IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Design And Implementation Of Ahb To Apb Bridge Using Verilog

[1]Sajidha Thabassum B, [2]Kishan S P, [3]Nagesh D

[1]Assistant Professor, [2]Student, [3]Student

[1]Electronics and communication,

[1]Dr Ambedkar Institute of Technology, Bengaluru, India

Abstract: This project aims to design and implement a bridge between AHB and APB protocols using Verilog. It delves into the AMBA bus architecture, emphasizing the high-performance capabilities of AHB and the energy-efficient design of APB. The project addresses the need for seamless communication between these buses in embedded systems. Key objectives include ensuring efficient data transfer, optimizing resource usage, and complying with AMBA standards

Index Terms **– AHB-APB Bridge, Design using verilog, AMBA Protocol.**

I. INTRODUCTION

As embedded systems grow more complex, the need for effective communication between high-speed and low-power components has become essential. ARM's AMBA (Advanced Microcontroller Bus Architecture) protocol provides a standardized framework to address this requirement. Within AMBA, the Advanced High-performance Bus (AHB) supports fast data transfers, while the Advanced Peripheral Bus (APB) focuses on low-power operations for peripheral devices. To ensure seamless interaction between these two buses, a reliable bridge is necessary for efficient data transfer and system integration. This research centers on the design and implementation of an AHB to APB bridge using Verilog. The bridge facilitates interoperability between high-performance processors and low-power peripherals, aiming to optimize performance and minimize latency while adhering to AMBA standards. Rigorous verification methods ensure its reliability across diverse use cases, addressing key challenges in bus communication and contributing to the advancement of embedded system design.

II. AMBA PROTOCOL STACK

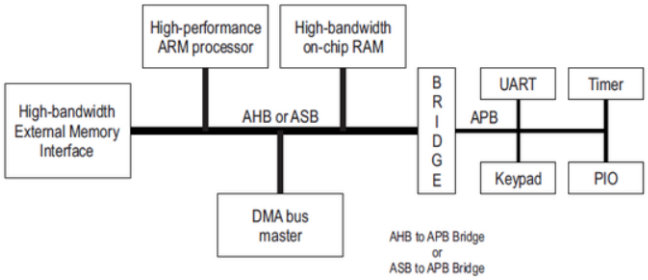

Fig 1: AMBA based microcontroller

1)Advanced High-Performance Bus (AHB)
2)Advanced Peripheral Bus (APB)
3)Advanced System Bus (ASB)

2.1 AHB (ADVANCED HIGH-PERFORMANCE BUS)

The AHB is a high-performance bus used in ARM architectures, designed for fast data transfer and low latency. It connects components like CPUs, memory controllers, and peripherals, supporting burst transfers and efficient pipelining. AHB helps ensure smooth and high-speed communication in complex systems.

2.2 APB (ADVANCED PERIPHERAL BUS)

The APB is a simpler bus used for low-speed peripherals in ARM-based systems. Unlike AHB, it's designed for tasks that don't require high throughput, offering a more efficient interface for components like timers, UARTs, and GPIOs. APB's simplicity reduces power consumption and improves overall system efficiency.

2.3 ASB (ADVANCED SYSTEM BUS)

ASB was an earlier bus used in ARM-based systems, providing a high-speed interface between the processor and peripherals. It is less complex than AHB but offers more features than the APB. ASB supports pipelined transfers and can be used for a range of system components requiring moderate bandwidth.

III. LITERATURE SURVEY

The AHB2APB Bridge facilitates efficient data transfer between high-performance AHB and low-power APB domains. This paper by A Kharade and Jayashree V introduces a synthesizable RTL code that ensures seamless communication between these buses while maintaining design simplicity and practicality. The results demonstrate significant improvements in design efficiency, highlighting the feasibility of the proposed implementation for real-world applications [1]. Similarly, M K Kumar and Amritha Sajja focus on the design and FPGA implementation of an AMBA APB Bridge, employing a clock skew minimization technique to optimize system resource utilization and enhance data transfer reliability. Their methodology addresses clock-related challenges, ensuring a stable and efficient interface, with FPGA validation confirming the effectiveness of the design [2]. Lastly, K. Rawat, Kanika Sahni, and S. Pandey investigate the synthesis and simulation of an AMBA ASB and APB interface at the system-on-chip level. Their work provides detailed insights into power management, resource optimization, and the overall design utilization summary, offering valuable guidance for power-efficient and resource-effective implementations in complex SoC environments [3].

IV. DESIGN OF AHB TO APB BRIDGE

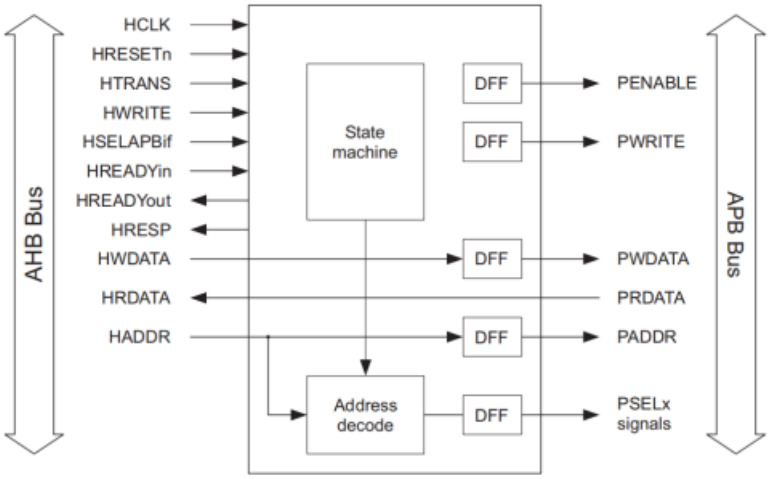

Fig 2: Block diagram of AHB to APB Bridge

The AHB to APB Bridge design consists of key architectural components, including an AHB slave bus interface, an APB transfer state machine, and an APB output signal generator. The AHB slave interface receives transactions from the AHB bus, latches addresses, and controls signals to drive APB peripherals. The transfer state machine governs the sequence of APB operations, adding necessary wait states to accommodate the non-pipelined nature of the APB. Separate PRDATA and PWDATA buses handle data flow in read and write operations, avoiding turnaround delays and simplifying bus management.

The design incorporates address decoding logic to map APB peripherals based on base addresses, allowing seamless integration or modification of peripherals within the system. Control signals like PENABLE and PSELx are generated to synchronize APB access timing and select the appropriate peripheral. PADDR provides

the required address signals for decoding within peripherals, while PWRITE determines the transfer direction. These signals ensure accurate and reliable operation for both read and write cycles, minimizing conflicts between the bridge and connected peripherals.

The system operates efficiently across a wide range of clock frequencies and phases for AHB and APB. HREADYin and HREADYout signals manage transfer readiness and completion, ensuring smooth transaction flow. The bridge's design also enables continuous driving of PWDATA during write operations, while PRDATA is multiplexed for read transactions. The integration of these mechanisms ensures robust data handling and compatibility between high-speed AHB masters and low-power APB peripherals.

V. APB TRANSFER STATE MACHINE

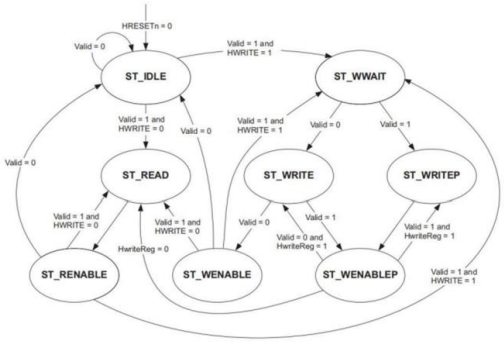

Fig 3: FSM of APB controller

5.1 ST_IDLE:

In this state, the APB buses retain their last values, and PSEL and PENABLE are driven LOW. The FSM enters ST_IDLE upon reset or when no transfers are pending. It transitions to ST_READ for a valid read transfer or ST_WWAIT for a valid write transfer.

5.2 ST_READ:

During this state, the address is decoded and driven onto PADDR, PSEL is asserted HIGH, and PWRITE is driven LOW to indicate a read operation. A wait state is inserted to ensure the APB read data is available on HRDATA. The next state is always ST_RENABLE.

5.3 ST_RENABLE:

The PENABLE signal is driven HIGH to enable the current APB transfer. Other APB outputs remain unchanged from the previous cycle. The FSM transitions to ST_READ for subsequent reads, ST_WWAIT for a write transfer, or ST_IDLE if no further transfers are pending.

5.4 ST_WWAIT:

This state allows the AHB side of a write transfer to complete and ensures the write data is available on HWDATA. The FSM transitions from ST_WWAIT to ST_WRITE to initiate the APB write transfer.

5.5 ST_WRITE:

In this state, the address is decoded, PSEL is asserted HIGH, and PWRITE is driven HIGH to indicate a write operation. No wait state is inserted. The FSM transitions to ST_WENABLE for single writes or ST_WENABLEP for pending transfers.

5.6 ST_WENABLE:

The PENABLE signal is driven HIGH to enable the current APB transfer. Outputs remain unchanged from the previous cycle. The FSM moves to ST_READ or ST_WWAIT for additional transfers, or ST_IDLE if no further transfers are pending.

5.7 ST_WRITEP:

This state supports pipelined write transfers. The address is decoded, PSEL is asserted HIGH, and PWRITE is driven HIGH. A wait state is always inserted to ensure proper sequencing between pending and current transfers. The FSM transitions to ST_WENABLEP for additional writes.

5.8 ST_WENABLEP:

A wait state is inserted if a read follows a write to ensure the write completes before the read begins. The FSM transitions to ST_READ for a read transfer, ST_WRITE for a single write, or remains in ST_WRITEP for additional pending writes.

VI. RESULTS

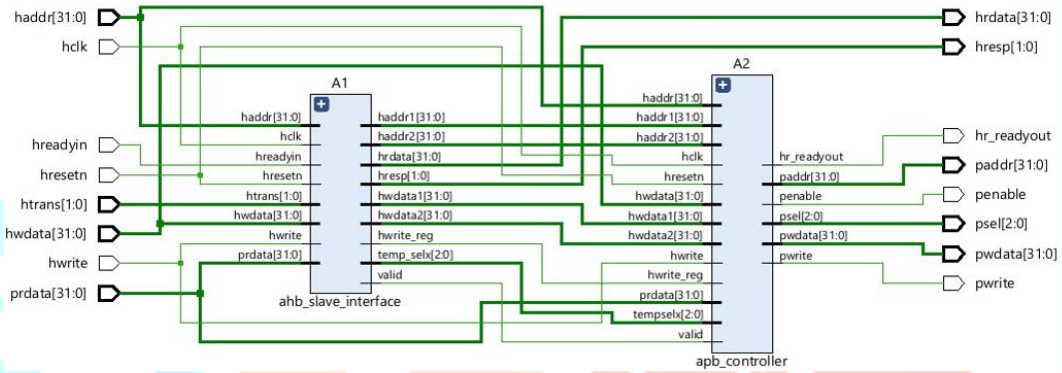

Fig 4: Synthesized netlist of bridge

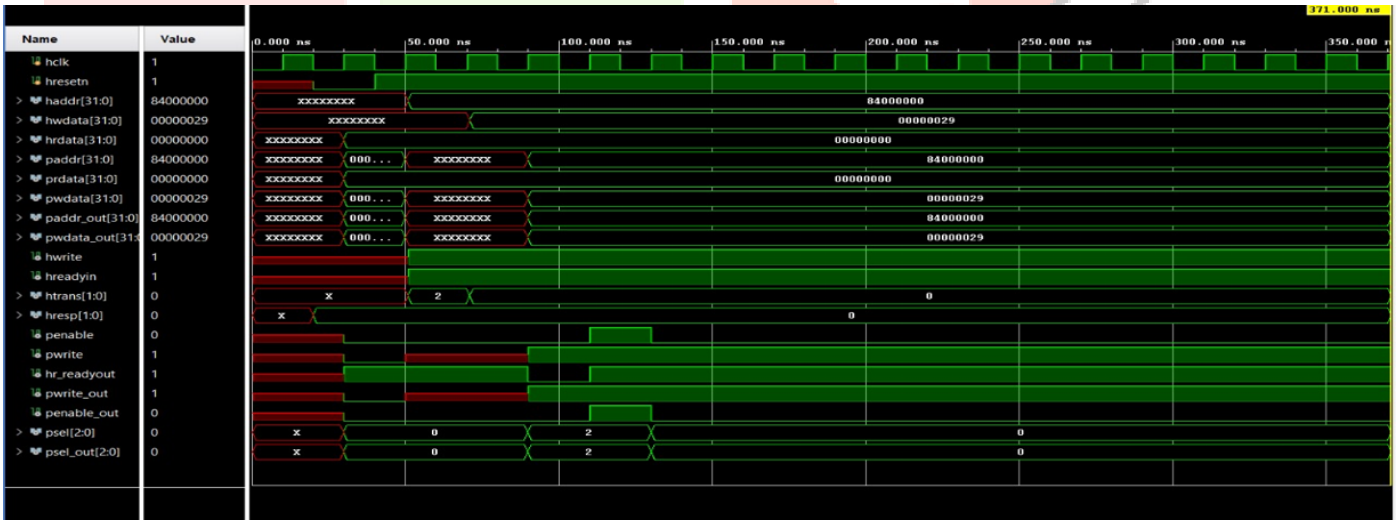

Fig 5: Single Write Transfer

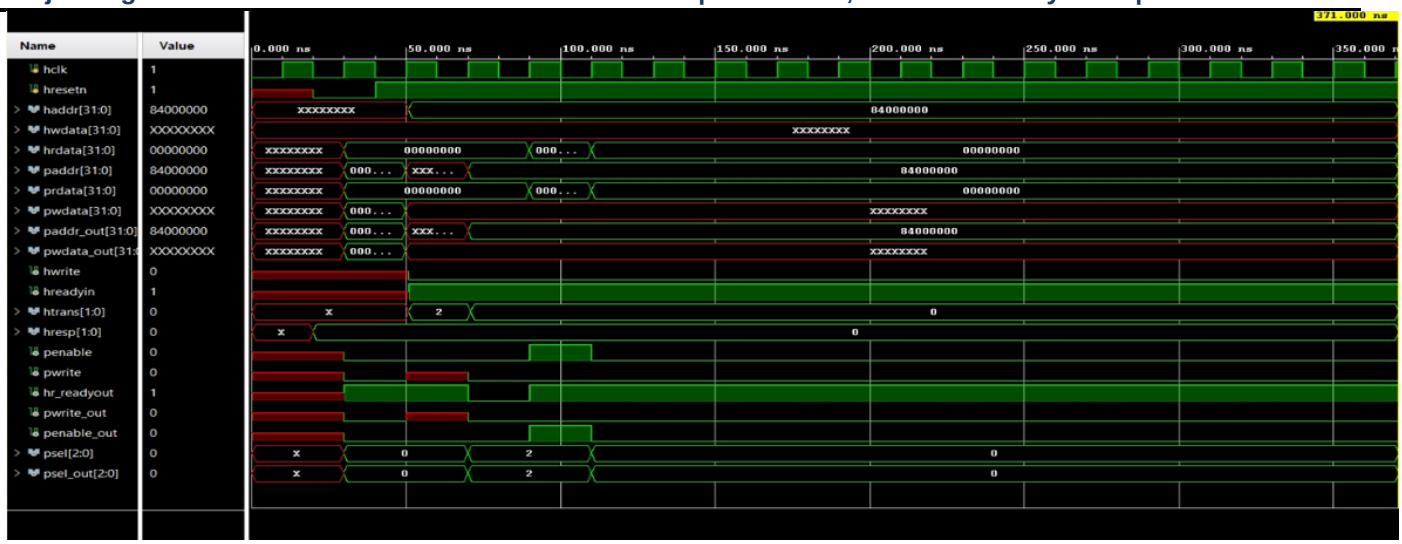

Fig 6: Single Read Transfer

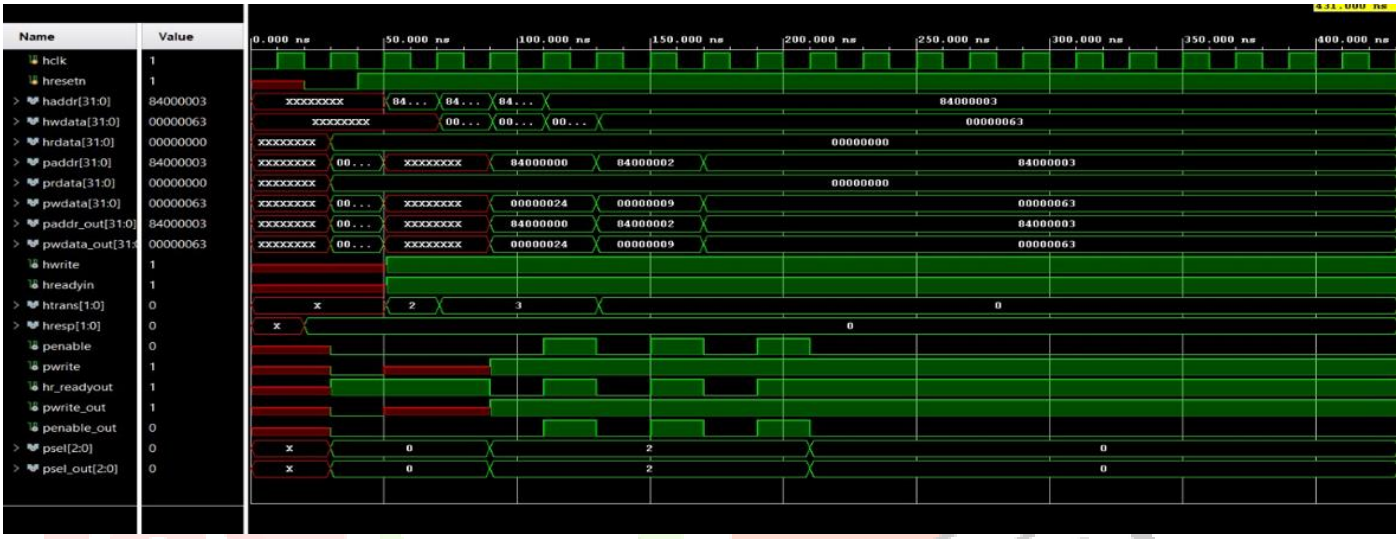

Fig 7: Burst write Transfer

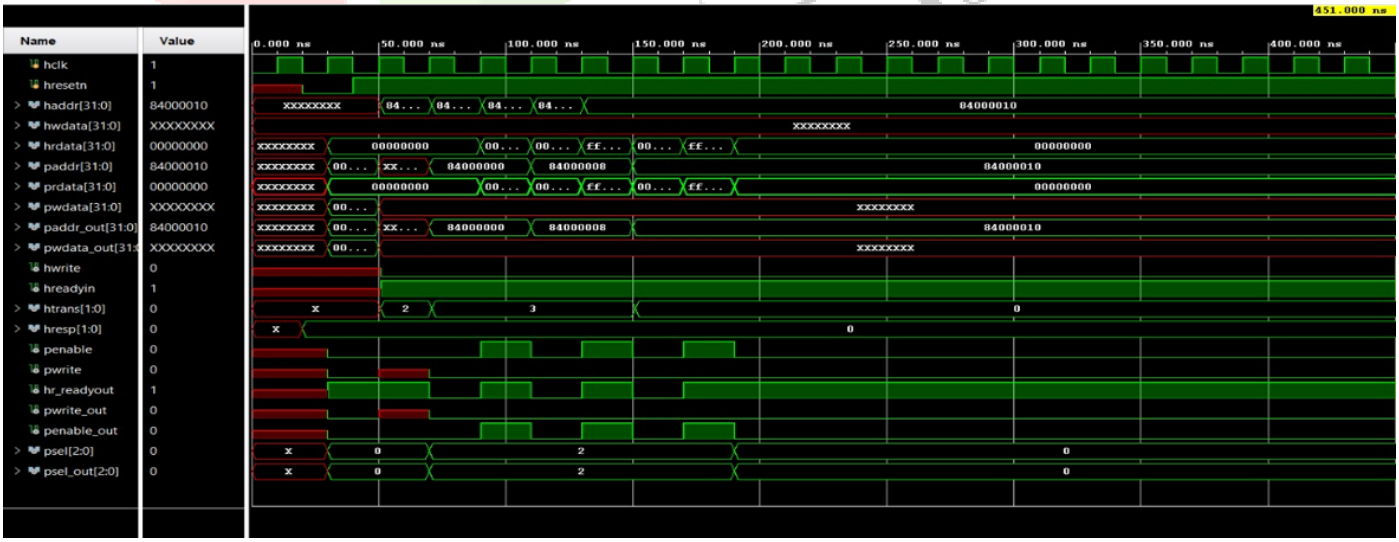

Fig 8: Burst Read Transfer

VII. CONCLUSION

The AHB to APB Bridge design is a fundamental component that enables efficient communication between high-speed AHB masters and low-power APB peripherals. Its well-structured architecture, incorporating an AHB slave interface, a state machine for APB transfers, and precise control signal generation, ensures seamless data transfers across different clock domains and protocols. By utilizing states like ST_IDLE, ST_READ, ST_WRITE, and ST_WENABLE, the bridge effectively manages transfer sequencing while maintaining synchronization between the two buses. The use of separate PRDATA and PWDATA buses minimizes contention, while wait states ensure data integrity and proper operation during pipelined AHB transactions. Additionally, the modular design simplifies system integration, allowing for easy expansion or modification of APB peripherals. This bridge not only optimizes system performance but also supports low-power consumption, making it an indispensable element in modern embedded systems. Through its robust design and adaptability, the AHB to APB Bridge plays a crucial role in bridging the gap between high-speed processing units and energy-efficient peripherals.

VIII. REFERENCES

[1] A. Kharade; Jayashree V. “VLSI Design of AMBA Based AHB2APB Bridge” 2021 IEEE International Conference on Electronics and Communication Systems (ICECS).

[2] M. K. Kumar; Amritha Sajja. “Design and FPGA Implementation of AMBA APB Bridge with Clock Skew Minimization Technique” 2021 IEEE International Symposium on VLSI Design and Test (VDAT).

[3] K. Rawat; Kanika Sahni; S. Pandey. “RTL Implementation for AMBA ASB APB Protocol at System on Chip Level” 2020 IEEE International Conference on System-on-Chip Design and Implementation (SOC-DI).

[4] K. Rawat; Kanika Sahni; S. Pandey. “Design of AMBA APB Bridge with Reset Controller for Efficient Power Consumption” 2020 IEEE International Conference on Sustainable Computing in Electrical and Electronics Engineering (ICSCEE).

[5] B. N. Manu P.; Prabhavathi. “Design and Implementation of AMBA ASB APB Bridge” 2019 IEEE International Conference on Embedded Systems (ICES).

[6] Chenghai Ma; Zhijun Liu; Xiaoyue Ma. “Design and Implementation of APB Bridge Based on AMBA 4.0” 2018 IEEE International Conference on Microelectronics (ICM).

[7] L. Deeksha; B. Shivakumar. “Effective Design and Implementation of AMBA AHB Bus Protocol using Verilog” 2018 IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI).

[8] Padmaprabha Jain; Satheesh Rao. “Design and Verification of Advanced Microcontroller Bus Architecture-Advanced Peripheral Bus (AMBA-APB) Protocol” 2020 IEEE International Conference on Recent Advances in Communication and Computing (ICRACC).

[9] Shaila S. Math; R. Manjula; S. Manvi; Paul Kaunds. “Data Transactions on System-on-Chip Bus Using AXI4 Protocol” 2020 IEEE International Conference on VLSI Design and Test (VDAT).