I.J. Image, Graphics and Signal Processing, 2016, 8, 37-46
Published Online August 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2016.08.06

Prototyping an Automated Video Surveillance System Using FPGAs

Sanjay Singh, Sumeet Saurav, Chandra Shekhar
CSIR - Central Electronics Engineering Research Institute (CSIR-CEERI) Pilani - 333031, Rajasthan, India
Email: sanjay.csirceeri@gmail.com

Anil Vohra
Electronic Science Department, Kurukshetra University,
Kurukshetra - 136119, Haryana, India.

***Abstract*—**Because of increasing terrorist activities, the resolution of video cameras and the number of cameras deployed for surveillance are increasing exponentially – producing huge amount of video data. Manual analysis of this large volume of video data by human operators for crime scene and forensic analysis is neither reliable nor scalable. This has generated enormous interest in research activities related to automation of video surveillance systems which allows real-time automatic extraction and analysis of information from live incoming video streams and enables automatic detection and tracking of targets without human intervention. To meet the real-time requirements of automated video surveillance systems, very different technologies and design methodologies have been used in literature. These range from use of General Purpose Processors (GPPs) or special purpose Digital Signal Processors (DSPs) or Graphics Processing Units (GPUs) to Application Specific Integrated Circuits (ASICs) or Applications Specific Instruction Set Processors (ASIPs) or even programmable logic devices like Field Programmable Gate Arrays (FPGAs). FPGAs provide real-time performance that is hard to achieve with GPPs/DSPs, limit the extensive design work, time, and cost required for ASICs, and allow algorithmic changes in later stages of system development. Due to these features FPGAs are being increasingly used for prototyping automated video surveillance system quickly. In this paper we present the top level description of a complete automated video surveillance system along with the elaboration of different challenges/issues involved in its design and implementation, a comparative analysis of design methodologies and existing FPGA platforms, complete design flow for prototyping the FPGA-based automated video surveillance system, and details of various primary input/output interfaces required for designing smart automated video surveillance systems for future.

***Index Terms*—**Automated Video Surveillance Systems, FPGA-based Smart Surveillance System, Smart Camera System.

I. Introduction

Recent terrorist activities in different countries of the world have resulted in serious security concerns. One way to cope with this problem is the widespread use of video surveillance technology at different public places and buildings of national importance which are usually the targets of terrorists. Video surveillance technology allows remotely monitoring of a given circumscribed area by using video cameras. Moreover, the use of video surveillance cameras can also assist police in handling many societal issues and crimes efficiently. Typically, the traditional approach used for video surveillance is based on conventional closed-circuit television (CCTV) camera. A basic conventional closed-circuit television (CCTV) based video surveillance system consists of a network of video cameras which has a coverage of the circumscribed area defined by the field of views of the cameras. These video cameras capture the appearance of a scene electronically and the video streams are transmitted to a central location to be displayed on one or several video monitors and analyzed by security personnel or stored for later observation or analysis [1]. The block diagram of a conventional CCTV based video surveillance system is shown in Fig. 1.

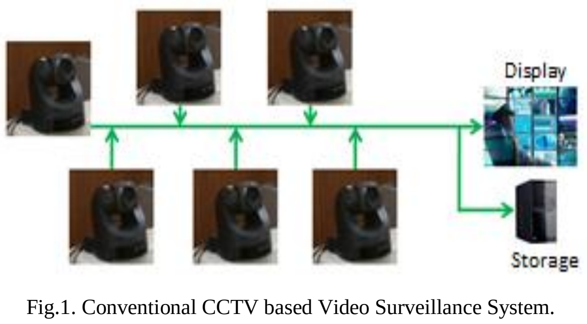

Fig.1. Conventional CCTV based Video Surveillance System.

The unusual events occur infrequently and their detection requires focused observation of multiple video screens by the security personnel for twenty four hours a day continuously in order to alert security officers if there is an emergency. In recent years, due to increasing security and safety concerns more and more cameras have been deployed producing huge volume of video data which results in their storage issue on one hand and on other hand it requires a large analysis time when the police or security services need to actually analyze it. Also, with this explosion in the number of cameras that are required to be monitored the accruing costs of monitoring personnel (one person can observe typically four cameras at a time with good accuracy of event detection [2]. It therefore requires expensive human resources for real-time video surveillance of large areas using the tradition technology) and the limitations of human operators to maintain sustained levels of concentrations severely circumscribe the efficaciousness of these systems [3]. These issues have raised serious concerns on the role of traditional CCTV based video surveillance systems in managing safety and security of organizations and public places. To overcome these concerns, there is a need for the development of automated video surveillance systems and researchers from all over the world are striving hard to develop such surveillance systems that are reliable, easy-to-deploy, easy-to-manage and that can perform real-time analysis of video scenes for generating automated real-time alerts along with effective retrieval of archived footage.

II. Automated Video Surveillance System Overview

Automated video surveillance is a rapidly evolving area and has been gaining importance in the research community in recent years due to its capabilities of performing more efficient and effective surveillance by employing intelligent video processing algorithms. The advantage of an automated video surveillance system over the conventional CCTV based system lies in the fact that it is a self-contained system capable of performing real-time analysis of a scene [4] and can intelligently decide on when to generate alert for abnormal activity, what to store for further processing (e.g. motion detection algorithm/unit allows more efficient hard disk storage by only archiving video frames where a motion threshold is passed), what to communicate, and what to track, etc. These systems not only potentially cut the cost of human resources observing the output of the camera but also eliminate the chances of human errors in analyzing multiple video streams and taking decisions based on analysis. Furthermore, the automated video surveillance system outputs the features extracted from the captured images or a high-level description of a scene, which is fed to an automated control system for decision making, whereas the conventional CCTV based video surveillance system outputs the complete captured images/video streams by the cameras. For this reason, the conventional CCTV based surveillance system has a large output bandwidth requirement (in direct proportion to the resolution of the video camera used for capturing), while automated video surveillance system has a very low data bandwidth requirement at the output (for example, it can be just one bit in the simplest case, with ‘1’ meaning “there is motion” and ‘0’ meaning “there is no motion”) [5].

Over the last few years many research programs have been funded worldwide for designing automated video surveillance systems. There are many surveillance products commercially available in the market (a complete list of manufacturers is available in [6]). Most of these are expensive and have been implemented as a separate software system to which several cameras are connected (e.g. latest S3 IBM surveillance system [7]). A similar trend can be seen in academia also where either large systems are implemented in software or isolated algorithms and/or their implementations in hardware are published.

The main objective of most of the automated video surveillance systems is to select targets based on motion detection in a video stream and track their movement. Therefore, most of the automated video surveillance systems have two major subsystems: a target (object/region of interest) detection system and a target tracking system. From the architectural point of view, they can be further decomposed into more stages like scene acquisition, region/object of interest detection, object tracking, camera feedback, decision making to generate alerts, and result display.

A conceptual overview of the functional decomposition of an automated video surveillance system is depicted in Fig. 2. Starting at the input, the camera feeds the system with a real-time video stream (a sequence of images) of a scene. After receiving the pixel data, the *Region of Interest Detection* module find the regions/objects of interest in video scene by running the intelligent object detection algorithms. The output of this block is used by *Object Tracking* module for tracking the objects which need to be tracked. The *Object Tracking* module extracts and updates certain useful parameters internally, such as co-ordinates of new location of tracked object in the current frame. These parameters are used by the *Camera Feedback* module and the *Object Tracking* module itself in the next frame. Based on the values of new co-ordinates of the tracked object in current frame, the *Camera Feedback* module generates the necessary commands for purposive camera movement (pan-tilt) in required directions so that it may follow the tracked object. *PTZ (Pan-Tilt-Zoom) Camera* is used to cover a large field of view and follow the tracked object over a larger area. The output of *Object Tracking* module together with inputs from *Region of Interest Detection* block can be used to generate the alarm signal for security personnel in case any relevant motion is detected in restricted areas. At any point of time, the output of any module can be viewed on the display monitor. The displayed results are not just images but objects at higher levels of abstraction (e.g. extracted regions of interest, tracked object, filtered frames of interest, etc.). At the heart of the system are different efficient hardware units for executing intelligent algorithms.

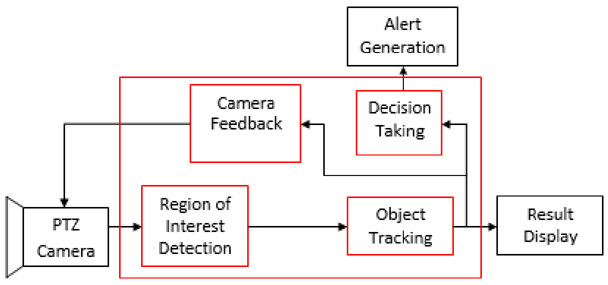

Fig.2. Conceptual Block Diagram of the Video Surveillance System.

III. Implementation Challenges

Developing a complete automated video surveillance system has many open unresolved design and research challenges such as real-time performance requirements, memory constraints, automatic camera PTZ features control, relevant motion detection, and so on. Addressing of these issues requires the designing of efficient hardware architectures and memory controllers, implementing of camera and display device interfaces for real-time video capturing and viewing, interface design for automatically controlling camera pan-tilt-zoom features in real-time, and so on. These challenges can be handled at different abstraction levels and by various techniques. A brief description of the major challenges has been discussed below.

A. Real-time Performance Constraints

Because of increasing video frames rates, resolutions, and functionalities of automated video surveillance systems, meeting of real-time performance requirements is most *crucial* aspect as it has to run many complex and computationally intensive computer vision algorithms in parallel. Within computer vision community, there exists a large number of algorithms for designing automated video surveillance systems, and unfortunately, most of these are not suitable for real-time processing either because of computational complexity or available resource limitations on development platform. Addressing of these issues requires the design of computational resource efficient algorithms and their hardware architectures to achieve real-time performance.

B. Memory Constraints

Most of the intelligent computer vision algorithms used for designing automated video surveillance systems require huge amounts of storage space for input vectors, intermediate variables, results, and key information from multiple video frames for taking decisions. To address this issue, careful consideration need to be given to the analysis of memory requirements and design of memory efficient algorithms, efficient memory interfaces, and memory-efficient hardware/VLSI architectures.

C. Automatic Camera Movement Control

The smart automated video surveillance systems require a camera with pan, tilt, and zooming (PTZ) capabilities to cover a wide field of view and to track the target objects over a larger area; and these PTZ features must be controlled automatically by system intelligence rather than manual interventions [8]. Therefore, design and implementation of real-time camera feedback/movement controller is another important issue which needs a special attention in order to satisfy the goal of designing a fully automated smart video surveillance system.

D. Significant Vs Insignificant Motion

Motion detection is one of the most important and primary component of any automated video surveillance system and outputs of motion detection affects the further operations or actions taken by automated video surveillance system for automatic analysis purpose. As in real-world situations the backgrounds are pseudo-stationary (.e.g. there may be moving leaves of trees, moving fan, moving clouds, etc. in the scene), therefore, while maintaining sensitivity, the automated video surveillance system must include algorithms to help distinguish the "uninteresting" motions from really "interesting" ones.

Addressing above mentioned challenges in toto requires drawing on the research from several domains and is truly a challenging and an interesting problem in Computer Vision and Integrated Systems communities.

IV. Design Space Exploration

Designing an efficient real-time automated video surveillance system for practical applications is really a complex task and requires the proper optimization of many parameters like processing speed or throughput, design time, cost, accuracy, and robustness. Due to expanding video resolutions, on one hand there is a need for high performance while at the same time on the other hand the architectures need to be flexible enough to allow for easy and quick upgradability. For these reasons, very different technologies and design methodologies have been proposed in the literature to exploit various characteristics of automated video surveillance systems. These range from the use of general purpose processors or special purpose digital signal processors to application specific integrated circuits or even programmable logic devices like field programmable gate arrays. In this section, we discuss the merits and demerits of these approaches.

A. General Purpose Processors (GPPs)

Despite all the improvements and special features, general purpose processors are still generic computational units, designed to execute any algorithm that can be converted into a computer program, and suffer from performance bottleneck which arises due to the sequential nature of program execution. The expanding video resolutions and frames rates of video surveillance cameras produce a huge volume of video data while at the same time the increasing functional requirements of automated video surveillance systems require many complex operations on video data. These issues make it very difficult to achieve the real-time analysis of a video scene with the help of general purpose/serial processor [9]. Also, general purpose processor based implementation results in increased power budget. Therefore, general purpose processors are used for designing solutions for limited budget and shorter design time applications where real-time performance and power dissipation are not major constraints.

B. Digital Signal Processors (DSPs)

For special purpose image/video processing, several alternatives to general purpose processors exist. One alternative is to use special purpose digital signal processors which have many similarities with a general purpose processor but contain additional features such as dedicated multiply-accumulate (MAC) hardware, multi-port on-chip/external memory interfaces, and special address generation units to improve the execution of signal/image processing algorithms. These features make the DSPs a good solution for executing certain low-level image processing algorithms extremely quickly. However, for real-time analysis of a video stream, this approach can be over specialized, since automatic scene analysis requires a large range of low, medium, and high level image/video processing operations [10].

C. VLSI Design Approach

Often, the automated video surveillance system is intended for real-time applications and standalone operation, which pose constraints related to performance requirements, area requirements, and power consumption. Therefore, it is not feasible to use programmable solution (General Purpose Processors or Digital Signal Processors) for designing such systems. An alternative method for designing such computational power hungry systems is to leverage Very Large Scale Integrated Circuit (VLSI) design approach to achieve real-time performance at low power. Depending upon the system specifications, VLSI Design approach may follow either of the two paths i.e. design of application specific integrated circuits or developing field programmable gate array based solutions. With the full freedom to customize the hardware, both ASICs and FPGAs can achieve much better system performance when compared to other technologies. The merits and demerits of each are discussed below.

D. Application Specific Integrated Circuits (ASICs)

Application specific integrated circuit design approach involves the design of an application specific hardware for specific scene analysis/computer vision algorithms, in order to perform a high rate of operations per second and satisfy the desired constraints. Many image/video processing algorithms allow efficient mapping onto an application specific architecture due to their inherent parallelism. Since the dedicated architectures are designed to compute specific tasks, therefore, datapaths, memories, and controllers in such architectures are optimized to meet the design goals. This leads to the highest performance and energy efficiency. However, the design and test of an application specific integrated circuit (ASIC) for specific computer vision applications can be quite strenuous, owing to many technology and financial constraints that often restrict the developer's pool of resources [10]. This is due to several factors such as the high non-recurring costs and efforts and long development time/cycle required for designing and testing these circuits. For example, when adding up costs for masks and wafers, software, design verification, and layout, development of a typical 90-nm standard ASIC can cost as much as US Dollar 30 million. Only a high volume consumer market can justify such pricey development costs [8]. Furthermore, application specific circuits do not admit the possibility of performing algorithmic changes in later stages of system development. ASICs can be cost effective only if the implementation is to be mass produced.

E. Field Programmable Gate Arrays (FPGAs)

Due to rapid advancements in fabrication technology (adoption of finer chip geometries down to 28 nm and higher levels of integrations), there has been stunning growth in the size, functionality, and performance of field programmable gate arrays (FPGAs) in recent years. The size and speed of current generation FPGAs are comparable to ASICs (though ASICs are typically faster, low power, and occupy less area as compared to FPGAs, given the same manufacturing technology). On the other hand, FPGAs limit the extensive design work required for ASICs, shorten the development cycle (results in reduced cost), and admit the possibility of performing algorithmic changes in later stages of system development as well. Furthermore, FPGA structure is also suitable for exploiting spatial and temporal parallelism inherent in computer vision, and pattern recognition applications. Even if one is following the ASIC development route, the functional verification on an FPGA before manufacturing an ASIC is always advised since many errors can be detected and corrected that way by running the FPGA emulation of the ASIC over extended periods and data sets.

F. Analysis of Implementation Choice

System implementation choices for designing an automated video surveillance system include GPPs, DSPs, ASICs, and FPGAs. Each of these approaches has its own advantages and disadvantages, with the ultimate choice depending upon the end system requirements and solution availability. For prototyping an automated video surveillance system intended for developmental purposes, with an easy upgradability and flexibility, rapid prototyping, and is not meant to be produced in volumes, given the trends discussed above, the clear choice would be via FPGAs.

V. FPGA PLATFORM SELECTION

There exists a variety of FPGA development platforms from different manufactures. It is very difficult to choose the right platform as there are/is no easy guidelines/process for this purpose, because the specifications vary widely. Based on the specifications and requirements of target system, one proceeds to assess as to which FPGA platform is suitable for the selected application/system. For designing the automated video surveillance system, the FPGA development platform should have at least the following essentials.

The first key issue to consider is the availability of some kind of input interface on the development board for getting video data into the FPGA platform and output interface for displaying the results/ processed video data/ frames of computer vision algorithms. The input may be analog video signal or VGA (Video Graphics Array) input source, or digital video signals (digitized using some daughter video cards) and output may be DVI (Digital Visual Interface) or VGA or HDMI.

The second key issue when considering an FPGA platform for designing automated video surveillance system, is the availability of computational resources (in terms of number of CLBs). Automated video surveillance systems have to run many complex algorithms in parallel for scene analysis, thereby they require a large number of computational resources on the FPGA.

The third key factor in determining whether an FPGA platform is best suited for designing automated video surveillance system is the availability of sufficient on-chip memory to buffer one or more video frames and support internal algorithmic/application storage requirements. In addition to the on-chip memory, there must be some kind of external memory and associated interface for storing multiple frames of video data as required by many automated video surveillance system algorithms.

For product level prototyping, an important consideration is the availability of a Compact Flash Memory and associated interface for storing the configuration file and supporting the automatic booting of the system at power on. The availability of embedded processor such as PowerPC makes the platform more suited for designing automated video surveillance systems having both hardware components as well as software components.

In addition to all the above main issues, there are other less critical but equally important considerations, such as which other interfaces are required by the automated video surveillance systems, which may also play an important role in selecting the suitable FPGA platform. For example, UART interface is needed for transferring the commands to the camera for purposive automatic camera movement (Pan/Tilt/Zoom) and Ethernet interface is required for connecting the automated video surveillance system on Local Area Network (LAN) or internet. In addition, the large number of high speed discrete I/O ports on an FPGA development platform can often be used to implement standard or custom interfaces.

Keeping the above requirements in mind, system designer needs to search through a wide variety of FPGA development boards for selecting appropriate development platform. Some examples of suitable available FPGA boards from Xilinx are Xilinx Virtex-IIPro XUP Board, Xilinx ML507 Board, Xilinx ML510 Board, and Xilinx Spartan6 Board. A comparison of the FPGA resources available on these four FPGA platforms [11]-[14] is given in Table 1.

The V2P (Virtex-IIPro) FPGA board and ML507 (Virtex-5 FX70T) FPGA board are available under Xilinx University Program (XUP) and therefore are low cost development boards. But the number of CLBs (Computational resources) on both these boards are much less as compared to ML510 (Virtex-5 FX130T) FPGA board and Spartan6 (Spartan-6 LX150T) FPGA board. Also the number of Block RAMs (On-chip memory) are much less as compared to ML510 (Virtex-5 FX130T) FPGA board. Spartan6 FPGA board has some more CLBs as compared to ML510 FPGA board but it has much less on-chip memory. Moreover, no on-chip embedded processor is available on Spartan6 board. ML510 FPGA board has the number of CLBs comparable to that of Spartan6 FPGA board and has on-chip memory that is more than double the size of memory available on Spartan6 FPGA board. In addition to this, ML510 FPGA board has two embedded processors, two RS232 ports, two Ethernet ports, 512 MB Flash Memory, and two DDR memory Interface ports (each of 512 MB). The only negative aspect of ML510 (Virtex-5 FX130T) FPGA board is the non-availability of video input/camera input interface port. But the presence of a large number of high speed input/output ports allows the interfacing of camera using custom designed daughter card.

Table 1. Comparison of Available Resources on Existing FPGA based Video Processing Platforms.

Resources	**Virtex-2Pro**	**ML507**	**ML510**	**Spartan6**
CLBs	13696	11200	20480	23038
BRAMs	2448 Kb	5328 Kb	10728Kb	4824 Kb
Processor	2 PowerPC	1 PowerPC	2 PowerPC	NA
Video IN	Video	VGA	Not Available	Video
Video OUT	VGA	DVI	DVI	DVI
RS232 Port	1	1	2	1
Ethernet Port	1	1	2	1
Compact Flash	32MB	32MB	512MB	32MB
DDR Interface	One 256 MB	One 256MB	Two (Each of 512 MB)	One 128MB

VI. FPGA-Based Automated Video Surveillance System Design Flow

The scope of automated video surveillance system design using FPGA design flow involves translating a given set of specifications of the system into configuration bit files that are used to program the FPGA device. This translation task is very complex in nature as it requires expertise from three different fields namely computer vision, system design, and VLSI design.

Therefore, it cannot be accomplished in one step. It is accomplished through a succession of translation steps of manageable complexity shown in Fig. 3. Each translation step translates a more abstract (less detailed) design representation into a less abstract (more detailed) design representation. When we move from one level to the next lower level in this design flow, there are many valid design solutions at that level, each with a different cost-performance-power implication, thus offering trade-offs to the designer. Therefore, the central issue in design process is managing increasing complexity and making optimum choices as one goes through the successive translation steps.

The first step, System Specification, is common to all engineering design problems and involves listing of different features or functionalities of the system. At this step, one need to clearly identify the different functionalities of the automated video surveillance system.

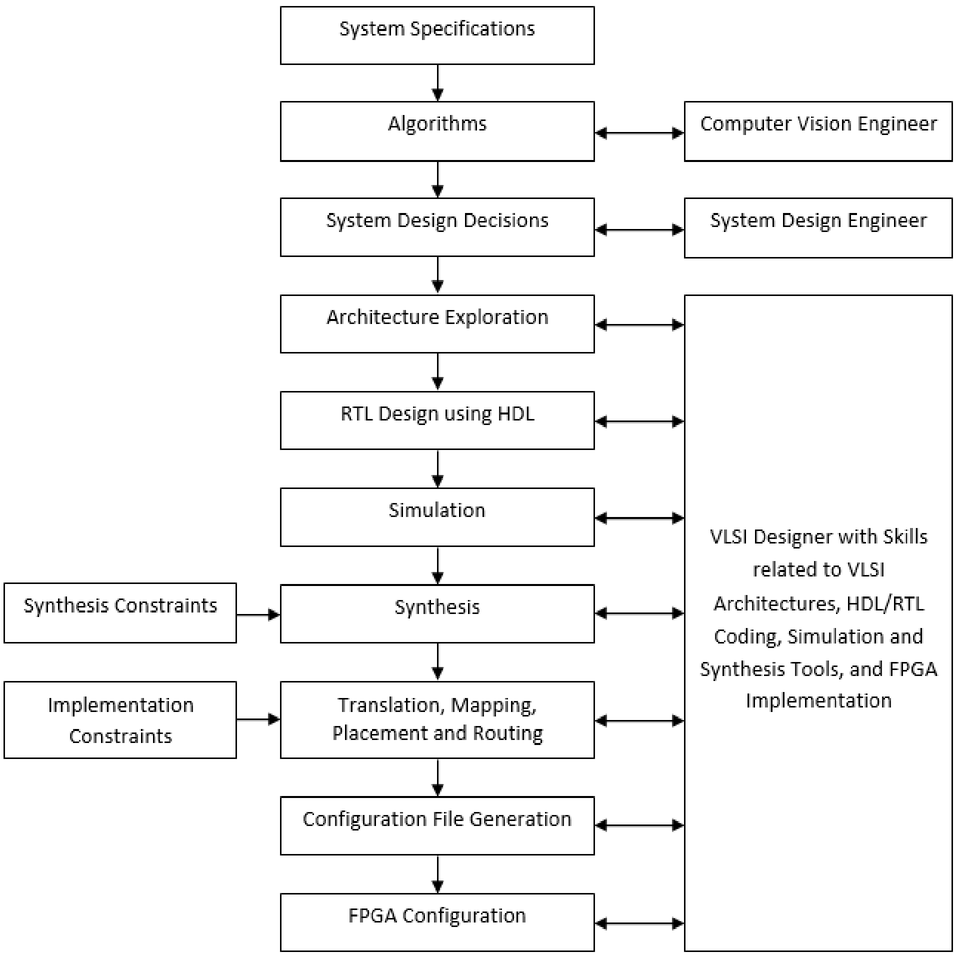
System Specifications
Algorithms
Computer Vision Engineer
System Design Decisions
System Design Engineer
Architecture Exploration
RTL Design using HDL
Simulation
VLSI Designer with Skills related to VLSI Architectures, HDL/RTL Coding, Simulation and Synthesis Tools, and FPGA Implementation
Synthesis Constraints
Synthesis
Implementation Constraints
Translation, Mapping, Placement and Routing
Configuration File Generation
FPGA Configuration

Fig.3. Sequence of steps to be followed during design of an FPGA-based Automated Video Surveillance System.

For FPGA-based automated video surveillance system design, the next step is Algorithms and it involves development of algorithms, which can be regarded as problem solving tasks, where the aim is to find a sequence of image processing/mathematical operations that progressively transforms the image into desired results, satisfying the functional requirements outlined in the system specifications. This step requires the expertise in computer vision field. At this stage, a proper analysis of all possible algorithms is desired and a selection has to be done based on parameters such as computational complexity and memory requirements in order to fully take advantage of the potential benefits offered by FPGAs. For algorithm development, one can use MATLAB or JAVA or C/C++ programming language with OpenCV (Open Computer Vision) libraries for reading/viewing images/videos.

Once the algorithm is finalized and implemented, the next step is to take System Design Decisions which effect system performance, cost, design time, and size. This requires knowledge of system implementation level issues and methods. First thing to consider at this level is the selection of FPGA development platform. The next thing to consider is system configuration i.e. standalone or hosted. For standalone configuration, all the tasks are performed by the implemented system, whereas in the case of hosted configuration, the implemented system works as a co-processor and complements the host computer for specific task. Further, in latter case, the video capturing, displaying, and camera movement control are performed by the host computer and not by the implemented system. For implementing a standalone system, FPGA board must have all input/output interfaces necessary for video capturing, displaying, and camera movement control.

The fourth stage is Architecture Exploration. This stage and all subsequent stages require the expertise in VLSI Design. One can conceive several architectural alternatives for a given algorithm including the possibilities of parallelism and pipelining. Architectural alternatives encompass pure combinational architectures, sequential architectures, fully pipelined architecture, parallel architectures, combinations of pipelined and parallel architectures, or programmable architectures. A rough estimation of throughput/speed, area, and power is required for selecting an optimal architecture best-suited for the given algorithm.

Once the architecture is finalized, the next stage is RTL Design using HDL. It involves designing of Register-Transfer Level architecture with enough details of individual block (processing elements, storage elements, and controllers). After designing the RT level architecture, the individual blocks of architecture need to be coded using Hardware Description Language (HDL). The different available HDLs are VHDL, Verilog, and SystemC. Functional verification of different modules coded in HDL is done through Simulation. The simulation softwares are available from different vendors. Once the design is functionally verified, the remaining steps (Synthesis, Translation, Mapping, Placement and Routing, Configuration File Generation, and FPGA Configuration) can be carried out by using different FPGA design tools. For this purpose, the Xilinx ISE (Integrated Software Environment) Project Navigator tool is available for Xilinx FPGAs. During Synthesis, the VHDL files of the design modules are converted to a gate level net-list representing the actual hardware that is to be constructed on the FPGA. During synthesis process, the synthesis tool also requires synthesis constraints to optimize the designs (e.g. whether to optimize the design for area or performance or power consumptions). The file generated during synthesis process is Xilinx Netlist Format (XNF) file. The next stage is implementation stage. This has three processes i.e. Translation, Mapping, and Placement and Routing. During Translation, the gate-level netlist in Xilinx Netlist Format (XNF) file is translated into Xilinx Native Generic Database (NGD) primitives describing the logical design in terms of logic elements such as Flip-flops, Multiplexers, Gates, etc. The Xilinx Native Generic Database (NGD) file generated by translation step, which contains a logical description of the design, is mapped to specific components (LUTs, Multipliers, DSP Slices, Block RAMS, etc.) available on target FPGA device during Mapping process. The output of the Mapping process is a Native Component Description (NCD) file which is the physical description of the design mapped onto the components available on FPGA device. The Placement and Routing process takes NCD file as input, places the mapped components with particular logic block on the FPGA device and determines the routing required to connect the logic blocks, memories, and I/Os. The implementation process also needs implementation constraints in a User Constraint File (UCF) specifying the association of inputs/outputs of design with particular FPGA pins and timing constraints. The placement and routing is done based on these constraints and a fully placed and routed NCD file is generated as output file.

The second last stage is to generate the configuration file required to program the FPGA device. The Configuration File Generation process takes the fully placed and routed NCD file as input and produces a configuration bit-stream (a binary file) with .bit extension. The .bit file contains all the device specific internal logic and interconnections information. The final step is to configure the FPGA. During FPGA Configuration, the .bit configuration file is used. The .bit file can be loaded onto the target FPGA development board using JTAG/USB cable to verify that the design works as intended and correctly interacts with other components within the system. For implementing an FPGA-based automated video surveillance system for its use as a standalone system, the configuration files need to be stored in Compact Flash memory and the FPGA is configured by automatically loading the configuration file from Compact Flash memory at power-on.

VII. Design of Different Interface Controllers

Another very important aspect of designing the FPGA-based automated video surveillance system is designing of different interface controllers for getting pixel data from incoming video stream, displaying processed video data on display monitor, controlling the camera pan-tilt movement, and writing and reading pixel data in/from DDR external memory. Design and implementation of these different interface controllers on FPGA require dedicated design efforts. These interfaces are detailed in this section.

A. Input Camera Interface

The main objective of the input camera interface controller is to extract pixel data and timing information from the incoming video signals coming from video daughter card. The pixel data is in form of raw RGB data or YCrCb data. The video timing information contains different video timing signals like Horizontal Sync (Hsync), Vertical Sync (Vsync), Blanking (Blank), Horizontal Blanking (Hblank), Vertical Blanking (Vblank), and Pixel Clock (PClk). These signals are used by different modules running on FPGA platform for synchronization of different tasks with incoming video data. The implementation details of this interface for Xilinx ML510 FPGA board are available in [15].

B. Output Display Interface

Another important interface is output video display interface. It can be DVI (Digital Visual Interface), VGA (Video Graphics Array), or HDMI (High-Definition Multimedia Interface). The output display interface takes the processed pixel data from processing block and the timing information signals from input camera interface and generates the output signals necessary for displaying the video through available DVI/VGA/HDMI port on FPGA board.

C. Camera Movement Interface

For designing the complete automated video surveillance system, the pan-tilt-zoom features of the cameras must be controlled automatically based on the outputs from logic functions running on FPGA. For this purpose, a dedicated Camera Movement interface needs to be designed which takes inputs from the processing module running on FPGA and timing signals from the video timing generation block and generates the necessary commands for purposive movement of the PTZ camera based on the inputs received, and sends these commands to the PTZ Camera through the UART port or any other port available on FPGA board. The command signals are generated based on the activities/movement of the target objects present in the video scene. Designing of this interface is a really complex and interesting problem as it requires real-time control of camera movement speed and direction depending on the movement of the target object to be tracked.

D. DDR2 External Memory Interface

The algorithms used for designing automated video surveillance systems require storage space for input data, intermediate variables, intermediate video frames, and results. On FPGA board there are mainly two types of memories available: on-chip memory and off-chip external memory. Register files and Block RAMs are available on FPGA device as on-chip memories and these are used for storing frequently accessed information and intermediate results for calculations. These are high-speed memories and can sustain high bandwidth requirements. The amount of on-chip memory (Block RAMs) available of FPGA boards is limited. Due to this limited amount of on-chip memory (e.g. 10728 KB on Xilinx ML510 FPGA board), external memory is required for storing the large amounts of data necessary for the automated video surveillance system algorithms. Double data rate (DDR) memory is available as external memory on almost all FPGA boards. Its storage capacity vary from 128MB to 1GB or even more.

With a high-data bandwidth and complicated timing parameters of DDR memory, the design of a DDR2 memory interface for storing image and video data is a challenging task. Clock signal, together with data and command signals, are transferred between external DDR memory and processing modules running on FPGA, therefore, it is very important to ensure that all data and command signals are valid at right timings.

For accessing the external DDR memory requires proper interface controller because the clock frequency for input data arrival from camera interface module or processed data arrival from processing module is different from the clock frequency at which the data is sent to the external DDR memory module for writing. Similarly, the clock frequency at which the data is available from external DDR memory module is different from the clock frequency at which the data is sent to the processing module or display controller. This can be achieved by designing interface FIFOs for reading and writing data from and to external DDR memory and associated controller.

E. Integration of Different Interfaces

The last but most important step is the integration of all the four interfaces discussed above and to achieve the goals of capturing real-time video streams, displaying results, controlling camera pan-tilt-zoom features, and accessing external DDR memory for image/video pixel data storage. The block level integration is shown in Fig. 4. The four interfaces are shown by light gray backgrounds. The modules inside red line form the Processing Module which features all intelligent video analysis and decision making.

Camera Interface module receives the digitized real-time video data from Physical Camera Interface and extracts the pixel data and timing information. It sends the pixel data to Processing Module or/and stores the pixel data into External DDR Memory through DDR Memory Interface depending upon the application/algorithmic requirements. The Processing Module performs the desired set of operations on the data based on algorithm/application and stores the results into External DDR Memory through DDR Memory Interface depending upon the application/algorithmic requirements or/and sends the processed data to Display Interface for display. Display Interface receives the processed data from Processing Module and displays the results on Display Monitor through display port available on FPGA board. Camera Movement Controller controls the pan-tilt of camera through UART port based on the inputs received from Processing Module. The Camera Interface also provides the timing signals to all modules running on FPGA device for proper functioning and synchronization of data in individual blocks and in overall system

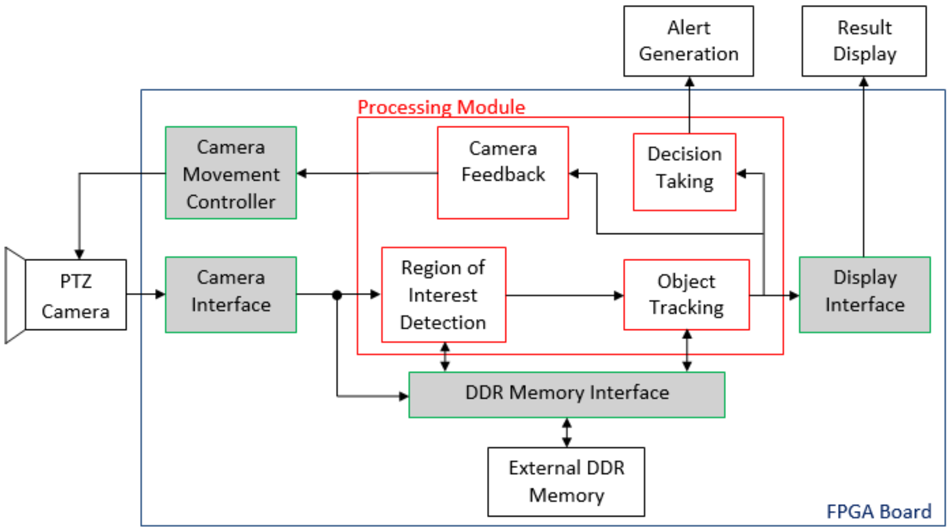

Fig.4. Dataflow Level Block Diagram of Different Interfaces Integration.

VIII. Conclusions

This paper has presented the top level description of a complete automated video surveillance system along with the elaboration of different issues involved in its design and implementation. A comparative analysis of design methodologies and platforms for implementing an automated video surveillance system has been carried out and it has been found that FPGA-based hardware implementation approach is best suited for prototyping these systems quickly. The complete design flow for prototyping the FPGA-based automated video surveillance system is presented and explained step-by-step. The details of various input/output interfaces, which are primary parts of an automated video surveillance system and require dedicated design efforts, have also been presented together with discussion on their integration. In our opinion, the discussions in this article can provide a valuable insight to those who wants to quickly prototype the computer vision algorithms on FPGAs.

References

[1] A. Senior, an Introduction to Automatic Video Surveillance. In Protecting Privacy in Video Surveillance; Senior, A; Springer London, pp. 1-9, 2009.

[2] E.J. Delp, Smart Video Surveillance: Mission, Solution, and Impact, PURVAC. (https://engineering.purdue.edu/PURVAC/pdf_fliers/PURVAC10_VideoTracking.pdf)

[3] Y. Nam, S. Rho, and J.H. Park, Intelligent Video Surveillance System: 3-tier Context-aware Surveillance System with Metadata, Multimedia Tools and Applications, vol. 57, no. 2, pp. 315-334, 2012.

[4] H. Jiang, H. Ardo, and V. Owall, A Hardware Architecture for Real-time Video Segmentation Utilizing Memory Reduction Techniques, IEEE Transactions on Circuits and Systems for Video Technology, vol. 19, no. 2, pp. 226-236, 2009.

[5] Y. Shi and S. Lichman, Smart Cameras: A Review, IMAGEN-National Information and Communications Technology Australia (NICTA) Australian Technology Park, Bay 15 Locomotive Workshop, Eveleigh, NSW 1430, Australia.

[6] Video Surveillance Homepage. (http://www.videosurveillance.com/manufacturers/)

[7] IBM Official Homepage, 2007. (http://researcher.watson.ibm.com/researcher/view_group.php?id=1394)

[8] Altera White Paper, Video and Image Processing Design Using FPGAs. (http://www.altera.com/literature/wp/wp-video0306.pdf)

[9] C.T. Johnston, K.T. Gribbon, and D.G. Bailey, Implementing Image Processing Algorithms on FPGAs, In Proceedings: Eleventh Electronics New Zealand Conference, pp. 118-123, 2004.

[10] H. Meng, N.E. Pears, and C. Bailey, FPGA based Video Processing System for Ubiquitous Applications, In Proceedings: Conference on Perspectives in Pervasive Computing, pp. 57-63, 2005.

[11] Xilinx Official Homepage for Virtex-IIPro Development Board. (http://www.xilinx.com/products/boards-and-kits/XUPV2P.htm)

[12] Xilinx Official Homepage for ML507 Development Board. (http://www.xilinx.com/products/boards-and-kits/HW-V5-ML507-UNI-G.htm)

[13] Xilinx Official Homepage for ML510 Development Board. (http://www.xilinx.com/products/boards-and-kits/HW-V5-ML510-G.htm)

[14] Xilinx Official Homepage for Spartan6 Industrial Video Processing Board. (http://www.xilinx.com/products/boards-and-kits/AES-S6IVK-LX150T-G.htm)

[15] S. Singh, A.K. Saini, R. Saini, Interfacing the Analog Camera with FPGA Board for Real-time Video Acquisition, International Journal of Image, Graphics and Signal Processing, Vol. 6, No. 4, pp. 32-38, 2014.

Authors' Profiles

Sanjay Singh is working as Scientist in CSIR-Central Electronics Engineering Research Institute, Pilani, India - A Government of India Research Laboratory. He is Member of IEEE - USA, IACSIT - Singapore, IAENG - Hong Kong, and SSI - India. He is involved in various projects sponsored by Indian Government on Computer Vision and VLSI Design. His research interests are VLSI Architectures for Computer Vision, FPGA Prototyping, and VLSI Design. He received his Ph.D. in VLSI Design for Computer Vision, M.Tech. in Microelectronics & VLSI Design, M.Sc. in Electronics, and B.Sc. in Electronics & Computer Science from Kurukshetra University, Kurukshetra, Haryana, India. He earned Gold Medal (First Position in University) during his M.Tech. and M.Sc. He topped college during B.Sc. He received more than 20 Merit Certificates and Scholarships during his academic career.

Sumeet Saurav is working as Scientist in CSIR-Central Electronics Engineering Research Institute, Pilani, India - A Government of India Research Laboratory. His research interests are Computer Vision and VLSI Design. He received his M.Tech. in Advanced Semiconductor Devices from Academy of Scientific & Innovative Research (AcSIR), India. He is University rank holder in B.E.

Chandra Shekhar is the former director of CSIR-Central Electronics Engineering Research Institute, Pilani, India - A Government of India Research Laboratory. His research interests are VLSI architectures and VLSI Design Methodologies. He received his Ph.D. from BITS, Pilani, India.

Anil Vohra is professor with Electronic Science Department, Kurukshetra University, Kurukshetra, India. At present, he is holding the positions of Dean Academics and Dean R&D at Kurukshetra University, Kurukshetra, India. He received his M.Sc. and Ph.D. from Punjab University, India.

How to cite this paper: Sanjay Singh, Sumeet Saurav, Chandra Shekhar, Anil Vohra,"Prototyping an Automated Video Surveillance System Using FPGAs", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.8, No.8, pp.37-46, 2016.DOI: 10.5815/ijigsp.2016.08.06