AMBA Peripheral Bus Controller

Data Sheet
AMBA Peripheral Bus Controller
Data Sheet

Copyright © 1997 Advanced RISC Machines Ltd (ARM). All rights reserved.

Release Information

<table>
<thead>
<tr>
<th>Issue</th>
<th>Date</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Dec 1995</td>
<td>Created</td>
</tr>
<tr>
<td>B</td>
<td>Jan 1996</td>
<td>Minor edits</td>
</tr>
<tr>
<td>C</td>
<td>Apr 1997</td>
<td>Minor edits</td>
</tr>
</tbody>
</table>

Proprietary Notice

ARM and the ARM Powered logo are trademarks of Advanced RISC Machines Ltd.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM in good faith. However, all warranties implied or expressed, including but not limited to implied warranties or merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM Ltd shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

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

AMBA Peripheral Bus Controller Data Sheet

Chapter 1 AMBA Peripheral Bus Controller

1.1 Overview ..................................................................................................... 1-2
1.2 Hardware Interface and Signal Description ................................................. 1-3
1.3 Peripheral Memory Map .............................................................................. 1-8
1.4 Function and Operation of Block ................................................................. 1-9
Chapter 1
AMBA Peripheral Bus Controller

This module converts Advanced System Bus (ASB) signals to Advanced Peripheral Bus (APB) signals.

- Overview on page 1-2
- Hardware Interface and Signal Description on page 1-3
- Peripheral Memory Map on page 1-8
- Function and Operation of Block on page 1-9
1.1 Overview

This module provides an interface between the Advanced System Bus (ASB) and the Advanced Peripheral Bus (APB).

The implementation of this block contains:
- a state machine, which is independent of the device memory map
- combinatorial address decoding logic to produce PSELxx signals

To add new peripherals or alter the system memory map only this section needs to be modified.
1.2 Hardware Interface and Signal Description

This module converts ASB transactions to APB transactions, as described in the following tables:

- Table 1-1 describes the ASB signals used in this module.
- Table 1-2 on page 1-4, describes the APB signals produced.
- Table 1-3 on page 1-4, describes the signals produced by the APB address decoding sub-module.

### Table 1-1 ASB signal descriptions

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Source/Destination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCLK</td>
<td>In</td>
<td>System (bus) clock</td>
<td>System (bus) clock. This clock times all bus transfers. The clock has two distinct phases—phase 1 in which BCLK is LOW and phase 2 in which BCLK is HIGH.</td>
</tr>
<tr>
<td>BD[31:0]</td>
<td>InOut</td>
<td>Bus master</td>
<td>This is the bidirectional system data bus. The data bus is driven by the current bus master during write transfers, and by this block during read transfers from the peripheral bus.</td>
</tr>
<tr>
<td>BnRES</td>
<td>In</td>
<td>Reset controller</td>
<td>This active LOW signal indicates the reset status of the bus and is driven by the reset controller.</td>
</tr>
<tr>
<td>BWAIT</td>
<td>Out</td>
<td>System decoder and current bus master</td>
<td>This signal is driven by the selected bus slave to indicate if the current transfer may complete. If BWAIT is HIGH, a further bus cycle is required. If BWAIT is LOW, the transfer may complete in the current bus cycle. When no bus transfer is taking place, this signal is driven by the system decoder. When selected, the peripheral bus controller drives it in the LOW phase of BCLK and it is valid set up to the rising edge of BCLK.</td>
</tr>
<tr>
<td>BLAST</td>
<td>Out</td>
<td>System decoder and current bus master</td>
<td>This signal is driven by the selected bus slave to indicate if the current transfer should be the last of a burst sequence. It is always driven low. When no bus transfer is taking place, this signal is driven by the bus decoder. When selected, the peripheral bus controller drives it in the LOW phase of BCLK and it is valid set up to the rising edge of BCLK.</td>
</tr>
<tr>
<td>DSEL</td>
<td>In</td>
<td>From Bus Decoder</td>
<td>This signal indicates that the peripheral bus controller has been selected. It becomes valid during the BCLK HIGH phase before the data transfer and remains active until the last BCLK HIGH phase of the transfer.</td>
</tr>
</tbody>
</table>
A transfer error is indicated by the selected bus slave using the BERROR signal. When BERROR is HIGH, a transfer error has occurred. When BERROR is LOW, the transfer is successful.

When no bus transfer is taking place, it is driven by the system decoder. When selected, the peripheral bus controller drives this signal in the LOW phase of BCLK and it is valid set up to the rising edge of BCLK.

This signal indicates a write cycle when HIGH and a read cycle when LOW. It has the same timing as the address bus and is driven by the bus master.

This is the peripheral address bus, which is used by individual peripherals for decoding register accesses to that peripheral. The addresses become valid before PSTB goes HIGH and remain valid after PSTB goes LOW.

This is the bidirectional peripheral data bus. The data bus is driven by this block during write cycles (when PWRITE is HIGH) and by the selected peripheral bus slave during read cycles (when PWRITE is LOW).

This strobe signal is used to time all accesses on the peripheral bus. The falling edge of PSTB is coincident with the falling edge of BCLK.

This signal indicates a write to a peripheral when HIGH and a read from a peripheral when LOW. It has the same timing as the peripheral address bus. It becomes valid before PSTB goes HIGH and remains valid after PSTB goes LOW.
There is one of these signals for each APB peripheral present in the system. The signal indicates that the slave device is selected and a data transfer is required. This signal has the same timing as the peripheral address bus. It becomes valid before PSTB goes HIGH and remains valid after PSTB goes LOW.

**Figure 1-2 APB write cycle**
Figure 1-3 APB read cycle
Figure 1-4 APB error
1.3 Peripheral Memory Map

The bridge controls the memory map for the peripherals and generates a select signal for each peripheral.

![Peripheral Memory Map Diagram]

Figure 1-5 Peripheral Memory Map
1.4 Function and Operation of Block

The APB bridge responds to transaction requests from the currently enabled bus master. The ASB transactions are converted into APB transactions. The state machine, see Figure 1-6, controls the depipelining of the ASB transaction, and controls the latches and drivers for the PA, PD and BD buses, also producing the PSTB signal. This example design uses the DSEL signal from a centralised decoder to select the Peripheral bus controller as an ASB slave.

The individual PSELxx signals are decoded from BA, using the state of the state machine to enable their output.

If an undefined location is accessed then BERROR is asserted and no peripheral is selected, see Figure 1-4 on page 1-7.

![Figure 1-6 State machine for APB controller](image)

The signal APBError is internal, and is asserted when the address points to an undefined APB area.