World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:4, No:12, 2010

A Novel VLSI Architecture for Image Compression Model Using Low power Discrete Cosine Transform

Vijaya Prakash.A.M , K.S.Gurumurthy

Abstract—In Image processing the Image compression can improve the performance of the digital systems by reducing the cost and time in image storage and transmission without significant reduction of the Image quality. This paper describes hardware architecture of low complexity Discrete Cosine Transform (DCT) architecture for image compression[6]. In this DCT architecture, common computations are identified and shared to remove redundant computations in DCT matrix operation. Vector processing is a method used for implementation of DCT. This reduction in computational complexity of 2D DCT reduces power consumption. The 2D DCT is performed on 8x8 matrix using two 1-Dimensional Discrete cosine transform blocks and a transposition memory [7]. Inverse discrete cosine transform (IDCT) is performed to obtain the image matrix and reconstruct the original image. The proposed image compression algorithm is comprehended using MATLAB code. The VLSI design of the architecture is implemented Using Verilog HDL. The proposed hardware architecture for image compression employing DCT was synthesized using RTL complier and it was mapped using 180nm standard cells. . The Simulation is done using Modelsim. The simulation results from MATLAB and Verilog HDL are compared. Detailed analysis for power and area was done using RTL compiler from CADENCE. Power consumption of DCT core is reduced to 1.027mW with minimum area[1].

Keywords—Discrete Cosine Transform (DCT), Inverse Discrete Cosine Transform (IDCT), Joint Photographic Expert Group (JPEG), Low Power Design, Very Large Scale Integration (VLSI) .

I. Introduction

IMage data compression has been an active research area for image processing [1] over the last decade and has been used in a variety of applications. This paper investigates the implementation of an image data compression method with VLSI hardware that could be used in practical coding systems to compress JPEG signals.

In practical situations, an image is originally defined over a large matrix of picture elements (pixels), with each pixel represented by a 8- or 16-bit gray scale value. This representation could be so large that it is difficult to store or transmit. The purpose of image compression is to reduce the size of the representation and, at the same time, to keep most of the information contained in the original image [10]. DCT based coding and decoding systems play a dominant role in real-time applications. However, the DCT is computationally intensive. In addition, 2D-DCT has been recommended by standard organizations the Joint Photographic Expert Group (JPEG). The standards developed by these groups aid industry manufacturers in developing real-time 2D-DCT chips for use in various image transmission and storage systems. The sequential DCT based mode of operation comprises the baseline JPEG algorithm. This technique can produce very good compression ratios, while sacrificing image quality. The sequential DCT based mode achieves much of its compression through quantization, which removes entropy from the data set. Although this baseline algorithm is transform based, it does use some measure of predictive coding called the differential pulse code modulation (DPCM). After each input 8x8 block of pixels is transformed to frequency space using the DCT, the resulting block contains a single DC component, and 63 AC components. The DC component is predicatively encoded through a difference between the current DC value and the previous [8]. This mode only uses Huffman coding models, not arithmetic coding models which are used in JPEG extensions. This mode is the most basic, but still has a wide acceptance for its high compression ratios, which can fit many general applications very well.

The remaining paper is organized as follows, Section 2 explains the DCT Equations;Section 3 explains the DCT Process and computation reduction techniques. Section 4 describes the IDCT process, Section 5 presents the MATLAB implementation of DCT algorithm. The proposed architectures and Interpretation of Results are presented in Section 6. Then, the synthesis results and conclusions are given in Section 7.

Research scholar, Dr.MGR University, Chennai, India, am_vprakash@yahoo.co.in
Professor, Department of ECE, UVCE, Bangalore, India, drksgurumurthy@gmail.com

II. DCT Equations

DCT algorithm is very effective due to its symmetry and simplicity. It is good replacement of FFT due to consideration of real component of the image data. In DCT we leave the unwanted frequency[3] components while opting only required frequency components of the image. Image is divided into blocks and each block is compressed using quantization. Moreover, many simulation tools like MATLAB are available to estimate the results prior to realization of design in real time. Equations 1 and 2 gives the one dimension DCT standard equations for the data out.

A. One Dimensional DCT

The most common DCT definition of a 1-D sequence of length N is:

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

For u=0, 1, 2, N-1. Similarly, the inverse transformation is defined as

$$f(x) = \sum_{x=0}^{N-1} \alpha(u)c(u)cos\left[\frac{\pi(2x+1)u}{2N}\right] \quad (2)$$

For x = 0,1,2,N-1. In both equations (1) and (2) $\alpha(u)$ is defined as

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u \neq 0 \end{cases} \quad (3)$$

It is clear from (1) that for

$$u = 0, c(u = 0) = \sqrt{\frac{1}{N}} \sum_{x=0}^{N-1} f(x) \quad (4)$$

Thus, the first transform coefficient is the average value of the sample sequence. In literature, this value is referred to as the DC Coefficient. All other transform coefficients are called the AC Coefficients.

B. Two-Dimensional DCT

The 2-D DCT is a direct extension of the 1-D case and is given by

$$C(u,v) = \alpha(u)\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)cos\left[\frac{\pi(2x+1)u}{2N}\right] cos\left[\frac{\pi(2y+1)v}{2N}\right] \quad (5)$$

For u,v = 0,1, 2, N-1 and (u) and (v) are defined in Equation (4)

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v)c(u,v)cos\left[\frac{\pi(2x+1)u}{2N}\right] cos\left[\frac{\pi(2y+1)v}{2N}\right] \quad (6)$$

The inverse transform is defined as for x, y = 0, 1, 2, N-1.The 2-D basis functions can be generated by multiplying the horizontally oriented 1-D basis functions with vertically oriented set of the same functions.

III. DCT Process

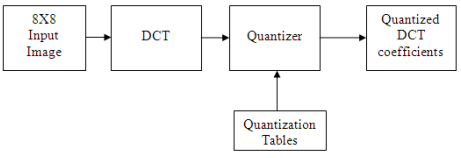

Fig. 1: Block diagram of DCT Process

In the DCT process input image is divided into non-overlapping blocks of 8 x 8 pixels, and input to the baseline encoder. The pixel values are converted from unsigned integer format to signed integer format, and DCT computation is performed on each block. DCT transforms the pixel data into a block of spatial frequencies that are called the DCT coefficients. Since the pixels in the 8 x 8 neighborhood typically have small variations in gray levels, the output of the DCT will result in most of the block energy being stored in the lower spatial frequencies. On the other hand, the higher frequencies will have values equal to or close to zero and hence, can be ignored during encoding without significantly affecting the image quality[4]. The selection of frequencies based on which frequencies are most important and which ones are less important can affect the quality of the final image.

The selection of quantization values is critical since it affects both the compression efficiency, and the reconstructed image quality.

High frequency coefficients have small magnitude for typical video data, which usually does not change dramatically between neighboring pixels. Additionally, the human eye is not as sensitive to high frequencies as to low frequencies. It is difficult for the human eye to discern changes in intensity or colors that occur between successive pixels. The human eye tends to blur these rapid changes into an average hue and intensity. However, gradual changes over the 8 pixels in a block are much more discernible than rapid changes. When the DCT is used for compression purposes, the quantizer unit attempts to force the insignificant high frequency coefficients to zero while retaining the important low frequency coefficients[4]. The 2-D DCT transforms an 8x8 block of spatial data samples into an 8x8 block of spatial frequency components. The IDCT performs the inverse of DCT, transforming spatial frequency components back into the spatial domain.

A. Proposed Two Dimensional DCT Architecture

The Discrete Cosine Transform is an orthogonal transform consisting of a set of basis vectors that are sampled cosine functions. The 2-D DCT of a data matrix is defined as,

$$Y = C.X.C^T \quad (7)$$

Where X is the data matrix, C is the matrix of DCT Coefficients, and C^T is the Transpose of C. The 2-D DCT (8 x 8 DCT) is implemented by the row-column decomposition technique. We first compute the 1-D DCT (8 x 1 DCT) of each column of the input data matrix X to yield CT X. after appropriate rounding or truncation, the transpose of the resulting matrix, CTX, is stored in an intermediate memory. We then compute another 1-D DCT (8 x 1 DCT) of each row of CTX to yield the desired 2-D DCT as defined in equation. A block diagram of the design is shown in Figure 2[8].
Figure 2 shows the most commonly used row-column method for 1-D DCT, by which the 2-D 8 x 8 DCT can berealized with 16 passes of 1-D 8-point DCT. Besides, byapplying first-level even/odd decomposition, the 1-D 8-point DCT can be decomposed as follows. In 8 x 1 row DCT, each column of the input data X is computed and the outputs are stored in the transformation memory. Another 8 x 1 column DCT is performed to yield desired 64 DCT coefficients. The output bit-width is one of the parameter that determines the quality of image. The outputs of the row DCT are truncated to N bits before they are stored in transposition memory and sent to the input of the column DCT The 8 x 8 1-D DCT transform is expressed as

$$z_k = \frac{c(k)}{2} \sum_{i=0}^{7} x_i cos\frac{(2i+1)k\pi}{16}, k = 0,1,2,\ldots,7.$$

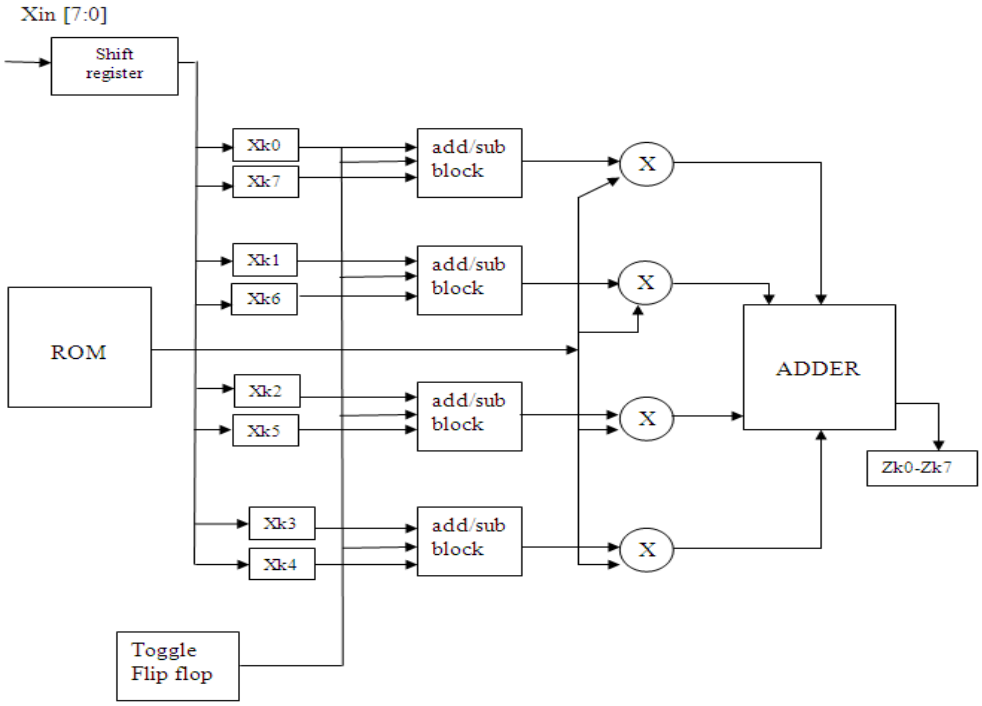

Fig. 2: 1D DCT Architecture

c(k)= $\frac{1}{2}, k = 0$
c(k)= 1,otherwise (8)

This equation is represented as vector-matrix form as

$$z = T.x^t \tag{9}$$

where T is an 8 x 8 matrix whose elements are cosine functions . xand z are the row and the column vectors, respectively. Since 8 x 8 coefficients matrix T in Equation(9) is symmetrical, the 1-D DCT matrix can be rearranged and expressed as,

$$\begin{bmatrix} z_1 \\ z_3 \\ z_5 \\ z_7 \end{bmatrix} = \begin{bmatrix} a & c & e & g \\ c & -g & -a & -e \\ e & -a & g & c \\ g & -e & c & -a \end{bmatrix} \begin{bmatrix} x_0 - x_7 \\ x_1 - x_6 \\ x_2 - x_5 \\ x_3 - x_4 \end{bmatrix}$$

$$\begin{bmatrix} z_0 \\ z_2 \\ z_4 \\ z_6 \end{bmatrix} = \begin{bmatrix} d & d & d & d \\ b & f & -f & -b \\ d & -d & -d & d \\ f & -b & b & -f \end{bmatrix} \begin{bmatrix} x_0 + x_7 \\ x_1 + x_6 \\ x_2 + x_5 \\ x_3 + x_4 \end{bmatrix} \tag{10}$$

Where ck = cos kπ/16, , a = c1, b = c2, c = c3, d = c4, e = c5, f = c6, g = c7.

Odd and even DCT can be easily changed to

$$\begin{bmatrix} z_1 \\ z_3 \\ z_5 \\ z_7 \end{bmatrix} = (x_0 - x_7) \begin{bmatrix} a \\ c \\ e \\ g \end{bmatrix} + (x_1 - x_6) \begin{bmatrix} c \\ -g \\ -a \\ -e \end{bmatrix} + (x_2 - x_5) \begin{bmatrix} e \\ -a \\ g \\ c \end{bmatrix} + (x_3 - x_4) \begin{bmatrix} g \\ -e \\ -c \\ -a \end{bmatrix}$$

$$\begin{bmatrix} z_0 \\ z_2 \\ z_4 \\ z_6 \end{bmatrix} = (x_0 + x_7) \begin{bmatrix} d \\ b \\ b \\ f \end{bmatrix} + (x_1 + x_6) \begin{bmatrix} d \\ -f \\ -d \\ -b \end{bmatrix} + (x_2 + x_5) \begin{bmatrix} d \\ -f \\ d \\ b \end{bmatrix} + (x_3 + x_4) \begin{bmatrix} d \\ -b \\ -d \\ -f \end{bmatrix} \tag{11}$$

8 x 8 DCT matrix multiplication can be expressed as additions of vector scaling operations, which allows us to apply our proposed low complexity design technique to DCT implementation[8].

A one dimensional DCT is implemented on the input pixels first. The output of this is intermediate value is stored in a RAM. The 2nd 1D-DCT operation is done on this stored value to give the final 2D-DCT output dct_2d. The inputs are 8 bits wide and the 2d-dct outputs are 9 bits wide. In 1D section the input signals are taken one pixel at a time in the order x00 to x07, x10 to x07 and so on up to x77. These inputs are fed into a 8 bit shift register. The outputs of the 8 bit shift registers are registered by the div8clk which is the CLK signal divided by 8. This will enable us to register in 8 pixels (one row) at a time. The pixels are paired up in an adder/ subtractor block in the order xk0,xk7:xk1,xk6:xk2,xk5:xk3,xk4. The adder/subtractor are tied to CLK. For every clk, the adder/subtractor module alternately chooses addtion and subtraction. This selection is done by the toggle flop. The output of the add/sub is fed into a multiplier whose other input is connected to stored values in registers which act as memory. The outputs of the 4 multipliers are added at every CLK in the final adder. The output of the adder Zk (0-7) is the 1D-DCT values given out in the order in which the inputs were read in.

B. Transposed buffer

The outputs of the adder are stored in RAMS. When WR is high, the corresponding RAM address takes the write operation. Otherwise, the contents of the RAM address are read. The period of the address signals is 64 times of the input clocks. Two RAMs are used so that data write can be continuous. The 1st valid input for the RAM1 is available at the 15th clk. RAM1 enable is active after 15 clks. After the write operation continues for 64 clks. At the 65th clock, since z_out is continuous, we get the next valid z_out_00. These 2nd sets of valid 1D-DCT coefficients are written into RAM2 which is enabled at 15+64 clks. So at 65th clk, RAM1 goes into read mode for the next 64 clks and RAM2 is in write mode. After this for every 64 clks, the read and write switches between the 2 RAMS.

RAM is enabled, data is written into the RAM1 for 64 clk cycles. After this Data is written in into each consecutive 64 locations are written. RAM1 goes into read mode and RAM2 goes into write mode. The cycle then repeats. For either RAM, data is written into each consecutive location. Data is read in a different order. Data is assumed to be written in each row at a time, in an 8x8 matrix, data is read each column at a time. When RAM1 is full, the 2^{nd} 1D calculations can start.

The second 1D implementation is the same as the 1st 1D implementation with the inputs now coming from either RAM1 or RAM2. Also, the inputs are read in one column at a time in the order z00 to z70, z10 to z70 upto z77. These inputs are fed into a 8 bit shift register. The outputs of the 8 bit shift registers are registered by the div8clk which is the CLK signal divided by 8. This will enable us to register in 8 pixels (one row) at a time. The pixels are paired up in an adder/ subtractor block in the order Zk0:Zk7, Zk1:Zk6, Zk2:Zk5, Zk3:Zk4. The adder/ subtractor is tied to CLK. For every clk, the adder/subtractor module alternately chooses addition and subtraction. This selection is done by the toggle flop. The output of the add/sub is fed into a multiplier whose other input is connected to stored values in registers which act as memory. The outputs of the 4 multipliers are added at every CLK in the final adder. The outputs from the adder in the 2nd section are the 2D-DCT coefficients values given out in the order in which the inputs were read in..When 2D DCT computation is over the rdy_out signal goes high indicating the output.

C. Two Dimensional DCT CORE

This section describes the architecture of the DCT. Below is I/O schematic of DCT core. This Core performs forward 2Dimension discrete cosine transform.

The DCT core is two dimensional discrete cosine transform implementation designed for use in compression systems like JPEG. Architecture is based on parallel distributed arithmetic with butterfly computation. Done as row/column decomposition where two 1D DCT units are connected through transposition matrix memory. Core has simple input interface. DCT core takes 8 bit input data and produces 12 bit output using 12 bit DCT matrix coefficients. The 8 bit signed input pixels will provide a 12 bit signed output coefficient for the DCT. The data bits are multiplied by 3 bits can result 11 bits and the sign bit to give a total of 12 bits. Vector processing using parallel multipliers is a method used for implementation of DCT. The advantages in vector processing method are regular structure, simple control and interconnect and good balance between performance and complexity of implementation.

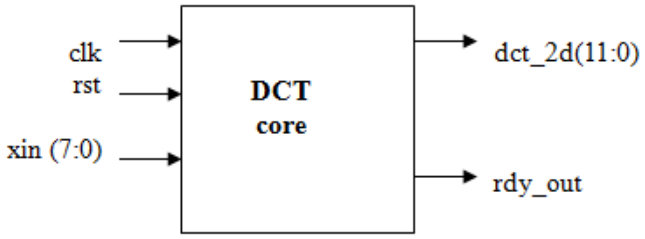

Fig. 3: Top level schematic for DCT core

IV. Inverse Discrete Cosine Transform

The figure.4 shows the 1D-IDCT is implemented on the input DCT values. The output of this called the intermediate value is stored in a RAM. The 2nd 1D-IDCT operation is done on this stored value to give the final 2D-IDCT output idct_2d. The inputs are 12 bits wide and the 2d-idct outputs are 8 bits wide. 1st 1D section, the input signals are taken one pixel at a time in the order x00 to x07, x10 to x07 and so on up to x77. These inputs are fed into an 8 bit shift register. The outputs of the 8 bit shift registers are registered at every 8th clock .This will enable us to register in 8 pixels (one row) at a time. The pixels are fed into a multiplier whose other input is connected to stored values in registers which act as memory. The outputs of the 8 multipliers are added at

every CLK in the final adder. The output of the adder z_out is the 1D-IDCT values given out in the order in which the inputs were read in. It takes 8 clks to read in the first set of inputs, 1 clk to get the absolute value of the input, 1 clk for multiplication, 2 clk for the final adder and total of 12 clks to get the 1st z_out value. Every subsequent clk gives out the next z_out value. So to get all the 64 values we need (12+64)=76 clks.

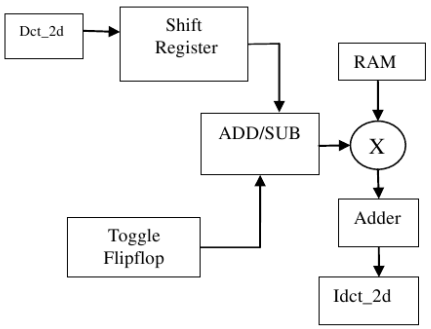

Fig. 4: 1-D IDCT Architecture

In this paper low power design, employ parallel processing units that enable power savings from a reduction in clock speed. This architecture save power by disabling units that are not in use. When the units are in a standby mode, they consume a minimum amount of power. The common low power design technique is to reorder the input data so that a minimum number of transitions occur on the input data lines. DCT architecture based on the proposed low complexity vector scalar design technique is effectively used to remove redundant computations, thus reducing computational complexity in DCT operations. This DCT architecture shows lower power consumption, which is mainly due to the efficient computation sharing and the advantage of using carry save adders as final adders.

V. Matlab Implementation

Each pixel value in the 2-D matrix is quantized using 8 bits which produces a value in the range of 0 to 255 for the intensity/luminance values and the range of -128 to + 127 for the chrominance values. All values are shifted to the range of -128 to + 127[10]] before computing DCT . All 64 values in the input matrix contribute to each entry in the transformed matrix . The value in the location F[0,0] of the transformed matrix is called the DC coefficient and is the average of all 64 values in the matrix[3]. The other 63 values are called the AC coefficients and have a frequency coefficient associated with them. Spatial frequency coefficients increase as we move from left to right (horizontally) or from top to bottom (vertically). Low spatial frequencies are clustered in the left top corner[11]. The Discrete Cosine Transform (DCT) separates the frequencies contained in an image. The original data could be reconstructed by Inverse DCT. Quantization is achieved by dividing each element in the transformed image matrix D by corresponding element in the quantization matrix, and then rounding to the nearest integer value. For the following step, quantization matrix Q50 is used.

$$C_{i,j} = round(D_{i,j}/Q_{i,j}) \tag{12}$$

Recall that the coefficients situated near the upper-left corner correspond to the lower frequencies to which the human eye is most sensitive of the image block. In addition, the zeros represent the less important, higher frequencies that have been discarded, giving rise to the lossy part of compression. As mentioned earlier, only the remaining nonzero coefficients will be used to reconstruct the image. The IDCT is next applied to matrix, which is rounded to the nearest integer. Finally, 128 is added to each element of that result, giving us the decompressed JPEG version N of our original 8x8 image block M. The output of IDCT and the input image matrices, when compared there is a remarkable result, considering that nearly 70% of the DCT coefficients were discarded prior to image block decompression/reconstruction. Given that similar results will occur with the rest of the blocks that constitute the entire image, it should be no surprise that the JPEG image will be scarcely distinguishable from the original. Remember, there are 256 possible shades of gray in a black-and-white picture, and a difference of, say, 10, is barely noticeable to the human eye. DCT takes advantage of redundancies in the data by grouping pixels with similar frequencies together. And moreover if we observe as the resolution of the image is very high, even after sufficient compression and decompression there is very less change in the original and decompressed image. Thus, we can also conclude that at the same compression ratio the difference between original and decompressed image goes on decreasing as there is increase in image resolution[9].

Simulation steps were done exactly like the steps of the previous one. MATLAB was used for generating the test vectors, and analysis of the output. The core was used to obtain IDCT coefficients and compared with MATLAB results. As 8x8 blocks is taken and processed, the whole image is done using block processing in Matlab.

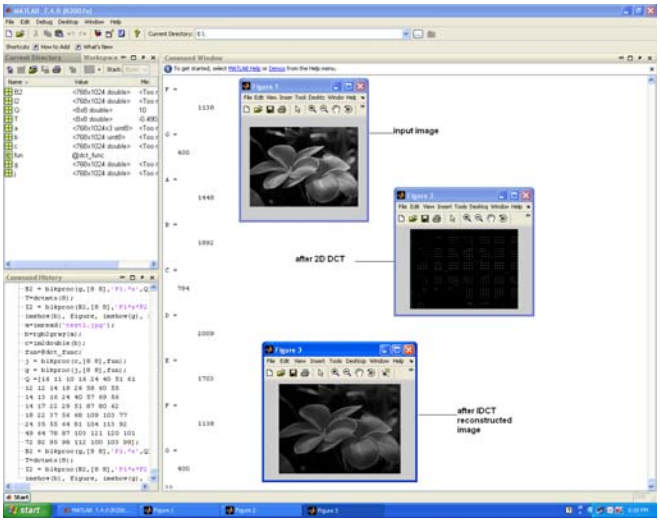

Fig. 5: Output of the Image From Matlab

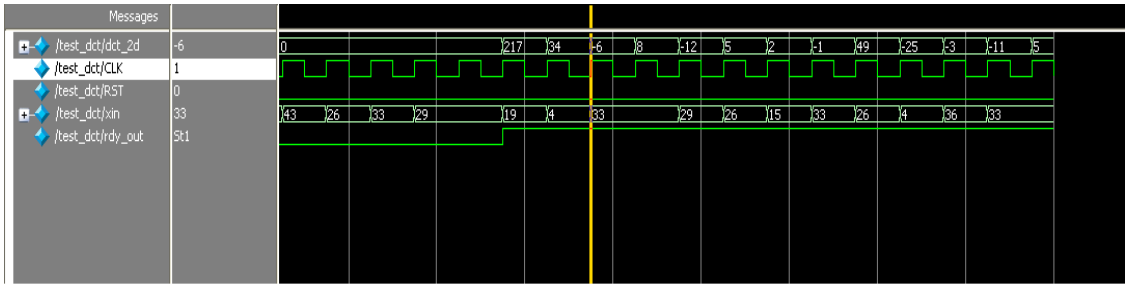

Fig. 6: DCT HDL simulation results

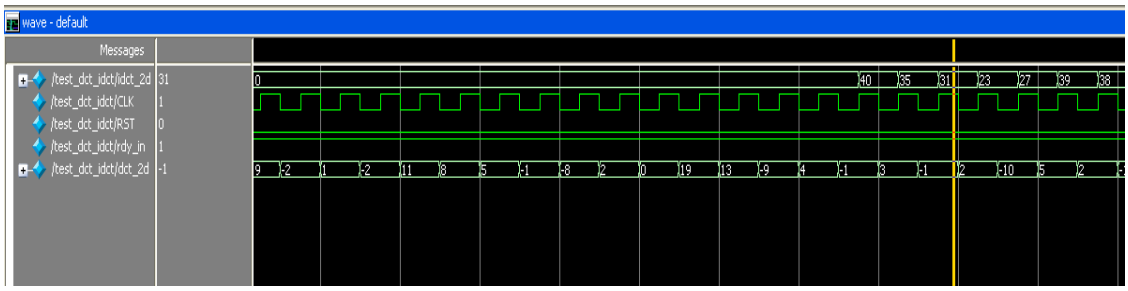

Fig. 7: IDCT HDL simulation results

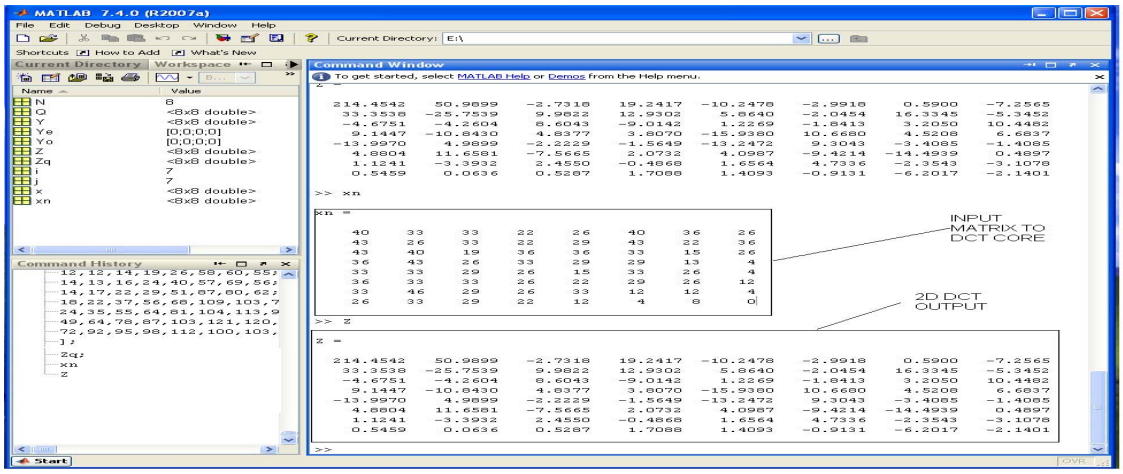

Fig. 8: 2D DCT Matlab output

VI. Interpretation of Results

DCT and IDCT both the blocks were written in the form of synthesizable Verilog code. Verilog test-benches were written and Cadence IUS simulator was used for behavioral simulation. MATLAB 5.2 from Math Works was used to generate input image matrix for to the input of the core. MATLAB reads both input and output calculates DCT/IDCT coefficients of the input file and compares them to the output of the core. After behavioral verification was done[4], RTL Complier was used to synthesize the Verilog code. Cadence RTL compiler was used to synthesize and generate schematic for both chips.

After synthesis, power analysis was performed.The core was targeted on ASIC technologies. The core was mapped to synthesize for using the power optimization. The timing simulation showed that the chip could operate at a maximum speed of 100MHz[6]. Simulation using CADENCE IUS simulator for DCT and IDCT core are given below in fig 3 and 4. The simulation results for 2D-DCT are compared with MATLAB results. Figure 5 shows the 2D DCT output obtained from MATLAB.

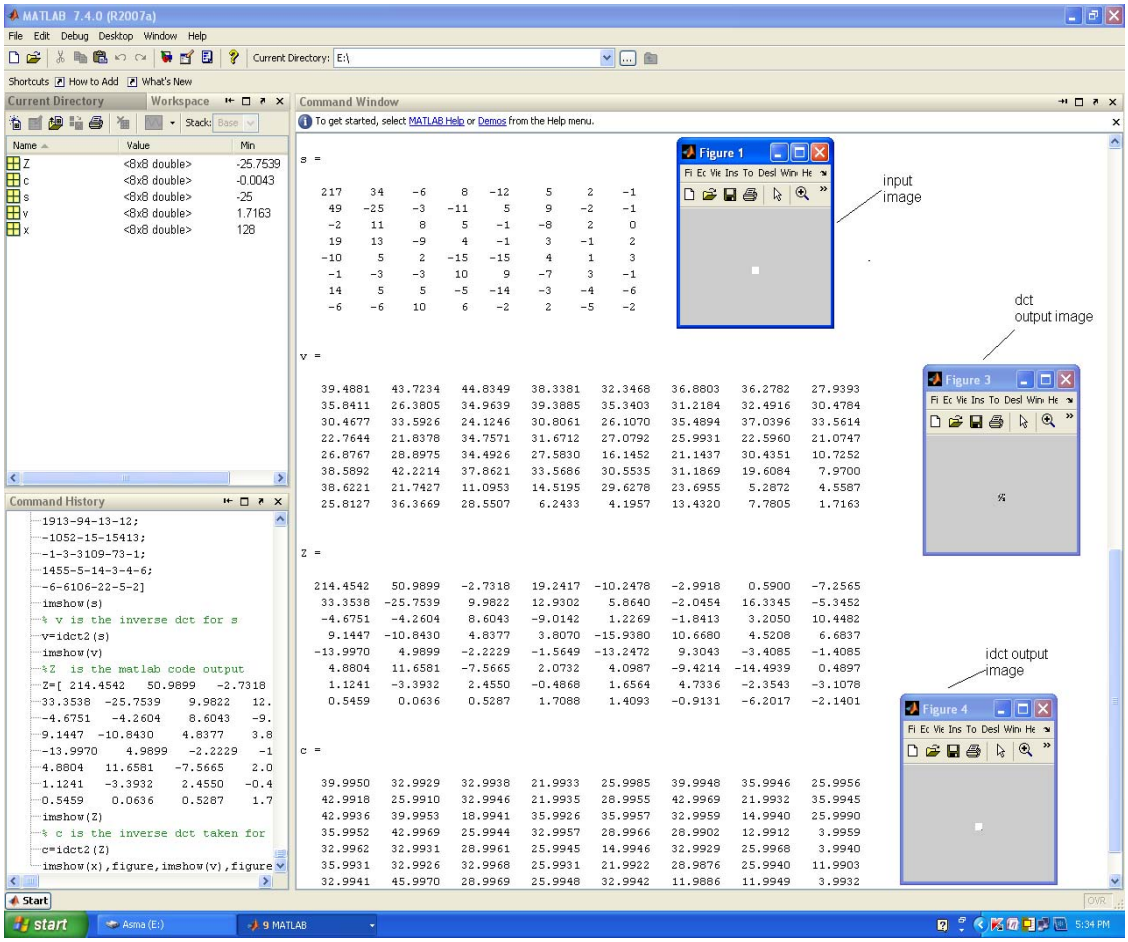

Fig. 9: 2D IDCT Matlab output

A. *Power analysis*

CADENCE tool is used to Simulate and Synthesize the Verilog HDL code written. After synthesis, power and area analysis is done by mapping on to 180 nm TSMC library. Schematic for DCT and IDCT chip is also generated using RTL compiler. The characteristics of DCT and IDCT is tabulated below in Table 1.

TABLE I: Characteristic of DCT and IDCT

Features	DCT	IDCT
Power	1.0278mW	1.268mW
Area	06281 mm^2	0.7652 mm^2
Latency	92 cycles	85 cycles
Block size	8x8	8x8
No. of Cells	6773	8571

The power when compared to DCT core presented in [1] is given in Table 2.

TABLE II: Performance Comparision

DCT Architecture	Power in mW	Area in mm^2
Proposed DCT core	1.0278	0.6281
DCT Architecture[1]	29.78	0.343
DCT Core[12]	29.92	0.569

VII. Conclusion

An ASIC implementation of 2D DCT core was designed to meet low power constraints. DCT and IDCT algorithm was written using Verilog HDL, then simulated and synthesized successfully. Simulation is done by IUS simulator using nclaunch from cadence. The image coefficients are obtained by using MATLAB and are applied as inputs to DCT core. The compressed image is thus reconstructed by using MATLAB. The compressed image thus obtained is compared with the simulation results from MATLAB and CADENCE IUS simulator. Power analysis was done using RTL compiler with 180 nm TSMC library and low power is achieved.

REFERENCES

[1] Jongsun Park Kaushik Roy *A Low Complexity Reconfigurable DCT Architecture to Trade off Image Quality for Power Consumption* Received: 2 April 2007 / Revised: 16 January 2008 / Accepted: 30 April 2008 / Published online: 3 June 2008.

[2] S. Ramachandran And S. Srinivasan Department of Electrical Engineering *A Novel, Automatic Quality Control Scheme for Real*

[3] Sungwook Yu and Earl E. Swartzlander Jr., Fellow, IEEE, *DCT Implementation with Distributed Arithmetic*, IEEE Transactions on Computers, VOL. 50, NO. 9, September 2001.

[4] B. Heyne and J. Goetz *A low-power and high-quality implementation of the discrete cosine transformation*,Adv. Radio Sci., 5, 305311, 2007.

[5] Y.P Lee, *A cost effective architecture for 8x8 two-dimensional DCT/IDCT using direct method IEEE Transactions on circuit and system for video technology* vol 7 N0 3 June 1997.

[6] C. Hemasundara Rao and M. Madavi Latha *A Novel VLSI Architecture of Hybrid Image Compression Model based on Reversible Blockade Transform*. .

[7] Sherif T.EID Shams University Cairo. *A Low Power 1-D DCT/IDCT Core*. 1999-Y2k.

[8] Ken Cabeen and Peter Gent Math 45 College of the Redwoods *Image Compression and Discrete Cosine Transform*.

[9] Andrew B.Watson,*Image compression using the discrete cosine transform*, Mathematical Journal, 4(1), 1994, p, 81-88.

[10] VijayaPrakash and K.S.Gurumurthy.*A Novel VLSI Architecture for Digital Image Compression UsingDiscrete Cosine Transform and Quantization* IJCSNS September 2010.

[11] Syed Ali Khayam, *The Discrete Cosine Transform (DCT): Theory and Application*. ECE 802 602: Information Theory and Coding, March 10th 2003.

[12] Xanthopoulos .T and Chandrakasan A P. *A Low Powr DCT core using Adaptive Bitwidth and arithmetic Activity exploiting signals correlations and Quantization.* IEEE journals of solid state circuits 35(5) 740-750 may 2000.

VijayaPrakashA.M He is graduatedB.E from UVCE Bangalore, of Bangalore University in the year 1992, Post graduated in M.E from SDMCET Dharwad of Karnataka University in the year 1997 and presently pursuing Ph.D. from Dr.M.G.R University Chennai. He has been actively guiding PG and UG students in the area of VLSI and Image Processing. He has published more than 4 National/international papers. Currently he has been working as Assistant Professor in Electronics and Communication Engineering Department, Bangalore Institute of Technology Bangalore. His research interests are Low Power VLSI, Image Processing, Synthesis and Optimization of Digital Circuits. He is a member of IMAPS and ISTE.

K.S Gurumurthy Obtained his B.E degree from M.C.E Hassan of Mysore University in the year 1973. He got his M.E Degree from University of Roorkee (now IIT-Roorkee) in 1982. He joined UVCE in 1982 and he has since been teaching Electronics related subjects. He obtained his Ph.D. degree in 1990 from IISc Bangalore. Presently he is a professor in the DOS in E&C, UVCE, BU, Bangalore-1 He is a University gold medal winner from University of Roorkee and a recipient of the Khosla award for the best technical paper published in 1982. He has successfully guided 4 Ph.D., 2 M.Sc-Engg (by research) candidates and guided a number of UG and PG projects. He has around 75 technical paper publications in journals and international conferences. He has presented papers in JAPAN, FRANCE, MALAYASIA and USA. His interests are Low power VLSI, Multi valued logic circuits, Deep submicron Devices. He is a member of IEEE and ISTE.