Advanced Engineering Forum *Online: 2012-09-26*
ISSN: 2234-991X, Vols. 6-7, pp 659-664
doi:10.4028/www.scientific.net/AEF.6-7.659

The Realization of Rapid Median Filter Algorithm on FPGA

Enshun Kang[a], Yuxi Zhao[b]

Information Science and Engineering, NEU, Shenyang, 110000, China

[a] kangenshun@mail.neu.edu.cn, [b]yuxihappy@163.com

Keywords: Median filter; FPGA; Image pre-processing ; Parallel processing

Abstract. Traditional median filter algorithm has the long processing time, which goes against the real-time image processing. According to its shortcomings, this paper puts forward the rapid median filter algorithm, and uses DE2 board of the company called Altera to do the realization on FPGA (CycloneII 2C35). The experimental results show that the image pre-processing system is able to complete a variety of high-level image algorithms in milliseconds, and FPGA's parallel processing capability and pipeline operations can dramatically improve the speed of image processing, so the FPGA-based image processing system has broad prospects for development.

Introduction

In 1971,the famous scholar put forward the non-linearity denoising algorithm which was called median filter. While denoising, it kept the edge not be faint. Median filter is effective to pulse noise called salt&pepper noise as well, because the noise is overlying the image by black and white point[1].

The realization of traditional median filter algorithm puts the pixels of window inside in order. The process of ordering is the process of comparing and exchanging the pixels. The comparing times are the important reason of impacting on taxis speed. For a window series of pixels on n, the first step is traversing the whole series and recording the maximum; The second step is traversing the whole series again in addition to the maximum in the previous step and recording the second maximum. It is ended until leaving the last value according to this method, and takes out the $(n+1)/2$ value, that is median output. In the whole comparing process, the comparing operation times are $n*(n-1)/2$, and the time complexity is $O(n^2)$[2,3]. This algorithm has the long processing time, which goes against the real-time image processing.

The presentation of rapid median filter algorithm

According to the shortcomings of the traditional median filter algorithm, this paper puts forward the rapid median filter algorithm, which makes full use of the rich FPGA hardware resources, has the advantage of parallel processing, owns smart design, and avoids a lot of comparing operations. It only needs to sort the pixels, and the time complexity is $O(n)$. The algorithm is described below:

We can suppose that the pixel data inside 2D median filter window are Pl, P2, P3, P4, P5, P6, P7, P8, P9. Window pixel arrangement is described as Table 1.

Table 1 Window pixel arrangement list

	The first column	The second column	The third column
The first row	P1	P2	P3
The second row	P4	P5	P6
The third row	P7	P8	P9

There are three steps to realize the algorithm. We can use *max* to take the maximum operation, *med* to take the median operation, and *min* to take the minimum operation.

The first step: Sort the pixel data in every row. We can get the maximum, median, minimum of every row apart. The taxis of three rows can be made parallel processing at the same time. If we mark the pixel maximum, median and minimum of the first row as Max_Row1 = max[P1, P2, P3],

Med_Row1 = med[P1, P2, P3] and Min_Row1 = min[P1, P2, P3], the second row and the three row follow the way as the first row. During three times compare of every row, we will get three data set under nine times compare, which are maximum, median and minimum set. It can be showed in the following:

The maximum set: Max=[Max_Row1,Max_Row2,Max_Row3];

The median set: Med=[Med_Row1,Med_Row2,Med_Row3];

The minimum set: Min=[Min_Row1,Min_Row2,Min_Row3];

The second step: Process the three sets data apart. Get the minimum *Max_min* from the maximum set, the median *Med_med* from the median set and the maximum *Min_max* from the minimum set. We need to compare seven times to gain the three value.

The three step: Find the median *Final_med* which is the finally result from the rest of the three value *Max_min*, *Med_med*, *Min_max*. The compare process above-mentioned is described as Fig.1.

The realization of rapid median filter algorithm on FPGA

Using FPGA to realize, in order to satisfy the real time requirement, it can use pipeline technology and parallel processing mode[4]. After one comparison operation of each data order, the data will be transmitted to the next level register, and further processed in the next clock circle. Then the level register will receive the data from the higher level register and do the corresponding processing. This design of eight levels can process the image data of three rows order at the same time. After gaining the first window data with eight clock circles later, then we will get median along with the template moving continuously[5].

If we gather the image of 352*288, which has 288 rows image data and 352 pixels in every row. After the signal are decoded, in order to ensure the real time of the system, it has three roads eight bits signals, which are Y, U, V. The design thoughts in this article adopt separate filter for every road signal, so three cache modules and three filter modules are needed[6]. For every road signal, the cache module consists of four Ram modules of 1024*8 bit. The specific constitute is showed by Fig.2.

Cache work process is that Ram1 receives the video data of the i+2 row at the same time when doing the i row data filter of Ram3. After the i row filter finished, each Ram content does serial storage downwards in turn, which is that Ram3 content be saved into Ram4, and Ram2 content be saved into Ram3, and Ram1 content be saved into Ram2. Do the upper processing again.

When using 3*3 window to filter, nine pixels data are needed from the three rows of video data in the cache. If the address of the filter window center pixel is P, other pixels address in the window are showed by Table 2.

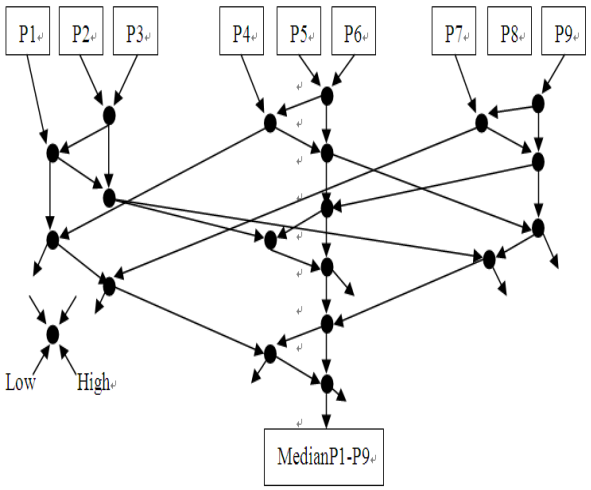

Fig.1 Figure of fast median filtering algorithm process

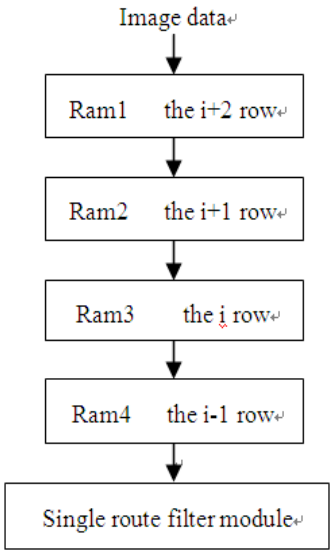

Fig.2 Filter modules schematic diagram of the buffer

Table 2 Address pixels within the window

P-289	P-288	P-287
P-1	P	P+l
P+289	P+288	P+287

After the window median filter finished, one pixel address glides backwards. The first row and the two hundred and eighty-eighth row don't participate in the median filter. Because of using Verilog Hardware Description Language to design rapid median filter algorithm, which we can see from the article above, more times of parallel processing need to use three data orders. Because of this, defining the task called Task sort3 does the order operation especially, then the result of order will be put out. When Task sort3 is transferred, the hardware can integrate several same parts, realize parallel processing, and improve process speed. The whole median filter flow uses a finite state machine to realize. The state transfer is showed by Fig.3.

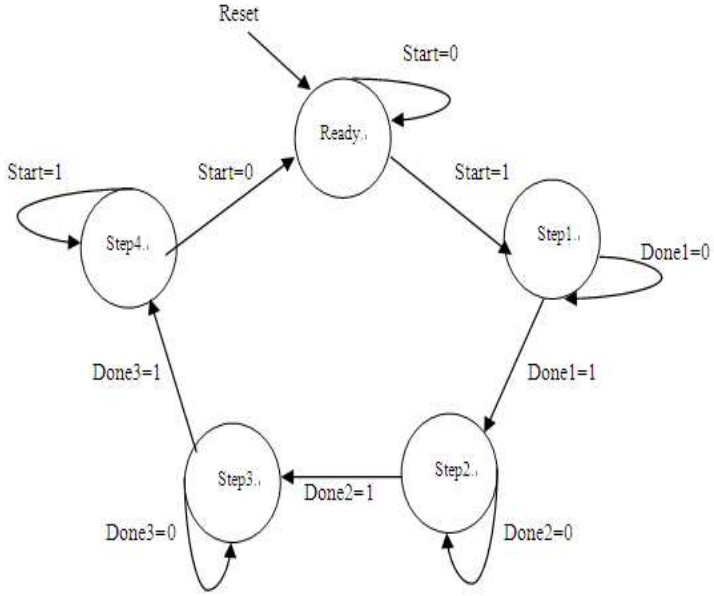

Fig.3 Filter module state transition figure

This state machine is the synchronous state machine of asynchronous reset, and the execution of the state transfer and the code in the state are controlled by the system clock, which is synchronous with the raise edge of the system clock[7]. In the state image, there are five correlative states, including Ready, Step1, Step2, Step3 and Wait. Ready is the original state, which is the state of the asynchronous reset. Ready state is that all the registers of storing the results clear, and sign bits are cleaned up in the state machine. Step1, Step2, and Step3 state correspond to the three steps in the rapid algorithm. Once detecting that Start signal is high level, it will enter into Step1 state after resetting with the raise edge of the next clock, or the primary state is always waiting. When entering into the Step1 state, it can transfer the sort3 task, and order the pixels of the three rows in the window. After the order is finished, the sign bit will set Done1. With the Step1 state, the raise edge of every clock can detect the sign bit. If being set shows that the first step order has finished. Similarly, call three times and one time order sort3 in Step2 and Step3 respectively, and at the same time detect the sign bit. When Step3 is completed, the median filter results will be sent to the designated registers, and it turns to Wait state, which is setting up for preventing a filter order from being executive by many times. When Start signal is a low level it turns into Ready prepartion state, waiting for the window to move down. Then filter once again, or stay in Wait state. Provide Verilog source code calling many times of the sorting purpose task in the following.

```
task sort3；  // This task completed the order of the three input number
input[7:0]a,b,c;
output[7:0]min,med,max;
```

```
reg[7:0]min_reg,med_reg,max_reg;// Define the register of the three storage result
begin
if(a<b)    begin
min_reg=a;
med_reg=b;
end
else        begin
min_reg=b；
med_reg=a；
end
if(c<min reg)    begin
max_reg=med_reg;
med_reg=min_reg;
min_reg=c;
end
else    if(c>med_reg)
maxe_reg=c;
else    begin
max_reg=med_reg;
med_reg=c;
end
min=min_reg;// The output of the task order result
max=max_reg;
med=med_reg;
end
endtask
```

Through the operation test, this method keeps the advantage of the traditional median filter method, remove the sudden noise point effectively, especially salt&pepper noise, but not affect the edge[8]. The image before filter is showed by Fig.4. The image after the rapid median filter algorithm is showed by Fig.5.

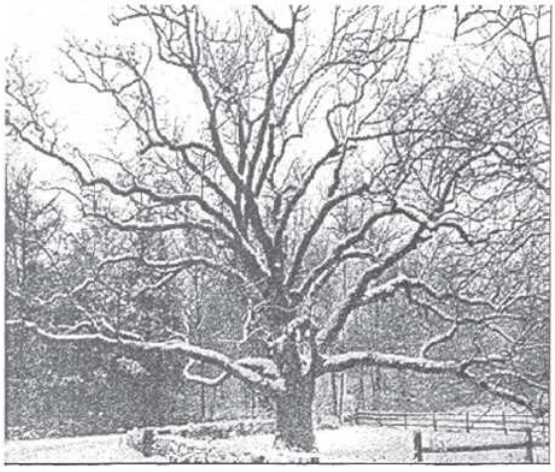

Fig.4 The image before filter

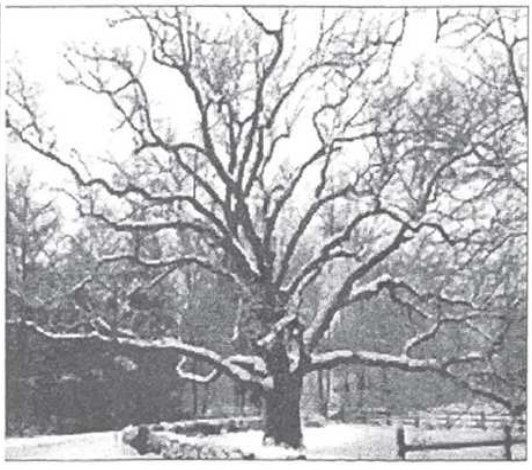

Fig.5 The image after rapid median filter algorithm

The analysis of rapid median filter result

This algorithm through 19 times comparison gets the median results of 3 * 3 filtering window. If there are the same numbers do not affect judgment, the algorithm through the logic inference discards the data which are not to meet the requirements. Take the minimum Max_rain from the maximum group. For the other two numbers in the group, they are at least more than 5 pixels in the window. Therefore it can not be as the middle value. Take the maximum Min_max from the minimum group. For the other two numbers in the group, they are at least less than 5 pixels in the

window. Therefore it can not be as the middle value. Take the median Mcd_mcd from the median group. For the maximum of the group, it is at least more than 5 pixels in the window. For the minimum of the group, it is at least less than 5 pixels in the window. So they are not as the middle value. That leaves three pixels as this, and sort them once again. What we can prove is that, and that no matter which row the maximum comes from, 5 pixels will be at least less than itself. So the final result is the median for the three pixels.

Traditional median filter algorithm is that nine data sortings in the window need at least thirty(8+ 7 + 6 + 5 + 4 = 30) compare times, and the time required is at least 30 clock cycles. Because of fast algorithm with parallel sort way, it makes full use of the FPGA hardware resources[9]. Compare times is reduced to 19, and compare time is greatly reduced to nine clock cycles. By testing, dealing with the gray image of a frame 352 * 288 pixels just needs about 1.87 ms or so. That takes up logical unit for 334, less than the FPGA (CycloneII 2C35[10]) logic resources of 1%. As shown in table 3.

Table 3 The occupancy resources situation of rapid median filter algorithm

Total Logic Elements	334/33216(1%)
Total Pins	55/672(8%)
Total Virtual Pins	0

If using the traditional median filter algorithm, through the simulation test, it needs about 5.17ms to deal with a frame 352 * 288 grayscale image. In addition if a picture of the same size will transform into data files, inputing to the samsung S3C2440 ARM chip of the main frequency for 405 MHZ for processing[11], through calculation, it needs about more than 150ms to finish.

Conclusions

（1）Traditional median filter algorithm is that nine data sortings in the window need at least thirty compare times, and the time required is at least 30 clock cycles. However, this article uses the fast median filter algorithm to reduce to 19 times, and to take up the time about one third of the traditional median filter algorithm. So the fast median filter algorithm is beneficial to real-time image processing.

（2）Dealing with the same picture, traditional median filter algorithm is needed about three times processing speed of the fast median filter. So the median filter algorithm used in this article can greatly enhance the system processing speed.

（3）Dealing with the same picture, the main frequency of ARM chips is eight times of FPGA chip, but processing speed is about 75 times as much as the FPGA chip. The results show that, FPGA compared with the ARM has the unique advantage in parallel processing aspect.

References

[1] Chong Zhang, Xiaolin Yu. FPGA application in image processing, Electronic quality，2004，(3)：13～19.

[2] Qin Hu, Cheng Peng, Xiaoning Sun. The optimization design of the FIR digital filters[J]. The space program measurement techniques，2006，26(6)：48-55.

[3] Chengqi Li. The design and realization of video collection system Based on FPGA technology [D]. Harbin university of technology, 2008，40-41.

[4] Deliang Liu, Chunlian Yao, Wui Li. FPGA logical design of more resolution image real-time data collection system[J]. Electronic technology application, 2003，(3)：69-72.

[5] Yitian Wang. FPGA image processing subsystem design of embedded video monitoring system[D]. 2009，48-56.

[6] Gonzalez. Digital image processing (the third edition)[M]. Beijing: Electronics Industry Publishing House. 2011，196-245，443-508.

[7] Li Peng, Jianye Qin, Yuzhuo Fu. The design and test of asynchronous FIFO[J]. Computer engineering and application. 2005，(03)：98—101．

[8] Richard Williams. Increase Image Processing System Performance with FPGAs. Xcell Journal,2004,16(2):10-11.

[9] Yehong Yin, Tao Wang, Ying Chen. The image preprocessing filter based on FPGA. Optical and photoelectric technology 2004，2(5)：61-63.

[10] DE2 Development and Education Board User Manual[EB/OL].Terasic Corporation,2006[2008-02-28].http://www.terasic.com.cn/downloads/cdrom/de2/.

[11] McErlean M. An FPGA Implementation of Hierarchical Motion Estimation for Embedded Object Tracking[J], Signal Processing and Information Technology, 2006 IEEE International Symposium on, 2006, 3(10):242-247.