National Cheng Kung University

Department of Electrical Engineering

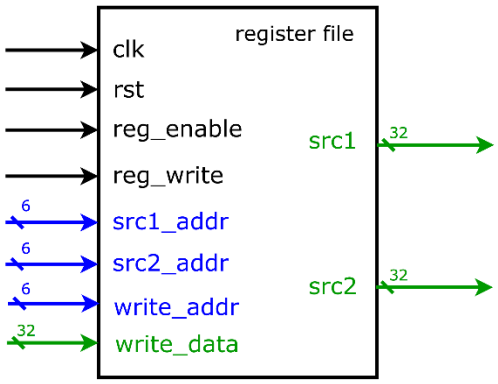
Introduction to VLSI CAD (Spring 2022)

Lab Session 4

Register Files and Manhattan Distance

Yu-Chi Chu

Prob A: Register File



1. Based on the register file structure in LabA, please design a 64 x 32 register file by yourself. *We do not provide any reference code in the file package, but you can refer the code in Lab4 tutorial PDF.*
2. Port list

Signal	Type	Bits	Description
clk	input	1	clock
rst	input	1	reset
reg_enable	input	1	0 → off 1 → on
reg_write	input	1	0 → read 1 → write
src1_addr	input	6	source1 address
src2_addr	input	6	source2 address
write_addr	input	6	write address
write_data	input	32	write data
src1	output	32	read data source1
src2	output	32	read data source2

3. You should follow the file name rules as follow.
   - Register file
     - File name: **regfile.v**
     - Module name: **regfile**
   - Register file testbench
     - File name: **regfile_tb.v**
     - Module name: **regfile_tb**

4. You should verify your code by the following test patterns.

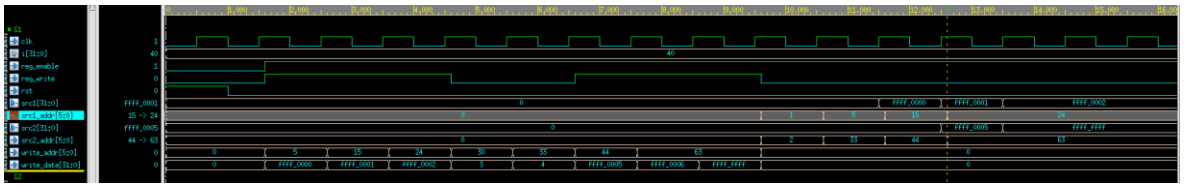
➢ Write data into register file

$time	reg_write	write_addr	write_data
16	1	6’d5	32’hffff_0000
26	1	6’d15	32’hffff_0001
36	1	6’d24	32’hffff_0002
46	0	6’d30	32’hffff_0003
56	0	6’d33	32’hffff_0004
66	1	6’d44	32’hffff_0005
76	1	6’d63	32’hffff_0006
86	1	6’d63	32’hffff_ffff

➢ Read data from register file

$time	reg_write	src1_addr	src2_addr
96	0	6’d1	6’d2
106	0	6’d5	6’d33
116	0	6’d15	6’d44
126	0	6’d24	6’d63

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



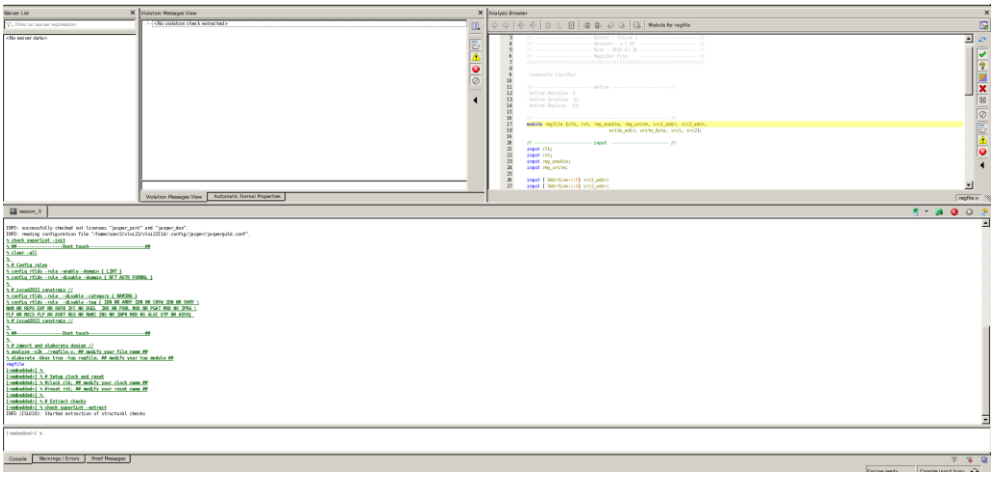
- **Initial State (rst = 1):** When the reset signal (rst) is 1, all registers are cleared.

- **Write Mode (clk 2-10, reg_enable = 1, reg_write = 1):** On the second clock cycle, reg_enable and reg_write are both 1. This enables the write operation, and the value ffff_0000 is written into REG[5]. On the third clock cycle, ffff_0001 is written into REG[15], and so on. This continues until the tenth clock cycle.

- **Read Mode (clk 6, reg_write = 0):** On the sixth clock cycle, reg_write becomes 0, transitioning the circuit to read mode. However, at this point, REG[0] hasn't been written to yet, so both src1 and src2 read 0.

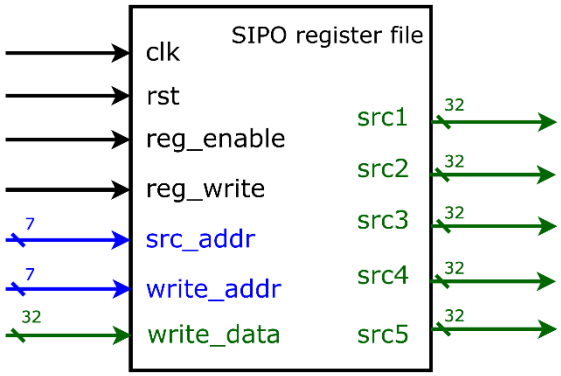
- **Later Read Operation (clk 12):** On the twelfth clock cycle, src1_addr is 15. Therefore, the value stored in REG[15] (which is ffff_0001 from the third clock cycle) is written into src1. The subsequent operations follow a similar pattern.

6. Show SuperLint coverage



Coverage = 1

Prob B: Serial-In Parallel-Out Register File



1. Based on the SIPO register file in LabB, please design a 128 x 32 SIPO register file by yourself. *We do not provide any reference code in the file package, but you can refer the code in Lab4 tutorial PDF.*
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. You should follow the file name rules as follow.
   - SIPO Register file
     - File name: **regfile_sipo.v**
     - Module name: **regfile_sipo**
   - SIPO Register file testbench
     - File name: **regfile_sipo_tb.v**
     - Module name: **regfile_sipo_tb**

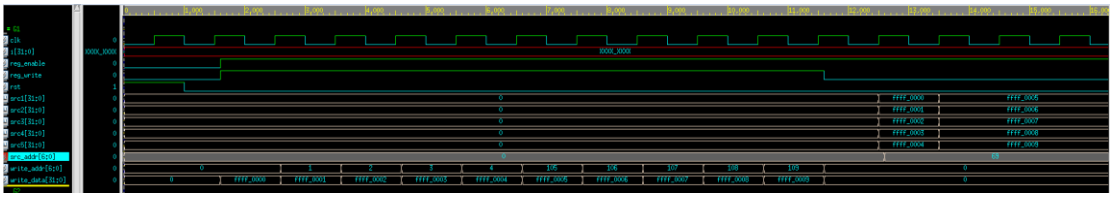
4. You should verify your code by the following test patterns.
   - Write data into register file

reg_write	write_addr	write_data
1	7’d0	32’hffff_0000
1	7’d1	32’hffff_0001
1	7’d2	32’hffff_0002
1	7’d3	32’hffff_0003
1	7’d4	32’hffff_0004
1	7’d105	32’hffff_0005
1	7’d106	32’hffff_0006
1	7’d107	32’hffff_0007
1	7’d108	32’hffff_0008
1	7’d109	32’hffff_0009

   - Read data from register file

reg_write	src_addr
0	7’d0
0	7’d105

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

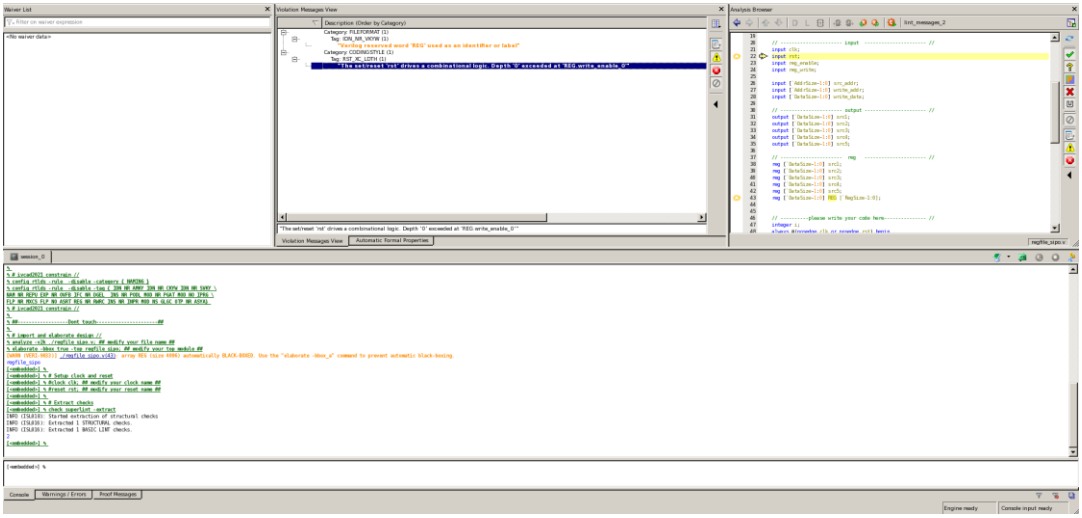


This section describes a similar register file but with a single address input (src_addr).

Write Mode (clk 2-10, reg_write = 1): Similar to the previous example, data is written sequentially to the registers based on the incrementing address. For example, ffff_0000 is written to REG[0] on the second clock cycle, ffff_0001 to REG[1] on the third, and so on.

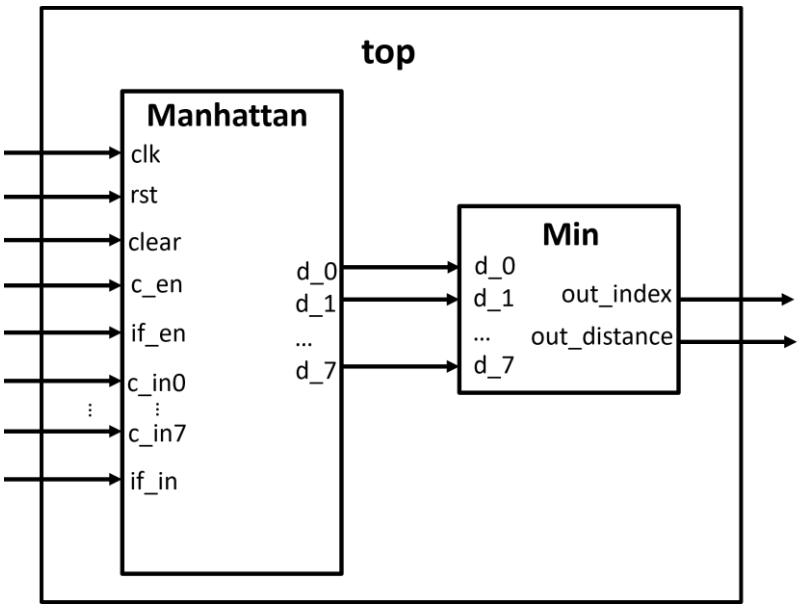
Read Mode (clk 13): On the thirteenth clock cycle, src_addr is 0. Data is read from the registers and written to src_1 through src_5. src_1 reads from REG[src_addr], src_2 reads from REG[src_addr + 1], and so on, until src_1 through src_5 are all loaded. The process then repeats.

6. Show SuperLint coverage



Coverage = 1

Prob C: Finding Smallest Distance



1. Based on ProbC, please design a Manhattan unit and a MIN unit by yourself.

Manhattan unit calculate all distance between c_in0~7 and if_in. The distance calculation method is "$d = |R_i - R_j| + |G_i - G_j| + |B_i - B_j|$". The outputs distances d_0~7 are send to Min unit to find the minimum distance.

Min unit find the minimum distance and output the minimum distance and its index. Notice that you can use the hierarchical tree structure to compare all distances.

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 weight data.
if_in	input	24	Input input feature map data.
d_0~7	output	10 each	Output distance data.

Min:

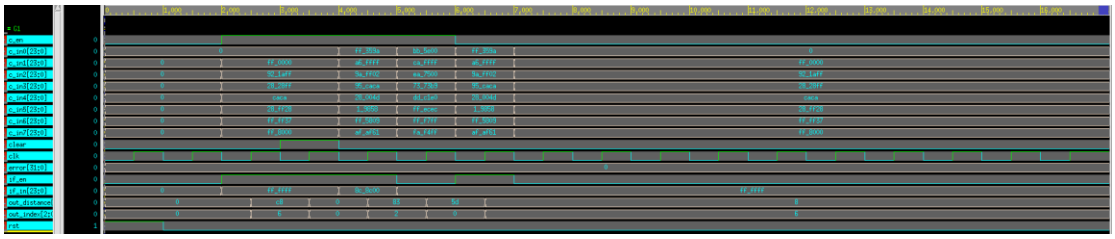
Signal	Type	Bits	Description
d_0~7	input	10 each	Input data.
out_index	output	3	Output index.
out_distance	output	10	Output minimum distance.

3. You should follow the file name rules as follow.

- Manhattan
  - File name: **Manhattan.v**
  - Module name: **Manhattan**
- MIN
  - File name: **Min.v**
  - Module name: **Min**
- Top
  - File name: **top.v**
  - Module name: **top**
- Testbench
  - File name: **top_tb.v**
  - Module name: **top_tb**

4. There should be a register file in Manhattan module so that the outputs delay half cycle and meets the testbench requirement. But the Min module is totally combinational.

5. Show waveforms to explain that your Manhattan and Min module are correct.



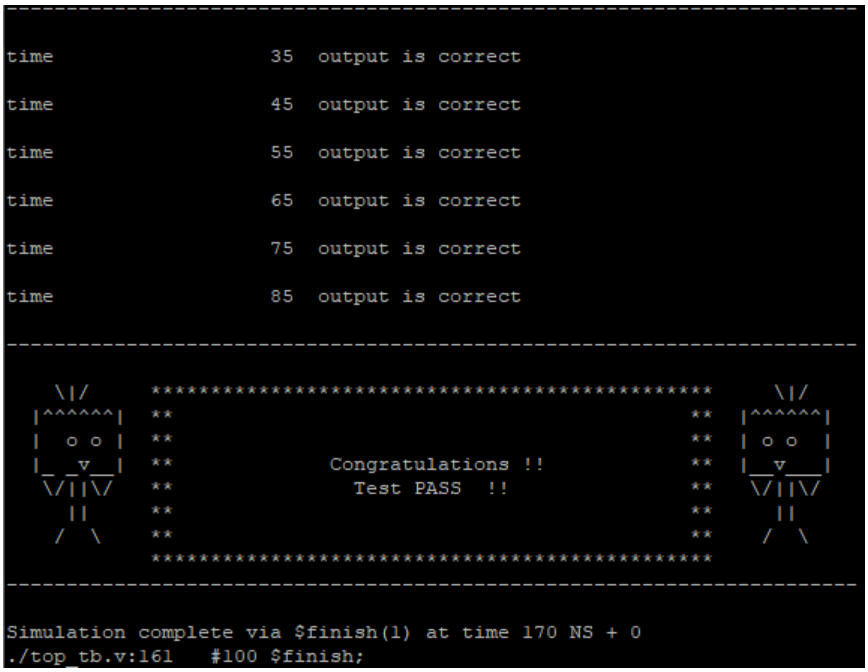
Initial State (rst = 1): When rst is 1, all registers in the Manhattan distance calculation circuit are cleared. Both distance and index are initialized to 0.

Data Input and Output (clk 3, if_en = 1, c_en = 1): Starting from the third clock cycle, when both if_en (input enable) and c_en (calculate enable) are 1, data is loaded into the circuit. The calculated out_distance and out_index are then outputted.

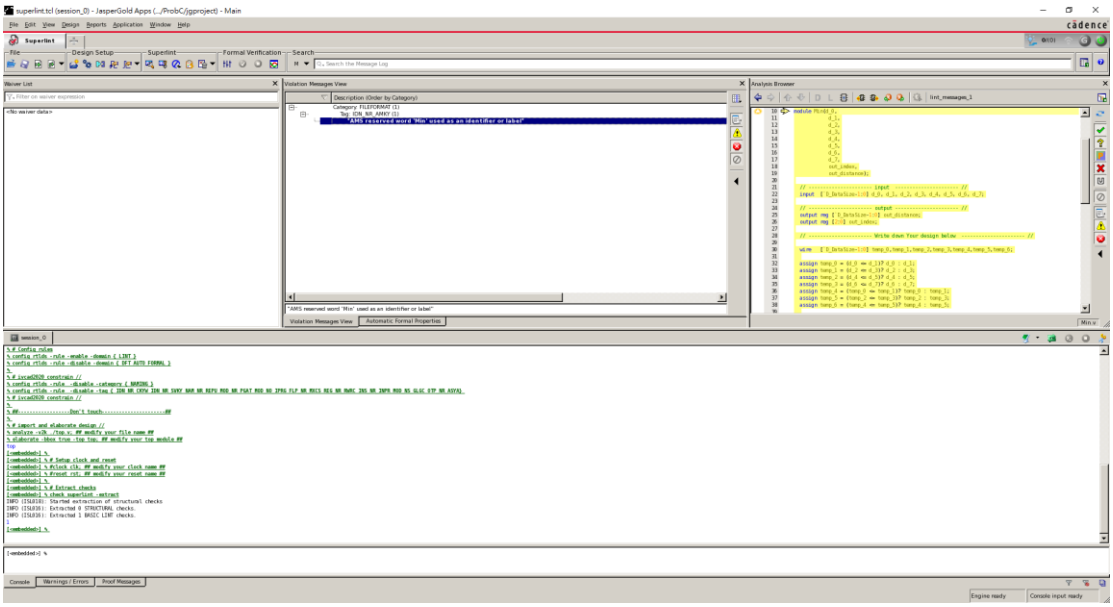
Clear Operation (clk 4, clear = 1): On the fourth clock cycle, clear is 1. This clears all registers in the Manhattan distance calculation circuit, resetting the outputs to 0.

Stable Output (clk 8 onwards, if_en = 0): From the eighth clock cycle onwards, if_en remains 0. This means the input data (if_in) remains constant. Also, c_en is 0, so the calculated values (c_in) also remain constant. Consequently, the output distance and index values remain unchanged.

6. Show the simulation result on the terminal.



7. Show SuperLint coverage



Coverage=1

Appendix A : Commands we will use to check your homework

Prob	Syn	command
A	pre	ncverilog regfile_tb.v +define+FSDB +access+r
B	pre	ncverilog regfile_sipo_tb.v +define+FSDB +access+r
C	pre	ncverilog top_tb.v +define+FSDB +access+r