International Journal of Computer Applications (0975 – 8887)
Volume 4 – No.4, July 2010

Design and Analysis of Prototype Hardware for Secret Sharing Using 2-D Image Processing

R.Amirtharajan
Assistant Professor
School of Electrical & Electronics Engineering
SASTRA University, India

Dr. R.John Bosco Balaguru
Associate Dean Research
School of Electrical & Electronics Engineering
SASTRA University, India

Vivek Ganesan
Assistant Systems Engineer-Trainee
Tata Consultancy Services

ABSTRACT

Information security has been a major concern with the striking advancement in communication and information techniques. Details of significance such as corporate data, secret message and private details have to be shielded from any unauthorized handling or spiteful approaches. Protection of such valuable information over insecure networks can be accomplished by monitoring or filtering of all packets in the data hiding technology. Images can be used for covert sharing of data. Any type of data can be practically hidden inside any image, without detectably affecting the quality of the image. This can be done by substituting some information of the image with the secret information in carefully chosen ways. The image can then be sent by electronic mail, where it appears as a casual attachment. The receiver can get the secret data by applying reverse transform. Hardware modeling of this process can impart portability and improve the speed of the same. This paper discusses the design and analysis of prototype hardware to perform secret sharing using 2-D Image Processing.

Categories and Subject Descriptors

D.2.11 Information hiding
D.4.6 Security and Protection

General Terms

Information Security, Image Processing

Keywords

LSB Steganography, Information hiding.

1. INTRODUCTION

Today's world scenario is clearly proving the statement "Information is wealth". With the improvement of information technology, one can get almost any casual information in seconds. Casual information neither has any strategic importance nor is it privately held. There is another class of information called classified information, which is usually not given to the public. The classified information has been conventionally protected by Cryptosystems [4].

Due to the increase in computer power, the internet and with the development of digital signal processing, information theory and coding theory, steganography became digital. Steganography has created an atmosphere of corporate vigilance that has generated various interesting applications, as a result its continuing evolution is guaranteed.

Furthermore the main concern of the music, film, book and software publishing industries is to provide copyright protection. Since the audio, video and other media are available in digital form, it leads to massive unauthorized copying. Thus, there has been significant research in digital "watermarks" (hidden copyright messages) and "finger prints" (hidden serial numbers); the idea is that the latter can help to identify copyright violators, and the former to prosecute them.

Steganography comes under the broader classification of information hiding. Steganography was first explained by the prisoner problem [7, 8]. Alice and Bob are two prisoners and they want to communicate with each other for making an escape plan. Wendy is a warden who can be passive, active or malicious. The type of warden may vary depending upon how she can change the message passed by Alice and Bob. If Wendy is a passive warden then she eavesdrops and won't make any changes. But if she is an active warden she might make small changes and if she is a malicious warden she will either make large changes or suppress the message. If the warden determines that a message has been communicated between the prisoners then they will be put into solitary confinement and they won't be able to communicate and further could not execute their escape plans.

Most of the steganographic methods are subliminal. The secret information which is being passed from Alice to Bob is hidden from Wendy, but it is necessary, that the stego keys are exchanged [1-3, 5-8] in prior to sending messages. The goal of steganography fails if Wendy perceives that some information has been sent. The Goal of Steganography can be summed up as the transmission of hidden information by avoiding suspicion. If the message is being detected then steganographic method fails. The goal of steganography is no restricted to subliminal methods of steganography.

Presently, all the steganographic methods [1-8] need a computer to hide the secret. This poses serious restrictions on portability, cost and speed performance of the system. The proposed work is FPGA Implementation of LSB embedding and recovery modules to overcome the above stated limitations. Instead a chip is designed that automatically embeds the given data in an image. If the user has a flash drive, the chip receives the data, embeds it within an image and sends it back to the flash drive. This is the encryption module. A retrieving module is also designed that retrieves the embedded data from the cover.

This paper aims at explaining one such method of secret sharing through images and developing a hardware prototype for the process.

2. DEFINITION OF TERMS

Cover Image:

Cover Image is a normal image, unrelated to the secret being shared. It acts as a carrier for secret information.

Stego Image:

Stego Image is the image obtained after embedding the secret information into the cover image. It is the image sent to the recipient as an electronic mail attachment.

2.1 ALGORITHMS INVOLVED

2.1.1 Two Dimensional Sampling of Images

The images are stored in computers as digital data. The conversion of real world image of infinite bandwidth into a digital image of finite bandwidth is accomplished by means of a process called two dimensional sampling.

In the process of two dimensional sampling, the real world image is sampled in both x and y directions. The intensity values of the particular positions of the image are stored and these positions are called as pixels.

The intensity levels of the real world image are quantized into 256 levels of intensity represented by 8 bits. Thus, a pixel's intensity can be stored in 8 bits. Thus, intensities of all the pixels are sequentially stored in the memory and read back to reconstruct the sampled version of the image. If the sampling interval in space is very small, the viewer cannot distinguish between the real world image and the sampled and digitally reconstructed image.

2.1.2 Distortion analysis of images

The images can be distorted by changing the bits in the stored pixels. The visual quality of the image is not noticeably distorted till four least significant bits of each of its pixel are modified.
This fact can be made use of in the field of data hiding. It is possible to embed the secret data in the four least significant bits of image pixels of the image. This does not change the way the image looks to a normal observer.

2.2 Assumptions

i. Let $A=\{a_k\}$, $k=1,2,3,\ldots,t$ be the textual information to be hidden, where a_k is the ASCII value of the character at kth position from beginning.
ii. Let the nontext media employed be a monochrome bitmap image I* with w as width and h as height.
iii. Let $B_k(X)$ denote the kth bit, starting from LSB of any integer X. For example, $B_0(X)$ denotes the LSB of integer X.

2.2.1 Hiding Algorithm

Step-1: Start

Step-2: Take normalized 2 dimensional samples of I* to give $I=\{i_{mn}\}$; $m=1,2,3,\ldots,w$; $n=1,2,3,\ldots,h$, such that i_{mn} is in the range [0.255] for all m,n. Here i_{mn} is an unsigned integer, referring to brightness of the image at pixel with spatial coordinates (m,n).

Step-3: Initialize m=1, n=1, k=1, b=0.

Step-4: while (k<=1) do

```
{
        while (b<8) do
        {
                $B_0(i_{mn}) = B_b(a_k)$
                n=n+1
                b=b+1
                if  (n==h+1) then
                {
                        m=m+1
                }
                if (m==w+1) then
                {
                   print "Insufficient cover size"
                }
        }
        b=0
        k=k+1
}
```

Step-5: Store the modified I as bitmap image (say J*)

Step-6: Transmit J* as attachment with an email.

Step-7: Stop.

2.2.2 Recovery Algorithm

Step-1: Start.

Step-2: Take normalized 2 dimensional samples of J* to give $J = \{j_{mn}\}$; $m=1,2,3,\ldots,w$; $n=1,2,3,\ldots,h$. Here j_{mn} is an unsigned integer referring to brightness of image at pixel with spatial coordinates (m,n).

Step-3: Initialize m=1, n=1, k=1, b=0.

Step-4: while (true) do

```
{       $a_k=0$
        while (b<8) do
        {
                $B_b(a_k) = B_0(j_{mn})$
                n=n+1
                b=b+1
                if (n==h+1) then
                {
                        m=m+1
                }
                if (m==w+1) then
                {
                        break
                }
        }
        b=0
        k=k+1
}
```

Step-5: Save $A = \{ a_k \}$ as recovered secret.

Step-6: Stop

3. HARDWARE MODELING

3.1 Need for hardware Implementation

In order to impart the following properties to the secret sharing process, hardware implementation is attempted.

i. Portability
ii. Connectivity with other systems
iii. Improved processing speed

3.2 Embedder Hardware Architecture

The embedding module embeds the bits of text data in the least significant bit of the cover image data and gives the stego image with the text embedded in it. This has an architecture which utilizes D-Flip Flops, Multiplexers, etc. as shown in Fig-1.

The two D Flip Flops take image data and text data as inputs. The text input is taken from the memory which has the text data. The image read frequency is eight times the text read frequency.

The blocks labeled '@k', where k=5,6,7,8 give out a logical '1' from 5th,6th, 7th and 8th clock pulses till the end of the 8th clock pulse in a 8-period cycle of image read frequency. This output signal selects the text bits to be output at the time interval corresponding to the least 'n' significant bits.

The value of k is chosen according to the relation **k=9-n.**

The @k blocks are made of 8bit shift registers which are initially loaded with all '1's in the first (9-k) bit positions and zeros in all other bit positions. The output of the LSB is fed back to the MSB of the shift register. This arrangement makes sure that a logical '1' is given out from start of kth clock pulse to end of 8th clock pulse. The 4:1 multiplexer selects the output of any one of the @k blocks based on the input n_minus_1 which is a binary representation for (n-1) where text has to be embedded in n lsbs of the image data.

The 2:1 multiplexer selects either image data bit or the text data bit depending on the output of the 4:1 multiplexer. Thus, n bit LSB embedding is possible using this simple architecture. the above block diagram shown in figure 1, the block labeled "@k" gives out clock pulses at bit intervals of range [k,8]. Here, k<9 and k € N. 'n' is the number of least significant bits to be used for embedding secret information. This hardware expects the cover image to be a two dimensional sampled image, with 8 bit resolution for each pixel.

3.3 RECOVERER HARDWARE ARCHITECTURE

In the recoverer block diagram as shown in Fig 2, the block labeled "clk@k" gives out the clock pulse at bit interval k. Here, k<9 and k€N. 'n' is the number of least significant bits to be used for embedding secret information. This hardware outputs the secret information when a stego image is input in it.

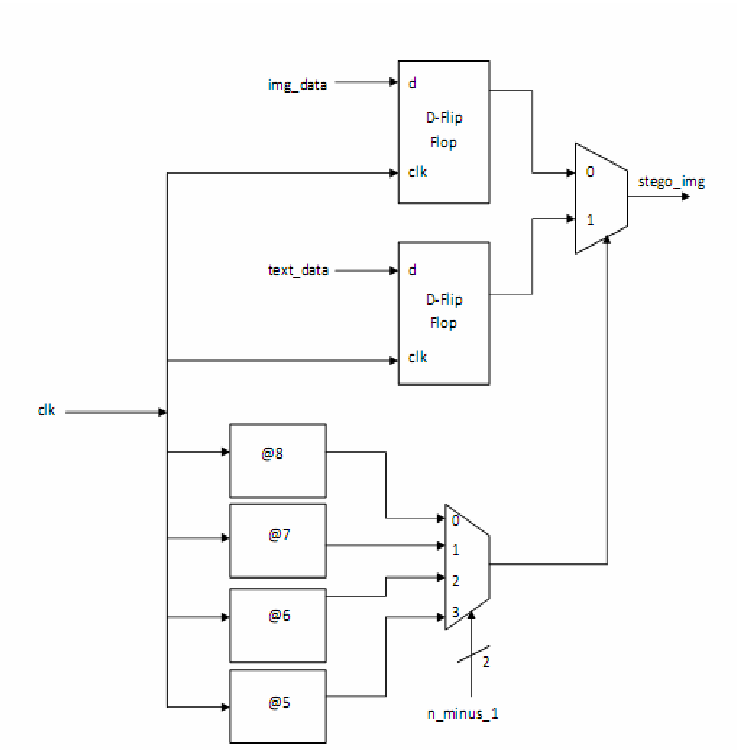


Fig.1. A block diagram showing hardware architecture of the embedder module.

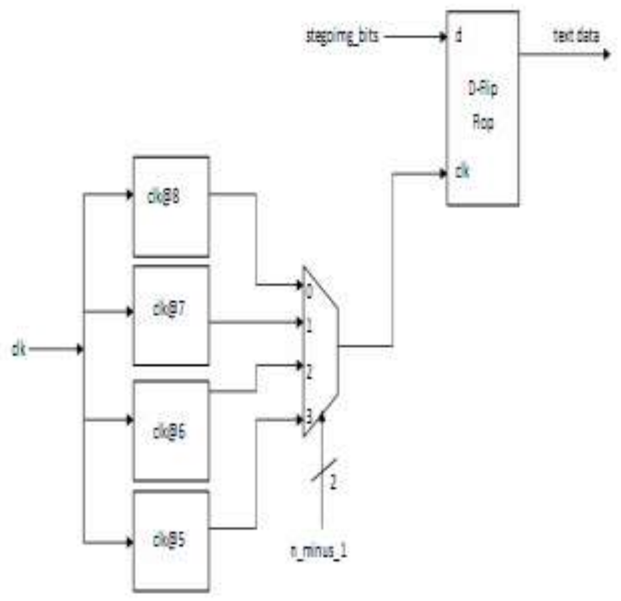


Fig.2. A block diagram showing hardware architecture of the recoverer

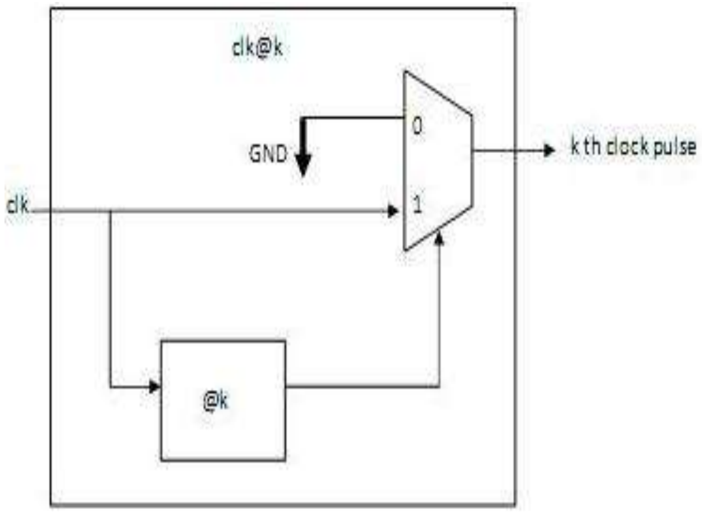


Fig - 3: A block diagram showing the architecture of clk@k block of recovery module.

The recovery module recovers the data embedded in the least significant bit of the stego image data and gives the text data back in its original form. This also has an architecture which utilizes D-Flip Flops, Multiplexers, etc. as shown in Fig-2.

The D-Flip Flop takes the stego image data as the input. This D Flip Flop is triggered by a clock only at the time intervals corresponding to n LSBs so that the text data is continuously given out at the output of D-Flip Flop.

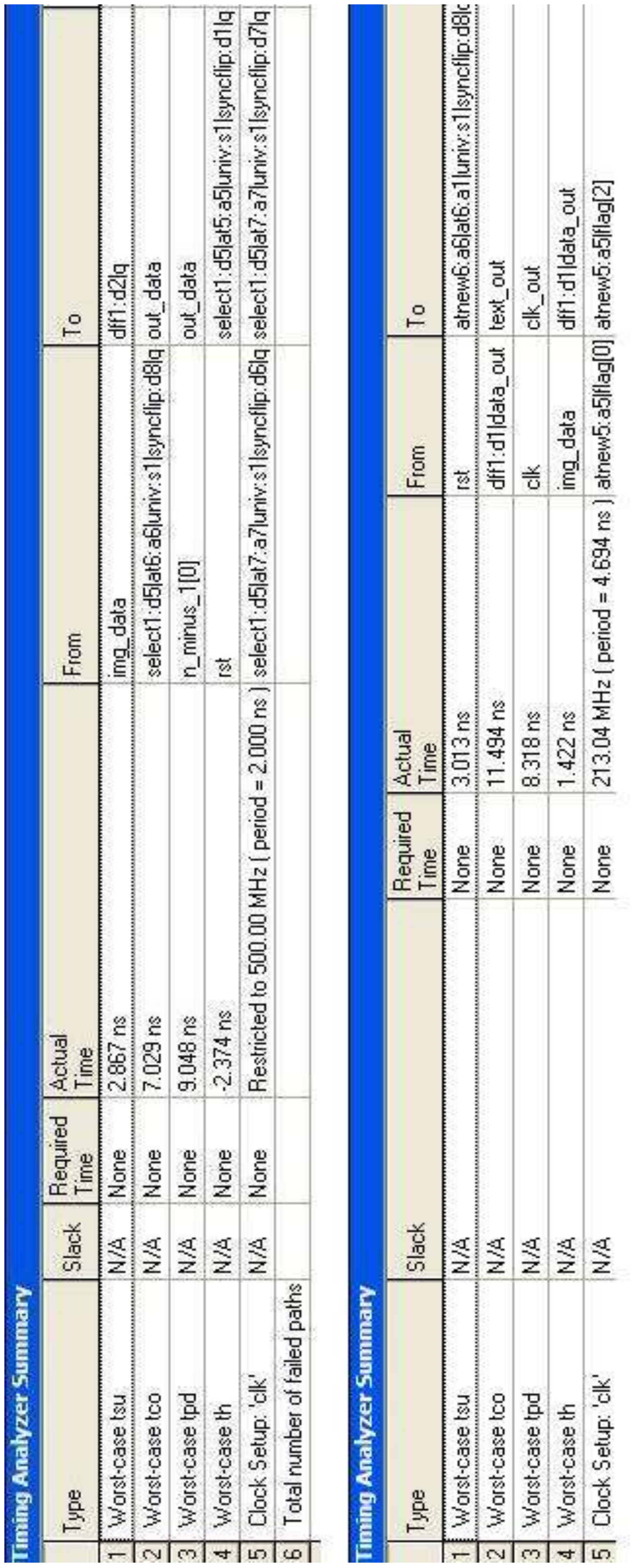
The blocks labeled 'clk@k' where k=5,6,7,8 give out a replica of the input clock pulses from k^{th} to 8^{th} clock pulse in a 8-period cycle of image read frequency. The output signal selects the text bits to be output at the time interval corresponding to the 'n' least significant bits.

The value of k is same as in the case of embedding.

The clk@k blocks have the architecture as shown in Fig-3. They are made of corresponding '@k' blocks along with a 2:1 multiplexer. This multiplexer selects either a logical '0' or the input clock based on the output from '@k' block. This arrangement makes sure that clock pulses are given only from k^{th} to 8^{th} bit interval of the image data.

The 4:1 multiplexer selects the output of any one of the clk@k blocks based on the input n_minus_1 which is a binary representation for (n-1) where text has to be embedded in n lsbs of the image data.

The output of this 4:1 multiplexer is given to the clk input of the D-Flip Flop which outputs the text portion of the input stego image. Thus, this architecture enables recovery.



Timing Analyzer Summary

	Type	Slack	Required Time	Actual Time	From	To
1	Worst-case tsu	N/A	None	2.867 ns	img_data	dff1:d2|q
2	Worst-case tco	N/A	None	7.029 ns	select1:d5|at6:a6|univ:s1|syncflip:d8|q	out_data
3	Worst-case tpd	N/A	None	9.048 ns	n_minus_1[0]	out_data
4	Worst-case th	N/A	None	-2.374 ns	rst	select1:d5|at5:a5|univ:s1|syncflip:d1|q
5	Clock Setup: 'clk'	N/A	None	Restricted to 500.00 MHz (period = 2.000 ns)	select1:d5|at7:a7|univ:s1|syncflip:d6|q	select1:d5|at7:a7|univ:s1|syncflip:d7|q
6	Total number of failed paths					

Timing Analyzer Summary

	Type	Slack	Required Time	Actual Time	From	To
1	Worst-case tsu	N/A	None	3.013 ns	rst	atnew6:a6|at6:a1|univ:s1|syncflip:d8|c
2	Worst-case tco	N/A	None	11.494 ns	dff1:d1|data_out	text_out
3	Worst-case tpd	N/A	None	8.318 ns	clk	clk_out
4	Worst-case th	N/A	None	1.422 ns	img_data	dff1:d1|data_out
5	Clock Setup: 'clk'	N/A	None	213.04 MHz (period = 4.694 ns)	atnew5:a5|flag[0]	atnew5:a5|flag[2]

Fig 4 a & b: A timing analysis report of VHDL implementation of LSB Embedder & Recoverer in Quartus II Software

5. DISTORTION ANALYSIS

Distortion is measured by means of two parameters namely, Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR).

The MSE is calculated by using the equation,

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(X_{i,j} - Y_{i,j}\right)^2 \qquad (1)$$

where *M* and *N* denote the total number of pixels in the horizontal and the vertical dimensions of the image *Xi, j* represents the pixels in the original image and *Yi, j*, represents the pixels of the stego-image.
The Peak Signal to Noise Ratio (PSNR) is expressed as
The PSNR is calculated using the equation,

$$PSNR = 10\log_{10}\left(\frac{I_{\max}^2}{MSE}\right) dB \qquad (2)$$

where I_{max} is the intensity value of each pixel which is equal to 255 for 8 bit gray scale images. Higher the value of PSNR better the image quality

Distortion Analysis of stego image using the software based secret sharing algorithm with 100% embedding of data gave the following results.

6. RESULTS AND DISUSSION

In this present implementation Lena and baboon 256 × 256 digital images has been taken as cover images as shown in Figure 5 a & b and tested for full embedding capacity for k= 1to 4 and the result is given in Table I

Fig.5a&b. The chosen cover images(Lena & Baboon).

TABLE I

Distortion Analysis of a 256x256 cover image

Cover Image	No. of Bits (n)	1	2	3	4
Lena	MSE (no unit)	0.5012	2.5761	10.2769	40.3637
	PSNR dB	51.131	44.0210	35.7189	32.0708
Baboon	MSE (no unit)	0.5031	2.5699	10.3083	39.4619
	PSNR dB	51.1134	44.0316	37.9989	32.1690

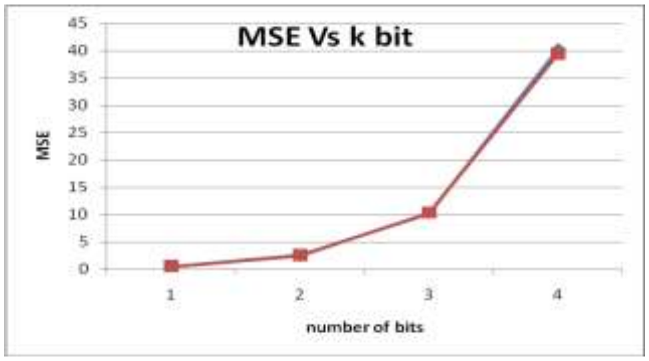


Fig.6. A Graphical Representation of MSE Vs. No.of LSBs used for embedding.

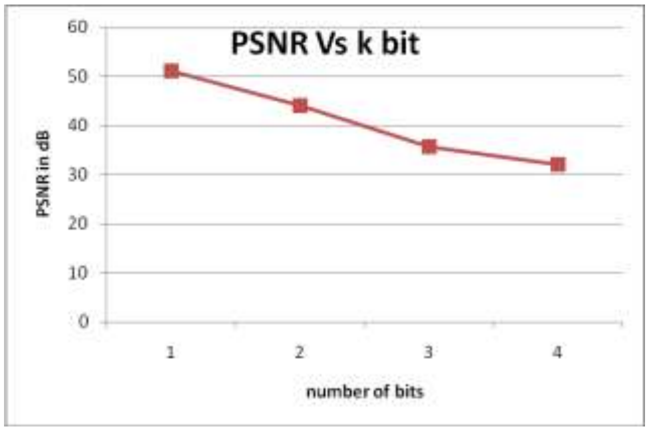


Fig.7. A Graphical Representation of PSNR Vs. number.of LSBs used for embedding.

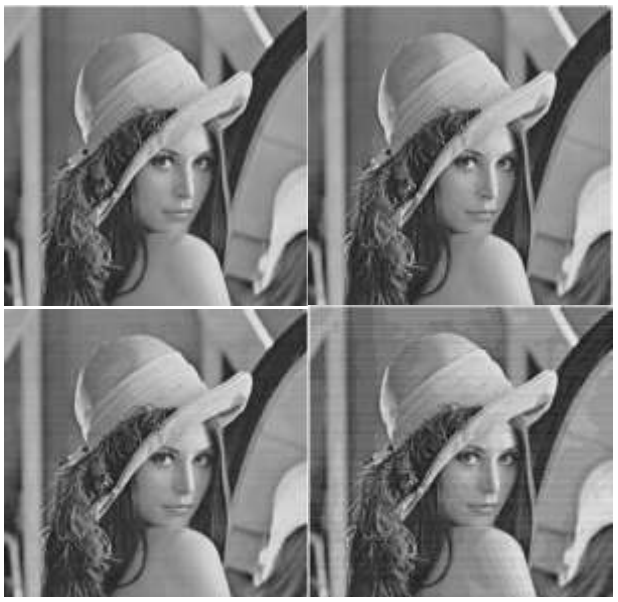


Fig 8 a, b, c and d are Lena output for k=1, 2, 3, and 4 for full embedding capacity.

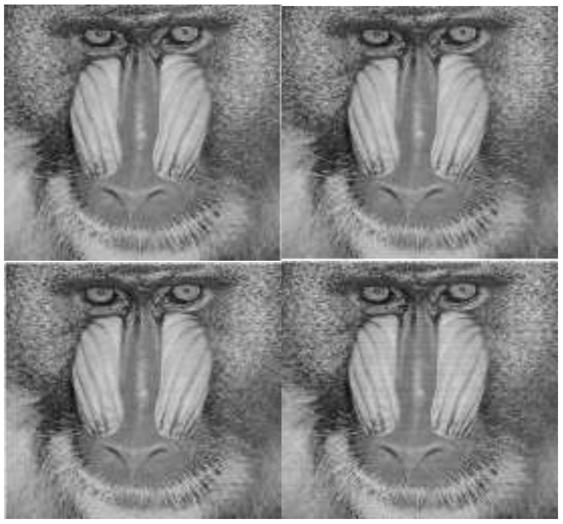


Fig 9 a, b, c and d are Baboon output for k=1, 2, 3, and 4 for full embedding capacity.

6.1 Timing Analysis

The time taken for embedding the secret into the cover image (256x256 pixels) for various choices of 'n' is tabulated in Table II

TABLE II

Timing Analysis for a 256x256 cover image

No. of Bits (n)	1	2	3	4
Average Time Taken by software algorithm in seconds	8.337	8.465	8.522	8.486
Average Time Taken by hardware in seconds *	0.0115	0.0115	0.0115	0.0115

*Based on Classic Timing Analysis of hardware

The timing analysis shows that the hardware can do the secret sharing about 740 times faster than the software architecture. Also, the speed of the software algorithm entirely depends on the run time environment of the process. But, the response time of hardware is always constant for a given size of the cover image. This can be an advantage when we try to interface this module with some other systems, because of predictable time performance.

7. CONCLUSION

The proposed system offers better processing speed than the software based system since the time required for embedding and recovery obviously decrease as they use dedicated hardware for processing. The resultant hardware device could be carried anywhere as it is light-weight and portable. Furthermore this portable hardware device could be extended to support USB and thereby able to send or receive and decode the secret information from MMS stego images. Last but not the least it would consume less power than the existing computer based systems.

8. ACKNOWLEDGMENTS

The authors wish to thank Dr.R.Varadharajan, Professor / ECE and Dr. K.Thenmozhi Associate Dean ECE / SEEE/ SASTRA University for their valuable guidance and moral support.

9. REFERENCES

[1]. Abbas Cheddad, Joan Condell, Kevin Curran, Paul Mc Kevitt (2010), Digital image steganography: Survey and analysis of current methods Signal Processing 90 pp727–752

[2]. R.Amirtharajan , R. Akila, P.Deepikachowdavarapuohn (2010), A Comparative Analysis of Image Steganography International Journal of Computer Applications. **2(3)**.pp 41-47

[3]. W. Bender, D. Gruhl, N. Morimoto, A. Lu (1996), Techniques for data hiding, IBM Syst. J. 35 (3&4) pp 313–336.

[4]. Bruice Schneier, Applied Cryptography Protocols, Algorithm and Source Code in C. Second edition. Wiley India edition 2007

[5]. Chi-Kwong Chan, L.M. Cheng.(2004), "Hiding data in images by simple LSB substitution ", Pattern Recognition 37pp 469 – 474.

[6]. J. Fridrich, M. Goljan, D. Hogea (2003), New methodology for breaking steganographic techniques for JPEGs, in: Proceedings of SPIE: Security and Watermarking of Multimedia Contents, vol. 5020, pp. 143–155.

[7]. S. Katzenbeisser, F.A.P. Petitcolas, Information Hiding Techniques for Steganography and Digital Watermarking, Artech House, Norwood, MA, 2000.

[8]. F.A.P. Petitcolas, R.J. Anderson, M.G. Kuhn(1999), Information hiding—a survey, Proc. IEEE 87 (7) pp 1062–107