SoC Design and Practice

Jungrae Kim (dale40@skku.edu)
Semiconductor Systems Engineering

Project1: AXI to APB Bridge

Goal

- Implement a AXI-to-APB Bridge.

Login and git clone

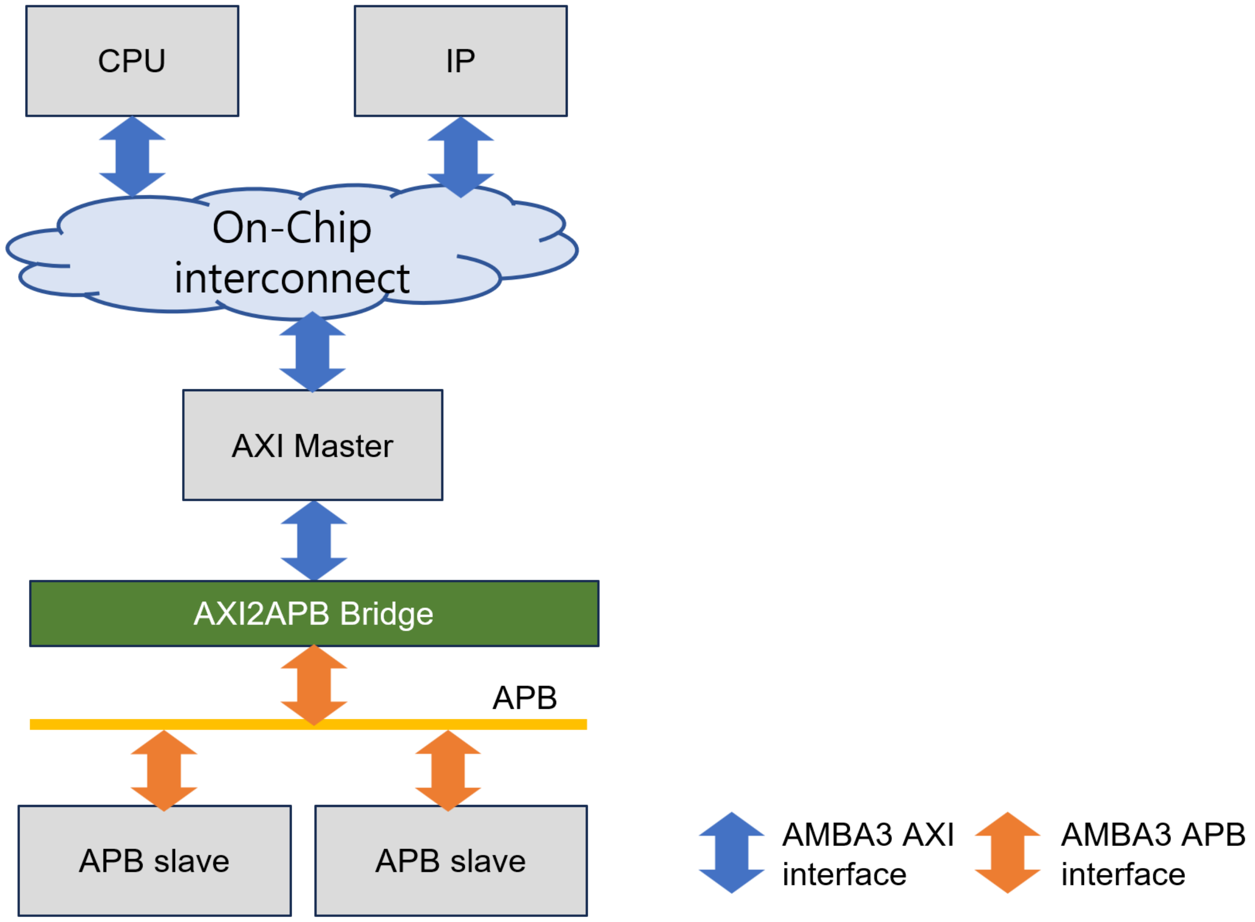
- `$ git clone https://github.com/2025-Spring-SoC/Project1_AXI2APB_Bridge.git`

Application – AXI2APB Bridge

- Used to connect
  AXI-based systems to APB peripherals

- Converts AXI protocol to APB

- In our project,
  Implement AXI-to-APB Bridge

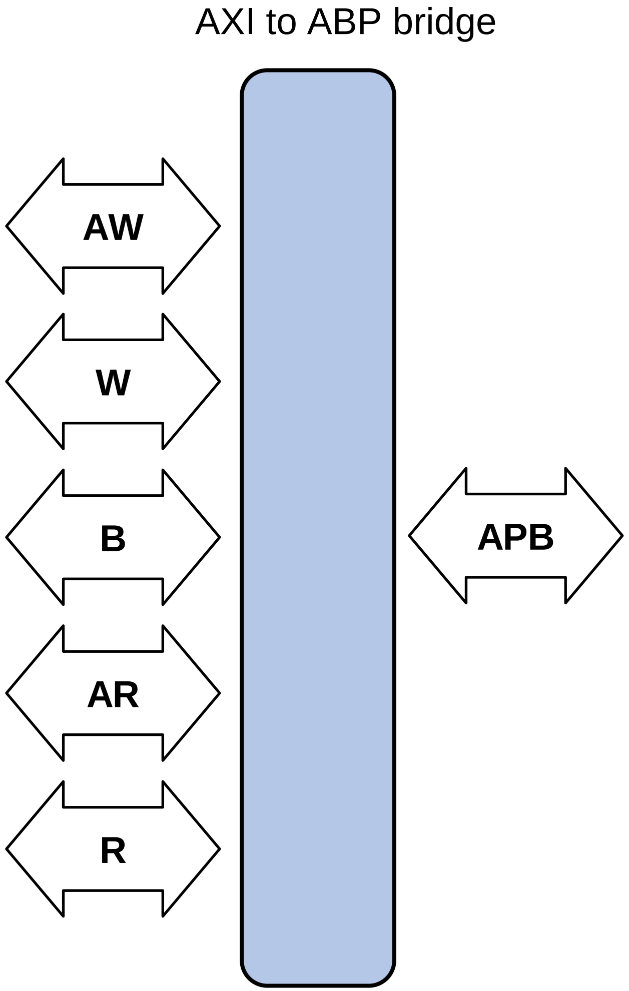


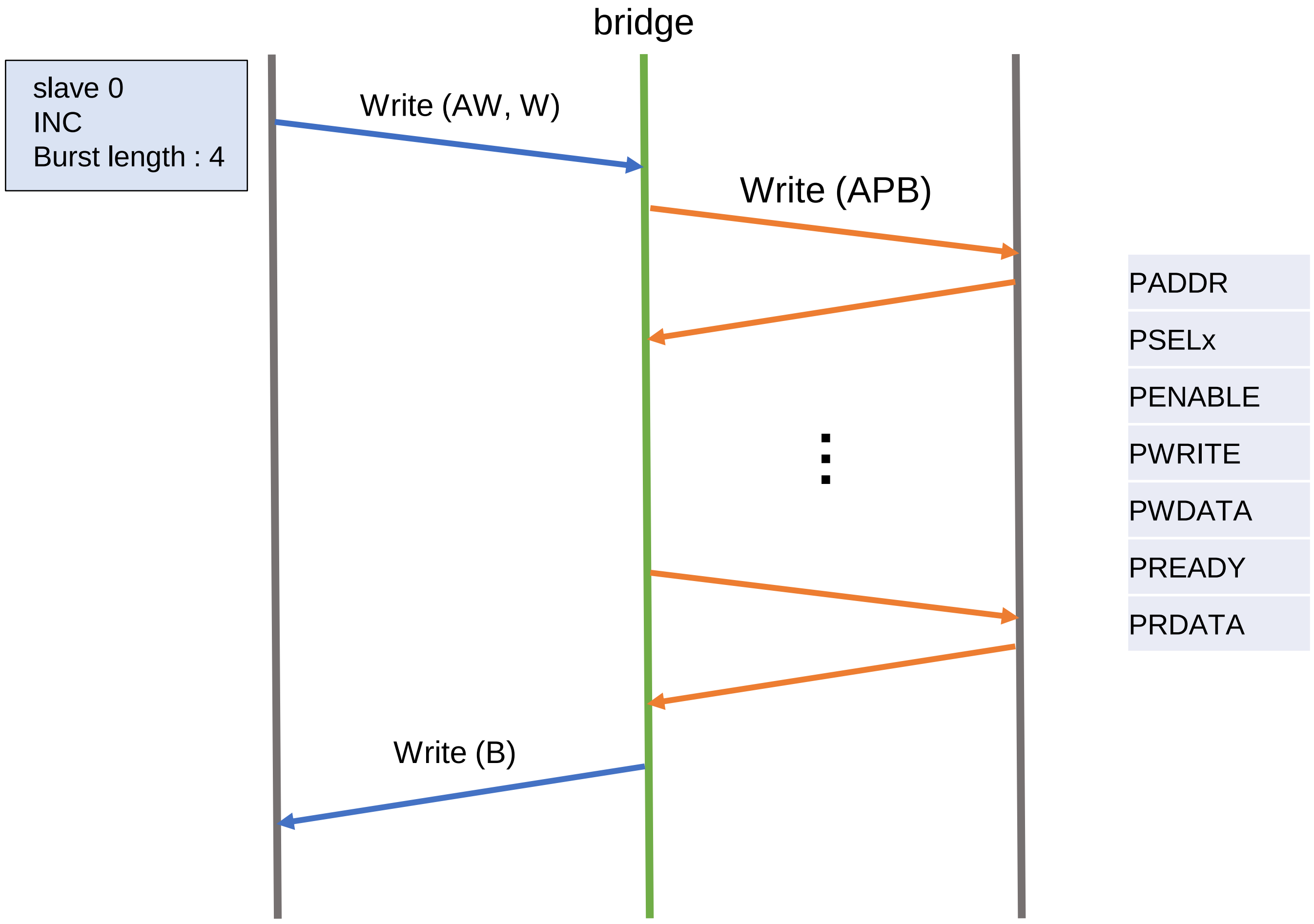
AXI vs APB

- The AXI2APB bridge handles protocol conversion, burst breaking, and timing alignment between AXI and APB.

	AXI	**APB**
Purpose	High-speed memory/processor interface	Low-speed peripheral connection (e.g., UART)
Speed	**High** (optimized for performance)	**Low** (optimized for simplicity)
Burst Transfer	**Supported (burst, out-of-order, pipelined)**	**Not supported (single transfer only)**
Handshake Mechanism	Valid/Ready handshake, pipelined	PREADY-based simple response
Channel Architecture	**Separate read/write address and data channels**	**Single request-response channel**
Implementation Effort	High (requires complex controller)	Low (simple peripheral logic)

Block Diagram



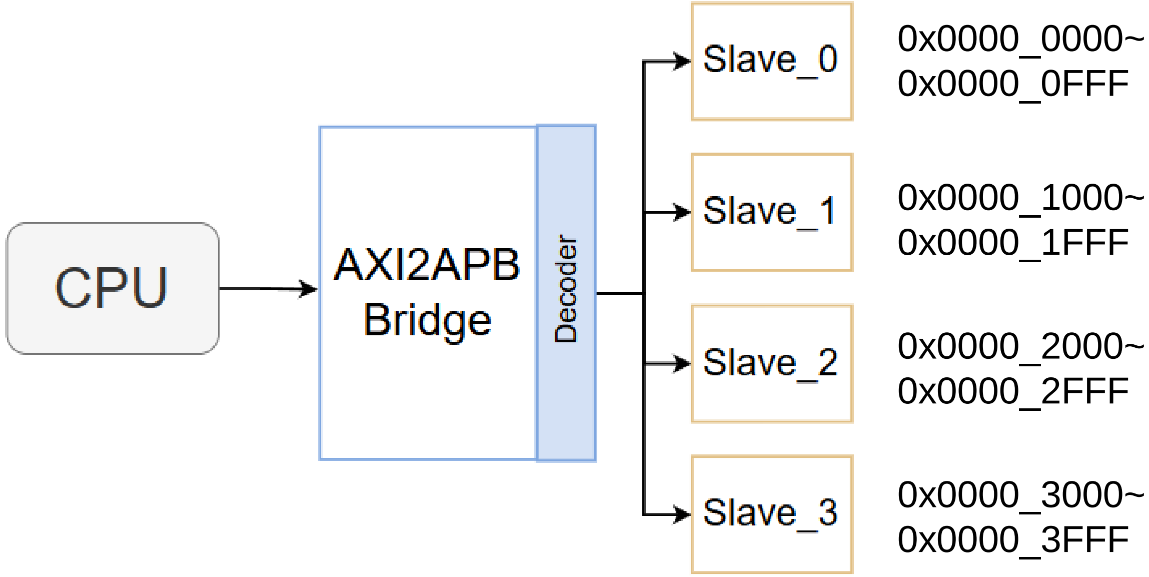


Address Decoding

- Address decoding to support multi slave (Timer, UART, GPIO…)
    - Each slave is assigned a 4KB address space

- If, AXADDR = 0x1030 → Slave_1

  PSEL[3:0] = 'b0010



Our project

- Num of slaves : 2

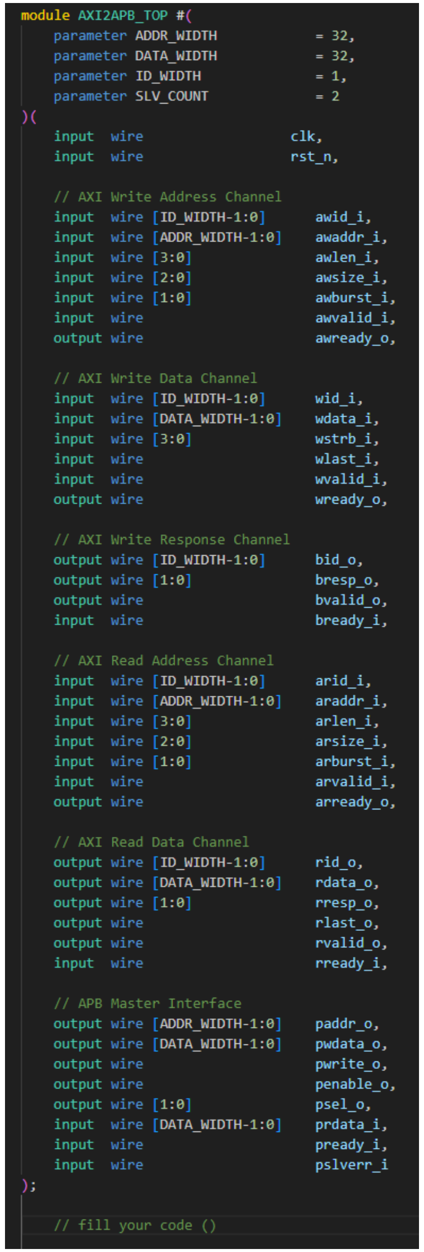
- Data width : Fixed to 4B

- Address ranges:
  - slave_0 : 0x0001_F000 ~ 0x0001_FFFF
  - slave_1 : 0x0002_F000 ~ 0x0002_FFFF

  *Accesses to other addresses are blocked for testing

- Burst type: FIXED, INC
- Burest length: 1~16 transfers

- Goal: burst read/write operations via AXI2APB bridge



```
module AXI2APB_TOP #(
    parameter ADDR_WIDTH            = 32,
    parameter DATA_WIDTH            = 32,
    parameter ID_WIDTH              = 1,
    parameter SLV_COUNT             = 2
)(
    input  wire                   clk,
    input  wire                   rst_n,

    // AXI Write Address Channel
    input  wire [ID_WIDTH-1:0]      awid_i,
    input  wire [ADDR_WIDTH-1:0]    awaddr_i,
    input  wire [3:0]               awlen_i,
    input  wire [2:0]               awsize_i,
    input  wire [1:0]               awburst_i,
    input  wire                     awvalid_i,
    output wire                     awready_o,

    // AXI Write Data Channel
    input  wire [ID_WIDTH-1:0]      wid_i,
    input  wire [DATA_WIDTH-1:0]    wdata_i,
    input  wire [3:0]               wstrb_i,
    input  wire                     wlast_i,
    input  wire                     wvalid_i,
    output wire                     wready_o,

    // AXI Write Response Channel
    output wire [ID_WIDTH-1:0]      bid_o,
    output wire [1:0]               bresp_o,
    output wire                     bvalid_o,
    input  wire                     bready_i,

    // AXI Read Address Channel
    input  wire [ID_WIDTH-1:0]      arid_i,
    input  wire [ADDR_WIDTH-1:0]    araddr_i,
    input  wire [3:0]               arlen_i,
    input  wire [2:0]               arsize_i,
    input  wire [1:0]               arburst_i,
    input  wire                     arvalid_i,
    output wire                     arready_o,

    // AXI Read Data Channel
    output wire [ID_WIDTH-1:0]      rid_o,
    output wire [DATA_WIDTH-1:0]    rdata_o,
    output wire [1:0]               rresp_o,
    output wire                     rlast_o,
    output wire                     rvalid_o,
    input  wire                     rready_i,

    // APB Master Interface
    output wire [ADDR_WIDTH-1:0]    paddr_o,
    output wire [DATA_WIDTH-1:0]    pwdata_o,
    output wire                     pwrite_o,
    output wire                     penable_o,
    output wire [1:0]               psel_o,
    input  wire [DATA_WIDTH-1:0]    prdata_i,
    input  wire                     pready_i,
    input  wire                     pslverr_i
);

    // fill your code ()
```

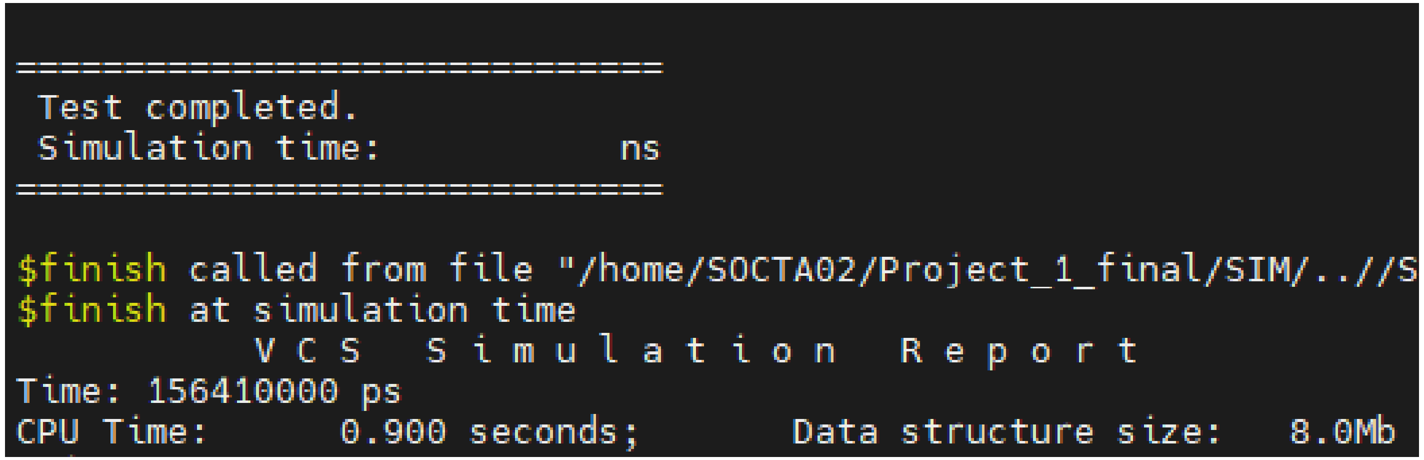
Notation

- 1) Module-level modifications are free.
  - Do not modify the top module's interface.

- 2) Any implementation that contradicts the intent will receive a score of 0.
  - Simply storing the write data, and return it without performing a proper operation.

Score

- Evaluation
  - Simulation Pass (75%)



```
==============================
 Test completed.
 Simulation time:           ns
==============================

$finish called from file "/home/SOCTA02/Project_1_final/SIM/..//S
$finish at simulation time
           V C S   S i m u l a t i o n   R e p o r t
Time: 156410000 ps
CPU Time:      0.900 seconds;       Data structure size:   8.0Mb
```

  - Performance (25%)
    - Based on Simulation time

Submission

- **Upload a zip file (Project1_Name_Student-ID.zip) on icampus**
  - The submission should include **all** of the files in AXI2APB_Bridge/RTL (include filelist)

- **Due: Sun. 4/27, 23:59**

- **Any Copied work will get 0**