Design and Implementation of High Performance AHB Reconfigurable Arbiter for Onchip Bus Architecture

Ashutosh Kumar Singh
Digitech Solution ,
Operation Manager,
Indore,India,
akzhse@gmail.com

Anurag Shrivastava
Dept of Electronics & Commn,
Priyatam institute of Technology &
Management, Indore, India
shrivastavaanurag@rediffmail.com

G.S. Tomar
Machine Intelligence Research
(MIR) Labs, Gwalior 474011
Inida
gstomar@ieee.org

***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 reconfigurable architecture so that it can be interface with any common IP core of such a system using the specification of AMBA bus protocol .The scheme involves the typical AMBA features of 'single clock edge transition ', Split transaction ','several bus masters ', 'burst transfer '.The bus arbiter ensures that only one bus master at a time is allowed to initiate data transfers. Here we have proposed and implemented the reconfigurable arbitration algorithm, such as highest priority or fair access and round robin 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 Virtex2 series.**

Keywords- *reconfigurable* Arbiter, Round Robin, Split Transfer, AMBA, IP,VHD***)***

I. INTRODUCTION

All We have developed an Arbitration algorithm so that it can be interface with any type of arbitration to choose the next bus master. 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. The sixteen AMBA BUS master0 through Master Slave 0 through slave 15 are the sixteen AMBA Bus slaves. The AMBA AHB Bus Arbiter/Decoder [3],[12] contains a default master-master (), and a default slave –slave().

AMBA AHB Bus Arbiter features are summarized

1. AMBA AHB Bus arbiter function
2. Round robin arbitration
3. Default master-master ()
4. Default slave- slave ()

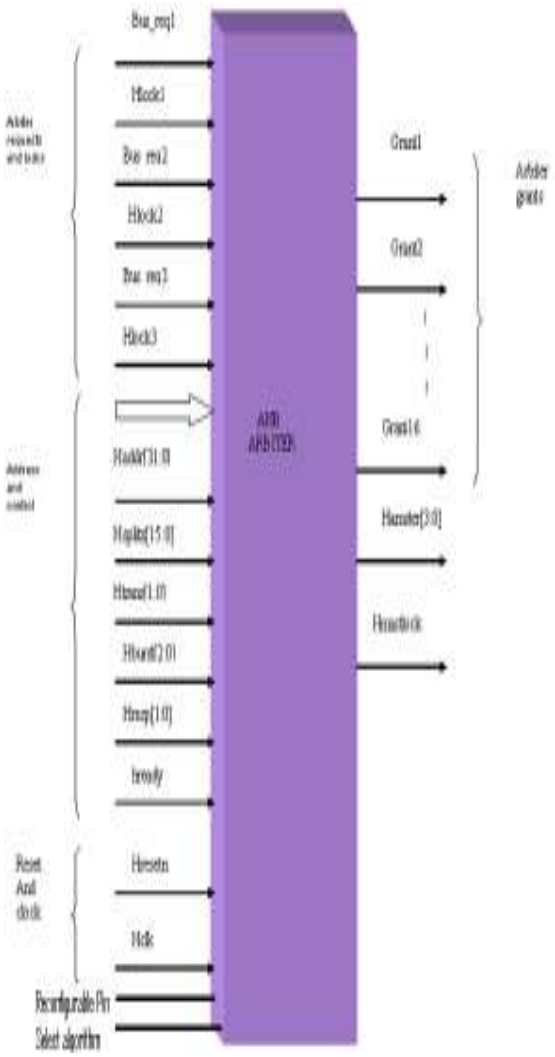

Figure 1. AHB Reconfigurable Arbiter

978-0-7695-4437-3/11 $26.00 © 2011 IEEE
DOI 10.1109/CSNT.2011.99

The arbiter Block monitors the AMBA Bus for requ8est and chooses the master with highest priority request as the next AMBA bus transaction master. If there are no request, the Default Master is chosen as the master to drive the next AMBA Bus transaction. The arbiter block monitors the AMBA Bus for request and chooses the master with highest priority request as the next AMBA bus transaction master to drive the next AMBA Bus transaction.

II. THE BUS SPECIFICATION

AHB (AMBA High-performance Bus) is a bus protocol introduced in AMBA specification version 2 published by ARM limited Company.[1] In addition to previous release, it has the following features:

- Single edge clock protocol
- Split transaction
- Several BUS Master
- Burst transfers
- Pipelined operations
- Single cycle bus master handover
- Non-tristate implementation
- Large bus-widths(64/128 bit)

A simple transaction on the AHB Consist of an address phase and a subsequent data phase (without wait states: only two bus –cycles), Access to the target device is controlled through a MUX (non-tristate), thereby admitting bus – access to one bus master access at a time The AMBA on-chip bus is an established, open specification that serves as a framework for System-on-chip (SoC) designs.[3],[12] AMBA 2 comprises two system buses: the Advanced High performance bus (AHB) and the Advance peripheral Bus (APB). As increasing numbers of companies adopt AMBA, it is rapidly emerging as a de-facto Standards for SoC construction and intellectual property (IP) library development, AMBA provides the "digital glue" that binds IP core together and is a Key enabler of IP reuse [7]. The AHB ARBITER IP[5],[9]can be broken into subsystem. The two major components of the system under design are the controller and data path.

III. DATA PATH

The Data Path Are further divided into several subsystem blocks, few of them are controlled by controller. Following are different data path with different functionality.

1. Priority logic block: The priority storage block is implemented through FSM approach[11]. The priority scheme follows the round robin theorem of priority. The bus request have highest priority will get Grant First and Rest of request will wait for there priority.
2. Priority Storage block: The priority storage block is responsible to store the priority level. IT is the block which is responsible to enable the Bus req. Block and Interface Block. Grants are the input pin of this block
3. Mux arrangement for grant and bus request: 16:1 MUX for both grant and request
4. OR gate: This block contains two sets of OR gate, each set contains 16 OR gate. First OR gate set is used to OR the output of each individual priority logic block ,second one is used to OR the individual grant form each priority logic.
5. Counter
6. D-flip-flop

IV. DESIGN DESCRIPTION

The following are the major steps involves in the Arbiter Designs form architectural or functional point of view. The backbone architecture is shown in figure 2 and is self explanatory as per the clear depiction of the blocks.

1. The bus request of different master has to pass through the bus_req. block. Which is responsible to p[ass the request to other logical blocks this block is depends upon the enable3 pin which is coming from priority storage block.
2. Bus req. further pass through interface block and goes to priority logical block [11]. The interface block is giving the enable signal ; to priority logical block and interface bock is responsible for monitoring the data transaction through data_done signal; it can assert and de-assert the enable pin depends upon the data_done.
3. This bus_ req. goes to the priority logic block, this block further decides that which master request will get the highest priority depending upon the priority this block is generate the grant signal.
4. 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 and Hsplit.
5. At the same time when master samples the signal to the Arbiter Grant signals[12] 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
6. 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.
7. Here some modification has been done in the previous architecture that athe selsection of arbitration is based on the requirement of the IP interface.
8. The grant output from the priority 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 depending upon the transaction.

V. RESULTS AND SIMULATION

Here the graph is plotted[12] which shows the average number of CPU bus requests (x-axis) against

REQUEST_RATE (y-axis) values of 50, 40, 30, 20, 15, 10, 5 for three different cases (a request counter must be kept for each CPU that is incremented each time the CPU makes a request. The average number of requests is total # of requests divided by the number of CPUs):

1. ROUND_ROBIN = TRUE, OVERLAP_GRANT = TRUE
2. ROUND_ROBIN = FALSE, OVERLAP_GRANT = TRUE
3. ROUND_ROBIN = TRUE, OVERLAP_GRANT = FALSE
4. FAIR ACCESS = TRUE, OVERLAP_GRANT = FALSE
5. FAIR ACCESS = FALSE, OVERLAP_GRANT = TRUE
6. FIXED = TRUE, OVERLAP_GRANT = FALSE
7. FIXED= = FALSE, OVERLAP_GRANT = TRUE

All three cases should be plotted on the same graph for comparison purposes.

VI. DEVICE UTILIZATION SUMMARY

Selected Device: 2v500fg256-5

Number of Slices	1566
Number of Slice Flip Flops	535
Number of 4 input LUTs	2750
Number of bonded IOBs	64

Table.1

Total equivalent gate count for design: 20,095

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		348

Vccint 1.50V:	10	15
Vccaux 3.30V:	100	330
Vcco33 3.30V:	1	3

Fig.4. Waiting time for first come first serve arbitration algorithm

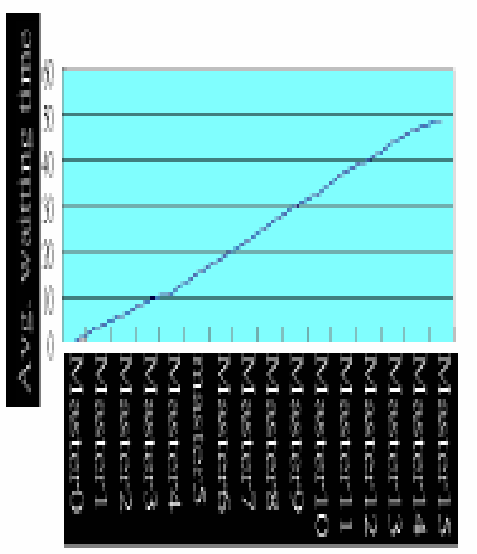

Fig.4. Waiting time for first come first serve arbitration algorithm

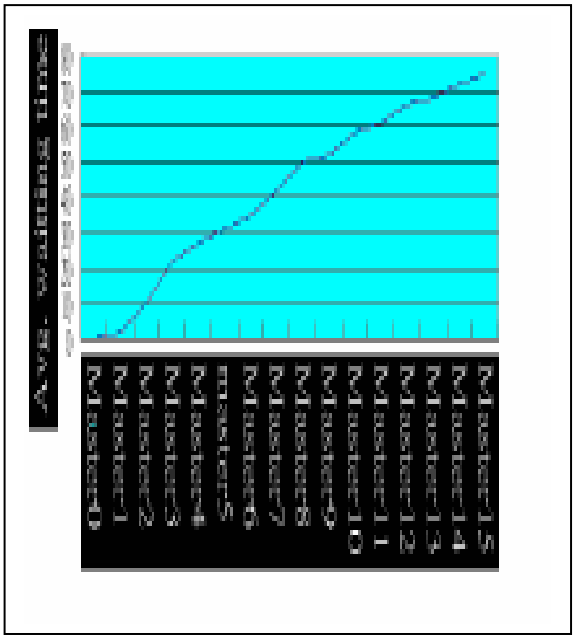

Figure 5.Average waiting time

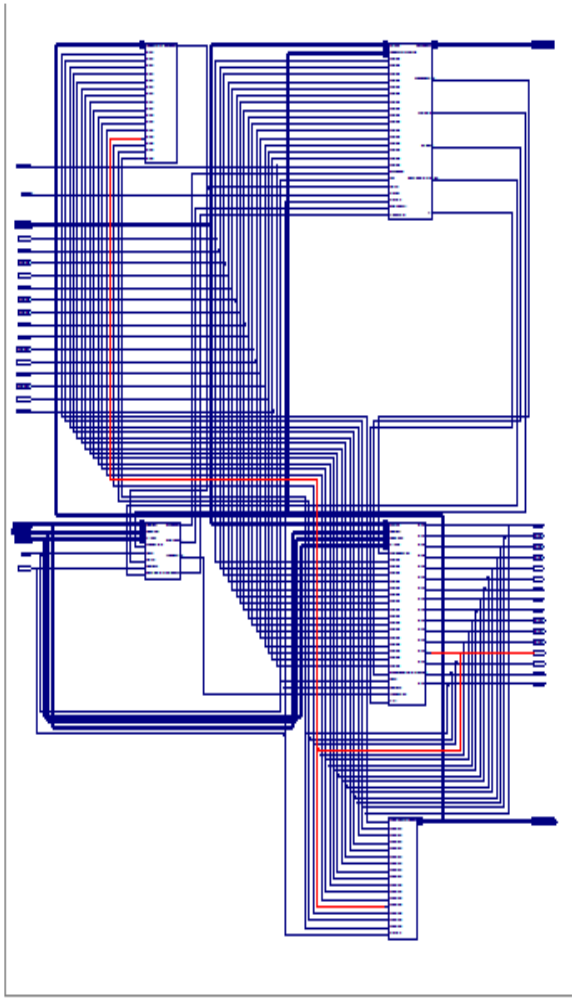

Figure 6. RTL view

VII. RESULTS AND SIMULATION

The design is simulated on modelsim & verified through effective test bench. The RTL is implemented on vertex2 (XC2V80, CS144 package) The advantage of this design is that we have taken care of latch formation, as it is a FPGA implementation[8],[5] 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.

REFERENCES

[1] AMBA Specification Rev 2.0. ARM Ltd., 1999.

[2] Ruibing Lu, Aiqun Cao, and Cheng-Kok Koh, Senior Member, IEEE," SAMBA-Bus:A High Performance Bus Architecture for System-on-Chips" IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 15, No. 1, January 2007.

[3] Xilinx "OPB Synchronous DRAM (SDRAM) Controller" DS426 July 21, 2005.

[4]H.-M. Lin, C.-C. Yen, C.-H. Shih, , and J.-Y. Jou, "On compliance test of on-chip bus for SOC," in *Proceedings of the 2004 conference on Asia South Pacific design automation*, 2004.

[4] Xilinx "OPB Double Data Rate (DDR) Synchronous DRAM (SDRAM) Controller" DS424(V1.9.1) September 19, 2003.

[5] Xilinx "Pico blaze Initial Design for Spartan-3E Starter Kit (LCD Display Control)" Ken Chapman Xilinx Ltd.16th February 2006.

[6] Yu-Jung Huang, Chih-Feng Liu, Shao-Pin Chang, Feng-Yuan Chuang, Chang-Chan Chen Department of Electronic Engineering, I-ShoUniversity, Kaohsiung, Taiwan 80424, ROC,

[7] "Design of LCD driver IP for SoC Applications" 2004 IEEE Asia-Pacific Conference onAdvanced System Integrated Circuits(AP-ASIC2004)/Aug. 4-5, 2004

[8] FPGA PROTOTYPING BY VHDL EXAMPLES, Xilinx Spartan™-3 Version, Pong P. Chu, Cleveland State University, A John Wiley & Sons, Inc., Publication 2008.

[9] Digital Logic and Microprocessor Design with VHDL, Enoch O. Hwang, La Sierra University Riverside, Thomson Publication.

[10] VHDL Programming by Examples, Douglas L. Perry, Tata McGraw-Hill Publishing Company Limited.

[11]Lahiri, K.; Raghunathan, A.; Lakshminarayana, G.; " The LOTTERYBUS on-chip communication architecture", IEEE Trans. On Very Large Scale Integration (VLSI) Systems, Vol. 14, No. 6, 2006, pp. 596 – 608.

[12] E. S. Shin, V. J. Mooney III, G. F. Riley, "Round-robin Arbiter Designand Generation," Georgia Institute of Technology, Atlanta, GA,Technical Report GIT-CC-02-38, 2002.

[13]Synopsys, Design Compiler, Available HTTP: http://www.synopsys.com/products/logic/design_comp_cs.html

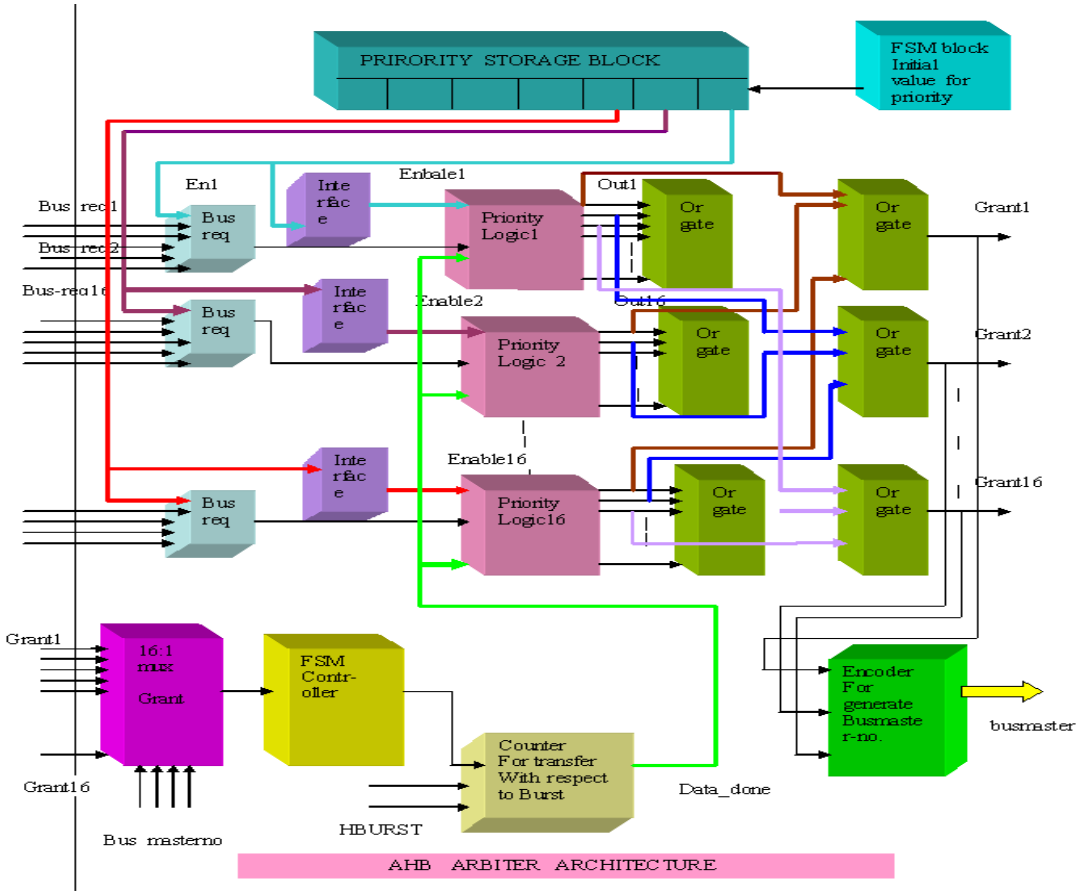

Fig3.1 Backbone OF Abiter

Figure 2. Backbone of Arbiter

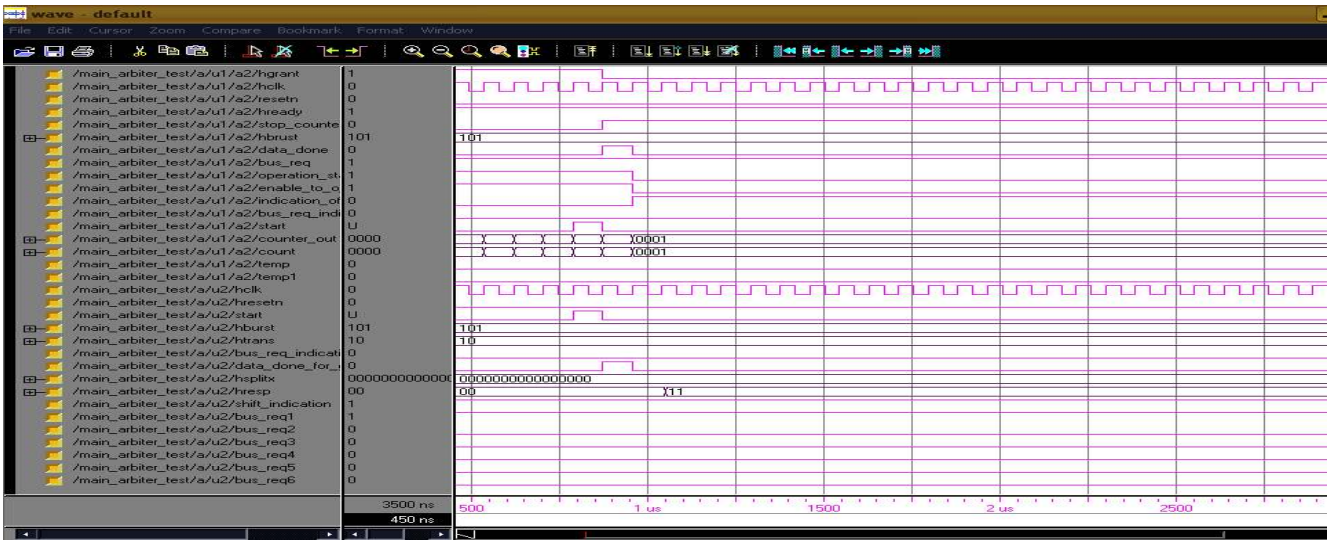

Figure 3. Simulation waveform