ARM PrimeCell™ External Bus Interface (PL220)

Revision: r0p0

Technical Reference Manual

ARM DDI 0249B

ARM PrimeCell External Bus Interface (PL220)
Technical Reference Manual

Release Information

Change history

Date	Issue	Change
31 October 2002	A	First release
12 December 2002	B	Removal of references to obsolete devices

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Product Status

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

Web Address

`http://www.arm.com`

Contents
ARM PrimeCell External Bus Interface (PL220) Technical Reference Manual

List of Tables
ARM PrimeCell External Bus Interface (PL220) Technical Reference Manual

List of Figures
ARM PrimeCell External Bus Interface (PL220) Technical Reference Manual

Preface

This preface introduces the *ARM PrimeCell External Bus Interface (PL220) Technical Reference Manual* and its user documentation. It contains the following sections:

About this book

This is the *Technical Reference Manual* (TRM) for the *ARM PrimeCell External Bus Interface* (EBI) (PL220).

Intended audience

This document has been written to help hardware and software developers of ARM processor based systems who are using the ARM PrimeCell EBI (PL220) in their design.

Using this book

This book is organized into a single chapter:

Chapter 1 ***ARM PrimeCell External Bus Interface (PL220)***

Read this chapter for details of the ARM PrimeCell EBI Interface (PL220).

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.

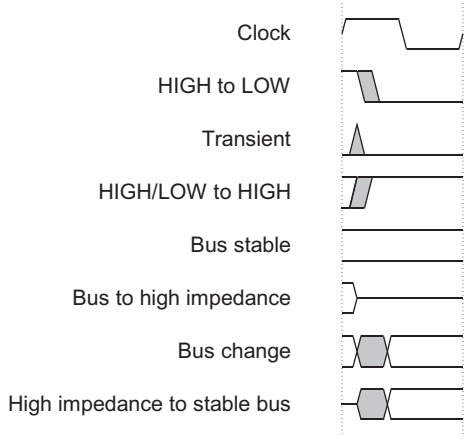
`monospace` Denotes text that can be entered at the keyboard, such as commands, file and program names, and source code.

`mono`space 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.

Timing diagram conventions

This manual contains one or more timing diagrams. The figure below 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.

Other conventions

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

- *Signals* on page xii
- *Bytes, halfwords, and words* on page xii
- *Bits, bytes, K, and M*
- *Register fields*.

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 or APB reset signals. These are named **HRESETn** and **PRESETn** respectively.

Prefix H AHB signals are prefixed by an uppercase H.

Prefix P APB signals are prefixed by an uppercase P.

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.

Further reading

This section lists publications from ARM Limited that provide additional information on developing code for the ARM family of processors.

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

ARM publications

The following publications provide information about related ARM PrimeCell and other related ARM products:

- *ARM Architecture Reference Manual* (ARM DDI 0100)
- *AMBA Specification (Rev 2.0)* (ARM IHI 0011)
- *ARM PrimeCell External Bus Interface (PL220) Integration Manual* (PL220 INTM 0000)
- *ARM PrimeCell Static Memory Controller (PL092) Technical Reference Manual* (ARM DDI 0203)
- *ARM PrimeCell Synchronous Static Memory Controller (PL093) Technical Reference Manual* (ARM DDI 0236)
- *ARM PrimeCell Multiport Memory Controller (PL172) Technical Reference Manual* (ARM DDI 0215)
- *ARM PrimeCell Multiport Memory Controller (PL175) Technical Reference Manual* (ARM DDI 0230).

Feedback

ARM Limited welcomes feedback on both the ARM PrimeCell EBI (PL220), and its documentation.

Feedback on the ARM PrimeCell External Bus Interface

If you have any comments or suggestions about this product, contact your supplier giving:

- the product name
- a concise explanation of your comments.

General suggestions for additions and improvements are also welcome.

Feedback on this document

If you have any comments on this document, send email to `errata@arm.com` giving:

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

Chapter 1
ARM PrimeCell External Bus Interface (PL220)

This chapter introduces the ARM PrimeCell EBI (PL220). It includes the following sections:

1.1 About the ARM PrimeCell EBI

In the design of an ASIC it is often necessary for different memory controllers to use the same set of external pins for data and address. This requirement can be due to several reasons such as:

Cost reduction

By sharing the same pins, the pin count and final package size of the ASIC are reduced.

Low bandwidth designs

Where dedicated buses are not required to carry volume traffic.

In the implementation of a shared pin scheme, an EBI must arbitrate between the memory controllers requiring access to the pins. Support is provided for three devices to share the external bus. For example, a *Synchronous Dynamic Random Access Memory Controller* (SDRAMC), synchronous or asynchronous *Static Memory Controller* (SMC), and *Test Interface Controller* (TIC) can be connected.

The EBI enables only the address and data signals to be shared. The control signals required for each memory type are not shared and so must be routed directly to the system pads.Memory controllers can be clocked at different rates within the system. The EBI operates with devices having clocks that are integer ratios of the fastest clock within the system.

Note

Pins can only be shared when pads and voltages match. DDR-SDRAM and Flash memory cannot share pins.

Figure 1-1 on page 1-3 shows the top-level diagram of the EBI.

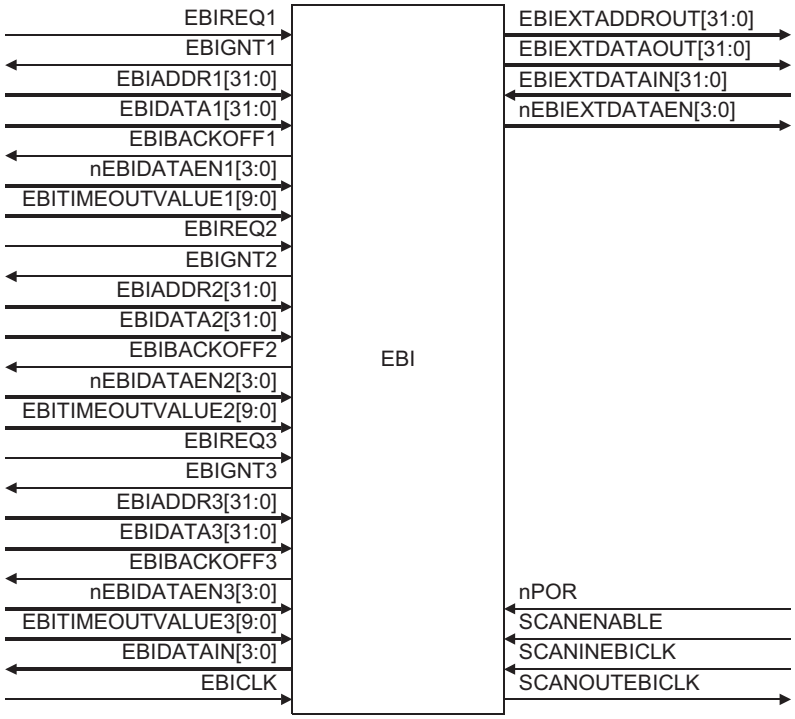


Figure 1-1 Top-level diagram of EBI

1.2 Handshake mechanism

The handshaking between the ARM PrimeCell EBI and the memory controller consists of a three-wire interface, **EBIREQ**, **EBIGNT**, and **EBIBACKOFF**, all active HIGH:

EBIREQ This signal is asserted by a memory controller to indicate that it requires external bus access.

EBIGNT The respective arbitrated **EBIGNT** is issued to the highest priority memory controller.

EBIBACKOFF

This signal is output by the EBI to inform the memory controller that it must complete the current transfer and release the bus.

The EBI arbitration scheme keeps track of the memory controller that is currently granted and waits for the transaction from that memory controller to show that it has finished by driving **EBIREQ** LOW before it grants the next memory controller. If a higher priority memory controller requests the bus, then the signal **EBIBACKOFF** is issued to ask the currently granted memory controller to terminate the current transfer as soon as possible. The priorities are determined by the time out values.When two memory controllers simultaneously request the EBI, the controller with the higher priority is granted its use.The EBI as a peripheral relies on the memory controllers to release their external requests for the external bus when they are idle, because it has no other knowledge of when a transfer starts or completes.A simple handshake example is shown in Figure 1-2. Here a device requests the external bus and is immediately granted because no other devices are requesting the bus.

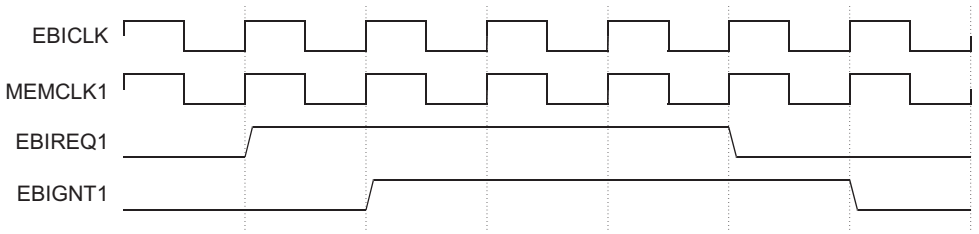


Figure 1-2 Timing of EBIREQ and EBIGNT signals, access granted immediately

If a higher priority device requests the bus, when a lower priority device is in control of the external bus, then the **EBIBACKOFF** signal is used to inform the lower priority device to release the bus as soon as possible. An example of this is shown in Figure 1-3 on page 1-5. The priorities are determined by the **EBITIMEOUTVALUE** signal, see *Control and arbitration block* on page 1-8.

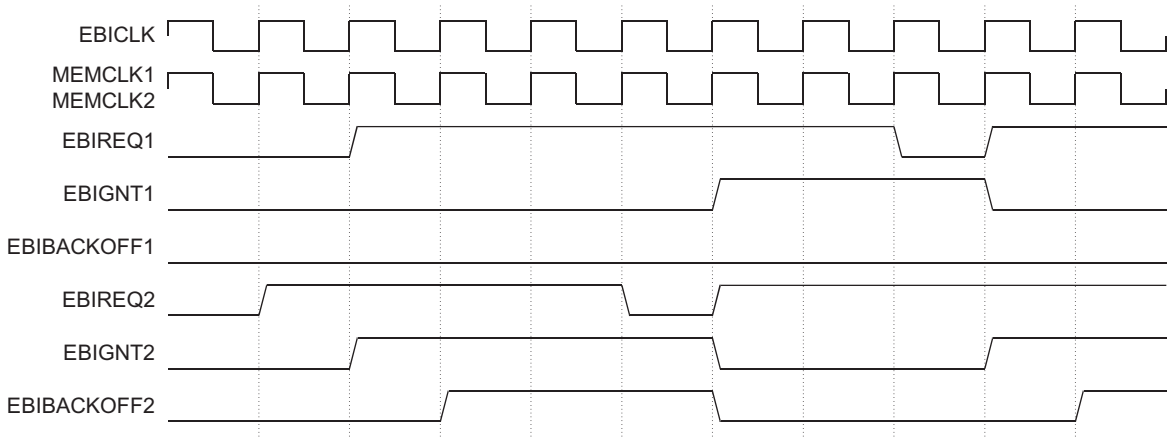


Figure 1-3 Use of EBIBACKOFF signal (EBICLK=MemCLK1=MemCLK2)

Here a device has been granted the bus, but shortly after higher priority device requests the bus. The **EBIBACKOFF** signal is used to tell the granted device to end the access early. The higher priority device is granted the bus and completes its transfer. When complete, the lower priority device is granted the bus again and completes the interrupted transfer. The **EBIREQ2** signal must be kept LOW for at least one clock cycle by the lower priority device and can then be re-asserted after this. For details of how the priority is assigned, see *Control and arbitration block* on page 1-8.

1.2.1 EBI and memory controller clocks

The ARM PrimeCell EBI supports the use of memory controllers using different frequency clocks. All of the clocks must be synchronous and an integer multiple of each other. The fastest clock must be connected to **EBICLK**. This enables a system where, for example, one memory interface operates at twice the frequency of the other. An example of such a situation is where the memory interface of an SDRAM Controller is operating at 100MHz while the memory interface of a Synchronous Static Memory Controller is operating at 50MHz. In this case, **EBICLK** is always connected to the fastest clock.

Figure 1-4 on page 1-6 illustrates the handshaking where the SDRAM controller is connected as the highest priority device.

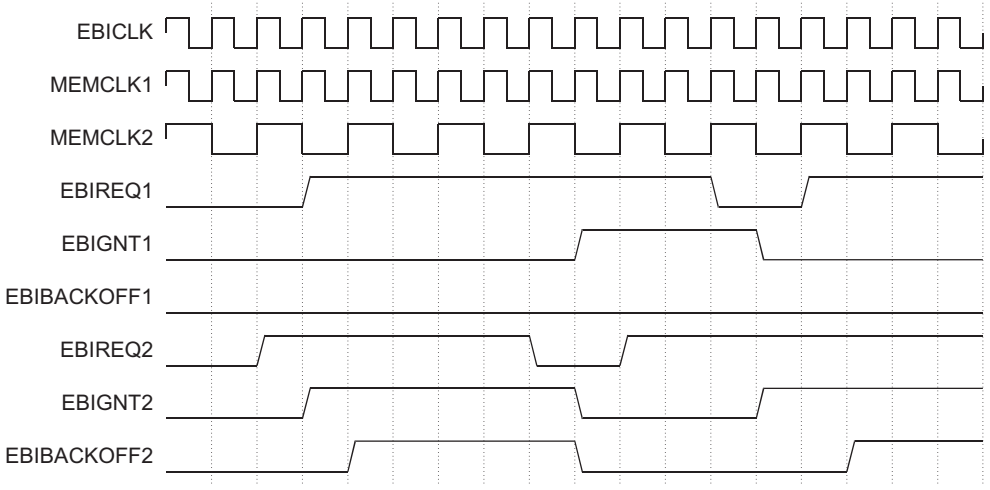


Figure 1-4 Use of EBIBACKOFF signal (EBICLK=MemCLK1 MemCLK2=1/2*EBICLK)

It is also possible for a slower device to be connected as the higher priority device. An example of the handshaking in such a case is shown in Figure 1-5.

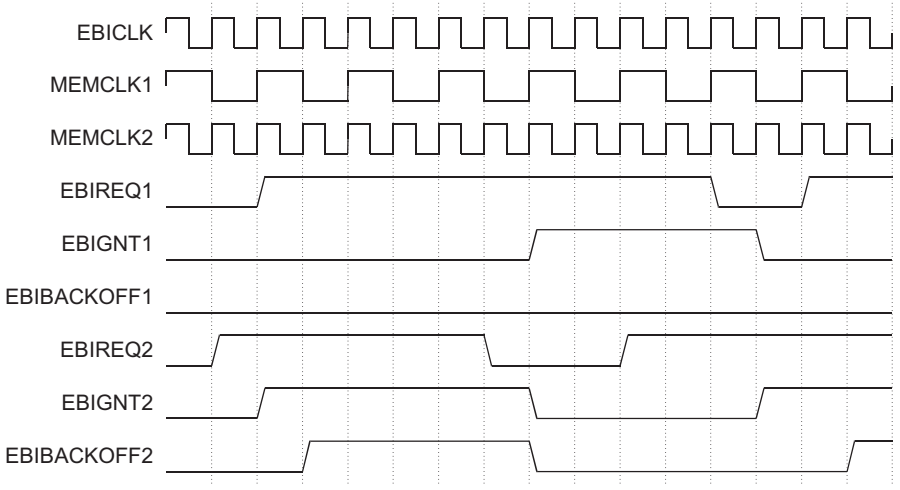


Figure 1-5 Use of EBIBACKOFF signal (EBICLK=MemCLK2 MemCLK1=1/2*EBICLK)

1.3 Structure of the ARM PrimeCell EBI module

Figure 1-6 shows the main components of the EBI module.

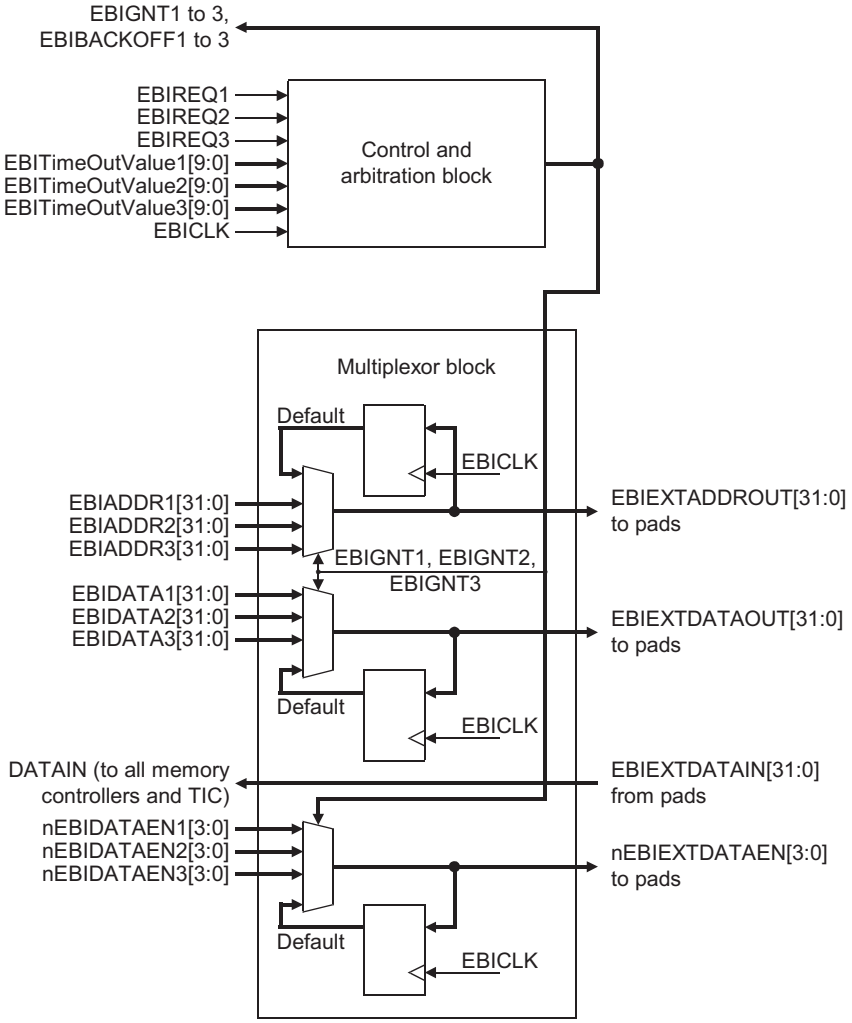


Figure 1-6 Structure of the EBI module

The clock supplied to the EBI must be the same as the fastest memory controller clock. All memory controller clocks must be synchronous. They can only be integer ratios of the fastest clock.

None of the control and clock signals for each memory controller are used by the EBI. These must be routed directly to the pads as required.

The two blocks in the EBI are described in:

- *Control and arbitration block*
- *Multiplexor block* on page 1-9.

1.3.1 Control and arbitration block

The control and arbitration block consists of two parts:

Control block

Controls the **EBIBACKOFF** and **EBIGNT** signals.

Arbitration block

Arbitrates between the requesting ports.

The two parts of the control and arbitration block are shown in more detail in Figure 1-7.

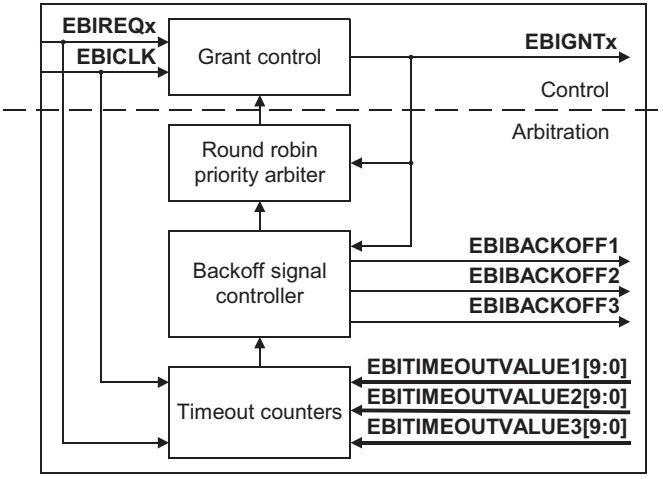


Figure 1-7 Control and arbitration block

The arbitration block arbitrates between the requesting ports. A separate block is used so that modifying a single block can change the arbitration algorithm used.

There is no fixed priority for the ports. Instead, the time-out counters are used to determine priority. If two ports time out at the same time or requests are made for the bus at the same time then a round-robin scheme is used.

Each of the three ports has a 10-bit counter that is loaded with a value from the **EBITIMEOUTVALUEx[9:0]** input whenever a request is made for the bus.

When the counter reaches 0, then the port currently granted the bus is requested to release the bus by asserting the **EBIBACKOFF** signal. The first port counter to reach 0 is given the highest priority for the bus, followed by the second device.

If the signal **EBITIMEOUTVALUE** is zero then the backoff signal is asserted immediately when a request is made.If more than one port issues a request for the bus at the same time then a round-robin arbitration scheme is used to decide which port is granted the bus.

At reset, or when no device is requesting the bus then no device is granted the bus.

1.3.2 Multiplexor block

The multiplexor block provides highly constrained combinatorial paths for address, data, and data enable signals. It also contains a set of flip-flops to store the default values when no device memory controller is selecting the bus.

The handshaking methodology used means that the signals can be used without retiming by the EBI. The retiming of the address and data signals is left to the individual memory controllers. This enables different clocking strategies to be used for each controller.

Retiming is where the external memory signals (input and output) are clocked into registers using the memory interface clock. For synchronous devices this ensures that the output signals change at the correct time and the input data is registered correctly. The data, address, and data enable are not registered inside the EBI but are instead multiplexed to the pads. This means that there is an extra delay in the output path for these signals, and allowance for this must be made when synthesizing the design. There is no multiplexor in the input data path and this is connected directly to all the memory controllers.

For more details of the retiming of the signals within the memory controllers see the technical reference manuals of the individual memory controllers. For details of the memory components timing requirements see the memory data sheets.

1.4 System-level integration of the EBI module

Figure 1-8 shows the connection of an example system with the EBI module.

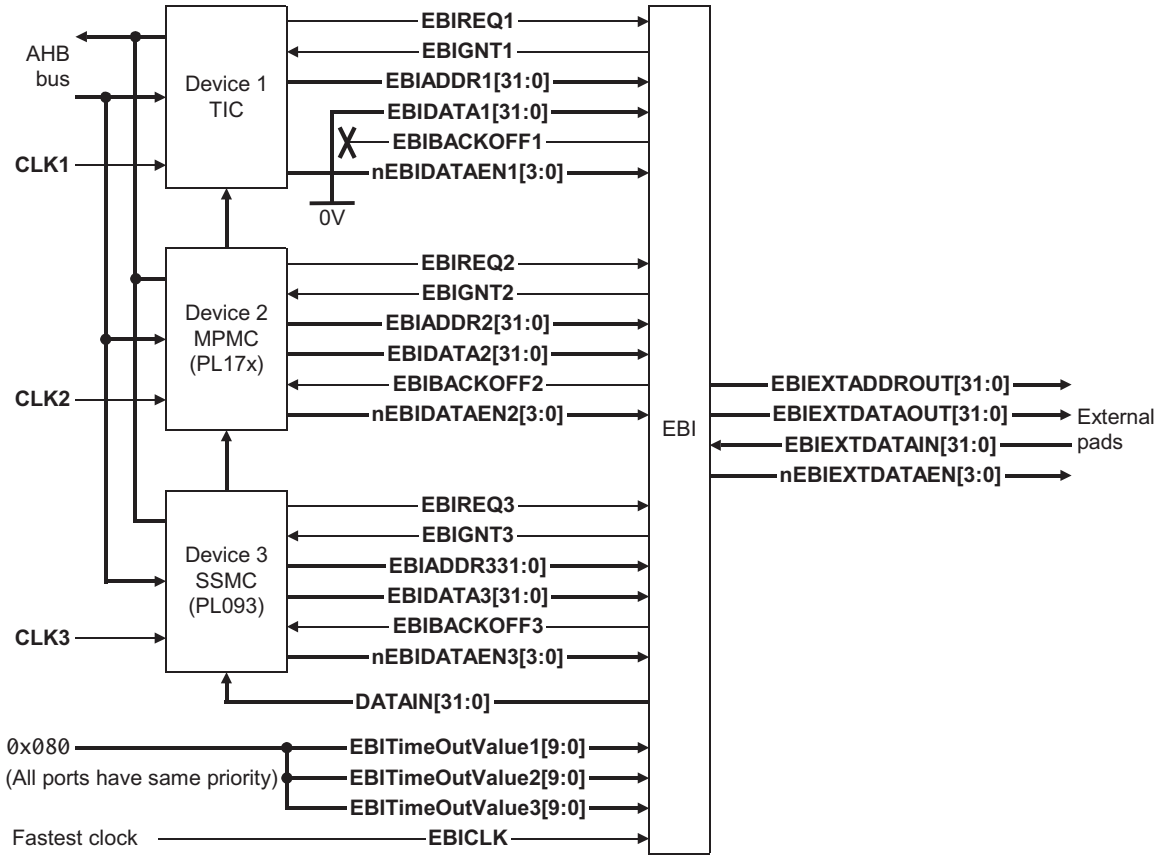


Figure 1-8 Connection of an example system with an EBI module

Note

In Figure 1-8:

- **EBIBACKOFF1** is not connected
- **EBIDATA1[31:0]** is tied LOW.

Up to three devices can be connected to the EBI as shown in Figure 1-8.

This configuration is recommended so that the TIC is granted the bus at reset. The priority of each port in this example is the same. Different values of **EBITIMEOUTVALUE** must be used to change the priorities. The TIC block is contained in PL093. This PrimeCell has additional logic to assist with interfacing to the EBI. The TIC block does not use the **EBIADDR** bus. This is tied LOW in Figure 1-8 on page 1-10 so that it is driven to a known value during testing.

Each device can be clocked by a different clock. These must be integer divisions of the fastest clock. The fastest clock must be connected to the **EBICLK**. The signal **EBIBACKOFF1** is unconnected because the TIC has no support for this functionality. No access to memory is required during TIC testing so a backoff feature is not required.

Some example configurations of memory controllers are shown in Table 1-1.

Table 1-1 Example memory controller configurations

Device 1	Device 2	Device 3
TIC (Part of PL093)	PL172 *MultiPort Memory Controller* (MPMC)	PL093 SSMC
TIC (Part of PL093)	PL175 MPMC	PL093 SSMC

1.4.1 Interfacing devices to the EBI module

Interfacing devices to the EBI module is described in:

- *SMC (PL092) interconnection notes*
- *SSMC (PL093) interconnection notes* on page 1-12
- *Interfacing a 32-bit MultiPort Memory Controller to the EBI module* on page 1-13
- *Interfacing a 64-bit MultiPort Memory Controller to the EBI module* on page 1-13

SMC (PL092) interconnection notes

To interface to a device, the memory controller clock must be the same as the EBI clock. You must generate a new grant signal for the device by using the following logic:

NewEBIGNT = EBIGNT & **~EBIBACKOFF**

If the memory clock is not the same as the EBI clock then you must ignore the **EBIBACKOFF** signal.

Figure 1-9 shows how PL092 SMC, is connected to the EBI. You must connect The remaining port on the EBI to a second device but this is not shown to simplify Figure 1-9.

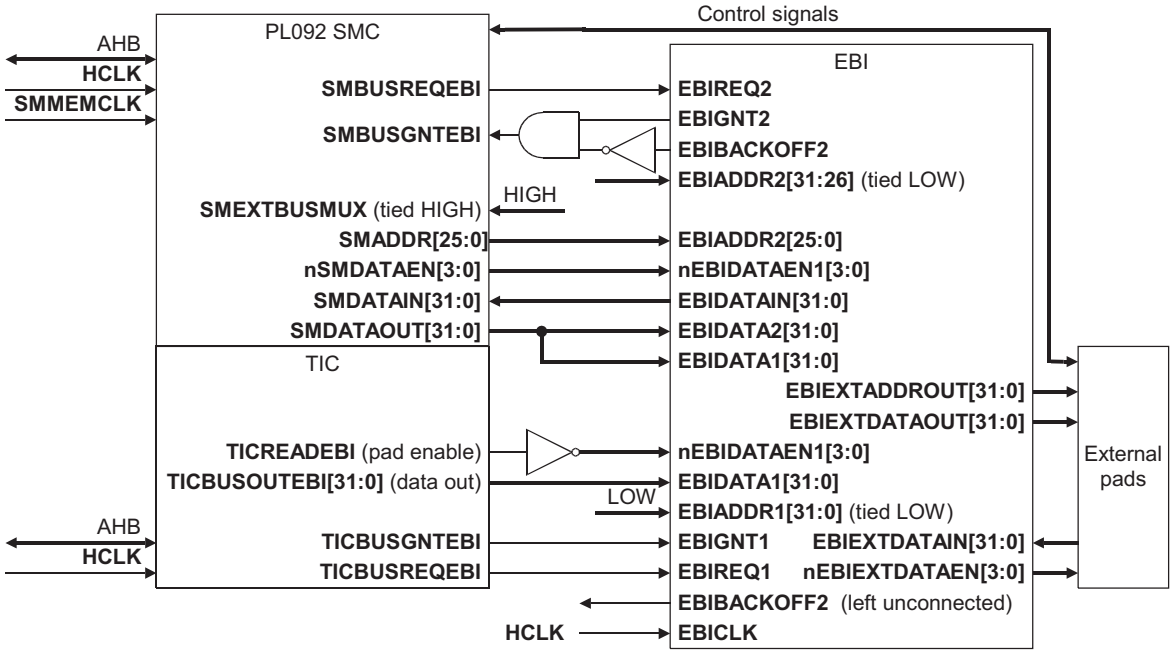


Figure 1-9 EBI connection with SMC (PL092)

SSMC (PL093) interconnection notes

1. The SSMC contains a TIC module. This shares the data signals with the SSMC.
2. The address signals are not used during TIC mode and are driven to a valid output by the SSMC.
3. There are separate Req, Gnt, and BackOff signals for the TIC module.
4. The SSMC can operate with the clock **SMMemCLK** = **HCLK**, **HCLK**/2 or **HCLK**/3. This clock is connected to the SSMC. **HCLK** is connected to the EBI and SDRAMC.
5. The **SMEXTBUSMUX** input is tied HIGH to indicate to the SSMC that the external bus interface must be used instead of the internal multiplexor (between TIC and SSMC).

Interfacing a 32-bit MultiPort Memory Controller to the EBI module

You can connect these devices as shown in Figure 1-8 on page 1-10.

Interfacing a 64-bit MultiPort Memory Controller to the EBI module

To make full use of the SDRAM refreshing features of the PL176 MPMC, the **MPMCAPOUT** output must bypass the EBI and be connected directly to the auto precharge input on the address bus of the SDRAM device. This is address bit A8 for 32-bit DDR devices, and address bit A10 for all other SDRAM devices. All other address input bits of the device must be connected as normal to the external address bus.

Note

The whole of the address bus must be driven through the EBI. This is because address bits A10 and A8 are still used for accesses to other memory devices.

This extra auto precharge connection allows SDRAM devices to be refreshed by the PL176 MPMC without being granted control of the external address bus through the EBI.

If a 64-bit data bus is used, then you must connect the upper 32 bits directly to the external memory bus. This is because the EBI only controls the lower 32 bits.

If all 64 bits of the data bus must be shared then you must modify the EBI appropriately by extending the width of the data bus. See the *ARM PrimeCell External Bus Interface (PL220) Integration Manual* for more information.

1.5 Signal descriptions

Table 1-2 lists the ARM PrimeCell EBI signals.

Table 1-2 EBI signals

Signal name	Type	Source/destination	Description
EBIADDR1[31:0]	Input	Device 1	EBI address for port 1.
EBIADDR2[31:0]	Input	Device 2	EBI address for port 2.
EBIADDR3[31:0]	Input	Device 3	EBI address for port 3.
EBIBACKOFF1	Output	Device 1	Signals that the current transfer must be completed as soon as possible. Active HIGH.
EBIBACKOFF2	Output	Device 2	Signals that the current transfer must be completed as soon as possible. Active HIGH.
EBIBACKOFF3	Output	Device 3	Signals that the current transfer must be completed as soon as possible. Active HIGH.
EBICLK	Input	System clock controller	EBI clock. Fastest of all the clocks connected to the memory controllers/devices.
EBIDATA1[31:0]	Input	Device 1	EBI data output for port 1.
EBIDATA2[31:0]	Input	Device 2	EBI data output for port 2.
EBIDATA3[31:0]	Input	Device 3	EBI data output for port 3.
EBIDATAIN[31:0]	Output	To all devices.	Data in. Connected to all devices.
EBIEXTADDROUT[31:0]	Output	Pads	Address output to pads.
nEBIEXTDATAEN[3:0]	Output	Pads	Data enable to data pads.
EBIEXTDATAIN[31:0]	Input	Pads	Data input from data pads.
EBIEXTDATAOUT[31:0]	Output	Pads	Data output to data pads.
EBIGNT1	Output	Device 1	EBI grant for port 1. Active HIGH.
EBIGNT2	Output	Device 2	EBI grant for port 2. Active HIGH.
EBIGNT3	Output	Device 3	EBI grant for port 3. Active HIGH.
EBIREQ1	Input	Device 1	EBI request for port 1. Active HIGH.
EBIREQ2	Input	Device 2	EBI request for port 2. Active HIGH.

Table 1-2 EBI signals (continued)

Signal name	Type	Source/destination	Description
EBIREQ3	Input	Device 3	EBI request for port 3. Active HIGH.
EBITIMEOUTVALUE1[9:0]	Input	System controller	Value to be loaded into timeout counter for port 1. Can be tied off to a fixed value or supplied by the system controller.
EBITIMEOUTVALUE2[9:0]	Input	System controller	Value to be loaded into timeout counter for port 2. Can be tied off to a fixed value or supplied by the system controller.
EBITIMEOUTVALUE3[9:0]	Input	System controller	Value to be loaded into timeout counter for port 3. Can be tied off to a fixed value or supplied by the system controller.
nEBIDATAEN1[3:0]	Input	Device 1	Data enable for port 1.
nEBIDATAEN2[3:0]	Input	Device 2	Data enable for port 2.
nEBIDATAEN3[3:0]	Input	Device 3	Data enable for port 3.
nPOR	Input	Reset controller	Active LOW reset.
SCANENABLE	Input	System controller	Dummy pin for use as a scan enable input.
SCANINEBICLK	Input	System controller	Dummy pin for use as a dedicated **EBICLK** scan chain input.
SCANOUTEBICLK	Output	System controller	Dummy pin for use as a dedicated **EBICLK** scan chain output.

Index

R

S

T

W

