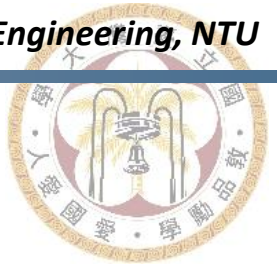
Advanced Computer-Aided VLSI System Design

Homework 2: Local Binary Pattern

Graduate Institute of Electronics Engineering, National Taiwan University

NTU GIEE

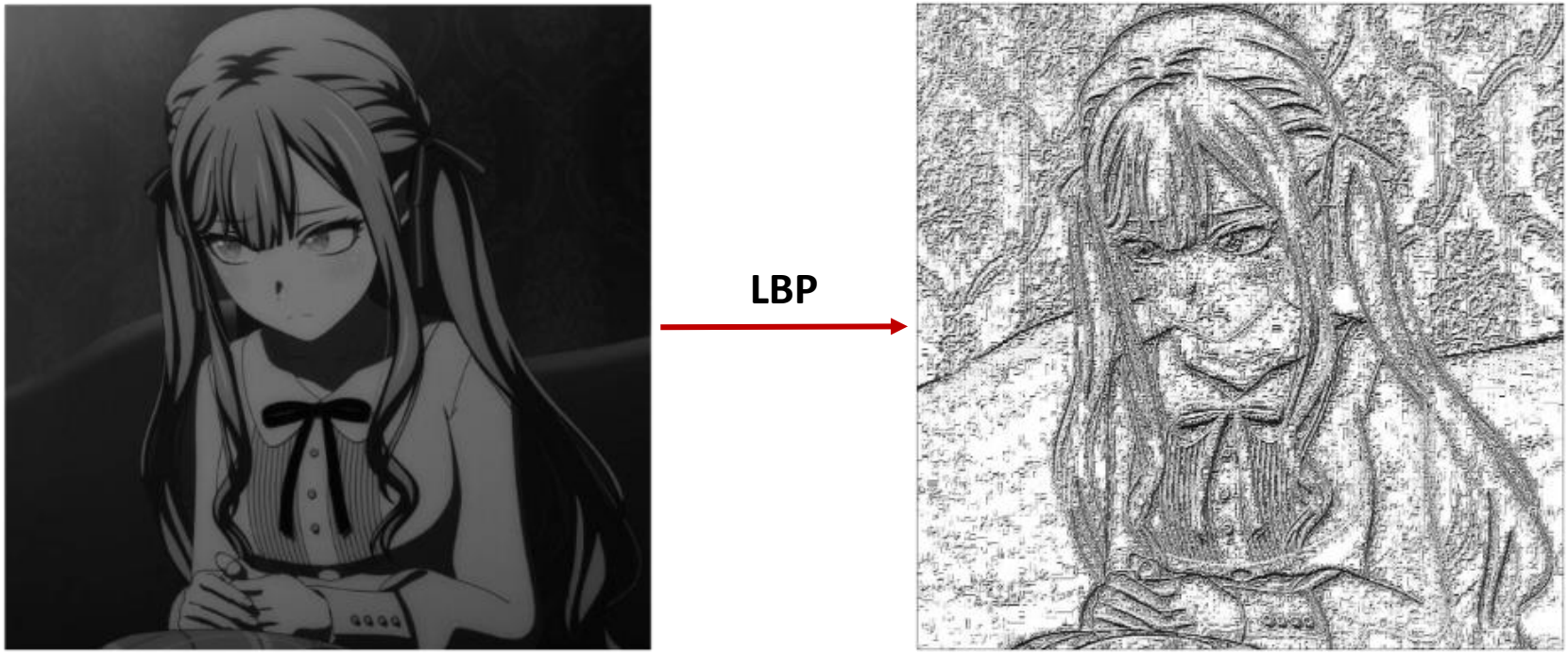


Goals

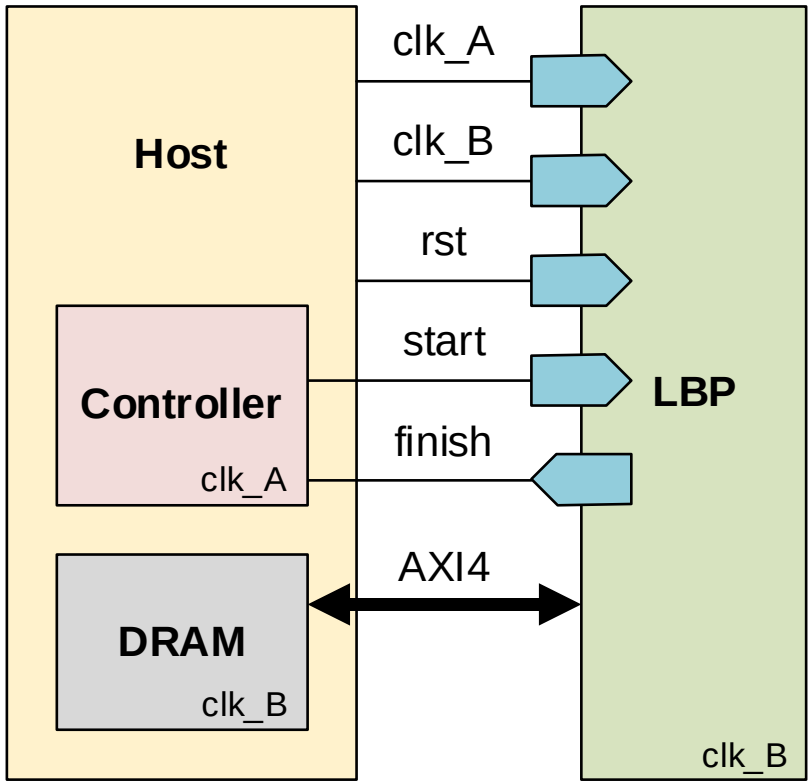
- In this homework, you will learn
  - Clock Domain Crossing technique
  - Communication between modules with AXI bus

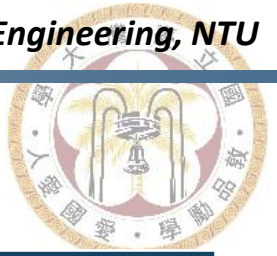
Introduction

- You are asked to design a Local Binary Patterns (LBP), which can be used to describe local texture features.



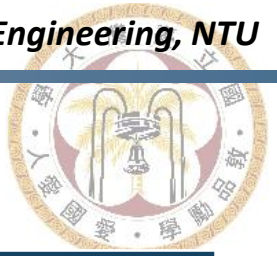
Block Diagram





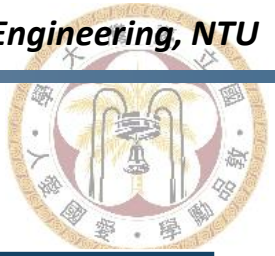
Input/Output

Signal Name	I/O	Width	Simple Description
clk_A	I	1	Clock for control signal (positive edge trigger). Inputs delays **positive** edge clock by **1ns**. Outputs should be synchronized at clock **rising** edge.
clk_B	I	1	Clock for design and axi channel (positive edge trigger). Inputs are synchronized with the **positive** edge clock. Outputs should be synchronized at clock **rising** edge.
rst	I	1	Active **high** asynchronous reset.
start	I	1	Input start signal.
finish	O	1	Output finish signal. The signal should **be asserted high for three cycles** to indicate the end of computation.



Input/Output

Signal Name	I/O	Width	Simple Description
data_awaddr data_araddr	O	15	AXI Write / Read request channel. Address of first transfer in a transaction.
data_awlen data_arlen	O	8	AXI Write / Read request channel. Total number of transfers in a transaction.
data_awsize data_arsize	O	3	AXI Write / Read request channel. Maximum number of bytes in each data transfer within a transaction.
data_awburst data_arburst	O	2	AXI Write / Read request channel. How the address increments between transfers in a transaction.
data_awvalid data_arvalid	O	1	AXI Write / Read request channel. Write / Read request valid indicator.
data_awready data_arready	I	1	AXI Write / Read request channel. Write / Read request ready indicator.

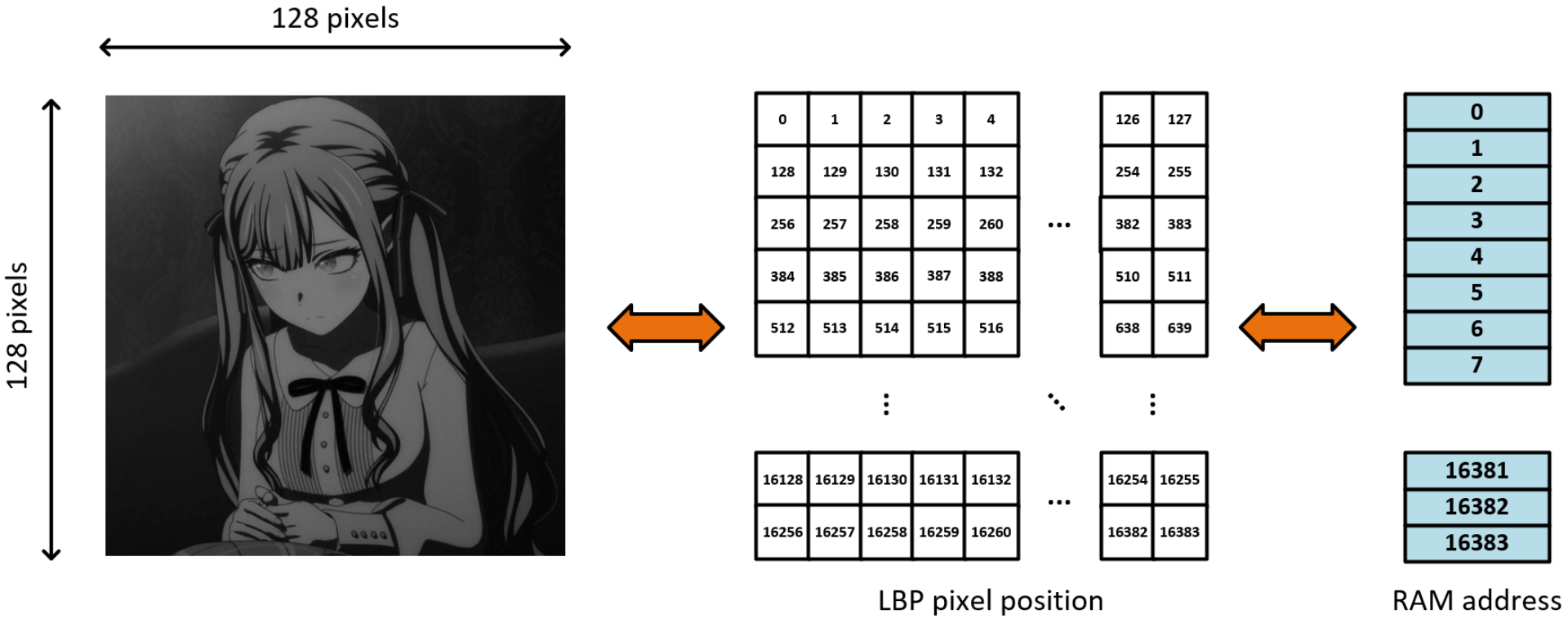


Input/Output

Signal Name	I/O	Width	Simple Description
data_wdata data_rdata	O I	8	AXI Write / Read data channel. The actual data signal.
data_wlast data_rlast	O I	1	AXI Write / Read data channel. Indicates the last write data transfer of a transaction.
data_wvalid data_rvalid	O I	1	AXI Write / Read data channel. Write / Read data valid indicator.
data_wready data_rready	I O	1	AXI Write / Read data channel. Write / Read data ready indicator.
data_wstrb	O	1	AXI Write data channel. Indicates which byte lanes of write data contain valid data in a write transaction.
data_rresp	I	2	AXI Read data channel. Response for transactions on the read channels.

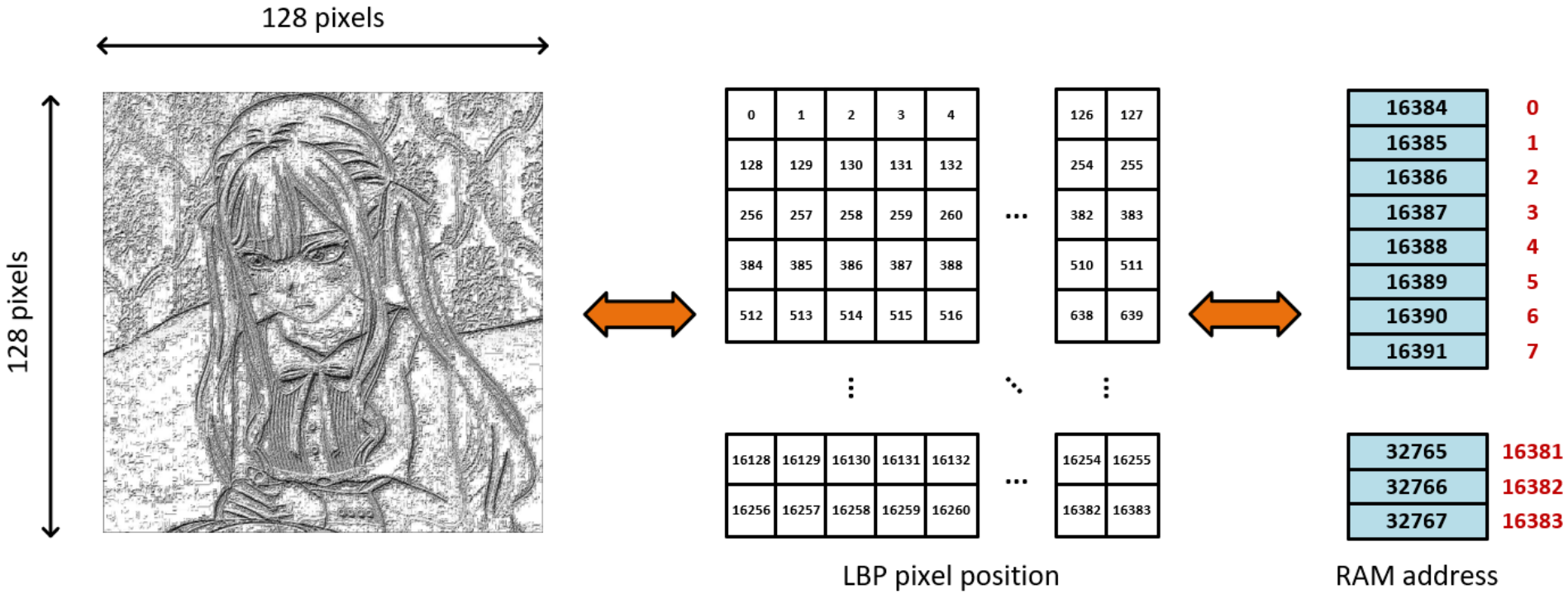
Design Description

- The image has a fixed size of 128×128 pixels, with each pixel represented by an 8-bit value (ranging from 0 to 255). This image is stored in the provided RAM that can be communicated through AXI4 protocol at address 0 to 16383.



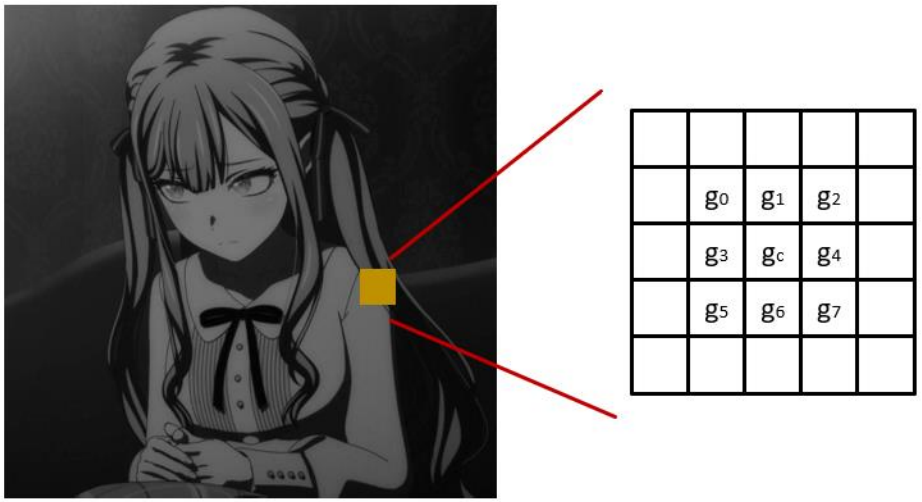
Design Description

- The output Local Binary Pattern image also has a fixed size of 128×128 pixels. The image should be store back to the provided RAM with AXI4 protocol at address 16384 to 32767.



Design Description

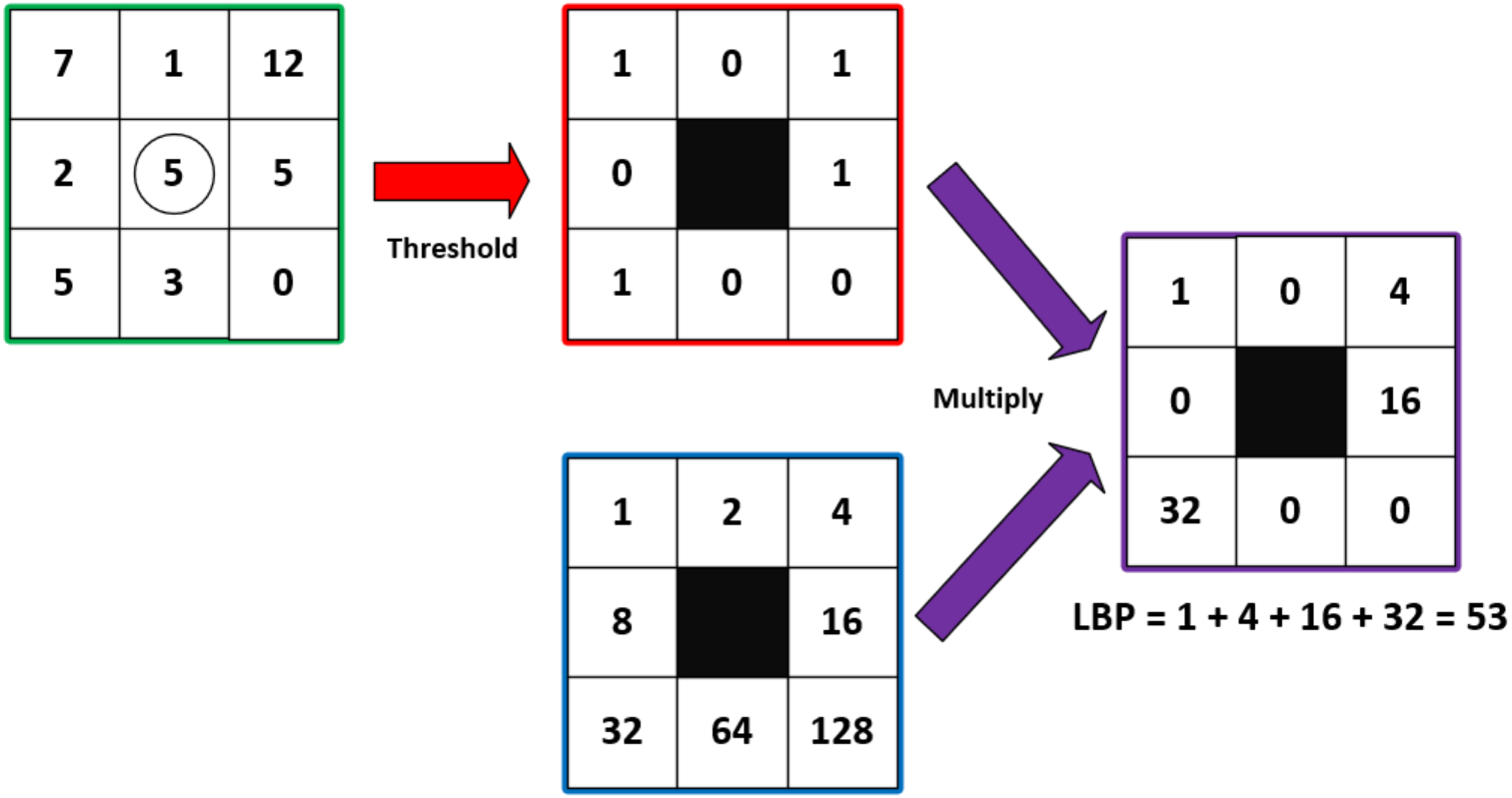
- LBP encoding method calculates the relationship between each pixel and its neighboring pixels.



$$\mathrm{LBP}(x,y)=\sum_{p=0}^{P-1}s(g_p-g_c)\cdot 2^p,\quad s(z)=\begin{cases}1, & z\geq 0\\ 0, & otherwise\end{cases}$$

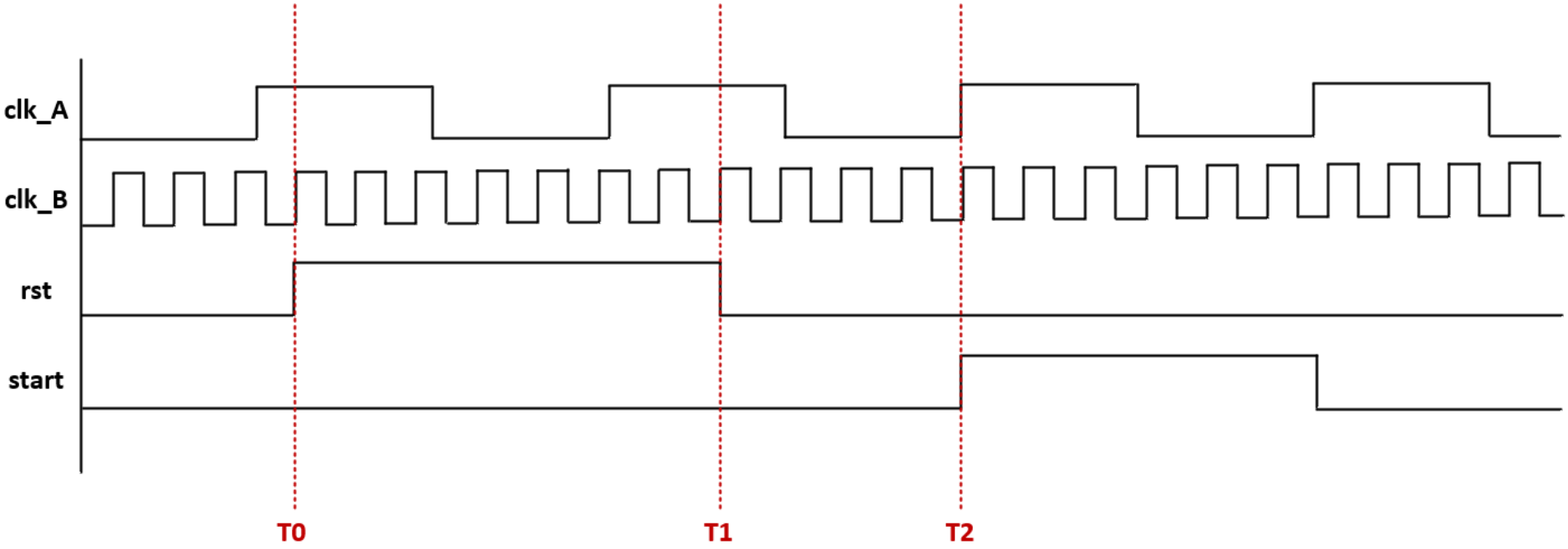
Design Description

- LBP computation example



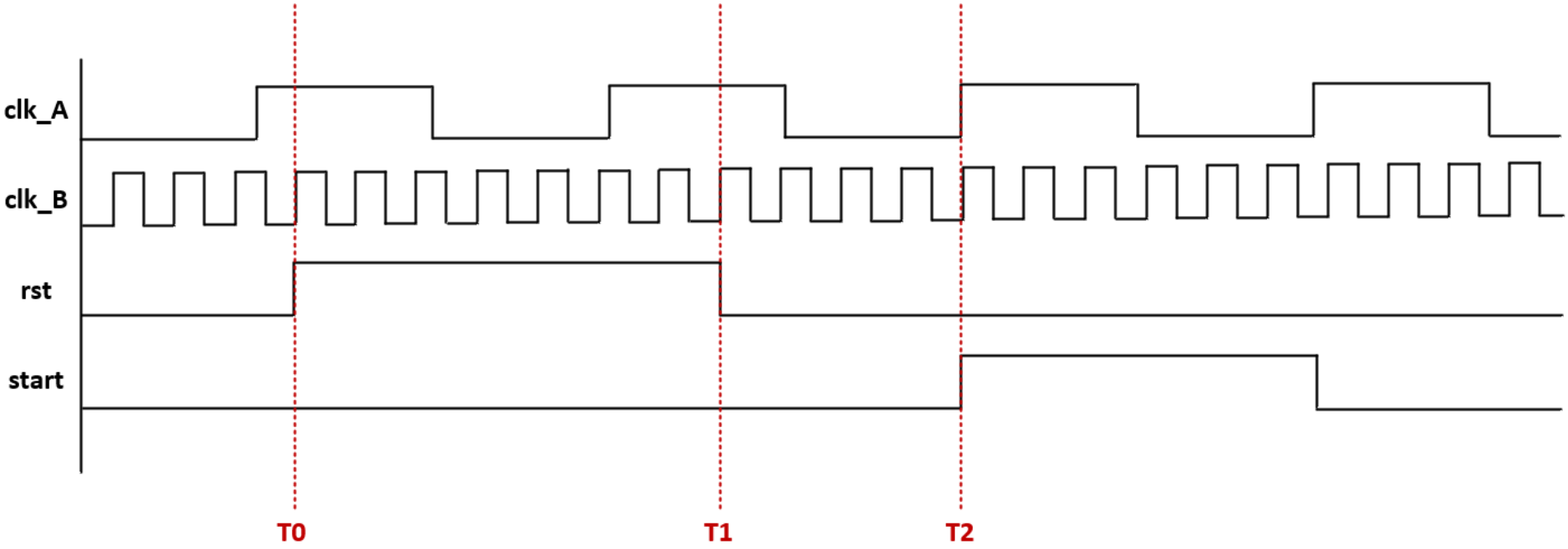
Specification (1)

- LBP is initialized between T0~T1.



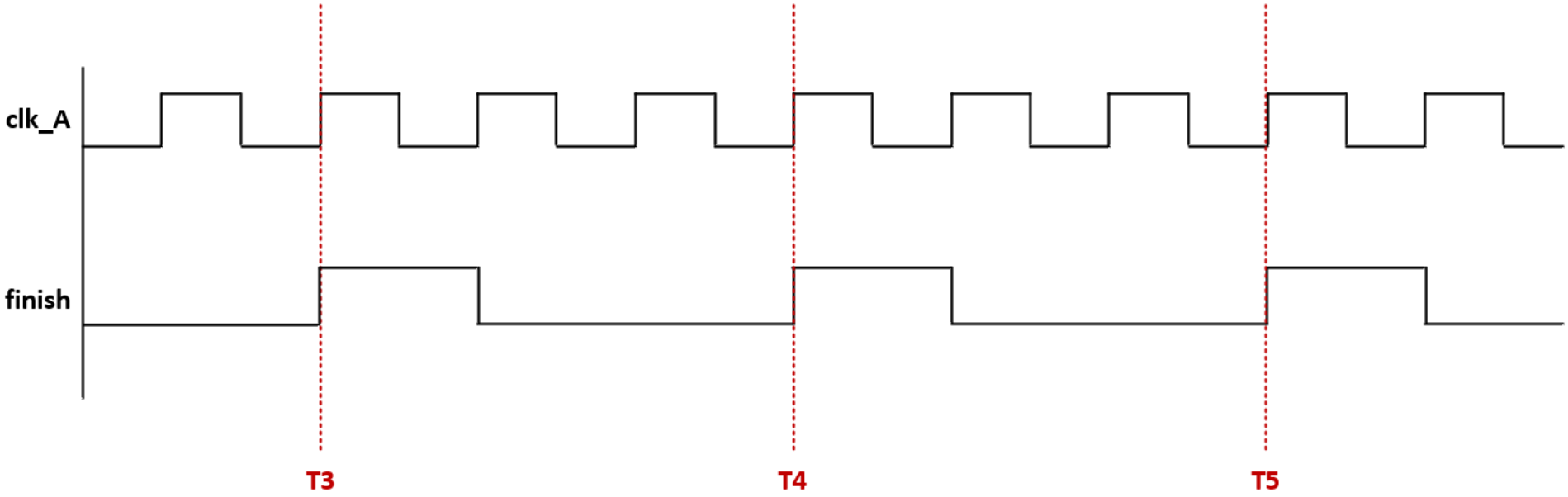
Specification (2)

- Start is set to high at T2. The signal is synchronized with clk_A. You should perform CDC to run under clk_B in your design.



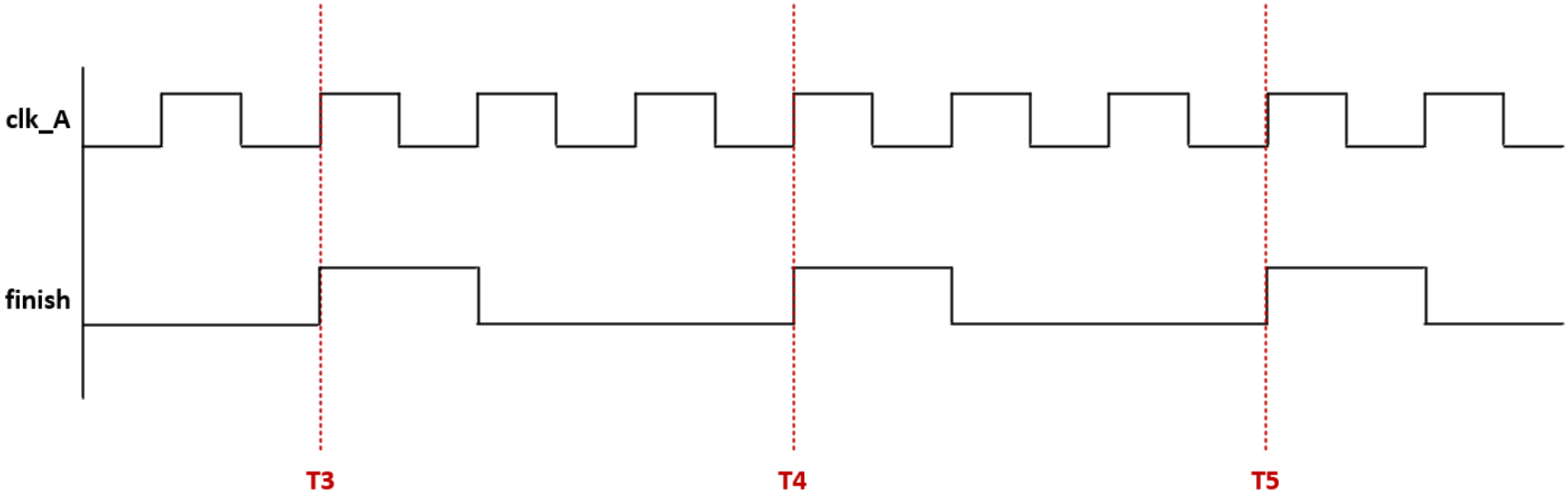
Specification (3)

- You should finish all computation for LBP and store the result back to AXI memory before T3. The finish signal should be set to high for three times then. You should perform CDC to sync finish signal with clk_A.



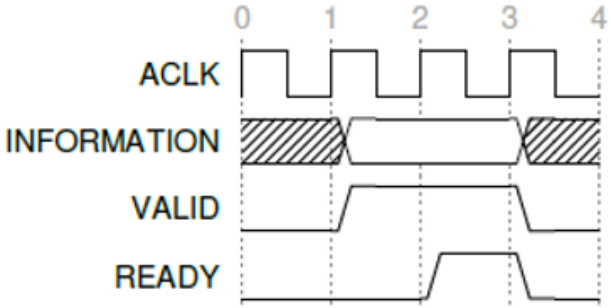
Specification (4)

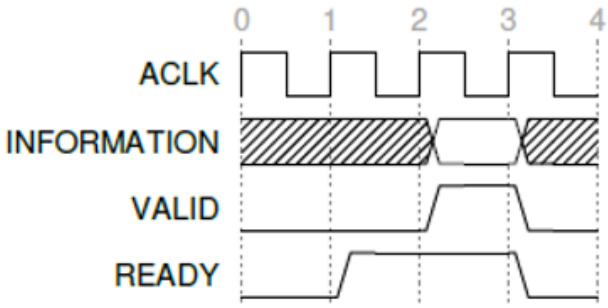
- Testbench will check the result in AXI memory with golden answer at T5.

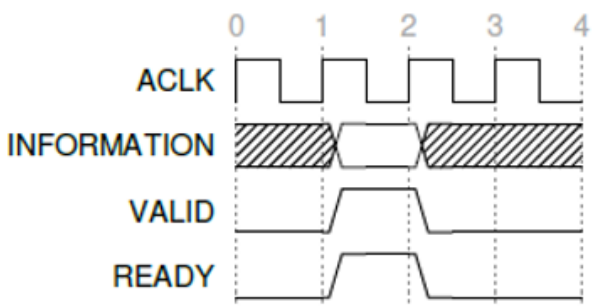


Specification (5)

- You should communicate the memory with AXI bus. There are three ways to perform handshake with READY and VALID signal. You cannot set VALID according to READY, but you can set READY according to VALID.

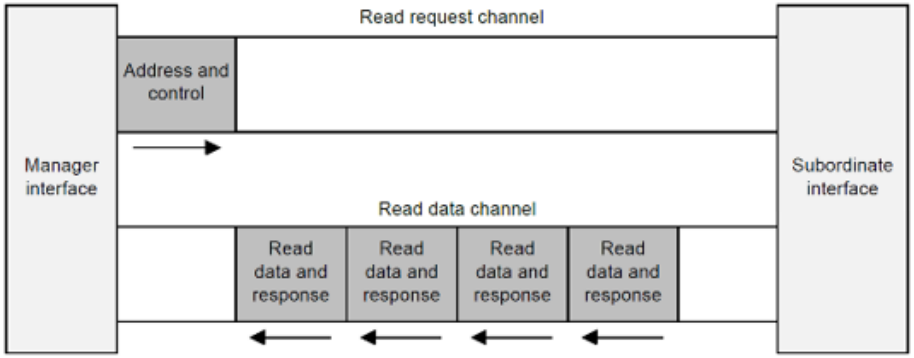


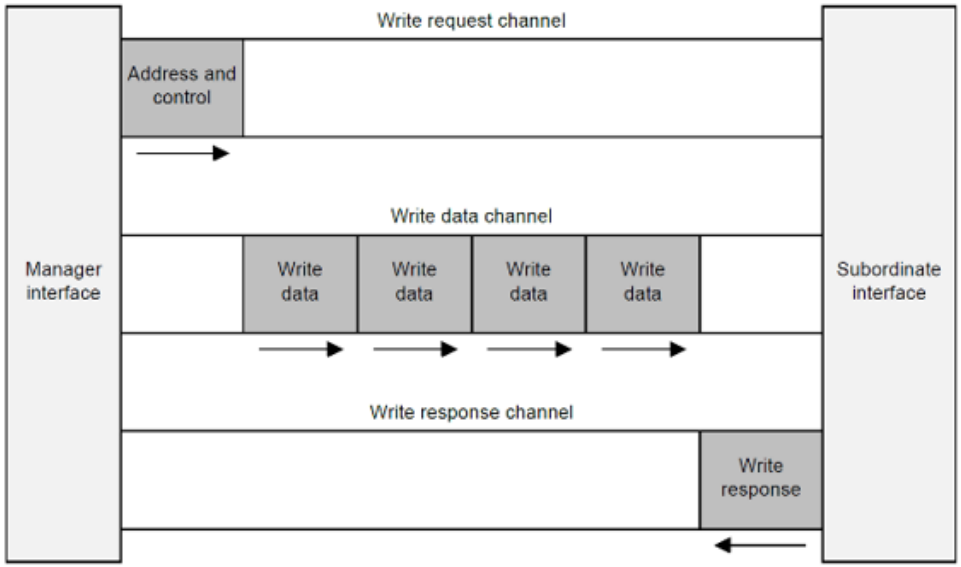


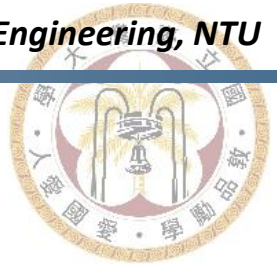


Specification (6)

- You are required to transfer address and control through request channel first, then you are able to transfer data through data channel.







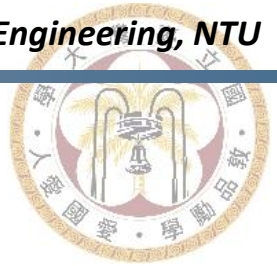
Hint

- Separate your LBP design and AXI memory access control into different modules.
- Perform CDC at the topmost module so start and finish signal can be correctly input and output.
- Sending multiple data at a time is possible using AXI protocol if you use **len** port correctly.
- The sdc file is not completed. Remember to add constraint for CDC to get correct synthesis result.

Submission

- Create a folder named studentID_hw2 and follow the hierarchy below

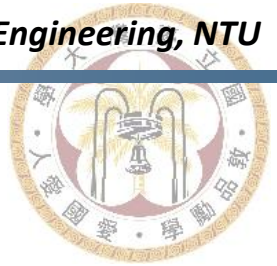
```
r13943008_hw2
        ├── 01_RTL
        │       ├── LBP.v      (along with other verilog files)
        │       └── filelist.f (include all your verilog files)
        ├── 02_SYN
        │       ├── LBP.area
        │       └── LBP.timing
        ├── 03_GATE
        │       ├── LBP_syn.sdf
        │       └── LBP_syn.v
        └── report.txt
```



Submission – Workstation

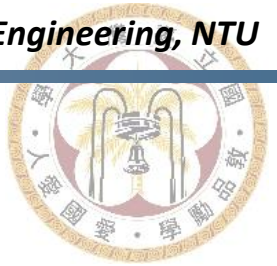
- Pack the folder **studentID_hw2** into a tar.gz file named **acvsdxxx-hw2.tar.gz**
  - **tar -zcvf acvsdxxx-hw2.tar.gz [path to studentID_hw2]**
  - Use **lowercase** for all the letters (e.g. acvsd000-hw2.tar.gz)
  - Pack the folder on ADFP server to avoid OS related problems
  - Place the tar.gz file at the root of your ADFP account

- TA will only check the last version after the homework deadline
- Reminder
  - Files uploaded to ADFP workstation must be kept as a copy in your local folder or computer



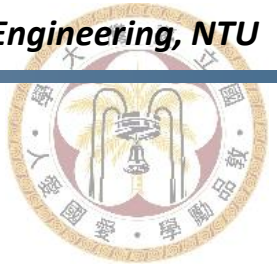
Submission – NTU COOL

- For design files (01_RTL and reports)
- Pack the folder **studentID_hw2** into a **tar.gz** file named **acvsdxxx-hw2-vk.tar.gz** (k is the number of version, k =1,2,...)
  - **tar -zcvf acvsdxxx-hw2-vk.tar.gz [path to studentID_hw2]**
  - Use **lowercase** for all the letters. (e.g. acvsd000-hw2.tar.gz)
  - Pack the folder on IC Design LAB server to avoid OS related problems
- Submit to NTU Cool



LBP.v

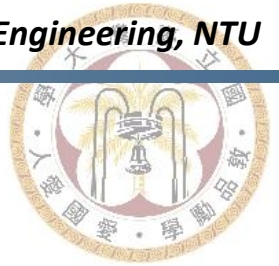
```
module LBP # (
    parameter DATA_WIDTH = 8,              // AXI4 data width
    parameter ADDR_WIDTH = 15,             // AXI4 address width
    parameter STRB_WIDTH = (DATA_WIDTH/8)  // AXI4 strobe width
)
(
    // Clock and synchronous high reset
    input                   clk_A,
    input                   clk_B,
    input                   rst,

    input                   start,
    output                  finish,

    // Data AXI4 master interface
    output [ADDR_WIDTH-1:0] data_awaddr,
    output [           7:0] data_awlen,
    output [           2:0] data_awsize,
    output [           1:0] data_awburst,
    output                  data_awvalid,
    input                   data_awready,
    output [DATA_WIDTH-1:0] data_wdata,
    output [STRB_WIDTH-1:0] data_wstrb,
    output                  data_wlast,
    output                  data_wvalid,
    input                   data_wready,
    // input  [           1:0] data_bresp,
    // input                   data_bvalid,
    // output                  data_bready,
    output [ADDR_WIDTH-1:0] data_araddr,
    output [           7:0] data_arlen,
    output [           2:0] data_arsize,
    output [           1:0] data_arburst,
    output                  data_arvalid,
    input                   data_arready,
    input  [DATA_WIDTH-1:0] data_rdata,
    input  [           1:0] data_rresp,
    input                   data_rlast,
    input                   data_rvalid,
    output                  data_rready
);
```



filelist.f

- Filelist

```
// Add your RTL files
./LBP.v

// tb
../00_TESTBED/axi/arbiter.v
../00_TESTBED/axi/axi_interconnect.v
../00_TESTBED/axi/priority_encoder.v
../00_TESTBED/axi/axi_ram.v
../00_TESTBED/testfixture.v
```



Report

- report.txt (record the power and processing time of gate-level simulation)

```
StudentID: r13943008
Area: 1725.027870 (um^2)
Total Runtime: 3097640.00 ns
```

Grading Policy

- Simulation:

	Score
RTL simulation	30%
Gate-level simulation	30%
Hidden pattern (Gate-level)	10%

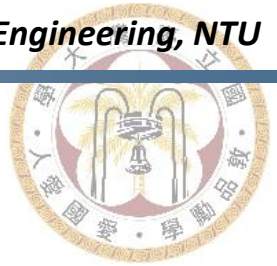
- Performance:

Score = Time*Area

Unit: Time(ns), Area(um^2)

Baseline = 10^{10}

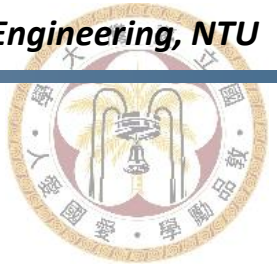
	Score
Baseline	10%
Ranking (Need to pass Baseline)	20%



Area

- Area: Cell area from synthesis report (ex. 1719.740190um^2 below)

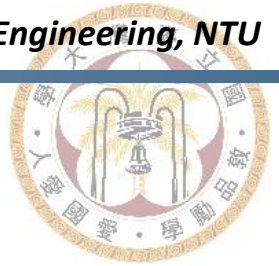
```
Information: Updating design information... (UID-85)
Library(s) Used:

    N16ADFP_StdCellss0p72vm40c_ccs (File: /usr/cad/DAFP0203001_2_X/Ex

Number of ports:                          504
Number of nets:                          5413
Number of cells:                         5017
Number of combinational cells:           4431
Number of sequential cells:               574
Number of macros/black boxes:               0
Number of buf/inv:                       2557
Number of references:                       9

Combinational area:               1156.343059
Buf/Inv area:                      573.868803
Noncombinational area:             563.397131
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                  1719.740190
Total area:                 undefined
```



Time

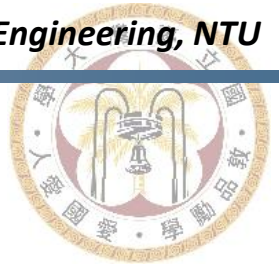
- Time: processing time from simulation (ex. 3097640.00ns below)

```
--------------------------------------------------------

Congratulations! All data have been generated successfully!

Total cost time: 3097640.00 ns
-------------------------PASS-------------------------
```



Grading Policy

- TA will use **01_run** and **03_run** to run your code at RTL and gate-level simulation.
- Do not memorize the answers directly in any way
- **No delay submission is allowed**
- Lose **5 point** for any wrong naming rule or format for submission
- **No plagiarism**



References

- [1] IC Design Contest, 2016.
- [2] AMBA AXI Protocol Specification.
- [3] R. Ginosar, Metastability and Synchronizers: A Tutorial, 2011.
- [4] Clifford E. Cummings, Clock Domain Crossing (CDC) Design & Verification Techniques Using SystemVerilog, 2008.