iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** IV **Month of publication:** April 2023

DOI: https://doi.org/10.22214/ijraset.2023.51179

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 IV Apr 2023- Available at www.ijraset.com

RTL Implementation of AMBA AHB Protocol using Verilog

Sri Phanindra Perali[1], Nithin Krishna Madadi[2], Rohith Reddy[3]
Department of EECE, GITAM (Deemed to be University), Hyderabad, Telangana, India

Abstract: ***The enactment of a computer system heavily depends on the design of its bus interconnect. A poorly designed system bus can hinder the transmission of instructions and data between the processor and memory or between peripheral devices and memory. To address these challenges, the Advanced Microcontroller Bus Architecture (AMBA) provides an open standard for connecting and managing functional blocks in a System-on-Chip (SoC). This architecture allows for developing multi-processor designs with many controllers and peripherals while ensuring the system is designed correctly the first time. Furthermore, the AMBA specifications are royalty-free and platform-independent. They can be used with any processor architecture. The project will provide an RTL view and an extracted design summary of the AMBA AHB module at the system-on-chip level. The AMBA High-performance Bus (AHB) is another part of the AMBA family of conventions. The AHB is designed to support high-performance, high-clock system modules and serves as the system's high-performance backbone bus. The AHB enables the efficient connection of low-power peripheral functions to processors, on-chip memories, and external off-chip memory interfaces. All signal transitions in the AHB relate only to the rising clock edge, allowing AHB peripherals to be easily integrated into any design flow. The project also describes the AMBA AHB design and implementation using Verilog, with read/write operations implemented using the Xilinx simulator.***

I. INTRODUCTION

In today's complex System-on-Chip (SoC) designs, interconnect protocols play a critical role in ensuring the correct functioning of the system. The Advanced Microcontroller Bus Architecture (AMBA) family of standards provides an open and standardised way to connect and manage functional blocks in an SoC design. One of the key components of this family of standards is the AMBA AHB (Advanced High-performance Bus) protocol, which is specifically designed to support high-performance, high-clock system modules.

The AMBA AHB protocol serves as the backbone bus of the system. It efficiently connects low-power peripheral functions to processors, on-chip memories, and external off-chip memory interfaces. Its high-performance capabilities make it suitable for various applications, from high-performance computing to embedded systems.

One of the critical features of the AMBA AHB protocol is that all signal transitions relate only to the rising clock edge. This makes integrating AHB peripherals into any design flow easy and ensures the system operates correctly. In addition, the protocol provides a wide range of features, including burst transfers, split transactions, and pipelining, which help to optimise system performance and ensure efficient data transfer. This paper provides an overview of the AMBA AHB protocol, RTL view, its key features, and its benefits for SoC design. We also discuss its use in various applications and highlight some of the challenges and opportunities associated with its implementation. By the end of this paper, readers will better understand the AMBA AHB protocol and its role in enabling high-performance SoC designs.

II. LITERATURE REVIEW

The AMBA (Advanced Microcontroller Bus Architecture) AHB (Advanced High-performance Bus) protocol is a widely used on-chip bus protocol in system-on-chip (SoC) designs. ARM Holdings developed it to provide a high-performance and scalable interface between different components in an SoC. In this literature review, we will examine some essential research papers related to the AMBA AHB protocol.

"A Method [1] for Verifying AHB-based System-on-Chip Designs" by J. Lee and J. Lee (2007) This paper proposes a method for verifying AHB-based SoC designs using the assertion-based verification (ABV) technique. The authors use System Verilog Assertions (SVA) to describe the expected behaviour of the design and to check if the design conforms to the AHB protocol specification. The proposed method is validated on several AHB-based designs, and the results show that it effectively detects protocol violations.

"An Automated Framework [2] for Checking the Conformance of AHB Protocol Implementations" by C. Peng and M. Hsiao (2010) This paper presents an automated framework for checking the conformance of AHB protocol implementations. The framework uses the AHB protocol specification and a set of formal rules to generate test cases that can be used to verify the performance. The authors demonstrate the framework's effectiveness on several AHB-based designs.

"A Comparative Study [3] of the Performance of AMBA AHB and AXI Interconnects" by M. Alam et al. (2016) This paper presents a comparative study of the performance of AMBA AHB, and AXI interconnects in a multi-core system. The authors use simulation-based experiments to evaluate the performance of both interconnects in terms of latency, throughput, and power consumption. The results show that AXI is generally faster than AHB but more power-efficient.

"AHB Performance [4] Optimization Using Burst-Mode Transactions" by S. Kumar and S. Kumar (2018) This paper proposes a technique for optimising the performance of the AHB protocol using burst-mode transactions. The authors modify the AHB protocol to support burst-mode transactions and demonstrate the technique's effectiveness on several benchmark designs. The results show that burst-mode transactions can significantly improve the performance of the AHB protocol.

"Formal Verification [5] of AMBA AHB Protocol Using SystemVerilog Assertions" by S. Sharma et al. (2020) This paper presents a formal verification approach for the AMBA AHB protocol using SystemVerilog Assertions (SVA). The authors develop a set of SVA properties to describe the protocol specification and use a formal verification tool to check if the properties are satisfied by design. The approach is validated on several AHB-based designs, and the results show that it effectively detects protocol violations.

III. AMBA-BASED MICROCONTROLLER: A HIGH-PERFORMANCE AND SCALABLE SOLUTION FOR SOC INTEGRATION

Fig.1 shows the AMBA-based microcontroller. The architecture of an AMBA-based microcontroller consists of three main components - the processor, the memory, and the peripherals. The processor is usually a 32-bit RISC processor designed to work with the AMBA AHB or APB bus protocol. The memory can be a combination of SRAM, ROM, and flash memory, and it is used to store program code and data. The peripherals are the input/output devices used to communicate with the outside world, such as UART, SPI, I2C, GPIO, and timers.

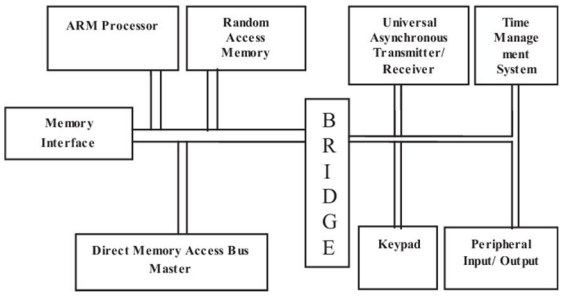


Fig 1: AMBA-based microcontroller

The AMBA-based microcontroller architecture provides a high-performance and scalable interface between the processor, memory, and peripherals. The AMBA AHB protocol is used for high-speed data transfer between the processor and the memory. In contrast, the AMBA APB protocol is used for low-speed data transfer between the processor and the peripherals. The AMBA bus architecture also supports multiple masters and slaves, which allows for the efficient sharing of resources between different components in the system.

Regarding design considerations, the microcontroller architecture should be optimised for power consumption, size, and cost. Power consumption is an essential consideration for battery-powered devices, and the microcontroller should be designed to minimise power usage while still providing the required functionality. The size of the microcontroller should be small enough to fit into the target device, while the cost should be kept low to make the device affordable.

The AMBA AHB (Advanced High-Performance Bus) bus system consists of four main components:

1) *AHB Master:* An AHB master is a device that initiates data transactions on the AHB bus. A processor or DMA controller can read or write data from/to a memory or a peripheral device.
2) *AHB Slave:* An AHB slave is a device that responds to transactions initiated by an AHB master. It can be a memory or a peripheral device that stores or receives data.

3) *AHB Bus Matrix:* The AHB Bus Matrix is the central component of the AHB bus system. It provides a connection between the AHB masters and AHB slaves, allowing multiple masters and slaves to communicate with each other simultaneously. The bus matrix also provides arbitration between the masters to ensure that only one master can access the bus at any time.
4) *AHB Interconnect:* The AHB interconnect is a network of wires and switches that connect the AHB masters, AHB slaves, and the AHB bus matrix. It provides a high-speed data path between the different components of the system and ensures that data is transferred efficiently.

In addition to these components, the AHB bus system includes various other signals, such as control signals, address signals, and data signals, which control the data transfer between the AHB masters and AHB slaves.
The AMBA AHB bus system provides a high-performance and efficient interconnect between different components in a system, allowing for efficient communication and data transfer.
Overall, the architecture of an AMBA-based microcontroller provides a high-performance and scalable solution for SoC integration, allowing designers to integrate multiple components into a single system efficiently.

IV. THE AMBA AHB PROTOCOL: AN INSIGHT INTO ITS WORKING AND ADVANTAGES

The AMBA AHB (Advanced High-Performance Bus) is a bus protocol used for high-performance interconnects in complex system-on-chip (SoC) designs. The working of the AMBA AHB can be explained as follows:

1) *Bus Transactions:* An AHB bus transaction is initiated by an AHB master that wants to access memory or a peripheral device. The transaction includes the address of the memory or peripheral, the type of transaction (read or write), and the data to be read or written.
2) *Bus Matrix:* The AHB bus matrix is responsible for arbitrating between multiple AHB masters that want to access the bus simultaneously. It determines which master gets access to the bus and for how long. The bus matrix also performs the necessary protocol conversions between the system's AHB and other bus protocols.
3) *Address Decoding:* When an AHB master initiates a transaction, the address decoder in the AHB slave decodes the address and selects the appropriate memory or peripheral device accessed.
4) *Data Transfer:* Once the AHB slave has been selected, the data transfer occurs between the AHB master and the AHB slave. The data is transferred over the AHB bus and is buffered by the AHB interconnect to ensure that the data is delivered to the intended destination.
5) *Control Signals:* The AMBA AHB uses various control signals to manage data transfer. These signals include HTRANS, which indicates the type of transaction (nonsequential, sequential, or idle), HREADY, which indicates when the AHB slave is ready to accept data and HSEL, which selects the AHB slave that is being accessed.

The AMBA AHB (Advanced High-Performance Bus) protocol offers several advantages for system-on-chip (SoC) designs. Here are some key advantages of the AMBA AHB protocol:

a) *High-performance Interconnect:* The AMBA AHB protocol is designed to provide a high-performance interconnect for efficient communication and data transfer between different system parts. It supports high-bandwidth data transfer and can handle multiple transactions simultaneously, making it suitable for complex SoC designs.
b) *Scalability:* The AMBA AHB protocol is highly scalable and can support multiple masters and slaves, efficiently sharing resources and peripherals. It can also support different types of transactions, making it suitable for a wide range of SoC designs.
c) *Flexibility:* The AMBA AHB protocol supports different types of bus transactions, including non-sequential, sequential, and burst transactions, enabling efficient communication and data transfer between multiple masters and slaves.
d) *Low Power Consumption:* The AMBA AHB protocol uses a burst transfer mode to minimise the number of accesses to the bus and reduce power consumption. It also supports low-power modes for individual masters and slaves.
e) *Ease of Integration:* Many vendors widely use and support the AMBA AHB protocol, making it easy to integrate into SoC designs. It also supports seamless integration with other AMBA protocols, such as APB and AXI, for greater flexibility.

The AMBA AHB protocol offers an efficient and scalable solution for high-performance SoC designs, enabling efficient communication and data transfer between different system parts. Its flexibility, scalability, and ease of integration make it a popular choice for complex SoC designs.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

V. APPLICATIONS OF AMBA AHB

Some of the common applications of AMBA AHB:

1) *Multimedia Applications:* The AMBA AHB protocol is often used in multimedia applications such as video processing and image recognition, where high-speed data transfer is required for real-time processing.
2) *Communication Systems:* The AMBA AHB protocol is used in communication systems such as routers and switches, where high-speed interconnects are needed to enable efficient communication between multiple devices.
3) *Automotive Applications:* The AMBA AHB protocol is used in automotive applications such as advanced driver assistance systems (ADAS) and infotainment systems, where high-speed data transfer is required for real-time processing and communication.
4) *Aerospace Applications:* The AMBA AHB protocol is used in aerospace applications such as satellites and aircraft. High-performance interconnects are required to enable efficient communication and data transfer between different system parts.
5) *Industrial Automation:* The AMBA AHB protocol is used in industrial automation applications such as programmable logic controllers (PLCs) and robotics, where high-speed interconnects are required to enable real-time communication and control.

Overall, the AMBA AHB protocol is widely used in applications with high-performance interconnects to enable efficient communication and data transfer between multiple devices in a complex system. Its flexibility, scalability, and ease of integration make it a popular choice for various applications in various industries.

VI. RTL IMPLEMENTATION

RTL (Register Transfer Level) implementation of AMBA AHB (Advanced High-Performance Bus) involves designing the digital circuits at the register transfer level using hardware description languages such as Verilog or VHDL. Here are some of the key considerations for RTL implementation of AMBA AHB:

1) *Protocol Compliance:* The RTL implementation must comply with the specifications of the AMBA AHB protocol, including the timing diagrams, signal timings, and state machine requirements.
2) *Bus Arbitration:* The RTL design must include bus arbitration logic to manage access to the shared bus by multiple masters. This involves implementing priority-based or round-robin arbitration schemes to ensure fair access to the bus.
3) *Bus Decoding:* The RTL design must include bus decoding logic to interpret the bus transactions and route them to the appropriate slave devices.
4) *Data Transfer:* The RTL design must include logic to handle data transfers between the masters and slaves. This involves implementing data transfer protocols based on the system's requirements, such as burst or non-sequential transfers.
5) *Clock and Reset:* The RTL design must include clock and reset signals to synchronise the operations of the different components of the system.
6) *Verification:* The RTL design must be thoroughly verified to ensure that it meets the functional and timing requirements of the system. This involves using simulation and verification tools to validate the design against the specifications of the AMBA AHB protocol.

VII. MASTER AND SLAVE TECHNOLOGY

The AMBA AHB (Advanced High-Performance Bus) protocol is designed to provide a high-performance and flexible interconnect fabric for system-on-chip (SoC) designs. The bus architecture is based on a multi-master and multi-slave model, where multiple masters and slaves can share the same bus to transfer data. The protocol defines a standard interface for the masters and slaves, ensuring interoperability between different vendors' components. Here's a detailed explanation of the master and slave technology of AMBA AHB:

1) *Master Technology:* A master in the AMBA AHB protocol is a component that initiates a bus transaction by driving the address and control signals on the bus. The master can be a high-performance processor, a DMA (Direct Memory Access) controller, or a custom logic block. The master technology enables multiple masters to share the same bus and access the slave devices in a controlled manner. The bus arbitration logic ensures that only one master can access the bus simultaneously to prevent conflicts. The AMBA AHB protocol supports multiple bus transfer types, including single transfers, burst transfers, and block transfers, depending on the master's requirement.

2) *Slave Technology:* A slave in the AMBA AHB protocol is a component that responds to the bus transactions initiated by the masters. The slave devices can be memory, peripheral, or custom logic blocks. The slave technology provides the logic to decode the address, control signals, and respond to the master's requests. The slaves can be single or multiple, and each slave can have a unique address range on the bus. The slaves can be accessed sequentially or randomly based on the type of bus transaction. The AMBA AHB protocol supports different response types for the slave devices, including OKAY, ERROR, RETRY, SPLIT, and WAIT states.

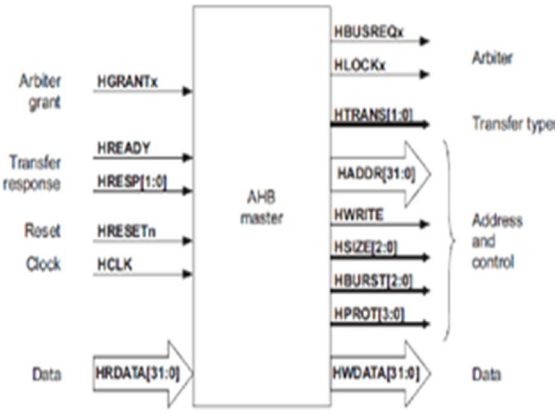


Fig. 2: AHB master

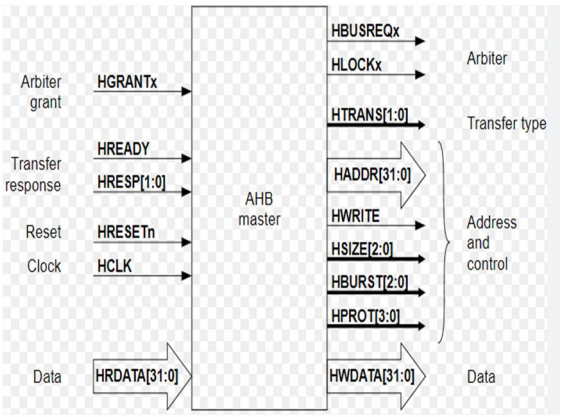


Fig. 3: AHB slave

The AMBA AHB protocol defines several other features to support the multi-master and multi-slave bus architecture, including:

1) *Bus Arbitration:* The AMBA AHB protocol provides a mechanism for bus arbitration to ensure that only one master has access to the bus at a time. The bus arbitration logic supports priority-based and round-robin schemes to provide fair access to the bus.
2) *Pipelining:* The AMBA AHB protocol supports pipelining of the bus transactions to improve the bus's efficiency and reduce the latency of the data transfers. The pipelining allows the master to initiate multiple bus transactions in advance, which can be overlapped to reduce the idle time of the bus.
3) *Burst Transfers:* The AMBA AHB protocol supports burst transfers, which allow the master to transfer a block of data without releasing the bus after each transfer. The burst transfers can be either incremented or wrapped, depending on the type of data transfer.
4) *Split Transactions:* The AMBA AHB protocol supports split transactions, which enable the masters to release the bus temporarily during a long data transfer to allow other masters to access the bus.

The master and slave technology of AMBA AHB enables efficient communication and data transfer between multiple components in a complex SoC design. The protocol provides a standard interface for the masters and slaves, ensuring interoperability between different vendors' components. This makes designing and integrating complex SoC designs easier and reduces the time to market for the end product.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

VIII. RESULTS AND ANALYSIS

This section presents the results obtained during the course of the work. The RTL schematic of the AMBA controller shown in Fig. 4.

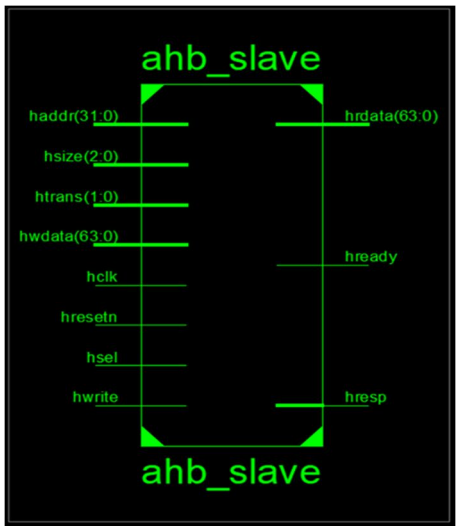


Fig. 4: RTL Schematic of AMBA

The RTL schematic is abbreviated as the register transfer level; it denotes the blueprint of the architecture and is used to verify the designed architecture to the ideal architecture that we are in need of development. The HDL language is used to convert the description or summary of the architecture to the working summary by use of the coding language i.e. Verilog, VHDL. The RTL schematic even specifies the internal connection blocks for better analysing.

The figure represented below shows the RTL schematic diagram of the designed architecture. Here HCLK is used to get the resultant outcome allocated to the controller, or else zero otherwise. HRESTn used to reset both the bus and the system. TRANS is a signal to select different types of operations, data required to write/read will be stored in the register WDATA, HRDATA respectively.

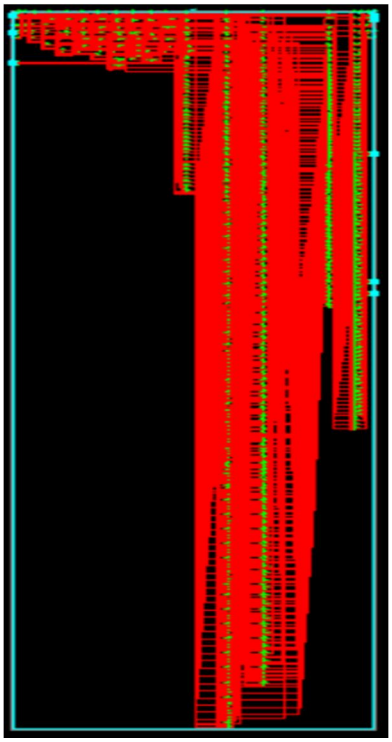


Fig. 5: Technology schematic view

The technology schematic makes the representation of the architecture in the LUT format, where the LUT is considered as the parameter of the area used in VLSI to estimate the architecture design. The LUT is regarded as a square unit. The memory allocation of the code is represented in their LUTs in FPGA.

SIMULATION: The simulation is the final verification process concerning its working, whereas the schematic is the verification of the connections and blocks. The simulation window is launched as shifting from implantation to the simulation on the tool's home screen, and the simulation window confines the output as waveforms. Here it has the flexibility of providing different radix number systems.

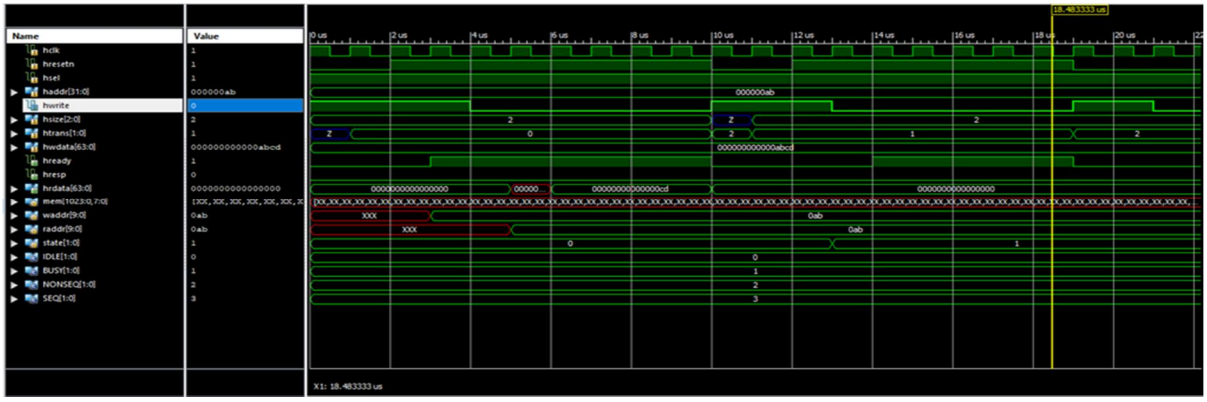


Fig. 6: Simulated Waveform for Idle and Busy Modes

Fig. 7: Simulated Waveform in Non- Sequential mode

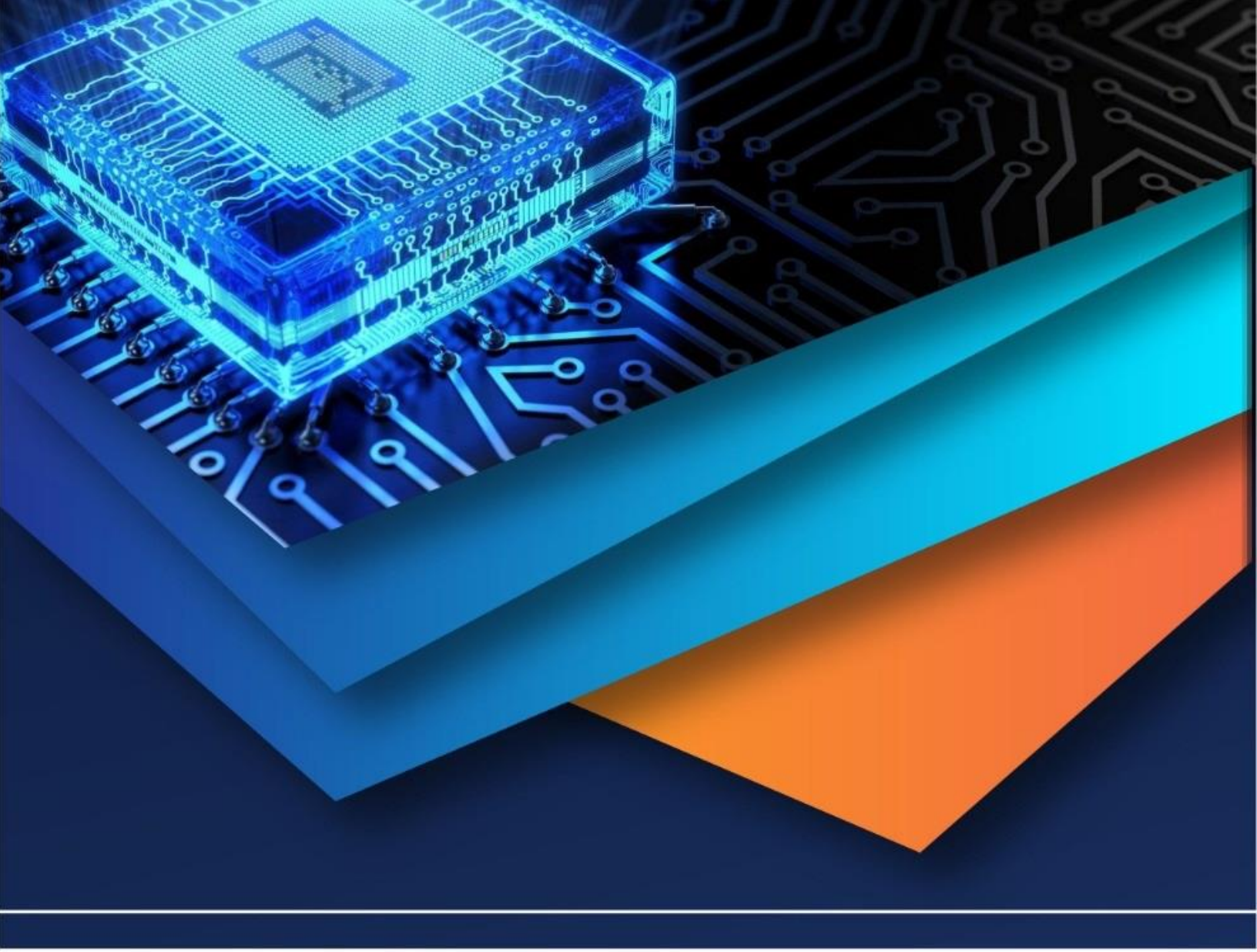
Fig. 8: Simulated waveform for Sequential mode

IX. CONCLUSION

The project involves creating a controller between memory and other components in an electronic system. These memory controllers will be specifically designed for transferring images, which typically require high throughput and low latency. This may involve optimising the controller's performance for image transfer applications by, for example, increasing its bandwidth or reducing its response time. The project involves designing and implementing a controller for the widely-used AMBA protocol and designing memory controllers optimised for image transfer applications. By utilising Xilinx tools and simulators, the project aims to develop efficient and reliable controllers that can effectively manage data transfer within electronic systems. Successful completion of this project involves developing and testing software control and manage data transfer within an electronic system effectively.

REFERENCES

[1] S. Ramagundam et al., "Design and Implementation of a High-Performance Master/Slave Memory Controller with Microcontroller Bus Architecture", Conf. Rec. - IEEE instrument. My. Technology. Conf., no. May, p. 10–15, 2014.
[2] R. S. Kurmi and A. Somkuwar, "AHB Protocol Block Design for Advanced Microcontrollers", Int. J. Compute. Appl. (0975, vol. 32, no. 8, pp. 23–29, 2011.
[3] Kiran Rawat, Kanika Sahni and Sujata Pandey,"RTL Implementation for AMBA ASB APB Protocol at System on Chip Level".
[4] Deeksha L and Shivakumar B.R," Effective Design and Implementation of AMBA AHB Bus Protocol using Verilog".
[5] D. Jayapraveen and T. G. Priya, "Design of a memory controller based on the AMBA AHB protocol", Elixir Comp. Science. Engg. 51A, vol. 2, pages 11115–11119, 2012.
[6] PS Shete and S. Oza, "Designing an Efficient FSM for an AMBA AHB Master Implementation", Int. J. Adv. Res. Calculation. Science. Software Eng., vol. 4, no. 3, p. 267–271, 2014.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY