Design and Implementation of APB-Based Memory Communication

Ch.Chaitanya Laxmi Prasad,
Department of Electrical and Communication Engineering, Kalasalingam Academy of Research and Education,
Krishnankoil,Tamilnadu
chaitanyacheerala@gmail.com

Dr.J.Charles Pravin,
Department of Electrical and Communication Engineering, Kalsalingam Academy of Research and Education,
Krishnankoil,Tamilnadu
charles@klu.ac.in

Ramesh G,
Manager- DFTTI,
rameshg@dfttraining.in

Abstract

The Advanced Peripheral Bus (APB) is a widely adopted protocol in applications prioritizing low power consumption and moderate bandwidth, typically for connecting peripheral devices. This paper presents the design and implementation of a memory communication system based on the APB protocol, aimed at enabling efficient data exchange among memory units within embedded systems. The proposed design leverages the simplicity of APB to achieve reliable, low-energy communication and reduced system complexity. A well-structured APB bridge facilitates data transfer between memory modules, ensuring synchronized and accurate communication. The system is implemented and verified through simulation, with a focus on key performance parameters such as latency, power consumption, and throughput. While APB offers significant advantages in terms of simplicity and power efficiency, it does come with limitations—most notably, restricted bandwidth and the absence of pipelining, which can impact performance in high-throughput scenarios. However, for low-power embedded and IoT systems where high-speed data transfer is not a critical requirement, APB remains a practical and cost-effective solution. This study highlights the utility of APB in memory interfacing and provides an economical option for resource-constrained environments.

Keywords: Memory Interaction, Integrated Systems, Communication with Low Power , Information Transfer ,APB Connector,Delay ,Energy Usage ,IoT Frameworks, Complexity of the System, Modeling and Validation.

I. Introduction:

The The Advanced Peripheral Bus (APB) is a commonly utilized bus protocol in low-power and low-bandwidth scenarios, primarily intended for connecting peripheral devices such as UART, GPIO, and timers. Unlike other AMBA (Advanced Microcontroller Bus Architecture)[1] protocols like AXI (Advanced eXtensible Interface) or AHB (Advanced High-Performance Bus)[2], APB employs a straightforward, non-pipelined design, making it a favorable choice where power efficiency and implementation simplicity are prioritized over high-speed data throughput.

In many embedded and System-on-Chip (SoC) [4] designs, memory communication is typically handled by high-speed interconnects [3]. However, for targeted low-power applications, utilizing APB for memory-to-memory data transfers can deliver notable advantages in terms of power efficiency [5], cost savings, and reduced design complexity. That said, APB is not without limitations. Its non-pipelined nature and restricted bandwidth can lead to increased latency, especially in larger or more complex memory systems. Furthermore, scaling APB-based architectures to accommodate growing memory demands or parallel data transfers presents significant challenges, potentially limiting its applicability in more demanding environments.

This document investigates the development and implementation of an APB-based memory communication system aimed at enabling efficient data exchange among memory units in constrained embedded settings [6]. The proposed architecture uses an APB bridge to manage data transfers, ensuring effective synchronization and reliable communication. A key focus of this study is to evaluate latency, throughput, and power consumption [7] in memory-interfacing scenarios utilizing APB. Through simulation and implementation, we demonstrate the viability of APB for memory-to-memory communication in low-power embedded and IoT systems [8], while also acknowledging its design trade-offs and limitations.

The remainder of the paper is organized as follows: Section II provides an overview of APB and its role in embedded systems. Section III details the design methodology and system architecture. Section IV discusses implementation specifics and simulation results. Finally, Section V concludes the paper with key observations and suggestions for future research directions.

II. Methodology ;

The Advanced Peripheral Bus (APB) is extensively employed for communication with low-power and high-efficiency peripherals [9]. The optimization of its interfacing methodology is critical for achieving rapid data transmission, minimal latency, and effective handshaking protocols. The subsequent methodology delineates essential components requisite for the establishment of an optimized APB-based communication system.

Importance Of The APB.

The Advanced Peripheral Bus (APB) constitutes a fundamental element of the Advanced Microcontroller Bus Architecture (AMBA), specifically engineered to facilitate low-power and low-complexity interactions between a microprocessor and peripheral devices [10]. It assumes a pivotal function in establishing connections with register-based peripherals, including General-Purpose Input/Output (GPIO), Universal Asynchronous Receiver-Transmitter (UART), Serial Peripheral Interface (SPI), timers, and watchdog timers. The APB operates utilizing a singular clock cycle and adheres to a two-phase protocol comprising a setup phase and an access phase, thereby ensuring straightforward and efficient data transfers. In contrast to high-performance buses such as the Advanced High-performance Bus (AHB) and the Advanced eXtensible Interface (AXI), the APB is specifically optimized for applications characterized by lower bandwidth requirements, rendering it particularly suitable for peripherals that do not necessitate high-speed communication [11]. Its synchronous architecture diminishes power consumption and simplifies the implementation process, establishing it as a favored option in embedded systems and System-on-Chip (SoC) designs.

However, when scaling APB to multi-memory setups or memory-to-memory communication systems, the simple structure of APB introduces synchronization challenges. These include managing signal arbitration, ensuring accurate sequencing of read/write operations across multiple slaves, and avoiding bus contention or timing violations due to non-pipelined access. The following methodology includes design provisions to mitigate such issues.

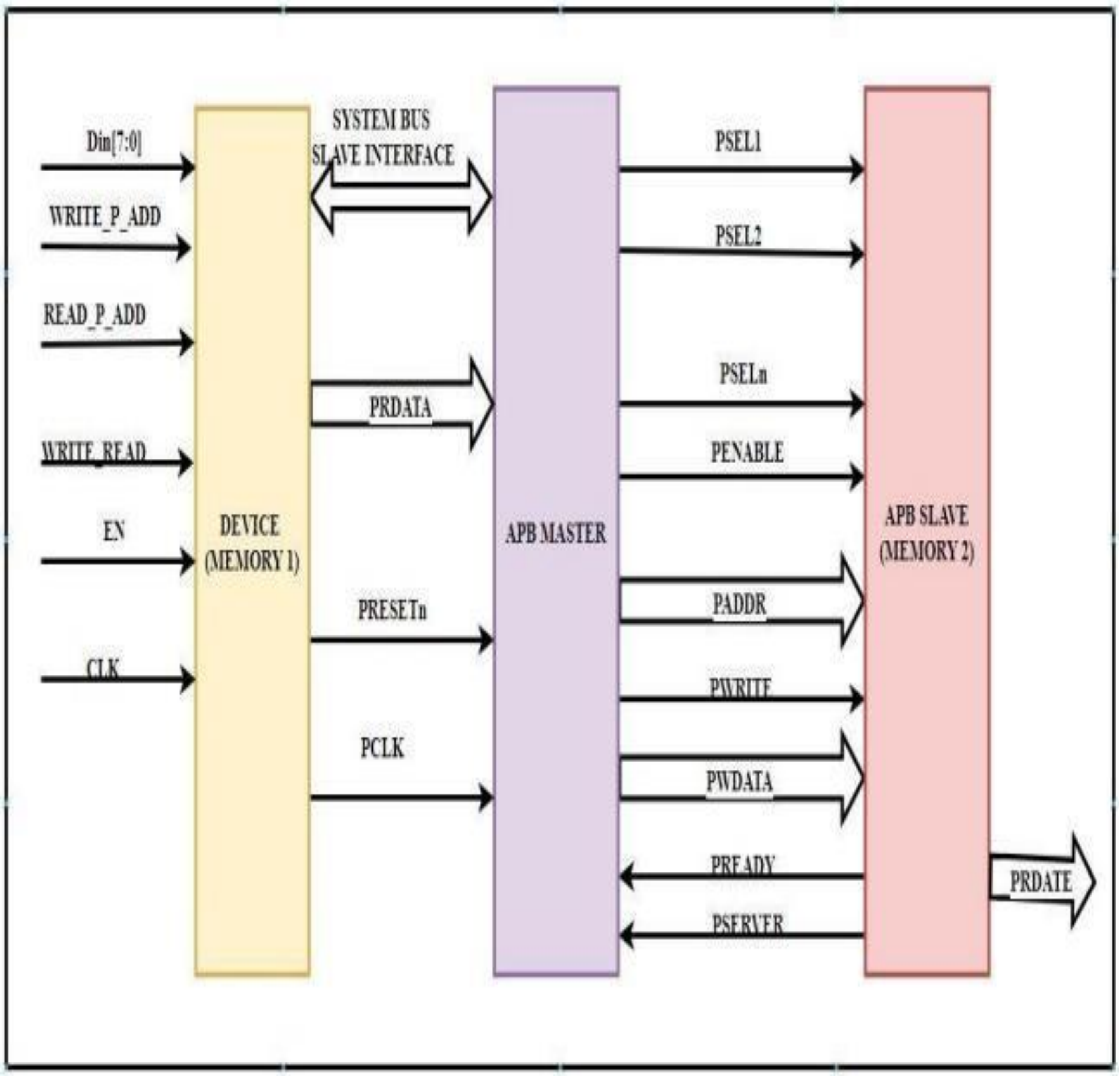


Fig. 1: Block diagram of Proposed Model

This illustration Fig. 1 shows a communication setup between two memory units via an APB (Advanced Peripheral Bus) interface, enabling effective data exchange. Memory 1 (designated as Device) interfaces with the APB Master via a system bus slave connection, taking in signals like Din[7:0], WRITE_P_ADD, READ_P_ADD, WRITE_READ, EN (Enable), and CLK (Clock). The slave interface of the system bus handles data transfer by sending PRDATA to the APB Master. The APB Master manages communication with Memory (APB Slave) through control signals such as PSEL (Peripheral Select), PENABLE (Enable), PADDR (Address), PWRITE (Write Enable), and

PWDATA (Write Data). It additionally tracks the PREADY (Ready) and PSERVER (Slave Response) signals to guarantee correct data transmission. The system depends on synchronizing clocks with PCLK and resets using PRESETn. This design allows for smooth read/write processes, guaranteeing data integrity and effective communication between the two memory units.

System Design & Architectural Considerations:

Establish the hierarchical configuration of APB master-slave relationships, ensuring adequate signal synchronization. Choose a suitable clock frequency (PCLK) to achieve a harmonious balance between power consumption and performance efficiency. Enhance the PADDR (Address Bus) width in accordance with peripheral specifications to prevent superfluous utilization of address space.

A. SIGNAL ANALYSIS

i) Clock (PCLK) and Control Signals Behavior:

The Peripheral Clock (PCLK) is characterized as a periodic waveform that facilitates synchronized operations among components. The PENABLE signal transitions to a high state upon the initiation of an active transfer, thereby indicating the commencement of a valid address and control phase. The PWRITE signal ascertains the nature of the operation, distinguishing between a write (indicated by a high state) and a read (indicated by a low state). The PREADY signal reflects the readiness of the slave device to finalize the current data transfer; when this signal is low, the master device is required to defer further actions until the signal indicates readiness. The PSEL (Peripheral Select) signal is activated when the master device designates a specific slave device for the purpose of communication.

In a setup involving several memory modules interfaced to a single APB master, synchronization is achieved using a deterministic access scheme. The master issues PSEL for one slave at a time, avoiding contention. Moreover, timing skew between PREADY signals of various slaves is managed by inserting wait states where required.

ii) Data Transactions during Read/Write Operations: Write Operations:

The PWDATA signal transmits data from the master device to the slave device, and this data is considered valid when the PENABLE signal is asserted high. The address designated for the write operation is established through the PADDR signal and is maintained in a stable state for the duration of the operation. Read Operations: The PRDATA signal is responsible for conveying the data that is read from the slave device back to the master device. The validity of this read data is contingent upon the PREADY signal being high, thereby ensuring the successful completion of the read cycle.

iii) Optimization in APB-Based Interfacing: Minimized Latency:
The optimization of control signal timing is executed to achieve minimal latency between successive data transactions. Efficient Handshaking: The transitions of the PREADY and PENABLE signals are meticulously streamlined to mitigate the occurrence of wait states. Stable Addressing is the integrity of the address mapping is rigorously verified to ensure consistency, thereby guaranteeing accurate data routing.

B. PERFORMANCE OPTIMIZATION

1. Optimization of the Address Phase

The Peripheral Address (PADDR) is captured at an earlier stage, thereby minimizing the setup duration prior to the commencement of data transmission. The timely assertion of the PSEL signal effectively reduces the occurrence of idle cycles.

2. Optimization of the Data Phase

The PENABLE signal is activated at an optimal moment to facilitate swift data acknowledgment. The PREADY signal is meticulously observed to avert superfluous wait cycles during the transfer process.
C. Enhancement of Write and Read Path In the context of write operations, the data within PWDATA is rendered accessible one cycle in advance, thus mitigating setup delays. For read operations, the PRDATA is captured immediately subsequent to the PREADY signal reaching a high state, thereby avoiding unnecessary wait states.

C.Efficient Handshaking Mechanism (PREADY, PENABLE Timing)

Proper Synchronization of PREADY and PENABLE The assertion of PENABLE occurs solely subsequent to the stabilization of PSEL, thereby guaranteeing the accurate selection of the intended peripheral. The slave dynamically regulates PREADY, signifying its preparedness to accept or transmit data without incurring superfluous delays. Optimized PREADY Response In instances where PREADY = 1 concurrently with PENABLE, the data transfer is executed instantaneously (Single-cycle transaction). Conversely, if PREADY = 0, the master is required to await the slave's processing of the request, thus preventing the occurrence of data corruption. C. Pipeline-based Approach The adoption of a preemptive response mechanism, wherein the slave anticipates PREADY behavior, can substantially diminish the number of wait cycles. The application of clock gating techniques serves to reduce power consumption while simultaneously ensuring prompt signal transitions.

III. RESULTS AND DISCUSSION

1. Memory Interface Waveform

The presented image illustrates a GTKWave simulation pertaining to an APB-oriented memory interface, scrutinizing the intricacies of read and write operations. The principal signals under examination include:

i)Clock (clk):

Facilitates the synchronization of memory transactions. Data Signals (data_in[7:0], data_out[8:0]): Represent the input and output data buses. Control Signals (en, we): Constitute the Enable and Write Enable signals that govern memory access protocols. Address Signals (read_paddr[12:0], write_paddr[12:0]): Designate specific memory locations for the execution of read and write operations.
The waveform delineates systematically organized memory transactions, thereby ensuring the optimization of data flow between the processor and the memory subsystem. The signal transitions signify effective data management, characterized by appropriate address decoding and the execution of write operations at various timestamps.The illustration in Fig. 2 presents a waveform simulation conducted in GTKWave, pertaining to an APB (Advanced Peripheral Bus) oriented memory interface. The signals exhibited encompass clk (clock), data_in, data_out, en (enable), read_addr, write_addr, and we (write enable), all of which are essential for facilitating high-velocity memory read/write transactions.

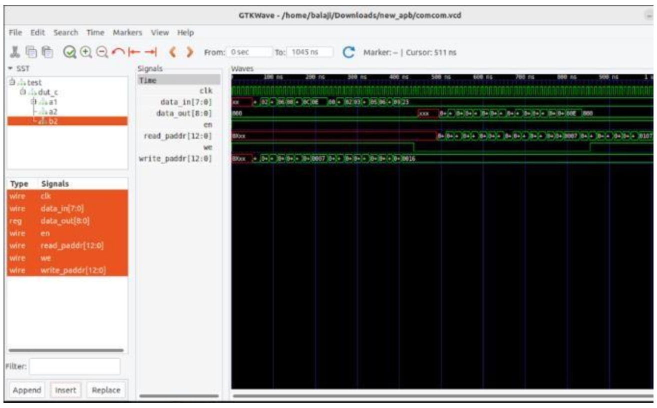


Fig. 2 Memory one interface waveform in APB

B. APB waveform analysis:

The illustration in Fig. 3 depicts a waveform produced via GTKWave, a sophisticated tool utilized for visualizing VCD (Value Change Dump) files generated from digital simulations. This specific waveform encapsulates signals pertaining to an APB (Advanced Peripheral Bus) protocol simulation.

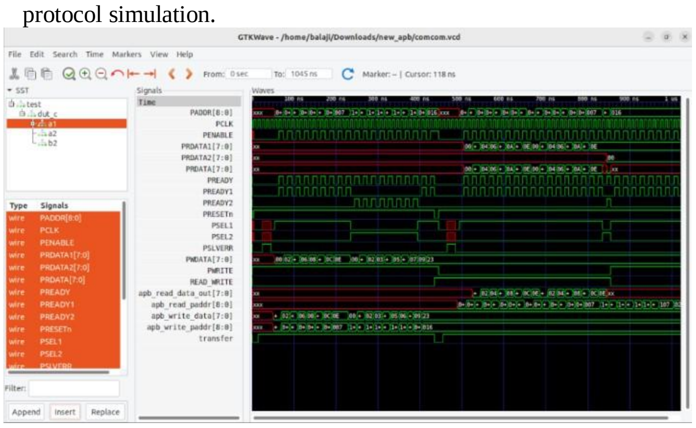


Fig. 3 APB waveform

Signal List (Left Panel):

Exhibits various APB signals including PADDR, PCLK, PENABLE, PREADY, PWRITE, and PSEL, in conjunction with data signals (PRDATA, PWDATA). These signals govern read/write operations on the APB bus.

Waveform Display (Right Panel):

The X-axis denotes time (expressed in nanoseconds, ns). The Y-axis enumerates the APB signals alongside their transitions over time. Distinct signals oscillate between 0, 1, and undefined (x) states, signifying different phases of APB transactions.

Clock Signal (PCLK):

A continuous square waveform that facilitates the synchronization of data transfer.

Write Operation (PWRITE = 1):

Data (PWDATA) is asserted onto the bus for a write operation. PENABLE is activated to signify the active transaction phase. PREADY is asserted by the slave when it is prepared to receive data.

Read Operation (PWRITE = 0):

Data (PRDATA) is manifested on the bus subsequent to the acknowledgment of the read request. PENABLE and PREADY signals orchestrate the data transfer.

Address Selection (PSEL1, PSEL2):

These signals select distinct slave devices on the APB bus.This waveform encapsulates a simulation of APB protocol transactions, illustrating data read and write sequences alongside control signals. It serves to validate the correct functionality of an APB interface, ensuring effective communication between master and slave components within a SoC architecture.

C. MEMORY READ/WRITE OPERATION

The illustrated image in Fig. 4 presents a GTKWave simulation pertaining to a memory interface, wherein diverse control and data signals are scrutinized temporally. The waveform is produced from a VCD (Value Change Dump) file throughout the simulation process to validate memory read/write functionalities.

i)Signal List (Left Panel):

clk: The clock signal that regulates data transactions. data_in[7:0]: An 8-bit input data bus employed for writing into memory. data_out[8:0]: A 9-bit output data bus utilized for reading from memory. en: An enable signal designed to initiate read/write operations. read_addr[12:0]: A 13-bit read address bus delineating the memory location designated for reading. write_addr[12:0]: A 13-bit write address bus indicating the memory location intended for writing. we: A write enable signal that is asserted during the data write process into memory.

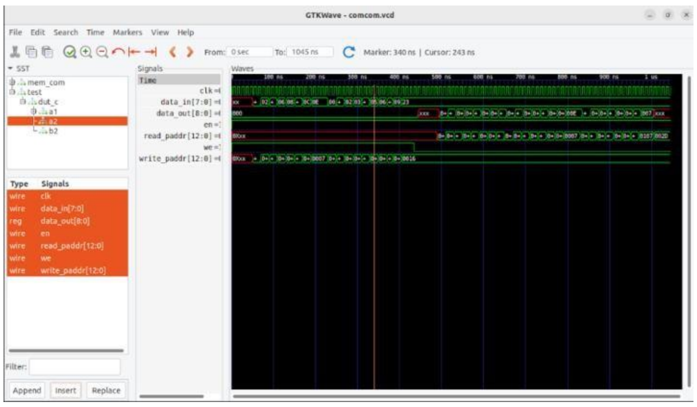


Fig. 4 Memory Read/Write output

ii)Waveform Display (Right Panel):

Clock Signal (clk):

A periodic signal that provides synchronization for memory transactions. Write Operation (we = 1): Data from data_in is deposited into memory at the specified write_addr. The values within write_addr signify distinct memory locations. Read Operation (we = 0, en = 1): Memory is accessed via read_addr, resulting in data being presented on data_out. The transitions observed in data_out signify successful memory reads. The waveform illustrates valid data transactions over time, with data being recorded and subsequently retrieved from specific memory addresses. The undefined values (x) at the initial segment imply uninitialized memory states prior to the inaugural write operation. The synchronization of the we, en, and address buses guarantees accurate memory access.

This waveform encapsulates a memory read/write simulation, serving as an instrumental tool for validating a memory controller or a register file within a digital system. The signal transitions affirm that data is appropriately recorded into memory and subsequently retrieved in accordance with address inputs.

FEATURE	**APB**	**AHB**	**AXI**
Complexity	Simple	Moderate	Complex
Throughput	~50-100 MB/S	~800MB/s – 1.6 GB/s	~6.4 GB/s – 12.8 GB/s
Latency	~10 – 20 cycles	~3 – 10 cycles	~1 – 3 cycles
Burst transfer	Not Supported	Supported	Supported
Pipeline Stages	Single- Stage	Multi- Stage	Multi- Stage
Power Consumption	~1 – 10mW	~10 – 100mW	~100 Mw – 1W
Frequency Range	0 – 100 MHz	100 – 200 MHz	200 MHz – 1GHz+
Usage	Peripheral Registers	Memory and High-Speed Peripheral	High-Speed Memory and Peripheral
Example Devices	GPIO,UART	SDRAM,FLASH Memory	High speed DAM, Processor Interconnects

Table 1: Comparative parameters of a communication system

This Table 1 contrasts the three AMBA protocols: APB, AHB (Advanced High-performance Bus), and AXI. APB is the most straightforward, featuring low complexity, throughput, and power usage, which makes it ideal for peripheral registers such as GPIO and UART. It employs a single-stage pipeline and lacks support for burst transfers. AHB, with moderate complexity, offers increased throughput and average latency. It accommodates burst transfers and employs multi-stage pipelining, ideal for memory and high-speed devices such as SDRAM and Flash Memory. AXI is the most intricate but provides extremely high throughput and minimal latency through multi-stage pipelining and burst transfer capabilities. It uses more energy but is perfect for fast memory and peripheral devices, such as high-speed DAM and processor connections. The selection of a bus is determined by the application's needs regarding speed, complexity, and energy usage

D. APB BASED MEMORY READ/WRITE TRANSACTION

This simulation conducted within GTKWave elucidates the dynamics between an Advanced Peripheral Bus (APB) interface and a memory module, illustrating in Fig. 5 the signal transitions that occur during both read and write transactions.

Control & Data Signals (Left Panel):

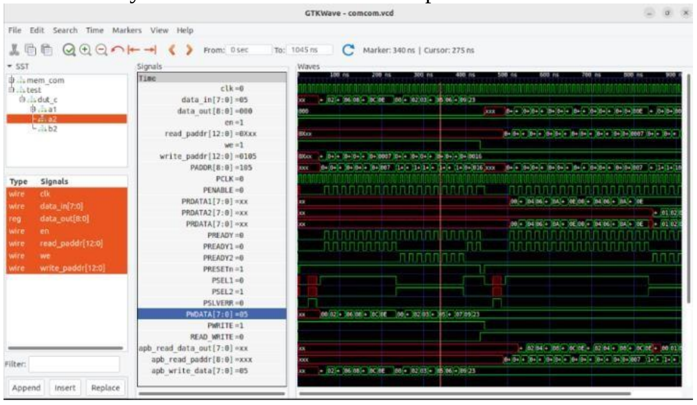
Clock (clk): Regulates the timing of data transactions. Input Data (data_in[7:0]): Represents the data being inputted into the memory. Output Data (data_out[8:0]): Reflects the data that has been retrieved from the memory. Enable (en): Initiates the memory operations. Read Address (read_addr[12:0]): Designates the specific memory location for data retrieval. Write Address (write_addr[12:0]): Identifies the location in memory where data is to be stored. Write Enable (we): Indicates a write operation when activated.

APB Interface Signals:

PADDR[8:0]: Represents the address bus utilized for peripheral access. PCLK: Serves as the clock signal for the APB. PENABLE: Signifies the active data phase during APB transactions. PWDATA[7:0]: Denotes the data that is being written to an APB peripheral. PWRITE: Distinguishes between read (0) and write (1) operations. PREADY: Indicates the readiness of the slave device. PSEL1, PSEL2: Facilitates the selection of APB peripherals for communication purposes**.**

Write Operation (PWRITE=1)

The assertion of PWDATA = 0x05 at the address PADDR = 0x105 signifies a write transaction. The assertion of PENABLE corroborates the presence of an active APB transfer. The memory module successfully stores the value 0x05 at the specified address 0x105.



```
Type  Signals
wire  clk
wire  data_in[7:0]
reg   data_out[8:0]
wire  en
wire  read_paddr[12:0]
wire  we
wire  write_paddr[12:0]
```

```
Time
clk=0
data_in[7:0]=05
data_out[8:0]=000
en=1
read_paddr[12:0]=0Xxx
we=1
write_paddr[12:0]=0105
PADDR[8:0]=105
PCLK=0
PENABLE=0
PRDATA1[7:0]=xx
PRDATA2[7:0]=xx
PRDATA[7:0]=xx
PREADY=0
PREADY1=0
PREADY2=0
PRESETn=1
PSEL1=0
PSEL2=1
PSLVERR=0
PWDATA[7:0]=05
PWRITE=1
READ_WRITE=0
apb_read_data_out[7:0]=xx
apb_read_paddr[8:0]=xxx
apb_write_data[7:0]=05
```

Fig. 5: APB based memory read/write transaction

Read Operation (PWRITE=0)

The activation of the read_addr signal enables the fetching of data from memory. The data out signal exhibits the value that has been retrieved.

Synchronization Between APB & Memory:

The APB bus engages in communication with the memory, thereby ensuring the accurate flow of data. The transitions of signals are appropriately executed, demonstrating the anticipated behaviour of the APB-memory interface.

This waveform substantiates the validity of APB-based memory transactions, confirming the effective transfer of data between an APB master (processor/controller) and a memory module. The observed transitions validate the successful execution of both read and write operations, thereby assisting in the debugging and validation processes of SoC memory interfacing**.**

Conclusion

This work presented the design and implementation of a memory communication system leveraging the APB protocol, underscoring its suitability for low-power and resource-constrained applications. By capitalizing on the protocol's simplicity, the proposed architecture enables reliable and efficient data exchange between memory units while minimizing system overhead and design complexity. Implemented using Verilog HDL and validated through FPGA-based synthesis and simulation, the design was evaluated in terms of latency, power consumption, and throughput. Results indicate that despite APB's limited bandwidth and lack of pipelining, it effectively balances power efficiency with sufficient performance for applications that do not demand high-speed communication. This makes APB a compelling choice for embedded and IoT systems where energy optimization is paramount. Future work may explore enhancements to APB for improved efficiency, hybrid interconnect strategies, and scalability across systems with multiple memory interfaces.

References:

[1]. Sudheer, H., PS, J., Saji, A.K., Avarachan, A., Sebastian, R. and Gopalakrishnan, S., 2021, July. AMBA APB Memory Controller Functional Verification using SystemVerilog. In Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems-ICICNIS.

[2].Karthikeyan, V., Balamurugan, K., Namamula, L.R. and Brindha, F.J., 2023. Development of SRAM-APB protocol interface and verification. Engineering Research Express, 5(4), p.045017.

[3].Velagaleti, S., 2023, March. A Low Power APB with an Area Efficient Structure. In 2023 2nd International Conference for Innovation in Technology (INOCON) (pp. 1-6). IEEE.

[4]..R. Sravani, S. and Deepak, C., 2025. Dynamic analysis of Network on Chip topologies and designing a novel Mesh of Spidergon topology. *Journal of Circuits, Systems and Computers*.

[5]. Liu, Y., Zhang, L., Xu, S., Wang, J., Xu, C., Zhou, L. and Wang, N., 2020, December. Design and verification of UART circuit of SoC based on AMBA bus. In 2020 7th International Conference on Information Science and Control Engineering (ICISCE) (pp. 2370-2374). IEEE.

[6]. DG, H.B., Titiksha, V., Surya, M., Saravanakumar, R. and Elango, S., 2022, December. Performance Enhancement of SoC with Five Port Router by Replacing APB Protocol. In 2022 Smart Technologies, Communication and Robotics (STCR) (pp. 1-5). IEEE.

[7].Sahu, N., Kumar, A. and Kandari, R., 2022, April. Design and verification of APB IP core using different verification methodologies. In 2022 International Conference on Breakthrough in Heuristics And Reciprocation of Advanced Technologies (BHARAT) (pp. 150-154). IEEE.

[8].Jain, P. and Rao, S., 2021, February. Design and verification of advanced microcontroller bus architecture-advanced peripheral bus (AMBA-APB) protocol. In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 462-467). IEEE.

[9].Liu, Y., Li, D., Yang, Z., Zhang, C., Zhang, Y., Li, X., Cao, M. and Yin, S., 2025. A Chiplet Platform for Intelligent Radar/Sonar Leveraging Domain-Specific Reusable Active Interposer. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*.

[10].Krishnegowda, D., 2021, December. Developing a bus functional model for APB slave using universal verification methodology. In 2021 Second International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE) (pp. 1-5). IEEE.

[11] Veerasamy, B., Durga, B.G., Kumaran, T.H.M., Devika, I.V. and Akshaya, M.J., 2023, April. Soldier health detection and position tracking system using LoRaWAN sensor for low power and long-range access. In *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 9-13). IEEE.