Open Source Digital Camera on Field Programmable Gate Arrays

Cristinel Ababei, Shaun Duerr, Joe Ebel, Russell Marineau, Milad Ghorbani Moghaddam, and Tanzania Sewell

Dept. of Electrical and Computer Engineering, Marquette University, WI, USA

ABSTRACT

We present an open source digital camera implemented on a field programmable gate array (FPGA). The camera functionality is completely described in VHDL and tested on the DE2-115 educational FPGA board. Some of the current features of the camera include video mode at 30 fps, storage of taken snapshots into SDRAM memories, and grayscale and edge detection filters. The main contributions of this project include 1) the actual system level design of the camera, tested and verified on an actual FPGA chip, and 2) the public release of the entire implementation including source code and documentation. While the proposed camera is far from being able to compete with commercial offerings, it can serve as a framework to test new research ideas related to digital camera systems, image processing, computer vision, etc., as well as an educational platform for advanced digital design with VHDL and FPGAs. As examples of that, we report two spin-off projects developed on top of or starting from the presented digital camera system.

Keywords: Field programmable gate arrays; digital camera; grayscale filter; Sobel operator; edge detection; face detection.

INTRODUCTION

Field programmable gate arrays (FPGAs) have become extremely popular in virtually all application domains. If in the early days FPGAs were used mostly as digital glue logic or for prototyping purposes, today they are used as integral parts of complex designs ranging from consumer electronics to communications, military, and space systems. The popularity of FPGAs has continuously increased not only because of the reduced performance gap between FPGAs and ASICs but also because of the great flexibility that reconfiguration offers when it comes to product development, maintenance and updates. Today, FPGAs represent the hardware platform of choice to implement and test digital designs for many circuit designers and educators. These include also contributors to online design resources such as Opencores (Opencores, 2016) who test and validate their open source designs on various FPGA chips. Testing and validation on real FPGA chips increase the credibility and confidence in the correctness of these publicly available design resources.

In this paper, we present an open source baremetal digital camera completely described in VHDL and tested and validated on an FPGA chip. While one can find portions of the presented camera design publicly available, we are not aware of any publicly available, self-contained open source design that integrates the same functionality presented here. We make our implementation publicly available (including complete source code and documentation) with the hope that it will

serve as a framework to test new research ideas related to digital camera systems, image processing, computer vision, etc., as well as an educational platform for advanced digital design with VHDL and FPGAs. Note that an early version of this paper was reported as a conference paper in (Ababei et al., 2016). Here, we report two additional projects, which were developed starting from the digital camera project.

The remainder of the presentation is structured as follows. First, we present details about the current implementation of the digital camera system. Then, we discuss experimental results achieved with an actual FPGA board. Spin-off projects developed on top of or starting from the framework provided by the digital camera design are then reported. We then discuss the main merits of this paper as well as future work. Finally, we conclude our presentation in the last section.

BAREMETAL DIGITAL CAMERA SYSTEM

In this section, we present the proposed baremetal digital camera system. We discuss some of the design decisions and the main features.

Block Diagram

The simplified block diagram of the proposed baremetal digital camera system on FPGA is shown in Figure 1. The main functionality of the camera includes video mode at 30 fps, the ability to take a snapshot and store it on the SDRAM memory or on the SD Card, the ability to fetch a snapshot from the SDRAM or SD Card and display it on the VGA display, grayscale filtering, and edge detection based on Sobel operator.

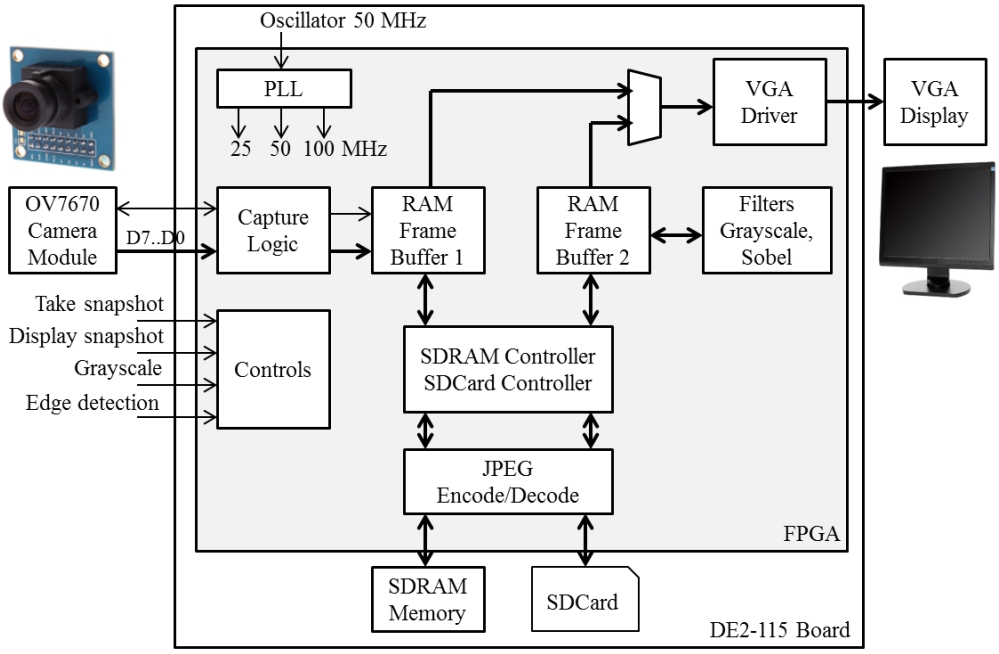


Figure 1: Block diagram of the proposed baremetal digital camera system.

Next, we present details about each of the main features shown in the block diagram in Figure 1.

Video mode feature. This is the simplest operation mode of the digital camera. In this mode, the video stream coming out of the OV7670 CMOS camera module (OV7670, 2016) is buffered inside the *RAM Frame Buffer 1* at a rate of 30 fps and a default frame size of 320x240 pixels. This buffer, like the *RAM Frame Buffer 2* in Figure 1, is implemented using existing RAM memory blocks in the Cyclone IV E FPGA. These buffers can be written and read asynchronously at different write and read clocks.

Each frame is used to drive, via the *VGA Driver*, the VGA display connected to the DE2-115 evaluation board. Writing and reading from the *RAM Frame Buffer 1* is completed with a 25 MHz clock generated by one of the four PLL blocks within the Cyclone IV E FPGA chip. Thus, the *Capture Logic*, *RAM Frame Buffer 1*, and the *VGA Driver* form a datapath that pipelines the video frame data to the VGA display.

The *Capture Logic* block from Figure 1 has the ability to program the camera module by writing into its registers desired values to control image format, color saturation, hue, and gamma. This is done through an I2C like interface connecting the camera module and the *Capture Logic* block. The camera module is connected using 16 of the GPIO connectors available on the DE2-115 board.

Take and display snapshot feature. Two of the four push buttons of the DE2-115 board are used to control the process of recording and displaying on request a single snapshot. In this case, when the *take-snapshot* button is pressed, a frame from *RAM Frame Buffer 1* is taken and stored inside either SDRAM or SDCard memory. When the *display-snapshot* button is pressed, the frame or image data previously saved is retrieved from SDRAM or SDCard and buffered inside *RAM Frame Buffer 2*. The output of *RAM Frame Buffer 2* is then steered through a multiplexer to the *VGA Driver*. The multiplexer shown in Figure 1 is controlled such that the snapshot is displayed on the VGA display only while the *display-snapshot* button is pressed.

Grayscale filter feature. A third push button of the board is used to apply a grayscale filter to the snapshot that is displayed. When this *grayscale* button is pressed, the image data from *RAM Frame Buffer 2* is read pixel by pixel, processed using the averaging technique, and then written back into the buffer at a clock rate of 25 MHz. The averaging technique calculates the average of three colors and is the simplest one among other grayscale techniques. For a given RGB image, we compute the grayscale value for a pixel with the following expression:

$$Grayscale = \frac{R+G+B}{3} \quad (1)$$

where *R*, *G*, and *B* are the color codes for each of the three color channels red, green, and blue of the given pixel. This filter can be implemented using two adders and a divider circuit that can be implemented using very little FPGA resources.

Edge detection filter feature. The fourth push button of the board is used to apply the popular Sobel operator to the snapshot that is displayed. This effectively implements an edge detection filter, which can be very useful for various realtime computer vision applications.

The implementation of this filter is a bit more involved and requires more FPGA resources. Our implementation is similar to that described in (Moore, Devos, & Stroobandt, 2009; Singh, 2013). It basically uses a shift register (whose length is *2W*, where *W = 320* is the width of the image as number of pixels per row or line) and an approximate calculation of the so called gradient magnitude as illustrated in Figure 2.

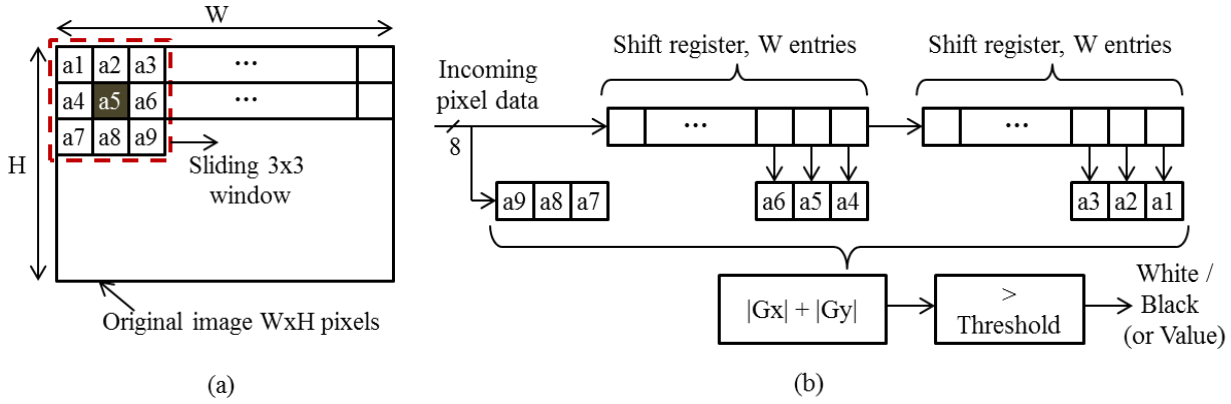


Figure 2: (a) Illustration of the application of Sobel kernels using a sliding window approach that starts with being centered on pixel (0, 0), labeled a1 and ends on pixel (W-1, H-1). (b) The hardware implementation uses two shift registers (Wx8 bits), three simple load registers (Wx3), a comparator, and a circuit to compute $|\boldsymbol{G}_x| + |\boldsymbol{G}_y|$.

The Sobel operator is a discrete differentiation operator, which computes an approximation of the gradient of the image intensity function (Sobel, 2013). Because the Sobel operator is based on convolving the image with a small, separable, and integer valued filter in both horizontal and vertical directions, it is relatively inexpensive in terms of computations - hence easily implemented on an FPGA.

The operator uses two 3x3 kernels, which are convolved with the original image to calculate approximations of the derivatives - for horizontal and vertical changes. If we define **A** as the source image, and $\mathbf{G}_x$ and $\mathbf{G}_y$ are two images which at each point contain the horizontal and vertical derivative approximations, then, the computations are as follows:

$$\boldsymbol{G}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \boldsymbol{A} \tag{2}$$

$$\boldsymbol{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \boldsymbol{A} \tag{3}$$

where * denotes the 2-dimensional convolution operation. The gradient magnitude is given by the following equation:

$$\boldsymbol{G} = \sqrt{\boldsymbol{G}_x^2 + \boldsymbol{G}_y^2} \approx |\boldsymbol{G}_x| + |\boldsymbol{G}_y| \quad (4)$$

The circuit from Figure 2.b generates the value in the above equation for each pixel of the grayscale image and compares it to a threshold (e.g., decimal 127 when working with colors represented on 8 bits per channel; but this threshold can be fine-tuned by the user) and then sets the final result as either white or the value itself (which can also be replaced with black if black and white type edge detection is desired).

Current Status

While the block diagram from Figure 1 includes a JPEG encoder/decoder component, our current implementation (discussed in the next section) does not include this component yet. The implementation of the JPEG encoder/decoder block is under development. Currently, images are saved to memory and fetched from memory as raw pixel data, in an uncompressed format. Note that while the JPEG compression is desirable for the sake of reducing the amount of storage space, working with raw pixel data is preferable in realtime computer vision like image processing applications.

EXPERIMENTAL RESULTS

In this section, we present the experimental setup and results obtained with the current implementation of the digital camera system.

The entire functionality of the camera, as discussed in the previous section, was coded in VHDL, synthesized, placed, and routed with Quartus II Web Edition 14.1 design environment (Quartus II, 2015). The camera design was tested and verified on the DE2-115 development board (DE-115, 2015), which contains Altera's Cyclone IV E FPGA chip (Cyclone IV, 2016).

Relevant key aspects of the summary report provided by Quartus II tool are reported in Table 1. Note that the overall resource utilization is very small at around 1%. This leaves plenty of room for additional functionality that may be added to the system. Also note that, most of the embedded block RAMs existing on the Cyclone IV E FPGA chip are used (79%). This higher memory utilization is due to the two buffers, *RAM Frame Buffer 1* and *RAM Frame Buffer 2* from Figure 1, which are sized such that they can store data for 320x240 pixel images. The design uses one of the four PLLs available on the FPGA chip. The PLL is used to generate four different clock signals with frequencies of 25 MHz, 50 MHz, 100 MHz, and 100 MHz with a clock phase shift of -3 ns. The 50 MHz signal is used by the capture logic block, the 100 MHz signals are required by the SDRAM controller, and the 25 MHz signal is used by most other blocks in the design.

Table 1: Summary of the fitter report as given by Quartus II tool.

Item	Report
Family	Cyclone IV E
Device	EP4CE115F29C7
Total logic elements	1,616 / 114,480 (1%)
Dedicated logic registers	818 / 114,480 (< 1%)
Total registers	818
Total pins	94 / 529 (18%)
Total memory bits	3,152,384 / 3,981,312 (79%)
Total PLLs	1 / 4 (25%)

The experimental set-up from Figure 3 includes the DE2-115 with an OV7670 camera module attached via the GPIO connector and a 640x480 VGA monitor attached via the VGA port available on the board. The four push buttons (already debounced, by design of the DE2-115 board) available on the DE2-115 board are used as *take-snapshot*, *display-snapshot*, *grayscale*, and *edge-detection* buttons to control the system. The default mode of operation is the video mode, which works at 30 fps.

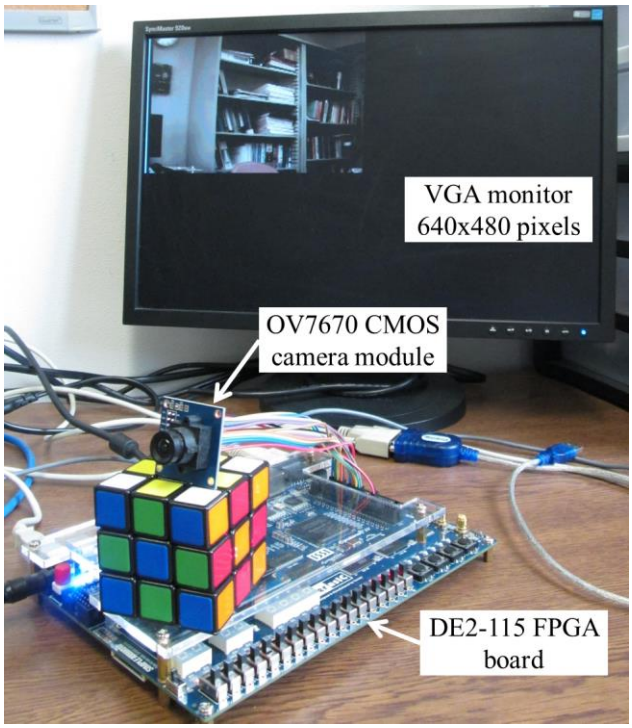
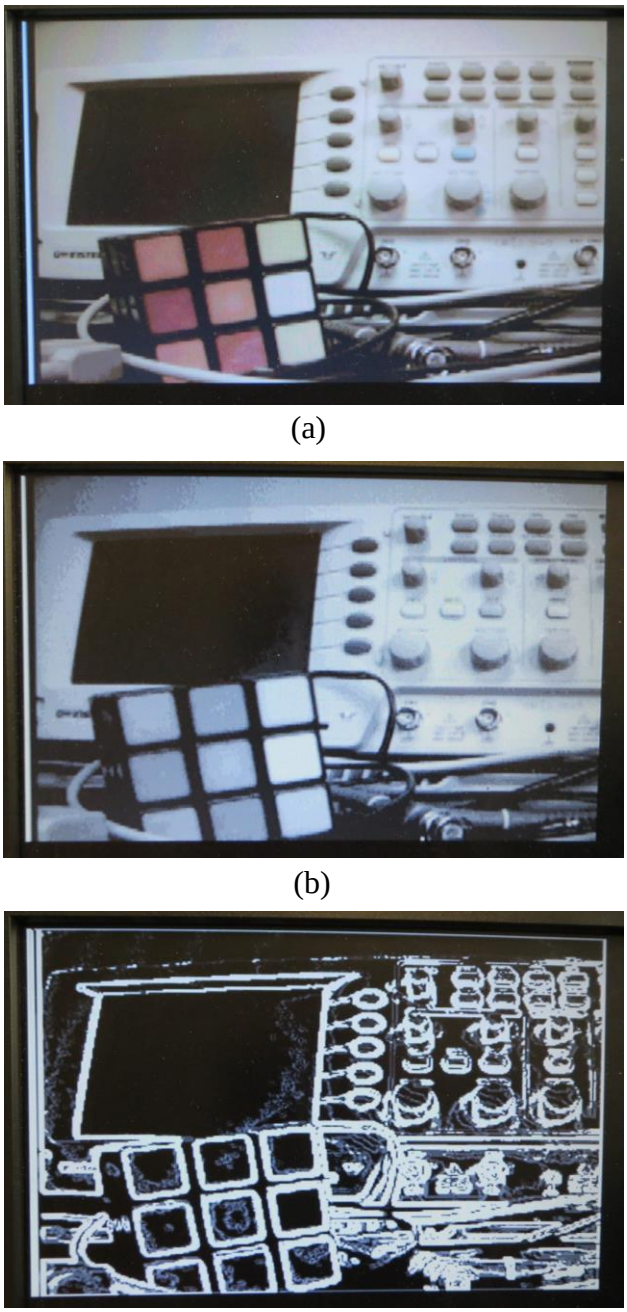


Figure 3: The experimental setup includes the DE2-115 board that has the OV7670 camera module and a VGA monitor attached to it. Images are captured by the camera and displayed on the VGA display as 320x240 pixel images.

Figures 4 and 5 illustrate the operation of the camera system when a snapshot is taken, saved, and displayed (parts (a) of these figures), when the grayscale filter is applied to the snapshot

retrieved from the memory (parts (b)), and when the edge detection (i.e., Sobel) filter is applied (parts (c)).



(a)

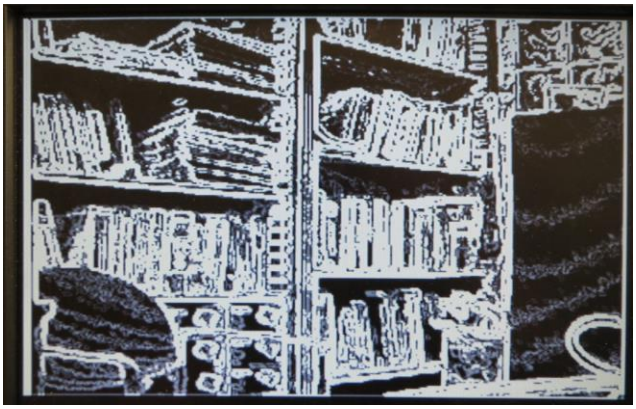
(b)

(c)

Figure 4: (a) Example of taken snapshot; objects in the image are within 30 cm of the camera module. (b) The grayscale filter applied to the snapshot. (c) The edge detection filter applied to the snapshot.

(a)

(b)



(c)

Figure 5: (a) Example of taken snapshot; objects in the image are within 3 m of the camera module. (b) The grayscale filter applied to the snapshot. (c) The edge detection filter applied to the snapshot.

ADDITIONAL PROJECTS DEVELOPED USING THE CAMERA FRAMEWORK

In this section, we describe two extensions that we have worked on as two separate projects forked out from the digital camera project. Both of these projects are also made publicly available as documentation and complete source code.

Edge Detection in Video Mode

In the first project, we modify the camera framework in order to investigate the achievable performance with the implementation of the edge detection algorithm operated in video mode in realtime. Basically, the video stream from the camera is either displayed normally or first processed as grey-filter + edge-detection and displayed on the VGA monitor. The block diagram of this project in Figure 6. Figure 7 shows an operation example of this design. This design is capable of operating at 5 fps on the same FPGA board. The entire Quartus II project files can be downloaded from (MESS Lab, 2016).

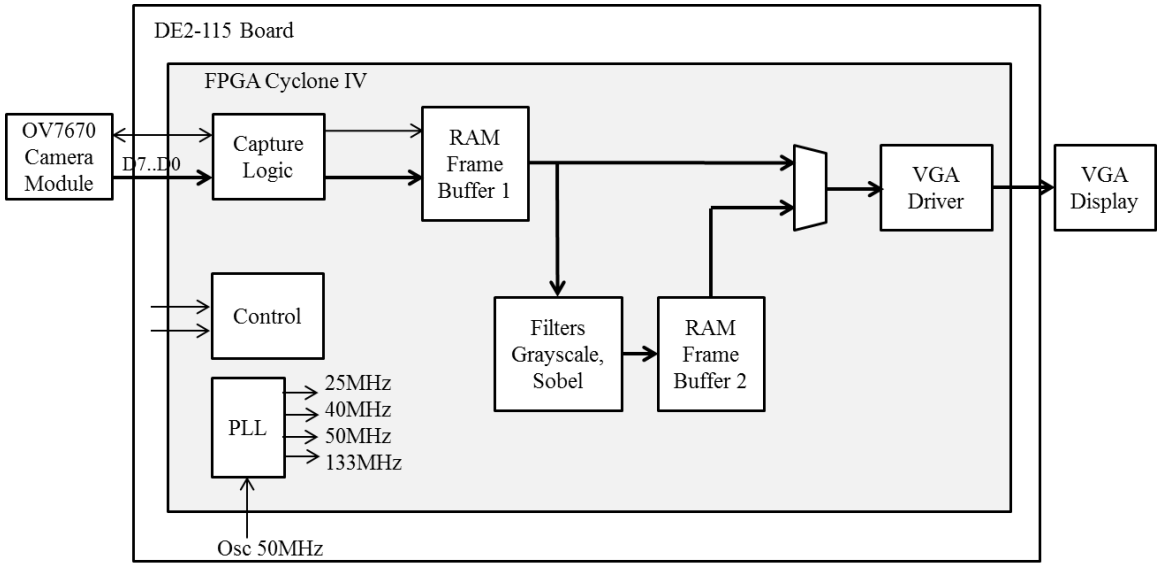
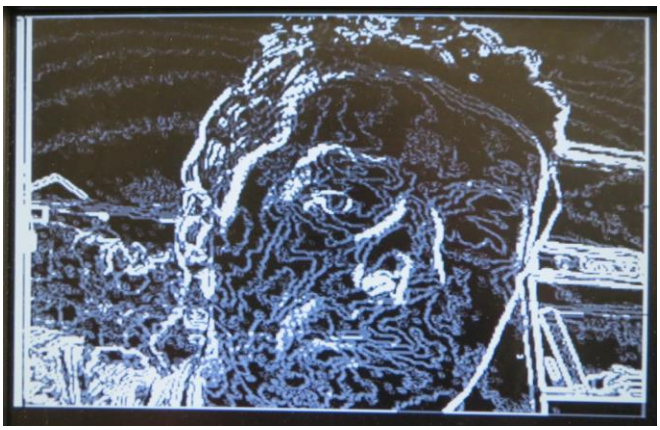


Figure 6: Block diagram of the edge detection system for video mode operation.

(a)



(b)

Figure 7: Example snapshots when the edge detection system is operated in video mode. The normal and edge detection modes can be switched with the help of a push button on the development board.

Face Detection System

In the second project, we start from the framework provided by the digital camera project to develop a complete implementation in VHDL of the popular Viola-Jones (VJ) face detection algorithm (Viola & Jones, 2001). The primary objective of this project is to study the achievable performance with a low-end FPGA chip based implementation.

We found that the face detection design achieves an average performance of 4.4 fps irrespective of the number of faces in the video frame, for tests that we performed with up to ten faces in the frame. Complete description of this project is presented in great details in another paper (Irgens et al., 2016). The block diagram of this project in Figure 8. Figure 9 shows an operation example of this design.

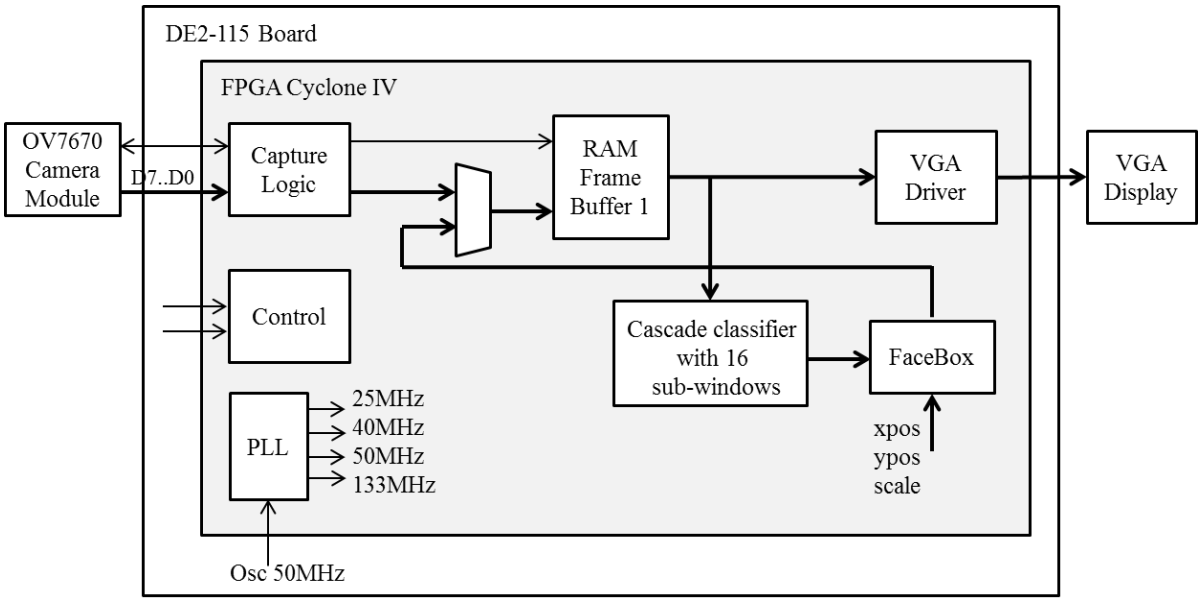


Figure 8: Block diagram of the face detection system.

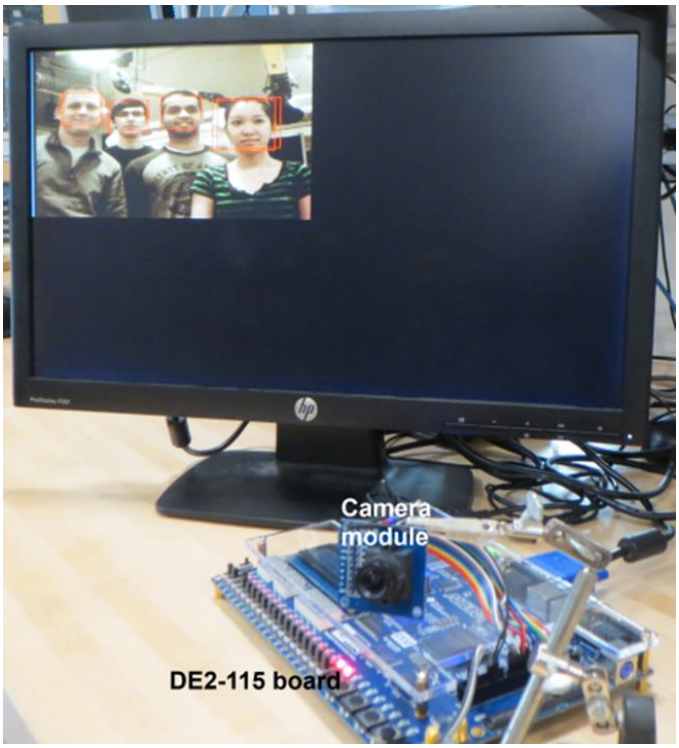


Figure 9: Snapshot of the experimental setup for the face detection design. Images are captured by the camera and displayed on the VGA display as 320x240 pixel images. Detected faces are indicated as red color rectangle indicators superimposed on the video frames.

DISCUSSION AND FUTURE WORK

The proposed baremetal digital camera has been successfully prototyped and tested on real hardware. We consider as its main contributions and merits the following:

- The system level design in VHDL and deployment for testing and verification on a widely popular FPGA board. The actual implementation requires a very small amount of existing resources on the Altera's Cyclone IV E FPGA chip.
- The entire design sources are made publicly available. This includes completed and working VHDL source code as well as detailed documentation. Our hope is that this open source digital camera design will encourage further research and educational ideas in the areas of image processing and advanced VHDL design and FPGAs. The entire project can be downloaded from (MESS Lab, 2016).

As ongoing and future work, we are designing and will manufacture a small PCB board - which will also be made open source - to implement the presented digital camera as a stand-alone yet extendable application. This PCB board is built around the same FPGA chip and will also include one SDRAM memory chip, one SD Card slot, a small Flash memory, a USB driver, a driver for 3.2-inch color TFT 320X240 display, four push buttons, and four LEDs. The footprint of this custom PCB is similar to that of a regular Arduino board.

CONCLUSION

We introduced an open source baremetal digital camera, which was entirely specified in VHDL. The camera system was tested and validated on the popular DE2-115 educational FPGA board, which is constructed around the Cyclone IV FPGA family. The entire design is made publicly available including source code and documentation. We hope that this project will serve as a framework to test new research ideas related to digital camera systems, image processing, computer vision, etc., as well as an educational platform for advanced digital design with VHDL and FPGAs.

ACKOWLEDGMENT

This work was supported by the Department of Electrical and Computer Engineering and the OPUS College of Engineering at Marquette University.

REFERENCES

Ababei, C., Duerr, S., Ebel, J., Marineau, R., Moghaddam, M.G., & Sewell, T. (2016, May). *Open source digital camera on field programmable gate arrays*. Paper presented at IEEE Int. Conf. on Electro Information Technology (EIT), Grand Forks, ND.

Cyclone IV FPGA Family: Lowest Cost, Lowest Power, Integrated Transceivers, (2015). Retrieved September 13, 2016, from http://www.altera.com/devices/fpga/cyclone-iv/cyiv-index.jsp

DE2-115 Development and Education Board, (2015). Retrieved September 13, 2016, from http://www.altera.com/education/univ/materials/boards/de2-115/unv-de2-115-board.html

Irgens, P., Bader, C., Le, T., Saxena, D., & Ababei, C. (2016, under review). *An efficient and cost effective FPGA based implementation of the Viola-Jones face detection algorithm*. Submitted for consideration of publication.

MESS Lab, Open Source HW Projects, Marquette University, (2016). Retrieved September 13, 2016, from http://dejazzer.com/hardware.html

Moore, C., Devos, D., & Stroobandt, D. (2009). *Optimizing the FPGA memory design for a Sobel edge detector*. Paper presented at Int. Conf. on Engineering of Reconfigurable Systems and Algorithms (ERSA).

Opencores - community for development of hardware IP cores as open source, (2016). Retrieved September 13, 2016, from http://opencores.org

OV7670 Camera Module with OmniVision CMOS Sensor, Datasheet, (2015). Retrieved September 13, 2016, from http://www.cutedigi.com/pub/sensor/Imaging/OV7670-Datasheet.pdf

Quartus II Web Edition Software 14.1, (2015). Retrieved September 13, 2016, from http://www.altera.com/products/software/quartus-ii/web-edition/qts-we-index.html

Singh, S., Saini, A.K., Saini, R., Mandal, A.S., Shekhar, C., & Vohra, A. (2013). *Area optimized FPGA-based implementation of the Sobel compass edge detector*. Hindawi ISRN Machine Vision.

Sobel, I. (2014), *History and Definition of the Sobel Operator*. Retrieved September 13, 2016, https://www.researchgate.net/publication/239398674_An_Isotropic_3_3_Image_Gradient_Operator

Viola, P.A., & Jones, M.J. (2001). *Rapid object detection using a boosted cascade of simple features*. Paper presented at IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR).