EECE 277 FPGA DESIGN

Fall 2014 Final Project Proposal

Video processing controller for a CMOS camera

Marco Beccani, Trevor Bruns

1. Overview

- **PURPOSE:** The aim of this project is to implement an FPGA based CMOS camera controller in VHDL using the ALTERA Cyclone IV DE2-115 Development board.
- **GOALS:** The main goal of this project is to display the image sensor data in real time on an external display using a standard VGA interface.
- **BENEFITS:** An FPGA architecture can achieve faster data rates when compared to traditional micro-controlled systems. VHDL hardware resources, such as frame buffers, can be implemented with the FPGA in order to achieve image data transmission in real time.

2. Objectives

The proposed platform consists of a CMOS camera (OV7670, Omnivision, USA), the CMOS video processing controller, a VGA monitor and the user interface. Images acquired from the camera are handled by the CMOS Video Processing Controller and displayed on the VGA monitor in real time. The user interface (e.g keyboard/buttons/switches) will allow the user to access and change some of the camera settings.

Frame Rate	20 fps
Camera Display Resolution	QVGA, 320 x 240 pixels VGA 640 x 480 pixels.
Color Output Format	RGB 4 bit , Grayscale 4 bits.

Table 1 - General Specification of the CMOS Video Processing Controller.

3. Resources

The members working on this project are Trevor Bruns and Marco Beccani. Each is able to devote, on average, about 5-10 hours per week. Project components are listed in Table 2.

Component Name	Lead Time	Price ($)
VGA Camera Module OV7670	1 week	$20
DE2-115	Provided	$309 (academic)
Quartus II and Modelsim	Instant Download	$0 (web edition)
Oscilloscope	On hand	N/A
Saleae Logic 8 (logic analyzer)	On hand	$199

Table 2- Component list.

4. Background Information and Motivation

Interfacing directly with even a low resolution CMOS camera can be highly resource intensive. If a microcontroller is used, the main limitation is usually speed (i.e. frame rate). For instance, an Arduino Uno, with a 16 MHz clock speed, can barely manage 1 FPS with optimized code [1]. However, even with a much faster processor there can still be a bottleneck in terms of memory storage and transfer, as many microcontrollers do not contain much on-board memory. Furthermore, the processor will likely not have much time to run other instructions. For these reasons, an FPGA is well-suited for this task. In addition, the FPGA will have likely have abundant resources left to perform additional tasks such as image processing. For instance, previous research in this area has been applied to process image data from high-speed cameras (1000 FPS) in order to track objects in real time and respond to them via robotic interfaces (e.g. perfectly swing a bat at a random pitch) [2]. Our current research interests lie in the medical robotics field. In particular, we are striving to reduce the invasiveness of many procedures (e.g. laparoscopic). Since the surgeon no longer has direct field of view, cameras are used for visualization. It might be useful to acquire and process the video stream in real-time without the need for a bulky setup (i.e. a large cart in the operating room full of equipment). The potential to dramatically decrease the size of the system (space in the OR is precious) while also increasing the functionality is our driving motivation for developing an FPGA-based system.

5. Strategy

Initially, we will begin by mapping out the main modules of the overall design. From there we will determine what the input and output should be for each. Data structures will be created so that each module handles the data in a consistent manner. Once complete, simulation block will be created to create mock data for

testing purposes. First, the VGA output component will be tested to ensure that, if given valid data, it will display properly. Next, the camera will be configured with default/conservative settings (e.g. low resolution/frame rate) and we will attempt to retrieve image data and display it. Afterwards, settings will be optimized to produce the best image possible (largest, fastest, most accurate colors, etc.). Next, simulated user input will be used to modify some video settings before interfacing eventually with the camera and integrating an actual keyboard.

6. Details of the proposal

The video processing controller architecture is represented in Fig.1 with its components. In particular the camera interface handles the camera configuration and frame capture. Raw image data are stored into a dedicated memory frame buffer. A VGA controller guarantees VGA timings and feeds the external VGA display with the memory content. Finally, the user interface controller is responsible for capturing user inputs (e.g buttons, switches, keyboard) in order to change camera settings as well as control other visualization interfaces (e.g 7 segments and LCD).

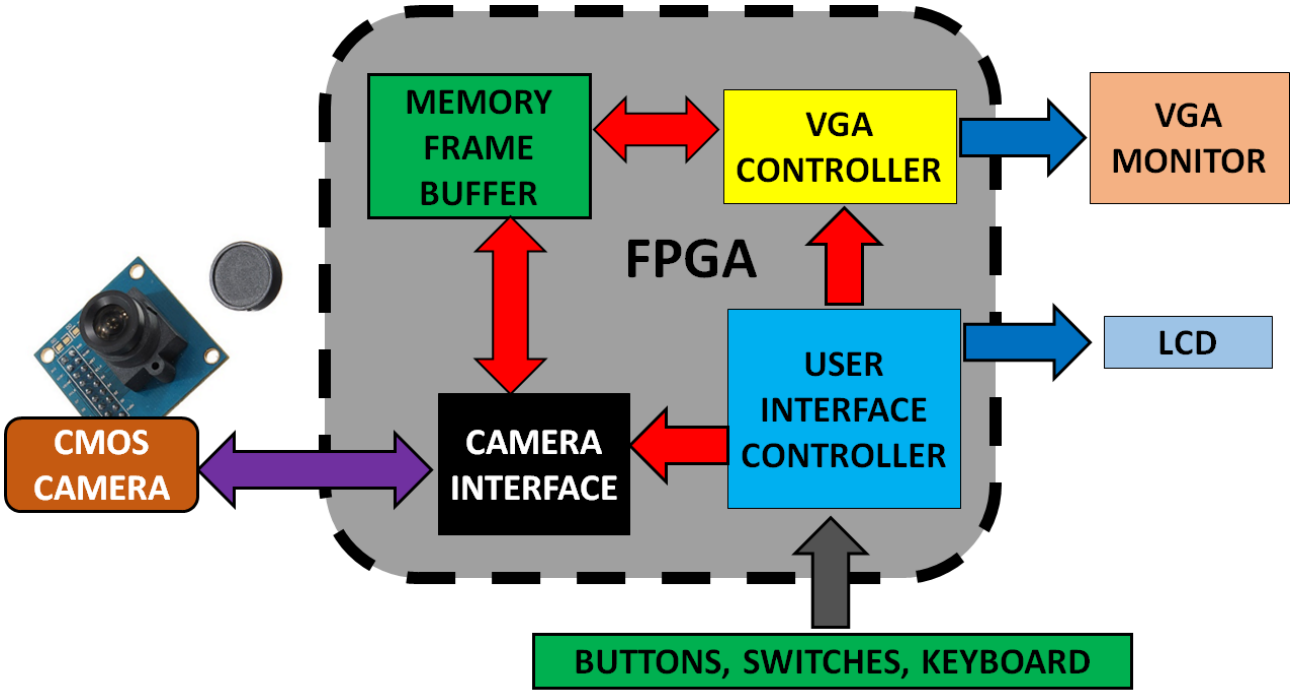

Fig.1- System architecture with its main functional modules

7. Action Plan

Timeline with items done in parallel (**Black**) and individual tasks (Marco- **Red**, Trevor- **Blue**).

Week of	Goal
11/3	**Finalize background research (understand relevant datasheets, schematics, etc.)** **Rough outline code for each module**. **Order remaining components**
11/10	**Complete VGA display code using memory** **Successfully display simulated camera data on a display via VGA** **Wire camera to DE2-115 GPIO pins** **Begin writing code for configuring camera settings (via SBCC)**
11/17	**Verify configuration communication with camera using a logic analyzer** **Determine simplest camera parameters and initialize the camera with those** **Finish code for retrieving image data from camera** **Display camera data on display (non optimized settings)**
11/24	**Incrementally work towards optimizing camera settings for resolution, frame rate, and color quality Finish code for simulating user input from a keyboard to change camera settings during operation. Test that settings can be changed during operation in real-time**
12/3	**Finish code for interfacing with a keyboard** **Verify that entire system is operating as expected** **Clean up/finalize all code**
12/4	**Demonstrate working project to instructor**
12/8	**Compile final report** **Create final presentation**

Table 3 - Action Plan.

8. Assumption/ Risks

Distributors for the required components have been located to ensure timely delivery, thus this should not be a cause of concern. The primary risks we face will be due to time constraints. As we are both PhD students, individual research and paper deadlines have the potential to severely limit our time commitment. At the very least, it is likely we will not be able to work on a consistent schedule. This will require adequate time management to ensure that work is completed ahead of time to prevent falling behind the proposed schedule.

In addition, there is some concern about implementing the SBCC protocol for configuring the camera. This is similar to the I^2C protocol, which both of us have experience using with microcontrollers, but could be difficult to debug. We hope to find existing code, which we can modify, to simplify and speed up this

process. Sensor interfaces have a multitude of settings. Incorrect or incompatible settings might result in unusable data. Furthermore, displaying real-time image data can be tricky due to memory and timing synchronization issues.

9. Deliverables

- Video stream acquired from user-configurable camera and displayed in real-time via VGA
- Fully documented code
- Final report which will include full system description and analysis

10. References

[1] On line, available at:http://forum.arduino.cc/index.php?topic=157903.0

[2]On line, available at: http://www.k2.t.u-tokyo.ac.jp