IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 7, February 2016)

A Novel Arbitration Technique of AMBA AHB

Pallavi Kumari Gautam

Naveen Upadhyay

Abstract – **Resolution is a big issue in SOC (System on Chip) while dealing with number of master trying to sense a single data bus. The effectiveness of a system to resolve this priority resides in its ability to logical assignment of the chance to transmit data width of the data, response to the interrupts etc. The purpose of this paper is to propose the scheme to implement such a system using the specification of AMBA bust protocol. The scheme involves the typical AMBA features of 'single clock edge transition', 'Split transaction', 'several bus masters' and 'burst transfer'. The bus arbiter ensures that only one bus master at a time is allowed to initiate data transfers. Even though the arbitration protocol is fixed, any arbitration algorithm, such as highest priority or fair access can be implemented depending on the application requirements. The design architecture is written using VHDL (Very High Speed Integrated Circuits Hardware Description Language) code using Xilinx ISE Tools. The architecture is modeled and synthesized using RTL (Register Transfer Level) abstraction and Implemented on Virtex-2 series.**

Keywords – **AMBA, SOC, RTL, VHDL.**

I. INTRODUCTION

Systems-on-Chip (SoC) and in particular embedded real-time systems typically consist of several computational elements. These elements fulfil different tasks for processing an overall solution. Let's take a set-top box for TV sets as an example [1]. A set-topmost generate a TV-signal for a particular TV channel from a digital satellite signal. This process takes different tasks. One task is to split the incoming digital signal into data streams, such as video and audio. Another task is to convert the video stream into an actual TV-signal. One more conversion has to be made to turn the audio stream into an audio signal for the TV set. Meanwhile, another task handles the user input such as changing the channel when the remote control is pressed. All these tasks have to be done in parallel and are bound by real-time deadlines. The cost of missing these deadlines is visible as black boxes on the screen or audible as noise. This is unacceptable and therefore it is necessary to always deliver this data within hard real-time deadlines. These computational elements are either general-purpose processors or digital signal processors. Nowadays, multiple of them are integrated into a System-on-Chip solution. A processor needs to interact with other processors, memories or I/O devices to complete a task. Currently busses are used to interconnect these IP blocks. The current research in the field suggests using Networks-on-Chip (NoC) to interconnect IP blocks, because NoCs allow more flexibility than busses [1].

With the need of application, chip with a single processor can't meet the need of more and more complex computational task. We are able to integrate multiple processors on a chip thanks to the development of integrated circuit manufacturing technology Now as there are multiprocessing units and processors is getting faster, so compatibility with slow communication architectures a bit difficult furthermore this slow and conventional communication architecture limits the throughput.

To improve the performance we have to develop such efficient on chip Architecture which will be much faster system on chip solution which removes the limitation of communication architecture one of the solution is "AHB bus" but it can't give perfect parallelism as it can allow only one master to communicate at one slave only. While in our design there are five independent transfer channels which make multiple masters access multiple slaves at the same time and gain a perfect parallelism performance in MPSOC design. The bus arbiter ensures that only one bus master at a time is allowed to initiate data transfers. Even though the arbitration protocol is fixed, any arbitration algorithm, such as highest priority or fair access can be implemented depending on the application requirements.

AHB is a new generation of AMBA bus which is intended to address the requirements of high-performance synthesizable designs. It is a high performance system bus that supports multiple bus masters and provides high bandwidth operation.

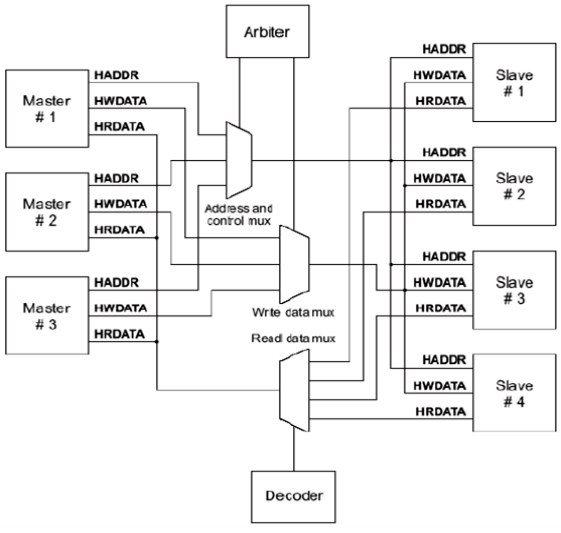

Figure 1: AMBA AHB Block diagram

IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 7, February 2016)

AMBA AHB implements the features required for high performance, high clock frequency systems including:

- Burst transfers
- Split transactions
- Single-cycle bus master handover
- Single-clock edge operation
- Non-tristate implementation
- Wider data bus configurations (64/128 bits)

In this paper, an industry-standard ARBITER is implemented for AHB (AMBA). The specification describes the bus attributes, the protocol definition and types of transactions, bus management, and the programming interface required to design.

II. PROPOSED METHODOLOGY

Arbitration to choose the next bus master uses a round robin arbitration algorithm. This ensures that no master gets starved. When a master has locked the bus, the round robin arbitration is overridden and the master with the lock retains highest priority to the bus.

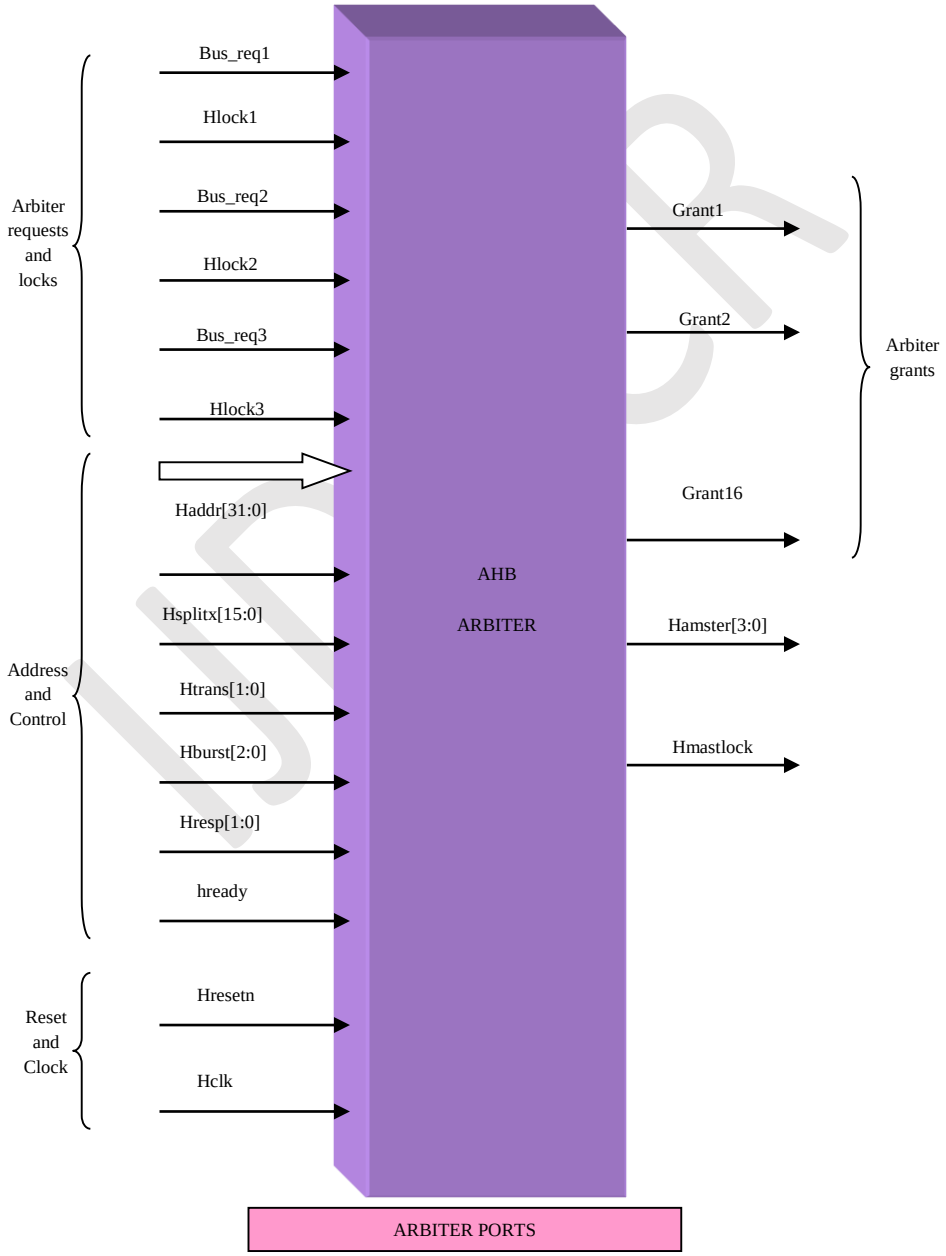

Figure 2: Pin Diagram for AHB Arbiter

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 7, February 2016)

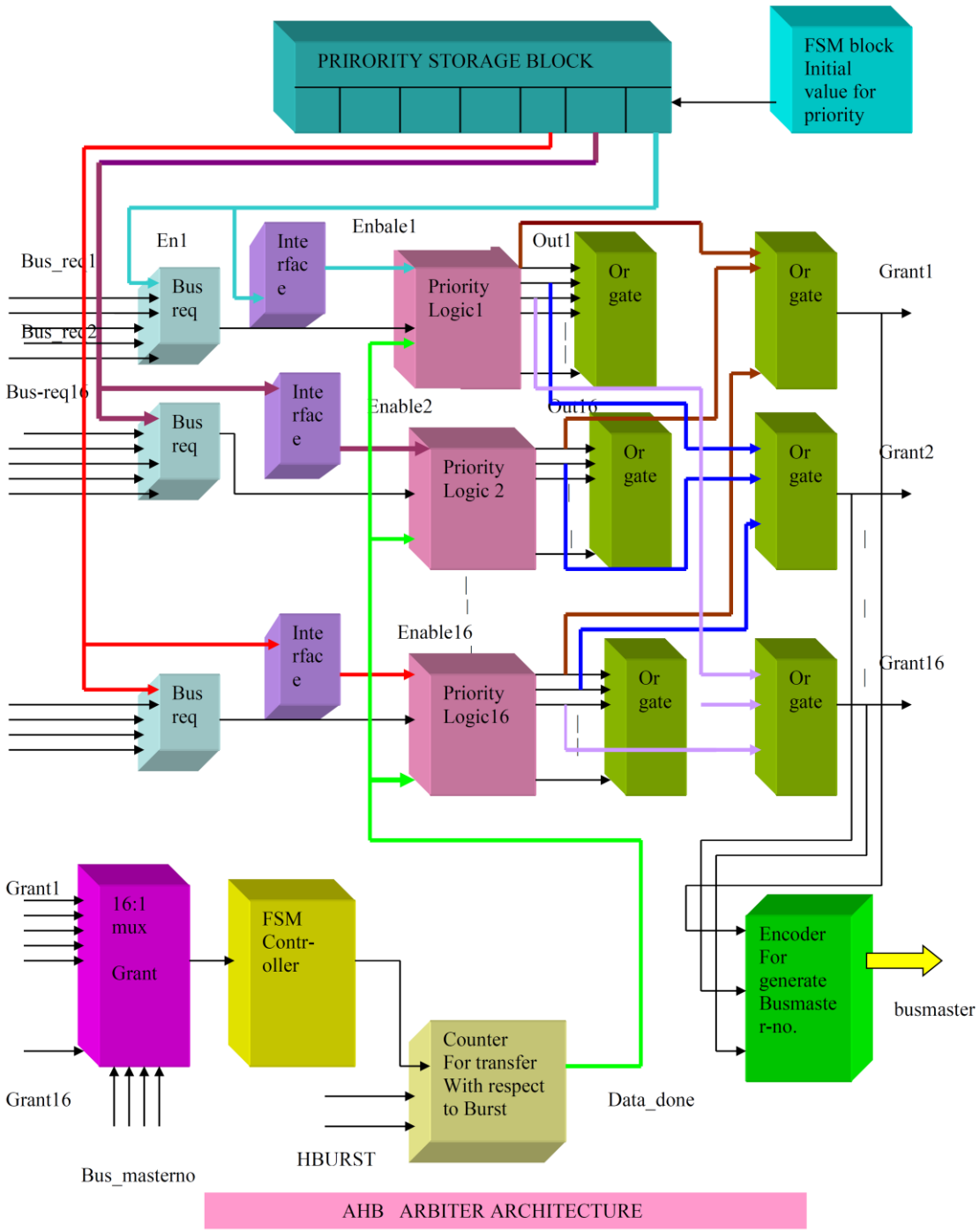

Figure 3: Block diagram of Arbiter

The sixteen AMBA Bus masters are Master 0 through Master 15. Slave 0 through Slave 15 are the sixteen AMBA Bus slaves. The AMBA AHB Bus Arbiter/Decoder contains a default master-Master0, and a default slave- Slave 0.

AMBA AHB Bus Arbiter features are summarized:

- AMBA AHB Bus arbiter function
- Round robin arbitration
- Default master- Master 0
- Default slave- Slave 0

The Arbiter block monitors the AMBA Bus for requests and chooses the master with highest priority request as the next AMBA bus transaction master. If there are no

IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 7, February 2016)

requests, the Default Master is chosen as the master to drive the next AMBA Bus transaction.

Testing the IP

The method of testing the IP was kept simple since it was more important to concentrate on the functionality of the IP.

The Subsystem Specifications

The AHB ARBITER IP can be broken into two subsystems. The two major components of the system under design are the controller and data path.

Controller

The controller is essentially a Mealy state machine. It keeps track of the different sections of an ARBITER transaction. The first state of the controller is the start state. The next state to check the grant if grant is there then it will make the necessary signal high which will further control the counter, and counter interface block.

Data path

The Data path are further divided into several sub system blocks, few of them are controlled by controller. Followings are different data path with different functionality.

1. Priority logical block
2. Priority storage block
3. Mux arrangement for grant and bus request.
4. Or gate
5. Counter
6. D-flip_flop.

For details please refer Figure 3.

Overall Description of Design

The Followings are the major steps involves in the Arbiter designs form architectural or functional point of view:

- The bus_request of different masters has to pass through the bus_req block. Which is responsible to pass the request to other logical blocks. This block is depends upon the enable pin which is coming from priority storage block.
- Bus_request further pass through interface block and goes to priority logical block. The interface block is giving the enable signal to priority logical block and interface block is responsible for monitor the data transaction through data_done signal, it can assert and deassert the enable pin depends upon the data_done.
- This bus_req goes to the priority logic block. this block further decide that which master request will get the highest priority depending upon the priority this block is generate the Grant signal .
- This Grant signal goes to priority storage block, encoder block and as out_put port to interact with Master. After getting the Grant signal Master will send Address, Burst to indicate the type of transfer, and Slave will also send Hready, Hresp, Hsplit.
- At the same time when master samples the signals to the Arbiter Grant signals which are the output of the Arbiter pass through the mux,inside the Arbiter ,for this mux bus_master no is select line ,which indicates that which master is accessing the bus.
- The out_put mux then passes to the controller block which will generate the necessary signals for counter, i.e. the controller will control the operation of counter.
- The grant output from the priority logic block is Or and then sent to the priority storage block which will store the priority and pass the enable signal to the next priority depending upon the grant value. And the whole operation is repeated depends upon the transaction mode.

III. SIMULATION RESULTS

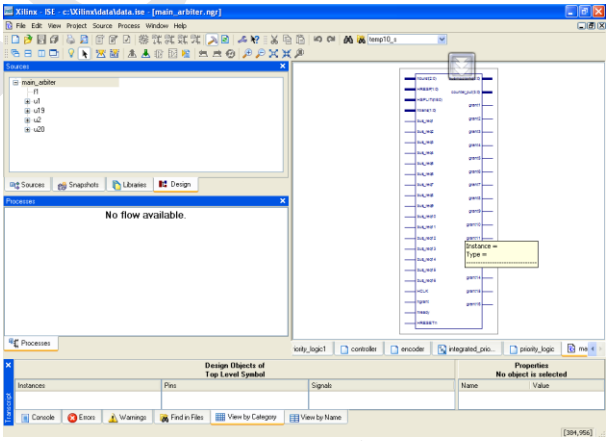

Figure 4: Pin diagram of top arbiter

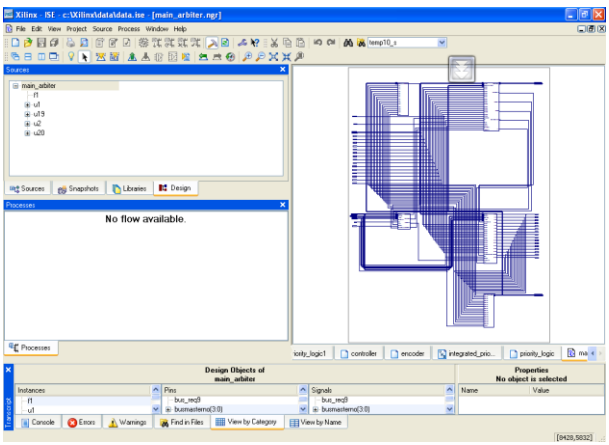

Figure 5: RTL Schematic of arbiter

IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 7, February 2016)

Device utilization summary for arbiter:

Selected Device	:	2v80cs144-6
Number of Slices	:	1566
Number of Slice Flip Flops	:	533
Number of 4 input LUTs	:	2752
Number of bonded IOBs	:	64
IOB Flip Flops	:	4
Number of GCLKs	:	2

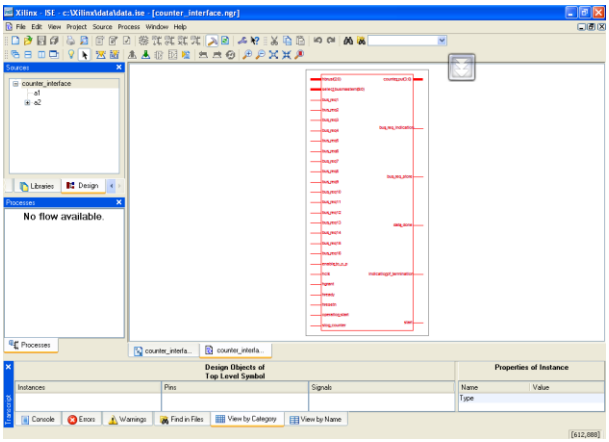

Figure 6: Pin diagram of Counter Interface

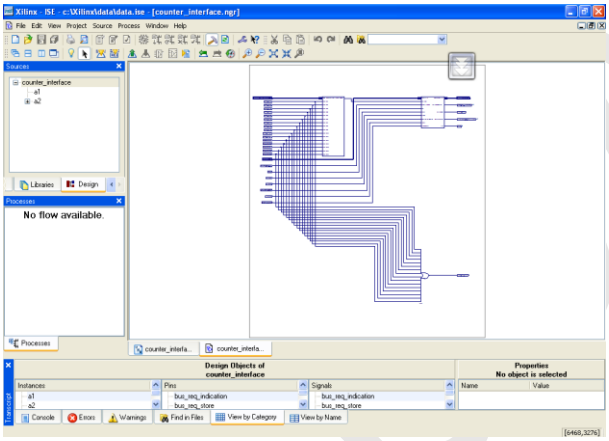

Figure 7: RTL Schematic of Counter interface

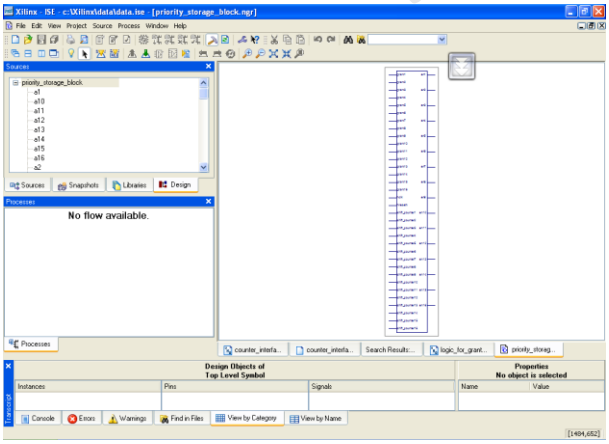

Figure 8: Pin diagram of priority storage

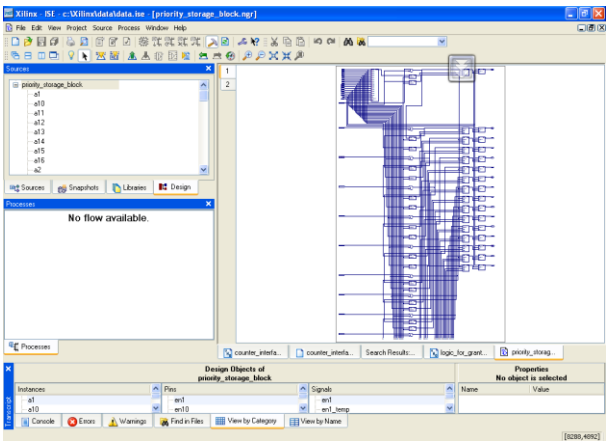

Figure 9: RTL Schematic of priority storage

IV. CONCLUSION

The design is simulated on Modelsim 6.5 & verified through effective test bench. The RTL is implemented on Vertex-2 (XC2V80, CS144 package). The advantage of this design is that we have taken care of latch formation, as it is a FPGA implementation hence with less latch & maximum flip-flop have enhanced our area efficiency. Cyclic FSM (grey encoding) has been done for controller design &we controlled power Consumption the design can further be optimized for ASIC design

REFERENCE

[1] Kees Goossens, Om Prakash Gangwal, Jens Rover, and A. P. Niranjan, "Interconnectand Memory Organization in SOCs for advanced Set-Top Boxes and TV-Evolution, Analysis, and Trends", In JariNurmi, HannuTenhunen,JouniIsoaho, and Axel Jantsch, editors, Interconnect-Centric Designfor Advanced SoC and NoC, chapter 15, pages 399–423. Kluwer, April 2004.

[2] Poletti, Francesco, Davide Bertozzi, Luca Benini, and Alessandro Bogliolo. "Performance analysis of arbitration policies for SoC communication architectures." Design Automation for Embedded Systems 8, no. 2-3, pp. 189-210, 2003.

[3] Huang, Yu-Jung, Yu-Hung Chen, Chien-Kai Yang, and Shih-Jhe Lin. "Design and implementation of a reconfigurable arbiter", In 7th WSEAS International conference on signal, speech and image processing (SSIP'07), Beijing, China. 2007.

[4] Shanthi, D., and R. Amutha, "Design Approach to Implementation Of Arbitration Algorithm In Shared Bus Architectures (MPSoC)", Computer Engineering and Intelligent Systems 2, no. 4, pp. 185-196, 2011.

[5] E. Raja, K.V. Ramana, "Implementation of Multilayer AHB Bus matrix for ARM", International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-1, Issue-5, November 2011.

[6] Dr. Fazal Noorbasha, B. Srinivas, Venkata Aravind Bezawada, V. Sai Praveen, "Implementation of an Adaptive-Dynamic Arbitration Scheme for the Multilayer AHB Busmatrix", International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622, Vol. 2, Issue 4, pp.825-831, July-August 2012.

[7] Jalle, Javier, Jaume Abella, Eduardo Quinones, Luca Fossati, Marco Zulianello, and Francisco J. Cazorla. "AHRB: A high-performance time-composable AMBA AHB bus." IEEE 20th Symposium in Real-Time and

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 7, February 2016)

Embedded Technology and Applications (RTAS), , pp. 225-236, 2014.

[8] Shraddha Divekar, Archana Tiwari, "Multichannel AMBA ARB with Multiple Arbitration Technique", IEEE, International Conference on Communication and Signal Processing, April 3-5, pp. 1854-1858, 2014.

[9] Pravin S. Shete, Dr. Shruti Oza, "Design of an AMBA AHB Reconfigurable Arbiter for On-chip Bus Architecture", International Journal of Application or Innovation in Engineering & Management (IJAIEM), ISSN 2319 – 4847, Vol. 3, Issue 5, May 2014.

[10] Shashidhar R., Sujay S. N., Pavan G. S., "Implementation of Bus Arbiter Using Round Robin Scheme", International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753, Vol. 3, Issue 7, July 2014.

[11] Mrs. R. Ramya, Ms. Preethi Vadivel, "Slave-Side Arbitration and Implementation For Multilayer - AHB Bus Matrix", International Journal of Engineering Research and Reviews, ISSN 2348-697X, Vol. 2, Issue 3, pp: 56-60, September 2014.

[12] "AMBA AXI Protocol specification". www.arm.com/armtech/AXI