Available online @ https://jjem.jnnce.ac.in
https:www.doi.org/10.37314/JJEM.2020.040202
Indexed in International Scientific Indiexing (ISI)
Impact factor: 1.025 for 2018-19
Published on: 30 March 2020

Development of Flexible Verification Environment for AMBA APB

Nithin HV[1]* and Kunjan D. Shinde[2]

[1*,2]Asst. Professor, Department of Electronics and Communication Engineering,
PESITM, Shivamogga, Karnataka, India

hv.nithin@gmail.com, kunjan18m@gmail.com

Abstract

The SoC (System on Chip) uses AMBA (Advanced Microcontroller Bus Architecture) as an on chip bus. APB (Advanced Peripheral Bus) is one of the components of the AMBA bus architecture. APB is low bandwidth and low performance bus used to connect the peripherals like UART, Keypad, Timer and other peripheral devices to the bus architecture. This paper introduces the AMBA APB bus architecture design. The design is created using the verilog HDL and is tested by a System verilog test- bench. This design is verified using SV. A reuse based methodology for SoC design has become essential in order to meet these challenges. The work embodied in this paper presents the design of APB Protocol and the Verification of slave APB Protocol. Coverage analysis is a vital part of the verification process; it gives idea that to what degree the source code of the DUT has been tested. The functional coverage analysis increases the verification efficiency enabling the verification engineer to isolate the areas of untested function. The design and verification IP is built by developing verification components using Verilog and System Verilog respectfully with relevant tools such as Questasim and Cadence, which provides the suitable building blocks to design the test environment..

Keywords: *AMBA APB, SoC.*

1. Introduction

Nowadays, the number of Integrated chips on a board is constantly increasing; IC density also increased to an extreme level. So the interface problem causes major time delay and more issues. So, we are in need to find better solution for interface control. There are several types of verification; we focus on creating a test environment for the slaves (DUT) of the AMBAAPB protocol and verify its performance by imitating the master of this protocol in test bench. The 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 designs. It facilitates development of multi-processor designs with large number of controllers and peripherals. The scope of AMBA has, despite its name, gone far beyond microcontroller devices. Today, AMBA is widely used on a range of ASIC and SOC parts including application processors used in modern portable mobile devices like smart phones. AMBA was first introduced in 1996. The first buses were Advanced System Bus (ASB) and Advanced Peripheral Bus (APB). APB is designed for low bandwidth control accesses, for example register interfaces on system peripherals. This bus has an address and data phase similar to AHB, but a much reduced, low complexity signal list. APB is low bandwidth and low performance bus used to connect the peripherals like UART, Keypad, Timer and other peripheral devices to the bus architecture. Since APB has un-pipelined protocol. Therefore, it interfaces to the low bandwidth peripherals that do not demand the high performance of the pipelined bus interface. All the signal transitions are associated with the rising edge of the clock which makes it simple to integrate APB peripherals into any design flow. APB can

interface with the AMBA AHB-Lite and AMBA Advanced Extensible Interface (AXI).
APB was developed in the earlier decades, when the testing tools were not developed so much. It is a low bandwidth protocol that can be used in various applications, and is high cost effective. While designing any system, only 20% of the time is spent on design and rest 80% of the time is spent on verification process. It is necessary to test and verify this design and explore the different possibilities of the system, to expand its usage. The present test benches require the DUT of the master and various peripherals on the system. It is highly necessary for the Test Engineer to develop a test environment based on the specification given by the customer. A Verification Engineer must be able to replicate the DUT (imitation of actual design), even before the design is completed and create an environment that tests the actual design with the replica. The test environment must be such that, it should cover all the corner cases of the design. The test environment has to very easy to interface, like plug and play. The environment has to easily modifiable for testing different cases & also to be used in future without any fuss in creating a new test environment.

Bala Krishna and Bheema Raju [1] have designed and tested AMBA APB protocol. Kiran et al., [2] have implemented the AMBA APB protocol at system chip level. Kiran et al., [3] performed the implementation of AMBA APB Bridge with Efficient Deployment of System Resources. Shankar et al.,[4] have verified the functionality of AMBA APB protocol and described its methods to test.

2. Design

An AMBA AHB design may include one or more bus masters, typically a system would contain at least the processor and the test interface. However, it would also be common for a Direct Memory Access (DMA) or Digital Signal Processor (DSP) to be included as bus masters. The external memory interface, the Advanced Peripheral Bus (APB) Bridge and any internal memory are the most common AHB slaves. Any other peripheral in the system could also be included as an AHB slave. However, low bandwidth peripherals typically reside on the APB. In this design lets consider the SLAVE module of APB and verify as shown in the figure 1.

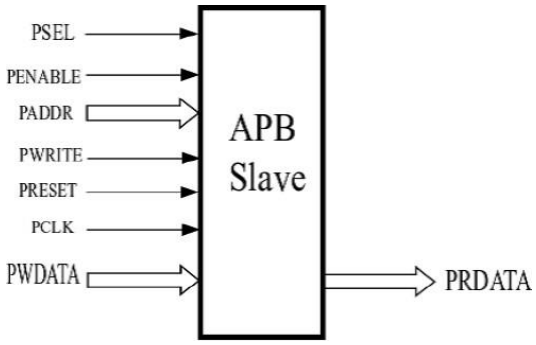

Figure 1: Pin outs of the salve in a APB protocol

The write and read cycle with no wait states are shown in figures 2 and 3 respectively, and this will act as a reference to verify the functionality.

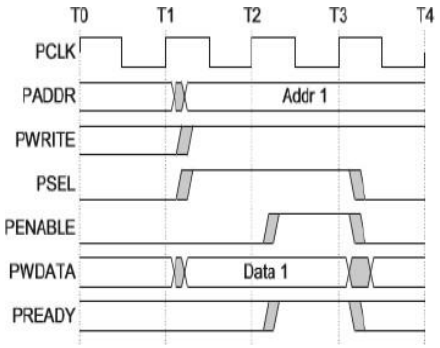

Figure 2: Write transfer with no wait states

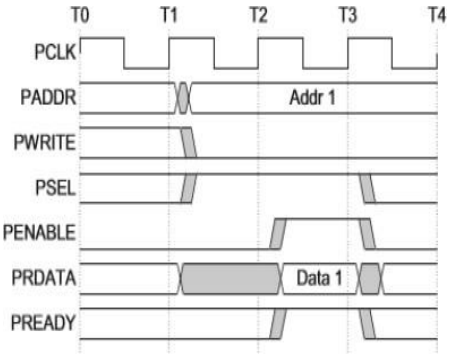

Figure 3: Read transfer with no wait states

3. Algorithm

The AMBA APB slave block that was mentioned in the figure 1 shall be tested using the flow diagram shown in figure 4.

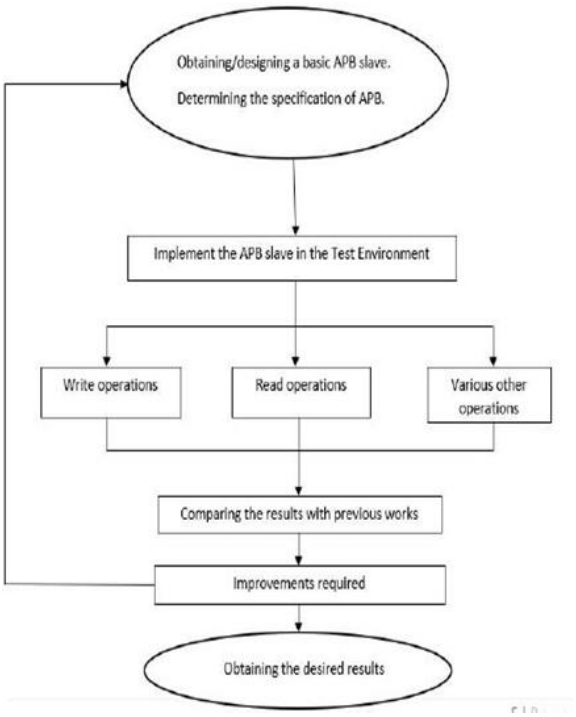

Figure 4: Algorithm for designing the test bench

4. Results and Discussion

The design and verification IP is built by developing verification components using Verilog and System Verilog respectively with relevant tools such as Questasim and Cadence, which provides the suitable building blocks to design the test environment

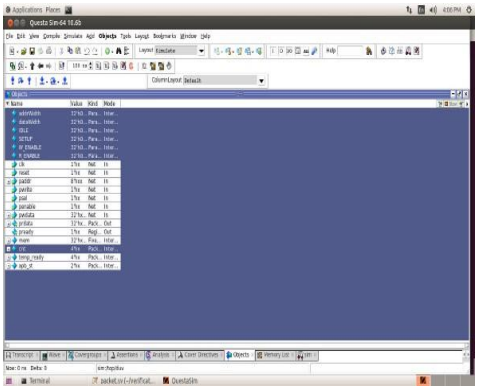

Figure 5: Objects(input/output) required for simulation

The input and output required for the simulation of the APB slave is shown in figure 5. The signals that are selected are also the pinouts of the design. The signals that are not selected, are the internals variables inside the design. These variables won't be available when fabricated and works internally along with the normal processing of the data. The selected signals are added to the waveforms and they are verified using the same set of signals through-out the verification environment.

The screenshot (figure 6), taken during the simulation depicts the behavior of the protocol.

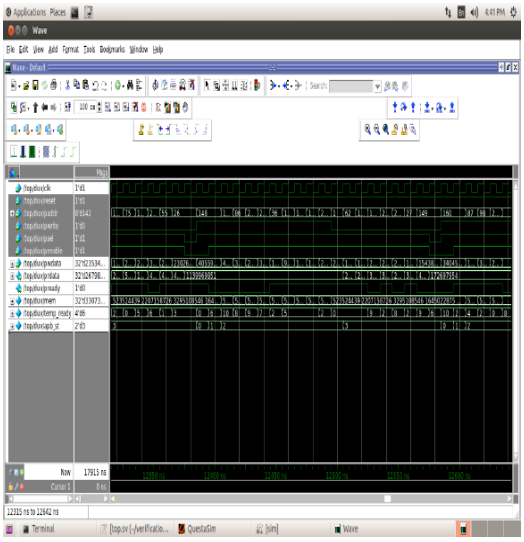

Figure 6: Outputs

Here, the clock signals are generated in a continuous form and the reset signal is made logic low for initial few cycles and then it is made high to begin the processing of data. To select any slave the 'PSEL' signal has to go high, else the slave will be inactive. We use the 'PWRITE' signal to perform the read and write operations. When the signal is low the slave is designed to read from its memory and when the signal is high, the slave takes the input from 'PWRITE' pins. In the following clock signal the 'PENABLE' will go high, enabling the whole slave design for operating. When the PSEL is high the slave is only chosen but, the control is within PENA-

BLE. Only when this signal is high the design will be able to communicate and perform operations.

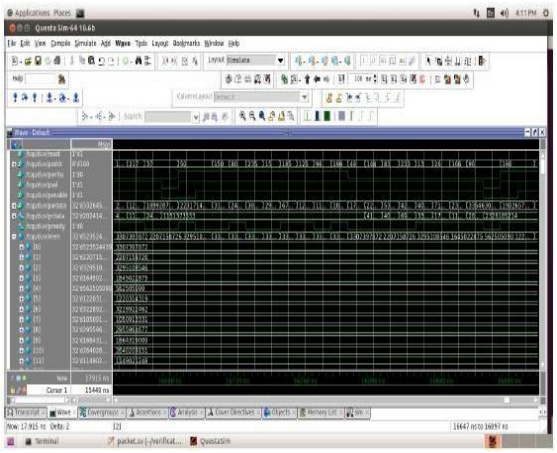

Figure 7: Memory of the Design under Test (DUT)

The memory of the design under test is shown in figure 7. In this verification process, the memory is fully utilized with write and read operations including multiple writes to the same memory location. The memory is accessed using the PWRITE signal with logic low. While rewriting a memory location, the old data is erased and replaced with new data. The same is updated efficiently in the verification environment for validating the design.

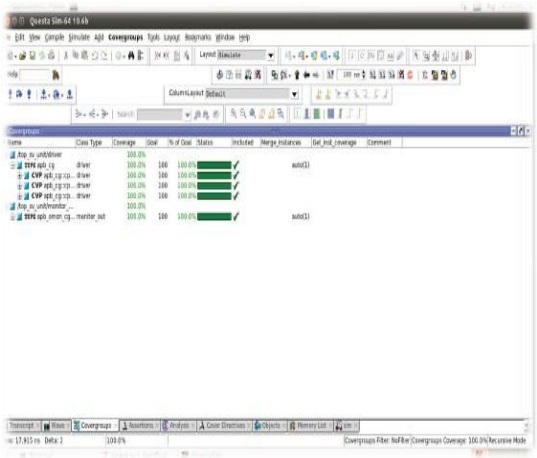

Figure 8: Coverage

There is a unique function in system verilog to keep a watch on the data signals and to measure the quality of the test-bench and test cases. The coverage (figure 8) is functionality check for the test-bench to check if all the test cases are hit at least once during the simulation.

The APB bus is designed using the verilog HDL according to the specification and is verified using System Verilog. The simulation results show that the data read from a particular memory location is same as the data written to the given memory location. Hence, the design is functionally correct. The SV report summary also ensures the functional correctness of the design along with 100% coverage of protocol mismatch, interface correctness and component composition and showed how these methods lend themselves to interface synthesis and model checking.

5. Conclusion

Here the design and verification of low peripheral processor's data transfer protocol has been discussed. The APB bus is designed using the verilog HDL according to the specification and is verified using system verilog. The simulation results show that the data read from a particular memory location is same as the data written to the given memory location. Hence, the design is functionally correct. The SV report summary also ensures the functional correctness of the design along with 100% coverage of protocol mismatch, interface correctness and component composition and showed how these methods lend themselves to interface synthesis and model checking.

References

1. Kommirisetti Bheema Raju and Bala Krishna Konda, Design and verification of AMBA APB protocol, 2017, International Journal of Engineering Research and Applications, Vol. 7, Issue 1, 2017, 87-90.

2. Kiran Rawat, Kanika Sahani and Sujata Pandey. RTL implementation for AMBA ASB APB Protocol at System on Chip Level, Proceedings of the 2nd International Conference on Signal Processing and Integrated Networks (SPIN), 19-20 Feb. 2015, Noida, India, 927-930.

3. Kiran Rawat, Kanika Sahani, Sujata Pandey, Jyoti Rawat and Sudhanshu Tripathi, Implementation of AMBA APB Bridge with Efficient Deployment of System Resources, Proceedings of the International Conference on Computer, Communication and Control (IC4), 10-12 Sept. 2015, Indore, India

4. Shankar, Dipti Girdhar and Neeraj Kr. Shukla, Design and Verification of AMBA APB Protocol, International Journal of Computer Applications (0975 – 8887), Vol. 95, No.21, June 2014, 29-35.