Nile University

FPGA Research Internship Report

LSB-Steganography based on Eye Sensitivity and edge detection FPGA Implementation

Team members:

Sara Walid Ahmed 19100815

Hager Sayed Soliman 19100797

Ahmed Mohamed Gaber 19100683

Supervised by Dr Madian and RA Alaa AbdlRhman

Contents

Table of Figures

FPGA:

What is FPGA ?

The Field Programmable Gate Array (FPGA) is an integrated circuit that consists of internal hardware blocks with user-programmable interconnects to customize operation for a specific application. A basic FPGA architecture (***Figure 1***) consists of thousands of fundamental elements called configurable logic blocks (CLBs) surrounded by a system of programmable interconnects, called a fabric, that routes signals between CLBs. Input/output (I/O) blocks interface between the FPGA and external devices.

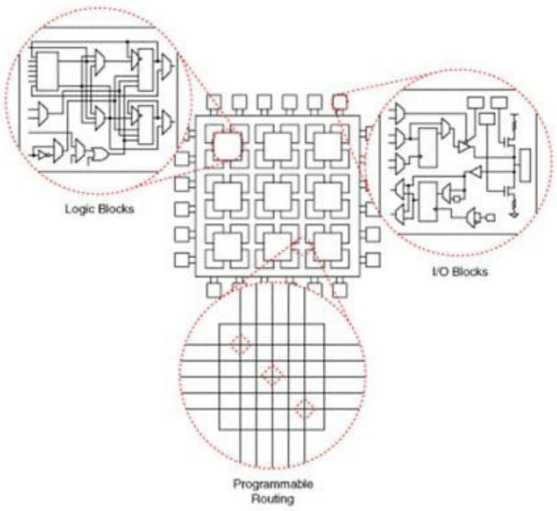

Figure 1The fundamental FPGA architecture

An individual CLB (***Figure 2***) is made up of several logic blocks. A lookup table (LUT) is a characteristic feature of an FPGA. An LUT stores a predefined list of logic outputs for any combination of inputs: LUTs with four to six input bits are widely used. Standard logic functions such as multiplexers (mux), full adders (FAs) and flip-flops are also common.

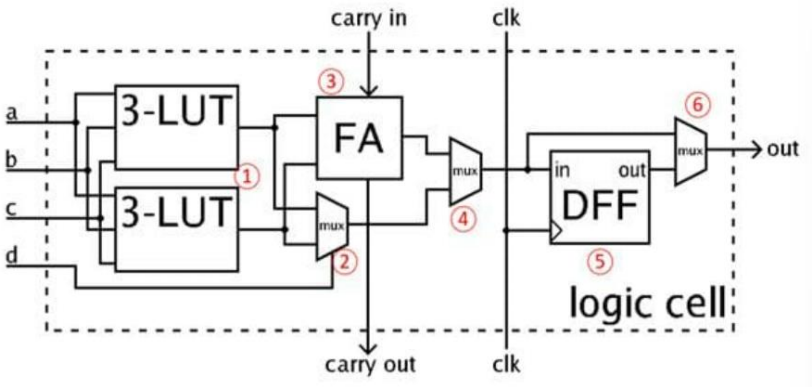

Figure 2 A simplified CLB: The four-input LUT is formed from two three-input units

Choosing Between FPGA and Microcontroller

Generally, processors including microcontrollers are more suitable for routine control of circuits, such as using a switch to turn on and off a device. FPGAs are suitable for applications that are more customized and require higher processing power or speeds. For example, processing high-resolution video data would be best with a FPGA platform. Embedded engineers often use microcontrollers in embedded devices as they are easier to program, easier to debug and design, and are often lower cost to implement. However, they lack on the flexibility front. FPGAs that can allow for reprogramming of hardware/firmware, microcontrollers only allow for reprogramming of firmware, which greatly limits its options for any modifications.

Another benefit of FPGAs is their parallel processing ability, or parallel execution of identical operations. Because of the hundreds or thousands of CLBs processing synchronously, applications including image processing or artificial intelligence are more feasible. Alternatively, microcontrollers perform sequential processing, meaning it reads and processes each line of the program one after the other, which is less powerful in comparison.

FPGA Families:

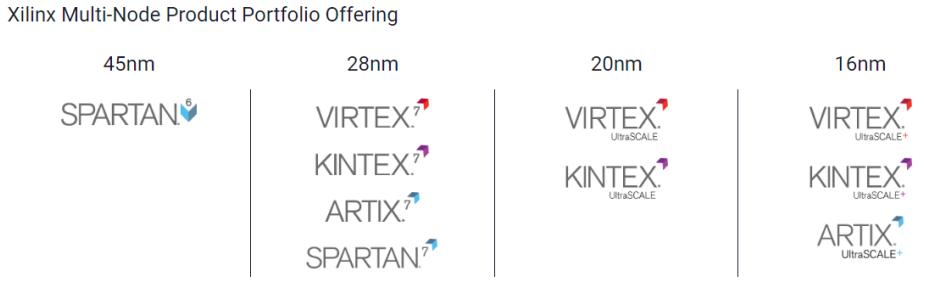

Figure 3 FPGA Families

Figure 4 Genesys2 kintex7

Figure 5 Nexys4 Artix7

Verilog

Verilog Basics

1. Verilog Module:

   How to build a module

   - Input
   - Output
   - Output reg

2. Verilog Coding Style
   - Structural modeling
   - Dataflow
   - Behavioral modeling
3. Building Full adder, Half adder and Muxs
4. Data Types
   - Reg
   - Wire
   - Wire tri
   - Integer
   - Parameter
   - Local parameter
5. Numbers representation
   - Base formats hex, bin, decimal, signed and unsigned
6. Data operators
   - Logical
   - Bit-wise
   - Relational
   - Arithmetic
   - Reduction
   - Shift
   - Miscellaneous

7. Continuous assignment statement
8. Procedural assignment block
    - Initial block
    - Always block
    - Begin/end
    - Fork/join
9. Testbench
10. Verilog condition
    - Behavioral statements
    - If-else
    - Case statement

11. Verilog loop
    - While loop
    - For loop
    - Wait statement
    - Forever statement
    - Repeat statement
12. FSM
    - FSM is used to model a system that transits among some states.
    - In MOORE machines the output only depends on the state.
    - In MEALY machines the output depends on the state and the input.
13. Practicing Verilog (Solving problems)
    - https://hdlbits.01xz.net/wiki/Problem_sets

Vivado Soft-Ware

- Design sources
- Constraints files
- Simulation and Testbench
- IP Catalog
- Clocking Wizard
- Configure differential clock (Genesys2)
- Block Memory Generator (Ram/Rom)

- Run Simulation
- Run synthesis
- Run implementation
- Generating Bit Stream
- Uploading Files To FPGA

FPGA Projects using Vivado Soft-Ware

Linking between leds and Switches

Linking between leds and push-buttons

7-segment Display using Nexys4

2 7-Segmet Display by FSM using Nexys4

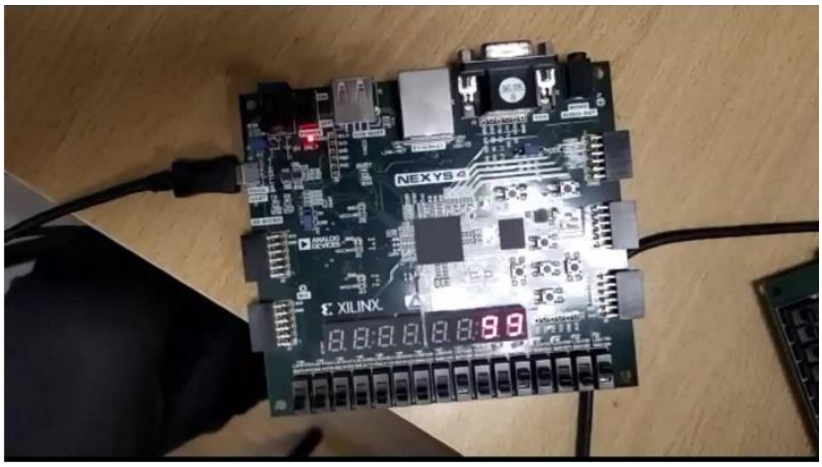

Figure 6 Two 7 Segments Project

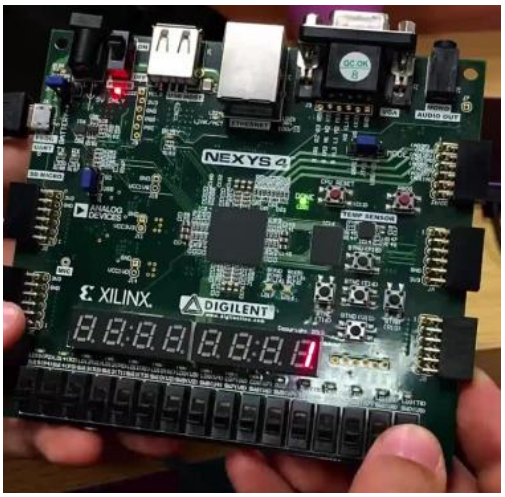

Figure 7 one 7 segments

Steganography

What is Steganography?

Is a method of hiding secret data, by embedding it into an audio, video, image, or text file. It is one of the methods employed to protect secret or sensitive data from malicious attacks; the secret data is then extracted at its destination while the recipient of the image must be aware of the same algorithm to know which pixels, he or she must select to extract the message.

The difference between cryptography and steganography

Cryptography and steganography are both methods used to hide or protect secret data. However, they differ in the respect that cryptography makes the data unreadable, or hides the meaning of the data, while steganography hides the existence of the data.

Image Steganography

cover image: selected image to put data in

stego image: image obtained after steganography

Steganalysis

Detection of steganography is called Steganalysis

Steganography different types:

1. Text steganography
2. Image steganography
3. Audio steganography
4. Video steganography

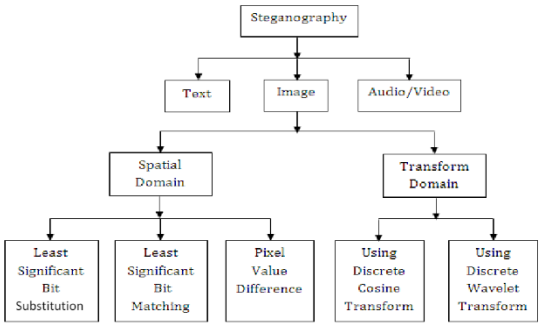

Figure 8 Steganography types

There are two different methods for image steganography:

Spatial methods

There are many methods in spatial Domain, Least significant bit (LSB) method is a common, simple approach to embedding information in a cover file. Steganography using LSB is a simple technique for data hiding inside cover image. The cover image pixels low value bits are replaced by secret message bits in spatial domain

Transform method

transform domain steganography is one of the techniques used for hidden exchange of information in frequency domain and it can be defined as the study of invisible communication that deals with the ways of hiding the existence of the communicated message.

Steganography Disadvantages**:**

1. Size of the image: A Steganographic image has a huge storage size when compared to regular image of the same dimensions, if the original image storage size would be few KBs, the Steganographic image could be several MBs in size. This again varies with the resolution and type of image used.
2. Noise in image: A Steganographic image has noise when compared to a regular image. This is the reason why initially little noise is added to the cover image, so that the Steganographic image doesn't appear very noisy when compared to the original cover image.
3. Embedded in spatial domain, losses can occur such as when the image is cropped.

Steganography based on Eye Sensitivity

Vision and RGB colors sensitivity

The electromagnetic spectrum includes a wide range of wavelengths that start from 0.001 nm for the Gamma rays up to 100 feet for the Radio waves. The visual spectrum is very limited in this long spectrum and lies between 380 nm and 780 nm. The typical human eye is sensitive to the visual spectrum of the whole electromagnetic spectrum which is considered as very limited frequencies. The visual wavelengths represent the colors that can be seen and differentiated against each other. The narrow band of frequencies which include visual colors for the human eye starts from Red and ends with Violet. Not all the colors appear on this narrow spectrum but only the pure colors and other colors appear as a mix between different pure colors.

light pass through the pupil inside the human eye to strike the sensing mem-brane at the back of the eye called retina. The retina is a very sensitive part of the eye and consists of light sensing cells call rods and cones. The rods are sensitive to the light intensity, while the cones are sensitive to different colors. The cones are separated into three types based on their color sensitivity. Each one is sensitive to its color wavelength spectrum.

There are Blue color sensing cones, Red color sensing cones and the Green color sensing cones. The individual color-receptor mechanisms Contribute to the total foveal sensitivity of a single human subject. Regarding the sensitivity of the foveal to the color channels, if given a maximum value of 1.0, so, the Blue share is 0.053, Green is 0.575, while Red is 0.542.

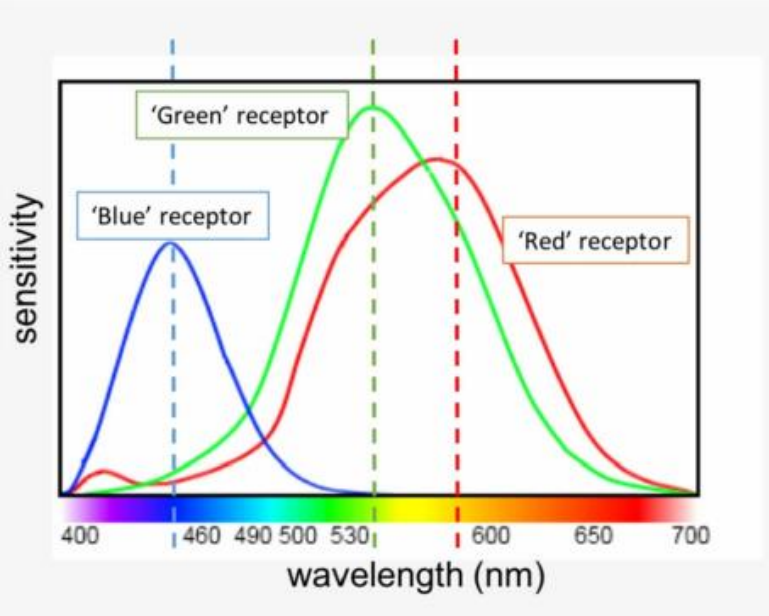

Figure 9 Spectral sensitivity of the RGB color cones

Edge detection

Edge detection is a technique of image processing used to identify points in a digital image with discontinuities, simply to say, sharp changes in the image brightness. These points where the image brightness varies sharply are called the edges (or boundaries) of the image.

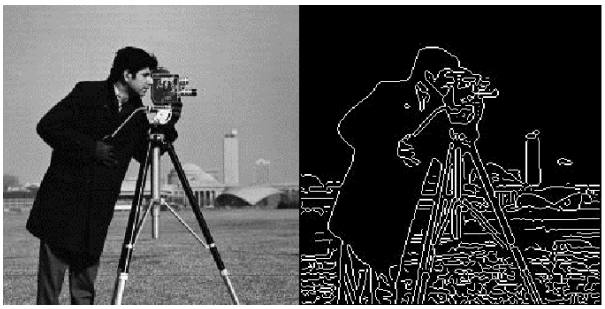

Figure 10 edge detection

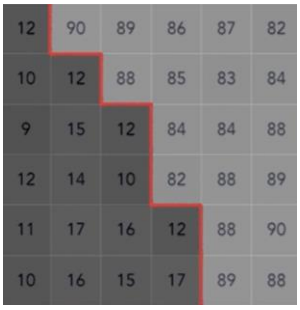

Figure 11 edge detection matrix

Methods of Edge Detection

There are various methods, and the following are some of the most used methods-

- Prewitt edge detection
- Sobel edge detection
- Laplacian edge detection
- Canny edge detection

Prewitt Edge Detection

This method is a commonly used edge detector mostly to detect the horizontal and vertical edges in images. The following are the Prewitt edge detection filters.

Laplacian Edge Detection

The Laplacian edge detectors vary from the previously discussed edge detectors. This method uses only one filter (also called a kernel). In a single pass, Laplacian detection performs second-order derivatives and hence are sensitive to noise. To avoid this sensitivity to noise, before applying this method, Gaussian smoothing is performed on the image.

Canny Edge Detection

This is the most used highly effective and complex compared to many other methods. It is a multi-stage algorithm used to detect/identify a wide range of edges.

Steps of edge detection

1. Gray scale
2. Gaussian filter (blur)
3. Sobel Operator
4. Hysteresis thresholding
5. Angle indication

Sobel Edge Detection Using Linear Shift Buffer and Zero Embedding Technique

Edge Detection is when we use matrix math to calculate areas of different intensities of an image. Areas where there are extreme differences in the intensities of the pixel usually indicate an edge of an object. After finding all of the large differences in intensities in a picture, we have discovered all of the edges in the picture. Sobel Edge detection is a widely used algorithm of edge detection in image processing. Along with Canny and Prewitt, Sobel is one of the most popular edge detection algorithms used in today's technology.

When using Sobel Edge Detection, the image is processed in the X and Y directions separately first, and then combined together to form a new image which represents the sum of the X and Y edges of the image.

When using a Sobel Edge Detector, it is first best to convert the image from an RGB scale to a Grayscale image. Then from there, we will use what is called kernel convolution. A kernel is a 3 x 3 matrix consisting of differently (or symmetrically) weighted indexes. This will represent the filter that we will be implementing for an edge detection.

When we want to scan across the X direction of an image for example, we will want to use the following X Direction Kernel to scan for large changes in the gradient. Similarly, when we want to scan across the Y direction of an image, we could also use the following Y Direction Kernel to scan for large gradients as well.

-1	0	+1
-2	0	+2
-1	0	+1

+1	+2	+1
0	0	0
-1	-2	-1

Figure 12 Sobel matrixes

Linear shift buffer technique is a delay line shift register to limit the filter kernel from reading the same pixel value multiple times from an external memory, this solution reads a pixel value from the external memory. A new pixel value is read every clock cycle, it enters the first shift register (bottom row of the filter kernel). After m clock cycles this pixel value is shifted into the pixel buffer. The pixel buffer is W − m cells wide. so this method is used to pass though each pixel of image to detect if there is an edge or not. A 3*3 sobel matrix is multiplied with 9 pixels of cover image but the first pixel is detected after the first row and 2 columns as shown in (figure 13 b) and the pixels in grey is embedding as zeros so the first pixel is detected after 258 cycles if the cover image 256*256 pixels.

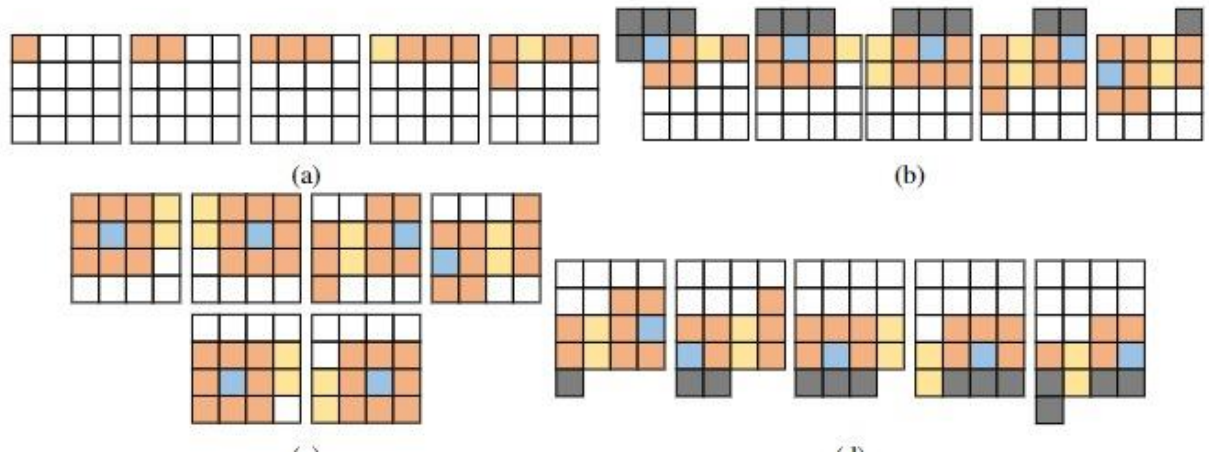

Figure 13 zero embedding and linear buffering

Implementation

Image steganography implementation using python (opencv2).

The proposed approach we used presents a new state of the art technique which uses the human eye color sensitivity and edge detection rules to embed a secret message inside a cover image with low corruption to the resultant stego image. Using the most common and simple technique for data hiding (LSB) to make the implementation apart from being computationally expensive. We considered the computational complexity in our application as much as possible for better performance.

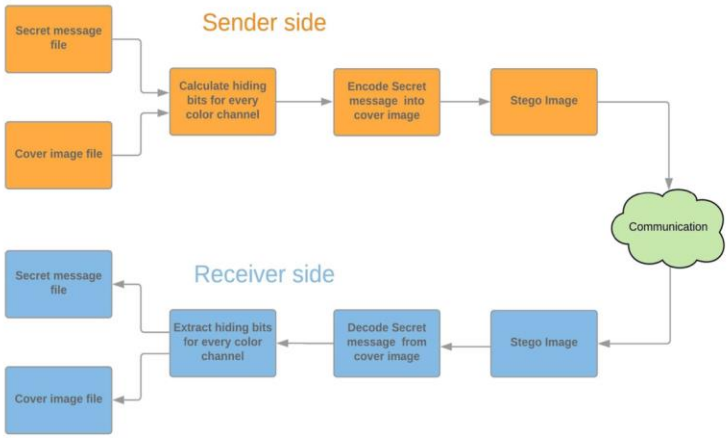

Figure 14 image Steganography process

Algorithm:

- Our Approach Is Built on First edge detection then the Sensitivity of the Eye to the Different Color Channels. Using These two points, We Can Embed Data in The Blue Channel Then the Red One and Finally the Green Channel if there is an edge.
- Depending on the size of image to the secret message, the hiding bit for each pixel is calculated which could be from 1 to 6 bits each pixel so

If Number of hiding bits%3 (modulus) equal 0 then all color channels hiding bits are equal to Number of hiding bits/3. 2. Else if Number of hiding bits%3 (modulus) equal 1 then that color channel hiding bits are equal to Number of hiding bits/3, for the Red and Green channels and equal to (Number of hiding bits/3)+1 for the Blue channel. 3. Else when Number of hiding bits%3

(modulus) equal 2 then the color channel hiding bits are equal to Number of hiding bits/3, for the Green channel and equal to (Number of hiding bits/3)+1 for the Red and Blue channels.

- So, Start with The Less Sensitive Color and End with The Most Sensitive One for Achieving the Perfect Photo.
- Steganography Using LSB Is a Simple Technique for Data Hiding Inside Cover Image. The Cover Image Pixels Low Value Bits Are Replaced by Secret Message Bits In Spatial Domain.

Figure 15 eye sensitivity embedding technique

Results:

Embedding a large audio in an image cause significant change in the stego image.

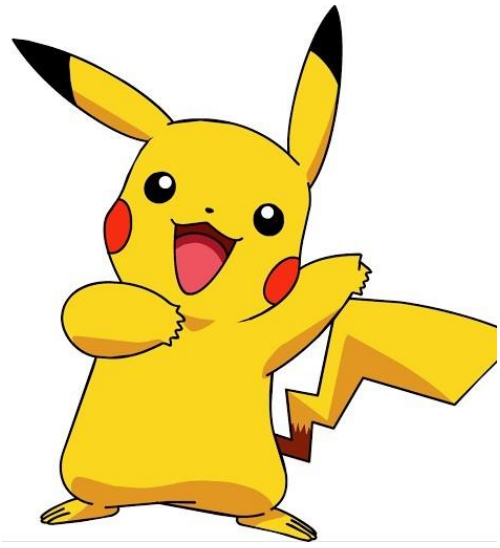

Figure 16 cover image

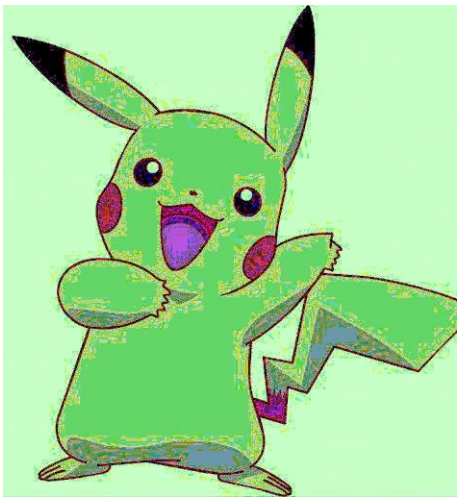

Figure 17 Stego- image

Calculating PSNR

Is the ratio between the maximum possible power of an image and the power of corrupting noise that affects the quality of its representation. To estimate the PSNR of an image, it is necessary to compare that image to an ideal clean image with the maximum possible power.

PSNR Value =29 %

RESULTS AFTER EMBEDDING TEXT MESSAGE IN COVER IMAGE AND PSNR =65

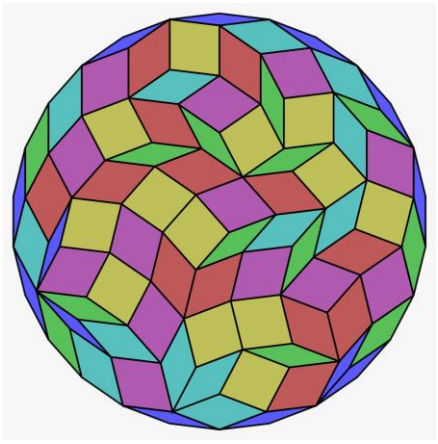

Figure 18 cover image

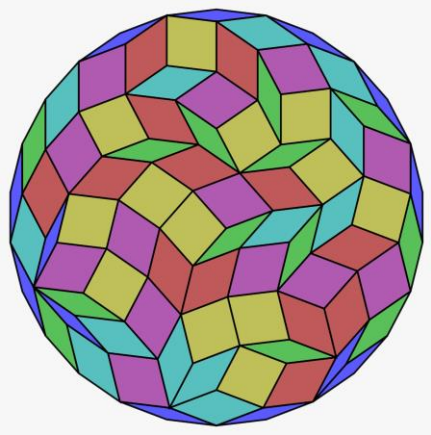

Figure 19 stego image

Implementation on FPGA

To apply our method steganography on FPGA we need to have a controller to control the address of memory then we need to pass it on the encoder to embed message in it using color eye sensitivity approach then pass it to the decoder to get the message. We add the part of edge detection to embed our message in the edges because eye is not sensitive to edges.

We made two passes for the cover image, First to RGBTOGREY Module which convert the RGB pixel to grey scale then the output is added to buffer module. After 266 cycles there kernel is ready for multiplication process, the output result from x-axis kernel and y-axis kernel are added to sqrt module to calculate the threshold (pixels average to detect edges) and the flag of starting bits from the second cover image memory path is triggered.

Finally, the average module compares between squared pixels and its average (summation of them over count of these numbers). If the input squared pixels is bigger than the average so there is an edge detected in the pixel and the flag of edge detected is triggered after that the memory of Date (secret message start sending addresses) then the encoding and decoding process starting depends on the number of bits embedded per each pixel.

1. Block Memory Generator: We used block memory generator from IP catalog to store our image and message.
2. **Controller Module**: it consists of 3 counter counter_data , counter_img_buffer and counter_img_encoder which used to control the memory addresses of the image

depending on the number of bits that is embedded and the two flags from average module.

3. **Encoder Module**: used to embed message in the cover image based on the number of bits of the message (switch cases). It starts after edge_ detected flag with 2 clock cycles as memory start sending data after two clock cycles.
4. **Decoder Module**: used to generate the original message as output from the cover image depends on the flag coming from encoder.
5. **Buffer Module**: Is a delay line shift register to limit the filter kernel from reading the same pixel value multiple times from an external memory.
6. **Kernel Module**: Start the process of multiplication by sobel filter after buffer flag is triggered.
7. **Sqrt Module**: module used to square root the values of Sobel edge detection to get the average value of the used in the threshold
8. **Threshold(average):** used to detect the edges using a dynamic threshold.

Final Results

The Cover image (256*256 pixel)

Figure 20 cover image

Stego image MATLAB Output

Figure 21 stego image

The original message (32*32 pixel)

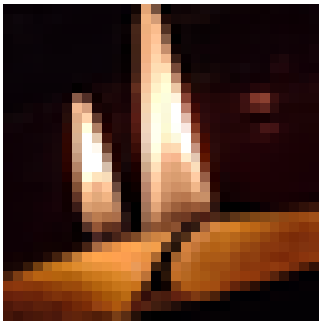

Figure 22 secret message

The extracted message MATLAB output

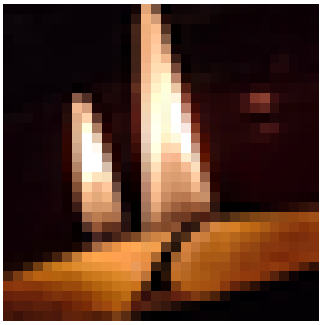

Figure 23 extracted message

Finding the difference between two message

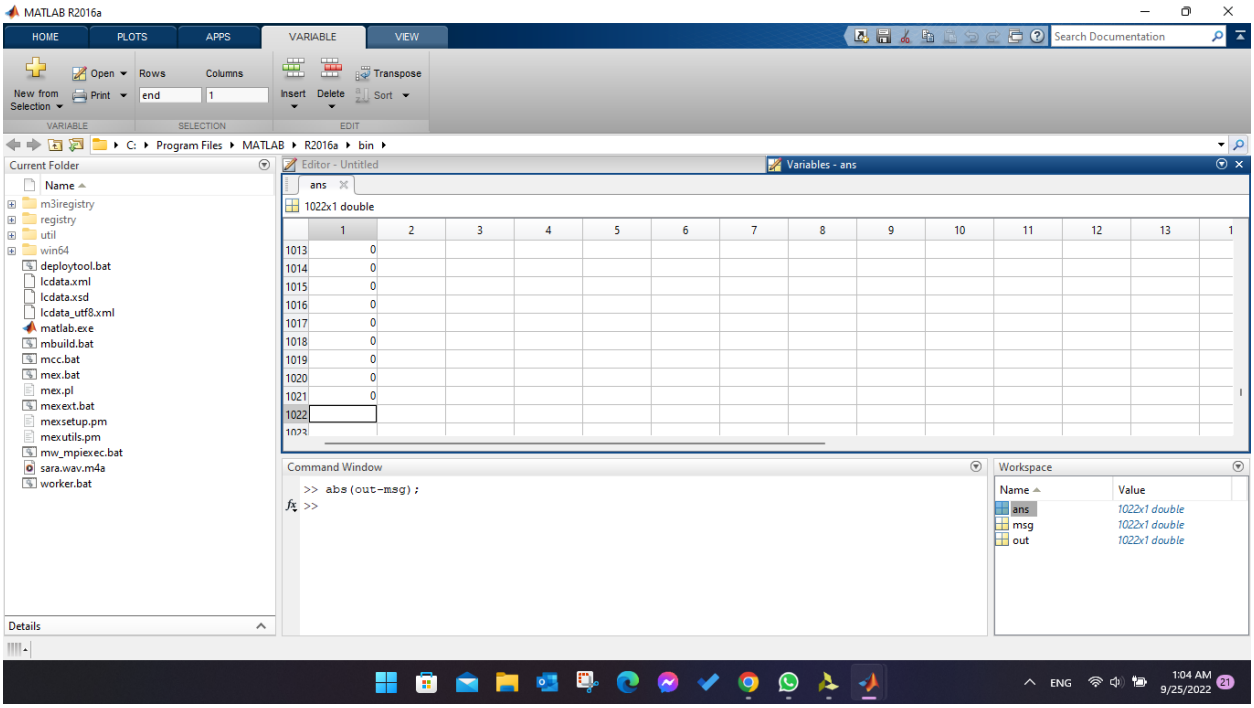

Figure 24 difference between input and output messages

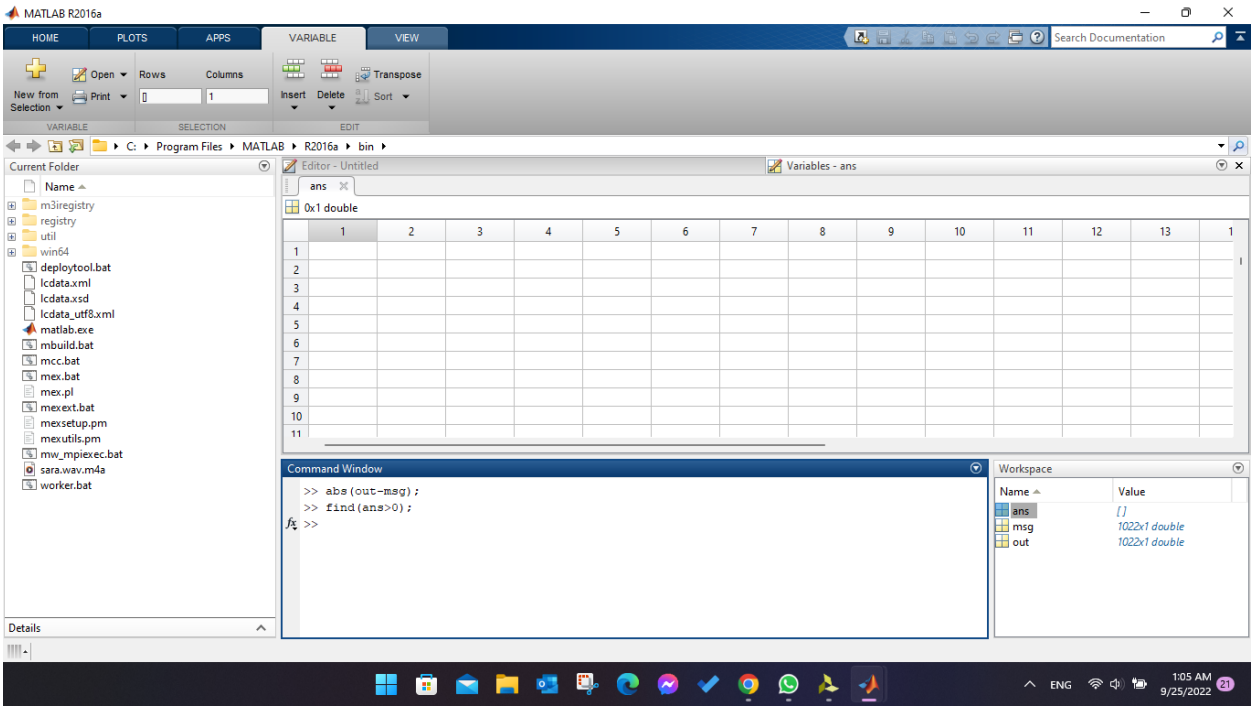

Figure 25 difference output matrix

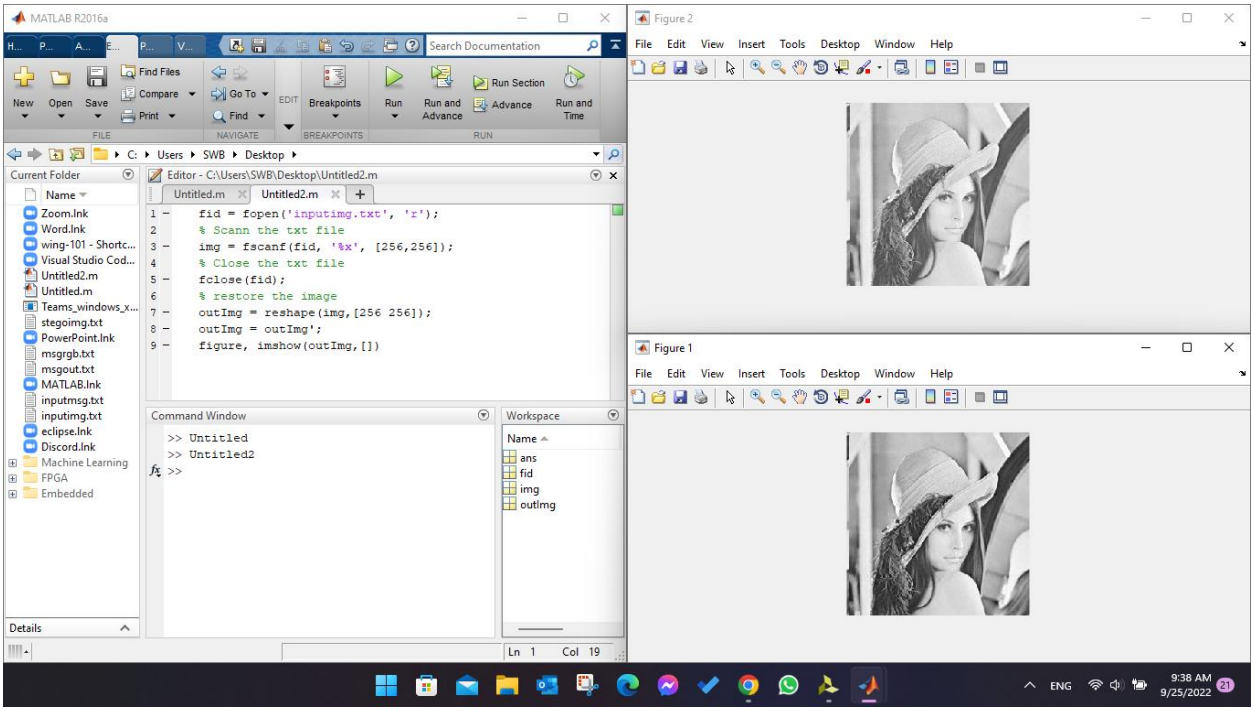

Figure 26 cover image and stego image in grey scale

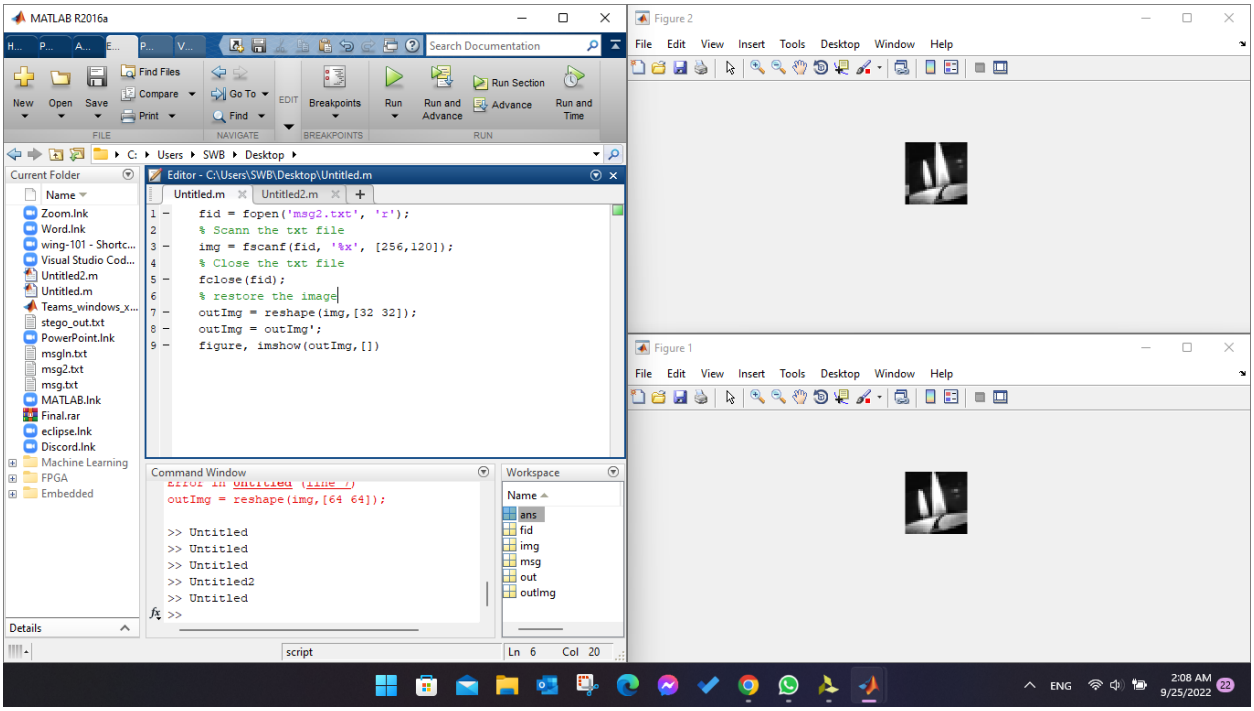

Figure 27 message and extracted message in grey scale

Simulation Results

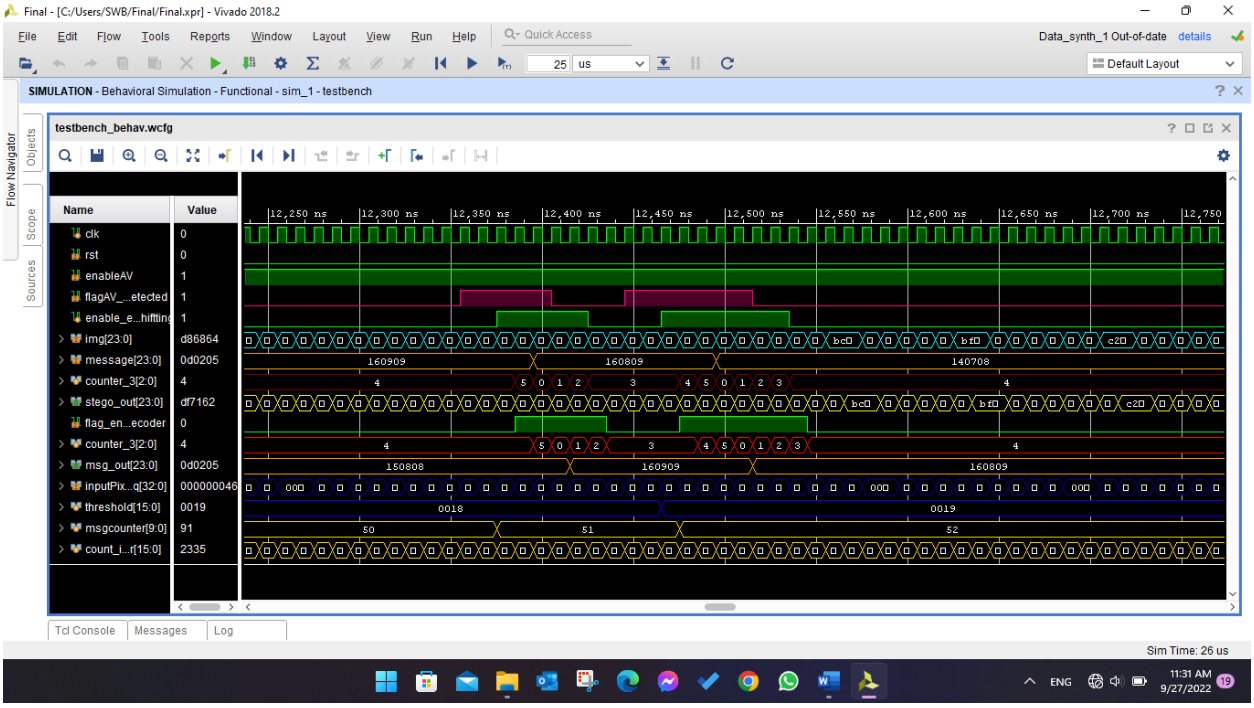

Figure 28 output on simulation

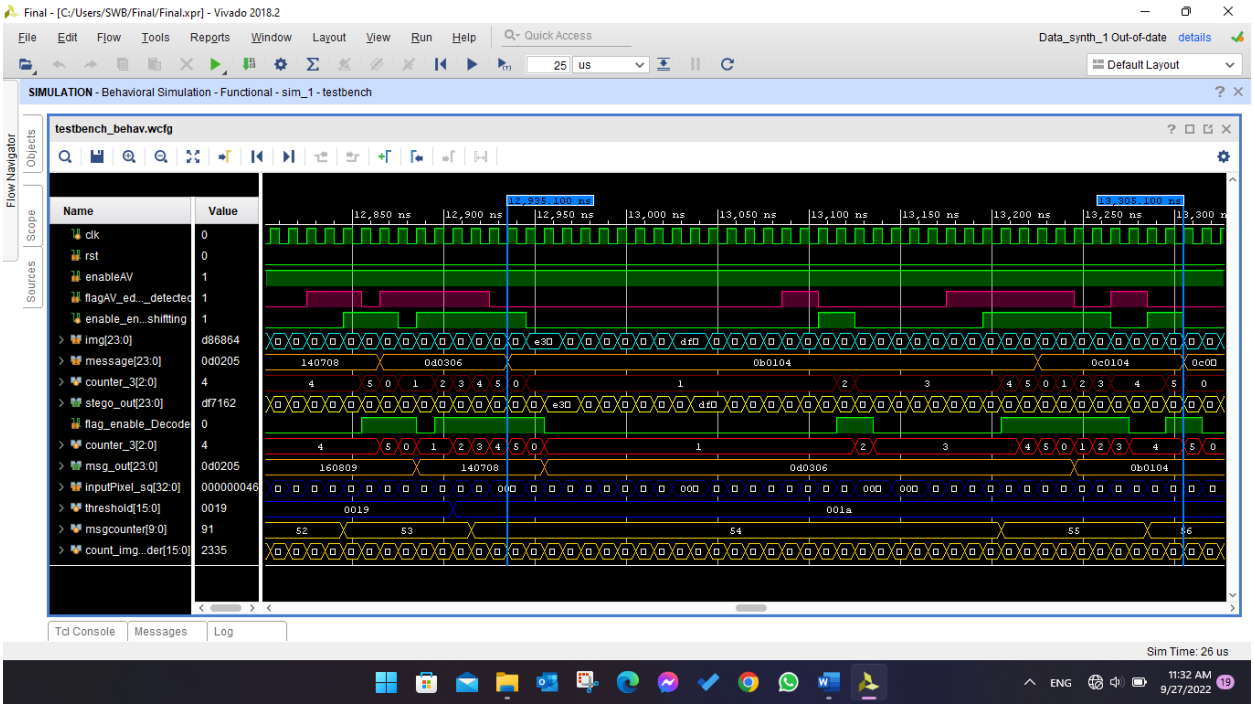

Figure 29 output on simulation 2

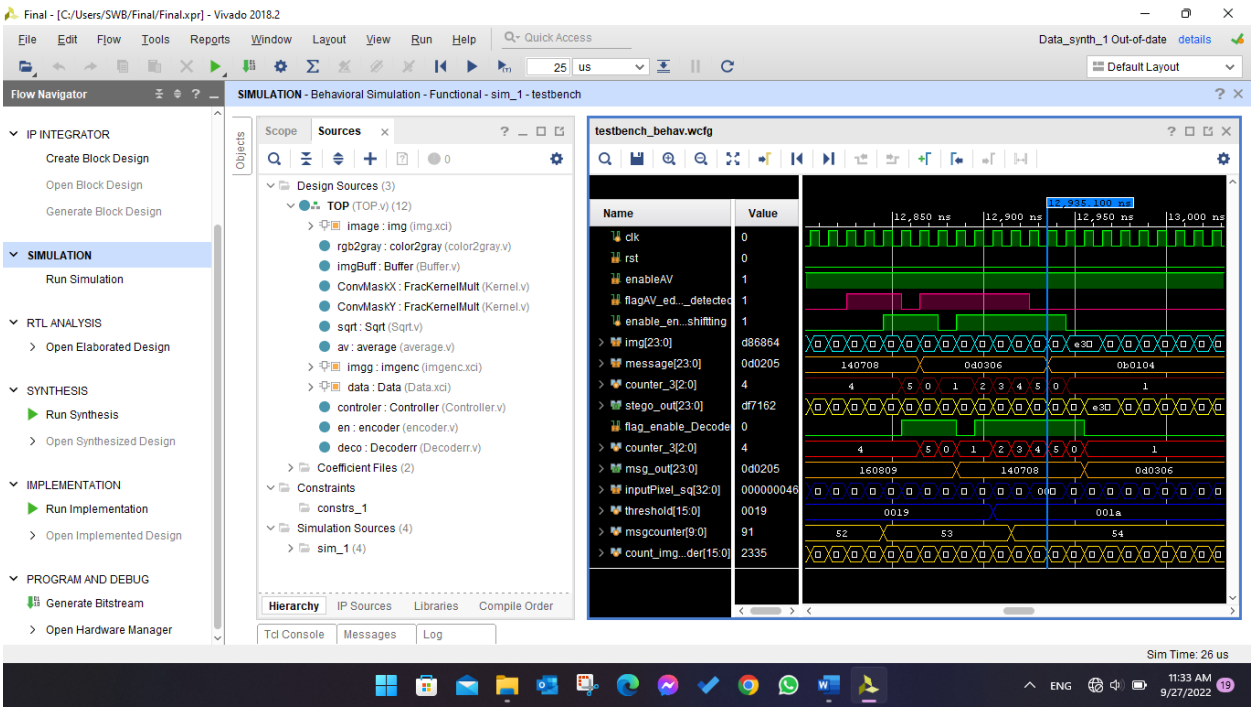

Figure 30 output on simulation 3

References

1. Abd-El-Atty B, El-Latif AAA, Amin M (2016) New quantum image steganography scheme with Hadamard transformation. In: International conference on advanced intelligent systems and informatics. Springer, pp 342–352
2. Akyuz AO, Reinhard E (2006) Color appearance in high-dynamic-range imaging. J Electron Imag 15(3):033001
3. Attaby AA, Ahmed MFMM, Alsammak AK (2017) Data hiding inside jpeg images with high resistance to steganalysis using a novel technique: Dct-m3. Ain Shams Engineering Journal
4. Johnson, Neil F. "Steganography." Http://www.jjtc.com/pub/tr_95_11_nfj/sec101.html. N.p., Nov. 1995. Web.
5. L. M. Russo, E. C. Pedrino, E. Kato, and V. O. Roda. Image convolution processing: A GPU versus FPGA comparison. In Proc. Southern Conf. Pro grammable Logic, pages 1–6, March 2012. doi: 10.1109/SPL.2012.6211783. Cited on pages 1, 2, 13, 39, and 41
6. Niels Provos, and Peter Honeyman. “Hide and Seek: An Introduction to Steganography.” IEEE Security & Privacy Magazine, May-June 2013. Web.
7. Peter Ševčik. Traffic sign recognition system based on the FPGA device uti lization. Proceedings in ITS 2013-Inteligent Transportation Systems, (1), 2013. Cited on page 42.
8. R. G. Shoup. Parameterized convolution filtering in a field programmable gate array interval. Technical report, 1993. Cited on page 12.
9. Shikha, and Vidhu Kiran Dutt. "International Journal of Advanced Research in Computer Science and Software Engineering." Http://www.ijarcsse.com/. N.p., Sept. 2014. Web.