Architecture for filtering images using Xilinx System Generator

Alba M. Sánchez G., Ricardo Alvarez G., Sully Sánchez G.; FCC and FCE BUAP

***Abstract*—**This paper presents an architecture for filters pixel by pixel and regions filters for image processing using Xilinx System Generator (XSG). This architecture offer an alternative through a graphical user interface that conbines MATLAB, Simulink and XSG and explore important aspects concerned to hardware implementation.

***Keywords*—**Digital image processing, Matlab, Xilinx System Generator.

I. Introduction

THE handling of digital images has become in recent decades a subject of widespread interest in different areas such as medical and technological applications, among others. We may cite lots of examples where image processing helps to analyze, infer and make decisions. The main objective of image processing is to improve the quality of the images for human interpretation, or the perception of the machines independently. This paper focuses in the processing pixel to pixel of an image and in the modification of pixel neighborhoods and of course the transformation can be applied to the whole image or only a partial region. The need to process the image in real time, leading to the implementation level hardware, which offers parallelism, and thus significantly reduces the processing time, wich was why decided to use Xilinx System Generator, a tool with a graphical interface under the Matlab Simulink, based blocks wich makes it very easy to handle with respect to other software for hardware description. In addition to offering all the tools for an easy graphical simulation level.

The filtering of images is a technique which gives an enhancement of the image characteristics. The purpose of filtering techniques is to process an image in a way that is more appropriate than the original specific application.

Among the applications of filter image are: the elimination of noise, enhancing edges and contours, and so on. This article presents an architecture filtering images System Generator, which is an extension of Simulink and consists of a bookstore called "blocks Xilinx", which are mapped architectures, entities, signs, ports and attributes, which Script file to produce synthesis in FPGAs, HDL simulation and developmentst tools. The tool retains the hierarchy of Simulink when it is converted into VHDL.

II. Xilinx system generator

Xilinx System Generator (XSG) is an integrated design environment (IDE) for FPGAs, which uses Simulink, as a development environment and is presented in the form of blockset. It has an integrated design flow, to move directly to the configuration file (*. bit) necessary for programming the FPGA.

One of the most important features of Xilinx System Generator is possessed abstraction arithmetic, that is working with representation in fixed point with a precision arbitrary, including quantization and overflow. You can also perform simulations both as a fixed point double precision. XSG automatically generates VHDL code and a draft of the ISE model being developed. Make hierarchical VHDL synthesis, expansion and mapping hardware, in addition to generating a user constraint file (UCF), simulation and testbech and test vectors among other things.

Xilinx System Generator was created primarily to deal with complex Digital Signal Processing (DSP) applications , but it has other applications like the theme of this work. The blocks in Xilinx System Generator operate with Boolean values or arbitrary values in fixed point, for a better approach to hardware implementation. In constrast Simulink works with numbers of double-precision floating point. The connection between blocks Xilinx System Generator and Simulink blocks are the gateway blocks.

In Fig. 1 shows the broad flow design Xilinx System Generator. As already mentioned, you can then move to the configuration file to program the FPGA, and the synthesis and implementation steps are optional for the user but not for

Manuscript received March 10 , 2007; Revised May 31, 2007

This work was supported in part by the Puebla Benemerit Autonomus University (BUAP).

Alba M. Sánchez G. is with the Computer Science Faculty at the Puebla Benemerit Autonomus University, Av. San Claudio y 14 Sur Cd. Universitaria Z.P. 72570 (01 222 229 55 00 ext. 7218 e-mail: agalvez@ cs.buap.mx).

Ricardo Alvarez G. is with the Electronic Science Faculty at the Puebla Benemerit Autonomus University, Av. San Claudio y 18 Sur Cd. Universitaria Z.P. 72570 (01 222 229 55 00 ext. 7408 e-mail: algor@lece. buap.mx).

Sully Sánchez G. is with the Computer Science Faculty at the Puebla Benemerit Autonomus University, Av. San Claudio y 14 Sur Cd. Universitaria Z.P. 72570 (01 222 229 55 00 ext. 7217 e-mail: ssanchez@ solarium.cs.buap.mx).

System Generator [5].

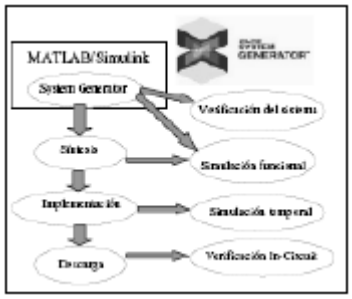


Fig. 1. Design flow in System Generator.

III. Design

For the design of filters should be meet hardware requirements, it is therefore necessary for image preprocessing prior to the same architecture.

Unlike the level software processing, where the image is a two-dimensional arrangement n x m, and processed as such, at hardware this matrix must be an array of one dimension, namely a vector. Then save the image information in a ROM memory. [6].

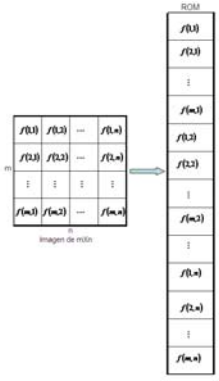


Fig. 2. Store the image values in a ROM memory

The coordinate (i, j) suffers the following transformation

$$(i, j) \rightarrow (j-1)m + i$$

Coordinate latter is the position that it occupies in the ROM. We have sorted by columns, is the first to say this in memory the first column and then the second and so on until end with the values, you can choose to be kept in rows. Once processed image, it applies a reverse transformation that reversed the process of a settlement to a one-dimensional two-dimensional.

To access the information, you must address the ROM that contains the value or values of the pixels that are required for processing, depending on the filter to be used and thus obtain the new values of the image. Therefore requires two more blocks, the generator and address block processing operations.

As shown in Fig. 3, the processing stage consists of three blocks: the address generator, which feeds the block ROM, it is the entire image information, this in turn is used by the block Operational information processing.

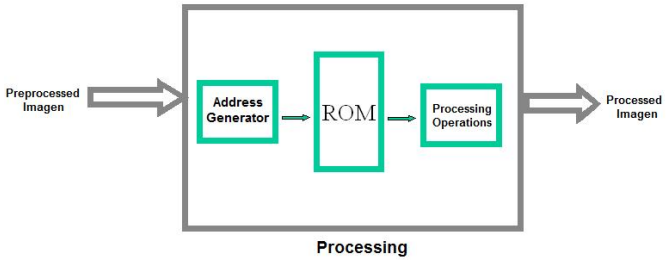


Fig. 3. Block diagram of image processing.

IV. Image Processing

We are presenting the main filter processing technique of the picture: the processing pixel by pixel and the pixel neighborhood [1]. Both are working with the intensity of the image.

a) Processing pixel by pixel

This technique is the simplest. It works with the intensity of each pixel [2], as shown in Figure 4.

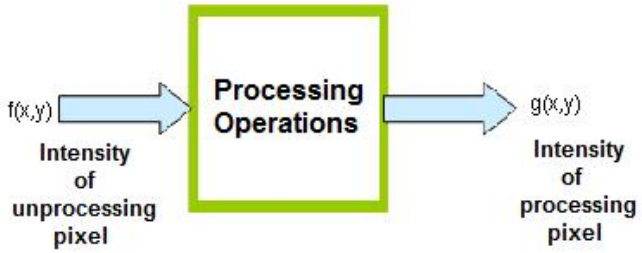


Fig. 4. f(x,y) is the intensity of unprocessing píxel, in the output we have other intensity, the coordinates are unchanged, only the intensity was modificated.

b) Processing of pixel neighborhoods

Basically consist in transform the value of a pixel in the position (i, j) taking into account the values of neighbors pixels. For example, if we consider a pixel neighborhood and multiply with a different weight, by consider the values of the neighbors, the result of this amount is the value of the new pixel of the image output in the same position (i, j). All that remains is to define the values of the weights, which is usually defining a mask with constant values. The mask is actually a filter, so that depending on its nature, and will be the end result. For example, if we define the mask next

$$\begin{bmatrix} f(i-1,j-1) & f(i-1,j) & f(i-1,j+1) \\ f(i,j-1) & f(i,j) & f(i,j+1) \\ f(i+1,j-1) & f(i+1,j) & f(i+1,j+1) \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \quad (2)$$

In (1) we have the strengths of the region's image dimension equal to the mask, in this case is 3 × 3, in subsection (2) is the

mask, i.e. the weights it will multiply each of the intensities of the pixels. So the intensity of f (i, j) is replaced by:

$$g(i,j)=h_{11}f(i-1,j-1)+h_{21}f(i,j-1)+h_{31}f(i+1,j-1)+h_{12}f(i-1,j)+$$
$$h_{22}f(i,j)+h_{32}f(i+1,j)+h_{13}f(i-1,j+1)+h_{23}f(i,j+1)+h_{33}f(i+1,j+1)$$

V. RESULT S

Among the image operations we have the shine, contrast threshold applied to the image in Fig. 5.

Fig. 5 Original image

The images with low contrast can be due to various causes, such as poor lighting, lack of dynamic range in the sensor or even incorrect selection of the opening of the lens during the capture of the image. Fig. 6 shows a typical conversion used to enhance the contrast and in Fig. 7 architecture.

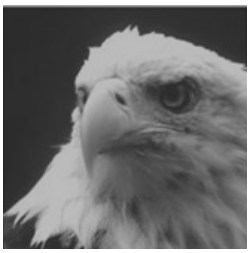


Fig. 6 Contrast

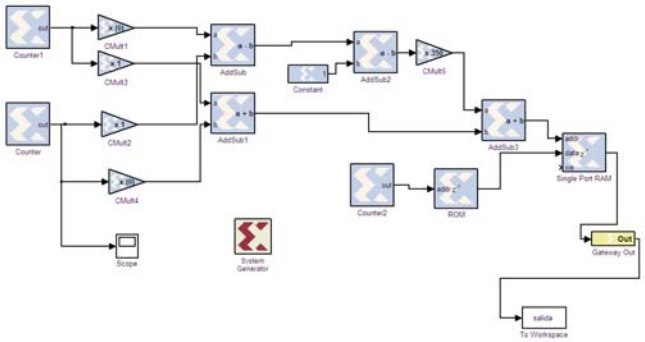


Fig. 7 Architecture for contrast

The operator threshold transforms an image of binary output from an image of gray scale, where the level of transition is given by the input parameter p1. Fig. 8 shows the implementation of this transformation and Fig. 9 architecture.

Fig. 8 Treshold

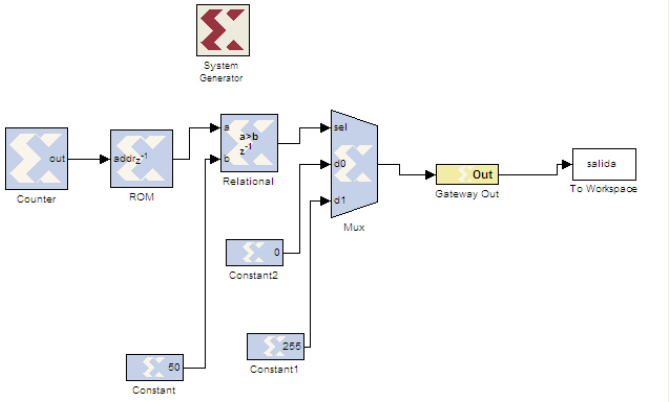


Fig. 9 Treshold architecture

The negatives digitized images are useful in many applications, such as medical imaging and representation in photographs of a monochrome screen with films with the idea of using the resulting negative slides as normal. Fig. 10 illustrates the use of this simple transformation and architecture in Fig. 11.

Fig. 10 Negative

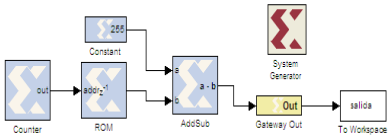


Fig. 11 Negative architecture

Fig. 13 shows the negative image of the figure 12 and the fig. 14 architecture.

Fig. 12 Original Image

Fig. 13 Color negative

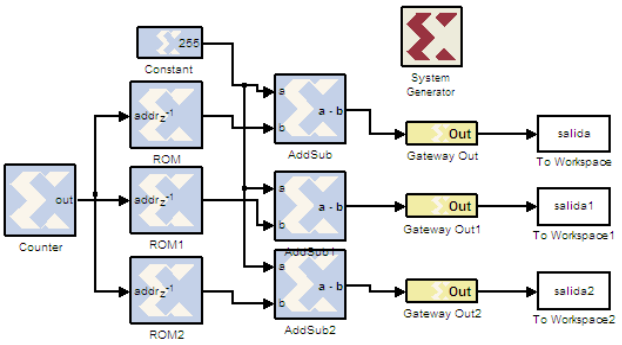


Fig. 14 Color negative archityecture

Figure 15 and figure 16 shows the brightness and architecture respectively applied to the image of the Figure 5.

Fig. 15 Brightness

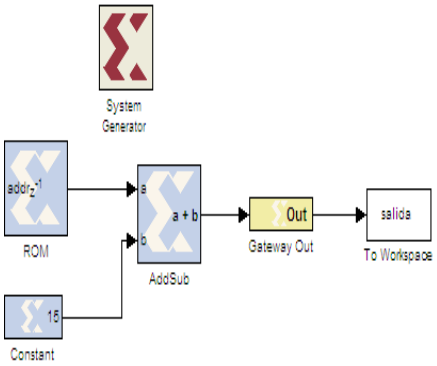


Fig. 16 Brightness Architecture

In the operations of pixel neighborhood noted for its functionality operations like softened edges and extraction. The former is aimed at improving the quality of the image, while the latter allow for the underlying edges.

The filtering operations based its operations in the convolution of the image using the so-called core convolution. There are two types of filters: high-pass and low-pass, which in the context of the theory of signals supposed to miss the first high frequency signal and the latter casualties. In the case of the pictures we refer to specific frequencies. This high frequency is associated with sudden changes in intensity intervals small space, i.e. edges, while low frequencies refer to slow changes in the intensity.

In Fig. 17 shows the effect of a low-pass filter and the result of a high pass filter in Fig. 18.

Fig. 17 Low pass

Fig. 18 High pass

The softening is precisely to mitigate or eliminate the high-frequency components (edges and noise). In the case of the edge effect is a mitigating them and in the case of noise, the desired effect is the elimination of it. The softening can be done by statistical techniques: Arithmetic mean, weighted average, half Gaussian.

The simplest model is the arithmetic mean, it considers an average of the pixels in an environment focused on a pixel (i, j). Fig. 20 shows the result when applying to the image of the fig. 19.

Fig. 19 Original image

Fig. 20 Arithmetic mean

It is possible model for other processes such as the average weighted average, it considers changing the weight that the central pixel located in (i, j) to an arbitrary value positive, then what we should do is to preserve the status output the convolution matrix, we need to change the weight of pixels, ie the sum of the coefficients of the matrix is unity, so be sure not leave the range. This transformation is shown in fig. 21.

Fig. 21 Weight mean for p=12

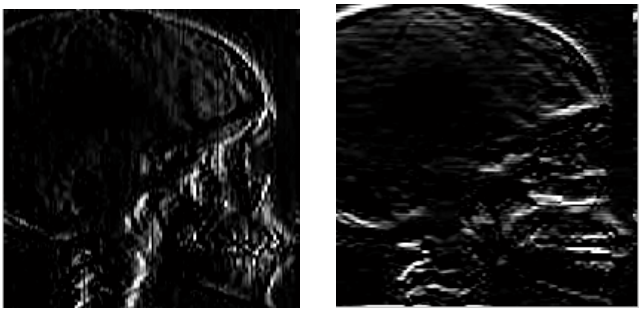
The average Gaussian filter is based and consider a Gaussian curve of revolution around the center of a pixel matrix convolution. The fig. 22 shows the effect.

Fig. 22 Gaussian mean

The first derivative is zero in all regions of constant intensity and has a constant value throughout the transition of intensity. The second derivative, in contrast, is zero at all points except at the beginning and end of a transition intensity. The first derivative is used to detect the presence of an edge, as well as the sign of the second derivative. They are different operators for extracting edges based on the first and second derivative.

The first derivative operators are based on the concept of vector gradient and led to the filters: gradient image, Sobel Prewit and Roberts.

The gradient of an image f (x, y) at a point (x, y) is defined as a two-dimensional array, which consists of the derivative in the x direction and the direction and Daughter, which is denoted respectively Gx, Gy . The values and Gx Gy, can be implemented by convolution of the image masks figure known as operators Sobel. The results are shown in Fig. 23, once it is applied Sobel operator to the image of the fig. 18.



a) b)

Fig. 23 Image process in a) Gx , b) G_y

Fig. 24 $|G| = |G_x| + |G_y|$

The operator of Prewitt is similar to Sobel, differing in the coefficients of masks. In Fig. 26 shows the results after applying this filter to the image of the figure 25.

Fig. 25 Original image

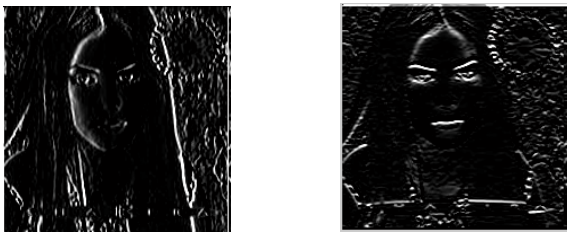


Fig. 26 Image process in G_x , and G_y

Fig. 27 $|G| = |G_x| + |G_y|$

The architecture for processing Prewitt is shown in fig. 28, in Fig. 29, the sum of the absolute values.

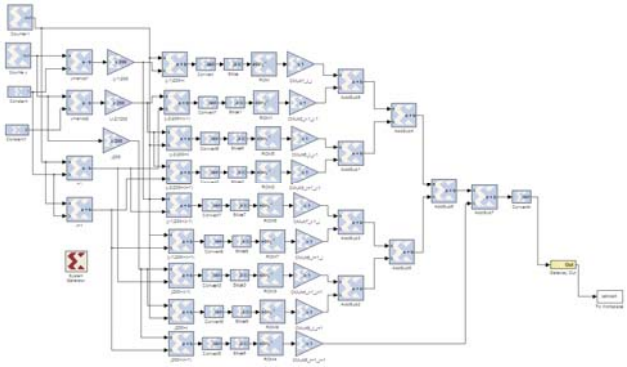


Fig. 28 Prewitt Architecture

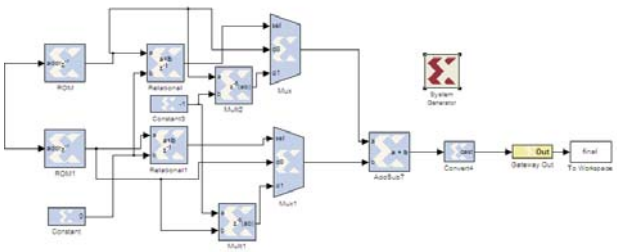
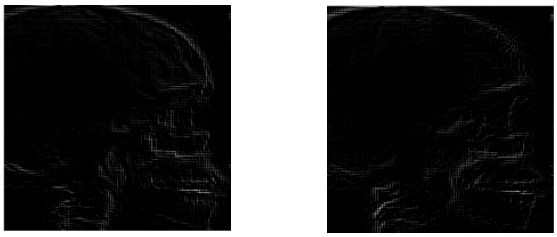


Fig. 29 Add modules

The operator Roberts, unlike the previous two, it marks only the points edge, without reporting them guidance. It is a very simple operator who works very well in binary images. Fig. 30 shows the transformation of the image to apply the filter.



a) b)

Figure 30 Image process in a)D_1 and , b)D_2

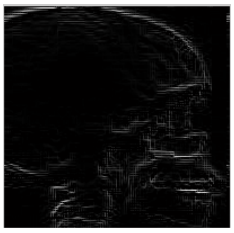


Figure 31 $D = |D_1| + |D_2|$

Within the second derivative traders, highlights the Laplacian operator. In fig. 32 shows the result of applying the Laplace operator.

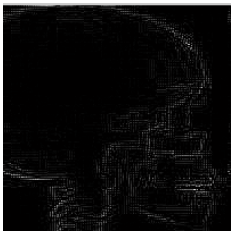


Fig. 32 Laplacian

In mapping, graphic arts and scientific applications are required to display information on the changes in the image, this relates obviously directional derivative.

Using another type of derivative can be achieved different lighting effects. The fig. 33 was derived using the type Robinson with the eight directions given by the wind rose.

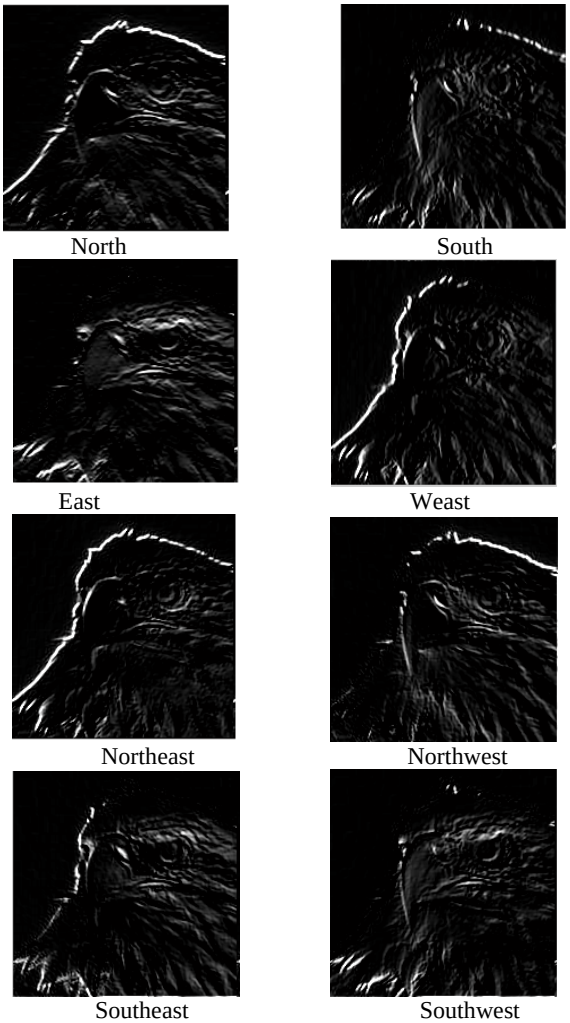


Fig. 33 Lighting in the eight directions

VI. CONCLUSIONS

The Xilinx System Generator tool is a new application in image processing, and offers a friendly environment design for the processing, because the filters are designed by blocks. This tool support software simulation, but the most important is that can synthesize in FPGAs hardware, with the parallelism, robust and speed, this features are essentials in image processing.

REFERENCES

[1] González R. C., Woods R. E.: Tratamiento digital de imágenes. Copublicación de Addison-Wesley Iberoamericana, S. A. y Ediciones Díaz de Santos, S. A. (1996) 180–243

[2] Pajares G., de la Cruz J. M., Molina J. M., Cuadrado J. López A.: Imágenes Digitales Procesamiento práctico con Java. Alfaomega, Ra-Ma (2004) 13-58

[3] Matlab website,http:// www.mathworks.com.

[4] Pajares G., de la Cruz J. M.: Visión por Computador Imágenes digitales y aplicaciones, Ra-Ma (2001) 89-177.

[5] Xilinx System Generator User's Guide , http:// www. Xilinx.com.

[6] Tejeda J.C., Arias M.: Arquitectura FPGA para filtrado de imágenes en Tiempo Real, Tesis de Maestría, INAOE,(2001).

[7] Notas de procesamiento de imágenes http://www.cs.buap.mx/~mmartin

[8] R. J. Vidmar. (1992, August). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. *21(3).* pp. 876—880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar