because the internal memory controller is busy. The first transaction of the read burst is completed. The address phase of the single write is accepted by an internal memory controller. The AHB interface is still holding the data from the first write burst. The AHB interface drives hready high to indicate the next address phase can be accepted but the write data isn't stored because the AHB interface is still holding the data from the write burst.

The data phase of the single write occurs to the internal memory controller but the data is incorrect. The WLAST and WSTRB signal may also be incorrect as they will be generated from a subsequent AHB transfer. This means the incorrect data that is passed to the internal memory controller may have incorrect write strobes. The WLAST may be low, failing to indicate that this is the last data transfer in the single burst. This breaks the protocol but will not cause internal memory controller to lockup.

Implications

The data from AHB single, bufferable writes may be written incorrectly to memory.

Workaround

The error only occurs because bufferable writes occur followed by a read burst that is "broken" by a single bufferable write. AHB INCR bursts of non-multiples of 4 are also interpreted as "broken" bursts by the speculative reading and writing.

If "broken" read bursts are avoided this will not occur.

If bufferable write bursts are avoided this will not occur.

This can be guaranteed if the HPROT[2] is tied low

392506: Unlocking AXI transaction missing when a single locked read closely follows a locked write**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

Locked access means that only a particular AHB port can access either the DMC or SMC. All of the other ports must wait until the locked access is completed.

There is a case when the protocol of the interconnect inside PL24x will be broken. A "locked" stream of data access will be started but not properly finished.

The error will occur in a quite specific case.

- 1) An AHB sequence of locked transactions occurs
- 2) The sequence is followed by some idle cycles
- 3) A second locked sequence occurs but only contains a single read transaction
- 4) Idle cycles or any further accesses occur

The interconnect requires an extra "unlocking" transfer to occur at the end of each locked burst to ensure that the interconnect is unlocked. This unlocking transfer is added after the burst described in 1). If 3) occurs before 1)'s unlocking transfer is completed then the unlocking transfer that should be associated with 3) is not issued. In this case the interconnect will remain locked. No other AHB ports will be able to access the locked DMC or SMC. The interconnect may be inadvertently unlocked by subsequent accesses but this is entirely dependent on the address of the subsequent accesses.

ARM processors only use locked access to perform the SWP instruction which is a read modify write. These will always occur in pairs so if the AHB system doesn't break locked pairs then the problem will never arise. For generic systems that may use locked bursts, if the AHB arbitration will not break locked bursts then the problem will never arise.

Implications

Locked access means that only a particular AHB port can access either the DMC or SMC. All of the other ports must wait until the locked access is completed.

If the interconnect is not unlocked, no other AHB ports will be able to access the locked DMC or SMC. The interconnect may be inadvertently unlocked by subsequent AHB accesses.

In a system, this would look like specific AHB port(s) being denied access to a specific memory controller for undefined and possibly infinite time periods.

Workaround

The error only occurs if a single locked read occurs after a locked write burst.

A single locked read is useless as it is not locked to anything it may aswell have been just a read.

To avoid a single, locked read occurring there are a number of options.

Don't use locked transactions

Don't allow the AHB arbitration to break locked bursts

Don't use masters that issue single locked reads

Ensure any locked transaction is followed by an unlocked transaction (read or write) to the same memory location (therefore unlocking the interconnect).

401306: The Bridge can fail to stall the AHB writes after dealing with a broken read**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The bridge has 4 hazard buffers which captures the address of write transfers. The hazard buffers are cleared on getting the relevant BVALID signal. Stall signal is asserted by the hazard detection logic if all the 4 hazard buffers are filled.

The corner case arises in the following scenario :- 1. Three single buffered write are accepted on AHB side but not completed in AXI. 2. A INCR read comes in which gets broken by a 4th single buffered write which fills up the hazard buffer causing hazard detection logic to assert stall. 3. On completion of the remaining read data beats in AXI the bridge accepts next transaction instead of stalling

W <a1> <d1> word SINGLE P0100 - fills 1st slot

W <a2> <d2> word SINGLE P0100 - fills 2nd slot

W <a3> <d3> word SINGLE P0100 - fills 3rd slot

R 00000060 00000060 word INCR 0100

S 00000064

S 00000068

W <a4> <d4> word SINGLE P0100 - breaks the read and fills the 4th (last) slot causing stall to be asserted.

Implications

On completion of the remaining read data beats of the broken burst, the bridge comes out of broken read state and accepts next transaction without checking for the stall state.

The hazard buffer is currently full and stores addresses of 4 buffered writes that are yet to complete. If the read address falls within the 4K granularity of any address in hazard buffer, accepting such an address may result in a RAW hazard.

Workaround

The error occurs in the following scenario :-

1. Three single buffered write are accepted on AHB side but not completed in AXI.
2. A INCR read comes in which gets broken by a 4th buffered write which fills up the hazard buffer causing hazard detection logic to assert stall

To avoid this scenario there are a number of options :-

- * Don't break read burst[Undefined INCR not a multiple of 4 is considered as broken read]
- * Don't use buffered writes if successive reads are within 4K granularity of the writes

410561: tAVH violation on reads in mux-mode operation**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

When reads are made to mux-mode memory devices, the timing between the deassertion of ADV and the address changing (tAVH) is violated.

This is because the address is changed on the same clock edge as ADV is deasserted.

tAVH is a mux-mode parameter and hence this violation does not occur with non-mux-mode memory devices.

Implications

This tAVH violation would cause protocol errors on the memory interface and could result in data corruption.

Workaround

This problem can be avoided by not performing async mux-mode reads. Synchronous mux-mode reads are not affected by this defect.

414339: Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

When you send a locked transfer on any AHB port of the memory controller followed by a locked transfer to a different slave on the same AHB port, the memory controller will lock up the next time the same AHB port is selected for a transfer.

In other words the problem occurs when on the same port, HSEL and HMASTLOCK are active for the first transfer and only HMASTLOCK is active for the second transfer. The memory controller may lock up when next transfer occurs on that port.

Implications

This erratum can cause HREADY on the AHB port to be never returned and hence cause the system to hang.

Workaround

None.

415054: Error when asynchronous write follows a synchronous write and is to a different chip-select**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

When an asynchronous write immediately follows a synchronous write and is therefore to a different chip-select, then the asynchronous write is treated as a synchronous read.

Implications

This erratum results in the asynchronous write being treated as a synchronous read by the memory device.

Workaround

Program all chip selects to run in the same mode of operation.

415513: CS remaining asserted between transactions is not supported by CRAM 1.0**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 2, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The SMC does not deassert chip-select between immediately consecutive transfers to memory. This behaviour is not supported by CRAM 1.0

Implications

Not deasserting CS between transactions could cause undefined behaviour from the memory. One of the results could be the memory returning incorrect data for back to back read transfers.

Workaround

None.

Errata - Category 3

408515: Memory Configuration Register does not indicate mux_mode correctly

Status

Affects: product AHB SRAM Controller .

Fault status: Cat 3, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The Memory Configuration Register does not correctly report if an interface is in mux_mode as stated in the TRM.

Implications

none

Workaround

none

409123: Setting tWP to 1 in async mux_mode gives a tWP value of 0.**Status**

Affects: product AHB SRAM Controller .

Fault status: Cat 3, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

Setting tWP to 1 in async mux_mode gives a write pulse width of 0.

Implications

none

Workaround

To work around the problem set tWP = 2 which will give a longer but valid write pulse width.

418214: Register update mechanism mismatching manager commands

Status

Affects: product AHB SRAM Controller .

Fault status: Cat 3, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The PL350 controller series implement a mechanism to synchronise updates to format and timing configuration registers with updates to memory devices' internal registers.

If a "ModeReg And UpdateRegs" direct command is applied whilst a "ModeReg" direct command to the same chip is being processed in the controller's pipeline then the update logic incorrectly updates the controller's internal registers when the first ModeReg command completes on the memory interface.

For example, a PSRAM device has a Bus Configuration Register (BCR) that dictates parameters such as burst length. A use case for programming the controller and memory to a new burst length is to issue a "ModeReg and UpdateRegs" direct command.

PSRAM devices also contain a Refresh Configuration Register (RCR) to control its self-refresh function. A use case for programming the memory to a new setting is to issue a "ModeReg" direct command.

If the RCR update is performed immediately prior to the BCR, the described error condition could occur.

Implications

If an AXI access is accepted into the format stage of the pipeline after the internal registers have updated, but before the second ModeReg command, then it will be formatted with the new settings. This will result in unpredictable behaviour.

Workaround

The recommended workaround is to always perform "ModeReg and UpdateRegs" commands before "ModeReg" commands.

For example, in the case described above involving PSRAM, perform the BCR then the RCR.

Errata - Documentation

381891: Row boundary behaviour not documented

Status

Affects: product AHB SRAM Controller .

Fault status: Doc, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

The documentation does not specify the behaviour of the controller with respect to memory row boundaries.

The controller has no knowledge of row boundary information so cannot prevent memory bursts crossing such boundaries. However, the burst alignment functionality of the controller allows bursts to always be aligned, and not cross, memory burst address boundaries. Since memory burst boundaries are smaller than row boundaries, in this mode row boundaries will not be crossed.

Implications

Some memories, for example Cellular RAM, do not support row boundary crossing. The burst_align bit of the set_opmode register must be left at 1'b0 so that bursts are aligned to memory burst address boundaries. This will ensure that memory row boundaries are never crossed.

Workaround

none

400894: Burst align bit behaviour different to that documented**Status**

Affects: product AHB SRAM Controller .

Fault status: Doc, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

From the TRM (smc_set_opmode and smc_opmode sections):

burst_align 1'b0 - memory burst may cross a row boundary

burst_align 1'b1 - memory tranfers are aligned to a memory burst boundary

The behaviour of the RTL is the inverse of this, i.e.

burst_align 1'b0 - memory tranfers are aligned to a memory burst boundary

burst_align 1'b1 - memory burst may cross a row boundary

The reset value of this bit is 1'b0

Implications

none

Workaround

none

408029: Discrepancy in waveform in TRM**Status**

Affects: product AHB SRAM Controller .

Fault status: Doc, Present in: r0p0-00rel0, Fixed in r0p1-00rel0.

Description

In Fig 2-17 of the TRM, the WE waveform is incorrect. It is shown as being asserted, that is going low, many cycles after CS gets asserted.

WE should be shown to go low, that is get asserted on the same cycle as CS.

WE in this figure should also be shown to be asserted for twp+2 cycles instead of twp, since in asynchronous mux-mode WE is asserted for twp+2 cycles.

Implications

none

Workaround

none

Errata – Driver Software

There are no Errata in this Category