AMBA® APB Protocol

Version: 2.0

Specification

ARM IHI 0024C (ID042910)

AMBA APB Protocol
Specification

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
25 September 2003	A	Non-Confidential	First release for v1.0
17 August 2004	B	Non-Confidential	Second release for v1.0
13 April 2010	C	Non-Confidential	First release for v2.0

Proprietary Notice

Words and logos marked with ® or ™ are registered trademarks or trademarks of ARM® in the EU and other countries, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

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

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

This document is intended only to assist the reader in the use of the product. ARM 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.

Where the term ARM is used it means “ARM or any of its subsidiaries as appropriate”.

ARM AMBA Specification Licence

THIS END USER LICENCE AGREEMENT (“LICENCE”) IS A LEGAL AGREEMENT BETWEEN YOU (EITHER A SINGLE INDIVIDUAL, OR SINGLE LEGAL ENTITY) AND ARM LIMITED (“ARM”) FOR THE USE OF THE RELEVANT AMBA SPECIFICATION ACCOMPANYING THIS LICENCE. ARM IS ONLY WILLING TO LICENSE THE RELEVANT AMBA SPECIFICATION TO YOU ON CONDITION THAT YOU ACCEPT ALL OF THE TERMS IN THIS LICENCE. BY CLICKING “I AGREE” OR OTHERWISE USING OR COPYING THE RELEVANT AMBA SPECIFICATION YOU INDICATE THAT YOU AGREE TO BE BOUND BY ALL THE TERMS OF THIS LICENCE. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENCE, ARM IS UNWILLING TO LICENSE THE RELEVANT AMBA SPECIFICATION TO YOU AND YOU MAY NOT USE OR COPY THE RELEVANT AMBA SPECIFICATION AND YOU SHOULD PROMPTLY RETURN THE RELEVANT AMBA SPECIFICATION TO ARM.

“LICENSEE” means You and your Subsidiaries.

“Subsidiary” means, if You are a single entity, any company the majority of whose voting shares is now or hereafter owned or controlled, directly or indirectly, by You. A company shall be a Subsidiary only for the period during which such control exists.

1. Subject to the provisions of Clauses 2, 3 and 4, ARM hereby grants to LICENSEE a perpetual, non-exclusive, non-transferable, royalty free, worldwide licence to:

(i) use and copy the relevant AMBA Specification for the purpose of developing and having developed products that comply with the relevant AMBA Specification;

(ii) manufacture and have manufactured products which either: (a) have been created by or for LICENSEE under the licence granted in Clause 1(i); or (b) incorporate a product(s) which has been created by a third party(s) under a licence granted by ARM in Clause 1(i) of such third party’s ARM AMBA Specification Licence; and

(iii) offer to sell, sell, supply or otherwise distribute products which have either been (a) created by or for LICENSEE under the licence granted in Clause 1(i); or (b) manufactured by or for LICENSEE under the licence granted in Clause 1(ii).

2. LICENSEE hereby agrees that the licence granted in Clause 1 is subject to the following restrictions:

(i) where a product created under Clause 1(i) is an integrated circuit which includes a CPU then either: (a) such CPU shall only be manufactured under licence from ARM; or (b) such CPU is neither substantially compliant with nor marketed as being compliant with the ARM instruction sets licensed by ARM from time to time;

(ii) the licences granted in Clause 1(iii) shall not extend to any portion or function of a product that is not itself compliant with part of the relevant AMBA Specification; and

(iii) no right is granted to LICENSEE to sublicense the rights granted to LICENSEE under this Agreement.

3. Except as specifically licensed in accordance with Clause 1, LICENSEE acquires no right, title or interest in any ARM technology or any intellectual property embodied therein. In no event shall the licences granted in accordance with Clause 1 be construed as granting LICENSEE, expressly or by implication, estoppel or otherwise, a licence to use any ARM technology except the relevant AMBA Specification.

4. THE RELEVANT AMBA SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES EXPRESS, IMPLIED OR STATUTORY, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF SATISFACTORY QUALITY, MERCHANTABILITY, NONINFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE.

5. No licence, express, implied or otherwise, is granted to LICENSEE, under the provisions of Clause 1, to use the ARM tradename, or AMBA trademark in connection with the relevant AMBA Specification or any products based thereon. Nothing in Clause 1 shall be construed as authority for LICENSEE to make any representations on behalf of ARM in respect of the relevant AMBA Specification.

6. This Licence shall remain in force until terminated by you or by ARM. Without prejudice to any of its other rights if LICENSEE is in breach of any of the terms and conditions of this Licence then ARM may terminate this Licence immediately upon giving written notice to You. You may terminate this Licence at any time. Upon expiry or termination of this Licence by You or by ARM LICENSEE shall stop using the relevant AMBA Specification and destroy all copies of the relevant AMBA Specification in your possession together with all documentation and related materials. Upon expiry or termination of this Licence, the provisions of clauses 6 and 7 shall survive.

7. The validity, construction and performance of this Agreement shall be governed by English Law.

ARM contract references: LEC-PRE-00490-V4.0 ARM AMBA Specification Licence

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 APB Protocol Specification

List of Tables
AMBA APB Protocol Specification

List of Figures
AMBA APB Protocol Specification

Preface

This preface introduces the *AMBA APB Protocol Specification*. It contains the following sections:

- *About this book* on page viii
- *Feedback* on page x.

About this book

This book is for the AMBA APB Protocol Specification.

Intended audience

This book is written for hardware and software engineers who want to become familiar with the *Advanced Microcontroller Bus Architecture* (AMBA) *Advanced Peripheral Bus* (APB) protocol.

Using this book

This book is organized into the following chapters:

Chapter 1 *Introduction*

Read this for an overview of the APB protocol.

Chapter 2 *Signal Descriptions*

Read this for descriptions of the APB signals.

Chapter 3 *Transfers*

Read this for information about the different types of APB transfer.

Chapter 4 *Operating States*

Read this for descriptions of the APB operating states.

Appendix A *Revisions*

Read this for a description of the technical changes between released issues of this book.

Conventions

Conventions that this book can use are described in:
- *Typographical*
- *Timing diagrams* on page ix
- *Signals* on page ix.

Typographical

The typographical conventions are:

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

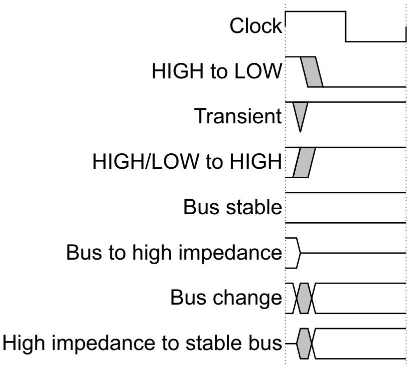
< and > Enclose replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>
```

Timing diagrams

The figure named *Key to timing diagram conventions* explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

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.



Key to timing diagram conventions

Timing diagrams sometimes show single-bit signals as HIGH and LOW at the same time and they look similar to the bus change shown in *Key to timing diagram conventions*. If a timing diagram shows a single-bit signal in this way then its value does not affect the accompanying description.

Signals

The signal conventions are:

Signal level The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:

- HIGH for active-HIGH signals
- LOW for active-LOW signals.

Lower-case n At the start or end of a signal name denotes an active-LOW signal.

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, `http://infocenter.arm.com`, for access to ARM documentation.

ARM publications

This book contains information that is specific to this product. See the following documents for other relevant information:

- *AMBA AXI Protocol Specification* (ARM IHI 0022)

Feedback

ARM welcomes feedback on this product and its documentation.

Feedback on this product

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

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to `errata@arm.com`. Give:

- the title, AMBA APB Protocol Specification
- the number, ARM IHI 0024C
- the page numbers to which your comments apply
- a concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Chapter 1
Introduction

This chapter provides an overview of the APB protocol. It contains the following sections:

- *About the APB protocol* on page 1-2
- *APB revisions* on page 1-3.

1.1 About the APB protocol

The *Advanced Peripheral Bus* (APB) is part of the *Advanced Microcontroller Bus Architecture* (AMBA) protocol family. It defines a low-cost interface that is optimized for minimal power consumption and reduced interface complexity.

The APB protocol is not pipelined, use it to connect to low-bandwidth peripherals that do not require the high performance of the AXI protocol.

The APB protocol relates a signal transition to the rising edge of the clock, to simplify the integration of APB peripherals into any design flow. Every transfer takes at least two cycles.

The APB can interface with:

- AMBA *Advanced High-performance Bus* (AHB)
- AMBA *Advanced High-performance Bus Lite* (AHB-Lite)
- AMBA *Advanced Extensible Interface* (AXI)
- AMBA *Advanced Extensible Interface Lite* (AXI4-Lite)

You can use it to access the programmable control registers of peripheral devices.

1.2 APB revisions

The *APB Specification Rev E*, released in 1998, is now obsolete and is superseded by the following three revisions:

- AMBA 2 APB Specification
- AMBA 3 APB Protocol Specification v1.0
- AMBA APB Protocol Specification v2.0.

1.2.1 AMBA 2 APB Specification

The AMBA 2 APB Specification is detailed in *AMBA Specification Rev 2* (ARM IHI 0011A).

This specification defines the interface signals, the basic read and write transfers, and the two APB components the APB bridge and the APB slave.

This version of the specification is referred to as APB2.

1.2.2 AMBA 3 APB Protocol Specification v1.0

The *AMBA 3 APB Protocol Specification v1.0* defines the following additional functionality:

- Wait states. See Chapter 3 *Transfers*.
- Error reporting. See *Error response* on page 3-6.

The following interface signals support this functionality:

PREADY	A ready signal to indicate completion of an APB transfer.
PSLVERR	An error signal to indicate the failure of a transfer.

This version of the specification is referred to as APB3.

1.2.3 AMBA APB Protocol Specification v2.0

The *AMBA APB Protocol Specification v2.0* defines the following additional functionality:

- Transaction protection. See *Protection unit support* on page 3-8.
- Sparse data transfer. See *Write strobes* on page 3-4.

The following interface signals support this functionality:

PPROT	A protection signal to support both non-secure and secure transactions on APB.
PSTRB	A write strobe signal to enable sparse data transfer on the write data bus.

This version of the specification is referred to as APB4.

Chapter 2
Signal Descriptions

This chapter describes the AMBA APB signals. It contains the following section:

- *AMBA APB signals* on page 2-2.

2.1 AMBA APB signals

Table 2-1 lists the APB signals.

Table 2-1 APB signal descriptions

Signal	Source	Description
PCLK	Clock source	Clock. The rising edge of **PCLK** times all transfers on the APB.
PRESETn	System bus equivalent	Reset. The APB reset signal is active LOW. This signal is normally connected directly to the system bus reset signal.
PADDR	APB bridge	Address. This is the APB address bus. It can be up to 32 bits wide and is driven by the peripheral bus bridge unit.
PPROT	APB bridge	Protection type. This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access.
PSELx	APB bridge	Select. The APB bridge unit generates this signal to each peripheral bus slave. It indicates that the slave device is selected and that a data transfer is required. There is a **PSELx** signal for each slave.
PENABLE	APB bridge	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
PWRITE	APB bridge	Direction. This signal indicates an APB write access when HIGH and an APB read access when LOW.
PWDATA	APB bridge	Write data. This bus is driven by the peripheral bus bridge unit during write cycles when **PWRITE** is HIGH. This bus can be up to 32 bits wide.
PSTRB	APB bridge	Write strobes. This signal indicates which byte lanes to update during a write transfer. There is one write strobe for each eight bits of the write data bus. Therefore, **PSTRB[n]** corresponds to **PWDATA[(8n + 7):(8n)]**. Write strobes must not be active during a read transfer.
PREADY	Slave interface	Ready. The slave uses this signal to extend an APB transfer.
PRDATA	Slave interface	Read Data. The selected slave drives this bus during read cycles when **PWRITE** is LOW. This bus can be up to 32-bits wide.
PSLVERR	Slave interface	This signal indicates a transfer failure. APB peripherals are not required to support the **PSLVERR** pin. This is true for both existing and new APB peripheral designs. Where a peripheral does not include this pin then the appropriate input to the APB bridge is tied LOW.

2.1.1 Data buses

The APB protocol has two independent data buses, one for read data and one for write data. The buses can be up to 32 bits wide. Because the buses do not have their own individual handshake signals, it is not possible for data transfers to occur on both buses at the same time.

Chapter 3
Transfers

This chapter describes typical AMBA APB transfers, the error response, and protection unit support. It contains the following sections:

- *Write transfers* on page 3-2
- *Write strobes* on page 3-4
- *Read transfers* on page 3-5
- *Error response* on page 3-6.
- *Protection unit support* on page 3-8

3.1 Write transfers

This section describes the following types of write transfer:

- *With no wait states*
- *With wait states*.

3.1.1 With no wait states

Figure 3-1 shows a basic write transfer with no wait states.

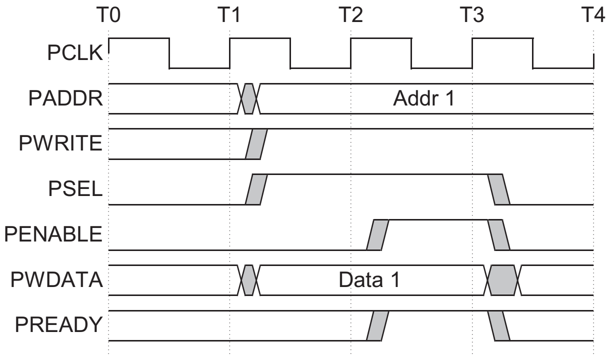


Figure 3-1 Write transfer with no wait states

At `T1`, a write transfer starts with address **PADDR**, write data **PWDATA**, write signal **PWRITE**, and select signal **PSEL,** being registered at the rising edge of **PCLK**. This is called the Setup phase of the write transfer.

At `T2`, enable signal **PENABLE**, and ready signal **PREADY**, are registered at the rising edge of **PCLK**.

When asserted, **PENABLE** indicates the start of the Access phase of the transfer.

When asserted, **PREADY** indicates that the slave can complete the transfer at the next rising edge of **PCLK**.

The address **PADDR**, write data **PWDATA**, and control signals all remain valid until the transfer completes at `T3`, the end of the Access phase.

The enable signal **PENABLE**, is deasserted at the end of the transfer. The select signal **PSEL**, is also deasserted unless the transfer is to be followed immediately by another transfer to the same peripheral.

3.1.2 With wait states

Figure 3-2 on page 3-3 shows how the slave can use the **PREADY** signal to extend the transfer. During an Access phase, when **PENABLE** is HIGH, the slave extends the transfer by driving **PREADY** LOW. The following signals remain unchanged while **PREADY** remains LOW:

- address, **PADDR**
- write signal, **PWRITE**
- select signal, **PSEL**
- enable signal, **PENABLE**
- write data, **PWDATA**
- write strobes, **PSTRB**
- protection type, **PPROT.**

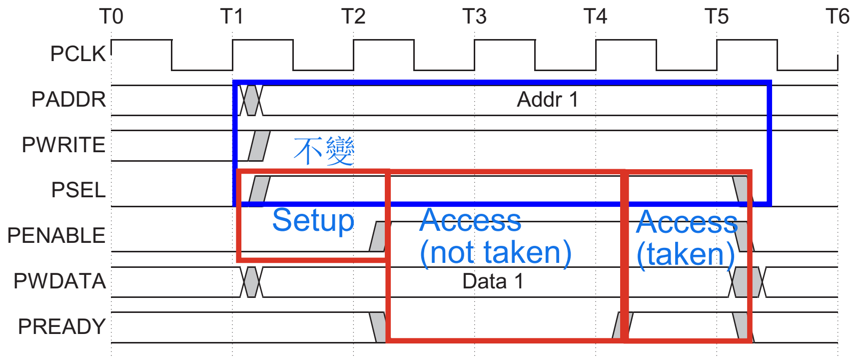


Figure 3-2 Write transfer with wait states

PREADY can take any value when **PENABLE** is LOW. This ensures that peripherals that have a fixed two cycle access can tie **PREADY** HIGH.

Note

It is recommended that the address and write signals are not changed immediately after a transfer, but remain stable until another access occurs. This reduces power consumption.

3.2 Write strobes

The write strobe signals, **PSTRB**, enable sparse data transfer on the write data bus. Each write strobe signal corresponds to one byte of the write data bus. When asserted HIGH, a write strobe indicates that the corresponding byte lane of the write data bus contains valid information.

There is one write strobe for each eight bits of the write data bus, so **PSTRB[n]** corresponds to **PWDATA[(8n + 7):(8n)]**. Figure 3-3 shows this relationship on a 32-bit data bus.

31 24	23 16	15 8	7 0
PSTRB[3]	PSTRB[2]	PSTRB[1]	PSTRB[0]

Figure 3-3 Byte lane mapping

Note

For read transfers the bus master must drive all bits of **PSTRB** LOW.

3.3 Read transfers

Two types of read transfer are described in this section:

- *With no wait states*
- *With wait states*.

3.3.1 With no wait states

Figure 3-4 shows a read transfer. The timing of the address, write, select, and enable signals are as described in *Write transfers* on page 3-2. The slave must provide the data before the end of the read transfer.

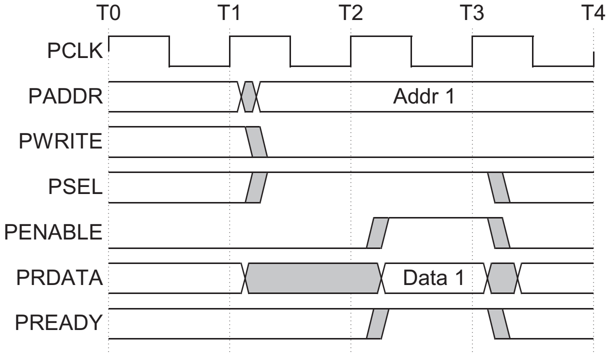


Figure 3-4 Read transfer with no wait states

3.3.2 With wait states

Figure 3-5 shows how the **PREADY** signal can extend the transfer. The transfer is extended if **PREADY** is driven LOW during an Access phase. The protocol ensures that the following remain unchanged for the additional cycles:

- address, **PADDR**
- write signal, **PWRITE**
- select signal, **PSEL**
- enable signal, **PENABLE**
- protection type, **PPROT**.

Figure 3-5 shows that two cycles are added using the **PREADY** signal. However, you can add any number of additional cycles, from zero upwards.

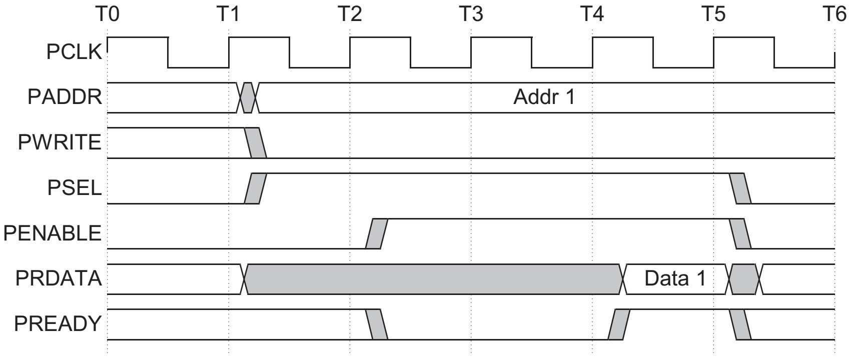


Figure 3-5 Read transfer with wait states

3.4 Error response

You can use **PSLVERR** to indicate an error condition on an APB transfer. Error conditions can occur on both read and write transactions.

PSLVERR is only considered valid during the last cycle of an APB transfer, when **PSEL**, **PENABLE**, and **PREADY** are all HIGH.

It is recommended, but not mandatory, that you drive **PSLVERR** LOW when it is not being sampled. That is, when any of **PSEL**, **PENABLE**, or **PREADY** are LOW.

Transactions that receive an error, might or might not have changed the state of the peripheral. This is peripheral-specific and either is acceptable. When a write transaction receives an error this does not mean that the register within the peripheral has not been updated. Read transactions that receive an error can return invalid data. There is no requirement for the peripheral to drive the data bus to all 0s for a read error.

APB peripherals are not required to support the **PSLVERR** pin. This is true for both existing and new APB peripheral designs. Where a peripheral does not include this pin then the appropriate input to the APB bridge is tied LOW.

3.4.1 Write transfer

Figure 3-6 shows an example of a failing write transfer that completes with an error.

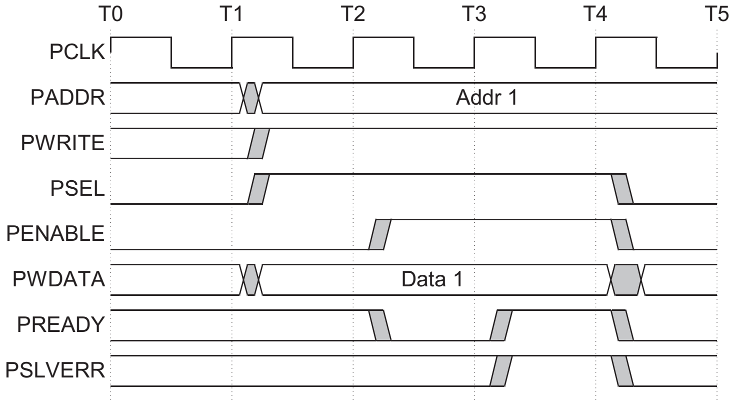


Figure 3-6 Example failing write transfer

3.4.2 Read transfer

A read transfer can also complete with an error response, indicating that there is no valid read data available. Figure 3-7 shows a read transfer completing with an error response.

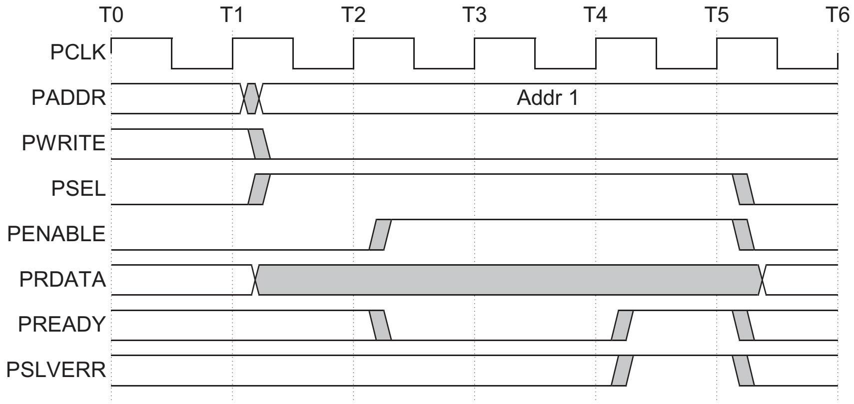


Figure 3-7 Example failing read transfer

3.4.3 Mapping of PSLVERR

When bridging:

From AXI to APB An APB error is mapped back to **RRESP**/**BRESP** = SLVERR. This is achieved by mapping **PSLVERR** to the AXI signals **RRESP[1]** for reads and **BRESP[1]** for writes.

From AHB to APB **PSLVERR** is mapped back to **HRESP** = ERROR for both reads and writes. This is achieved by mapping **PSLVERR** to the AHB signal **HRESP[0]**.

3.5 Protection unit support

To support complex system designs, it is often necessary for both the interconnect and other devices in the system to provide protection against illegal transactions. For the APB interface, this protection is provided by the **PPROT[2:0]** signals.

The three levels of access protection are:

Normal or privileged, PPROT[0]

- LOW indicates a normal access
- HIGH indicates a privileged access.

This is used by some masters to indicate their processing mode. A privileged processing mode typically has a greater level of access within a system.

Secure or non-secure, PPROT[1]

- LOW indicates a secure access
- HIGH indicates a non-secure access.

This is used in systems where a greater degree of differentiation between processing modes is required.

Note

This bit is configured so that when it is HIGH then the transaction is considered non-secure and when LOW, the transaction is considered as secure.

Data or Instruction, PPROT[2]

- LOW indicates a data access
- HIGH indicates an instruction access.

This bit gives an indication if the transaction is a data or instruction access.

Note

This indication is provided as a hint and is not accurate in all cases. For example, where a transaction contains a mix of instruction and data items. It is recommended that, by default, an access is marked as a data access unless it is specifically known to be an instruction access.

Table 3-1 summarizes the encoding of the **PPROT[2:0]** signals.

Table 3-1 Protection encoding

PPROT[2:0]	Protection level
[0]	1 = privileged access 0 = normal access
[1]	1 = nonsecure access 0 = secure access
[2]	1 = instruction access 0 = data access

Note

The primary use of **PPROT** is as an identifier for Secure or Non-secure transactions. It is acceptable to use different interpretations of the **PPROT[0]** and **PPROT[2]** identifiers.

Chapter 4
Operating States

This chapter describes the AMBA APB operating states. It contains the following section:

4.1 Operating states

Figure 4-1 shows the operational activity of the APB.

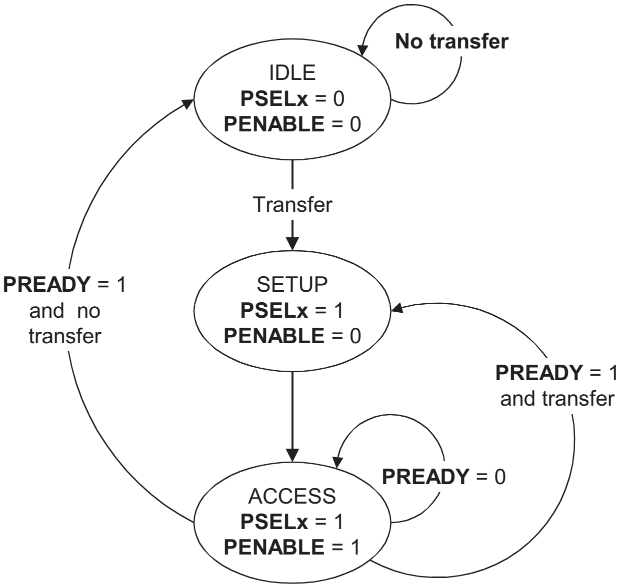


Figure 4-1 State diagram

The state machine operates through the following states:

IDLE This is the default state of the APB.

SETUP When a transfer is required the bus moves into the SETUP state, where the appropriate select signal, **PSELx**, is asserted. The bus only remains in the SETUP state for one clock cycle and always moves to the ACCESS state on the next rising edge of the clock.

ACCESS The enable signal, **PENABLE**, is asserted in the ACCESS state. The address, write, select, and write data signals must remain stable during the transition from the SETUP to ACCESS state.

Exit from the ACCESS state is controlled by the **PREADY** signal from the slave:

- If **PREADY** is held LOW by the slave then the peripheral bus remains in the ACCESS state.
- If **PREADY** is driven HIGH by the slave then the ACCESS state is exited and the bus returns to the IDLE state if no more transfers are required. Alternatively, the bus moves directly to the SETUP state if another transfer follows.

Appendix A
Revisions

This appendix describes the technical changes between released issues of this book.

Table A-1 Issue A

Change	Location	Affects
First release	-	-

Table A-2 Differences between issue A and issue B

Change	Location	Affects
APB signal **PREADY** added	• Table 2-1 on page 2-2. • *Write transfers* on page 3-2 • *Read transfers* on page 3-5 • *Error response* on page 3-6 • *Operating states* on page 4-2	All revisions
APB signal **PSLVERR** added	• Table 2-1 on page 2-2. • *Error response* on page 3-6	All revisions

Table A-3 Differences between issue B and issue C

Change	Location	Affects
Section added listing the changes made to this specification at each revision of the document.	*APB revisions* on page 1-3	–
APB signal **PPROT** added	• Table 2-1 on page 2-2 • *Protection unit support* on page 3-8	All revisions
APB signal **PSTRB** added	• Table 2-1 on page 2-2 • *Write strobes* on page 3-4	All revisions