International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 17 (2017) pp. 6534-6541
© Research India Publications. http://www.ripublication.com

Verification IP for AMBA AXI Protocol using System Verilog

Murali .M.
M.Tech (VLSI Design), School of Electronics Engineering (SENSE),VIT University Chennai Campus, Chennai, India.

Umadevi. S
Assistant Professor (Sr.G), School of Electronics Engineering (SENSE), VIT University Chennai Campus, Chennai, India.
Orcid Id: 0000-0001-7742-9209

Sakthivel.S.M.
Assistant Professor (Sr.G), School of Electronics Engineering (SENSE),VIT University Chennai Campus, Chennai, India.
Orcid Id: 0000-0002-8486-3238

Abstract

The growth of CMOS technologies has increased the design of more complex digital systems using reusable IP's. This lead to the process of functional verification more and more complex. In this scenario, the verification engineers come with an inbuilt verification environment called as verification IP(Intellectual Property) Cores to reduce the time span of verification and increase the functionality check accuracy. This verification environment accurately monitors and captures the functional errors for all the cases and makes the process of design/verification so easy. Hence in this research article, the Advanced Microcontroller Bus Architecture is designed and a verification environment for 100% functional verification is also proposed. In this verification methodology the individual modules of the AXI and verification environment are designed using System Verilog HDL and simulated using Mentor Graphics Questasim®. In this verification scenario, the functionality is verified for five different test-cases and then with a code coverage enabled verification process. For all the verification test-cases the performance of the bus is assessed by the calculated values of Valid Count, Busy Count and Bus Utilization factor.

Keywords: Verification-IP, AMBA-AXI Protocol, System Verilog, Verififcation Environment.

INTRODUCTION

The rapid growth in CMOS technology and Computer Aided Design leads to usage of large number of IP (Intellectual Property Cores) in the complex digital designs[1]. With the help of these IP cores integration, Nowadays the SOC (System on Chip) design becomes more popular and intensively used for many applications[2]. Also all the SOC design uses bus protocols for making the data communication and synchronization[4-6]. Hence the reusability of large number of IP cores in the complex design makes the functional verification process so crucial (i.e. as it involves 70% percent of the time span as compared to the 30% of the time span for design process). In order to overcome this difficulty and large time span the verification engineers proposed a methodology for verifying the functionality of the chip using an inbuilt verification environment called as Verification IP (VIP)[7-10].

Normally the bus protocols used in the modern SOCs are APB (Advanced Peripheral Bus), AHB (Advanced high performance Bus) and AXI (Advanced Extensible Interface). On comparing these three bus protocols, the AXI bus protocols gives better performance and consumes moderate power. Hence all the SOC design involves the usage of AMBA AXI bus in the internal architecture. Normally in the functional verification, the failure mode analysis and fault robustness is performed for the design. For overcoming the slag in the functional verification process, a verification environment based methodology is proposed using System Verilog in coverage enabled mode[11-17]. In this verification mode environment, the two operation are carried out as i) Proper test-cases are defined for the DUV(Device under Verification) & ii) Determine the number of test-cases to analyze and cover all the functionalities of the design. During the verification process, the coverage enabled mode easily identifies the bugs in the design by identifying the coverage of the code and validating the functional behavior of the design for all test scenarios.

The rest of the paper is organized as follows: i) AMBA Bus Architecture, ii) Proposed Verification Environment, iii) Verification Plan and iv) Results and Discussion.

AMBA-AXI BUS ARCHITECTURE

AMBA (Advanced Microcontroller bus architecture) is an on chip bus protocol from ARM. It tells about the on chip interconnect specifications for the establishing the connection between the processor, functional blocks and peripherals. The AXI bus protocol architecture is the most suitable and usable in modern SOCs and FPGA. This is due to the characteristics of large bandwidth capability, interfacing with complex bridges, large memory access and high backward compatibility with peripherals. Also the AXI bus architecture is having separate read and write data phases with individual data channels, unaligned data transactions and burst mode access. This section explains the bus architecture of AXI protocol used for the data communication and synchronization operations. In this bus architecture the data transactions are established in two phases as i) Read Transaction phase and ii) Write Transaction phase in the bus. The corresponding phases of operation are explained in the subsections with the master – slave configurations.

Read Transaction Phase

In the read transaction phase, the read address (AR) and read data response(R) signals are involved in the data communication and synchronization process. In the read transaction the first the valid address (AR)and Read channel access is communicated by master to initialize the transactions and then the slave responds with a signal AR ready(i.e making the address ready for the communication). After the reception of AR ready the master checks the address and then receives the RVALID and read channel data signals. Following to that the master establishes the channel communication with control signal with a valid RR ready and establishes the read transactions. The entire architecture/operation of the read transaction is pictorially represented in the fig.1.

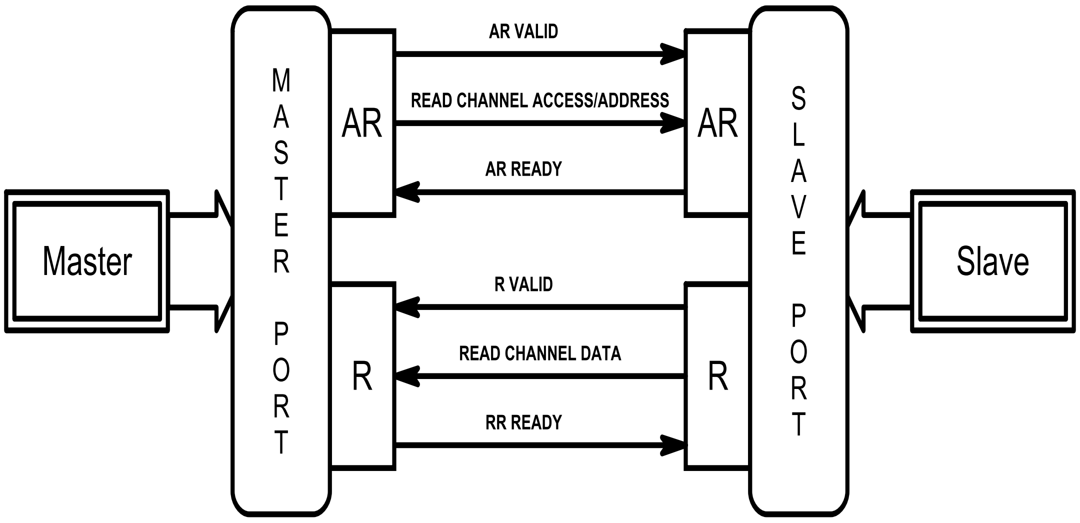


Figure 1: AXI Read Transactions

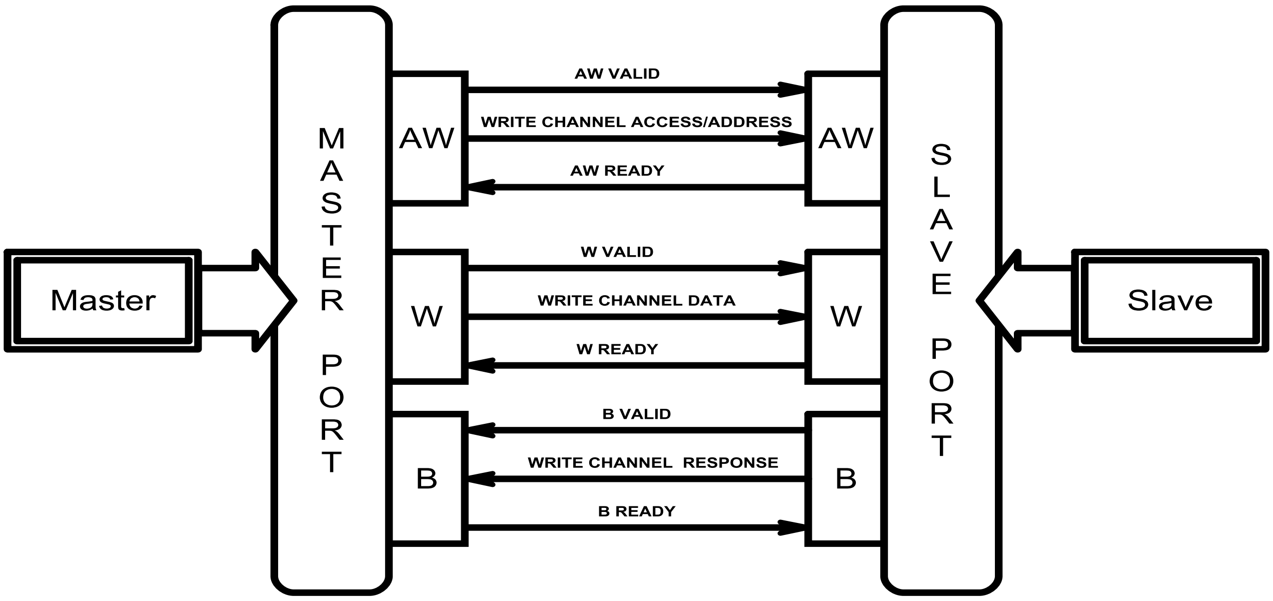


Figure 2: AXI Write transactions

Write Transaction Phase

The write transactions are happening in the following three modes as i)Write address channel, ii)Write data channel and iii)Write response channel. In the write address channel mode, the master starts the communication with the control signals AW VALID and Write channel access/address to the slave(i.e it sends the valid address and channel control and address). Immediately the slave responds with a valid AW ready. In the same manner during the Write data channel mode, the master initializes the transactions with a WVLAID and Write channel data, following to that the slave acknowledges by a valid WREADY signal. Similar to the data and address mode, the Write response channel mode involves the transaction started by master with the control signals BVALID and write response channel. Upon receiving the control signal, the slave establishes the communication by a valid BREADY signal. The operation of the write transactions are shown in the fig.2

PROPOSED VERIFICATION ENVIRONMENT

The verification environment is a standard method of verifying the DUV in code coverage mode. The following components are present in the proposed verification environment i) Test-suite, ii) Virtual sequencer, iii) Sequencer, iv) Driver, v) AXI-Monitor, vi) Protocol checker, vii) Bus Monitor, viii) Coverage Collector and xi)Scoreboard.

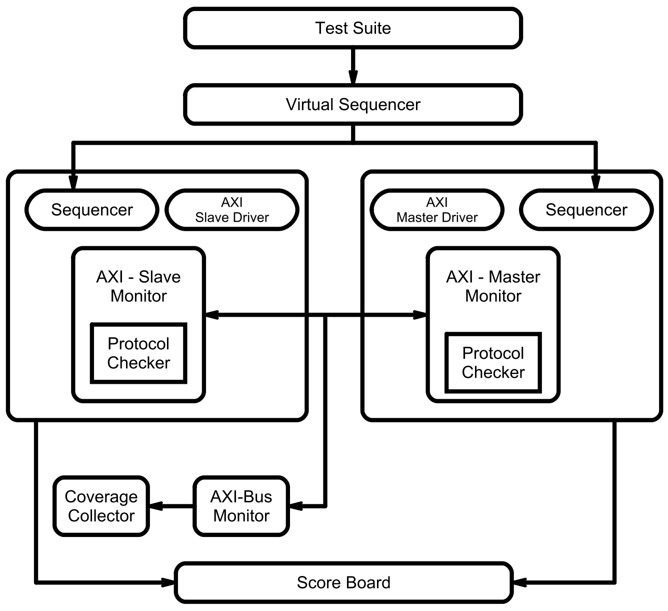


Figure - 3 Verification Environment

Test-Suite

Test-suite consists of all the test-cases for checking the specific key features of the DUV and the functionality in a constrained random mode. The test-case is of the form constrained random mode or direct random test-case. The test is regressively performed on the design and analyzed. It normally contains set of stimulus for running the design.

Virtual Sequencer

Generally the sequencer generates random sequences for verifying the key properties of the design. The generated sequence is called as top level sequence and splitted into corresponding level base and primitive sequences. Thus the virtual sequencer verifies the functionality of each sub-component by driving/forcing them with the set of sequences.

Sequencer

Sequencer is a component, which generates set of sequences. The generated sequences are of two types active and passive. The active sequence drives the signal whereas the passive does not drive the signal. The sequencer aligns and generates the sequence accordingly to the criteria of functionality checks.

Driver

Driver is the component which interfaces or forces all other components to respond accordingly to the feeded stimulus (i.e simply it drives the input to the corresponding blocks).

Monitor

The monitor watches all the data transactions of the all the components and reports the result to the coverage collector. It has a TLM port for analysis and virtual interface to handle the points of the DUV signals.

Protocol checker

In this module, the DUV functionality is verified, the interface compliance is also checked with the temporal behavior of the inputs. Also the responses are monitored for all the combinations of the input sequences.

Bus monitor

Bus monitor screens the functional operation of individual components and connects the master/slave monitor. Finally the transactions are collected in the coverage collector as database.

Coverage Collector

Coverage collector collects all the code coverage information like branch, statement and FSM in the DUV is covered by the specified test-case or not. Normally the verification engineers aim for 100% of coverage during the functional verification.

The above prescribed components are used in both the master and slave modes in the AMBA AXI bus architecture. The individual modules are modeled in system Verilog and integrated as a main top module. The system Verilog instantiation for individual modules and top modules are given as follows

```
module top;
  reg clk, rst;
  axi_intf vif(clk, rst);
  axi_slave dut(
        clk,
        rst,
        vif.awid,
        vif.awaddr,
 . . . . . . .
 . . . . . . .
);
  axi_tb tb();
  axi_assertion axi_assert();
 . . . . . . .
 . . . . . . .
 . . . . . . .
  initial begin
        $value$plusargs("test_no=%d",
axi_config::test_no);
        ->axi_config::e;
  end
endmodule
```

In the same manner the AXI Master/Slave and the individual components of the verification environment are modeled using System Verilog. The instances of the AXI top module pseudo-code is given partially as above. The above prescribed Verification Environment is used for the functional verification of the AMBA Bus protocol.

Verification Plan

The verification plan tells the property has to be verified and drives the coverage criteria to be satisfied. A good plan contains measurable metrics and successful test-case to achieve the coverage and functionality check. The plan of verification tells what to be verified, how to verify it. In this AMBA AXI verification the properties to be verified are i)System connectivity during read and write cycle, ii)Transaction routing and iii)finally the Data integrity. In the system connectivity checking test, the system integration is monitored by the corresponding acknowledgement signals. Then during the routing stage the bus data occupancy structure is verified by the bus utilization and efficiency. Finally the data integrity is noted for every transactions and the functionality check pass is carried out for each of the stage.

In this AXI protocol, the verifications plan focuses the coverage of the following scenarios as

- Read phase
- Write Phase
- Write & Read Phase from same address
- Write & Read Phase from different address
- Overlapped transactions

For the corresponding scenarios the test-cases are generated and the simulation is carried out for the functionality verification. Using the following bus performance metrics Valid Count, Busy Count and Bus Utilization factor, the performance of the bus is assessed. If the bus is not verified accurately in the corresponding test-cases, then the test-cases are modified accordingly to reach the specific property checks.

Also in the verification, the code coverage based verification mode analysis is carried out for the DUV; if the 100% coverage is not achieved during the analysis the test-case is altered to cover all the code parts of the design.

RESULTS & DISCUSSIONS

This section discusses the simulation results of the AMBA AXI protocol for the five different test-scenarios and the performance assessment using the quality metrics for the data metrics. For carrying out the simulation analysis, fist the individual modules AXI Master/Slave is modeled using System Verilog and integrated with the modeled test-environment for the verification. In the verification environment, the DUV is integrated with sequencer, interface, driver and monitor to assess the performance in constrained mode. After the integration of the top module with DUV the simulation is carried out using the Mentor Graphics QuestaSim®. Here the test-case is written for five different scenarios as i) Read phase, ii) Write phase, iii) Read & Write from the same address, iv) Read & Write from the different address and finally v)Overlapped Transactions.

Verification of Read Phase

In this phase the read data transactions of the AMBA AXI protocol is verified with the Master/Slave configuration mode. During the ready cycle transactions the signals ARVALID, ARREADY, RVALID, RREADY, RLAST, RDATA and ARSIZE are monitored and verified. Here the functionality check primarily involves two modes of operation namely i) Read Address and ii) Read Response channel. For the every positive edge of clock, the read address channel will fetch the address at the high state values of ARVALID and ARREADY. Similarly after a time delay, the response will be instantiated to high mode with the high values of the signal on RVALID and RREADY. The last transaction of the read operation is indicated by the high values of RLAST. The values of ARSIZE and ARLEN are measured and indicate the number of transactions occurred in read mode. For verifying the read phase a test-case is generated and simulated to test the functionality of the read operation. The working of the read mode includes two channels as explained above with the measure of the quality metrics for the read address, read channel data with channel response. The entire operation of the read phase is pictorially represented as a waveform in the fig.4

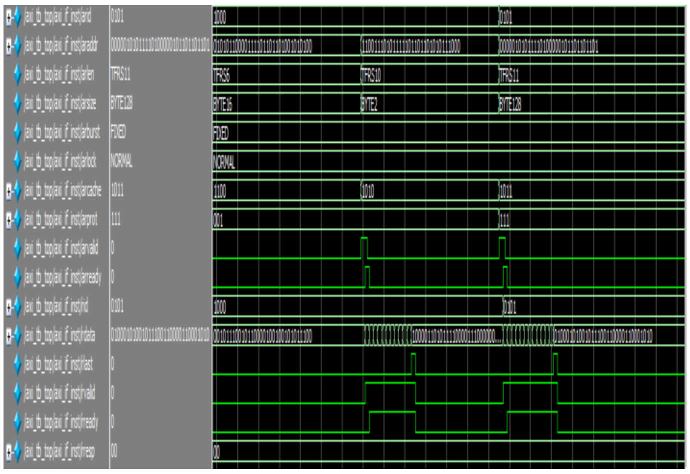


Figure 4: Read Phase Simulation Response

The performance assessment using the quality metrics in Read Phase is given as

- VALID COUNT = 10
- BUSY COUNT = 13
- BUS UTILIZATION = (10/13)*100 = 76.92 %

Verification of Write Phase

Similar to the Read Phase, the Write Phase of the AMBA AXI protocol involves the verification of AWID, AWADDR, AWLEN, AWSIZE, and AWVALID AND AWREADY signals in Master/Slave configuration. The primary operation of the Write Phase involves three major operation as i)Write Address Channel, ii) Write Channel Data and iii)Finally Write response Channel. For every positive clock cycles, the signals WVALID and WREADY takes the high logic state for fetching the valid write address with an acknowledgement AWID. In the same manner, the write response operation will happen with the high transitions of the BVALID and BREADY. The control signal AWLEN holds the values 0000 in the starting and gets incremented for every transaction and the signal AWSIZE indicates the size of the each transactions. The Write Transactions is pictorially represented in the fig.5.

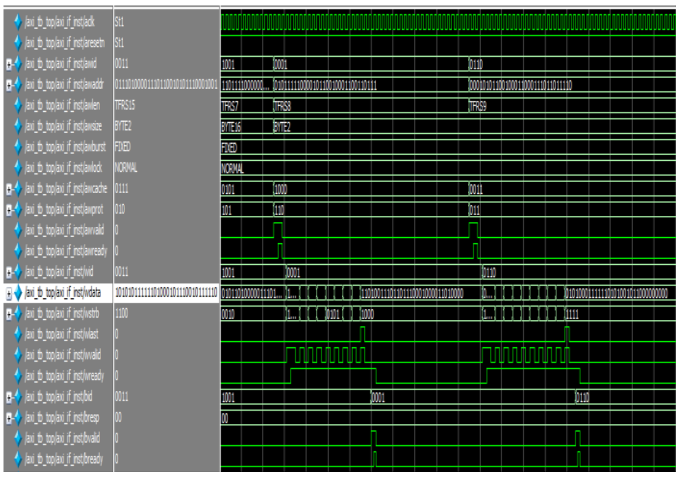


Figure 5: Write Phase Simulation Response

The measure values of the quality metrics for the assessment of Write phase is give as

- VALID COUNT = 13
- BUSY COUNT = 16
- BUS UTILIZATION = (13/16)*100 = 81.25 %

Verification of Read & Write from Same Address Locations

In the Read and Write phase from the same address location both the read and write transactions are occurred in the same address locations. The address for each of the write and read operation is given by the signal AWADDR/ARADDR with a size of 32 bit length in Hexadecimal format. The signal AWADDR indicates the write address and is same for the read operation. Hence the value of ARADDR is same as the AWADDR (i.e. the read and write operation are happening in the same address location). The value of the read data is stored in the signal RDATA, whereas the value for the write operation is stored WDATA. With these effective monitoring of the signals in the test-case the read and write operation in the same address location is verified successfully for the AMBA AXI protocol. The entire verification of the Read & Write phase operation in the same address location is shown in the fig.6.

The performance of the Read & Write operation at the same address location is indicated by the values of quality metrics as

- VALID COUNT = 42
- BUSY COUNT = 44
- BUS UTILIZATION = (42/44)*100 = 95.45%.

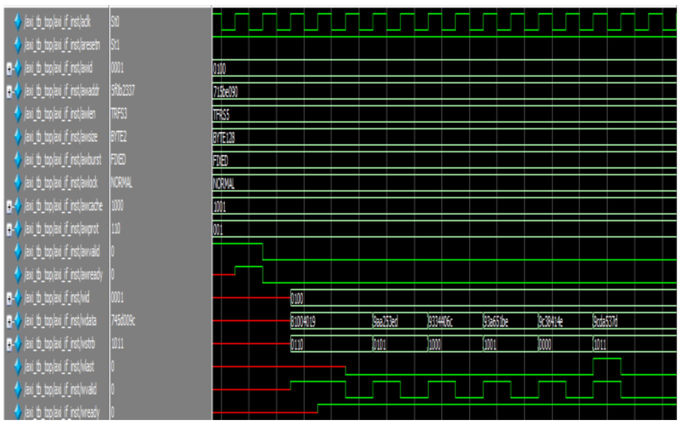


Figure 6: Read & Write Phase in Same Address Location Response

Verification of Read & Write from Different Address locations

Similar to the Read and Write operation in the same address location, the address for each of the read and write operation is incorporated in 32 bit Hexa-Decimal format. Here the values of AWADDR & ARADDR as well as the RDATA & WDATA are different for this verification scenario. For example, the address 000004E9 is considered for write phase and the address 22ABFEC1 for the read operation phase. By having the two different address location it is inferred that the read and write are happening successfully at different memory locations. The simulation waveform for the read and write at different location is shown in the fig.7.

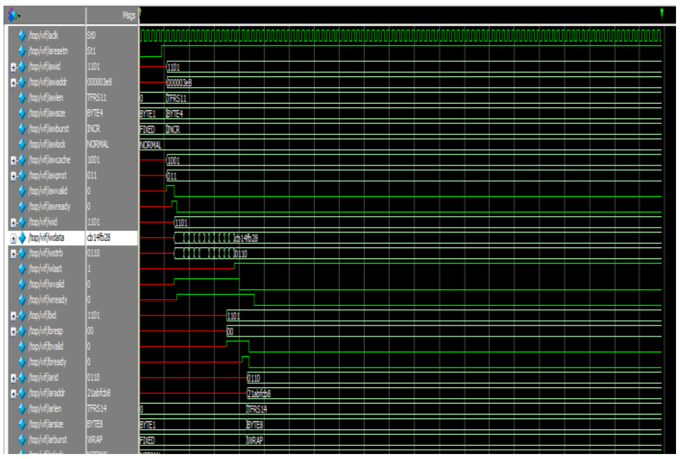


Figure 6: Read & Write Phase in Different Address Location Response

The assessment of the Read and Write operation at different location is validated by the calculated performance metrics is given as

- VALID COUNT = 43
- BUSY COUNT = 45
- BUS UTILIZATION = (43/44)*100 = 95.55%.

Overlapped Transactions

In the overlapped transactions both the read and write will happen the using the shared address in burst mode. Here the AXI protocol is operating in the burst mode of read and write. The burst modes of the overlapped read and write operations for the AXI Bus protocol is shown in the fig7. From the simulation waveform it is inferred that the overlapped transaction is also supported by the AXI bus.

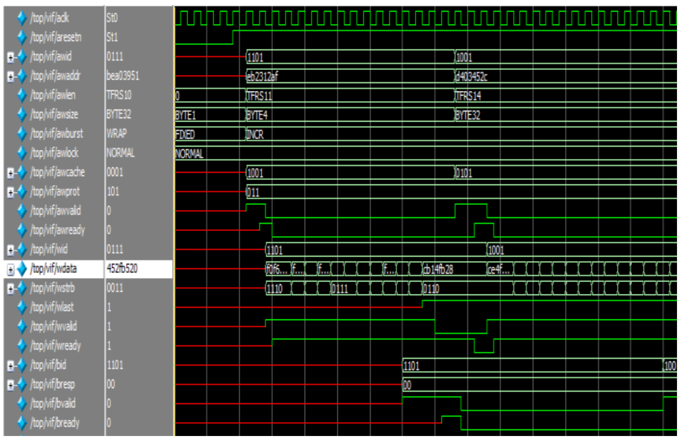


Figure 7: Simulation Response for Overlapped Transactions

From the simulation results, the bus utilization is compared for the different test-case scenarios and plotted as graph in the fig.8. It is inferred that the combined read and write operation occupies the bus efficiently when compared with the individual read and write phase operations. For the individual read and write phase the bus utilization factor value is around 85% only. Then for the combined read and write operation the bus utilization is increased up to 95%.

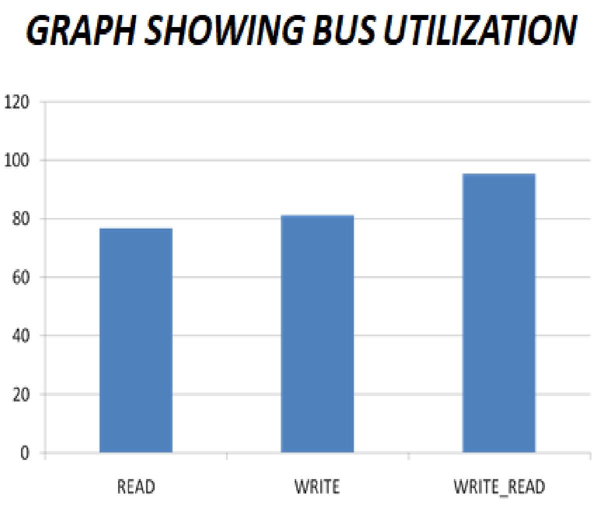


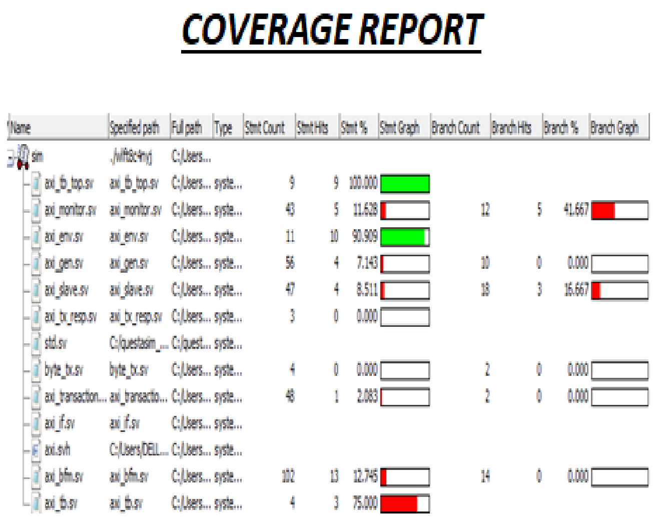
Figure 8: Bus utilization Comparison for different Verification Scenarios

Code Coverage Mode Analysis

The Code Coverage is used as a metric for the progress of the functional verification. In this analysis the design is verified for the functionality and to identify the uncovered areas in the code. The coverage report is automatically collected in the coverage collector automatically. It tells how the test-case is covering the entire DUV. There exist different categories of code coverage as

- Statement Coverage
- Block Coverage
- Expression Coverage
- Branch Coverage
- Toggle Coverage
- FSM Coverage.

Normally in the coverage analysis, the test-case is analyzed for achieving the 100% coverage. Otherwise the test-case is modified for the covering the entire code blocks of the DUV. The code coverage based simulation of the AXI protocol using the Questasim is shown in the fig.9. From the coverage analysis, it is inferred that there is 100% coverage of code for the prescribed test-case. Thus the functional verification of the AXI protocol is carried out in code coverage mode.



COVERAGE REPORT

Name	Specified path	Full path	Type	Stmt Count	Stmt Hits	Stmt %	Stmt Graph	Branch Count	Branch Hits	Branch %	Branch Graph
sim	./wlft8c4nyj	C:/Users...									
axi_tb_top.sv	axi_tb_top.sv	C:/Users...	syste...	9	9	100.000	🟩				
axi_monitor.sv	axi_monitor.sv	C:/Users...	syste...	43	5	11.628	🟥	12	5	41.667	🟥
axi_env.sv	axi_env.sv	C:/Users...	syste...	11	10	90.909	🟩				
axi_gen.sv	axi_gen.sv	C:/Users...	syste...	56	4	7.143	🟥	10	0	0.000	
axi_slave.sv	axi_slave.sv	C:/Users...	syste...	47	4	8.511	🟥	18	3	16.667	🟥
axi_tx_resp.sv	axi_tx_resp.sv	C:/Users...	syste...	3	0	0.000					
std.sv	C:/questasim_...	C:/quest...	syste...								
byte_tx.sv	byte_tx.sv	C:/Users...	syste...	4	0	0.000		2	0	0.000	
axi_transaction...	axi_transactio...	C:/Users...	syste...	48	1	2.083		2	0	0.000	
axi_if.sv	axi_if.sv	C:/Users...	syste...								
axi.svh	C:/Users/DELL...	C:/Users...	syste...								
axi_bfm.sv	axi_bfm.sv	C:/Users...	syste...	102	13	12.745	🟥	14	0	0.000	
axi_tb.sv	axi_tb.sv	C:/Users...	syste...	4	3	75.000	🟥				

Figure 9: Coverage Analysis Report

CONCLUSION

In this research article, AMBA-AXI bus protocol is designed and implemented using System Verilog HDL. Then the simulation is carried out for the functionality check using Mentor Graphics EDA tool. Here the bus is verified for the five different test-case scenarios and also code coverage is performed. Finally the bus efficiency is also calculated for the proposed test environment using the performance metrics.

REFERENCES

[1]. K. Swaminathan , G. Lakshmi narayanan a, Seok-Bum Ko “Design and verification of an efficient WISHBONE-based network interface for network on chip”. Computers and Electrical Engineering 40 (2014) 1838–1857

[2]. S. Saponara a, L. Fanucci a, M. Coppola “Design and coverage-driven verification of a novel network-interface IP macro cell for network-on- chip interconnects”. Microprocessors and Microsystems 35 (2011) 579–592

[3]. Alan P. Su, Jiff Kuo, Kuen-Jong Lee, Ing-Jer Huang, Guo-An Jian” A Multi-core Software/Hardware Co-debug Platform with ARM CoreSightTM, On-chip Test Architecture and AXI/AHB Bus Monitor”.

[4]. Attia B, Zitouni A, Tourki R. Design and implementation of network interface compatible OCP for packet based NOC. In: International conference on design & technology of integrated systems in nanoscale era; 2010. p. 1–8.

[5]. Technical Reference Manual of Prime Cell AXI Configurable Interconnect (PL300), ARM, Cambridge, U.K., 2010.

[6]. Singh Sanjay Pratap, Bhoj Shilpa, Balasubramanian Dheera, Nagda Tanvi, Bhatia Dinesh, Balsara Poras. “Generic network interfaces for plug and play NoC based architecture”. In: Lecture notes in computer science Springer reconfigurable computing: architectures and applications; 2006. p. 287–98.

[7]. Chien-Hung Chen, Jiun-Cheng Ju, and Ing-Jer Huang, “A Synthesizable AXI Protocol Checker for SoC Integration”, IEEE transl, ISOCC, Vol 8,pp.103-106, 2010

[8]. Lai Yong-Long, Yang Shyue-Wen, Sheu Ming-Hwa, Hwang Yin- Tsung, Tang Hui-Yu, Huang Pin-Zhang. A high-speed network interface design for packet-based NoC. In: IEEE ICCCAS; 2006. p.2667–71.

[9]. J. Shao and B. T. Davis, “A burst scheduling access reordering mechanism, “in Proc. IEEE 13th Int. Symp. High Perform. Comput. Archit.,Feb. 2007, pp. 285–294.

[10]. Fattah M, Manian A, Rahimi A, Mohammadi S. “A high throughput low power FIFO used for GALS NoC buffers”. In: IEEE annual symposium on VLSI; 2010. pp. 333–8.

[11]. Anurag srivastava,G.S.tomar, kamal k karla, “Efficient Design and Performance Analysis for AMBA Bus Architecture Based System On

Chip",International conference on Computational Intelligence and communication networks,2010.

[12]. H.W wang,C.S lai,Hwang,and Y-H Lin,"on chip interconnection design and soc integration with open core protocol(ocp),"in proc.IEEE int.symp.VLSI Design 2008,pp. 25-28

[13]. X.Xiao and Lee,"A true parallel deadlock detection algorithm for single unit resource system and its hardware implementation,"IEEE trans.parallel Distrib.syst.,vol. 21,pp 4-19, jan 2010.

[14]. A.T.Tran and B.M.Bass,"Roshaq:High performance onchip router with shared queues ,"in proc.IEEE 29th int.conf.comput.design,oct,2011,pp.232-238

[15]. ARM, 2011.AMBA AXI protocol v.1.0 specification. ARM Limited.http://www.arm.com/.

[16]. G.Mahesh, Sakthivel.S.M." Verification of Memory transactions in AXI protocol using System Verilog Approach", International Conference on Communication and Signal Processing (ICCSP) 2015, Melmaruvathur India.

[17]. G.Mahesh, Sakthivel.S.M." Functional verification of the Axi2OCP bridge using system verilog and effective bus utilization calculation for AMBA AXI 3.0 protocol", International Conference on Communication and Signal Processing (ICCSP) 2015, Melmaruvathur India.