Design and Implementation of Efficient FSM for AMBA AHB Protocol
Sagar Latti
PG Scholar
Department of VLSI & Embedded System
Visvesvaraya Technological University-Belagavi Karnataka-591018, India

Abstract— Nowadays industries are more focused on the development of System On-Chip (SOC) which integrates all the electronic components on single IC with reusable IP cores. As per the De Moore’s law the integration of transistors goes on increasing, hence it is required to provide efficient lossless on chip communication for these reusable IP cores. The AMBA protocol from ARM Ltd. is designed to overcome this issue. The AMBA AHB protocol provides the high bandwidth data bus for the IP cores on SOC devices. In this paper the design and synthesis of AMBA AHB protocol using HDL Verilog in Xilinx ISE tools are explained.

Key words: Advanced High-Performance Bus, AHB Master, AHB Slave, System on Chip

I. INTRODUCTION

Advanced Microcontroller Bus Architecture (AMBA) is developed by ARM Ltd. which provides high bandwidth lossless data communication between the functional blocks in system on chip (SOC).

1) AMBA assists the progress of right-first-time development of multiprocessor designs with large number of control peripherals.
2) This Advanced Microcontroller Bus Architecture has the ability to reuse the IP design, this reuse of IC reducing the development costs and timescales for SOC’s.
3) AMBA is a standard specification that makes sure of the compatibility between IP components provided by different design teams and different vendors.

There are three different bus architectures are defined with the AMBA specification- Advanced System Bus (ASB), Advanced Peripheral Bus (APB), Advanced High-speed Bus (AHB). In this paper the design of AHB Master and Slaves are explained. The entire paper is divided into three sections. First section introduces AHB system its architecture and features, Second section explains the AHB master and slave blocks, Third sections shows simulations results.

Fig. 1: Typical AMBA system

A. Advanced System Bus (ASB)

The AMBA ASB defines the high performance bus that can be utilized in the design of high performance embedded microprocessor, The AMBA ASB also supports the well-organized on-chip memory connections, processors, external memories with low power microcell functions.

The pipelined structures can be designed with ASB which provides capability of supporting multiple bus masters. The fundamental of bus operation is as follows.

- The Arbiter determines which bus should be granted to the master.
- Master initiates the transfer once it is granted
- Then the slave is selected by the decoder by using the high order address line.
- The data transfer takes place between the master and the slave after the slave provides the response to the master.

B. Advanced Peripheral Bus (APB)

The AMBA APB can be used in the peripherals requiring low power and low performance. The APB is used to reduce the interface complexity and to use minimum power. The APB can be used in several designs easily with the following advantages.

- Can be used as peripheral bus.
- Can be bridged to system bus.
- It is provides synchronous transfer.
- Non-pipelined and supports single pipelined.

C. Advanced High Performance Bus (AHB)

AHB is new generation of AMBA bus intended to address the requirements of high-performance synthesizable designs. AHB provides high bandwidth operations and also supports multiple bus masters.

The AHB AHB has the following features required for the high performance bus operations.

- Burst transfers.
- Split transfers
- Single cycle bus master handover.
- Single-clock edge operation.
- Wider data bus configurations.

Bridging between higher level of bus and current ASB/APB can be done efficiently to ensure that any design can be easily integrated, The other advantage is that it supports multiple masters like processors, DSP’s, DMA’s.

A typical AMBA AHB system contains the following components as shown in Fig 2.

- AHB Master: Master initiates read and write operations when it is granted access to the bus from the arbiter AHB can support total of 16 masters. These masters are typically processors, digital signal processors, Direct memory access (DMA’s).
- AHB Slaves: Slaves are the components which are selected by masters to perform read and write operations, these slaves are selected based on the higher order address information by decoders. Master can access one slave in each operation.

All rights reserved by www.ijsrd.com | 70
AHB arbiter: Arbiter is decides the bus access to the masters. It takes the response from the masters and grants the bus access to the masters based on the round robin based arbitration. It also provides the bus access to the masters based on the higher priorities to the masters.

II. OVERVIEW OF AMBA OPERATION

The AHB transfer operation is initiated by master by asserting a request signal to the arbiter, the arbiter takes the signal from the masters and based on the priorities or round robin based arbitration it gives bus access to the masters. Once master gets the grant signal form the arbiter it sends the address and control information to the central decoder. The central decoder selects the slaves based on the higher order address information received from the master. Once the slave is selected master sends the data to perform the read and write operation. A write data bus is used to write the data from master to slaves and the read data bus is used to read the data from slave to master. Hence every transfer consists of two cycles.

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

Fig. 2: AMBA AHB System with Multiplexer Interconnection

A. AMBA AHB Master Interface

The AMBA AHB Interface diagram is as shown in the Fig 3. AHB Bus Master is the most complex system in the AHB protocol. It takes the input signals HREADY and HRESP form the slaves and HGRANT from the arbiter and externals HRESET and HCLK signals and 32-bit of write data and read data signals. The address phase can’t be extended during the address phase slaves also sense the address only one time but the data phase can be extended several times by HREADY and HRESP signal form the slave. The data phase can be extended by asserting the HREADY signal to LOW also when HRESP is ERROR then the data phase is extended until HRESP gets OKAY which indicates the completion of data phase.

B. Finite State Machine for AMBA AHB Master Interface

Fig. 4: FSM for AHB Master

The FSM for AHB Master is as Shown in the Fig 4. The Master can have the following states:

1) IDLE

Initially the Master will be in the IDLE states during this state no data transfer takes place. Buses are free and masters are not accessing any buses during this state. The Master goes to BUSREQ state from the IDLE state when it wants to perform the data transfer. The IDLE state can be changed to BUSREQ state by asserting the busreq signal. The Master waits in this state until it gets grant request from the Arbiter.

2) BUSREQ: (Bus Request)

The master jumps to BUSREQ state form IDLE state when it sends the bus request to arbiter, the mater waits in this state until it gets the grant access from the arbiter. When the hgrant signal is asserted to high then the master jumps to NSEQD or NSEQWR state based on the hwrite signal.

3) NSEQWR: (Non Sequential Write)

As discussed earlier the address and data information are sent in two phases, when the master grants the bus access it jumps to NSEQWR state when the hwrite is ‘1’ to perform the write operations. In this state the address is given by the master to the slaves and then the data is transferred. In this stage if data transfer is single then the master jumps to WRWAIT state until it receives the next data inputs, if the sequence of data is to transferred then the master jumps to SEQWR. If the master is lost the bus access during the transfer it jumps to LASTWR state freeing the bus.

4) SEQWR: (Sequential Write)

This state involves the sequential transfer. The sequential transfer takes place when the htrans is ‘10’ during this state the address is incremented based on the hburst signal until
the sequential transfer finishes. Based on the size of the data, the address is incremented until the transfer completed. If the size is in byte the address is incremented by ‘1’ if the size is in half word the address is incremented by 2. If it is word then address is incremented by 4. This transfer is also called as burst transfer based on the value of hburst the SEQRD state is completed and then it jumps to WRWAIT state for the final data of the burst. If the bus access is lost during this state the master jumps to LASTWR state, then the burst transfer completes when the bus access is issued by the arbiter.

5) **WRWAIT (Write Wait)**
Mater will be in this state, when the transfer is single or it is the last transfer of the burst. The final data is written when the hready is high and hresp from the slave is OKAY signal

6) **LASTWR (Last Write)**
The master jumps to this state when hgrant goes low and it loose the bus access. This state the data of the address in the previous cycle is written then the master will continue the execution until the burst is completed. The arbiter decides the grant access to the bus in the next stage based on the master priorities assigned to it, If in the middle the slave produces the ERROR, SPLIT, RETRY responses then master will stop the transaction and move to the IDLE state.

7) **NSEQRD (Non Sequential Read)**
The Master is in this state when the hwrite signal is ‘0’., In this state the master is doing the read operation, In this state the master gives the address and control information to the central decoder which selects the slaves and the data is being read once the data read is completed then the slave asserts the OKAY signal then the master jumps to RDWAIT state for the final data to be read. If the data is to be read sequentially then htrans is made “10” and the master jumps to SEQRD based on the burst information the data is read and finally state changes to WEWAIT state for the final data to be read. If the grant access is lost during this state the master jumps to LASTRD state.

8) **SEQRD (Sequential Read)**
The master is transferred to this state if the sequence of the data is to be read by the master. This state is achieved by making htrans “10” and also asserting the control information size and hburst. If the size is in byte the address is incremented by ‘1’. if the size is in half word the address is incremented by 4. If it is word then address is incremented by 4. This transfer is also called as burst transfer based on the value of hburst the SEQRD state is completed and then it jumps to RDWAIT state for the final data of the burst. If the bus access is lost during this state the master jumps to LASTRD state, then the burst transfer completes when the bus access is issued by the arbiter.

9) **RDWAIT (Read Wait)**
The master will be in this state if the master wants to read the single data or the last data of the burst transfer this state can be achieved by asserting the htrans=“01” and making hready as ‘high’ and receiving the OKAY response from the Slave.

10) **LASTRD (Last Read)**
The master will be in this state when the grant access is asserted to low by the arbiter. In this state the address and control information are used in the next cycle to read the data from the address where the master lost the bus grant access during the burst transfer.

C. **AMBA AHB Slave Interface**

---

![AMBA AHB Slave Interface](image)

The AMBA AHB interface is as shown in the fig 5. The Slave is selected by master based on the address information provided by the master. The slaves are SRAM, DMA, FIFO which can be able to store the data information provided by the masters.

The slaves are selected by hselx signal and it produces the outputs HREADY, HRESP and HSPLIT these outputs form the slaves are given to the master to determine the next cycle operations.

1) **HSPLIT**
For slaves that can issue a SPLIT transfer response, bus deadlock is prevented by ensuring that the slave can withstand a request from every master in the system, up to a maximum of 16. The slave does not need to store the address and control information for every transfer, it simply needs to record the fact that a transfer request has been made and a SPLIT response issued. Eventually all masters will be at a low priority and the slave can then work through the requests in an orderly manner, indicating to the arbiter which request it is servicing, thus ensuring that all requests are eventually serviced.

2) **HREADY**
HREADY signal is the output signal form the slaves which is used to put the wait condition during the data transfer phase in the masters when HREADY is asserted to low the masters waits for the data to be read or write in to the slaves

3) **HRESP**
This is 2-bit signal form the slaves which is used as acknowledgment signal for the master to indicate whether the data transfer is completed by asserting the OKAY response to the master If the response is ERROR, SPLIT, RETRY then master takes one more next cycle to finish the data transfer until the slave produces the OKAY response indicating the completion of the data transfer.

III. **SIMULATION RESULTS**
The simulations and synthesis are carried out using Xilinx ISE tools v14.7 using the Verilog hardware descriptive language. The FSM design for the master is verified for the proper lossless communication. The write and read operations are shown in the figure.
Design and Implementation of Efficient FSM for AMBA AHB Protocol

The AHB Master Simulation results are shown in the fig 6. When hwrite='1' the write transfer takes place, when hwrite='0', the read operations are performed. When htrans='10' the Master performs Burst transfer. The 8 Beat Burst operations are performed, during the transfer operations if hready goes low then master waits data transfer until it becomes high. The split operations are shown in the fig 7. When hresp=01, from the slaves then the Master performs the splits operations the data and address transfers takes in the next cycle when hresp and hready goes high again. The master put into wait mode if the bus access is lost and in the next cycle the operations are completed when the arbiter grants the bus access. When htrans='00' then the Master goes to IDLE state.

IV. CONCLUSION

The AHB master interface is designed using the finite state machines in Verilog hardware description language and the design is simulated with the help of Xilinx 14.7v and the synthesis of the design is done in Xilinx ISE design tool. The completed AMBA AHB system is then checked for proper lossless communication between master and slave interface.

REFERENCES