

High-Definition Surveillance Systems Using Low Cost FPGAs

Suhel Dhanani, Senior Manager – DSP, Altera Corporation and Mankit Lo, CEO, EyeLytics

Surveillance is the latest video market segment to move towards high-definition (HD) and demand correspondingly high performance video processing capabilities. HD surveillance cameras need to directly encode the image with a high-quality H.264 Encoder, so the image can be realistically transmitted over a standard Ethernet connection.

A main profile H.264 Encoder that can encode an HD video stream in real time requires a silicon platform that combines a high-performance signal processing fabric with low-cost and also low-power—making the latest generation low-cost FPGAs an ideal choice for implementation.

This article describes the architecture of a HD surveillance camera and shows how the entire system is built using a low-cost FPGA.

The HD Surveillance Camera Architecture

The newer high-definition IP cameras are end-points in an all IP-networked digital system. These IP cameras capture HD video, pre-process, encode, and send out the encoded stream via Ethernet. All the signal processing has to be implemented in a single device to meet the stringent cost and power requirements of such a system.

Figure 1 shows a top-level architecture of such a camera. Note that in addition to the encoding engine, a cost-effective design also integrates the camera sensor pre-processing, a frame buffer memory controller, an embedded processor for system control, and an Ethernet MAC. The objective is to achieve maximum integration and the lowest cost/power metric for the entire system.

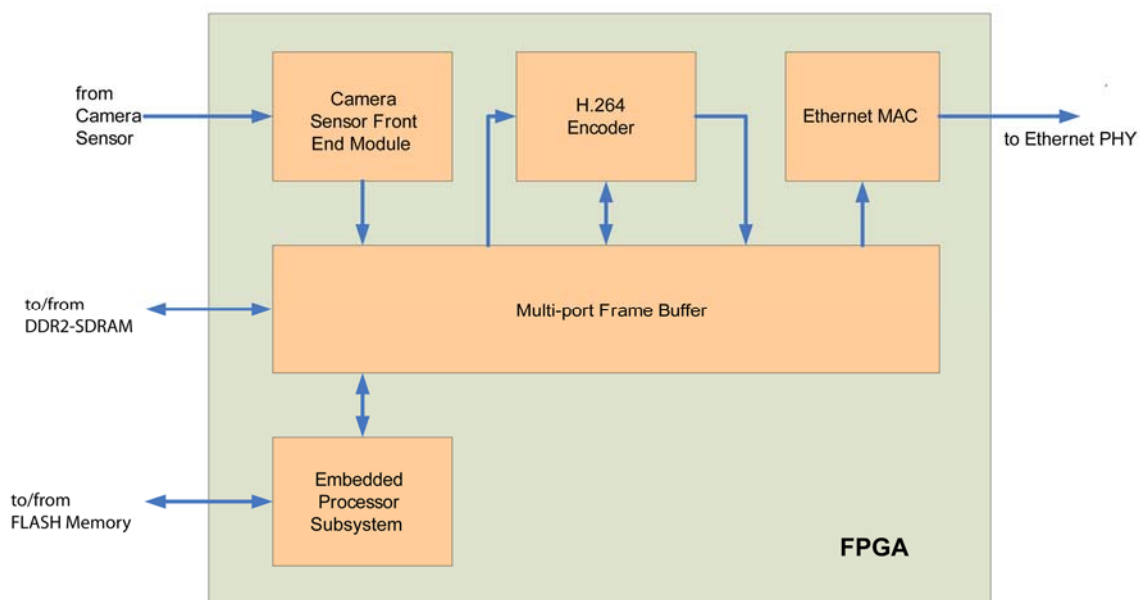


Figure 1. Top Level Architecture of an HD Surveillance System

This design includes a camera sensor front end module, a video compression module, an Ethernet MAC module, an embedded processor, and a multi-port frame buffer that provides memory storage to all other modules.

The multi-port frame buffer serves as a hub. All other modules send and receive data to and from the frame buffer that communicates with other modules. Video images received by the camera sensor are sent to the camera sensor front end module. The camera sensor front end module

processes the video data and stores the video to the frame buffer. Next, the H.264 Encoder reads that video data from the frame buffer and performs the encoding process. The H.264 Encoder then stores the compressed bit stream back to the frame buffer. At the last step, the Ethernet MAC module reads the compressed bit stream from the frame buffer and sends it to the Ethernet.

With such integration, the only other components on the board are a camera sensor, DDR2-SDRAMs, Flash memory, and an Ethernet PHY chip. Figure 2 shows the different components in such a system.

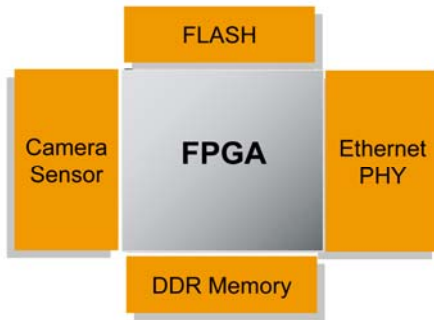


Figure 2. Different Components of an HD Surveillance Camera

Design Details: H.264 Encoder Module

The H.264 Encoder used in this design is an IP core available from [EyeLytics](http://www.eyelytics.com) that has optimized this core for surveillance applications. This core contains many surveillance features including multi-channel support, constant quality rate control, intra/inters modes, QPEL, CABAC, and a low gate count.

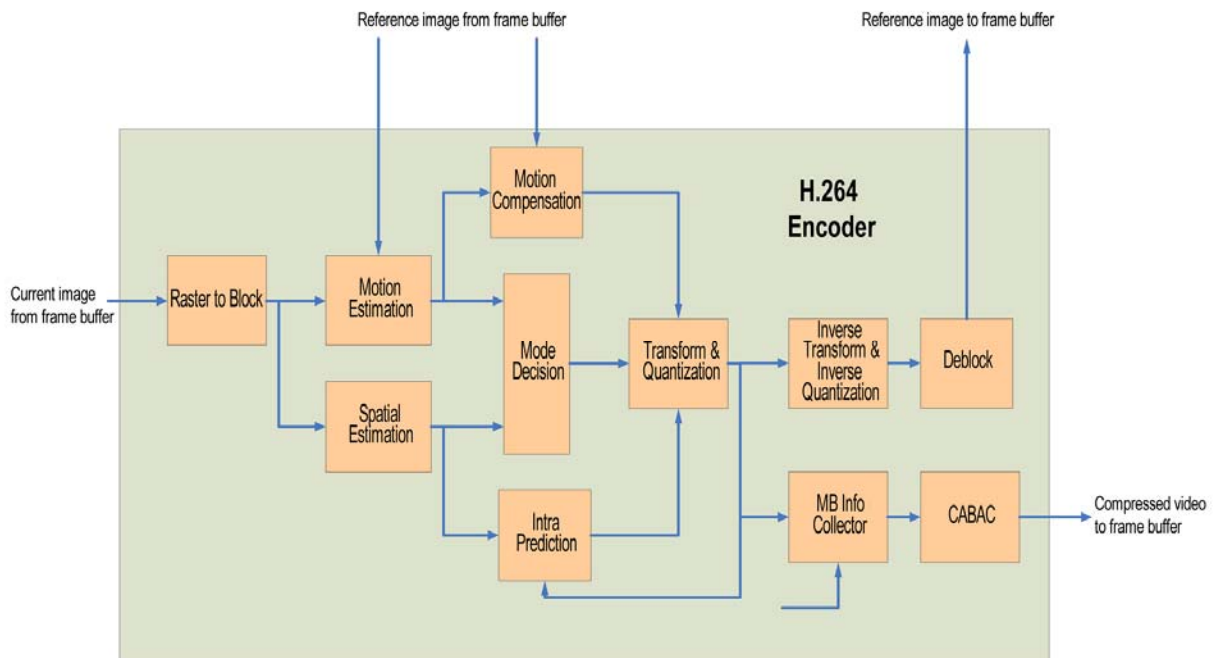


Figure 3. The H.264 Encoder Architecture

The 'Raster to Block' module (shown at the left of Figure 3) reads images from the frame buffer in raster scan order and rearranges it in macroblock format. The macroblock is then sent to the

Motion Estimation Engine and the Spatial Estimation Engine. The Motion Estimation Engine reads a reference image from the frame buffer and finds the motion vector of the current macroblock by searching the reference image. The Motion Estimation Engine also determines the best partition used for each macroblock.

The H.264 specification allows four different inter-prediction macroblock partitions and four different sub-macroblock partitions. The best motion vector and the best partition together with the corresponding estimated macroblock coding cost are sent to the Mode Decision Module. The best motion vector and the best partition information are also sent to the Motion Compensation Engine. The Motion Compensation Engine fetches the corresponding reference region, performs half pel and quarter pel filtering and generates the inter prediction for the current macroblock.

The Spatial Estimation Engine finds the best estimation of the current macroblock by using neighboring pixel values within the same image. There are a total of nine intra 4x4 luma modes, four intra 16x16 luma modes and four intra chroma modes defined in the H.264 specification. The Spatial Estimation Engine determines the best luma and chroma modes to be used for the current macroblock. The best modes together with the corresponding estimated macroblock coding cost are sent to the Mode Decision Module. The Mode Decision Module compares the macroblock coding costs and determines the best prediction to be used. This prediction can either be inter or intra prediction.

The best intra mode is sent to the Intra Prediction Engine. The Intra Prediction Engine uses neighboring pixel values to generate the intra prediction for the current macroblock.

Based on the Mode Decision result, the Transform and Quantization Engine subtracts the prediction values from the current macroblock. Then, it performs transformation and quantization to generate the quantized coefficients. The quantized coefficients and other macroblock information such as motion vector, inter prediction mode, and intra prediction mode is collected by the Macroblock Info Collector.

The collected information is sent to the CABAC Module to generate the final bit stream. This is done by using context-based adaptive binary arithmetic coding. The resulting bit stream is then sent to the frame buffer.

The reference images used by the Motion Estimation Engine and the Motion Compensation Engine are generated by three steps: inverse quantization, inverse transformation, and deblock. The Deblock Module reduces blocky artifacts from the image. The processed images are then sent to Frame Buffer and are used as reference images in the next frame period.

Design Details: Camera Sensor Front-End Module

The camera sensor used in this design is a MT9P031, a 5MP sensor from Aptina. This camera sensor is programmed to send 1920x1080 video images to the FPGA at 30 frames per second. The video data format used is 12-bit RGB Bayer pattern. The data transfer clock is set to 100MHz.

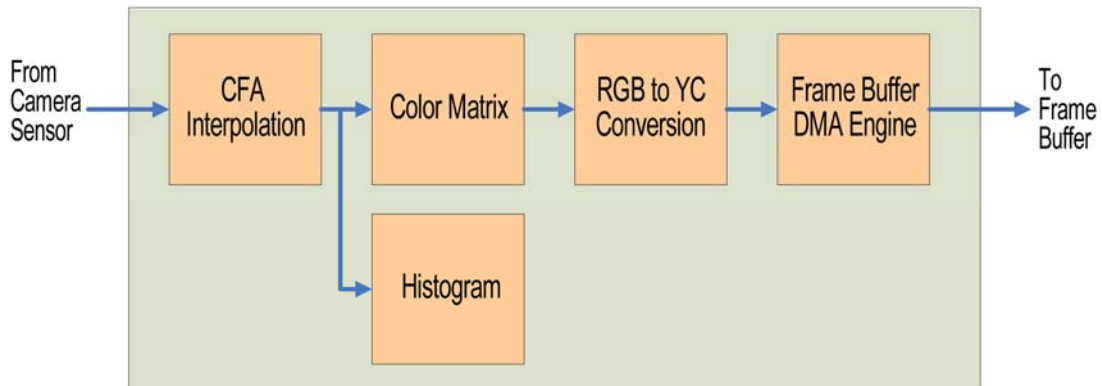


Figure 4. The Camera Sensor Front-End Block Diagram

Figure 4 shows CFA Interpolation used to convert the Bayer RGB pattern to a regular RGB image. The color matrix is used to adjust RGB color to make it match with the real world. RGB to YC conversion is done next. It generates YC 4:2:2 video data that is then saved to the Frame Buffer using the DMA Engine.

Other Design Details

An embedded processor is used for programming the various registers within the different modules as well as to run the TCP/IP stack for streaming the compressed video. Working with the Ethernet MAC module, the embedded processor runs the lwIP (a lightweight implementation of the TCP/IP stack), a streaming application, and a web server application.

The multi-port Frame Buffer connects to two DDR2-SDRAM chips with a 32-bit data bus and runs at 150MHz. This gives a maximum of 1.2GB per second memory bandwidth.

Putting it All Together

Figure 5 shows the entire FPGA-based system.



Figure 5. A Complete HD Surveillance System based on an Altera Cyclone III Development Platform

The system is composed of three boards:

- An Altera Cyclone III FPGA Development Board – with a 3C120 device

- An Aptina 5MPixel Sensor Module
- An EyeLytics Adapter that connects the Aptina sensor with the Cyclone III FPGA development platform

A demo design showing the camera sensor images being encoded and streamed through the network to a PC is available.

The entire design takes about 50,000 logic elements (LEs) with the Encoder Module using approximately 30,000 LEs when implementing a main profile H.264 encoding for a 720p/30 video stream.

The FPGA Advantage

Surveillance systems are not alike; this is especially true with HD surveillance systems. Systems designed for homeland security will be different than the ones used in casinos, even though they both employ HD camera sensors.

An FPGA-based architecture is completely flexible and customizable, while at the same time offering the peace of mind of being implemented on a standard, well-understood silicon platform. The system architecture described in this article can be enhanced and modified to suit different system requirements. Increasing the camera resolution or adding custom video processing or video analytics engine is easy since the design is implemented with standard HDL.

The H.264 Encoder Module supports multi-channel encoding, and streams with different resolutions enabling frame rate videos to be encoded at the same time.

Also since the design can be targeted to any FPGA, adopting an open design allows you to target the newest FPGA, thus receiving the higher performance and lower cost/power ratios as newer FPGAs are introduced.

About the Authors



Mankit Lo, CEO / CTO, EyeLytics Inc.

Prior to founding EyeLytics, Mankit was the engineering head of the Multimedia, Video and Image team at Xilinx Inc. Mankit was also the Director of Hardware Engineering at Pinnacle systems responsible for developing video systems. Prior to working at Pinnacle, he was employed by Adaptec, ShoGraphics, ULSI, Cray Research and NCR. Mankit has a broad knowledge of video, audio, 3D graphics, mathematics, and computer architecture. Mankit Lo holds a BSEE from San Jose State University and a MSCE from Santa Clara University.



Suhel Dhanani, senior manager, DSP, Altera Corp.

As senior manager in the software, embedded and DSP marketing group, Mr. Dhanani is responsible for all product marketing aspects of Altera's DSP solutions. He has over 15 years of industry experience in semiconductors – with both large companies such as Xilinx and VLSI Technology as well as with Silicon Valley startups such as Anadigm and Tabula. Mr. Dhanani has completed a graduate certificate in Management Science from Stanford University and also holds M.S.E.E. and M.B.A. degrees from Arizona State University.