Blood Group Identification using FPGA

Stella J[a], Tejal Deshpande[b]

[a]Dept of Electronics and Communication Loyola Institute of Technology & Science Kanyakumari, India
stellaeceengineer@gmail.com
[b]Dept of Electronics & Telecommunication Xavier Institute of Engineering, Mahim, Mumbai tejal.d@xavier.ac.in

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract: Detection of Blood Group is the first essential proce- dure for major medical surgery. Traditional ways of laboratory testing techniques like plate test and tube test are susceptible to human fallibility and, the approach of detecting blood groups is still prevalent in our modern era of digitization. As it is time-consuming, the results are difficult to reproduce, and there is a need for a domain expert. Our present times necessitate automation of the evaluation process. This approach would be a lifesaving method for Blood Group Identification.

The proposed system uses a technique of testing using FPGA. Canny Edge Detection Algorithm developed using MATLAB Simulink and converted to HDL Code for Hardware Implemen- tation on Arty 7000 Zync FPGA using Xilinx Vivado software as FPGAs have the advantages of speed and reconfigurability for image processing applications. The Detected edges counted to classify blood group. The proposed Hardware Model FPGA demonstrates 2.5 V power supply with the power usage of 1.83 W. Its Hardware implementation requires only 10.773ns, while Intel Core i5 processor with 1.60 GHz clock rate and 8 GB RAM takes 120 s upon software-based MATLAB implementation. An accuracy of 99% achieved when tested with each type of blood with more than 100 various blood samples produced advantages like portable, affordable, durable, and more limited power consumption than conventional methods of detecting blood group by the proposed hardware solution.

Keywords: ABO system, Agglutination, Simulink, Canny edge detection, FPGA, Blood grouping

1. Introduction

Approximately 1.35 million people die each year, and the over-speeding of vehicles being the biggest reason for casualties [1]. It is at these times that one has to perform an urgent blood transfusion. Often the blood groups are not known, and one has to determine a blood group of the victim. Also, in public health centers, there is a large volume of blood groups to be determined. As far as India is concerned, the rural population lacks access to labs and trained technicians. The Population of India is close to 1.37 billion or 1,369 million in 2020. 68.86% of Indian lives in rural areas and 31.14% lives in urban areas [2]. At the time of natural and human-made disasters, blood group transfusions are of utmost importance, and human errors can lead to catastrophic results. The conventional system to find the blood group types may take a long time, and also trained health technicians may not be available. Even if present, they have to make quick tests and decisions. Often in these cases, the universal donor blood is administrated. It may result in some reactions risking the patient's life, and also Universal donor blood stock levels decrease. The proposed system is an automatic blood group detection system which is, implemented with Hardware and Software Co-Simulation. It can perform the basic pre- transfusion test easily and quickly with high reliability and even in remote locations.

A. ABO System

Presence and Absence of Antigen A and B on the surface of the Red blood cells (RBC) classify the human Blood group. It categorized into four Groups named Group A has A antigen, Group B has B antigen, Group AB has A and B antigen, and Group O has no antigen on the surface of the RBC as shown in Figure 1.

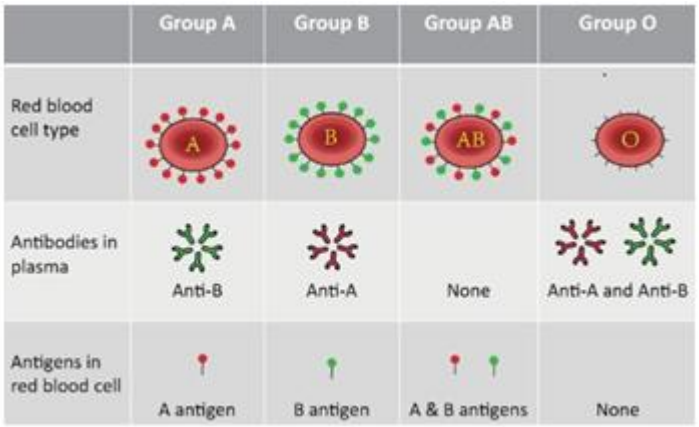

Fig. 1. ABO System

B. Rhesus (Rh) System

The Rhesus System is the second Blood group system. Presence and Absence of the Rh antigen called Rh factor classify the blood into Rh-positive and Rh-negative. The Absence of Rh antigen in the blood is said to be Rh-negative, and the presence of Rh antigen to be Rh-positive blood type. RBC Compatibility using ABO and Rh system as shown in Figure 2.

2. Related Work

Some scholars researched in the field of medicine, and pub- lished papers such as publications that have been studied, and evaluated during the implementation of the proposed method- ology. The author in [6] used Plate test over the captured image and used the Vivado System Generator for various image.

Recipient	Donar							
	O -ve	O +ve	A -ve	A +ve	B -ve	B +ve	AB -ve	AB +ve
O -ve	✓	×	×	×	×	×	×	×
O +ve	✓	✓	×	×	×	×	×	×
A -ve	✓	×	✓	×	×	×	×	×
A +ve	✓	✓	✓	✓	×	×	×	×
B -ve	✓	×	×	×	✓	×	×	×
B +ve	✓	✓	×	×	✓	✓	×	×
AB -ve	✓	×	✓	×	✓	×	✓	×
AB +ve	✓	✓	✓	✓	✓	✓	✓	✓

Fig. 2. Red Blood Cell Compatibility

processing algorithms to detect Edges and considered to define the blood group. A model-based design approach applied for image processing algorithms furthermore implemented in hardware Software co-simulation in [7]. FPGA based design implemented for the Canny edge detection algorithm in [9] as it detects the edges in the sharp discontinuities. In [3] the Sobel edge detection algorithm is employed to detect the checker box edges to generate HDL code. Space complexity is reduced compared with the other edge detection approaches as it is best for parallelism in [4] Implementation of parallel processing algorithm on FPGA are being used with image processing procedures to categorize considerable amounts of blood samples and the proof of concept in [8] with edge detection in MATLAB Simulink and Xilinx System generator. The author proposed [10] Verilog HDL Implementation for the Sobel edge method using Parallel Processing. The system utilizes Xilinx ISE 12.4 resources at the hardware model with a maximum of 200MHz frequency. The author [5] used Vertex 6 FPGA from Xilinx ISE software, and it requires 6.6 s to identify when it tested with 500 different blood samples. In [11] the author has proposed an embedded system that uses image processing algorithms to perform a blood test based on ABO and Rh blood typing system. System developed [12]to perform blood tests in less time. IMAQ Vision from National Instruments was used to obtain the slide test images. Image Processing algorithms applied to captured images, and blood classified. A database of the patients was maintained to store the information.

3. Design Methodology

A. FPGA

Image Processing and Video Processing is preferably suit- able for FPGAs (Field Programmable Gate Arrays) because of its reconfigurability of computing Architecture. As the name implies reconfigurable, it can reprogram several times depends upon the designer. A wide array of interconnects such as Flip flop (FF), Random Access Memory (RAM), Lookup Table (LUT) wired together in FPGA. The FPGA Architectural designs allow lots of various logics and design that can integrate into the new design based on the resources in processors. Comparing to DSP and ASIC Nowadays, FPGAs are designed and implemented in Parallel Design Architectures because the design for DSP is not suitable for Parallel design, and ASIC only allows the designer to develop in the gate-level model. FPGA uses Hardware languages like Verilog or VHDL for designing a model.

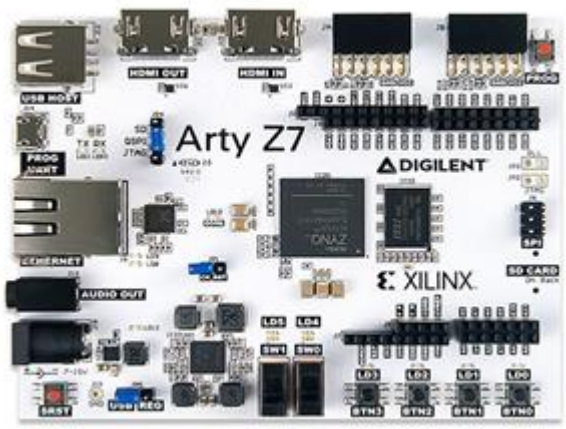

Fig. 3. FPGA Board

The Proposed Model uses Arty Zync 7000 Board for the development of Blood Group Identification design. Arty Z7 is a development platform for embedded vision. A dual-core, ARM Cortex -A9 with 650Mhz processor and Xilinx 7-series is integrated into Zynq 7000 as shown in Figure 3. The target application with both the combination gives the ability to enclose a powerful processor with controllers and aunique set of software-based peripherals.

B. Experimental Setup

Experimental Setup shown in Figure 4 as the plate test image is captured and send to the personal computer for further processing.

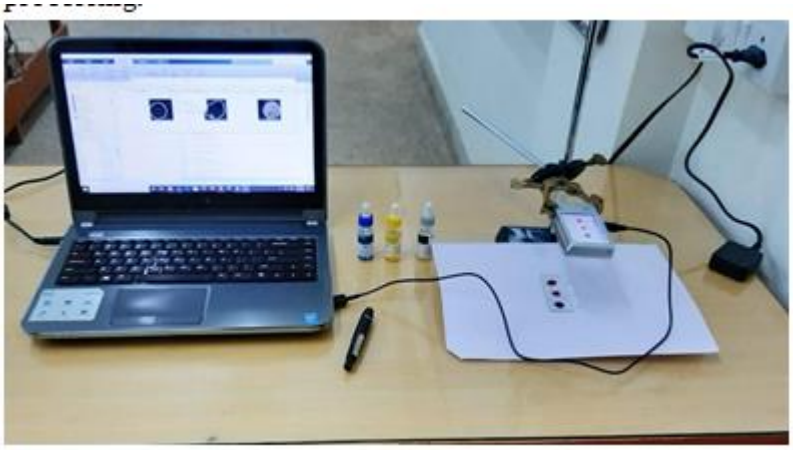

Fig. 4. Experimental Setup

C. Flow Diagram

The traditional approach of the Plate test in which the blood samples (antigens) taken in the slide and blood drops mixed with the reagents (antibodies). Agglutination occurs based on the reaction between antigens and antibodies. Laptop with Core i5 processor, 8GB RAM and Windows 10 - a 64-bit operating system which process a Model designed in MATLAB Simulink in which the model contains number of blocks with the help of Simulink library browser from that,

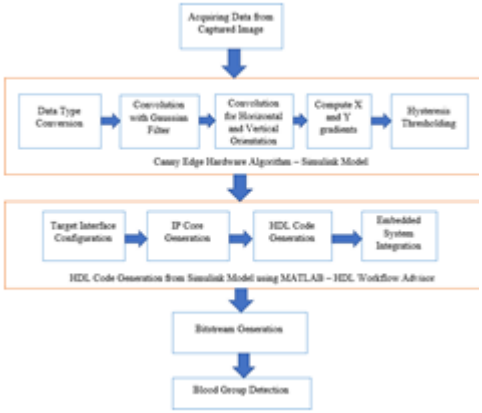

Fig. 5. Experimental Setup

Canny edge Algorithm function block which is selected to generate the HDL code and it is downloaded into the Xilinx Arty Zync 7000 board to identify the detection of blood as shown in the Figure 5

4. Implementation

A. Acquiring Data

The Input image subsystem block contains a block named Image from file from the Simulink library that stores the grayscale format of the input. The paper presents O negative blood sample for evaluation shown in Figure 6. Generally, a Grayscale image indicates where each of its pixels is representing a range of a particular amount of light. After, Grayscale computation, it serialized to compute the canny edge algorithm.

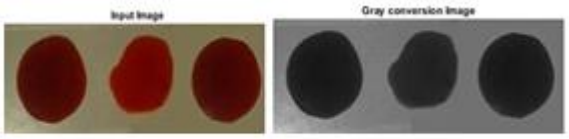

Fig. 6. a) RGB Image O -ve b)Gray Converted Image O –ve

B. Canny Edge Algorithm - Simulink Model

- ***Data Type Conversion***: Convert block, which converts the data type into a fixed-point data type. Serialize block uses a matrix transpose block where the transposed image represented in 1 Dimensional, and the samples sent through buffer. The buffer sends the samples to the Canny edge algorithm with a delay. as shown in Figure 7. The

Fig. 7. Processing of Canny Edge Algorithm

algorithm has four steps, which are (a) convolution with Gaussian filter of the serialized image (b) Convolution with Canny Filter for Horizontal and Vertical orientation

(c) Compute X-gradient and Y-gradient (d) Hysteresis Thresholding Convolution with Gaussian Filter: Gaussian Filter is applied to the samples to separate the noise to prevent erroneous exposure in the image. Convolution of an image with 5x5 Gaussian Kernel shown in equation (1) smoothens the given input to diminish the noise present on edge.

$$B = 1/59 * \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 5 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * \text{image} \qquad (1)$$

Convolution for Horizontal and Vertical orientation: Convolution with Gaussian Filter representation with any of the filter masks from Sobel, Roberts, Perwitts is accomplished to identify the diagonal, horizontal, and vertical edges upon the blurred image.

The proposed system uses Sobel Mask that uses 3x3 Kernels used to delay the input pixel to have the pixels in a 3-by-3 window centering at the current output location.

- ***Compute X-gradient and Y-gradient***: Sobel Mask for vertical direction (Gx) shown in equation (2) and hor- izontal direction (Gy) shown in equation (3) is applied to identify the approximations of directives to convolve with the Gaussian Filtered image.

$$Gx = \begin{bmatrix} 2 & 4 & 5 \\ 4 & 9 & 12 \\ 5 & 12 & 5 \end{bmatrix} * B \qquad (2)$$

$$Gy = \begin{bmatrix} 2 & 4 & 5 \\ 4 & 9 & 12 \\ 5 & 12 & 5 \end{bmatrix} * B \qquad (3)$$

From the x and y gradient, Edge magnitude shown in the equation (4) and Edge direction as shown in the equation (5) are computed.

$$G = Gx^2 + Gy^2 \qquad (4)$$

$$\theta = atan2(Gy, Gx) \qquad (5)$$

- ***Hysteresis Thresholding***: Weak gradient value is sepa- rated and high gradient value preserved from the edge pixels by selecting the threshold values low and high. In hysteresis thresholding, the edge pixel with high gradient value remains, and the edge pixel with a weak gradient gets suppressed to provide a more reliable result

C. Canny Edge Algorithm - HDL Code Generation

The HDL Workflow Advisor guides through several stages to generate HDL code for a Simulink subsystem and the Embedded system Integration. The Subsystem to generate the HDL code, selected by right-clicking on it to open the HDL Workflow advisor. The left pane lists the folders in the hierarchy shown in the Figure. Each folder represents a group of categories or related tasks.

- ***Target Interface Configuration***: The Target Interface Port is chosen for the input and output interface to generate an HDL code as shown in Figure 8.

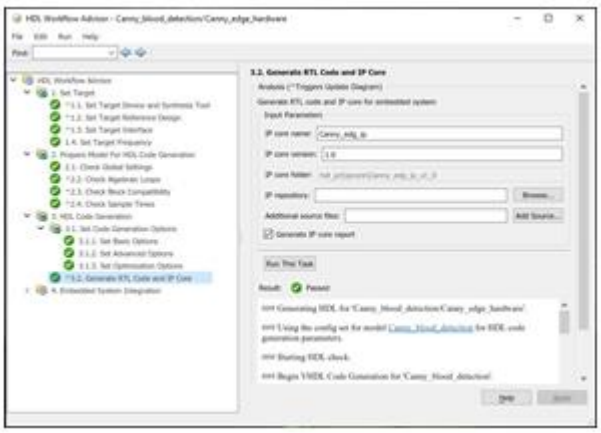

Fig. 8. HDL Workflow advisor

- ***IP Core Generation***: AXI4-Lite acts as an interface between the IP core and embedded processor, here the processor and IP core acts as a master and slave. IP core is controlled by the processor that can read and write data into it, and the generated register accessed through the AXI4-Lite interface as shown in Figure 9. This IP

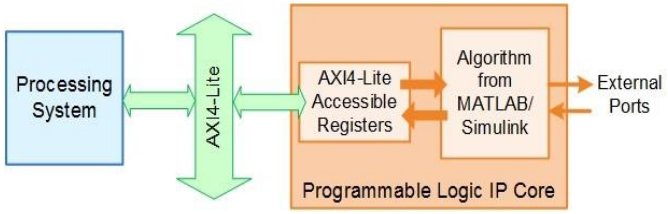

Fig. 9. AXI4 Lite interface

core supports the External Port interface. To connect the external ports to the FPGA external IO pins, it needs to add FPGA pin assignment constraints in the Xilinx Vivado environment.

- ***HDL Code Generation***: HDL Code and model advisory report of each stage in the hierarchy is produced once every task in the hierarchy completed successfully.

- ***Processor/FPGA Synchronization***: The Free running mode is chosen on the Target Interface as there is no explicit synchronization between embedded processor software execution (SW) and the IP core (HW). SW and HW run independently. The data written from the processor to IP core takes effect immediately, and the data read from the IP core is the latest data available on the IP core output ports.

- ***Embedded System Integration***: Embedded System Inte- gration tasks in HDL Workflow Advisor helps to integrate the generated IP core into the Xilinx Vivadoenvironment.

D. Xilinx Vivado Environment

The Generated IP core is a zip file that added to the IP repository on the Vivado Project. The HDL Coder Generated IP added to the Vivado block design. AXI4 Lite port, Clock, and reset of IP core connected to the ports of Embedded Processor shown in Figure 10. FPGA bitstream generated

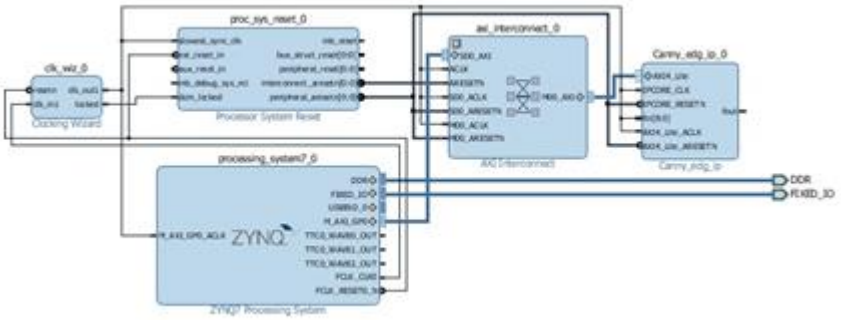

Fig. 10. Block Design of Canny Approach

to integrate into FPGA (Arty Zync 7000 FPGA Board) for evaluation. The device Utilization shown in the Figure 11

Resource	Utilization	Available	Utilization %
LUT	430	53200	0.81
LUTRAM	62	17400	0.36
FF	572	106400	0.54
BUFG	3	32	9.38
MMCM	1	4	25.00

Fig. 11. Device Utilization

E. Classification of Blood

The FPGA Board generates a text file of bitstreams for detected edges shown in Figure 12. The bitstreams loaded in the MATLAB and the image is divided into three parts to count each edge present in the image shown in Figure 13, and the values of the counted edges are,

Group A = 1 Group B = 1

Group Rsh = 1 as shown in the Figure 14.

Fig. 12. Edge Detected Output of blood type O -ve

5. Experimental Results

Based on the Threshold value, the Result is categorized. The minimum threshold value taken in the proposed system is 5. Depend upon the agglutination on the blood sample, the detected edge may vary. The categorization of blood samples depends on the counted edge value is more or less than the threshold value shown in Figure 13,14. The threshold values obtained for different kinds of blood sample are shown in the Table I

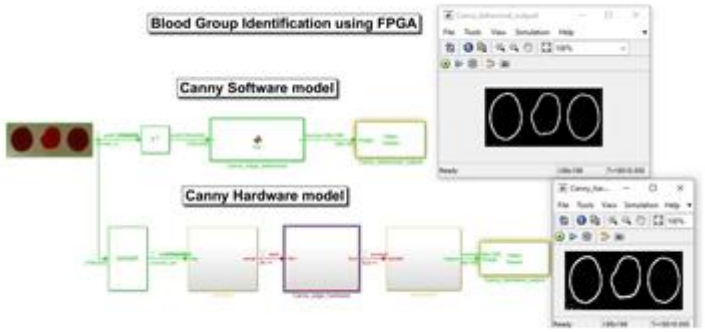

Fig. 13. Blood Type O -ve after Canny Edge Detection

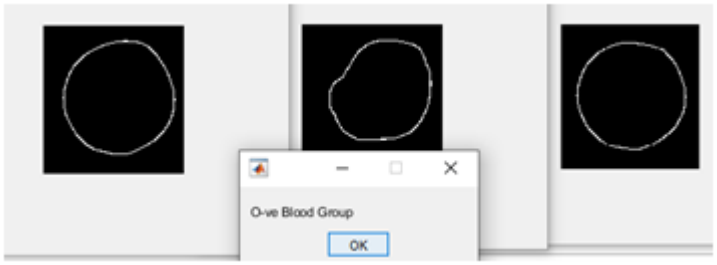

Fig. 14. Counted Edges to detect the type of Blood

A. Comparative Analysis

The Sobel Edge Detection Methods in [3],[10] shows limita tion over inaccurate edge prediction with time consumption of 30 s and 6.6 s where the new proposed canny edge method pro- duces accurate results as it detects the edges in discontinuity with faster response of 10.773ns. The Comparative Analysis of Various Edge Detection Algorithms are Tabulated as shown in Table II

6. Conclusion

The Hardware Implementation showed in the paper is for individuals. The method is very accurate to detect the edges, and it performed with Real-time Blood samples are collected with different age groups of peoples, and it is fully automated. Implementation on Arty Zync 7000 has less utilization with

1.83 W usage of power from a 2.5 V supply voltage. This

TABLE I COUNTED EDGES OF BLOOD SAMPLES

Sample	Detected Edges of each Blood Group: Number of edges in Part A	Number of edges in Part B	Number of edges in Rh factor
O+ve	3	1	25
A+ve	12	2	40
B+ve	2	30	36
AB+ve	43	25	44
O-ve	1	1	1
A-ve	25	2	2
B-ve	3	14	2
AB-ve	16	25	2

TABLE II COMPARATIVE ANALYSIS OF PREVIOUS AND PROPOSED TECHNIQUE

Approach	Operator	Operation	Software Used	Time Consumes
Previous Systems	Sobel	Gradient magnitude Gx, Gy is calculated for the input image with Mask	MATLAB Simulink + Vivado System Generator	30 s
			MATLAB Simulink + Xilinx ISE	6.6 s
Proposed System	Canny	Edges identified by local maxima of the gradient (Gx and Gy) are calculated by taking derivative of Gaussian Filter	MATLAB Simulink + Xilinx Vivado	10.773 ns

procedure requires only 10.773ns, while MATLAB implementation on an Intel i5 Core processor with 1.60 GHz clock rate and 8 GB RAM takes 120 s. So the proposed system achieves the advantages of accuracy, reliability, reconfigurability with faster response

References

1. Available at https://www.who.int/news-room/fact-sheets/detail/road- traffic-injuries Accessed on 11th July 2020.
2. Available at http://www.indiaonlinepages.com/population/india-current- population.html Accessed on 11th July 2020.
    A. B. Krishnan and K. P. Peeyush, ”Blood group determination using vivado system generator in Zynq SoC,” in 7th WACBE World Congress on Bioengineering 2015, 2015.
3. D. P. Guragain, P. Ghimire and K. Budhathoki, ”Implementation of FPGA Based Image Processing Algorithm Using Xilinx System Gen- erator,” International Research Journal of Engineering and Technology (IRJET), vol. 5, pp. 2395-56, 2018.
4. S. Ravi, A. Rahim and F. Shaik, ”Fpga based design and implementation of image edge detection using xilinx system generator,” International Journal of Engineering Trends and Technology, vol. 4, pp. 4657-4660, 2013.
5. Available at https://in.mathworks.com/help/hdlverifier/examples/verify- sobel-edge-detection-algorithm-in-matlab-to-hdl-workflow.html.
6. N. Nausheen, A. Seal, P. Khanna and S. Halder, ”A FPGA based imple- mentation of Sobel edge detection,” Microprocessors and Microsystems, vol. 56, pp. 84-91, 2018.
7. S. K. Areefabegam and T. Narendrakumar, ”FPGA based design and implementation of image edge detection using Xilinx System Generator,” International Journal of New Trends in Electronics and Comunication, vol. 2, pp. 18-21, 2014.

8. V. Sanduja and R. Patial, ”Sobel edge detection using parallel archi- tecture based on FPGA,” International Journal of Applied Information Systems, vol. 3, pp. 20-24, 2012.
9. T. Zarifi and M. Malek, ”FPGA implementation of image processing technique for blood samples characterization,” Computers & Electrical Engineering, vol. 40, pp. 1750-1757, 2014.
10. Vinay M “Automated Blood Group Detection System using Image Processing” International Journal of Advanced Research in Electronics and Communication Engineering, 2017.
11. Dada, D. Beck and G. Schmitz, ”Automation and data processing in blood banking using the Ortho AutoVue® Innova System,” Transfusion Medicine and Hemotherapy, vol. 34, pp. 341-346, 2007.