HW/SW Design-Based Implementation of Vector Median Rational Hybrid Filter

Anis Boudabous[1], Ahmed Ben Atitallah[1,3], Lazhar Khriji[2], Patrice Kadionik[3], and Nouri Masmoudi[1]
[1]Laboratory of Electronics and Information Technology, Tunisia
[2]Department of Electrical and Computer Engineering, Sultan Qaboos University, Oman
[3]IMS Laboratory, University Bordeaux I, France

Abstract: A *new code sign implementation of vector median rational hybrid filter based on efficient hardware/software implementation is introduced and applied to colour image filtering problems. This filter is used essentially to remove impulsive and Gaussian noise in colour images. In our design we start by implementing the software solution in system on programmable chip context using NIOS-II softcore processor and µClinux as operating system. We evaluate the execution time of the whole filtering process. Than we add a hardware accelerator part. This latter is implemented using fast parallel architecture. Compared to the software solution results, the use of the hardware accelerator improves clearly the filtering speed and maintains the good filtering quality as shown by simulations.*

Keywords: *Filtering, co-design, FPGA implementation, SoPC, NIOS-II processor.*

Received January 1, 2008; accepted September 11, 2008

1. Introduction

Filtering is one of the most important elements of color image processing system. Its most important applications are noise removal, image enhancement, and image restoration. Many classes of vector nonlinear filtering techniques have appeared in the literature such as order statistics filters, polynomial filters, morphological filters [1, 2, 5, 9, 17]. The well known vector filters include the Vector Median Rational Hybrid Filter (VMRHF) [10]. This nonlinear filter performs well in suppressing additive noise. The main contribution of this paper is the implementation of the VMRHF using Hardware/Software techniques in order to accelerate the execution time of the software solution. Generally Hardware implementation is better than software implementation in processing speed and power consumption. Also, co-design gives good flexibility to implement complex systems. Recently, several VMRHF filter have been developed also as software based applications, [7, 10] or hardware based FPGA device [11, 16]. In this work, the Hardware part of VMRHF Filter is implemented using VHSIC Hardware Description language (VHDL) language and integrated with the NIOS-II softcore processor in a single stratix II EP2S60F672C3 Field Programmable Gate Array (FPGA) device. The Software part can be running on NIOS-II softcore processor embedded in FPGA. This partitioning has been chosen in order to achieve better timing results.

This paper is structured as follows. Section 2 presents an overview of VMRHF filter. Description of the HW/SW codesign Platform and VMRHFF FPGA architecture is the object of section 3. In the next section, we give a comparison between the software solution and HW/SW implementation of the VMRHF filter. Finally, conclusions are drawn in section 5.

2. Overview of VMRHF Filter

The VMRHF filter uses three Vector Median Filter (VMF) filters [2, 12] in its first stage to remove impulsive noise and the Vector Rational Function (VRF) in the second stage. In addition to its detail preserving capability, the VRF removes Gaussian noise and small magnitude impulsive noise [8, 10]. Figure 1 shows the VMRHF. $\underline{x}(n)$ and $\underline{y}(n)$ are the input (noisy pixel) and the output samples, respectively. The output vector of the VMRHF is given by:

$$\underline{y}(n) = \underline{\Phi}_2(n) + \frac{\sum_{i=1}^{3} \beta_i \underline{\Phi}_1(n)}{h + k.\left\|(\underline{\Phi}_1(n) - \underline{\Phi}_3(n)\right\|} \quad (1)$$

The vector of coefficients $\beta = [-1 \;\; 2 \;\; 1]^T$ satisfies the unbiased system condition, $\sum_{i=0}^{2} \beta_i = 0$. The filter parameters h and k are two empirical constants [7, 11]. To find $\underline{\Phi}_1(n)$, $\underline{\Phi}_2(n)$ and $\underline{\Phi}_3(n)$ we use the VMF filter. The median filter (gray scale image) is a nonlinear operator that arranges the pixels in a local window according to the size of their intensity values and replaces the value of the pixel in the result image by the middle value in this order.

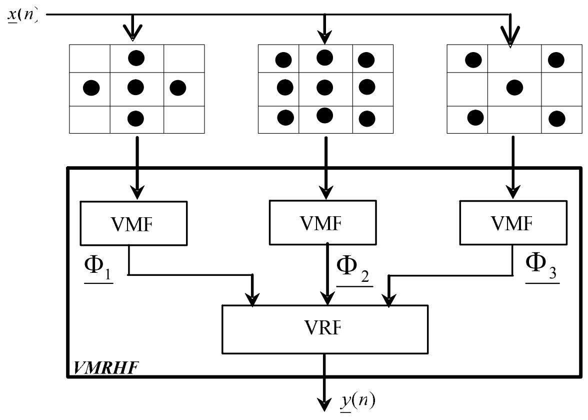

Figure 1. VMRHF block diagram.

This idea has been extended to vector-valued images [17, 18], based on the fact that the median is also the value that minimizes the L_2 norm between all the pixels in the window. We considered $\underline{x}_1, \underline{x}_2, \ldots \underline{x}_N$ input set determined by a filter window of size N. The VMF filter is based on euclidian distance calculation expressed as follow:

$$d_i = \sum_{j=1}^{N} \left\| \underline{x}_i - \underline{x}_j \right\| \tag{2}$$

We consider is the input vector of the VMF. The output vector of the VMF is given by:

$$\underline{y}(n) = \underset{\underline{x}(n) \in \underline{X}_i}{Argmin} \left(\sum_{j=1}^{N} \left\| \underline{x}_i - \underline{x}_j \right\| \right) \tag{3}$$

Argmin is the function given the value that minimizes the L_2 norm between all the pixels in the filter window.

3. NIOS-II Development Board

The two major FPGA manufacturers provide commercial softcore processors; Xilinx offers MicroBlaze processor [20], also, ALTERA offers NIOS and NIOS-II processors [14].

The ALTERA NIOS-II softcore processor (fast version) is a 32-bits scalar RISC with Harvard architecture, 6 stages in pipeline, 1-way direct-mapped 64KB data cache, 1-way direct-mapped 64KB instruction cache and can execute up to 150 MIPS [15]. The main interest of this softcore processor is its extensibility and adaptability. Indeed, users can incorporate custom logic directly into the NIOS-II Arithmetic Logic Unit (ALU) [16]. Furthermore, users can connect into the FPGA the on-chip processor and custom peripherals to a dedicated bus (avalon bus). Thus, users can define their instructions and processor peripherals to optimize the system for a specific application.

For SW implementation of image processing algorithms, the use of a microprocessor is required. The use of additional HW, for optimization, affects the overall performance of the algorithm [3]. For the highest degree of HW/SW integration, a softcore processor was used. System on Programmable Chip (SoPC) builder mega-wizard [19] is used to create NIOS embedded processor systems with 32-bits CPU core, built-in peripherals, on-and off-chip ROM and RAM support and bus support for external hardware modules.

4. HW/SW Design and Partitioning

A HW/SW co-design approach is then possible and a particular functionality can be developed in software for flexibility and upgrading completed with hardware IP blocks (intellectual property) for cost reduction and performances. The block diagram of the implemented filter is shown in Figure 2.

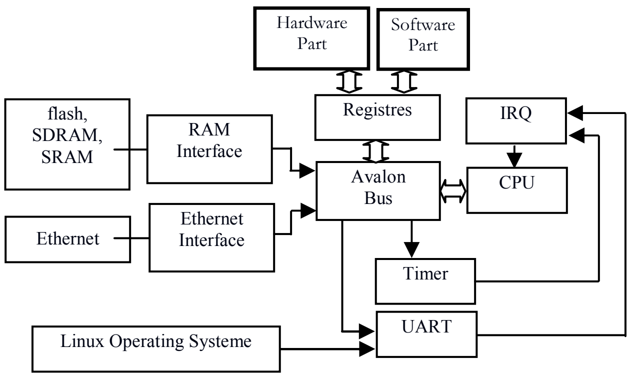

Figure 2. HW/SW partitioning of VMRHF filter.

The design is structured in three parts: NIOS-II CPU, VMRHF filter calculation and part three is avalon Bus and Peripherals. The main processing core of our system is the NIOS-II CPU which is connected to hardware peripherals via a custom ALTERA's Avalon bus. The bus is a configurable bus architecture that is auto generated to fit the interconnection needs of the designer peripherals. The avalon bus consists of the control, data and address signals and arbitration logic that are connected to the peripheral components.

In this work, we use the μClinux as an operating system to control the functionality of the design and extract the filtered image. Linux for embedded systems (or embedded Linux) gives us several benefits:

- It is ported to most of processors with or without Memory Management Unit (MMU).
- A Linux port is available for the NIOS-II softcore.

Most of classical peripherals are ported to Linux. A file system is available for data storage; a network connectivity based on Ethernet protocols is well suited for data recovering.

4.1. Software Part

The software part is used, basically, to optimize the reading of noisy image, the reconstruction of filtered image and the loading pixels to and from hardware core. The used idea of loading pixels is shown in Figure 3 and described by:

- The first filter window is formed by sending nine pixels to the hard core.
- The second filter window is obtained by exploiting the last six pixels from the previous window and sending only three new pixels to the hard core.
- The j^th filter window is obtained by exploiting the last six pixels from the (j-1)^th window and sending only three new pixels to the hard core.

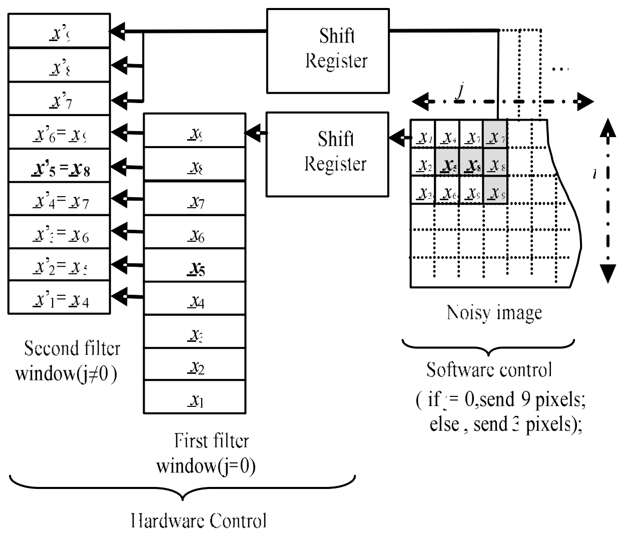

Figure 3. Optimization of the pixels loading.

The total SW code of VMRHF filter was rebuilt and tested using the NIOS-II processor. The execution times have been measured with the timer that provides the number of processor clock cycles. Afterwards, the SW critical parts were implemented in HW in VHDL language. The main idea of our filter implementation is to exploit advantages of the parallel structures which can be efficiently implemented in hardware. The interest of a parallel hardware structure is a reduced number of operation and ability to parallel execution.

4.2. Hardware Part

The amount of hardware required for an accurate implementation is quite large. This fact is due to the use of the L_2 norm [11], which requires the computation of the function in equation 4.

$$\|\underline{\Phi}_1-\underline{\Phi}_3\| = \sqrt{(\underline{\Phi}_{11}-\underline{\Phi}_{31})^2+(\underline{\Phi}_{12}-\underline{\Phi}_{32})^2+(\underline{\Phi}_{33}-\underline{\Phi}_{13})^2} \quad (4)$$

Thus we can exploit this feature in order to reduce the complexity of the system itself.

$$\|\underline{\Phi}_1-\underline{\Phi}_3\| = \frac{|\underline{\Phi}_{11}-\underline{\Phi}_{31}|+|\underline{\Phi}_{12}-\underline{\Phi}_{32}|+|\underline{\Phi}_{33}-\underline{\Phi}_{13}|}{c} \quad (5)$$

where $c = 1$; 3/4 or 2 depending if respectively two, one or none of the addends are equal to 0. This is a coarse approximation [11] which allows us to still have a good estimate for the L_2 norm, but which is much simpler to implement.

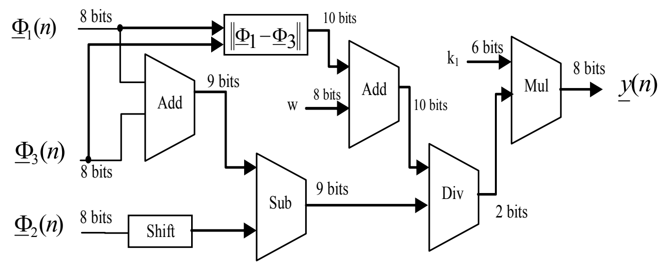

Figure 4. Hardware architecture of VMRHF filter (2nd stage).

The second approximation has been introduced in order to simplify the evaluation of the division. Before explaining the approximation itself, we need to consider the features of the specific rational function used. In equation 1 the parameters h and k are user-specified depending on the application [11]. For our purposes we can rewrite the former equation for the i^th component in the following way:

$$\underline{y}(n) = \underline{\Phi}_2(n) + \frac{k_1\sum_{i=1}^{3}\beta_i\underline{\Phi}_i}{W + .\|(\underline{\Phi}_1(n) - \underline{\Phi}_3(n))\|} \quad (6)$$

with: $k_1=1/k$; $w=h/k$; for $k=0.025$; $h=6$; $k_1=40$; $w=240$; Figure 4 shows functional body of VMRHF hardware part.

5. Experimental Results

The results discussed in this section are based on SW and HW/SW implementations of the VMRHF filter, which are tested on the ALTERA NIOS-II development board with stratix II EP2S60F672C3 FPGA. This FPGA reaches 48352 Adaptive look-up tables (ALUTs), 310KB of Embedded System Blocs (ESBs), 288 DSP blocks and 493 Input/Output Blocks (IOBs) [14, 15]. The results illustrate the tradeoffs between process performance and filtering speed. The whole design is described using VHDL (RTL level) fitted into the FPGA. The summary of these results is displayed in Table 1.

Table 1. The implementation results in stratix FPGA.

	Resources Required	
ALUTs	13,807 / 48,352	(29 %)
Registers	5618	
Pins	136 / 493	(28 %)
DSP block	116 / 288	(40 %)
Fmax (MHz)	60 MHz	

We have determined the filter-processing time before timing's optimization. Both clocks, of the processor core and the system, are set to 60 MHz (≈16 ns period). The totality of VMRHF design requires

29% of Adaptive Look-Up Tables (ALUTs), 5618 registers, 28% of pins and 40% of the DSPs blocks. Finally, the whole design works with 60MHz system clock.

Table 2. Comparison between SW and HW/SW implementation.

	Images	**Noisy**	**SW**	**HW/SW**
PSNR	Fruits	23.49	27,52	27,71
	Butterfly	25,15	28,58	28,82
	Flower	23,88	30,60	30,86
	Peppers	24,08	30,57	30,83
Number of cycles			188281151	2299398
Total time (ms)			3138	38

Table 2 shows PSNR and clock cycles comparisons (necessary to filter different images) using SoftWare (SW) solution and HardWare/SoftWare (HW/SW) solution applied to diverse color images. Different type of images are used in the test (size 176x144x3) and corrupted with a mixed noise (1% impulsive noise and Gaussian noise with variance $\sigma^2 = 100$). These images are displayed in Figure 5 for visual judgment.

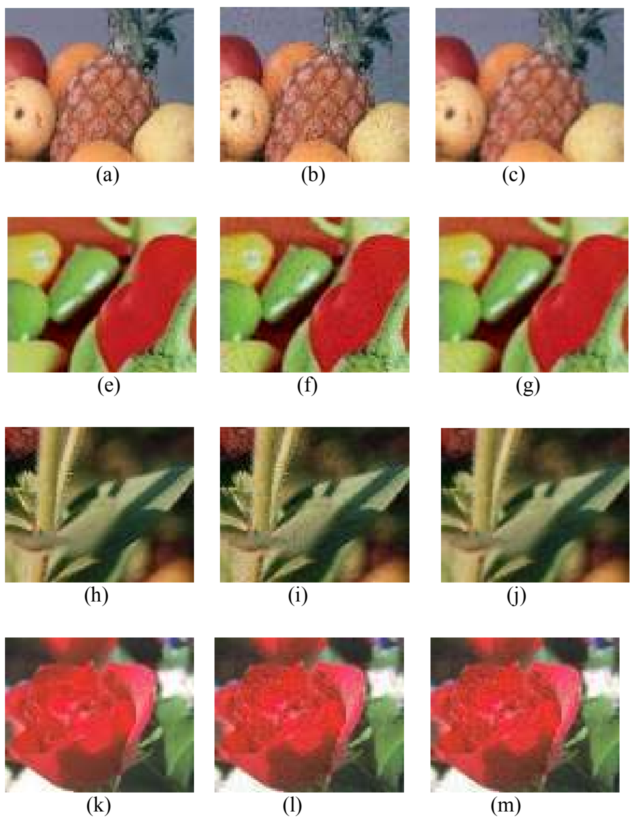

Figure 5. (a, e, h, k) Original images, (b, f, I, l) noisy images, (c, g, j, m) filtered images using HW/SW implementation of VMRHF.

On one hand, we can note that the HW/SW implementation of the VMRHF filter provides a good time improvement in filtering speed compared to the software solution. On the other hand, referring to [4], the VMRHF filter requires more time that the VMF Filter. This difference is caused by the introduction of the rational function in the VMRHF filter. In reality, division implemented in the FPGA, increases unfortunately area occupation and execution time.

6. Conclusion

A new efficient HW/SW codesign implementation of the VMRHF filter based on SoPC system has been described. We have developed a hardware acceleration portion to achieve better filtering speed. So, the actual co-design based implementation constitutes a balance between the two well known requirements: time and area. Our design is working at 60 MHz system clock. The execution time using hardware acceleration is acceptable and can be applied to image processing (26 frames/s) applications that do not need fast processing. Also, better improvement in processing time is possible via different FPGA platforms having higher operating frequency. Finally, we showed that our HW/SW solution improves considerably the filtering speed (82 times faster) compared to software solution.

References

[1] Astola J. and Kuosmanen P., *Fundamentals of Nonlinear Digital Filtering*, Boca Raton, New York, 1997.

[2] Astola J., Haavisto P., and Neuvo Y., "Vector Median Filter," *in Proceedings of IEEE*, Canada, pp. 678-689, 1990.

[3] Ben Atitallah A., Kadionik P., Ghozzi F., Nouel L., Masmoudi N., and Levi H., "An FPGA Implementation of HW/SW Codesign Architecture for H.263 Video Coding," *Computer Journal of Elsevier AEU International Journal of Electronics and Communications,* vol. 60, no. 6, pp. 435-442, 2006.

[4] Boudabous A., Khriji L., Ben Atitallah A., Kadionik P., and Masmoudi N., "Efficient Architecture and Implementation of Vector Median Filter in Co-design Context," *Computer Journal of International Radioengineering,* vol. 16, no. 3, pp. 113-119, 2007.

[5] Cheikh F., Hamila A., Gabbouj R. and Astola J., "Impulsive Noise Removal in Highly Corrupted Color Images," *in Proceedings of International Conference on Image Processing*, Switzerland, pp. 997-1000, 1996.

[6] Cong J., Fan Y., Han G., Jagannathan A., Reinman G., and Zhang Z., "Instruction Set Extension with Shadow Registers for Configurable Processors," *in Proceedings of International Symposium on Field Programmable Gate Arrays*, USA, pp. 25-28, 2005.

[7] Gabbouj L. and Khriji A., "A New Class of Multichannel Image Processing Filters," *Computer Journal of IEICE Transactions on*

Information and Systems, vol. 2, no. 12, pp. 1589-1596, 1999.

[8] Gabbouj L. and Khriji M., "Rational-Based Adaptive Fuzzy Filters," *International Journal of Computational Cognition*, vol. 2, no. 1, pp. 113-132, 2004.

[9] Huang H. and Lee C., "Data Adaptive Filters for Signal and Image Denoising Using a Generalized SURE Criterion," *Computer Journal of IEEE Signal Processing Letters*, vol. 54, no. 4, pp. 561-562, 2006.

[10] Khriji L. and Gabbouj M., "Generalized Class of Nonlinear-Type Hybrid Filters," *in Proceedings of IEE Electronics Letters 5th*, vol. 38, no. 25, pp. 88-96, 2002.

[11] Khriji L., Gabbouj G., Bernacchia M., and Sicuranza G., "Hardware Implementation of the Median Rational Hybrid Filters for Colour Images," *in Proceedings of IEEE EURASIP,* Turkey, pp. 124-128, 2000.

[12] Kim J. and Wills S., "Fast Vector Median Filter Implementation Using the Color Pack Instruction Set," *in Proceedings of IEEE Digital Signal Processing Workshop,* pp. 339-343, 2002.

[13] Moore M., Gabbouj M., and Mitra S., "Vector Filter for Removal of Impulsive Noise from Color Images," *in Proceedings of ECMCS'99 EURASIP Conference DSP for Multimedia Communications and Services Kraków*, Poland, pp. 24-26, 1999.

[14] Nios II Development Kit, http://www .altera.com/ products /devkits /altera/kitniosii 2S30.html. 2006.

[15] Nios II Integrated Development Environment, http:// www.altera.com /literature/ lit-index.html, 2005.

[16] Palaniswamy K., Sinha C., Sharkawy M., Salama P., Lyshevski S., and Gundrum H., "ASIC Memory Design of 2D Median Filters," *in Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems*, pp. 354-357, Australia, 2000.

[17] Pitas I. and Venetsanopoulos N., "Order Statistics in Digital Image Processing," *in Proceedings of the IEEE*, Canada, pp. 1893-1921, 1992.

[18] Pitas I. and Venetsanopoulos N., *Nonlinear Digital Filters*: *Principles and Applications*, Klumer Academic, 1990.

[19] SOPC Builder Applications ALTERA 2006, http://www.altera.com/products/software/products /sopc/sop-index.html. 2006.

[20] xilinx, www.xilinx.com, 2009.

Anis Boudabous received BE degree in electrical engineering from the National Engineering School of Sfax in 2003, and his MS degree in electronics from University of Sfax in 2004, wreath, he is lecturer at the University of Sfax, Tunisia and persuading his PhD studies with the Laboratory of Electronics and Information Technology. His research interests include signal and image processing, nonlinear filtering, hardware implementation using FPGA /DSP, embedded systems technology.

Ahmed Ben Atitallah received his PhD degree in electronics from IMS laboratory, University Bordeaux1 in 2007. He is currently an assistant professor at Higher Institute of Electronic and Communication of Sfax, Tunisia. His main research activities are focused on video signal processing and hardware/software implementation on FPGA target.

Lazhar Khriji received his BS degree in electronics, and his MS and PhD degrees in electrical engineering from University of Tunis II, Tunisia, in 1990, 1992 and 1999, respectively. In 2002, he received the doctor of technology degree in information technology from Signal Processing Institute, Tampere University of Technology, Finland. He is currently associate professor at University of Sousse, Tunisia.

Patrice Kadionik received his ENSEIRB engineer diploma in 1989 and the PhD degree in instrumentation and measurement from the University of Bordeaux, France, in 1992. He is currently associate professor at the ENSEIRB School of Electrical Engineering. His main research activities include system on chip for video compression and for sensor networks and FPGA testing.

Nouri Masmoudi received electrical engineering degree from the Faculty of Sciences and Techniques - Sfax, Tunisia, in 1982, the DEA degree from the National Institute of Applied Sciences-Lyon and University Claude Bernard-Lyon, France in 1984. The National School Engineering of Tunis, Tunisia in 1990.