International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 9, November- 2012

Design And Implementation Of APB Bridge Based On AMBA AXI 4.0

Santhi Priya Sarekokku[1], K.Rajasekhar[2]
[1](*Electronics and Communication Department, ASR College/ JNTUniversity, India)*
[2](*Electronics and Communication Department, ASR College/ JNTUniversity, India)*

Abstract

With the rapid development of technology in deep Submicron, the scale of integrated circuit(IC)grows increasingly. Intellectual property (IP) reuse has been taken popularly. This design method is widely applied in the system on chip (SoC) field. The on chip bus design is most cared in IP-reused based SoC design. Several bus standards were proposed. Among those standards, the advanced microcontroller bus architecture (AMBA) is the most favored. ARM introduced the Advanced Microcontroller Bus Architecture (AMBA) 4.0 specifications in March 2010, which includes Advanced eXtensiable Interface (AXI) 4.0.This is very high speed bus. AMBA bus protocol has become the de facto standard SoC bus. That means more and more existing IPs must be able to communicate with AMBA 4.0 bus. Based on AMBA 4.0 bus, we designed an Intellectual Property (IP) core of Advanced Peripheral Bus (APB) bridge, which translates the AXI4.0 transactions into APB 4.0 transactions. The bridge provides an interface between the high-performance AXI bus and low-power APB domain.

Keywords- *AMBA, APB, AXI, IP,SoC;*

1. INTRODUCTION

Integrated circuits have entered the era of System-on-a-Chip (SoC), which refers to integrating all components of a computer or other electronic system into a single chip. It may contain digital, analog, mixed-signal, and often radio-frequency functions – all on a single chip substrate. With the increasing design size, IP is an inevitable choice for SoC design. And the widespread use of all kinds of IPs has changed the nature of the design flow, making On-Chip Buses (OCB) essential to the design. Of all OCBs existing in the market, the AMBA bus system is widely used as the de facto standard SoC bus.

On March 8, 2010, ARM announced availability of the AMBA 4.0 specifications. As the de facto standard SoC bus, AMBA bus is widely used in the high-performance SoC designs. The AMBA specification defines on-chip communication standard for designing high-performance embedded microcontrollers. The AMBA 4.0 specification defines five buses/interfaces:

- Advanced eXtensible Interface (AXI)
- Advanced High-performance Bus(AHB)
- Advanced System Bus (ASB)
- Advanced Peripheral Bus (APB)
- Advanced Trace Bus (ATB)

AXI, the next generation of AMBA interface defined in the AMBA 4.0 specification, is targeted at high performance, high clock frequency system designs and includes features which make it very suitable for high speed sub-micrometer interconnect.

- separate address/control and data phases
- support for unaligned data transfers using byte strobes
- burst based transactions with only start address issued
- issuing of multiple outstanding addresses
- easy addition of register stages to provide timing closure

2. TOP VIEW

2.1Block Diagram

The APB bridge provides an interface between the high-performance AXI domain and the low-power APB domain. It appears as a slave on AXI bus but as a master on APB that can access up to sixteen slave peripherals. Read and write transfers on the AXI bus are converted into corresponding transfers on the APB. The AXI4.0 to APB bridge block diagram is shown in Fig. 1. AXI supports Out of order processing and multiple outstanding operations, these shows AXI as a efficient protocol.AXI is operating at high speeds and APB consists of slow devices there should be FIFOs on the Bridge to store the data and addresses .AXI devices can communicate among themselves and to bridge but APB devices cannot communicate themselves.APB devices communicate with bridge only. Always master initiate the transaction. At a time a master can access only one slave. A slave can be connected to one master at a time.

The AXI protocol is burst-based, and the master begins each burst by driving transfer control information and the address of the first byte in the transfer. As the burst transaction progresses, it is the responsibility of the slave to calculate the addresses of subsequent transfers in the burst.

ISSN: 2278-0181
Vol. 1 Issue 9, November- 2012

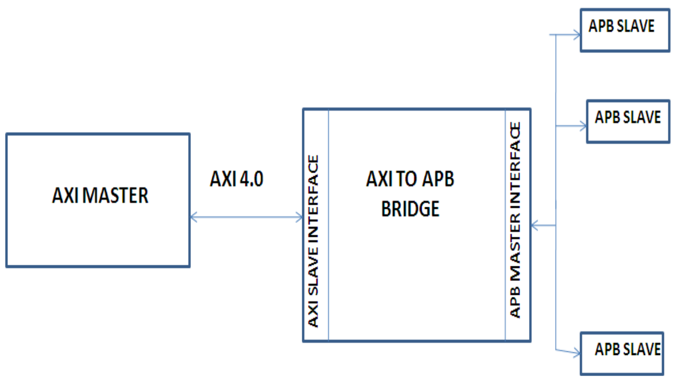

Figure.1. Block diagram

2.2 Signal Connections

Signal connections are shown in the Fig.2**.** The bridge uses:

- AMBA AXI signals as described in the AMBA AXI 4.0 protocol specification.
- AMBA APB signals as described in the AMBA APB4.0 protocol specification.

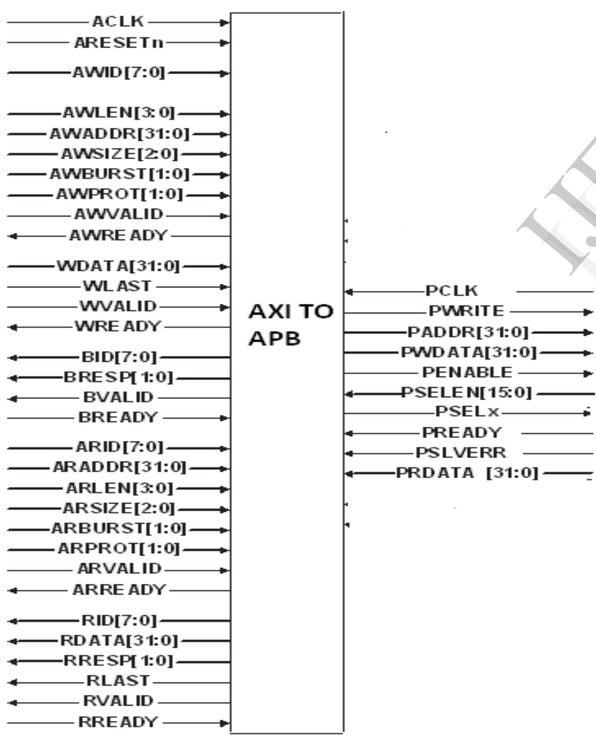

Figure.2. Signal Connections

2.3 AXI Handshake Mechanism

In AXI 4.0specification each channel has VALID and READY signals for handshaking. The source asserts VALID when the control information or data is available. The destination asserts READY when it can accept the control information or data. Transfer occurs only when both the VALID and READY are asserted. Fig.3 shows all possible cases of VALID/READY handshaking. When source asserts VALID, the corresponding control information or data must also be available at the same time. A transfer takes place at the Positive edge of the clock. Therefore, the source needs a register input to sample the READY signal. In the same way, the destination needs a register input to sample the VALID signal. As combinational circuits need one cycle to pull low VALID/READY and sample the VALID/READY again at another cycle. When they sample the VALID/READY again at the same cycle there should be another transfer ,which is an error.AXI protocol is suitable register input and combinational output circuit.

The APB bridge buffers address, control and data from AXI4.0, drives the APB peripherals and returns data and response signal to the AXI4.0. It decodes the address using an internal address map to select the peripheral. The bridge is designed to operate when the APB and phase. For every AXI channel, invalid commands are not forwarded and an error response generate. That is once an peripheral accessed does not exist, the APB bridge will generate DECERR as response through the channel (read or write). And if the target peripheral exists, but asserts PSLVERR, it will give a SLVERR response. In any transaction:

- the VALID signal of one AXI component must not be dependent on the READY signal of the other component in the transaction
- the READY signal can wait for assertion of the VALID signal

to prevent a deadlock situation.

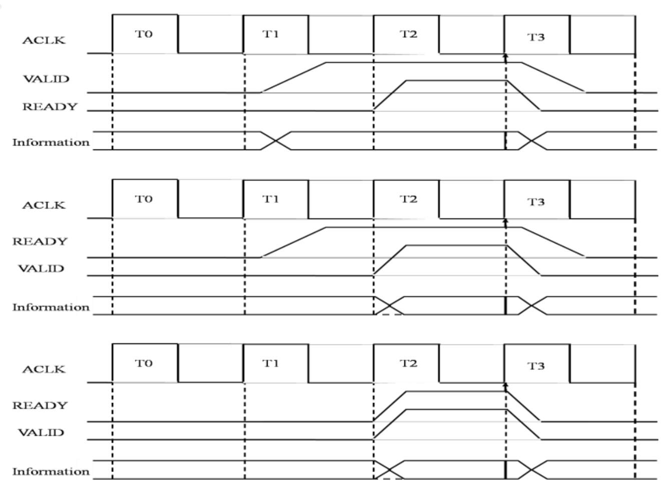

Figure.3. Hand Shaking Mechanism

2.4 Read Write transaction

There are no ordering restrictions between read and write transactions and they are allowed to complete in any order. If a master requires a given relationship between read and write transaction then it must ensure that the earlier transaction is complete before issuing the later transaction. In the case of reads the earlier transaction can be considered

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 9, November- 2012

complete when the last read data is returned to the master. In the case of writes the transaction can only be considered complete when the write response is received by the master, it is not acceptable to consider the write transaction complete when all the write data is sent. For address regions occupied by peripherals this typically means waiting for earlier transactions to complete when switching between read and write transactions that require an ordering restriction. For memory regions, it is possible for a master to implement an address check against outstanding transactions, to determine if a new transaction could be to the same, or overlapping, address region.

3. CLOCK DOMAIN CROSSING

A clock domain crossing (CDC) is when a signal crosses from one clock domain into another. If a signal does not assert long enough and is not registered, it may appear asynchronous on the incoming clock boundary. Metastability happens when signal changes within the setup/hold time window. Synchronizing a signal that crosses into a higher clocked domain can be accomplished by registering the signal through a flip-flop that is clocked by the source domain thus holding the signal long enough to be detected by the higher clock domain destination. Synchronizing a signal traversing into a slower clock domain is more cumbersome. This requires a register in each clock domain with a form of feedback from the destination domain to the source domain, indicating that the signal was detected.

3.1 Metastability

Metastability cannot be avoided, but a solution for handling the metastable signal enables proper functioning of the design. The metastability occurrences cab be predicted by using the mean time between failures (MTBF). Designers can use special metastable hardened flops for increasing the MTBF.In the AXI4 to APB bridge, we use synchronizer block designs for communicate between the AXI and APB clock domain. Two flip-flop synchronizer is used, by this arrangement metastability can be avoided.

4. FINITE STATE MACHINE

A finite state machine is a mathematical abstraction sometimes used to design digital logic or computer programs. It is a behavior model composed of a finite number of states, transitions between those states, and actions, similar to a flow graph in which one can inspect the way logic runs when certain conditions are met. The state transition diagram is a picture of our state machine model. Figure.4 is the state transition diagram of our FSM.

The state machine operates through the following states:

- IDLE. This is the default state of the FSM.
- WRITE_SETUP. When a write transfer request is asserted, the FSM moves into the WRITE_SETUP state
- READ_SETUP. When a read transfer request is asserted, the FSM moves into the READ_SETUP state.
- WRITE_ACCESS.The enable signal, PENABLE, is asserted in the WRITE_ACCESS state.
- READ_ACCESS. The enable signal, PENABLE, is asserted in the READ_ACCESS state.
- WRITE_WAIT. When the AXI write response channel is not ready for receiving signal BRESP, then stay in WRITE_WAIT state.
- READ_WAIT. When the AXI read data channel is not ready for receiving signal RRESP, then stay in READ_WAIT state.

States READ_WAIT and WRITE_WAIT are wait states are added during transfers between the APB and AXI interface. According AXI specification, the read address channel, write address channel and write data channel are completely independent. Each channel has a set of forward signals and a feedback signal for handshaking .A read and writes requests may be issued simultaneously from AXI4.0 to APB, the AXI to APB bridge will give more priority to the read request than to the write request.

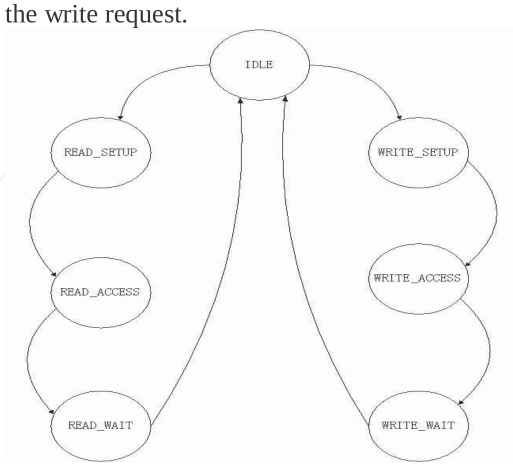

Figure.4. State transition diagram

5. SIMULATION & IMPLEMENTATION

The timing diagram shown in Fig.5 illustrates the AXI to APB bridge operation for various read and writes transfers.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 9, November- 2012

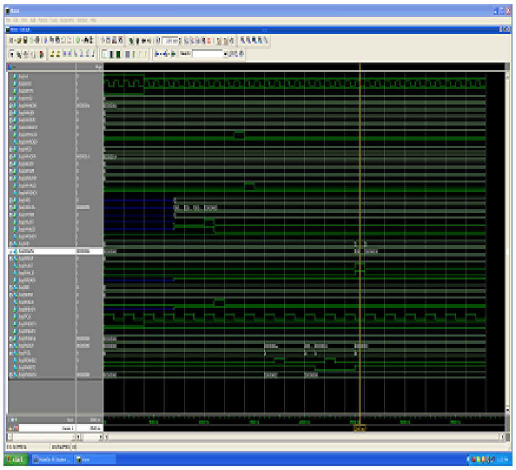

Figure.5. Read and Write transfer

6. CONCLUSIONS

In this study, we provide an implementation of AXI to APB bridge which has the following features:

- 32-bit AXI slave and APB master interfaces.
- PCLK clock domain completely independent of ACLK clock domain.
- Support up to 16 APB peripherals with a burst length of 4 of incrementing burst type.
- Support the PREADY signal which translates to wait states on AXI.
- An error on any transfer results in SLVERR as the AXI read/write response.
- The transfer time reduces to half when compared with AXI-Lite to APB bridge.

REFERENCES

[1] ARM, "AMBA Protocol Specification 4.0" www.arm.com, 2010.
[2] Ying-ZeLiao, "System Design and Implementation of AXI Bus", National Chiao Tung University, October 2007.
[3] Clifford E. Cummings, "Coding And Scripting Techniques For FSM Designs With Synthesis-Optimized, Glitch-Free Outputs," SNUG (Synopsys Users Group Boston, MA 2000) Proceedings, September2000.
[4] Clifford E. Cummings, "Synthesis and Scripting Techniques for Designing Multi-Asynchronous Clock Designs," SNUG 2001
[5] Chris Spear, "*systemverilog for verification*, 2nd Edition", Springer, www.springeronline.com, 2008.
[6] Lahir, K., Raghunathan A., Lakshminarayana G., "LOTTERYBUS: a new high-performance communication architecture for system-on-chip deisgns," in Proceedings of Design Automation Conference, 2001
[7] Sanghun Lee, Chanho Lee, Hyuk-Jae Lee, "A new multi-channel on- chip-bus architecture for system-on-chips," in Proceedings of IEEE international SOC Conference, September 2004
[8] Martino Ruggiero, Rederico Angiolini, Francesco Poletti, DavideBertozzi, Luca 8
[9] Benini, Roberto Zafalon, "Scalability Analysis of Evolving SoC Interconnect Protocols," Int. Symposium on System-on-Chip, 2004. Lukai Cai, Daniel Gajski, "Transaction level modeling: an overview," in Proceedings of the 1st IEEE/ACM/IFIP international conference on[Hardware/software codesign and system synthesis, October 2003]
[10] Min-Chi Tsai, "Smart Memory Controller Design for Video Applications," Master thesis: National Chiao Tung University, July 2006.