# DESIGN AND DEVELOPMENT OF CAMERA INTERFACE CONTROLLER WITH VIDEO PRE-PROCESSING MODULES ON FPGA FOR MAVS

**O. Ranganathan [1], *Abdul Imran Rasheed[2]**

1- M.Sc [Engg.] student, 2-Assistant Professor
Department of Electronics and Electrical Engineering
M. S. Ramaiah School of Advanced Studies, Bangalore-58.
*imran@msrsas.org

## *Abstract*

*Over the past few years the idea of micro sized aerial vehicles shortly known as Micro Air Vehicles (MAVs) has gained mounting interest. These MAVs while flying acquire terrain images through their on board cameras and transmit the images for interpretation to base station. The onboard cameras acquire images and these images are processed and results are used for locating the next position and also the velocity with which the MAV should reach to its next position. The flying direction and speed of the MAV depends on the speed with which the images are acquired and processed. The time taken to acquire image is limited by camera frame speed, the processor speed and capability, which is observed to be greater than 30 ms from the literature. Hence the acquisition of real-time images in MAVs plays a major role in its overall functionality and performance which necessitates the dedicated camera interface module with high processing speed.*

*The current paper deals with the design of a new camera interface controller that has high processing speed. The first focus of the design is on developing the input interface controller module for interfacing CMOS sensor with FPGA module for converting the raw data into RGB pixel information and the output interface controller. The second focus is to synchronize all the modules of the design for proper streaming of video. A 5 MP CMOS sensor having a resolution of 2592 column by 1944 row with the frame rate of 15 fps is chosen. The captured information from the sensor is then converted into RGB pattern and processed for streaming in VGA display.*

*The controller is synthesized using Altera Quartus II v 12.0 and implemented on Cyclone IV E FPGA board from Altera. The synthesis of the design reports, the total number of logic elements utilized by the design as 1739 with 1062 logic registers, total number of pins configured with I/Os of the design as 455 with 44752 memory bits and one PLL circuitry utilization. The power analysis reports the total power dissipation of the design as 328.29 mW. The total acquisition time of the image from the sensor to the display is 28 ms which is less than 30 ms that makes this controller suitable for MAV applications. Further low power ASIC synthesis and implementation of the controller can be opted as a consideration for high power consumption and payload implications on MAVs*

**Key Words:** CMOS sensor interface, Camera interface controller, MAV, Camera interface with FPGA

**Nomenclature**

| | | |
|---|---|---|
| fps | – | Frames Per Second |
| Hz | – | Hertz |
| mW | – | Milli Watt |
| ms | – | Milli Second |

**Abbreviations**

| | | |
|---|---|---|
| ADC | - | Analog to Digital Converter |
| ASIC | - | Application Specific Integrated Circuit |
| CMOS | - | Complementary Metal Oxide Semiconductor |
| FPGA | - | Field Programmable Gate Array |
| HDL | - | Hardware Description Language |
| HSMC | - | High Speed Mezzanine Card |
| LCD | - | Liquid Crystal Display |
| MAV | - | Micro-Air Vehicle |
| PLL | - | Phase Locked Loop |
| RAM | - | Random Access Memory |
| RAW | - | Raw Image Data |
| RGB | - | Red, Green and Blue |
| RTL | - | Register Transfer Logic |
| TFT | - | Thin Film Transistor |

## 1. INTRODUCTION

Over the past few years the idea of micro sized aerial vehicles shortly known as Micro Air Vehicles (MAVs) has gained mounting interest. The primary loads of these MAVs (~15 centimeters or 6 inches wingspan) are miniature image sensors [1]. These MAVs are used in acquiring real-time visual information with its image sensor for wide range of applications like reconnaissance and surveillance, targeting, tagging and bio-chemical sensing. The MAVs while flying acquire terrain images through their on board cameras and transmit the images for interpretation to base station. Most of the MAVs have vision based navigation system. The onboard cameras acquire images and these images are processed as per a particular algorithm and results are used for locating the next position and also the velocity with which the MAV should reach to its next position. The flying direction and speed of the MAV depends on the speed with which the images are acquired and processed. Hence the acquisition of real-time images in MAVs plays a major role in its overall functionality and performance. The time taken to acquire image is limited by camera frame speed and the processing of images depend on processor capability and operating speed. At present the processors which are being used for such application take more than a second to acquire and process the data thus limiting the flying speed of MAV. Normally MAVs fly at 10-12m/s and for stable flying of MAVs at such a speed, the images are required to be acquired in less than 30ms. Hence it is necessary to develop fast processors, if not algorithms have to be parallelized and executed. Any development in FPGA technology or ASIC technology has its own merits and demerits which impose great challenges to design engineers. Prototyping of the controller is made in the FPGA board.

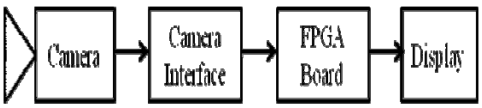


**Fig. 1 Basic flow of camera interface with FPGA**

The camera has to be interfaced with the FPGA board for which the camera interface module has to be developed as shown in Figure 1. With the help of camera interface module the camera is interfaced with the FPGA board for data capturing and streaming it in the display unit. Three major functions of the camera interface controller are capturing of information from the camera, storing the pixel information in the RAM of FPGA and streaming them to the output.

Most of the application that requires camera interface needs additional circuitry to interface the camera with FPGA [3]. Video pre-processing blocks like colour space conversion, scaling and smoothing provides high pixel processing rate, high performance in low cost FPGA and high throughput. Synchronization issues in streaming the video limits the frame rate of video [2]. Any camera interface controller designed for the particular application is based on the speed of the camera. For example interface controllers, which are designed for 2 Mbps speed camera, do not function with 4 Mbps camera [3]. Hence a separate interface controller has to be developed matching the device and camera speed. This necessitates the need of generalized controller for the camera interface with the hardware. Further there are no video pre-processing is made during the streaming of video that obviously helps in high pixel processing rate, high performance in low cost FPGA and high throughput [4]. This paper deals mainly with the design of a new camera interface controller which can able to acquire the real-time images and stream the video to the display.

## 2. PROBLEM DEFINITION

Existing controllers have higher video acquisition time and are not generic. In this paper, a controller has been designed with faster data acquisition speed which is generic to the cameras but specific to the FPGA board and the interface port.

## 3. METHODOLOGY

Identification of suitable pre-processing blocks for FPGA implementation is done based on the literature survey. The complete blocks that makeup the controller functionality and meeting the requirements have been designed. RTL codes are developed using Verilog HDL for the blocks of camera interface controller module along with pre-processing modules. The functionality of the developed RTL is verified using Verilog based testbenches. The design is synthesized using Altera Quartus II tool to check for compatibility with hardware, simulated and synthesized RTL code is implemented on Cyclone IV E FPGA.

## 4. DESIGN AND IMPLEMENTATION

Development of camera interface module with the FPGA board is done using the HSMC port [5] available in the FPGA board. The display unit is also interfaced to the FPGA to display the real-time video captured using the camera that is interfaced with the FPGA board.

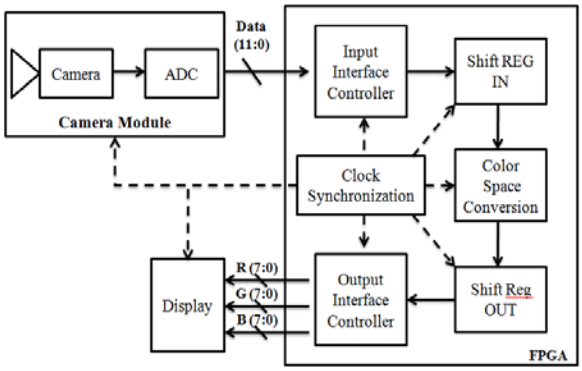


**Fig. 2 Blocks of camera interface controller**

Figure 2 shows the proposed block diagram for the camera interface with the FPGA. The design has two interface controllers. First is the input interface controller where the camera is interfaced with the HSMC port of the FPGA and the second is the output interface controller where the LCD screen is interfaced with the FPGA.

### 4.1 HDL Modeling of Input Interface Controller

The input interface controller is the main block which interacts with the camera module. The input interface controller module with all its input and output signals communicating with other modules of the design is shown in Figure 3. It communicates with the camera module from which it captures the real-time video using TRDB_D5M CMOS sensor [6] [7] and other modules as in Figure 3. Signals PIXLCLK of 96 MHz and Reset are from the synchronization circuit.

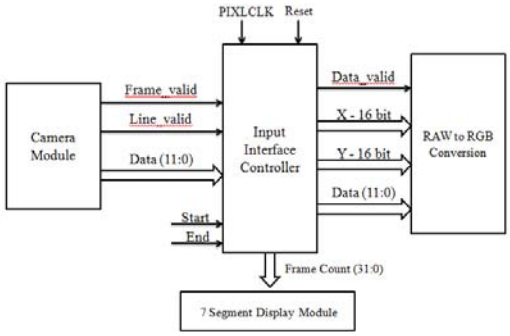


**Fig. 3 Input interface controller**

The camera module outputs frame_valid and line_valid are used to represent the valid data coming out of the camera module. The input interface controller reads the 12 bit data from the camera module only when both frame_valid and line_valid are high. A counter is designed as the part of the input interface controller and counts the number of frames read per PIXCLK and outputs as 32 bit hexadecimal signal named Frame_Count to the 7 segment display module.

The counter designed for counting the frame rate uses the Start and End signals coming from the control logic designed. In correspondence with the frame_valid and line_valid signals from the camera module, the input interface controller produces data_valid signal as the output with the corresponding the X and Y coordinates of the pixel. All the signals data_valid, X, Y and the 12 bit data from the input interface controller are sent to RAW to RGB converter.

### 4.2 HDL Modeling of RAW to RGB Converter

Verilog HDL modeling of RAW to RGB conversion [8] [9] involves two steps. First the input data coming from the input interface controller is passed on to a line buffer which performs the operation of shifting. The input data is buffered and shifted out along with one more signal of required format called Tap_signal. Signals are tapped in required format for conversion which produces two tap signals each of which is 12 bit and together a 24 bit tap signal. These tap signals are named wData0 and wData1. These signals are used in converting the RAW to RGB format in the second step. The conversion taking place depends on the X and Y values coming from the input interface controller as in Figure 4.

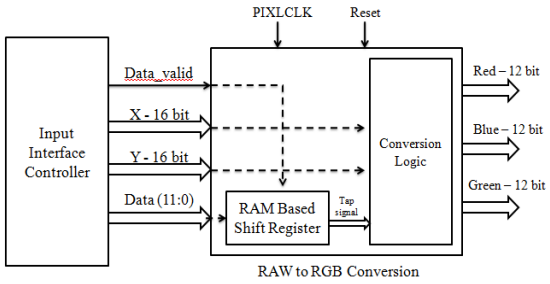


**Fig. 4 RAW to RGB converter**

After conversion this module produces three output signals called Red, Blue and Green each of which is 12 bit which is again given to the shift registers. They are then shifted to a control unit where 12 bit of RGB gets encoded into two data signals each of 16 bit before sending it to the display module.

### 4.3 HDL Modeling of Seven Segment Display

In seven segment display, appropriate combination of segments is used to represent decimal numerals. For a decimal numeral to be displayed its corresponding segments gets asserted and it gets displayed.

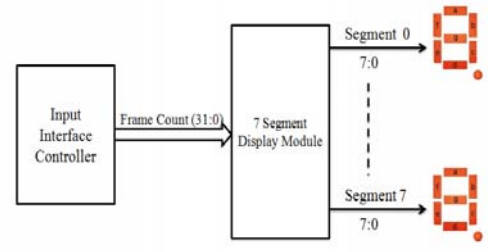


**Fig. 5 Seven segment display module**

For displaying the decimal numeral 1, segments b and c are made high and the corresponding signal will be of 7 bit 0000110 (gfedcba). This logic is used for modelling the seven segment display module.

### 4.4 HDL Modeling of Output Interface Controller

The output of the RAW to RGB conversion is taken to the control logic block where the 12 bit data of R, G and B values are encoded into two datas of 16 bit, Data1 and Data2. This is done so in order to synchronize the data between the modules.

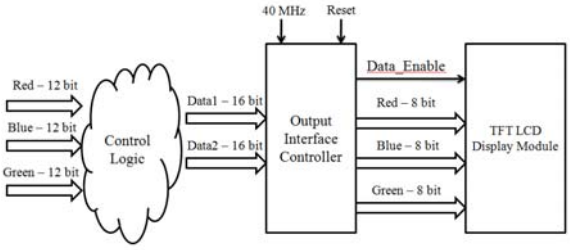


**Fig. 6 Output interface controller for TFT LCD display**

The display works at 40 MHz frequency and hence 40 MHz clock from the PLL module is given as the clock input to the output interface controller. The Data1 and Data2 are the inputs to the output interface controller module which then synchronize the data producing 8 bits of R, G and B values which is given to the display. The RGB values are configured to the FPGA according to the corresponding pin descriptions of the HSMC port for LCD display.

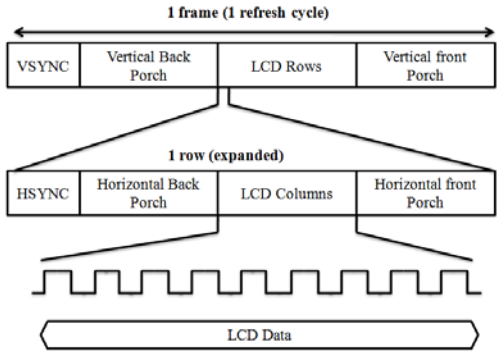


**Fig. 7 LCD timing and line timing parameters [11]**

Figure 7 shows the frame timing and line timing parameters required for modelling LCD interface controller. Horizontal back porch is the number of pixel clock cycles to be inserted at beginning of each line/row of pixels. Horizontal front porch is the number of pixel clock cycles to be inserted at end of each line/row of pixels. Vertical back porch is the number of pixel clock cycles to be inserted at beginning of each frame/column of pixels. Vertical front porch is the number of pixel clock cycles to be inserted at end of each frame/column of pixels. The front porch, back porch and LCD rows and columns are parameterized according to the LCD specification in the HDL model in designing the LCD controller. According to the frame and line timing the controller produces the valid display at the LCD.

## 5. SIMULATION RESULTS

### 5.1 Simulation of Input Interface Controller

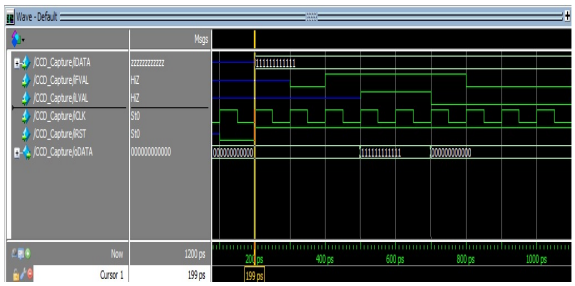

**Fig. 8 Input data read-out by input interface controller**

The camera module outputs frame_valid and line_valid are used to represent the valid data coming out of the camera module which are the input signals of the input interface controller. The input interface controller reads the 12 bit data from the camera module only when both frame_valid and line_valid are high as shown in Figure 8. At 200ps considering some junk data is present. The signals frame_valid and line_valid indicates whether the available data is valid or not. So the data available at this time period is not being read by the input interface controller. At the period of 500 ps both frame_valid and line_valid are high and hence the data is read at the output signal oDATA. Again when the line_valid goes low at 700 ps the data is invalid and so the oDATA reaches its initial state. This explains the input data readout conditions.

### 5.2 Simulation of RAW to RGB Converter

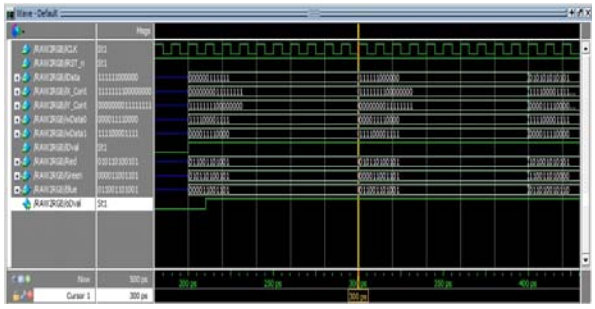

**Fig. 9 Generated RGB values from input data**

Figure 9 shows the simulation result of RAW to RGB conversion module. As shown in Figure 9, 12 bit RGB datas are generated from the input pixel information.

### 5.3 Simulation of Seven Segment Display

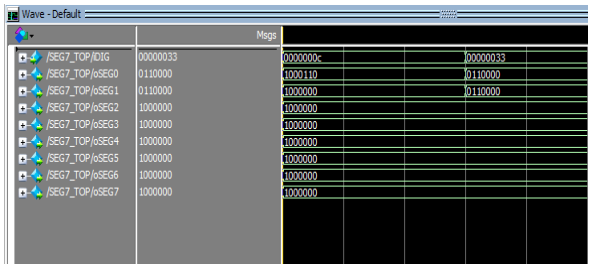

**Fig. 10 Output of seven segment display**

The design of seven segment display outputs the data in active low format. When the signal is low the segment of the display glows. In Figure 10 first input is '0000000C' which has its corresponding output as 1000110 in the first segment. Remaining segment displays the value 0.

### 5.4 Simulation of LCD Display Module

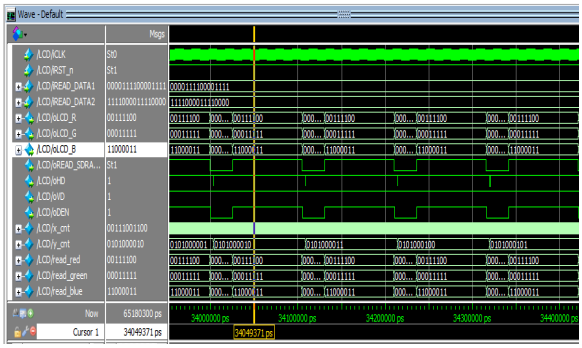

**Fig. 11 Simulation result of LCD module**

Figure 11 shows that for the input data of READ_DATA1, READ_DATA2 and the synchronizing clock signal the corresponding R, G and B component that need to be displayed on the LCD are generated.

## 6. RESULTS AND DISCUSSIONS

### 6.1 Synthesis Report

The synthesis of the design is made using Quartus II v 12.0 and the reports are given in Table 1.

**Table 1. Synthesis summary**

| Resources | Usage |
|---|---|
| Estimated Total Logic Elements | 1739 |
| Total Combinational Functions | 1356 |
| Logic elements usage by number of LUT inputs | |
| 4 input functions | 591 |
| 3 input functions | 285 |
| <= 2 input functions | 480 |
| Logic elements by mode | |
| Normal mode | 1010 |
| Arithmetic mode | 346 |
| Total Registers | |
| Dedicated logic registers | 1062 |
| I/O registers | 0 |
| I/O Pins | 455 |
| Total Memory Bits | 44752 |
| Total PLLs | 1 |
| Maximum Fan-out | 681 |
| Total Fan-out | 10003 |
| Average Fan-out | 2.85 |

### 6.2 Power Analysis Report

**Table 2. Power play power analyzer results**

| | |
|---|---|
| Total Thermal Power Dissipation | 328.29 mW |
| Core Dynamic Thermal Power Dissipation | 43.30 mW |
| Core Static Thermal Power Dissipation | 102.39 mW |
| I/O Thermal Power Dissipation | 182.60 W |

As per the power analysis report shown in Table 2 the total thermal power dissipation is 328.29 mW which constitutes 43.3 mW of dynamic thermal power dissipation, 102.39 mW of static thermal power dissipation and 182.6 mW of I/O thermal power dissipation.

### 6.3 FPGA Implementation

The bit file is generated and configured to the target device. Once the board is configured the sensor starts capturing the real time video and displays it the LCD module interfaced with the FPGA board.

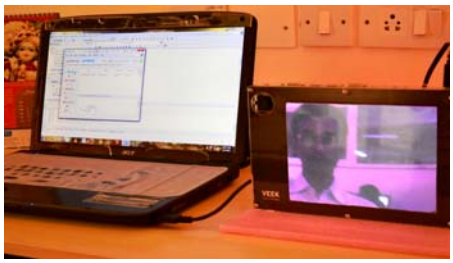

**Fig. 12 LCD display output of camera interface controller**

USB Blaster is used to program the FPGA board with the generated bit file. Once the board is configured, the sensor captures the real-time video and displays it in the LCD module.

**Fig. 13 Seven segment display output**

Figure 13 shows the frame count values in the seven segment display of the Cyclone IV E FPGA board.

**Fig. 14 VGA display output**

The video streaming is made in VGA display also. The controller generates the HSYNCH, VSYNCH and corresponding RGB signals required for the VGA display.

Table 3 indicates the time taken by individual module for processing the data. Input interface module takes around 6.8 ms for capturing the data from the camera according to frame_valid and line_valid signals and to process the information of X count and Y count. The data from the input interface controller is then converted into RGB format which takes around 12.6 ms and the output interface controller takes 6.2 ms for processing of data suitable for display. Other resources all together constitutes to a period of around 2.4 ms.

**Table 3. Processing period by modules**

| Modules | Time Taken to Process the Data (ms) |
|---|---|
| Input Interface Controller | 6.8 |
| Colour Space Conversion | 12.6 |
| Output Interface Controller | 6.2 |
| Other Resources | 2.4 |

Total time taken for capturing the image from the sensor to the display is 28ms.

## 7. CONCLUSIONS

A 5 MP CMOS sensor having a resolution of 2592 column by 1944 row is chosen providing 12 bit pixel output. The obtained 12 bit pixel data is read by the input interface controller; the raw data is then converted to RGB format and given to LCD and VGA displays with proper synchronization. Synthesis of the designed controller reports, the total number of logic elements utilized by the design as 1739 with 1062 logic registers, total number of pins configured with I/Os of the design as 455 with 44752 memory bits and one PLL circuitry utilization. The power analysis shows that, I/Os contribute mostly to the total power dissipation of the design by contributing 147.48 mW in 328.29 mW. The designed controller acquires the image with the speed of less than 30 ms (28 ms) with the frame rate of 10 fps which overcome the limitation of lower acquiring speeds (>=30 ms) reported in the literature.

## 8. REFERENCES


[1] Spoerry T. and Wong K. C. (2006), 'Design And Development Of A Micro Air Vehicle (Mav) Concept: Project Bidule', School of Aerospace, Mechanical and Mechatronic Engineering, University of Sydney

[2] Daniel Crispell (2009), 'Implementation of a Streaming Camera Using an FPGA and CMOS Image Sensor', Brown University

[3] Sanjay Singh et. Al. (2012), 'Real-Time Video Acquisition and PTZ Camera Movement Control for FPGA based Automated Video Surveillance System', International Journal of Research and Reviews in Computer Science (IJRRCS), Vol. 3, No. 2, ISSN: 2079-2557

[4] Sheng Peng (2005) 'Camera Interface Discussion' LCLS Control Group

[5] Altera (2009) High Speed Mezzanine Card, Retrieved December 30, 2012 from <www.altera.com>

[6] Terasic (2010) TRDB_D5M 5 Mega Pixel Digital Camera Development Kit, Terasic Help Centre

[7] Terasic (n.d.) Terasic D5M Hardware Specification, Terasic Help Centre

[8] Adrian Ford and Alan Roberts (1998) Colour Space Conversion Retrieved December 1, 2012 from < www.poynton.com/>

[9] Guillermo Luijk (2008) Zero Noise Virtual RAW Retrieved December 30, 2012 from http://www.guillermoluijk.com/article/virtualraw/index_en.htm

[10] Enciris Technologies (2008), 'PP-01 Video Pre-Processing IP' [Online] available from < www.enciris.com> [Accessed December 16, 2012]

[11] Ampere Ltd. (2009) Specifications for LCD Module, Retrieved December 30, 2012 from < http://www.ampire.com.tw/en/index.asp>

[12] Addison (2004) Video Hardware, Part 2, Retrieved March 01, 2013 from < http://www.devhardware.com/c/a/Video-Cards/Video-Hardware-Part-2/9/>