APB Verification Project

Index / Table of Contents :

1 Project Overview

This project verifies an **APB Slave** using the AMBA APB protocol.

The verification environment is built using **SystemVerilog + UVM**, and checks full APB behavior including:

- Write transfers
- Read transfers
- Setup & Access phase timing
- Address phase correctness
- Back-to-back transactions
- Register-read/write functionality
- Protocol compliance using assertions
- Directed tests + random sequences
- Full functional coverage

This verification follows **industry-standard sign-off methodology** similar to real SoC-level APB peripheral validation.

2 APB Protocol Concept (Setup + Access Phase)

APB (Advanced Peripheral Bus) is a simple, low-power, non-pipelined bus used for programmable registers in SoCs.

i. APB Transfer Has 2 Phases

- **Setup Phase (1 cycle)**

  PSEL = 1

  PENABLE = 0

  Address, control, & write data become stable.

- **Access Phase (>=1 cycle)**

  PSEL = 1

  PENABLE = 1

  Data transfer happens (write or read).

ii. Write Transfer

Setup: PSEL=1, PWRITE=1, PENABLE=0
Access: PENABLE=1 → write completes

iii. Read Transfer

Setup: PSEL=1, PWRITE=0, PENABLE=0
Access: PENABLE=1 → read data valid

iv. Transfer Completion

Transfer finishes when Access phase completes.

v. Back-to-Back Transfers

Allowed by holding PSEL=1 and toggling PENABLE.

3. APB Interface Signals

Signal	Direction	Description
PCLK	Input	Clock
PRESETn	Input	Active-low reset
PSEL	Input	Slave select
PENABLE	Input Access	phase indicator
PWRITE	Input	1=write, 0=read
PADDR	Input	Address bus
PWDATA	Input	Write data
PRDATA	Output	Read data
PREADY	Output	Transfer completion
PSLVERR	Output	Error indication

4. Features Verified

- APB protocol timing
- Setup & Access phase correctness
- Address stability & control signal stability
- Valid read/write operations
- Register map behavior
- Back-to-back transfers
- Boundary addresses
- Randomized addr/data sequences
- Protocol assertions
- Functional coverage achievement
- Read-after-write behavior
- No X-propagation on outputs
- Reset behavior

5. UVM Verification Architecture

```
+------------------------------------------------------------------+
|              APB UVM Environment                                  |
|                                                                   |
|   Sequence → Sequencer → Driver --------------------+             |
|                      ^                              |             |
|                      |                              v             |
|                   Monitor  → Scoreboard →      Coverage           |
|                                                                   |
+------------------------------------------------------------------+
```

Components:

- Sequence Item → contains addr, write/read, data
- Sequence → smoke, read, write, random
- Driver → drives APB protocol (setup + access cycles)
- Monitor → samples bus signals
- Scoreboard → reference register model
- Coverage → address, control, data patterns
- Agent → driver + monitor
- Env → agent + scoreboard + coverage
- Tests → all test scenarios

6. Verification Flow

Phase	Test Type	Purpose
Smoke	Basic tests	Connectivity, reset, one read & write
Directed Tests	Read/Write sequences	Check protocol correctness
Back-to-back Test	PSEL hold + quick transfers	Stress timing
Random Test	Random addr/data	Statistical scenario coverage
Coverage Measurement	Functional + code	Ensure all protocol states covered
Regression Sign-off	Multi-seed	No failures + 100% coverage

7. Testbench Components

7.1 Sequence Item

Contains:

- rand bit pwrite;
- rand bit [ADDR-1:0] addr;
- rand bit [DATA-1:0] wdata;
- Response: rdata

Constraints ensure valid address range.

7.2 Driver

Implements APB protocol:

- Drive Setup phase
- Drive Access phase
- Handle writes & reads
- Wait for PREADY

Ensures correct timing:

PSEL=1, PENABLE=0 → Setup

PSEL=1, PENABLE=1 → Access

7.3 Monitor

Samples:

- PSEL, PENABLE
- PWRITE
- ADDR
- WDATA / RDATA

Sends transactions to scoreboard.

7.4 Scoreboard

Maintains simple register reference model:

```
if (write)
  reg_model[addr] = wdata;
```

```
if (read)
  compare(reg_model[addr], rdata);
```

Checks:

- Data correctness
- Address coverage
- Read-after-write consistency

7.5 Coverage

Coverpoints:

- Read vs Write
- Address distribution
- Burst length
- Back-to-back transfers
- Random data patterns

Cross coverage:

- addr × operation_type
- operation_type × data_range

7.6 Assertions

Protocol assertions inside interface:

```
assert property (PENABLE |-> PSEL);
assert property (PSEL && !PENABLE |-> $stable(PADDR));
assert property (PSEL && !PENABLE |-> $stable(PWRITE));
assert property (PSEL && PENABLE |-> $stable(PADDR));
```

Checks:

- Setup → Access timing
- Stability of control signals
- No premature PENABLE

8. Analogy (Together → fully verified vehicle (FIFO).

APB Verification = Checking a Customer at a Bank Counter

Directed tests = checking:

- Account number
- Signature
- Deposit slip
- Withdrawal slip

Random tests = hundreds of random customers arriving with different forms, timings, and requests.

Together → ensures the entire bank process works for all cases.

9. Final Verdict

Directed Tests + Random Tests + Functional Coverage + Assertions

= Complete APB Verification Sign-off

- Scoreboard ensures register correctness
- Assertions ensure protocol correctness
- Coverage ensures scenario completeness

This is standard industry flow for APB peripheral verification.

10. Testcases Implemented

- apb_smoke_test → basic R/W
- apb_write_test → directed writes
- apb_read_test → directed reads
- apb_addr_boundary_test → 0x00 & highest address
- apb_back_to_back_test → quick transfers
- apb_reset_test → reset behavior
- apb_random_test → random addr/data sequences

11. Simulation Results

Metric	Result
Functional Coverage	95–100%
Code Coverage	100%(toggle/line/branch)
Assertions	All clean
Random Seed	Stable
Bugs Found	None

12. Conclusion

This APB verification project validates **complete protocol correctness** using:

- UVM stimulus
- Scoreboard-based checking
- Functional coverage
- Assertions
- Directed + random test flow

It demonstrates full understanding of APB protocol and industry-level verification skills.