Design of AHB Reconfigurable Master Arbiter

Jasmin N. Surani[1]
[1]M.E. Student
[1]VLSI & Embedded System Design Department
[1]Gujarat Technological University, Ahmedabad, Gujarat, India

Abstract— Select the right master, 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. The purpose of this paper is to propose the scheme to implement reconfigurable architecture. This scheme involves typical Advanced Microcontroller Bus Architecture (AMBA) features like "Single clock edge operation", "non-tristate implementation", "burst transfer", etc. In this paper, Advanced Peripheral Bus (APB) is used to reconfigure the arbiter. In this arbiter there are some register which are reconfigured with the help of APB bus. Also there are three arbitration scheme are used for arbitration. Round robin, First Come First serve (FCFS), highest priority schemes are used. The design architecture is written using the verilog hardware description language using the Xilinx ISE tools.

Keywords: Reconfigurable arbiter, APB slave, three arbitration schemes.

I. INTRODUCTION

In the AMBA, there are three sub-buses Advanced High performance bus (AHB), Advanced Peripheral Bus (APB) and Advanced System Bus (ASB). Now a day, there are also other buses invented like Advanced Trace Bus (ATB), Advanced eX-tensible Interface (AXI), etc. For AHB, there is proposed arbiter and conversation logic for efficient arbitration. Arbitration scheme is developed which can be interface with any typical type of AMBA buses to give the grant to the masters. There is proposed plan of simple two master and one slave and also proposed arbiter which has configurable registers bank. These are configurable with the help of APB bus. APB bus has strobing facility which helps us to configure the registers. The arbiter Block monitors the AMBA Bus for request and chooses the master with highest priority request as the next AHB bus transaction master. If there are no requests, the no master will get grant for next AHB Bus transaction. Pin diagram of the proposed arbiter is shown nearby in Fig. 1.

II. SPECIFICATION

AHB (AMBA High-performance Bus)-lite is a bus protocol introduced in AMBA specification version 3 published by ARM limited Company [4]. In addition to previous release, it has the following features [4]:

(1) Single edge clock protocol
(2) Several BUS Master
(3) Burst transfers
(4) Pipelined operations
(5) Single cycle bus master handover
(6) Non-tristate implementation
(7) Large bus-widths(64/128 bit)

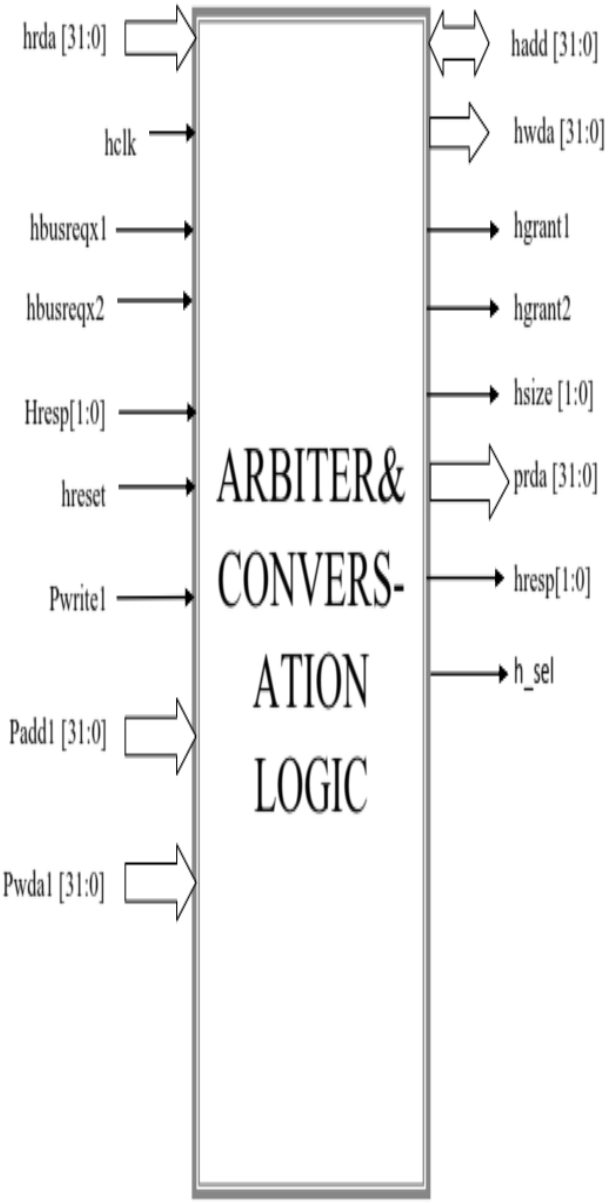

Fig.1: Arbiter and conversation logic

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. AMBA 3 comprises two system buses: the Advanced High performance bus (AHB) and the Advance peripheral Bus (APB)[4]. As increasing numbers of companies adopt AMBA. The two major components of the system under design are the controller to decide that value of registers are greater or lesser and data flow after that how data flows through the system.

III. DESIGN OF ARBITER

This arbiter is divided into two parts mainly, one is register bank and other one is main arbitration. All the design of arbiter implemented using FSM approach. In this arbiter, there four register which are used to give priority to the master. The four register are as follow:

(1) MAX_DATA_TRANS
(2) MAX_CLK_TRANS
(3) MAX_CONST_GRANT
(4) ARB_SELC

MAX_DATA_TRANS is of 11 bit. Masters can transfer data maximum of 2 KB only. If the limit of counter of particular master excides than automatically the will not get grant after that. MAX_CLK_TRANS is of 7 bit. Masters can get grant one time for maximum 128 clock cycles after its grant will get cancel. MAX_CONST_GRANT is of 2 bit. Master can consecutively get maximum 4 times grant, after that it will not get grant on next request. ARB_SELC is of 2 bit. There are three arbitration scheme is used for arbiter. Arbitration scheme can be chosen by proper selection of these bits[3]. These registers' values are configurable by using APB slave. We can write into these registers using write operation of APB slave also can read values of those registers.

All the arbitration schemes are implemented using the FSM approach. FSM designs of Round robin, First Come First Serve (FCFS) and Highest Priority are shown below:

A. *Round robin FSM:*

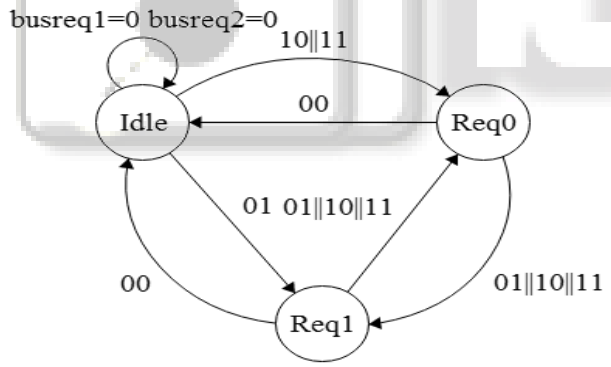

Fig.2: FSM approach of Round robin arbitration

B. *Highest priority fsm:*

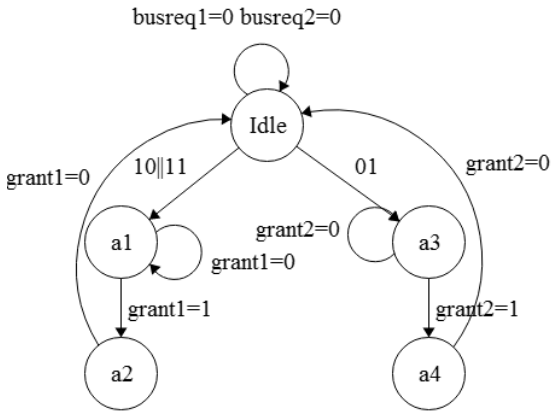

Fig.3: FSM approach of highest priority arbitration

C. *First come first serve fsm:*

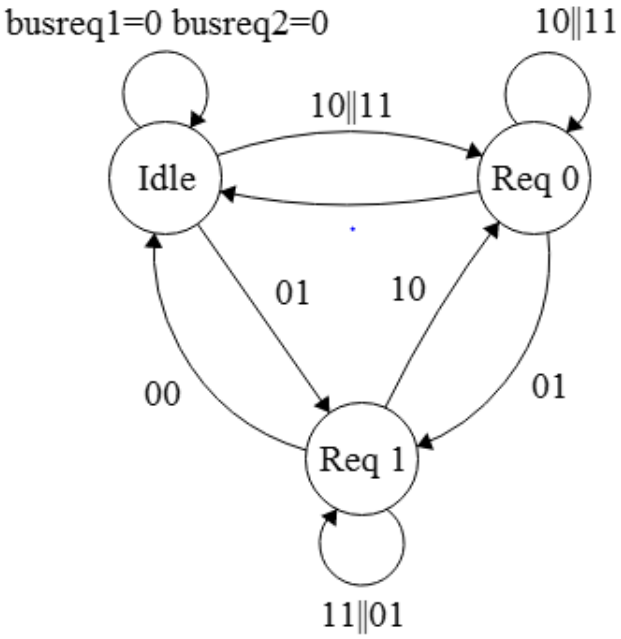

Fig.4: FSM approach of FCFS arbitration

Now there is proposed main FSM approach which includes these all FSM and also the checking masters by the help of register bank.

D. *Main FSM of Arbiter:*

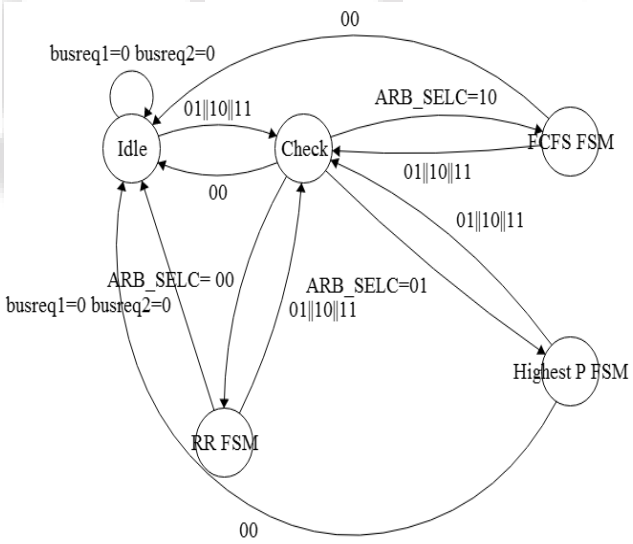

Fig.5: FSM approach of Arbiter

In this way the design of arbiter is made using all these FSM approaches.

IV. DESIGN DESCRIPTION

Design will be described in the three parts as per the architectural and function point of view:

(1) Whole arbiter is design using arbiter FSM approach shown in Fig. 5. In which arbiter will be always in the idle state when there is no busreq from any master. Now at the time when any master inserts busreq arbiter goes into the check state.
(2) Now arbiter will in the check state. In this state using register bank arbiter will check the respective status of particular master which had insert the busreq. If status

of that particular master excides the value of any register than at that time arbiter will not give grant to particular master. Now if all goes fine and master successively clear the comparison then it goes into the 3rd stage which of arbitration scheme stage which will be selected as per register ARB_SELC [3] as shown in fig. 5.

(3) Now this is the last and final stage of arbiter where particular master gets the grant from the arbiter. In this stage particular arbitration algorithm will decide to whom it has give grant when there are more than two busreqs. After these arbiter goes into appropriate stage as per the FSM approach shown above in Fig. 5.

(4) We can make round robin and highest priority with the only help of these FSM approaches but to make the FCFS arbitration we have also use the counters for this arbitration. These all are four only two masters. In round robin, Req0 will give grant to master 1 and Req1 will give grant to the master 2. Same as in the highest priority arbitration is there.

V. RESULTS

Here are the some waveforms taken with the help of RTL coding. All waveform are of different conditions. First of all, there are simulation waveforms of all three arbitration scheme:

A. Round robin arbitration:

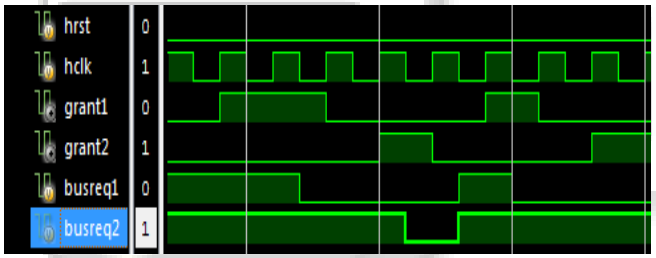

Fig.6: Waveform of Round robin

B. Highest priority arbitration:

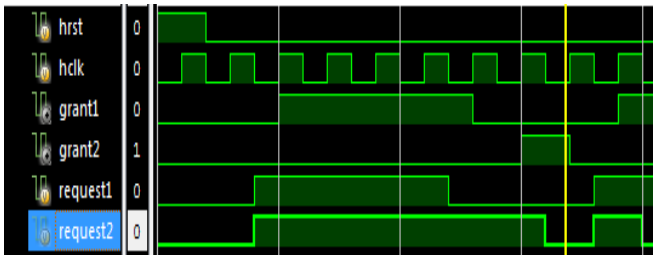

Fig.7: Waveform of Highest priority

C. First Come First Serve arbitration:

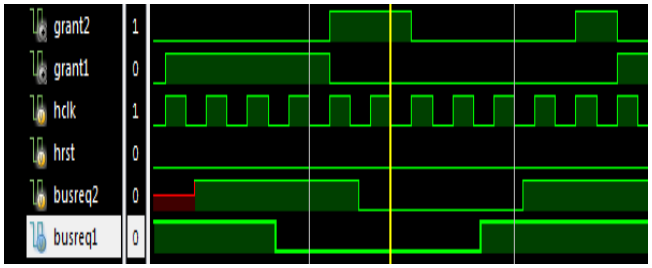

Fig.8: Waveform of FCFS arbitration

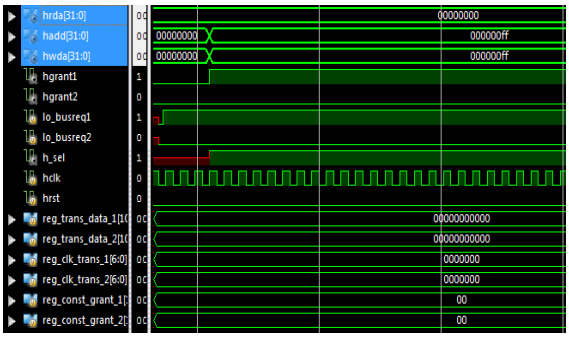

Fig.9: Waveform of Arbiter

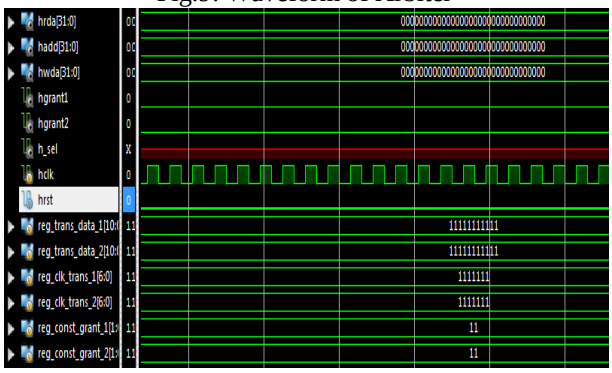

Fig.10: Waveform of Arbiter when values of respective master excides

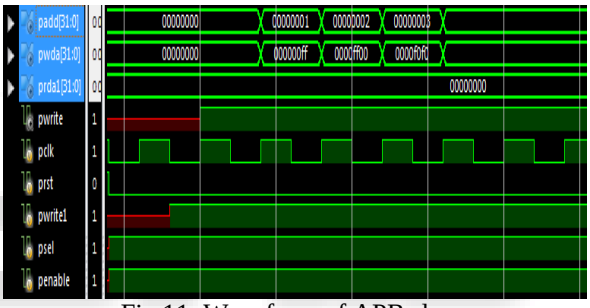

Fig.11: Waveform of APB slave

Values of reg_data_trans, reg_clk_trans and reg_const_grant are of respective master which gives real value of how master has transfer data and all those. Values of these respective counters will compare with the register bank. If values of these counters are less than value of registers, further process of arbitration scheme will run otherwise it will reject grant of particular bus request.

VI. RTL & REPORTS

The RTL coding of the proposed arbiter is done in verilog language using Xilinx ISE tools. All the timing, power reports are also display here.

A. Timing report

Fig.12: Timing report

By providing various clock, input delay and output delay, we can get the timing summary that is shown above for this design.

B. Rtl view

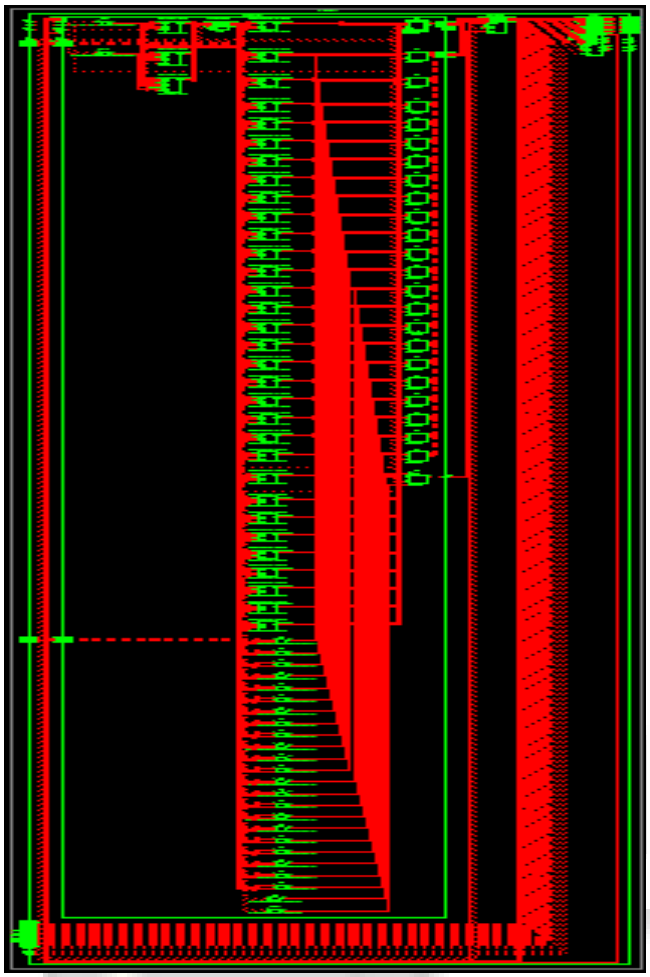

Fig. 13 rtl view

RTL view created by doing RTL coding in Xilinx is shown above how top module of the proposed plan looks like.

C. Power report

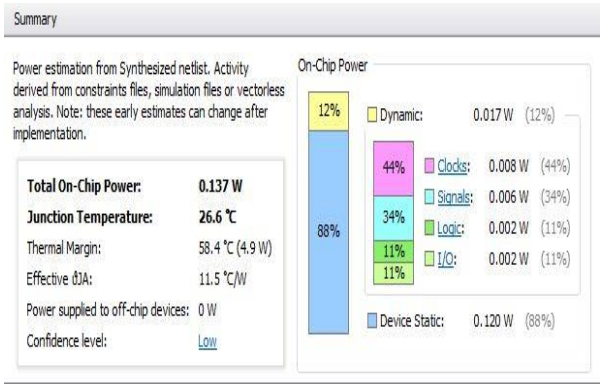

Fig.14: Power report

As observed from the results, most power is utilized by signals used in the design and total on-chip power is 0.137 w. This power report is generated on basis of synthesized netlist and also we can see the timing report where all timing constraints are working well.

VII. Conclusion

In this proposed plan, some basic ideas from previous paper are accepted as basic and then after this efficient arbiter is design on the verilog language. Three different arbitration schemes is basic idea from paper. Also give some logic which helps in decision of approve grant to master. This logic is of register banks which are configurable, so that values can be adjusted as per need.

Acknowledgement

Any accomplishment requires the effort of many people. So I take opportunity to heartily thank our project guide respected Mrs. Shaila Hooli for her valuable guidance and touch of inspiration and motivation throughout research work without whose help the work would not have been in the shape what it is.

References

[1] Ashutosh Kumar Singh, Anurag Shrivastava, G.S.Tomar, "Design and Implementation of High Performance AHB Reconfigurable Arbiter foronchip Bus Architecture",Communication systems and Network Technologies (CSNT), 2011 International Conference. Page. No-455,456.

[2] Mary K. Vernon, Udi Manber, "Distributed Round-Robin and First-Come First-Serve Protocolsand Their Application to Multiprocessor Bus Arbitration",Computer Architecture, 1988.15th International Symposium. Page no-271,272.

[3] Yu-Jung Huang, Ching-Mai Ko, and Hsien-Chiao Teng, "Design and Performance Analysis of a Reconfigurable Arbiter", WSEAS TRANSACTIONS on ELECTRONICS, Issue 4, Volume 5, April 2008. Page no-132 to 135.

[4] AMBA specification (REV 2.0), Copyright ARM limited 1999.

[5] AMBA 3 AHB-Lite Protocol specification, Copyright 2001, 2006 ARM Limited.

[6] "Introduction to AMBA 4 ACE and big. LITTLE Processing Technology", June 2011-www.arm.com/files/pdf/CacheCoherencyWhitepaper_6June2011.pdf.

