National Cheng Kung University

Department of Electrical Engineering

Introduction to VLSI CAD (Spring 2022)

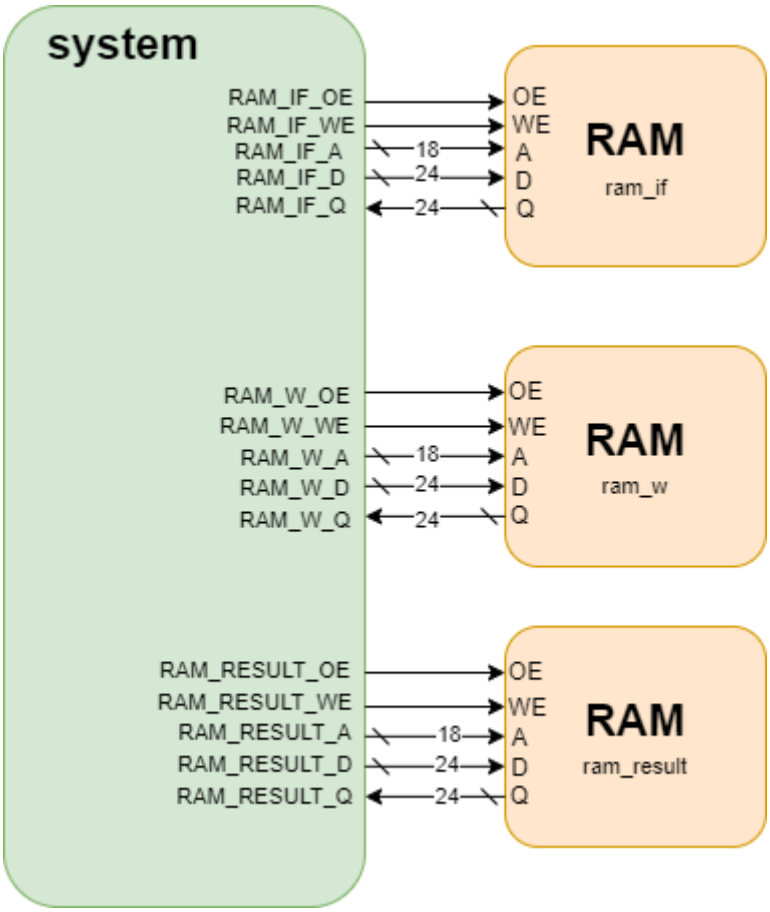
Lab Session 7

SOM Processing System

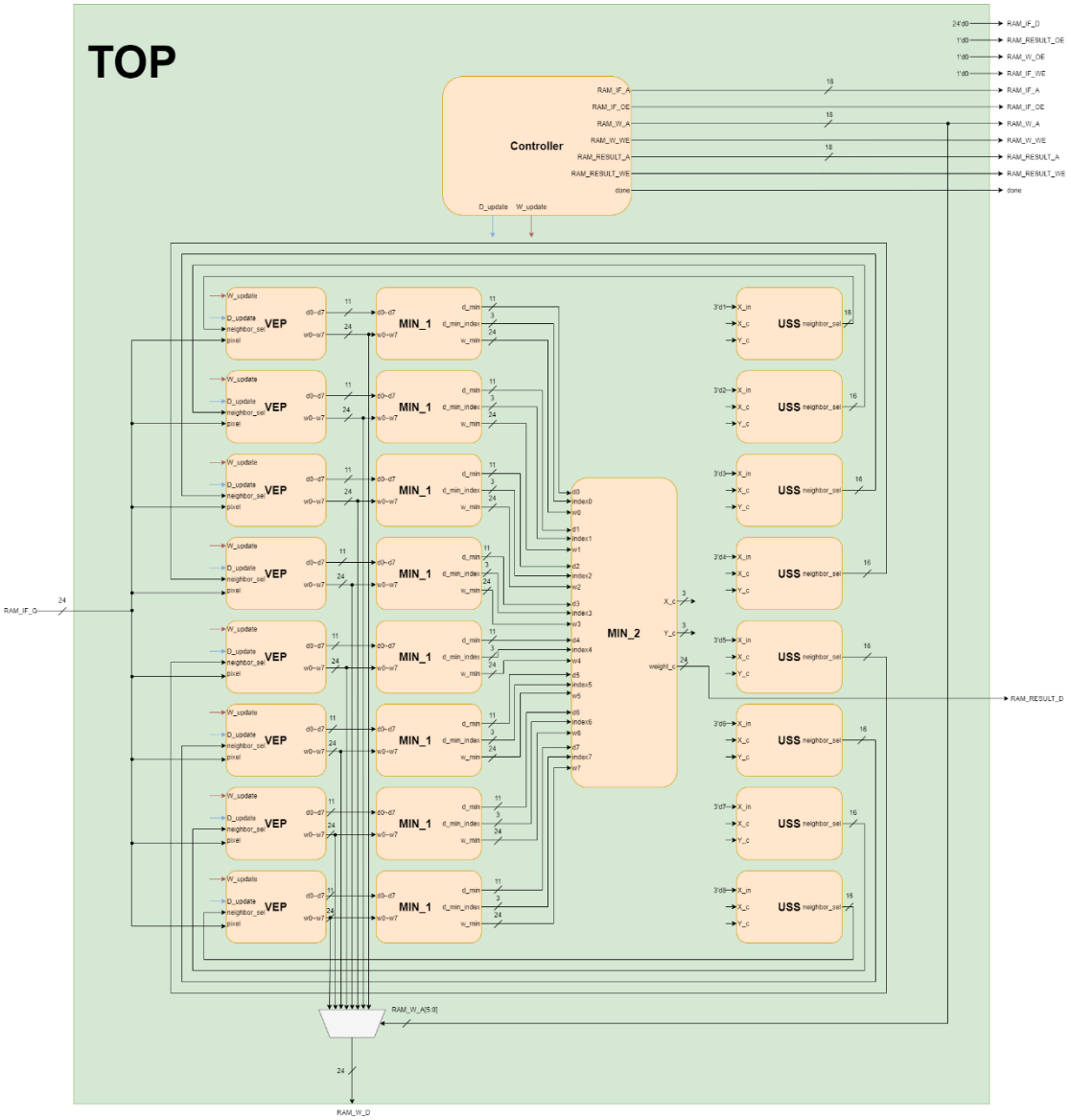
Yu-Chi Chu

Lab 7_1

You are about to integrate all components (VEP, MIN_1, MIN_2, USS, Controller…) to form a SOM processing system. The block diagram of system is as shown in **Fig2** and **Fig3**. (Clock pin and reset pin is ignored in the graph, but you should implement it)



▲Fig2. The block diagram of system (external)



▲Fig3. The block diagram of system (internal)

- **Port list of each module:**
- Controller

Signal	I/O	bit	Description
clk	Input	1	Clock
rst	Input	1	Reset signal, active high
D_update	Output	1	Distance update enable, active high
W_update	Output	1	Weight update enable, active high
RAM_IF_A	Output	18	Input feature RAM address
RAM_IF_OE	Output	1	Input feature RAM output enable
RAM_W_A	Output	18	Weight RAM address
RAM_W_WE	Output	1	Weight RAM write enable
RAM_RESULT_A	Output	18	Result RAM address
RAM_RESULT_WE	Output	1	Result RAM write enable
done	Output	1	Pull to 1 if the system is done

- **VEP**

Signal	I/O	Bit	Description
clk	Input	1	Clock
rst	Input	1	Reset signal, active high
W_update	Input	1	Weight update enable, active high
D_update	Input	1	Distance update enable, active high
neighbor_sel	Input	16(2x8)	Neighborhood function of 8 VEP weights (00=>1, 01=>0.25, 10=>0.125, 11=>0)
pixel	Input	24	Input pixel from RAM_if
d0~d7	Output	11	Manhattan distance between 8 weights
w0~w7	Output	24	8 weights

- **MIN_1**

Signal	I/O	bit	Description
clk	Input	1	Clock
rst	Input	1	Reset signal, active high
d0~d7	Input	11	Manhattan distance between 8 weights
w0~w7	Input	24	8 weights
d_min	Output	11	Minimum distance between d0~d7
d_min_index	Output	3	Index of minimum distance
W_min	output	24	Weight of minimum distance

- **MIN_2**

signal	I/O	bit	Description
clk	Input	1	clock
rst	Input	1	Reset signal, active high
d0~d7	Input	11	Minimum distance from MIN_1
w0~w7	Input	24	Weight of minimum distance from MIN_1
index0~index7	Input	3	Index of minimum distance from MIN_1
X_c	Output	3	The X coordinate of center weight
Y_c	Output	3	The Y coordinate of center weight
weight_c	output	24	Center weight

- **USS**

signal	I/O	bit	Description
clk	input	1	Clock
rst	Input	1	Reset signal, active high
X_in	Input	3	USS module index
X_c	Input	3	The X coordinate of center weight
Y_c	Input	3	The Y coordinate of center weight
neighbor_sel	Output	16(2x8)	Neighborhood function of 8 VEP weights (00=>1, 01=>0.25, 10=>0.125, 11=>0)

- Top

Signal	I/O	bit	Description
clk	Input	1	Clock
rst	Input	1	Reset signal, active high
RAM_IF_Q RAM_W_Q RAM_RESULT_Q	Input	24	Data output from RAM
RAM_IF_OE RAM_W_OE RAM_RESULT_OE	Output	1	RAM output enable signal
RAM_IF_WE RAM_W_WE RAM_RESULT_WE	Output	1	RAM write enable signal
RAM_IF_A RAM_W_A RAM_RESULT_A	Output	18	RAM address
RAM_IF_D RAM_W_D RAM_RESULT_D	output	24	Data written into RAM
done	Output	1	Pull to 1 if the system is done

- Understanding the function:

  Once system is initialized, it

  a) Read input pixel from RAM_if
  b) Calculate Manhattan distance and fin the minimum distance
  c) Update the weight memory
  d) Repeats the process step(a)~(c)until the last pixel of RAM_if is read
  e) writes the trained codebook to the RAM_w
  f) read input pixel from RAM_if and inference the picture
  g) writes the lossy compression picture to the RAM_result
  h) repeats the process step (f)~(g) until the last pixel of RAM_result is writed;
  i) flags “done” when system is completed

- Describe your design in detail. You can draw internal architecture or block diagram to describe your dsign.
  - VEP

VEP

sel[1:0] ⟶ 2'b00 ⟶ t_W0 = W0 + MAN(Pixel, W0) >> 2
2'b01 ⟶ t_W0 = W0 + MAN(Pixel, W0) >> 4
2'b10 ⟶ t_W0 = W0 + MAN(Pixel, W0) >> 3
2'b11 ⟶ t_W0 = W0

sel[15:14] ⟶ ⋮

MAN(Pixel, W0) >> 2

Pixel[7:0] ≥ W0[7:0] ? Pixel[7:0] - W0[7:0] : W0[7:0] - Pixel[7:0]

Pixel[15:8] ≥ W0[15:8] ? ..

Pixel[23:16] ≥ W0[23:16] ? ..

分別存入 temp_0 , temp_1 , temp_2

temp_b = temp_0 >> 2 ⇒ temp_B = temp_b + W0[7:0]

temp_g = temp_1 >> 2 ⇒ temp_G = temp_g + W0[15:8]

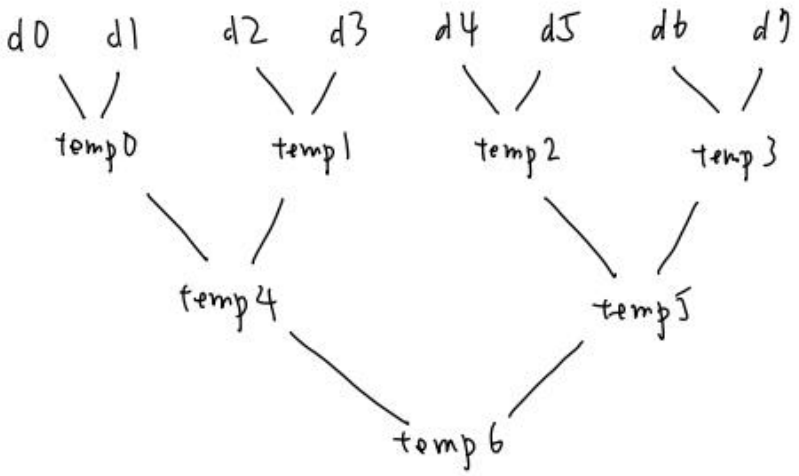
temp_r = temp_2 >> 2 ⇒ temp_R = temp_r + W0[23:16]

⇒ t_W0 = {tempR, temp G, temp B} 完成運算

後面以此類推

■ MIN_1

MIN_1



if temp6 = d7 (∵相等要取最後的)

{ d_min = d7
d_min_index = 7
w_min = w7

else if temp6 = d6

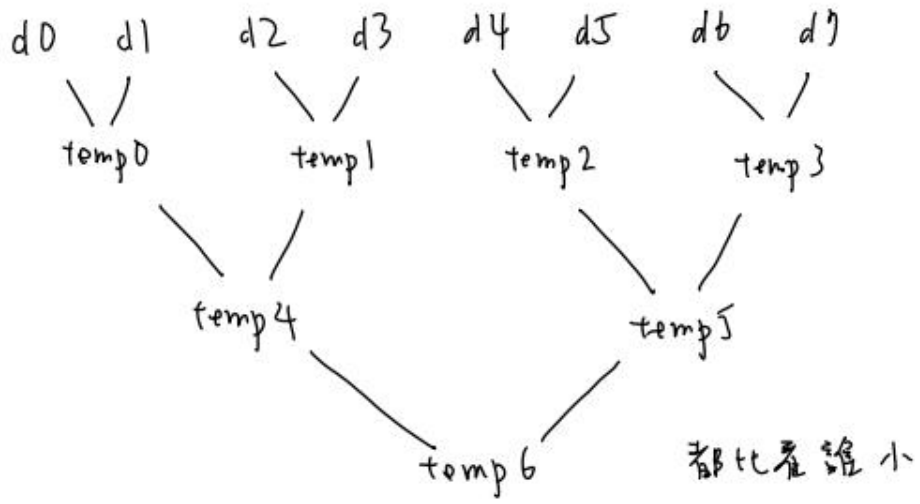
{ d_min = d6
d_min_index = 6
w_min = w6

⋮

以此類推

■ MIN_2

MIN_2



if temp6 = d7 (∵相等要取最後的)

{ X_c = 7
Y_c = index_7
weight_c = w7

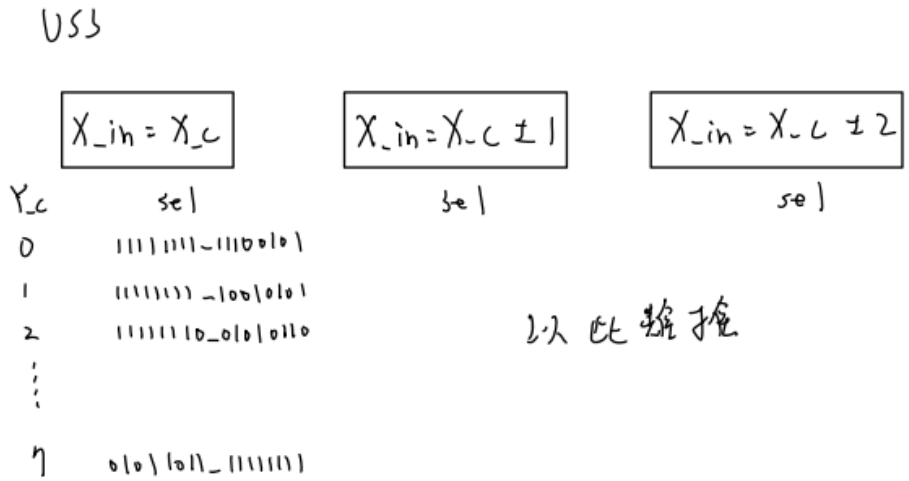
if temp6 = d6

{ X_c = 6
Y_c = index_6
weight_c = w6

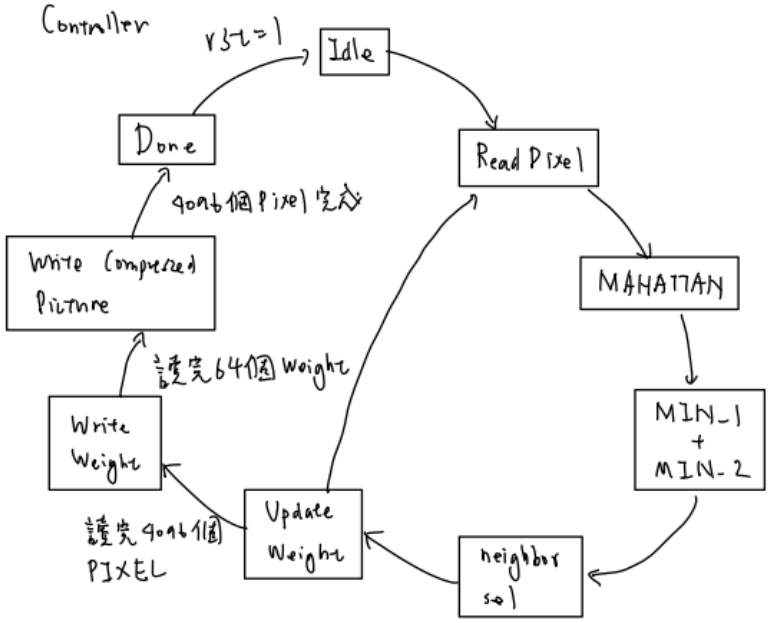
⋮

以此類推

- USS



- Controller
  - Draw your state diagram in controller and explain it



1) Complete the Controller, VEP, MIN_1, MIN_2, USS, and TOP module, in the system.

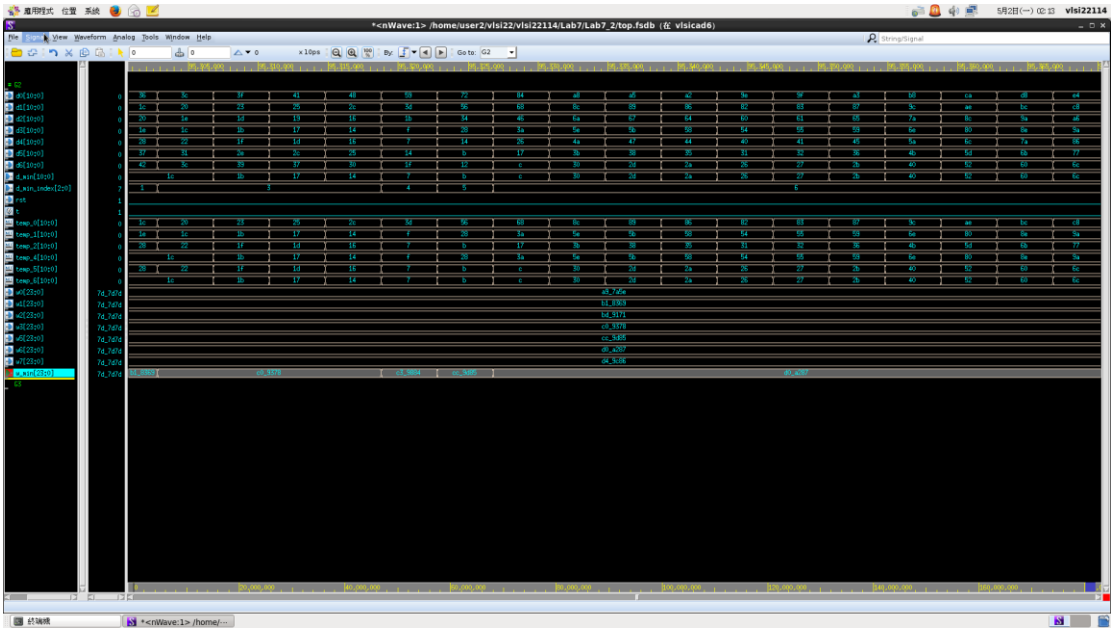
2) Compile the verilog code to verify the operations of this module works properly.

3) Synthesize your *TOP.v* with following constraint:

- Clock period: no more than 20 ns.
- Don't touch network: clk.
- Wire load model: saed14rvt_ss0p72v125c.
- Synthesized verilog file: *top_syn.v*.
- Timing constraint file: *top_syn.sdf*.

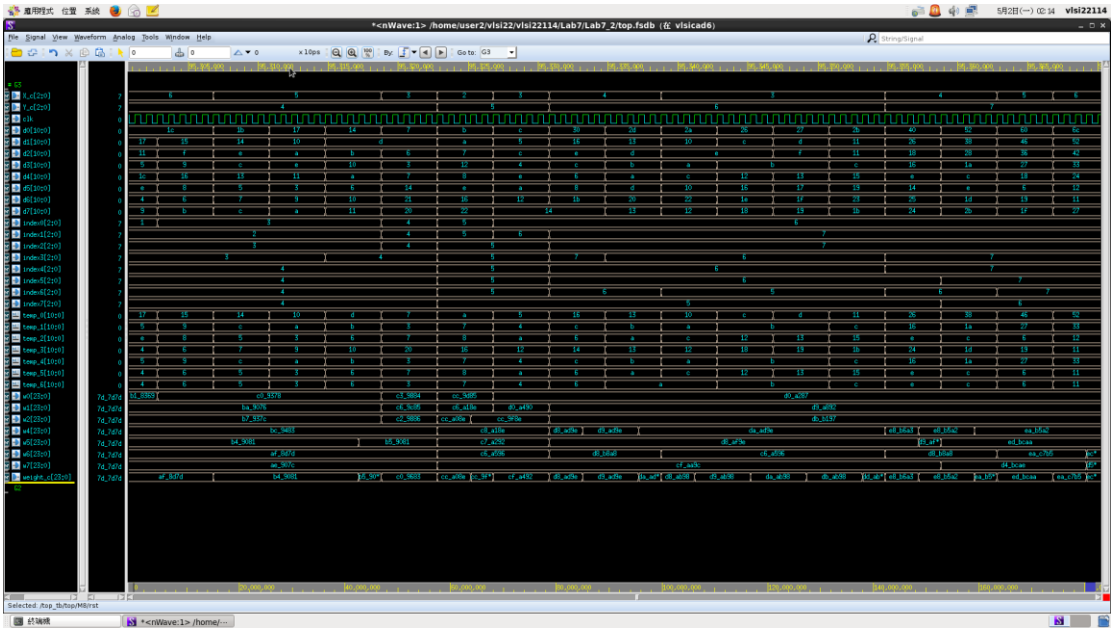
4) Please **attach your waveforms** and **specify your operations** on the waveforms.

MIN_1



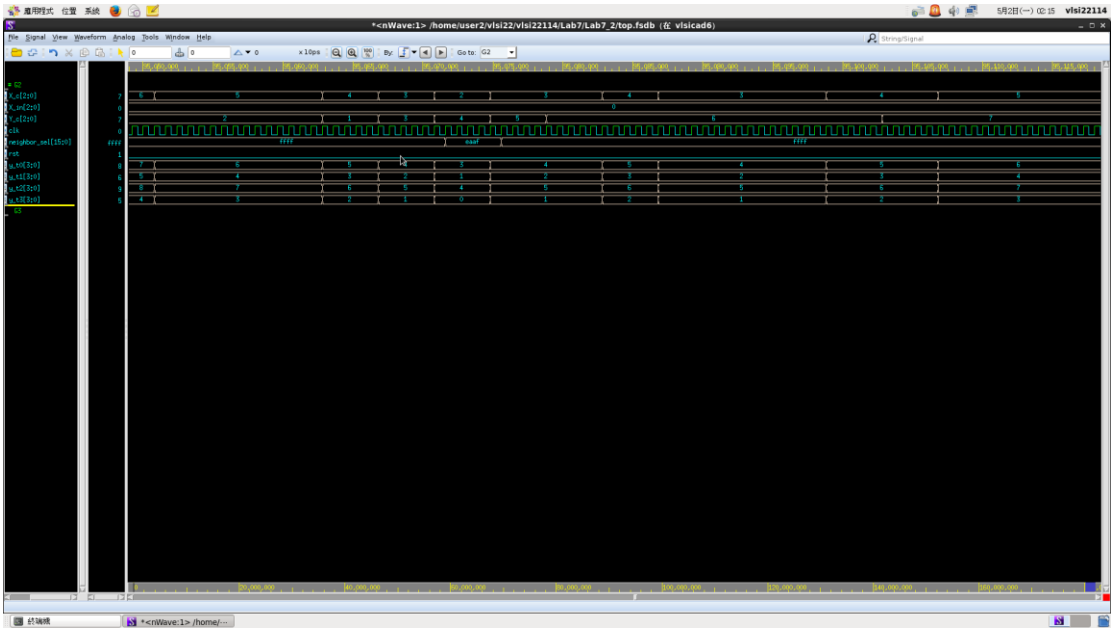
Temp0 is used to store the smaller of d0 and d1, Temp1 is used to store the smaller of d2 and d3, Temp2 is used to store the smaller of d4 and d5, and Temp3 is used to store the smaller of d6 and d7. Therefore, it can be seen that Temp0 is 1c when the first data comes in, Temp1 is 1e, Temp2 is 28, and Temp3 is 28. Temp4 stores the smaller of Temp0 and Temp1, which is 1c at this time. Temp5 stores the smaller of Temp2 and Temp3, which is 28 at this time. Finally, Temp6 is used to store the smaller of Temp4 and Temp5. Temp6 is 1c at this time. Finally, check whose distance is equal to Temp6. For details, please refer to the hierarchy diagram.

MIN_2



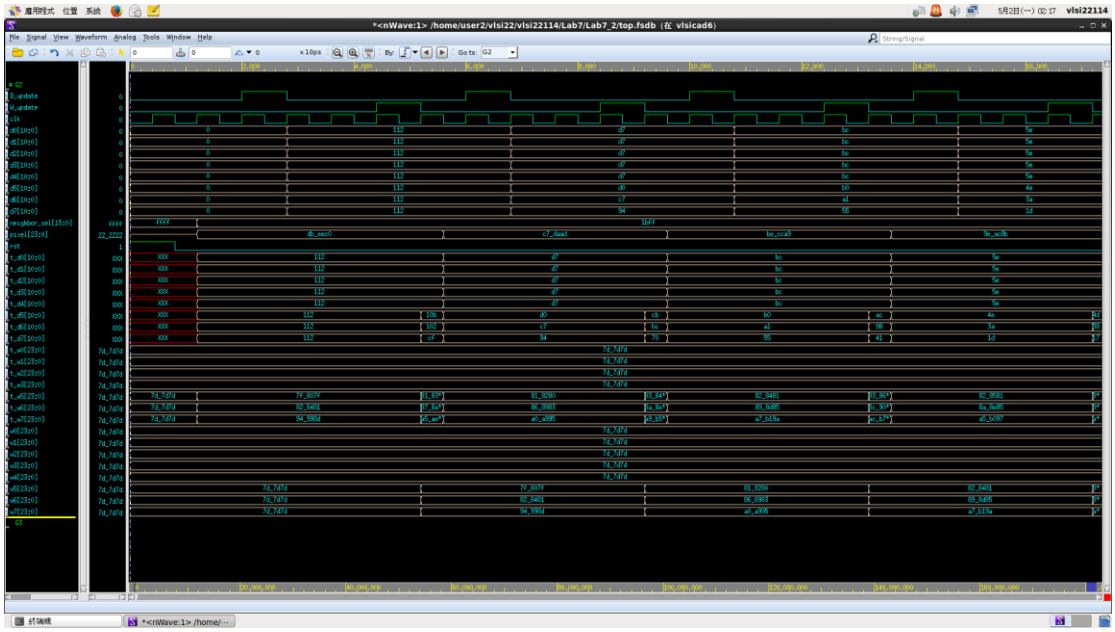
Same as min1, the only difference is that the outputs are X_c, Y_c, and weight_c. If the smallest value is d6, then X_c at this time is 6, X_y is index6, and weight_c is w6.

USS



y_t0 and y_t1 store the positive and negative 1 of X_c respectively, and y_t2 and y_t3 store the positive and negative 2 of X_c respectively. Then check which X_in is equal to and determine which case I should enter. It can be seen that the first data X_c is 6, so y_t0 is 7, y_t1 is 5, y_t2 is 8, and y_t3 is 4. Then, according to my Y_c, write neighbor_sel.

VEP

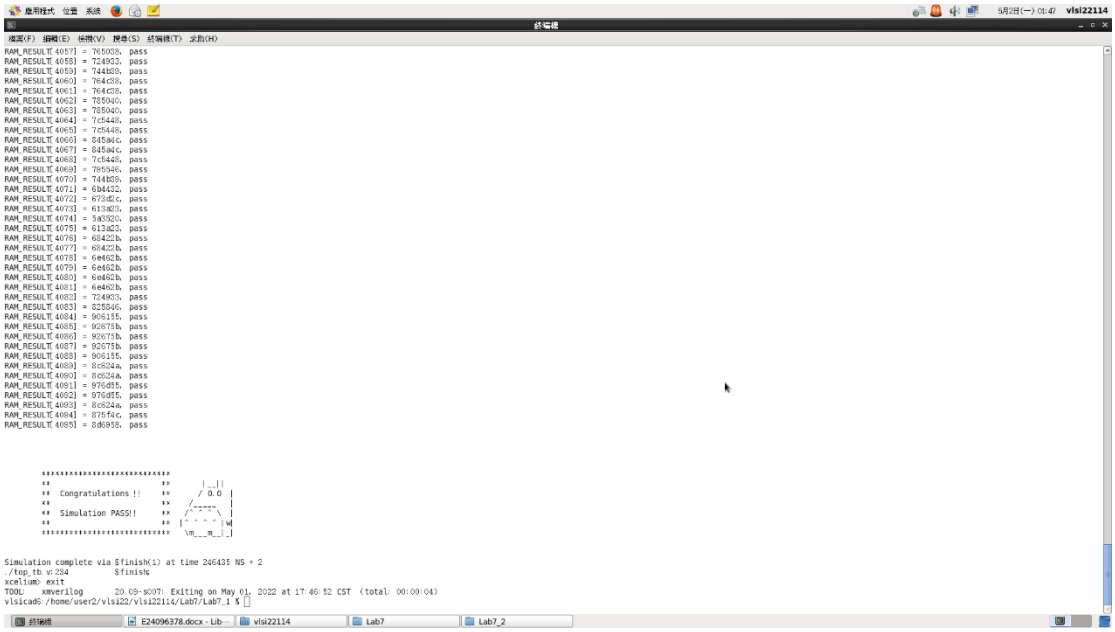


It mainly uses t_w and t_d to store the updated weight and distance at this time. Then, when W_update and D_update are respectively equal to 1, the updated values are output to w and d. For details, please refer to the hierarchy diagram above.

Controller

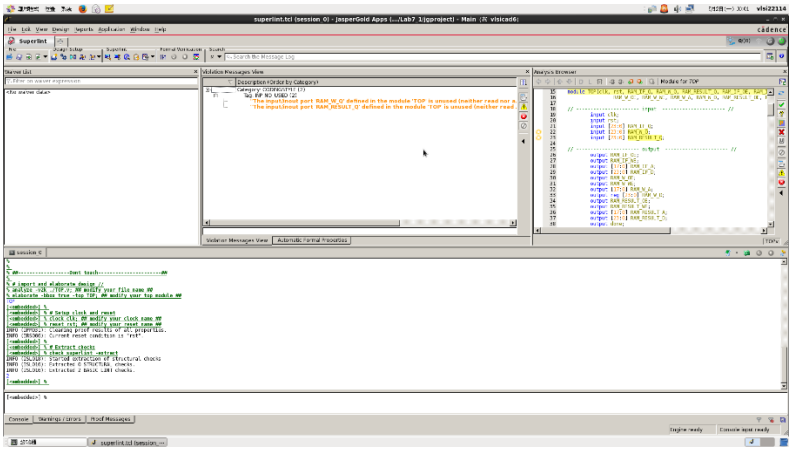
Use a counter to calculate the clk I have passed and calculate the memory location. The detailed FSM is drawn above.

5) Show simulation result



Coverage = 0.99

6) Show SuperLint coverage (TOP.v)



Coverage = 0.99

7) Your clock period, total cell area, post simulation time (TOP.v)

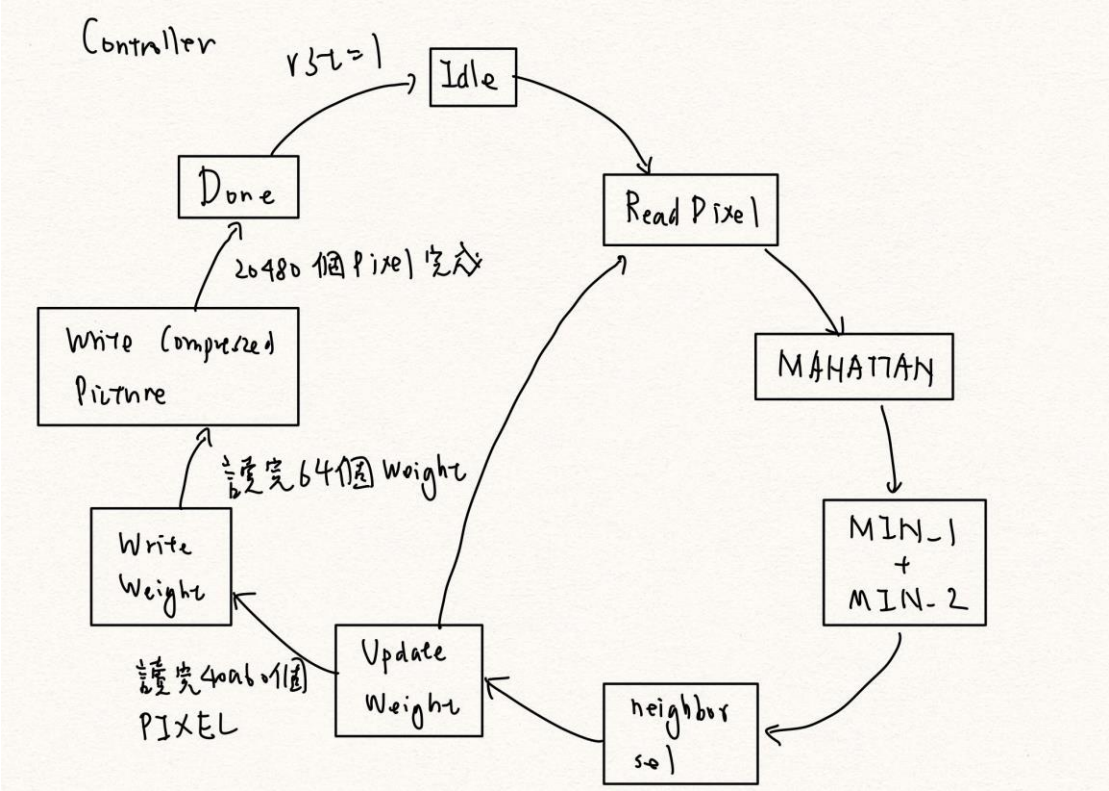
Clock period:20ns

Total cell area:25628.922838

Post simulation time:246435

Lab 7_2

- Draw your state diagram and explain your design. You can draw internal architecture to describe your design



8) Complete the Controller, VEP ,MIN_1, MIN_2, USS, and TOP module, in the system.

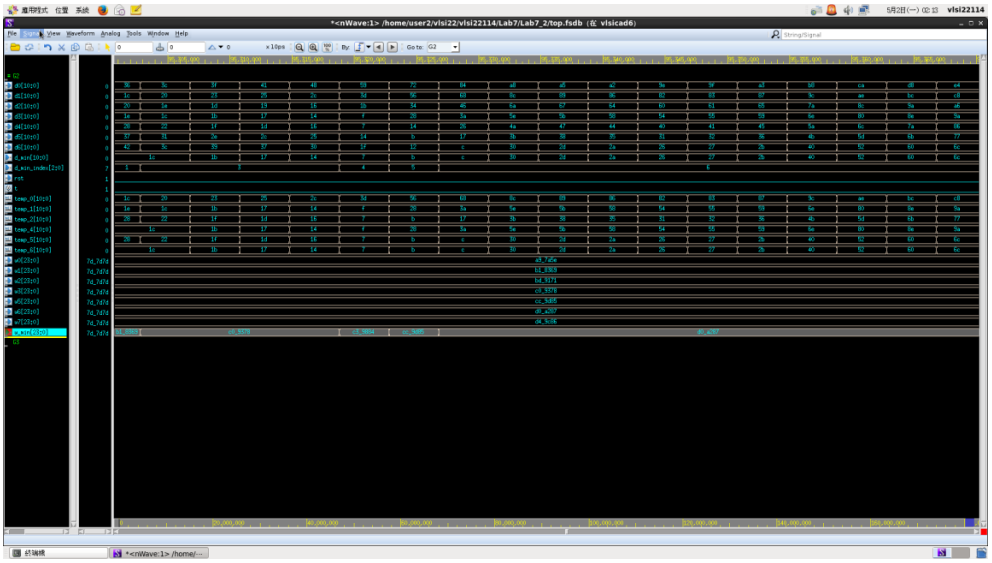
9) Compile the verilog code to verify the operations of this module works properly.

10) Synthesize your *TOP.v* with following constraint:

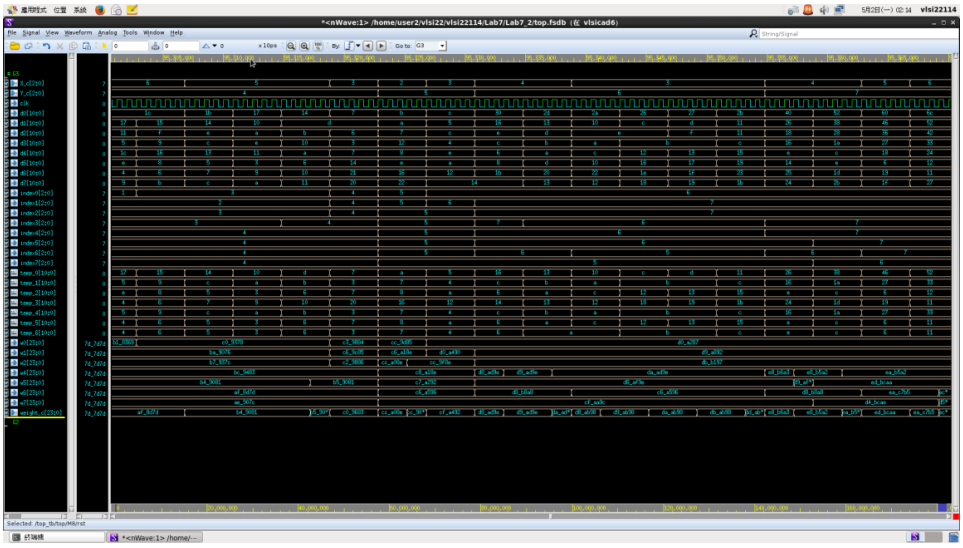
- Clock period: no more than 20 ns.
- Don't touch network: clk.
- Wire load model: saed14rvt_ss0p72v125c.
- Synthesized verilog file: *top_syn.v*.
- Timing constraint file: *top_syn.sdf*.

11) Please **attach your waveforms** and **specify your operations** on the waveforms.

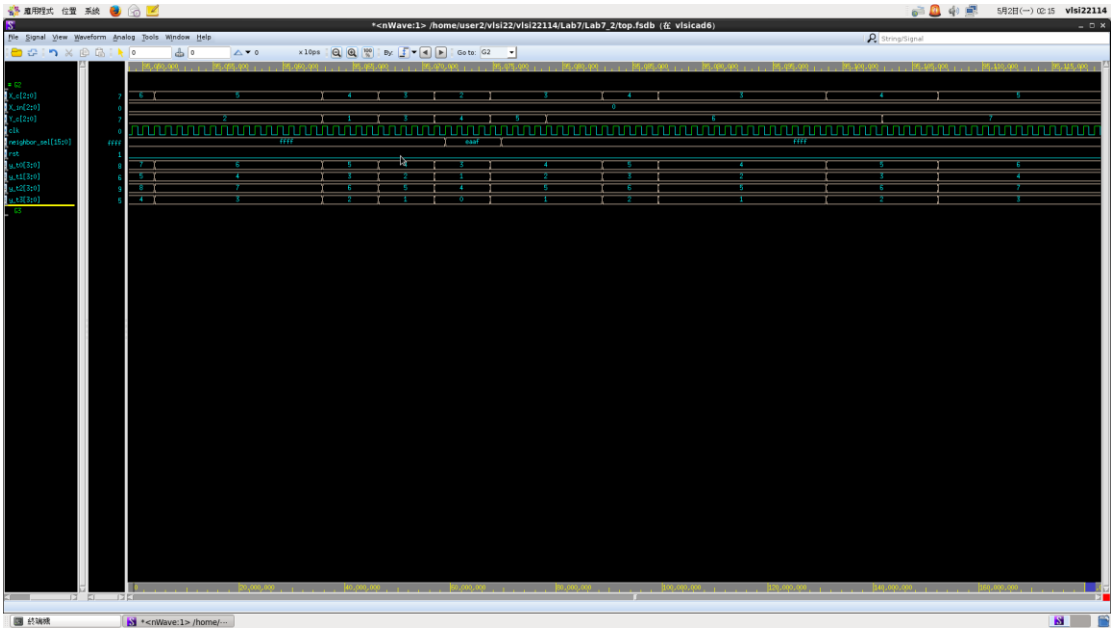
MIN_1



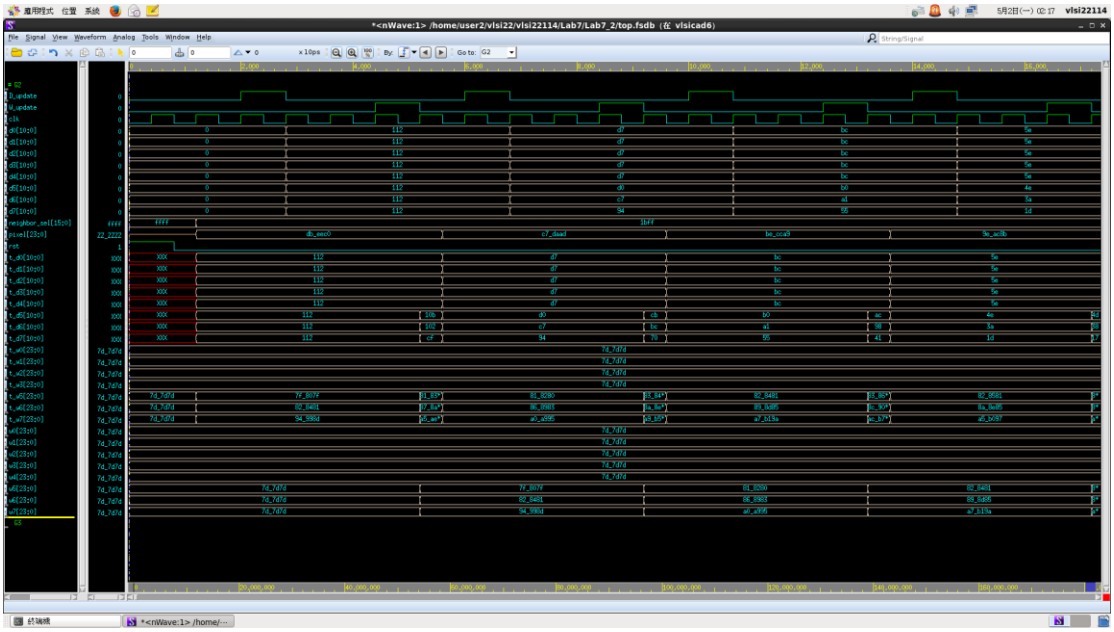
MIN_2



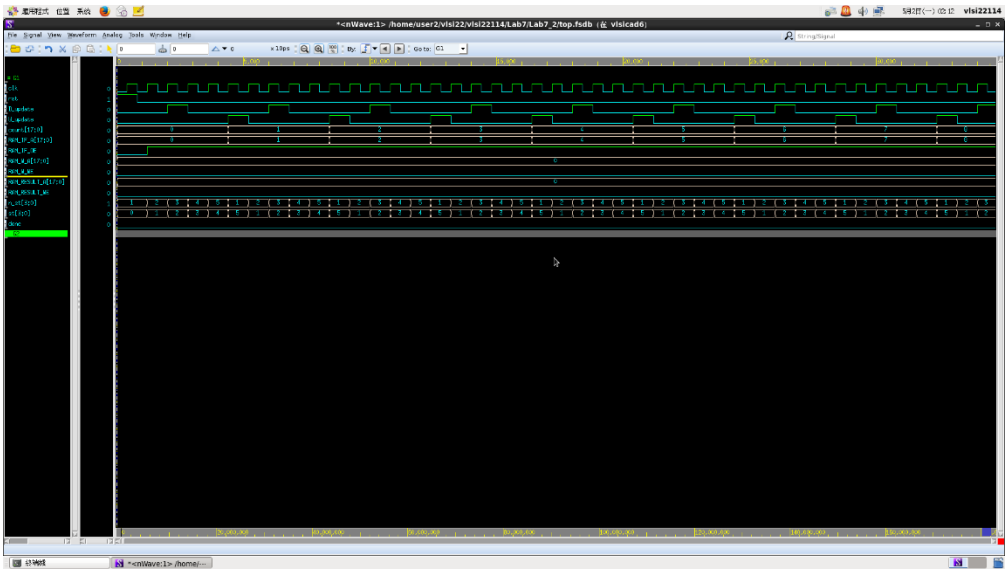
USS



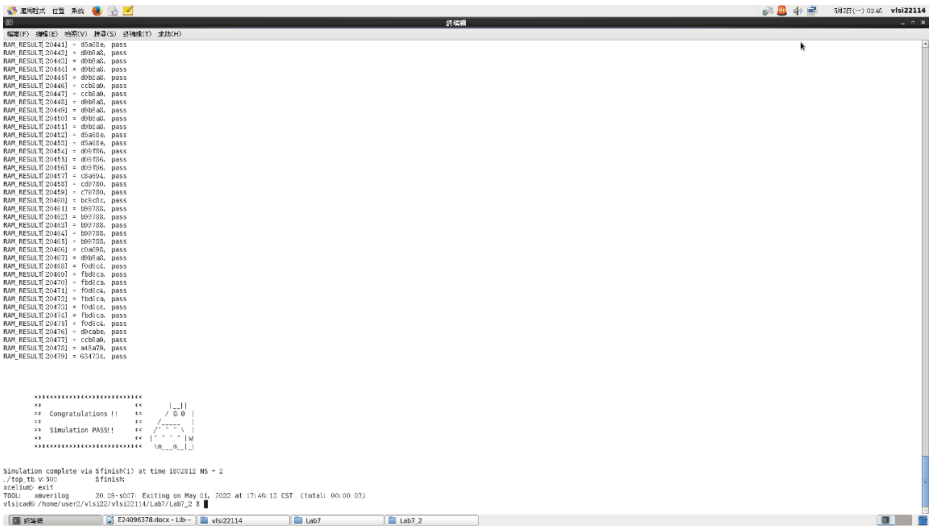
VEP



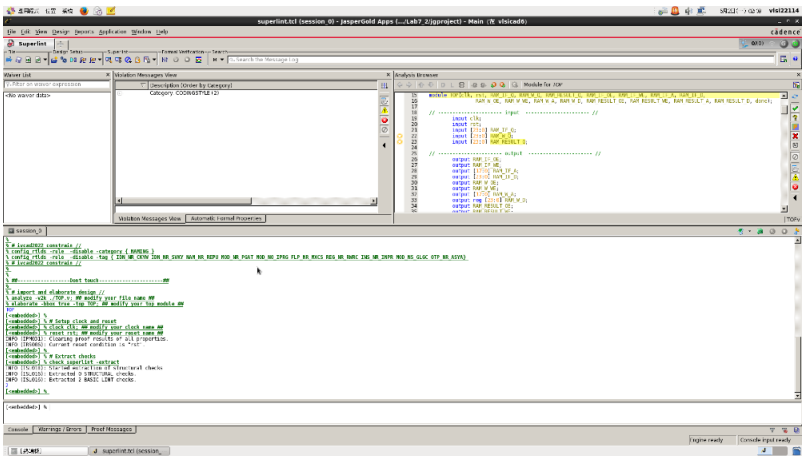
Controller



12) Show simulation result



13) Show SuperLint coverage (TOP.v)



Coverage = 0.99

14) Your clock period, total cell area, post simulation time (TOP.v)

Clock period:20ns

Total cell area:25641.621238

Post simulation time:1802812

Please compress all the following files into one compressed file (".tar " format) and submit through Moodle website:

※ NOTE:

1. If there are other files used in your design, please attach the files too and make sure they're properly included.
2. Simulation command

Problem	Command
Lab7_1(pre-sim)	ncverilog top_tb.v +define+X (WEIGHT, RESULT, FULL)
Lab7_1 (pre-sim with waveform)	ncverilog top_tb.v +access+r +define+FSDB+X
Lab7_1(post-sim)	ncverilog top_tb.v +define+syn+X
Lab7_1 (post-sim with waveform)	ncverilog top_tb.v +access+r +define+FSDB+syn+X
Lab7_2(pre-sim)	ncverilog top_tb.v +define+X (WEIGHT, RESULT0, RESULT1, RESULT2, RESULT3, RESULT4, FULL)
Lab7_2 (pre-sim with waveform)	ncverilog top_tb.v +access+r +define+FSDB+X
Lab7_2(post-sim)	ncverilog top_tb.v +define+FSDB+syn+X
Lab7_2 (post-sim with waveform)	ncverilog top_tb.v +access+r +define+FSDB+syn+X