RESEARCH ARTICLE OPEN ACCESS

A Hardware FPGA Implementation of Adaptive Stegano Analysis Using LSB Technique

L. V. S Subbaraju*, P. Praveen**, U. Yedukondalu***
* Assistant Professor, (Department of Electronics and Communication Engineering, Aditya Engineering College, Surampalem, Andhra Pradesh)
** Assistant Professor, (Department of Electronics and Communication Engineering, Aditya Engineering College, Surampalem, Andhra Pradesh)
***Associate Professor, (Department of Electronics and Communication Engineering, Aditya Engineering College, Surampalem, Andhra Pradesh)

Abstract:
This paper deals with the detection of hidden bits in the Least Significant Bit (LSB) plane of a natural image. The mean level and the covariance matrix of the image, considered as a quantized Gaussian random matrix, are unknown. An adaptive statistical test is designed such that its probability distribution is always independent of the unknown image parameters, while ensuring a high probability of hidden bits detection. This test is based on the likelihood ratio test except that the unknown parameters are re-placed by estimates based on a local linear regression model. It is shown that this test maximizes the probability of detection as the image size becomes arbitrarily large and the quantization step vanishes. This provides an asymptotic upper-bound for the detection of hidden bits based on the LSB replacement mechanism. System is developed on Xilinx Spartan3 Field Programmable Gate Array (FPGA) device using embedded development kit (EDK) tools from Xilinx.
Keywords- Adaptive detection, information hiding,, FPGA, EDK, Micro Blaze.

I.INTRODUCTION

Many embedded DSP systems make use of a DSP chip utilizing a single processing core with high-bandwidth memory connections to implement DSP algorithms. In this investigation, we developed an alternative approach based on an embedded FPGA system for image processing. Field Programmable Gate Array (FPGA) is widely used in embedded applications such as automotive, communications, industrial automation, motor control, medical imaging etc. FPGA is chosen due to its Reconfigurable ability. Without requiring hardware change-out, the uses of FPGA type Devices expand the product life by updating data stream files. FPGAs have grown to have the capability to hold an entire system on a single chip meanwhile; it allows in-Platform testing and debugging of the system. Furthermore, it offers the opportunity of utilizing hardware/software co-design to develop a high performance system for different applications by incorporating processors (hardware core processor or software core processor), on-chip busses, memory, and hardware accelerators for specific software functions.

Steganography means to hide secret information into innocent data. Digital images are ideal for hiding secret information. An image containing a secret message is called a cover image. First, the difference of the cover image and the Stegano image should be visually unnoticeable.

The embedding itself should draw no extra attention to the Stegano image so that no hackers would try to extract the hidden message illegally. Second, the message hiding method should be reliable. It is impossible for someone to extract the hidden message if she/he does not have a special extracting method and a proper secret key. Third, the maximum length of the secret message that can be hidden should be as long as possible. This paper is organized as follows: Section II briefly reviews message hiding technique. Section III discusses the design flow. Section IV covers different architecture for LSB technique and compares their performance. Section V is the conclusion part.

II.STEGANOSYSTEM

The Stegano system is conceptually similar to the cryptosystem

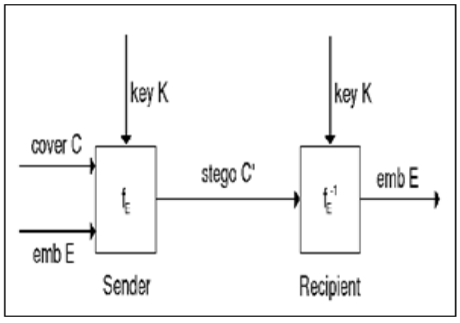

Figure1: Block Diagram

EMB: The message to be embedded. It is anything that can be represented as a bit stream (an image or text).
COVER: Data/Medium in which emb will be embedded.
STEGANO: Modified version of the cover that contains the embedded message.
EMB.KEY: Additional data that is needed for embedding & extracting.
F_E: Stegano graphic function that has cover, emb & key as parameters.

Here is a graphical version of the Stegano system:

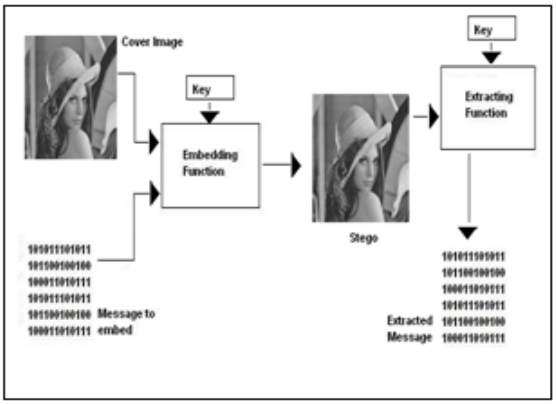

Figure 2: Embedding Process

Steganography refers to the science of "invisible" communication. Unlike cryptography, where the goal is to secure communications from an eavesdropper, Steganography techniques strive to hide the very presence of the message itself from an observer. Although Steganography is an ancient subject, the modern formulation of it is often given in terms of the prisoner's problem [1] where Alice and Bob are two inmates who wish to communicate in order to hatch an escape plan. However, all communication between them is examined by the warden, Wendy, who will put them in solitary confinement at the slightest suspicion of covert communication. Specifically, in the general model for Steganography, we have Alice wishing to send a secret message to Bob. In order to do so, she" embeds" into a cover-object, to obtain the Stegano-object. The Stegano-object is then sent through the public channel. The warden, Wendy, who is free to examine all messages exchanged between Alice and Bob, can be passive or active. A passive warden simply examines the message and tries to determine if it potentially contains a hidden message. If it appears that it does, she then takes appropriate action, else, she lets the message through without any action. An active warden, on the other hand, can alter messages deliberately, even though she may not see any trace of a hidden message, in order to foil any secret communication that can nevertheless be occurring between Alice and Bob. The amount of change the warden is allowed to make depends on the model being used and the cover objects being employed. For example, with images, it would make sense that the warden is allowed to make changes as long as she does not alter significantly the subjective visual quality of a suspected Stegano-image.

A. Information Hiding:

Characters in the ASCII code can be represented using 8 bits. The values pixels of original image can be manipulated slightly without being noticed by visual inspection. This project is based on the premise that the bits of ASCII characters can be included in each one LSB of pixel of original image without resulting in a visible appearance in the so constructed image. If the number was large, 1028 for example, and the LSB was changed from zero to one, the number would be changed from 1028 to 1029, which is a change of only 0.097%. Pixel values in an 8-bit gray-scale image usually range from zero to 255 inclusive for an 8-bit image. If the LSB of 255 were changed to a zero from a one, the result would be 254, a change of 0.39%.

B. Image Compression:

The wavelet transform has proved to be an indispensable tool in data compression due to its ability to decor relates data effectively and efficiently. There are basically two types of compression: Lossy & Lossless. Unlike lossless compression, Lossy image compression can provide acceptable image quality while also providing dramatic reductions in image size. However in applications where quality of reconstructed image from compressed one is must, one need to go for lossless image compression method.

C. Least Significant Bit Insertion (LSB):

The idea behind the LSB algorithm is to insert the bits of the hidden message into the least significant bits of the pixels.

Simplified Example with a 24 bit pixel:
1 pixel:
(00100111 11101001 11001000)
Insert 101:
(00100111 11101000 11001001)
Red green blue

Simplified Example with an 8 bit pixel:
1 pixel:
(00 01 10 11)
White red green blue
Insert 0011:
(00 0<u>0</u> 1<u>1</u> 11)
White white blue blue.

D. Lifting Scheme

Lifting schemes, also known as integer-based wavelets, differ from wavelet transforms in that they can be calculated in-place. Similar to wavelet transformations, lifting schemes break a signal, the image, into its component parts 'trends 'that approximates the original values and 'details' which refers to the noise or high frequency data in the image. A lifting scheme produces integers and this allows the original space to be used to hold the results. Lifting operation requires two steps, one to calculate the trends and another to calculate the details.

Flow chart for Steganography:

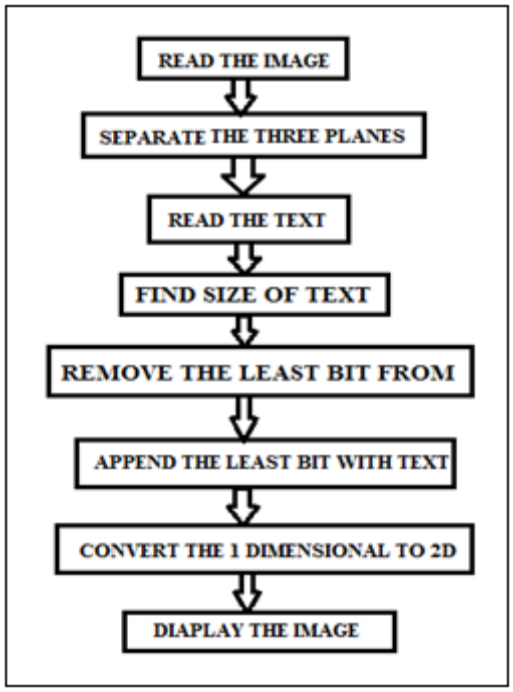

Figure 3: flow chart

Usually 24-bit or 8-bit files are used to store digital images. The former one provides more space for information hiding; however, it can be quite large. The colored representations of the pixels are derived from three primary colors: red, green and blue. 24-bit images use 3 bytes for each pixel, where each primary color is represented by 1 byte. Using 24-bit images each pixel can represent 16,777,216 color values. We can use the lower two bits of these color channels to hide data, then the maximum color change in a pixel could be of 64-color values, but this causes so little change that is undetectable for the human vision system. This simple method is known as Least Significant Bit insertion. Using this method it is possible to embed a significant amount of information with no visible degradation of the cover image. Fig. 2 shows the process.

Several versions of LSB insertion exist. It is possible to use a random number generator initialized with a Stegano-key and its output is combined with the input data, and this is embedded to a cover image. For example in the presence of an active warden it is not enough to embed a message in a known place (or in a known sequence of bits) because the warden is able to modify these bits, even if he can't decide whether there is a secret message or not, o r he can't read it because it is encrypted. The usage of a Stegano-key is important, because the security of a protection system should not be based on the secrecy of the algorithm itself, instead of the choice of a secret key. Fig. 3 shows this process. The LSB inserting usually operates on bitmap images. 'Stegano's for Windows' and 'Webstegano' are LSB inserting software products which are able to embed data (in clear or encrypted format) in a bitmap image. The embedded data cannot be considered as a watermark, because even if a small change occurs in a picture (cropping, lossy compression, and color degradation) the embedded information will be lost – although the change which is occurred during the embedding process is invisible.

III.DESIGN FLOW

To build an embedded system on Xilinx FPGAs, the embedded development kit (EDK) is used to complete the reconfigurable design. In the traditional software design using C/C++ language or hardware design using hardware description languages, the EDK enables the integration of both hardware and software components of an embedded system. . For the hardware side, the design entry from VHDL/Verilog is such as Look-up tables, flip-flops, and block memories. The location and interconnections of these device resources are then placed and routed to meet with the timing Constraints. A downloadable .bit file is created for the whole hardware platform.

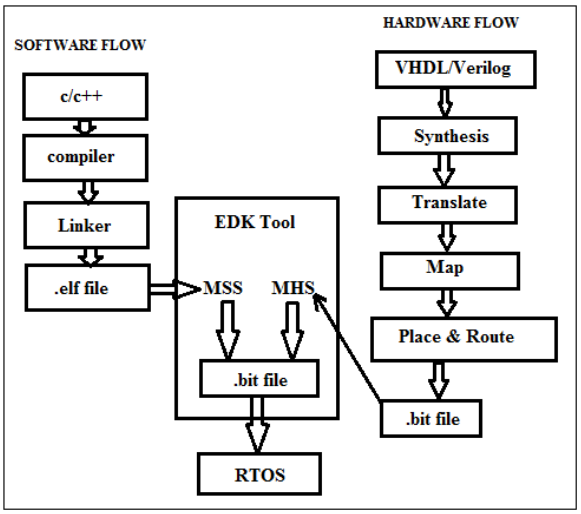

Figure 4: Design flow

The software side follows the standard embedded software flow to compile the source codes

into an executable and linkable file (ELF) format. Meanwhile, a microprocessor software specification (MSS) file and a microprocessor hardware specification (MHS) file are used to define software structure and hardware connection of the system. The EDK uses these files to control the design flow and eventually merge the system into a single downloadable file. The whole design runs on a real-time operating system (RTOS).

IV.EMBEDDING CO−PROCESSOR

There are different ways to include processors inside Xilinx FPGA for System-on-a-Chip (SoC): PowerPC hard processor core, or Xilinx Micro Blaze soft processor core, or user-defined soft processor core in VHDL/Verilog. In this work, The 32-bit Micro Blaze processor is chosen because of the flexibility. The user can tailor the processor with or without advance features, based on the budget of hardware. The advance features include memory management unit, floating processing unit, hardware multiplier, hardware divider, instruction and data cache links etc. The architecture overview of the system is shown in figure 2. It can be seen that there are two different buses (i.e., processor local bus (PLB) and fast simplex link (FSL bus) used in the system [5-6]. PLB follows IBM Core connect bus architecture, which supports high bandwidth master and slave devices, provides up to 128- bit data bus, up to 64-bit address bus and centralized bus Arbitration. It is a type of shared bus. Besides the access overhead, PLB potentially has the risk of hardware/software incoherent due to bus arbitration. On the other hand, FSL supports point-to-point unidirectional communication. A pair of FSL buses (from processor to peripheral and from peripheral to processor) can form a dedicated high speed bus without arbitration mechanism. Xilinx provides C and assembly language support for easy access. Therefore, most of peripherals are connected to the processor through PLB; the DWT coprocessor is connected through FSL instead.

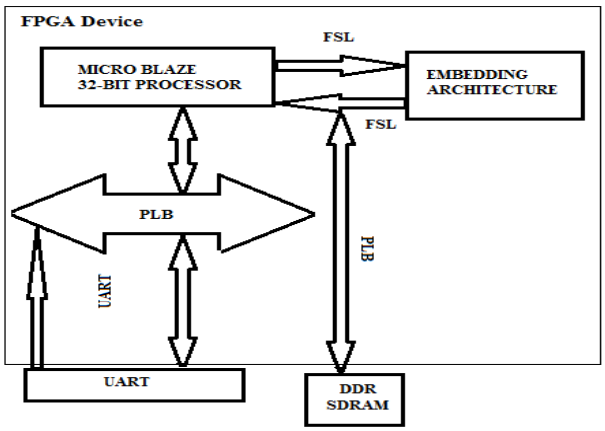

Figure 5: System Overview

The current system offers several methods for distributing the data. These methods are a UART, and VGA, and Ethernet controllers. The UART is used for providing an interface to a host computer, allowing user interaction with the system and facilitating data transfer. The VGA core produces a standalone real-time display. The Ethernet connection allows a convenient way to export the data for use and analysis on other systems. In our work, to validate the DWT coprocessor, an image data stream is formed using VISUAL BASIC, then transmitted from the host computer to FPGA board through UART port.

V.EXPERIMENTAL RESULTS

Experiments are performed on gray level images to verify the proposed method. These images are represented by 8 bits/pixel and size is 128 x 128. Image used for experiments are shown in below figures 6 & 7.

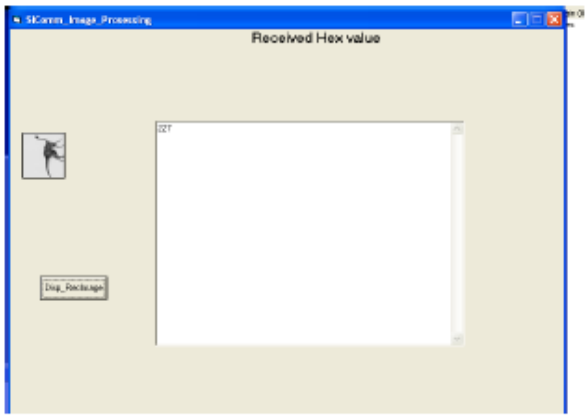

Figure 6: Cover Image In Vb

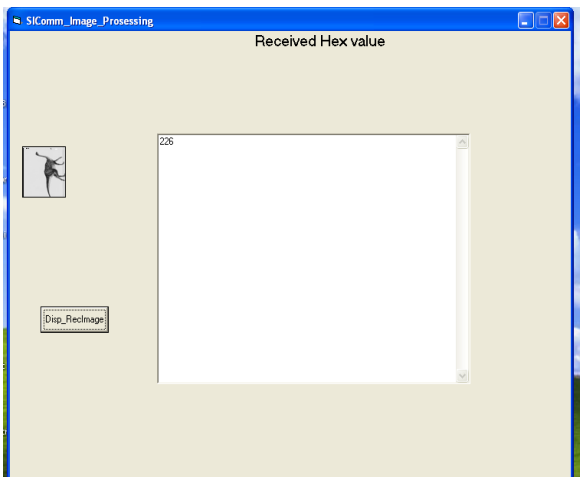

Figure 7: Stegano Image In VB

The Synthesis Report Is Below

```
Selected Device : 3s200tq144-4

 Number of Slices:                     1880  out of   1920    97%
 Number of Slice Flip Flops:           2118  out of   3840    55%
 Number of 4 input LUTs:               2971  out of   3840    77%
    Number used as logic:              2418
    Number used as Shift registers:     297
    Number used as RAMs:                256
 Number of IOs:                          62
 Number of bonded IOBs:                  62  out of     97    63%
    IOB Flip Flops:                      64
 Number of BRAMs:                         4  out of     12    33%
 Number of MULT18X18s:                    3  out of     12    25%
 Number of GCLKs:                         4  out of      8    50%
 Number of DCMs:                          1  out of      4    25%
```

Timing Report

```
Speed Grade: -4

   Minimum period: 15.304ns (Maximum Frequency: 65.342MHz)
   Minimum input arrival time before clock: 6.569ns
   Maximum output required time after clock: 16.181ns
   Maximum combinational path delay: No path found
```

VI. CONCLUSION AND FUTUREWORK

In this Project Image processing system will be implemented on the Xilinx FPGA board using Xilinx EDK Tools. It will digitize and display on visual base window in a real time mode. Through this project, a hardware/software co design method using FPGA will be explored for video and Image process applications.

REFERENCES

[1] H. Sencar, M. Ramkumar, and A. Akansu, Data Hiding Fundamentalsand Applications: Content Security in Digital Multimedia. Elsevier:Academic, 2004.

[2] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker , Digital Water-marking and Steganography. San Francisco, CA: Morgan Kaufmann,2007.

[3] J. Fridrich , Steganography in Digital Media—Principles, Algorithms,and Applications . New York: Cambridge Univ. Press, 2009.

[4] R. Böhme , Advanced Statistical Steganalysis. New York: Springer,2010.

[5] N. Provos and P. Honeyman, "Hide and seek: An introduction to steganography," IEEE Secur. Priv. J., vol. 1, no. 3, pp. 32–44, 2003.

[6] X.-Y. Luo, D.-S. Wang, P. Wang, and F.-L. Liu, "A review on blinddetection for image steganography," Signal Process., vol. 88, no. 9,pp.2138–2157, Sep. 2008.

[7] A. Nissar and A. Mir, "Classification of steganalysis techniques: Astudy," Digit. Signal Process. , vol. 20, no. 6, pp. 1758–1770, 2010.

[8] Li, S.L., Leung, K.C., Cheng, L.M., Chan, C.K. "Data Hiding in Images by Adaptive LSB Substitution Based on the Pixel-Value Differencing" , icicic 2006, IS16-005.

[9] Chang, C.C., Lin, M.H., Hu, Y.-C., 2002, "A fast and secure image hiding scheme based on LSB substitution", Int. Journal of Pattern Recognition. And Artif. Intel. 16 (4), pp 399-416.

[10] Chan, C.K., Cheng, L.M., 2004, "Hiding data in images by simple LSB substitution", Pattern Recognition 37, pp 469–474.

[11] Wang, H., Wang, S, October 2004, "Cyber warfare: Steganography vs. Steganalysis", Communications of the ACM, 47:10.

[12] Sharp, T., 2001, "An implementation of key-based digital signal Steganography", Proc. 4th International Workshop on Information Hiding, Springer LNCS, vol. 2137, pp.13-26.