International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 04 Issue: 08 | Aug -2017 www.irjet.net p-ISSN: 2395-0072

Area Efficient VHDL implementation of AHB arbiter IP

Tamanna Sheikh[1], Dr. Pranay Joshi [2], Dr. Sunil joshi[3]

[1] M.Tech. Student, Department of Electronics and Communication, CTAE, Udaipur, Rajasthan, India
[2]Faculty, Department of Electronics and Communication, CTAE, Udaipur, Rajasthan, India
[3]Professor & Head, Department of Electronics and Communication, CTAE, Udaipur, Rajasthan, India

---

Abstract - *In SoC a processor needs to interact with other processors,memories or input/output devicescto complete the task. In SoC data integrity become most vital challenge and communication planning under different processors with shared bus system needs bottleneck free communication.To justify the multiprocessor environment a central bus controller is required called as arbiter. The propose work develop the IP of AHB arbiter for AMBA bus solution in VHDL. Proposed arbiter can access handshaking signals of 16 master and split capable slave and give grant to respective master according to priority defined by round robin algorithm. The design is implemented and tested in Xilinx. Further area, speed and power is calculated as a performance evaluation parameters.*

Key Words: **AMBA, Arbiter, Round-Robin, IP, SOC, VHDL.**

1.INTRODUCTION

Advanced Microcontroller Bus Architecture (AMBA) [1] is not properly speak a bus, but a family of buses, defined by ARM. The main buses of this Family are Advanced eXtensible Interface (AXI), Advanced High-Performance Bus (AHB) and Advanced Peripheral Bus (APB). APB [2] is a bus for interconnecting low-rate IPs, i.e. having little data to be transferred, for example keyboard-like IPs or UART (Universal Asynchronous Receiver Transmitter asynchronous, managing serial links). APB allows a very simple and very small read / write operations, with only one master per bus. In particular, this bus allows only unit operations, that is to say transfer a single data word by transfer request. As soon as the IP becomes more bandwidth-intensive, APB and its specifications are becoming a limiting factor. In particular, the burst mode, which allows several words to be transferred after a single query operation, is not present in APB.AHB is a bus that allows reads / writes of different sizes and supporting the burst mode, as well as several masters. The different masters in competition for access to the bus, the latter has an arbitrator to distribute access. It is possible to perform operations on 4, 8 or 16 successive words, but also on unspecified burst sizes, which will continue as long as the requesting IP requires it, or the bus arbitrator decides to give Access to another IP. This bus also has error signals allowing slaves to notify the master who initiated the transaction that not succeed. These two types of buses are relatively conventional, each having its interest depending on the type of IPs to be linked.In a system is constantly increasing. Depending on the case, access to a single bus, central, for all IPs, can constitute a limiting bottleneck.

2.AMBA BUS

The AMBA 2.0 bus is a built-in bus in SOC design. The AMBA standard is documented, free access and is used for System-on-chip processors (example: smartphones). The specification contains BUS types: AHB (Advance High-performance Bus), ASB (Advance System Bus) and APB (Advance Peripheral Bus). The arbitrator / bus controller is configurable in order to perform various arbitration modes, "Round-robin" or "priority-fix" mode. When a query from a master is supported by the BUS arbiter, the request is transmitted to all the slaves and the arbitrator selects the slave concerned. The AHB bus is multiplexed and therefore can be implemented in FPGAs, as can be moreover see it in the following figure:

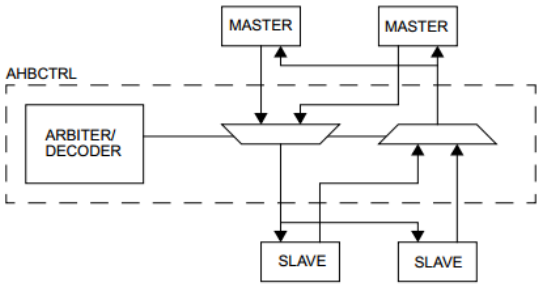

Fig-1: AHB controller principle diagram (AHBCTRL)

The bus arbiter allows one master at a time to communicate to the slaves. Since the processors are master and the slave memory controller are on the AHB bus, our component will have to monitor the actions of the components (AHB master) that access the memory controller. The bus arbiter allows one master at a time to communicate to the slaves. Since the processors are master and the slave memory controller are on the AHB bus, our component will have to monitor the actions of the components (AHB master) that access the memory controller.

3.Literature Review

The author present a self-motivated arbitration scheme according to data length transfer form master [4]. They have combined the length and priority to process the arbiter, this scheme work well with priority, but dynamic behaviour claimed, will create bottleneck under different clock structure [5]. Author present similar type of work by multilayer based design approach to justify the self-motivated arbitration, for dynamic behaviour justification of arbitration. But it cost additional hardware to create different layer for arbiter and maximum allowable latency is already defined as 16 clock for any type of transfer, hence multilayer approach for AHB2.0 is not suggestible, when concern of area is there. In [8], authors explore the performance under different beat and burst mode of operation in AHB arbiter, and its impact on power and throughput.[8] Most of work carried in 4 master and single slave To show the arbitration, our proposed work takes 16 master [9] with burst and beat capabilities and slave with split capabilities.

4.Arbiter Design

The proposed arbiter consist of priority shift and priority logic with round robin algorithm. There will be separate controller and counter to track the operation according the operation initiated by master in AHB system. There will be some or gates and an encoder to combine the different grants from arbiter. The flows follows the sequence initiated by different master as a bus request to access the bus, arbiter takes the responsibility for the monitoring of handshaking signals from different slave and controls from master. According to priority arbiter generates the grant to respective master. The following are the objective or scope of work.

Objective :

- To design round robin arbiter in VHDL.
- The arbiter is capable to handle 16 bus request and gives centralized command as a grant to respective master in priority list. To avoid bottleneck, protocols claims maximum latency of 16 clk.
- Split and error handling capable features in arbiter according to protocol.
- Operation and control according to burst mode.

4.1 Priority block:

The priority logic block is implemented through FSM approach. 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 their priority .In1 signal is the input for this block that is nothing but the Bus_req, Out1,Out2…….Out16 these are the Grant signal as output, these signal is further OR and sent to the output port,as Shown in figure 2.

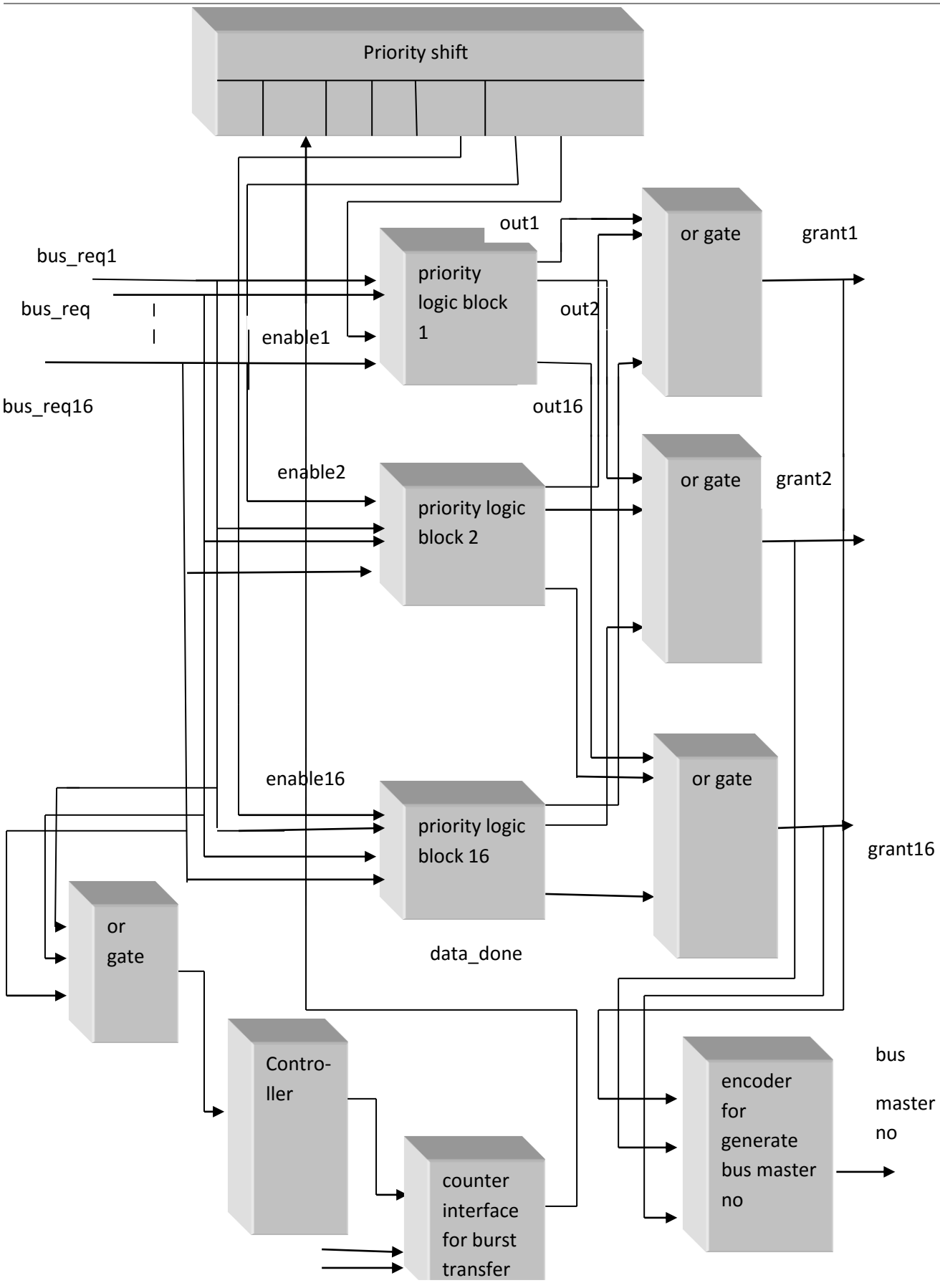

Fig 2: Block diagram of AHB arbiter

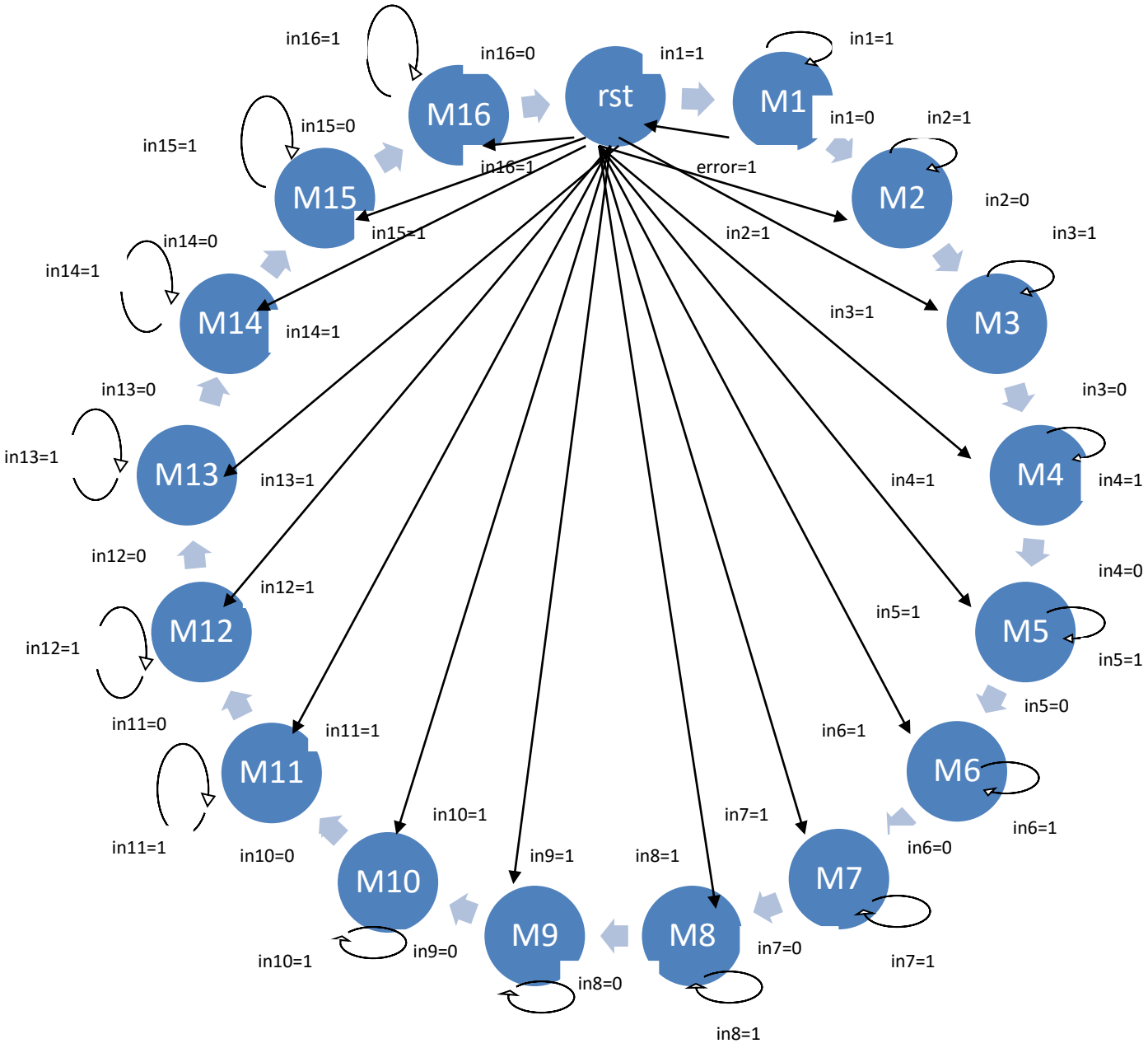

Fig 3: State diagram for priority logic

4.2 Counter and Mux

This block contains 16:1 Mux, and Counter. The counter is used here to control the all operation of arbiter with respect to the master and slave input. All Bus_req are mapped with Mux and with the help of select line Mux will select the appropriate master Busreq ,because the undefined burst is depend upon the Master bus_req.As counter is used to generate the start of transfer and end of transfer Depends upon the burst.

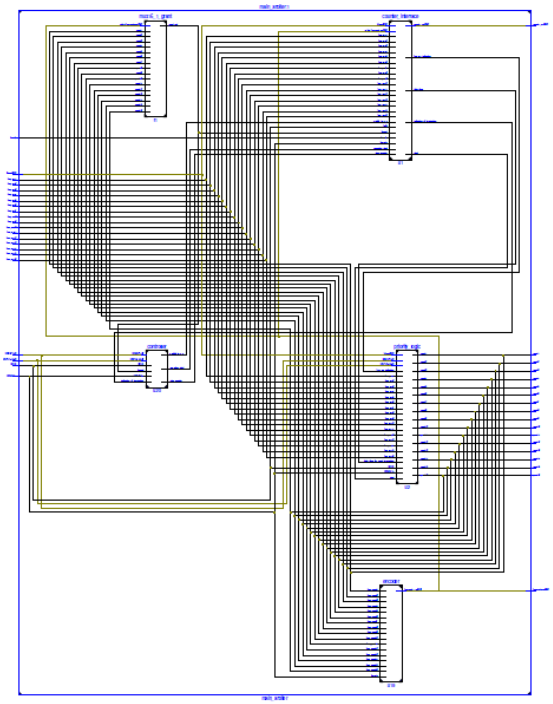

Fig-4: RTL schematic for top entity

Synthesis Report:
Device utilization summary:
Selected Device : 4vlx15sf363-12
Number of Slices: 1273 out of 6144 20%
Number of Slice Flip Flops: 830 out of 12288 6%
Number of 4 input LUTs: 2297 out of 12288 18%
Number of IOs: 64
Number of bonded IOBs: 64 out of 240 26%
IOB Flip Flops: 4
Number of GCLKs: 1 out of 32 3%

Timing Summary:
Speed Grade: -12
Minimum period: 3.130ns (Maximum Frequency: 319.519MHz)
Minimum input arrival time before clock: 6.164ns
Maximum output required time after clock: 5.388ns

4. Simulation

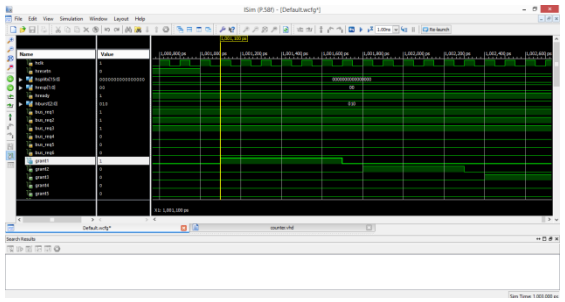

Fig-5: Simulation of integrated Arbiter

The above figure depicts the functionality of arbitration, where three request occurs simultaneously and grant1 gets first access, once the operation of Master 1 is over then grant is shifted to request 2,and so on accordingly.

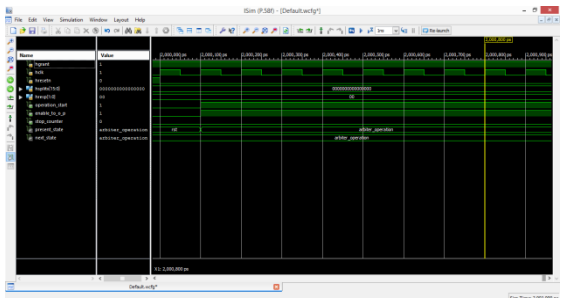

Fig-6: simulation of controller

The above figure depicts the simulation of controller, it is finite state machine which take inputs from other interface and gives control signal to counter for synchronous operation. It works in two state “rst” and “arbiter-operation”. Initially it will be in “rst” state, once start signal is there then it will shift to “arbiter-operation” state and gives outcomes as operation start signal high.

Xilinx Power Summary of ARBITER:

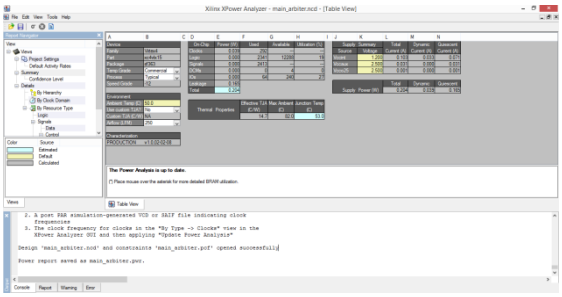

Fig-7: Power report of AHB arbiter

Table 2: Comparison Table:

	Previous design [9]	Proposed design
Number of Slices	1566	1273
Number of Slice Flip Flops	533	830
Number of 4 input LUTs	2752	2297
Number of bonded IOBs	64	64
IOB Flip Flops	4	4
Number of GCLKs	2	1
POWER	N.A	198.21mW
SPEED	N. A.	319.519MHz
No. of Master and Slave	16,1	16,1

The above table explains the comparative outcomes of proposed method. In earlier approach there was an interface block to connect the enable to all priority blocks, which is with more conditional statements, hence it has created the more latches with references of 4 input Look up tables. The earlier approach has used priority storage block along with finite state machine based controller, which cost additional hardware and complex handshaking. The proposed method removes priority storage and fsm instead of that priority shift is introduced with simple shift controller by data_done coming from counter to control the occurrence of shift. The proposed method has removed the interface block and directly connected the enable from priority shift to priority logic blocks. This approach has reduces the hardware from earlier work. Hence it is known for FPGA design approach there are dedicated flip-flops are available for design inference ,hence we have designed our code in such a way which more flip-flops rather than latches(4 input LUT),hence our Flip-flops counts are more than earlier work,. Flip-flops based design ensure more stability and predictability then latches. Overall methods give better hardware utilization than earlier approach.

5.CONCLUSION

The IP module has been designed as an Arbiter of the AMBA AHB bus [2], so which can communicate with the processor for it to configure, it also has the ability to observe transfers between the processor and a peripheral, and is capable of controlling the communication that occur in such transfers. The design consider the system model with bus request of 16 master and one slave. Additional feature of split control is also considered in proposed IP. The design has been developed using VHDL code and synthesized and simulation using Xilinx ISE. The designed is performed in Virtex device of Xilinx and claimed 5.5 % improvement in area occupancy in devices. The speed and power is calculated as 319.519MHz and 198.21mW respectively. The advantage of this design is that we have taken care of latch formation, with less latch & maximum flip-flop have enhanced our area efficiency.

REFERENCES

[1] https://www.arm.com/products/system-ip/amba-specifications
[2] Bacciarelli, L., Lucia, G., Saponara, S., Fanucci, L. and Forliti, M., 2006, June. Design, testing and prototyping of a software programmable I2C/SPI IP on AMBA bus. In Research in Microelectronics and Electronics 2006, Ph. D.(pp. 373-376). IEEE.
[3] Conti, M., Caldari, M., Vece, G.B., Orcioni, S. and Turchetti, C., 2004, June. Performance analysis of different arbitration algorithms of the AMBA AHB bus. In Proceedings of the 41st annual Design Automation Conference (pp. 618-621). ACM.
[4] Shete, P.S. and Oza, S., 2014. Design of an AMBA AHB Reconfigurable Arbiter for On-chip Bus Architecture. International Journal of Application or Innovation in Engineering & Management, 3(5), pp.245-252.
[5] Hwang, S.Y., Kang, D.S., Park, H.J. and Jhang, K.S., 2010. Implementation of a self-

motivated arbitration scheme for the multilayer AHB busmatrix. IEEE transactions on very large scale integration (VLSI) systems, 18(5), pp.818-830.

[6] Conti, M., Caldari, M., Vece, G.B., Orcioni, S. and Turchetti, C., 2004, June. Performance analysis of different arbitration algorithms of the AMBA AHB bus. In Proceedings of the 41st annual Design Automation Conference (pp. 618-621). ACM.

[7] Jamadagni, T. and Kumar, J.M., Implementation of a Special Arbitration Scheme for AHB of AMBA.

[8] Mitić, M. and Stojčev, M., 2006. An overview of on-chip buses. Facta universitatis-series: Electronics and Energetics, 19(3), pp.405-428.

[9] Gautam, P.K. and Upadhyay, N., A Novel Arbitration Technique of AMBA AHB.