International Journal of Ethics in Engineering & Management Education
Website: www.ijeee.in (ISSN: 2348-4748, Volume 1, Issue 3, March 2014)

Design & Implementation of Image Compression using Huffman Coding through VHDL

T. Krishnarjuna Rao
Associate Professor, Dept. of ECE
RVR Institute of Engg & Tech
Hyderabad, India
krishna410@gmail.com

P. Aruna Sreee
Assistant Professor, Dept. of ECE
RVR Institute of Engg & Tech
Hyderabad, India
arunasree03@gmail.com

Kumar.Keshamoni
Assistant Professor, Dept. of ECE
RVR Institute of Engg & Tech
Hyderabad, India
kumar.keshamoni@gmail.com

***Abstract:* The DCT is a loss-less and reversible mathematical transformation that converts a spatial amplitude representation of data into a spatial frequency representation. The DCT has been used in many applications such as filtering, transmultiplexers, speech coding, image coding (still frame, video and image storage), pattern recognition, image enhancement, and SAR/IR image coding. The DCT is widely used in image compression applications, especially in lossy image compression. The energy compaction property of the DCT is well suited for image compression since, as in most images, the energy is concentrated in the low to middle frequencies, and the human eye is more sensitive to the middle frequencies. On the other hand, the Fast Fourier Transform (FFT) is easy to implement, but does not provide an adequate compression ratio. Huffman codes are instantaneous; that is, the receiver knows immediately when a complete symbol has been received, and does not have to look further to correctly identify the symbol. Instantaneous codes are just those with the prefix property: no code word is a prefix of another code word. Parallel coding is simplified since the code bits for all pixels are disjoint and independent.**

***Key words:* EM spectrum, IFS, RIFS, DIP, VHDL, DCT, DFT**

1. INTRODUCTION

In this paper we implement a lossless image compression technique where the discrete cosine transform of the signal is performed, then calculated a threshold based on the compression ratio required by the user. The compression is done by using Huffman coding.

1.1 Digital Image Processing: Digital image processing is an area characterized by the need for extensive experimental work to establish the viability of proposed solutions to a given problem. An important characteristic underlying the design of image processing systems is the significant level of testing & experimentation that normally is required before arriving at an acceptable solution. This characteristic implies that the ability to formulate approaches and quickly prototype candidate solutions generally plays a major role in reducing the cost and time required to arrive at a viable system implementation. An image may be defined as a two-dimensional function f(x,y), where x and y are spatial coordinates, and the amplitude of f at any pair of coordinates (x,y) is called the intensity (or) gray level of the image at that point. When x, y and the amplitude values of "f" are all finite, discrete quantities. We call the image a digital image. The field of DIP refers to processing digital image by means of digital computer. Digital image is composed of a finite number of elements, each of which has a particular location and value. The elements are called pixels.

Vision is the most advanced of our sensor, so it is not surprising that image play the single most important role in human perception. However, unlike humans, who are limited to the visual band of the EM spectrum, imaging machines cover almost the entire EM spectrum, ranging from gamma to radio waves. They can operate also on images generated by sources that humans are not accustomed to associating with image. These include ultra sound, electron microscopy, and computer- generated images. Thus, digital image processing encompasses a wide and various fields of applications. There is no general agreement among authors regarding where image processing stops and other related areas, such as image analysis and computer vision, start. Sometimes a distinction is made by defining image processing as a discipline in which both the input and output at a process are images. This is limiting and somewhat artificial boundary. The area of image analysis (image understanding) is in between image processing and computer vision.

There are no clear-cut boundaries in the continuum from image processing at one end to complete vision at the other. However, one useful paradigm is to consider three types of computerized processes in this continuum: low-, mid-, and high-level processes. Low-level processes involve primitive operations such as image processing to reduce noise, contrast enhancement, and image sharpening. A low- level process is characterized by the fact that both its inputs and outputs are images. Mid-level processes on images involve tasks such as segmentation, description of that object to reduce them to a form suitable for computer processing, and classification of individual objects. A mid-level process is characterized by the fact that its inputs generally are images, but its outputs are attributes extracted from those images; finally higher- level processing involves "Making sense" of an ensemble of recognized objects, as in image analysis, and at the far end of the continuum, performing the cognitive functions normally associated with human vision. Digital image processing, as already defined, is used successfully in a broad range of areas of exceptional social and economic value.

2. IMAGE COMPRESSION TECHNIQUES

One of the important aspects of image storage is its efficient compression. To make this fact clear let's see an example. An image, 1024 pixel x 1024 pixel x 24 bit, without compression, would require 3 MB of storage and 7 minutes for transmission, utilizing a high speed, 64 Kbit/s, ISDN line. If the image is compressed at a 10:1 compression ratio, the storage requirement is reduced to 300 KB and the transmission time drops to under 6 seconds. Seven 1 MB images can be compressed and transferred to a floppy disk in less time than it takes to send one of the original files, uncompressed, over an AppleTalk network.

In a distributed environment large image files remain a major bottleneck within systems. Compression is an important component of the solutions available for creating file sizes of manageable and transmittable dimensions. Increasing the bandwidth is another method, but the cost sometimes makes this a less attractive solution. Platform portability and performance are important in the selection of the compression/decompression technique to be employed. Compression solutions today are more portable due to the change from proprietary high end solutions to accepted and implemented international standards. JPEG is evolving as the industry standard technique for the compression of continuous tone images.

2.1 Data Compression Techniques: Two categories of data compression algorithm can be distinguished: lossless and 'lossy'. Lossy techniques cause image quality degradation in each compression/decompression step. Careful consideration of the human visual perception ensures that the degradation is often unrecognizable, though this depends on the selected compression ratio.

In general, lossy techniques provide far greater compression ratios than lossless techniques. Different types of data compression techniques are explained below.

Lossless Coding Techniques

- Run length encoding
- Huffman encoding
- Entropy coding (Lempel/Ziv)
- Area coding

Lossy Coding Techniques

- Transform coding (DCT/Wavelets/Gabor)
- Vector quantization
- Segmentation and approximation methods
- Spline approximation methods (Bilinear Interpolation/Regularization)
- Fractal coding (texture synthesis, iterated functions system [IFS], recursive IFS [RIFS])
- Efficiency and quality of different lossy compression techniques

2.1.1 Lossless Coding Techniques: Lossless coding guaranties that the decompressed image is absolutely identical to the image before compression. This is an important requirement for some application domains, e.g. medical imaging, where not only high quality is in demand, but unaltered archiving is a legal requirement. Lossless techniques can also use for the compression of other data types where loss of information is not acceptable, e.g. text documents and program executable.

Some compression methods can be made more effective by adding a 1D or 2D delta coding to the process of compression. These deltas make more effectively use of run length encoding, have (statistically) higher maxima in code tables (leading to better results in Huffman and general entropy coding), and build greater equal value areas usable for area coding. Some of these methods can easily be modified to be lossy. Lossy element fits perfectly into 1D/2D run length search. Also, logarithmic quantization may be inserted to provide better or more effective results.

2.1.2. Huffman Encoding: This algorithm, developed by D.A. Huffman, is based on the fact that in an input stream certain tokens occur more often than others. Based on this knowledge, the algorithm builds up a weighted binary tree according to their rate of occurrence. Each element of this tree is assigned a new code word, whereat the length of the code word is determined by its position in the tree. Therefore, the token which is most frequent and becomes the root of the tree is assigned the shortest code. Each less common element is assigned a longer code word. The least frequent element is assigned a code word which may have become twice as long as the input token.

The compression ratio achieved by Huffman encoding uncorrelated data becomes something like 1:2. On slightly correlated data, as on images, the compression rate may become much higher, the absolute maximum being defined by the size of a single input token and the size of the shortest possible output token (max. compression = token size[bits]/2[bits]). While standard palletized images with a limit of 256 colors may be compressed by 1:4 if they use only one color, more typical images give results in the range of 1:1.2 to 1:2.5.

Huffman Encoding Steps

- Label each node with one of the source symbol probabilities
- Merge the nodes labeled by the two smallest probabilities into a parent node

- Label the parent node with the sum of the two children's probabilities
- Among the elements in reduced alphabet, merge two with smallest probabilities.
- Label the parent node wth the sum of the two children probabilities.
- Repeat steps 4 & 5 until only a single super symbol remains

2.2. Transform Coding (DCT/Wavelets/Gabor): A general transform coding scheme involves subdividing an NxN image into smaller nxn blocks and performing a unitary transform on each sub image. A unitary transform is a reversible linear transform whose kernel describes a set of complete, orthonormal discrete basic functions. The goal of the transform is to decorrelate the original signal, and this decorrelation generally results in the signal energy being redistributed among only a small set of transform coefficients. In this way, many coefficients may be discarded after quantization and prior to encoding. Also, visually lossless compression can often be achieved by incorporating the HVS contrast sensitivity function in the quantization of the coefficients. Transform coding can be generalized into four stages:

1. Image subdivision
2. Image transformation
3. Coefficient quantization
4. Huffman encoding.

For a transform coding scheme, logical modeling is done in two steps: a segmentation one, in which the image is subdivided in dimensional vectors (possibly of different sizes) and a transformation step, in which the chosen transform (e.g. KLT, DCT, Hadamard) is applied.

Quantization can be performed in several ways. Most classical approaches use 'zonal coding', consisting in the scalar quantization of the coefficients belonging to a predefined area (with a fixed bit allocation), and 'threshold coding', consisting in the choice of the coefficients of each block characterized by an absolute value exceeding a predefined threshold. Another possibility, that leads to higher compression factors, is to apply a vector quantization scheme to the transformed coefficients. The same type of encoding is used for each coding method. In most cases a classical Huffman code can be used successfully. The JPEG and MPEG standards are examples of standards based on transform coding.

3. DISCRETE COSINE TRANSFORM

3.1 Fourier Analysis: Signal analysts already have at their disposal an impressive arsenal of tools. Perhaps the most well-known of these is Fourier analysis, which breaks down a signal into constituent sinusoids of different frequencies. Another way to think of Fourier analysis is as a mathematical technique for transforming our view of the signal from time-based to frequency-based. For many signals, Fourier analysis is extremely useful because the signal's frequency content is of great importance. So we need other techniques, like DCT analysis. Fourier analysis has a serious drawback. In transforming to the frequency domain, time information is lost. When looking at a Fourier transform of signal, it is impossible to tell when a particular event took place.

If the signal properties do not change much over time — that is, if it is what is called a stationary signal—this drawback isn't very important. However, most interesting signals contain numerous no stationary or transitory characteristics: drift, trends, abrupt changes, and beginnings and ends of events. These characteristics are often the most important part of the signal, and Fourier analysis is not suited to detecting them.

3.2 Short-Time Fourier analysis: In an effort to correct this deficiency, Dennis Gabor (1946) adapted the Fourier transform to analyze only a small section of the signal at a time—a technique called windowing the signal. Gabor's adaptation, called the Short-Time Fourier Transform (STFT), maps a signal into a two-dimensional function of time and frequency. The STFT represents a sort of compromise between the time- and frequency-based views of a signal. It provides some information about both when and at what frequencies a signal event occurs. However, you can only obtain this information with limited precision, and that precision is determined by the size of the window.

While the STFT compromise between time and frequency information can be useful, the drawback is that once you choose a particular size for the time window, that window is the same for all frequencies. Many signals require a more flexible approach one where we can vary the window size to determine more accurately either time or frequency.

3.3 DCT Analysis: Just as the Fourier transform uses sine and cosine waves to represent a signal, the DCT only uses cosine waves. There are several versions of the DCT, with slight differences in their mathematics. As an example of one version, imagine a 129 point signal, running from sample 0 to sample 128. Now, make this a 256 point signal by duplicating samples 1 through 127 and adding them as samples 255 to 130. That is: 0,1,2,3...127,128,127...2,1.

Taking the Fourier transform of this 256 point signal results in a frequency spectrum of 129 points, spread between 0 and 128. Since the time domain signal was forced to be symmetrical, the spectrum's imaginary part will be composed of all zeros. When the DCT is taken of an 8x8 group, it results in an 8x8 spectrum. In other words, 64 numbers are changed into 64 other numbers. All these values are real; there is no complex mathematics here. Just as in Fourier analysis, each value in the spectrum is the amplitude of a basis function. Figure 3.1 shows 6 of the 64 basis functions used in an 8x8 DCT, according to where the amplitude sits in the spectrum. The 8x8 DCT basis functions are given by:

$$b[x,y] = \cos\left[\frac{(2x+1)\,u\pi}{16}\right]\cos\left[\frac{(2y+1)\,v\pi}{16}\right]$$

DCT basis functions. The variables x & y are the indexes in the spatial domain, and u & v are the indexes in the frequency spectrum. This is for an 8x8 DCT, making all the indexes run from 0 to 7. The low frequencies reside in the upper-left corner of the spectrum, while the high frequencies are in the lower-right. The DC component is at [0,0], the upper-left most value. The basis function for [0, 1] is one-half cycle of a cosine wave in one direction, and a constant value in the other. The basis function for [1,0] is similar, just rotated by 90 degrees.

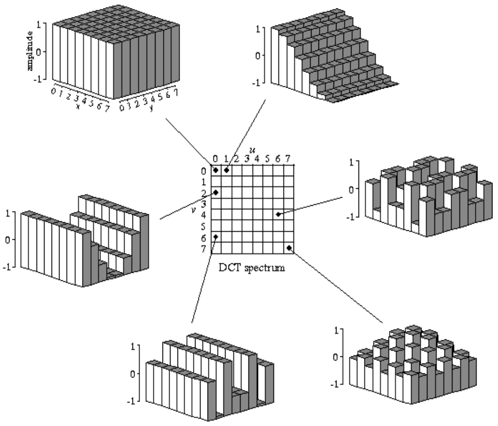


Fig. 1. DCT Spectrum

The DCT basis functions. The DCT spectrum consists of an 8x8 array, with each element in the array being amplitude of one of the 64 basis functions. Six of these basis functions are shown here, referenced to where the corresponding amplitude resides. The DCT calculates the spectrum by correlating the 8x8 pixel group with each of the basic functions. That is, each spectral value is found by multiplying the appropriate basis function by the 8x8 pixel group, and then summing the products.

Two adjustments are then needed to finish the DCT calculation (just as with the Fourier transform). First, divide the 15 spectral values in row 0 and column 0 by two. Second, divide all 64 values in the spectrum by 16. The inverse DCT is calculated by assigning each of the amplitudes in the spectrum to the proper basis function, and summing to recreate the spatial domain. No extra steps are required. These are exactly the same concepts as in Fourier analysis, just with different basis functions.

The DCT better than the Fourier transform for image compression because the DCT has one-half cycle basis functions, i.e., S[0,1] and S[1,0]. As shown in Fig.3.1, these gently slope from one side of the array to the other. In comparison, the lowest frequencies in the Fourier transform form one complete cycle. Images nearly always contain regions where the brightness is gradually changing over a region. Using a basis function that matches this basic pattern allows for better compression.

3.3.1 Formal definition: Formally, the discrete cosine transform is a linear, invertible function F: R^N -> R^N (where R denotes the set of real numbers), or equivalently an N × N square matrix. There are several variants of the DCT with slightly modified definitions. The N real numbers x_0... x_{N-1} are transformed into the N real numbers f_0, ..., f_{N-1} according to one of the formulas:

DCT-I:

$$f_j = \frac{1}{2}(x_0 + (-1)^j x_{N-1}) + \sum_{n=1}^{N-2} x_n \cos\left[\frac{\pi}{N-1} jn\right]$$

Some authors further multiply the x_0 and x_{N-1} terms by √2, and correspondingly multiply the f_0 and f_{N-1} terms by 1/√2. This makes the DCT-I matrix orthogonal (up to a scale factor), but breaks the direct correspondence with a real-even DFT. A DCT-I of N=5 real numbers abcde is exactly equivalent to a DFT of eight real numbers abcdedcb (even symmetry), here divided by two. (In contrast, DCT types II-IV involve a half-sample shift in the equivalent DFT.) Note, however, that the DCT-I is not defined for N less than 2. (All other DCT types are defined for any positive N.)Thus, the DCT-I corresponds to the boundary conditions: x_n is even around n=0 and even around n=N-1; similarly for f_j.

DCT-II:

$$f_j = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N} j \left(n + \frac{1}{2}\right)\right]$$

The DCT-II is probably the most commonly used form, and is often simply referred to as "the DCT". This transform is exactly equivalent (up to an overall scale factor of 2) to a DFT of 4N real inputs of even symmetry where the even-indexed elements are zero. That is, it is half of the DFT of the 4N inputs y_n, where $y_{2n} = 0$, $y_{2n+1} = x_n$ for , and $y_{4N-n} = y_n$ for 0 < n < 2N.The f_0 term by 1/√2 (see below for the corresponding change in DCT-III). This makes the DCT-II matrix orthogonal (up to a scale factor), but breaks the direct correspondence with a real-even DFT of half-shifted input. The DCT-II implies the boundary conditions: x_n is even around n=-1/2 and even around n=N-1/2; f_j is even around j=0 and odd around j=N.

DCT-III:

$$f_j = \frac{1}{2}x_0 + \sum_{n=1}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(j + \frac{1}{2}\right) n\right]$$

Because it is the inverse of DCT-II (up to a scale factor, see below), this form is sometimes simply referred to as "the inverse DCT" ("IDCT"). The x_0 term is multiplied by √2 (see above for the corresponding change in DCT-II). This makes the DCT-III matrix orthogonal (up to a scale factor), but breaks the direct correspondence with a real-even DFT of half-shifted output.

The DCT-III implies the boundary conditions: x_n is even around n=0 and odd around n=N; f_j is even around j=-1/2 and odd around j=N-1/2.

DCT-IV:

$$f_j = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(j+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)\right]$$

The DCT-IV matrix is orthogonal (up to a scale factor).A variant of the DCT-IV, where data from different transforms are overlapped, is called the modified discrete cosine transform (MDCT).The DCT-IV implies the boundary conditions: x_n is even around n=-1/2 and odd around n=N-1/2; similarly for f_j.

DCT V-VIII:

DCT types I-IV are equivalent to real-even DFTs of even order. In principle, there are actually four additional types of discrete cosine transform (Martucci, 1994), corresponding to real-even DFTs of logically odd order, which have factors of N+1/2 in the denominators of the cosine arguments. However, these variants seem to be rarely used in practice.(The trivial real-even array, a length-one DFT (odd length) of a single number a, corresponds to a DCT-V of length N=1.)

The Process: The following is a general overview of the JPEG process.

- The image is broken into 8x8 blocks of pixels.
- Working from left to right, top to bottom, the DCT is applied to each block.
- Each block is compressed through quantization.
- The array of compressed blocks that constitute the image is stored in a drastically reduced amount of space.
- When desired, the image is reconstructed through decompression, a process that uses the Inverse Discrete Cosine Transform (IDCT).

3.4 Block Diagram of Image Compression:

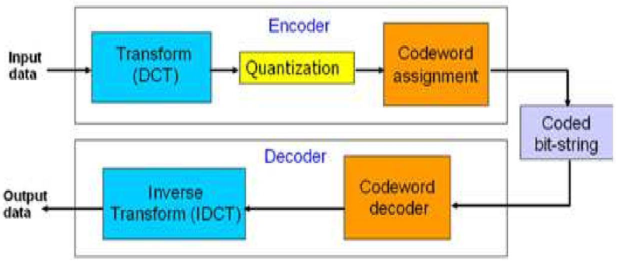
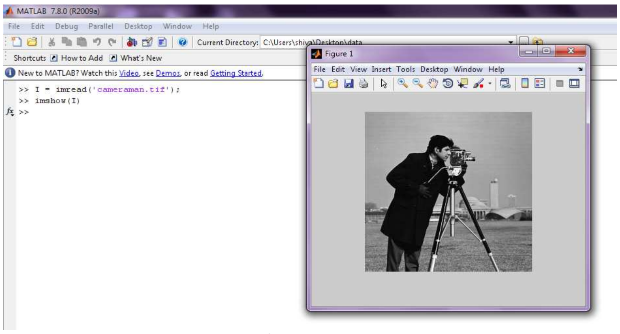


Fig.2. Block Diagram of Image compression

3.5 DCT compared to the DFT: The discrete cosine transform (DCT) is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. It is equivalent to a DFT of roughly twice the length, operating on real data with even symmetry (since the Fourier transform of a real and even function is real and even), where in some variants the input and/or output data are shifted by half a sample. (There are eight standard variants, of which four are common.)

4. SIMULATION RESULTS

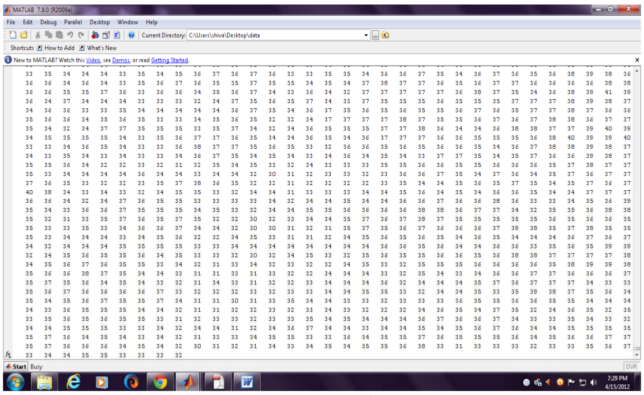
In this chapter the simulation results and snapshots of interfacing with the FPGA module (Spartan-3) are included. Input of our project is an image and that image is compressed at transmission side and decompressed at the reception side. The image is read by using matlab commands as shown below.


MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
Current Directory: C:\Users\shiva\Desktop\data
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Figure 1
File Edit View Insert Tools Desktop Window Help

```
>> I = imread('cameraman.tif');
>> imshow(I)
>>
```

Fig.3. Reading of an Image through Matlab

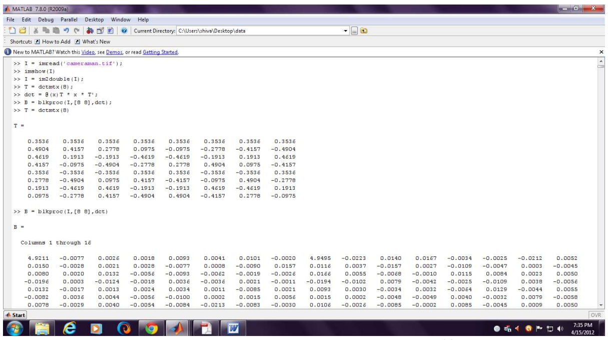
The image is reading by using the function imread() as shown in figure8.1 and the output will contain image values which will give by the following window.


MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
Current Directory: C:\Users\shiva\Desktop\data
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Start Busy
OVR
7:29 PM
4/15/2012

```
33 35 34 34 34 33 35 34 35 36 37 36 37 36 33 33 35 35 34 36 36 37 35 34 36 37 36 35 36 38 39 38 36
36 36 34 36 34 33 35 36 37 36 35 35 37 35 35 34 35 34 37 38 37 37 36 35 36 37 37 36 36 36 36 38 38
36 36 35 35 37 36 33 36 36 34 35 36 37 34 33 36 34 32 37 37 37 37 37 36 38 37 35 34 36 38 39 41 39
36 34 37 34 34 34 33 33 33 32 34 37 35 36 35 37 34 33 37 35 35 35 36 35 35 35 37 37 37 38 39 38 37
34 36 36 33 33 35 34 34 34 34 34 36 37 35 34 36 37 35 36 35 36 35 35 36 37 36 35 37 37 38 37 36 36
35 36 36 34 35 36 35 33 33 34 35 36 35 32 32 34 37 37 37 37 38 37 35 35 36 37 36 37 38 38 36 37 37
35 34 32 34 37 37 35 35 35 33 35 37 34 32 34 36 35 35 35 37 37 38 36 34 34 36 38 38 37 37 39 40 39
34 35 35 35 35 34 33 35 36 37 37 36 35 34 34 36 35 34 36 37 37 37 36 36 35 35 35 36 38 40 39 39 40
33 33 34 36 35 34 33 33 36 38 37 37 35 36 35 33 32 36 36 35 36 35 36 36 35 34 36 37 38 38 39 38 37
34 33 35 34 33 34 33 33 34 36 37 35 34 35 34 33 34 36 34 35 34 33 37 37 35 34 35 37 36 36 39 38 37
35 35 36 34 32 32 33 32 31 32 35 34 35 33 32 34 33 33 33 35 35 36 36 35 35 36 36 36 35 37 38 37 37
35 33 34 34 34 34 36 34 34 33 34 34 32 30 31 32 33 33 32 33 36 36 37 35 34 37 36 34 35 37 36 37 37
37 36 35 33 32 32 33 35 37 38 36 35 32 32 31 32 32 32 32 33 35 34 34 35 36 35 37 35 34 35 37 36 37
40 38 34 33 34 33 32 34 35 35 33 32 34 34 31 33 33 33 34 34 35 36 34 35 34 36 36 35 34 34 37 37 37
36 36 34 32 34 37 36 35 35 33 33 33 33 34 32 34 34 35 34 34 36 36 37 36 36 38 36 33 33 34 35 36 39
35 34 33 36 36 37 35 35 35 34 35 33 32 34 34 35 35 36 36 36 36 38 38 36 37 37 34 32 35 35 36 38 38
35 32 31 33 35 37 36 35 37 35 32 32 30 32 33 34 34 35 37 36 37 38 37 35 35 35 35 35 36 35 36 36 35
35 33 33 35 33 34 36 36 37 34 34 32 30 30 31 32 31 35 37 35 36 37 36 36 36 37 39 38 35 37 38 35 35
35 33 34 34 34 33 34 35 36 32 32 34 35 33 31 31 32 34 35 36 35 36 35 34 36 35 34 34 34 36 37 36 37
34 32 34 34 34 35 35 35 35 33 33 34 34 34 34 34 34 34 34 36 36 35 36 34 34 36 36 33 35 36 35 39 39
32 34 35 36 35 35 36 34 35 33 33 32 30 32 34 35 33 32 35 36 35 35 36 36 35 36 38 38 37 37 37 37 38
34 35 36 37 36 35 35 33 34 32 31 33 34 32 33 32 32 34 35 33 32 35 35 35 36 36 36 36 35 38 39 39 38
35 36 36 38 37 35 34 34 33 31 31 33 31 33 32 32 34 34 34 33 32 35 34 33 34 36 36 37 37 36 36 36 37
35 37 35 36 34 35 34 33 32 31 34 33 31 32 32 33 34 34 34 36 32 34 34 34 35 37 36 37 37 37 34 33 33
35 36 37 36 36 36 36 37 33 32 32 32 33 32 33 33 34 34 35 35 33 32 34 34 35 33 35 39 38 37 35 36 34
35 34 35 36 37 35 35 37 34 31 31 30 31 33 35 34 34 33 33 32 33 33 33 36 35 35 36 36 35 35 34 34 34
34 33 36 35 35 35 34 34 32 31 31 32 32 33 32 33 34 33 32 32 32 34 36 35 34 37 35 32 34 36 35 32 35
33 35 36 35 36 35 35 33 31 32 33 33 32 33 33 35 34 35 34 34 34 36 37 36 36 37 34 33 33 35 34 33 32
34 34 35 35 35 33 33 34 32 34 34 31 32 34 36 37 34 34 33 34 34 35 34 35 36 37 36 34 34 35 35 35 35
35 37 36 34 35 34 33 34 32 31 33 33 34 35 35 33 33 36 34 35 35 37 36 35 35 35 36 34 35 36 36 37 37
35 37 36 36 36 34 35 34 32 30 31 32 31 34 33 34 35 34 35 35 36 38 33 31 33 33 33 32 33 33 35 36 37
33 34 34 35 35 33 33 33 32
```

Fig.4: Image Coefficients

After reading the image values the one dimensional DCT operation is performed by using the command dctmtx() and the two dimensional 8X8 block operation is performed by using the command blkproc(Image, [8 8], dct).


MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
Current Directory: C:\Users\shiva\Desktop\data
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Start
OVR
7:35 PM
4/15/2012

```
>> I = imread('cameraman.tif');
>> imshow(I)
>> I = im2double(I);
>> T = dctmtx(8);
>> dct = @(x)T * x * T';
>> B = blkproc(I,[8 8],dct);
>> T = dctmtx(8)

T =

    0.3536    0.3536    0.3536    0.3536    0.3536    0.3536    0.3536    0.3536
    0.4904    0.4157    0.2778    0.0975   -0.0975   -0.2778   -0.4157   -0.4904
    0.4619    0.1913   -0.1913   -0.4619   -0.4619   -0.1913    0.1913    0.4619
    0.4157   -0.0975   -0.4904   -0.2778    0.2778    0.4904    0.0975   -0.4157
    0.3536   -0.3536   -0.3536    0.3536    0.3536   -0.3536   -0.3536    0.3536
    0.2778   -0.4904    0.0975    0.4157   -0.4157   -0.0975    0.4904   -0.2778
    0.1913   -0.4619    0.4619   -0.1913   -0.1913    0.4619   -0.4619    0.1913
    0.0975   -0.2778    0.4157   -0.4904    0.4904   -0.4157    0.2778   -0.0975

>> B = blkproc(I,[8 8],dct)

B =

  Columns 1 through 16

    4.9211   -0.0077    0.0026    0.0018    0.0093    0.0041    0.0101   -0.0020    4.9495   -0.0223    0.0140    0.0167   -0.0034   -0.0025   -0.0212    0.0052
    0.0150   -0.0028    0.0021    0.0028   -0.0077    0.0008   -0.0090    0.0157    0.0116    0.0037   -0.0157    0.0027   -0.0109   -0.0047    0.0003   -0.0045
    0.0080    0.0020    0.0132   -0.0056   -0.0093   -0.0062   -0.0019   -0.0026    0.0166    0.0055   -0.0068   -0.0010    0.0115    0.0084    0.0023    0.0050
   -0.0196    0.0003   -0.0124   -0.0018    0.0036   -0.0036    0.0021   -0.0011   -0.0194   -0.0102    0.0079   -0.0042   -0.0025   -0.0109    0.0038   -0.0056
    0.0132   -0.0017    0.0013    0.0024    0.0034    0.0011   -0.0085    0.0021    0.0093    0.0030   -0.0034    0.0032   -0.0064    0.0129   -0.0044    0.0055
   -0.0082    0.0036    0.0044   -0.0056   -0.0100    0.0002    0.0015    0.0056    0.0015    0.0002   -0.0048   -0.0049    0.0040   -0.0032    0.0079   -0.0058
    0.0078   -0.0029    0.0040   -0.0054   -0.0084   -0.0213   -0.0083   -0.0030    0.0106   -0.0026   -0.0085   -0.0002    0.0085   -0.0045    0.0009    0.0050
```

Fig.5. DCT Operation on the Image Coefficients

After performing the DCT operations the values are encode using Huffman encoding technique. For this the coding was written in VHDL. The Huffman encoding and decoding is performed and after that the inverse DCT operation is performed using the command invdct().

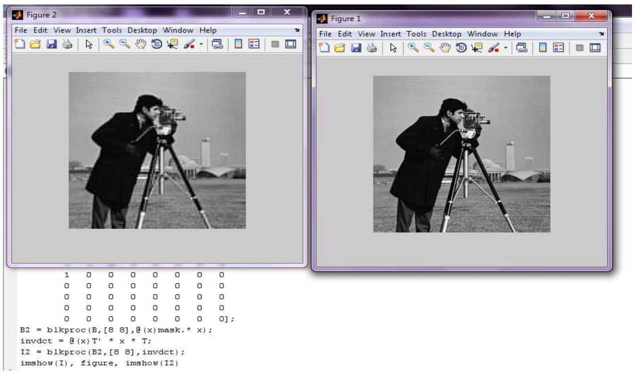


Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help

```
        1    0    0    0    0    0    0    0
        0    0    0    0    0    0    0    0
        0    0    0    0    0    0    0    0
        0    0    0    0    0    0    0    0
        0    0    0    0    0    0    0    0];
B2 = blkproc(B,[8 8],@(x)mask.* x);
invdct = @(x)T' * x * T;
I2 = blkproc(B2,[8 8],invdct);
imshow(I), figure, imshow(I2)
```

Fig.6. IDCT Operation on the Image Coefficients

The simulation results using Xilinx are as follows.

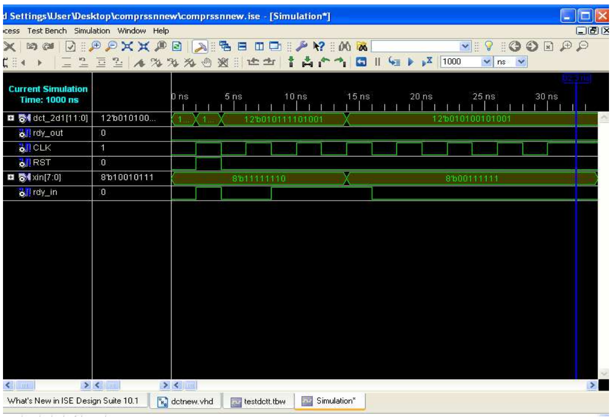


Fig.7. Output test bench wave form of DCT

The above figure represents the output test bench wave forms of Discrete Cosine Transformation for the inputs named as xin(8’b11111110). The output is obtained only when the clock is in active state i.e., 1 also the reset should be zero.

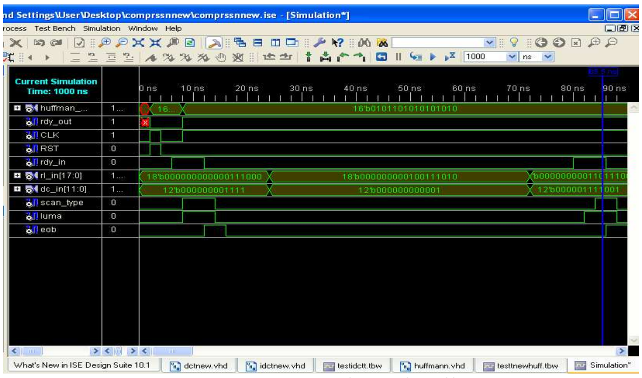


Fig.8. Output test bench wave form of Huffman coding

The above figure represents the output test bench wave forms of Huffman coding which includes encoding and decoding. The inputs here are r1_in and dc in and the output is Huffman out as shown in the figure.

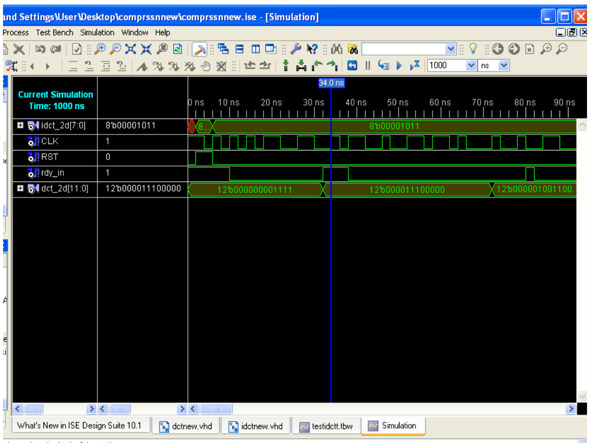


Fig.9.Output test bench wave form of IDCT

The above figure represents the output test bench wave forms of Inverse Discrete Cosine Transformation for the Discrete Cosine Transformation input (12’b000000001111).

REFERENCES

[1]. *Digital image Processing*,R.C. Gonzalez & R.E.Woods, Addison Wesley/Peason education,2nd edition,2002.

[2]. K. Sayood and J. C. Borkenhagen, “*Use of residual redundancy in the design of joint source/channel coders*,” IEEE Transactions on Communications, 39(6):838-846, June 1991.

[3]. N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete cosine transform,” IEEE Transactions on Computers, vol. C-32, pp. 90-93, Jan. 1974.

[4]. R. J. Clark, “Transform Coding of Images,” New York: Academic Press, 1985.

[5]. P. Duhamel and M. Vetterli, "Fast Fourier transforms: a tutorial review and a state of the art," *Signal Processing* **19**, 259–299 (1990).

[6]. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Section 16.3, pp. 385–392.

[7]. http://en.wikipedia.org/wiki/Discrete_cosine_transform

[8]. http://vision.arc.nasa.gov/publications/mathjournal94.pdf

[9]. http://en.wikipedia.org/wiki/Huffman_coding

[10]. http://www.dcd.zju.edu.cn/~jun/Courses/Multimedia2011.../complementary/DCT_Theory%20and%20Application.pdf

About the authors:

T.Krishnarjuna Rao received B. Tech degree in Electronics and communication Engineering from the University of JNTU Hyderabad and M.Tech in VLSI System Design from the JNTU Hyderabad**.** He is currently working as an Assoc. Professor in ECE department at RVR Institute of Engineering & Technology, Hyderabad. Up to now attended several National and International conferences, Workshops. I am very interest to research on VLSI Design, Communications, Image processing, Digital signal processing.

P. Aruna Sreee received B.Tech degree in Electronics and Instrumentation Engineering from the University of JNTU and M.Tech in in Embedded Systems in JNTU Hyderabad. She is currently working as an Assistant. Professor in ECE department at RVR Institute of Engineering & Technology, Hyderabad. Up to now attended several National & International Conferences, Workshops. Research interests in Embedded systems, VLSI, Control and Instrumentation.

Kumar. Keshamoni obtained his B.Tech in Electronics & Communication Engineering from the University of JNTU, his M.Tech in VLSI Design from the University of JNTU-Hyderabad. He is currently the Asst. Professor in Department of Electronics & Communication Engineering, RVR Institute of Engineering in University of the JNTU-Hyderabad. Kumar. Keshamoni is Chair of World Committees Associate Member of UACEE (Universal Association of Computer & Electronics Engineers), Member of IEEE Vehicle Community, Member of IAENG and an Advisory Board of WATT (World Association of Technology Teachers). He is also on the Advisory & Editorial Board of IJETAE, an Associate Editor of IJPRET, Editor of IJOART, Editorial Board & Review Committee of IJSER, an Editorial & Review Board of IJTRA, Editor, writer & Peer Reviewer of edition International Publisher, Peer Reviewer of AJET, Review Committee of IJERT, Editorial & Review Board of IJOAR, Review Board of IJASCSE, Advisory& Editorial Board of IJRDET, Editorial Board of IJEERT Member of IAENG, Writer& Review Board of IJEVS, Advisory Board of WRJET, Scientific Committee & Editorial Review Board on Engineering and Physical Sciences of WASET and Editorial Board of IAASSE. He is also working as a member for 10 International conference committees. He has published about 20 Research articles in International Journals and 5 papers in National and International Conferences and attended several National, International Conferences, and Workshops & Faculty Development Programs in different organizations. His recent research interests lie in the area of Embedded, VLSI Design, Digital Signal Processing and Nanotechnology.