International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (01), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

Hardware Implementation of Real time Image Processing on FPGA

Ms. Aayushi Jain[1] , Prof. Sunil Shah[2]

[1]*M Tech (Embedded System and VLSI Design)*

Gyan Ganga Institute of Science and Technology, Jabalpur (India)

[2]*Deptt of Electronics and Communication*

Gyan Ganga Institute of Science and Technology, Jabalpur (India)

ABSTRACT

Most of the image processing lgorithms are sequential in nature. High-level languages like Mat-lab, C++, Open-CV etc. are the common platforms to develop and validate such algorithms. These platforms are most suited in applications, which does not have any time restrictions. In other words, where the response time is not so important. In real-time systems, the high level platforms will not be a good choice due to time and resource constraints. The hardware architecture presented in this work is suitable for the efficient implementation of machine vision systems. This architecture supports robust, high speed, low latency, and low power smart camera applications. Configuring the imaging sensor for a reduced synchronization overhead can either increase the maximum frame speed and, simultaneously, reduce the latency or reduce the power consumption at a maintained frame speed. The majority of the dissipated dynamic power stems from the clock nets. Aggressive parallelization of computation and memory accesses maintaining the clock nets at a lower frequency would appear to be a good strategy with regards to achieving a low dynamic power for DSP systems on an FPGA. Static power can be controlled by selecting an FPGA device which has a size that matches the size of the application.

Keywords—Image Processing Using FPGA, hardware implementation Digital Image Processing, VHDL Image processor;

I. MOTIVATIONS FOR HARDWARE IMAGE PROCESSING

As explained in the previous section, a micro-controller/dsp processor executes algorithm sequences sequentially. If multiple hardware circuits can be designed to carry out different algorithm sequences in parallel, there will be considerable increase in overall execution speed. Suppose a system has to be designed such that the brightness of the incoming frames has to be increased. Brightness of an image can be increased by multiplying each pixel gray level with a constant α, and then adding a gain constant β to it as in equation1.1

$$g(i, j) = f(i, j) \times \alpha + \beta$$

In a typical Micro-Controller/DSP processor based design, this will involve storing the frames in a buffer, and then performing the operations mentioned in equation1.1 to each pixel gray level, in a loop. Suppose each addition instruction takes 12 clock cycle, and each multiplication instruction takes 36 clock cycle, then total number of clock cycles required to process one pixel will be 48. If the incoming frames are of size 100 × 100, then such a design will need 100 × 100 × 48 clock cycles to process the entire frame. Now suppose, there are 10000 adder and multiplier circuits, one cosponsoring to each pixel. In such a design, all the pixels can be processed in parallel. Thus total operation can be implemented in just two clock cycles. In principle, such a system will be 12000 times faster than that of a micro controller/dsp processor based system. In actual designs, algorithm will be divided in to parallel blocks and will be executed simultaneously.

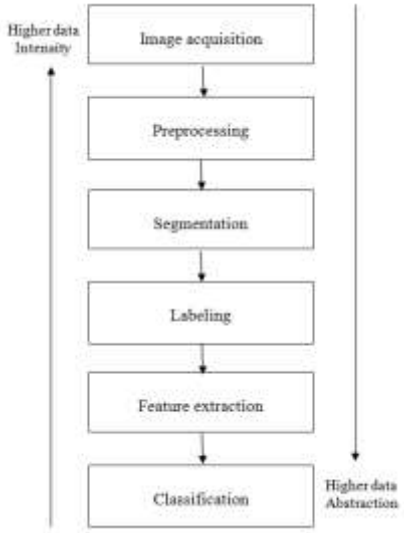

Fig 1 Fundamental steps of a image processing system

In a nutshell, the significant increase in processing speed is the major motivation behind hardware image processing. If the processing time is less, the power consumption also will be reduced. Hence it can be concluded that hardware image processing systems give better performance in time critical applications. In the current scenario, most of the image processing algorithms are running in a sequential environment. Hence a research in FPGA based image processing has grater significance and scope in time critical applications.

II. AFFILIATED WORK

Using FPGA and DSP structure can well solve real-time image acquisition system requirements; The image data of the dim place is based on the logarithmic stretching algorithm, which can effectively enhance the image, and make the uneven distribution of the image become clear;The parallel JPEG compression algorithm is used to block the two-dimensional images before FPGA sending data, and the processing time is reduced; Although the compression algorithm of JPEG can reduce the amount of the information greatly, it can preserve the details of the original image and achieve the expected target.

There are some solutions which gives scalable modular hardware solution for real-time Face Recognition (FR) on large databases. Existing hardware solutions use algorithms with low recognition accuracy suitable for real-time response. In addition, database size for these solutions is limited by on-chip resources making them unsuitable for practical real time applications. Due to high computational complexity we do not choose algorithms in literature with superior recognition accuracy. Instead, we come up with a combination of Weighted Modular Principle Component Analysis (WMPCA) and Radial Basis Function Neural Network (RBFNN) which outperforms algorithms used in existing hardware solutions on highly illumination and pose variant face databases. We propose a hardware solution for real-time FR which uses parallel streams to perform independent modular computations.

One more solution for real-time face recognition is to address problems of low recognition accuracy and small database size support in existing solutions. The combination of WMPCA and RBFNN which shows better recognition accuracy on images with considerable variations in pose and illumination. A modular hardware solution for the algorithm to support high frame rate requirements. The proposed architecture shows scalability with respect to database size and image dimensions due to availability of large external memory. A novel storage format on external memory is followed to minimize memory latencies. The FPGA emulation for the hardware solution is able to perform face recognition on images of dimension 128×128 at 450 recognitions per second with 450 classes of face images.

While performing more and more tasks on software the energy requirement of the vision sensor node is increased. Hence we will avoid a task partitioning strategy having more modules in software implementation. Similarly, shifting more tasks to hardware results in increased hardware cost, as well as increased design and development time. We are considering that partitioning tasks between hardware and software at the vision sensor node affects the energy requirement of the vision sensor node.

A Tone Mapping Operator (TMO) which adjusts the High Dynamic Range (HDR) of image sensor data to the limited dynamic range of conventional displays with analog signal processing. It is based on Photographic Tone Reproduction (PTR) and suitable for analog circuit design in a CMOS image sensor in order to reduce the hardware cost and operation time for real-time image processing. The proposed analog TMO does not require access to any other pixel of the image sensor and processes in the analog domain at the focal plane. Furthermore, the proposed TMO also provides a well tone mapped image quality depending on a suitable calibration and user settings. It specially benefits for customer-tailored applications, which strongly require a real time processing with low power consumption, e.g., security monitoring, traffic monitoring, advanced drive system, medical technology, etc.

III.THE ADVANTAGE OF USING FPGAs

Image processing is difficult to achieve on a serial processor. This is due to the large data set required to represent the image and the complex operations that need to be performed on the image. Consider video rates of

25 frames per second, a single operation performed on every pixel of a 768 by 576 color image (Standard PAL frame) equates to 33 million operations per second. Many image processing applications require that several operations be performed on each pixel in the image resulting in an even large number of operations per second. Thus the perfect alternative is to make use of an FPGA. Continual growth in the size and functionality of FPGAs over recent years has resulted in an increasing interest in their use for image processing applications.

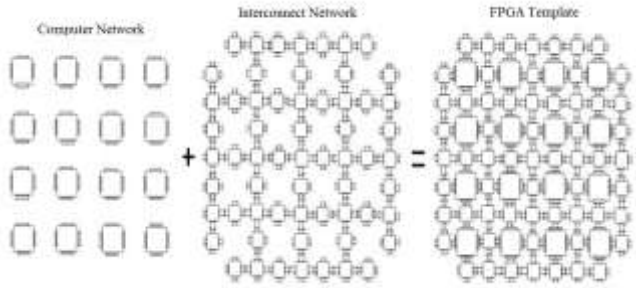

Fig 2 Programmable Logic Blocks of an FPGA

The main advantage of using FPGAs for the implementation of image processing applications is because their structure is able to exploit spatial and temporal parallelism. FPGA implementations have the potential to be parallel using a mixture of these two forms. For example, the FPGA could be configured to partition the image and distribute the resulting sections to multiple pipelines all of which could process data concurrently. Such parallelization is subject to the processing mode and hardware constraints of the system.

In Figure 1, an FPGA consists of a matrix of logic blocks that are connected by an interconnect network. Both the logic blocks and the interconnect network are reprogrammable allowing application specific hardware to be constructed, while at the same time maintaining the ability to change the functionality of the system with ease. As such, an FPGA offers a compromise between the flexibility of general purpose processors and the hardware-based speed of ASICs.

Hardware Constraints

There are three modes of processing: stream, offline and hybrid processing. In stream processing, data is received from the input device in a raster nature at video rates. Memory bandwidth constraints dictate that as much processing as possible can be performed as the data arrives. In offline processing there is no timing constraint. This allows random access to memory containing the image data. The speed of execution in most cases is limited by the memory access speed. The hybrid case is a mixture of stream and offline processing. In this case, the timing constraint is relaxed so the image is captured at a slower rate. While the image is streamed into a frame buffer it can be processed to extract the region of interest. This region can be processed by an offline stage which would allow random access to the region's elements.

Timing Constraints

If there is no requirement on processing time then the constraint on timing is relaxed and the system can revert to offline processing. This is often the result of a direct mapping from a software algorithm. The constraint on bandwidth is also eliminated because random access to memory is possible and desired values in memory can be obtained over a number of clock cycles with buffering between cycles. Offline processing in hardware therefore closely resembles the software programming paradigm; the designer need not worry about constraints to any great extent. This is the approach taken by languages that map software algorithms to hardware. The goal is to produce hardware that processes the input data as fast as possible given various automatic and manual ptimization techniques.

Bandwidth Constraints

Frame buffering requires large amounts of memory. The size of the frame buffer depends on the transform itself. In the worst case (rotation by 90º, for example) the whole image must be buffered. A single 24-bit (8-bits per color channel) color image with 768 by 576 pixels requires 1.2 MB of memory. FPGAs have very limited amounts of on-chip RAM. The logic blocks themselves can be configured to act like RAM, but this is usually an inefficient use of the logic blocks. Typically some sort of off-chip memory is used but this only allows a single access to the frame buffer per clock cycle, which can be a problem for the many operations that require simultaneous access to more than one pixel from the input image. For example, bilinear interpolation requires simultaneous access to four pixels from the input image. This will be on a per clock cycle basis if real-time processing constraints are imposed.

Parallelism

Most of the image processing algorithms have inherent parallelism in them. The processing speed can be improved by executing the sequences concurrently. In principle, all the algorithm sequences can be implemented in a separate processor. But if each step depends on previous algorithm sequences, the processors will have to wait for the results from previous stages. Thus the reduction in response time of the system will be very little. For practical implementations in a parallel architecture, algorithm should have significant number of parallel operations. This is Known as Amdahl's law [1] . Let 's' be the proportion of total number sequences in an algorithm, that has to be executed sequentially. Let 'p' be the proportion of the algorithm that can be executed in parallel, using N different processors. Then the best possible speed up that can be obtained is given by equation1.

$$Speedup \leq \frac{s+p}{s+\frac{p}{N}} = \frac{N}{1+(N-1)s}$$

The equality can only be achieved if there are no additional overhead like communication, introduced as a result of conversion of sequential algorithm to a parallel one. In practical scenario, the actual speed up will be always less than the number of processors N. As N increases, the execution speed also increases. Ideally, if N tends to infinity, the over all execution speed of the algorithm depends solely on the proportion of the algorithms, that has to be executed sequentially. This is given in equation

$$\lim_{N\to\infty} Speedup = \frac{1}{s}$$

Thus to achieve significant speed-up, the proportion of algorithms which can be executed in parallel should be more. Most of the image processing algorithms are parallel in nature.

IV. IMPLEMENTATION USING FPGA

An FPGA based design is inherently parallel in nature. Different algorithm sequences will be mapped to different hardware modules in a FPGA, which operates concurrently. The main reasons for choosing FPGA as an embedded image processing platform are as given below.

- Parallel operation
- Speed of execution
- Flexibility
- Low power design

Further research will investigate the CMOS-integrated circuit design of the proposed analog TMO. The analog circuit noise, inaccuracies and overflow problems, specially in arithmetical circuits, will be considered in circuit design. Furthermore, a stage-pipeline for speed-up of the analog computing could also be implemented as an enhancement to this study at the next step.

In this proposal we describes a hardware architecture for real-time image component labeling and the computation of image component feature descriptors. These descriptors are object related properties used to describe each image component. Embedded machine vision systems demand a robust performance and power efficiency as well as minimum area utilization, depending on the deployed application. In the proposed architecture, the hardware modules for component labeling and feature calculation run in parallel. A CMOS image sensor will be used to capture the images. The architecture will be synthesize and implement on a Xilinx virtex-6 FPGA. The developed architecture will be capable of processing as much as 350 video frames per second of size 640 × 480 pixels. Dynamic power consumption will be maintained to be the minimum.

V. SUMMARY

FPGAs are often used as implementation platforms for real-time image processing applications because their structure can exploit spatial and temporal parallelism. Such parallelization is subject to the processing mode and hardware constraints of the system.

Using high-level languages and compilers to hide the constraints and automatically extract parallelism from the code does not always produce an efficient mapping to hardware. The code is usually adapted from a software implementation and thus has the disadvantage that the resulting implementation is based fundamentally on a serial algorithm.

Manual mapping removes the 'software mindset' restriction but instead the designer must now deal more closely with timing, resource and bandwidth constraints, which complicate the mapping process Timing or processing constraints can be met using pipelining. This only adds to latency rather than changing the algorithm, which is why automated pipelining is possible. Meeting bandwidth constraints on the other hand is more difficult because the underlying algorithm may need to be completely redesigned, an impossible task for a compiler. This paper presented some general techniques for evaluating complex expressions to help deal with resource constraints by reducing logic block usage.

Window operations require local caching and control mechanisms but the underlying algorithm remains the same. Global operations such as chain coding require random access to memory and cannot be easily implemented under stream processing modes. This forces the designer to reformulate the algorithm.

VI. CONCLUSION

Object identification requires the use of an efficient signal processing system. Although processing is achievable on serial processors, it can be beneficial to take advantage of the parallelism, low cost, and low power consumption offered by FPGAs. The successful implementation of this image processing algorithm illustrates that the digital signal processing required for high rate sensing application can be efficiently implemented on FPGA hardware.

The gray scale transformation has been used to remove the coherence of the background and the target to be tracked. The Delta Frame-based segmentation and Thresholding combine two intensive operations into one step, eliminating the need for large numbers of parallel comparators. The resulting optimized enhanced image fits on a small FPGA, such as the Xilinx Spartan- III XC3S500E, with sufficient resources available for an application to make use of the derived tracking information.

REFERENCES

[1.] H. C. van Assen, H. A. Vrooman, M. Egmont-Petersen et al."Automated calibration in vascular X-ray images using theaccurate localization of catheter marker bands," *InvestigativeRadiology*, vol. 35, no. 4, pp. 219–226, 2000.

[2.] D.Meng, C. Yun-feng, andW. Qing-xian, "Autonomous craters detection from planetary image," in *Proceedings of the 3rd International Conference on Innovative Computing Information and Control (ICICIC '08)*, June 2008.

[3.] C. Steger, M. Ulrich, and C. Wiedemann, *Machine Vision Algorithems and Applications*, Wiley-VCH, New York,NY,USA,2008.

[4.] D. G. Bailey, C. T. Johnston, and N. Ma, "Connected components analysis of streamed images," in *2008 International Conference on Field Programmable Logic and Applications, FPL*,pp. 679–682, September 2008.

[5.] M. G. Pinheiro, "Image descriptors based on the edge orientation," in *Proceedings of the4th InternationalWorkshop on Semantic Media Adaptation and Personalization (SMAP '09)*, pp. 73–78, December 2009.

[6.] T. Jabid, M. H. Kabir, and O. Chae, "Local Directional Pattern (LDP)—a robust image descriptor for object recognition," in *Proceedings of the 7th IEEE International Conference on Advanced Video and Signal Based (AVSS '10)*, pp. 482–487, Boston, Mass, USA, September 2010.

[7.] A. A. Mohamed and R. V. Yampolskiy, "An improved LBP algorithm for avatar face recognition," in *Proceedings of the 23rd International Symposium on Information, Communication and Automation Technologies (ICAT '11)*, Sarajevo, Bosnia and Herzegovina, October 2011.

[8.] Y. Yoon, K.-D. Ban, H. Yoon, and J. Kim, "Blob extraction based character segmentation method for automatic license plate recognition system," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '11)*, pp. 2192–2196, Anchorage, Alaska, USA, October 2011.

[9.] C. T. Johnston and D. G. Bailey, "FPGA implementation of a single pass connected components algorithm," in *Proceedings of the 4th IEEE International Symposium on Electronic Design, Test and Applications (DELTA '08)*, pp. 228–231, Hong Kong, China, January 2008.

[10.] N. Ma, D. G. Bailey, and C. T. Johnston, "Optimised single pass connected components analysis," in *Proceedings of the International Conference on ICECE Technology (FPT '08)*, pp. 185–192, December 2008.

[11.] K. Benkrid S Sukhsawas D Crookes and A. Benkrid, "An FPGA-Based Image Connected Component Labeller," in *Field Programmable Logic and Application*, vol. 2778 of *Lecture Notes in Computer Science*, pp. 1012–1015, 2003.

[12.] M. Jabło´nski and M. Gorgo´n, "Handel-C implementation of classical component labelling algorithm," in *Proceedings of the EUROMICRO Systems on Digital System Design (DSD '04)*, pp. 387–393, September 2004.

[13.] M. Mylona, D. Holding, and K. Blow, "DES developed in handel-C," in *Proceedings of the London Communications Symposium*, 2002.

[14.] F. Chang, C.-J. Chen, and C.-J. Lu, "A linear-time component labeling algorithm using contour tracing technique," *Computer Vision and Image Understanding*, vol. 93, no. 2, pp. 206–220, 2004.

[15.] M. F. Ercan and Yu-Fai Fung, "Connected component labeling on a one dimensional DSP array," in *Proceedings of the IEEE Region 10 Conference (TENCON '99)*, vol. 2, pp. 1299–1302, December 1999.

[16.] H. C. van Assen, M. Egmont-Petersen, and J. H. C. Reiber, "Accurate object localization in gray level images using the center of gravity measure: accuracy versus precision," *IEEE Transactions on Image Processing*, vol. 11, no. 12, pp. 1379–1384, 2002.

[17.] Digillent Incorporation, http://www.digilentinc.com/.

[18.] Aptina Imaging Corporation, http://www.aptina.com/.

[19.] Visual Applets at SILICONSOFTWARE GmbH, http://www.silicon-software.info/en/.