FPGA Implementation of Lifting DWT based LSB Steganography using Micro Blaze Processor

B.Vasantha Lakshmi*1, B.Vidheya Raju*2
Associate Professor, Department of ECE Pragati Engineering College, Surampalem, India.
M. Tech Student (Embedded Systems) Pragati Engineering College, Surampalem, India.

ABSTRACT: *We present an information hiding technique that utilizes lifting schemes to effectively hide information in images. A successful information hiding should result in the extraction of the hidden data from the image with high degree of data integrity. Current trends favor using digital image files as the cover file to hide another digital file that contains the secret message or information. The Least Significant Bit (LSB) embedding technique suggests that data can be hidden in the least significant bits of the cover image and the human eye would be unable to notice the hidden information in the cover file. This paper explains the LSB Embedding technique with lifting based DWT schemes by using Micro blaze Processor implemented in a FPGA using System C coding.*

Keywords: *Steganography, DWT, LSB, Micro Blaze, FPGA.*

1. INTRODUCTION

Steganography is the art of invisible communication by concealing information inside other information. The term steganography is derived from the Greek and literally means "covered writing". A steganography system consists of three elements: 1) cover-object (which hides the secret message), 2) the secret message and 3) the stego-object (which is the cover object with message embedded inside it.) Given the proliferation of digital images on the internet, and the large redundant bits present in the digital representation of an image, images are the most popular cover objects for steganography.

A digital image is described using a 2-D matrix of the intestines at each grid point (i.e. pixel). Typically, gray images use 8 bits, whereas colored utilizes 24 bits to describe the color model, such as RGB model. The steganography system which uses an image as the cover object is referred to as an image steganography system.

The shift from cryptography to steganography is due to that concealing the image existence as stego-images enable to embed the secret message to cover images. Steganography conceptually implies that the message to be transmitted is not visible to the informal eye. Steganography has been used for thousands of years to transmit data without being intercepted by unwanted viewers. It is an art of hiding information inside other information. The main objective of Steganography is mainly concerned with the protection of contents of the hidden information. Images are ideal for information hiding because of the large amount of redundant space is created in the storing of images. Secret messages are transferred through unknown cover carriers in such a manner that the very existence of

the embedded messages is undetectable. Carriers include images; audio, video, text or any other digitally represented code or transmission. The hidden message may be plaintext, cipher text, integer values or anything that can be represented as a bit stream.

2. THE LSB TECHNIQUE

We have implemented the LSB steganography algorithm in gray scale images to reduce the complexity of the system. It is the process of embedding data within the domain of another data, this data can be text, image, audio, or video contents and the scope of the current paper covers only codes (integer values). The embedded data is invisible to the human eye i.e., it is hidden in such a way that it cannot be retrieved without knowing the extraction algorithm. In this paper we evaluated the technique using gray scale images of size 64*64 in which each pixel value was represented with 8 bit representation.
For Example:
Take the number 300, and its binary equivalent is 100101100 embedded into the least significant bits of pixel values of the Cover image. If we overlay these 9 bits over the LSB of the 9 bytes cover image pixel values, we get the following (where bits in bold have been changed)

10010101 00001100 11001000
10010111 00001110 11001011
10011111 00010000 11001010

After embedding the message into the cover image, the stego image will be obtained, then this stego image will be transformed with DWT transformation technique so that any hacker can't find where the message was embedded. At the receiver end the inverse DWT is applied, after LSB decryption the original image and message will be obtained.

3. PROPOSED METHODOLOGY

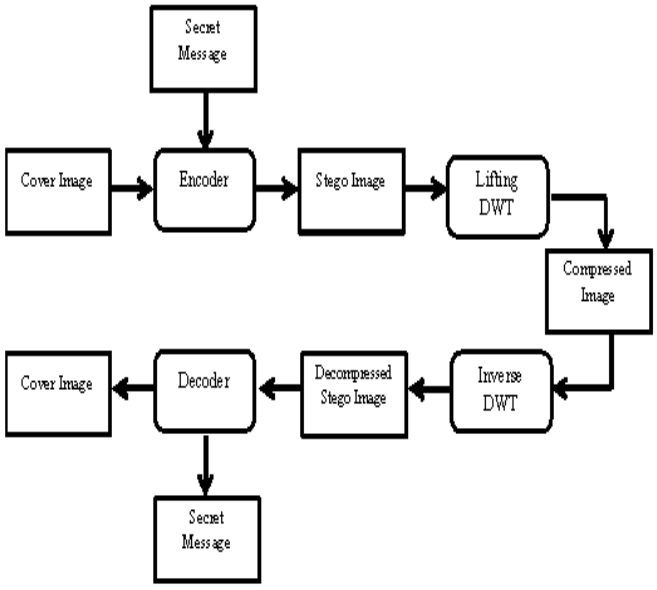

Fig 3.1: Block Diagram of LSB Steganography

3.1. THE LSB ENCRYPTION AND DECRYPTION PROCESS

LSB encryption process consists of two steps namely masking process and generation of stego image.
Masking process is done by replacing LSB of all pixel values with '0'. This is done by performing AND operation with "1111 1110" (254). Now the LSB of all pixels will be zeros.

Now the binary value of the information is placed in the LSB of the pixel values. This is done by identifying the position of 1s and placing them in the LSB of respective pixel value.
For Example, we have n-bit binary message (binary value of the integer), now AND operation is performed between n-bit binary message and '1' in the nth position, rest all

0s. Then OR operation is performed between the output of the above AND operation and the binary values of nth pixel. This operation is repeated for (n-1), (n-2), (n-3)……till it completes LSB.

So for an 8 bit binary message, the above operations start with 8th bit and completes till 1st bit of the message. Now the LSB of pixel values are embedded with message, and now the image is termed as stego image.

LSB decryption process contains extraction of the message from LSB of pixel values.

4. LIFTING BASED DISCRETE WAVELET TRANSFORM

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 i.e. low frequency values and another to calculate the details i.e High frequency values. Trends give the original signal values i.e. low frequency components and details give the noise values i.e. High frequency components. In this lifting scheme based discrete wavelet transformation scheme we used the mathematical calculation method to convert the image into frequency domain. In our project we propose a two level transformation in lifting based DWT schemes to convert image pixel values into frequency domain. The lifting schemes which we have implemented in our project are based on Haar lifting schemes. The Haar lifting calculation of the High and the low frequency values for image pixels are shown below respectively.

High frequency value = odd-even samples

Low frequency values = even + high/2 samples

$$\text{High frequency}: \quad s_{i,2j+1} = s_{i,2j+1} - s_{i,2j}, j = 0,1,\ldots,n/2$$

$$\text{Low frequency}: \quad s_{i,2j} = s_{i,2j} + \frac{s_{i,2j+1}}{2}, j = 0,1,\ldots,n/2$$

Where i, j are the rows and columns of 2D pixel matrix of a stego image.

4.1. 2-D TRANSFORM HEIRARCHY

The 1-D wavelet transform can be extended to a two-dimensional (2-D) wavelet transform using separable wavelet filters. With separable filters the 2-D transform can be computed by applying a 1-D transform to all the rows of the input, and then repeating on all of the columns.

`

LL	HL
LH	HH

Fig 4.1.1: Sub band Labeling Scheme for a one level, 2-D Wavelet Transform

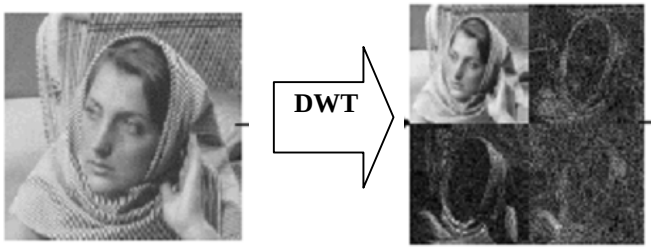

Fig 4.1.2: Pictorial representation of Sub band Labeling Scheme for a one level, 2-D Wavelet Transform

4.2. LIFTING BASED DWT SCHEMES

It is composed of three basic operation stages:

- **Splitting:** where the signal is split into even and odd pixels
- **Predicting:** Even samples are added by a prediction factor derived from odd and even pixels to get low frequency values
- **Updating :** The detailed coefficients computed by the predict step are multiplied by the update factors and then the results are subtracted to the even samples to get the high frequency values
- **Merging/Combining** : the reverse process of DWT has done to merge all the LL,LH,HL,HH to reconstruct the original image

In this procedure Image pixels values are divided into even samples and odd samples then for getting high frequency value= odd-even samples and for low frequency values =even + high/2 samples then this procedure is repeated for two phases . The first phase is known as Column Filter that means performing the DWT calculations on columns to get Low and High frequency values and second phase is known as Row filter. That means we are going to apply DWT calculation on rows in order to get the LL, LH, HL, HH.

The Inverse DWT is also fallows the same process in reverse manner to construct original image pixel values from LL, LH, HL, HH values.it is a process converting frequency domain to image pixel values.

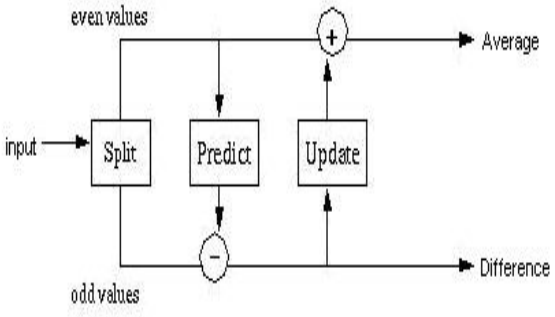

Fig 4.2.1: DWT based Lifting Scheme Process

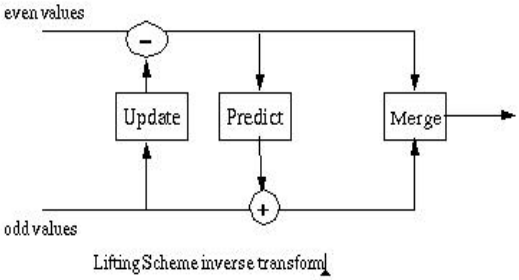

Fig 4.2.2: Inverse DWT based Lifting Scheme Process Flow

5. EXPERIMENTAL SETUP AND RESULTS

5.1. XILINX PLATFORM STUDIO

The Xilinx Platform Studio (XPS) is the development environment or GUI used for designing the hardware portion of your embedded processor system. B. Embedded Development Kit Xilinx Embedded

Development Kit (EDK) is an integrated software tool suite for developing embedded systems with Xilinx Micro Blaze and PowerPC CPUs. EDK includes a variety of tools and applications to assist the designer to develop an embedded system right from the hardware creation to final implementation of the system on an FPGA. System design consists of the creation of the hardware and software components of the embedded processor system and the creation of a verification component is optional. A typical embedded system design project involves: hardware platform creation, hardware platform verification (simulation), software platform creation, software application creation, and software verification. Base System Builder is the wizard that is used to automatically generate a hardware platform according to the user specifications that is defined by the MHS (Microprocessor Hardware Specification) file. The MHS file defines the system architecture, peripherals and embedded processors]. The Platform Generation tool creates the hardware platform using the MHS file as input. The software platform is defined by MSS (Microprocessor Software Specification) file which defines driver and library customization parameters for peripherals, processor customization parameters, standard 110 devices, interrupt handler routines, and other software related routines. The MSS file is an input to the Library Generator tool for customization of drivers, libraries and interrupts handlers.

The creation of the verification platform is optional and is based on the hardware platform. The MHS file is taken as an input by the Simgen tool to create simulation files for a specific simulator. Three types of simulation models can be generated by the Simgen tool: behavioral, structural and timing models. Some other useful tools available in EDK are Platform Studio which provides the GUI for creating the MHS and

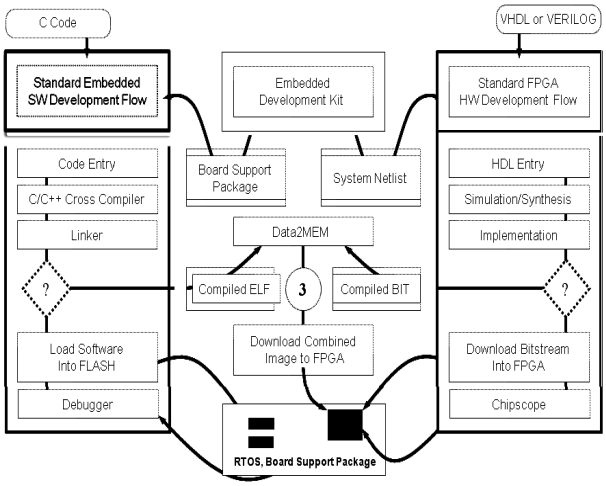

Fig 5.1.1: Embedded Development Kit Design Flow

MSS files. Create / Import IP Wizard which allows the creation of the designer's own peripheral and import them into EDK projects. Platform Generator customizes and generates the processor system in the form of hardware net lists. Library Generator tool configures libraries, device drivers, file systems and interrupt handlers for embedded processor system. Bit stream initializer tool initializes the instruction memory of processors on the FPGA shown in figure2. GNU Compiler tools are used for compiling and linking application executables for each processor in the system [6]. There are two options available for debugging the application created using EDK namely: Xilinx Microprocessor Debug (XMD) for debugging the application software using a

Microprocessor Debug Module (MDM) in the embedded processor system, and Software Debugger that invokes the software debugger corresponding to the compiler being used for the processor. C. Software Development Kit Xilinx Platform Studio Software Development Kit (SDK) is an integrated development environment, complimentary to XPS, that is used for C/C++ embedded software application creation and verification. SDK is built on the Eclipse opensource framework. Soft Development Kit (SDK) is a suite of tools that enables you to design a software application for selected Soft IP Cores in the Xilinx Embedded Development Kit (EDK).The software application can be written in a "C or C++" then the complete embedded processor system for user application will be completed, else debug & download the bit file into FPGA. Then FPGA behaves like processor implemented on it in a Xilinx Field Programmable Gate Array (FPGA) device.

5.2. FEATURES OF MICRO BLAZE PROCESSOR

The Micro Blaze soft core processor is highly configurable and the feature set of the processor includes:

- Five Stage Pipeline
- Thirty-two 32-bit general purpose registers
- 32-bit instruction word with three operands and two addressing modes, 32-bit address bus and Single issue pipeline

In addition to these fixed features, the Micro Blaze processor is parameterized to allow selective enabling of additional functionality. Xilinx recommends that all new designs use the latest preferred version of the Micro Blaze processor.

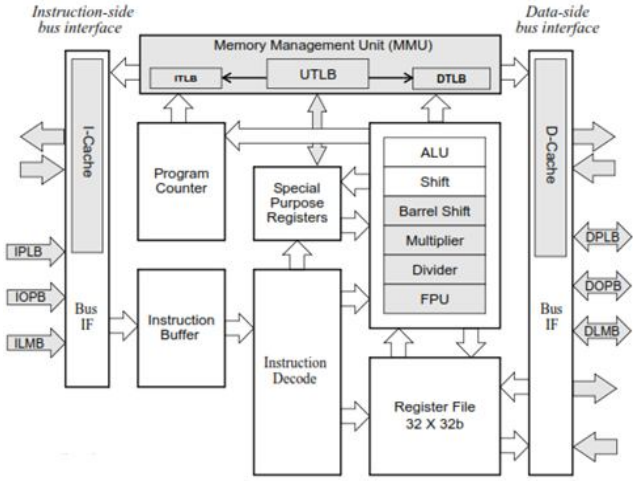

Fig 5.2.1: Micro Blaze processor

The Algorithm is implemented in Micro blaze Processor and the results are furnished below

5.3. RESULTS:

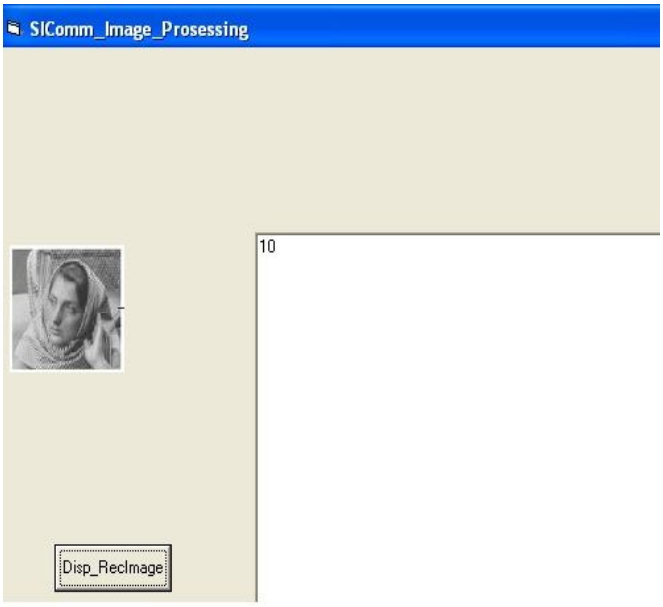

Fig 5.3.1: Input Image reading through Visual Basic (VB)

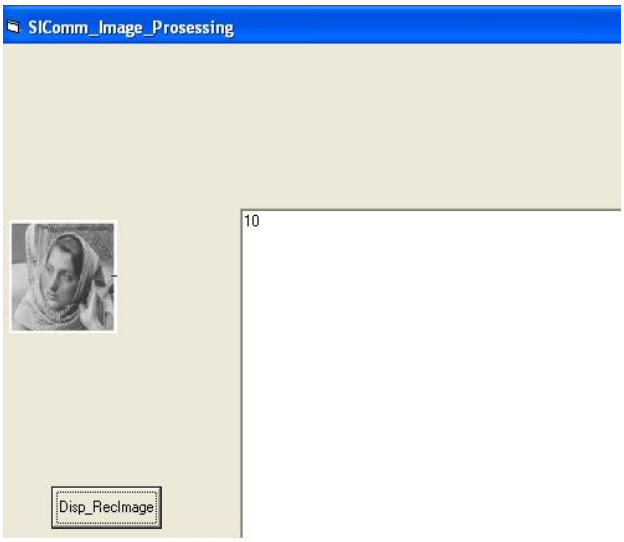

Fig 5.3.2: Stego Image through Visual Basic (VB)

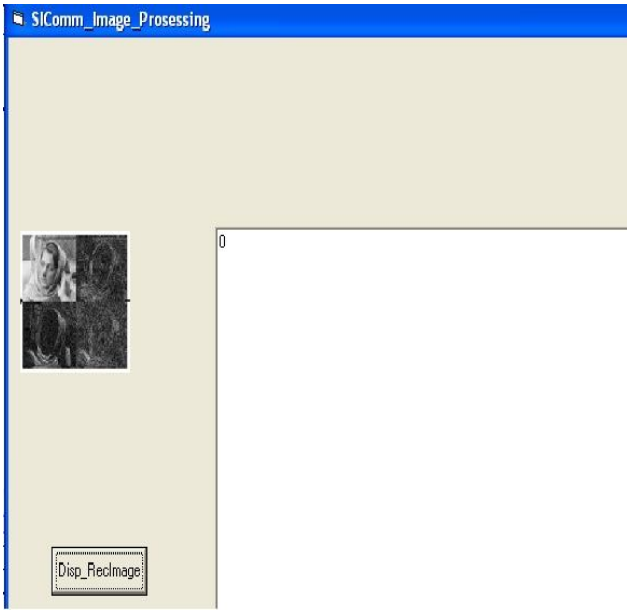

Fig 5.3.3: Discrete Wavelet Image through Visual Basic (VB)

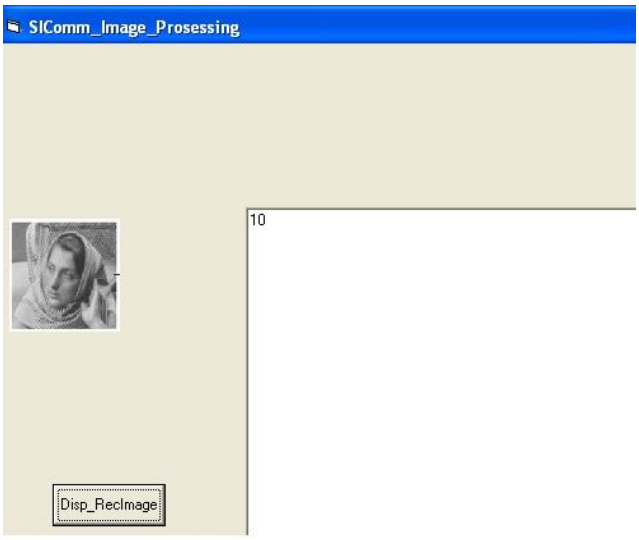

Fig 5.3.4: Original Image Reconstructed through Visual Basic (VB)

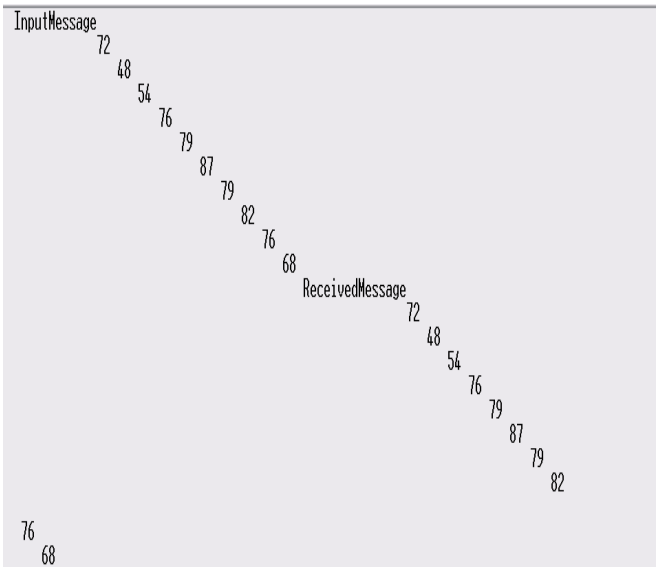

Fig 5.3.5: Input and Output Message displayed on HyperTerminal

```
Timing Summary:
---------------
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

Timing Detail:
--------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default period analysis for Clock 'sys_clk_pin'
  Clock period: 13.098ns (frequency: 76.348MHz)
  Total number of paths / destination ports: 111905 / 6937
-------------------------------------------------------------------------
```

```
Device utilization summary:
---------------------------

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%
```

Fig 5.3.6: Synthesis Report

Fig 5.3.7: Timing Analysis

6. CONCLUSION:

In this paper we have presented a new method of LSB Steganography using lifting DWT Process. This process was implemented by developing Micro Blaze processor in FPGA.

Future work can be extended to RGB or color image processing and can be extended to video processing level also.

REFERENCES:

[1] Pfitzmann Birgit. Information Hiding Terminology, First International Workshop, Cambridge, UK, Proceedings, Computer Science, 1174. pp. 347-350, May–June.

[2] Andreas Westfeld and Andreas Pfitzmann, Attacks on Steganographic Systems, Third International Workshop, IH'99Germany Dresden , Proceeding Computer Science October s, 1768. pp. 61- 76, 1999.

[3] Moerland, T., “Steganography and Steganalysis”, *Leiden Institute of Advanced Computing Science*,Silman, J., “Steganography and Steganalysis: An Overview”, *SANS Institute*, 2001 Jamil, T., “Steganography: The art of hiding information is plain sight”, *IEEE Potentials*, 18:01, 1999

[4] Wei Zhang,Zhe Jiang,Zhiyu Gao, and Yanyan Liu,”An efficient VLSI architecture for Lifting based discrete wavelet

transform,"IEEE Trans.Circuits and systems,vol.59,NO.3,pp. 158-162,Mar.2012.

[5] G. Xing, J. Li, and Y. Q. Zhang, "Arbitrarily shaped videoobject coding by wavelet," IEEE Trans. Circuits Syst. Video Technol., vol. 11, no. 10,pp. 1135–1139, Oct. 2001.

[6] S. C. B. Lo, H. Li, and M. T. Freedman, "Optimization of wavelet decomposition for image compression and feature preservation," IEEE Trans.Med. Imag., vol. 22, no. 9, pp. 1141–1151, Sep. 2003.

[7] K. K. Parhi and T. Nishitani, "VLSI architecture for discrete wavelet transforms," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 1, no. 2, pp. 191–202, Jun. 1993.

[8] X. X. Qin and M. Wang, "A review on detection of LSB matching steganography," *Inf. Technol. J.*, vol. 9, pp. 1725–1738, 2010.

[9] A. D. Ker, "Locating steganographic payload via WS residuals," in*ACM Proc. 10th Multimed. Secur. Workshop*, 2008, pp. 27–31.

[10] A. D. Ker, "A general framework for the structural steganalysis of LSBreplacement," in *Proc. 7th Inf. Hiding Workshop, ser. Springer LNCS*, 2005, vol. 3727, pp. 296–311.

[11] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB steganography in color and grayscale images," *IEEE Multimedia*, vol. 8, no. 4, pp. 22–28, 2001.

[12] S. Dumitrescu, X.Wu, and Z.Wang, "Detection of LSB steganography via sample pair analysis," *IEEE Trans. Signal Process.*, vol. 51, pp.1995–2007, Jun. 2003.

[13] A. Ker, "Steganalysis of LSB matching in grayscale images," *Signal Process. Lett.*, vol. 12, no. 6, pp. 441–444, Jun. 2005.

[14] A. D. Ker, "A fusion of maximum likelihood and structural steganalysis," in *Proc. 9th Inf. Hiding Workshop, ser. Springer LNCS*, 2007, vol. 4567, pp. 204–219.

[15] K. Lee, A. Westfeld, and S. Lee, "Generalised category attack—Improving histogram-based attack on JPEG LSB embedding," *Inf.Hiding'07*, pp. 378–391, 2007.

[16] J. Fridrich and M. Goljan, "On estimation of secret message length in LSB steganography in spatial domain," in *Secur., Steganogr. Watermarkingof Multimed. Contents VI, ser. Proc. SPIE*, 2004, vol. 5306,pp. 23–34