IMPLEMENTATION OF 2D CONVOLUTION ALGORITHM ON FPGA FOR IMAGE PROCESSING APPLICATION

[1]SRIRAM V.B., [2]PRASAD SAWANT, [3]KAVIT KAMATH, [4]KANIKA WADHWA, [5]GANESH GORE, [6]SNEHA REVANKAR

[1,2,3,4,6]Electronics and Telecommunication Department,
Fr. Conceicao Rodrigues Institute of Technology, Vashi, Navi Mumbai, Maharashtra, India
[5]Industry Expert
E-mail: [1]vbsriramvb@gmail.com, [2]prasadsawant7@gmail.com, [3]kamath.kavit@gmail.com, [4]kanika12395@gmail.com, [5]ganesh@eduvance.in, [6]sneha.revankar@gmail.com

Abstract— Image processing is a growing field with tremendous potential and scope for development. With the advent of advanced visual technologies, there is a need to have an ultra high speed processing machines to match the quality of the high definition domain. The high end DSPs have drawbacks and limitations which can be addressed by implementing a processing unit as a hardware design. An optimum architecture can be developed by prototyping it on Field Programmable Gate Arrays. An ideal hardware architecture can be designed according to the specific requirement which can outperform the more generic processors.

Keywords— Convolution, FPGA, Image processing, Pipelining.

I. INTRODUCTION

Image processing is a part of day to day life of every individual. High technological advancements in the field of image processing have led to many startling results. Over several years of development greater degree of enhancement of images, real time processing along with a large amount of data processed due to high resolution images in every frame.

For such large amount of pixel processing, a generic processor like will not suffice the speed required. The main drawback of a controller or a processor is that it is sequential in nature, i.e. it follows a specific path for performing various operations. The operation flow is non adaptive. This would not be conducive for real time processing where speed of processing is the most critical factor. Parallel processing of data is a need for real time processing. Parallel processing of data would increase the data computation rate. An application specific flow of operation should be undertaken which would provide real time image processing with the desired output and without showing any noticeable degradation of quality due to any processing delay. Application Specific Integrated Circuits are an ideal option provided that is a perfect hardware architecture which suffices all for such requirements. The data flow and its processing could be completely customized to ensure the best of quality without any kind of operational delay or error that would result in any degradation. The ASICs are permanent hardwired devices which are only practical if there is a perfect architecture and mass production. But for prototyping purposes, to design the ideal hardware architecture FPGAs are used.

Field Programmable Gate Arrays are a hardware programmable device that consists of configurable logic blocks and interconnects. These CLBs as well as interconnects are programmable. Hardware descriptive languages such as VHDL and Verilog are used to program the FPGA. According to the HDL code, the hardware is created on the FPGA. Thus FPGAs can create customized hardware according to the need and the flow of operation too. Parallel processing of data can thus be incorporated into the hardware design.

II. PLATFORM USED

2.1. Features

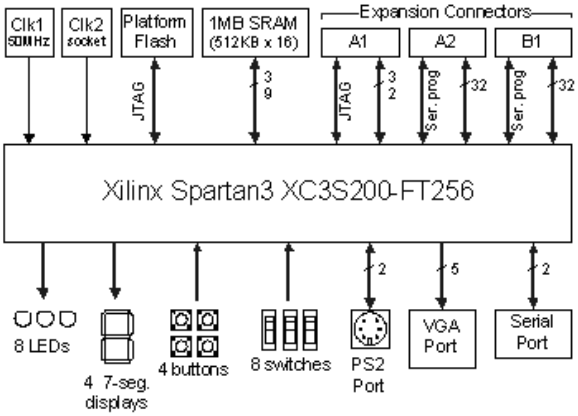

Fig.1. Basys2 board block diagram

The paper was implemented using Basys2 Digilent board.

The Basys2 board is built around a Xilinx Spartan-3E Field Programmable Gate Array. [ref5]Atmel AT90USB2 Full-speed USB2 port is used for delivering both programming interface and power supply. The board consists of 4 push buttons, 8 slide switches, 8 LEDs and 4 seven segment displays for user I/O. It has ESD protection for all I/O signals. PS/2 port is available for interfacing PS/2 mouse or keyboard. An 8-bit VGA port is available for interfacing the VGA monitor.

The board has user settable clock speed of 25MHz, 50MHz or 100MHz. It also consists of 6-pin header expansion connectors.

2.2. Resources Utilized:

The paper aims at demonstrating image processing using FPGA. The 8-bit VGA connector was connected to a VGA monitor compatible with the given pixel resolution. The code was burnt using Digilent Adept which can be used to directly program the Basys2 board. The USB2 port is used as the programming interface and for power supply. The code was run using 50MHz in-built clock.

III. IMAGE FILTERING

Image filtering is a method of image enhancement. Image filtering is a use of filters to perform a specific task by which the entire image or a region of interest is enhanced or modified. An image kernel can be used for filtering of an image. A kernel is a small matrix which can be convolved with the image. There are different types of kernels for different purposes [ref1]Designing of a kernel is a complex mathematical process. Though various readymade kernels with predefined values are available for various tasks like image sharpening , blurring, edge detection, etc. In images a low pass filter, smoothens the edges in an image, as edges or a sudden discontinuity in an image is taken as a high frequency component. Thus low pass filtering would try to remove these high frequency components and thus it smoothens the edges. On the contrary a high pass filter would enhance the edges of the image.

IV. CONVOLUTION

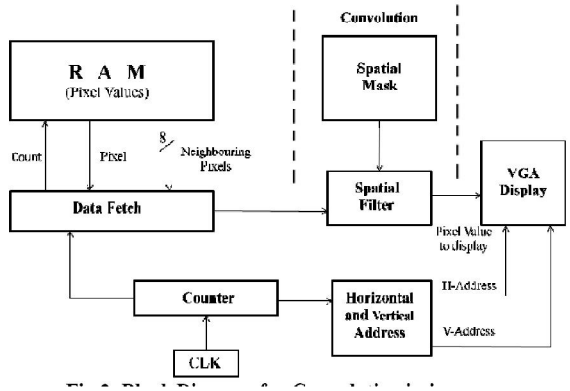

Fig.2. Block Diagram for Convolution in images.

4.1. Image storage

Image filtering was initially performed on an image stored in the FPGA memory. The image was of the size 16*16. The 256 pixel values were stored in a .dat file format. Every pixel value was represented in the eight bit pixel format.

The image was stored in the form of a linear array in the FPGA. Each position consisted of eight bit data representing the pixel value. The image was created using progressive scanning. The first 16 pixel values represent the first line of the image from left to right. Similarly the next 16 pixel values represent the second line of the image.

4.2. Data fetch

The data fetch unit takes input from the counter. [ref2,3]The counter value represents the location of the pixel in the memory where the image is stored and also the location of the pixel in the processed image. According to the counter value, the data fetch unit identifies the pixel location in the image and fetches the respective neighboring pixels from the FPGA memory for convolution along with the pixel value pointed out by the counter.

4.3. Counter

The counter is an eight bit counter used to point out the pixel locations in the memory. The same counter value can also be used to evaluate the pixel location in the processed image. The last 4 bits (least significant bits) of the counter value represent the horizontal location of the pixel value and the first four bits (most significant bits) represent the vertical address.

4.4. Image filter

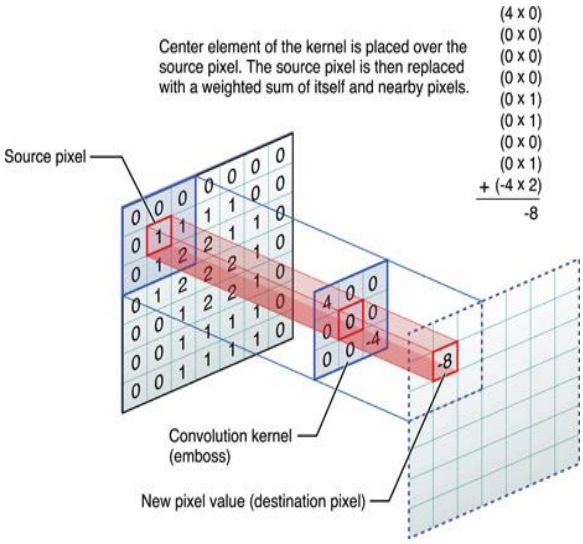

Fig.3.Convolution of image and the image kernel

This block is the core of the entire system. The image is processed in this block. [ref4]The input to this block is the central pixel value to be processed and along with its neighboring pixels. The output of the image filtering block is the pixel value to be displayed. If the image pixels are represented by f(x, y) where x and y are the pixel co-ordinates and the kernel is represented by h(x, y), F(x, y) is the convolved output pixel given by the equation below.

$$F(x,y) = \frac{\sum_{x=1}^{3}\sum_{y=1}^{3} f(x,y).h(x,y)}{\sum_{x=1}^{3}\sum_{y=1}^{3} h(x,y)}$$

4.5. VGA Display

The image was displayed with the help of a VGA connector. The pixel value is of the eight bit pixel format. This pixel values along with the horizontal and vertical address of the pixel is sent to the VGA monitor display unit, This unit sends information to the monitor for displaying the processed image.

V. RESULTS

Two filtering operations were implemented using convolution matrix kernels.

5.1. All Pass Filter

0	0	0
0	1	0
0	0	0

Fig.4. All pass filter image kernel

An all pass filter in the spatial domain is fed as input to the convolution block along with the image pixel values. The output of the processed image will completely resemble the original image.

5.2. Low Pass Filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Fig.5.(a).Averaging low pass filter kernel

A low pass filter in general smoothens the edges in the images since edges are taken as high frequency components in the images. Low pass filtering action is achieved by using an averaging filter image kernel. Better results can be obtained if more emphasis is towards the central pixel during the convolution operation. In averaging filter, the filter values corresponding to the image have equal values. Thus a weighted average filter is used which is shown in Fig 5.(b). The convolution of filter and the image smoothens the edges of the filters.

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

Fig.5.(b)Weighted Averaging low pass filter kernel

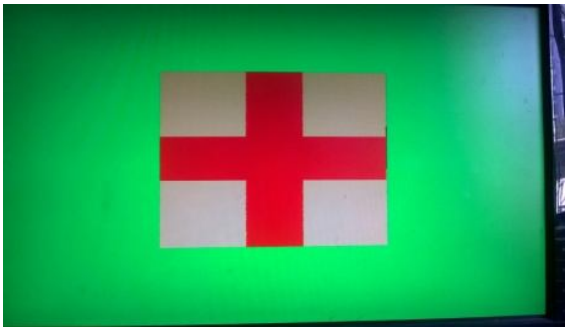

Fig.6.(a)Image before convolution

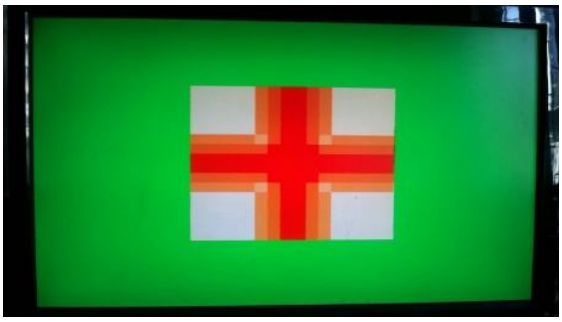

Fig.6.(b) Image after convolution

VI. ADVANTAGES

The main advantage of implementing this architecture on FPGA is its ability to perform parallel processing. Programming in HDL creates separate hardware for every unit i.e. the data fetch unit functions are independent of the display unit. This parallel processing is a very important parameter in image processing as it improves the overall speed.

The maximum output time required after clock is 5.962ns. This allows us to use higher clock rates for given image resolution which will result in more frames per second for persistence of vision.

Another advantage of FPGA is the flexibility it provides in changing the hardware. This flexibility can be harnessed for creating the most optimum architecture.

VII. LIMITATIONS

The propagation delay was found to be 5.962ns for a 16*16 image resolution. This delay will increase with an improved pixel resolution. Thus, there will be a constraint in the clock speed that can be used with the increase in resolution.

The convolution techniques used provide satisfactory results for still images. For processing images in real time, the processing time should be minimum to maintain persistence of vision.. For constructing an architecture with minimum delay, the hardware used should be minimized as some propagation delay is introduced by every hardware component.

Hence, there was a need for an architecture which can support high pixel resolution in real time.

VIII. PIPELINING ARCHITECTURE

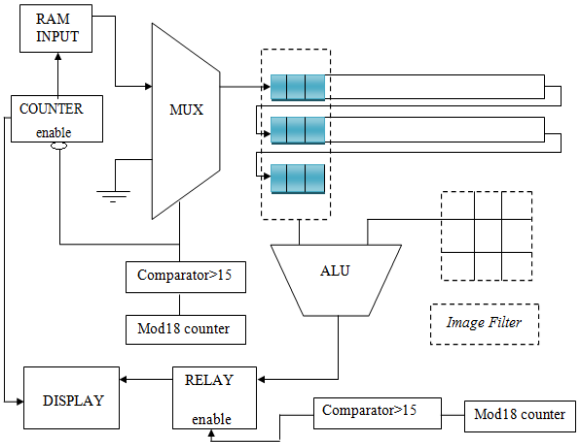

Fig.7. Pipelined architecture

In convolution the kernel mask is made to shift over the entire image. Different pixel values are given to it as input and the output is computed respectively. In pipe-lining shift registers are used to move the pixel values sequentially. The kernel position remains fixed. This eliminates the supplementary hardware.

The input to the shift registers comes from the image pixel values. After every line, 2 zeros are inserted into the shift registers. This is implemented with the help of a multiplexer as shown in the figure. This is to perform zero padding on the image to enable spatial domain filtering on the entire image.

Due to the insertion of zeros in the shift registers, the zeros will also pass through the central position of the kernel. Hence a switch is needed to control the output going to the display unit. After the display of every line the output has to wait for 2 clock cycles.

Implementation of pipe-lining in HDL will need state machines. Two state machines need to be running simultaneously for both inserting zeros in shift registers and for pausing the filtering unit. With the implementation of pipe-lining architecture the overall propagation delay is reduced. This allows the use of a faster clock for further implementations.

The difference between time delays of convolution architecture with and without pipe-lining is negligible for a 16*16 image, but for bigger images with higher resolutions, the hardware created with pipe-lining would be saving a lot of computational time.

IX. FUTURE SCOPE

9.1. Real Time Image Filtering

Performing image processing in real time requires processing the pixels at a speed high enough to maintaining the persistence of vision for the entire image. This means to convolve all the pixels within a very short time span. This requires architecture with an optimum and dedicated hardware.

The pipelined architecture provides means for image filtering using less hardware. An FPGA allows parallel processing. Thus multiple units of pipelined architecture can be implemented for making it possible for processing images in real time.

ACKNOWLEDGMENTS

We are grateful to our college faculty who were a constant source of inspiration and encouragement for writing this paper. We would like to thank our college for giving us the exposure to Eduvance who were instrumental in teaching us about FPGA technology. Remo Lab offered by Eduvance was of an immense help in verifying the results of our work.

REFERENCES

[1] Gonzales and Woods, "*Digital Image Processing*", Pearson Education, India, Third Edition

[2] Clement Fabaret, Cyril Poulet, Jefferson Y Han, Yann LeCun, *"CNP: An FPGA-based processor for convolutional networks"*

[3] Ben Cope, "*Implementation of 2D Convolution on FGPA, GPU and CPU*"

[4] https://developer.apple.com/library/ios/documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html (Last accessed: 17.10.2015)

[5] www.digilentinc.com (Last accessed:20.10.2015)

★★★