NYCU-EE IC LAB - SPRING2024

Lab07 Exercise

Design: Matrix Multiplication with Clock Domain Crossing

Data Preparation

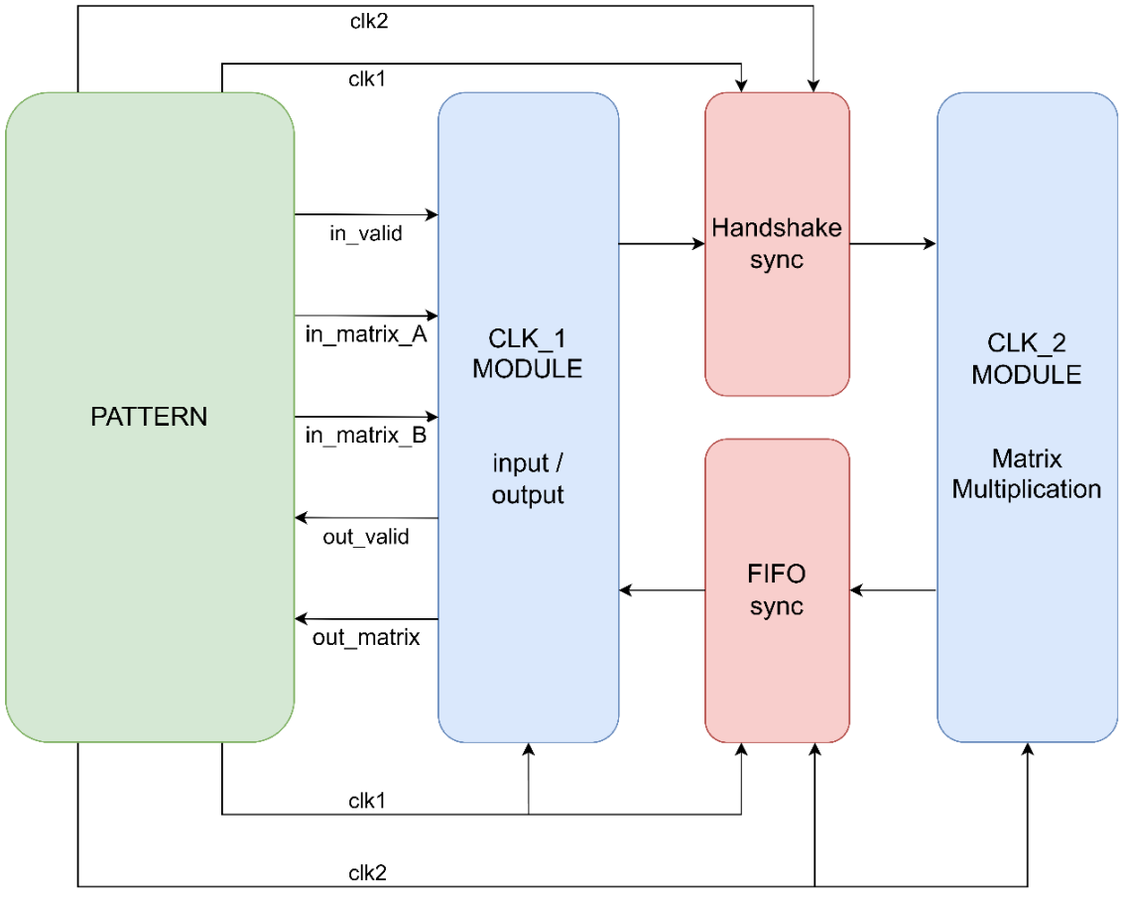
1. Extract Lab directory from TA's directory:

   % tar -xvf ~iclabTA01/Lab07.tar

Basic Concept

In this lab, you will receive matrix A_{16x1} and matrix B_{1x16} and you should calculate $A_{16x1} * B_{1x16} = C_{16x16}$. Then, output the matrix C element by element. The detailed structure is described below.

1. The input matrix A and matrix B is given in clk1 domain.
2. Use Handshake synchronizer to transfer the data into clk2 domain.
3. Calculate the result in clk2 domain.
4. Use FIFO synchronizer to transfer the data to clk1 domain.
5. Output the result element by element in the clk1 domain.



Design Description

In this lab, you are asked to implement Matrix Multiplication.

For input signal in_matrix_A and in_matrix_B, you will receive the matrix element as the following sequence:

A_0 、A_1 、A_2 、…… 、A_{13} 、A_{14} 、A_{15}

B_0 、B_1 、B_2 、…… 、B_{13} 、B_{14} 、B_{15}

After implement the Matrix Multiplication: A_{16x1} * B_{1x16} = C_{16x16}. You should output the result as the following sequence:

C_0 、C_1 、C_2 、…… 、C_{253} 、C_{254} 、C_{255}

A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15

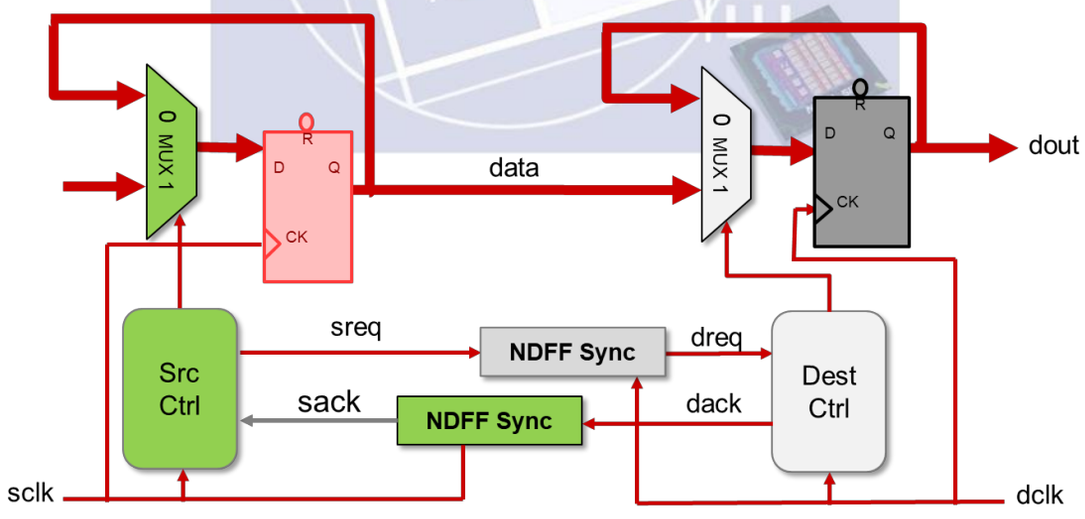
X

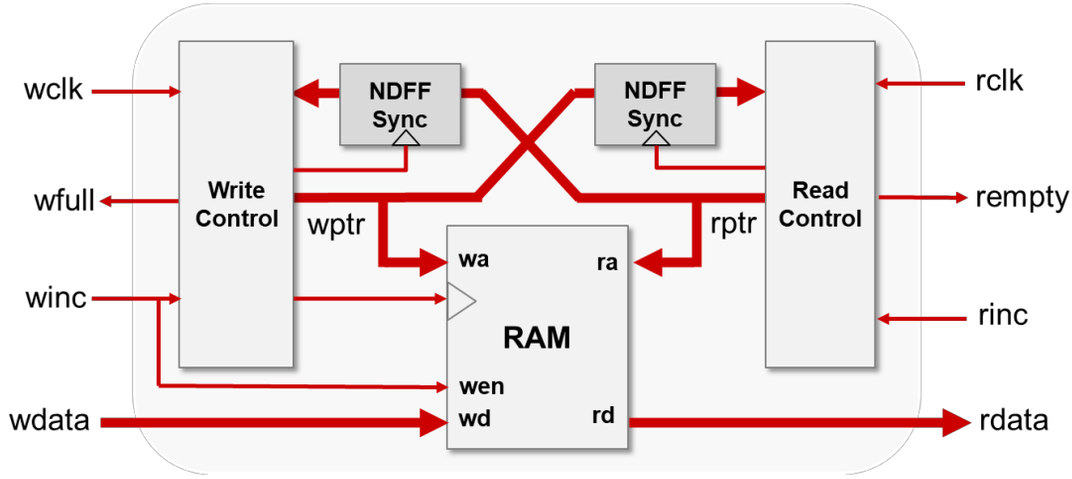
B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15

=

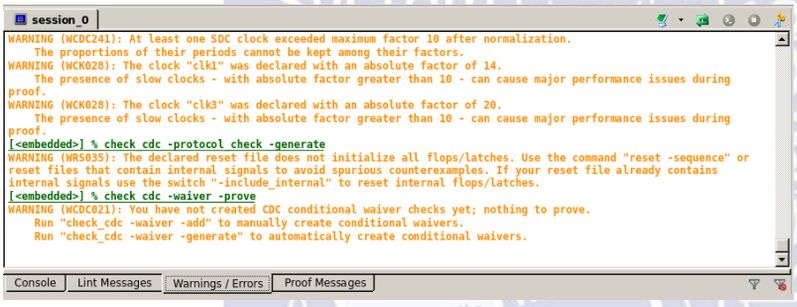
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	C26	C27	C28	C29	C30	C31
C224	C225	C226	C227	C228	C229	C230	C231	C232	C233	C234	C235	C236	C237	C238	C239
C240	C241	C242	C243	C244	C245	C246	C247	C248	C249	C250	C251	C252	C253	C254	C255

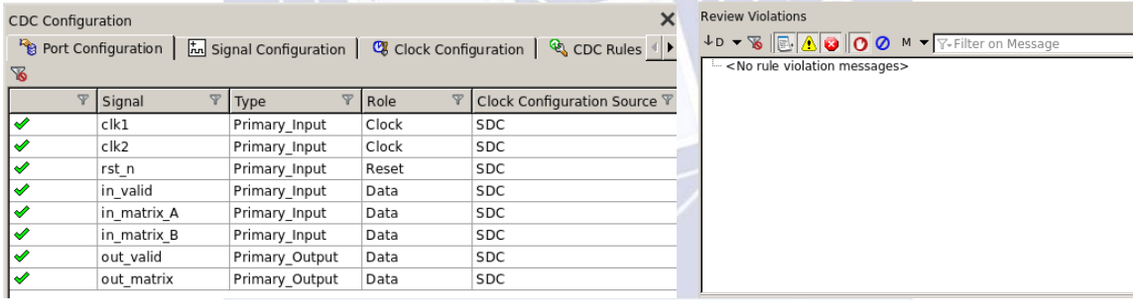
In this lab, you will also deal with the CDC (cross-chronological domain) problem. The Handshake synchronizer is used to cross clk1 to clk2. The FIFO synchronizer is used to cross clk2 to clk1. Handshake and FIFO circuit structures are shown below:





In this lab, you should use JG to verify the CDC design. After run the JG, there should not be error message in console, violation message and anything in violation.csv. (like the figure below)





Inputs

I/O	Signal name	Bits	Description
Input	**clk1**	1	In 01_RTL： Positive edge trigger clock by clock 1 with 4 different clock period 4.1ns, 7.1ns, 17.1ns, 47.1ns In 03_GATE： Positive edge trigger clock by clock 1 with clock period 47.1ns
Input	**clk2**	1	Positive edge trigger clock by clock 2 with clock period 10.1 ns
Input	**rst_n**	1	Asynchronous reset active low reset
Input	**in_valid**	1	Indicate in_matrix_A and in_matrix_B signals are valid when in_valid is high level. This signal is triggered by **clk1 for 16 cycles**
Input	**in_matrix_A**	4	The **unsigned** data for Matrix Multiplication. These 16 data are for matrix A_{16x1}. This signal is triggered by **clk1 for 16 cycles**
Input	**in_matrix_B**	4	The **unsigned** data for Matrix Multiplication. These 16 data are for matrix B_{1x16}. This signal is triggered by **clk1 for 16 cycles**

Outputs

I/O	Signal name	Bits	Description
output	**out_valid**	1	Should be set to low after reset and not be raised when **invalid** is high. Should set to high when your **out_matrix** is ready. Should be pulled up for 256 cycles, not required to be continuous. This signal is triggered by **clk1.**
output	**out_matrix**	8	The **unsigned** result for matrix multiplication. Should be output for 256 cycles, not required to be continuous. This signal is triggered by **clk1.**

Specifications

Top module

1. Top module name : MM_TOP (File name: MM_TOP.v)
2. Submodule name : CLK_1_MODULE, CLK_2_MODULE
   (File name: DESIGN_MODULE.v)
3. Synchronizer name : Handshake_syn, FIFO_syn
   (File name: Handshake_syn.v, FIFO_syn.v in synchronizer folder)
4. Synchronizer from TA: NDFF_syn, NDFF_BUS_syn
   (File name: NDFF_syn.v, NDFF_BUS_syn.v in synchronizer folder)
5. Dual port SRAM : DUAL_64X8X1BM1 (04_MEM folder)
   (Words: 64, Bits: 8, Dual port SRAM)
6. Input pins : **clk1, clk2, rst_n, in_valid, [3:0] in_matrix_A, [3:0] in_matrix_B**
   Output pins : **out_valid, [7:0] out_matrix**

Reset

7. Use **asynchronous** reset active low architecture.
8. 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.

Input/Output Signal

9. Input and output data are synchronous to clk1.
10. The out_matrix should be correct when out_valid is high.
11. The out_matrix should be reset after your out_valid is pulled down.
12. Output signal out_valid and out_matrix should be zero when in_valid is high.
13. The next input pattern will come in 1~3 clk1 cycles after getting 256 output data.
14. The output should be raised for 256 cycles and is not required to be continuous.

Synthesis and Prime Time

15. Output delay is 0.5 * clk1 Clock Period.
16. Input delay is 0.5 * clk1 Clock Period.
17. The output loading is set to 0.05.
18. Your design area should not > 2000000.
19. In the synchronizer you design, please use the NDFF_syn, NDFF_BUS_syn, provided by TA if needed. Prime Time will ONLY set_annotated_check to the NDFF_syn module provided by TA.
20. The MM_TOP.sdc is complete by TA. DO NOT modify it. This file is to set the asynchronous clock groups of clk1 and clk2.

```
set_clock_groups -name group1 -asynchronous -group {clk1} -group {clk2}
```

21. After synthesis, check the “MM_TOP.area” and “MM_TOP.timing” in the folder “Report”. The area report is valid only when the slack in the end of “MM_TOP.timing” is non-negative.

22. The synthesis result cannot contain any latch, error, violation, mismatch (in syn.log).
23. After run Prime Time, the slack in the end of "MM_TOP_pt.timing" should be also non-negative.
24. The Prime Time result cannot contain any error, violation (in syn.log).

Gate level simulation

25. You can't have timing violation in gate-level simulation.

Clock period and Latency

26. The design should be able to operate at different output cycles. Please take advantage of the FIFO synchronizers. TA will demo your design at 4 different clk1 period (4.1ns, 7.1ns, 17.1ns, 47.1ns) in the 01_RTL stage.
27. In the 02_SYN and 03_GATE stages, the clk1 period will be fixed at 47.1ns, and latency is calculated based on this.
28. The latency is from the falling edge of in_valid to the falling edge of out_valid for the last output, including the output cycles!!!!!!
29. Your latency should be smaller than 5000 cycles in clk1.

Dos and Don'ts

30. Changing top module is prohibited.
31. Don't use Designware IP.
32. Calculate the result in CLK_2_MODULE, use CLK_1_MODULE to perform the Matrix Multiplication is prohibited.
33. Changing clock period is prohibited. Use the clocks listed above.
34. TA had generated dual port SRAM for the FIFO synchronizer, and the files are stored in 04_MEM. Don't modify them.
35. You should use dual port SRAM provided by TA to design your FIFO synchronizers to maintain the fairness of area performance.
36. Don't modify the parameter "WIDTH" in Handshake_syn.v and FIFO_syn.v
37. Don't use any wire/reg/submodule/parameter name called *error*, *congratulation*, *pass*, *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.
38. Don't write chinese comments or other language comments in the file you turned in. Otherwise, you will get 5 deduct points.

Supplement

39. Some pre-defined flags are reserved for you to optimize your design.
40. It's acceptable for the following two warning.

```
Warning-[SDFCOM_CFTC] Cannot find timing check
MM_TOP_SYN_pt.sdf, 35902
module: QDFFRBS, "instance: TESTBED.I_MM.u_FIFO_syn_wptr_reg_0_"
  SDF Warning: Cannot find timing check $hold(posedge CK,posedge RB,...)
```

```
** MEM_Warning: Read and Write the same Address, DO is unknown (      3275112850 ps) in clock of TESTBED.I_MM.u_FIFO_syn.u_dual_sram.ErrorMessage
```

Grading Policy

- Function correct 70% (01_RTL to 03_GATE, 4 different clk1 cycle times in 01_RTL)
- Jasper Gold correct 25%
- Performance: Latency * Area^2 5% (Latency is calculated in clk1)
  If you didn't pass Function or JG, your score would not include performance.
- The grade of 2nd demo would be 30% off.
  Ex1: Pass function but fail JG in 1st demo. 70%
  Ex2: Pass function and JG in 2nd demo. (70% + 25% + performance 5%) * 0.7
- The latency is from the falling edge of in_valid to the falling edge of out_valid for the last output, including the output cycles.

Note

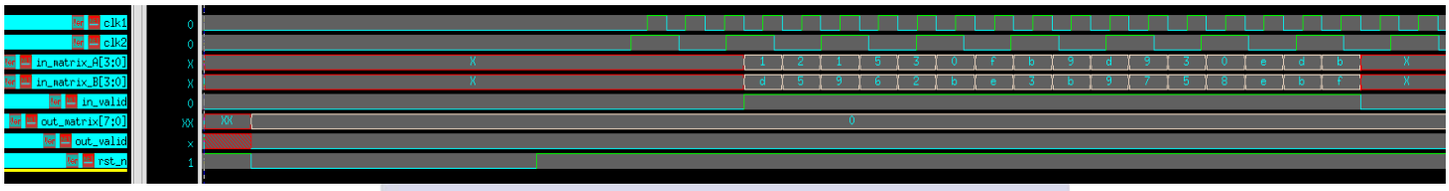
Template folders and reference commands:

1. 01_RTL/ (RTL simulation)
   I. **./ 01_run_vcs_rtl (<CYCLE_TIME_clk1>)**
      The default of CYCLE_TIME_clk1 is 47.1ns.
2. 02_SYN/ (Synthesis)
   I. ./01_run_dc_shell
      (Check the design which contains **latch and error** or not in **syn.log**)
   II. **./02_run_pt**
      (**set_annotated_check** for the first FF of NDFF synchronizer)
      (Check the design's timing in /Report/ MM_TOP_pt.timing)
3. 03_GATE_SIM/ (GL simulation)
   I. **./01_run_vcs_gate**
      (We will only run 47.1ns for clk1 period in 03 Gate Level simulation)
      (Check no timing violation)
4. 05_JG/ (CDC verification)
   I. **./01_run_jg**
5. 09_SUBMIT/ (submit file)
   I. **./00_tar**
   II. **./01_submit**
   III. **./02_check**
   1st_demo deadline: 2024/4/22 (Mon.) 12:00:00
   2nd_demo deadline: 2024/4/24 (Wed.) 12:00:00
6. You can key in **./09_clean_up** to clear all log files and dump files in each folder.

7. **You need to upload your design and system will name them as DESIGN_module_iclabxx.v, Handshake_syn _iclabxx.v and FIFO_syn_iclabxx.v. (you should check with ./02_check)**
8. **If the uploaded file violating the naming rule, you will get 5 deduct points.**

Waveform Example

1. Asynchronous reset and active-low and reset all output.
2. 16 cycles for input signals



3. 256 cycles for output signals and not required to be continuous