**National Cheng Kung University**

**Department of Electrical Engineering**

***Introduction to VLSI CAD (Spring 2022)***

**Lab Session 7**

**SOM Processing System**

| Name | Student ID | | |
| --- | --- | --- | --- |
| 吳紹齊 | E34073071 | | |
| 張育誠 | E24066200 | | |
| Practical | | Points | Marks |
| Lab 7\_1 | | 45 |  |
| Lab 7\_2 | | 45 |  |
| Demo | | 10 |  |
| Notes | |  |  |
|  | | | |

**Due: 15:00 April 27, 2022@ 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!

1. All homework requirements should be uploaded in this file hierarchy.
2. NOTE: 1. Please **DO NOT** upload waveforms (.fsdb or .vcd)!
3. If you upload a dead body which we can’t even compile, you will get NO credit!

1. All Verilog file should get at least 90% SuperLint Coverage.
2. All homework requirements should be uploaded in this file hierarchy or you will not get full credit, if you want to use some sub modules in your design but you do not include them in your tar file, you will get 0 point.

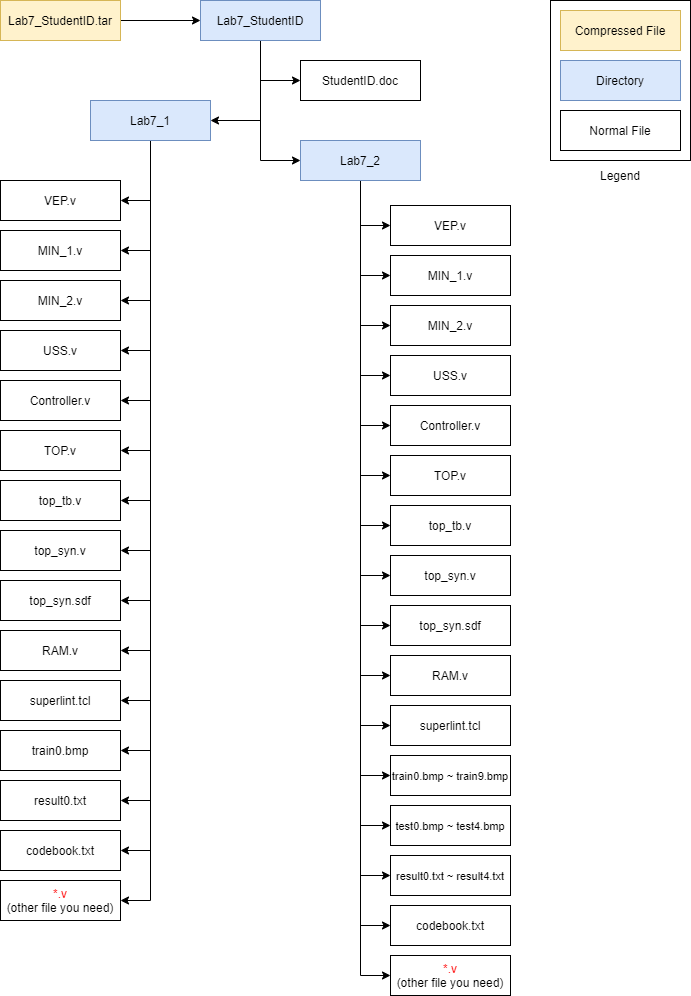
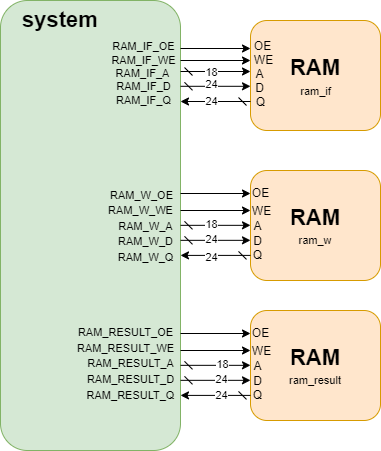


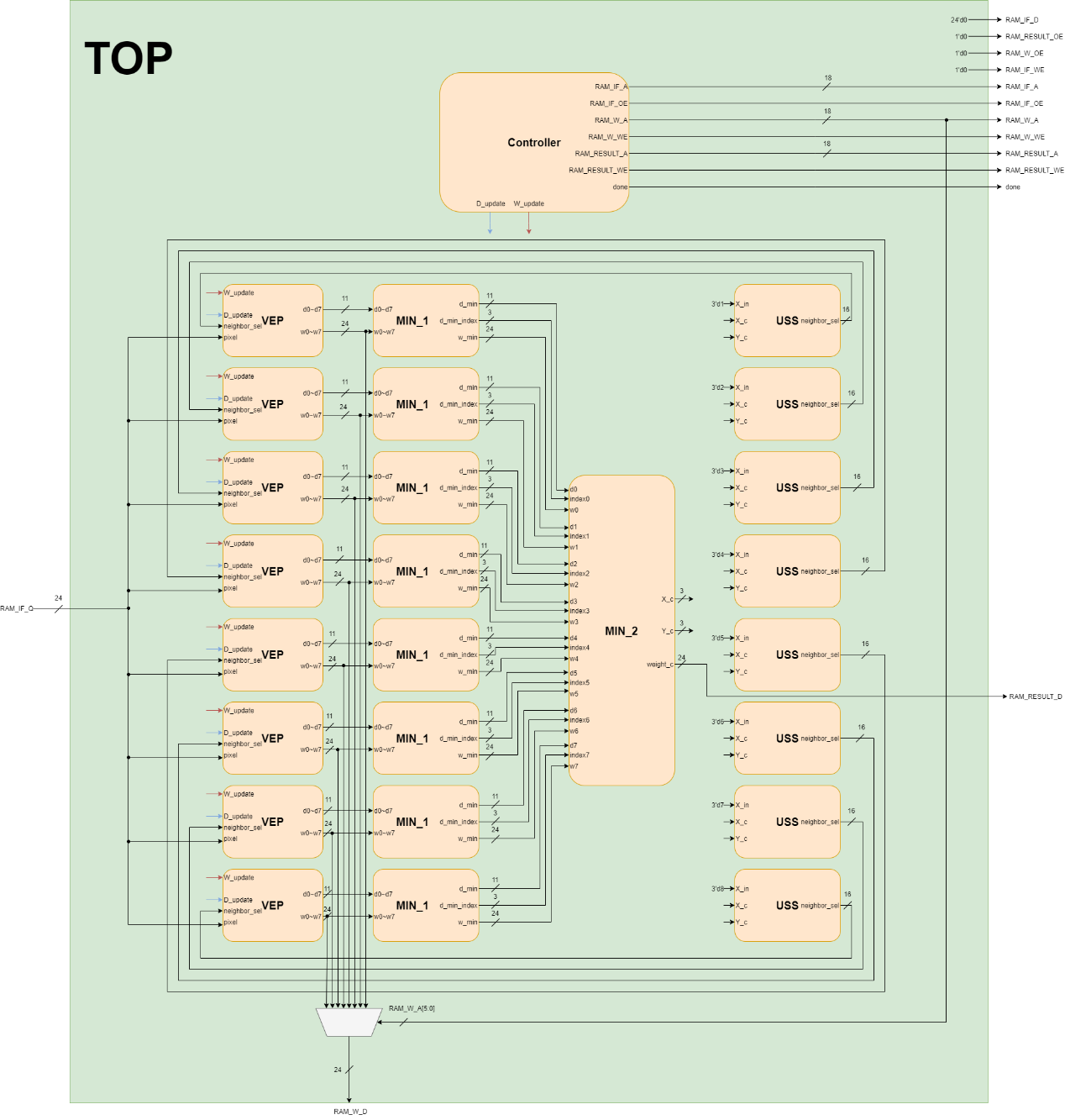
Fig.1 File hierarchy for Homework submission

*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)

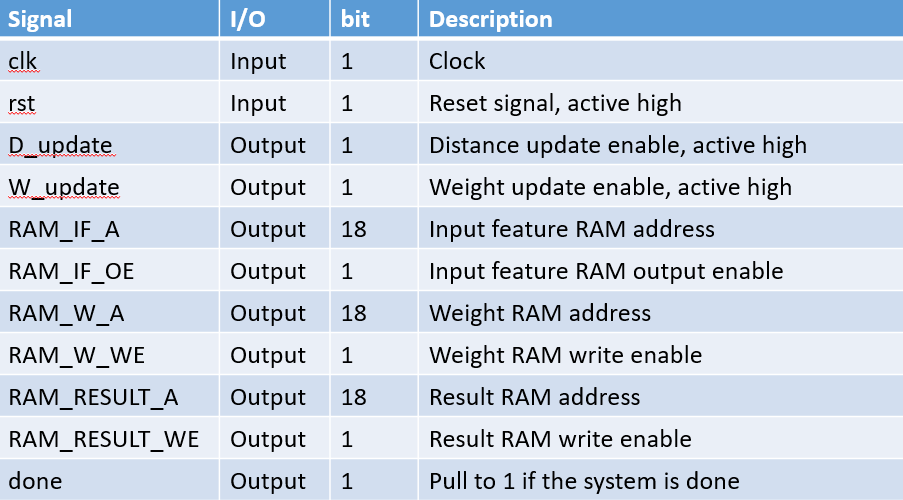


▲Fig2. The block diagram of system (external)

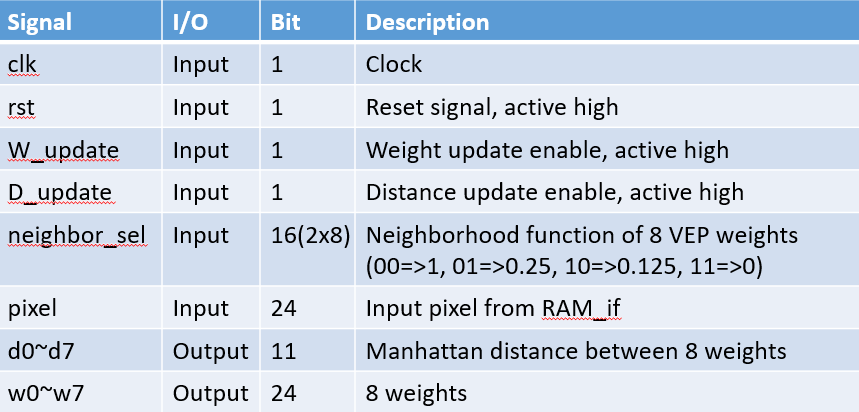
****

▲Fig3. The block diagram of system (internal)

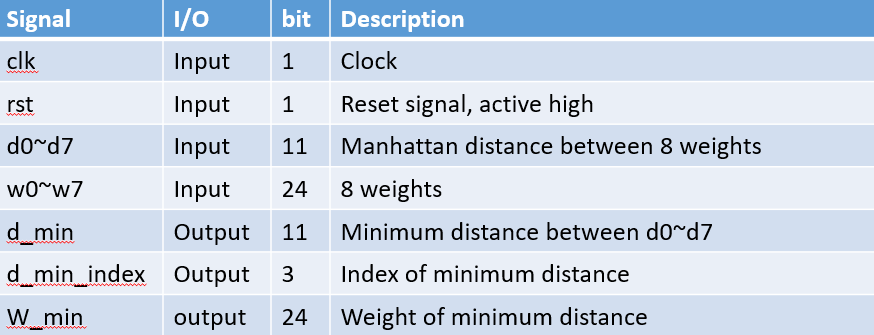
* **Port list of each module:**
* Controller



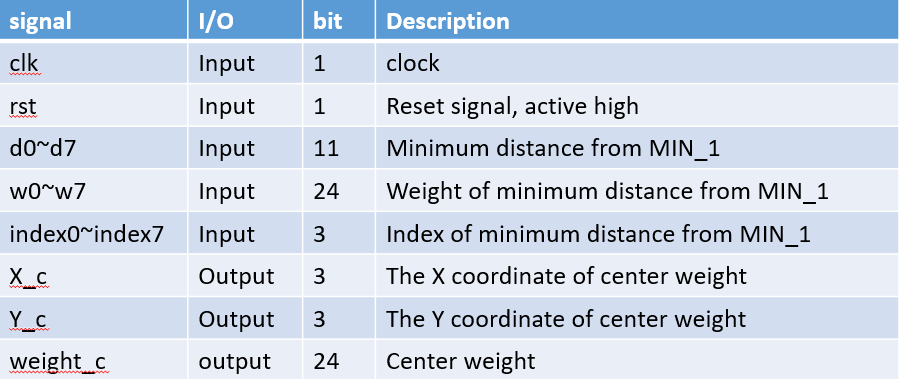
* **VEP**



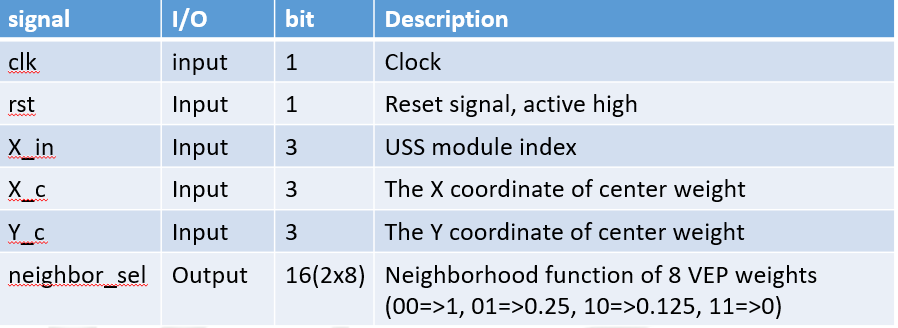
* **MIN\_1**



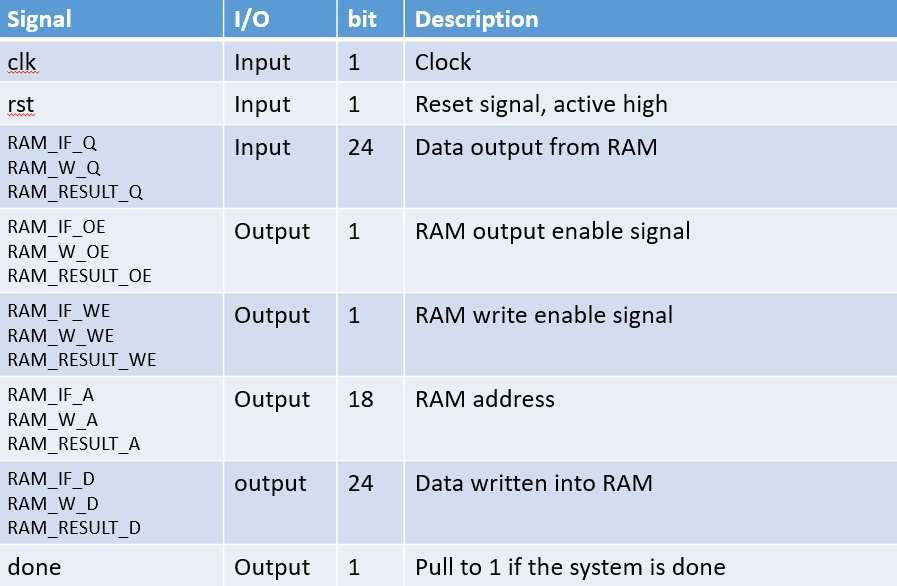
* **MIN\_2**



* **USS**



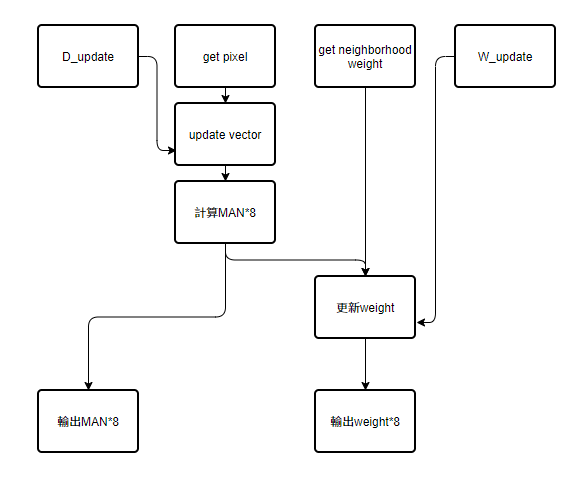
* Top



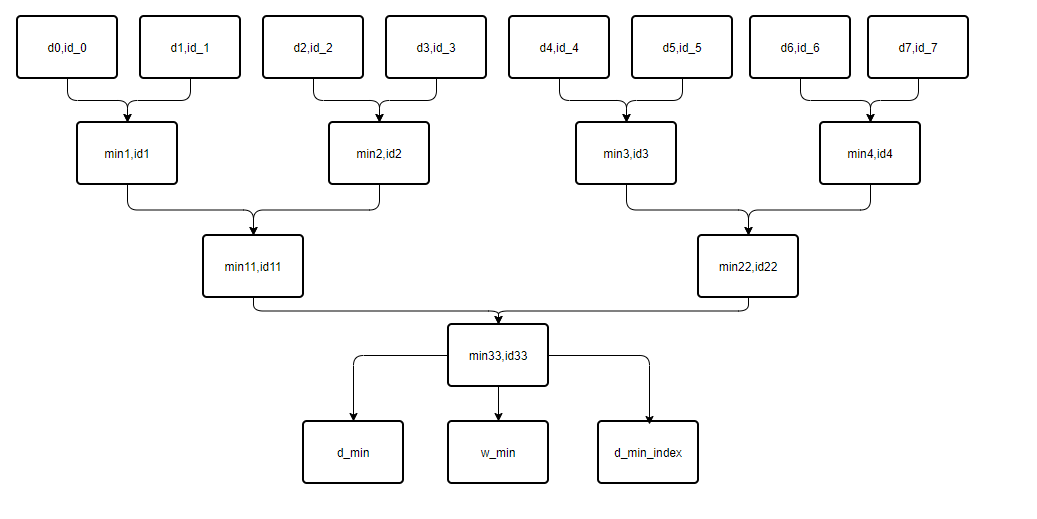
* Understanding the function:

Once system is initialized, it

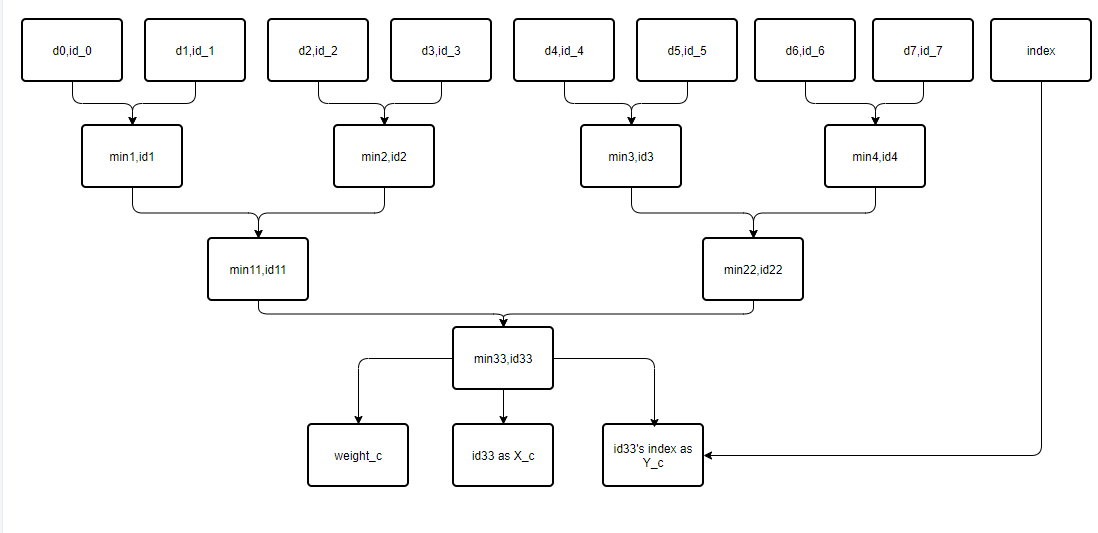
* 1. Read input pixel from RAM\_if
  2. Calculate Manhattan distance and fin the minimum distance
  3. Update the weight memory
  4. Repeats the process step(a)~(c)until the last pixel of RAM\_if is read
  5. writes the trained codebook to the RAM\_w
  6. read input pixel from RAM\_if and inference the picture
  7. writes the lossy compression picture to the RAM\_result
  8. repeats the process step (f)~(g) until the last pixel of RAM\_result is writed;
  9. flags “done” when system is completed
* Describe your design in detail. You can draw internal architecture or block diagram to describe your dsign.
  + VEP



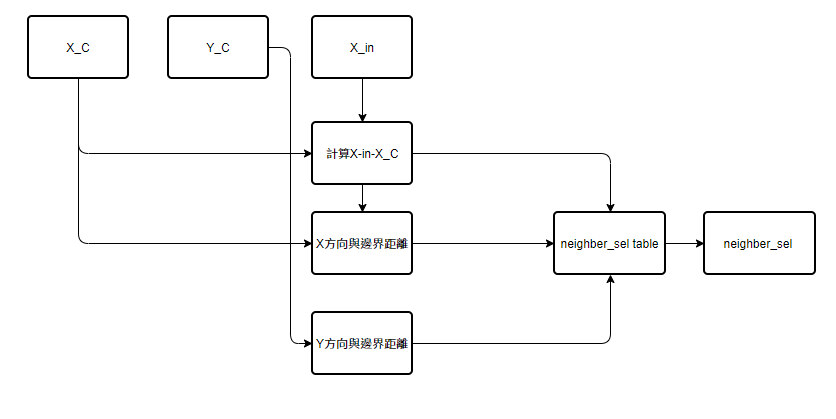
* + MIN\_1



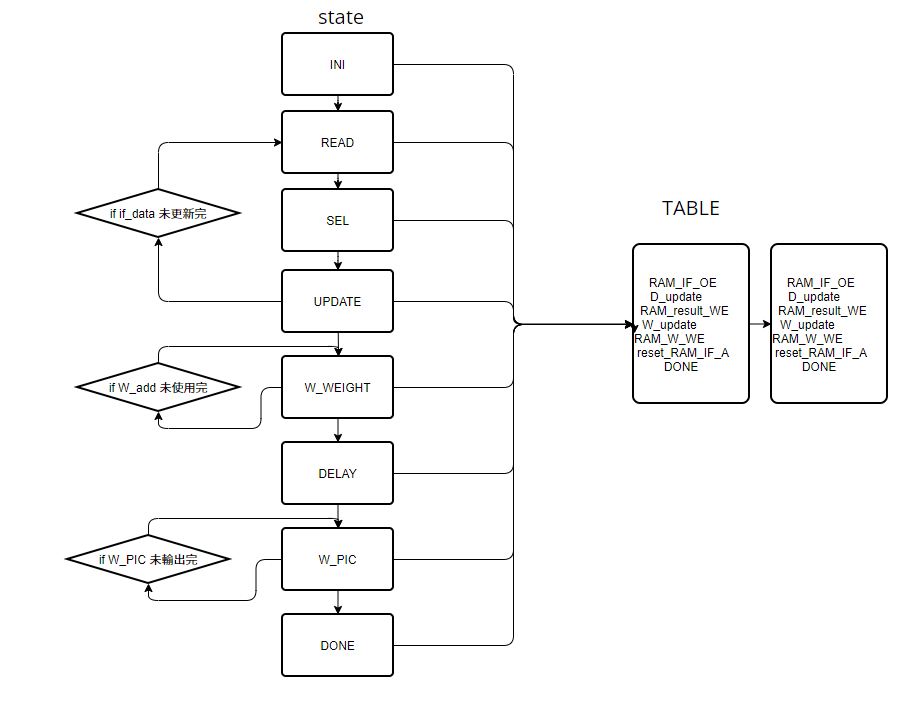
* + MIN\_2



* + USS



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

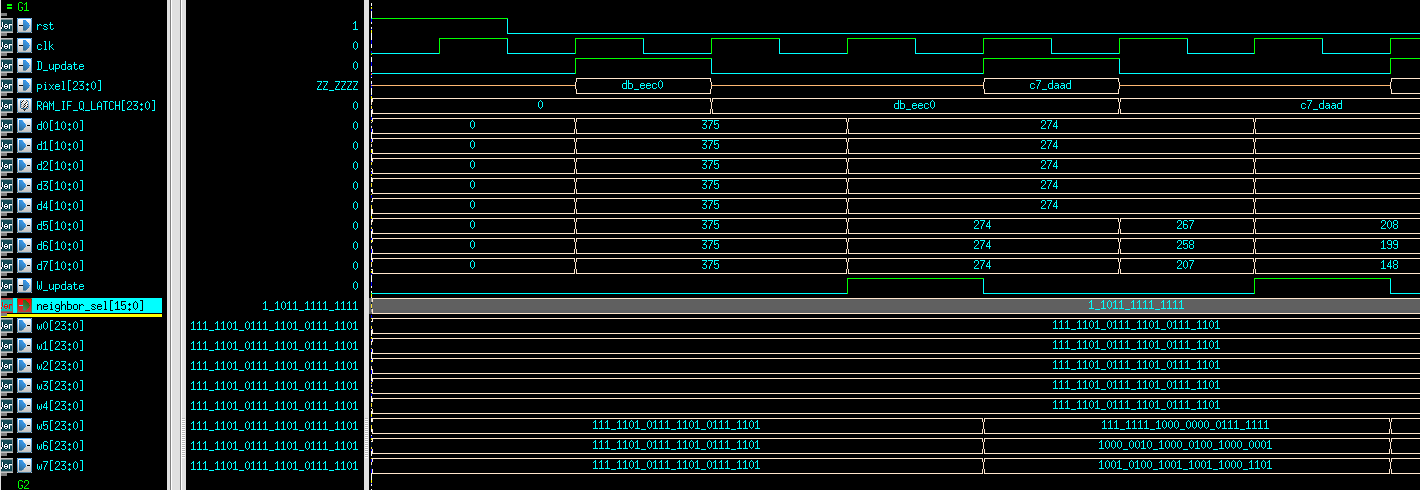


1. Complete the Controller, VEP, MIN\_1, MIN\_2, USS, and TOP module, in the system.
2. Compile the verilog code to verify the operations of this module works properly.
3. Synthesize your *TOP.v* with following constraint:

* 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*.

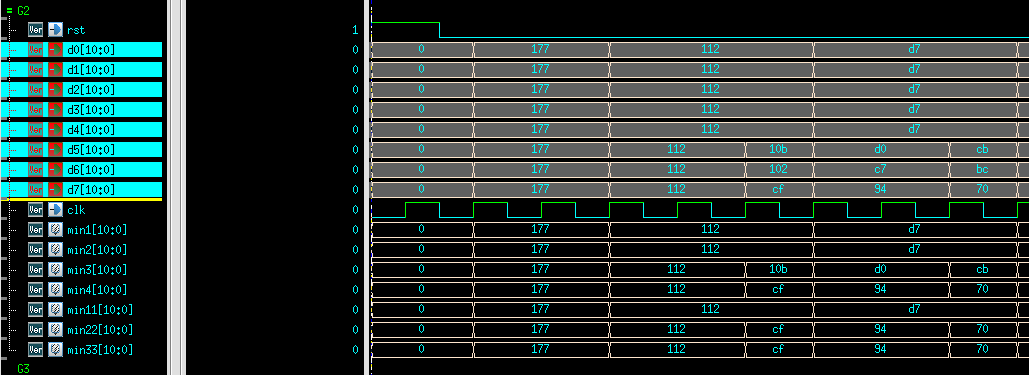
1. Please **attach your waveforms** and **specify your operations** on the waveforms.

VEP(VEP7為例)



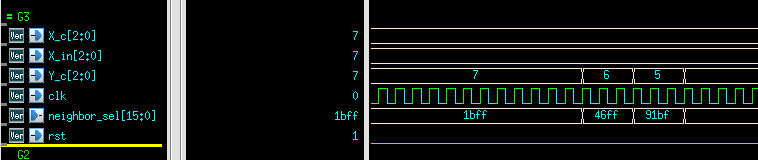
Reset後所有D變成0，所有W變成(125,125,125)並在第一個D\_update時進來第一個pixel，並在下一個clk吃進RAM\_IF\_Q\_LATCH，在經過combinational的運算後於下一個clk輸出到D0~D7，下方的W0~W7也在W\_update之後改變數值。

MIN(MIN7為例)

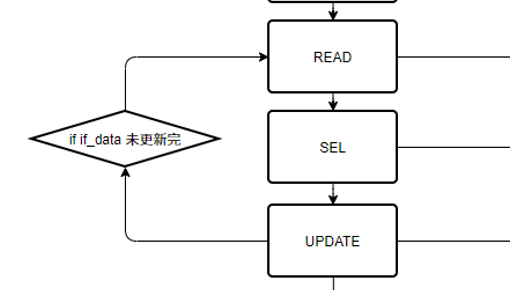


由於全部都是combinational所以只要有posedge，所有數值全部更新

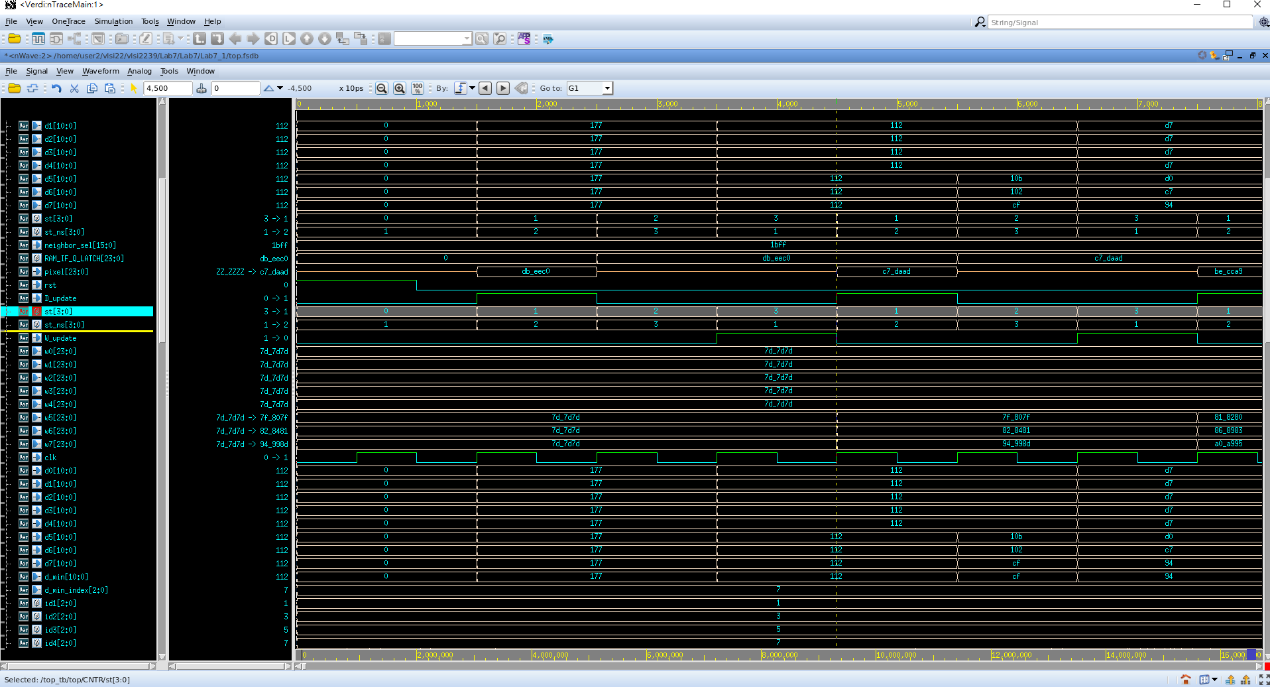
USS(USS7為例)



根據table輸出



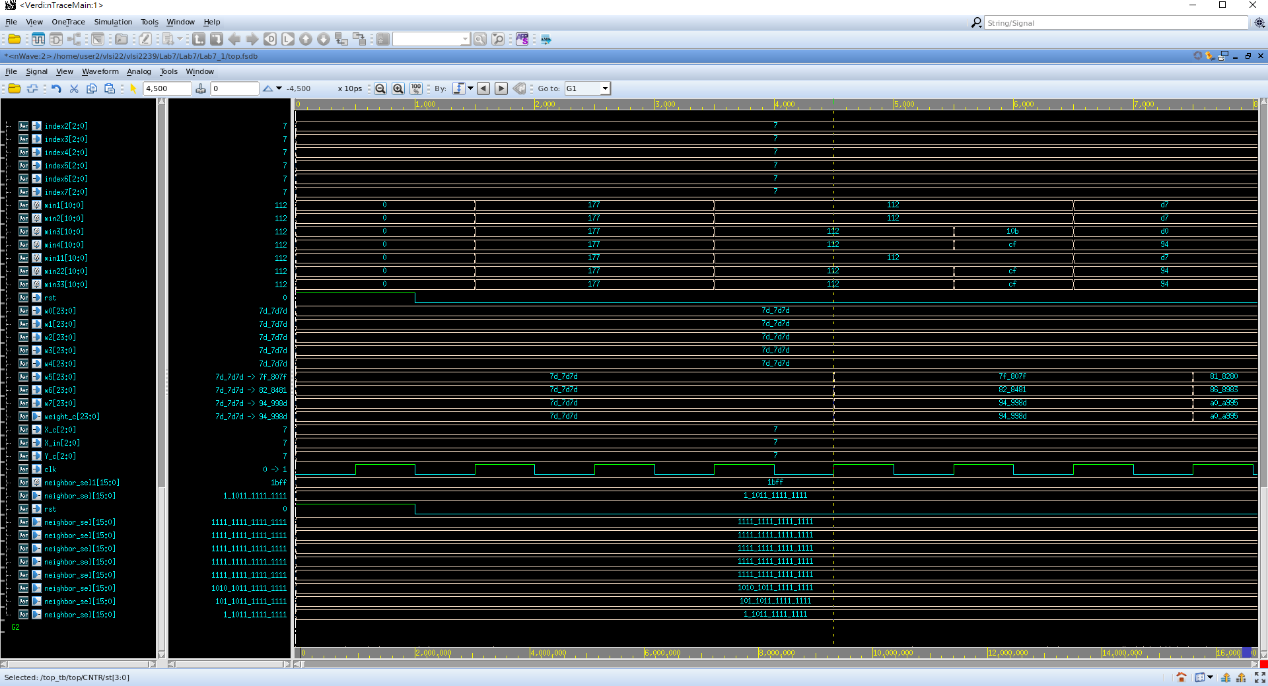
在訓練的部分，我分成3個state，第1個state是READ主要是將資料讀進來，例如第一筆資料是db\_eec0，資料讀進來後，儲存到RAM\_IF\_Q\_LATCH，RAM\_IF\_Q\_LATCH會維持db\_eec0直到Update完權重。

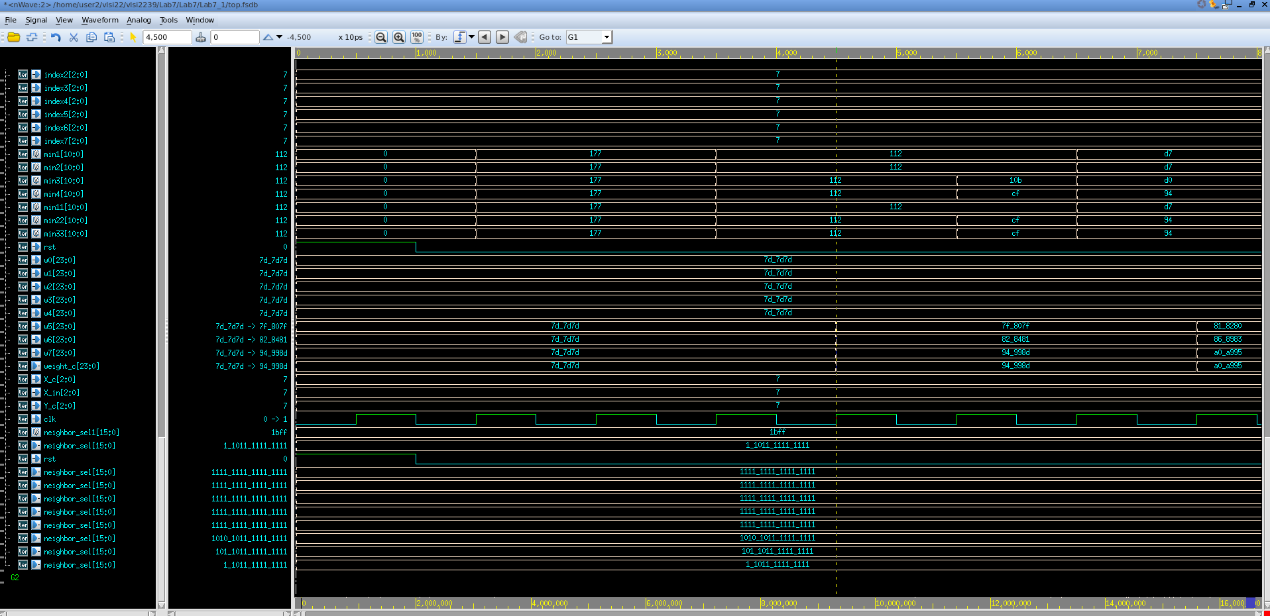


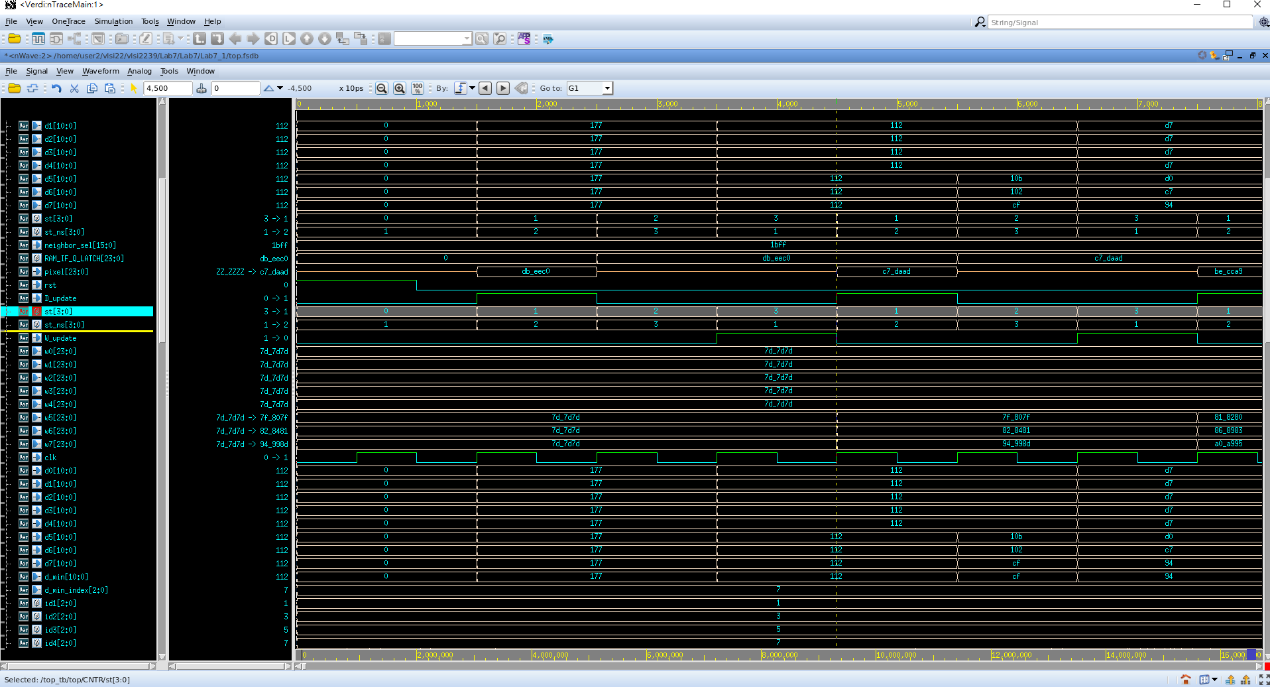
第2個state是SEL，主要是計算的部分，這裡我設計的都是 combinational block，MIN\_1、MIN\_2、USS都在1個clk裡，所以1個clk就可以算完neighbor\_sel，並輸出neighbor\_sel。

下圖可以看到完整的neighbor function

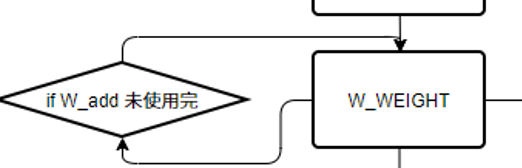
因為第一個更新中心點在(7,7)，所以是以左下角的點為中心。



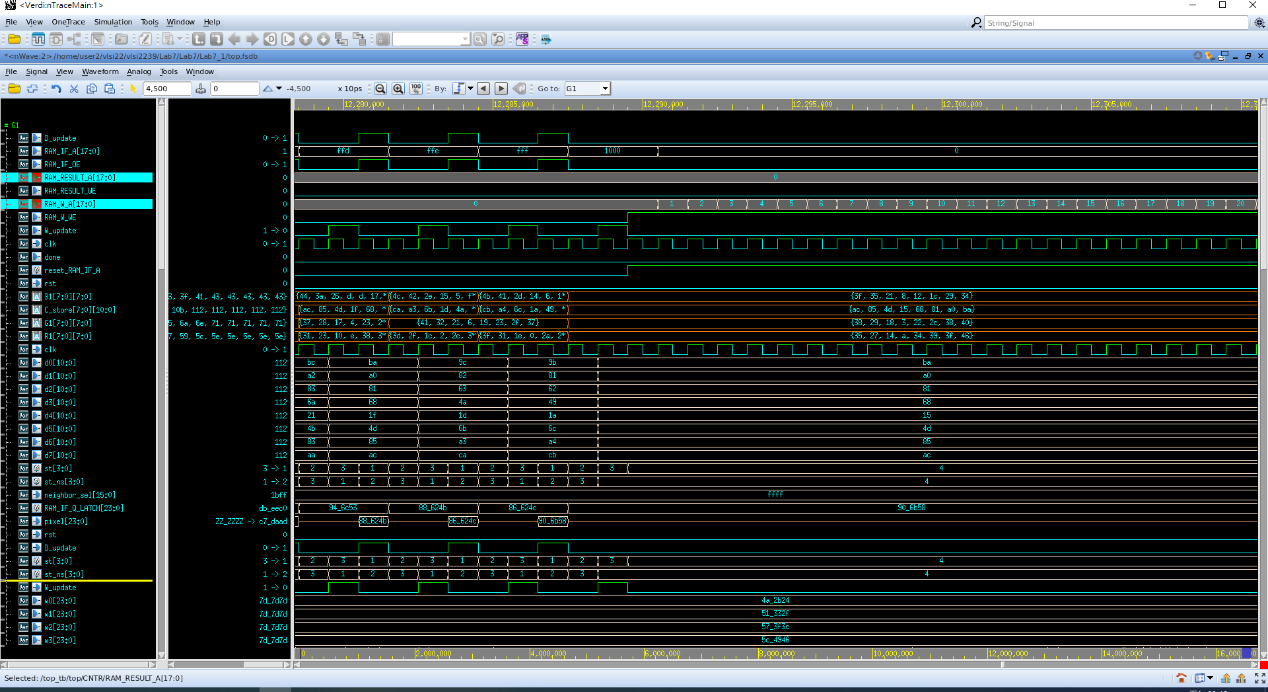
第三個state是UPDATE，主要是更新權重，根據接收到neighbor\_sel來更新權重，可以看到在UPDATE state時，W\_update升高，所以在下一個state更新了權重，也就是在4500ns時更新了新的權重。



在訓練了4096次後，拿到了訓練完的權重，接著進到W\_WEIGHT，更新 codebook資料，將更新的權重寫入RAM\_W\_Q。

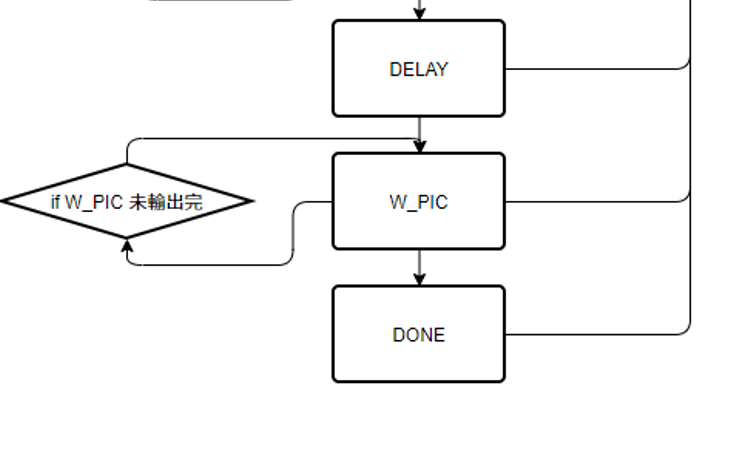


可以看到下圖的波型圖，進到W\_WEIGHT，RAM\_W\_WE為1，開始更新codebook資料，一共64組資料。

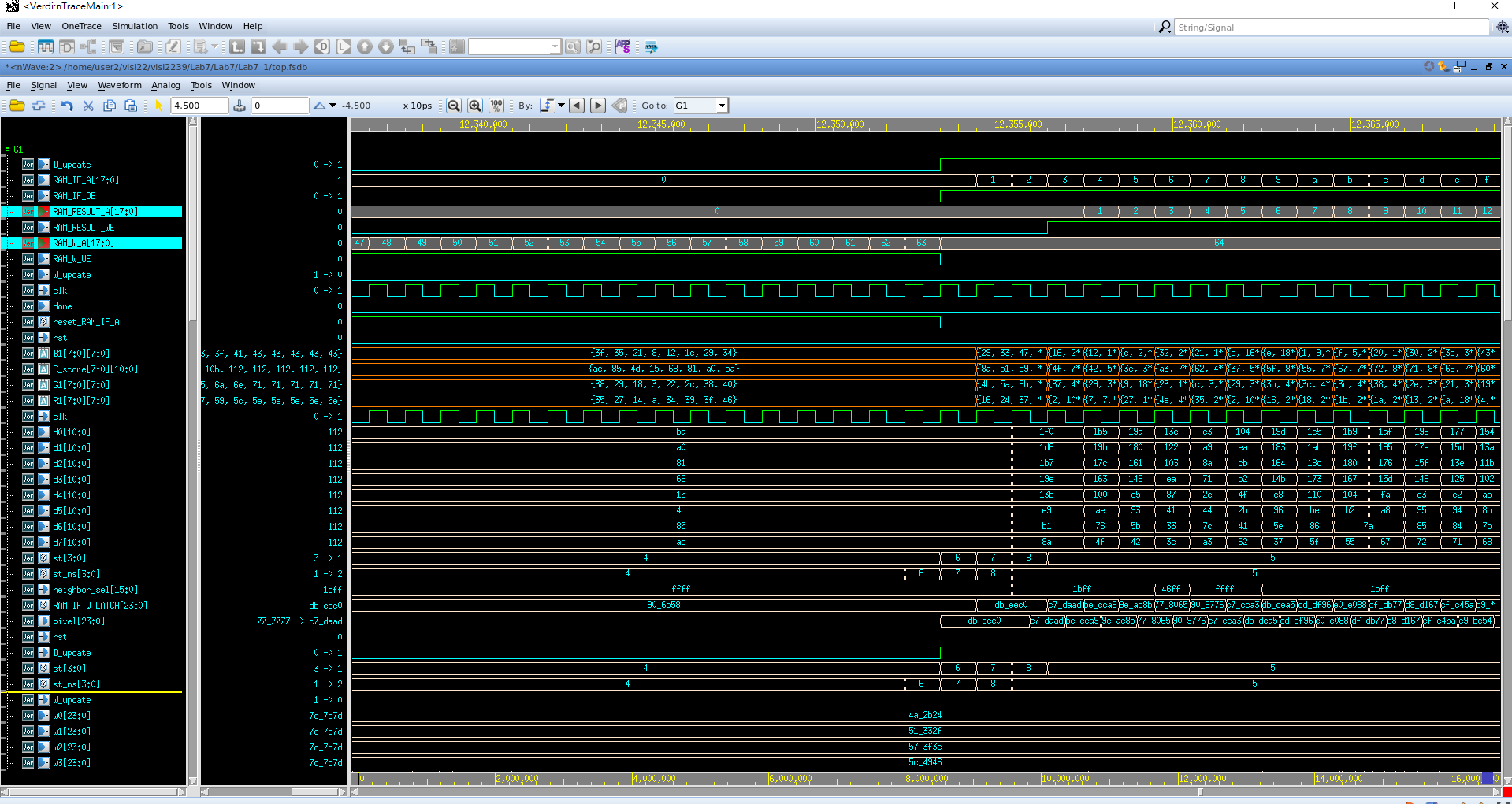


更新完codebook資料接著進到DELAY，這裡設計是因為有delay clk所以多設計一個state來符合傳輸資料的clock。

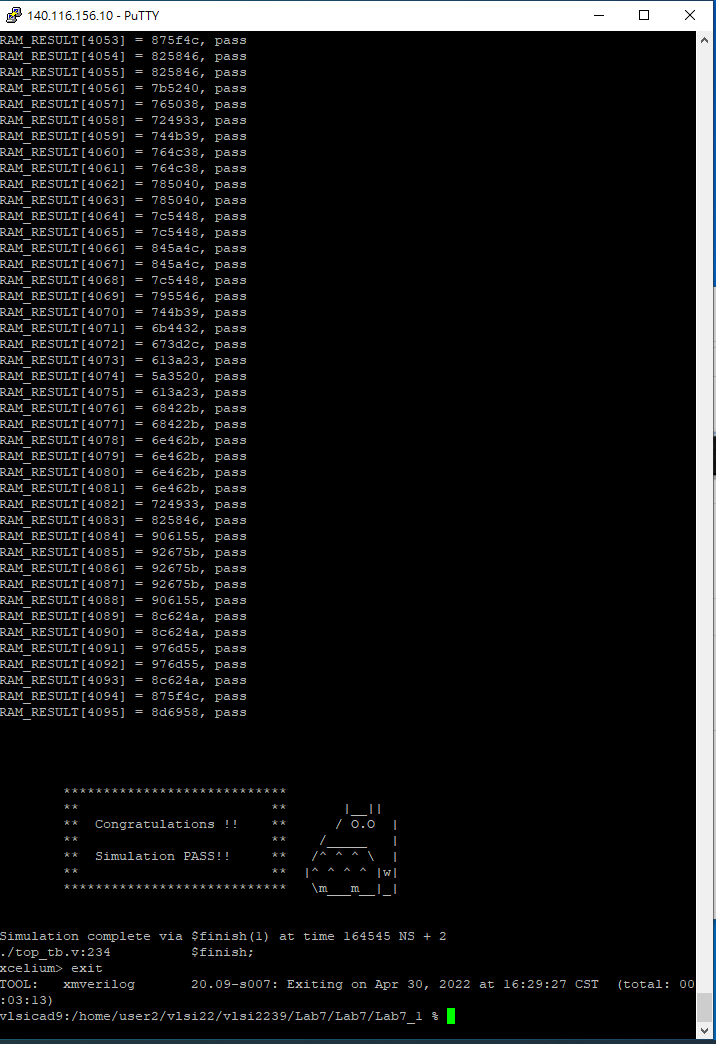
這裡主要是W\_PIC。



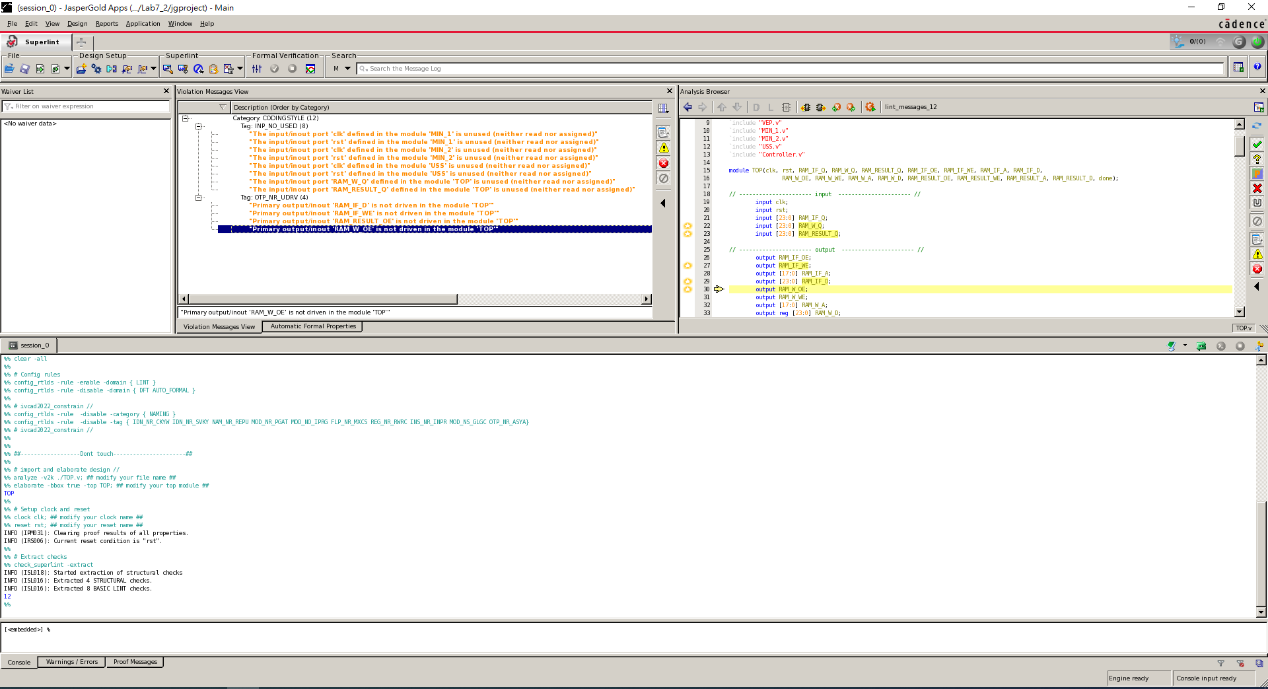
這裡write compress picture result，可以使用pipeline的方式來更新，這裡資料傳遞方式是先從RAM\_IF讀資料到RAM\_IF\_Q\_LATCH然後進到VEP，算calculate manhattan distance，進到MIN\_1比較最小的index，最後進到MIN\_2找中心點然後直接寫入RAM\_RESULT，這裡跟1~3 STATE的差別就是不計算neighbor function和更新權重，所以可以在一個clk就完成所有事情



1. Show simulation result



1. Show SuperLint coverage (TOP.v)



99%

都是unused的訊號

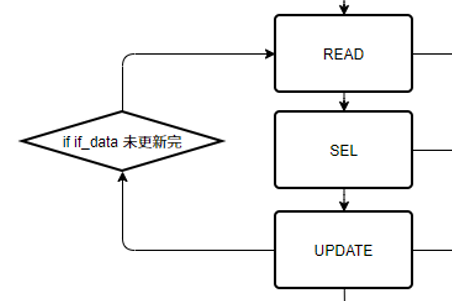
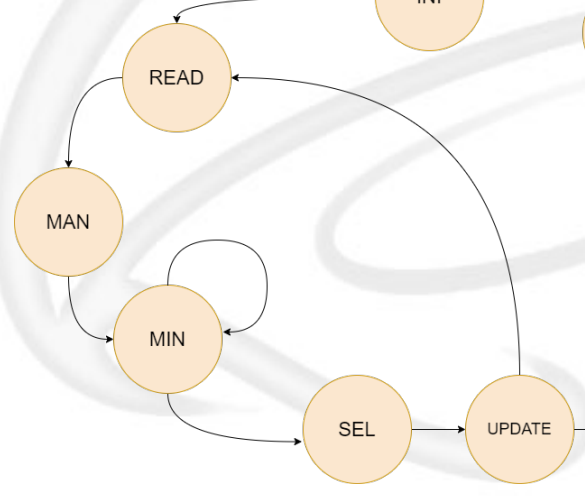
1. Your clock period, total cell area, post simulation time (TOP.v)

Clock period:10

Total cell area: 24657.273257

Post simulation time:164545ns

1. Please describe how you optimize your design when you run into problems in synthesis .ex: plug in some registers between two instances to shorten your datapath, resource sharing for some registers to reduce your cell area.



上圖可以看到原本助教的在更新state切成6個state，但其實不用切到那麼細，

我設計成3個state就可以更新一次weight了，甚至還有足夠的時間去壓clock。

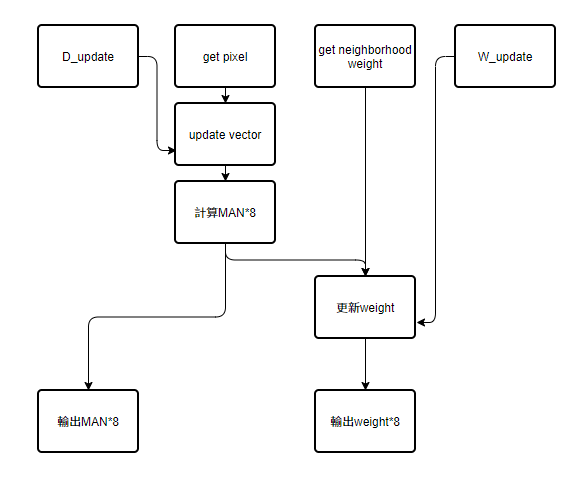
縮小面積的部分是我原本是用counter來控制state的切換，但後來發現RAM\_IF\_A、RAM\_W\_A、RAM\_RESULT\_A就是天然的counter，直接利用他們來控制state的切換，省下另外建一個counter的register。

* Lessons learned from this lab

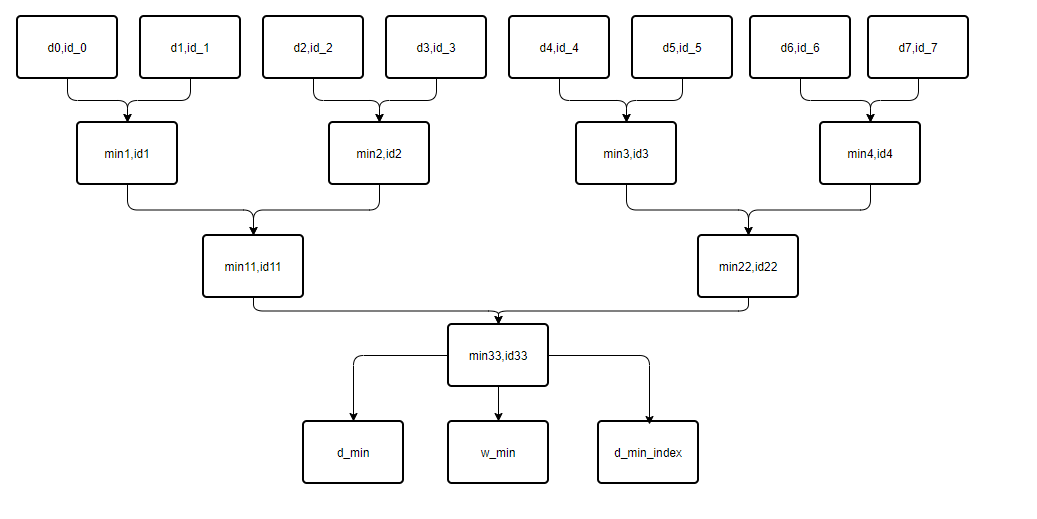
這次的電路架構是最大的一次，所以debug起來相對較難，寫架構花了2天，debug也花了2天，尤其是更新權重的部分特別難debug，因為我是中間有少部分地方算錯，所以導致最後結果錯，前面的波行都是對的，最後一個一個module檢查，才發現是自己耍白癡複製貼上參數忘記改，經過這次其中project，感覺比之前前幾個lab進步幅度多很多。

*Lab 7\_2*

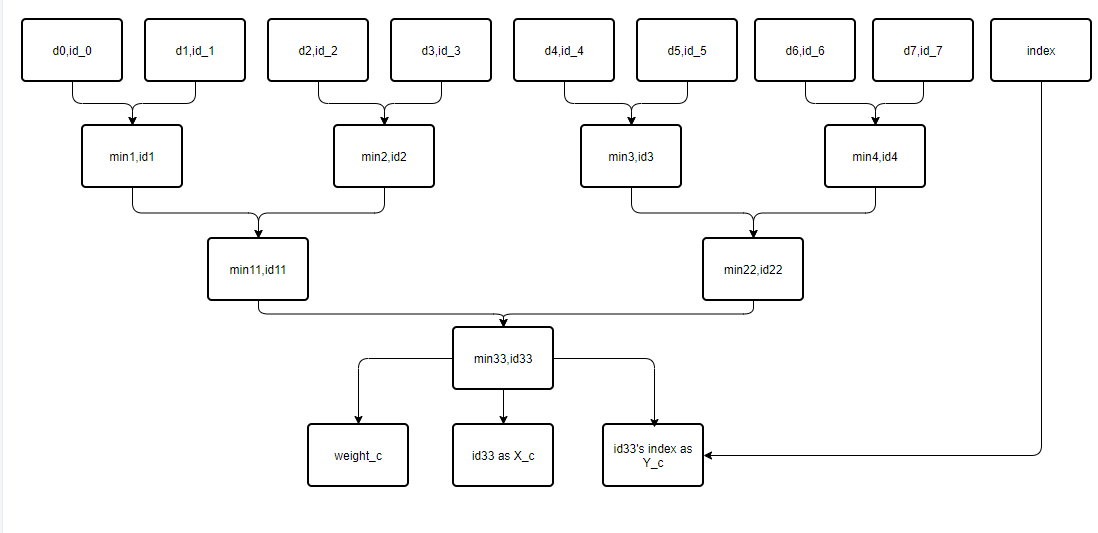
* Draw your state diagram and explain your design. You can draw internal architecture to describe your design
* Describe your design in detail. You can draw internal architecture or block diagram to describe your dsign.
  + VEP



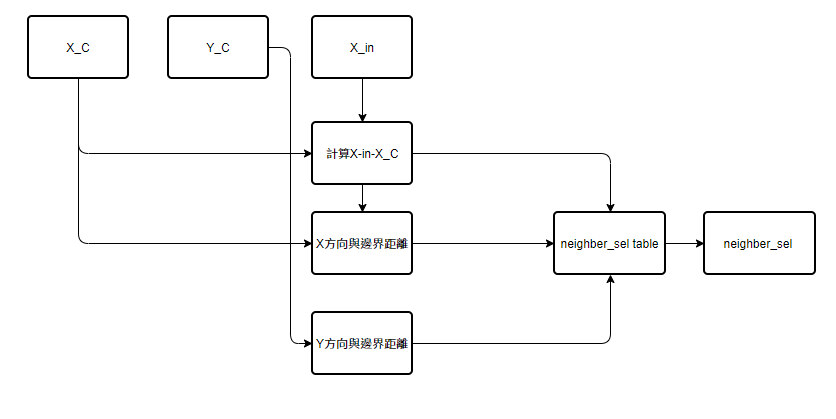
* + MIN\_1



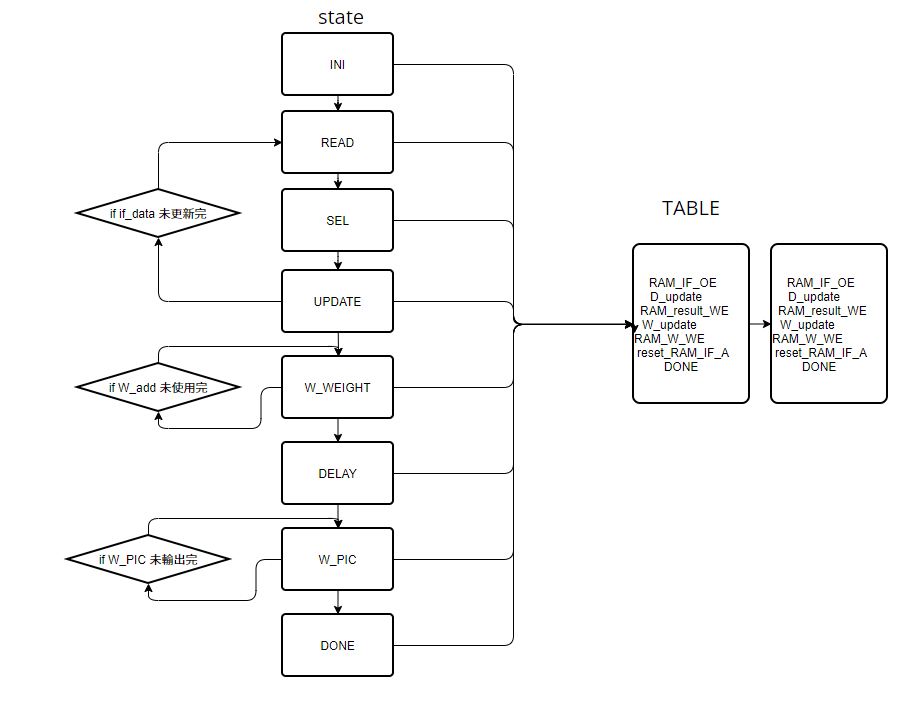
* + MIN\_2



* + USS

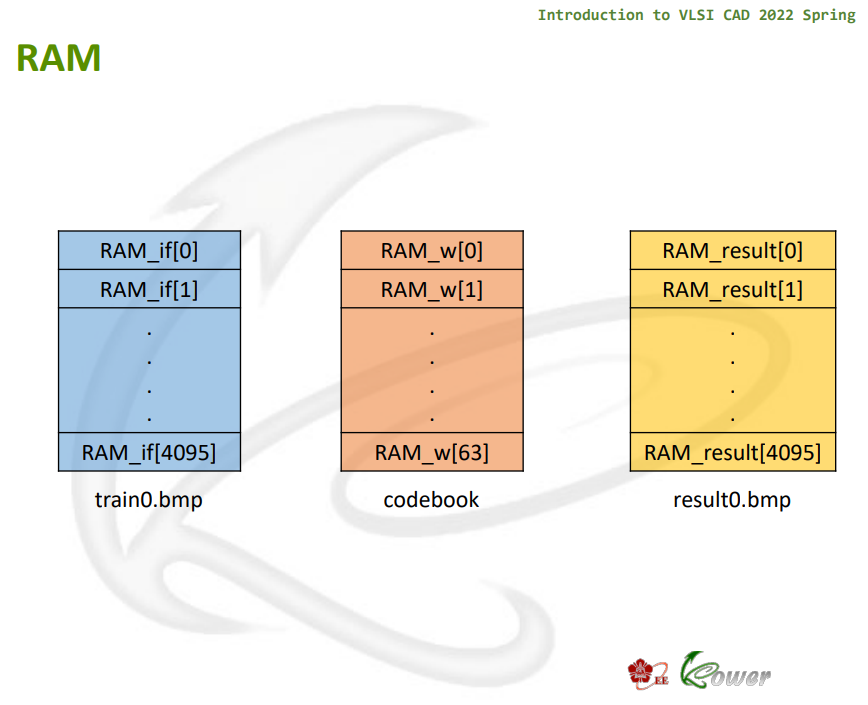
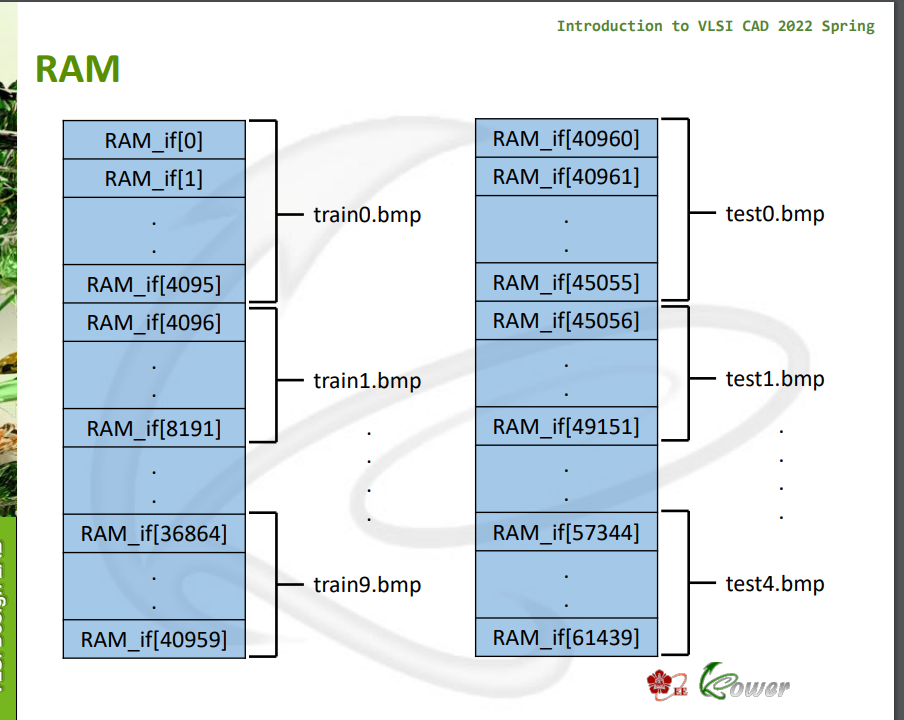
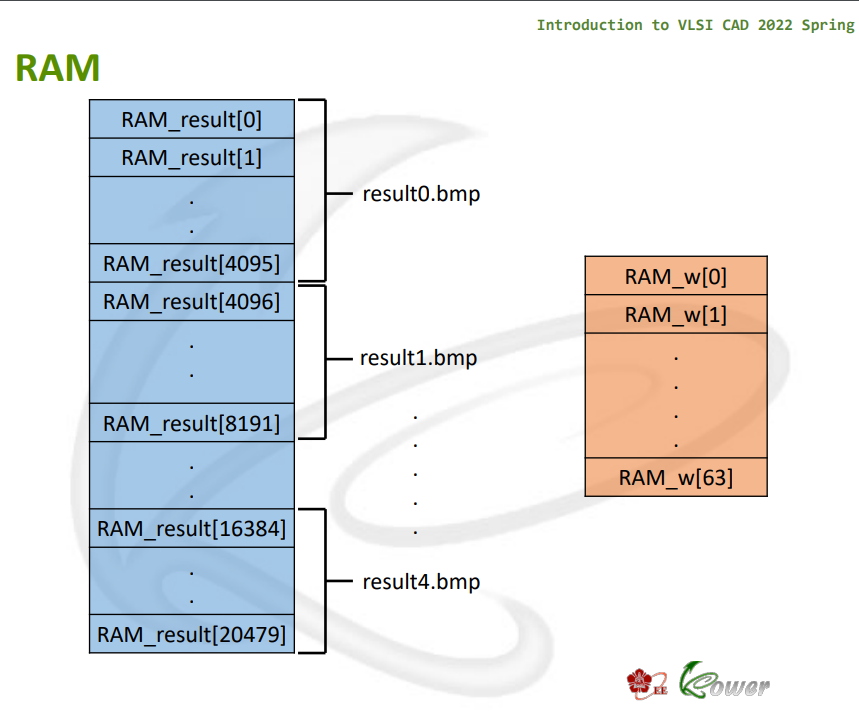


* + Controller
    - Draw your state diagram in controller and explain it



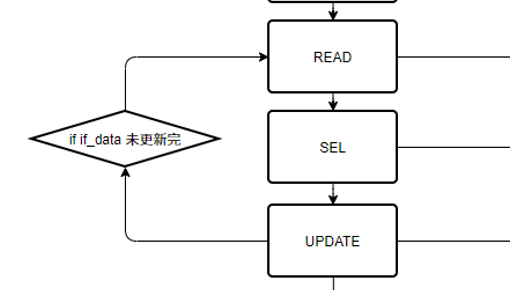
1. Complete the Controller, VEP ,MIN\_1, MIN\_2, USS, and TOP module, in the system.
2. Compile the verilog code to verify the operations of this module works properly.
3. Synthesize your *TOP.v* with following constraint:

* 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*.

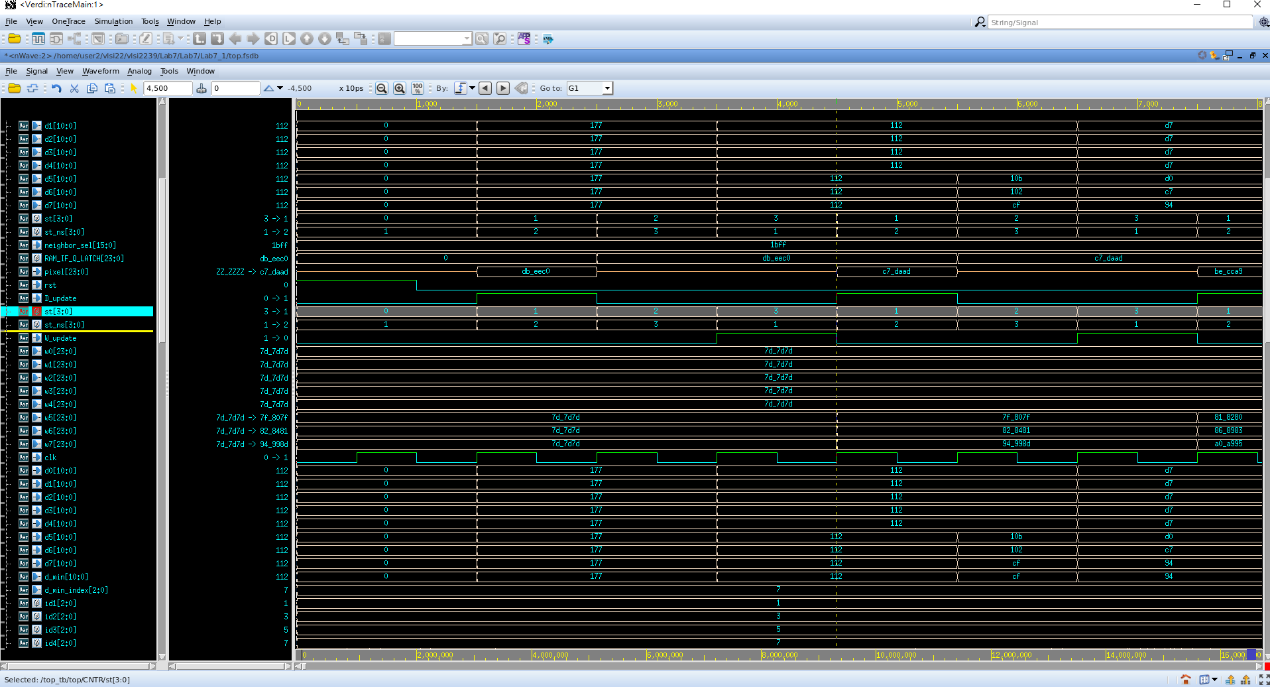


1. Please **attach your waveforms** and **specify your operations** on the waveforms.

Lab7\_1和Lab7\_2的區別是在Train picture和Inference picture的數量， Lab7\_2的數量分別是Lab7\_1的10倍Train picture，Inference picture的5倍



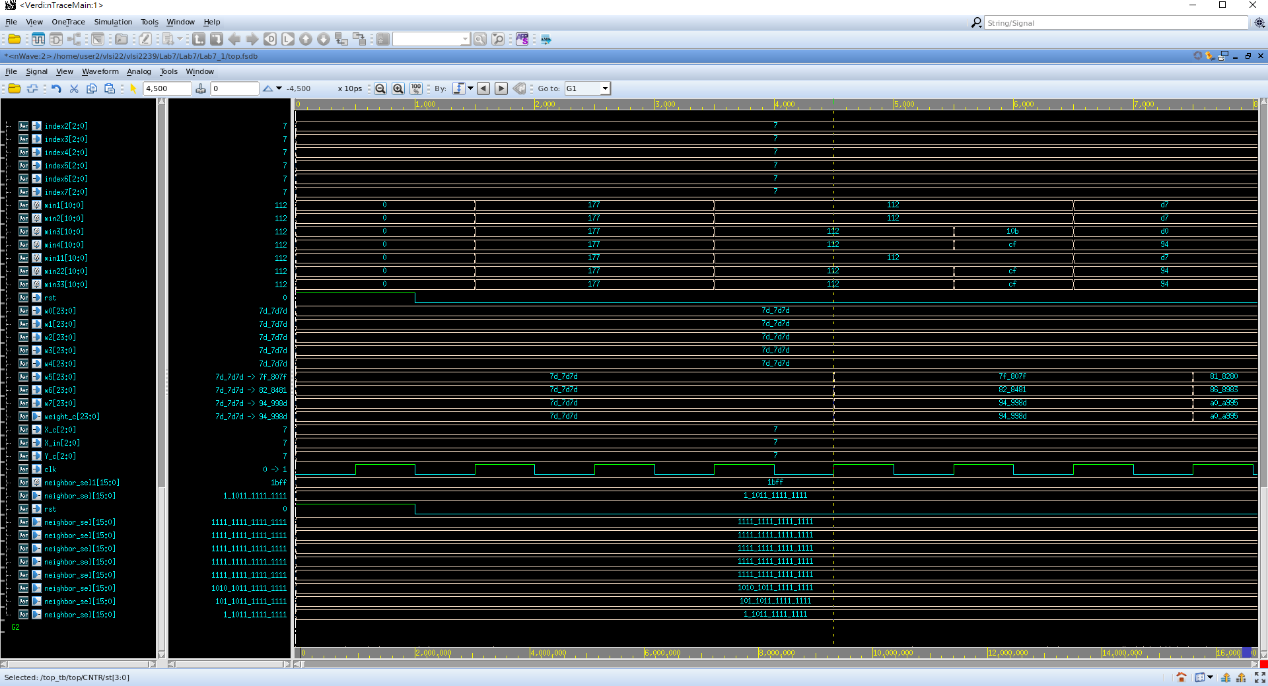
在訓練的部分，我分成3個state，第1個state是READ主要是將資料讀進來，例如第一筆資料是db\_eec0，資料讀進來後，儲存到RAM\_IF\_Q\_LATCH，RAM\_IF\_Q\_LATCH會維持db\_eec0直到Update完權重。

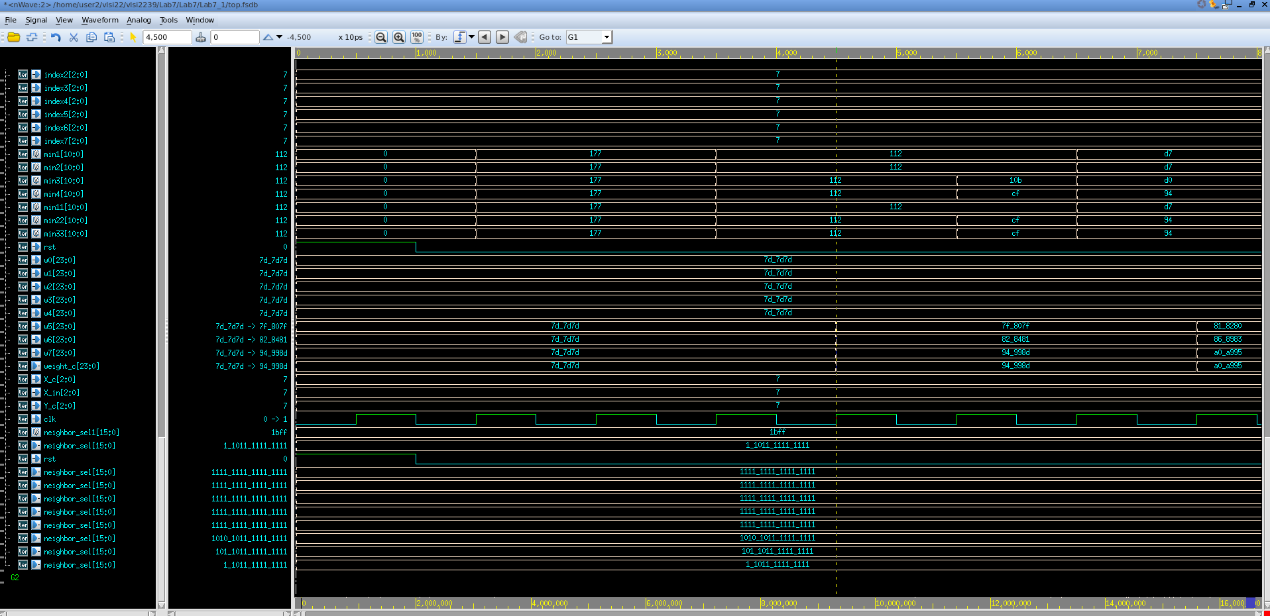


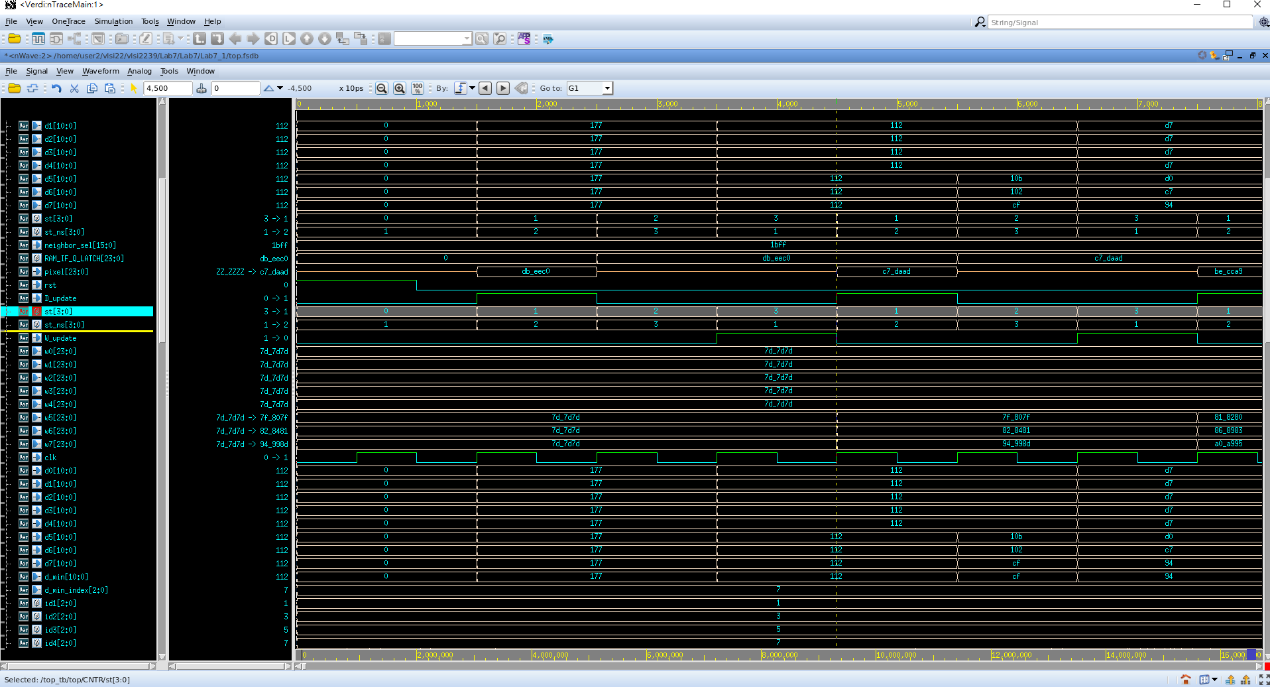
第2個state是SEL，主要是計算的部分，這裡我設計的都是 combinational block，MIN\_1、MIN\_2、USS都在1個clk裡，所以1個clk就可以算完neighbor\_sel，並輸出neighbor\_sel。

下圖可以看到完整的neighbor function

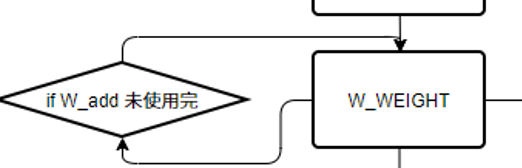
因為第一個更新中心點在(7,7)，所以是以左下角的點為中心。



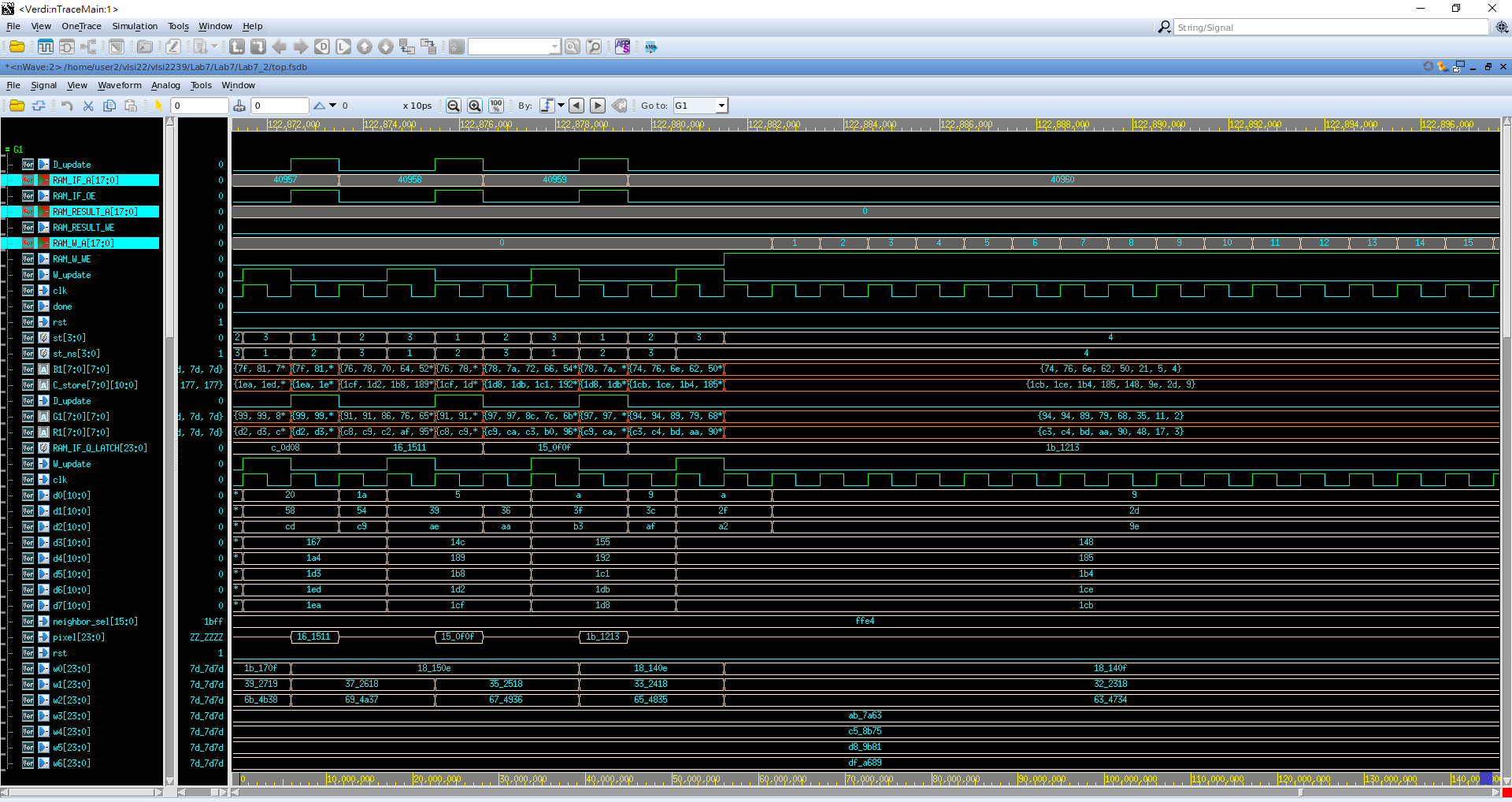
第三個state是UPDATE，主要是更新權重，根據接收到neighbor\_sel來更新權重，可以看到在UPDATE state時，W\_update升高，所以在下一個state更新了權重，也就是在4500ns時更新了新的權重。



在訓練了40960次後，拿到了訓練完的權重，接著進到W\_WEIGHT，更新 codebook資料，將更新的權重寫入RAM\_W\_Q。

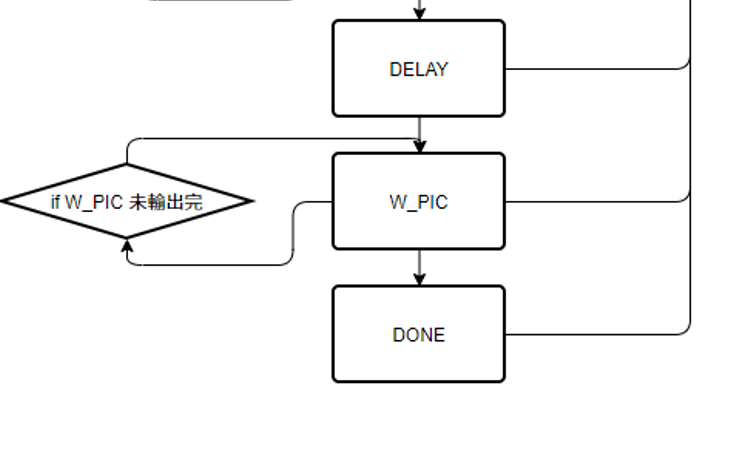


可以看到下圖的波型圖，進到W\_WEIGHT，RAM\_W\_WE為1，開始更新codebook資料，一共64組資料。



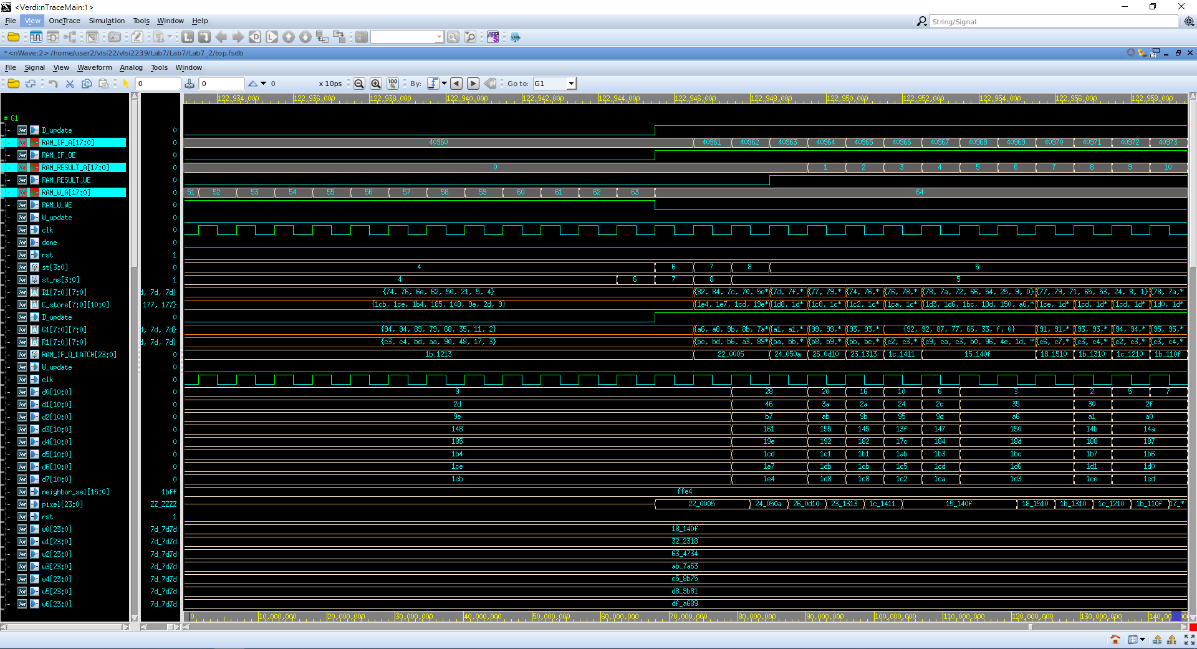
更新完codebook資料接著進到DELAY，這裡設計是因為有delay clk所以多設計一個state來符合傳輸資料的clock。

這裡主要是W\_PIC。

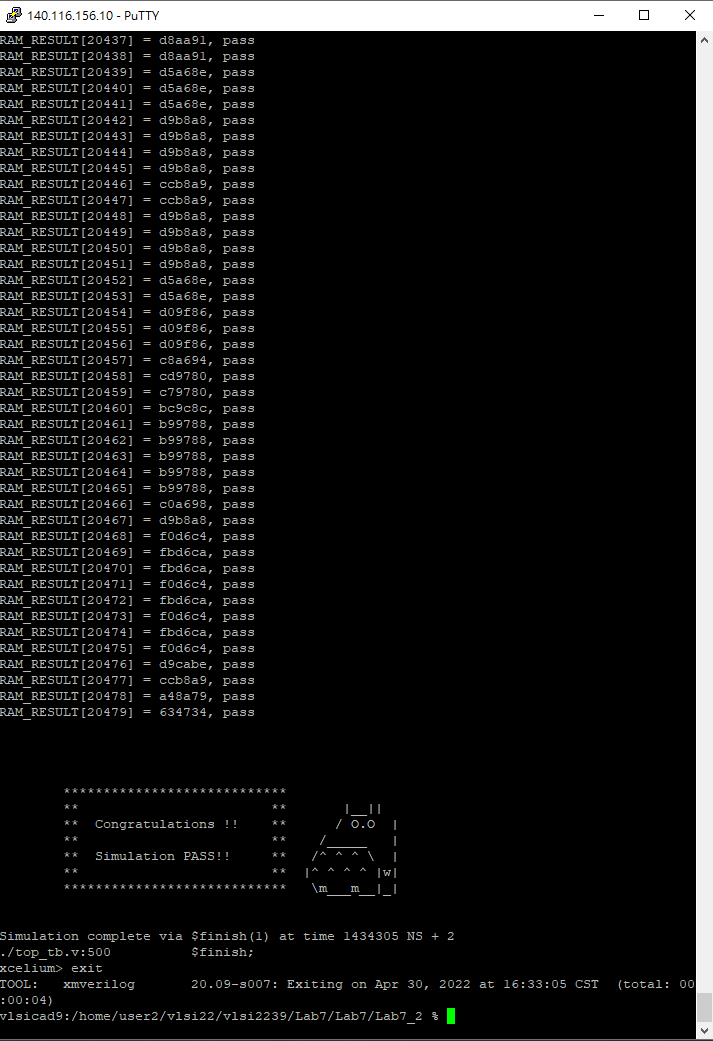


這裡write compress picture result，可以使用pipeline的方式來更新，這裡資料傳遞方式是先從RAM\_IF讀資料到RAM\_IF\_Q\_LATCH然後進到VEP，算calculate manhattan distance，進到MIN\_1比較最小的index，最後進到MIN\_2找中心點然後直接寫入RAM\_RESULT，這裡

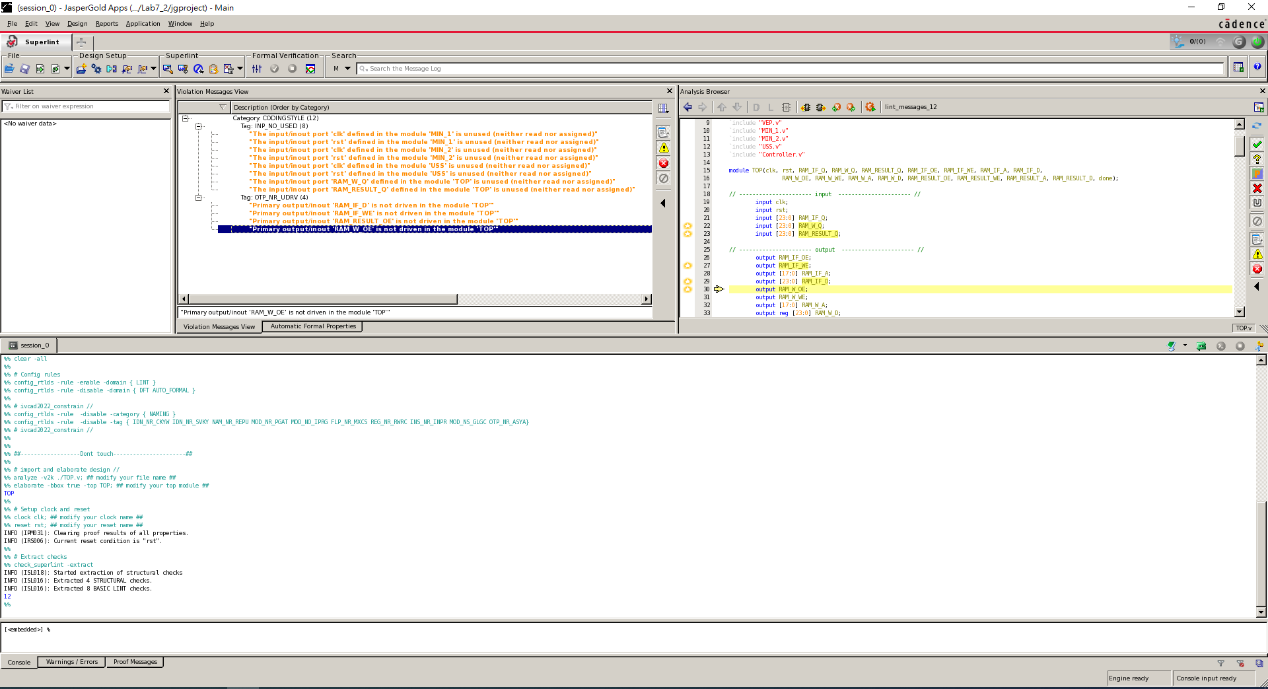
跟1~3 STATE的差別就是不計算neighbor function和更新權重，所以可以在一個clk就完成所有事情



1. Show simulation result



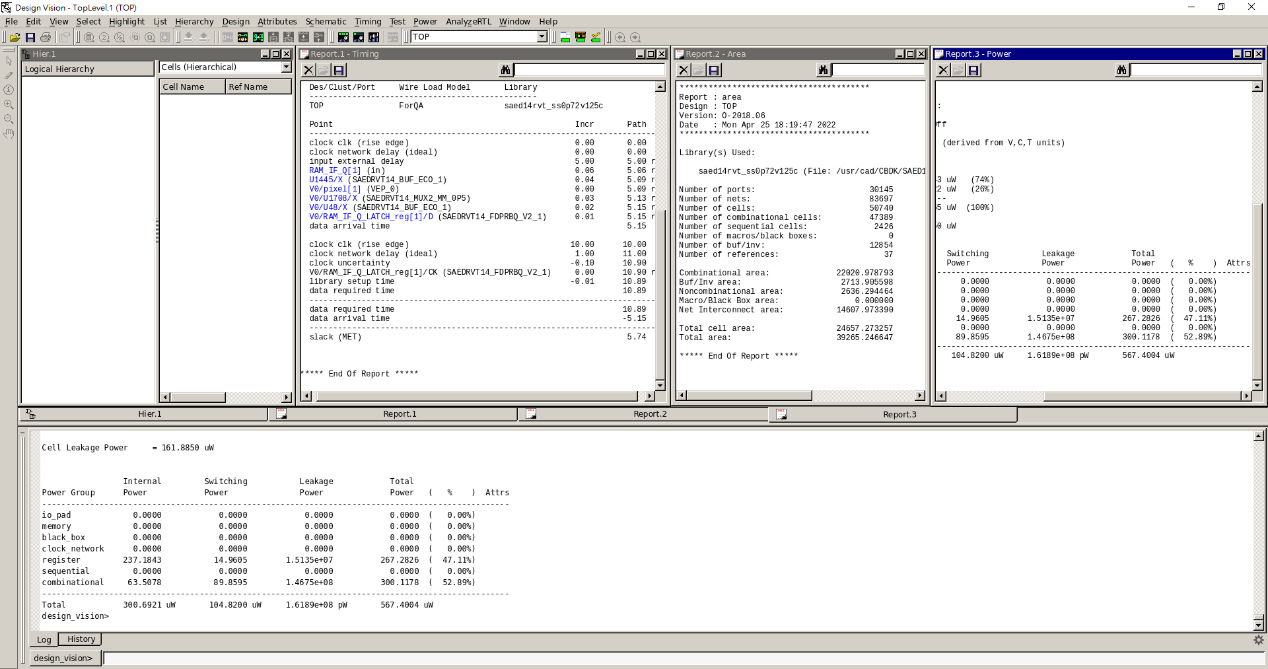
1. Show SuperLint coverage (TOP.v)



99%

都是unused的訊號

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



Clock period:10

Total cell area:24657.273257

Post simulation time:1434305ns

* Lessons learned from this lab

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

一張含有 桌 的圖片

自動產生的描述