Review on HDL Implementation of Digital Image Display on VGA

Rohit Raj
Student/Research Scholar
BGIET, Sangrur
Distt. Sangrur-148001, India

Monika Aggarwal
Associate Prof.
BGIET, Sangrur
Distt. Sangrur-148001, India

Gaurav Mittal
Assistant Prof.
BGIET, Sangrur
Distt. Sangrur-148001, India

ABSTRACT
Image processing is always a very instance application while working on embedded platforms. It is still a very challenging task. To accelerate this image processing applications, Field-programmable gate arrays (FPGAs) offer a suitable platform as a customized hardware to accelerate this. The recent image processing algorithms are concentrated on pixel-based modules for simple preprocessing tasks. And these tasks are mainly defined using MATLAB. This paper deals with the implementation of image/digital captured data into real time hardware using HDLs with the motto of relatively inexpensive and adaptable technology. Thus, it offers modules and interfaces to perform operations and incorporate software defined operations. This paper shows the digital image (from any source/formats) can be stored into a memory (RAM/ROM) onto the FPGA and then it can be further processed for larger display devices i.e. on VGA monitors.

Keywords
FPGA, Simulation, Synthesis, Core generation, VGA, Xilinx ISE, HDLs.

1. INTRODUCTION
The efficiency and capability of available FPGAs are growing exponentially in the recent decades. Today, low or mid range devices allow the implementation of Networks on chip (NoC) or even complex Systems on Chip (SoC) on a FPGA fabrics. The Field Programmable Gate Arrays (FPGAs) offers the great flexibility to generate the required hardware structures. These hardware structures are so optimized that it simply fit the given algorithm in order to achieve the efficiently parallelism working environment. In addition the growing capacities make FPGAs a dominated platform with the capability of integration of even more complex image processing tasks in the near future. Thus it makes the Field Programmable Gate Array (FPGAs) to become an alternative for the software algorithms implementation. The major example of this is in many applications from video surveillance to remote sensing image applications. Field programmable gate arrays are the large-scale integrated circuit that can be reconfigured where "field programmable" refers to ability of changing the operation of the device as per the application demands.

As previously mentioned, the implementations of real-time image processing algorithms can be done on general purpose microprocessors or microcontrollers. Due to the high computational density and parallel processing capability of the FPGAs it has a major impact on image or video processing when compared to a general purpose microprocessor or microcontrollers. If both of these features coupled together with the ability of FPGA of being reprogrammable it leads to the flexibility in the development of image processing algorithms on FPGA. That's why during the recent years FPGAs have become the dominant form of programmable logic implementation.

The presented work has been divided into several tasks which typically start from the storing of an image which might be taken from any source such as HDDs, MicroSDs etc. or any other media file in BMP, GIF or Jpeg format. The next step is to convert that image information into digital information form which is said to be a coefficient file, for this we must have some familiarity with MATLAB. And at the last stage, the previous gathered information would be sent to store into a FPGA ROM. For this purpose, Xilinx Core generator is best tool that delivers the cores optimized for Xilinx FPGAs. The CORE Generator is mainly used to create high density, high performance designs in Xilinx FPGAs in less time. The CORE Generator is comes with almost all of the XILINX ISE version(s) with an extensive library of Xilinx LogiCORE Intellectual Property Block. These could include DSP functions, memories, storage elements, math functions and a variety of basic elements to high speed transceivers.

2. RELATED WORK
Image processing on embedded platform is still a very challenging work as stated by **Michael Schaeferling et. al. (2015),** [1]. His work basically related to the development of frame work for image processing with FPGAs. They also stated that major available image processing frameworks for FPGAs concentrated only on pixel-based modules for simple preprocessing tasks. His work presents the framework which also aims to cover the integration of higher-level algorithms. Thus, his work offers modules and interfaces to perform window-oriented filter operations and incorporate software defined operations. They also incorporated several complex and higher-level algorithms, such as undistortion and rectification; natural feature description, edge detection and Hough transform have been adopted and integrated into the frameworks image processing implementation. His work completely describes the framework with several interfaces and some of the existing modules used. Finally, some applications which were already implemented using this framework are also presented.

In another research given by **Mariangela Genovese and Ettore Napoli (2014),** [2] as per the background identification/clarification is a most common feature in many video processing systems. There work proposed two hardware implementations of the Open CV version of the Gaussian mixture model (GMM), a background identification algorithm. There implemented versions of the algorithm allows a fast initialization of the background model while an hardware-oriented, innovative and the formulation of the GMM equations makes the proposed circuits able to perform real-time background identification on high definition (HD) video sequences. The proposed circuit is designed with commercial field-programmable gate-array (FPGA) devices

as target at first with implementation on Virtex6 vlx75t, the proposed circuit process 91 HD fps (frames per second) and uses 3% of FPGA logic resources while the second circuit is oriented to the implementation in UMC-90 nm CMOS standard cell technology.

There was another work presented by **Carlos Gonzalez et. al. (2013),** [3] on hyper spectral imaging that is a growing area in remote sensing in which an imaging spectrometer that collects hundreds of images (at different wave lengths) for the same area on the surface of the Earth. These images are extremely high-dimensional, and require most advanced on-board processing algorithms able to satisfy near real time constraints in applications such as mapping of oil spill, wild land fire monitoring and sand chemical contamination, etc. The most widely used techniques for analyzing hyper spectral images is spectral unmixing, which allows for sub pixel data characterization. Since the available spatial resolution in hyper spectral images is typically of several meters, and therefore it is affordable to assume that several spectrally pure substances (called end members in hyper spectral imaging terminology) can be found within each imaged pixel, thus it is particularly very important. In this work they explore the role of hardware accelerators in hyper spectral remote sensing missions and further provided the comparison of the two types of solutions: field programmable gate arrays (FPGAs) and graphics processing units (GPUs). They had implemented and tested the full spectral unmixing chain, using both types of accelerators, in the context of areal hyper spectral mapping application using hyper spectral data collected by NASA's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS). There work gives a thoughtful impression on the capability and existing challenges of applying these types of accelerators in remote sensing missions, indicating that the reprogram ability of FPGA systems and the low cost of GPU systems may open various new perspectives toward fast on-board and on-the-ground processing of hyper spectral images used in remote sensing applications.

One another different image processing approaches are provided by **Horace Josh et. al., (2013),** [4] that for the Monash Vision Group prosthetic vision device. In a companion work they described that is developing a bionic eye based on the implantation of stimulation tiles on the primary visual cortex. And they lead up to an expected 2014 first in-human trial of this device, potential image processing techniques and intuitive user interfaces need to be developed and checked. The main bottleneck of this approach is the (often reduced) bandwidth connection between the station and satellite, which drastically limits the information that can be sent and processed in real time by **Carlos González et. al., (2012),** [5]. They provide the possible way to overcome this problem by including onboard computing resources able to preprocess the data, reducing its size by magnitude orders. Thus Reconfigurable field-programmable gate arrays (FPGAs) are most promising platform that allows hardware /software codesign and the potential to provide powerful onboard computing capability and flexibility at the same time of frame. And also FPGAs can implement custom hardware solutions; they can reach very high performance levels of degree.

3. METHODOLOGY

Image/Video processing algorithms are normally classified into one of three modes: started with Low-level algorithms always operate on individual pixels or neighborhoods pixel data especially for concurrency. Then going upto intermediate-level algorithms either convert/change pixel data into a different representation depending on the application need, such as coordinate or chain code, a histogram, or to operate on one of any higher representations that previously done. And at last the High-level algorithms aim to extract meaning from the image using information from the previous defined levels. This may be in the form of adding, rejecting a component or identifying where an object is within an image. While coming from low to the high-level data representations there are a significant decrease in credulous parallelism due to the change from pixel data to more informative representations. And also there is a reduction in the sufficient amount of data that must be processed, thus allowing more time to do the processing.

The FPGAs are most adaptable, due to their internal structure, and provide ease of use for computationally intensive tasks which form the vast majority of low and intermediate-level operations for image/video processing. Due to this the large data sets and regular repetitive nature of the operations can be easily imposed. For this reason it has been become tradition in most of the systems for the FPGAs to handle the low-level operations and then pass to the microprocessor or microcontroller to process the data to which then executes the high-level operations. Now with the advent of newer technologies, the size of FPGAs are increasing relatively, so it is now possible to implement processor architectures on the reconfigurable fabric, which means that the FPGA can form the core of the system which can be further used in any processing applications.

The application of image or video processing in FPGAs is a rapidly growing research area due to the recent increases in the power and structures of these programmable devices. Although there are various challenges to implement the working algorithms despite of the potential gains in speed, make it an attractive topic for research in image/video processing. Due to these factors there is a rapid growth in the field of VLSI especially while dealing with FPGAs, mostly newcomers consider simply porting an existing software algorithm to an FPGA implementation.

This Project basically aims to help those researchers who wishing to use FPGAs to accelerate image/video processing algorithms through some of the basics, and provide a range of techniques that result in an efficient implementation, both algorithmatically, computationally and in terms of resource requirements and utilizations. This work will start from the basics of the image that can be found in any type of storage media or system file(s) etc.

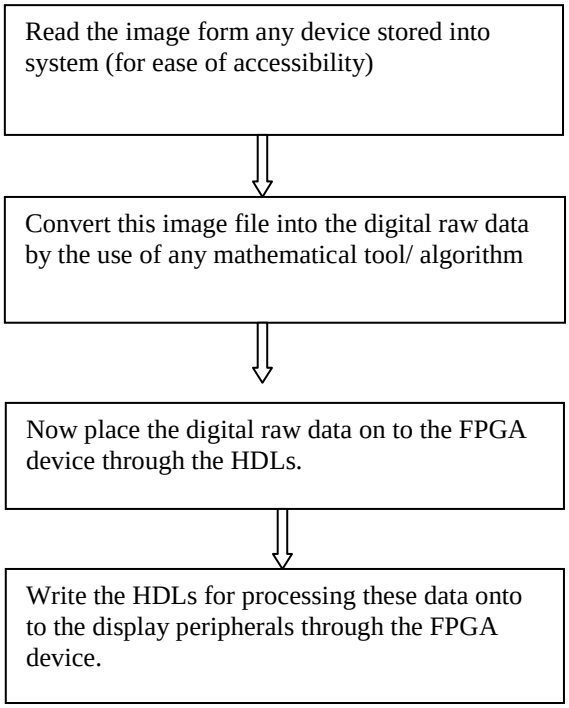

Fig. 1: Block Diagram of Proposed Work.

The steps involved into this work would be like below:

- Select an image from a media file stored into system (for ease of access) in either JPEG/BMP/ GIF format.

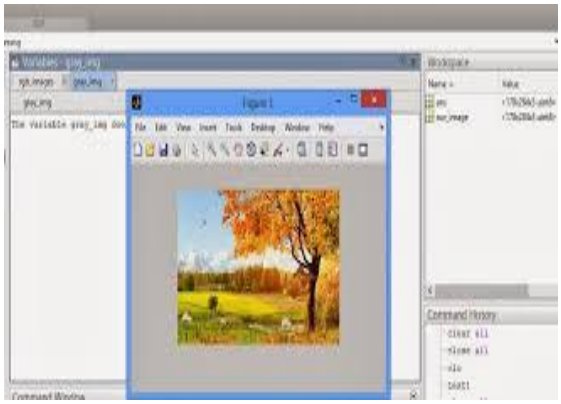

Fig. 2: Image read from MATLAB

- Now the second step would be involved in writing the algorithm for converting these image data into the digital data as the most HDLs behaves very well with digital information.
- There are various algorithm / procedure available to perform this in this particular work we would prefer MATLAB to convert those image information into the raw digital data which is often said to as the "Coefficient File" with extension ".COE" for Xilinx FPGAs and ".MIF" for Altera FPGAs.

```
memory_initialization_radix=16;
memory_initialization_vector=
201F, A21F, E420, 6681, 1801, 5A2F, 993F, 301D,
7283, B81D, 201E, C000, F025, 201C, 0201, F025,
F025, 0000, 0000, 0000, 0000, 0000, 0000, 0000,
0000, 0000, 0000, 0000, 0000, 0000, 0000, 0000,
1234, 3022, 4321, DEAD, 1111, DEAD, DEAD, 3028,
DEAD, 300D, 0001;
```

Fig. 3: COE format of an Image

- After generating the Coe file of an image the next step is to provide these information into the FPGA ROM. For this purpose we will Xilinx Core generator will be use. Xilinx core is nothing but a Logic Core System that generates and delivers cores optimized for targeted Xilinx FPGAs. CORE Generator is mainly used to create high density, high performance designs in Xilinx FPGAs in very less time.

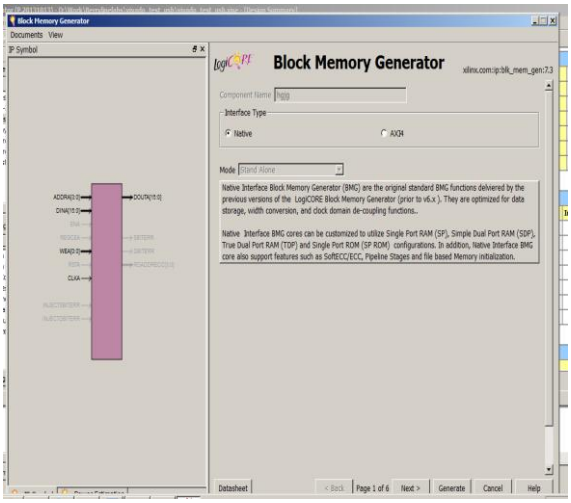

Fig.4: Core Generator block form Xilinx ISE

- Now the foremost step would be to write the HDL for VGA connector.

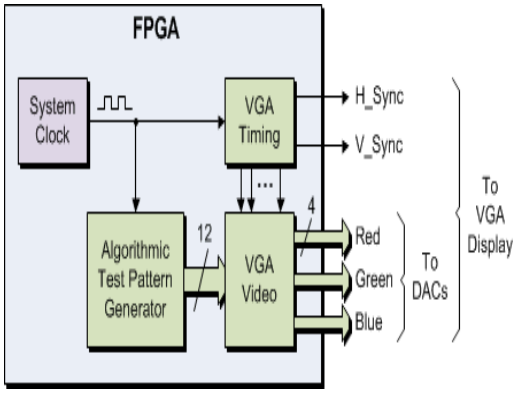

Fig.5: FPGA to VGA connector signal interfacing

4. CONCLUSSION

The major tasks as shown in previous sections are shown in very easiest format. But now the whole system has to be written into HDLs so that it can perform very well while implemented onto the FPGAs. The complete system can be implemented on any FPGA devices but in our case as per our

availability we are going to use XC3s500e FPGA starter board who has the on board facility of VGA connector peripherals along with different set of DIPs and Push switches.

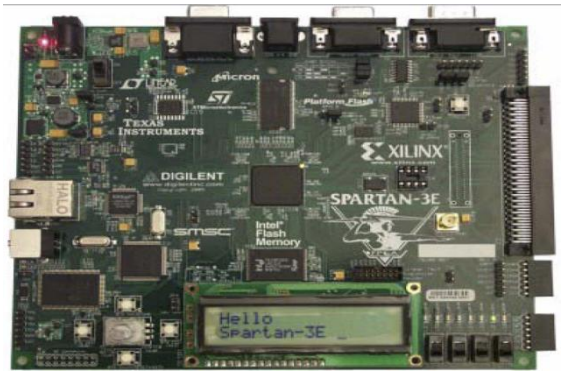

Fig.6: Target FPGA board/device

This work would be go through various tools and different levels of HDLs code writing which helps in the upcoming research work. Also the use of two different tools for the single applications in terms of understanding the interfacing or dealing with the different vendors for a single or unique application will make the work interesting and complex. This will help in to gain the knowledge about FPGA device architectures in order to get some specialization to create the most compact, high performance solution.

5. REFERENCES

[1] Michael Schaeferling, Markus Bihler, Matthias Pohl, Gundolf Kiefer 2015, ASTERICS - An Open Toolbox for Sophisticated FPGA-Based Image Processing.

[2] Mariangela Genovese and Ettore Napoli, 2014. ASIC and FPGA Implementation of the Gaussian Mixture Model Algorithm for Real-Time Segmentation of High Definition video. IEEE transactions on very large scale integration (VLSI) systems.

[3] Carlos Gonzalez et. al. 2013, Use of FPGA or GPU-based architectures for remotely sensed hyper spectral image processing.

[4] Horace Josh et. al. IEEE 2013, Psychophysics Testing of Bionic Vision Image Processing Algorithms Using An FPGA Hatpack.

[5] Carlos González et. al. IEEE 2012, FPGA Implementation of the N-FINDR Algorithm for Remotely Sensed Hyperspectral Image Analysis.

[6] G. Bradski, The Open CV Library, Dr. Dobb's J. Software Tools, 2000.

[7] J. Dı´az, E. Ros, F. Pelayo, E. Ortigosa, and S. Mota, IEEE 2006, FPGA-Based Real-Time Optical-Flow System.

[8] S. Sabatini, G. Gastaldi, F. Solari, K. Pauwels, M. Van Hulle, J. Dı´az, E. Ros, N. Pugeault, and N. Kru¨ ger,2010. A Compact Harmonic Code for Early Vision Based on Anisotropic Frequency Channels.

[9] A. Bruhn, J. Weickert, and C. Schnorr,2005 Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods.

[10] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnoerr,2006.A Multigrid Platform for Real-Time Motion Computation with Discontinuity-Preserving Variational Methods.

[11] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, 1992, Hierarchical Model-Based Motion Estimation.

[12] Hutchings, B. and Villasenor, J 1998, The Flexibility of Configurable Computing IEEE Signal Processing Magazine.