Indonesian Journal of Electrical Engineering and Computer Science
Vol. 22, No. 2, May 2021, pp. 818~825
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v22.i2.pp818-825

ISODATA SOPC-FPGA implementation of image segmentation using NIOS-II processor

Radjah Fayçal[1], Ziet Lahcene[2], Benoudjit Nabil[3]
[1,2]LEPCI-Laboratory Ectronics Department, Ferhat Abbas Setif-1 University, Setif, Algeria
[3]Laboratoire d'Automatique Avancée et d'Analyse des Systèmes, Electronics Department, batna-2 University, Batna, Algeria

Article Info

Article history:

Received Aug 7, 2020
Revised Mar 26, 2021
Accepted Apr 2, 2021

Keywords:

Binarisation
Embedded system
FPGA
ISODATA
NIOS system
Segmentation thresholding
System on programmable chip
VHDL

ABSTRACT

This paper presents an FPGA image segmentation-binarization system based on iterative self organizing DATA (ISODATA) threshold using histogram analysis for embedded systems. The histogram module computes pixels levels statistics which are used by the ISODATA algorithm module to determine the segmentation threshold. In our case, this threshold binarizes a gray-scale image into two values 0 or 255. The prototype of the complete system uses an ALTERA CYCLONE-II DE2 kit with a lot of component and interfaces, such as the SD-CARD reader or a camera to read the image to be segmented, the FPGA which will implement the intellectual property (IP) core calculation with the NIOS processor, the VGA interface to view the results, and possibly of the ETHERNET interface for data transfer via internet. The use of FPGA contains the ISODATA, histogram, NIOS processor and others custom altera IPs hardware modules greatly improves processing speed and allows the binarization application to be embedded on a single chipbbn. For the project elaboration, we have used QUARTUS-II software for the hardware development part with VHDL description, SOPC-builder or QSYS for the integration of NIOS-system, and NIOS-II-STB-ECLIPSE for the software program with eclipse c++ langage.

Corresponding Author:

Ziet Lahcene
Department of Electronics, Faculty of Technology
Ferhat Abbas, Setif-1 University, Setif, Algeria
Email: lahcene.ziet@univ-setif.dz

1. INTRODUCTION

Image or video segmentation process is required as a preprocessing step in several image processing and analysis applications such finding what objects are presented in the image, determining the region of interest (ROI) from an image, researching for image-document in optical character recognition (OCR) operations and detecting moving objects in human gait recognition, biomretrics or target tracking [1], [2]. Using segmented images in binarized mode reduces the overall computational load in a specific application. The useful way to binarize an image is to threshold and separates the background and foreground based on pixel intensities.

Several methods (algorithms) such as otsu, isodata, brensen, niblack, sauvola, are implemented to determine the value of the threshold which decides the set of pixels which will take the whitest color and those which take the blackest value [3], [4]. For high-speed real-time applications and embedded systems, it is recommended to use hardware resources (FPGA), since in the case of PCs or DSPs, arithmetic operations and memory demand take a consistent time especially when the size images is great [5], [6].

An embedded system is a computer system that is embedded within a product or component. Consequently, an embedded system is usually designed to perform one specific task, or a small range of specific tasks, often with real-time constraints. Currently, many signal or image processing applications have been embedded in systems to promote a real-time aspect. These applications cover several areas such as: computer vison [7], network, telecommunications [8], medicine, automation, and power electronics [9]-[12].

This work describes a design of efficient hardware architecture for image segmentation-binarization using ISODATA thresholding algorithm. Image gray-scale histogram module is associated to the Iterative Self Organizing DATA (ISODATA) circuit for the analyzing and binarizing operation of data image pixel. The NIOS processor is used in the design to control the traffic between every component and interface.

2. CO-DESIGN METHODOLGY IMPLEMENTATION

Unlike other designs where IPs are directly translated by cogeneration and co-simulation tools with system generator or dsp-builder [13]-[15], this design uses two parts combining hardware and software.

2.1. The structure scheme of the hardware system module

The global hardware mainly includes the main chip EP35F672C6 which is the Altera's Cyclone II series chip [16], peripheral, clock circuit, reset /reconfiguration circuit, power supply, SDRAM, FLASH, SRAM memory circuit and so on chip. The central processing unit and all of the peripherals of NIOS II kernel is custom-designed by QSys or SOPC builder tool, this tool module mainly configurates the CPU of Nios, JTAG, UART, avalon tri-state bridge, on chip memory, SDRAM controller and common flash interface [17], [18].

In the case of this project, the system is designed on programmable chip "SOPC" for an image segmentation system based on the ISODATA thresholding technique Figure 1. In this architecture, the NOIS, soft core processor, delivered by altera controls all the elements instantiated in the architecture by the avalon bus. For different tasks of the segmentation procedure, custom altra university program intellectuals properties (IPs) such: storage, display, memory transfer, acquisition, and transmission are used and instancieded in association with the proposed threshold segmentations modules.

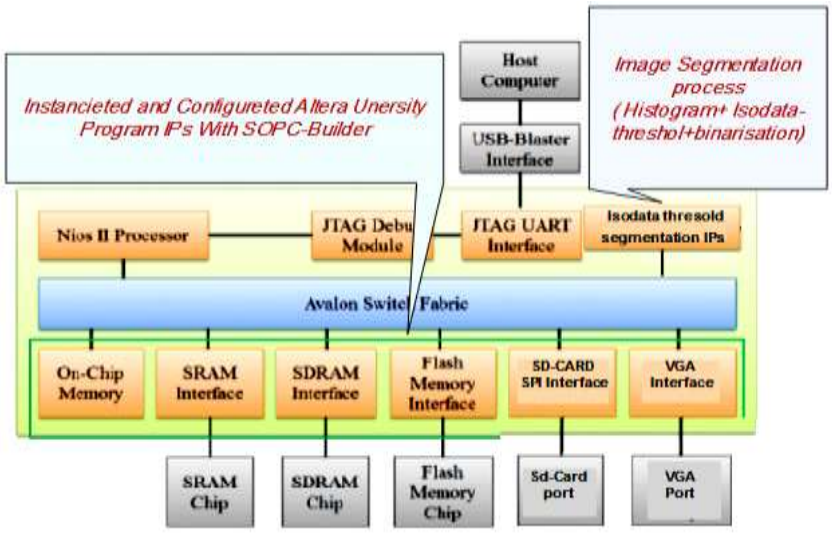

Figure 1. Architecture of image segmentation design with system on programmable chip (sopc)

The implementation of the complete system involves co-design hardware-software procedure:

a) SOPC Builder functionality, which accordingly connects the soft-hardware components to construct a complete computer system that can be controlled on any of the FPGA chips and is also capable of producing interconnect logic automatically Figure 2.
b) Eclipse IDE framework and the eclipse C development toolkit (CDT) plug-ins, all software development tasks, including editing, building and debugging, can be accomplished using NIOS II IDE [19].

Figure 2. NIOS-II to compnent connections port-map

2.2. Hardware isodata image thresholding

Thresholding by ISODATA algorithm consists of finding a threshold by separating the histogram into two classes iteratively with the prior knowledge of the values associated with each class. This method begins by dividing the interval in non-zero values representing *background population*C_0 *and foreground population* C_1 of the histogram into two equidistant parts, then calculating the arithmetic means m1 and m2 of each class. Repeat the calculation of the threshold T until convergence to the value closest to (m1 + m2) / 2, and each time updating the two averages m1 and m2: see the steps of the algorithm below [20].

Algorithm1: Isodata" threshold selection based on the iterative method by Ridler and Calvard
1: K ← size (h) -- *number of intensity levels*
2: T ← mean (h, 0, K−1) -- *set initial threshold to overall mean*
3: **repeat**
4: C_0← count (h, 0, T) -- *background level population*
5: C_1← count (h, T+1, K−1) -- *foreground level population* $\boldsymbol{Count(h,a,b)} = \sum_{g=a}^{b} \boldsymbol{h(g)}$
6: **If** (C_0=0) (C_1=0) **then**
7: **return -1** -- *background or foreground empty*
8: m1← mean (h, 0, T) -- *background mean*
9: m2← mean (h, T+1, K−1) -- *foreground mean*
10: T' ← T -- *keep previous threshold*
11: T ← [m1 + m2]/2
12: **Until** T = T' -- *end of the loop if no-change*
13: **return T**

The histogram of a gray-scale image, gives an account of the number of pixels in an image as a function of pixel value. The two main steps associated with using histograms are used to build the histogram, and to extract data from it and use it for processing of the image.

$$h[i] = \sum_{x,y} \begin{cases} +1, when & I[x,y] = i \\ 0, & otherwise \end{cases} \quad (1)$$

Histogram is then a diagram used in analyzing digital data, which depicts how many pixels of an image or a video frame have certain intensity. It is often required for many applications in image and video processing or for evaluation measure. Figure 3 depicts how to use counters or memory to implement histogram of an image [21], [22].

For this application, a memory with dual port data and dual port address is used with a decoder and accumulator adder. Input image pixels are accumulated for each gray level of image, at the end, the memory contains values which represent the histogram. The Isodata threshold method calculation unit, Figure 4, consists of several parts and operates according to Algorithm 1.

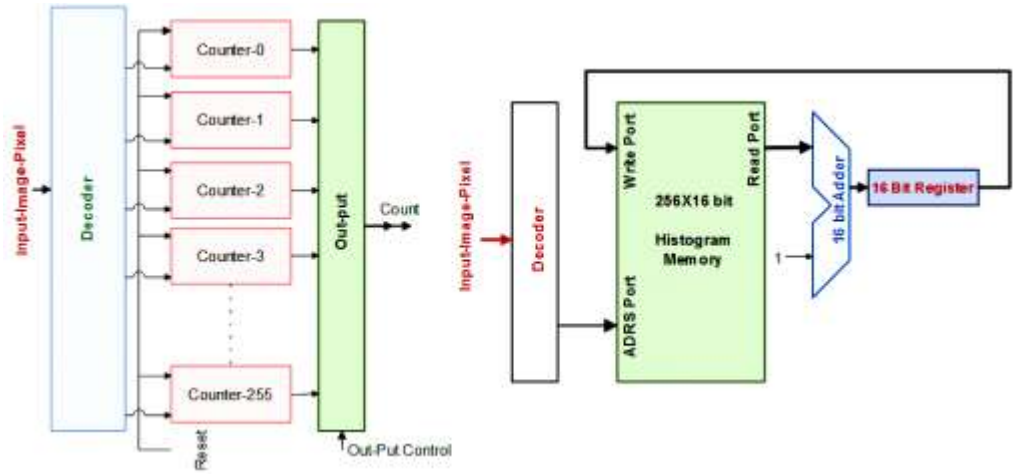

Figure 3. Histogram implementation circuit with memory VS counter

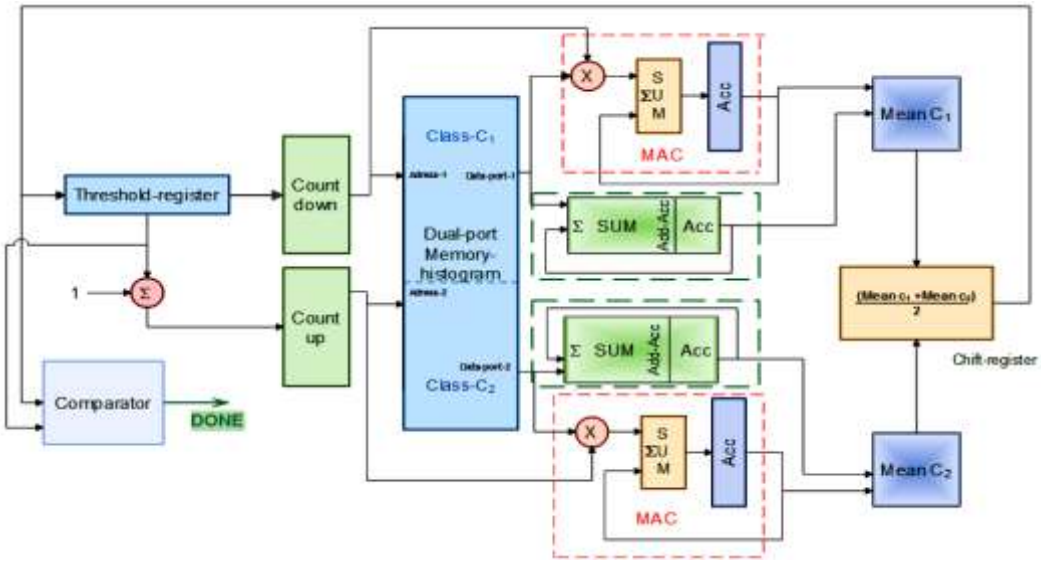

Figure 4. ISODATA-thresold compute unit

After obtaining the histogram, Figure 3, the level pixel values statistics are inside a dual input port and dual address port memory and will be divided into two classes C1 and C2 using the Threshold-register. At the beginning this register contains an initial threshold and then the values of the thresholds calculated in the following iterations.

To scan the two areas class-C1 and class-C2, the computing unit uses a down-counter initiated by the value of the threshold register and an up-counter initiated by the value threshold +1. This makes processing faster for iterations process. The two outputs of the memory are weighted and accumulated respectfully by the contents of values of the down-counter and of the up-counter by the multiplication accumulated MAC module. At the same time the values of each class (memory data output) are accumulated by an Accumulating adder ''Add-Acc''. The results obtained by the MAC and ADD-ACC operations lead to obtain the respective moments for each class of the histogram.

The average of these moments produced by a right shift register is transferred to the threshold register for a new iteration if this value is not equal to the value calculated in the previous iteration, otherwise a signal done is sent to the processor for indicates the end of this step and the start of the binarization step.

The last unit used in the segmentation process is the construction of the binirized image. In this phase each pixel in the image-memory is transformed in background (pixel=0) or forground (pixel=255) according the value of Isodata-threshold Figure 5.

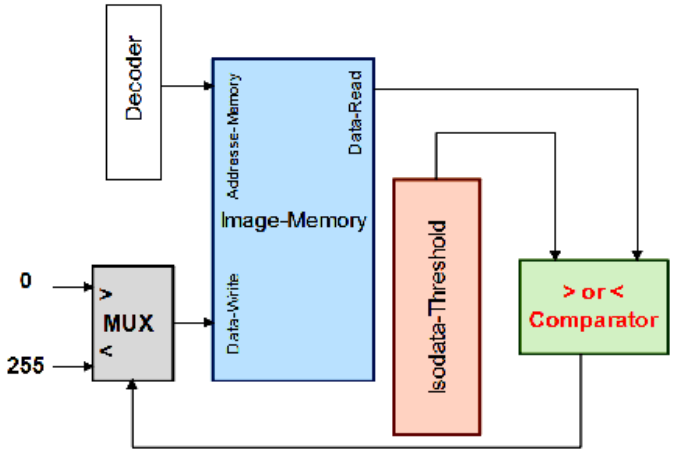

Figure 5. Image binarization circuit

2.3. Software design phase

In order to test the SOPC, an application program for NIOS II processor should be written. Integrated development environment is a software development graphical user interface (GUI) for NIOS II processor. It is based on the eclipse IDE framework and the eclipse C development toolkit (CDT) plug-ins. All software development tasks, including editing, building and debugging, can be accomplished using NIOS II IDE.

The application begins with initializations of the various interfaces and variables. It then checks for the presence of a storage or capture element element (SD-Card in this case or camera in use). The NIOS processor gives the start of the treatment entrusted to the hardware module of binarization. Once finished, a signal is emitted indicating the end of the treatment. The binarized image is now ready to be displayed or stored in the SD-Card Figure 6.

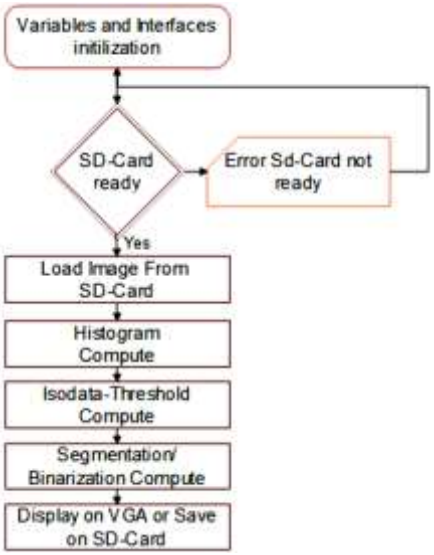

Figure 6. Software algorithm for NIOS-II processor

3. SYNTHESIS AND SIMULATION AND DISCUSSION

The synthesis result obtained for the design architecture is presented in Table 1. The design is described using structural VHDL [22] and synthesized on the cyclone FPGA II EP2C35F672C6. The RTL level view of a part of MAC module is represented in Figure 7 and the report of its synthesis operation is shown in Figure 8.

Table 1. Synthesis results of the designed architectures-modules

Modules	Logic Elements	registers	Memory Bits	DSP Multipliers
Dual-Port Histogram Memory	-	-	4 K	-
MAC	33 x2	32 x2	-	2 x2
SUMAcc	17 x2	16 x2	-	-
Increment	8	-	-	-
Threshold Register	8	8	-	-
Count-UP Count-Down	8 x2	8 x2	-	-
Comparator	5	1		
Means-C1 Means-C2	858 x2	-	-	-
Total	1853	121	4 K	4

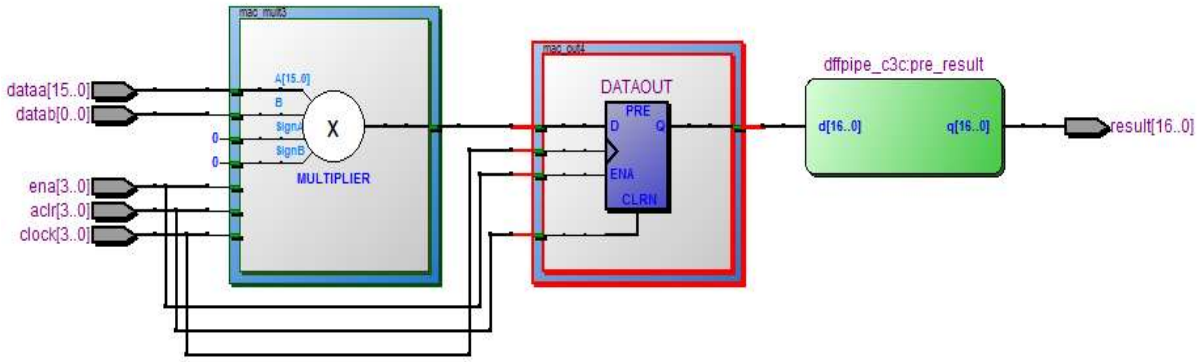

Figure 7. Sample of register transfer level of a portion of the system

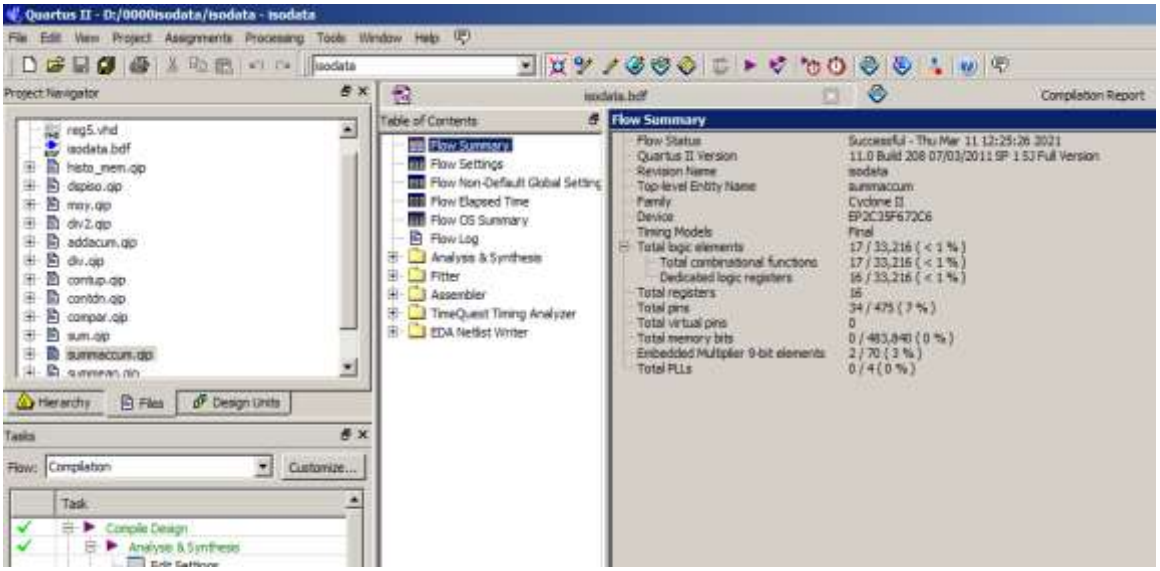

Figure 8. Synthesis compilation report delivred by quartus compilation

4. RESULTS AND DISCUSSION

To highlight the effectiveness of this work, the developed application is subjected to various tests. Figure 9(a) and Figure 9(b) show respectly the DE2-Kit, the out-put state of the prototyping card DE2 with initialized configuration. Figure 9(c) shows the display for the blank system without loading the program into the program flash memory.

Figures 10 shows the display of the test images (originals images old-printing and hand-writing) [23], [24]. The results of the segmentation operation are given by Figure 11. We noticed that the results obtained are very close to those obtained using software applications, despite the low precision of the number of bits to represent the processing data.

Figure 9. Altera DE2-kit and primary display

Figure 10. Exemple of original old-documents images display [23], [24]

Figure 11. Old-document images thresholded

This choice of image is justified to see the improvement in the execution latency compared to other works in particular the work of Khitas *et al.* [25] to implement his method. Compared to conventional procedures, performed on a Pentium IV 3.00 Ghz, the execution speed is significantly better when the NIOS is clocked by a clock of only 100 Mhz. This speed improvement is due to the parallel aspect of task execution in hardware systems.

5. CONCLUSION

The hardware implementation can be more beneficial especially for image or video processing applications since the pipeline is more suitable and the FPGA modules are dedicated for this type of tasks. This work puts forward an image pre-processing design scheme based on SOPC on DE2 carte. The objectives of the project were to review developments in embedded system design and future trends, and to explore board-level rapid prototyping using FPGAs (DE2). A good example is the designed application of image binarization by calculation of the ISODATA threshold analyzing the histogram. It shows that the use of the hardware approach in the images designs gives better results in terms of speed, size and cost. The next idea

would be to continue to increase the implementation complexity level for typical image processing for OCR, Document image analysis, Finger-vein pattern extraction and ngerprint preprocessing embedded applications.

REFERENCES

[1] N. Chaki *et al.,* "Exploring Image Binarization Techniques, Studies in Computational Intelligence," *Exploring Image Binarization Techniques*, pp. 5-15, 2014, doi: 10.1007/978-81-322-1907-1_1.

[2] Wan Azani Mustafa *et al.,* "Binarization of Document Images: A Comprehensive Review," *International Conference on Green and Sustainable Computing (ICoGeS),* vol. 1019, no. 1, 2018, doi: 10.1088/1742-6596/1019/1/012023.

[3] Moghaddam, R.F., Cheriet, M. "AdOtsu: an adaptive and parameter less generalization of Otsu's method for document image binarization," *Pattern Recogn,* vol. 45, no. 6, p. 24192431, 2012, doi: 10.1016/j.patcog.2011.12.013.

[4] Gatos, B., Pratikakis, I., Perantonis, S.J.: "Adaptive degraded document image binarization," *Pattern Recogn,* vol. 39, no. 3, pp. 317-327, 2006.

[5] Fahad Siddiqui, Sam Amiri, Umar Ibrahim Minhas, Tiantai Deng, Roger Woods, Karen Rafferty and Daniel "Crookes FPGA-Based Processor Acceleration for Image Processing Applications," *Journal of Imaging*, vol. 5, no. 1, p. 16, 2019, doi: 10.3390/jimaging5010016.

[6] Tomasz Kryjak, Mateusz Komorkiewicz Marek Gorgon, "Real-time hardware–software embedded vision system for ITS smart camera implemented in Zynq SoC," *Journal of Real-Time Image Processing*, vol. 15, no. 1, pp. 123-159, 2018, doi: 10.1007/s11554-016-0588-9.

[7] Eric Chung *et al.,* "Accelerating Persistent Neural Networks at Datacenter Scale", *HotChips*, vol. 29, 2017.

[8] Hebibi Amar, Arres Bartil, Lahcene Ziet, "Comparison of two new methods for implementation BPSK modulator using FPGA," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 19, no. 2, pp. 819-827, 2020, doi: 10.11591/ijeecs.v19.i2.pp819-827.

[9] J.M.P. Cardoso, "On combining temporal partitioning and sharing of functional units in compilation for reconfigurable architectures," *Computers, IEEE Transactions on*, vol. 52, no. 10, pp. 1362-1375, 2003, doi: 10.1109/TC.2003.1234532.

[10] Peng Li *et al.,* "FPGA Acceleration for Simultaneous Medical Image Reconstruction and Segmentation," *Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, p. 172, 2014, doi: 10.1109/FCCM.2014.54.

[11] Mahmoud Matar, Reza Iravani, "FPGA Implementation of the Power Electronic Converter Model for Real-Time Simulation of Electromagnetic Transients," *IEEE Transactions on Power Delivery*, vol. 25, no. 2, pp. 852-860, 2010, doi: 10.1109/TPWRD.2009.2033603.

[12] Shanker Shreejith and Suhaib, "A. Fahmy. Extensible FlexRay Communication Controller for FPGA-Based Automotive Systems," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 2, pp. 453-465, 2015, doi: 10.1109/TVT.2014.2324532.

[13] Taoufik Saidani *et al.,* "Using Xilinx System Generator for Real Time Hardware Co-simulation of Video Processing System," *Lecture Notes in Electrical Engineering*, vol. 60, pp. 227-236, 2020, doi: 10.1007/978-90-481-8776-8_20.

[14] Lei Zhang, "System generator model-based FPGA design optimization and hardware co-simulation for Lorenz chaotic generator," *2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, 2017, pp. 170-174, doi: 10.1109/ACIRS.2017.7986087.

[15] Raya Kahtan Mohammed, Hamsa Abdulkareem, "Implementation of digital and analog modulation systems using FPGA," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, no. 1, pp. 485-493, 2020, doi: 10.11591/ijeecs.v18.i1.pp485-493.

[16] https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=53&No=30&PartNo=2

[17] Altera Embedded Peripherals IP Guide. 2011. Available www.altera.com/literature/ug/ug_embedded_ip.pdf.

[18] Altera Audio/Video Configuration Core for DE2-Series Boards. (July 2010). [Online]. Available:

[19] ftp://ftp.altera.com/up/pub/Altera_Material/10.1/Universiy_Program_IP_Cores/Audio_Video/Audio_and_Video_Config.pdf.

[20] T. W. Ridler and S. Calvard. "Picture thresholding using an iterative selection method," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, no. 8, pp. 630-632, 1978.

[21] Ziet Lahcene, Khitas Mehdi and Radjah Fayçal, "Implementation of image histogram for image binarization," *International Conference On Advances In Science*, Istambul 31 august-2 september 2016.

[22] https://standards.ieee.org/standard/1076-2019.html.

[23] Pratikakis I, Gatos B and Ntirogiannis "K 2013 ICDAR 2013 document image binarization contest" *Proceedings of the International Conference on Document Analysis and Recognition*, ICDAR, pp. 1471, 2013, doi: 10.1109/ICDAR.2011.299.

[24] Pratikakis I, Zagoris K, Barlas G and Gatos B "ICDAR2017 Competition on Document Image Binarization," (DIBCO 2017) *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 1, pp. 1395-403, 2018, doi: 10.1109/ICDAR.2017.228

[25] Khitas Mehdi, Ziet Lahcene, Saad Bouguezl, "Improved Degraded Document Image Binarization Using Median Filter for Background Estimation," *Elektronika ir Elektrotechnika*, vol. 24, no. 3, pp. 82-87, 2018, doi: 10.5755/j01.eie.24.3.20982.