National Cheng Kung University

Department of Electrical Engineering

Introduction to VLSI CAD (Spring 2024)

Lab Session 4

Register Files, Manhattan Distance and LFSR

Name	Student ID	
劉冠妤	E24116152	
Practical Sections	**Points**	**Marks**
Prob A	30	
Prob B	30	
Prob C	20	
Report	15	
File hierarchy, naming…etc.	5	
Notes:		

Due Date: 15:00, March 27, 2024 @ moodle

Deliverables

1) All Verilog codes including testbenches for each problem should be uploaded.
   NOTE: Please **DO NOT** include source code in the paper report!
2) All homework requirements should be uploaded in this file hierarchy or you will not get the full credit.
   NOTE: Please **DO NOT** upload waveforms!
3) Important! TA will use the command in Appendix A to check your design under SoC Lab environment, if your code can not be recompiled by TA successfully using the commands, you will not get the full credit.
4) If you upload a dead body which we can't even compile you will get **NO** credit!
5) All Verilog file should get at least 90% superLint Coverage.
6) File hierarchy should not be changed; it may cause your code can not be recompiled by TA successfully using the autograding commands

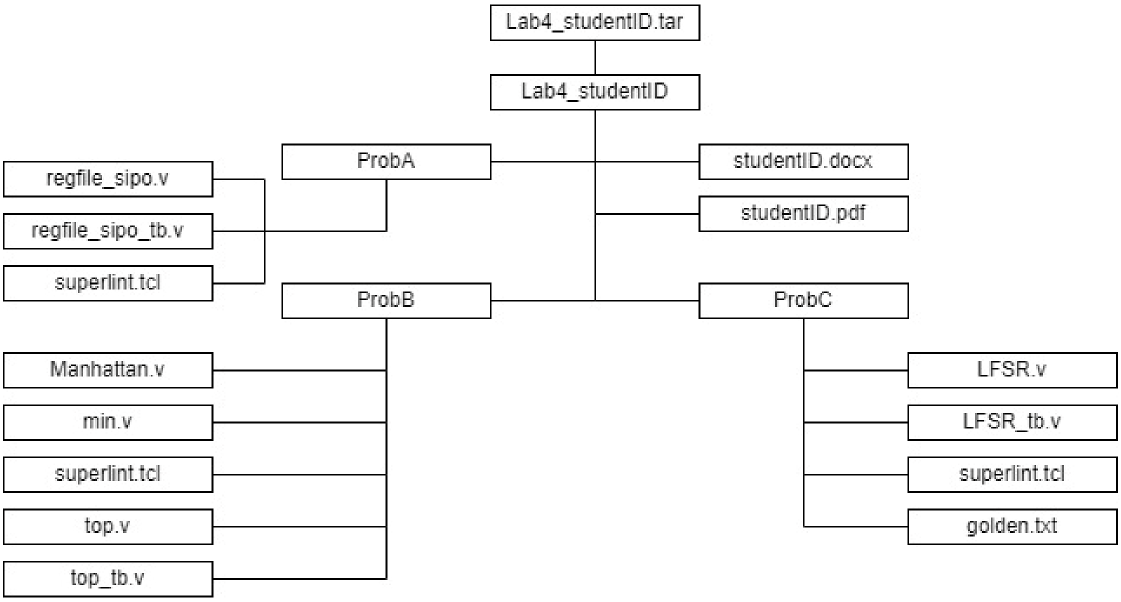
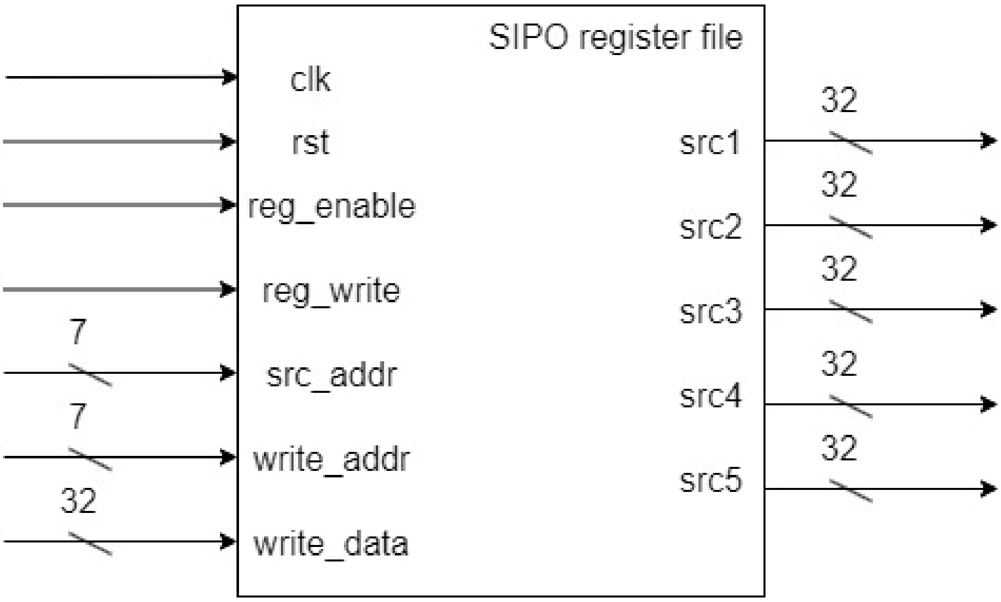


Fig.1 File hierarchy for Homework submission

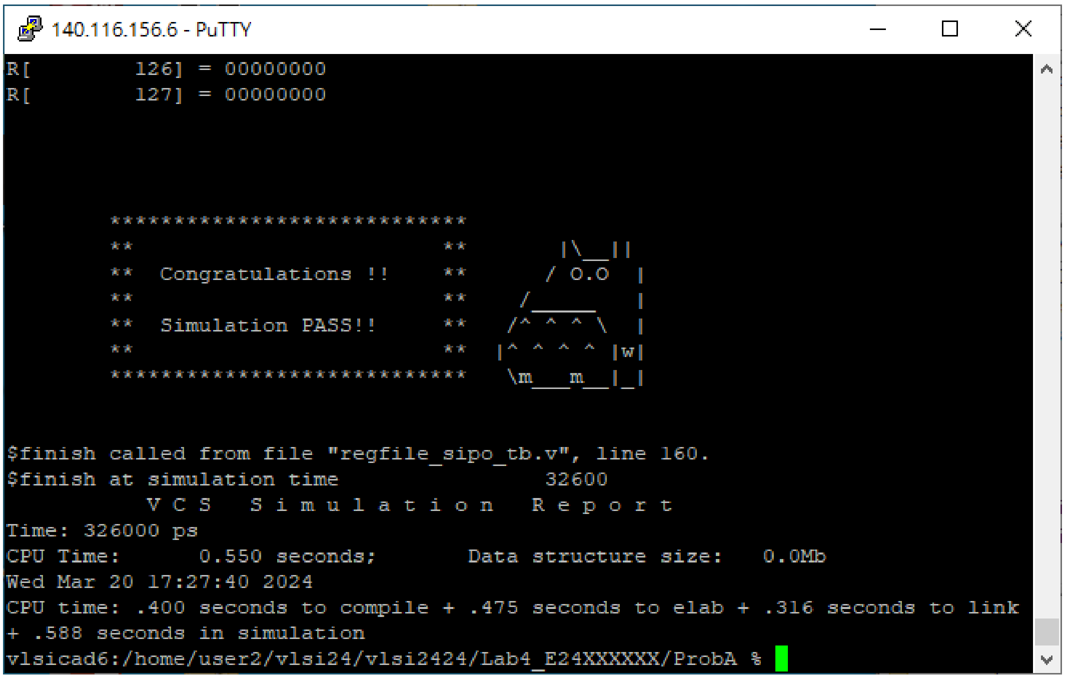
Prob A: SIPO Register File



1. Based on the SIPO register file structure in LabA, please design a 128 x 32 SIPO register file with 5 output ports.
2. Port list

Signal	Type	Bits	Description
clk	input	1	clock
rst	input	1	reset
reg_enable	input	1	register file enable
reg_write	input	1	0 → read 1 → write
src_addr	input	7	source address
write_addr	input	7	write address
write_data	input	32	write data
src1	output	32	read data source1
src2	output	32	read data source2
src3	output	32	read data source3
src4	output	32	read data source4
src5	output	32	read data source5

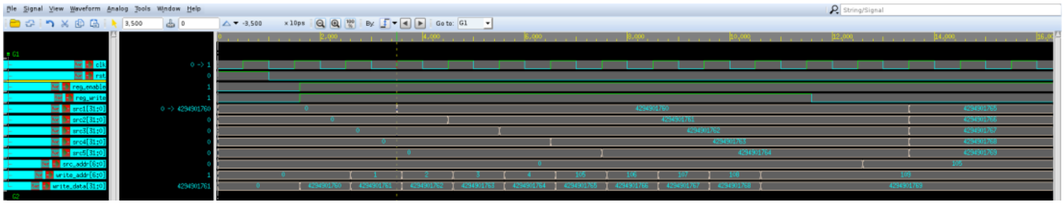
3. Show the simulation result on the terminal.



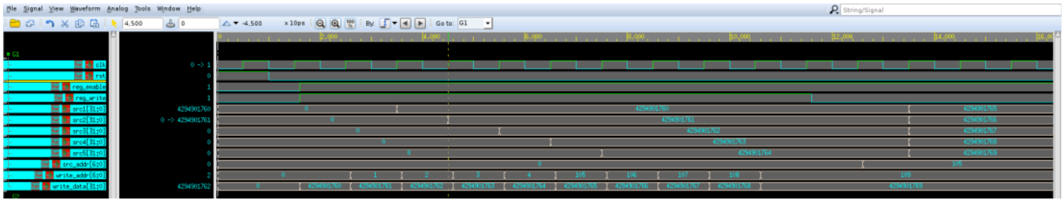
```
R[         126] = 00000000
R[         127] = 00000000



        ****************************
        **                        **       |\__||
        **  Congratulations !!    **      / O.O  |
        **                        **    /_____   |
        **  Simulation PASS!!     **   /^ ^ ^ \  |
        **                        **  |^ ^ ^ ^ |w|
        ****************************   \m___m__|_|

$finish called from file "regfile_sipo_tb.v", line 160.
$finish at simulation time                32600
           V C S   S i m u l a t i o n   R e p o r t
Time: 326000 ps
CPU Time:      0.550 seconds;       Data structure size:   0.0Mb
Wed Mar 20 17:27:40 2024
CPU time: .400 seconds to compile + .475 seconds to elab + .316 seconds to link
+ .588 seconds in simulation
vlsicad6:/home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbA % 
```

4. Show waveforms to explain that your register work correctly when read and write.

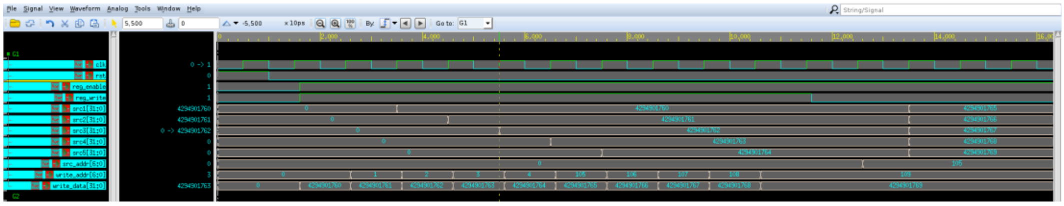
When the pulse is triggered on the positive edge, and the register file is enabled and in read mode, src_addr = 1 → write_data is written into src1



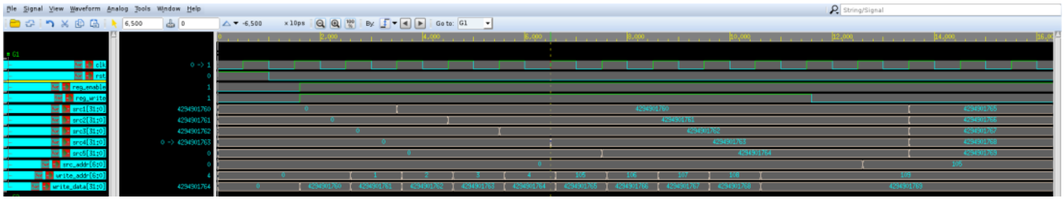
When the pulse is triggered on the positive edge, and the register file is enabled and in read mode, src_addr = 2 → write_data is written into src2



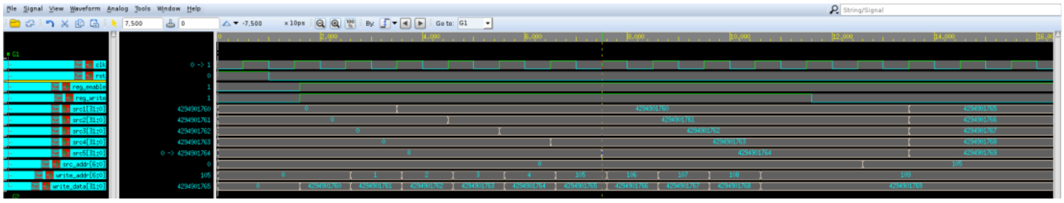
When the pulse is triggered on the positive edge, and the register file is enabled and in read mode, src_addr = 3 → write_data is written into src3



When the pulse is triggered on the positive edge, and the register file is enabled and in read mode, src_addr = 4 → write_data is written into src4

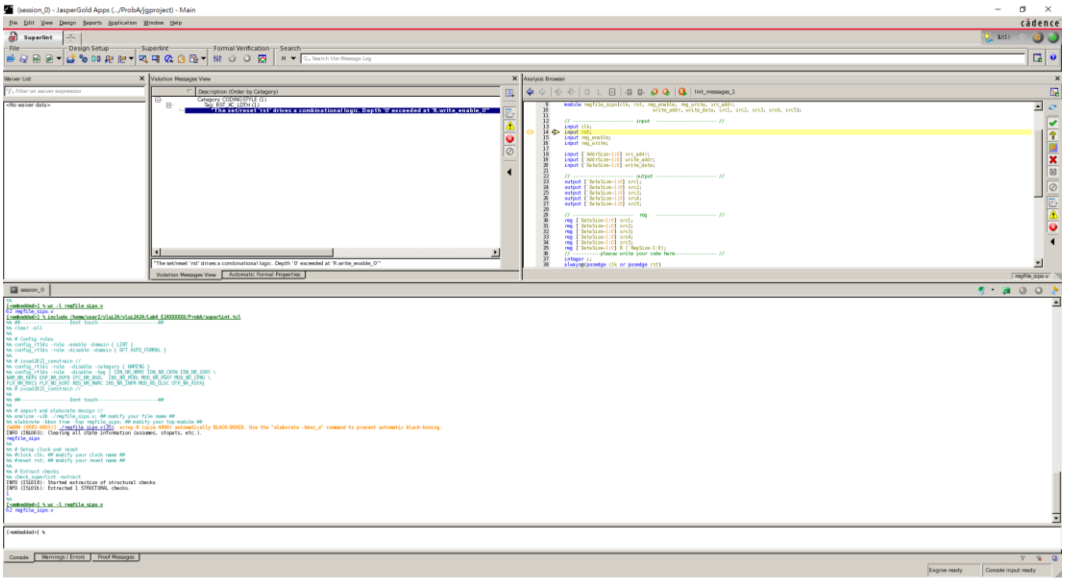


When the pulse is triggered on the positive edge, and the register file is enabled and in read mode, src_addr = 5 → write_data is written into src5



We can understand the operation of Serial-In Parallel-Out Register File through waveform diagrams. The latter part of the waveform diagram is reading. When the clock signal rises along the positive edge, the read value is output.

5. Show SuperLint coverage



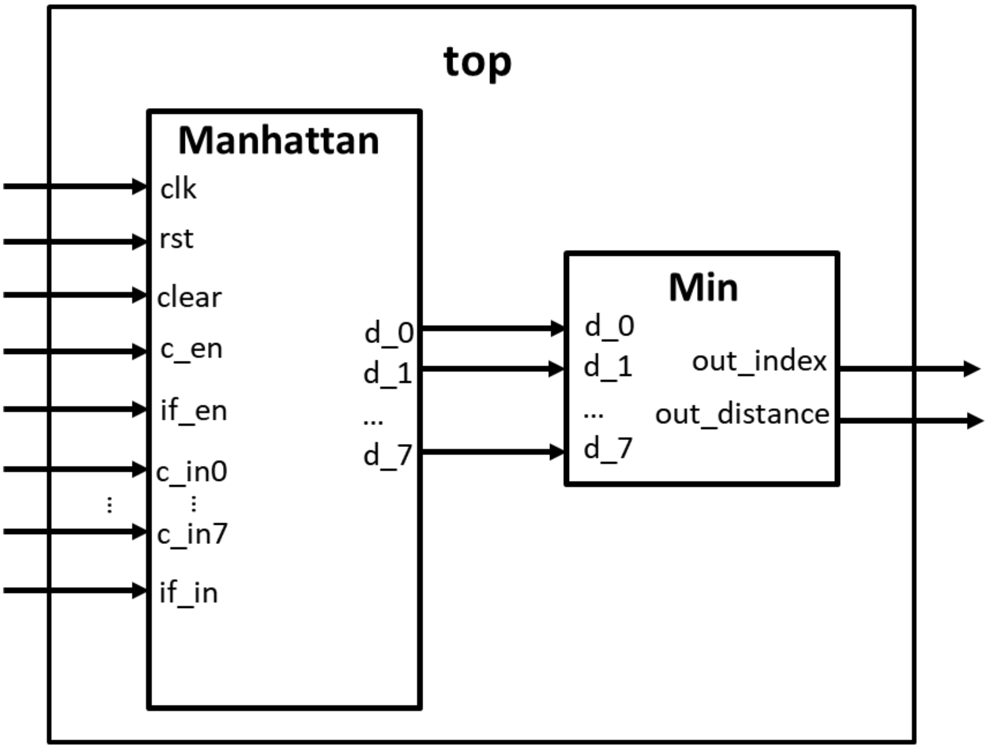
"The set/reset 'rst' drives a combinational logic. Depth '0' exceeded at 'R.write_enable_0'"

```
module regfile_sipo(clk, rst, reg_enable, reg_write, src_addr,
                    write_addr, write_data, src1, src2, src3, src4, src5);

// --------------------- input  --------------------- //
input clk;
input rst;
input reg_enable;
input reg_write;

input [`AddrSize-1:0] src_addr;
input [`AddrSize-1:0] write_addr;
input [`DataSize-1:0] write_data;

// --------------------- output --------------------- //
output [`DataSize-1:0] src1;
output [`DataSize-1:0] src2;
output [`DataSize-1:0] src3;
output [`DataSize-1:0] src4;
output [`DataSize-1:0] src5;

// ---------------------  reg   --------------------- //
reg [`DataSize-1:0] src1;
reg [`DataSize-1:0] src2;
reg [`DataSize-1:0] src3;
reg [`DataSize-1:0] src4;
reg [`DataSize-1:0] src5;
reg [`DataSize-1:0] R [`RegSize-1:0];
// ----------please write your code here-------------- //
integer i;
always@(posedge clk or posedge rst)
```

```
%%
[<embedded>] % wc -l regfile_sipo.v
62 regfile_sipo.v
[<embedded>] % include /home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbA/superlint.tcl
%% ##------------------Dont touch---------------------##
%% clear -all
%%
%% # Config rules
%% config_rtlds -rule -enable -domain { LINT }
%% config_rtlds -rule -disable -domain { DFT AUTO_FORMAL }
%%
%% # ivcad2021_constrain //
%% config_rtlds -rule  -disable -category { NAMING }
%% config_rtlds -rule  -disable -tag { IDN_NR_AMKY IDN_NR_CKYW IDN_NR_SVKY \
NAM_NR_REPU EXP_NR_OVFB IFC_NR_DGEL  INS_NR_PODL MOD_NR_PGAT MOD_NO_IPRG \
FLP_NR_MXCS FLP_NO_ASRT REG_NR_RWRC INS_NR_INPR MOD_NS_GLGC OTP_NR_ASYA}
%% # ivcad2021_constrain //
%%
%% ##------------------Dont touch---------------------##
%%
%% # import and elaborate design //
%% analyze -v2k ./regfile_sipo.v; ## modify your file name ##
%% elaborate -bbox true -top regfile_sipo; ## modify your top module ##
[WARN (VERI-9033)] ./regfile_sipo.v(35): array R (size 4096) automatically BLACK-BOXED. Use the "elaborate -bbox_a" command to prevent automatic black-boxing.
INFO (INL003): Clearing all state information (assumes, stopats, etc.).
regfile_sipo
%%
%% # Setup clock and reset
%% #clock clk; ## modify your clock name ##
%% #reset rst; ## modify your reset name ##
%%
%% # Extract checks
%% check_superlint -extract
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 1 STRUCTURAL checks.
1
%%
[<embedded>] % wc -l regfile_sipo.v
62 regfile_sipo.v
```

```
[<embedded>] %
```

Coverage = 98.38%



1. Please design a circuit that will find the smallest distance between the input feature and input colors, based on the structure given in the LAB4 slide.

2. Port list

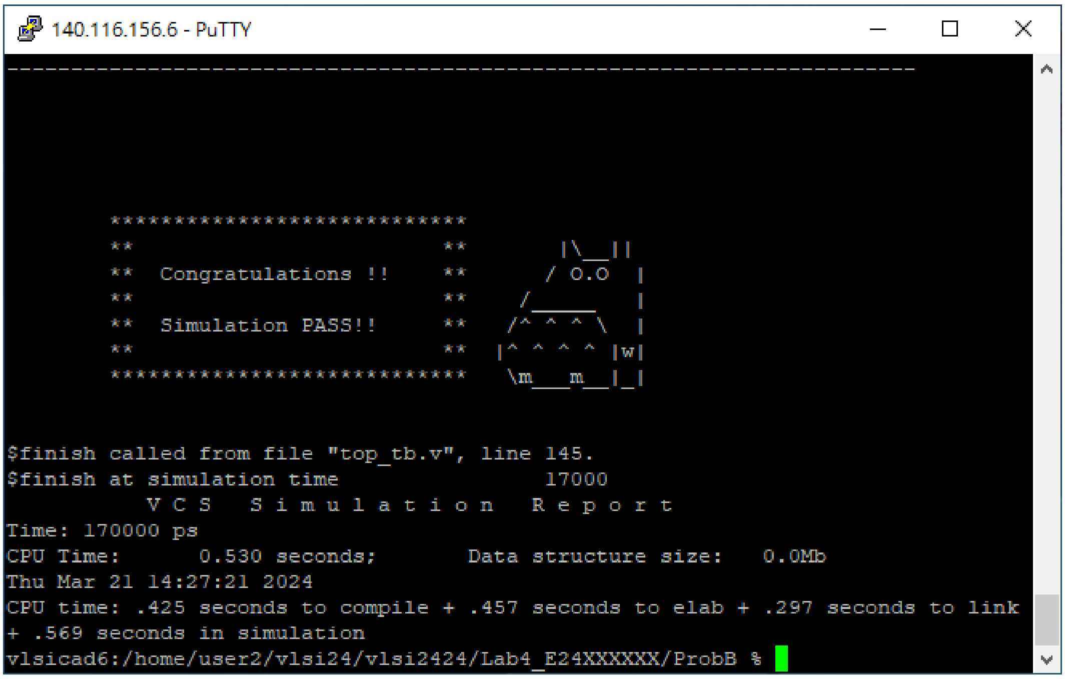
Manhattan:

Signal	Type	Bits	Description
clk	input	1	Clock pin.
rst	input	1	Reset pin.
clear	input	1	Set all registers to 0.
c_en	input	1	Write compared colors enable. When c_en is high, then c_in0~7 is available.
if_en	input	1	Write input pixel enable. When if_en is high, then if_in is available.
c_in0~7	input	24 each	Input color data.
if_in	input	24	Input input feature data.
d_0~7	output	10 each	Output distance data.

Min:

Signal	Type	Bits	Description
d_0~7	input	10 each	Input data.
out_index	output	3	Output index.
out_distance	output	10	Output minimum distance.

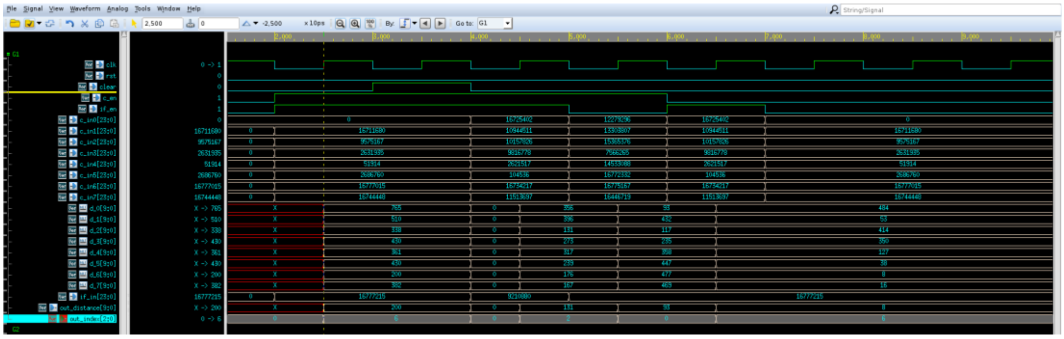
3. Show the simulation result on the terminal.



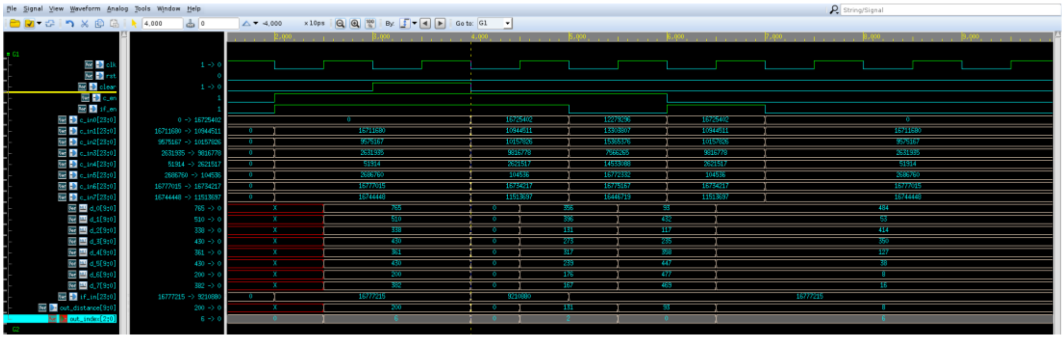
```
--------------------------------------------------------------------------



        ****************************
        **                        **       |\__||
        **  Congratulations !!    **      / O.O  |
        **                        **    /_____   |
        **  Simulation PASS!!     **   /^ ^ ^ \  |
        **                        **  |^ ^ ^ ^ |w|
        ****************************   \m___m__|_|


$finish called from file "top_tb.v", line 145.
$finish at simulation time                17000
           V C S   S i m u l a t i o n   R e p o r t
Time: 170000 ps
CPU Time:      0.530 seconds;       Data structure size:   0.0Mb
Thu Mar 21 14:27:21 2024
CPU time: .425 seconds to compile + .457 seconds to elab + .297 seconds to link
+ .569 seconds in simulation
vlsicad6:/home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbB %
```

4. Show waveforms to explain that your design works correctly.

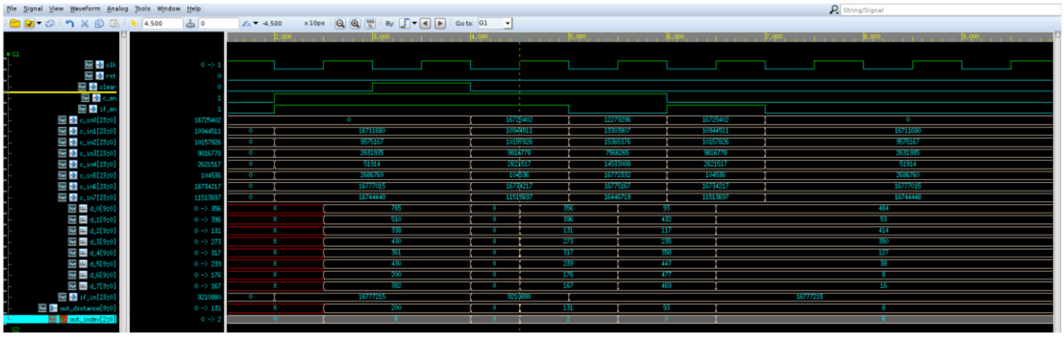
From the waveform diagram, we can see that after the data is read in first, the calculated minimum distance and indexs are output only when the positive edge is triggered.



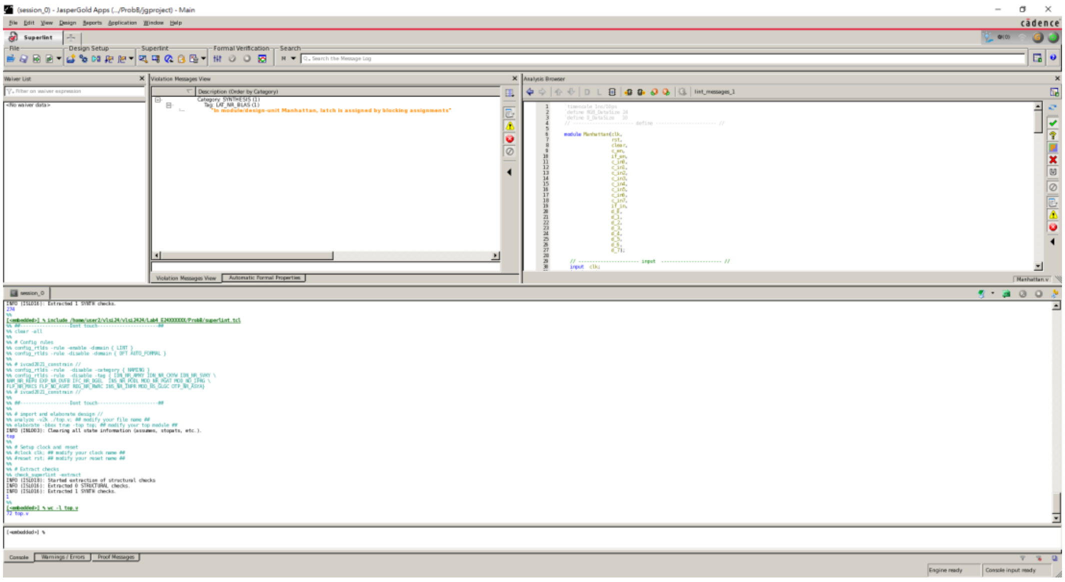
But when the clear signal is positive, it is not triggered with the positive edge of the clock, because this is my design.



Under normal circumstances, the changes are triggered by the positive edge of the clock signal.



5. Show SuperLint coverage



```
Category: SYNTHESIS (1)
  Tag: LAT_NR_BLAS (1)
    "In module/design-unit Manhattan, latch is assigned by blocking assignments"
```

```
INFO (ISL016): Extracted 1 SYNTH checks.
274
%%
[<embedded>] % include /home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbB/superlint.tcl
%% ##-------------------Dont touch----------------------##
%% clear -all
%%
%% # Config rules
%% config_rtlds -rule -enable -domain { LINT }
%% config_rtlds -rule -disable -domain { DFT AUTO_FORMAL }
%%
%% # ivcad2021_constrain //
%% config_rtlds -rule  -disable -category { NAMING }
%% config_rtlds -rule  -disable -tag { IDN_NR_AMKY IDN_NR_CKYW IDN_NR_SVKY \
NAM_NR_REPU EXP_NR_OVFB IFC_NR_DGEL  INS_NR_PODL MOD_NR_PGAT MOD_NO_IPRG \
FLP_NR_MXCS FLP_NO_ASRT REG_NR_RWRC INS_NR_INPR MOD_NS_GLGC OTP_NR_ASYA}
%% # ivcad2021_constrain //
%%
%% ##-------------------Dont touch----------------------##
%%
%% # import and elaborate design //
%% analyze -v2k ./top.v; ## modify your file name ##
%% elaborate -bbox true -top top; ## modify your top module ##
INFO (INL003): Clearing all state information (assumes, stopats, etc.).
top
%%
%% # Setup clock and reset
%% #clock clk; ## modify your clock name ##
%% #reset rst; ## modify your reset name ##
%%
%% # Extract checks
%% check_superlint -extract
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 0 STRUCTURAL checks.
INFO (ISL016): Extracted 1 SYNTH checks.
1
%%
[<embedded>] % wc -l top.v
72 top.v
```

```
[<embedded>] %
```

Coverage = 98.61%

Prob C: LFSR

1. Please design an 8-bit-LFSR, with the given feedback function in the LAB4 slide.

2. Port list

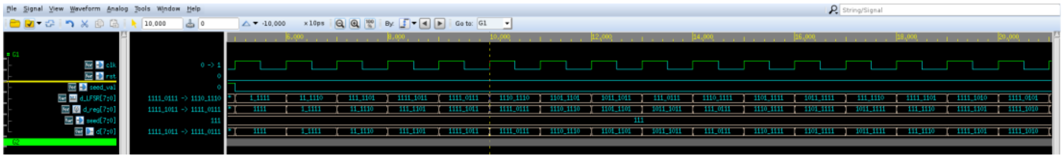
Signal	Type	Bits	Description
clk	input	1	Clock pin.
rst	input	1	Reset pin. Reset all of the flip flops to zeros.
seed_val	input	1	1: the flip flops take seed as the initial state. 0: the flip flops works as linear feedback shift register.
seed	input	8	Initial state value of LFSR.
d	output	8	Output value of LFSR

3. Feedback function

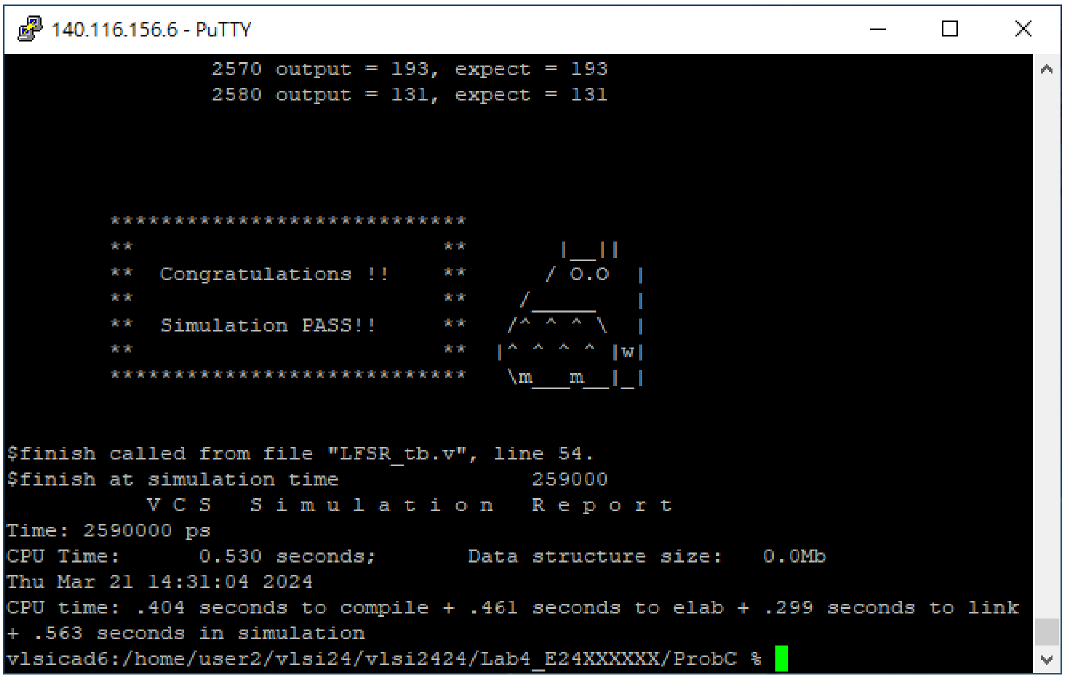
$$\boldsymbol{d[0] = (d[7] \wedge d[5]) \wedge (d[4] \wedge d[2])}$$

4. Show waveforms to explain that your LFSR module works correctly.

Take this wave pattern as an example:
d = 1111_1011, seed_val = 0
On the positive edge of the next clock signal, d shift left one bit, and d[0] is equal to (d[7]^d[5]) ^ (d[4] ^ d[2]) which is (1^1)^(1^0) = 1.
So the next d = 1111_0111, the waveform results are consistent with the expected results.

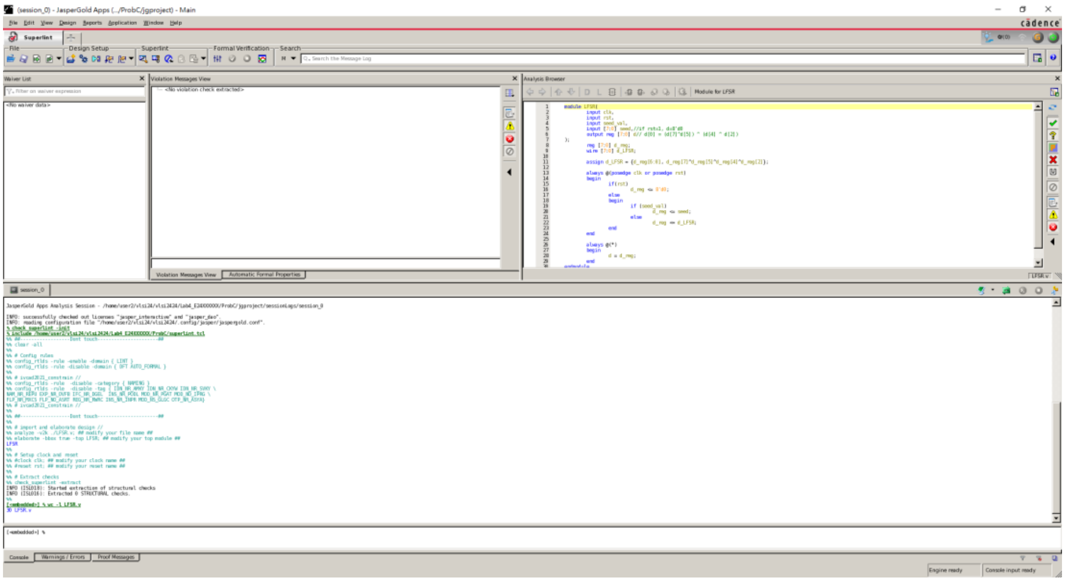


5. Show the simulation result on the terminal.



```
                 2570 output = 193, expect = 193
                 2580 output = 131, expect = 131




        ****************************
        **                        **        |__||
        **   Congratulations !!   **      / O.O  |
        **                        **    /_____   |
        **   Simulation PASS!!    **   /^ ^ ^ \  |
        **                        **  |^ ^ ^ ^ |w|
        ****************************   \m___m__|_|


$finish called from file "LFSR_tb.v", line 54.
$finish at simulation time               259000
           V C S   S i m u l a t i o n   R e p o r t
Time: 2590000 ps
CPU Time:      0.530 seconds;       Data structure size:   0.0Mb
Thu Mar 21 14:31:04 2024
CPU time: .404 seconds to compile + .461 seconds to elab + .299 seconds to link
 + .563 seconds in simulation
vlsicad6:/home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbC %
```

6. Show SuperLint coverage



```
module LFSR(
        input clk,
        input rst,
        input seed_val,
        input [7:0] seed,//if rst=1, d=8'd0
        output reg [7:0] d// d[0] = (d[7]^d[5]) ^ (d[4] ^ d[2])
);
        reg [7:0] d_reg;
        wire [7:0] d_LFSR;

        assign d_LFSR = {d_reg[6:0], d_reg[7]^d_reg[5]^d_reg[4]^d_reg[2]};

        always @(posedge clk or posedge rst)
        begin
                if(rst)
                        d_reg <= 8'd0;
                else
                begin
                        if (seed_val)
                                d_reg <= seed;
                        else
                                d_reg <= d_LFSR;
                end
        end

        always @(*)
        begin
                d = d_reg;
        end
endmodule
```

```
JasperGold Apps Analysis Session - /home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbC/jgproject/sessionLogs/session_0

INFO: successfully checked out licenses "jasper_interactive" and "jasper_dao".
INFO: reading configuration file "/home/user2/vlsi24/vlsi2424/.config/jasper/jaspergold.conf".
% check_superlint -init
% include /home/user2/vlsi24/vlsi2424/Lab4_E24XXXXXX/ProbC/superlint.tcl
%% ##-----------------Dont touch---------------------##
%% clear -all
%%
%% # Config rules
%% config_rtlds -rule -enable -domain { LINT }
%% config_rtlds -rule -disable -domain { DFT AUTO_FORMAL }
%%
%% # ivcad2021_constrain //
%% config_rtlds -rule  -disable -category { NAMING }
%% config_rtlds -rule  -disable -tag { IDN_NR_AMKY IDN_NR_CKYW IDN_NR_SVKY \
NAM_NR_REPU EXP_NR_OVFB IFC_NR_DGEL  INS_NR_PODL MOD_NR_PGAT MOD_NO_IPRG \
FLP_NR_MXCS FLP_NO_ASNT REG_NR_MWMC INS_NR_INPR MOD_NS_GLGC OTP_NR_ASYA}
%% # ivcad2021_constrain //
%%
%% ##-----------------Dont touch---------------------##
%%
%% # import and elaborate design //
%% analyze -v2k ./LFSR.v; ## modify your file name ##
%% elaborate -bbox true -top LFSR; ## modify your top module ##
LFSR
%%
%% # Setup clock and reset
%% #clock clk; ## modify your clock name ##
%% #reset rst; ## modify your reset name ##
%%
%% # Extract checks
%% check_superlint -extract
INFO (ISL018): Started extraction of structural checks
INFO (ISL016): Extracted 0 STRUCTURAL checks.
%%
[<embedded>] % wc -l LFSR.v
30 LFSR.v

[<embedded>] %
```

Coverage = 100%

At last, please write the lesson you learned from Lab4

In this experiment, I learned what Serial-In Parallel-Out Register is. Although this is something I have never been exposed to before, the lecture notes are very clear and easy to understand. In addition, I also learned about Manhattan distance and applied it in practice. I found that the concepts of blocking and nonblocking in Verilog are very important, because I made many mistakes in the experiment because of unclear concepts. Finally, I learned the principle and practical application of Linear feedback shift register, and I found that it was not as difficult as I thought. After these implementations, I am more familiar with Verilog's syntax and design concepts.

Appendix A : Commands we will use to check your homework

Problem		Command
Prob A	Compile	% vcs -R regfile_sipo.v -full64
	Simulate	% vcs -R regfile_sipo_tb.v -debug_access+all -full64 +define+FSDB
Prob B	Compile	% vcs -R top.v -full64
	Simulate	% vcs -R top_tb.v -debug_access+all -full64 +define+FSDB
Prob C	Compile	% vcs -R LFSR.v -full64
	Simulate	% vcs -R LFSR_tb.v -debug_access+all -full64 +define+FSDB