iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** VI **Month of publication:** June 2022

DOI: https://doi.org/10.22214/ijraset.2022.44313

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue VI June 2022- Available at www.ijraset.com

A Review on Design Implementation and Verification of AMBA AXI- 4 lite Protocol for SoC Integration

Hemanthraju N[1], Prakruthi R[2], A Keerthi Sagar[3], Nimisha D[4], Dr. Suchitra M[5], Panchami S V[6]
[1, 2, 3, 4, 5, 6(]*Department of Electronics and Communication Engineering), Vidyavardhaka College of Engineering, Mysuru, India*

Abstract: ***The present modern bus protocols used for communication between different functional blocks on a System-on-Chip (Soc) designs face many different challenges among which complexity and communication management are the most important factors. These on-chip communications directly impact performance and functionality, hence depending on the application where the bus protocol is to be used, a perfect communication protocol is chosen. AMBA (Advanced Microcontroller Bus Architecture) provides various types of protocols to be used as IP, of which AXI4 (Advance Extensible Interface), is one of the widely used protocols for SoC designs. Out of its three different interconnect protocols: lite, stream and burst, AXI4-lite has the simplest architecture design that best suits to be used in applications where power, area and performance play a vital role. This paper briefs about using AMBA AXI4-lite bus protocol with more bus efficiency and performance in an SoC design for proper communication between various functional blocks present. The RTL is verified using UVM (Universal Verification Methodology) and various designing process is carried out using cadence tools.***

Keywords*: AMBA, SoC, AXI4, AXI4-lie, Performance, efficiency, functional blocks, RTL, Cadence tools and UVM*

I. INTRODUCTION

The Advanced microcontroller bus architecture (AMBA) family enables extensive testing of intellectual property (IPs) such as from ARM and other IP suppliers and system-on-chip (SoC) design through metric-driven protocol compliance verification. It delivers high-frequency operation using sophisticated bridges and supports tremendous performance and high-frequency. It is appropriate for high-bandwidth and low latency designs. The Advanced eXtensible Interface 4 (AXI4) bus family, defined as part of the ARM - AMBA standard's fourth version. The AMBA – AXI4-lite bus protocol is a subset of the AXI4 bus protocol with a simpler interface than the full-featured AXI-4 bus protocol. It only supports one ID thread per master, therefore it's best for an endpoint that only has to connect with one master at a time. There are five channels in the AXI4-Lite interface: read data, read address, write data, write address, and write response. Burst lengths up to 256 beats are supported by the AXI4 family, however only 1 beat is supported by the AXI4-lite.

A. Handshake Process

To send address, data, and control information between master and slave, all transaction channels use the same VALID or READY handshake process. This two-way flow control technique demonstrates the rate at which the data is exchanged by both master and slave. The VALID signal is sent by the information source to indicate that the address, data, or any other control information is available. The READY signal is sent by the destination which indicates that the destination is ready to accept any information that is available. When both the READY and VALID signals in a channel are timed during the rising edge of the clock, the handshake is considered to be complete.

II. METHODOLOGY

The channels: Read Address, Read Data, Write Address, Write Data and Write Response are the main communication channels in AXI4-lite. The functions of each channels between master and slave are as follows:

1) *Write Address Channel:* This channel transfers data containing the address to the slave where the write operation is to be performed. This address is fetched, and a suitable write operation is performed in the form of bursts and the same is verified.
2) *Write Data Channel:* After the successful write address, the signal containing the data is transferred to the previously fetched address in the slave.
3) *Write Response Channel:* After successful write address and write data operations, the slave sends a response or acknowledgment to the master.

©IJRASET: All Rights are Reserved

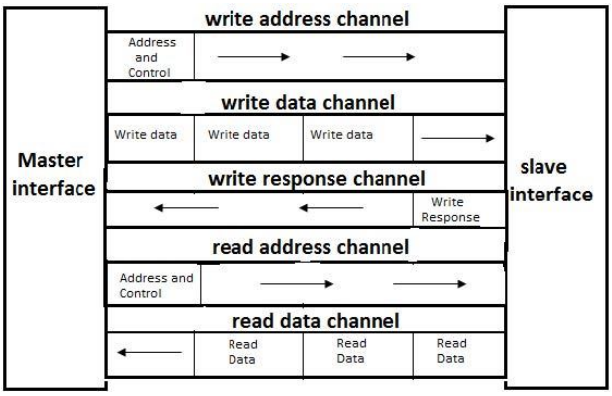


Fig. 1 Read and write transaction on AMBA AXI4-lite bus protocol

4) *Read Address Channel:* The slave receives the address from the master, where the data has to be read.
5) *Read Data Channel:* The data from the address is read and sent to the master and the same acts as a read response flag.

III.LITERATURE REVIEW

After extensive study of different papers, referring to design implementation and verification of AMBA AXI-4 lite bus protocol, we have listed our review in this paper.

A. Based on Performance

The authors [1] have proposed an efficient method for selecting an appropriate inter-connect bus protocol interface, such as AXI4 - stream, burst, or lite, for any given application, taking into account a variety of other parameters to ensure that the chosen architecture performs at maximum utilization and efficiency. Using the Zynq-7000 Xilinx platform for hardware/software (HW/SW) interactions, the authors compared and examined various on-chip interfaces by applying various code modifications. Different tests have been undertaken to investigate and assess the communication between the main processor and an accelerator, both of which are programmable hardware. AMBA AXI4 bus protocol with several optimization pragmas like array partitioning, data, flow, and loop pipelining are evaluated during the performance evaluation in relation to wireless sensor networks (WSN). Various results are gathered, including execution time when these pragmas were applied, FPGA performance results in milliseconds (ms) in log scale for three different AXI4 interfaces (stream, lite, and burst) without applying pragmas, average percentage difference considering only the computation time, total execution time, and communication time for three different AXI4 interfaces (stream, lite, and burst) without applying pragmas, and average percentage difference considering only the computation According to the findings, the percentage difference increases as the application's input data size grows larger, and the area occupied is examined both with and without pragmas. The authors found that good performance can be attained for many additional SoC configurations by carefully selecting the communication protocol and applying optimization pragmas. The goal of this paper's extension is to provide an analytical design in which the model recommends an AXI4 bus interface for a specific application.

The authors [2] have proposed a design on AMBA AXI-4 Lite protocol for effective read and write transactions with a special customized memory which consists of a set of five different channels from AXI4 for developing a simple interconnect between Master and Slave peripherals in performing effective READ/WRITE transactions. The design is progressed in Verilog HDL where functionality checking is done by the ISIM simulator and synthesized using Xilinx ISE 14.4 tool. This paper started out with a discussion about SoC and Bus protocols demonstrating AMBA significance, along with its versions which meant for adequate interface for a given application in a glimpse. The present digital era has more number of different peripherals on the system board with many processor cores; among which the AMBA-AXI of ARM's an open-standard, on-chip communication standard supports a rich set of bus signals, administrating several peripherals for a framework on a chip. The work focuses on the designing of AXI4-Lite which is meant for low memory peripheral requirements Concluding with the synthesis of AXI4-LITE Master Core and a Slave Memory peripheral along with the simulation waveforms exhibiting how a master performing effective Memory-mapped data transactions of one burst length integrated for 32-bits in the customized slave peripheral/data file. All the synthesis reports are carried out in Xilinx ISE 14.4 tool, followed by simulation using ISIM simulator.

©IJRASET: All Rights are Reserved

A work [3] on the implementation and design of AMBA AXI 4.0 Master for excessive and best performance the usage of Verilog HDL Code and simulation of the use of Xilinx tool is reported. It mentions the introduction of AXI 4.0 in order to overcome the deprivation of AHB such as separate address and data bus, high frequency also separate read and write channels. Before the actual data transfer, the address is sent and supports multiple transactions. Smartphones, tablets and high-speed computers are some of the applications mentioned. This paper's authors have also reviewed a few papers can be seen. AXI Master architecture has been mentioned which consists of five different channels (i.e. write data, write address, write response, read data and read address). Master transactions happen using the handshake process and the signals used are VALID/READY. The authors have also mentioned the hardware implementation, in which the data is been transferred from AXI4 Master to the seven segments at variable speeds of 25Mbps, 50Mbps and 100Mbps. In the simulation results, they have designed and executed a master interface which can transfer at the high speed of 1GHz using the Xilinx tool. In conclusion and in future scope, they have mentioned the fact why the choose AXI4 bus protocol. Concluding, the READ and WRITE operation from master to slave requires 100MHz on Xilinx FPGA. This AXI design can be easily used to connect UART, I2C to non-AMBA based processors. Future scope: At present AXI4 protocol is used in application processors such as smartphones, it can be extended to implement the latest version of AXI4 bus protocol: AXI4 LITE.

Few writers [4] have effectively proposed how to implement an AMBA AXI4 data transaction in a SoC bus. AMBA AXI-4 is a plug-and-play IP protocol developed by ARM that defines each bus standard as well as a technology-independent approach for creating, enforcing, and testing bespoke high-integration embedded interfaces. The master sends data to the slave, which is decrypted and read or written to a specific address region on the slave. The AMBA AXI-4 protocol, which was modeled in Verilog hardware description language (HDL), was used to perform data transactions, and the data results for simulation and address read and write operations were displayed using the Verilog compiler simulator (VCS) tool. Around 100MHz is the operational frequency. To accomplish multiple read and write operations, two check cases are run. The module requires 565ns to perform a write operation and it requires 160ns to perform a read operation.

The authors [5] employed system Verilog (SV) to create a verification like environment for the AXI bus protocol, and then used a saleable test bench design to confirm AMBA AXI Protocol performing successful Read and Write operations for incrementing and enhancing the burst feature. System Verilog is used to examine the AXI protocol's functional specification. System Verilog is used to create a verification environment for the AXI protocol. The outcome of memory transaction verification for the AXI protocol, which was modelled in System Verilog and simulated with the help of Mentor Graphics Questa - Sim EDA tool for a data bit size of 32 bits. Currently, the verification study is being carried out on three cases. Case 1: Increase AWLEN, fix AWSIZE, and randomize all values. Case 2: Increase AWSIZE, fix AWLEN, and randomize all values. Case 3: randomize all values and fix AWADDR, ARADDR, AWLEN, and ARLEN. The basic concept is to do write and read transactions in order to augment burst functions. The signals used in each channel of the AXI Protocol are simulated and examined using waveform evaluation. Simulation is done with the help of Mentor Graphics' Questa sim 10.4e tool.

B. Based on Verification

The authors [6] have mentioned on verification of AMBA-AXI protocol the usage of UVM. The verification technique of the structure of AXI which incorporates 5 channels using which memory transactions are made among a master and slave is mainly mentioned on this paper. The authors have used VIP primarily based methodology for the verification procedure. The VIP used here generates complete check cases which shortens SoC verification and increases test coverage. The usage of the VIP the entire take a look at environment is designed and modeled the use of UVM where examine and write transactions are executed at same and one-of-a-kind reminiscence places. After the verification method, it's been analyzed that for write operation BC = sixteen, VC = 13, and BU = eighty-one.25%; study operation BC = 13, VC = 10 and BU = seventy-six.92%. For read and write operations at different places, the bus usage is 66.25% and for examine and write operations from identical region, the bus utilization is ninety-five.45%. subsequently, the channels are efficaciously used during memory transactions from same place. The authors have concluded that the entire verification is achieved the use of code coverage mode analysis this is, the verification is done the use of pseudo random coverage pushed verification wherein it is relevant to complicated designs using UVM.

Some authors [7] presented a design and testbench for the AMBA AXI protocol for SoC integration, which includes work on how to build an AXI bus testbench using device Verilog and coverage of all functional aspects, rating-boarding, and assertions using the proposed and customized integrated verification like environment. The author's main purpose is to utilise Verilog to establish communication between one master and one slave, then verify the customized design with system Verilog. With machine Verilog, the verification environment for the AXI bus has evolved. This environment is organized in a hierarchical layered shape that allows it to be held and reused with different designs underneath verification.

The major goal is to validate the design AXI by using specified inputs. In this research, we present a powerful testbench that can simulate most AXI signal cases, mechanically test all sent data, and do a full coverage analysis during the entire simulation. As a result, the environment can boost its coverage while also reducing verification time.

The AMBA AXI on-chip communication protocol was verified by the authors [8]. The author used System Verilog (SV) to create a testbench for the AXI protocol, where he used a saleable check bench architecture to test the AMBA AXI Protocol for successful Read and Write Operations for incrementing and programming the burst function. The simulation is performed using the Mentor Graphics Questa- Sim EDA application with a statistics bit size of 32 bits. Write address, write data, write response, read address, and read data are the five primary channels that are validated. Every channel has its own set of handshaking indicators, such as Write address, which has the handshaking signals AWVALID and AWREADY, and valid transactions can only take place when both AWVALID and AWREADY are 1. As a result, deal with and manipulate notifications in the write address section. In the Write data segment, information is written based on the control signal that determines the length of the information (AWSIZE, AWLEN), and AWBURST determines whether the burst is wrapping or incrementing. The final signal indicates that the transmission is complete. When the BRESP signal is 1, the writer has finished the verification evaluation in three occurrences. Case 1: Increase AWLEN, maintain AWSIZE, and shuffle all Values. Case 2: Increase AWSIZE, maintain AWLEN, and shuffle all values. Fix AWADDR, ARADDR, AWLEN, and ARLEN and randomize all available values in case 3. They have concentrated on read and write operations with no information loss in this work.

The authors [9] have used a reusable in-constructed verification referred to as Verification IP [Intellectual property] Cores for discount of time in verification and simulation using device Verilog and Mentor graphics Questasim. They have taken 5 form of test instances and have checked the useful verification and the overall performance was calculated based on valid count number, busy remember and bus utilization thing. They've achieved operations i.e., DUT [Device Under Test] to define the test cases and decide the quantity of test cases. This paper includes the Bus architecture of AMBA where the examine and WRITE transactions of AXI the usage of various channels and alerts. Check Suite [includes all test cases for checking the functionality and key features of DUV], digital sequence [bring in arbitrary sequences, design key property is verified], Sequencer [generates and line up the sequence based on functionality check], driver [blocks are driven with corresponding inputs], monitor [keeps an eye on all the data transactions and reviews the result], Protocol Checker [to verify DUV functionality and monitors the responses for all the possible inputs], Bus monitor [links between master-slave and each components functional operation is refined] and coverage Collector [all transactions are accumulated as data] these kind of are the main components are modeled in machine Verilog and implemented in pinnacle module, has been mentioned in this paper used for testbench. This paper aimed to confirm the bus for five various check cases and perform code coverage, additionally by means of the usage of the performance metrics bus performance has been determined.

A multi-slave interface for the AXI bus was devised and implemented by a few authors [10]. The authors have designed and implemented a multi slave interface for the AXI bus in this article. Because a single processor chip can't do all of the computations, the Multi-Processor System on Chip (MPSoC) was developed. This necessitates high data bandwidth and parallelism on-chip communication systems. Furthermore, typical bus protocols only enable one master to access one slave at a time, limiting the system's overall performance. The authors of this study have built multislave and multimaster in the AXI-3 bus, which translates data in bursts of up to 16 transactions. They also implemented two slaves with one master, namely internal memory and register bank. Fixed, increment, and wrap are the three primary types of burst transactions. To check for faults, the Verilog code for the whole Design is compiled in the Modelsim simulator. They used AXI-3, however AXI-4, which is an update to AXI-3, can also be used because it has a high bandwidth and frequency. And it's fast over AXI-3. The design's functioning is verified by developing a test bench, which analyses the waveforms of write and read burst transactions in fixed, incremental, and wrapped modes.

C. Synthesis and Design

The authors [11], have studied the demanding situations faced and the complete experimental analysis within the designing of on-chip communication using AXI-4 protocol. Also, the authors have analyzed the communication complexity and its significance in Network on Chip (NoC) the use of advanced versions of AMBA AXI protocol. Even though the protocol used is advanced version but yet interfacing and registering with multiple masters in many applications. The authors have also furnished with an architecture with good transmission rate, efficient satisfactory of service, much less complexity in routing interfaces and bandwidth challenges for all burst lengths in data transmission. The AXI protocol presents excessive bandwidth and very low latency, for this reason it permits high frequency of operation wherein no complicated bridges are used. AXI-4 has an advantage that it used separate data phases and address control where it helps unaligned data transferring. On a whole the, authors have concluded that AXI4 interface as fully specified, consistent and extensible.

The authors have also designed a actual-time network interface, thinking about master and slave conversation by installing AXI interfaces. So, mainly in a on-chip communication widely uses AXI4 for data transfer protocol where it acts as a perfect data communication.

Few authors [12] used UVM to design and construct an AMBA-AXI4 bus-based system on chip VIP protocol. The work of the author was presented on paper. The ARM Advanced Microcontroller Bus Architecture is an on-chip connection architecture used in System-on-a-Chip designs to connect and manage functional units. The Advanced Extensible Interface (AXI), which is a parallel high-performance, high-frequency, synchronous, multi-master, multi-slave communication interface primarily developed for on-chip communication, is included in the ARM Advanced Microcontroller Bus Architecture 3 (AXI3) and 4 (AXI4) specifications. The author has created the AXI4 Slave Interface and implemented it using Verilog RTL Coding in this article. This paper introduces the design environment and how to construct the AXI bus verification environment using the universal verification technique. The proposed integrated verification environment implements functional coverage, code coverage, score-boarding, and assertions. The author designed a 100MHz AXI 4 slave with a 10ns clock cycle and a maximum of 1024 data transfers per burst. The five channels between master and slave in AXI-4 are the Address Write Channel (AW Channel), Write Data Channel (W Channel), Write Response Channel (B Channel), Address Read Channel (AR Channel), and Read Data Channel (R Channel). Multiple outstanding Transactions (can boost efficiency and speed. Out-Of-Order (OOO) transactions are used in the AXI-4 protocol to reduce transaction latency. All five channels have been confirmed, including write and read transactions for all possible burst types and sizes, multiple different outstanding addresses transactions, and out-of-order transactions.

Authors [13] have studied the complexity in designing an SoC has been increasing enormously as there are addition of different blocks and IPs integrated on a chip. This has led to difficulty in verifying the properties of on-chip communication. To overcome such difficulty, they have proposed a rule based synthesizable AMBA AXI Protocol Checker inclusive of 44 rules: 31 rules related to Master, 12 rules related to Slave and 1 Slave rule by default. This Checker works with a frequency of 242MHz, critical path is 4.13ns, 70.7K gate counts under TSMC 180nm CMOS 1P6M Technology. They have used Synopsys Verification IP for verifying the AXI checker. They have mentioned few drawbacks of DUT [Device Under Test] which is non-synthesizable and the introduction of AMBA AXI Checker to overcome them. This paper includes the architecture of the Checker, which could be our main focus and also contrast between AMBA AXI 3.0 with AMBA AHB 2.0. Some of the highlights of the architectural blocks are: Protocol Checker – It takes input as the bus signals of AXI and outputs the error signal of 44bits. Each bit corresponds to a rule. If any violation occurred it outputs the corresponding ID number of Master or Slave. Also, briefs about checker's sub-modules.; Configuration Register – Some of the parameters such as mask, protocol checker enable are set. Mask is used to disable the unrelated rules only when only limited rules have to be verified and enable is used to enable or disable the checker.; Error Reference Table [ERT] – Displays each and every error occurred. In the table, rows [44 rules = 44 columns] specify each error and columns [16 Masters + 16 Slaves = 32 columns] specify the Master or Slave ID number. The verification strategy includes the generation of bus signals and Synopsys VIP using RTL trigger. The experimental results show that the bus monitor VIP and AXI Protocol Checker are set side by side and the contrasts can be observed. Conclusion, the proposed verification strategy can summarize the total errors occurred and furthermore for displaying details of error information a GUI software analyser is developed.

For software analysis they have designed a transaction-based techniques. At the earliest, the authors [14] have utilized AMBA ACE-Lite architecture for analysis and physical hardware design. The AMBA AXI4 Bus Interconnect was previously used for designing a hardware system, but it didn't match practical design standards, thus the suggested AMBA ace-lite architecture delivered the desired outcomes with low complexity. Several functional blocks on the chip can be readily integrated using the AMBA ace-lite design for hardware system, in a similar manner to how transaction-based simulation models are designed. The major goal of this study is to present a way for dealing with complexity simplification challenges in hardware design by incorporating the benefits of transaction-based verification (TBV). In the SoC design procedure, the authors also established Transaction Level Modelling (TLM) over Register Transfer Level (RTL). Using untimed and timed models, TLM abstractions may now span SoC design processes ranging from early software development to architectural analysis and functional verification.

The authors [15] present a rule-based synthesizable AMBA AXI protocol checker in this study. The AXI protocol checker has 44 criteria for ensuring the accuracy of on-chip communication features. The Synopsys VIP (Verification IP) is used in the verification approach to verify the AXI protocol checker. The AMBA AXI4 IP protocol from ARM provides a bus specification as well as a technology-agnostic approach for developing, implementing, and testing customised high-integration embedded interfaces. The master is assumed to have provided the data to be read or written to the slave, and it is read or written to a specific slave address location through decoder. The slave was modelled in Verilog with a 100MHz working frequency, and the simulation results were displayed using the Modelsim tool.

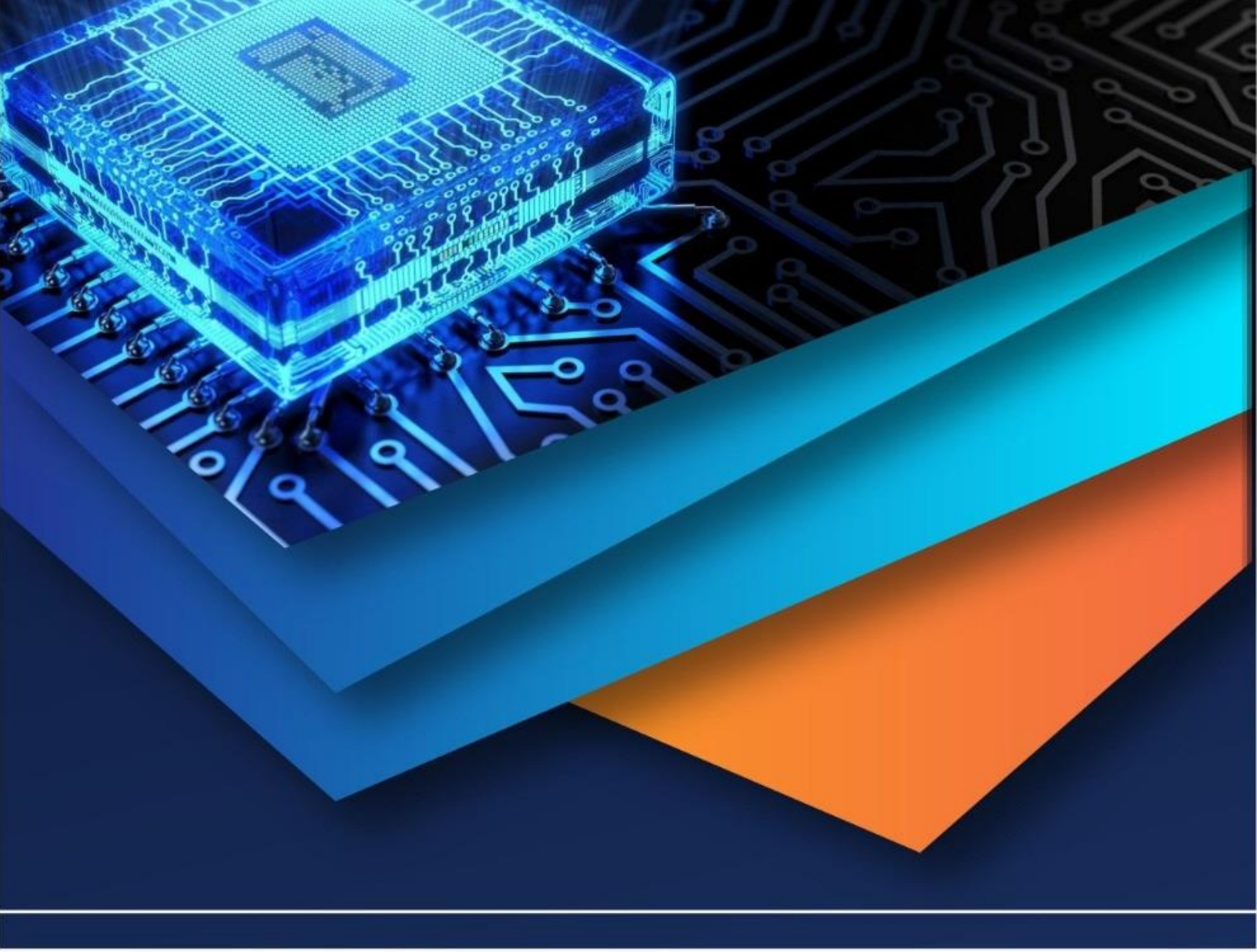
It took 160ns to execute a single read operation and 565ns to perform a single write operation. The AXI protocol checker has a chip cost of 70.7K gate counts and a critical path of 4.13 ns using TSMC 0.18um CMOS 1P6M Technology (approximately 242 MHz).
Certain authors [16] have proposed a 32-bit RISC-V AXI4-lite bus protocol-based Microcontroller with a 10-bit SAR ADC. Using the open-source RISC-V instruction set, the authors built and developed a completely synthesized 32-bit microcontroller in 130nm CMOS technology. For connectivity, it is connected to AXI4-Lite and APB buses. The microcontroller utilized here has a 10-bit SAR ADC, a 12-bit DAC, an 8-bit GPIO module, a 4kB RAM, an SPI AXI slave interface for output verification, and an SPI APB slave interface for validating the APB bridge's right behavior. A RISC-V and SPI AXI master interface controls all peripherals and is used to program the device and inspect data flowing through all slaves.. Density of total power for this RISC-V microprocessor is reported to be 167W/MHz, with a decreased footprint of 798µm×484µm.The proposed architecture shows the interconnection between the RISC-V, the SPI AXI master, and all the peripherals attached to the AXI4-Lite and APB buses, explaining details of the implementation. Power and area results show that a reduced RISC-V architecture can be used to replace ARM-MO based microcontrollers with similar performance. The mRISC-V provides the path for future implementations for particular and general applications in the realm of IoT with open-source devices, thanks to the developing RISC-V community and the current toolchain and software around this new instruction set.

IV. CONCLUSIONS

Efficient communication with high performance in a SoC is always adds a great advantage in SoC designs. In simple SoC application design, AXI4-lite is more efficient with highest bus utilization for functional blocks communication. Currently AXI4 protocol being used in application processors of smart phones can be extended to implement latest versions such as AXI4-lite. We are using UVM environment with system Verilog for design verification and Cadence tools for synthesizing, physical design, static time analysis and power analysis. In future work, we are planning to implement adaptive burst length technique using some features of AXI-Stream protocol, that is multiple data streams using same set of wires.

REFERENCES

[1] Mariem Makni, Mouna Baklouti, Smail Niar and Mohamed Abid, “Performance exploration of AMBA AXI-4 Bus Protocols for Wireless Sensor Networks”, 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications, DOI 10.1109/AICCSA.2017.26
[2] A. Sainath Chaithanya, Sameera Sulthana, B. Yamuna and Ch Haritha, “Design of AMBA AXI4-Lite for Effective Read/Write Transactions with a Customized Memory", International Journal on Emerging Technologies 11(1): 396-402(2020).
[3] Shaila S Math, Manjula R. B, S.S. Manvi, Paul Kaunds, “Data Transactions on System-on-chip Bus Using AXI4 Protocol”, 2011 International Conference on Recent Advancements in Electrical, Electronics and Control Engineering.
[4] P. Naveen Kalyan and K. Jaya Swaroop, “AMBA-AXI Protocol Verification by Using UVM”, International Journal of Electronics and Communication Engineering and Technology (IJECET), Volume 7, Issue4, July-August 2016.
[5] Nikhil Gaikwad & Vijay.N.Patil. “Verification of AMBA AXI on-chip Communication Protocol”, Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018.
[6] Pradeep S R & Laxmi C, “Design and Verification Environment for AMBA AXI Protocol for SoC Integration”, IRJET: International Journal of Research in Engineering and Technology, Volume: 03 Special Issue: 03 | May-2014.
[7] Neethika Tidala, “High Performance Network On Chip using AXI4 protocol Interface on an FPGA”, Proceedings of the 2nd International conference on Electronics, Communication and Aerospace Technology (ICECA 2018).*FLEXChip Signal Processor (MC68175/D)*, Motorola, 1996.
[8] Murali M, Umadevi S, Sakthivel S M, “Verification IP for AMBA AXI Protocol using System Verilog”, International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 17 (2017) pp. 6534-6541 © Research India Publications.
[9] Chien-Hung Chen, Jiun-Cheng Ju, and Ing-Jer Huang, “A Synthesizable AXI Protocol Checker for SoC Integration”, ISOCC 2010.
[10] Syeda Nasiha, Veeresh Pujari, Dr. Baswaraj Gadgay, “Design and Implementation of AMBA AXI 4.0 Master for HighSpeed Performance SoC”, IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 3 Issue 7, July 2016.
[11] Narra Venkata Krishna, “Design and Development of AMBA-AXI4Bus Based System on Chip VIP Protocol Using UVM", International Journal of Innovative Research in Science, Engineering and Technology. 6, Issue 4, April 2017.
[12] Nikhil Gaikwad and Vijay.N.Patil,“Verification of AMBA AXI on-chip Communication Protocol", 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA),978-1-5386-5257.
[13] Anitha.H.T and Dr.Nataraj.K.R, “IMPLEMENTATION OF MULTI-SLAVE INTERFACE FOR AXI BUS", IEEE
[14] Chiranjeet Kumar & Dr. M. Gurunadha Babu, “A Design of AMBA AXI4-Lite ACE Interconnect Protocol for Transaction based SoC Design Techniques Integration”, International Journal of Engineering And Computer Science, Volume 6, Issue 6, June 2017.
[15] M. Siva Prasad Reddy, B. Babu Rajesh, Tvs Gowtham Prasad, “A Synthesizable Design of AMBA-AXI Protocol for SoC Integration”, International Journal of Engineering Inventions, www.ijeijournal.com, Volume 1, Issue 3, September 2012.
[16] Ckristian Duran, Luis Rueda D, Giovanny Castillo, Anderson Agudelo, Camilo Rojas, Luis Chaparro, “A 32-bit RISC-V AXI4-lite bus based Microcontroller with 10-bit SAR ADC”, VII Latin American Symposium on Circuits and Systems (LASCAS) 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)