An FPGA-Based Region-Growing Architecture for Binary Images

Emerson Carlos Pedrino[1], José Hiroki Saito[1,2], Hermes Senger[1]
[1]Department of Computer Science
Federal University of São Carlos, UFSCar
Sao Carlos, SP, Brazil
[2]Faculdade Campo Limpo Paulista, FACCAMP
Campo Limpo Paulista, SP, Brazil
{emerson, saito, hermes}@dc.ufscar.br

Valentin Obac Roda, Marcelo Marinho
Department of Electrical Engineering
University of São Paulo, USP
São Carlos, SP, Brazil
{valentin, mar.marinho}@sel.eesc.usp.br

Abstract — In this work we present the development of an algorithm for region growing in binary images using CPLDs and FPGAs devices. Generally, gray-level images with spherical objects present undesired effects when they are captured by a vision system, because such images will present holes when they are converted to the binary format. Thus, the proposed algorithm in this work solves this kind of drawback. A simulation example of the algorithm operation is performed where we can see the performance of the system for real time operations and the obtained results are compared with the results from a software implementation. The system is capable of processing binary images with 320x240 pixels at 60 frames/s.

Keywords- PLDs; region growing .

I. INTRODUCTION

Region growing methods are used frequently in image processing tasks to fill small holes in images. Thus, it's possible to perform measures for characterization of particles in many situations. These small holes are found in spherical particles images with vertical illumination that is close to the object [1]. This kind of procedure is normally required in microscopic images analysis of blood cells [2], in images obtained from electronic scan microscope [3], to name just a few. Thus, in this paper it is presented an algorithm and the development of a system in a FPGA for region growing in binary images that is based on the connectivity of the pixel under analysis with its neighbors. The system is capable of processing images in real time using only low-density CPLDs (Complex Programmable Logic Devices) and FPGAs (Field Programmable Logic Devices). A simulation process of the hardware algorithm operation is performed to compare its results with the results obtained by the same implementation in software. The system can be used for real time operations.

This article is organized as a brief introduction describing the method motivation, section 1; a description of the algorithm, section 2; a description of the proposed hardware, section 3; the results are presented in section 4; section 5 presents the conclusions.

II. PROPOSED ALGORITHM

Region growing algorithms are based normally on logical and morphological operations starting from a point that is placed inside a desired region of the image [4]. Some methods are based on the connectivity of the background pixels with the external edge of the image. Many methods presented in the literature have the disadvantage that it's necessary to know the position of a certain pixel that is located inside the object under consideration. This disadvantage is a drawback for practical systems that need to perform their processing in real time.

In this article, we propose an algorithm that is based on the connectivity of background pixels with the external edge of an image. The algorithm works as follows: 1-) it's used a binary image and an auxiliary image that contains an external edge of pixels with values 0 (background) and inner pixels of values 1 (image). In Fig. 1 it's shown an example of a binary image to be processed and the respective auxiliary image which will be performed the region growing operation. 2-) To verify if a pixel should be filled, each pixel from binary image and its equivalent pixel on auxiliary image are identified. The four neighbors within a 3x3 window on auxiliary image are verified too. Thus, the following criterion is applied:

if $I_{aux}(x,y)=1$ and $I_{bin}(x,y)=0$, it's verified the 4-neighbor of the current pixel,
else, if $I_{aux}(x,y)=I_{aux}(x,y)$;

The pixels $I_{aux}(x,y)$ and $I_{bin}(x,y)$ are the points with coordinates (x,y) of auxiliary image and binary image, respectively.

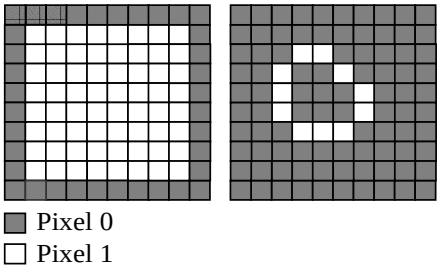

Figure 1. The left image is the auxiliary image and the right one is the binary image to be processed.

If the first condition of the algorithm is satisfied, the 4-connected pixels of the $I_{aux}(x,y)$, $I_{aux}(x,y-1)$, $I_{aux}(x+1,y)$, $I_{aux}(x,y+1)$ and $I_{aux}(x-1,y)$, are verified. If at least one is 0, it's assigned a value 0 to $I_{aux}(x,y)$, otherwise the $I_{aux}(x,y)$ maintains its current value. Thus, this process is repeated until all inner pixels are filled. In Fig. 2 it's shown an example of the algorithm's operation. This example has been implemented using the LabView software.

Figure 2. Example of the algorithm's operation for a binary image. The original image (to the left) is filled by the algorithm and the result can be seen in the resulting image to the right (auxiliary image).

III. PROPOSED HARDWARE

In this section we present the development of a reconfigurable system to region growing of binary images. The proposed system has been implemented using the Verilog HDL language and it has been simulated in the Altera Quartus II software. In this project it was used the EPF10K20RC240-4 FPGA device from FLEX 10K family to implement the proposed algorithm in section 2. In Fig. 3 it's shown the block diagram of the proposed system.

The circuit uses a memory to store the original binary image, a memory to store the auxiliary image and four memories to store the 4-neighbors of the current pixel under analysis. In this work we used the TC551001BPL-70 memory from Toshiba [5] to the final implementation of the proposed hardware. At the beginning of the processing, the auxiliary memory and the memories containing the 4-neighbors pixels are initialized with the same values. The system works as follows: during the positive edge of the vertical synchronism, supplied by a RGB system, the addressing variables of original and auxiliary memories are initialized. The addresses from memories are obtained by means of expression 1. During the positive edge of the system clock, the memory addresses are incremented.

$$Add = col \times i + j + 1, \quad (1)$$

In expression 1, *col* is the number of pixels by line, *i* is the number of image lines and *j* refers to the columns number. On the positive edge of clock, the current pixels of binary image and auxiliary image are read from the respective memories. The memories containing the 4-neighbors pixels of the current pixel are controlled by means of a CPLD device from MAX7000S family from Altera. After the data processing, according the algorithm presented in section 1, the processed pixels are shown at the system output and the auxiliary memories and the memories that store the 4-neighbors pixels are updated by CPLD. The output data are sent to a DA converter that assembly the composite video signal, that can be seen by any display system, using a dedicated control logic. During the positive edge of the vertical synchronism the process is repeated until the algorithm converges to an optimal condition.

IV. RESULTS

In Fig. 4 it's shown a result from a simulation of the proposed system to a hypothetical binary image with 25 pixels of 8-bit resolution, which contains a central hole (value 0) forming a diamond figure. On the 14^{o} clock pulse, the data bus *data_i* (data bus from original memory) is equal to 0 and the data bus *data_a* (data bus from auxiliary memory) is equal to 255, thus, the current pixel from auxiliary memory has 4-neighbors with values 255 (white). We can see that this pixel is not connected with the image edge, thus, it is filled with the value 255 shown by signal *out* in Fig. 4.

In Table 1 it's shown the resources used by the FPGA device to implement the proposed system. According to Table, it was used few logic elements and there is space to work with a larger neighborhood to the current pixel under analysis. The results obtained from simulation are in according with the result obtained by LabView software. In this work, it was used a 13.5MHz clock that was supplied by a CMOS digital video sensor. The resolution of the images that can be processed by the system is 320x240 pixels and the frame rate is 60 frames/s.

V. CONCLUSIONS

In this paper it has been presented the development of an algorithm for region growing in binary images using CPLDs and FPGAs devices. The hardware new algorithm has been implemented in Verilog HDL. Generally, gray-level images with spherical objects present undesired effects when they are captured by a vision system Thus, the proposed algorithm solves this kind of drawback and can be used for real time applications. A simulation example of the algorithm operation is performed where we can see the performance of the system and it has been verified that the results are equivalent to a software implementation. The system is capable of processing binary images with 320x240 pixels at 60 frames/s.

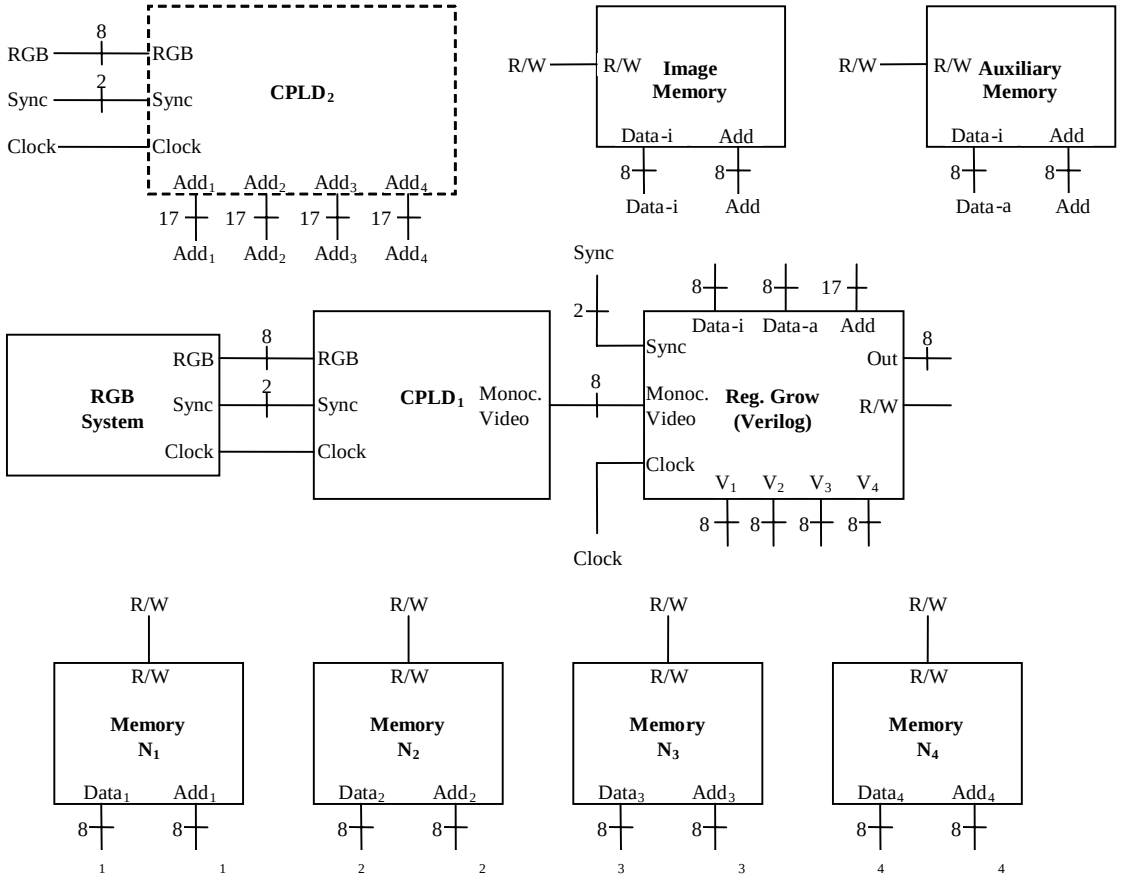

Figure 3. Block diagram of the proposed system.

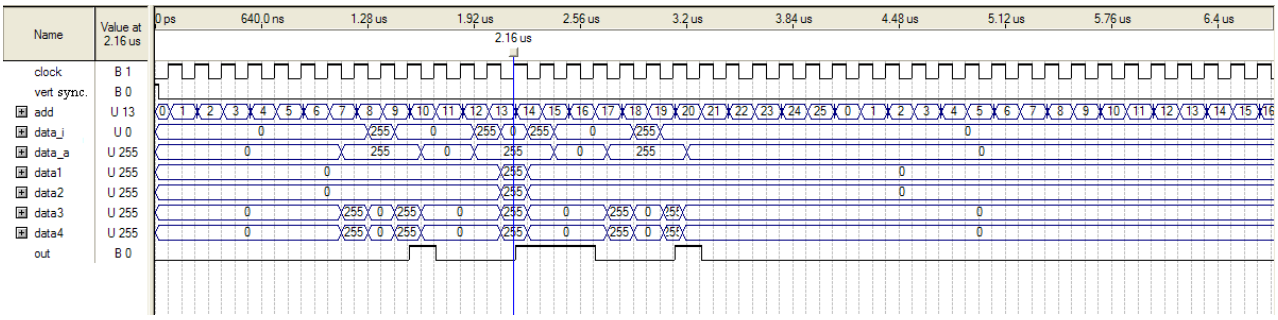

Figure 4. Obtained result of the developed system to a hypothetical binary image with 25 pixels of 8-bit resolution, with contains a central hole (value 0) forming a diamond figure.

TABLE I. FPGA SUMMARY.

Device	Log. Cells	Registers	Lut	Pins	Lut
EPF10K20RC240-4	91 (8%)	64	27	56 (30%)	27

ACKNOWLEDGMENT

Hermes Senger thanks FAPESP (process number 2009/09847-0) and CNPQ (process number 201812/2009-6) for their support.

REFERENCES

[1] J. C. Russ, “The image processing handbook,” CRC Pres, Boca Raton, 1995.

[2] C. Ruberto, A. Dempster, S. Khan, and B. Jarra, “Analysis of infected blood cell images using morphological operators,” Image and Vision Computing, vol. 20, no. 2, pp. 133–146, Feb. 2002.

[3] A. Zaruba, E. Krepper, H. -M. Prasser, and B. N. R. Vanga, “Experimental study on bubbly montion in a rectangular bubble column using high-speed video observations,” Flow Measurement and Instrumentation, vol. 16, no. 5, pp. 277–287, Oct. 2005.

[4] R. C. Gonzalez, R. E. Woods, “Digital Image processing”, PrenticeHall,UpperSaddleRiver,2002.

[5] Silicon gate CMOS 131,072 words x 8 bit static RAM, www.alldatasheet.com, 2006.