ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Design of High-Performance Master/Slave Memory Controller with AHB Architecture

Pemma Ramya
PG Student [VLSI & ES],
Department of ECE,
Rise Krishna Sai Prakasam Group of Inistitutions,
Vallur,Prakasam(Dt),AP, India.

Venkata Rao Param
Assistant Professor,
Department of ECE,
Rise Krishna Sai Prakasam Group of Inistitutions,
Vallur,Prakasam(Dt),AP, India.

Abstract:

The on-chip interconnection system known as advanced microcontroller bus architecture (AMBA) is a well established open specification for the proper management of functional blocks comprising system-on-chips (SOCs). In the subject paper, the design and implementation details of AMBA high-performance bus (AHB) master and slave with memory controller (MC) interface are discussed. A bridge between AHB master and slave with supportive application of MC is also proposed and the resultant efficiency in respect of area overhead and speed is provided. The realization of the control structure is based on the concept of conventional finite state machines (FSMs). The intellectual property (IP) blocks of AHB master and slave are simulated on a Xilinx Spartan3 field programmable gate array (FPGA) .

Keywords:

Advanced microcontroller bus architecture (AMBA) high-performance bus (AHB), field programmable gate array (FPGA), finite-state machines (FSMs), memory controller (MC), system-on-chips(SOCs).

1.INTRODUCTION:

Embedded real-time systems in particular and system-on-chips (SOCs) in general are typically comprised of various types of computational elements. In the realm of processing, these elements perform different tasks in order to realize an overall solution. Consider a set-top box for television (TV) sets as an example. A set-top box must generate inputs for a particular television channel from the received digital satellite signal.

The entire process incorporates several phases. The first is to split the incoming digital signal into its component video and audio data streams. The next phase is to convert the video data stream into its actual television signal. Also, the audio data stream has to be changed into audio signal for the TV set. Besides, an additional job to look after will be to handle the user input or changing the channel when the remote control will be pressed. All these operations have to be completed not only simultaneously but within a certain time frame. The cost of not accomplishing these within the stipulated time frame or deadline will be manifested in the form of blank screen or audible noises. This is obviously unacceptable and hence, it is necessary to always deliver the data within real-time deadlines. The needed computational elements here will be either general-purpose or special-purpose processors (such as signal processors). Nowadays, multitudes of these devices are integrated in the form of an SOC.

A processor needs to interact with other processors, memories or input/output (I/O) devices to complete a task. Currently, bus systems are used to interconnect the intellectual property (IP) blocks. The current research in this field suggests that using instead network-on-chips (NOCs) to interconnect these IP blocks might allow more flexibility than buses . However, to get NOCs acceptable to the industry is still an open question. A major problem in the present context is to deal with large amount of data storage. Usually, higher storage volumes are handled using on chip dynamic memory. The manufacturing process for the memories is different from that for standard logic. The dynamic memories have high data rates and large storage capabilities. Not with standing, the dynamic memories lack compliance in accessing different locations efficiently at random and also necessitate periodic refreshing to keep the storage content intact.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

In order to allow efficient communication between the IP blocks and a shared memory requires the use of a memory controller (MC). The IP blocks, which communicate with the memory, are named as requestors. The MC arbitrates among the requestors and manages the memory accesses. The target of this paper is to design an MC that will provide guarantee in real-time for efficient communication between the IP blocks and memory. In the following, we first present an overview of the MC in general and AMBA system in particular.

2.DISCUSSION ON MEMORY CONTROLLER (MC) AND AMBA SYSTEM:

The performance of microprocessors has been improving rapidly over the years. In contrast, the memory latencies and bandwidths have progressed relatively little. As a consequence, the memory access has been a real bottleneck in the context of improving the system performance. The MC should be efficiently combined with the interconnections. The MC is designed as a master that operates on different protocols and interacts with the memory by sending or receiving data with the help of its slave. The biggest challenge in an SOC design is the integration of processors and memory modules. An increasing numbers of companies have adopted the advanced microcontroller bus architecture (AMBA) system, which has rapidly evolved as the de facto standard for the SOC interconnections and IP library development. The AMBA enhances a reusable design methodology by defining a common backbone for the SOC modules. An AMBA-based microcontroller (Fig.1) typically consists of a high-performance system backbone bus called AMBA high-performance bus (AHB) or advanced system bus (ASB), able to sustain external memory bandwidth, on which the central processing unit (CPU), on-chip memory and other direct memory access (DMA) devices reside.

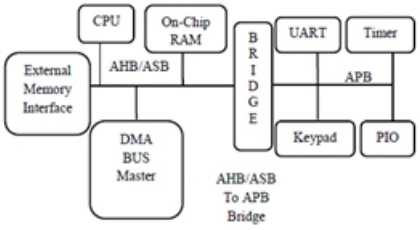

Fig.1: AMBA-based microcontroller

This bus provides a high bandwidth interface between those that are involved in most of the transfers.

The design and implementation of an AMBA-based MC with the aid of AHB-compliant MCs are primarily targeted in this paper The AMBA-based MC furnishes ease of integration for sub-frame extraction of various data structures in SOCs. The AMBA-based interaction deals with role-specific operations. It is categorized into two dedicated features, viz. decision (AHB master) and response (AHB slave).

Moreover, an AHB master enables transfer in burst mode, while AHB master-to-AHB slave supports incrementing and wrapping addressing modes and completes data transfer where the data width of read and write is different by asymmetric asynchronous first-in firstout (FIFO). In the following section, a bridge between an AHB with application of MC is shown; also, the efficiency in terms of area overhead and speed is discussed. The control structure is designed with conventional finite state machines (FSMs). The IP of an AHB master and slave is realized and its interface with the MC is designed and tested.

3. ARCHITECTURE OF AMBA HIGH-PERFORMANCE BUS (AHB) MEMORY CONTROLLER (MC) :

The AHB is a new generation AMBA bus which is intended to address the requirements of high-performance synthesizable designs. It is a system bus that supports multiple bus masters and provides high bandwidth operation. The AHB implements features required for high clock frequency systems including: i) burst transfers; ii) split transactions; iii) single-cycle bus master handover; iv) single-clock edge operations ; v) non-tristate implementations; and vi) wider data bus configurations (64/128 bits).

3.1. Bus interconnection:

The AHB bus protocol is designed to be used with a central multiplexer (MUX) interconnection scheme. Using this scheme, all bus masters drive out the address and control signals, indicating the transfer they wish to perform, while 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 MUX, which selects the appropriate signals from the slave that is involved in the transfer.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

3.2. Basic transfer:

The AHB data transfer consists of two distinct phases:

i) address phase, which lasts only one single cycle;

ii) data phase, which may require several cycles. This is realized by using the HReady signal. The top-level implementation of an AHB-compliant MC is shown in Fig 2.

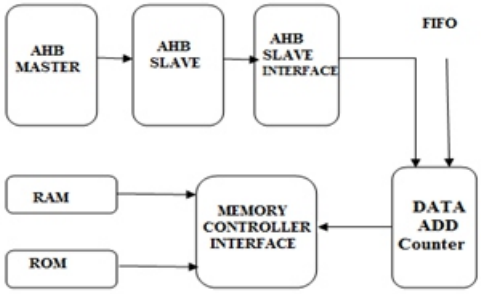

Fig.2: Top architecture of AHB memory controller (MC).

The data are initiated by the master and communicated through the slave to the MC. Initially, the master generates the data and control signals; those controls cannot directly communicate with any given generic memory and hence, data are processed through the slave; further, data pass through the slave interface.

We have used FIFO for data and control buffering so that even the slave and memory are in different clocks (CLKs), ensuring that our communication is guaranteed. It reduces the complexity. Further, data are transmitted through the random-access memory (RAM) or read-only memory (ROM), depending upon the read or write communication.

3.3 AMBA high-performance bus (AHB) master:

An AHB master (Fig.3) has the most complex bus interface in an AMBA system. It contains mainly the address generator and controller. Initially, the first address is given by the CPU; then, it passes to the MUX. Depending on the selection, it sends its data to the address generation block.

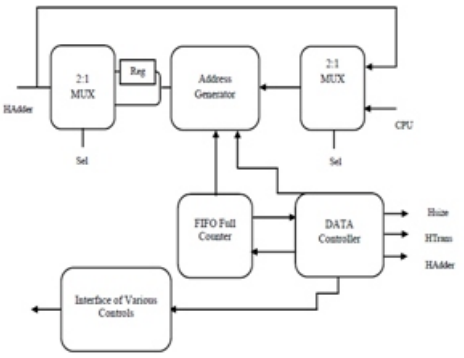

Fig. 3: Top architecture of AHB bus master

The number of addresses generated is given by a counter which decides how many addresses are required depending on the HBurst signal. There is a controller implemented by FSM which takes care of the entire operation of the different blocks. Fig 8 presents the structural implementation of the address generator block, which is a combination of 2:1 MUX and 8:1 MUX. In this architecture, the initial address is given by the CPU and next addresses are generated by the various blocks such as an increment operator (INCR).

3.4. AMBA high-performance bus (AHB) slave:

An AHB slave responds to transfers initiated by the bus masters within the system. The slave uses a select control signal (HSELx) from the decoder to determine when it should respond to a bus transfer. All other signals required for the transfer, viz. the address and control information will be generated by the bus master. Fig 4 shows the state diagram of a slave interface.

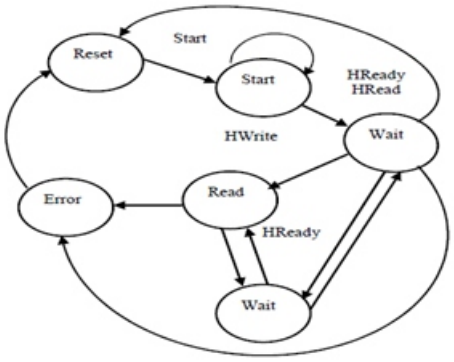

Fig. 4 : State diagram of slave interface.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

It is an FSM implementation; the initial condition is the reset state that is an idle state when no operation is involved. As the start signal arrives, the FSM triggers. Depending on the HRead and HWrite signals, it decides upon to which state it has to move. If HReady is low, the system will stay in the starting state; if HReady is high, based on HWrite, it goes to read or write state. If HWrite is high, the state passes on to the write state; otherwise, if HRead is high, the system enters the read state. If HReady is low, the system transfers to the wait state; it remains in this state until HReady becomes high again. If any error occurs, the system enters the error state.

3.5. Memory controller (MC):

It is also an FSM implementation; the initial condition is reset state which is an idle state when no operation is there. When start signals arrive, the FSM triggers; depending upon the instruction, its operation is decided on by command (CMD) control state. Based on the instruction, it moves to RAM read, RAM write and ROM read operations. Two important tasks for the MC are to handle all the scheduling among the different commands and to keep track of which rows are presently activated. The activation of rows is time consuming and therefore, the MC has a look-ahead functionality where the arbitrator can report which command is going to be executed after the current one has been completed. The arbitrator makes it possible to activate the row in advance if the next command is not retrieving the same bank or chip as the current command.

3.6. First-in first-out (FIFO):

The 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 order they are entered. Fig 5 shows a FIFO buffer consisting of two modules (called source and sink) connected to one another.

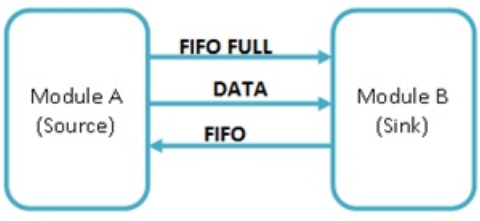

Fig. 5: First-in first-out (FIFO).

When data are 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. In Fig 6, three signals are shown between the two modules A and B: data, FIFO full and FIFO empty. The line marked data represents the wire that actually passes the data from the source to sink. The FIFO full and FIFO empty are known as handshaking signals, which allow the source and sink to communicate when it is time to pass the data. The FIFO full signal indicates that the FIFO is full and puts valid data on the data line. FIFO full is what is called a state signal – it is high only when data are valid. If data are not valid on the data line during a particular cycle, valid should be low during that cycle.

4. SIMULATION RESULTS:

The pin diagrams and register-transfer level (RTL) views of the master MC, and slave controller interface are Shown in Fig.6.

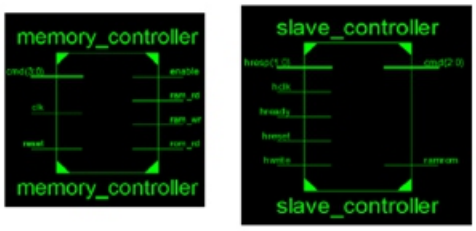

Fig.6: Pin Diagram of Master and slave Controller

The RTL views shown are implemented in Xilinx web pack using Verilog hardware description language (HDL) and simulated in ModelSim. The simulation results of the FIFO block operations: FIFO full or empty, signals high or low, depending on the read or write control signals are shown in Fig 14.

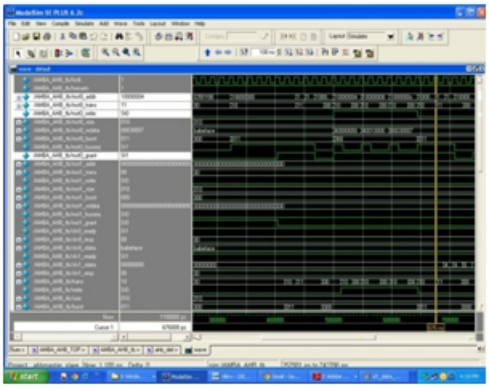

Fig:7: Simulation result of Memory Controller

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

If a write operation is to be executed, then, the FIFO is empty (default) and once the entire data are written, the FIFO turns to full. The MC outputs depend upon CMD control. There are three outputs: RAM_RD, RAM_WR and ROM_RD, as shown in Fig.7.

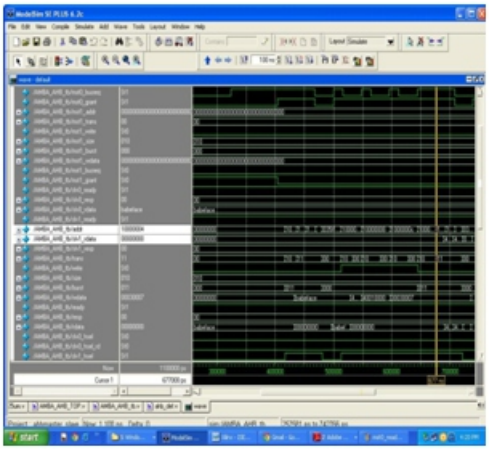

Fig.8: Simulation Result Of First in First Out

Furthermore, the read or write operations are forwarded to the MC by the slave interface. In terms of speed, the minimum period is 6.425ns (maximum frequency of 155.647 MHz). The minimum input arrival time before the clock pulse is 7.269ns. The maximum required output time after the clock pulse is 6.141ns; these results rank grade-5 with respect to speed. The power analysis was generated by Xilinx XPower analyzer. The AMBA MC optimizes the power – the total power dissipated by the AMBA MC is 27.34 mW with a maximum frequency of 155.64 MHz.

5.Conclusions:

An AHB-based MC is designed, realized and tested in the present paper. The implementation was carried out using Verilog HDL for SOC solutions. The design has taken care of balance between area overhead and speed. The master has been implanted and its address generator block is synthesized with a single adder instead of typical 64 adders.

The design has been realized following a modular approach. To avoid handshaking complexity, we have used the FIFO for the MC and slave interface. Even though this increases the latency, it makes design simple and free of bottleneck. The design has been synthesized on Xilinx 13.1 Spartan3 and simulated in ModelSim.

REFERENCES:

1.K. Goossens, O. P. Gangwal, J. Rover, and A. P. Niranjan, “Interconnect and memory organization in SOCs for advanced settop boxes and TV – evolution, analysis, and trends”, in J. Nurmi, H. Tenhunen, J. Isoaho, and A. Jantsch, Editors, Interconnect-Centric Design for Advanced SoC and NoC, Chapter 15, pp. 399–423,Kluwer, 2004.

2.Y. Hu and B. Yang, “Building an AMBA AHB compliant memory controller”, Proceedings of the Third International Conference on Measuring Technology and Mechatronics Automation, Vol. 01, 2011, pp. 658–661.

3.AMBA Specification (Rev 2.0), ARM Inc., 1999.

4.AHB Example – AMBA System, Technical Reference Manual, ARM Inc.,1999.