International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 /doi:10.14445/22315381/IJETT-V69I9P207

Volume 69 Issue 9, 51-55, September, 2021
©2021 Seventh Sense Research Group®

Original Article

Enhanced MAC Controller Design for 2D Convolution Image Processing on FPGA

Chiranjeevi G N[1], Dr Subhash Kulkarni[2]

[1]*Assistant Professor, Department of Electronics & Communication, PESIT –BSC*

[2]*Principal, PESIT-BSC (PESIT-Bangalore South Campus)*

(Affiliated to Visveswaraya Technological University, Belgaum)

Bangalore, Karnataka, India

[1]chiranjeevign@pes.edu , [2]sskul@pes.edu

Abstract - *Majorities of image processing algorithms are two-dimensional (2D) and localized by their very nature. As a result, the 2D convolution function has significant implications for the requirements involving image processing. 2D Convolution and MAC design is the process used to do a variety of image analysis tasks, including picture blurring, softening, feature extraction, and image classification. The major purpose of this study is to create a more efficient MAC control block-based architecture for 2D convolution. This 2D algorithm can be implemented in hardware with a smaller number of modules, multipliers, adders, and control blocks, resulting in substantial hardware savings and reduced LUTs. Simulations were carried out in Verilog and developed and tested using Xilinx Vertex family Field Programmable Gate Arrays (FPGA) technology. In comparison to the conventional 2D convolution implementation, the recommended 2D convolution architectural technique is substantially faster and requires much fewer hardware resources.*

Keywords — *Xilinx Vertex, 2D convolution, MAC, Image processing FPGA, Image enhancement, vertex, and Zynq7000 SOC*

I. INTRODUCTION

In the industry, FPGA is commonly utilized for testing. And also has a wide range of applications such as DSP applications, medical field [1], and close observation like surveillance. Three key applications drive the development of digital image processing technology. The very first function was to promote the better interpretation of input images. As a result, every image that intends to improve the quality of the image is implying that the picture wishes to improve the characteristics of the input image. As a result of which the image will have a better appearance than the original. The autonomous machine is the second key use of digital image processing technology. The efficient storage and transmission of data is the third use. If the image is stored on a computer, It will necessitate the storing of a specific amount of data. In order to limit the quantity of data space necessary to save the photographs otherwise, additional memory will be required to store the photos. Two-dimensional (2D) convolution is used in image and video processing applications to cover a broad area. Picture smoothing, image sharpening, and edge detection all require the use of the 2D convolution block in digital image processing. [2]

MAC design is among the most important operations (Like Edge Detection) [3,4] in image processing applications and algorithms. It could work with various values of kernel-based on user requirement and constraints, allowing for different voice, image processing applications, respectively. Convolution and MAC design is an image retrieval approach extensively employed in image analysis. And recursive sum value, as well as for biomedical devices[5]. As a result, the optimal trade-off between area (reduced LUTs), speed, and specific strength is desired. On the Xilinx Vertex FPGA [6] platform, a 2D convolution framework is proposed to be implemented. When developing image processing techniques with memory components [7], it is crucial for embedded systems to even have low power consumption and high throughput[8]. There are various steps to the implementation process[9]. The target platform is the Vertex FPGA and higher versions. Every pixel value, such as hue, brightness, and so on, has its own meaning[10]. Data is kept in a matrix format. The first step in acquiring a picture out of any input is to transcribe that into image pixels & store it inside this matrix. MAC operations(viz, reconstruction and regression)[11] on those matrices are then executed using a kernel selected by the user.

The remainder of this work is arranged in the following manner. The design methodology for the convolution operator is introduced in Section 2. The proposed architecture and hardware implementation for MAC design as convolution operator are discussed in Section 3. Section 4 of this paper explains the Computational results; the simulation data is discussed in Section 5, tracking systems and performance evaluation parameters are discussed in Section 6, and the article's conclusion is discussed in Section 7.

II. DESIGN METHODOLOGY

Image processing is useful in a wide range of situations[12,13]. For particular, whenever an image is sent or acquired or when an image is compressed, noise signals are easily injected into the original image. As a result, reducing the disturbances in the original image necessitates an additional step[14]. The technique[15,16] of image filters is critical in digital image processing. A kernel, which is a tiny array applied to each neighboring pixel in the image, can be used to define a filter. The kernels centric are usually aligned with the current pixel throughout many applications. The final image is obtained by convolving an image with the kernel. Kernel matrices play a significant part in the convolution idea; different kernel matrices produce distinct image resultants. Blurring, smoothing, contrast enhancement, and other operations based on convolution can all be accomplished depending upon on kernel matrices selected [17,18].

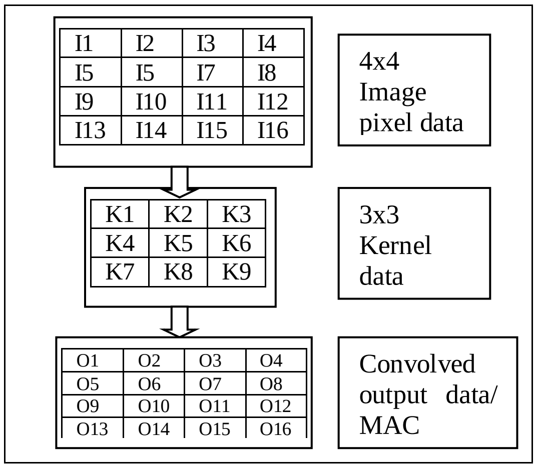


Fig 1: **matrices of the image and the kernel are convolved**.

The image pixel principle stating I(x, y), the kernel convolution measure shows K(x, y), and the convolution resulting for the frame O(x, y). The convolution technique is depicted in Image kernel matrices are regarded 3*3 in Figure 1, as well as image convolution. The resultant matrices convolution is provided by 4*4 matrices. The resultant of the convolution output values[20] is substituted for the pixels' original values, and the resultant array's portions, which are the same size as the kernel, were ultimately averaged. The process will be repeated, with the convolution of the generated values replacing the entire picture in the matrices, pixel values. The convolution pixel value resultant should be 0 whenever the kernel pixel value is outside the matrix. The block is fed with the core pixel values to be analyzed, as well as its neighboring pixels.

Figure 1 depicts the convolution of the picture with kernel matrices, where the picture represents 4x4 matrices, the kernel matrices represent 3x3, and the final picture represents 4x4. The convolution process should be conducted as shown in the equation.

The recursive multiplication and addition operators derive all of the outputs of convolve data. Get the Output convolution resultant (O1 to O16) values based on this. The technique is repeated, other values are discovered. When performing the convolution operation, the mean multiplication and additions rise, implying that the propagation latency increases as well. As a result, create an efficient architecture for 2D convolution.

III. 2D Convolution architecture and MAC hardware Implementation Design

Multiplication and summation will be performed by the model, which is necessary for image processing applications. Essentially, this approach obtains the pixels and kernel values, then activates the multiplier pixels with the kernel and adds them together.

Different kernels, such as identification, edge detection, sharpen, box blur, Gaussian blur 3x3, and so on, must be chosen based on user preferences.

Blur = 1/9 [1,1,1 : 1,1,1 : 1,1,1]

I.e., each pixel is multiplied by a factor of one. We totaled all of a pixel's eight neighbors and divided the result by nine. Alternatively, you might use a smoothing effect or the average of one pixel. Box blur is used as a case study for our MAC hardware implementation.

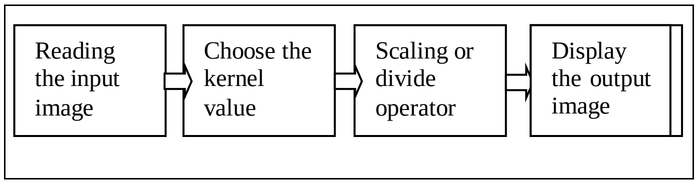


Fig 2: **implementation of MAC hardware in MatLab**

In FPGA, the same mechanism is used. Assuming row-by-row data from the preprocessing unit [REF]. For MAC operation, 24 bits per row and a total of three rows of data are regarded as input pixels data. Finally, 72-bit pixel data is handled as system data at each clock cycle.

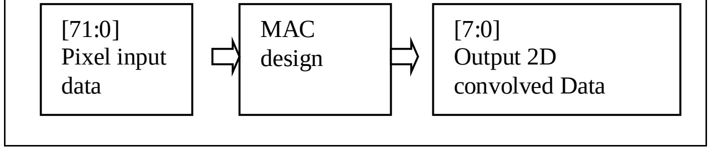


Fig 3: External view of the 2D MAC Implementation Design

An image kernel is a tiny matrix that can be used to perform effects like blurring, sharpening, outlining, and embossing, similar to those seen in Photoshop. They're also employed in a process called 'feature extraction,' which determines the essential parts of an image. In this context, the process is referred to as "convolution" in a broader sense.

To begin, a nine-element kernel is chosen, with each value having an 8-bit wide representation at most. To fit this into memory, a two-dimensional array is created using the following declaration.

```
Reg [7:0] kernel [8:0]
```

Fig 4: **2D array declaration for kernel**

This begins the process of assigning or writing the nine values to kernel memory. Because all of the elements in our case study have the same value, we can use a recursive technique to assign a common value to all of the positions in the 2D array of memory. Considering this as task 1 of the pipeline stage in the proposed design

```
Always@ (posedge (clk))
For (i=0;i<9;i=i+1)
Begin
Kernel[i] =1;
End
```

Fig 5: 2D array declaration for the kernel with values assigned as a case study of "blur" kernel 9 values.

The very next activity is multiplication (task 2 of pipeline stage un suggested architecture), which entails retrieving all nine values from a 2D array memory and multiplying them with the incoming pixel data. Tasks 1 and 2 are seen to be multitasking and executed in pipeline mode.

```
Always@ (posedge (clk))
for (i=0;i<9;i=i+1)
Begin
Multiplied_data<= Kernel[i]* input pixel data [i*+:8];
End
```

Fig 6: multiplication operation as a case study of "blur" kernel 9 values

To generate the output convolved resulting pixels, the next operation is an addition (add them all together).

```
Always@ (posedge (clk))
for (i=0;i<9;i=i+1)
Begin
sum_data<= sum_data+multiplied_data[i];
End
```

Fig 7: **addition operation as a case study of "blur" kernel 9 values**

The matrix is divided with a value of [1/9] at the final stage of MAC design, which is indicated in figure 8

```
Always@ (posedge (clk))
Begin
Convolved_output<= Sum_data/9;
End
```

Fig 8: division operation as a case study of "blur" kernel 9 values.

IV. PERFORMANCE EVALUATION AND ARCHITECTURE ASSESSMENT

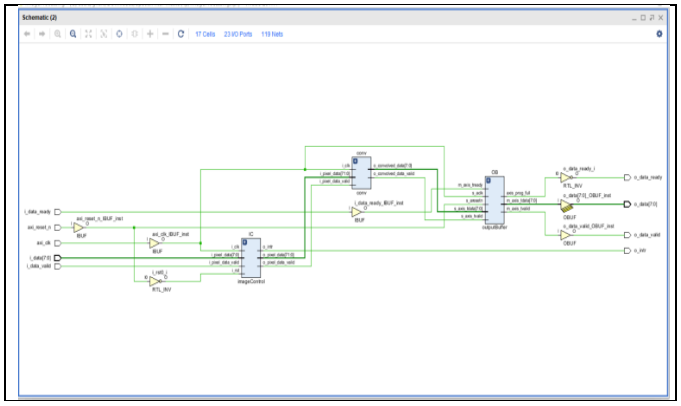


Fig 9: RTL schematic

The RTL schematic, which is the transformation of an HDL description together into a formally digital circuit schematic, or in plain terms, a netlist, is shown in Figure 9.

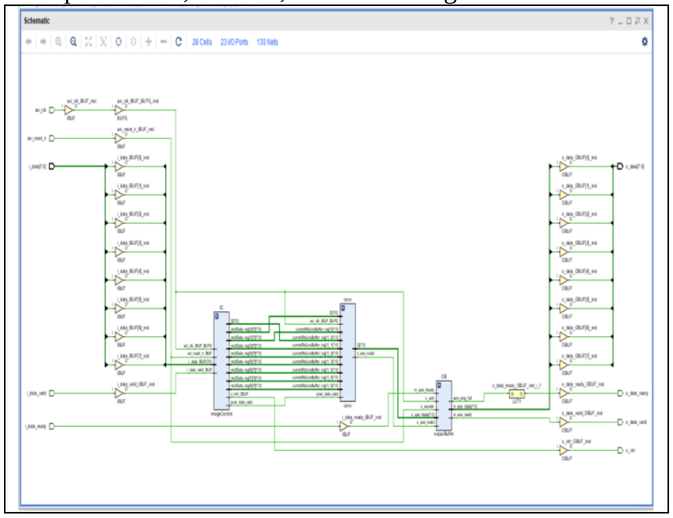
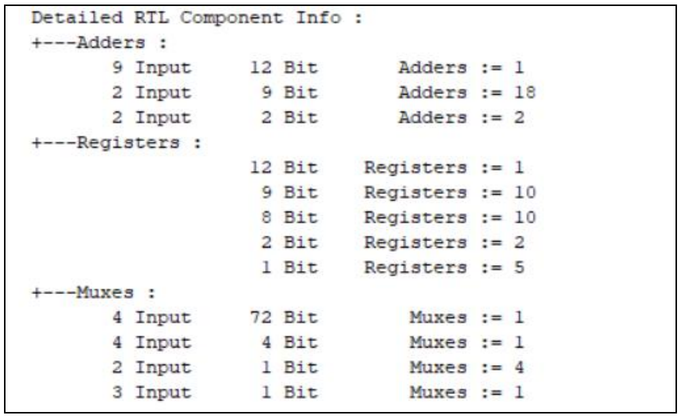


Fig 10: Technologic Schematic analysis in terms of LUT's

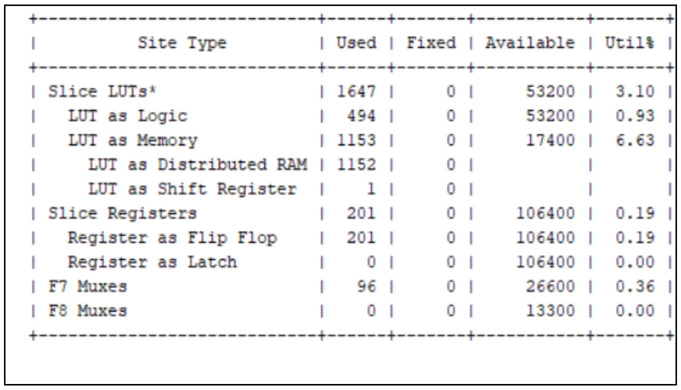
Figure 10 shows a Look-Up Table-based technological schematic; it is a representation of a netlist superimposed over the structural properties of a specific device (Vertex board).



```
Detailed RTL Component Info :
+---Adders :
         9 Input     12 Bit       Adders := 1
         2 Input      9 Bit       Adders := 18
         2 Input      2 Bit       Adders := 2
+---Registers :
                     12 Bit    Registers := 1
                      9 Bit    Registers := 10
                      8 Bit    Registers := 10
                      2 Bit    Registers := 2
                      1 Bit    Registers := 5
+---Muxes :
         4 Input     72 Bit        Muxes := 1
         4 Input      4 Bit        Muxes := 1
         2 Input      1 Bit        Muxes := 4
         3 Input      1 Bit        Muxes := 1
```

Fig 11: detailed RTL component information

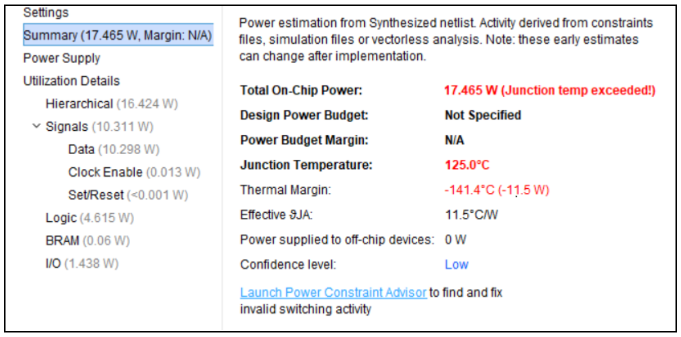
Figure 11 depicts the comprehensive components of a report that were used in the synthesis.



```
+----------------------------+------+-------+-----------+-------+
|         Site Type          | Used | Fixed | Available | Util% |
+----------------------------+------+-------+-----------+-------+
| Slice LUTs*                | 1647 |     0 |     53200 |  3.10 |
|   LUT as Logic             |  494 |     0 |     53200 |  0.93 |
|   LUT as Memory            | 1153 |     0 |     17400 |  6.63 |
|     LUT as Distributed RAM | 1152 |     0 |           |       |
|     LUT as Shift Register  |    1 |     0 |           |       |
| Slice Registers            |  201 |     0 |    106400 |  0.19 |
|   Register as Flip Flop    |  201 |     0 |    106400 |  0.19 |
|   Register as Latch        |    0 |     0 |    106400 |  0.00 |
| F7 Muxes                   |   96 |     0 |     26600 |  0.36 |
| F8 Muxes                   |    0 |     0 |     13300 |  0.00 |
+----------------------------+------+-------+-----------+-------+
```

Fig 12: report on the use of space components inside target device

The area utilization report in terms of inside structural components like logic, registers, and mux which is shown in Figure 12. The primary finding is that selecting the vertex as a target consumes less than 1% of the total available resource on the target device.



Settings
Summary (17.465 W, Margin: N/A)
Power Supply
Utilization Details
Hierarchical (16.424 W)
Signals (10.311 W)
Data (10.298 W)
Clock Enable (0.013 W)
Set/Reset (<0.001 W)
Logic (4.615 W)
BRAM (0.06 W)
I/O (1.438 W)

Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

Total On-Chip Power:	17.465 W (Junction temp exceeded!)
Design Power Budget:	Not Specified
Power Budget Margin:	N/A
Junction Temperature:	125.0°C
Thermal Margin:	-141.4°C (-11.5 W)
Effective ϑJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

Fig 13: Report on Energy Consumption

The power consumption of the capacity in the targeted system is depicted in Figure 13.

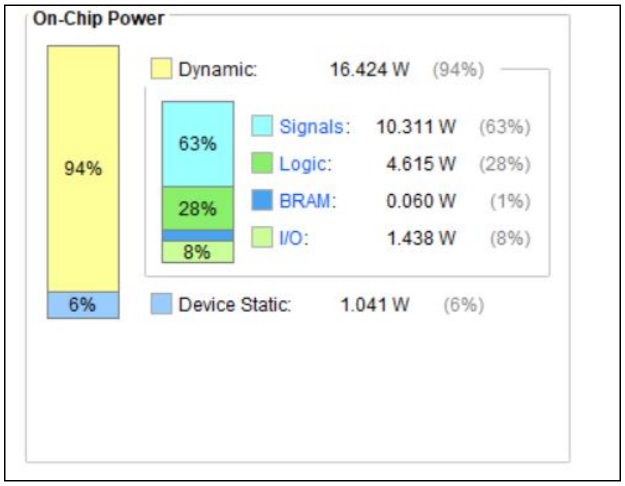


Fig 14: report on on-chip power

Figure 14 shows the complete on-chip power data on dynamic and static power dissipation. The essential point to note is that the power slot is split into two categories: dynamic and static. Where the dynamic power consumption is 16.24W (94%), and the static power consumption is 1.041W (which is 6 percent). Signals and logic are responsible for the device's dynamic power usage. I/O and RAM are blocked.

V. POST IMPLEMENTATION REPORT

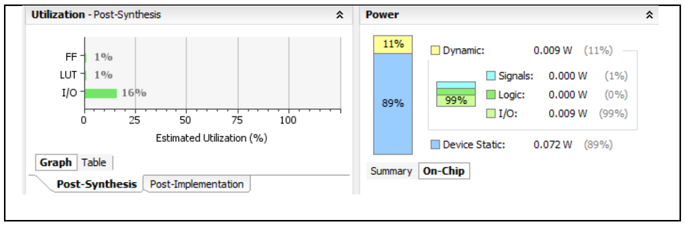
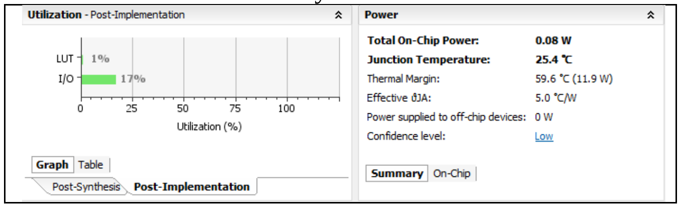


Fig 15: Report on the Synthesis for the target device

Multiple test cases are produced by using test bench, validated, and used the vivado tool, then hardware synthesis is triggered for various testing vectors by using test bench application, and also the architecture is generated and used in the verilog building design. For comparison purposes, a few characteristics such as power, area, and efficiency can be extracted from those in the synthesis results.



Total On-Chip Power:	**0.08 W**
Junction Temperature:	**25.4 ℃**
Thermal Margin:	59.6 ℃ (11.9 W)
Effective ϑJA:	5.0 ℃/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Fig 16: Report on the Results of the Implementation

The reporting system for synthesis and execution is shown in Figures 15 and 16. In terms of the number of LUTs utilized for the algorithms, the reconfigurable internal components are employed in 39 instances; for synchronization mostly with an external clock, yet another flip-flop is used.

VI. CONCLUSIONS

The subject of this research report is the hardware implementation of MAC design with convolution operator in Verilog Platform. The proposed hardware design is represented in Verilog and synthesized on an FPGA device with the target device as a vertex and higher versions of SOC. When compared to conventional architecture, critical path time is lowered more in this device, and it also uses fewer slices. The propagation delay was determined to be 5.030 ns for a picture of an order of 16*16 resolutions. When the image resolution is 720*480 or 600*560, the delay will be longer. The convolution values, as a result, are stored in memory when the convolution procedure is completed. This paper is primarily about a conceptual schema for the MAC implementation with a convolution operator is proposed. With only a few mux, ALU blocks for multiplication, addition, division, and control logic blocks for the generation of control signals in each stage of the pipeline stage, it is a basic and efficient hardware implementation with pipeline techniques for the reconfigurable platform and helps in reducing the critical path delay

ACKNOWLEDGMENT

This task has been completed at PESIT Bangalore South Campus (PESIT-BSC) is an approved and affiliated campus of Visvesvaraya Technological University (VTU) in Belagavi, Karnataka, India.

REFERENCES

[1] Nikhil, R. Bluespec System Verilog: Efficient, correct RTL from high-level specifications. In Proceedings of the Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design (MEMOCODE ’04), San Diego, CA, USA, 23–25 June (2004) 69–70

[2] Richard Shoup. Parameterized Convolution Filtering in a Field Programmable Gate Array Interval. Technical Report, Palo Alto, California .(1993).

[3] Hong Shan Neoh and Asher Hazanchuk, Adaptive Edge Detection for Real-Time Video Processing using FPGAs, GSPx 2004 Conference, (2004).

[4] Wang, J.; Zhong, S.; Yan, L.; Cao, Z. An Embedded System-on-Chip Architecture for Real-time Visual Detection and Matching. IEEE Trans. Circuits Syst. Video Technol., 24 (2014) 525–538.

[5] Mondal, P.; Biswal, P.K.; Banerjee, S. FPGA based accelerated 3D affine transform for real-time image processing applications. Comput. Electr. Eng. (2016).

[6] Kadric, E.; Lakata, D.; Dehon, A. Impact of Parallelism and Memory Architecture on FPGA Communication Energy. ACM Trans. Reconfigurable Technol. Syst., 9 (30) (2016) 1–30, 23

[7] Kaur, I.; Rohilla, L.; Nagpal, A.; Pandey, B.; Sharma, S. Different Configuration of Low-Power Memory Design Using Capacitance Scaling on 28-nm Field-Programmable Gate Array. In System and Architecture; Springer: New York, NY, USA, (2018) 151–161.

[8] Pezzarossa, L.; Kristensen, A.T.; Schoeberl, M.; Sparsø, J. Using dynamic partial reconfiguration of FPGAs in real-Time systems. Micro process. Microsyst., 6 (2018) 198–206

[9] Stephen D.Brown, R.J. Francis, J.Rose, Z.G.Vranesic. Field Programmable Gate Arrays,. Shinichi Hirai, Masakazu Zakouji, Tatsuhiko Tsuboi, Implementing Image Processing Algorithms on FPGA-based Real-time Vision System, Proc. 11th Synthesis and System Integration of Mixed Information Technologies (SASIMI2003), Hiroshima, (1992) 378-385.

[10] Torres-Huitzil, C.; Nuño-Maganda, M.A. Areatime Efficient Implementation of Local Adaptive Image Thresholding in Reconfigurable Hardware. ACM SIGARCH Comput. Arch. News 42 (2014) 33–38

[11] Sungheetha, Akey, and Rajesh Sharma. A Novel CapsNet based Image Reconstruction and Regression Analysis. Journal of Innovative Image Processing (JIIP) 2(03) (2020) 156-164.

[12] Dutta, Sayantan, and Ayan Banerjee. Highly Precise Modified Blue Whale Method Framed by Blending Bat and Local Search Algorithm for the Optimality of Image Fusion Algorithm. Journal of Soft Computing Paradigm (JSCP) 2(04) (2020) 195-208.

[13] P. Lysaght, B. Blodget, J. Mason, J. Young, and B. Bridgford, Invited paper: enhanced architectures, design methodologies, and CAD tools for dynamic reconfiguration of Xilinx FPGAS, in Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '06) 1–6,

[14] Madrid, Spain, August 2006 Dougherty, G., Image analysis in medical imaging: recent advances in selected examples. Biomed. Imaging Interv. J. 6(3) (2010) e32,.

[15] John C. Russ, The Image Processing Handbook, 6th Edition, CRC Press, (2011).

[16] Zainalabedin Navabi, Digital Design and Implementation with Field Programmable Devices, Kluwer Academic Publishers, (2011).

[17] G. N. Chiranjeevi and S. Kulkarni, Pipeline Architecture for N=K*2L Bit Modular ALU: Case Study between Current Generation Computing and Vedic Computing, 2021 6th International Conference for Convergence in Technology (I2CT), (2021) 1-4, doi: 10.1109/I2CT51068.2021.9417917.

[18] B. S. Durgakeri and G. N. Chiranjeevi, Implementing Image Processing Algorithms using Xilinx System Generator with Real-Time Constraints, 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), (2019) 230-234, doi: 10.1109/RTEICT46194.2019.9016962.

[19] C. G. Narasimhamurthy and S. Kulkarni, Fast Architecture for Low-Level Vision and Image Enhancement for Reconfigurable Platform, 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), (2021) 1-4, doi: 10.1109/ICAECT49130.2021.9392425.

[20] Chiranjeevi Gubbi Narasimhamurthy, Dr. Subhash kulkarni, Validation of the FPGA-Based Image Processing Techniques using the efficient tool like Xilinx Device Generators, International Journal of Emerging Trends in Engineering Research, doi.org/10.30534/ijeter/2021/16942021, 9(4) (2021).

[21] Y. Aoyagi and T. Asakura, A study on traffic sign recognition in scene image using genetic algorithms and neural networks, inProc. 22ndIEEE Int. Conf. Ind. Electron., Control Instrum., Taipei, Taiwan, R.O.C, 3 (1996) 1838–1843.

[22] Yiannacouras, P.; Steffan, J.G.; Rose, J. VESPA: Portable, scalable, and flexible FPGA-based vector processors. In Proceedings of the 2008 International Conference on Compilers, Architectures and Synthesis for Embedded Systems, Atlanta, GA, USA, 19–24 October (2008) 61–70.

[23] Neuendorffer, S.; Li, T.; Wang, D. Accelerating OpenCV Applications with Zynq-7000 All Programmable SoC Using Vivado HLS Video Libraries; Technical Report; Xilinx Inc.: San Jose, CA, USA, (2015).

[24] So, H.K.H.; Liu, C. FPGA Overlays. In FPGAs for Software Programmers; Springer: Berlin, Germany, 2016; pp. 285–305.

[25] Kelly, C.; Siddiqui, F.M.; Bardak, B.; Woods, R. Histogram of oriented gradients front end processing: An FPGA based processor approach. In Proceedings of the 2014 IEEE Workshop on Signal Processing Systems (SiPS), Belfast, UK, 20–22 October (2014) 1–6.

[26] Keerti Kulkarni, Dr. P. A. Vijaya Separability Analysis of The Band Combinations For Land Cover Classification of Satellite Images, International Journal of Engineering Trends and Technology 69(8) (2021) 138-144.

[27] Srikanth Khanna, Venkatachalam Chandrasekaran, Fractional Differentiation-based Hybrid Active Contour Model for Noisy Image Segmentation, International Journal of Engineering Trends and Technology 69(8)(2021) 243-259.