Int. J. Chem. Sci.: 14(S3), 2016, 769-776
ISSN 0972-768X
www.sadgurupublications.com

MULTI-MASTER AMBA AHB PROTOCOL VERIFICATION USING TLM BASED UVM ENVIRONMENT

SINNY P. SUNNY, S. SIVANANTHAM* and C. PRALINE RAJABAI

Department of Micro and Nanoelectronics, School of Electronics Engineering, VIT University, VELLORE (T.N.) INDIA

ABSTRACT

In the due course of time, due to rising development cost and density of VLSI chips and turnaround time, it turns out to be critical to have a verification methodology, which empowers first pass chips to be entirely functional and error free. Universal Verification Methodology (UVM) facilitates the communication through TLM interface. On account of its excellent architecture of AMBA and simplicity of AHB bus it has been widely used in several SOC designs. This paper is focused on developing a Verification IP (VIP) for Multi-master AMBA AHB protocol using System Verilog based UVM environment. AMBA-AHB provides a high bandwidth system bus which can perform multiple operations in parallel. This project also depicts how Verification IP is used to verify the AHB Components-Arbiter, Slave, Master and Decoder. With the UVM based VIP, it was able to achieve MDV (Metric Driven Verification) and assertion based verification which has drastically minimized the time spent on verification of a design.The Verification IP is developed using Cadence tool Ncsim and can be reused to verify any AHB based system design.

Key words: AMBA-AHB, VIP, TLM, UVM, CDV, Coverage, Assertions, Burst, Arbiter.

INTRODUCTION

Designs are emerging as highly reusable and portable units, which in turn calls for rising need of an efficient, reusable, plug n play verification infrastructure. Room for bugs in a design has become negligible due to the reduced time to market. As a result, functional verification is one among the challenging tasks. Verification Intellectual Property (VIP) comprises of the overall verification environment, that aids design architects and verification engineers for ensuring the design functionality. The VIP(s) are utilized as a part of almost all

*Author for correspondence; E-mail: ssivanantham@vit.ac.in

sections in simulation based verification. This reusable verification module includes BFMs, protocol monitors, coverage blocks and traffic generators.

System on chip (SoC) designs calls for need of an on chip bus, which provides efficient integration of various system components including memories (RAM, EEPROM etc), application specific cores, processors (CPUs, DSPs). The rising complexity of designs demands for a greater bandwidth system bus, which handles multiple operations simultaneously. Multi-master AHB bus is been widely used in most SoC designs due to its transparency and efficient architecture. In this project, System Verilog-based methodologies, like Universal Verification Methodology (UVM), can be used to create a verification environment for verification of ARM's AMBA AHB Communication Protocol[4]. AHB is a system bus with advanced higher performance and it can handle more than one master and slave. It implements burst transfers (incr/wrap), split transactions, single-cycle handover of bus master handover.

Verification IP is the one, which provides a smart way to verify the AHB Components such as Master, Slave, Arbiter and Decoder. UVM Methodology is one of the most suited methodology to be used for verification as it facilitates features like automation, self-checking test bench and coverage analysis thereby reducing the time spent on verifying a design[3]. It includes verification components like test, driver, monitor, environment, agent and scoreboard, which are reusable. The verification environment is built with the test bench components like the test, environment, agent, driver, sequencer, monitor, and scoreboard. Verification is essential in all kinds of logic design[5-8].

The paper intends to propose a verification IP for verifying multi master ahb protocol based on System Verilog under Universal Verification Methodology (UVM) guidelines[2]. The methodology aims to create a verification approach that creates a reusable automatic process for verifying any AHB based SoC system design. The remainder of this paper is organized as follow: Section II depicts the idea of functional verification, System Verilog and UVM methodologies and also gives a brief overview of AMBA-AHB Protocol. Segment III explains the verification environment and verification plan, which was developed for verification of AHB. Section IV gives the detailed description of top module and each classes of the UVM environment. Section V includes the results and completes with conclusion.

Design and methodology

The Verification of the AMBA-AHB Protocol is done in SV/UVM based test environment. The advantage of UVM based test bench environment is that it is coverage-

driven random based test environment. Both the directed test cases as well as the random test cases are tested on the design IP to meet the full coverage of the design. AHB slave is used to program memory registers. Fig. 1 shows an overview block diagram of the design IP.

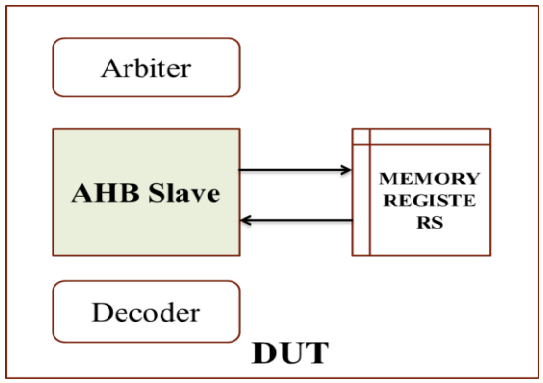

Fig. 1: Components of AHB-Slave DUT

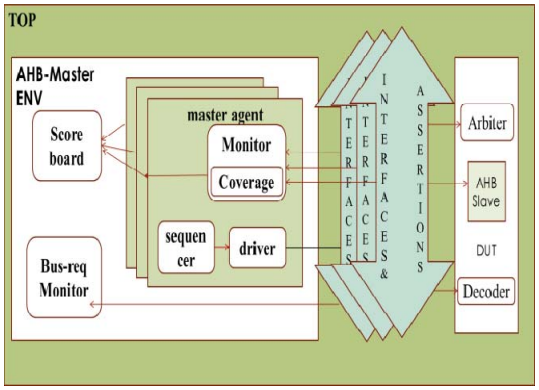

Fig. 2: UVM environment

The verification of the AHB-Slave DUT is carried out by building the UVM TB as shown in Fig. 2 and the same is explained as follows.

AHB Top module

AHB DUT instantiated along with the interfaces. The Clock generator is also implemented in top module. Here in the top module inbuilt method run test () is executed and this method will execute all the test cases for the design. These Interfaces are used for interaction with the DUT and the TB. Assertions are added to validate protocol, timing, etc. Environment also includes the test classes and coverage collectors.

AHB Multi master environment

Environment class implements verification environments in UVM. In environment class, multiple instances of master agents, and scoreboard is created using build () method. Then various port connections details are made using the connect () method. The AHB Multi-master Environment has 3 AHB master agents.

AHB Master agent

The register configuration of the DUT is done through the AHB interface and this is achieved by the AHB agent. All the register read/write operation are done by the AHB agent. Agents typically consist of three sub components Driver, Sequencer and Monitor. Active

agents contain these three sub components. However, passive agent generally requires monitor only. AHB Master agent is an active agent. It extends from uvm agent class. In agent class object instances of the driver, monitor and sequencer are created in build phase. Connections between the driver, sequencer and monitor are specified in the connect phase. Signals are sent from driver to scoreboard using analysis ports.

AHB Driver

Drives the signals to reset state during the reset and when it is out of reset, driver requests and receives the transaction from sequencer. And it request the arbiter for bus access by asserting the bus request signal and once it receives the grant, the transaction is converted to pin wiggles as per the direction of operation and type of burst. When burst type is wrap, it calculates the wrap boundary address and it is driven correspondingly. It also monitors for the Response from the slave & Grant going low and takes appropriate decision. If slave response is error then the transaction is terminated. If slave response is split/retry or grant goes low then transaction is terminated and the driver creates a new transaction with the unsent transfers in the burst and requests for the BUS again.

AHB Sequence

A sequence is series of transaction and sequencer is used for controlling the flow of transaction generation. UVM sequencer has a port seq item export, which is used to connect to uvm driver for transaction transfer. This sequence of transactions should be defined in body () method of uvm sequence class. UVM has macros and methods to define the transaction types.

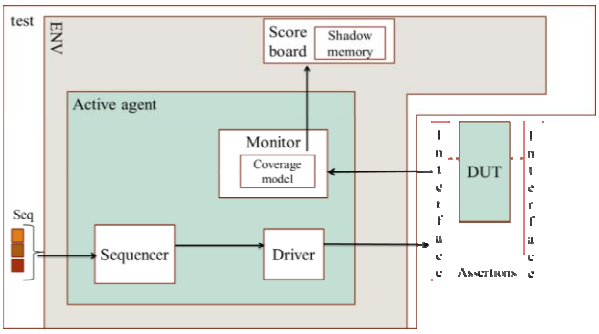

Fig. 3: AHB Single master agent

AHB Monitor

Monitor is defined by extending uvm monitor. Monitor gets the input signals from the interface using mod ports. These are used for getting the idea of whether data is

generated as required. It is not necessary that master requires a monitor. In this project, there is an additional bus-request Monitor that monitors the bus-request signals from each master agent. This is used to obtain multi-master coverage.

AHB Scoreboard

Scoreboard is implemented by extending uvm scoreboard. A shadow memory, duplicating the memory in DUT is maintained inside scoreboard. Whenever a WRITE transaction is received from monitor, shadow memory is updated with the write data into that particular address. Similarly, whenever a READ transaction is received from monitor, expected value is taken from shadow memory and the actual value from slave DUT. Expected and Actual values are compared and displays PASS/FAIL.

RESULTS AND DISCUSSION

The Simulation is done using Cadence Ncsim EDA Tool. Test cases to verify various usecases were run and results obatained for few of them are added below.

Multi master test

Here we have 3 masters requesting the bus with priorities in the order master 0 > master 1 > master 2 as shown in Fig. 4. Even during a transaction by a master of lower priority if higher priority master requests for the bus the transaction will be stopped temporarily and grant will be assigned to higher priority master. The lower priority master will regain the access of the bus only after this master transaction gets completed.

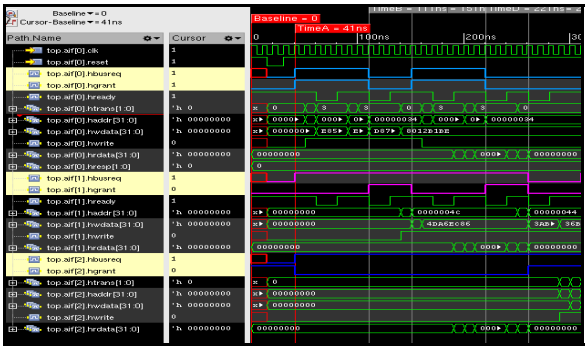

Fig. 4: Multi master wrap 4 test case

WRAP4 Write test

For burst type wrap (Fig. 5), it calculates the wrap boundary address based on the starting address and the address gets wrapped while reaching this boundary after it’s

incremented. Here DUT supports size of 32 bits so increment happens by 4 bytes. Start address given is 0 x 38, which calculates the wrap boundary as 0 x 3f, so during the third cycle it wraps to 0 x 30 and increment continues. The transfer type signal indicates the current transfer type. During the first cycle of any transfer it should be non-sequential and for the remaining it should be sequential.

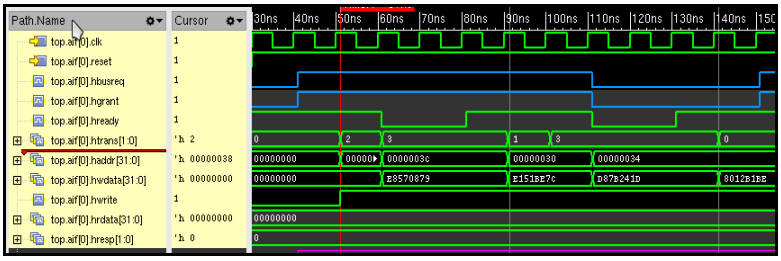

Fig. 5: wrap 4 write test case

INCR 4 Read test

For burst type incr, there is no boundary unlike in wrap. Start address given is 0 x 58 and increment continues for 4 addresses by 4 bytes. The transfer type signal indicates the current transfer type. During the first cycle of any transfer it should be non-sequential and for the remaining it should be sequential.

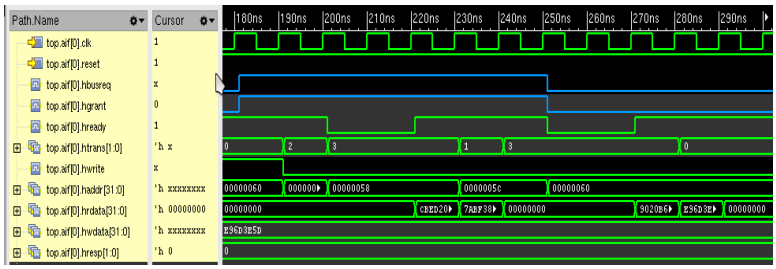

Fig. 6: INCR 4 read test case

Locked transfer test and coverage analysis

Locked transfer feature ensures that any lower priority master using the bus for performing an important transfer which shouldn't be interrupted requests for the locked access of the bus which prevent any other higher priority bus from getting access of the bus. Here master 1 wants to perform uninterrupted transfer so it requires a locked access of bus. During master 1 performing operation master 0 with higher priority requests for the bus but it receives the grant only after master 0 has completed its operation.

Fig. 7: Locked transfer test case

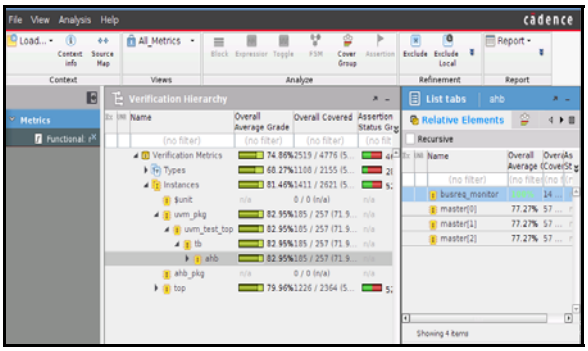

Fig. 8: Coverage report

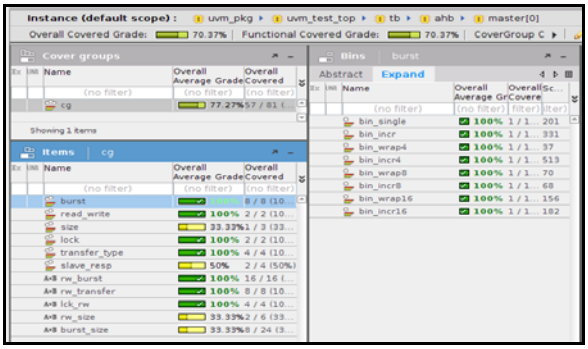

Fig. 9: Cover points analysis report

The coverage report analysis is done to measure the functional coverage. It indicates the use cases of the DUT tested and which are missed in the test cases. The uses cases are defined as cover points based on certain conditions called bins. If these conditions are met then that functionality is said to be hit or covered. Here the overall coverage percentage obtained is 71.9% and overall average grade is 82.95%. Coverage value less than 100%

indicates some of the functionalities are missed. Here the DUT is having certain limitations, (i) DUT supports only 32 bit size and (ii) DUT doesn't support retry/split response. These cover points are not hit due to the limitation and therefore the coverage is less than 100%.

CONCLUSION

This project is aimed at performing Verification of multi master amba ahb protocol using System Verilog and UVM methodology. Test cases were coded to check each of the features of AHB including multi master operation, single and all types of burst operations both read and write, locked transfer operation, error case like invalid address. Coverage model was also implemented to analyze the functional coverage report. Overall average functional coverage value was obtained as 82.95%. This highlights few limitations of DUT and calls for improvement of the DUT.

REFERENCES

1. ARM AMBA 2 AHB Protocol Specification AHB2.
2. www.accellera.org, Universal Verification Methodology 1.1 User's Guide.
3. www.verificationacademy.com, System Verilog and UVM Guidelines.
4. ARM AMBA protocol specifications and design tools [www.arm.com]
5. S. Aravind Babu, S. Babu Ramki and S. Sivanantham, Design of Parallel Architecture Co-Processor for Particle Swarm Optimization Algorithm, Indian J. Sci. Technol., **8(36)**, Art. No. 90316 (2015).
6. C. Patel, M. Srikanth, K. C. Kumar and S. Sivanantham, Monte-Carlo Black-Scholes Implementation Using Open CL Standard, Indian J. Sci. Technol., **8(36)**, Art. No. 90318 (2015).
7. J. More, R. Suryavanshi, G. Dasarwar, S. Sivanantham and K. Sivasankaran, FPGA Implementation of Universal Asynchronous Transmitter and Receiver, IC-GET 2015, Proceedings of 2015 Online International Conference on Green Engineering and Technologies, Art. No. 7453796 (2015).
8. J. Jean Jenifer Nesam and S. Sivanantham, An Efficient Single Precision Floating Point Multiplier Architecture Based on Classical Recoding Algorithm, Indian J. Sci. Technol., **9(5)**, Art. No. 87159 (2016).

Accepted : 11.10.2016