2025 Digital IC Design

Midterm Exam

Meeting Room Number：3

Please make sure you enter the exact meeting room. Each meeting room has different exam questions. Tas will verify your code according to your student id, the mismatch between student id and meeting room number may cause you to lose part of the score.

Question 1: Six-Input Median Finder

Median finder is a combinational digital circuit that find the median value among a sequence of numbers. In this question, you are requested to design a 6-input median finder, which find the median value among 6 input numbers. The median finder could be constructed by two input Comparator2 (C2) module. The specifications and functions of the C2 and Median Finder are detailed in the following sections.

1.1. Dual-Input Comparator2:

The Comparator2 is a basic module that takes two 4-bit numbers as inputs and outputs their minimum and maximum values. That is, ***max*** will be assigned the maximum value among ***A***, ***B***; while ***min*** will be assigned the minimum value.

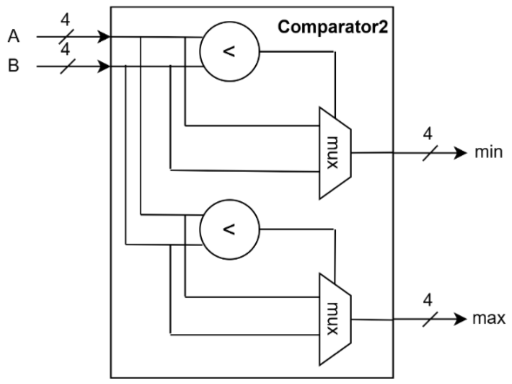


Table I. I/O interface of the C2 module

Signal Name	I/O	Width
A	I	4
B	I	4
min	O	4
max	O	4

1.2. Six-Input Median Finder:

The specification and I/O interface of the six-input Median Finder for this question is listed in Table II. The six-input Median Finder module takes six 4-bit input numbers and finds the median value. In this question, you must construct the six-input Median Finder circuit with your dual-input Comparator2 modules.

Table II. I/O interface of the six-input Median Finder

Signal Name	I/O	Width	Description
num1 ~ 6	I	4	Input number 1 ~ 6.
median	O	4	The median value of the six input numbers

Please design the six-input Median Finder with your Comparator2 modules. Otherwise, no score will be given. The testbench will call these two modules to validate the function of the circuit.

1.3 File Description:

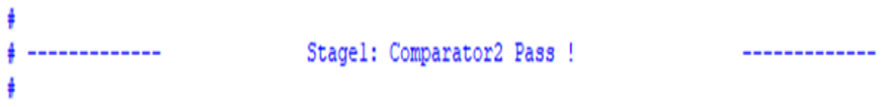
File Name	Description
Comparator2.v	The module of dual-input comparator.
MedianFinder_6num.v	The module of 6-input Median Finder.
testfixture.sv	The testbench file. The content in this file is **not allowed** to be modified.
golden_c2.dat	Test data and golden data for C2 verification.
golden_mf6.dat	Test data and golden data for 6-input Median Finder verification.

1.4 Scoring of Question 1 [50%]

The scoring is fully based on the functional simulation results in Modelsim. There is no necessary to synthesis the Verilog codes.

1.4.1: Stage1: Comparator2 [10%]

All of the results should be generated correctly, and you will get the following message in ModelSim simulation.



```
#
# -------------          Stage1: Comparator2 Pass !          -------------
#
```

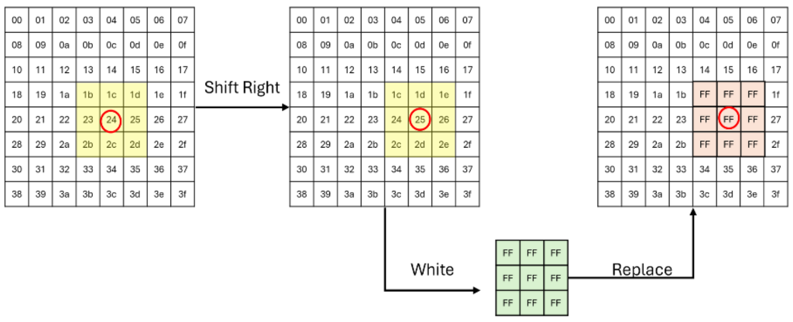
1.4.2 Stage2: Six-Input Median Finder [40%]

All of the results should be generated correctly, and you will get the following message in ModelSim simulation.

```
-------------              Stage2: MedianFinder_6num Pass !          -------------

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

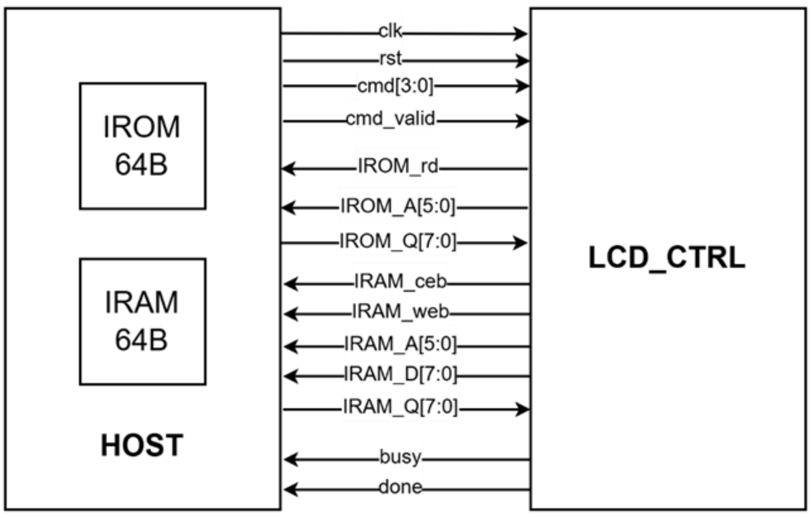
Question 2: LCD Controller



The task requires completing the Image Display Control Circuit (LCD_CTRL circuit). The input grayscale image is stored in the input image ROM module (IROM) on the Host side. The LCD_CTRL circuit must read grayscale image data from the IROM memory module on the Host side and perform the corresponding operations as required.

After processing, the results must be written to the output image RAM module (IRAM) on the Host side. Once the entire image processing is complete, the done signal should be set High, and the system will then verify the correctness of the processed image data. The circuit signal definitions and LCD_CTRL operation methods are detailed in the following sections.

2.1 Block Overview



2.2 I/O Interface

Table III. I/O interface of the LCD_CTRL module.

Signal Name	I/O	Width	Description
clk	Input	1	System clock signal. System should be triggered by the positive edge of clock.
rst	Input	1	System reset signal. Active high, asynchronous reset.
cmd	Input	4	Command input signal. This controller supports a total of **fifteen command inputs**. For detailed command descriptions, please refer to **Table IV**. A command input is considered **valid** only when **"cmd_valid" is high** and **"busy" is low**.
cmd_valid	Input	1	When this signal is **high**, indicates "**cmd**" input is valid.
IROM_rd	Output	1	Image ROM memory read enable signal. When **high**, indicates the **LCD_CTRL** requests data from the **Host**.
IROM_A	Output	6	Image ROM address bus. The **LCD_CTRL** uses this bus to request grayscale image data from the corresponding address in the **Host's read-only memory**.
IROM_Q	Input	8	Image ROM data bus. **Host** uses this bus to send grayscale image data from the **Host's read-only memory** to the **LCD_CTRL**.
IRAM_ceb	Output	1	Image RAM chip enable signal. When **high**, indicates the **Image RAM** is available for read and write operations.
IRAM_A	Output	6	Image RAM address bus. The **LCD_CTRL** uses this bus to specify the address in the **Host's Image RAM** where the grayscale image data should be read or written.
IRAM_D	Output	8	Image RAM input data bus. The **LCD_CTRL** uses this bus to write image data into the **Host's Image RAM**.
IRAM_web	Output	1	Image RAM read/write select signal. When **High**, it indicates that **LCD_CTRL** is reading data from **Image RAM**; otherwise, it indicates writing data to **Image RAM**.
IRAM_Q	Input	8	Image RAM output data bus. The **Host's Image RAM** uses this bus to output image data to the **LCD_CTRL**.
busy	Output	1	System busy signal. When this signal is **high**, indicates the controller is currently executing the **current command** and cannot accept new command inputs. When this signal is **low**, the system is ready to receive new commands.
done	Output	1	When the controller completes writing to **IRAM**, it sets **done** to **high** to indicate completion.

2.3 File Description

File Name	Description
LCD_CTRL.v	The module of rails, which is the top module in this design.
testfixture.v	The testbench file. You can only change the test pattern number.
image1.dat	Test image for LCD_CTRL.
cmdX.dat	Test command X for LCD_CTRL.
goldenX.dat	Golden data X for LCD_CTRL verification.

2.4 Function Description

2.4.1 Description

The controller must process user input commands to determine the display coordinates (origin) and data parameters, enabling functions such as shifting and averaging. The original 8*8 grayscale image is stored in the off-chip IROM, the

LCD controller can fetch the image from IROM through related buses. Once the image processing operations are completed, the processed image data is written to the off-chip IRAM through related buses and pull done signal high to let Host know the task is completed.

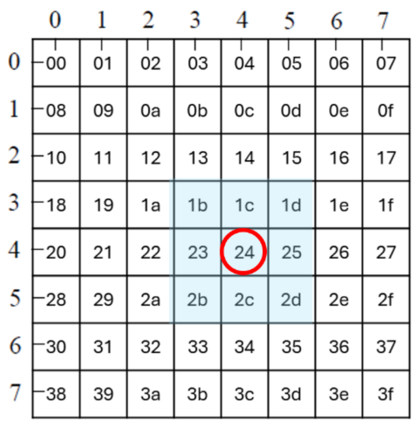
- **Operating Point Definition:**

  The operation point refers to a specific coordinate in the image data. The four neighboring pixels (top, bottom, left, and right) surrounding the operation point are considered operational image data, which the controller uses for processing.

  In this task, the coordinate system for the input image is predefined:

  - The horizontal direction (X-axis) represents the width.
  - The vertical direction (Y-axis) represents the height.
  - The coordinate range for both X and Y axes is 0 to +8.
  - To ensure that the operational image data does not exceed the image boundaries, the operation point is restricted to the range X = +1 to +7 and Y = +1 to +7.

  Based on this coordinate system, the display shift (Shift) function must be designed to adjust the output image accordingly.



* The default Operating Point is (4, 4)

- **Command Definition:**

Table IV. Command table

Command	Definition
0(0000)	Write
1(0001)	Shift Up
2(0010)	Shift Down
3(0011)	Shift Left
4(0100)	Shift Right
5(0101)	Max
6(0110)	Min
7(0111)	Average

8(1000)	White
9(1001)	Black
10(1010)	Rotate
11(1011)	Restore
12(1100)	Copy
13(1101)	Paste
14(1110)	-
15(1111)	Finish

- **Write**

  When executing the Write command, the controller writes image data into IRAM from left to right and top to bottom.

- **Shift Up**

  Activate the Shift Up mode. This will decrease the Y-coordinate of the operation point by 1, but the Y-coordinate cannot be lower than 1. When the Y-coordinate is 1, if an additional Shift Up command is received, the operation point will remain unchanged.

- **Shift Down**

  Activate the Shift Down mode. This will increase the Y-coordinate of the operation point by 1, but the Y-coordinate cannot exceed 7. When the Y-coordinate is 7, if an additional Shift Down command is received, the operation point will remain unchanged.

- **Shift Left**

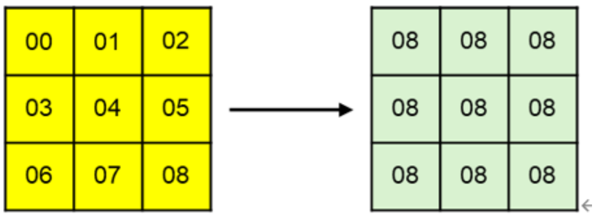
  Activate the Shift Left mode. This will decrease the X-coordinate of the operation point by 1, but the X-coordinate cannot be lower than 1. When the X-coordinate is 1, if an additional Shift Left command is received, the operation point will remain unchanged.

- **Shift Right**

  Activate the Shift Right mode. This will increase the X-coordinate of the operation point by 1, but the X-coordinate cannot exceed 7. When the X-coordinate is 7, if an additional Shift Right command is received, the operation point will remain unchanged.

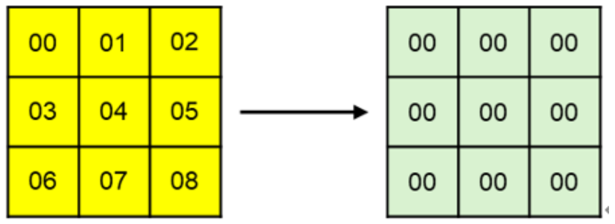
- **Max**

  Perform the calculation of the maximum value for the image pixels at the current operation point coordinates. This means comparing the nine corresponding image pixels at the current coordinates and determining the maximum value. Then, replace these nine original image pixels with the calculated maximum value.



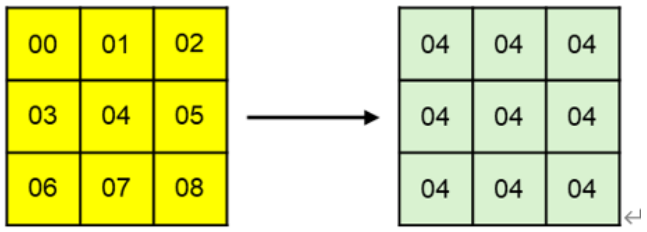
- **Min**

  Perform the calculation of the minimum value for the image pixels at the current operation point coordinates. This means comparing the nine corresponding image pixels at the current coordinates and determining the minimum value. Then, replace these nine original image pixels with the calculated minimum value.



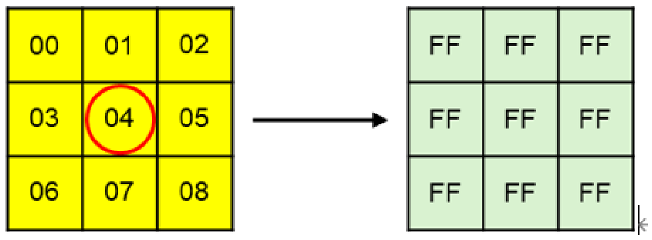
- **Average**

  Perform the calculation of the average value for the image pixels at the current operation point coordinates. This means comparing the nine corresponding image pixels at the current coordinates and determining the average value. Then, replace these nine original image pixels with the calculated average value.



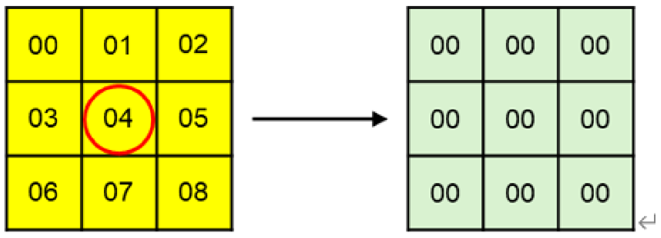
- **White**

  Replace these nine original image pixels with white pixel (0xFF).



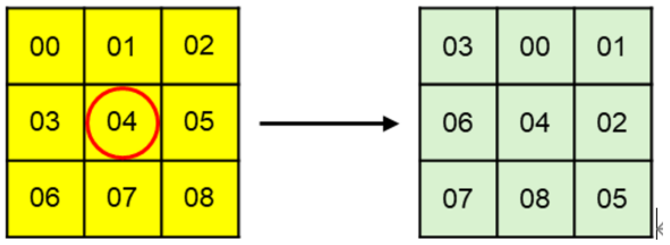
- **Black**

  Replace these nine original image pixels with black pixel (0x00).



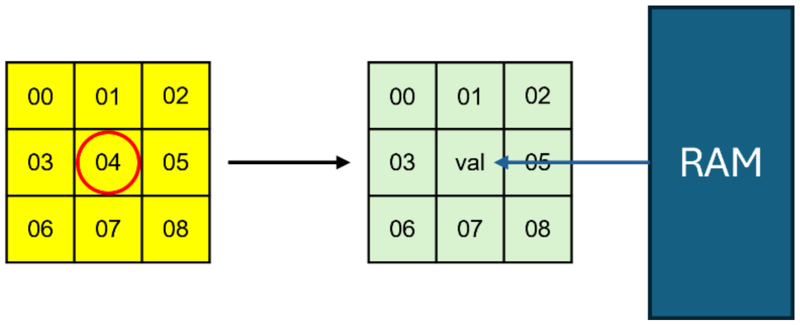
- **Rotate**

  Rotate these nine original image pixels clockwise according to the center pixel.



- **Restore**

  Restore the corresponding address pixel from IRAM and replace it with the original pixel.

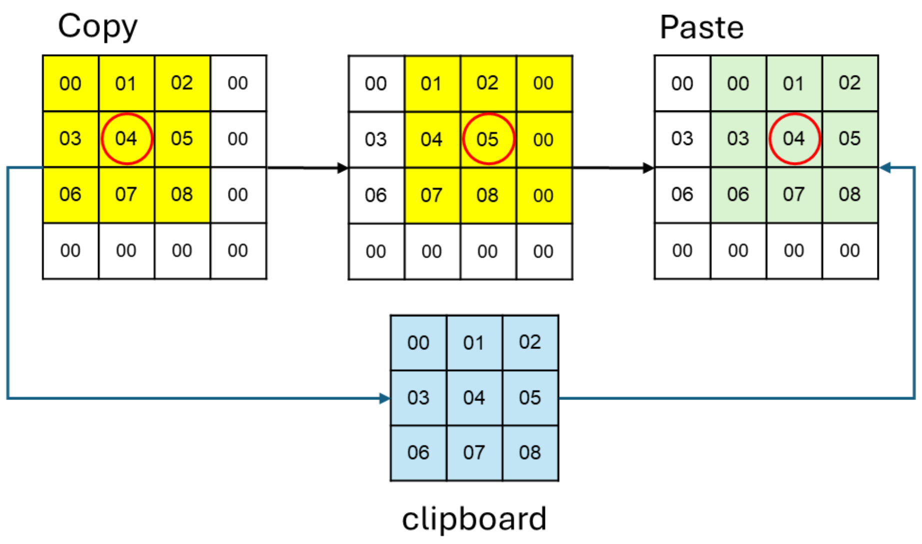


- **Copy**

  Copy these nine image pixels according to the operating point into clipboard.

- **Paste**

  Paste these nine image pixels according to the operating point from clipboard.

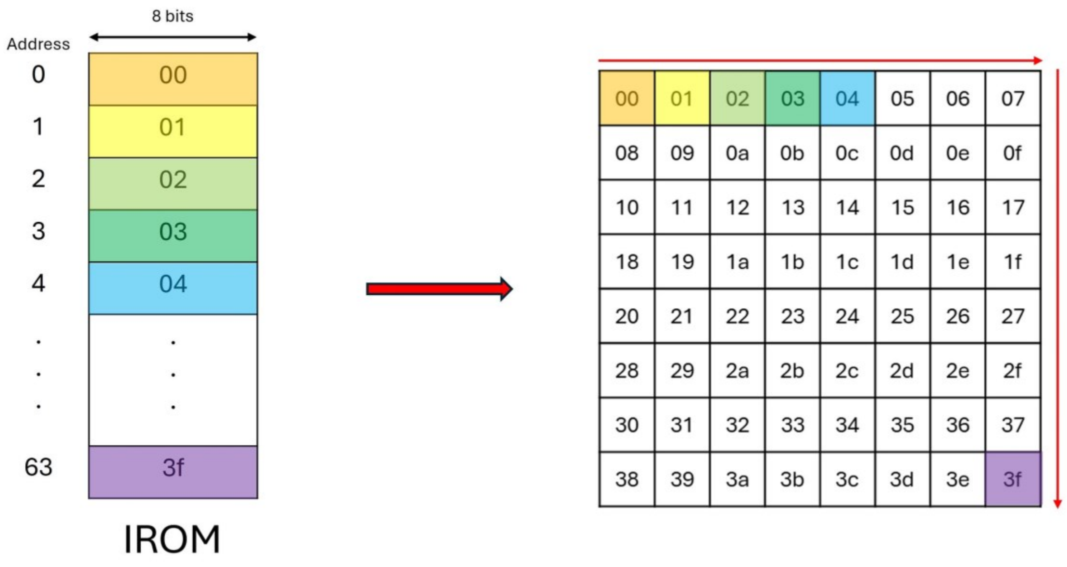


- **Finish**

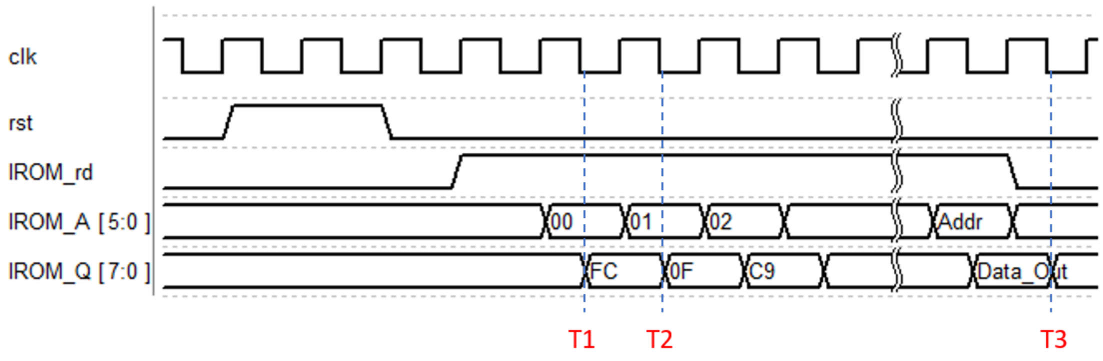
  When executing the Finish command, the LCD_CTRL pulls up **done** signal to high.

➢ **IROM Mapping Method and Timing Specification:**

The input raw grayscale image has a fixed size of 8x8 pixels, with each pixel consisting of 8 bits of data. Therefore, the Host-side grayscale image contains a total of 64 pixels. The IROM memory has a data width of 8 bits and 64 addresses. Each address stores 8 bits of data, which corresponds to the grayscale image data of one pixel. The mapping is as shown in the following diagram: The input raw grayscale image has a fixed size of 8x8 pixels, with each pixel consisting of 8 bits of data. Therefore, the Host-side grayscale image contains a total of 64 pixels. The IROM memory has a data width of 8 bits and 64 addresses. Each address stores 8 bits of data, which corresponds to the grayscale image data of one pixel. The mapping is as shown in the following diagram:

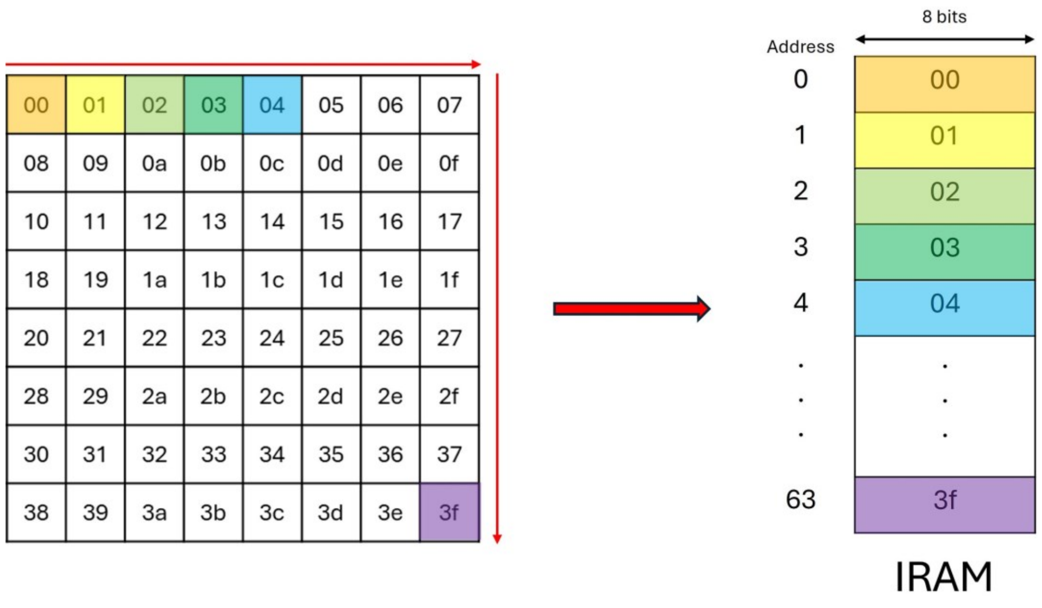


The timing of the IROM memory is as shown in the diagram below. The IROM will, on each falling edge of the clock signal, if the IROM_rd signal is High (as in T1 and T2 time periods in the diagram), immediately send the data from the address specified by the IROM_A signal to the LCD_CTRL end via the IROM_Q bus. If the IROM_rd signal is Low (as in T3 time period in the diagram), the IROM will not perform any action. No read delay is considered for this memory.



➢ **IRAM Mapping Method and Timing Specification:**

The output grayscale image after computation by the LCD_CTRL is 8x8 pixels, with each pixel consisting of 8 bits of data. Therefore, the Host-side output result grayscale image memory (IRAM) has a total of 64 addresses, which are used to store the processing results for each pixel. The mapping is as shown in the following diagram:



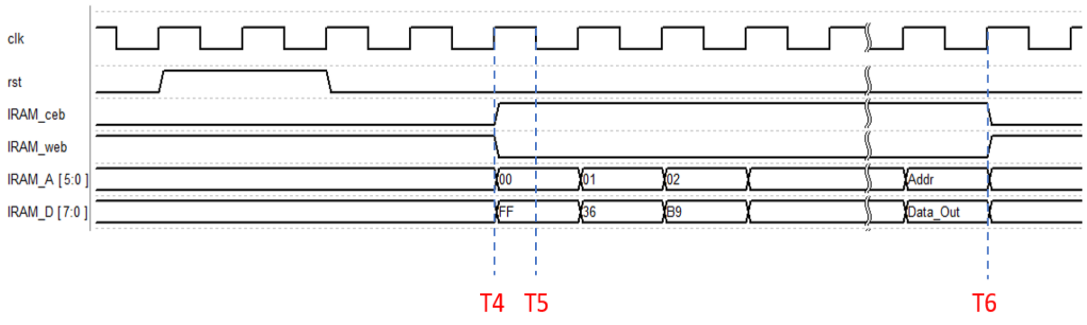
00	01	02	03	04	05	06	07
08	09	0a	0b	0c	0d	0e	0f
10	11	12	13	14	15	16	17
18	19	1a	1b	1c	1d	1e	1f
20	21	22	23	24	25	26	27
28	29	2a	2b	2c	2d	2e	2f
30	31	32	33	34	35	36	37
38	39	3a	3b	3c	3d	3e	3f

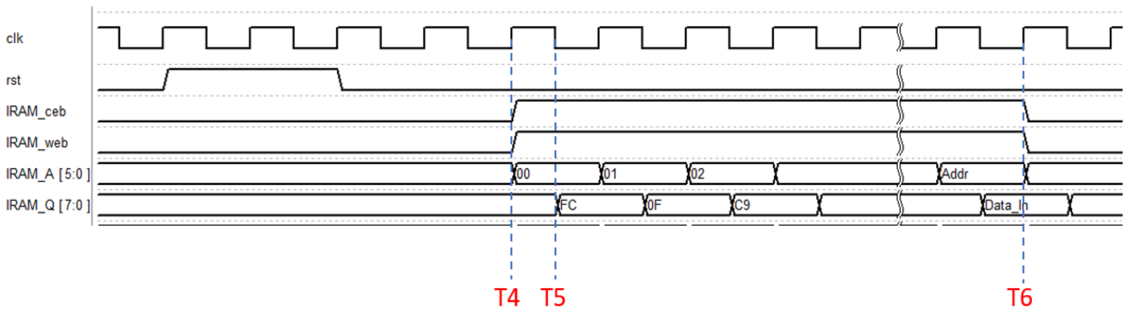
The timing for the output result grayscale image memory (IRAM) is as shown in the following diagram. The IRAM will, on each falling edge of the clock signal, if the IRAM_ceb signal is High (as in T4 time), determine the operation based on the IRAM_web signal.

- If IRAM_web is High (as the upper diagram), IRAM will output the data corresponding to the IRAM_A address from the IRAM_Q.

- If IRAM_web is Low (as the lower diagram), it will write the data from IRAM_D to the memory location corresponding to the IRAM_A address.

The HOST will trigger the write or read operation on the falling edge of the clock signal at T5 and stop when IRAM_ceb be pulled down at T6.

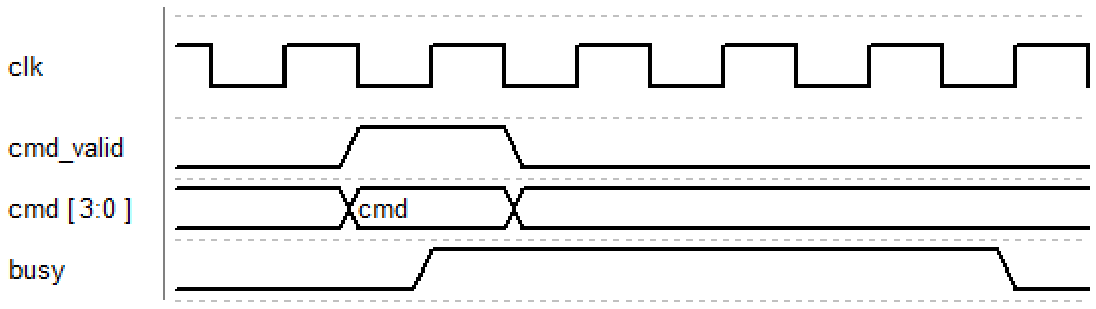




2.4.2 Timing Specifications

The timing specification diagrams for other control commands (Shift up, Shift down, Shift left, Shift right, Max, Min, Average, …) are as shown in the following diagram.

Throughout the entire processing operation, the **busy** signal remains **high**. Once the output is completed, the **busy** signal is set **low** to allow the system to accept new command inputs.

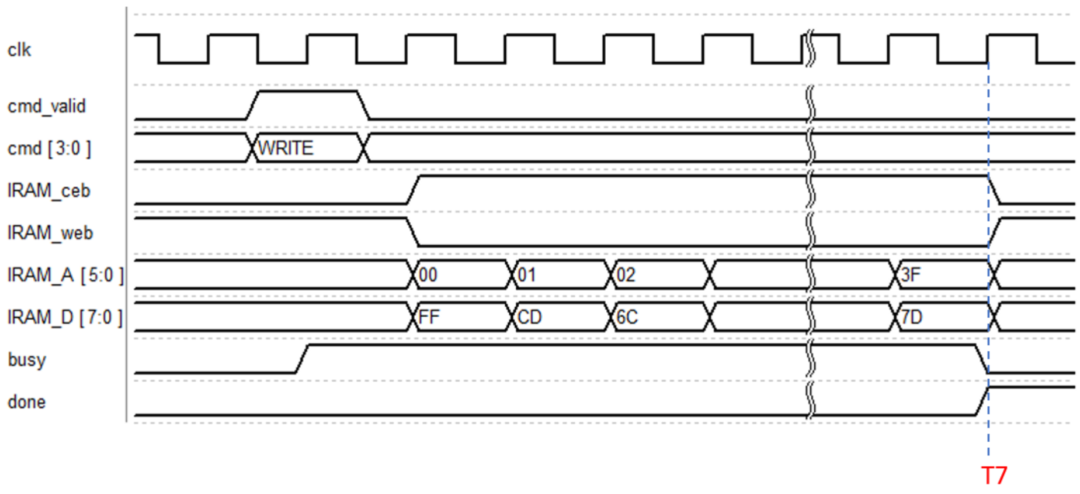


The timing specification diagram for the **write** command is shown in the following diagram.

- When executing the **write** command, the controller writes the processed image data into **IRAM**.
- When **IRAM_ceb** is **high** and **IRAM_web** is **low**, it indicates a write operation to **IRAM**. At this moment, the address signal can be provided to

store the image data into **IRAM**.

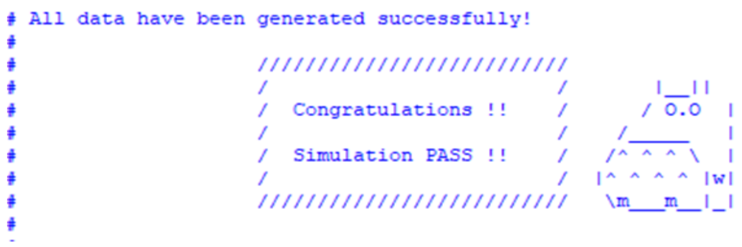
- Throughout the entire processing operation, the **busy** signal remains **high**. Once the output is completed, the **busy** signal is set back to **low** to allow the system to accept new command inputs.
- After the write operation is completed, the **done** signal is set to **high** at T7, indicating that the write process is finished. At this point, the test fixture will compare the data written to **IRAM** with the golden pattern for verification.



2.5 Scoring of Question 2 [50%]

The scoring is fully based on the functional simulation results in Modelsim. There is no necessary to synthesis the Verilog codes. Please don’t design specifically for the test pattern. Otherwise, you will get 0 point.

There are three test patterns in this question, testbench will verify if the outputs are correctly generated. If all the results are correct, you will get the following message in ModelSim simulation.



```
# All data have been generated successfully!
#
#                    //////////////////////////
#                    /                        /       |__||
#                    /  Congratulations !!    /      / O.O  |
#                    /                        /     /_____  |
#                    /  Simulation PASS !!    /    /^ ^ ^ \  |
#                    /                        /   |^ ^ ^ ^ |w|
#                    //////////////////////////    \m___m__|_|
#
```

The scoring of the three test patterns is as following：

- **Pattern 0 [10%]**
- **Pattern 1 [20%]**
- **Pattern 2 [20%]**

Submission:

1. Submitted files

You should classify your files into two directories and compress them to .zip format. The naming rule is StudentID.zip. If your file is not named according to the naming rule, you will lose five points. Please submit your .zip file to folder “Meeting Room X Submission” in Moodle according to your meeting room number X.

	StudentID.zip
Q1	
*.v	All of your Verilog RTL code for Question 1
Q2	
*.v	All of your Verilog RTL code for Question 2

2. Notes

Deadline: 2025/4/21 12:00.

No late submissions will be accepted, please pay attention to the deadline.