**AMBA 3 APB PROTOCOL SPECIFICATION**

**V. Vinay Chowdary**

**INDEX**

**CONTENTS PAGE NO**

1. INTRODUCTION 6-8

1.1) About the Protocol

1.2) APB Versions

1.2.1) AMBA 2 APB Specifications

1.2.2) AMBA 3 APB Specifications v1.0

1.2.3) AMBA 3 APB Specifications v2.0

1.3) AMBA APB Interface Architecture

1. APB Interface Signals 9-10
2. AMBA APB Master 11
3. AMBA APB Slave 12
4. DM Registers 12-13
5. APB Operating states 14-15

6.1) IDEAL phase

6.2) SETUP phase

6.3) ACCESS phase

1. TRANSFERS 16-19

7.1) APB Write Transfer with no wait states

7.2) APB Write Transfer with wait states

7.3) APB Read Transfer with no wait states

7.4) APB Read Transfer with wait states

1. APB Error Response 19-20

8.1) APB Write transfer with error response

8.2) APB Read transfer with error response

**LIST OF FIGURES**

**FIG NO PAGE NO**

1.3) AMBA APB Interface Architecture 8

3.0) AMBA APB Master Block with input and output ports 11

4.0) AMBA APB Slave Block with input and output ports 12

6.0) State diagram of APB Protocol 14

7.1) APB Basic Write transfer with no wait states 16

7.2) APB Basic Write transfer with wait states 17

7.3) APB Basic Read transfer with no wait states 18

7.4) APB Basic Read transfer with wait states 19

8.1) APB Write transfer with Error response 20

8.2) APB Read transfer with Error Transfer 21

**LIST OF TABLES**

**TABLE NO PAGE NO**

2.0) APB Interface Signals 9-10

5.0) DM Registers 13

**1. INTRODUCTION**

**1.1 About the APB Protocol**

* APB Protocol stands for Advanced Peripheral Bus.
* The APB is a part of the Advanced Microcontroller Bus Architecture (AMBA) family.
* It provides a low cost interface that is optimized for minimal power consumption and reduced interface complexity.
* It is a Non Pipelined Protocol (it can run only one transaction at a time).
* Mainly used as a general purpose register based peripheral such as Timers, interrupt controllers, UART’s, IO ports etc.
* APB interfaces with any peripherals that are low bandwidth and do not require the high performances of a pipelined bus interface.
* Easy to Interface.
* It is connected to the System bus via bridge and helps in reducing system power consumption.
* It can Interface with AHB, AHB-Lite, AXI, AXI4-Lite etc..
* APB Protocol has 2 independent data buses, one for write data and another for read data.
* Buses can be upto 32 bits wide.
* Every transaction takes at least two cycles (one cycle is called SETUP phase and another cycle is called ACCESS phase).

**1.2 APB Protocol Versions**

The APB specification released in 1998, is now obsolete and is superseded by the following three versions

AMBA 2 APB Specification

AMBA 3 APB Protocol Specification v1.0

AMBA 3 APB Protocol Specification v2.0

**1.2.1 AMBA 2 APB Specification**

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 as APB2

**1.2.2 AMBA 3 APB Protocol Specification v1.0**

The AMBA 3 APB specification v1.0 defines the following additional functionality

1. WAIT States : A ready signal **PREADY** to extend an APB transfer
2. Error Reporting: An error signal **PSLVERR** to indicate the failure of a transfer

This version of specification is referred to as APB3

**1.2.3 AMBA 3 APB Protocol Specification v2.0**

The AMBA 3 APB specification v2.0 defines the following additional functionality

1. Transaction protection : A signal called **PPROT** protection signal to support secure and non secure transactions on APB
2. Sparse data transfer : A write strobe signal **PSTRB** to enable sparse data transfer on the write data bus.

This version of specification is referred to as APB4

The current Specification follows AMBA 3 APB specification v1.0 (APB3)

**1.3 AMBA APB INTERFACE ARCHITECTURE**

****

**Fig 1.3 AMBA APB Interface Architecture**

**2. APB INTERFACE SIGNALS:**

| **SIGNALS** | **SOURCE** | **DESCRIPTION** |
| --- | --- | --- |
| **Pclock** | clock source | Clock  1 bit wide |
| **Presetn** | System bus equivalent | Reset. The APB reset is active low and 1 bit wide |
| **Paddr** | APB Bridge | APB address bus, 32 bit wide |
| **Psel** | APB Bridge | Slave select signal  1 bit wide |
| **Penable** | APB Bridge | Enable signal  1 bit wide |
| **Pwrite** | APB Bridge | Direction when 1 write access else read access  1 bit wide |
| **Pwdata** | APB Bridge | Write data bus , 32 bit wide |
| **Pready** | Slave Interface | Ready signals, Slave used these signal to extend an APB Transfer  1 bit wide |
| **Prdata** | Slave Interface | read data,32 bits wide |
| **Psilverr** | Slave Interface | Indicates transfer failure, Slave error response  1 bit wide |
| **transfer** | coming from Top | Indicates start of transaction  1 bit wide |
| **read** | coming from Top | Indicates read transaction  1 bit wide |
| **write** | coming from Top | Indicates write transaction  1 bit wide |
| **apb\_paddr** | coming from Top | address signal,  32 bit wide |
| **apb\_write\_data** | coming from Top | Write data signal,  32 bit wide |
| **apb\_read\_data\_out** | Coming from Top | Data out signal going to JTAG |

**table 2.0 Interface Signals**

**3. AMBA APB MASTER**

****

**Fig 3.0 AMBA APB Master Block with input and output ports**

* If there is a single master on APB, no need for the Arbiter.
* Master drives the address and data busses and also performs a combinational decode of the address to decide which PSEL signal to activate.
* It is also responsible for driving the PENABLE to time the transfer
* It drives the APB data onto the system bus during a read transfer

**4. AMBA APB SLAVE**

****

**fig 4.0 AMBA APB SLAVE Block with outputs and input ports**

* APB Slave has a very simple and Flexible Interface
* The Interface will be dependent on the design style employed and many different options are available
* PSLVERR and PREADY are the two main signals which protect the lost data while transfer of data is taking place

**5. DM Registers**

* All the registers are 32-bit wide unless specified explicitly.
* Each DM has a base address and since there is only one DM in this implementation, all the addresses mentioned are absolute addresses.
* Upon write condition the Slave will write the data into the DM register as per the address.
* Upon read condition the Slave will read the data from the DM registers on the given address.

| **ADDRESS** | **REGISTER NAME** | **DESCRIPTION** |
| --- | --- | --- |
| 0x04 - 0x0f | data0 - data1 | Abstract data0 - Abstract data11 |
| 0x10 | dmcontrol | Debug module control |
| 0x11 | dmstatus | Debug module status |
| 0x12 | hartinfo | Hart information |
| 0x16 | abstractcs | Abstract control and status |
| 0x17 | command | Abstract command |
| 0x20 - 0x2f | progbuf0 - progbuf15 | Program buffer 0 - Program buffer 15 |

**table 5.0 DM Registers**

**6. APB OPERATING STATES**

****

**fig 6.0 State diagram for APB**

**6.1 IDLE PHASE:**

* Default state of APB

**6.2 SETUP PHASE:**

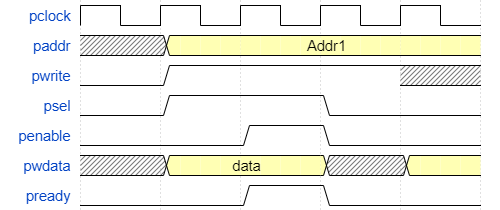
* When a transfer is required the bus moves into SETUP state, when the appropriate select signals PSEL is asserted
* Bus only remains in the SETUP state for only one clock cycle and always move sto ACCESS state on the next rising edge of the clock
* Address, Write, select and write data signals must remain stable during the transition from SETUP state to ACCESS state.

**6.3 ACCESS PHASE:**

* The enable signal Penable is asserted in ACCESS state
* Exit from the ACCESS state is controlled by the PREADY signal from the slave
* During the 1st cycle of the ACCESS phase, if PREADY is held low by the slave then the peripheral bus remains in the ACCESS state.
* During the 2nd cycle of ACCESS phase, 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
* If there are any other transfers followed, the bus moves directly into SETUP state.

**7. TRANSFERS:**

**7.1 APB Write transfer with no wait states:**

****

**fig 7.1 Basic write transfer with no wait states**

* Write transfer starts with address, write data, write signal, and select signals, and all will be changed during the rising edge of the clock.
* The first clock cycle of the transfer is called SETUP phase
* After following clock edge, The enable signal PENABLE is asserted and it indicates the ACCESS phase is talking phase.
* The address data and the control signals all remain valid throughout the ACCESS phase. Transfer completes at the end of cycle.
* The PENABLE is deasserted at the end of the transfer.
* The select signal PSEL also goes low unless the transfer is to be followed immediately by another transfer to the same peripheral.

**7.2 APB transfer with wait states:**

****

**fig 7.2 APB write transfer with wait states**

* As demonstrated in the figure, how the PREADY signal can extend the transfer
* During an ACCESS phase, when PENABLE is high, the transfer can be extended by driving PREADY low.
* Following signals remains same for the additional cycles of access phase

PADDR, address signal

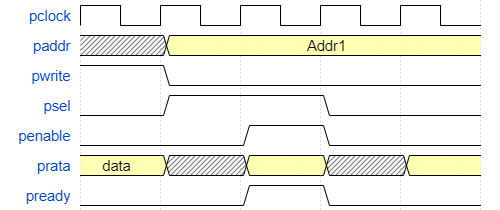
PSEL, select signal

PWRITE, write signal

PWDATA, write data

PENABLE, enable signal

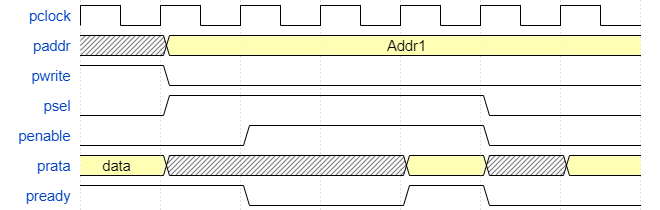
**7.3 APB Read transfer with no wait states**



**fig 7.3 basic read transfer with no wait states**

* Similarly read transfer starts with write address, write signal and select signals and all will be changing during the rise edge of the clock
* The first cycle of the transfer is called SETUP phase
* After the clock edge, the enable signal PENABLE is asserted and it indicates the ACCESS phase is the talking phase
* The address, data and control signals all will remain valid throughout the ACCESS phase
* The enable signal PENABLE is deasserted at the end of the transfer
* The select signal PSEL, also go unless the transfer is to be followed immediately by another transfer to the same peripheral

**7.4 APB Read transfer with wait states**

****

**fig 7.4 APB Read transfer with wait states**

* As determined in the figure, how the PREADY signal can extend the transfer
* During the ACCESS phase, when PENABLE is high, the transfer can be extended by driving PREADY low
* Following signals will be the same for the additional cycles of ACCESS phase

PADDR, address signal

PSEL, select signal

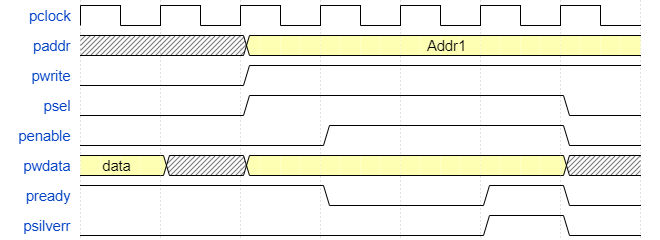
PWRITE, write signal

PENABLE, enable signal

**8. APB ERROR RESPONSE**

* Use PSLVERR to indicate an error condition on the APB transfer
* Error condition can occur in both read and write transactions
* PSLVERR is only considered valid during the last cycle of the APB transaction, when PSEL, PENABLE, PREADY all signals are high
* Transaction that receive an error, might or might not have changed the state of peripheral, It is a peripheral specific
* APB peripherals are not required to support the PSLVERR pin, If peripheral does not include this pin then input to APB Bridge can be tied low.

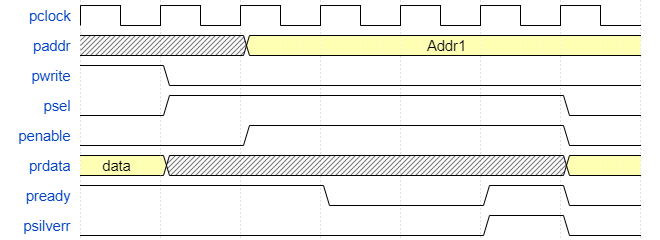
**8.1 APB Write transfer with Error Response:**

****

**fig 8.1 APB write transfer with error response**

* When a write transaction receives an error, it does not mean that the register within the peripheral has not been updated

**8.2 APB Read transfer with error response**



**fig 8.2 APB Read transfer with error response**

* Read transactions that receive an error can return invalid data. There is no recruitment for the peripheral to drive the data bus to all 0’s for a read error