AHB2APB BRIDGE DESIGN

By-Diksha Jain

WHAT IS AHB2APB ?

1. AHB (Advanced High-performance Bus) to APB (Advanced Peripheral Bus) bridge is a hardware component that facilitates communication between high-performance and peripheral buses in a system-on-chip (SoC) design, enabling efficient data transfer between different subsystems.

2. AHB2APB is a bridge protocol in the ARM Advanced Microcontroller Bus Architecture (AMBA) developed by companies like QUALCOMM.

AMBA BASED MICROCONTROLLER ?

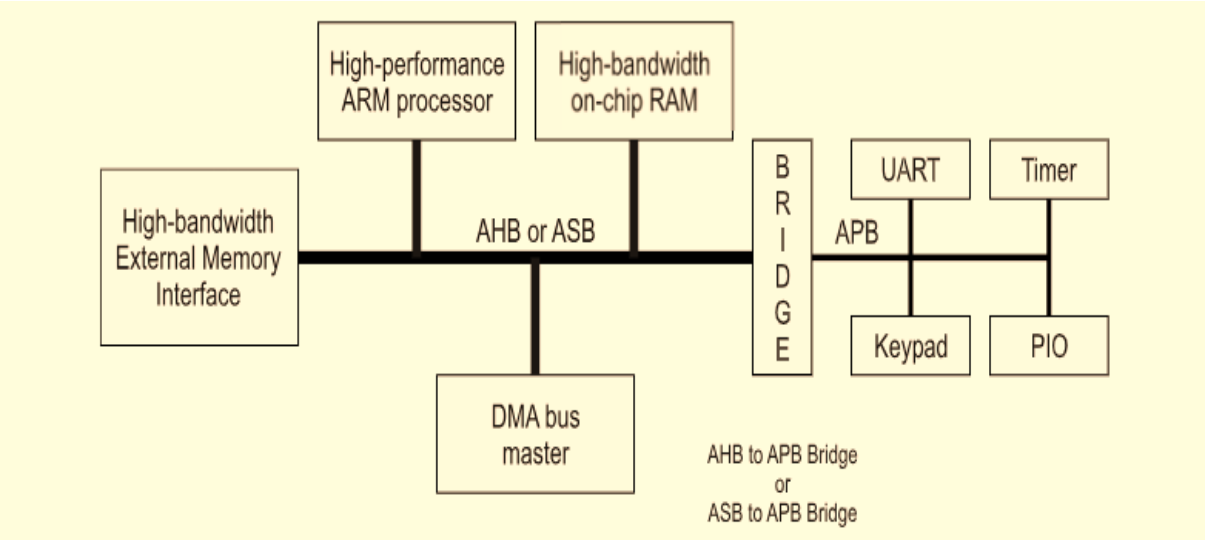

Fig: Design of AMBA based AHB2APB bridge

1. A bridge communicates between :
   High frequency (AHB) → Low frequency (APB)

2. High-Frequency Components: Processor cores (CPU/GPU), high-speed memory interfaces

3. Low-Frequency Components: Peripheral interfaces (APB), communication interfaces (UART, SPI, I2C), wireless modules (Bluetooth, Wi-Fi)

FIRST PROCESSES ARCHITECTURE ?

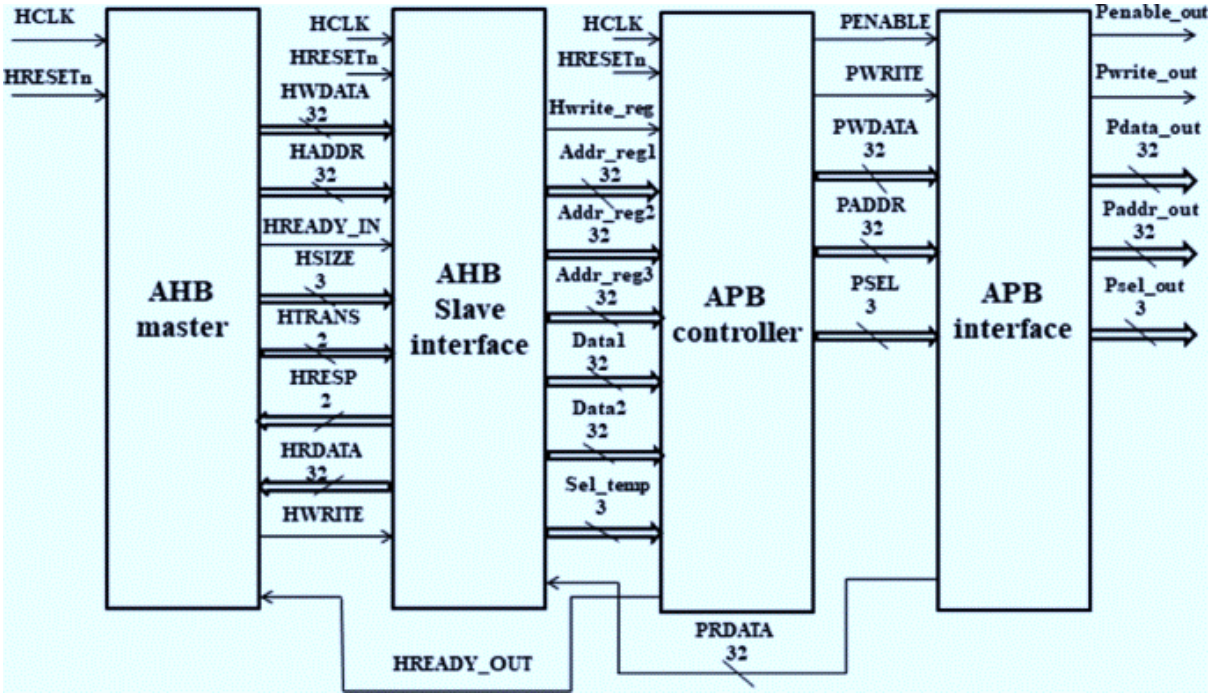

Fig 3. implementation of high performance bus

SOFTWARE USED ?

1. ModelSim is a hardware description language (HDL) simulation and debugging tool. It is widely used for simulating and verifying designs described in languages like VHDL and Verilog.

2. Quartus Prime is an integrated development environment (IDE) provided by Intel for designing, simulating, and programming FPGAs (Field-Programmable Gate Arrays).

STEPS ?

1. Verilog code and waveform generation on model sim for :
    a) AHB_slave_interface.v , AHB_slave_interface_tb.v
    b) APB_controller.v , APB_controller_tb.v
    c) Interface_AHB
    d) Interface_APB
    e) Top_Bridge.v , Top_Bridge_tb.v (combing the 2 sub-blocks i.e AHB_slave_interface and APB_controller)

2. Designing on quarts prime

NOTE: While saving the Verilog code save in working directory of Verilog code and testbench carefully

CONCEPT BEHIND LOGIC USED

A Finite State Machine (FSM) diagram is a visual representation of the different states and transitions that a system undergoes. It illustrates the control logic that manages the communication between the AHB and APB.

1. **ST_IDLE:** Initial state, awaiting a request from either AHB or APB for data transfer.
2. **ST_WWAIT:** Waiting state during a write operation, allowing time for necessary operations to complete.
3. **ST_WRITE:** State for initiating and managing the write operation from AHB to APB.
4. **ST_WRITEP:** State for processing and completing a write operation, acknowledging data reception from AHB.
5. **ST_WENABLE:** State for enabling the write operation, preparing for data transfer from AHB to APB.
6. **ST_WENABLEP:** State for processing and enabling a write operation, confirming the availability of data for transfer.
7. **ST_READ:** State for initiating and managing the read operation from APB to AHB.
8. **ST_REANBLE:** State for processing and enabling a read operation, confirming data availability for transfer to AHB.

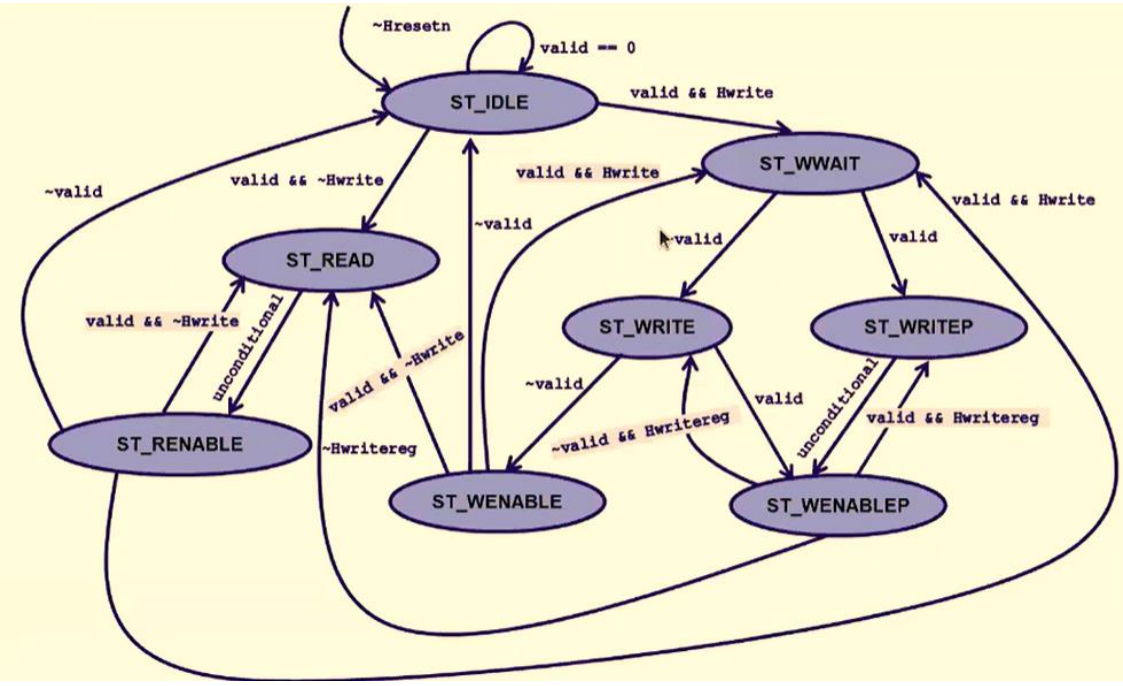

Fig 3. APB controller

Protocol

The AHB to APB (Advanced High-performance Bus to Advanced Peripheral Bus) bridge protocol typically involves:

1. AHB master initiates a transaction.
2. AHB to APB bridge converts AHB signals to APB signals.
3. APB slave responds to the transaction.
4. Bridge translates AHB address and data to corresponding APB signals.
5. Bridge manages bus protocols, timing, and potential signal width differences.

Block Diagram

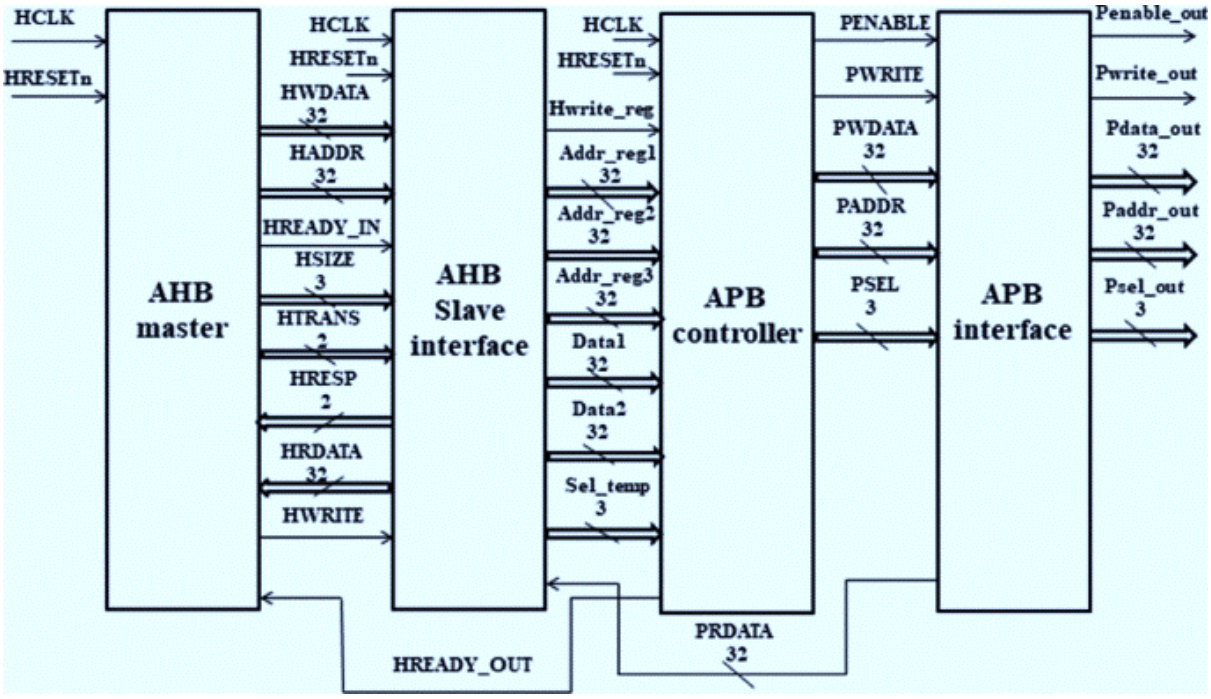

Fig4: block diagram of AHB2APB

Synthesis

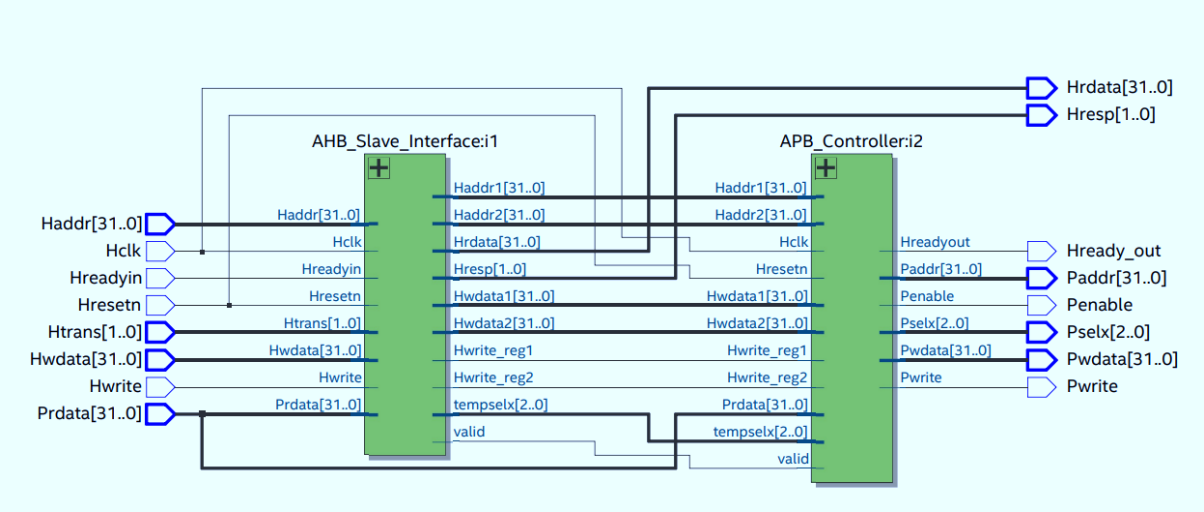

Fig5: Bridge Top Design / Top module block diagram

Output waveforms

Fig 6(a): Single read

Fig 6(b): Single write

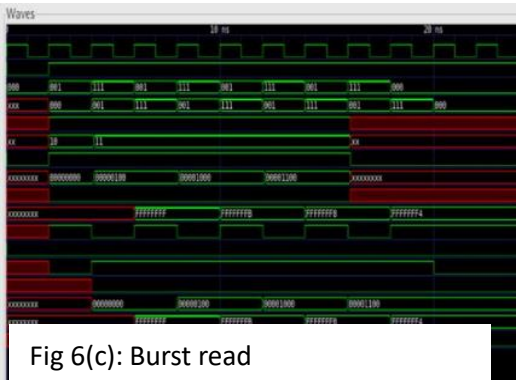

Fig 6(c): Burst read

Fig 6(d): Burst write