Design of an efficient finite state machine for the implementation of AMBA AHB Master

[1]*Bhaumik Vaidya* [2]*Anupam devani*

[1 2] **Department of Electronics Engineering,**
Gujarat Technological University,
Gandhinagar, Gujarat, India.

[1]*vaidya.bhaumik@gmail.com*
[2]*anupam_devani@yahoo.com*

ABSTRACT : Due to Moore's law more and more amount of logic is being placed onto a single silicon die and it is driving the development of highly integrated SoC designs. So this high computational power must be matched with interconnect fabric which can handle it. There are many interconnect buses that are widely used in the industry like AMBA, Wishbone, CoreConnect, Avalon etc. AMBA is most proffered among all of them because it has a hierarchy of buses with AHB (Advance high performance bus) can be connected to high performance peripherals and APB (Advance Peripheral Bus) that can be connected to low performance peripherals. AHB can be used in high performance and high bandwidth system because of its high performance features like burst transfer and split transaction. In AMBA system, AHB master is the component that initiates the read and writes transaction so in this paper efficient finite state machine design is describe for implementation of AHB master in any hardware description language like Verilog or VHDL.

KEY WORDS: *AMBA, AHB master, System on Chip*

1. Introduction

The Advanced Microcontroller Bus Architecture (AMBA) is a protocol that is used as an open standard, on-chip interconnect specification for the connection and management of functional blocks in a system-on-chip (SoC)

[1]. AMBA assists the progress of right-first-time development of multiprocessor designs with large number of controllers & peripherals

[2]. The Advanced Microcontroller Bus Architecture (AMBA) has the ability to re-use designs and here it means it has the ability to re-use IP. IP re-use in today's technology is an important factor in reducing the development costs and timescales for System-on-chip (SoC)

[3]. AMBA is a standard interface specification that makes sure of the compatibility between IP components provided by different design teams or vendors.

The world wide reception of AMBA specifications all over the semiconductor industry has driven a comprehensive market in third party IP products and tools to support the development of AMBA based systems.

There are three different buses defined within the AMBA specification - Advanced High Performance Bus (AHB), Advanced System Bus (ASB), and Advance Peripheral Bus (APB).

This paper mainly deals with AMBA AHB and particularly AHB master. It is divided in to three sections. First section introduces AHB system, its features and some signals associated with AHB master. Second section has the description of state machine designed for AHB master. Third section contains the results of implementing the state machine in the Verilog hardware description language.

1. Advance high performance bus [AHB] :

AHB implements many features that are 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).

An AMBA AHB design may contain one or more bus masters, typically a system would contain at least the processor as an AHB master. 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 like SRAM and ROM, APB Bridge and any internal memory are the common AHB slaves. Any other peripheral in the system could also be included as an AHB slave.

An AHB bus master can initiate read and write operations by providing an address and control information. Only one bus master is allowed to transfer the data on the bus at any one time. This decision is made by an Arbiter in the system. A bus slave responds to a operation within a given address-space range. The bus slave signals back to the active master the success or the failure of the data transfer. It can also insert the wait state if data is not available.

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
• 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.
Every transfer consists of:

• An address and control cycle
• One or more cycles for the data.

The address cannot be extended and therefore all slaves must sample the address during this time. The data, however, can be extended using the **HREADY** signal. When LOW this signal causes wait states to be inserted into the transfer and allows extra time for the slave to provide or sample data. During a transfer the slave shows the status using the response signals, hresp. OK response indicates transfer has completed successfully. Error respond indicate error in the transfer. Retry and split responses are also supported on the advance slaves.

The Figure 1 shows the typical multi master and multi slave AMBA AHB based system. As can be seen that arbiter is used to control the multiplexors that select which master can transfer address and data while decoder is used to select an appropriate slave according to the address generated by the master.

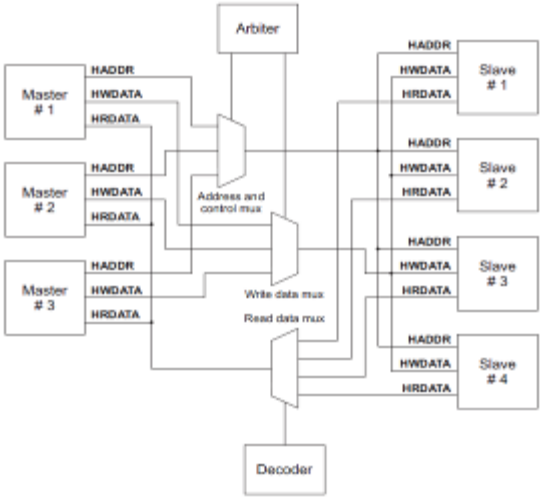

Figure 1AMBA AHB System

2. Finite State Machine for AHB master :

As discussed earlier , Main component in the AMBA system is the master that can initiate the read or write transfer to any slave so it is imperative that master is properly designed for an AHB system to work. In addition AHB master implementation has to support advance features like burst transfers defined in the specification. So here in Figure 2 finite state machine is shown which support the features in the specification.

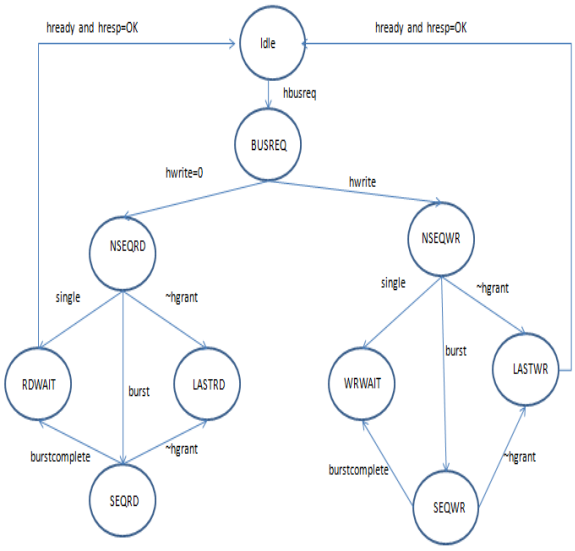

Figure 2 FSM of AHB master

The following section contains the brief description of every state in the diagram.

1) Idle :

This is the default state in the state machine. When master has no data to transfer or it has just finished the transaction then it will stay in this state. When it wants to perform another transaction then it will have to request for a bus from the arbiter so it will move in to BUSREQ state.

2) BUSREQ :

The AHB master will wait in this state until the bus has been granted by the AHB arbiter. Once the bus is granted it will move to NSEQRD state if it wants to perform a read transfer or move to NSEQWR state for write transfer.

3) NSEQRD :

As AMBA AHB has the pipelined architecture , so address and control information is send first in this cycle and based on the control information like burst size (hburst) and size of transfer (hsize) , next state

and address are decided. If it is a single burst then next state will be RDWAIT for getting data for this address and if it is incrementing burst transfer then the next state will be SEQRD. If grant to the master is lost due to some reason then it will move to LASTRD state for latching data and freeing bus.
In this state, it is a single transfer or a first transfer of a burst so htrans value will be '10' which shows non sequential transfer.

4) SEQRD :

As this is a sequential transfer, the value for htrans will be '11' which shows sequential transfer. The next address will be generated based on the size of the transfer. If the size is byte, then previous address will be incremented by 1 , If the size is half word then previous address will be incremented by 2 and if it is word then previous address will be incremented by 4. Based on the value of hburst , burst complete signal is asserted to show the last transfer on the burst and state is transferred to RDWAIT to latch the final data of the burst. If grant is lost in the middle of the burst then state will be transited to LASTRD and then master will re arbitrate for the bus for completing the remaining burst.

5) RDWAIT :

The AHB master will be in this state when it is a single transfer or the last transfer of the burst. The final data is latched in this state and if hready and OK response is asserted by the slave it shows that the read transaction has finished successfully on the bus and master can move to idle state.

6) LASTRD :

The AHB master will be in this state when it lost the grant of the bus in the middle of the transaction so in this state it will latch the data of the address transferred in the previous cycle due to pipelining and then re-arbitrate for the bus for completing the remaining burst.

7) NSEQWR :

Here hwrite signal will be 1 to indicate the write transaction from the master. If it is a single beat write transfer then next state will be WRWAIT for transferring data for this address and if it is incrementing burst transfer then the next state will be SEQWR. If grant to the master is lost due to some reason then it will move to LASTWR state for writing the last data and freeing bus.
In this state, it is a single transfer or a first transfer of a burst so htrans value will be '10' which shows non sequential transfer.

8) SEQWR :

As this is a sequential transfer, the value for htrans will be '11' which shows sequential transfer. The next address will be generated based on the size of the transfer. If the size is byte, then previous address will be incremented by 1 , If the size is half word then previous address will be incremented by 2 and if it is word then previous address will be incremented by 4. Based on the value of hburst , burst complete signal is asserted to show the last transfer on the burst and state is transferred to WRWAIT to write the final data of the burst. If grant is lost in the middle of the burst then state will be transited to LASTWR and then master will re arbitrate for the bus for completing the remaining burst.

9) WRWAIT :

The AHB master will be in this state when it is a single transfer or the last transfer of the burst. The final data is written in this state and if hready and OK response is asserted by the slave it shows that the write transaction has finished successfully on the bus and master can move to the idle state.

10) LASTRD :

The AHB master will be in this state when it lost the grant of the bus in the middle of the transaction so in this state it will write the data of the address then re-arbitrate for the bus for completing the remaining burst.

If in the middle of the transaction, error, split or retry response comes from the slave then master will stop the transaction and move to the idle state.

3. Result

After designing the finite state machine, any hardware description language is used to implement it and check it functionality for correctness. In this Paper, the state machine is implemented in Verilog and VCS simulation tool is used to simulate the design and generate the waveforms

Write Transfer

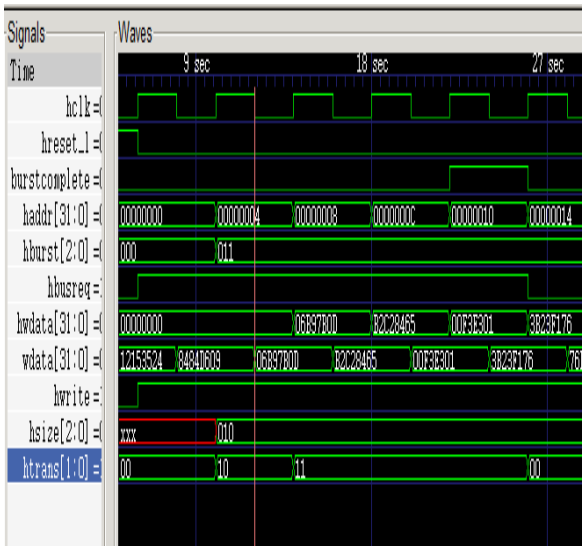

Figure 3 AHB master 4 beat write burst waveforms

The Figure 3 above indicates 4 beat write transfer with size of the transfer is 32 bit. As we can see, master starts the transaction by asserting hbusreq signal for requesting the bus. When request is granted master start the transaction by giving address and the control information. The value of hburst is 11 that indicate it is a 4 beat transfer and hwrite is 1 indicating write transfer. Htrans value is 10 for first transfer indicating non sequential transfer and after that it is 11 indicating sequential transfer. Hsize is 10 indicating word transfer so as you can see address is incremented by 4 after every cycle. Burstcomplete signal at the end is asserted to show the completion of the burst.

Read Transfer

The Figure 4 indicates the 4 beat read burst transfer. Size is byte so address will be incremented by 1 after each transfer.

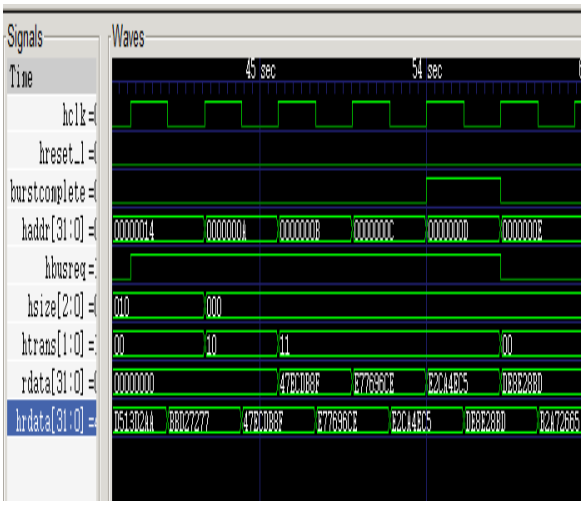

Figure 4 AHB master 4 beat read burst waveforms

4. Conclusion:

Over the years AMBA has continued to provide state-of-the-art solutions for SoC interconnects and AMBA master is the integral part of the system so designing it with efficient technique will lead to area and timing efficient design of the full system on chip. So in this paper efficient finite sate machine design is proposed for AMBA AHB master. Synthesizing it using Synopsys design compiler shows that it can work on high frequency designs.

5. Acknowledgement:

This work was greatly supported by Gujarat Technological University and Seer Akademi by providing state of the art tools like VCS and Design Compiler from Synopsys

6. References:

[1] AMBA specification, version 2.0
[2] Hu Yueli, Yang Ben "Building an AMBA AHB compliant Memory Controller "
[3]"AHB Example AMBA System", Technical Reference Manual, ARM Inc.
[4] Priyanka Gandhani, Charu Patel "Moving from AMBA AHB to AXI Bus in SoC Designs: A Comparative Study", 2011