Reconfigurable Hardware for Median Filtering for Image Processing Applications

Tripti Jain, Prashant Bansod, C. B. Singh Kushwah and Mayenk Mewara
Department of Electronics & Instrumentation Engineering
Shri G. S. Institute of Technology & Science
Indore, India
tjain25@gmail.com, ppbansod@iitb.ac.in, chandrabhansingh4@gmail.com, ermayank2004@yahoo.com

***Abstract*—Median filter is a non-linear filter used in image processing for impulse noise removal during morphological operations, image enhancement and other image processing operations. It finds its typical application in the situations where edges are to be preserved for higher level operations like segmentation, object recognition etc. Real-time applications, such as video and high speed acquisition cameras often require fast algorithms for processing. Reconfigurable hardware filters can be embedded with image acquisition system to achieve this goal. In this paper we propose a hardware implementation of a median filter with programmable window sizes ranging from 3x3 to 7x7. This median filter was designed, simulated and synthesized on the Xilinx family of FPGAs (XC3S500E of Spartan-3E). The performance of the same was evaluated for variety of images. For 3x3 window size, the maximum operating frequency achieved was approximately 89 MHz and for 5x5 window sizes, the maximum operating frequency achieved was approximately 89 MHz. The VHDL was used to design the above 2-D median filter using ISE (Xilinx) tool. The proposed hardware implementation took on an average 0.00246 m sec. For a frame of size 4x4 (for window size 3x3), 0.22472 m sec on for a frame of size 16x16 (for window size 5x5) and 0.468 m sec on a average for a frame of size 16x16 (for window size 7x7). The algorithm proposed for the median filter is based on sorting pixel samples and extracting their median values.**

Keywords-Median filter; Image processing; VHDL; FPGA

I. INTRODUCTION

Image processing is used in many fields like computer vision, remote sensing, medical imaging, robotics etc. In many of these applications the existence of impulsive noise in the acquired images is one of the most common problems [1]. This noise is generally removed from an image by using median filter as it preserves the edges during noise removal. Images can also corrupted by the shot noise, called salt-and-pepper noise. This noise is characterized by spots on the image and is usually associated with the acquired image due to errors in image sensors and data transmission [8]. 2-D median filter also finds application in the removal of salt and pepper noise. Median filter is non linear filter but is more robust than the traditional linear filter [1]. It is more effective method for noise reduction. It is used in image processing as well as speech processing. Software as well as hardware implementations for median filters in one or two dimensions are possible. Software implemented median filter as compared to hardware in general purpose processors for real time does not usually give good results since it is not fast enough. Therefore a software implementation with flexibility is required. Generally real time image processing system require a high speed and FPGA has fast executing speed, large memory and has capable of flexible logic control [3] that why FPGAs are often used as implementation platforms for real-time image processing application [2].

The proposed study was carried out to evaluate the performance of FPGA (XC3S500E of Spartan-3E) for image processing operations. We have implemented and tested the median filtering operations and compared the processing time with other offline methods. In section II related work is described briefly. Section III presents the proposed system architecture for implementing the median filter by means of FPGA. Section IV is implementation. In section V experimental results are analyzed and finally conclusion is presented in section VI.

II. RELATED WORK

Wnuk has purposed different architectures for implementation of image processing algorithm in hardware [5]. It have been shown that such an implementation is cost effective and offers a low power implementation with FPGA chip the design is also supported VHDL compilers, libraries, etc. Rodrigue et al. has purposed a new PCI based system with reconfigurable hardware for real time image processing [6]. In this system they have used the HOT2-XL PCI board. After the comparison of this hardware implementation with software they have observed that time require for FPGA based reconfigurable system is 85 times smaller than software. Maheshwari et al. has purposed parallel-serial input scheme and algorithm [7]. According to their algorithm first the number are sorted vertically then numbers are sorted horizontally and then in step 3 numbers are sorted cross diagonal elements and pick up the middle element this middle element is the median of the nine elements. They have ob-served that the hardware requirement of their design is very less.

III. SYSTEM ARCHITECTURE

The architecture of the proposed system is shown in Fig. 1. The system is composed of PC, Universal asynchronous receiver and transmitter (UART), Random access memory (RAM) and median filter. PC interfacing is optional and carried only for the testing purpose. In actual

978-0-7695-4246-1/10 $26.00 © 2010 IEEE
DOI 10.1109/ICETET.2010.172

implementation, image from the acquisition system will be fed for processing. Image data is taken from the PC and processed image is again sent back to PC. The three main components UART, RAM and the median filter are implemented in FPGA as a single logic unit as shown by dashed line in Fig. 1.

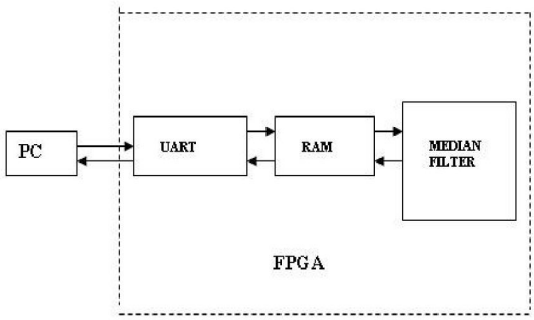

Figure 1. System architecture.

A. UART

UART includes a transmitter and a receiver. The transmitter is essentially a special shift register that loads data in parallel and then shifts it out bit by bit at a specific rate. The receiver, on the other hand, shifts in data bit by bit and then reassembles the data. No clock information is conveyed through the serial line. Therefore before the transmission begins, the transmitter and receiver must agree on a set of parameters in advance, which include the baud rate, number of data bits, stop bits, and, parity bit. UARTs are frequently used in conjunction with the EIA (Electronic Industries Alliance) RS-232 standard, which specifies the electrical, mechanical, functional, and procedural characteristics of two data communication equipments. Since the voltage level defined in RS-232 is different from that of FPGA I/O, a voltage converter chip is needed between a serial port and an FPGA's I/O pins. In our system UART is transmit the data from the PC to RAM and after processing the data is again send back to PC through UART. The design is customized for a UART with a 19,200 baud rate, 8 data bits, 1 stop bit, and no parity bit. We use an oversampling scheme to estimate the middle points of transmitted bits and then retrieve them at these points accordingly.

B. RAM

RAM is used as a temporary storage for the image data before and after processing through the median filter block. It accepts the image data from the UART and stores the image pixels in the column sorted fashion as required for the median filter algorithm. The size of the RAM depends on the maximum number of pixels in the input image. We have implemented the RAM to store an image of 128x128 pixels. RAM implementation in FPGA device puts a number of constraints on performance and speed [9]. It is implemented in FPGA by using smaller memories connected via multiplexors and decoders. The high-level entity consists of one decoder and a set of basic RAM components and multiplexors. The multiplexor and the decoder description use a generic VHDL behavior description. The basic RAM component depends on the specific device. In our implementation we have targeted Xilinx XC3S500E of Spartan-3E.

C. Median Filter

Median filter is nonlinear spatial filter. It is particularly effective in the presence of impulse noise also called salt and pepper noise [4]. Edges are preserved during the median filtering. Hence it is the primary choice in the situations where the image is to be further used for high level operations like segmentation, content based retrieval etc. Median filter is applied on a 2-D mask of size NxN, where N is an odd number. For implementing the median filter, the pixels in the chosen mask need to be sorted in the ascending order and then the center pixel of the mask is replaced by the median pixel value from the sorted list. Fig. 2 illustrates the procedure.

For the 3x3 mask with center pixel intensity of 17, the neighborhood pixels are sorted in the ascending order of their intensities. The center pixel intensity is replaced by the median value (5th) of the sorted list, which in this case is 17 as before. We have incorporated the comparator based approach. The comparator count and delay are considered as criteria for a given number of inputs.

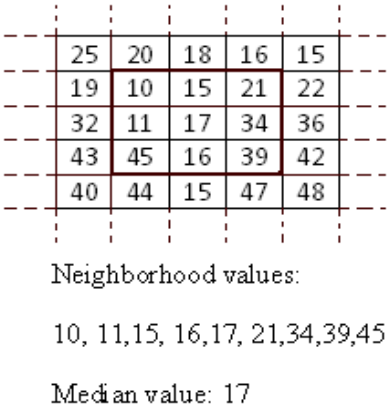

Figure 2. Application of the median filter

IV. IMPLEMENTATION

The median filter is implemented using three user selectable windows 3x3, 5x5 and 7x7. Bigger masks have a tendency to eliminate small edges [1]. The proposed architecture for median filter was tested on the images ranging from 4x4 up to 128x128 pixels. The images were transferred to the target FPGA Spartan-3E (XC3S500E) via UART for user selected median filter windows out of 3x3, 5x5 and 7x7. The median filtered images were transferred back to the PC for comparison purposes. The implementation

was carried out at the clock of 50 MHz. The processing time through the proposed architecture was recorded for each of the image. Also the percent utilization of the target device was estimated.

Table I shows the results for the above implementation for a median filter window size of 3x3. It is observed that as the size of image increases, the processing time increases along with the percentage utilization of the resources of the FPGA. This increase is linear with the size of the image after the image size exceeds 64x64 pixels. However the percentage increase in the CLBs utilized grows nonlinearly. Similarly table II and table III shows the results with 5x5 and 7x7 median filter respectively. Fig. 3, 4 and 5 give a plot between the image size and processing time. We have also compared our results with the processing time for the same operation on same images in MATLAB and C++. The off line processing was carried out on p-IV system with 1.99 GHz clock frequency with 1 GB RAM. These are shown in Table IV, V and VI respectively and plotted in the Fig. 3, 4 and 5.

TABLE I. SYNTHESIS RESULT FOR MEDIAN FILTER WITH WINDOW SIZE 3X3 USING XC3S500E OF SPARTAN-3E XILINX DEVICE

Image Size	Processing Time (m sec)	Utilization of Slices (in percentage)
4x4	0.00246	4 (200 out of 4656)
16x16	0.11766	4 (221 out of 4656)
32x32	0.54006	5 (237 out of 4656)
64x64	2.306	5 (250 out of 4656)
128x128	9.525	5 (259 out of 4656)

TABLE II. SYNTHESIS RESULT FOR MEDIAN FILTER WITH WINDOW SIZE 5X5 USING XC3S500E OF SPARTAN-3E XILINX DEVICE

Image Size	Processing Time (m sec)	Utilization of Slices (in percentage)
16x16	0.22472	5(251 out of 4656)
32x32	1.223	5(263 out of 4656)
64x64	5.616	5(278 out of 4656)
128x128	23.986	5(277 out of 4656)

TABLE III. SYNTHESIS RESULT FOR MEDIAN FILTER WITH WINDOW SIZE 7X7 USING XC3S500E OF SPARTAN-3E XILINX DEVICE

Image Size	Processing Time (m sec)	Utilization of Slices (in percentage)
16x16	0.468	7(340 out of 4656)
32x32	2.028	13(651 out of 4656)
64x64	10.092	34(1620 out of 4656)
128x128	44.652	more than 100 percent

TABLE IV. EXPERIMENTAL RESULTS OF FPGA, MATLAB AND C++ FOR THE MEDIAN FILTER (3X3 WINDOW SIZE)

Image Size	Processing Time (FPGA) (in m sec)	Processing Time (MATLAB) (in m sec)	Processing Time (C++) (in m sec)
32x32	0.54006	6.097	6.443
64x64	2.306	6.863	6.606
128x128	9.525	8.510	8.711

TABLE V. EXPERIMENTAL RESULTS OF FPGA, MATLAB AND C++ FOR THE MEDIAN FILTER (5X5 WINDOW SIZE)

Image Size	Processing Time (FPGA) (in m sec)	Processing Time (MATLAB) (in m sec)	Processing Time (C++) (in m sec)
32x32	1.223	14.143	14.055
64x64	5.616	15.385	15.029
128x128	23.986	19.315	20.782

TABLE VI. EXPERIMENTAL RESULTS OF FPGA, MATLAB AND C++ FOR THE MEDIAN FILTER (7X7 WINDOW SIZE)

Image Size	Processing Time (FPGA) (in m sec)	Processing Time (MATLAB) (in m sec)	Processing Time (C++) (in m sec)
32x32	2.028	12.138	11.851
64x64	10.092	14.994	13.674
128x128	44.632	21.804	21.412

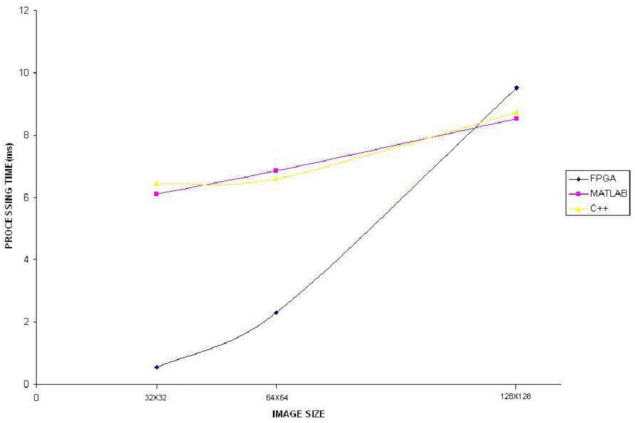

Figure 3. Comparison between FPGA, MAT-LAB and C++ for window size 3x3

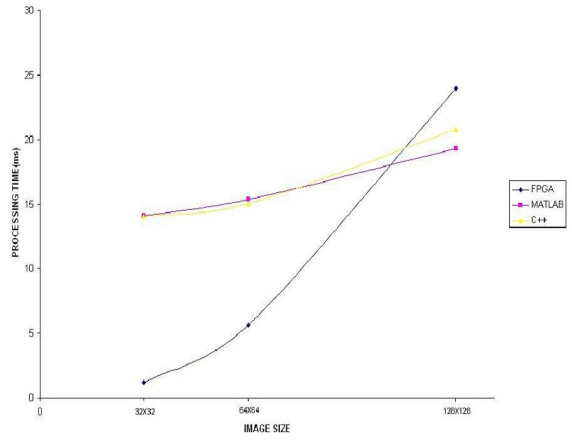

Figure 4. Comparison between the FPGA, MATLAB and C++ for window size 5x5

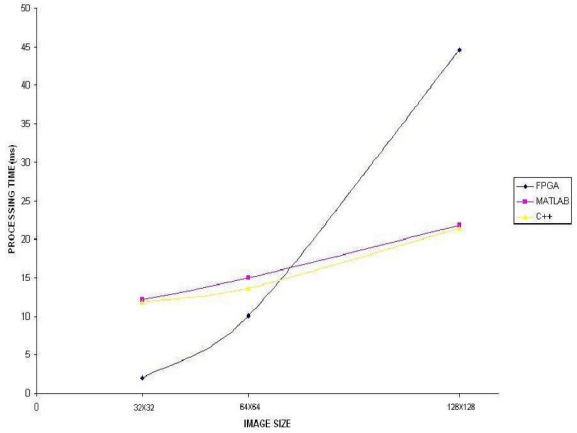

Figure 5. Comparison between FPGA, MAT-LAB and C++ for window size 7x7

V. RESULTS AND DISCUSSION

It is observed that the processing time for median filtering with MATLAB function and C++ routing are comparable for various image sizes and window sizes respectively. As the image size increases the processing time is increases as indicated in tables. FPGA implementation of the median filter offered much less time as compared to MATLAB and C++ for the image sizes 100x100 to 120x120 when the 3x3 and 5x5 median filter was implemented. however the with 7x7 median filter window the processing time was less for smaller image sizes typically 70x70. As the image size is increases the processing time in hardware implementation also went up and was of the same order that of MATLAB and C++ for image sizes greater than 120x120. Further the increase in processing time was much faster in FPGA implementation.

This is suggested that hardware implementation of above median filter is faster for the image size around 100 x100 or 120x120 for the targeted device (XC3S500E of Spartan-3). A graph indicates that hardware implementation performed poorer for image size above than 100x100. An improved performance for larger image size is possible by selecting a higher version of the FPGA or increasing the operating clock frequency.

VI. CONCLUSION

In this work we have presented an approach for reconfigurable hardware and implemented a median filter. We have targeted Xilinx FPGA XC3S500E for fitting UART, RAM and median filter. The implementation was tested with images ranging from 4x4 up to 128x128. The design offers implementation of median filter with various window sizes. The results indicate improvement and speed as compare to offline methods. Other image processing operations like edge filtering and segmentation can be integrated with in the same hardware. This will offer a real time image processor chip for medical as well as non medical applications.

REFERENCES

[1] Miguel A. Vega-Rodriguez, Juan M.Sanchez-Perez and Juan A. Gomez-Pulido, “An FPGA based implementation for median filter meeting the real-time requirements of automated visual inspection systems,” Proc. of the 10th Mediterranean Conference on Control and Automation -MED2002, Lisbon, Portugal, July 9-12, 2002, pp. 1-7, ISBN: 972-9027-03-X.

[2] C. T. Johnston, K. T. Gribbon and D. G. Bailey, “Implementing Image Processing Algorithms on FPGAs,” Proc. of the 11th Electronics New Zealand Conference (ENZCon '04), Palmerston North, New Zealand, November 2004, pp. 118-123.

[3] Duan Jinghong, Deng Yaling and Liang Kun, “Development of image processing system based on DSP and FPGA,” The Eigth International conference on Electronic Measurement and instruments, ICEMI’2007, Electronic Measurement and Instruments, 2007, pp. 2-791 - 2-794.

[4] Rafael C. Gonzalez and Richard E. Woods, “Digital Image Processing,” 2nd ed., Pearson education 2004.

[5] Marek Wnuk, “Remarks on hardware implementation of image processing algorithms,” Int. J. Appl. Math. Comput. Sci., vol. 18, March 2008, pp. 105-110, ISSN: 1641-876X.

[6] Miguel A. Vega-Rodriguez, Juan M. Sanchez-Pkrez and Juan A. Gbmez-Pulido, “Real Time Image Processing With Reconfigurable Hardware, 2001,” pp. 213-216.

[7] Rajul Maheshwari, S. S. S. P. Rao and P. G. Poonacha, “FPGA implementation of median filter,” tenth International Conference on VLSI Design, Jan 1997, pp. 523-524.

[8] Zdenek Vasicek and Lukas Sekanina, “An area-efficient alternative to adaptive median filtering in FPGAs,” FPL 2007 international conference, 27-29 Aug 2007, pp. 216-221.

[9] R. Senhadji-Navarro, I. Garca-Vargas, G. Jimnez-Moreno and A. Civit-Balcells, “FPGA-Based Implementation of RAM with Asymmetric Port Widths for Run-Time Reconfiguration,” Proc. IEEE International Conference on VLSI, 2007, pp. 178-181.