FPGA IMPLEMENTATION OF WATERMARKING USING BIT PLANE CODING

A PROJECT REPORT

Submitted by

ARUNKUMAR.N	71206106009
MALARVANAN.M	71206106029
MANIKANDAN.E	71206106031
SURESH.M	71206106054

in partial fulfillment for the award of the degree

of

P-3056

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY : CHENNAI 600 025

APRIL 2010

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"FPGA IMPLEMENTATION OF WATERMARKING USING BIT PLANE CODING"** is the bonafide work of **"ARUNKUMAR.N, MALARVANAN.M, MANIKANDAN.E, SURESH.M"** who carried out the project work under my supervision.

13/4/10

SIGNATURE

Dr. RAJESWARI MARIAPPAN

HEAD OF THE DEPARTMENT

Electronics and
Communication Engineering
Kumaraguru College of Technology
Coimbatore – 641006.

13/4/10

SIGNATURE

Ms. S.SASIKALA

SUPERVISOR

SENIOR LECTURER

Electronics and
Communication Engineering
Kumaraguru College of Technology
Coimbatore – 641006.

The candidates with university register numbers **71206106009, 71206106029, 71206106031, 71206106054** were examined by us in the project viva-voce examination held on ..16..4..10.....

16-04-2010

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

We wish to express sincere thanks and deep sense of gratitude to our respected Director **Dr.J.Shanmugam,Ph.D,** for permitting us to undertake this project.

We are greatly indebted to our beloved Principal **Dr.S.Ramachandran,Ph.D.,** who has been the backbone of all our deeds.

We profusely thank **Dr.Rajeswari Mariappan, M.E., Ph.D., B.Tech.Ed., FIE., MISTE** Head of the Department, Department of Electronics and Communication Engineering, Kumaraguru College of Technology for leading a help hand in this project.

We are highly grateful to our beloved Project Coordinator **Ms.A.Vasuki, M.E., Assistant Professor**, and project guide **Ms.S.Sasikala, M.Tech., Senior Lecturer**, ECE department for their valuable guidance, timely helps, constant encouragement and advice rendered throughout the project period for complete successful completion of the project.

We are also grateful to the faculty members of Electronics and Communication Engineering Department, who have helped us in innumerable ways.

We also thank our parents without whom we could not have come so far and friends for their timely help that culminated as good in end.

ABSTRACT

The use of digital images is increasing rapidly with the expansion of multimedia broadcasting, networked databases, electronic publishing and so on. The transmission of data through internet with security is a challenging one and many techniques are used for security purposes. Here, the data to be secured is hided into an image using watermarking.

In this project, we propose a image data hiding algorithm based on integer wavelet transform that can hide data into the original image, and it is implemented using Xilinx FPGA. The data can be retrieved or extracted without any distortion. This algorithm hides data into least significant bit-plane(s) of the integer wavelet transform coefficients in the HH frequency sub band. Here the data to be hided is the information about the image or a part of the host image which can be used for further processing of image.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE **PAGE NO.**

1. INTRODUCTION

With the increasing use of internet and effortless copying, tempering and distribution of digital data, copyright protection for multimedia data has become an important issue. Digital watermarking emerged as a tool for protecting the multimedia data from copyright infringement. In digital watermarking an imperceptible signal "mark" is embedded into the host image, which uniquely identifies the ownership. After embedding the watermark, there should be no perceptual degradation. These watermarks should not be removable by unauthorized person and should be robust against intentional and unintentional attacks. Different watermarking techniques have already been published in the literature.

Watermarking techniques can be broadly classified into two categories: such as spatial domain methods and transform domain methods. Spatial domain methods are less complex as no transform is used, but are not robust against attacks. Transform domain watermarking techniques are more robust in comparison to spatial domain methods. This is due to the fact when image is inverse wavelet transformed watermark is distributed irregularly over the image, making the attacker difficult to read or modify. Among the transform domain watermarking techniques discrete wavelet transform (DWT) based watermarking techniques are gaining more popularity because DWT has a number of advantages over other transform such as progressive and low bit-rate transmission, quality scalability and region-of-interest (ROI) coding demand more efficient and versatile image coding that can be exploited for both, image compression and watermarking applications. The compression standard JPEG2000 is based on the discrete wavelet

transform (DWT) to meet the requirements. Therefore, we think it is imperative to consider the wavelet transform domain for watermarking applications.

Integer DWT is a new emerging technique in which the wavelet coefficients are ceilinged (round off to next integer value). Because of this the registers used for process are reduced. So, the complexity of calculation is decreased and the system becomes a simplest one.

In our project, systemc language is chosen as to describe watermarking implementation. Systemc is an emerging language for hardware description and it is supported by computer aided design software of all major FPGA device vendors.

The image used for watermarking is not directly given to FPGA. A raw image is created from digital image using MATLAB and that raw image is used for watermarking.

The verified systemc code is then synthesized and optimized using FPGA. The obtained net list is then exported to Xilinx foundation series to create the implementation. The Xilinx Platform Studio (XPS) tool is used for simulation.

2. WHAT IS WATERMARKING?

2.1. DEFNITION:

A digital watermark is a suitable signal embedded in host data causing an imperceptible change to the host data – be it images, video or audio. The easiest to implement is an image watermark, which invisibly embeds a message in an image using a password.

A digital watermark can be an identification code carrying out information (an author's signature, a company logo, etc.) about the copyright owner, the creator of the work, the authorized consumer and so on. It is permanently embedded into digital data for copyright protection and for checking if the data has been modified. Watermark embeds an imperceptible signal or perceptible signal into data such as audio, video and messages, for variety of purposes, including captioning, robustness and copyright control. Early work identified redundant properties of an image (or its coding) that can be modified to encode watermark information. The early emphasis was on hiding data, since the end visional applications were not concerned with signal distortions or intentional tampering that might remove watermark. Watermarks are increasingly used for purposes of copyright control.

Watermarking is applicable to any type of objects, and abstracts to any messages, given a message M of N bits, one seeks to embed smaller message 'm',

or watermark 'w', of n bits into the original image as way of identifying 'm' is found, the watermark w' can be readily extracted, thus establishing the provenance.

"A watermark is in digital code, typically contains information about the origin status or recipient of lost data".

2.2. WATERMARK REQUIREMENTS:

a) **Perceptual Transparency**: In most applications, the watermark inserted should not affect the quality of the cover image or data and hence remain undetectable. The watermark should go unnoticed as long as the data is not compared with the original data. This requirement also arises from the fact that perceptible signals are much easier to remove and also do not have the built in advantage of stealth.

b) **Robustness**: Robustness is a measure of the ability of the embedding algorithm to introduce the watermark n such a way that it is retained in the image despite several stages of image processing. The image may be filtered (high-pass or low-pass or median) rotated, translated, cropped, scaled etc. as part of image processing. A good watermarking algorithm embeds the watermark in the spatial or frequency region of the image, which would be least affected by such processing. Good correlation is possible between the recovered watermark and the original watermark in spite of noise errors introduced in it by processing. There is a special class of watermarks called "fragile" watermarks, which are intentionally made non-robust. These are intended for authentication of original material rather

than tracing it back to a source after being processed. A fragile watermark is lost with the slightest of image processing since such processing alters the image in a manner not intended for by the original owner of the material. "semi-fragile' watermarks are able to survive standard unintentional image processing such as image compression for storage.

c) **Security**: Security of a watermarking technique can be judged the same way as with encryption techniques. Assuming that unauthorized parties know the algorithm used for the embedding, the security of the algorithm lies in the selection of key. Thus the algorithm is truly secure if knowing the exact algorithm to embed and extract data does not help an unauthorized party in actually recovering the data from the watermarked image.

d) **Payload of watermark**: The amount of information that can be stored in watermark depends on the application. For example, in copy protection purposes, a payload of one bit is more than sufficient. The latest proposal for audio watermarking standard specifies that an audio watermark be at least 20 bit per second. (This is however almost impractical and so will be reduced to only few bits). For intellectual copyrights such as ISBN or ISRC a length of 60-70 bits would be sufficient. "Watermarking Granularity" is a term used to refer to the number of bits that are actually needed to represent the entire watermark in the image. Generally for video information the watermarking information can be spread over a few frames. Although this decreases the robustness, this approach still suffices for most applications.

e) **Oblivious Vs Non-oblivious**: In applications such as copyright protection and data monitoring the watermark extraction algorithms can use the original unwatermarked data to find the watermark. This is called non-oblivious

watermarking. In other applications, such as copy protection and indexing the watermark extraction algorithms cannot access the unwatermarked image. This significantly raises the difficulty of extraction. Such methods are called oblivious, public or blind watermarking algorithms.

2.3. WATERMARKING SYSTEM

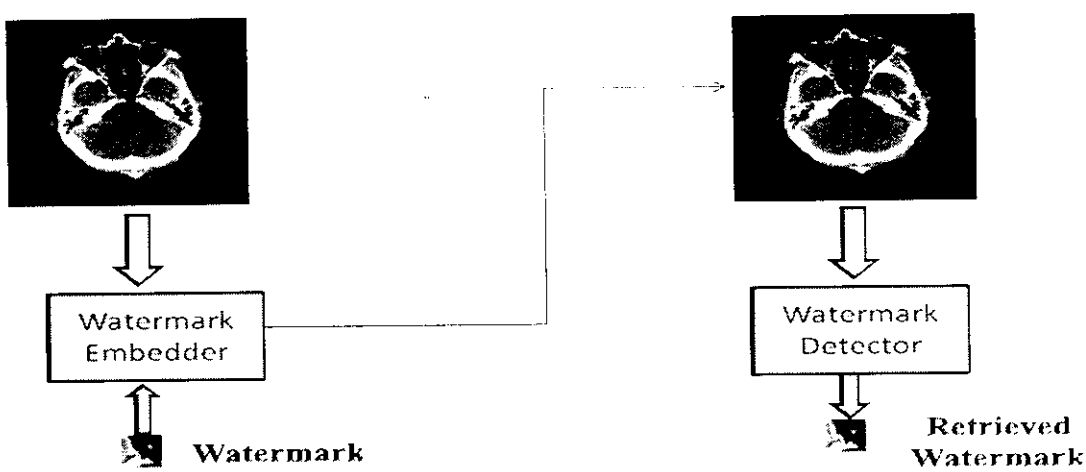


Fig.2.3.1. A Watermarking System

The figure shows the basic process of how a watermark can be embedded into an image through watermark embedder and its retrieval through a watermark detector. The watermark embedder inserts the watermark, which is certain binary information that carries the copyright to ownership information into the original image. The watermark itself can be an image or logo. The image with the watermark embedded is called the watermarked image. Given the watermarked image, one can retrieve the watermark using the watermark detector. Watermark detection can also be referred to correlating the watermarked image with a preset watermark to determine the presence of preset watermark. Instead of determining the presence of the preset watermark, the watermark detector can extract the

watermark without using the preset watermark. Watermark techniques can be categorized mainly into 2 types, private and public, according to the way they detect the watermark. Private watermark systems need the original image to be subtracted from the watermarked image to detect the watermark. On the other hand, public watermark systems do not need the original image to detect the watermark. To resolve the rightful ownership it is better to adopt public watermark systems.

Private watermark systems are restricted to be used by authorized users or content owners, who have the access to the original image. Moreover requiring the original image to detect the watermark needs extra storage of the original image at the detector's side or extra bandwidth to send the original image from the embedder to the detector. There are two basic modalities for image watermark embedding: Spatial domain techniques (Spatial watermarks) and Frequency domain techniques (Spectral watermarks). Many of the spatial watermarking techniques provide simple and effective schemes for embedding an invisible watermark into an image but are not robust against common attacks.

Another way to mark an image is to transform it into the frequency domain using DCT, Wavelet or other types of transform. The mark is incorporated directly into the transform coefficient of the image. These types of algorithms commonly use frequency sensitivity of the human visual system to ensure that watermark is invisible. Many of these techniques are not image adaptive. The image adaptive algorithms use formal visual models to determine how to embed watermark by taking advantage of image characteristics to provide perceptual transparency as well as robustness to attack.

2.4. TYPES OF WATERMARKING:

Visible and invisible watermarks both serve to determine theft but they do so in very different ways. Visible watermarks are especially useful for conveying an immediate claim of ownership. The main advantage of visible watermarks, in principle at least, is that they virtually eliminate the commercial value of the document without lessening the document's utility for legitimate, authorized purposes. A familiar example of a visible watermark is in the video domain where CNN and other television networks place their translucent logo at the bottom right of the screen image.

Purpose	Visible	Invisible
Validation of intended recipient	-	p
Non-repudiable transmission	-	p
Deterrence against theft	p	p
Diminish commercial value without	-	p
Utility		
Discourage unauthorized duplication	p	s
Digital notarization and authentication	s	p
Identify source	p	s

Table 2.4.1. Comparison of Visible and Invisible Watermarks

Invisible watermarks, on the other hand, are more of an aid in catching the thief than discouraging the theft in the first place. Though neither exhaustive nor definitive, Table 2.4.1 shows some anticipated primary (p) and secondary (s) benefits to digital watermarking.

2.5. APPLICATIONS OF DIGITAL WATERMARKING:

Watermarking is not restricted to just retaining information of the author in the work, there are various other purposes for which watermarking may be incorporated into an object. Some of them are:

a) **Copyright Protection**: For the protection of intellectual property, the data owner can embed a watermark representing the copyright information in his data.

b) **Fingerprinting**: To trace the source of illegal copies, the owner can use a fingerprinting technique. This requires the owner to embed different information onto copied of the work provided to different customers. The information embedded can be a serial number, customer id etc.

c) **Copy Protection**: the watermark represents a single copy prohibit bit and the watermark detectors in the recording device determine whether the data offered to the recorder can be stored or not.

d) **Broadcast Monitoring**: By embedding watermarks in the commercials, an automated monitoring system can determine whether the commercial was broadcasted or not. Also other TV programs which might represent significant intellectual property such as the news.

e) **Data Authentication**: Introducing fragile watermarks into the data can help ensure that the data is not processed or modified in anyway by the user.

f) **Indexing**: Introducing watermarks in video mail, movies, news items can be used to index the data.

g) **Data Hiding**: Watermarking may be used to embed longer bits of information in the data. The earliest form of this was in ancient Greece, where an author could hide his name in the text of the literary work. The term used to describe data hiding, "Steganography" originated in Greece. This was also used by the Germans/Allies in WWII to send sensitive information to outposts by hiding it in postcards.

h) **Medical Safety**: Watermarks containing the name of the patient can be embedded onto the X-Rays, MRI Scans and other test results help in instant identification of the result as belonging to a patient and thus avoid mix-ups which can lead to catastrophic consequences.

3. Water marking techniques

3.1 SPACIAL DOMAIN TECHNIQUES:

3.1.1. LSB substitution:

This is the most straightforward method of water mark embedding. In this method, the watermark is inserted into the least significant bit of the cover object. Given the extraordinarily high channel capacity of using the entire cover image for transmission, a smaller object may be inserted multiple times into the image. Even if most of these are lost due to attacks, a single surfing watermark would be considered a success.

3.1.2. LSB substitution drawbacks:

It can survive simple operations such are cropping; any addition of noise or lossy compression is going to defeat the watermark .an even better attack is to set all the LSB bits to '1'fully defeating the watermark at the cost of negligible perceptual impact on the cover object .furthermore, once the algorithm was discovered, it would be very easy for an intermediate party to alter the watermark. An improvement of the LSB substitution is to be for the embedding based on a "seed" or key be shared between the authorized users. This method, through simple, lacks the basic robustness that may be expected in any watermarking application.

3.2. CORRELATION BASED TECHNIQUES

Another technique for embedding is to exploit the correlation properties of additive pseudo random noise patterns as applied to image. A pseudo random noise (PN)pattern W(x,y) is added to the cover image I(x,y) according to the equation:

$$IW(x, y) = I(x, y) + k*W(x, y) \quad \ldots\ldots (5)$$

Where 'k' denotes the gain factor and IW is the resultant watermarked image. As 'k' increased the robustness of the watermark increases but perceptual quality of the image decreases. To retrieve the watermark, the same PN generator is seeded with the key and the correlation between the noise pattern and the image computed. If the correlation exceeds a certain a threshold T, the watermark is said to be detected.

Correlation based approach improvements, include replacing the '0' or'1' bit by 2 different PN sequences. So that the threshold decision made is not whether the bit is '0'or '1', but rather its "is this a '0'sequence, if not, is it a '1'sequence. Another improvement that has been applied is to high pass filter the image before applying the water mark, since reducing the correlation between the cover image and the PN sequence increases the immunity to noise. This approach is also referred to as the CDMA approach since the minimum required information, this ensures survival of the watermark by redundancy. Each data bits are represented by a large number of bits out of which a significant portion may be lost without totally losing the watermark information.

3.3. FREQUENCY DOMAIN IMPLEMENTATIONS

3.3. The Discrete Cosine Transform:

The Discrete cosine transform is real domains transform, which represents the entire the image as coefficients of different frequencies of cosines (which are the basic vectors for this transform). The DCT of the image is calculated by taking 8x8 blocks of the image, which are then transformed individually. The 2D DCT of an image gives the result matrix such that top left corner represents lowest frequency coefficient while the bottom right corner is the highest frequency.DCT also forms the basis of the JPEG image compression algorithm, which is one of the most widely used image data storage formats. The DCT approaches are able to withstand some forms of attack very well such as low-pass filtering\median filtering etc.

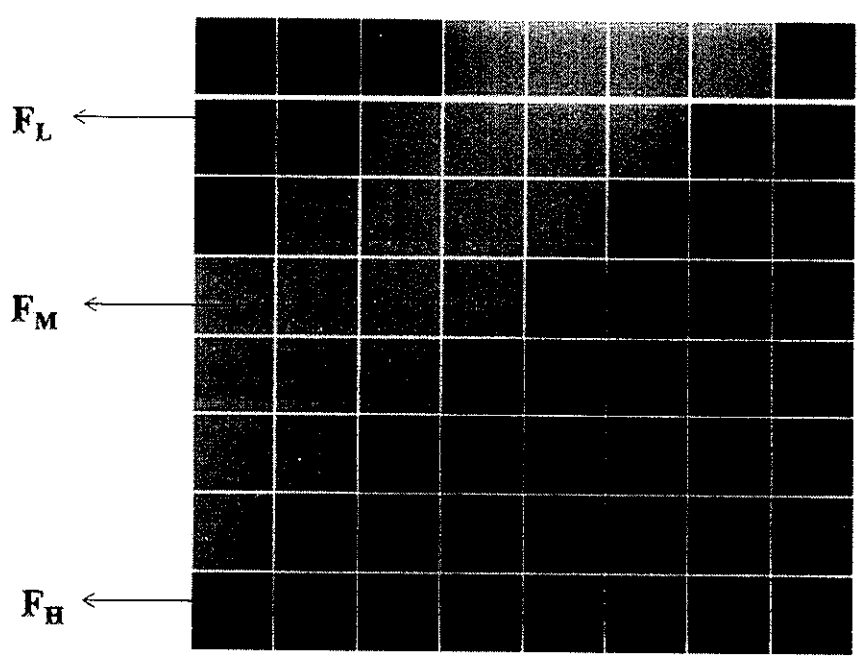


Fig.3.3.1. Definition of DCT Regions.

3.4. THE WAVELET TRANSFORM

The Wavelet Transform provides the time-frequency representation of a given signal. (There are other transforms, which give this information too, such as short time Fourier Transform, Wigner distributions, etc.)

Often times a particular spectral component occurring at any instant can be of particular interest. In these cases it may be very beneficial to know the time intervals these particular spectral components occur. For example, in EEGs, the latency of an event-related potential is of particular interest (Event-related potential is the response of the brain to a specific stimulus like flash-light, the latency of this response is the amount of this response is the amount of time elapsed between the onset of the stimulus and the response).

Wavelet transform is capable of providing the time and frequency information simultaneously, hence giving a time-frequency representation of the signal.

The Wavelet transform is defined using a single signal as bias for representation. The simplest and most popular basis function is the HAAR mother wavelet. It is a simple 2 level signal mathematically represented as:

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

One of the main advantages over the wavelet transform is that that it is believed to more accurately model aspects of the Human Visual System (HVS) as compared to the FFT or DCT. This allows us to use higher energy watermarks in regions that the HVS is known to be less sensitive to, such as the high resolution detail bands (LH, HL, and HH). Embedding watermarks in these region allow us to increase the robustness of our watermark, at little to no additional impact on image quality.

3.5. DCT-BASED TECHNIQUES:

3.5.1. Mid-band Coefficient Exchange

One of the more common techniques utilizes the comparison of middle band DCT coefficients to encode a single bit into a DCT block. To begin, we define the middle-band frequencies (F_M) of an 8x8 DCT block as shown in the preceding section. F_L is used to denote the lowest frequency components of the block, while F_H is used to denote the higher frequency components. F_M is chosen as the embedding region as to provide additional resistance to lossy compression techniques, while avoiding significant modification of the cover image.

3.6. WAVELET BASED TECHNIQUES

Another possible domain for watermark embedding is that of the wavelet domain. The DWT (Discrete Wavelet Transform) separates an image into a lower resolution approximation image (LL) as well as horizontal (HL), vertical (LH) and diagonal (HH) detail components. The process can then be repeated to computes multiple "scale" wavelet decomposition, as in the 2-scale wavelet transform shown below in figure 3.6.1.

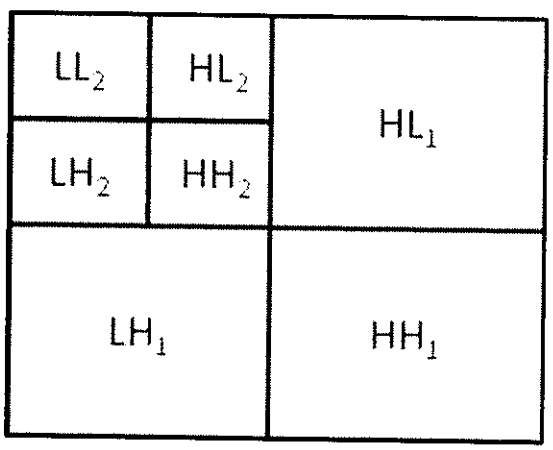


Fig.3.6.1. 2 scale 2-Dimensional Discrete Wavelet Transform

One of the most straightforward techniques is to use a similar embedding technique that used in the DCT, the embedding of a sequence of bits in the detail bands (LL, HL, LH and HH).

3.7. The Wavelet Transform

The wavelet transform (WT) has gained widespread acceptance in signal processing and image compression. Because of their inherent multi-resolution nature, wavelet-coding schemes are especially suitable for applications where scalability and tolerable degradation are important. Recently the JPEG committee has released its new image coding standard, JPEG-2000, which has been based upon DWT. Wavelet transform decomposes a signal into a set of basis functions. These basis functions are called wavelets. Wavelets are obtained from a single prototype wavelet y(t) called mother wavelet by dilations and shifting:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

Where, a is the scaling parameter and b is the shifting parameter.

The wavelet transform is computed separately for different segments of the time-domain signal at different frequencies.

Multi-resolution analysis: Analyzes the signal at different frequencies giving different resolutions.

MRA is designed to give good time resolution and poor frequency resolution at high frequencies and good frequency resolution and poor time resolution at low frequencies.

The 1-D wavelet transform is given by:

$$W_f(a,b) = \int_{-\infty}^{\infty} x(t)\psi_{a,b}(t)dt$$

The inverse 1-D wavelet transform is given by:

$$x(t) = \frac{1}{C}\int_{0}^{\infty}\int_{-\infty}^{\infty} W_f(a,b)\psi_{a,b}(t)db\frac{da}{a^2}$$

$$\text{where } C = \int_{-\infty}^{\infty}\frac{|\psi\omega)|^2}{\omega}d\omega < \infty$$

3.8. The Discrete Wavelet Transform

Discrete Wavelet Transform (DWT), which transforms a discrete time signal to a discrete wavelet representation. It converts an input series x_0, x_1,..x_m, into one high-pass wavelet coefficient series and one low-pass wavelet coefficient series (of length n/2 each) given by:

$$H_i = \sum_{m=0}^{k-1} x_{2i-m} \cdot s_m(z) \qquad (1)$$

$$L_i = \sum_{m=0}^{k-1} x_{2i-m} \cdot t_m(z) \qquad (2)$$

Where $s_m(Z)$ and $t_m(Z)$ are called wavelet filters, K is the length of the filter, and i=0, ..., [n/2]-1. In practice, such transformation will be applied recursively on the low-pass series until the desired number of iterations is reached.

3.8.1. LIFTING SCHEME

Lifting schema of DWT has been recognized as a faster approach. The basic principle is to factorize the poly phase matrix of a wavelet filter into a sequence of alternating upper and lower triangular matrices and a diagonal matrix. This leads to the wavelet implementation by means of banded-matrix multiplications.

Two Lifting Schemes:

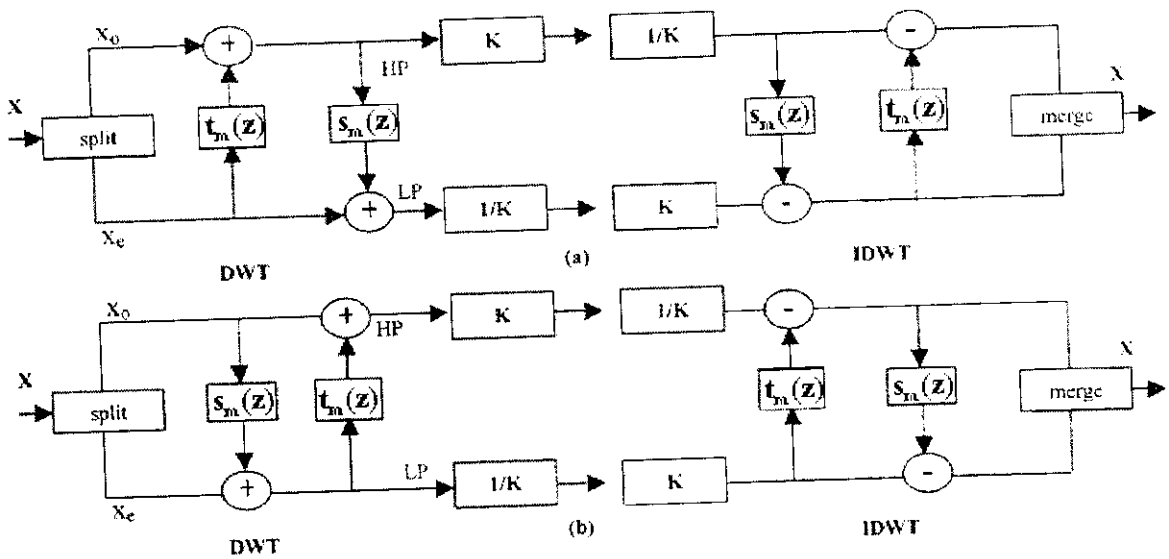


Fig.3.8.1.1. Two lifting schemes.

$$\widetilde{\mathbf{P}}_1(Z) = \begin{bmatrix} k & 0 \\ 0 & \frac{1}{k} \end{bmatrix} \prod_{i=1}^{m} \begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix} \qquad (1)$$

$$\widetilde{\mathbf{P}}_2(Z) = \begin{bmatrix} k & 0 \\ 0 & \frac{1}{k} \end{bmatrix} \prod_{i=1}^{m} \begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix} \qquad (2)$$

Where $s_i(z)$ (primary lifting steps) and $t_i(z)$ (dual lifting steps) are filters and K is a constant. As this factorization is not unique, several $\{s_i(z)\}$, $\{t_i(z)\}$ and K are admissible.

2-D DWT Subband Coding for Image

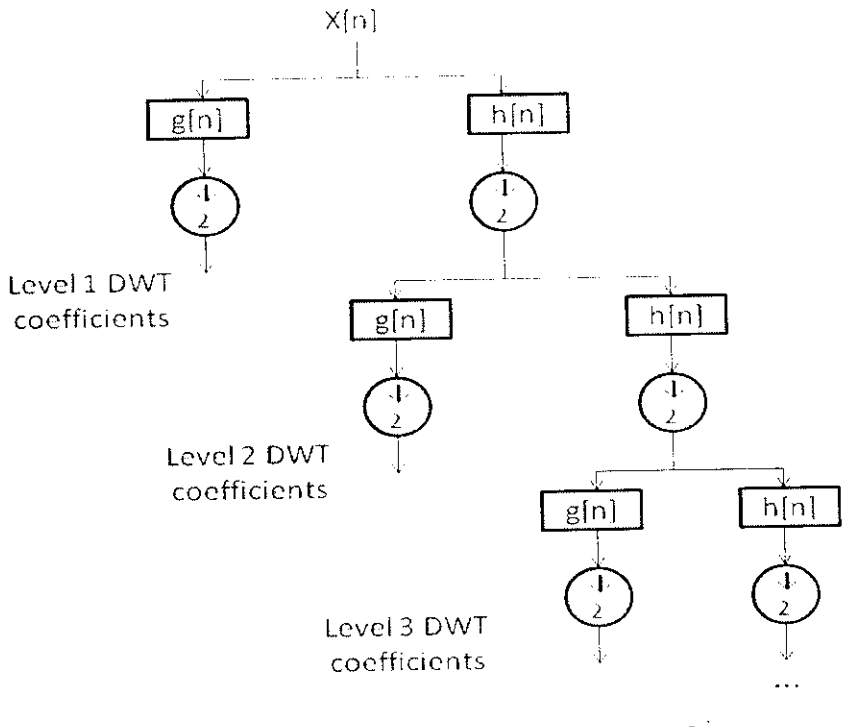


Fig.3.8.1.2. Two-D DWT Subband Coding Algorithm.

Example:

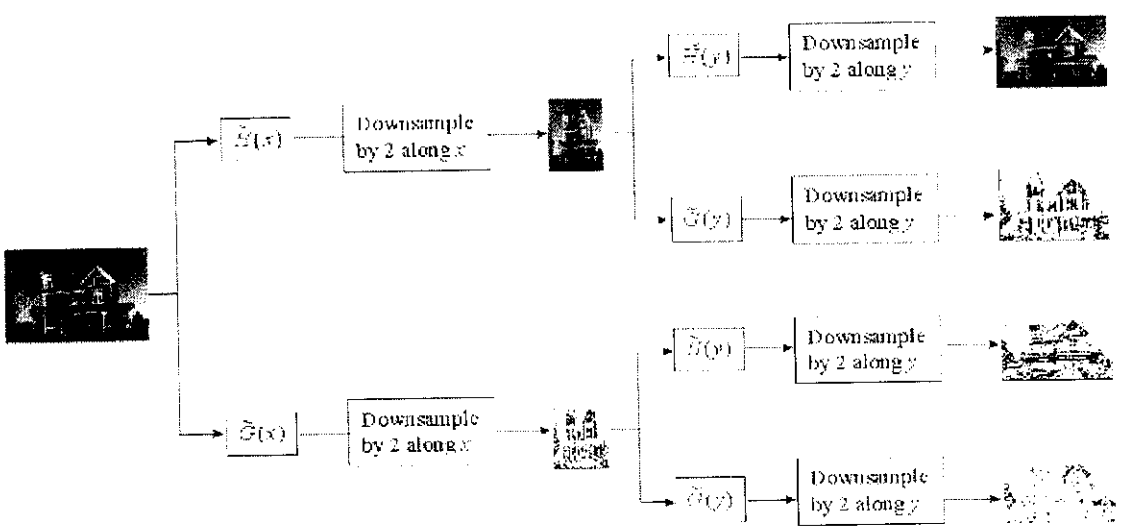


Fig.3.8.1.3. Two-D DWT for an image.

3.9. Integer DWT

Integer DWT is a more efficient approach to lossless compression. The Integer DWT coefficients are exactly represented by finite precision numbers. IWT can be computed starting from any real valued wavelet filter by means of a straightforward modification of the lifting schema. Be able to reduce the number of bits for the sample storage (memories, registers and etc.) and to use simpler filtering units.

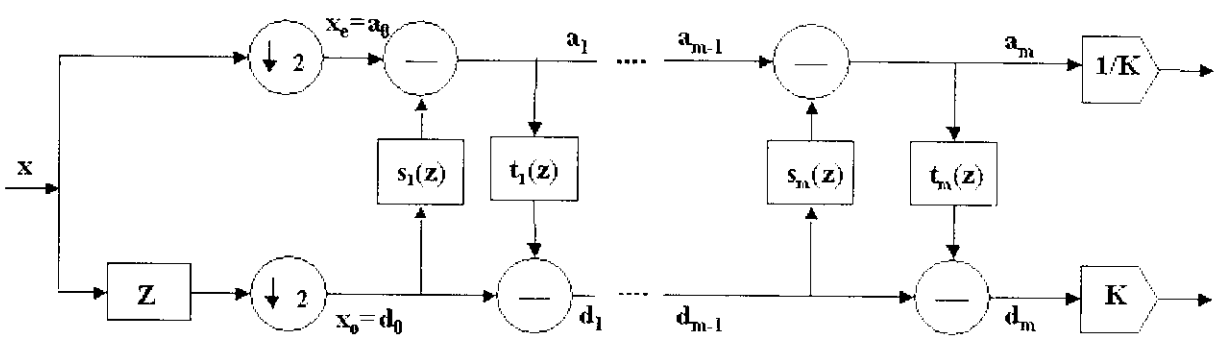


Fig.3.9.1. Integer-DWT block.

3.9.1. HAAR WAVELET

In mathematics, the Haar wavelet is a certain sequence of functions. It is now recognised as the first known wavelet. This sequence was proposed in 1909 by Alfréd Haar. Haar used these functions to give an example of a countable orthonormal system for the space of square-integrable functions on the real line. The Haar wavelet is also the simplest possible wavelet. The technical disadvantage of the Haar wavelet is that it is not continuous, and therefore not differentiable. This property can, however, be an advantage for the analysis of signals with sudden transitions, such as monitoring of tool failure in machines.

The Haar wavelet's mother wavelet function ψ(t) can be described as

$$\psi(t) = \begin{cases} 1 & 0 \leq t < 1/2, \\ -1 & 1/2 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases}$$

And its scaling function φ(t) can be described as

$$\phi(t) = \begin{cases} 1 & 0 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases}$$

3.9.2. HAAR TRANSFORM

The Haar transform is the simplest of the wavelet transforms. This transform cross-multiplies a function against the Haar wavelet with various shifts and stretches, like the Fourier transform cross-multiplies a function against a sine wave

with two phases and many stretches. The Haar transform is derived from the Haar matrix. An example of a 4x4 Haar matrix is shown below.

$$H_4 = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

The Haar transform can be thought of as a sampling process in which rows of the transform matrix act as samples of finer and finer resolution.

Analysis Decomposition and Synthesis reconstruction of Haar Transform

$$a_{l+1,k} = \left(a_{l,2k} + a_{l,2k+1}\right)/2$$

$$d_{l+1,k} = a_{l,2k+1} - a_{l,2k}$$

Analysis (Decomposition)

$$where \qquad a_{0,k} = s_k$$

$$a_{l,2k} = a_{l+1,k} + d_{l+1,k} / 2$$

$$a_{l,2k+1} = a_{l+1,k} - d_{l+1,k} / 2$$

Synthesis (Reconstruction)

Integer version of Haar Transform is obtained by directly truncating the co-efficients.

Integer Decomposition

$$a_{l+1,k} = \left\lfloor \frac{a_{l,2k} + a_{l,2k+1}}{2} \right\rfloor$$

$$d_{l+1,k} = a_{l,2k+1} - a_{l,2k}$$

Integer Synthesis

$$a_{l,2k} = a_{l+1,k} - \left\lfloor \frac{d_{l+1,k}}{2} \right\rfloor$$

$$a_{l,2k+1} = a_{l+1,k} + \left\lfloor \frac{d_{l+1,k}}{2} + 0.5 \right\rfloor$$

Both, approximation and detail, al+1, k and dl+1, k, have to be preserved – not in-place.

Haar transform with in-place structure

$$d_{l+1,k} = a_{l,2k+1} - a_{l,2k}$$

$$a_{l+1,k} = a_{l,2k} + \left\lfloor \frac{d_{l+1,k}}{2} \right\rfloor$$

Analysis

(Decomposition)

$$a_{l,2k} = a_{l+1,k} - \left\lfloor \frac{d_{l+1,k}}{2} \right\rfloor$$

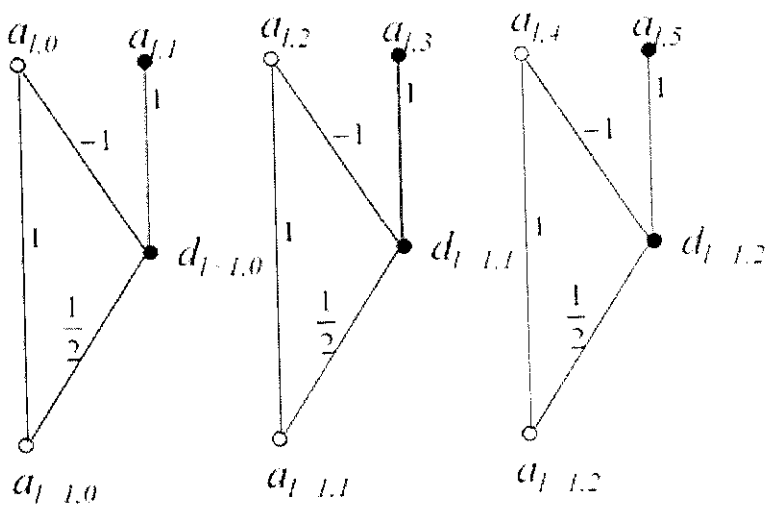
$$a_{l,2k+1} = d_{l+1,k} + a_{l,2k}$$

Synthesis

(Reconstruction)

The integer DWT is obtained by rounding of the division products in the Haar transform immediately to the next integer value.

In-place implementation



Input sequence

High-pass output

$$d^1_{l+1,k} = d^0_{l+1,k} - a^0_{l,2k}$$

Low-pass output

$$a^1_{l+1,k} = a^0_{l+1,k} + \frac{1}{2} d^1_{l+1,k}$$

4. DEVELOPING ENVIRONMENT

4.1. Hardware Environment

XILINX SPARTAN 3E FPGA (XC3S500E-4FG320C

4.2. Software Environment

Xilinx Platform Studio 8.1i Tool

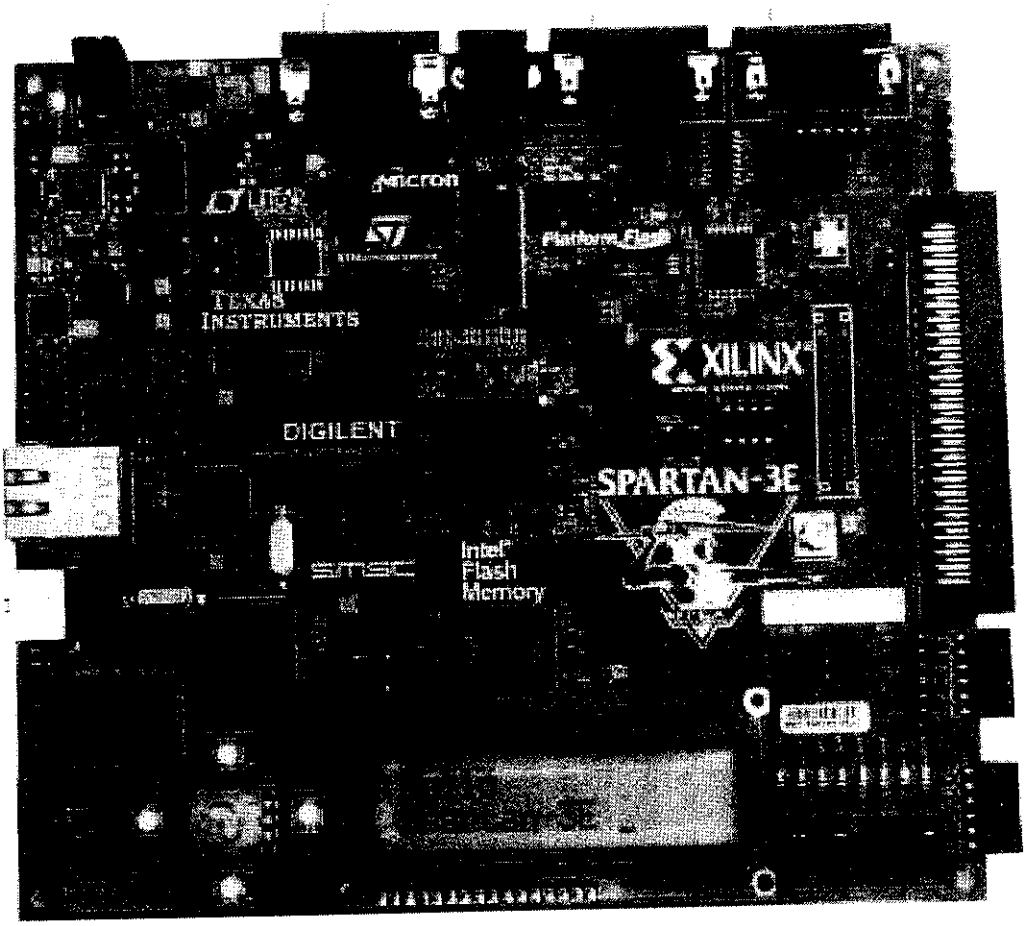
MATLAB 7.0

4.3. SPARTAN-3E STARTER KIT:

The Spartan-3E Starter Kit board highlights the unique features of the Spartan-3E FPGA family and provides a convenient development board for embedded processing applications.

The board highlights these features:

- Spartan-3E specific features
  - Parallel NOR Flash configuration
  - MultiBoot FPGA configuration from Parallel NOR Flash PROM
  - SPI serial Flash configuration
- Embedded development
  - MicroBlaze™ 32-bit embedded RISC processor
  - PicoBlaze™ 8-bit embedded controller
  - DDR memory interfaces



4.3.1 KEY COMPONENTS AND FEATURES:

The key features of the Spartan-3E Starter Kit board are:

- Xilinx XC3S500E Spartan-3E FPGA
  - Up to 232 user-I/O pins
  - 320-pin FBGA package
  - Over 10,000 logic cells

- Xilinx 4 Mbit Platform Flash configuration PROM
- Xilinx 64-macrocell XC2C64A CoolRunner CPLD
- 64 MByte (512 Mbit) of DDR SDRAM, x16 data interface, 100+ MHz
- 16 MByte (128 Mbit) of parallel NOR Flash (Intel StrataFlash)
  - FPGA configuration storage
  - MicroBlaze code storage/shadowing
- 16 Mbits of SPI serial Flash (STMicro)
  - FPGA configuration storage
  - MicroBlaze code shadowing
- 2-line, 16-character LCD screen
- PS/2 mouse or keyboard port
- VGA display port
- 10/100 Ethernet PHY (requires Ethernet MAC in FPGA)
- Two 9-pin RS-232 ports (DTE- and DCE-style)
- On-board USB-based FPGA/CPLD download/debug interface
- 50 MHz clock oscillator
- SHA-1 1-wire serial EEPROM for bitstream copy protection
- Hirose FX2 expansion connector
- Three Digilent 6-pin expansion connectors
- Four-output, SPI-based Digital-to-Analog Converter (DAC)
- Two-input, SPI-based Analog-to-Digital Converter (ADC) with programmable-gain pre-amplifier
- ChipScope™ SoftTouch debugging port
- Rotary-encoder with push-button shaft
- Eight discrete LEDs
- Four slide switches

4.3.2 CONFIGURATION METHODS:

A typical FPGA application uses a single non-volatile memory to store configuration images. To demonstrate new Spartan-3E capabilities, the starter kit board has three different configuration memory sources that all need to function well together. The extra configuration functions make the starter kit board more complex than typical Spartan-3E applications.

The starter kit board also includes an on-board USB-based JTAG programming interface. The on-chip circuitry simplifies the device programming experience. In typical applications, the JTAG programming hardware resides off-board or in a separate programming module, such as the Xilinx Platform USB cable.

4.3.3 VOLTAGES FOR ALL APPLICATIONS:

The Spartan-3E Starter Kit board showcases a triple-output regulator developed by Texas Instruments, the **TPS75003** specifically to power Spartan-3 and Spartan-3E FPGAs. This regulator is sufficient for most stand-alone FPGA applications. However, the starter kit board includes DDR SDRAM, which requires its own high-current supply. Similarly, the USB-based JTAG download solution requires a separate 1.8V supply.

4.3.4 JTAG:

JTAG primary purpose is to allow a computer to take control of the state of all the IO pins on a board. In turn, this allows each device connectivity to other devices on the board to be tested. Standard JTAG commands can be used for this purpose.

FPGAs are JTAG-aware and so all the FPGA IO pins can be controlled from the JTAG interface. FPGAs add the ability to be configured through JTAG (using proprietary JTAG commands).

JTAG consists of 4 signals: TDI, TDO, TMS and TCK. A fifth pin, TRST, is optional. A single JTAG port can connect to one or multiple devices (as long as they are all JTAG-aware parts). With multiple devices, you create what is called a "JTAG chain". The TMS and TCK are tied to all the devices directly, but the TDI and TDO form a chain: TDO from one device goes to TDI of the next one in the chain. The master controlling the chain (a computer usually) closes the chain.

4.3.5 RS 232:

As shown in Figure 4.5.1, the Spartan-3E Starter Kit board has two RS-232 serial ports: a female DB9 DCE connector and a male DTE connector. The DCE-style port connects directly to the serial port connector available on most personal computers and workstations via a standard straight-through serial cable. Null modem, gender changers, or crossover cables are not required.

Use the DTE-style connector to control other RS-232 peripherals, such as modems or printers, or perform simple loop back testing with the DCE connector.

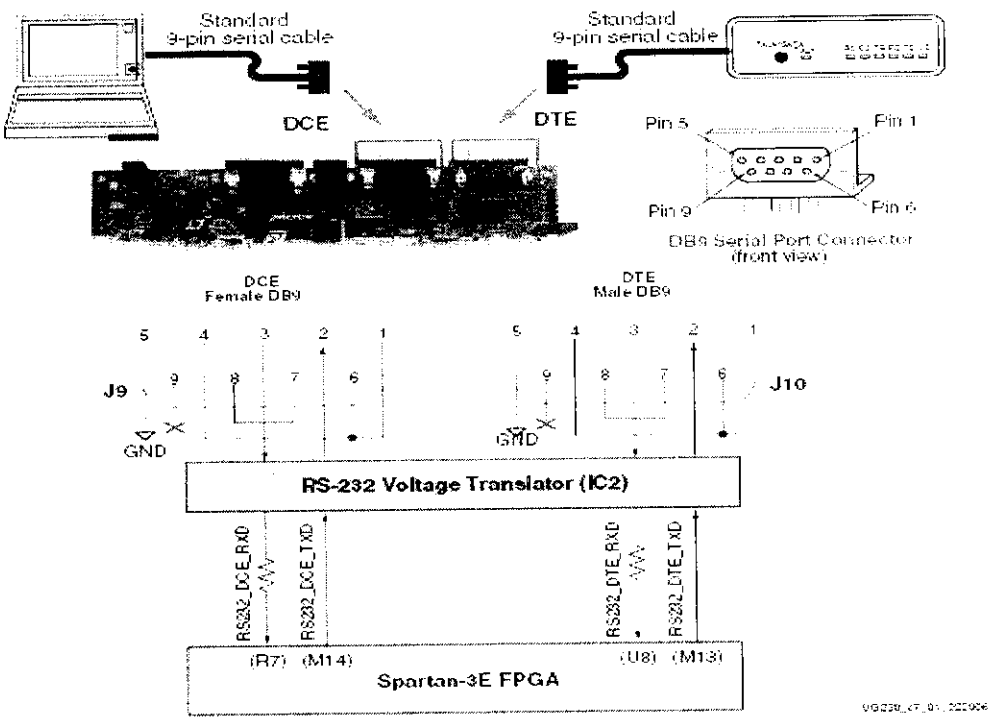


Figure 7-1: **RS-232 Serial Ports**

Fig.4.5.1. Connection between the FPGA and the two DB9 connectors.

The FPGA supplies serial output data using LVTTL or LVCMOS levels to the Maxim device, which in turn, converts the logic value to the appropriate RS-232 voltage level. Likewise, the Maxim device converts the RS-232 serial input data to LVTTL levels for the FPGA. A series resistor between the Maxim output pin and the FPGA's RXD pin protects against accidental logic conflicts.

Hardware flow control is not supported on the connector. The port's DCD, DTR, and DSR signals connect together, as shown in Figure 9. Similarly, the port's RTS and CTS signals connect together.

5. IMPLEMENTATION

5.1. Watermarking Algorithm

A typical watermarking algorithm consists of 3 processes. They are

a) Watermark Generation

b) Watermark Embedding

c) Watermark Detection

The above three processes are explained with the block diagrams given below. The embedding block diagram includes the watermark generation also.

5.2. EMBEDDING PROCESS – BLOCK DIAGRAM

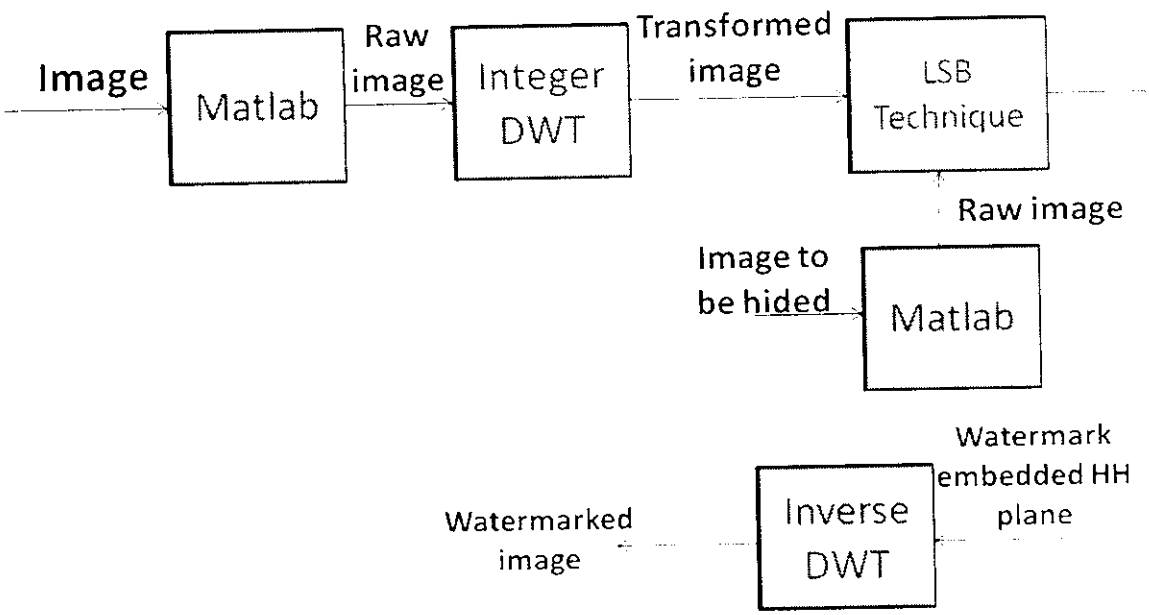


Fig.5.2.1. Watermark Embedding Process Block Diagram

5.2.1. Raw Image preparation

1. The image to be watermarked is read by MATLAB using 'imread' function.

2. Then the image is converted into raw image by writing it as a two dimensional matrix in text format.

3. The commands used for this process are 'fopen' and 'fprintf'.

4. If the data to be hided is also an image then the same method is used to create the raw image of watermark image.

5. If the data to be hided is a text then the ASCII values of the characters is the elements of the raw image.

5.2.2 Watermark Embedding Algorithm:

1. The raw image is first read and transformed into frequency domain by 1 level two dimensional Integer DWT.
2. One level Integer DWT consists of 4 sub bands. They are LL, HL, LH, HH sub bands. In this HH sub band is reproduced without loss after inverse transform. So, watermark is embedded in HH sub band.

The decomposition equation is

$$d_{l+1,k} = a_{l,2k+1} - a_{l,2k}$$

$$a_{l+1,k} = a_{l,2k} + \text{ceil}((d_{l+1,k})/2)$$

Where,

l – Decomposition level.

a_k – k^{th} element in the matrix.

Using this decomposition equation, the high pass and low pass matrices are obtained and then once again using this equation on high pass and low pass matrices the 4 sub bands (HH, HL, LH and LL) are obtained. In these 4 sub bands only the HH sub band has detailed coefficients and all other sub bands have approximate coefficients due to truncation.

3. The watermarking process is done on LSB plane of HH sub band by replacing the LSB bits of HH sub band coefficients with the bits of data or image to be hided.
4. Inverse IDWT is done using the watermarked HH sub band and LL, HL, LH sub bands to get watermarked image.

The Reconstruction equation is

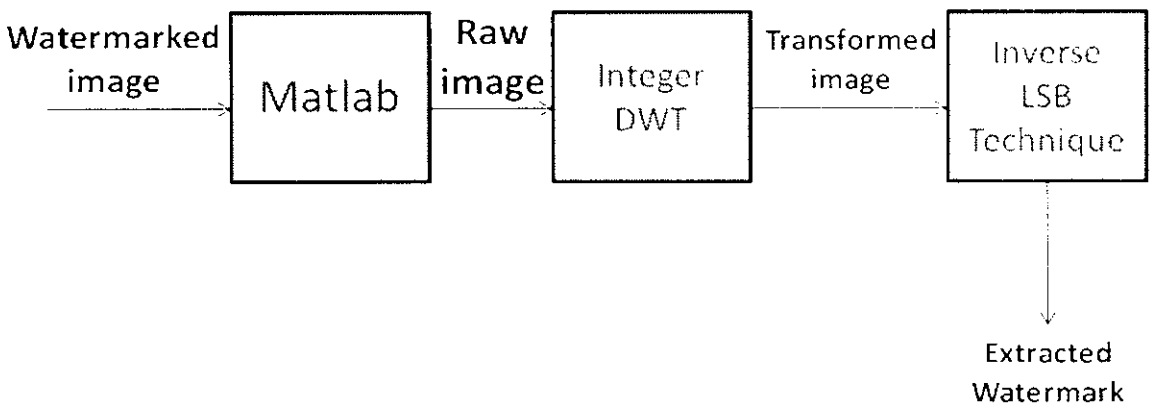
$a_{l,2k} = a_{l+1,k} - \text{ceil}((d_{l+1,k})/2)$

$a_{l,2k+1} = d_{l+1,k} + a_{l,2k}.$

The high pass and low pass matrices are obtained first using this equation and applying this equation once again on high pass and low pass matrices, the watermarked image is obtained.

5. The watermarked image is then sent to the receiver through a network.

5.3. Detection Process – Block Diagram



Fg.5.3.1. Watermark Detection Process Block Diagram

5.3.1. Watermark Extraction Algorithm:

1. The extraction process is a reverse process of embedding process. The received image is first converted into raw image.
2. The Integer Discrete Wavelet Transform is performed on raw image to get sub bands.
3. From the LSB plane of HH sub band the watermarked data is extracted by inverse LSB technique. The LSB bits of HH sub band coefficients are read and they are given at the output.

We demonstrate the performance of our method. We use test image of size 128x128 and chose the decomposition level to be one. The band to embed the watermark is the HH band. Our binary watermark image is a 16x16 image. The embedded watermark is retrieved in a lossless manner. The Integer DWT coefficients are only 8 bit long. So total memory occupied by this algorithm is very less and the implementation process is fast when compared to DWT.

6. RESULTS

In this work two test images were watermarked with their respective region of interest which carries useful information and these watermarks were extracted from those watermarked images successfully. The results are shown below.

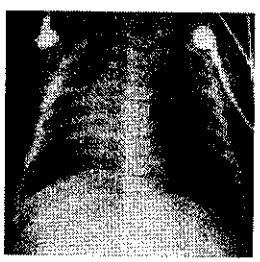


Fig. 6.1. Original Image
Size = 128 x 128

Fig. 6.2. Watermark Image
size = 16 x 16

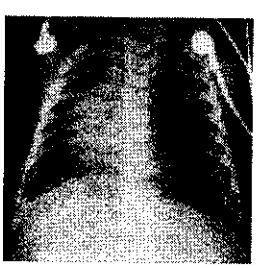
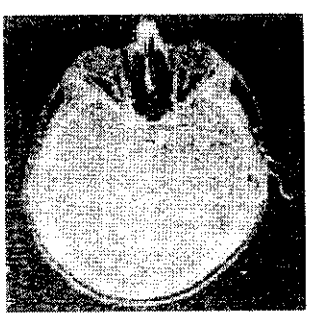


Fig. 6.3. Watermarked image
Size = 128 x 128

Fig. 6.4. Extracted watermark
size = 16 x 16

6.5. Original Image
128x128 pixels

6.7. Watermark Image
16x16 pixels



6.7. Watermarked image
128x128 pixels

6.8. Extracted watermark
16x16 pixels

The fig.6.1 & 6.5 are the original test images (X-ray image of lungs & CT scan of head) in which watermarking is to be done. The size of the images is 128 x 128. The fig.6.2 & 6.6 are watermark image. The watermarks are Region of Interest (ROI) containing important information and were taken from the respective test images by cropping. The size of the watermarks is 16 x 16. The fig.6.3 & 6.7 are the watermarked images. Due to integer rounding a small amount of distortion is there. The extracted watermarks are in fig.6.4 & 6.8. The extracted watermarks are as that of embedded watermarks. There is no distortion in watermark. The integrity of watermark is protected.

7. CONCLUSION

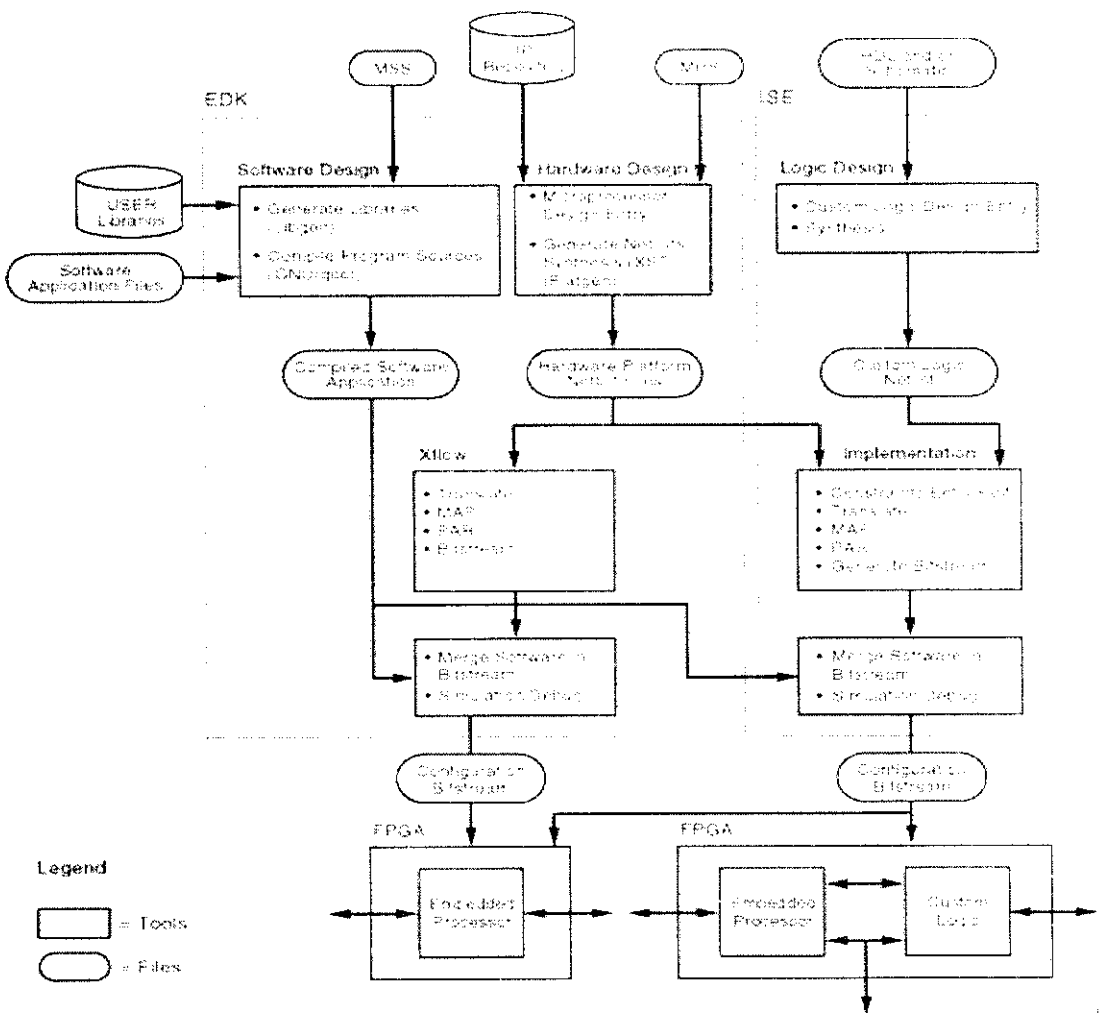
The watermarking using Integer Discrete Wavelet Transform (Integer DWT) and Bit Plane Coding is implemented in FPGA using Xilinx Platform Studio 8.1i software tool. This system requires less memory and the process is very fast. It guarantees the security of watermark data and the hardware implementation reduces the requirement of computers and microprocessors. This method can be used for medical field to ensure the security of patient information and can be used for secure transmission of region of interest as the watermark and also in military applications to ensure the security of secrete informations.

Here, for taking Region of Interest simple cropping was used. In future separate segmentation algorithms may be proposed for taking region of interest. The test images used here are of 128 x 128 dimension. Due to complexity and memory of FPGA, the size of the image is restricted. In future the dimension may be expanded to greater size.

APPENDIX

HARDWARE AND SOFTWARE ARCHITECTURE DEVELOPMENT

XILINX PLATFORM STUDIO (XPS):



XPS includes a graphical user interface (GUI), along with a set of tools that aid in project design. From the XPS GUI, you can design a complete embedded processor system for implementation within a Xilinx FPGA device. The XPS main window is shown in the figure below.

Note that the XPS main window is divided into three areas:

i. The Project Information Panel
ii. The System Assembly Panel
iii. The Connectivity Panel

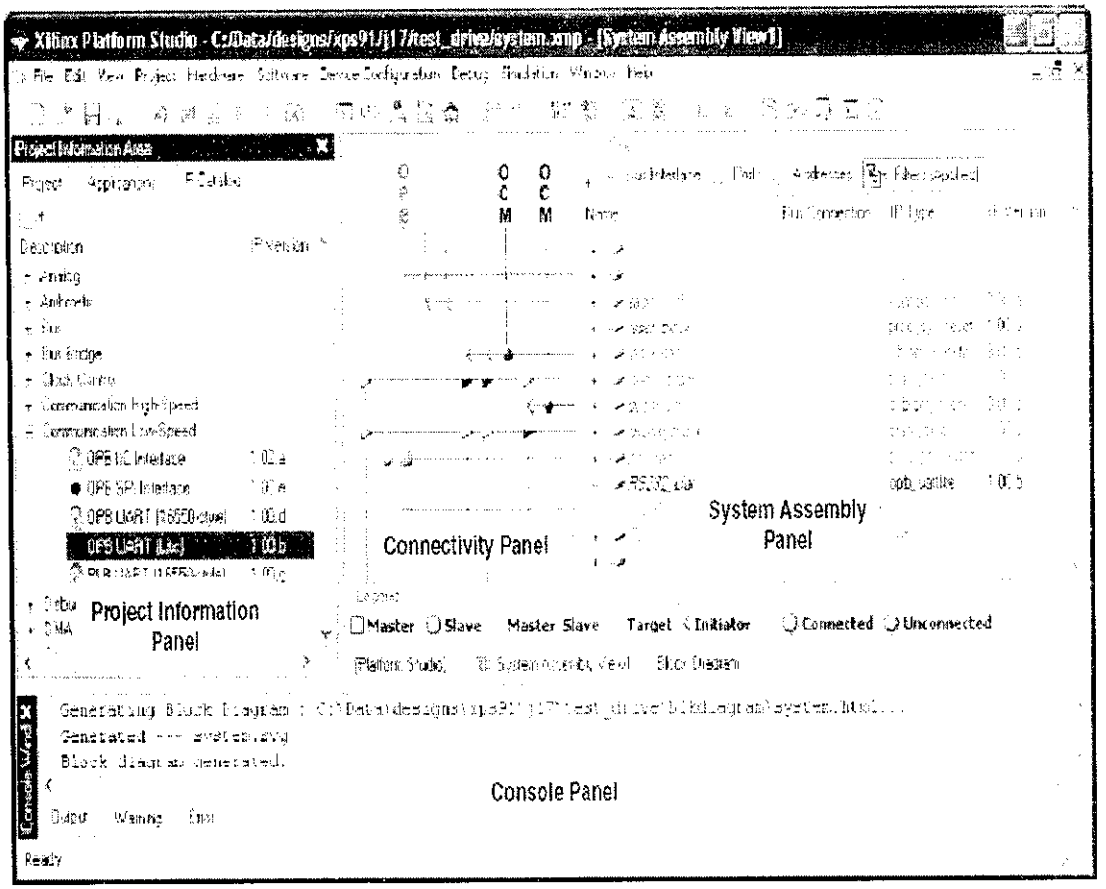


Figure 3-1: Xilinx Platform Studio GUI

i. PROJECTION INFORMATION PANEL:

The Project Information panel offers control over and information about your project. The Project Information panel provides Project, Applications, and IP Catalog tabs

- **THE PROJECT TAB:**

The Project Tab lists references to project related files. Information is grouped in the following general categories:

Project Files:

All project-specific files such as the Microprocessor Hardware Specification (MHS) files, Microprocessor Software Specification (MSS) files, User Constraints

File (UCF) files, Impact Command files, Implementation Option files, and Bitgen Option files.

Project Options:

All project-specific options, such as Device, Netlist, Implementation, Hardware Description Language (HDL), and Sim Model options.

Reference Files:

All log and output files produced by the XPS implementation processes.

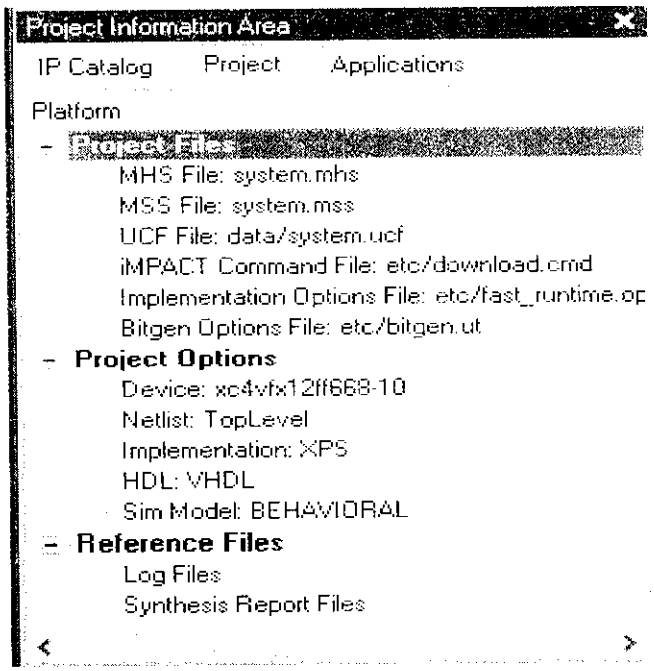


Figure 3-2: **Project Information Area: Project Tab**

- **APLLICATION TAB:**

The Applications tab lists all software application option settings, header files, and source files associated with each application project. With this tab selected, you can:

- Create and add a software application project, build the project, and load it to the block RAM.
- Set compiler options.
- Add source and header files to the project.

- **IP CATALOG TAB:**

The IP Catalog tab lists all the EDK IP cores and any custom IP cores you created. If a project is open, only the IP cores compatible with the target Xilinx device architecture are displayed. The catalog lists information about the IP cores, including release version, status (active, early access or deprecated), lock

(not licensed, locked, or unlocked), processor support, and a short description. Additional details about the IP core, including the version change history, data sheet, and Microprocessor Peripheral Description (MPD) file, are available in the right-click menu. By default, the IP cores are grouped hierarchically by function.

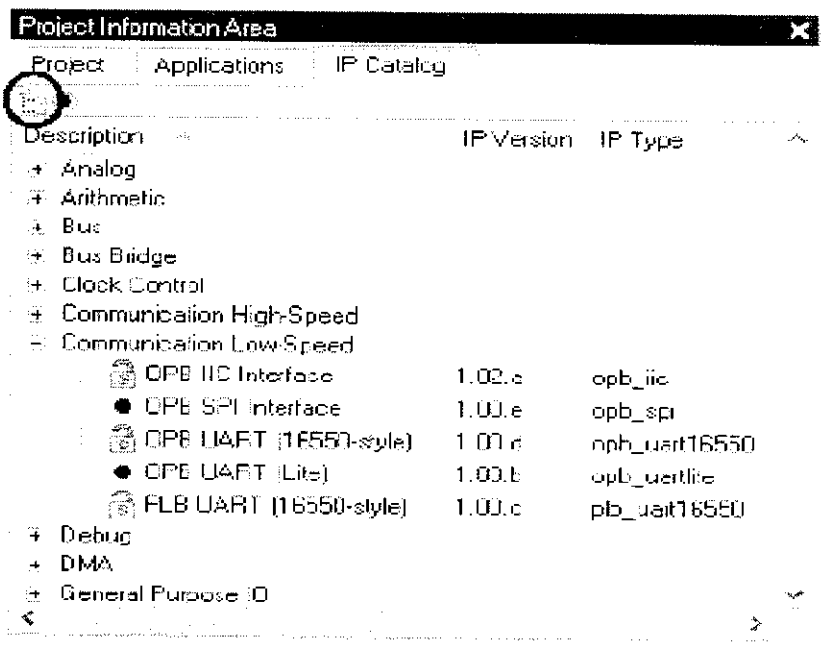


Figure 3-4: **Project Information Area: IP Catalog Tab**

ii. THE SYSTEM ASSEMBLY PANEL:

The System Assembly Panel is where you view and configure system block elements. If the System Assembly Panel is not already maximized in the main window, click the **System Assembly** tab at the bottom of the pane to open it.

BUS INTERFACE, PORTS, and ADDRESS FILTERS:

XPS provides **Bus Interface**, **Ports**, and **Addresses** radio buttons in the System Assembly Panel (shown in the figure below), which organize information about your design and allow you to edit your hardware platform more easily.

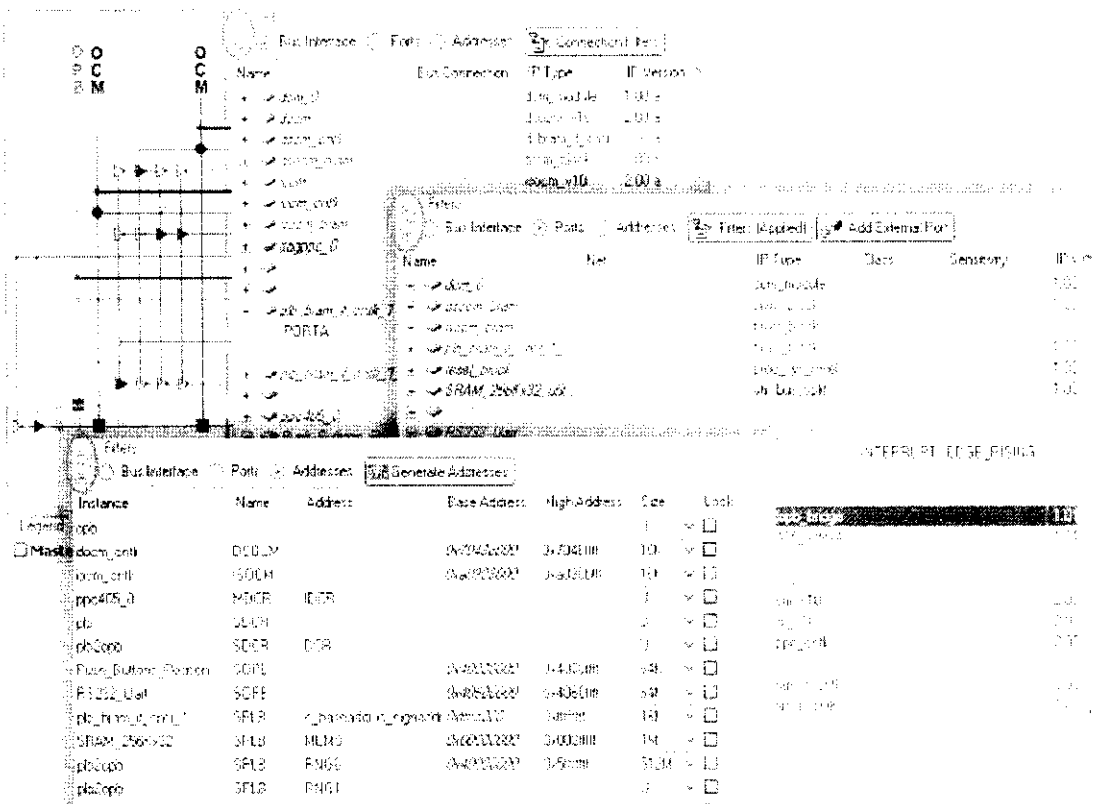


Figure 3-5: **System Assembly Panel Views**

iii. THE CONNECTIVITY PANEL:

With the Bus Interface filter selected, you'll see the Connectivity Panel, highlighted by the dashed line in Figure 3-5. The Connectivity Panel is a graphical representation of the hardware platform interconnects.

- A vertical line represents a bus, and a horizontal line represents a bus interface to an IP core.
- If a compatible connection can be made, a connector is displayed at the intersection between the bus and IP core bus interface.

- The lines and connectors are color-coded to show the compatibility.
- Differently shaped connection symbols indicate mastership of the IP core bus interface.
- A hollow connector represents a connection that you can make, and a filled connector represents a connection made. To create or disable a connection, click the connector symbol.

INTEGRATED SOFTWARE ENVIRONMENT (ISE):

ISE is the foundation for Xilinx FPGA logic design. Because FPGA design can be an involved process, Xilinx has provided software development tools that allow the designer to circumvent some of this complexity. Various utilities such as constraints entry, timing analysis, logic placement and routing, and device programming have all been integrated into ISE.

EMBEDDED DEVELOPMENT KIT (EDK)

Microprocessor Hardware Specification (MHS):

XPS provides an interactive development environment that allows you to specify all aspects of your hardware platform. XPS maintains your hardware platform description in a high-level form, known as the Microprocessor Hardware Specification (MHS) file. The MHS, an editable text file, is the principal source file representing the hardware component of your embedded system. XPS synthesizes the MHS source file into Hardware Description Language (HDL) netlists ready for FPGA place and route.

The MHS file is integral to your design process. It contains all peripherals along with their parameters. The MHS file defines the configuration of the embedded

processor system and includes information on the bus architecture, peripherals, processor, connectivity, and address space.

Microprocessor Software Specification (MSS):

XPS maintains an analogous software system description in the Microprocessor Software Specification (MSS) file. The MSS file, together with your software applications, are the principal source files representing the software elements of your embedded system. This collection of files, used in conjunction with EDK installed libraries and drivers and any custom libraries and drivers for custom peripherals you provide, allows XPS to compile your applications. The compiled software routines are available as an Executable and Linkable Format (ELF) file. The ELF file is the binary ones and zeros that are run on the processor hardware. The figure below shows the files and flow stages that generate the ELF file.

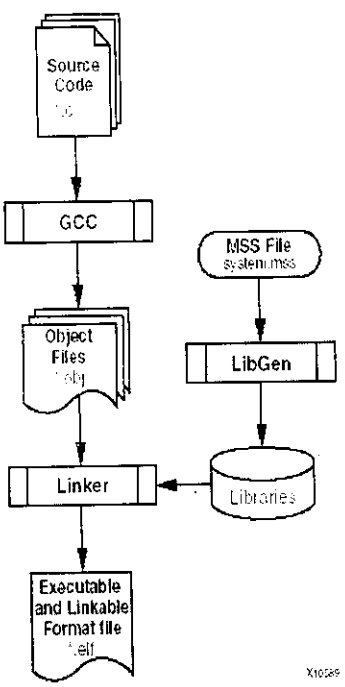


Figure 6-1: Elements and Stages of ELF File Generation

CREATING THE PROJECT IN XPS:

Building the User Application:

In EDK 8.2, XPS provides the ability for the user to create multiple software projects. These projects can include

Source files, header files, and linker scripts. Unique software projects allow the designer to specify the following options for each software project:

- Specify compiler options
- Specify which projects to compile
- Specify which projects to download
- Build entire projects

Software application code development can be managed by selecting the Applications tab as shown in Figure 5.

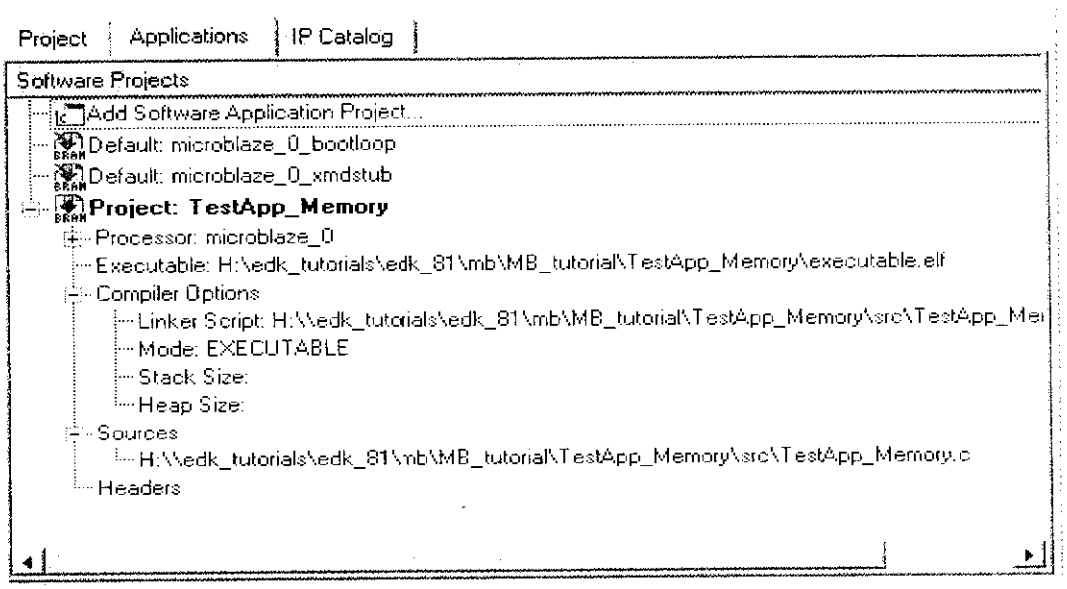


Figure 5: Applications Tab

Compiling the Code:

. Select ***Software → Build All User Applications*** to run. Compiles the source files.

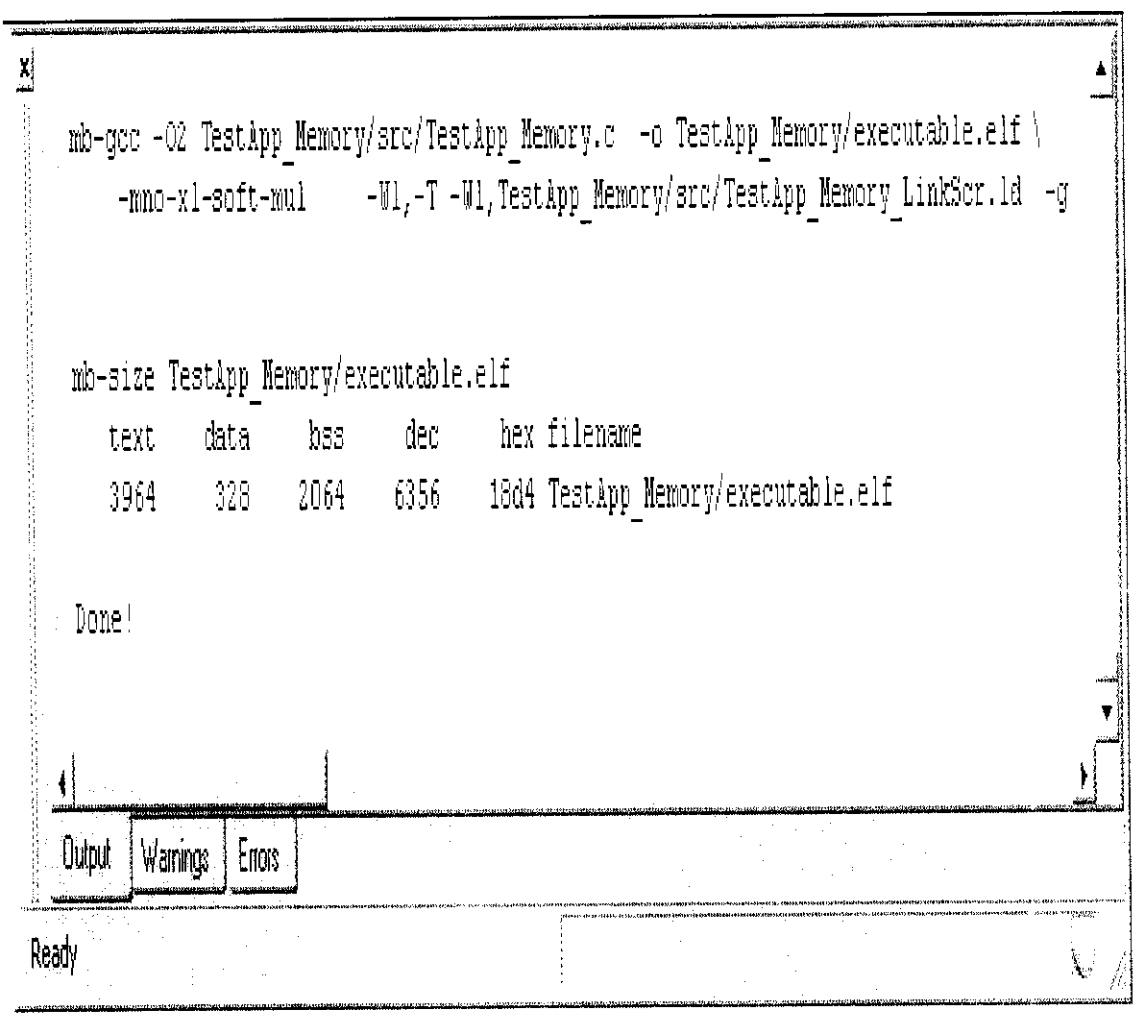


Figure 6: XPS Output Window - Software Compiled

Downloading the Design:

Now that the hardware and software designs are completed, the device can be configured. Follow these steps to download and configure the FGPA:

- Connect the host computer to the target board, including connecting the Parallel-JTAG cable and the serial cable.
- Start a hyperterminal session with the following settings:

    - com1 – This is dependant on the com port your serial cable is connected to.
    - Bits per second: 57600
    - Data bits: 8
    - Parity: none
    - Stop bits: 1
    - Flow control: none

- Connect the board power
- In ISE, Select system_stub.vhd in the source window
- In the process window, double click on Update Bitstream with Processor Data.
- In the process window, double click on Configure Device (iMPACT) under Generate Programming File
- With iMPACT, configure the FPGA using system_stub_download.bit located in the project_navigator directory choosing to bypass all of the other chips in the JTAG chain

After the configuration is complete, you should see a display similar to that in shown in Figure 7:

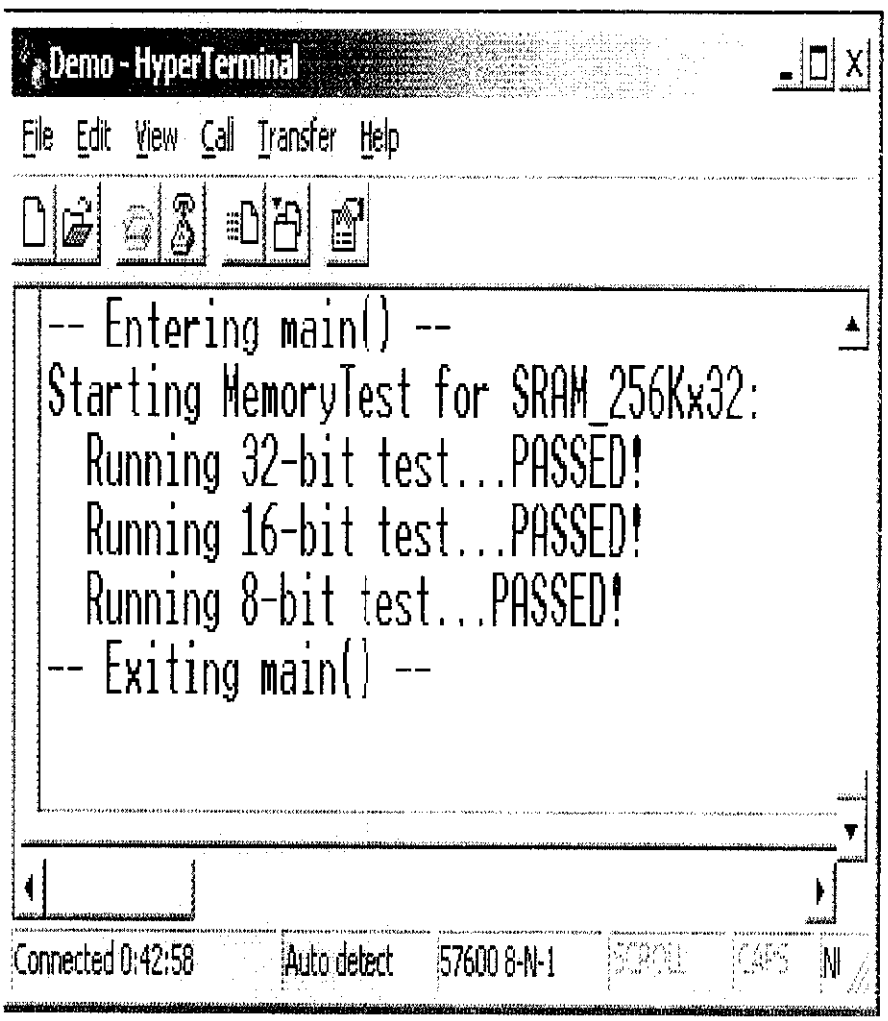


Figure 7: Hyperterminal Output

MICROBLAZE:

OVERVIEW:

The MicroBlaze™ embedded processor soft core is a reduced instruction set computer (RISC) optimized for implementation in Xilinx® Field Programmable Gate Arrays (FPGAs). Figure 1-1 shows a functional block diagram of the MicroBlaze core.

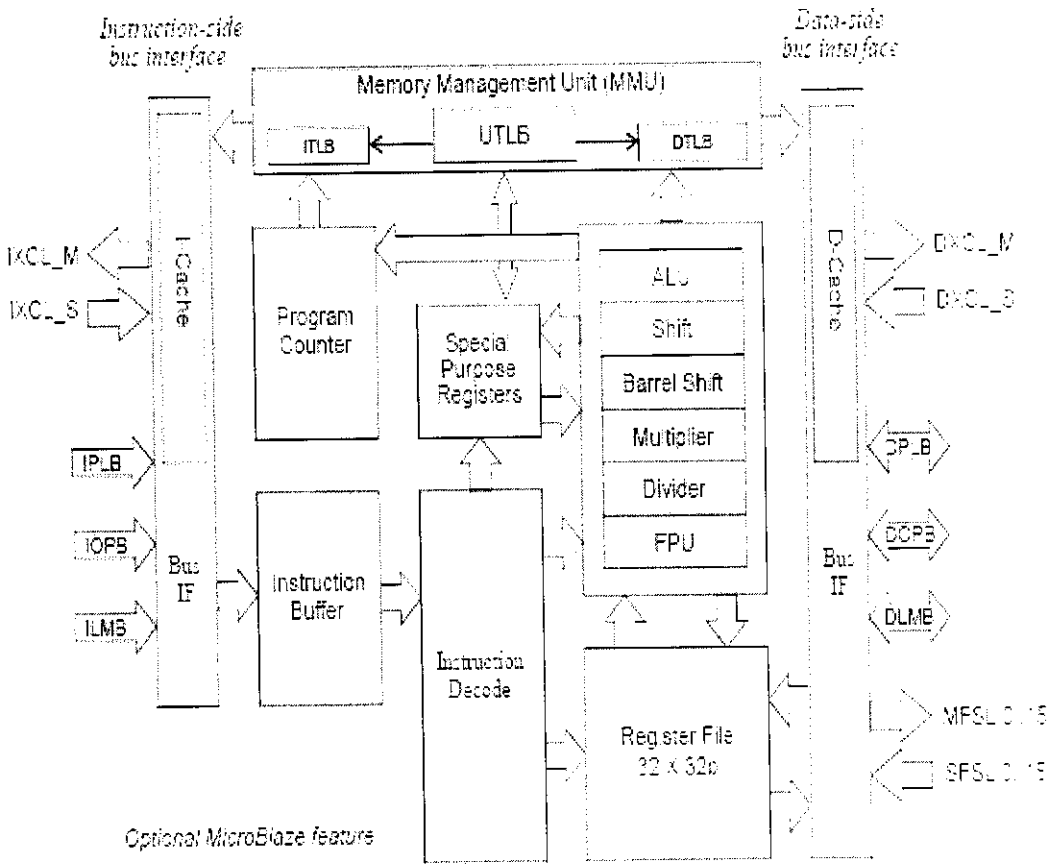


Figure 1-1: **MicroBlaze Core Block Diagram**

FEATURES:

The MicroBlaze soft core processor is highly configurable, allowing you to select a specific set of features required by your design.

The fixed feature set of the processor includes:

- Thirty-two 32-bit general purpose registers
- 32-bit instruction word with three operands and two addressing modes
- 32-bit address bus

• Single issue pipeline

In addition to these fixed features, the MicroBlaze processor is parameterized to allow selective enabling of additional functionality. Older (deprecated) versions of MicroBlaze support a subset of the optional features described in this manual. Only the latest (preferred) version of MicroBlaze (v7.00) supports all options.

Xilinx recommends that all new designs use the latest preferred version of the MicroBlaze processor.

MICROBLAZE I/O OVERVIEW:

The core interfaces shown in Figure 1-1 are defined as follow:

DPLB: Data interface, Processor Local Bus

DOPB: Data interface, On-chip Peripheral Bus

DLMB: Data interface, Local Memory Bus (BRAM only)

IPLB: Instruction interface, Processor Local Bus

IOPB: Instruction interface, On-chip Peripheral Bus

ILMB: Instruction interface, Local Memory Bus (BRAM only)

MFSL 0-15: FSL master interfaces

SFSL 0-15: FSL slave interfaces

IXCL: Instruction side Xilinx Cache Link interface (FSL master/slave pair)

DXCL: Data side Xilinx Cache Link interface (FSL master/slave pair)

Core: Miscellaneous signals for: clock, reset, debug, and trace.

Processor Local Bus (PLB) Interface Description:

The MicroBlaze PLB interfaces are implemented as byte-enable capable 32-bit masters. Please refer to the PLBV46 Interconnect and Interfaces document for details.

On-Chip Peripheral Bus (OPB) Interface Description:

The MicroBlaze OPB interfaces are implemented as byte-enable capable masters. Please refer to the OPB Usage in Xilinx FPGA document for details.

Local Memory Bus (LMB) Interface Description:

The LMB is a synchronous bus used primarily to access on-chip block RAM. It uses a minimum number of control signals and a simple protocol to ensure that local block RAM are accessed in a single clock cycle. LMB signals and definitions are shown in the following table. All LMB signals are active high.

REFERENCES

[1] S. Kurshid Jinna, Dr. L. Ganesan. (2009) "Reversible Image Watermarking using Bit Plane Coding and Integer Wavelet Transform", in IJCSNS International Journal of Computer Science and Network Security, 250 VOL.9 No.11, November.

[2] Kamran Hameed, Adeel Mumtaz, and S.A.M. Gilani. (2006) "Digital Image Watermarking in the Wavelet Transform Domain", Proceedings of World Academy of Science, Engineering and Technology, Volume 13, May.

[3] Rafael C. Gonzalez, Richard E.Woods. "Digital Image Processing", Third Edition.

[4] www.mathworks.com

[5] http://en.wikipedia.org.