ISSN:0975-9646

Verification of AMBA based AXI 4 Slave Interface

Krithi B[1], Sudarshan Bhat[2], Yogesh Panchaksharaiah[3]

[1]*Final year M.Tech (VLSI Design & Embedded Systems) Student, Department of E&CE, Mangalore Institute of Technology and Engineering, Mangalore, Karnataka, India*

[2]*Associate Professor, Department of E&CE, Mangalore Institute of Technology and Engineering Mangalore, Karnataka, India*

[3]*Technical Manager, Mindtree Ltd, Bangalore, Karnataka, India*

Abstract— **The increasing amount of logic that can be placed onto a single silicon die is driving the development of highly integrated SoC designs. An important feature for any of the SoC is based on how they interconnect. AMBA protocols are today the de facto standard for 32-bit embedded processors because they are well documented and can be used without royalties. The AMBA AXI 4 protocol supports high-performance, high-frequency system designs. It is suitable for high-bandwidth and low-latency designs and provides high-frequency operation without using complex bridges. It provides flexibility in the implementation of interconnect architectures and is backward-compatible with existing AHB and APB interfaces. This Project is aimed at the Verification of the AMBA based Design of the AXI4 Slave Interface and the Verification Environment is built using SystemVerilog coding. This slave interface can be used to connect different peripherals into AMBA based processors without using bridge. The developed slave interface can also be used to connect different peripherals like SPI, I2C, UART etc., into non AMBA based processors by developing wrapper around AXI4 slave interface.**

Keywords— ***Advanced Microcontroller Bus Architecture (AMBA), Advanced Peripheral Bus (APB), AMBA High-performance Bus (AHB), Advanced Extensible Interface (AXI).***

I. INTRODUCTION

The Advanced Extensible Interface (AXI) is a part of the Advanced Microcontroller Bus Architecture (AMBA) which is developed by ARM (Advanced RISC Machines) company. It is an On-Chip communication protocol. The AMBA AXI protocol supports high-performance, high-frequency system designs.

The AXI protocol is suitable for high-bandwidth and low-latency designs. It provides high-frequency operation without using complex bridge. It meets the interface requirements of a wide range of components. AXI protocol provides flexibility in the implementation of interconnect architectures. It is backward-compatible with existing AHB and APB interfaces.

The key features of the AXI protocol are that it has separate address/control and data phases & support for unaligned data transfers, using byte strobes. It utilizes burst-based transactions with only the start address issued. It has separate read and write data channels that provide low-cost Direct Memory Access (DMA). It supports for issuing multiple outstanding addresses. It support for out-of-order transaction completion. It permits easy addition of register stages to provide timing closure.

II. LITERATURE REVIEW

The Advanced Microcontroller Bus Architecture (AMBA) is an open-standard, on-chip interconnect specification for the connection and management of functional blocks in system-on-a-chip (SoC) designs. It facilitates development of multi-processor designs with large numbers of controllers and peripherals. Since its inception, the scope of AMBA has, despite its name, gone far beyond micro controller devices. Today, AMBA is widely used on a range of ASIC and SoC parts including applications processors used in modern portable mobile devices like smartphones.

AMBA was introduced by ARM in 1996. The first AMBA buses were Advanced System Bus (ASB) and Advanced Peripheral Bus (APB) [1&2]. In its second version, AMBA 2, ARM added AMBA High-performance Bus (AHB) that is a single clock-edge protocol [3-5]. In 2003, ARM introduced the third generation, AMBA 3 [6&7], including AXI to reach even higher performance interconnect and the Advanced Trace Bus (ATB) as part of the CoreSight on-chip debug and trace solution. In 2010 the AMBA 4 specifications were introduced starting with AMBA 4 AXI4, then in 2011 extending system wide coherency with AMBA 4 ACE with a re-designed high-speed transport layer and features designed to reduce congestion [8-10].

Advanced microcontroller bus architecture (AMBA) protocol family provides metric-driven verification of protocol compliance, enabling comprehensive testing of interface intellectual property (IP) blocks and system-on-chip (SoC) designs. The AMBA advanced extensible interface 4 (AXI4) update to AMBA AXI3 includes the following: support for burst lengths up to 256 beats, updated write response requirements, removal of locked transactions and AXI4 also includes information on the interoperability of components. AMBA AXI4 protocol system supports 16 masters and 16 slaves interfacing. The design is implemented using Verilog- HDL [11-13].

III. AXI PROTOCOL SPECIFICATIONS

A typical system consists of a number of master and slave devices connected together through the Interconnect. The AXI protocol provides a single interface definition, for the interfaces:

• Between a master and the interconnect

• Between a slave and the interconnect
• Between a master and a slave.

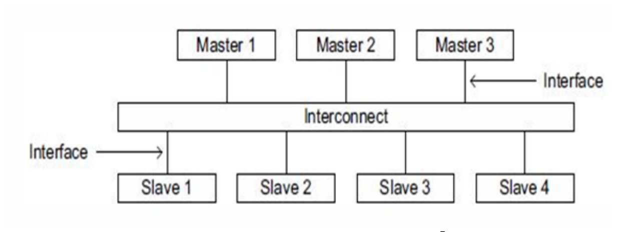

Fig 3-1 AXI System Topology

The AXI protocol is burst-based and defines the following independent transaction channels: Read Address Channel, Read Data Channel, Write Address Channel, Write Data Channel and the Write Response Channel.
An address channel carries control information that describes the nature of the data to be transferred. The data is transferred between master and slave using either:
A write data channel to transfer data from the master to the slave. In a write transaction, the slave uses the write response channel to signal the completion of the transfer to the master.
A read data channel to transfer data from the slave to the master.
The AXI protocol permits address information to be issued ahead of the actual data transfer. It supports multiple outstanding transactions. It also supports out-of-order completion of transactions.

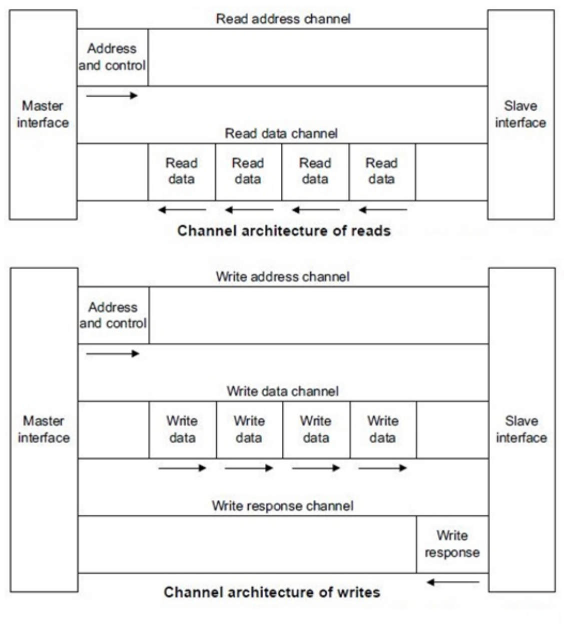

Fig 3-2 AXI Channel Architecture

IV. Proposed Verification Environment

The AXI Slave has been designed and verified using Master-Verification IP. The design for slave has the five bi-directional channels as the Input/ Output. The design is done using the finite state machine approach. The verification methodology adopted covers the principles like Constrained-Random stimulus, Functional Coverage, Common test bench for all tests. Test-specific code is kept separate from test bench. Automation Scripts support various switches so that we can re-use the test cases for many different scenarios. The Verification Environment is as shown in Fig 4-1.

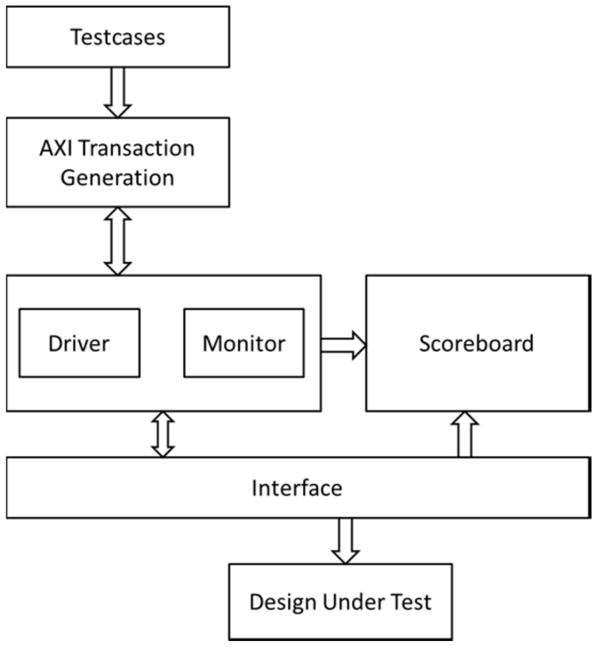

Fig 4-1 Verification Environment

The basic components of the verification environment are described below:

1. Testbench
Testbench mimics the environment in which the design resides. It checks whether the RTL implementation meets the design specification or not. This environment creates invalid and unexpected as well as valid and expected conditions to test the design.

2. Test Cases
Test cases are written for different scenarios, which cover the functionality, corner cases. Basic read write transactions, various burst modes are verified. Parameterization will be randomized for different test cases and are written to check all the possible scenarios. These test cases are run in regression with multiple seeds.

3. DUT
The verification environment is organized in a hierarchical layered structure which helps to maintain and reuse it with the DUT.

4. Virtual Interface
Virtual interfaces provide a mechanism for separating abstract models and test programs from the actual signals that make up the design. It allows the same subprogram to operate on different portions of design, and to dynamically control the set of signals associated with the subprogram. Instead of referring to the actual set of signals directly, users are able to manipulate a set of virtual signals.

5. Scoreboard
The scoreboard collects write address and control information on address channel, and data on write data channel at the output of AXI slave and compares it with the address, data and control information at the output of DUT.

It collects the responses at the DUT side and compares it with that on the input side. Also, it collects read address and control information on address channel at the AXI slave input side and compares it with the address and control information at the output of the DUT. It collects the read response and data at the DUT side and compares it with that on the input side.
The IP architecture for the AXI 4 Verification is shown in Fig 4-2.

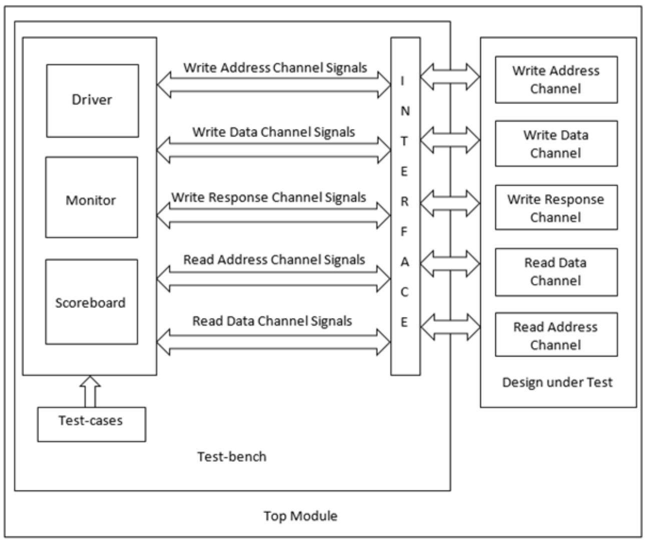

Fig 4-2 Verification IP Architecture

V. Conclusion and Future scope

In this paper, an effective verification environment for AXI bus is developed with SystemVerilog. The proposed multi-layer testbench is comprised of AXI master, AXI slave, assertions, scoreboard and coverage analysis. The AXI 4 slave has been designed and verified using Directed-Test bench methodology. Following the latest AXI specifications, the designed Slave can easily be used to connect different peripherals like SPI, I2C, UART etc., into non AMBA based processors by developing wrapper around AXI4 slave interface.

Acknowledgment

The authors wish to thank Mindtree Limited. This work was supported by a grant from Mindtree Limited.

References

[1] *"AMBA Peripheral Bus Controller Data Sheet" Copyright © 1996 Advanced RISC Machines Ltd (ARM).*
[2] *Ramagundam, S.; Dept. of Computer Sci., Troy Univ., Montgomery, AL, USA ; Das, S.R. ; Morton, S. ; Biswas, S.N. , "Design and implementation of high-performance master/slave memory controller with microcontroller bus architecture", Instrumentation and Measurement Technology Conference (I2MTC) Proceedings, 2014 IEEE International, 12-15 May 2014.*
[3] *"AMBA™ Specification (Rev 2.0)" 13th May 1999-A, First release, Copyright ARM Limited 1999.*
[4] "Soo-Yun Hwang; Dept. of Comput. Eng., ChungNam Nat. Univ., Taejon, South Korea; Kyoung-Sun Jhang "An improved implementation method of AHB Bus Matrix", SOC Conference, 2005. Proceedings. IEEE International, 25-28 Sept. 2005
[5] *Hu Yueli; Key Lab. of Adv. Display & Syst. Application., Shanghai Univ., Shanghai, China ; Yang Ben "Building an AMBA AHB Compliant Memory Controller", Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference, 6-7 Jan. 2011.*
[6] "AMBA AXI Specification (AR500-DA-10008)" 16 June, 2003-A, First release, Copyright ARM Limited 2003
[7] Paunikar, A.; Sch. of Electron. Eng., VIT Univ., Vellore, India ; Gavankar, R.; Umarikar, N. ; Sivasankaran, K. , "Design and implementation of area efficient, low power AMBA 3-APB Bridge for SoC" , Green Computing Communication and Electrical Engineering (ICGCCEE), 2014 International Conference, 6-8 March 2014
[8] "AMBA AXI 4 and ACE Protocol Specification" 28 October 2011 D Non-Confidential First release of AMBA AXI 4 and ACE Protocol Specification.
[9] Xu Yang ; Harbin Inst. of Technol., Harbin ; Zhang Qing-li ; Fu Fang-fa ; Yu Ming-yan, "NISAR: An AXI compliant on-chip NI architecture offering transaction reordering processing" ASIC, 2007. ASICON '07. 7th International Conference, 22-25 Oct. 2007.
[10] Manjula, R.B. ; Manvi, S.S. ; Kaunds, P. "Data transactions on system-on-chip bus using AXI4 protocol" Recent Advancements in Electrical, Electronics and Control Engineering (ICONRAEeCE), 2011 International Conference, 15-17 Dec. 2011.
[11] "Verilog-A Language Reference Manual Analog Extensions to Verilog HDL", Version 1.0, Open Verilog International August 1, 1996
[12] "Verilog-AMS Language Reference Manual**"**, Release 2.3.1, Accellera Systems Initiative , 06-2009
[13] "Verilog-AMS Language Reference Manual**".** Release 2.4, Accellera Systems Initiativ, 06-2014.