Hardware Implementation of 2D convolution on FPGA

Conference Paper · March 2023

CITATION
1

READS
865

2 authors, including:

Shefa Dawwd
University of Mosul
60 PUBLICATIONS **538** CITATIONS

SEE PROFILE

Hardware Implementation of 2D convolution on FPGA

Shefa A. Dawwd, Faris S. Fathi
Computer Engineering Department, College of Engineering, University of Mosul,Iraq
shefadawwd@uomcoe.org

Abstract—The aim of the work proposed in this paper is to present a general architecture for the systolic array that can be used in implementing the operations of the two-dimensional convolution. Two techniques (pipeline and parallelism) are used in designing architecture to achieve maximum efficiency and to increase the throughput. Moreover, the dual port FPGA RAM is used in implementing the first in-first out (FIFO) buffer which makes the designed architecture more flexible in processing images of different sizes; enlarging their size without changing the size of the circuit buffer. So that the silicon used area are reduced. The adder tree technique is used to implement addition operations of the convolution. That is to reduce the computing time through parallel processing and to start the subsequent operations before finishing the ongoing ones. Two designs are proposed for the systolic array architecture. A comparison between them with respect to building area and the number of frames per second is performed. Each of the design architecture is modeled using VHDL and implemented on Spartan3-E FPGA chip. It is found that the throughput of the second proposed architectures become twise (380 frame/sec) the throughput of the first proposed one (190 frame/sec). However, the second proposed architecture which processes two-subsequent windows in one clock requires an area less than that of the first one.

***Index Terms*—Convolution, FPGA digital filter, VLSI .**

I. INTRODUCTION

Modern image processing and computer vision algorithms require high computational capability especially when high resolution images have to be elaborated under real-time requirements. In such applications (e.g. image filtering, image restoration, feature recognition, object tracking, template matching, etc.) the spatial domain two-dimensional (2-D) convolution plays a fundamental role. Therefore, the design of efficient convolvers receives great interest for both commercial and military purposes [1].

Implementing the mentioned applications on a generable purpose computer can be easier but not very efficient in terms of speed. The reason being the additional constraints put on memory and other peripheral device management. Application specific hardware offers much greater speed than a software implementation. There are two types of technologies available for hardware design. Full custom hardware design also called as Application Specific Integrated Circuits (ASIC) and semi custom hardware device, which are programmable devices like Digital signal processors (DSP's) and Field Programmable Gate Arrays (FPGA's). Full custom ASIC design offers highest performance, but the complexity and the cost associated with the design is very high. The ASIC design cannot be changed; time taken to design the hardware is also very high. ASIC designs are used in high volume commercial applications. In addition, if an error exist in the hardware design, once the design is fabricated, the product goes useless.

DSP's are a class of hardware devices that fall somewhere between an ASIC and a PC in terms of the performance and the design complexity. DSP's are specialized microprocessor, typically programmed in C, perhaps with assembly code. It is well suited to extremely complex math intensive tasks such as image processing. Hardware design knowledge is still required, but the learning curve is much lower than some other design choices. Field Programmable Gate Arrays are programmable devices. They are also called reconfigurable devices. Reconfigurable devices are processors which can be programmed with a design, and the design can be by reprogramming the devices. Hardware design techniques such as parallelism and pipelining techniques can be developed on a FPGA, which is not possible in dedicated DSP designs. So FPGAs are ideal choice for implementation of real time image processing algorithms[2].

The last few years have seen an unprecedented effort by researchers in the field of image processing based 2D convolution using FPGA. In 2000, A. Nelosn [3] implemented some of image processing algorithms on Altera and Xilinx FPGA for real time video application. In 2005, S. Wong et al [4] described a parallel pipelined architecture for 2D convolution written in Handel-C for a reconfigurable computing platform. The resulting system offered a 400 times increase in speed over software written in C. In the same year, G. Saldana et al [5] presented a work which focuses on the development of a reconfigurable systolic-based architecture for low level image processing. The architecture is customizable providing the possibility to perform window operations for masks of 3x3, 5x5 and 7x7 coefficients. The results show that window-based operations can be performed in real time, processing an image frame in 5 ms, achieving a throughput of around 3.6 GOPS. In 2006, D. Venkateshwar et al [6] proposed an FPGA image processor operates in high throughput and consumes low silicon area. The processor consumes 479 slices for mask of 3x3. A 10,752 slices are used in the work of Zhang and Asari. In this work a fully pipelined multiplierless digital architecture for computing 2D

convolution utilizing the quadrant symmetry of the 22x22 kernels is presented [7]

The aim of the work proposed in this paper is to present a general architecture for the systolic array that can be used in implementing the operations of the two-dimensional convolution. The architecture is evaluated by comparing it with some previous works in term of resolution, speed and hardware recourses.

II. Convolution and Filtering

A. 1D Convolution

The *convolution operation* is a mathematical operation which takes two functions $f(x)$ and $g(x)$ and produces a third function $h(x)$. Mathematically, convolution is defined as[8]:

$$h(x) = f(x) * g(x) = \int_{-\infty}^{\infty} f(\tau).g(x-\tau)d\tau \qquad (1)$$

$g(x)$ is referred to as the *filter*.

B. 2D Convolution

2D-Convolution, is most important to modern image processing. The basic idea is that a window of some finite size and shape is scanned over an image. The output pixel value is the weighted sum of the input pixels within the window where the weights are the values of the filter assigned to every pixel of the window. The window with its weights is called the *convolution mask*. Mathematically, convolution on image can be represented by the following equation [9]:

$$y(i, j) = \sum_{l=0}^{M-1} \sum_{k=0}^{N-1} h(k,l)\, x(i-k, j-l) \qquad (2)$$

C. Moving Window Operation

The algorithms implemented in this work use the moving window operator. The moving window operator usually process one pixel of the image at a time, changing its value by some function of a local region of pixels (covered by the window). The operator moves over the image to process all the pixels in the image. Each pixel in the output image is produced by sliding an N x M window over the input image and computing an operation according to the input pixels under the window and the chosen window operator. The result is a pixel value that is assigned to the centre of the window in the output image, as shown below in Figure 1.

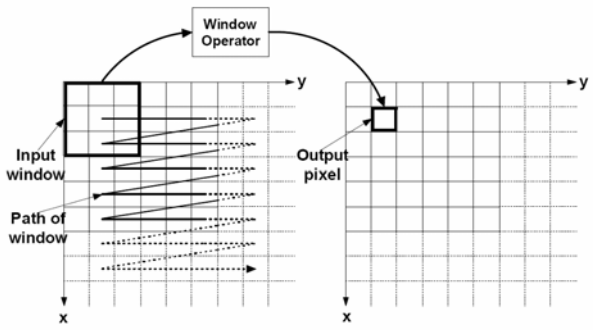

Figure 1. Conceptual example of 3x3 window operator

D. 1D and 2D Finite Impulse Response Filter (FIR)

According to their impulse response, filters are divided into those with a limited impulse response(FIR=Finite impulse response) and those with an impulse response of infinite length(IIR=infinite impulse response). For an FIR filter with an impulse response comprising N samples[9]:

$$y(i) = \sum_{i=0}^{N-1} h(k)x(i-k) \qquad (3)$$

The corresponding system function is

$$H(z) = \sum_{k=0}^{N-1} h(k)z^{-k} \qquad (4)$$

from eq.(4), we can simply draw the structure of FIR filter as shown in Figure 2

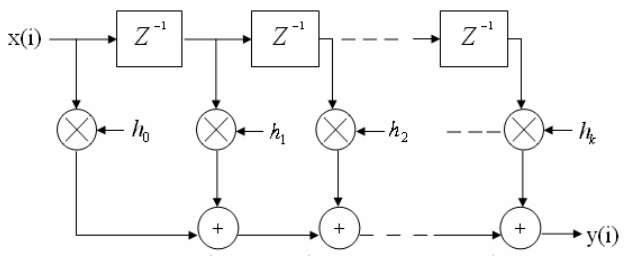

Figure 2. The structure of 1D FIR filter

In the area of image signal processing, filters are used that carry out filtering in both dimentions(horizontal and vertical). Both image data and impulse response are doubly indexed accordingly. Thus, the convolution of an impulse response must be extended to two dimensions as shown in eq.(2). Its Z-transform must also be formulated in two dimentions. The 2D Z-transform of eq.(2) is:

$$Y(z_1,z_2)=H(z_1,z_2).X(z_1,z_2) \qquad (5)$$

In this case, the system function is:

$$H(z_1,z_2) = \sum_{l=0}^{M-1}\sum_{k=0}^{N-1} h(k,l)z_1^{-k}z_2^{-l} \qquad (6)$$

The indexing is chosen such that the first index is applied to the horizontal direction, the second to the vertical. The term z_1^{-1} and z_2^{-1} represent delays of one sampling interval in the horizontal or vertical direction accordingly. Thus, in row-sequential data input, as is generally used in image signals z_2^{-1} is the delay of one image row.

The inner sum of eq.6 can be interpreted as 1D filter for a fixed value of the variable l as:

$$H_l(z_1) = \sum_{k=0}^{N-1} h(k,l)z_1^{-k} \qquad (7)$$

Substitution in eq.6 leads to:

$$H(z_1, z_2) = \sum_{l=0}^{M-1} H_l(z_1) z_2^{-l} \qquad (8)$$

This means that the 2D filter is implemented by M 1D filters arranged vertically as given by delay z_2^{-1} . Figure 3 shows the resulting filter structure.

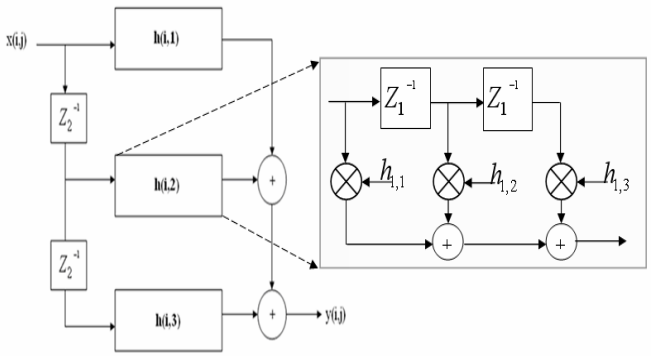

Figure 3. Implementation of a 2D filter through M parallel 1D filters (M=3).

III. CONVOLUTION HARDWARE IMPLEMENTATION

A. *Systolic Array for Single Moving Window*

Architectures of high flexibility are proposed in this paper. Pipelining, parallelism and systolic techniques are used to make the computation performs in high efficiency. Moreover, different image processing algorithms are taken into account to be implemented using the proposed architectures.

For processing purposes, the straightforward approach is to store the entire input image into a frame buffer, accessing the neighborhood pixels and applying the function as needed to produce the output image. If real-time processing of the video stream is required, N x M pixel values are needed to perform the calculations each time the window is moved and each pixel in the image is read up to N x M times. Memory bandwidth constraints make obtaining all these pixels each clock cycle impossible unless some form of local caching is performed. Input data from the previous 1 N _ rows can be cached using a shift register (or FIFO buffer) for when the window is scanned along subsequent lines. This leads to the block diagram shown in Figure 4.

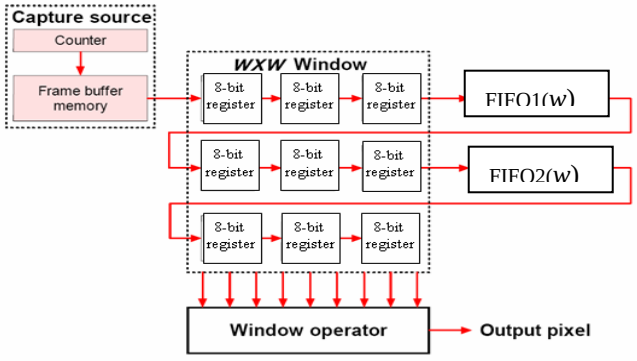

Figure 4. Block diagram for hardware implementation of systolic array for single moving window

Note that, instead of sliding the window across the image, the above implementation now feeds the image through the window.

For the pipelined implementation of image processing algorithms all the pixels in the moving window operator must be accessed at the same time for every clock. In order to access all the pixels in a moving window system, a design was devised that took advantage of certain features of FPGAs. The First In First Out(FIFO) buffers are used to create the effect of moving an entire window of pixels through the memory for every clock cycle. A FIFO consists of a block of memory and a controller that manages the traffic of data to and from the FIFO. The FIFO's are implemented using circular buffers constructed from multiport block RAM with an index keeping track of the front item in the buffer. The availability of multi-port block RAM in the Xilinx Spartan 3E FPGA helps in achieving the read and write operations of the RAM in the same clock cycle. This allows a throughput of one pixel per clock cycle. The same effect can be achieved using double-width RAMs implemented in lookup tables on the FPGA. However, the use of block RAMs is more efficient and has less

associated logic for reading and writing. For a 3x3 moving window two FIFO buffers are used. The size of the FIFO buffer is given as w-3, where w is the width of the image. To access all the values of the window for every clock cycle the two FIFO buffers must be full. Figure 5 shows the architecture of the 3x3 moving window. For every clock cycle, a pixel is read from the RAM and placed into the bottom left corner location of the window.

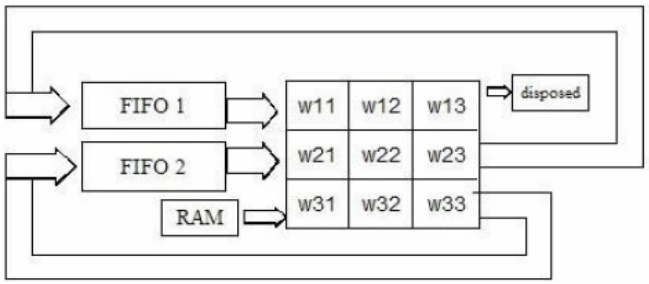

Figure 5. Architecture of 3x3 moving window

B. *Systolic Array Architecture for 2D Convolution*

An efficient architecture of high performance is required in the applications that use the 2D convolution to be processed in real time. For this purpose, a systolic array architecture is presented in this work (Figure 6).

The algorithms implemented in this work use the moving window operator. The moving window operator usually processes one pixel of the image at a time, changing its value by some function of a local region of pixels (covered by the window). The operator moves over the image to process all the pixels in the image. A 3x3 moving window is considered in Figure 6.

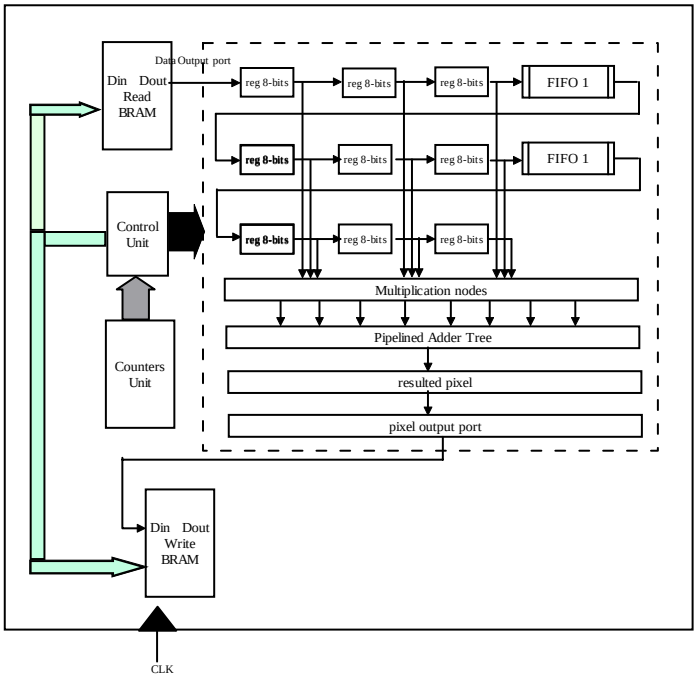

Figure 6. The architecture of 2D convolution for 3x3 moving window operator

In addition to the Block RAM used for the multiplier, there are two other block RAMs that are used to store the image to be processed and the resulted pixels consequently. A control unit is used to provide addresses and other control signals to the entire block RAMs and computation units used.
When convolution on 8-bit pixels is required, each multiplier performs one 8×8 pixel-weight product. In this system we intended to design the convolution engine to be as a part of larger system. Therefore we do not use any of FPGA embedded hardware multipliers and they are left to be used by other part system and we designed our custom hardware multiplier. Based on that, and in order to minimize the area cost and the total power consumption and also to simplify the implementations, Look up tables (LUTs) have been used in the multiplication operations. A constant coefficient multiplier is built in by using two ROMs memories to parallelize the read operation and to reduce the size of the LUT, and consequently reducing the size of the silicon area. The multiplication with constant coefficients operation that used in FIR filters leads us to use this type of multiplier. In principle, the evaluation of any finite function can be carried out using a look-up table (LUT) memory that is addressed with the argument for the evaluation and whose output is the result of the evaluation. Unfortunately, the use of a single LUT for the multiplication is unlikely to be practical for any but the smallest argument, because the table size grows rapidly with the width of the argument. Therefore the solution is to split the argument, use LUTs, and then use a tree of adders [10]. An example of this is given in Figure 7.

The nine 16-bit independent multiplications obtained in this way, are then added by the Adder Tree, which generates a 16-bit

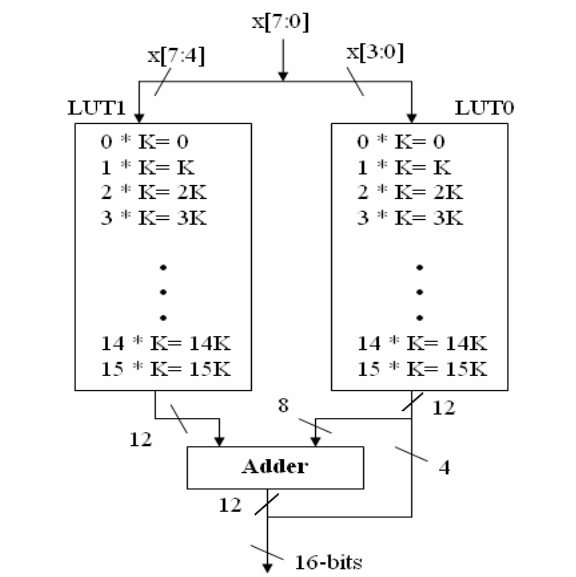

Figure 7: Look-up table based multiplication for 8-bit wide coefficient

result. The purpose of using adder tree is to reduce the computing time through parallel processing and to start the subsequent operations before finishing the ongoing ones.

C. *Systolic Array Architecture using two moving windows*

To improve the performance, a modified architecture is proposed. This architecture is based on using two moving windows in parallel to process the image. Two pixels should be processed simultaneously. This is achieved by sharing two adjacent pixels the same memory address. Thus, 16-bit memory width is used for this purpose. The idea behind this type of processing is extracted from the fact that there is a common pixels between two adjacent moving windows as shown in Figure 8.

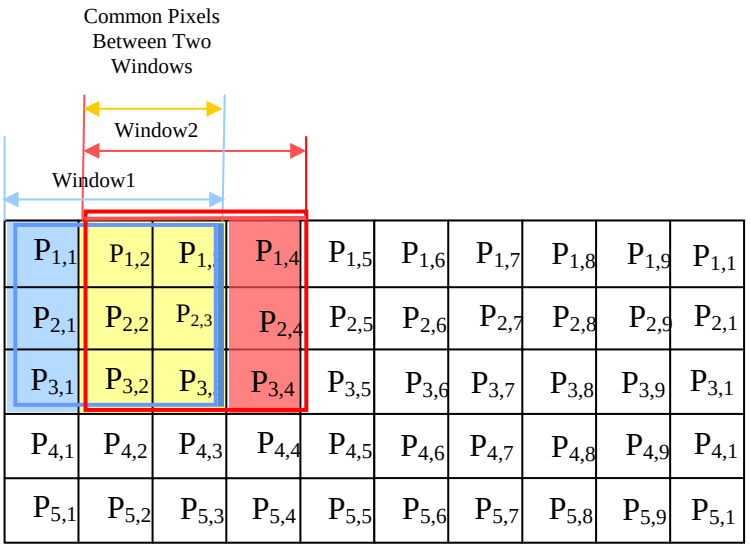

Figure 8. Common Pixels between two windows for an image size of 5x10

The first pixel should be stored in the least significant byte(LSB) of the first memory location, and the 2nd pixel in the MSB and so on for all the rest of pixels. From Figure 7, one can see that there are six common pixels between two adjacent windows. Therefore, the data of two 3x3 windows can be available by just reading one window of 3x4.

The architecture of this technique is shown in Figure 9.

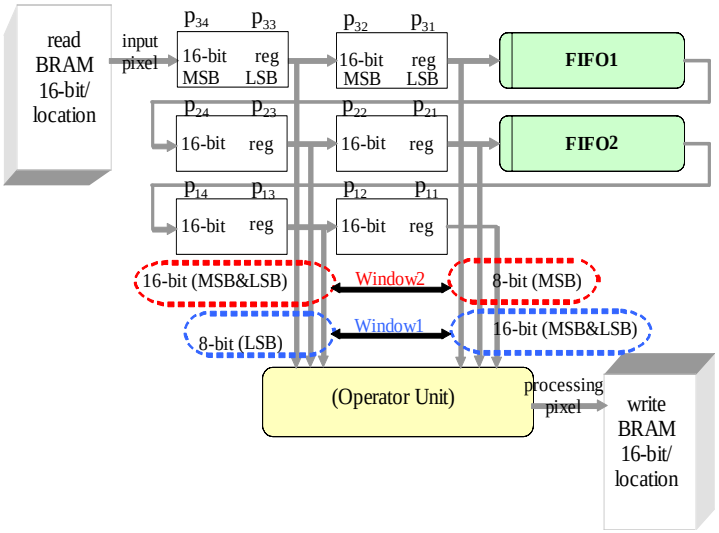

Figure 9. The architecture of 2D convolution using two 3x3 moving window operators

From Figure 8, one can see that the register width is extended to 16-bits. That is to enable two pixels to be shifted with each clock cycle.

The advantage of using this technique is that the operation required to process one image can be duplicated at the same time duration.

IV. RESULTS AND DISCUSSION

The design architecture is modeled by using VHDL language and implement on Spartan3-E FPGA chip through ISE 9.2i program as a target technique.

The hardware architecture presented in the previous section is dedicated to window-based operation. An architecture adapted to 3x3, 5x5, and 7x7 moving window is designed. The main objective during architecture conception is to achieve real time operation; therefore an analysis in performance parameters is required. The time required to process an input image with a window-based operator is composed of two main times the *latency time* and the *parallel processing time* [5].

$$T = Latency_{Time} + Parallel\ Processing_{Time} \quad (9)$$

where:

$$Latency_{Time} = [\ L(w-1) + w\ ] \frac{1}{f}$$

and

$$Parallel\ Processing_{Time} = Reading\ Window_{Time} + Execution_{Time}$$

where

$$Reading\ Window_{Time} = [\ M - (w-2)\ ] x\ N\ (\frac{1}{f})$$

L: FIFO buffer length
M: no. of image rows
N: no. oif image columns
w: window length
f: FPGA operating frequency

From the above rule, it can calculate the time required to process an image of 512x512x8-bits/pixel by using 3x3 moving window in Spartan 3E platform of 50 MHz operating frequency. The execution time required to process one window is calculated by multiplying the number of required clock cycles by operating frequency which yields 0.16 µs. The total time required to process one image using the first proposed architecture is 5.25334 ms. This means that the throughput is equal to 190.335 image per second. If the second proposed architecture is used, the reading window time will reduce to the half. Then, the total time required to process one image using the first proposed architecture is 2.626 ms. Therefore, the throughput will duplicate to 380.74 frame per second.

Table 1 with the throughput values mentioned above is used to calculate the number of operations per second for each of the proposed architecture. In the first architecture, the total number of operations is calculated to be equal to 5242880 (5.242 MOPF) which in term equal to 997.94 MOPS. While the number of operations per second in the second architecture equals to 1.995 GOPS.

Table 1: no. of operation for processing an image by using 2D convolution

Elementral Operations	Number of Executions
Multiplication	w^2 x M x N
Addition	(w^2 -1) x M x N
Division	M x N
Load	M x N
Store	M x N

Area and speed are the two main measurements in evaluating the hardware performance of this system. Area is measured by number of slices occupied. Speed can be measured by the maximum allowable clock frequency. These two factors are compared between the system proposed in this paper and some previous related papers.

In table 2, one can see that both the silicon cost and the speed will increase when increasing the window size. One can see that with the increasing of window size, the maximum allowable frequency can increase. That is because the

parallelism (operations that implement simultaneously) increases.

Table 2: The effect of increasing window size in area consumption and maximum allowable frequency when single moving window is used

window size	3x3	5x5	7x7
Slices	254	517	958
Max frequency(MHz)	130.42	132.11	140.5

The advantage of using the second moving window architecture in term of speed can be seen in table 3. The silicon cost is still low enough.

Table 3: The effect of used architecture in area consumption, maximum allowable frequency, and MOPS

Architecture	Single moving windows	Two moving windows
Slices	254	338
Max frequency(MHz)	130.42	128.75
MOPS	997.94	1995

An image of 256x256x8-bit/pixel is used to compare the results achieved from the first proposed architecture with the system proposed in [6] as shown in table 4.

Table 4: Cost and performance comparison between the first proposed architecture and the system presented in [6]

Architecture	slices	Time(ms)	Max freq(MHz)	Device
First proposed architecture for 3x3 Convolution LUT based Multiplication	328	1.316	166	Spartan 3E
3x3 Convolution LUT based Multiplication [6]	479	1.31	50.99	Vertix-E

The cost and the performance of the second proposed architecture are compared with the system presented in [5] when a picture of 200x640 is used. Table 5 shows this comparison.

Table 5: Cost and performance comparison between the second proposed architecture and the system presented in [5]

Architecture	slices	Mega pictures per second(MPPS)	Device
Second proposed architecture for 3x3 Convolution	541	99.614	Spartan 3E
7x7 Convolution [5]	2855	61.44	Vertix-E

From the table shown above, one can see that the throughput and cost of the second proposed architecture are better than that presented in [5] although using 7x7 moving window.

The first row of Figure 10 shows the input image in which some operators have been applied. The second row of Figure 10 shows the output image of post synthesis for the Sobel Edge detector operator, while the last row of Figure 10 shows the Sobel Edge detector when matlab environment is used. One can that the differences of the proposed hardware system with the software version of the systems (where floating point calculations are used) are negligible. This is due to the calculations that achieved in high resolution.

Although the area consumed when using the embedded multipliers are less than that used when LUT multipliers are used, the LUT based multiplier convolution can operate in a highest allowable operation frequency (166MHz vs. 128MHz). The reason for this is that the embedded multipliers place far from the slices where the LUTs exists.

V. CONCLUSION

In this work, two designs are proposed for the systolic array architecture. Then a comparison among them with respect to building area and the number of frames processed per second is performed. Each of the design architecture is modeled by using VHDL language and implement on Spartan3-E FPGA chip through ISE 9.2i program as a target technique. It is found that the throughput of the second proposed architectures become twice (380 frame/sec) the throughput of the first proposed one (190 frame/sec). The images resulted from the second proposed architecture are similar to that of the first architecture. Consequently, it is found that the image(512x512) processing time in first proposed architecture equals 5.25334m sec, while the time for second proposed architectures is equal to half of that for first one. Building on this; one can conclude that the second proposed architecture can be used in very high speed and video applications, in real time. Using LUTs based multiplication leads to further improvement on the overall throughput, and the system speed consequently increases.

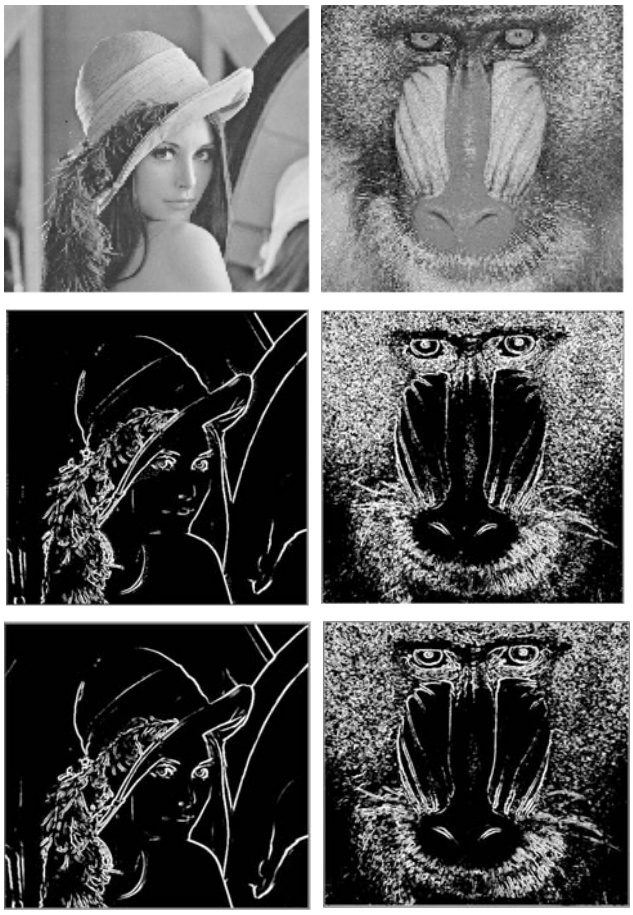

Figure.10:Sobel Edge detection for Lena and Baboon images

REFERENCES

[1] S. Perri, M. Lanuzza, P. Corsonello and G. Cocorullo,"SIMD 2-D Convolution for Fast FPGA-Based Image and Video Processing", IEEE VLSI System Transaction, Vol.7, 2003, pp1-3.

[2] V. Daggu, "Design and Implementation of an Efficient Reconfigurable Architecture for Image Processing Algorithms Using Handel-C ", M.Sc Thesis, Electrical and Computer Engineering, University of Nevada, Las Vegas, U.S.A, 2003.

[3] A. Nelosn, " Implementation of Image Processing Algorithms on FPGA Hardware ", M.Sc Thesis, Electrical Engineering, Graduate School of Vanderbilt University, May 2000.

[4] S. Wong Dsto, M. Jasiunas and D. Kearney, "Fast 2D Convolution Using Recofigurable Computing", IEEE Proceedings of the Eighth International Symposium on Signal Processing and Its Applications, 2005.

[5] G. Saldana and M. Arias-Estrad, " FPGA-Based Customizable Systolic Architecture for Image Processing Applications ", IEEE International Conference on Reconfigurable Computing and FPGAs(ReConFig 2005) ISBN 0-7695-2456-7, 2005.

[6] D. Venkateshwar, Sh. Patil, N. Anne and V. Muthukumar, "Implementation and Evaluation of Image Processing Algorithems on Reconfigurable Architecture Using C-based HDL, International Journal of Theoretical and Applied Computer Science Vol.1, No. 1, pp9-34, 2006.

[7] M. Zhang and V. Asari, "A Fully Pipelined Multiplierless Architecture for 2D Convolution with Quadrant Symmetric Kernels", IEEE Asia Pacific Conference on Circuits and Systems, 2006.

[8] C. Johnston, K. Gribbon and D. Baily, " Implementation Image Purocessing Algorithms on FPGAs", Institute of Information Sciences and Technology, 2004.

[9] Peter Pirsch, " Architectures Digital Signal Processing", By John Wiley & Sons Ltd, ISBN 0-471-97145-6, 1998.

[10] E. Jamro and K. Wiater, "FPGA Implementation of Addition as a Part of The Convolution", IEEE Euromicro Sysmposium on Digital System Design ISBN 0-7695-1239-9, 2001.