iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VIII **Month of publication:** Aug 2023

DOI: 55446

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 11 Issue VIII Aug 2023- Available at www.ijraset.com

Design and Implementation of Advanced Extensible Interface using Verilog

Vishwas B R[1], Dr. Sowmya K B[2]
Department of Electronics and Communication, R V College of Engineering

Abstract: ***Advanced eXtensible Interface, is an interface protocol defined by ARM as part of the AMBA standard. They comprise of AXI4, AXI4 Lite, AXI4 Stream. The AXI specification describes a point-to-point protocol between two interfaces, a master and a slave. The AXI protocol uses 5 channels, 2 for read transactions and 3 for write transactions. One of the key features of AXI4 is its support for burst transactions, which allows for efficient transfer of multiple data items in a single transaction, enhancing data throughput. AXI4-Lite is designed to provide a lightweight and efficient interface for communication between a master device and simpler peripheral devices or memory-mapped registers in a digital system. AXI4-Stream is developed for high-throughput, unidirectional data transfer between different components in a digital system. The design is done in Verilog.***
Keywords: AMBA, ARM, AXI, AXI4, AXI4-Lite, AXI4 Stream, VLSI, SoC

I. INTRODUCTION

Protocols in VLSI refer to standardized communication and data exchange mechanisms that facilitate interactions between different components, modules, or IP blocks within an integrated circuit or System-on-Chip. These protocols play a vital role in ensuring efficient, reliable, and standardized data transfer within digital systems. The AXI protocol is a fundamental communication standard in the VLSI and semiconductor industries. Developed by ARM, the AXI protocol plays a crucial role in facilitating efficient data transfer and communication between [1] various on-chip components within modern digital systems.

AXI, which means Advanced eXtensible Interface, is an interface protocol defined by ARM as part of the AMBA standard. There are 3 types of AXI4-Interfaces.

- AXI4 (Full AXI4): For high-performance memory-mapped [2] requirements.
- AXI4-Lite: For simple, low-throughput memory-mapped communication.
- AXI4-Stream: For high-speed streaming data.

The AXI protocol defines 5 channels. 2 are used for Read transactions (read address, read data) and 3 are used for Write transactions (write address, write data, write response). The AXI specification describes a point-to-point protocol [3] between two interfaces: a master and a slave. The following Figure. 1 shows the five main channels that each AXI interface uses for communication.

Fig. 1 AXI Channels

II. DESIGN AND IMPLEMENTATION

The AXI protocol is transactions-based and defines five independent channels.

1) Write request, which has signal names beginning with AW.
2) Write data, which has signal names beginning with W.
3) Write response, which has signal names beginning with B.
4) Read request, which has signal names beginning with AR.
5) Read data, which has signal names beginning with R.

A request channel carries control information that describes the nature of the data [4] to be transferred. This is known as a request. The data is transferred between Manager and Subordinate using either:

a) A write data channel to transfer data from the Manager to the Subordinate [5]. In a write transaction, the Subordinate uses the write response channel to signal the completion of the transfer to the Manager [6].
b) A read data channel to transfer data from the Subordinate to the Manager [7].

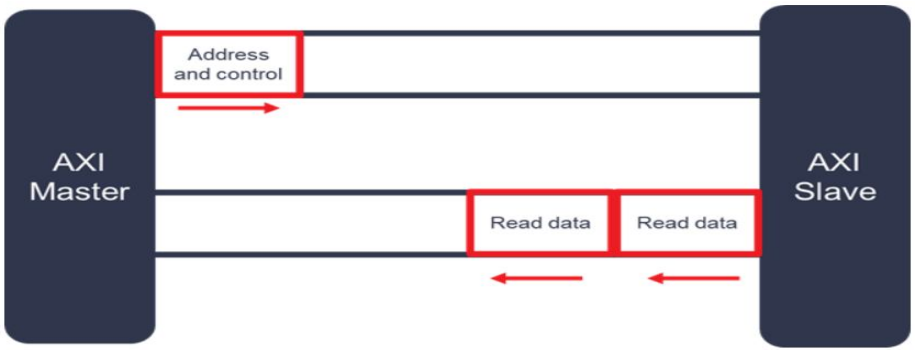


Fig. 2 AXI4 Read transaction

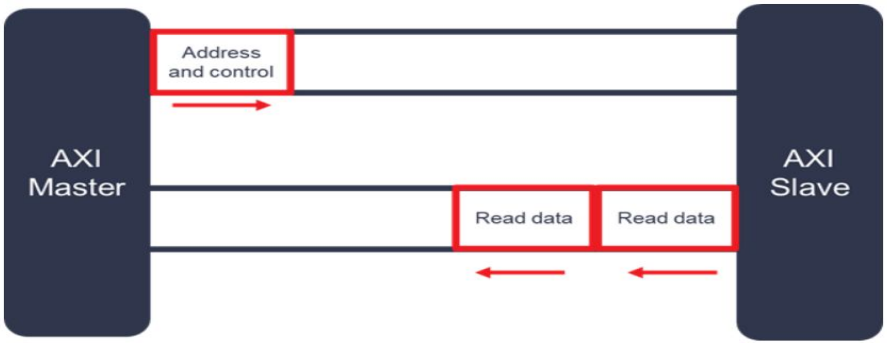


Fig. 3 AXI4 Write transaction

A. AXI4-Lite Protocol

AXI4 Lite, is a simplified variant of the AXI protocol developed by ARM. It is designed to provide a lightweight and efficient interface for communication between a master device and simpler peripheral devices [8] or memory-mapped registers in a digital system. AXI4-Lite focuses on minimizing complexity and latency for basic read and write [9] transactions. The AXI4-Lite interface consists of five channels: Read Address, Read Data, Write Address, Write Data, and Write Response. AXI4-Lite is incapable of burst transfers.

Some of the features of AXI4-Lite include:

1) No burst transaction is enabled or supported
2) Single address
3) Single data
4) Very small size
5) The AXI interconnect is automatically generated
6) Data width can be 32 bits or 64 bits

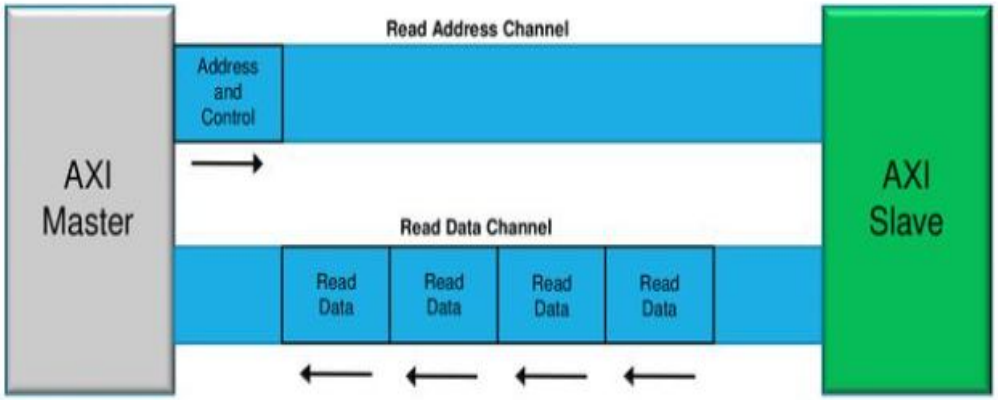


Fig. 4 AXI4-Lite Read transaction

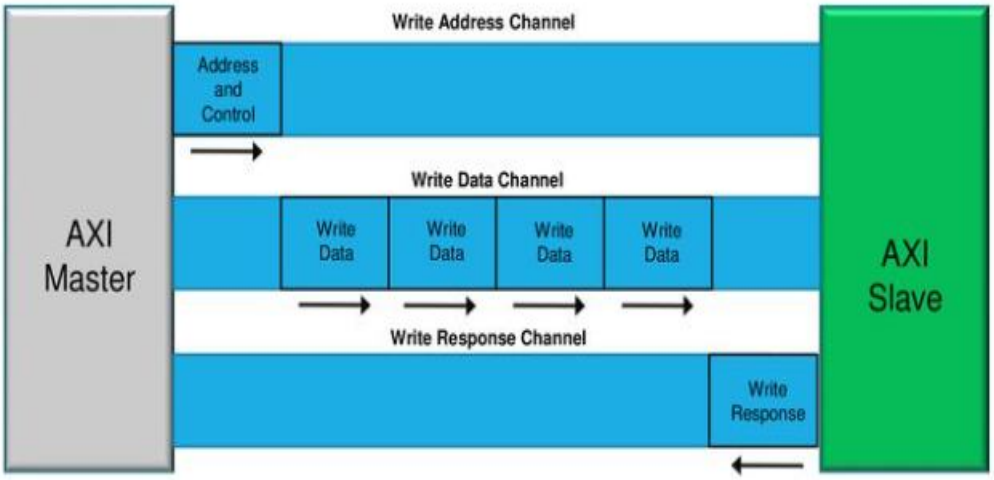


Fig. 5 AXI4-Lite Write transaction

B. AXI4 Stream

AXI4-Stream is designed for scenarios where a continuous stream of data needs to be moved from a source to a destination without the complexities of memory addressing [10] and control. The AXI4-Stream protocol is used as a standard interface to connect components that wish to exchange data [11]. The interface can be used to connect a single master [14], that generates data, to a single slave [12], that receives data. The protocol can also be used when connecting larger numbers of master [15] and slave components. The protocol supports multiple data streams using the same set of shared wires [13], allowing a generic interconnect to be constructed that can perform upsizing, downsizing [16] and routing operations.

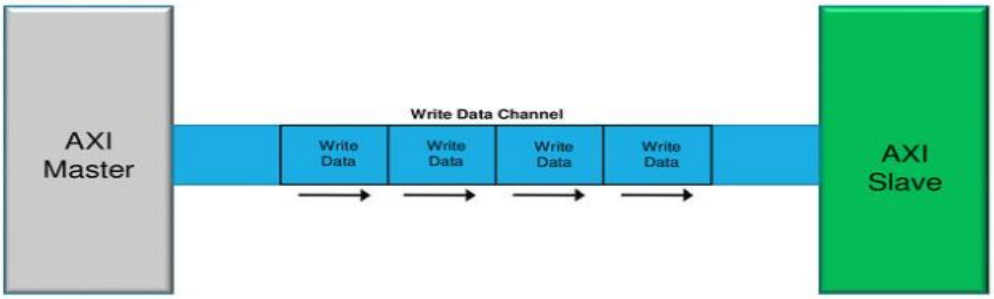


Fig. 6 AXI4 Stream

III. METHODOLOGY

The design methodology is as follows:

1) Clearly define the requirements for your AXI4 interface. Determine the data width, address width, number of channels, and any specific functionality you need (e.g., read, write, burst transfers).
2) Design the architecture of your AXI4 interface based on the defined requirements. Decide whether it will be a master, slave, or both, and whether it will support burst transfers or other advanced features.
3) Write Verilog RTL code to implement your AXI4 interface, depending on the design requirements.
4) Use Verilog simulation tools (e.g., ModelSim, XSIM) to simulate the AXI4 design.
5) Once AXI4 design is functionally correct and meets the specifications, it can be synthesized using a synthesis tool such as Xilinx Vivado.

IV. RESULTS AND ANALYSIS

The results comprise of the simulation of the protocols discussed. The simulation is carried out using Xilinx Vivado. The RTL code as well as testbench is written using Verilog.

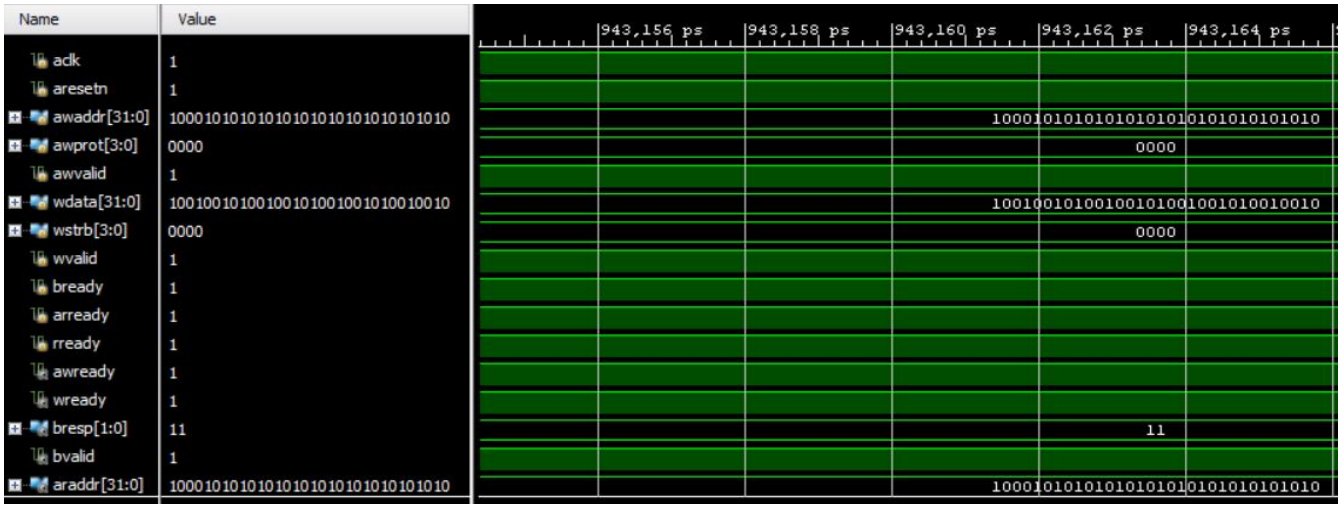


Fig. 7 AXI4 Protocol simulation

The simulation of AXI4 is depicted in Figure 7. The inputs are given and the outputs are observed accordingly and the functionality is verified. AXI4 protocol supports burst transactions.

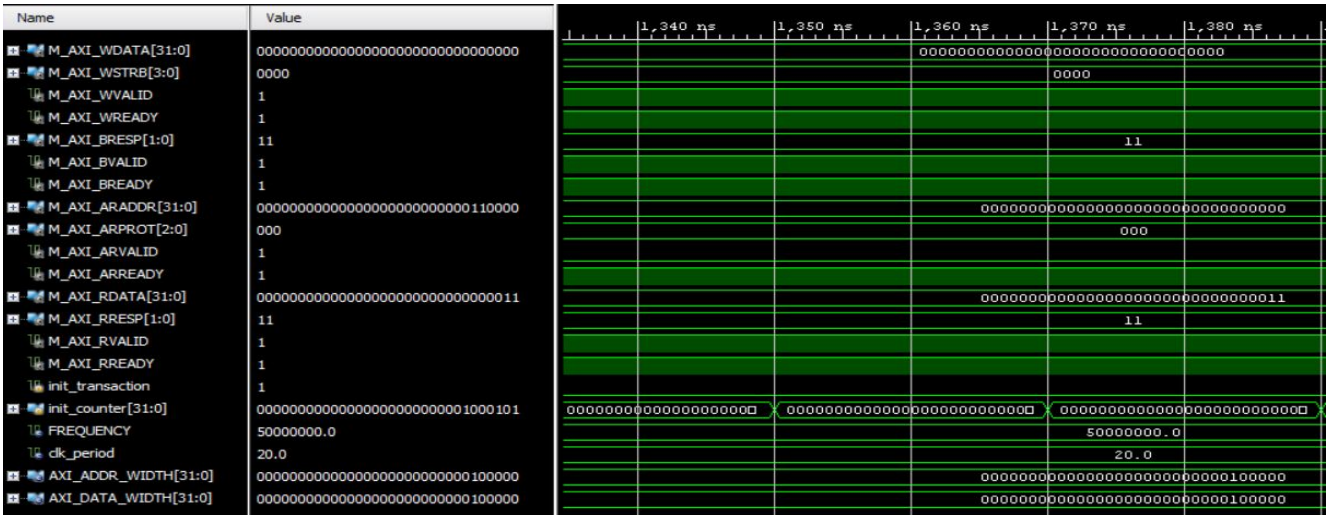


Fig. 8 AXI4 Lite simulation

The simulation of AXI4-Lite master is depicted in Figure 8. The inputs are given and the outputs are observed accordingly and the functionality is verified. AXI4-Lite is a similar version to AXI4 protocol. The only difference is that AXI4-Lite does not support burst transaction.

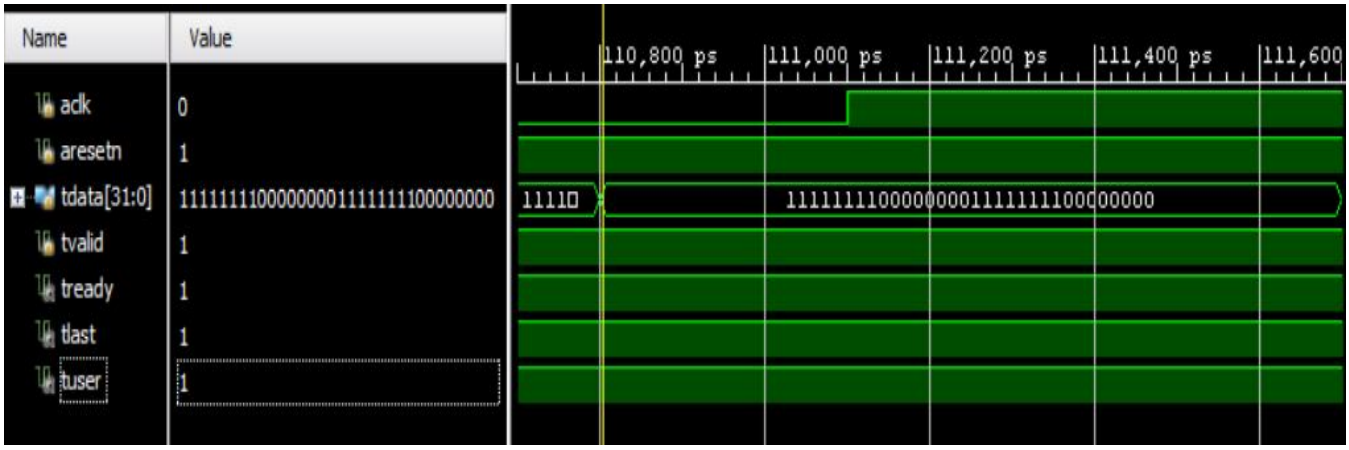


Fig. 9 AXI4 Stream simulation

The simulation of AXI4-Stream is depicted in Figure 9. The inputs are given and the outputs are observed accordingly and the functionality is verified. AXI4 Stream is the simplest and less complex protocol compared to the other two.

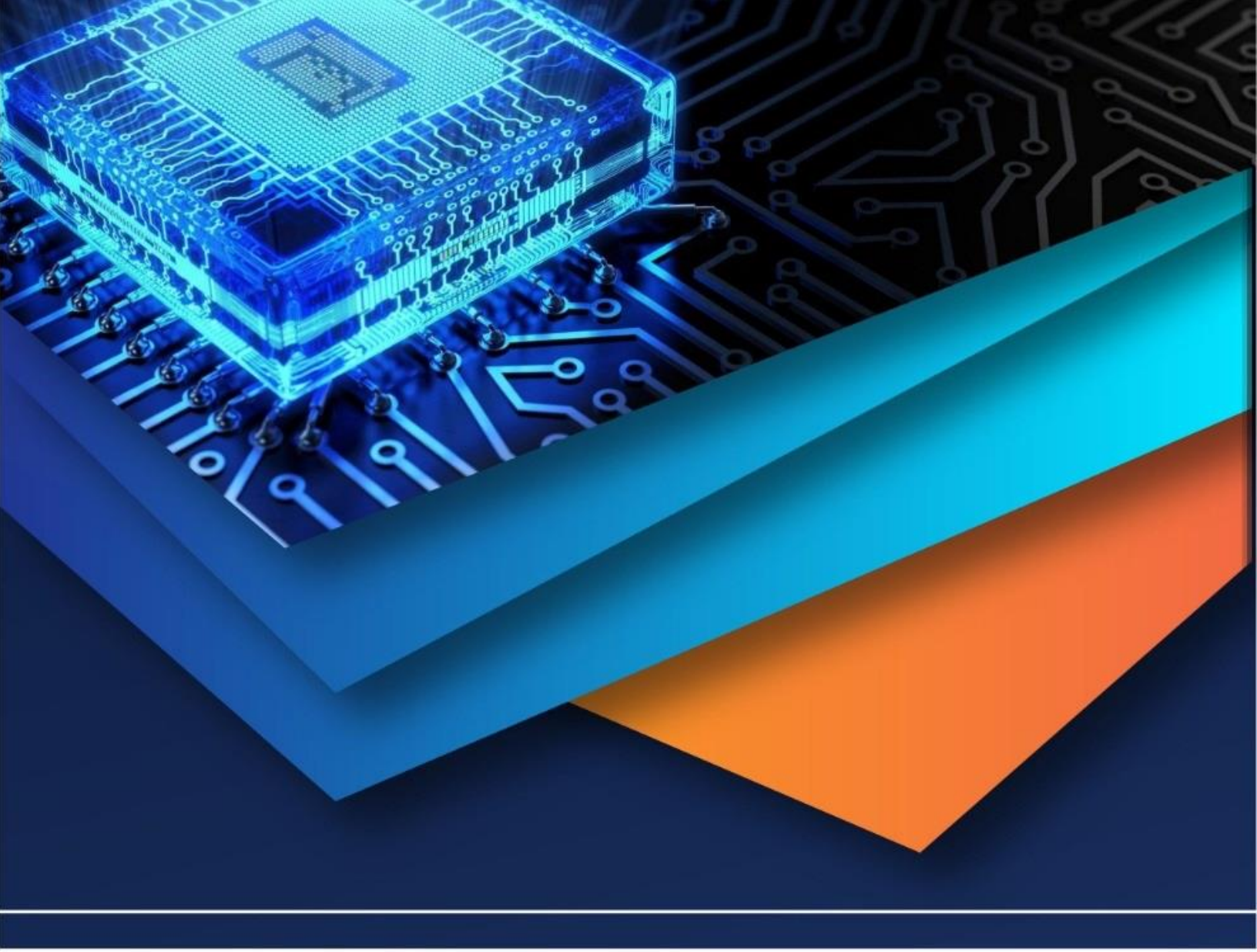
V. CONCLUSIONS

The AXI4 protocol has branched into AXI4, AXI4 Lite, AXI4 Stream. Xilinx Vivado is used to design and implement the same. The AXI4 protocol serves as a standardized and efficient interface for communication between various components within an SoC or digital system. Its ability to support features like burst transactions, streaming, and multiple outstanding transactions makes it invaluable for achieving high performance and scalability. AXI4 has a total on-chip power of 0.07 W. It comprises of a total of 119 nets, 196 input/output ports and 117 cells. AXI4-Lite has a total on-chip power of 4.35 W. It comprises of a total of 160 nets, 155 input/output ports and 150 cells. AXI4 Stream It has a total on-chip power of 2.113 W. It comprises of a total of 41 nets and 37 cells. The code and testbench is written using Verilog. The use of Verilog for RTL design provides flexibility and compatibility with industry-standard design practices.

REFERENCES

[1] P. Subramanyan, B.-Y. Huang, Y. Vizel, A. Gupta, and S. Malik, “Template-based parameterized synthesis of uniform instruction-level abstractions for soc verification,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 37, no. 8, pp. 1692–1705, 2018.
[2] C. Yu, W. Brown, D. Liu, A. Rossi, and M. Ciesielski, “Formal verification of arithmetic circuits by function extraction,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 35, no. 12, pp. 2131–2142, 2016.
[3] J. Kumar, Y. Miyasaka, A. Srivastava, and M. Fujita, “Formal verification of integer multiplier circuits using binary decision diagrams,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 42, no. 4, pp. 1365–1378, 2023.
[4] S. El-Ashry, M. Khamis, H. Ibrahim, A. Shalaby, M. Abdelsalam, and M. W. El Kharashi, “On error injection for noc platforms: A uvm-based generic verification environment,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 5, pp. 1137–1150, 2020.
[5] V. Melikyan, S. Harutyunyan, A. Kirakosyan, and T. Kaplanyan, “Uvm verification ip for axi,” in 2021 IEEE East-West Design Test Symposium (EWDTS), 2021, pp. 1–4.
[6] N. K. P, D. V, A. M, S. K. R, and E. S, “Design and verification of amba axi3 protocol for high-speed communication,” in 2022 Smart Technologies, Communication and Robotics (STCR), 2022, pp. 1–5.
[7] P. Dwivedi, N. Mishra, and A. Singh-Rajput, “Assertion functional coverage driven verification of amba advance peripheral bus protocol using system verilog,” in2021International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), 2021
[8] T. Strauch, “Dynamic inside-out verification using inverse transactions in tlm,” in2018 Forum on Specification Design Languages (FDL), 2018, pp. 5–16.
[9] H. H. Ardakani, A. M. Gharehbaghi, and S. Hessabi, “A performance and functional assertion-based verification methodology at transaction-level,” in2007 InternatIonal Conference on Microelectronics, 2007, pp. 133–136.
[10] M. Siegel, “Achieving earlier verification closure using advanced formal verification,” in Formal Methods in Computer Aided Design, 2010, pp. 275–275.
[11] L. Duan, Y. Hu, H. Liu, W. Feng, and J. Gan, “An efficient formal verification method in i/o multiplexing module based on vc formal cc,” in 2020IEEE3rd International Conference on Electronics and Communication Engineering (ICECE),2020, pp. 112–116.

[12] M. W. Anwar, M. Rashid, F. Azam, A. Naeem, M. Kashif, and W. H. Butt, "A unified model-based framework for the simplified execution of static and dynamic assertion-based verification," IEEE Access, vol. 8, pp. 104 407–104 431, 2020.
[13] P. Gurha and R. R. Khandelwal, "SystemVerilog assertion-based verification of amba-ahb," in 2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE), 2016, pp. 641–645.
[14] H. Sangani and U. Mehta, "UVM based verification of read and write transactions in axi4-lite protocol," in 2022 IEEE Region 10 Symposium (TENSYMP), 2022, pp. 1–5. doi: 10.1109/TENSYMP54529.2022.9864552.
[15] M. P. Deepu and R. Dhanabal, "Validation of transactions in axi protocol using system verilog," in 2017 International conference on Microelectronic Devices, Circuits and Systems (ICMDCS), 2017, pp. 1–4. doi: 10.1109/ICMDCS.2017.8211605.
[16] G. Mahesh and S. M. Sakthivel, "Verification of memory transactions in axi protocol using system verilog approach," in 2015 International Conference on Communications and Signal Processing (ICCSP), 2015, pp. 0860–0864. doi: 10.1109/ICCSP.2015.7322617



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY