Design and Verification of AMBA AHB

Perumalla Giridhar
Department of ECE,
NIT Agartala
Agartala, India
ravigiridhar786@gmail.com

Dr Priyanka Choudhury
Department of ECE,
NIT, Agartala,
Agartala, India
priyanka.choudhury22@gmail.com

Abstract: **The AHB (Advanced High-performance Bus) is a high-performance bus in AMBA (Advanced Microcontroller Bus Architecture) family. It is a standard for intercommunication of modules in a system. AHB standards are defined by ARM and supports the communication of on-chip memories processors and interfaces of off-chip external memory. In this paper we present, design and perform verification of AHB which support one master and four slaves. In this work, the design of the AHB Protocol is developed comprising of the basic blocks such as Master, Slave, decoder and multiplexers. This AMBA-AHB protocol can be used in any application provided the design should be an AHB compliant. The building blocks of the design master, slaves, decoder and multiplexers are developed in Verilog. The verification environment is developed in system Verilog (SV). QuestaSim (Advanced verification tool from Mentor Graphics) is used to simulate and verify the design and calculate code and functional coverages.**

Keywords- AHB, AMBA-AHB, QuestaSim, ARM

I. INTRODUCTION

Advanced Microcontroller Bus Architecture defines the on-chip communication standard for designing high-performance embedded microcontrollers. Three types of buses are defined in AHB specification by ARM, they are Advance High Performance Bus (AHB), Advanced System Bus (ASB) and Advanced Peripheral Bus (APB). In this paper we limit our discussion to AHB.

An AMBA-based microcontroller contains a high-performance system bus (AMBA AHB or ASB), on-chip memory, CPU cores, and DMA (Direct Memory Access) devices. APB is a secondary bus or peripheral bus that provide communication between low bandwidth devices (peripheral devices). There is a bridge between high-performance bus and peripheral bus for translation of signal standards. Figure 1 represents a generic AMBA-based microcontroller.

Bus Cycle in the AMBA AHB protocol description is defined from rising edge to rising edge transitions**. AHB master** is able to initiate read and write operations by providing an address and control information. Only one bus master is allowed to actively use the bus at any one time. **AHB slave** responds to a read or write operation within a given address-space range. The bus slave signals back to the active master the success, failure or waiting of the data transfer. **Bus transfer** is read/write operation between master and slave which may take one or more bus cycles. **AHB decoder** is used to decode the address of each transfer and provide a select signal for the slave that is involved in the transfer. A single centralized decoder is required in all AHB implementations.

After designing the AHB supporting one master and four slave. We have developed a verification environment in SV to verify the read/write transactions between the master and slave for randomized address and control signals in a given range and constraints so that maximum space of the input output combinations are verified. The intent of functional verification is to make sure that the design is working as intended. Functional verification is measured using coverage. There are two coverages in functional verification, they are code coverage and functional coverage. Code coverage is the amount of code triggered during the verification. It can help finding dead code and false paths which is not triggered for any given combination of the inputs. Functional coverage tells the percentage of transactions which are successful.

The article is structured as follows. Section II contains the design of the system. Section III the verification of the design. Section IV shows the verification results. Section V explains the future scope of our system, and concludes the article.

II. DESIGN

A. AMBA AHB master

AMBA AHB master is designed using Finite State Machines (FSM). It is three states FSM. Address and control signal information is provided by the master to initiate read/write operation. Since we are considering only single beat transactions in this project. Some of the signals can neglected. The simplified master interface is shown in Figure 2. It consist of three states IDLE, READ, WRITE Figure 3 shows the state diagram of master FSM.

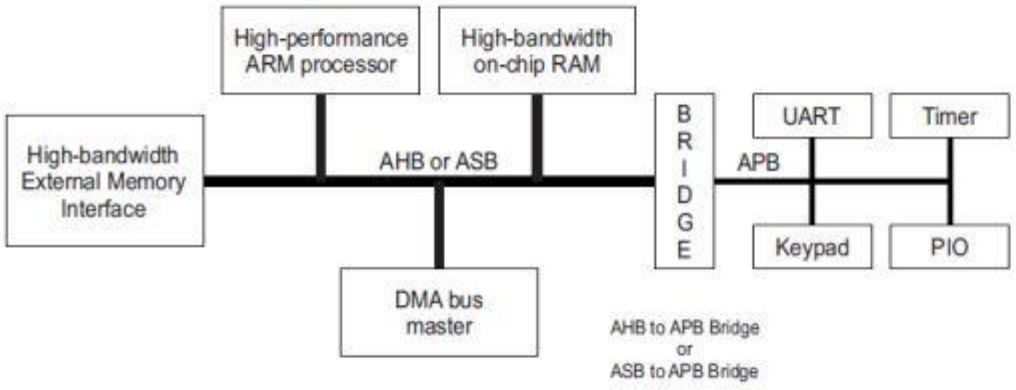

High-performance ARM processor
High-bandwidth on-chip RAM
High-bandwidth External Memory Interface
AHB or ASB
B R I D G E
APB
UART
Timer
Keypad
PIO
DMA bus master
AHB to APB Bridge or ASB to APB Bridge

Figure 1. Typical AMBA based Micro-controller

Initially, the master fsm will be IDLE state. Depending on type of transaction and control signals the master goes to either READ state or WRITE state. If it is a read transaction (HTRANS = 1 and HWRITE =0) the master fsm goes to READ state. if it is a write transaction (HTRANS =1 and HWRITE =1) the master fsm goes to WRITE state. In READ/WRIE state the master specifies the address through HADDR and also forwards the control signal to the slave, so that the slave understands the type of transaction. In read transaction the master samples the read data from HRDATA signal only when HREADY signal from slave is high. In write transaction the master extents the availability of the address and control signal information until the slave gives the HREADY signal. Once the HREADY signal is made high by the slave the master goes back to the IDLE state.

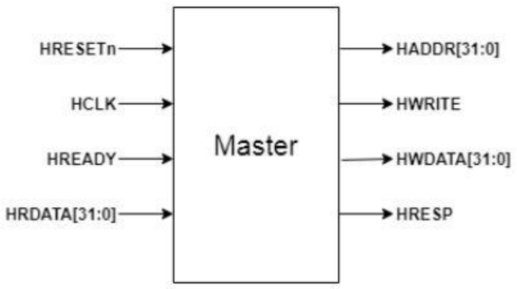


Figure 2. Master Interface Signals

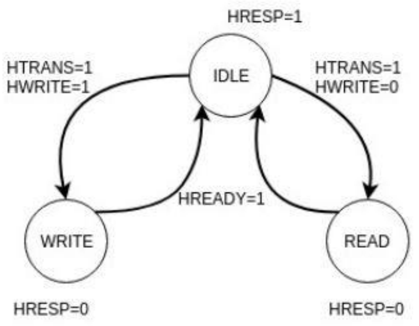


Figure 3. Finite State Machine Diagram for Master

This FSM should be integrated with CPU's and processors to make them AHB compatible.

B. *AMBA AHB slave*

AMBA AHB slave is also designed using Finite State Machines (FSM). It is three states FSM. A slave responds to the transactions initiated by the master. The slave uses the HSELx select signal from the decoder to control when it responds to a bus transfer. Since we are considering only single beat transactions in this project. Some of the signals can neglected. The simplified slave interface is shown in Figure 4. It consist of three states IDLE, READ, WRITE Figure 6 shows the state diagram of master FSM.

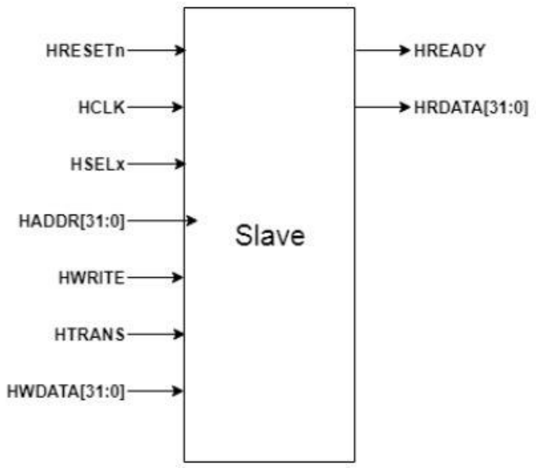


Figure 4. Simplified Slave Interface

Initially, the slave will be inactive. The slave fsm will get activated by the HSELx signal. The decoder component in the bus is responsible for making the HSELx signal high for the particular slave based on the address (HADDR). The last two bits of the address signal (HADDR) represents the slave that

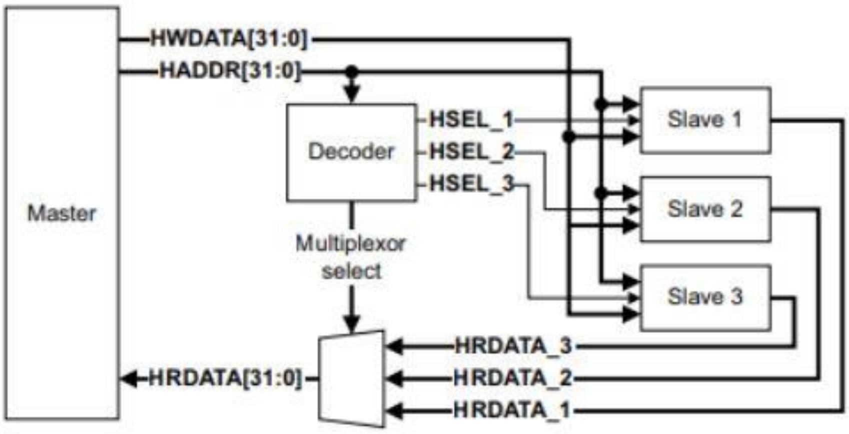


Figure 5. Single master AHB system diagram

has to participate in the transaction. The decoder will decode the last two bits of the address and activates that slave with HSELx signal. Once the slave is activated, based on the control signals (HTRANS and HWRITE) sent from the master the slave fsm goes to read/write state. If it is a read transaction (HTRANS = 1 and HWRITE = 0), the slave will decode the address and sends the data in that address to the master through HRDATA signal and also makes the HREADY signal high indicating that the data in HRDATA is valid. If it is a write transaction(HTRANS = 1 and HWRITE = 1) the slave will write the data present in the signal HWDATA into the address location specified by the HADDR signal and makes the HREADY signal high indicating the end of transaction.

When master receives data HREADY from slave it finishes the transaction and makes HRESP high. When HRESP is made high the slave goes back to IDLE state from either READ or WRITE state. This FSM should be integrated with memory modules like RAM to make them AHB compatible.

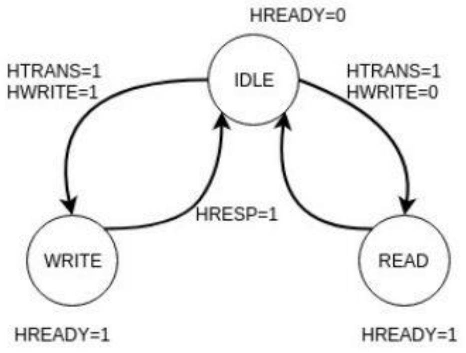


Figure 6. Finite State Machine Diagram from Slave

C. Single master with three slaves system

Figure 5 depicts an AHB system design with the one AHB master and three AHB slaves. The bus interconnects logic consists of one address decoder and one slave-to-master multiplexor. The decoder monitors the address from the master so that the right slave is selected and the multiplexor routes the corresponding slave output data back to the master. AHB also supports multi-master designs by the use of an interconnect component that provides arbitration and routing signals from different masters to the appropriate slaves. Figure 5 shows only the main address and data buses and typical data routing. Not all signals are shown. In our design we have designed one master four slave system.

III. VERFICATION

In this project Verification Environment is developed using System Verilog language. Verification using environment is called as simulation based verification. Different components of general verification environment, depicted in Figure 7 are as follows.

Packet is an object which carries the values of inputs and outputs at a given time. Packet class defines the type of input output it should carry.

Generator generates the packet with input randomized to values defined in the constraints and pushes it to the driver.

Driver gets the packet from generator and drives it into other environment components as required. Here it drives into the DUT (Design Under Test) and Scoreboard simultaneously.

Scoreboard collects the packets from driver and calculates the expected output. This output is compared with output of the Design that has to be verified. Scoreboard decides the success/failure of a transaction and stores the result in a log file.

Monitor gets the output signals from the design under verification and converts into packets to send it to the scoreboard.

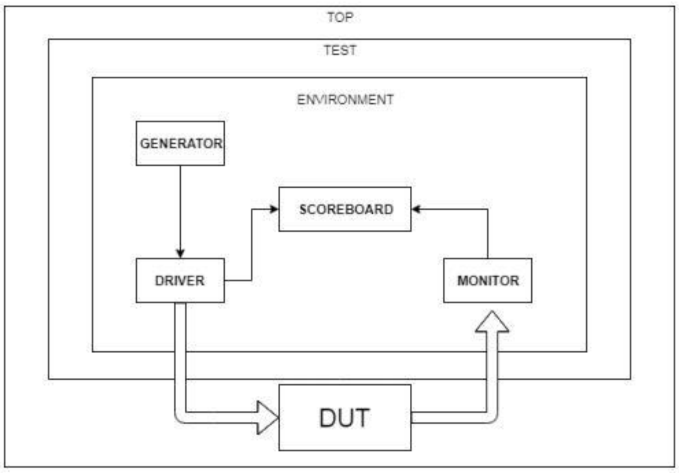


Figure 7. Generic verification environment

Architecture of Verification Environment (Env) with system included in it is shown in the Figure 8. It has all the components as mentioned earlier. The environment interacts only with the master. It makes the master generate address and control signals and collects the signals sent from the slave. Since these are only FSM's which should be integrated with actual on-chip components like CPU's and memory here we are integrating master with verification environment and slave with another logic which can generate data for read transactions and store data from write transactions. For example if the address of read transaction is 11 in decimal it adds one to this value and sends 12 in decimal as HRDATA signal. This same logic is included in the scoreboard so that it can compare that value with the value sent by the slave. If the values are same then read transaction is successful. The priority of the slave requests are processed in first-come, first-served basis.

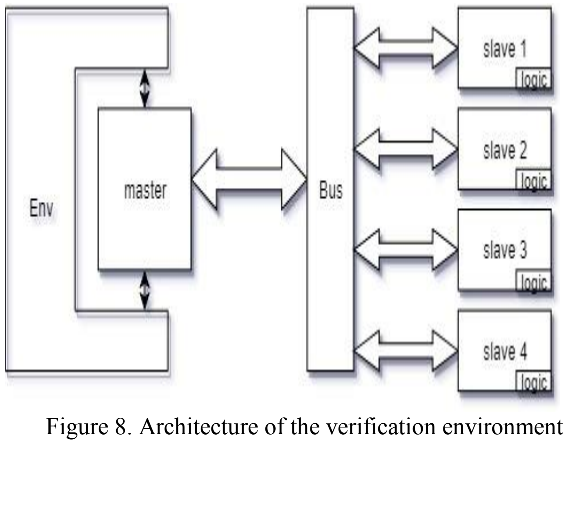


Figure 8. Architecture of the verification environment

IV. RESULTS

Figure 9 shows the simulation result of one master and four slave system. The result shows three consecutive read transactions with HADDR as 11. The logic in slave 1 gives the HRDATA as 12. Only when HREADY signal is high the value 12 is sampled and into the master with rd_data signal. To carry out multiple transactions more than 100 forcing the inputs no more helps. So, we develop verification environment.

A. Scoreboard Results (Functional Coverage)

If all the transactions are successful then functional coverage is 100%. Figure 10 shows a snippet of scoreboard results saved in a log file.

B. Code Coverage

Code coverage is calculated by the Verification Tool QuestaSim. Figure 11 shows the code coverage of the design. By improving the test plan and changing the constraints code coverage values can be increased up to 100%. This is called constraint random based verification.

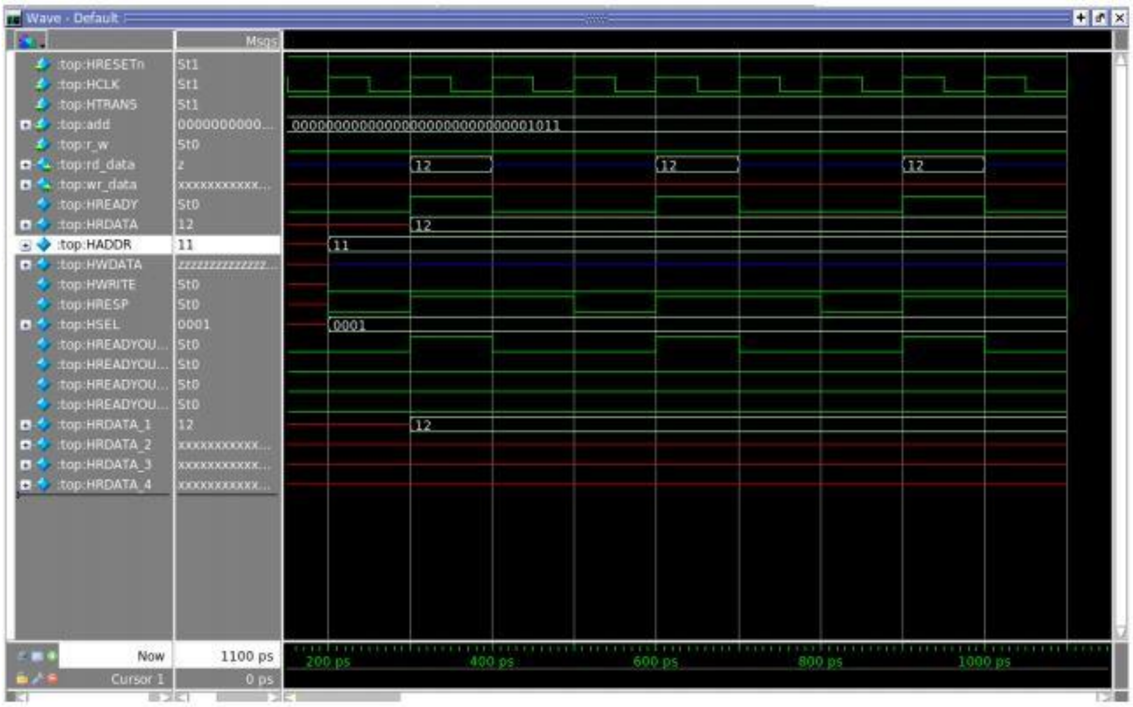
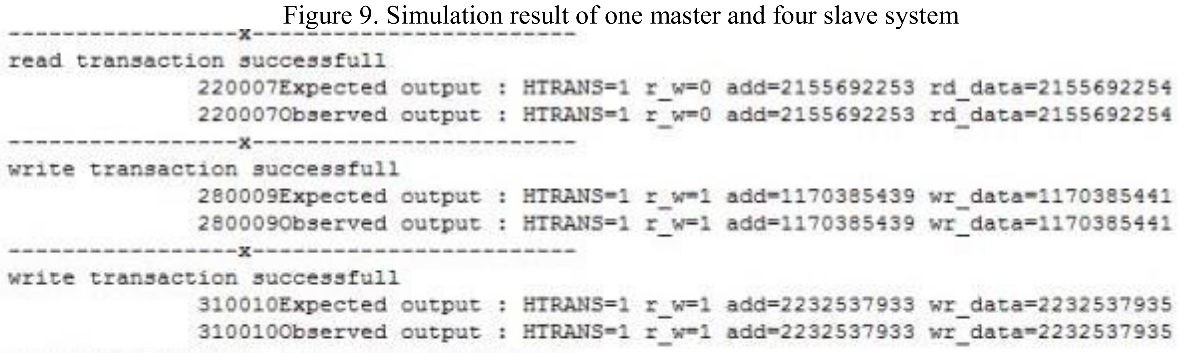
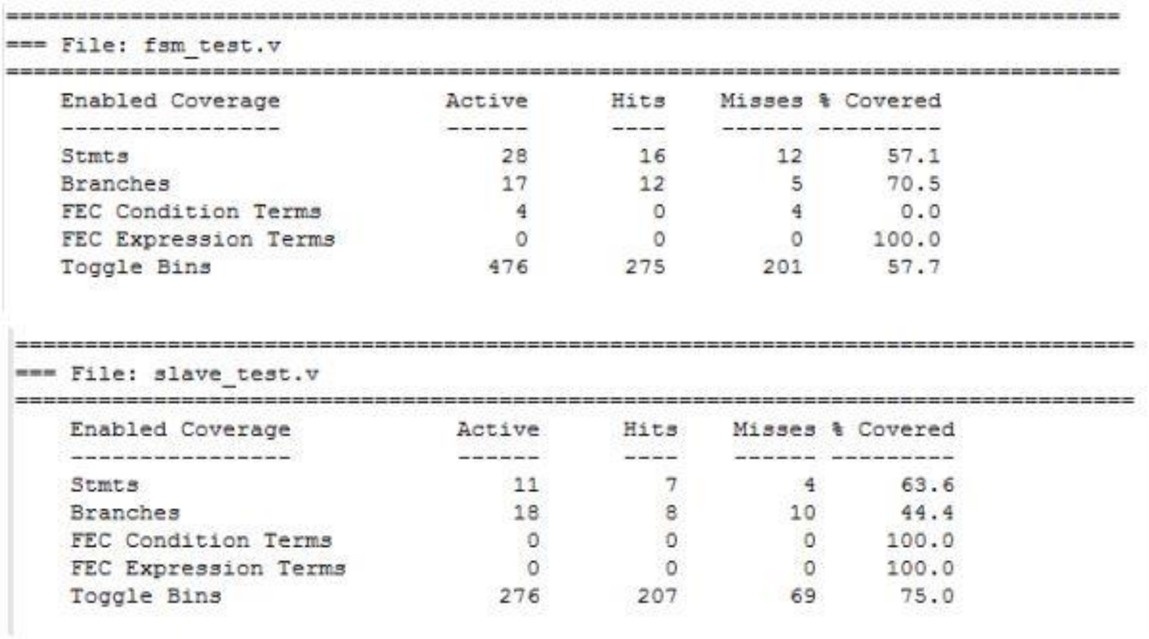


Figure 9. Simulation result of one master and four slave system



```
-----------------x------------------------
read transaction successfull
              220007Expected output : HTRANS=1 r_w=0 add=2155692253 rd_data=2155692254
              220007Observed output : HTRANS=1 r_w=0 add=2155692253 rd_data=2155692254
-----------------x------------------------
write transaction successfull
              280009Expected output : HTRANS=1 r_w=1 add=1170385439 wr_data=1170385441
              280009Observed output : HTRANS=1 r_w=1 add=1170385439 wr_data=1170385441
-----------------x------------------------
write transaction successfull
              310010Expected output : HTRANS=1 r_w=1 add=2232537933 wr_data=2232537935
              310010Observed output : HTRANS=1 r_w=1 add=2232537933 wr_data=2232537935
```

Figure 10. Functional coverage results of the design



```
=================================================================================
=== File: fsm_test.v
=================================================================================
    Enabled Coverage            Active      Hits    Misses % Covered
    ----------------            ------      ----    ------ ---------
    Stmts                           28        16        12      57.1
    Branches                        17        12         5      70.5
    FEC Condition Terms              4         0         4       0.0
    FEC Expression Terms             0         0         0     100.0
    Toggle Bins                    476       275       201      57.7


=================================================================================
=== File: slave_test.v
=================================================================================
    Enabled Coverage            Active      Hits    Misses % Covered
    ----------------            ------      ----    ------ ---------
    Stmts                           11         7         4      63.6
    Branches                        18         8        10      44.4
    FEC Condition Terms              0         0         0     100.0
    FEC Expression Terms             0         0         0     100.0
    Toggle Bins                    276       207        69      75.0
```

Figure 11. Code coverage results of the design

V. CONCLUSION

AMBA AHB protocol defines the communication standards for high performance embedded systems. In this work we have design and verified the system depending upon the specifications, data transfer that are supported by AMBA bus architecture. Verification of AHB Protocol for and Single Master - Four Slaves has been verified by developing the Verification IP using system Verilog using constraint random based verification technique. The simulation and verification result shows that the communication between master and slave fsm's using AHB protocol is as intended. All of the address and data signals are successfully transferred from master fsm to slave fsm following AMBA-AHB protocol. There is no loss of address, data and control signal information. The master fsm should be integrated with modules in the system which tend to initiate the transactions and slave fsm should be integrated with memory modules in the system to make the system AHB compatible. The Verification environment developed using tool Questa and it is sure that developed VIP is also compatible with other verification tools from Cadence, Synopsys etc.

In the present work we have designed AHB system which support only one master four slaves. Developing the system which can support 16 masters and 16 slaves with burst transactions compatible could be the future work of this of this project in the design. In verifying the system, developing VIP (Verification Intellectual property) using standard methodologies like UVM (Universal Verification Methodology) rather than using a generic verification environment would be appreciable. VIP's developed in UVM could be reusable for any design with minimal changes in the architecture. The verification environment developed in this work is not reusable for verifying other designs.

REFERENCES

[1] ARM Limited. AMBA specification (rev 2.0), 1999.

[2] M.D. Nguyen, M. Thalmaier, M.Wedler, D. Stoffel, and W. Kunz. A reuse methodology for formal soc protocol compliance. In Proc. Forum on Specification & Design Langliages(FDLJ. Sophia Antipolis. France. September 2009.

[3] Minh D. Nguyen, Max Thalmaier, Markus Wedler, Jorg Bormann, Dominik Stoffel. and Wolfgang Kunz. Unbounded protocol compliance verification using interval property checking with invariants. IEEE Transactions on Computer-Aided Design, 27(11):2068- 2082, November 2008.

[4] Chrisspear, system Verilog for verification, New York: springer, 2006. Rath A.W, Esen.V and Ecker.W , A transaction oriented UVM-based library for verification of analog behavior Publication Year: 2014 Page(s): 806 811

[5] Soo-Yun Hwang and Kyoung-Sun Jhang, An Improved Implementation Method Of AHB BusMatrix SOC Conference 2005, Proceedings, IEEE International pp. 211-214

[6] Mulani, Level Verification Using SystemVerilog Emerging Trends in Engineering and Technology (ICETET), 2009 2nd International Conference on 16-18 Dec.2009 pp.378- 380

[7] Pockrandt, M. Herber, P and Glesner, S, Model checking a SystemC/TLM design of the AMBA AHB Protocol Embedded Systems for Real-Time Multimedia (ESTIMedia), 2011 9th IEEE Symposium on 13-14 Oct.2011 pp.66 75

[8] IEEE Draft Standard for System Verilog - Unified Hardware Design,Specification and Verification Language, IEEE P1800/DS, February2012 pp.1-1304

[9] Soo Yun Hwang, Dong Soo Kang, Hyeong Jun Park, Kyoung Son Jhang, "Implementation of a Self-Motivated Arbitration Scheme for the Multilayer AHB Busmatrix", IEEE Transactions on Very Large Scale Integration (VLSI) Systems (Volume: 18 , Issue: 5 , May 2010)