International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 10, No. 3, November 2021, pp. 230~236
ISSN: 2089-4864, DOI: 10.11591/ijres.v10.i3.pp230-236

Image processing using a reconfigurable platform: Pre-processing block hardware architecture

Chiranjeevi G. N., Subhash Kulkarni
Department of Electronics and Communication Engineering, PESIT-Bangalore South Campus, Karnataka, India

Article Info

Article history:

Received Apr 26, 2021
Revised May 27, 2021
Accepted Jun 22, 2021

Keywords:

Block memory access
Boundary padding
Kernel architecture
Multi-byte fetching
Pre-processing block
Reconfigurable hardware

ABSTRACT

Real time image processing is a challenging task in which fetching the sub image requires offset memory access apart from core processing needs. This paper aims at overcoming the offset needs for memory addressing in pre-processing blocks. Another feature of this present work is to appending the image data with customized algorithmic reequipments viz duplicating, zero padding. For KxK kernel size, the proposed hardware architecture can be programmed to fetch K pixels in one cycle, reducing the data access time. Results have been compared with software-based processing for KxK spatial filtering. performance indicates significant timing improvement using proposed pre-processing hardware block.

Corresponding Author:

Chiranjeevi G. N.
Department of Electronics and communication Engineering
PESIT-Bangalore South Campus
Affiliated to Visveswaraya Technological University, Belgaum
Bangalore, Karnataka, India
Email: chiranjeevign@pes.edu

1. INTRODUCTION

The majority of the time, image processing algorithms/architecture [1], [2] perform best when given specific types of data as inputs. However, in the vast majority of instances, the input picture fails to meet critical requirements. Prior to the application-specific processing [3], preprocessing takes place. The image storage problem is a significant issue in image processing. Many image file formats have been developed over the years with the aim of representing images in a streamlined and premium manner that can be used on a variety of platforms [4]. According to preprocessing, different images of the same type can have a different scale of signal intensities. The operations that are usually needed prior to the main data operations in IP [5] core are grouped as preprocessing functions. Hardware architecture is used as the initial method or pre-processing block in image processing applications with a higher degree of accuracy in reading pixel values in this study. As a result, these images are processed in such a way that they can be used for operations, reducing data storage access time [6]-[8]. Pre-processing often entails the elimination of unnecessary or irrelevant regions, as well as the enhancement of contrast and service features like zero padding.

Digital images containing a finite set of image components, usually known as pixels, are used to display two-dimensional images. Digital image processing allows for the retrieval, delivery, and representation of image data in a human-readable format [9]. In the implementation areas of image processing, a variety of techniques are applied to the chosen image data set for image pre-processing. This work has proposed a technique to improve memory read and write operations, which are needed by IP cores. The chosen pre-processing hardware block has proven to be the most effective method for reading pixel

values in group or concatenated form with kernel-defined size. FPGAs have become the fully independent implementation framework for a number of computer vision applications due to advancements in FPGA technology [10].

This research paper is organized as follows. Section 2 discusses about Pre processing Blocks in Image Processing, Section 3 explains clearly about Pre processing block Hardware Architecture used for this research work. Section 4 shows Pre-procesing block memory. section 5 shows Experimental evaluation over reconfigurable platform carried throughout the preprocessing process. Finally, section 6 concludes the research work with its findings.

2. PRE PROCESSING BLOCKS IN IMAGE PROCESSING

Preparation of data is the primary goal of most preprocessing systems. So that following blocks can make optimum use of them. The main aim of pre-processing is to enhance the image's quality [11], [12] so that we can properly analyze it. We can remove unwanted distortions [13] and improve some features (reconstruction and regression) [14], [15] that are essential for the application we're working on by preprocessing. Those characteristics can differ depending on the application so that other types of algorithms [16] can use them effectively (general image processing, image enhancement, or image analysis).

The enormous amount of information needed to depict images is one of their most distinguishing features (architecture using Vedic computing) [17]. Even a gray-scale image with a moderate resolution, such as 512 by 512, requires 512 *512*8= 2*106 bits to represent. As a result, in order to store and transmit digital images (using XSG blocks) [18], some type of image compression and image edge detection [19] or the use of pre-processing hardware is required.

Within FPGA pre processing sub-systems (DSP modules) [20], algorithms evolve from standard software-suitable representations [21] to more hardware-friendly ones, which can completely exploit data parallelism [22], [23] across application-specific hardware architectures(signal and video processing architecture) [24], which are often significantly different from the conventional Von Neumann model, such as dataflow [25].

The aim of the architecture is to prepare data and make image processing activities easier [26], [27]. The general structure of the strategy suggested in this study is depicted in Figure 1. The image preliminary pre-processing method and the image screening algorithm are the foundations of this article.

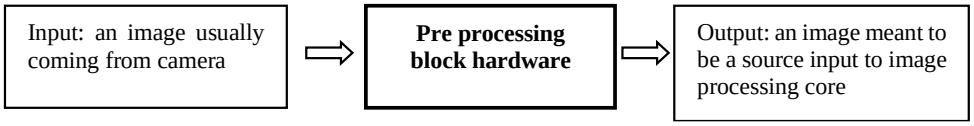

Figure 1. Architecture of the work

3. PRE PROCESSING BLOCK MEMORY

Design of preprocessing block memory, two constraints need to be taken care from the user end. The first one is writing one pixel value which is of one byte into the memory at each clock cycle of the target device. The second one is the size of the input image (example: 512*512). The main target we are considering in this block memory is flexibility in modifying the size of kernel during read and writes operations. The ability to choose the kernel size during read operations is seen as a benefit over the inbuilt IP core model. A detail the IP core can access the data in terms of 2^{powers} bit [*i.e.* 2, 4,8,16,32,64]data during read operation, but in the proposed design it can access based on the kernel requirements and not reserved to any specific values. A general 8bit pixel value * kernel size is the data accessed during read operation.

3.1. Write mode

As indicated by the address pointer, one pixel of data is written into the memory during this operation with regard to the clock cycle.

3.2. Read mode

To get around the first-in first-out (FIFO) paradigm, the proposed hardware architecture activates read operations ‘N' times depending on the user's needs. The read operation iteration is activated based on the kernel size. For example, if the kernel size is 3 by 3, the values from three adjacent positions are read. This will decide whether enough data is available for IP core architecture. The user must specify the memory

hardware location from which we will read. That is, the read output is extracted as a concatenation of three-pixel data values highlighted by the read pointer from the pre-processing block hardware. Finally, data was accessed simultaneously from all three locations.

3.3. Additional user choice

Around the edges of an image, image padding adds new pixels. When advanced filtering methods are used, the border provides space for annotations or serves as a boundary. As user preference inputs, the three separate case studies are used.

3.4. Duplicate mode

Two rows and two columns have been added. The first row's and last row's pixel values are copied for the new row that comes before the first row and after the last row, respectively which is indicated in Figure 2. Similarly, the values of the first and last columns of the original image are copied to the new columns. Leading to the cascading of two additional rows and two additional columns, the image would be 514 by 514 after duplication.

3.5. Zero padding

Additional rows are added, with all pixel values set to zero which is indicated in Figure 3. Similarly, columns with a pixel value of zero are introduced.

3.6. Non duplicate mode

For IP core operations, function with existing/ignore the boundaries in this case model. Ignore the value of the edge pixel and compute for those pixels that have all of their neighbors which are highlighted in Figure 4.

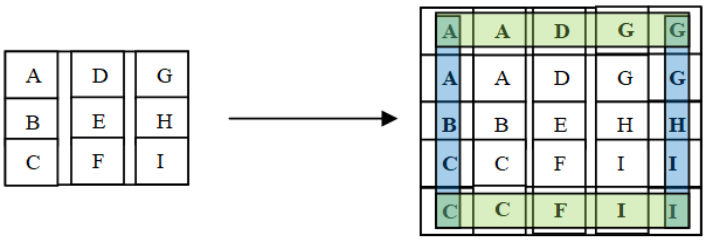

Figure 2. Duplicate mode: Original image v/s with padding extra row and col

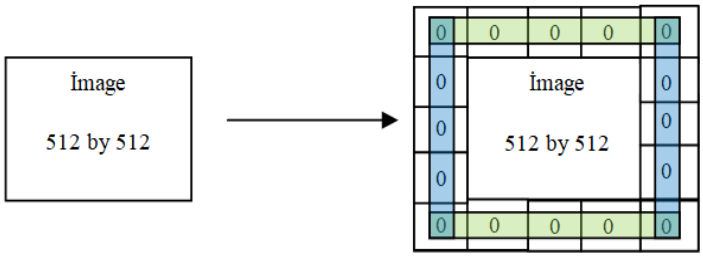

Figure 3. Zero padding modes: Original image v/s with padding extra row and col filled with zero

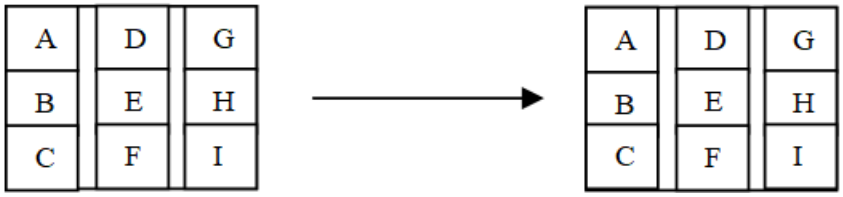

Figure 4. Non - Duplicate mode: Remains same as original image (no additional row and col)

4. EXPERIMENTAL EVALUATION OVER RECONFIGURABLE PLATFORM

On an FPGA, hardware design strategies such as parallelism and pipelining are feasible, which are not possible in dedicated DSP modules. The use of reconfigurable hardware to implement image processing algorithms reduces time-to-market costs, allows for rapid prototyping of complex algorithms, and simplifies debugging and verification. As a result, FPGAs are an excellent alternative for real-time image processing algorithms. Verilog coding is used in this research paper to build the preprocessing hardware architecture. The verification of functionality is carried out on reconfigurable hardware with the help of design flow diagram which is shown in Figure 5.

This is the external view that is included in the 5 by 5 kernel shown in Figure 6. The output is indicated as 40 bits in this case. In one clock cycle, that's 5 times the 8-bit pixel value which depicts in the above figure. It is having access to all the external input and output pins.

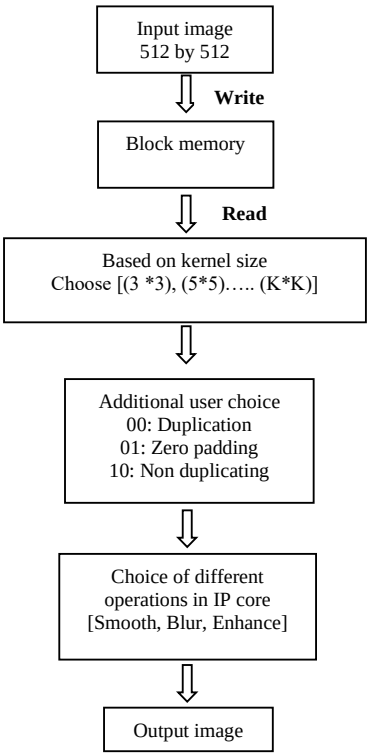

Figure 5. Design flow for preprocessing architecture of the work

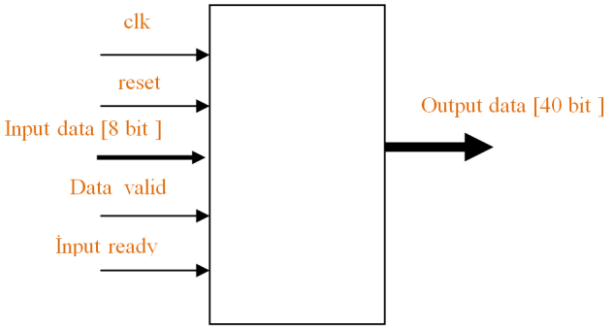

Figure 6. External view for preprocessing architecture of the work

Figure 7 depicts, Register-transfer level (RTL) is a design abstraction for modeling a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers and the logical operations performed on those signals in digital circuit design.

The findings from Figure 8, demonstrate the effectiveness of FPGA-based reconfigurable systems in image processing applications, demonstrating a significant speedup over software versions as operations become more complex. Then, using our scheme, we will be able to relax the number of operations in a real-time application, enabling us to incorporate more complex algorithms.

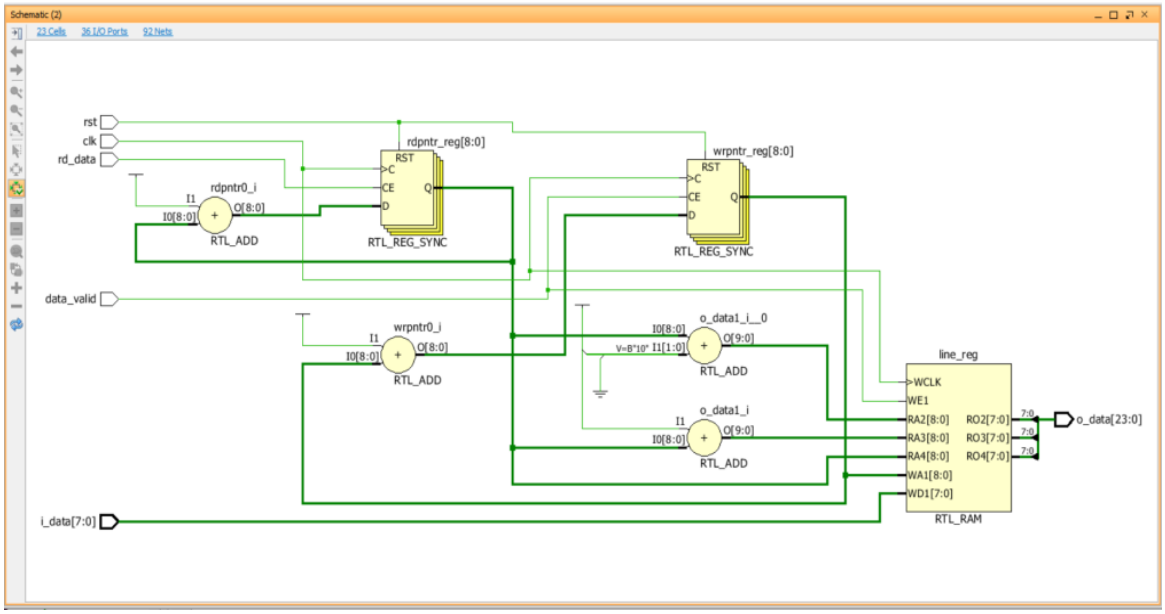

Figure 7. RTL design for preprocessing architecture of the work

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	399	0	20800	1.92
LUT as Logic	111	0	20800	0.53
LUT as Memory	288	0	9600	3.00
LUT as Distributed RAM	288	0		
LUT as Shift Register	0	0		
Slice Registers	18	0	41600	0.04
Register as Flip Flop	18	0	41600	0.04
Register as Latch	0	0	41600	0.00
F7 Muxes	8	0	16300	0.05
F8 Muxes	0	0	8150	0.00

Figure 8. Utilization report for preprocessing architecture of the work

5. RESULTS AND DISCUSSION

Convolution is a general-purpose image filter effect that determines the value of the center pixel by adding the weighted values of all its neighbors. The product of convolution of 512*512 image matrix with kernel of [(3 *3), (5*5) … (K*K)] is a new modified filtered image. Overlap the kernel on top of the image, calculate the product of the mutually overlapping pixels and their sum in each case, and the result will be the value of the output pixel at that particular location

Figure 9 depicts input pixels being written into memory and accessed through a read operation; the above results are for a 3 by 3 matrix kernel. In other words, it's turning on the consecutive read operation three times in a row.

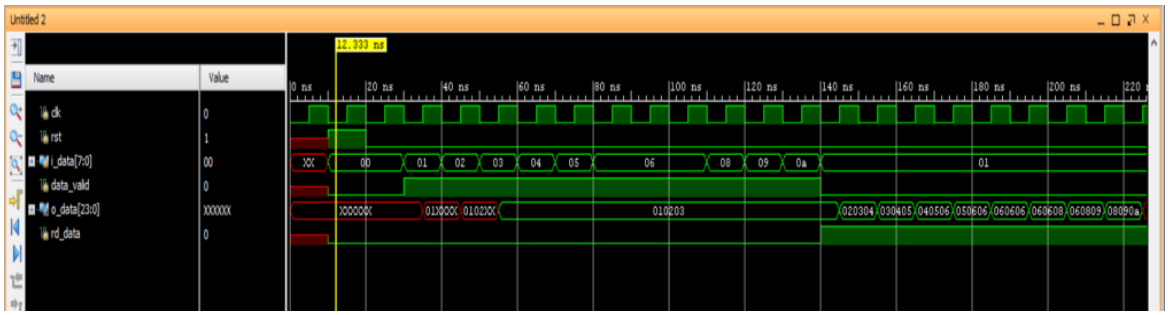

Figure 9. Simulation results for preprocessing architecture of the work

6. CONCLUSION

The pre-processing technique used in this study aids in the access of data required by the IP core and the enhancement of image quality by the use of various image processing techniques. The findings are analyzed and checked with a standard reconfigurable platform (Zynq board), and the consistency in terms of hardware utilization is also evaluated (area) The focus of this research is to concentrate on selecting the appropriate memory operations, such as multiple read and write techniques, as well as taking into account the user's choice of kernel size as a primary input. Not only does the pre-processing technique minimize memory access time. The information gathered as a result of this process could be useful in furthering this study. FPGAs as deployment reconfigurable platforms for high-level image processing applications include efficient mapping of high-level descriptions of image frames to low-level memory systems.

ACKNOWLEDGEMENTS

This work is being done at the PESIT Bangalore South Campus's Research Center of Electronics and Communications, which is recognized and affiliated by Visvesvaraya Technological University Belagavi, Karnataka, India.

REFERENCES

[1] T. F. Smith and M.S.Waterman, "Identification of common molecular subsequence's," *Journal of Molecular Biology (JMB)*, vol. 147, no. 1, pp. 195–197, 1981, doi: 10.1016/0022-2836(81)90087-5.

[2] R. Nikhil, "Bluespec system verilog: Efficient, correct RTL from high level specifications," *Proceedings. Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design*, 2004. MEMOCODE '04., 2004, pp. 69-70, doi: 10.1109/MEMCOD.2004.1459818.

[3] R. Shoup, "Parameterized convolution filtering in a field programmable gate array interval," *Technical Report*, Palo Alto, California, 1993.

[4] H. S. Neoh and A. Hazanchuk, "Adaptive edge detection for real-time video processing using FPGAs," *Global Signal Processing*, vol. 7, no. 3, pp. 2-3, 2004

[5] J. Wang, S. Zhong, L. Yan and Z. Cao, "An embedded system-on-chip architecture for real-time visual detection and matching," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 3, pp. 525-538, March 2014, doi: 10.1109/TCSVT.2013.2280040.

[6] P. Mondal, P. K. Biswal and S. Banerjee, "FPGA based accelerated 3D affine transform for real-time image processing applications," *Computers & Electrical Engineering*, vol. 49, pp. 69-83, 2016, doi: 10.1016/j.compeleceng.2015.04.017.

[7] E. Kadric, D. Lakata and A. Dehon, "Impact of parallelism and memory architecture on FPGA communication energy," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 9, no. 4, pp. 1-23, 2016, doi: 10.1145/2857057.

[8] I. Kaur, L. Rohilla, A. Nagpal, B. Pandey, and S. Sharma, "Different configuration of low-power memory design using capacitance scaling on 28-nm field-programmable gate array," in *System and Architecture*, S. K. Muttoo Springer: New York, 2018, pp. 151–161.

[9] L. Pezzarossa, A. T. Kristensen, M. Schoeberl and J. Sparsø, "Using dynamic partial reconfiguration of FPGAs in real-time systems," *Microprocessors and Microsystems*, vol. 61, 198-206, 2018, doi: 10.1016/j.micpro.2018.05.017.

[10] S. D. Brown, R. J. Francis, J. Rose, and Z. G. Vranesic, *Field Programmable Gate Arrays*, vol. 180, Springer Science & Business Media, 1992.

[11] S. Hirai, M. Zakouji, T. Tsuboi, "Implementing image processing algorithms on FPGA-based realtime vision system," *Proc. 11th Synthesis and System Integration of Mixed Information Technologies (SASIMI 2003)*, Hiroshima, Apr. 2003, pp. 378-385.

[12] C. Torres-Huitzil and M. A. Nuño-Maganda, "Area time efficient implementation of local adaptive image thresholding in reconfigurable hardware," *ACM SIGARCH Computer Architecture News*, vol. 42, no. 4, pp. 33–38, 2014, doi: 10.1145/2693714.2693721.
[13] John C. Russ, *The image processing handbook*, 6th ed, CRC Press, 2011
[14] A. Sungheetha, R. Sharma R., "A novel CapsNet based image reconstruction and regression analysis," *Journal of Innovative Image Processing (JIIP)*, vol. 2, no. 03, 156-164, 2020, doi: 10.36548/jiip.2020.3.006.
[15] Zainalabedin Navabi, *Digital design and implementation with field programmable devices*, USA: Springer, 2011.
[16] 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," *2006 International Conference on Field Programmable Logic and Applications*, 2006, pp. 1-6, doi: 10.1109/FPL.2006.311188.
[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, pp. 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, pp. 230-234, doi: 10.1109/RTEICT46194.2019.9016962.
[19] S. Ravi, B. A. Rahim, F. shaik, "FPGA based design and implementation of image edge detection using Xilinx system generator," *International Journal of Engineering Trends and Technology (IJETT)*, vol. 4, no. 10, pp. 4657-4660, Oct 2013.
[20] N. P. Raut and A. V. Gokhale, "FPGA implementation for image processing algorithms using Xilinx system generator," *IOSR Journal of VLSI and SignalProcessing*, vol. 2, no. 4, pp.26-36, 2013, doi: 10.9790/4200-0242636.
[21] V. C. Deepthi and P. S. Pradad, "Medical image fusion using Xilinx system generator in FPGA," *International Journal of VLSI System Design and Communication Systems*, vol. 4, no. 10, pp. 0990-0993, 2016.
[22] C. G. Narasimhamurthy and S. 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*, vol. 9, no. 4, pp. 431-434, 2021, doi: 10.30534/ijeter/2021/16942021.
[23] C. Li, R. Huang, Z. Ding, J. C. Gatenby, D. N. Metaxas and J. C. Gore, "A level set method for image segmentation in the presence of intensity inhomogeneities with application to MRI," in *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 2007-2016, Jul. 2011, doi: 10.1109/TIP.2011.2146190.
[24] A. Bose and K. Mali, "Fuzzy-based artifcial bee colony optimization for gray image segmentation," *Signal, Image and Video Processing*, vol. 10, pp. 1089-1096, 2016, doi: 10.1007/s11760-016-0863-z.
[25] Q. Liu, M. Jiang, P. Bai, and G. Yang, "A novel level set model with automated initialization and controlling parameters for medical image segmentation," *Computerized Medical Imaging Graphics*, vol. 48, pp. 21-29, 2016, doi: 10.1016/j.compmedimag.2015.12.005.
[26] M. S. Farid, M. Lucenteforte and M. Grangetto, "DOST: A distributed object segmentation tool," *Multimedia Tools and Applications volume*, vol. 77, pp. 20839–20862, 2018, doi: 10.1007/s11042-017-5546-4.
[27] M. N. Qureshi and M. V. Ahamad, "An improved method for image segmentation using K-means clustering with neutrosophic logic," *Procedia Computer Science*, vol. 132, pp. 534–540, 2018, doi: 10.1016/j.procs.2018.05.006.

BIOGRAPHIES OF AUTHORS

Chiranjeevi G. N received his Bachelor in Electronics and Communication Engineering from Visvesvaraya Technological university, Belgaum, Karnataka, India, in 2010 and M. Tech in VLSI Design from National Institute of Technology Karnataka (NITK Surathkal), Karnataka, India in 2012. Currently working as Assistant Professor at PES University EC Campus (PESIT Bangalore South Campus); Bangalore, Karnataka, India. Areas of interest are in the field of VLSI, FPGA and Image processing applications. Currently the research interest in low power VLSI, FPGA and Medical Image processing.

Subhash Kulkarni obtained PhD degree in 2002 from E&ECE Department, Indian Institute of Technology, Kharagpur, India. He completed his Masters' degree from CEDT, Indian Institute of Science, Bangalore in 1995 with Electronic Design and Technology specialization. He completed B.E in ECE from PDA College of Engineering, Gulbarga, Karnataka. His research interests include Image Processing, Control Systems, and High-Speed Architectures using Vedic Maths. He has been into teaching in Electronics and Communication Engineering since 1989. He is presently working as Principal and Professor in ECE at PESIT Bangalore South Campus, Bangalore. He has guided 9 PhD Scholars till date and has published 120 Research Articles in Journals and Conferences.