RESEARCHERID
THOMSON REUTERS

*[Saluja,* 6(6): June 2019]
IDSTM-2019

ISSN 2348 – 8034
Impact Factor- 5.070

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

# DESIGN AND IMPLEMENTATION OF FSM BASED MEMORY CONTROLLER AND ITS INTERFACE COMPATIBLE WITH AMBA

**Hitanshu Saluja & Dr. Naresh Grover**
Manav Rachna International Institute of Research & Studies, Faridabaad, India

## ABSTRACT

With the recent development, the computing elements are being fabricated on chip which is known as "System-on-Chip". The connections in this system-on-chip (SoCs) system are provided through buses so that the processor can communicate with memory, input-output devices and with other processors to complete the specified task. To improve the performance we have to develop such efficient on chip architecture which will be much faster system on chip solution and removes the limitation of communication architecture. One of the arrangements is "AMBA-AHB bus".

This work elaborates the AMBA bus interface bridge between memory controller and other supporting peripheral. The work claims the design and implantation of AHB system i.e AHB bus master, AHB bus slave and memory controller compatible with AHB system with FIFO for buffering the request. The AHB master initiates the read and writes signals i.e. necessary for operation and AHB slave will respond to that signals. These signals can't directly communicate with memory so a slave to application interface is highly required. VHDL code is utilized to develop the design and it is synthesized in Xilinx Spartan 3 device (3s100evq100-5). The design claims improvement in both area and speed.

***Keywords:*** *AMBA, AHB, SoC, VHDL, Xilinx.*

## I. INTRODUCTION

The Advanced Microcontroller Bus Architecture (AMBA) is immensely used standard design and it is registered trade mark of ARM ltd. ARM is a leading manufacturer of processors and microcontrollers for embedded systems that used in many ultra modern electronic devices i.e. media players, mobile phones and especially in SoC (System-on-Chip) designs. ARM is recognized leader for IP (Intellectual Property) standards of AMBA. All the libraries of AMBA are provided by the ARM. AMBA system has been adopted by many companies nowadays and hence it is grabbing more and more attention. Thus, it has become a standardized system for the development of SoC interconnections.

Early born buses in AMBA family were Advanced System Bus (ASB) and Advanced Peripheral Bus (APB). A short time ago high-performance AHB bus introduced by AMBA used to contain more than one bus masters or at least have a testing interface and a processor. Digital Signal Processor (DSP) or Direct Memory Access (DMA) are the commonly used AHB Master and the most commonly used AHB slaves are any internal memory and external memory interface. Other peripherals could also be used as AHB slaves but the peripherals have low bandwidth will remain on APB due to their less energy consumption and quicker response. Ensuring the integration of any current design is necessary for bridging between the existing Advanced System Bus (ASB)/Advanced Peripheral Bus (APB) designs.

The components included in an AMBA-AHB design are:

***AHB Master***: Bus masters are generally used for reading and writing operations with the help of address and control data. But at a particular instant only one bus master can be active and use the bus which is known as AHB master.

***AHB Slave****:* Bus slaves are used to react read/write operation provided within an address-space range. It sends back an acknowledge signal to the bus master about the data transfer status i.e. success, failure or waiting is known as AHB slave.

***AHB Arbiter**:* The work of arbiter is to allow only one bus master at a time to transfer the data. The arbitration protocol is generally fixed, but according to the application requirements, arbitration algorithms like fair-access or high-priority algorithms can be implemented.

***AHB Decoder**:* The address provided for data transfer needs to be decoded. Hence, AHB decoder decodes the address and also sends a select signal to the slave which performs the transfer. For all AHB implementations, a single centralized decoder is needed.

AHB supports burst mode operation for data communication from processor to memory or other interface. It allows several words to be transferred after a single query operation, which is five to six times the normal operation. Early born buses of AMBA i.e. APB, ASB doesn't have burst feature that is why we turned to AHB. It is a bus that allows reads / writes of different sizes and supporting the burst mode as well as several masters. In case of multiple master arbitration scheme is required to distribute the access among multiple masters. AHB bus also has error signals allowing the slave to notify the master who initiated the transaction that not succeed.

## II. PROPOSED METHOD

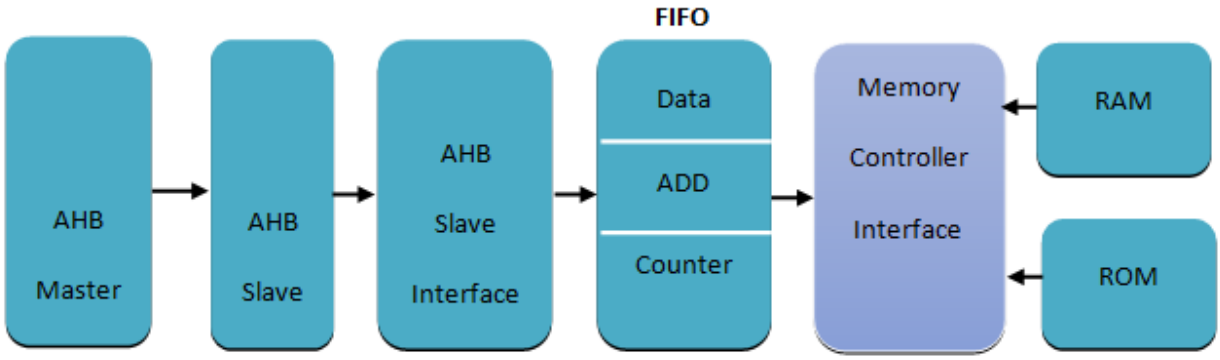


***Figure 1: Top architecture of AHB memory controller***

AHB compliant's master's top architecture is depicted in Figure 1. Here in the proposed method FIFO is used for data and control buffering. Depending upon the READ and WRITE operations data is further communicated through RAM or ROM.

**AHB Master**

The AHB Master is an interface unit which allows the processor to initiate data transfer to the AHB slave. The master takes care of address and data transfer between processor and memory. It generates the corresponding control signal as it is triggered by an arbitration unit as an HGRANTx signal. It supports burst mode operation and generates the sequence of operations according to all handshaking signals [11, 12, and 13].

ISSN 2348 – 8034
Impact Factor- 5.070

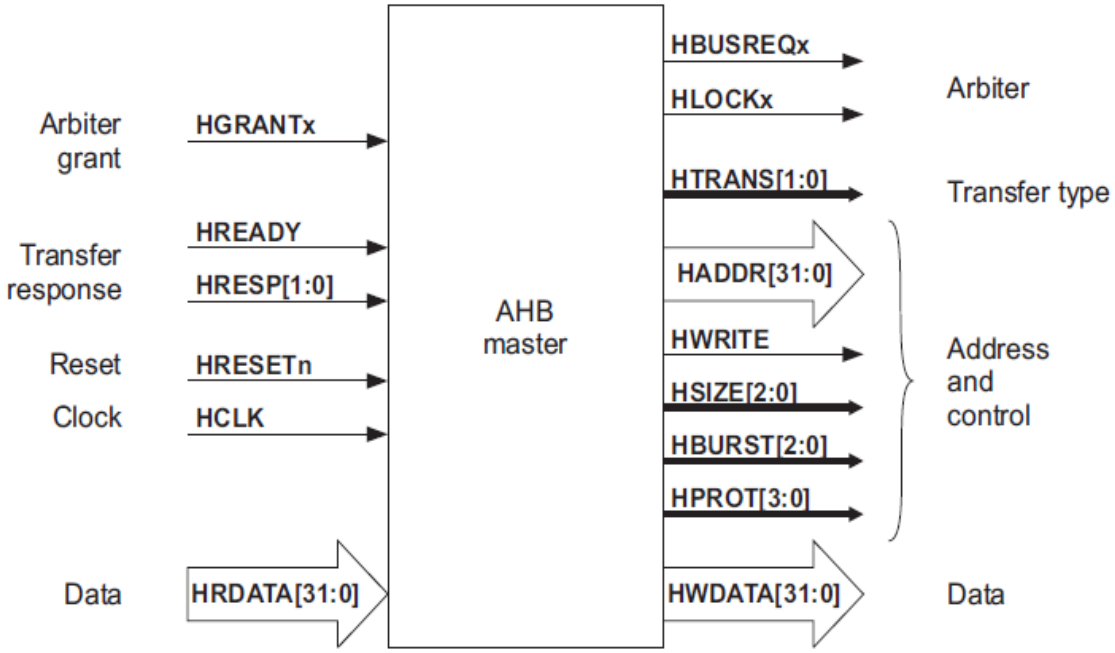


***Figure 1. AHB Bus Master Interface [14]***

The bus master handles all four types of slave response and its wait state, i.e. HREADY. It generates four transfer types of HTRANS to complete the handshaking with the other interfaces of the slave. Figure 1 shows the pin diagram of the AHB Master Interface, which is necessary for establishing the interface between the AHB Master and the AHB Slave with arbitration unit.

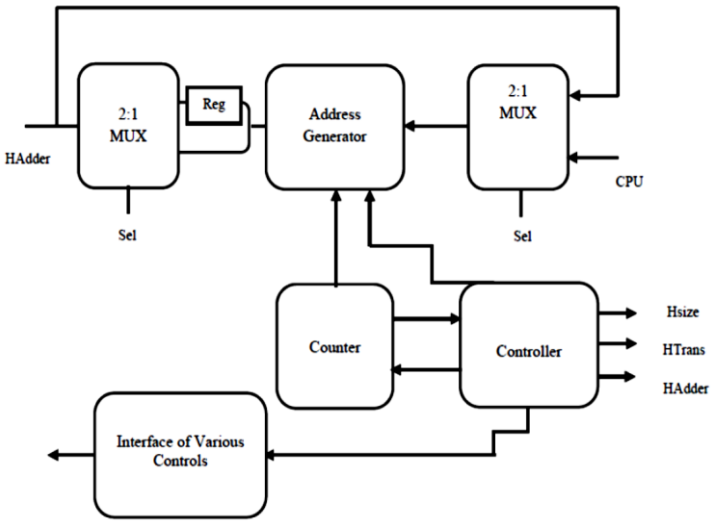


***Figure 2. Top Architecture of the AHB Master [15]***

Figure 2 shows an implementation model of the AHB Master. It consists of mainly five modules:

***Address Generator:*** It generates the address according to the HBURST signal. The initial address is initialized by the processor; the rest of address is generated by the AHB Master for support of multiple sequence of the operation. Figure 3 explains the internal block diagram of the Address Generator of the AHB Master.

***Counter:*** It counts the number of operations according to the HBURST signal. It generates a control signal to control the address generator. When HREADY is low, the counter stops incrementing its value and that value is entered in the wait state. HREADY is a handshaking signal from the AHB Slave to indicate the slave is ready to accept the data.

***Multiplexer:*** There are two multiplexers that are used for address communication with external interfaces. The multiplexer initializes the processor address based on non-sequential operation for the one clock cycle. After that, in the second clock cycle onwards, it takes the generated address as input for the next address generation based on sequential operation. The second multiplexer takes care of early burst termination events and stores the generated address in the register. The address can be used further once operation resumes.

***Interface of Various Controls:*** It is a collection of registers that buffers the value and assigns it to the concerned interface based on the control signal.

***Controller:*** It is a finite state machine implementation which controls the rest of the AHB Master interface internally. It generates the HTRANS signal for external handshaking with the slave.

**A Simulation Result of AHB Master**

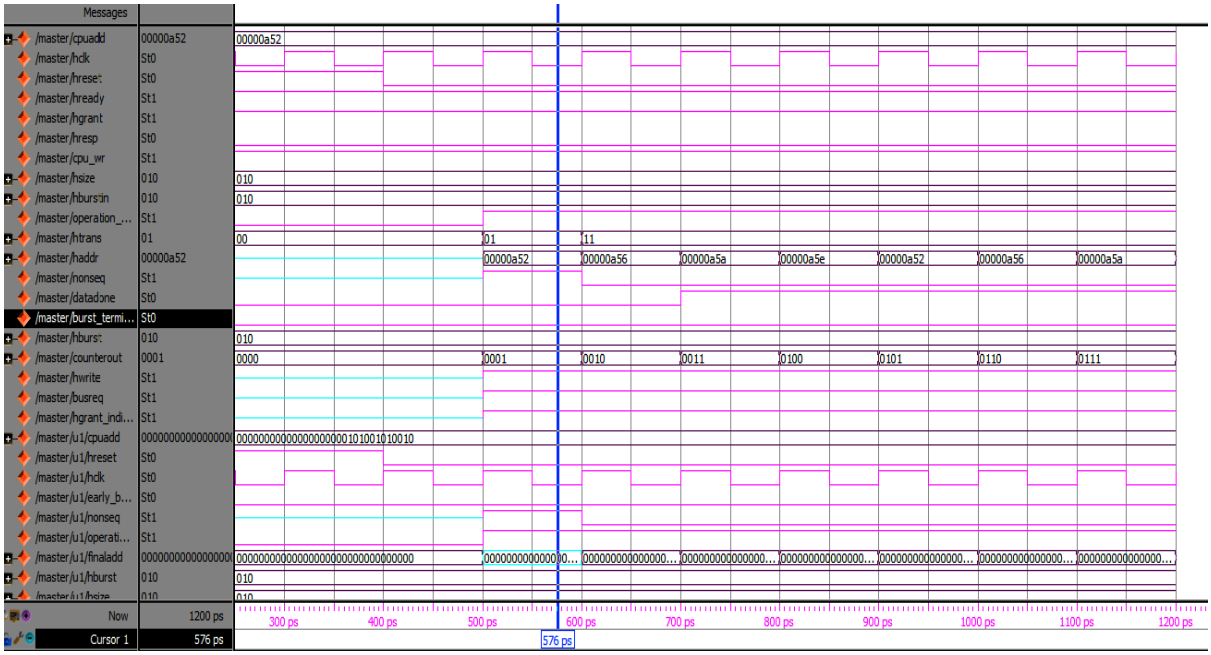


***Figure 5: Simulation result of AHB Master***

The above figure shows the simulation results of master. CPU initiates the first address and rest of address is generated by master according to HBURST and HSIZE. Master will generate HTRANS signal as a response. When htrans is 00 then master is in idle state, once htrans is 01 then master is in non-sequential state and that the start of first operation. When htrans is 11 then master enters in sequential state, it generate address according to Hburst.When hready is low then master will not generate it next address. Once hready is high then master starts is next address generation.

ISSN 2348 – 8034
Impact Factor- 5.070

**AHB Slave Bus**

An AHB bus slave shown in Figure 6 responds to transfers initiated by bushmasters within the system. The slave uses a HSELx select signal from the decoder to determine when it should respond to a bus transfer. All other signals required for the transfer, such as the address and control information, will be generated by the bus master [13].

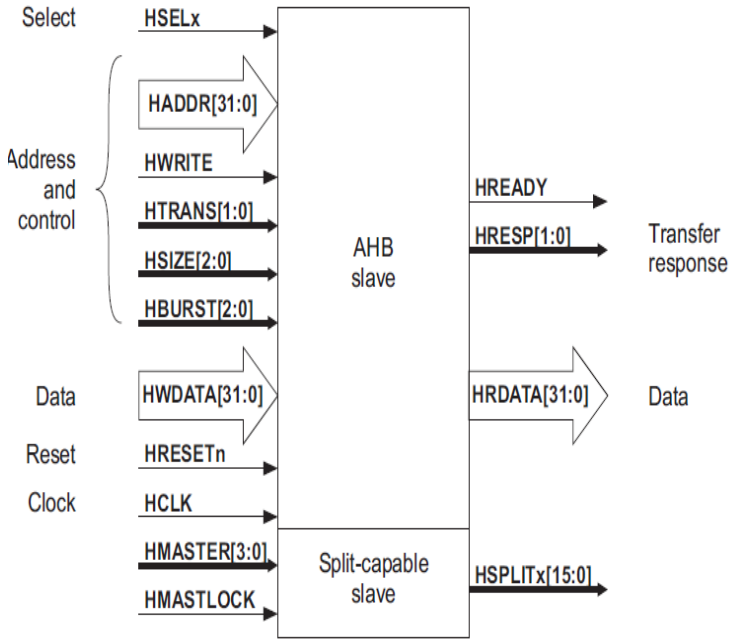


*Figure 6: AHB bus slave interface [2]*

ISSN 2348 – 8034
Impact Factor- 5.070

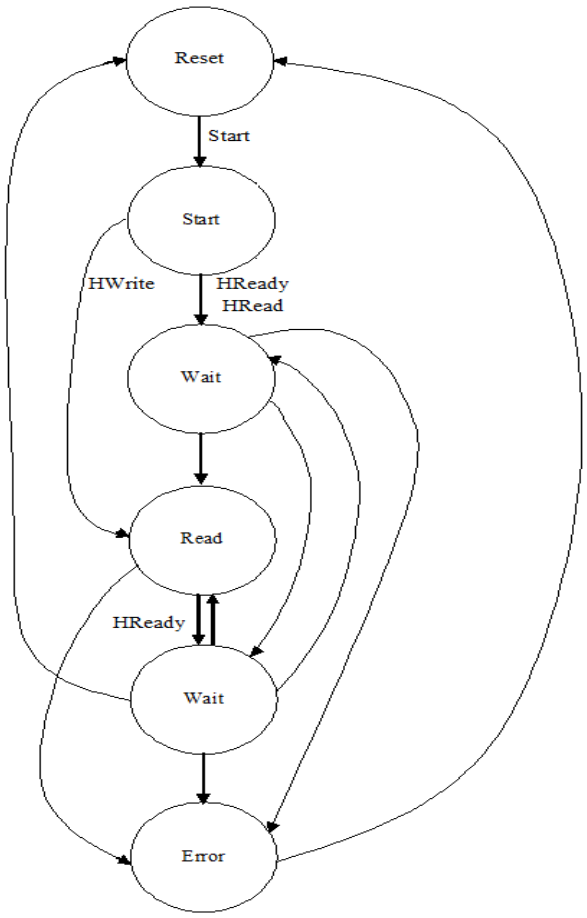


***Figure 7: Slave Interface [2]***

Figure 7 shows the state diagram of slave interface. It is a finite state machine implementation initial condition is reset state which is an idle state when no operation is there. When start signals arrived then this finite state machine (FSM) triggers, depends upon the hready and hwrite signal it decides in which further state it has to move. If hraedy is low then it will be start state only,if hready is one then depends upon hwrite it move to read or write state, if Hwrite is one then state moves to write otherwise it moves to read state. If hready will become low between these states then it moves to wait state, and it will remains in these sate until hready will become one .if any error occurs which is indicated by Hresp then state moves to error state.

**B. Simulation Result of AHB Slave**

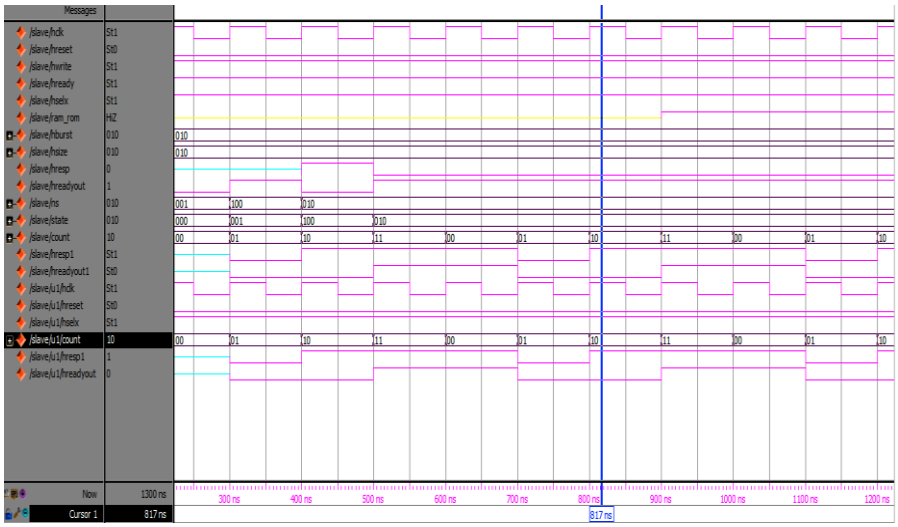

***Figure 10. Simulation result of AHB slave interface***

The above result shows the handshaking between slave and slave interface where slave is directly interacting with master and its response is given by slave, Further the read write operation is forwarded to memory controller by slave interface.

**Memory Controller**

The flow diagram of Memory Controller (MC) is depicted in Figure 3.4. This memory controller is a finite state machine implementation. Initially it is in "idle" condition that is the system "reset state" where no operation is being executed. When the START signal is received then, the FSM is triggered and the CMD state decides the further operation depending upon the instructions given. According to the Read/Write instructions, it moves to RAM READ, RAM WRITE and ROM READ operation.

The main memory controller has to perform two main functions i.e. to tackle the time intervals between two consecutive commands and to keep the record of the rows that are active. This activation of row consumes more time and hence the core memory controller is comprises of look-ahead function where an arbitrator has an authority to decide which instruction will take place after the current instruction is finished. There is a possibility that a row can be activated in advance only if the upcoming command is not going to access the same bank/chip which is being used by the current command.

ISSN 2348 – 8034
Impact Factor- 5.070

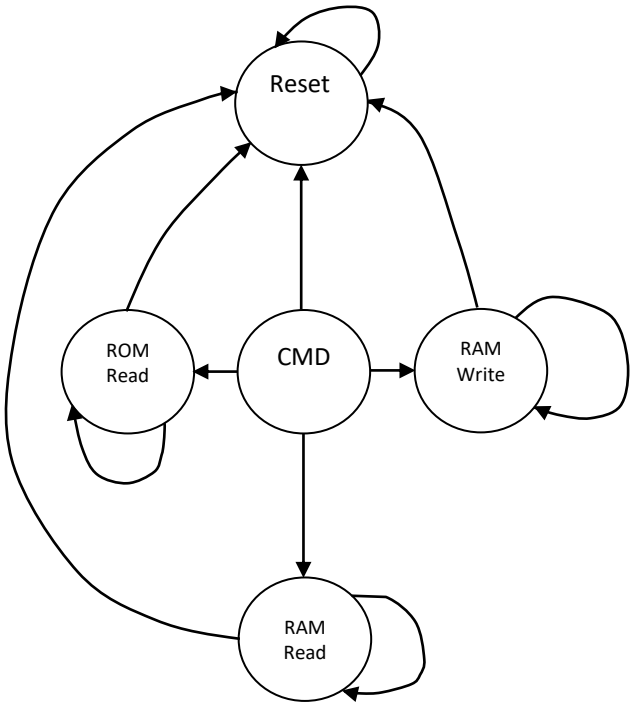


*Figure 4: Memory controller*

**FIFO**

FIFO is a method of processing and retrieving data. In a FIFO system, the first items entered are the first ones to be removed. In other words, the items are removed in the same order they are entered. FIFO is used for data and control buffering. This is done with the goals of assigning different clock cycles (CLK) to slave and memory which in turn gives a reliable communication and also decrease the complexity. Depending upon the READ and WRITE operations data is further communicated through RAM or ROM.

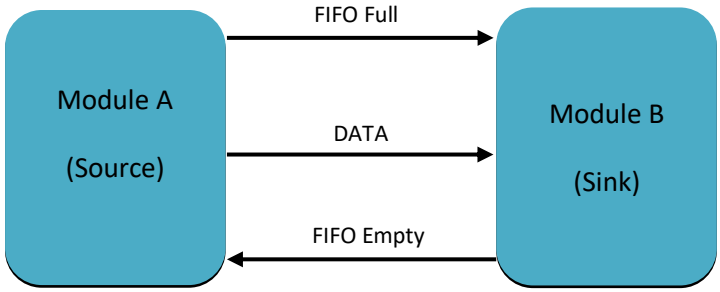


*Figure 5: FIFO*

**RAM**

A RAM is an integrated circuits. Contents of each cell can be read or written and must be able to vary. Unlike the case of ROMs, A reading or writing is carried out in the case of a reading, the bits which constitute the address are received by an address decoder which locates the desired cell. Depending on whether this cell contains a 0 or a 1, the data is routed to the output on the read / write line of a 0 or the read / write line of a 1. For a write operation, the address is also decoded by the address decoder which locates the search cell and wants to write a 0 or a 1, the read / write line of a 0 or the read / write line of a 1 is used to route the data to the desired cell.

**ROM**

A ROM is also an integrated circuit. This implies it is a lot of circuits interconnected to perform a specific. For this situation, the capacity is to store a specific number of data and return it as required. The ROM enclosure therefore comprises the various units (Bits) and the circuits needed to deliver the required information part when and requests the address of the requested information cell.

Simulation Waveforms

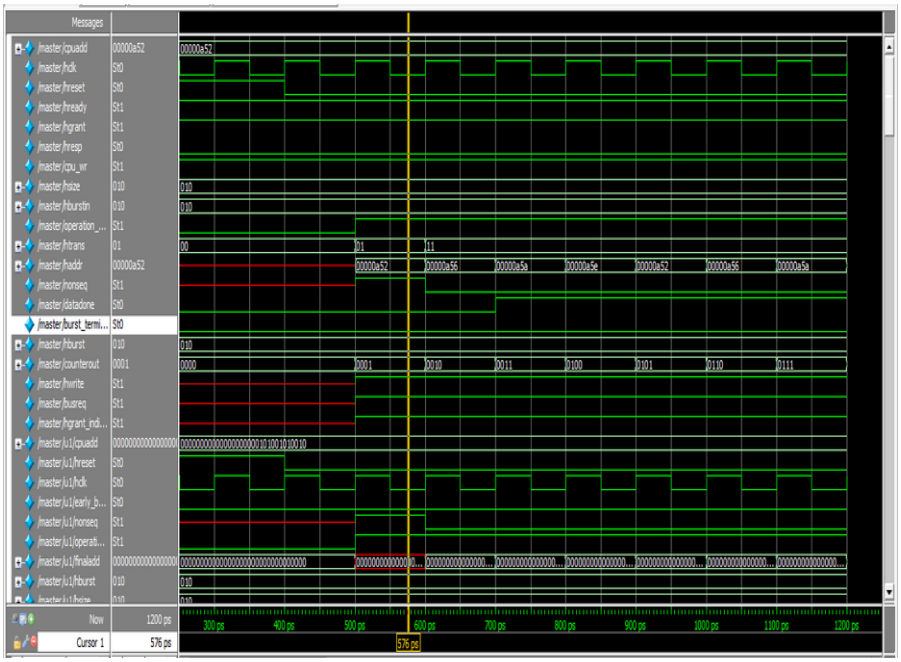

***Figure 6: Simulation waveform of master interface of memory controller***

The above figure 6 shows the simulation results of master. CPU initiates the first address and rest of address is generated by master according to HBURST and HSIZE. Master will generate HTRANS signal as a response. When hready is low then master will not generate it next address. Once hready is high then master starts is next address generation.

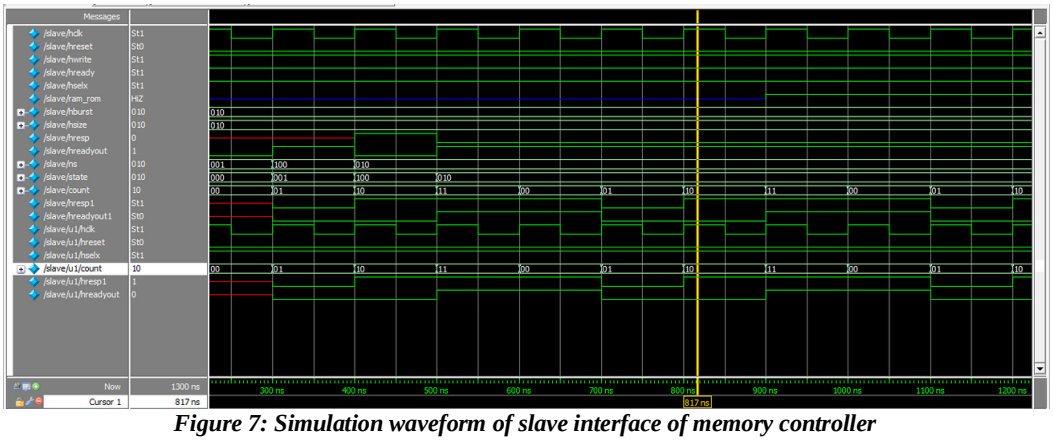

***Figure 7: Simulation waveform of slave interface of memory controller***

The above result shows the handshaking between slave and slave interface where slave is directly interacting with master and its response is given by slave, Further the read write operation is forwarded to memory controller by slave interface.

**RTL Schematic**

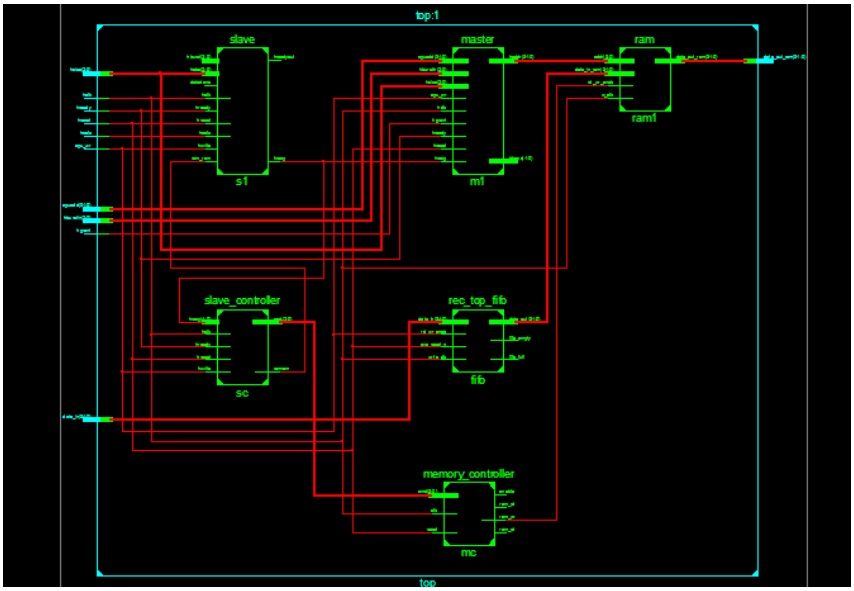


**Figure 8: Top architecture of AHB master-slave memory controller**

**Result comparison with previous research**

*Table 3: Result comparison with previous research*

| Logic | No. of Slices | | No. of Slice Flip Flops (FFs) | | No. of 4-input look-up tables (LUTs) | | No. of bonded input output blocks (IOBs) | | No. of global clocks (GCLKs) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Previous [16] | Proposed | Previous [16] | Proposed | Previous [16] | Proposed | Previous [16] | Proposed | Previous [16] | Proposed |
| Used | 160 | **208** | 111 | **203** | 273 | **235** | 79 | **83** | 3 | **2** |
| Available | 768 | **4800** | 1536 | **203** | 1536 | **2400** | 124 | **102** | 8 | **10** |
| Utilization | 20% | **4%** | 7% | **100%** | 17% | **9%** | 63% | **81%** | 37% | **20%** |

## III. RESULT

In this paper, AHB system i.e AHB bus master, AHB bus slave and memory controller compatible with AHB system with FIFO for buffering the request is designed. The design has been implemented using VHDL for SOC solution. Every implementation is in structural approach hence it's become a well-documented frame. There is separate implementation of slave and slave interface with inter faces the memory controller further. To avoid the handshaking complexity we have used FIFO for slave interface. The design has been developed using VHDL code and synthesized using Xilinx Spartan 3 device (6slx4tqg144-3). The design claims improvement in both area and speed with better results then previous.

ISSN 2348 – 8034
Impact Factor- 5.070