IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 3, Issue 3, October 2014)

An Efficient AXI Read and Write Channel for Memory Interface in System-on-Chip

Abhinav Tiwari
M. Tech. Scholar, Embedded System and VLSI Design
Acropolis Institute of Technology and Research, Indore (India)
tiwari.abhinav21@gmail.com

Jagdish Nagar
Asst. Professor,
Acropolis Institute of Technology and Research, Indore (India)
jagdishnagar@acropolis.in

Abstract –**System-on-a-Chip (SoC) design has become more and more complexly. Because difference functions components or IPs (Intellectual Property) will be integrated within a chip. The challenge of integration is "how to verify on-chip communication properties". Although traditional simulation-based on-chip bus protocol checking bus signals to obey bus transaction behaviour or not, however, they are still lack of a chip-level dynamic verification to assist hardware debugging. This paper proposes an efficient AXI read and write channel for memory interface in SOC.**

Keywords –**Intellectual Property, SoC, AXI.**

I. Introduction

With the need of application, chip with a single processor can't meet the need of more and more complex computational task. We are able to integrate multiple processors on a chip thanks to the development of integrated circuit manufacturing technology Now as there are multiprocessing units and processors is getting faster, so compatibility with slow communication architectures a bit difficult furthermore this slow and conventional communication architecture limits the throughput.

To improve the performance we have to develop such efficient on chip Architecture which will be much faster system on-chip solution which removes the limitation of communication architecture. One of the solution is "AHB bus" but it can't give perfect parallelism as it can allow only one master to communicate at one slave only. While in our design there are five independent transfer channels which make multiple masters access multiple slaves at the same time and gain a perfect parallelism performance in MPSOC design

The AMBA AXI protocol is targeted at high-performance, high-frequency system designs and includes a number of features that make it suitable for a high-speed submicron interconnect. The objectives of the latest generation AMBA interface are to be suitable for high-bandwidth and low-latency designs Enable High Frequency operation using complex bridges meet the interface requirements of a wide range of components be suitable for memory controllers with high initial access latency provide flexibility in the implementation of interconnect architectures be backward-compatible with existing AHB and APB interfaces. The key features of the AXI protocol are:

- Separate address/control and data phases.
- Support for unaligned data transfers using byte strobes.
- Burst-based transactions with only start address issued.
- Separate read and write data channels to enable low-cost Direct Memory Access (DMA) ability to issue multiple outstanding addresses.
- Out-of-order transaction completion.
- Easy addition of register stages to provide timing closure.

II. Literature Survey

Previous method describes the implementation of AXI compliant DDR3 memory controller. It discusses the overall architecture of the DDR3 controller along with the detailed design and operation of its individual sub blocks, the pipelining implemented in the design to increase the design throughput. It has discusses the advantage of DDR3 memories over DDR2 memories and the AXI protocol operation it has also included combining and reordering the Read/Write commands. For attaining the maximum throughput from the memory, it operates all the memory banks in parallel and minimizes the effect of pre-charge/refresh and other DDR3 internal operation [1] before these paper describes the SoC platform and the bus encoding architecture [2].

In [3] method focuses on design and implementation of AXI bus protocol-based MPSoC architecture. The performance of MPSoC is determined not only by the capacity of processor,

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 3, Issue 3, October 2014)

but also by on-chip communication architecture [4] [5]. Now the speed of processor is becoming higher and higher, the performance is mainly limited by the communication architecture. However, the significantly increased design complexity of the MPSoC system causes the unacceptable simulation time with the traditional simulation methods. Hence, in [2], it was necessary to develop an FPGA prototype, which can provide accurate performance evaluation under real application and prediction for future ASIC design. In [2] focuses on the design and implement of AXI bus based MPSoC prototype, which translates data in burst, maximal length of which is up to 16 transactions. Besides, it only needs to translate the head address of the burst in this transaction. Owing to that feature, multiple masters accessing multiple slaves at one time becomes possible in sharing address bus architecture.

In [6] VIPACES (Verification Interface Primitives for the development of AXI Compliant Elements and Systems) was presented, a simple environment for the verification of AMBA 3 AXI systems in Verification IP (VIP) production. The elements come from the necessity of creating generic modules, in the verification phase, for this widely used protocol. These primitives are presented as a not compiled library written in SystemC where interfaces are the core of the library. The definition of interfaces instead of generic modules let the user construct custom modules improving the resources spent during the verification phase as well as easily adapting his own modules to the AMBA 3 AXI protocol.

In [7] a new methodology flow which will allow the visual definition of a complex SoC through instantiation of parametric were proposed Script based automation helps in integrating any IP with any configurations, selects relevant and corresponding Verification IPs (in-house developed-if Design IPs are standard) , uses suitable Bus rappers(OCP,EBI,Avalon,MicroBlaze,PicoBlaze,PIF,AXI,AHB,APB,Generic and others) and stitches all the components design as well verification(synthesizable testbench components) together and making use of TLMs, BFM (replacing CPUs with Master BFMs) or Process Core based designs creates an CSOC environment. The framework reduces the time to build integration and verify the functionality-it also has the complete set up from assembler to DFT. The main perspective of the complete CSOC system is that it not only integrates various designs IPs but also integrates corresponding inbuilt standard VIPs (Verification IPs) that are needed to verify a SoC in real life. These VIPs sit on the same bus that is inside the SoC and share the bus with various Design IPs.

In [8], the method for application-specific low-power bus architecture synthesis at system-level were presented. This paper has two contributions. First, this build a bus power model of AMBA AXI bus communication architecture. Second, is power model into a low-power architecture exploration algorithm were incorporated that enables system designers to rapidly explore the target bus architecture. The proposed exploration algorithm reduces power consumption.

The improvement of the process technology has enabled more and more functionality to be integrated into a single chip, which has increased the amount of on-chip communication between the integrated components. In a highly integrated system, the on-chip communication architecture becomes a critical factor affecting overall system performance and power consumption [9]. Designers therefore must give special emphasis on the choosing and optimizing of the on-chip communication architecture early in the design flow, preferably at the system level. The on-chip communication architecture, such as the bus matrix [5], has customizable topologies and parameters, which create a vast exploration space [10]. Different configurations in this space may have various power budgets and performance characteristics. Hence, it is crucial to find the rapid exploration technique which minimizes the power consumption of the target bus architecture under the given performance constraints, especially for mobile devices that have a limited energy budget.

III. ARCHITECTURE OF AXI PROTOCOL

The AXI protocol is burst-based. Every transaction has address and control information on the address channel that describes the nature of the data to be transferred. The data is transferred between master and slave using a write data channel to the slave or a read data channel to the master. In write transactions, in which all the data flows from the master to the slave, the AXI protocol has an additional write response channel to allow the slave to signal to the master the completion of the write transaction The AXI protocol enables Address information to be issued ahead of the actual data transfer Support for multiple outstanding transactions Support for out-of-order completion of transactions.

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 3, Issue 3, October 2014)

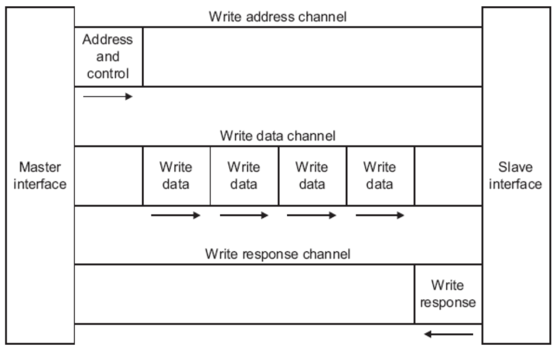


Figure 1: Write Address Channel

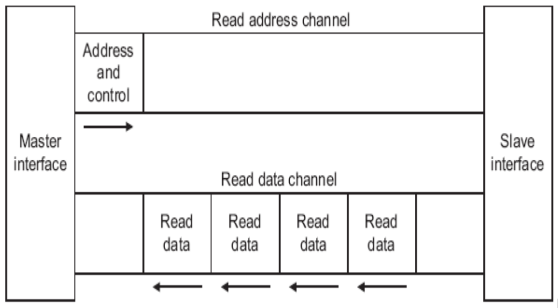


Figure 2: Read Address Channel

Research Specification

The AMBA AXI protocol is targeted at high-performance, high-frequency system designs and Includes a number of features that make it suitable for a high-speed submicron interconnect. The objectives of the latest generation AMBA interface are to:

1) Be suitable for high-bandwidth and low-latency designs.
2) Enable high-frequency operation without using complex bridges.
3) Meet the interface requirements of a wide range of components.
4) Be suitable for memory controllers with high initial access latency.
5) Provide flexibility in the implementation of interconnect architectures.
6) Be backward-compatible with existing AHB and APB interfaces.

The key features of the AXI protocol are:

1) Separate address/control and data phases.
2) Support for unaligned data transfers using byte strobes.
3) Burst-based transactions with only start address issued.
4) Separate read and write data channels to enable low-cost Direct Memory Access (DMA).
5) Ability to issue multiple outstanding addresses.
6) Out-of-order transaction completion.
7) Easy addition of register stages to provide timing closure.
8) Support for burst lengths up to 256 beats.
9) QUALITY of Service (QoS) signaling.
10) Support for multiple region interfaces.
11) Updated write response requirements.
12) Updated AWCACHE and ARCACHE signaling details.
13) Additional information on ordering requirements.
14) Details of optional User signaling.
15) Removal of locked transactions.
16) Removal of write interleaving.

Each of the five independent channels consists of a set of information signals and uses a two-way VALID and READY handshake mechanism. The information source uses the VALID signal to show when valid data or control information is available on the channel. The destination uses the READY signal to show when it can accept the data. Both the read data channel and the write data channel also include a LAST signal to indicate when the transfer of the final data item within a transaction takes place. Read and write transactions each have their own address channel. The appropriate address channel carries all of the required address and control information for a transaction. The AXI protocol supports the following mechanisms like variable-length bursts, from 1 to 16 data transfers per burst, bursts with a transfer size of 8-1024 bits, wrapping, incrementing, and non-incrementing bursts.

The read data channel conveys both the read data and any read response information from the slave back to the master. The read data channel includes- the data bus, that can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide a read response indicating the completion status of the read transaction. The write data channel conveys the write data from the master to the slave and includes- the data bus that can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide, one byte lane strobe for every eight data bits, indicating which bytes of the data bus are valid. Write data channel information is always treated as buffered, so that the master can perform write transactions without slave acknowledgement of previous write transactions. The write response channel provides a way for the slave to respond to write transactions. All write transactions use completion signaling. The completion signal occurs once for each burst, not for each individual data transfer within the burst.

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 3, Issue 3, October 2014)

Channel Definition

Each of the five independent channels consists of a set of information signals and uses a two way VALID and READY handshake mechanism. The information source uses the VALID signal to show when valid data or control information is available on the channel. The destination uses the READY signal to show when it can accept the data. Both the read data channel and the write data channel also include a LAST signal to indicate when the transfer of the final data item within a transaction takes place.

Read and write address channels. Read and write transactions each have their own address channel. The appropriate address channel carries all of the required address and control information for a transaction. The AXI protocol supports the following mechanisms:

- Variable-length bursts, from 1 to 16 data transfers per burst.
- Bursts with a transfer size of 8-1024 bits.
- Wrapping, incrementing, and non-incrementing bursts
- Atomic operations, using exclusive or locked accesses.
- System-level caching and buffering control
- Secure and privileged access.

Read Data Channel

The read data channel conveys both the read data and any read response information from the slave back to the master. The read data channel includes:

- The data bus that can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide.
- A read response indicating the completion status of the read transaction.

Write Data Channel

The write data channel conveys the write data from the master to the slave and includes:

- The data bus that can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide
- One byte lane strobe for every eight data bits, indicating which bytes of the data bus are valid.

Write data channel information is always treated as buffered, so that the master can perform write transactions without slave acknowledgement of previous write transactions.

Write Response Channel

The write response channel provides a way for the slave to respond to write transactions. All write transactions use completion signaling. The completion signal occurs once for each burst, not for each individual data transfer within the burst.

IV. Simulation And Results

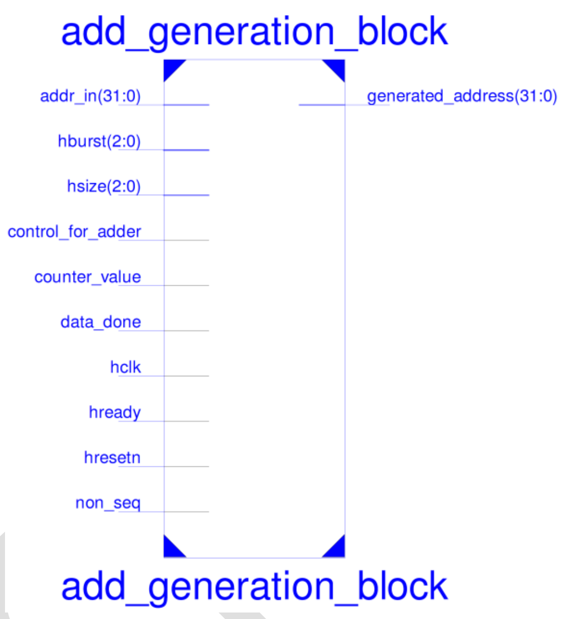


Figure 3: Pin diagram of address generation block

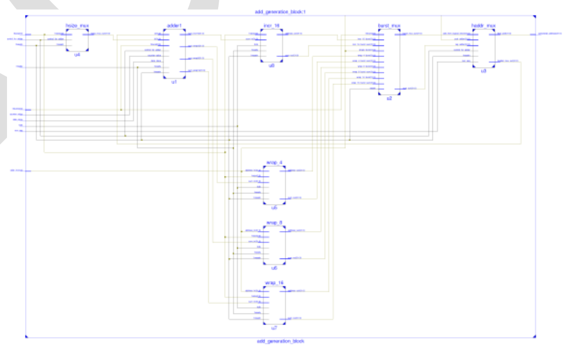


Figure 4: RTL Schematic of address generation block

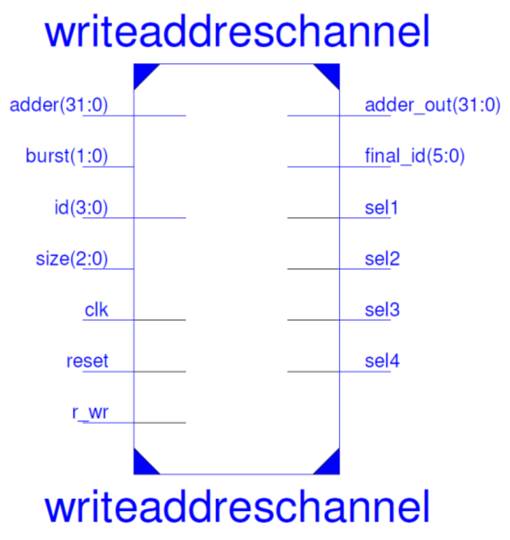


Figure 5: Pin diagram of write address channel

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 3, Issue 3, October 2014)

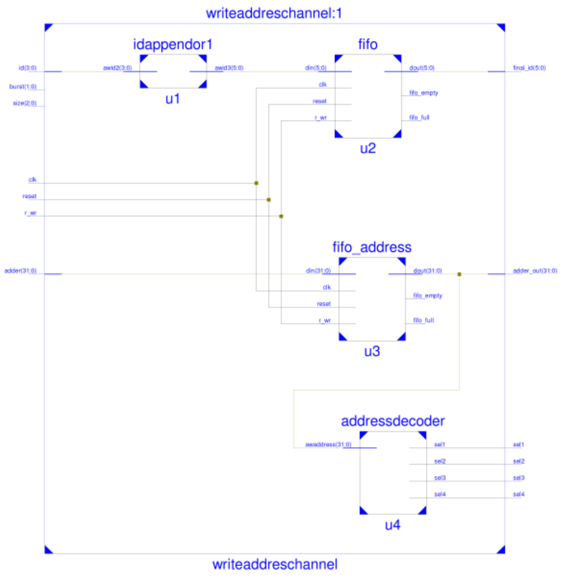


Figure 6: RTL Schematic of write address channel

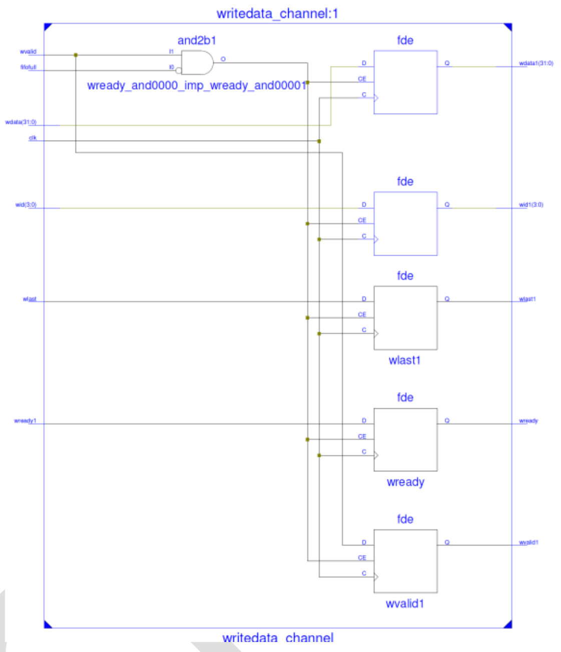


Figure 8: RTL Schematic of write data channel

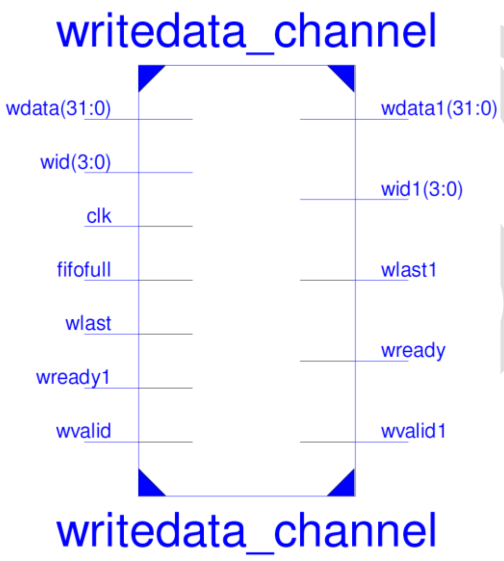
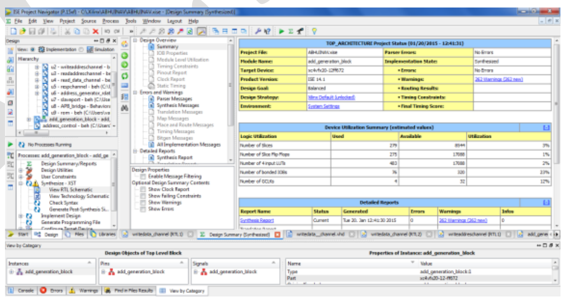


Figure 7: Pin diagram of write data channel



TOP_ARCHITECTURE Project Status (01/20/2015 - 12:41:31)			
Project File:	AXI4LITE.xise	Parser Errors:	No Errors
Module Name:	add_generation_block	Implementation State:	Synthesized
Target Device:	xc4vfx20-12ff672	• Errors:	No Errors
Product Version:	ISE 14.1	• Warnings:	262 Warnings (262 new)
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	279	8544	3%
Number of Slice Flip Flops	275	17088	1%
Number of 4 input LUTs	483	17088	2%
Number of bonded IOBs	76	320	23%
Number of GCLKs	4	32	12%

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Tue 20. Jan 12:41:30 2015	0	262 Warnings (262 new)	0

Properties of Instance: add_generation_block

Name	Value
Type	add_generation_block:1
Part	xc4vfx20-12-ff672

Figure 9: Device utilization summary

V. Conclusion

AMBA AXI bus specification and a technology independent methodology for designing of IP Core Read and Write operation is satisfied. In this paper, data transactions were carried out using AMBA AXI protocol for read and write channel for memory interface in System-on-Chip, which is modelled in VHDL using Xilinx tool.

Reference

[1] Vijaykumar, R K Karunavathi, Vijay Prakash, "Design of Low Power Double Data Rate 3 Memory Controller with AXI compliant", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume 1, Issue 5, June 2012.

[2] Osborne, S., Erdogan, A.T. Arslan, T., Robinson, D., "Bus encoding architecture for low- power implementation of an AMBA-based SoC platform", IEEE Proceedings on Computers and Digital Techniques, Vol. 149, Issue 4, July 2002.

[3] Fu-ming Xiao, Dong-sheng Li, Gao-Ming Du, Yu-kun Song, "Design of AXI bus based MPSoC on FPGA", 3rd International Conference on Anti-counterfeiting,

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 3, Issue 3, October 2014)

Security, and Identification in Communication (ASID), 2009.

[4] Terry Tao Ye, Luca Benini, Giovanni De Micheli, "Packetized On-Chip Interconnect Communication Analysis for MPSoC", Proceedings of Design Automation and Test in Europe, 2003.

[5] Sudeep Pasricha, Nikil Dutt, "On-Chip Communication Architectures: System on Chip Interconnect", Morgan Kaufmann, 2010.

[6] Sánchez-Peña, A., Carballo, P. P., García, L., & Núñez, A., "VIPACES, Verification Interface Primitives for the Development of AXI Compliant Elements and Systems", IEEE 9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools, 2006.

[7] Ashwin K. Kumaraswamy , V. A. Chouliaras , T. R. Jacobs, J. L. Nunez-yanez, "System-on-Chip Design Framework (SDF) unifying Specification Capture and Design Modeling", In Proceedings of the 2005 Electronic Design Processes (EDP) Workshop, pp. 6-8, April.

[8] Na, Sangkwon, Giwon Kim, and Chong-Min Kyung, "Lifetime maximization of video blackbox surveillance camera", IEEE International Conference on Multimedia and Expo (ICME), 2011.

[9] Na, S., Yang, S., & Kyung, C. M., "Low-power bus architecture composition for AMBA AXI", Journal of Semiconductor Technology and Science, Vol. 9, Issue 2, 2009.

[10] S. Pasricha, N. Dutt, M. Ben-Romdhane, "Constraint-Driven Bus Matrix Synthesis for MPSoC", ASPDAC 2006.