International Journal of Engineering Research and General Science Volume 3, Issue 1, January-February, 2015
ISSN 2091-2730

Verification IP for an AMBA-AXI Protocol using System Verilog

Golla Mahesh[1] , Sakthivel.S.M[2]

VIT UNIVERSITY Chennai Campus, golla.mahesh2013@vit.ac.in, 9092044806

Abstract— In this paper, a coverage driven verification methodology to verify the AMBA AXI Bus protocol with its verification environment is proposed. The whole verification process is carried out using the system verilog based modeling approach. The AXI verification scenario includes the Read and Write transaction phases, which are getting verified with their values of valid count, busy count and bus utilization factor. The functional verification of the AXI is carried out using Mentor Graphics Questa- sim in code coverage enabled mode.

Keywords— Verification IP development, AMBA-AXI protocol, Code coverage, Coverage driven verification, Transactions, System verilog, QUESTA-SIM.

INTRODUCTION

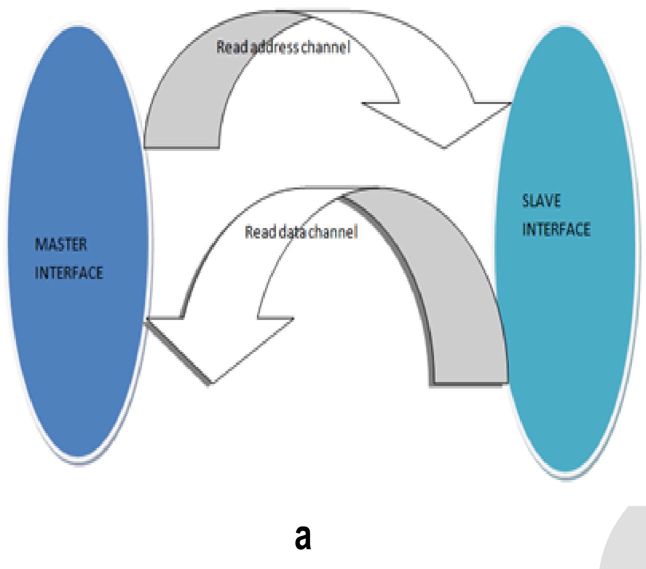
Today's System on Chip (SOC) has many intellectual property cores inbuilt in them and the proper synchronization between the individual cores during the data communication is a crucial task [1]. This modern SOC's majorly use the common bus protocols like advanced peripheral bus (APB), advanced high performance bus (AHB) and advanced extensible interface (AXI) for their synchronized communication[2]. Hence during the development of these kind SOC's the verification of this technologies is very important and a crucial task as it covers 70% time as compared to the design stage which requires only 30% of the time [3]. Due to this large time span for verifying an on chip many engineers are involved in verifying the functional properties and synchronization between them using an inbuilt verification environment called as Verification IP [4]. The bus protocols used in the modern SOC's are classified based on their performance and power consumption. Among the three protocols APB bus structure consumes less power when compared with AHB bus structure but lacks in performance as compared with AHB. The only thing in AHB bus structure based SOC's will have slightly higher percentage of power consumption as compared with APB bus structure. Similarly the AXI bus consumes moderate power and gives a better performance as compared with AHB and APB bus structures [6]. So the AXI bus structure can be selected as an alternative bus standard for the modern SOC design. In this paper a system verilog based Verification IP has been designed for verifying the best of AMBA protocols (i.e. AMBA AXI) using a coverage driven verification methodology [7].

AMBA AXI BUS ARCHITECTURE

This section explains the bus architecture of the AMBA AXI protocol for the data communication and synchronization operation with respect to READ Transaction phase and WRITE Transaction phase in their buses.

READ TRANSACTION PHASE

Generally the read transaction phase is divided into two modes namely channels of read address (AR) and read data plus read response(R). The transaction verification usually happens between the master and salve interface which is initiated by the signal in the read address channel and read data channels. The architecture for the AXI read transaction with read address and read data between the master and slave interface is shown in the Fig-1(a).



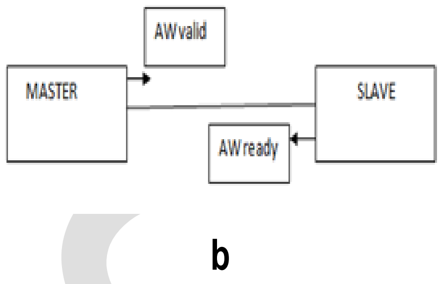
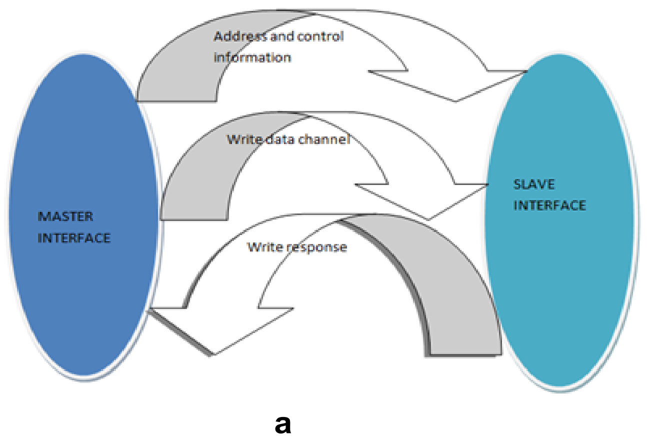


Fig-1 a) AXI read Transactions with address read and data read. b) AXI read Transactions with Address ready and Address Valid Signal.

Here each channel has dedicated valid and ready signals, based on their responses the data communication usually happens between the master and slave interface. The master slave communication with address valid (AW) and address ready (AR) is clearly illustrated in the Fig-1(b). During the read operation the read phase .master will give the read address request with address and control information based on which slave will respond accordingly.

WRITE TRANSACTION PHASE

In the write transaction phase initially the master gives the request for write address with its address and its control information. These two actions address request and control information passing will be happening one after one between the master and slave in write phase mode. Similar to the read phase the write phase will involve the three modes as writing the channel address in write address phase(AR), writing data to the channel in write data phase(W) and finally giving back the response for write operation in write response phase(B). The master slave communication with their write data channel, write response and control information along with their address is clearly illustrated in the Fig-2(a).



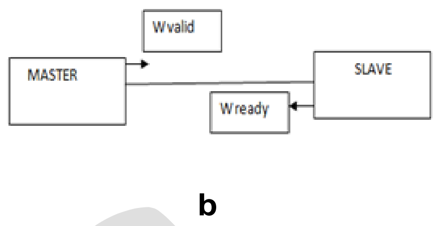


Fig-2 a) AXI write Transactions with address write and data write. b) AXI write Transactions with Address ready and Address Valid Signal.

In the whole operation of the write phase master initiates the request for write operation with write data request and with write data correspondingly the slave writes the response to the master. The entire master slave write data transaction with the data valid and data ready signal are shown in the Fig-2(b).

VERIFICATION IP ENVIRONMENT

The methodology to verify the system components in SOC using the intellectual verification IP concepts is more and more beneficial, as it saves the time for verifying the chip and reduces the time to manufacture without any faults. This type of verification environment allows us to reuse it for any type of component verification (i.e. we can tune to verify the functionality of any devices).Because of this possibilities it is easy to develop so many test cases to verify the DUV under verification. Nowadays the verification process is initiated by means of code coverage and functional coverage to verify the functionalities of the entire design under different test scenario.

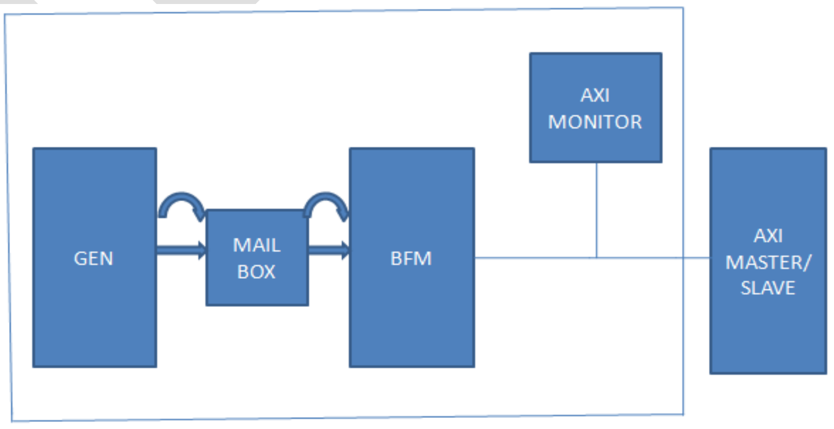


Fig-3 AXI Verification Environment.

The verification IP environment consists of a Generator, Bus functional Model (BFM), Mailbox, Monitor, AXI Interface and AXI Master/Slave. The entire verification environment for verifying the AXI Master/Slave is pictorially given in the Fig-3. The generator in the verification environment will be involved in generating different test cases according to the verification criteria. In the same manner the bus functional model will collect the transactions from the generator via mailbox. Here the mail box is an intermediate agent to convert the signal from one form to another form which is acceptable by BFM. The BFM collects all the transaction on drives into the AXI interface. The duty of the AXI interface is to connect the AXI monitor, BFM and AXI Master/Slave. Here the monitor keeps tracking the data transfers inside the test environment and gives an alert message for each transaction.

VERIFICATION PLAN

The verification plan tells the details about the properties going to be verified in the Design under verification (DUV) with respect to the corresponding test strategies. The properties which are being verified in the DUV are listed below as follows

- Verifying the system connectivity during read and write cycles
- Transaction routing.
- Data integrity

For the effective verification of these properties, the coverage driven verification methodology is followed. By using this type verification plan able to achieve 100% effectiveness in the verification process.

RESULTS AND DISCUSSION

In the verification process of the AXI Master/Slave bus protocol system verilog is used for modeling the AXI Master/Slave with their verification environments. The verification environment consists of generator, mailbox, BFM and AXI Interface all of these are modeled in system verilog and used for the verification process of this bus protocol. Mentor graphics Questa-Sim tool is used to verify the functionally of this design in the code coverage enabled mode to do the entire verification of this bus protocol. During the verification of this bus protocol first the read architecture and write architecture with all the channels are verified and then checked using verification IP environment in code coverage enable report mode.

VERIFICATION OF WRITE ARCHITECTURE

In this verification stage all three write signals named write address, write data and write response are verified for each transaction. The write address includes AWID, AWADDR, AWLEN, AWSIZE, AWVALID AND AWREADY signals toggles for every positive high edge of the global clock and finally writes the address in the channel. AWID is a write address ID which represents a particular tag for each write address; it should match with the write data WID. During the toggling action of the clock at positive edges with the high enable logic value in WVALID and WREADY, the write data channel acknowledgement will takes place. Similarly the write response will happen at the high state of BVALID and BREADY signals. Here the signal AWLEN is of four bit size [0:3] which generates different transactions from one to sixteen. During that generation process if AWLEN is 0100 then it will have 0101 transactions which mean it will increments the transactions by one. This is clearly illustrated in the waveform clearly at Fig-4. From the waveform it is observed that AWSIZE indicates the size of each transaction. The entire write architecture is simulated and verified for all the signal toggle counts which is clearly show in the waveform in Fig-4.

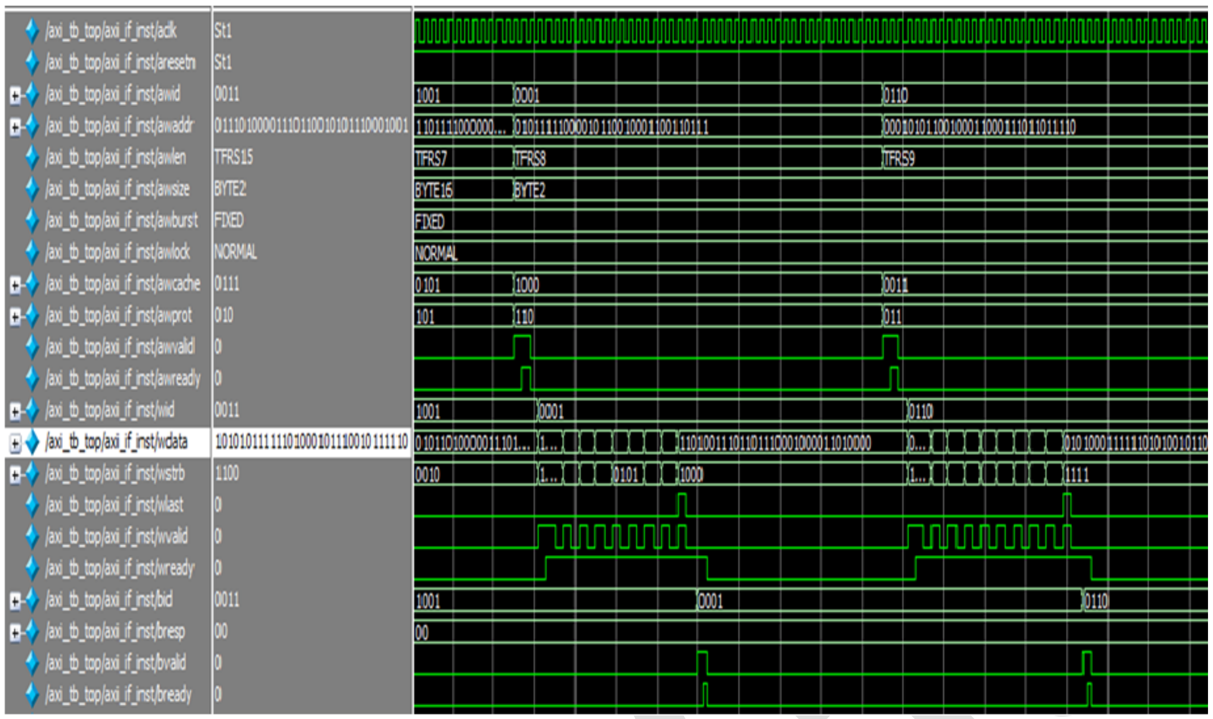

/axi_tb_top/axi_if_inst/aclk
/axi_tb_top/axi_if_inst/aresetn
/axi_tb_top/axi_if_inst/awid
/axi_tb_top/axi_if_inst/awaddr
/axi_tb_top/axi_if_inst/awlen
/axi_tb_top/axi_if_inst/awsize
/axi_tb_top/axi_if_inst/awburst
/axi_tb_top/axi_if_inst/awlock
/axi_tb_top/axi_if_inst/awcache
/axi_tb_top/axi_if_inst/awprot
/axi_tb_top/axi_if_inst/awvalid
/axi_tb_top/axi_if_inst/awready
/axi_tb_top/axi_if_inst/wid
/axi_tb_top/axi_if_inst/wdata
/axi_tb_top/axi_if_inst/wstrb
/axi_tb_top/axi_if_inst/wlast
/axi_tb_top/axi_if_inst/wvalid
/axi_tb_top/axi_if_inst/wready
/axi_tb_top/axi_if_inst/bid
/axi_tb_top/axi_if_inst/bresp
/axi_tb_top/axi_if_inst/bvalid
/axi_tb_top/axi_if_inst/bready

Fig-4 AXI Write Cycle Response.

The signals verified in this test case are AWADDR, AWVALID, AWREADY, and WDATA along with their write address, write data's signals. Also the signals WLAST, WVALID, WREADY, BRESP, BVALID and BREADY are also verified for every transactions. The parameters VALID COUNT, BUSY COUNT, BUS UTILIZATION are calculated practically with this test case and the bus utilization is shown in percentage numbers.

Write phase is divided into three channels response. The necessity to verify write phase is whether hand shaking of signals is happening perfectly or not in all the three channels.

VALID COUNT for read phase = 13.

BUSY COUNT for read phase = 16.

BUS UTILIZATION = (13/16)*100 = 81.25 percent.

VERIFICATION OF READ ARCHITECTURE

In this read cycle verififcation, all the read architecture signals ARVALID, ARREADY, RVALID, RREADY, RLAST, RDATA and ARSIZE are verfified for each transcations. The read architecture includes two channels i.e. read address and read response channels. The read address channel will intialize its address fetching at the high state of ARVALID and ARREADY signals for every positive edge of global clock. Similarly after a gap period of delay read response will instantiated to high mode for every positive edge of RVALID and RREADY signals. RLAST indicates the last transaction in the RDATA signal. Similarly ARSIZE and ARLEN are same as compared to that of write architecture. The entire waveform for the verififcation satge of the read architure is given Fig-5.

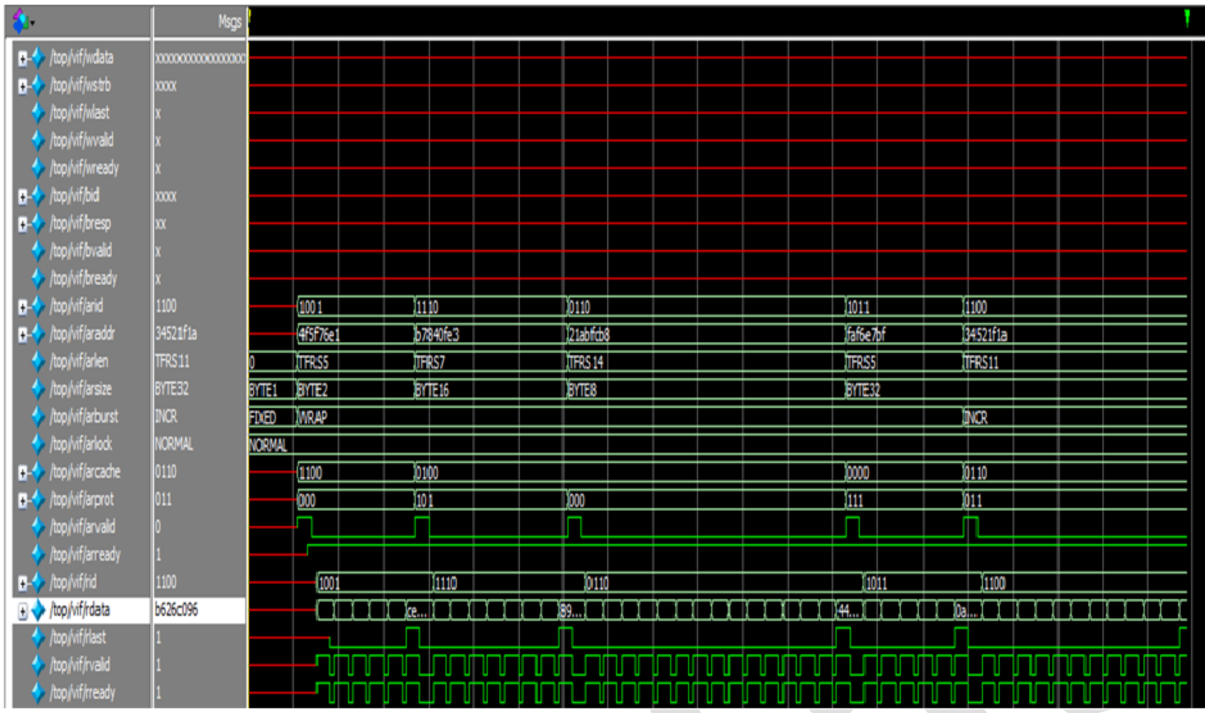


Fig-5 AXI Read Cycle Response

Verifying only the read phase is the main criteria of this test case. Neglecting the write phase signal values and focussing only on the read operation related signals and calculate the parameters are taken into consideration. The working of the read phase which includes two channels is same as explained above, the main focus is on verifying the parameters which leads to succesful measurment of bus utilization practically.The signals that are verified in this test case are ARADDR, ARVALID, ARREADY, ARID, RID, RDATA, RLAST, RRESP, RVALID, RREADY. Read phase is divided into two channels read address and read data plus read response. The necessity to verify the read phase is to cross check the hand shaking of signals for each channel then only a proper read phase will happen.

VALID COUNT for read phase = 10.

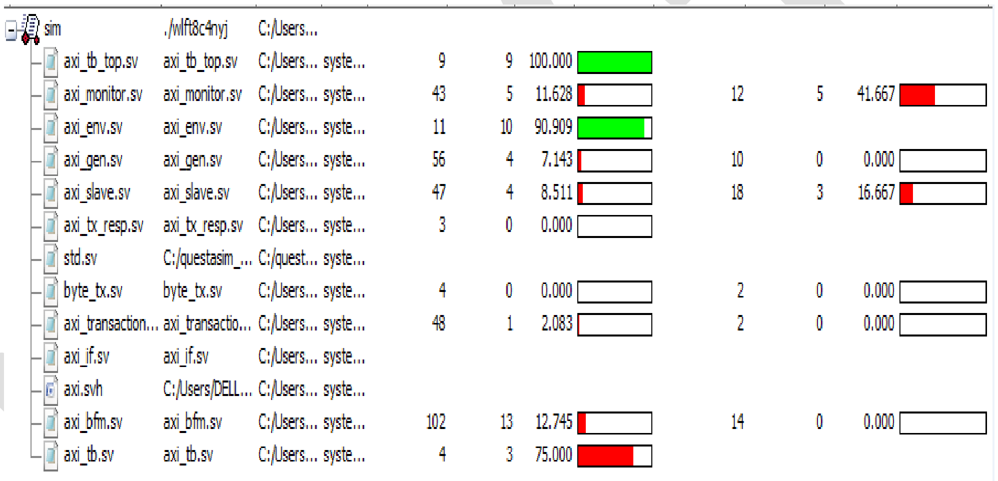
BUSY COUNT for read phase = 13.

BUS UTILIZATION = (10/13)*100 = 76.92 percent.

The same read and write phase is verified using the code coverage mode analysis and the coverage driven report is given in the Fig-5. The code coverage mode analysis is covering about the 80% of verification IP using the random test bench based verification methodology.

The sample coverage report for the AXI Slave/Master for the various analyses like state machine, branches, transitions and toggle counts etc., is given below as follows

```
                       ----------------    ------    ----    ------ ---------

Enabled Coverage                 Active          Hits        Misses   % Covered

                       ----------------    ------    ----    ------ ---------

                  Statements          62      32      30     51.6

                  Branches            30      16      14     53.3

               FEC Condition Terms         0       0       0    100.0

               FEC Expression Terms        0       0       0    100.0

                  States          0       0       0    100.0

                  Transitions          0       0       0    100.0

      Toggle Bins                         576      284      292      49.3
```



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			

Fig-6 Coverage Mode Analysis of Read & Write Response.

CONCLUSION

The AXI protocol verification, and the signals used in each channel are verified and analyzed using the code coverage mode analysis. The main advantage of this kind of verification is using the pseudo random coverage driven verification, where the time to market is less and applicable for complex designs using system verilog verification. In future we develop a test case to verify both the write and read phase simultaneously from the same location and different locations of read and write.

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 Core Sight TM, 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] 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

[7] Benini L, De Micheli G. Networks on chips: a new SoC paradigm. IEEE Computers 2002; 35:70–8.

[8] 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.

[9] Beigne E, Vivet P. *Design of on-chip and off-chip interfaces for a GALS NoC architecture*. In: 12th IEEE international symposium on asynchronous circuits and systems; 2006. p. 172–83.

[10] 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.

[11] 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.

[12] 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