NYCU-EE IC LAB – Fall 2023

Lab07 Exercise

Design: Pseudo Random Number Generator (PRNG)

Data Preparation

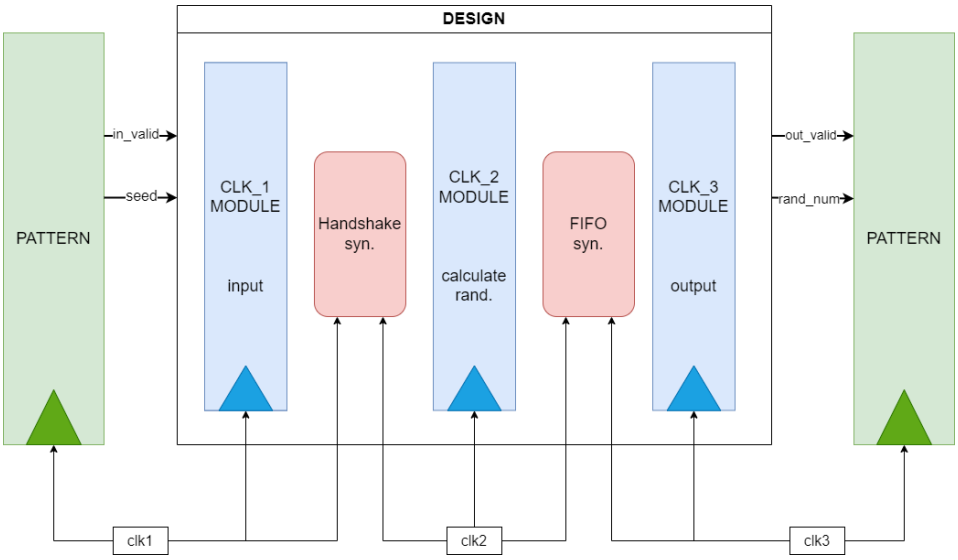
1. Extract test data from TA's directory:
   % openssl des3 -d -k SRMMZwzAvfY= -salt -in ~iclabTA01/Lab07.tar | tar xvf -
2. The extracted LAB directory contains:
   a. **00_TESTBED**
   b. **01_RTL**
   c. **02_SYN**
   d. **03_GATE**

Basic Concept

There are many algorithms for pseudo-random number generation (PRNG) to generate random number. Xorshift family of PRNGs are known for their simplicity and speed, especially on modern hardware.

In this lab, you should calculate 256 random numbers by the given seed. The structure are describe below.

1. The seed is given in clk1 domain.
2. Use Handshake synchronizer to transfer the seed into clk2 domain.
3. Calculate the random number in clk2 domain.
4. Store the result into FIFO synchronizer.
5. Get the random number from FIFO synchronizer and output the result in the clk3 domain.



Design Description

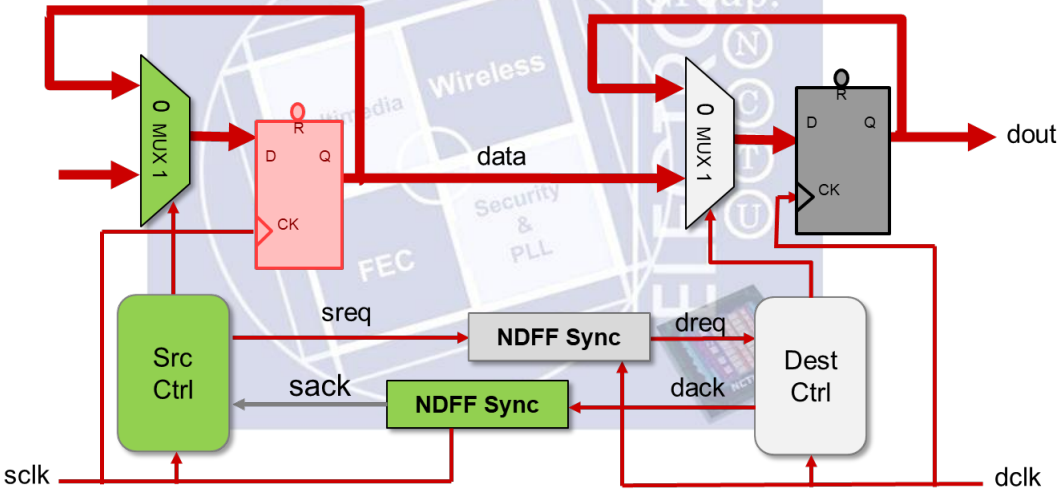
In this lab, you are asked to implement a Xorshift algorithm. The algorithm is described below:

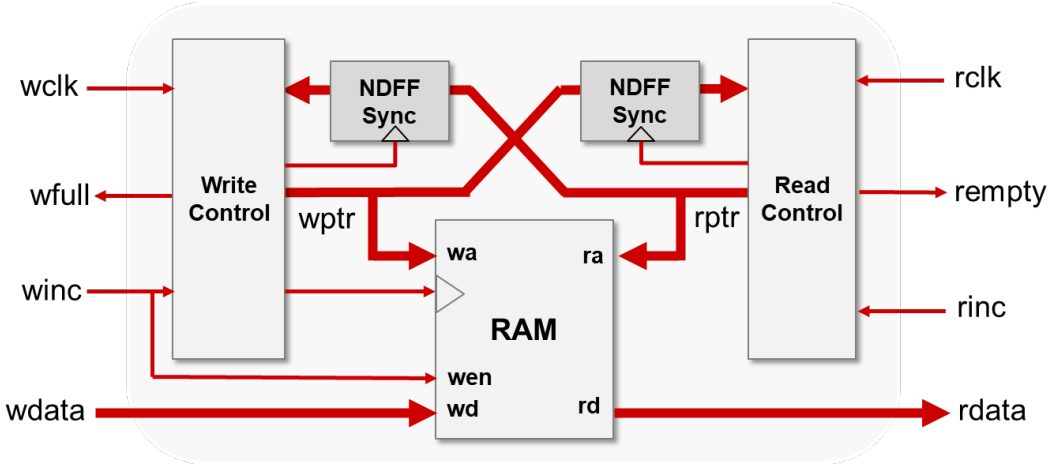
1. **Initialization**: start with a non-zero seed X_0
2. **Generation**: for each desired random number:
   - i. $X_{n+1} = X_n$
   - ii. $X_{n+1} = X_{n+1} \oplus (X_{n+1} \ll a)$ (left shift and XOR)
   - iii. $X_{n+1} = X_{n+1} \oplus (X_{n+1} \gg b)$ (right shift and XOR)
   - iv. $X_{n+1} = X_{n+1} \oplus (X_{n+1} \ll c)$ (left shift and XOR)
   - v. Return X_{n+1}

   Here, $\oplus$ denotes the XOR operation, $\ll$ denotes left shift, $\gg$ denotes right shift, and a, b, c are chosen constants.

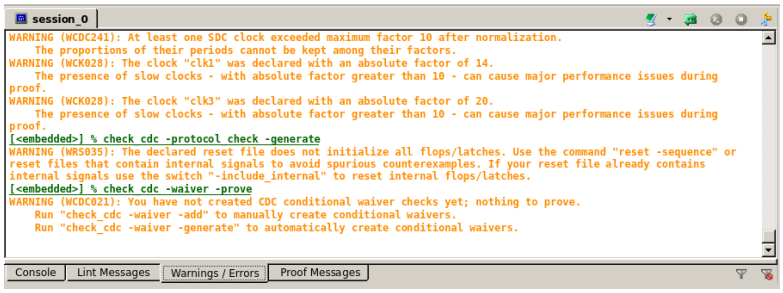
   For 32-bit version, a=13, b=17, c=5.

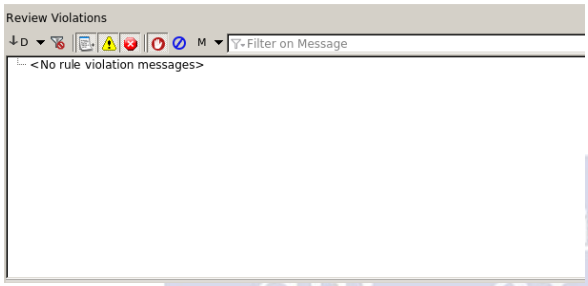
In this lab, you will also deal with the CDC (cross-chronological domain) problem. The Handshake synchronizer is used to cross clk1 to clk2. The FIFO synchronizer is used to cross clk2 to clk3. The circuit structure are shown below:





In this lab, you should use JG to verify the CDC design. After run the JG, there should not be no error message in console, no violation message and nothing in violation.csv.





Inputs

I/O	Signal name	Bits	Description
Input	**clk1**	1	Positive edge trigger clock by clock 1 with clock period 14.1ns
Input	**clk2**	1	Positive edge trigger clock by clock 2 with clock period 3.9ns
Input	**clk3**	1	Positive edge trigger clock by clock 3 with clock period 20.7ns (01_RTL to 03_GATE) 4 different clk3 period (4.1ns, 20.7ns, 67.3ns, 99.9ns) in the 01_RTL stage
Input	**rst_n**	1	Asynchronous reset active low reset
Input	**in_valid**	1	**seed** are valid when **in_valid** is high This signal is trigger by **clk1 for one cycle**
Input	**seed**	32	The seed for Xorshift algorithm. A positive number X_0.

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 **rand_num** is ready. Should be pulled up for 256 cycles, not required to be continuous.
output	**rand_num**	32	The generated number $X_1, X_2, \ldots\ldots, X_{256}$. Should be output for 256 cycles, not required to be continuous.

Specifications

1. Top module name : PRGN_TOP (File name: PRGN_TOP.v)
2. Submodule name : CLK_1_MODULE, CLK_2_MODULE, CLK_3_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, PULSE_syn
   (File name: NDFF_syn.v, NDFF_syn.v, PULSE_syn.v in synchronizer folder)
5. Dual port SRAM : DUAL_64X32X1BM1 (04_MEM folder)
   (Words: 64, Bits: 32, Dual port SRAM)
6. Input pins : **clk1, clk2, clk3, rst_n, in_valid, [31:0] seed**
   Output pins : **out_valid, [31:0] rand_num**
7. Use **asynchronous** reset active low architecture.
8. Input data are synchronous to clk1, output data are synchronous to clk3.
9. 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.
10. The rand_num should be correct when out_valid is high.
11. The rand_num should be reset after your out_valid is pulled down.
12. **Changing clock period is prohibited.**
13. **Changing top module is prohibited.**
14. Some pre-defined flags are reserved for you to optimize your design.
15. **Calculate the result in CLK_2_MODULE, use other module to perform the Xorshift algorithm is prohibited.**
16. 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 clk3 period (4.1ns, 20.7ns, 67.3ns, 99.9ns) in the 01_RTL stage**.
17. **In the 02_SYN and 03_GATE stages, the clk3 period will be fixed at 20.7ns, and latency is calculated based on this.**
18. The PRGN_TOP.sdc is complete by TA. DO NOT modify it. This file is to set

the asynchronous clock groups of clk1, clk2, and clk3.

```
set_clock_groups -name group1 -asynchronous -group {clk1} -group {clk2} -group {clk3}
```

19. TA had generated dual port SRAM for the FIFO synchronizer, and the files are stored in 04_MEM. DO NOT modify them.
20. **You should use SRAM provided by TA to design your FIFO synchronizers** to maintain the fairness of area performance.
21. In the synchronizer you design, please use the NDFF_syn, NDFF_BUS_syn, PULSE_syn provided by TA if needed. Prime Time will ONLY **set_annotated_check** to the NDFF_syn module provided by TA.
22. After synthesis, check the "PRGN_TOP.area" and "PRGN_TOP.timing" in the folder "Report". The area report is valid only when the slack in the end of "PRGN_TOP.timing" is **non-negative**.
23. After run Prime Time, the slack in the end of "PRGN_TOP_pt.timing" should be also non-negative.
24. The synthesis result **cannot** contain any **latch**(in syn.log).
25. The output loading is set to 0.05.
26. Output delay is 0.5 * clk3 Clock Period.
27. Input delay is 0.5 * clk1 Clock Period.
28. You can't have timing violation in gate-level simulation.
29. **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.**
30. Your latency should be smaller than 2000 cycles in clk3.
31. The output should be raised for 256 cycles and is not required to be continuous.
32. The next input pattern will come in 1~3 clk1 cycles after getting 256 output data.

Grading Policy

- Function correct 70% (01_RTL to 03_GATE, 4 different clk3 cycle times in 01_RTL)
- Jasper Gold correct 25%
- Performance: Latency * Area^2 5% (Latency is calculated in clk3)
- The grade of 2nd demo would be 30% off.
- 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_clk3>)**
      The default of CYCLE_TIME_clk3 is 20.7ns.

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/ PRGN_TOP_pt.timing)
3. 03_GATE_SIM/ (GL simulation)
   - I. **./01_run_vcs_gate**
     (We will only run 20.7ns for clk3 period in GL 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: 2023/11/13 (Mon.) 12:00:00
   2nd_demo deadline: 2023/11/15 (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 name them as DESIGN_module_iclabxx.v, Handshake_syn _iclabxx.v and FIFO_syn_iclabxx.v.**

Waveform Example