DOI. 10.1109/SACI.2009.5136263

Steganography on embedded devices

Daniela Stanescu, Valentin Stangaciu, Ioana Ghergulescu, Mircea Stratulat,
Department of Computer and Software Engineering (DCSE), "Politehnica" University of Timisoara, 2, Vasile Parvan Blvd., 300223 – Timisoara, Romania
daniela.stanescu@cs.upt.ro, valys@dsplabs.cs.upt.ro, ioanag@ms.upt.ro, smircea@cs.upt.ro,

***Abstract* — Steganography is a science dealing with writing hidden messages in a specific way that only the sender and the intended recipient are able to decipher. Important domains, besides classic computing, where steganography can be applied are domains using mobile and embedded devices. This paper focuses on the implementation of a steganographic algorithm on embedded devices – microcontrollers. Furthermore, this article presents experimental results obtained from testing a steganographic algorithm on embedded devices. The main purpose of implementing such an algorithm on a microcontroller is to bring steganography and its advantages on low and medium cost mobile and dedicated devices.**

***Keywords* — steganography, concealing, embedded, mobility.**

I. INTRODUCTION

Nowadays, an important aspect of the modern way of life is *communication*. Many devices present today have the ability to transmit various information between themselves using different ways of communication, like insecure public networks, different types of wireless networks and the most used: the Internet. In some cases it is needed to keep the information travelling through different kinds of channels secret.

Mainly there are two ways of concealing information: cryptography and steganography.

Cryptography's main aspect is that the information is somehow distorted, scrambled by the sender using normally an *encryption key* also known only by the intended receiver who decrypts the message. The problem with cryptography is that a user intercepting the message, although he cannot decrypt it, he might detect that there is an encrypted, secret information.[1]

On the other hand steganography is able even to hide this aspect making sure that even the fact that there is secret information, is concealed. Steganography's main aspect is that it is embedding the secret message into another message [2].

Mainly, steganography can be used for concealing important information within computer files such as documents or image files in such a way that only so called authorized users know and can extract the information. The advantage over classic cryptography is that messages hidden using steganography techniques do not attract attention on themselves.

Before continuing this discussion additional terminology needs to be added. In general, steganography terminology is analogous to more conventional radio and communication technologies. The most important terms in steganography are the following:

- *The payload* – is the data that is needed to be transported, the data needed to be hidden
- *The carrier* – is the signal, data file or stream into which the valid data, the payload is hidden
- *Channel* – is a term used to refer to the type of input
- *The package, stego file* or *covert message* – is usually the resulting signal, stream or data file

II. RELATED WORK

Significant results have been obtained hiding information into text. The main goal was to discourage illegal document copying by making documents with line-shift encoding, the lines of text being shifted up or down thereby encoding a serial number. [3]

Steganography was also used to embed data into an audio signal by manipulating characteristics of the audio signal below the level of perceptibility. These techniques were useful in applications such as annotation, captioning and the automatic monitoring of radio advertisements. [4]

Important steps have been made hiding information using digital watermarking using improved techniques based on the decorrelation property of the Karhunen-Loeve Transform [5] as well as a method of hiding messages in digital images based on YUV format and its derivatives [6]. These techniques were used especially against a current common issue like illicit copying and distribution of copyright material.

Studies and tests were made for elaborating different techniques and algorithms to embed large amount of data into a picture as well as the requirements needed.

Other steps have been taken towards eliminating detection of steganography and counteracting attacks meant to extract the hidden information. New and more complex algorithm have been developed to avoid the detection of hidden data as well as embedding hidden information in preprocessed images and in images where compression was applied. [7]

Other related work on the subject is about using steganography to insert a video or audio message in the cover in real time, using a secret key steganographic micro-architecture employing Field Programmable Gate Arrays [8]. Furthermore, devices such as Field Programmable Gate Arrays (FPGA) also hosted steganalysis (the reverse process of steganography) algorithms.

III. Brief Description of Embedded Hardware Used For Implementing Steganography

This paper focuses on the aspect of using steganography for hiding information in communication between various mobile or embedded devices.

As shown above significant steps have been made in bringing steganography on dedicated devices using FPGAs. The major advantage of using a FPGA for steganography is speed, a FPGA being able to execute steganographic algorithms much faster than other devices. The disadvantage in using an FPGA is cost, making them impossible to be used in mobile phones for example.

This paper tries to enlarge the possibilities of using steganographic algorithms. The main concern of this paper is to bring steganographic algorithms like the LSB algorithm on mobile and embedded devices without significantly rising their price. One possible solution presented here is using microcontrollers for hosting and executing a steganographic algorithm.

The hardware used consists of the *Olimex LPC-H2294* development board. The board's main components that were used in the implementation are: an ARM7 based microcontroller unit (*NXP Phillips LPC-2294*), 1 MB (256 K x 32 bit) 12 ns 71V416 SRAM.

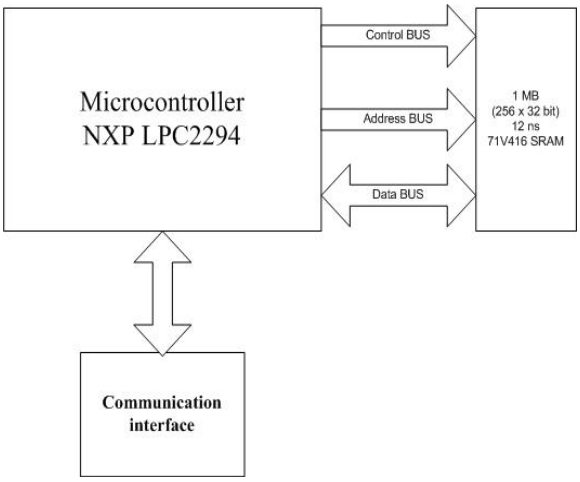


Figure 1. Brief system schematic

The main processor is an ARM7TDMI-S based microcontroller with 256 KB of embedded high speed flash memory having a large variety of peripheral interfaces.

The ARM architecture is based on *Reduced Instruction Sets Computer* (RISC) principles. The main characteristic of a microprocessor or microcontroller based on the ARM architecture is simplicity resulting in high speed and high instruction throughput. Also pipeline techniques are present within the architecture making sure that all parts of the processing and memory systems can operate continuously. [10], [11], [12].

The Phillips LPC-2294 microcontroller also has a peripheral module (EMC – external memory controller) allowing it to be connected with external memory through a dedicated address and data busses. The EMC unit has an important feature, allowing the external memory to be transparent for the embedded programmer.

The Olimex LPC-H2294 development board also consists of a JTAG debugging interface. The main development tool used was *uVision 3* provided by *Keil*.

IV. Implementation Details

The implementation of steganography algorithms on embedded devices was mainly focused on the procedures regarding the decoding an encrypted image within a carrier image.

The first step in the implementation of the decoding steganography algorithms was designing a proper memory organization. The main aspects that were considered during the memory organization process were influenced by the size of the images that needed to be stored and processed. The Phillips LPC-H2294 microcontroller disposes of internal RAM memory containing only 16 KB of memory. The amount of RAM memory contained is insufficient for storing the image needed to be decoded. Another memory is contained within the microcontroller: 256 KB of internal FLASH memory, also being insufficient for storing the image, most part of the memory being used by the code and, being a FLASH memory, it is low on speed, making decoding a large time consuming process.

The solution in storing the image to be processed is to use the available external RAM memory provided on the Olimex development board presented above. The external RAM memory is connected to the microcontroller's core via the External Memory Module (EMC) provided by Phillips LPC2294.

Fig. 1 shows a block schematic of the system used for implementation.

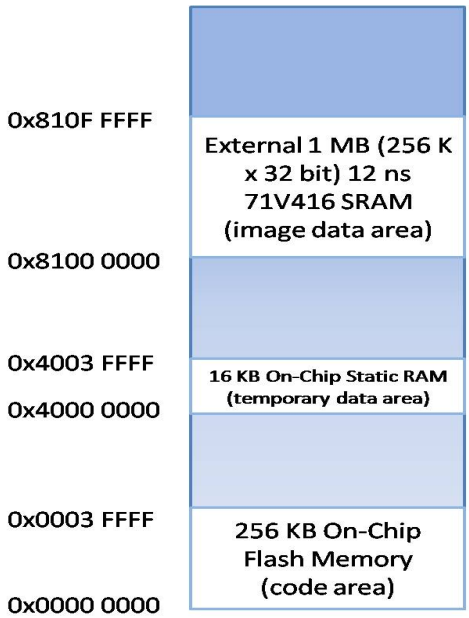


Figure 2. Embedded system memory map

Fig.2 presents the memory map used in the implementation of the steganography algorithms. As shown, the external 1 MB memory is mapped by the Phillips LPC2294 microcontroller at address 0x81000000.

The internal 16 KB RAM memory provided by the microcontroller is only used for temporary and intermediate data.

The image to be decoded is stored in the external memory discussed above and is overwritten by the resulting image, the payload. All images processed by the device use RGB pixel format. Each pixel is composed by the three color components (red, green, blue) each using 8 bits of memory – 1 byte. The total size of one pixel is 3 bytes.

The external memory stores only the pixel matrix of the image, other information not being related to this topic. The pixel matrix is organized in a linear way, a flatten matrix as shown:

$$R_0G_0B_0R_1G_1B_1...R_{n-1}G_{n-1}B_{n-1} \quad (1)$$

The R_i, G_i, B_i represent the color components for pixel *i*. Giving the dimension of the pixel matrix as *MxN*, the parameter *n* can be defined as: $n = M \cdot N$. Thus the memory size being 1 MB and using the pixel memory size as above, at the value of 3 bytes, the maximum image size is:

$$n_{[B]} = \frac{1MB}{3B} = \frac{1048576}{3} \approx 349525 \quad (2)$$

Using this linear format for storing the image facilitates the implementation of any pixel manipulating algorithm over the image.

The steganographic algorithm that was implemented on the device is the LSB algorithm.

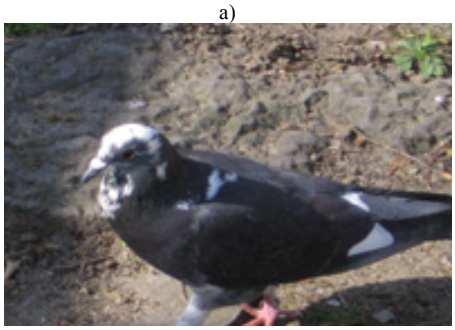
The LSB algorithm is altering the last significant bit, changing it with a bit from the hidden message [13]. Figure 3 presents the pseudo code of the LSB algorithm. By altering only the LSB of a pixel, human eye will still not be able to distinguish a difference.

```
for (every_pixel)
   read pixel
   get the RGB color components of pixel
   for(every RGB color component)
    if LSB(color component)!=message(k)
       then LSB(color component)=message(k)
    endif
    increment(k)
   endfor
   write new RGB pixel to image
endfor
```

Figure 3. Pseudo code of the LSB algorithm

The image used for testing the implementation has the following dimension: 200x135. The total amount of memory space being used by the image consists of a total of 81000 B.

a)



b)

Figure 4. a) carrier image b) payload

The images shown above in Fig. 4 represent the payload image (Fig 4. b) to be hidden within the carrier image in Fig 4.a.

After the encrypting algorithm was applied on the two images above a third image was generated, an image containing the payload hidden within the carrier, the "stego image":

Figure 5. Stego image

The image above contains the payload image hidden within and shows no sign of the payload image and the fact that there may be any hidden messages within, thus respecting the steganography principles.

This resulting image was constructed using a general purpose personal computer; this paper only focuses on the implementation of the decoding algorithms on an embedded system. There are certain limitations when encoding a payload image into the carrier. After the encoding was made, the package image containing both the payload and the carrier may not be converted into any image format that uses compression, or any other format that requires further image processing.

The image above shown in Fig. 5 is a result of applying the 2 bit LSB algorithm on the images shown in Fig. 4.

The same result is obtained after applying the 3 bit LSB algorithm on the same images.

The notable difference between the 2 bit LSB algorithm and the 3 bit LSB algorithm occurs at the decoding level, the 3 bit LSB algorithm being able to reconstruct the payload image from the package with higher precision.

a)

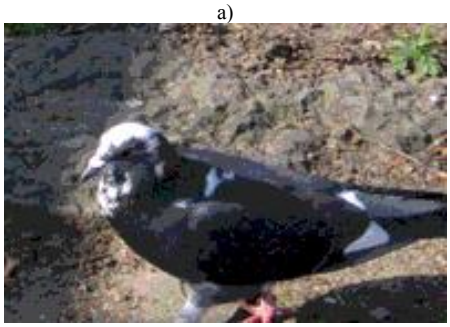


Figure 6. a) decoded payload using 2 bit LSB algorithm b) decoded payload using 3 bit LSB algorithm

As shown above in Fig. 5, the 3 bit LSB algorithm was able to extract the payload with higher efficiency than the 2 bit LSB algorithm.

Measurements show that although the 3 bit LSB extracts the payload with greater accuracy, it consumes more time than it's simpler version, the 2 bit LSB algorithm.

TABLE I

EXECUTION TIME OF IMPLEMENTED STEGANOGRAPHY ALGORITHMS FOR DECRYPTION

Algorithm	***Execution time***
LSB – 2 bit	39.83 ms
LSB – 3 bit	48.07 ms

The table above shows the execution time of the two algorithms implemented. The execution time was measured on the embedded platform using dedicated timers provided by the Phillips LPC2294 microcontroller.

The measurements were made on the test images presented above, at the size of 200 x 135.

Additional measurements show that the execution time linearly increases as the image size increases.

TABLE II

EXECUTION TIME MEASUREMENTS FOR DIFFERENT IMAGE RESOLUTIONS A) 2 BIT LSB ; B) 3 BIT LSB

Image resolution	*Execution time [ms]*
200x135	39.83
300x300	132.75
450x450	298.69
500x500	368.75
590x590	513.45

A)

Image resolution	*Execution time [ms]*
200x135	48.07
300x300	160.22
450x450	360.49
500x500	445.05
590x590	619.69

B)

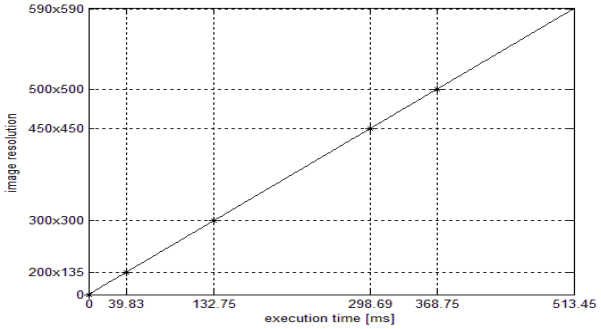


Figure 6 – relation between execution time and image resolution for 2 bit LSB algorithm

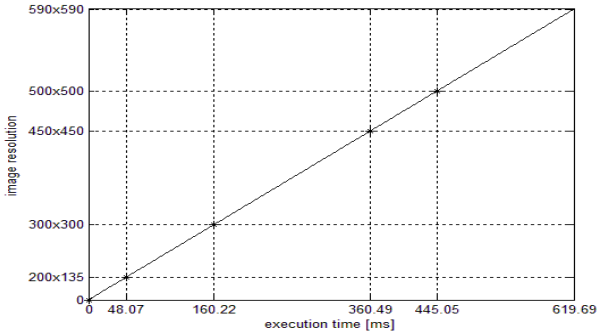


Figure 7 – relation between execution time and image resolution for 3 bit LSB algorithm

The two figures above (Fig. 6, Fig 7) show the relation between the image size and the time needed by the Phillips LPC2294 microcontroller to decode the encrypted payload image.

The execution time measured above is highly dependent on the external memory access times and more particular on the performance of the EMC peripheral within the Phillips LPC2294 microcontroller.

Better results may be obtained by using high speed memory connected to the processor on a higher speed dedicated bus. The fact in this situation is that the memory is connected to the processor via the External Memory Controller which is a peripheral working at a speed lower than the main processor core. A solution would be to design a configuration in which a high speed memory is connected directly to the processor core.

V. CONCLUSIONS

This paper presents a possible implementation of steganography algorithms on an embedded platform as well the analysis regarding the amount of time needed by the embedded microprocessor to extract the payload image from the carrier for different image resolutions.

The main benefit of this implementation is that it brings steganography on a level very common nowadays, the mobility. Using steganography on mobile devices may improve security on data transfers without additional significant cost.

Furthermore, this paper focuses on using microcontrollers or microprocessors for executing the steganographic algorithms instead of using Field Programmable Gate Arrays. This way, when adding steganography capabilities on an embedded device the cost is not mainly influenced as it could be if a FPGA module is added to the device.

Steganography may also help hiding secret information in communication lines, for example embedded modules with steganographic encrypting and decrypting capabilities connected between two systems. An embedded device with steganography implemented on it can be used in secret communication.

The algorithms used in this paper are spatial algorithms, processing the images on the pixel level. Further work on this subject may be implementing frequency based algorithms on embedded devices, as well as perfecting the hardware used in the implementation for obtaining better execution time for the decoding operation.

Furthermore, an important task is also to bring steganographic coding algorithms on embedded or mobile devices.

REFERENCES

[1] Neil F. Johnson, Sushil Jajodia, "Exploring Steganography: Seeing the Unseen", *IEEE COMPUTER, vol. 31, 1998, pp. 26--34*

[2] József Lenti, "Steganographic methods", *Department of Control Engineering and Information Technology, Budapest University of Technology and Economics, H-1521, Budapest, Hungary, June 2000, pp. 249-258*

[3] J. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman, "Electronic marking and identification techniques to discourage document copying," *Selected Areas in Communications, IEEE Journal on, vol. 13, 1995, pp. 1495-1504.*

[4] D. Gruhl, W. Bender, A. Lu, "Echo hiding", in *Information hiding: First International Workshop, R.J. Anderson, Ed. Vol. 1174 of Lecture Notes in Computer Science, Isaac Newton Institute, Cambridge, England, May 1996, pp. 295 – 315*

[5] D. Stanescu, M. Stratulat, B. Ciubotaru, D. Chiciudean, R. Cioarga, and D. Borca, "Digital Watermarking using Karhunen-Loeve transform," *Applied Computational Intelligence and Informatics, 2007. SACI '07. 4th International Symposium on, 2007, pp. 187-190*

[6] Stanescu, D.; Stratulat, M.; Groza, V.; Ghergulescu, I.; Borca, D. "Steganography in YUV color space", *Robotic and Sensors Environments, 2007. ROSE 2007. International Workshop on Volume , Issue , 12-13 Oct. 2007*

[7] Taras Holotyak, Jessica Fridrich, Sviatoslav Voloshynovskiy, "Blind Statistical Steganalysis of Additive Steganography Using Wavelet Higher Order Statistics", *Communications and Multimedia Security, 2005, Volume 3677/2005, ISBN* 978-3-540-28791-9, *pp. 273-274*

[8] H.A. Farouk and M. Saeb, "Design and implementation of a secret key steganographic micro-architecture employing FPGA," *Proceedings of the 2004 conference on Asia South Pacific design automation: electronic design and solution fair, Yokohama, Japan: IEEE Press, 2004, pp. 577-578.*

[9] Kang Sun, Xuezeng Pan, Jimin Wang, Lingdi Ping, "Signal Processing for Image Enhancement and Multimedia Processing", *Pages 269 – 278, Springer US, December 2007*

[10] ARM ARM7TDMI-S (Rev 4), "Technical Reference Manual"

[11] Phillips Semiconductors, "LPC2119/2129/2194/2292/2294 User Manual", *May 2004*

[12] Trevor Martin, "The Insiders Guide to the Phillips ARM7-Based Microcontrollers, An Engineer's Introduction to the LPC2100 Series"*, 2005, Hitex Ltd.*

[13] Neeta, D.; Snehal, K.; Jacobs, D., "Implementation of LSB Steganography and Its Evaluation for Various Bits", *Digital Information Management, 2006 1st International Conference on Volume, Issue, 6-6 Dec. 2006 Page(s): 173 – 178 Digital Object Identifier 10.1109/ICDIM.2007.369349*