Configurable Platform for Real Time Video Processing and Vision Systems

José Germano, Ricardo Baptista, Leonel Sousa
INESC-ID/IST, R. Alves Redol, 9, 1000-029, Lisbon, Portugal
Email: {jahg, rmsb, las}@inesc-id.pt

***Abstract*— This paper presents an inexpensive and cost-effective configurable platform for designing real time video processing and vision systems. The platform is designed around a Field Programmable Gate Array and provides video input and output interfaces. It has been used to implement several different image and video processing systems, namely with the purpose of prototyping and teaching courses in the area of video processing systems. Experimental results show that this platform provides enough resources and speed to implement even complex systems in real time.**

I. Introduction

Nowardays configurable Field-Programmable Gate Array (FPGA) technology is capable of executing complex video processing and vision tasks in real time. These applications include histogram calculation [1], intrusion detection [2] or visual encoding [3]. In such applications, a FPGA core may receive the image data captured by a digital camera, process an image or sequence of images and display the resulting frames on a monitor or pack the results an send it through a bus. Also, for some applications like [3], there are specific requirements that can only be fulfilled by creating a custom made design. There are some FPGA boards with real time video capabilities like [4], [5]. Nevertheless, analyzing the characteristics of such systems, the first provides a rudimentary video port while the second is complex, expensive and not easy to use.

This paper describes a small size configurable platform capable of real time video processing based on a FPGA. This inexpensive and cost-effective platform can be easily used and configured for different types of applications. Image capture and display modules are independent and may be integrated with the image processing modules. These modules are described in VHDL and can be configured for video input and output with different interfaces. This platform is now available in small quantities being useful for prototyping and teaching in the area of video and image processing [6].

This paper continues in Section II and III by presenting the architecture and discussing the implementation of the video processing platform. Section IV describes image processing systems that have been developed on this platform and presents experimental results. Finally, Section V concludes the paper.

II. Video Processing Platform Architecture

The proposed platform provides interfaces with an image capture device, a digital camera, and an output video peripheral a video display. It is also equipped with a standard bus connector to allow debug or stream data output. In fig. 1 is presented the block diagram of the platform developed in the scope of this paper.

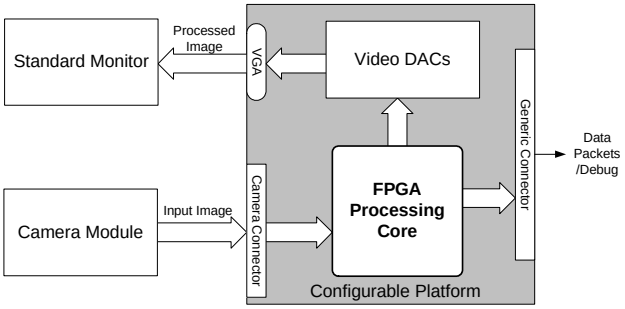

Fig. 1. Block diagram of the platform.

To achieve a complete video processing system the following main modules have to be considered: *i)* image acquisition module; *ii)* processing core; *iii)* output peripherals, namely for video display. Since the platform was designed around a FPGA, it has been tried to exploit as much as possible the capacity of the FPGA to implement all these modules. The main characteristics of these modules, that are organized accordingly to the architecture depicted in fig. 1, are discussed in the following paragraphs.

Image Acquisition

The image capture device must provide color information and have a digital output to avoid the usage of Analog to Digital Converters (ADC). Image size and format must be configurable in order to adapt these parameters to the applications. Currently, there are many CMOS image sensors that meet these requirements. The usage of CMOS sensors has the advantage of low power consumption, when compared to the CCD sensors.

The board uses an interface compatible with most of the development camera modules from OmniVision [7]. These modules can be configured using a protocol similar to the Inter Integrated Circuit (I2C) protocol, and thereby provide access to the internal camera registers that control all image parameters and characteristics.

Processing Core

This is the most important component of the platform. It is required the FPGA to be able to process video in real time, which requires a device with a high operation

frequency and with sufficient hardware resources. The design of the circuits is made on an application basis and by using an Hardware Description Languages Hardware Description Languages (HDL), such as VHDL.

The processing core is responsible not only for video processing but also for generating the synchronization signals required for the Video Graphics Array (VGA) port and for configuring and controlling the operation of the digital camera module. The core must be connected to the several devices of the platform, namely the digital camera, the video ports, a generic I/O and some on-board debug devices. To enable a boot time configuration it is also required to include a solid state non-volatile memory device, like a ROM or a PROM.

Output Peripherals

To ensure high compatibility, the video port follows the standard VGA interface. This port requires tree high speed video Digital to Analog Converter (DAC)s, one per color component: Red, Green and Blue (RGB). DACs convert the digital output from the FPGA to an analog level suitable for a standard monitor. Also, to ensure correct impedance adaptation, output circuits condition the signal to provide an on-board load resistance of 75Ω. The synchronization signals required for the interface are generated in the FPGA. The platform also provides another I/O bus to enable debug or stream data output. This interface is compatible with expansion modules commercially available [8]. In order to achieve some basic on-board debugging, I/O devices were included in the design.

III. Configurable Platform Implementation

The developed platform was implemented on a four layer Printed Circuit Board (PCB) in which the two inner are power planes while the outside layers are used for rooting. The board was developed using Altium Protel 2004, a complete board-level Computer Added Design (CAD) software tool. Along with the developed board, generic VHDL modules were also created for configuring and controlling the image capture and display.

This section describes the particular characteristics of this board and the design options that were considered when implementing the architecture introduced in the previous section.

A. Prototype processing core

The developed board uses the XILINX SPARTAN-3 XC3S400 [9] FPGA as the processing core. This represents the actual best trade-off between cost and capacity, providing enough resources to process video in real time. The XC3S400 has $400\ k$ system gates, a total memory of $288\ kb$ distributed in 16 RAM blocks, 16 dedicated 18-bit multipliers and fast look-ahead carry logic. The dedicated multipliers allow to save system gates and to increase the operating frequency, which is important for processing images with high resolution. This FPGA requires three independent supply voltages ($1.2\ V$, $2.5\ V$ and $3.3\ V$, the last one to I/O).

Although XC3S400 has a maximum of 141 I/O ports, this design does not use them all. Two external connectors are provided: *i)* one is dedicated to the interface with the digital camera; it uses 25 pins including two Global Clock (GCLK) inputs and is compatible with several camera modules from Omnivision; *ii)* the other is a generic I/O connector with 40 pins in which four of them are GCLK inputs and three are power, this expansion is provided for debug, to output results or to connect accessories or peripherals; it supports several expansion modules provided by Digilent [8]. Each DAC is connected to the FPGA through a dedicated 8-bit wide bus. It receives the sample clock from the FPGA and the synchronization signals necessary to the VGA port.

Finally basic debug is achieved using a small onboard user interface. It consists on a 4 way slide switch, two push buttons and 8 leds. All the available I/O signals can be seen in fig. 2.

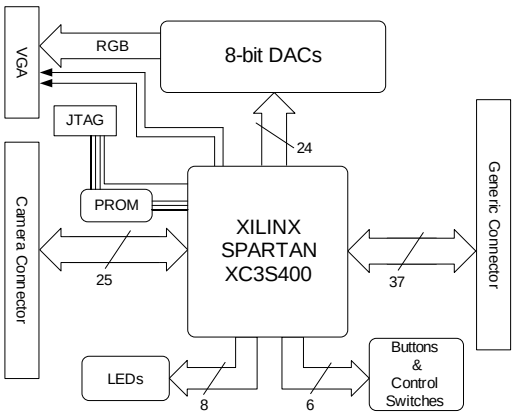

Fig. 2. FPGA available I/O signals.

The FPGA can be configured using the JTAG configuration pins available through a six pin header or by a PROM device. It employs the XCF02S serial configuration flash PROM to store FPGA configuration data. Using this design, the FPGA automatically boots from the on-board PROM whenever the power is applied. The configuration of the PROM is also done through the JTAG header.

B. Board

Figure 3 shows a picture of the actual developed PCB board, were the main components are highlighted.

Since FPGAs can implement an almost infinite number of applications at undetermined frequencies and in multiple clock domains, it is hard to estimate its current demands. These transient currents are the main cause of ground bounce thus making the decision on the regulators to use on the Power Distribution System (PDS) very important. Two choices were considered to generate the required voltage levels: linear regulators and Pulse Width Modulation (PWM) regulators. While the first one requires less external components, its efficiency is highly dependent on the dropout voltage. In opposition, the PWM regulators are able to achieve high efficiency levels even

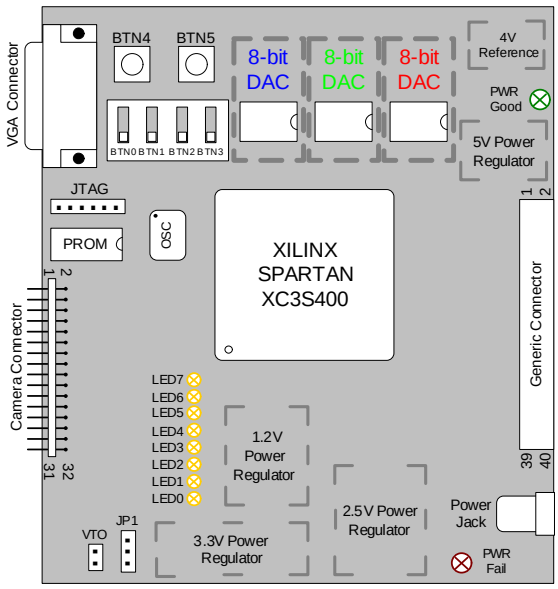

Fig. 3. Video platform board.

with high voltage dropouts, thus saving thermal dissipation area. Based on such arguments, the choice was to use linear regulators for the 5 V power grid and PWM regulators for all the supply voltages of the FPGA (1.2 V, 2.5 V and 3.3 V). It was used the MAX1830/MAX1831 a low voltage PWM, step-down regulator, which delivers a current of up to 3 A with a peak efficiency of 94%.

To design an autonomous system, it is necessary to include a power supplier for the digital camera. As the required voltage is 5 V or 2.5 V, this device can use a power source already included in the design. Another necessary component is a clock generator, for this design it was employed the CFPS-73, a 50 MHz, HCMOS 3.3 V oscillator.

The power plane was divided in six islands, one for each of the voltage levels required for the FPGA and one for each DAC. The ground plane is also divided, but in a more simple manner. All the digital circuitry shares the same plane, only the analog part of the DAC has a separate ground. The usage of the split planes lead to a small distance between the capacitors and the power pins therefore increasing the decoupling efficiency.

C. VGA display port

The display port is based in three separate 8-bit DAC converters, TLC5602C, one per RGB channel. This DAC presents a low power consumption (typically 80 mW), 20 MHz conversion rate, single 5 V power supply and TTL digital input voltage. In order to ensure the correctness of the digital output logic levels from the FPGA, the V_{CCO} voltage must be set to 3.3 V. Another necessary input for this DAC is a reference voltage of 4 V, this voltage is generated with a MAX6004, a low power and low dropout voltage reference IC.

In order to decrease the circuit noise, each DAC has a separate analog power and analog ground plane, being isolated from the digital circuitry and from the other DACs. The signal conditioning was performed using high speed video buffers, with a 75 Ω resistance in series with the output. Since a VGA monitor represents a load resistance of 75 Ω , it is necessary to include a video buffer with a gain of 2. The design employed the MAX4219, a triple video buffer with a preset gain of 2, in order to reduce the required components.

The voltage level required for the DACs is generated by a TPS78601, a ultra low-noise, low-dropout linear regulator with a maximum output current of 1.5 A. Since the maximum input voltage for this regulator is 6 V, the maximum dropout voltage is 1 V leading to a minimum efficiency of over 80%.

D. Image Capture and Display

For image acquisition two modules were created, one to capture the input image and the other to make the power-on configuration of the digital camera by programming the internal registers. A display module was also created to produce a screen visual image. This module is able to display original as well as processed images.

Image capture

The input image is acquired by using a digital camera module from Omnivision [7]. The board is compatible with the camera module based on the OV7620 CMOS image sensor or other modules with more recent sensors like the OV9650. All camera functions can be configured using a serial data transmission protocol, Serial Camera Control Bus (SCCB) [10], which is a simplified version of the Philips I2C [11] protocol. The OV9650 sensor is able to capture a window resolution of 4.13 $mm \times 3.28$ mm, generating an array of 1300×1028 with a resolution of 3.18 $\mu m \times 3.18$ μm. The digital video port supports 8-bit YUV/YCrCb, GRB 4:2:2 or raw RGB data format. The sensor has a signal to noise ratio greater then 40 dB and a dynamic range of more than 62 dB. The supply voltage for this sensor is 1.8 V for the core and 2.5 V for the I/O and analog circuitry and it requires less then 50 mW when it is active and only 30 μW while in standby.

The camera can be programmed to capture frames in SXGA, VGA, QVGA and other specific formats or other programmed format. Frame synchronization is performed by detecting a high pulse in the vertical sync (`VSYN`) signal and a new line occurs with a low pulse on the horizontal reference (`HREF`) signal, moreover, the camera also provides the pixel clock (`PCLK`) signal. Figure 4 shows the temporal diagrams of those signals. The pixels information is updated every falling edge

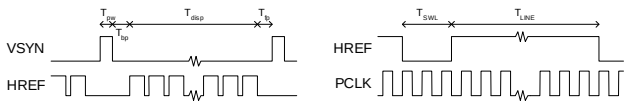

Fig. 4. Camera module sync signals.

of the `PCLK` signal, which means that pixel information can be read on the rising edge of `PCLK`. An auxiliary module was implemented for converting YUV 4:2:2 format to RGB 4:4:4 format.

A hardware module was developed for configuring the camera internal registers. The SCCB configuration protocol was implemented by using an hierarchical state machine with tree layers. At the first layer, `camera_config`, commands are sent to start a read or a write cycle, supplying the register address and value. The next layer consists of two state machines: one to implement the write cycle (`write_cycle`), and the other the read cycle (`read_cycle`); only one can be active at a time. The third layer is responsible for generating all the bit sequences necessary to implement the SCCB protocol. After determining a suitable register configuration set to a given image capture and format, the configuration values can be set as constants and included in the VHDL description to make the power-on configuration.

Image display

This module generates all synchronization signals necessary for a standard VGA monitor. The adopted configuration uses a resolution of 640×480 and a refresh rate of 60 Hz [12]. This module corresponds to a simple state machine that generates the signals necessary to control the image display. The image to be visualized can be stored into a frame memory block or be provided directly by the processing module.

IV. Image Processing and Vision Systems

The platform has been intensively used and its functionalities were completely tested. This section describes three image processing systems that have been designed based on the developed configurable platform. The first, for histogram calculation, is a simple processing module that can be integrated in a more complex design. The second one is a complete motion detection system and the last one is the implementation of a complex visual encoding system that performs a retina-like processing. This last module produces a sequence of spike impulses that can be used to excite the human visual cortex. Experimental results about the performance of the system are presented in table I.

A. Histogram Calculation

Histogram calculation is one of the essential steps in histogram equalization or binarization, methods that are used for image enhancement and segmentation. There are several examples in which these technics significantly improve the image quality or reduce the size of the data presented in the images. Typical histogram equalization algorithms and applications can be found in [13].

Let us consider a grayscale image, and that the histogram is calculated by counting all the image pixels that have each of the intensity levels presented in the image. A circuit capable of performing this operation if represented in fig. 5.

The capture module was configured to process a frame in XVGA size. The required hardware for this calculation is low considering the available resources: it only employs 54 slices (1%) and 1 RAM blocks (6%) . The maximum operation frequency is only limited by the image capture device, in XVGA the maximum frame rate of the OV7950 sensor is 15 $frames\ per\ second(fps)$.

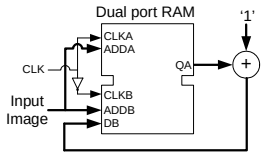

Fig. 5. Histogram calculation block diagram.

B. Passive Motion Sensing

This subsection refers a motion detector based on a stationary mounted camera. Although there are several other approaches, like [2] or [14], the chosen application uses time recursive image processing. The adopted technic evaluates the differences in the intensity level in consecutive frames [2]. The required processing can be described by (1) and (2)

$$d(k,l,n) = i(k,l,n) - a \cdot \sum_{j=1}^{\infty} b^{j-1} \cdot i(n-j) \qquad (1)$$

$$b(k,l,n) = \begin{cases} 1 & ; \ d(k,l,n) > S \\ 0 & ; \ \text{else} \end{cases} \quad , S \in \mathbb{N}_0 \qquad (2)$$

where *k* and *l* are the coordinates of a pixel, *n* is the temporal discrete dimension, *i* represents the input image, *a* and *b* are filter coefficients, and *S* defines the movement threshold for a pixel. Hence, (2) provides information about the pixels were movement was detected. Gathering the information on all pixel changes, movement can be detected by summing all the pixels information and comparing it to a given threshold for the entire image. Considering a Field-Programmable Logic (FPL) implementation the required processing can be accomplished by the circuit represented in fig. 6.

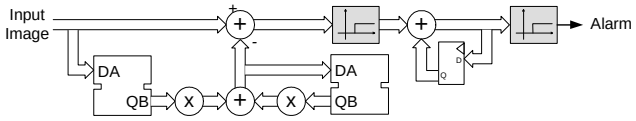

Fig. 6. Motion sensing block diagram.

The input image was represented in greyscale with a window size of 32×32, although, different sizes can be employed. With this resolution, considering also the costs of the image capture and display, this system uses 379 slices (7%), 2 multipliers (12%) and 4 RAM blocks (25%). The design is able to process frames at a maximum rate of 71 $kfps$, much higher then the rate corresponding to real time. These results can also be seen in table I.

The system delivered the expected results. Although, for a correct operation, the threshold parameters are critical.

C. A Bio-Inspired Visual Encoding System

Following a similar approach to the used by the human visual system, the developed system uses a bio-inspired (retina-like) visual processing front-end [3]. The purpose is to map the visual stimulus in a sequence of action potentials, also

called spike trains. It transforms the visual world into electrical signals that can be used to excite, in real time, the neurons in the visual cortex.

The developed modules for the bio-inspired processing application are divided in two distinct parts: *Retina Early Layers* and *Neuromorphic Pulse Coding*. Figure 7 represents the block diagram of this bio-inspired processing module.

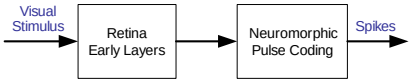

Fig. 7. Classic Bio-Inspired processing module.

It is specified that the *Retina Early Layers* module should be able to process frames at a maximum rate of 100 Hz and that the maximum spike rate generation for the *Neuromorphic Pulse Coding* must, at most, be 1 kHz per location [15].

The retina early layers

The full processing mechanism for the *Retina Early Layers* is presented in fig. 8.

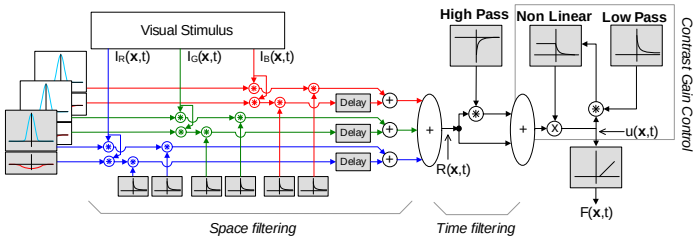

Fig. 8. *Retina Early Layers* diagram.

The spatial calculus is done using one Difference of Gaussians (DoG) filter, if space-time independency is considered. Nevertheless, the spatial filtering should be done using two gaussians in order to independently convolve the center and surround since several authors have suggested that space-time filtering is not completely separable [16]. Thus this implies the use of a low pass filter, after each gaussian. As different colors are treated differently by the retina, the spatial filter can be expanded into three separate filters, one per each RGB channel. As depicted in fig. 8, considering space-time dependency there will be six receptive fields, two per RGB channel. The time filtering is done using the high-pass filter and the Contrast Gain Control (CGC) non-linear function was implemented using a look-up table and a low-pass pass.

Neuromorphic pulse coding

As the human brain responds to action potentials, the instantaneous firing rate at the output of the *Early Layers* must be modified to meet this new representation. This task is carried out by the *Neuromorfic Pulse Coding* block. An approach to fulfill this goal is to use the standard integrate-and-fire mechanism with leakage factor. Figure. 9 shows the schematic representation of this block.

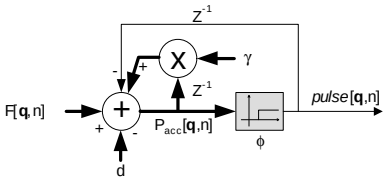

Fig. 9. Integrate-and-fire block diagram.

The module can be described by (3) and (4)

$$P_{acc}[\mathbf{q},n] = F[\mathbf{q},n]+\gamma\cdot P_{acc}[\mathbf{q},n-1]-pulse[\mathbf{q},n-1]-d \quad (3)$$

$$pulse[\mathbf{q},n] = H(P_{acc}[\mathbf{q},n]-\phi) \quad (4)$$

where γ is the feedback loop gain, d is a leakage factor, ϕ is the firing threshold and $\mathbf{q}$ represents the discrete spatial dimensions.

FPL Implementation

The developed processing module was configured to have as input the luminance component, and a frame window of 32×32. The visual frames were captured with a 128×128 window size. Filtering followed by a decimation process lead to a reduced image with 32×32 pixels. Filtering was done line by line delivering an output in the same way.

The architecture was described in VHDL and implemented and tested in the developed platform. The input was represented in 8-bit greyscale and a 12-bit representation was internally used to reduce computational and filter discretization errors. It was also necessary to develop other modules, as depicted in fig. 10.

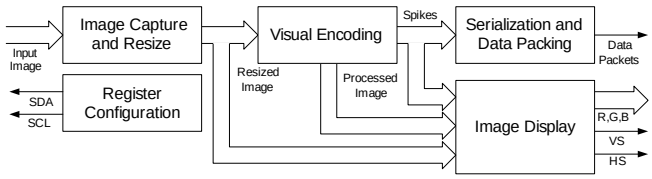

Fig. 10. Digital logic block diagram.

The implementation here reported only processes one of the images components and considers space-time separability. The global FPL architecture for one RGB channel is shown in fig. 11 and fig. 12 shows the adopted FPL architecture for generating the spikes.

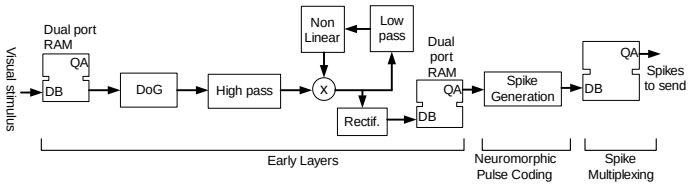

Fig. 11. *Early Layers* FPL full architecture (adapted from [15]).

To validate the actual spike generation it was developed a module that recovers the image using spike information.

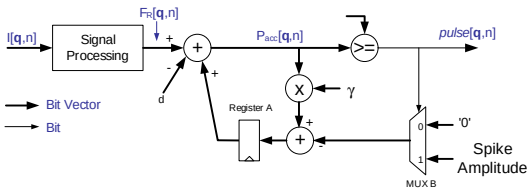

Fig. 12. Integrate-and-fire adopted architecture.

This recovery system was accomplished using a low-pass filter with a very low cut-off frequency. This filter was implemented using a chain of two low-pass Infinite Impulsive Response (IIR) filters.

As the captured image is in greyscale, so is the output. Also, since an array of 32×32 pixels if very small for visual analysis, the output image was enlarged four times by repeating the same pixel of the original image 16 times in the output image resulting in a window of 256×256 pixels. The signals generated by this circuit only control the image display.

The processing module is much more complex then the previous ones and the required hardware is an evidence of this. It employs 811 slices (15%), 4 multipliers (25%) and 7 RAM blocks (44%). This way, images obtained with this processing module can be observed at fig. 13.

Fig. 13. Photographs of the experimental results obtained with the visual encoding system.

In the upper left corner is the input color image, in the upper right is the result of applying the DoG filter, the bottom left is the DoG filter image after the high pass temporal filter and at last in the lower right corner is the result of recovering the image from the spike data. To perform the display of the processed image in the several stages of computation, the VGA output module was modified to display four different images. The full design, including the capture and display modules, required 1298 slices (24%), 12 multipliers (75%) and 15 RAM blocks (93%). The maximum operating frequency is 75 $kfps$, which is much higher than the 25 Hz required for real time. In table I are the implementations costs for all the processing applications.

V. Conclusions

This paper presented a configurable platform for developing video processing and vision systems. Several systems were developed in this platform and experimental results show that it can implement complex systems and still be able of real time processing. In fact, even for both motion detection and visual encoding, disregarding the required memory, the system is still able to process an image frame with a resolution of 1280x1024 at 54 fps. The power requirements also follow the expected results: using the full implementation of the most complex system on the FPGA the board requires 700 mW, value that increases to 760 mW when the digital camera is connected. The developed board can be employed as a development platform for several systems. The platform is inexpensive and cost-effective, being adequate to be applied also to pedagogical purposes.

TABLE I

Resource occupancy for the several applications.

Modules	**Slice Occupation** *(out of 5408)*		**RAM blocks** *(out of 16)*	**Multipliers** *(out of 16)*	**fps**
Histogram Calculation	54	1%	1	0	66*
Passive Motion Sensing	379	7%	4	2	71k**
Visual Encoding System	1298	24%	15	12	75k**

* image resolution of 1280x1024
** image resolution of 32x32

References

[1] A. M. Reza, "Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for Real-Time Image Enhancement," *Journal of VLSI Signal Processing*, vol. 38, pp. 35 – 44, August 2004.

[2] A. Teuner, M. Hillebrand, B. J. Hosticka, S.-B. Park, J. E. S. Conde, , and N. Stevanovic, "Surveillance Sensor Systems Using CMOS Imagers." 10th International Conference on Image Analysis and Processing, September 1999.

[3] "Cortical Visual Neuroprosthesis for the Blind - CORTIVIS," http://cortivis.umh.es.

[4] Digilent, Inc., "S3BOARD - Spartan-3 Starter Board," http://www.digilentinc.com/info/S3BOARD.cfm.

[5] ——, "XUP V2P - Virtex-II Pro Development System," http://www.digilentinc.com/info/XUPV2P.cfm.

[6] "Video Processing Board," http://sips.inesc-id.pt/videodevkit.

[7] OmniVision Technologies, Inc., "OmniVision Product Families," http://www.ovt.com/i_products.html.

[8] Digilent, Inc., "Accessory Boards and Peripheral Modules," http://www.digilentinc.com/products/Accessory.cfm.

[9] Xilinx, Inc., "Spartan-3 FPGA Family: Complete Data Sheet," http://direct.xilinx.com/bvdocs/publications/ds099.pdf.

[10] OmniVision Technologies, Inc., "The Serial Camera Control Bus Functional Specifications," http://www.ovt.com/pdfs/ds note.pdf.

[11] Philips Semiconductors, "THE I2C-BUS SPECIFICATION," www.semiconductors.philips.com/acrobat/literature/9398/39340011.pdf.

[12] Video Electronics Standads Association, "VGA Standard," http://www.vesa.org/.

[13] R. Gonzalez and R. Woods, *Digital Image Processing*. Addison-Wesley Publishing Company, 1993.

[14] J. E. S. Conde, A. Teuner, and B. J. Hosticka, "Hierarchical locally adaptive multigrid motion estimation using CMOS imagers," *ICASSP '99. Proceedings*, vol. 6, pp. 3365–3368, March 1999.

[15] L. Sousa, P. Tomás, F. Pelayo, A. Martinez, C. A. Morillas, and S. Romero, *An FPL Bioinspired Visual Encoding System to Stimulate Cortical Neurons in Real-Time*. Springer-Verlag GmbH, September 2003, pp. 691 – 700.

[16] M. Meister and M. J. B. II, "The Neural Code of the Retina," *Neuron*, vol. 22, pp. 435–450, March 1999.