



ARM PrimeCell
AHB Memory Controller (PL244)
Errata Notice

This document contains all errata known at the date of issue in releases up to and including revision r0p1-00rel0 of AHB DDR & NAND 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	6
ERRATA SUMMARY TABLE	9
ERRATA - CATEGORY 1	10
428335: No support for INCR or broken bursts when accessing NAND	10
ERRATA - CATEGORY 2	12
378649: Bridge hazard detection can fail at the end of a locked burst	12
391189: Write data lost when a read transfer is broken by a write while previous writes are still pending	14
392506: Unlocking AXI transaction missing when a single locked read closely follows a locked write	15
393151: Following a period with nand_booten asserted, interrupt not always cleared	17
400134: tRFC violation when entering low-power mode	18
400912: Arbiter can cause lock up if a qos_min transaction has non-qos_min dependancy's	19
401306: The Bridge can fail to stall the AHB writes after dealing with a broken read	20
414339: Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave	22
ERRATA - CATEGORY 3	23
381382: Memory FSM incorrectly coded for use with NAND boot enable	23
395374: Minimum-latency QoS is not enabled with QoS-Enable	24
401005: Incomplete set of synthesis constraints	25
404770: NAND boot mode problem for some NAND memory device protocols	27
437116: Performance impact of breaking up bursts to NAND memory	28
ERRATA - DOCUMENTATION	29
408924: Burst align bit behaviour different to that documented	29
437114: The behaviour of bursts to NAND devices is not described in the TRM	30
442367: Use of NAND Memory in multi-master systems.	31
442375: Clarification of restrictions when building NAND command phase transactions.	32
442624: Restrictions on the use of ClearCS and End Commands not documented in TRM	33
ERRATA – DRIVER SOFTWARE	34

There are no Errata in this Category

34

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

14 Jun 2007: Changes in Document v7

Page	Status	ID	Cat	Summary
10	Updated	428335	Cat 1	No support for INCR or broken bursts when accessing NAND
12	Updated	378649	Cat 2	Bridge hazard detection can fail at the end of a locked burst
28	New	437116	Cat 3	Performance impact of breaking up bursts to NAND memory
23	Updated	381382	Cat 3	Memory FSM incorrectly coded for use with NAND boot enable
33	New	442624	Doc	Restrictions on the use of ClearCS and End Commands not documented in TRM
32	New	442375	Doc	Clarification of restrictions when building NAND command phase transactions.
31	New	442367	Doc	Use of NAND Memory in multi-master systems.
30	New	437114	Doc	The behaviour of bursts to NAND devices is not described in the TRM

16 Mar 2007: Changes in Document v6

Page	Status	ID	Cat	Summary
10	New	428335	Cat 1	Primecell does not support INCR or broken bursts when accessing NAND

04 Dec 2006: Changes in Document v5

Page	Status	ID	Cat	Summary
22	New	414339	Cat 2	Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave
29	New	408924	Doc	Burst align bit behaviour different to that documented

27 Sep 2006: Changes in Document v4

Page	Status	ID	Cat	Summary
27	New	404770	Cat 3	NAND boot mode problem for some NAND memory device protocols
20	New	401306	Cat 2	The Bridge can fail to stall the AHB writes after dealing with a broken read

15 Aug 2006: Changes in Document v3

Page	Status	ID	Cat	Summary
19	New	400912	Cat 2	Arbiter can cause lock up if a qos_min transaction has non-qos_min dependancy's
18	New	400134	Cat 2	tRFC violation when entering low-power mode
17	New	393151	Cat 2	Following a period with nand_booten asserted, interrupt not always cleared
15	New	392506	Cat 2	Unlocking AXI transaction missing when a single locked read closely follows a locked write
14	Updated	391189	Cat 2	Write data lost when a read transfer is broken by a write while previous writes are still pending
12	Updated	378649	Cat 2	Bridge hazard detection can fail at the end of a locked burst
25	New	401005	Cat 3	Incomplete set of synthesis constraints

24	New	395374	Cat 3	Minimum-latency QoS is not enabled with QoS-Enable
23	Updated	381382	Cat 3	Memory FSM incorrectly coded for use with NAND boot enable
33	New	400894	Doc	Burst align bit behaviour different to that documented

30 May 2006: Changes in Document v2

Page	Status	ID	Cat	Summary
14	New	391189	Cat 2	Write data lost when a read transfer is broken by a write while previous writes are still pending

15 May 2006: Changes in Document v1

No Errata in this document revision

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	r0p0-01rel0	r0p1-00rel0
408924	Doc	Burst align bit behaviour different to that documented	X	
437114	Doc	The behaviour of bursts to NAND devices is not described in the TRM		X
442367	Doc	Use of NAND Memory in multi-master systems.	X	X
442375	Doc	Clarification of restrictions when building NAND command phase transactions.	X	X
442624	Doc	Restrictions on the use of ClearCS and End Commands not documented in TRM		X
428335	Cat 1	No support for INCR or broken bursts when accessing NAND	X	
378649	Cat 2	Bridge hazard detection can fail at the end of a locked burst	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	
393151	Cat 2	Following a period with nand_booten asserted, interrupt not always cleared	X	
400134	Cat 2	tRFC violation when entering low-power mode	X	
400912	Cat 2	Arbiter can cause lock up if a qos_min transaction has non-qos_min dependancy's	X	
401306	Cat 2	The Bridge can fail to stall the AHB writes after dealing with a broken read	X	
414339	Cat 2	Bridge locks up for two successive locked transfers on same port and second transfer is to a different slave	X	
381382	Cat 3	Memory FSM incorrectly coded for use with NAND boot enable	X	
395374	Cat 3	Minimum-latency QoS is not enabled with QoS-Enable	X	
401005	Cat 3	Incomplete set of synthesis constraints	X	
404770	Cat 3	NAND boot mode problem for some NAND memory device protocols	X	
437116	Cat 3	Performance impact of breaking up bursts to NAND memory		X

Errata - Category 1

428335: No support for INCR or broken bursts when accessing NAND

Status

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 1, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

The PL240 bridge internally changes INCR accesses to INCR4.

The PL240 bridge also internally completes any AHB interface broken bursts . This means that extra data beats may be added to complete the burst.

-- Read Bursts

For read bursts, extra data is read and discarded if not required by the AHB interface.

This works for SRAM and dynamic memories where the address is passed with the transaction and read data can be repeatedly requested.

However, NAND memories contain their own address pointer and data cannot be read multiple times.

For example an INCR8 that is broken after 4 beats and then completed by the master. The AHB interface would receive data from addresses 0x0 0x1 0x2 0x3 and 0x8 0x9 0xa 0xb because data from address 0x4 0x5 0x6 0x7 will be discarded.

The internal address pointer in the NAND device has moved on.

-- Write bursts

For write bursts, extra data is written with strobes indicating that the data should not be written to memory. This works for SRAM and dynamic memories where the address is passed with the transaction and strobes are used for write data.

However, NAND memories contain their own address pointer and write strobes are not used.

For example an INCR8 that is broken after 4 beats and then completed by the master. The NAND array would receive the first 4 valid data beats, then 4 extra beats with strobes indicating they should not be written to memory. The strobes are not used by the NAND device so the NAND array is corrupted with the extra beats of data.

The internal address pointer in the NAND device has moved on.

The broken burst cause of this issue can be avoided by careful system design to ensure bursts cannot be broken.

The same issue occurs when INCR bursts are extended to INCR4 unless you can ensure that INCR bursts are always in groups of 4.

Implications

Incorrect data can be read or written to a NAND device.

Workaround

None

Errata - Category 2

378649: Bridge hazard detection can fail at the end of a locked burst

Status

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

The bridge contains hazard detection logic. This catches the case when a read follows closely after a write to that location.

The hazard detection is required because the bridge can issue an early write response to bufferable write transactions. There is a potential hazard if a read is requested to that same address as the bufferable write. The DMC has a queue of transactions to process and can reorder them for efficiency. The read could be processed before the write, returning the old data. The hazard detection tracks addresses of bufferable write transactions and stalls any reads that could impact them.

The hazard detection can work incorrectly at the end of a locked burst. The situation is described below

- * locked write(s)
- * unlocked write to address A0
 - unlocking burst is added to clear locks
 - A0 is added to hazard detection logic
- * read from address A0
 - the read is stalled while A0 is in hazard detection buffer
 - the first bvalid response is a result of the unlocking AXI
 - this incorrectly clears the A0 hazard monitor
 - the read is allowed to occur

Theoretically the write hasn't completed so the read could be performed before the write.

In practice this is unlikely to cause a problem because the write to A0 is the only thing for the DMC to do so it should start doing this before the read enters the queue. So they can't be re-ordered. However if this situation occurs when the DMC has been busy and required dynamic memory refreshes have built up there will be multiple items in the queue. The read and write to A0 may be re-ordered and the data returned would be the old data, not the new data.

Implications

This errata would result in "old" data being read from the dynamic memory and not the data you had just written to there.

Workaround

The problem occurs after a locked burst when a subsequent bufferable write occurs and a read occurs to that location written to.

This can be avoided by

- not using locked bursts
- not using bufferable writes

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

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, 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 DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, 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 as well 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).

393151: Following a period with nand_booten asserted, interrupt not always cleared**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

The signal nand_booten is asserted when the controller is used to boot from NAND memory devices. The controller will suspend operation during busy periods by internally waiting for the interrupt before continuing.

When nand_booten is de-asserted, it is possible for an interrupt to be outstanding. If this interrupt is not cleared before the next read, software may try to access the nand chip whilst it is busy.

Implications

If the interrupt is not cleared when exiting boot mode, on the first read to NAND software may incorrectly assume the busy period has expired and access the memory whilst it is busy.

Workaround

The fix for this involves updating the fsm to clear the interrupt when entering the data state, and ensuring that the boot_busy flag is only set when moving out of command to idle state.

A software fix would be to always manually clear the interrupt upon exit of nand_booten mode.

400134: tRFC violation when entering low-power mode**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

When entering low-power mode PL340 issues a precharge_all command to all chips (if there are any open rows) followed by a self-refresh command to all chips.

However, the bank_fsms only check the banks for their chip select before issuing a command.

As this precharge_all command affects all the chips it means that if there is a single idle chip the command will be issued.

This may violate tRFC of another chip if that chip has just been issued with a refresh.

Note that in single chip configurations this is not a problem.

Implications

For multi-chip configurations it is possible that a memory device tRFC (auto refresh time) may be violated.

Workaround

To avoid this potential problem of unknown behaviour in a memory device with a finished design it would be necessary to avoid entering low power mode.

400912: Arbiter can cause lock up if a qos_min transaction has non-qos_min dependancy's**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

In the arbitration algorithm QOS_min (which can be set with qos_override) has the highest priority followed by QOS_max.

However, dependents (i.e. previous outstanding transactions with the same ID) are only raised to QOS_max when flagged as hazards on a new high priority transaction.

The arbiter will prepare the bank for the earliest QOS_min transaction first ie the new high priority transaction. Once the bank is prepared the transaction cannot be available to the arbiter as it has dependents and it is removed from the arbitration algorithm. The next highest priority entry is started.

There are two possible causes of the failure

- 1) A later QOS_min transaction to a different row of the same bank. The arbiter would prepare the bank for the second QOS_min transaction, causing the first to re-appear as a qos_min closed row access. This then becomes the highest priority entry in the queue creating the loop.
- 2) A dependent access to a different row of the same bank.

Any dependents that would cause the row required by the QOS_min read to be closed will create the loop.

Implications

It is possible to get stuck in an infinite loop of PRECHARGE and ACTIVATE commands.

Workaround

It is possible to avoid this problem if it can be ensured that the qos_override (or qos_min) for a particular ID are never transitioned from 0 to 1 when that ID has outstanding reads.

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

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, 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

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

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 2, Present in: r0p0-01rel0, 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.

Errata - Category 3

381382: **Memory FSM incorrectly coded for use with NAND boot enable**

Status

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 3, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

When in NAND boot mode, the controller uses the interrupt to pause operation during busy periods.

At the end of each command access the interrupt is cleared, so that it can be set again after the busy period has elapsed.

The controller will resume operation when the interrupt is set again (on the rising edge of the NAND busy signal).

An error in the FSM causes the interrupt to be cleared before the data read can start, and therefore the controller gets stalled indefinitely.

Implications

The NAND boot feature cannot be guaranteed to work as expected.

Workaround

None

395374: Minimum-latency QoS is not enabled with QoS-Enable**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 3, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

For minimum-latency quality-of-service (QoS) the programmer has access to the id_<n>_cfg registers through the APB interface at address locations 0x100-0x200.

By setting bit[1] (qos_min) of a given register, minimum latency QoS is set for a particular ID. It is intended that QoS is only activated if bit[0] (qos_enable) is also set.

However, if bit[1] is set, qos_min is applied to the ID irrespective of the value of bit[0].

Implications

If qos_min is set without qos_enable, minimum latency QoS will be applied in error. This will improve the read access latency of a particular ID to the detriment of other IDs, whose read access latency may get worse.

Workaround

qos_min (bit[1]) should not be set without qos_enable (bit[0]) in the id_<n>_cfg registers.

401005: Incomplete set of synthesis constraints**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 3, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

Synthesis constraints are incomplete. When the product is synthesised as delivered, it generates violations. The following changes should be made to the scripts:

1. The following changes need to be made in pl244_ahbmc_constraints.tcl:

1.1 dmc_ebigrant has been constrained too tightly in the design. The constraint should be relaxed from 70% to 15%.

```
set_input_delay $MCYCLE70pc -max -clock DMC_MCLK_V [get_ports dmc_ebigrant]
to
```

```
set_input_delay $MCYCLE15pc -max -clock DMC_MCLK_V [get_ports dmc_ebigrant]
```

Please make sure MCYCLE15pc is defined in your script as

```
set MCYCLE15pc [ expr 0.15 * $rm_dmc_mclock_period ]
```

1.2 You must add the following multicycle path constraints:

```
# multicycle path physical HCLK to dmc_mclx2n
set_multicycle_path 2 -setup -from [get_clocks HCLK_P] -to [get_clocks DMC_MCLKX2N_P]
set_multicycle_path 1 -hold -from [get_clocks HCLK_P] -to [get_clocks DMC_MCLKX2N_P]
```

```
# multicycle path virtual HCLK to dmc_mclx2n
set_multicycle_path 2 -setup -from [get_clocks HCLK_V] -to [get_clocks DMC_MCLKX2N_P]
set_multicycle_path 1 -hold -from [get_clocks HCLK_V] -to [get_clocks DMC_MCLKX2N_P]
```

1.3 Input constraint on remap port is also very high. This should be relaxed from 60% to 40%.

```
set_input_delay $HCYCLE60pc -max -clock HCLK_V [get_ports remap]
```

to

```
set_input_delay $HCYCLE40pc -max -clock HCLK_V [get_ports remap]
```

1.4 -add_delay option should be added to constraints where ports have been constrained to multiple clocks.

2. The following changes need to be made in pl244_compile.tcl script:

2.1 DMC_DQS_V and DMC_DQSN_V should be defined as below with correct clock periods:

```
create_clock -name DMC_DQS_V -period $rm_dmc_mclock_period
set_clock_uncertainty $rm_clock_uncertainty [ get_clocks {DMC_DQS_V} ]
```

```
set_clock_latency $rm_clock_latency [get_clocks {DMC_DQS_V} ]
```

```
create_clock -name DMC_DQSN_V -period $rm_dmc_mclock_period \  
-w [list $rm_dmc_mclock_period_h $rm_dmc_mclock_period] set_clock_uncertainty  
$rm_clock_uncertainty [ get_clocks {DMC_DQSN_V} ]  
set_clock_latency $rm_clock_latency [get_clocks {DMC_DQSN_V} ]
```

2.2 DMC_MCLKFB_P and DMC_MCLKFB_V clocks should have fb_dmc_clk_delay set as its clock latency rather than rm_clock_latency.

```
set_clock_latency $rm_clock_latency [get_clocks {DMC_MCLKFB_P} ]
```

```
.....
```

```
set_clock_latency $rm_clock_latency [get_clocks {DMC_MCLKFB_V} ]
```

to

```
set_clock_latency $fb_dmc_clk_delay [get_clocks {DMC_MCLKFB_P} ]
```

```
.....
```

```
set_clock_latency $fb_dmc_clk_delay [get_clocks {DMC_MCLKFB_V} ]
```

Where fb_dmc_clk_delay is

```
set fb_dmc_clk_delay [ expr (0.5 * $rm_dmc_mclock_period) + $rm_clock_latency ]
```

Implications

When the product is synthesised as delivered it generates violations.

Workaround

none

404770: NAND boot mode problem for some NAND memory device protocols**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Cat 3, Present in: r0p0-01rel0, Fixed in r0p1-00rel0.

Description

In NAND boot mode, the static memory controller stalls operation during the NAND busy period following the command phase of a read access.

The flag that controls this stall internally is currently set when moving from command state to idle state, for example when finishing the second command of a page read program.

For memories that do not require this second command, this flag is not being set and hence the controller does not stall operation during the NAND busy period.

Implications

When using the NAND boot feature, if the PL350 controller does not issue a second command in a read program, then it will not stall during the memory device's busy period.

Therefore, if the memory device cannot accept a second command and the data access is attempted before the busy period elapses a violation will occur.

Workaround

Some NAND devices automatically power-on in page-read mode.

Memories which cannot support a second command as part of a page read program, but power-on in read mode could be booted from by only issuing read data accesses without the command phase.

The necessary code for booting and loading the driver would need to fit into the first page of memory as other pages could not be accessed without a read command.

437116: Performance impact of breaking up bursts to NAND memory**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

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

Description

This revision of the memory controller contains logic required for operations with NAND memories. Each AHB port has a logic block that changes the incoming burst type if the burst is to NAND memory. The parameter "BROKEN_BURSTS_TO_NAND" is used to determine if a burst must be modified.

BROKEN_BURSTS_TO_NAND is set for each AHB port by a 'define in the top level verilog file. eg. BROKEN_BURSTS_TO_NAND0 sets the parameter for AHB port 0

BROKEN_BURSTS_TO_NAND = 1 (default)

This setting is used if the AHB arbitration can break bursts to NAND memory.

In this case, all bursts to NAND must be converted to SINGLE transfers to ensure the address pointers inside the NAND memory do not become corrupted. This will have a significant performance impact on bursts to NAND memory.

BROKEN_BURSTS_TO_NAND = 0

This setting can only be used if the system design ensures that the AHB arbitration cannot break bursts to NAND memory.

In this case, only undefined length INCR bursts to NAND must be converted to SINGLE transfers.

All fixed length bursts to NAND can be left unchanged.

This will ensure the address pointers inside the NAND memory do not become corrupted.

This will be a performance impact if undefined length INCR bursts are used to access NAND. In this case the performance of fixed length bursts is not affected.

Implications

If the AHB arbitration cannot ensure that bursts to NAND are not broken, then all bursts will be performed as SINGLE transfers. This will have a significant performance impact for access to NAND memories.

If AHB arbitration can ensure bursts to NAND are not broken, undefined length INCR bursts must be converted to SINGLE transfers. In this case there will be a significant performance penalty using undefined length INCR bursts to NAND.

Workaround

Design the AHB arbitration such that bursts to NAND memory cannot be broken.

Always use defined length bursts to access NAND memory.

Errata - Documentation

408924: Burst align bit behaviour different to that documented

Status

Affects: product .

Fault status: Doc, Present in: r0p0-01rel0, 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

437114: The behaviour of bursts to NAND devices is not described in the TRM**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

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

Description

This revision of the memory controller contains logic required for operations with NAND memories. Each AHB port has a logic block that changes the incoming burst type if the burst is to NAND memory. The logic is contained in the file pl240_BurstToSingle.v

The logic determines whether the current AHB access is to NAND memory by decoding the address. If the transfer is to NAND then the burst type may need to be modified. The parameter "BROKEN_BURSTS_TO_NAND" is used to determine if a burst must be modified.

BROKEN_BURSTS_TO_NAND is set for each AHB port by a 'define in the top level verilog file. eg. BROKEN_BURSTS_TO_NAND0 sets the parameter for AHB port 0

BROKEN_BURSTS_TO_NAND = 1 (default)

This setting is used if the AHB arbitration can break bursts to NAND memory.

In this case, all bursts to NAND must be converted to SINGLE transfers to ensure the address pointers inside the NAND memory do not become corrupted.

BROKEN_BURSTS_TO_NAND = 0

This setting can only be used if the system design ensures that the AHB arbitration cannot break bursts to NAND memory.

In this case, only undefined length INCR bursts to NAND must be converted to SINGLE transfers.

All fixed length bursts to NAND can be left unchanged.

This will ensure the address pointers inside the NAND memory do not become corrupted.

Implications

The TRM will not describe the behaviour of the device when bursts occur to NAND memory

Workaround

None

442367: Use of NAND Memory in multi-master systems.**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Doc, Present in: r0p0-01rel0, r0p1-00rel0, Open.

Description

The PL24x IG does not make it clear that it is the responsibility of software to ensure that only one master can access NAND memories at any one time.

Implications

If software allows two masters to try and access nand devices at the same time the controller will pass all commands through to the memory. However, the memory may not return the data expected by one master if the other has interleaved commands.

Workaround

Software should use a semaphore (or similar) based system to ensure only one master can access the NAND memory at a time.

442375: Clarification of restrictions when building NAND command phase transactions.**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

Fault status: Doc, Present in: r0p0-01rel0,r0p1-00rel0, Open.

Description

The description in the PL24x TRM (Section 2.6.5) concerning the restrictions placed on AHB commands used to build NAND command phase accesses is not detailed enough.

Implications

Failure to follow the documented restrictions can result in undefined operation of the NAND controller.

Workaround

Clarification of the TRM section 2.6.5 on NAND command phase transfers.

The address data sent to the NAND memory is passed to the controller as AHB write data and the number of address beats to be used is specified on the AHB address bus (HADDR).

To ease system integration the NAND command phase can be built up using up to 4 transfers. However, these transfer must obey the following restrictions:

- 1) The size of the transfers must all be the same.
- 2) The bottom three bits of the address must correctly indicate which bytes are being written
e.g. A 4 address cycle command phase sent with two half word AHB transactions should
be sent with address offsets 0x0 and 0x2.
- 3) There must be no unused data beats.

To guarantee that point 1) is honoured it is recommended that address is packed out to the nearest word boundary. So Command phase accesses with 0-3 address bytes should be treated as a register and sent as a word access.

Command phase access with 4-7 address bytes should be packed out to 8 bytes (2 words) and send as two word accesses. When two word accesses are used then the address offsets for the two transactions should be 0x0 and 0x4.

442624: Restrictions on the use of ClearCS and End Commands not documented in TRM**Status**

Affects: product AHB DDR & NAND Controller -Perpetual.

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

Description

The new pl240_BurstToSingle block splits all undefined length INCR transactions and possibly all bursts to singles (depending on the value of the BROKEN_BURSTS_TO_NAND parameter).

The NAND controller therefore receives one access for each beat of the undefined length INCR.

If ClearCS or end_command_valid is asserted for that burst the controller will try to clear the CS (or insert an end command) after each beat.

Implications

This can lead to incorrect behaviour on the memory interface

Workaround

Software should ensure that the last data phase access (which has end_command_valid) asserted is a single (or undefined length INCR of length 1) to ensure the command is not duplicated.

If the BROKEN_BURST_TO_NAND parameter is set to not break up bursts then the end command can also be sent with a fixed length burst.

The same restriction applies to the ClearCS bit for memories that require CS to remain asserted (i.e. when nand_csl is tied low).

Errata – Driver Software

There are no Errata in this Category