A REVIEW ON ADVANCED PERIPHERAL BUS INTERFACE PROTOCOL

E.PRAVEEN[1] YVNM SARMA [2]

[1] M.Tech Student, Dept of ECE, RK College of Engineering, Krishna-521456, A.P., India

[2] Assistant Professor, Dept of ECE, RK College of Engineering, Krishna-521456, A.P., India

ABSTRACT

The **Advanced Peripheral Bus (APB)** is a low-power, low-bandwidth bus that is part of the **Advanced Microcontroller Bus Architecture (AMBA)** developed by ARM. It is widely used to connect low-speed peripheral devices such as timers, UARTs, GPIOs, SPI, and I²C controllers in System-on-Chip (SoC) designs. Since APB is responsible for communication between the processor and peripheral devices, verifying its functionality is essential to ensure reliable data transfer and protocol compliance.

This project presents the **verification of the APB protocol using the Universal Verification Methodology (UVM)**. UVM is a standardized verification framework built on SystemVerilog that enables the development of reusable, scalable, and modular verification environments. The project implements a complete UVM-based testbench consisting of transaction objects, sequences, sequencer, driver, monitor, agent, environment, scoreboard, and functional coverage components to verify the APB design.

The verification environment generates both directed and randomized read/write transactions to validate the APB protocol under different operating conditions. The monitor captures the DUT activity, while the scoreboard compares the expected and actual results to identify mismatches. Functional coverage is collected to ensure that all important protocol features, including read operations, write operations, valid address ranges, wait states, and transfer types, are exercised during simulation.

Simulation results confirm that the APB design behaves according to the protocol specification, with successful verification of all functional scenarios. The UVM-based environment improves verification efficiency, enhances code reusability, and provides comprehensive functional coverage. This project demonstrates an industry-standard verification approach for AMBA APB and serves as a strong foundation for verifying more complex bus protocols such as AHB and AXI.

INTRODUCTION

The rapid growth of **System-on-Chip (SoC)** technology has led to the integration of multiple processors, memories, and peripheral devices onto a single chip. As the complexity of these systems increases, efficient communication between different hardware components becomes essential. To address this challenge, standardized on-chip communication protocols are widely adopted. One of the most popular standards is the **Advanced Microcontroller Bus Architecture (AMBA)** developed by ARM. AMBA defines a family of bus protocols that enable reliable communication among processors, memories, and peripheral devices within an SoC.

Among the AMBA protocols, the **Advanced Peripheral Bus (APB)** is specifically designed for low-bandwidth and low-power peripheral devices. APB provides a simple and efficient interface for peripherals such as Universal Asynchronous Receiver Transmitters (UARTs), General Purpose Input/Output (GPIO), timers, watchdog timers, Serial Peripheral Interface (SPI), and Inter-Integrated Circuit (I^2C) controllers. Unlike high-performance buses such as AHB and AXI, APB has a non-pipelined architecture with minimal control signals, making it easy to implement while consuming less power and silicon area.

As modern semiconductor devices become increasingly complex, functional verification has become one of the most important phases in the design cycle. Industry reports indicate that nearly **70% of the overall design effort** is dedicated to verification. An undetected design error in a communication protocol may lead to incorrect data transfers, system failures, or costly redesigns after fabrication. Therefore, it is essential to verify the APB protocol thoroughly before implementation in silicon.

Traditional Verilog-based testbenches have several limitations, including poor reusability, limited scalability, and manual test development. These challenges become more significant as the complexity of digital designs increases. To overcome these limitations, the semiconductor industry has widely adopted the **Universal Verification Methodology (UVM)**, which is built on System Verilog. UVM provides a standardized framework for developing modular, reusable, and scalable verification environments. It supports constrained random stimulus generation, functional coverage, transaction-level modeling, automated reporting, and reusable verification components.

PROBLEM STATEMENT

With the rapid advancement of **System-on-Chip (SoC)** technology, modern electronic systems integrate multiple processors, memories, and peripheral devices onto a single chip. These components communicate through standardized bus protocols, among which the **Advanced Peripheral Bus (APB)** is widely used for connecting low-speed peripherals such as UART, GPIO, SPI, I²C, timers, watchdog timers, and other control devices. Although the APB protocol is relatively simple compared to high-performance buses like AHB and AXI, it plays a critical role in ensuring reliable communication between the processor and peripheral devices.

In practical SoC designs, any error in the implementation of the APB protocol can lead to incorrect data transfers, communication failures, timing violations, and malfunctioning of peripheral devices. Such errors may remain undetected if the verification process is not comprehensive. Detecting these bugs after chip fabrication is extremely expensive and time-consuming, often requiring redesign and additional manufacturing costs. Therefore, verifying the APB protocol before hardware implementation is an essential step in the digital design flow.

Traditional Verilog-based verification methods rely mainly on manually written directed test cases. These methods have several limitations, including poor code reusability, limited scalability, inadequate test coverage, and increased debugging effort. As the number of test scenarios increases, it becomes difficult to verify all possible combinations of addresses, data values, read/write operations, and protocol timing conditions using conventional verification techniques.

Another major challenge is ensuring that every feature of the APB protocol is exercised during simulation. Without proper functional coverage analysis, some protocol scenarios may remain untested, increasing the possibility of undetected functional bugs. Manual comparison of simulation outputs is also inefficient and prone to human error, making the verification process slower and less reliable.

To overcome these challenges, a standardized and reusable verification methodology is required. The **Universal Verification Methodology (UVM)** provides an efficient framework for developing modular and scalable verification environments. It supports constrained-random stimulus generation, automatic result checking using scoreboards, protocol monitoring, and functional coverage collection. These features significantly improve verification quality while reducing development time and maintenance effort.

Therefore, the problem addressed in this project is to develop a complete UVM-based verification environment capable of thoroughly verifying the functionality and protocol

compliance of the Advanced Peripheral Bus (APB). The verification environment must validate read and write transactions, protocol timing, address decoding, wait-state handling, and data integrity while achieving high functional coverage and automatically detecting protocol violations.

The successful implementation of this verification environment ensures that the APB design functions correctly under various operating conditions and provides a reusable verification framework that can be extended to more complex AMBA protocols in future semiconductor designs.

OBJECTIVES OF THE PROJECT

The primary objective of this project is to develop a robust and reusable verification environment for the **Advanced Peripheral Bus (APB)** protocol using the **Universal Verification Methodology (UVM)**. The verification environment is designed to ensure that the APB protocol operates according to the AMBA specification under various operating conditions. By adopting UVM, the project aims to improve verification efficiency, maximize functional coverage, and reduce the time required to identify and debug design errors.

The specific objectives of this project are as follows:

1. To understand the APB protocol

- Study the architecture and working principles of the Advanced Peripheral Bus (APB).
- Understand the APB communication process, including the Setup Phase and Access Phase.
- Analyze APB timing diagrams, control signals, and transaction flow.

2. To develop a UVM-based verification environment

- Design a modular and reusable UVM testbench for APB protocol verification.
- Implement standard UVM components such as transaction, sequence, sequencer, driver, monitor, agent, scoreboard, environment, and test classes.
- Ensure that the verification environment follows industry-standard UVM guidelines.

3. To verify APB read and write operations

- Validate correct execution of APB write transactions.
- Verify APB read transactions for different addresses and data values.
- Ensure correct data transfer between the master and peripheral devices.

4. To verify APB protocol compliance

- Check the proper sequencing of APB control signals such as **PSEL**, **PENABLE**, **PWRITE**, and **PREADY**.

- Verify protocol timing during both setup and access phases.
- Detect any protocol violations or incorrect signal behavior.

5. To perform constrained-random verification

- Generate randomized transactions using SystemVerilog constraints.
- Execute multiple verification scenarios to improve bug detection.
- Verify the design under various combinations of addresses, data values, and transaction types.

6. To implement automatic result checking

- Develop a scoreboard that compares expected outputs with actual outputs received from the Design Under Test (DUT).
- Automatically identify mismatches and report functional errors.
- Reduce manual waveform analysis and debugging effort.

7. To achieve high functional coverage

- Develop covergroups and coverpoints to measure verification completeness.
- Verify that all important APB features, including read operations, write operations, address ranges, wait states, and reset conditions, are exercised during simulation.
- Analyze coverage reports to identify any untested scenarios.

8. To improve verification reusability

- Build reusable verification components that can be integrated into future projects.
- Design the verification environment so that it can be extended for verifying other AMBA protocols such as AHB and AXI.

9. To analyze simulation results

- Simulate the APB design using a SystemVerilog/UVM-compatible simulator.
- Verify the correctness of protocol operations through simulation waveforms.
- Analyze transaction logs, protocol behavior, and functional coverage reports.

10. To develop an industry-standard verification methodology

- Demonstrate the practical application of UVM in protocol verification.
- Gain hands-on experience with modern verification techniques used in semiconductor industries.
- Establish a scalable verification framework suitable for complex digital designs.

SUMMARY OF OBJECTIVES

The main objectives of this project can be summarized as follows:

- To study the APB protocol and its operation.

- To design and implement a reusable UVM verification environment.
- To verify APB read and write transactions.
- To validate APB protocol timing and signal behavior.
- To perform constrained-random verification.
- To implement automatic checking using a scoreboard.
- To achieve maximum functional coverage.
- To improve verification quality and reduce debugging effort.
- To analyze simulation results and verify protocol correctness.
- To develop an industry-standard verification environment for future protocol verification projects.

SCOPE OF THE PROJECT

The scope of this project is to develop and validate a **Universal Verification Methodology (UVM)** based verification environment for the **Advanced Peripheral Bus (APB)** protocol. The project focuses on verifying the functional correctness and protocol compliance of the APB interface by generating different types of transactions, monitoring bus activity, checking output correctness, and measuring functional coverage. The verification environment is designed to ensure that the APB protocol operates according to the AMBA specification under normal and boundary operating conditions.

The project includes the development of a complete UVM testbench consisting of standard verification components such as the transaction class, sequence item, sequence, sequencer, driver, monitor, agent, scoreboard, environment, and test class. These components are integrated to create a reusable and scalable verification framework capable of validating the APB Design Under Test (DUT).

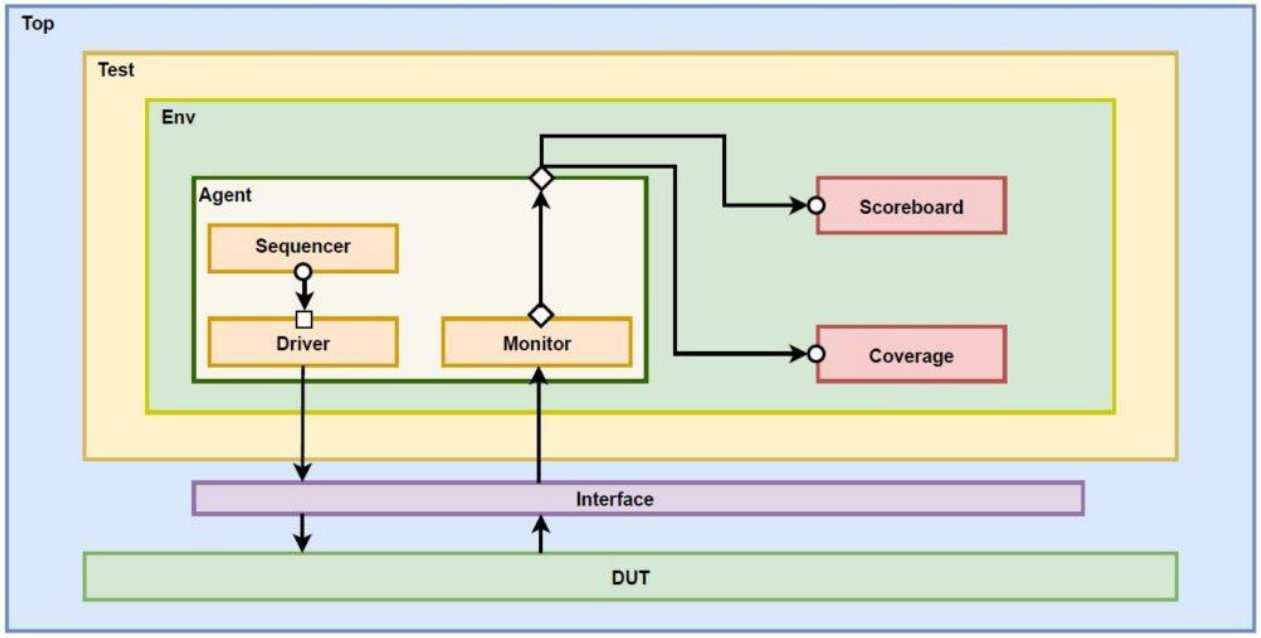
The verification environment supports both **read** and **write** transactions. Different address locations and data values are generated using directed and constrained-random test cases to verify the functionality of the APB protocol. The driver applies these transactions to the DUT, while the monitor observes the protocol signals and collects transaction information. The scoreboard automatically compares the expected and actual outputs to identify any functional mismatches during simulation.

Another important aspect of the project is **functional coverage analysis**. Covergroups and coverpoints are implemented to measure how effectively the verification environment exercises the protocol features. Coverage is collected for various parameters such as read operations, write operations, address ranges, data values, reset conditions, and wait-state handling. The generated coverage report

helps determine whether all important functional scenarios have been verified.

The project also includes simulation and waveform analysis using SystemVerilog/UVM-compatible simulation tools. The simulation results are analyzed to verify protocol timing, signal sequencing, and transaction correctness. Waveforms are examined to ensure that APB control signals such as **PSEL**, **PENABLE**, **PWRITE**, **PREADY**, **PADDR**, **PWDATA**, and **PRDATA** operate according to the protocol specification.

ARCHITECTURE



CONCLUSION AND FUTURE SCOPE

This project successfully developed a UVM-based verification environment for the Advanced Peripheral Bus (APB) protocol. The environment verifies APB read and write operations using reusable UVM components such as sequences, drivers, monitors, scoreboards, and agents. Functional coverage and automatic result checking ensure reliable verification and improve overall verification efficiency. The project demonstrates that UVM is an effective methodology for verifying modern digital communication protocols**.**

FUTURE SCOPE

The project can be enhanced in the future by:

- Extending the verification environment to AHB and AXI protocols.
- Implementing advanced constrained-random and regression testing.
- Integrating assertions and formal verification techniques.
- Supporting complex SoC-level verification environments.
- Improving coverage using advanced verification strategies**.**

REFERENCES

1. Accellera Systems Initiative, **Universal Verification Methodology (UVM) 1.2 User's Guide**, Accellera Systems Initiative, 2015.
2. Arm Holdings, **AMBA 3 APB Protocol Specification**, ARM IHI 0024C, 2004.
3. Chris Spear and Greg Tumbush, *SystemVerilog for Verification: A Guide to Learning the Testbench Language and Functional Verification*, 3rd Edition, Springer, 2012.
4. IEEE, **IEEE Standard 1800-2017 – SystemVerilog Language Reference Manual**, IEEE Standards Association, 2017.
5. Janick Bergeron, *Writing Testbenches Using SystemVerilog*, Springer, 2006.
6. Accellera Systems Initiative, **UVM Class Reference Manual**, Version 1.2, 2015.
7. Stuart Sutherland, *SystemVerilog for Design: A Guide to Using SystemVerilog for Hardware Design and Modeling*, Springer, 2006.
8. IEEE, **IEEE Standard 1666-2011 – SystemC Language Reference Manual**, IEEE Standards Association, 2011.
9. Accellera Systems Initiative, **SystemVerilog Standard Class Library Documentation**, Accellera Systems Initiative.
10. EDA Playground, *EDA Playground Documentation*, Available online.