12th International Conference on Recent Innovations in Science, Engineering and Management
Sri Venkateswara College of Engineering & Technology,Etcherla, Srikakulam (ICRISEM'18)
17th February 2018 www.conferenceworld.in ISBN: 978-93-87793-01-9

Developing of AMBHA AHB Based Memory Controller using VHDL

E.Venni[1], K. Govinda Rao[2]

[1]*PG Scholar Dept. Of ECE, SVCET, Srikakulam, AP, (India)*

[2]*Asst. Professor Dept. Of ECE, SVCET, Srikakulam, AP, (India)*

ABSTRACT

In this paper, the design and implementation of an AMBA based Memory controller is proposed. The AMBA based Memory controller gives an ease of integration for sub-frame extraction of various data structures in SOC. AMBA based interaction deals with role specific operation. It is majorly categorized in two dedicated feature i.e. decision (AHB MASTER) and response (AHB SLAVE). The Advanced Microcontroller Bus Architecture (AMBA) specification defines an on chip communications standard for designing high-performance embedded microcontrollers. This paper focuses on how to build an AMBA Advanced High performance Bus (AHB) based memory controller that can work efficiently in multi- master and multi- slave communication model.

Keywords:AHB, AMBA, Memory Controller

I.INTRODUCTION

As increasing numbers of companies adopting the AMBA system, it has rapidly emerged as the de-facto standard for SoC interconnection and IP library development. AMBA enhances a reusable design methodology by defining a common backbone for SoC modules. AHB is a new generation of AMBA bus which is intended to address the requirements of high-performance synthesizable designs. It is a high performance system bus that supports multiple bus masters and provides high bandwidth operation. AMBA AHB implements the features required for high performance, high clock frequency systems including:

- Burst transfers
- Split transactions
- Single-cycle bus master handover
- Single-clock edge operation
- Non-tristate implementation
- Wider data bus configurations (64/128 bits)

Bridging between this higher level of bus and the current ASB/APB can be done efficiently to ensure that any existing designs can be easily integrated. An AMBA AHB design may contain one or more bus masters, typically a system would contain at least the processor and test interface. However, it would also be common for a Direct Memory Access (DMA) or Digital Signal Processor (DSP) to be included as bus masters. The external memory interface, APB Bridge and any internal memory are the most common AHB slaves. Any other

peripheral in the system could also be included as an AHB slave. However, low-bandwidth peripherals typically reside on APB.

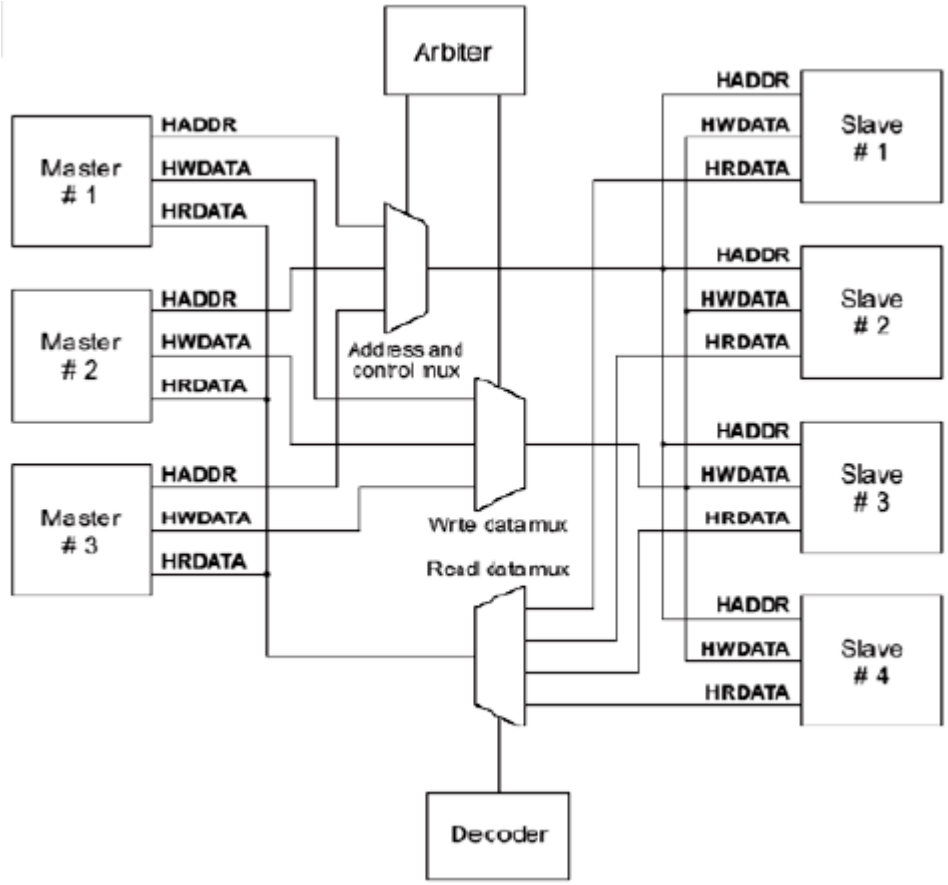

Figure 1: AMBA AHB Block diagram

A typical AMBA AHB system design contains the following components:

- AHB Master
- AHB Slave
- AHB Arbiter
- AHB Decoder

AHB Master- A bus master is able to initiate read and write operations by providing an address and control information. Only one bus master is allowed to actively use the bus at any one time.

AHB Slave - A bus slave responds to a read or write operation within a given address-space range. The bus slave signals back to the active master the success, failure or waiting of the data transfer.

AHB Arbiter - The bus arbiter ensures that only one bus master at a time is allowed to initiate data transfers. Even though the arbitration protocol is fixed, any arbitration algorithm, such as highest priority or fair access can be implemented depending on the application requirements. An AHB would include only one arbiter, although this would be trivial in single bus master systems.

AHB decoder - The AHB decoder is used to decode the address of each transfer and provide a select signal for the slave that is involved in the transfer. A single centralized decoder is required in all AHB implementations.

The AMBA AHB bus protocol is designed to be used with a central multiplexer interconnection scheme. Using this scheme all bus masters drive out the address and control signals indicating the transfer they wish to perform and the arbiter determines which master has its address and control signals routed to all of the slaves. A central decoder is also required to control the read data and response signal multiplexer, which selects the appropriate signals from the slave that is involved in the transfer. Fig.1 illustrates the structure required to implement an AMBA AHB design with three masters and four slaves. Before an AMBA AHB transfer can commence the bus master must be granted access to the bus. This process is started by the master asserting a request signal to the arbiter. Then the arbiter indicates when the master will be granted use of the bus. A granted bus master starts an AMBA AHB transfer by driving the address and control signals. These signals provide information on the address, direction and width of the transfer, as well as an indication if the transfer forms part of a burst.

Two different forms of burst transfers are allowed: incrementing bursts, which do not wrap at address boundaries; and wrapping bursts, which wrap at particular address boundaries. A write data bus is used to move data from the master to a slave, while a read data bus is used to move data from a slave to the master.

II.ARCHITECTURE OF AHB MEMORY CONTROLLER

Figure 2 shows the top level implementation of AHB compliant Memory controller. Data is initiated by master and communicated through slave to memory controller. Initially master generates the data and control signals and further those controls cannot directly communicate with any given generic memory, hence data processed through slave .further data passes through slave interface .we have used FIFO for data and control buffering so that even slave and memory are in different CLK then also our communication is full proof. It reduces the complexity. Further data is communicated through ram or rom depends upon the read and write communication.

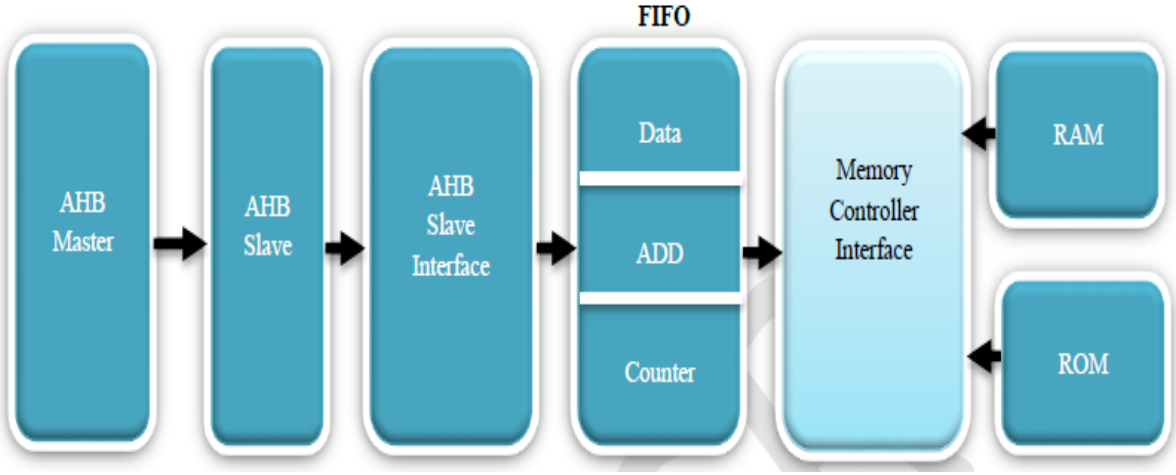

Figure 2. Top architecture of AHB memory controller

A. Memory Controller

The two main tasks for the core memory controller are to handle all the timings between different commands and to keep track of which rows that are currently activated. The activation of rows are time consuming and

therefore the core memory controller has a look ahead functionality where the arbitrator can notify which command that is in turn to be executed after the current one has finished. This makes it possible to activate the row in advance if the next command is not accessing the same bank or chip as the current command.

B. 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.

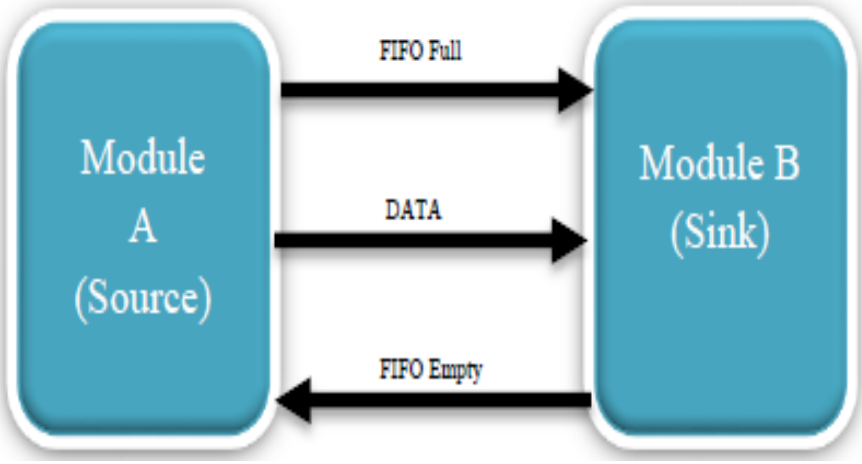

Figure 3. FIFO

In Figure 3, two modules (called the source and the sink) are connected to one another. When data is being passed from module to module, the source is the module that is outputting data. The sink is the module that is receiving that data. Figure 3 also shows three signals between the two:

Data, FIFO Full, and FIFO Empty. Data is the wire that actually passes data from the source to the sink. FIFO Full and FIFO Empty are known as handshaking signals which allow the source and the sink to communicate with regards to when it is time to pass the data. The FIFO Full signal indicates that the FIFO is full, put valid data on the Data line. FIFO Full is what is called a state signal: it is high only when data is valid. If data is not valid on the Data line during a particular cycle, Valid should be low during that cycle. The FIFO Empty signal indicates that the FIFO is Empty to receive new data. FIFO Empty can be asserted as soon as the sink is ready to receive new data. Whenever the sink is not ready to receive new data, Ready should be low. The FIFO Interface handshake ensures that data passes from the source to the sink only when the source has valid data to pass and when the sink is ready to receive that data. In other words, when FIFO Full and FIFO Empty are both high, data on Data will be latched into the sink on the next rising edge.

III.MEMORY SYSTEM

In the arm architecture, instructions are all 32-bits,while instructions are 8-bits in the external ROM and SRAM. Therefore the lowest two addresses of ROM and SRAM are not connected to the external address bus. Additionally, to support byte writing, SRAM needs to be separated as four independent banks or has a byte-write enable signal. The basic memory system architecture is shown in Figure 4.

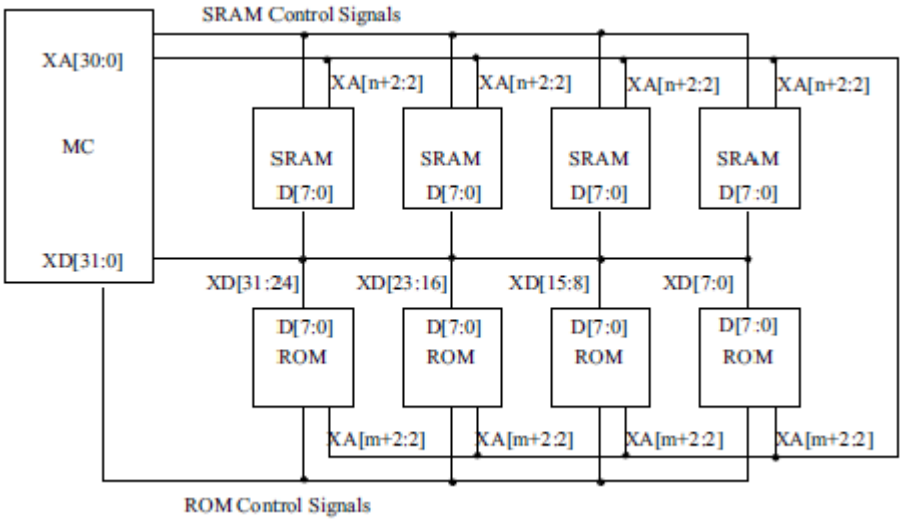

Figure 4: Memory system architecture

A. RAM:

Random-Access memory (**RAM**) is a form of computer data storage. A random-access device allows stored data to be accessed directly in any random order. In contrast, other data storage media such as hard disks, CDs, DVDs and magnetic tape, as well as early primary memory types such as drum memory, read and write data only in a predetermined order, consecutively, because of mechanical design limitations. Therefore the time to access a given data location varies significantly depending on its physical location. Today, random-access memory takes the form of integrated circuits. Strictly speaking, modern types of DRAM are not random access, as data is read in bursts, although the name DRAM / RAM has stuck. However, many types of SRAM, ROM, OTP, and NOR flash are still random access even in a strict sense. RAM is normally associated with volatile types of memory (such as DRAM memory modules), where its stored information is lost if the power is removed.

B. ROM:

Read only memory devices are a special case of memory where, in normal system operation, the memory is read but not changed. Read only memories are non-volatile, that is, stored information is retained when the power is removed.

As in computer terminology read means transferring data instruction from an input source to the computers main memory (or CPU) and write is transferring data/instructions from computers main memory to an output device. Therefore, Read only means data/instruction can be retrieved from the ROM Chip but cannot be modified. The read only memory cell usually consists of a single transistor. The threshold voltage of the transistor determines whether it is a"1" or "0." During the read cycle, a voltage is placed on the gate of the cell. Depending on the programmed threshold voltage, the transistor will or will not drive a current. The sense amplifier will transform this current, or lack of current, into a "1"or "0." Figure 5 shows the basic principle of how a Read Only Memory works.

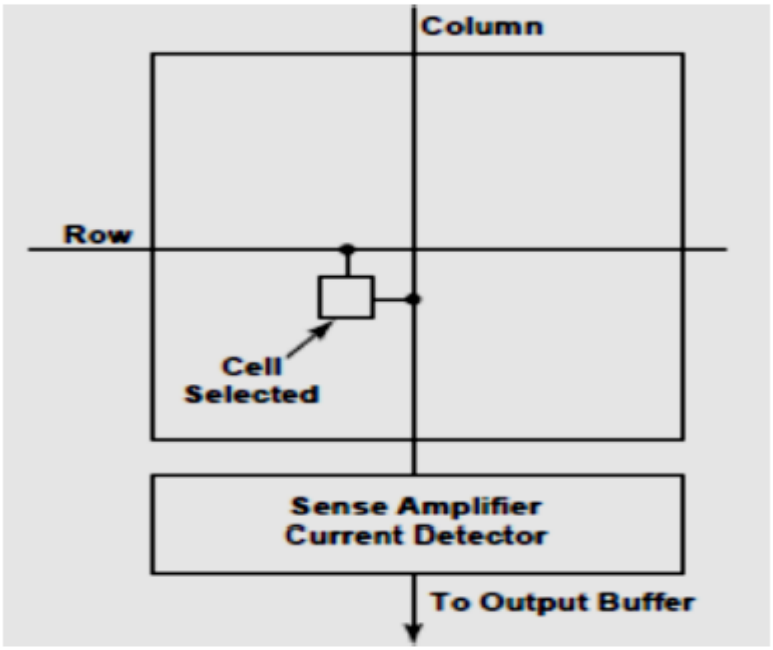

Fig 5. Read only Memory

IV.SIMULATION RESULTS

The simulation waveforms of a simple test code are shown. Figure 6 shows read with zero wait states form the external ROM. The address is registered at rising edge of hclk (AHB bus clock), after which ex_oen (external memory read enable) signal goes high, then read data reach hrdata (AHB read data bus) at falling edge of hclk.

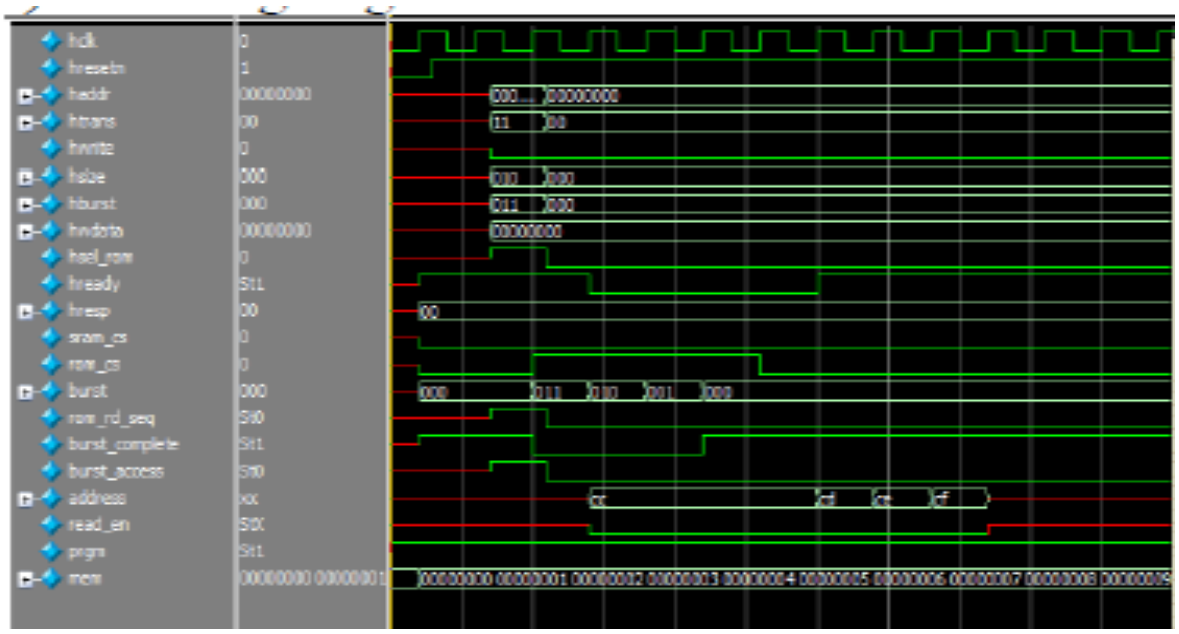

Fig 5. Read with zero wait state from ROM

Write with zero states to the external RAM is shown in Figure 7. A write operation is initiated by hwrite going high. Then the address is send to external memory address bus and ex_wen (external memory write enalbe) signal goes low to enable the data from hwdata (AHB write data bus) stored in the RAM.

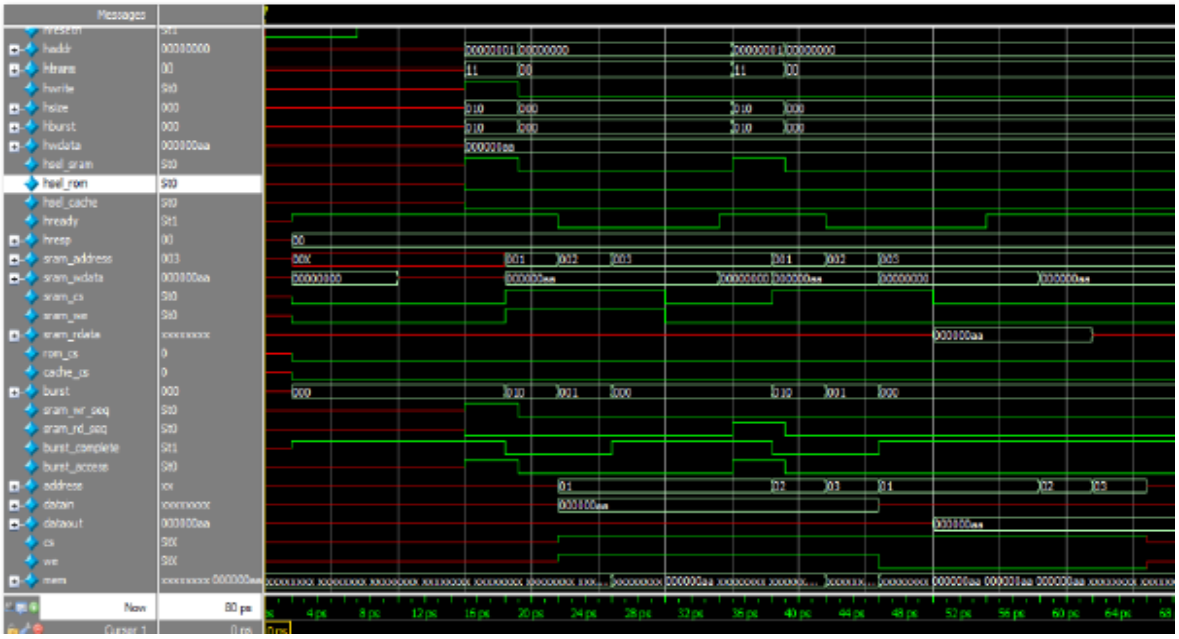

Figure 6. Write with zero wait state to RAM

V. CONCLUSION

This paper presents an approach for the "Building an AMBA AHB Compliant Memory Controller". The memory controller is the part of the system that, well, controls the memory. The memory controller is normally integrated into the system chipset. This project shows how to build an Advanced Microcontroller Bus Architecture (AMBA) compliant MC as an Advanced High-performance Bus (AHB) slave. The MC is designed for system memory control with the main memory consisting of SRAM and ROM.

REFERENCES

[1] "AMBA Specification (Rev2.0)", March 03,2012, ARM Inc.

[2] "PrimeCell AHB SRAM/NOR Memory Controller", Dec 20,2006 Technical Reference Manual, ARM Inc.

[3] "AHB Example AMBA System", Technical Reference Manual, ARM Inc.

[4] "AMBA AXI Protocol Specification", Copyright @ 2003 ARM Limited. All Rights Reserved.

[5]" Simulation and Synthesis Techniques for Asynchronous FIFO Design", SNUG San Jose 2002 Rev 1.2, Clifford E. Cummings, Sunburst Design, Inc.

[6]" Synthesis and Scripting Techniques for Designing Multi- Asynchronous Clock Designs", SNUG San Jose 2001 Rev 1.2, Clifford E. Cummings, Sunburst Design, Inc. Periodicals

[7] Carter, J.; Hsieh,W., High-Performance Computer Architecture, 9-13 Jan 1999. Proceedings. Fifth International Symposium On. Dept. of Computer. Sci., Utah Univ., Salt Lake City, UT.

[8] Clifford E. Cummings, "Synthesis and scripting Techniques for Designing Multi-Asynchronous Clock Designs", jan 20,2005 Sunburst Design, Inc.

[9]. Nithin Joe and Anjali Brite, "A Novel Design Approach to an AMBA AHB Compliant Memory Controller", International Journal of Advances in Engineering & Technology, Vol. 6, Issue 5, pp. 2053-2063, Nov. 2013.

[10] Shilpa Rao and Arati S. Phadke, "Implementation of AMBA compliant Memory Controller on a FPGA", IJETEE, 2013.

[11] Archana C. Sharma1, Prof.Zoonubiya Ali, "Construct High-Speed SDRAM Memory Controller Using Multiple FIFO's for AHB Memory Slave Interface", IJETAE, 2013.

[12] S. Lakshma Reddy, A .Krishna Kumari, "Architecture of An AHB Compliant SDRAM Memory Controller", International Journal of Innovations in Engineering and Technology, 2013.

E. Venni pursuing her M.Tech in the department of Electronics and Communication Engineering (VLSI), Sri Venkateswara College of Engineering & Technology, Etcherla, Srikakulam, A.P., India. Affiliated to Jawaharlal Nehru Technological University, Kakinada. Approved by AICTE, NEW DELHI. She obtained her B.Tech(ECE) from Sarada Institute of Science Technology and Management.

K. Govinda Rao working as Assistant Professor, in the Department of Electronics and Communication Engineering(VLSI), Sri Venkateswara College of Engineering & Technology, Etcherla, Srikakulam. He obtained his M.Tech from Gokul Institute of Technology and Sciences.