Lab5: Median filter

Instructor: Lih-Yih Chiou

Speaker: Jay

Date: 2024/03/21

EE

LPHPLAB
VLSI Design LAB

Goal

- In Lab5, you will learn
  - How to read/write the memory.
  - How to implement a simple image processing algorithm with a more complex FSM.

Outline

- Introduce to image filter
  - Mean filter (Only introduce)
  - **Median filter (This Lab)**
- Hardware description
  - Block diagram
  - I/O Information
  - Memory mapping
  - Flow in Lab5
- Criteria
  - Simulation Result
  - Grading policy
  - Requirement & file format

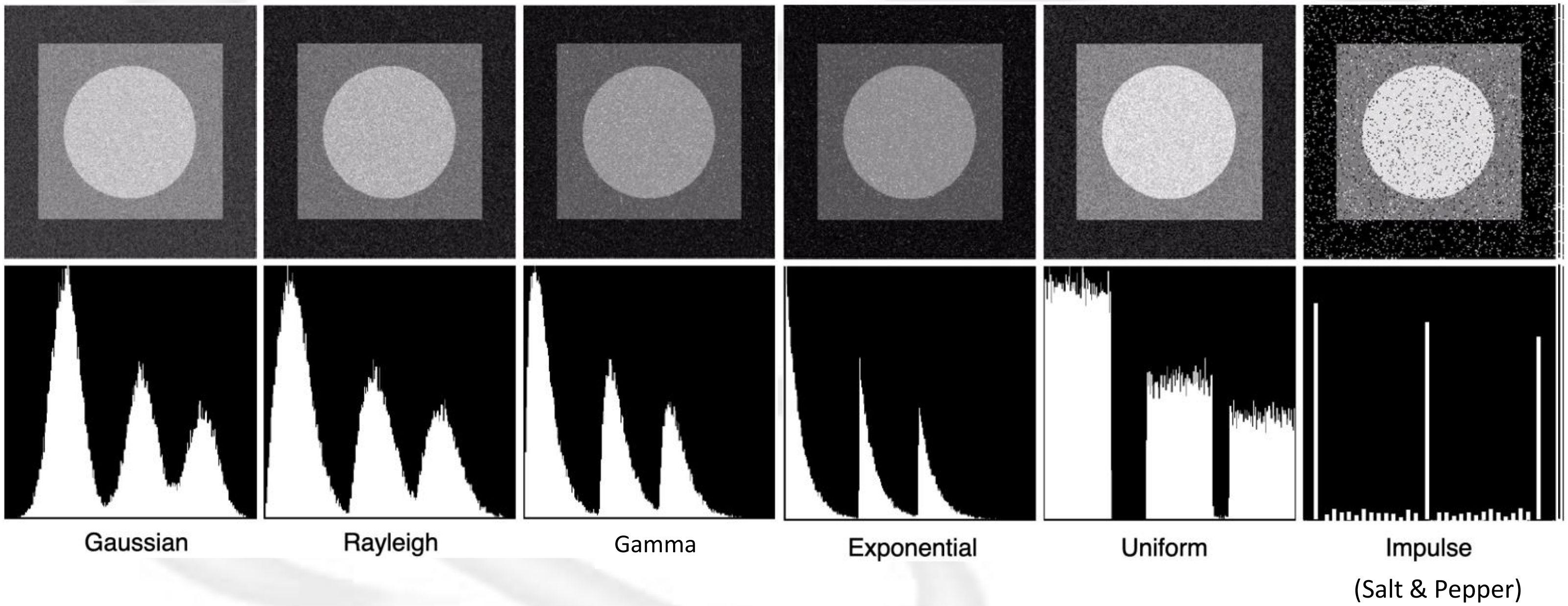
Introduce to image filter

□ Noise and Images

- Principal sources of noise in digital images: during image acquisition, during image transmission.
- Image acquisition: image sensor might produce noise because of environmental conditions or quality of sensing elements.
- Image transmission: interference in the channel.

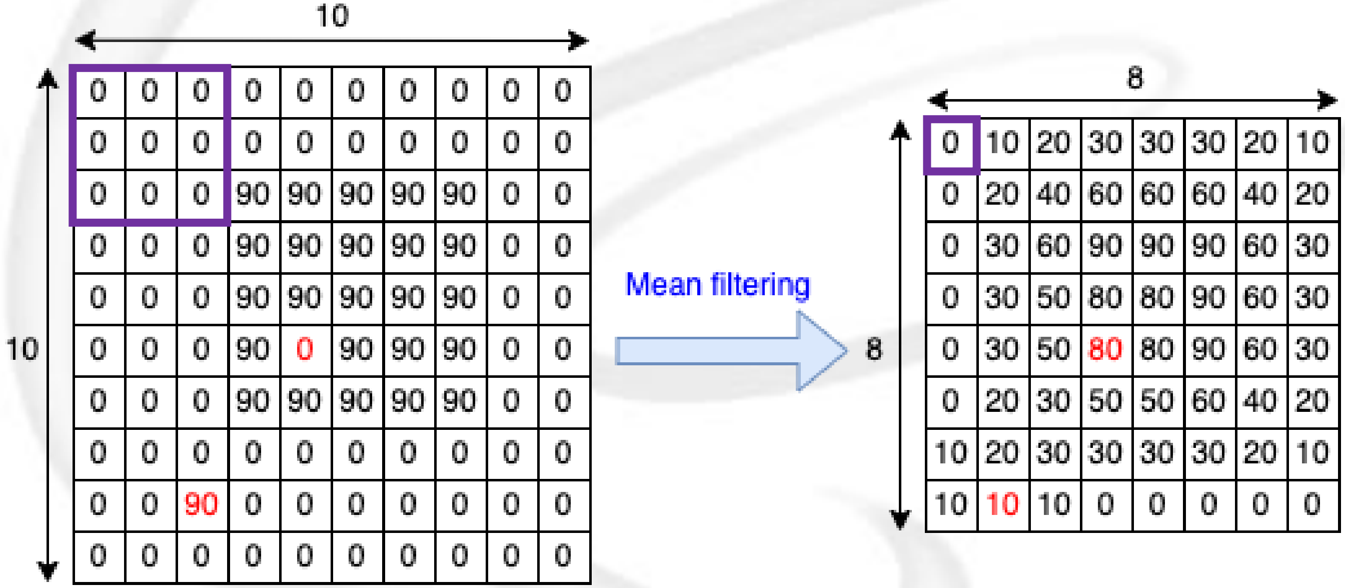
Introduce to image filter

- Different types of image noise
  - Most common noise found in image processing: Gaussian noise, Rayleigh noise, Gamma noise, Exponential noise, Uniform noise, Impulse noise



Introduce to image filter

- Filtering to Remove Noise
  - We can use spatial filters of different kinds to remove different kinds of noise
- Mean filter
  - Taking the average of pixel values within a fixed region(ex:3X3 window)
  - This is implemented as the simple smoothing filter Blurs the image to remove noise

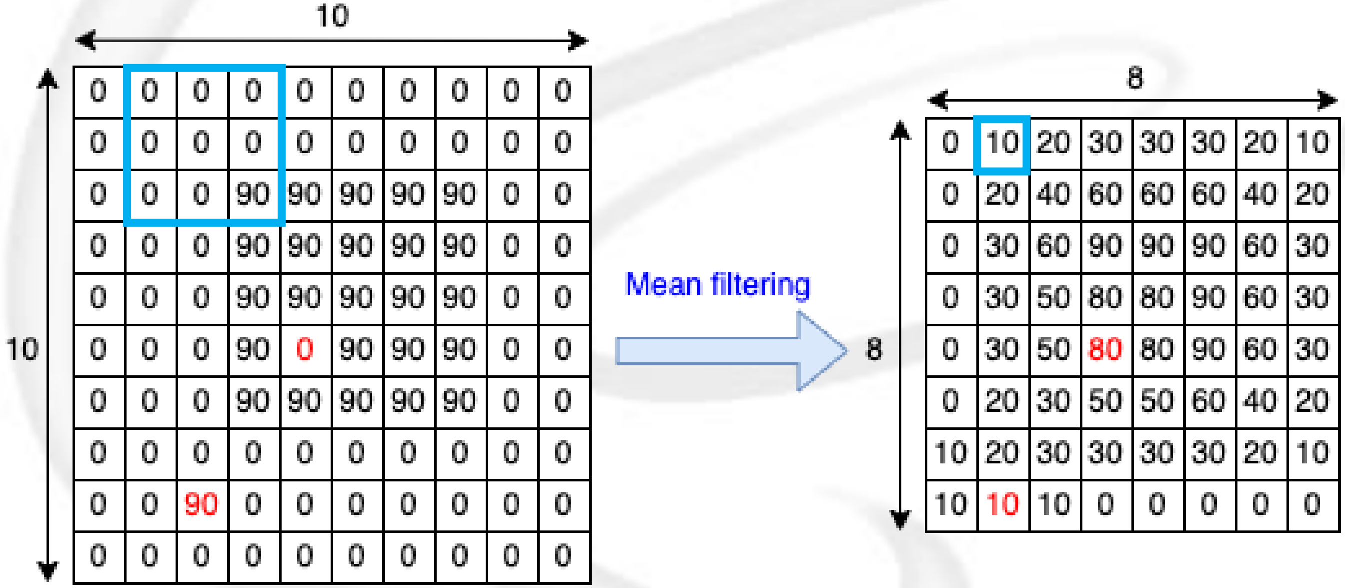


0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	10	20	30	30	30	20	10
0	20	40	60	60	60	40	20
0	30	60	90	90	90	60	30
0	30	50	80	80	90	60	30
0	30	50	80	80	90	60	30
0	20	30	50	50	60	40	20
10	20	30	30	30	30	20	10
10	10	10	0	0	0	0	0

Introduce to image filter

- Filtering to Remove Noise
  - We can use spatial filters of different kinds to remove different kinds of noise
- Mean filter
  - Taking the average of pixel values within a fixed region(ex:3X3 window)
  - This is implemented as the simple smoothing filter Blurs the image to remove noise

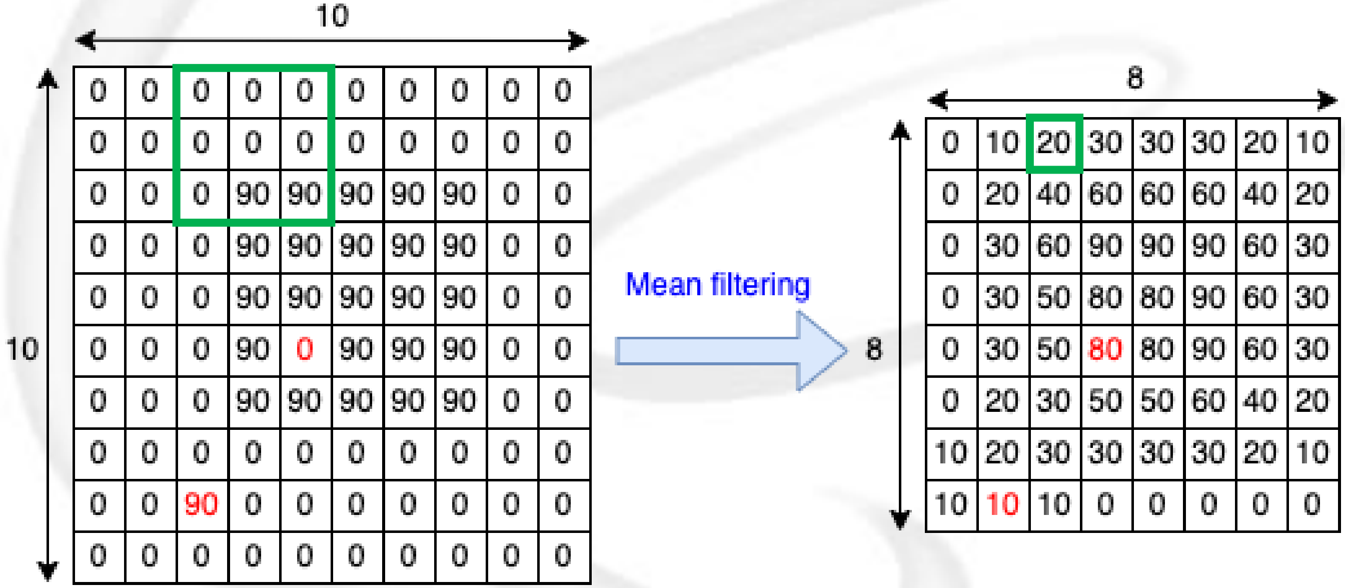


0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	10	20	30	30	30	20	10
0	20	40	60	60	60	40	20
0	30	60	90	90	90	60	30
0	30	50	80	80	90	60	30
0	30	50	80	80	90	60	30
0	20	30	50	50	60	40	20
10	20	30	30	30	30	20	10
10	10	10	0	0	0	0	0

Introduce to image filter

- Filtering to Remove Noise
  - We can use spatial filters of different kinds to remove different kinds of noise
- Mean filter
  - Taking the average of pixel values within a fixed region(ex:3X3 window)
  - This is implemented as the simple smoothing filter Blurs the image to remove noise

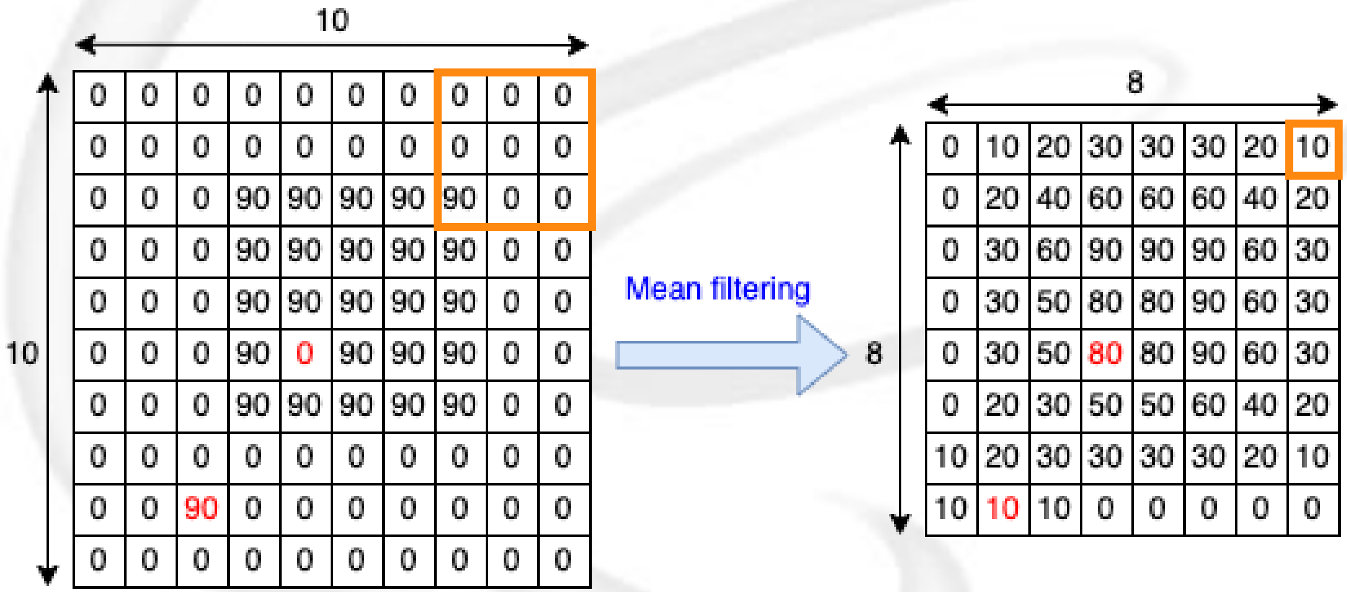


0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	10	20	30	30	30	20	10
0	20	40	60	60	60	40	20
0	30	60	90	90	90	60	30
0	30	50	80	80	90	60	30
0	30	50	80	80	90	60	30
0	20	30	50	50	60	40	20
10	20	30	30	30	30	20	10
10	10	10	0	0	0	0	0

Introduce to image filter

- Filtering to Remove Noise
  - We can use spatial filters of different kinds to remove different kinds of noise
- Mean filter
  - Taking the average of pixel values within a fixed region(ex:3X3 window)
  - This is implemented as the simple smoothing filter Blurs the image to remove noise



0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	10	20	30	30	30	20	10
0	20	40	60	60	60	40	20
0	30	60	90	90	90	60	30
0	30	50	80	80	90	60	30
0	30	50	80	80	90	60	30
0	20	30	50	50	60	40	20
10	20	30	30	30	30	20	10
10	10	10	0	0	0	0	0

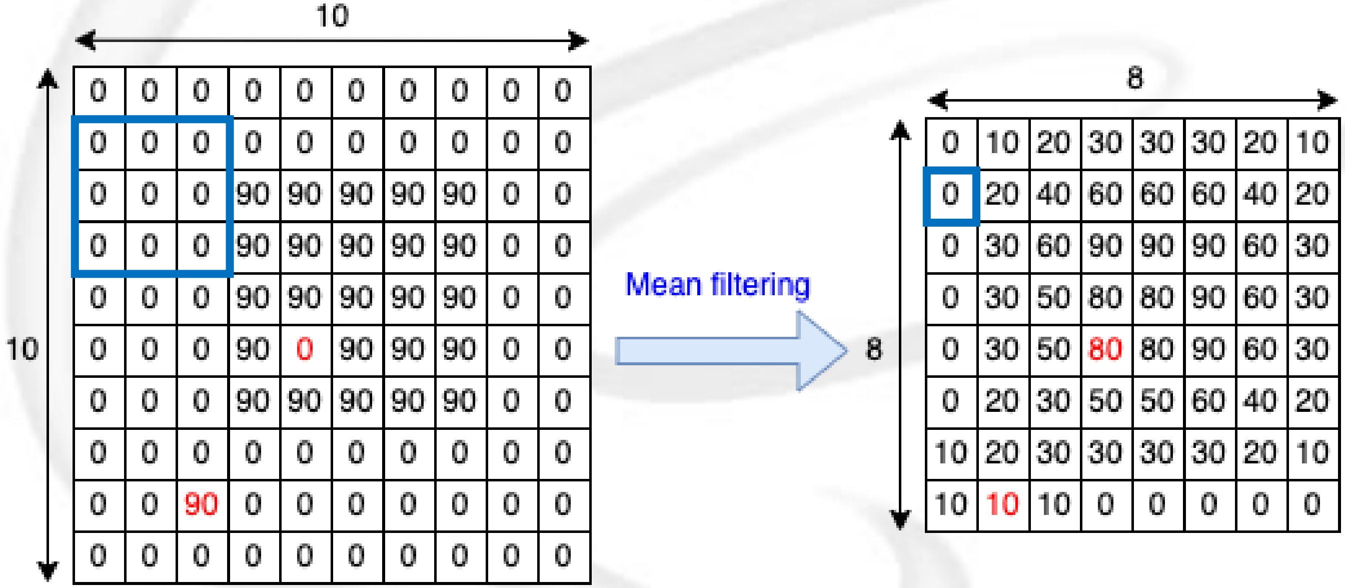
Introduce to image filter

□ Filtering to Remove Noise

- ➔ We can use spatial filters of different kinds to remove different kinds of noise

□ Mean filter

- ➔ Taking the average of pixel values within a fixed region(ex:3X3 window)
- ➔ This is implemented as the simple smoothing filter Blurs the image to remove noise

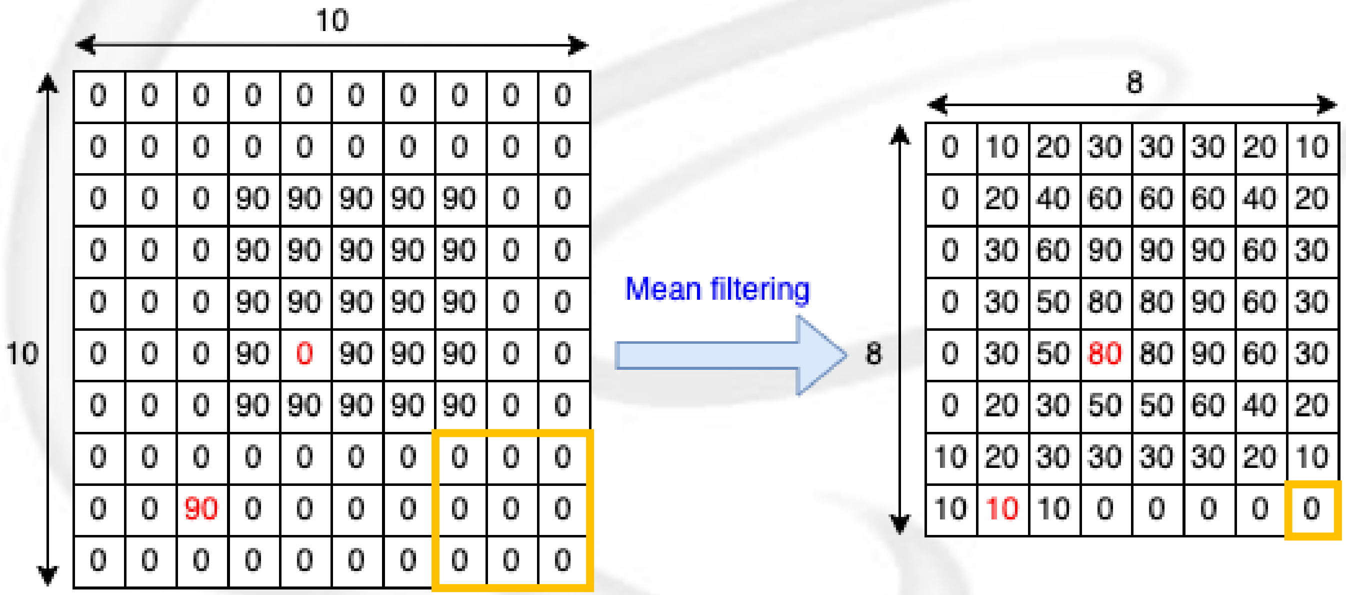


0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	10	20	30	30	30	20	10
0	20	40	60	60	60	40	20
0	30	60	90	90	90	60	30
0	30	50	80	80	90	60	30
0	30	50	80	80	90	60	30
0	20	30	50	50	60	40	20
10	20	30	30	30	30	20	10
10	10	10	0	0	0	0	0

Introduce to image filter

- Filtering to Remove Noise
  - ➔ We can use spatial filters of different kinds to remove different kinds of noise
- Mean filter
  - ➔ Taking the average of pixel values within a fixed region(ex:3X3 window)
  - ➔ This is implemented as the simple smoothing filter Blurs the image to remove noise



0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	10	20	30	30	30	20	10
0	20	40	60	60	60	40	20
0	30	60	90	90	90	60	30
0	30	50	80	80	90	60	30
0	30	50	80	80	90	60	30
0	20	30	50	50	60	40	20
10	20	30	30	30	30	20	10
10	10	10	0	0	0	0	0

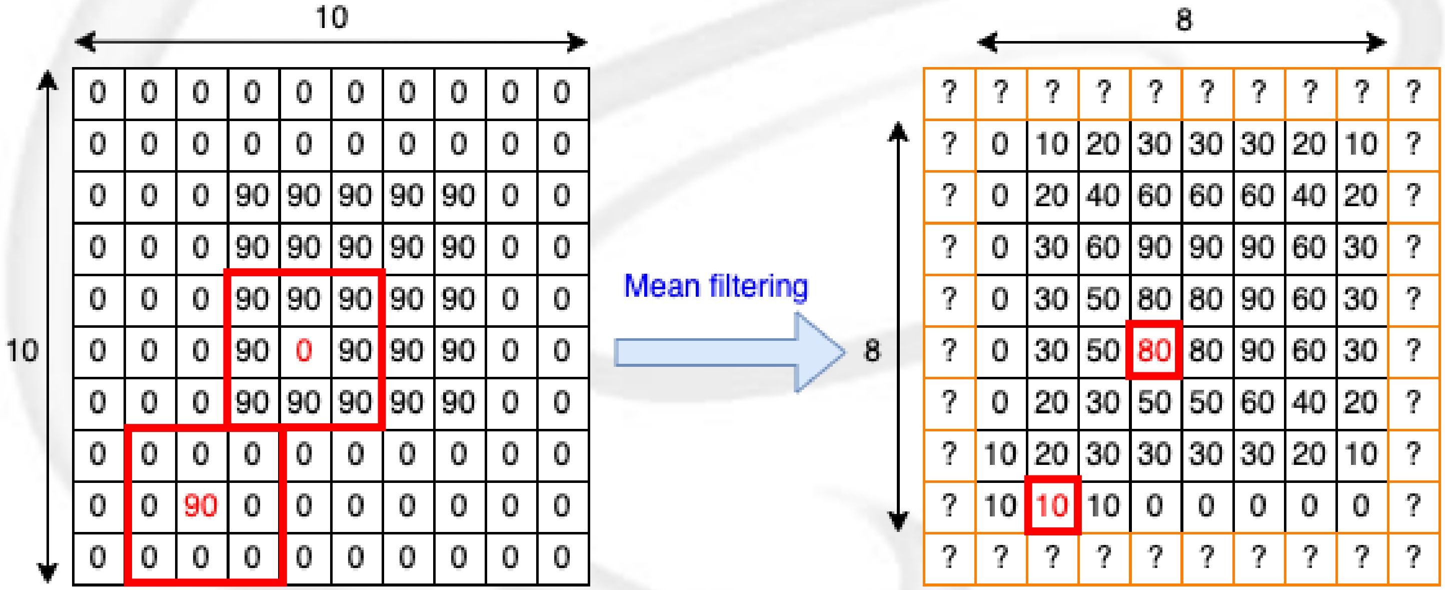
Introduce to image filter

□ Filtering to Remove Noise

- ➔ We can use spatial filters of different kinds to remove different kinds of noise

□ Mean filter

- ➔ Taking the average of pixel values within a fixed region(ex:3X3 window)
- ➔ This is implemented as the simple smoothing filter Blurs the image to remove noise



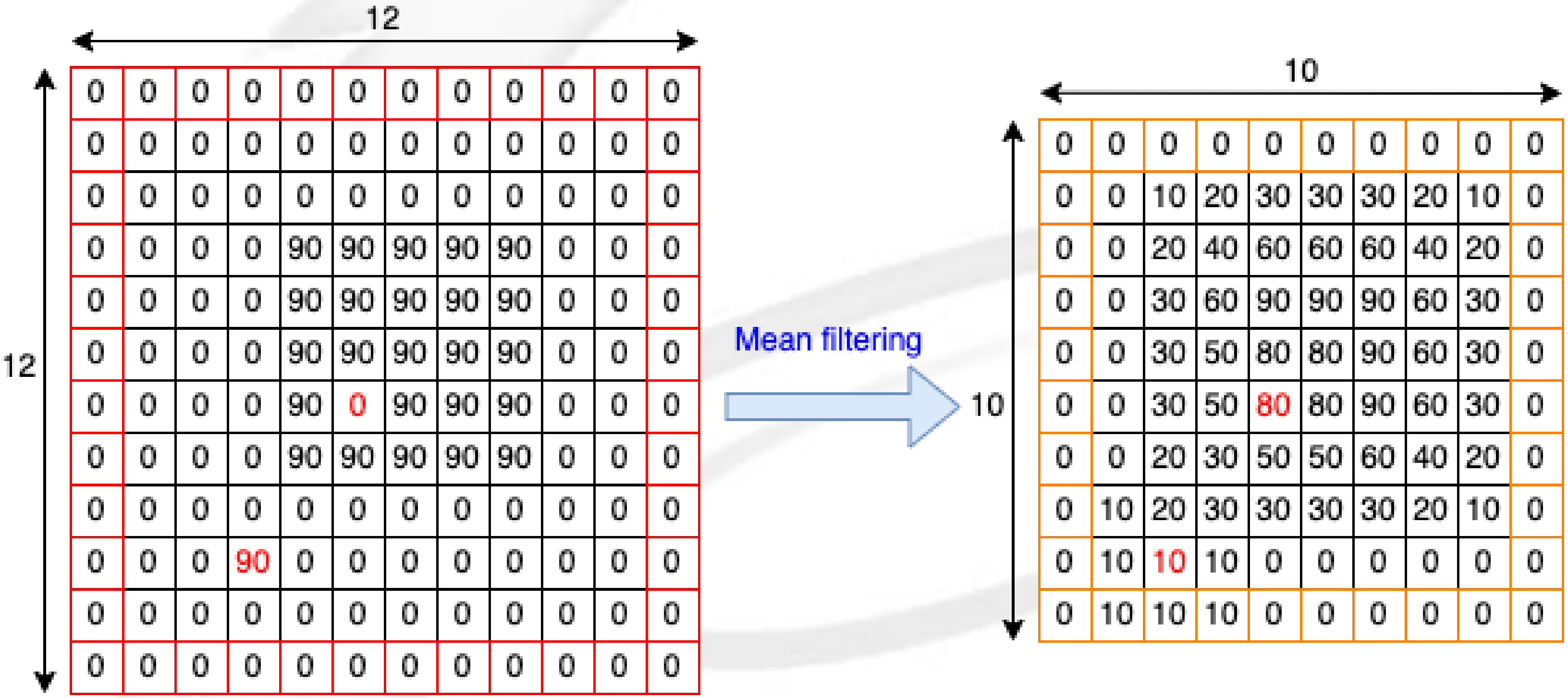
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

?	?	?	?	?	?	?	?	?	?
?	0	10	20	30	30	30	20	10	?
?	0	20	40	60	60	60	40	20	?
?	0	30	60	90	90	90	60	30	?
?	0	30	50	80	80	90	60	30	?
?	0	30	50	80	80	90	60	30	?
?	0	20	30	50	50	60	40	20	?
?	10	20	30	30	30	30	20	10	?
?	10	10	10	0	0	0	0	0	?
?	?	?	?	?	?	?	?	?	?

Boundary issue

Introduce to image filter

- How to solve the boundary issue?
  - ➔ Padding (Zero padding)



0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	90	0	90	90	90	0	0	0
0	0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

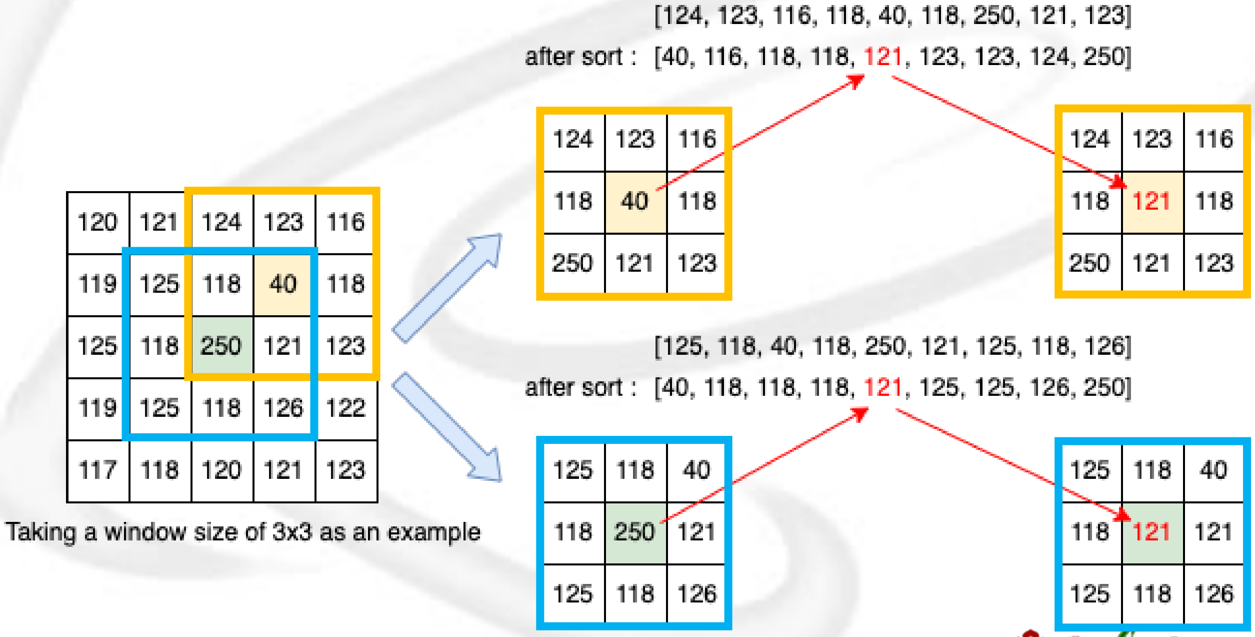
0	0	0	0	0	0	0	0	0	0
0	0	10	20	30	30	30	20	10	0
0	0	20	40	60	60	60	40	20	0
0	0	30	60	90	90	90	60	30	0
0	0	30	50	80	80	90	60	30	0
0	0	30	50	80	80	90	60	30	0
0	0	20	30	50	50	60	40	20	0
0	10	20	30	30	30	30	20	10	0
0	10	10	10	0	0	0	0	0	0
0	10	10	10	0	0	0	0	0	0

→ You need to do the zero padding in this Lab !

Introduce to image filter

□ Median filter

- ➔ Find the median of all pixel values within a fixed range and replace the original central pixel value with this median value.
- ➔ Excellent at noise removal, without the smoothing effects that can occur with other smoothing filters .
- ➔ Particularly good when salt and pepper noise is present.
- ➔ In Lab5, the window size is 3*3.



120	121	124	123	116
119	125	118	40	118
125	118	250	121	123
119	125	118	126	122
117	118	120	121	123

Taking a window size of 3x3 as an example

[124, 123, 116, 118, 40, 118, 250, 121, 123]

after sort : [40, 116, 118, 118, 121, 123, 123, 124, 250]

124	123	116
118	40	118
250	121	123

124	123	116
118	121	118
250	121	123

[125, 118, 40, 118, 250, 121, 125, 118, 126]

after sort : [40, 118, 118, 118, 121, 125, 125, 126, 250]

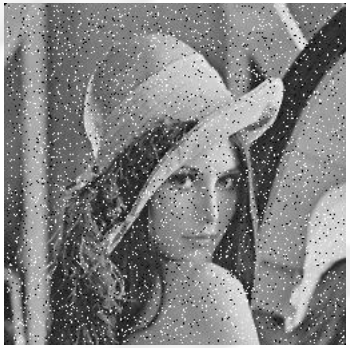
125	118	40
118	250	121
125	118	126

125	118	40
118	121	121
125	118	126

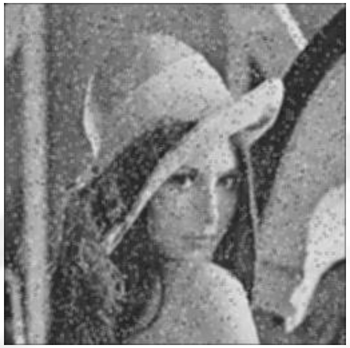
Introduce to image filter

- The effects of different spatial filters

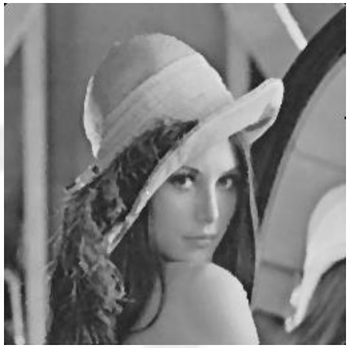
The original image



Add salt-and-pepper



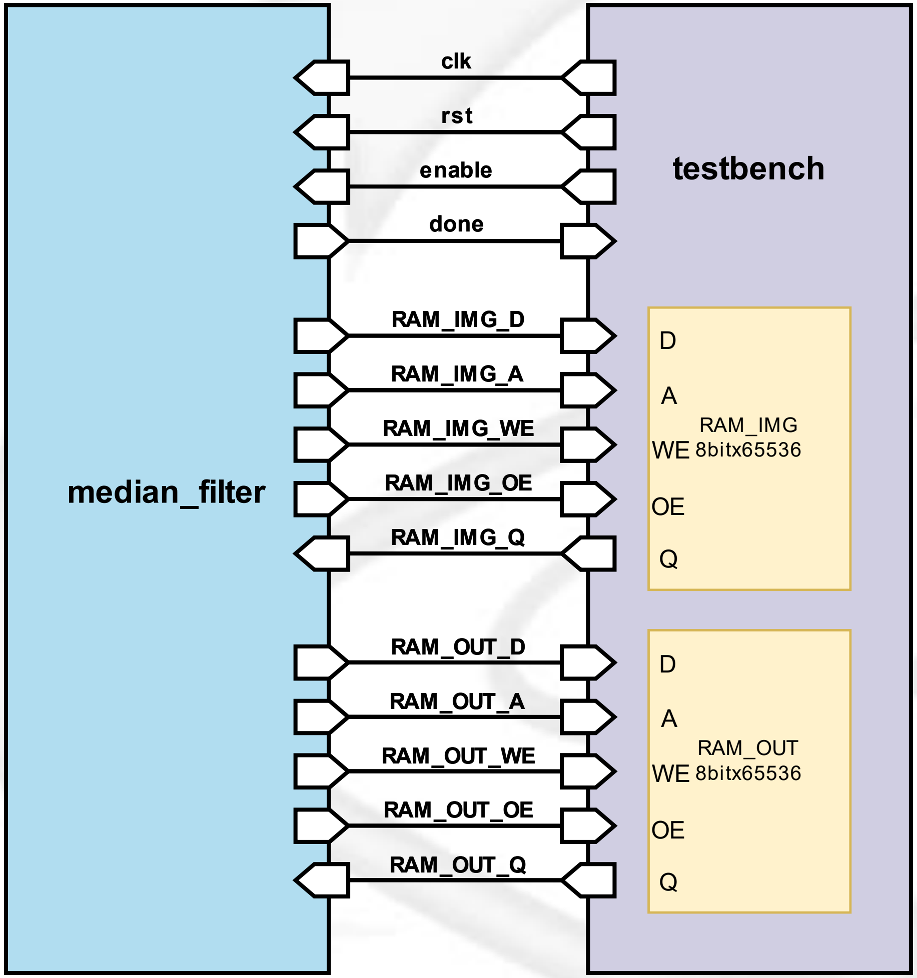
Mean filter(3 x 3)



Median filter(3 x 3)

Hardware description

□ Block diagram



256x256
input salt_and_pepper

256x256
median_filter output

Hardware description

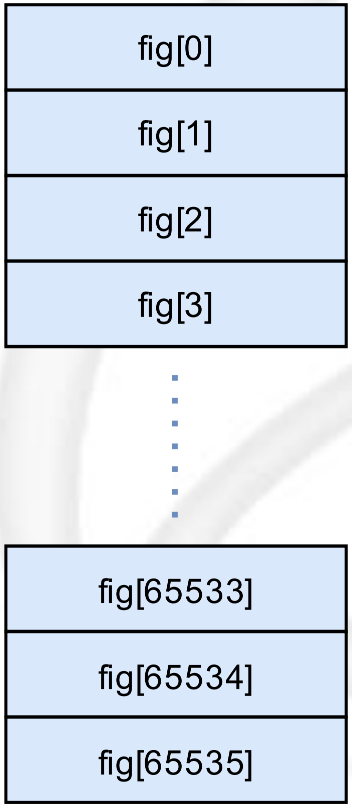
I/O Information

Signal	I/O	width	Desc.
clk	I	1	positive-edged triggered
rst	I	1	asynchronous positive-edged triggered
enable	I	1	enable signal to start processing
*_Q	I	8	8-bit data to be transmitted
*_OE	O	1	Active high read enable signal
*_WE	O	1	Active high write enable signal
*_A	O	16	Address
*_D	O	8	Data
done	O	1	Finish signal

Hardware description

Memory mapping(1/2)

- Content in RAM_IMG



fig[0]

fig[1]

fig[2]

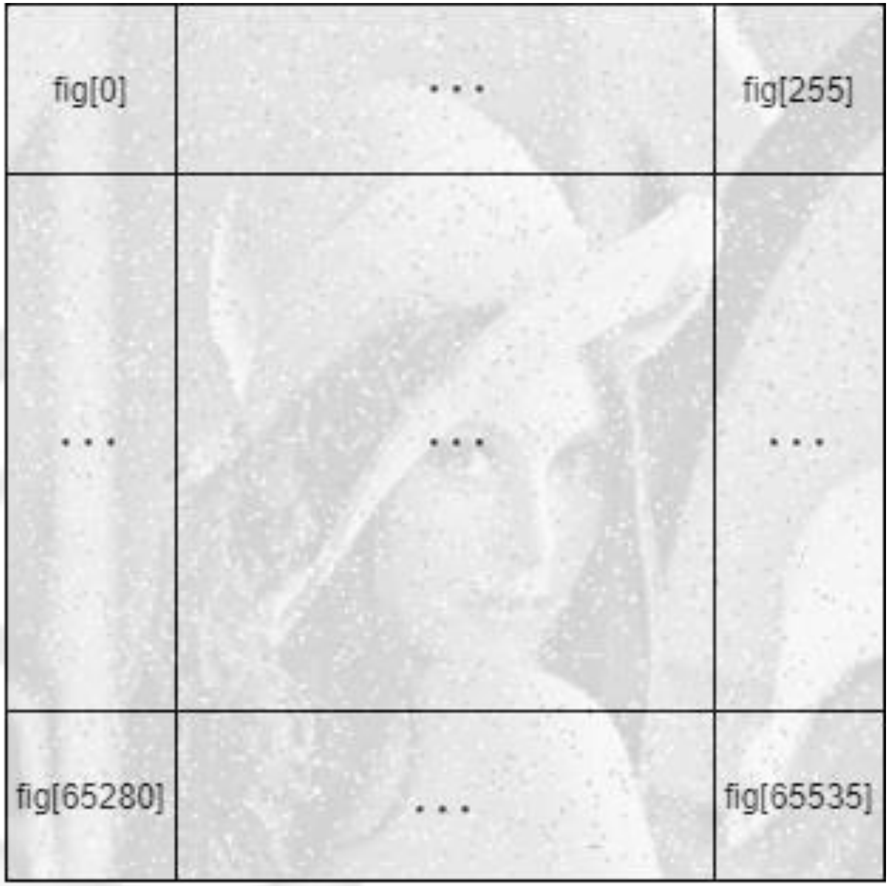
fig[3]

fig[65533]

fig[65534]

fig[65535]

- Storage format for input BMP

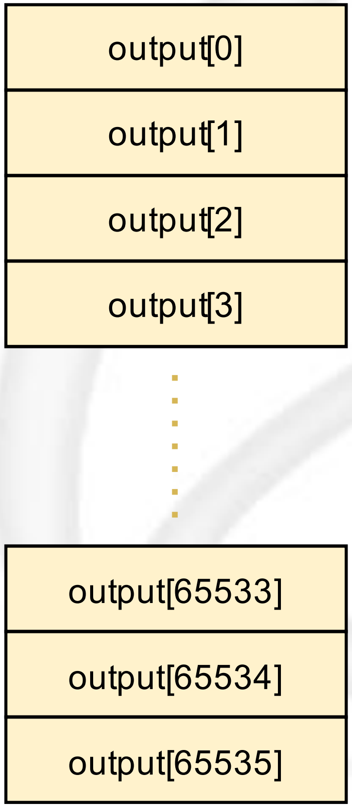


fig[0]	…	fig[255]
…	…	…
fig[65280]	…	fig[65535]

Hardware description

□ Memory mapping(2/2)

- ➢ Content in RAM_OUT



output[0]

output[1]

output[2]

output[3]

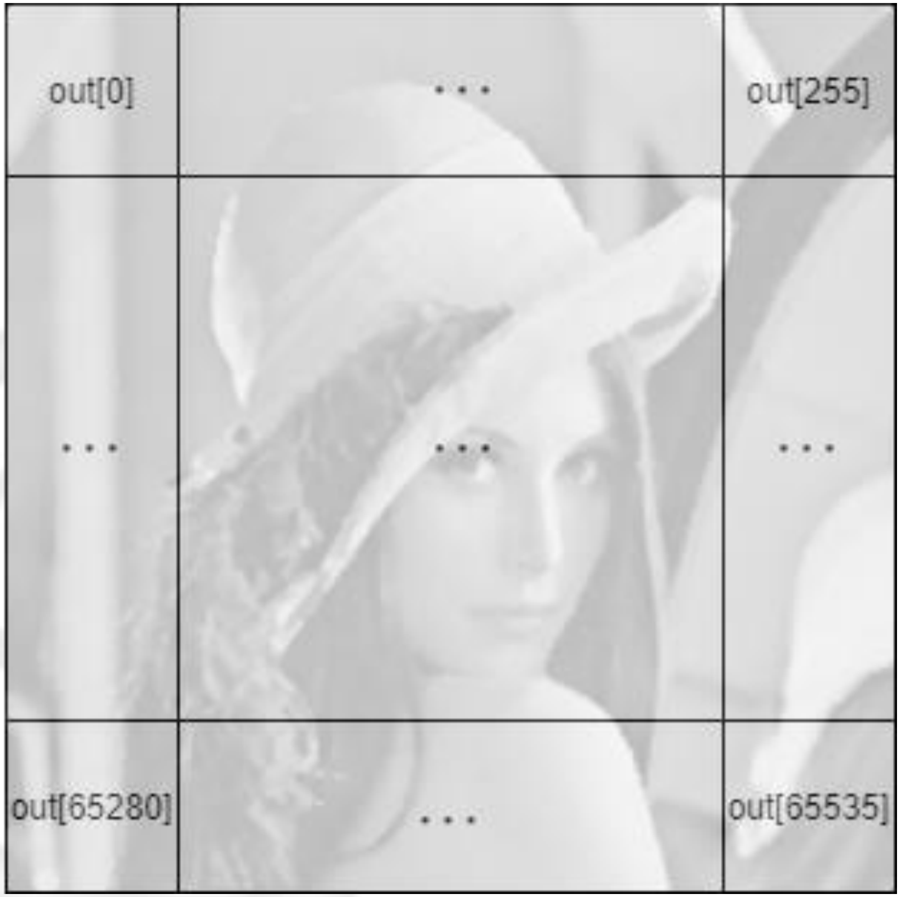
⋮

output[65533]

output[65534]

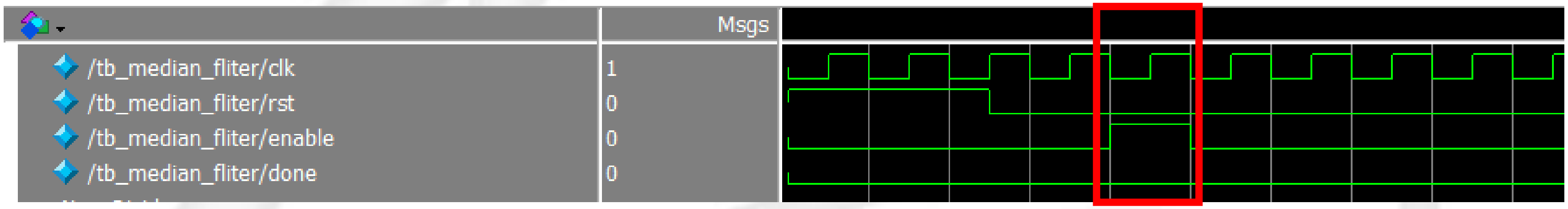
output[65535]

- ➢ Storage format for output BMP



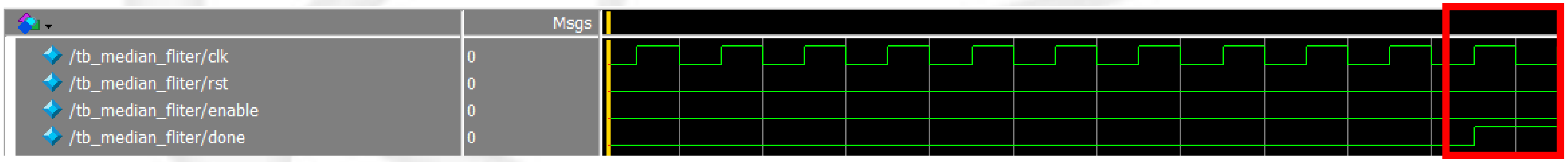
Hardware description

- enable & done signal
  - The enable signal will be pulled up to HIGH after reset, and will be maintained for only one cycle.



```
                           Msgs
/tb_median_fliter/clk      1
/tb_median_fliter/rst      0
/tb_median_fliter/enable   0
/tb_median_fliter/done     0
```

  - When the testbench receives the done signal, it will start comparing the values inside RAM_OUT.



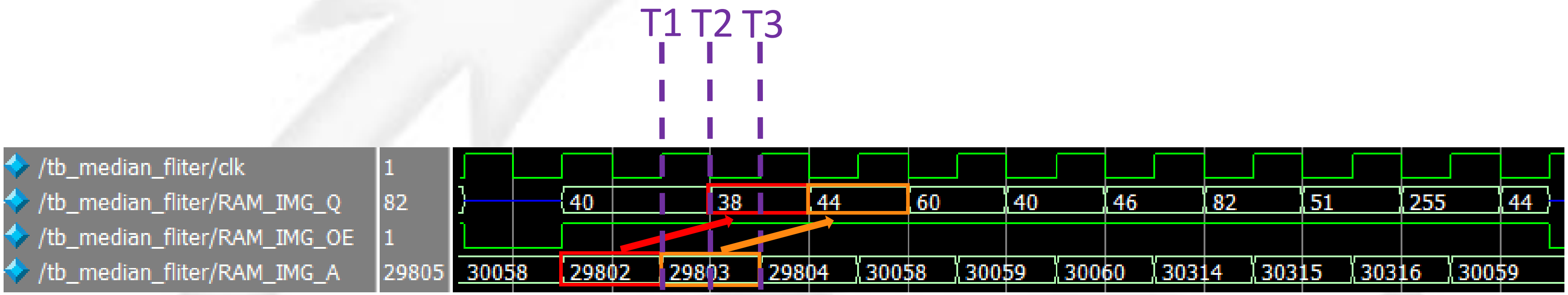
```
                           Msgs
/tb_median_fliter/clk      0
/tb_median_fliter/rst      0
/tb_median_fliter/enable   0
/tb_median_fliter/done     0
```

Flow in Lab5(1/2)

- Step1
  - Read each address and its surrounding eight pixels sequentially from RAM.
- Step2
  - Utilize the sort module to sort the nine pixels and find the median.
- Step3
  - After finding the median, write it into the memory RAM_OUT.
  - Repeat step1-step3 until the entire input image is processed.

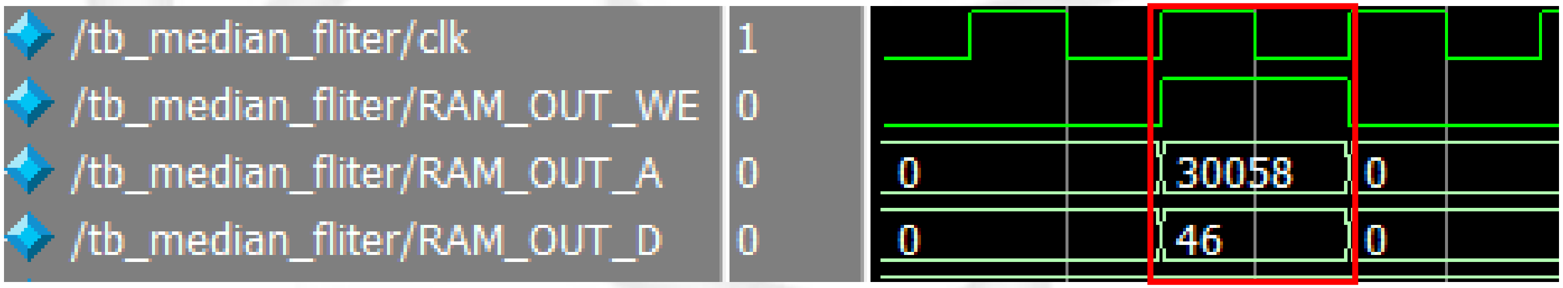
Flow in Lab5(2/2)

- The timing information for Read/Write SRAM
  - ➔ Read operation (delay one cycle)



```
/tb_median_fliter/clk         1
/tb_median_fliter/RAM_IMG_Q   82      40  38  44  60  40  46  82  51  255  44
/tb_median_fliter/RAM_IMG_OE  1
/tb_median_fliter/RAM_IMG_A   29805   30058  29802  29803  29804  30058  30059  30060  30314  30315  30316  30059
```

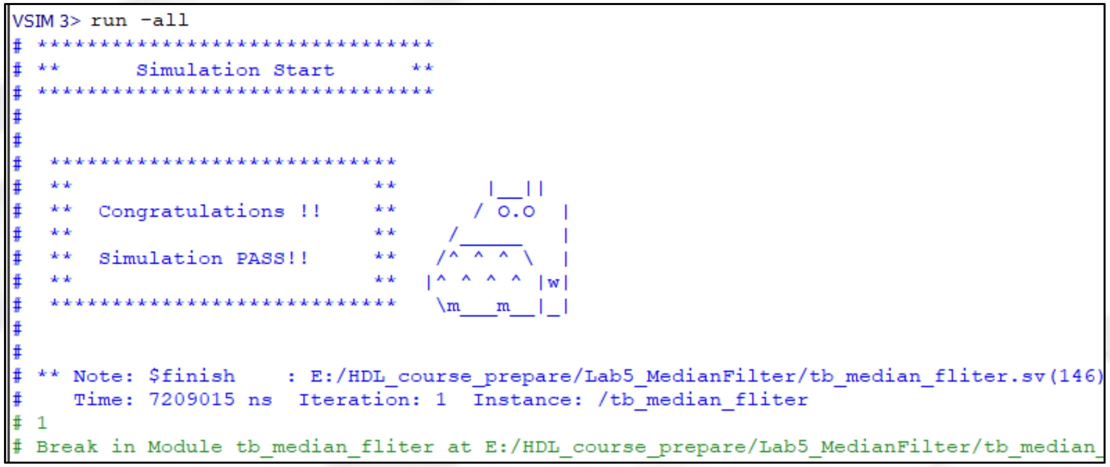
  - ✓ The memory will output values on the negative edge(T2), and you need to capture data on the positive edge(T3).
  - ➔ Write operation



```
/tb_median_fliter/clk          1
/tb_median_fliter/RAM_OUT_WE   0
/tb_median_fliter/RAM_OUT_A    0     0  30058  0
/tb_median_fliter/RAM_OUT_D    0     0  46     0
```

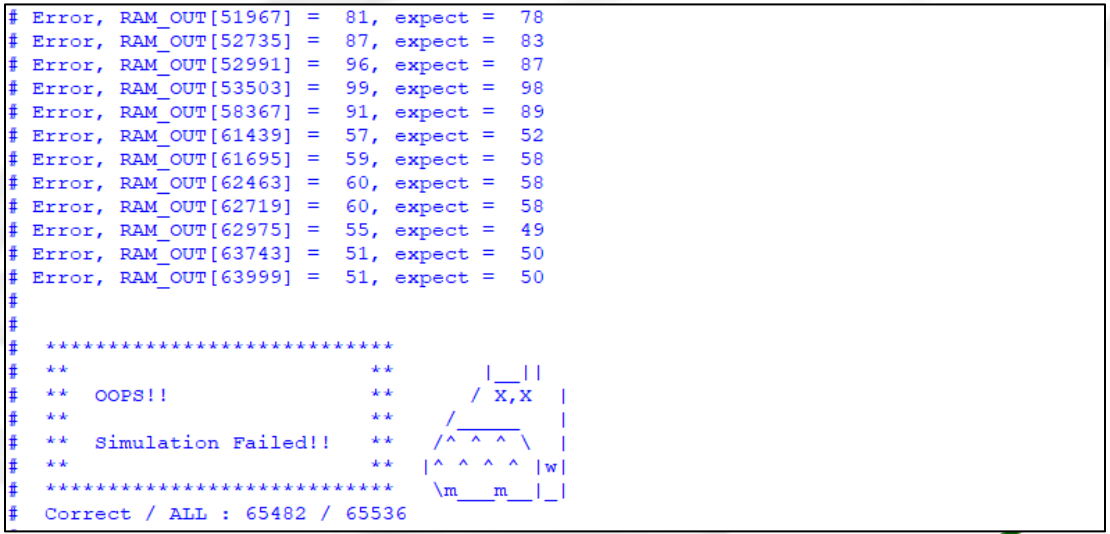
Criteria

- Simulation result
  - ➔ Pass



```
VSIM 3> run -all
# ********************************
# **      Simulation Start      **
# ********************************
#
#
#  ****************************
#  **                        **       |__||
#  **  Congratulations !!    **      / O.O  |
#  **                        **    /______  |
#  **  Simulation PASS!!     **   /^ ^ ^ \  |
#  **                        **  |^ ^ ^ ^ |w|
#  ****************************   \m___m__|_|
#
#
# ** Note: $finish    : E:/HDL_course_prepare/Lab5_MedianFilter/tb_median_fliter.sv(146)
#    Time: 7209015 ns  Iteration: 1  Instance: /tb_median_fliter
# 1
# Break in Module tb_median_fliter at E:/HDL_course_prepare/Lab5_MedianFilter/tb_median_
```

  - ➔ Failed

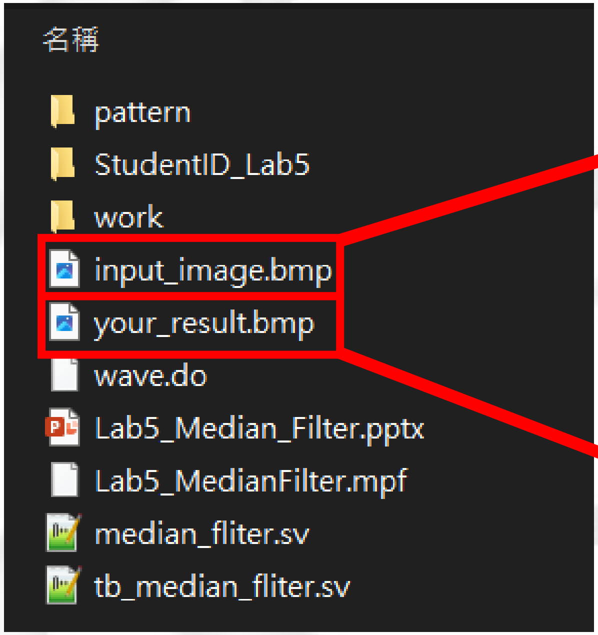


```
# Error, RAM_OUT[51967] =  81, expect =  78
# Error, RAM_OUT[52735] =  87, expect =  83
# Error, RAM_OUT[52991] =  96, expect =  87
# Error, RAM_OUT[53503] =  99, expect =  98
# Error, RAM_OUT[58367] =  91, expect =  89
# Error, RAM_OUT[61439] =  57, expect =  52
# Error, RAM_OUT[61695] =  59, expect =  58
# Error, RAM_OUT[62463] =  60, expect =  58
# Error, RAM_OUT[62719] =  60, expect =  58
# Error, RAM_OUT[62975] =  55, expect =  49
# Error, RAM_OUT[63743] =  51, expect =  50
# Error, RAM_OUT[63999] =  51, expect =  50
#
#
#  ****************************
#  **                        **       |__||
#  **  OOPS!!                **      / X,X  |
#  **                        **    /______  |
#  **  Simulation Failed!!   **   /^ ^ ^ \  |
#  **                        **  |^ ^ ^ ^ |w|
#  ****************************   \m___m__|_|
#  Correct / ALL : 65482 / 65536
```

Criteria

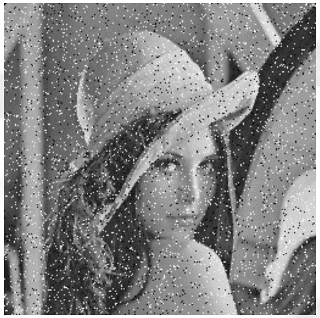
□ Simulation result – Visualization

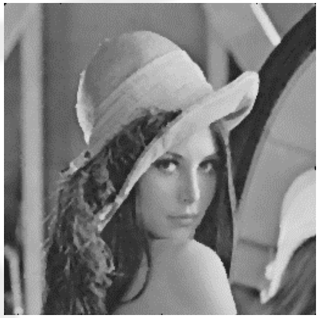
- ➔ It will generate the input picture and your output result in a BMP file when your simulation is finished.



名稱

- pattern
- StudentID_Lab5
- work
- input_image.bmp
- your_result.bmp
- wave.do
- Lab5_Median_Filter.pptx
- Lab5_MedianFilter.mpf
- median_fliter.sv
- tb_median_fliter.sv





Criteria

- Grading policy(100%)
  - Lab5
    - Simulation pass (90%)
    - Report (10%)

Lab5 Requirement & file format

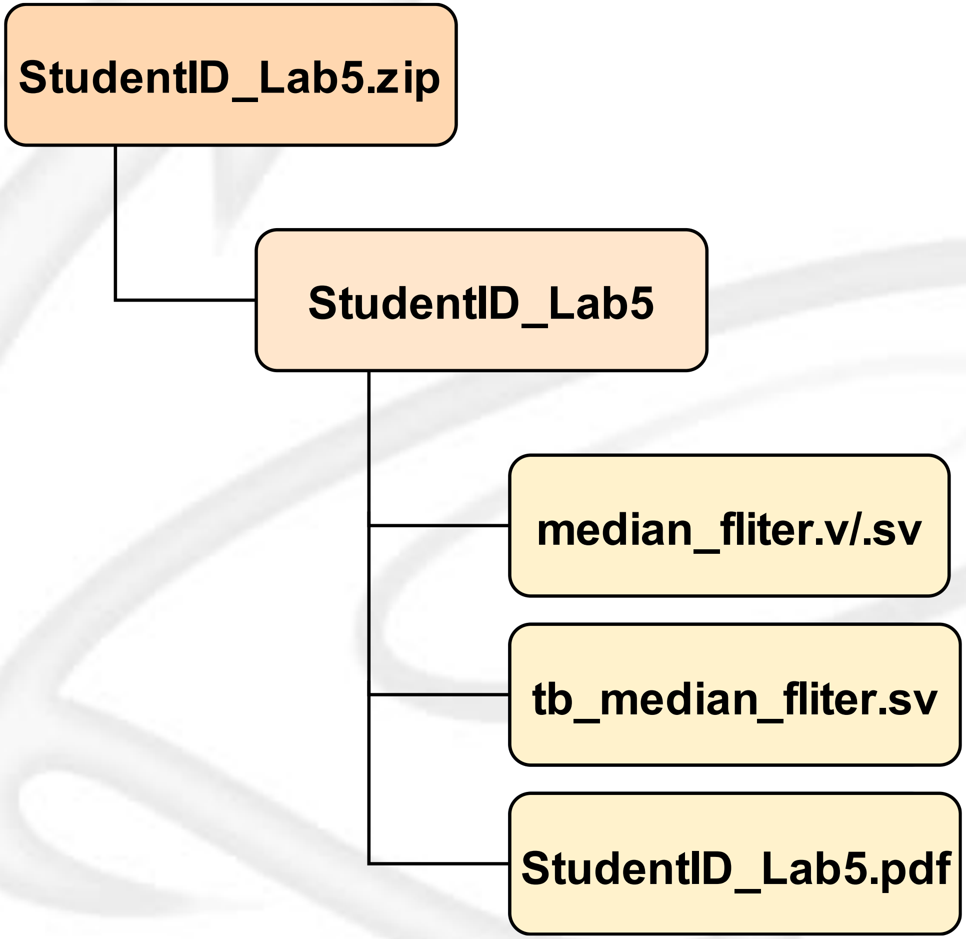
- You must finish median_filter.v/.sv and pass all patterns
- For Lab5, you need to submit
  - median_filter.v / median_filter.sv
  - tb_median_fliter.sv
  - StudentID_Lab5.pdf
- Deadline:2024/03/28 08:59 a.m. (No late submission)

Lab5 Requirement & file format

- Friendly reminder
  - Please complete the assignment by your own, discussion with peers is recommended, but do not cheat.
  - Warning! Any dishonesty found will result in zero grade.
  - Warning! Any late submission will also receive zero.
  - Warning! Please make sure that your code can be compiled in Modelsim, any dead body that we cannot compile will also receive zero.
  - Warning! Please submit your work according to the specified file format, making sure not to include any unnecessary files. Any unnecessary file found, will lead to 10% deduction from the overall score.

Lab5 Requirement & file format

- File format



StudentID_Lab5.zip

StudentID_Lab5

median_fliter.v/.sv

tb_median_fliter.sv

StudentID_Lab5.pdf

Thanks for listening