ARM PrimeCell™ MultiPort Memory Controller (PL176)

Revision: r0p1

Technical Reference Manual

ARM DDI 0269A

ARM PrimeCell MultiPort Memory Controller (PL176)
Technical Reference Manual

Release Information

The following changes have been made to this document.

Change history

Date	Issue	Change
June 2003	A	First release

Proprietary Notice

Confidentiality Status

Product Status

The information in this document is final, that is for a developed product.

Web Address

`http://www.arm.com`

Contents

ARM PrimeCell MultiPort Memory Controller (PL176) Technical Reference Manual

List of Tables
ARM PrimeCell MultiPort Memory Controller (PL176) Technical Reference Manual

List of Figures
ARM PrimeCell MultiPort Memory Controller (PL176) Technical Reference Manual

Preface

This preface introduces the ARM PrimeCell *MultiPort Memory Controller* (MPMC) (PL176) *Technical Reference Manual* (TRM). It contains the following sections:

- *About this document* on page xx
- *Feedback* on page xxvi.

About this document

This document is the technical reference manual for the ARM PrimeCell MPMC (PL176).

Intended audience

This document has been written for hardware and software engineers implementing *System-on-Chip* (SoC) designs. It provides information to enable designers to integrate the peripheral into a target system as quickly as possible.

Using this manual

This document is organized into the following chapters:

Chapter 1 *Introduction*

Read this chapter for an introduction to the ARM PrimeCell MPMC (PL176).

Chapter 2 *Functional Overview*

Read this chapter for a description of the major functional blocks of the ARM PrimeCell MPMC (PL176).

Chapter 3 *Programmer's Model*

Read this chapter for a description of the ARM PrimeCell MPMC (PL176) registers and programming details.

Chapter 4 *Programmer's Model for Test*

Read this chapter for a description of the logic in the ARM PrimeCell MPMC (PL176) for functional verification and production testing.

Chapter 5 *Static Memory Controller*

Read this chapter for details of the Static Memory Controller in the ARM PrimeCell MPMC (PL176).

Chapter 6 *NAND Flash Memory Controller*

Read this chapter for details of the NAND Flash Memory Controller in the ARM PrimeCell MPMC (PL176).

Chapter 7 *Dynamic Memory Controller*

Read this chapter for details of the Dynamic Memory Controller in the ARM PrimeCell MPMC (PL176).

Chapter 8 ***Test Interface Controller***

Read this chapter for details of the Test Interface Controller in the ARM PrimeCell MPMC (PL176).

Chapter 9 ***System Connectivity***

Read this chapter for details of the ARM PrimeCell MPMC (PL176) system connectivity.

Chapter 10 ***Off-chip Connectivity***

Read this chapter for details of the ARM PrimeCell MPMC (PL176) off-chip connectivity.

Chapter 11 ***Power Strategy*** Read this chapter for details of the power strategy aspects relating to the ARM PrimeCell MPMC (PL176).

Chapter 12 ***Delay Locked Loop***

Read this chapter for details of the ARM PrimeCell MPMC (PL176) timing.

Appendix A ***Pad Interface Timing***

Read this appendix for details of the ARM PrimeCell MPMC (PL176) pad interface timing.

Appendix B ***Troubleshooting***

Read this appendix for details of troubleshooting the ARM PrimeCell MPMC (PL176).

Appendix C ***Signal Descriptions***

Read this appendix for details of the ARM PrimeCell MPMC (PL176) signals.

Product revision status

The r*n*p*n* identifier indicates the revision status of the product described in this document, where:

r*n*	Identifies the major revision of the product.
p*n*	Identifies the minor revision or modification status of the product.

Typographical conventions

The following typographical conventions are used in this book:

italic	Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes ARM processor signal names. Also used for terms in descriptive lists, where appropriate.
`monospace`	Denotes text that can be entered at the keyboard, such as commands, file and program names, and source code.
`monospace` (with "mono" underlined)	Denotes a permitted abbreviation for a command or option. The underlined text can be entered instead of the full command or option name.
`monospace italic`	Denotes arguments to commands and functions where the argument is to be replaced by a specific value.
`monospace bold`	Denotes language keywords when used outside example code.

Note

For Table 7-15 on page 7-32 to Table 7-91 on page 7-108, the typographical conventions differ as follows:

- values in non-highlighted text indicate row accesses
- values in bold indicate bank accesses
- values in italic indicate column accesses
- values in bold italic indicate segment accesses.

Other conventions

This document uses other conventions. They are described in the following sections:

- *Signals* on page xxiii
- *Bytes, Halfwords, and Words* on page xxiii
- *Bits, bytes, k, and M* on page xxiii
- *Register fields* on page xxiii
- *Timing diagram conventions* on page xxiv.

Signals

When a signal is described as being asserted, the level depends on whether the signal is active HIGH or active LOW. Asserted means HIGH for active high signals and LOW for active low signals:

Prefix n	Active LOW signals are prefixed by a lowercase n except in the case of AHB reset signals. These are named **HRESETn**.
Prefix H	AHB signals are prefixed by an upper case H.

Bytes, Halfwords, and Words

Byte	Eight bits.
Halfword	Two bytes (16 bits).
Word	Four bytes (32 bits).
Quadword	16 contiguous bytes (128 bits).

Bits, bytes, k, and M

Suffix b	Indicates bits.
Suffix B	Indicates bytes.
Suffix K	When used to indicate an amount of memory means 1024.When used to indicate a frequency means 1000.
Suffix M	When used to indicate an amount of memory means 1024^2 = 1 048 576. When used to indicate a frequency means 1 000 000.

Register fields

All reserved or unused address locations must not be accessed as this can result in unpredictable behavior of the device.

All reserved or unused bits of registers must be written as zero, and ignored on read unless otherwise stated in the relevant text.

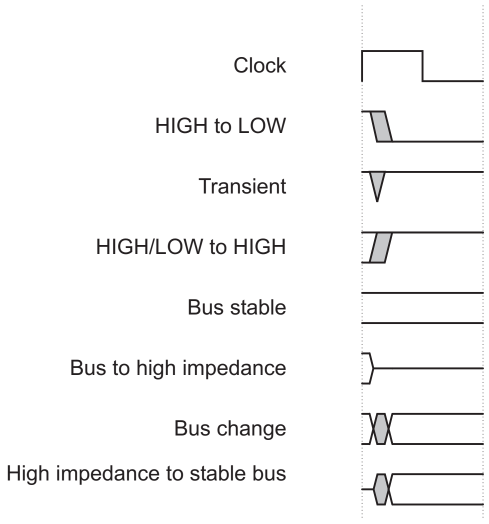
All registers bits are reset to logic 0 by a system reset unless otherwise stated in the relevant text.

Unless otherwise stated in the relevant text, all registers support read and write accesses. A write updates the contents of the register and a read returns the contents of the register.

All registers defined in this document can only be accessed using word reads and word writes, unless otherwise stated in the relevant text.

Timing diagram conventions

This manual contains one or more timing diagrams. The following figure explains the components used in these diagrams. Any variations are clearly labeled when they occur. Therefore, no additional meaning must be attached unless specifically stated.



Key to timing diagram conventions

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

Further reading

This section lists publications by ARM Limited.

ARM periodically provides updates and corrections to its documentation. See `http://www.arm.com` for current errata sheets, addenda, and the ARM Frequently Asked Questions list.

ARM publications

This document contains information that is specific to the ARM PrimeCell MPMC (PL176). Refer to the following documents for other relevant information:

- *AMBA Specification (Rev 2.0)* (ARM IHI 0011)
- ARM11 AMBA Extensions Specification (PR022-GENC-001011)
- *AMBA Design Kit Technical Reference Manual* (ARM DDI 0243)
- *ARM PrimeCell External Bus Interface Technical Reference Manual* (PL220) (ARM DDI 0249).

Feedback

ARM Limited welcomes feedback both on the ARM PrimeCell MPMC (PL176), and on the documentation.

Feedback on the ARM PrimeCell MPMC (PL176)

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, send an email to `errata@arm.com` giving:

- the document title
- the document number
- the page number(s) to which your comments refer
- a concise explanation of your comments.

General suggestions for additions and improvements are also welcome.

Chapter 1
Introduction

This chapter introduces the ARM PrimeCell MPMC (PL176). It contains the following sections:

1.1 About the ARM PrimeCell MPMC (PL176)

The PrimeCell MPMC is an *Advanced Microcontroller Bus Architecture* (AMBA) compliant *System-on-Chip* (SoC) peripheral that is developed, tested, and licensed by ARM Limited. It connects to the *Advanced High-performance Bus* (AHB).

1.1.1 Features of the PrimeCell MPMC

The PrimeCell MPMC offers:

- AMBA 64-bit AHB support, including ARM11 AMBA extensions.
- Ten AHB interfaces for accessing external memory, four 64-bit interfaces and six 32-bit interfaces. Expandable to 16 with any combination of 32-bit and 64-bit interfaces.
- A separate AHB interface for programming the MPMC registers. Enables the MPMC registers to be situated in memory with other system peripheral registers.
- Locked AHB transactions supported.
- Support for all AHB burst types.
- Four chip selects for dynamic memory and four chip selects for static memory devices.
- Dynamic memory interface supports DDR-SDRAM, SDRAM, and low-power variants. It also supports Micron SyncFlash types of memory.
- Asynchronous static memory device support including RAM, ROM, Flash, and NAND Flash, with or without asynchronous page mode.
- 16-bit, 32-bit, and 64-bit wide data bus SDRAM and SyncFlash memory support. 16-bit and 32-bit wide data bus DDR-SDRAM support.
- 8-bit, 16-bit, and 32-bit wide static memory support.
- Static memory features include:
  - asynchronous page mode read
  - programmable wait states
  - bus turnaround cycles
  - output enable, and write enable delays
  - extended wait.
- Read and write buffers to reduce latency and to improve performance.

- Controller supports 2K, 4K, and 8K row address synchronous memory parts. That is, typical 512Mb, 256Mb, 128Mb, and 16Mb parts, with 8, 16, 32, or 64 DQ (data) bits per device.
- Dynamic memory self-refresh mode supported by a *Power Management Unit* (PMU) interface or by software.
- Power-saving modes dynamically control **MPMCCKEOUT** and **MPMCCLKOUT**.
- Two reset domains enable dynamic memory contents to be preserved over a soft reset.
- Little, big, and mixed-endian support.
- Specifically designed for cached processors.
- Designed to work with noncritical word first, and critical word first processors, such as the ARM926EJ-S.
- Support for the *External Bus Interface* (EBI) that enables the MPMC pads to be shared.
- Integrated *Test Interface Controller* (TIC).
- PrimeCell ID support.

Note

Synchronous static memory devices (burst mode devices) are not supported.

1.2 Supported dynamic memory devices

This section provides examples of dynamic memory devices that are supported by the PrimeCell MPMC:

- *DDR-SDRAM devices*
- *SDRAM devices*
- *Micron style synchronous flash devices* on page 1-5
- *Micron style V-synchronous flash devices* on page 1-5
- *JEDEC low-power SDRAM devices* on page 1-5.

——— **Note** ———

This is not an exhaustive list of supported devices.

———————————

1.2.1 DDR-SDRAM devices

The following DDR-SDRAM devices are supported:

- 64Mb devices:
  - Micron MT46V2M32.
- 128Mb devices:
  - Micron MT46V16M8
  - Micron MT46V8M16.
- 256Mb devices:
  - Micron MT46V32M8.
  - Micron MT46V16M16.

1.2.2 SDRAM devices

The following SDRAM devices are supported:

- 16Mb devices:
  - Micron MT48LC1M16A1S
  - Samsung K4S160822D-G/F
  - Samsung K4S641632C.
- 64Mb devices:
  - Micron MT48LC2M32B2-6
  - Micron MT28S4M162C-10

  - Micron MT48LC4M16A2
  - Elpida μPD4564323-10
  - Hitachi HM5264805F-75.
- 128Mb devices:
  - Micron MT48LC4M32B2
  - Micron MT48LC16M8A2
  - Micron MT48LC8M16A2.
- 256Mb devices:
  - Elpida μPD45256163-10
  - Micron MT48LC16M16A2-8E
  - Hitachi HM522532F-B6
  - Hitachi HM5225805B-75.
- 512Mb devices:
  - Elpida HM5257805B-A6.

1.2.3 Micron style synchronous flash devices

The following 64Mb devices are supported:

- Micron MT28S4M16LC-10
- Micron MT28S4M16LC-12.

1.2.4 Micron style V-synchronous flash devices

The following 128Mb devices are supported:

- Micron MT28V4M32LL.

1.2.5 JEDEC low-power SDRAM devices

The following JEDEC low-power SDRAM devices are supported:

- 64Mb Micron MT48LC2M32LFFC-8
- 128Mb Infineon HYB25L128160AC.

1.3 Supported static memory devices

This section provides examples of static memory devices that are supported by the PrimeCell MPMC:

- *Examples of ROM devices*
- *Examples of page mode ROM devices*
- *Examples of SRAM devices*
- *Examples of flash devices*
- *Examples of page mode flash devices*
- *Examples of NAND flash memory devices* on page 1-7.

——— **Note** ———

This is not an exhaustive list of supported devices.

———————————

1.3.1 Examples of ROM devices

The PrimeCell MPMC supports the 128Mb Samsung K3N9V(U)1000M-YC.

1.3.2 Examples of page mode ROM devices

The PrimeCell MPMC supports the 128Mb Samsung K3P9V(U)1000M-YC.

1.3.3 Examples of SRAM devices

The PrimeCell MPMC supports the following devices:

- 256Kb IDT IDT71V256SA20Y
- 256Kb Micron MT5C2568-12
- 4Mb Samsung K6F8016R6M
- 4Mb Samsung K6R4016CK-12
- 8Mb Samsung K6T8016C3M-70
- 8Mb Samsung K6F8008R2M.

1.3.4 Examples of flash devices

The PrimeCell MPMC supports the 4Mb Micron MT28F004B5.

1.3.5 Examples of page mode flash devices

The PrimeCell MPMC supports the 8Mb Intel 28F800F3 and the 4Mb Intel E28F320J3A110.

1.3.6 Examples of NAND flash memory devices

The PrimeCell MPMC supports:

- 4Mb Samsung K9F4008W0A
- 64Mb AMD Am30LV0064D
- 256Mb Samsung K9F5608U08-Y
- 1Gb Toshiba TH58100FT
- 2Gb Samsung K9K2G08Q0M.

Chapter 2
Functional Overview

This chapter describes the major functional blocks of the ARM PrimeCell MPMC (PL176). It contains the following sections:

2.1 PrimeCell MPMC functional description

Figure 2-1 shows a simplified top-level block diagram of the PrimeCell MPMC.

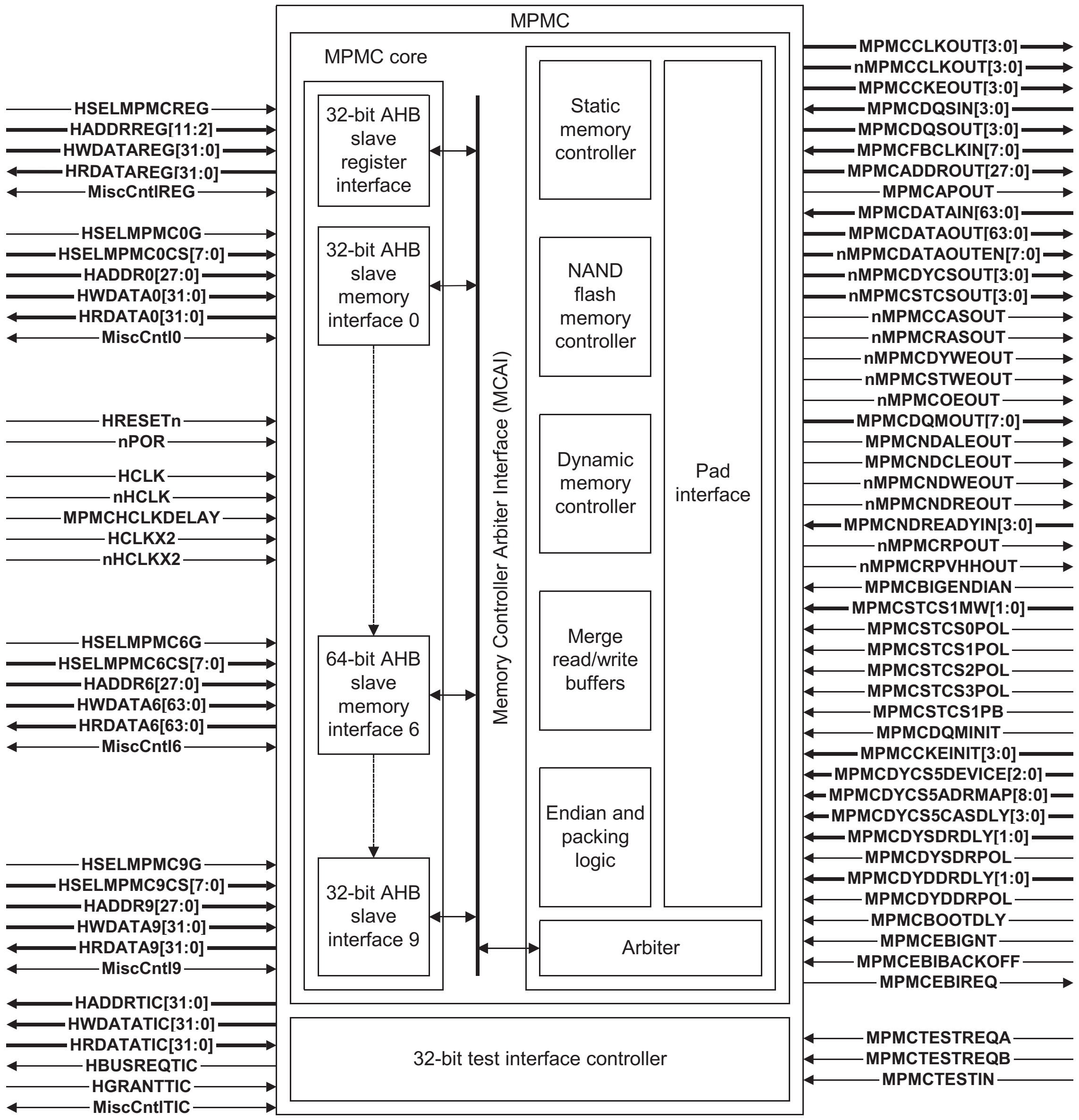


Figure 2-1 PrimeCell MPMC block diagram

The AHB memory port configuration is shown in Table 2-1.

Table 2-1 AHB memory port configuration

Port	Type	Width	Priority
0	AMBA AHB v2.0	32-bit	0 (highest)
1	AMBA AHB v2.0	32-bit	1
2	AMBA AHB v2.0	32-bit	2
3	AMBA AHB v2.0 with ARM11 extensions	64-bit	3
4	AMBA AHB v2.0 with ARM11 extensions	64-bit	4
5	AMBA AHB v2.0 with ARM11 extensions	64-bit	5
6	AMBA AHB v2.0 with ARM11 extensions	64-bit	6
7	AMBA AHB v2.0	32-bit	7
8	AMBA AHB v2.0	32-bit	8
9	AMBA AHB v2.0	32-bit	9 (lowest)

The functions of the PrimeCell MPMC blocks are described in the following sections:

- *Multiport memory controller block*
- *AHB slave register interface* on page 2-5
- *AHB slave memory interfaces* on page 2-6
- *Data buffers* on page 2-9
- *Endian and packing logic* on page 2-21
- *Arbiter* on page 2-21
- *MPMC state machine* on page 2-21
- *External DLL block* on page 2-22

2.1.1 Multiport memory controller block

The multiport memory controller block optimizes and controls external memory transactions. The block contains the following:

- *Command sequencer* on page 2-4
- *Memory transfer state machine* on page 2-4
- *Static memory controller register bank* on page 2-4
- *Dynamic memory controller register bank* on page 2-4.

Command sequencer

The command sequencer holds up to 10 requests in its internal buffer. It prioritizes and rearranges accesses to maximize memory bandwidth and minimize transaction latency.

For example, if AHB interfaces 3 and 2 simultaneously request a data transfer from dynamic memory, to different memory banks, and the port 3 request address is to a closed page, but port 2 address is for an already open page, the following sequence occurs:

1. The ACT command is sent to open the SDRAM row specified by the AHB interface 3 address.
2. The AHB interface 2 access is completed.
3. AHB interface 3 access is completed.

The access priority is modified to take into account the ease of getting data to complete each transfer, but the access priority is always biased to the highest priority AHB interface.

Memory transfer state machine

The memory transfer state machine controls memory transactions.

Static memory controller register bank

There are four static memory controller register banks. Each contains the registers for a static memory bank.

Dynamic memory controller register bank

There are four dynamic memory controller register banks. Each contains the registers for a dynamic memory bank.

Note

The dynamic memory controller supports only SINGLE access bursts, that is:

- single transfer bursts for 64-bit wide external memory
- bursts of two transfers for 32-bit wide external memory
- bursts of four transfers for 16-bit wide external memory.

2.1.2 AHB slave register interface

The AHB slave register interface block enables the registers of the PrimeCell MPMC to be programmed. This module also contains most of the registers and performs most of the register address decoding. The register bank uses a 32-bit wide AHB interface. To connect a 64-bit wide AHB master to the register interface of the MPMC, an external downsizer, similar to that shown in Figure 2-2, is required. A suitable downsizer is available as part of the *AMBA Design Kit* (ADK).

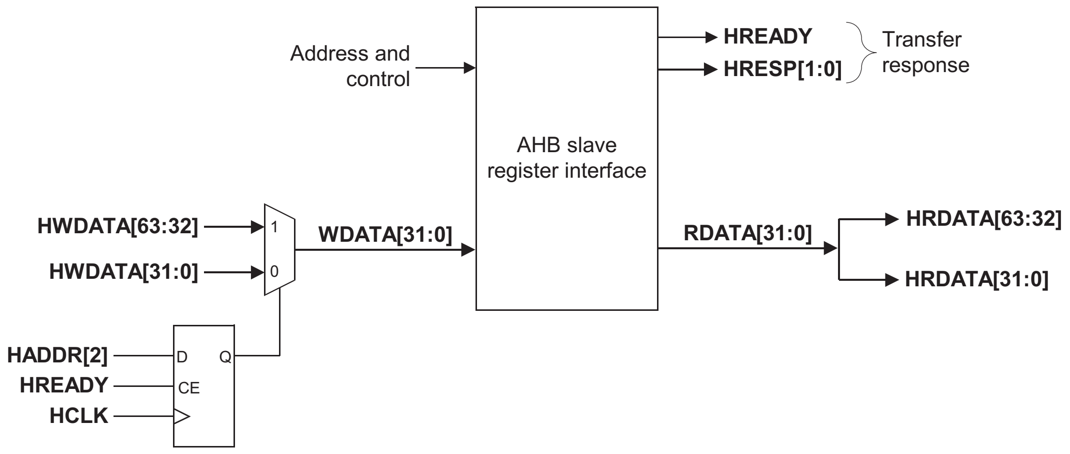


Figure 2-2 64-bit master to 32-bit slave register interface downsizer

The AHB slave register interface must be connected to the ARM processor AHB bus to enable the MPMC to be programmed.

AHB slave register transaction endianness and transfer width

To eliminate the possibility of endianness problems, all data transfers to and from the registers of the PrimeCell MPMC must be 32 bits wide.

Note

If an access is attempted with a size other than a word, 32 bits, it causes an ERROR response on **HRESP[0]** and the transfer is terminated.

2.1.3 AHB slave memory interfaces

The AHB slave memory interfaces enable devices to access the external memories. The memory interfaces are prioritized, with interface 0 having the highest priority. Having more than one memory interface enables high-bandwidth peripherals direct access to the PrimeCell MPMC, without data having to pass over the main system bus. This reduces memory access latency.

Note

- All AHB burst types are supported, enabling the most efficient use of memory bandwidth.
- The AHB interfaces do not generate SPLIT and RETRY responses.

AHB masters cannot be made to function on a narrower bus than originally intended, unless there is some mechanism included within the master to limit the width of transfers that the bus master attempts. The MPMC is designed to provide a selected mix of 32-bit and 64-bit AHB interfaces on a port-by-port basis. The required port mix can be configured as required for the application, either by adding or removing 32-bit or 64-bit ports, or by changing the order and priority of the port instantiations. If it is necessary to connect a 32-bit AHB master to one of the 64-bit AHB memory slave interfaces, an external conditioning block, similar to that shown in Figure 2-3, is required.

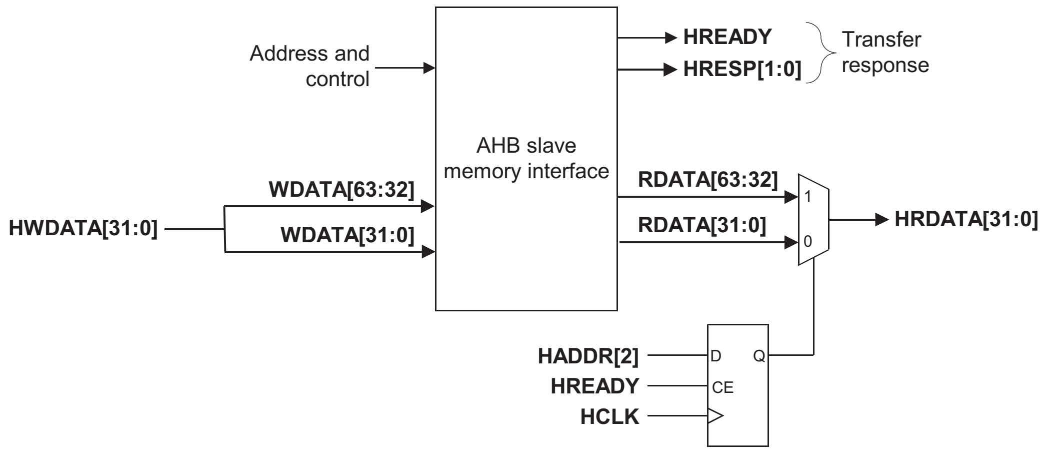


Figure 2-3 32-bit master to 64-bit slave memory interface

Memory transaction endianness

The following rules are applicable when handling memory transactions in the MPMC:

32-bit AHB ports

If the global **MPMCBIGENDIAN** signal is LOW, all 32-bit AHB ports are fixed as little-endian.

If the global **MPMCBIGENDIAN** signal is HIGH, all 32-bit AHB ports are fixed as big-endian.

64-bit AHB ports

If the global **MPMCBIGENDIAN** signal is LOW and the **MPMCBSTRBENx** signal of a port is LOW, that port is fixed as little-endian.

If the global **MPMCBIGENDIAN** signal is LOW and the **MPMCBSTRBENx** signal of a port is HIGH, that port is fixed as mixed-endian.

If the global **MPMCBIGENDIAN** signal is HIGH, all ports are fixed as big-endian irrespective of the state of their **MPMCBSTRBENx** signal.

External memory banks

If the global **MPMCBIGENDIAN** signal is LOW, all external memory banks are fixed as little-endian.

If the global **MPMCBIGENDIAN** signal is HIGH, all external memory banks are fixed as big-endian.

Table 2-1 on page 2-3 shows a truth table for the MPMC endianness conditions.

Table 2-2 MPMC endianness options

MPMCBIGENDIAN	MPMCBSTRBENx	64-bit AHB port	32-bit AHB port	External memory bank
0	0	Little-endian	Little-endian	Little-endian
0	1	Mixed-endian	Little-endian	Little-endian
1	X (don’t care)	ARM big-endian	ARM big-endian	ARM big-endian

Note

MPMCBSTRBENx is set HIGH to enable byte strobes on a port, and to indicate that a port is compliant with ARMv6 masters. The master provides byte strobes on the **HBSTRB[7:0]** lines. When set HIGH, the byte strobe signals are always used by the MPMC and therefore must be driven to the correct values on all transfers, regardless of the **HUNALIGN** value. The ARM11 drives the byte strobes in this way.

MPMCBSTRBENx is set LOW to indicate that a port is compliant with ARMv5 and earlier masters only. The endianness of the data transfers to and from the external memories is then ARMv5 little-endian or ARMv5 big-endian, determined by the state of the global endian mode signal, **MPMCBIGENDIAN**.

The power-on reset value of the global **MPMCBIGENDIAN** signal can be overridden through the register interface by accessing the MPMCConfig Register.

The setting of ARM11 mixed-endian mode for a port cannot be overridden through the register interface. It is fixed at power-on reset.

All data in the MPMC must be flushed when switching between little-endian and big-endian modes. This is so that data residing in the merge buffers is transferred correctly.

Memory transaction size

Memory transactions can be 8, 16, 32, or 64 bits wide. Any access attempted with a size greater than the width of the AHB memory ports causes an ERROR response on **HRESP[0]** and the transfer is terminated.

Unpopulated memory areas

Accesses to address space within a memory bank which is not populated generates an ERROR response on **HRESP[0]** and the transfer is terminated. The system decoder defines the populated address space within a memory bank using a full decode of the corresponding **HSELMPMCxCSy** select signal.

The PrimeCell MPMC is not able to detect unpopulated memory areas, so the system decoder must fully define the memory map and ensure that the **HSELMPMCxCSy** signals are only driven when populated memory locations are addressed.

Write protected memory areas

Write transactions to write-protected memory areas generate an ERROR response on **HRESP[0]** and the transfer is terminated.

ARM11 AMBA extensions

64-bit AHB ports provide support for the ARM11 AMBA extensions. This includes the following:

- exclusive accesses using extended **HPROT** and **HRESP** lines
- mixed-endian transfers using the **HBSTRB** byte strobe signals
- unaligned transfers using the byte strobe and unalign signals, **HBSTRB** and **HUNALIGN**.

For further information, see the *ARM11 AMBA Extensions Specification*.

2.1.4 Data buffers

Each AHB memory interface uses a single Dword write buffer and read cache to merge data to and from external memory to improve memory bandwidth. When reading word (32-bit), halfword, or byte wide data, external memory accesses are reduced to a single 64-bit wide external memory access. If consecutive address hits occur to data cached in the 64-bit wide read buffer, no external memory accesses are necessary. Similarly, when writing word, halfword, or byte data, external memory accesses are reduced to a single 64-bit wide access if consecutive address hits occur to the 64-bit wide write buffer. The merge write or read buffers for each port can be disabled using the register interface by writing to the MPMCAHBControlx Register for the AHB port.

Buffered write and read transactions on a single AHB port to the same memory location are always coherent.

Note

For 32 and 64-bit masters which use separate read and write ports, the associated AHB port buffers must be disabled to ensure that data coherency is maintained.

The internal merge buffers of both 32 and 64-bit ports are fixed at 64 bit, and any external read accesses always perform enough transfers to fill the 64-bit buffer, assuming they are enabled and **HPROT** indicates it can be cached or buffered.

The AHB port data buffers are only used for dynamic memory transfers. The static memory controller uses a 32-bit buffer to optimize reads and writes.

The buffers are automatically disabled during ARM11 Exclusive Access transactions.

Read transaction buffer operation

If an 8-bit, 16-bit, or 32-bit wide AHB read transfer is performed with the buffers enabled, the read transaction submitted to the memory is at the maximum width of the memory chip-select, with enough transfers to fill the 64-bit buffer. Therefore a chip-select with a 64-bit data bus returns 64 bits of data. This data is then placed into the buffer. Data is provided from the buffer to the AHB bus as necessary. This reduces the number of read transactions to external memory for 8-bit, 16-bit, or 32-bit wide AHB transactions. The MPMC can then re-arbitrate to a different AHB port and perform memory transactions while the data is being transferred from the data buffer.

Note

- Enabling the buffers has no impact on 64-bit wide read transactions.
- The MPMC re-arbitrates to an open page transfer during a buffered transaction.

Enabling read buffer

For read transactions the respective AHB port buffer is only used if:

- The AHB respective AHB port buffer is enabled.
- One of the 8-bit, 16-bit, or 32-bit wide AHB read transactions is performed. 64-bit wide transactions do not use the buffer, because this does not improve performance.
- A multi-word AHB burst is performed. AHB burst SINGLE transfers do not use the buffer.
- AHB HPROT protection information indicates that the transfer is cacheable, indicating to the MPMC that the particular read transaction can be buffered.
- The transaction is not a locked transfer (**HMASTLOCK** is LOW). This ensures that atomic AHB transactions are performed straight to memory and are not re-arbitrated during the transaction.
- The transaction is not an ARM11 Exclusive Access read.

Note

If an AHB master does not provide AHB HPROT protection information, the relevant HPROT bits can be tied off as required.

Read data re-use

Only the AHB burst that fetched the data from the memory into the buffer can use the data in the buffer. Subsequent AHB bursts do not make use of the read data in the buffer even if the transaction is to the same memory area.

Advantages of read buffering

Enabling read buffering:

- reduces power consumption because fewer commands are issued to memory.
- increases memory bandwidth for 8-bit, 16-bit, and 32-bit wide memory transactions, because the MPMC can re-arbitrate to service a different AHB port while the data is being read from the buffer.

Note

Only AHB ports that have a memory request to open pages of SDRAM memory are serviced.

Read buffer example

The example described is for the case where the AHB port buffers are enabled and cacheable transfers are performed, with AHB port 0 and AHB port 1 performing INCR4, 16-bit wide read transactions. Accesses are to a 64-bit wide dynamic memory chip-select.

Read buffer enabled

Table 2-3 shows that with the read buffer enabled the memory is operating at maximum efficiency and providing 64 bits of data on each clock cycle.

Table 2-3 Read buffer enabled

Clock cycle	AHB 0	AHB 1
1	64-bit read from memory and placed in buffer. Data 0 returned on AHB.	AHB port waiting for data.
2	Data 1 returned on AHB.	64-bit read from memory and placed in buffer. Data 0 returned on AHB

Table 2-3 Read buffer enabled (continued)

Clock cycle	AHB 0	AHB 1
3	64-bit read from memory and placed in buffer. Data 2 returned on AHB.	Data 1 returned on AHB.
4	Data 3 returned on AHB.	64-bit read from memory and placed in buffer. Data 2 returned on AHB.
5	AHB port available.	Data 3 returned on AHB.

Read buffer disabled

Table 2-4 shows that with the read buffer disabled the memory provides 32 bits of data on each clock cycle. The MPMC therefore only provides half the amount of bandwidth compared to the case with the buffers enabled.

Table 2-4 Read buffer disabled

Clock cycle	AHB 0	AHB 1
1	32-bit read from memory. Data 0 returned on AHB.	AHB port waiting for data.
2	32-bit read from memory. Data 1 returned on AHB.	AHB port waiting for data.
3	32-bit read from memory. Data 2 returned on AHB.	AHB port waiting for data.
4	32-bit read from memory. Data 3 returned on AHB.	AHB port waiting for data.
5	AHB port available.	32-bit read from memory. Data 0 returned on AHB.
6	AHB port available.	32-bit read from memory. Data 1 returned on AHB.
7	AHB port available.	32-bit read from memory. Data 2 returned on AHB.
8	AHB port available.	32-bit read from memory. Data 3 returned on AHB.

Disadvantages of read buffering

The disadvantage with enabling read buffering is that an 8-bit, 16-bit, or 32-bit wide read transfer can take longer to complete, because the AHB port is re-arbitrated to maximize bandwidth while the data is being read from the buffer. If buffering is not required, read buffering can be disabled by either:

- setting the Buffer enable (E) field of the MPMCAHBControl Register inactive
- setting the AHB HPROT protection information to indicate a non-cacheable access.

Worst case additional latency

The worst case additional latency is for the case where one AHB port, in this example AHB port 0, performs INCR16 32-bit wide read transactions. Other AHB ports, in this case AHB port 1 and 2, perform continuous, in page, INCR16 64-bit read transactions. Accesses are to a 32-bit wide dynamic memory chip-select.

Read buffer enabled

With the read buffer enabled the memory provides 64 bits of data on every clock cycle. The MPMC re-arbitrates to a different AHB port when data is being read from the buffer. The latency of a buffer transfer might therefore be longer than a nonbuffered transfer.

The worst case additional latency for a buffered transfer is when a buffered INCR16 32-bit wide burst is being performed and the other AHB ports are performing INCR16 64-bit wide transfers. Table 2-5 shows that the INCR16 32-bit wide read can take up to 128 cycles to complete.

Table 2-5 Read buffer enabled

Clock cycle	AHB 0	AHB 1	AHB 2
1	64-bit read from memory and placed in buffer. 16-bit data 0 returned on AHB.	AHB port waiting for data	AHB port waiting for data
2	32-bit data 1 returned on AHB.	AHB port waiting for data	AHB port waiting for data
3	AHB port waiting for data.	64-bit read zero from memory	AHB port waiting for data

Table 2-5 Read buffer enabled (continued)

Clock cycle	AHB 0	AHB 1	AHB 2
4-17	AHB port waiting for data.	64-bit read 1-14 from memory	AHB port waiting for data
18	AHB port waiting for data.	64-bit read 15 from memory	AHB port waiting for data
19	64-bit read from memory and placed in buffer. 32-bit data 2 returned on AHB.	Next INCR16 transfer	AHB port waiting for data
20	32-bit data 3 returned on AHB.	AHB port waiting for data	AHB port waiting for data
21	AHB port waiting for data.	AHB port waiting for data	64-bit read zero from memory
22-37	AHB port waiting for data.	AHB port waiting for data	64-bit read 1-14 from memory
38	AHB port waiting for data.	AHB port waiting for data	64-bit read 15 from memory
39	AHB port waiting for data.	AHB port waiting for data	Next INCR16 transfer
40	64-bit read from memory and placed in buffer. 32-bit data 4 returned on AHB.	AHB port waiting for data	AHB port waiting for data
41	32-bit data 5 returned on AHB.	AHB port waiting for data	AHB port waiting for data

Note

Table 2-5 on page 2-13 shows the first 41 cycles of the transaction. Cycles 42 to 136 are a continuation of the sequence shown.

Read buffer disabled

Table 2-6 shows that if read buffers are not enabled, an INCR16 32-bit wide read burst takes 16 cycles to complete when the read data starts being returned from the memory.

Table 2-6 Read buffer disabled

Clock cycle	AHB 0	AHB 1	AHB 2
1	32-bit read from memory. Data 0 returned on AHB.	AHB port waiting for data.	AHB port waiting for data
2-15	32-bit read from memory. Data 1-14 returned on AHB.	AHB port waiting for data.	AHB port waiting for data
16	32-bit read from memory. Data 15 returned on AHB.	AHB port waiting for data.	AHB port waiting for data
17	AHB port available.	64-bit read from memory. Data 0 returned on AHB.	AHB port waiting for data
18-31	AHB port available.	64-bit read from memory. Data 1-14 returned on AHB.	AHB port waiting for data
32	AHB port available.	64-bit read from memory. Data 15 returned on AHB.	AHB port waiting for data3

Write transaction buffer operation

If an 8-bit, 16-bit, or 32-bit wide AHB write transfer is performed with the buffers enabled the write data is merged into the buffer, so that a write of the maximum width of the memory chip-select is performed. Therefore, for a chip-select with a 64-bit data bus, 64 bits of data are written at a time. This reduces the number of write transactions to external memory for 8-bit, 16-bit, or 32-bit wide AHB transactions. The MPMC can then re-arbitrate to a different AHB port and perform memory transactions while the data is being written into the data buffer.

——— **Note** ———

- Enabling the buffers has no impact on 64-bit wide write transactions.
- The MPMC re-arbitrates to an open page transfer during a buffered transaction.

———————————

Enabling write buffer

For write transactions the respective AHB port buffer is only used if:

- The respective AHB port buffer is enabled.
- An 8-bit, 16-bit, or 32-bit write transaction is performed. 64-bit wide transactions do not make use of the buffer because this does not improve performance.
- A multi-word AHB burst is performed. AHB burst SINGLE transfers do not use the buffer.
- AHB HPROT protection information indicates that the transfer is bufferable. This indicates to the MPMC that the particular write transaction can be buffered.
- The transaction is not a locked transfer (**HMASTLOCK** is LOW). This ensures that atomic AHB transactions are performed straight to memory and are not re-arbitrated during the transaction.
- The transaction is not an ARM11 Exclusive Access write.

Note

If an AHB master does not provide AHB **HPROT** protection information the relevant **HPROT** bits can be tied off as required.

Write data re-use

If an AHB port performs a burst write into a buffer subsequent AHB burst writes do not merge data into the same buffer, even if the subsequent burst is to the same area of memory. Instead the data in the buffer from the first write is submitted to memory and only then can the second burst start. This ensures that the memory is updated at the end of each burst write so that if another AHB port reads from the same memory location the memory is updated.

Advantages of write buffering

Enabling write buffering:

- reduces power consumption as fewer commands are issued to memory
- increases memory bandwidth for 8-bit, 16-bit, and 32-bit wide memory transactions, because the MPMC can re-arbitrate to service a different AHB port while the data is being written to the buffer.

Note

Only AHB ports that have a memory request to open pages of SDRAM memory are serviced.

Write buffer example

The write buffer example describes the case where the AHB port buffers are enabled and bufferable transfers are performed, with AHB port 0 and AHB port 1 performing INCR4, 32-bit wide write transactions. Accesses are to a 64-bit wide dynamic memory chip-select.

Write buffer enabled

Table 2-7 shows that with the write buffer enabled the memory is operating at maximum efficiency and providing 64 bits of data on each clock cycle.

Table 2-7 Write buffer enabled

Clock cycle	AHB 0	AHB 1
1	32-bit write data 0 written into buffer.	AHB port waiting for data.
2	32-bit write data 1 written into buffer. 64-bit write data written to memory.	32-bit write data 0 written into buffer.
3	32-bit write data 2 written into buffer.	32-bit write data 1 written into buffer. 64-bit write data written to memory.
4	32-bit write data 3 written into buffer.	32-bit write data 2 written into buffer.
5	AHB port available.	32-bit write data 3 written into buffer. 64-bit write data written to memory.

Write buffer disabled

Table 2-8 shows that with the write buffer disabled 32 bits of data is written to memory on each clock cycle. The MPMC therefore only provides half the amount of bandwidth compared to the case with the buffers enabled.

Table 2-8 Write buffer disabled

Clock cycle	AHB 0	AHB 1
1	32-bit write data 0 written into memory	AHB port waiting for data
2	32-bit write data 1 written into memory	AHB port waiting for data
3	32-bit write data 2 written into memory	AHB port waiting for data
4	32-bit write data 3 written into memory	AHB port waiting for data
5	AHB port available	32-bit write data 0 written into memory
6	AHB port available	32-bit write data 1 written into memory
7	AHB port available	32-bit write data 2 written into memory
8	AHB port available	32-bit write data 3 written into memory

Disadvantages of write buffering

The disadvantage with enabling write buffering is that an 8-bit, 16-bit, or 32-bit wide write transfer can take longer to complete, because the AHB port is re-arbitrated to maximize bandwidth while the data is being written into the buffer. If buffering is not required, write buffering can be disabled by:

- setting the Buffer enable (E) field of the MPMCAHBControl Register inactive
- setting the AHB **HPROT** protection information to indicate a nonbufferable access.

Worst case additional latency

The worst case additional latency is for the case where one AHB port, in this case AHB port 0, performs INCR16 32-bit wide write transactions. Other AHB ports, in this case AHB port 1 and 2, perform continuous, in page, INCR16 64-bit write transactions. Accesses are to a 64-bit wide dynamic memory chip-select.

Write buffer enabled

With the write buffer enabled the memory writes 64 bits of data on every clock cycle.

The MPMC re-arbitrates to a different AHB port when data is being written to the buffer. The latency of a buffer transfer might therefore be longer than a nonbuffered transfer.

The worst case additional latency for a buffered transfer is when a buffered INCR16 32-bit wide burst is being performed and the other AHB ports are performing INCR16 64-bit wide transfers. See Table 2-9 for an example of the worse case scenario.

Table 2-9 Write buffer enabled

Clock cycle	AHB 0	AHB 1	AHB 2
1	32-bit write data 0 written into buffer.	AHB port waiting for data	AHB port waiting for data
2	32-bit write data 1 written into buffer. 64-bit write to memory.	AHB port waiting for data	AHB port waiting for data
3	32-bit write data 2 written into buffer.	64-bit write 0 to memory	AHB port waiting for data
4	32-bit write data 3 written into buffer.	64-bit write 1 to memory	AHB port waiting for data
5-17	AHB port waiting for data.	64-bit write 2-14 to memory	AHB port waiting for data
18	AHB port waiting for data.	64-bit write 15 to memory	AHB port waiting for data
19	64-bit write to memory.	Next INCR16 transfer	AHB port waiting for data
20	32-bit write data 4 written into buffer.	AHB port waiting for data	AHB port waiting for data
21	32-bit write data 5 written into buffer.	AHB port waiting for data	64-bit write 0 to memory
23-37	AHB port waiting for data.	AHB port waiting for data	64-bit write 1-14 to memory
38	AHB port waiting for data.	AHB port waiting for data	64-bit write 15 to memory
39	AHB port waiting for data.	AHB port waiting for data	Next INCR16 transfer
40	64-bit write to memory.	AHB port waiting for data	AHB port waiting for data

Note

Table 2-9 on page 2-19 shows the first 39 cycles of the transaction. Cycles 40 to 122 are a continuation of the sequence shown.

Write buffer disabled

Table 2-10 shows that if write buffers are not enabled an INCR16 32-bit wide write burst takes 16 cycles to complete when the write data starts being written to memory.

Table 2-10 Write buffer disabled

Clock cycle	AHB 0	AHB 1	AHB 2
1	32-bit write 0 to memory	AHB port waiting for data	AHB port waiting for data
2-15	32-bit write 2-15 to memory	AHB port waiting for data	AHB port waiting for data
16	32-bit write 16 to memory	AHB port waiting for data	AHB port waiting for data
17	AHB port available	64-bit write 0 to memory	AHB port waiting for data
18-31	AHB port available	64-bit write 1 to memory	AHB port waiting for data
32	AHB port available	64-bit write 2-14 to memory	AHB port waiting for data
33	AHB port available	64-bit write 16 to memory	64-bit write 0 to memory
34-47	AHB port available	AHB port waiting for data	64-bit write 1 to memory
48	AHB port available	AHB port waiting for data	64-bit write 2-14 to memory
49	AHB port available	AHB port available	64-bit write 16 to memory

Zero wait state write transfers

When the buffers are disabled, all write transfers receive **HREADY** LOW wait states until the transfer has started to be performed by the MPMC. **HREADY** then goes HIGH. This functionality is important for multi-port masters to ensure data coherency over multiple ports, or when data is being passed between masters. For example, the ARM11 read and write data ports require the write transfers to receive wait states until the write is being performed to ensure coherency between the two ports. This is the default operation of the MPMC after reset.

With the buffers enabled and empty, a write transfer receives no wait states and completes immediately, enabling reduced write latency if there are no data coherency issues. Subsequent writes are waited until the first write has completed.

To enable zero wait state writes to be performed without using the data merge buffers, the buffers must be enabled and the **HPROT[3:2]** lines driven correctly to indicate that reads are not cacheable and writes are not bufferable.

Note

Exclusive write transfers always have a minimum of two wait states inserted, even when the merge buffer is used.

2.1.5 Endian and packing logic

The endian and packing block performs the following:

- little-endian and big-endian conversion for ARMv5
- AHB byte strobe handling for ARM11 unaligned data support
- data packing within the read and write merge buffers.

2.1.6 Arbiter

The arbiter arbitrates between the AHB slave memory interfaces. AHB interface 0 has the highest access priority, and AHB interface 9 has the lowest priority. For more information see *Arbitration* on page 2-32.

2.1.7 MPMC state machine

The MPMC state machine comprises two functional blocks:

- a static memory controller (including the NAND flash controller)
- a dynamic memory controller.

Low transaction latency and high memory bandwidth are conflicting design parameters.

A memory controller designed to support high bandwidth has logic to reorder transactions to maximize memory efficiency. This logic increases transaction latency.

A memory controller designed to reduce latency has less logic for the transaction to pass through, reducing the amount of logic used to maximize the transaction efficiency. This decreases the supported memory bandwidth.

The MPMC is designed with both these parameters in mind, and has both good latency and memory bandwidth.

Providing multiple AHB memory ports in the MPMC enables the memory bandwidth to be shared over multiple AHB buses. This reduces the load on performance-critical buses.

To make full use of dynamic memory bandwidth a multiple-port design is required so that:

- the transaction order can be rearranged to maximize the number of in page accesses
- the dynamic memory transactions can be pipelined
- the dynamic memory transactions can be interleaved.

2.1.8 External DLL block

An external *Delay Locked Loop* (DLL) is required to enable a programmable delay to be added to the fed-back clocks or data strobes used by the PrimeCell MPMC so that data can be captured accurately. For each byte of the SDRAM read data bus there is an associated data strobe, **MPMCDQSIN[3:0]**. Each input data strobe is routed with its associated data signals on the PCB to ensure that they are delayed by the same amount.

For reads, the data strobe input signals from the memory device are edge-aligned with the read data. The function of the external DLL is to generate a calibrated delay which, when applied to the data strobes, ensures that the read data is registered accurately at the MPMC.

External DLL calibration

The PrimeCell MPMC provides standard DLL calibration signals, **MPMCDLLCALIBREQ** and **MPMCDLLCALIBACK**, that can be used by systems using a DLL block. DLL calibration is enabled by setting the MPMCDynamicControl DE bit. If DLL calibration is disabled, the calibration request signal **MPMCDLLCALIBREQ** does not go active.

DLL calibration is requested by the PrimeCell MPMC. If enabled, the calibration request signal, **MPMCDLLCALIBREQ**, goes active during SDRAM refresh, self-refresh exit and after DLL calibration has been enabled. DLL calibration can also be initiated by software by setting the MPMCDynamicControl DLL Calibrate, DC, bit. When the calibration request signal goes active, the DLL block brings the calibration acknowledge signal, **MPMCDLLCALIBACK**, HIGH. The DLL then performs calibration. When the DLL has performed its calibration it brings the acknowledge signal LOW. The MPMC then brings its calibration request signal LOW and enables the memory to be accessed.

——— **Note** ———

No memory transactions, except for SDRAM refresh, take place while the DLL is being calibrated.

Software programmed external DLL calibration

If DLL calibration is initiated by software by asserting the MPMCDynamicControl DC bit, the **MPMCDLLCALIBREQ** signal goes active. The DLL then brings the **MPMCDLLCALIBACK** signal HIGH and starts calibration. When calibrated, the **MPMCDLLCALIBACK** signal is brought LOW by the DLL block. The MPMC MPMCStatus DLL Status, DS, bit then goes HIGH, to indicate that the DLL has been calibrated. The software polls the DS bit until it goes HIGH, and then clears the MPMCDynamicControl DC bit. The memory can then be accessed.

2.2 Overview of a PrimeCell MPMC, ASIC, or ASSP system

Figure 2-4 shows the PrimeCell MPMC (PL176) in an example system.

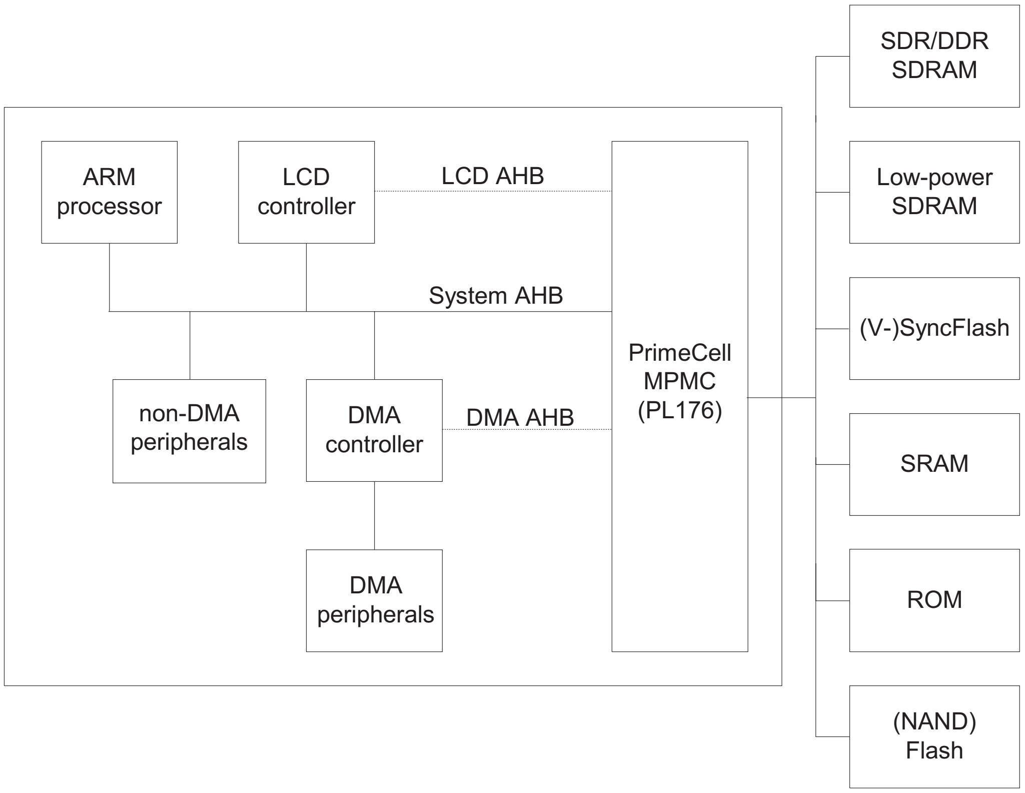


Figure 2-4 PrimeCell MPMC (PL176) in an example system

The example system uses two types of buses:

- *External bus*
- *Internal bus* on page 2-25.

2.2.1 External bus

The off-chip bus that contains data, address, and control signals, connects the ASIC or ASSP to the external memory.

Note

- Connecting a large number of memory devices externally impacts on performance because of signal loading.
- Only one memory device can be accessed at a time.

2.2.2 Internal bus

The on-chip bus enables communication between the on-chip peripherals. The PrimeCell MPMC appears as a standard slave on the on-chip bus and controls the memory on the external bus.

Providing multiple AHB interfaces improves system performance by enabling several access requests to be presented to the MPMC at the same time. This enables the MPMC to pipeline many of the operations (for example, bank activate and precharge), and so reduce the average system access latency and improve utilization of external memory. The use of multiple AHB interfaces also improves system performance by removing heavy DMA traffic from the main AHB bus.

2.3 AHB slave memory interface priority

AHB interface 0 has the highest access priority, followed by AHB interface 1 through to AHB interface 8. AHB interface 9 has the lowest priority.

2.3.1 AHB memory port latency

Each AHB memory port has a programmable TimeOut register. When a memory request is made to the port the value of the counter is loaded. Every cycle where the port transaction is not serviced the TimeOut register counts down. When the TimeOut counter reaches zero, the priority of the port is increased to a level higher than all ports which have not timed out. This functionality enables each AHB memory port be programmed with a deterministic latency. This also enables the amount of bandwidth that a port consumes to be indirectly defined.

2.4 Low power operation

In many systems, the contents of the memory system have to be maintained during low-power sleep modes. The PrimeCell MPMC provides two features to enable this:

- dynamic memory refresh over soft reset
- a mechanism to place the dynamic memories into self-refresh mode.

Self-refresh mode can be entered automatically by hardware or manually by software:

- It can be entered manually setting the SREFREQ bit in the MPMCDynamicControl Register and polling the SREFACK bit in the MPMCStatus Register.
- It can be entered automatically using a *Power Management Unit* (PMU). This is typically present to control the safe transition between the following modes:
  - power-up
  - reset
  - normal
  - sleep.

The PMU can be used to enable self-refresh mode to be entered automatically. To do this, the PMU asserts the **MPMCSREFREQ** signal when self-refresh mode is to be entered. The MPMC then closes any open memory banks and puts the external memory into self-refresh mode. The MPMC then asserts the **MPMCSREFACK** signal to indicate to the PMU that self-refresh mode is entered. The system must ensure that the memory subsystem is idle before asserting **MPMCSREFREQ**. Any transactions to memory that are generated while the MPMC is in self-refresh mode are rejected and an error response is generated (**HRESP** = ERROR). Deasserting **MPMCSREFREQ** returns the memory to normal operation. See the memory data sheet for refresh requirements.

Note

- If **MPMCSREFREQ** is not required this signal must be tied LOW.
- Static memory can be accessed as normal when the SDRAM memory is in self-refresh mode.

2.4.1 Low-power SDRAM deep sleep mode

The PrimeCell MPMC supports JEDEC low-power SDRAM deep-sleep mode. Deep sleep mode can be entered by setting the deep-sleep mode (DP) bit in the MPMCDynamicControl Register. The device is then put into a low-power mode where the device is powered down and no longer refreshed. All data in the memory is lost.

2.4.2 Low-power SDRAM partial array refresh

The MPMC supports JEDEC low-power SDRAM partial array refresh. Partial array refresh can be programmed by initializing the SDRAM memory device appropriately. When the memory device is put into self-refresh mode only the memory banks specified are refreshed. The memory banks that are not refreshed lose their data contents.

2.5 Lock and semaphores

Locked accesses on the AHB bus, transactions where **HMASTLOCK** is HIGH, are processed appropriately. When the MPMC performs a locked transfer, it does not re-arbitrate to another AHB port until the locked transfer is completed.

2.6 Burst types

All AHB burst types are supported.

Note

INCR transfers are split internally into four-word bursts.

2.7 Busy transfer type

When an AHB master generates busy (AHB HTRANS=BUSY) cycles the MPMC waits until busy is inactive before completing the transfer. This increases transfer latency. The MPMC does not re-arbitrate to another AHB port if busy goes active.

2.8 Arbitration

This section describes the PrimeCell MPMC AHB memory port arbitration strategy.

Note

- It is recommended that the AHB arbiter re-arbitrate on a burst boundary, because this leads to the most efficient bus utilization.
- AHB protocol does not enable AHB masters to break a defined length burst transfer (INCR4, WRAP4, INCR8, WRAP8, INCR16, and WRAP16). An AHB master can only break an undefined length burst transfer (INCR).
- It is not possible for an AHB burst to cross an SDRAM column, because AHB bursts must not cross a 1KB boundary, and the smallest SDRAM column length supported is 1KB long.

2.8.1 Re-arbitration occurrence

The re-arbitration occurrence is the time when the MPMC re-arbitrates. The following lists AHB transfers which affect re-arbitration:

- When a locked transfer (as defined by the AHB **HLOCK** signal) has started the AHB memory port is not re-arbitrated until the locked transfer has completed and the AHB lock signal is deasserted.
- When a fixed-length AHB burst transfer has started, the AHB memory port is not re-arbitrated until the burst has completed.
- If an 8-bit, 16-bit, or 32-bit wide AHB burst is submitted and the AHB port buffers are enabled, the MPMC can re-arbitrate to a different AHB port when data is being written to or read from the buffer.

  Note

  The longest AHB transfer is 16 AHB transfers long (INCR16/WRAP16).

- The MPMC does not re-arbitrate to another AHB port if **HTRANS**=BUSY.

2.8.2 Re-arbitration priority

The following lists the re-arbitration scheme:

- Auto-refresh is generated

- The highest priority AHB memory port is a port where the TimeOut register has timed out.

  If more than one port has timed out, the port with the lowest port number is selected. In other words port 0 is selected over port 1.

- If none of the AHB memory ports have timed out, the next highest priority port is for the port where the transaction is to an already open SDRAM memory row.

  If more than one port transaction is to an already-open SDRAM memory row, the port with the lowest port number is selected.

- If none of the AHB memory ports have timed out and none of the accesses are to open SDRAM memory rows, the memory request to the lowest port number is selected. In other words port 0 is selected over port 1.

  ——— **Note** ———

  When a buffered 8-bit, 16-bit, or 32-bit wide transfer is in progress the MPMC does not re-arbitrate to a different AHB port if the access is to an unopened SDRAM row.

  ————————

2.9 Worst case transaction latency

The worst case transaction latency for the highest priority AHB memory port is 58 clock cycles. The worst case latency of the lower priority AHB memory ports is TimeOut + 58 cycles, assuming that another higher priority port has not timed out. These values are based on the assumptions given in the following sections:

- *Worst case transaction latency for the highest priority AHB memory port*
- *Worst case transaction latency for the lower priority AHB memory ports* on page 2-35
- *System factors affecting worst case latency* on page 2-37.

2.9.1 Worst case transaction latency for the highest priority AHB memory port

The following assumptions were made for calculating the worst case transaction latency:

- System factors are ignored in these calculations. For information on system factors see *System factors affecting worst case latency* on page 2-37.
- SDRAM memory latency values are:
  - precharge = 3
  - active = 3
  - CAS = 3
  - auto-refresh (t_{RFC}) = 7.
- The SDRAM memory chip selects have a 64-bit wide data bus.
- There are no devices connected to the static memory chip selects.
- AHB port 0 TimeOut register is programmed to be less than or equal to the slowest transfer. This is normally either a INCR16 or WRAP16 read to an unopened SDRAM page.

Note

Using SDRAM memory chip selects with a 16-bit or 32-bit wide data bus, SDRAM memory with larger latency values, or using slow static memory devices in a system might impact the worst case latency.

For the PrimeCell MPMC the worst case latency scenario is as follows:

- A lower priority port is performing an INCR16 read request. The read data is to a different row from the one already opened. Therefore a precharge, activate, and read command must be sent to the SDRAM.

- An auto-refresh is generated after the INCR16 read is submitted.
- The highest priority AHB memory port performs an INCR16 read transaction. The read data is to a different row from the one already opened, therefore a precharge, activate, and read command must be sent to the SDRAM.

The read data is to unopened SDRAM memory rows.

Before the first data from the highest priority AHB memory port is returned the following transactions occur:

1. The INCR16 read is performed. The read data is to a different row from the one already opened. This transaction takes 30 cycles.
2. The SDRAM auto-refresh is generated. This transaction takes 13 cycles.
3. The AHB memory port 0 priority read transaction is performed. The read data is to a different row from the one already opened. The first data is returned after 15 cycles. The following 15 transfers of the burst take an additional 15 cycles to be performed.

Worst case transaction latency for highest priority AHB memory port = 30+13+15 = 58 cycles (first data returned).

Worst case latency for highest priority AHB memory port to complete = 30+13+15+15 = 73 cycles (last piece of longest AHB data returned).

2.9.2 Worst case transaction latency for the lower priority AHB memory ports

The following assumptions were made for calculating the worst case transaction latency:

- System factors are ignored in these calculations. For information on system factors see *System factors affecting worst case latency* on page 2-37.
- SDRAM memory latency values are:
  - — precharge = 3
  - — active = 3
  - — CAS = 3
  - — auto-refresh (t_{RFC}) = 7.
- The SDRAM memory chip selects have a 64-bit wide data bus.
- There are no devices connected to the static memory chip selects.

- The required AHB port latency is programmed into the appropriate TimeOut register. The value programmed is normally the required timeout latency minus the latency of the slowest burst transfer.
- No other AHB ports have their TimeOut registers programmed.

Note

Using SDRAM memory chip selects with a 16-bit or 32-bit wide data bus, SDRAM memory with larger latency values, or using slow static memory devices in a system might impact the worst case latency.

For the PrimeCell MPMC the worst case latency for the lower priority memory port scenario is as follows:

- The AHB memory port that you are interested in submits a read transaction. However this transaction is not performed because there are higher priority transactions. The read data is to a different row from the one already opened. Therefore a precharge, activate, and read command must be sent to the SDRAM.
- An INCR16 read is submitted to a higher priority port. The read data is to unopened SDRAM memory rows.
- The TimeOut register for the AHB memory port you are interested in times out.
- An auto-refresh is generated after the INCR16 read is submitted.
- The AHB memory port you are interested in performs a read access. The read data is to a different row from the one already opened. Therefore a precharge, activate, and read command must be sent to the SDRAM.

The following transactions occur before the first data is returned from the port you are interested in:

1. Read transaction submitted.
2. The INCR16 read is performed for higher priority port. The read data is to a different row from the one already opened. This transaction takes 30 cycles.
3. The TimeOut counter for the port you are interested in counts to 0.
4. The SDRAM auto-refresh is generated. This transaction takes 13 cycles.
5. The highest priority read transaction is performed. The read is to an unopened SDRAM memory row. The first data is returned after 15 cycles.The following 15 transfers of the burst take an additional 15 cycles to be performed.

Worst case transaction latency for highest priority AHB memory port = 30+13+15 = 58 cycles.

Worst case latency for highest priority AHB memory port to complete = 30+13+15+15 = 73 cycles.

2.9.3 System factors affecting worst case latency

This section describes the system factors that can affect the MPMC worst case latency to a memory request.

MPMC AHB Memory port priority

The MPMC AHB memory ports are prioritized so that the most efficient transfers are performed first. High priority ports, AHB ports with a low value, are selected in preference to lower priority ports. The AHB TimeOut register enables the maximum latency for a port to be programmed.

AHB bus

If there are multiple AHB masters on the same AHB bus, then a master can only receive ownership of the bus, and submit transactions, when the AHB arbiter grants the bus to the master. The arbitration strategy in the AHB arbiter then affects the worst case latency for a system with multiple masters on an AHB bus.

AHB masters

When an AHB master generates busy (AHB HTRANS=BUSY) cycles the MPMC waits until busy is inactive before completing the transfer. This increases transfer latency. The MPMC does not re-arbitrate to another AHB port if busy goes active.

If an AHB master performs locked (AHB HMASTLOCK=LOCK) cycles this means that the MPMC cannot re-arbitrate until the locked transaction has completed. This can then increase worst case transfer latency, because the MPMC is not able to re-arbitrate to a higher priority port until the locked access has completed.

Memory devices

If slow SDRAM memories and/or a 16-bit SDRAM chip select are used, then transactions take longer to complete. This affects transfer latency.

Pad multiplexing

If the memory bus is multiplexed externally, for example, by using an EBI, the worst case transfer latency is affected because the external bus is shared by multiple devices.

2.10 Sharing memory bandwidth between AHB ports

By programming the AHB port TimeOut values appropriately the bandwidth of the MPMC can be shared between the AHB ports of the MPMC.

2.10.1 Typical AHB port TimeOut value given bandwidth requirement

To meet a given bandwidth requirement the TimeOut value must be programmed less than the value provided by the following formula:

$$\text{TimeOut value} = \frac{\text{(AHB frequency) x (number of bytes in average AHB burst)}}{\text{(required AHB bandwidth)}} - \text{(number of transactions in AHB burst)}$$

Note

This formula assumes that the AHB masters on each of the AHB ports always have requests pending.

2.10.2 Typical AHB port TimeOut value given percentage of memory bandwidth requirement

To compute the TimeOut values that must be programmed for the case where each AHB port requires a percentage of the total memory bandwidth, first the expected memory bandwidth of the system must be calculated. From this the bandwidth per port can be calculated. Finally you can use the formula in *Typical AHB port TimeOut value given bandwidth requirement*.

2.10.3 Typical AHB bandwidth requirement example

This section provides an example of how to program the AHB port TimeOut registers given multiple AHB ports and their bandwidth requirements.

System

The MPMC, AHB bus, and SDR-SDRAM memory operate at 100MHz. The SDR-SDRAM has a 64-bit wide data bus.

Bandwidth requirement

The AHB bandwidth requirements are:

- AHB port 0 requires at least 40MB/s
- AHB port 1 requires at least 20MB/s
- AHB port 2 requires at least 2MB/s.

The AHB burst types are:

- AHB port 0 normally performs AHB INCR16, 64-bit wide transfers
- AHB port 1 normally performs AHB INCR8, 32-bit wide transfers
- AHB port 2 normally performs AHB INCR4, 16-bit wide transfers.

Calculations required

The TimeOut values can be calculated as indicated below.

AHB port 0 TimeOut value Each INCR16 transfer provides 128 bytes of data. Therefore there are 327,680 (40MB divided by 128B) INCR16 bursts required per second to satisfy the minimum bandwidth requirement. On average there are 305 clock cycles (100M clock cycles divided by 327,680) between AHB port 0 transactions to meet the requirement. Each transaction takes about 16 cycles to complete.

$$\text{TimeOut value} = \frac{\text{100MHz x 128}}{\text{40MB}} - 16 = 289 \text{ cycles}$$

Therefore the AHB TimeOut value for AHB port 0 must be programmed to less than 289 clock cycles.

AHB port 1 TimeOut value Each INCR8 transfer provides 32 bytes of data. Therefore there are 655,360 (20MB divided by 32B) INCR8 bursts required per second to satisfy the minimum bandwidth requirement. On average there are 152 clock cycles (100M clock cycles divided by 655,360) between AHB port 1 transactions to meet the requirement. Each transaction takes about eight cycles to complete.

$$\text{TimeOut value} = \frac{\text{100MHz x 32}}{\text{20MB}} - 8 = 144 \text{ cycles}$$

Therefore the AHB TimeOut value for AHB port 1 must be programmed to less than 144 clock cycles.

AHB port 2 TimeOut value Each INCR4 transfer provides eight bytes of data. Therefore there are 262,144 (2MB divided by 8B) INCR4 bursts required per second to satisfy the minimum bandwidth requirement. On average there are 381 clock cycles (100M clock cycles divided by 262,144) between AHB port two transactions to meet the requirement. Each transaction takes about four cycles to complete.

$$\text{TimeOut value} = \frac{100\text{MHz} \times 8}{2\text{MB}} - 4 = 377 \text{ cycles}$$

Therefore the AHB TimeOut value for AHB port 1 must be programmed to less than 377 clock cycles.

2.10.4 Typical AHB percentage bandwidth example

This section provides an example of how to program the AHB port TimeOut registers given multiple AHB ports and their percentage of bandwidth requirements.

System

The MPMC, AHB bus, and SDR-SDRAM memory operate at 100MHz. The SDR-SDRAM has a 64-bit wide data bus.

Bandwidth requirement

The AHB bandwidth requirements are:

- AHB port 0 requires 10% of memory bandwidth minimum
- AHB port 1 requires 5% of memory bandwidth minimum
- AHB port 2 requires 0.5% of memory bandwidth minimum.

The AHB burst types are:

- AHB port 0 normally performs AHB INCR16, 64-bit wide transfers
- AHB port 1 normally performs AHB INCR8, 32-bit wide transfers
- AHB port 2 normally performs AHB INCR4, 16-bit wide transfers.

Calculations required

The TimeOut values can be calculated as indicated below.

Compute the real system bandwidth Theoretically the SDRAM memory can provide 781MB/s. However, we assume that the real system bandwidth is 400MB/s because of the system burst types and page hit rates.

Note

The theoretical value of 781MB/s is obtained by multiplying 100MHz by 8 bytes (64-bit), and dividing by 1024 twice to convert into MB.

Calculate the memory bandwidth per AHB port For AHB port 0 10% of the 400MB/s MPMC bandwidth is 40MB/s.

For AHB port 1 5% of the 400MB/s MPMC bandwidth is 20MB/s.

For AHB port 2 0.5% of the 400MB/s MPMC bandwidth is 2MB/s.

Compute the TimeOut value using the following formula

$$\text{TimeOut value} = \frac{\text{(AHB frequency) x (number of bytes in average AHB burst)}}{\text{(required AHB bandwidth)}} - \text{(number of transactions in AHB burst)}$$

For the AHB port 0 the TimeOut value is:

$$\text{TimeOut value} = \frac{\text{100MHz x 128}}{\text{40MB}} - 16 = 289 \text{ cycles}$$

For the AHB port 1 the TimeOut value is:

$$\text{TimeOut value} = \frac{\text{100MHz x 32}}{\text{20MB}} - 8 = 144 \text{ cycles}$$

For the AHB port 2 the TimeOut value is:

$$\text{TimeOut value} = \frac{\text{100MHz x 8}}{\text{2MB}} - 4 = 377 \text{ cycles}$$

2.11 Memory bank select

Eight independently-configurable memory banks are supported, with a separate chip select output for each bank. Banks 0 to 3 are used to select static memory devices, and banks 4 to 7 used to select dynamic memory devices.

The chip selects are selected by eight separate **HSELMPMCxCSn** signals, as shown in Table 2-11.

Table 2-11 Memory bank selection

AHB select	Bank	Device	Chip select
HSELMPMCxCS0	0	Static Mem 0	**nMPMCSTCSOUT[0]**
HSELMPMCxCS1	1	Static Mem 1	**nMPMCSTCSOUT[1]**
HSELMPMCxCS2	2	Static Mem 2	**nMPMCSTCSOUT[2]**
HSELMPMCxCS3	3	Static Mem 3	**nMPMCSTCSOUT[3]**
HSELMPMCxCS4	4	Sync Mem 0	**nMPMCDYCSOUT[0]**
HSELMPMCxCS5	5	Sync Mem 1	**nMPMCDYCSOUT[1]**
HSELMPMCxCS6	6	Sync Mem 2	**nMPMCDYCSOUT[2]**
HSELMPMCxCS7	7	Sync Mem 3	**nMPMCDYCSOUT[3]**

The amount of memory allocated to a chip select is determined by the system AHB decoder. A further select line, **HSELMPMCxG**, is provided at each AHB slave port to select the MPMC slave. The largest amount of memory that can be allocated to a single chip select is 256Mb.

Note

Chip selects connected to Micron SyncFlash or V-SyncFlash must have **HADDR[28:27]** available for use as the software control for generating *Hardware Command Sequences* (HCS). If HCSs are used, the AHB decoder can only use **HADDR[31:29]** for decoding the chip select. If HCSs are not used, the chip select must be located at an address where **HADDR[28]** is zero.

2.12 Memory map

The MPMC provides hardware support for booting from external nonvolatile memory. During booting the nonvolatile memory must be located at address `0x00000000` in memory. When the system is booted the SRAM or SDRAM memory can be remapped to address `0x00000000` by modifying the address map in the AHB decoder.

2.12.1 Chip select 1 static memory configuration

The memory width, chip select polarity, and byte lane select polarity of static memory chip select 1 can be configured by using several input tie-off signals. This enables you to boot from chip select 1.

The configuration tie-off signals are **MPMCSTCS1MW[1:0]**, memory width select, **MPMCSTCS1POL**, chip select polarity, and **MPMCSTCS1PB**, byte lane select polarity.

2.12.2 Chip select 5 SDRAM memory configuration

The address mapping and memory device type of SDRAM memory chip select 5 can be configured by using several input tie-off signals. This enables you to boot from chip select 5.

The configuration tie-off signals are:

- **MPMCDYCS5ADRMAP[8:0]**, address mapping
- **MPMCDYCS5DEVICE[2:0]**, memory device
- **MPMCDYCS5CASDLY[3:0]**, CAS delay for chip select 5
- **MPMCDYSDRPOL**, polarity of the flip-flop in which the read data is first captured
- **MPMCDYSDRDLY[1:0]**, SDR clocking scheme for data capture.

2.12.3 Example of a boot from flash, SRAM remapped after boot

The system set up is:

- chip select 1 is connected to the boot flash device
- chip select 0 is connected to the SRAM to be remapped to `0x00000000` after boot
- the AHB decoder contains the functionality to remap the address.

The boot sequence is as follows:

1. At power on the reset chip select 1 is located at `0x00000000` in the AHB address map. The following signals are configured so that the nonvolatile memory device can be accessed:
   - **MPMCSTCS1MW[1:0]**
   - **MPMCSTCS1PB**
   - **MPMCSTCS1POL** (also **MPMCSTCS0POL**, **MPMCSTCS2POL**, and **MPMCSTCS3POL**).
2. When the power-on reset (**nPOR**) and AHB reset (**HRESETn**) go inactive, the processor starts booting from `0x00000000` in memory.
3. The software programs the optimum delay values in the flash memory so that the boot code can run at full speed.
4. The other chip selects are initialized.
5. The AHB decoder address map is modified so that the SRAM is located at address `0x00000000`.
6. The ARM reset and interrupt vectors are copied from flash memory to SRAM that can then be accessed at address `0x00000000`.
7. More boot, initialization, or application code is executed.

2.12.4 Example of a boot from flash, SDRAM remapped after boot

The system set up is:

- chip select 1 is connected to the boot flash device
- chip select 4 is connected to the SDRAM to be remapped to `0x00000000` after boot
- the AHB decoder contains the functionality to remap the address.

The boot sequence is as follows:

1. At power on the reset chip select 1 is located at `0x00000000` in the AHB address map. The following signals are configured so that the nonvolatile memory device can be accessed:
   - **MPMCSTCS1MW[1:0]**
   - **MPMCSTCS1PB**
   - **MPMCSTCS1POL** (also **MPMCSTCS0POL**, **MPMCSTCS2POL**, and **MPMCSTCS3POL**).

2. When the power-on reset (**nPOR**) and AHB reset (**HRESETn**) go inactive, the processor starts booting from `0x00000000` in memory.
3. The software programs the optimum delay values in the flash memory so that the boot code can run at full speed.
4. The other chip selects are initialized.
5. The AHB decoder address map is modified so that the SDRAM is located at address `0x00000000`.
6. The ARM reset and interrupt vectors are copied from flash memory to SDRAM that can then be accessed at address `0x00000000`.
7. More boot, initialization, or application code is executed.

2.12.5 Memory aliasing

Memory aliasing is allowed.

2.12.6 Unused AHB HADDRx address bits

If some of the **HADDRx** address bits are not required, the unused address bits must be tied LOW.

2.13 Sharing memory interface signals

The memory interface signals, and the MPMC primary input and output signals, can be shared with other peripherals by using an *External Bus Interface* (EBI) block. For more information on the EBI see *EBI connectivity* on page 9-18.

Chapter 3
Programmer's Model

This chapter describes the ARM PrimeCell MPMC (PL176) registers and provides details required when programming the microcontroller. It contains the following sections:

3.1 About the programmer's model

The base address of the PrimeCell MPMC is not fixed, but is determined by the AHB decoder, and can be different for any particular system implementation. However, the offset of any particular register from the base address is fixed. The registers of the PrimeCell MPMC are selected by the **HSELMPMCREG** signal.

The external memory addresses are not fixed and can be different for any particular system implementation. Transfers to the MPMC external memories are selected by the **HSELMPMCxCSn** signals.

Note

AHB masters that are 64 bits wide cannot use load or store multiple instructions to access the control registers, and writes must be aligned to the transfer size. For ARM11, the control registers must be placed in device memory space (ARMv6 memory region type).

The PrimeCell MPMC registers are shown in Table 3-1.

Table 3-1 PrimeCell MPMC register summary

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCControl	`0x000`	Read/write	2	`0x1`	-	See *Control Register, MPMCControl* on page 3-12
MPMCStatus	`0x004`	Read	4	-	`0x15`	See *Status Register, MPMCStatus* on page 3-13
MPMCConfig	`0x008`	Read/write	1	-	0x00	See *Configuration Register, MPMCConfig* on page 3-13
MPMCDynamic Control	`0x020`	Read/write	14	-	`0x006`	See *Dynamic Memory Control Register, MPMCDynamicControl* on page 3-14
MPMCDynamic Refresh	`0x024`	Read/write	11	-	`0x0`	See *Dynamic Memory Refresh Timer Register, MPMCDynamicRefresh* on page 3-16
MPMCDynamic ReadConfig	`0x028`	Read/write	6	-	`0x----`[a]	See *Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig* on page 3-17

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCDynamic tRP	`0x030`	Read/write	4	-	`0xF`	See *Dynamic Memory Precharge Command Period Register, MPMCDynamictRP* on page 3-18
MPMCDynamic tRAS	`0x034`	Read/write	4	-	`0xF`	See *Dynamic Memory Active To Precharge Command Period Register, MPMCDynamictRAS* on page 3-19
MPMCDynamic tSREX	`0x038`	Read/write	7	-	`0x7F`	See *Dynamic Memory Self-refresh Exit Time Register, MPMCDynamictSREX* on page 3-19
MPMCDynamic tWR	`0x044`	Read/write	4	-	`0xF`	See *Dynamic Memory Write Recovery Time Register, MPMCDynamictWR* on page 3-20
MPMCDynamic tRC	`0x048`	Read/write	5	-	`0x1F`	See *Dynamic Memory Active To Active Command Period Register, MPMCDynamictRC* on page 3-21
MPMCDynamic tRFC	`0x04C`	Read/write	5	-	`0x1F`	See *Dynamic Memory Auto-refresh Period Register, MPMCDynamictRFC* on page 3-21
MPMCDynamic tXSR	`0x050`	Read/write	8	-	`0xFF`	See *Dynamic Memory Exit Self-refresh Register, MPMCDynamictXSR* on page 3-22
MPMCDynamic tRRD	`0x054`	Read/write	4	-	`0xF`	See *Dynamic Memory Active Bank A To Active Bank B Time Register, MPMCDynamictRRD* on page 3-22
MPMCDynamic tMRD	`0x058`	Read/write	4	-	`0xF`	See *Dynamic Memory Load Mode Register, MPMCDynamictMRD* on page 3-23
MPMCDynamic tCDLR	`0x05C`	Read/write	4	-	`0xF`	See *Dynamic Memory Last Data In To Read Command Time Register, MPMCDynamictCDLR* on page 3-24
MPMCStatic ExtendedWait	`0x080`	Read/write	10	-	`0x000`	See *Static Memory Extended Wait Register, MPMCStaticExtendedWait* on page 3-24

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCDynamicConfig0	`0x100`	Read/write	13	-	`0x 00000000`	See *Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3* on page 3-25
MPMCDynamicRasCas0	`0x104`	Read/write	8	-	`0x783`	See *Dynamic Memory RAS and CAS Delay Registers 0-3, MPMCDynamicRasCas0-3* on page 3-30
MPMCDynamicConfig1	`0x120`	Read/write	13	-	`0x 0000----`[a]	See *Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3* on page 3-25
MPMCDynamicRasCas1	`0x124`	Read/write	8	-	`0x--3`[a]	See *Dynamic Memory RAS and CAS Delay Registers 0-3, MPMCDynamicRasCas0-3* on page 3-30
MPMCDynamicConfig2	`0x140`	Read/write	13	-	`0x 00000000`	See *Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3* on page 3-25
MPMCDynamicRasCas2	`0x144`	Read/write	8	-	`0x783`	See *Dynamic Memory RAS and CAS Delay Registers 0-3, MPMCDynamicRasCas0-3* on page 3-30
MPMCDynamicConfig3	`0x160`	Read/write	13	-	`0x 00000000`	See *Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3* on page 3-25
MPMCDynamicRasCas3	`0x164`	Read/write	8	-	`0x783`	See *Dynamic Memory RAS and CAS Delay Registers 0-3, MPMCDynamicRasCas0-3* on page 3-30
MPMCStaticConfig0	`0x200`	Read/write	7	-	`0x 0000-0`[a]	See *Static Memory Configuration Registers 0-3, MPMCStaticConfig0-3* on page 3-30

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCStatic WaitWen0	`0x204`	Read/write	4	-	`0x0`	See *Static Memory Write Enable Delay Registers 0-3, MPMCStaticWaitWen0-3* on page 3-32
MPMCStatic WaitOen0	`0x208`	Read/write	4	-	`0x0`	See *Static Memory Output Enable Delay Registers 0-3, MPMCStaticWaitOen0-3* on page 3-33
MPMCStatic WaitRd0	`0x20C`	Read/write	5	-	`0x1F`	See *Static Memory Read Delay Registers 0-3, MPMCStaticWaitRd0-3* on page 3-33
MPMCStatic WaitPage0	`0x210`	Read/write	5	-	`0x1F`	See *Static Memory Page Mode Read Delay Registers 0-3, MPMCStaticWaitPage0-3* on page 3-34
MPMCStatic WaitWr0	`0x214`	Read/write	5	-	`0x1F`	See *Static Memory Write Delay Registers 0-3, MPMCStaticWaitWr0-3* on page 3-34
MPMCStatic WaitTurn0	`0x218`	Read/write	4	-	`0xF`	See *Static Memory Turn Round Delay Registers 0-3, MPMCStaticWaitTurn0-3* on page 3-35
MPMCStatic Config1	`0x220`	Read/write	7	-	`0x 0000--`[a]	See *Static Memory Configuration Registers 0-3, MPMCStaticConfig0-3* on page 3-30
MPMCStatic WaitWen1	`0x224`	Read/write	4	-	`0x0`	See *Static Memory Write Enable Delay Registers 0-3, MPMCStaticWaitWen0-3* on page 3-32
MPMCStatic WaitOen1	`0x228`	Read/write	4	-	`0x0`	See *Static Memory Output Enable Delay Registers 0-3, MPMCStaticWaitOen0-3* on page 3-33

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCStatic WaitRd1	`0x22C`	Read/write	5	-	`0x1F`	See *Static Memory Read Delay Registers 0-3, MPMCStaticWaitRd0-3* on page 3-33
MPMCStatic WaitPage1	`0x230`	Read/write	5	-	`0x1F`	See *Static Memory Page Mode Read Delay Registers 0-3, MPMCStaticWaitPage0-3* on page 3-34
MPMCStatic WaitWr1	`0x234`	Read/write	5	-	`0x1F`	See *Static Memory Write Delay Registers 0-3, MPMCStaticWaitWr0-3* on page 3-34
MPMCStatic WaitTurn1	`0x238`	Read/write	4	-	`0xF`	See *Static Memory Turn Round Delay Registers 0-3, MPMCStaticWaitTurn0-3* on page 3-35
MPMCStatic Config2	`0x240`	Read/write	7	-	`0x 0000-0`[a]	See *Static Memory Configuration Registers 0-3, MPMCStaticConfig0-3* on page 3-30
MPMCStatic WaitWen2	`0x244`	Read/write	4	-	`0x0`	See *Static Memory Write Enable Delay Registers 0-3, MPMCStaticWaitWen0-3* on page 3-32
MPMCStatic WaitOen2	`0x248`	Read/write	4	-	`0x0`	See *Static Memory Output Enable Delay Registers 0-3, MPMCStaticWaitOen0-3* on page 3-33
MPMCStatic WaitRd2	`0x24C`	Read/write	5	-	`0x1F`	See *Static Memory Read Delay Registers 0-3, MPMCStaticWaitRd0-3* on page 3-33
MPMCStatic WaitPage2	`0x250`	Read/write	5	-	`0x1F`	See *Static Memory Page Mode Read Delay Registers 0-3, MPMCStaticWaitPage0-3* on page 3-34

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCStatic WaitWr2	`0x254`	Read/write	5	-	`0x1F`	See *Static Memory Write Delay Registers 0-3, MPMCStaticWaitWr0-3* on page 3-34
MPMCStatic WaitTurn2	`0x258`	Read/write	4	-	`0xF`	See *Static Memory Turn Round Delay Registers 0-3, MPMCStaticWaitTurn0-3* on page 3-35
MPMCStatic Config3	`0x260`	Read/write	7	-	`0x 0000-0`[a]	See *Static Memory Configuration Registers 0-3, MPMCStaticConfig0-3* on page 3-30
MPMCStatic WaitWen3	`0x264`	Read/write	4	-	`0x0`	See *Static Memory Write Enable Delay Registers 0-3, MPMCStaticWaitWen0-3* on page 3-32
MPMCStatic WaitOen3	`0x268`	Read/write	4	-	`0x0`	See *Static Memory Output Enable Delay Registers 0-3, MPMCStaticWaitOen0-3* on page 3-33
MPMCStatic WaitRd3	`0x26C`	Read/write	5	-	`0x1F`	See *Static Memory Read Delay Registers 0-3, MPMCStaticWaitRd0-3* on page 3-33
MPMCStatic WaitPage3	`0x270`	Read/write	5	-	`0x1F`	See *Static Memory Page Mode Read Delay Registers 0-3, MPMCStaticWaitPage0-3* on page 3-34
MPMCStatic WaitWr3	`0x274`	Read/write	5	-	`0x1F`	See *Static Memory Write Delay Registers 0-3, MPMCStaticWaitWr0-3* on page 3-34
MPMCStatic WaitTurn3	`0x278`	Read/write	4	-	`0xF`	See *Static Memory Turn Round Delay Registers 0-3, MPMCStaticWaitTurn0-3* on page 3-35

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCNAND Control	`0x300`	Read/write	28	-	`0x 00010000`	See *NAND Memory Control Vector Register, MPMCNANDControl* on page 3-36
MPMCNAND Address1	`0x304`	Read/write	32	-	`0x 00000000`	See *NAND Memory Address Vectors 1-4 Register, MPMCNANDAddress1* on page 3-37
MPMCNAND Address2	`0x308`	Read/write	8	-	`0x00`	See *NAND Memory Address Vector 5 Register, MPMCNANDAddress2* on page 3-38
MPMCNAND Timing1	`0x320`	Read/write	25	-	`0x 1FFFFFF`	See *NAND Memory Timing Value 1 Register, MPMCNANDTiming1* on page 3-38
MPMCNAND Timing2	`0x324`	Read/write	18	-	`0x 1F71F1F`	See *NAND Memory Timing Value 2 Register, MPMCNANDTiming2* on page 3-39
MPMCNAND Status	`0x328`	Read	6	-	`0x00`	See *NAND Status Information Register, MPMCNANDStatus* on page 3-41
MPMCAHB Control0	`0x400`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status0	`0x404`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut0	`0x408`	Read/write	10	`0x0`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control1	`0x420`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status1	`0x424`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut1	`0x428`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control2	`0x440`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCAHB Status2	`0x444`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut2	`0x448`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control3	`0x460`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status3	`0x464`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut3	`0x468`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control4	`0x480`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status4	`0x484`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut4	`0x488`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control5	`0x4A0`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status5	`0x4A4`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut5	`0x4A8`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control6	`0x4C0`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status6	`0x4C4`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut6	`0x4C8`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control7	`0x4E0`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCAHB Status7	`0x4E4`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut7	`0x4E8`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control8	`0x500`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status8	`0x504`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut8	`0x508`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCAHB Control9	`0x520`	Read/write	1	`0x0`	-	See *AHB Control Registers 0-9, MPMCAHBControl0-9* on page 3-42
MPMCAHB Status9	`0x524`	Read/write	1	`0x0`	-	See *AHB Status Registers 0-9, MPMCAHBStatus0-9* on page 3-42
MPMCAHB TimeOut9	`0x528`	Read/write	10	`0x000`	-	See *AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9* on page 3-43
MPMCPeriph ID4	`0xFD0`	Read	8	`0x9`	`0x9`	See *Additional Peripheral Identification Registers, MPMCPeriphID4-7* on page 3-43
MPMCPeriph ID5	`0xFD4`	Read	8	`0x0`	`0x0`	Reserved for peripheral identification register
MPMCPeriph ID6	`0xFD8`	Read	8	`0x0`	`0x0`	Reserved for peripheral identification register
MPMCPeriph ID7	`0xFDC`	Read	8	`0x0`	`0x0`	Reserved for peripheral identification register
MPMCPeriph ID0	`0xFE0`	Read	8	`0x76`	`0x76`	See *Peripheral Identification Registers, MPMCPeriphID0-3* on page 3-45
MPMCPeriph ID1	`0xFE4`	Read	8	`0x11`	`0x11`	See *Peripheral Identification Registers, MPMCPeriphID0-3* on page 3-45

Table 3-1 PrimeCell MPMC register summary (continued)

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCPeriph ID2	`0xFE8`	Read	8	`0x04`	`0x04`[b]	See *Peripheral Identification Registers, MPMCPeriphID0-3* on page 3-45
MPMCPeriph ID3	`0xFEC`	Read	8	`0x99`	`0x99`	See *Peripheral Identification Registers, MPMCPeriphID0-3* on page 3-45
MPMCPCell ID0	`0xFF0`	Read	8	`0x0D`	`0x0D`	See *PrimeCell Identification Registers 0-3, MPMCPCellID0-3* on page 3-48
MPMCPCell ID1	`0xFF4`	Read	8	`0xF0`	`0xF0`	See *PrimeCell Identification Registers 0-3, MPMCPCellID0-3* on page 3-48
MPMCPCell ID2	`0xFF8`	Read	8	`0x05`	`0x05`	See *PrimeCell Identification Registers 0-3, MPMCPCellID0-3* on page 3-48
MPMCPCell ID3	`0xFFC`	Read	8	`0xB1`	`0xB1`	See *PrimeCell Identification Registers 0-3, MPMCPCellID0-3* on page 3-48

a. Tie-off dependent.
b. Revision dependent.

3.2 Register descriptions

This section describes the PrimeCell MPMC registers.

3.2.1 Control Register, MPMCControl

The MPMCControl Register is a two-bit, read/write register that controls the MPMC operation. The register fields can only be altered in idle state, when no external memory accesses are being performed. This register is accessed with zero wait states. Table 3-2 shows the bit assignments for the MPMCControl Register.

Table 3-2 MPMCControl Register bit assignments

Bits	Name	Description
[31:3]	Reserved	Reserved, read undefined, do not modify.
[2]	Low-power mode (L)	Indicates normal or low-power mode: 0 = normal mode (reset value on **nPOR** and **HRESETn**) 1 = low-power mode. Entering low-power mode reduces MPMC power consumption. Dynamic memory is refreshed as necessary. The MPMC returns to normal functional mode by clearing the low-power mode bit (L), or by AHB, or power-on reset. You must only modify this bit when the MPMC is in idle state.[a][b]
[1]	Reserved	Reserved, read undefined, do not modify.
[0]	MP Enable (E)	Indicates if the PrimeCell MPMC is enabled or disabled: 0 = disabled 1 = enabled (reset value on **nPOR** and **HRESETn**). Disabling the MPMC reduces power consumption. When the MPMC is disabled the memory is not refreshed. The MPMC is enabled by setting the enable bit, or by system, or power-on reset. The MPMC produces an ERROR response (on **HRESP**) to external memory accesses when this bit is LOW. You must only modify this bit when the MPMC is in idle state.[a][b]

a. The external memory cannot be accessed in low-power and/or disable states. If a memory access is performed an error response is generated.

b. The MPMC AHB register programming port can be accessed normally. The PrimeCell MPMC registers can be programmed in low-power and disabled state.

3.2.2 Status Register, MPMCStatus

The four-bit read-only MPMCStatus Register provides PrimeCell MPMC status information. This register is accessed with zero wait states. Table 3-3 shows the bit assignments for the MPMCStatus Register.

Table 3-3 MPMCStatus Register bit assignments

Bits	Name	Description
[31:5]	Reserved	Reserved, read undefined.
[4]	DMC Busy (DYB)	This bit is used to ensure that the MPMC enters the self-refresh mode cleanly: 0 = dynamic memory controller is idle (reset value on **nPOR** and **HRESETn**) 1 = dynamic memory controller is busy performing memory transactions.
[3]	SMC Busy (STB)	This bit is used to ensure that the MPMC enters the low-power or disabled mode cleanly: 0 = static memory controller is idle (reset value on **nPOR** and **HRESETn**) 1 = static memory controller is busy performing memory transactions.
[2]	Self-refresh acknowledge, **SREFACK** (SA)	This bit indicates the operating mode of the dynamic memory controller: 0 = normal mode 1 = self-refresh mode (reset value on **nPOR**).
[1]	Reserved	Reserved, read undefined.
[0]	Busy (B)	This bit is used to ensure that the MPMC enters the low-power or disabled mode cleanly. This is a combination of the status of the static, dynamic, and NAND flash memory interfaces. It shows when the NAND flash interface is busy, but not whether a data transfer is currently in progress. The MPMCNANDStatus Register NDTX bit must be checked before entering low-power or disabled mode to ensure that a NAND flash access has ended: 0 = MPMC is idle (reset value on **nPOR** and **HRESETn**) 1 = MPMC is busy performing memory transactions.

3.2.3 Configuration Register, MPMCConfig

The one-bit, read/write, MPMCConfig Register configures the operation of the MPMC. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the

PrimeCell MPMC is idle and then entering low-power or disabled mode. This register is accessed with one wait state. Table 3-4 shows the bit assignments for the MPMCConfig Register.

Table 3-4 MPMCConfig Register bit assignments

Bits	Name	Description
[31:1]	Reserved	Reserved, read undefined, do not modify.
[0]	Endian mode (N)	Endian mode: 0 = little-endian mode (reset value on **nPOR**) 1 = big-endian mode. The value of the endian bit on **nPOR** is determined by the **MPMCBIGENDIAN** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.[a][b]

a. The value of the **MPMCBIGENDIAN** signal is not reflected in this field. This field reflects the last value that was written into it.

b. Setting little-endian mode in this register has no effect on 64-bit ports which are hardwired as mixed-endian by tying their **MPMCBSTRBENx** signal HIGH. Setting big-endian in this register forces all ports to big-endian mode, irrespective of the state of the **MPMCBSTBRENx** signal.

3.2.4 Dynamic Memory Control Register, MPMCDynamicControl

The fourteen-bit, read/write, MPMCDynamicControl Register is used to control dynamic memory operation. The control bits can be altered during normal operation. This register is accessed with zero wait states. Table 3-5 shows the bit assignments for the MPMCDynamicControl Register.

Table 3-5 MPMCDynamicControl Register bit assignments

Bits	Name	Description
[31:16]	Reserved	Reserved, read undefined, do not modify.
[15]	SyncFlash reset/power down voltage signal, **MPMCRPVHHOUT**	0 = normal voltage (reset value on **nPOR**) 1 = set **MPMCRPOUT** high voltage. This output can be used externally in conjunction with the **nMPMCRPOUT** signal to indicate a high output voltage, see Table 3-6 on page 3-16.
[14]	SyncFlash reset/power down signal, **nMPMCRPOUT** (nRP)	0 = **nMPMCRPOUT** signal LOW (reset value on **nPOR**) 1 = set **nMPMCRPOUT** signal HIGH.
[13]	Low-power SDRAM deep-sleep mode (DP)	0 = normal operation (reset value on **nPOR**) 1 = enter deep power down mode.
[12]	Reserved	Reserved, read undefined, do not modify.

Table 3-5 MPMCDynamicControl Register bit assignments (continued)

Bits	Name	Description
[11]	DLL enable (DE)	Enable DLL hand shaking (**MPMCDLLCALIBREQ**) during auto-refresh cycles: 0 = DLL hand shaking disabled (reset value on **nPOR**) 1 = DLL hand shaking enabled.
[10]	DLL calibrate (DC)	Software control of the DLL hand shaking (**MPMCDLLCALIBREQ**) for initial DLL calibration: 0 = DLL hand shaking disabled (reset value on **nPOR**) 1 = DLL hand shaking enabled.
[9]	DLL status (DS)	Indicates the value of the **MPMCDLLCALIBACK** signal for software control of DLL hand shaking: 0 = DLL calibrating 1 = DLL calibration completed.
[8:7]	SDRAM initialization (I)	00 = issue SDRAM NORMAL operation command (reset value on **nPOR**) 01 = issue SDRAM MODE command 10 = issue SDRAM PALL (precharge all) command 11 = issue SDRAM NOP (no operation) command).
[6]	Reserved	Reserved, read undefined, do not modify.
[5]	Memory clock control (MCC)	0 = **MPMCCLKOUT** enabled (reset value on **nPOR**) 1 = **MPMCCLKOUT** disabled.[a]
[4]	Inverted memory clock control (IMCC)	0 = **nMPMCCLKOUT** enabled (reset value on **nPOR**) 1 = **nMPMCCLKOUT** disabled.[b]
[3]	Self-refresh clock control (SRMCC)	0 = **MPMCCLKOUT** and **nMPMCCLKOUT** run continuously (reset value on **nPOR**). When clock control is LOW the output clocks **MPMCCLKOUT** and **nMPMCCLKOUT** run during self-refresh mode. 1 = **MPMCCLKOUT** and **nMPMCCLKOUT** stop during self-refresh mode.[c]
[2]	Self-refresh request, **MPMCSREFREQ** (SR)	0 = normal mode 1 = enter self-refresh mode (reset value on **nPOR**). By writing 1 to this bit self-refresh mode can be entered under software control. Writing 0 to this bit returns the MPMC to normal mode. The self-refresh acknowledge bit in the MPMCStatus Register must be polled to determine the current operating mode of the MPMC.
[1]	Dynamic memory clock control (CS)	0 = **MPMCCLKOUT** stops when all SDRAMS are idle and during self-refresh mode1 = **MPMCCLKOUT** runs continuously (reset value on **nPOR**). Programming the dynamic memory clock enable HIGH (CE=1) and dynamic memory clock control LOW (CS =0) results in unpredictable behavior.[d]
[0]	Dynamic memory clock enable (CE)	0 = clock enable of idle devices are deasserted to save power (reset value on **nPOR**) 1 = all clock enables are driven HIGH continuously.[e]

a. Disabling **MPMCCLKOUT** can be performed if there are no SDRAM memory transactions. When enabled this field can be used in conjunction with the dynamic memory clock control (CS) field.
b. Disabling **nMPMCCLKOUT** can be performed if there are no DDR-SDRAM memory transactions that require a differential clock. When enabled this field can be used in conjunction with the dynamic memory clock control (CS) field.
c. This functionality is supported by SDR-SDRAM and DDR-SDRAM.
d. This functionality is only supported by SDR-SDRAM.
e. Clock enable must be HIGH during SDRAM initialization.

Table 3-6 shows the output voltage settings for different combinations of bits [15:14].

A block external to the MPMC can use the values of the **nMPMCRPOUT** and **MPMCRPVHHOUT** signals to generate the required voltage settings for the Micron SyncFlash reset signal. Some systems do not have an 8V output, or do not require the functionality to raise **nRP** to 8V. In these cases **nMPMCRPVHHOUT** can be ignored.

Table 3-6 Output voltage settings

nMPMCRPOUT	MPMCRPVHHOUT	Output
0	0	0V
0	1	0V
1	0	3V
1	1	8V

3.2.5 Dynamic Memory Refresh Timer Register, MPMCDynamicRefresh

The 11-bit, read/write, MPMCDynamicRefresh Register configures dynamic memory operation. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. However, these control bits can, if necessary be altered during normal operation. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-7 shows the bit assignments for the MPMCDynamicRefresh Register.

Table 3-7 MPMCDynamicRefresh Register bit assignments

Bits	Name	Description
[31:11]	Reserved	Reserved, read undefined, do not modify.
[10:0]	Refresh timer (REFRESH)	`0x0` = refresh disabled (reset value on **nPOR**) `0x1` 1(x16) = 16 **HCLK** cycles between SDRAM refresh cycles `0x8` 8(x16) = 128 HCLK cycles between SDRAM refresh cycles `0x1-0x7FF` n(x16) = 16n HCLK cycles between SDRAM refresh cycles.

For example, for the refresh period of 16µs, and an **HCLK** frequency of 50MHz, the following value must be programmed into this register:

$$\frac{16 \times 10^{-6} \times 50 \times 10^{6}}{16} = 50 \text{ or } \texttt{0x32}$$

Note

The refresh cycles are evenly distributed. However, there might be slight variations when the auto-refresh command is issued depending on the status of the MPMC.

3.2.6 Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig

The six-bit, read/write, MPMCDynamicReadConfig Register enables you to configure the dynamic memory read strategy. This register must only be modified during system initialization. This register is accessed with one wait state. See *Pad interface methodology* on page 10-22 for more information. Table 3-8 shows the bit assignments for the MPMCDynamicReadConfig Register.

Table 3-8 MPMCDynamicReadConfig Register bit assignments

Bits	Name	Description
[31:13]	Reserved	Reserved, read undefined, do not modify.
[12]	DDR-SDRAM read data capture polarity (DRP)	0 = data capture on the negative edge of **HCLK**. 1 = data capture on the positive edge of **HCLK**.[a]
[11:10]	Reserved	Reserved, read undefined, do not modify.

Table 3-8 MPMCDynamicReadConfig Register bit assignments (continued)

Bits	Name	Description
[9:8]	DDR-SDRAM read data strategy (DRD)	00 = clock out delayed strategy, using **MPMCCLKOUT** (command not delayed, clock out delayed). 01 = command delayed strategy, using **MPMCCLKDELAY** (command delayed, clock out not delayed). 10 = command delayed strategy plus one clock cycle, using **MPMCCLKDELAY** (command delayed, clock out not delayed). 11 = command delayed strategy plus two clock cycles, using **MPMCCLKDELAY** (command delayed, clock out not delayed).[b]
[7:5]	Reserved	Reserved, read undefined, do not modify.
[4]	SDR-SDRAM read data capture polarity (SRP)	0 = data capture on the negative edge of **HCLK**. 1 = data capture on the positive edge of **HCLK**.[c]
[3:2]	Reserved	Reserved, read undefined, do not modify.
[1:0]	SDR-SDRAM read data strategy (SRD)	00 = clock out delayed strategy, using **MPMCCLKOUT** (command not delayed, clock out delayed). 01 = command delayed strategy, using **MPMCCLKDELAY** (command delayed, clock out not delayed). 10 = command delayed strategy plus one clock cycle, using **MPMCCLKDELAY** (command delayed, clock out not delayed) 11 = command delayed strategy plus two clock cycles, using **MPMCCLKDELAY** (command delayed, clock out not delayed).[d]

a. The value of the DRP field on **nPOR** is determined by the **MPMCDYDDRPOL** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.

b. The value of the DRD field on **nPOR** is determined by the **MPMCDYDDRDLY[1:0]** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.

c. The value of the SRP field on **nPOR** is determined by the **MPMCDYSDRPOL** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.

d. The value of the SRD field on **nPOR** is determined by the **MPMCDYSDRDLY[1:0]** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.

3.2.7 Dynamic Memory Precharge Command Period Register, MPMCDynamictRP

The four-bit, read/write, MPMCDynamictRP Register enables you to program the precharge command period, t_{RP}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{RP}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-9 shows the bit assignments for the MPMCDynamictRP Register.

Table 3-9 MPMCDynamictRP Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify
[3:0]	Precharge command period (t_{RP})	`0x0`-`0xE` = n + 1 clock cycles `0xF` = 16 clock cycles (reset value on **nPOR**).

3.2.8 Dynamic Memory Active To Precharge Command Period Register, MPMCDynamictRAS

The four-bit, read/write, MPMCDynamictRAS Register enables you to program the active to precharge command period, t_{RAS}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{RAS}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-10 shows the bit assignments for the MPMCDynamictRAS Register.

Table 3-10 MPMCDynamictRAS Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify
[3:0]	Active to precharge command period (t_{RAS})	`0x0`-`0xE` = n+1 clock cycles `0xF` = 16 clock cycles (reset value on **nPOR**)

3.2.9 Dynamic Memory Self-refresh Exit Time Register, MPMCDynamictSREX

The seven-bit, read/write, MPMCDynamictSREX Register enables you to program the self-refresh exit time, t_{SREX}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{SREX}, for devices without this parameter you must use the same value as t_{XSR}. For some DDR-SDRAM data sheets this parameter is known as t_{XSNR}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-11 shows the bit assignments for the MPMCDynamictSREX Register.

Table 3-11 MPMCDynamictSREX Register bit assignments

Bits	Name	Description
[31:7]	Reserved	Reserved, read undefined, do not modify
[6:0]	Self-refresh exit time (t_{SREX})	`0x0`-`0x7F` = n+1 clock cycles `0xF` = 128 clock cycles (reset value on **nPOR**)

3.2.10 Dynamic Memory Write Recovery Time Register, MPMCDynamictWR

The four-bit, read/write, MPMCDynamictWR Register enables you to program the write recovery time, t_{WR}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{WR}, t_{DPL}, t_{RWL}, or t_{RDL}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-12 shows the bit assignments for the MPMCDynamictWR Register.

Table 3-12 MPMCDynamictWR Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify
[3:0]	Write recovery time (t_{WR})	`0x0`-`0xE` = n+1 clock cycles `0xF` = 16 clock cycles (reset value on **nPOR**)

3.2.11 Dynamic Memory Active To Active Command Period Register, MPMCDynamictRC

The five-bit, read/write, MPMCDynamictRC Register enables you to program the active to active command period, t_{RC}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{RC}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-13 shows the bit assignments for the MPMCDynamictRC Register.

Table 3-13 MPMCDynamictRC Register bit assignments

Bits	Name	Description
[31:5]	Reserved	Reserved, read undefined, do not modify
[4:0]	Active to active command period (t_{RC})	`0x0`-`0x1E` = n+1 clock cycles `0x1F` = 32 clock cycles (reset value on **nPOR**)

3.2.12 Dynamic Memory Auto-refresh Period Register, MPMCDynamictRFC

The five-bit, read/write, MPMCDynamictRFC Register enables you to program the auto-refresh period, and auto-refresh to active command period, t_{RFC}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{RFC}, or sometimes as t_{RC}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-14 shows the bit assignments for the MPMCDynamictRFC Register.

Table 3-14 MPMCDynamictRFC Register bit assignments

Bits	Name	Description
[31:5]	Reserved	Reserved, read undefined, do not modify
[4:0]	Auto-refresh period and auto-refresh to active command period (t_{RFC})	`0x0`-`0x1E` = n+1 clock cycles `0x1F` = 32 clock cycles (reset value on **nPOR**)

3.2.13 Dynamic Memory Exit Self-refresh Register, MPMCDynamictXSR

The eight-bit, read/write, MPMCDynamictXSR Register enables you to program the exit self-refresh to active command time, t_{XSR}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{XSR}, but is sometimes called t_{XSNR} in some DDR-SDRAM data sheets. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-15 shows the bit assignments for the MPMCDynamictXSR Register.

Table 3-15 MPMCDynamictXSR Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify
[7:0]	Exit self-refresh to active command time (t_{XSR})	`0x0`-`0xFE` = n+1 clock cycles `0xFF` = 256 clock cycles (reset value on **nPOR**)

3.2.14 Dynamic Memory Active Bank A To Active Bank B Time Register, MPMCDynamictRRD

The four-bit, read/write, MPMCDynamictRRD Register enables you to program the active bank A to active bank B latency, t_{RRD}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{RRD}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-16 shows the bit assignments for the MPMCDynamictRRD Register.

Table 3-16 MPMCDynamictRRD Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify
[3:0]	Active bank A to active bank B latency (t_{RRD})	`0x0-0xE` = n+1 clock cycles `0xF` = 16 clock cycles (reset value on **nPOR**)

3.2.15 Dynamic Memory Load Mode Register, MPMCDynamictMRD

The four-bit, read/write, MPMCDynamictMRD Register enables you to program the load mode register to active command time, t_{MRD}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{MRD} or t_{RSA}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-17 shows the bit assignments for the MPMCDynamictMRD Register.

Table 3-17 MPMCDynamictMRD Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify
[3:0]	Load mode register to active command time (t_{MRD})	`0x0-0xE` = n+1 clock cycles `0xF` = 16 clock cycles (reset value on **nPOR**)

3.2.16 Dynamic Memory Last Data In To Read Command Time Register, MPMCDynamictCDLR

The four-bit, read/write, MPMCDynamictCDLR Register enables you to program the last data in to read command time, t_{CDLR}. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. This value is normally found in SDRAM data sheets as t_{CDLR}. This register is accessed with one wait state.

Note

This register is used for all four dynamic memory chip selects. Therefore the worse case value for all of the chip selects must be programmed.

Table 3-18 shows the bit assignments for the MPMCDynamictCDLR Register.

Table 3-18 MPMCDynamictCDLR Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify
[3:0]	Last data in to read command time (t_{CDLR})	`0x0`-`0xE` = n+1 clock cycles `0xF` = 16 clock cycles (reset value on **nPOR**)

3.2.17 Static Memory Extended Wait Register, MPMCStaticExtendedWait

The 10-bit, read/write, MPMCStaticExtendedWait Register is used to time long static memory read and write transfers (that are longer than can be supported by the MPMCStaticWaitRd0-3 or, MPMCStaticWaitWr0-3 Registers) when the EW bit of the MPMCStaticConfig Registers is enabled. There is only one MPMCStaticExtendedWait Register, which is used by the relevant static memory chip select if the appropriate *Extended Wait* (EW) bit in the MPMCStaticConfig Register is set. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. However, if necessary, these control bits can be altered during normal operation. This register is accessed with one wait state.

Table 3-19 shows the bit assignments for the MPMCStaticExtendedWait Register.

Table 3-19 MPMCStaticExtendedWait Register bit assignments

Bits	Name	Description
[31:10]	Reserved	Reserved, read undefined, do not modify
[9:0]	External wait time out (EXTENDEDWAIT)	`0x0` = 16 clock cycles (reset value on **nPOR**) `0x1-0x3FF` = (n+1) x16 clock cycles

For example, for a static memory read or write transfer time of 16µs, and an **HCLK** frequency of 50MHz, the following value must be programmed into this register:

$$\frac{16 \times 10^{-6} \times 50 \times 10^{6}}{16} - 1 = 49$$

3.2.18 Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3

The 13-bit, read/write, MPMCDynamicConfig0-3 Registers enable you to program the configuration information for the relevant dynamic memory chip select. These registers are normally only modified during system initialization. These registers are accessed with one wait state.

Table 3-20 shows the bit assignments for the MPMCDynamicConfig0-3 Registers.

Table 3-20 MPMCDynamicConfig0-3 Register bit assignments

Bits	Name	Description
[31:21]	Reserved	Reserved, read undefined, do not modify.
[20]	Write protect (P)	0 = writes not protected (reset value on **nPOR**) 1 = write protected.
[19:16]	Reserved	Reserved, read undefined, do not modify.

Table 3-20 MPMCDynamicConfig0-3 Register bit assignments (continued)

Bits	Name	Description
[15:7]	Address mapping (AM)	`0x000` = 16Mb (2Mx8) 2 banks, row length = 11, column length = 9 (reset value for chip selects 4, 6, and 7, on **nPOR**) For `0x001-1FF`, see Table 3-21 on page 3-27. `00000000` = reset value on **nPOR**.[a] The value of the chip select 5 address mapping field on **nPOR** is determined by the **MPMCDYCS5ADRMAP[8:0]** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.
[6:3]	Reserved	Reserved, read undefined, do not modify.
[2:0]	Memory device (MD)	000 = SDR-SDRAM (reset value for chip selects 4, 6, and 7 on **nPOR**)001 = SDR Micron SyncFlash010 = low-power SDR-SDRAM011 = SDR Micron V-SyncFlash100 = DDR-SDRAM101 = DDR Micron SyncFlash110 = low-power DDR-SDRAM111 = DDR Micron V-SyncFlash. The value of the chip select 5 memory device field on **nPOR** is determined by the **MPMCDYCS5DEVICE[2:0]** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.

a. The SDRAM column and row width and number of banks are computed automatically from the address mapping.

Note

DDR-SDRAM device chip selects must only have a 16 or 32-bit wide data bus.

Table 3-21 on page 3-27 lists all combinations of external bus width, mapping strategy, device capacity, and device width which are supported by MPMC. Address mappings that are not shown in Table 3-21 on page 3-27 are reserved. Bits [15:7] are allocated as follows:

Bit[15:14] External bus width (chip select width) which can be 16-bit (00), 32-bit (01), or 64-bit (10).

Bit[13:12] Mapping strategy, which can be high performance (row, bank, column (RBC) (00)), low power (bank, row, column (BRC) (01)), or V-SyncFlash (bank, segment, row, segment, column (BSRSC) (10)).

Bit[11:9] Device capacity, which can be 16Mb (000), 64Mb (001), 128Mb (010), 256Mb (011), or 512Mb (100).

Bits [8:7] Device width, which can be 8-bit (00), 16-bit (01), or 32-bit (10).

Table 3-21 Address mapping

[15:14]	[13:12]	[11:9]	[8:7]	Description
16-bit external bus high-performance SDRAM address mapping (Row, Bank, Column)				
00	00	000	00	16Mb (2Mx8), 2 banks, row length = 11, column length = 9
00	00	000	01	16Mb (1Mx16), 2 banks, row length = 11, column length = 8
00	00	001	00	64Mb (8Mx8), 4 banks, row length = 12, column length = 9
00	00	001	01	64Mb (4Mx16), 4 banks, row length = 12, column length = 8
00	00	010	00	128Mb (16Mx8), 4 banks, row length = 12, column length = 10
00	00	010	01	128Mb (8Mx16), 4 banks, row length = 12, column length = 9
00	00	011	00	256Mb (32Mx8), 4 banks, row length = 13, column length = 10
00	00	011	01	256Mb (16Mx16), 4 banks, row length = 13, column length = 9
00	00	100	00	512Mb (64Mx8), 4 banks, row length = 13, column length = 11
00	00	100	01	512Mb (32Mx16), 4 banks, row length = 13, column length = 10
16-bit external bus low-power SDRAM address mapping (Bank, Row, Column)				
00	01	000	00	16Mb (2Mx8), 2 banks, row length = 11, column length = 9
00	01	000	01	16Mb (1Mx16), 2 banks, row length = 11, column length = 8
00	01	001	00	64Mb (8Mx8), 4 banks, row length = 12, column length = 9
00	01	001	01	64Mb (4Mx16), 4 banks, row length = 12, column length = 8
00	01	010	00	128Mb (16Mx8), 4 banks, row length = 12, column length = 10
00	01	010	01	128Mb (8Mx16), 4 banks, row length = 12, column length = 9
00	01	011	00	256Mb (32Mx8), 4 banks, row length = 13, column length = 10
00	01	011	01	256Mb (16Mx16), 4 banks, row length = 13, column length = 9
00	01	100	00	512Mb (64Mx8), 4 banks, row length = 13, column length = 11
00	01	100	01	512Mb (32Mx16), 4 banks, row length = 13, column length = 10
32-bit external bus high-performance SDRAM address mapping (Row, Bank, Column)				
01	00	000	00	16Mb (2Mx8), 2 banks, row length = 11, column length = 9

Table 3-21 Address mapping (continued)

[15:14]	[13:12]	[11:9]	[8:7]	Description
01	00	000	01	16Mb (1Mx16), 2 banks, row length = 11, column length = 8
01	00	001	00	64Mb (8Mx8), 4 banks, row length = 12, column length = 9
01	00	001	01	64Mb (4Mx16), 4 banks, row length = 12, column length = 8
01	00	001	10	64Mb (2Mx32), 4 banks, row length = 11, column length = 8
01	00	010	00	128Mb (16Mx8), 4 banks, row length = 12, column length = 10
01	00	010	01	128Mb (8Mx16), 4 banks, row length = 12, column length = 9
01	00	010	10	128Mb (4Mx32), 4 banks, row length = 12, column length = 8
01	00	011	00	256Mb (32Mx8), 4 banks, row length = 13, column length = 10
01	00	011	01	256Mb (16Mx16), 4 banks, row length = 13, column length = 9
01	00	011	10	256Mb (8Mx32), 4 banks, row length = 13, column length = 8
01	00	100	00	512Mb (64Mx8), 4 banks, row length = 13, column length = 11
01	00	100	01	512Mb (32Mx16), 4 banks, row length = 13, column length = 10
32-bit external bus low-power SDRAM address mapping (Bank, Row, Column)				
01	01	000	00	16Mb (2Mx8), 2 banks, row length = 11, column length = 9
01	01	000	01	16Mb (1Mx16), 2 banks, row length = 11, column length = 8
01	01	001	00	64Mb (8Mx8), 4 banks, row length = 12, column length = 9
01	01	001	01	64Mb (4Mx16), 4 banks, row length = 12, column length = 8
01	01	001	10	64Mb (2Mx32), 4 banks, row length = 11, column length = 8
01	01	010	00	128Mb (16Mx8), 4 banks, row length = 12, column length = 10
01	01	010	01	128Mb (8Mx16), 4 banks, row length = 12, column length = 9
01	01	010	10	128Mb (4Mx32), 4 banks, row length = 12, column length = 8
01	01	011	00	256Mb (32Mx8), 4 banks, row length = 13, column length = 10
01	01	011	01	256Mb (16Mx16), 4 banks, row length = 13, column length = 9
01	01	011	10	256Mb (8Mx32), 4 banks, row length = 13, column length = 8
01	01	100	00	512Mb (64Mx8), 4 banks, row length = 13, column length = 11

Table 3-21 Address mapping (continued)

[15:14]	[13:12]	[11:9]	[8:7]	Description
01	01	100	01	512Mb (32Mx16), 4 banks, row length = 13, column length = 10
64-bit external bus high-performance SDRAM address mapping (Bank, Row, Column)				
10	00	000	01	16Mb (1Mx16), 2 banks, row length = 11, column length = 8
10	00	001	01	64Mb (4Mx16), 4 banks, row length = 12, column length = 8
10	00	001	10	64Mb (2Mx32), 4 banks, row length = 11, column length = 8
10	00	010	01	128Mb (8Mx16), 4 banks, row length = 12, column length = 9
10	00	010	10	128Mb (4Mx32), 4 banks, row length = 12, column length = 8
10	00	011	01	256Mb (16Mx16), 4 banks, row length = 13, column length = 9
10	00	011	10	256Mb (8Mx32), 4 banks, row length = 13, column length = 8
10	00	100	01	512Mb (32Mx16), 4 banks, row length = 13, column length = 10
64-bit external bus low-power SDRAM address mapping (Bank, Row, Column)				
10	01	000	01	16Mb (1Mx16), 2 banks, row length = 11, column length = 8
10	01	001	01	64Mb (4Mx16), 4 banks, row length = 12, column length = 8
10	01	001	10	64Mb (2Mx32), 4 banks, row length = 11, column length = 8
10	01	010	01	128Mb (8Mx16), 4 banks, row length = 12, column length = 9
10	01	010	10	128Mb (4Mx32), 4 banks, row length = 12, column length = 8
10	01	011	01	256Mb (16Mx16), 4 banks, row length = 13, column length = 9
10	01	011	10	256Mb (8Mx32), 4 banks, row length = 13, column length = 8
10	01	100	01	512Mb (32Mx16), 4 banks, row length = 13, column length = 10

A chip select can be connected to a single memory device. In this case the chip select data bus width is the same as the device width. Alternatively the chip select can be connected to several external devices. In this case the chip select data bus width is the sum of the memory device data bus widths.

For example, for a chip select connected to:

- a 64-bit wide memory device, choose a 64-bit wide address mapping
- a 32-bit wide memory device, choose a 32-bit wide address mapping
- a 16-bit wide memory device, choose a 16-bit wide address mapping

- four x 16-bit wide memory devices, choose a 64-bit wide address mapping
- four x 8-bit wide memory devices, choose a 32-bit wide address mapping
- two x 8-bit wide memory devices, choose a 16-bit wide address mapping.

3.2.19 Dynamic Memory RAS and CAS Delay Registers 0-3, MPMCDynamicRasCas0-3

The eight-bit, read/write, MPMCDynamicRasCas0-3 Registers enable you to program the RAS and CAS latencies for the relevant dynamic memory. These registers must only be modified during system initialization. This value is normally found in SDRAM data sheets as t_{RAS}. These registers are accessed with one wait state.

Table 3-22 shows the bit assignments for the MPMCDynamicRasCas0-3 Registers.

Table 3-22 MPMCDynamicRasCas0-3 Register bit assignments

Bits	Name	Description
[31:11]	Reserved	Reserved, read undefined, do not modify.
[10:7]	CAS latency (CAS)	`0x0` = reserved`0x1` = 0.5 **HCLK** cycles`0x2` = 1 **HCLK** cycle`0x3` = 1.5 **HCLK** cycles`0x4`-`0xD` = n/2 **HCLK** cycles (2-6.5)`0xE` = 7 **HCLK** cycles`0xF` = 7.5 **HCLK** cycles (reset value for chip selects 4, 6, and 7, on **nPOR**). The value of the upper three bits of the CAS field for dynamic bank one (chip select 5) on **nPOR** is determined by the **MPMCDYCS5CASDLY[3:0]** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.
[6:4]	Reserved	Reserved, read undefined, do not modify.
[3:0]	RAS latency (active to read or write delay) (RAS)	`0x0` = reserved`0x1`-`0xE` = n **HCLK** cycles (reset value on **nPOR** = `0x3`)`0xF` = 15 **HCLK** cycles.

3.2.20 Static Memory Configuration Registers 0-3, MPMCStaticConfig0-3

The eight-bit, read/write, MPMCStaticConfig0-3 Registers are used to configure the static memory configuration. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle and then entering low-power or disabled mode. These registers are accessed with one wait state.

Table 3-23 shows the bit assignments for the MPMCStaticConfig0-3 Registers.

Table 3-23 MPMCStaticConfig0-3 Register bit assignments

Bits	Name	Description
[31:21]	Reserved	Reserved, read undefined, do not modify.
[20]	Write protect (P)	0 = writes not protected (reset value on **nPOR**) 1 = write protected.
[19:9]	Reserved	Reserved, read undefined, do not modify.
[8]	Extended wait (EW)	0 = Extended wait disabled (reset value on **nPOR**) 1 = Extended wait enabled. Extended wait (EW) uses the MPMCStaticExtendedWait Register to time both the read and write transfers rather than the MPMCStaticWaitRd and MPMCStaticWaitWr Registers. This enables much longer transactions.[a]
[7]	Byte lane state (PB)	0 = For reads all the bits in **nMPMCBLSOUT[3:0]** are HIGH. For writes the respective active bits in **nMPMCBLSOUT[3:0]** are LOW (reset value for chip select 0, 2, and 3 on **nPOR**). 1 = For reads the respective active bits in **nMPMCBLSOUT[3:0]** are LOW. For writes the respective active bits in **nMPMCBLSOUT[3:0]** are LOW. The value of the chip select 1 byte lane state field on **nPOR** is determined by the **MPMCSTCS1PB** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.[b]
[6]	Chip select polarity (PC)	0 = active LOW chip select 1 = active HIGH chip select. The value of the chip select polarity on **nPOR** is determined by the relevant **MPMCSTCSxPOL** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.[c]
[5]	Reserved	Reserved, read undefined, do not modify.
[4]	NAND flash (ND)	0 = standard SRAM/ROM/flash device (reset value on **nPOR**) 1 = NAND flash device. Sets the type of memory device found on the chip select, either a standard SRAM interface device (SRAM, ROM, flash), or a NAND flash device which has a different interface. If a NAND device is selected, the NAND control registers must be used to control memory accesses.

Table 3-23 MPMCStaticConfig0-3 Register bit assignments (continued)

Bits	Name	Description
[3]	Page mode (PM)	0 = disabled (reset value on **nPOR**) 1 = asynchronous page mode enabled (page length four). In page mode the MPMC can burst up to four external accesses. Therefore devices with asynchronous page mode burst four or higher devices are supported. Asynchronous page mode burst two devices are not supported and must be accessed normally.
[2]	Reserved	Reserved, read undefined, do not modify.
[1:0]	Memory width (MW)	00 = 8-bit (reset value for chip select 0, 2, and 3 on **nPOR**). 01 = 16-bit 10 = 32-bit 11 = reserved. The value of the chip select 1 memory width field on **nPOR** is determined by the **MPMCSTCS1MW[1:0]** signal. This value can be overridden by software. This field is unaffected by **HRESETn**.

a. Extended wait and page mode cannot be selected simultaneously.

b. For chip select 1 the value of the **MPMCSTCS1PB** signal is reflected in this field. When programmed this register reflects the last value that is written into it.

c. The value of the relevant **MPMCSTCSxPOL** signal is reflected in this field. When programmed this register reflects the last value that is written into it.

3.2.21 Static Memory Write Enable Delay Registers 0-3, MPMCStaticWaitWen0-3

The four-bit, read/write, MPMCStaticWaitWen0-3 Registers enable you to program the delay from the chip select to the write enable. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. These registers are accessed with one wait state.

Table 3-24 shows the bit assignments for the MPMCStaticWaitWen0-3 Registers.

Table 3-24 MPMCStaticWaitWen0-3 Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify.
[3:0]	Wait write enable (WAITWEN)	Delay from chip select assertion to write enable: `0x0` = one **HCLK** cycle delay between assertion of chip select and write enable (reset value on **nPOR**) `0x1`-`0xF` = (n + 1) **HCLK** cycle delay[a].

a. The delay is (WAITWEN +1) x t_{HCLK}.

3.2.22 Static Memory Output Enable Delay Registers 0-3, MPMCStaticWaitOen0-3

The four-bit, read/write, MPMCStaticWaitOen0-3 Registers enable you to program the delay from the chip select or address change, whichever is later, to the output enable. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. These registers are accessed with one wait state.

Table 3-25 shows the bit assignments for the MPMCStaticWaitOen0-3 Registers.

Table 3-25 MPMCStaticWaitOen0-3 Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify.
[3:0]	Wait output enable (WAITOEN)	Delay from chip select assertion to output enable: `0x0` = No delay (reset value on **nPOR**) `0x1`-`0xF` = n cycle delay[a].

a. The delay is WAITOEN x t_{HCLK}.

3.2.23 Static Memory Read Delay Registers 0-3, MPMCStaticWaitRd0-3

The five-bit, read/write, MPMCStaticWaitRd0-3 Registers enable you to program the delay from the chip select to the read access. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. These registers are not used if the Extended Wait (EW) bit is enabled in the MPMCStaticConfig0-3 Registers. These registers are accessed with one wait state.

Table 3-26 shows the bit assignments for the MPMCStaticWaitRd0-3 Registers.

Table 3-26 MPMCStaticWaitRd0-3 Register bit assignments

Bits	Name	Description
[31:5]	Reserved	Reserved, read undefined, do not modify.
[4:0]	Nonpage mode read wait states or asynchronous page mode read first access wait state (WAITRD)	Nonpage mode read or asynchronous page mode read, first read only: `0x00-0x1E` = (n + 1) **HCLK** cycles for read accesses[a] `0x1F` = 32 **HCLK** cycles for read accesses (reset value on **nPOR**).

a. For nonsequential reads, the wait state time is (WAITRD + 1) x $t_{\mathbf{HCLK}}$.

3.2.24 Static Memory Page Mode Read Delay Registers 0-3, MPMCStaticWaitPage0-3

The five-bit, read/write, MPMCStaticWaitPage0-3 Registers enable you to program the delay for asynchronous page mode sequential accesses. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. These registers are accessed with one wait state.

Table 3-27 shows the bit assignments for the MPMCStaticWaitPage0-3 Registers.

Table 3-27 MPMCStaticWaitPage0-3 Register bit assignments

Bits	Name	Description
[31:5]	Reserved	Reserved, read undefined, do not modify.
[4:0]	Asynchronous page mode read after the first read wait states (WAITPAGE)	Number of wait states for asynchronous page mode read accesses after the first read: `0x00-0x1E` = (n+ 1) **HCLK** cycle read access time[a] `0x1F` = 32 **HCLK** cycle read access time (reset value on **nPOR**).

a. For asynchronous page mode read for sequential reads, the wait state time for page mode accesses after the first read is (WAITPAGE + 1) x $t_{\mathbf{HCLK}}$

3.2.25 Static Memory Write Delay Registers 0-3, MPMCStaticWaitWr0-3

The five-bit, read/write, MPMCStaticWaitWr0-3 Registers enable you to program the delay from the chip select to the write access. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. These registers are not used if the Extended Wait (EW) bit is enabled in the MPMCStaticConfig0-3 Registers. These registers are accessed with one wait state.

Table 3-28 shows the bit assignments for the MPMCStaticWaitWr0-3 Registers.

Table 3-28 MPMCStaticWaitWr0-3 Register bit assignments

Bits	Name	Description
[31:5]	Reserved	Reserved, read undefined, do not modify.
[4:0]	Write wait states (WAITWR)	SRAM wait state time for write accesses after the first read:`0x00-0x1E` = (n + 2) **HCLK** cycle write access time[a]`0x1F` = 33 **HCLK** cycle write access time (reset value on **nPOR**).

a. The wait state time for write accesses after the first read is WAITWR x $t_{\mathbf{HCLK}}$

3.2.26 Static Memory Turn Round Delay Registers 0-3, MPMCStaticWaitTurn0-3

The four-bit, read/write, MPMCStaticWaitTurn0-3 Registers enable you to program the number of bus turnaround cycles. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. These registers are accessed with one wait state.

Table 3-29 shows the bit assignments for the MPMCStaticWaitTurn0-3 Registers.

Table 3-29 MPMCStaticWaitTurn0-3 Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify.
[3:0]	Bus turnaround cycles (WAITTURN)	`0x0-0xE` = (n + 1) **HCLK** turnaround cycles[a]`0xF` = 16 **HCLK** turnaround cycles (reset value on **nPOR**).

a. Bus turnaround time is (WAITTURN + 1) x $t_{\mathbf{HCLK}}$

To prevent bus contention on the external memory data bus, the WAITTURN field controls the number of bus turnaround cycles added between static memory read and write accesses.

The WAITTURN field also controls the number of turnaround cycles between static memory and dynamic memory accesses.

3.2.27 NAND Memory Control Vector Register, MPMCNANDControl

The 28-bit, read/write, MPMCNANDControl Register enables you to program the NAND flash command vector values, and to set other control values for the NAND flash transfer. The register fields can be altered during normal operation, but it is recommended that they are only modified before a NAND access is initiated. This register is accessed with zero wait states.

Table 3-30 shows the bit assignments for the MPMCNANDControl Register.

Table 3-30 MPMCNANDControl Register bit assignments

Bits	Name	Description
[31]	Transfer type (NDTXRW)	0 = read transfer (reset value on **nPOR**) 1 = write transfer. Defines the type of transfer that is performed, either a read, or a write. A transfer that performs a data read must be programmed as a read. A transfer that performs a data write or has no data phase, such as a block erase, must be programmed as a write.
[30:27]	Reserved	Reserved, read undefined, do not modify.
[26]	Short read timing (NDSHORTRD)	0 = standard read transfer (reset value on **nPOR**) 1 = short read transfer. Identifies when a read transfer is performed without a check for the ready/busy output status. A standard read transfer checks that the device ready/busy output indicates the device is ready before performing the data phase of the transfer. For a short read transfer, the t_{CLR} timing value is used to time the read transfer delay between the deassertion of the command latch enable, and the assertion of the read enable. Commands such as random page read require this setting because the ready/busy output is not used to control the delay to the data phase. This bit does not have to be set for a Status Read, because this transfer is automatically detected and is performed differently because it does not have an address phase.
[25]	ID read command (NDIDRD)	0 = command is not ID read (reset value on **nPOR**) 1 = read ID command is performed. Identifies that the transfer being performed is reading the device ID, enabling the correct timing to be applied during the transfer (t_{AR1}).
[24]	Second command phase (NDCMDPH2)	0 = no second command phase (reset value on **nPOR**) 1 = second command phase is performed. Controls whether a second command phase is performed at the end of the transfer. Commands such as page program might require a second command phase in the transfer.

Table 3-30 MPMCNANDControl Register bit assignments (continued)

Bits	Name	Description
[23]	Data phase (NDDATAPH)	0 = no data phase (reset value on **nPOR**) 1 = data phase is performed. Controls whether a data phase is performed during the transfer. Commands such as block erase might not require a data phase in the transfer.
[22:20]	Address phase (NDADDRPH)	`0x0` = no address phase (reset value on **nPOR**) `0x1-0x5` = n address vectors in address phase `0x6-0x7` = reserved. Controls whether an address phase is performed during the transfer, and the number of address vectors in the address phase. Commands such as a status read might not require an address phase in the transfer.
[19:16]	Chip select for transfer (NDCS)	0001 = chip select 0 (reset value on **nPOR**) 0010 = chip select 1 0100 = chip select 2 1000 = chip select 3. Sets the chip select to use for the NAND access. Only one bit of the register can be valid at a time.
[15:8]	Second command vector (NDCMDV2)	`0x00-0xFF` = NAND flash second command vector (reset value on **nPOR** is `0x00`). See the applicable specification for the NAND device for the commands supported. This field is programmed for each NAND access that is performed. The second command vector is only used if the NDCMDPH2 bit of the MPMCNANDControl Register is set HIGH.
[7:0]	First command vector (NDCMDV1)	`0x00-0xFF` = NAND flash first command vector (reset value on **nPOR** is `0x00`). See the applicable specification for the NAND device for the commands supported. This field is programmed for each NAND access that is performed.

3.2.28 NAND Memory Address Vectors 1-4 Register, MPMCNANDAddress1

The 32-bit, read/write, MPMCNANDAddress1 Register enables you to program the NAND flash address phase values, for vectors 1-4. The register fields can be altered during normal operation, but it is recommended that they are only modified before a NAND access is initiated. This register is accessed with zero wait states.

Table 3-31 shows the bit assignments for the MPMCNANDAddress1 Register.

Table 3-31 MPMCNANDAddress1 Register bit assignments

Bits	Name	Description
[31:24]	Fourth address vector (NDADDRV4)	`0x00-0xFF` = NAND flash fourth address vector (reset value on **nPOR** is `0x0`). This value is only used if the NDADDRPH bits of the MPMCNANDControl Register are set to four or greater.
[23:16]	Third address vector (NDADDRV3)	`0x00-0xFF` = NAND flash third address vector (reset value on **nPOR** is `0x0`). This value is only used if the NDADDRPH bits of the MPMCNANDControl Register are set to three or greater.
[15:8]	Second address vector (NDADDRV2)	`0x00-0xFF` = NAND flash second address vector (reset value on **nPOR** is `0x0`). This value is only used if the NDADDRPH bits of the MPMCNANDControl Register are set to two or greater.
[7:0]	First address vector (NDADDRV1)	`0x00-0xFF` = NAND flash first address vector (reset value on **nPOR** is `0x0`). This value is only used if the NDADDRPH bits of the MPMCNANDControl Register are set to a nonzero value.

3.2.29 NAND Memory Address Vector 5 Register, MPMCNANDAddress2

The 8-bit, read/write, MPMCNANDAddress2 Register enables you to program the NAND flash address phase values, for vector 5. The register field can be altered during normal operation, but it is recommended that it is only modified before a NAND access is initiated. This register is accessed with zero wait states.

Table 3-32 shows the bit assignments for the MPMCNANDAddress2 Register.

Table 3-32 MPMCNANDAddress2 Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify.
[7:0]	Fifth address vector (NDADDRV5)	`0x00-0xFF` = NAND flash fifth address vector (reset value on **nPOR** is `0x0`). This value is only used if the NDADDRPH bits of the MPMCNANDControl Register are set to five.

3.2.30 NAND Memory Timing Value 1 Register, MPMCNANDTiming1

The 25-bit, read/write, MPMCNANDTiming1 Register enables you to program the first set of NAND flash timing parameters. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions.

This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. Software must poll the NDTX bit in the MPMCNANDStatus Register before programming the timing register. This register is accessed with zero wait states.

Table 3-33 shows the bit assignments for the MPMCNANDTiming1 Register.

Table 3-33 MPMCNANDTiming1Register bit assignments

Bits	Name	Description
[31:25]	Reserved	Reserved, read undefined, do not modify.
[24:20]	NAND ID read time, ALE to RE falling edge (t_{AR1})	`0x00`-`0x1E` = n **HCLK** cycles`0x1F` = 31 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{AR1} on all NAND flash devices in the system must be programmed. This parameter is used when the NDIDRD bit in the MPMCNDControl Register is set, indicating that the current transfer is an ID read.
[19:16]	NAND status read time, CLE LOW to RE LOW (t_{CLR})	`0x0`-`0xE` = n **HCLK** cycles`0xF` = 15 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{CLR} on all NAND flash devices in the system must be programmed.[a] This parameter is used when the current transfer is a status read, detected by the MPMC as a transfer without an address phase.
[15:12]	NAND data hold time from WE rising edge (t_{DH})	`0x0`-`0xE` = n **HCLK** cycles`0xF` = 15 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{DH} on all NAND flash devices in the system must be programmed.
[11:8]	NAND write enable pulse width time (t_{WP}, $t_{WC/2}$)	`0x0`-`0xE` = (n+1) **HCLK** cycles`0xF` = 16 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameters t_{WP} and half of t_{WC} on all NAND flash devices in the system must be programmed.
[7:4]	NAND control hold time, CLE, CE, and ALE from WE rising edge (t_{CLH}, t_{CH}, t_{ALH})	`0x0`-`0xE` = n **HCLK** cycles`0xF` = 15 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameters t_{CLH}, t_{CH}, and t_{ALH} on all NAND flash devices in the system must be programmed.
[3:0]	NAND control setup time, CLE, CE, and ALE to WE falling edge (t_{CLS}, t_{CS}, t_{ALS})	`0x0`-`0xE` = n **HCLK** cycles`0xF` = 15 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameters t_{CLS}, t_{CS}, and t_{ALS} on all NAND flash devices in the system must be programmed.

a. Some devices use the parameter t_{CLS} instead for status reads, but this value must still be programmed into the t_{CLR} bits even though t_{CLS} is previously defined in bits [3:0] of this register.

3.2.31 NAND Memory Timing Value 2 Register, MPMCNANDTiming2

The 18-bit, read/write, MPMCNANDTiming1 Register enables you to program the second set of NAND flash timing parameters. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the MPMC is idle, and then entering low-power or disabled mode. Software must poll the NDTX bit in the MPMCNANDStatus Register before programming the timing register. This register is accessed with zero wait states.

Table 3-34 shows the bit assignments for the MPMCNANDTiming2 Register.

Table 3-34 MPMCNANDTiming2Register bit assignments

Bits	Name	Description
[31:25]	Reserved	Reserved, read undefined, do not modify.
[24:20]	NAND CE HIGH hold time at last serial read (t_{CEH})	`0x00-0x1E` = (n+1) **HCLK** cycles`0x1F` = 32 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{CEH} on all NAND flash devices in the system must be programmed.
[19]	Reserved	Reserved, read undefined, do not modify.
[18:16]	NAND RE HIGH hold time (t_{REH})	`0x0-0x6` = (n+1) **HCLK** cycles`0x7` = 8 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{REH} on all NAND flash devices in the system must be programmed.
[15:13]	Reserved	Reserved, read undefined, do not modify.
[12:8]	NAND read cycle time (t_{RC})	`0x00-0x1E` = (n+1) **HCLK** cycles`0x1F` = 32 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{RC} on all NAND flash devices in the system must be programmed.
[7:5]	Reserved	Reserved, read undefined, do not modify.
[4:0]	NAND ready to RE falling edge (t_{RR})	`0x00-0x1E` = n **HCLK** cycles`0x1F` = 31 **HCLK** cycles (reset value on **nPOR**). The worst case value for the parameter t_{RR} on all NAND flash devices in the system must be programmed.

Note

No timing value is specified for the read access time t_{REA}, because this is calculated from the t_{RC} and t_{REH} values. For all read transfers, t_{REA}=t_{RC}-t_{REH}. Some devices specify a longer read access time for ID read transfers. For these devices, an increased t_{RC} value must be programmed for the ID read. It can then be set to the standard read value for all subsequent standard data reads. Some devices might not specify a requirement for the t_{CEH}parameter. The timing value can be programmed to zero for these devices to reduce the read access time.'

3.2.32 NAND Status Information Register, MPMCNANDStatus

The 6-bit, read-only, MPMCNANDStatus Register enables you to read various NAND status information. This register is accessed with zero wait states.

Table 3-35 shows the bit assignments for the MPMCNANDStatus Register.

Table 3-35 MPMCNANDStatus Register bit assignments

Bits	Name	Description
[31:6]	Reserved	Reserved, read undefined, do not modify.
[5]	NAND interface busy/idle (NDINTFBUSY)	0 = NAND interface is idle (reset value on **nPOR**) 1 = NAND interface is busy. This bit indicates when the NAND interface is busy or idle. It determines when it is safe for the control and address registers to be updated during a NAND access. The interface is typically idle during a device busy phase (waiting for the NAND flash internal operation to complete), or when the device is waiting for a write transfer to be performed. The interface is typically busy when a start command has been issued but the NAND interface has not been granted control of the external bus to initiate the transfer, while command and address phases are being performed, or during a data access. The control and address registers must not be updated during a control or address phase. Updating results in unpredictable behavior.
[4]	NAND transfer (NDTX)	0 = no NAND transfers are performed (reset value on **nPOR**) 1 = a NAND transfer is in progress. This bit indicates when a NAND transfer is in progress, and can be used to determine when it is possible to start a new NAND flash transfer, or cleanly enter the low-power or disabled mode in conjunction with the B bit of the MPMCStatus Register.
[3]	Chip select 3 NAND ready status (NDRDYCS3)	0 = NAND device is ready (reset value on **nPOR**) 1 = NAND device is busy. This status bit is an inverted registered version of the chip select 3 device ready/busy output.
[2]	Chip select 2 NAND ready status (NDRDYCS2)	0 = NAND device is ready (reset value on **nPOR**) 1 = NAND device is busy. This status bit is an inverted registered version of the chip select 2 device ready/busy output.
[1]	Chip select 1 NAND ready status (NDRDYCS1)	0 = NAND device is ready (reset value on **nPOR**) 1 = NAND device is busy. This status bit is an inverted registered version of the chip select 1 device ready/busy output.
[0]	Chip select 0 NAND ready status (NDRDYCS0)	0 = NAND device is ready (reset value on **nPOR**) 1 = NAND device is busy. This status bit is an inverted registered version of the chip select 0 device ready/busy output.

3.2.33 AHB Control Registers 0-9, MPMCAHBControl0-9

The MPMCAHBControl0-9 Registers are one-bit, read/write registers used to control the AHB interface operation. The register fields can be altered during normal operation.

Table 3-36 shows the bit assignments for the MPMCAHBControl0-9 Registers.

Table 3-36 MPMCAHBControl0-9 Register bit assignments

Bits	Name	Function
[31:1]	Reserved	Reserved, read undefined, do not modify.
[0]	Buffer enable (E)	0 = disable buffer (reset value on **nPOR**) 1 = enable buffer and zero wait write states.[a]

a. For 32 and 64-bit masters that use separate read and write ports, the associated AHB port buffers must be disabled to ensure coherency with the data read port. For the ARM 11 data write port, this bit must be set to 0 to ensure coherency with the data write port.

Note

The buffer is only used if:

- The buffer is enabled, MPMCAHBControl E bit is HIGH for the AHB port.
- For writes the AHB HPROT[2] bufferable bit is HIGH and the AHB HPROT[5] Outer memory Load/Store Exclusive bit is LOW for the transfer. For reads the AHB HPROT[3] cacheable bit is HIGH for the transfer and the AHB HPROT[5] Outer memory Load/Store Exclusive bit is LOW for the transfer.

3.2.34 AHB Status Registers 0-9, MPMCAHBStatus0-9

The MPMCAHBStatus0-9 Registers are single-bit, read-only registers and provide AHB interface status information.

Table 3-37 shows the bit assignments for the MPMCAHBStatus0-9 Registers.

Table 3-37 MPMCAHBStatus0-9 Register bit assignments

Bits	Name	Function
[31:2]	Reserved	Reserved, read undefined, do not modify
[1]	Buffer status (S)	0 = buffer empty (reset value on **nPOR**) 1 = buffer contains data
[0]	Reserved	Reserved, read undefined, do not modify

3.2.35 AHB Timeout Registers 0-9, MPMCAHBTimeOut0-9

The MPMCAHBTimeOut0-9 Registers are ten-bit, read or write registers that are used to ensure that each AHB port is serviced within a programmed number of cycles. When an AHB request goes active the values in the MPMCAHBTimeOut0-9 Registers are loaded into a down counter. If the transfer does not get processed by the time the TimeOut has counted down to 0 then the priority of the AHB port is increased until the request is serviced.

These registers therefore enable the transaction latency, and indirectly the bandwidth, for a particular port to be defined. The value programmed into these registers depends on the latency required for the particular port.

The register fields can be altered during normal operation.

Table 3-38 shows the bit assignments for the MPMCAHBTimeOut0-9 Registers.

Table 3-38 MPMCAHBTimeOut0-9 Register bit assignments

Bits	Name	Function
[31:10]	Reserved	Reserved, read undefined, do not modify
[9:0]	AHB time out (AHBTIMEOUT)	`0x000` = time out disabled (reset value on **nPOR**)`0x1FF` = 1-1023 **HCLK** cycles before time out is reached

3.2.36 Additional Peripheral Identification Registers, MPMCPeriphID4-7

The MPMCPeriphID4-7 Registers are four eight-bit read-only registers, that span address locations `0xFD0-0xFDC`. The registers can conceptually be treated as a single register that holds a 32-bit additional peripheral ID value.

Table 3-39 shows the bit assignments for the conceptual 32-bit MPMC Additional Peripheral ID Register.

Table 3-39 Conceptual MPMC Additional Peripheral ID Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify.
[7:0]	Configuration1	Additional peripheral configuration information

The configuration options are peripheral-specific. For MPMC, the four, eight-bit peripheral identification registers are described in the following subsections:

- *Additional Peripheral Identification Register 4, MPMCPeriphID4*
- *Additional Peripheral Identification Register 5-7, MPMCPeriphID5-7*.

Additional Peripheral Identification Register 4, MPMCPeriphID4

The MPMCPeriphID4 Register is hard-coded and the fields within the register determine the reset value. Table 3-40 shows the bit assignments for the MPMCPeriphID4 register.

Table 3-40 MPMCPeriphID4 Register bit assignments

Bits	Name	Description
[31:4]	Reserved	Reserved, read undefined, do not modify.
[3:0]	Configuration	Number of memory ports: 0000-1111 = n + 1 memory ports 1001 = 10 memory ports (value for PL176).

Additional Peripheral Identification Register 5-7, MPMCPeriphID5-7

The MPMCPeriphID5-7 Registers are reserved. Table 3-41 shows the bit assignments for the MPMCPeriphID5-7 Registers.

Table 3-41 MPMCPeriphID5-7 Register bit assignments

Bits	Name	Description
[31:0]	Reserved	Reserved, read undefined, do not modify.

3.2.37 Peripheral Identification Registers, MPMCPeriphID0-3

The MPMCPeriphID0-3 Registers are four eight-bit read-only registers, that span address locations `0xFE0`-`0xFEC`. The registers can conceptually be treated as a single register that holds a 32-bit peripheral ID value.

Table 3-42 shows the bit assignments for the conceptual 32-bit MPMC Peripheral Identification Register.

Table 3-42 Conceptual MPMC Peripheral ID Register bit assignments

Bits	Name	Description
[31:24]	Configuration	Configuration options are peripheral-specific. See *Peripheral Identification Register 3, MPMCPeriphID3* on page 3-47.
[23:20]	Revision	The peripheral revision number is revision dependent.
[19:12]	Designer	Designer ID number. This is `0x41` for ARM.
[11:0]	Part number	Identifies the peripheral. The part number for PL176 is `0x176`.

Figure 3-1 shows the correspondence between bits of the MPMCPeriphID0-3 Registers and the conceptual 32-bit MPMC Peripheral ID Register.

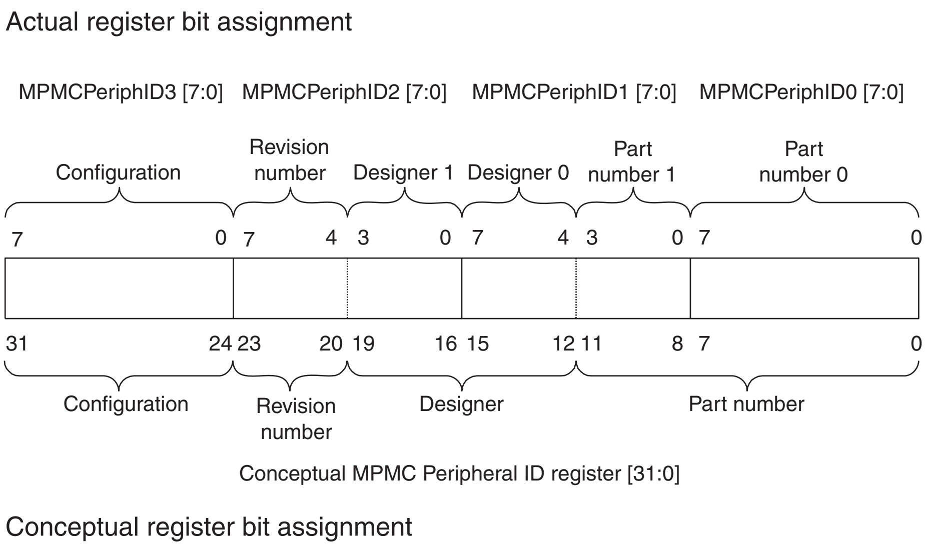


Figure 3-1 Peripheral identification register bit assignment

The four eight-bit peripheral identification registers are described in the following subsections:

- *Peripheral Identification Register 0, MPMCPeriphID0*
- *Peripheral Identification Register 1, MPMCPeriphID1*
- *Peripheral Identification Register 2, MPMCPeriphID2* on page 3-47
- *Peripheral Identification Register 3, MPMCPeriphID3* on page 3-47.

Peripheral Identification Register 0, MPMCPeriphID0

The MPMCPeriphID0 Register is hard-coded and the fields within the register determine the reset value. Table 3-43 shows the bit assignments for the MPMCPeriphID0 Register.

Table 3-43 MPMCPeriphID0 Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify.
[7:0]	PartNumber0	These bits read back as `0x76`

Peripheral Identification Register 1, MPMCPeriphID1

The MPMCPeriphID1 Register is hard-coded and the fields within the register determine the reset value. Table 3-44 shows the bit assignments for the MPMCPeriphID1 Register.

Table 3-44 MPMCPeriphID1 Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify.
[7:4]	Designer0	These bits read back as `0x1`
[3:0]	PartNumber1	These bits read back as `0x1`

Peripheral Identification Register 2, MPMCPeriphID2

The MPMCPeriphID2 Register is hard-coded and the fields within the register determine the reset value. Table 3-45 shows the bit assignments for the MPMCPeriphID2 Register.

Table 3-45 MPMCPeriphID2 Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify.
[7:4]	Revision	These bits read back as the revision number, that can be between 0 and 15
[3:0]	Designer1	These bits read back as `0x4`

Peripheral Identification Register 3, MPMCPeriphID3

The MPMCPeriphID3 Register is hard-coded and the fields within the register determine the reset value. Table 3-46 shows the bit assignments for the MPMCPeriphID3 Register.

Table 3-46 MPMCPeriphID3 Register bit assignments

Bits	Name	Description
[31:8]	Reserved	Reserved, read undefined, do not modify.

Table 3-46 MPMCPeriphID3 Register bit assignments (continued)

Bits	Name	Description
[7]	Configuration	Static memory controller: 0 = no 1 = yes. For PL176 this field is set to 1.
[6:4]	Configuration	Indicates the AHB master bus width: 000 = 32-bit wide 001 = 64-bit wide 010 = 128-bit wide 011 = 256-bit wide 100 = 512-bit wide 101 = 1024-bit wide 110-111 = reserved. For PL176 this field is set to 001.
[3:0]	Configuration	Number of AHB slave ports: 0000 = 1 slave port 0001 = 2 slave ports 0010 = 3 slave ports 0011 = 4 slave ports 0100 = 5 slave ports 0101 = 6 slave ports 0110 = 7 slave ports 0111 = 8 slave ports 1000 = 9 slave ports 1001 = 10 slave ports 1010 = 11 slave ports 1011 = 12 slave ports 1100 = 13 slave ports 1101 = 14 slave ports 1110 = 15 slave ports 1111 = 16 slave ports. For PL176 this field is set to 1001.

3.2.38 PrimeCell Identification Registers 0-3, MPMCPCellID0-3

The MPMCPCellID0-3 Registers are four eight-bit wide registers, that span address locations `0xFF0-0xFFC`. The registers can conceptually be treated as a single register that holds a 32-bit PrimeCell ID value. The register can be used for automatic BIOS configuration. The MPMCCellID Register is set to `0xB105F00D`.

The register can be accessed with one wait state.

Table 3-47 shows the bit assignments for the conceptual PrimeCell ID Register.

Table 3-47 Conceptual PrimeCell ID Register bit assignments

PrimeCell ID Register		MPMCPCellID0-3 Register		
Bits	**Reset value**	**Register**	**Bits**	**Description**
-	-	MPMCPCellID3	[31:8]	Reserved, read undefined, do not modify
[31:24]	`0xB1`	MPMCPCellID3	[7:0]	These bits read back as `0xB1`
-	-	MPMCPCellID2	[31:8]	Reserved, read undefined, do not modify
[23:16]	`0x05`	MPMCPCellID2	[7:0]	These bits read back as `0x05`
-	-	MPMCPCellID1	[31:8]	Reserved, read undefined, do not modify
[15:8]	`0xF0`	MPMCPCellID1	[7:0]	These bits read back as `0xF0`
-	-	MPMCPCellID0	[31:8]	Reserved, read undefined, do not modify
[7:0]	`0x0D`	MPMCPCellID0	[7:0]	These bits read back as `0x0D`

Chapter 4
Programmer's Model for Test

This chapter describes the additional logic for integration testing. It contains the following sections:

- *PrimeCell MPMC test harness overview* on page 4-2
- *Production test* on page 4-3
- *Test registers* on page 4-4.

4.1 PrimeCell MPMC test harness overview

The additional logic for functional verification and integration vectors enables:

- capture of input signals to the block
- stimulation of the output signals.

The integration vectors provide a way of verifying that the PrimeCell MPMC is correctly wired into a system. This is done by separately testing three groups of signals:

AMBA signals
These are the AMBA bus signals.

Non-AMBA intra-chip signals
Non-AMBA intra-chip signals are on-chip signals that are not part of the AMBA bus, for example, the interrupt request signals.

Primary inputs and outputs The primary input and output signals are those that go off and on the chip.

4.1.1 AMBA test strategy

These signals are tested by performing various AMBA bus transactions.

4.1.2 Non-AMBA intra-chip integration test strategy

Non-AMBA intra-chip signals are on-chip signals that are not part of the AMBA bus, for example, the interrupt request signals.

Many of these signals are difficult to control and observe. Therefore PrimeCells have integration test logic to make this task easier. These test features are enabled by programming the MPMC Integration Test Control Register (MPMCITCR). The intra-chip input signal values can then be read using the MPMCITIP1 register. The intra-chip output signals can be controlled using the MPMCITOP register.

4.1.3 Primary I/O test strategy

The primary I/O signals are tested by performing a sequence of memory accesses.

4.2 Production test

The PrimeCell MPMC is designed to simplify:

- insertion of scan test cells
- use of *Automatic Test Pattern Generation* (ATPG).

Scan insertion and ATPG is the recommended method of manufacturing test.

4.3 Test registers

The PrimeCell MPMC test registers are memory-mapped as shown in Table 4-1.

Table 4-1 Test registers memory map

Register	Offset from base	Type	Width	Reset HRESETn	Reset nPOR	Description
MPMCITCR	`0xF00`	Read/write	1	`0x0`	`0x0`	See *Test Control Register, MPMCITCR*
MPMCITIP1	`0xF20`	Read/write	32	-	`0x0`	See *Test Input 1 Register, MPMCITIP1*
MPMCITIP2	`0xF24`	Read/write	12	-	`0x0`	See *Test Input 2 Register, MPMCITIP2* on page 4-7
MPMCITOP	`0xF40`	Read/write	4	-	`0x0`	See *Test Output Register, MPMCITOP* on page 4-8
MPMCITScratch	`0xF60`	Read/write	32	`0x0`	`0x0`	See *Test Scratch Register, MPMCITScratch* on page 4-9

4.3.1 Test Control Register, MPMCITCR

The MPMCITCR Register is a single-bit control register. The T bit in this register controls the input test multiplexors. This register must only be used in test mode. This register is accessed with one wait state.

Table 4-2 shows the bit assignments for the MPMCITCR Register.

Table 4-2 MPMCITCR Register bit assignments

Bits	Name	Description
[31:1]	Reserved	Reserved, read undefined, do not modify.
[0]	Test control register (T)	0 = normal mode (reset value on **nPOR** or **HRESETn**) 1 = test mode

4.3.2 Test Input 1 Register, MPMCITIP1

The MPMCITIP1 Register is a 32-bit register. This register must only be used in test mode. This register is accessed with one wait state.

Table 4-3 shows the bit assignments for the MPMCITIP1 Register.

Table 4-3 MPMCITIP1 Register bit assignments

Bits	Name	Description
[31]	MPMCDYDDRPOL	DDR-SDRAM pad interface polarity signal **MPMCDYDDRPOL**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYDDRPOL** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[30:29]	MPMCDYDDRDLY	DDR-SDRAM pad interface configuration signal **MPMCDYDDRDLY**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYDDRDLY** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[28]	MPMCDYSDRPOL	SDR-SDRAM pad interface polarity signal **MPMCDYSDRPOL**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYSDRPOL** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[27:26]	MPMCDYSDRDLY	SDR-SDRAM pad interface configuration signal **MPMCDYSDRDLY**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYSDRDLY** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[25:22]	MPMCDYCS5CASDLY	Chip select5 CAS delay signal **MPMCDYCS5CASDLY**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYCS5CASDLY** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[21:13]	MPMCDYCS5ADDRMAP	Chip select5 address map signal **MPMCDYCS5ADDRMAP**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYCS5ADDRMAP** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[12:10]	MPMCDYCS5DEVICE	Chip select5 device signal **MPMCDYCS5DEVICE**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDYCS5DEVICE** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.

Table 4-3 MPMCITIP1 Register bit assignments (continued)

Bits	Name	Description
[9]	MPMCDLLCALIACK	Value of DLL calibration acknowledge signal **MPMCDLLCALIBACK**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDLLCALIBACK** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[8]	MPMCSTCS1PB	Polarity of byte lane for chip select1 signal **MPMCSTCS1PB**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSTCS1PB** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[7]	MPMCSTCS3POL	Polarity of chip select3 signal **MPMCSTCS3POL**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSTCS3POL** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[6]	MPMCSTCS2POL	Polarity of chip select2 signal **MPMCSTCS2POL**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSTCS2POL** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH
[5]	MPMCSTCS1POL	Polarity of chip select1 signal **MPMCSTCS1POL**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSTCS1POL** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH
[4]	MPMCSTCS0POL	Polarity of chip select0 signal **MPMCSTCS0POL**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSTCS0POL** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.

Table 4-3 MPMCITIP1 Register bit assignments (continued)

Bits	Name	Description
[3:2]	MPMCSTCS1MW	Chip select one memory width **MPMCSTCS1MW**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSTCS1MW** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[1]	MPMCBIGENDIAN	Big-endian enable signal **MPMCBIGENDIAN**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCBIGENDIAN** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH
[0]	MPMCSREFREQ (SR)	Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSREFREQ** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.

4.3.3 Test Input 2 Register, MPMCITIP2

The MPMCITIP2 Register is a 12-bit register. This register must only be used in test mode. This register is accessed with one wait state.

Table 4-4 shows the bit assignments for the MPMCITIP2 Register.

Table 4-4 MPMCITIP2 Register bit assignments

Bits	Name	Description
[31:12]	-	Reserved, read undefined, do not modify.
[11]	MPMCBOOTDLY	Boot delay signal **MPMCBOOTDLY**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCBOOTDLY** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[10:7]	MPMCBSTRBEN [6:3]	Byte Strobe Enable Signal **MPMCBSTRBENx**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCBSTRBEN[6:3]** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.

Table 4-4 MPMCITIP2 Register bit assignments (continued)

Bits	Name	Description
[6:3]	MPMCCKEINIT	Clock enable signal **MPMCCKEINIT**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCCKEINIT** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[2]	MPMCDQMINIT	Data mask signal **MPMCDQMINIT**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDQMINIT** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH 1 = Set this field if the MPMCITCR T bit is HIGH.
[1]	MPMCEBIBACKOFF	EBI backoff signal **MPMCEBIBACKOFF**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCEBIBACKOFF** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH 1 = Set this field if the MPMCITCR T bit is HIGH.
[0]	MPMCEBIGNT	EBI grant signal **MPMCEBIGNT**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCEBIGNT** input signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH 1 = Set this field if the MPMCITCR T bit is HIGH.

4.3.4 Test Output Register, MPMCITOP

The MPMCITOP Register is a four-bit register. This register must only be used in test mode. This register is accessed with one wait state.

Table 4-5 shows the bit assignments for the MPMCITOP Register.

Table 4-5 MPMCITOP Register bit assignments

Bits	Name	Description
[31:10]	Reserved	Reserved, read undefined, do not modify.
[9]	MPMCDLLCALIBREQ	Value of DLL calibration request signal **MPMCDLLCALIBREQ**. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCDLLCALIBREQ** output signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field if the MPMCITCR T bit is HIGH. 1 = Set this field if the MPMCITCR T bit is HIGH.
[8:3]	Reserved	Reserved, read undefined, do not modify.
[2]	MPMCEBIREQ	Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCEBIREQ** output signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field and the **MPMCEBIREQ** output signal if the MPMCITR T bit is HIGH 1 = Set this field and the **MPMCEBIREQ** output signal if the MPMCITR T bit is HIGH.
[1]	MPMCRPVHHOUT	Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCRPVHHOUT** output signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field and the **MPMCRPVHHOUT** output signal if the MPMCITR T bit is HIGH. 1 = Set this field and the **MPMCRPVHHOUT** output signal if the MPMCITR T bit is HIGH.
[0]	MPMCSREFACK (SA)	Self-refresh acknowledge. Read: Read the value of this field if the MPMCITCR T bit is HIGH. Read the value of the **MPMCSREFACK** output signal if the MPMCITCR T bit is LOW. Write: 0 = Clear this field and the **MPMCSREFACK** output signal if the MPMCITR T bit is HIGH. 1 = Set this field and the **MPMCSREFACK** output signal if the MPMCITR T bit is HIGH.

4.3.5 Test Scratch Register, MPMCITScratch

The MPMCITScratch Register is a 32-bit general-purpose register. This register is accessed with one wait state.

Table 4-6 shows the bit assignments for the MPMCITScratch Register.

Table 4-6 MPMCITScratch Register bit assignments

Bits	Name	Description
[31:0]	Test Scratch Register	Test scratch register: `0x0000` = Reset value on **nPOR** or **HRESETn** `0x0000-0xFFFF` = All bits are general-purpose read/write. This register is used to store and retrieve data during device test.

Chapter 5
Static Memory Controller

This chapter describes the *Static Memory Controller* (SMC). It contains the following sections:

5.1 Static memory device selection

Table 5-1 shows suggested configurations for the static memory controller with different types of memory devices. These fields are found in the MPMCStaticConfig0-3 Registers, for more information see Chapter 3 *Programmer's Model*.

Table 5-1 Static memory controller configurations

Device	Write protect	Page mode
ROM	Enabled	Disabled
Page mode ROM	Enabled	Enabled
Extended wait ROM	Enabled	Disabled
SRAM	Disabled, (or enabled)[a]	Disabled
Page mode SRAM	Disabled, (or enabled)[a]	Enabled
Extended wait SRAM	Disabled, (or enabled)[a]	Disabled
Flash	Disabled, (or enabled)[a]	Disabled
Page mode Flash	Disabled, (or enabled)[a]	Enabled
Extended wait Flash	Disabled, (or enabled)[a]	Disabled
Memory mapped peripheral	Disabled, (or enabled)[a]	Disabled

a. SRAM and FLASH memory devices can be write protected if required.

Note

Extended wait and Page mode cannot be enabled simultaneously.

5.2 Write-protection

Each static memory chip select can be configured for write protection. Usually SRAM is unprotected and ROM devices must be write-protected (to avoid potential bus conflict when performing a write access to ROM), but the P field in the MPMCStaticConfig Register can be set to write-protect SRAM as well as ROM devices. If a write access is made to a write-protected memory bank, an error is indicated by the **HRESP[1:0]** signal.

Note

If a write access is made to a memory bank containing ROM devices and the chip select is not write-protected, an error indication is not returned, and the write access proceeds as normal. In this case, output enable is not brought active LOW. Depending on the ROM device, this can lead to bus conflict.

5.3 Extended wait transfers

The static memory controller supports extremely long transfer times. In normal use the memory transfers are timed using the MPMCStaticWaitRd and MPMCStaticWaitWr Registers. These registers enable transfers with up to 32 wait states. However, if an extremely slow static memory device has to be accessed you can enable the MPMCStaticConfig Extended Wait (EW) bit. When this bit is enabled the MPMCStaticExtendedWait Register is used to time both the read and write transfers. This register enables transfers to have up to 16368 wait states.

Note

Using extremely long transfer times might mean that SDRAM devices are not refreshed correctly.

Very slow transfers can severely degrade system performance because the external memory interface is tied up for long periods of time. This has detrimental effects on time-critical services, such as interrupt latency and low latency devices, for example video controllers.

5.4 Static memory initialization

Static memory must be initialized as required after Power-On Reset, **nPOR**, by programming the relevant registers in the MPMC, and the configuration registers in the external static memory device. The PrimeCell MPMC static memory configuration registers are described in the following sections:

- *Access sequencing and memory width*
- *Wait state generation*
- *Static memory read control* on page 5-6
- *Static memory write control* on page 5-21
- *Bus turnaround* on page 5-27.

5.4.1 Access sequencing and memory width

The data width of each external memory bank must be configured by programming the appropriate bank configuration register MPMCStaticConfig[n]. When the external memory bus is narrower than the transfer initiated from the current AMBA bus master, the internal bus transfer takes several external bus transfers to complete. For example, if bank 0 is configured as 8-bit wide memory and a 32-bit read is initiated, the AMBA AHB bus stalls while the PrimeCell MPMC reads four consecutive bytes from the memory. During these accesses the static memory controller block demultiplexes the four bytes into one 32-bit word on the AMBA AHB bus.

5.4.2 Wait state generation

Each bank of the PrimeCell MPMC must be configured for external transfer wait states in read and write accesses. This is achieved by programming the appropriate fields of the bank control registers:

- MPMCStaticConfig[n]
- MPMCStaticWaitWen[n]
- MPMCStaticWaitOen[n]
- MPMCStaticWaitRd[n]
- MPMCStaticWaitWr[n]
- MPMCStaticWaitPage[n]
- MPMCStaticWaitTurn[n]
- MPMCStaticExtendedWait.

The number of cycles in which an AMBA transfer completes is controlled by two additional factors:

- access width
- external memory width.

Each bank of the PrimeCell MPMC has a programmable enable for the extended wait (EW):

- The WAITRD wait state field of the MPMCStaticWaitRd0-3 Registers can be programmed to select from 1 to 32 wait states for either:
  - read memory accesses to SRAM and ROM
  - the initial read access to page mode devices.
- The WAITWR wait state field of the MPMCStaticWaitWr0-3 Registers can be programmed to select from 1 to 32 wait states for write access to SRAM.
- The MPMCStaticWaitPage0-3 Registers can be programmed to select from 1 to 32 wait states for page mode accesses.

5.4.3 Static memory read control

The static memory read controls are described in the following sections:

- *Output enable programmable delay*
- *ROM, SRAM, and Flash*
- *Asynchronous page mode read* on page 5-17.

Output enable programmable delay

The delay between the assertion of the chip select and the output enable is programmable from 0 to 15 cycles using the WAITOEN bits of the MPMCStaticWaitOen0-3 Registers. This is used to reduce the power consumption for memories that cannot provide valid output data immediately after the chip select has been asserted. The output enable is always deasserted at the same time as the chip select, at the end of the transfer.

ROM, SRAM, and Flash

The PrimeCell MPMC uses the same read timing control for ROM, SRAM, and Flash devices. Each read starts with the assertion of the appropriate memory bank chip select signals **nMPMCSTCSOUT[n]** and memory address **MPMCADDROUT[27:0]**. The read access time is determined by the number of wait states programmed for the WAITRD field of the MPMCStaticWaitRd0-3 Registers. The WAITTURN field in the MPMCStaticWaitTurn0-3 Registers determines the number of bus turnaround wait states added between external read and write transfers.

Figure 5-1 shows an external memory read transfer with the minimum zero wait states (WAITRD = 0). A minimum of six AHB wait states are inserted during all single-read transfers. See Table 5-2 for the programmed values.

Table 5-2 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	NA
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

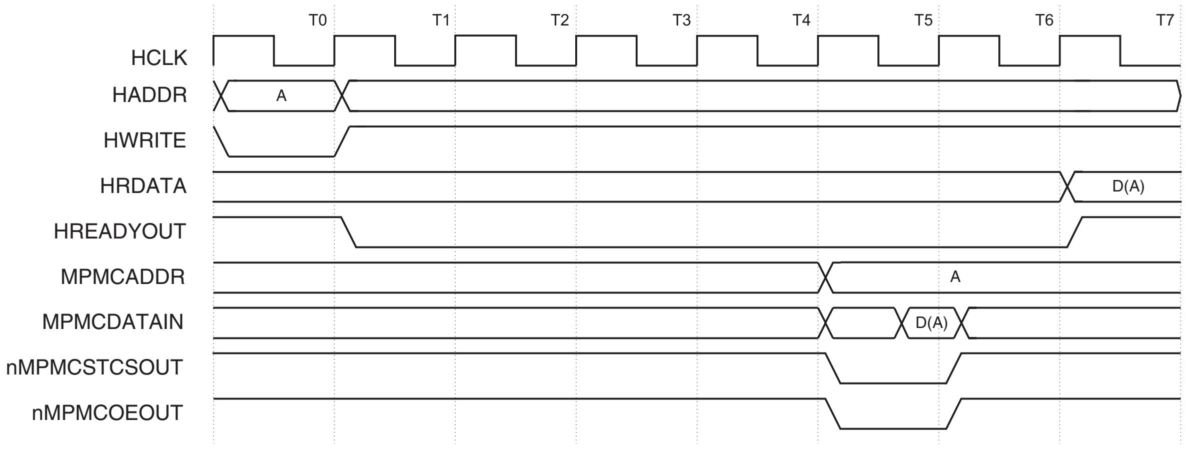


Figure 5-1 External memory zero wait state read timing diagram

Table 5-3 describes the transactions in Figure 5-1 on page 5-7.

Table 5-3 External memory zero wait state read

Cycle	Description
T0	AHB address provided to memory controller
T0-T1	AHB transaction processing
T1-T2	Arbitration of AHB memory ports
T2-T3	Memory controller processing
T3-T4	Memory controller processing
T4-T5	Static memory address, chip select, and control signals submitted to static memory
T5-T6	Read data returned from the static memory
T6-T7	Data is provided to AHB

Figure 5-2 on page 5-9 shows an external memory read transfer with two wait states (WAITRD = 2). Eight AHB wait states are inserted during the transfer, six for the standard read access, and an additional two because of the programmed wait states added (WAITRD). See Table 5-4 for the programmed values.

Table 5-4 Static memory timing parameters

Timing parameter	Value
WAITRD	2
WAITOEN	0
WAITPAGE	NA
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

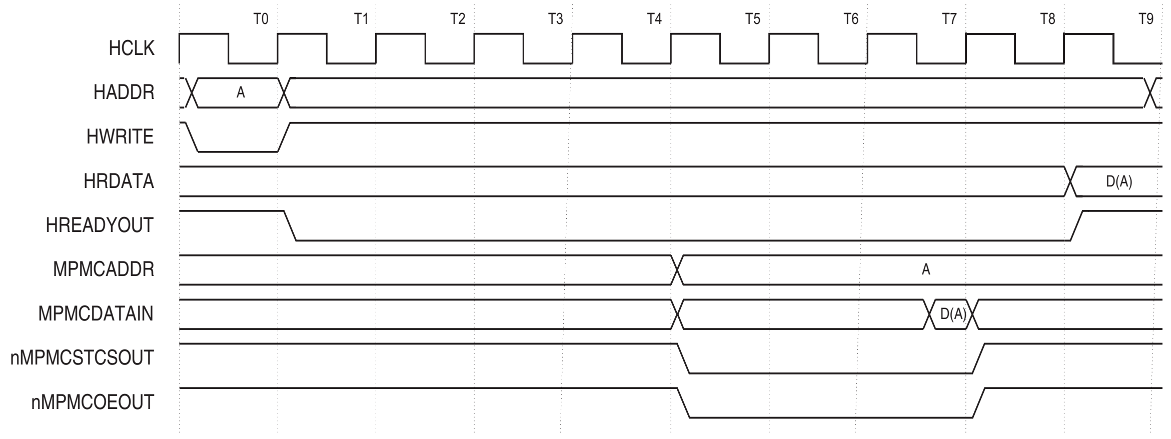


Figure 5-2 External memory two wait state read timing diagram

Table 5-5 describes the transactions for Figure 5-2.

Table 5-5 External memory two wait state read

Cycle	Description
T0	AHB address provided to memory controller
T0-T1	AHB transaction processing
T1-T2	Arbitration of AHB memory ports
T2-T3	Memory controller processing
T3-T4	Memory controller processing
T4-T5	Static memory address, chip select, and control signals submitted to static memory
T5-T6	Read wait state 1
T6-T7	Read wait state 2
T7-T8	Read data returned from the static memory
T8-T9	Data is provided to the AHB

Figure 5-3 shows an external memory read transfer with two output enable delay states (WAITOEN = 2). Eight AHB wait states are inserted during the transfer, six for the standard read, and an additional two because of the output enable delay states added. See Table 5-6 for the programmed values.

Table 5-6 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	2
WAITPAGE	NA
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

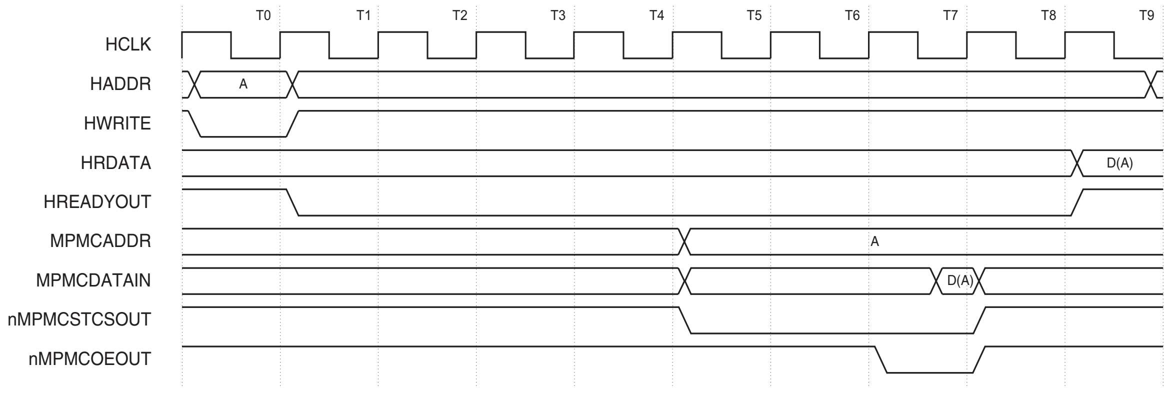


Figure 5-3 External memory two output enable delay state read timing diagram

Table 5-7 describes the transactions for Figure 5-3 on page 5-10.

Table 5-7 External memory two output enable delay state

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory address, chip select, and control signals submitted to static memory. Static memory output enable inactive.
T5-T6	Static memory output enable inactive.
T6-T7	Static memory output enable active.
T7-T8	Read data returned from the static memory.
T8-T9	Data is provided to the AHB.

Figure 5-4 on page 5-12 shows external memory read transfers with zero wait states (WAITRD = 0). These can be nonsequential transfers, or sequential transfers of unspecified burst length. All transfers are treated as separate reads, so have the minimum of six AHB wait states added. See Table 5-8 for the programmed values.

Table 5-8 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	NA
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

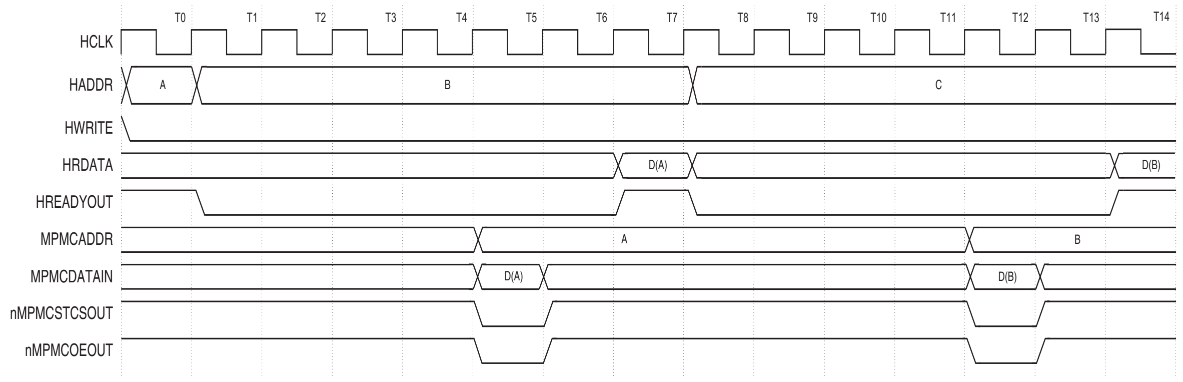


Figure 5-4 External memory two zero wait state read timing diagram

Table 5-9 describes the transactions for Figure 5-4.

Table 5-9 External memory two zero wait state reads

Cycle	Description
T0	AHB address provided to memory controller
T0-T1	AHB transaction processing
T1-T2	Arbitration of AHB memory ports
T2-T3	Memory controller processing
T3-T4	Memory controller processing
T4-T5	Static memory address, chip select, and control signals submitted to static memory
T5-T6	Read data returned from the static memory
T6-T7	Data is provided to the AHB
T7-T8	AHB transaction processing
T8-T9	Arbitration of AHB memory ports
T9-T10	Memory controller processing
T10-T11	Memory controller processing

Table 5-9 External memory two zero wait state reads (continued)

Cycle	Description
T11-T12	Static memory address, chip select, and control signals submitted to static memory
T12-T13	Read data returned from the static memory
T13-T14	Data is provided to the AHB

Figure 5-5 on page 5-14 shows a burst of zero wait state reads with the length specified. Because the length of the burst is known, it is possible to hold the chip select asserted during the whole burst, and generate the external transfers before the current AHB transfer has completed. Therefore, the first read has six AHB wait states added, and the three following sequential reads have zero AHB wait states added because of the automatic generation of the external transfers. See Table 5-10 for the programmed values.

Table 5-10 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	NA
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

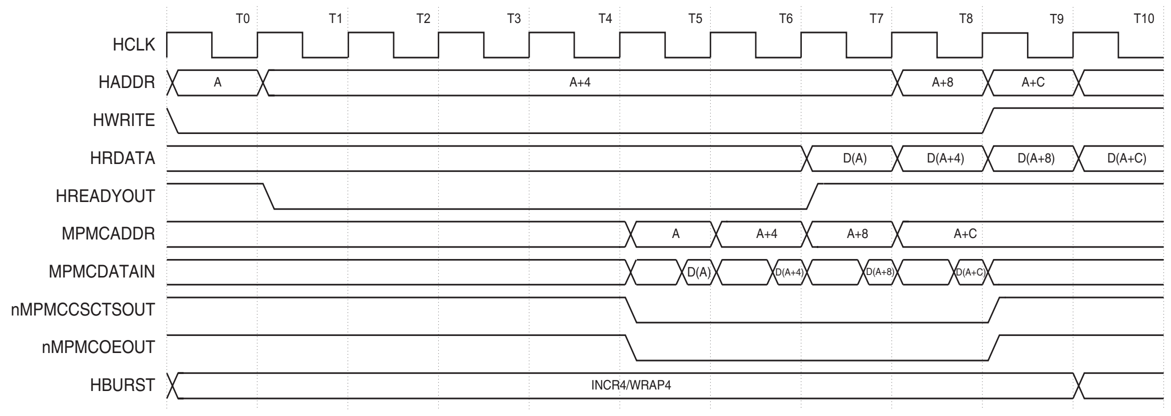


Figure 5-5 External memory zero wait fixed length burst read timing diagram

Table 5-11 describes the transactions for Figure 5-5.

Table 5-11 External memory zero wait fixed length burst read

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory read 0 address, chip select, and control signals submitted to static memory.
T5-T6	Static memory read 1 address, chip select, and control signals submitted to static memory. Read data 0 returned from the static memory.
T6-T7	Static memory read 2 address, chip select, and control signals submitted to static memory. Read data 1 returned from the static memory. Read data 0 is provided to the AHB.

Table 5-11 External memory zero wait fixed length burst read (continued)

Cycle	Description
T7-T8	Static memory read 3 address, chip select, and control signals submitted to static memory. Read data 2 returned from the static memory. Read data 1 is provided to the AHB.
T8-T9	Read data 3 returned from the static memory. Read data 2 is provided to the AHB.
T9-T10	Read data 3 is provided to the AHB.

Figure 5-6 on page 5-16 shows a burst of two wait state reads with the length specified. The WAITRD value is used for all transfers in the burst, with the first read having eight AHB wait states inserted, and all sequential transfers having two AHB wait states. See Table 5-12 for the programmed values.

Table 5-12 Static memory timing parameters

Timing parameter	Value
WAITRD	2
WAITOEN	0
WAITPAGE	NA
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

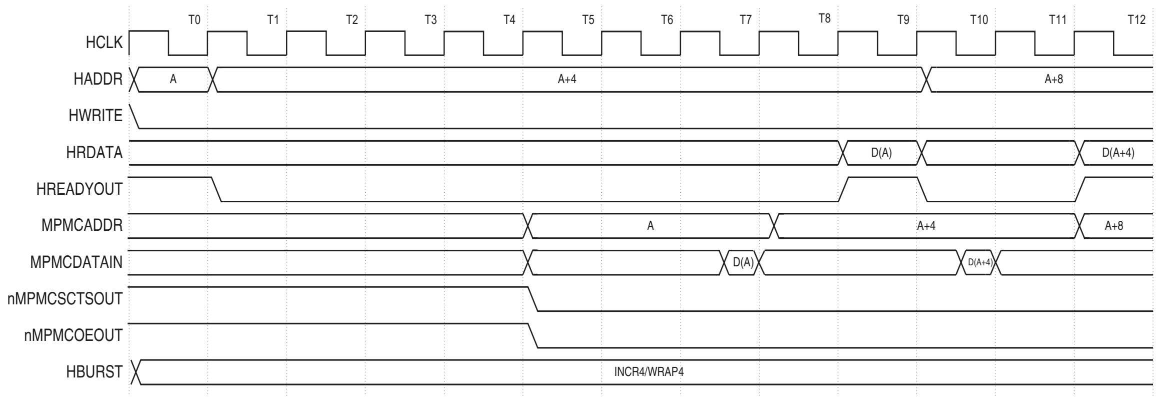


Figure 5-6 External memory two wait states fixed length burst read timing diagram

Table 5-13 describes the transactions for Figure 5-6.

Table 5-13 External memory two wait states fixed length burst read

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory address, chip select, and control signals submitted to static memory.
T5-T6	Read wait state 1.
T6-T7	Read wait state 2.
T7-T8	Read data 0 returned from the static memory. Static memory transfer 1, address, chip select, and control signals submitted to static memory.
T8-T9	Read wait state 1. Read data 0 is provided to the AHB.

Table 5-13 External memory two wait states fixed length burst read (continued)

Cycle	Description
T9-T10	Read wait state 2.
T10-T11	Read data 1 returned from the static memory.
T11-T12	Read wait state 1. Read data 1 is provided to the AHB.

Asynchronous page mode read

The PrimeCell MPMC supports asynchronous page mode read of up to four memory transfers by updating address bits A[1] and A[0]. This feature increases the bandwidth by using a reduced access time for the read accesses that are in page mode. The first read access takes MPMCStaticWaitRd and WAITRD cycles. Subsequent read accesses that are in page mode take MPMCStaticWaitPage and WAITPAGE cycles. The chip select and output enable lines are held during the burst, and only the lower two address bits change between subsequent accesses. At the end of the burst the chip select and output enable lines are deasserted together.

Figure 5-7 on page 5-18 shows an external memory page mode read transfer with two initial wait states, and one sequential wait state. The first read has eight AHB wait states inserted, and the following, up to 3, sequential transfers have only one AHB wait state. This gives increased performance over the equivalent nonpage mode ROM timing shown in Figure 5-6 on page 5-16. See Table 5-14 for the programmed values.

Table 5-14 Static memory timing parameters

Timing parameter	Value
WAITRD	2
WAITOEN	0
WAITPAGE	1
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

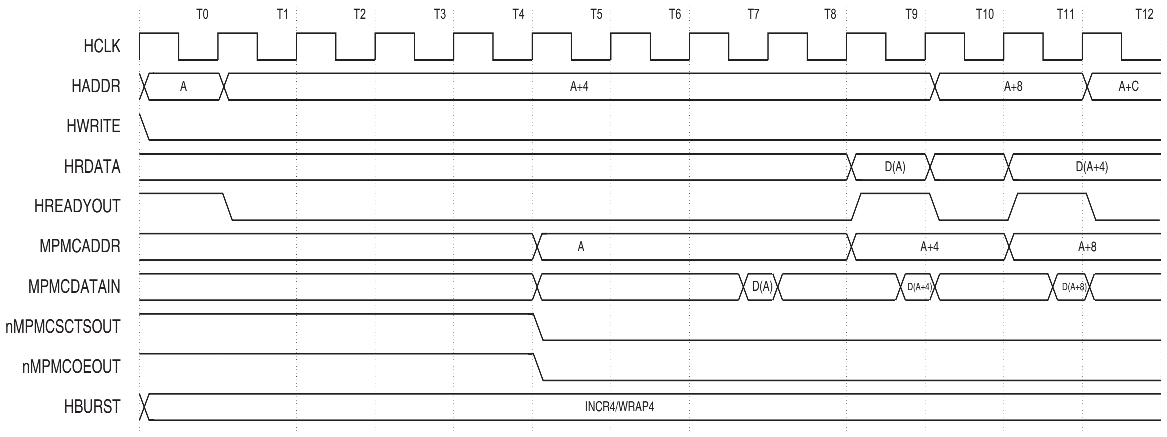


Figure 5-7 External memory page mode read transfer timing diagram

Table 5-15 describes the transactions for Figure 5-7.

Table 5-15 External memory page mode read

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory transfer 0, address, chip select, and control signals submitted to static memory.
T5-T6	Read wait state 1.
T6-T7	Read wait state 2.
T7-T8	Read data 0 returned from the static memory. Static memory transfer 1, address, chip select, and control signals submitted to static memory.
T8-T9	Read page mode wait state1. Read data 0 is provided to the AHB.

Table 5-15 External memory page mode read (continued)

Cycle	Description
T9-T10	Read data 1 returned from the static memory. Static memory transfer 2, address, chip select, and control signals submitted to static memory.
T10-T11	Read page mode wait state1. Read data1 is provided to the AHB.
T11-T12	Read data 2 returned from the static memory. Static memory transfer 3, address, chip select, and control signals submitted to static memory.

Figure 5-8 on page 5-20 shows a 32-bit read from an 8-bit page mode ROM device, causing four burst reads to be performed. A total of six AHB wait states are added during this transfer for the first external read. WAITRD and WAITPAGE are zero. See Table 5-16 for the programmed values.

Table 5-16 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	0
WAITWR	NA
WAITWEN	NA
WAITTURN	NA

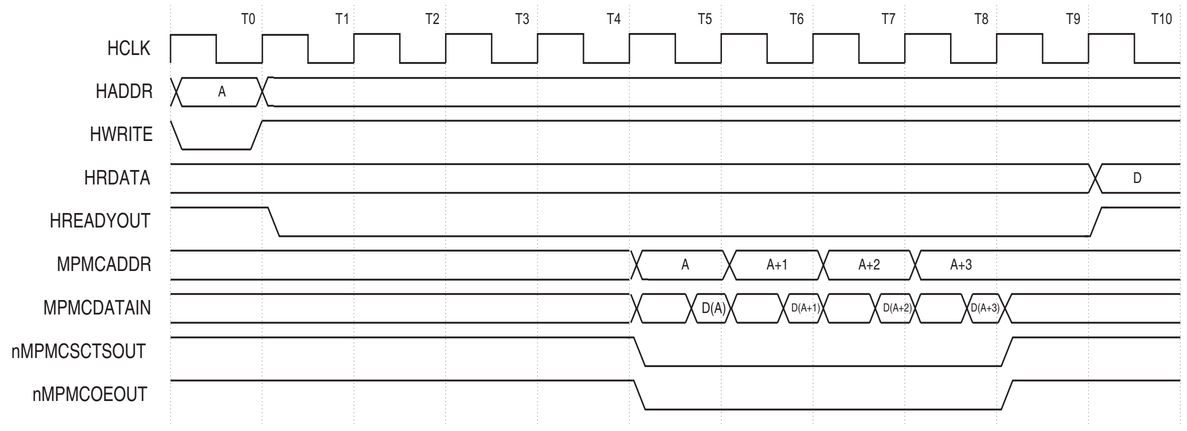


Figure 5-8 External memory 32-bit burst read from 8-bit memory timing diagram

Table 5-17 describes the transactions for Figure 5-8.

Table 5-17 External memory 32-bit burst read from 8-bit memory

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory transfer 0, address, chip select, and control signals submitted to static memory.
T5-T6	Static memory transfer1, address, chip select, and control signals submitted to static memory. Read data byte 0 returned from the static memory.
T6-T7	Static memory transfer 2, address, chip select, and control signals submitted to static memory. Read data byte 1 returned from the static memory.
T7-T8	Static memory transfer 3, address, chip select, and control signals submitted to static memory. Read data byte2 returned from the static memory.
T8-T9	Read data byte 3 returned from the static memory.
T9-T10	Read data 32-bit word is provided to the AHB.

5.4.4 Static memory write control

Write timing is described in the following sections:

- *Write enable programmable delay*
- *SRAM*
- *Flash memory* on page 5-27.

Write enable programmable delay

The delay between the assertion of the chip select and the write enable is programmable from 0-15 cycles using the WAITWEN bits of the MPMCStaticWaitWen0-3 Registers. This is used to reduce the power consumption for memories. The write enable is asserted on the rising edge of **HCLK** after the assertion of the chip select for zero wait states. The write enable is always deasserted a cycle before the chip select, at the end of the transfer. **nMPMCBLSOUT** has the same timing as **nMPMCSTWEOUT** for writes to 8-bit devices that use the byte lane selects instead of the write enables.

SRAM

Write timing for SRAM starts with assertion of the appropriate memory bank chip selects **nMPMCSTCSOUT[n]** and address signals **MPMCADDROUT[27:0]**. The write access time is determined by the number of wait states programmed for the WAITWR field of the MPMCStaticWaitWr Register. The WAITTURN field in the bank control register determines the number of bus turnaround wait states added between external read and write transfers.

Figure 5-9 on page 5-22 shows a single external memory write transfer with minimum zero wait states (WAITWR = 0). No AHB wait states are added. See Table 5-18 for the programmed values.

Table 5-18 Static memory timing parameters

Timing parameter	Value
WAITRD	NA
WAITOEN	NA
WAITPAGE	NA
WAITWR	0
WAITWEN	0
WAITTURN	NA

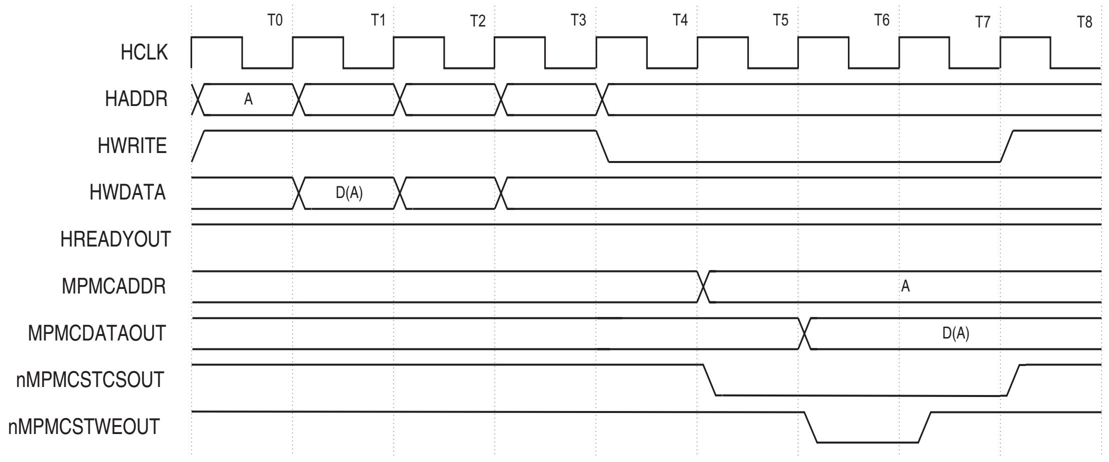


Figure 5-9 External memory zero wait state write timing diagram

Table 5-19 describes the transactions for Figure 5-9.

Table 5-19 External memory zero wait state write

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports. Write data is read from the AHB memory port.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory transfer 0, address, chip select, and control signals submitted to static memory. Write enable inactive.
T5-T6	Write enable taken active. Write data submitted to static memory. Static memory writes the data.
T6-T7	Static memory writes the data. Write enable taken inactive.
T7-T8	Static memory control signals taken inactive.

Figure 5-10 shows a single external memory write transfer with two wait states (WAITWR = 2). No AHB wait states are added. See Table 5-20 for the programmed values.

Table 5-20 Static memory timing parameters

Timing parameter	Value
WAITRD	NA
WAITOEN	NA
WAITPAGE	NA
WAITWR	2
WAITWEN	0
WAITTURN	NA

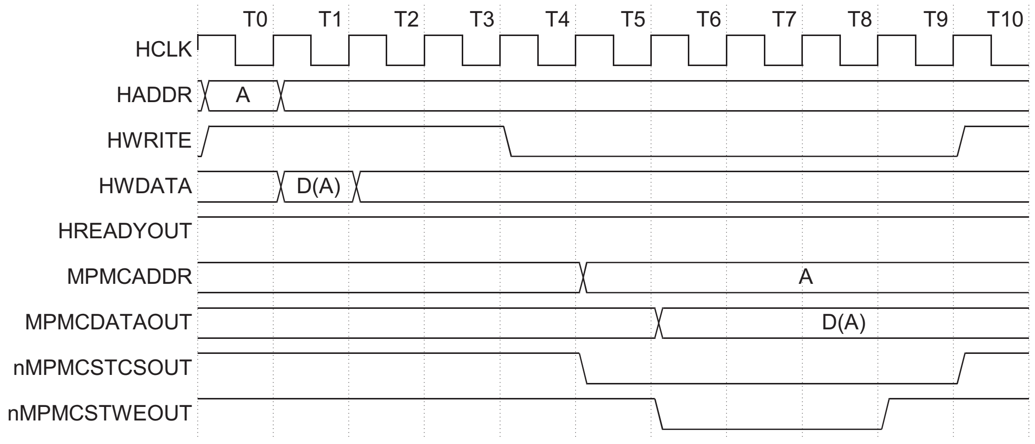


Figure 5-10 External memory two wait state write timing diagram

Table 5-21 describes the transactions for Figure 5-10 on page 5-23.

Table 5-21 External memory two wait state write

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports. Write data is read from the AHB memory port.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory transfer 0, address, chip select, and control signals submitted to static memory. Write enable inactive.
T5-T6	Write enable taken active. Write data submitted to static memory.
T6-T7	Wait state 1.
T7-T8	Wait state 2.
T8-T9	Static memory writes the data. Write enable taken inactive.
T9-T10	Static memory control signals taken inactive.

Figure 5-11 on page 5-25 shows a single external memory write transfer with two write enable delay states (WAITWEN = 2). No wait states are added. See Table 5-22 for the programmed values.

Table 5-22 Static memory timing parameters

Timing parameter	Value
WAITRD	NA
WAITOEN	NA
WAITPAGE	NA
WAITWR	0
WAITWEN	2
WAITTURN	NA

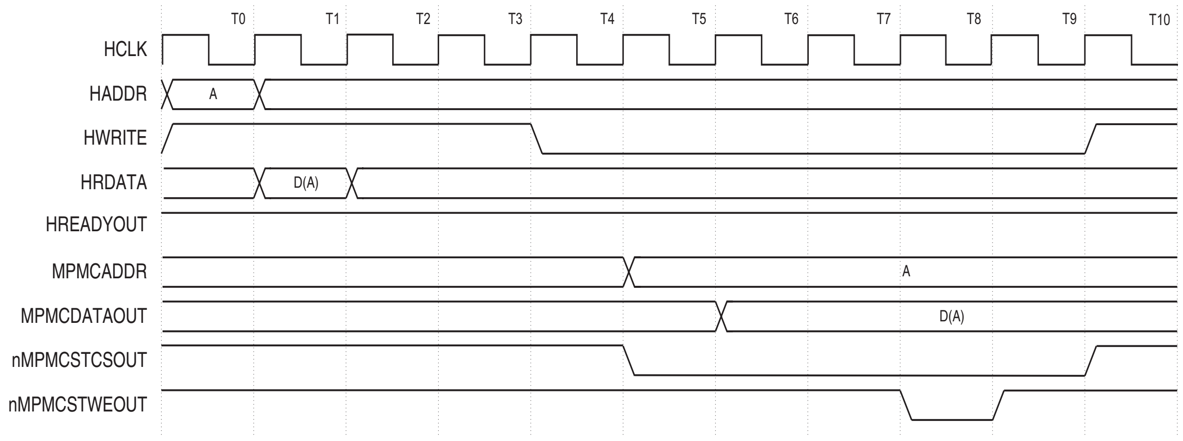


Figure 5-11 External memory two write enable delay write timing diagram

Table 5-23 describes the transactions for Figure 5-11.

Table 5-23 External memory two write enable state write

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports. Write data is read from the AHB memory port.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory transfer 0, address, chip select, and control signals submitted to static memory. Write enable inactive.
T5-T6	Write data submitted to static memory. Write enable wait state 1.
T6-T7	Write enable wait state 2.
T7-T8	Write enable taken active.
T8-T9	Static memory writes the data. Write enable taken inactive.
T9-T10	Static memory control signals taken inactive.

Figure 5-12 shows two external memory write transfers with zero wait states (WAITWR = 0). This is the timing of any sequence of write transfers, nonsequential to nonsequential, or nonsequential to sequential, with any value of **HBURST**. The maximum speed of write transfers is controlled by the external timing of the write enable relative to the chip select, so all external writes must take two cycles to complete, the cycle that write enable is asserted, and the cycle that write enable is deasserted. See Table 5-24 for the programmed values.

Table 5-24 Static memory timing parameters

Timing parameter	Value
WAITRD	NA
WAITOEN	NA
WAITPAGE	NA
WAITWR	0
WAITWEN	0
WAITTURN	0

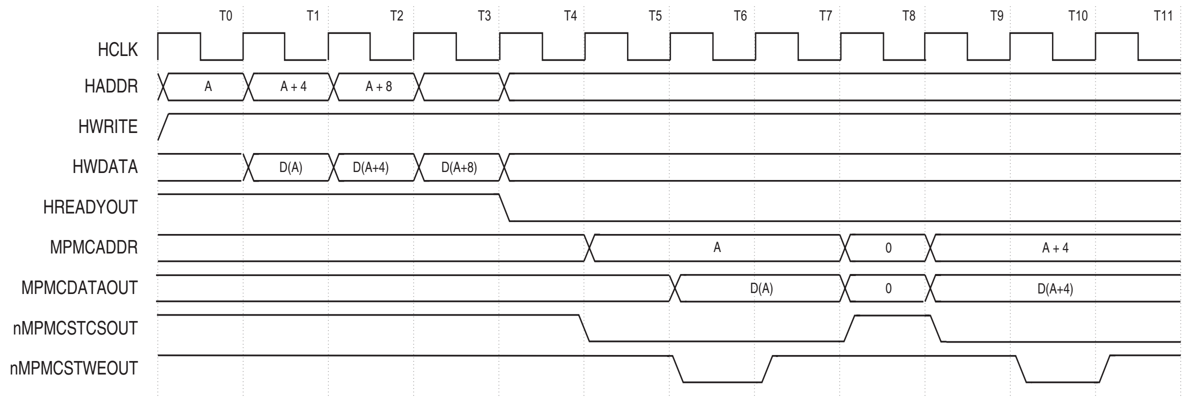


Figure 5-12 External memory two zero wait writes timing diagram

Table 5-25 describes the transactions for Figure 5-12 on page 5-26.

Table 5-25 External memory two zero wait writes

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports. Write data 0 is read from the AHB memory port.
T2-T3	Memory controller processing. Write data 1 is read from AHB memory port.
T3-T4	Memory controller processing.
T4-T5	Static memory transfer 0, address, chip select, and control signals submitted to static memory. Write enable inactive.
T5-T6	Write enable taken active. Write data submitted to static memory.
T6-T7	Static memory writes data 0. Write enable taken inactive.
T7-T8	Static memory control signals taken inactive.
T8-T9	Static memory transfer 1, address, chip select, and control signals submitted to static memory. Write enable inactive.
T9-T10	Write enable taken active. Write data submitted to static memory.
T10-T11	Static memory writes data 1. Write enable taken inactive.

Flash memory

Write timing for flash memory devices is the same as for SRAM devices.

5.4.5 Bus turnaround

The PrimeCell MPMC can be configured for each memory bank to use external bus turnaround cycles between read and write memory accesses. The WAITTURN field can be programmed for 1-16 bus turnaround wait states. This is to avoid bus contention on the external memory data bus. Bus turnaround cycles are generated between external bus transfers as follows:

- read to read (different memory banks)
- read to write (same memory bank)
- read to write (different memory banks).

Figure 5-13 shows a zero wait read followed by a zero wait write with default turnaround between the transfers of two cycles because of the timing of the AHB transfers. Standard AHB wait states are added to the transfers, six for the read, and zero for the write. See Table 5-26 for the programmed values.

Table 5-26 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	NA
WAITWR	0
WAITWEN	0
WAITTURN	0

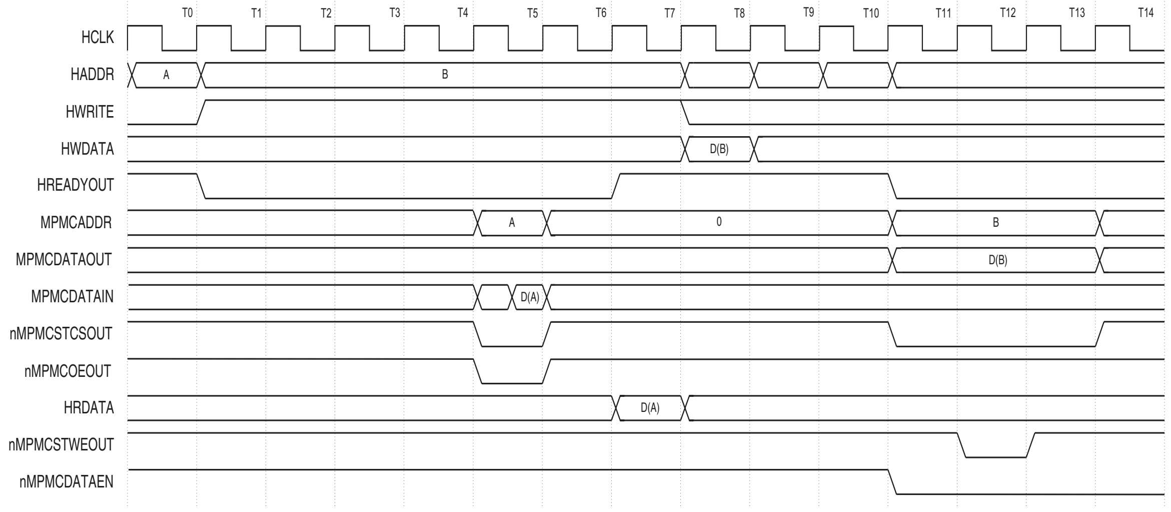


Figure 5-13 Read followed by write (both zero wait) with no turnaround

Table 5-27 describes the transactions for Figure 5-13 on page 5-28.

Table 5-27 Read followed by write (both zero wait) with no turnarounds

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory address, chip select, and control signals submitted to static memory.
T5-T6	Read data returned from the static memory.
T6-T7	Data is provided to the AHB.
T7-T8	Arbitration of AHB memory ports.
T8-T9	Memory controller processing.
T9-T10	Memory controller processing.
T10-T11	Static memory address, chip select, and control signals submitted to static memory.
T11-T12	Static memory transfer address, chip select, and control signals submitted to static memory. Write enable inactive.
T12-T13	Write enable taken active. Write data submitted to static memory.
T13-T14	Static memory write the data. Write enable taken inactive.

Figure 5-14 on page 5-30 shows a zero wait write followed by a zero wait read with default turnaround between the transfers of one cycle. Zero wait states are added to the write transfer, but 12 are added to the read. The 12 wait states for the read transfer comprises two parts. First there are six wait states to enable the previous write access to complete, and then the standard six wait states for the read transfer. See Table 5-28 for the programmed values.

Table 5-28 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	NA
WAITWR	0
WAITWEN	0
WAITTURN	0

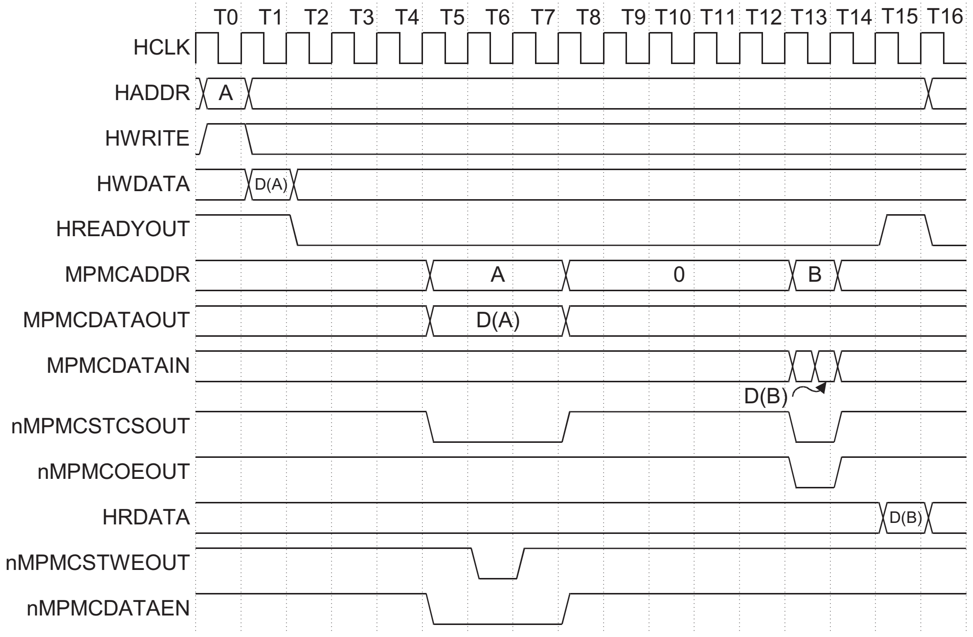


Figure 5-14 Write followed by read (both zero wait) with no turnaround

Table 5-29 describes the transactions for Figure 5-14 on page 5-30.

Table 5-29 Write followed by read (both zero wait) with no turnaround

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory address, chip select, and control signals submitted to static memory. Write enable inactive. AHB read address provided to memory controller
T5-T6	Write enable taken active. Write data submitted to static memory.
T6-T7	Static memory writes the data. Write enable taken inactive.
T7-T8	Static memory control signals taken inactive.
T9-T10	Arbitration of AHB memory ports.
T10-T11	Memory controller processing.
T11-T12	Memory controller processing.
T12-T13	Static memory address, chip select, and control signals submitted to static memory.
T13-T14	Read data returned from the static memory.
T14-T15	Data is provided to the AHB.

Figure 5-15 shows a zero wait read followed by a zero wait write with two turnaround cycles added. The standard minimum of six AHB wait states are added to the read transfer, and zero wait states are added to the write (as for any read-write transfer sequence). See Table 5-30 for the programmed values.

Table 5-30 Static memory timing parameters

Timing parameter	Value
WAITRD	0
WAITOEN	0
WAITPAGE	NA
WAITWR	0
WAITWEN	0
WAITTURN	2

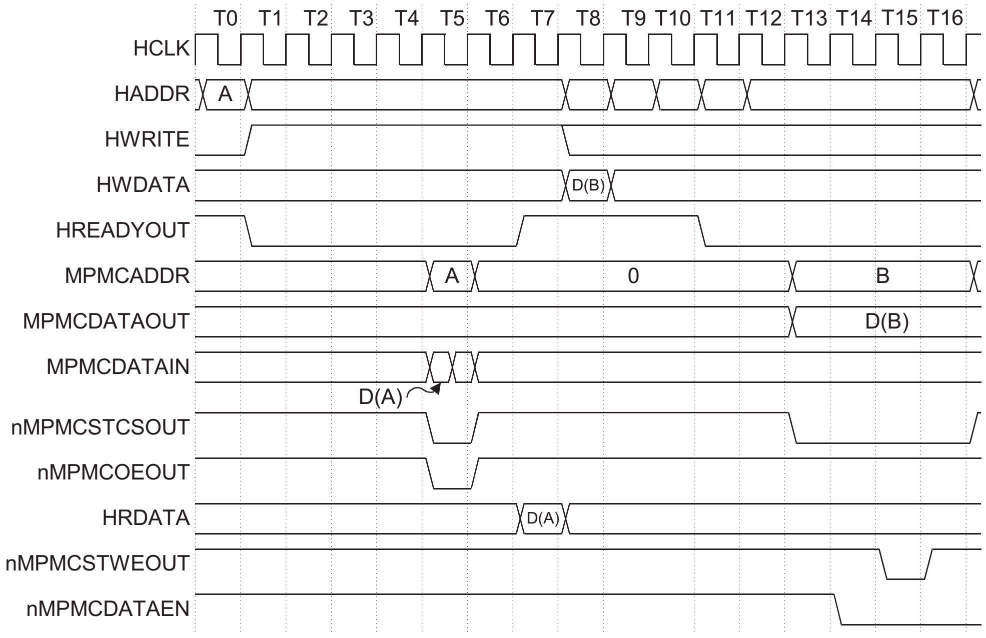


Figure 5-15 Read followed by a write (all zero wait state) with two turnaround cycles

Table 5-31 describes the transactions for Figure 5-15 on page 5-32.

Table 5-31 Read followed by a write (all zero wait state) with two turnaround cycles

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Memory controller processing.
T4-T5	Static memory address, chip select, and control signals submitted to static memory.
T5-T6	Read data returned from the static memory.
T6-T7	Turnaround cycle 1. Data is provided to the AHB.
T7-T8	Turnaround cycle 2.
T8-T9	AHB address provided to memory controller.
T9-T10	Arbitration of AHB memory ports.
T10-T11	Memory controller processing.
T11-T12	Memory controller processing.
T12-T13	Static memory transfer address, chip select, and control signals submitted to static memory. Write enable inactive.
T13-T14	Write enable taken active. Write data submitted to static memory.
T14-T15	Static memory write the data. Write enable taken inactive.

5.5 Elimination of floating bytes on the external interface

The PrimeCell MPMC uses the programmed external memory width of each bank and the endianness to determine which remaining bytes it requires to drive to ensure that the external bus is never floating. The input/output pads for the external write data bus lines are controlled by **nMPMCDATAOUTEN[3:0]** as shown in Table 5-32. Data is driven out on **MPMCDATAOUT[31:0]** when **nMPMCDATAOUTEN** is asserted LOW.

Table 5-32 MPMCDATAOUT[31:0] controlled by nMPMCDATAOUTEN[3:0]

nMPMCDATAOUTEN	MPMCDATAOUT
[3]	[31:24]
[2]	[23:16]
[1]	[15:8]
[0]	[7:0]

5.6 Byte lane control

The PrimeCell MPMC generates byte lane control signals **nMPMCBLSOUT[3:0]** according to:

- little or big-endian operation
- AMBA transfer width, indicated by **HSIZE[2:0]**
- external memory bank data bus width, defined within each control register
- the decoded **HADDR[1:0]** value for write accesses only.

Doubleword transfers are the largest size transfers supported by the 64-bit ports of the MPMC, and word transfers the largest supported by the 32-bit ports. Any access attempted with a size greater than the port size causes an ERROR response to be generated. Each memory chip select can be 8, 16, or 32 bits wide. The type of memory used determines how the **nMPMCSTWEOUT** and **nMPMCBLSOUT** signals are connected to provide byte, halfword and word access. For read accesses, you must control the **nMPMCBLSOUT** signals by driving them either all HIGH, or all LOW. This is done by programming the Byte Lane State (PB) bit within the MPMCStaticConfig0-3 Registers. *Memory banks constructed from 8-bit or non byte-partitioned memory devices* and *Memory banks constructed from 16 or 32-bit memory devices* on page 5-36 explain why different connections are required, in respect of **nMPMCSTWEOUT** and **nMPMCBLSOUT[3:0]**, for different memory configurations.

5.6.1 Memory banks constructed from 8-bit or non byte-partitioned memory devices

For memory banks constructed from 8-bit or non byte-partitioned memory devices, it is important that the Byte Lane State (PB) bit is cleared to 0 within the respective memory bank control register. This forces all **nMPMCBLSOUT[3:0]** lines HIGH during a read access as the byte lane selects are connected to the device write enables.

Figure 5-16 on page 5-36 shows 8-bit memory being used to configure memory banks that are 8, 16, and 32 bits wide. In each of these configurations, the **nMPMCBLSOUT[3:0]** signals are connected to write enable (nWE) inputs of each 8-bit memory. The **nMPMCSTWEOUT** signal from the PrimeCell MPMC is not used. For write transfers, the relevant **nMPMCBLSOUT[3:0]** byte lane signals are asserted LOW, and steer the data to the addressed bytes. For read transfers, all of the **nMPMCBLSOUT[3:0]** lines are deasserted HIGH, that enables the external bus to be defined for at least the width of the accessed memory.

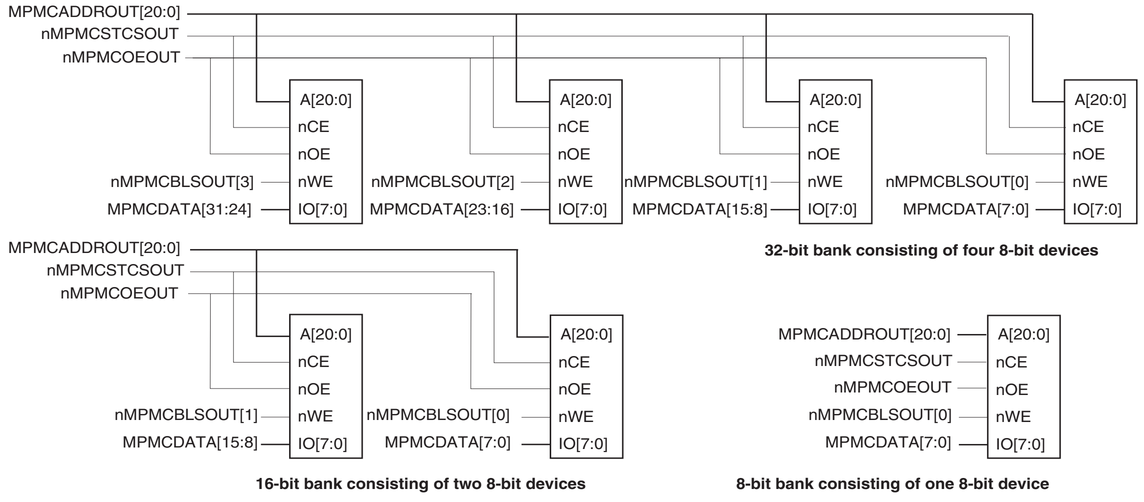


Figure 5-16 Memory banks constructed from 8-bit memory

5.6.2 Memory banks constructed from 16 or 32-bit memory devices

For memory banks constructed from 16 or 32-bit memory devices, it is important that the Byte Lane Select (PB) bit is set to 1 within the respective memory bank control register. This asserts all **nMPMCBLSOUT[3:0]** lines LOW during a read access as during a read all bytes of the devices must be selected to avoid undriven byte lanes on the read data value. In the case of 16 and 32-bit wide memory devices, byte select signals exist and these must be appropriately controlled, as shown in Figure 5-17 and Figure 5-18 on page 5-37.

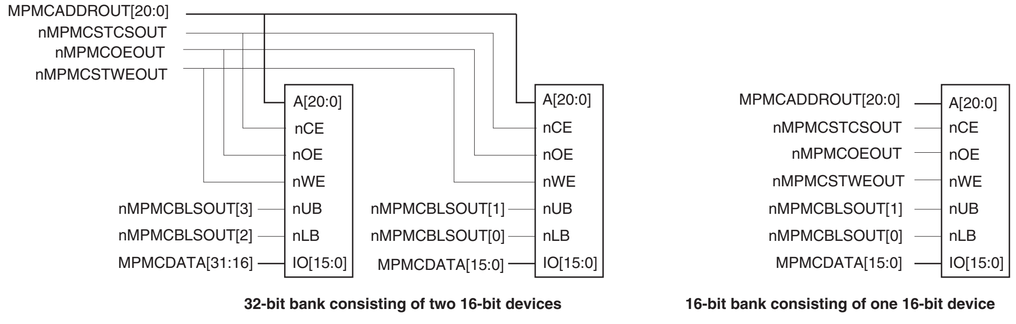
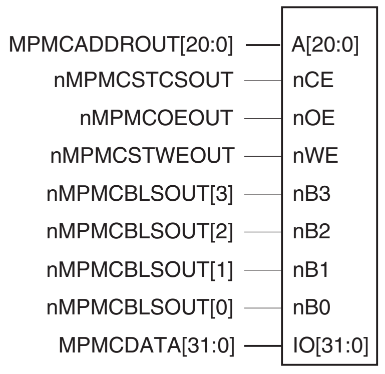


Figure 5-17 Memory banks constructed from 16-bit memory



32-bit bank consisting of one 32-bit device

Figure 5-18 Memory bank constructed from 32-bit memory

Figure 5-19 on page 5-38 shows connections for a typical memory system with different data width memory devices.

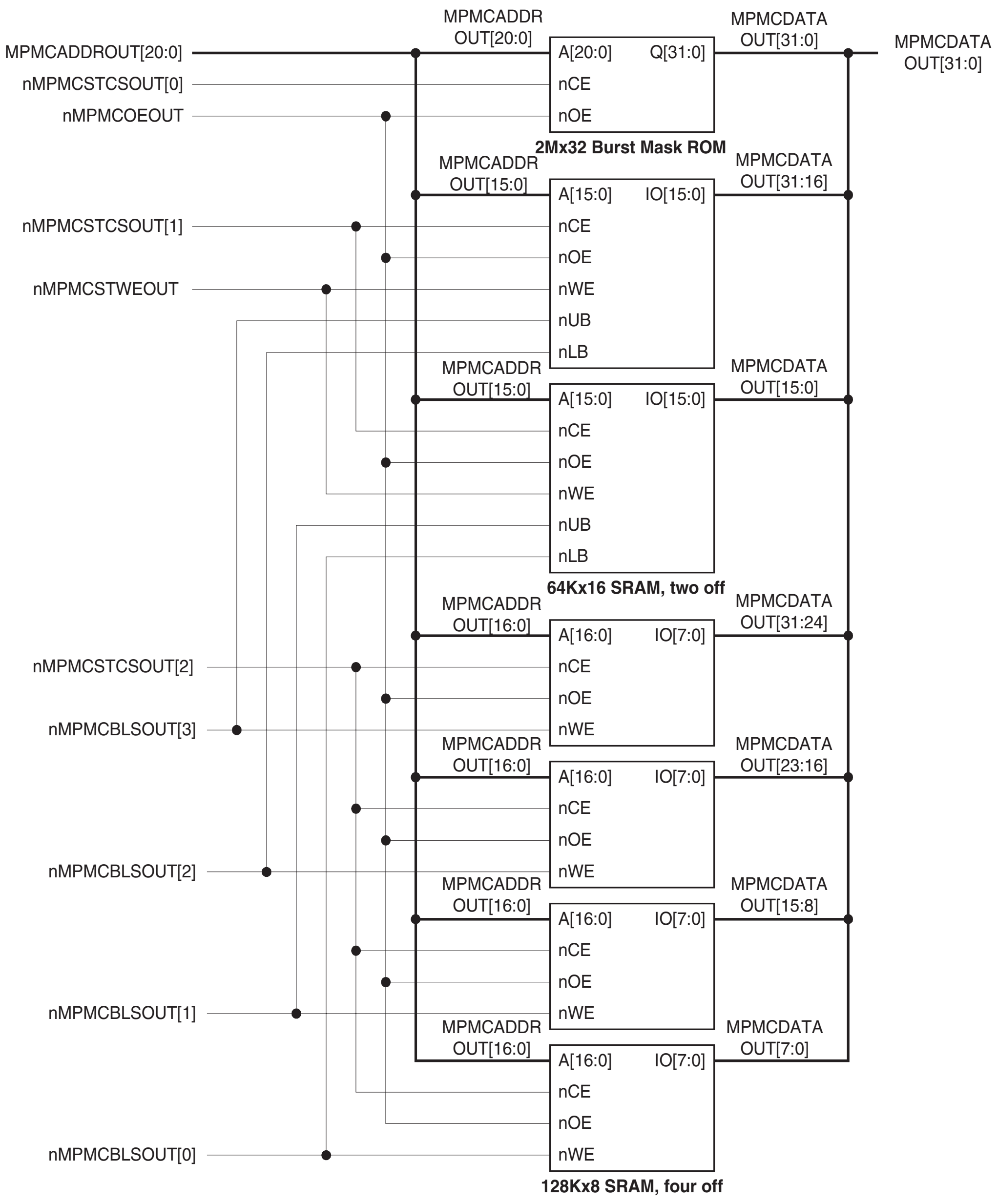


Figure 5-19 Typical memory connection diagram

5.7 Byte lane control and data bus steering

Table 5-33 to Table 5-44 on page 5-46 show the relationship of signals **HSIZE[2:0]**, **HADDR[1:0]**, **MPMCADDROUT[1:0]**, and **nMPMCBLSOUT[3:0]** and mapping of data between the AHB system data bus and the external memory data bus. This mapping applies to both the static and dynamic memory controllers.

Table 5-33 Little-endian read, 8-bit external bus

Internal transfer width	Access: Read, little-endian, 8-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[1:0]**	**nMPMCBLS OUT[0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (4 transfers)	010 010 010 010	-- -- -- --	11 10 01 00	0 0 0 0	[7:0] - - -	- [7:0] - -	- - [7:0] -	- - - [7:0]
Halfword (2 transfers)	001	1-	11 10	0 0	[7:0] -	- [7:0]	- -	- -
Halfword (2 transfers)	001	0-	01 00	0 0	- -	- -	[7:0] -	- [7:0]
Byte	000	11	11	0	[7:0]	-	-	-
Byte	000	10	10	0	-	[7:0]	-	-
Byte	000	01	01	0	-	-	[7:0]	-
Byte	000	00	00	0	-	-	-	[7:0]

Table 5-34 Little-endian read, 16-bit external bus

Internal transfer width	Access: Read, little-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	HADDR [1:0]	MPMCADDR OUT[0]	nMPMCBLS OUT[1:0]	[31:4]	[23:16]	[15:8]	[7:0]
Word (2 transfers)	010 010	-- --	1 0	00 00	[15:8] -	[7:0] -	- [15:8]	- [7:0]
Halfword	001	1-	1	00	[15:8]	[7:0]	-	-
Halfword	001	0-	0	00	-	-	[15:8]	[7:0]
Byte	000	11	1	01	[15:8]	-	-	-
Byte	000	10	1	10	-	[7:0]	-	-
Byte	000	01	0	01	-	-	[15:8]	-
Byte	000	00	0	10	-	-	-	[7:0]

Table 5-35 Little-endian read, 32-bit external bus

Internal transfer width	Access: Read, little-endian, 32-bit external bus			External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	HADDR [1:0]	nMPMCBLS OUT[3:0]	[31:24]	[23:16]	[15:8]	[7:0]
Word	010	--	0000	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	1-	0011	[31:24]	[23:16]	-	-
Halfword	001	0-	1100	-	-	[15:8]	[7:0]
Byte	000	11	0111	[31:24]	-	-	-
Byte	000	10	1011	-	[23:16]	-	-
Byte	000	01	1101	-	-	[15:8]	-
Byte	000	00	1110	-	-	-	[7:0]

Table 5-36 Little-endian write, 8-bit external bus

Internal transfer width	**Access: Write, little-endian, 8-bit external bus**				**System data mapping on to external data bus MPMCDATA to HRDATA**			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[1:0]**	**nMPMCBLS OUT[0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (4 transfers)	010 010 010 010	-- -- -- --	11 10 01 00	0 0 0 0	- - - -	- - - -	- - - -	[31:24] [23:16] [15:8] [7:0]
Halfword (2 transfers)	001	1-	11 10	0 0	- -	- -	- -	[31:24] [23:16]
Halfword (2 transfers)	001	0-	01 00	0 0	- -	- -	- -	[15:8] [7:0]
Byte	000	11	11	0	-	-	-	[31:24]
Byte	000	10	10	0	-	-	-	[23:16]
Byte	000	01	01	0	-	-	-	[15:8]
Byte	000	00	00	0	-	-	-	[7:0]

Table 5-37 Little-endian write, 16-bit external bus

Internal transfer width	**Access: Write, little-endian, 16-bit external bus**				**System data mapping on to external data bus MPMCDATA to HRDATA**			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[0]**	**nMPMCBLS OUT[1:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (2 transfers)	010 010	-- --	1 0	00 00	- -	- -	[31:24] [15:8]	[23:16] [7:0]
Halfword	001	1-	1	00	-	-	[31:24]	[23:16]
Halfword	001	0-	0	00	-	-	[15:8]	[7:0]
Byte	000	11	1	01	-	-	[31:24]	-

Table 5-37 Little-endian write, 16-bit external bus (continued)

Internal transfer width	Access: Write, little-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[0]**	**nMPMCBLS OUT[1:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Byte	000	10	1	10	-	-	-	[23:16]
Byte	000	01	0	01	-	-	[15:8]	-
Byte	000	00	0	10	-	-	-	[7:0]

Table 5-38 Little-endian write, 32-bit external bus

Internal transfer width	Access: Write, little-endian, 32-bit external bus			System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**nMPMCBLS OUT[3:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word	010	--	0000	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	1-	0011	[31:24]	[23:16]	-	-
Halfword	001	0-	1100	-	-	[15:8]	[7:0]
Byte	000	11	0111	[31:24]	-	-	-
Byte	000	10	1011	-	[23:16]	-	-
Byte	000	01	1101	-	-	[15:8]	-
Byte	000	00	1110	-	-	-	[7:0]

Table 5-39 Big-endian read, 8-bit external bus

Internal transfer width	Access: Read, big-endian, 8-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[1:0]**	**nMPMCBLS OUT[0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (4 transfers)	010 010 010 010	-- -- -- --	11 10 01 00	0 0 0 0	- - - [7:0]	- - [7:0] -	- [7:0] - -	[7:0]- - -
Halfword (2 transfers)	001	1-	11 10	0 0	- -	- -	- [7:0]	[7:0] -
Halfword (2 transfers)	001	0-	01 00	0 0	- [7:0]	[7:0] -	- -	- -
Byte	000	11	11	0	-	-	-	[7:0]
Byte	000	10	10	0	-	-	[7:0]	-
Byte	000	01	01	0	-	[7:0]	-	-
Byte	000	00	00	0	[7:0]	-	-	-

Table 5-40 Big-endian read, 16-bit external bus

Internal transfer width	Access: Read, Big-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[0]**	**nMPMCBLS OUT[1:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (2 transfers)	010 010	-- --	1 0	00 00	- [15:8]	- [7:0]	[15:8] -	[7:0] -
Halfword	001	1-	1	00	-	-	[15:8]	[7:0]
Halfword	001	0-	0	00	[15:8]	[7:0]	-	-
Byte	000	11	1	10	-	-	-	[7:0]

Table 5-40 Big-endian read, 16-bit external bus (continued)

Internal transfer width	Access: Read, Big-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[0]**	**nMPMCBLS OUT[1:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Byte	000	10	1	01	-	-	[15:8]	-
Byte	000	01	0	10	-	[7:0]	-	-
Byte	000	00	0	01	[15:8]	-	-	-

Table 5-41 Big-endian read, 32-bit external bus

Internal transfer width	Access: Read, big-endian, 32-bit external bus			External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**nMPMCBLS OUT[3:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word	010	--	0000	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	1-	1100	-	-	[15:8]	[7:0]
Halfword	001	0-	0011	[31:24]	[23:16]	-	-
Byte	000	11	1110	-	-	-	[7:0]
Byte	000	10	1101	-	-	[15:8]	-
Byte	000	01	1011	-	[23:16]	-	-
Byte	000	00	0111	[31:24]	-	-	-

Table 5-42 Big-endian write, 8-bit external bus

Internal transfer width	Access: Write, big-endian, 8-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[1:0]**	**nMPMCBLS OUT[0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (4 transfers)	010 010 010 010	-- -- -- --	11 10 01 00	0 0 0 0	- - - -	- - - -	- - - -	[7:0] [15:8] [23:16] [31:24]
Halfword (2 transfers)	001	1-	11 10	0 0	- -	- -	- -	[7:0] [15:8]
Halfword (2 transfers)	001	0-	01 00	0 0	- -	- -	- -	[23:16] [31:24]
Byte	000	11	11	0	-	-	-	[7:0]
Byte	000	10	10	0	-	-	-	[15:8]
Byte	000	01	01	0	-	-	-	[23:16]
Byte	000	00	00	0	-	-	-	[31:24]

Table 5-43 Big-endian write, 16-bit external bus

Internal transfer width	Access: Write, big-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[0]**	**nMPMCBLS OUT[1:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word (2 transfers)	010 010	-- --	1 0	00 00	- -	- -	[15:8] [31:24]	[7:0] [23:16]
Halfword	001	1-	1	00	-	-	[15:8]	[7:0]
Halfword	001	0-	0	00	-	-	[31:24]	[23:16]
Byte	000	11	1	10	-	-	-	[7:0]

Table 5-43 Big-endian write, 16-bit external bus (continued)

Internal transfer width	Access: Write, big-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**MPMCADDR OUT[0]**	**nMPMCBLS OUT[1:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Byte	000	10	1	01	-	-	[15:8]	-
Byte	000	01	0	10	-	-	-	[23:16]
Byte	000	00	0	01	-	-	[31:24]	-

Table 5-44 Big-endian write, 32-bit external bus

Internal transfer width	Access: Write, big-endian, 32-bit external bus			System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	**HADDR [1:0]**	**nMPMCBLS OUT[3:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word	010	--	0000	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	1-	1100	-	-	[15:8]	[7:0]
Halfword	001	0-	0011	[31:24]	[23:16]	-	-
Byte	000	11	1110	-	-	-	[7:0]
Byte	000	10	1101	-	-	[15:8]	-
Byte	000	01	1011	-	[23:16]	-	-
Byte	000	00	0111	[31:24]	-	-	-

Table 5-45 to Table 5-56 on page 5-65 show the relationship of signals **HSIZE[2:0]**, **HADDR[2:0]**, **MPMCADDROUT[1:0]**, and **nMPMCBLSOUT[3:0]**, and mapping of data between a 64-bit AHB port data bus and the external SRAM data bus.

Table 5-45 Little-endian read, 8-bit external bus

Internal transfer width	Access: Read, little-endian, 8-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMC ADDR OUT[2:0]	nMPMC BLS OUT[0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (8 transfers)	011 011 011 011 011 011 011 011	--- --- --- --- --- --- --- ---	111 110 101 100 011 010 001 000	0 0 0 0 0 0 0 0	[7:0] - - - - - - -	- [7:0] - - - - - -	- - [7:0] - - - - -	- - - [7:0] - - - -	- - - - [7:0] - - -	- - - - - [7:0] - -	- - - - - - [7:0] -	- - - - - - - [7:0]
Word (4 transfers)	010 010 010 010	1-- 1-- 1-- 1--	111 110 101 100	0 0 0 0	[7:0] - - -	- [7:0] - -	- - [7:0] -	- - - [7:0]	- - - -	- - - -	- - - -	- - - -
Word (4 transfers)	010 010 010 010	0-- 0-- 0-- 0--	011 010 001 000	0 0 0 0	- - - -	- - - -	- - - -	- - - -	[7:0] - - -	- [7:0] - -	- - [7:0] -	- - - [7:0]
Halfword (2 transfers)	001 001	11- 11-	111 110	0 0	[7:0] -	- [7:0]	- -	- -	- -	- -	- -	- -
Halfword (2 transfers)	001 001	10- 10-	101 100	0 0	- -	- -	[7:0] -	- [7:0]	- -	- -	- -	- -
Halfword (2 transfers)	001 001	01- 01-	011 010	0 0	- -	- -	- -	- -	[7:0] -	- [7:0]	- -	- -
Halfword (2 transfers)	001 001	00- 00-	001 000	0 0	- -	- -	- -	- -	- -	- -	[7:0] -	- [7:0]

Table 5-45 Little-endian read, 8-bit external bus (continued)

Internal transfer width	Access: Read, little-endian, 8-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[2:0]	nMPMCBLSOUT[0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Byte	000	111	111	0	[7:0]	-	-	-	-	-	-	-
Byte	000	110	110	0	-	[7:0]	-	-	-	-	-	-
Byte	000	101	101	0	-	-	[7:0]	-	-	-	-	-
Byte	000	100	100	0	-	-	-	[7:0]	-	-	-	-
Byte	000	011	011	0	-	-	-	-	[7:0]	-	-	-
Byte	000	010	010	0	-	-	-	-	-	[7:0]	-	-
Byte	000	001	001	0	-	-	-	-	-	-	[7:0]	-
Byte	000	000	000	0	-	-	-	-	-	-	-	[7:0]

Table 5-46 Little-endian read, 16-bit external bus

Internal transfer width	Access: Read, little-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	**HADDR [2:0]**	**MPMCADDROUT[1:0]**	**nMPMCBLSOUT[1:0]**	**[63:56]**	**[55:48]**	**[47:40]**	**[39:32]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Doubleword (4 transfers)	011 011 011 011	--- --- --- ---	11 10 01 00	00 00 00 00	[15:8] - - -	[7:0] - - -	- [15:8] - -	- [7:0] - -	- - [15:8] -	- - [7:0] -	- - - [15:8]	- - - [7:0]
Word (2 transfers)	010 010	1-- 1--	11 10	00 00	[15:8] -	[7:0] -	- [15:8]	- [7:0]	- -	- -	- -	- -
Word (2 transfers)	010 010	0-- 0--	01 00	00 00	- -	- -	- -	- -	[15:8] -	[7:0] -	- [15:8]	- [7:0]
Halfword	001	11-	11	00	[15:8]	[7:0]	-	-	-	-	-	-
Halfword	001	10-	10	00	-	-	[15:8]	[7:0]	-	-	-	-
Halfword	001	01-	01	00	-	-	-	-	[15:8]	[7:0]	-	-
Halfword	001	00-	00	00	-	-	-	-	-	-	[15:8]	[7:0]
Byte	000	111	11	01	[15:8]	-	-	-	-	-	-	-
Byte	000	110	11	10	-	[7:0]	-	-	-	-	-	-
Byte	000	101	10	01	-	-	[15:8]	-	-	-	-	-
Byte	000	100	10	10	-	-	-	[7:0]	-	-	-	-

Table 5-46 Little-endian read, 16-bit external bus (continued)

Internal transfer width	Access: Read, little-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[1:0]	nMPMCBLSOUT[1:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Byte	000	011	01	01	-	-	-	-	[15:8]	-	-	-
Byte	000	010	01	10	-	-	-	-	-	[7:0]	-	-
Byte	000	001	00	01	-	-	-	-	-	-	[15:8]	-
Byte	000	000	00	10	-	-	-	-	-	-	-	[7:0]

Table 5-47 Little-endian read, 32-bit external bus

Internal transfer width	Access: Read, little-endian, 32-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[0]	nMPMCBLSOUT[3:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (2 transfers)	011 011	--- ---	1 1	0000 0000	[31:24] -	[23:16] -	[15:8] -	[7:0] -	- [31:24]	- [23:16]	- [15:8]	- [7:0]
Word	010	1--	1	0000	[31:24]	[23:16]	[15:8]	[7:0]	-	-	-	-
Word	010	0--	0	0000	-	-	-	-	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	1	0011	[31:24]	[23:16]	-	-	-	-	-	-

Table 5-47 Little-endian read, 32-bit external bus (continued)

Internal transfer width	Access: Read, little-endian, 32-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[0]	nMPMCBLSOUT[3:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	10-	1	1100	-	-	[15:8]	[7:0]	-	-	-	-
Halfword	001	01-	0	0011	-	-	-	-	[31:24]	[23:16]	-	-
Halfword	001	00-	0	1100	-	-	-	-	-	-	[15:8]	[7:0]
Byte	000	111	1	0111	[31:24]	-	-	-	-	-	-	-
Byte	000	110	1	1011	-	[23:16]	-	-	-	-	-	-
Byte	000	101	1	1101	-	-	[15:8]	-	-	-	-	-
Byte	000	100	1	1110	-	-	-	[7:0]	-	-	-	-
Byte	000	011	0	0111	-	-	-	-	[31:24]	-	-	-
Byte	000	010	0	1011	-	-	-	-	-	[23:16]	-	-
Byte	000	001	0	1101	-	-	-	-	-	-	[15:8]	-
Byte	000	000	0	1110	-	-	-	-	-	-	-	[7:0]

Table 5-48 Little-endian write, 8-bit external bus

Internal transfer width	Access: Write, little-endian, 8-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[2:0]	nMPMCBLS OUT[0]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (8 transfers)	011 011 011 011 011 011 011 011	--- --- --- --- --- --- --- ---	111 110 101 100 011 010 001 000	0 0 0 0 0 0 0 0	- - - - - - - -	- - - - - - - -	- - - - - - - -	[63:56] [55:48] [47:40] [39:32] [31:24] [23:16] [15:8] [7:0]
Word (4 transfers)	010 010 010 010	1-- 1-- 1-- 1--	111 110 101 100	0 0 0 0	- - - -	- - - -	- - - -	[63:56] [55:48] [47:40] [39:32]
Word (4 transfers)	010 010 010 010	0-- 0-- 0-- 0--	011 010 001 000	0 0 0 0	- - - -	- - - -	- - - -	[31:24] [23:16] [15:8] [7:0]
Halfword (2 transfers)	001 001	11- 11-	111 110	0 0	- -	- -	- -	[63:56] [55:48]

Table 5-48 Little-endian write, 8-bit external bus (continued)

Internal transfer width	Access: Write, little-endian, 8-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[2:0]	nMPMCBLS OUT[0]	[31:24]	[23:16]	[15:8]	[7:0]
Halfword (2 transfers)	001 001	10- 10-	101 100	0 0	- -	- -	- -	[47:40] [39:32]
Halfword (2 transfers)	001 001	01- 01-	011 010	0 0	- -	- -	- -	[31:24] [23:16]
Halfword (2 transfers)	001 001	00- 00-	001 000	0 0	- -	- -	- -	[15:8] [7:0]
Byte	000	111	111	0	-	-	-	[63:56]
Byte	000	110	110	0	-	-	-	[55:48]
Byte	000	101	101	0	-	-	-	[47:40]
Byte	000	100	100	0	-	-	-	[39:32]
Byte	000	011	011	0	-	-	-	[31:24]
Byte	000	010	010	0	-	-	-	[23:16]
Byte	000	001	001	0	-	-	-	[15:8]
Byte	000	000	000	0	-	-	-	[7:0]

Table 5-49 Little-endian write, 16-bit external bus

Internal transfer width	Access: Write, little-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[1:0]	nMPMCBLS OUT[1:0]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (4 transfers)	011 011 011 011	--- --- --- ---	11 10 01 00	00 00 00 00	- - - -	- - - -	[63:56] [47:40] [31:24] [15:8]	[55:48] [39:32] [23:16] [7:0]
Word (2 transfers)	010 010	1-- 1--	11 10	00 00	- -	- -	[63:56] [47:40]	[55:48] [39:32]
Word (2 transfers)	010 010	0-- 0--	01 00	00 00	- -	- -	[31:24] [15:8]	[23:16] [7:0]
Halfword	001	11-	11	00	-	-	[63:56]	[55:48]
Halfword	001	10-	10	00	-	-	[47:40]	[39:32]
Halfword	001	01-	01	00	-	-	[31:24]	[23:16]
Halfword	001	00-	00	00	-	-	[15:8]	[7:0]
Byte	000	111	11	01	-	-	[63:56]	-
Byte	000	110	11	10	-	-	-	[55:48]
Byte	000	101	10	01	-	-	[47:40]	-

Table 5-49 Little-endian write, 16-bit external bus (continued)

Internal transfer width	Access: Write, little-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[1:0]	nMPMCBLS OUT[1:0]	[31:24]	[23:16]	[15:8]	[7:0]
Byte	000	100	10	10	-	-	-	[39:32]
Byte	000	011	01	01	-	-	[31:24]	-
Byte	000	010	01	10	-	-	-	[23:16]
Byte	000	001	00	01	-	-	[15:8]	-
Byte	000	000	00	10	-	-	-	[7:0]

Table 5-50 Little-endian write, 32-bit external bus

Internal transfer width	Access: Write, little-endian, 32-bit external bus			System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	nMPMCBLS OUT[3:0]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (2 transfers)	011 011	--- ---	0000 0000	[63:56] [47:40]	[55:48] [39:32]	[31:24] [15:8]	[23:16] [7:0]
Word	010	1--	0000	[63:56]	[55:48]	[47:40]	[39:32]
Word	010	0--	0000	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	0011	[63:56]	[55:48]	-	-

Table 5-50 Little-endian write, 32-bit external bus (continued)

Internal transfer width	Access: Write, little-endian, 32-bit external bus			System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	nMPMCBLS OUT[3:0]	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	10-	1100	-	-	[47:40]	[39:32]
Halfword	001	01-	0011	[31:24]	[23:16]	-	-
Halfword	001	00-	1100	-	-	[15:8]	[7:0]
Byte	000	111	0111	[63:56]	-	-	-
Byte	000	110	1011	-	[55:48]	-	-
Byte	000	101	1101	-	-	[47:40]	-
Byte	000	100	1110	-	-	-	[39:32]
Byte	000	011	0111	[31:24]	-	-	-
Byte	000	010	1011	-	[23:16]	-	-
Byte	000	001	1101	-	-	[15:8]	-
Byte	000	000	1110	-	-	-	[7:0]

Table 5-51 Big-endian read, 8-bit external bus

Internal transfer width	Access: Read, Big-endian, 8-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[2:0]	nMPMCBLSOUT[0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (8 transfers)	011 011 011 011 011 011 011 011	--- --- --- --- --- --- --- ---	111 110 101 100 011 010 001 000	0 0 0 0 0 0 0 0	- - - [7:0] - - - -	- - [7:0] - - - - -	- [7:0] - - - - - -	[7:0] - - - - - - -	- - - - - - - [7:0]	- - - - - - [7:0] -	- - - - - [7:0] - -	- - - - [7:0]- - -
Word (4 transfers)	010 010 010 010	1-- 1-- 1-- 1--	111 110 101 100	0 0 0 0	- - - [7:0]	- - [7:0] -	- [7:0] - -	[7:0] - - -	- - - -	- - - -	- - - -	- - - -
Word (4 transfers)	010 010 010 010	0-- 0-- 0-- 0--	011 010 001 000	0 0 0 0	- - - -	- - - -	- - - -	- - - -	- - - [7:0]	- - [7:0] -	- [7:0] - -	[7:0] - - -
Halfword (2 transfers)	001 001	11- 11-	111 110	0 0	- -	- -	- [7:0]	[7:0] -	- -	- -	- -	- -
Halfword (2 transfers)	001 001	10- 10-	101 100	0 0	- [7:0]	[7:0] -	- -	- -	- -	- -	- -	- -
Halfword (2 transfers)	001 001	01- 01-	011 010	0 0	- -	- -	- -	- -	- -	- -	- [7:0]	[7:0] -
Halfword (2 transfers)	001 001	00- 00-	001 000	0 0	- -	- -	- -	- -	- [7:0]	[7:0] -	- -	- -
Byte	000	111	111	0	-	-	-	[7:0]	-	-	-	-

Table 5-51 Big-endian read, 8-bit external bus (continued)

Internal transfer width	Access: Read, Big-endian, 8-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[2:0]	nMPMCBLSOUT[0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Byte	000	110	110	0	-	-	[7:0]	-	-	-	-	-
Byte	000	101	101	0	-	[7:0]	-	-	-	-	-	-
Byte	000	100	100	0	[7:0]	-	-	-	-	-	-	-
Byte	000	011	011	0	-	-	-	-	-	-	-	[7:0]
Byte	000	010	010	0	-	-	-	-	-	-	[7:0]	-
Byte	000	001	000	0	-	-	-	-	-	[7:0]	-	-
Byte	000	000	000	0	-	-	-	-	[7:0]	-	-	-

Table 5-52 Big-endian read, 16-bit external bus

Internal transfer width	Access: Read, Big-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[1:0]	nMPMCBLSOUT[1:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (4 transfers)	011 011 011 011	--- --- --- ---	11 10 01 00	00 00 00 00	- [15:8] - -	- [7:0] - -	[15:8] - - -	[7:0] - - -	- - - [15:8]	- - - [7:0]	- - [15:8] -	- - [7:0] -
Word (2 transfers)	010 010	1-- 1--	11 10	00 00	- [15:8]	- [7:0]	[15:8] -	[7:0] -	- -	- -	- -	- -
Word (2 transfers)	010 010	0-- 0--	01 00	00 00	- -	- -	- -	- -	- [15:8]	- [7:0]	[15:8] -	[7:0] -
Halfword	001	11-	11	00	-	-	[15:8]	[7:0]	-	-	-	-
Halfword	001	10-	10	00	[15:8]	[7:0]	-	-	-	-	-	-
Halfword	001	01-	01	00	-	-	-	-	-	-	[15:8]	[7:0]
Halfword	001	00-	00	00	-	-	-	-	[15:8]	[7:0]	-	-
Byte	000	111	11	10	-	-	-	[7:0]	-	-	-	-
Byte	000	110	11	01	-	-	[15:8]	-	-	-	-	-
Byte	000	101	10	10	-	[7:0]	-	-	-	-	-	-
Byte	000	100	10	01	[15:8]	-	-	-	-	-	-	-

Table 5-52 Big-endian read, 16-bit external bus (continued)

Internal transfer width	Access: Read, Big-endian, 16-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[1:0]	nMPMCBLSOUT[1:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Byte	000	011	01	10	-	-	-	-	-	-	-	[7:0]
Byte	000	010	01	01	-	-	-	-	-	-	[15:8]	-
Byte	000	001	00	10	-	-	-	-	-	[7:0]	-	-
Byte	000	000	00	01	-	-	-	-	[15:8]	-	-	-

Table 5-53 Big-endian read, 32-bit external bus

Internal transfer width	Access: Read, Big-endian, 32-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[0]	nMPMCBLSOUT[3:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (2 transfers)	011 011	--- ---	1 1	0000 0000	[31:24] -	[23:16] -	[15:8] -	[7:0] -	- [31:24]	- [23:16]	- [15:8]	- [7:0]
Word	010	1--	1	0000	[31:24]	[23:16]	[15:8]	[7:0]	-	-	-	-
Word	010	0--	0	0000	-	-	-	-	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	1	1100	-	-	[15:8]	[7:0]	-	-	-	-

Table 5-53 Big-endian read, 32-bit external bus (continued)

Internal transfer width	Access: Read, Big-endian, 32-bit external bus				External data mapping on to system data bus HRDATA to MPMCDATA							
	HSIZE [2:0]	HADDR [2:0]	MPMCADDROUT[0]	nMPMCBLSOUT[3:0]	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	10-	1	0011	[31:24]	[23:16]	-	-	-	-	-	-
Halfword	001	01-	0	1100	-	-	-	-	-	-	[15:8]	[7:0]
Halfword	001	00-	0	0011	-	-	-	-	[31:24]	[23:16]	-	-
Byte	000	111	1	1110	-	-	-	[7:0]	-	-	-	-
Byte	000	110	1	1101	-	-	[15:8]	-	-	-	-	-
Byte	000	101	1	1011	-	[23:16]	-	-	-	-	-	-
Byte	000	100	1	0111	[31:24]	-	-	-	-	-	-	-
Byte	000	011	0	1110	-	-	-	-	-	-	-	[7:0]
Byte	000	010	0	1101	-	-	-	-	-	-	[15:8]	-
Byte	000	001	0	1011	-	-	-	-	-	[23:16]	-	-
Byte	000	000	0	0111	-	-	-	-	[31:24]	-	-	-

Table 5-54 Big-endian write, 8-bit external bus

Internal transfer width	Access: Write, little-endian, 8-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[2:0]	nMPMCBLS OUT[0]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (8 transfers)	011 011 011 011 011 011 011 011	--- --- --- --- --- --- --- ---	111 110 101 100 011 010 001 000	0 0 0 0 0 0 0 0	- - - - - - - -	- - - - - - - -	- - - - - - - -	[39:32] [47:40] [55:48] [63:56] [7:0] [15:8] [23:16] [31:24]
Word (4 transfers)	010 010 010 010	1-- 1-- 1-- 1--	111 110 101 100	0 0 0 0	- - - -	- - - -	- - - -	[39:32] [47:40] [55:48] [63:56]
Word (4 transfers)	010 010 010 010	0-- 0-- 0-- 0--	011 010 001 000	0 0 0 0	- - - -	- - - -	- - - -	[7:0] [15:8] [23:16] [31:24]
Halfword (2 transfers)	001 001	11- 11-	111 110	0 0	- -	- -	- -	[39:32] [47:40]

Table 5-54 Big-endian write, 8-bit external bus (continued)

Internal transfer width	Access: Write, little-endian, 8-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[2:0]	nMPMCBLS OUT[0]	[31:24]	[23:16]	[15:8]	[7:0]
Halfword (2 transfers)	001 001	10- 10-	101 100	0 0	- -	- -	- -	[55:48] [63:56]
Halfword (2 transfers)	001 001	01- 01-	011 010	0 0	- -	- -	- -	[7:0] [15:8]
Halfword (2 transfers)	001 001	00- 00-	001 000	0 0	- -	- -	- -	[23:16] [31:24]
Byte	000	111	111	0	-	-	-	[39:32]
Byte	000	110	110	0	-	-	-	[47:40]
Byte	000	101	101	0	-	-	-	[55:48]
Byte	000	100	100	0	-	-	-	[63:56]
Byte	000	011	011	0	-	-	-	[7:0]
Byte	000	010	010	0	-	-	-	[15:8]
Byte	000	001	000	0	-	-	-	[23:16]
Byte	000	000	000	0	-	-	-	[31:24]

Table 5-55 Big-endian write, 16-bit external bus

Internal transfer width	Access: Write, big-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[1:0]	nMPMCBLS OUT[1:0]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (4 transfers)	011 011 011 011	--- --- --- ---	11 10 01 00	00 00 00 00	- - - -	- - - -	[47:40] [63:56] [15:8] [31:24]	[39:32] [55:48] [7:0] [23:16]
Word (2 transfers)	010 010	1-- 1--	11 10	00 00	- -	- -	[47:40] [63:56]	[39:32] [55:48]
Word (2 transfers)	010 010	0-- 0--	01 00	00 00	- -	- -	[15:8] [31:24]	[7:0] [23:16]
Halfword	001	11-	11	00	-	-	[47:40]	[39:32]
Halfword	001	10-	10	00	-	-	[63:56]	[55:48]
Halfword	001	01-	01	00	-	-	[15:8]	[7:0]
Halfword	001	00-	00	00	-	-	[31:24]	[23:16]
Byte	000	111	11	10	-	-	-	[39:32]
Byte	000	110	11	01	-	-	[47:40]	-
Byte	000	101	10	10	-	-	-	[55:48]

Table 5-55 Big-endian write, 16-bit external bus (continued)

Internal transfer width	Access: Write, big-endian, 16-bit external bus				System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	MPMCADDR OUT[1:0]	nMPMCBLS OUT[1:0]	[31:24]	[23:16]	[15:8]	[7:0]
Byte	000	100	10	01	-	-	[63:56]	-
Byte	000	011	01	10	-	-	-	[7:0]
Byte	000	010	01	01	-	-	[15:8]	-
Byte	000	001	00	10	-	-	-	[23:16]
Byte	000	000	00	01	-	-	[31:24]	-

Table 5-56 Big-endian write, 32-bit external bus

Internal transfer width	Access: Write, big-endian, 32-bit external bus			System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	nMPMCBLS OUT[3:0]	[31:24]	[23:16]	[15:8]	[7:0]
Doubleword (2 transfers)	011 011	--- ---	0000 0000	[63:56] [47:40]	[55:48] [39:32]	[31:24] [15:8]	[23:16] [7:0]
Word	010	1--	0000	[63:56]	[55:48]	[47:40]	[39:32]
Word	010	0--	0000	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	1100	-	-	[47:40]	[39:32]

Table 5-56 Big-endian write, 32-bit external bus (continued)

Internal transfer width	Access: Write, big-endian, 32-bit external bus			System data mapping on to external data bus MPMCDATA to HRDATA			
	HSIZE [2:0]	HADDR [2:0]	nMPMCBLS OUT[3:0]	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	10-	0011	[63:56]	[55:48]	-	-
Halfword	001	01-	1100	-	-	[15:8]	[7:0]
Halfword	001	00-	0011	[31:24]	[23:16]	-	-
Byte	000	111	1110	-	-	-	[39:32]
Byte	000	110	1101	-	-	[47:40]	-
Byte	000	101	1011	-	[55:48]	-	-
Byte	000	100	0111	[63:56]	-	-	-
Byte	000	011	1110	-	-	-	[7:0]
Byte	000	010	1101	-	-	[15:8]	-
Byte	000	001	1011	-	[23:16]	-	-
Byte	000	000	0111	[31:24]	-	-	-

Chapter 6
NAND Flash Memory Controller

This chapter describes the NAND flash memory controller interface implemented in the ARM PrimeCell MPMC (PL176). It contains the following sections:

6.1 About the NAND flash memory controller

NAND flash devices do not have a standard SRAM interface. A common command/address/data bus is used, with chip select and read, write, command, and address enable signals.

A register-based interface is provided to control NAND flash transfers, because it is not possible to directly perform simple read and write transfers as it is for SRAM.

NAND transfers are based on several separate phases:

- command
- address
- data
- busy.

Different types of transfer use different sequences of the above phases, and the same command on different devices can also use a different sequence. For example, large devices often require more address phases than small devices.

For this reason, the interface is split between hardware and software control, enabling maximum transfer flexibility and future transfer compatibility, with the minimum hardware required to perform efficient transfers. The hardware provides the physical interface to the NAND flash memory, and also controls the operation of transfers, applying the correct timing to the control signals. The software provides the command instruction values, address, and data values, and also configures the sequence of phases performed by the hardware to generate the correct transfer.

6.2 Initialization

Before any NAND flash transfers are performed, you must configure a set of timing registers. These are used to control the timing of the NAND flash control signals during the transfer. The default values for all of the timing registers are the maximum possible, but the majority of devices can to operate with faster timing. The timing registers apply to all NAND devices in the system, so the slowest timing values must be programmed to ensure that all devices in the system are accessed correctly. If the devices have very different timing requirements, the timing registers can be reprogrammed between accesses to different devices.

To identify which chip selects are connected to NAND flash devices, the ND bit of the MPMCStaticConfig Register must be set accordingly for memory banks 0-3.

The other bits of the MPMCStaticConfig Registers must be set according to the NAND flash devices used. The PM, PB, and EW bits are not applicable to NAND flash.

Before each external NAND flash transfer is performed, the control and address registers must be set up for the transfer.

The MPMCNANDControl Register is used to define the sequence of phases that are performed during the transfer. There is always a command phase at the start of the transfer to indicate to the memory what type of transfer is being performed. This is followed by a combination of the optional address, data, and second command phases, depending on the transfer type and device being accessed.

For example, a simple data read transfer is:

- one command phase
- three address phases
- a busy phase where the data is moved from the memory to the NAND flash device output data registers
- several data phases depending on the amount of data being read.

The NDCMDV1/2 bits of the MPMCNANDControl Register are used to store the two 8-bit command vectors. The first vector value must always be programmed, but the second value is only used if the NDCMDPH2 bit is set, indicating that a second command phase is performed during the transfer.

There is only one set of control and address registers used to drive all connected NAND flash devices, so the NDCS bits are used to select which chip select is used for the programmed transfer. This ensures that the transfer is only performed to the correct device.

The NDADDRPH bits define whether there is an address phase, and how many address vectors are performed during the address phase. The maximum number of address vectors allowed is 5. Each section of the MPMCNANDAddress1/2 Registers must be programmed with the address vectors required for the transfer. Unused address vectors do not require their values changing.

The NDDATAPH bit defines whether a data phase is performed. The number of data transfers is not defined, because AHB read/write transfers are converted into external read/write transfers.

The NDIDRD bit indicates that the current transfer is an ID read. This is required because of the special timing requirements of ID read transfers, which are performed differently to standard data reads.

The NDSHORTRD bit indicates that a read transfer does not have to check the status of the ready/busy output before performing the data phase of the transfer. The programmed t_{CLR} value is used to control the timing of the read transfer.

The NDTXRW bit indicates the type of transfer being performed, either a read or a write. A transfer that performs a data read must be programmed as a read. A transfer that performs a data write or has no data phase, such as a block erase, must be programmed as a write.

6.3 Performing NAND flash transfers

When the control and address registers have been programmed with the relevant values, the NAND flash transfer can be performed. NAND flash transfers are described in the following sections:

- *NAND flash transfer control*
- *NAND flash transfer types* on page 6-7
- *Accessing other memory during NAND flash transfers* on page 6-8
- *NAND flash transfer error responses* on page 6-8.

6.3.1 NAND flash transfer control

NAND flash transfers are controlled with read or write transfers to the chip select programmed into the NDCS bits of the MPMCNANDControl Register. An access to a different chip select does not perform a NAND access. The lower bits of the address used for the NAND transfer control the operation performed, as shown in Table 6-1.

Table 6-1 NAND flash transfer addresses

Transfer	HADDR[11:0]	Description
Start	`0x000`	Initiate transfer, or start new command and address phase
Command-2	`0x008`	Ends the current write data phase, performing a second command phase if required
Stop	`0x010`	Ends the current transfer, performing a second command phase if required
Read/Write	`0x011` and higher	Read or write data transfers

The address value used for the transfer is unrelated to the NAND device address, because this is already programmed into the MPMCNANDAddress1/2 Registers. When it is granted access to the external bus, the internal control state machine drives the control signals to generate the NAND flash transfer. Read and write data is buffered in both directions, depending on the sizes of the AHB transfers and the NAND flash devices used, in the same way as for normal SRAM transfers.

Transfers to the Start, Command-2, and Stop addresses can be performed using either AHB reads or writes. A read returns undefined data which must be discarded. The data value of a write transfer is not used by the memory controller.

The Start address is used to initiate the external NAND transfer. The programmed control and address phases are performed up to, but not including, the first data phase. The value of the NDTXRW bit determines if the transfer is a read or a write.

Note

A NAND transfer with no data phase must be initiated with a single AHB write of any value to the required chip select Start address. Initiating a transfer with no data phase using an AHB read can result in unpredictable behavior.

Performing a second access to the Start address enables NAND transfers that require multiple sets of control and address phases to be performed, for example a copy-back program. If required, the status of the ready/busy output from the NAND flash device must be checked before performing a second access to the Start address, because the hardware does not check its status. This enables support for NAND flash devices which allow transfers such as status reads to be performed during a page program or erase while the ready/busy output is asserted.

Note

Write transfers with multiple parts where a data phase is not followed by a program command (for example, the first sequence in a Page Program Operation with Random Data Input transfer) must perform a Command-2 transfer after the data phase before reprogramming the control registers and performing the Start transfer for the next sequence. The NDCMDPH2 bit of the MPMCNANDControl Register must be cleared so that a second command is not performed.

The Command-2 address is used to insert a second command phase at the end of a write data phase when the transfer is not ended, and further control/address phases are performed, for example, during a cache program transfer. This command has no effect during a read transfer, because the second command phase is performed before the data phase.

The Stop address is used to end the current transfer. The MPMC buffers a Stop request until the transfer can be safely ended, after any data transfers have been performed. The data value for the transfer is not used, so the data phase must be completed before performing a Stop access. During a write transfer, a second command phase is performed if the NDCMDPH2 bit of the MPMCNANDControl Register is set.

Normal data phase read and write transfers are performed to an address which is `0x011` or greater. This enables reads and writes to be performed to static, random, or incrementing addresses. During a NAND read transfer, only AHB reads can be performed. During a NAND write transfer, both AHB reads and writes can be performed, enabling a status read to be performed without deasserting the chip select to perform a separate read transfer.

During data writes, the 64-bit write buffer is emptied at the end of each AHB burst, when a nonsequential or idle transfer is received, or when it is full.

Some NAND transfers require multiple command and address phases to be performed while the chip select is held asserted. This is possible through accessing the Start address each time a new set of command and address phases is to be performed, but the control and address registers must be updated with new values for the subsequent phases, and the status of the ready/busy output must be checked if required. The NDINTFBUSY bit of the MPMCNANDStatus Register determines when it is safe to write new values to the control and address registers. When this bit is sampled HIGH, it indicates that the NAND interface is busy and the registers cannot be updated because they might be in use. When this bit is sampled LOW, new values can be written to the control and address registers.

——— **Note** ———

Only aligned AHB transfers must be used when performing NAND reads and writes. Because the address of the AHB transfer is not related to the address of the NAND transfer, it is not possible for the unaligned data to be correctly applied to the NAND device. Performing unaligned accesses to NAND flash can result in unpredictable behavior.

Only AHB transfers with a size equal to or greater than the NAND flash memory size are allowed, for both reads and writes. That is, accesses to 16-bit NAND flash devices must be halfword or greater, because byte accesses can result in unpredictable behavior.

Sequential row read transfers insert a busy phase between rows. This must be checked by software using the NAND flash status register, because the hardware is unable to detect when the last read of a row is performed.

After programming the MPMCNANDControl Register, a minimum of three clock cycles must be allowed before issuing a Start command, to enable the new register values to propagate through the memory controller. Programming this register before the MPMCNANDAddress1/2 Registers enables this small delay to be hidden.

6.3.2 NAND flash transfer types

The exact ordering of the NAND flash transfer depends on the type of AHB transfer used to initiate it and the settings in the control register:

- a transfer with no address phase and a data phase is assumed to be a status read, which does not use the ready output to control the timing of the data phase, so t_{CLR} is used
- a read transfer with a second command phase and data phase performs the second command phase before the data phase

- a write transfer with a second command phase and data phase performs the second command phase after the data phase
- all reads other than status and ID reads wait for the ready output to be deasserted and asserted before reading back any data if the NDSHORTRD bit of the MPMCNANDControl Register is cleared
- all writes that have received a Stop address wait for the ready output to be deasserted after all data and command phases before ending the transfer and deasserting the chip select to the device.

6.3.3 Accessing other memory during NAND flash transfers

Because of the long access times of NAND devices when compared to SRAM and SDRAM devices, the memory interface enables accesses to other memory devices to be performed during idle times in a NAND transfer. For example, this enables instruction or data fetches to be performed during a long page read/write operation to the NAND device, or other system activity to take place during a long page program operation.

Note

Only one NAND device can be accessed at a time.

When the NDINTFBUSY bit of the MPMCNANDStatus Register indicates the interface is idle, the internal arbitration enables SRAM or SDRAM accesses to be performed if any are pending. The normal arbitration scheme applies, and the NAND access can continue when the AHB port performing the NAND access is granted control of the external memory. The NAND control signals and the device chip select are held stable throughout the accesses to other memory devices. The NDTX bit of the MPMCNANDStatus Register indicates that a NAND transfer is still in progress.

Note

Only a single master can access the NAND device at a time. Check the NDTX status bit to determine when the NAND interface is idle. Simultaneous NAND accesses by multiple AHB masters can result in unpredictable behavior.

6.3.4 NAND flash transfer error responses

The following transfers generate AHB error responses:

- An access to a chip select set as NAND flash in the MPMCStaticConfig Register, but which is not programmed into the NDCS bits of the MPMCNANDControl Register. This indicates that the control register has not been correctly configured to perform an access to the current chip select.

- A write to the control or address registers when the NDINTFBUSY bit is HIGH, indicating that the NAND interface is busy so the control and address registers might be in use and cannot be modified. The status of the NDINTFBUSY bit must be checked before writing to the control or address registers.
- A write to the timing registers when the NDTX bit is HIGH, indicating that a NAND transfer is currently in progress, so the timing parameters are in use. The status of the NDTX bit must be checked before writing to the timing registers.

6.4 Performance implications

The timing of NAND flash devices is generally much slower than SRAM and SDRAM. This means that NAND flash transfers use the external bus for a relatively long time, especially during the busy periods of the device for a read, program, or erase operation which range from microseconds to milliseconds.The NAND flash interface is designed to enable SRAM and SDRAM accesses to be performed during the idle periods in a NAND flash transfer, but there might still be large delays while the command and address phases of a NAND access are being performed. SDRAM refreshes are not affected by NAND flash accesses, and are always performed when required.The system designer must be aware of the implications of NAND flash accesses in systems that also contain high-speed SRAM or SDRAM, and devices that make regular use of the external bus for data transfers.

6.5 Example transfers

Two example transfers are described in this section:

- *Samsung K9F5608U0A read*
- *Samsung K9K2G08QOM page program* on page 6-12.

6.5.1 Samsung K9F5608U0A read

This device uses a read sequence consisting of the following:

- command phase
- three address phases
- a busy phase
- several data read phases.

The following must be performed for a read transfer:

1. After reset, the MPMCStaticConfig Register is configured for the chip select to which the NAND flash device is connected:
    - MW is set to 8-bit
    - PM is left disabled
    - ND is set to NAND flash
    - PC is set to active LOW
    - PB and EW are left deasserted
    - P is set according to the device requirements.
2. Check the status of the NDTX bit to ensure that the NAND interface is not currently performing a transfer.
3. Initialize the MPMCNANDTiming1/2 Registers with the correct values for the device, or values to match the slowest NAND flash device in the system.
4. Initialize the command register for the read transfer:
    - set NDCMDV1 to `0x00` or `0x01` for the Read 1 command, depending on the address used being in the first or second half of the registers
    - leave NDCMDV2 at the previous value because it is not used
    - set NDCS to match the chip select of the NAND flash device
    - set NDADDRPH to `0x3`, because three address vectors are required
    - set NDDATAPH to one because there is a data phase in the transfer
    - set NDCMDPH2 to zero because there is no second command phase

    - set NDIDRD and NDSHORTRD to zero because this is a standard data read that uses the ready/busy output to control the read data timing
    - set NDTXRW to zero because this is a read transfer.

5. Initialize the NDADDRV1-3 sections of the MPMCNANDAddress1 Register with the three address values required for the transfer. The NDADDRV4-5 sections can remain with their previous values because they are not used.

6. Perform a single transfer to the chip select connected to the NAND flash device, with the lower address bits set to the Start value. This NAND flash interface waits to be granted control of the external bus, performs the command and address phases, and waits for the read data to be copied to the output registers as indicated by the ready/busy output of the memory device. When the device busy phase has completed, it is ready to perform a data transfer.

7. Perform as many reads as required, up to a maximum of 528 bytes for the full page.

8. Perform a single transfer to the Stop location on the NAND flash chip select to indicate that the read transfer has completed. The control signals are then deasserted, including the chip select, and the NDTX bit is cleared to indicate that no NAND flash transfers are being performed.

6.5.2 Samsung K9K2G08QOM page program

This device uses a page program sequence consisting of the following:

- command phase
- five address phases
- a second command phase
- a busy phase
- several data write phases.

The following must be performed for a page program transfer:

1. After reset, the MPMCStaticConfig Register is configured for the chip select to which the NAND flash device is connected:
    - MW is set to 8 or 16-bit depending on the device used
    - PM is left disabled
    - ND is set to NAND flash
    - PC is set to active LOW
    - PB and EW are left deasserted
    - P is set to not protected to enable the data write to be performed.

2. Check the status of the NDTX bit to ensure that the NAND interface is not currently performing a transfer.
3. Initialize the MPMCNANDTiming1/2 Registers with the correct values for the device, or values to match the slowest NAND flash device in the system.
4. Initialize the command register for the read transfer:
    - set NDCMDV1 to `0x80` for the serial data input command
    - set NDCMDV2 to `0x10` for the program command
    - set NDCS to match the chip select of the NAND flash device
    - set NDADDRPH to `0x5`, because five address vectors are required
    - set NDDATAPH to one because there is a data phase in the transfer
    - set NDCMDPH2 to one because there is a second command phase
    - set NDIDRD and NDSHORTRD to zero because this is a write transfer
    - set NDTXRW to one because this is a write transfer.
5. Initialize the NDADDRV1-5 sections of the MPMCNANDAddress1-2 Registers with the five address values required for the transfer.
6. Perform a single transfer to the chip select connected to the NAND flash device, with the lower address bits set to the Start value. This is delayed for several cycles while the NAND flash interface is granted control of the external bus, and performs the command and address phases.
7. Perform as many writes as required, up to a maximum of 2112 bytes for the full page. For a 16-bit device, an even number of bytes must be written to ensure that the last write performed is a full 16 bits of valid data.
8. If a status read is performed, continue with step 9. If the transfer is completed without a status read, perform a single transfer to the Stop location on the NAND flash chip select to indicate that the page program can be completed. The NAND flash interface then performs the second command phase, waits for the ready/busy output to indicate that the page programming has begun, and then ends the transfer. The control signals are then deasserted, including the chip select, and the NDTX bit is cleared to indicate that no NAND flash transfers are being performed. Steps 9 to 13 indicate the sequence required to perform a status read without ending the transfer.
9. For a status read, write any value to the Command-2 location on the NAND flash chip select to indicate that the page programming can begin, but the transfer is not finished. The NAND flash interface then performs the second command phase, and waits for the ready/busy output to indicate that the page programming has begun before clearing the NDINTFBUSY bit.

10. After checking the status of the NDINTFBUSY bit to ensure that the NAND interface is idle, reprogram the command register for the status read:
    - NDCMDV1 is set to `0x70` for the read status command
    - NDCMDV2 is left at the previous value because it is not used
    - NDCS is set to match the chip select of the NAND flash device
    - NDADDRPH is set to `0x0`, because no address vectors are required
    - NDDATAPH is set to one because there is a data phase in the transfer
    - NDCMDPH2 is set to zero because there is no second command phase
    - NDIDRD is set to zero because this is a status read which is different to an ID read
    - NDSHORTRD is set to zero, because a status read is automatically detected
    - NDTXRW is set to zero because this is a read transfer.
11. Poll the ready status bit NDRDYCS for the NAND flash device currently being accessed to determine when the page program has completed. When the device is no longer busy, perform a single transfer to the device chip select with the lower address bits set to the Start value. No busy cycle is inserted by the memory device during a status read.
12. Perform a single data read transfer to sample the status read value.
13. Perform a single transfer to the Stop location on the NAND flash chip select to indicate that the read transfer has completed. The control signals are then deasserted, including the chip select, and the NDTX bit is cleared to indicate that no NAND flash transfers are being performed.

Chapter 7
Dynamic Memory Controller

This chapter describes the *Dynamic Memory Controller* (DMC). It contains the following sections:

7.1 Write-protection

Each dynamic memory chip select can be configured for write-protection by setting the relevant bit in the write-protect (P) field in the MPMCDynamicConfig Register. If a write access is performed to a write-protected memory bank, an ERROR response is generated on the **HRESP[0]** signal.

7.2 Access sequencing and memory width

The data width of each external memory bank must be configured by programming the appropriate MPMCDynamicConfig0-3 Registers. When the external memory bus is narrower than the transfer initiated from the current AMBA bus master, the internal bus transfer takes several external bus transfers to complete. For example, if bank 4 is configured as 16-bit wide memory and a 64-bit read is initiated, the AHB bus stalls while the PrimeCell MPMC reads four consecutive 16-bit halfwords from the memory. During these accesses the memory controller block demultiplexes the four 16-bit halfwords into one 64-bit doubleword and places the result in its read cache, if enabled, and also onto the requesting AHB bus.

7.3 Arbitration

The PrimeCell MPMC AHB memory port arbitration strategy is described in the following sections:

- *Re-arbitration occurrence*
- *Re-arbitration priority*.

7.3.1 Re-arbitration occurrence

Re-arbitration always takes place whenever an AHB memory port requests access to external memory, except in the following circumstances:

- When a locked transfer has started, as defined by the AHB **HMASTLOCK** signal. The AHB memory port is not re-arbitrated until the AHB **HMASTLOCK** signal is deasserted.
- When a fixed length AHB burst transfer has started, the AHB memory port is not re-arbitrated until the burst has completed.

The longest AHB transfer is 16 AHB transfers, INCR16, or WRAP16.

Note

INCR transfers are interpreted as INCR4 transfers by the memory controller.

7.3.2 Re-arbitration priority

The following re-arbitration rules apply:

- The highest priority AHB memory port is a port where the Time Out register has timed out.

  If more than one port has timed out, the port with the lowest number is selected, for example, port 0 is selected over port 1.
- If none of the AHB memory ports have timed out, the highest priority port is the port where the transaction is to an already open SDRAM memory row.

  If more than one port has a transaction to an already open SDRAM memory row, the port with the lowest number is selected.
- If none of the AHB memory ports have timed out and none of the transactions are to open SDRAM memory rows, the port with the lowest number is selected.

7.4 Dynamic memory transaction latency

The ARM PrimeCell MPMC (PL176) transfer latencies are shown in Table 7-1 to Table 7-3 on page 7-7. The values assume CAS latency 2 SDRAM, where CAS, RAS and Precharge (t_{RP}) are two clock cycles. The AHB latency heading shows the latency seen by the AHB interface, for the case where the request is dealt with immediately. Memory latency indicates the amount of time it takes for the AHB request to go active until the memory transfer has completed at the SDRAM.

7.4.1 Latency tables

Table 7-1 shows the 64-bit SDRAM transfer latencies.

Table 7-1 64-bit chip select width transfer latencies

Access type	Burst type	SDRAM commands	AHB latency	Memory latency
1 x 64-bit read, page open	SINGLE	Read	9	9
1 x 64-bit read, bank precharged	SINGLE	Active, Read	11	11
1 x 64-bit read, incorrect page open	SINGLE	Precharge, Active, Read	13	13
1 x 64-bit write, page open	SINGLE	Write	1	5
1 x 64-bit write, bank precharged	SINGLE	Active, Write	1	7
1 x 64-bit write, incorrect page open	SINGLE	Precharge, Active, Read	1	9
4 x 64-bit read, page open	INCR, INCR4, WRAP4	Read	12	12
4 x 64-bit read, bank precharged	INCR, INCR4, WRAP4	Active, Read	14	14
4 x 64-bit read, incorrect page open	INCR, INCR4, WRAP4	Precharge, Active, Read	16	16
4 x 64-bit write, page open	INCR, INCR4, WRAP4	Write	4	8
4 x 64-bit write, bank precharged	INCR, INCR4, WRAP4	Active, Write	4	10
4 x 64-bit write, incorrect page open	INCR, INCR4, WRAP4	Precharge, Active, Write	4	12
8 x 64-bit read, page open	INCR8, WRAP8	Read	16	16
8 x 64-bit read, bank precharged	INCR8, WRAP8	Active, Read	18	18
8 x 64-bit read, incorrect page open	INCR8, WRAP8	Precharge, Active, Read	20	20
8 x 64-bit write, page open	INCR8, WRAP8	Write	8	12

Table 7-1 64-bit chip select width transfer latencies (continued)

Access type	Burst type	SDRAM commands	AHB latency	Memory latency
8 x 64-bit write, bank precharged	INCR8, WRAP8	Active, Write	14	18
8 x 64-bit write, incorrect page open	INCR8, WRAP8	Precharge, Active, Write	16	20
16 x 64-bit read, page open	INCR16, WRAP16	Read	24	24
16 x 64-bit read, bank precharged	INCR16, WRAP16	Active, Read	26	26
16 x 64-bit read, incorrect page open	INCR16, WRAP16	Precharge, Active, Read	28	28
16 x 64-bit write, page open	INCR16, WRAP16	Write	16	20
16 x 64-bit write, bank precharged	INCR16, WRAP16	Active, Write	30	34
16 x 64-bit write, incorrect page open	INCR16, WRAP16	Precharge, Active, Write	32	36

Table 7-2 shows the 32-bit SDRAM transfer latencies.

Table 7-2 32-bit SDRAM transfer latencies

Access type	Burst type	SDRAM commands	AHB latency	Memory latency
1 x 64-bit read, page open	SINGLE	Read	10	10
1 x 64-bit read, bank precharged	SINGLE	Active, Read	12	12
1 x 64-bit read, incorrect page open	SINGLE	Precharge, Active, Read	14	14
1 x 64-bit write, page open	SINGLE	Write	1	6
1 x 64-bit write, bank precharged	SINGLE	Active, Write	1	8
1 x 64-bit write, incorrect page open	SINGLE	Precharge, Active, Read	1	10
4 x 64-bit read, page open	INCR, INCR4, WRAP4	Read	16	16
4 x 64-bit read, bank precharged	INCR, INCR4, WRAP4	Active, Read	18	18
4 x 64-bit read, incorrect page open	INCR, INCR4, WRAP4	Precharge, Active, Read	20	20
4 x 64-bit write, page open	INCR, INCR4, WRAP4	Write	4	12
4 x 64-bit write, bank precharged	INCR, INCR4, WRAP4	Active, Write	4	14
4 x 64-bit write, incorrect page open	INCR, INCR4, WRAP4	Precharge, Active, Write	4	16

Table 7-2 32-bit SDRAM transfer latencies (continued)

Access type	Burst type	SDRAM commands	AHB latency	Memory latency
8 x 64-bit read, page open	INCR8, WRAP8	Read	24	24
8 x 64-bit read, bank precharged	INCR8, WRAP8	Active, Read	26	26
8 x 64-bit read, incorrect page open	INCR8, WRAP8	Precharge, Active, Read	28	28
8 x 64-bit write, page open	INCR8, WRAP8	Write	8	20
8 x 64-bit write, bank precharged	INCR8, WRAP8	Active, Write	14	22
8 x 64-bit write, incorrect page open	INCR8, WRAP8	Precharge, Active, Write	16	24
16 x 64-bit read, page open	INCR16, WRAP16	Read	40	40
16 x 64-bit read, bank precharged	INCR16, WRAP16	Active, Read	42	42
16 x 64-bit read, incorrect page open	INCR16, WRAP16	Precharge, Active, Read	44	44
16 x 64-bit write, page open	INCR16, WRAP16	Write	17	36
16 x 64-bit write, bank precharged	INCR16, WRAP16	Active, Write	30	38
16 x 64-bit write, incorrect page open	INCR16, WRAP16	Precharge, Active, Write	32	40

Table 7-3 shows the 32-bit DDR transfer latencies.

Table 7-3 32-bit DDR transfer latencies

Access type	Burst type	SDRAM commands	AHB latency	Memory latency
1 x 64-bit read, page open	SINGLE	Read	10	10
1 x 64-bit read, bank precharged	SINGLE	Active, Read	12	12
1 x 64-bit read, incorrect page open	SINGLE	Precharge, Active, Read	14	14
1 x 64-bit write, page open	SINGLE	Write	1	7
1 x 64-bit write, bank precharged	SINGLE	Active, Write	1	9
1 x 64-bit write, incorrect page open	SINGLE	Precharge, Active, Read	1	11
4 x 64-bit read, page open	INCR, INCR4, WRAP4	Read	13	13
4 x 64-bit read, bank precharged	INCR, INCR4, WRAP4	Active, Read	15	15

Table 7-3 32-bit DDR transfer latencies (continued)

Access type	Burst type	SDRAM commands	AHB latency	Memory latency
4 x 64-bit read, incorrect page open	INCR, INCR4, WRAP4	Precharge, Active, Read	17	17
4 x 64-bit write, page open	INCR, INCR4, WRAP4	Write	4	10
4 x 64-bit write, bank precharged	INCR, INCR4, WRAP4	Active, Write	4	12
4 x 64-bit write, incorrect page open	INCR, INCR4, WRAP4	Precharge, Active, Write	4	14
8 x 64-bit read, page open	INCR8, WRAP8	Read	17	17
8 x 64-bit read, bank precharged	INCR8, WRAP8	Active, Read	19	19
8 x 64-bit read, incorrect page open	INCR8, WRAP8	Precharge, Active, Read	21	21
8 x 64-bit write, page open	INCR8, WRAP8	Write	8	14
8 x 64-bit write, bank precharged	INCR8, WRAP8	Active, Write	14	20
8 x 64-bit write, incorrect page open	INCR8, WRAP8	Precharge, Active, Write	16	22
16 x 64-bit read, page open	INCR16, WRAP16	Read	25	25
16 x 64-bit read, bank precharged	INCR16, WRAP16	Active, Read	27	27
16 x 64-bit read, incorrect page open	INCR16, WRAP16	Precharge, Active, Read	29	29
16 x 64-bit write, page open	INCR16, WRAP16	Write	17	22
16 x 64-bit write, bank precharged	INCR16, WRAP16	Active, Write	30	36
16 x 64-bit write, incorrect page open	INCR16, WRAP16	Precharge, Active, Write	32	38

7.5 Transaction examples

The following section provides examples of dynamic memory transactions, and contains the following subsections:

- *Single 64-bit read from SDR-SDRAM, row open* on page 7-10
- *Single 64-bit write to SDR-SDRAM, row open* on page 7-12
- *Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open* on page 7-14
- *Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, bank precharged* on page 7-16
- *Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, different row open* on page 7-18
- *Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open* on page 7-20
- *Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, bank precharged* on page 7-22
- *Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, different row open* on page 7-24
- *Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open* on page 7-26
- *Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open* on page 7-28.

Unless explicitly specified the following transactions use the parameters in Table 7-4 when:

- no other AHB is requesting the bus
- no other memory controller is using the EBI.

Table 7-4 Transaction details

Parameter	Value
CAS latency	Two cycles
RAS latency	Two cycles

Table 7-4 Transaction details (continued)

Parameter	Value
Precharge delay (tRP)	Two cycles
Memory controller state	Idle when request goes active
Pad interface scheme	Command delayed

7.5.1 Single 64-bit read from SDR-SDRAM, row open

Figure 7-1 on page 7-11 and Table 7-5 show an example of a single 64-bit AHB read from 64-bit wide SDR-SDRAM. The row to be read from (in SDRAM) is open.

Table 7-5 Single 64-bit read from 64-bit SDR-SDRAM, row open

Cycle	Description
T0	AHB address provided to memory controller
T0-T1	AHB transaction processing
T1-T2	Arbitration of AHB memory ports
T2-T3	Memory controller processing
T3-T4	Read command submitted to SDRAM
T4-T5	SDRAM CAS access latency
T5-T6	SDRAM CAS access latency
T6-T7	Read data capture
T7-T8	Provide read data to AHB

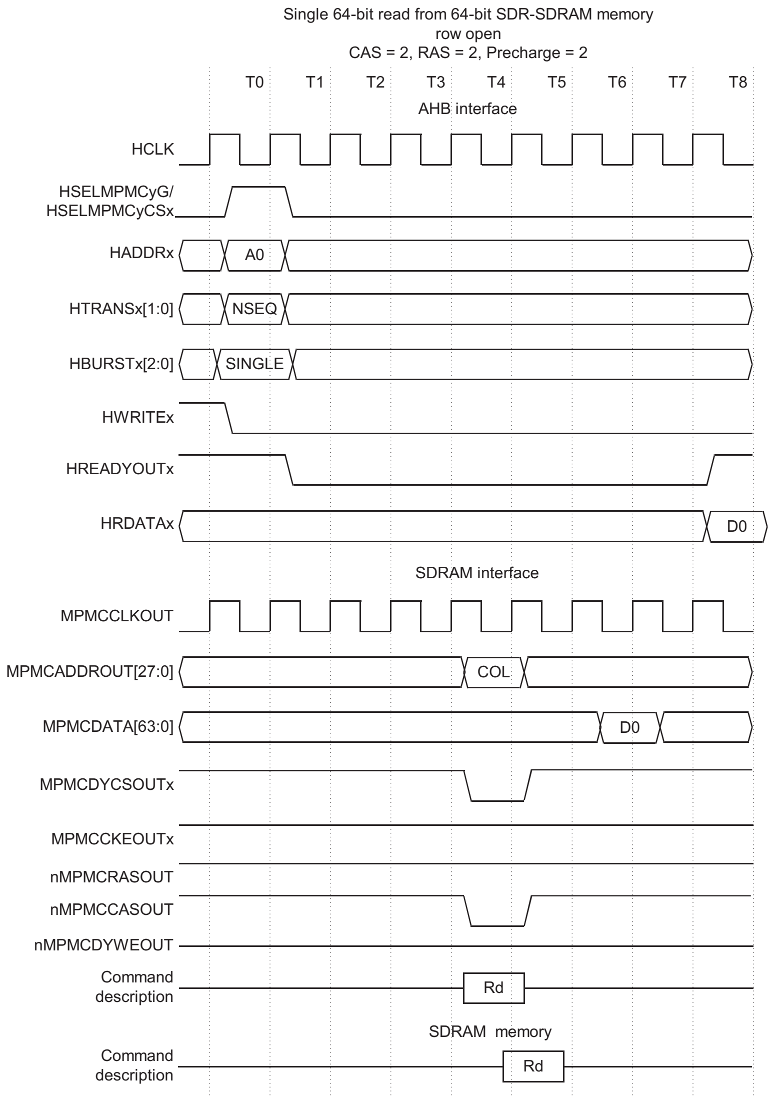


Figure 7-1 Single 64-bit read from 64-bit SDR-SDRAM, row open

7.5.2 Single 64-bit write to SDR-SDRAM, row open

Figure 7-2 on page 7-13 and Table 7-6 show an example of a single 64-bit AHB write to 64-bit wide SDR-SDRAM. The row to be written to (in SDRAM) is open.

Table 7-6 Single 64-bit write to 64-bit SDR-SDRAM, row open

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	AHB write data provided to memory controller. Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Write command submitted to SDRAM.

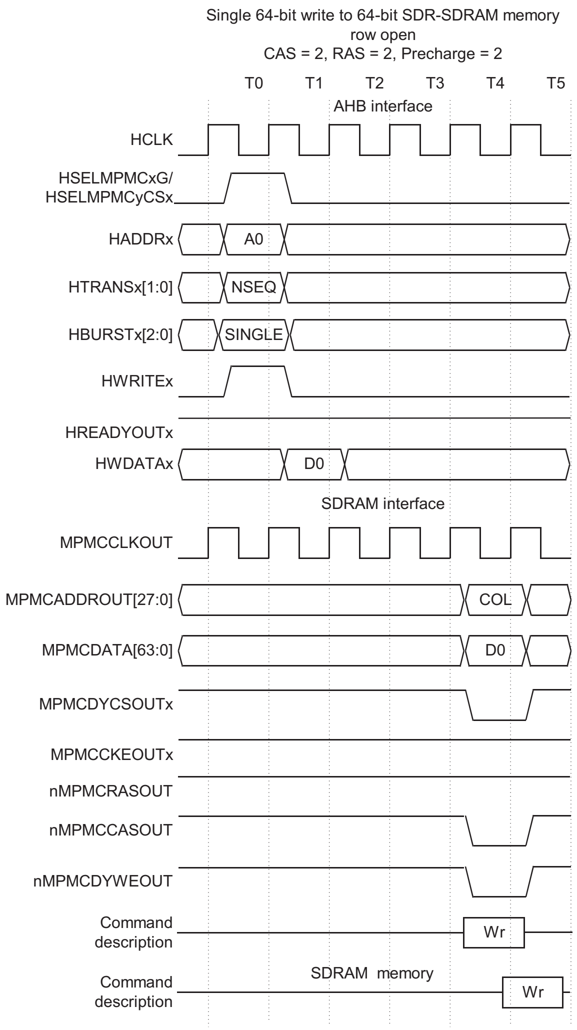


Figure 7-2 Single 64-bit write to 64-bit SDR-SDRAM, row open

7.5.3 Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open

Figure 7-3 on page 7-15 and Table 7-7 show an example of a burst 4, 64-bit AHB read from 64-bit wide SDR-SDRAM. The row to be read from (in SDRAM) is open.

Table 7-7 Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Read command submitted to SDRAM.
T4-T5	SDRAM CAS access latency.
T5-T6	SDRAM CAS access latency.
T6-T7	Read data 0 capture.
T7-T8	Read data 1 capture. Provide read data 0 to AHB.
T8-T9	Read data 2 capture. Provide read data 1 to AHB.
T9-T10	Read data 3 capture. Provide read data 2 to AHB.
T10-T11	Provide read data 3 to AHB.

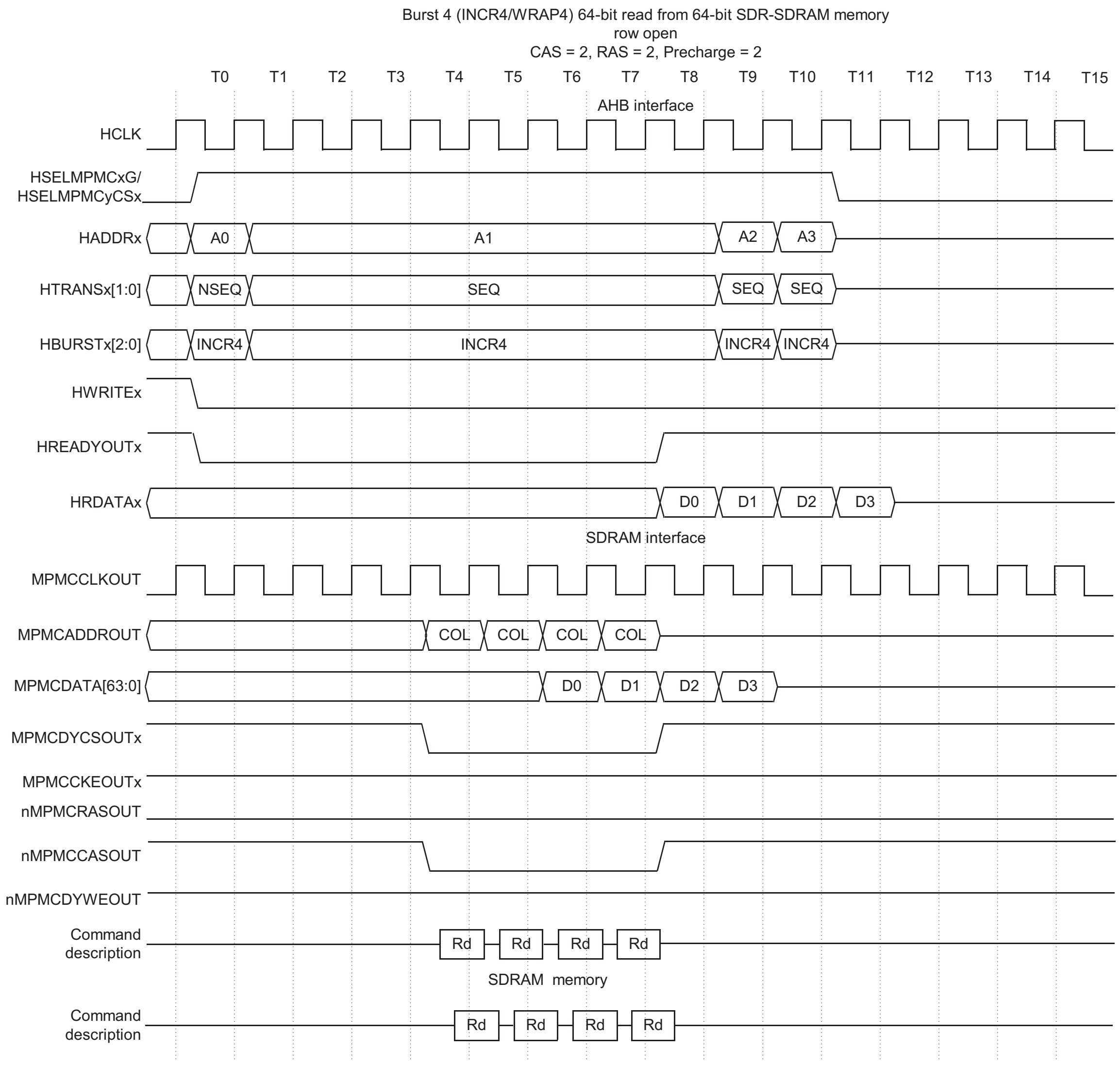


Figure 7-3 Burst 4 (INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open

7.5.4 Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, bank precharged

Figure 7-4 on page 7-17 and Table 7-8 show an example of a burst 4, 64-bit AHB read from 64-bit wide SDR-SDRAM. The row to be read from (in SDRAM) is precharged.

Table 7-8 Burst 4 (INCR/INCR4/WRAP4) 32-bit read from 32-bit SDR-SDRAM, bank precharged

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Active command submitted to SDRAM.
T4-T5	SDRAM active command latency.
T5-T6	Read command submitted to SDRAM.
T6-T7	SDRAM CAS access latency.
T7-T8	SDRAM CAS access latency.
T8-T9	Read data 0 capture.
T9-T10	Read data 1 capture. Provide read data 0 to AHB.
T10-T11	Read data 2 capture. Provide read data 1 to AHB.
T11-T12	Read data 3 capture. Provide read data 2 to AHB.
T12-T13	Provide read data 3 to AHB.

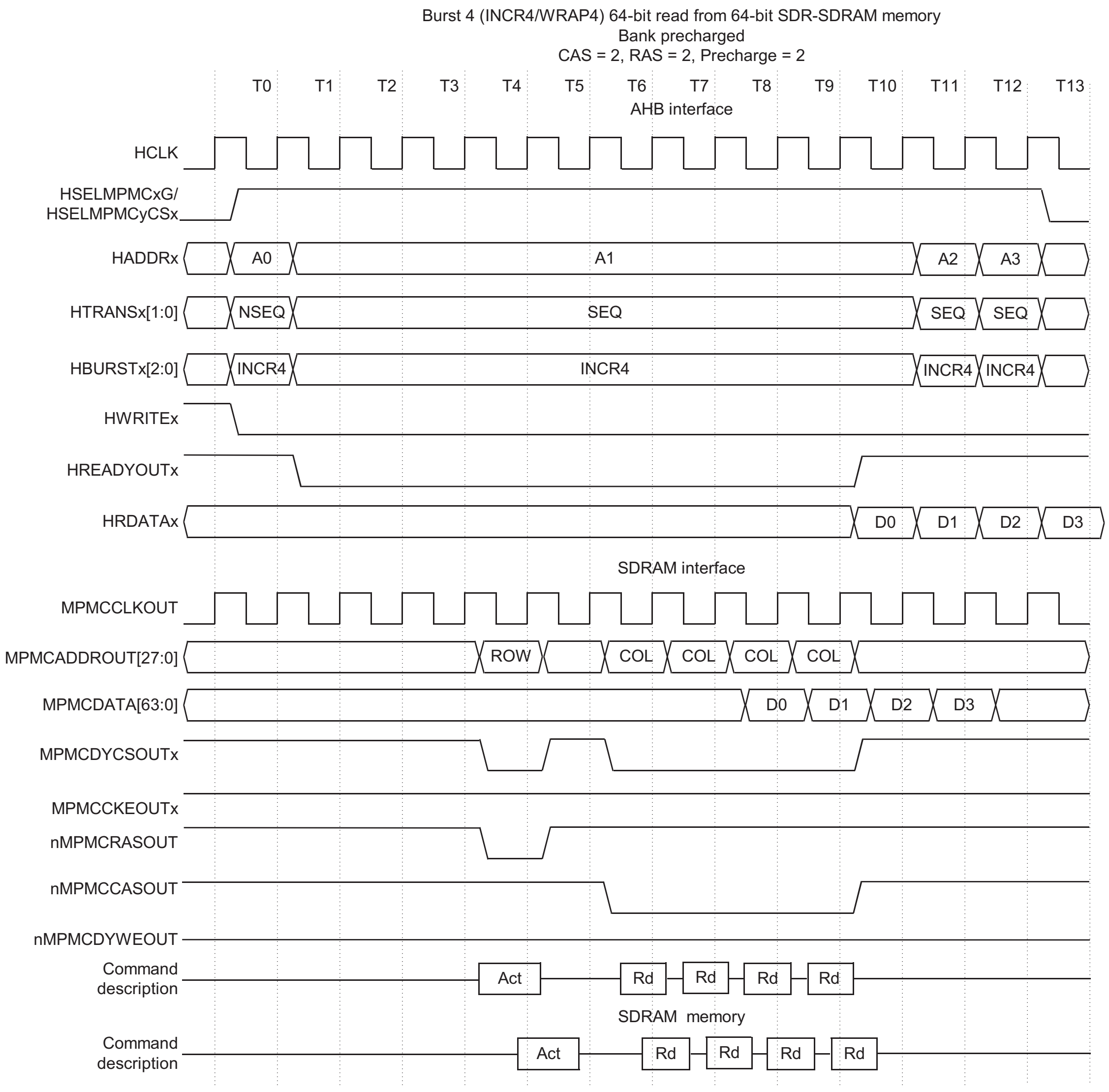


Figure 7-4 Burst 4 (INCRA4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, bank precharged

7.5.5 Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, different row open

Figure 7-5 on page 7-19 and Table 7-9 show an example of a burst 4, 64-bit AHB read from 64-bit wide SDR-SDRAM. A different row from the one that is required to be read from is open.

Table 7-9 Burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, different row open

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Precharge command submitted to SDRAM.
T4-T5	SDRAM precharge command latency.
T5-T6	Active command submitted to SDRAM.
T6-T7	SDRAM active command latency.
T7-T8	Read command submitted to SDRAM.
T8-T9	SDRAM CAS access latency.
T9-T10	SDRAM CAS access latency.
T10-T11	Read data 0 capture.
T11-T12	Read data 1 capture. Provide read data 0 to AHB.
T12-T13	Read data 2 capture. Provide read data 1 to AHB.
T13-T14	Read data 3 capture. Provide read data 2 to AHB.
T14-T15	Provide read data 3 to AHB.

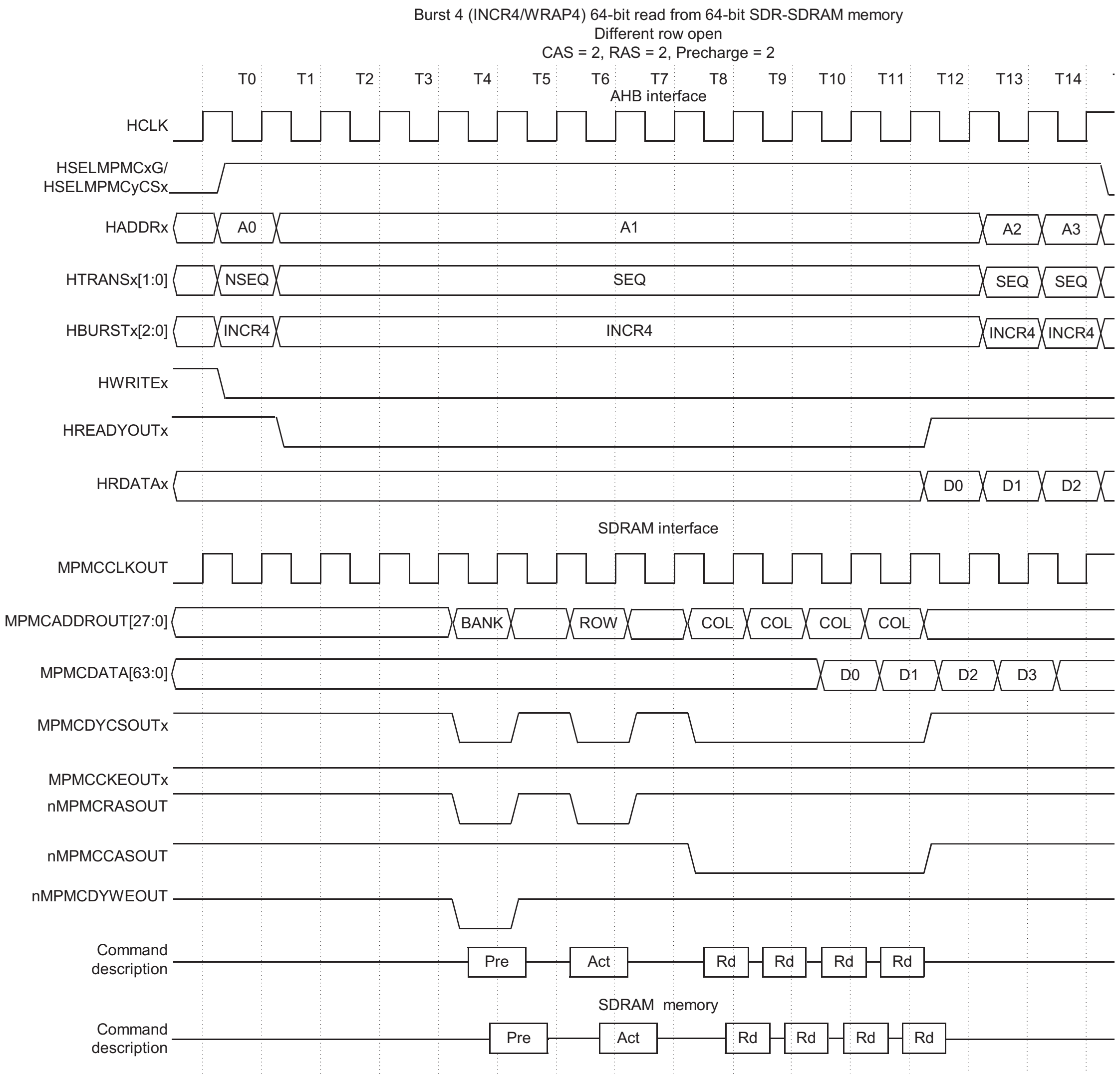


Figure 7-5 Burst4 (INCRA4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, different row open

7.5.6 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open

Figure 7-6 on page 7-21 and Table 7-10 show an example of a burst 4, 64-bit AHB write to 64-bit wide SDR-SDRAM. The row to be written to (in SDRAM) is open.

Table 7-10 Burst4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	AHB write data provided to memory controller. Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Write data 0 submitted to SDRAM.
T4-T5	Write data 1 submitted to SDRAM.
T5-T6	Write data 2 submitted to SDRAM.
T6-T7	Write data 3 submitted to SDRAM.

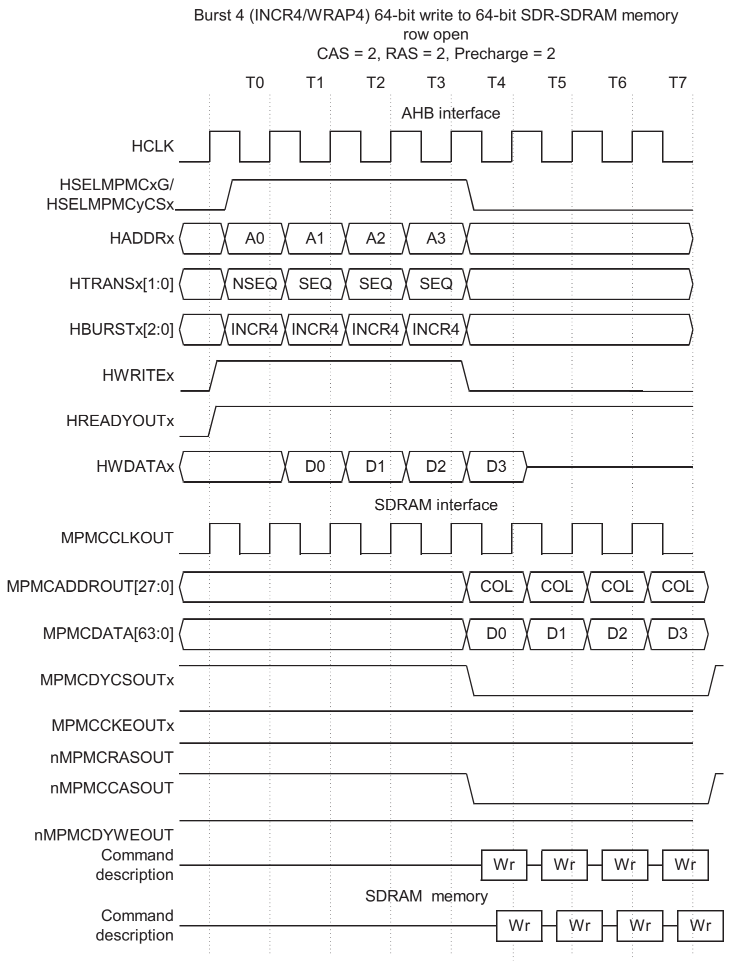


Figure 7-6 Burst 4 (INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open

7.5.7 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, bank precharged

Figure 7-7 on page 7-23 and Table 7-11 show an example of a burst 4, 64-bit AHB write to 64-bit wide SDR-SDRAM. The bank to be written to (in SDRAM) is precharged.

Table 7-11 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, bank precharged

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	AHB write data provided to memory controller. Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Active command submitted to SDRAM.
T4-T5	SDRAM active command latency.
T5-T6	Write data 0 submitted to SDRAM.
T6-T7	Write data 1 submitted to SDRAM.
T7-T8	Write data 2 submitted to SDRAM.
T8-T9	Write data 3 submitted to SDRAM.

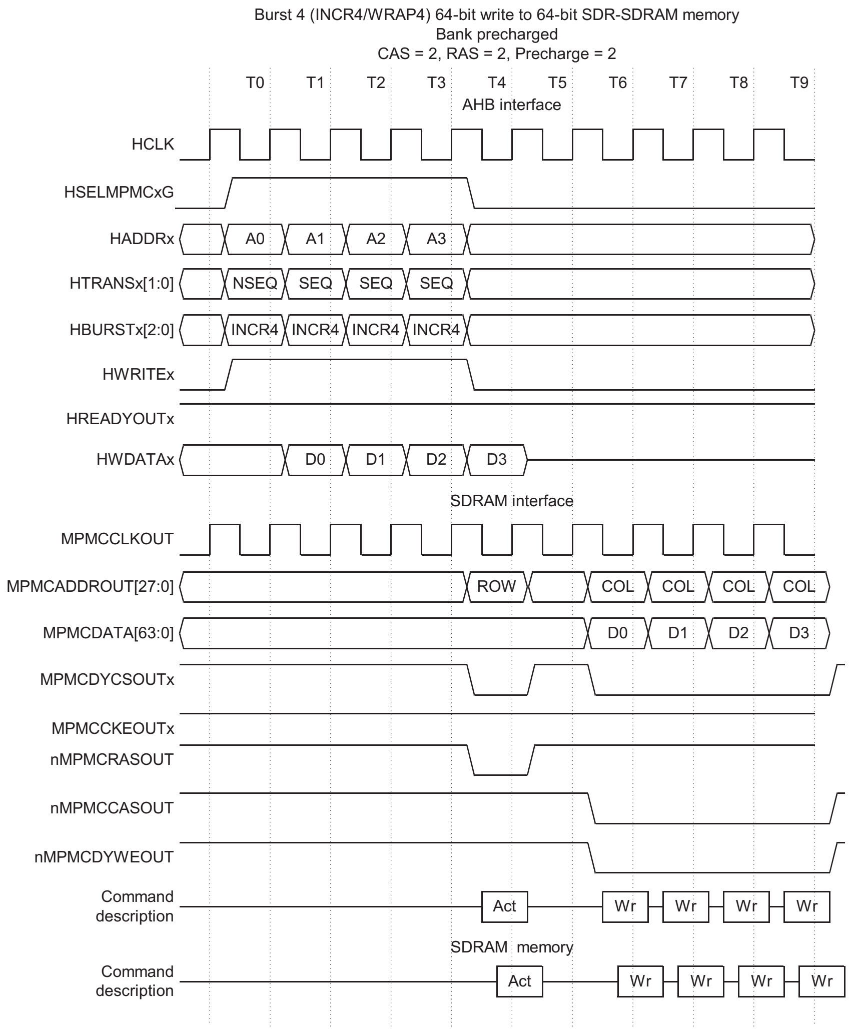


Figure 7-7 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, bank precharged

7.5.8 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, different row open

Figure 7-8 on page 7-25 and Table 7-12 show an example of a burst 4, 64-bit AHB write to 64-bit wide SDR-SDRAM. A different row to the one being written to is open.

Table 7-12 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, different row open

Cycle	Description
T0	AHB address provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	AHB write data provided to memory controller. Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Precharge command submitted to SDRAM.
T4-T5	SDRAM precharge command latency.
T5-T6	Active command submitted to SDRAM.
T6-T7	SDRAM active command latency.
T7-T8	Write data 0 submitted to SDRAM.
T8-T9	Write data 1 submitted to SDRAM.
T9-T10	Write data 2 submitted to SDRAM.
T10-T11	Write data 3 submitted to SDRAM.

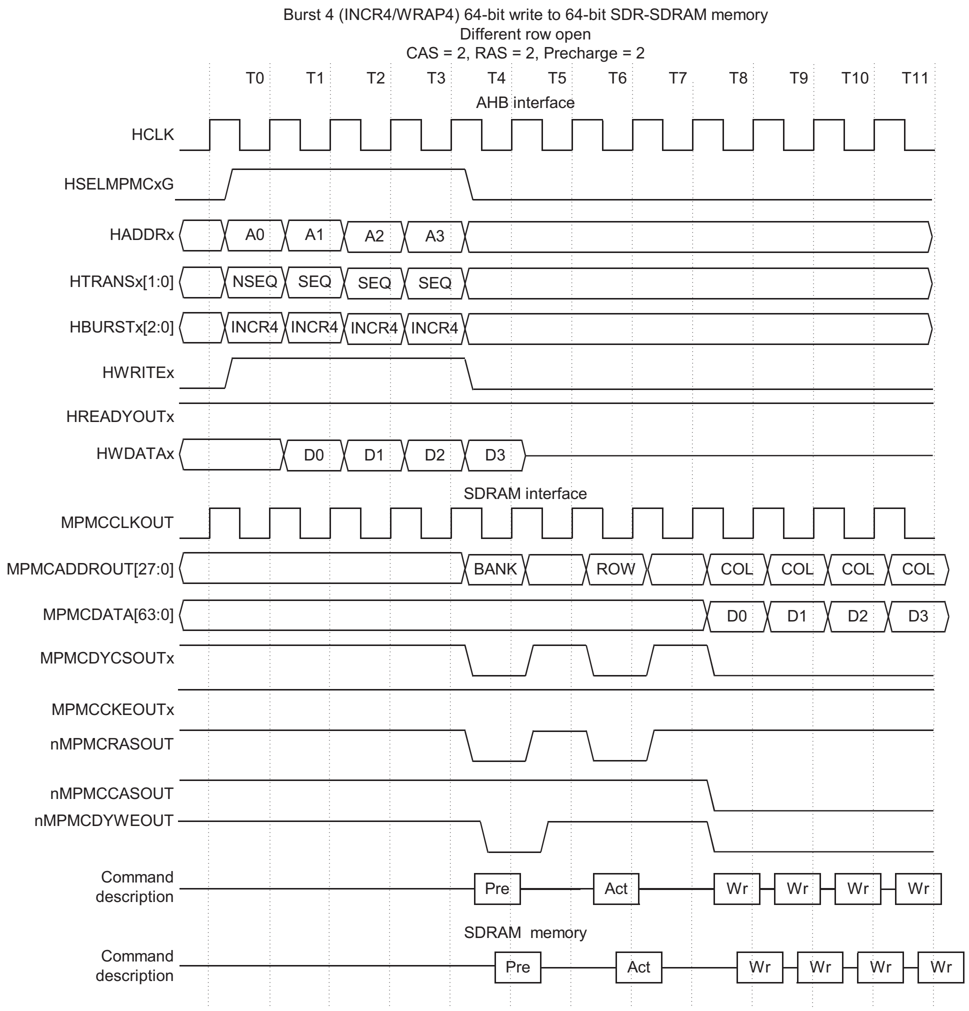


Figure 7-8 Burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, different row open

7.5.9 Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open

Figure 7-9 on page 7-27 and Table 7-13 show an example of two back-to-back burst 4, 64-bit AHB read from 64-bit wide SDR-SDRAM. The row to be read from (in SDRAM) is open.

Table 7-13 Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open

Cycle	Description
T0	AHB address 1 provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Read command submitted to SDRAM.
T4-T5	SDRAM CAS access latency.
T5-T6	SDRAM CAS access latency.
T6-T7	Read data 0 capture.
T7-T8	Read data 1 capture. Provide read data 0 to AHB.
T8-T9	Read data 2 capture. Provide read data 1 to AHB.
T9-T10	Read data 3 capture. Provide read data 2 to AHB.
T10-T11	Provide read data 3 to AHB. AHB address 2 provided to memory controller.
T11-T12	AHB transaction processing.
T12-T13	Arbitration of AHB memory ports.
T13-T14	Memory controller processing.
T14-T15	Read command submitted to SDRAM.
T15-T16	SDRAM CAS access latency.
T16-T17	SDRAM CAS access latency.
T17-T18	Read data 0 capture.
T18-T19	Read data 1 capture. Provide read data 0 to AHB.

Table 7-13 Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open (continued)

Cycle	Description
T19-T20	Read data 2 capture. Provide read data 1 to AHB.
T20-T21	Read data 3 capture. Provide read data 2 to AHB.
T21-T22	Provide read data 3 to AHB

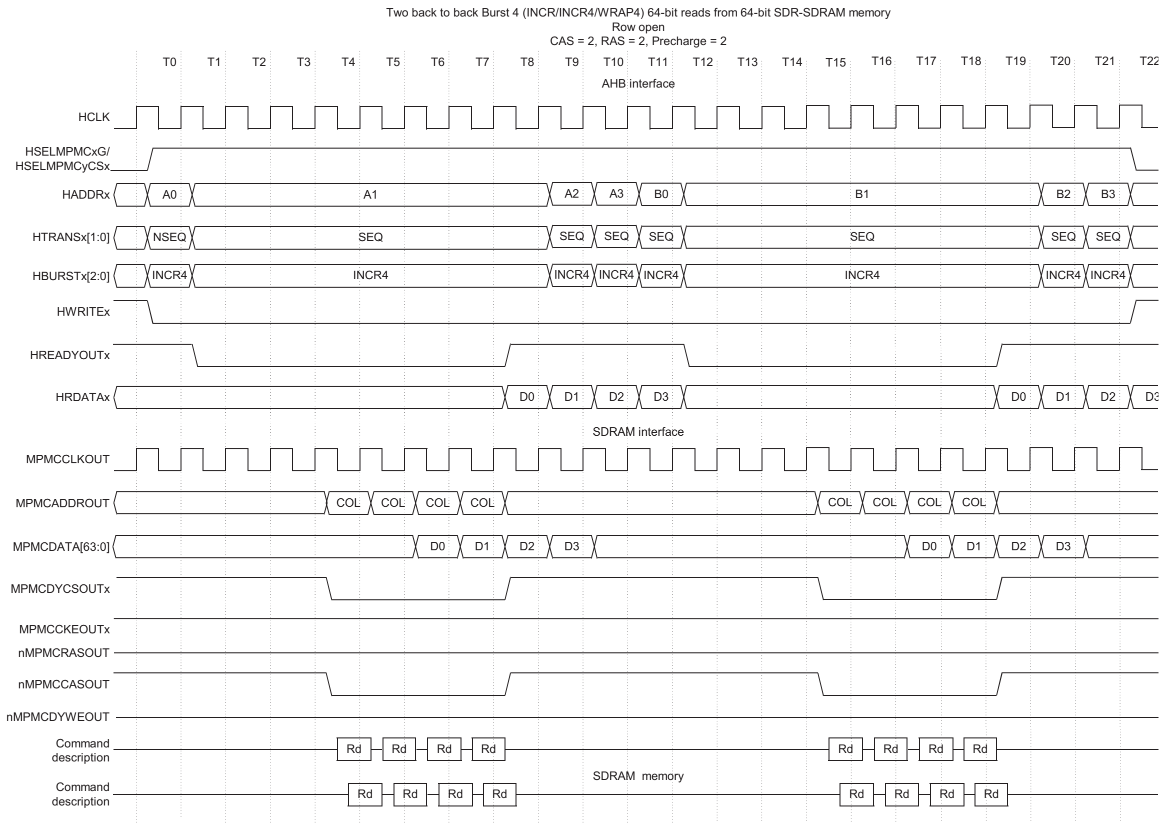


Figure 7-9 Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit read from 64-bit SDR-SDRAM, row open

7.5.10 Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open

Figure 7-10 on page 7-29 and Table 7-14 show an example of two back-to-back burst 4, 64-bit AHB write to 64-bit wide SDR-SDRAM. The row to be written to (in SDRAM) is open.

Table 7-14 Two back-to-back burst4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open

Cycle	Description
T0	AHB address 1 provided to memory controller.
T0-T1	AHB transaction processing.
T1-T2	AHB write data provided to memory controller. Arbitration of AHB memory ports.
T2-T3	Memory controller processing.
T3-T4	Write data 0 submitted to SDRAM. AHB address 2 provided to memory controller.
T4-T5	Write data 1 submitted to SDRAM. AHB transaction processing.
T5-T6	Write data 2 submitted to SDRAM. AHB write data provided to memory controller. Arbitration of AHB memory ports.
T6-T7	Write data 3 submitted to SDRAM. Memory controller processing.
T7-T8	Write data 0 submitted to SDRAM.
T8-T9	Write data 1 submitted to SDRAM.
T9-T10	Write data 2 submitted to SDRAM.
T10-T11	Write data 3 submitted to SDRAM.

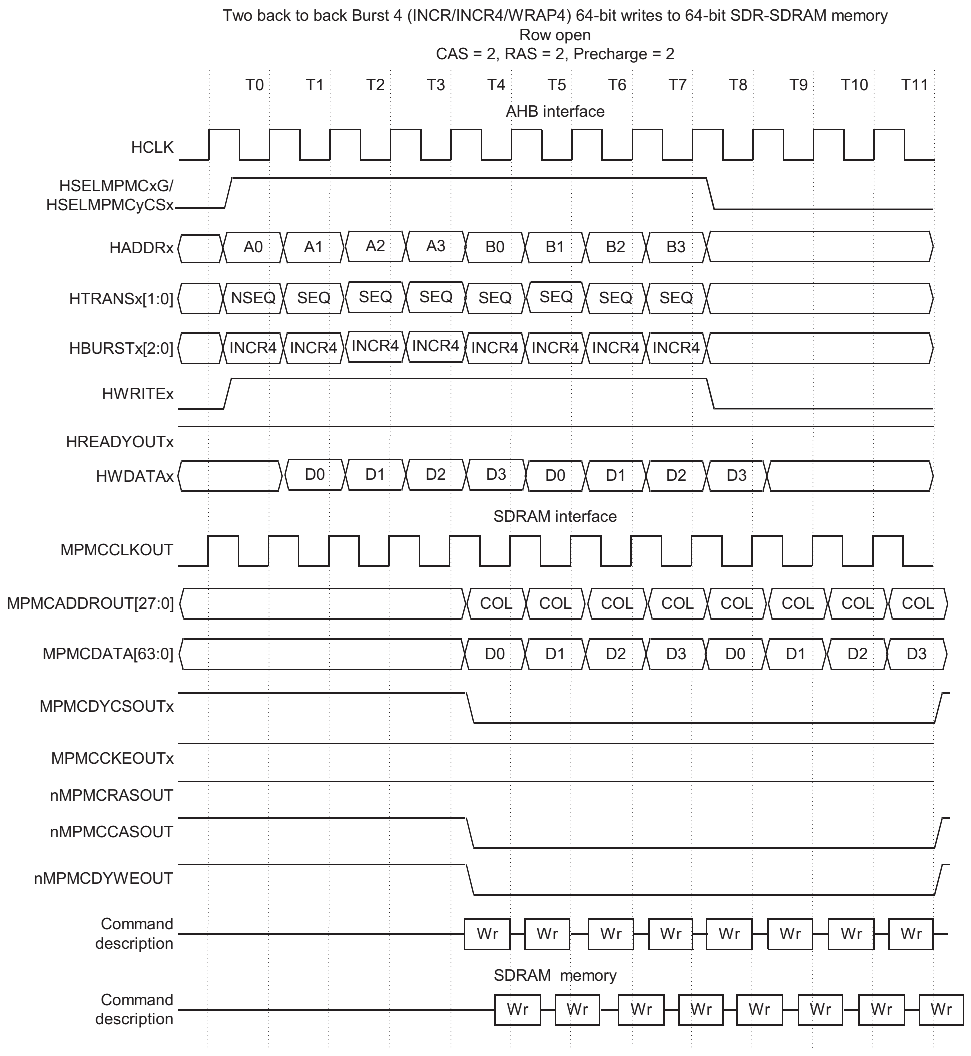


Figure 7-10 Two back-to-back burst 4 (INCR/INCR4/WRAP4) 64-bit write to 64-bit SDR-SDRAM, row open

7.6 Address mapping

The tables in this section provide the mapping of AHB address bus addresses **HADDR[31:0]** to the external dynamic memory address **MPMCADDROUT[18:0]** for various memory configurations and bus widths. The address mapping is selected by programming the *Address Mapping* (AM) bits in the MPMCDynamicConfig[0-3] Registers. The outputs from the PrimeCell MPMC tables indicate the address mapping out of the MPMC. The following conventions are used in the address mapping tables:

MPMC output address

Indicates the address lines output from the MPMC, **MPMCADDROUT[18:0]**.

Memory device connections

Indicates the device signals that must be connected to the **MPMCADDROUT[18:0]** lines.

Row Indicates the **MPMCADDROUT** address bits that are mapped to the **HADDR** address bits for a row access.

Column Indicates the **MPMCADDROUT** address bits that are mapped to the **HADDR** address bits for a column access.

Bank Indicates the **MPMCADDROUT** address bits that are mapped to the **HADDR** address bits for a bank access.

MPMC Indicates that the bit value is determined by the MPMC SDRAM controller, not directly by the AHB port **HADDR** address input.

The MPMC SDRAM controller always transfers 64 bits of data at a time:

- For little-endian memory and chip selects with a 32-bit wide data bus, the SDRAM controller performs two transfers with the lowest bit of the column address set consecutively to `0b` and `1b`.
- For little-endian memory and chip selects with a 16-bit wide data bus, the SDRAM controller performs four transfers with the lowest two bits of the column address set consecutively to `00b`, `01b`, `10b`, and `11b`.

32-bit DDR devices use address bit A8 for the auto precharge input. All SDR and other DDR devices use address bit A10. Table 7-15 on page 7-32 to Table 7-91 on page 7-108 show the auto precharge using address bit A10. The **MPMCAPOUT** output must be connected to the auto precharge input of the memory device instead of the relevant address pin. This enables SDRAM refreshes to be performed at any time.

——— **Note** ———

For Table 7-15 on page 7-32 to Table 7-91 on page 7-108:

- **BA**, **BA0**, and **BA1** indicate the bank address signals. AP indicates the auto precharge signal (normally address bit 10). **SA0**, **SA1**, and **SA2** indicate the segment address signals for V-SyncFlash.
- The typographical conventions differ from those defined in *Typographical conventions* on page xxii, as follows:
  - values in non-highlighted text indicate row accesses
  - values in bold indicate bank accesses
  - values in italic indicate column accesses
  - values in bold italic indicate segment accesses.

The address mapping is described in the following sections:

- *32-bit wide data bus address mappings (BRC) (SDR-SDRAM)*
- *32-bit wide data bus address mappings (RBC) (SDR-SDRAM)* on page 7-44
- *16-bit wide data bus address mappings (BRC) (SDR-SDRAM)* on page 7-57
- *16-bit wide data bus address mappings (RBC) (SDR-SDRAM)* on page 7-67
- *64-bit wide data bus address mappings (BRC) (SDR-SDRAM)* on page 7-77
- *64-bit wide data bus address mappings (RBC) (SDR-SDRAM)* on page 7-85
- *32-bit wide data bus address mappings (BSRSC) (V-SyncFlash)* on page 7-93
- *16-bit wide data bus address mappings (BSRSC) (V-SyncFlash)* on page 7-99
- *64-bit wide data bus address mappings (BSRSC) (V-SyncFlash)* on page 7-102.

7.6.1 32-bit wide data bus address mappings (BRC) (SDR-SDRAM)

The 32-bit wide data bus address mappings for SDRAM devices in a *Bank, Row, Column* (BRC) mapping scheme are shown in:

- *32-bit wide external data bus 16M SDRAM (1Mx16, BRC)* on page 7-32
- *32-bit wide external data bus 16M SDRAM (2Mx8, BRC)* on page 7-33
- *32-bit wide external data bus 64M SDRAM (2Mx32, BRC)* on page 7-34
- *32-bit wide external data bus 64M SDRAM (4Mx16, BRC)* on page 7-35
- *32-bit wide external data bus 64M SDRAM (8Mx8, BRC)* on page 7-36
- *32-bit wide external data bus 128M SDRAM (4Mx32, BRC)* on page 7-37
- *32-bit wide external data bus 128M SDRAM (8Mx16, BRC)* on page 7-38
- *32-bit wide external data bus 128M SDRAM (16Mx8, BRC)* on page 7-39
- *32-bit wide external data bus 256M SDRAM (8Mx32, BRC)* on page 7-40
- *32-bit wide external data bus 256M SDRAM (16Mx16, BRC)* on page 7-41

- *32-bit wide external data bus 256M SDRAM (32Mx8, BRC)* on page 7-42
- *32-bit wide external data bus 512M SDRAM (32Mx16, BRC)* on page 7-43
- *32-bit wide external data bus 512M SDRAM (64Mx8, BRC)* on page 7-44.

32-bit wide external data bus 16M SDRAM (1Mx16, BRC)

Table 7-15 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (1Mx16, pin 21 used as bank select).

Table 7-15 Address mapping for 16M SDRAM (1Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	-	-	9	A7	A7
22	-	-	8	A6	A6
21	A13	BA	7	A5	A5
20	A10	AP	6	A4	A4
19	A9	A9	5	A3	A3
18	A8	A8	4	A2	A2
17	A7	A7	3	A1	A1
16	A6	A6	MPMC	A0	A0
15	A5	A5	1:0	-	-

32-bit wide external data bus 16M SDRAM (2Mx8, BRC)

Table 7-16 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (2Mx8, pin 22 used as bank select).

Table 7-16 Address mapping for 16M SDRAM (2Mx8, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A3	A3
27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	-	-	11	A0	A0
24	-	-	10	A8	A8
23	-	-	9	A7	A7
22	A13	BA	8	A6	A6
21	A10	AP	7	A5	A5
20	A9	A9	6	A4	A4
19	A8	A8	5	A3	A3
18	A7	A7	4	A2	A2
17	A6	A6	3	A1	A1
16	A5	A5	MPMC	A0	A0
15	A4	A4	1:0	-	-

32-bit wide external data bus 64M SDRAM (2Mx32, BRC)

Table 7-17 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (2Mx32, pins 21 and 22 used as bank select).

Table 7-17 Address mapping for 64M SDRAM (2Mx8, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	-	-	9	A7	A7
22	A14	BA1	8	A6	A6
21	A13	BA0	7	A5	A5
20	A10	AP	6	A4	A4
19	A9	A9	5	A3	A3
18	A8	A8	4	A2	A2
17	A7	A7	3	A1	A1
16	A6	A6	MPMC	A0	A0
15	A5	A5	1:0	-	-

32-bit wide external data bus 64M SDRAM (4Mx16, BRC)

Table 7-18 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (4Mx16, pins 22 and 23 used as bank select).

Table 7-18 Address mapping for 64M SDRAM (4Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	A14	BA1	9	A7	A7
22	A13	BA0	8	A6	A6
21	A11	A11	7	A5	A5
20	A10	AP	6	A4	A4
19	A9	A9	5	A3	A3
18	A8	A8	4	A2	A2
17	A7	A7	3	A1	A1
16	A6	A6	MPMC	A0	A0
15	A5	A5	1:0	-	-

32-bit wide external data bus 64M SDRAM (8Mx8, BRC)

Table 7-19 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (8Mx8, pins 23 and 24 used as bank select).

Table 7-19 Address mapping for 64M SDRAM (8Mx8, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A3	A3
27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	-	-	11	A0	A0
24	A14	BA1	10	A8	A8
23	A13	BA0	9	A7	A7
22	A11	A11	8	A6	A6
21	A10	AP	7	A5	A5
20	A9	A9	6	A4	A4
19	A8	A8	5	A3	A3
18	A7	A7	4	A2	A2
17	A6	A6	3	A1	A1
16	A5	A5	MPMC	A0	A0
15	A4	A4	1:0	-	-

32-bit wide external data bus 128M SDRAM (4Mx32, BRC)

Table 7-20 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (4Mx32, pins 22 and 23 used as bank select).

Table 7-20 Address mapping for 128M SDRAM (4Mx32, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	A14	BA1	9	A7	A7
22	A13	BA0	8	A6	A6
21	A11	A11	7	A5	A5
20	A10	AP	6	A4	A4
19	A9	A9	5	A3	A3
18	A8	A8	4	A2	A2
17	A7	A7	3	A1	A1
16	A6	A6	MPMC	A0	A0
15	A5	A5	1:0	-	-

32-bit wide external data bus 128M SDRAM (8Mx16, BRC)

Table 7-21 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (8Mx16, pins 23 and 24 used as bank select).

Table 7-21 Address mapping for 128M SDRAM (8Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A3	A3
27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	-	-	11	A0	A0
24	A14	BA1	10	A8	A8
23	A13	BA0	9	A7	A7
22	A11	A11	8	A6	A6
21	A10	AP	7	A5	A5
20	A9	A9	6	A4	A4
19	A8	A8	5	A3	A3
18	A7	A7	4	A2	A2
17	A6	A6	3	A1	A1
16	A5	A5	MPMC	A0	A0
15	A4	A4	1:0	-	-

32-bit wide external data bus 128M SDRAM (16Mx8, BRC)

Table 7-22 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (16Mx8, pins 24 and 25 used as bank select).

Table 7-22 Address mapping for 128M SDRAM (16Mx8, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	A14	BA1	11	A9	A9
24	A13	BA0	10	A8	A8
23	A11	A11	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 256M SDRAM (8Mx32, BRC)

Table 7-23 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (8Mx32, pins 23 and 24 used as bank select).

Table 7-23 Address mapping for 256M SDRAM (8Mx32, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	A14	BA1	10	A0	A0
23	A13	BA0	9	A7	A7
22	A12	A12	8	A6	A6
21	A11	A11	7	A5	A5
20	A10	AP	6	A4	A4
19	A9	A9	5	A3	A3
18	A8	A8	4	A2	A2
17	A7	A7	3	A1	A1
16	A6	A6	MPMC	A0	A0
15	A5	A5	1:0	-	-

32-bit wide external data bus 256M SDRAM (16Mx16, BRC)

Table 7-24 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (16Mx16, pins 24 and 25 used as bank select).

Table 7-24 Address mapping for 256M SDRAM (16Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A3	A3
27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	A14	BA1	11	A0	A0
24	A13	BA0	10	A8	A8
23	A12	A12	9	A7	A7
22	A11	A11	8	A6	A6
21	A10	AP	7	A5	A5
20	A9	A9	6	A4	A4
19	A8	A8	5	A3	A3
18	A7	A7	4	A2	A2
17	A6	A6	3	A1	A1
16	A5	A5	MPMC	A0	A0
15	A4	A4	1:0	-	-

32-bit wide external data bus 256M SDRAM (32Mx8, BRC)

Table 7-25 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (32Mx8, pins 25 and 26 used as bank select).

Table 7-25 Address mapping for 256M SDRAM (32Mx8, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	A14	BA1	12	A0	A0
25	A13	BA0	11	A9	A9
24	A12	A12	10	A8	A8
23	A11	A11	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 512M SDRAM (32Mx16, BRC)

Table 7-26 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (32Mx16, pins 25 and 26 used as bank select).

Table 7-26 Address mapping for 512M SDRAM (32Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	A14	BA1	12	A0	A0
25	A13	BA0	11	A9	A9
24	A12	A12	10	A8	A8
23	A11	A11	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 512M SDRAM (64Mx8, BRC)

Table 7-27 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (64Mx8, pins 26 and 27 used as bank select).

Table 7-27 Address mapping for 512M SDRAM (64Mx8, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A1	A1
27	A14	BA1	13	A0	A0
26	A13	BA0	12	A11	A11
25	A12	A12	11	A9	A9
24	A11	A11	10	A8	A8
23	A10	AP	9	A7	A7
22	A9	A9	8	A6	A6
21	A8	A8	7	A5	A5
20	A7	A7	6	A4	A4
19	A6	A6	5	A3	A3
18	A5	A5	4	A2	A2
17	A4	A4	3	A1	A1
16	A3	A3	MPMC	A0	A0
15	A2	A2	1:0	-	-

7.6.2 32-bit wide data bus address mappings (RBC) (SDR-SDRAM)

The 32-bit wide data bus address mappings for SDRAM devices in a *Row, Bank, Column* (RBC) mapping scheme are shown in:

- *32-bit wide external data bus 16M SDRAM (1Mx16, RBC)* on page 7-45
- *32-bit wide external data bus 16M SDRAM (2Mx8, RBC)* on page 7-46
- *32-bit wide external data bus 64M SDRAM (2Mx32, RBC)* on page 7-47
- *32-bit wide external data bus 64M SDRAM (4Mx16, RBC)* on page 7-48
- *32-bit wide external data bus 64M SDRAM (8Mx8, RBC)* on page 7-49
- *32-bit wide external data bus 128M SDRAM (4Mx32, RBC)* on page 7-50

- *32-bit wide external data bus 128M SDRAM (8Mx16, RBC)* on page 7-51
- *32-bit wide external data bus 128M SDRAM (16Mx8, RBC)* on page 7-52
- *32-bit wide external data bus 256M SDRAM (8Mx32, RBC)* on page 7-53
- *32-bit wide external data bus 256M SDRAM (16Mx16, RBC)* on page 7-54
- *32-bit wide external data bus 256M SDRAM (32Mx8, RBC)* on page 7-55
- *32-bit wide external data bus 512M SDRAM (32Mx16, RBC)* on page 7-56
- *32-bit wide external data bus 512M SDRAM (64Mx8, RBC)* on page 7-57.

32-bit wide external data bus 16M SDRAM (1Mx16, RBC)

Table 7-28 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (1Mx16, pin 10 used as bank select).

Table 7-28 Address mapping for 16M SDRAM (1Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A3	A3
27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	-	-	11	A0	A0
24	-	-	10	A13	BA
23	-	-	9	A7	A7
22	-	-	8	A6	A6
21	A10	AP	7	A5	A5
20	A9	A9	6	A4	A4
19	A8	A8	5	A3	A3
18	A7	A7	4	A2	A2
17	A6	A6	3	A1	A1
16	A5	A5	MPMC	A0	A0
15	A4	A4	1:0	-	-

32-bit wide external data bus 16M SDRAM (2Mx8, RBC)

Table 7-29 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (2Mx8, pin 11 used as bank select).

Table 7-29 Address mapping for 16M SDRAM (2Mx8, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA
24	-	-	10	A8	A8
23	-	-	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 64M SDRAM (2Mx32, RBC)

Table 7-30 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (2Mx32, pins 10 and 11 used as bank select).

Table 7-30 Address mapping for 64M SDRAM (2Mx32, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	-	-	10	A13	BA0
23	-	-	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 64M SDRAM (4Mx16, RBC)

Table 7-31 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (4Mx16, pins 10 and 11 used as bank select).

Table 7-31 Address mapping for 64M SDRAM (4Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	-	-	10	A13	BA0
23	A11	A11	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 64M SDRAM (8Mx8, RBC)

Table 7-32 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (8Mx8, pins 11 and 12 used as bank select).

Table 7-32 Address mapping for 64M SDRAM (8Mx8, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A1	A1
27	-	-	13	A0	A0
26	-	-	12	A14	BA1
25	-	-	11	A13	BA0
24	A11	A11	10	A8	A8
23	A10	AP	9	A7	A7
22	A9	A9	8	A6	A6
21	A8	A8	7	A5	A5
20	A7	A7	6	A4	A4
19	A6	A6	5	A3	A3
18	A5	A5	4	A2	A2
17	A4	A4	3	A1	A1
16	A3	A3	MPMC	A0	A0
15	A2	A2	1:0	-	-

32-bit wide external data bus 128M SDRAM (4Mx32, RBC)

Table 7-33 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (4Mx32, pins 10 and 11 used as bank select).

Table 7-33 Address mapping for 128M SDRAM (4Mx32, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	-	-	10	A13	BA0
23	A11	A11	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 128M SDRAM (8Mx16, RBC)

Table 7-34 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (8Mx16, pins 11 and 12 used as bank select).

Table 7-34 Address mapping for 128M SDRAM (8Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A1	A1
27	-	-	13	A0	A0
26	-	-	12	A14	BA1
25	-	-	11	A13	BA0
24	A11	A11	10	A8	A8
23	A10	AP	9	A7	A7
22	A9	A9	8	A6	A6
21	A8	A8	7	A5	A5
20	A7	A7	6	A4	A4
19	A6	A6	5	A3	A3
18	A5	A5	4	A2	A2
17	A4	A4	3	A1	A1
16	A3	A3	MPMC	A0	A0
15	A2	A2	1:0	-	-

32-bit wide external data bus 128M SDRAM (16Mx8, RBC)

Table 7-35 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (16Mx8, pins 12 and 13 used as bank select).

Table 7-35 Address mapping for 128M SDRAM (16Mx8, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A0	A0
27	-	-	13	A14	BA1
26	-	-	12	A13	BA0
25	A11	A11	11	A9	A9
24	A10	AP	10	A8	A8
23	A9	A9	9	A7	A7
22	A8	A8	8	A6	A6
21	A7	A7	7	A5	A5
20	A6	A6	6	A4	A4
19	A5	A5	5	A3	A3
18	A4	A4	4	A2	A2
17	A3	A3	3	A1	A1
16	A2	A2	MPMC	A0	A0
15	A1	A1	1:0	-	-

32-bit wide external data bus 256M SDRAM (8Mx32, RBC)

Table 7-36 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (8Mx32, pins 10 and 11 used as bank select).

Table 7-36 Address mapping for 256M SDRAM (8Mx32, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A2	A2
27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	A12	A12	10	A13	BA0
23	A11	A11	9	A7	A7
22	A10	AP	8	A6	A6
21	A9	A9	7	A5	A5
20	A8	A8	6	A4	A4
19	A7	A7	5	A3	A3
18	A6	A6	4	A2	A2
17	A5	A5	3	A1	A1
16	A4	A4	MPMC	A0	A0
15	A3	A3	1:0	-	-

32-bit wide external data bus 256M SDRAM (16Mx16, RBC)

Table 7-37 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (16Mx16, pins 11 and 12 used as bank select).

Table 7-37 Address mapping for 256M SDRAM (16Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A1	A1
27	-	-	13	A0	A0
26	-	-	12	A14	BA1
25	A12	A12	11	A13	BA0
24	A11	A11	10	A8	A8
23	A10	AP	9	A7	A7
22	A9	A9	8	A6	A6
21	A8	A8	7	A5	A5
20	A7	A7	6	A4	A4
19	A6	A6	5	A3	A3
18	A5	A5	4	A2	A2
17	A4	A4	3	A1	A1
16	A3	A3	MPMC	A0	A0
15	A2	A2	1:0	-	-

32-bit wide external data bus 256M SDRAM (32Mx8, RBC)

Table 7-38 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (32Mx8, pins 12 and 13 used as bank select).

Table 7-38 Address mapping for 256M SDRAM (32Mx8, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A0	A0
27	-	-	13	A14	BA1
26	A12	A12	12	A13	BA0
25	A11	A11	11	A9	A9
24	A10	AP	10	A8	A8
23	A9	A9	9	A7	A7
22	A8	A8	8	A6	A6
21	A7	A7	7	A5	A5
20	A6	A6	6	A4	A4
19	A5	A5	5	A3	A3
18	A4	A4	4	A2	A2
17	A3	A3	3	A1	A1
16	A2	A2	MPMC	A0	A0
15	A1	A1	1:0	-	-

32-bit wide external data bus 512M SDRAM (32Mx16, RBC)

Table 7-39 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (32Mx16, pins 12 and 13 used as bank select).

Table 7-39 Address mapping for 512M SDRAM (32Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A0	A0
27	-	-	13	A14	BA1
26	A12	A12	12	A13	BA0
25	A11	A11	11	A9	A9
24	A10	AP	10	A8	A8
23	A9	A9	9	A7	A7
22	A8	A8	8	A6	A6
21	A7	A7	7	A5	A5
20	A6	A6	6	A4	A4
19	A5	A5	5	A3	A3
18	A4	A4	4	A2	A2
17	A3	A3	3	A1	A1
16	A2	A2	MPMC	A0	A0
15	A1	A1	1:0	-	-

32-bit wide external data bus 512M SDRAM (64Mx8, RBC)

Table 7-40 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (64Mx8, pins 13 and 14 used as bank select).

Table 7-40 Address mapping for 512M SDRAM (64Mx8, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A14	BA1
27	A12	A12	13	A13	BA0
26	A11	A11	12	A11	A11
25	A10	AP	11	A9	A9
24	A9	A9	10	A8	A8
23	A8	A8	9	A7	A7
22	A7	A7	8	A6	A6
21	A6	A6	7	A5	A5
20	A5	A5	6	A4	A4
19	A4	A4	5	A3	A3
18	A3	A3	4	A2	A2
17	A2	A2	3	A1	A1
16	A1	A1	MPMC	A0	A0
15	A0	A0	1:0	-	-

7.6.3 16-bit wide data bus address mappings (BRC) (SDR-SDRAM)

The 16-bit wide data bus address mappings for SDRAM devices in a BRC mapping scheme are shown in:

- *16-bit wide external data bus 16M SDRAM (1Mx16, BRC)* on page 7-58
- *16-bit wide external data bus 16M SDRAM (2Mx8, BRC)* on page 7-59
- *16-bit wide external data bus 64M SDRAM (4Mx16, BRC)* on page 7-60
- *16-bit wide external data bus 64M SDRAM (8Mx8, BRC)* on page 7-61
- *16-bit wide external data bus 128M SDRAM (8Mx16, BRC)* on page 7-62
- *16-bit wide external data bus 128M SDRAM (16Mx8, BRC)* on page 7-63

- *16-bit wide external data bus 256M SDRAM (16Mx16, BRC)* on page 7-64
- *16-bit wide external data bus 256M SDRAM (32Mx8, BRC)* on page 7-65
- *16-bit wide external data bus 512M SDRAM (32Mx16, BRC)* on page 7-66
- *16-bit wide external data bus 512M SDRAM (64Mx8, BRC)* on page 7-67.

16-bit wide external data bus 16M SDRAM (1Mx16, BRC)

Table 7-41 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (1Mx16, pin 20 used as bank select).

Table 7-41 Address mapping for 16M SDRAM (1Mx16, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A4	A4
26	-	-	12	A3	A3
25	-	-	11	A2	A2
24	-	-	10	A1	A1
23	-	-	9	A0	A0
22	-	-	8	A7	A7
21	-	-	7	A6	A6
20	A13	BA	6	A5	A5
19	A10	AP	5	A4	A4
18	A9	A9	4	A3	A3
17	A8	A8	3	A2	A2
16	A7	A7	MPMC	A1	A1
15	A6	A6	MPMC	A0	A0
14	A5	A5	0	-	-

16-bit wide external data bus 16M SDRAM (2Mx8, BRC)

Table 7-42 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (2Mx8, pin 21 used as bank select).

Table 7-42 Address mapping for 16M SDRAM (2Mx8, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	-	-	9	A8	A8
22	-	-	8	A7	A7
21	A13	BA	7	A6	A6
20	A10	AP	6	A5	A5
19	A9	A9	5	A4	A4
18	A8	A8	4	A3	A3
17	A7	A7	3	A2	A2
16	A6	A6	MPMC	A1	A1
15	A5	A5	MPMC	A0	A0
14	A4	A4	0	-	-

16-bit wide external data bus 64M SDRAM (4Mx16, BRC)

Table 7-43 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (4Mx16, pins 21 and 22 used as bank select).

Table 7-43 Address mapping for 64M SDRAM (4Mx16, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A4	A4
26	-	-	12	A3	A3
25	-	-	11	A2	A2
24	-	-	10	A1	A1
23	-	-	9	A0	A0
22	A14	BA1	8	A7	A7
21	A13	BA0	7	A6	A6
20	A11	A11	6	A5	A5
19	A10	AP	5	A4	A4
18	A9	A9	4	A3	A3
17	A8	A8	3	A2	A2
16	A7	A7	MPMC	A1	A1
15	A6	A6	MPMC	A0	A0
14	A5	A5	0	-	-

16-bit wide external data bus 64M SDRAM (8Mx8, BRC)

Table 7-44 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (8Mx8, pins 22 and 23 used as bank select).

Table 7-44 Address mapping for 64M SDRAM (8Mx8, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	A14	BA1	9	A8	A8
22	A13	BA0	8	A7	A7
21	A11	A11	7	A6	A6
20	A10	AP	6	A5	A5
19	A9	A9	5	A4	A4
18	A8	A8	4	A3	A3
17	A7	A7	3	A2	A2
16	A6	A6	MPMC	A1	A1
15	A5	A5	MPMC	A0	A0
14	A4	A4	0	-	-

16-bit wide external data bus 128M SDRAM (8Mx16, BRC)

Table 7-45 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (8Mx16, pins 22 and 23 used as bank select).

Table 7-45 Address mapping for 128M SDRAM (8Mx16, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	A14	BA1	9	A8	A8
22	A13	BA0	8	A7	A7
21	A11	A11	7	A6	A6
20	A10	AP	6	A5	A5
19	A9	A9	5	A4	A4
18	A8	A8	4	A3	A3
17	A7	A7	3	A2	A2
16	A6	A6	MPMC	A1	A1
15	A5	A5	MPMC	A0	A0
14	A4	A4	0	-	-

16-bit wide external data bus 128M SDRAM (16Mx8, BRC)

Table 7-46 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (16Mx8, pins 23 and 24 used as bank select).

Table 7-46 Address mapping for 128M SDRAM (16Mx8, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	-	-	11	A0	A0
24	A14	BA1	10	A9	A9
23	A13	BA0	9	A8	A8
22	A11	A11	8	A7	A7
21	A10	AP	7	A6	A6
20	A9	A9	6	A5	A5
19	A8	A8	5	A4	A4
18	A7	A7	4	A3	A3
17	A6	A6	3	A2	A2
16	A5	A5	MPMC	A1	A1
15	A4	A4	MPMC	A0	A0
14	A3	A3	0	-	-

16-bit wide external data bus 256M SDRAM (16Mx16, BRC)

Table 7-47 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (16Mx16, pins 23 and 24 used as bank select).

Table 7-47 Address mapping for 256M SDRAM (16Mx16, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	A14	BA1	10	A0	A0
23	A13	BA0	9	A8	A8
22	A12	A12	8	A7	A7
21	A11	A11	7	A6	A6
20	A10	AP	6	A5	A5
19	A9	A9	5	A4	A4
18	A8	A8	4	A3	A3
17	A7	A7	3	A2	A2
16	A6	A6	MPMC	A1	A1
15	A5	A5	MPMC	A0	A0
14	A4	A4	0	-	-

16-bit wide external data bus 256M SDRAM (32Mx8, BRC)

Table 7-48 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (32Mx8, pins 24 and 25 used as bank select).

Table 7-48 Address mapping for 256M SDRAM (32Mx8, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	A14	BA1	11	A0	A0
24	A13	BA0	10	A9	A9
23	A12	A12	9	A8	A8
22	A11	A11	8	A7	A7
21	A10	AP	7	A6	A6
20	A9	A9	6	A5	A5
19	A8	A8	5	A4	A4
18	A7	A7	4	A3	A3
17	A6	A6	3	A2	A2
16	A5	A5	MPMC	A1	A1
15	A4	A4	MPMC	A0	A0
14	A3	A3	0	-	-

16-bit wide external data bus 512M SDRAM (32Mx16, BRC)

Table 7-49 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (32Mx16, pins 24 and 25 used as bank select).

Table 7-49 Address mapping for 512M SDRAM (32Mx16, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	A14	BA1	11	A0	A0
24	A13	BA0	10	A9	A9
23	A12	A12	9	A8	A8
22	A11	A11	8	A7	A7
21	A10	AP	7	A6	A6
20	A9	A9	6	A5	A5
19	A8	A8	5	A4	A4
18	A7	A7	4	A3	A3
17	A6	A6	3	A2	A2
16	A5	A5	MPMC	A1	A1
15	A4	A4	MPMC	A0	A0
14	A3	A3	0	-	-

16-bit wide external data bus 512M SDRAM (64Mx8, BRC)

Table 7-50 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (64Mx8, pins 25 and 26 used as bank select).

Table 7-50 Address mapping for 512M SDRAM (64Mx8, BRC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A1	A1
26	A14	BA1	12	A0	A0
25	A13	BA0	11	A11	A11
24	A12	A12	10	A9	A9
23	A11	A11	9	A8	A8
22	A10	AP	8	A7	A7
21	A9	A9	7	A6	A6
20	A8	A8	6	A5	A5
19	A7	A7	5	A4	A4
18	A6	A6	4	A3	A3
17	A5	A5	3	A2	A2
16	A4	A4	MPMC	A1	A1
15	A3	A3	MPMC	A0	A0
14	A2	A2	0	-	-

7.6.4 16-bit wide data bus address mappings (RBC) (SDR-SDRAM)

The 16-bit wide data bus address mappings for SDRAM devices in an RBC mapping scheme are shown in:

- *16-bit wide external data bus 16M SDRAM (1Mx16, RBC)* on page 7-68
- *16-bit wide external data bus 16M SDRAM (2Mx8, RBC)* on page 7-69
- *16-bit wide external data bus 64M SDRAM (4Mx16, RBC)* on page 7-70
- *16-bit wide external data bus 64M SDRAM (8Mx8, RBC)* on page 7-71
- *16-bit wide external data bus 128M SDRAM (8Mx16, RBC)* on page 7-72
- *16-bit wide external data bus 128M SDRAM (16Mx8, RBC)* on page 7-73

- *16-bit wide external data bus 256M SDRAM (16Mx16, RBC)* on page 7-74
- *16-bit wide external data bus 256M SDRAM (32Mx8, RBC)* on page 7-75
- *16-bit wide external data bus 512M SDRAM (32Mx16, RBC)* on page 7-76
- *16-bit wide external data bus 512M SDRAM (64Mx8, RBC)* on page 7-77.

16-bit wide external data bus 16M SDRAM (1Mx16, RBC)

Table 7-51 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (1Mx16, pin 9 used as bank select).

Table 7-51 Address mapping for 16M SDRAM (1Mx16, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	-	-	11	A1	A1
24	-	-	10	A0	A0
23	-	-	9	A13	BA
22	-	-	8	A7	A7
21	-	-	7	A6	A6
20	A10	AP	6	A5	A5
19	A9	A9	5	A4	A4
18	A8	A8	4	A3	A3
17	A7	A7	3	A2	A2
16	A6	A6	MPMC	A1	A1
15	A5	A5	MPMC	A0	A0
14	A4	A4	0	-	-

16-bit wide external data bus 16M SDRAM (2Mx8, RBC)

Table 7-52 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (2Mx8, pin 10 used as bank select).

Table 7-52 Address mapping for 16M SDRAM (2Mx8, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A2	A2
26	-	-	12	A1	A1
25	-	-	11	A0	A0
24	-	-	10	A13	BA
23	-	-	9	A8	A8
22	-	-	8	A7	A7
21	A10	AP	7	A6	A6
20	A9	A9	6	A5	A5
19	A8	A8	5	A4	A4
18	A7	A7	4	A3	A3
17	A6	A6	3	A2	A2
16	A5	A5	MPMC	A1	A1
15	A4	A4	MPMC	A0	A0
14	A3	A3	0	-	-

16-bit wide external data bus 64M SDRAM (4Mx16, RBC)

Table 7-53 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (4Mx16, pins 9 and 10 used as bank select).

Table 7-53 Address mapping for 64M SDRAM (4Mx16, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13		A2
26	-	-	12		A1
25	-	-	11		A0
24	-	-	10	A14	BA1
23	-	-	9	A13	BA0
22	A11	A11	8	A7	A7
21	A10	AP	7	A6	A6
20	A9	A9	6	A5	A5
19	A8	A8	5	A4	A4
18	A7	A7	4	A3	A3
17	A6	A6	3	A2	A2
16	A5	A5	MPMC	A1	A1
15	A4	A4	MPMC	A0	A0
14	A3	A3	0	-	-

16-bit wide external data bus 64M SDRAM (8Mx8, RBC)

Table 7-54 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (8Mx8, pins 10 and 11 used as bank select).

Table 7-54 Address mapping for 64M SDRAM (8Mx8, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	-	-	10	A13	BA0
23	A11	A11	9	A8	A8
22	A10	AP	8	A7	A7
21	A9	A9	7	A6	A6
20	A8	A8	6	A5	A5
19	A7	A7	5	A4	A4
18	A6	A6	4	A3	A3
17	A5	A5	3	A2	A2
16	A4	A4	MPMC	A1	A1
15	A3	A3	MPMC	A0	A0
14	A2	A2	0	-	-

16-bit wide external data bus 128M SDRAM (8Mx16, RBC)

Table 7-55 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (8Mx16, pins 10 and 11 used as bank select).

Table 7-55 Address mapping for 128M SDRAM (8Mx16, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	-	-	10	A13	BA0
23	A11	A11	9	A8	A8
22	A10	AP	8	A7	A7
21	A9	A9	7	A6	A6
20	A8	A8	6	A5	A5
19	A7	A7	5	A4	A4
18	A6	A6	4	A3	A3
17	A5	A5	3	A2	A2
16	A4	A4	MPMC	A1	A1
15	A3	A3	MPMC	A0	A0
14	A2	A2	0	-	-

16-bit wide external data bus 128M SDRAM (16Mx8, RBC)

Table 7-56 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (16Mx8, pins 11 and 12 used as bank select).

Table 7-56 Address mapping for 128M SDRAM (16Mx8, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A0	A0
26	-	-	12	A14	BA1
25	-	-	11	A13	BA0
24	A11	A11	10	A9	A9
23	A10	AP	9	A8	A8
22	A9	A9	8	A7	A7
21	A8	A8	7	A6	A6
20	A7	A7	6	A5	A5
19	A6	A6	5	A4	A4
18	A5	A5	4	A3	A3
17	A4	A4	3	A2	A2
16	A3	A3	MPMC	A1	A1
15	A2	A2	MPMC	A0	A0
14	A1	A1	0	-	-

16-bit wide external data bus 256M SDRAM (16Mx16, RBC)

Table 7-57 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (16Mx16, pins 10 and 11 used as bank select).

Table 7-57 Address mapping for 256M SDRAM (16Mx16, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A1	A1
26	-	-	12	A0	A0
25	-	-	11	A14	BA1
24	A12	A12	10	A13	BA0
23	A11	A11	9	A8	A8
22	A10	AP	8	A7	A7
21	A9	A9	7	A6	A6
20	A8	A8	6	A5	A5
19	A7	A7	5	A4	A4
18	A6	A6	4	A3	A3
17	A5	A5	3	A2	A2
16	A4	A4	MPMC	A1	A1
15	A3	A3	MPMC	A0	A0
14	A2	A2	0	-	-

16-bit wide external data bus 256M SDRAM (32Mx8, RBC)

Table 7-58 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (32Mx8, pins 11 and 12 used as bank select).

Table 7-58 Address mapping for 256M SDRAM (32Mx8, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A0	A0
26	-	-	12	A14	BA1
25	A12	A12	11	A13	BA0
24	A11	A11	10	A9	A9
23	A10	AP	9	A8	A8
22	A9	A9	8	A7	A7
21	A8	A8	7	A6	A6
20	A7	A7	6	A5	A5
19	A6	A6	5	A4	A4
18	A5	A5	4	A3	A3
17	A4	A4	3	A2	A2
16	A3	A3	MPMC	A1	A1
15	A2	A2	MPMC	A0	A0
14	A1	A1	0	-	-

16-bit wide external data bus 512M SDRAM (32Mx16, RBC)

Table 7-59 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (32Mx16, pins 11 and 12 used as bank select).

Table 7-59 Address mapping for 512M SDRAM (32Mx16, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A0	A0
26	-	-	12	A14	BA1
25	A12	A12	11	A13	BA0
24	A11	A11	10	A9	A9
23	A10	AP	9	A8	A8
22	A9	A9	8	A7	A7
21	A8	A8	7	A6	A6
20	A7	A7	6	A5	A5
19	A6	A6	5	A4	A4
18	A5	A5	4	A3	A3
17	A4	A4	3	A2	A2
16	A3	A3	MPMC	A1	A1
15	A2	A2	MPMC	A0	A0
14	A1	A1	0	-	-

16-bit wide external data bus 512M SDRAM (64Mx8, RBC)

Table 7-60 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (64Mx8, pins 12 and 13 used as bank select).

Table 7-60 Address mapping for 512M SDRAM (64Mx8, RBC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A14	BA1
26	A12	A12	12	A13	BA0
25	A11	A11	11	A11	A11
24	A10	AP	10	A9	A9
23	A9	A9	9	A8	A8
22	A8	A8	8	A7	A7
21	A7	A7	7	A6	A6
20	A6	A6	6	A5	A5
19	A5	A5	5	A4	A4
18	A4	A4	4	A3	A3
17	A3	A3	3	A2	A2
16	A2	A2	MPMC	A1	A1
15	A1	A1	MPMC	A0	A0
14	A0	A0	0	-	-

7.6.5 64-bit wide data bus address mappings (BRC) (SDR-SDRAM)

The 64-bit wide data bus address mappings for SDRAM devices in a BRC mapping scheme are shown in:

- *64-bit wide external data bus 16M SDRAM (1Mx16, BRC)* on page 7-78
- *64-bit wide external data bus 64M SDRAM (2Mx32, BRC)* on page 7-79
- *64-bit wide external data bus 64M SDRAM (4Mx16, BRC)* on page 7-80
- *64-bit wide external data bus 128M SDRAM (4Mx32, BRC)* on page 7-81
- *64-bit wide external data bus 128M SDRAM (8Mx16, BRC)* on page 7-82
- *64-bit wide external data bus 256M SDRAM (8Mx32, BRC)* on page 7-83

- *64-bit wide external data bus 256M SDRAM (16Mx16, BRC)* on page 7-84
- *64-bit wide external data bus 512M SDRAM (32Mx16, BRC)* on page 7-85.

64-bit wide external data bus 16M SDRAM (1Mx16, BRC)

Table 7-61 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (1Mx16, pin 22 used as bank select).

Table 7-61 Address mapping for 16M SDRAM (1Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A3	A3
28	-	-	13	A2	A2
27	-	-	12	A1	A1
26	-	-	11	A0	A0
25	-	-	10	A7	A7
24	-	-	9	A6	A6
23	-	-	8	A5	A5
22	A13	BA	7	A4	A4
21	A10	AP	6	A3	A3
20	A9	A9	5	A2	A2
19	A8	A8	4	A1	A1
18	A7	A7	3	A0	A0
17	A6	A6	2	-	-
16	A5	A5	1	-	-
15	A4	A4	0	-	-

64-bit wide external data bus 64M SDRAM (2Mx32, BRC)

Table 7-62 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (2Mx32, pins 22 and 23 used as bank select).

Table 7-62 Address mapping for 64M SDRAM (2Mx32, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A3	A3
28	-	-	13	A2	A2
27	-	-	12	A1	A1
26	-	-	11	A0	A0
25	-	-	10	A7	A7
24	-	-	9	A6	A6
23	A14	BA1	8	A5	A5
22	A13	BA0	7	A4	A4
21	A10	AP	6	A3	A3
20	A9	A9	5	A2	A2
19	A8	A8	4	A1	A1
18	A7	A7	3	A0	A0
17	A6	A6	2	-	-
16	A5	A5	1	-	-
15	A4	A4	0	-	-

64-bit wide external data bus 64M SDRAM (4Mx16, BRC)

Table 7-63 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (4Mx16, pins 23 and 24 used as bank select).

Table 7-63 Address mapping for 64M SDRAM (4Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A3	A3
28	-	-	13	A2	A2
27	-	-	12	A1	A1
26	-	-	11	A0	A0
25	-	-	10	A7	A7
24	A14	BA1	9	A6	A6
23	A13	BA0	8	A5	A5
22	A11	A11	7	A4	A4
21	A10	AP	6	A3	A3
20	A9	A9	5	A2	A2
19	A8	A8	4	A1	A1
18	A7	A7	3	A0	A0
17	A6	A6	2	-	-
16	A5	A5	1	-	-
15	A4	A4	0	-	-

64-bit wide external data bus 128M SDRAM (4Mx32, BRC)

Table 7-64 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (4Mx32, pins 23 and 24 used as bank select).

Table 7-64 Address mapping for 128M SDRAM (4Mx32, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A3	A3
28	-	-	13	A2	A2
27	-	-	12	A1	A1
26	-	-	11	A0	A0
25	-	-	10	A7	A7
24	A14	BA1	9	A6	A6
23	A13	BA0	8	A5	A5
22	A11	A11	7	A4	A4
21	A10	AP	6	A3	A3
20	A9	A9	5	A2	A2
19	A8	A8	4	A1	A1
18	A7	A7	3	A0	A0
17	A6	A6	2	-	-
16	A5	A5	1	-	-
15	A4	A4	0	-	-

64-bit wide external data bus 128M SDRAM (8Mx16, BRC)

Table 7-65 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (8Mx16, pins 24 and 25 used as bank select).

Table 7-65 Address mapping for 128M SDRAM (8Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A2	A2
28	-	-	13	A1	A1
27	-	-	12	A0	A0
26	-	-	11	A8	A8
25	A14	BA1	10	A7	A7
24	A13	BA0	9	A6	A6
23	A11	A11	8	A5	A5
22	A10	AP	7	A4	A4
21	A9	A9	6	A3	A3
20	A8	A8	5	A2	A2
19	A7	A7	4	A1	A1
18	A6	A6	3	A0	A0
17	A5	A5	2	-	-
16	A4	A4	1	-	-
15	A3	A3	0	-	-

64-bit wide external data bus 256M SDRAM (8Mx32, BRC)

Table 7-66 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (8Mx32, pins 24 and 25 used as bank select).

Table 7-66 Address mapping for 256M SDRAM (8Mx32, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A3	A3
28	-	-	13	A2	A2
27	-	-	12	A1	A1
26	-	-	11	A0	A0
25	A14	BA1	10	A7	A7
24	A13	BA0	9	A6	A6
23	A12	A12	8	A5	A5
22	A11	A11	7	A4	A4
21	A10	AP	6	A3	A3
20	A9	A9	5	A2	A2
19	A8	A8	4	A1	A1
18	A7	A7	3	A0	A0
17	A6	A6	2	-	-
16	A5	A5	1	-	-
15	A4	A4	0	-	-

64-bit wide external data bus 256M SDRAM (16Mx16, BRC)

Table 7-67 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (16Mx16, pins 25 and 26 used as bank select).

Table 7-67 Address mapping for 256M SDRAM (16Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A2	A2
28	-	-	13	A1	A1
27	-	-	12	A0	A0
26	A14	BA1	11	A8	A8
25	A13	BA0	10	A7	A7
24	A12	A12	9	A6	A6
23	A11	A11	8	A5	A5
22	A10	AP	7	A4	A4
21	A9	A9	6	A3	A3
20	A8	A8	5	A2	A2
19	A7	A7	4	A1	A1
18	A6	A6	3	A0	A0
17	A5	A5	2	-	-
16	A4	A4	1	-	-
15	A3	A3	0	-	-

64-bit wide external data bus 512M SDRAM (32Mx16, BRC)

Table 7-68 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (32Mx16, pins 26 and 27 used as bank select).

Table 7-68 Address mapping for 512M SDRAM (32Mx16, BRC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A1	A1
28	-	-	13	A0	A0
27	A14	BA1	12	A9	A9
26	A13	BA0	11	A8	A8
25	A12	A12	10	A7	A7
24	A11	A11	9	A6	A6
23	A10	AP	8	A5	A5
22	A9	A9	7	A4	A4
21	A8	A8	6	A3	A3
20	A7	A7	5	A2	A2
19	A6	A6	4	A1	A1
18	A5	A5	3	A0	A0
17	A4	A4	2	-	-
16	A3	A3	1	-	-
15	A2	A2	0	-	-

7.6.6 64-bit wide data bus address mappings (RBC) (SDR-SDRAM)

The 64-bit wide data bus address mappings for SDRAM devices in an RBC mapping scheme are shown in:

- *64-bit wide external data bus 16M SDRAM (1Mx16, RBC)* on page 7-86
- *64-bit wide external data bus 64M SDRAM (2Mx32, RBC)* on page 7-87
- *64-bit wide external data bus 64M SDRAM (4Mx16, RBC)* on page 7-88
- *64-bit wide external data bus 128M SDRAM (4Mx32, RBC)* on page 7-89
- *64-bit wide external data bus 128M SDRAM (8Mx16, RBC)* on page 7-90

- *64-bit wide external data bus 256M SDRAM (8Mx32, RBC)* on page 7-91
- *64-bit wide external data bus 256M SDRAM (16Mx16, RBC)* on page 7-92
- *64-bit wide external data bus 512M SDRAM (32Mx16, RBC)* on page 7-93.

64-bit wide external data bus 16M SDRAM (1Mx16, RBC)

Table 7-69 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 16M SDRAM (1Mx16, pin 11 used as bank select).

Table 7-69 Address mapping for 16M SDRAM (1Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A2	A2
28	-	-	13	A1	A1
27	-	-	12	A0	A0
26	-	-	11	A13	BA
25	-	-	10	A7	A7
24	-	-	9	A6	A6
23	-	-	8	A5	A5
22	A10	AP	7	A4	A4
21	A9	A9	6	A3	A3
20	A8	A8	5	A2	A2
19	A7	A7	4	A1	A1
18	A6	A6	3	A0	A0
17	A5	A5	2	-	-
16	A4	A4	1	-	-
15	A3	A3	0	-	-

64-bit wide external data bus 64M SDRAM (2Mx32, RBC)

Table 7-70 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (2Mx32, pins 11 and 12 used as bank select).

Table 7-70 Address mapping for 64M SDRAM (2Mx32, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A1	A1
28	-	-	13	A0	A0
27	-	-	12	A14	BA1
26	-	-	11	A13	BA0
25	-	-	10	A7	A7
24	-	-	9	A6	A6
23	A10	AP	8	A5	A5
22	A9	A9	7	A4	A4
21	A8	A8	6	A3	A3
20	A7	A7	5	A2	A2
19	A6	A6	4	A1	A1
18	A5	A5	3	A0	A0
17	A4	A4	2	-	-
16	A3	A3	1	-	-
15	A2	A2	0	-	-

64-bit wide external data bus 64M SDRAM (4Mx16, RBC)

Table 7-71 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 64M SDRAM (4Mx16, pins 11 and 12 used as bank select).

Table 7-71 Address mapping for 64M SDRAM (4Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A1	A1
28	-	-	13	A0	A0
27	-	-	12	A14	BA1
26	-	-	11	A13	BA0
25	-	-	10	A7	A7
24	A11	A11	9	A6	A6
23	A10	AP	8	A5	A5
22	A9	A9	7	A4	A4
21	A8	A8	6	A3	A3
20	A7	A7	5	A2	A2
19	A6	A6	4	A1	A1
18	A5	A5	3	A0	A0
17	A4	A4	2	-	-
16	A3	A3	1	-	-
15	A2	A2	0	-	-

64-bit wide external data bus 128M SDRAM (4Mx32, RBC)

Table 7-72 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (4Mx32, pins 11 and 12 used as bank select).

Table 7-72 Address mapping for 128M SDRAM (4Mx32, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A1	A1
28	-	-	13	A0	A0
27	-	-	12	A14	BA1
26	-	-	11	A13	BA0
25	-	-	10	A7	A7
24	A11	A11	9	A6	A6
23	A10	AP	8	A5	A5
22	A9	A9	7	A4	A4
21	A8	A8	6	A3	A3
20	A7	A7	5	A2	A2
19	A6	A6	4	A1	A1
18	A5	A5	3	A0	A0
17	A4	A4	2	-	-
16	A3	A3	1	-	-
15	A2	A2	0	-	-

64-bit wide external data bus 128M SDRAM (8Mx16, RBC)

Table 7-73 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M SDRAM (8Mx16, pins 12 and 13 used as bank select).

Table 7-73 Address mapping for 128M SDRAM (8Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A0	A0
28	-	-	13	A14	BA1
27	-	-	12	A13	BA0
26	-	-	11	A8	A8
25	A11	A11	10	A7	A7
24	A10	AP	9	A6	A6
23	A9	A9	8	A5	A5
22	A8	A8	7	A4	A4
21	A7	A7	6	A3	A3
20	A6	A6	5	A2	A2
19	A5	A5	4	A1	A1
18	A4	A4	3	A0	A0
17	A3	A3	2	-	-
16	A2	A2	1	-	-
15	A1	A1	0	-	-

64-bit wide external data bus 256M SDRAM (8Mx32, RBC)

Table 7-74 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (8Mx32, pins 11 and 12 used as bank select).

Table 7-74 Address mapping for 256M SDRAM (8Mx32, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A1	A1
28	-	-	13	A0	A0
27	-	-	12	A14	BA1
26	-	-	11	A13	BA0
25	A12	A12	10	A7	A7
24	A11	A11	9	A6	A6
23	A10	AP	8	A5	A5
22	A9	A9	7	A4	A4
21	A8	A8	6	A3	A3
20	A7	A7	5	A2	A2
19	A6	A6	4	A1	A1
18	A5	A5	3	A0	A0
17	A4	A4	2	-	-
16	A3	A3	1	-	-
15	A2	A2	0	-	-

64-bit wide external data bus 256M SDRAM (16Mx16, RBC)

Table 7-75 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M SDRAM (16Mx16, pins 12 and 13 used as bank select).

Table 7-75 Address mapping for 256M SDRAM (16Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A0	A0
28	-	-	13	A14	BA1
27	-	-	12	A13	BA0
26	A12	A12	11	A8	A8
25	A11	A11	10	A7	A7
24	A10	AP	9	A6	A6
23	A9	A9	8	A5	A5
22	A8	A8	7	A4	A4
21	A7	A7	6	A3	A3
20	A6	A6	5	A2	A2
19	A5	A5	4	A1	A1
18	A4	A4	3	A0	A0
17	A3	A3	2	-	-
16	A2	A2	1	-	-
15	A1	A1	0	-	-

64-bit wide external data bus 512M SDRAM (32Mx16, RBC)

Table 7-76 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M SDRAM (32Mx16, pins 13 and 14 used as bank select).

Table 7-76 Address mapping for 512M SDRAM (32Mx16, RBC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:29	-	-	14	A14	BA1
28	-	-	13	A13	BA0
27	A12	A12	12	A9	A9
26	A11	A11	11	A8	A8
25	A10	AP	10	A7	A7
24	A9	A9	9	A6	A6
23	A8	A8	8	A5	A5
22	A7	A7	7	A4	A4
21	A6	A6	6	A3	A3
20	A5	A5	5	A2	A2
19	A4	A4	4	A1	A1
18	A3	A3	3	A0	A0
17	A2	A2	2	-	-
16	A1	A1	1	-	-
15	A0	A0	0	-	-

7.6.7 32-bit wide data bus address mappings (BSRSC) (V-SyncFlash)

The 32-bit wide data bus address mappings for V-SyncFlash devices in a *Bank, Segment, Row, Segment, Column* (BSRSC) mapping scheme are shown in:

- *32-bit wide external data bus 128M V-SyncFlash (4Mx32, BSRSC)* on page 7-94
- *32-bit wide external data bus 128M V-SyncFlash (8Mx16, BSRSC)* on page 7-95
- *32-bit wide external data bus 256M V-SyncFlash (8Mx32, BSRSC)* on page 7-96
- *32-bit wide external data bus 256M V-SyncFlash (16Mx16, BSRSC)* on page 7-97
- *32-bit wide external data bus 512M V-SyncFlash (16Mx32, BSRSC)* on page 7-98

- *32-bit wide external data bus 512M V-SyncFlash (32Mx16, BSRSC)* on page 7-99.

32-bit wide external data bus 128M V-SyncFlash (4Mx32, BSRSC)

Table 7-77 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M V-SyncFlash (4Mx32, pins 22 and 23 used as bank select).

Table 7-77 Address mapping for 128M V-SyncFlash (4Mx32, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A5	A5
26	-	-	12	A4	A4
25	-	-	11	A3	A3
24	-	-	10	A2	A2
23	A14	BA1	9	A1	A1
22	A13	BA0	8	A0	A0
21	A17	SA2	7	A15	SA0
20	A16	SA1	6	A4	A4
19	A11	A11	5	A3	A3
18	A10	A10	4	A2	A2
17	A9	A9	3	A1	A1
16	A8	A8	MPMC	A0	A0
15	A7	A7	1	-	-
14	A6	A6	0	-	-

32-bit wide external data bus 128M V-SyncFlash (8Mx16, BSRSC)

Table 7-78 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M V-SyncFlash (8Mx16, pins 23 and 24 used as bank select).

Table 7-78 Address mapping for 128M V-SyncFlash (8Mx16, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A4	A4
26	-	-	12	A3	A3
25	-	-	11	A2	A2
24	A14	BA1	10	A1	A1
23	A13	BA0	9	A0	A0
22	A17	SA2	8	A15	SA0
21	A16	SA1	7	A5	A5
20	A11	A11	6	A4	A4
19	A10	A10	5	A3	A3
18	A9	A9	4	A2	A2
17	A8	A8	3	A1	A1
16	A7	A7	MPMC	A0	A0
15	A6	A6	1	-	-
14	A5	A5	0	-	-

32-bit wide external data bus 256M V-SyncFlash (8Mx32, BSRSC)

Table 7-79 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M V-SyncFlash (8Mx32, pins 23 and 24 used as bank select).

Table 7-79 Address mapping for 256M V-SyncFlash (8Mx32, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A5	A5
26	-	-	12	A4	A4
25	-	-	11	A3	A3
24	A14	BA1	10	A2	A2
23	A13	BA0	9	A1	A1
22	A17	SA2	8	A0	A0
21	A16	SA1	7	A15	SA0
20	A12	A12	6	A4	A4
19	A11	A11	5	A3	A3
18	A10	A10	4	A2	A2
17	A9	A9	3	A1	A1
16	A8	A8	MPMC	A0	A0
15	A7	A7	1	-	-
14	A6	A6	0	-	-

32-bit wide external data bus 256M V-SyncFlash (16Mx16, BSRSC)

Table 7-80 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M V-SyncFlash (16Mx16, pins 24 and 25 used as bank select).

Table 7-80 Address mapping for 256M V-SyncFlash (16Mx16, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A4	A4
26	-	-	12	A3	A3
25	A14	BA1	11	A2	A2
24	A13	BA0	10	A1	A1
23	A17	SA2	9	A0	A0
22	A16	SA1	8	A15	SA0
21	A12	A12	7	A5	A5
20	A11	A11	6	A4	A4
19	A10	A10	5	A3	A3
18	A9	A9	4	A2	A2
17	A8	A8	3	A1	A1
16	A7	A7	MPMC	A0	A0
15	A6	A6	1	-	-
14	A5	A5	0	-	-

32-bit wide external data bus 512M V-SyncFlash (16Mx32, BSRSC)

Table 7-81 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M V-SyncFlash (16Mx32, pins 24 and 25 used as bank select).

Table 7-81 Address mapping for 512M V-SyncFlash (16Mx32, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A5	A5
26	-	-	12	A4	A4
25	A14	BA1	11	A3	A3
24	A13	BA0	10	A2	A2
23	A17	SA2	9	A1	A1
22	A16	SA1	8	A0	A0
21	A18	A13	7	A15	SA0
20	A12	A12	6	A4	A4
19	A11	A11	5	A3	A3
18	A10	A10	4	A2	A2
17	A9	A9	3	A1	A1
16	A8	A8	MPMC	A0	A0
15	A7	A7	1	-	-
14	A6	A6	0	-	-

32-bit wide external data bus 512M V-SyncFlash (32Mx16, BSRSC)

Table 7-82 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M V-SyncFlash (32Mx16, pins 25 and 26 used as bank select).

Table 7-82 Address mapping for 512M V-SyncFlash (32Mx16, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A4	A4
26	A14	BA1	12	A3	A3
25	A13	BA0	11	A2	A2
24	A17	SA2	10	A1	A1
23	A16	SA1	9	A0	A0
22	A18	A13	8	A15	SA0
21	A12	A12	7	A5	A5
20	A11	A11	6	A4	A4
19	A10	A10	5	A3	A3
18	A9	A9	4	A2	A2
17	A8	A8	3	A1	A1
16	A7	A7	MPMC	A0	A0
15	A6	A6	1	-	-
14	A5	A5	0	-	-

7.6.8 16-bit wide data bus address mappings (BSRSC) (V-SyncFlash)

The 16-bit wide data bus address mappings for V-SyncFlash devices in a BSRSC mapping scheme are shown in:

- *16-bit wide external data bus 128M V-SyncFlash (8Mx16, BSRSC)* on page 7-100
- *16-bit wide external data bus 256M V-SyncFlash (16Mx16, BSRSC)* on page 7-101
- *16-bit wide external data bus 512M V-SyncFlash (32Mx16, BSRSC)* on page 7-102.

16-bit wide external data bus 128M V-SyncFlash (8Mx16, BSRSC)

Table 7-83 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M V-SyncFlash (8Mx16, pins 22 and 23 used as bank select).

Table 7-83 Address mapping for 128M V-SyncFlash (8Mx16, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A5	A5
26	-	-	12	A4	A4
25	-	-	11	A3	A3
24	-	-	10	A2	A2
23	A14	BA1	9	A1	A1
22	A13	BA0	8	A0	A0
21	A17	SA2	7	A15	SA0
20	A16	SA1	6	A5	A5
19	A11	A11	5	A4	A4
18	A10	A10	4	A3	A3
17	A9	A9	3	A2	A2
16	A8	A8	MPMC	A1	A1
15	A7	A7	MPMC	A0	A0
14	A6	A6	0	-	-

16-bit wide external data bus 256M V-SyncFlash (16Mx16, BSRSC)

Table 7-84 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M V-SyncFlash (16Mx16, pins 23 and 24 used as bank select).

Table 7-84 Address mapping for 256M V-SyncFlash (16Mx16, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A5	A5
26	-	-	12	A4	A4
25	-	-	11	A3	A3
24	A14	BA1	10	A2	A2
23	A13	BA0	9	A1	A1
22	A17	SA2	8	A0	A0
21	A16	SA1	7	A15	SA0
20	A12	A12	6	A5	A5
19	A11	A11	5	A4	A4
18	A10	A10	4	A3	A3
17	A9	A9	3	A2	A2
16	A8	A8	MPMC	A1	A1
15	A7	A7	MPMC	A0	A0
14	A6	A6	0	-	-

16-bit wide external data bus 512M V-SyncFlash (32Mx16, BSRSC)

Table 7-85 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M V-SyncFlash (32Mx16, pins 25 and 26 used as bank select).

Table 7-85 Address mapping for 512M V-SyncFlash (32Mx16, BSRSC)

AHB address HADDR[31:14]	MPMC output address	Memory device connections	AHB address HADDR[13:0]	MPMC output address	Memory device connections
31:27	-	-	13	A5	A5
26	-	-	12	A4	A4
25	A14	BA1	11	A3	A3
24	A13	BA0	10	A2	A2
23	A17	SA2	9	A1	A1
22	A16	SA1	8	A0	A0
21	A18	A13	7	A15	SA0
20	A12	A12	6	A5	A5
19	A11	A11	5	A4	A4
18	A10	A10	4	A3	A3
17	A9	A9	3	A2	A2
16	A8	A8	MPMC	A1	A1
15	A7	A7	MPMC	A0	A0
14	A6	A6	0	-	-

7.6.9 64-bit wide data bus address mappings (BSRSC) (V-SyncFlash)

The 64-bit wide data bus address mappings for V-SyncFlash devices in a BSRSC mapping scheme are shown in:

- *64-bit wide external data bus 128M V-SyncFlash (4Mx32, BSRSC)* on page 7-103
- *64-bit wide external data bus 128M V-SyncFlash (8Mx16, BSRSC)* on page 7-104
- *64-bit wide external data bus 256M V-SyncFlash (8Mx32, BSRSC)* on page 7-105
- *64-bit wide external data bus 256M V-SyncFlash (16Mx16, BSRSC)* on page 7-106
- *64-bit wide external data bus 512M V-SyncFlash (16Mx32, BSRSC)* on page 7-107

- *64-bit wide external data bus 512M V-SyncFlash (32Mx16, BSRSC)* on page 7-107.

64-bit wide external data bus 128M V-SyncFlash (4Mx32, BSRSC)

Table 7-86 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M V-SyncFlash (4Mx32, pins 23 and 24 used as bank select).

Table 7-86 Address mapping for 128M V-SyncFlash (4Mx32, BSRSC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A5	A5
27	-	-	13	A4	A4
26	-	-	12	A3	A3
25	-	-	11	A2	A2
24	A14	BA1	10	A1	A1
23	A13	BA0	9	A0	A0
22	A17	SA2	8	A15	SA0
21	A16	SA1	7	A4	A4
20	A11	A11	6	A3	A3
19	A10	A10	5	A2	A2
18	A9	A9	4	A1	A1
17	A8	A8	3	A0	A0
16	A7	A7	2	-	-
15	A6	A6	1:0	-	-

64-bit wide external data bus 128M V-SyncFlash (8Mx16, BSRSC)

Table 7-87 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 128M V-SyncFlash (8Mx16, pins 24 and 25 used as bank select).

Table 7-87 Address mapping for 128M V-SyncFlash (8Mx16, BSRSC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	-	-	12	A2	A2
25	A14	BA1	11	A1	A1
24	A13	BA0	10	A0	A0
23	A17	SA2	9	A15	SA0
22	A16	SA1	8	A5	A5
21	A11	A11	7	A4	A4
20	A10	A10	6	A3	A3
19	A9	A9	5	A2	A2
18	A8	A8	4	A1	A1
17	A7	A7	3	A0	A0
16	A6	A6	2	-	-
15	A5	A5	1:0	-	-

64-bit wide external data bus 256M V-SyncFlash (8Mx32, BSRSC)

Table 7-88 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M V-SyncFlash (8Mx32, pins 24 and 25 used as bank select).

Table 7-88 Address mapping for 256M V-SyncFlash (8Mx32, BSRSC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A5	A5
27	-	-	13	A4	A4
26	-	-	12	A3	A3
25	A14	BA1	11	A2	A2
24	A13	BA0	10	A1	A1
23	A17	SA2	9	A0	A0
22	A16	SA1	8	A15	SA0
21	A12	A12	7	A4	A4
20	A11	A11	6	A3	A3
19	A10	A10	5	A2	A2
18	A9	A9	4	A1	A1
17	A8	A8	3	A0	A0
16	A7	A7	2	-	-
15	A6	A6	1:0	-	-

64-bit wide external data bus 256M V-SyncFlash (16Mx16, BSRSC)

Table 7-89 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 256M V-SyncFlash (16Mx16, pins 25 and 26 used as bank select).

Table 7-89 Address mapping for 256M V-SyncFlash (16Mx16, BSRSC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	-	-	13	A3	A3
26	A14	BA1	12	A2	A2
25	A13	BA0	11	A1	A1
24	A17	SA2	10	A0	A0
23	A16	SA1	9	A15	SA0
22	A12	A12	8	A5	A5
21	A11	A11	7	A4	A4
20	A10	A10	6	A3	A3
19	A9	A9	5	A2	A2
18	A8	A8	4	A1	A1
17	A7	A7	3	A0	A0
16	A6	A6	2	-	-
15	A5	A5	1:0	-	-

64-bit wide external data bus 512M V-SyncFlash (16Mx32, BSRSC)

Table 7-90 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M V-SyncFlash (16Mx32, pins 25 and 26 used as bank select).

Table 7-90 Address mapping for 512M V-SyncFlash (16Mx32, BSRSC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A5	A5
27	-	-	13	A4	A4
26	A14	BA1	12	A3	A3
25	A13	BA0	11	A2	A2
24	A17	SA2	10	A1	A1
23	A16	SA1	9	A0	A0
22	A18	A13	8	A15	SA0
21	A12	A12	7	A4	A4
20	A11	A11	6	A3	A3
19	A10	A10	5	A2	A2
18	A9	A9	4	A1	A1
17	A8	A8	3	A0	A0
16	A7	A7	2	-	-
15	A6	A6	1:0	-	-

64-bit wide external data bus 512M V-SyncFlash (32Mx16, BSRSC)

Table 7-91 on page 7-108 shows the outputs from the PrimeCell MPMC and the corresponding inputs to the 512M V-SyncFlash (32Mx16, pins 26 and 27 used as bank select).

Table 7-91 Address mapping for 512M V-SyncFlash (32Mx16, BSRSC)

AHB address HADDR[31:15]	MPMC output address	Memory device connections	AHB address HADDR[14:0]	MPMC output address	Memory device connections
31:28	-	-	14	A4	A4
27	A14	BA1	13	A3	A3
26	A13	BA0	12	A2	A2
25	A17	SA2	11	A1	A1
24	A16	SA1	10	A0	A0
23	A18	A13	9	A15	SA0
22	A12	A12	8	A5	A5
21	A11	A11	7	A4	A4
20	A10	A10	6	A3	A3
19	A9	A9	5	A2	A2
18	A8	A8	4	A1	A1
17	A7	A7	3	A0	A0
16	A6	A6	2	-	-
15	A5	A5	1:0	-	-

7.7 SDRAM refreshing

The **MPMCAPOUT** auto precharge output is provided to enable SDRAM devices to be refreshed when:

- the MPMC does not have control of the external address bus
- a NAND flash or SRAM access is being performed.

This output must be connected to the auto precharge bit of the SDRAM device address bus input (A8 or A10 depending on the device) instead of the relevant bit of the external address bus. The output is driven according to the device currently being refreshed, so can be connected to several devices that use both address bits A8 and A10 for auto precharging. All other bits of the device address input must be connected as normal to the external address bus.

——— **Note** ———

Address bus bits A10 and A8 must not be used to drive the auto precharge input of SDRAM devices. The **MPMCAPOUT** auto precharge output is normally used for this purpose.

Micron SyncFlash and V-SyncFlash devices use the auto precharge information during the Active Terminate command to determine whether all banks are terminated. Because of this, the **MPMCAPOUT** auto precharge output must be connected to all Micron SyncFlash and V-SyncFlash devices in the system, instead of the A10 address bit.

7.8 Merged preamble and postamble

For back-to-back transfers, the memory controller joins the postamble from the first transfer to the preamble for the second transfer, creating a combined postamble and preamble period. The **MPMCDQSOUT** outputs are not deasserted during this combined period.

Some memory devices specify a maximum postamble time, which might be violated for back-to-back transfers. This maximum postamble time is not a physical device requirement, but is a system issue because extended postambles can lead to long bus turnaround times when multiple devices are accessing the memory. This is not a problem if the PrimeCell MPMC is the only device driving the external memory bus.

For a single transfer, the **MPMCDQSOUT** outputs are only held for the maximum postamble time, and are not extended.

7.9 Dynamic memory controller command descriptions

The dynamic memory controller block in the PrimeCell MPMC supports the SDRAM memory commands shown in Table 7-92 and Table 7-93.

The commands listed in Table 7-92 are generated automatically.

Table 7-92 Synchronous memory commands used by MPMC

Mnemonic	Operation
ACT	Opens an SDRAM row
REF	CAS before RAS style refresh
SREF	Self-refresh
PRE	Precharge, closes a bank
RD	Read from an open row, row left open
WR	Write to an open row, row left open

The commands in Table 7-93 are generated under software control by programming the SDRAM initialization (I) and deep-sleep mode (DP) fields of the MPMCDynamicControl Register.

Table 7-93 Synchronous memory commands programmed by software

Mnemonic	Operation
MRS	Mode register set, programs SDRAM mode register
NOP	No operation, used during the SDRAM initialization sequence
PALL	Precharge all, used during the SDRAM initialization sequence
DSM	Deep sleep mode, for low-power SDRAM

7.10 Generic SDRAM initialization example

On power-on reset, **nPOR**, software must initialize the MPMC and each of the synchronous memories connected to the controller. Check the dynamic memory data sheet for the start up procedure, an example sequence is:

1. Wait 100µs after the power is applied and the clocks have stabilized.
2. Bring the MPMC out of self-refresh state, using either the power man control signals, or software programming of the self-refresh bits of the MPMCDynamicControl Register.
3. Set the SDRAM Initialization (I) value to NOP in the MPMCDynamicControl Register. This automatically issues a NOP to the SDRAM memories.
4. Wait 200µs.
5. Set the SDRAM Initialization (I) value to PALL in the MPMCDynamicControl Register. This automatically issues a precharge all instruction (PRE-ALL) to the SDRAM memories. This precharges all banks and places the device into the all banks idle state.
6. Perform several refresh cycles, by writing 1 into the MPMCDynamicRefresh Register. This provides a memory refresh every 16 AHB clock cycles.
7. Wait until eight SDRAM refresh cycles have occurred (128 AHB clock cycles).
8. Program the operational value into the refresh register, MPMCDynamicRefresh.
9. Program the operational value into the latency register, MPMCDynamicRasCas.
10. Program the operational values into the MPMCDynamicConfig Register. The buffers must be disabled during initialization.
11. Set the SDRAM initialization value (I) to MODE in the MPMCDynamicControl Register.
12. Program the SDRAM memories mode register. The mode register enables the following parameters to be programmed:
    - burst length
    - burst type
    - CAS latency
    - operating mode
    - write burst mode.

    Table 7-94 on page 7-113 shows the mode register settings.

A read transaction from the SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register. The row address bits contain the value to be programmed. The bank select signals **BA0** and **BA1** must both be 0 to program the mode register.

The SDRAM mode register values shown in Table 7-94 must be used.

Table 7-94 SDRAM mode register settings for generic SDRAM

Option	Value
Burst length	2 for a 32-bit wide external data bus, or 4 for a 16-bit wide external data bus
Burst type	Sequential
CAS latency	Dependent on operating frequency
Operating mode	Standard operation
Write burst mode	Programmed burst length

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

13. Set the SDRAM Initialization (I) value to NORMAL in the MPMCDynamicControl Register.

7.11 Micron MT48LC4M16A2 SDRAM initialization example

You can use the following procedure to initialize two Micron MT48LC4M16A2 SDRAM (64Mb, 4Mb x 16) devices, speed grade -8E, configured to provide a 32-bit bus. **HCLK** and **MPMCLK** are 100MHz:

1. Wait 100μs after the power is applied and the clocks have stabilized.
2. Set the SDRAM Initialization (I) value to NOP in the MPMCDynamicControl Register. This automatically issues a NOP to the SDRAM memories.
3. Set the SDRAM Initialization (I) value to PALL in the MPMCDynamicControl Register. This automatically issues a precharge all instruction, PRE-ALL, to the SDRAM memories. This precharges all the banks and places the device into the all banks idle state.
4. Write 2 into the refresh timer register, MPMCDynamicRefresh. This provides a refresh cycle every 32 AHB clock cycles.
5. Wait until two SDRAM refresh cycles have occurred (64 AHB clock cycles).
6. Program the operational value into the refresh register, MPMCDynamicRefresh. This device requires a memory refresh every 15.625μs. Therefore, with a 100MHz **HCLK** the refresh register must be programmed with (15.625μs x 100MHz)/16 = 97.
7. Program the operational value into the latency register, MPMCDynamicRasCas. The -8E speed grade devices support CAS latency 2 at 100MHz operation. Therefore, `0x0202` must be programmed into the register.
8. Program the operational values into the configuration register, MPMCDynamicConfig. The buffers must be disabled during initialization. For this memory device the fields must be set as shown in Table 7-95.

Table 7-95 Field settings for Micron MT48LC4M16A2 SDRAM

Field	Value
Memory device (MD)	SDRAM (000)
Address mapping (AM)	32-bit bus, 64Mb, 4Mb x 16 devices, RBC mapping (010000101)
Write protect (P)	Writes not protected (0)

The value `0x00004280` is required to program the MPMCDynamicConfig Register for this device.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

9. Set the SDRAM Initialization (I) value to MODE in the MPMCDynamicControl Register.

10. Program the SDRAM memories mode register. The mode register enables the following parameters to be programmed:
    - burst length
    - burst type
    - CAS latency
    - operating mode
    - write burst mode.

    Table 7-96 shows the mode register settings.

    A read transaction from the SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register, in this case 32-bit, 64M, 4M x 16, RBC (010000101). The row address bits contain the value to be programmed. The bank select signals **BA0** and **BA1** must both be 0 to program the mode register.

Table 7-96 SDRAM mode register settings for Micron SDRAM

Option	Value	Micron device
Burst length	2 (for 16-bit data bus)	**A[2:0]**=001
Burst type	Sequential	**A[3]**=0
CAS latency	2 (for -8E device @100MHz)	**A[6:4]**=010
Operating mode	Standard operation	**A[8:7]**=00
Write burst mode	Programmed burst length	**A[9]**=0
Reserved	0	**A[11:10]**=00

The value required to program the SDRAM mode register to this configuration is `0x021`.

The **HADDR** to SDRAM memory address mapping is 32-bit 64M SDRAM (4M x 16, RBC). The address mapping tables are shown in *Address mapping* on page 7-30. You can see the SDRAM memory row address bits are mapped to **HADDR [23:12]**. The SDRAM memory bank address bits are mapped to **HADDR [11:10]**.

The address to be accessed is `0x20000`.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

11. Set the SDRAM Initialization (I) value to NORMAL in the MPMCDynamicControl Register.

7.12 Micron MT46V8M16 DDR-SDRAM initialization example

You can use the following procedure to initialize Micron MT46V8M16 128Mb (8 M x 16) DDR-SDRAM devices configured to provided a 16-bit bus. In this example **HCLK** and **MPMCLK** are 100MHz:

1. Wait 200μs after the power is applied and the clocks have stabilized.
2. Set the SDRAM Initialization (I) value to NOP in the MPMCDynamicControl Register. This automatically issues a NOP to the SDRAM memories.
3. **CKE** must then be brought HIGH by setting the CKE bit in the MPMCDynamicControl Register.
4. Set the SDRAM Initialization (I) value to PALL in the MPMCDynamicControl Register. This automatically issues a precharge all instruction, PRE-ALL, to the SDRAM memories. This precharges all the banks and places the device into the all banks idle state.
5. Program the operational value into the latency register, MPMCDynamicRasCas. The -8 speed grade devices support CAS latency 2 at 100MHz operation. Therefore `0x0202` must be programmed into the register.
6. Program the operational values into the configuration register, MPMCDynamicConfig. The buffers must be disabled during initialization. For this memory device the fields must be set as shown in Table 7-97.

Table 7-97 Field settings for Micron MT48LC4M16A2 SDRAM

Field	Value
Memory device (MD)	DDR-SDRAM (100)
Address mapping (AM)	16-bit bus, 128Mb, 8M x 16 devices, RBC mapping (000001001)
Write protect (P)	Writes not protected (0)

The value `0x00000484` is required to program the MPMCDynamicConfig Register for this device.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

7. Set the SDRAM Initialization (I) value to MODE in the MPMCDynamicControl Register.
8. The DDR-SDRAM memories extended mode register must be programmed to enable the DLL. The extended mode register enables the following parameters to be programmed:
   - DLL enable
   - drive strength
   - QFC control
   - operating mode.

   Table 7-98 shows the mode register settings.The bank select signals **BA1** and **BA0** must be 0 and 1 to select the extended mode register. A read transaction from the DDR-SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register, in this case 16-bit, 128Mb, 8M x 16, RBC. The row address bits contain the value to be programmed.

   Settings for the Micron device are shown in Table 7-98.

Table 7-98 SDRAM extended mode register settings for Micron DDR-SDRAM

Option	Value	Micron device
DLL	Enable	**A[0]**=0
Drive strength	Normal	**A[1]**=0
QFC	Disabled	**A[2]**=0
Operating mode	Normal operation	**A[11:3]**=00000000

The value required to program the DDR-SDRAM extended mode register to this configuration is `0x00`.

The **HADDR** to SDRAM memory address mapping is 16-bit 128Mb SDRAM (8M x 16, RBC). The address mapping tables are shown in *Address mapping* on page 7-30. You can see that the SDRAM memory row address bits are mapped to **HADDR [23:12]**. The SDRAM memory bank address bits are mapped to **HADDR [11:10]**.

Therefore, the address to be accessed is `0x00400`.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

9. The DDR-SDRAM memories mode register must be programmed to reset the DLL and to program the operating parameters. The mode register enables the following parameters to be programmed:
   - burst length
   - burst type
   - CAS latency
   - operating mode
   - write burst mode

   Table 7-99 shows the mode register settings.

   A read transaction from the SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register, in this case 16-bit, 128Mb, 8M x 16, RBC. The row address bits contain the value to be programmed. The bank select signals **BA0** and **BA1** must both be 0 to program the mode register

   Settings for the Micron device are shown in Table 7-99.

Table 7-99 SDRAM mode register settings for Micron DDR-SDRAM

Option	Value	Micron device
Burst length	4 (for 16-bit data bus)	**A[2:0]**=010
Burst type	Sequential	**A[3]**=0
CAS latency	2 (for -8 device @100MHz)	**A[6:4]**= 010
Operating mode	Normal operation/reset DLL	**A[11:7]**=00010

The value required to program the DDR-SDRAM mode register to this configuration is `0x122`.

The **HADDR** to SDRAM memory address mapping is 16-bit 128Mb SDRAM (8M x 16, RBC). The address mapping tables are shown in *Address mapping* on page 7-30. You can see that the SDRAM memory row address bits are mapped to **HADDR [23:12]**. The SDRAM memory bank address bits are mapped to **HADDR [11:10]**.

Therefore, the address used must be `0x121000`.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

10. Wait 200 AHB clock cycles.
11. Set the SDRAM initialization (I) value to PALL in the MPMCDynamicControl Register. This automatically issues a precharge all instruction, PRE-ALL, to the SDRAM memories. This precharges all the banks and places the device into the all banks idle state.
12. Write 2 into the refresh timer register, MPMCDynamicRefresh. This provides a refresh cycle every 32 AHB clock cycles.
13. Wait until two SDRAM refresh cycles occur (64 AHB clock cycles).
14. Program the operational value into the refresh register, MPMCDynamicRefresh. This device requires a memory refresh every 15.625µs. Therefore, with a 100MHz **HCLK** the refresh register must be programmed with (15.625µs x 100MHz)/16 = 97.
15. Set the SDRAM initialization (I) value to MODE in the MPMCDynamicControl Register.
16. The DDR-SDRAM memories mode register must be programmed with the reset DLL bit deactivated, the other operating parameters must be the same as before.

    The address to be accessed is therefore `0x021000`.
17. Set the SDRAM initialization (I) value to NORMAL in the MPMCDynamicControl Register. The SDRAM is now ready for normal operation.

7.13 Low-power SDRAM initialization example

The following procedure can be used to initialize Infineon HYB/E 25L128160AC-A 128Mb (8Mb x 16) devices configured to provide a 16-bit bus. **HCLK** and **MPMCLK** are 50MHz:

1. Wait 200µs after the power is applied and the clocks have stabilized.
2. Set the SDRAM Initialization (I) value to PALL in the MPMCDynamicControl Register. This automatically issues a precharge all instruction (PRE-ALL) to the SDRAM memories. This precharges all the banks and places the device into the all banks idle state.
3. Perform several refresh cycles, by writing 2 into the refresh register, MPMCDynamicRefresh. This provides a memory refresh every 32 AHB clock cycles.
4. Wait until eight SDRAM refresh cycles have occurred (256 AHB clock cycles).
5. Program the operational value into the refresh register, MPMCDynamicRefresh. This device requires a memory refresh every 64µs, therefore with a 100MHz **HCLK** the refresh register must be programmed with (64µs x100MHz)/16 = 400.
6. Program the operational value into the latency register, MPMCDynamicRasCas. The -8 speed grade devices support CAS latency 2 at 100MHz operation. Therefore, `0x0202` must be programmed into the register.
7. Program the operational values into the configuration register, MPMCDynamicConfig. The buffers must be disabled during initialization. For this memory device the fields must be set as shown in Table 7-100.

Table 7-100 Field settings for Micron MT48LC4M16A2 SDRAM

Field	Value
Memory device (MD)	Low-power SDRAM (010)
Address mapping (AM)	16-bit bus, 128Mb, 8M x 16 devices, RBC mapping (000001001)
Write protect (P)	Writes not protected (0)

The value `0x00000482` is required to program the MPMCDynamicConfig Register for this device.

——— **Note** ———

- The AHB memory port must be used to perform this transaction.

- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

8. Set the SDRAM Initialization (I) value to MODE in the MPMCDynamicControl Register.
9. The SDRAM memories mode register must be programmed. The mode register enables the following parameters to be programmed:
   - burst length
   - burst type
   - CAS latency
   - operating mode
   - write burst mode.

   Table 7-101 shows the mode register settings.

   A read transaction from the SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register, in this case 16-bit, 128Mb, 8M x 16, RBC. The row address bits contain the value to be programmed. The bank select signals **BA0** and **BA1** must both be 0 to program the mode register

   The settings for the low-power device are shown in Table 7-101.

Table 7-101 SDRAM mode register settings for low-power SDRAM

Option	Value	Micron device
Burst length	4 (for 16-bit data bus)	**A[2:0]**=010
Burst type	Sequential	**A[3]**=0
CAS latency	2 (for -8E device @100MHz)	**A[6:4]**=010
Operating mode	Standard operation	**A[11:7]**=00000

The value required to program the low-power SDRAM memories mode register is `0x022`.

The **HADDR** to SDRAM memory address mapping is 16-bit 128Mb SDRAM (8M x 16, RBC). The address mapping tables are shown in *Address mapping* on page 7-30. You can see the SDRAM memory row address bits are mapped to **HADDR [23:12]**. The SDRAM memory bank address bits are mapped to **HADDR [11:10]**.

Therefore, the address used must be `0x21000`.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

10. The low-power SDRAM memories extended mode register must be programmed. The mode register enables the following parameters to be programmed:
    - temperature compensated self-refresh
    - partial array self-refresh.

    The bank select signals **BA1** and **BA0** must be 1, 0 to select the extended mode register. A read transaction from the SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register, in this case 16-bit, 128Mb, 8M x 16, RBC. The row address bits contain the value to be programmed.

11. Table 7-102 shows the SDRAM extended mode register settings.

Table 7-102 Low power SDRAM extended mode register settings

Option	Value	Micron device
Partial array self-refresh	All banks	**A[2:0]**=000
Temperature compensated self-refresh	70°C	**A[4:3]**=00

The value required to program the low-power SDRAM devices extended mode register is `0x00`.

Therefore the address used must be `0x00`.

The **HADDR** to SDRAM memory address mapping is 16-bit 128Mb SDRAM (8M x 16, RBC). The address mapping tables are shown in *Address mapping* on page 7-30. You can see that the SDRAM memory row address bits are mapped to **HADDR [23:12]**. The SDRAM memory bank address bits are mapped to **HADDR [11:10]**.

The address to be accessed is therefore `0x00000`.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

12. Set the SDRAM Initialization (I) value to NORMAL in the MPMCDynamicControl Register.

7.14 Micron MT28F4M16S2 SyncFlash initialization example

You can use the following procedure to initialize Micron MT28S4M16LC SyncFlash (64Mb, 4M x 16) devices configured to provide a 16-bit bus. **HCLK** and **MPMCLK** are 100MHz:

1. Wait for the power to be applied and the clocks have stabilized.
2. Set the RP value in the MPMCDynamicControl Register HIGH.
3. Wait 100µs.
4. Program the operational value into the latency register, MPMCDynamicRasCas. The -10 speed grade devices support CAS latency 3 at 100MHz operation. Therefore `0x0303` must be programmed into the register.
5. Program the operational values into the configuration register, MPMCDynamicConfig. The buffers must be disabled during initialization. For this memory device the fields must be set as shown in Table 7-103.

Table 7-103 Field settings for Micron MT48LC4M16A2 SDRAM

Field	Value
Memory Device (MD)	Micron SyncFlash (001)
Address Mapping (AM)	64Mb, 4M x 16 devices, RBC mapping (000000101)
Write Protect (P)	Writes not protected (0)

The value `0x00000281` is required to program the MPMCDynamicConfig Register for this device.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

6. Set the SDRAM Initialization (I) value to MODE in the MPMCDynamicControl Register.
7. The SyncFlash memories mode register must be programmed. The mode register enables the following parameters to be programmed:
   - burst length
   - burst type

- CAS latency
- operating mode
- write burst mode.

Table 7-104 shows the mode register settings.

A read transaction from the SDRAM memory programs the mode register. The address of the transfer contains the value to be programmed. The mapping from AHB address bus, **HADDR**, to the SDRAM memories address lines depends on the address mapping value selected in the MPMCDynamicConfig Register, in this case 16-bit, 64Mb, 4M x 16, RBC. The row address bits contain the value to be programmed. The bank select signals **BA0** and **BA1** must both be 0 to program the mode register

The settings for the SyncFlash device are shown in Table 7-104.

Table 7-104 SDRAM mode register settings for SyncFlash SDRAM

Option	Value	Micron device
Burst length	4 (for 16-bit data bus)	**A[2:0]**=010
Burst type	Sequential	**A[3]**=0
CAS latency	3 (for -10 device @100MHz)	**A[6:4]**=011
Operating mode	Standard operation	**A[8:7]**=00
Write burst mode	Programmed burst length	**A[9]**=0
Reserved (0)	0	**A[11:10]**=0

The value required to program the Micron SyncFlash SDRAM memories mode register is `0x032`.

The **HADDR** to SDRAM memory address mapping is 16-bit 64Mb SDRAM (4M x 16, RBC). The address mapping tables are shown in *Address mapping* on page 7-30. You can see the SDRAM memory row address bits are mapped to **HADDR [22:11]**. The SDRAM memory bank address bits are mapped to **HADDR [9:10]**.

Therefore, the address used must be `0x1C00`.

Note

- The AHB memory port must be used to perform this transaction.
- When initializing the memory device the appropriate chip select must be activated. Therefore, depending on the AHB decoder address map the address programmed might require modifying.

8. Set the SDRAM Initialization (I) value to NORMAL in the MPMCDynamicControl Register.

7.15 Micron SyncFlash and V-SyncFlash commands

Table 7-105 shows the Micron SyncFlash and V-SyncFlash commands supported by the MPMC.

Table 7-105 SyncFlash and V-SyncFlash commands

SyncFlash operation	AHB acces s	HADDR	HWDATA
Read device configuration	Read	CMD1 = 1 CMD0 = 0 Other = row + CA address	N/a
Read status register	Read	CMD1 = 1 CMD0 = 1 Other = row address	N/a
Clear status register	Write	CMD1 = 1 Other = don't care	`0xXXXX50XX` (`X` = don't care)
Erase setup/confirm	Write	CMD1 = 1 Other = bank and row address	`0xXXXX20D0`
Program setup/program	Write	CMD1 = 0 Other = row, column and bank address	Din, the data to be written into the array
Protect block/confirm	Write	CMD1 = 1 Other = bank and row address	`0xXXXX6001`
Protect device/confirm	Write	CMD1 = 1 Other = bank address	`0xXXXX60F1`
Unprotect blocks/confirm	Write	CMD1 = 1 Other = bank address	`0xXXXX60D0`
Erase nvmode register	Write	CMD1 = 1 Other = bank address	`0xXXXX30C0`
Program nvmode register	Write	CMD1 = 1 Other = bank address	`0xXXXXA0XX`

To execute the command set, read or write transfers must be performed while the address is set appropriately. The commands can be found in Table 7-105. CMD1 in the table indicates **HADDR[28]** and CMD0 indicates **HADDR[27]**. For example to perform a SyncFlash Protect block or confirm command, a write transfer to the memory must be performed to an address where **HADDR[28]** is HIGH, and the rest of the address indicates the bank and row address. The value `0x6001` must be written to the memory.

——— **Note** ————

The AHB port buffers must be disabled while performing write transfers or executing SyncFlash or V-SyncFlash commands, but can be enabled during standard read transfers.

The commands in Table 7-105 on page 7-128 are not supported in hardware for SyncFlash and V-SyncFlash devices where the standard address mapping uses **HADDR[27]**, because there are not enough spare address bits to drive CMD0 and CMD1. However, you can still generate the device programming commands in software.

Only AHB writes of type single to SyncFlash devices are supported by the MPMC, even if the device supports write bursts. AHB burst writes to SyncFlash devices can result in unpredictable behavior.

The maximum AHB write size supported is the size of the SyncFlash device, although smaller writes are allowed.

7.15.1 Micron SyncFlash programming example

The following subsections describe how some Micron SyncFlash commands can be performed.

Reading the ID register of the manufacturer

To read the ID register of the manufacturer the read device configuration command must be performed. This command requires a read to be performed with the AHB address configured so that:

- the appropriate chip select is selected
- the CMD1 bit is HIGH, and the CMD0 bit is LOW
- the low order address bits are LOW to select the ID register of the manufacturer.

Reading the status register

To read the status register the read status register command must be performed. This command requires a read to be performed with the AHB address configured so that:

- the appropriate chip select is selected
- the CMD1 and CMD0 bits are HIGH
- the low order address bits select the row address.

7.16 Byte lane control and data bus steering

Table 7-106 to Table 7-109 on page 7-133 show the relationship of signals **HSIZE[2:0]** and **HADDR[1:0]** and the mapping of data between a 32-bit AHB port data bus and a 64-bit external memory bus. For 16 and 32-bit external memories, the AHB size, data, and address values shown in *Byte lane control and data bus steering* on page 5-39 for SRAM are applicable. SDRAM does not support single 8-bit devices.

For a 64-bit AHB port, the external byte lanes are mapped directly onto the AHB byte lanes.

Table 7-106 Little-endian read, 64-bit external bus

Internal transfer width	Access: Read, little-endian, 64-bit external bus		External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	HADDR [2:0]	[31:24]	[23:16]	[15:8]	[7:0]
Word	010	1--	[63:56]	[55:48]	[47:40]	[39:32]
Word	010	0--	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	[63:56]	[55:48]	-	-
Halfword	001	10-	-	-	[47:40]	[39:32]
Halfword	001	01-	[31:24]	[23:16]	-	-
Halfword	001	00-	-	-	[15:8]	[7:0]
Byte	000	111	[63:56]	-	-	-
Byte	000	110	-	[55:48]	-	-
Byte	000	101	-	-	[47:40]	-
Byte	000	100	-	-	-	[39:32]
Byte	000	011	[31:24]	-	-	-
Byte	000	010	-	[23:16]	-	-
Byte	000	001	-	-	[15:8]	-
Byte	000	000	-	-	-	[7:0]

Table 7-107 Big-endian read, 64-bit external bus

Internal transfer width	Access: Read, big-endian, 64-bit external bus		External data mapping on to system data bus HRDATA to MPMCDATA			
	HSIZE [2:0]	**HADDR [2:0]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word	010	1--	[63:56]	[55:48]	[47:40]	[39:32]
Word	010	0--	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	-	-	[47:40]	[39:32]
Halfword	001	10-	[63:56]	[55:48]	-	-
Halfword	001	01-	-	-	[15:8]	[7:0]
Halfword	001	00-	[31:24]	[23:16]	-	-
Byte	000	111	-	-	-	[39:32]
Byte	000	110	-	-	[47:40]	-
Byte	000	101	-	[55:48]	-	-
Byte	000	100	[63:56]	-	-	-
Byte	000	011	-	-	-	[7:0]
Byte	000	010	-	-	[15:8]	-
Byte	000	001	-	[23:16]	-	-
Byte	000	000	[31:24]	-	-	-

Table 7-108 Little-endian write, 64-bit external bus

Internal transfer width	**Access: Write, little-endian, 64-bit external bus**		**System data mapping on to external data bus MPMCDATA to HRDATA**							
	HSIZE [2:0]	**HADDR [2:0]**	**[63:56]**	**[55:48]**	**[47:40]**	**[39:32]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word	010	1--	[31:24]	[23:16]	[15:8]	[7:0]	-	-	-	-
Word	010	0--	-	-	-	-	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	[31:24]	[23:16]	-	-	-	-	-	-
Halfword	001	10-	-	-	[15:8]	[7:0]	-	-	-	-
Halfword	001	01-	-	-	-	-	[31:24]	[23:16]	-	-
Halfword	001	00-	-	-	-	-	-	-	[15:8]	[7:0]
Byte	000	111	[31:24]	-	-	-	-	-	-	-
Byte	000	110	-	[23:16]	-	-	-	-	-	-
Byte	000	101	-	-	[15:8]	-	-	-	-	-
Byte	000	100	-	-	-	[7:0]	-	-	-	-
Byte	000	011	-	-	-	-	[31:24]		-	-
Byte	000	010	-	-	-	-	-	[23:16]	-	-
Byte	000	001	-	-	-	-	-	-	[15:8]	-
Byte	000	000	-	-	-	-	-	-	-	[7:0]

Table 7-109 Big-endian write, 64-bit external bus

Internal transfer width	**Access: Write, big-endian, 64-bit external bus**		**System data mapping on to external data bus MPMCDATA to HRDATA**							
	HSIZE [2:0]	**HADDR [2:0]**	**[63:56]**	**[55:48]**	**[47:40]**	**[39:32]**	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
Word	010	1--	[31:24]	[23:16]	[15:8]	[7:0]	-	-	-	-
Word	010	0--	-	-	-	-	[31:24]	[23:16]	[15:8]	[7:0]
Halfword	001	11-	-	-	[15:8]	[7:0]	-	-	-	-
Halfword	001	10-	[31:24]	[23:16]	-	-	-	-	-	-
Halfword	001	01-	-	-	-	-	-	-	[15:8]	[7:0]
Halfword	001	00-	-	-	-	-	[31:24]	[23:16]	-	-
Byte	000	111	-	-	-	[7:0]	-	-	-	-
Byte	000	110	-	-	[15:8]	-	-	-	-	-
Byte	000	101	-	[23:16]	-	-	-	-	-	-
Byte	000	100	[31:24]	-	-	-	-	-	-	-
Byte	000	011	-	-	-	-	-	-	-	[7:0]
Byte	000	010	-	-	-	-	-	-	[15:8]	-
Byte	000	001	-	-	-	-	-	[23:16]	-	-
Byte	000	000	-	-	-	-	[31:24]	-	-	-

Chapter 8
Test Interface Controller

This chapter describes the *Test Interface Controller* (TIC). It contains the following sections:

- *About the TIC* on page 8-2
- *Sequence of events leading to entry into TIC test mode* on page 8-3.

8.1 About the TIC

The TIC is used to convert externally applied test vectors into internal transfers on the TIC AHB bus. A three-wire external handshake protocol is used, with two inputs, **MPMCTESTREQA** and **MPMCTESTREQB**, controlling the type of vector that is applied and a single output, **MPMCTESTACK**, that indicates when the next vector can be applied.The **MPMCTESTIN** input is used to indicate that TIC test mode must be entered. Typically, the TIC is the highest priority AMBA bus master, ensuring test access under all conditions. TIC testing requires a 32-bit data bus. Therefore, if TIC test support is required for systems with a 16-bit **MPMCDATA** data bus, the upper 16 bits of the data bus, **MPMCDATA[31:16]**, must be multiplexed onto other pads during test mode.

——— **Note** ———

- For more information on TIC testing see the *AMBA Design Kit Technical Reference Manual*
- The TIC is used for production testing of some hard macro ARM processors.
- **MPMCTESTIN** is used by the MPMC to enter test mode. This signal can be used by the system to switch **HCLK** to test speed.
- The TIC must not be the default master because the TIC only operates at the test speed. The TIC might not function correctly if granted the bus at normal operating frequency.

8.2 Sequence of events leading to entry into TIC test mode

The following description assumes that, within a typical system, the processor is always requesting access to the external bus, but the TIC has the highest priority to bus access. A typical sequence of events in using the TIC to apply test patterns is:

1. Apply reset asynchronously.
2. Assert the **MPMCTESTIN** test input to enter test mode.
3. When the reset is removed asynchronously, the PrimeCell MPMC enters TIC test mode.
4. The TIC block asserts its **HBUSREQTIC** signal to the arbiter, requesting access to the AHB bus. Because the TIC is the highest priority, **HGRANTTIC** is asserted and the granted TIC takes ownership of the AHB bus.

——— **Note** ———

The **MPMCTESTIN** signal is usually fed to the clock generation logic so that **HCLK** can be switched to test speed.

Chapter 9
System Connectivity

This chapter describes the system connectivity aspects relating to the MPMC. It contains the following sections:

9.1 On-chip signals

The on-chip signals are described in the following sections:

- *AHB register interface connectivity*
- *AHB memory interface connectivity* on page 9-4
- *TIC test connectivity* on page 9-4
- *Pad interface connectivity* on page 9-4
- *Reset controller connectivity* on page 9-12
- *Boot methodologies* on page 9-14
- *EBI connectivity* on page 9-18
- *Static memory configuration* on page 9-22
- *Dynamic memory configuration* on page 9-24
- *Methodologies in setting configuration tie-offs* on page 9-26.

9.1.1 AHB register interface connectivity

The AHB register interface must be connected to the AHB bus with the microprocessor so that the microprocessor can program the MPMC.

Figure 9-1 on page 9-4 shows the MPMC system interconnection diagram.

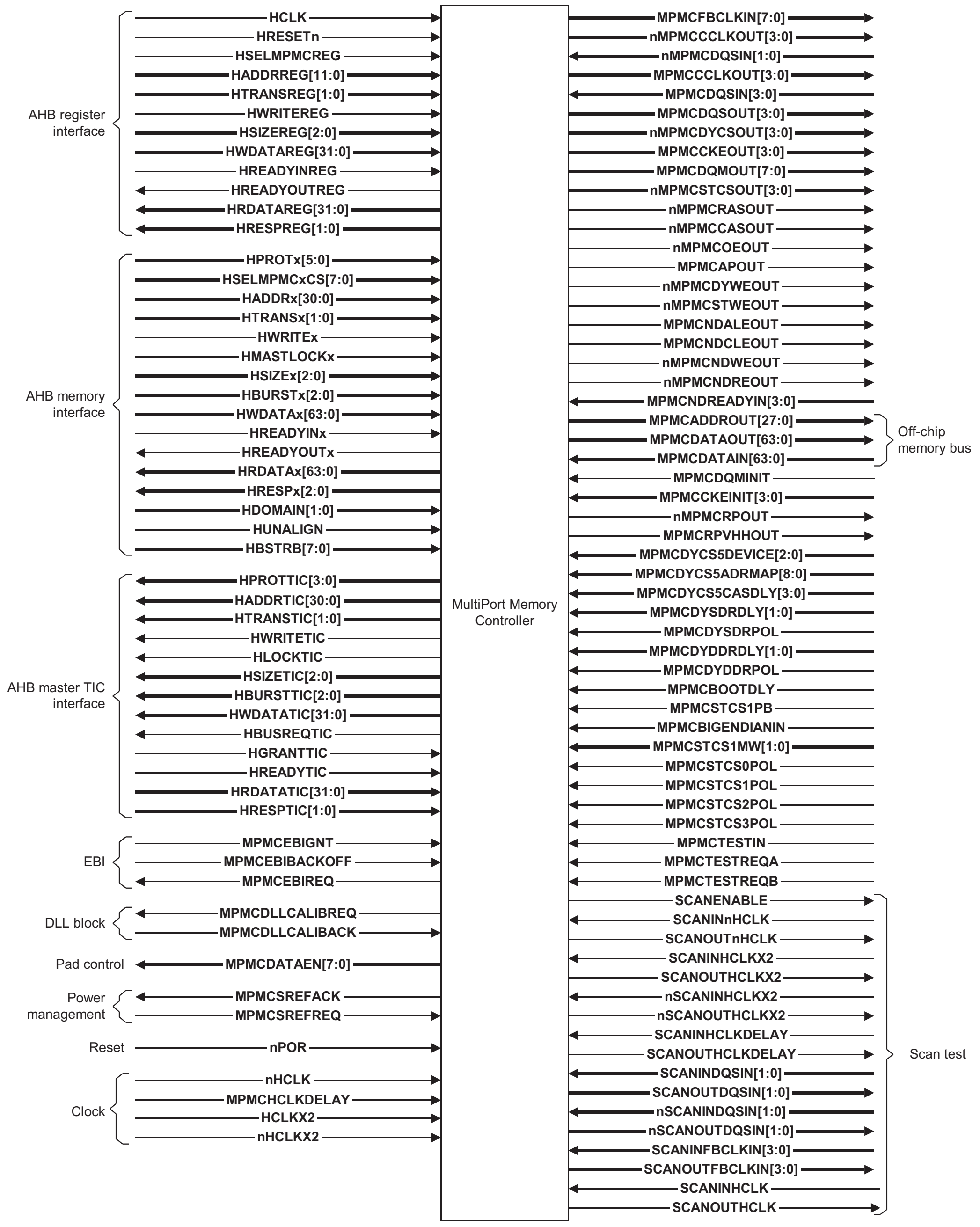

AHB register interface
HCLK
HRESETn
HSELMPMCREG
HADDRREG[11:0]
HTRANSREG[1:0]
HWRITEREG
HSIZEREG[2:0]
HWDATAREG[31:0]
HREADYINREG
HREADYOUTREG
HRDATAREG[31:0]
HRESPREG[1:0]
AHB memory interface
HPROTx[5:0]
HSELMPMCxCS[7:0]
HADDRx[30:0]
HTRANSx[1:0]
HWRITEx
HMASTLOCKx
HSIZEx[2:0]
HBURSTx[2:0]
HWDATAx[63:0]
HREADYINx
HREADYOUTx
HRDATAx[63:0]
HRESPx[2:0]
HDOMAIN[1:0]
HUNALIGN
HBSTRB[7:0]
AHB master TIC interface
HPROTTIC[3:0]
HADDRTIC[30:0]
HTRANSTIC[1:0]
HWRITETIC
HLOCKTIC
HSIZETIC[2:0]
HBURSTTIC[2:0]
HWDATATIC[31:0]
HBUSREQTIC
HGRANTTIC
HREADYTIC
HRDATATIC[31:0]
HRESPTIC[1:0]
EBI
MPMCEBIGNT
MPMCEBIBACKOFF
MPMCEBIREQ
DLL block
MPMCDLLCALIBREQ
MPMCDLLCALIBACK
Pad control
MPMCDATAEN[7:0]
Power management
MPMCSREFACK
MPMCSREFREQ
Reset
nPOR
Clock
nHCLK
MPMCHCLKDELAY
HCLKX2
nHCLKX2
MultiPort Memory Controller
MPMCFBCLKIN[7:0]
nMPMCCCLKOUT[3:0]
nMPMCDQSIN[1:0]
MPMCCCLKOUT[3:0]
MPMCDQSIN[3:0]
MPMCDQSOUT[3:0]
nMPMCDYCSOUT[3:0]
MPMCCKEOUT[3:0]
MPMCDQMOUT[7:0]
nMPMCSTCSOUT[3:0]
nMPMCRASOUT
nMPMCCASOUT
nMPMCOEOUT
MPMCAPOUT
nMPMCDYWEOUT
nMPMCSTWEOUT
MPMCNDALEOUT
MPMCNDCLEOUT
nMPMCNDWEOUT
nMPMCNDREOUT
MPMCNDREADYIN[3:0]
MPMCADDROUT[27:0]
MPMCDATAOUT[63:0]
MPMCDATAIN[63:0]
Off-chip memory bus
MPMCDQMINIT
MPMCCKEINIT[3:0]
nMPMCRPOUT
MPMCRPVHHOUT
MPMCDYCS5DEVICE[2:0]
MPMCDYCS5ADRMAP[8:0]
MPMCDYCS5CASDLY[3:0]
MPMCDYSDRDLY[1:0]
MPMCDYSDRPOL
MPMCDYDDRDLY[1:0]
MPMCDYDDRPOL
MPMCBOOTDLY
MPMCSTCS1PB
MPMCBIGENDIANIN
MPMCSTCS1MW[1:0]
MPMCSTCS0POL
MPMCSTCS1POL
MPMCSTCS2POL
MPMCSTCS3POL
MPMCTESTIN
MPMCTESTREQA
MPMCTESTREQB
SCANENABLE
SCANINnHCLK
SCANOUTnHCLK
SCANINHCLKX2
SCANOUTHCLKX2
nSCANINHCLKX2
nSCANOUTHCLKX2
SCANINHCLKDELAY
SCANOUTHCLKDELAY
SCANINDQSIN[1:0]
SCANOUTDQSIN[1:0]
nSCANINDQSIN[1:0]
nSCANOUTDQSIN[1:0]
SCANINFBCLKIN[3:0]
SCANOUTFBCLKIN[3:0]
SCANINHCLK
SCANOUTHCLK
Scan test

Figure 9-1 System interconnection diagram

9.1.2 AHB memory interface connectivity

AHB memory interface 0 is the highest priority interface and is normally connected to the AHB bus with high-bandwidth, low-latency AHB masters. AHB memory interface 1 is the next highest priority interface and is normally connected to lower-bandwidth, medium-latency peripherals. Interfaces having greater numbers are lower priority, with AHB memory interface 9 having the lowest priority. For example, AHB memory interface 0 can be connected to a video controller, AHB 1 to a high-performance DMA controller, AHB 2 to a microprocessor, and AHB 3 to AHB 9 to lower performance masters.

Note

Any unused AHB interfaces must have their AHB input signals tied LOW.

9.1.3 TIC test connectivity

The TIC AHB master interface, if required, must be connected to the same bus as the microprocessor. The TIC must be the highest priority master on this bus. The **MPMCTESTIN** signal must be asserted during reset to enter TIC test mode. The **MPMCTESTIN** signal must also be sent to the clock generation logic so that the appropriate test clock is selected.

Note

- The TIC must not be the default master. Because the TIC master only operates at test speed, it might not function correctly if granted the bus at functional speed.
- The TIC must be granted the bus in test mode. This is normally performed by making the TIC the highest priority master on the AHB bus.

9.1.4 Pad interface connectivity

The following sections describe the pad interface connectivity for:

- *Static memory* on page 9-5
- *TIC* on page 9-5
- *SDR-SDRAM dynamic memory* on page 9-5
- *DDR-SDRAM memory* on page 9-5.

Static memory

The static memory output signals, including NAND flash output signals, are generated using **HCLK**, and then combined with the dynamic memory and TIC output signals. See Figure 9-3 on page 9-8 and Figure 9-6 on page 9-11.

The static memory input signals are registered by **HCLK**, see Figure 9-2 on page 9-7.

TIC

The TIC output signals are generated using **HCLK**, and then combined with the static memory and dynamic controller output signals, see Figure 9-3 on page 9-8.

The TIC input signals are registered by **HCLK**, see Figure 9-2 on page 9-7.

SDR-SDRAM dynamic memory

The SDR-SDRAM memory output signals are generated either using **HCLK** in the clock delayed pad interface methodology, or by **MPMCHCLKDELAY** clock using the command delayed pad interface methodology, see *Pad interface methodology* on page 10-22. The output signals are then combined with the static memory and TIC output signals. See Figure 9-3 on page 9-8, Figure 9-4 on page 9-9 and Figure 9-6 on page 9-11.

The SDR-SDRAM memory input signals are first registered by the feedback clocks, **MPMCFBCLKIN[7:0]**. Depending on the edge of **HCLK** chosen to then register the read data, the read data is either:

- captured straight onto the positive edge of **HCLK**
- first captured on the negative of **HCLK** using **nHCLK** and then half a cycle later captured on the positive edge of **HCLK**.

See Figure 9-2 on page 9-7.

DDR-SDRAM memory

The majority of the DDR-SDRAM memory output signals are generated either using **HCLK** in the clock delayed pad interface methodology, or by **MPMCHCLKDELAY** using the command delayed pad interface methodology. See *Pad interface methodology* on page 10-22. The output signals are then combined with the static memory and TIC output signals.

The write data, **MPMCDATAOUT[31:0]**, and data mask, **MPMCDQMOUT[7:0]** signals are generated using the **nHCLKX2** clock.

The output data strobe, **MPMCDQSOUT[3:0]** is generated using the **HCLKX2** clock.

See Figure 9-3 on page 9-8 to Figure 9-6 on page 9-11.

The DDR-SDRAM memory input read data is provided at double the clock rate. The read data is therefore required to be captured on both edges of the data strobe. To resolve issues with instantiating negative edge d-types in the MPMC, in addition to the normal delayed data strobe signal, **MPMCDQSIN[3:0**], the MPMC requires an inverted data strobe signal **nMPMCDQSIN[3:0]**, to capture the read data on the negative edge of the data strobe.

The first item of data is captured using the positive edge of the **MPMCDQSIN[3:0]** data strobe. The second item of data is captured using the positive edge of the **nMPMCDQSIN[3:0]** data strobe, half a cycle later. The first item of data captured on **MPMCDQSIN[3:0]** is registered on **nMPMCDQSIN[3:0]** after being captured in the **MPMCDQSIN[3:0]** domain. This means that both the first and second data items are in the **nMPMCDQSIN[3:0]** clock domain, making it easier to retime the read data back to the **HCLK** clock domain.

Note

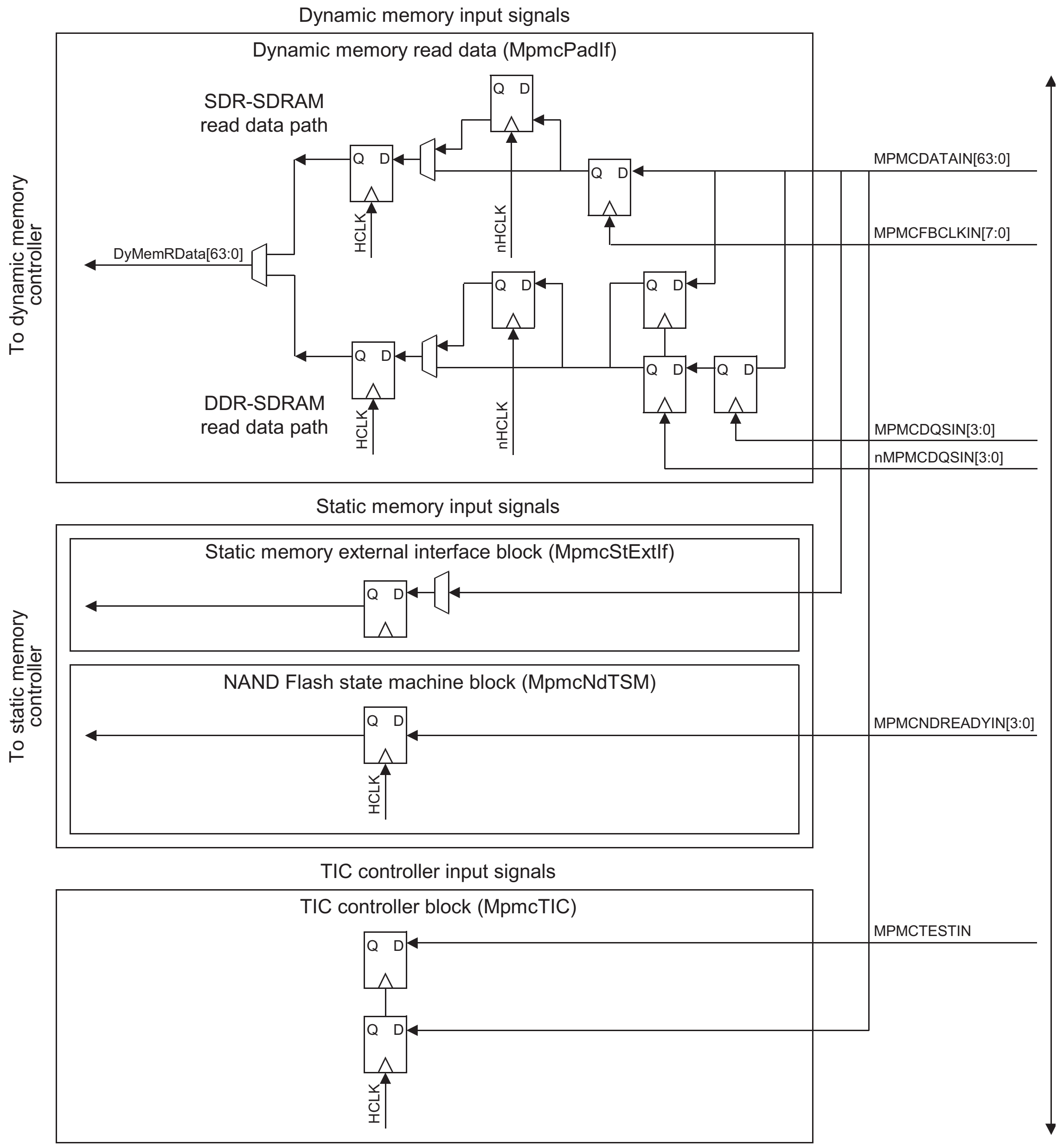
32-bit DDR-SDRAM devices have a single DQS connection, rather than one for each byte lane. This is because they are expected to be used in systems where the difference in delays between byte lanes is small. The single DQS on the memory device must be connected to all four DQS ports of the MPMC.

It is not recommended that 32-bit DDR-SDRAM devices are mixed with 8-bit or 16-bit DDR-SDRAM devices, because of the differences in DQS signals between the devices.

Depending on the edge of **HCLK** chosen to register the read data, the read data is either:

- captured straight onto the positive edge of **HCLK**
- first captured on the negative of **HCLK** using **nHCLK** and then half a cycle later captured on the positive edge of **HCLK**.

See Figure 9-2 on page 9-7.



Dynamic memory input signals
Dynamic memory read data (MpmcPadIf)
SDR-SDRAM read data path
DDR-SDRAM read data path
To dynamic memory controller
DyMemRData[63:0]
HCLK
nHCLK
Q D
MPMCDATAIN[63:0]
MPMCFBCLKIN[7:0]
MPMCDQSIN[3:0]
nMPMCDQSIN[3:0]
Static memory input signals
Static memory external interface block (MpmcStExtIf)
NAND Flash state machine block (MpmcNdTSM)
To static memory controller
MPMCNDREADYIN[3:0]
TIC controller input signals
TIC controller block (MpmcTIC)
MPMCTESTIN

Figure 9-2 Pad interface input signals

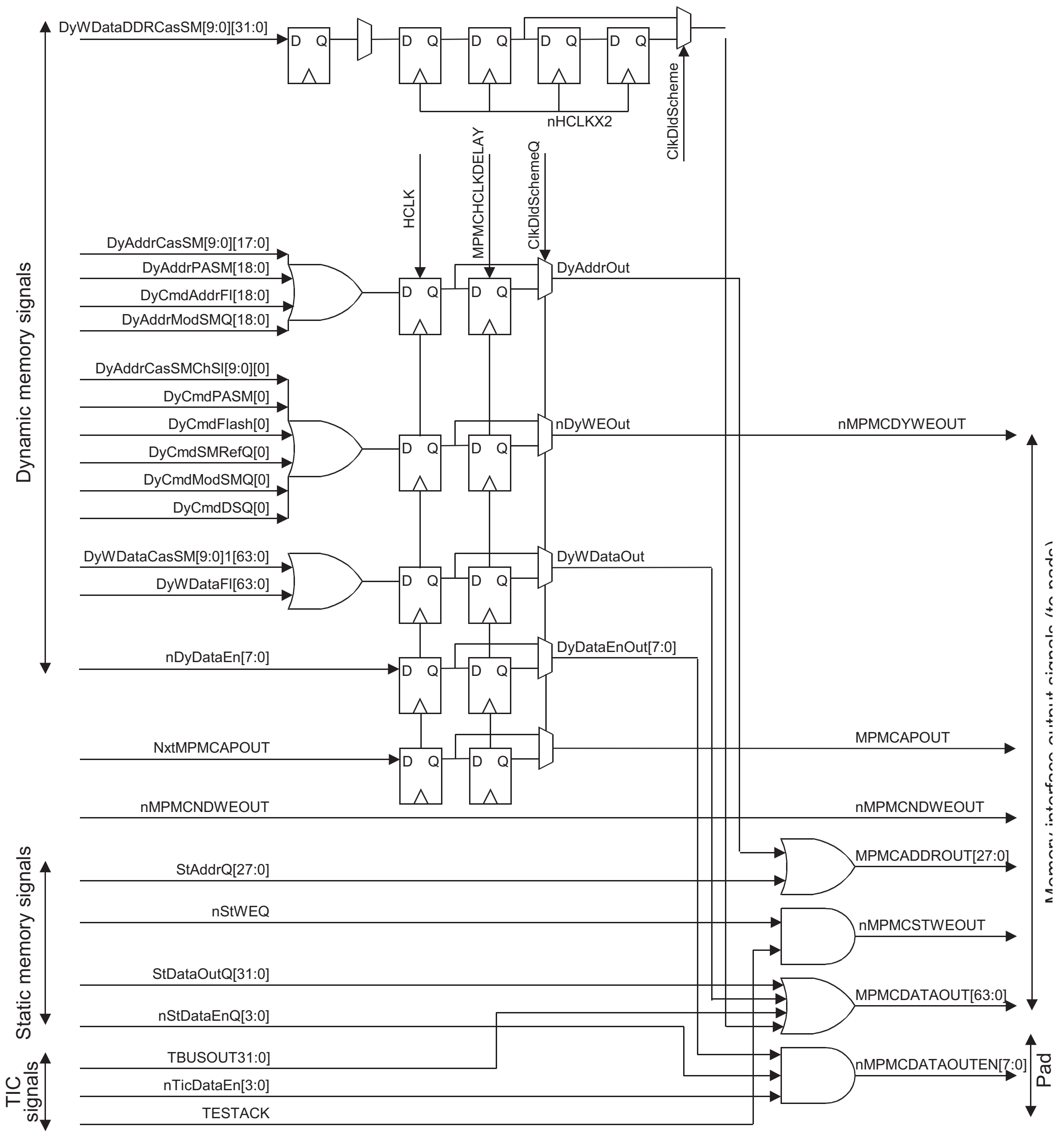


Figure 9-3 Pad interface output signals

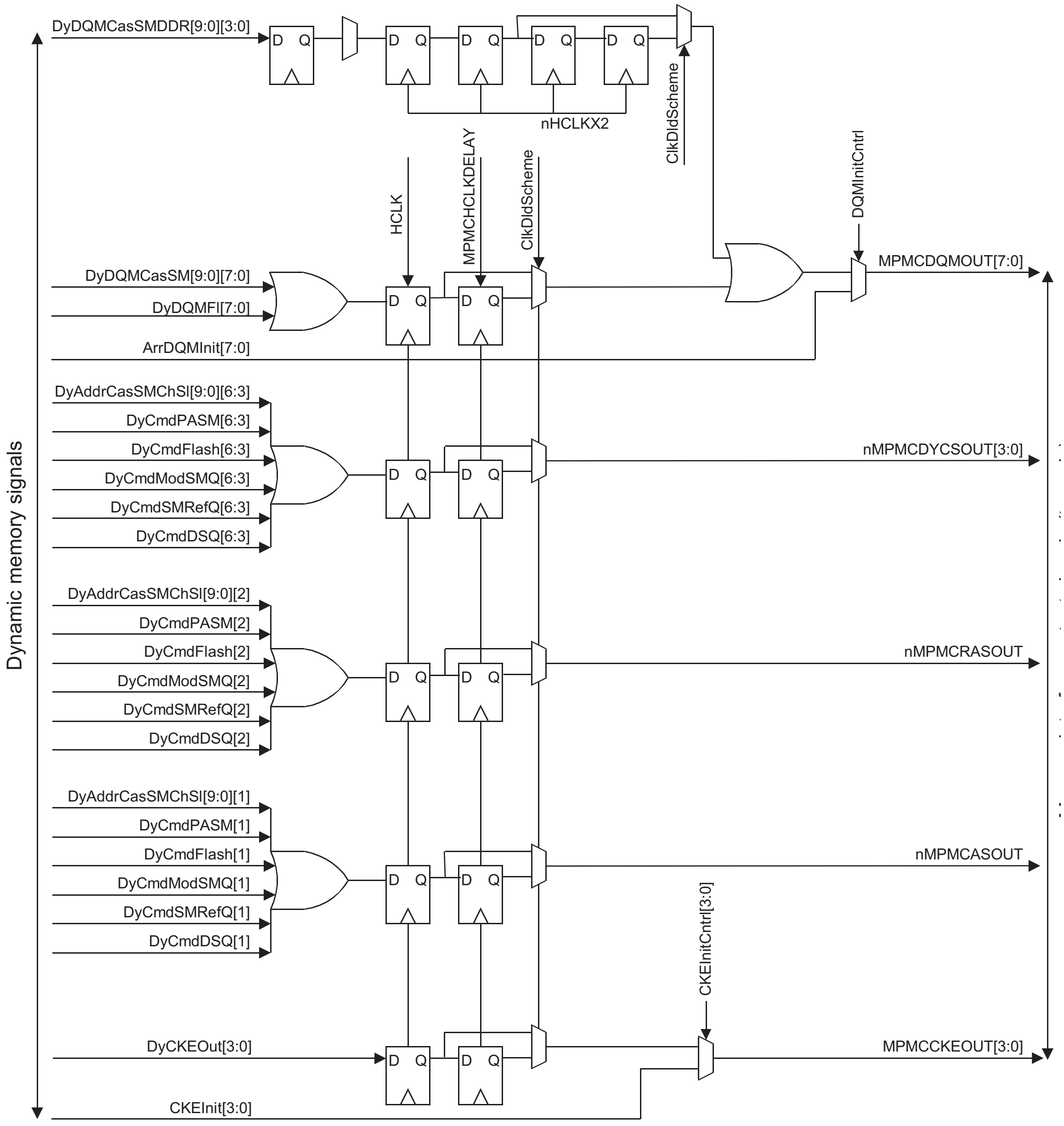


Figure 9-4 Pad interface dynamic memory output signals

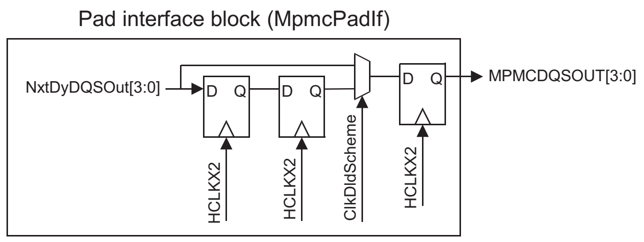


Figure 9-5 Pad interface dynamic memory output signals

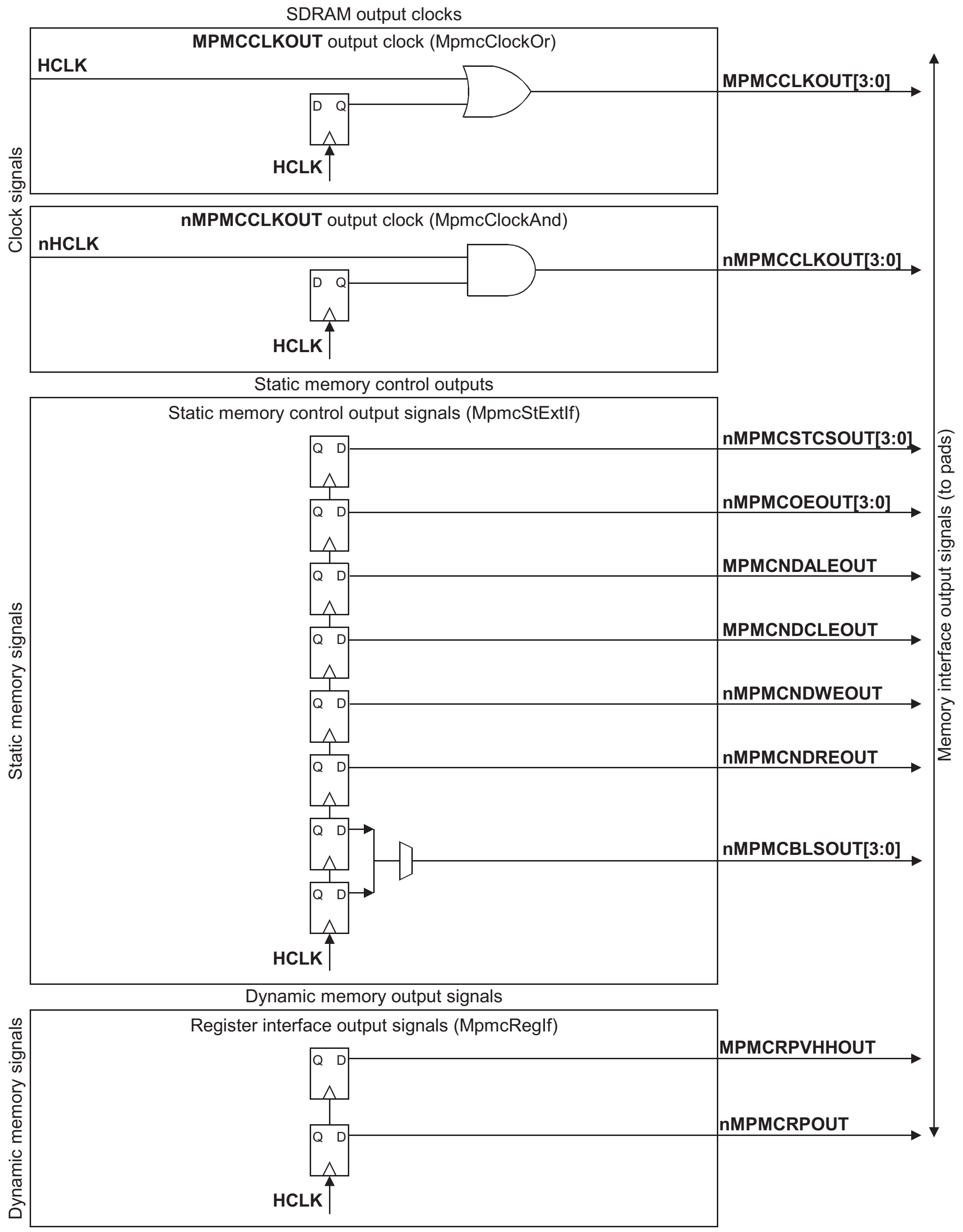

SDRAM output clocks
MPMCCLKOUT output clock (MpmcClockOr)
HCLK
D Q
HCLK
MPMCCLKOUT[3:0]
Clock signals
nMPMCCLKOUT output clock (MpmcClockAnd)
nHCLK
D Q
HCLK
nMPMCCLKOUT[3:0]
Static memory control outputs
Static memory control output signals (MpmcStExtIf)
Q D
nMPMCSTCSOUT[3:0]
nMPMCOEOUT[3:0]
MPMCNDALEOUT
MPMCNDCLEOUT
nMPMCNDWEOUT
nMPMCNDREOUT
nMPMCBLSOUT[3:0]
HCLK
Static memory signals
Memory interface output signals (to pads)
Dynamic memory output signals
Register interface output signals (MpmcRegIf)
MPMCRPVHHOUT
nMPMCRPOUT
HCLK
Dynamic memory signals

Figure 9-6 Pad interface output signals

9.1.5 Reset controller connectivity

There are two reset signals:

- power-on reset, **nPOR**
- soft reset, **HRESETn**.

Both **nPOR** and **HRESETn** are active LOW reset signals. Both signals can be asynchronously asserted, but must be synchronously deasserted with respect to **HCLK**:

- on power-on reset, both **nPOR** and **HRESETn** must go active
- on soft reset, **HRESETn** must go active.

Having two reset signals enables the system to be booted using soft reset if the system is in a low-power state, and the SDRAM is in self-refresh mode.

Note

ARM PrimeCell designs use asynchronous reset. The minimum amount of time the reset signals must be enabled depends on the cell library.

nPOR reset signal

nPOR resets the majority of the logic in the MPMC. If power-on reset is performed, the MPMC registers must be initialized.

HRESETn reset signal

HRESETn sets the Enable (E) field of the MPMCControl Register, ensuring that the MPMC is enabled and can be booted from. Any outstanding memory requests and memory transactions in progress are lost when **HRESETn** is asserted.

HRESETn does not reset any of the other programmer visible registers. The MPMC does not have to be re-initialized over soft reset.

For more information on the effects of the reset signals, see Table 9-1 and Figure 9-7 on page 9-14.

Table 9-1 Reset signal cycles

Cycle	Description
T0-T1	Power-on reset is asserted asynchronously. Power-on reset requires both **nPOR** and **HRESETn** to be asserted.
T1-T2	Power-on reset active.
T2-T3	Power-on reset active.
T3-T4	Power-on reset is deasserted synchronously. Power-on reset exit requires both **nPOR** and **HRESETn** to be deasserted.
T4-T5	Normal operation.
T5-T6	Soft reset, **HRESETn**, is asserted asynchronously.
T6-T7	Soft reset active.
T7-T8	Soft reset active.
T8-T9	Soft reset is deasserted synchronously.
T9-T10	Normal operation.
T10-T11	Soft reset, **HRESETn**, is asserted asynchronously.
T11-T12	Soft reset active.
T12-T13	Soft reset active.
T13-T14	Soft reset is deasserted synchronously.
T14-T15	Normal operation.
T15-T16	Power-on reset is asserted asynchronously. Power-on reset requires both **nPOR** and **HRESETn** to be asserted.
T16-T17	Power-on reset active.
T17-T18	Power-on reset is deasserted synchronously. Power-on reset exit requires both **nPOR** and **HRESETn** to be deasserted.

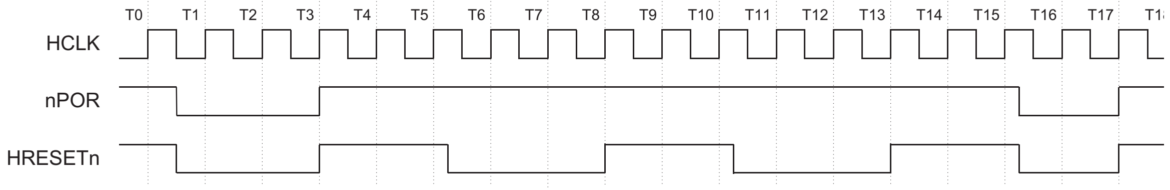


Figure 9-7 Reset signal timing

Systems with hard reset only

Systems with hard reset only must assert both **nPOR** and **HRESETn** on reset. All the register values in the MPMC are then lost on reset. The MPMC must be re-initialized on exiting reset. See *Boot methodologies* for more information on restarting the system using power-on reset during self refresh.

9.1.6 Boot methodologies

The following boot methodologies are available:

- *System first powered up* on page 9-15
- *Soft reset performed on self-refresh exit* on page 9-15
- *Hard reset performed on self-refresh exit* on page 9-15
- *Restarting the system when it has locked up using soft reset, possible data loss* on page 9-16
- *Restarting the system when it has locked up using hard reset, possible data loss* on page 9-17
- *Restarting the system when it has locked up using soft reset, retaining SDRAM data* on page 9-17
- *Restarting the system when it has locked up using hard reset, retaining SDRAM data* on page 9-18.

——— **Note** ———

The **MPMCCKEINIT[3:0]** and **MPMCDQMINIT** inputs control the initial values of the **MPMCCKEOUT[3:0]** and **MPMCDQMOUT[7:0]** outputs respectively after an **nPOR** reset. These are used to ensure that the memory control signals are driven to the correct values for the memory devices in use, depending on whether the device is uninitialized or in self-refresh mode.

System first powered up

The following procedure can be used when the system is first powered up:

1. Both **nPOR** and **HRESETn** must go active.
2. Both **nPOR** and **HRESETn** must go inactive synchronously to the clock.
3. The MPMC must be initialized.
4. The memory devices must be initialized.

Soft reset performed on self-refresh exit

The following procedure can be used when a soft reset is performed on a self-refresh exit:

1. SDRAM self-refresh mode entered.
2. Power management unit disables clocks to save power.
3. Power management unit restarts system by performing a soft reset, asserting and then deasserting **HRESETn**.
4. On deassertion of **HRESETn** the MPMC can be used.

   ——— **Note** ———

   The MPMC and memory devices do not have to be initialized.

   ————————

5. The CPU reads a system status bit to determine the mode of the system.
6. Self-refresh mode is exited by:
   - using the self-refresh hardware signals
   - using the software register fields.

Hard reset performed on self-refresh exit

The following procedure can be used when a hard reset is performed on a self-refresh exit:

1. SDRAM self-refresh mode entered.
2. Power management unit disables clocks to save power.
3. Power management unit powers of part of the ASIC, including the MPMC, to save additional power. The MPMC register contents are lost.

4. Power management unit restarts the system by re-enabling power.

5. The power management unit performs a hard reset, asserting both **nPOR** and **HRESETn**, and then deasserting both reset signals.

   ——— **Note** ———

   The MPMC must be initialized before accessing dynamic memory. The memory devices do not have to be initialized.

   ————————

6. The CPU reads a system status bit to determine the mode of the system.

7. The CPU initializes the MPMC.

8. Self-refresh mode is exited, by using the software register fields.

Restarting the system when it has locked up using soft reset, possible data loss

The following soft reset procedure can be used to restart the system when it has locked up:

1. Power management unit performs a soft reset by asserting **HRESETn**.

   ——— **Note** ———

   During reset the MPMC does not perform auto-refresh commands and the memory contents of SDRAM might be corrupted.

   ————————

2. Power management unit exits soft reset by deasserting **HRESETn**.

3. On deassertion of **HRESETn**, the MPMC can be used.

   ——— **Note** ———

   The MPMC and memory devices do not have to be initialized.

   ————————

4. The CPU reads a system status bit to determine the mode of the system.

   ——— **Note** ———

   If the time **HRESETn** is active is short enough, the contents of the SDRAM memory might not be corrupted.

   ————————

Restarting the system when it has locked up using hard reset, possible data loss

The following hard reset procedure can be used to restart the system when it has locked up:

1. Power management unit performs a hard reset, by asserting both **nPOR** and **HRESETn**.

   ——— **Note** ———

   During reset the MPMC does not perform auto-refresh commands and the memory contents of SDRAM might be corrupted.

   ————————

2. The power management unit exits hard reset, by deasserting **nPOR** and **HRESETn**.

3. On deassertion of **HRESETn**, the MPMC can be used.

   ——— **Note** ———

   The MPMC must be initialized before accessing dynamic memory. Specifically the auto-refresh counter must be re-initialized as early as possible to avoid SDRAM data loss. The memory devices do not have to be initialized.

   ————————

4. The CPU reads a system status bit to determine the mode of the system.

5. The CPU initializes the MPMC.

   ——— **Note** ———

   If the time **HRESETn** is active is short enough, the contents of the SDRAM memory might not be corrupted.

   ————————

Restarting the system when it has locked up using soft reset, retaining SDRAM data

The following soft reset procedure can be used to restart the system when it has locked up:

1. Power management unit forces the MPMC into self-refresh mode.
2. Power management unit performs a soft reset, by asserting **HRESETn**.
3. Power management unit exits soft reset, by deasserting **HRESETn**.
4. On deassertion of **HRESETn** the MPMC can be used.

Note

The MPMC and memory devices do not have to be initialized.

5. The CPU reads a system status bit to determine the mode of the system.
6. Self-refresh mode is exited by one of the following:
   - using the self-refresh hardware signals
   - using the software register fields.

Restarting the system when it has locked up using hard reset, retaining SDRAM data

The following hard reset procedure can be used to restart the system when it has locked up:

1. Power management unit forces the MPMC into self-refresh mode.
2. Power management unit performs a hard reset, by asserting both **nPOR** and **HRESETn**.
3. The power management unit exists hard reset, by deasserting both **nPOR** and **HRESETn** reset signals.

   Note

   The MPMC must be initialized before accessing dynamic memory. The memory devices do not have to be initialized.

4. The CPU reads a system status bit to determine the mode of the system.
5. The CPU initializes the MPMC.
6. Self-refresh mode is exited by using the software register fields.

9.1.7 EBI connectivity

The memory interface signals can be shared with other peripherals by using an *External Bus Interface* (EBI) block. The EBI block enables several peripherals to share the same pads. The EBI contains arbitration logic to decide which peripheral drives the external bus.

For more information on the EBI, see *ARM PrimeCell External Bus Interface Technical Reference Manual (PL220)*.

Note

The MPMC does not perform any commands or memory transactions unless it has been granted the bus by the EBI.

EBI connectivity is described in the following sections:

- *Pin sharing*
- *EBI signals*
- *EBI connectivity options*
- *Example EBI interface transaction* on page 9-21.

Pin sharing

The EBI enables the address and data signals to be shared. The MPMC control signals are not shared and must be directly connected to the pads.

EBI signals

The MPMC provides the following signals to interface to the EBI:

MPMCEBIREQ This signal is an output to the EBI and is used to request the external bus.

MPMCEBIGNT This signal is an input from the EBI and indicates that the EBI has granted the MPMC the bus.

MPMCEBIBACKOFF This signal is an input from the EBI and indicates that the MPMC must complete its current transaction and no longer request the bus, so that another peripheral can be granted the bus.

EBI connectivity options

There are two options available for EBI connectivity:

EBI required If the use of the EBI is required, the following signals must be connected to the EBI (see Figure 9-8 on page 9-20):

- MPMC EBI interface signals:
  - **MPMCEBIREQ**
  - **MPMCEBIGNT**
  - **MPMCBACKOFF**.
- address signal:
  - **MPMCADDROUT[27:0]**.

- data signals:
  - **MPMCDATAOUT[63:0]**
  - **MPMCDATAIN[63:0]**.

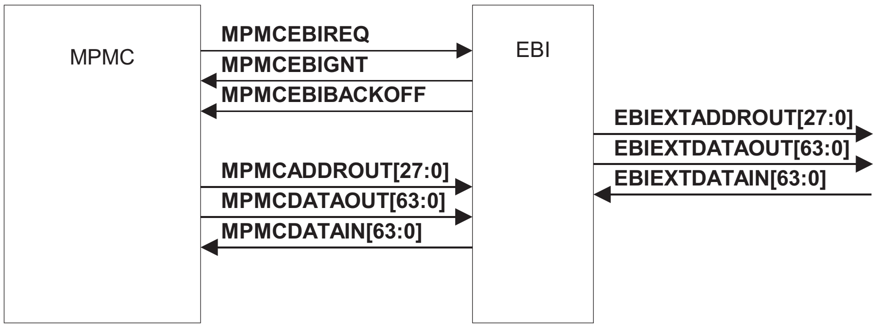


Figure 9-8 EBI connectivity, EBI required

EBI not required If the EBI is not required, the **MPMCEBIGNT** signal must be tied HIGH, indicating that the bus is always granted. The **MPMCEBIBACKOFF** signal must be tied LOW, indicating that the MPMC must not back off from the bus. The **MPMCEBIREQ** signal can be left unconnected (see Figure 9-9).

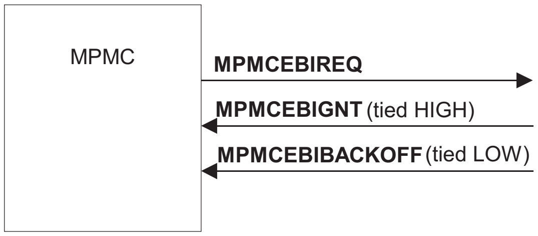


Figure 9-9 EBI connectivity, EBI not required

Example EBI interface transaction

Table 9-2 and Figure 9-10 on page 9-22 show example EBI signal cycles.

Table 9-2 EBI signal cycles

Cycle	Description
T0-T1	The MPMC requests the external bus, by asserting the **MPMCEBIREQ** signal.
T1-T2	MPMC idle waiting to be granted the bus.
T2-T3	MPMC idle waiting to be granted the bus.
T3-T4	The EBI grants the MPMC the external bus.
T4-T5	The MPMC performs transactions.
T5-T6	The MPMC performs transactions.
T6-T7	The MPMC performs transactions.
T7-T8	The MPMC no longer requires the bus, and deasserts the **MPMCEBIREQ** signal.
T8-T9	The EBI degrants the memory bus by deasserting the **MPMCEBIGNT** signal.
T9-T10	MPMC idle.
T10-T11	The MPMC requests the external bus by asserting the **MPMCEBIREQ** signal.
T11-T12	The EBI grants the MPMC the external bus.
T12-T13	The MPMC performs transactions.
T13-T14	The MPMC performs transactions.
T14-T15	The MPMC performs transactions. The EBI wants to re-arbitrate the memory bus to enable another peripheral to perform transactions, and raises the **MPMCBACKOFF** signal.
T15-T16	The memory deasserts the **MPMCEBIREQ** signal as soon as it is able to.
T16-T17	The EBI degrants the memory bus by deasserting the **MPMCEBIGNT** signal.

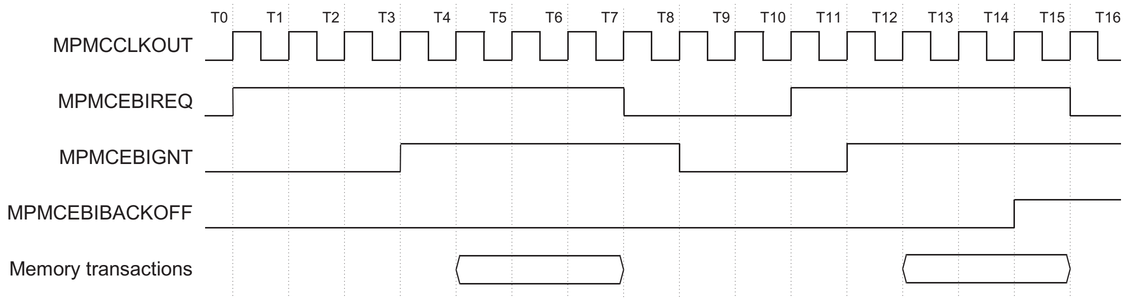


Figure 9-10 EBI signal timing

9.1.8 Static memory configuration

The static memory chip selects can be configured on power-on reset by using tie-off signals.

Booting from static memory

The MPMC can be configured to boot from static memory.

Static memory chip select 1 contains additional functionality enabling it to be booted from. This chip select can be configured on power-on reset by setting a number of tie-offs. The value of these tie offs must not be changed during normal operation. The tie-offs enable the following to be configured:

- polarity of the chip select
- polarity of the byte lane strobe
- memory width.

The power-on reset value of these signals can be determined by reading the appropriate fields in the MPMCStaticConfig1 Register. The tie-off values can be overridden by writing to the appropriate register field fields. The register then shows the updated value.

The tie-off signals are shown in Table 9-3.

Table 9-3 MPMCStaticConfig1 Register

Configuration input signal	MPMCStaticConfig1 Register field	Configuration option
MPMCSTCS1MW[1:0]	Memory width (MW)	Chip select 1 data bus width:00 = 8-bit01 = 16-bit10 = 32-bit11 = reserved.
MPMCSTCS1PB	Byte lane state (PB)	Chip select 1 byte lane state:0 = **nMPCSBLSOUT[3:0]** signals are HIGH for reads and LOW for writes1 = **nMPMCBLSOUT[3:0]** signals are LOW for reads and LOW for writes.
MPMCSTCS1POL	Chip select polarity (PC)	Chip select 1 chip select polarity:0 = active LOW chip select1 = active HIGH chip select.

Note

If these tie-offs are not required these signals must be tied LOW.

Configuring chip selects 0, 2, and 3 on power-on reset

Chip selects 0, 2, and 3 chip select polarity can be configured by the use of tie-offs to minimize the risk of bus clash. The value of these tie offs must not be changed while the MPMC is in normal operation. The power-on reset value of these signals can be determined by reading the appropriate chip select polarity (PC) field of the MPMCStaticConfig Register. The tie-off values can be overridden by writing to the chip select polarity (PC) field. The register then shows the updated value.

The tie-off signals are shown in Table 9-4.

Table 9-4 MPMCStaticConfig Register

Configuration input signal	MPMCStaticConfig Register field	Configuration option
MPMCSTCS0POL	Chip select polarity (PC)	Chip select 0 chip select polarity:0 = active LOW chip select1 = active HIGH chip select.
MPMCSTCS2POL	Chip select polarity (PC)	Chip select 2 chip select polarity:0 = active LOW chip select1 = active HIGH chip select.
MPMCSTCS3POL	Chip select polarity (PC)	Chip select 3 chip select polarity:0 = active LOW chip select1 = active HIGH chip select.

Note

If these tie-offs are not required these signals must be tied LOW.

9.1.9 Dynamic memory configuration

The dynamic memory chip selects can be configured on power-on reset by using tie-off signals.

Booting from dynamic memory

The MPMC supports booting from Micron SyncFlash dynamic memory. This requires both the pad interface and a chip select to be configured on power-on reset.

The pad interface mode and edge of the clock on which the read data is captured for SDR-SDRAM operation, can be configured on power-on reset by setting a number of tie-offs. The value of these tie offs must not be changed while the MPMC is in normal operation. The power-on reset value of these signals can be determined by reading the SDR-SDRAM read data strategy (SRD) and SDR-SDRAM read data capture polarity (SRP) fields of the MPMCDynamicReadConfig Register. The tie-off values can be overridden by writing to the fields by software. The register then shows the updated value.

The tie-off signals are shown in Table 9-5.

Table 9-5 MPMCDynamicReadConfig Register

Configuration input signal	MPMCDynamic ReadConfig Register field	Configuration option
MPMCDYSDRDLY[1:0]	SDR-SDRAM read data strategy (SRD)	Select the read data strategy:00 = clock delayed mode01 = command + 0 delayed mode10 = command + 1 delayed mode11 = command + 2 delayed mode.
MPMCDYSDRPOL	SDR-SDRAM read data capture polarity (SRP)	Select whether the read data is captured on the positive or negative edge of **HCLK**: 0 = data is captured on the negative edge of **HCLK** 1= data is captured on the positive edge of **HCLK**.

Note

If these tie-offs are not required these signals must be tied LOW.

Dynamic memory chip select 5 contains additional functionality to enable it to be booted from. This functionality enables the chip select to be configured on power-on reset by setting a number of tie-offs. The value of these tie offs must not be changed while the MPMC is in normal operation. The tie-offs enable the following to be configured:

- CAS latency
- memory device
- address mapping
- memory width.

The power-on reset value of these signals can be determined by reading the MPMCDynamicConfig1 and MPMCDynamicRasCas1 Registers. The tie-off values can be overridden by writing to the fields by software. The register then shows the updated value.

The tie-off signals are shown in Table 9-6 and Table 9-7.

Table 9-6 MPMCDynamicConfig1 Register

Configuration input signal	MPMCDynamic Config1 Register field	Configuration option
MPMCDYCS5ADRMAP[8:0]	Address mapping (AM). Bits [15:7]	Select the address mapping, and width of the chip select
MPMCDYCS5DEVICE[2:0]	Memory device (MD)	Chip select 5 memory device

Table 9-7 MPMCDynamicRasCas1 Register

Configuration input signal	MPMCDynamic RasCas1 Register field	Configuration option
MPMCDYCS5CASDLY[3:0]	Cas latency (CAS)	Chip select 5 CAS latency

Note

If these tie-offs are not required these signals must be tied LOW.

9.1.10 Methodologies in setting configuration tie-offs

The power-on reset configuration tie-off signals can be set in several ways.

Hard coding tie-off value

If the exact values of the tie-offs are known by the system designer, or the relevant tie-off is not used the tie-off can be tied HIGH or LOW. Figure 9-11 on page 9-27 shows a hard coding tie-off.

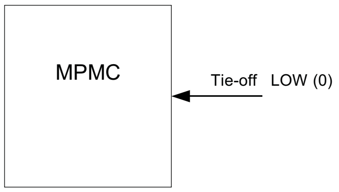


Figure 9-11 Hard coding tie-off

Setting tie-off value by a register

If the tie-off pin value is system-dependent, the tie-off value can be set by a register in another block. Figure 9-12 shows a setting tie-off value by a register.

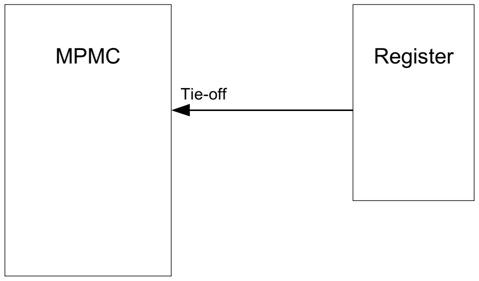


Figure 9-12 Setting tie-off value by a register

Setting tie-off value externally

If the tie-off pin value is system-dependent, the tie-off value can be set externally to the ASIC. However, this does use a pin on the ASIC. Figure 9-13 on page 9-28 shows a setting tie-off value external to the chip.

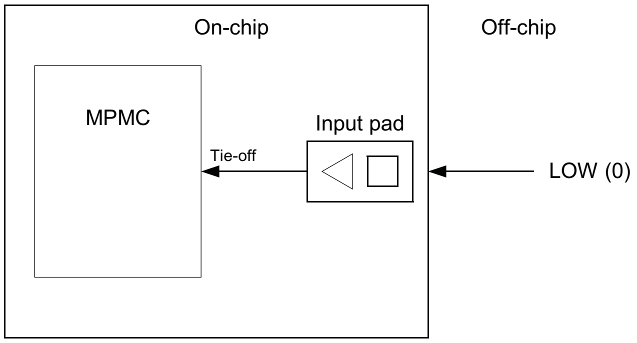


Figure 9-13 Setting tie-off value externally

Setting tie-off value externally using a shared pin

If the tie-off pin value is system-dependent, the tie-off value can be set externally to the ASIC. The ASIC pin can be shared by registering the input value on power-on reset, in for example, the PMU block. Figure 9-14 shows a setting tie-off value external to the chip using a shared pin.

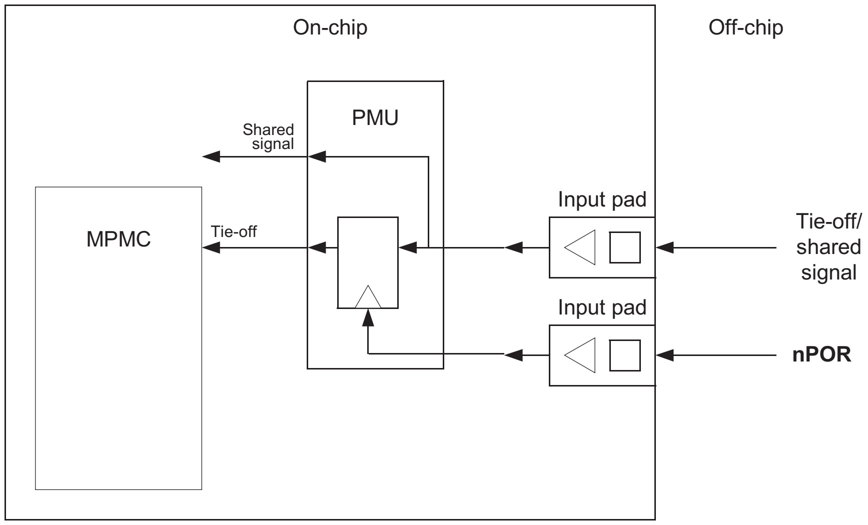


Figure 9-14 Setting tie-off value externally using a shared pin

9.2 Self-refresh entry

The PMU self-refresh signals must be connected to the PrimeCell MPMC **MPMCSREFREQ** and **MPMCSREFACK** signals. When **MPMCSREFREQ** is asserted, the controller closes any open memory banks, and then puts the memory into self-refresh mode. The **MPMCSREFACK** signal is used to indicate to the PMU that the external memories are in self-refresh state. The system must ensure that the memory subsystem is idle before asserting **MPMCSREFREQ**. Deasserting **MPMCSREFREQ** returns the memory to normal operation. See the memory data sheet for refresh requirements. If the PMU does not provide the required signals to automatically enter self-refresh mode then the PrimeCell MPMC **MPMCSREFREQ** signal must be tied LOW. Self-refresh mode can instead be entered manually by programming the SR bit in the MPMCDynamicControl Register.

The section describes:

- *Power Management Unit (PMU) self-refresh entry*
- *Manual self-refresh entry* on page 9-31
- *Forcing multilayer AHB masters idle* on page 9-32.

——— **Note** ———

- After self-refresh exit some devices require several auto-refresh cycles to take place. If this is the case the auto-refresh cycles must be generated by reprogramming the refresh period and waiting for the required number of refresh commands to occur.
- A PMU is not supplied with the MPMC.

9.2.1 Power Management Unit (PMU) self-refresh entry

PMU self-refresh entry uses a PMU to handshake with the MPMC to enter self-refresh mode. These signals are shown in Figure 9-15 on page 9-30.

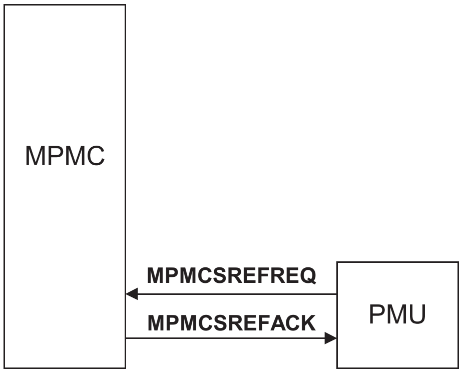


Figure 9-15 PMU self-refresh

When the PMU requires the SDRAM memory to enter self-refresh mode it must first ensure that there are no masters performing transactions to the MPMC. Various methodologies to perform this are described in *Forcing multilayer AHB masters idle* on page 9-32.

When the AHB masters cannot perform transfers to the MPMC the PMU can then assert the self-refresh request signal, **MPMCSREFREQ**. The MPMC then flushes its buffers and puts the SDRAM memory into self-refresh mode. When the memory is placed into self-refresh mode the MPMC asserts the self-refresh acknowledge signal, **MPMCSREFACK**.

The PMU can then gate **HCLK** as required, see *a in Figure 9-16 on page 9-31.

For exit from self-refresh, the PMU re-enables the clocks, and then brings **MPMCSREFREQ** LOW. The SDRAM controller then exits self-refresh mode. When self-refresh is exited the MPMC does not enable any access to the SDRAM until the values in the MPMCDynamictSREX and MPMCDynamictXSR Registers time out. When the SDRAM controller has exited self-refresh it lowers the **MPMCSREFACK** signal. The PMU can then enable the AHB masters to access the AHB bus. This can be performed in several ways:

- by performing soft reset
- by generating an interrupt
- by enabling a peripheral transfer to complete, which is used to stall the ARM processor
- by lowering the signal to the arbiter so that a dummy AHB master is no longer granted the bus.

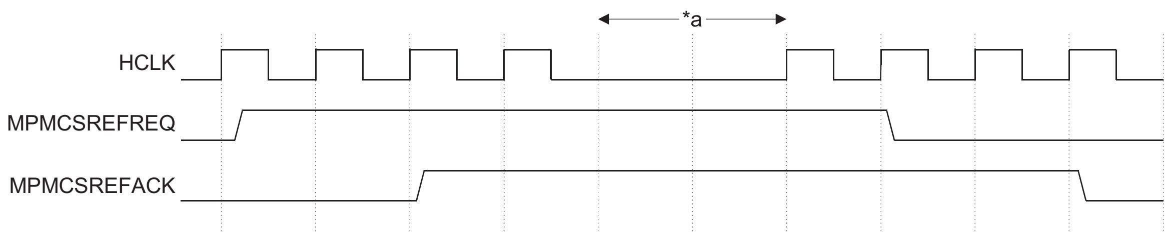


Figure 9-16 PMU self-refresh waveforms

9.2.2 Manual self-refresh entry

Self-refresh can be entered manually in systems where the ARM processor can execute code from internal memory, or from another memory controller.

To enter self-refresh mode the processor must first ensure that there are no masters currently performing transactions to the MPMC. Various methodologies to perform this are described in *Forcing multilayer AHB masters idle* on page 9-32.

While the ARM processor is executing code from internal memory or another memory controller it must then set the SR bit (self-refresh request) in the MPMCDynamicControl Register. The MPMC then flushes its buffers and puts the memory into self-refresh mode. When the memory is placed into self-refresh mode the MPMC asserts the SA bit (self-refresh acknowledge) in the MPMCStatus Register. The ARM processor must wait until this bit goes active, before continuing the power-down procedure. The PMU must gate **HCLK** as necessary.

For exit from self-refresh, the PMU re-enables the clocks and the ARM processor starts executing code. The ARM processor then clears the self-refresh request (SR) bit in the MPMCDynamicControl Register. The MPMC then clears the self-refresh acknowledge bit. All transactions are stalled while the MPMC is exiting self-refresh. The SDRAM controller then proceeds to exit self-refresh mode. This takes about 200 clock cycles. When the SDRAM controller has exited self-refresh it clears the self-refresh acknowledge (SA) bit in the MPMCStatus Register. The ARM processor can then reprogram the AHB masters to enable them to perform transactions to the MPMC.

9.2.3 Forcing multilayer AHB masters idle

Multilayer AHB masters can be forced idle by the following methods:

- The ARM processor can be used to program or interrupt the other masters to stop performing transfers. The ARM processor must then either enter a forever loop, or alternatively read a register in the PMU that waits the transfer until the processor has to re-awake.
- The PMU can assert a signal to the AHB arbiters that puts a dummy AHB master on the bus. For multi-layer AHB systems a MuxM2S component is required. Masters with AHB-Lite interfaces also require an AHB-Lite to AHB wrapper.

  See Figure 9-17 for a diagram showing how to force AHB masters idle.

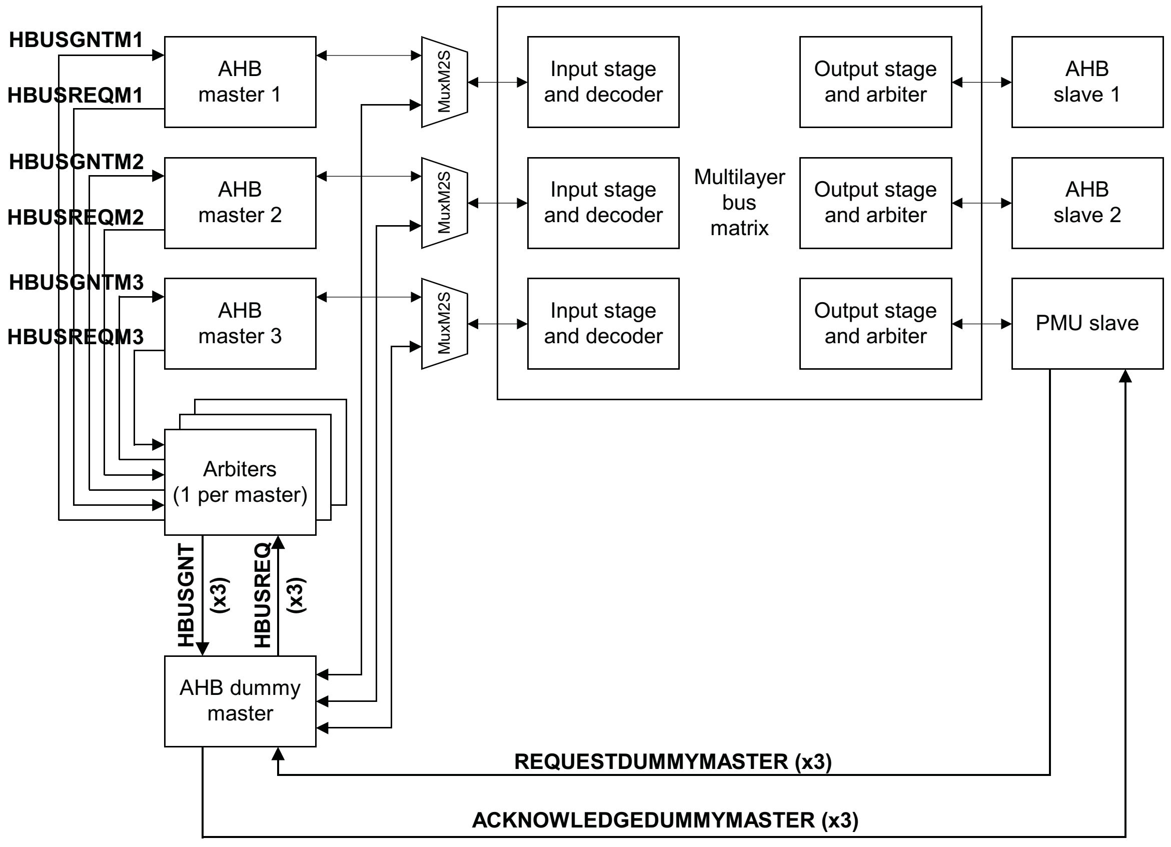


Figure 9-17 Forcing AHB masters idle

9.3 Example system

This section describes an example system. See Figure 9-18.

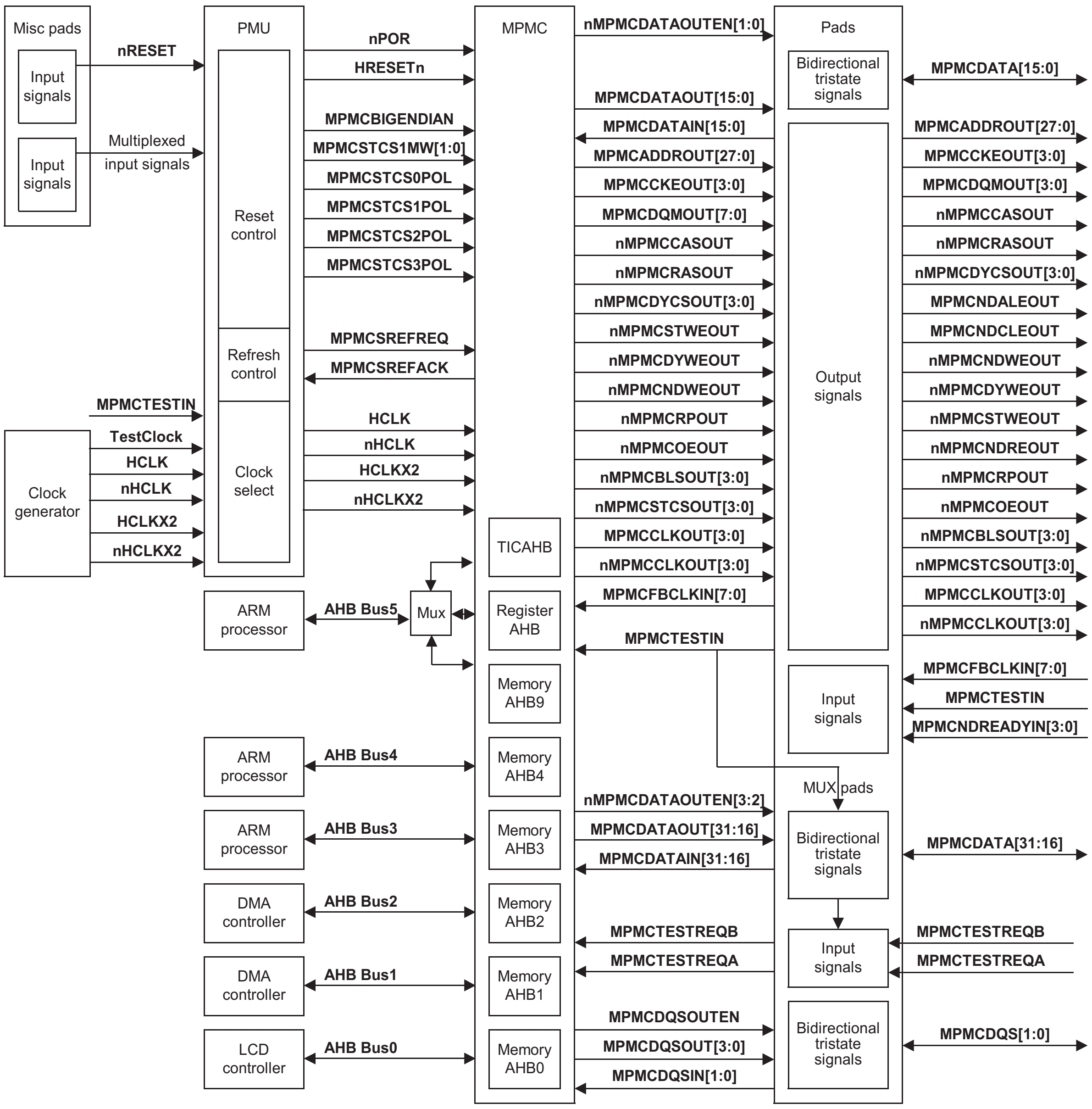


Figure 9-18 Example system

The example system is described in the following sections:

- *AHB buses*
- *PMU* on page 9-35
- *PrimeCell MPMC off-chip signals* on page 9-35.

9.3.1 AHB buses

The example system has the following AHB buses:

AHB bus 0 The highest priority PrimeCell MPMC AHB memory port, port 0, is connected to high-priority, low-latency peripheral, in this case a LCD controller.

AHB bus 1 The next highest priority PrimeCell MPMC AHB memory port, port 1, is connected to the next highest priority peripheral, in this case a DMA controller.

AHB bus 2 The PrimeCell MPMC AHB memory port 2 is connected to a DMA controller.

AHB bus 3 The PrimeCell MPMC AHB memory port 3 is connected to a second ARM processor.

AHB bus 4 The PrimeCell MPMC AHB memory port 4 is connected to a third processor.

AHB bus 5-8 These buses are unconnected.

AHB bus 9 The lowest priority AHB memory port and the AHB register port of the PrimeCell MPMC are connected to AHB bus 9 by the use of AMBA AHB multiplexor blocks. The ARM processor is also connected to this bus. The ARM processor is therefore able to program the PrimeCell MPMC and to access external memory.

AHB TIC bus

The PrimeCell MPMC AHB TIC bus is connected to the ARM processor TIC interface.

Note

Some ARM processors use the TIC interface for production test.

9.3.2 PMU

The PMU consists of the following:

- *Reset controller*
- *Refresh controller*
- *Clock select*.

Reset controller

The reset controller generates the power-on reset, **nPOR**, and AMBA reset signal, **HRESETn**.

The **MPMCBIGENDIAN**, **MPMCSTCS1MW[1:0]**, and **MPMCSTCS[3:0]POL** signals are generated by registering the value of several inputs to the chip when the power-on reset goes inactive. These input signals are reused during normal operation.

Refresh controller

The refresh controller can place the SDRAM memory devices into self-refresh mode by using the **MPMCSREFREQ** and **MPMCSREFACK** signals.

Clock select

The clock select logic is used to select the various clocks. In TIC test mode, when **MPMCTESTIN** is HIGH, **HCLK** must operate at no more than 10MHz.

9.3.3 PrimeCell MPMC off-chip signals

The PrimeCell MPMC off-chip signals are:

- *Functional mode signals*
- *Test mode signals* on page 9-36.

Functional mode signals

The system has a 16-bit data bus **MPMCDATA[15:0]**. **MPMCDATAOUT[15:0]** and **MPMCDATAIN[15:0]** are connected to bidirectional tristate pads. The direction of these pads is controlled by **MPMCDATAOUTEN[7:0]**.

It is recommended that you use clock pads, if available, for the clock signals **MPMCCLKOUT[3:0]** and **MPMCFBCLKIN[7:0]**.

The output signals, for example **MPMCADDROUT[27:0]**, must be skew balanced with the data bus. This can be performed by selecting appropriate pad cells. One possible way is to use the same bidirectional pads for the output signals as used for the data signals.

The input signals do not have to be skew balanced and so an appropriate input pad must be chosen.

Test mode signals

This system requires the use of the TIC for ARM production test. The TIC requires a 32-bit data bus, **MPMCDATA[31:0]**, and three control signals:

- **MPMCTESTREQA**
- **MPMCTESTREQB**
- **MPMCTESTACK**.

MPMCTESTREQA is multiplexed with the **nMPMCSTWEOUT** signal. The **MPMCTESTREQB**, **MPMCTESTACK**, and upper 16 bits of the data bus, **MPMCDATA[31:16]**, are signals that are required in test mode. To minimize the number of pads used these signals can be multiplexed with other signals.

Chapter 10
Off-chip Connectivity

This chapter describes the off-chip connectivity aspects relating to the MPMC. It contains the following sections:

10.1 Memory device selection

The system designer must ensure that all the memory devices connected to the MPMC work together on the same external memory bus.

10.1.1 DC characteristics

All the memory devices connected to the external memory bus of the MPMC must have the same DC characteristics. See Table 10-1 for a list of the DC characteristics for currently available memory devices.

Table 10-1 Memory device family DC characteristics

Device	3.3V	2.5V	1.8V
Asynchronous static memory	Yes	Yes	Yes
Asynchronous page mode static memory	Yes	Yes	Yes
Synchronous static memory	Yes	Yes	Yes
NAND flash	Yes	Not currently available	Yes
SDR-SDRAM	Yes	Not currently available	Not currently available
Low-power SDR-SDRAM	Yes	Yes	Yes
DDR-SDRAM	Not currently available	Yes	Not currently available
Micron SyncFlash	Yes	Not currently available	Not currently available
Micron V-SyncFlash	Not currently available	Not currently available	Yes

Note

Synchronous static memory is not supported by the MPMC.

10.1.2 Input and output cell family

Assuming that the DC characteristics of the external memory devices are the same it might be possible to connect LVTTL, LVCMOS, and SSTL_2 devices together. However, the specific drive characteristics of I/O cells for the ASIC and memory devices must be compared to ensure that they are able to work together. See Table 10-2 for a list of memory devices and the I/O cell family.

Table 10-2 Memory device I/O cell family

Device	LVTTL pads	LVCMOS pads	SSTL_2 pads
Asynchronous static memory	Yes	Yes	Not currently available
Asynchronous page mode static memory	Yes	Yes	Not currently available
Synchronous static memory	Yes	Yes	Not currently available
NAND flash	Yes	Yes	Not currently available
SDR-SDRAM	Yes	Not currently available	Not currently available
Low-power SDR-SDRAM	Yes	Yes	Not currently available
DDR-SDRAM	Not currently available	Not currently available	Yes
Micron SyncFlash	Yes	Not currently available	Not currently available
Micron V-SyncFlash	Yes	Not currently available	Not currently available

Note

If DDR-SDRAM is used it is recommended that the ASIC uses SSTL_2 pads, because they are designed to have good high-performance characteristics.

10.2 Off-chip signals

Table 10-3 shows the pins that are required to be bonded out for full memory support.

- for a 64-bit wide external data bus, 153 pins are required
- for a 32-bit wide external data bus, 111 pins are required
- for a 16-bit wide external data bus, 82 pins are required.

The DDR-SDRAM data strobes and the SDRAM fed-back clocks are on separate pins so that both memory types can be mixed on a bank-by-bank basis.

Note

The tie-off signals are not included in Table 10-3. Depending on the system requirements, these can be tied off either on-chip or off-chip.

Table 10-3 Bonded out pins

Pin Name	SRAM	NAND flash	SDRAM	TIC
MPMCBOOTDLY	N/a	N/a	Delay memory accesses	N/a
MPMCCLKOUT [3:0] DDR-SDRAM +ve clock out	N/a	N/a	DDR-SDRAM +ve clock out/SDRAM clock out	N/a
nMPMCCLKOUT [3:0] DDR-SDRAM -ve clock out	N/a	N/a	DDR-SDRAM -ve clock out	N/a
MPMCFBCLKIN [7:0] SDRAM fed-back clock in	N/a	N/a	SDRAM fed-back clock in	N/a
MPMCCKEOUT [3:0]	N/a	N/a	Clock enable	N/a
MPMCDQS[3:0] DDR-SDRAM data strobe	N/a	N/a	DDR-SDRAM bidirectional data strobe (**MPMCDQSOUT [3:0]** / **MPMCDQSIN[3:0]**)	N/a
nMPMCDYCSOUT [3:0]	N/a	N/a	Chip select	N/a

Table 10-3 Bonded out pins (continued)

Pin Name	SRAM	NAND flash	SDRAM	TIC
nMPMCSTCSOUT [3:0]	Chip select	Chip select	N/a	N/a
MPMCADDROUT [27:15]	Address out [27:15]	N/a	N/a	N/a
MPMCADDROUT [14:0]	Address out [14:0]	N/a	Address out [14:0]	N/a
MPMCAPOUT	N/a	N/a	SDRAM auto precharge, Micron SyncFlash/ V-SyncFlash Active Terminate control	N/a
nMPMCCASOUT	N/a	N/a	Column address strobe	N/a
nMPMCRASOUT	N/a	N/a	Row address strobe.	N/a
nMPMCDYWEOUT	N/a	N/a	Write enable	N/a
nMPMCSTWEOUT	Write enable	N/a	N/a	**TESTACK**
nMPMCOEOUT	Output enable	N/a	N/a	N/a
MPMCDATA[63:32]	N/a	N/a	SDRAM bidirectional data (**MPMCDOUT[63:32]/MPMCDIN[63:32]**)	N/a
MPMCDATA[31:16]	Bidirectional data (**MPMCDOUT [31:16]/MPMCDIN[31:16]**)	N/a	Bidirectional data (**MPMCDOUT [31:16]/MPMCDIN[31:16]**)	Bidirectional data (**MPMCDOUT [31:16]/MPMCDIN[31:16]**)
MPMCDATA[15:8]	Bidirectional data (**MPMCDOUT [15:8]/MPMCDIN [15:8]**)	Bidirectional data (**MPMCDOUT [15:8]/ MPMCDIN[15:8]**)	Bidirectional data (**MPMCDOUT [15:8]/ MPMCDIN[15:8]**)	Bidirectional data (**MPMCDOUT [15:8]/MPMCDIN[15:8]**)
MPMCDATA[7:0]	Bidirectional data (**MPMCDOUT [7:0] / MPMCDIN[7:0]**)	Command, address and bidirectional data (**MPMCDOUT[7:0]/ MPMCDIN[7:0]**)	Bidirectional data (**MPMCDOUT[7:0] / MPMCDIN[7:0]**)	Bidirectional data (**MPMCDOUT[7:0] / MPMCDIN[7:0]**)

Table 10-3 Bonded out pins (continued)

Pin Name	SRAM	NAND flash	SDRAM	TIC
MPMCDQMOUT [7:4]	N/a	N/a	Byte select upper word SDRAM	N/a
MPMCDQMOUT [3:0]	N/a	N/a	Byte select lower word SDRAM and upper/lower word DDR-SDRAM	N/a
nMPMCBLSOUT [3:0]	Byte lane select	N/a	N/a	N/a
MPMCNDALEOUT	N/a	Address latch enable	N/a	N/a
MPMCNDCLEOUT	N/a	Command latch enable	N/a	N/a
nMPMCNDWEOUT	N/a	Write enable	N/a	N/a
nMPMCNDREOUT	N/a	Read enable	N/a	N/a
MPMCNDREADYIN [3:0]	N/a	Ready/busy	N/a	N/a
nMPMCRPOUT	N/a	N/a	Reset power down	N/a
MPMCRPVHHOUT	N/a	N/a	Voltage control for **nMPMCRPOUT**	N/a
MPMCTESTREQA	N/a	N/a	N/a	TIC test request A
MPMCTESTREQB	N/a	N/a	N/a	TIC test request B
MPMCTESTIN Test mode	N/a	N/a	N/a	TIC test mode select

Note

V-SyncFlash devices require an additional three address outputs for the segment address. 512Mb devices also require a 14th address bit output.

The following sections show how the pin count can be reduced.

The **MPMCTESTREQA** and **MPMCTESTREQB** signals are only required in TIC test mode and can therefore be multiplexed with other signals.

The **MPMCRPVHHOUT** signal is only required to be bonded out if:

- Micron SyncFlash support is required
- high voltage VHH nRP support is required
- the high voltage generator is not on the chip.

10.2.1 Pin count reduction by reducing data bus width

The data bus width can be reduced from 64 bits to 32 bits or 16 bits. However, this impacts on the performance of the MPMC for SDR-SDRAM.

The widths of the following control signals can also be reduced according to the data bus width and memory devices used:

- **MPMCCLKOUT[3:0]**
- **nMPMCCLKOUT[3:0]**
- **MPMCFBCLKIN[7:0]**
- **MPMCCKEOUT[3:0]**
- **MPMCDQS[3:0]**
- **MPMCDQMOUT[7:0]**
- **nMPMCBLSOUT[3:0]**.

Note

DDR-SDRAM data bus width can only be 16 or 32 bits. This provides the same performance as SDR-SDRAM with a 32 or 64-bit data bus width respectively.

10.2.2 Pin count reduction by removing functionality

You can reduce the pin count if you do not require certain functionality. The pin count reductions that can be achieved by removing functionality are described in the following sections:

- *Dynamic and static memory, NAND flash memory, and TIC* on page 10-8
- *Dynamic and static memory, and TIC* on page 10-8
- *Dynamic and static memory, and NAND flash memory* on page 10-9
- *Dynamic and static memory* on page 10-9
- *Dynamic and NAND flash memory, and TIC* on page 10-10
- *Dynamic memory and TIC* on page 10-10
- *Static and NAND flash memory, and TIC* on page 10-11
- *Static memory and TIC* on page 10-12
- *NAND flash memory and TIC* on page 10-13
- *Dynamic memory* on page 10-14

- *Static memory* on page 10-15
- *NAND flash memory* on page 10-16.

Note

The signals that are normally multiplexed or not taken off chip are not counted in Table 10-4 to Table 10-15 on page 10-16. The signals are **MPMCRPVHHOUT**, **MPMCTESTREQA**, and **MPMCTESTREQB**.

Dynamic and static memory, NAND flash memory, and TIC

Table 10-4 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic and static memory, NAND flash memory, and a TIC.

Table 10-4 Pin counts for system with dynamic and static memory, NAND flash memory, and TIC

Data bus	Control	Address	Total
16	38	28	82
32	51	28	111
64	61	28	153

Dynamic and static memory, and TIC

Table 10-5 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic and static memory and a TIC.

Table 10-5 Pin counts for system with dynamic and static memory, and TIC

Data bus	Control	Address	Total
16	30	28	74
32	43	28	103
64	53	28	145

The following signals are not required (NAND flash memory):

- **MPMCNDALEOUT**
- **MPMCNDCLEOUT**
- **nMPMCNDWEOUT**
- **nMPMCNDREOUT**

- **MPMCNDREADYIN[3:0]**.

Dynamic and static memory, and NAND flash memory

Table 10-6 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic and static memory, and NAND flash memory.

Table 10-6 Pin counts for system with dynamic and static memory, and NAND flash memory

Data bus	Control	Address	Total
16	37	28	81
32	50	28	110
64	60	28	152

The following signals are not required (test):

- **MPMCTESTIN**.

Dynamic and static memory

Table 10-7 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic and static memory.

Table 10-7 Pin counts for system with dynamic and static memory

Data bus	Control	Address	Total
16	30	28	73
32	42	28	102
64	52	28	144

The following signals are not required (test):

- **MPMCTESTIN**.

The following signals are not required (NAND flash memory):

- **MPMCNDALEOUT**
- **MPMCNDCLEOUT**
- **nMPMCNDWEOUT**
- **nMPMCNDREOUT**
- **MPMCNDREADYIN[3:0]**.

Dynamic and NAND flash memory, and TIC

Table 10-8 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic and NAND flash memory, and a TIC.

Table 10-8 Pin counts for system with dynamic and NAND flash memory, and TIC

Data bus	Control	Address	Total
16	33	14	63
32	46	14	92
64	56	14	134

The following signals are not required (address):

- **MPMCADDROUT[27:15]**
- **MPMCADDROUT[8/10]**.

The following signals are not required (static memory):

- **nMPMCOEOUT**
- **nMPMCBLSOUT[3:0]**.

Note

Systems using V-SyncFlash devices require an additional three or four address outputs, depending on the device size.

Dynamic memory and TIC

Table 10-9 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic memory and a TIC.

Table 10-9 Pin counts for system with dynamic memory and TIC

Data bus	Control	Address	Total
16	21	14	51
32	34	14	80
64	44	14	122

The following signals are not required (address):

- **MPMCADDROUT[27:15]**

- **MPMCADDROUT[8/10]**.

The following signals are not required (static memory):

- **nMPMCOEOUT**
- **nMPMCBLSOUT[3:0]**
- **nMPMCSTCSOUT[3:0]**.

The following signals are not required (NAND flash memory):

- **MPMCNDALEOUT**
- **MPMCNDCLEOUT**
- **nMPMCNDWEOUT**
- **nMPMCNDREOUT**
- **MPMCNDREADYIN[3:0]**.

——— **Note** ———

Systems using V-SyncFlash devices require an additional three or four address outputs, depending on the device size.

Static and NAND flash memory, and TIC

Table 10-10 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing static and NAND flash memory, and a TIC.

Table 10-10 Pin counts for system with static and NAND flash memory, and TIC

Data bus	Control	Address	Total
16	17	28	61
32	19	28	79
64	-	-	-

——— **Note** ———

64-bit wide static memory is not supported.

The following signals are not required (dynamic memory):

- **MPMCBOOTDLY**
- **MPMCCLKOUT[3:0]**
- **nMPMCCLKOUT[3:0]**
- **MPMCFBCLKIN[7:0]**

- **MPMCCKEOUT[3:0]**
- **MPMCDQS[3:0]**
- **nMPMCDYCSOUT[3:0]**
- **MPMCAPOUT**
- **nMPMCCASOUT**
- **nMPMCRASOUT**
- **nMPMCDYWEOUT**
- **MPMCDATA[63:32]**
- **MPMCDQMOUT[7:0]**
- **nMPMCRPOUT**.

Static memory and TIC

Table 10-11 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing static memory and a TIC.

Table 10-11 Pin counts for system with static memory and TIC

Data bus	Control	Address	Total
16	9	28	53
32	11	28	71
64	-	-	-

Note

64-bit wide static memory is not supported.

The following signals are not required (dynamic memory):

- **MPMCBOOTDLY**
- **MPMCCLKOUT[3:0]**
- **nMPMCCLKOUT[3:0]**
- **MPMCFBCLKIN[7:0]**
- **MPMCCKEOUT[3:0]**
- **MPMCDQS[3:0]**
- **nMPMCDYCSOUT[3:0]**
- **MPMCAPOUT**
- **nMPMCCASOUT**
- **nMPMCRASOUT**

- **nMPMCDYWEOUT**
- **MPMCDATA[63:32]**
- **MPMCDQMOUT[7:0]**
- **nMPMCRPOUT**.

The following signals are not required (NAND flash memory):

- **MPMCNDALEOUT**
- **MPMCNDCLEOUT**
- **nMPMCNDWEOUT**
- **nMPMCNDREOUT**
- **MPMCNDREADYIN[3:0]**.

NAND flash memory and TIC

Table 10-12 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing NAND flash memory and a TIC.

Table 10-12 Pin counts for system with NAND flash memory and TIC

Data bus	Control	Address	Total
16	14	-	30
32	-	-	-
64	-	-	-

Note

32-bit and 64-bit wide NAND flash memory is not supported.

The following signals are not required (address):

- **MPMCADDROUT[27:0]**.

The following signals are not required (static memory):

- **nMPMCOEOUT**
- **nMPMCBLSOUT[3:0]**.

The following signals are not required (dynamic memory):

- **MPMCBOOTDLY**
- **MPMCCLKOUT[3:0]**
- **nMPMCCLKOUT[3:0]**
- **MPMCFBCLKIN[7:0]**

- **MPMCCKEOUT[3:0]**
- **MPMCDQS[3:0]**
- **nMPMCDYCSOUT[3:0]**
- **MPMCAPOUT**
- **nMPMCCASOUT**
- **nMPMCRASOUT**
- **nMPMCDYWEOUT**
- **MPMCDATA[63:32]**
- **MPMCDQMOUT[7:0]**
- **nMPMCRPOUT**.

Dynamic memory

Table 10-13 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing dynamic memory.

Table 10-13 Pin counts for system with dynamic memory

Data bus	Control	Address	Total
16	20	14	50
32	33	14	79
64	43	14	121

The following signals are not required (address):

- **MPMCADDROUT[27:15]**
- **MPMCADDROUT[8/10]**.

The following signals are not required (static memory):

- **nMPMCOEOUT**
- **nMPMCBLSOUT[3:0]**
- **nMPMCSTCSOUT[3:0]**.

The following signals are not required (NAND flash memory):

- **MPMCNDALEOUT**
- **MPMCNDCLEOUT**
- **nMPMCNDWEOUT**
- **nMPMCNDREOUT**
- **MPMCNDREADYIN[3:0]**.

The following signals are not required (test):

- **MPMCTESTIN**.

——— **Note** ———

Systems using V-SyncFlash devices require an additional three or four address outputs, depending on the device size.

Static memory

Table 10-14 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing static memory.

Table 10-14 Pin counts for system with static memory

Data bus	Control	Address	Total
16	8	28	52
32	10	28	70
64	-	-	-

——— **Note** ———

64-bit wide static memory is not supported.

The following signals are not required (dynamic memory):

- **MPMCBOOTDLY**
- **MPMCCLKOUT[3:0]**
- **nMPMCCLKOUT[3:0]**
- **MPMCFBCLKIN[7:0]**
- **MPMCCKEOUT[3:0]**
- **MPMCDQS[3:0]**
- **nMPMCDYCSOUT[3:0]**
- **MPMCAPOUT**
- **nMPMCCASOUT**
- **nMPMCRASOUT**
- **nMPMCDYWEOUT**
- **MPMCDATA[63:32]**
- **MPMCDQMOUT[7:0]**
- **nMPMCRPOUT**

- **MPMCRPVHHOUT**.

The following signals are not required (NAND flash memory):

- **MPMCNDALEOUT**
- **MPMCNDCLEOUT**
- **nMPMCNDWEOUT**
- **nMPMCNDREOUT**
- **MPMCNDREADYIN[3:0]**.

The following signals are not required (test):

- **MPMCTESTIN**.

NAND flash memory

Table 10-15 shows the pin counts that can be achieved for 64-bit, 32-bit, and 16-bit systems containing NAND flash memory.

Table 10-15 Pin counts for system with NAND flash memory

Data bus	Control	Address	Total
16	12	-	28
32	-	-	-
64	-	-	-

Note

32-bit and 64-bit wide NAND flash memory is not supported.

The following signals are not required (address):

- **MPMCADDROUT[27:0]**.

The following signals are not required (static memory):

- **nMPMCOEOUT**
- **nMPMCBLSOUT[3:0]**.

The following signals are not required (dynamic memory):

- **MPMCBOOTDLY**
- **MPMCCLKOUT[3:0]**
- **nMPMCCLKOUT[3:0]**
- **MPMCFBCLKIN[7:0]**

- **MPMCCKEOUT[3:0]**
- **MPMCDQS[3:0]**
- **nMPMCDYCSOUT[3:0]**
- **MPMCAPOUT**
- **nMPMCCASOUT**
- **nMPMCRASOUT**
- **nMPMCDYWEOUT**
- **MPMCDATA[63:32]**
- **MPMCDQMOUT[7:0]**
- **nMPMCRPOUT**
- **MPMCRPVHHOUT**.

The following signals are not required (test):

- **MPMCTESTIN**.

10.2.3 Address pin reduction

Depending on the size of the memory parts, not all the address pins have to be bonded out. For example, if 32Kx32 or 32Kx16 or smaller devices static memory devices are supported, address lines **MPMCADDROUT[27:15]** are not required. In this case, removing these address lines does not reduce the number of dynamic memory devices supported.

10.2.4 Chip select pin reduction

Depending on the number of memory chip selects required, not all of the chips selects have to be bonded out.

10.2.5 Device support

If the SyncFlash and V-SyncFlash devices are not required to be supported, the **nMPMCRPOUT** pin is not required.

10.2.6 Multiplexing static and dynamic memory pins

SDRAM memories use a multiplexed address. These devices use up to 15 address pins.

Static memories do not perform address multiplexing and so require more address pins.

For systems where the SDRAM memory chip selects data bus is wider than the static memory chip selects data bus, the upper static memory address bits can be multiplexed with the upper bits of the SDRAM data bus to reduce the pin count.

For example, in a system where the static memory devices use a 16-bit data bus and 24 address pins, and the SDRAM memory devices use a 32-bit data bus and 15 address pins, the static memory controller uses more address pins than the dynamic memory controller, and the dynamic memory controller uses more data pins than the static memory controller. Because only one transaction can be active at a time, it is possible to multiplex the top eight bits of the SDRAM data pins with the top eight bits of the static memory address pins. This can be performed by ORing the relevant **MPMCADDROUT** and **MPMCDATAOUT** signals externally.

——— **Note** ———

If the above optimization is to be performed then appropriate testing is required, ensuring that the pads are driven in the correct direction for static and dynamic memory accesses.

————————

10.2.7 Multiplexing PrimeCell MPMC pins

Total pin count can be reduced by multiplexing the MPMC signals with another device. This is normally performed using an EBI. For more information on the EBI, see *EBI connectivity* on page 9-18.

——— **Note** ———

Multiplexing PrimeCell MPMC pins has a detrimental effect on the MPMC pad interface performance, because both on and off-chip signal loading is increased. For many system designs this trade-off is acceptable.

————————

10.3 Clock relationships

HCLK is used to clock the majority of the logic in the MPMC, including the AHB interface logic, the registers, and the MPMC state machines. **HCLK** is also used to generate the output clock **MPMCCLKOUT**.

The **nHCLK** signal is used to register the read data on to the negative edge of **HCLK** before being registered on the positive edge of **HCLK**. **nHCLK** is also used to generate the differential output clock **nMPMCCLKOUT**. **nHCLK** can be tied LOW if the differential output clock and capturing data on the negative edge of **HCLK** is not required.

The **MPMCHCLKDELAY** signal is used in the command delayed pad interface methodology. **MPMCHCLKDELAY** can be tied LOW if not required. For more information on **MPMCHCLKDELAY** requirements see *Pad interface methodology* on page 10-22.

All the clocks must have a 50:50 duty cycle for maximum performance.

The positive edge of **HCLK** and **HCLKX2** must be aligned. The positive edge of **HCLK** must be aligned with the negative edge of **nHCLK**. The positive edge of **HCLKX2** must be aligned with the negative edge of **nHCLKX2**.

Figure 10-1 shows the clock relationships.

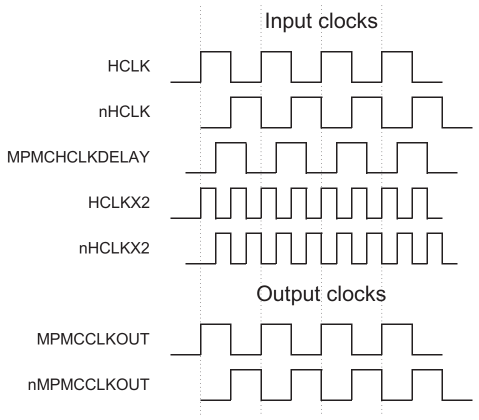


Figure 10-1 Clock relationships

10.4 Pad interface timing

The on-chip path (the path from the PrimeCell MPMC to the pads), and the off-chip path (the path from the chips I/O pins to the memory devices) are critical for high-speed operation.

10.4.1 On-chip timing path

To enable high-speed operation the skew between the MPMC pad interface output signals must be minimized.

- This is achieved by the MPMC:
  - providing four output clocks, **MPMCCLKOUT[3:0]**, to reduce clock signal loading
  - registering all the output signals (ignoring a minimal amount of multiplexing) before they are sent off chip to minimize skew between the signals.
- The system designer must ensure that:
  - the MPMC is situated as close to the pads as possible to minimize signal delay and skew
  - the MPMC output signals are not multiplexed, because this can unnecessarily load the signals.
  - appropriate strength pads are used for the output signals.

The skew between the MPMC pad interface input signals must be minimized.

- This is achieved by the MPMC:
  - using a feedback clock that is ideally feedback from the memory devices clock pin
  - providing eight feedback clocks, **MPMCFBCLKIN[7:0]**, to better track the delays of the return data path
  - registering the input signals, using the **MPMCFBCLKIN[7:0]** signals, when they come on chip to minimize skew.
- The system designer must ensure that:
  - the MPMC is situated as close to the pads as possible to minimize signal delay and skew
  - the MPMC output signals are not multiplexed, because this can unnecessarily load the signals.
  - appropriate strength pads are used for the input signals

— an external DLL is used to delay the data strobe input signals by an appropriate amount to enable the DDR-SDRAM read data to be captured correctly.

10.4.2 Off-chip timing path

You must minimize the skew between the various signals during PCB routing.The off-chip path from the ASIC pads to the dynamic memory devices must be as lightly loaded as possible to enable high-speed operation. Therefore, you must minimize the number of memory devices for high-speed operation.

For systems that require a large number of memory devices, additional buffering of the I/O signals might be required. For example, the signals to the static memory devices can be buffered. This adds additional delay when accessing static memories, but enables the dynamic devices to run at a higher speed.

10.5 Pad interface methodology

The SDRAM clock must not be coincident with the SDRAM command signals, address, or write data, otherwise the SDRAM setup and hold requirements are violated. To meet these requirements the MPMC supports two pad interface methodologies:

- *Clock delay methodology* on page 10-23
- *Command delay methodology* on page 10-29.

——— **Note** ———

The SDR-SDRAM and DDR-SDRAM pad interface methodologies are programmed by separate register fields. The programmed methodologies must be the same although the delays might be different.

10.6 Clock delay methodology

The clock methodology is described in the following sections:

- *Clock delay*
- *Clock delay advantages* on page 10-26
- *Clock delay disadvantages* on page 10-27
- *Data capture requirements* on page 10-27
- *Generation of the delayed clock* on page 10-27
- *Read data capture polarity* on page 10-28.

10.6.1 Clock delay

The clock delay methodology requires the output clock to be delayed externally to the MPMC. The output clock must be delayed by more than the SDRAM setup time. However, the output clock must not be delayed too much as the SDRAM hold requirements must also be met.

This methodology is most suited to systems where the SDRAM clock frequency is not high, and where the worst case read data delay is less than a cycle.

Clock delay methodology is generally not suited for DDR-SDRAM memory devices. This is because with DDR-SDRAM devices, the read data is returned on both edges of the data strobes. The second data item is therefore delayed by half a cycle more than the first data item, meaning that the read data delay is highly constrained.

This clock delay methodology can be selected by programming the MPMCDynamicReadConfig Registers, DDR-SDRAM read data strategy (DRD), or SDR-SDRAM read data strategy (SRD) fields.

Figure 10-2 on page 10-24 shows the clock delayed pad interface read timing.

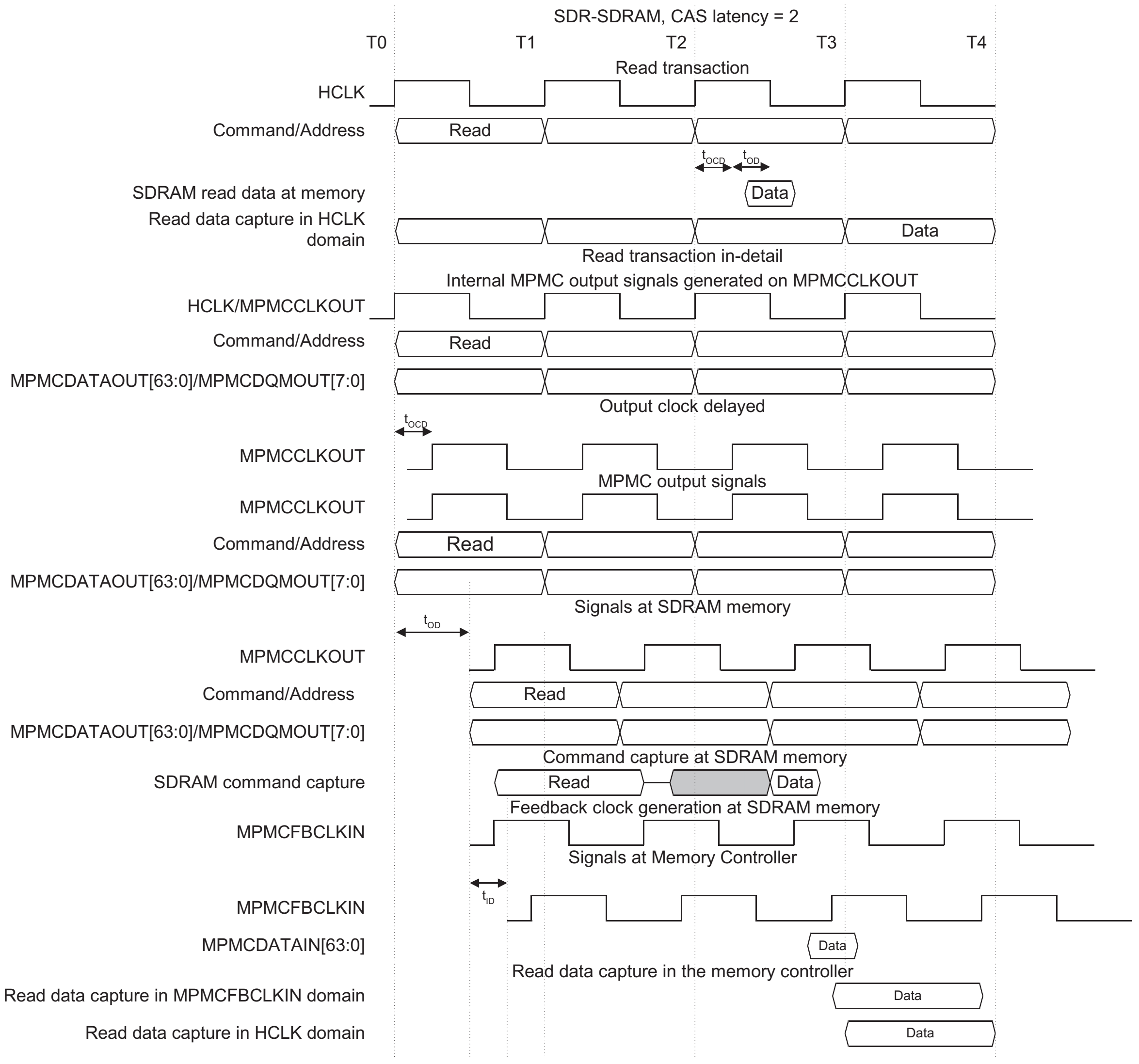

SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
Read transaction
HCLK
Command/Address
Read
t_{OCD}
t_{OD}
SDRAM read data at memory
Data
Read data capture in HCLK domain
Data
Read transaction in-detail
Internal MPMC output signals generated on MPMCCLKOUT
HCLK/MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/MPMCDQMOUT[7:0]
Output clock delayed
t_{OCD}
MPMCCLKOUT
MPMC output signals
MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/MPMCDQMOUT[7:0]
Signals at SDRAM memory
t_{OD}
MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/MPMCDQMOUT[7:0]
Command capture at SDRAM memory
SDRAM command capture
Read
Data
Feedback clock generation at SDRAM memory
MPMCFBCLKIN
Signals at Memory Controller
t_{ID}
MPMCFBCLKIN
MPMCDATAIN[63:0]
Data
Read data capture in the memory controller
Read data capture in MPMCFBCLKIN domain
Data
Read data capture in HCLK domain
Data

Figure 10-2 Clock delayed pad interface read timing

Figure 10-2 is described in the following sections:

- *Read transaction* on page 10-25
- *Internal MPMC output signals generated on MPMCCLKOUT* on page 10-25
- *Output clock delayed* on page 10-25

- *MPMC output signals*
- *Signals at SDRAM memory* on page 10-26
- *Command capture at SDRAM memory* on page 10-26
- *Feedback clock generation at SDRAM memory* on page 10-26
- *Signals at MPMC* on page 10-26
- *Read data capture in the MPMC* on page 10-26.

Read transaction

The SDRAM command, address, and clock are posted to the SDRAM during cycle T0-T1.

The SDRAM memory provides the read data (at the memory) during cycle T2-T3.

The read data is captured in the MPMC, in the **HCLK** domain at T3.

Internal MPMC output signals generated on MPMCCLKOUT

Internally the logic of the MPMC is clocked by **HCLK**. This clock is also used to generate the external clock **MPMCCLKOUT**.

To meet SDRAM memory setup and hold requirements the SDRAM clock and command transitions must not be coincident.

Output clock delayed

In the clock delayed pad interface methodology, the output clock **MPMCCLKOUT** must be delayed externally to the MPMC. This is to ensure that the commands meet the SDRAM setup and hold requirements.

Generally SDR-SDRAM memory devices require about 1.5ns of setup and hold time. In Figure 10-2 on page 10-24 this *Output Clock Delay* (OCD) is indicated as t_{OCD}.

MPMC output signals

The resulting signals provided to the pads before they go off chip are shown. The command and address are not delayed.

The output clock is delayed by t_{OCD}. These signals now meet the SDRAM memory requirements.

Signals at SDRAM memory

The output signals (clock, address, and commands) are subjected to an *Output Delay* (OD), going to the SDRAM memory. t_{OD} is composed of the pad delay and the PCB delay.

Command capture at SDRAM memory

The SDRAM read command is captured on the positive edge of **MPMCCLKOUT** at the SDRAM memory.

For CAS 2 latency SDR-SDRAM reads, the read data is valid on the second positive edge of the memory clock after the read command is captured. SDR-SDRAM provides valid read data a few nanoseconds before and after the SDRAM clock edge to make capturing the read data easier.

Feedback clock generation at SDRAM memory

The feedback clock is ideally generated from an output clock with a similar PCB and pad delay as the SDRAM clock. This is so that the feedback clocks positive edge is aligned in the middle of the time when the read data is valid.

Signals at MPMC

The input signals (read data and feedback clock) are subjected to an *Input Delay* (ID) from the SDRAM to the MPMC.

t_{ID} is composed of a PCB delay and a pad delay.

Read data capture in the MPMC

The read data is captured using the feedback clock **MPMCFBCLKIN**. This ensures that the read data is captured cleanly. The read data is then captured in the **HCLK** clock domain.

10.6.2 Clock delay advantages

The clock delay methodology provides data a cycle earlier for a read access than the command delay methodology. This improves the read burst access latency by a cycle compared to the command delay methodology.

10.6.3 Clock delay disadvantages

The main disadvantage with the clock delay methodology is that the read capture timing is more tightly constrained.

The read delay time must be less than a clock period.

Another disadvantage is that the changing, dynamic, read skew can be affected by the jitter added by the delayed clock.

10.6.4 Data capture requirements

Both the worst case and best case dynamic skew, and read data delay must meet the following requirements otherwise the MPMC is not able to capture the read data correctly.

Maximum read data delay

The maximum read data delay is: Maximum read data delay = **HCLK** clock period - $t_{setup\mathbf{FBCLK}}$ - $t_{setup\mathbf{HCLK}}$

Alternatively this can be expressed as:($t_{OCD} + t_{OD} + t_{ID} + t_{setup\mathbf{FBCLK}} + t_{setup\mathbf{HCLK}}$) < **HCLK** clock periodWhere t_{OCD} is the output clock delay, t_{OD} is the output delay, and t_{ID} is the input delay.

Minimum read data delay

The minimum read data delay is $t_{hold\mathbf{HCLK}}$

Maximum dynamic skew

The maximum dynamic skew allowed in this pad methodology is: Maximum dynamic skew = maximum read delay - minimum read data delay Maximum dynamic skew = **HCLK** clock period - $t_{setup\mathbf{FBCLK}}$ - $tsetup\mathbf{HCLK}$ - $t_{hold\mathbf{HCLK}}$

10.6.5 Generation of the delayed clock

To meet both the setup and hold requirements for the SDRAM a 180-degree phase shifted output clock can be used. A simple way to generate this is to invert the output clock externally to the MPMC. This adds an additional half a clock skew to the return read data. This methodology means there is no additional clock dynamic skew.

10.6.6 Read data capture polarity

It is recommended that the read data is captured on the positive edge of **HCLK** in clock delayed mode.

The read data can be captured either on the positive edge of **HCLK** or the negative of **HCLK** by programming the MPMCDynamicReadConfig Register.

10.7 Command delay methodology

The command delay methodology is described in the following sections:

- *Command delay plus zero* on page 10-30
- *Command delay advantages*
- *Command delay disadvantages*
- *Generation of the delayed clock* on page 10-30
- *Read data capture polarity* on page 10-30
- *Command delay plus zero SDRAM timing diagrams* on page 10-33
- *Command delay plus one* on page 10-35
- *Command delay plus one SDRAM timing diagrams* on page 10-37
- *Command delay plus two* on page 10-40
- *Command delay plus two SDRAM timing diagrams* on page 10-42.

10.7.1 Command delay advantages

The main advantage of the command delay methodology is that the read data can be more easily captured than for the clock delay methodology.

For command delay plus zero mode the read data delay must be less than a **HCLK** cycle. For command delay plus one mode the read data delay must be less than two **HCLK** cycles. For command delay plus two mode the read data delay must be less than three **HCLK** cycles.

The worst case maximum dynamic read skew is half a **HCLK** period.

10.7.2 Command delay disadvantages

The command delay methodology plus 0 provides the read data a cycle later than the clock delay methodology, increasing the read burst access latency by a clock cycle.

The command delay methodology plus 1 provides the read data two cycles later than the clock delay methodology, increasing the read burst access latency by two clock cycles.

The command delay methodology plus 2 provides the read data three cycles later than the clock delay methodology, increasing the read burst access latency by three clock cycles.

10.7.3 Generation of the delayed clock

To meet both the setup and hold requirements for the SDRAM a 180-degree phase shifted **MPMCHCLKDELAY** can be used. A simple way to generate this is to invert the output clock externally to the MPMC.

10.7.4 Read data capture polarity

The read data can be captured either on the positive or negative edge of **HCLK**, the value chosen depending on the read data delay:

- If the minimum and maximum read data delay does not overlap the positive edge of **HCLK** (but overlaps the negative edge of **HCLK**), the read data must be captured on the positive edge of **HCLK**.
- If the minimum and maximum read data delay does not overlap the negative edge of **HCLK** (but overlaps the negative edge of **HCLK**), the read data must be captured on the negative edge of **HCLK**.

The read data can be captured either on the positive or the negative edge of **HCLK** by programming either the MPMCDynamicReadConfig Register, SDR-SDRAM read data capture polarity (SRP) field, or the DDR-SDRAM read data capture polarity (DRP) polarity field depending on the device being configured.

10.7.5 Command delay plus zero

The command delay methodology requires the output signals (the address, control signals and write data) to be delayed compared to the output clock **MPMCCLK**. The output signals are delayed by using the **MPMCHCLKDELAY** clock signal. The output signals must be delayed by more than the SDRAM hold time. The output signals must not be delayed too much, because the SDRAM setup requirements must also be met.

This methodology is most suited to systems where the SDRAM clock frequency is high. The command delay methodology is suited for DDR-SDRAM memory devices. This command delay methodology can be selected by programming the MPMCDynamicReadConfig Registers, DDR-SDRAM read data strategy (DRD), or SDR-SDRAM read data strategy (SRD) fields. Figure 10-3 on page 10-31 shows the command delayed pad interface read timing.

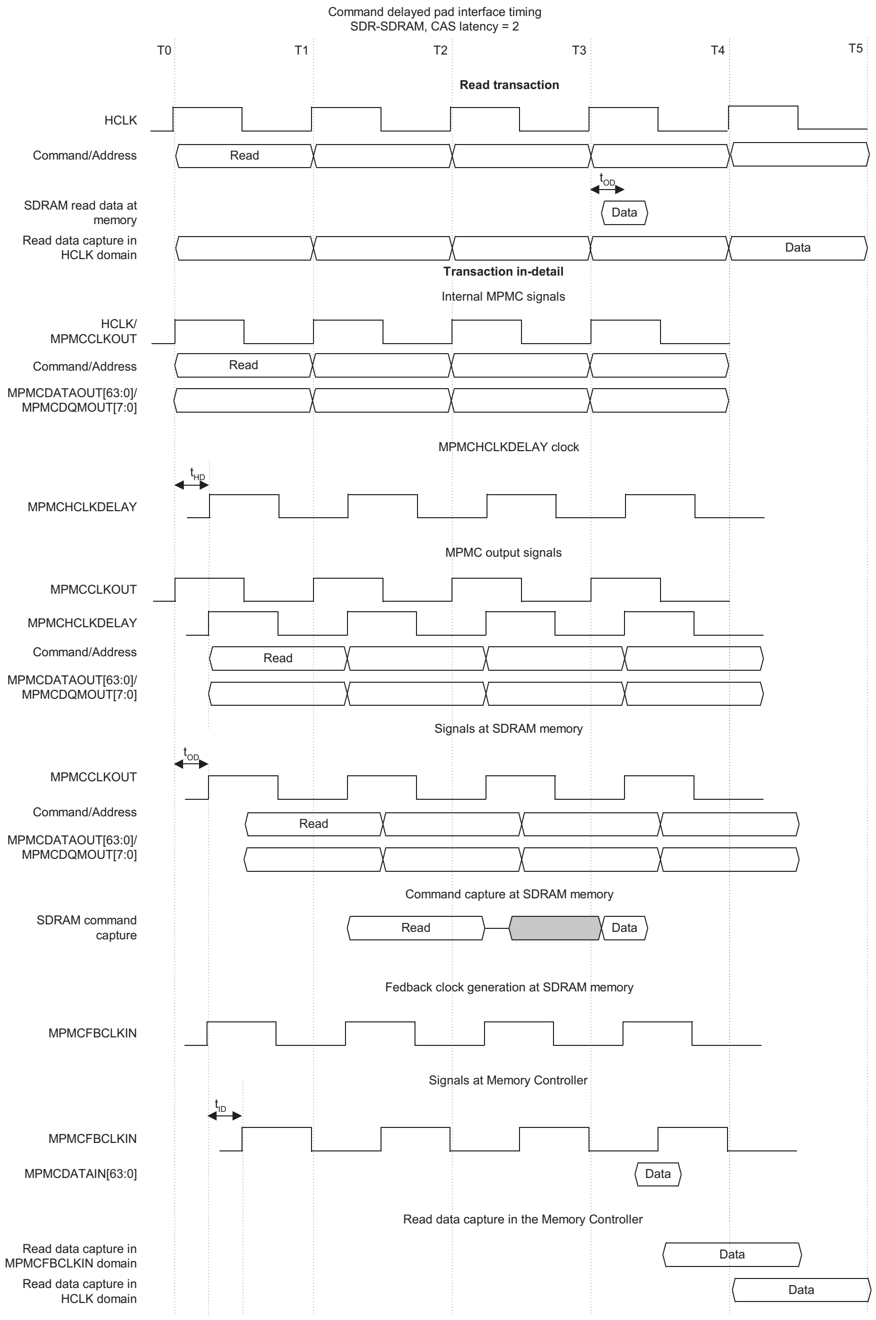

Command delayed pad interface timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
Read transaction
HCLK
Command/Address
Read
t_{OD}
SDRAM read data at memory
Data
Read data capture in HCLK domain
Data
Transaction in-detail
Internal MPMC signals
HCLK/ MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
MPMCHCLKDELAY clock
t_{HD}
MPMCHCLKDELAY
MPMC output signals
MPMCCLKOUT
MPMCHCLKDELAY
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
Signals at SDRAM memory
t_{OD}
MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
Command capture at SDRAM memory
SDRAM command capture
Read
Data
Fedback clock generation at SDRAM memory
MPMCFBCLKIN
Signals at Memory Controller
t_{ID}
MPMCFBCLKIN
MPMCDATAIN[63:0]
Data
Read data capture in the Memory Controller
Read data capture in MPMCFBCLKIN domain
Data
Read data capture in HCLK domain
Data

Figure 10-3 Command delayed pad interface timing

Figure 10-3 on page 10-31 is described in the following sections:

- *Read transaction*
- *Internal MPMC output signals generated on MPMCCLKOUT*
- *MPMCHCLKDELAY clock*
- *MPMC output signals*
- *Signals at SDRAM memory* on page 10-33
- *Command capture at SDRAM memory* on page 10-33
- *Feedback clock generation at SDRAM memory* on page 10-33
- *Signals at MPMC* on page 10-33
- *Read data capture in the MPMC* on page 10-33.

Read transaction

The SDRAM command, address, and clock are posted to the SDRAM at time T0-T1.

The SDRAM memory provides the read data (at the memory) during cycle T3-T4.

The read data is captured in the MPMC, in the **HCLK** domain at T4.

Internal MPMC output signals generated on MPMCCLKOUT

Internally the MPMC logic is clocked by **HCLK**. This clock is also used to generate the external clock **MPMCCLKOUT**.

To meet SDRAM memory setup and hold requirements, the SDRAM clock and command transitions must not be coincident.

MPMCHCLKDELAY clock

In the command delayed pad interface methodology, the delayed clock **MPMCHCLKDELAY** is used to clock the output address, commands, and write data. This ensures that the commands meet the SDRAM setup and hold requirements.

Generally SDR-SDRAM memory devices require approximately 1.5ns of setup and hold time. This delay is indicated as t_{HD} (**HCLK** Delay).

MPMC output signals

The resulting signals provided to the pads before they go off chip are shown. The command, address, and write data are delayed by t_{HD}, and the output clock is not delayed. These signals now meet the SDRAM memory requirements.

Signals at SDRAM memory

The output signals (clock, address, and commands) are subjected to a delay t_{OD} (Output Delay) going to the SDRAM memory. t_{OD} is composed of the pad delay and the PCB delay.

Command capture at SDRAM memory

The SDRAM read command is captured on the positive edge of **MPMCCLKOUT** at the SDRAM memory.

For CAS 2 latency SDR-SDRAM reads, the read data is valid on the second positive edge of the memory clock after the read command has been captured. SDR-SDRAM provides valid read data a few nanoseconds before and after the SDRAM clock edge to make capturing the read data easier.

Feedback clock generation at SDRAM memory

The feedback clock is ideally generated from an output clock with a similar PCB and pad delay as the SDRAM clock, so that the feedback clocks positive edge is aligned in the middle of the time when the read data is valid.

Signals at MPMC

The input signals (read data and feedback clock) are subjected to a delay t_{ID} (Input Delay) from the SDRAM to the MPMC. t_{ID} is composed of a PCB delay and a pad delay.

Read data capture in the MPMC

The read data is captured using the feedback clock **MPMCFBCLKIN**. This ensures that the read data is captured cleanly. The read data is then captured in the **HCLK** clock domain.

10.7.6 Command delay plus zero SDRAM timing diagrams

This section shows the command delay plus zero SDR-SDRAM timing diagrams and data capture requirements.

——— **Note** ———

The DDR-SDRAM timing diagrams have not been shown. However, the timing relationships are very similar to SDR-SDRAM.

Figure 10-4 shows the command delay plus zero methodology where the read data is captured from the feedback clock to the **HCLK** domain using the positive edge of **HCLK**.

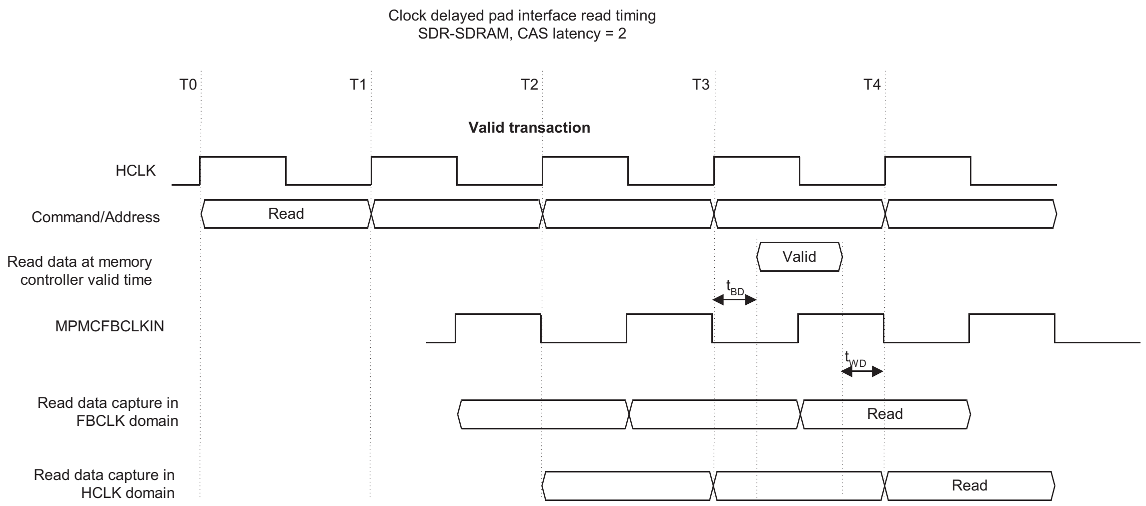

Clock delayed pad interface read timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
Valid transaction
HCLK
Command/Address
Read
Read data at memory controller valid time
Valid
tBD
MPMCFBCLKIN
tWD
Read data capture in FBCLK domain
Read
Read data capture in HCLK domain
Read

Figure 10-4 Command delay plus zero SDR-SDRAM, data captured on positive edge of HCLK

Data capture requirements, data captured on positive edge of HCLK

Both the worst case and best case dynamic skew, and read data delay must meet the following requirements otherwise the MPMC is not able to capture the read data correctly.

Maximum read data delay

The maximum read data delay is:

HCLK clock period - $t_{setup\mathbf{FBCLK}}$ - $t_{setup\mathbf{HCLK}}$

Alternatively this can be expressed as:($t_{OD} + t_{ID} + t_{setup\mathbf{FBCLK}} + t_{setup\mathbf{HCLK}}$) < **HCLK** clock periodWhere t_{OD} is the output delay and t_{ID} is the input delay.

Minimum read data delay

The minimum read data delay is $t_{hold\mathbf{HCLK}}$

Maximum dynamic skew

The minimum read delay and maximum read delay must not overlap the positive edge of **HCLK**. Therefore the maximum worst case dynamic skew is half a clock cycle.

Data capture requirements, data captured on negative edge of HCLK

This scheme is not useful.

10.7.7 Command delay plus one

This scheme is similar to the command delay plus zero pad interface methodology except that the data is captured on the next **HCLK** clock cycle. This provides an additional cycle for the read data to return from the SDRAM memory.

Figure 10-5 on page 10-36 shows a read transaction in detail.

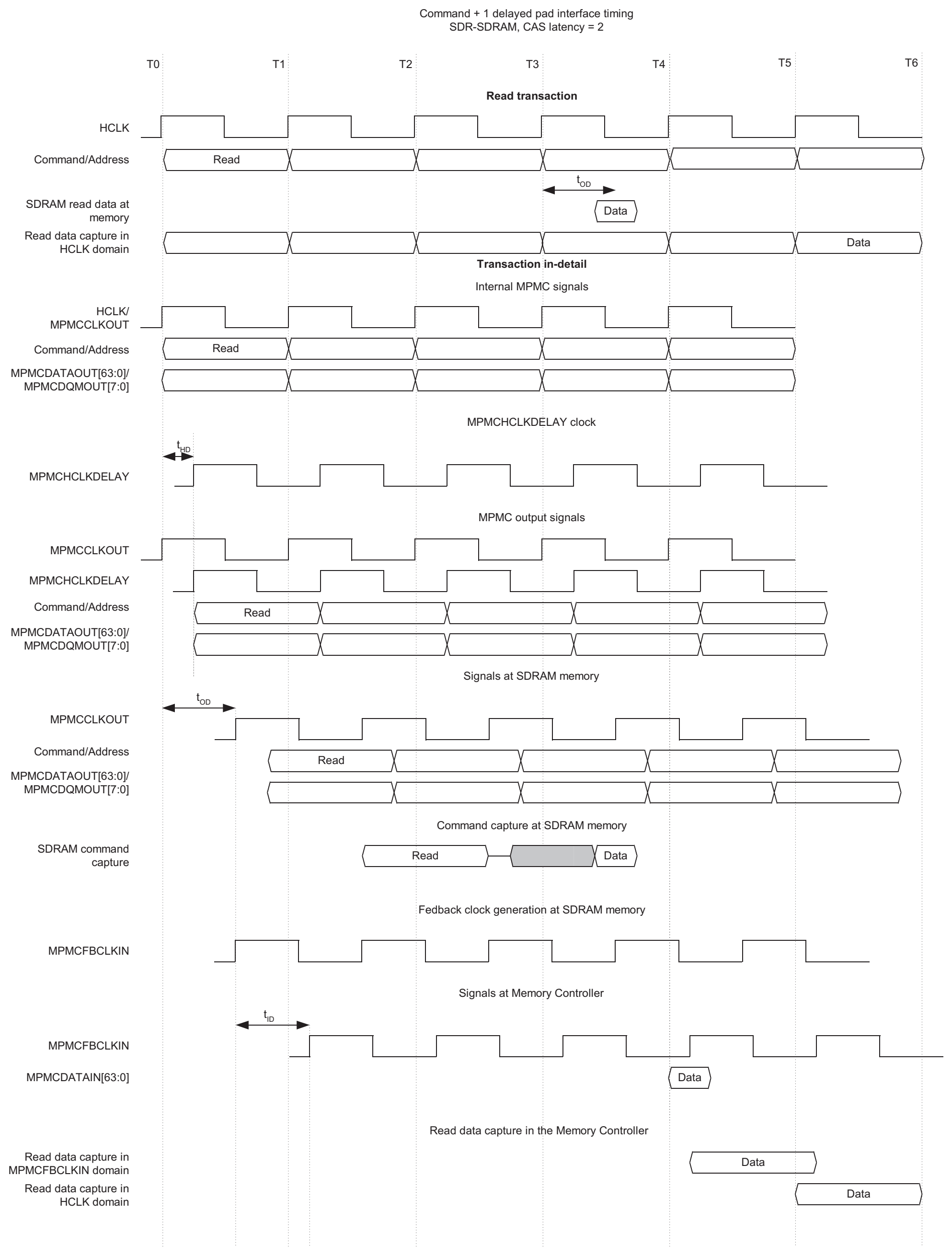

Command + 1 delayed pad interface timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
T6
Read transaction
HCLK
Command/Address
Read
tOD
SDRAM read data at memory
Data
Read data capture in HCLK domain
Data
Transaction in-detail
Internal MPMC signals
HCLK/ MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
MPMCHCLKDELAY clock
tHD
MPMCHCLKDELAY
MPMC output signals
MPMCCLKOUT
MPMCHCLKDELAY
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
Signals at SDRAM memory
tOD
MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
Command capture at SDRAM memory
SDRAM command capture
Read
Data
Fedback clock generation at SDRAM memory
MPMCFBCLKIN
Signals at Memory Controller
tID
MPMCFBCLKIN
MPMCDATAIN[63:0]
Data
Read data capture in the Memory Controller
Read data capture in MPMCFBCLKIN domain
Data
Read data capture in HCLK domain
Data

Figure 10-5 Command delayed plus one pad interface timing

10.7.8 Command delay plus one SDRAM timing diagrams

This section shows the command delay plus one SDR-SDRAM timing diagrams and data capture requirements.

Figure 10-6 shows the command delay plus one methodology where the read data is captured from the feedback clock to the **HCLK** domain using the negative edge of **HCLK**.

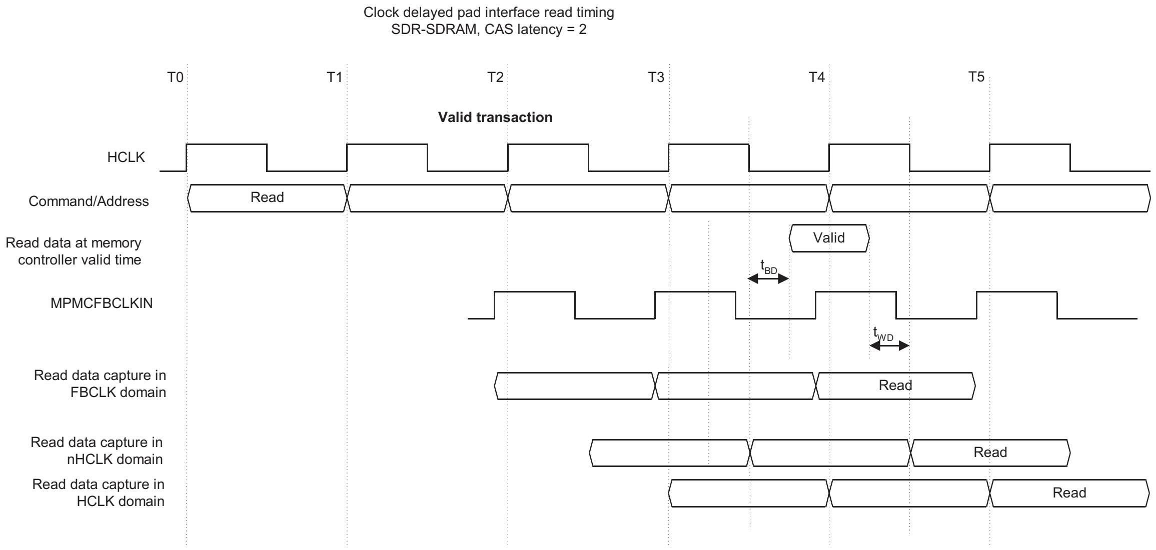

Clock delayed pad interface read timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
Valid transaction
HCLK
Command/Address
Read
Read data at memory controller valid time
Valid
t_{BD}
MPMCFBCLKIN
t_{WD}
Read data capture in FBCLK domain
Read
Read data capture in nHCLK domain
Read
Read data capture in HCLK domain
Read

Figure 10-6 Command delay plus one SDR-SDRAM, data captured on negative edge of HCLK

Data capture requirements, data captured on negative edge of HCLK

Both the worst case and best case dynamic skew, and read data delay must meet the following requirements otherwise the MPMC is not able to capture the read data correctly.

Maximum read data delay

Maximum read data delay is:

(one and a half **HCLK** clock period) - $t_{setup\mathbf{FBCLK}}$ - $t_{setup\mathbf{HCLK}}$

Minimum read data delay

Minimum read data delay is:

(half **HCLK** clock period) + $t_{hold\mathbf{HCLK}}$

Maximum dynamic skew

The minimum read delay and maximum read delay must not overlap the negative edge of **HCLK**. Therefore the maximum worst case dynamic skew is half a clock cycle.

Data capture requirements, data captured on positive edge of HCLK

Figure 10-7 shows the command delay plus one methodology where the read data is captured from the feedback clock to the **HCLK** domain using the positive edge of **HCLK**.

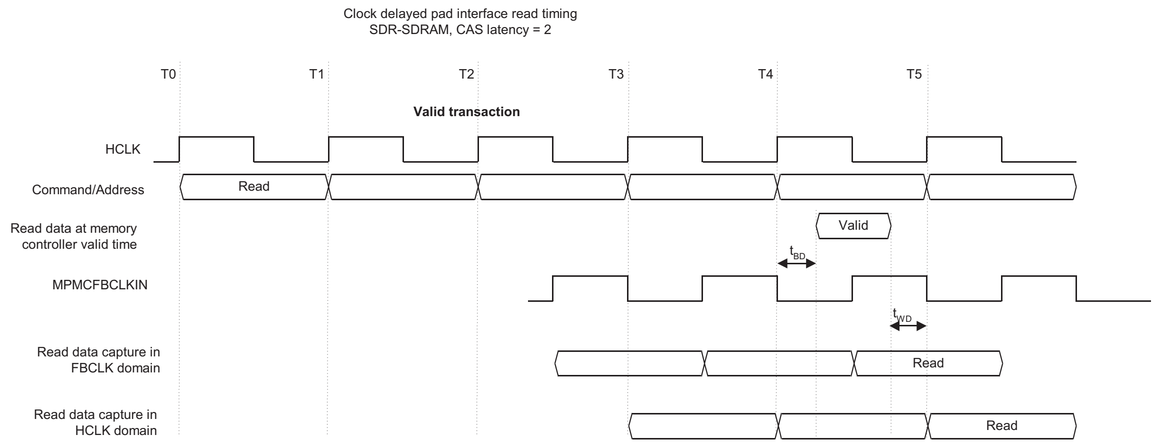

Clock delayed pad interface read timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
Valid transaction
HCLK
Command/Address
Read
Read data at memory controller valid time
Valid
t_{BD}
MPMCFBCLKIN
t_{WD}
Read data capture in FBCLK domain
Read
Read data capture in HCLK domain
Read

Figure 10-7 Command delay plus one SDR-SDRAM, data captured on positive edge of HCLK

The read data is captured on both the positive and negative edge of the data strobe. The second read data item is therefore returned to the MPMC half a clock cycle after the first data item. The read data delay is therefore half a cycle more compared to SDR-SDRAM.

Figure 10-8 on page 10-39 shows the command delay plus one methodology where the read data is captured from the feedback clock to the **HCLK** domain using the positive edge of **HCLK**.

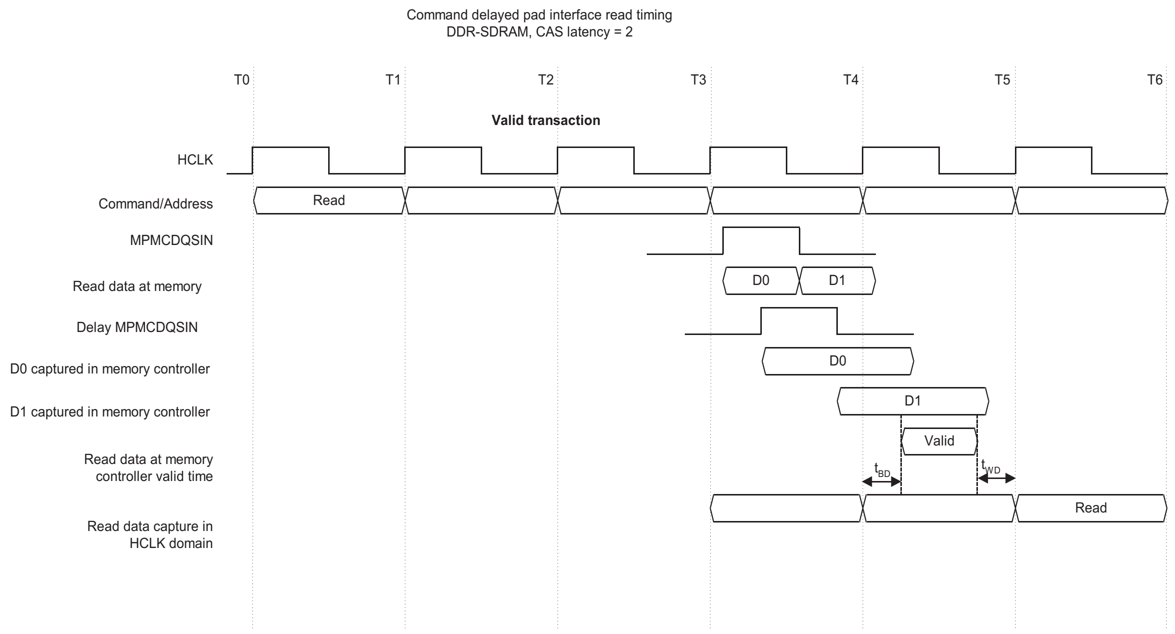

Command delayed pad interface read timing
DDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
T6
Valid transaction
HCLK
Command/Address
Read
MPMCDQSIN
Read data at memory
D0
D1
Delay MPMCDQSIN
D0 captured in memory controller
D0
D1 captured in memory controller
D1
Read data at memory controller valid time
Valid
t_{BD}
t_{WD}
Read data capture in HCLK domain
Read

Figure 10-8 Command delay plus one DDR-SDRAM, data captured on positive edge of HCLK

Both the worst case and best case dynamic skew, and read data delay must meet the following requirements otherwise the MPMC is not able to capture the read data correctly.

Maximum read data delay

The maximum read data delay is:

(two **HCLK** clock period) - $t_{setup\mathbf{FBCLK}}$ - $t_{setup\mathbf{HCLK}}$

Minimum read data delay

The minimum read data delay is:

(**HCLK** clock period) - $t_{hold\mathbf{HCLK}}$

Maximum dynamic skew

The minimum read delay and maximum read delay must not overlap the positive edge of **HCLK**. Therefore the maximum worst case dynamic skew is half a clock cycle.

10.7.9 Command delay plus two

This scheme is similar to the command delay plus zero pad interface methodology except that the data is captured two **HCLK** clock cycles later than the command plus zero scheme. This provides two additional cycles for the read data to return from the SDRAM memory. Figure 10-9 on page 10-41 shows a read transaction in detail.

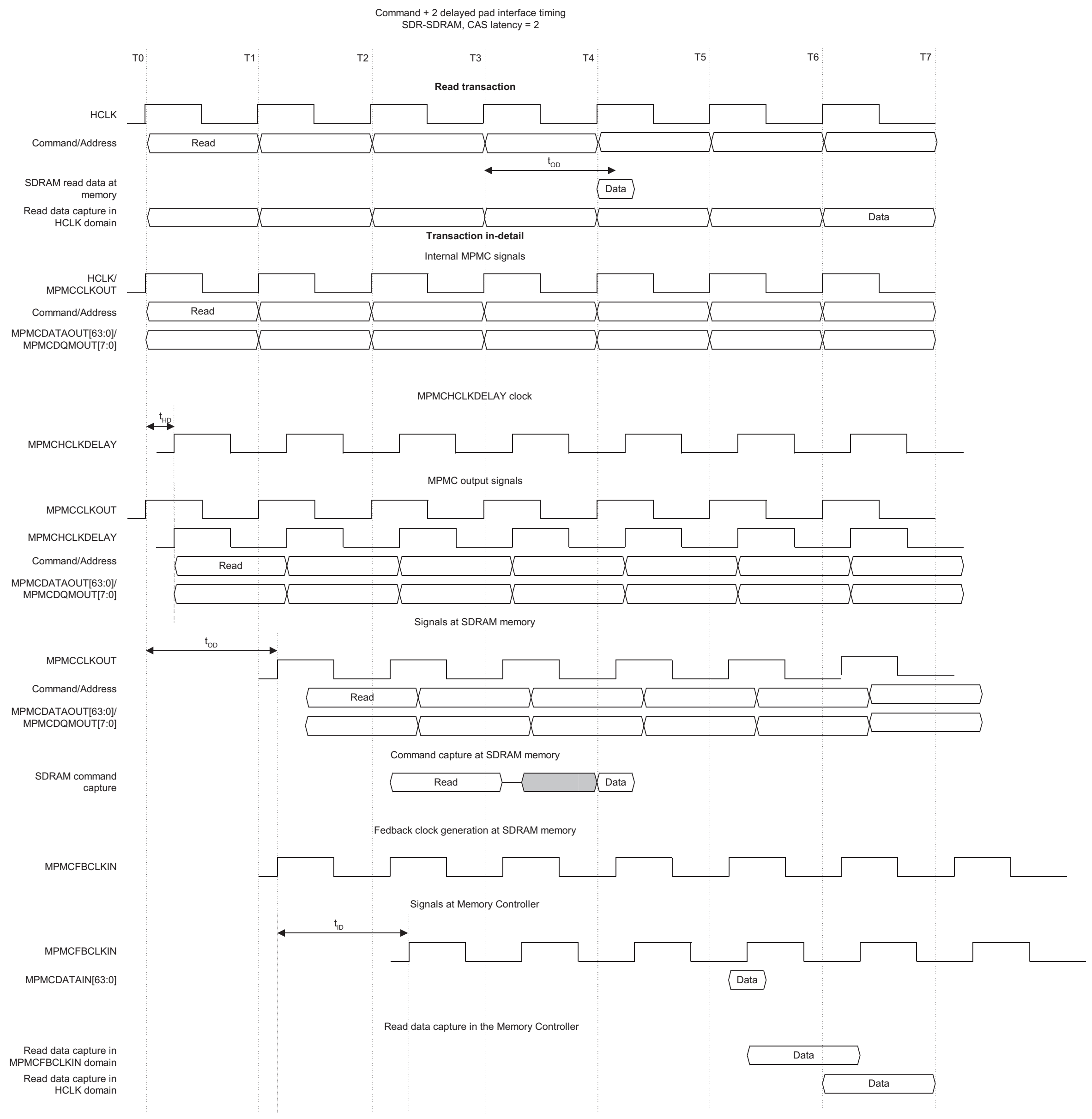

Command + 2 delayed pad interface timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
T6
T7
Read transaction
HCLK
Command/Address
Read
t_{OD}
SDRAM read data at memory
Data
Read data capture in HCLK domain
Data
Transaction in-detail
Internal MPMC signals
HCLK/ MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
MPMCHCLKDELAY clock
t_{HD}
MPMCHCLKDELAY
MPMC output signals
MPMCCLKOUT
MPMCHCLKDELAY
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
Signals at SDRAM memory
t_{OD}
MPMCCLKOUT
Command/Address
Read
MPMCDATAOUT[63:0]/ MPMCDQMOUT[7:0]
Command capture at SDRAM memory
SDRAM command capture
Read
Data
Fedback clock generation at SDRAM memory
MPMCFBCLKIN
Signals at Memory Controller
t_{ID}
MPMCFBCLKIN
MPMCDATAIN[63:0]
Data
Read data capture in the Memory Controller
Read data capture in MPMCFBCLKIN domain
Data
Read data capture in HCLK domain
Data

Figure 10-9 Command delayed plus two pad interface timing

10.7.10 Command delay plus two SDRAM timing diagrams

This section shows the command delay plus two SDR-SDRAM timing diagrams and data capture requirements.

Figure 10-10 shows the command delay plus two methodology where the read data is captured from the feedback clock to the **HCLK** domain using the negative edge of **HCLK**.

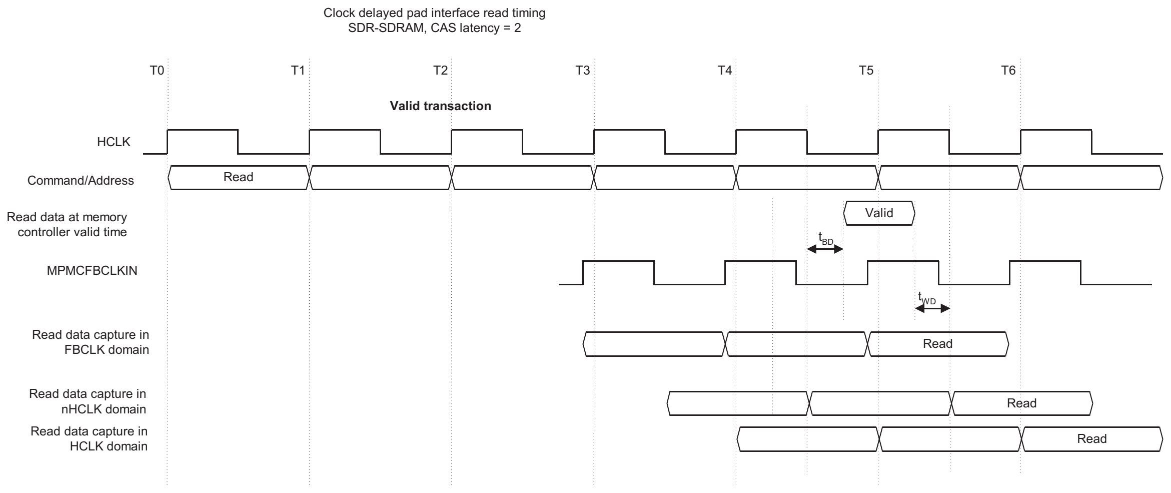

Clock delayed pad interface read timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
T6
Valid transaction
HCLK
Command/Address
Read
Read data at memory controller valid time
Valid
t_{BD}
MPMCFBCLKIN
t_{WD}
Read data capture in FBCLK domain
Read
Read data capture in nHCLK domain
Read
Read data capture in HCLK domain
Read

Figure 10-10 Command delay plus two SDR-SDRAM, data captured on negative edge of HCLK

Data capture requirements, data captured on negative edge of HCLK

Both the worst case and best case dynamic skew, and read data delay must meet the following requirements otherwise the MPMC is not able to capture the read data correctly.

Maximum read data delay

The maximum read data delay is:

(two and a half **HCLK** clock period) - $t_{setup\mathbf{FBCLK}}$ - $t_{setup\mathbf{HCLK}}$

Minimum read data delay

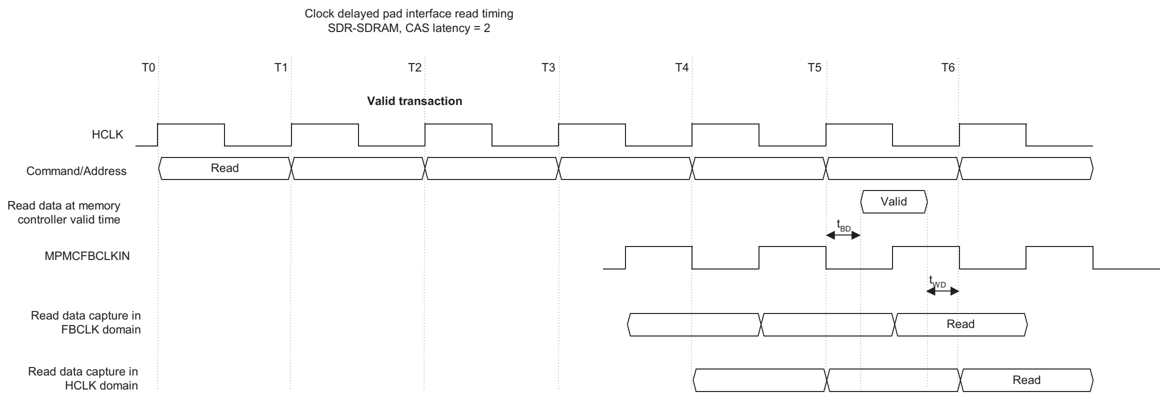
The minimum read data delay is: Minimum read data delay = (one and a half **HCLK** clock period) + $t_{hold\mathbf{HCLK}}$

Maximum dynamic skew

The minimum read delay and maximum read delay must not overlap the negative edge of **HCLK**. Therefore the maximum worst case dynamic skew is half a clock cycle.

Data capture requirements, data captured on positive edge of HCLK

Figure 10-11 shows the command delay plus two methodology where the read data is captured from the feedback clock to the **HCLK** domain using the positive edge of **HCLK**.



Clock delayed pad interface read timing
SDR-SDRAM, CAS latency = 2
T0
T1
T2
T3
T4
T5
T6
Valid transaction
HCLK
Command/Address
Read
Read data at memory controller valid time
Valid
tBD
MPMCFBCLKIN
tWD
Read data capture in FBCLK domain
Read
Read data capture in HCLK domain
Read

Figure 10-11 Command delay plus two SDR-SDRAM, data captured on positive edge of HCLK

Both the worst case and best case dynamic skew, and read data delay must meet the following requirements otherwise the MPMC is not able to capture the read data correctly.

Maximum read data delay

The maximum read data delay is:

(three **HCLK** clock period) - $t_{\text{setup}\mathbf{FBCLK}}$ - $t_{\text{setup}\mathbf{HCLK}}$

Minimum read data delay

The minimum read data delay is:

(two **HCLK** clock period) + $t_{\text{hold}\mathbf{HCLK}}$

Maximum dynamic skew

The minimum read delay and maximum read delay must not overlap the positive edge of **HCLK**. Therefore the maximum worst case dynamic skew is half a clock cycle.

10.8 Clock feedback in MPMC designs

There are three methods of clock feedback generation:

- *Generating feedback clock off-chip*
- *Generating feedback clock from bidirectional pad*
- *Generating feedback clock on-chip* on page 10-46.

10.8.1 Generating feedback clock off-chip

In this methodology the feedback clock is generated off-chip from the output clock. It is recommended that the feedback clock is generated near to the memory device as possible, and follows the data bus back on-chip.

This methodology takes into account the output pad loading, output pad delay, PCB trace, and input pad delay.

This methodology is the best way to generate the feedback clock, however this methodology requires additional input pins.

The output clock is fed back to the PrimeCell MPMC as shown in Figure 10-12.

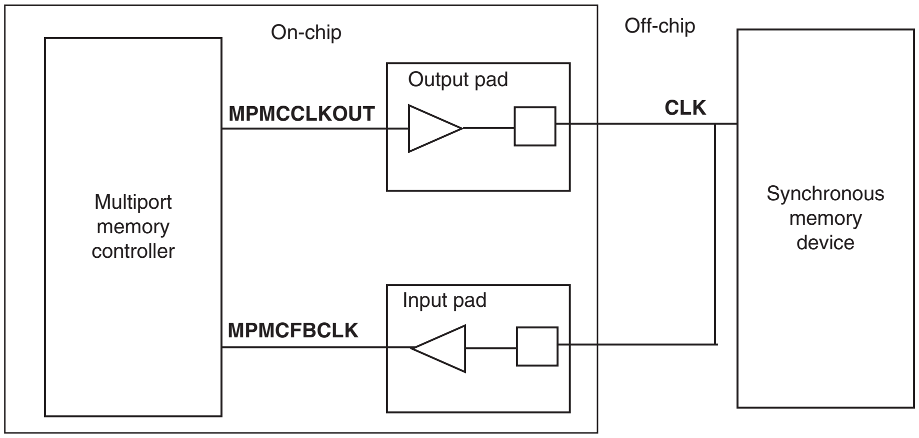


Figure 10-12 Generating feedback clock off-chip

10.8.2 Generating feedback clock from bidirectional pad

In this methodology, the feedback clock is generated on-chip, using the input of the tristate pad that outputs the memory clock.

This methodology takes into account the output pad loading and output pad delay. It is normally the best way to generate the feedback clock without generating the feedback clock externally off-chip.

The output clock is feed back from a bidirectional pad as shown in Figure 10-13.

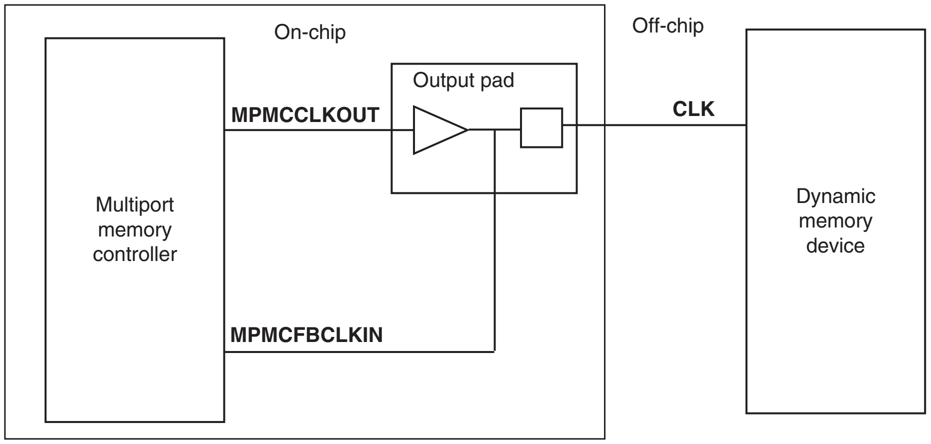


Figure 10-13 Generating feedback clock from bidirectional pad

10.8.3 Generating feedback clock on-chip

In this methodology, the feedback clock is generated on-chip by, for example, inverting the output clock.

This methodology does not take into account the loading of the output pad. It is the least accurate way to generate the feedback clock.

The output clock is inverted to generate the feedback clock as shown in Figure 10-14 on page 10-47.

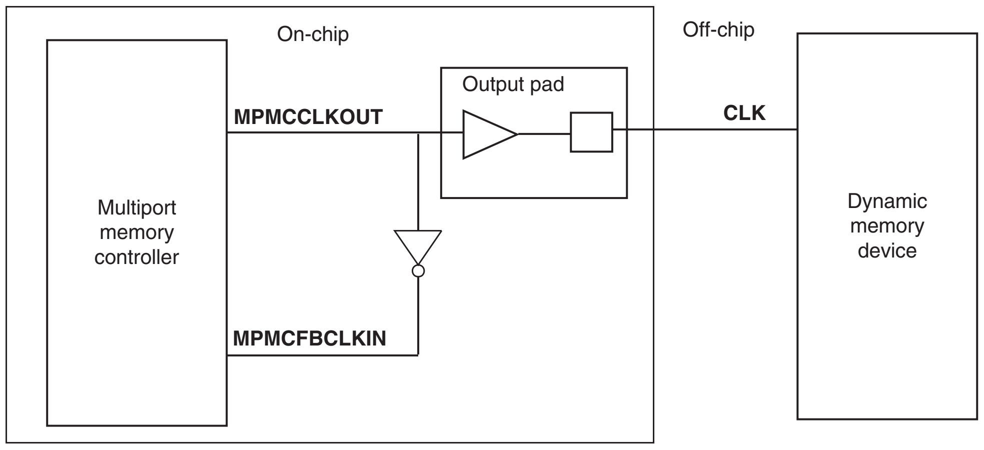


Figure 10-14 Generating feedback clock on-chip

10.9 Memory clock and feedback clock strategy

The PrimeCell MPMC has four output clocks, four data strobes, and eight feedback clocks.

The output clocks are used to clock the external dynamic memory devices. The data strobes are used to register the DDR-SDRAM read data. The feedback clocks are used to register the read data from single data rate SDRAM memory devices.

Depending on the system architecture and system performance the output clocks, data strobes, and feedback clocks can be used in several different ways.

The memory clock and feedback clock strategy is described in the following sections:

- *Output clocks*
- *Data strobes for DDR-SDRAM*
- *Feedback clocks for SDR-SDRAM* on page 10-49
- *Signal termination* on page 10-50
- *High performance systems* on page 10-51
- *Medium performance systems* on page 10-60
- *Lower performance systems* on page 10-61.

10.9.1 Output clocks

Providing four output clocks, **MPMCCLKOUT[3:0]**, enables the load to the clock signals to be reduced. For example, by connecting each output clock to a different memory device.

For systems where performance is less of an issue, or where there are a small number of dynamic memory devices, fewer output clocks can be bonded out.

10.9.2 Data strobes for DDR-SDRAM

DDR-SDRAM memory devices use data strobe signals. These signals are used to indicate when the read and write data must be registered. For each eight bits of data there is an associated data strobe signal. The PrimeCell MPMC has eight data strobe signals:

- **MPMCDQSIN[0]** is associated with **MPMCDATA[7:0]** on the positive edge of the DDR-SDRAM strobe
- **nMPMCDQSIN[0]** is associated with **MPMCDATA[7:0]** on the negative edge of the DDR-SDRAM strobe
- **MPMCDQSIN[1]** is associated with **MPMCDATA[15:8]** on the positive edge of the DDR-SDRAM strobe

- **nMPMCDQSIN[1]** is associated with **MPMCDATA[15:8]** on the negative edge of the DDR-SDRAM strobe
- **MPMCDQSIN[2]** is associated with **MPMCDATA[23:16]** on the positive edge of the DDR-SDRAM strobe
- **nMPMCDQSIN[2]** is associated with **MPMCDATA[23:16]** on the negative edge of the DDR-SDRAM strobe
- **MPMCDQSIN[3]** is associated with **MPMCDATA[31:24]** on the positive edge of the DDR-SDRAM strobe
- **nMPMCDQSIN[3]** is associated with **MPMCDATA[31:24]** on the negative edge of the DDR-SDRAM strobe.

For writes to memory the data strobe is an output, (generated by the MPMC). The data strobe edge transitions in the middle of the write data period. The data strobe is used by the DDR-SDRAM memory to capture the write data.

For reads from memory the data strobe is an input, (generated by the DDR-SDRAM memory). The data strobe transitions at the same time as the read data changes. The data strobe signals are normally delayed on-chip so that they can be used to capture the read data.

The data strobe signals must follow the same path as the data they are associated with so that they are delayed by the same amount.

The ideal situation is where:

- the data strobes follow a similar path on the PCB as the associated data signals
- the loading on the data bus and the data-strobes is similar.

——— Note ———

32-bit DDR-SDRAM devices have a single DQS connection, rather than one for each byte lane. This is because they are expected to be used in systems where the difference in delays between byte lanes is small. The single DQS on the memory device must be connected to all four DQS ports of the MPMC.

It is not recommended that 32-bit DDR-SDRAM devices are mixed with 8-bit or 16-bit DDR-SDRAM devices, because of the differences in DQS signals between the devices.

10.9.3 Feedback clocks for SDR-SDRAM

There are eight feedback clocks. Each feedback clock is associated with eight bits of the data bus:

- **MPMCFBCLKIN[0]** registers bits **MPMCDATAIN[7:0]**

- **MPMCFBCLKIN[1]** registers bits **MPMCDATAIN[15:8]**
- **MPMCFBCLKIN[2]** registers bits **MPMCDATAIN[23:16]**
- **MPMCFBCLKIN[3]** registers bits **MPMCDATAIN[31:24]**
- **MPMCFBCLKIN[4]** registers bits **MPMCDATAIN[39:32]**
- **MPMCFBCLKIN[5]** registers bits **MPMCDATAIN[47:40]**
- **MPMCFBCLKIN[6]** registers bits **MPMCDATAIN[55:48]**
- **MPMCFBCLKIN[7]** registers bits **MPMCDATAIN[63:56]**.

It is recommended that the feedback clock signals are generated off-chip from an output clock near the furthest memory device from the ASIC. The feedback clock signals must then follow the same path as the read data associated with them so that they are delayed by the same amount. This ensures that the read data is registered correctly.

The ideal situation is where:

- the feedback clock follows a similar path on the PCB as the associated data signals
- the loading on the data bus and the feedback clock is similar.

For high-performance systems the feedback clocks are generated as described above, from an output clock near the furthest memory device from the ASIC. The feedback clock must then be routed with the data it is associated with on the PCB to match the path read data. Higher performance systems can use four separate feedback clock signals so that each byte of the data bus is matched individually.

Medium performance systems can use fewer feedback clocks. These feedback clocks can be connected to the **MPMCFBCLKIN[7:0]** on-chip signals as necessary.

For systems where performance is less of an issue, or where there are a small number of dynamic memory devices, the feedback clock can be generated on chip by delaying the output clock.

10.9.4 Signal termination

The off-chip signals must be terminated appropriately.

Series termination is used in lower performance systems. The signal can normally only be tapped at the end of the wire and cannot be connected to form a loop.

Parallel termination is used in higher performance systems. The signal can be tapped at various points and can be connected to form a loop (for example **MPMCCLKOUT** to **MPMCFBCLKIN**).

Termination is beyond the scope of this document. However, care must be taken at high clock speeds to terminate the signals appropriately. Additionally, signal drive capabilities, voltage swings, and circuit board signal characteristics must all be taken into account.

10.9.5 High performance systems

This subsection describes the following devices:

- *x8 DDR-SDRAM memory devices*
- *Mixed width DDR-SDRAM memory devices* on page 10-52
- *x8 SDRAM memory devices* on page 10-53
- *x16 SDRAM memory devices* on page 10-55
- *x32 SDRAM memory devices* on page 10-56
- *Mixed width SDRAM memory devices* on page 10-57
- *SRAM and SDRAM memory devices* on page 10-58
- *DDR-SDRAM and SDRAM systems* on page 10-60.

x8 DDR-SDRAM memory devices

Figure 10-15 on page 10-52 shows an example of a way to connect x8 DDR-SDRAM devices. Chip select 4 and chip select 5 both have two x8 DDR-SDRAM devices providing a 16-bit data bus.

MPMCCLKOUT[0] is used to clock the memory devices providing the lower eight bits of data. **MPMCCLKOUT[1]** is used to clock the memory devices providing the upper eight bits of data. **MPMCDQS[0]** is generated from the memory devices providing the lower eight bits of data. **MPMCDQS[1]** is generated from the memory devices providing the upper eight bits of data.

This methodology reduces the load on the output clocks and enables the read data to be registered accurately.

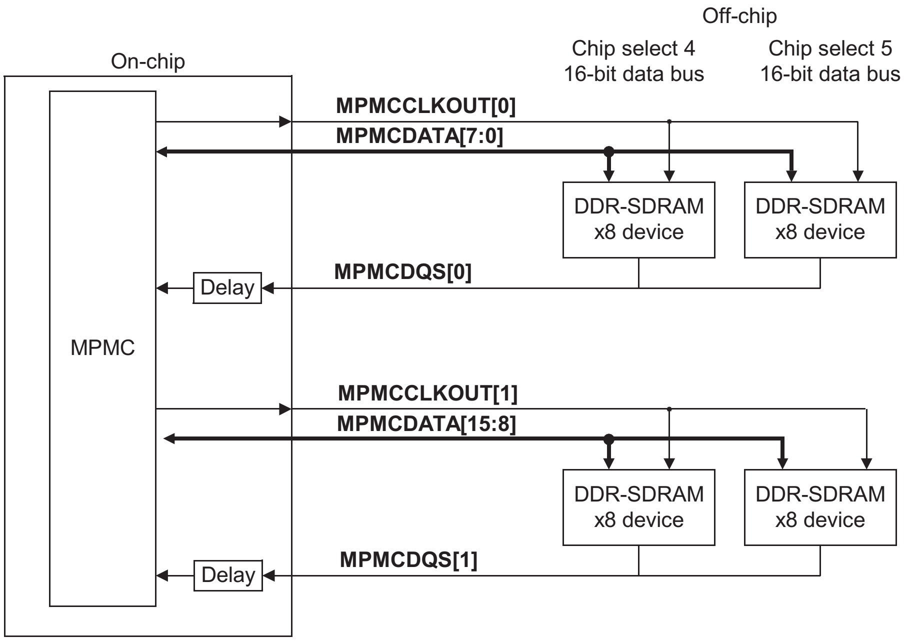


Figure 10-15 x8 DDR-SDRAM device

Mixed width DDR-SDRAM memory devices

Figure 10-16 on page 10-53 shows an example of a way to connect mixed width memory devices. Chip select 4 has two x8 DDR-SDRAM devices providing a 16-bit data bus. Chip select 5 has one x16 DDR-SDRAM device also providing a 16-bit data bus:

- **MPMCCLKOUT[0]** is used to clock chip select 4 lower eight-bit memory part
- **MPMCCLKOUT[1]** is used to clock the memory device of chip select 5
- **MPMCCLKOUT[2]** is used to clock the high order memory part of chip select 4.

MPMCDQS[0] is generated by the memory part of chip select 5, and the lower memory part of chip select 4. **MPMCDQS[1]** is generated by the memory part of chip select 5, and the upper memory part of chip select 4.

This spreads the load on the output clocks and ensures that the feedback clocks delay matches the read data delay.

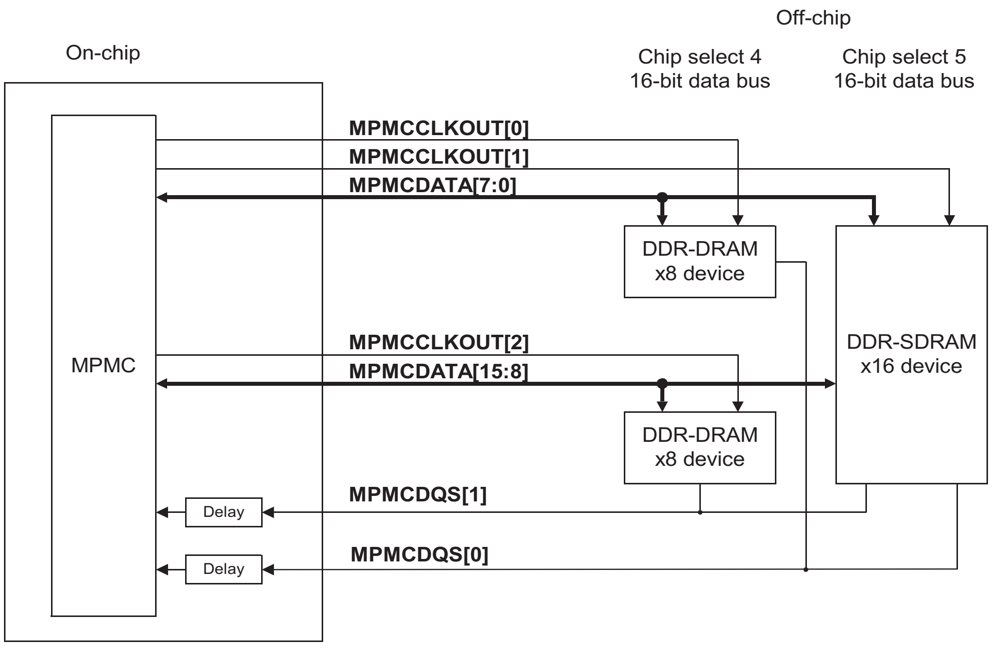


Figure 10-16 Mixed width DDR-SDRAM memory devices

x8 SDRAM memory devices

Figure 10-17 on page 10-54 shows an example of a way to connect x8 SDRAM devices. Chip select 4 has four x8 SDRAM devices providing a 32-bit data bus. Chip select 5 has two x8 SDRAM devices providing a 16-bit data bus.

MPMCCLKOUT[0] is used to clock the memory devices providing the low eight bits of data. **MPMCFBCLKIN[0]** is generated from **MPMCCLKOUT[0]**. The memory devices that provide the next eight bits of data are clocked by **MPMCCLKOUT[1]**. The read data is registered by **MPMCFBCLKIN[1]**. The same scheme is used for the other bytes of data.

This methodology reduces the load on the output clocks and enables the read data to be registered accurately.

Figure 10-17 on page 10-54 shows parallel termination on the feedback clocks.

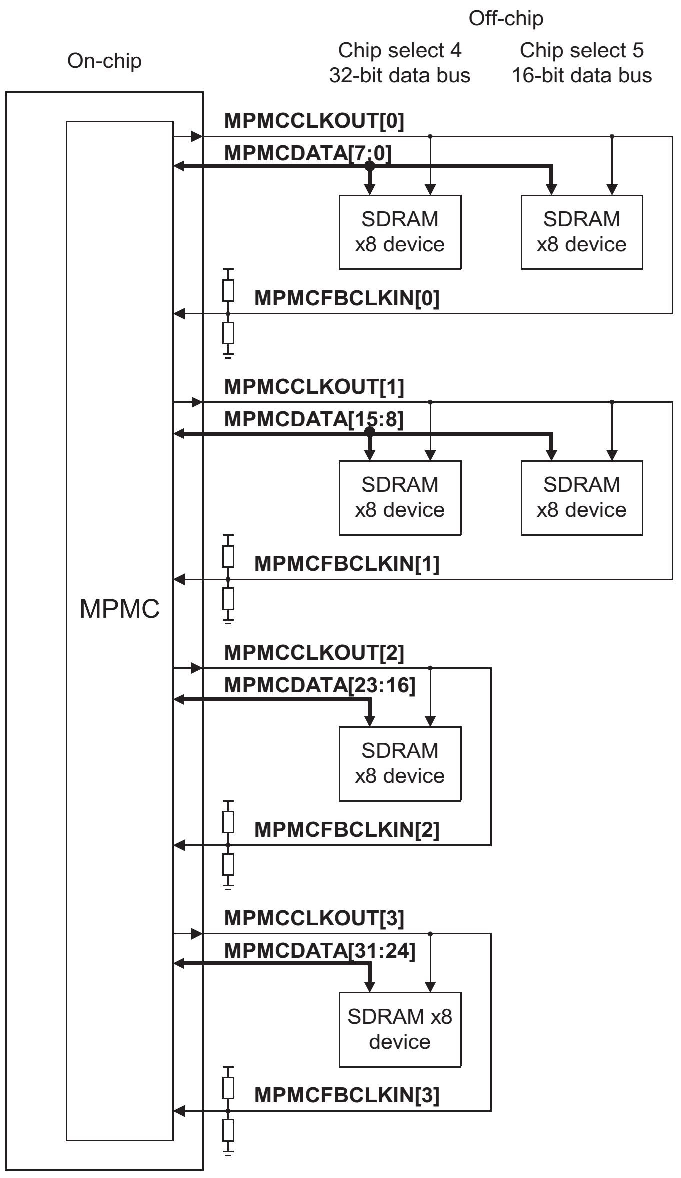


Figure 10-17 x8 SDRAM device

x16 SDRAM memory devices

Figure 10-18 on page 10-56 shows an example of a way to connect x16 SDRAM devices. Chip select 4 has two x16 SDRAM devices providing a 32-bit data bus. Chip select 5 has one x16 SDRAM device providing a 16-bit data bus.

MPMCCLKOUT[0] is used to clock the memory devices providing the low 16 bits of data. **MPMCFBCLKIN[0]** and **MPMCFBCLKIN[1]** are generated from **MPMCCLKOUT[1]**.

MPMCCLKOUT[2] is used to clock the memory devices providing the upper 16 bits of data. **MPMCFBCLKIN[2]** and **MPMCFBCLKIN[3]** are generated from **MPMCCLKOUT[3]**.

——— **Note** ———

MPMCFBCLKIN[3:0] can be connected together on-chip as required.

————————

Figure 10-18 on page 10-56 shows series termination on the output clocks used to generate the feedback clocks.

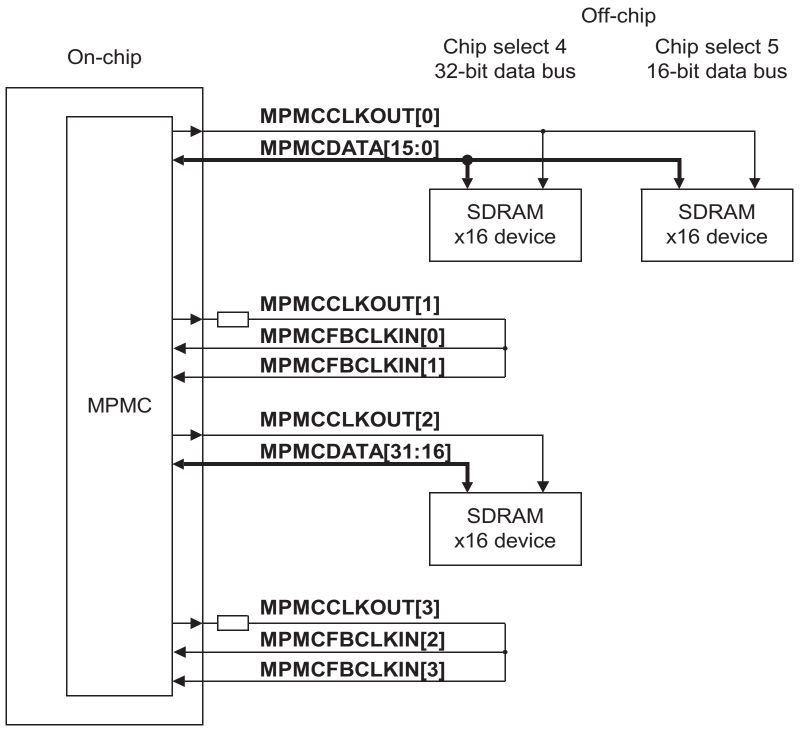


Figure 10-18 x16 SDRAM connection

x32 SDRAM memory devices

Figure 10-19 on page 10-57 shows an example of a way to connect x32 SDRAM devices. Chip select 4 and chip select 5 each have one x32 SDRAM device providing a 32-bit data bus.

MPMCCLKOUT[0] is used to clock the two memory devices. Two feedback clocks are generated from **MPMCCLKOUT[1]**, and the other two feedback clocks are generated from **MPMCCLKOUT[2]**.

——— **Note** ———

MPMCFBCLKIN[3:0] can be connected together on-chip as required.

———————

Figure 10-19 on page 10-57 shows series termination on the output clocks used to generate the feedback clocks.

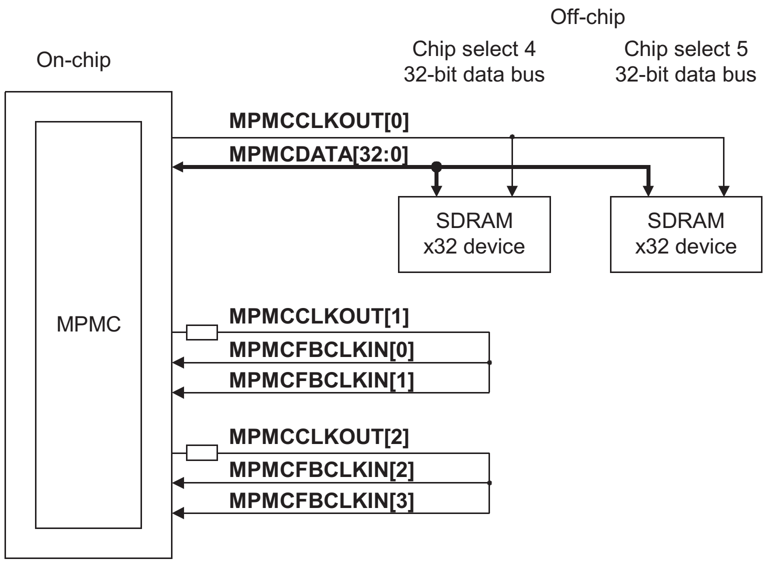


Figure 10-19 x32 SDRAM interconnection

Mixed width SDRAM memory devices

Figure 10-20 on page 10-58 shows an example of a way to connect mixed width memory devices. Chip select 4 has two x8 SDRAM devices providing a 16-bit data bus. Chip select 5 has one x16 SDRAM device also providing a 16-bit data bus:

- **MPMCCLKOUT[0]** is used to clock chip select the 4 lower eight-bit memory parts
- **MPMCCLKOUT[1]** is used to clock the memory device of chip select 5
- **MPMCCLKOUT[2]** is used to clock the high order memory part of chip select 4
- **MPMCFBCLKIN[0]** and **MPMCFBCLKIN[1]** are generated from **MPMCCLKOUT[3]**.

This spreads the load on the output clocks and ensures that the feedback clocks delay matches the read data delay.

Figure 10-20 on page 10-58 shows series termination on the output clocks used to generate the feedback clocks.

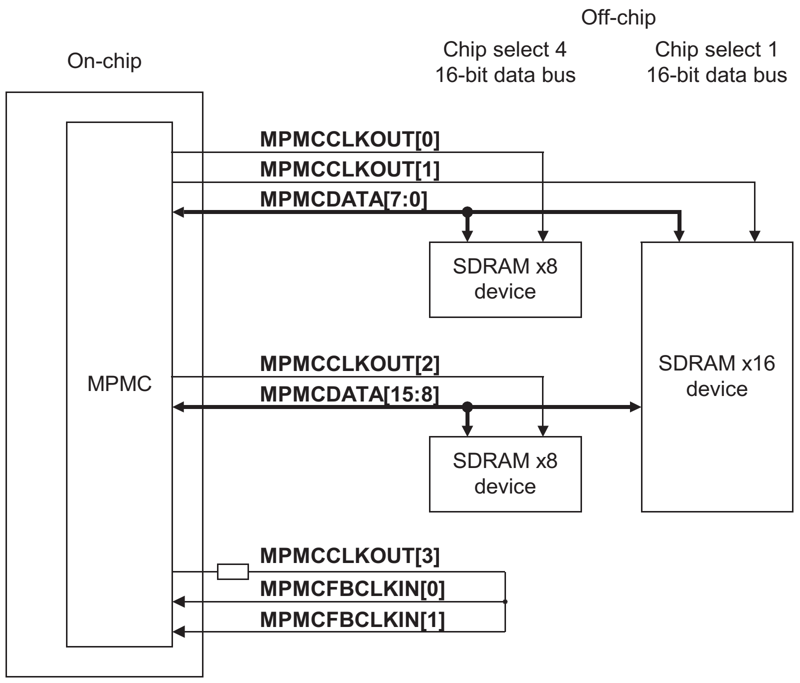


Figure 10-20 Mixed width SDRAM memory devices

SRAM and SDRAM memory devices

Figure 10-21 on page 10-59 shows an example of a way to connect SDRAM and SRAM memory devices. Chip select 4 and chip select 1 each have four x8 SDRAM devices providing a 32-bit data bus.

MPMCCLKOUT[0] is used to clock the memory devices providing the low eight bits of data. **MPMCFBCLKIN[0]** is generated from **MPMCCLKOUT[0]**. The memory devices that provide the next eight bits of data are clocked by **MPMCCLKOUT[1]**. The read data is registered by **MPMCFBCLKIN[1**]. The same scheme is used for the other bytes of data.

This methodology reduces the load on the output clocks and enables the read data to be accurately registered by the feedback clocks.

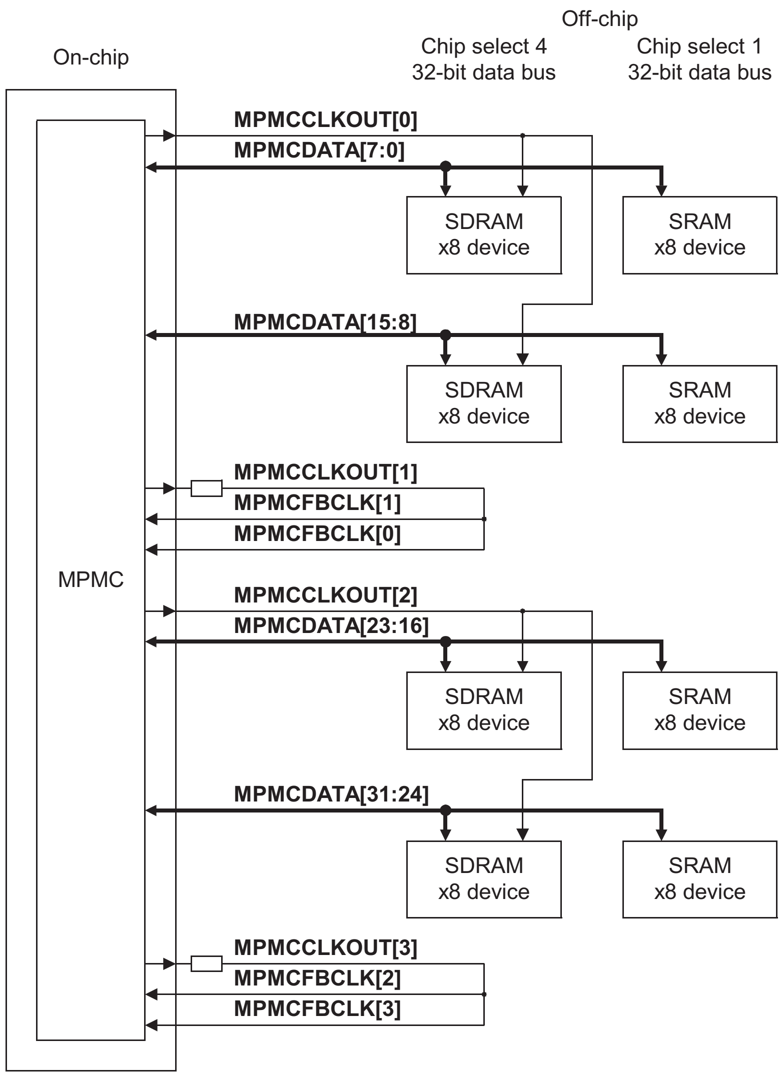


Figure 10-21 SRAM and SDRAM memory devices

Figure 10-21 shows series termination on the output clocks used to generate the feedback clocks.

DDR-SDRAM and SDRAM systems

DDR uses SSTL_2 and SDRAM memories use LVTTL. It is therefore difficult to share pins between the DDR-SDRAM and SDRAM interface. The ability to do this is pad-dependent and requires a pad specific to your requirements. While it is envisaged that both SDRAM and DDR memories are controlled by one controller, consideration must be made as to the following:

- the requirements of the memory devices use
- the characteristics of the I/O cells that are used.

If the memory devices are chosen so that they can all be driven by the same I/O cell, they are able to save pins. However, it is the responsibility of the ASIC designer and system designer to ensure that the external devices and I/O cells are all compatible.

10.9.6 Medium performance systems

For medium performance systems, or for systems where the number of pads available is limited a single output clock, data strobe, and feedback clock can be used.

MPMCCLKOUT is used to clock all the memory devices. The single feedback clock **MPMCFBCLKIN** generates the eight **MPMCFBCLKIN[7:0]** signals internally on chip.

Figure 10-22 on page 10-61 shows parallel termination on the feedback clocks.

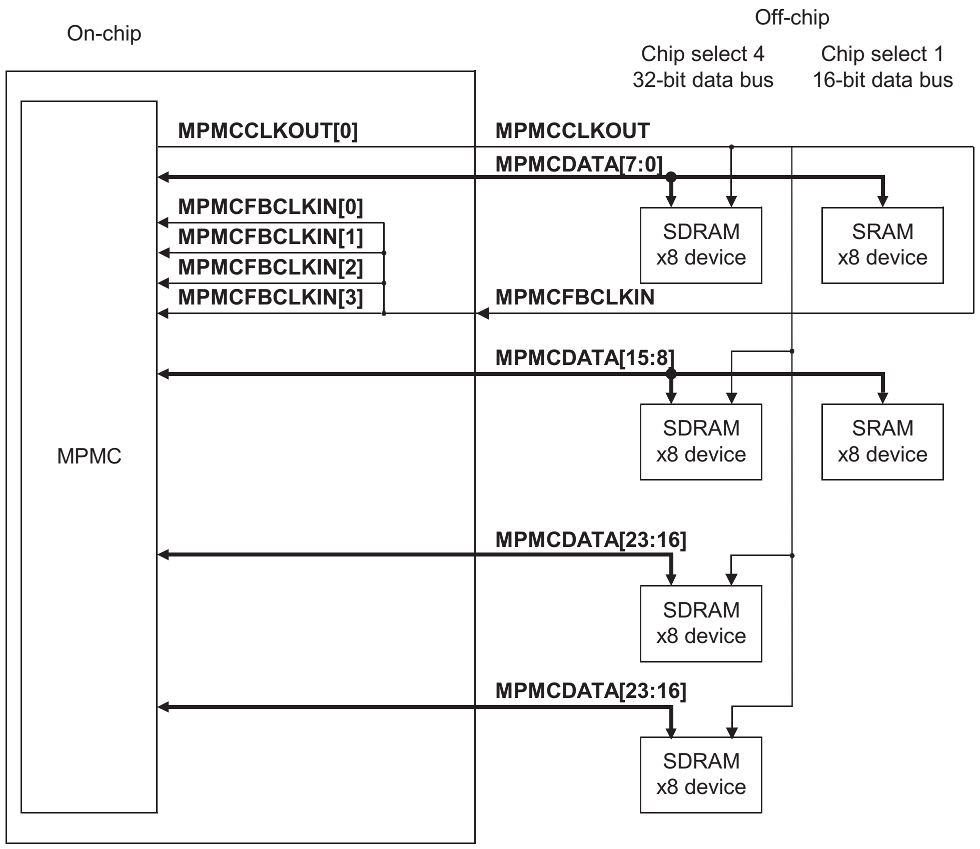


Figure 10-22 Medium performance systems

10.9.7 Lower performance systems

For lower performance systems, or for systems where the number of pads available is limited, a single output clock can be used. The feedback clock can be generated on-chip by delaying the output clock.

The data strobe and the feedback clock delay can be generated by several delay elements, or by using a more complex temperature and voltage compensated delay block, for example a DLL.

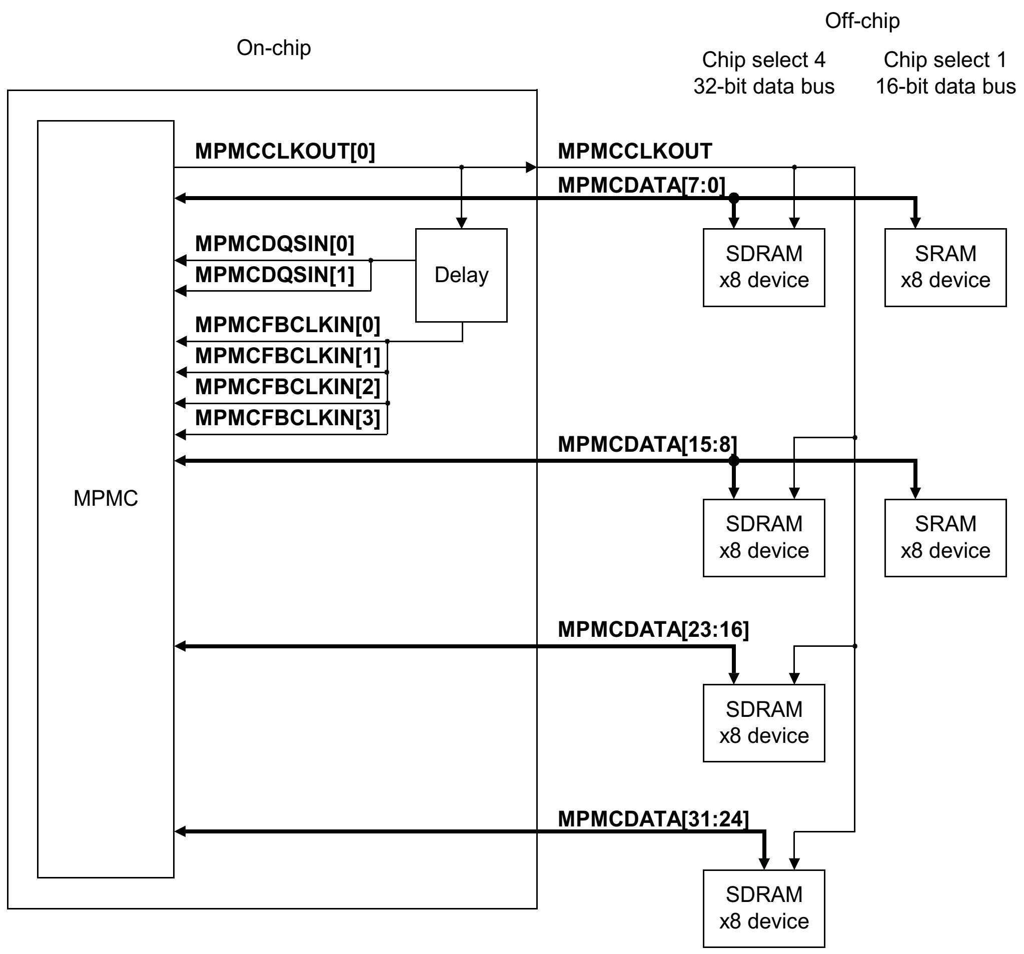


Figure 10-23 Lower performance systems

10.10 NAND flash connection methodology

Figure 10-24 shows the typical connections in a system which uses SRAM, SDRAM, and NAND flash.

The common data bus is connected to all devices, and the three separate write enable outputs are connected to the relevant devices.

The NAND flash device has an additional ready status output which is connected to the **MPMCNDREADYIN** input of the MPMC. To operate correctly, the output from the NAND flash device must be tied HIGH using a suitable value pull-up resistor. The value of this resistor can be calculated according to the equations in the device data sheet.

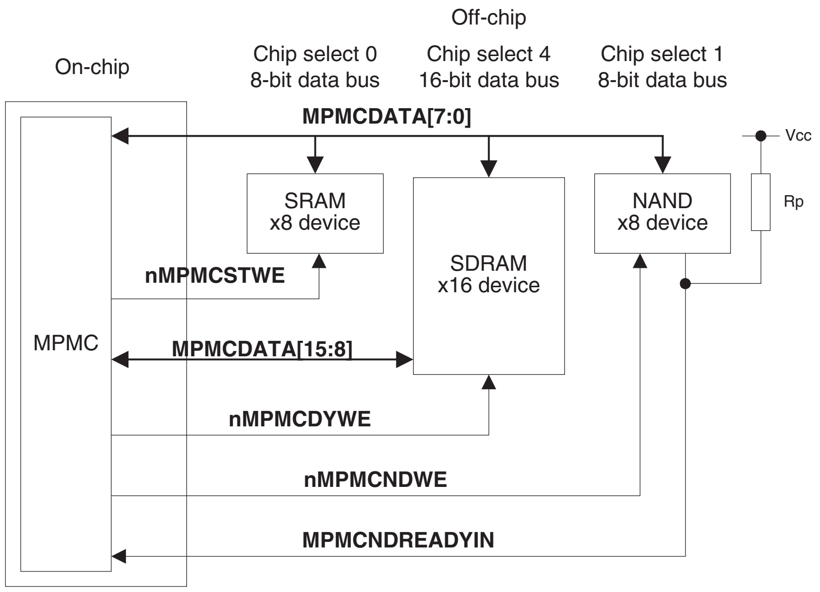


Figure 10-24 NAND flash connection with SRAM and SDRAM

Chapter 11
Power Strategy

This chapter describes the power strategy aspects relating to the ARM PrimeCell MPMC (PL176). It contains the following section:

- *Factors affecting power consumption* on page 11-2.

11.1 Factors affecting power consumption

This section describes the factors that affect the power consumption of the system. It also describes ways to reduce power consumption.

The following factors affect system power consumption:

- *ASIC silicon process and cell library*
- *ASIC design decisions*
- *Memory device selection* on page 11-5.

11.1.1 ASIC silicon process and cell library

The silicon process and cell library used by the ASIC has a direct impact on the maximum frequency of the design and the amount of power consumed by the ASIC.

Power is consumed both when there is no activity in the design, static power consumption, and while the design is active, dynamic power consumption.

Note

Generally a process optimized for low-power consumption does not operate as fast as nonpower optimized processes.

Static power consumption (static leakage current)

The static power consumption is because of leakage current in the process. Static power consumption is proportional to:

- the size of the design
- the silicon process.

Dynamic power consumption (gate transitioning)

Dynamic power consumption is because of gate transitioning.

Dynamic power consumption is proportional to:

- the voltage of the design
- the frequency of operation
- the size of the design
- the silicon process.

11.1.2 ASIC design decisions

This section describes ways to reduce the power consumption of the ASIC design.

Clock gating

Clock gating can be performed at several levels:

- high-level clock gating
- module-level (architectural) clock gating
- low-level clock gating.

Many systems use all three approaches to reduce power consumption.

High-level clock gating High-level clock gating involves gating the clock before it reaches the block. This type of clock gating is normally performed in the PMU or in the clock generator

High-level clock gating provides the coarsest form of clock control. When the clock is gated the block is disabled, and there is no power consumed in the clock tree or the blocks to which the clock is connected.

Module-level (architectural) clock gating Architectural clock gating involves gating the clock automatically to modules in a block when the clock is not required. This type of clock gating is performed internally to a block.

Architectural clock gating provides a more fine-grained form of clock control. When the clock is gated only the specific unused module is disabled in the block.

Low-level clock gating Low-level clock gating involves grouping several d-type flip-flops together and only enabling a clock to them when required. This type of clock gating is normally performed using EDA tools, for example Synopsys Power Compiler.

Low-level clock gating provides the most fine-grained form of clock control. The clock enabling and disabling is performed automatically. When the clock is gated the d-types in the group are disabled and the rest of the design functions as normal.

Power management controller

Power management controllers normally contain the following functionality:

- Disable and enable clocks in the design when necessary.
- To provide lower frequency clocks when the system is not required to operate at the maximum frequency.
- To put devices in low-power modes when they are not required, for example by placing SDRAM into self-refresh mode.

Providing a large number of clock outputs from the PMU enables finer grained control over which blocks are enabled or disabled at the expense of increasing design complexity of the PMU.

Improving system efficiency

System designs that perform tasks more efficiently, generally have lower power consumption.

Designs that make use of memory more efficiently (by reducing the average memory latency) complete a given task in fewer clock cycles and reduce power consumption. This can be performed in several ways:

- by increasing processor cache size
- the addition of on chip memory
- by using blocks that access the external memory more efficiently.

——— **Note** ———

Memory transactions on-chip are much more power efficient than off-chip transactions.

————————

Cache and cache size Using a cached processor, or increasing the cache size reduces the number of external memory transactions and reduces the average transfer latency. This means a given task is completed in fewer clock cycles, and the power consumption is reduced. Additionally, memory transactions to on-chip memory consume less power than off-chip memory transactions.

However, having a large cache increases the caches static power consumption, and increases the amount of power consumed for each cache transaction because of additional cache loading. Normally this increase in power consumption is offset by the power saved elsewhere in the system.

Additional cache levels Additional cache levels can reduce the number of external memory transactions and reduce the average transfer latency for both processors and peripherals (assuming the cache can be shared). This means a given task is completed in fewer clock cycles, and the power consumption is reduced. Additionally, memory transactions to on-chip memory consume less power than off-chip memory transactions.

However, having additional levels of cache increases the static power consumption of the ASIC. Dynamic power consumption is not as high as having a single large level 1 cache, because the cache loading is smaller for the majority of the transactions. Normally the increase in power consumption is offset by the power saved elsewhere in the system.

On-chip memory On-chip memory reduces the number external memory transactions and reduces the average transfer latency for both processors and peripherals (assuming the cache can be shared). This means a given task is completed in fewer clock cycles, and the power consumption is reduced. Additionally, memory transactions to on-chip memory consume less power than off-chip memory transactions.

However, on-chip memory increases the static power consumption of the ASIC. Normally the increase in power consumption is offset by the power saved elsewhere in the system.

Efficient transaction patterns Blocks that perform efficient external memory access patterns (transactions that take into account the external memories characteristics) have lower average transfer latency, complete the task in fewer clock cycles, and reduce power consumption.

Memory interface Choose appropriate pads to drive the external signals. Using pads with larger drive strength than required wastes power unnecessarily. However, using pads with too small a drive strength might mean that the system might not be able to be clocked at the required frequency.

11.1.3 Memory device selection

The exact type and number of memory devices used impacts system power consumption.

Number of memory devices

Having a large number of external memory devices connected to the MPMC increases the external load that the pads have to drive. Therefore higher drive strength pads are required and this increases power consumption. Additionally, a system with a very large external load might mean that the memory interface is unable to be clocked at the frequency required.

Memory device DC characteristics

Any memory devices that you choose have specific DC (voltage) requirements. These requirements must be similar otherwise they are not able to operate on the same memory interface.

Note

Lower voltage memory devices generally consume less power, and are slower than higher voltage parts.

SDRAM memory device characteristics SDRAM memory has several features to reduce power consumption:

- *Clock enable* (CKE). The SDRAM clock can be gated internally to the memory device by enabling or disabling the SDRAM CKE. This reduces power consumption of the SDRAM memory when the device is idle.

  The MPMC supports dynamic enabling and disabling of CKE, by programming the MPMCDynamicControl Register, dynamic memory clock enable (CE) field.
- Self-refresh mode. If the SDRAM memory does not have to be accessed, but the contents of the memory must be retained the memory can be placed in self-refresh mode. In this mode the SDRAM memory performs the required refresh sequences internally. The ASIC can then be put in a low-power mode, and the clocks can be disabled.

  The MPMC supports the self-refresh mode by the use of the self-refresh request (SR) field in the MPMCDynamicControl Register, the self-refresh acknowledge (SA) field in the MPMCStatus Register, or by hardware by using the **MPMCSREFREQ**, and **MPMCSREFACK** signals.
- SDRAM clock gating. The PrimeCell MPMC supports another low-power mode where the SDRAM memory clock is gated in the memory. This provides additional power reduction over using the CKE methodology because the heavily loaded external SDRAM clock signal is disabled before it goes off chip. This means that the output pad and PCB do not toggle unnecessarily.

  The MPMC supports SDRAM output clock gating by the use of the memory clock static enable (MSE), memory clock dynamic enabled (MDE), and memory clock self-refresh enable (MRE) fields of the MPMCDynamicControl Register. Additionally, you can disable the unused differential clock **nMPMCCLKOUT** using the differential memory clock static enable (DMSE) field of the MPMCDynamicControl Register.

Note

Programming both the dynamic enabling and disabling of CKE and the external clock signal reduces power consumption during normal operation to a minimum.

JEDEC low-power SDRAM JEDEC low-power SDRAM has several additional power reduction features compared to standard SDRAM.

One of the main ways that low-power SDRAM memory parts reduce power consumption over standard SDRAM parts is that they operate at a lower voltage and therefore consume less power than standard SDRAM memories.

Low-power SDRAM parts are designed to work in systems where power consumption is an important consideration, and where the clock frequency of the design might not be particularly high. These devices therefore generally enable lower CAS latency values, (for example CAS latency 1), compared to standard SDRAM.

Self-refresh mode power consumption is also reduced for low-power SDRAM devices. The devices enable the ambient temperature to be programmed into the device. The frequency of the self-refresh commands generated internally by the SDRAM are then scheduled accordingly. To use this functionality the ASIC must be able to measure the temperature. This temperature value must then be programmed into the SDRAM memory device (using the MPMC) before self-refresh mode is entered.

Low-power SDRAM provides a facility for selecting which memory banks to refresh in self-refresh mode. The memory banks that are not refreshed lose their data. When using this feature it is beneficial to use the bank, row, column (BRC) address-mapping scheme in the MPMC, so that a linear area of memory is not mapped over multiple SDRAM memory banks.

Finally low-power SDRAM memories also support deep sleep mode. In this mode the SDRAM memory is powered down. The content of the memory is lost.

DDR-SDRAM DDR-SDRAM memory has the same power reduction as SDRAM.

DDR-SDRAM enables data to be transferred at double the rate of the clock, address and command signals. The memory interface uses half the number of data pins for a given bandwidth compared to SDR-SDRAM.

The DDR-SDRAM memory interface inherently consumes less power than standard SDR-SDRAM memory. However, when performing power analysis calculations comparing SDR and DDR memory systems you must bear in mind that DDR memory requires a differential external memory clock, and a *Delay Locked Loop* (DLL) on the system ASIC.

DDR-SDRAM memory parts also have DLLs on the memory devices themselves. These DLLs mean that the memory clocks output from the ASIC must be kept enabled while the memory device is in normal operating mode. Dynamically disabling the clock is not allowed.

Note

DDR-SDRAM memory devices have a minimum clock frequency, usually about 80MHz.

DLL DDR-SDRAM requires DLL blocks in the ASIC to capture the read data. The DLLs lock on to the period of a reference clock and from this can generate a temperature and voltage compensated delay. The DLL blocks can consume a reasonable amount of power. To reduce power consumption the DLLs can be disabled when the memory is not required to be accessed, for example when the SDRAM is placed into self-refresh mode. The DLL architecture also affects the amount of power that it is consumed. DLLs that continuously calibrate to the reference clock consume more power than DLLs that periodically recalibrate. For periodically recalibrating DLLs the recalibration frequency must be frequent enough to adjust for any temperature and voltage variations. In many systems the periodic DLLs are recalibrated during the auto-refresh cycles. This is because the refresh cycles are frequent enough to ensure that the DLL stays locked and enables the DLL to be calibrated while there are no read transactions in progress.

Low-power DDR-SDRAM Low-power DDR-SDRAM memory has the same power reduction as low-power SDRAM.

SDR-SDRAM memory power analysis For information on the power consumption of a particular low-power SDR-SDRAM memory device, see the respective data sheet.

Low power SDR-SDRAM memory power analysis For information on the power consumption on a particular low-power SDR-SDRAM memory device, see the respective data sheet.

DDR-SDRAM memory power analysis For information on the power consumption on a particular DDR-SDRAM memory device, see the respective data sheet.

11.1.4 PrimeCell MPMC

The MPMC is designed to reduce external memory consumption by enabling the memory clocks to be gated and supporting the low power features of the memory devices.

Power consumption can be reduced in the MPMC itself by disabling the clocks to the MPMC when they are not required. You must ensure that these clocks are gated cleanly.

MPMC clock gating

Clocks that are not required by a specific configuration can be tied off, or gated:

- **HCLK** is used to clock the AMBA interface signals, the static memory controller and the TIC. This is required for SDRAM auto-refresh commands.

  If the memory does not have to be accessed, and the SDRAM memory does not have to be refreshed, the clock can be gated.

- **MPMCHCLKDELAY** is required if the command delayed pad interface mode is used in the design. If the clock-delayed mode is used then this clock can be disabled or tied off.

- **nHCLK** is required if the read data has to be registered on the negative edge of **HCLK**. This might be required to meet timing requirements for capturing the read data from SDR-SDRAM if the data is unable to be transferred from the feedback clock to the **HCLK** domain directly. If DDR-SDRAM is used, capturing the read data on **nHCLK** might be required if the read data from the data strobe clock domain is unable to be transferred directly to the **HCLK** domain.

  If the read data does not have to be captured on the negative edge of **HCLK**, this clock can be disabled or tied off.

- **HCLKX2** is used for DDR-SDRAM read and write transactions. If DDR-SDRAM is not used in a system, or the DDR-SDRAM is idle, this clock can be gated, or tied off.

- **nHCLKX2** is used for DDR-SDRAM read and write transactions. If DDR-SDRAM is not used in a system, or the DDR-SDRAM is idle, this clock can be gated, or tied off.

——— **Note** ———

Clock gating must be performed with care and fully verified. If clock gating is performed incorrectly the system might not function.

Memory clock gating

Clocks that are not required by a specific configuration can be tied off or gated.

MPMCCLKOUT is used by SDR-SDRAM and DDR-SDRAM. This clock can be disabled automatically when the SDRAM memory is not in use, or manually by software by programming the MPMCDynamicControl Register appropriately.

DDR-SDRAM requires this clock to be enabled during normal operation. This clock can be disabled during self-refresh.

——— **Note** ———

- SDR-SDRAM only requires this clock to be enabled during transactions. This clock can be disabled when there are no transactions, and during self-refresh.
- DDR-SDRAM requires this clock to be enabled during normal operation. This clock can be disabled during self-refresh.

———————————

The **nMPMCCLKOUT** is used only by DDR-SDRAM.

The clock can be disabled automatically when the DDR-SDRAM memory is not in use, or manually by software by programming the MPMCDynamicControl Register appropriately.

——— **Note** ———

DDR-SDRAM requires this clock to be enabled during normal operation. This clock can be disabled during self-refresh.

———————————

Memory clock enable

The SDRAM clock enable signal, **MPMCCKEOUT**, is used to enable or disable the clock in the SDRAM memory device, to save power.

This signal can be disabled automatically when the SDRAM memory is not in use, or manually by software by programming the MPMCDynamicControl Register appropriately.

——— **Note** ———

SDR-SDRAM only requires **CKE** to be enabled during transactions. This signal can be disabled when there are no transactions.DDR-SDRAM requires **CKE** to be enabled during normal operation.

The **MPMCCKEINIT[3:0]** input controls the initial value of **MPMCCKEOUT[3:0]** after an **nPOR** reset.

Chapter 12
Delay Locked Loop

This chapter describes the *Delay Locked Loop* (DLL). It contains the following sections:

12.1 About the DLL

DDR-SDRAM provides data strobe signals that are used to register read and write data. For every eight bits of data bus, there is an associated data strobe. Each data strobe is intended to be routed with its associated data signals on the PCB to ensure that they are delayed by the same amount.

For reads (from the MPMC) the data strobe input signals are edge aligned with the read data, see Figure 12-1. The input data strobes are normally delayed on chip so that the data strobes can be used to register the read data.

The read data is only valid for a short period of time, so that strobe placement must be accurate. Therefore, it is normal to use a temperature and voltage compensated delay block, a *Delay Locked Loop* (DLL).

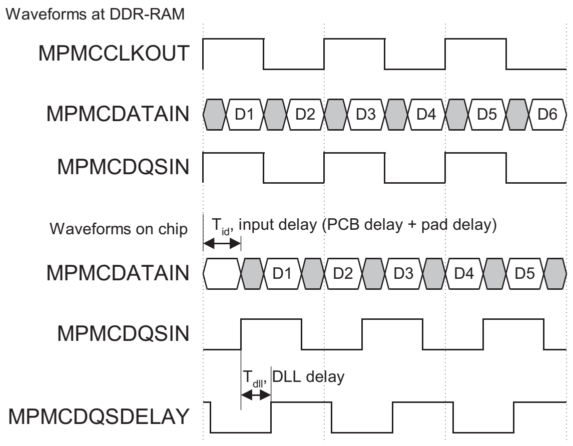


Figure 12-1 Read data and data strobe timing diagram

Note

The T_{dll} delay is required so that **MPMCDQSIN** can be used to capture **MPMCDATAIN**.

12.2 DLL calibration

The PrimeCell MPMC provides signals, **MPMCDLLCALIBREQ** and **MPMCDLLCALIBACK**, that can be used by systems using a DLL block, see Figure 12-2. DLL calibration can be enabled by setting the MPMCDynamicControl, *DLL Enable* (DE) field. If DLL calibration is disabled the calibration request signal, **MPMCDLLCALIBREQ**, does not go active.

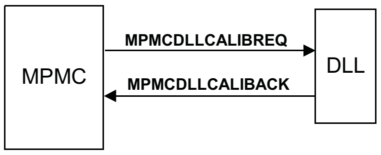


Figure 12-2 DLL calibration signals

If enabled the DLL calibration request signal, **MPMCDLLCALIBREQ**, goes active during SDRAM refresh, self-refresh exit and after DLL calibration is enabled. DLL calibration can also be initiated by software by setting the MPMCDynamicControl, *DLL Calibrate* (DC) field. This is useful if the system has just powered up and the DLLs are not calibrated. The *DLL Status* (DS) field of the MPMCDynamicControl Register can be accessed to determine when calibration is complete. When the calibration request signal goes active the DLL block brings the calibration acknowledge signal, **MPMCDLLCALIBACK**, HIGH. The DLL then performs calibration. When the DLL has performed its calibration it brings the acknowledge signal LOW. The PrimeCell MPMC then brings its calibration request signal LOW and enables the memory to be accessed. A waveform of the calibration signals is shown in Figure 12-3.

——— **Note** ———

No memory transactions, apart from SDRAM refresh, take place while the DLL is being calibrated.

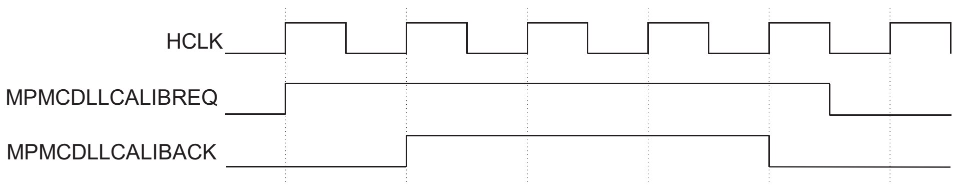


Figure 12-3 DLL calibration waveforms

12.2.1 Software programmed calibration sequence

If DLL calibration is initiated by software by asserting the MPMCDynamicControl DC field, the **MPMCDLLCALIBREQ** signal goes active. The DLL then brings the **MPMCDLLCALIBACK** signal HIGH and starts calibration. When calibration has completed the **MPMCDLLCALIBACK** signal is brought LOW by the DLL block. The MPMCDynamicControl Register DS field then goes HIGH, to indicate that the DLL is calibrated. The software polls the DS field until it goes HIGH and then clears the MPMCDynamicControl DC field. The memory can then be accessed.

12.3 DLL implementations

DLL blocks can either perform voltage and temperature calibration continuously, or at set intervals. The disadvantage of continuous calibration is that it can use considerable power. For this reason recalibration at set intervals, for example during SDRAM refresh, is most often used.

There are three main DLL architectures:

Absolute delay
The absolute delay value is determined. A clock is used as a reference. See *Absolute delay*.

Percent-of-clock delay The delay value is determined as a percentage of the clock. A clock is used as a reference. See *Percent-of-clock delay* on page 12-6.

Selectable delay Read transfers are performed with various delay values. The most appropriate delay is chosen. See *Selectable delay* on page 12-7.

12.3.1 Absolute delay

Figure 12-4 on page 12-6 shows a diagram of the absolute delay DLL. The delay control logic selects the output tap from the delay line of the master DLL and provides a delay equal to the reference clock period. When the delay control logic is calibrated it can then select the appropriate delay line output tap from the delay lines of the slave DLLs. For example, assume that you want to delay the data strobe signals by half a reference clock period, and that ten delay elements are required to provide a delay equal to the reference clock period. In this case the delay control must select five delay elements in the delay lines of the slave DLLs.

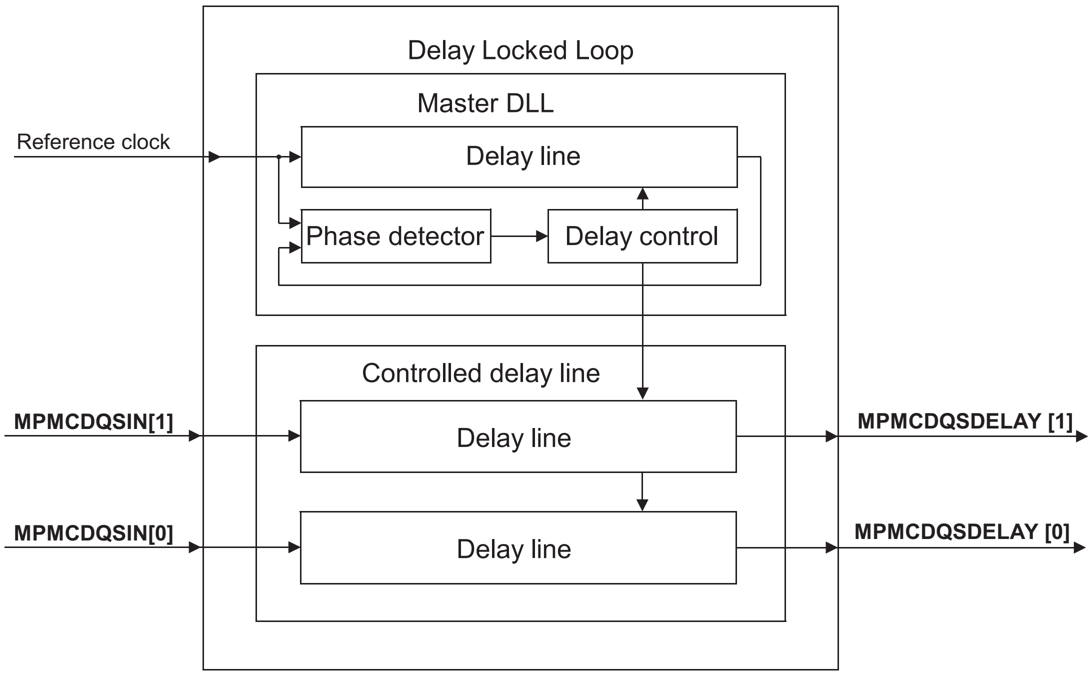


Figure 12-4 Absolute delay DLL

12.3.2 Percent-of-clock delay

Figure 12-5 on page 12-7 shows a diagram of the percent-of-clock delay DLL.

The percent-of-clock delay DLL is similar to the absolute delay DLL. However, in this case the delay line is composed of several blocks that contain delay elements. The number delay blocks in the delay line of the master DLL and the slave DLLs are chosen to provide the appropriate delay ratio.

The delay control logic of the master DLL selects the required number of delay elements in the each of delay blocks and provides a delay equal to the reference clock period. When calibration is complete the delay control logic selects the appropriate delay block output tap in the delay line of the slave DLLs.

For example, assume that the data strobe signals are required to be delayed by 25% of the reference clock period. Four identical blocks delay blocks can be used in the delay line of the master DLL. A single delay block is used in the delay line of the slave DLL. The delay control logic tunes the delay in all four of the delay blocks in the master DLL by the same amount so that the total delay equals the reference clock. The delay control logic can then select the same number of delay elements in the delay block of the slave DLLs to provide the required delay.

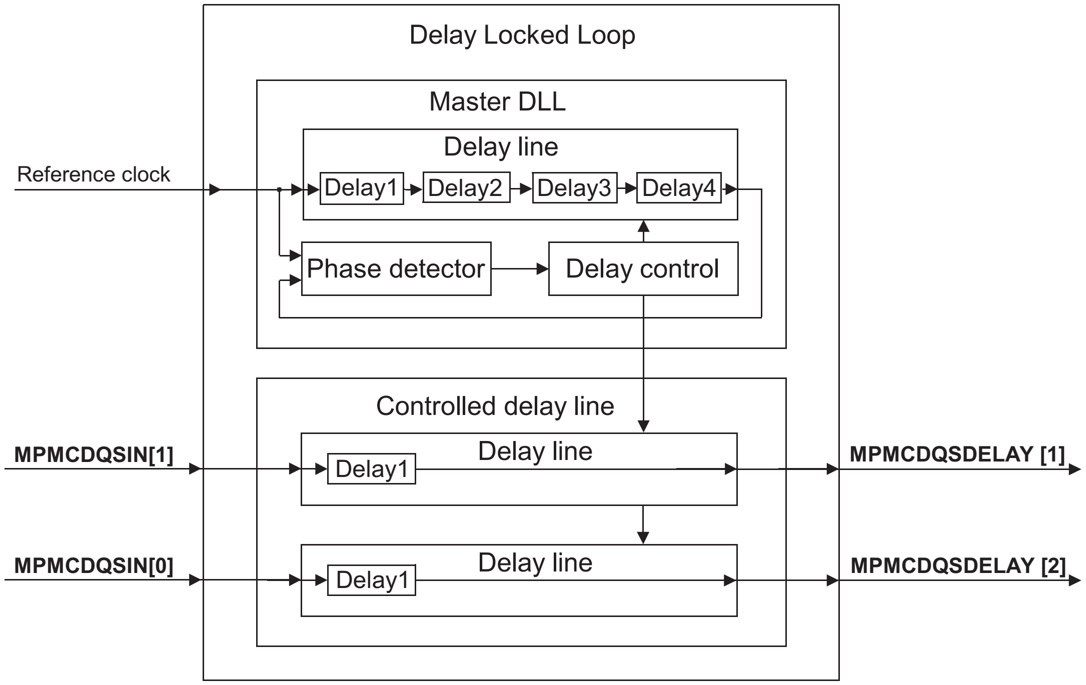


Figure 12-5 Percent -of-clock DLL

12.3.3 Selectable delay

Figure 12-6 on page 12-8 shows a diagram of the selectable delay DLL.

At specific intervals, for example after SDRAM refresh, data is read from the memory devices using different delay values. A delay value where the delay is too small (and the data is corrupted), and a delay value where the delay value is too large is found. Using this information a delay value in between these two extremes is programmed.

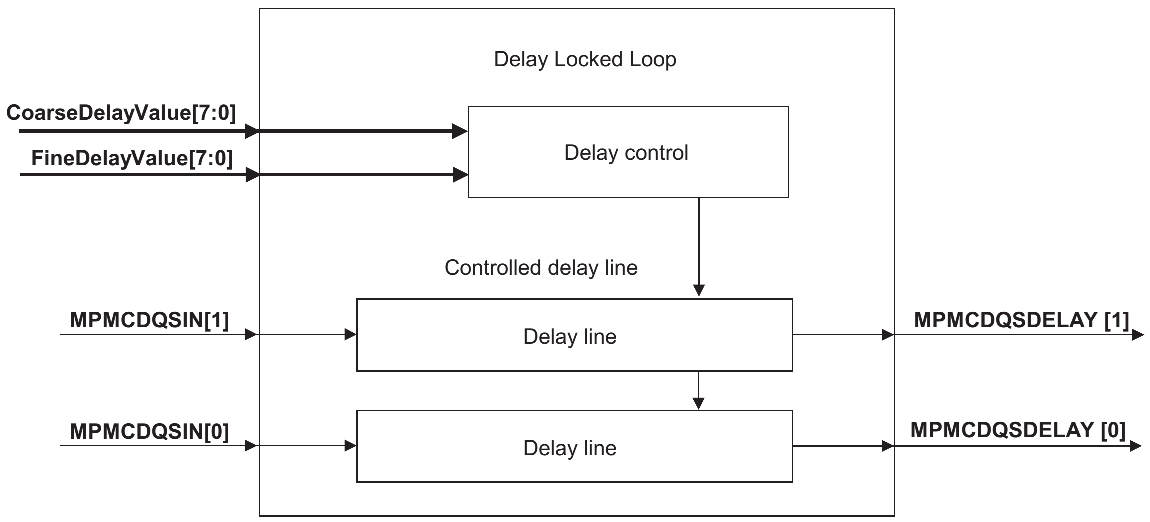


Figure 12-6 Selectable delay DLL

Appendix A
Pad Interface Timing

This appendix describes the signals that interface with the ARM PrimeCell MPMC block. It contains the following sections:

A.1 About pad interface timing

The performance of the PrimeCell MPMC is normally limited by the speed that data, address and control information can be sent to, and data can be read from, the SDRAM memory devices.

There are two factors that limit pad interface performance:

- signal delay to and from the SDRAM memory devices
- skew between the signals.

Note

Process shrinks do not generally improve the performance of the pads. Therefore a memory controller whose performance is limited by the pad interface does not generally run any faster when a smaller process geometry is used.

A.2 Signal delay

The pads add a large delay and skew to the signals going both off and on chip. This delay is, in part, dependent on the load of the PCB interconnect and the devices to be driven.

The PCB interconnect also add a delay, and skew, on the signals. Figure A-1 shows an on-chip to off-chip delay for a single signal.

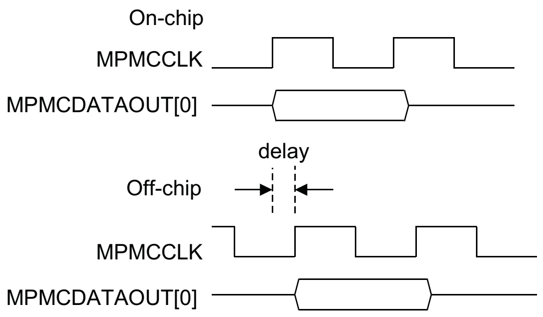


Figure A-1 On-chip to off-chip delay for a single signal

A.3 Method to reduce delay

You can reduce signal delay by using appropriately sized pads, and minimizing the load on the pads. For example by reducing the number of memory devices in the system.

If you use more than one output clock, **MPMCCLKOUT**, it enables the load on the clock to be spread.

A.3.1 Signal skew

Signal skew can be split into static, dynamic, and environmental effects.

A.3.2 Static effects

The types of static effects are:

Routing mismatch

The paths of the different signals might be different.

Different loading

The signals might be loaded differently.

Source timing different

The signals might be generated at different times.

A.3.3 Dynamic effects

The types of dynamic effects are:

Data pattern

The data pattern might affect the time when the signal becomes valid. For example a signal changing from 0 to 1 might take a different time from a signal changing from 1 to 0.

***Simultaneously Switching Outputs* (SSO)**

If a large number of outputs change at the same time it might take more time to reach the valid value than when a small number of signals change.

Figure A-2 on page A-5 shows on-chip to off-chip delay with skew for a single bit of the data bus **MPMCDATAOUT[0]**. Registering this signal on the rising edge of the clock, in this case, captures the data correctly.

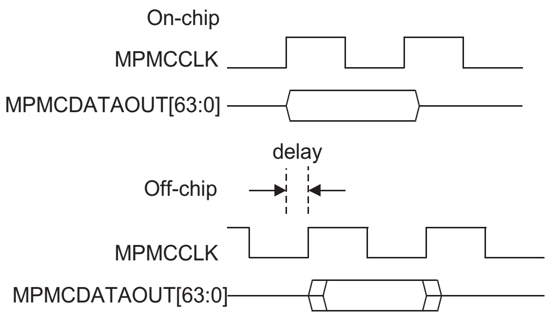


Figure A-2 On-chip to off-chip delay with skew for a single signal

Figure A-3 shows on-chip to off-chip delay with skew for multiple signals, **MPMCDATAOUT[63:0]**. You can see that the time when all the signals on the data bus are valid at the same time is smaller. In this case registering the group of signals on the rising edge of the clock does not capture the data correctly.

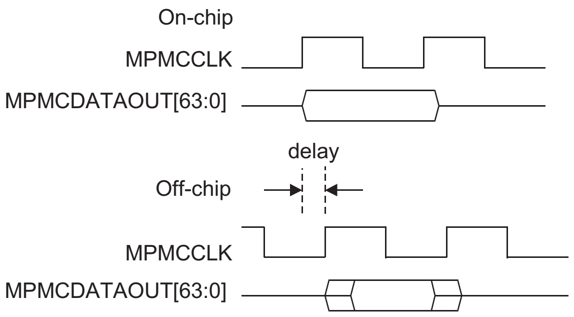


Figure A-3 On-chip to off-chip delay with skew for multiple signals

A.3.4 Environmental effects

The above sources of skew are affected by both temperature and voltage. This adds additional skew, because the design must work at both:

- high voltage, low temperature
- low voltage, high temperature.

A.4 Methods to reduce skew

Signal skew can be reduced by using appropriate pads. In some systems this could, for example, be performed by using the same tristate bidirectional pads for the data, address, and control signals.

The PCB routing for each of the signals, and the loading of the signals must ideally be similar.

A.5 Methods to minimize the effects of delay and skew

The effects of delay and skew can be minimized on the read data path by using feedback clocks that are used to register the data in the center of the data valid window. There are eight feedback clocks, **MPMCFBCLK[7:0]**. Each feedback clock is associated with eight bits of the data bus:

- **MPMCFBCLK[0]** is associated with **MPMCDATA[7:0]**
- **MPMCFBCLK[1]** is associated with **MPMCDATA[15:8]**
- **MPMCFBCLK[2]** is associated with **MPMCDATA[23:16]**
- **MPMCFBCLK[3]** is associated with **MPMCDATA[31:24]**
- **MPMCFBCLK[4]** is associated with **MPMCDATA[39:32]**
- **MPMCFBCLK[5]** is associated with **MPMCDATA[47:40]**
- **MPMCFBCLK[6]** is associated with **MPMCDATA[55:48]**
- **MPMCFBCLK[7]** is associated with **MPMCDATA[63:56]**.

Ideally the feedback clocks are generated off chip from the output clock. They must then be routed with their associated data to the ASIC. Therefore, the feedback clock is delayed by a similar amount as the read data.

A.6 Example SDRAM memory timing diagram

Figure A-4 shows an SDRAM timing diagram for the MT48LC4M16A2 Micron device, speed grade -7E, CAS=2. You can see that the control, address, and write (output) data all have the same set-up and hold timing values. The read data has different timing parameters to the other signals. The read data is valid 2.1ns before the positive edge of the clock and valid 2.7ns after the clock edge. Therefore a feedback clock generated from the clock at the memory device is aligned with the center of the read data valid window.

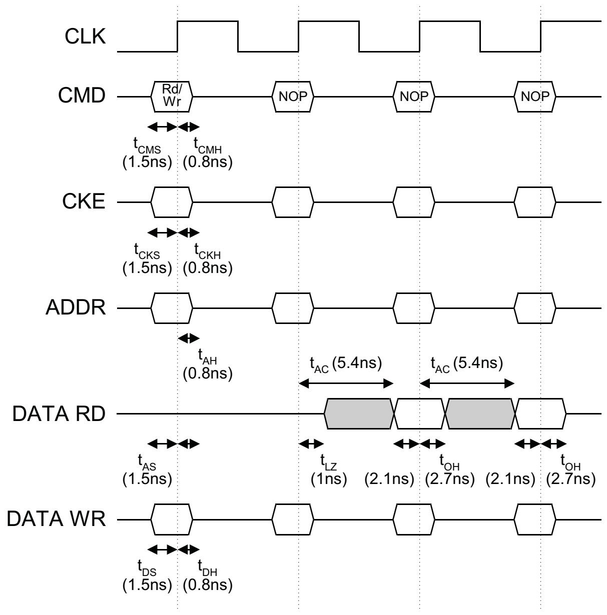


Figure A-4 SDRAM timing diagram

A.7 SDRAM memory timing paths

There are two main timing paths:

- MPMC to the SDRAM memory (control, address, and write data transfers)
- SDRAM memory to the MPMC (read data transfers).

A.7.1 MPMC to SDRAM memory

The command, address, and write data are generated by an on chip clock. These signals are sent off chip to the SDRAM memory. The signals arrive at the SDRAM memory delayed by t_{PID} (output pad and interconnect delay), see Figure A-5.

Note

Figure A-5 assumes there is no skew. To ensure that the command, address, and write data are captured correctly, either they or the clock must be delayed to ensure that the clock edge occurs in the capture window.

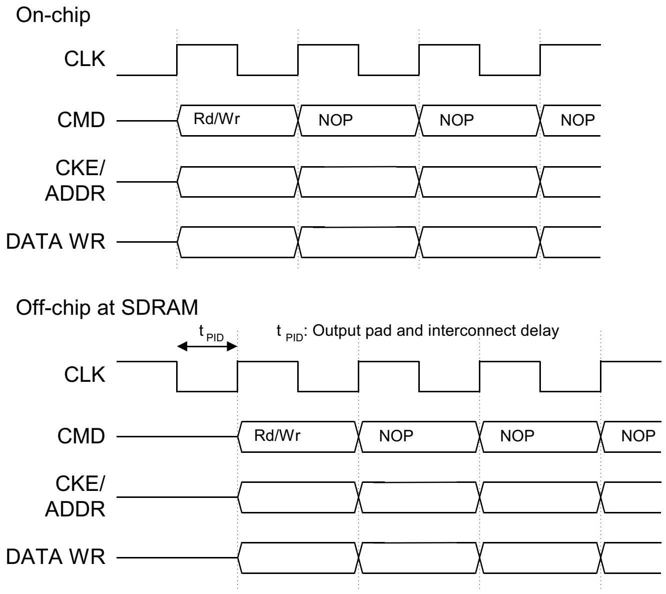


Figure A-5 MPMC to SDRAM memory timing path

A.7.2 SDRAM memory to MPMC

The feedback clock is generated from the output clock. The feedback clock and read data are sent on chip to the MPMC, delayed by t_{IPD} (interconnect and input pad delay) as shown in Figure A-6. The on-chip feedback clock edge occurs in the center of the read data capture window as it is delayed by the same amount as the read data.

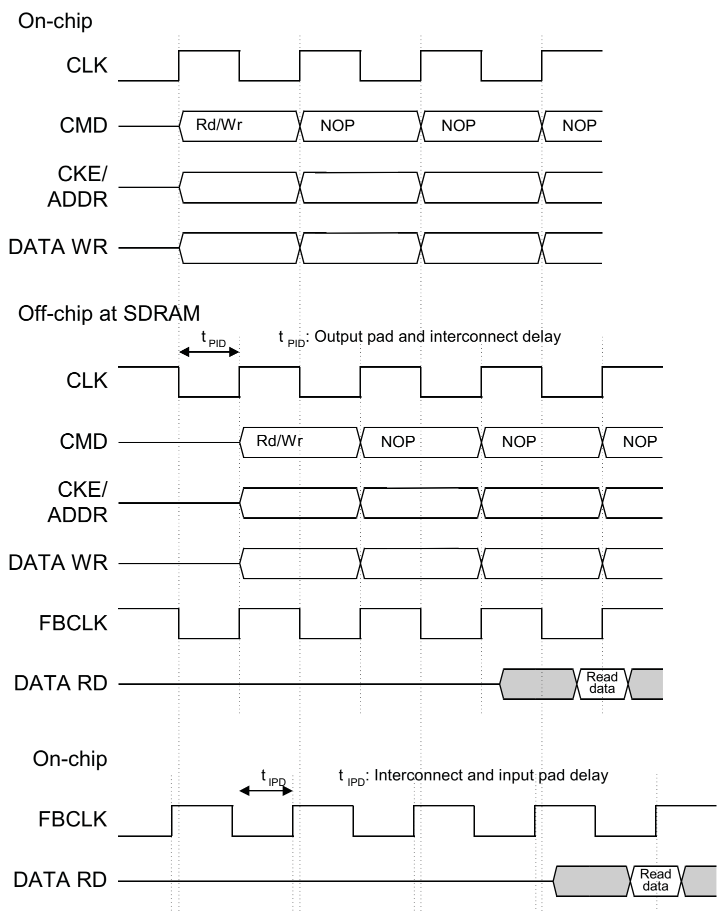


Figure A-6 SDRAM memory to MPMC timing path

A.8 Example DDR-SDRAM memory timing diagram

Figure A-7 shows a DDR-SDRAM diagram for the MT46V32M4 Micron device, speed grade -8, CAS=2. The control and address all have the same set-up and hold timing values. The read and write data have different timing parameters to the other signals. The read data is valid up to 0.8ns after each clock edge until up to 0.8ns before each clock edge.

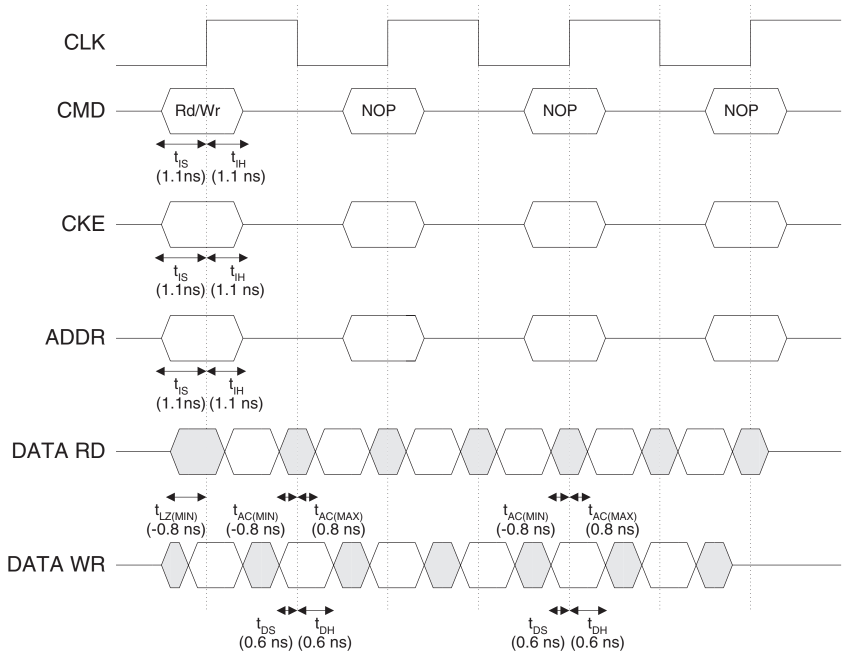


Figure A-7 DDR-SDRAM timing diagram

A.9 Example NAND flash memory timing diagrams

Figure A-8 to Figure A-13 on page A-14 show NAND flash timing diagrams for the K9F5608U0B Samsung device, with the timing values found in the MPMCNDTiming1/2 registers included. See the device data sheet for full timing diagrams of all transfer types supported.

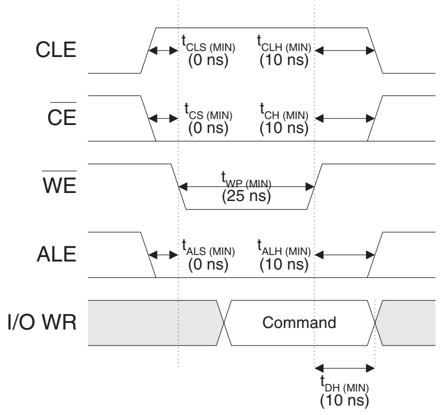


Figure A-8 NAND flash timing diagram command latch cycle

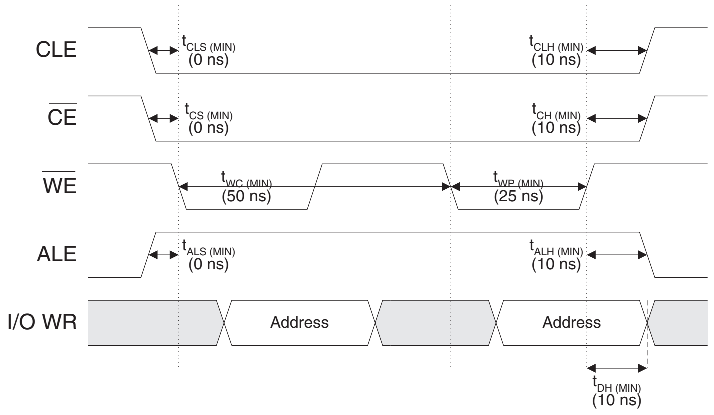


Figure A-9 NAND flash timing diagram address latch cycle

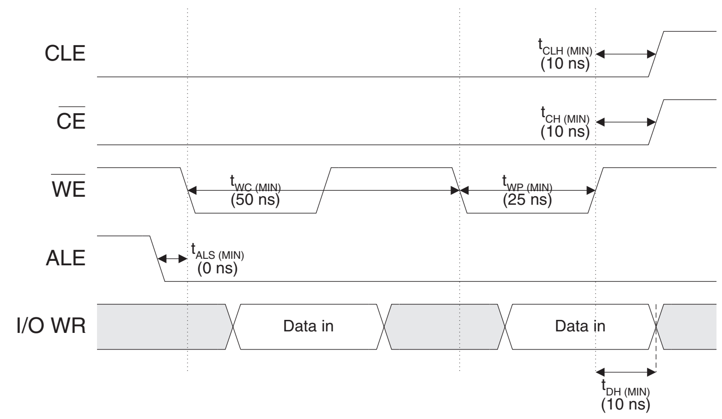


Figure A-10 NAND flash timing diagram data input cycle

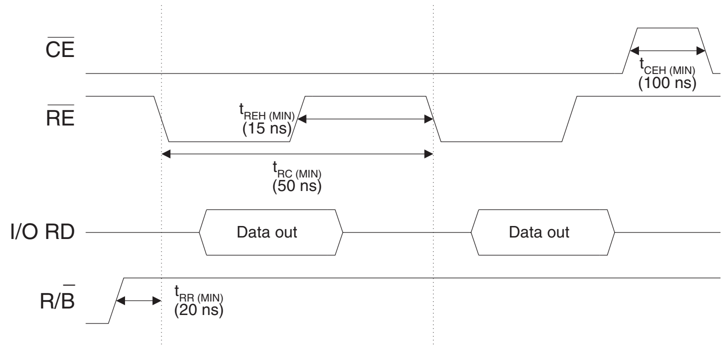


Figure A-11 NAND flash timing diagram data output cycle

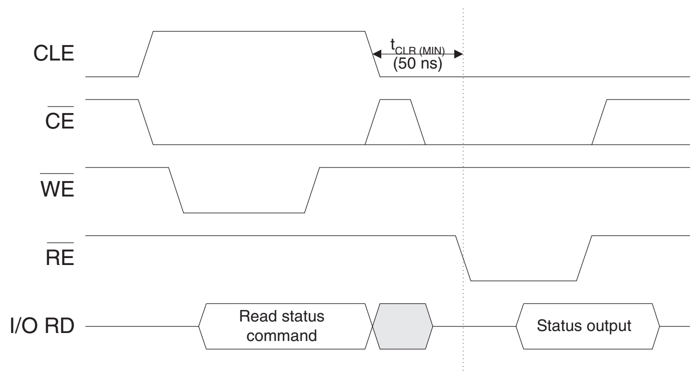


Figure A-12 NAND flash timing diagram status read cycle

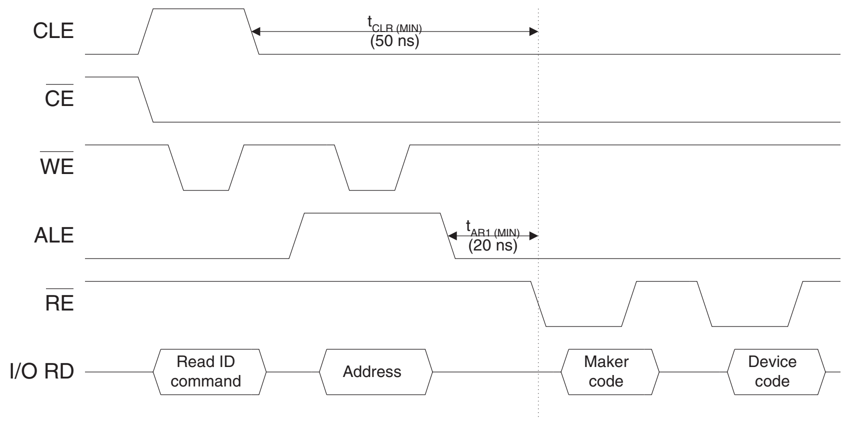


Figure A-13 NAND flash timing diagram ID read cycle

Appendix B
Troubleshooting

This appendix describes how to troubleshoot the ARM PrimeCell MPMC block. It contains the following sections:

- *Software* on page B-2
- *System* on page B-6.

B.1 Software

This section covers typical software problems. See Table B-1 for a list of problems and their suggested remedies.

Table B-1 Software troubleshooting

Problem	Suggested remedy
Dynamic memory does not work. However, static memory operates correctly.	Ensure that the SDRAM memory is initialized correctly. The appropriate address mapping, burst size, and CAS latency must be programmed.[a]
Mode programming and initialization issues.	The auto-refresh value programmed in the MPMCDyRef Register must be greater than the precharge period and the time required for applying the auto-refresh requests to all the dynamic memory chip selects. Auto-refreshValue > (MpmcDytRPQ + 1) + (MpmcDytRFCQ + 1) + 4) where MpmcDytRPQ is the precharge period, MpmcDytRFCQ is the auto-refresh latency and the four extra clock cycles are required to apply the auto-refresh cycles. A value programmed equal to or less than this always keeps the refresh state machine running and the MPMC ends up giving only refresh cycles. The SDRAM memory initialization can take two possible variations: • NOP, PALL, MODE, NORMAL (normal SDRAM sequence) • MODE, NORMAL (Micron SyncFlash sequence). In both the cases, there must be at least one idle cycle between the first and second steps. For normal SDRAMs, a 200µs delay must be maintained between the first and second steps. Therefore, the one-clock idle is automatically satisfied. Nothing specific is mentioned in the SyncFlash initialization sequence. This restriction comes because of the necessity of the mode state machine to be able to restart the clock and CKE, after exit from the deep-sleep state.[b]
The MPMC does not seem to be programmed correctly.	If a cached processor is being used to initialize the MPMC ensure that the MPMC register address space is marked as noncacheable, and nonbufferable.
The MPMC is in self-refresh mode on power-on reset.	The MPMC sets the self-refresh request and self-refresh acknowledge software bits HIGH on power-on reset. This enables SDRAM memory that has been placed into self-refresh mode to be held if the ASIC is powered down. The self-refresh request bit must be programmed LOW to enable the SDRAM memory to be accessed. This takes the SDRAM **CKE** signal HIGH, bringing the SDRAM memory out of self-refresh. The MPMC then ensures that the SDRAM self-refresh exit time is adhered to. If the SDRAM memory was not previously placed into self-refresh mode, this sequence has no effect on the memory.[c]
The MPMC initialization code that has been developed does not work.	Examine the initialization sequences provided in this document.
	Examine the device driver code provided in the delivery.

Table B-1 Software troubleshooting (continued)

Problem	Suggested remedy
How do I initialize the SDRAM memory?	Initialization sequence: 1. Keep the CE and CS bit in the MPMCDyControl Register HIGH. 2. Perform the mode initialization. During the mode-programming process the MPMCDyControl Register must be written to repeatedly (for altering the SdramInit field). Ensure that the CE and CS bits (residing in the same register) are always kept at 11 value. 3. After mode-programming is finished, reprogram to enable or disable the buffers as required. 4. Program the CS and CE bits to the desired configuration, when the mode-programming is done. If the CE and CS bits are not kept HIGH, the refresh state machine switches off the clock or clock-enable during the mode-programming sequence.
	Mode-register programming is done only at the very beginning (after reset). Multiple mode programming is required only if the controller is programmed for deep-sleep mode entry. If there is no deep-sleep entry, no multiple-mode programming is allowed, and is not necessary.
	After the controller exits deep-sleep mode, all the chip-selects must be mode-programmed (in case a few chips are deep-sleep SDRAMs and the rest are normal SDRAMs).
Reads to flash memory are fine, however flash memory writes do not function correctly.	Ensure that the flash memory space is marked as noncacheable and nonbufferable when writing to or controlling the flash memory.
	Ensure that the flash AHB port buffers are disabled when writing to or programming the flash memory.
Transactions to a peripheral connected to a static memory chip select do not function correctly.	Ensure that the peripheral memory space is marked as noncacheable and nonbufferable.
	Ensure that the buffers are disabled for the relevant AHB port.
Static memory page mode accesses do not function correctly. However, normal static memory accesses function correctly.	Ensure that the address output signal, **MPMCADDROUT**, is connected to the correct address bits on the static memory. The least significant address bit of the MPMC output address must be connected to the least significant address bit of the static memory device.
	The MPMC address output signal must be connected unscrambled to the memory devices address bus.

Table B-1 Software troubleshooting (continued)

Problem	Suggested remedy
Accessing a byte wide static memory chip select functions correctly. However, accessing 16-bit or 32-bit chip selects do not function correctly.	Ensure that the address output signal, **MPMCADDROUT**, is connected to the correct address bits on the static memory. The least significant address bit of the MPMC output address must be connected to the least significant address bit of the static memory device.
The SDRAM memory does not seem to be initialized correctly.	Ensure that the SDRAM memory space is marked as noncacheable and nonbufferable when programming the SDRAM memory.
	Ensure the buffers for the relevant chip select are disabled when programming the SDRAM memory.
The SDRAM read data returns earlier or later than expected.	Ensure that the MPMC and SDRAM memory have been programmed with the same CAS latency.
	Ensure that the clocking scheme chosen (see *Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig* on page 3-17) is the same as the scheme chosen in the testbench/system.
The bank-row-column or row-bank-column SDRAM addressing scheme works, however the other addressing scheme does not function correctly.	Ensure that when programming SDRAM memory the correct address mapping scheme is used.
Static memory byte lane state issues.	For most static memory chip selects constructed from eight-bit or nonbyte-partitioned memory devices, the byte lane state (PB) bit must be cleared to 0 within the respective MPMCStConfig Register. For most static memory chip selects constructed from 16 or 32-bit memory devices, the byte lane select (PB) bit must be set to 1 within the respective MPMCStConfig Register.
The MPMC seems to be performing additional accesses to SDRAM memory or the SDRAM memory contents appear to be incorrect.	Ensure that the MPMC has been initialized correctly.
	SDRAM memory devices connected to a chip select with a 64-bit wide data bus must be programmed to perform a burst of one.SDRAM memory devices connected to a chip select with a 32-bit wide data bus must be programmed to perform a burst of two.SDRAM memory devices connected to a chip select with a 16-bit wide data bus must be programmed to perform a burst of four.
	Ensure the value used to program the SDRAM memory is correct. The value used to program the SDRAM memory depends on the address mapping scheme chosen.

Table B-1 Software troubleshooting (continued)

Problem	Suggested remedy
When can the MPMC be enabled and disabled?	The MpmcEnable bit of the MPMCControl Register can be made LOW only when: • the busy bit of the MPMCStatus Register is LOW • the buffer status (S) bit of the MPMCStatus Register is LOW • the memory control and configuration registers (both dynamic and static) can only be initialized during system initialization or when the MPMCEnable bit is disabled.
NAND flash accesses are not possible.	Ensure the ND bit of the NAND flash device chip select MPMCStaticConfig Register is set to 1, and that the NDCS bits of the MPMCNANDControl Register are set to the same chip select.
Programming the NAND flash registers is not possible.	The NDINTFBUSY bit in the MPMCNANDStatus Register must be checked before attempting to program the control or address registers. A current NAND flash transfer must be idle before the registers can be programmed.
	The NDTX bit in the MPMCNANDStatus Register must be checked before attempting to program the timing registers. There must be no active NAND flash transfers before the registers can be programmed.
NAND flash read transfers never complete, and SRAM/SDRAM accesses are not possible.	Ensure the ready output of the device is connected to a pull-up resistor of a size specified in the device data sheet. This is required to enable the ready output to be sampled correctly by the MPMC.

a. During initialization the AHB port buffers must be disabled, and CKE (MPMCDynamicControl, CE = 1) must be HIGH.

b. The normal sequence for SDRAMs can also be applied to SyncFlash memory parts (especially in the situation where the chip selects are populated with both SyncFlash and SDRAM devices). However, if only SyncFlash devices are used in a system, and SDRAM is not used, you can use just the SyncFlash sequence.

c. The **MPMCCKEINIT[3:0]** and **MPMCDQMINIT** inputs control the initial values of the **MPMCCKEOUT[3:0]** and **MPMCDQMOUT[7:0]** outputs respectively after an **nPOR** reset. These are used to ensure that the memory control signals are driven to the correct values for the memory devices in use, depending on whether the device is uninitialized or in self-refresh mode.

B.2 System

This section covers typical system problems. See Table B-2 for a list of problems and their suggested remedies.

Table B-2 System troubleshooting

Problem	Suggested remedy
When can the self-refresh signal go LOW?	The **MPMCSREFREQ** pin, or the self-refresh request (SR) field in the MPMCDynamicControl Register must not go LOW until the **MPMCSREFACK** pin and the self-refresh acknowledge (SA) bit of the MPMCStatus Register are asserted by the MPMC.
What is the SynchFlash sequence?	During SyncFlash initialization, the RP line is handled entirely by software. The **nMPMCRPOUT** and **MPMCRPVHHOUT** outputs are directly controlled by the respective bits in the MPMCDyControl Register. The flash-command-set operations (as against the normal array read mode, similar to SDRAMs) must be performed in the unbuffered mode.
What is the low-power deep-sleep sequence?	Before setting the deep-sleep bit in the MPMC, ensure the dynamic memory controller is idle, reflected by the busy bit in the MPMCStatus Register. The low-power, deep-sleep exit is not automatic. That is, the low-power, deep-sleep bit has to be explicitly reset by software and only then can normal accesses be started.
How must **HSELMPMCxG** and **HSELMPMCxS[7:0]** function?	**HSELMPMCxG** must go active whenever there is an access to the MPMC. At the same time the relevant bit of **HSELMPMCxS[7:0]** must go active.**HSELMPMCxS[7:0]** indicates which memory chip select the transfer is for.
What are the power-on reset values of input pins?	On power-on reset, the static memory chip select polarity pins, **MPMCSTCS[0, 1, 2, 3]POL**, must indicate the polarity of the respective chip selects (0 for active LOW and 1 for active HIGH). These values are used until the MPMCStConfig0-3 Registers are written into. When the configuration register of corresponding chip select is programmed, the register value of the chip select polarity bit in the register determines the chip select polarity.
	On power-on reset, the static memory chip select 1 data bus width pin **MPMCSTCS1MW[1:0]** must indicate the width of the chip select (00 for 8-bit, 01 for 16-bit and 10 for 32-bit data bus width). This value is used until the MPMCStConfig1 Register is written into. When the register is programmed, the memory width bits of the register determine the memory width.
	On power-on reset, the endian mode of the MPMC must be driven on the **MPMCBIGENDIAN** pin. This value is used until the MPMCConfig Register is written into. When this register is programmed, the register value of the Endian mode bit determines the endian mode.

Table B-2 System troubleshooting (continued)

Problem	Suggested remedy
How must the self-refresh signals be connected?	The **MPMCSREFREQ** pin must be connected to the *Power Management Unit* (PMU) in the system. The PMU must assert the **MPMCSREFREQ** signal to enter the self-refresh mode. When asserted, the **MPMCSREFREQ** pin must not go LOW until the **MPMCSREFACK** pin is asserted by the MPMC. In case the PMU does not support this mechanism, the self-refresh mode can be entered manually by software, by setting the **MPMCSREFREQ** bit of the MPMCDyControl Register and polling the **MPMCSREFACK** bit of the MPMCStatus Register.
How must the static memory write enable signals be connected?	For static memory chip selects constructed from 8-bit or non byte-partitioned memory devices, the **nMPMCBLSOUT[3:0]** signals must be connected to write enable (nWE) inputs of each 8-bit memory. The **nMPMCSTWEOUT** signal from the MPMC must not be used. For static memory chip selects constructed from 16-bit or 32-bit memory devices, the **nMPMCBLSOUT[3:0]** signals must be connected to the byte select inputs of each memory device. The **nMPMCSTWEOUT** signal from the MPMC must be connected to the write enable inputs of each memory device.
What are the requirements to enter test mode?	The TIC must be the highest priority master on the respective AHB bus.
	When the MPMC has entered test mode, it is assumed that there are no normal mode accesses. The MPMC enters test mode only if the sequence of events as indicated below is followed. The following sequence of events has to be followed to use the TIC of the MPMC to apply test patterns: 1. Apply reset (**HRESETn**) asynchronously. 2. Assert the **MPMCTESTIN** test input to enter test mode. 3. When the reset is removed (synchronously), the MPMC enters TIC test mode. 4. The TIC block asserts its **HBUSREQTIC** signal to the arbiter, requesting access to the AHB bus. Because the TIC is the highest priority, **HGRANTTIC** is asserted and the granted TIC takes ownership of the AHB bus.
	The **MPMCTESTIN** signal is usually fed to the clock generation logic so that **HCLK** can be switched to test speed on entry into test mode.

Appendix C
Signal Descriptions

This appendix describes the signals that interface with the ARM PrimeCell MPMC controller block. It contains the following sections:

C.1 AHB register signals

Table C-1 describes the AHB register signals.

Table C-1 AHB register signal descriptions

Name	Type	Source/ destination	Description
HADDRREG[11:2]	Input	AHB master layer	The AHB address bus.
HRDATAREG[31:0]	Output	AHB master layer	Read data bus. The read data bus is used to transfer data from the PrimeCell MPMC to the bus master during read operations.
HREADYINREG	Input	AHB slave layer	Transfer done. When HIGH, the **HREADYIN** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer. An alternate slave generates this signal.
HREADYOUTREG	Output	AHB master and AHB master layer	Transfer done. When HIGH, the **HREADYOUT** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPREG[1:0]	Output	AHB master layer	Transfer response. The transfer response provides additional information on the status of a transfer. The SDRAM controller can respond with either the OKAY or ERROR responses. The ERROR response is generated when the transfer size is not 32 bits wide. **HRESPREG[1]** = 0, because SPLIT and RETRY responses are not supported.
HSELMPMCREG	Input	AHB decoder	Slave select. PrimeCell MPMC controller register select signal. This signal indicates an access to the MPMC control registers.
HSIZEREG[2:0]	Input	AHB master layer	Transfer size. Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit) or word (32-bit). Only word (32-bit) transfers are allowed (**HSIZE[2:0]** = `0b010`) to access the PrimeCell MPMC registers. Transfer sizes other than 32 bits generate an ERROR response on **HRESPREG[0]**.
HTRANSREG	Input	AHB master layer	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE, or BUSY. Only **HTRANS[1]** is used, indicating if a transfer is required or not.
HWDATAREG[31:0]	Input	AHB master layer	Write data bus. The write data bus is used to transfer data from the master to the bus slaves during write operations.
HWRITEREG	Input	AHB master layer	Transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer.

C.2 AHB memory signals

Table C-2 describes the AHB memory signals. The signal names for each port can be found by substituting the port number, 0, 1, 2, through to 9 for the symbol x. For signals specific to 64-bit ports, the symbol x can only have a value of 3 to 6. Unused ports are disabled by connecting their inputs to logic 0. The PrimeCell MPMC does not support RETRY or SPLIT transactions.

Table C-2 AHB memory signal descriptions

Name	Type	Source/ destination	Description
HADDRx[28:0]	Input	AHB master layer	The AHB address bus.
HBSTRBx[7:0]	Input	ARM11 AHB master	Byte lane enables for individual byte lanes [63:56], [55:48], [47:40], [39:32], [31:24], [23:16], [15:8], and [7:0].
HBURSTx[2:0]	Input	AHB master layer	Burst type. Indicates if the transfer forms part of a burst. 4, 8, and 16 beat bursts are supported and the burst can be either incrementing or wrapping. Burst type INCR, incrementing burst of unspecified length, is interpreted as INCR4 by the MPMC.
HDOMAINx[1:0]	Input	Arbiter	The **HDOMAINx[1:0]** signals from the system indicate which bus domain is currently performing a transfer and determine which ARM11 compliant master is executing an Exclusive Access transfer. Support for monitoring a maximum of four domains is provided.
HMASTLOCKx	Input	AHB master layer	Locked transfers. When HIGH this signal indicates that the master requires locked access to the SDRAM and no other master must be granted the bus until this signal is LOW.
HPROTx[5] **HPROTx[3:2]**	Input	AHB master	Protection control signals. **HPROT[3:2]** provide additional information about a bus access. These signals indicate whether the transfer is: • an opcode fetch or a data access • a privileged mode access or a User mode access • a cacheable access or a noncacheable access • a bufferable access or a nonbufferable access. For ARM11 compliant bus masters, **HPROT[5]** indicates an outer memory load/store exclusive transfer.

Table C-2 AHB memory signal descriptions (continued)

Name	Type	Source/ destination	Description
HRDATAx[63:0]	Output	AHB master layer	Read data bus. The read data bus is used to transfer data from the PrimeCell MPMC to the bus master during read operations. 32-bit ports use the lower 32 bits of the 64-bit data bus, **HRDATA[31:0]**
HREADYINx	Input	AHB slave layer	Transfer done. When HIGH, the **HREADYIN** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer. An alternate slave generates this signal.
HREADYOUTx	Output	AHB master layer	Transfer done. When HIGH, the **HREADYOUT** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPx[1:0][a]	Output	AHB master layer	Transfer response. The transfer response provides additional information on the status of a transfer. The SDRAM controller can respond with either the OKAY or ERROR responses. The ERROR response is generated when: • the transfer size is greater than allowed • the memory access is a write to a write-protected region • the memory is accessed with the MCEnable bit disabled or the LowPowerMode bit is asserted.
HRESPx[2]	Output	AHB master layer	Transfer response. The transfer response provides additional information on the status of an exclusive transfer. The STREX response is generated when the port acknowledges an ARM11 store exclusive instruction.
HSELMPMCxCS[7:0]	Input	AHB decoder	Slave select. MPMC select signal. These signals indicate an access to a particular memory bank.
HSELMPMCxG	Input	AHB decoder	Slave select. PrimeCell MPMC global select signal. This signal indicates an access to memory.
HSIZEx[2:0]	Input	AHB master layer	Transfer size. Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit), word (32-bit), or Dword (64-bit). Byte (8-bit), halfword (16-bit), word (32-bit), and Dword (64-bit) transfers are allowed to access the external memory. Transfer sizes greater than 64 bits generate an ERROR response.
HTRANSx[1:0]	Input	AHB master layer	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE, or BUSY.

Table C-2 AHB memory signal descriptions (continued)

Name	Type	Source/ destination	Description
HUNALIGNx	Input	ARM11 AHB master	Indicates that the current transfer is part of an unaligned data transfer. Unaligned write transfers are not supported by all memory devices.
HWDATAx[63:0]	Input	AHB master layer	Write data bus. The write data bus is used to transfer data from the master to the bus slaves during write operations. 32-bit ports use the lower 32 bits of the 64-bit data bus, **HWDATA[31:0]**.
HWRITEx	Input	AHB master layer	Transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer.

a. **HRESPx[1]** = 0 because SPLIT and RETRY responses are not supported.

C.3 Miscellaneous signals

The MPMC miscellaneous signals are described in:

- *Tie-off signals*
- *Test signals* on page C-11
- *Clock and reset signals* on page C-11
- *DLL and self-refresh signals* on page C-12
- *External Bus Interface signals* on page C-12.

C.3.1 Tie-off signals

Table C-3 describes the tie-off signals.

Table C-3 Tie-off signal descriptions

Name	Type	Source/ destination	Description
MPMCBIGENDIAN	Input	Reset controller	Endian select. Determines the default endianness of the AHB slave ports and external memory banks. If this bit is HIGH, big-endian mode is selected. The value can be overridden by writing to the N field of the MPMCConfig Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.
MPMCBOOTDLY	Input	Reset controller	When HIGH, indicates that no accesses must be performed. Used for dynamic memory devices that cannot be accessed immediately after reset. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.
MPMCBSTRBENx	Input	Reset controller	Byte strobe enable. Enables ARM11 AHB byte strobe inputs **HBSTRB[7:0]** for unaligned data support. Enabling the byte strobes for a given port also fixes that port into byte-invariant mode for ARM11 mixed-endian support.
MPMCCKEINIT[3:0]	Input	Reset controller	Defines the power-on level for the **MPMCCKEOUT** signals after **nPOR**. When SDR/DDR memory is in self-refresh mode this value must be `0x0`. For uninitialized SDR/DDR memory this value must be `0xF`.

Table C-3 Tie-off signal descriptions (continued)

Name	Type	Source/ destination	Description
MPMCDQMINIT	Input	Reset controller	Defines the power-on level for the **MPMCDQMOUT** signals after **nPOR**. When SDR/DDR memory is in self-refresh mode this value must be 0. For uninitialized SDR/DDR memory this value must be 1.
MPMCDYCS5ADRMAP[8:0]	Input	Reset controller	Indicates the address map for chip select 5. Used for dynamic memory devices. The value can be overridden by writing to the AM field of the MPMCDynamicConfig1 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3* on page 3-25.
MPMCDYCS5CASDLY[3:0]	Input	Reset controller	Indicates the upper four bits of CAS delay for chip select 5. Used for dynamic memory devices. The value can be overridden by writing to the upper three bits of the CAS field of the MPMCDynamicRasCas1 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory RAS and CAS Delay Registers 0-3, MPMCDynamicRasCas0-3* on page 3-30.
MPMCDYCS5DEVICE[2:0]	Input	Reset controller	Indicates the type of device connected to chip select 5. Used for dynamic memory devices. The value can be overridden by writing to the MD field of the MPMCDynamicConfig1 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory Configuration Registers 0-3, MPMCDynamicConfig0-3* on page 3-25.

Table C-3 Tie-off signal descriptions (continued)

Name	Type	Source/ destination	Description
MPMCDYDDRDLY[1:0][a]	Input	Reset controller	DDR clocking scheme for data capture. Used for dynamic memory devices. The value can be overridden by writing to the DRD field of the MPMCDynamicReadConfig Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig* on page 3-17.
MPMCDYDDRPOL[a]	Input	Reset controller	The polarity of the register in which the DDR read data is first captured. Used for dynamic memory devices. The value can be overridden by writing to the DRP field of the MPMCDynamicReadConfig Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig* on page 3-17.
MPMCDYSDRDLY[1:0]	Input	Reset controller	SDR clocking scheme for data capture. Used for dynamic memory devices. The value can be overridden by writing to the SRD field of the MPMCDynamicReadConfig Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig* on page 3-17.
MPMCDYSDRPOL	Input	Reset controller	The polarity of the register in which the SDR read data is first captured. Used for dynamic memory devices. The value can be overridden by writing to the SRP field of the MPMCDynamicReadConfig Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value. For more information on the tie-off values see *Dynamic Memory Read Configuration Register, MPMCDynamicReadConfig* on page 3-17.

Table C-3 Tie-off signal descriptions (continued)

Name	Type	Source/ destination	Description
MPMCSTCS0POL	Input	Reset controller	Chip select 0 polarity select on **nPOR**. A HIGH value indicates CS active HIGH, a LOW value indicates CS active LOW. Used for static memory devices. The value can be overridden by writing to the PC field of the MPMCStaticConfig0 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.
MPMCSTCS1MW[1:0]	Input	Reset controller	Chip select 1 memory width selection: 00 = eight-bit wide memory 01 = 16-bit wide memory 10 = 32-bit wide memory 11 = reserved. Used for static memory devices. The value can be overridden by writing to the MW field of the MPMCStaticConfig1 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.
MPMCSTCS1PB	Input	Reset controller	Chip select 1 byte lane polarity select.0 = for reads all the bits in **nMPMCBLSOUT[3:0]** are HIGH. For writes the respective active bits in **nMPMCBLSOUT[3:0]** are LOW.1 = for reads the respective active bits in **nMPMCBLSOUT[3:0]** are LOW. For writes the respective active bits in **nMPMCBLSOUT[3:0]** are LOW.Used for static memory devices. The value can be overridden by writing to the PB field of the MPMCStaticConfig1 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.

Table C-3 Tie-off signal descriptions (continued)

Name	Type	Source/ destination	Description
MPMCSTCS1POL	Input	Reset controller	Chip select 1 polarity select on **nPOR**. A HIGH value indicates CS active HIGH, a LOW value indicates CS active LOW. Used for static memory devices. The value can be overridden by writing to the PC field of the MPMCStaticConfig1 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.
MPMCSTCS2POL	Input	Reset controller	Chip select 2 polarity select on **nPOR**. A HIGH value indicates CS active HIGH, a LOW value indicates CS active LOW. Used for static memory devices. The value can be overridden by writing to the PC field of the MPMCStaticConfig2 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.
MPMCSTCS3POL	Input	Reset controller	Chip select 3 polarity select on **nPOR**. A HIGH value indicates CS active HIGH, a LOW value indicates CS active LOW. Used for static memory devices. The value can be overridden by writing to the PC field of the MPMCStaticConfig3 Register. This signal must not be altered during normal operation. This signal can be generated by registering an input signal on **nPOR** in the reset controller. Alternatively, this signal can be tied to the appropriate value.

a. This signal is reserved for future use and must be tied LOW.

C.3.2 Test signals

Table C-4 describes the test signals.

Table C-4 Test signal descriptions

Name	Type	Source/ destination	Description
MPMCTESTIN	Input	Pad	Test mode, used to place the MPMC into TIC test mode.
MPMCTESTREQA	Input	Pad	Test bus request A. This pin is used in TIC test mode. In test mode this pin is the test bus request A input signal and is required as a dedicated device pin. During normal system operation the **MPMCTESTREQA** signal is used to request entry into the test mode. During test **MPMCTESTREQA** is used, in combination with **MPMCTESTREQB**, to indicate the type of test vector that is applied in the following cycle.
MPMCTESTREQB	Input	Pad	Test bus request B. This pin is used in TIC test mode. During test this signal is used, in combination with **MPMCTESTREQA**, to indicate the type of test vector that is applied in the following cycle. The test bus acknowledge signal gives external indication that the test bus has been granted and also indicates when a test access has completed. When **MPMCTESTACK** is LOW, the current test vector must be extended until **MPMCTESTACK** becomes HIGH.

C.3.3 Clock and reset signals

Table C-5 describes the clock and reset signals.

Table C-5 Clock and reset signal descriptions

Name	Type	Source/ destination	Description
HCLK	Input	Clock source	Bus clock. This clock times all bus transfers. All signal timings are related to the rising edge of **HCLK**.
nHCLK	Input	Clock source	Inverted **HCLK** signal that can be used to capture the read data from SDRAM.
HCLKX2	Input	Clock source	Double frequency clock that is synchronous to **HCLK**. This clock is used for pad interface DDR timings. This clock frequency is **HCLK** x 2 and is synchronous to **HCLK**. It is not required in non-DDR memory systems.

Table C-5 Clock and reset signal descriptions (continued)

Name	Type	Source/ destination	Description
nHCLKX2	Input	Clock source	Inverted double frequency clock that is synchronous to **HCLK**. This clock is used for the negative edge triggered flip-flops in DDR operations. It is not required in non-DDR memory systems.
MPMCHCLK DELAY	Input	Clock source	Delayed version of **HCLK** used in command delayed mode to ensure that SDRAM setup and hold requirements are met.
HRESETn	Input	Reset controller	Reset. The bus reset signal is active LOW and is used to reset the system and the bus.
nPOR	Input	Reset controller	Power-on reset. Active LOW.

C.3.4 DLL and self-refresh signals

Table C-6 describes the DLL and self-refresh signals.

Table C-6 DLL and self-refresh signal descriptions

Name	Type	Source/ destination	Description
MPMCDLLCALIBREQ	Output	DLL block	DLL calibration request.
MPMCDLLCALIBACK	Input	DLL block	DLL calibration acknowledge.
MPMCSREFREQ	Input	Power management unit	Self-refresh request.
MPMCSREFACK	Output	Power management unit	Self-refresh acknowledgement.

C.3.5 External Bus Interface signals

Table C-7 on page C-13 describes the *External Bus Interface* (EBI) signals.

Table C-7 EBI signal descriptions

Name	Type	Source/ destination	Description
MPMCEBIGNT	Input	EBI	The EBI grant signal goes active HIGH when the EBI grants the external memory bus.
MPMCEBIBACKOFF	Input	EBI	The EBI backoff signal goes active HIGH when the EBI wants to remove the MPMC from the memory bus so that another memory controller can be granted the memory bus.
MPMCEBIREQ	Output	EBI	The EBI request signal goes active HIGH when the MPMC requires the memory bus.

C.4 Pad interface and control signals

Table C-8 describes the pad interface and control signals.

Table C-8 Pad interface and control signal descriptions

Name	Type	Source/ destination	Description
MPMCADDROUT[27:0]	Output	Pad	Address output. Used for both static and SDRAM devices. 256Mb maximum per static memory bank. SDRAM uses bits [14:0]. Not used by NAND flash. Auto-precharge connections use the **MPMCAPOUT** output instead of bits [8] or [10] of **MPMCADDROUT**.
MPMCAPOUT	Output	Pad	SDRAM auto-precharge address bit. Micron SyncFlash and V-SyncFlash active terminate command control bit.
MPMCCKEOUT[3:0]	Output	Pad	SDRAM clock enables. Used for SDRAM devices.
MPMCCLKOUT[3:0]	Output	Pad	Positive differential clocks. Four clocks provide support for 16-bit minimum width SDRAM devices when using 64-bit wide memory. 8-bit devices are not supported for 64-bit wide memory.
MPMCDATAIN[63:0]	Input	Pad	Read data from memory. Used for the static memory controller, the dynamic memory controller and the TIC:**MPMCDATAIN[63:32]** = Read upper data word for SDRAM.**MPMCDATAIN[31:0]** = Read lower data word for SDRAM, read upper and lower data word for DDR-SDRAM, read data word for static memory and read data word for TIC. **MPMCDATAIN[7:0]** = Read data bus for NAND flash devices. 16-bit NAND flash devices also use **MPMCDATAIN[15:8]**.
MPMCDATAOUT[63:0]	Output	Pad	Data output to memory. Used for the static memory controller, the dynamic memory controller and the TIC:**MPMCDATAOUT[63:32]** = Write upper data word for SDRAM.**MPMCDATAOUT[31:0]** = Write lower data word for SDRAM, write upper and lower data word for DDR-SDRAM, write data word for static memory and write data word for TIC.**MPMCDATAOUT[7:0]** = Control, address and write data bus for NAND flash devices. 16-bit NAND flash devices also use **MPMCDATAOUT[15:8]** for write data only.

Table C-8 Pad interface and control signal descriptions (continued)

Name	Type	Source/ destination	Description
MPMCDQMOUT[7:0]	Output	Pad	Data mask output to SDRAMs. Used for SDRAM devices:**MPMCDQMOUT[7:4**] = Data mask output, one per byte, for upper word of SDRAM.**MPMCDQMOUT[3:0]** = Data mask output, one per byte, for lower word of SDRAM or upper and lower word of DDR-SDRAM.
MPMCDQSIN[3:0]	Input	Pad	Input data strobes, one per byte, for DDR-SDRAM upper or lower word.
MPMCDQSOUT[3:0]	Output	Pad	Output data strobe, one per byte, for DDR-SDRAM upper or lower word.
MPMCFBCLKIN[7:0]	Input	Pad	Positive differential fed-back clock. Used for SDRAM devices.
MPMCNDALEOUT	Output	Pad	NAND flash address latch enable.
MPMCNDCLEOUT	Output	Pad	NAND flash command latch enable.
MPMCNDREADYIN [3:0]	Input	Pad	NAND flash device ready outputs, for all four possible devices. Deasserted LOW to indicate that the device is busy. Unused inputs must be tied HIGH.
MPMCRPVHHOUT	Output	Pad	Voltage control for Micron Syncflash reset signal (RP).
nMPMCBLSOUT[3:0]	Output	Pad	Byte lane select, active LOW, for static memories. Used for static memory devices.
nMPMCCASOUT	Output	Pad	Column address strobe. Used for SDRAM devices.
nMPMCCLKOUT[3:0]	Output	Pad	Negative differential clocks. Four clocks provide support for 16-bit minimum width SDRAM devices when using 64-bit wide memory. 8-bit devices are not supported for 64-bit wide memory.

Table C-8 Pad interface and control signal descriptions (continued)

Name	Type	Source/ destination	Description
nMPMCDATAOUTEN [7:0]	Output	Pad	**nMPMCDATAOUTEN[7:4]** = Tristate I/O pad enable for the external memory data bus **MPMCDATA[63:32]**, active LOW. Enables the external memory data bus byte lanes [63:56], [55:48], [47:40], and [39:32] independently, for upper output data word for SDRAM. **nMPMCDATAOUTEN[3:0]** = Tristate I/O pad enable for the external memory data bus **MPMCDATA[31:0]**, active LOW. Enables the external memory data bus byte lanes [31:24], [23:16], [15:8], and [7:0] independently, for lower output data word for SDRAM, output data word for static memory or TIC, or upper and lower output data word for DDR-SDRAM. The lower 1 or 2 byte lanes are used for NAND flash
nMPMCDQSIN[3:0]	Input	Pad	Half-cycle delayed input data strobes, one per byte, for DDR-SDRAM upper or lower word.
nMPMCDQSOUTEN[3:0]	Output	Pad	Tristate I/O pad enable for the output data strobes, active LOW. Enables the memory device byte lanes [31:24], [23:16], [15:8], and [7:0] for DDR-SDRAM output data strobes independently.
nMPMCDYCSOUT[3:0]	Output	Pad	Active LOW chip selects for SDRAM. CS[7:4].
nMPMCDYWEOUT	Output	Pad	Write enable for dynamic memory.
nMPMCNDREOUT	Output	Pad	Read enable for NAND flash memory.
nMPMCNDWEOUT	Output	Pad	Write enable for NAND flash memory.
nMPMCOEOUT	Output	Pad	Output enable for static memories. Used for static memory devices.
nMPMCRASOUT	Output	Pad	Row address strobe. Used for SDRAM devices.
nMPMCRPOUT	Output	Pad	Reset power down to SyncFlash memory. Used for the dynamic memory controller.
nMPMCSTCSOUT[3:0]	Output	Pad	Default active LOW chip selects for static memory and NAND flash. CS[3:0].
nMPMCSTWEOUT (MPMCTESTACK)	Output	Pad	Write enable for static memory. (Test bus acknowledge signal for TIC test mode.)

C.5 Test Interface Controller (TIC) AHB signals

Table C-9 describes the TIC signals.

Table C-9 TIC signal descriptions

Name	Type	Source/ destination	Description
HADDRTIC[31:0] Address bus	Output	AHB slave layer	The 32-bit AHB address bus.
HBUSREQTIC Bus request	Output	AHB arbiter	A signal from the TIC to the bus arbiter indicates that it requires the bus.
HBURSTTIC[2:0] Burst type	Output	AHB slave layer	Indicates if the transfer forms part of a burst. The TIC always performs incrementing bursts of unspecified length.
HGRANTTIC Bus grant	Input	AHB arbiter	This signal indicates that the TIC is currently the highest priority master. Ownership of the address and control signals changes at the end of a transfer when **HREADYINTIC** is HIGH, so the TIC gains access to the bus when both **HREADYINTIC** and **HGRANTTIC** are HIGH.
HLOCKTIC Locked transfers	Output	AHB arbiter	When HIGH this signal indicates that the TIC requires locked access to the bus. No other master must be granted the bus until this signal is LOW.
HPROT[3:0] Protection control	Output	AHB slave layer	The protection control signals indicate if the transfer is an opcode fetch or data access, as well as if the transfer is a supervisor mode access or a User mode access. These signals also indicate if the current access is cacheable or bufferable.
HRDATATIC[31:0] Read data bus	Input	AHB master layer	The read data bus is used to transfer data from the AHB slave to the TIC during read operations.
HREADYINTIC Transfer done	Input	AHB slave layer	When HIGH the **HREADYINTIC** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPTIC[1:0] Transfer response	Input	AHB slave layer	The transfer response provides additional information on the status of a transfer. Four different responses are provided, OKAY, ERROR, RETRY, and SPLIT.
HSIZETIC[2:0] Transfer size	Output	AHB slave layer	Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit), or word (32-bit). The TIC does not support larger transfer sizes.

Table C-9 TIC signal descriptions (continued)

Name	Type	Source/ destination	Description
HTRANSTIC[1:0] Transfer type	Output	AHB slave layer	Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE, or BUSY. The TIC does not use the BUSY transfer type.
HWDATATIC[31:0] Write data bus	Output	AHB slave layer	The write data bus is used to transfer data from the master to the bus slaves during write operations.
HWRITETIC Transfer direction	Output	AHB slave layer	When HIGH, this signal indicates a write transfer and when LOW a read transfer.

C.6 Scan test signals

Table C-10 describes the scan test signals. It might be possible to reduce the number of scan chains required if synchronous clock domains are balanced during clock tree insertion.

Table C-10 Scan test signal descriptions

Name	Type	Source/ destination	Description
SCANENABLE	Input	Test controller	Scan enable
SCANINHCLK	Input	Test controller	Scan data input for **HCLK** scan chain
SCANOUTHCLK	Output	Test controller	Scan data output for **HCLK** scan chain
nSCANINHCLK	Input	Test controller	Scan data input for **nHCLK** scan chain
nSCANOUTHCLK	Output	Test controller	Scan data output for **nHCLK** scan chain
SCANINHCLKX2	Input	Test controller	Scan data input for **HCLKX2** scan chain
SCANOUTHCLKX2	Output	Test controller	Scan data output for **HCLKX2** scan chain
nSCANINHCLKX2	Input	Test controller	Scan data input for **nHCLKX2** scan chain
nSCANOUTHCLKX2	Output	Test controller	Scan data output for **nHCLKX2** scan chain
SCANINHCLKDELAY	Input	Test controller	Scan in for **MPMCHCLKDELAY** scan chain
SCANOUTHCLKDELAY	Output	Test controller	Scan data output for **MPMCHCLKDELAY** scan chain
SCANINFBCLKIN[7:0]	Input	Test controller	Scan in for **MPMCFBCLKIN[7:0]** scan chains
SCANOUTFBCLKIN[7:0]	Output	Test controller	Scan out for **MPMCFBCLKIN[7:0]** scan chain
SCANINDQSIN[3:0]	Input	Test controller	Scan in for **MPMCDQSIN[3:0]** scan chain
SCANOUTDQSIN{3:0]	Output	Test controller	Scan out for **MPMCDQSIN[3:0]** scan chain
nSCANINDQSIN[3:0]	Input	Test controller	Scan in for **nMPMCDQSIN[3:0]** scan chain
nSCANOUTDQSIN{3:0]	Output	Test controller	Scan out for **nMPMCDQSIN[3:0]** scan chain

Index