ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

Design and Verification of AMBA 3 AHB Lite Protocols by using GO2UVM Package

[1]Asharani Pattedar, [2]Saroja V. Siddamal
[1] M Tech student, Department of ECE B.V.B. College of Engineering and Technology, Hubli India
[2] Professor, Department of ECE B.V.B. College of Engineering and Technology, Hubli India

Abstract: **-- The AMBA 3 AHB protocol design act as an interface between two different IP cores. The current project emphasis on AMBA 3 AHB lite where the design and verification of a flexible burst operation is proposed. Basically, AHB lite burst operation is a sequence of operations that happens with respect to the size given and it supports only three burst sizes. During the burst operation, the size acts as one of the inputs to the master. After each burst operation, the master or slave will go to the IDLE stage. The AHB lite design contains basic blocks like master and slave, and the working of these blocks is without arbitration scheme. According to this scheme one master can access the bus at one time. The present work is on AHB-Lite master and slave model, at different test cases and describing their simulation Accuracy is 100%. It is built by the standard language GO2UVM package on the relevant simulator Riviera.**

***Index Terms*— Advanced Microcontroller Bus Architecture (AMBA), Advanced High Performance Bus(AHB), Advanced Peripheral Bus(APB), Universal Verification Methodology (UVM).**

I. INTRODUCTION

Recently in the industry, higher priorities are given in the advancement of Silicon on Chip (SOC) devices with reusable IP cores. The major challenge is in ensuring appropriate lossless communication between various IP cores in SOC devices. This can be guaranteed with the assistance of well designed communication protocols like AMBA from ARM Ltd. It describes a general set of buses for use in ASICs, SoCs, and traditional micro-controllers.

The verification process plays a vital role in outline cycle of an ASIC. Since the complexity of System On Chip (SOC) expanding, verification has also became one of the difficult tasks for design engineers. Hence verification as per specifications decides the preciseness of the design.

In this paper the design and synthesize of AMBA 3 AHB Lite Protocol master and slave integration has been conveyed and verified by utilizing the GO2UVM package. The interfaces are proficient of reacting to “okay and error” responses during a basic read and write transfer. The AMBA 3 AHB Lite system is designed by utilizing Verilog (Hardware description language) synthesized and verified by using GO2UVM package.

The developing complexity and size of computerized designs have made functional verification a tremendous challenge. In the most recent decade several new technologies have emerged in the region of verification and some of them have caught their place as a necessity in the verification process. 40 to 50% of project resources go to functional design verification. The design verification cost component is about 40% of the aggregate outline cost. A fundamental role for functional verification is utilization to recognize non attendance of progress with the goal that bugs can be determined and altered before it gets gave to costumer. If the designer makes an error in designing or coding, this results as a bug in the Chip. If this bug is executed, in specific circumstances the framework will convey wrong results, making a failure of the chip.

II. RELATED WORK

The AMBA Bus, developed in 1996, is an open standard which allows all soc manufacturers to use arm-based processors in their SOCS. AMBA architecture consist of two types of buses the one which is connected to core IPS is advanced high performance bus, other which is connected to multitude

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

of peripheral and components is advanced peripheral bus which is flexible in nature. The AMBA advanced high performance bus ahb are commonly used to connect a DSP, A processor and memory controllers to be used for connecting different peripheral IPS. AMBA is flexible for upgrading SOCS by replacing corresponding IPs with a newer design (like FM receiver) or version (like Bluetooth module) it also contains a bridge, connecting the APB and AHB buses. Bridges are bus-to-bus interfaces that will allow IPS to connect different buses for standardized communication

The AHB buses were first implemented in AMBA 2 protocol, and were later upgraded in AMBA 3. A simple AHB system is made up of one or more masters which are connected to the slave devices (like memory controllers). This type of arrangement in AHB is known as layer.

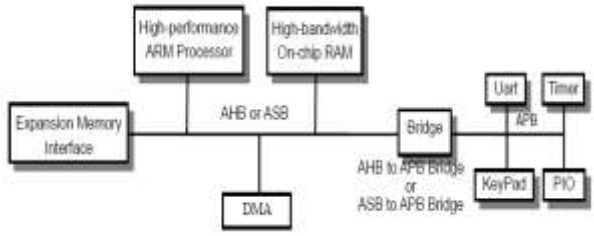

Figure 1: Architecture of AMBA BUS

In a layer it contains a single decoder which helps to resolves the master requests into a slave activation signals. An arbiter is also included, if more than 2 masters, to decide which master have to get the access to the bus at any one time. AHB masters helps in performing burst transfers in which multiple data elements can be read or written from or to a slave in a single transaction. AHB exist in different variants as discussed below. In a multi-layer AHB, each of the connected masters contains single layer. Each slave is provided with a slave arbiter that can choose which layer is given access and in case two masters requests access to the slave at the same time. Hence multi-layer arrangement permits more than one master to be effectively conversing with a slave [13]. Arbitration scheme for each slave-arbiter may be different such as fixed priority or round robin arbiter. Of course, the interconnected matrix becomes more complex with an increase in number of masters and slaves. To overcome this problem AHB Lite is proposed, in which a few slaves are made local to a layer. Hence it will remove the need for a separate arbiter and allows multiple slaves being accessed as a single device by using multi-port slaves which do not need arbiters. Thus it allows multiple masters on a single layer i.e. decreasing the number of decoders [4].

In VLSI industry there are various different verification methodologies and languages are used to verify the design. System Verilog is a one type of design and verification language (HVDL) which gives composite data types and new constructs expected to develop a verification environment and is most routinely used now days for verification purposes. Mainly an approach is applied to a language in an ordered to arrange efficient way for doing verification of a design. UVM is the most recent verification methodology being utilized as a part of VLSI industry for verification. The Universal verification Methodology (UVM) is a standardized hybrid approach for verifying complex configuration in the VLSI industry. UVM has full industry wide backing and standardized under the Accelerate Systems Initiative [7].

UVM is capable of verifying everything without exception in the universe at least all things in the domain of integrated circuits. Different companies using different languages or methodologies for this UVM is universal for all the methodologies, in which different methodologies like VMM, OVM, AVM etc, these methodologies to do same thing but some methods performing better than others, all requiring retaining and expansive conversion cost, best known methods are not always shared, resulting in deferential and expansive to repair design. Verification solutions are not easily packaged.[5]

UVM developed by from a combination of other verification methodologies for example, OVM and UVM. UVM gives the best structure to achieve coverage driven verification (CDV). CDV combines automatic test generation, self checking test benches, and coverage metrics to essentially lesson the time spent verifying a design. Universal verification methodology (UVM) will be upheld by the various EDA vendors [6].

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

III. METHODOLOGY

Design methodology

Protocol AMBA 3 AHB-Lite is used to address the requirement conditions for high-performance synthesizable designs. It is bus interface which can help in supporting a single bus master and can provide a higher-bandwidth operation.

AHB-Lite protocol extracts the features needs for high clock frequency, high-performance systems including the following

- Burst transfers.
- Single-clock edge operation.
- Non-tristate implementation.
- It supports data bus configurations such as 64, 128, 256, 512, and 1024 bits.

The most widely used AHB-Lite slaves includes internal memory devices, external Memory interfaces and higher bandwidth peripherals Although AHB-Lite slaves includes low bandwidth peripherals they typically resides on AMBA Advanced Peripheral Bus (APB) because of system performance reasons an AHB-Lite slave, called APB bridge was used to bridge the higher level of the bus with the APB Figure 1 depicts the design of a single master AHB-Lite system with single master and 3 slaves. The bus circuit logic contains an address decoder with a slave-to-master multiplexor. The decoder helps in monitoring the address send by the master so that an appropriate slave can be selected. The multiplexor sends back the corresponding slave output data to the master. Figure 2 depicts the block diagram of AHB-Lite. The main components of an AHB-Lite system are Master, Slave, Mux and Decoder.

Master: A master is a device, which is connected to a high-throughput system bus. It is an IP which has a capacity to initiate communications on the bus. Figure 3 shows the master signal. Slave: A slave is a device, which gives response to requests for communication from the master. During peripheral devices, example network controllers and memory controllers are act as slaves and even processing elements such as DSPs may act as slaves.

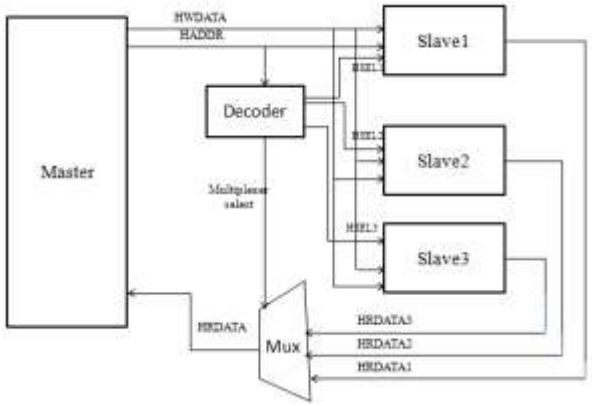

Figure 2: AHB-Lite Master

Decoder: This component helps in decoding the address for every data transfer by providing a select signal for the slave. A control signal is also send to the multiplexor along with the select signal. In all AHB-Lite implementations, a single centralized decoder is required when it has more than 2 slaves.

Mux: In order to multiplex the read data bus and the corresponding response signals send which comes from slaves to master, a Mux is needed. The decoder provides a control signal for the multiplexor. Single centralized multiplexors with two or more slaves are commonly used in all AHB Lite implementations.

The signals are associated to master and slaves are as follows:

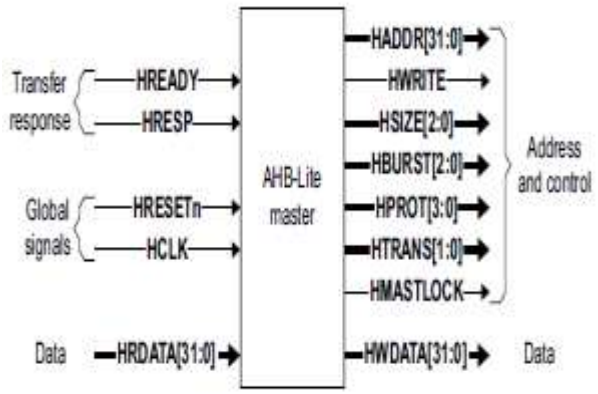

Figure 3: AHB-Lite Master

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

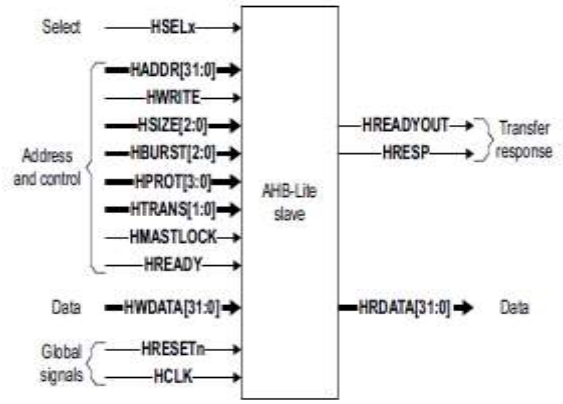

Figure 4: AHB-Lite Slave

1. ***Data bus****:* It is used to read data and write data. The width of data bus is fixed in the form of bits. In AHB lite Protocol data bus width is of 32-bits. The master and slave of a SoC have the capacity to read or write data throughout data bus. In order to maintain the data consistency, only single device can use the data bus at a certain point of time.

2. ***Address bus:*** Physical address is specified by IPs address bus .The width of the address bus is of 32-bits. In the system every IPs are connected to the address bus. At any stage, only single IP is allowed to read data or write data. Decoder reads the address bus, which allows generating slave-select signals, which in turn allows masters to raise the specific slaves.

3. ***HCLK:*** It indicates the signal timings are associated to the positive edge of HCLK.

4. ***HRESETn*** : indicates reset signal which is active low and resets the system and the bus. It is the only one active low AHB-Lite signal.

5. ***HBURST [2:0]:*** The burst type indicates, that if it is a single transfer or if it forms a part of a burst. Fixed length bursts of 4, 8, and 16 beats are supported. The burst can be wrapping or incrementing, undefined length of incrementing bursts are also supported.

6. ***HMASTLOCK:*** When mastlock signal is high then the current data transfer is part of a locked sequence and has the same timing as like control and address signals.

7. ***HPROT [3:0]:*** The protection control signal gives extra information about the bus access and is mainly intended for use by some module that needs to execute some level of protection.

8. ***HSIZE [2:0]:*** The size of the transfer which is in terms of byte, half word, or word. This protocol will allow for larger sizes of up to a maximum of 1024 bits.

9. ***HWRITE*****:** Indicates direction of the transfer. When this signal is high indicates a write transfer and when this signal is low then a read transfer takes place.

10. ***HTRANS [1:0]:*** This indicates the types of transfer, there are 4 types of transfers:

    - HTRANS => 00 => idle transfer, master will not perform any operation.
    - HTRANS => 01 => BUSY transfer, master will insert wait state.
    - HTRANS => 10 => NONSEQ transfer; here the transfer address is not depending upon the previous transfer address. It is first transfer of the burst
    - HTRANS => 11 => SEQ transfer, here the transfer address is depend upon the previous transfer address.

11. ***HREADY*****:** When this signal is high it indicates that a transfer has completed on the bus. It has to be driven low to extend the transfer.

12. ***HRESP:*** When HRESP signal zero, implies the transfer status is "OKAY". When HRESP signal one, implies the transfer status is "ERROR".

Verification methodology:

The functional verification is carried out in the GO2UVM package environment, In order to analyze the functionality of assertion models; these models are simulated using Rivera pro simulator.

The main objective of a test bench is to verify the accuracy of the design under test (DUT). This is accomplished by the following steps.

- Generate stimulus.
- Apply those stimuli to the DUT.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

- Capture the response.
- Check for correctness.
- Compute progress against the all verification goals.

The Environments created through System Verilog might be various depending on implementer, while verification environment developed utilizing UVM continues as before for various vendors, i.e. it can be reused for various vendor IPs. We can tell that usage of a particular verification strategy is valuable to the extent better correspondence among architects and reusability of verification environment. Verifworks GO2UVM package created through UVM package, which is the one of the verification methodology, this methodology is not digress from the UVM. UVM approach is complex and tedious for expansive number of tests, to address these issues by utilizing a verifworks GO2UVM package, which hides all difficulties of UVM.

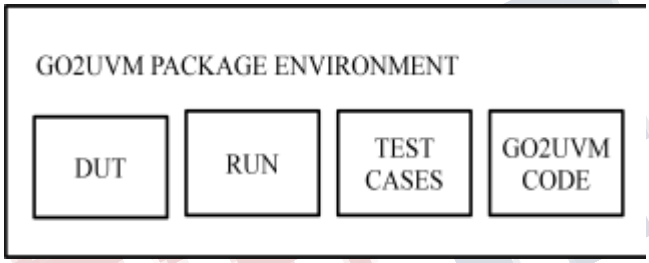

Figure 5: GO2UVM Environment

GO2UVM package which helps the user to reduce the complexity in the coding, the package is nice frame work to build reusable IPs, verification environment. GO2UVM package is shipped as a pre-compiled library much similar to standard UVM that gets shipped with EDA tool.

Goal of the GO2UVM package

1. Provide simple frame work for writing high quality simulation traces.
2. Not to deviate from UVM approach.
3. Introduce UVM to Verilog users.
4. Make them future ready for more powerful UVM.
5. Simple package on top of standard UVM released package in open source.
6. Hides all the complexities of UVM.
7. Provide a simple task based on driving stimulus to users.

GO2UVM Environment

This includes DUT, Run test, Test cases and GO2UVM code. By using this package we can perform functional verification for all models, in which we have shown in AHB lite we have performed a functional verification AHB Lite by using GO2UVM package environment as shown in figure 5.

DUT: Design under test is the verilog based design for this design we are perform the functional verification by using GO2UVM package.

Run: It includes file list, make file, run commands, and log result generated.

Test cases: Generating test cases (stimulus) for functional verification GO2UVM environment:

GO2UVM code block consist of the following modules

- Package.
- Interface.
- Clocking block.
- Modport.

Packages:

Verilog has very limited scoping capabilities but VHDL has package which can be widely used in design, synthesis and test bench. Hence System Verilog (SV) adds packages. Packages in SV are very similar to VHDL packages. It encapsulates a set of parameters, types, class.

Interface

An Interface enables the integration between the modules It permits the smooth flow of design among the modules. Interfaces encapsulate communication and interconnection between blocks. It is having heading as input, output and inout also. Interfaces can also have parameters like modules. Interface revelation is much the same as a module statement. Uses keywords interface, end interface for characterizing. Inside a module, use various leveled names for signals in an interface.

Look up DUT, interface and UVM

- The top module of a GO2UVM package very similar to verilog.
- Initially create module tb top.
- We Instantiate the DUT.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

- Instantiate interface.
- Generate clock.
- Instantiate UVM test.
- Call run test ().

In Interface block class is introduced, Class is the basic building block of system verilog verification, it is defines the abstract characteristic and behavior methods of an object. Class is blueprint that allows creating one or more objects of the same type. In the RTL design every functionality is inside a module similarly in verification everything happens within class. class can have four parts.

- Members declaration
- Methods/action (functions and tasks)
- Constraints
- Coverage

Clocking block

Clocking block can be characterized by using the words like clocking and end clocking. A clocking block is basically utilized as a part of the test bench in order to avoid race around conditions. Clocking block provides synchronization to the all blocks with in a test bench. Clocking blocks are utilized to assemble all the signals. They are helpful in isolating clocking activities from its main data activities. The revelation and instantiation of clocking block can both happen inside the module.

Modports:

Modports describe directions of ports as if they are declared inside the module. To connect interface access inside of modports are utilized. Modports module consist of inputs, inout and outputs also ports.

AHB LITE and GO2UVM Advantages

The advantage of AHB-Lite protocol is as follows:

- Master does not lose the ownership of a bus.
- Master must not have the early terminated bursts. It does not require that the master has restructured a burst. Because the master can access at any time.
- Bus master do not have the Retry or Split transfer responses. Because it cannot hold the address of the previous transfer.

Advantages of using Interface module:

- It permits organized data stream between blocks.
- It can contain anything that could be in a module with the special case of other module definitions or instance.
- Port definitions are autonomous from modules.
- Expands the reusability.
- It Interface can be announced in a different record and can be compiled separately.
- Interfaces can contain tasks and functions; strategies imparted by all modules uniting to this information can be in one place.
- Reduces bugs which can bring about amid module connections.
- Simple to include or evacuate signal. Simple viability.

IV. RESULT AND ANALYSIS

The AHB Llite is designed and verified using Verilog HDL and system verilog, it is simulated using model sim and Riviera tool.

Figure 6, shows the verification output of WRAP4 WRITE operation. In this waveform the size of the burst is 32 bit and number of beats are 4 so it needs 16 bytes location to complete the transfer. Hence the address is incremented by 4. If the starting address is 0x38 and wraps takes place at 0x3c address.

Figure 7, shows the describes about WRAP8 READ operation. In this waveform the size of the burst is 32 bit and number of beats are 8 so it needs 32 bytes location to complete the transfer. Hence the address is incremented by 4. If the starting addresses 0x34 and wraps takes place at 0x20 addresses.

Figure 8, shows the verification output of INCR4 WRITE operation. In this waveform the size of the burst is 32 bit and no of beats are 4 so it needs 16 bytes location to complete the transfer. Hence the address is incremented by 4. If the starting address 0x38 and continuous increase in address the last address is 0x44.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 3, Issue 10, October 2016

Figure 6: Output of WRAP4 WRITE Operation

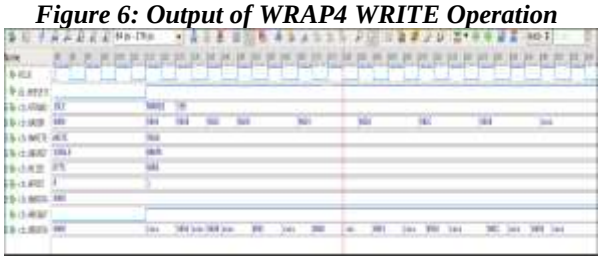

Figure 7: Output of WRAP8 READ operation

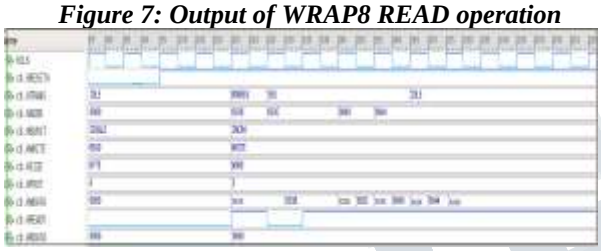

Figure 8: Output of INCR4 WRITE operation

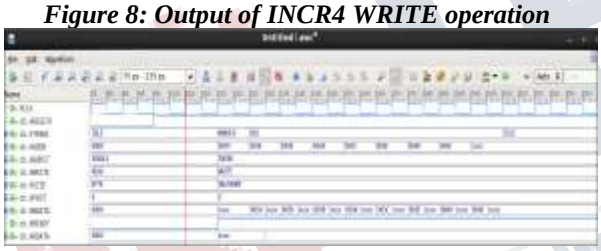

Figure 9: Output of INCR8 WRITE operation

Figure 9, shows the verification output of INCR8 WRITE operation. In this waveform the size of the burst is 16 bit and no of beats are 8 so it needs 16 bytes location to complete the transfer. Hence the address is incremented by 2. If the starting address 0x34 and continuous increase in address the last address is 0x42.

V. CONCLUSION AND FUTURE WORK

In this project, the design and verification of AHB Lite is proposed. Here single master and more than two slaves are used to transfer the data. The master can access the data at one time. While data transfer the decoder and multiplexer are used to choose appropriate signal. Hence we can observe simple READ and WRITE transfer, INCR4 INCR8, WRAP4, WRAP8 burst transfer. This protocol is verified by using GO2UVM package. By using this method the complexity of the verification has been reduced. It consumes less time to verify the design and cost is less as compared to other methodology. The future work is to use all the modes of HPROT signal.

REFERENCES

[1] URL:http:www.GO2UVMPackage.org.in.

[2] Heli Shah, Chinmay Modi" Design and Verification of APB Protocol with Coverage" IJAET, JUNE 2015.

[3] Mrs. Bhavana L. Mahajan, Dr. A. S. Hiwale, \Implementation of AHB Protocol using FPGA, IJAET, AUG 2012.

[4] P. Harishankar, Mr. Chusen Duarii, "Design and Synthesis of Efficient FSM for Master and Slave Interface in AMBA AHB" IJADR, JUNE 2014.

[5] Akhilesh Kumar, Richa Sinha, "Verification analysis of AHB Protocol with Coverage" IJAET, Nov 2015.

[6] Ben cohen, Srinivasan venkataramanan,Ajeetha kumara and Lisa piper, " System Verilog Assertions Handbook3rd edition for dynamic and formal verification, Vhdl" Cohen publishing 2013, IJAET, Nov 2015.

[7] Akhilesh Kumar, Richa Sinha,"Design and Verification analysis of AHB Protocol with Coverage" IJAET, Nov 2015.

[8] M. Caldari, M. Conti, M. Coppola, M. Giuliodori, C. Turchetti: C++ based System-on-chip IEEE Canadian Journal of Electrical and computer Engineering" vol. 26, no. 3/4, July/Oct. 2001, pp. 115-123.

[9] Santhi Priya Sarekokku, K. Rajasekhar, "Design and Implementation of APB Bridge based on AMBA AXI 4.0, IJERT, Vol.1, Issue 9, Nov 2012.

[10] R. Domer, Daniel D. Gajski: Reuse and protection of Intellectual Property in the SpecC system University of California, Irvine,http://www.ics.uci.edu.

[13] ARM, \ AMBA Speci_cation (rev2.0) and Multi layer AHB specification, Arm: http: www.arm.com, 2012.