Implementation of Medical Image Processing Algorithm on Reconfigurable Hardware

Iuliana Chiuchisan
Electrical Engineering and Computer Sciences
"Stefan cel Mare" University of Suceava
Suceava, Romania
iulia@eed.usv.ro

***Abstract*—In order to implement the upcoming digital image processing algorithms and to process the amount of data captured from sources such as medical instruments, intelligent high speed real-time systems have become imperative. In this paper an efficient FPGA-based design and implementation of image processing algorithm are presented using hardware description language. The FPGA provides the necessary hardware for image processing algorithms with flexibility to support medical image processing by using point operations. The proposed pseudo-color algorithm is designed using Verilog HDL, simulated, tested and synthesized with Xilinx ISE Design Suite 14.5.**

Keywords— digital image processing, pseudo-color algorithm, image enhancement, Verilog, FPGA.

I. INTRODUCTION

Medical imaging occupies a fundamental role in the diagnosis of many diseases, assisted and developed as a multidisciplinary field. Obviously, regarding to the final diagnosis, the decision is taken by the human analyst, but works closely with specialists in other technical fields, because the equipment was developed to be as useful and as efficient as possible, so that diagnosis can be made as quickly and accurately as possible. The digital processing of images has a very important role in the field of medical technology, such as: radiography, ultrasound, computed tomography, computer-aided analysis and interpretation of microscopic cell images.

Digital image processing has developed into one of the most important fields within the area of scientific imaging, due to the rapid and continuing progress in computerized medical image visualization and advances in analysis methods and computer-aided diagnosis [1].

In medical applications, image processing is a vital part of the early detection, diagnosis and treatment of cancer. The challenge is to process and analyze the images in order to effectively extract, quantify, and interpret this information to gain understanding and insight into the structure and function of the organs being imaged [2]. Interpretation of the resulting images requires sophisticated image processing methods that enhance visual interpretation and image analysis methods that provide automated or semi-automated tissue detection, measurement, and characterization [3].

Many image processing applications, especially medical image processing applications for object detection, require that several operations be performed on each pixel in the image resulting in an even larger number of operations per second [4].

Also, in order to process the large amount of data from medical instruments, intelligent high speed real-time systems, which may process data before passing it to the human analyst, have become imperative [5].

The image processing operations can be grouped according to the type of data that they process, as represented in Fig.1 [6].

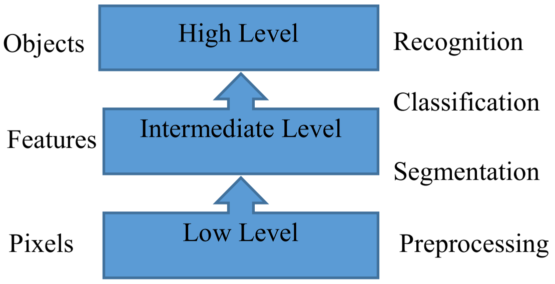


Fig. 1. Image processing operations.

The main goal of any image processing using enhancement operations is to obtain a more suitable result compared to the original, depending on a specific application. Point processing operation is performed in order to enhance an image and details not clearly visible in the original image, but could become visible through the application of a point operator.

This paper presents the image processing techniques, starting with simple generic methods such as low-level preprocessing operators (point operations), followed in future papers by more sophisticated approaches directed at a specific problem. The use of hardware description languages to provide signal processing results is a rather new technique, replacing the classical simulations and offering a direct connection to hardware VLSI implementations.

978-1-4799-2373-1/13/$31.00 ©2013 IEEE

Implementing image processing algorithms for medical applications on FPGA minimizes the time-to-market cost, enables rapid prototyping of complex algorithms and simplifies debugging and verification [4].

In order to describe a system using reprogrammable logic circuits, such as FPGA circuits, we can use packages of computer aided design (CAD). First, the design must be coded in some readable form and this code represents the design using a *hardware description language*. This HDL code contains the essential aspects of the design and allows to be simulated and synthesized onto a FPGA [6].

Verilog hardware description language is preferred by the authors for hardware implementation of a point operation because it allows the description of high complexity systems and because the following advantages:

- The possibility to create functional designs of systems in the form of descriptions of a higher level, without detailing the gates elementary structure, thus reducing significantly the time required for the description of complex systems;
- HDL independence - descriptions are completely independent of a particular circuit, and this allows us to use a description in several projects, such as the implementation of the system in a FPGA circuit, and also in another type of programmable logic circuit;
- Easily modifying the description of a system.

This paper provides an innovative method for image processing techniques simulation, followed by immediate implementation possibility using FPGA.

II. Implementation of Image Processing Algorithm using Verilog HDL

The primary focus of this paper is design representation using hardware description language and the hardware implementation of image processing algorithm performed at the register transfer level (RTL).

Verilog is a hardware description language (HDL) used to model electronic systems. The language supports the design, testing, and implementation of analog, digital, and mixed-signal circuits at various levels of abstraction. The language differs from a conventional programming language in that the execution of statements is not strictly linear. A subset of statements in the language is synthesizable. If the modules in a design contain only synthesizable statements, software can be used to transform or synthesize the design into a netlist that describes the basic components and connections to be implemented in hardware. The netlist may then be transformed into a form describing the standard cells of an integrated circuit or a bitstream for a programmable logic device [5].

Some of the simplest, yet useful image processing operations involve the adjustment of brightness, contrast or false color of an image. A reason for manipulating these attributes is the need to compensate for difficulties in image acquisition, and with image processing we can increase the overall brightness of the object of interest and magnify the tiny residual variations in its contrast. These image processing operations can reveal enough detail to allow proper interpretation [7].

After a digital image has been captured, the first preprocessing steps include two classes of operations, point operations and geometric operations. *Point operations* modify the gray values of individual pixels depending only on the gray value and possibly on the position of the pixels. For example, to invert the image, a gain of -1 is used, as shown in Fig. 3.

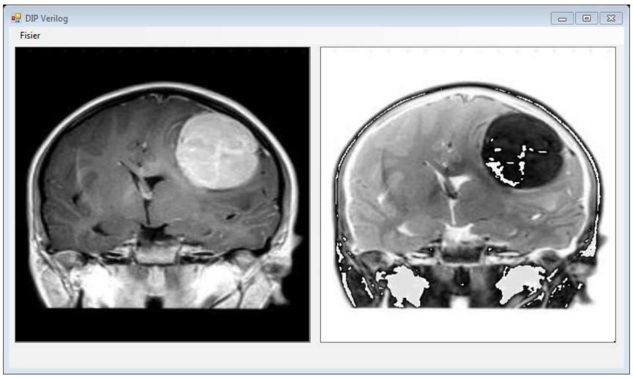


Fig. 3. Inverting a medical image using a Verilog HDL description.

In Reference [7] the author described a class of image processing algorithms designed and tested using Verilog HDL, which include: contrast manipulation, brightness manipulation, inverting image, threshold operation.

A. Pseudo-Color Image Processing Algorithm

The image processing algorithm for enhancing of an image, described and tested in this paper, using hardware description language, Verilog, is a pseudo-color algorithm.

Pseudo-color image processing is one of the most common tasks for the colorization of grayscale images. This image processing technique is used for colorizing satellite images, restoring photographs and medical images in order to enhance their visual appeal or to colorize a specific area of interest.

The pseudo-color described in this paper is used to color, in real-time, some areas of interest in a medical image.

The main purpose of this pseudo-color image processing algorithm is to highlight and make more visible to a human viewer specific areas in the original image. The method is useful because the ability of the human visual system is to better distinguish colors than shades of gray and the coloring occurs in real time.

The technique consists in mapping a grayscale image onto a color image, in which a separate color is assigned to each pixel value.

In the image processing algorithm described below, the values of the gray-level have been represented on the 8-bit field. This new range of colors can be built based on any kind of rules, in order to mark some specific areas. For example, in the presented Verilog code, pixels with gray level between 0 and 29 will be displayed in black, pixels with gray level between 30 and 70 will be displayed in blue, pixels with gray level between 71 and 189 will be displayed in green, pixels with gray level between 190 and 249 will be displayed in red, and remaining pixels are white.

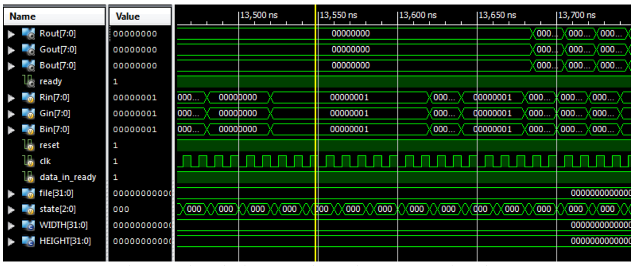
The Verilog code for pseudo-color filter in order to obtain a pseudo-color image is shown in Table I.

TABLE I. VERILOG CODE FOR PSEUDO-COLOR DESIGN

```
module Pseudo-colouring_filter (
      Rin, Gin, Bin,
      Rout, Gout, Bout,
      reset, clk, data_in_ready,
      ready                              );

    input [7:0] Rin;
    input [7:0] Gin;
    input [7:0] Bin;
    input clk;
    input data_in_ready;
    input reset;
    output reg [7:0] Rout;
    output reg [7:0] Gout;
    output reg [7:0] Bout;
    output reg ready;
    reg [9:0] value;
    reg [9:0] value2;
    reg [9:0] value4;

always @ (posedge clk)
  begin
     if (reset == 1)         ready = 1;

     if (data_in_ready == 1)
     begin
       ready = 0;
       value2 = (Rin + Gin + Bin) / 2;
       value4 = (Rin + Gin + Bin) / 4;
       value = (value2 + value4) / 2;
       if (value == 0 && value < 30) begin
                   Rout = 0;
                   Gout = 0;
                   Bout = 0;
      end
      if (value >= 30 && value <= 70) begin
                   Rout = 0;
                   Gout = 0;
                   Bout = 255;
     end
     if (value > 70 && value < 190) begin
                   Rout = 0;
                   Gout = 255;
                   Bout = 0;
    end
    if (value >= 190 && value <= 249) begin
                   Rout = 255;
                   Gout = 0;
                   Bout = 0;
    end
    if (value >= 250) begin
                   Rout = 255;
                   Gout = 255;
                   Bout = 255;
   end
end
end
endmodule
```

B. *Experimental Results*

The image enhancement operation was described using Verilog hardware description language and the design was tested with Verilog test-benches and simulated using *ISIM Simulator* component from *Xilinx ISE Design Suite 14.5* (Fig.4) and synthesized on a platform *Xilinx Virtex-6 FPGA ML605* based hardware.



Name	Value
Rout[7:0]	00000000
Gout[7:0]	00000000
Bout[7:0]	00000000
ready	1
Rin[7:0]	00000001
Gin[7:0]	00000001
Bin[7:0]	00000001
reset	1
clk	1
data_in_ready	1
file[31:0]	00000000000(
state[2:0]	000
WIDTH[31:0]	00000000000(
HEIGHT[31:0]	00000000000(

Fig. 4. Simulation results using ISIM Simulator.

Verilog cannot handle the standard image formats so the image was converted to a binary-file, using an application created by the author using C Sharp programming language. The binary-file was applied as vector to the Verilog block design. The output file was similarly converted and viewed using the same application, in order show the original image and the results of the enhancement method.

The experimental result obtained by applying the pseudo-color filter to an image, described at RTL level using Verilog HDL, is shown in Fig. 5 and Fig. 6. A medical image (MRI) was applied as an input image (shown in left panel).

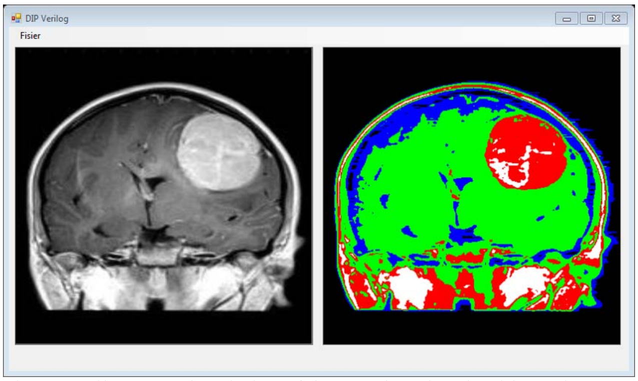


Fig. 5. Verilog HDL description of the pseudo-color algorithm using RGB colors (Filter I).

During Verilog hardware implementation, the logic and memory resources used was maximized to achieve better timing performance. The Verilog design was synthesized and implemented on a platform *Xilinx Virtex-6 FPGA ML605* based hardware and the results with frequency and processing time are shown in Table II.

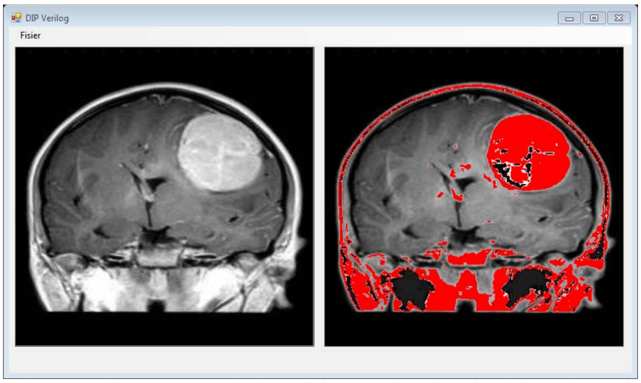


Fig. 6. Verilog HDL description of the pseudo-color algorithm using red color (Filter II).

TABLE II
EXPERIMENTAL RESULTS WITH FREQUENCY AND PROCESSING TIME

	Xilinx Virtex-6 FPGA ML605	
	Frequency [MHz]	**Time [ns]**
Filter I	1281.72	0.780
Filter II	1517.45	0.659

The solution proposed in this paper offer a partial shift from imaging systems based on Digital Signal Processors (DSP) to image processing systems based on Reconfigurable Hardware (FPGA).

The Verilog design described in this paper is part of a larger research oriented towards the use of hardware description languages in medical image processing area. Other hardware medical image processing is to be studied as future work in order to evaluate and prove the advantage of this kind of approach.

III. Conclusion

The main advantage of using hardware description languages to simulate image processing of any logical inputs is related with the possibility of an immediate FPGA based hardware implementation. Since the hardware description language syntax is always related to a hardware structure, the timing information of the potential hardware implementation is also available allowing specific speed optimizations. Out of that, the use of hardware description languages means hardware portability and on-the-fly re-programmability. The main challenge is to transpose the validated algorithms into a non-programming language, as hardware description languages are. Also, the input and output data files need to be reshaped to match the binary content permitted into the hardware simulators.

References

[1] Dougherty, G., “Image analysis in medical imaging: recent advances in selected examples”. Biomed. Imaging Interv. J. 6(3), e32, 2010.

[2] Geoff Dougherty, “Medical Image Processing. Techniques and Applications”, Springer, ISSN 1618-7210, 2011.

[3] Beutel, J., Kundel, H.L., Van Metter, R.L., “Handbook of Medical Imaging”, vol. 1. SPIE, Bellingham, Washington, 2000.

[4] Iuliana Chiuchişan, M. Cerlincă, “Implementation of Real-Time System for Medical Image Processing using Verilog Hardware Description Language”, Proceedings of the 9th International Conference on Cellular and Molecular Biology, Biophysics and Bioengineering (BIO’13), ISSN: 1790-5125, ISBN: 978-960-474-326-1, pp.66-69, Chania, Grecia, 2013.

[5] Sparsh Mittal, Saket Gupta, S. Dasgupta, “FPGA: An Efficient and promising Platform for Real Time Image Processing Applications”, Proceedings of the National Conference on Research and Development in Hardware & Systems (CSI-RDHS 2008), 2008.

[6] Donald G. Bailey, “Design for Embedded Image Processing on FPGAs”, John Wiley & Sons, ISBN: 978-0-470-82851-9, 2011.

[7] Iuliana Chiuchisan, M. Cerlinca, A.D. Potorac, A. Graur, “Image Enhancement Methods Approach using Verilog Hardware Description Language”, Poceedings of the 11th International Conference on Development and Application Systems, “□tefan cel Mare” University of Suceava, ISSN 1844-5039, pp. 144-148, Romania, 2012.

[8] Wilhelm Burger, Mark J. Burge, “Principles of Digital Image Processing – Fundamental Techniques”, Undergraduate Topics in Computer Science, DOI 10.1007/978-1-84800-191-6_4, Springer-Verlag London Limited, 2009.

[9] R. Woods, J. McAllister, G. Lightbody, Y. Yi, “FPGA-based implementation of signal processing systems”, John Wiley, 2008.

[10] Bernd Jahne, “Digital Image Processing”, 5th revised and extended edition, Springer, 2002.

[11] R. Gonzales, R. Woods, “Digital Image Processing”, Prentice Hall, 2008.

[12] John C. Russ, “The Image Processing Handbook”, 6th Edition, CRC Press, 2011.

[13] Zainalabedin Navabi, “Digital Design and Implementation with Field Programmable Devices”, Kluwer Academic Publishers, 2005.

[14] Oana Geman, “Data Mining Tools used in Deep Brain Stimulation Treatments - Analysis Results”, Proceedings, IFIP AICT (Advances in information and Communication Technology) 364, ISBN 978-3-642-23956-5, Springer, 2011 IFIP International Federation for Information Processing, pp. 259-264, 2011.

[15] Oana Geman, Cornel Turcu, “Data Mining Tools and Deep Brain Stimulation Target Evaluation”, Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, September, Prague, Czech Republic, IEEE Catalog Number CFP11803-PRT, ISBN 978-1-4577-1423-8, pp. 701-705, 2011.