AXI and OCP protocol Interface for Sytem on Chip

Ms. Monica Damor[1], Mr Gardas Naresh Kumar[2], Mr. Santosh Jagtap[3]

[1] Research Scholar, GTU PG School,Gujarat,India
[2] Course Co-Ordinator, CDAC ACTS, Maharashtra, India
[3] Wipro Ltd, Pune, Maharashtra, India

ABSTRACT

Many Controllers and processors a uses different protocols as number of protocols are available nowadays. A single protocol has its separate characteristics, functionality, Signals, commands. When a controller and processor needs to communicate with each other it uses protocols for data transfer. Now these protocols need a interface between them, by using it they can establish a proper communication. Here, two well known protocols, AXI-Advanced eXtensible Protocol and OCP-Open Core Protocol. The Interface is created using System Verilog Hardware Description Language.

Keywords: - *AXI Protocol, OCP Protocol, Interface, System Verilog HDL, System on chip.*

1. Introduction

ICs which are known as Integrated Circuits are small chips, whereas System on chip (SoC), which gives integration of each and every parts of a system on a single chip. In place of a system that combines number of chips and components on a circuit board, the SoC integrates all required circuits into one chip. Design size and complexity of system has increased, intellectual property has become an essential factor for the system on chip. With the increase in process decency and regularity, more number of protocols and IPs are used in a System on Chip substantially which has changed the flow of the design by making on chip bus a essential element in the design. As result of this reusability and configurability has become a dominant factor in chip protocol. When the number of integrated IP cores substantially increases, the communication between IP cores also increase and it becomes quite frequent that two or more master IPs would request data from different slaves at the same time. In shared bus architecture only one bus transaction can be supported at a time which often cannot provide efficient communication. two bus protocols have been proposed, to resolve this Complexity,. The First One is the Advanced extensible Interface protocol (AXI) developed by ARM and the other is Open core Protocol (OCP) which is developed by OCP-IP(Open core protocol- International Partnership).

The ARM Advanced Microcontroller Bus Architecture (AMBA) is an open standard, on chip interconnect. ARM introduced the first specification of AMBA in 1996. The first version of AXI was introduced by AMBA 3 Specification. Open core protocol is a common standard for intellectual property (IP) core interfaces. It is an openly licensed, core centric protocol. It is socket based bus architecture which was developed by OCP- International Partnership (IP). The bridge allows the interface between the high performance AXI bus and Open Core Protocol.

2. AXI (ADVANCED EXTENSIBLE INTERFACE).

AXI is the Advanced Extensible Interface, of AMBA interface specification, is created for high speed in performance, increased clock frequency system design and suitable for high speed interconnection. It is a parallel bus protocol and used as a point to point connection between a single master and slave interface. AXI is suitable for low latency and high bandwidth designs, and also provides high frequency operation without any complexity. The main advantage of AXI can be used for AHB and APB interfaces which are the older versions of AXI interface of AMBA Bus.

Some of the characteristics of AXI are:

- Data and address/control phase are separate..
- It supports asynchronous data transfer with help of byte strobes.
- Uses transactions as burst-based with where the start address issued only.
- For low cost Direct memory access it use independent read and write data channels.
- It supports multiple outstanding addresses and out of order transactions.
- Registers can be easily added for timing closure .

1.1. AXI (ADVANCED EXTENSIBLE INTERFACE).

AXI is the Advanced Extensible Interface, of AMBA interface specification, is created for high speed in performance, increased clock frequency system design and suitable for high speed interconnection. It is a parallel bus protocol and used as a point to point connection between a single master and slave interface. AXI is suitable for low latency and high bandwidth designs, and also provides high frequency operation without any complexity. The main advantage of AXI can be used for AHB and APB interfaces which are the older versions of AXI interface of AMBA Bus.

1.2. AXI ARCITECTURE

An address channel transfer control information that give details of the nature of the data to be transferred. The data moves from master to slave by using: write data channel, which transmit data from the master side to the slave. While doing write transaction, the slave work with the write response channel to give signal of transfer complete to the master. A read data channel is used to transmit data- slave to master.

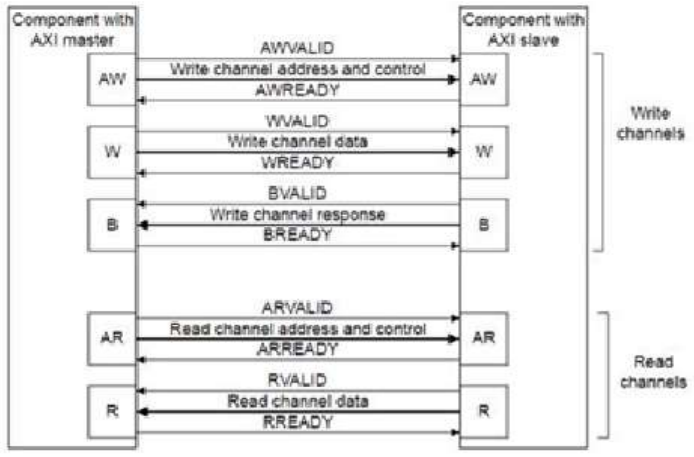


Fig.1. AXI write and read channels

3. OCP (OPEN CORE PROTOCOL).

The Open Core Protocol (OCP) work as a high speed performance, separate bus interface among IP cores that reduce the risk in design, design time, and manufacturing costs for SoC designs. Here an IP core can Possibly be a simple external core, a high speed microprocessor, or a wrapped on-chip bus.

3.1.Theory of Operation

- Point-to-Point Synchronous Interface

  To simplify timing analysis, physical design, and general comprehension, the OCP is composed of unidirectional signals driven with respect to, and sampled by the rising edge of the OCP clock. The OCP is fully synchronous and contains no multicycle timing paths with respect to the OCP clock. All signals other than the clock signal are strictly point to point.

- Bus Independence

  A core utilizing the OCP can be interfaced to any bus. A test of any bus independent interface is to connect a master to a slave without an intervening on-chip bus. This test not only drives the specification towards a fully symmetric interface but helps to clarify other issues. For instance, device selection techniques vary greatly among on-chip buses. Some use address decoders. Others generate independent device select signals (analogous to a board level chip select). This complexity should be hidden from IP cores, especially since in the directly-connected case there is no decode/selection logic. OCP-compliant slaves receive device selection information integrated into the basic command field.

- Pipelining

  The OCP allows pipelining of transfers. To support this feature, the return of read data and the provision of write data may be delayed after the presentation of the associated request.

- Address/Data

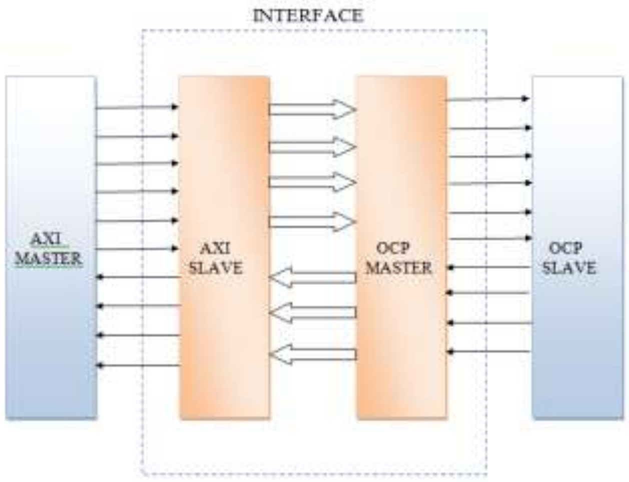
  Wide widths, characteristic of shared on-chip address and data buses, make tuning the OCP address and data widths essential for area-efficient implementation. Only those address bits that are significant to the IP core should cross the OCP to the slave. The OCP address space is flat and composed of 8-bit bytes.

- Commands

  There are two basic commands, Read and Write and five command extensions. The WriteNonPost and Broadcast commands have semantics that are similar to write command. The other command extensions, ReadExclusive, ReadLinked and WriteConditioanl, are used for synchronization between system initiator.

4. AXI TO OCP INTERFACE

In 2003, ARM launched AMBA 3, which included the Advanced eXtensible Interface (AXI) while in 2001 the Open Core Protocol (OCP-IP) organization started work on what became the OCP specification. In both cases, these standards purportedly de-coupled the interface choice from the interconnect topology, which was an improvement over traditional busses.



As shown in the figure, the main Inteface logic is to connect one master and one slave. In the case of AXI Master and OCP slave Interface, the bridge logic have AXI Slave mapped with OCP master using local interface.

Basically the bus bridge contains the AXI master and OCP slave at synchronized clock. So that AXI slave is getting data or information from AXI master and AXI slave is managed to pass those information to OCP master with port mapping in such a way that OCP master will accept the information.The AXI slave and OCP Master are connected to each-other with local interface. Local interface is used because if there is need to change one protocol from the bridge that can be done without complexity, one can simply remove the connection and add the other protocol at that place. This connection is very efficient. The signals from AXI Slave are send through FIFO to OCP Master.

4.1. AXI Slave Interface

This are the basic signals required for transaction in AXI module.

Table 1. Slave Interface write channel signals

Signal Name	Description	Direction
ACLK	Global Clock	Input
ARESETn	Global reset	Input
AWID	Write Address ID	Input
AWADDR	Write Address	Input
AWLEN	Burst length	Input
AWSIZE	Burst Size	Input
AWBURST	Burst Type	Input
AWLOCK	Lock Type	Input
AWCACHE	Cache Type	Input
AWVALID	Write Address valid	Input
AWREADY	Write Address ready	Output
WDATA	Write Data	Input
WSTRB	Write Strobe	Input

WLAST	Write Last	Input
WVALID	Write Valid	Input
WREADY	Write ready	Output
BID	Response ID	Output
BRESP	Write response	Output
BVALID	Write response valid	Output
BREADY	Response ready	Input

4.2. Asynchronous FIFO

An asynchronous FIFO refers to a FIFO design where data values are written to a FIFO buffer from one clock domain and the data values are read from the same FIFO buffer from another clock domain, where the two clock domains are asynchronous to each other.

To facilitate static timing analysis of the FIFO design, the design has been partitioned into the following modules with the following functionality and clock domains:

FIFO Memory: Both write and read clock domains are accessed by FIFO memory buffer. Here the buffer is like a dual-port RAM

FIFO Pointer: This is the FIFO Pointer module which has read and write pointers respectively empty and full are defined.

Sync read to write: This is a synchronizer module that is used to synchronize the read pointer into the write-clock domain.. This module only contains input outputs that are synchronized to the write clock. No other logic is included in this module.

Sync write to read: This is a synchronizer module that is used to synchronize the write pointer into the read-clock domain. This module only contains input and outputs that are synchronized to the read clock. No other logic is included in this module.

Compare read and write: This module will compare FIFO pointers and check if the FIFO is full or empty.

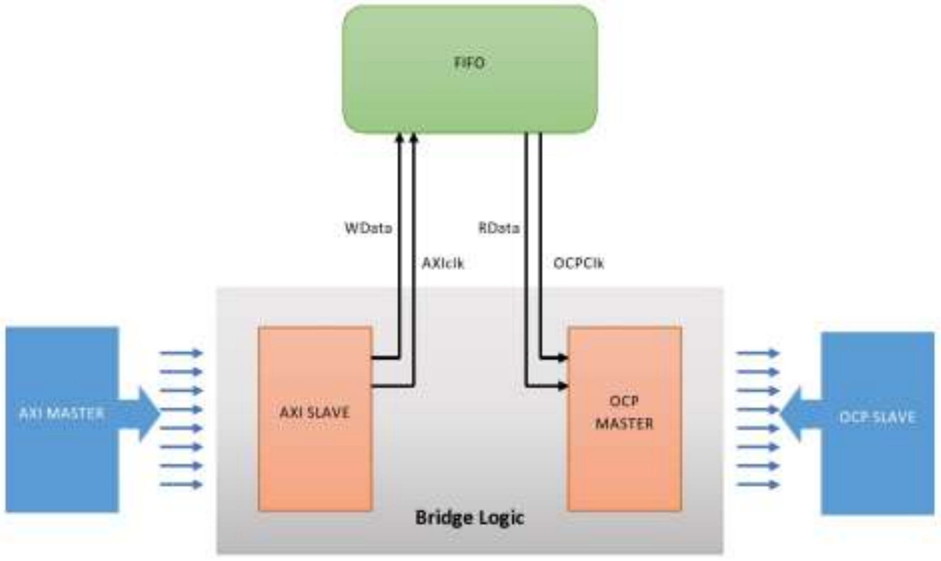
FIFO top module: This is the top-level module that includes all clock domains. The top module is only used as a module to instantiate all of the other FIFO modules which are used in the design.

4.3. OCP Interface

Signal Name	Description	Driver
Clk	Clock	Master
Reset	Reset	Master
MAddr	Transfer Address	Master
MCmd	Transfer Command	Master
MData	Write Data	Master
SCmdAccept	Response Accept Signal	Slave

SData	Read Data	Slave
SDataAccept	Slave Accepts write Data	Slave
SResp	Transfer Response	Slave

5. Top Level Module



The Detailed Diagram of the Interface is shown above. The Bridge logic has AXI Slave, FIFO, and OCP Master. Here AXI Slave and OCP Master are connected to FIFO through Local interface. The Data transaction will take place from AXI master to AXI Slave through Address, Data and Response channels. The Data from the AXI Slave is send to FIFO memory using local interface And OCP Master will read that data from FIFO memory and send it to OCP Slave. This is the basic functionality of the bridge. Signals used for AXI and OCP are shown in the figure in AXI module and OCP module.

6. Simulation Results

Basically the bus Interface contains the AXI master and OCP slave and a FIFO. The bridge will such that AXI slave is getting data or information from AXI master and AXI slave is managed to pass those information to OCP master with port mapping in such a way that OCP master will accept the information.

The Simulation result of each module and the interface between them is shown here:

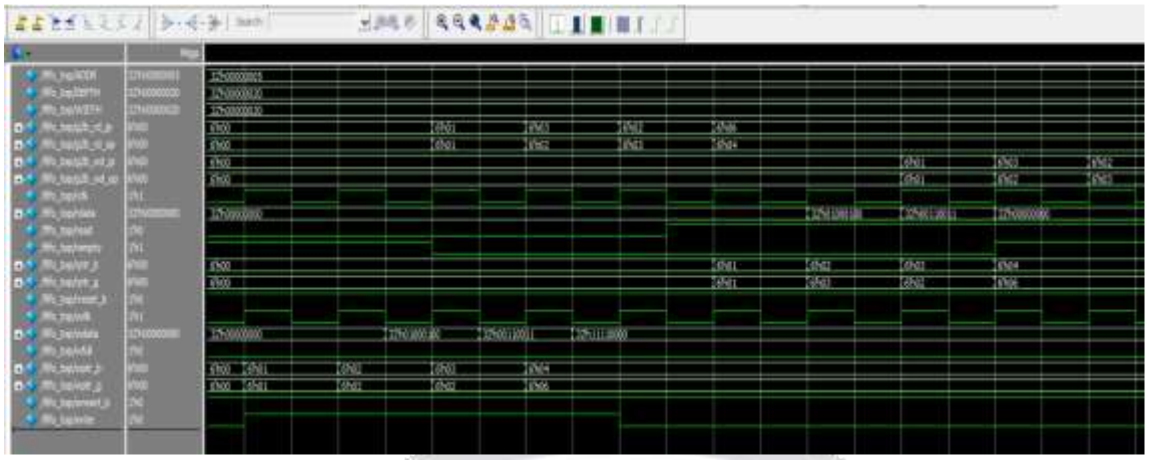


Fig. Simulation of FIFO top module

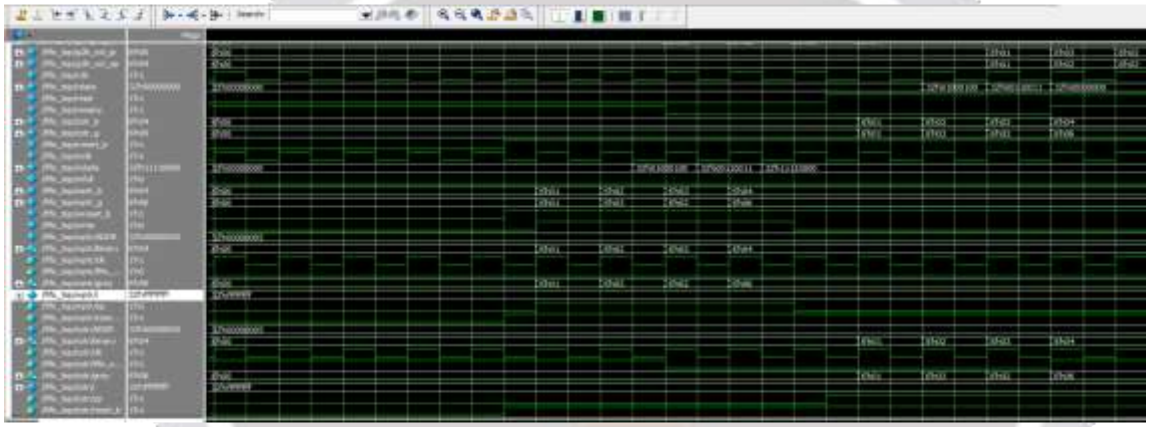


Fig. Simulation of FIFO with read and write pointers

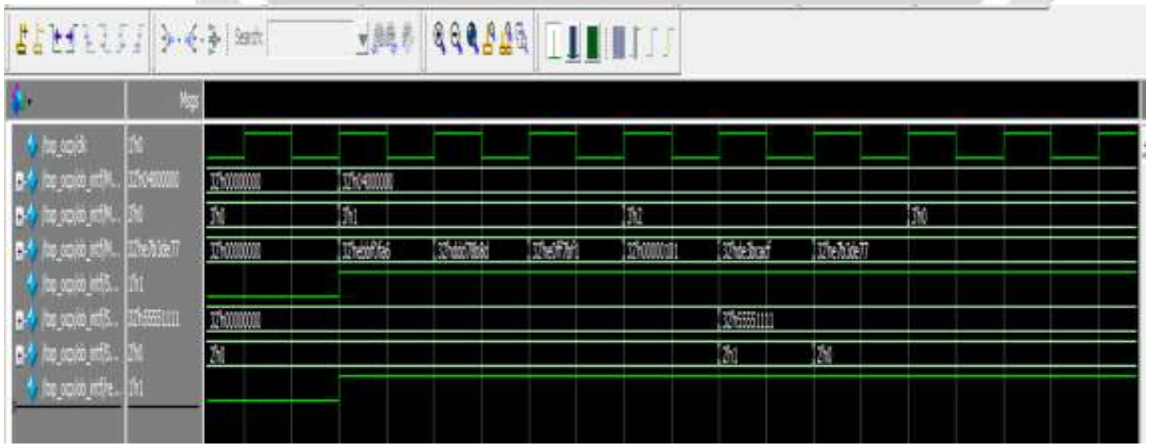


Fig. Simulation of OCP module

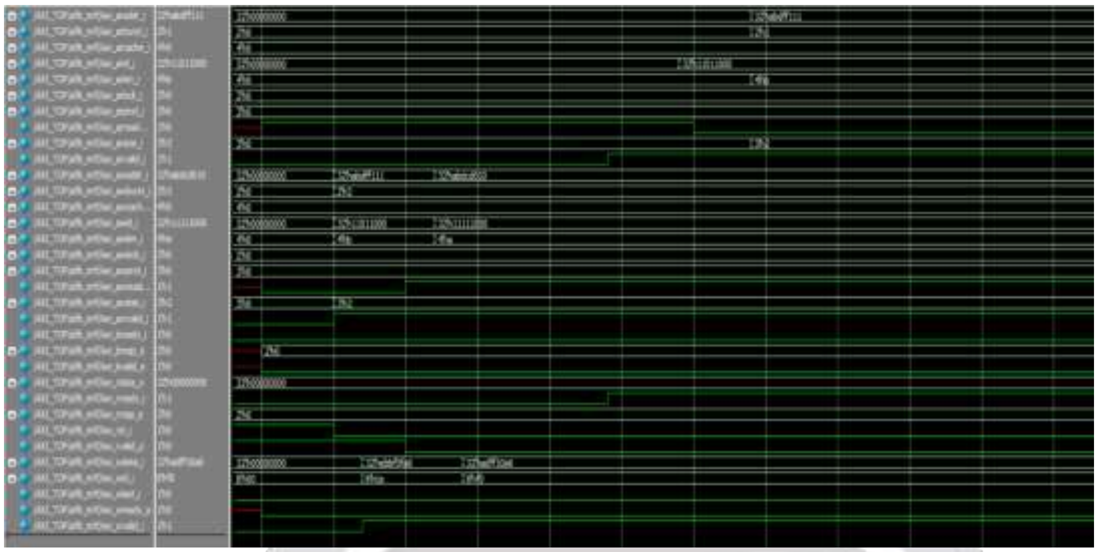


Fig. Simulation of AXI module

7. Conclusion

Interface between two protocols AXI and OCP is created. The simulation result for AXI Slave, Asynchronous FIFO and OCP is created and the result are as expected.AXI to OCP connect Used in System on Chip where processor is AXI convention based and controller piece are OCP convention based.

8. REFERENCES

[1] Abhinav Tiwari and Jagdish Nagar. An efficient axi read and write channel for memory interface in system-on-chip.

[2] Shaila S Math, RB Manjula, SS Manvi, and Paul Kaunds. Data transactions on system-on-chip bus using axi4 protocol. In Recent Advancements in Electrical, Elec tronics and Control Engineering (ICONRAEeCE), 2011 International Conference on, pages 423{427. IEEE, 2011.

[3] Open Core Protocol Specification. Release 2.2, ocp-ip, 2006.

[4] AXI Xilinx. Axi reference guide.

[5] Elina Rajan Varughese et al. Implementation of extended open core protocol interface memory system using verilog hdl. In Green Computing, Communication and Conservation of Energy (ICGCE), 2013 International Conference on, pages 130{135. IEEE, 2013.

[6] SR Pradeep and C Laxmi. Design and verification environment for amba axi protocol for soc integration.