Digital Signal Processing (DSP) Platforms for Embedded IoT Applications

Nikola Rendevski, Darko Pajkovski, Zoran Kotevski, Ilija Hristoski, Ramona Markoska
Faculty of Information and Communication Technologies
"St. Kliment Ohridski University" – Bitola
Republic of Macedonia
{nikola.rendevski, darko.pajkovski, zoran.kotevski, ramona.markoska@fikt.edu.mk}
{ilija.hristoski@uklo.edu.mk}

ABSTRACT: *In this work, we present a digital design of compact Field Programmable Gate Array (FPGA) - based architecture for real-time edge detection. At this point, we are focused on pure hardware realizations of filtering and edge-detection algorithms as widely used techniques for various industrial and entertainment applications. The advantage of FPGAs as digital signal processing (DSP) platforms for real-time video and image processing lays mainly in their reconfigurable and re-usable structure, capable to efficiently exploit spatial and temporal parallelism. Based on the results of the digital FPGA synthesis of pipelined Sobel edge detection architecture, we discuss on the potential and challenges towards design of low-complex pure hardware-based video processing devices for embedded IoT applications.*

Keywords: FPGA, Edge Detection, Sobel Algorithm, IoT

Received: 3 March 2021, Revised 12 July 2021, Accepted 19 July 2021

DOI: 10.6025/pms/2021/10/2/55-61

1. Introduction

Image Processing (IMP) and Video Processing (VP) have been used for a long time in various industrial, medical, security and entertainment systems. In example, within the industrial environments, such techniques are still used to control the product quality and inspection procedures by replacing manual activities traditionally conducted by humans. In any case, the control, inspection and detection procedures are based on off-line IMP, or real-time VP processing, mainly by implementation of preprocessing algorithms such as thresholding and filtering or processing algorithms such as pattern matching and edge detection. Considering the Industry 4.0 and Tactile Internet paradigms, efficient IMP and VP capable to provide responses towards meeting the requirement for sub-millisecond latencies are one of the key aspects for entirely remote operated facilities and smart factories. Furthermore, nowadays we are all witnesses of the mass proliferation of image capturing devices (different forms of cameras) in our daily lives: in computers, mobile devices, cars etc. In such developmental trends, with the advances of the microelectronics, cameras are becoming drastically smaller, conduct faster image acquisition and produce high-definition output as expected by the modern application scenarios which require strict QoS levels. Moreover, such sensitive QoS

inevitably requires more and more processing power which directly affect the complexity, cost and energy conservation capabilities of the systems. The future views of the fully networked society, based on technologies such Internet of Things (IoT), Cyber Physical Systems (CPS) or Wireless Sensor Networks (WSN), embedded as an inevitable part of the Fifth Generation Network (5G) framework, are expected to provide ubiquitous processing and multi-interface communication with limited DSP resources, mainly constrained by the size and energy consumption, towards justifying the vision for massive deployment on a batteryoperated power sources. Nowadays, energy-efficient wireless sensor nodes falling within the concept of WSN, are mainly driven by low-power RISC processors and microcontrollers which are capable to provide readings from simple conventional sensors, low-sampling A/D conversions, simple processing and short-range wireless transmission. Expectations from such reduced functionality devices (RFDs) for any type of image or video processing are impossible, except some ultralow- resolution imaging for realization of simple detection functionality. From the other side, fully functional devices (FFDs) pose processing capabilities enabled by certain type of embedded processor with higher processing power, capable to realize IMP and real-time VP, but still with limited performance. Such devices in the WSN model are equipped with additional power sources and energy harvesting mechanisms to achieve longer operation. Considering the above facts, the WSN FFD devices concept could be treated as model of a typical IoT embedded device. Various IoT development platforms exist such Raspberry Pi, Orange Pi and Arduinos, equipped with ARM 32- and 64-bit embedded processing cores. However, efficient higher-definition IMPs and VPs are still heavy load for those miniature computers as the dynamic imaging sensor fusion cause serious bottlenecks mainly because of speed-limited execution of software-based algorithms and memory access. This fact could lead to a conclusion that processor-centric systems based on general purpose processors (GPP) designed only on the conventional Von-Neumann (so called program-stored computation) architectures are not standalone candidates for the processing unit in next-generation IoT embedded devices. The recent advances in FPGA manufacturing processes, as recently declared by Intel (Altera), is moving to feasible 10 nm based on FinFET technology, which makes the future FPGAs speeds comparable to ASIC processor chips. Such trends truly support the System-on-Chip (SoC) integrated digital hardware architectures consisted of so-called Hard Processor Systems (HPS), interconnection networks (buses) and reconfigurable FPGA, not only on the same board, but also on the same VLSI chip. Considering the fact that 65 nm FPGAs nowadays are widely available with reasonable sizes, price and performance suitable for the vast IMP and VP IoT applications, positions them as valuable alternatives for processing and co-processing subsystems of a typical IoT device.

Considering the above-mentioned available technologies, a question naturally rises: Which IMP and VP functions and routines are more suitable for certain processing architecture (FPGA, HPS-FPGA, soft processor etc.)? In this paper, based on the practical experiences from the physical FPGA synthesis of digital logic architecture for performing Sobel Edge Detection of real-time video and images, we conduct analyses with goal to propose processing scheme suitable for design of IoT devices with IMP and VP capabilities. The rest of the paper is organized as follows: In Chapter 2 we present the introduction to low-complex edge detection (ED) algorithms. Chapter 3 is dedicated to the experimental testbed. The results from the FPGA synthesis and complexity analyses and the conclusions are presented in Chapter 4.

2. Edge Detection Algorithms

The understanding of edge detection (ED) techniques and algorithms and their mathematical background is crucial to analyze the complexity of appropriate digital processing architecture and FPGA synthesis, which are the main focus of this work towards realization of low-complex IoT devices. In the following we present brief explanation of what ED is, its purpose and types. Edge detection is one of the fundamental tools in image and video processing, machine and computer vision - particularly in all areas where feature detection and feature extraction are crucial. The main purpose of edge detection is capturing important events and variations in different properties of the physical objects which are observed, such changes in the material structure, surface orientation, and variations in scene illumination. Sometimes, few lines in an illustration are often sufficient to unambiguously describe an object or a scene. Numerous IoT usage models ranging from industrial applications to entertainment require such analyses, in both, still images and video. From the design and processing point of view, the edge detection of complex graphic and multimedia content is not a trivial task.

ED, in general, is a mathematical method for identifying points within an image at which there is a significant change in the color, brightness or discontinuity along a particular orientation. Such points are typically organized into a set of curved line segments which enable identification of edges. The stronger the local intensity change, the higher is the evidence for an existence of edge at that position.

Although certain literature has considered the detection of ideal step edges, the edges obtained from natural images are usually not that simple. Instead, they are normally affected by one or several of the following effects: focal blur, penumbral blur caused by shadows of various light sources and shadings of a smooth objects. In mathematics, the amount of change with respect to spatial distance is known as the first derivative of a function, and the simplest detectors are based on analyses of this function.

Let us assume a simple image with white region surrounded by a dark (black) background. The intensity profile along one image line is one-dimensional function *f(x)* and the first derivative of the function, $\boldsymbol{f'(x) = \frac{df}{dx}(x)}$, results in a positive peak where the intensity increases, and a negative where the function decays, as shown on the Figure 1.

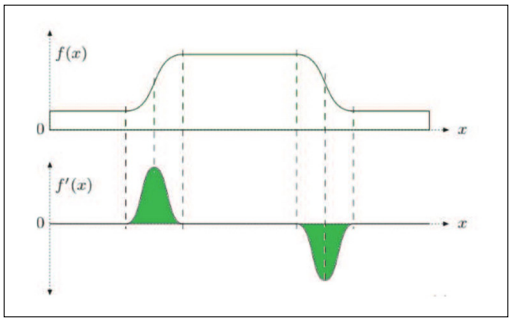

Figure 1. The intensity profile along one image line, one-dimensional function *f(x),* and the first derivative of the function

However, the line profile of real image is a discrete function *f(u)* and suitable estimation method is needed as the first derivative is undefined in the discrete domain (Figure 2.).

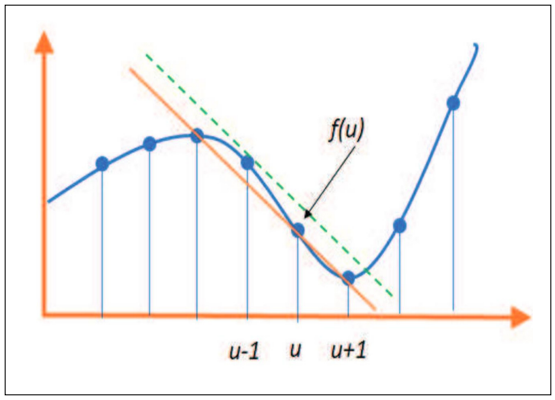

Figure 2. Line profile of real image - discrete function *f(u)* and estimation method

A simple method for roughly approximating the slope of the tangent for a discrete function *f(u)* at position *u* is to fit a straight line through the neighboring function values *f*(*u*–1) and *f*(*u*+1). The same applies for both horizontal and vertical direction to estimate the first derivative Eq.1. [1].

$$\frac{df}{du}(u) \approx \frac{f(u+1)-f(u-1)}{(u+1)-(u-1)} = \frac{f(u+1)-f(u-1)}{2} \tag{1}$$

A derivative of a multidimensional function at one of its coordinate axes is called a partial derivative. For example: $\frac{\partial I}{\partial u}(u,v), \frac{\partial I}{\partial v}(u,v)$ are partial derivatives of an image function *I(u,v)* along the *u* and *v* axes. The function $\nabla I(u,v)$ is so called gradient or gradient vector of the image function *I* at a position *(u,v)*: $\nabla I(u,v) = [\frac{\partial I}{\partial u}(u,v)\ \frac{\partial I}{\partial v}(u,v)]^T$.

The magnitude of the gradient (Equation 2.):

$$|\nabla \boldsymbol{I}(\boldsymbol{u},\boldsymbol{v})| = \sqrt{\left(\frac{\partial \boldsymbol{I}}{\partial \boldsymbol{u}}(\boldsymbol{u},\boldsymbol{v})\right)^2 + \left(\frac{\partial \boldsymbol{I}}{\partial \boldsymbol{v}}(\boldsymbol{u},\boldsymbol{v})\right)^2} \tag{2}$$

is not dependent from the image rotation and orientation of the underlying image structures. This property is important for isotropic localization of edges, and thus $|\nabla I|$ is the basis of many simple edge detection methods. The approximation of the first horizontal derivatives can be easily implemented by a linear filter (see with the coefficient matrix $\boldsymbol{L_x} = [-\frac{1}{2} \quad \boldsymbol{0} \quad \frac{1}{2}]$ and $\boldsymbol{L_y} = [-\frac{1}{2} \quad \boldsymbol{0} \quad \frac{1}{2}]^{\boldsymbol{T}}$, where the center pixel $I(u, v)$ is weighted with the zero coefficient or with other words - ignored.

The Sobel operator performs a 2D spatial gradient measurement of the image detecting regions of high spatial frequency that correspond to edges. It finds the approximate absolute gradient magnitude at each point of an input grayscale image. Corresponding to the application of the following filter masks to the image data, the Sobel operator is based on the following filters:

$$\boldsymbol{L_x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \textbf{ and } \boldsymbol{L_y} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{3}$$

The estimates for the local gradient components are obtained from the filter results by appropriate scaling. For the Sobel operator [1]:

$$|\nabla \boldsymbol{I}(\boldsymbol{u},\boldsymbol{v})| \approx \frac{\boldsymbol{1}}{\boldsymbol{8}} \begin{bmatrix} \boldsymbol{I} * \boldsymbol{L_x}(\boldsymbol{u},\boldsymbol{v}) \\ \boldsymbol{I} * \boldsymbol{L_x}(\boldsymbol{u},\boldsymbol{v}) \end{bmatrix} \tag{4}$$

It is worth to note that edge detection methods based on second derivatives exists, and the main idea behind this approach is that edges can be found at zero-crossings of the second derivatives within the image function. Since the second derivatives amplify the noise, methods for presmoothing must be applied using low-pass filters which will certainly affect the processing architecture complexity. Very used example of second derivative ED is Laplacian-of-Gaussian" (LoG) operator.

Considering the fact that in this work we are focused on analyses of low-complex ED solutions, digital FPGA synthesis will be conducted using first derivative method using Sobel operator.

3. Experimental FPGA-camera Testbed

In this paper, we synthesize digital architecture for processing Sobel Algorithm on video source using Altera DE- 1 SoC FPGA board. The capturing device is simple low complex Omni Vision OV7670 camera module. The OV7670 is a low voltage CMOS image sensor that provides full functionality of a single-chip VGA camera and image processor in a small footprint package. The OV7670 provides full-frame, sub-sampled or windowed 8-bit images in a wide range of formats, controlled through the Serial Camera Control Bus (SCCB) interface. The module has an image array of 656 x 488 pixels for a total of 320,128 pixels, of which 640 x 480 pixels are active (307,200 pixel). After the Analog Processing block, the raw signal is fed to a 10-bit analog-to-digital (A/D) converter shared by G and BR channels (see Ref. [2]). This A/D converter operates at speeds up to 12 MHz and is fully synchronous to the pixel rate (actual conversion rate is related to the frame rate). For the Altera DE-1 FPGA Development board, interested reader could refer to [5] for full board specification. The crucial specifications of the board, important for this demonstration, are presented in Table 1.

There are two possible approaches towards FPGA implementations of the digital architecture for processing Sobel ED algorithm. The first approach is by implementing a processing clock that is 4 times the maximum pixel clock. In this implementation, the FPGA will use up to two clock cycles to process a single pixel. However, such approach is not useful for Sobel ED, but for some other convolution operations that use different kernel coefficient which requires multiplication instead of shifting. This implementation is suitable for low resolutions like VGA at up to 60 fps (24MHz pixel clock and 100MHz processing clock). Second option is to design fully pipelined architecture, enabling the pixel clock serving as the processing clock. Such implementation naturally uses significantly more resources, but it is very frequency-efficient (pixel

FPGA Chip	Cyclone V (5CSEMA5F31C6)
Memory Controllers	2 Hard Memory Controllers
Embedded memory	4,450 Kbits
FPGA Memory	64MB (32Mx16) SDRAM on FPGA
Display Port	24-bit VGA DAC
Video Input	TV Decoder (NTSC/PAL/SECAM)
A/D	sample rate: 500 KSPS, Channel number: 8, Resolution: 12 bits, Analog input range: 0 ~ 4.096 V
Power	12V DC input
Clock	Four 50MHz clock sources from the clock generator

Table 1. Altera De-1 SoC Specifications

clock can be higher than 130MHz) and is capable to process HD images.

The Sobel filter, is nothing else, but a 2D spatial high-pass FIR (Finite Impulse Response). FIR filters process the output based on the input history, opposed to an Infinite Impulse Response Filter (IIR) that computes the output based on input history and output history. Considering the Sobel 3x3 filter mask, to generate a single pixel in the output image, access to 9 pixels in the input image is required, then multiply them by 9 values and adding it to the partial result. As 6 out of all 9 components are non-zero (Eq.3.), only 6 multiplications and 5 addition operations are processed. The procedure of multiplication of the kernel values with values from the input to generate a single output is pure convolution operation.

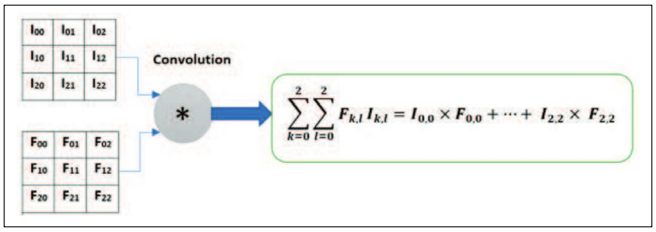

Figure 3. Convolution of the kernel (F) and input image pixels (I)

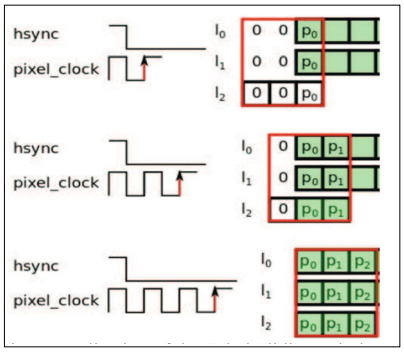

Figure 4. Realization of the Sobel Sliding Window

To develop a Sobel filter, we practically synthesize two main modules, fully described in VHDL: a 3x3 filter mask module and the Sobel arithmetic module. The used camera does not provide output in 2D stream of pixels, but 1D stream with a system signal indicating when to increment the second dimension (*hsync*). The minimum amount of data to store is two lines of the image and 3 pixels, to realize a 3x3 sliding window in the image [3] (Figure 4).

4. Synthesis Results and Conclusion

We realized synthesis of FPGA architecture for Sobel ED on real-time images captured from simple 0.3 Megapixel camera. The examples of the resulting EDs are presented on Figure 5. Comparing them with the results we got from MATLAB and C++, performing both Sobel and Canny Filters, the quality of the experimental ED is acceptable and very comparable from the aspect of the quality. On a Windows PC equipped with Intel Core I7 Processor with 8 GB of RAM, the Sobel ED in MATLAB took 5-7 seconds on the equal resolution still images from the same scenario, fed from the local hard drive. The FPGA architecture provides real-time almost instantaneous Sobel ED response. We experienced insignificant edge misses in images from more complex scenarios. Table I summarize the FPGA synthesis results. It can be seen that the Sobel kernel and arithmetic architecture fully accommodate on the FPGA chip of Altera DE-1 SoC board occupying only 2% of the adaptive logic modules (ALMs). Each ALM on Altera FPGA board consists of combinational logic, two registers, and two adders [5].

Considering the System on Chip (SoC) nature of the board, where ARM hard processor system exists and interconnection bus architecture is available to connect with the FPGA synthesized ED logic, the HPS-FPGA processing model andFPGA accelerated computing is serious candidate for the future embedded IoT devices with image processing capabilities. The future work is seriously considering involvement of the HPS system on the FPGA board for realization of IMP and VP.

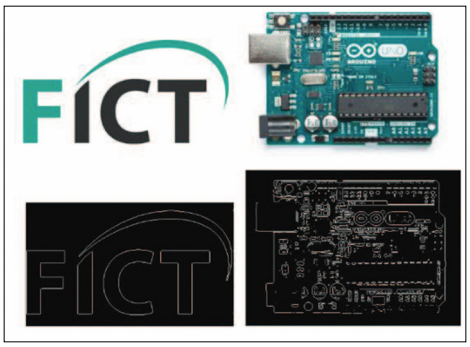

Figure 5. Results from the real-time Sobel ED taken from the VGA output of the FPGA board connected to standard monitor

Family	Cyclone V
Device	5CSEMA5F31C6
Logic utilization (in ALMs)	754 / 32,070 (2%)
Total registers	870
Total pins	94 / 457 (21%)
Total block memory bits	3,152,384 / 4,065,280 (78 %)
Total RAM Blocks	386 / 397 (97 %)
Total PLLs	1 / 6 (17%)

Table 1. FPGA Synthesis Summary

References

[1] Burger, Wilhelm, Mark James Burge, Mark James Burge, and Mark James Burge. *Principles of digital image processing*. London: Springer, 2009.

[2] OV760/761 Camera Module Specification. https://www.voti.nl/docs/OV7670.pdf (Accessed April 10, 2018)

[3] Gradient Filter Implementation on FPGA, Online Article. https://www.element14.com/community/groups/fpgagroup/blog/2015/05/27/gradient-filter-implementation-on-fpgapart2-first-modules. (Accessed April 12, 2018)

[4] A. Ben Amara, E. Pissaloux and M. Atri, "Sobel edge detection system design and integration on an FPGA based HD video streaming architecture, 2016 11th *International Design & Test Symposium (IDT)*, Hammamet, 2016, 160-164.

[5] DE1-SoC FPGA Board Specifications: https://www.altera.com/solutions/partners/partnerprofile/terasic-inc-/board/de1-soc-board.html. (Accessed April12, 2018)