Energy Consumption Measurement of a FPGA Full-HD Video Processing Platform

Heron A. Monteiro
Universidade Federal de Campina Grande-DSC
Campina Grande, Paraíba Brasil
heron@copin.ufcg.edu.br

Nelson C. S. Campos
Universidade Federal de Campina Grande-DEE
Campina Grande, Paraíba Brasil
nelson.campos@ee.ufcg.edu.br

Jozias P. Oliveira
Escola Superior de Tecnologia,
Universidade do Estado do Amazonas
Manaus, Amazonas Brasil
jpoliveira@uea.edu.br

Antonio Marcus N. Lima
Universidade Federal de Campina Grande-DEE
Campina Grande, Paraíba Brasil
amnlima@dee.ufcg.edu.br

Alisson V. Brito
Universidade Federal da Paraba-CI
João Pessoa, Paraíba, Brasil
alisson@ci.ufpb.br

Elmar U. K. Melcher
Universidade Federal de Campina Grande-DSC
Campina Grande, Paraíba Brasil
elmar@dsc.ufcg.edu.br

ABSTRACT

This paper presents an implementation of an energy consumption measurement in a FPGA hardware platform for Full-HD video processing. This platform is capable of processing the video signal as it is acquired, in a rate of 30 fps, preserving its original size without discarding any information. Energy consumption measurement was compared with estimated consumption obtained from simulation. To validate the solution, a Sobel gradient acting in the three color channels was implemented, showing a consumption of 0.24mJ per frame.

KEYWORDS

Embedded Platform, Energy Consumption, Sobel Gradient, Image Analysis.

1 INTRODUCTION

The core of the modern monitoring systems consists of intelligent cameras that are embedded systems which perform real-time video analysis tasks, and send high-level descriptions of the scenes or characteristics of objects of interest detected and may even be able to take decisions [1]. As embedded systems, they have limitations on data transmission rate, energy resources, weight, and footprint [2]. Specifically in monitoring tasks, to handle large volumes of data, many solutions are implemented using low-resolution images, grey scale and are even arbitrarily omitting some frames of the video to be analyzed [1].

Because of the favorable relationship between performance and consumption and for its flexibility and scalability [3] [4] [5], an FPGA platform was chosen.

In the context of building a smart camera, especially in the design of sustainable wireless sensor networks (WSNs) [6], where energy consumption is crucial, this research aims to propose and demonstrate the viability of a platform for acquisition and processing Full-HD videos, without using compression or discarding information, considering all frames and pixels.

The proposed platform consists of a digital camera that captures Full-HD video which is transmitted to the FPGA for processing and the result is transmitted to a monitor. With the energy measurement mechanism, directly coupled to the FPGA, it is possible to obtain the actual consumption of the implemented solutions.

Thus, we are proposing an embedded FPGA platform with enough performance to handle Full-HD resolution videos without discarding information, aiming for an efficient energy consumption to avoid need for active cooling. And, to verify its feasibility, the algorithm of edge detection Sobel, that is widely exploited in the researches of image treatment, has been implemented.

The rest of the paper is organized as follows: Section 2 highlights the related work; Section 3 describes the used platform; Section 4 shows the experiments; Section 5 discusses power consumption aspects, followed by conclusions in Section 6.

2 RELATED WORKS

Focusing on implementation of video processing images in FPGA platform, solutions have been proposed using Spartan 3 [7]; Spartan 6 [7]; Virtex-5 [8] [9] [10] [11]; Virtex 6 [12] [3] [13]; Altera Cyclone III [9], Altera Cyclone IV [14] and Altera Cyclone II [15]. The analyzed solutions manipulates images with different resolutions varying from 256×256 [8] to 1280×1024 [12].

For validation, verification and hardware/software co-design, programming languages such as Matlab [8] [7] [16] [15] [14] and C++ [13] were used. The OpenCV library was used to compare results [13]. For modelling and simulation of the solutions Simulink [8]; ModelSim [10] [13] [11] and Convey [13] were used.

To validate solutions, edge detection algorithms such as Sobel [8] [7] [10] [16] [14] [3]; Harris and Stephen filters [9]; HDR [12]; HOG to detect pedestrians [13] and face detection using LBP (Local Binary Pattern) [15] were used.

Some studies have evidenced the number of frames per second processed varying from 25 [3] to 185 [10], clock frequency from 27 MHz [11] [16] to 125 MHz [12], and energy consumption estimation [14] [12] [15].

Some of the works highlighted the structures used for storage and to assist the necessary calculations. Two FIFOs and a set of registers were used to make the displacement of the image in a 3×3 size calculation window to assist in the calculation of the Sobel gradient [10]. In another work, a similar approach is using three buffers [16] where, after a lag time required for filling the buffers, the magnitude of the gradient was calculated for each pixel in three clock cycles. For the treatment of image resolution 640 × 480, nine 16-bit registers buffers and two sixteen wide FIFO with depth of 640 were used. The buffers and FIFO registers store a 3 × 3 window that is sent to the gradient calculation cells and the result is sent to an image segmentation module. Each of the FIFO registers stores a line of the image. At a time t1 the first FIFO is filled with a line of the image. At the time t2 the content of the first FIFO is copied to the second FIFO, while the first FIFO is filled with a new line. When the two FIFOs are fully filled a pixel is passed to a 3 × 3 window, and after three clock cycles of the convolution is calculated. After $640 \times 2 + 3$ clock pulses the first 3 × 3 window will be filled [14]. In Wang et al. [15] a resolution of 640×480 image is received which is resized to 320×240 for processing. Only in work [11] the treatment of color images is made.

Various metrics are used for the evaluation of the implemented solutions. However, due to the fact that not all work inform values for all metrics, a thorough comparative study among these solutions is not feasible.

3 FPGA EMBEDDED PLATFORM

The Panasonic digital camera CMQ2148X with 3.6 mm lens was used as the image capture device that provides images in RGB format, full-HD, operating at a rate of 30 fps. Its output is available in SDI standard [17].

For the processing of the images a FPGA DE2-115 board was used, consisting of Cyclone IV E (EP4CE115), with 114,800 logic elements. The family of FPGA Cyclone is classified by the manufacturer as being low cost [18]. The HSMC interface (High Speed Mezzanine Card) [19] of the board was used to connect to the camera.

To make it possible to communicate between the camera and the FPGA board, a SDI to HDMI converter, a HDMI-DVI converter cable and HSMC-DVI adapter card, were used. The adapter card allows for reception and transmission. Images captured from the camera were connected at the receiving port. The sending port was connected to a monitor Samsung T240M. The Figure 1 resume the platform scheme.

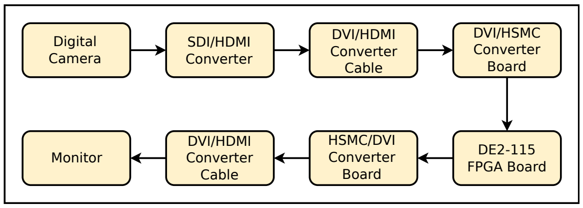

Figure 1: Platform scheme.

To check all platform components, using Quartus II$^{\text{TM}}$II version 15.1, the demo version of the HSMC-DVI card was implemented retaining only the loop-back mode, where the image sequence displayed on the monitor is equal to what is received from the camera.

To display the images, it was necessary to identify and include, in the application source code, the EDID (Extended Display Identification Data) of the monitor. The EDID is a data structure that describes the capabilities of digital video monitors. This code was stored in the HSMC-DVI card's NVRAM memory.

According to the EDID for the Full-HD resolution, in fact, 2200 × 1125 pixels are manipulated. However, only 1920 × 1080 pixels are captured by the camera sensor. The other pixels are for control information and picture synchronization. On the monitor only the pixels captured by the camera sensor are effectively displayed.

Considering a rate of 30 fps, every second 74.25 million pixels are handled. Since each pixel is represented by 24 bits of information, each second 1,782 Mbits are transferred, resulting in a transfer rate of approximately 1.66 Gbps. The pixel clock is 148.59 MHz.

The loop-back mode uses 434 logic elements, 105 registers and 1 PLL of the FPGA.

4 EXPERIMENT

To be able to obtain a better comparison with the majority of works mentioned, in this study all aforementioned values were reported. Regarding the amount of frames per second the goal is to process all frames that the capture device provides, not despising any information.

Full-HD images were processed, without artifice to reduce the original image, using three color channels. Run time, latency and frequency were highlighted. Power consumption information was also available, using available estimation tools and taking measurements on the platform.

Using the loop-back code as base, the Sobel operator was added, whose output was sent to the monitor instead of the loop-back output by positioning a switch on DE2-115 board. For applying the operator on a given pixel, it is necessary that the values of the neighbouring pixels are made available, requiring a total of nine pixels. Only the magnitude of the gradient was calculated.

To avoid storing an entire frame, which uses about 7 MB of memory, three circular buffers were implemented, one for each color channel, using a vector structure and a pointer to this vector, similar to Kong et al. [14]. The buffers used about 13 KB of RAM.

According to the static timing analysis obtained directly from the Quartus$^{\text{TM}}$II hardware synthesizer tool, it was not possible to calculate the sums of the three pixels of the partial gradients in the same clock cycle. So, at first, only two pixels are added (chosen the extremes), and to this result is added the double of the intermediate pixel, thus implementing a two-stage pipeline for this calculation. A third pipeline stage was implemented for calculating the sum of partial gradients and a fourth pipeline stage calculated the sum of absolute values of the partial gradients. The same operation was implemented for each one of the three color channels.

With respect to the implementation of the loop-back mode, the Sobel operator used additional 717 logic elements, 432 registers and 106,416 bits of memory. The latency of the solution was 4.9 μs, equivalent to the transmission time of 3,843 pixels, needed to fill the circular buffers. From then on, pixels are processed as they become available.

5 POWER CONSUMPTION

In order to obtain power consumption information, two approaches were used: one using the values provided by the synthesizing tool and another one by measuring directly on the FPGA board.

5.1 Power Consumption Estimation

To make a power consumption estimation Quartus^{TM}II Version 15.1 PowerPlay Power Analyzer tool and Synopsys VCS netlist simulation were used. Initially all functions of the demo application were removed, including the loop-back. This application was called empty mode and was aimed to make the estimation of the static consumption of the FPGA. The estimation of interest refers to the consumption of digital logic circuits which, according to the documentation of the FPGA used, is powered by VCCINT (net VCC1P2 of the printed circuit board). The tool report provides the information of static and dynamic consumption of this component.

To analyze the effect of pixel values in the simulation results, four different simulation runs were done with pixels: i) constants; ii) alternating between 000000 and FFFFFF; iii) random; and iv) of a real image. In the four sets the static consumption was about the same. In relation to the dynamic consumption, the lowest value was observed in the first set. The highest one was observed in the second and third sets (approximately 200% larger with respect to the first set); and, in the final set, the consumption was about 60% higher than in the first one. This last set of pixels was used in the simulations of the loop-back and Sobel modes.

A simulation testbench was developed with values of the input pixels maintained constants. In this empty mode the tool report provided, as values of static and dynamic current, assigned to VCCINT, respectively, 16.3 mA and 9.4 mA. According to the Power Analyzer tool documentation, the dynamic consumption includes consumption related to the configuration of the FPGA board. As combinational logic was removed and the input pixels were kept constant, the informed dynamic consumption in this mode should refer only to the board configuration. Thus, the information of interest was static consumption. As a VCCINT voltage is 1.2 V, the static power consumption estimation was 19.6 mW.

For loop-back mode were obtained as static and dynamic VCCINT values, respectively, 21.4 mA and 974.7 mA. Assuming the electric current required for configuration of the FPGA does not change, the dynamic VCCINT current value must represent the current consumed in execution (run time).

For Sobel mode the values 22.2 mA and 1315.6 mA for static and dynamic currents were reported by the tool, respectively, being observed an increase of 340 mA with respect to the loop-back mode, representing an increase of 409.0 mW, corresponding to the power consumption of the execution of the Sobel gradient, which represents an estimated consumption of 13.6 mJ per frame.

In order to validate the values reported by the tool, SobelR mode was created, which processes only one color channel (red), and thus excluded two circular buffers and the corresponding gradient calculation operations. With respect to the loop-back mode, this mode used additional 251 logic elements, 152 registers and 35,472 bits of memory. Compared with the Sobel mode, there was a decrease of respectively 59%, 47% and 33% of these values.

The power consumption estimation for static current value was the same for both SobelR and Sobel modes, but there was an increase of 54.62 mA of dynamic current in the SobelR mode. This result is inconsistent since there was a resource usage reduction and a power consumption increase of approximately 65.54 mW. The Thermal Power Dissipation report information was analyzed. It showed that SobelR mode consumption of combinatorial logic (35.22 mW) represented an increase of 23% compared to the consumption of Sobel mode (28,55 mW). However, a decrease of about 68% in the power consumption of registers was reported, falling from 4.27 mW in Sobel mode to 2.92 mW in SobelR mode.

5.2 Power Consumption Measurement

Due to the above results, an alternative for power consumption estimating was sought, making the measurement of power consumption in the physical FPGA platform.

The chosen measurement technique uses the parasitic resistance (R_L) of the inductor of the on-board switched power supply circuit to measure the current in the inductor L [20]. The sensor consists of a resistor R and a ceramic capacitor C_S in parallel with the inductor. The capacitor voltage V_{C_S} is the desired output of the measuring device, according to Figure 2.

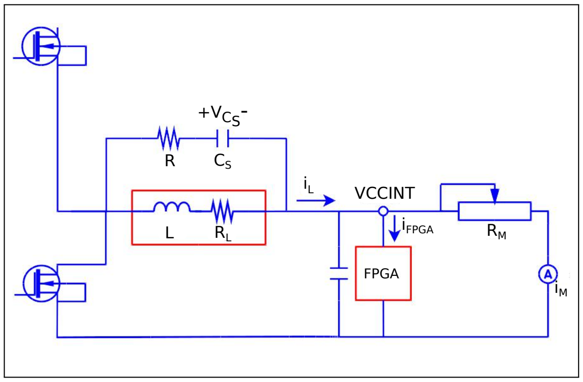

Figure 2: Calibration scheme [20] [21].

The appropriate values for C_s and R_L were chosen, so that

$$R.C_S = L/R_L. \quad (1)$$

The voltage of the capacitor is directly proportional to the inductor current, as Eq. 2.

$$V_{C_S} = i_L.R_L \quad (2)$$

Considering the value $L = 3,3\ \mu H$ provided by the data-sheet of the inductor manufacturer, and a measured resistance of $R_L = 38\ m\Omega$ (Figure 2), we chose to use $C_S = 110\ nF$ and $R = 5,6\ k\Omega$.

In order to calibrate the measurement, voltage V_{C_S} measurements were made by varying the current value through the use of different resistor values R_M. The FPGA board was configured in the empty mode.

The values and their respective linear approximation are shown in Figure 3. Projecting voltage V_{C_S} equal to zero leads to an estimate of the current $i_{FPGA} \approx 19\ mA$. Once this process of calibration done, resistor R_M and ampere-meter i_M were eliminated to retain only the voltage measurement V_{C_S}.

Voltage measurements were made for loop-back, SobelR and Sobel modes, presenting respectively the values 0.48 mV; 0.56 mV, and 0.70 mV. Considering the equation of the straight line obtained and VCCINT voltage value of 1.2 V, an increase of 2.5 mW consumption from loop-back mode for SobelR mode and an increase of 7.1 mW

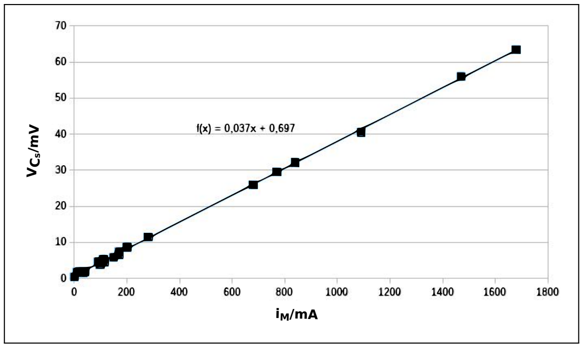

Figure 3: Measured voltage values depending on the current.

from loop-back mode for Sobel mode were calculated. This represents an increase of almost three times of the consumption of Sobel mode with respect to SobelR mode, which was more consistent than the values informed by the estimation tool.

Considering the Sobel mode increase in consumption of 7.1 mW, and knowing that thirty frames are processed per second, the Sobel processing consumption of each frame is 0.24 mJ. Several measurements were made with real images and their gradients and the same voltage value was observed. The only variation in voltage was occurring when an image of a blank wall was placed before the camera. In Sobel mode a voltage increase of 0.64 mV was measured, corresponding to an increased of consumption of 6.5 mW.

6 CONCLUSION

This paper presented a platform for video processing in Full-HD resolution, manipulating images at a rate of 30 fps, processing at a clock rate of 148.59 MHz, without artifice to reduce the original image, using three color channels, avoiding discarding any information, and allowing the power consumption measurement. The implementation of the Sobel gradient for the three color channels used 717 logic elements, 432 registers and 106,416 bits of RAM. The solution showed a latency of 4.9 μs, and an energy consumption of 0.24 mJ per frame.

Results for all the studied parameters were reported, being manipulated images with higher resolution. The measured consumption was lower than the estimated consumption, even when compared to the work presented here.

According to the results presented it was shown that it is possible to have a platform, which processes high-resolution videos without discarding information, affordable and energy-viable. Also, according to the amount of FPGA resources used, it is evident that more complex algorithms can be implemented.

REFERENCES

[1] Ahmed Nabil Belbachir and Peter Michael Göbel. Smart cameras: A historical evolution. In *Smart Cameras*, pages 3–17. Springer, 2009.

[2] Ricardo Carmona-Galán, François Berry, Richard Kleihorst, and Dominique Ginhac. Special issue on architectures of smart cameras for real-time applications. *Journal of Real-Time Image Processing*, pages 1–2, 2016.

[3] Xie Kunzhi, Qiao Fei, Qi Wei, and Yang Huazhong. Smart-eyes: a fpga-based smart camera platform with efficient multi-port memory controller. In *3rd International Conference on Multimedia Technology (ICMT-13)*. Atlantis Press, 2013.

[4] Micha l Fularz, Marek Kraft, Adam Schmidt, and Andrzej Kasiński. The architecture of an embedded smart camera for intelligent inspection and surveillance. In *Progress in Automation, Robotics and Measuring Techniques*, pages 43–52. Springer, 2015.

[5] Stephan Blokzyl, Matthias Vodel, and Wolfram Hardt. *A Hardware-accelerated Real-time Image Processing Concept for High-resolution EO Sensors*. Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV, 2012.

[6] Tifenn Rault, Abdelmadjid Bouabdallah, and Yacine Challal. Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104–122, 2014.

[7] Girish Chaple and RD Daruwala. Design of sobel operator based image edge detection algorithm on fpga. In *Communications and Signal Processing (ICCSP), 2014 International Conference on*, pages 788–792. IEEE, 2014.

[8] Dr. A.Y. Deshmukh Ankush R. Bhagat, Swati R. Dixit. Vhdl based sobel edge detection. *International Journal of Engineering Research and General Science*, 3(1):1217–1223, 2015.

[9] Merwan Birem and François Berry. Dreamcam: A modular fpga-based smart camera architecture. *Journal of Systems Architecture*, 60(6):519–527, 2014.

[10] Sanjay Singh, Anil Kumar Saini, Ravi Saini, AS Mandal, Chandra Shekhar, and Anil Vohra. Area optimized fpga-based implementation of the sobel compass edge detector. *ISRN Machine Vision*, 2013, 2013.

[11] Sanjay Singh, Anil Kumar Saini, and Ravi Saini. Real-time fpga based implementation of color image edge detection. *International Journal of Image, Graphics and Signal Processing*, 4(12):19, 2012.

[12] Pierre-Jean Lapray, Barthélémy Heyrman, Matthieu Rossé, and Dominique Ginhac. A 1.3 megapixel fpga-based smart camera for high dynamic range real time video. In *Distributed Smart Cameras (ICDSC), 2013 Seventh International Conference on*, pages 1–6. IEEE, 2013.

[13] Xiaoyin Ma, Walid A Najjar, and Amit K Roy-Chowdhury. Evaluation and acceleration of high-throughput fixed-point object detection on fpgas. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(6):1051–1062, 2015.

[14] Weili Kong, P Cheng, and Zhuo Bi. Real-time sobel edge detector. In *Proceedings of the 6th PSU-UNS International Conference on Engineering and Technology*, 2013.

[15] Nai-Jian Wang, Sheng-Chieh Chang, and Pei-Jung Chou. Real-time multi-face detection on fpga for video surveillance applications. *Journal of the Chinese Institute of Engineers*, 38(1):120–127, 2015.

[16] Jin Wang, Honggang Wang, Shaoen Wu, Xiaodong Lin, and Qing Yang. Design and implementation of real-time sobel edge detection on fpga for mobile device applications. In *Proceedings of the ACM International Workshop on Mobility and MiddleWare Management in HetNets*, pages 9–14. ACM, 2015.

[17] Charles Poynton. *Digital video and HD: Algorithms and Interfaces*. Elsevier, 2012.

[18] Cyclone iv fpgas, hide features overview. http://www.altera.com/products/fpga/cyclone-series/cyclone-iv/overview.html. Accessed: 2016-02-10.

[19] High speed mezzanine card (hsmc). http://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ds/hsmc_spec.pdf. Accessed: 2016-02-22.

[20] LINFINITY Application Note AN. 7: A simple-current sense technique eliminating a sense resistor. *Last visited*, 7, 2007.

[21] Altera cyclone iv development education board (de2-115). http://wiki.ntb.ch/infoportal/_media/fpga/cyclone_iv/de2_115_schematic.pdf. Accessed: 2016-05-02.