NYCU-EE IC LAB – Spring 2023

Lab05 Exercise

Design: Matrix Multiplication to find Trace

Data Preparation

1. Extract files from TA's directory:
   % tar xvf ~iclabta01/Lab05.tar
2. The extracted LAB directory contains:
   a. Practice/
   b. Exercise/

Design Description

In this lab, you need to create a calculator that can calculate **the trace of the product of multiple matrices** and also has the functionality to compute **the transpose of a matrix.**

In linear algebra, the trace of an n×n matrix A (generally denoted as tr(A) or Sp(A)) is equal to the sum of the eigenvalues of A. It is also equivalent to the sum of the diagonal elements of A.

$$tr(A) = \sum_{i=1}^{n} a_{ii} = a_{11} + a_{22} + \cdots + a_{nn}$$

The trace of the product of two matrices satisfies the commutative law. However, for the product of more than two square matrices, arbitrarily changing the order of matrix multiplication can result in different trace values.

$$tr(AB) = tr(BA)\ ,\ tr(ABC) \neq tr(ACB)$$
$$tr(ABC) = tr(BCA) = tr(CAB)$$

And The trace of a transpose matrix is equal to the trace of the original matrix.

$$tr(A^T) = tr(A)$$

Because the trace of a matrix has many properties, we can design a calculator to observe if there are more implicit properties in matrix multiplication and matrix transpose.

Because the matrix requires large space to store, you are suggested to **use memory(SRAM)** for finishing this lab.

- **Size :**

The **input matrix size** will be **2x2, 4x4, 8x8** and **16x16** according to the given size value which will be given at the beginning of each pattern.

■ Mode for the transpose matrix :

There are **four modes** for the transpose matrix. According to the given mode value, the mode indicates which matrices need to be transposed. (T : transpose)

mode = 2'b00 → $ABC(original)$ mode = 2'b01 → A^TBC

mode = 2'b10 → AB^TC mode = 2'b11 →ABC^T

■ Rules of input data :

In this exercise, you will get **a matrix size and 32 matrices continuously** at the beginning of each pattern. After that, you will get **the mode for the transpose matrix** and **three input matrix indices continuously**. **The order of the indices determines the order of matrix multiplication**. The indices range from 0 to 31. After **ten sets** of indices(each set containing three indices), you will get the next pattern.

■ Rules of output result :

After finishing matrix multiplication, compute the **trace** and output it for **one cycle**.

Inputs

Input	Bit Width	Definition and Description
clk	1	Clock.
rst_n	1	Asynchronous active-low reset.
in_valid	1	High when input signals are valid. It will be tied high for **32*current size of matrix(2x2, 4x4…) cycles** continuously for **matrix**. And the first cycle will give **matrix_size.**
in_valid2	1	High when input signals are valid. It will be tied high for 3 cycle continuously for **matrix_idx**. And the first cycle will give **mode.**
matrix	8	Elements of input matrix. It will be sent in **raster scan order** continuously when **in_valid** is high. The elements are signed integers, which are represented in 2's complement format.
matrix_size	2	The signal will determine the dimension of input matrix. It will be given in first cycle when **in_valid** is high. **2'b00:** 2x2. **2'b01:** 4x4. **2'b10:** 8x8. **2'b11:** 16x16.
matrix_idx	5	Input matrix index, this signal will be given when **in_valid2** is high.

mode	2	The signal will determine the mode for the transpose matrix. It will be given in first cycle when **in_valid2** is high. **2'b00:** ABC(original). **2'b01:** A^T BC. **2'b10:** AB^T C. **2'b11:** ABC^T

Outputs

Output	Bit Width	Definition and Description
out_valid	1	High when out is valid. It cannot be overlapped with **in_valid** and **in_valid2** signal.
out_value	50	It will output the trace of matrix after finishing matrix multiplication. The elements are signed integers, which are represented in 2's complement format.

1. The **matrix** signal is delivered in **raster scan order** for **32*current size of matrix(2x2, 4x4…) cycles continuously when in_valid is tied high**. When matrix finish delivered, it will be tied to unknown state, and **in_valid** will also be tied low.

2. The input of **matrix_size** is delivered for **only one cycle** during the **first cycle of in_valid tied high**. After that, the matrix_size signal is tied to unknown state.

3. Every time in_valid2 is triggered, it is tied high **for three cycle.**

4. The in_valid2 signal will be triggered **for total 10 times(for three cycles)** after in_valid is tied low in a single pattern. After each time in_valid2 triggers, your design will do the matrix multiplication with specific matrices and then out_valid will be tied high for one cycle.

5. In each pattern, the in_valid2 signal will be triggered 1~3 cycles after in_valid is tied low, and the other nine times in_valid2 will be triggered 1~3 cycles after after out_valid is tied low.

6. The next input pattern will be triggered 1~5 cycles after **the tenth** out_valid of this pattern falls.

7. The input of **matrix_idx** is delivered for **three cycles** during in_valid2 tied high. After that, the **matrix_idx** is tied to unknown state.

8. All input signals are synchronized at negative edge of the clock.

9. The **out_value** must be delivered for **one cycle** and **out_valid** should be high simultaneously.

Specifications

1. Top module name: MMT (Design file name：MMT.v)

2. **It is asynchronous reset and active-low architecture. If you use synchronous reset (considering reset after clock staring) in your design, you may fail to reset signals should be reset after the reset signal is asserted.**

3. **The reset signal (rst_n) would be given only once at the beginning of simulation. All output signals should be reset after the reset signal is asserted.**
4. You can adjust your clock period by yourself, but the maximum period is **20 ns**.
5. The data type in the synthesis result **CAN NOT** include any **LATCH**.
6. After synthesis, the area report is valid only when the slack in the end of timing report is **non-negative** and the result should be **MET**.
7. The next input pattern will come in **1~5** cycles after the tenth **out_valid** of this pattern is pulled down.
8. The **out_valid** cannot overlap with **in_valid** and **in_valid2**.
9. The execution latency is limited in **10000 cycles.** The latency is the clock cycles between the falling edge of the **in_valid2** and the rising edge of the **out_valid**.
10. In this lab, you **must use the memory and generate it yourself. The number of words and the bits per each word is defined by yourself. The total number and kind of memory is unlimited. We will check it at MMT.area in 02_SYN/Report/ folder. The area of Macro/Black Box must not be 0. The example is shown in following figure.**

```
Combinational area:          1821995.696653
Buf/Inv area:                 111973.280126
Noncombinational area:        343750.185371
Macro/Black Box area:         214305.703125
Net interconnect area:        undefined  (No wire load specified)

Total cell area:             2380051.585150
```

Fig 1. The area of your memory

11. The total cell area should not larger than **15,000,000μm^2. Also, the synthesis time should be less than 2 hours.**
12. **If any port of memory is connected with mismatch width, the memory will not be synthesized and you will get an error message as shown in Fig 2. Even though the design may still pass gate level simulation, this situation will be regarded as synthesis fail. In this case, memory area will be 0 in MMT.area. We will check it at syn.log and MMT.area.**

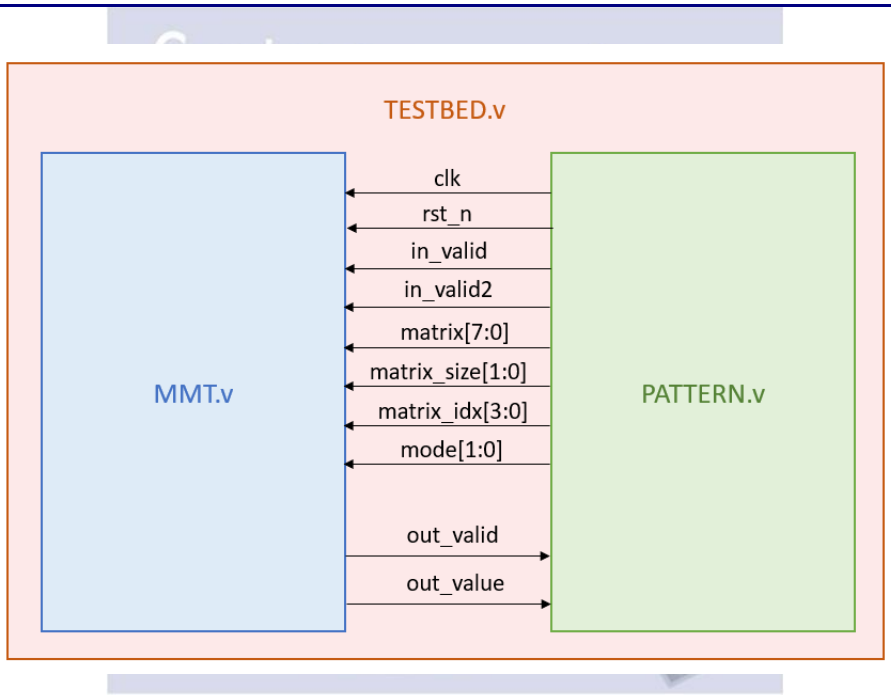
```
Error: Width mismatch on port 'A' of reference to 'RA1SH_256_32_4' in 'CNN'. (LINK-3)
Warning: Unable to resolve reference 'RA1SH_256_32_4' in 'CNN'. (LINK-5)
```

Fig 2. Memory port width mismatch error

13. **All numbers are signed integers and expressed in 2's complement format. Be sure the operations are done with signed operations.**
14. **Every** output signal should be correct when **out_valid** is high. And **Every** output signal should be low when **out_valid** is low.
15. The input delay is set to **0.5*(clock period)**.
16. The output delay is set to **0.5*(clock period)**, and the output loading is set to **0.05**.

17. The gate level simulation cannot include any timing violations without the *notimingcheck* command.
18. Don't use any wire/reg/submodule/parameter name called *error*, *congratulation*, *latch* or *fail* otherwise you will fail the lab. Note: * means any char in front of or behind the word. e.g: error_note is forbidden.
19. Don' t write Chinese comments or other language comments in the file you turned in.
20. Verilog commands //synopsys dc_script_begin, //synopsys dc_script_end //synopsys translate_off, //synopsys translate_on are only allowed during the usage of including and setting designware IPs, other design compiler optimizations are forbidden.
21. Using the above commands are allowed, however any error messages during synthesis and simulation, regardless of the result will lead to failure in this lab.

Block Diagram



Grading Policy

The performance is determined by the area and latency of your design. The less cost your design has, the higher grade you get.

- Function Validity: 70%
- Performance: 30% **Area^2 * Total Latency <Total latency=cycle time*(latency+1)>**

Note

1. **Please submit your files under 09_SUBMIT before 12:00 at noon on March. 27:**
   - If uploaded files violate the naming rule, you will get 5 deduct point.
   - In this lab, you can adjust your clock cycle time. Consequently, make sure to key in your clock cycle time after the command like the figure below. It's means that the TA will demo your design under this clock cycle time

```
[Exercise/09_SUBMIT]% ./01_submit 10.9
```

After that, you should check the following files under **09_SUBMIT/Lab05_iclabXXX/**

RTL design : **MMT_iclabXXX.v** (**XXX** is your account no.)
clock_cycle_iclabXXX.txt

Memory file : **MEMORY_NAME_iclabXXX.v**
MEMORY_NAME_iclabXXX.db
file_list_iclabXXX.f

If you miss any files on the list, you will fail this lab.

Then use the command like the fugure below to check the files are uploaded or not.

```
[Exercise/09_SUBMIT]% ./02_check 1st_demo
```

- ***Example:***
  - Submit your design files：
    MMT_iclab999.v
    10.9_iclab999.txt
  - Given two memories in your design, RA1SH1 and RA1SH2
    - **A. Submit these memory files:**
      RA1SH1_iclab999.v RA1SH1_iclab999.db
      RA1SH2_iclab999.v RA1SH2_iclab999.db
    - **B. Type following in file_list_iclab999.f and submit it：**
      ../04_MEM/RA1SH1.v
      ../04_MEM/RA1SH2.v

2. **Template folders and reference commands:**

   01_RTL/ (RTL simulation) **./01_run**
   02_SYN/ (Synthesis) **./01_run_dc**
   (Check **latch** by searching the keyword "**Latch**" in 02_SYN/**syn.log**)
   (Check the design's timing in /Report/MMT.timing)
   (Check the design's area in /Report/MMT.area)
   03_GATE/ (Gate-level simulation) **./01_run**
   04_MEM/ (Memory location)
   (You should generate your own memory and put the required files here)
   09_SUBMIT/ (submit your files) **./01_submit** **./02_check**

   ➢ You can key in **./09_clean_up** to clear all log files and dump files in each folder

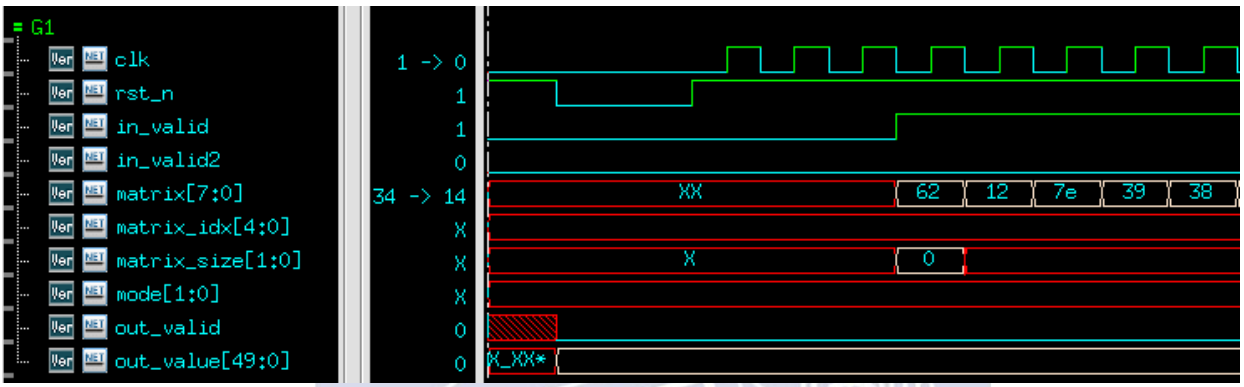
   ※You should make sure the **three clock period values identical** in 00_TESTBED/PATTERN.v and 02_SYN/syn.tcl

```
`ifdef RTL
    `timescale 1ns/10ps
    `include "MMT.v"
    `define CYCLE_TIME 20.0
`endif
`ifdef GATE
    `timescale 1ns/10ps
    `include "MMT_SYN.v"
    `define CYCLE_TIME 20.0
`endif
```

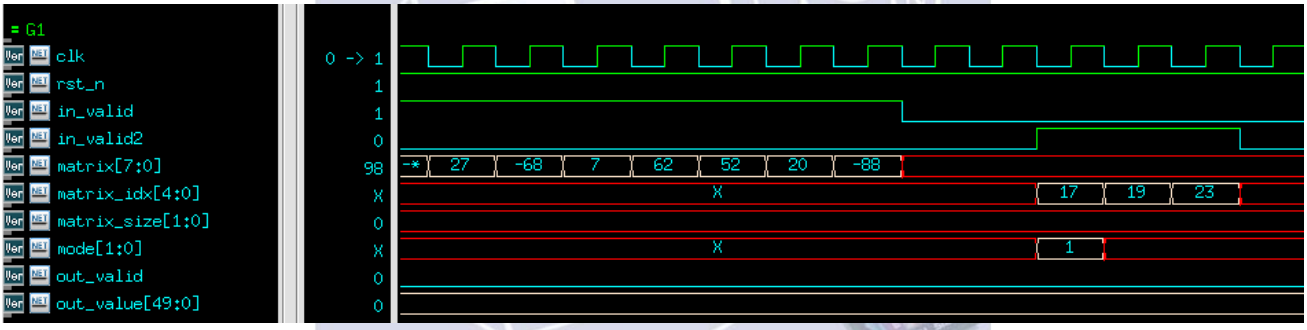
```
set DESIGN "MMT"
set CYCLE 20.0
```

Example Waveform

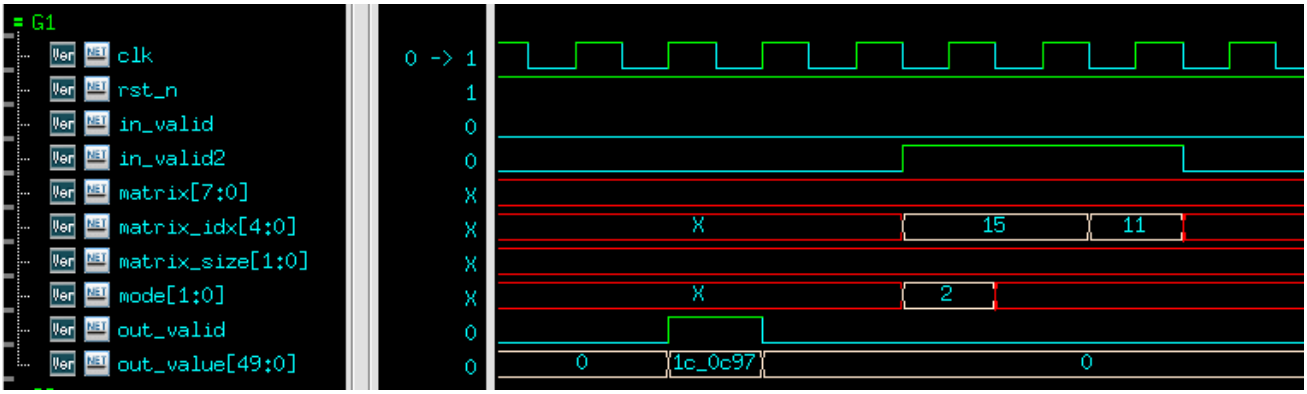
1.Asynchronous reset and active-low reset all output



2. The in_valid2 signal will be triggered 1~3 cycles after in_valid is tied low



3. The other nine times in_valid2 will be triggered 1~3 cycles after out_valid next



4. The next input pattern will be triggered 1~5 cycles after **the tenth** out_valid of this pattern falls.