

PrivacyCam: a Privacy Preserving Camera Using uCLinux on the Blackfin DSP

Ankur Chattopadhyay and T.E. Boulton
Vision and Security Technology (VAST) Lab
University of Colorado at Colorado Springs,
Colorado Springs, CO 80933

[achattop | tboulton] at vast.uccs.edu

Abstract

Considerable research work has been done in the area of surveillance and biometrics, where the goals have always been high performance, robustness in security and cost optimization. With the emergence of more intelligent and complex video surveillance mechanisms, the issue of “privacy invasion” has been looming large. Very little investment or effort has gone into looking after this issue in an efficient and cost-effective way. The process of PICO (Privacy through Invertible Cryptographic Obscuration) is a way of using cryptographic techniques and combining them with image processing and video surveillance to provide a practical solution to the critical issue of “privacy invasion”.

This paper presents the idea and example of a real-time embedded application of the PICO technique, using uCLinux on the tiny Blackfin DSP architecture, along with a small Omnivision camera. It demonstrates how the practical problem of “privacy invasion” can be successfully addressed through DSP hardware in terms of smallness in size and cost optimization. After review of previous applications of “privacy protection”, and system components, we discuss the “embedded jpeg-space” detection of regions of interest and the real time application of encryption techniques to improve privacy while allowing general surveillance to continue. The resulting approach permits full access (violation of privacy) only by access to the private-key to recover the decryption key, thereby striking a fine trade-off among privacy, security, cost and space.

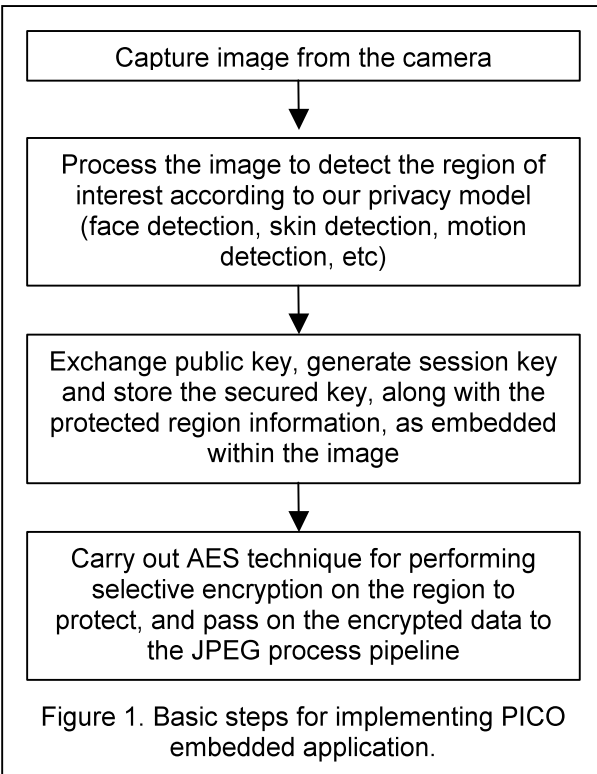
1. Introduction: privacy enhancing research and video surveillance

Before going into the discussion of the real-time embedded application of PICO, we briefly discuss the background of the issue of “privacy invasion”, in this section. The gravity of the situation of “privacy invasion” is to be understood in the light of the interpretation of today’s high standard security measures. The questions to be asked are – what exactly is essential for maintaining

security standards and what exactly can be compromised for a desirable amount of individual privacy preservation? While the requirement of an individual’s private data (like personal records and credentials) can be accepted for basic security norms, the capture and usage of an individual’s video/still images, as done in vision and biometrics, is a matter of concern, as far as privacy is involved. Moreover, there are legal issues of privacy in restricted places, as compared to public places. Together, all these factors make the issue of “privacy” a delicate and critical one, requiring some serious thought and supervision from the vision and surveillance system designers. So, while collecting data and imagery of individuals by surveillance cameras, balancing security levels and privacy becomes the fundamental responsibility of the vision system designers.

Now, talking about the common ways of video surveillance, research and statistics have shown that the employment of CCTV has had an overall deterrent effect on the crime scene by displacing crime and moving it out of the camera boundaries, as discussed in reference [7]. While the use of surveillance systems in public places puts the privacy of individuals at stake, the use of surveillance at restricted private places (like private households, restrooms) raises legal issues. This reveals the inadequacy and limitations of current surveillance systems, and questions the ability of surveillance system designers to apply technology at the right places in the right way, preserving privacy aspects and uplifting security standards at the same time, for the greater cause.

Much of the review on previous research, involving privacy enhancement, has shown that past works have lacked the right combination of effectiveness, robustness and cost optimization. While some technologies have been based on improving privacy by capturing image/video data at lesser details, as discussed in reference [8], other research has been unable to meet the minimum needs of privacy preservation. That’s where the PICO technique comes to the rescue (as discussed in reference [1]) – by involving a two-stage process, where we first encrypt the selective areas of a video image (for preserving privacy) and in case of legal authorization, we get back the full



original image by decryption (thereby meeting security standards). Unlike earlier work, such as the one discussed in reference [8], the private information is not permanently destroyed. A commercial implementation of a privacy-enhanced surveillance is now available from Emmitall, but a software implementation presumes that we trust the computer software not to make copies of imagery.

We believe it is critical for privacy protection to be implemented in the camera, so that those being observed can be sure no user can violate the expected operations without the keys. We still face the challenge of being able to implement it in the best possible way – which has the attributes of being efficient, robust, cost-effective and space-optimized. In this paper, we try to meet those challenges and show that PICO, when implemented as an embedded DSP application, in the Blackfin architecture, meets all the ideal requirements of surveillance – robustness, efficiency, privacy protection and affordability. We will discuss and demonstrate our idea in details, in the following sections.

2. Moving into hardware: notion of our real-time DSP application

Firstly, implementing PICO in hardware, must address detection of the regions to protect and then protecting it. But to be effective, it needs to require low cost, low space and system stability in a DSP hardware environment. A DSP system would typically be free from some of the typical vulnerabilities of software implementation, where the owner of the machine, operating the web cam, will have

access to operating system of the machine, and, thereby, can control the operation of the camera. In order to prevent the user from making easy changes to the settings of the camera, we adopt a DSP based embedded platform for our webcam (resulting from the integration of Blackfin DSP module with a small Omnivision camera). And the other driving factor is, of course, the low cost and small space requirements of Blackfin DSP hardware.

A few of the output results from our research, involving the application of selective encryption to DCT based JPEG compression, are shown in Figure 2. Here, we use selective encryption of the human face region. We next discuss the background subtraction model we use for detection of our region of interest (ROI), which is the human face in this case.

2.1. Our ROI detection technique

The technique, we use for detecting the ROI in a given surveillance image, is based on a background subtraction model. The process we follow is:

- Firstly we take two separate image frames, background model and a captured one, and consider the typical 8×8 DCT blocks that belong to each..
- We then perform a per coefficient difference computation for the blocks, considering two blocks - one block from each image, at a time.
- Then we compare the obtained difference with a model threshold value. If the majority of the coefficient differences are greater than the threshold, we encode that block, as our ROI; otherwise we don't encode that block, and for each pixel, we just put a new pixel value calculated as the average of the two corresponding pixel values.

This jpeg-based background subtraction method turns out to be effective, and can track significant changing object, which becomes our area of interest in the image. It, therefore, is not limited to only a technique for face detection. As shown in the images of Figure2, the performance of our method of ROI detection, in case of human face, is different from that of a typical face detector, which captures only the face. Our method captures not only the face but also the entire head and portions of the top of the head and is less susceptible to missed detection. This application of our method is shown in Figure 2. If only limited encryption of high-probability face regions is desired, we encode only those blocks at the top portion of the ROI. This of course could allow some privacy invasions, e.g. a person fell, so it's a mixed strategy to be considered carefully. But for this paper we use it as the example. For enhanced privacy, one can just encode all motion areas.

Incidentally, the pictures presented in Figure 2 have been taken from our Omnivision CMOS camera, using our embedded Blackfin application. The environment/platform issue of running our embedded DSP application will be discussed later in section 5.

For our privacy-enhancing research, we apply our process of ROI detection, with a fine-tuning, in the form of giving more weight to the pixel difference values in the middle frequency range. This means that if we classify our image into low, mid and high frequency regions, we apply a weighing to coefficient difference values in the mid-frequency region, which improves resistance to both low-frequency lighting variations, and also the high-frequency noise that can occur low light settings. Doing the background modeling in jpeg/frequency space simplifies these “frequency” related filtering. This improves the chances of detecting and encoding whose privacy protection is important.

As seen in the pictures (in Figure 2), captured through our embedded DSP application, running on our research camera, two scenarios have been given as examples. First, we take some images captured by the webcam, while monitoring the kitchen of a household. The motion happening in the kitchen has been covered in three different shots taken at three different instants – the empty kitchen; an instant after the person enters the kitchen and another example as the person starts working in the kitchen. As seen, our embedded camera ignores the first empty frame (where there’s nobody) and starts ROI detection once it tracks the person (as a changing object) after he enters the kitchen. The embedded camera application does ROI detection (capturing the top level blocks containing the face) and encryption in the second and third frames, as the person moves within the kitchen (which is evident from the pictures). Similarly, in the second scenario, while monitoring a household room, the camera captures an instant from what’s happened in the room. As we see, in the raw picture frame, there’s a man standing, and a lady just entering the frame. Our camera applies ROI detection on the man as well as the lady, in spite of her slightly visible presence in the first frame. So, the camera continues its operation of the face region hiding, thereby protecting privacy, irrespective of the fact that there’s a greater prominence/presence/visibility of an individual.

The images in Figure 2 show the behavior of the embedded camera in different situations, involving no human face, and involving one or more human faces at extreme angles and partial views which would befuddle most face detectors.. Whether there be motion/relative displacement or not, the camera functions successfully from one distinct frame to another, doing its job of ROI detection and performing privacy preservation in that region.

3. Introduction to Blackfin DSP processors

DSP processors are, in general, I/O (Input/Output) balanced processors offering a variety of high speeds modes/instructions. They are ideally designed in a way that they can be operated with very low or no overhead impact to the processor core, leaving enough CPU time for



Figure 2. Some encrypted and unencrypted output images listed from our research work. Each frame with a changing object is followed by an encrypted version using our ROI detection method.

running the OS (Operating System) and processing the incoming or outgoing data. A Blackfin DSP processor, which is the one we use, has multiple, flexible and independent Direct Memory Access (DMA) controllers. DMA transfers can occur between the processor’s internal memories and any of its DMA-capable peripherals.

Additionally, DMA transfers can be performed between any DMA-capable peripheral and external device connected to the external memory interfaces, including the SDRAM controller and the asynchronous memory controller. Besides other interfaces, the Blackfin processor provides a Parallel Peripheral Interface (PPI) that can

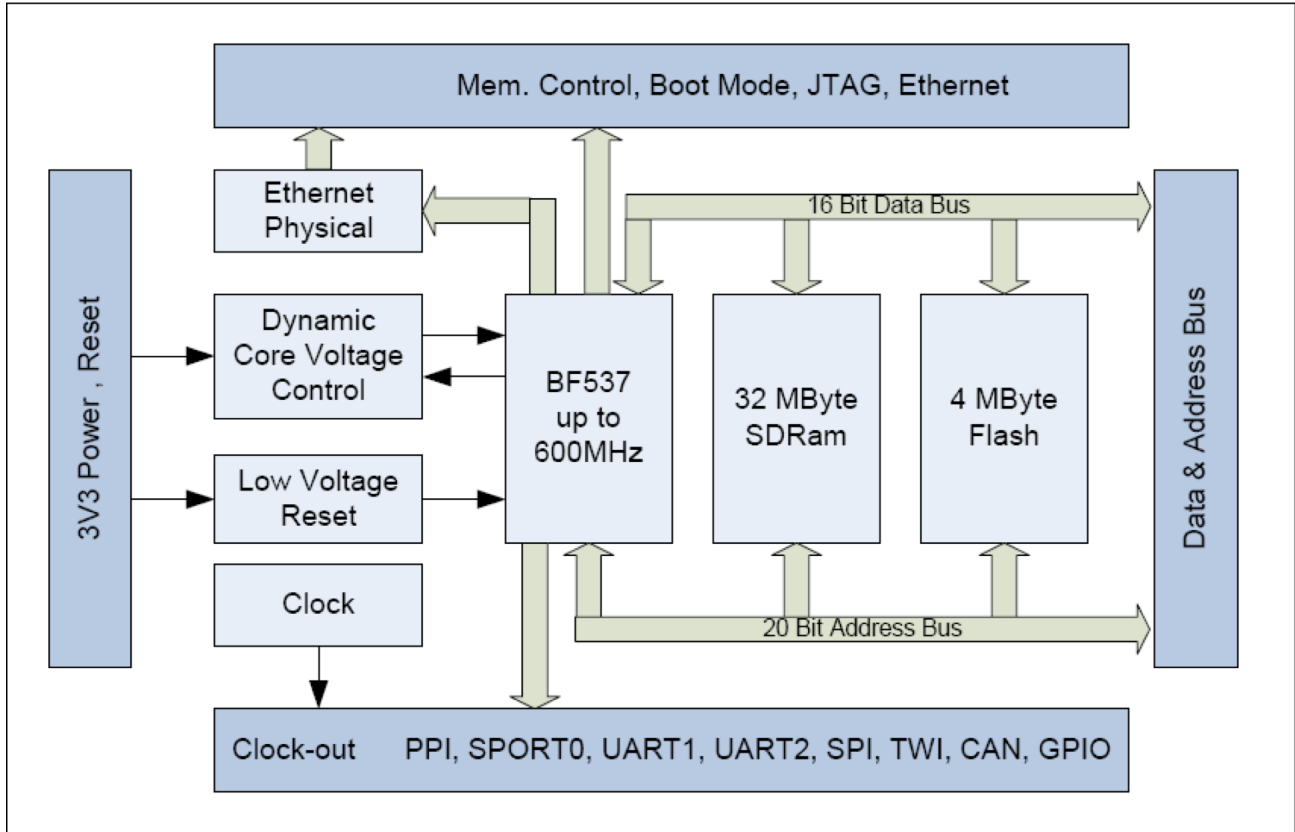


Figure 3. Detailed block diagram of CM-BF537E core module taken from reference [11].

connect directly to parallel D/A and A/D converters, ITU-R-601/656 video encoders and decoders, and other general-purpose peripherals, such as the CMOS camera sensors. The PPI consists of a dedicated input clock pin, up to 3 frame synchronization pins, and up to 16 data pins.

Overall, Blackfin processors offer a good price performance ratio (800 MMAC at 400 MHz for less than \$5/unit in quantities) and advanced power management functions. It provides a very low power, cheap and space-efficient solution. Besides, Blackfin's advanced DSP and multimedia capabilities qualify it not only for audio and video appliances, but also for all kinds of industrial, automotive, and communication devices. Moreover, Blackfin development tools are generally well tested, well documented and include everything necessary to get started and successfully finish in time. Much of the information, gathered on uLinux/Blackfin architecture for our research, has been taken from reference [3]. Thus, the above factors form the main motivating ground for our decision of moving PICO into Blackfin DSP architecture..

4. Components forming the Blackfin DSP architecture for our research

This section describes the components of the tinyboard Blackfin DSP of our embedded real-time DSP application.

4.1. CM-BF537E core module

We have chosen the Bluetechnix Blackfin DSP processor CM-BF537E, which is one of the latest members of the Blackfin family, and is unique because it is a tiny, high performance and low power DSP/RISC core module with an on-board 10/100Mbit Ethernet interface (that includes a 10/100 physical transceiver chip). The CM-BF537E module can be easily used as a stand-alone module for a camera system requiring only power supply and the direct attachment of a compatible video camera. This makes it ideal for our cheap PICO webcam application. The details of this DSP processor module are shown in Figure 3.

4.2. EVAL-BF5xx board

We have chosen the Bluetechnix EVAL-BF5xx Board as the low cost and lightweight platform for our CM-BF537E DSP processor. This small baseboard has all hardware necessary to test the performance of the processor core modules including a high-speed serial port directly connectable to a computer USB port, a digital video camera interface and a SD-Card mass storage device socket. The details of this component can be obtained from reference [11].

4.3. EXT-BF5xx camera board

To go along with our Blackfin DSP processor module, we have chosen the OV7660 Omnivision color camera that comes along with the Bluetechnix EXT-BF5xx-Camera Board. It is an extender plug-on board for any EVAL-BF5xx board and includes the option of two Omnivision CMOS cameras, that allow quick integration of mono or stereo imaging applications with any Blackfin based core module. The details of this component can be obtained from reference [11].

5. Why uCLinux?

We have chosen to focus on uCLinux as our embedded operating system because of its support to a very large number of existing applications, the provisions for easy system (kernel and user space) configuration and the low cost. Many of the standard Linux applications, released under various open source licenses, can be cross compiled to run on the uCLinux system and thus contribute to this real time development environment of Linux. This makes applications like the basic web services and OpenSSL, along with image processing tasks simpler.

It is worth noting that uCLinux, however, is not a formal real-time system. Since Linux was originally developed for server and desktop usage, it has no hard real-time capabilities like most other operating systems of comparable complexity and size. Nevertheless, Linux, and in particular, uCLinux have excellent so-called “soft real-time” capabilities. This means that while Linux or uCLinux cannot guarantee certain interrupt or scheduler latency compared to other operating systems of similar complexity, they show very favorable performance characteristics. In Linux kernel 2.6.x (the new stable kernel release that comes with uCLinux), the real-time development qualities have been improved with the introduction of the new O(1) scheduler. More performance related details are discussed in reference [3].

5.1. uCLinux Vs BlackSheep

BlackSheep environment comes as the default built-in platform for most of the Blackfin family DSP modules. It is a basic framework to begin working inside the Blackfin processor chip, and can be operated easily through the Windows Hyperterminal communication program. Unlike, the uCLinux kernel, it has a boot loader, which is smaller in size and easy to load, taking into consideration the limited 4 MB flash memory available in most of the Blackfin DSP chips (including ours). However, when it comes to building competitive real-time embedded applications, BlackSheep generally loses out to uCLinux because of its limited features like restricted set of basic commands as compared to that of uCLinux, which supports all commands and facilities provided by Linux. Besides, development of applications that can be run in BlackSheep environment requires the exclusive Visual DSP tool, unlike its rival uCLinux, which provides the same framework as Linux for application development. Overall,

uCLinux is the preferred real-time operating system, with its superior real-time qualities and universal cross-compiler support for a variety of applications, drivers and libraries. The timing results, which we get during image capture from our camera in the uCLinux environment, strengthen the claim of uCLinux as a better choice. These results are shown later in the next section 6. They show that the total frame capture time, for our real-time application in uCLinux, is in the order of thousands of microseconds only, including the time taken for image processing (image encryption and image compression).

6. Blackfin DSP architecture powered by uCLinux: implementation of our embedded application

The advantage of using the Blackfin processor in combination with uCLinux is the availability of a wide range of applications, drivers, libraries and protocols, as open source or free software. In most cases, there is only basic cross compilation necessary to get that software up and running. In addition, invaluable tools such as MySQL and PHP enable developers to have the opportunity to develop the most demanding, feature-rich applications in a very short time frame. There is often enough processing power left for future improvements and addition of new features. The uCLinux toolchain provides easy, user-friendly configuration features, supporting the Blackfin architecture (including our hardware board devices). The context switch time for a default Linux 2.6.x kernel running on Blackfin/uCLinux has been found to be in the order of micro seconds (data taken from reference [3].)

- For our research, we first implement the Blackfin processor module along with the CMOS sensor camera, as an ethernet-based webcam application. We then capture the webcam images in the Bitmap (.BMP) image format, and then apply our ROI (face) detection and selective encryption enabled JPEG compression technique to obtain the final privacy-preserved image. We end up applying the PICO technique to the images captured from the webcam, thereby implementing a real-time embedded application, meant for quality surveillance and privacy protection. Here is the procedure that is employed for our privacy-enhanced embedded DSP camera application:-
- The CM-BF537E DSP chip is embedded on the EVAL-BF5xx board (which is a USB device capable of being connected to a computer), with the relevant hardware settings done to start and access the internal chip environment, in boot mode, from a computer, via a serial communication protocol.
- The uCLinux environment is then used to customize kernel settings according to the Blackfin target module (CM-BF537E) requirements. This involves

inclusion of all necessary drivers and application interfaces meant for our target configuration.

- The whole kernel and user space is then compiled and linked with the relevant uClinux libraries under the uClinux environment, and the customized uClinux kernel image, which is formed, is moved to our DSP module (through TFTP). We then build a uClinux boot environment inside the chip.
- In the mean time, a frame capture driver application was developed into an executable, within the uClinux environment (using the Blackfin-gcc compiler). This executable application is transferred to the uClinux environment inside the DSP chip.
- The EXT-BF5xx camera module is then connected and mounted on to the DSP chip contained EVAL-BF5xx board. Now, once we start running our frame capture application within the bootable uClinux environment, inside the CM-BF537E chip, the program application keeps running and capturing images from the CMOS Omnivision camera on the EXT-BF5xx camera board. With a proper webserver program set up, together with the above mentioned application, we can convert the integrated system into an ethereal webcam, once we configure the network for our ethernet-based camera and target device.

To capture and obtain the images from the CMOS Omnivision camera, our frame capture application communicates with the camera through the PPI interface by opening the PPI device driver, performing I/O controls and setting the number of pixels per line and the number of lines to be captured. It reads the I2C (Inter-Integrated Circuit) bus, and accesses the camera-register values. After the application invokes the read system call, the PPI driver arms the DMA transfer. The start of a new frame is detected through the PPI peripheral, by monitoring the Line and Frame-Valid strobes. A special correlation between the two signals indicates the start of frame, and kicks-off the DMA transfer, capturing the total number of pixels, given by the number of pixels per line multiplied by the number of line per sample. The DMA engine stores the incoming samples at the address allocated by the application. Our research finds the typical frame-capture time to be around 30,000 microseconds for our DSP architecture. Some of the timing results, recorded from the performance of our embedded camera application, are encouraging for us. They are given in the next section.

6.1. Results from Our Embedded Real-Time Application

From our research data collected over a period of time and over multiple frames, the average time taken for capturing an entire image (single frame) in Bitmap (.BMP format), without any image processing operations (like image compression and image encryption), has been found to be within the range of 20,000 microseconds to 40,000 microseconds.

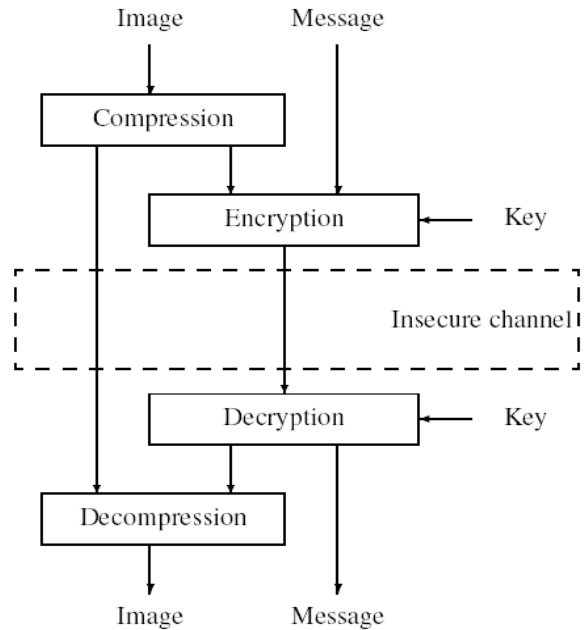


Figure 4. Outline diagram of PICO scheme. The actual encryption key is a public-key encrypted and stored with the data. To recover the key for decryption requires knowledge of the private key.

Our research has also indicated that the average time taken for capturing a JPEG image (single frame), without privacy encryption, is in the range of 210,000 microseconds to 240,000 microseconds.

Again, from the experimental data collected over a period of time and over several frames, the average time for capturing an image (single frame) of PICO encrypted JPEG format (i.e. after application of privacy-enhanced JPEG compression with ROI detection and encryption), is in the range of 240,000 microseconds to 260,000 microseconds.

All the above results discuss closed intervals, and a uniform image resolution of 320 x 240, with a constant DSP processor module master-clock frequency of 48 MHz. The image frame capture time varies according to the external factors like the position of the camera and the light exposure depending on the time of the day.

To enforce hiding of the relevant portions of the camera image like the human face (for protecting identity), we use the selective encryption technique for DCT based images like JPEG. The code implementation for this is carried out in uClinux, using C as the programming language (as employed in the source code for the JPEG library of the Independent JPEG Group as in reference [9]). The algorithmic process is discussed in the next section of our paper.

7. Selective encryption of Jpeg images for the PrivacyCam

We now discuss the cryptographic obscuration algorithm we use for the selective encryption of the Omnivision camera images. By selective encryption, we mean encryption of a certain area of the concerned image, which is our region of interest (ROI). This could be obscuring the human faces (based on face detection) or any other selected portion of the image, for the basic purpose of privacy enhancement in surveillance video. So, the selective encryption process may be used for hiding any part of the given image, with the option of recovering the data later, if legally warranted, using the encryption key. Since we can revert to the original encrypted sensitive data using decryption, the process is invertible. It is this aspect that makes it acceptable and advantageous in many legal scenarios.

We have mainly dealt with web images in JPEG format (most common in today's IP based web cameras) for our research. Dealing with JPEG images, we have the options of doing the partial encryption during or after the compression. In our research, we have taken the approach of applying the encryption during the JPEG compression process just after the DCT quantization but before the lossless Huffman encoding. We perform the encryption (blockwise) at a typical 8x8 block by block level. Here, we use the public-key AES encryption technique, storing the public key, the session encryption key and the encrypted region's definitions all inside the JPEG file header as comments. The decryption details (as in the file header) can be publicly read but not used with the private key, thereby preserving the privacy details for the individuals in the captured video image. If there's a legal issue (crime scene), then, with proper authorization, the session key and other decryption details can be provided for obtaining the identity of the individual from the original data/image. This revertible (invertible) aspect of our embedded application makes it unique. Since we are doing a face encryption, which means encryption at one place/region, a typical 8 x 8 block encryption helps our cause; as for larger block encryption, we might "round off" the useful data, thereby losing appropriate features of the image. In fact, AES algorithm involves boundary alignment or padding of the data boundaries. This might lead to "rounding off" or "blocking" artifacts at the boundaries.

8. Conclusions and future scope of work

This paper presents and demonstrates the idea of real-time embedded application of the PICO technique on uCLinux/Blackfin DSP architecture. Our research shows that when PICO technique is applied on a cheap, tiny, powerful DSP framework, implemented as a network/ethereal webcam, it can strike a fine balance between privacy and security, and address the problem of "privacy invasion" in a robust, efficient and cost-effective

way, in optimized space. It also establishes the fact that uCLinux/Blackfin architecture can make the implementation of the PICO concept much affordable and easy, thereby helping us to achieve further goals of more improved, enhanced vision and surveillance systems, where we can strike a perfect balance among privacy, security, cost and space.

There is scope of future research work in the implementation of the selective encryption technique on the uCLinux/Blackfin architecture, using the methods of motion detection or skin detection. The use of Blackfin core modules and boards, in implementing surveillance webcams can help in achieving security in more private places like restrooms, private household, inner chambers, etc. Moreover, since the Blackfin DSP modules and boards are flexible to use, and meant for audio and other multimedia applications, they have the potential to extend our current embedded application from only video to both audio and audio-video surveillance. Thus, with uCLinux/Blackfin DSP architecture providing a new, better way of developing real-time embedded applications, we can think in terms of more balanced, secure, efficient, cost-effective and space optimized computer vision applications, which will compete seriously with the currently existing, typical video surveillance systems.

References:

- [1] T.E. Boulton, *PICO: Privacy through Invertible Cryptographic Obscuration* - IEEE Computer Vision for Interactive and Intelligent Environments, 2005.
- [2] Gregory K. Wallace, *The JPEG Still Picture Compression Standard* - IEEE Transactions on Consumer Electronics, Vol. 38, No. 1, February 1992.
- [3] Michael Hennerich, *Linux on the Blackfin DSP Architecture* - Embedded Systems Conference Silicon Valley 2006.
- [4] J.M. Rodriguez, W. Puech and A.G. Borsb, *A Selective Encryption for Heterogeneous Color JPEG Images Based on VLC and AES Stream Cipher* - Third European Conference on Color in Graphics, Imaging and Vision, June, 2006.
- [5] W. Puech, P. Meuel, J.C. Bajard and M. Chaumont, *Face Protection by Fast Selective Encryption in a Video* - IET, Crime Security Conference June, 2006.
- [6] Marc Van Droogenbroeck, *Partial Encryption of Images for Real-time Applications* - Fourth IEEE Signal Processing Symposium, pages 11-15, April 2004.
- [7] D.H. Flaherty, *Video surveillance by public bodies: A discussion*, Investigation report P98-012, Information and Privacy Commissioner for British Columbia 1998 <http://www.oipcbc.org/investigations/reports/invrpt12.html>.

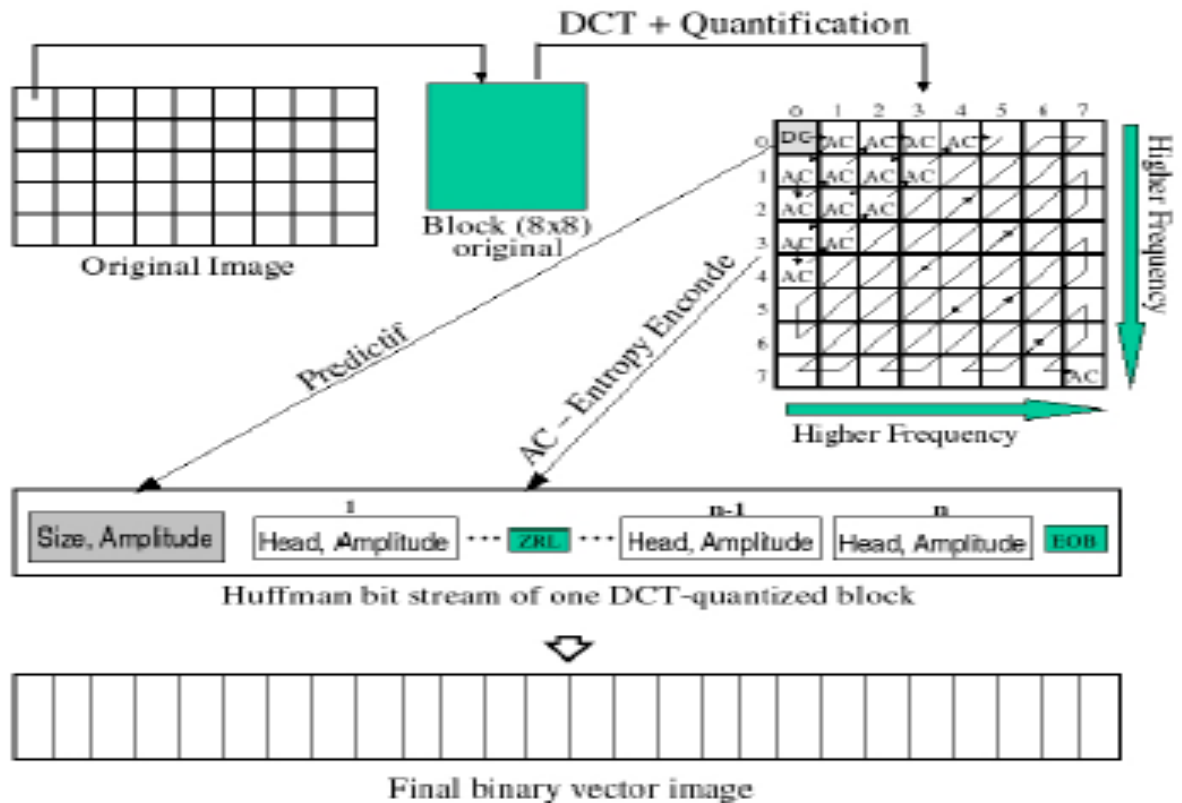


Figure 5. JPEG overview for the selective encryption process, take from reference [4].

- [8] Andrew Senior, Sharath Pankanti, Arun Hampapur, Lisa Brown, Ying-Li Tian, Ahmet Ekin, *Blinkering Surveillance: Enabling Video Privacy through Computer Vision* - IEEE Security & Privacy, Volume 3, (no 3), pages 50-57 in 2005.
- [9] Independent JPEG Group Website -<http://www.ijg.org/> Open Source Code for JPEG Algorithm.
- [10] OpenSSL Website <http://www.openssl.org/> -Open Source Library for Cryptographic Functions.
- [11] Bluetechnix Website - <http://tinyboards.com> Manuals, Information and Guidelines for Blackfin modules and boards used.
- [12] Blackfin uCLinux Group and Forum Website <http://blackfin.uclinux.org/gf/>.