International Journal of Latest Research in Engineering and Technology (IJLRET)
ISSN: 2454-5031
www.ijlret.com || Volume 03 - Issue 05 || May 2017 || PP. 24-28

A Fast Image Processing System based on Embedded Technology

PengLiu
College of Computer Science, Yangtze University, Hubei,China

Abstract: The main target of this work is to evaluate the performance of a host/co-processor architecture including an embedded Nios-II processor and utilizing a communication channel between host and hardware board, like a USB2.0 channel. The task-logic performed by the embedded accelerator can be any image function within the limitations of existing FPGA devices. For our purpose we built a typical image-processing application in order to target the FPGA co-processor. It consists of a noise filter followed by an edge-detector. Noise reduction and edge detection are two elementary processes required for most machine vision applications, like object recognition, medical imaging, lane detection in next-generation automotive technology, people tracking, control systems, etc.

Keywords: Image processing, Embedded technology, FPGA, SOPC

I. INTRODUCTION

The traditional hardware implementation of image processing uses Digital Signal Processors (DSPs) or Application Specific Integrated Circuits (ASICs). However, the growing need for faster and cost-effective systems triggers a shift to Field Programmable Gate Arrays (FPGAs),where the inherent parallelism results in better performance. When an application requires real-time processing,like video or television signal processing or real-time trajectory generation of a robotic manipulator, the specifications are very strict and are better met when implemented in hardware [1]. Computationally demanding functions like convolution filters, motion estimators, two-dimensional Discrete Cosine Transforms (2D DCTs) and Fast FourierTransforms (FFTs) are better optimized when targeted on FPGAs [2]. Features like embedded hardware multipliers, increased number of memory blocks and system-on-a-chip integration enable video applications in FPGAs that can outperform conventional DSP designs[3].

Depending on the system requirements other approaches may also be used. Using a video input card like the Altera DC Video-TVP5146N one can input video data to the FPGA and perform a range of image functions, like simple filtering, color processing, recasting to different video formats, compression, etc. [4]. In this case one can by pass the computer-based frame grabber and avoid the use of a host PC computer. This has the principal cost of losing the flexibility inherent in the software part of the system as well as losing the PC-based control over the cameras and frame grabber, but the result is an all-hardware system that excels in performance. Processing rate is in principle limited by the frame grabber, which in the case of NTSC composite input signal is 30 fps while for PAL video input it is 25 fps.

Similarly, processing at video-rate may be obtained by using the ASICs approach, which usually incorporates all parts in an all-hardware custom system. Image processing implementations of medium and high complexity appear in the literature and most of them can manipulate images in real time. They can reach a frame rate of 30 frames per second or even more in some custom systems. In some cases they can also incorporate a CCD interface. However, ASIC implementations are complex and expensive systems that suffer in terms of flexibility. They have slow design cycle and are certainly away from the plug and play approach adopted in the present article.

Pure DSP-based designs support thousands of MIPS and are comparable in performance with desktop PCs,even running at slower clock speeds. They also consume much less power. Being purely software-based they are much more flexible than hardware implementations. In addition, the control flow part of an algorithm is better mapped to software than to hardware. Moreover, they can easily incorporate video input and output channels with little additional hardware [5]. However, just as is the case with ordinary serial computers they cannot perform computationally demanding tasks at a high video-rate.

II. MODELING AND DESIGN OF EMBEDDED SYSTEM

A model of noise and edge filtering is made by using the Altera DSP Builder Libraries in Simulink. Noise reduction is applied with a Gaussian 3×3 kernel while edge detection is designed using typical Prewitt or Sobel filters. These functions can be applied combined in series to achieve edge detection after noise reduction. The main block diagram of our filter accelerator is shown in Figure 1.

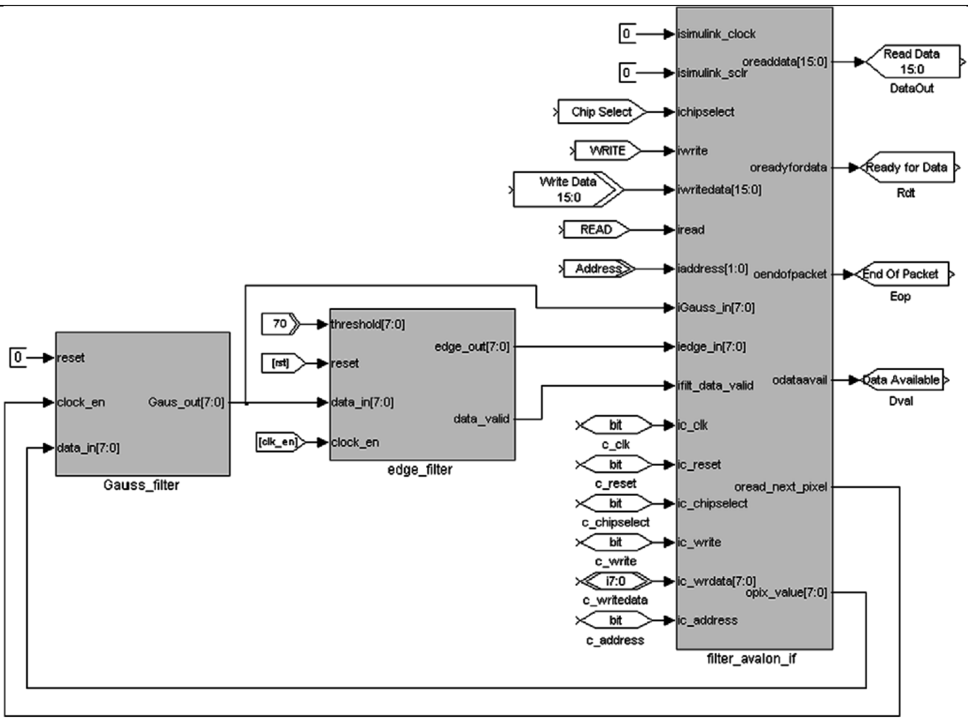


Figure 1.The main block diagram of filter accelerator

Apart from noise and edge filter blocks, there is also a block representing the intermediate logic between the data and control paths and our filter task logic. Such intermediate hardware fabric follows a specific protocol referred to as Avalon interface.This interface cannot be modeled in the Simulink environment and is rather inserted in the system as a Verilog file.Design examples implementing the Avalon protocol can be found in Altera reference designs and technical reports. In brief, our Avalon implementation consists of a 16-bit data-input and output path, the appropriate Read and Write control signals and a control interface that allows for selection between the intermediate output fromthe Gauss filter or the output from the edge detector. Data input and output to and from the task logic blocks is implemented with the help of Read and Write instances of a 4800 bytes FIFO register.Each image frame when received by the hardware board is loaded into an external SDRAM memory buffer and is converted into an appropriate 16-bit data stream by means of instruction code. Data transfer between external memory buffers and the data bus is achieved through Direct Memory Access (DMA) operations controlled by appropriate instruction code for the soft processor.

Incoming pixels are processed by means of a simple 2D digital Finite Impulse Response (FIR) filter convolution kernel, working on the grayscale intensities of each pixel's neighbors in a 3×3 region. Image lines are buffered through delay-lines producing primitive 3×3 cells where the filter kernel applies. The line-buffering principle is shown in Figure 2. A z1 delay block produces a neighboring pixel in the same scan line, while a z640 delay block produces the neighboring pixel in the previous image scan line.We assume image size of 640×480 pixels. The line-buffer circuit is implemented in the same manner for both noise and edge filters. Frame resolution is incorporated in the line-buffer diagram as a hardware built-in parameter. If achange in frame size is required we need to re-design and re-compile. The number of delay blocks depends on the size of the convolution kernel, while delay line depth depends on the number of pixels in each line. Each incoming pixel is at the center of the mask and the line buffers produce the neighboring pixels in adjacent rows and columns.Delay lines with considerable depth are implemented as dedicated RAM blocks in the FPGA chip and do not consume logical elements.

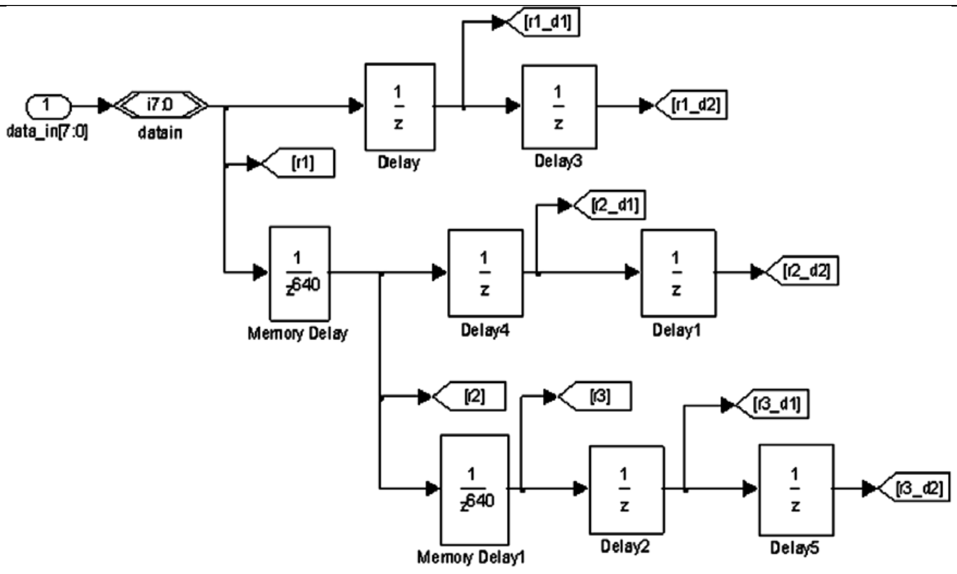


Figure 2.The line-buffering principle

After line buffering, pipelined adders and embedded multipliers calculate the convolution result for each central pixel. The model-design for implementation of the 3×3 Gauss kernel calculations. The model-based design transfers the necessary arithmetic into a parallel digital structure in a straightforward manner. Logic-consuming calculations, like multiplications are implemented using dedicated multipliers available in medium-scale Altera FPGAs, like the Cyclone II chip. When the two filters work in combination, the output of the Gaussian kernel is input to a 3×3 Sobel or Prewittedge detection kernel. First, the kernel-pixels are produced using a line-buffer block identical to the one in Figure 2. The edge detector is composed of horizontal and vertical components.

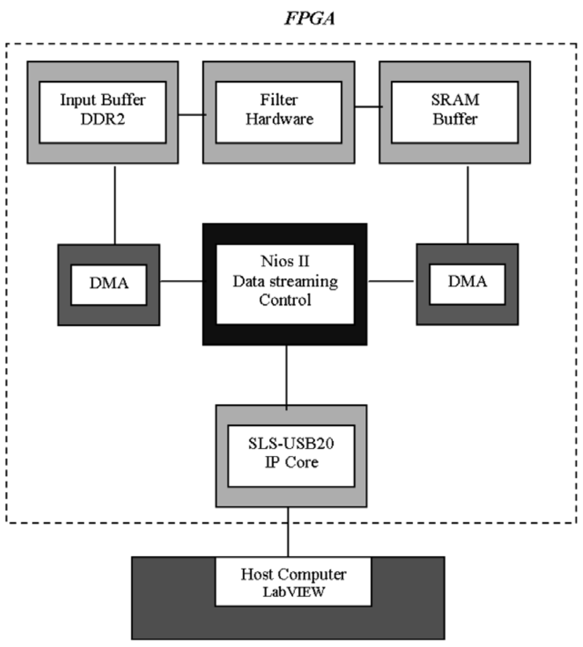


Figure3.The model-design for the Prewitt kernel calculations of horizontal image gradient

A similar implementation is used for vertical gradient. For simplicity we combine horizontal and vertical edge detection filtering by simply adding the corresponding magnitudes. A binary image is produced by thresholding the result by means of a comparator. These final hard system are shown in Figure4.

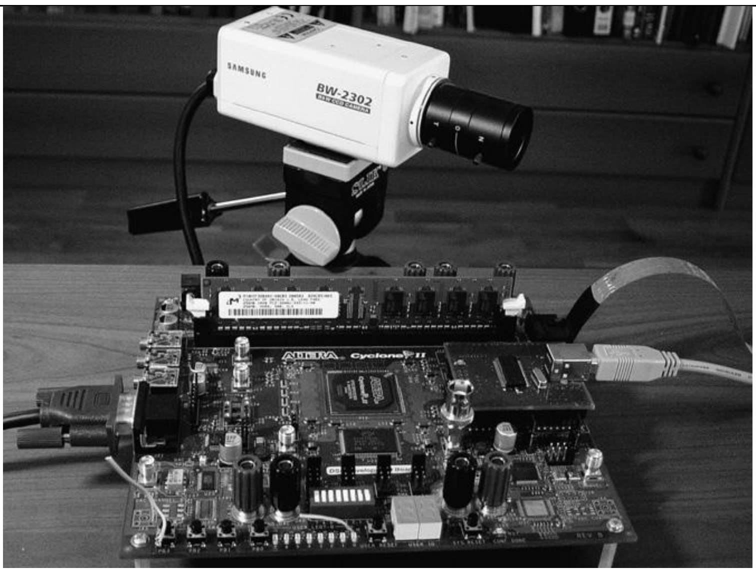
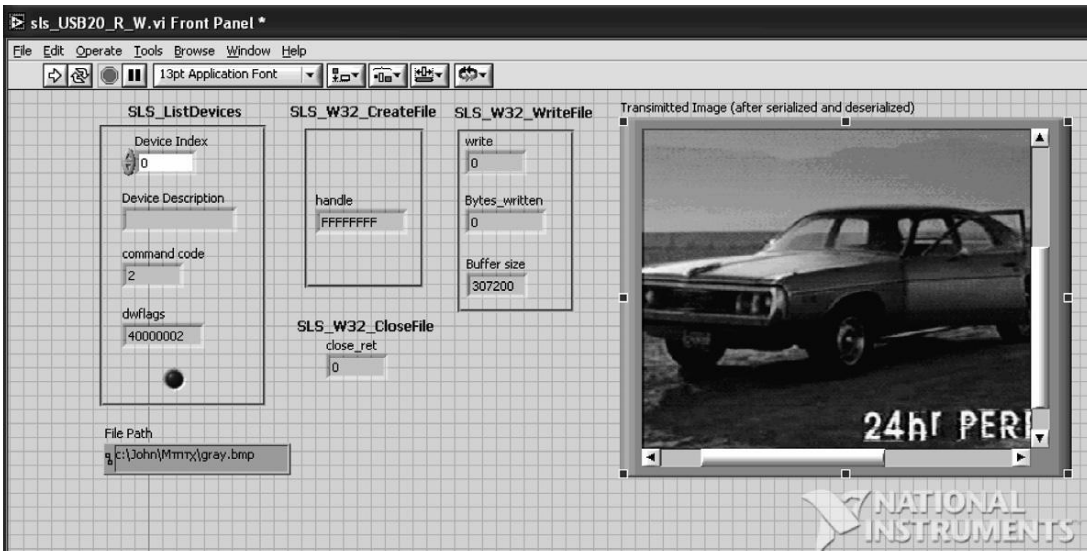


Figure4.The hard system

III. LABVIEW SIMULATION

On the host part a vision system is implemented, appropriate for a spectrum of industrial applications. The host computer is a Windows XP Pentium IV featuring an onboard high speed USB 2.0 controller and a NI 1408 PCI frame grabber. The host application is a LabVIEW virtual instrument (VI) that controls the frame grabber and performs initial processing of the captured images. The frame grabber can support up to five industrial cameras performing different tasks. In our system the VI application captures a full frame sized 640 · 480 pixels from a CCIR analog B& W CCD camera (Samsung BW-2302). It may then reduce the image resolution applying an image extraction function, down to 320×240 pixels. It can produce even smaller frames in order to trade size for transfer rate.

The LabVIEWhost application communicates with the USB interface using API functions and a Dynamic Link Library (DLL) and transmits a numeric array out of the captured frame. An advantage of using LabVIEWas a basis for developing the host application is that it includes a VISION library able to perform fast manipulation of image data or a preprocessing of the image if it is necessary.


sls_USB20_R_W.vi Front Panel *
File Edit Operate Tools Browse Window Help
13pt Application Font
24hr PERI
NATIONAL INSTRUMENTS

SLS_ListDevices
Device Index: 0
Device Description:
command code: 2
dwflags: 40000002

SLS_W32_CreateFile
handle: FFFFFFFF

SLS_W32_WriteFile
write: 0
Bytes_written: 0
Buffer size: 307200

SLS_W32_CloseFile
close_ret: 0

File Path: c:\John\Mπτχ\gray.bmp

Transimitted Image (after serialized and deserialized)

Figure5.LabVIEW simulation for a 640×480 pixel image

When the reception of the image array is completed at the hardware board end, the system loads the image data to the filter coprocessor and sends the output to the VGA controller via SRAM memory.

Alternatively the output can be sent back to the host application by means of a ‘‘write’’ Nios command. The procedure is repeated with the next captured frame.

IV. EVALUATION OF THE SYSTEM PERFORMANCE

The above set up was tested with various capture rates and frame resolutions. Using several test-versions of the SLS USB2.0 megafunction we measured receive (rx) and transmit (tx) throughput between the host PC and the target hardware system.We use a payload of 307,200 bytes for both directions. We find that using the Nios II HAL driver,the latest evaluation version 1.2 of the IP core transfers in high speed operation 65Mbits per second in receive mode and about 80 Mbps in transmit mode. In full speed the transfer rate is 9 Mbps.However, data transfer rate from the host computer to the hardware board is only one factor that affects the performance of an image processing system designed according to a host/co-processor architecture, like the one studied here. There are also software issues to be taken into account both at the host end and at the Nios-II embedded processor side. For example, frame capturing and serialization prior to transferring are factors that limit frame rate in video applications. On the other hand, the Nios-II embedded processor controls the data flow following instruction code downloaded to embedded memory, as described in Section 5.So, the overall performance of the system depends on the fine-tuning of all these factors. The LabVIEW software allows for an efficient handling of array structures and also possesses image grabbing and vision tools that reduce processing time on the host side. Beside the above software limitations, there are also hardware issues related to an integrated System-on-a-Programmable-chip, like the time needed for Direct Memory Access (DMA) transactions between units. The performance of the hardware board is divided into the processing rates of the hardware filter co-processor and the performance ofthe rest of system, like external memory buffers and the interconnect fabric. This second factor adds an overhead depending on memory clocking and the structure of the interconnect units.

V. CONCLUSION

The design of a general hardware/software system based on a host computer and a FPGA co-processor was presented in this article. The system performance was studied in the case of image processing algorithms and represents a design methodology that can be used for a wide range of custom applications. It is based on a hardware board featuring a medium FPGA device, which is configured as a system on a programmable chip, with a software processor in the system control path. Other hardware components are local and on-chip memory allowing fast communication with a host computer. An integrated system controlled by a software processor provides significant flexibility for the transferringand processing of image data. An optimum choice of hardware/software features may accelerate the system performance. Partitioning a machine vision application between a host computer and a hardware co-processor may solve a number of problems and can be appealing in an academic or industrial environment where compactness and portability of the system is not of primal importance.

VI. REFERENCES

[1]. Yang Zhao,Yanguang Cai and Guobing Fan, Dynamical Behavior for Fractional-order Shunting Inhibitory Cellular Neural Networks,*Journal of Nonlinear Science and Applications, 9(6),* 2016, 4589-4599.
[2]. Yang Zhao,Research on Active Control of the Dynamic Vibration for Underwater Robot,*Journal of the Balkan Tribological Association*,22(1A),2016,pp770-779.
[3]. Yang Zhao,Yanguang Cai and Xiaojun Yang,A Local Fractional Derivative with Applications to Fractal Relaxation and Diffusion Phenomena,*Thermal Science*,20(S3),2016,pp723-727.
[4]. Liya Wang,Yang Zhao,Yaoming Zhouand Jingbin Hao,Calculation of flexible printed circuit boards (FPC) global and local defect detection based on computer vision,*Circuit World*,42(2), 2016, pp. 49-54.
[5]. Yandong Zhang and Yang Zhao,Design & implementation of an Air Quality Monitoring System for Indoor Environment based on Microcontroller,*International Journal of Smart Home*,9(11), 2015, pp. 301-312.