ASIAN JOURNAL OF ELECTRICAL AND ELECTRONIC ENGINEERING
Vol. 1 No. 2 2021
E-ISSN: 2785-8189

Multiprocessor Arbitration for AMBA Interface in ASIC

M. I. R. Rokon [1]*, S. M. A. Motakabber [1], AHM Zahirul Alam [1],
M. Hadi Habaebi[1] and M. A. Matin [2]

[1]*Dept. of Electrical and Computer Engineering, Faculty of Engineering,
International Islamic University Malaysia, Kuala Lumpur, Malaysia*
[2]*Dept. of ECE, Faculty of Engineering, North South University, Dhaka, Bangladesh*

*Corresponding author: iqbal.rokon@gmail.com

(Received: 6th August 2021; Accepted: 15th September 2021)

Abstract— This paper addresses the multiprocessor arbitration for any System on Chip or ASIC. Any system, be it simple controller or very supplicated system, it needs processor to operate. There are series of processors offered by Intel, AMD or processor companies. Previously single processor used for any chop to access different targets. But technology advances, industry felt the need of multiprocessor access for higher performance. In order to allow multiprocessor to access it targets, system needs an efficient interface with very sophisticated arbitration system. This paper carried out the research to come up with an improved algorithm of hardware to allow multiprocessor access to the system. In order to design the hardware, modern HDL based design methodology has been used. There are two industry standard HDL by IEEE – VHDL and Verilog. Here Verilog is used. In HDL based d=hardware development, simulation is most important part to very verify design's functionality and make sure its 100% correct. Otherwise if any design problem goes forward undetected, that'll cost so much money and time in order to go back and fix, in some cases full respin. For hardware implementation Xilinx FPGA Device has been targeted on this research. AMBA bus protocol used in this research is the industry-standard protocol for processor access and very efficient and straightforward to use with any off the shelf macro available for the high tech industry.

Keywords: *Multiprocessor Arbitration, AMBA Bus Protocol, Processor Interface, FPGA, ASIC*

1. INTRODUCTION

In the past, a vacuum tube is an electronic device used in many older model radios, television sets, and amplifiers to control electric current flow. The transistor became cheaper in the 1960s and was much smaller, worked on lower voltages, and used less power. At this time, most radios, television sets, and amplifiers began using transistors instead. In an integrated circuit or IC, the components and interconnections are formed on the same substrate, typically a semiconductor such as doped silicon. An integrated circuit or monolithic integrated circuit (also referred to as an IC, a chip, or a microchip) is a set of electronic circuits on one small flat piece (or "chip") of semiconductor material, usually silicon.

Since the 1990s, chip technology has evolved to perform a complex operation as per the current industry's needs. To keep up with this advancement, a single need to integrate so many gates, ten million, is very common nowadays to deliver all the sophisticated controlling systems. In addition to size, speed and power dissipation became critical factors too. A single processor which was previously used is not capable enough to comply with this need. So current SoC or ASIC go for multiprocessor access. This multiprocessor needs an advanced arbitration system to have efficient access by the processor.

This research considers all these important factors and develops an improved version of the arbitration system for the multiprocessor interface. The arbitration controller is developed for AMBA AHB access by multiprocessors and performs all sorts of operations to access different targets under memory mapping. Verilog Hardware Description Language (HDL) was used to model the efficient arbiter for AMBA interface, verification was done Verilog simulator and hardware implementation was done using Xilinx Pegasus FPGA device.

2. AMBA PROTOCOL AND ARBITRATION

AMBA (Advanced Microcontroller Bus Architecture) is a high-performance backbone bus AHB that holds Processor, DMA controller, on-chip memory to access slave targets like register, RAM, FIFO, UART. etc. Multiprocessor access operation requires an arbitration inside its interface to allow any selected bus master to control the bus. The arbitration scheme inside the interface resolves the bus master access scheme based on the priority shown in Fig 1.

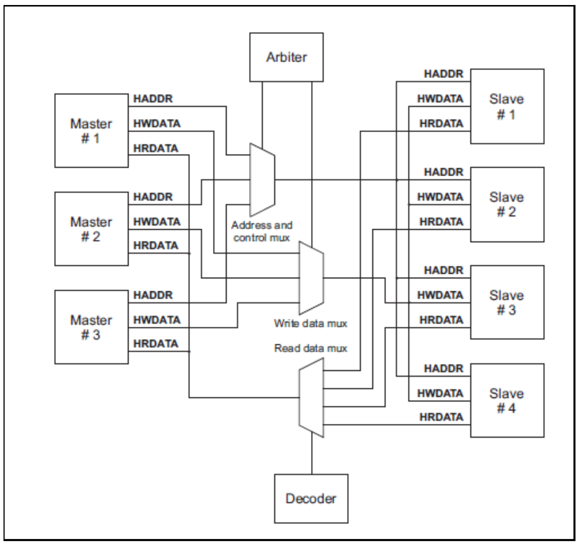


Fig. 1. AMBA Multiprocessor Interconnection [1].

In the arbitration scheme, only one out of multiple bus masters can access the bus to perform the access operation. Multiple processors assert a bus access request. Based on its arbitration algorithm, the arbiter in this research decides which master gets the priority and asserts the grant signal accordingly to give that master control of the bus [1]. In order to figure out how long a bus would be granted access, the arbiter looks at the value HTRANx[1:0] signal of that master and keeps asserting the grant signal as long as NOQ and NONSEQ transfers have occurred Signal description [1].

The signals involved in the AMBA Multiprocessor bus arbitration scheme have been discussed below [2]:

2.1 HCLK (Host bus clock):

AHB system clock HCLK is connected to all the flip flops in the address path, data path and control signals path to provide sequential access. The external system clock generator generates the system clock and drives all the signals in the system. As per AMBA AHB bus protocol, all signal timings are positive edge-triggered by HCLK.

2.2 HRESETn (Bus reset control signal):

The system reset signals comes from the reset controllers for AHB masters. It's active-low, i.e. it resets all flip flops inside the system and put those in a known value at the output, which is 0 in this research. Anytime a low (0) value at HRESETn signal will reset any sequential block in the system and put all state machines in a known state, address bus, data bus and control signals in known value. Both synchronous and asynchronous reset can be used. In this research work, synchronous rest has been used.

2.3 HBUSREQx (Bus request):

Whenever any AMBA processor needs to use the bus for access, it asserts it request a signal, HBUSREQx indicating its execution of the operation. Arbiter records that request and understands that that processor needs the grant to access the bus.

2.4 HTRANSx[1:0] (Bus transfer type):

2-bit signal HTRANSx comes from a particular master x and indicates its current transfer type of the master. It allows 4 types of bus access: IDLE, BUSY, SEQUENTIAL or NONSEQUENTIAL. Each of these transfer types is represented by a unique value or a bit combination of the HTRANSx signal.

2.5 HGRANTx (Bus grant):

A signal HGRANTx indicates that any bus master is allowed to get on the AHB bus. The central arbitration system generated this signal based on the arbitration scheme. It takes into account locked and SPLIT transfers to specify which master amongst the ones attempting bus access would get the highest priority. A locked transfer allows a master with bus access to complete a transfer by denying access to other masters requesting access to the bus. Similarly, a SPLIT response issued by a SPLIT-capable slave prevents new bus access unless it signals to the arbiter that it is ready to complete the transfer [1].

2.6 HMASTERx[3:0] (Master number):

A 4-bit signal, HMASTERx, where x denotes the master number, is generated by the arbiter. It indicates which master has been granted access to the bus and is currently performing a data transfer.

3. AHB BUS PROTOCOL IN AMBA BUS ARBITRATION

AMBA address decoder uses the processor address to create a select target signal for access operation [3-4].

3.1 Granting Bus ACCESS

The arbiter block resolves which processor would get bus access by using its arbitration controller and generates its grant signal and assigns the bus master number in HMASTER[3:0]. Fig 2 shows how arbiter issues GRANT signal to allow bus master as per its request and value of HTRANS as described in Fig 2.

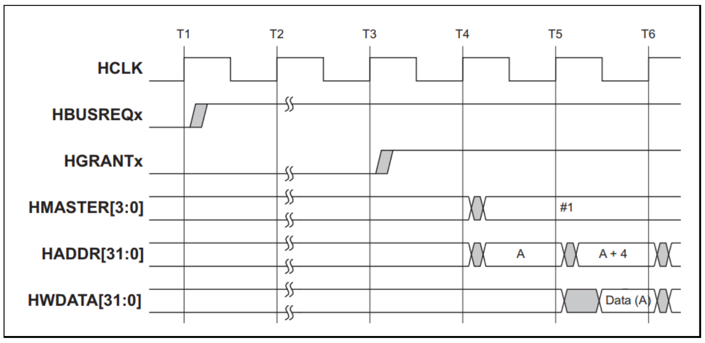


Fig. 2. AMBA BUS Request and Grant Protocol [1, 3].

3.2 Handover Bus Between Processors:

In a multiprocessor arbitration, the arbiter changes the current processor's HGRANT signal before the last address sampled and generates the GRANT signal for the next processor according to priority as it sends a request signal and according to the arbitration scheme, which is shown in Fig. 3.

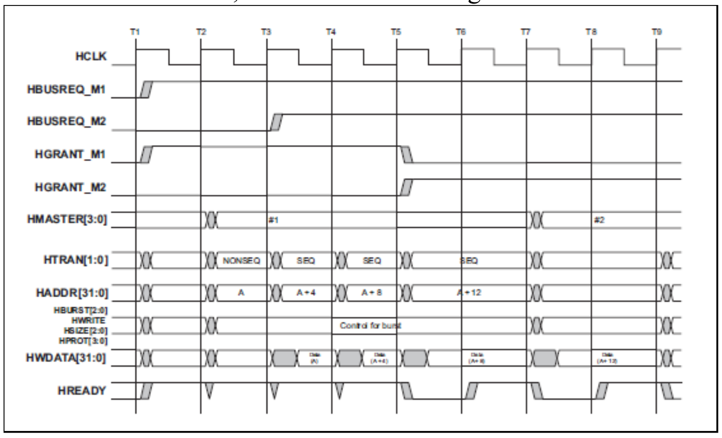


Fig. 3. Bus ownership for AMBA Multiprocessor [5].

3.3 Bus Master Grant Signal

Arbitration controller asserts HGRANTx as per the arbitration scheme that allows the corresponding processor to control the bus and initiate operation using its address, read/write data bus, and control bus.

4. HARDWARE DESIGN OF AMBA ARBITRATION SCHEME

Several types of schemes can be followed to implement an Arbitration system: fixed priority, daisy chain or round-robin. As in Fig. 4, fixed priority is implanted in this research using an efficient algorithm for faster arbitration targeting and low-sized faster grant signal generation logic [5 -6].

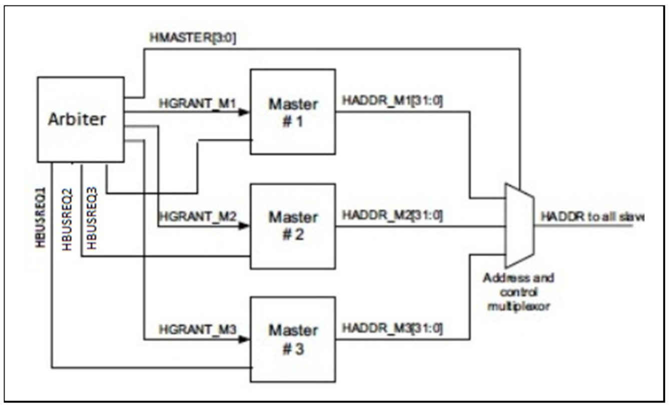


Fig. 4. AMBA bus arbitration scheme.

Three AMBA processors have been used on this multiprocessor arbitration system. AHB bus clock signal and the associated bus transfers have been timed

For bus master operation, AMBA processors use HBUSREQ1, HBUSREQ2 and HBUSREQ3. REQUEST SAMPLER module samples those request signals in HCLK and generates hbusreq1_a, hbusreq2_a and hbusreq3_a. The arbitration controller receives the sampled request and, as per the priority by the arbitration scheme, generates a pre-grant signal to allow the granted bus to do access operation. The grant receiver block samples the pre-GRANT signal to send the AHB grant signal to the processor. It allows the processor as long as it issues the NONSEQ cycle and keeps issuing the SEQ cycle. Additionally, the arbiter also sets the value of HMASTER. This 4-bit signal denotes which master has been granted access to the bus and is currently performing a data transfer to allow address, data, and control signals master to forward to the bus.

5. TOP-LEVEL BLOCK

The top block of the hierarchy, as shown in Fig. 5. It receives the AHB clock, reset, bus request signals from all the bus masters and as per the property of the arbitration scheme. In addition, the data transfer length creates a grant signal to allow a specific corresponding processor to control the AMBA bus. It also sends the write data and reserves read data.

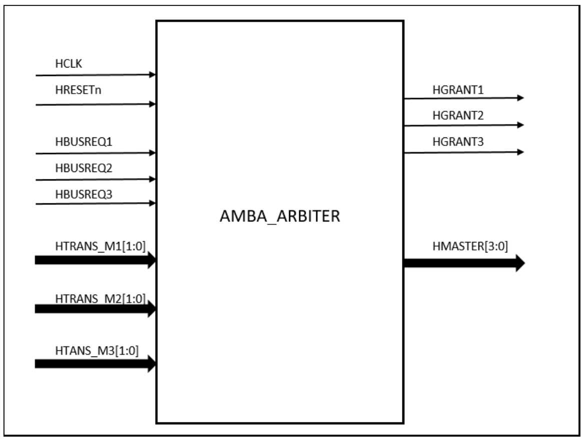


Fig. 5. Top Level Block of AMBA BUS Arbiter [7].

6. BLOCK DIAGRAM

The arbiter's top block consists of three sub-blocks, as in Fig. 6: Request Receiver, Arbitration Controller and Synchronous Grant Signal Generator for a processor. The AHB clock HCLOCK clocks all the data and control and the system is reset by AHB reset signal HRESET.

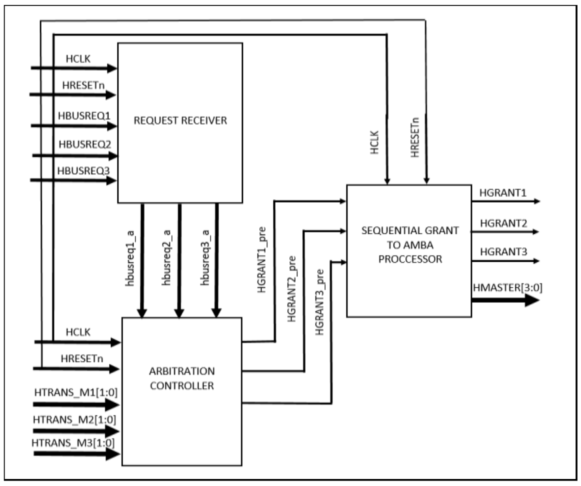


Fig. 6. Block Diagram of AMBA BUS Arbiter [7].

6.1 Request Receiver

Request receiver in Fig. 6 accepts bus requests signals from processors with the AHB clock HCLOCK after the request is asserted and then forward it to the arbitration controller.

6.2 Arbitration Controller

The main module of this multiprocessor access arbiter system is the arbitration controller in Fig. 6. is the major block of the arbiter. It uses a state machine to move the priority arbitration as requested by the bus request signals HBUSREQx and then, based on the transfer type of the bus masters, keep granting the HGRANTx signals to the allowed bus for its read-write access. Finally, it sets the value of HMSTER, which is used to figure out which address, HREAD, HWRITE and control signal will get control of the bus [5].

6.3 Synchronous Grant to AMBA Processor

The primary purpose of the block is to sample all the grant signs signals with the HCLK signal and send those grants to requesting processor. This sampling is done to ensure no setup and hold time violation occurs in static timing analysis after hardware implementation.

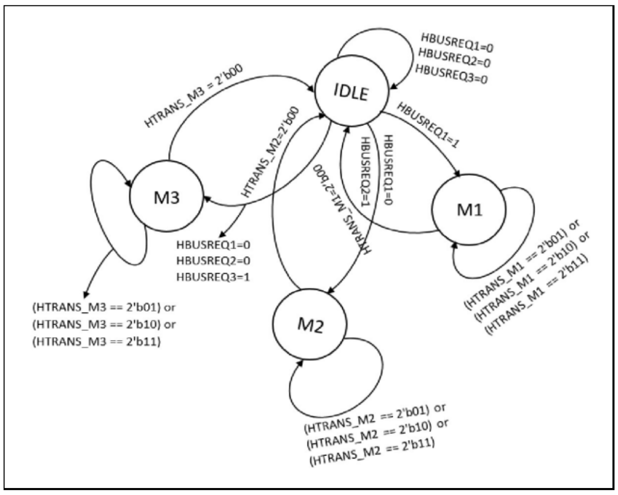


Fig. 7. Proposed Smart Arbitration State Machine Diagram [7].

6.4 *State Machine*

The state machine shown in Fig. 7 is a mail controlling part of the designed interface since it controls the priority, arbitration scheme and length of processor access. Three bus master M1, M2 and M3 are used in the design and the state machine is responsible for allowing access to them as per priority and avoid bus starvation. Two of the most significant categories of signals associated with the state machine are the bus request and transfer type signals generated by the processors. These signals' values give rise to certain transition conditions that cause transitions between the IDLE and either of the three data transfer states or transitions from one state to other.

6.5 *State Diagram*

The state diagram shows the state transitions of the state machine based on the current state and input, decides the next state and output. State transitions must consider processors priority scheme and any processors access length for the burst access. Based on the current priority, only a specific processor is granted access to the bus at a given time. These state transitions are shown in the state diagram in Fig. 7 and described below:

a) IDLE State (2'b00)

This state indicates there is no bus request from the processor.

b) M1 State (2'b00)

Processor M1 issues the BUSREQ1 signal and gets the bus control right away by the HGRANT1 signal as it has the highest priority. State machines stay in this state as long as master transfer data by issuing one NONSEQ and several SEQ transfers using the HTRANS_M1 signal.

c) M2 State (2'b01):

This state indicates that processor 1 doesn't have the request signal and processor 2 sends the request signal. It asserts HGRANT2 and keeps it high as long as processors issues one NONSEQ and several SEQ transfers using the HTRANS_M2 signal.

d) M3 State

This state indicates that processor 1 and processor 2 don't have the request signals and processor 3 sends the request signal. It asserts HGRANT3 and keeps it high as long as the processor issue one NONSEQ and several SEQ transfers using the HTRANS_M3 signal.

7. SIMULATION AND RESULT

Fig. 8, Fig. 9 and Fig. 10 simulation results of the arbitration system are shown. The simulation shows all the bus masters tried to access the bus by corresponding bus request signal created in the test bench, and the arbitration state machine provided grants as per the priority scheme.

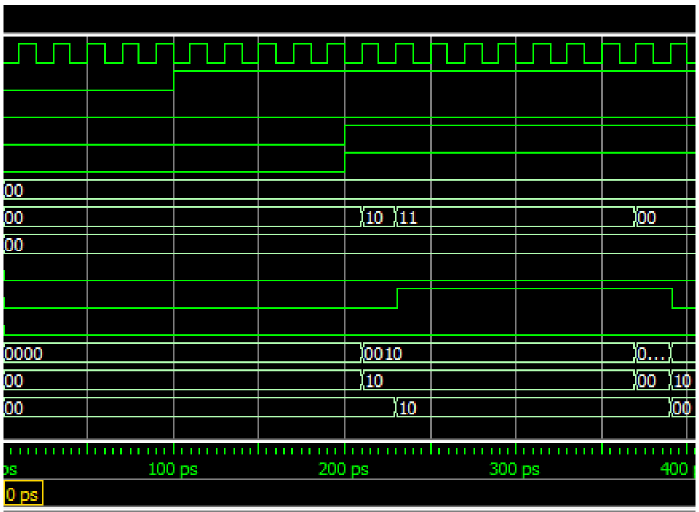


Fig. 8. Processor M1 and M2 send request, processor 2 gets grants as per priority.

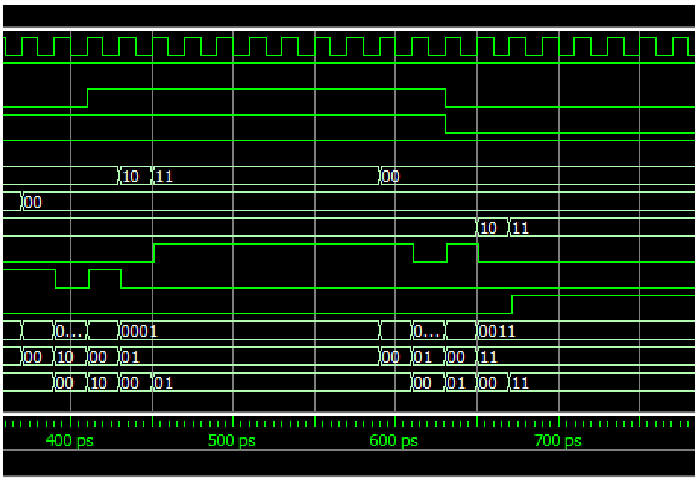


Fig. 9. All three processor send request, but highest priority Processor M1 gets the access.

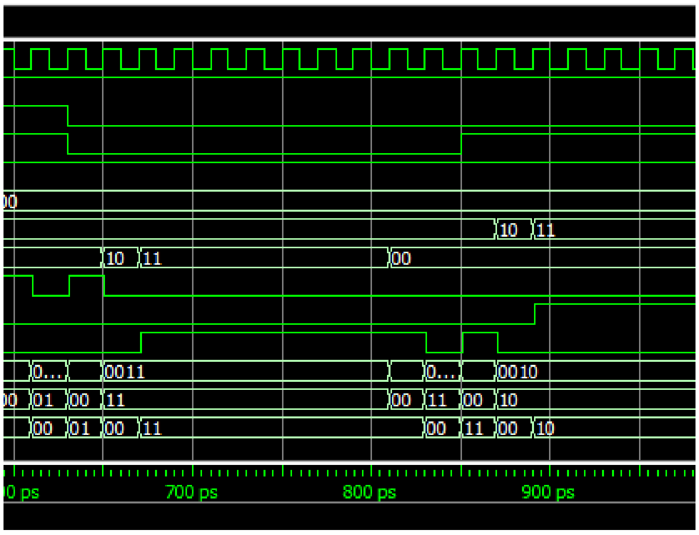


Fig. 10. No request from Processor M1 and M2, only processor 2 sends the request and automatically gets access as lowest property access.

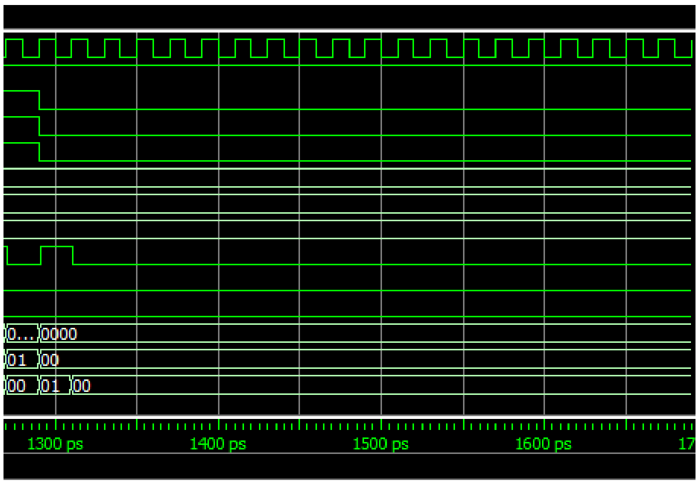
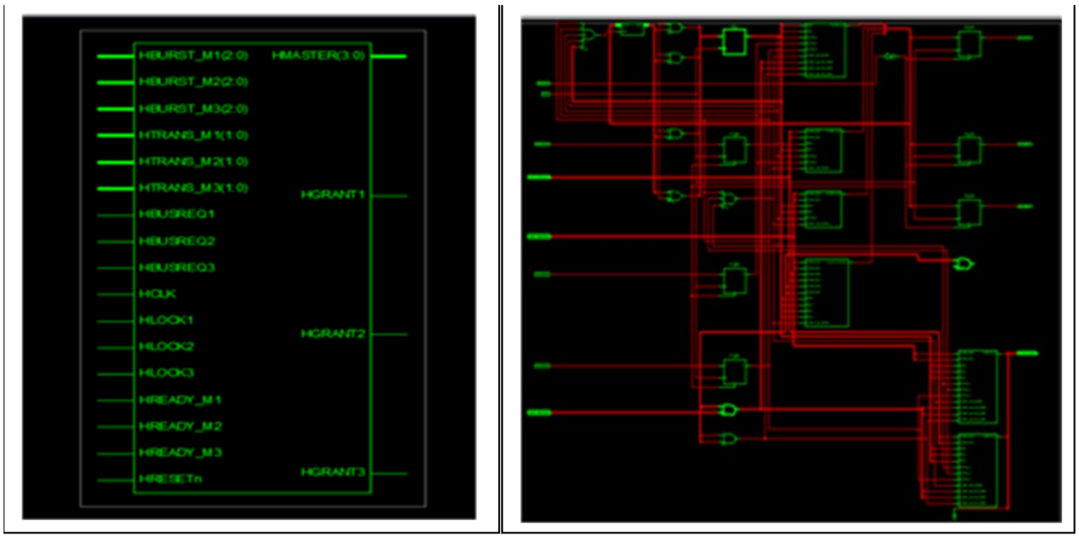


Fig. 11. No request from any processor, so bus activity and arbitration control state machine is in idle state.

8. SYNTHESIS RESULT

Fig. 11 shows the top-level block after synthesis. Fig. 11(b) shows the detailed schematic of the Arbitration Block after synthesis.



(a) Top Block (b) Schematic Diagram

Fig. 11. Top-Level Block after Synthesis.

9. CONCLUSION

Any SoC for all the high-tech application requires processor (external or internal) to run the hardware. Single processor is not enough to coop current need of complex and concurrent operations. This research targeted to use multi preprocessor three to access the ASIC. AMBA bus interface protocol is industry standard protocol for processor access. This research addressed the multiprocessor access by implementing an efficient arbitration process to allow processor to access the ASIC using its interface. Fictional design has been carried out by Verilog HDL and simulated using Cadence/ Modelsim Simulator. Finally, it was synthesized and implemented synthesized in Xilinx Pegasus FPGA device.

REFERENCES

[1] P. Giridhar and. P. Choudhury, "Design and Verification of AMBA AHB," 1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE), 19-20 Mar 2019, India.

[2] H. Saluja and N. Grover, "Multiple Master Communication in AHB IP using Arbiter," International Journal of Engineering and Manufacturing (IJEM), vol. 10(1), pp. 29-40, 2020.

[3] L. Deekhsa and B. R. Shivakumar, "Effective Design and Implementation of AMBA AHB Bus Protocol using Verilog," 2019 International Conference on Intelligent Sustainable Systems (ICISS), 21-22 Feb 2019, Palladam, India.

[4] J. Tu and C. Ou, "A practice of 2-to-2 fixed priority arbiter used to multi-core processors," International Conference on Applied System Innovation (ICASI 2017), 13-17 May 2017, Sapporo, Japan.

[5] A. Mishra, "Design of a Round Robin Bus Arbiter using System Verilog," International Research Journal of Engineering and Technology (IRJET), vol. 7(1), pp. 29-40, 2020.

[6] A. A. K. Qahtan and A. M. A. El-Kustaban, "A Bus Arbitration Scheme with an Efficient Utilization and Distribution," International Journal of Advanced Computer Science and Applications ((IJACSA 2017), vol. 8(3), pp. 1-6, No. 2017.

[7] M. I. R. Rokon, S. M. A. Motakabber, M. Hadi Habaebi, A. H. M. Zahirul Alam, M. A. Matin, "Smart Arbitration System for Multiprocessor AMBA Interface in System on Chip", 2021 8th International Conference on Computer and Communication Engineering (ICCCE).