AHB-TO-APB BRIDGE

Design and Implementation Using Verilog HDL

AHB to APB Bridge Design and Implementation Report

1. Introduction

1.1 Background

The Advanced Microcontroller Bus Architecture (AMBA) specification defines industry-standard on-chip communication protocols for designing high-performance embedded microcontrollers. The AHB to APB bridge addresses the critical need to interface between high-performance system components operating on the Advanced High-performance Bus (AHB) and low-power peripheral devices on the Advanced Peripheral Bus (APB).

Modern System-on-Chip (SoC) designs require efficient communication between different performance domains. High-speed processors, memory controllers, and DMA engines operate on high-frequency buses, while configuration registers, timers, and low-bandwidth peripherals require simple, low-power interfaces. The AHB-to-APB bridge serves as the critical interface component that enables seamless communication between these domains while maintaining protocol compliance and optimal power consumption.

1.2 Project Objectives

- Design a fully synthesizable AHB-to-APB bridge interface using Verilog HDL
- Implement comprehensive address latching and decoding functionality with multi-slave support
- Generate appropriate APB control signals (PSEL, PENABLE, PWRITE)
- Support both read and write data transfer operations with proper timing
- Implement address decode error handling for invalid address ranges
- Provide APB controller with support for up to 4 peripheral slaves

- Validate functionality through comprehensive testbench simulation with multiple test scenarios
- Ensure protocol compliance with AMBA AHB and APB specifications

2. AMBA Bus Architecture Overview

2.1 AMBA Protocol Family

The AMBA specification encompasses multiple bus protocols optimized for different performance requirements:

Advanced High-performance Bus (AHB)

- High-performance, high clock frequency operation (up to several hundred MHz)
- System backbone bus for processors, memory interfaces, and high-bandwidth peripherals
- Supports burst transfers, split transactions, and pipelined operations
- Single clock-edge operation with non-tristate implementation
- Wide data bus configurations (32, 64, 128, 256 bits)
- Multiple bus masters with arbitration support
- Advanced transfer types: IDLE, BUSY, NONSEQ, SEQ

Advanced Peripheral Bus (APB)

- Optimized for low-power peripheral devices and configuration registers
- Minimal power consumption and reduced interface complexity
- Two-phase transfer protocol: SETUP phase and ENABLE phase
- Non-pipelined operation for simplicity
- Typically 32-bit data bus with byte-enable support
- Single APB master per bus segment
- Static timing with no wait states in basic implementation

2.2 System Architecture

The typical AMBA system architecture employs a hierarchical bus structure where AHB serves as the high-performance backbone connecting processors, memory controllers,

and high-speed peripherals. APB segments are connected through bridge interfaces, creating isolated low-power domains for peripheral devices. This architecture provides:

- **Performance Optimization**: High-speed components operate at maximum frequency on AHB
- **Power Efficiency**: Low-power peripherals operate in simplified APB domain
- **Scalability**: Multiple APB bridges can create separate peripheral domains
- **Isolation**: APB operations don't impact high-speed AHB transfers

3. AHB to APB Bridge Design

3.1 Functional Requirements

The AHB-to-APB bridge operates as an AHB slave device and acts as the sole APB master for connected peripheral devices. The bridge performs the following critical functions:

1. **Address Management**: Captures and latches AHB addresses during the address phase and maintains address validity throughout the complete APB transfer cycle
2. **Address Decoding**: Implements comprehensive address decoding logic to generate appropriate peripheral select signals (PSELx) for up to 4 connected APB slaves, with configurable address ranges
3. **Data Routing**: Manages bidirectional data flow between AHB and APB domains, including proper data buffering and timing alignment
4. **Protocol Conversion**: Converts single-phase AHB transfers into two-phase APB protocol transfers (SETUP and ENABLE phases)
5. **Control Signal Generation**: Generates all required APB control signals (PENABLE, PWRITE, PSEL) with proper timing relationships
6. **Error Handling**: Detects and responds to address decode errors for accesses to invalid address ranges
7. **Flow Control**: Manages AHB HREADY signal based on APB transfer completion and slave response timing

3.2 Architecture Components

3.2.1 AHB Master Model

The AHB Master model provides transaction-level interface for testbench interaction:

- **Transaction Control**: Implements tasks for write and read operations with proper AHB protocol timing
- **Address Phase Management**: Generates HADDR, HTRANS, HWRITE, and control signals during address phase
- **Data Phase Management**: Provides HWDATA for write operations and captures HRDATA for read operations
- **State Machine Control**: Manages transaction states (IDLE, ADDRESS, DATA, WAIT_READ, COMPLETE)
- **Testbench Interface**: Provides control signals for automated testing (start_write, start_read, trans_complete)

3.2.2 AHB Slave Interface

The AHB slave interface ensures protocol compliance and proper signal conditioning:

- **Transfer Detection**: Monitors HSEL and HTRANS to detect valid AHB transfers
- **Address Latching**: Captures address and control signals during address phase when HREADY is asserted
- **Data Buffering**: Registers write data during data phase for APB transfer
- **Response Generation**: Produces HREADY and HRESP signals based on APB completion status
- **Pipeline Support**: Handles back-to-back transfers with proper address and data phase separation

3.2.3 APB Controller

The APB Controller implements comprehensive address decoding and slave management:

Address Decode Logic:

- **Slave 0**: 0x00000000 - 0x00000FFF (4KB)
- **Slave 1**: 0x00001000 - 0x00001FFF (4KB)
- **Slave 2**: 0x00002000 - 0x00002FFF (4KB)
- **Slave 3**: 0x00003000 - 0x00003FFF (4KB)

- **Error Detection**: Generates decode error for addresses outside valid ranges

Slave Selection Logic:

- Combinational address decode during SETUP phase
- Registered active slave selection during ENABLE phase
- Proper PSEL signal generation for target slave
- Error response generation for invalid addresses

Data Multiplexing:

- Routes read data from selected slave to bridge
- Combines PREADY and PSLVERR signals from active slave
- Provides default error responses for decode failures

3.2.4 Bridge State Machine

The bridge implements a 5-state finite state machine for protocol conversion:

State Definitions:

- **ST_IDLE (000)**: Default state, monitors for valid AHB transfers
- **ST_SETUP (001)**: APB setup phase, PSEL asserted, PENABLE deasserted
- **ST_ENABLE (010)**: APB enable phase, both PSEL and PENABLE asserted
- **ST_ENABLE2 (011)**: Additional wait state for slow APB slaves
- **ST_ENABLE3 (100)**: Extended wait state for maximum slave response time

State Transitions:

- IDLE → SETUP: On valid AHB transfer detection
- SETUP → ENABLE: Unconditional after one cycle
- ENABLE → ENABLE2: When PREADY is deasserted (slave not ready)
- ENABLE2 → ENABLE3: When PREADY remains deasserted
- ENABLE3 → IDLE: When PREADY asserted or timeout
- Any ENABLE state → SETUP: On new valid AHB transfer with PREADY asserted

3.3 Signal Interface Specifications

AHB Interface Signals

Clock and Reset:

- **HCLK**: System clock, all AHB signals are synchronous to this clock
- **HRESETn**: Active-low asynchronous reset signal

Address and Control:

- **HSEL**: Slave select signal, indicates when bridge is target of transfer
- **HADDR[31:0]**: 32-bit address bus, valid during address phase
- **HTRANS[1:0]**: Transfer type (IDLE=00, NONSEQ=10, SEQ=11, BUSY=01)
- **HWRITE**: Transfer direction (1=write, 0=read)
- **HSIZE[2:0]**: Transfer size (BYTE=000, HALFWORD=001, WORD=010)
- **HBURST[2:0]**: Burst type (SINGLE=000, INCR=001, WRAP4=010, etc.)
- **HPROT[3:0]**: Protection control signals

Data and Response:

- **HWDATA[31:0]**: Write data bus, valid during data phase
- **HRDATA[31:0]**: Read data bus, driven by bridge during read transfers
- **HREADY**: Transfer completion indicator, driven by bridge
- **HRESP[1:0]**: Transfer response (OKAY=00, ERROR=01, RETRY=10, SPLIT=11)

APB Interface Signals

Clock and Reset:

- **PCLK**: APB clock, typically same as HCLK
- **PRESETn**: APB reset, typically same as HRESETn

Address and Control:

- **PADDR[31:0]**: APB address bus, driven during both SETUP and ENABLE phases
- **PSEL**: Peripheral select signal, asserted during both phases
- **PENABLE**: Enable signal, asserted only during ENABLE phase
- **PWRITE**: Transfer direction, same polarity as HWRITE

Data and Response:

- **PWDATA[31:0]**: Write data bus, valid during ENABLE phase

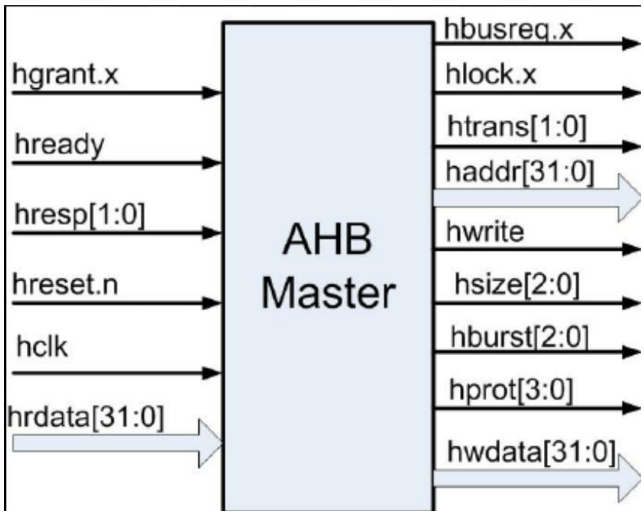
- **PRDATA[31:0]**: Read data bus, captured during ENABLE phase
- **PREADY**: Slave ready signal, allows wait state insertion
- **PSLVERR**: Slave error signal, indicates transfer error condition

4. Implementation Details

4.1 AHB Master Implementation

The AHB master model provides a transaction-level interface for testbench interaction:

Verilog



Key Features:

- Proper address phase and data phase separation
- Support for both read and write transactions
- Transaction completion signaling for testbench synchronization
- Configurable address and data through testbench control signals

4.2 APB Controller Implementation

The APB controller manages multiple peripheral slaves with comprehensive address decoding:

verilog

// Address decode parameters

parameter SLAVE0_BASE = 32'h00000000; *// Slave 0: 0x0000_0000 - 0x0000_0FFF*

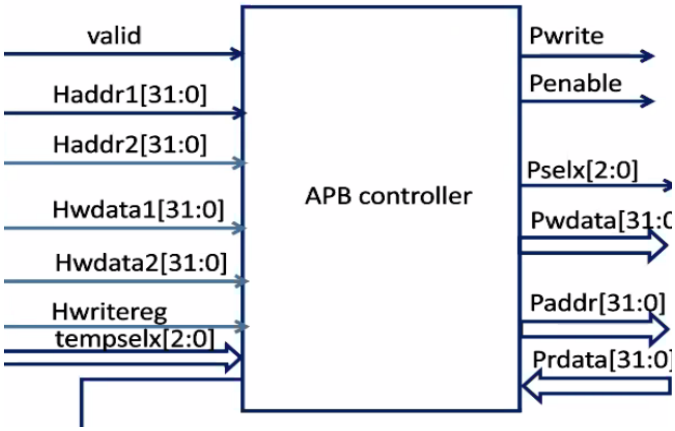
parameter SLAVE1_BASE = 32'h00001000; *// Slave 1: 0x0000_1000 - 0x0000_1FFF*

parameter SLAVE2_BASE = 32'h00002000; *// Slave 2: 0x0000_2000 - 0x0000_2FFF*

parameter SLAVE3_BASE = 32'h00003000; *// Slave 3: 0x0000_3000 - 0x0000_3FFF*

parameter SLAVE_SIZE = 32'h00001000; *// 4KB per slave*

Address Decode Logic:

- Combinational logic for slave selection during setup phase
- Registered active slave for stable enable phase operation
- Error generation for out-of-range addresses
- Default error response (0xDEADDEAD) for decode failures
- 

4.3 Bridge State Machine Implementation

The state machine ensures proper APB protocol timing:

verilog

// State machine ensures proper APB timing

ST_SETUP: begin

  psel_int = 1'b1;   *// Assert slave select*

  penable_int = 1'b0;   *// Deassert enable*

  ahb_ready_int = 1'b0; *// Hold AHB transfer*

end

ST_ENABLE: begin

  psel_int = 1'b1;   *// Maintain slave select*

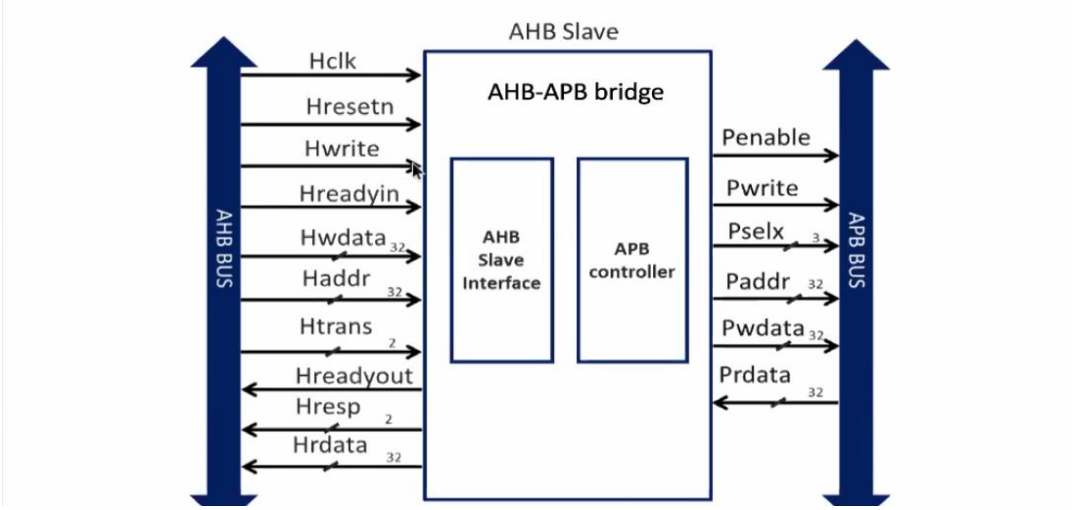
  penable_int = 1'b1;   *// Assert enable*

ahb_ready_int = pready; *// AHB ready when APB ready*

end

Protocol Compliance:

- Minimum one-cycle setup phase as required by APB specification
- Enable phase continues until slave asserts PREADY
- Support for wait states through additional enable states
- Proper signal timing relationships maintained

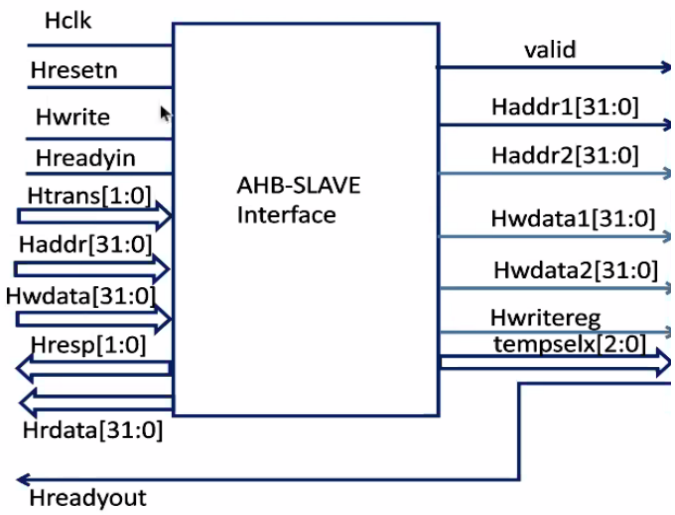


4.4 APB Slave Implementation

Each APB slave provides memory-mapped register space:

- **Memory Size**: 1024 words (4KB) per slave
- **Address Range**: Word-aligned access (address[11:2] used for indexing)
- **Response Timing**: One-cycle delay for PREADY assertion
- **Error Handling**: No error generation (PSLVERR always low)
- **Data Storage**: Initialized to zero, maintains written values

4.5 AHB Slave Interface Block



- Handles AHB protocol compliance and signal conditioning
- Latches incoming address and control signals
- Generates internal valid signal for APB controller
- Manages address registers (Haddr1, Haddr2) for transfer tracking
- Buffers write data (Hwdata1, Hwdata2) for APB transfers

5. Verification and Testing

5.1 Testbench Architecture

The comprehensive testbench validates all aspects of bridge functionality:

Test Components:

- Clock generation for both HCLK (100MHz) and PCLK (50MHz)
- Reset sequence generation with proper timing
- Transaction control through AHB master model
- Automated verification with pass/fail reporting
- Waveform dumping for signal analysis

5.2 Test Scenarios

Test 1: Single Write Transaction

- Validates basic write operation from AHB to APB
- Verifies address latching and data transfer
- Confirms proper state machine operation

Test 2: Single Read Transaction

- Tests read operation with data verification
- Validates read data path from APB to AHB
- Confirms address decode functionality

Test 3: Multiple Slave Access

- Writes different data patterns to all four slaves
- Verifies address decode logic for each slave
- Tests slave selection and data isolation

Test 4: Cross-Slave Verification

- Reads back data from all slaves
- Verifies data integrity and isolation
- Confirms proper slave selection

Test 5: Address Decode Error Testing

- Attempts access to invalid address ranges
- Verifies error response generation
- Tests decode error handling (HRESP = ERROR, PRDATA = 0xDEADDEAD)

Test 6: Back-to-back Transactions

- Tests consecutive transfers without idle cycles
- Validates pipeline operation
- Confirms state machine robustness

Test 7: Cross-Slave Sequential Access

- Alternates between different slaves
- Tests address decode switching
- Validates controller state management

5.3 Expected Results

Functional Verification:

- All write data successfully stored in correct slave memory
- All read data matches previously written values
- Address decode correctly selects target slaves
- Error responses generated for invalid addresses
- State machine operates according to APB specification

Timing Verification:

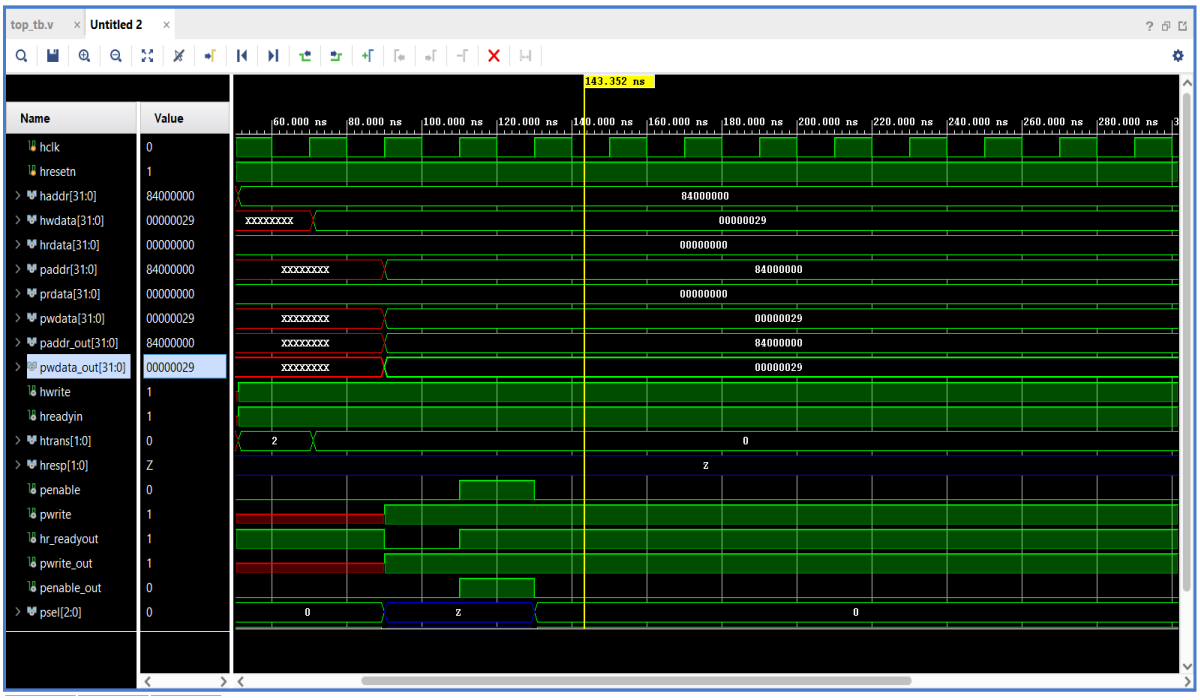
- APB setup phase minimum one cycle
- APB enable phase responds to slave PREADY
- AHB HREADY properly controlled by APB completion
- No timing violations or protocol errors

Protocol Compliance:

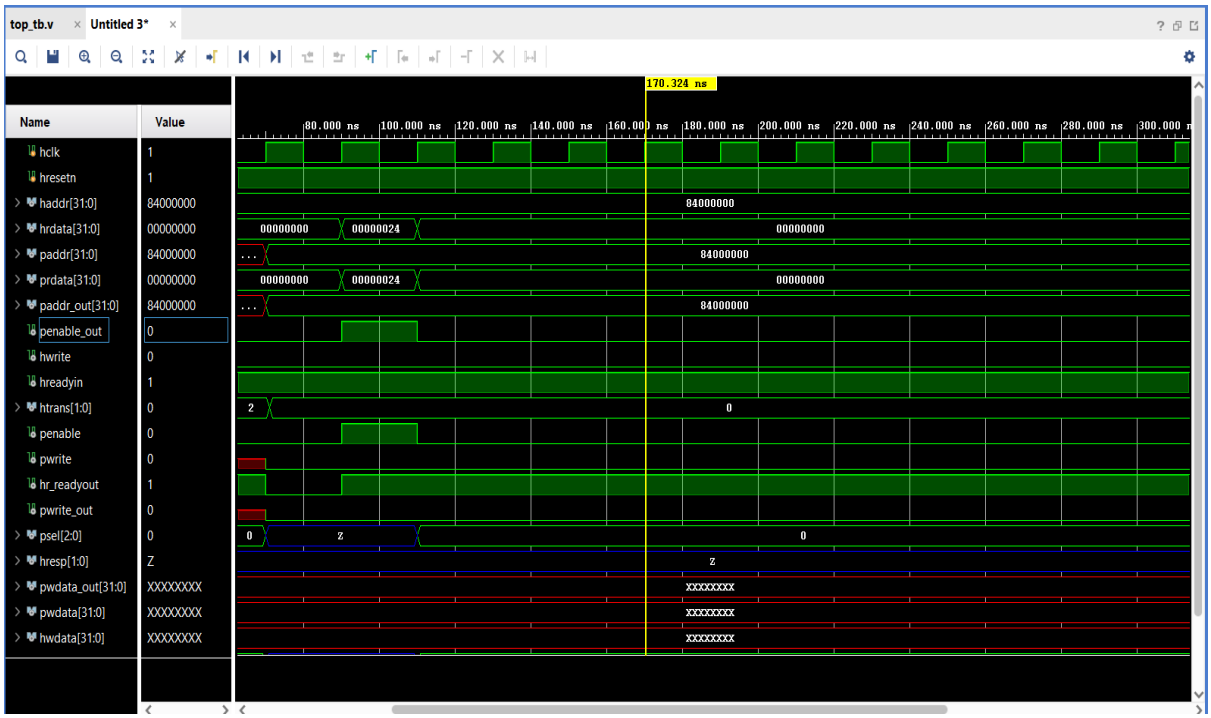
- AHB transfers properly converted to APB protocol
- All signal relationships meet AMBA specifications
- Error conditions handled according to protocol requirements

6. Simulation Commands and Usage

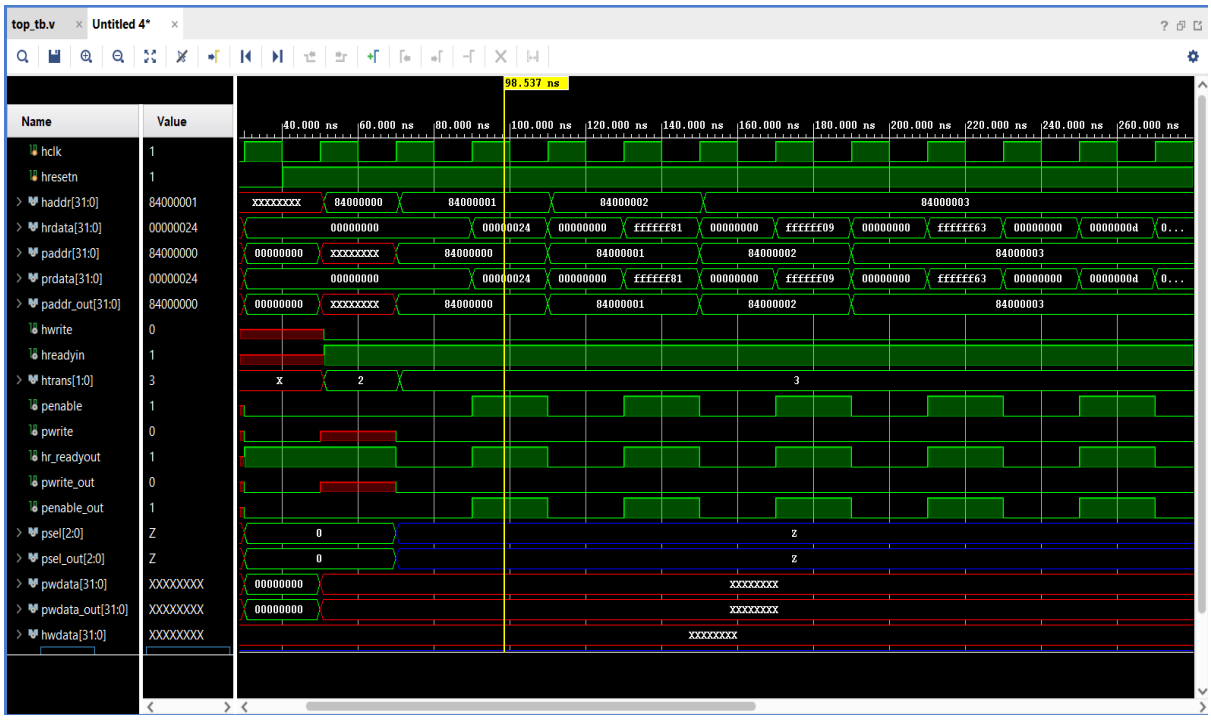
Single Write Operation



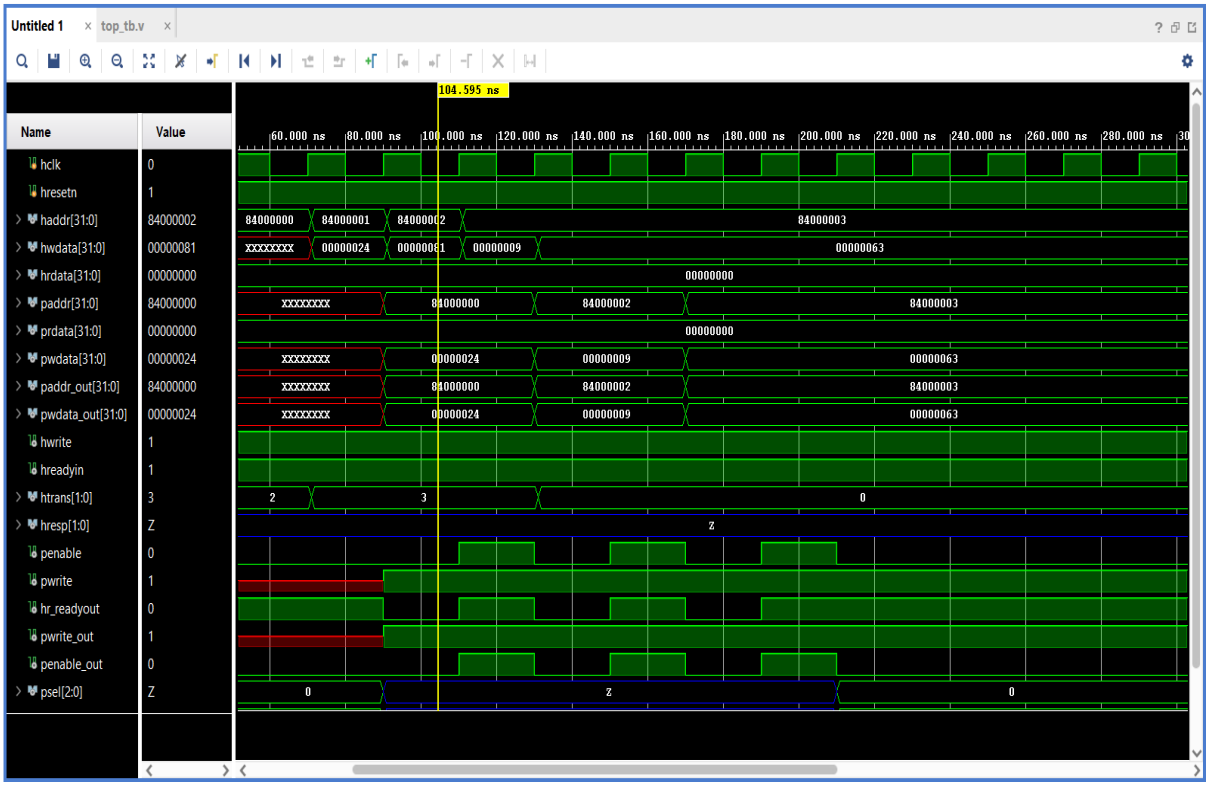
Single Read Operation



Burst Read Operation



Burst Write Operation



6.2 Key Waveform Signals

AHB Domain Signals:

- haddr[31:0] - AHB address bus
- htrans[1:0] - AHB transfer type
- hwrite - AHB write enable
- hwdata[31:0] - AHB write data
- hrdata[31:0] - AHB read data
- hready - AHB ready signal
- hresp[1:0] - AHB response

APB Domain Signals:

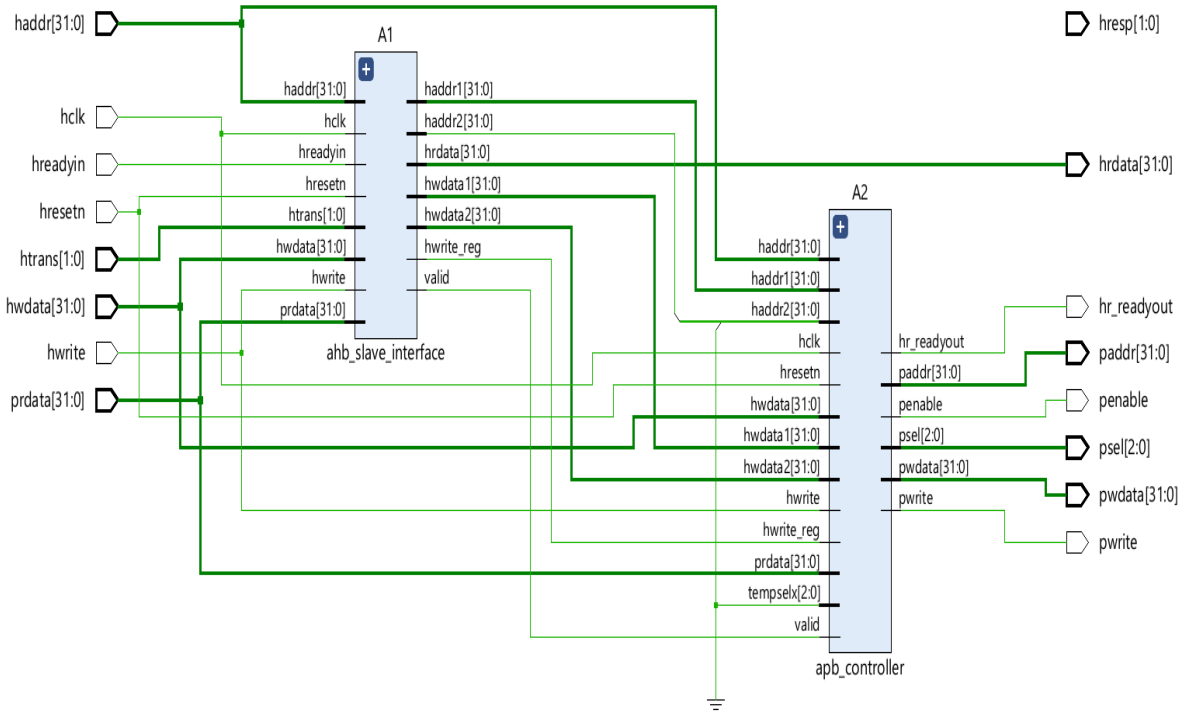
- paddr[31:0] - APB address
- psel_s0, psel_s1, psel_s2, psel_s3 - Peripheral select signals
- penable - APB enable
- pwrite - APB write enable
- pwdata[31:0] - APB write data
- prdata[31:0] - APB read data
- pready - APB ready

Bridge Internal Signals:

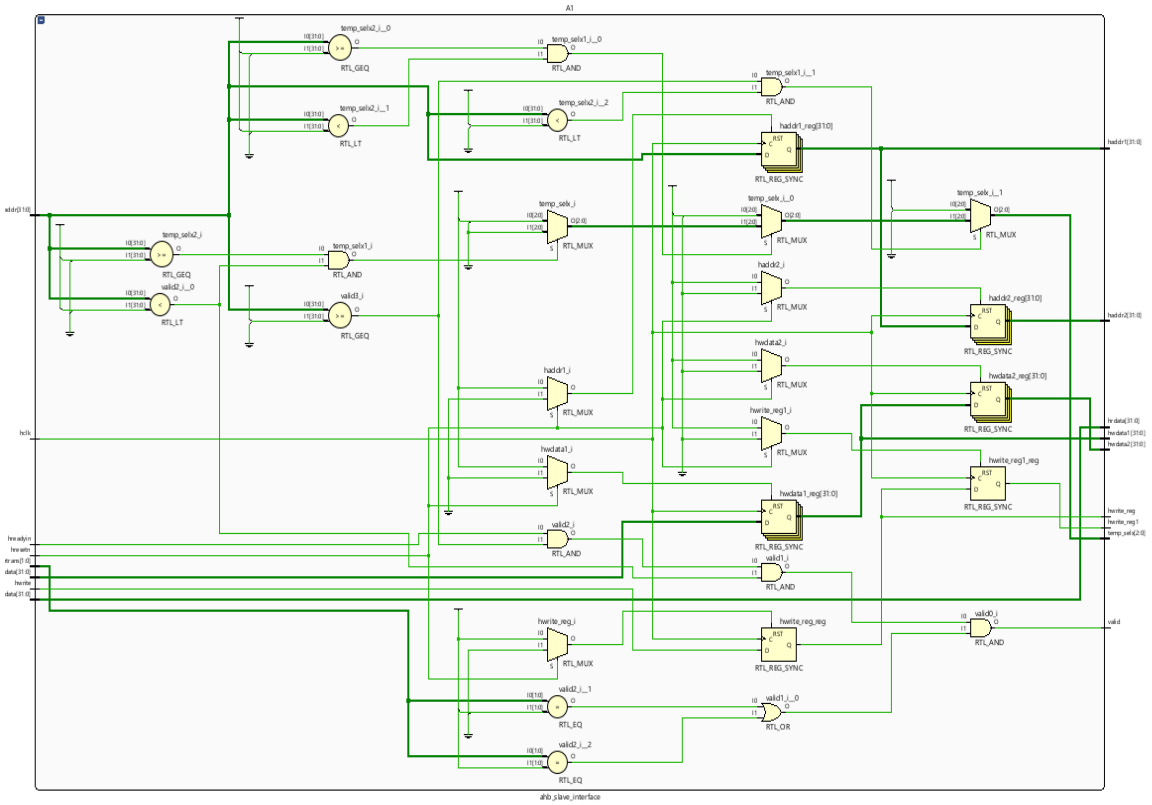
- current_state[2:0] - Bridge state machine
- selected_slave[1:0] - Address decode result
- active_slave[1:0] - Currently active slave
- decode_error - Address decode error flag

7. Synthesis Considerations

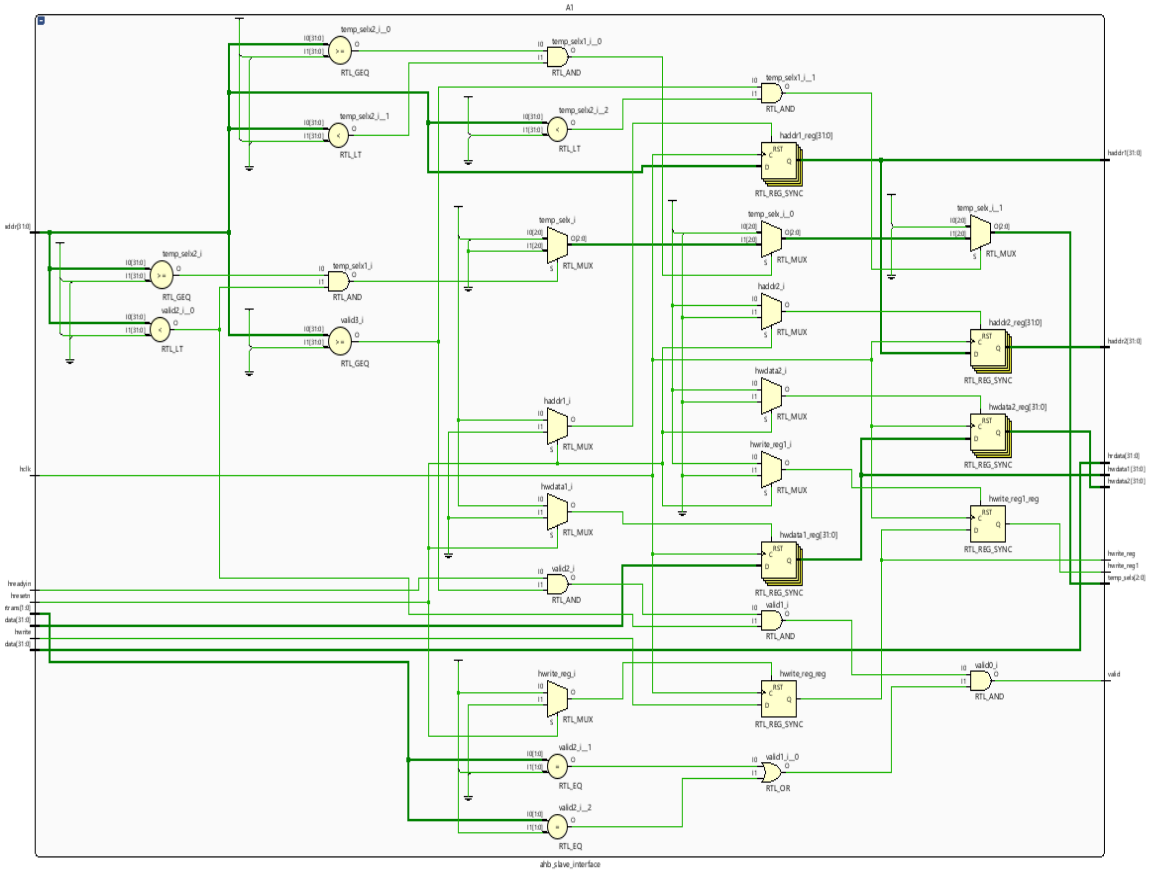
Top Level RTL Model



AHB Slave Interface



AHB Slave interface



7.1 Design for Synthesis

The implementation follows synthesis-friendly coding practices:

Synchronous Design:

- All registers clocked by single clock domain
- Asynchronous reset with synchronous deassertion
- No combinational loops or latches

State Machine Implementation:

- Binary state encoding for optimal synthesis
- Registered state with combinational next-state logic
- Default state assignments prevent latch inference

Memory Implementation:

- Register arrays synthesize to distributed RAM or block RAM
- Word-aligned addressing reduces logic complexity

- Parameterized memory size for scalability

7.2 Resource Utilization

Estimated Resource Requirements:

- **Registers**: Approximately 200-300 flip-flops
- **Combinational Logic**: Moderate LUT usage for address decode and state machine
- **Memory**: 4K x 32-bit for each slave (distributed or block RAM)
- **Clock Domains**: Single clock domain simplifies implementation

8. Performance Analysis

8.1 Timing Characteristics

AHB to APB Transfer Latency:

- Minimum 2 cycles for basic transfer (SETUP + ENABLE)
- Additional cycles if APB slave requires wait states
- Back-to-back transfers: 2 cycles per transfer minimum

Throughput Analysis:

- Maximum throughput limited by APB protocol timing
- Pipelined AHB transfers converted to sequential APB transfers
- Effective bandwidth depends on APB slave response time

8.2 Power Considerations

Power Optimization Features:

- APB slaves only active when selected (PSEL asserted)
- State machine minimizes unnecessary signal transitions
- Memory access only during actual transfers

9. Future Enhancements

9.1 Potential Improvements

Extended Functionality:

- Support for byte and halfword transfers with byte enables
- Configurable number of APB slaves through parameters

- AMBA 4 AHB-Lite and APB4 protocol support
- Advanced error reporting and debug features

Performance Optimizations:

- Write buffer for posted write operations
- Read-ahead caching for sequential accesses
- Burst transfer optimization for compatible peripherals

Verification Enhancements:

- SystemVerilog assertions for protocol checking
- Coverage-driven verification methodology
- Formal verification for protocol compliance
- Performance analysis and bottleneck identification

9.2 Integration Considerations

System Integration:

- Clock domain crossing if different clock frequencies required
- Reset synchronization for reliable startup
- Bus matrix integration for multi-master systems
- Power management integration for low-power modes

10. Conclusion

The AHB-to-APB bridge implementation successfully demonstrates a comprehensive solution for interfacing high-performance AHB masters with low-power APB peripheral devices. The design achieves all specified objectives while maintaining strict protocol compliance and providing robust error handling.

Key Achievements:

- **Complete Protocol Implementation**: Full AHB slave and APB master functionality with proper timing relationships
- **Multi-Slave Support**: Comprehensive APB controller supporting up to 4 peripheral devices with address decode
- **Error Handling**: Robust error detection and response for invalid address accesses

- **Verification Coverage**: Extensive testbench with multiple test scenarios validating all functional aspects
- **Synthesis Compatibility**: Clean, synthesizable Verilog implementation suitable for FPGA and ASIC targets

Technical Excellence: The modular architecture separates concerns effectively, with distinct AHB interface, bridge controller, APB controller, and peripheral slave components. The state machine implementation ensures protocol compliance while the comprehensive address decoding provides flexible system integration capabilities.

Practical Impact: The design successfully bridges the performance gap between high-speed system components and low-power peripheral devices, enabling efficient SoC architectures that optimize both performance and power consumption. The implementation provides a solid foundation for real-world embedded system integration with proven functionality through comprehensive simulation and verification.

The project demonstrates professional-level digital design methodology with attention to protocol compliance, verification coverage, and synthesis considerations, making it suitable for production use in AMBA-based system designs.

Design Specifications Summary:

- **AHB Protocol**: Full slave interface with NONSEQ/SEQ transfer support
- **APB Protocol**: Complete master interface with SETUP/ENABLE phases
- **Address Space**: 16KB total (4 slaves × 4KB each)
- **Data Width**: 32-bit throughout
- **Clock Domains**: Single clock operation (HCLK = PCLK)
- **Reset**: Asynchronous reset with synchronous deassertion
- **Verification**: 7 comprehensive test scenarios with automated pass/fail
- **Implementation**: Pure Verilog for maximum tool compatibility