ABSTRACT

With the advent of active safety technologies in the automotive industry, a need to record and replay the actual on-road vehicular scenario has risen, especially in systems involving camera-based vision. The primary objective of the paper is to propose a design of a system for real-time video acquisition. Hence, a design for a Camera Hardware simulator has been proposed in this paper. The system involves a camera that captures visual information through its image sensor. The system is designed such that it can do direct display; that is, it can generate vertical and horizontal synchronization signals, as per the specification of the camera and it can buffer the pixel clock coming from the camera and send it to another system that uses the video information being received such as an in-vehicle display to display it. It also includes the ability to record the incoming data stream in a computer for offline processing. As the aforementioned functionality is to be achieved for high incoming data rates and also handy interfacing to any recording device is required, we are using a Universal Serial Bus (USB) 2.0 (high speed). Many subsystems are to be designed on the same chip, so we propose the use of a Field Programmable Gate Array (FPGA) based system for fast data processing and miniaturization of the system. The system under consideration is comprised of a Complementary Metal-Oxide Semiconductor (CMOS) camera at the input and a high-speed USB interface at the output. The FPGA is programmed as a Video Graphics Array (VGA) Controller, buffer and a USB Controller. FPGA consumes less power when compared to any other embedded-based system, making it more usable inside an automobile. This system can be used for offline video processing and simulation of the exact on-road scenario in lab for vision based active safety system's testing purpose or for active safety algorithm improvement to achieve desired results (by tweaking the algorithm for desired results, based on the observations made from the recorded data without having the need to go on road again and again).

INTRODUCTION

In the designed system, the FPGA uses a 32 bit soft core processor, to control the data flow direction, to and from the USB, in case of recording. The design entry for direct display module was done using Verilog Hardware Description Language (HDL) and system design for data recording module was done using System on Chip (SOC) builder provided by FPGA supplier's Integrated Development Environment (IDE). Also, for programming the soft-core processor, in the SOC, embedded C was used. After designing the SOC system, a C-code is needed to control the functioning of the soft-core in order to perform the desired
operations. The system was successfully designed, and VGA Display and USB storage were demonstrated. The SOC architecture for high speed data transfer from an image sensor to USB is shown in Figure 1. The device used is a low power FPGA [4].

Figure 1. Block diagram for the acquisition mode

The FPGA-USB System works best for the implementation of high speed video data acquisition because, FPGAs deliver user programmable wire-speed multiprocessing capability unequalled by any microcontroller, and FPGAs provide multiple digital circuits that operate in parallel without interruptions and drive a video display, scan line by scan line, simultaneously without missing a single clock cycle. Thus, FPGAs are faster than micro-controllers and hence meet the high speed requirement of the system. In addition, other advantages of FPGAs include- low cost, low power, faster circuit speeds, a large portfolio of customizable IPs, and easy to use design-software [3].

Section 1 and section 2 (under system level design heading) present implementation of VGA Controller and USB Controller on FPGA respectively. Section 4 explains the methodology followed while developing the system.

SYSTEM LEVEL DESIGN

1. Implementation of VGA-Controller

A VGA video signal contains 5 active signals. Two signals compatible with TTL logic levels, horizontal synchronization and vertical synchronization, are used for synchronization of the video. Three analog signals with 0.7 to 1.0-Volt peak-to-peak levels are used to control the color. The color signals are Red, Green, and Blue. They are often collectively referred to as the RGB signals. By changing the analog levels of the three RGB signals all other colors are produced. In standard VGA format, the screen contains 640 by 480 picture elements or pixels. The video signal must redraw the entire screen 60 times per second to provide for motion in the image and to reduce flicker. This period is called the refresh rate. The human eye can detect flicker at refresh rates less than 30 to 60Hz. The color of each pixel is determined by the value of the RGB signals when the signal scans across each pixel. In 640 by 480-pixel mode, with a 60Hz refresh rate, this is approximately 40 ns per pixel. A 25MHz clock has a period of 40ns. A slightly higher clock rate will produce a higher refresh rate. The screen refresh process begins in the top left corner 1 pixel at a time from left to right. At the end of the first row, the row increments and the column address is reset to the first column. Each row is painted until all pixels have been displayed. Once the entire screen has been painted, the refresh process begins again. The video signal paints or refreshes the image using the following process.

The vertical sync signal, tells the monitor to start displaying a new image or frame, and the monitor starts in the upper left corner with pixel 0,0 [11].

![Figure 2. VSYNC for 640×480 resolution](image)

The horizontal sync signal, tells the monitor to refresh another row of 640 pixels.

![Figure 3. HSYNC Signal for 640×480 resolution](image)

After 480 rows of pixels are refreshed with 480 horizontal sync signals, a vertical sync signal resets the monitor to the upper left corner and the process continues. During the time when pixel data is not being displayed and the beam is returning to the left column to start another horizontal scan, the RGB signals should all be set to the color black (all zeros).

![Figure 4. VGA Image- 640×480 Pixel Layout](image)
The FPGA receives its input signals from the camera sensor through multiple parallel input/output pins. Video output can be developed using hardware inside the FPGA. The data captured through the sensor is converted to Bayer pattern data, which is then converted to digital RGB format, which is then buffered and sent to output lines. Only five signals or pins are required at the output, two sync signals and three RGB color signals. A simple resistor and diode circuit is used to convert Transistor-Transistor Logic (TTL) output pin signals from the FPGA to the low voltage analog RGB signals for the video signal. This supports two levels for each signal in the RGB data and thus produces a total of eight colors. This circuit and a VGA connector for a monitor are already installed on the development/prototyping boards. Phase Locked Loop (PLL) of FPGA can be used to generate clocks for a wide variety of video resolutions and refresh rates.

2. Implementation of the USB Controller

FPGA IDEs provide both embedded and HDL environment. FPGAs support a soft-core processor which can be included in systems to reduce the complexity of implementation. This can later be programmed in Embedded C to achieve the required task with a slight compromise in speed. The Graphic User Interface (GUI)-based SOC builder system in IDEs includes many IPs and resources which can be used in SOC system to be designed and it also connects the various on chip controllers and devices, using the memory mapped or streaming interfaces and interconnects used to connect the systems and components [7], [8].

The system shown in Figure 6 has a soft-core processor which is a programmable processor implemented on the FPGA for controlling the various processes implemented and performed on the SOC. The coding for its functionality is done in Embedded C using the FPGA IDE. Parallel input output (PIO) is used to connect the output of the camera to the FPGA chip. Its width is set to the number of bits per pixel incoming at every rising edge of pixel clock of the camera. The pixel data from the camera is taken from the input pins and sent to an External RAM. Direct Memory Access (DMA) is used so that future intervention of the soft-core processor is not required for this transfer after the initiation. It is used between the PIO and the SD-RAM. The external Synchronous-Dynamic Random Access Memory (SD-RAM) is being used as a buffer to store the incoming data (pixel/video data that is to be sent to the Personal Computer ie PC) temporarily, before it is transferred to the USB so an SD-RAM Controller is used, this controller controls the addressing and signaling tasks to read and write from/to the external SD-RAM chip. Clock Crossing Bridge is used to connect various components in the SOC design which work on different clocks. For example, it can be used between the USB controller operating at a higher clock frequency and the SD-RAM being controlled at a different clock frequency. An Interconnect Fabric is used to connect the various independent systems implemented using SOC builder. The on chips memory provided is low so the program should be stored on the external flash memory provided on the kit, and its designated address can be provided as the reset address of the processor while configuring it. A USB controller IP provided by the FPGA supplier was used, and a RAM based controller was selected for the application. This controller reads the data present on the SD-RAM and sends it out to the PC where it is stored. USB 2.0 is used here because the data transfer speed required for this application is very high and can be provided in the high speed mode of USB 2.0.
The USB 2.0 controller IP supports high speed mode offering a transfer rate of 480 Mbps (theoretically), and can also be configured for bulk, isochronous and interrupt-based transfers. USB Data Stream Controller (UDSC) interface performs data transfer from USB to external memory as well as external memory to USB device in stream fashion without any software interaction because of auto configuration of endpoint buffer registers. As USB is a serial communication protocol and the incoming data is parallel, buffering through SD-RAM along with the usage of the USB 2.0 IP provide the facility to convert parallel data into serial, thereby achieving data transference through USB 2.0. A Memory-Mapped master peripheral, such as a CPU, can off load memory transfer tasks to the UDSC controller. While the UDSC controller performs memory transfers, the master is free to perform other tasks in parallel. The UDSC Controller transfers data as efficiently as possible, reading and writing data at the maximum pace allowed by the source or destination. The UDSC controller is capable of performing transfers with flow control, enabling it to automatically transfer data to/from a slow peripheral with flow control, at the maximum pace allowed by the peripheral.

3. Recording
USB 2.0 interface is very common in modern day computers. USB is also capable of high speed data transference which is necessary for real time recording to happen. A USB device driver was programmed, which invokes a software (already installed) on the host computer that facilitates the storing of the incoming raw data stream in uncompressed binary format, this uncompressed video data can be exported using the software into a text file. The stored raw data can be processed into a standard video format (such as avi), after taking into account the pixel width and image frame size of the recorded video data, using any image processing software available. The advantages of recording raw data are numerous though such a recording will cost a lot of space. The raw data can be used to simulate the actual on road scenario and various video based active safety algorithms (such as extracting of ground truth data of the acquired video log, target acquisition etc) can be implemented on the recorded raw data and their advantages/disadvantages can be observed. After all the processing is done on the recorded data, the new data thus generated can also be converted to a standard video for observation purpose again. Thus, the need to take the vehicle on the road again and again for system testing is avoided. Figure 9 shows a snapshot of a video which was obtained after target identification algorithm was used to process the raw data which was then converted to a video.
4. Methodology

The code for VGA Controller and the Direct Display module was written in Verilog HDL using the IDE of the FPGA provider, and the SOC system of the Video acquisition module was designed using SOC Builder tool present in IDE, which included the usage of IPs provided by the FPGA OEM, and the functionality code for the working of the Soft-Core Processor, that controlled the designed SOC system, was written in Embedded C. Various files required were included in the C code, and the Hardware Abstraction Layer (HAL) corresponding to respective on-chip system modules were used in the code. A driver was designed so that as soon as the device is plugged onto a computer, an installed software is invoked which facilitates the recording.

5. Result

The system for Direct Display through the VGA port was designed and its functionality was tested by directly connecting the PIO pins of the prototyping/development board to the output of camera, performing RGB conversion on the raw data (using Verilog), and sending the RGB data to the Digital to analog converter connected to the VGA port of the board.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Utilization</th>
<th>Availability</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Logic Elements</td>
<td>1554</td>
<td>33216</td>
<td>5%</td>
</tr>
<tr>
<td>Total Combinational Functions</td>
<td>1266</td>
<td>33216</td>
<td>4%</td>
</tr>
<tr>
<td>Dedicated Logic Registers</td>
<td>1040</td>
<td>33216</td>
<td>3%</td>
</tr>
<tr>
<td>Total Registers</td>
<td>1055</td>
<td>N.A.</td>
<td>N.A.</td>
</tr>
<tr>
<td>Total Pins</td>
<td>425</td>
<td>475</td>
<td>89%</td>
</tr>
<tr>
<td>Total Memory</td>
<td>62416</td>
<td>483840</td>
<td>13%</td>
</tr>
<tr>
<td>Total PLLS</td>
<td>1</td>
<td>4</td>
<td>25%</td>
</tr>
</tbody>
</table>

The video acquisition system was developed using a SOC builder tool as described previously. This system was generated and instantiated in the top module of the system. Its functionality was observed by using the system generated to communicate with a computer and by performing video data (raw pixel data of the video) recording. A suitable code using the FPGA IDE was written in Embedded C for the soft-core processor, so that it can control the various sub-systems that were instantiated in the SOC design in order to obtain the desired functionality. The system thus designed was not only economical for mass production but also was better than the existing solutions in the same price range. Off-shelf designs such as TI DM643x (Da Vinci Processor) [13], is capable of just capturing the video data coming from external image sensor. But our design is capable of storing the raw image data and also displaying the data using a VGA port, also as FPGA's ports can be configured for both input and output our design can be extended to add video playback feature too. The raw pixel data stored can be read through the USB port and along with Vsync, Hsync and pixel clock (generated using the counters implemented on FPGAs) can be used to do direct display (as explained earlier under implementation of VGA controller heading), as FPGAs output pins can drive voltage signals. It was also found that 3.8 Giga bytes of raw data of resolution 640×480, equivalent to 370 minutes of recording saved the users from more than 40 hours of on-road system testing (trying to replicate the same scenario), and this application establishes itself in every software release.

CONCLUSION

The objective of designing and implementing a video acquisition system using FPGA was achieved. The designed system consumed little power and used a minimum of logical elements. It also overcame the problem of any delays as the USB was used in high-speed mode. The arbitration of the DMA channels helps in maintaining a balance between buffering and the transfer rates. It also helps in reducing the pressure on the input video buffer. For the video acquisition system to operate properly it was observed that for its optimum operation the pixel clock of the camera had to be reduced to 9.6 MHz (around 23 frames per second) as the system designed using the SOC Builder tool is slower than the one designed using Verilog HDL. The architecture suffices the requirements for low power video streaming and data transfer applications. As the whole system was implemented on a single FPGA chip, a good reduction in size was achieved.

The user could not only view the current on-road scenario directly through a VGA display inside a car (by using the direct display module) but also record the same scenario for future reference (in the form of raw data, using the video acquisition module simultaneously), making the need to physically replicate an on-road scenario negligible. The VGA based direct display system module could also be coupled with an Infrared illuminated camera (through PIO), thereby converting this system into a night vision display product.

The system can support recording for low standard resolutions not higher than 640×480 as higher resolutions would not only require more recording space but also a higher overhead of parallel data to serial conversion will be required which will involve large intermediate buffers and higher communication speed which cannot be practically achieved using this system (taking into account the slower
SOC design based on Embedded C) and maximum achievable speed is roughly 200 Mbps of the net recording system as a whole (because system also involves intermediate buffering by writing to SD-RAM which is then again read by USB controller). The minimum speed required for recording is roughly given by: resolution x number of frames per second (23 here) x pixel size (24 bits here), thus if a higher standard resolution of 1024×768 is used then the minimum transfer rate required should be 1024×768×23×24 = 434 Mbps (here the frame start and end indicators are not counted), as against 170 Mbps required for 640×480 (mostly used resolution for vision based active safety systems). The design can be made faster by coding the video acquisition system in Verilog HDL, but that will remove the ease of implementation achieved using SOC builder tool.

As Active safety systems are radar based also and in-fact at times can be based on the fusion of both camera and a radar, the design can be extended to take raw data (through PIO pins) from the output of the digital signal processing (DSP) based microcontroller attached to the radar and then recording it. Also an important feature of replaying the recorded scenario by reading the recorded/processed raw video data from the computer through the USB port and thus driving the VGA display (as explained earlier in results), is an extension that is being worked upon in the present design.

The behavior of the video based active safety system under varying road conditions can be observed by recording the Controller Area Network (CAN) log of the Electronic Control Unit ie ECU (linked to the vision based active safety system) under consideration and studying the responses of the ECU based on the situations faced on the road (which can be observed from the recorded data, after it is converted to a suitable format for viewing), this reproducible information can tell the user whether this system works properly (as expected) or not. Also, if any algorithm tweaks are required, the same can be modified and run on the same recoded scenario and the effects (desirable or not) can be observed. Thus the system need not be taken on the road time and again (for the same captured video sequence/on-road scenario) for testing.

REFERENCES

2. Orange Tree Technologies, “Implementing USB into Industrial Environments using FPGA Technology”.
5. ISP1761 Hi-Speed Universal Bus On-The-Go controller, Rev.04-5March2007.

ABBREVIATIONS

FPGA - Field Programmable Gate Array
SOC - System on Chip
CMOS - Complementary Metal-Oxide Semiconductor
TTL - Transistor Transistor Logic
VGA - Video Graphics Array
USB - Universal Serial Bus
UDSC - USB Data Streamer Controller
IDE - Integrated Development Environment
RGB - Red, Green, and Blue
PLL - Phase Locked Loop
RAM - Random Access Memory
ROM - Read only Memory
FIFO - First in First Out
HDL - Hardware Description Language
GUI - Graphic User Interface
JTAG - Joint Test Action Group
I2C - Inter Integrated Circuit
OEM - Original Equipment Manufacturer
CAN - Controller Area Network
ECU - Electronic Control Unit
DSP - Digital Signal Processing