APB(Advance Peripheral BUS)

Revision History

| Revision | Description | Date | Modified By |
| --- | --- | --- | --- |
| 1.0 | Initial Document | 13/05/2023 |  |
| 1.1 | -- | -- | -- |

Table of Contents

[Chapter 1: RAM Overview 6](#_Toc406486740)

[Chapter 2: RAM features 8](#_Toc406486741)

[Chapter 3: RAM Verification Plan 9](#_Toc406486744)

[3.1 Feature Extraction 9](#_Toc406486745)

[3.2 Coverage Plan 9](#_Toc406486746)

[3.3 Checker Plan 9](#_Toc406486747)

[3.4 Verification Environment Development 9](#_Toc406486747)

[3.5 Test suite development 9](#_Toc406486750)

[3.5.1 Directed Testcases 9](#_Toc406486751)

[3.5.2 Random Testcases 10](#_Toc406486752)

[Chapter 4: RAM Verification Environment Development 11](#_Toc406486753)

[4.1 Block Diagram 11](#_Toc406486754)

[4.2 Verification Architecture 11](#_Toc406486754)

[4.3 RAM components 12](#_Toc406486755)

[4.3.1 Transcation Class 12](#_Toc406486756)

[4.3.2 Generator Class 12](#_Toc406486757)

[4.3.3 Driver 13](#_Toc406486758)

[4.3.4 Monitor 14](#_Toc406486759)

[4.3.5 Reference Model 16](#_Toc406486760)

[4.3.6 Score Board 16](#_Toc406486761)

[Chapter 5: Running Simulation 24](#_Toc406486772)

[Chapter 6: Closure Reports 24](#_Toc406486772)

List of Figures

List of Tables

**No table of figures entries found.**

# APB Overview

* The APB protocol is a low-cost interface, optimized for minimal power consumption and reduced interface complexity. The APB interface is not pipelined and is a simple, synchronous protocol.
* Every transfer takes at least two cycles to complete.
* The APB interface is designed for accessing the programmable control registers of peripheral devices. APB peripherals are typically connected to the main memory system using an APB bridge.
* For example, a bridge from AXI to APB could be used to connect a number of APB peripherals to an AXI memory system. APB transfers are initiated by an APB bridge. APB bridges can also be referred to as a Requester. A peripheral interface responds to requests. APB peripherals can also be referred to as a Completer. This specification will use Requester and Completer.

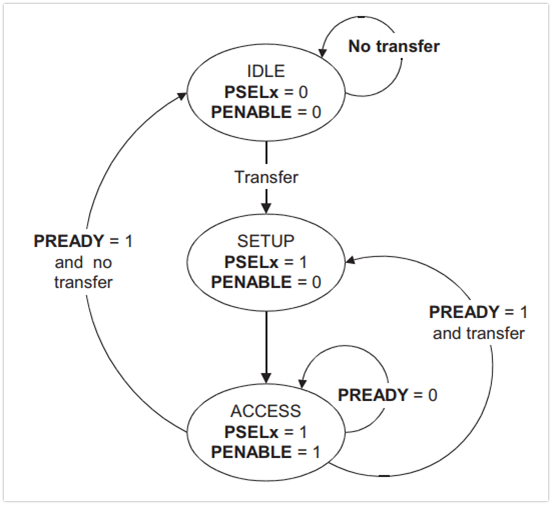


Figure 1.1 State Diagram of APB

# APB features

**The key features of the APB are:**

* + **At least** 2 Cycle DATA Transfer
  + Write without no wait
  + Write with no wait
  + Read without no wait
  + Read with no wait
  + Error Response
  + Low power consumption
  + Low cost
  + Separate Data Buses for Read and Write
  + Invalid Addresses 21 to 31(DUT Sepcific)

# APB Verification Plan

Verification involves studying the relevant specifications, extracting features from it that are to be tested, devising a strategy as to how these features are to be tested, developing a verification environment based on the strategy, writing testcases to cover all the scenarios and achieving 100% functional coverage figures.

## Feature Extraction

“Feature Extraction” involves listing out features to be tested from the specification. A feature list (spreadsheets) has been prepared using the ram\_feature\_list document.

ram\_feature\_list.xlsx

The above spreadsheet lists out features to be tested in the corresponding specification, assigns a feature id to them and tells how the feature is to be tested (testcase name or checker task name).

## Coverage Plan

A functional coverage plan needs to be made based on the feature extraction document. This plan lists out the various combinations of stimuli that need to be generated for the proper verification of the UVC. This has been included in the feature extraction spreadsheet itself.

## Checker Plan

A checker plan needs to be made based on the feature extraction document. Implementations for the features marked as “checker” are elaborated here. We have included it in the feature extraction spreadsheet itself.

## Verification Environment Development

Our environment is based on SystemVerilog.

## Test suite development

### Directed Testcases

Testcases written to test specific areas of the UVC or to generate a specific kind or sequence of transactions is known as a directed testcase. These testcases are helpful in the initial and final stages of verification. In the initial stages, when neither the verification environment nor the UVC is matured, these testcases help in checking and correcting specific pieces of code in both the UVC and the verification environment. In the final stages, they are used to hit specific functional or code coverage areas.

### Random Testcases

Random testcases are written to test the UVC extensively. Random scenarios are generated based on constraints provided in the testcase. These testcases are run several times with different seed numbers to generate different scenarios to achieve more functional coverage figure.

# RAM Verification Environment Development

## RAM Block Diagram

## Verification Architecture

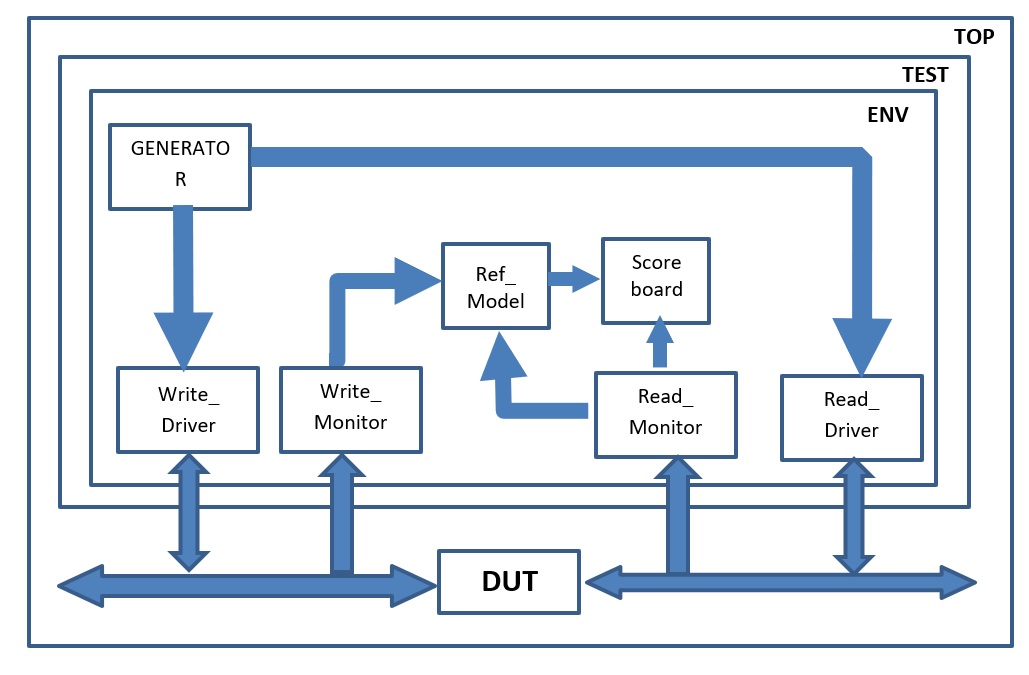


Figure 4.2: RAM Verification Architecture in System Verilog

## RAM Component

### Transaction Class

Write a description of Transaction class

### Generator Class

Write a description of Generator class

### Driver Class

Write a description of Driver class

### Monitor Class

Write a description of Monitor class

### Reference Model

Write a description of Reference class

### Scoreboard

Write a description of Scoreboard class

# Chapter 5: Running Simulation

# Chapter 6: Closure Report