

AHB TO APB BRIDGE VERIFICATION

Trainer : Manoj Kumar N

Mentor: : Sonia Stanley Louis

Prepared by : V Bhargava

Batch no: : BPRN10

# CONTENTS:

* INTRODUCTION
* ARCHITECTURE
* AHB SIDE
* AHB SIGNALS
* APB SIDE
* APB SIGNALS
* AHB 2 APB BRIDGE
* SCOREBOARD
* WAVE FORMS
* CONCLUSION

# INTRODUCTION:

SOC is a system on chip with multiple IP’s. Soc can be helpful by reducing size, area & portability, every operation can be performed on a single chip itself that is why it is called as system on chip.

ARM Advanced Microcontroller Bus Architecture (AMBA) is an open- standard, on-chip interconnect specification for the connection and management of functional blocks in system-on-a-chip (SoC) designs. It facilitates the development of multi-processor designs with large numbers of controllers and components with a bus architecture. Today, AMBA is widely used on a range of ASIC and SoC parts including applications processors used in modern portable mobile devices like smartphones.

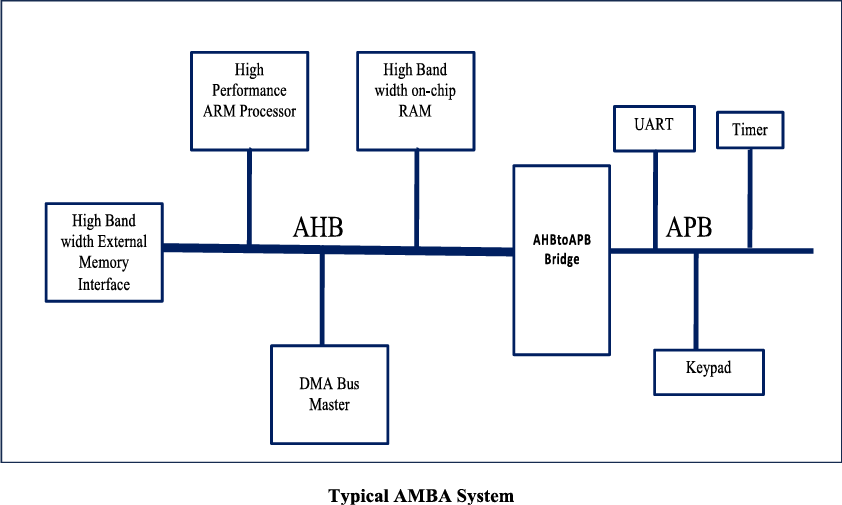
On a Soc we will be having lot of high-performance buses like AHB devices, low performance buses, and peripheral buses like UART, mouse, Keyboard. As we have high performance buses and low performance buses on SOC, to establish communication between these high-performance buses and low performance buses we use Bridge.

Bridge converts one protocol into another protocol i.e it converts high performance buses into low performance buses and hence establishes the communication between them. Generally, bus can be used to communicate between the IP’s but not between the protocols.

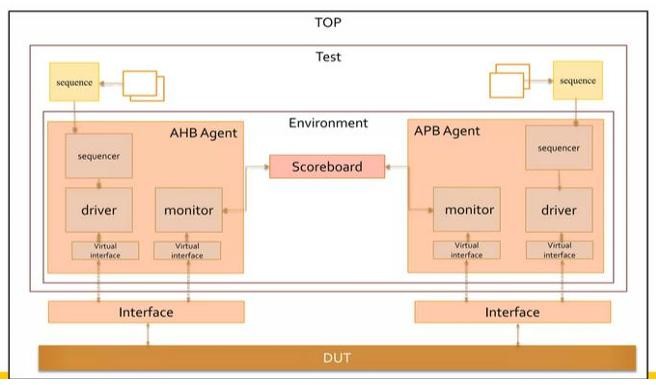
In the bridge itself we are going to store the data in the form of queue and use it whenever needed. The bridge makes the transition from AHB to APB i.e high performance bus to low-performance bus.

Here in our AHB to APB bridge project, the AHB side acts as master and the APB side acts as slave and the AHB to APB bridge is an AHB slave and the only APB master which provides an interface between the highspeed AHB and the low-

power APB. Read and write transfers on the AHB are converted into equivalent transfers on the APB.



# ARCHITECTURE:



**AHB SIDE:**

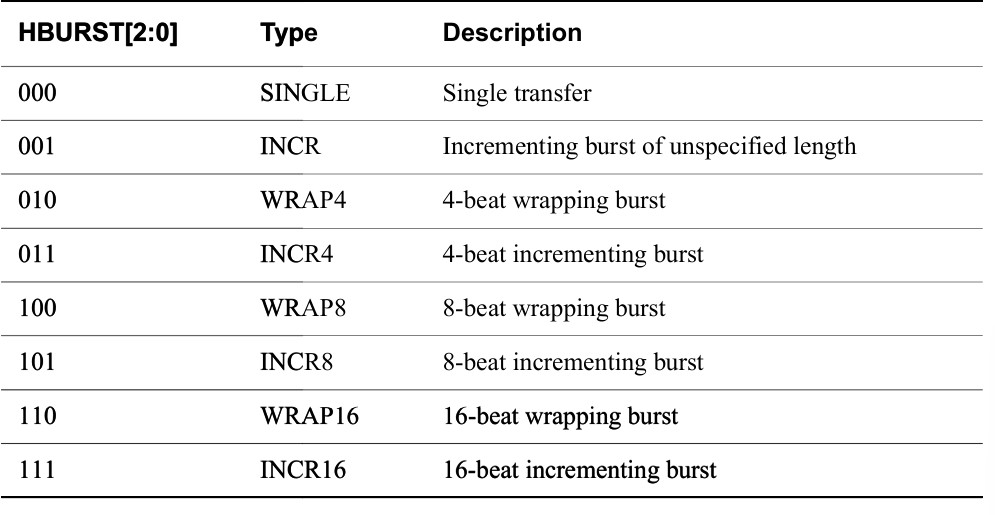
AHB is the advanced high-performance bus. In our project it is the master which initiates the transaction supports multiple features like:

* Split transactions
* Burst transactions
* Pipelined operations
* Multiple masters (supports up to 16 masters)
* Single clock edge operation

**Burst operation:**

Four, eight and sixteen-beat bursts are defined in the AMBA AHB protocol, as well as undefined-length bursts and single transfers. Both incrementing and wrapping bursts are supported in the protocol:

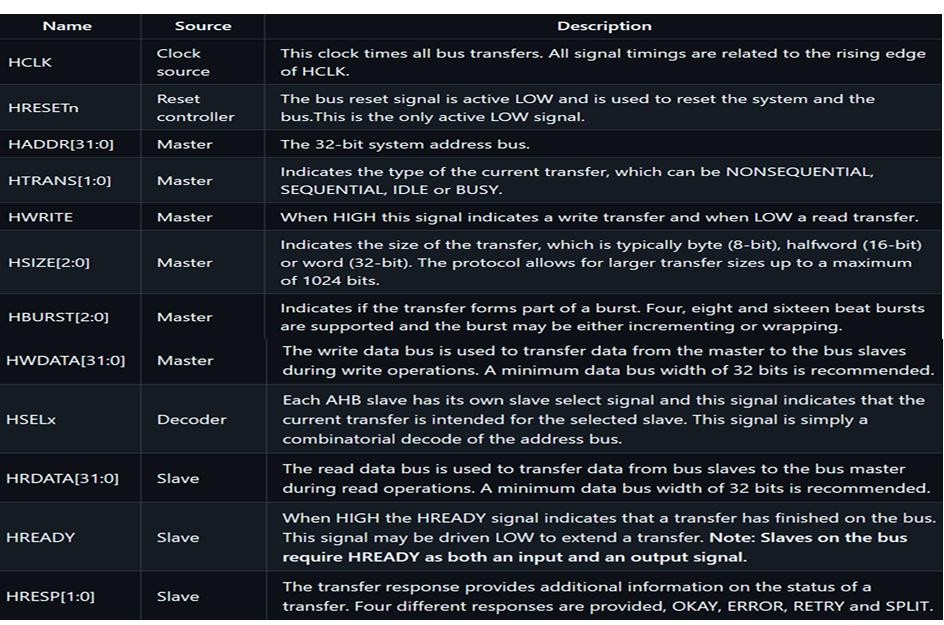
* Incrementing bursts access sequential locations and the address of each transfer in the burst is just an increment of the previous address.
* For wrapping bursts, if the start address of the transfer is not aligned to the total number of bytes in the burst (size × beats), then the address of the transfers in the burst will wrap when the boundary is reached. For example, a four-beat wrapping burst of word (4-byte) accesses will wrap at 16-byte boundaries. Therefore, if the start address of the transfer is 0x34, then it consists of four transfers to addresses 0x34, 0x38, 0x3C, and 0x30.



**Transfer direction:**

When HWRITE is HIGH, this signal indicates a write transfer and the master will broadcast data on the write data bus, HWDATA [31:0]. When LOW a read transfer will be performed and the slave must generate the data on the read data bus HRDATA [31:0].

**AHB Signals:**



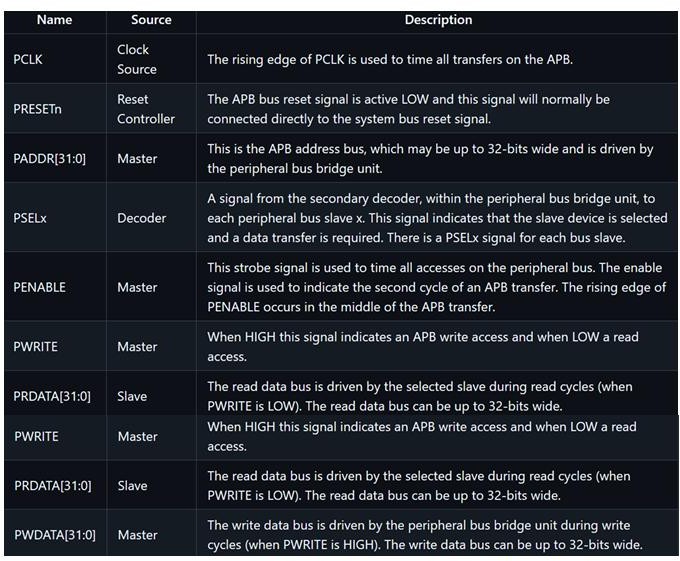
# APB SIDE:

APB is the advanced peripheral bus. In our project it is the slave which completes the transaction initiated by the slave. • The AMBA APB is for low-power peripherals which is used to interface to any peripherals which are having low bandwidth and does not require the high performance and this APB is a non- pipelined protocol. APB protocol require 2 clock cycles to complete the transfer and it has a simple interface.

Features of APB:

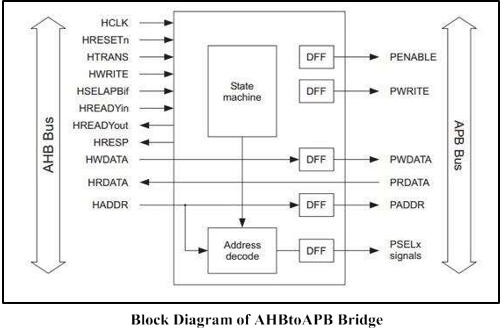
* Low power consumption.
* No pipeline stages.
* Configurable wait states.
* Single transfer Operations.

**APB Signals:**



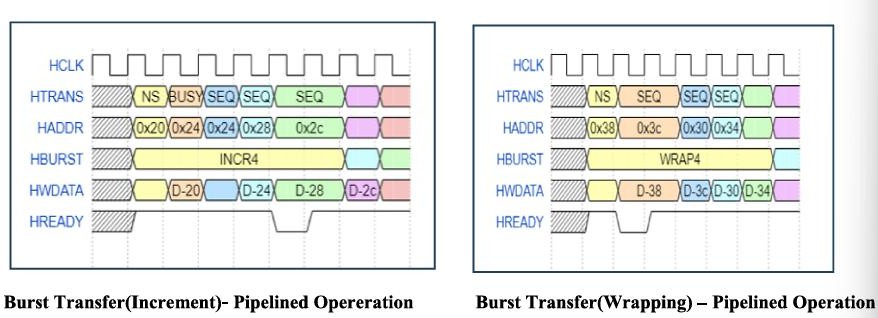
# AHB 2 APB BRIDGE:

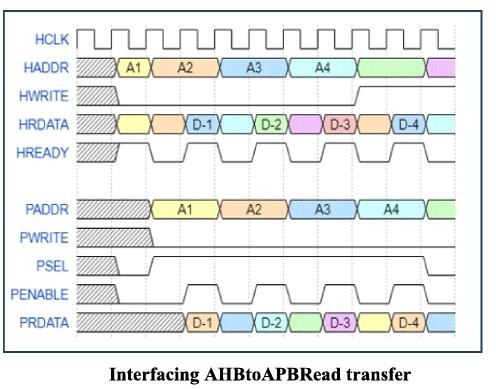
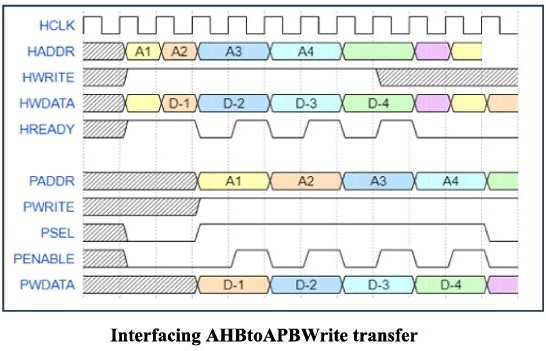
The AHB-to-APB bridge is an AHB slave, providing an interface between the high-speed AHB and the low-power APB. Read and write transfers on the AHB are converted into equivalent transfers on the APB. As the APB is not pipelined, then wait states are added during transfers to and from the APB when the AHB is required to wait for the APB.



The bridge unit converts system bus transfers into APB transfers and performs the following functions:

* Latches the address and holds it valid throughout the transfer.
* Decodes the address and generates a peripheral select, PSELx. Only one select signal can be active during a transfer.
* Drives the data onto the APB for a write transfer.
* Drives the APB data onto the system bus for a read transfer.
* Generates a timing strobe, PENABLE, for the transfer.





# SCOREBOARD:

* + In the scoreboard the data from the AHB monitor and APB monitor are broadcasted and compared.
  + The AHB and APB data will be stored in analysis fifo’s. We can get the data from the AHB fifo and APB fifo using get method. We are pushing AHB data into the queue, it will help to maintain the synchronization. While checking the data we are going to pop the data and will be used for comparisons.
  + Scoreboard will compare PDATA with HDATA while read and write transfers. While comparing we are checking HWRITE and HSIZE signal.
  + When Hwrite is 1, we will check the write data i.e HWDATA at AHB side with PWDATA at APB side.

# Take Away from the Project & difficulties faced during Project:

**Problems faced:**

* Clock Synchronization Issues
* Incorrect timing can cause data loss or mismatched addresses.

**Take aways:**

On working with this project, I have learnt how the soc’s will be working exactly and how the bridge works to communicate between various types of protocols present on the soc.

Verification methodologies: apply UVM for creating scalable test bench environment implement effective.

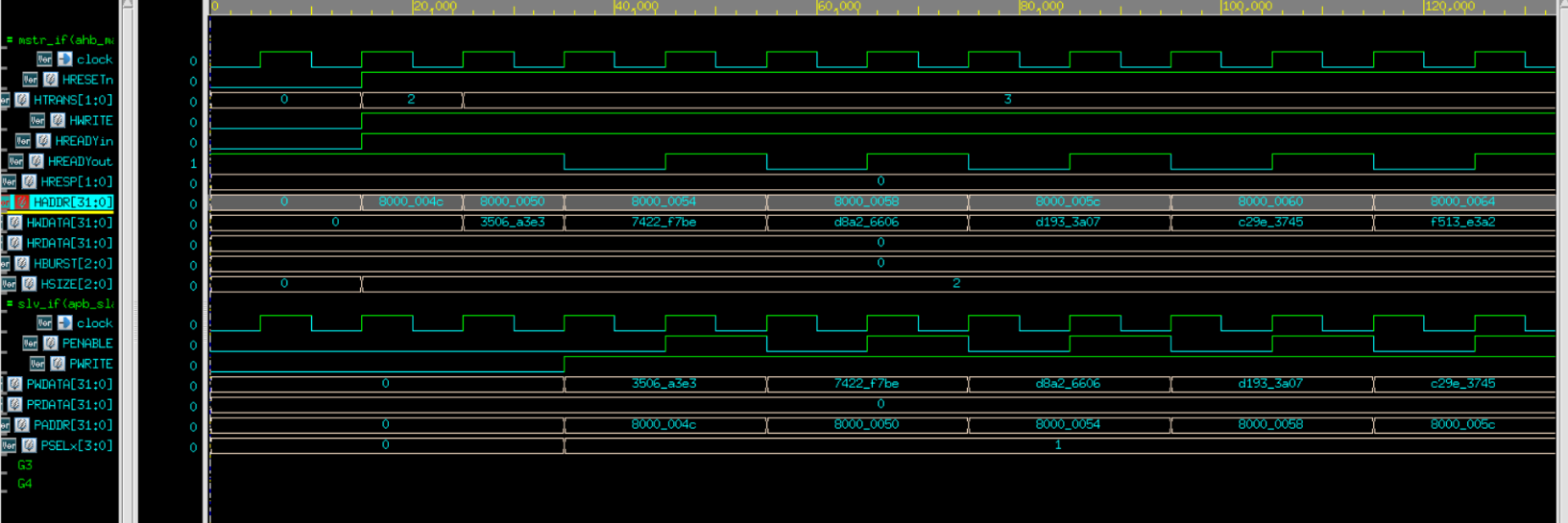
Functional verification verifies the correct functionality of the AHB to APB bridge under various scenarios and transaction types, validates the bridge ability to handle different data sizes, read and write operations and address ranges.

While getting the sequence printed in the terminal many times the monitor data is missing and it is rectified by changing the logic of driver and monitor

While working with the concept of Scoreboard, initially I was very confused with the address slicing part in order to compare the data then when I have checked the rtl files then I have got to know about the concept of little endian used in the bridge rtl based on which our design works. So, then I have got to know that referring to the rtl files before we proceed further is very important.

# OUTPUT Wave Forms:

**Increment 16 (write transfer):**



**Wrap 4 (Read Transfer):**



# CONCLUSION:

The Verification of AHB to APB Bridge is done using UVM (Universal Verification Methodology) by developing various test cases for testing the features. Both the AHB & APB drivers are driving the data properly and both the AHB & APB monitors are sampling the data properly and projecting them to the scoreboard. And in the scoreboard also the data we are getting from the master(AHB) side and the slave(APB) side are matching successfully through which we can conclude that the verification is done successfully which means that the bridge design is working properly

**Thank you**