Hardware and Software Codesign for Computer Screen Image Processing Applications using FPGA

P. Sivarama Prasad
Associate Professor, ECE Department
Turbomachinery Institute of
Tech & Sciences
Hyderabad, INDIA

K. Srinivasa Rao. Ph.D.
Principal & Professor of ECE Department
TRR College of Engineering
Hyderabad, INDIA

ABSTRACT

The computer screen processing based machine learning algorithms is envisaged for future knowledge management applications. The front end based information retrieval is quite similar to human method of information access from computer. The proposed architecture is highly computational intensive and mostly performs image processing algorithms for extracting the information from the screen image. An FPGA based hardware accelerator is proposed for this application. The Xilinx Spartan-6 FPGA board is used for realizing morphological image processing modules along with Microblaze soft core. The Microblaze software performs the control operation and provides 100 Mbps Ethernet access to PC. The image processing modules are verified working at 100 MHz clock with chipscope occupying 70% of the selected Spartan-6 LX45 device along with Microblaze soft core.

Keywords

knowledge management, machine learning, computer screen processing, morphological image processing

1. INTRODUCTION

The Knowledge Management (KM) [1] is biggest challenge for organizations. Over time when people leave the company then previous gained knowledge cannot be useful to subsequently executed projects. At present the computers are only used to store the data. The new area of research is to implement machine learning algorithms, which learn out of previous executed projects by reading the reports and build intelligence to provide information based on quarries.

As most of the knowledge is in the form of files stored in computer, one potential way to access the data is to access the computer screen and extract information from it. This requires image processing algorithms built at high speed to process the computer screen data. This kind of information access from computer can be considered as front end access, which is quite similar to the access by humans. Conventionally computer based algorithms are designed to access defined data content from a specific file format. This can be referred as backend access which requires the full information about the information format and is limited by such category of files only. In the knowledge management problem context the envisaged machine learning approach must be able to see anything on computer screen and extract information from it. Hence backend access is not suitable. For example when a PDF file is opened on computer screen the backend access can only extract information, if the PDF file provides copy access to its content. But in front end mechanism even if, there is no copy access and also if it is scanned image the front end mechanism can extract the information. The advantage with proposed approach is the information retrieval mechanism is same irrespective of which type of file it is. The second main advantage of this kind of data access is we do not require any support from OS or the application which is supposed to open the data content.

The total solution for knowledge management of this category needs to solve problems in two challenging disciplines. The machine learning algorithms pertaining to knowledge management with support to query based information retrieval systems. The second discipline is extracting information from various resources and making it available to these algorithms. The present work focuses on second problem addressing the real time image processing and content extraction, for a typical requirement of computer screen processing applications.

This architecture may also be used in several other problem areas pertaining to image processing on computer screen data. The image processing problem for computer screen captured images is different from conventional image processing techniques, which is explained in subsequent sections. As the rate at which the elementary image processing operations need to be performed for this application is very high, there is a need for hardware accelerator.

The development of image processing applications on processor platform or FPGA platform is a big debate from last two decades. As proved with example in [5] selected applications implemented on Field Programmable Gate Arrays(FPGAs) are accelerated by as much as two or three orders of magnitude over the same application running on a processor. This acceleration on hardware with a clock speed that is approximately one tenth of the processor clock speed is achieved primarily by exploiting parallelism and pipelining on FPGA hardware.

The implementation aspects of linear and morphological operations are explained in [1]. The images are sent from MATLAB platform and after FPGA processing they are read back in PC for comparing the input and output images. The paper [5] presents several low level image processing accelarators. This paper presents a frame work for window based processing for a given image. The paper discussed both the cases processing still image and motion. A coprocessor based approach for fast image processing on Altera FPGAs with NIOS soft processor is presented in [2]. The proposed method uses USB for host to FPGA communication. The paper given in [3] describes the I/O bandwidth capabilities between host processor and FPGA based reconfigurable hardware accelerator platform.

2. LAYERED ARCHITECTURE – HOST PC & FPGA MODULES

2.1 Simple Hardware accelarator Vs Proposed scheme

The proposed approach uses a Microblaze based Image processing platform on Xilinx FPGAs which can provide services to Host PC. The approach suggested in [4][6][7][8] target to use FPGA only as platform for realizing some low level image processing operations, such as digital filters, edge detectors and transforms. In this case the whole control is from PC. In our approach a complete Hardware and Software platform shall be realized in FPGA so that it can do higher abstraction software processes independently without waiting for controls from host PC. The software part running on FPGA is realized through Microblaze software processor. The Microblaze soft processor is selected as it can be instantiated on any Xilinx FPGA board and has less system implementation cost. The hardware part of the FPGA shall consist of blocks for low level image processing functions such as FIR filters, FFT and morphological image processing operations. All these blocks are interfaced to Microblaze so that it can get the basic image processing operations done on given image in few clock cycles.

2.2 Ethernet Communication between PC and Microblaze

The Ethernet communication scheme is chosen between host PC and embedded platform so that the developed frame work is scalable allowing the host PC and PC running at distinct places. The scalability can be achieved by connecting multiple FPGA platforms to a single host through Ethernet. The Ethernet also ensures the host can also be connected over Intranet or Internet where the services are taken over TCP/IP. The choice of Ethernet is justified in contrast with what discussed in [3], because our approach proposes an intelligent platform with hardware and software elements in FPGA. Hence basic image data and required operations on it are only the main pay load on communication channel.

The low data handling and controlling requirements on Host PC side is relevant in the proposed application context, as the host PC need to run computationally intensive machine learning algorithms on the computer screen processed data, which is sent by FPGA.

3. Knowledge Management Application

The total application proposed for knowledge management is visualized to be developed in four layers as indicated in figure 1.

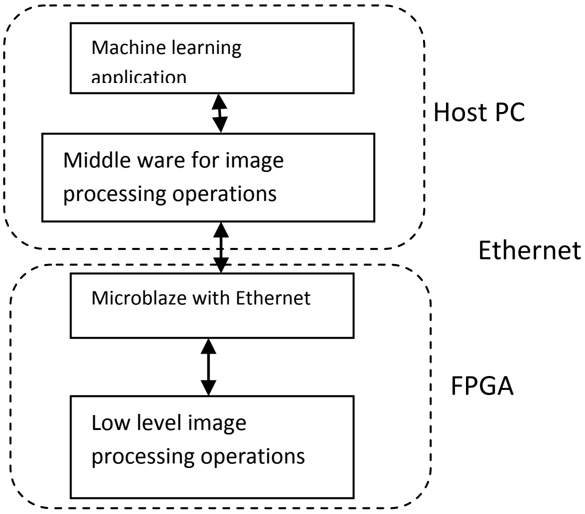

Machine learning application
Middle ware for image processing operations
Host PC
Ethernet
Microblaze with Ethernet
FPGA
Low level image processing operations

Fig 1: The layered implementation of the application

Layer H1: The first layer in host PC consists of core machine learning algorithms and contains data base management to archive the extracted information is standard formats. This layer tasks the middle ware (Layer H2) for taking the information from current screen data.

Layer H2: The middle ware layer responds to the layer H1 commands and performs few operations to control the applications on screen.

Layer F1: This layer implements soft processor core Microblaze with Ethernet running on Xilinx FPGA. This layer performs the image read based on the commands from middleware and provides services based on application program interfaces (APIs).

Layer F2: The Hardware implemented image processing block set enables the Microblaze to perform the complex image processing applications in few clock cycles.

As described earlier towards achieving the goal, the present work is to develop the FPGA modules (Layer F2) and profile them for speed and area requirements.

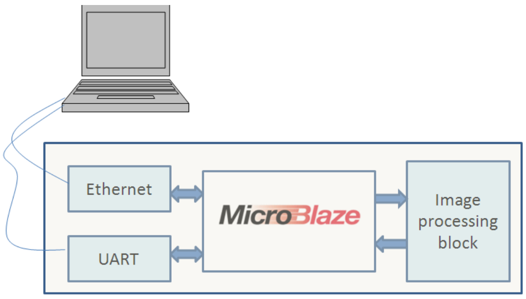
The proposed scheme considers the entire computer screen as image and accesses this from the video buffer. The image is preprocessed to result in 16 color image with same resolution. This image shall be of following size and will be transferred to FPGA core using Ethernet.

Image size in bits = NPH X NPV X 16 bits

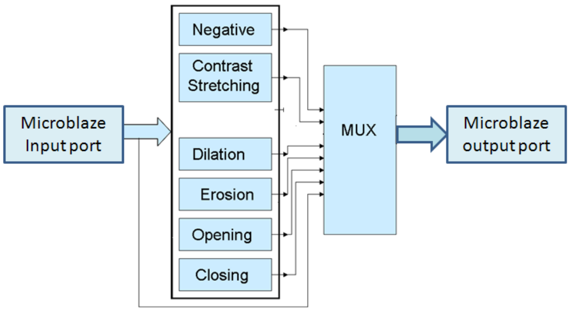
The targeted speed of operation of the screen is 1 modification per second. Hence the required bit rate is 12 Megabits/sec for 1024X768 screen resolution. This can be easily achieved by 100 Mbps LAN connectivity.

4. FPGA MODULES – HARDWARE AND SOFTWARE

The Microblaze[10] is used for establishing the processor core in the Xilinx Spartan 6 FPGA. The Microblaze is enabled with Ethernet core by which the data communication takes place. At present the morphological image processing modules are realized and interfacing with Microblaze is used to establish communication with PC. To enable the input and output ports to Microblaze GPIO ports are added in Microblaze. The figure 2, shows the details.



(a)



(b)

Figure 2. (a) High level block diagram

(b) Morphological processing blocks added to Microblaze ports

Mathematical morphology is a geometric approach to non linear image processing that was developed as a powerful tool for shape analysis in binary and grayscale images. Binary mathematical morphology is based on two basic operators: dilation and erosion. Both are defined in terms of the interaction of the original image A to be processed and the structuring element B. Morphological dilation is defined as the set union of the objects A obtained after the translation of the original image for each coordinate pixel b in the structuring element B. The set theory becomes the basic foundation mathematics for understanding morphological image processing.

4.1 Negative Image Operation

The data from Microblaze output port is given to Module which performs negative operation. Input in the form of gray scale format is taken from the pixel information in the form of 8 bit and that information is subtracted from maximum pixel value (i.e 255 or X"FF"). The resultant value is given as the output.

4.2 Contrast Stretching Operation

The image data is given to Module which performs Contrast Stretching Operation. The pixel data which is in the form of gray scale coded format, is given to the encoder (Comparator) which compares information (pixel) with pre-defined value(they are represented in diagram these value are in hexadecimal) and select which type of operation to be carried out and provides information to those block. Data is processed according to the logic in that block and generates the output given to the adder structure which work as a OR gate operation and generates pixel information. This technique is specially developed to implement on FPGA's only for Contrast Stretching. Figure 3, shows the RTL implementation of contrast stretching operation

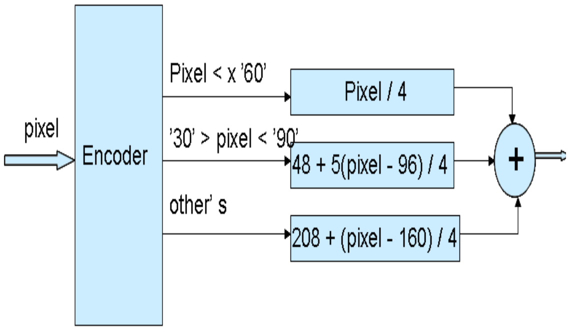


Fig 3: RTL blocks of contrast stretch operation

4.3 Dilation

The dilation of A by B then is the set of all displacements z, such that and A overlap by at least one element. The B is referred as structuring element.

$$A \oplus B = \left\{ z \mid \left(\hat{B}\right)_z \cap A \neq \varnothing \right\} \qquad (1)$$

The pixel data which is in the form of gray scale coded format is stored in the form of rows through row sels. Rows consists of Data banks which is equal to the dimensions (only width) of the image. The row sel first select row 1, and fills the data until it is full then selects row 2 for filling the data. Once row 2 is filled it fills the data in row 3 like wise once row 3 is filled it goes back to row 1 again and the process goes on till the image is completed. Each row as three columns. In each row which ever column has higher or maximum pixel value it is selected. Among the three rows the pixel having maximum value is taken as the resultant output. The figure 4, has the RTL implementation details for this module.

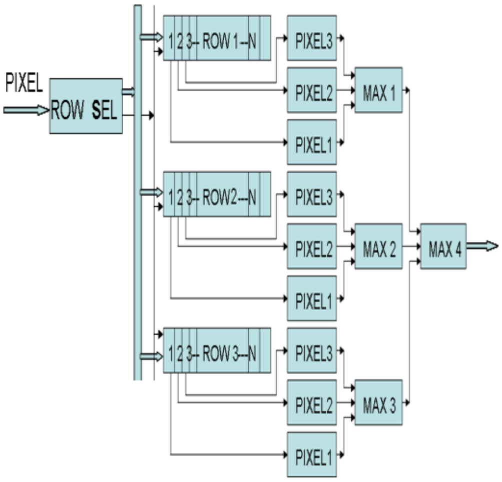


Fig 4: RTL block diagram for Dilation

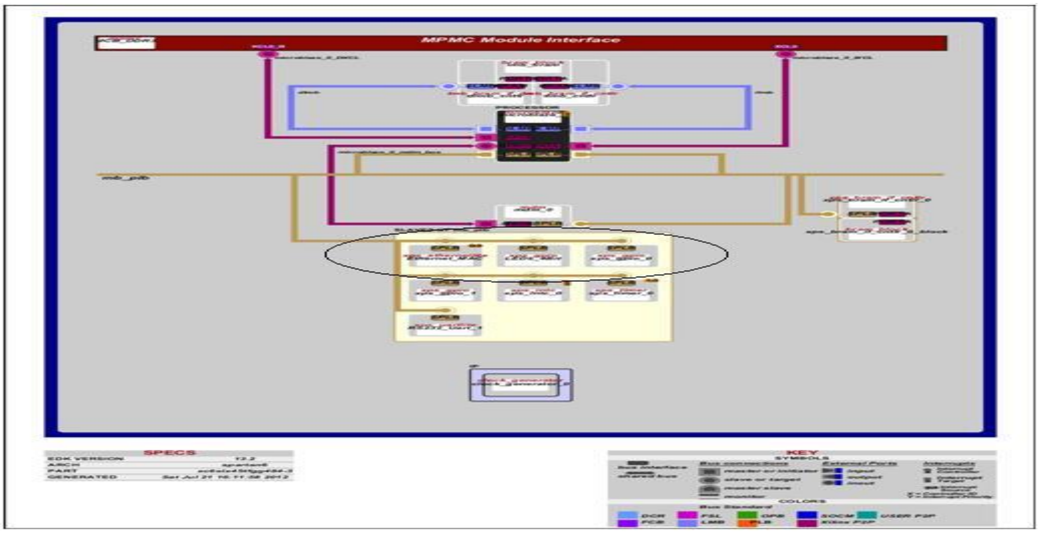


Fig 5: Ethernet Interface using Microblaze

4.4 Erosion

The erosion of A by B is set of all points z such that B, translated by z, is contained in A. The B will be referred as structuring element.

$$A \ominus B = \left\{ z \mid (B)_z \subseteq A \right\} \qquad (2)$$

The RTL implementation is same as shown in figure 5, except that maximum operator is replaced with minimum.

4.5 Opening and Closing

Both these operations are formed by using dilation and erosion. In opening, firstly image will be eroded and then it will be followed by dilation, where as in closing operation firstly image will be dilated and then followed by erosion.

5. SIMULATION, SYNTHESIS AND VERIFIATION

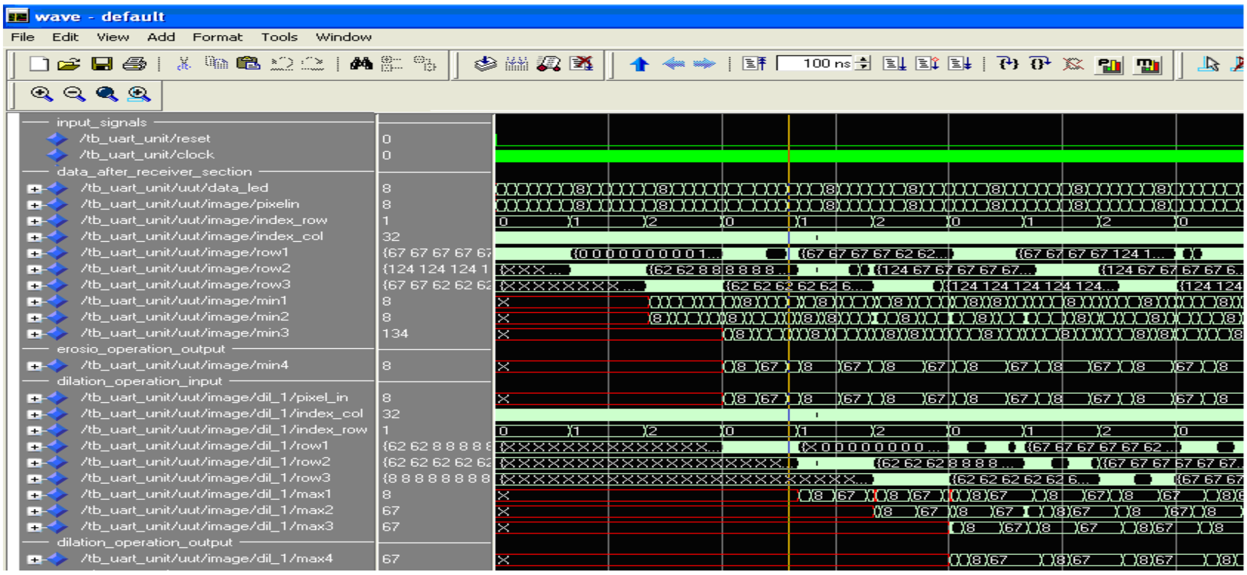
5.1 Ethernet interface using Microblaze

The MATLAB software is used on PC to transmit the samples images from PC. The Microblaze running Ethernet socket is configured as server to which the PC running client connects over TCP/IP is shown in figure 5.

5.2 Morphological Image processing

The simulation is performed Modelsim software and results are verified. As it is difficult to present all results here only the results for Erosion operation are presented here. The figure 6, Shows simulation results for opening operation.

Reset and clock are routed to all registers in design. The 8 bit parallel of pixel data is input to the module. Index 0 corresponds to first row and so on. The p00, p01, p03, p10, p11, p12, p20, p21, p22 are pixel values to be processed. Pipe_full is 1 when all the rows are filled with data. The pixelout data is the final processed data. The same results are obtained on chipscope tool while running on Spartan-6 FPGA.



```
input_signals
  /tb_uart_unit/reset                          0
  /tb_uart_unit/clock                          0
data_after_receiver_section
  /tb_uart_unit/uut/data_led                   8
  /tb_uart_unit/uut/image/pixelin              8
  /tb_uart_unit/uut/image/index_row            1
  /tb_uart_unit/uut/image/index_col            32
  /tb_uart_unit/uut/image/row1                 {67 67 67 67 67
  /tb_uart_unit/uut/image/row2                 {124 124 124 1
  /tb_uart_unit/uut/image/row3                 {67 67 62 62 62
  /tb_uart_unit/uut/image/min1                 8
  /tb_uart_unit/uut/image/min2                 8
  /tb_uart_unit/uut/image/min3                 134
erosio_operation_output
  /tb_uart_unit/uut/image/min4                 8
dilation_operation_input
  /tb_uart_unit/uut/image/dil_1/pixel_in       8
  /tb_uart_unit/uut/image/dil_1/index_col      32
  /tb_uart_unit/uut/image/dil_1/index_row      1
  /tb_uart_unit/uut/image/dil_1/row1           {62 62 8 8 8 8 8
  /tb_uart_unit/uut/image/dil_1/row2           {62 62 62 62 62
  /tb_uart_unit/uut/image/dil_1/row3           {8 8 8 8 8 8 8 8
  /tb_uart_unit/uut/image/dil_1/max1           8
  /tb_uart_unit/uut/image/dil_1/max2           67
  /tb_uart_unit/uut/image/dil_1/max3           67
dilation_operation_output
  /tb_uart_unit/uut/image/dil_1/max4           67
```

Fig 6: Simulation results of opening operation

The SP605 (Spartan 6 LX45T FPGA) board is used for hardware verification. All the implemented hardware blocks are profiled for speed and area. The table 1, shows the area occupancy and speed for the realized modules. Based on this analysis the total core is made to run 88 MHz which is the Microblaze operating clock.

TABLE I. Area and clock speed summary

Sl No	Area and Clock speed summary		
	Operator	*Area occupancy %*	*Max clock speed (MHz)*
1	Microblaze with Ethernet core	17	88
2	Negative image operation	0	175
3	Contrast streching operation	1	175
4	Dilation	6	115
5	Erosion	6	115
6	Opening	13	109
7	closing	13	109

*. As the slices are major blocks used in FPGA the slice utilization is given here

6. CONCLUSIONS AND FUTURE SCOPE

The FPGA based framework for realizing image processing algorithm is established. The morphological image processing blocks are realized in hardware with VHDL. At present the software running on Microblaze is only realized to perform Ethernet communication to PC and access the hardware image processing blocks. All modules are verified in simulation and subsequently in the Spartan 6 FPGA.

The present work will be extended as per the steps given below.

Realizing the linear image processing blocks and connecting them to Microblaze's GPIO ports.

Ethernet. The received text and information will be provided for high level application.

ACKNOWLEDGMENT

We acknowledge the support given by staff and faculty of ECE dept. TITS Hyderabad, for supporting in reviewing the work and providing valuable suggestions.

REFERENCES

[1] Tsung-Teng Chen 1, Maria R. Lee ," Revealing The Research Themes And Trends In knowledge Management Studies ", International Conference on Machine Learning and Cybernetics, pp 320 – 325, July 2008 .

[2] Ramirez, J.M.; Flores, E.M.; Martinez-Carballido, J.; Enriquez, R.; Alarcon-Aquino, V.; Baez-Lopez, D., "An FPGA-based Architecture for Linear and Morphological Image Filtering", 20th International Conference on Electronics, Communications and Computer (CONIELECOMP), pp 90 - 95 , Feb. 2010 .

[3] John A. Kalomiros , John Lygouras , "A host/co-processor FPGA-based Architecture for Fast Image Processing", Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, pp 373 – 378, September 2007.

[4] Miaoqing Huang', Olivier Serres, Sergio Lopez-Buedo , Tarek El-Ghazawil and Greg Newby, "An Image Processing Architecture To Exploit I/O Bandwidth On Reconfigurable Computers", 4th Southern Conference on Programmable Logic, ,pp257 – 260, March 2008.

[5] Haiqian Yu, Miriam Leeser, "Automatic Sliding Window Operation Optimization for FPGA-BasedComputing

Boards”, 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, FCCM '06, pp 76 – 88, April 2006.

[6] Griselda Saldaña1, Miguel Arias-Estrada2, “A low-level image processing algorithms accelerator platform”, 18th International Conference on Electronics, Communications and Computers, pp 117 – 122, March 2008.

[7] Jan Kloub, Petr Honz´ık and Martin Danˇek, “Reconfigurable Hardware Objects for Image Processing on FPGAs”, IEEE 13th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS), pp 121 – 122, April 2010.

[8] [12] Ramirez, J.M.; Flores, E.M.; Martinez-Carballido, J.; Enriquez, R.; Alarcon-Aquino, V.; Baez-Lopez, D., “An FPGA-based Architecture for Linear and Morphological Image Filtering”, 20th International Conference on Electronics, Communications and Computer (CONIELECOMP), pp 90 - 95 , Feb. 2010 .

[9] [14] Marek Gorgon, Jaromir Przybylo, “FPGA based controller for heterogenous image processings ystem” Euromicro Symposium on Digital Systems, Design, pp 453 – 457, August 2002.

[10] www.xilinx.com/support/documentation/sw.../mb_ref_guide.pdf