Hardware Implementation of a Secured Digital Camera with Built In Watermarking and Encryption Facility

Chinmayee Das

Swetalina Panigrahi

Department of Electronics & Communication Engineering

National Institute of Technology, Rourkela

Rourkela-769008, Odisha, India

Hardware Implementation of a Secured Digital Camera with Built In Watermarking and Encryption Facility

Thesis submitted in

May 2012

To the department of

Electronics & Communication Engineering

of

National Institute of Technology, Rourkela

in partial fulfillment of the requirements

for the degree of

Bachelor of Technology

In

Electronics & Instrumentation Engineering

By

Chinmayee Das & **Swetalina Panigrahi**

[Roll no 108EI008] [Roll no 108EI037]

Under the guidance of

Prof. Kamalakanta Mahapatra

Department of Electronics & Communication Engineering

National Institute of Technology, Rourkela

Rourkela-769008, Odisha, India

Department of Electronics & Communication Engineering

National Institute of Technology, Rourkela

Rourkela -769008, Odisha, India

Certificate

This is to certify that the work in the thesis entitled ***Hardware Implementation of Secured Digital Camera Architecture with Built in Watermarking and Encryption Facility*** by ***Chinmayee Das*** and ***Swetalina Panigrahi*** bearing Roll No. **108EI008** and **108EI037** respectively, is a record of an original research work carried out by them under my supervision and guidance in partial fulfillment of the requirement for the award of the degree of *Bachelor of Technology* in *Electronics & Instrumentation Engineering.* Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Prof. Kamala Kanta Mahapatra

May 14, 2012

Department of Electronics & Communication Engineering
National Institute of Technology, Rourkela

Acknowledgement

We would like to express our earnest gratitude to our project guide, Prof. Kamala Kanta Mahapatra for believing in our ability to work on the challenging field of protection of Digital Rights Management through designing secured digital camera architecture with built in watermarking and encryption facility. His profound insights have enriched our research work. The flexibility of work he has offered us has deeply encouraged us producing the research.

We wish to extend our sincere thanks to Mr. Vijay Kumar Sharma, M. Tech Research Scholar for his throughout support and encouragement in the entire project period. We would like to thank all the research scholar and staffs of VLSI lab of Department of Electronics & Communication Engineering of National Institute of Technology, Rourkela for extending their help and support as and when required.

We would conclude with our deepest gratitude to our parents, and all our loved ones. Our full dedication to the work would have not been possible without their blessings and moral support.

Chinmayee Das

Swetalina Panigrahi

Contents

List of Figures

List of Tables

ABSTRACT

The objective is to design an efficient hardware implementation of a secure digital camera for real time digital rights management (DRM) in embedded systems incorporating watermarking and encryption. This emerging field addresses issues related to the ownership and intellectual property rights of digital content. A novel invisible watermarking algorithm is proposed which uses median of each image block to calculate the embedding factor. The performance of the proposed algorithm is compared with the earlier proposed permutation and CRT based algorithms. It is seen that the watermark is successfully embedded invisibly without distorting the image and it is more robust to common image processing techniques like JPEG compression, filtering, tampering. The robustness is measured by the different quality assessment metrics- Peak Signal to Noise Ratio (PSNR), Normalized Correlation (NC), and Tampering Assessment Function (TAF). It is simpler to implement in hardware because of its computational simplicity. Advanced Encryption Standard (AES) is applied after quantization for increased security. The corresponding hardware architectures for invisible watermarking and AES encryption are presented and synthesized for Field Programmable Gate Array(FPGA).The soft cores in the form of Hardware Description Language(HDL) are available as intellectual property cores and can be integrated with any multimedia based electronic appliance which are basically embedded systems built using System On Chip (SoC) technology.

1. INTRODUCTION

1.1 Introduction & Motivation

The present 21st century is the era of information and technology. Use of internet has become a primary requirement for all which results the sharing of images & videos as common multimedia application. Manipulation of digital contents of the image is possible with various kinds of image processing tools by the unauthorized user. Hence the protection of the Digital Rights and its enforcement is one of the biggest challenges. The DRM (Digital Rights Management) of an image includes storage, representation, intellectual property rights management, distribution.

Many encryption techniques are there to convert the original data into a form known as cipher text using a key which is not understandable to anyone. Problem is not completely solved as decrypted data can be manipulated by the unauthorized user.

As a solution, a digital signature known as watermark can be embedded into the host image. The watermark should be imperceptible that is it should not be visible to the naked human eye, secured that is after various image manipulation it should be identifiable, it has a inherent quality of undergoing same transformation as that of the host image. Watermark can be extracted from the watermarked image by suitable extraction algorithm. Watermarking and encryption can be used to give double layer security for any kind of digital data.

Algorithms are designed with lesser complexity which will provide easy hardware implementation, as a software solution cannot provide real time performance. Hardware realization provides low power consumption, high speed, and low cost, easy availability. Appliances like digital movable cameras, digital still cameras, mobile phones, digital video displays (DVD) can have these techniques to provide real time performance.

1.2 Literature Review

The concept of digital watermark has been derived from the real life example. Whenever any artist does some kind of paintings, he puts his signature to attest the copyright i.e. no other person could claim to be the owner of that painting. Like that watermark is like a digital signature which is used to protect the ownership right of a digital data. Previously some research has been conducted to embed the watermark into the host image that includes both visible and invisible watermarking. In paper [4] the visible watermarking is done. For invisible watermarking two journal papers have been referred. First one [7] is permutation based watermarking algorithm. Here one binary watermark containing simple text is embedded using simple permutation algorithm. But the embedding factor is a constant value. So for different images it won't give satisfactory result. It won't give better result for jpeg compression. The second one [6] is using CRT (Chinese Remainder Theorem) in DCT domain. Here the one chosen DCT coefficient from each will be converted into a set of integers using pair wise relatively prime numbers. Each time comparison is done between those generated sets of integer and based on that 1 and 0 of the binary watermark are embedded and modified DCT value was noted.

Its computational complexity is very high. Hence it is difficult to implement in hardware. The third one [2] is based on both encryption and watermarking that provides real time performance.

After review of these articles in our project we have proposed a new algorithm for the invisible watermarking.

A comparison result has been presented between our proposed algorithm and previously proposed CRT based and permutation based watermarking algorithm. Also we have implemented both encryption and invisible watermarking unit in hardware. Because a hardware solution gives real time performance, as it is high speed, low power, low cost, easy available.

1.3 Objective

The objective of this project is to design a secured digital camera, that will posses built in watermarking and encryption facility.

The proposed algorithm is should be robust. That means after various image manipulation like jpeg compression, filtering, cropping, tampering the extracted watermark should be identifiable. It should give high PSNR, high NC, and low TAF values.

The proposed algorithm should be computationally less complex to enhance the hardware implementation.

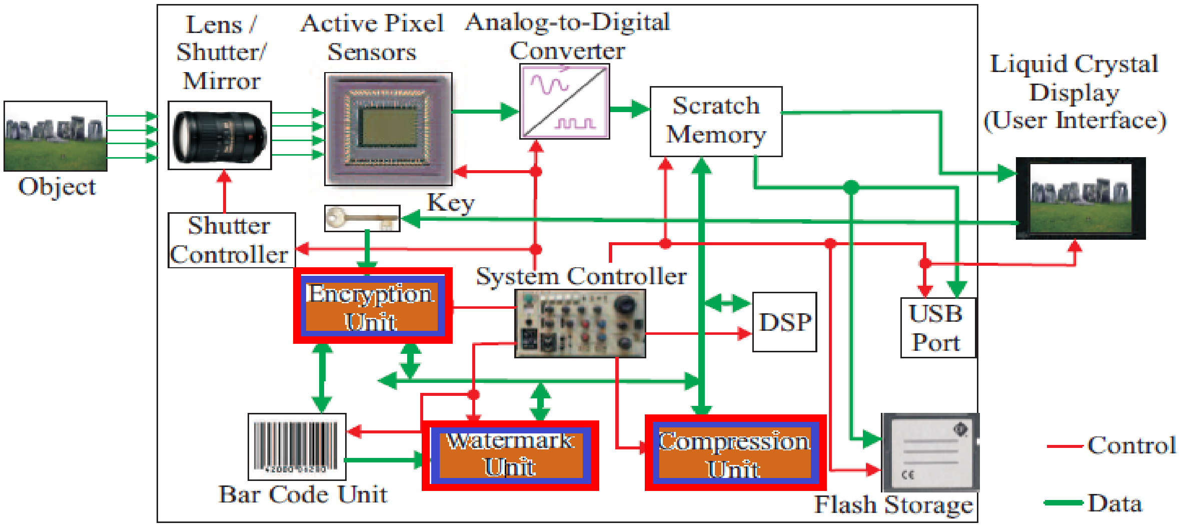


Fig-1.1: Architecture of a secured digital camera. [2]

1.4 Chapter wise organization of the thesis

Chapter-1 Introduction: This chapter describes the introduction and motivation of this project and the objective. It also possesses the literatures which we have referred.

Chapter-2 Discrete Cosine Transformation: This chapter describes the properties of DCT and the advantages to use the DCT domain for watermarking.

Chapter-3 Watermarking: This chapter describes various types of watermarking like visible and invisible watermarking, steps for embedding the watermark. It also possesses the flow of our project as well as the newly proposed algorithm and previously proposed algorithm.

Chapter-4 JPEG compression: This chapter describes the properties of jpeg compression and steps to accomplish this compression.

Chapter-5 Some Image Processing Techniques: This chapter describes various kinds of filtering techniques such as mean, median, and laplacian filtering. It also describes another image enhancement technique i.e. Histogram Equalization.

Chapter-6 Encryption: This chapter describes the properties of encryption and various steps to do this operation and its advantages.

Chapter-7 Comparison of Results: This chapter describes the quality assessment metrics and the comparison results between our proposed watermarking algorithm and previously proposed watermarking algorithm.

Chapter-8 Hardware Implementation: This chapter describes the various hardware modules of the secure digital camera block, RTL schematic and simulation results.

Chapter-9 Summary and Conclusions: This chapter presents the final conclusions and summary of the project.

1.5 Summary

In this introductory chapter, brief description of the encryption and watermarking techniques, the motivation towards the hardware implementation rather than the software solution of the product has been summarized. Literature review of the previous papers has been described. Finally chapter wise contribution of thesis has been summarized.

2. DISCRETE COSINE TRANSFORM (DCT)

2.1 Introduction

The smallest measuring unit of an image is a pixel. In an image each pixel possesses some amount of correlation with its neighboring pixels. This is called *interpixel redundancy*. Since memory management is one of the most important challenges in Hardware design, this redundancy is reduced by changing the domain of operation using a special transform know as DCT. It decorrelates the image data. It transforms the spatial domain in to frequency domain. [3]

2.2 One Dimensional DCT

For forward DCT the formula is

$$\mathrm{C(u)} = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\frac{\pi(2x+1)u}{2N} \qquad (2.1)$$

For u = 0, 1,…… N-1, and N is the length of DCT. In our case N = 8.

Similarly for the inverse Transform is

$$\mathrm{f\ (x)} = \sum_{u=0}^{N-1} \alpha(u)C(u) \cos\frac{\pi(2x+1)u}{2N} \qquad (2.2)$$

For x = 0, 1,…… N-1.

Where $\alpha(u) = \sqrt{(1/N)}$ when u = 0 and $\alpha(u) = \sqrt{(2/N)}$ when $u \neq 0$.

2.3 Two Dimensional DCT

For forward DCT the formula is

$$C(u, v) = \alpha(u)\,\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)\, cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right] \qquad (2.3)$$

For u, v = 0, 1,…… M-1, and N is the length of DCT. In our case N = 8.

Similarly for the inverse Transform is

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right] \qquad (2.4)$$

For x, y = 0, 1,…… N-1,

f(x, y) represents the pixel value of an image.

Where $\alpha(u) = \sqrt{(1/N)}$ when u = 0 and $\alpha(u) = \sqrt{(2/N)}$ when $u \neq 0$

And $\alpha(v) = \sqrt{(1/N)}$ when v = 0 and $\alpha(v) = \sqrt{(2/N)}$ when $v \neq 0$

2.4 Steps for calculation of DCT

(i) An image of any dimension is divided into small blocks of 8×8.

(ii) DCT is operated on each block.

(iii) For u, v = 0, $C\ (0,\ 0) = \frac{\sum_{x=0}^{N-1} f(x,\ y)}{N}$ This value contains the maximum information about an image and known as DC coefficient.

(iv) All other coefficients are known as AC coefficients.

2.5 Properties of DCT

(i) *Decorrelation :* It decorrelates the image data by redcucing the interpixel redundancy. The amplitude of autocorrelation is very less at all lags. [3]

(ii) *Energy Compaction:* It packs the image data into a few coefficients as possible. This allows the quantizer to discard data with small amplitude without introducing any visual distortion. The uncorrelated image contains high frequency content than the correlated image. [3]

(iii) *Symmetry:* The row and column transformations are identical.

$$f\,(x, y) \rightarrow C(u, y) \rightarrow C(u, v)$$

(iv) It is very fast. Computational complexity is very less in comparison to other transforms like DFT.

(v) It is used in JPEG compression which is a standard.

2.6 Conclusion

This summarizes the properties of discrete cosine transform, its advantages over other transforms.

3. WATERMARKING TECHNOLOGY

3.1 Introduction

To protect the owner's right to a particular object a digital signature is embedded into the multimedia data. This digital signature is known as the watermark. The embedded data may be visible or invisible. Accordingly watermarking technique has been divided into two categories. One is visible watermarking and another one is invisible watermarking.

3.2 Domain of Watermarking

For embedding the watermark frequency domain is chosen over spatial domain. As in frequency domain the watermark will be distributed in a wide area. Hence it is more tolerant to normal image cropping and tampering operations. Hence in our project we have preferred the DCT domain as it is very fast and computationally less complex. (Refer previous chapter)

3.3 Visible Watermarking

In visible watermarking of images, a secondary image (the watermark) is embedded in a primary (host) image such that watermark is perceptible to a human observer. [4]

The equation used for modifying the DCT is

$$C_{ij}(n) = \alpha_n C_{ij}(n) + \beta_n W_{ij}(n) \qquad (3.1)$$

Where n = 1, 2, …. is the block number

The α_n and β_n coefficients are for block n. α_n = scaling factor and β_n = embedding factor. These α_n and β_n values are found out using mathematical model developed by exploiting the texture sensitivity of the human visual system (HVS) such that the quality of the watermarked image is not degraded.

$$\alpha_n = \sigma_n exp^{-(\mu' - \mu)^\wedge 2} \qquad (3.2)$$

$$\beta_n = 1/ (\sigma_n)_{(1-} exp^{-(\mu' - \mu)^\wedge 2}{}_{)} \qquad (3.3)$$

σ_n is the variance of AC DCT coefficients.

3.4 Concepts used for selection of Scaling Factor (α_n) and Embedding Factor (β_n)

(i) The edge block should be least altered to avoid significant distortion of the image. That's why α_n becomes maximum and β_n becomes minimum.

(ii) Low frequency regions are perceptually visible than high frequency region. High frequency region means variance is more. α_n is directly proportional to the variance and β_n is inversely proportional to the variance. Watermark should be added in the low frequency region.

(iii) α_n should increase with μ_n as $\mu_n < \mu$ and should decrease with μ_n as $\mu_n > \mu$.

(iv) α_n and β_n should be scaled to the ranges (α_{min}, α_{max}) and (β_{min}, β_{max}). [4]

3.5 Results of PSNR comparison

PSNR (Peak Signal to Noise Ratio) It is an image quality assessment metric. It represents the maximum possible power of a signal and the power of the corrupting noise.

$$\text{PSNR} = 20 \log (b/ \text{rms}) \qquad (3.4)$$

Where b= largest possible value of the signal

Rms = root mean square difference between two images.

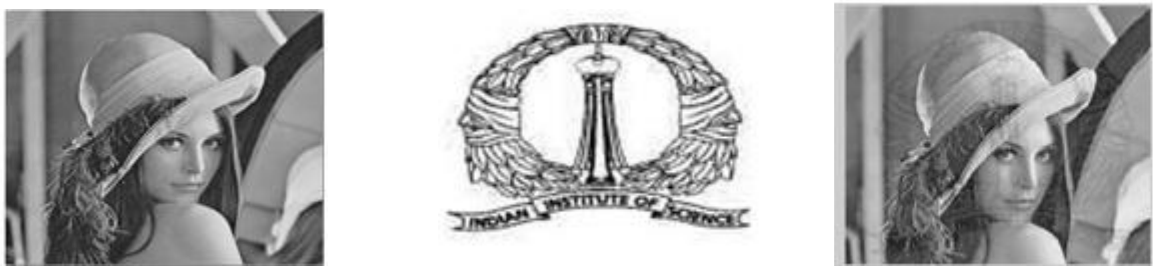


Fig -3.1 Visible watermarking of Lena image

Table 3.1- Results of PSNR comparison

Sl no	Quality Factor	PSNR without Watermark (db)	PSNR with watermark (db)
1	5	32.95	32.7
2	10	31.40	31
3	15	30.45	30.07
4	20	29.54	29.48
5	25	29.63	29.23
6	30	28.82	28.48
7	35	28.42	28.2

From the given set of results it is concluded that the PSNR value is higher for images without watermark. That is because when we are adding a watermark, we are adding some amount of noise to the host image.

3.6 Invisible Watermarking

In this watermarking technique the embedded watermark is imperceptible to the human eye.

This is preferred over visible watermarking, as

(i) It is undetectable by the hackers.

(ii) It provides more security.

(iii) It is perceptually invisible to the human eye.

In this thesis, we have proposed a new algorithm and we have compared the results of our algorithm with previously proposed two watermarking algorithm.

3.6 Description of previously proposed watermarking algorithms

(i) ***Permutation based watermarking algorithm:*** In this paper pseudo random permutation technique has been implemented to embed the watermark with a constant value. Since constant embedding factor is used, it won't give good result for all kind of images and also quality of the extracted watermark and host image is low at high compression. [7]

(ii) ***CRT based watermarking algorithm:*** In this paper Chinese Remainder Theorem is used to convert the DCT value Z to set of integers $\{R_1, R_2\}$ by a relatively pair wise prime numbers $\{M_1, M_2\}$. Each time comparison is done between R_1 and R_2. [6]

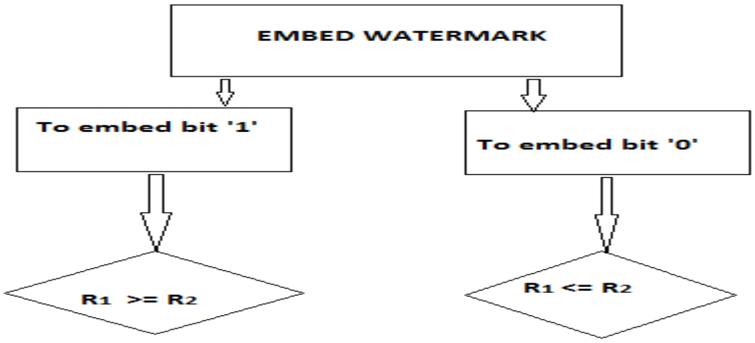


Fig.3.2-Embedding process of CRT algorithm

If these conditions are not satisfied than modification is done in R_1 and R_2. Accordingly from that modified DCT value will be calculated. During extraction inverse procedure will be followed. Since each time it is doing a lot of computations. Its computational complexity is very high. It is difficult to implement in hardware.

3.8 Proposed Algorithm

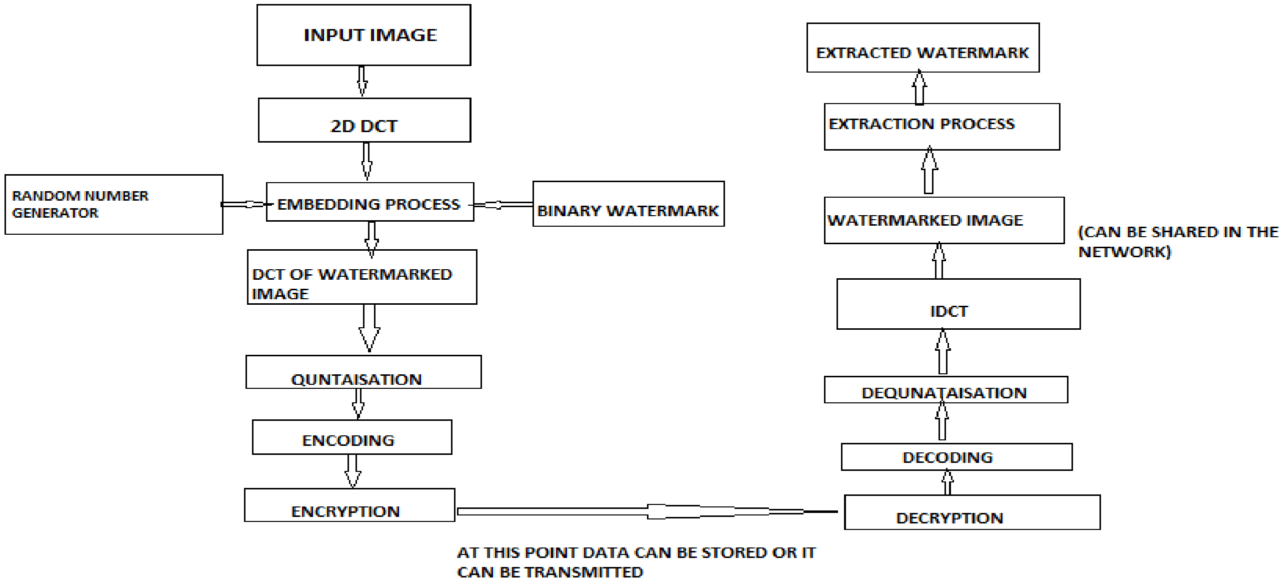


Fig -3.3 Flow diagram of the System

These individual blocks have been described in separate chapters. Here only the embedding and extraction process has been described.

Flow diagram of the embedding process is shown in Fig.3.4.

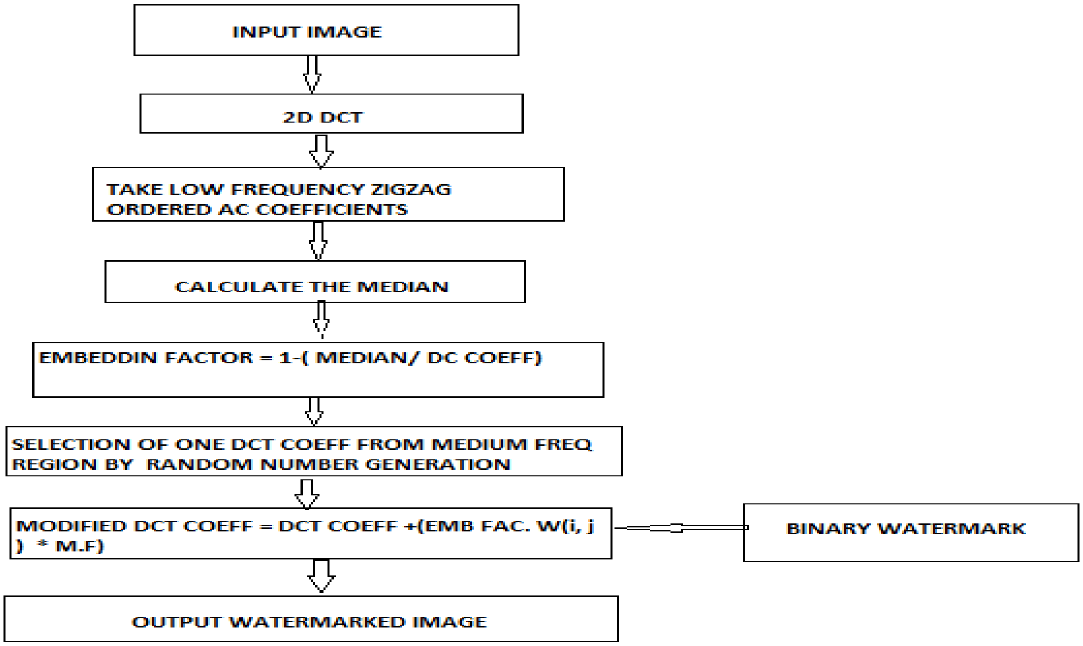

INPUT IMAGE
2D DCT
TAKE LOW FREQUENCY ZIGZAG ORDERED AC COEFFICIENTS
CALCULATE THE MEDIAN
EMBEDDIN FACTOR = 1-(MEDIAN/ DC COEFF)
SELECTION OF ONE DCT COEFF FROM MEDIUM FREQ REGION BY RANDOM NUMBER GENERATION
MODIFIED DCT COEFF = DCT COEFF +(EMB FAC. W(i, j) * M.F)
BINARY WATERMARK
OUTPUT WATERMARKED IMAGE

Fig. 3.4 Flow diagram of proposed algorithm

Here input image of size M × N is taken. It is than divided into small blocks of size 8×8 and 2D-DCT transform was performed. DCT decorrelates the data and gives the result in frequency domain. The result is in increasing order of frequency. Median is calculated from low order AC coefficients as low frequency regions contain more information about an image. Then the embedding factor is calculated using the formula

$$\text{Embedding Factor} = 1 - (\text{median/DC coeff}) \qquad (3.5)$$

DC coefficient of an image contains average information about an image. So when an image is more information DC coefficient will be more. The term (median/DC coeff) becomes less. So embedding factor becomes large. Hence it has direct relationship with the contents of an image.

Important points to note that, here we are calculating the median value for the embedding factor hence it won’t be affected by various kind of noise specially the impulse noise (salt and pepper type). Watermark should be embedded in the mid frequency region of the DCT to make it invisible because the low frequency regions are perceptible to the human eye and high frequency regions are lost in compression. A DCT coefficient is chosen from a set of 4 coefficients in the mid frequency domain range by a pseudo random number generator for embedding the water mark. The coordinates chosen for this are (2, 2), (2, 3), (3, 2) and (3, 3). The modification is done by

$$\text{Modified DCT} = \text{DCT} + \text{Embedding Factor} * W(i, j) * \text{M.F} \qquad (3.6)$$

W(i, j) represents the pixel of the binary watermark.

M.F is the Multiplication Factor.

As given in the fig-3.2, various operations are done after the embedding process. First operation is quantization to remove the psycho visual redundancy to make efficient memory utilization. Then encoding and encryption are done to send the data through insecure public network. After that decryption, decoding, dequantization operations are done. Quantization results in some loss of data as it is not reversible. Then IDCT (Inverse DCT) was performed to get the data in spatial domain. This is the watermarked image which can be shared in public network.

For the extraction following flow diagram is used.

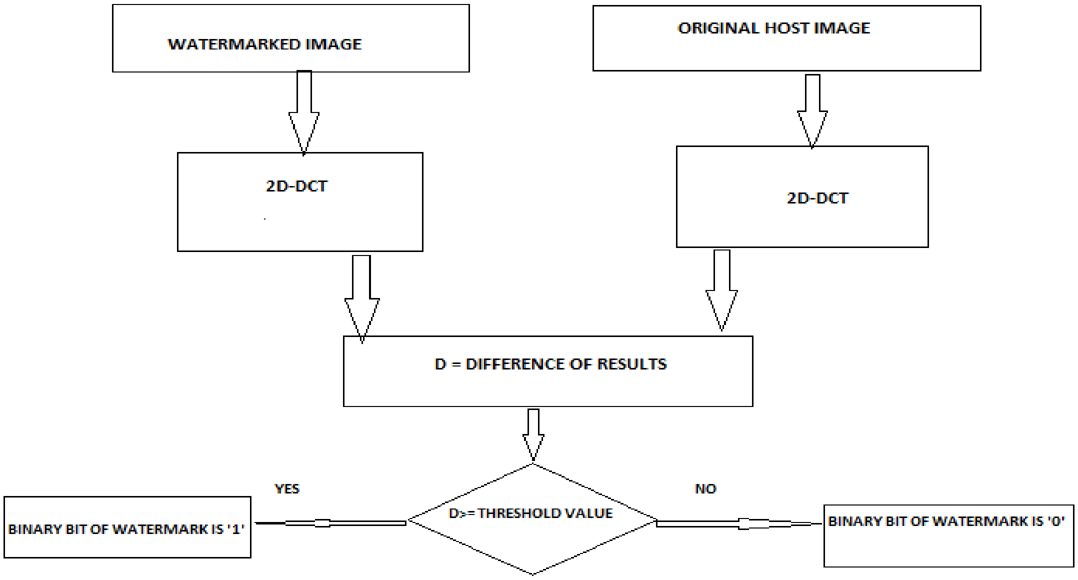

WATERMARKED IMAGE
ORIGINAL HOST IMAGE
2D-DCT
2D-DCT
D = DIFFERENCE OF RESULTS
YES
NO
D>= THRESHOLD VALUE
BINARY BIT OF WATERMARK IS '1'
BINARY BIT OF WATERMARK IS '0'

Fig 3.5- Flow diagram of extraction process

From the above procedure watermark was extracted from the watermarked image and the results are shown below.

Fig- 3.6 Original Lena Image

Fig- 3.7 Binary Watermark

Fig – 3.8 Watermarked Image without attack

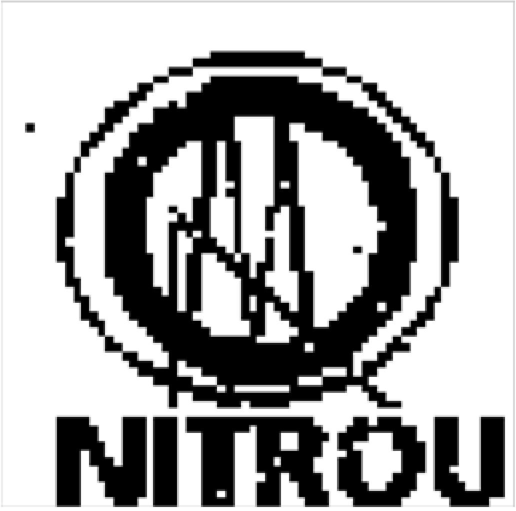


Fig- 3.9 Extracted Watermark

The above fig- 3.8 shows the invisible watermarking of the Lena Image with a binary watermark shown in fig 3.7 without any attack. Fig- 3.9 shows the extracted watermark. This watermark is well identifiable.

Fig -3.10 Cameraman Image

Fig – 3.11 Watermarked Image

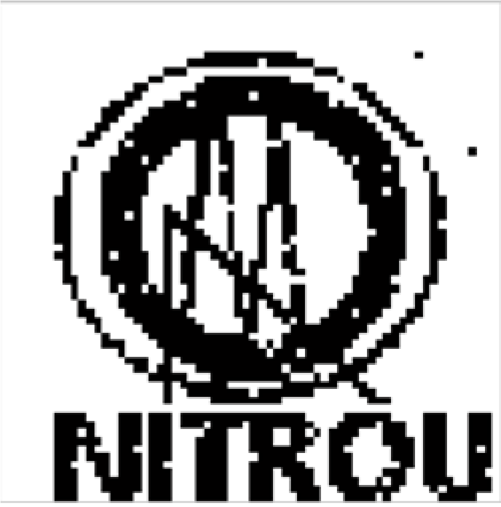


Fig -3.12 Extracted Watermark

The above diagram shows the invisible watermarking of the cameraman image with binary watermark given in fig 3.7. Fig – 3.11 represents the watermarked image and the fig 3.12 represents the extracted watermark.

3.9 Conclusion

The proposed algorithm works well for varied kind of images since the embedding factor takes into account the characteristics of the image.

4. JPEG COMPRESSION

4.1 Introduction

A digital image is a two-dimensional function of the spatial coordinates, $f(x, y)$, where f is the intensity or gray level of the image. It is a matrix of a finite number of picture elements or pixels. [1].Nowadays typical images require storage of the order of millions of bytes. Hence, use of digital images is not practical unless the storage and transmission costs can be reduced. Image Compression refers to the process of reducing the amount of data required to represent an image.JPEG (Joint Photographic Experts Group) is an international compression standard aimed at meeting all continuous tone still image applications. The JPEG method includes two compression methods, a DCT based method for lossy compression and predictive coding for lossless compression. JPEG features a lossy technique, the baseline method which is one of the DCT based methods and is the most widely used. [12]

4.2 Processing Steps for DCT based Coding

DCT based coding is the most preferred and commonly used lossy image compression method. Fig.4.1 and Fig.4.2 show the basic processing steps of this type of coding. The purpose is to remove different types of redundancies so that the same amount of information can be accommodated in less space. There exist three types of redundancies in an image.

1. *Coding Redundancy*: In an image, 8-bit codes are used to represent the intensities of the pixel values which contain more bits than are required to represent them. This coding redundancy is removed during encoding.[1]
2. *Interpixel Redundancy*: In an image, a pixel exhibits some kind of correlation with its neighbouring pixels. So, the information content in a single pixel is very less. This results

in redundancy which can be removed using various transform techniques like DCT which decorrelates the image pixels.[1]

3. *Psycho visual Redundancy*: In an image, there exists some information which is ignored by the human visual system. This limitation of the human eye is exploited and quantization aims at reducing what is called psycho-visual redundancy and in the process reduces memory requirements.[14]

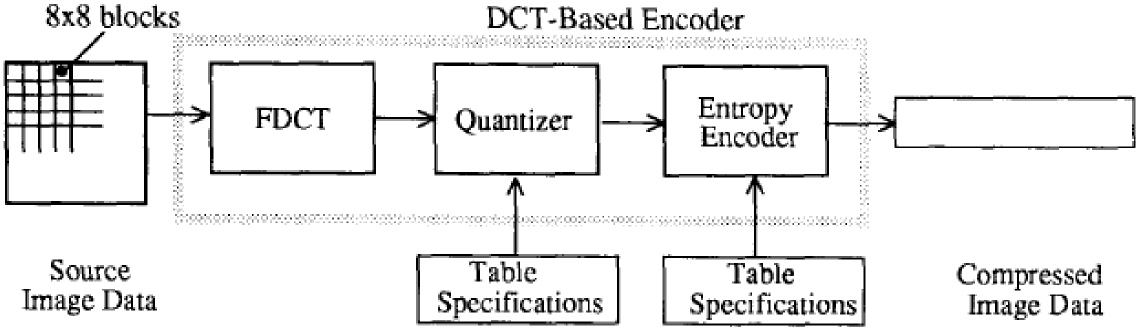


Fig.4.1 DCT based Encoder Processing Steps

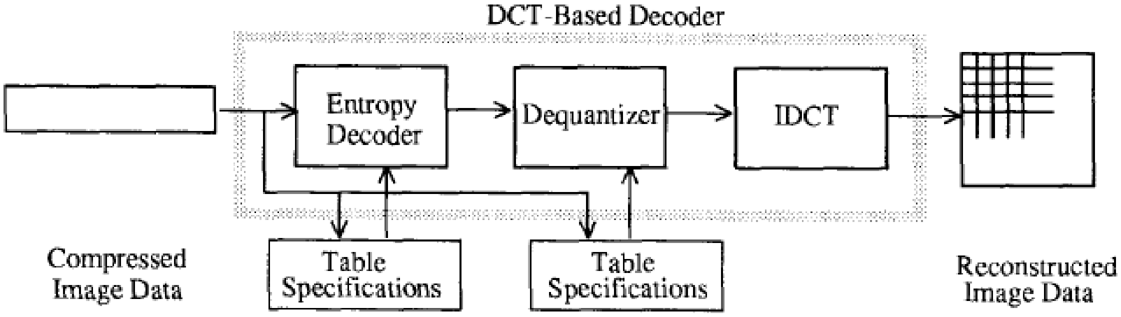


Fig.4.2 DCT based Decoder Processing Steps

4.2.1 FDCT and IDCT

The image is divided into 8 x 8 blocks, shifted from unsigned integers with range [0, 2^n-1] to signed integers with range [-2^{n-1}-1, 2^{n-1}] where n is the no of bits used to represent each image pixel. This is then input to the FDCT. At the decoder output, the inverse DCT is calculated to transform the image back to the spatial domain for display purposes. [12] The following are the mathematical equations of the 8 x 8 FDCT and 8 x 8 IDCT :

$$C(\mathrm{u, v}) = \alpha(\mathrm{u})\, \alpha(\mathrm{v}) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y)\, cos\, [\frac{\pi(2x+1)u}{2N}]\, cos\, [\frac{\pi(2y+1)v}{2N}] \qquad (4.1)$$

For u, v = 0, 1,…… N-1, and N is the length of DCT. In our case N = 8.

Similarly for the inverse transform

$$f(\mathrm{x, y}) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v)\, \cos[\frac{\pi(2x+1)u}{2N}]\, \cos\, [\frac{\pi(2y+1)v}{2N}] \qquad (4.2)$$

For x, y = 0, 1,…… N-1,

Where *f*(x, y) represents the pixel values of an image.

4.2.2 Quantisation

The JPEG recommended normalization array is used to quantize the DCT transformed array. If T is the transformed array, Z is the normalization array and S is the normalized transformed array, then

$$S(i, j) = round[\frac{T(i,j)}{Z(i,j)}] \qquad (4.3)$$

Where i, j = 0,1,…..,7

$$\mathrm{Z=} \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

This is done for each 8 x 8 block. [1]

4.2.3 Entropy Encoding

4.2.3 Entropy Encoding

This is the final step in the DCT based encoder. It achieves additional compression losslessly by encoding the quantized coefficients based on their statistical characteristics thus removing coding redundancy. [12] It consists of the following operations:

1. The quantized coefficients of each block are reordered using the zigzag pattern as shown below to form a 1-dimensional sequence of quantized coefficients. The zigzag pattern is shown in Fig.4.3. [1]

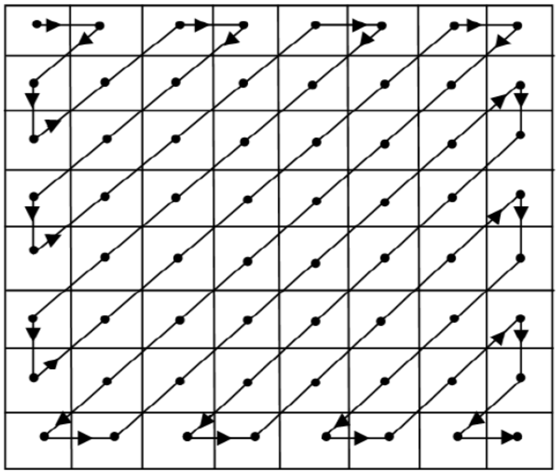


Fig.4.3 Zigzag Ordering for DCT Coefficients

2. The sequence generated is rewritten till the last non-zero coefficient of each block and then a special symbol called EOB is inserted to indicate the end of the block. For example-a sequence like [-26 -3 1 -3 -2 -6 2 -4 EOB] means after reordering in zigzag *Encoding of DC coefficients*: The difference between the current DC coefficient and that

of the previously encoded block is computed. The DC difference category is found from the table and then the base code is noted down for that category. For category K, an additional K bits are needed and computed as K LSB's of the positive difference or the K LSB's of the negative difference minus 1. [1]

Example-If the current DC coefficient is -26 and that of the previous block is -17, then [(-26)-(-17)]=-9 which lies in the DC difference category 4, for which the base code is 101. (-9), the difference is encoded as (0111)-1=0110 & the complete DPCM coded DC code word is 1010110. [1]

3. *Encoding of the AC coefficients:* The category of each non-zero AC coefficient is found out using the table. According to the number of zeros preceding each coefficient and its category, the base code is noted from the JPEG default AC code table. Then the extra K bits for category K is found in the same way as in DC encoding.

   There is a special JPEG code for a run of 15 zeros followed by a zero (Run length F & category 0-111111110111). The EOB is encoded as 1010.The completely coded array is now stored to get the compressed image data. [1]

3.2.4 The DCT based Decoder

The decoder constitutes the following steps whose final output is the reconstructed image.

1. The encoded stream is then decoded using the look up tables and reordered to get the normalized transformed array S in the matrix form.
2. S is renormalized to get the transformed array S'

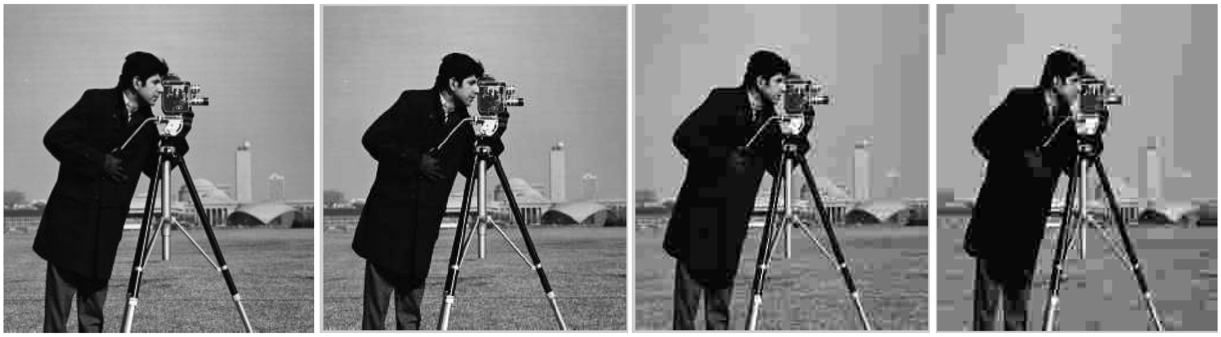
$$S'(i,j) = S(i,j).Z(i,j) \qquad (4.4)$$

3. The 2-D inverse DCT of the demoralized array is computed according to the equation (4.2).
4. Then, each coefficient is level shifted by 2^n to yield the final reconstructed image pixel array where n is the no of bits used to represent each image pixel.

4.3 Results and Observations

The degree of compression can be varied by multiplying higher factors to the quantization table. The compression ratio will be improved but at the cost of image quality. The compression ratio is calculated as the no of bits required to represent the original image divided by the no of bits in the encoded array.

The cameraman image was subjected to compression with different quality factors and the image quality was measured using the PSNR (Peak Signal to Noise Ratio).Some of the results are shown in table 3.1.



Original Image

QF=1, CR=9.64

QF=5, CR=26.72

QF=10, CR=39.82

Fig.4.4 Compression Results with different Quality Factors

4.4 Conclusions

The quality of the image measured by its PSNR decreases as the compression ratio increases but the memory required becomes lesser so it is a tradeoff between image quality and memory requirements.

5. SOME IMAGE PROCESSING TECHNIQUES

5.1 Introduction

In this chapter, common image processing techniques like filtering, histogram equalisation etc are described which will be later used to check the robustness of the watermarking algorithm.

5.2 Filtering

Filtering can be performed in spatial as well as frequency domain. Here spatial domain is used because it is computationally less intensive and easier to implement. A 3 X 3 filter mask is used. It is moved over the entire image and the centre coefficient is calculated as

$$R = \sum_{k=1}^{9} w\,(k).\,z(k)$$

Where w and z are the 9-dimensional vectors formed from the coefficients of the mask and the image pixels encompassed by the mask. [1]

There are different types of filtering which are performed, the most common among them being low pass filters like mean, median and high pass filters like laplacian. They have different uses as described below.

5.2.1 Mean Filtering

It is a low pass linear filter which is used for smoothing operation. Each pixel value is replaced by the average value of its neighbours including itself. Weighted or arithmetic mean may be used. [5]The masks for both types of filters are shown.

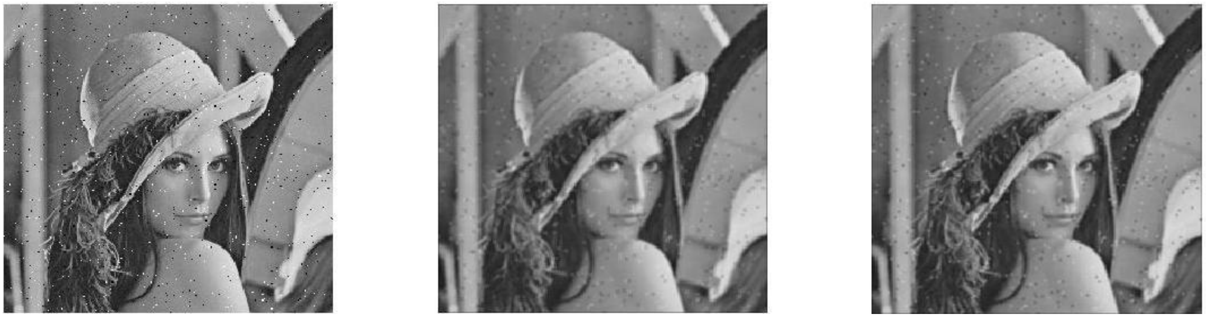
1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

Fig.5.1 Mask for weighted Mean

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Fig.5.2 Mask for Arithmetic Mean

The weighted filter gives better results because the weights are distributed according to the distance of the pixels from the centre. [1] The results of the two types of filtering are shown below.



(a) Noisy image (b) Using weighted mean filter (c) Using Arithmetic mean Filter

Fig.5.3 Results of Mean Filtering

5.2.2 Median Filtering

This is an order statistic filter where the response is based on the ordering of the pixels. The centre coefficient is replaced by the middle value among the 9 pixels including it. It is a non linear low pass filter mainly useful for removing impulse noise like salt and pepper noise.[1] The results of median filtering are shown below.

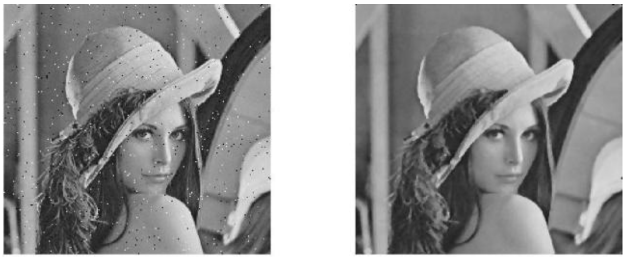


Fig.5.4 Results of Median Filtering

5.2.3 Laplacian Filtering

It is a high pass filter which finds the 2-D second order derivative. It is isotropic meaning it is invariant to rotational effects. It deemphasizes the slowly varying regions and highlights the gray level discontinuities.[5] The discrete laplacian of two variables is

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

The laplacian produces images with grayish edge lines and other discontinuities superimposed on a dark, featureless background. So, the original image is added or subtracted from the laplacian image to get the sharpened final image depending upon the sign of the centre coefficient

$$g(x,y) = f(x,y) + c[\nabla^2 f(x,y)]$$

Where c = 1 if the centre coefficient is positive and c = -1 if it is negative.[1]

The mask for the laplacian filters are shown below.

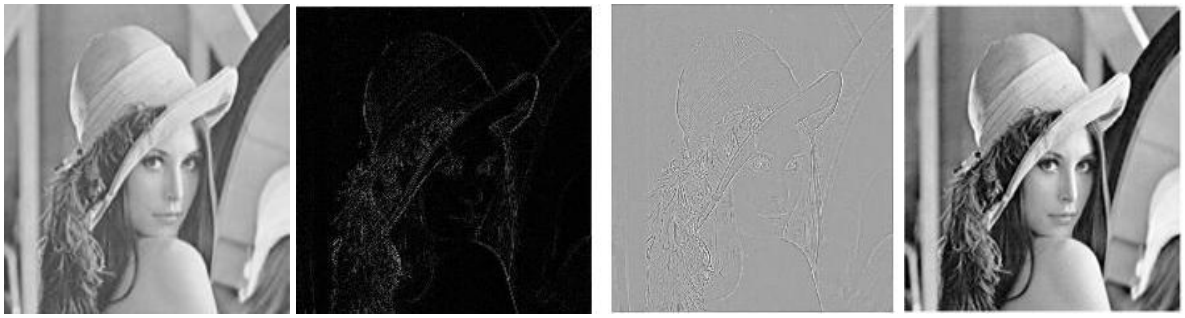
1	**1**	**1**
1	**-8**	**1**
1	**1**	**1**

Fig 5.5 Mask for laplacian filter (negative centre coefficient)

-1	**-1**	**-1**
-1	**8**	**-1**
-1	**-1**	**-1**

Fig 5.6 Mask for laplacian filter (positive centre coefficient)

The results of Laplacian filtering are shown below.



Blurred image, Laplacian image, Scaled laplacian image, Sharpened Image

Fig- 5.7 Results of Laplacian Filtering

5.3 Histogram Equalisation

It is a process used to enhance the contrast of an image. The transformation function is defined as follows.

$$s = T(r) = (L - 1)\int_0^r p_r(w)dw$$

Where r is the input intensity and s is the output intensity level, $p_r(w)$ is the PDF(probability density function) of the input intensity level.[book]

The integration ensures that the pixels are evenly distributed over all gray levels hence enhancing the contrast.[1]

5.4 Conclusion

The filtering and histogram equalization techniques are summarized here which will be required later in our comparison of the proposed and previously existing watermarking algorithms.

6. ENCRYPTION

6.1 Introduction

Encryption is the process of converting information called plaintext using an algorithmic approach into a form called the cipher that is not easily recognizable by an unauthorized user. Only authorized users possessing a unique key can decode the information. This is required to transmit the information over unsecured networks. [10] The cipher can then be decrypted at the receivers end by performing the reverse operations. There are various algorithms used for encrypting data, one of the most common being the AES (Advanced Encryption Standard) which is implemented here..

6.2 AES Algorithm

The AES algorithm uses a single key to encrypt and decrypt the information. The key length used is 128 bits although it can be 192 or 256 bits. [9] All the operations are performed on a two dimensional array of bytes called state where each byte consists of 8 bits. The state consists of 4 rows of bytes and each row has 4 bytes. At the input of encryption, the array of input bytes is mapped to the state array. The encryption/decryption are performed on the state to obtain the final value. The key of this algorithm can be mapped to four rows of bytes, the number of bytes in each row denoted by *Nk,* it being 4 in our case (length of the key is 128 bits). The AES algorithm is iterative where each iteration is a round. The number of rounds *Nr* depends on the key length, for *Nk* =4 ,*Nr*=10 which is applicable in this case.[9]

6.2.1 Encryption

First the input of 16 bytes is copied to the state array. The initial key is then xor-ed with the state. Then *Nr*-1 rounds are performed, each of the rounds consisting of 4 transformations-Sub Bytes, Shift Rows, Mix Columns and Add Round Key. The final round excludes the Mix Columns transformation. Also a round key is generated at every round using key expansion. [9] The Encryption process is shown in Fig.6.1.

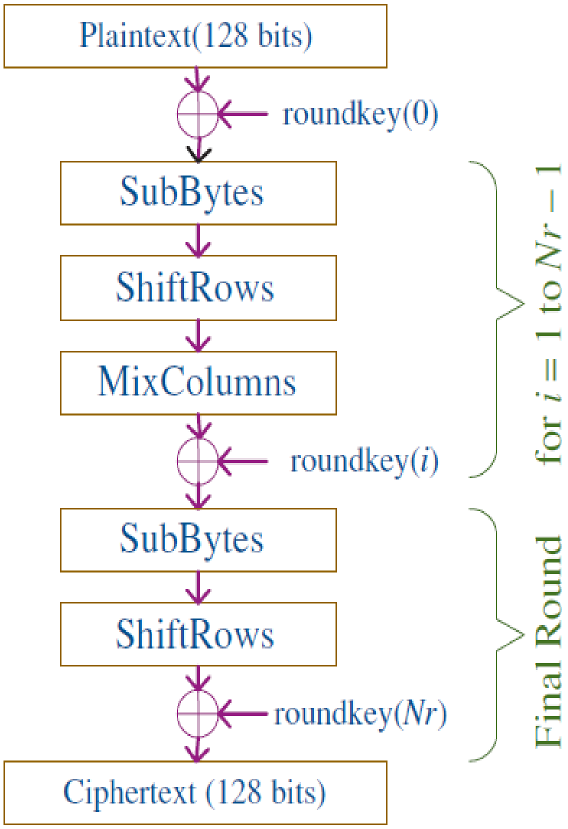


Fig 6.1 Encryption process of AES algorithm[9]

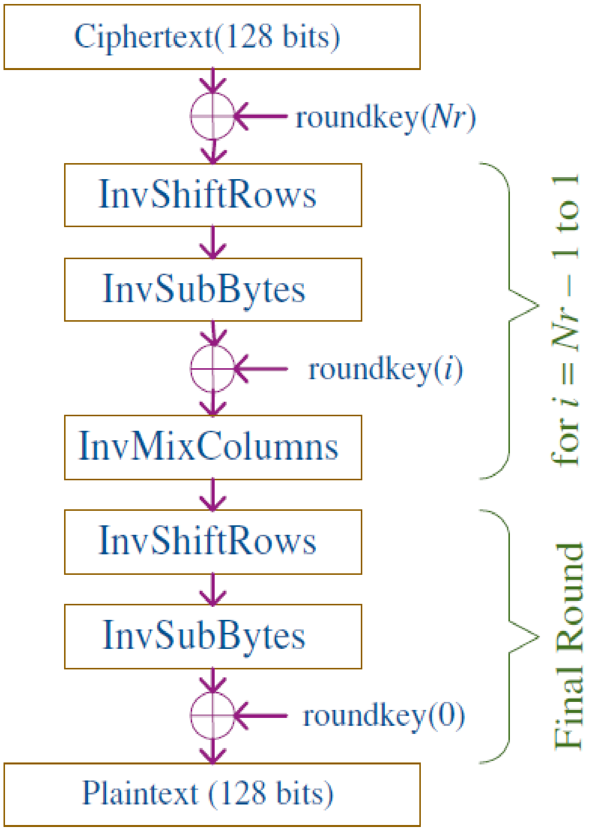


Fig 6.2 Decryption Process for the AES[9]

The different transformations are described below:

1. *Sub Bytes Transformation:* It is a mapping of the bytes using a substitution table(S box). This box is obtained by taking the multiplicative inverse in the finite field GF (2^8) with the irreducible polynomial m(x) = $x^8 + x^4 + x^3 + x + 1$.The element {00} is mapped to itself. [8]

2. *Shift Rows Transformation*: It cyclically shifts the rows of the state by different offsets.

.

3. *Mix Columns Transformation*: Each column is taken as a four term polynomial. The polynomial is then multiplied by modulo $x^4 + 1$ with a fixed polynomial [8]

$$a(x) =\{03\}\ x^3 + \{01\}\ x^2 + \{01\}\ x + \{02\}.$$

4. *Add Round Key Transformation*: A round key is added to the state by a simple bitwise XOR operation.[8]

Key Expansion: Each round key is a 128 bit array generated as a product of the previous round key, a constant that changes every round, and a series of S-box lookups for each key. The round key generated after each round is xor-ed with the Mix Column output. [8]

6.2.2 Decryption

All the operations described above are performed in the reverse order during decryption. The 128 bit cipher is fed at its input and finally converted back to plaintext. Add Round Key is the same as in encryption. However, the other processes have their corresponding inverses-Inverse Sub Bytes, Inverse Shift Rows, Inverse Mix Columns.[10] The flowchart of the decryption process is shown in Fig.6.2.

6.3 Conclusion

The Advanced Encryption Standard Algorithm is an iterative private key symmetric block cipher which processes data blocks of 128 bits with the help of a unique key available only to the user. The key length can vary like it can be of 128,192 or 256 bits. In our project, the watermarked image is quantized, encoded and then encrypted ensuring security of information when transmitted over public networks.

7. COMPARISON OF RESULTS

7.1 Introduction

In this chapter the results of the proposed watermarking algorithm is compared with the previously proposed watermarking algorithm on the basis of quality assessment metrics. The results are generated after doing various kind of image manipulation operation.

7.2 Quality Assessment Metrics

(i) ***Peak Signal to Noise Ratio:*** **PSNR** compares the quality of the watermarked image and the host image. It is given by the formula [6]

$$\text{PSNR (dB)} = 10\,\log_{10}\frac{255^2}{\frac{1}{M}\frac{1}{N}\sum_{i=0}^{M}\sum_{j=0}^{N}(A(i,j)^2 - A'(i,j)^2)} \qquad (7.1)$$

Higher the value of PSNR better is the algorithm. 40 dB is for a good quality image.

(ii) ***Normalized Correlation:*** **NC** gives the correlation between the extracted watermark and the original watermark taken. Higher the value of NC better is the algorithm. It is given by the formula. [7]

$$\text{NC} = \frac{\sum_i\sum_j W(i,j)W'(i,j)}{\sum_i\sum_j [W(i,j)]^2} \qquad (7.2)$$

(iii) ***Tampering Assessment Function:*** **TAF** gives how many bits of the extracted watermark and the original watermark are equal. It is given by the formula [6]

$$\text{TAF (\%)} = \frac{1}{mn}\sum_{i=0}^{m}\sum_{j=0}^{n} W(i,j)^{\wedge}\,W'(i,j) \times 100 \qquad (7.3)$$

The symbol ^ represents the XOR operation.

W(i, j) represents the original watermark image.

W’(i, j) represents the extracted watermark image.

Lower the value if TAF (< 15%) better is the algorithm.

7.3 Results

TABLE -7.1 Comparison Results of Proposed and CRT algorithm with Lena Image (512 X 512)

	WITHOUT ATTACK	JPEG COMP. CR=12.43	JPEG COMP. CR=19.04	JPEG COMP CR=21.1	MEAN FILTER	MEDIAN FILTER	HIGHPASS FILTER	NOISE + MEAN	HISTOGRAM EQUALISATION
PSNR(CRT) (dB)	41.42	38.46	--------	-------	-------	-------	26.5	35.5	27.3
PSNR(PROPOSED SCHEME) (dB)	42.29	39.2	37.49	36.58	39.7	40.8	30.1	38.6	32.13
TAF (CRT) (%)	0.51	4.91	--------	-------	------	------	11.5	11.4	15.82
TAF(PROPOSED SCHEME) (%)	0.73	2.38	9.62	11.69	5.21	5.54	7.24	12.0	9.57

TABLE -7.2 Results of Proposed algorithm with Cameraman Image (512 X 512)

	WITHOUT ATTACK	JPEG COMP CR=12.64	JPEG COMP CR=18.59	JPEG COMP CR=20.71	MEAN FILTER	MEDIAN FILTER	HIGHPASS FILTER	NOISE+ MEAN	HISTOGRAM EQUALISATION
PSNR(PROPOSED SCHEME) (dB)	42.4	39.28	37.59	36.61	39.12	40.25	30.48	38.06	29.43
NC(PROPOSED SCHEME)	0.999	0.9981	0.9923	0.9849	0.98721	0.9729	0.9776	0.9702	0.9687
TAF(PROPOSED SCHEME) (%)	1.6885	4.3	11.74	12.5	7.4	6.8	11.13	12.9	12.19

TABLE- 7.1 gives the comparison of results between our proposed algorithm and the previously proposed CRT based watermarking algorithm for the Lena image. It has been noted that the PSNR in dB for our algorithm is greater than that of CRT based watermarking algorithm for various kind of image manipulation techniques and also the TAF value is lower than that of the CRT based algorithm. TABLE-7.2 gives the result of various operations using the proposed algorithm for the cameraman image.

TABLE-7.3 Comparison Results of Proposed and Permutation algorithm with Lena Image (512 X 512)

COMPRESSION RATIO	**12.43**	**19.04**	**21.1**
PSNR(PROPOSED SCHEME) (dB)	39.2	37.49	36.58
NC(PROPOSED SCHEME)	0.998	0.991	0.984
COMPRESSION RATIO	9.05	9.81	10.74
PSNR(PERMUTATION SCHEME) (dB)	31.47	31.41	31.17
NC(PERMUTATION SCHEME)	0.661	0.493	0.413

TABLE-7.3 gives the comparison of results between our proposed algorithm and the previously proposed permutation based watermarking algorithm. It has been noted that the PSNR in dB for our proposed algorithm is greater than that of the previously proposed permutation based watermarking algorithm and NC value is also greater than that of the previously proposed watermarking algorithm. The watermarked image has been processed with various kind of image manipulation operation. And the images of the operation and extracted watermark are given below.

Fig – 7.1 Original Lena Image and the Original Binary Watermark

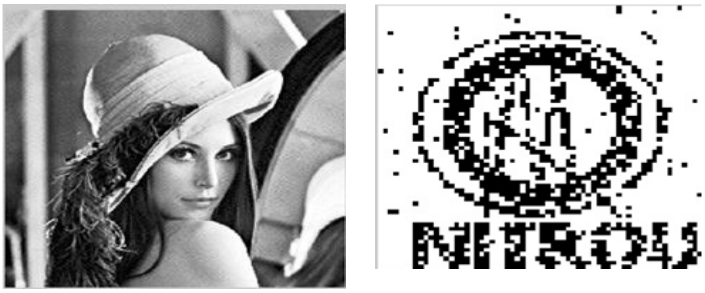


Fig- 7.2 Results of histogram equalization and the extracted Watermark

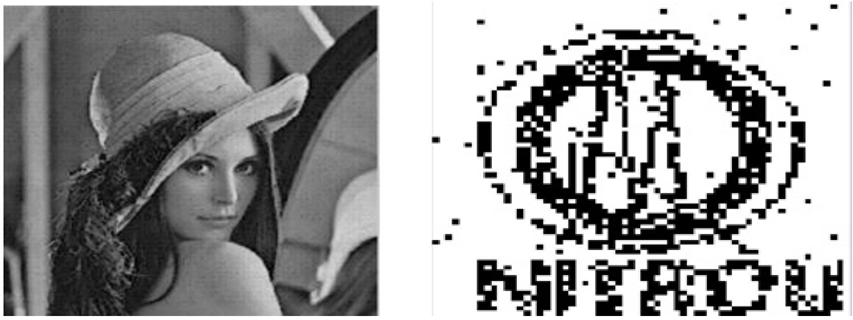


Fig- 7.3 Results of high pass Filtering and the extracted Watermark

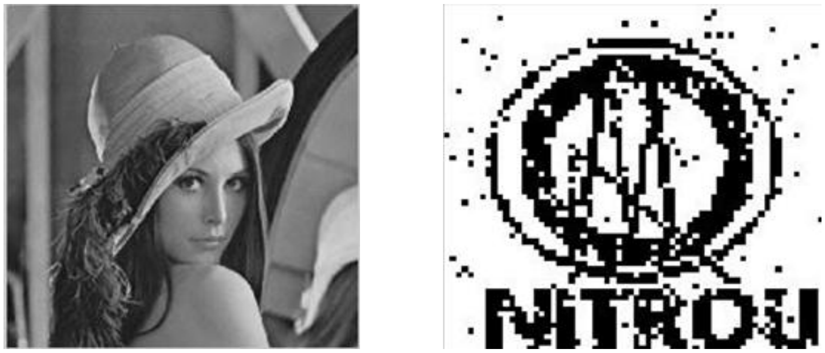


Fig – 7.4 Results of median Filtering and the extracted watermark

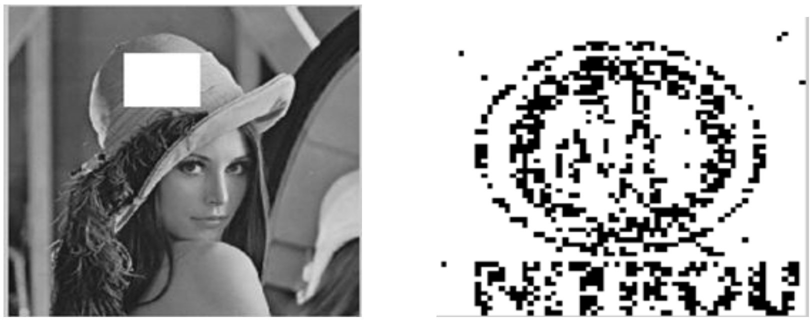


Fig – 7.5 Results of tampering and the extracted watermark

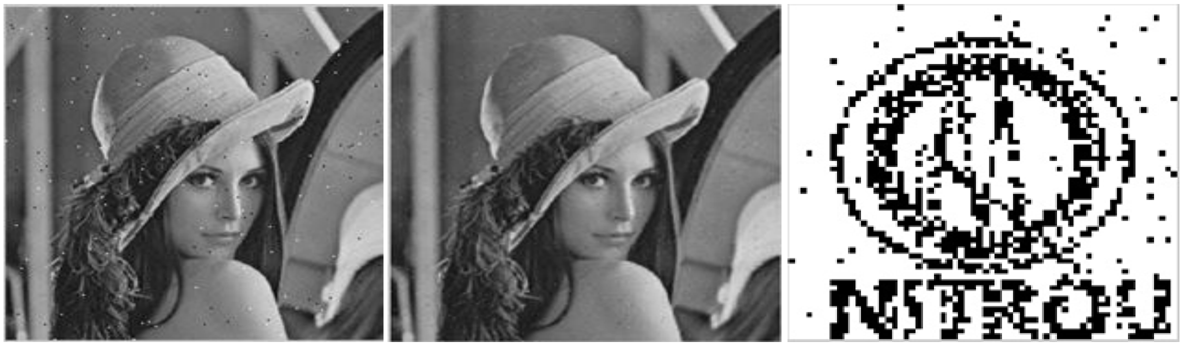


Fig- 7.6 Results of adding impulse noise , then median filtering and the extracted watermark

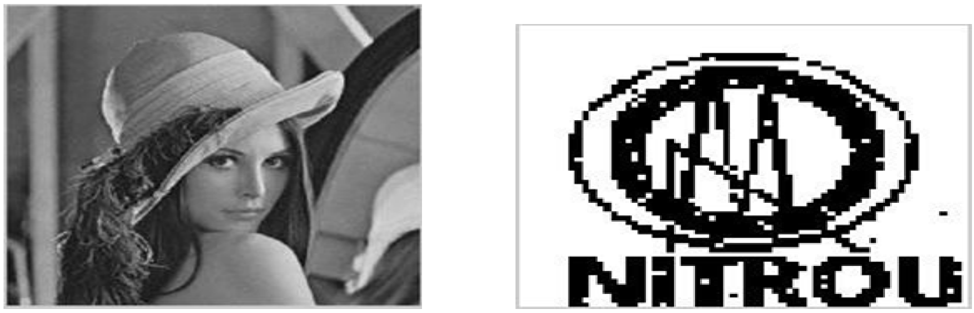


Fig- 7.7 Results after jpeg compression with a multiplication factor 1, and the extracted watermark

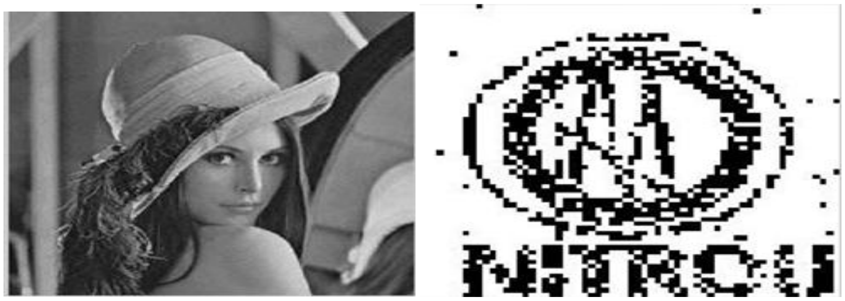


Fig- 7.8 Results after jpeg compression with a multiplication factor 2, and the extracted image

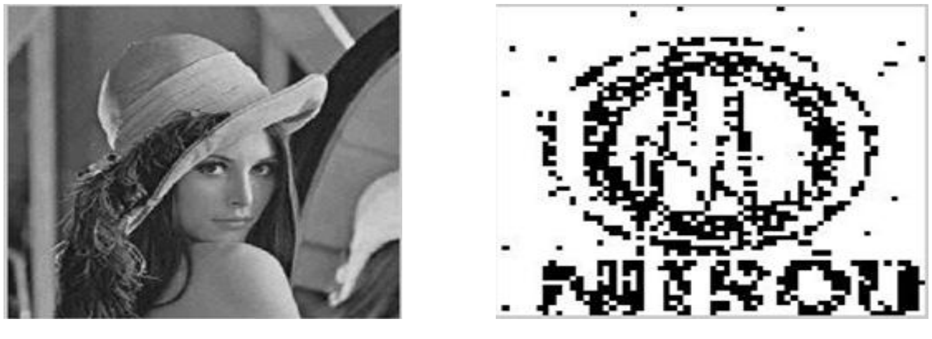


Fig- 7.9 Results after jpeg compression with a multiplication factor 2.5, and the extracted image

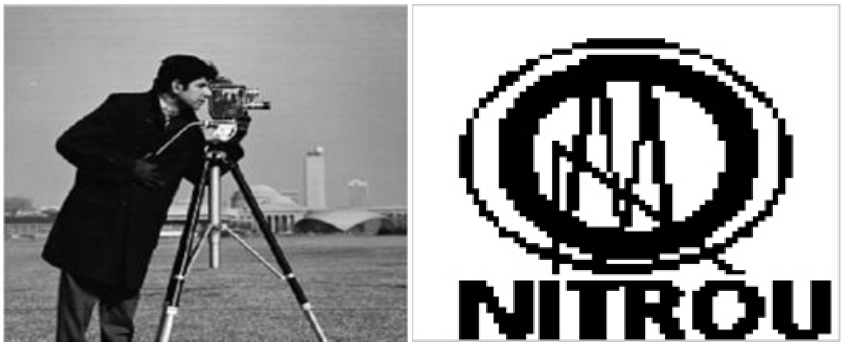


Fig – 7.10 Original Cameraman Image and the Original Binary

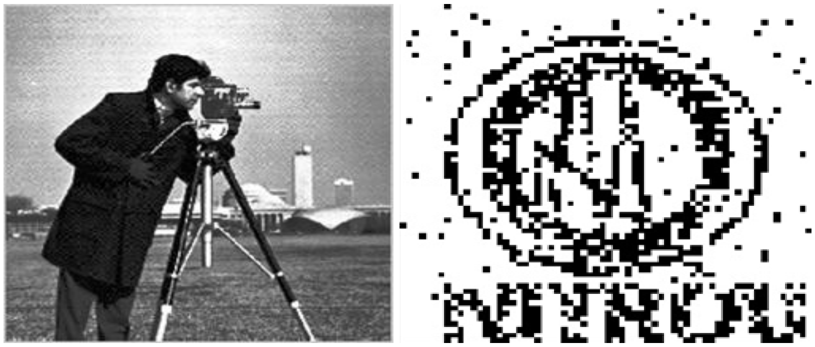


Fig- 7.11 Results of histogram equalization and the extracted watermark

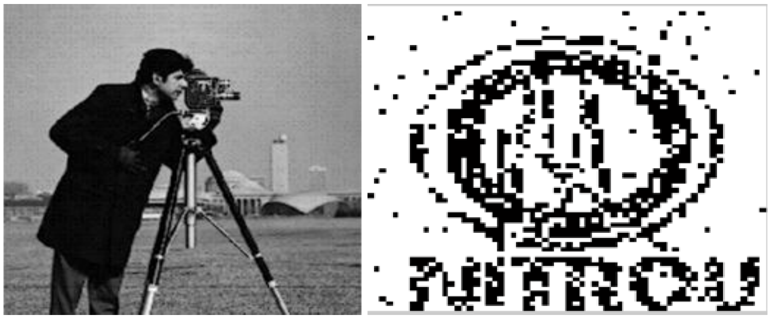


Fig- 7.12 Results of high pass Filtering and the extracted Watermark

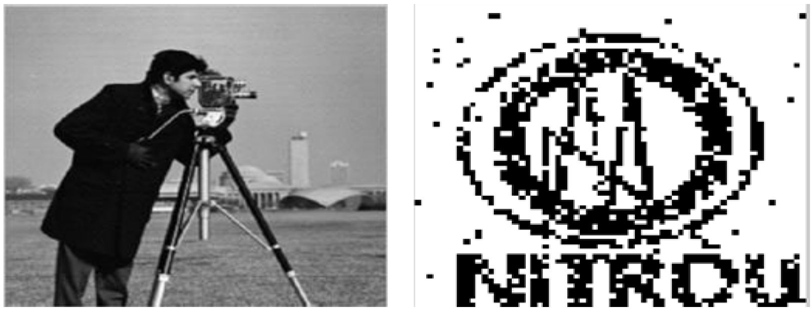


Fig- 7.13 Results of mean Filtering and the extracted Watermark

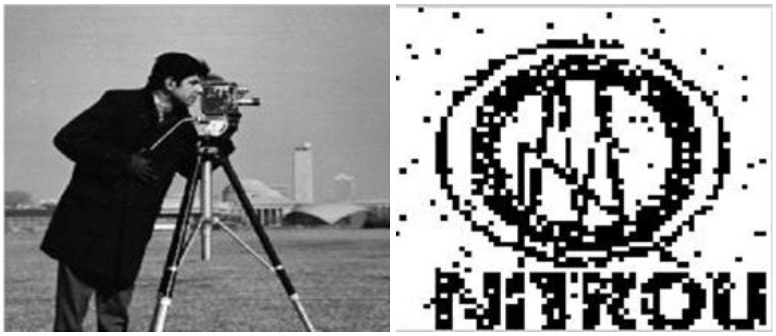


Fig- 7.14 Results of median Filtering and the extracted

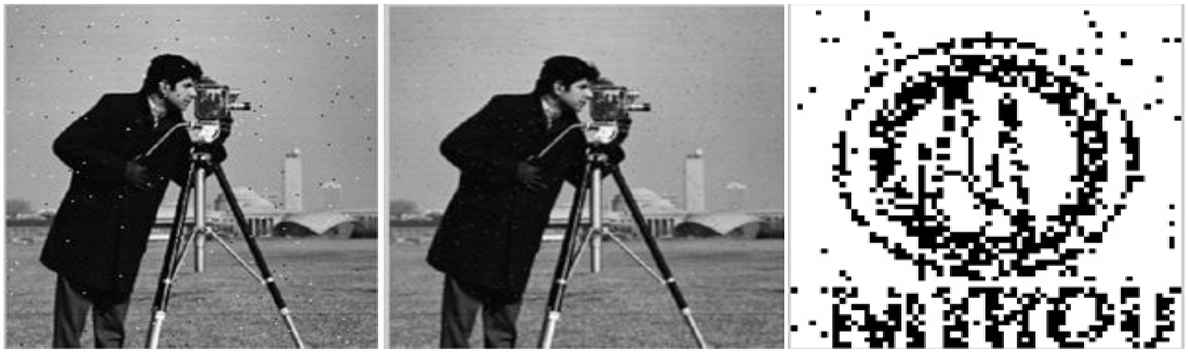


Fig- 7.15 Results of adding impulse noise to the watermarked Image, than doing median filtering operation and the extracted watermark

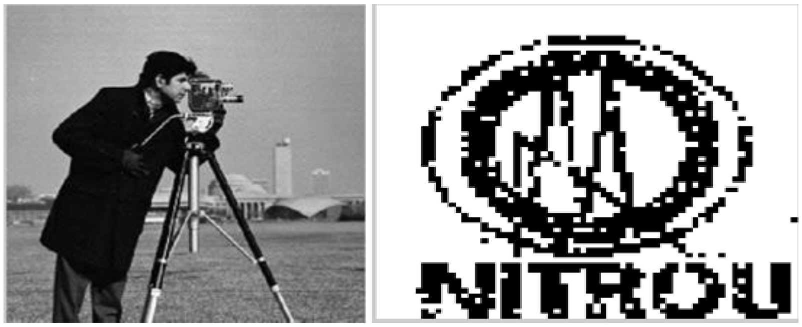


Fig- 7.16 Results after jpeg compression with a multiplication factor 1, and the extracted watermark

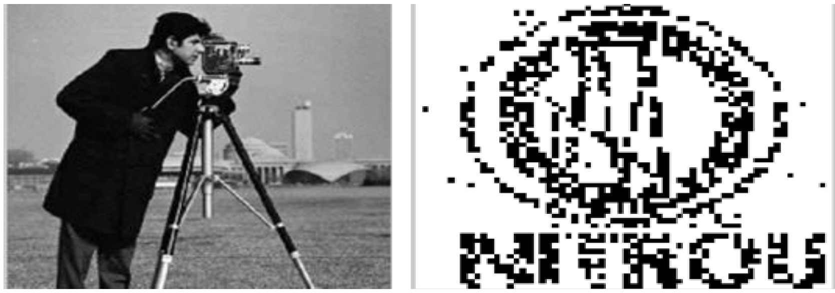


Fig- 7.17 Results after jpeg compression with a multiplication factor 2, and the extracted watermark

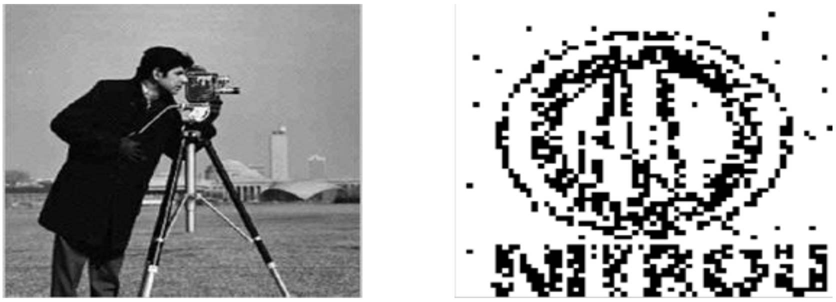


Fig- 7.18 Results after jpeg compression with a multiplication factor 2.5, and the extracted

7.4 Conclusion

From the above results it is concluded that our proposed watermarking algorithm gives better results in terms of PSNR, NC and TAF as compared to the previously proposed watermarking algorithms.

8. HARDWARE IMPLEMENTATION

8.1 Introduction

Today’s applications like digital television broadcasting, Internet protocol television, electronic passport etc demand real time performance. Hence, the need for hardware solutions because software only may not suffice since it is generally slower when compared with hardware.[2] The different modules –DCT, sorting, watermarking, quantization, encoding are designed in VHDL(Xilinx 10.1) and finally integrated into a top module whose input is the host image and output is the encoded data. The code is dumped into the FPGA Virtex II board (xc2vp30). This encoded data is then collected through the RS 232 port using MATLAB. The rest of the processing that is encryption/decryption, decoding, dequantization, IDCT etc are performed in software (MATLAB) and the watermarked image is obtained. The watermark is then extracted to authenticate the image. The hardware details of the individual modules are described below:

8.2 DCT Module

DCT is a computational intensive algorithm and is realized by a large number of additions and multiplications. The use of multipliers is not advisable as they consume high power and large area. So, Distributed Arithmetic (DA) approach is used to avoid the use of multipliers .In distributed arithmetic, one of the inputs is a constant array which can be represented in binary form(1’s and 0’s). So, it basically becomes an addition and shifting operation.[15]

For a 8 point 1-D DCT,

$$F(u) = \frac{1}{2}C(u)\sum_{i=0}^{7} X(i)cos\frac{(2i+1)u\pi}{16} \qquad (8.1)$$

Where C(u)=1/2 for u=0 and C(u)=1 for others.

Using periodicity properties, it can be written as

F(0)=[X(0) + X(1)+ X(2)+ X(3)+ X(4) + X(5) + X(6)+ X(7)]P (8.2a)

F(1)=[X(0)-X(7)]A +[X(1)-X(6)]B +[X(2)-X(5)]C +[X(3)-X(4)]D (8.2b)

F(3)=[X(0) - X(3)- X(4)+ X(7)]M +[X(1) + X(2) + X(5)+ X(6)]N (8.2c)

F(4)= [X(0) - X(1)- X(2)+ X(3)+ X(4) - X(5) - X(6)+ X(7)]P (8.2d)

F(5)=[X(0)-X(7)]C +[X(1)-X(6)](-A) +[X(2)-X(5)]D +[X(3)-X(4)]B (8.2e)

F(6)=[X(0) - X(3)- X(4)+ X(7)]N +[X(1) - X(2) - X(5)+ X(6)](-M) (8.2f)

F(7)=[X(0)-X(7)]D +[X(1)-X(6)](-C) +[X(2)-X(5)]B +[X(3)-X(4)](-A) (8.2g)

Where M= $\frac{1}{2} cos \frac{\pi}{8}$, N= $\frac{1}{2} cos \frac{3\pi}{8}$, P= $\frac{1}{2} cos \frac{\pi}{4}$, A= $\frac{1}{2} cos \frac{\pi}{16}$,

B= $\frac{1}{2} cos \frac{3\pi}{16}$, C= $\frac{1}{2} cos \frac{\pi}{16}$, D= $\frac{1}{2} cos \frac{7\pi}{16}$

The constant cosine coefficients can be represented in binary form. For example, F(1) can be written as,[11]

$$\cdot\left[\frac{1}{2}\cos\left({}^{\pi}\!/_{16}\right) \quad \frac{1}{2}\cos\left({}^{3\pi}\!/_{16}\right) \quad \frac{1}{2}\cos\left({}^{5\pi}\!/_{16}\right) \quad \frac{1}{2}\cos\left({}^{7\pi}\!/_{16}\right)\right]\cdot\begin{bmatrix}(X(0)-x(7))\\(X(1)-x(6))\\(X(2)-x(5))\\(X(3)-x(6))\end{bmatrix}$$

$$=\left[-2^{0} \quad 2^{-1} \quad \cdots \quad 2^{-12}\right]\begin{bmatrix}0&0&0&0\\0&0&0&0\\1&1&1&0\\1&1&0&0\\1&0&0&1\\1&1&0&1\\1&0&1&0\\0&1&1&0\\1&0&1&0\\1&0&0&1\\0&1&0&1\\0&1&0&1\\0&0&1&0\end{bmatrix}\cdot\begin{bmatrix}(X(0)-x(7))\\(X(1)-x(6))\\(X(2)-x(5))\\(X(3)-x(6))\end{bmatrix}$$

If F (1) can be represented as below,

$$F(1)=\begin{bmatrix}-2^{0} & 2^{-1} & \cdots & 2^{-12}\end{bmatrix}\begin{bmatrix}F^{0}(1)\\ F^{1}(1)\\ F^{2}(1)\\ F^{3}(1)\\ F^{4}(1)\\ F^{5}(1)\\ F^{6}(1)\\ F^{7}(1)\\ F^{8}(1)\\ F^{9}(1)\\ F^{10}(1)\\ F^{11}(1)\\ F^{12}(1)\end{bmatrix}$$

Where the powers of F denote the number of right shifting required, the 8 point DCT can be calculated using the adder/subtractor structure in Fig.8.1 and Table 8.1.

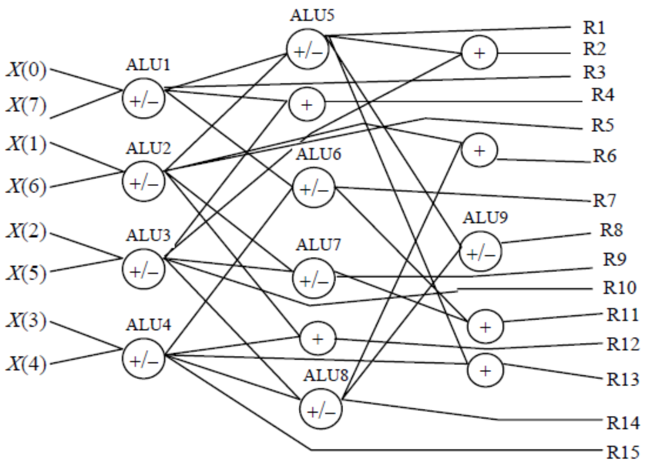
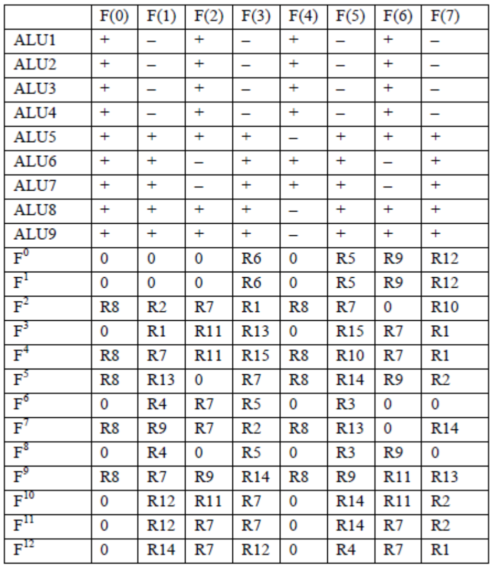


Fig.8.1 Adder/ Subtractor Structure for 8-point DCT [11]

The + and – signs in Table 8.1 indicates the function to be performed by the ALU’s in Fig 8.1.Each DCT coefficient is obtained by shifting the R_i values by the power of F and then summing each column.

Table 8.1

Function of each ALU to calculate the 8-point DCT[11]



	F(0)	F(1)	F(2)	F(3)	F(4)	F(5)	F(6)	F(7)
ALU1	+	–	+	–	+	–	+	–
ALU2	+	–	+	–	+	–	+	–
ALU3	+	–	+	–	+	–	+	–
ALU4	+	–	+	–	+	–	+	–
ALU5	+	+	+	+	–	+	+	+
ALU6	+	+	–	+	+	+	–	+
ALU7	+	+	–	+	+	+	–	+
ALU8	+	+	+	+	–	+	+	+
ALU9	+	+	+	+	–	+	+	+
F^0	0	0	0	R6	0	R5	R9	R12
F^1	0	0	0	R6	0	R5	R9	R12
F^2	R8	R2	R7	R1	R8	R7	0	R10
F^3	0	R1	R11	R13	0	R15	R7	R1
F^4	R8	R7	R11	R15	R8	R10	R7	R1
F^5	R8	R13	0	R7	R8	R14	R9	R2
F^6	0	R4	R7	R5	0	R3	0	0
F^7	R8	R9	R7	R2	R8	R13	0	R14
F^8	0	R4	0	R5	0	R3	R9	0
F^9	R8	R7	R9	R14	R8	R9	R11	R13
F^{10}	0	R12	R11	R7	0	R14	R11	R2
F^{11}	0	R12	R7	R7	0	R14	R7	R2
F^{12}	0	R14	R7	R12	0	R4	R7	R1

8.2.1 Hardware Architecture

The DCT 2-D module is implemented in structural style of modeling with two 1-D DCT modules as its components. It is implemented block wise (8 x 8 blocks). The 8-bit image pixel values are input to the 1-D module row wise one row in 1 clock cycle and the resulting values are stored in registers. After 8 clock cycles, column wise 1-D DCT is calculated which again takes 8 more cycles. Hence the final 2-D DCT takes 16 clock cycles in all.

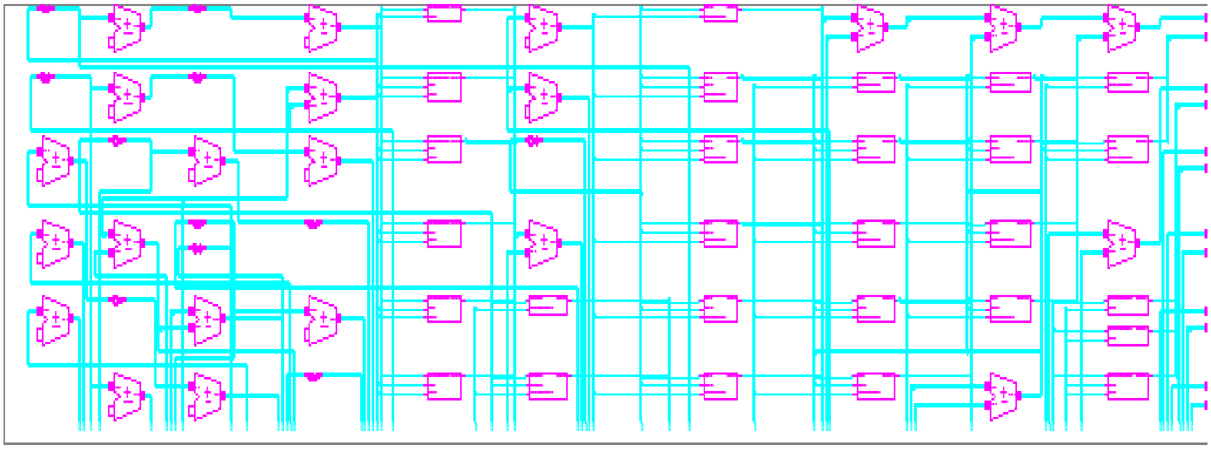


Fig.8.2 RTL Schematic of 8-point 1-D DCT

The simulation results of the 2-D DCT are shown in Fig.8.3.

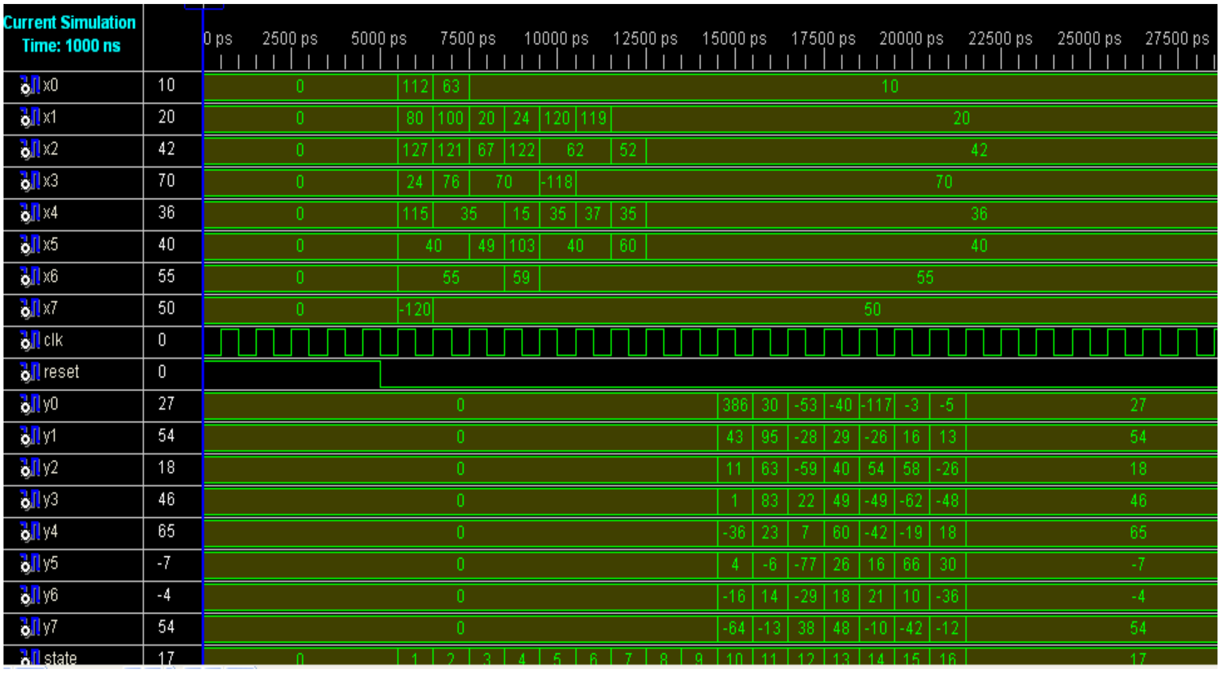

Current Simulation
Time: 1000 ns
0 ps
2500 ps
5000 ps
7500 ps
10000 ps
12500 ps
15000 ps
17500 ps
20000 ps
22500 ps
25000 ps
27500 ps
x0 10 0 112 63 10
x1 20 0 80 100 20 24 120 119 20
x2 42 0 127 121 67 122 62 52 42
x3 70 0 24 76 70 -118 70
x4 36 0 115 35 15 35 37 35 36
x5 40 0 40 49 103 40 60 40
x6 55 0 55 59 55
x7 50 0 -120 50
clk 0
reset 0
y0 27 0 386 30 -53 -40 -117 -3 -5 27
y1 54 0 43 95 -28 29 -26 16 13 54
y2 18 0 11 63 -59 40 54 58 -26 18
y3 46 0 1 83 22 49 -49 -62 -48 46
y4 65 0 -36 23 7 60 -42 -19 18 65
y5 -7 0 4 -6 -77 26 16 66 30 -7
y6 -4 0 -16 14 -29 18 21 10 -36 -4
y7 54 0 -64 -13 38 48 -10 -42 -12 54
state 17 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17

Fig.8.3 Simulation Results of 2-D DCT

The hardware utilization details are shown in Table 8.2.

Table 8.2

Hardware Details of DCT Module

Number of Slices	2442 out of 13696 (17 %)
Number of Slice Flip Flops	803 out of 27392(2%)
Number of 4 Input LUTs	4466 out of 27392(16 %)
Minimum Period	4.301ns
Maximum Frequency	232.519 MHz

8.3 Sorting Module

This module sorts the nine low frequency AC DCT coefficients and returns the median. First, the nine low frequency AC DCT coefficients taken in zigzag order are stored in nine registers. As soon as the first two coefficients are available, they are compared and sorted. This continues for the rest AC coefficients. This kind of sorting is called comparison sort. Pipelining is implemented here to improve throughput that is as soon as the coefficients are available, they are sorted. We do not wait for the entire block DCT to be completed. It takes 9 clock cycles to sort after the row-wise DCT is completed. Effectively, it is just 1 clock cycle more after the DCT 2-D module has been completed since it takes 8 clock cycles to compute the column wise DCT. After the sorting is completed, the middle value is sent to the watermarking module which will be used to calculate the embedding factor.

The RTL Schematic of the Sorting Module is shown in Fig.8.4.

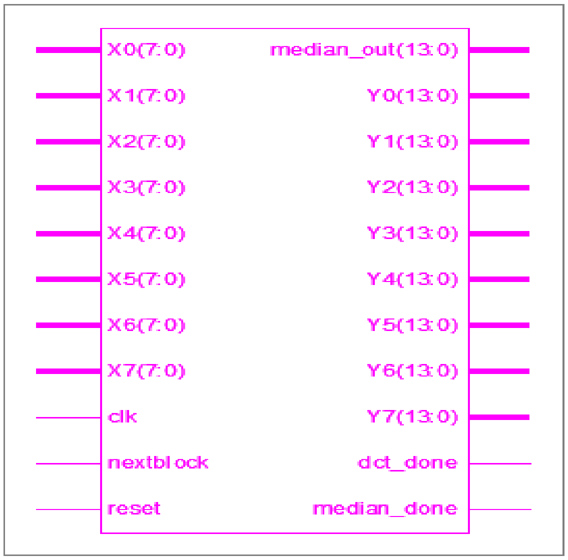


Fig.8.4 RTL Schematic of Sorting Module

The simulation results of the module is shown in Fig.8.5

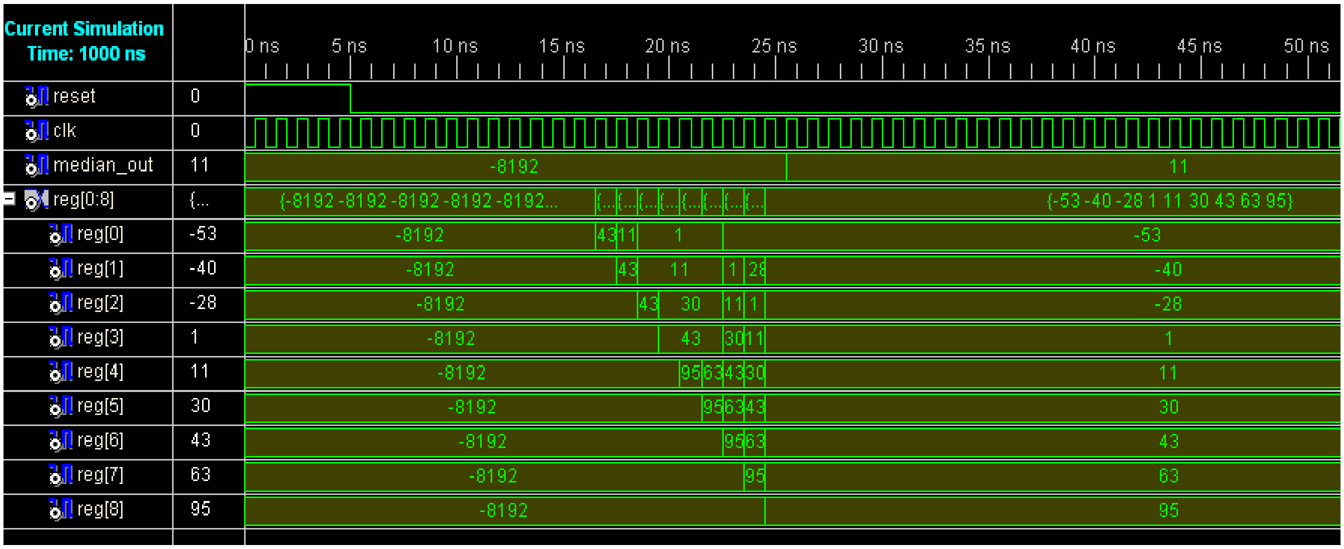


Fig.8.5 Simulation Results of Sorting Module

The hardware utilization details are shown in Table 8.3.

Table 8.3

Hardware Details of Sorting Module

Number of Slices	2982 out of 13696 (21 %)
Number of Slice Flip Flops	1016 out of 27392(3%)
Number of 4 Input LUTs	5504 out of 27392(20%)
Minimum Period	31.031ns
Maximum Frequency	32.226 MHz

8.4 Watermarking Module

This module takes the median of each block and embeds the watermark in one coefficient of each block and outputs the watermarked DCT coefficients.

Two ROMs (Read Only Memory) are used,

1. One to store the binary watermark (64 X 64) of depth 4096 each 1 bit wide.
2. The other to store the random values to determine which coefficient in each block is to be embedded. Its depth is equal to the number of blocks (512 X512)/64 =4096 each 2 bits wide. Since four coefficients(with coordinates (1,1),(1,2),(2,1),(2,2)) are chosen to embed the watermark,2 bits are required. Only one of the four is chosen according to the value specified in the ROM. These values are generated using a pseudo random number generator.

The DCT coefficients are stored in zigzag order and then after the median has been calculated, the watermark is embedded. For each block, the watermark value is checked and then according to the random value, it is embedded according to the formula,

DCT (4/5/6/7) = DCT (4/5/6/7) + median*scale factor (8.3)

The address of both the ROMs is incremented and the process is repeated for all blocks.

After the median has been calculated, it takes only 1 clock cycle to embed in 1 block.

The hardware details are shown in Table 8.4.

As can be seen from the table, two BRAMs are used as expected and other hardware resources are well within limits.

Table 8.4

Hardware Utilization of Watermarking Module

Number of Slices	3171 out of 13696 (23 %)
Number of Slice Flip Flops	1944 out of 27392(7 %)
Number of 4 Input LUTs	5855 out of 27392(21 %)
Number of BRAMs	2 out of 136(1%)
Minimum Period	31.031ns
Maximum Frequency	32.226 MHz

8.5 Quantization Module

This module takes the DCT outputs and divides them by the Quantization table to give the quantized coefficients.

A ROM is used to store the Quantization table values. The reciprocal of the values are stored

So that it can be directly multiplied with the DCT coefficients. The DCT outputs are stored in registers. A RAM (Random Access Memory) is used to store the quantized coefficients. After DCT is complete, one by one values are accessed and the quantized coefficient obtained by multiplying the register and the ROM value. The address of the register, ROM and RAM are incremented every clock cycle. Thus, it takes 64 clock cycles in all for one block.

The RTL schematic is shown in Fig.8.6.

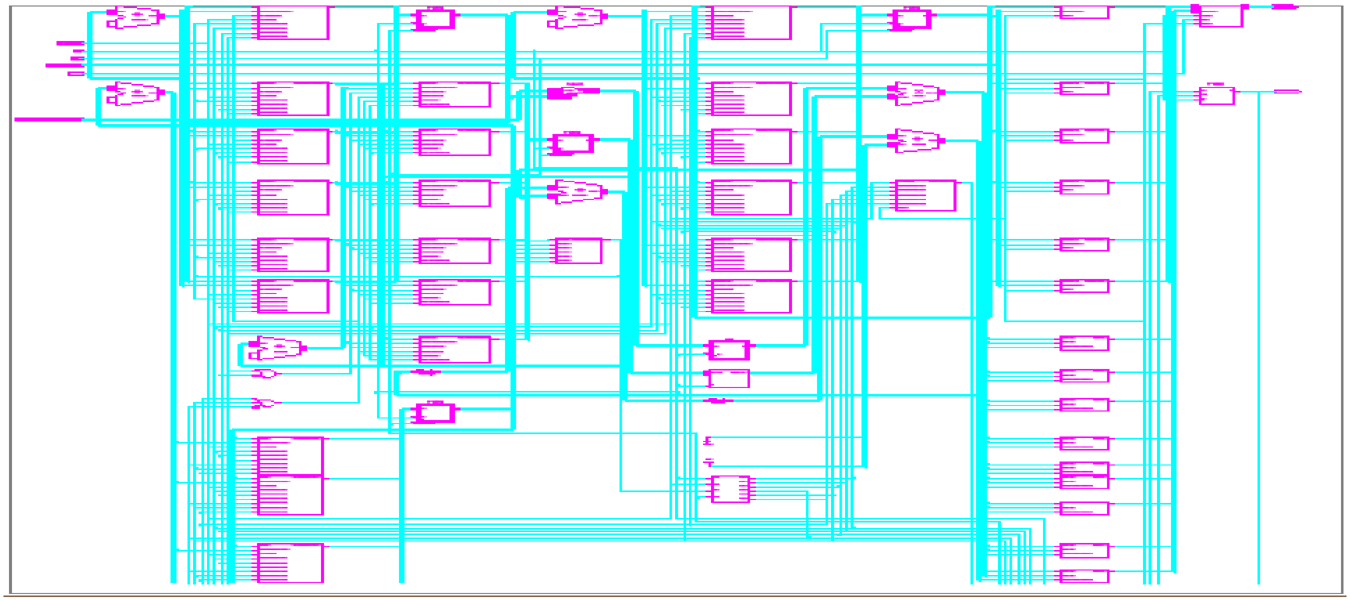


Fig.8.6 RTL Schematic of Quantization Module

The simulation results are shown in Fig.8.7

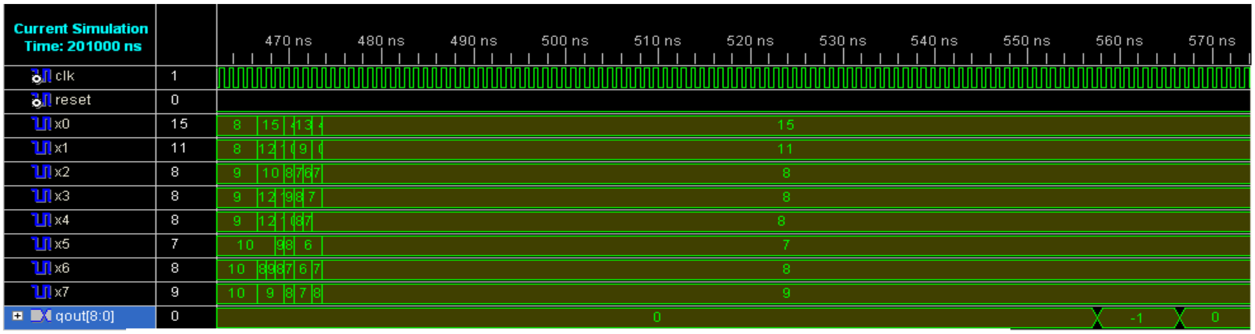


Fig.8.7 Simulation Results of Quantization Module

The hardware details are shown in Table 8.5.

Table 8.5

Hardware Details of Quantization Module

Number of Slices	309 out of 13696 (2 %)
Number of Slice Flip Flops	52 out of 27392(0%)
Number of 4 Input LUTs	613 out of 27392(2 %)
Number of BRAMs	1 out of 136(0%)
Minimum Period	8.318ns
Maximum Frequency	120.218 MHz

8.6 Encoding Module

This module takes the quantized coefficients as inputs and encodes them using Huffman encoding. Huffman coding is a variable length coding used to remove coding redundancy. Use of Huffman code tables makes the hardware implementation simple and high performing. [13]

Initially, the Huffman code tables for AC and DC coefficients are stored separately in memory. Then the Category is selected. The DC coefficient difference base code is taken from the DC base code table. The DC base code is extended with the binary value of DC difference coefficient. The AC coefficient base code is brought from the AC base code table. The AC base code is extended with the binary value of AC coefficient. It is repeated until all the AC coefficients are encoded. The number of clock cycles used is variable for every block depending on the values of the quantised coefficients.

The RTL schematic is shown in Fig.8.8.

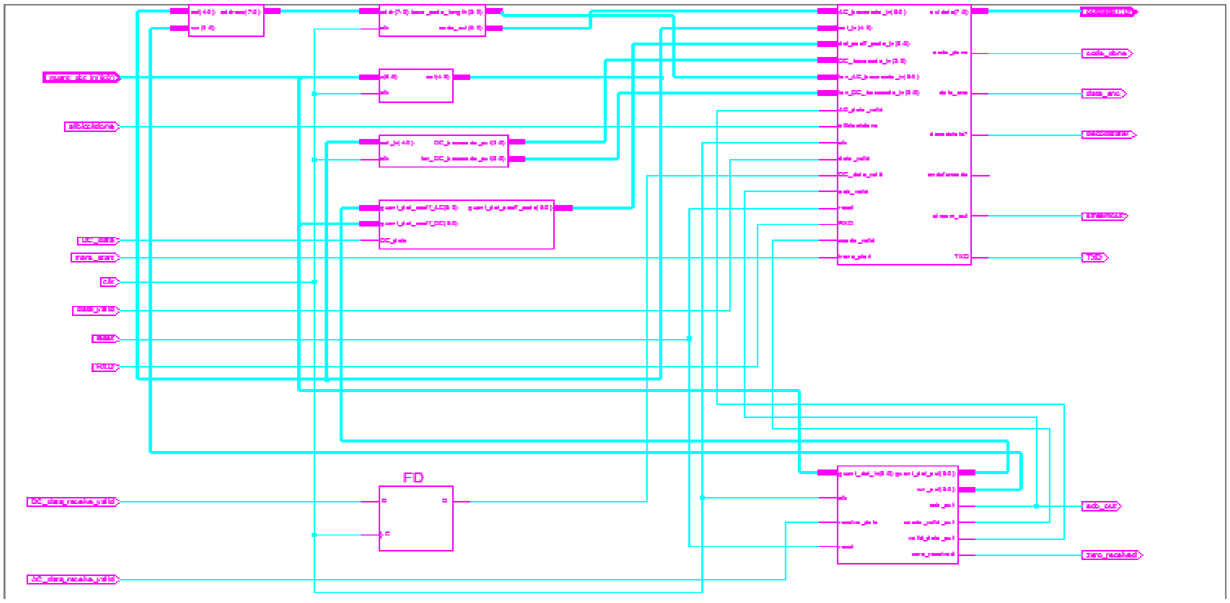


Fig.8.8 RTL Schematic of Encoding Block

The simulation results are shown in Fig.8.9

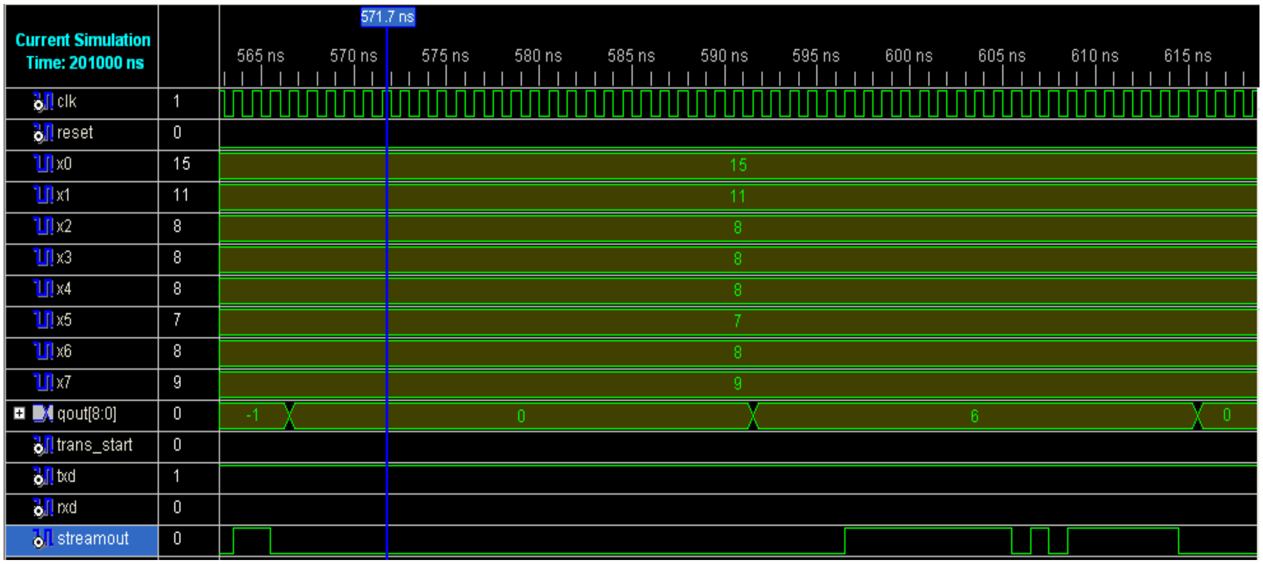


Fig.8.9 Simulation Results of Encoding Block

The hardware utilisation details are shown in Table 8.6

Table 8.6

Hardware Details of Encoding Module

Number of Slices	248 out of 13696 (1 %)
Number of Slice Flip Flops	191 out of 27392(0%)
Number of 4 Input LUTs	461 out of 27392(1 %)
Number of BRAMs	15 out of 136(11%)
Minimum Period	5.324ns
Maximum Frequency	187.817 MHz

8.7 Encryption Module

Its hardware implementation is done using AES (Advanced Encryption Algorithm). It will initially take 16 bytes key and 16 bytes data. The whole AES module has been designed using the structural model. Its individual sub modules are sub bytes, shift row, mixed column.[9] The initial 16 byte data and 16 byte key are xored.

(16 byte plaintext) XOR (16 byte key) = sig

This sig is sent to the sub bytes module which will have s_box one of its components. This coefficient file of s_box table in the form of .coe file is stored in ROM of width 8 bits and depth 256 generated by IP CORE. Based on the input to the sub bytes module the s_box takes it as an address and gives the corresponding value as the output of this sub byte module. The output of the sub byte module is input to the shift rows. Shift row module has been made clock independent. This shifting has been done only assigning the input to its correct position to the output. The output from the shift row is the input to the mixed column module. In mixed column operations different columns are multiplied with rotating version of a number. After this the first round of data is generated. At the same time the key expansion operations are also done. The initial key is sent through these operations. Key expansion output is than xored with the previously generated data cipher. Similar procedures are followed for 10 rounds. That whole process is done in pipelining manner to increase the throughput of the system. After ten rounds one set of cipher is generated. This procedure will be repeated for the complete image data. It takes 21 clock cycles to generate one 16 byte cipher. This module has been designed in Xilinx HDL using VHDL. It has not been integrated with the previously designed secured digital camera model. But the user has an option that it can be integrated by sending the output of the encoding block as the input to the encryption block.

The RTL schematic is shown in Fig.8.10.

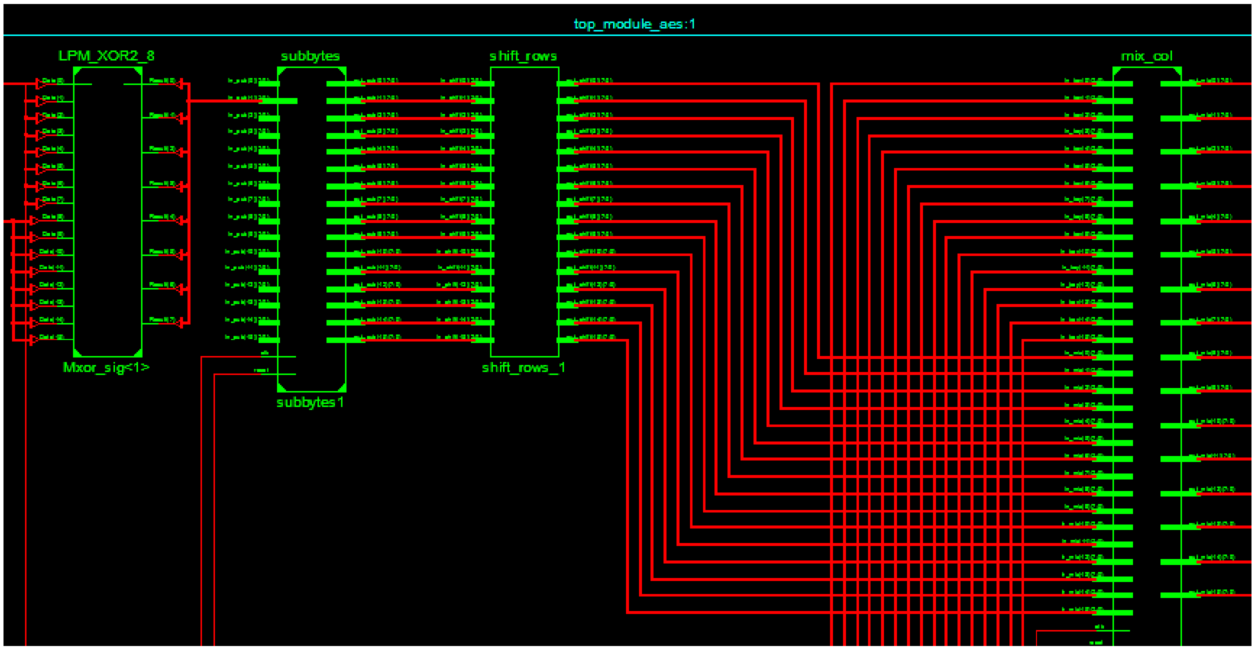


Fig.8.10 RTL Schematic of the Encryption Module

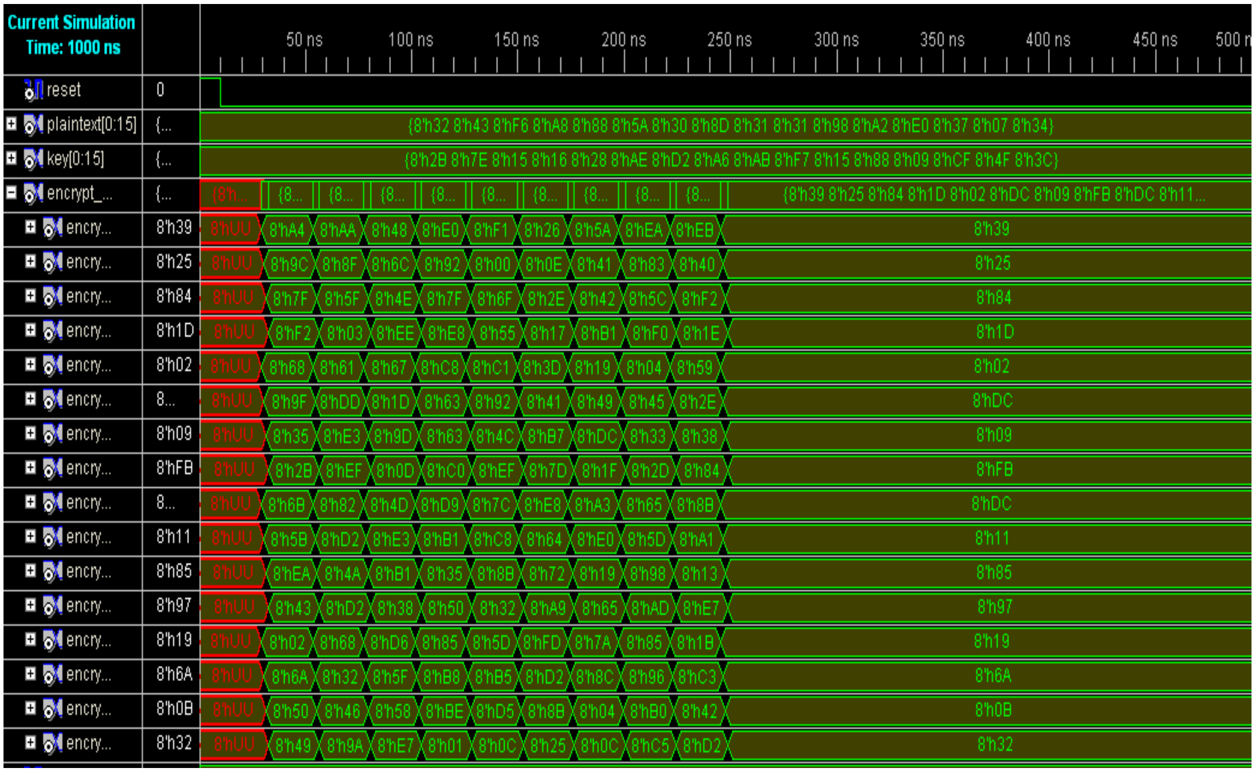

Current Simulation Time: 1000 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500 n
reset 0
plaintext[0:15] {...
{8'h32 8'h43 8'hF6 8'hA8 8'h88 8'h5A 8'h30 8'h8D 8'h31 8'h31 8'h98 8'hA2 8'hE0 8'h37 8'h07 8'h34}
key[0:15] {...
{8'h2B 8'h7E 8'h15 8'h16 8'h28 8'hAE 8'hD2 8'hA6 8'hAB 8'hF7 8'h15 8'h88 8'h09 8'hCF 8'h4F 8'h3C}
encrypt_... {...
{8'h... {8... {8... {8... {8... {8... {8... {8... {8... {8... {8'h39 8'h25 8'h84 8'h1D 8'h02 8'hDC 8'h09 8'hFB 8'hDC 8'h11...
encry... 8'h39 8'hUU 8'hA4 8'hAA 8'h48 8'hE0 8'hF1 8'h26 8'h5A 8'hEA 8'hEB 8'h39
encry... 8'h25 8'hUU 8'h9C 8'h8F 8'h6C 8'h92 8'h00 8'h0E 8'h41 8'h83 8'h40 8'h25
encry... 8'h84 8'hUU 8'h7F 8'h5F 8'h4E 8'h7F 8'h6F 8'h2E 8'h42 8'h5C 8'hF2 8'h84
encry... 8'h1D 8'hUU 8'hF2 8'h03 8'hEE 8'hE8 8'h55 8'h17 8'hB1 8'hF0 8'h1E 8'h1D
encry... 8'h02 8'hUU 8'h68 8'h61 8'h67 8'hC8 8'hC1 8'h3D 8'h19 8'h04 8'h59 8'h02
encry... 8... 8'hUU 8'h9F 8'hDD 8'h1D 8'h63 8'h92 8'h41 8'h49 8'h45 8'h2E 8'hDC
encry... 8'h09 8'hUU 8'h35 8'hE3 8'h9D 8'h63 8'h4C 8'hB7 8'hDC 8'h33 8'h38 8'h09
encry... 8'hFB 8'hUU 8'h2B 8'hEF 8'h0D 8'hC0 8'hEF 8'h7D 8'h1F 8'h2D 8'h84 8'hFB
encry... 8... 8'hUU 8'h6B 8'h82 8'h4D 8'hD9 8'h7C 8'hE8 8'hA3 8'h65 8'h8B 8'hDC
encry... 8'h11 8'hUU 8'h5B 8'hD2 8'hE3 8'hB1 8'hC8 8'h64 8'hE0 8'h5D 8'hA1 8'h11
encry... 8'h85 8'hUU 8'hEA 8'h4A 8'hB1 8'h35 8'h8B 8'h72 8'h19 8'h98 8'h13 8'h85
encry... 8'h97 8'hUU 8'h43 8'hD2 8'h38 8'h50 8'h32 8'hA9 8'h65 8'hAD 8'hE7 8'h97
encry... 8'h19 8'hUU 8'h02 8'h68 8'hD6 8'h85 8'h5D 8'hFD 8'h7A 8'h85 8'h1B 8'h19
encry... 8'h6A 8'hUU 8'h6A 8'h32 8'h5F 8'hB8 8'hB5 8'hD2 8'h8C 8'h96 8'hC3 8'h6A
encry... 8'h0B 8'hUU 8'h50 8'h46 8'h58 8'hBE 8'hD5 8'h8B 8'h04 8'hB0 8'h42 8'h0B
encry... 8'h32 8'hUU 8'h49 8'h9A 8'hE7 8'h01 8'h0C 8'h25 8'h0C 8'hC5 8'hD2 8'h32

Fig.8.11 Simulation Results of Encryption.

The simulation results are shown in Fig.8.11.

8.8 Top Module

The DCT, sorting, watermarking, quantization and encoding blocks are integrated into a top module whose input is the host image and output is the encoded data.

The RTL schematic and simulation results are shown in Fig.8.12 and Fig.8.13.

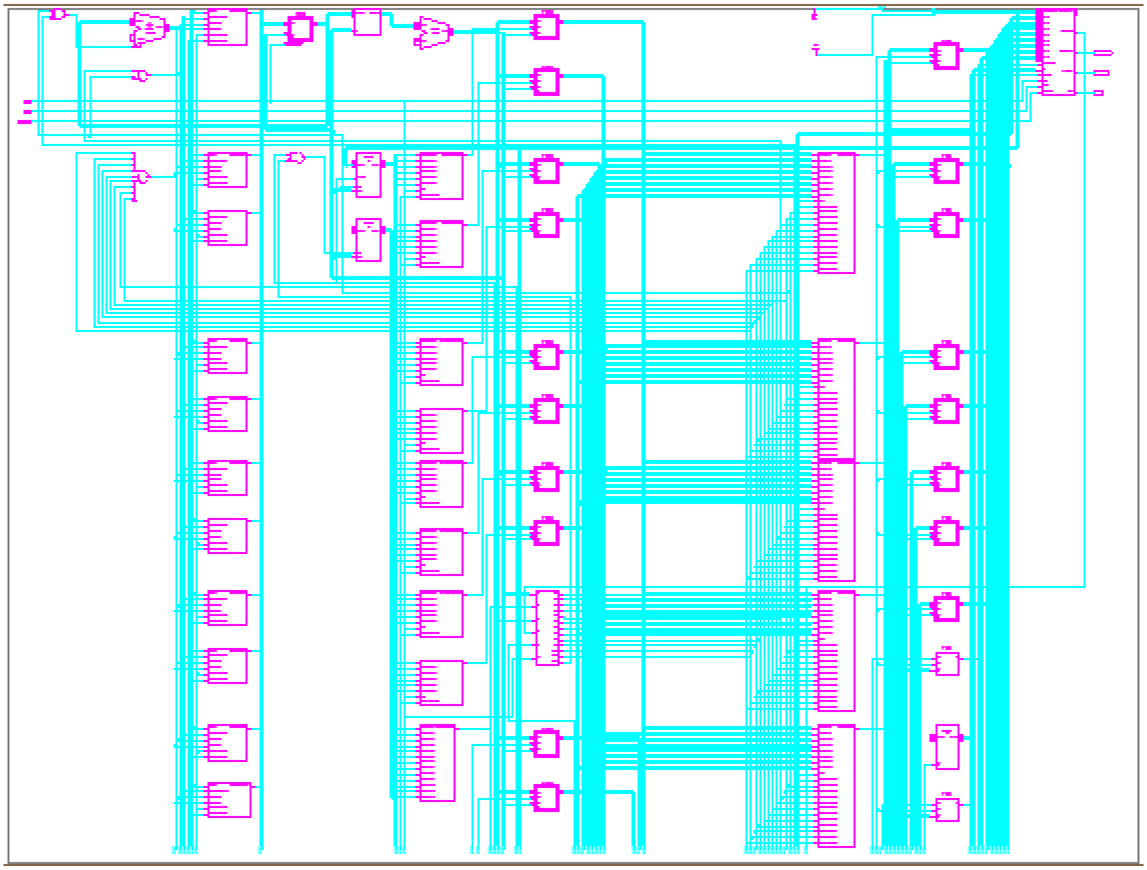
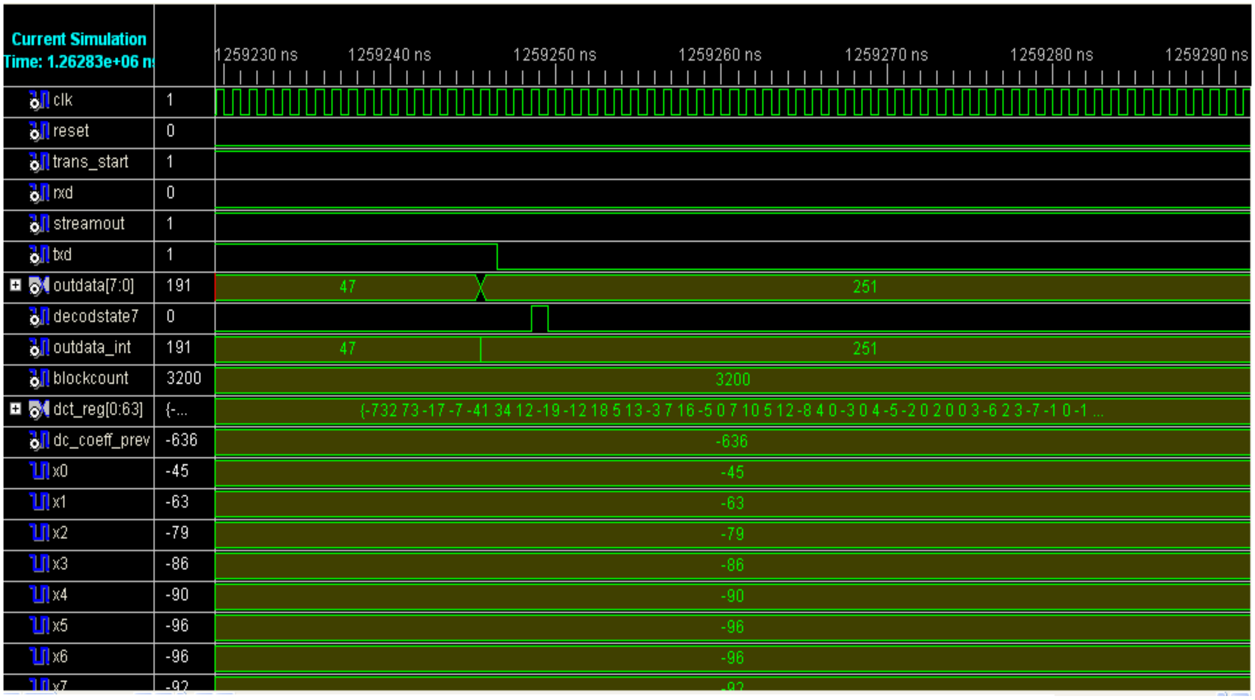


Fig.8.12 RTL Schematic of the Top Module



clk	1	
reset	0	
trans_start	1	
rxd	0	
streamout	1	
txd	1	
outdata[7:0]	191	47 251
decodstate7	0	
outdata_int	191	47 251
blockcount	3200	3200
dct_reg[0:63]	{-...	{-732 73 -17 -7 -41 34 12 -19 -12 18 5 13 -3 7 16 -5 0 7 10 5 12 -8 4 0 -3 0 4 -5 -2 0 2 0 0 3 -6 2 3 -7 -1 0 -1 ...
dc_coeff_prev	-636	-636
x0	-45	-45
x1	-63	-63
x2	-79	-79
x3	-86	-86
x4	-90	-90
x5	-96	-96
x6	-96	-96
x7	-92	-92

Fig.8.13 Simulation Results of Top Module

The hardware utilization details are shown in Table 8.7.

Table 8.7

Hardware Details of Top Module

Number of Slices	4556 out of 13696 (33 %)
Number of Slice Flip Flops	2834 out of 27392(10%)
Number of 4 Input LUTs	8007 out of 27392(29 %)
Number of BRAMs	120 out of 136(88%)
Minimum Period	31.031ns
Maximum Frequency	32.226 MHz

The code for the Top Module is dumped into the FPGA and the encoded data collected through the RS 232 port. The rest of the processing is done in MATLAB and the watermarked image is obtained. The watermark can then be extracted to authenticate the image.

8.9 Conclusion

The hardware architectures for the different modules are described with details of the hardware utilization and timing analysis. It is found that the resources used are within limits and easily implementable in hardware for real time performance.

9. CONCLUSION

In this thesis, new watermarking algorithm is proposed. This includes the calculation of embedding factor using the median value of low frequency DCT coefficients. Watermark is embedded in the medium frequency DCT region of an 8×8 block using pseudorandom number generator and distributes the watermark arbitrarily in the whole image. As median value is used it provides good result for all kind of images. The encryption unit and the watermarking unit provide two layer of security. The median based proposed algorithm gives better result in terms of PSNR, NC and TAF for various image manipulations operations as compared to the earlier proposed permutation based and CRT based watermarking algorithm. This is computationally less complex hence provides easier hardware implementation as compared to other algorithm. So the proposed algorithm is robust. Its hardware implementation provides efficient real time performance.

References

[1]Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing**”**, *Second Edition, Publisher: Pearson Education,* 2002.

[2]Saraju P. Mohanty, “A secure digital camera architecture for integrated real-time digital rights management”, *IEEE Journal of Systems Architecture,* Vol-55, pp. 468-480, 2009.

[3] Syed Ali Khayam, “The Discrete Cosine Transform (DCT): Theory and Application”, ECE 802 – 602: Information Theory and Coding,2003

[4] S. P. Mohanty, K. R. Ramakrishnan, M. S. Kankanhalli, “A DCT Domain Visible Watermarking Technique for Images”, in: Proceedings of the IEEE International Conference on Multimedia and Expo, 2000, pp. 1029–1032.

[5]Anup Sarma, Soubhagya Sutar, V.K Sharma, K.K Mahapatra, “An ASIP for Image Enhancement Applications in Spatial Domain using LISA”, *IEEE International Conference, Jadavpur Univ.,* 2011

[6] Jagdish C. Patra, Jiliang E. Phua, Cedric Bornand, “A novel DCT domain CRT-based watermarking scheme for image authentication surviving JPEG compression”, *Digital Signal Processing,* Vol 20, Issue 6, December 2010, pp. 1597–1611

[7] **C**hiou-Ting Hsu and Ja-Ling Wu, *Senior Member, IEEE ,“*Hidden digital watermarks in images ”, *IEEE Transactions on Image Processing,* Vol. 8,No. 1,January 1999,pp. 58-68.

[8]“Advanced Encryption Standard (AES)”, Federal Information Processing Standards Publication 197 ,November 26 ,2001

[9]Xinmiao Zhang and Keshab K. Parhi, “Implementation Approaches for the Advanced Encryption Standard Algorithm”, IEEE Circuits Syst. Mag., Vol. 2, no. 4, 2002, pp. 24–46.

[10] Atul M. Borkar, R. V. Kshirsagar, M. V Vyawahare, “Design of AES algorithm using FPGA”, *International Conference on Advanced Computing, Communication and Networks* , 2011

[11] Peng Chungan, Cao Xixin, Yu Dunshan, Zhang Xing, “A 250MHz optimized distributed architecture of 2D 8x8 DCT,” 7th International Conference on ASIC, pp. 189 – 192, Oct. 2007.

[12] Wallace, G.K., “The JPEG Still Picture Compression Standard”, *IEEE Transactions on Consumer Electronics,* Vol . 38,No. 1,February 1992,pp. xviii-xxxiv

[13] Luciano Volcan, Agostini, Ivan Saraiva Silva and Sergio Bampi, “Multiplierless and fully pipelined JPEG compression soft IP targeting FPGAs,” *Microprocessors and Microsystems*, vol. 31(8), Dec. 2007, pp.487-497.

[14] David L. McLaren, D. Thong Nguyen, “Removal of subjective redundancy from DCT coded images,”*IEEE Proceedings I, Communications, Speech and Vision,* Vol.3, pp.482 – 485, 2001.

[15] S. A. White, “Applications of distributed arithmetic to digital signal processing: a tutorial review,”*IEEE ASSP Magazine*, vol.6, no.3, Jul.1989, pp.4-19.

,