Design of Advanced High Performance Bus (AHB) to Advanced Peripheral Bus (APB) Bridge

K. Mounika[1], K. Koteswararao[2], K. Praneeth [3], K. BhanuPrakash[4]

[1,2,3,4] *ECE,RVRJCCE,INDIA*
E-mail:mounica.kancheti@gmail.com[1]
E-mail:kavurikoti904@gmail.com[2]
E-mail:praneethkondeti155@gmail.com[3]
E-mail:bhanukommareddy12@gmail.com[4]

Abstract-*The Advanced Microcontroller Bus Architecture (AMBA) has long served as the on-chip bus standard for System-on-Chip (SoC) designs, despite its inherent low performance. Within the AMBA framework, the Advanced High-performance Bus (AHB) is utilized to connect analog and digital signal processors, alongside high-performance memory controllers. Conversely, the AMBA Advanced Peripheral Bus (APB) facilitates connections to peripherals such as UARTs. Both buses are interconnected via a bridge, known as the AHB2APB Bridge, presenting a challenging proposition for the evaluation of AMBA-based systems. AMBA bus protocol, enables communication between different functional blocks or Intellectual Properties (IP), our endeavor focuses on building high-performance SoC designs. By exploiting the reusability of IP across different buses and supporting multiprocessor unit development, we strive to bridge the communication gap between high-performance AHB and low-speed APB. The simulation of the AHB2APB Bridge module, designed in Verilog HDL is conducted on the Xilinx Vivado platform. Our primary objective is to achieve high-speed pipelined data transfers while ensuring no data loss during transmission*

Key Words: System on chip, AHB2APB, Verilog HDL, Xilinx Vivado

1.INTRODUCTION

The Advanced Microcontroller Bus Architecture (AMBA) serves as a crucial System-on-Chip (SoC) communication protocol, facilitating seamless interaction between high-performance buses and low-power devices. Within the AMBA framework, the Advanced High-Performance Bus (AHB) functions as the backbone for connecting processors, digital signal processors, and high-performance memory controllers, while the Advanced Peripheral Bus (APB) is tailored for interfacing with peripherals like UARTs. Additionally, a bridge is incorporated to link the AHB and APB buses, ensuring comprehensive connectivity throughout the system.

AMBA has emerged as a widely adopted interconnection standard in System on Chip (SoC) design due to its versatility and effectiveness in meeting essential requirements. These requirements include streamlining the development process for embedded microcontroller products featuring multiple signal processors, ensuring technology independence for seamless migration across different IC processes, and optimizing on-chip and off-chip interaction for efficient manufacturing and operation.

The AHB bus protocol employs a multiplexer interconnection technique, wherein bus masters drive address and control signals to indicate transfer types (e.g., read or write).[4][5][6] Address routing to specific slaves occurs when the arbiter selects the master with the corresponding control signal. Central to this process is the central decoder, responsible for controlling data read and signal response multiplexers, ensuring the selection of relevant signals from slaves required for transfer. In contrast, the APB is tailored for connecting low-bandwidth peripherals that do not necessitate the high performance of a pipelined bus interface.

AMBA protocol plays a pivotal role in enabling efficient communication within SoC designs, fostering integration, scalability, and interoperability across diverse components and functionalities.

2. ADVANCED MICROCONTROLLER BUS ARCHITECTURE

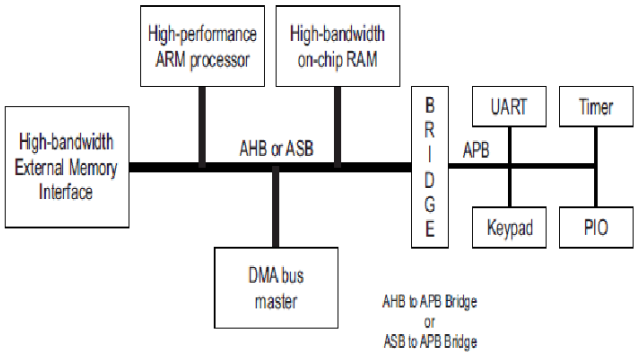

Fig -1: AMBA based System

In AMBA Revision 2.0 three distinct buses are described for facilitating on-chip communications. These are the Advanced High-Performance Bus (AHB), the Advanced System Bus (ASB), and the Advanced Peripheral Bus (APB). The AHB is the backbone of the system and is designed specifically for high performance, high-frequency

components. This includes the connections of processors, on-chip memories, and memory interfaces among others.

The ASB is an alternative to the AHB where some high-performance features are not needed. The APB is a simplified interface designed for low bandwidth peripherals that do not require the high performance of the AHB or the ASB. These include components like a UART, low-frequency GPIO, and timers.

3. AHB & APB

The Advanced High-performance Bus (AHB) represents a significant advancement within the framework of the AMBA (Advanced Microcontroller Bus Architecture) bus system, specifically tailored to meet the demands of high-performance synthesizable designs. AHB emerges as a robust channel of transport designed to cater to the essential requirements of high-frequency systems interfacing. It stands above the Advanced Peripheral Bus (APB) in the hierarchy, offering a set of features crucial for high-performance applications.

Features of AHB includes:

- **Burst Transfers**: allows efficient movement of data blocks between memory and peripherals, which is particularly advantageous in applications requiring rapid data processing.
- **Split Transactions**: AHB facilitates Split Transactions, enabling concurrent transactions to occur, thereby enhancing overall system throughput and efficiency.
- **Single clock edge operation**: AHB operates on a Single clock edge operation basis, streamlining the timing of data transfers and enhancing synchronization within the system. This ensures precise coordination between different components, minimizing latency and improving overall system responsiveness.
- **Single cycle bus master handover**: enables seamless and efficient handover of bus control between different masters within the system. This feature enhances system agility and responsiveness, particularly in multi-master configurations.
- **Non tri-state implementation**: AHB also eschews the traditional tri-state implementation in favor of a Non tri-state approach, which simplifies bus arbitration and reduces potential contention issues. This contributes to smoother data flow and improved system reliability.
- **Wider data bus configurations**: AHB supports Wider data bus configurations, allowing for the transfer of larger data payloads in a single cycle. This capability is instrumental in high-throughput applications where large volumes of data need to be processed rapidly.

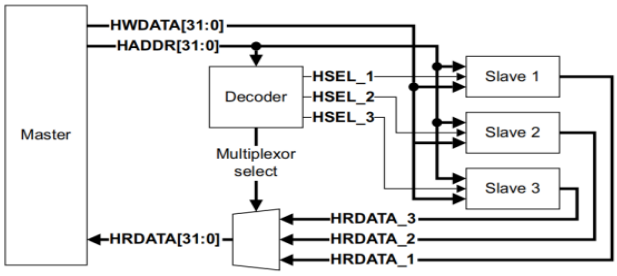

Fig -2: AHB Block Diagram

Fig-2 outlines the necessary components required to implement an AMBA AHB architecture, including the decoder, master, and slave modules.

The Advanced Peripheral Bus (APB) constitutes an integral component within the Advanced Microcontroller Bus Architecture (AMBA) hierarchy of buses, designed to optimize power consumption while minimizing data overhead. Unlike its counterparts, APB emphasizes simplicity and cost-effectiveness, catering to peripherals with lower bandwidth requirements that do not necessitate the sophisticated features of a pipelined transportation interface.

The latest iteration of APB ensures that all transactions are synchronized with the rising edge of the clock, streamlining the data transfer process and enhancing overall system efficiency. The refinement of APB peripheral protocols seamlessly integrates into end-to-end workflows, yielding several advantages.

The APB serves as a versatile interface within the AMBA architecture, capable of interfacing with various bus protocols to accommodate diverse system requirements.[5] Here's how the APB can interface with different AMBA bus protocols:

- AMBA Advanced High-Performance Bus
- AMBA Advanced High-Performance Bus Lite
- AMBA Advanced Extensible Interface (AXI)
- AMBA Advanced Extensible Interface Lite

4. FEATURES OF AHB2APB BRIDGE

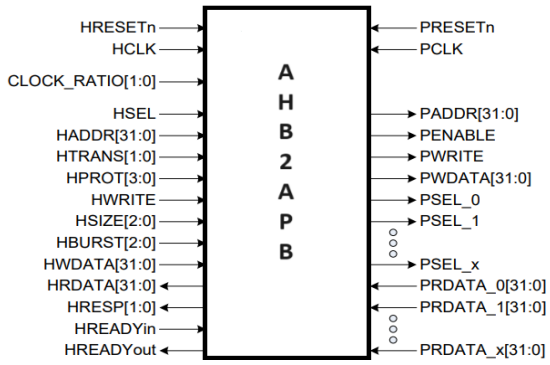

Fig -3: AHB2APB Block Diagram

The AHB to APB bridge is a hardware component used in system-on-chip (SoC) designs to interface between the Advanced High-performance Bus (AHB) and Advanced Peripheral Bus (APB) protocols. It allows for communication between high-speed and low-speed peripherals, managing data transfers and ensuring compatibility between different bus architectures. The bridge typically handles address decoding, data transfer arbitration, and protocol conversion between the two bus systems, enabling efficient communication within the system-on-chip (SoC). Block Diagram of AHB2APB bridge is shown in fig-3.

5. IMPLEMENTATION OF AHB2APB BRIDGE

i. Finite State Machine of AHB2APB:

The transfer state machine orchestrates the execution of APB transfers by responding to inputs from the AHB. The operational flow of this state machine, as depicted in Figure 4, is governed by its present state and interactions with the AHB slave interface.[8]

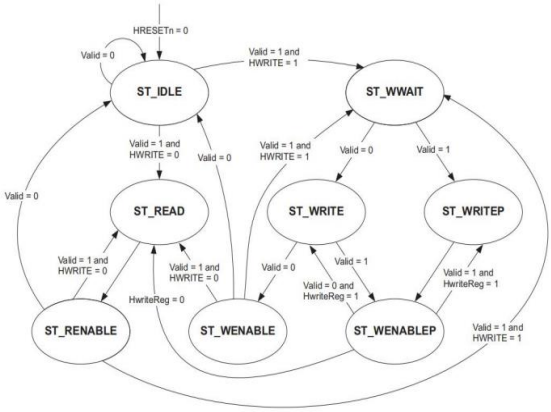

Fig -4: FSM of AHB2APB

In a typical AMBA AHB system configuration, several key components work together to facilitate efficient data communication. These components include the AHB master, AHB slave, AHB arbiter, and AHB decoder. The AHB master functions as the initiator of read and write operations, employing address and control information to trigger data transfers. It's important to note that only a single bus master is granted access to the bus at any given time, ensuring orderly communication.

On the other side, the AHB slave operates as the responder to read or write operations, reacting according to the specific address space range it covers. The role of the AHB arbiter is critical in maintaining bus integrity. It coordinates data transfers by allowing only one bus master to initiate each transaction, thereby preventing conflicts and ensuring data integrity. While the core protocol is standardized, the design of the arbitration algorithm can be customized to suit the specific needs of the application. An essential participant, the AHB decoder, tackles the task of address decoding for each transfer. This component deciphers the address information and generates select signals that guide the appropriate slave to respond. An important note is that a single, centralized decoder is a consistent requirement across all AHB implementations.

ii. Design Flow:

The state machine logic is incorporated in Controller. whereas decoder contains the path of selecting appropriate slave. AHB master uses Mux logic to retrieve data from the slave. The complete design Flow is shown in fig-5.

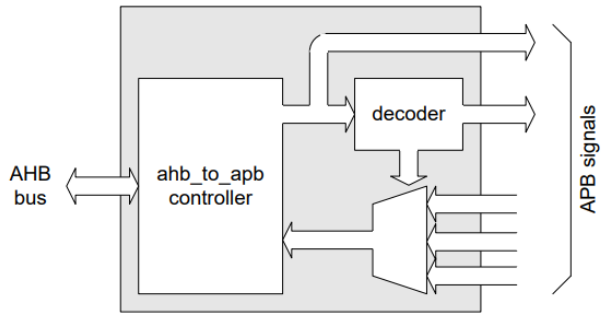

Fig -5: Design Flow

iii. Schematic:

The schematic of AHB2APB bridge module is represented in fig-6

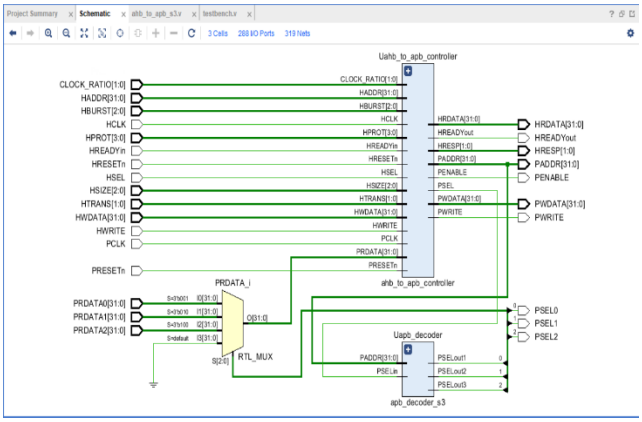

Fig -6: Schematic of AHB2APB

6. SIMULATION RESULTS

The simulation results of one master and three slave system for different transfers of single read, single write, burst read and burst write are shown. Appropriate Slave is selected based on the input address given.

1.Single Read:

The Behavioral simulation result of single read is represented in fig-7. In a single read operation, the AHB master sends address and control signals to the AHB2APB bridge, which translates and forwards the request to the APB device. After retrieving the data, the bridge sends it back to the AHB master, completing the operation.

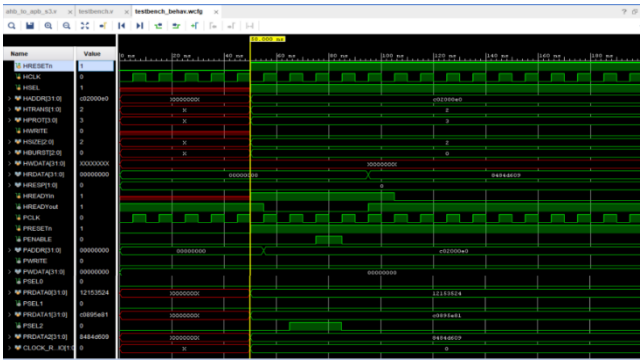

Fig -7: Single read waveform

2.Single Write:

The Behavioral simulation result of single write is represented in fig-8. In a single write operation, the AHB master sends data to the AHB2APB bridge, which translates and forwards the request to the APB device for writing, completing the operation

Fig -8: Single write waveform

3.Burst Read:

The Behavioral simulation result of burst read is represented in fig-9.In a burst read operation, the AHB master sends a series of consecutive read requests to the AHB2APB bridge, which translates and forwards them to the APB device, fetching multiple data values in a continuous sequence.

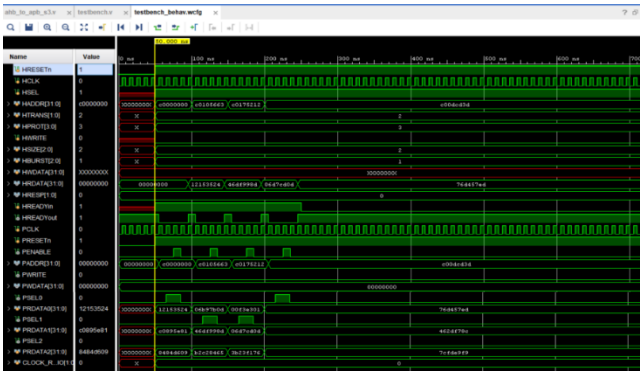

Fig -9: Burst Read waveform

4.Burst Write:

The Behavioral simulation result of burst write is represented in fig-10. In a burst write operation, the AHB master sends a sequence of consecutive write requests to the AHB2APB bridge, which translates and forwards them to the APB device, storing multiple data values in a continuous sequence.

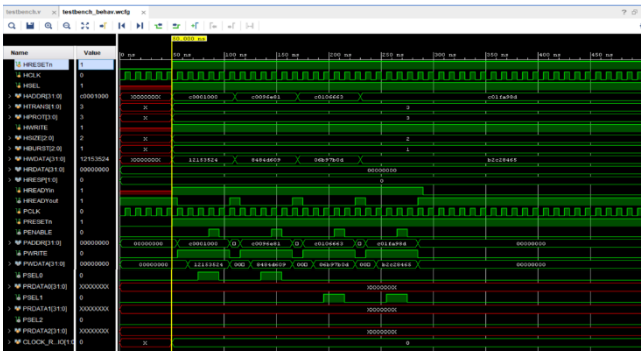

Fig -10: Burst Write waveform

7. CONCLUSIONS

In conclusion, the implementation of the AMBA (Advanced Microcontroller Bus Architecture) AHB (Advanced High-performance Bus) to APB (Advanced Peripheral Bus) bridge using Verilog showcases efficient communication between different bus architectures. Through operations like single read, single write, burst read, and burst write, the bridge demonstrates seamless data transfer and protocol conversion. This project not only highlights the versatility of Verilog in hardware description and simulation but also underscores the importance of interoperability in modern system-on-chip designs. Its potential impact extends to various applications, including embedded systems, IoT devices, and high-performance computing platforms.

REFERENCES

[1]. Deeksha L, Shiva Kumar B.R, “Effective Design and Implementation of AMBA AHB Bus Protocol using Verilog” International Conference on Intelligent Sustainable Systems (ICISS 2019).

[2]. Padmaprabha Jain, Satheesh Rao, "Design and Verification of Advanced Microcontroller Bus Architecture-Advanced Peripheral Bus (AMBA-APB) Protocol” ICICV 2021.

[3]. Aparna Kharade and V. Jayashree, “VLSI DESIGN OF AMBA BASED AHB2APB BRIDGE” International Journal of VLSI design & Communication Systems (VLSICS) Vol.9, No.3, June 2018.

[4]. AHB Example AMBA system; Technical Manual ARM1999.

[5]. AMBA Specification, Rev 2.0, ARM Limited.

[6]. A. B. Nithin Joe, "Design and analysis of a N Ovel Digital," Int. J. Adv. Eng. Technol., vol. 4, no. 2, pp. 2053–2063, 2013.

[7]. R. S. Kurmi and A. Somkuwar, "Design of AHB Protocol Block for Advanced Microcontrollers," Int. J. Comput. Appl. (0975, vol.32, no. 8, pp. 23–29, 2011.

[8]. D. Jayapraveen and T. G. Priya, "Design of memory controller based on AMBA AHB protocol," Elixir Comp. Sci. Engg. 51A, vol. 2, pp. 11115–11119, 2012.

[9]. P. S. Shete and S. Oza, "Design of an Efficient FSM for an Implementation of AMBA AHB Master," Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 4, no. 3, pp. 267–271, 2014.

[10]. Z. Khan, T. Arslan, and A. T. Erdogan, "A novel bus encoding scheme from energy and crosstalk efficiency perspective for AMBA based generic SoC systems," Proc. IEEE Int. Conf. VLSI Des., no. January, pp. 751–756, 2005.