TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KỸ THUẬT MÁY TÍNH

MEDIAN FILTER ALGORITHM

FINAL PROJECT REPORT

Class: CE213.L21.MTCL
Lecturer: Lâm Đức Khải
Student - ID: **Nguyễn Thiên Đạt - 18520581**
Nguyễn Chí Dũng - 18520635
Trương Nguyễn Trung Anh - 18520575

Thành phố Hồ Chí Minh – Năm 2021

ACKNOWLEDGMENTS

I would like to thank my lecturer Lam Duc Khai who has provided knowledge about digital system design and HDL. Making us aware of the learning process and forcing me to experiment with various methods of conversing knowledge about design science.

All of these helped to complete the project Median Filter algorithm with Verilog.

CONTENT

I. INTRODUCTION

The median filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Such noises reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise (but see the discussion below), also having applications in signal processing.

The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighboring entries. The pattern of neighbors is called the "window", which slides, entry by entry, over the entire signal. For one-dimensional signals, the most obvious window is just the first few preceding and following entries, whereas for region.

II. MEDIAN FILTER ALGORITHM

Given the sequence x1, x2, …, xn monotonous with increasing (or decreasing) sequence. Then the median of the series of symbols med({xn}) is defined:

- if n odd: $Med = x\left[\frac{n}{2} + 1\right]$
- if n even: $Med = x\left[\frac{n}{2}\right]$ $or\ Med = x\left[\frac{n}{2} + 1\right]$

Usually filter with kernel 3x3, 5x5, 7x7 so n is odd number.

Median filter is one of the well-known order-statistic filters due to its good performance for some specific noise types such as "Gausian", "random", "salt and pepper" noises.

Figure 1. Example median filter

To demonstrate using a window 3x3 with one entry immediately preceding and following each entry, a median filter will be applied to the following simple-dimensional signal.

Example of 2D median filtering using a window 3x3. Keep contour values unchanged

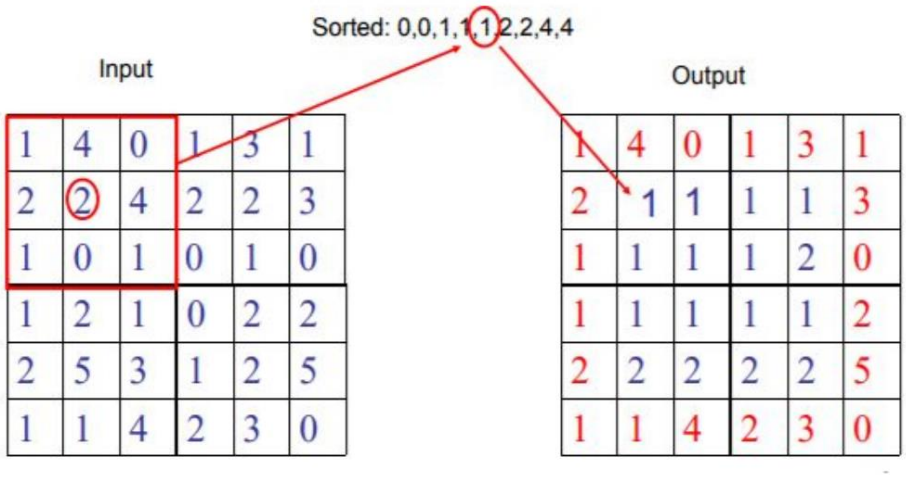

Figure 2.example median filter with matrix

The steps of the algorithm:

- Select a Sxy two-way window of size 3x3, with pixel processing P(x,y)
- Calculate the median Pmed of Sxy window
- The value of the processing pixel is replaced by Pmed
- Repeat the above step until the above steps are completed to get a complete image

Flowchart:

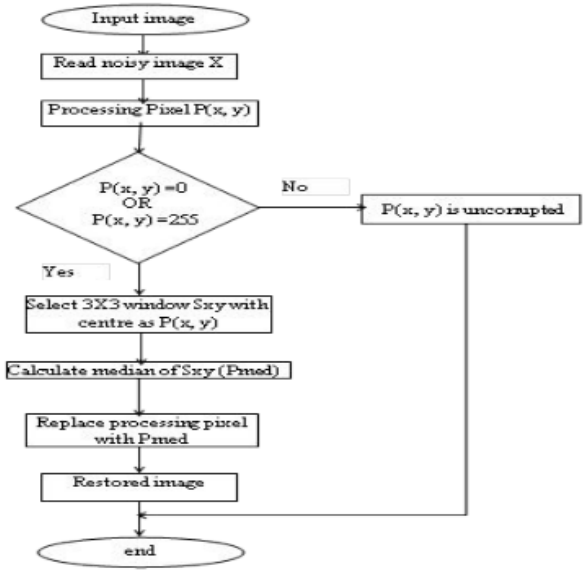

Figure 3. flowchart

III. SYSTEM

In this project, we will perform median filter by verilog code. This system gets a gray image with shape 256x256.

The steps of workflow:

- Implement Median by Python
- Convert image to binary.txt by Python
- Read binary.txt by RTL design and implement median filter by Verilog
- Compare the result of Python code and Verilog code

The RTL design consists of two block diagram are Controller and Datapath:

- Controller: Remote signal of datapath block
- Datapath: Implement median filter and write result

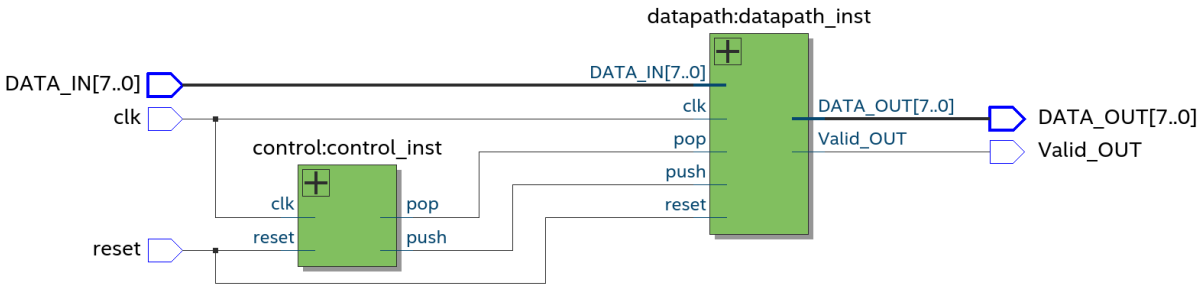

Figure 4. RTL Design

Signal	Function	Description
CLK	Input	Clock signal
Reset	Input	Reset signal
DATA_IN	Input	Pixel of 8 bit input image
DATA_OUT	Output	Pixel of 8 bit output image
Valid_OUT	Output	Valid output of data signal

1. DATAPATH

The datapath block consists of two block diagram are Median_filter and FIFO:

- Median_filer: Load image and calculate Median filter
- FIFO: Store and retrieval output

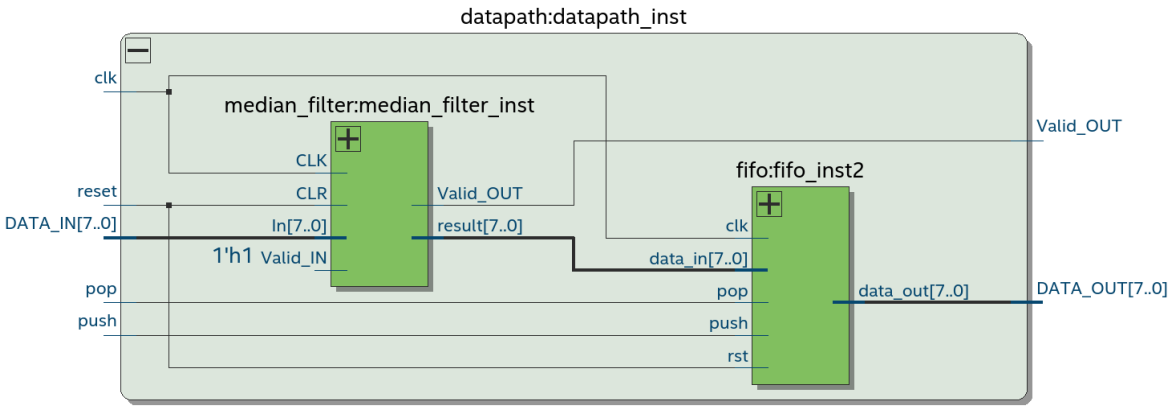

Figure 5.Datapath design

Signal	Function	Description
CLK	Input	Clock signal
Reset	Input	Reset signal
DATA_IN	Input	Pixel of 8 bit input image
Pop	Input	Pop signal of FIFO
Push	Input	Push signal of FIFO
VALID_OUT	Output	Valid output of data signal
DATA_OUT	Output	Pixel of 8 bit output image

1.1. Median_filter

The Median_filter block consists of two block diagram are linebuffer_with_kernel_3 and median:

- Linebuffer_with_kernel_3: Load and slide image with kernel 3x3
  Median: Calculate Median filter with kernel 3x3

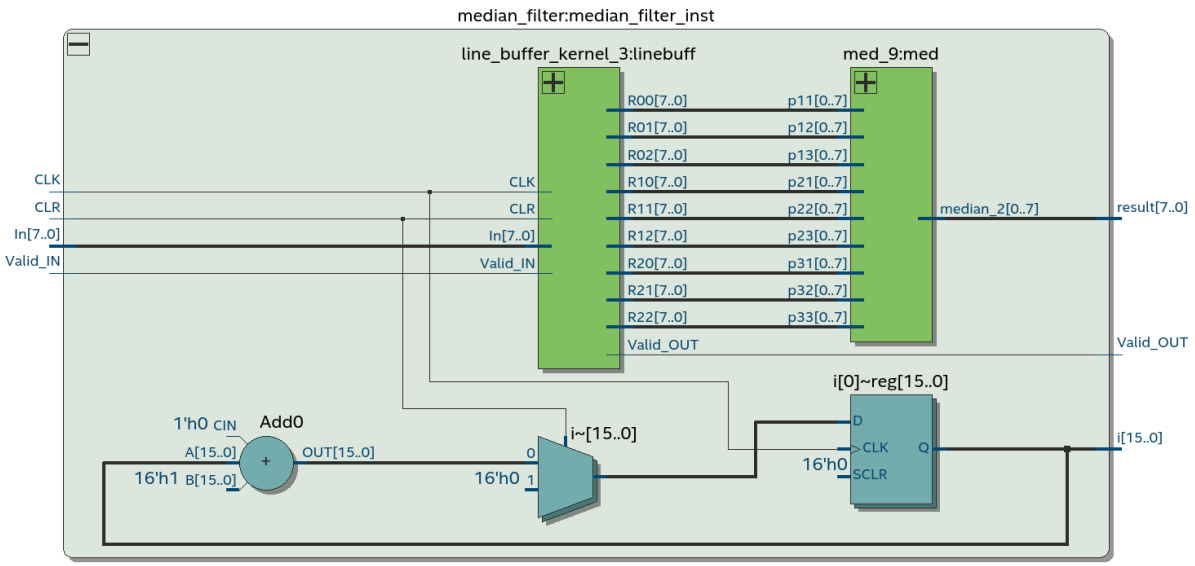

Figure 6. Median filter design

Signal	Function	Description
CLK	Input	Clock signal
CLR	Input	Reset signal
IN	Input	Pixel of 8 bit input image
Valid_IN	Input	Valid input of data signal
Result	Output	Pixel of 8 bit output image
Valid_OUT	Output	Valid output of data signal
i	Output	Count clk signal

a) Linebuffer_with_kernel_3

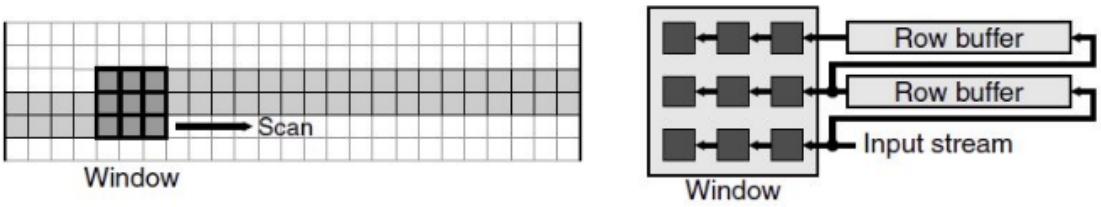

Figure 7. Line buffer with kernel 3x3

In digital image processing, line buffers are used very frequently. For example, when we need image matrix operations we need to buffer, such as image median filtering need line cache design

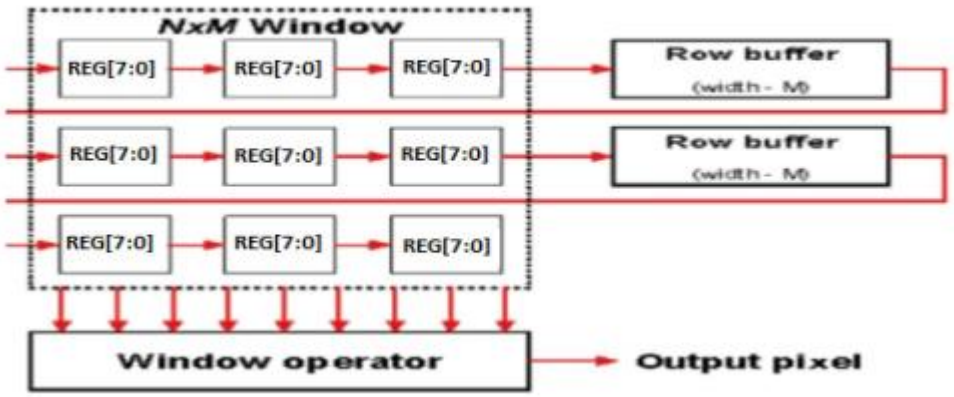

Figure 8.Kernel 3x3 design

The Linebuffer_with_kernel_3 consists of 6 block diagram are two line_buffer and three REG:

- Line_buffer: load 3 row image into REG and out row_0, row_1 of kernel 3
- REG: Out row_2 of kernel 3

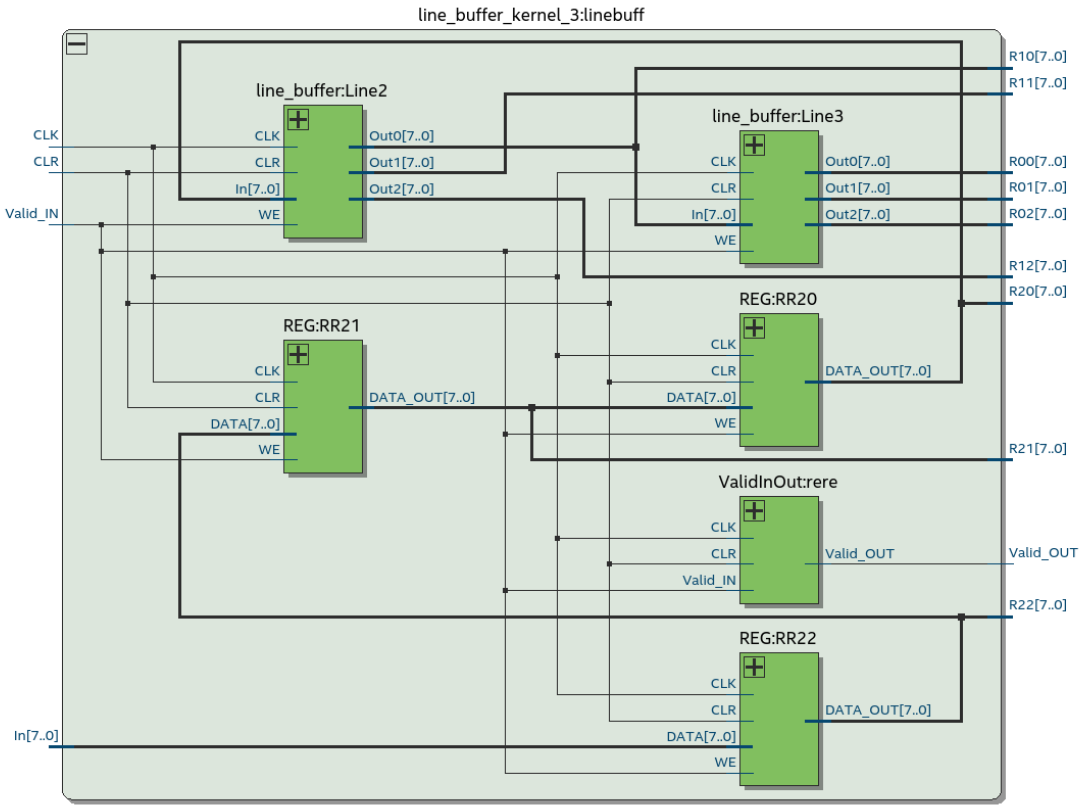

Figure 9. Line buffer with kernel 3 design

Signal	Function	Description
CLK	Input	Clock signal
CLR	Input	Reset signal
IN	Input	Pixel of 8 bit input image
Valid_IN	Input	Valid input of data signal
Valid_OUT	Output	Valid output of data signal
R00	Output	Pixel of 8 bit output image
R01	Output	Pixel of 8 bit output image
R02	Output	Pixel of 8 bit output image

R10	Output	Pixel of 8 bit output image
R11	Output	Pixel of 8 bit output image
R12	Output	Pixel of 8 bit output image
R20	Output	Pixel of 8 bit output image
R21	Output	Pixel of 8 bit output image
R22	Output	Pixel of 8 bit output image

b) Median

The Median block consists of four block diagram are med_3. The function of med_3 is calculate median of three number. the result of median of each row and calculate them is result of median of kernel 3x3.

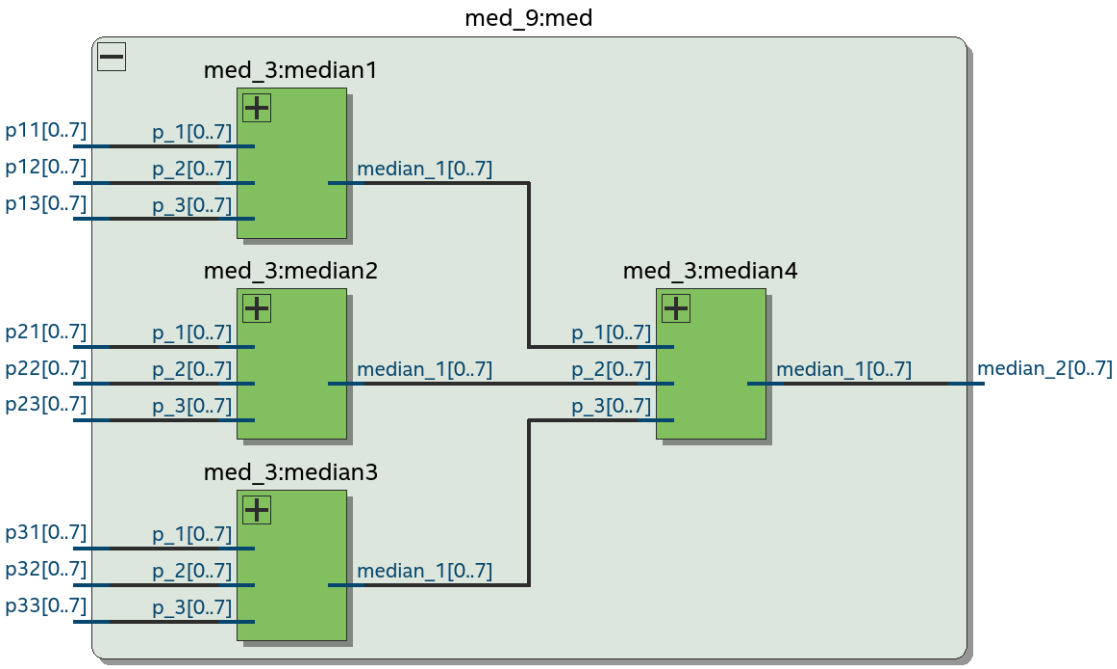

Figure 10. Median filer with kernel 3x3

1.2. FIFO

We use the FIFO have learned to store and retrieval output

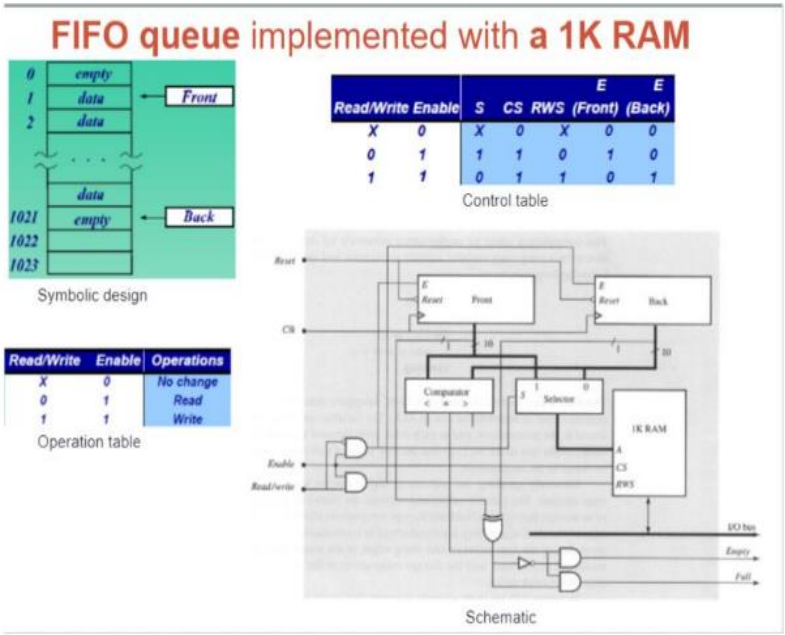

Figure 11. FIFO design

2. CONTROLLER

To remote the datapath clock, the controller block remote it.

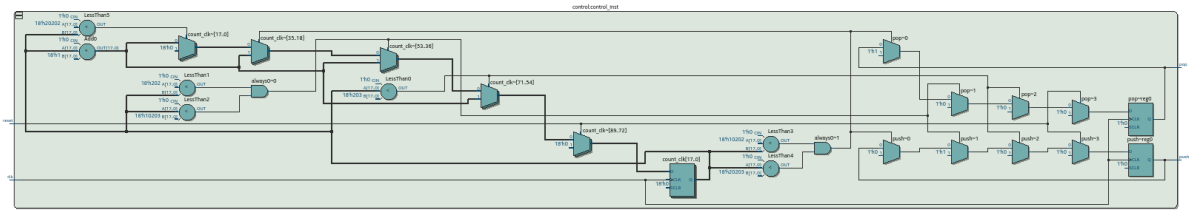

Figure 12. Controller design

Signal	Function	Description
CLK	Input	Clock signal
Reset	Input	Reset signal
Pop	Output	Pop signal of FIFO
Push	Output	Push signal of FIFO

IV. RESULT

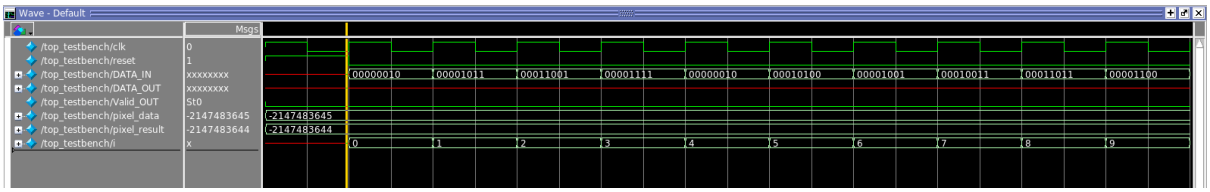

Figure 13. Load data

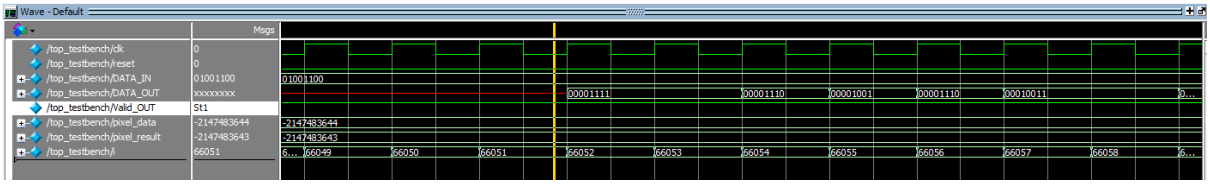

Figure 14. Store and Write Data

Figure 15. Verilog result

Figure 16..Original image

Figure 17. Python result

Analysis & Synthesis Resource Usage Summary

<<Filter>>

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	2330
2		
3	▼ Combinational ALUT usage for logic	300
1	-- 7 input functions	0
2	-- 6 input functions	83
3	-- 5 input functions	65
4	-- 4 input functions	69
5	-- <=3 input functions	83
4		
5	Dedicated logic registers	4499
6		
7	I/O pins	19
8	Total MLAB memory bits	0
9	Total block memory bits	524288
10		
11	Total DSP Blocks	0
12		
13	Maximum fan-out node	clk~input
14	Maximum fan-out	4563
15	Total fan-out	16757
16	Average fan-out	3.42

Figure 18. Resource sumary

REFERENCE

1. CE118 Digital Logic Design subject
2. CE213 DIGITAL SYSTEM DESIGN AND HDL subject
3. https://en.wikipedia.org/wiki/Median_filter