International Journal of Innovative Research in Electronics and Communications (IJIREC)
Volume 3, Issue 5, 2016, PP 1-5
ISSN 2349-4042 (Print) & ISSN 2349-4050 (Online)
DOI: http://dx.doi.org/10.20431/2349-4050.0305001
www.arcjournals.org

Study of Implementation of Image Analysis with Hardware and Software Co-Design on the Xilinx Platform

S.Karthik [1], M.Sathishkumar [2], N.Kannan[3]

[1,2,3] AP/ECE, Mahendra Institute of Technology, Namakkal, Tamil Nadu, India

Abstract: *Recently image processing applications are normally chosen in such fields as manufacturing computerization, safety, health, and traffic control in parallel with the development in technology. The most significant criterion for the applications used in these fields is to make sure the scheme to run at high speed and real time. Thus, FPGAs are usually used in such applications. In this paper, some of the essential real time images processing algorithms are implemented in a FPGA-based development kit. The Zed Board Znyq-7000 development kit produced by Xilinx Company was used in the study. As a effect of grayscale conversion and convolution operations, such applications as edge detection, sharpening and blurring were realized on the images taken by video source. The processed images were viewed by the use of a monitor coupled to HDMI output of the development kit. Verilog HDL was used for these operations. The RGB to grayscale conversion method is a commonly used method in image processing applications. The motive for the recurrent use of this way instead of colorful images is to decrease the computational requirements of the operations that will be performed on the image. In this means, a higher performance is obtained in these applications. There are more than a few ways of converting a RGB image to a gray image.*

Keywords: *FPGA, Xilinx, image processing.*

1. Introduction

Nowadays, the significance of image processing is quickly increasing in such fields as industrial automation, security, health, and traffic control in parallel with the developments in technology. The most demanding complexity in applications used in these fields is to make system run real-time. It is not forever probable to make the system run real-time by the use of a software used on a general purpose computer since the resources of memory, CPU and peripheral devices in computers are limited. In most image processing applications, dozens of operations are performed on each pixel. That these operations are performed by universal purpose processors sequentially leads to negative consequences in terms of both resource consumption and performance.

However, FPGAs (Field Programing Gate Arrays) has the capability to operate in a parallel way in terms of hardware, which distinguishes them from traditional processors. In this way, operations are divided into pieces in FPGAs and multiple operations can be done simultaneously.

2. Image Processing Algorithms Used in the Study

The RGB to grayscale conversion system is a usually used method in image processing applications. The reason for the normal use of this method as an alternative of colorful images is to reduce the computational requirements of the operations that will be performed on the image. In this way, a higher presentation is obtained in these applications. There are several ways of converting a RGB image to a gray image. In this study, we converted the images to gray by using the formula given in (1). This method was designed in agreement with the compatibility of human eye with brightness perception by using a weighted combination of RGB channels. Moreover, this method is frequently used in computer vision and the “rgb2gray” function of MATLAB also uses this formula.

$$Y\ 0.2989R\ 0.5870G\ 0.1140B \qquad (1)$$

Convolution is another important algorithm usually used in image processing. Convolution is an operation that execute the total of multiplication of image matrix values and kernel matrix values. Convolution is used to apply the image a spatial low and high pass filter. Depending on which kernel matrix is used in convolution, such operations as edge detection, sharpening and blurring can be performed on image.

3. System Design

3.1. The Fpga Card Used

The ZedBoard Zynq-7000 FPGA development kit was used in this study. Zynq-7000 is different from standard FPGAs in that it contains Artix-7 FPGA and ARM Cortex-A9 processor on the matching chip together. In a FPGA card having this system, those parts containing intense computations are performed on FPGA and the control parts not containing computations can be done on the processor by using software. The internal structure of Zynq is shown in Figure. 1. Zynq consists of two parts: giving out System and Programmable Logic. Processing System (PS) works like a traditional processor since it contains structures such as ARM Cortex-A9, Floating Point Unit, Memory Controller, Gigabit Ethernet Controller and USB Controller. Programmable Logic part, on the other hand, contains all the structures of a standard FPGA. The communication between PS and PL parts is provided by high performance data-paths.

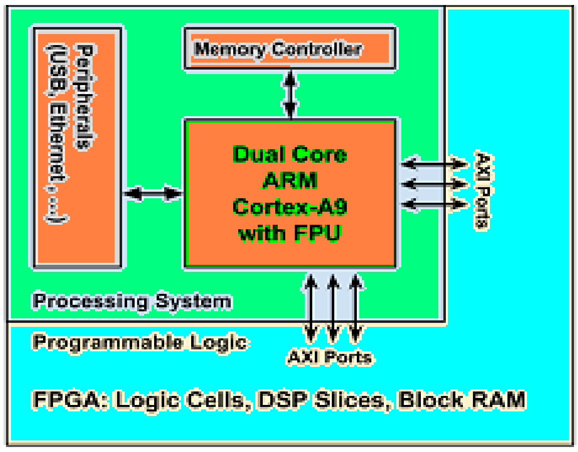

Memory Controller
Peripherals (USB, Ethernet, ...)
Dual Core ARM Cortex-A9 with FPU
AXI Ports
Processing System
Programmable Logic
AXI Ports
FPGA: Logic Cells, DSP Slices, Block RAM

Figure1. *The internal structure of Zynq.*

3.2. General System

The developed system consists of two parts: hardware and software. The software performs the operation of image capturing from the video source and transferring it to the hardware part. Any image source that can run on Linux (such as USB camera, network video stream, video file etc.) can be used in the software. In the hardware part, image processing algorithms are implemented. The schema of the general system fashioned by the co-use of software and hardware parts is shown in Figure. 2.

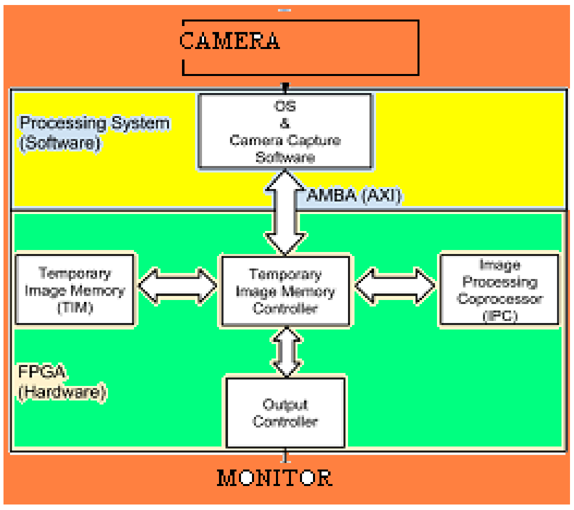

CAMERA
Processing System (Software)
OS & Camera Capture Software
AMBA (AXI)
Temporary Image Memory (TIM)
Temporary Image Memory Controller
Image Processing Coprocessor (IPC)
FPGA (Hardware)
Output Controller
MONITOR

Figure2. *The schema of the general system.*

In the system, video frame is firstly obtained from the video source via software and operating system. Then, the software transfers the parts to be processed on the video frame to the temporary image memory on the FPGA. The transfer of operation, on the hardware part, is controlled by Temporary Image Memory Controller (TIMC) module. The video is transferred to Temporary Image Memory (TIM) module via the TIMC.

The processed and untreated forms of the image can be stored in the TIM module. Image Processing Co-processor (IPC) module performs image giving out algorithms. The IPC module performs the image dispensation operations by fetching the pixels from TIM module and storing the processed forms of the pixels. These modules are explained in the 4th part. After the operations on the image are accomplished on the hardware part, the status statement is transmitted to the software. Following this stair, the processed image can be read via the software part if needed. The original and processed forms of the image are shown on the monitor by using the HDMI output of FPGA kit.

3.3. Hardware/Software Interface

In the study, Advanced Extensible Interface (AXI), which is an interface standard of Advanced Microcontroller Bus Architecture (AMBA) is used, for the connection between Processing System (PS) and Programmable Logic (PL). AXI is the whole of the standardized protocols which enable the data transfer among hardware modules. AXI can work in two unlike ways as memory mapped and stream. Memory mapped AXI is used in this study. Memory mapped AXI works by connect the memory and registers in the module with memory addresses on the part of PS. The right of entry to PL by PS is in the form of reading and writing operations on the associated address. In the study, two dissimilar AXI connections, image transfer and control, were used for the statement between PS and PL. In the AXI connection used for image transfer, it is possible to access the whole of impermanent Image Memory. The register that shows the progress of processing is also available on the control connection.

The AXI connection used for image transfer was run at 32 bit mode. Image in 24 bit RGB format was obtained from the camera, but it was transferred to FPGA as 32 bit. The reason for the transfer of pixels as 32 bit is to ensure one pixel transfer at each clock cycle. In this way, the address transformation in the hardware part and the read and write operations in the memory were implemented more easily and by using less resource. The alteration from 24 bit to 32 was performed as the resetting the highest 8 bit. Thus, the need for additional operations on the pixels in the software part was eliminated.

3.4. Software

Video capturing from sources such as USB camera, network video stream and video file is a very complicated and difficult operation on FPGAs when compared to the software. Moreover, for video capturing, it is essential to design special hardware for the video source. The aim of the system we developed is to perform real time operations on the video rather than video capturing itself. Consequently, using software for video capturing enabled us a great deal of expediency. The ARM processor on the development kit we used can run an operating system. Linux operating system and user space libraries provide a standard border for the hardware. By this means, it is possible to right to use video sources easily via the software. Thus, software developed on the Linux operating system was use in video capturing part. The software we developed captures images from the camera at sure intervals according to the frame rate used. After each pixel of the image to be process is transferred to hardware part, the operation is expected to be completed by doing busy waiting with software. When the function is complete, the hardware part notify the software about the completion status of the operation with a flag.

Then, the software passes to the next portion in the frame if it is obtainable. After all the pixels in the frame were processed, the next frame is transfer from the camera and the same operations are repeated. Each frame was used as a single image portion in the test system. This study was implemented by capturing images from USB camera via a software developed in C programming language by using a Video for Linux (v4l2) [10] library. Different image resources can also be used in the study by making changes only on the software and without any change in the hardware design. For example, network video stream or a video file on the memory card can be used as video source

4. Hardware Design

Image processing methods are performed by IPC module. TIM module consists of two memory units storing original images and the processed images. During the image dispensation operations, required pixels of the original image are read from TIM and processed pixels are written back. TIMC functions as a bridge during these operations. In Figure. 3, the symbolic diagram of the modules used in image processing and their contents are presented.

In this module, two memory systems of 24 bit width and pixel number depth were used. One of the memories stores the original form of the image while the other one stores the processed image. Address ranges of both memories are between zero and pixel number. Memories are addressed as 24 bit format rather than byte format.

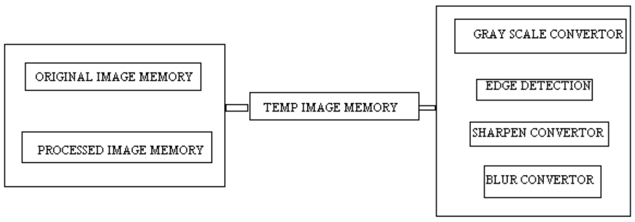
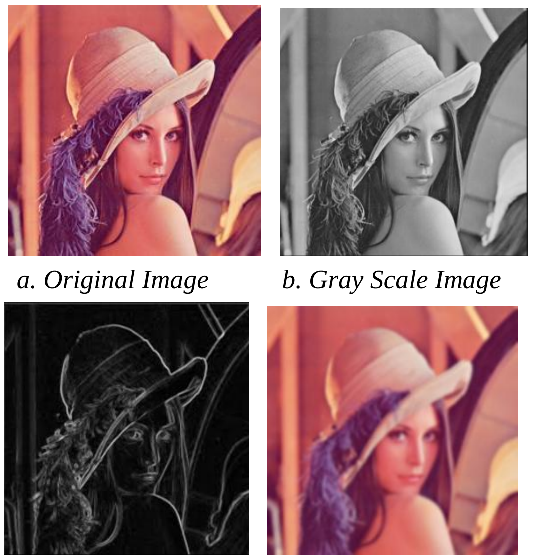


Figure3. *The symbolic diagram of the modules.*

As the used image format is 24 RGB, each memory cell was designed to capture one pixel. Memories were implemented in the format of "Dual-Port RAM with Synchronous Read". There are two ports in the RAM in this system. One of the ports can only perform reading operations while the other can perform both reading and writing operations. The use of a RAM with two ports enabled us to performed video processing operations and displays the images on the screen simultaneously.

5. Experimental Results

In following section the photo of the test system taken throughout the study is obtained. ZedBoard, HDMI monitor, standard USB camera and USB HUB were used in the test system. The design was implemented in the Xilinx ZedBoard Zynq-7020 FPGA by Vivado 2014.2. The convolution kernel being used can be changed during the study. In Figure 4, screen prints of some method used are shown.



a. Original Image *b. Gray Scale Image*

c. Edge detection *d. Blurring*

Figure4.

The operations were performed on a 256×256 image and difficulty coprocessor clock frequency was used as 150 Mhz in the developed system. The time spent for each frame is approximately 0.44 ms. Convolution coprocessor has a design that has the capacity to run at a higher speed than 2000 fps (frame per second). The transfer of frame takes roughly 25 ms with the implementation of the AXI we used. Therefore, around 40 fps speed could be reached in real time system.

6. Conclusion

This manuscript presents the realization of real time image processing algorithms with Hardware / Software Co-design on FPGA. At the end of the study, a real time working system of 256×256 image and around 40 fps were obtained. Although convolution coprocessor design is adequate for working on large scale images, this study is limited with small scale images. The motivation for that is the inadequacy of image transfer design to FPGA we developed. For further study, it is aimed to speed up the image transfer to FPGA and, by this means, work on high resolution images real time by the use of Direct Memory Access (DMA) implementation.

REFERENCES

[1] Przemysław Kupidur,"Semi-automatic method for built-up area intensity survey using morphological granulometry",pp271-277, 2010.

[2] Abhishek Acharya, et al. "FPGA Based Non Uniform Illumination Correction in Image Processing Applications Vol 2", pp349-358, 2009.

[3] Soohwan Ong and Myung H. Sunwoo, A Morphological Filter Chip Using a Modified Decoding Function, IEEE Transactions on circuit and systems-II: Analog and digital signal processing, vol. 47, no. 9, September 2000.

[4] Luca Breveglieril, Vincenzo piuri, Digital Median Filters, Journal of VLSI Signal Processing 31, 191–206, 2002

[5] Komal Vij,et al. "Enhancement of Images Using Histogram Processing Techniques Vol 2" , pp309313, 2009.

[6] M Rama Bai,"A New Approach For Border Extraction Using Morphological Methods", pp3832-3837,2010.

[7] T. M. Lehmann, C. Gonner, and K. Spitzer,"Survey: Interpolation Methods in medical image processing," IEEE Trans. Med. Imaging, vol.18, no. 11, pp. 1049–1075, Nov. 1999.

[8] konstantinides,V.Bhaskaran,G.Beretta," Image Sharpening in the JPEG domain",IEEE.Trans, on Image processing, vol.8(6),pp 874-878,1999. [9] S. Ridella, S. Rovetta, and R. Zunino, "IAVQ-intervalarithmetic vector quantization for image compression," IEEE Trans. Circuits Syst. II, AnalogDigit. Signal Process., vol. 47, no. 12, pp. 1378–1390, Dec. 2000.

[9] H.C.Kim, B.H.Kwon, and M.R. choi, "An image interpolation with image improvement for LCD controller", IEEE.Trans consumer Electron, vol.47,pp 263-271,May-2001.

[10] C. H. Kim, S. M. Seong, J. A. Lee, and L. S. Kim, "Winscale : An imagescaling algorithm using an area pixel model," IEEE Trans. Circuits Syst.Video Technol., vol. 13, no. 6, pp. 549–553, Jun. 2003.

[11] S. Schaller, J. E. Wildberger, R. Raupach, M. Niethammer, and K. Klingenbeck-Regn, "Spatial domain filtering for fast modification of the tradeoff between image sharpness and pixel noise in computed tomography," IEEE Trans. Med. Imag., vol. 22, no. 7, pp. 846–853,Jul. 2003.

[12] C. Weerasnghe, M. Nilsson, S. Lichman, and I. Kharitonenko, "Digital zoom camera with image sharpening and suppression," IEEE Trans.Consumer Electron., vol. 50, no. 3, pp. 777–786, Aug. 2004.

[13] Lei Zhang, and Xiaolin,"An Edge Guided Image interpolation Algorithm via Directional Filtering and Data Fusion",IEEE.Trans,image processing,pp no 105-149,2006.

[14] Q. Wang and R. K. Ward, " A new orientation-adaptive interpolation method," IEEE Trans. Image Process., vol. 16, no. 4, pp. 889-900, Apr. 2007.

[15] S. Saponara, L. Fanucci, S. Marsi, G. Ramponi, D. Kammler, and E. M. Witte, "Application-specific instructionset processor for Retinexlink image and video processing," IEEE Trans. Circuits Syst. II, Exp.Briefs, vol. 54, no. 7, pp. 596–600, Jul. 2007.

[16] C. C. Lin, Z. C. Wu, W. K. Tsai, M. H. Sheu, and H. K. Chiang, "The VLSI design of winscale for digital image scaling," in Proc. IEEE Int. Conf. Intell. Inf. Hiding Multimedia Signal Process., Nov. 2007,pp. 511–514.

[17] C. C. Lin, M. H. Sheu, H. K. Chiang, W. K. Tsai, and Z. C. Wu,"Real-time FPGA architecture of extended linear convolution for digital image scaling," in Proc. IEEE Int. Conf. Field-Program. Technol., 2008,pp. 381–384.

[18] A. Amanatiadis and I. Andreadis, "An integrated architecture for adaptive image stabilization in zooming operation," IEEE Trans. Consumer Electron., vol. 54, no. 2, pp. 600-608, May 2008.

[19] C. C. Lin, M. H. Sheu, H. K. Chiang, C. Liaw, and Z. C. Wu, "The efficient VLSI design of BI-CUBIC convolution interpolation for digital image processing," in Proc. IEEE Int Conf. Circuits Syst., May 2008,pp. 480–483. IJSER