IJAREEIE

e-ISSN: 2278-8875
p-ISSN: 2320-3765

International Journal of Advanced Research

in Electrical, Electronics and Instrumentation Engineering

Volume 15, Issue 5, May 2026

INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA

Impact Factor: 9.867

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE)

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 5, May 2026 ||

| DOI:10.15662/IJAREEIE.2026.1505017 |

Design and Verification of an APB based Memory Module

Mounika Y, Dr. Aijaz Ahamad Sharief

M. Tech, Dept. of VLSI Design & Embedded Systems, Shridevi Institute of Engineering and Technology, Tumakuru, India

Professor & HOD, Dept. of ECE, Shridevi Institute of Engineering and Technology, Tumakuru, India

ABSTRACT: The increasing complexity of System-on-Chip (SoC) designs necessitates robust and efficient on-chip communication protocols. The Advanced Peripheral Bus (APB) is widely used for interfacing low-power, low-bandwidth peripherals. This paper presents the design and functional verification of an APB-compliant slave memory module. The Register-Transfer Level (RTL) design is implemented using a Finite State Machine (FSM) in System Verilog, incorporating a 2KB memory array. The verification of an APB slave memory module used Universal Verification Methodology (UVM) environment consisting of sequencer, driver, monitor, scoreboard, agent, and environment components is developed to validate the design functionality. Various test scenarios including reset operation, read/write transfer, back-to-back transactions, read-after-write operations, and invalid address accesses are verified successfully. Simulation results confirm correct APB protocol behavior, reliable memory operation, and robust error handling suitable for modern SoC applications.

I. INTRODUCTION

Modern System-on-Chip (SoC) designs integrate numerous functional blocks, including processors, memory controllers, and peripherals, onto a single die. Efficient on-chip communication is critical for system performance and reliability. The ARM Advanced Microcontroller Bus Architecture (AMBA) provides a family of standard bus protocols to address this need [1] as shown in figure 1. Among these, the Advanced Peripheral Bus (APB) is optimized for simple, low-power, and low-frequency peripheral devices such as UARTs, timers, and GPIO modules. Its non-pipelined, two-phase transfer mechanism (SETUP and ENABLE) reduces hardware complexity and power consumption.

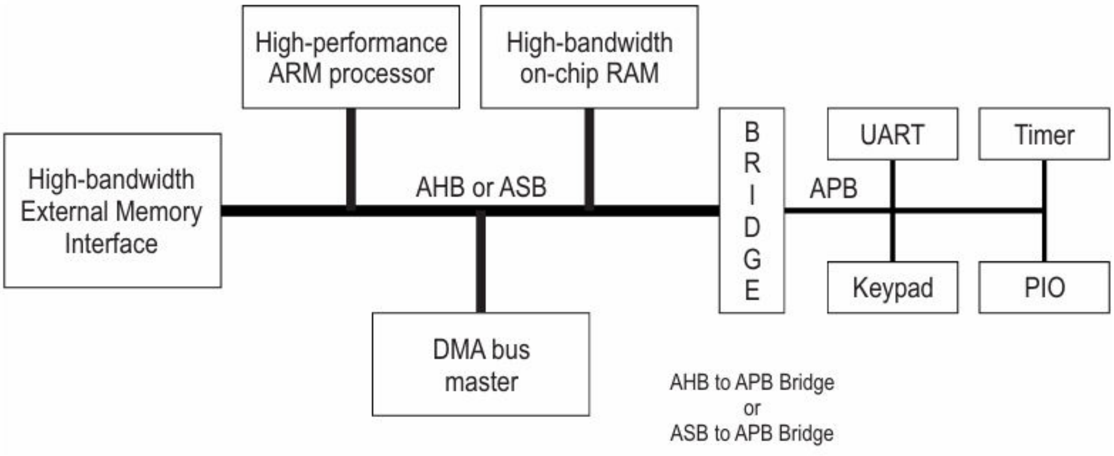


Fig.1 A Typical AMBA System

The primary challenge lies in ensuring that the interface between a peripheral memory and the APB bus strictly adheres to protocol timing, control signal sequencing, and handshake mechanisms. Any deviation can lead to data corruption or system instability. Therefore, a rigorous verification methodology is essential. The Universal Verification Methodology (UVM) offers a standardized, modular, and reusable approach for creating robust testbenches.

This paper details the design and UVM-based verification of an APB slave memory. The objectives are:
(1)to design a synthesizable, FSM-based APB slave memory module in System Verilog
(2)to develop a structured UVM test bench with automated checking
(3)to validate the design against protocol requirements under various test scenarios.

II. LITERATURE SURVEY

Several researchers have proposed APB-based memory systems and verification methodologies for SoC applications.

- B. Sowmya and K. P. Gagana presented a SystemVerilog-based verification environment for memory controller validation using constrained-random testing and coverage-driven verification techniques.
- Shobha Gopalakrishnan developed a layered verification environment for APB memory controller verification focusing on protocol compliance and transaction-level testing.
- Meghana Jain H K and Dr. Punith Kumar M B proposed APB V2.0 verification using assertion-based methodologies for validating timing behavior and wait-state handling.
- IEEE Conference Proceedings Dataset (2023) introduced a UVM-based APB verification framework containing reusable verification components such as driver, monitor, sequencer, and scoreboard.
- Ram Prakash Prasad et al. designed and verified an APB RAM module using SystemVerilog with constrained-random verification and protocol compliance checking.

These works provide the foundation for the proposed APB slave memory design and reusable UVM verification environment

III. PROPOSED SYSTEM

The proposed system consists of an APB slave memory module implemented using SystemVerilog. The architecture includes:

- 2 KB parameterized memory array
- FSM-based transaction controller
- APB slave interface
- Address decoding logic
- Read and write control logic
- Error handling mechanism
- PREADY and PSLVERR response generation

The APB memory supports:

- read/write transfer
- Zero wait-state operation
- Valid and invalid address detection
- Parameterized memory configuration

The design follows the APB two-phase transfer mechanism consisting of :1. Setup and 2. Access Phase

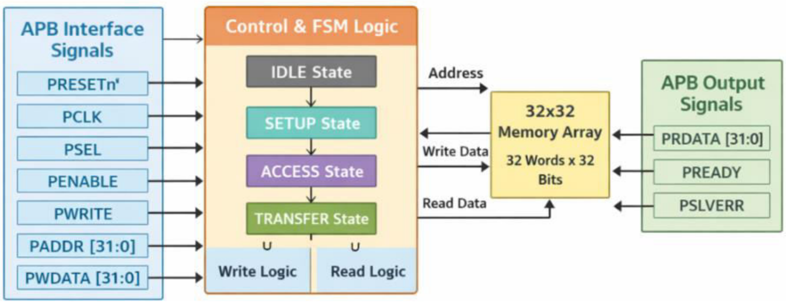


Fig.2 Design of APB Memory

APB slave memory communicates with the APB master through standard APB signals. Master initiates transactions using control and address signals, while slave responds with data and status signals.

IV. APB ARCHITECTURE

The APB slave memory communicates with the APB master through a set of standardized signals responsible for synchronization, control, and data transfer.

Table 1 APB Slave Interface

Signal	**Direction**	**Description**
PCLK	Input	Clock signal that synchronizes all APB transactions and internal operations.
PRESETn	Input	Active-low reset signal used to initialize the memory and control logic.
PSEL	Input	Indicates that the slave memory is selected for a transaction.
PENABLE	Input	Marks the access phase where the actual read/write operation occurs.
PWRITE	Input	Determines the type of operation: high for write and low for read.
PADDR	Input	Carries the address used to access a specific memory location.
PWDATA	Input	Provides write data from the master to the slave memory.
PRDATA	Output	Provides read data from the slave memory to the master.
PREADY	Output	Indicates that the slave has completed the transaction.
PSLVERR	Output	Signals an error condition such as invalid address access.

The APB protocol operates using a finite state machine (FSM) consisting of three states:

- IDLE State
- SETUP State
- ACCESS State

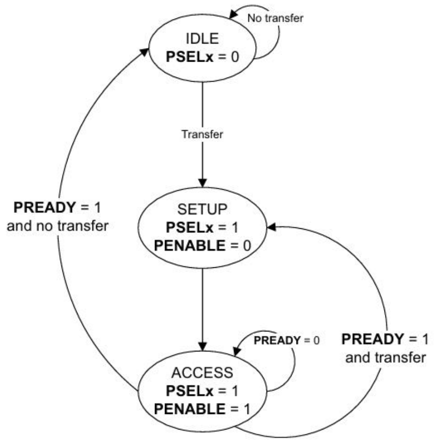


Fig. 3 APB Operation State Diagram

During the setup phase, the master asserts PSEL and provides address and control information. During the access phase, PENABLE is asserted and the actual read or write operation takes place.

The design supports zero wait-state communication where PREADY is asserted immediately after the access phase. Invalid address accesses are detected using address validation logic, and PSLVERR is asserted to indicate protocol errors.

V. UVM VERIFICATION ENVIRONMENT

The verification environment is developed using Universal Verification Methodology (UVM) to achieve modularity, reusability, and scalability.

The UVM test bench includes:

- Sequence
- Sequencer
- Driver
- Monitor
- Scoreboard
- Agent
- Environment
- Reporter

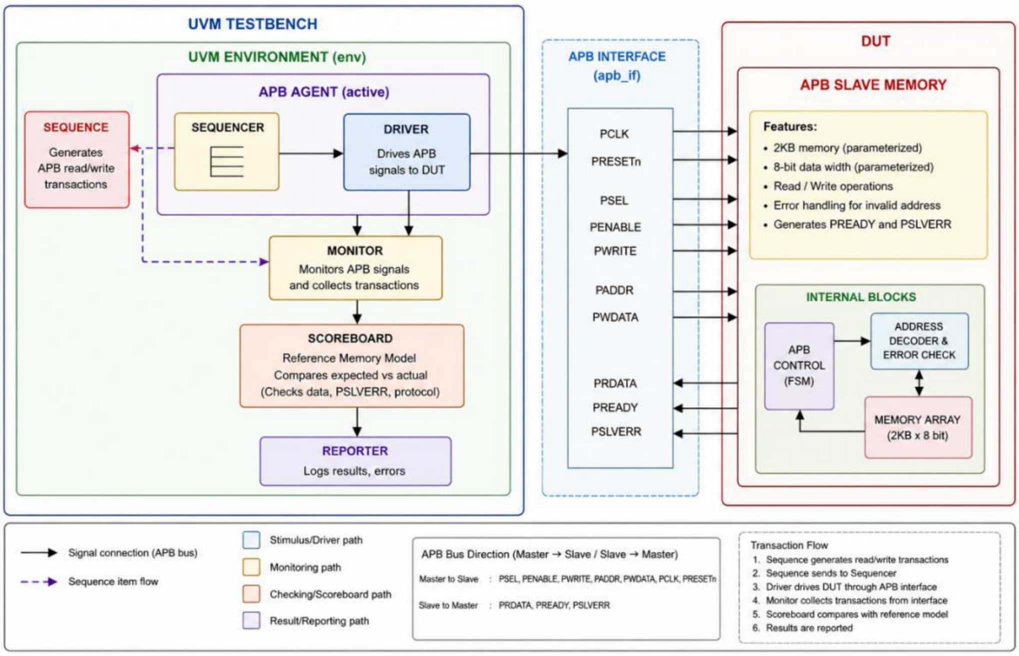


Fig.4 UVM Test bench Architecture for APB Slave Memory

Verification Flow

- Sequence generates APB transactions
- Sequencer controls transaction flow
- Driver converts transactions into APB pin-level signals
- DUT performs memory operation
- Monitor captures DUT activity
- Scoreboard compares expected and actual results

The scoreboard maintains a reference memory model to verify correct DUT functionality automatically.

VI. RESULTS AND DISCUSSION

Simulation results confirm successful implementation of the APB protocol and correct operation of the slave memory module.

Key Observations

- Correct APB setup and access phase sequencing
- Successful read and write operation
- Proper assertion of PREADY and PSLVERR
- Reliable invalid address detection
- Accurate read-after-write functionality
- Stable operation during continuous transactions

Test Scenarios

The verification plan included the following key test scenarios:

1. Reset Test: Verifies initialization after PRESETn assertion.

The DUT initializes correctly when PRESETn is asserted. The FSM enters the IDLE state and all outputs return to default values.

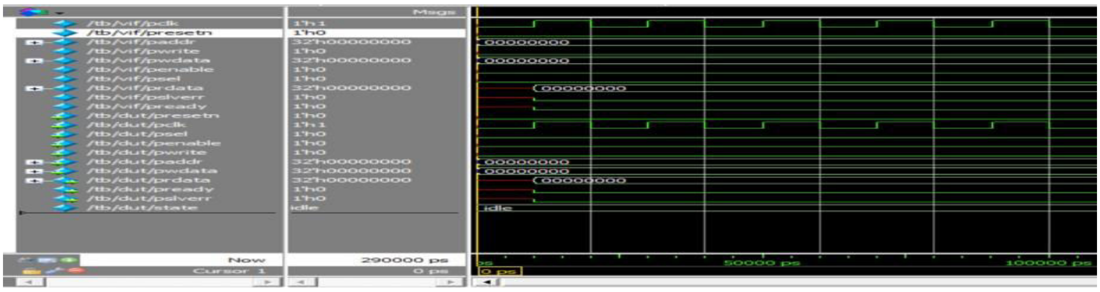


Fig.5 Waveform of Reset

2. Single Write Operation

The write waveform confirms that data on PWDATA is correctly written into the selected memory address following proper APB timing behavior.

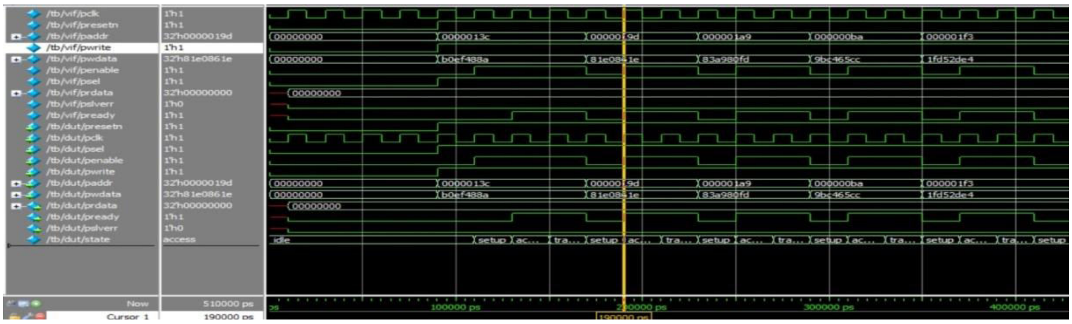


Fig.6 Waveform of Write

3. Back-to-Back Transactions: Checks FSM handling of continuous transactions.

The DUT correctly handles multiple consecutive read and write operations without idle cycles, confirming stable FSM operation and timing synchronization.

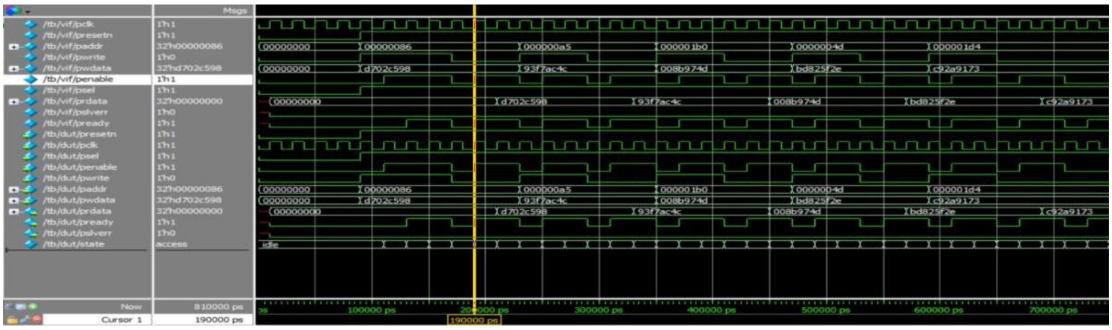

Msgs
/tb/vif/pclk 1'h1
/tb/vif/presetn 1'h1
/tb/vif/paddr 32'h00000086
/tb/vif/pwrite 1'h0
/tb/vif/pwdata 32'hd702c598
/tb/vif/penable 1'h1
/tb/vif/psel 1'h1
/tb/vif/prdata 32'h00000000
/tb/vif/pslverr 1'h0
/tb/vif/pready 1'h1
/tb/dut/presetn 1'h1
/tb/dut/pclk 1'h1
/tb/dut/psel 1'h1
/tb/dut/penable 1'h1
/tb/dut/pwrite 1'h0
/tb/dut/paddr 32'h00000086
/tb/dut/pwdata 32'hd702c598
/tb/dut/prdata 32'h00000000
/tb/dut/pready 1'h1
/tb/dut/pslverr 1'h0
/tb/dut/state access
Now 810000 ps
Cursor 1 190000 ps

Fig.7 Waveform of Read from written

4. Read after Write Operation: Ensures data consistency.
The data written into memory is successfully read back from the same location, confirming proper memory functionality and data integrity.

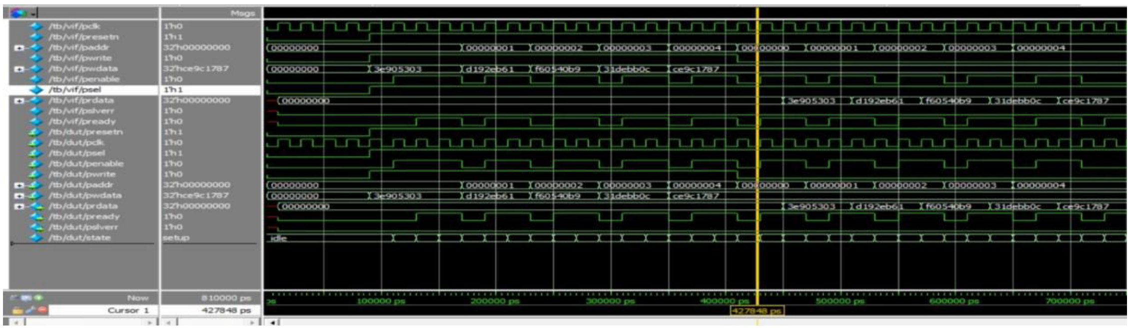

Msgs
/tb/vif/pclk 1'h0
/tb/vif/presetn 1'h1
/tb/vif/paddr 32'h00000000
/tb/vif/pwrite 1'h0
/tb/vif/pwdata 32'hce9c1787
/tb/vif/penable 1'h0
/tb/vif/psel 1'h1
/tb/vif/prdata 32'h00000000
/tb/vif/pslverr 1'h0
/tb/vif/pready 1'h0
/tb/dut/presetn 1'h1
/tb/dut/pclk 1'h0
/tb/dut/psel 1'h1
/tb/dut/penable 1'h0
/tb/dut/pwrite 1'h0
/tb/dut/paddr 32'h00000000
/tb/dut/pwdata 32'hce9c1787
/tb/dut/prdata 32'h00000000
/tb/dut/pready 1'h0
/tb/dut/pslverr 1'h0
/tb/dut/state setup
Now 810000 ps
Cursor 1 427848 ps

Fig.8 Waveform of Read after write

5. Invalid Address Test: Verifies error handling (PSLVERR assertion).
When the master accesses an invalid address, the DUT asserts PSLVERR and prevents unintended memory access, confirming reliable error handling behavior.

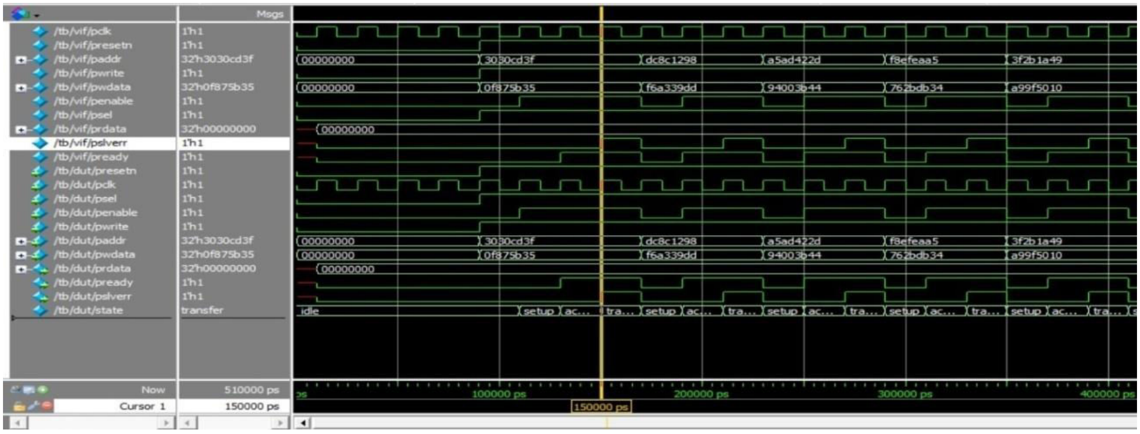

Msgs
/tb/vif/pclk 1'h1
/tb/vif/presetn 1'h1
/tb/vif/paddr 32'h3030cd3f
/tb/vif/pwrite 1'h1
/tb/vif/pwdata 32'h0f875b35
/tb/vif/penable 1'h1
/tb/vif/psel 1'h1
/tb/vif/prdata 32'h00000000
/tb/vif/pslverr 1'h1
/tb/vif/pready 1'h1
/tb/dut/presetn 1'h1
/tb/dut/pclk 1'h1
/tb/dut/psel 1'h1
/tb/dut/penable 1'h1
/tb/dut/pwrite 1'h1
/tb/dut/paddr 32'h3030cd3f
/tb/dut/pwdata 32'h0f875b35
/tb/dut/prdata 32'h00000000
/tb/dut/pready 1'h1
/tb/dut/pslverr 1'h1
/tb/dut/state transfer
Now 510000 ps
Cursor 1 150000 ps

Fig.9 Waveform of Write error

Table 2: Summary of Verification Results

Test Scenario	**Expected Result**	**Actual Result Status**
Reset Test	DUT initializes to IDLE state	DUT in IDLE,outputs reset Pass
Back-to-Back Writes	All writes successful	All addresses updated correctly Pass
Back-to-Back Write Reads	All reads return correct data	PRDATA valid for each cycle Pass
Read after Write	Read data equals written data	Match confirmed Pass
Invalid Address	PSLVERR asserted	PSLVERR = 1 Pass

VII. ADVANTAGES

- Simple and efficient APB architecture
- Low power consumption
- Easy SoC integration
- Parameterized and scalable design
- Reusable UVM verification environment
- Reliable protocol-compliant operation
- Robust error detection mechanism

VIII. APPLICATIONS

- SoC peripheral subsystems
- Embedded systems
- UART and GPIO controllers
- Memory-mapped peripherals
- Low-power communication systems
- Register-based control interfaces
- Educational and research-oriented SoC projects

IX. CONCLUSION

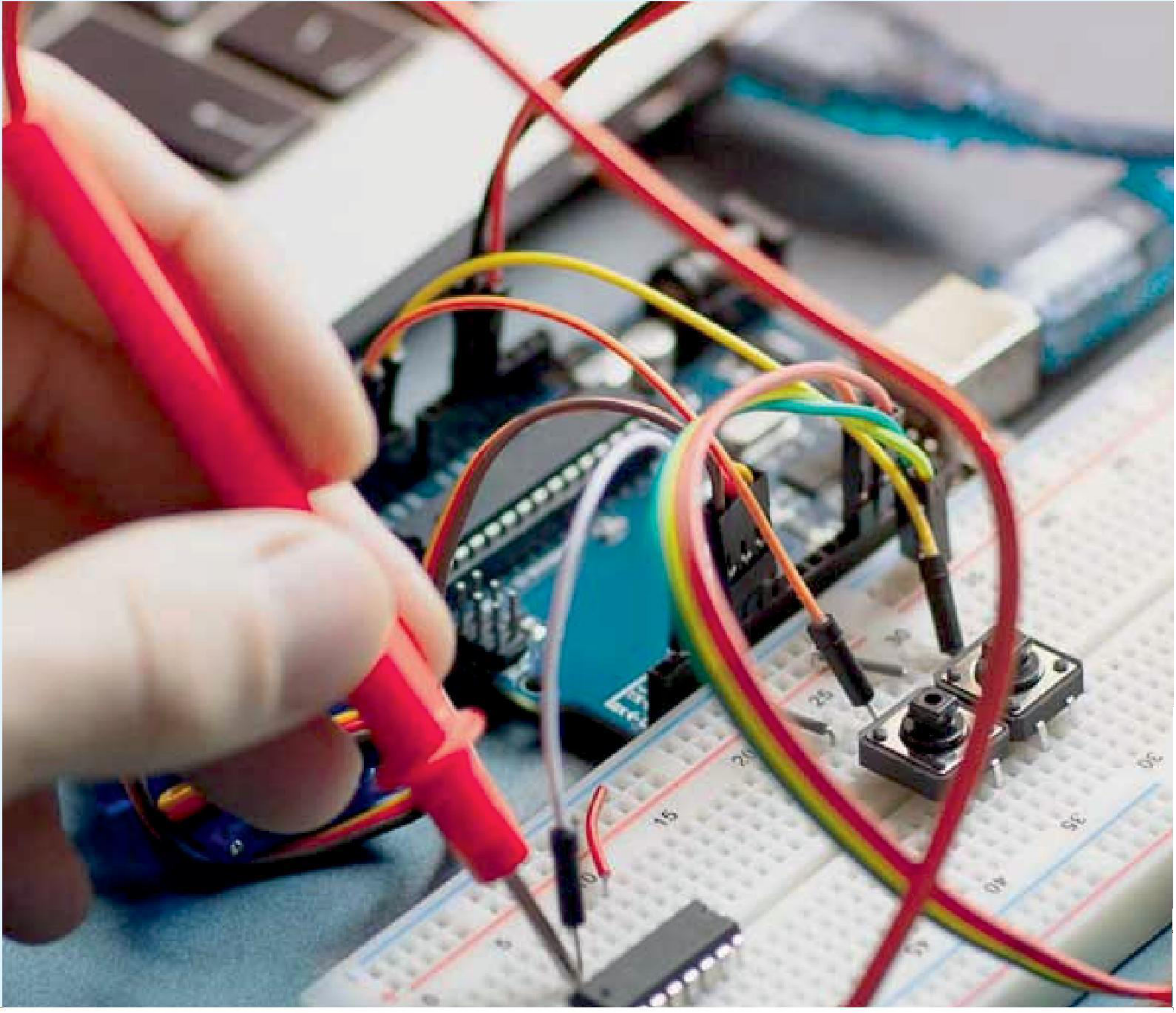
An APB-compliant slave memory module was successfully designed and verified using SystemVerilog and UVM methodology. The proposed design supports protocol-compliant read and write operations with reliable synchronization and error handling. A reusable UVM verification environment was developed to validate the DUT functionality under multiple operating scenarios including reset handling, continuous transactions, invalid address access, and read-after-write verification.

Simulation results demonstrate correct APB timing behavior, stable FSM operation, and reliable memory communication suitable for low-power SoC applications. The project provides practical understanding of APB protocol design and modern verification methodologies used in VLSI and embedded system development.

Future enhancements may include wait-state insertion, parameterized data width support, functional coverage implementation, assertion-based verification, and integration with AHB/APB bridge architectures.

REFERENCES

[1] ARM Limited, “AMBA 3 APB Protocol Specification,” ARM IHI 0024B, Cambridge, UK, 2006.
[2] ARM Limited, “AMBA 4 Specification,” ARM IHI 0034A, Cambridge, UK, 2010.
[3] S. Palnitkar, “Verilog HDL: A Guide to Digital Design and Synthesis”, 2nd ed., Prentice Hall, 2003.
[4] C. Spear and G. Tumbush, “SystemVerilog for Verification”, 3rd ed., Springer, 2012.
[5] B. Sowmya and K. P. Gagana, “A SystemVerilog Approach for Verification of Memory Controller,” IJEAT, vol. 9, no. 3, pp. 245–249, 2020.



International Journal of Advanced Research

in Electrical, Electronics and Instrumentation Engineering

9940 572 462 6381 907 438 ijareeie@gmail.com

www.ijareeie.com

Scan to save the contact details