Authentic Time Hardware Co-simulation of Edge Discovery for Video Processing System

R. NARESH
M. Tech Scholar,
Dept. of ECE

R. SHIVAJI
Assistant Professor,
Dept. of ECE

PRAKASH J. PATIL
Head of Dept.ECE,
Vijay Rural Engineering College, Nizamabad, Jntu-H

Abstract:
A hardware implementation of a real time video edge detector has been realized on a Xillinx Spartan-3A, the edge detection implementation is based on Sobel algorithm with a dynamically adjusted threshold. Threshold adjustment is based on both; a threshold seed calculated based on a moving average of pixel gradients and a user-input used to scale the calculated threshold seed. A methodology for implementing real-time DSP applications on a field programmable gate arrays (FPGA) using Xilinx System Generator (XSG) for Mat lab is presented in this paper.

It presents architecture for Edge Detection using Sobel Filter for image processing using Xilinx System Generator. The design was implemented targeting a Spartan3A DSP 3400 device (XC3SD3400A-4FGG676C) then a Virtex 5 (xc5vlx50-1ff676). The Edge Detection method has been verified successfully with no visually perceptual errors in the resulted images. To take full advantage of the FPGA capabilities, the hardware implementation is based on parallel communicating sequential processes. Inter-process synchronization is achieved with simple request-grant handshaking protocol.

Keywords: *FPGA, Sobel Filter, VPS*

1. Introduction

The Rising promote for video processing systems requires high-performance digital signal processing as well as low device costs appropriate for a volume application. Xilinx FPGA devices provide a platform with which to meet these two contrasting requirements. A Xilinx tool, the System Generator for DSP [1], offers an efficient and straightforward method for transitioning from a PC-based model in Simulink to a real-time FPGA based hardware implementation.

The system model can be simulated in the Simulink environment. This higher abstraction level reduces the analysis and debugging time. For real hardware testing, Xilinx System Generator supports the possibility to perform hardware in-the-loop co-simulation. This methodology provides easier hardware verification and implementation compared to HDL based approach. The Simulink simulation and hardware-in-the loop approach presents a far more cost efficient solution than other methodologies. The ability to quickly and directly realize a control system

design as a real-time embedded system greatly facilitates the design process. The goal of this project was to implement an image-processing algorithm applicable to Edge Detection system in a Xilinx FPGA using System Generator for DSP, with a focus on achieving overall high performance, low cost and short development time.

2. System Overview

Figure below shows the implemented platform for real time video edge detection. It consists of; 1) A Seattle Robotics CMUCam1 camera, 2) A Xilinx Sparatan-3A FPGA prototyping board and 3) A regular computer monitor with 60 Hz refresh rate. The video camera is connected to the board through a serial interface while the monitor is connected to it through a regular VGA cable. A brief description of the camera and board is provided next.

2.1 Input Video Specifications

The input video data will be streamed from a CMUCam1 compact video camera using RS232 serial interface. The CMUcam1 is a low-cost vision sensor developed by Carnegie Mellon University and can provide an image with a resolution of 80 x 143 pixels. Communication with the CMUcam1 can be done through certain serial commands starting with initializing its internal registers to proper values like the clock speed, brightness and contrast through simple commands like the dump frame command "DF/r". Because of the limited data rate of the RS232 serial interface, a maximum of 115,200 baud, the image raw data will be dumped at 17 columns per second in the following format [3]: **1 2 r g b r g b r g b r g b … 2 r g b r g b … 3**

Where:1 = Frame Start, 2 = New Column, 3 = Frame End, r = Red Value 16-240, g = Green Value 16-240, b = Blue Value 16-24

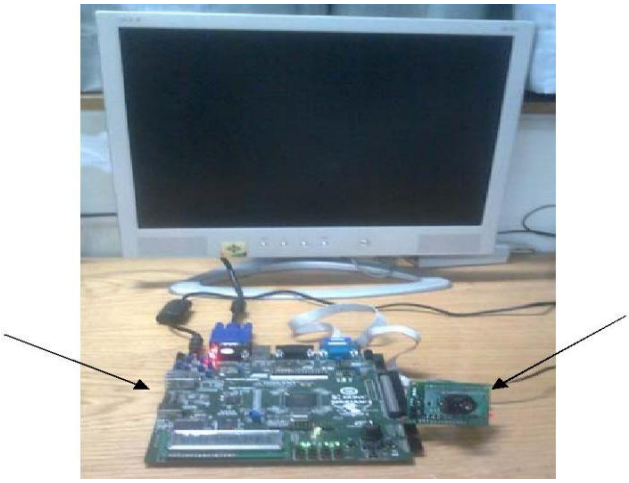

Fig. 1 Picture of the implemented real-time video edge detection platform.

2.2 Xilinx Sparatn-3A FPGA Prototyping Board

The Spartan-3A FPGA board is used to implement the edge detection circuitry as well as the data interface circuitry (a UART to read video from the camera, a VGA controller to drive the computer monitor and LCD controller to provide user instructions). The board's main features utilized in this work are listed below:

- **FPGA:** A Sparatan-3A XC3s700A FPGA with an equivalent gate density of 700,000 gates, twenty 18-Kbit dual-port block RAMs (BRAMs) totaling 360Kb of on-chip RAM, hundreds

of configurable I/O blocks (IOBs), and 12 digital clock managers (DCMs).

- **Configuration EEPROM:** A 4 Mbit Xilinx Platform Flash configuration PROM to hold the FPGA's configuration data
- **Clock:** A 50 MHz clock oscillator
- **Display:** Two-line 16-character LCD screen.
- **VGA Interface:** a 12-bit color VGA display port.
- **Serial Interface:** Two nine-pin RS-232 ports (DTE- and DCE-style).
- **User Inputs:** A rotary-encoder with push-button shaft used for threshold adjustment by the user, four slide switches, four push-button switches for other inputs like system reset, and eight discrete LEDs for monitoring the status of internal circuitry and debugging.

2.3 Design Challenges

Many design challenges have surfaced due to the limitations and incompatibilities of the used components. These challenges are listed below:

1. **Limited Interface:** The FPGA prototyping board does not come with a pre-designed USB PHY (physical interface and controller). This interface is very difficult to design and implement on the FPGA. As a result we were forced to use the serial interface for inputting the video stream since it is much easier to build a UART. This however, limited the maximum bit-rate to 115,200 bps, which limited the maximum full-color 'pixel' rate to 4,800/second. This meant that both resolution and frame rates had to be reduced.
2. **Serial Interface Camera:** The above limitation dictated the use of a video camera with a serial interface and reduced resolution and frame rate, CMUCam1.
3. **Image Distortion by Camera:** The CMUCam1 camera distorts image which require doubling the columns of the output images to correct it.
4. **Aspect Ratio Mismatch:** The camera output frames at 80 X 143 resolution, while the VGA monitor has a 640 X 480 resolution. To resolve this issue without distorting the image, the input images are trimmed down to 80 X 120 to match the VGA aspect ratio. Also each input image pixel is replicated 8 times per row (x 2 to correct the input distortion and x 4 to match the VGA aspect ratio) and 4 times per column achieving an effective image magnification factor of 4.
5. **Limited on-FPGA RAM:** The available BRAMs on the FPGA are not enough to hold more than one full image frame. Hence, the input buffer holds only a portion of the image requiring the overlap of frame loading, frame processing (applying the Sobel operator) and the production of the modified frame operations.
6. **Limited FPGA RAM Organization:** The FPGA's RAM blocks are made of 1K words, each 18-bit wide. They can be configured to be 1, 2, 4, 8 or 18-bit wide. To take advantage of these blocks without any waste, the design had to be restricted to these widths or multiple of them.
7. **Different data-width**: Camera produces 24-bit pixels, while the board's VGA interface supports 12-bit pixels only. So gradient calculations are performed on the full-color 24-bits pixels which are trimmed down to 12-bits before being outputted through the VGA port.
8. **Different data rates**: The VGA interface operates at 25 MHz frequency, while pixels are received from the camera at a much lower rate. Hence the image update is slowed down.
9. **Different data format**: The CMUCam1 camera produces video frames column by column while the VGA standard requires the video frames row by row as illustrated in Figure below. This meant that the image buffers on the FPGA had to be written to column by column but read from row by row. Also the Sobel operator mask had to be applied column wise. CMUCam1 produces images column by column VGA images are produced row by row.

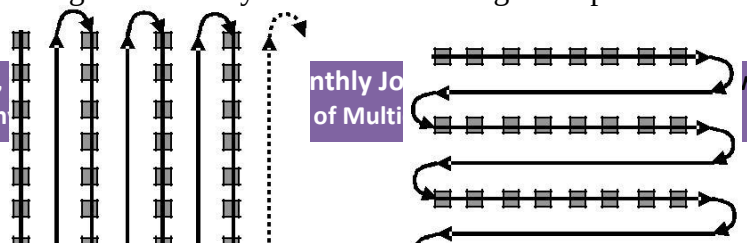

Fig. 2 Incompatible data format between Camera and VGA

Xilinx System Generator [1], is a system-level modeling tool from Xilinx that facilitates FPGA hardware design. It extends Simulink in many ways to provide a modeling environment well suited for hardware design. The software automatically converts the high level system DSP block diagram to RTL. The result can be synthesized to Xilinx FPGA technology using ISE tools. All of the downstream FPGA implementation steps including synthesis and place and route are automatically performed to generate an FPGA programming file.

Fig. 2 presents the design flow of XSG. System Generator automates the design process, debugs, and implements and verifies the Xilinx-based FPGAs. It provides a high-speed HDL co - simulation interface, system-level resource estimation, and accelerated simulation through hardware in the loop co-simulation interfaces which give up to a 1000x simulation performance increase.

It also provides a system integration platform for the design of DSP FPGAs that allows the RTL, Simulink, MATLAB and C/C++ components of a DSP system to come together in a single simulation and implementation environment. System Generator supports a black box block that allows RTL to be imported into Simulink and co-simulated with either ModelSim or Xilinx ISE Simulator.

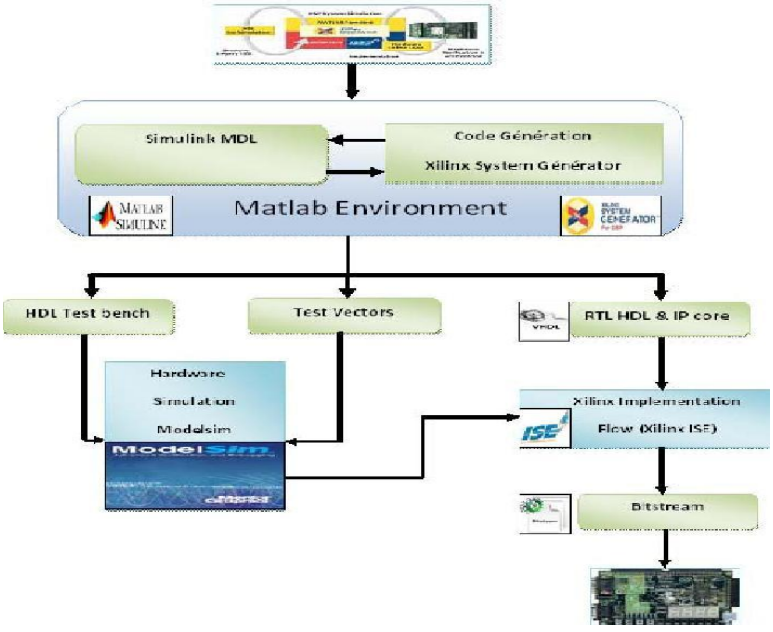

Fig. 3 System Generator design flow

3. Edge-Detection Algorithm

The Sobel algorithm basically detects the edges by looking for the maximum and minimum in the first derivative of the image (i.e. gradient). A pixel location is declared an edge location if the value of the gradient exceeds some threshold. The Sobel operator performs a 2-D spatial gradient measurement on an image. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image. The Sobel edge detector uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y- direction (rows). A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The actual Sobel masks are shown in below:

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Fig. 4 The Sobel Mask.

The magnitude of the gradient is approximated as: $|G| = |Gx| + |Gy|$
Using the above masks:
$Gx = +2*P(i,j+1) – 2*P(i, j-1) + P(i-1,j+1) – P(i-1, j-1) + P(i+1,j+1) – P(i+1, j-1)$,
$Gy = +2*P(i-1,j) – 2*P(i+1, j) + P(i-1,j-1) – P(i+1, j-1) + P(i-1,j+1) – P(i+1, j+1)$ below illustrates the algorithm.

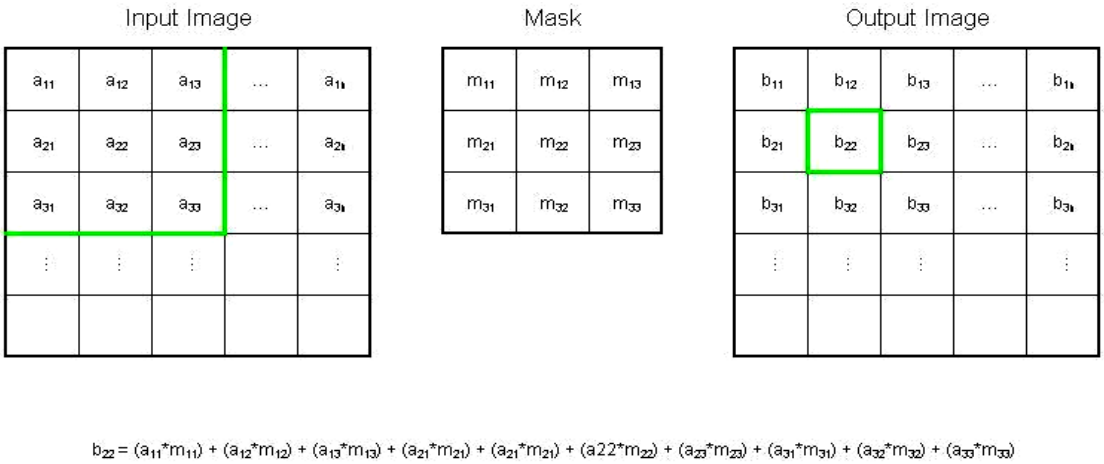

Fig. 5 Illustration of the Sobel Algorithm.

The gradients are calculated on each color plane separately and so is the edge threshold. An adaptive scheme is used for the threshold value (i.e. thresholds are not static). For each color, a threshold seed is calculated by averaging all the absolute values of gradients in the last frame (i.e. a moving average). This seed is then multiplied by a user-entered factor to increase or decrease edge sharpness. This adaptive scheme for seed calculation will automatically adjust the threshold to match the image quality; images with many details will have a larger threshold to keep the detected edges uncluttered, while images with fewer details will have a smaller

threshold to detect subtle changes in colors (soft edges).

4. Hardware/Software Co-Simulation In System Generator

Sometimes it is important to add one or more existing HDL modules to a System Generator design. The System Generator Black Box block allows VHDL, Verilog, and EDIF to be brought into a design. The Black Box block behaves like other System Generator blocks – it is wired into the design, participates in simulations, and is compiled into hardware. When System Generator compiles a Black Box block, it automatically wires the imported module and associated files into the surrounding netlist.

The Black Box block provides a way to incorporate hardware description language (HDL) models into System Generator.The design of our architecture with Xilinx System Generator The Black Box contains our defined VHDL description for sobel operator. The subsystems in the simulation model allow serialization and the reconstruction of the image when the pixel output is generated by the hardware model. System Generator simulates black boxes by automatically launching an HDL simulator, generating additional HDL as needed (analogous to an HDL testbench), compiling HDL, scheduling simulation events, and handling the exchange of data between the Simulink and the HDL simulator. This is called HDL co-simulation. System Generator provides hardware co-simulation, making it possible to incorporate a design running in an FPGA directly into a Simulink simulation. "Hardware Co-Simulation" compilation targets automatically create a bitstream and associate it to a block.

When the system design is simulated in Simulink, results for the compiled portion are calculated in actual FPGA hardware, often resulting in significantly faster simulation times while verifying the functional correctness of the hardware. System Generator for DSP supports Ethernet as well as JTAG communication between a hardware platform and Simulink.

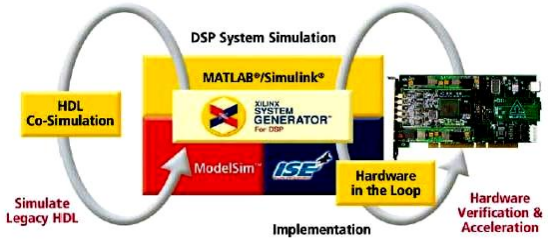

Fig. 6 FPGA based Hardware-Software (HW-SW) co- simulation

System Generator provides a generic interface that uses JTAG and a Xilinx programming cable (e.g., Parallel Cable IV or Platform Cable USB) to communicate with FPGA hardware. the model with the JTAG-based hardware co-simulation block implemented on Virtex 5 platform. Point-to-point Ethernet co-simulation provides a straightforward high-performance co-simulation environment using a direct, point-to-point Ethernet connection between a PC and FPGA platform. The target FPGA chip is Xilinx Spartan 3A DSP 3400 XC3SD3400A-4FGG676C and Virtex 5 xc5vlx50-1ff676. The optimization setting is for maximum clock speed. Table 1 details the resource requirements of the design. Note that in practice, additional blocks are needed for input/output interfaces, and synchronization.

Table 1. FPGA Resources Used In the Implementation for the Sobel Edge Detector

	Spartan 3A DSP 3400			Virtex 5 xc5vlx50-1ff676		
	Used	Available	%	Used	Available	%
Number of Slice Registers	2302	23872	9%	1798	28800	6 %
Number of Slice LUTs	1755	47744	3%	2299	28800	7 %
Number of LUT-FF pairs	3023	47744	6%	370	3727	9%
Number of bonded IOBs	34	469	7 %	34	440	7 %
Number of BUFG/BUFGCT	1	24	4 %	1	32	3 %
Number of DSP48s	3	126	2%	-	-	-
Maximum Frequency	59.552 MHz			103.616 MHz		

Table 2. Performance Comparison

	Our Design			Design [9]		
	Used	Available	%	Used	Available	%
Number of Slices	177	768	23 %	204	768	26 %
Number of Slice Flip Flop	401	1536	26 %	280	1536	18 %
Number of 4 input LUTs	277	1536	18 %	202	1536	13%
Number of bonded IOBs	34	124	27 %	81	124	65 %
Number of GCLKS	1	8	12 %	1	8	12 %
Maximum Frequency	54.505 MHz			134.756MHz		

5. Conclusion

Xilinx system generator has a unique hardware in the loop co-simulation feature that allows designers to greatly accelerate simulation while simultaneously verifying the design in hardware. The purpose of this paper was to demonstrate the use of System Generator to design a system Edge Detection for image processing. This design is implemented in the device Spartan 3A DSP 3400 (XC3SD3400A-4FGG676C) and Virtex 5 (xc5vlx50-1ff676). The implemented Sobel Edge Detector architecture using low cost available Spartan 3 development system with Xilinx chip XC3S50 -5PQ208 has 54.505 MHz maximum frequency and uses 177 CLB slices with 23% utilization, so there is possibility of implementing some more parallel processes with this architecture on the same FPGA.Future works include the use of the Xilinx System Generator development tools for the implementation of other blocks used in computer vision like feature extraction and object detection on Xilinx Programmable Gate Arrays (FPGA).

References

1. Abbasi, T. A. and Abbasi, M .U. "A proposed FPGA based architecture for sobel edge detection operator", J. of Active and Passive Electronic Devices, Vol. 2, pp. 271–277.
2. Dong, Q., Song, C., Ben, C., Quan, L., (2005). "A fast subpixel edge detection method

using Sobel-Zernike moments operator", Image and Vision Computing, Vol.23, pp.11-17, 2005.

3. http://www.mathworks.com/
4. Mittal, S., Gupta, S. and Dasgupta, S. (2008). "System Generator: The State-Of-Art FPGA Design Tool For DSP Applications", Third International Innovative Conference On Embedded Systems, Mobile Communication And Computing (ICEMC2 2008), August 11-14, Global Education Center, Infosys.
5. Neoh, H., Hazanchuk, A. (2004). "Adaptive Edge Detection for Real-Time Video Processing using FPGAs", Global Signal Processing.
6. Saidani, T. D. Dia, W. Elhamzi, Atri, M. and Tourki, R. (2009). "Hardware Co-simulation For Video Processing Using Xilinx System Generator" Proceedings of the World Congress on Engineering 2009 Vol I, WCE, July 1 - 3, 2009, London, U.K.
7. Shigeru.A, (2000). "Consistent Gradient Operators", IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (3).