4th International Conference on Multidisciplinary Research
Osmania University Centre for International Program, Osmania University Campus, Hyderabad (India) (ICMR-2019)
2nd February 2019, www.conferenceworld.in
ISBN:978-93-87793-67-5

A RECONFIGURABLE ARCHITECTURE FOR EFFICIENT AND SCALABLE APPROXIMATION OF DCT

Hareesh T M,Ajay Joseph[1],Harikumar T[2]

*U.G Students, Department of Electronics and Communication Engineering, College of EngineeringAdoor, Kerala,*India[1]

Assistant Professor, Department of Electronics and Communication Engineering, College of Engineering Adoor,Kerala, India[2]

ABSTRACT

One of the most important steps in image compression is applying discrete cosine transform. A two dimensional DCT when done to a signal or image gives a much energy compaction. The 2D DCT removes all the redundant data in an image and concentrates the important data in the higher frequencies of the image making it easier for further compression process. Approximation of discrete cosine transform (DCT) is useful for reducing its computational complexity without significant impact on its coding performance. This paper implements a reconfigurable architecture to approximate a two dimensional DCT using only additions and is able to compute either 32 point ,parallel computation of 16 point or 8 point DCT, as per requirement. This implementation hold good with its lower arithmetic complexity, reduced delay and higher hardware efficiency.

Keywords: Discrete cosine transform (DCT)

I. INTRODUCTION

Due to the increasing requirements for transmission of images in computer and mobile environments, the research in the field of image compression has increased significantly. Image compression plays a crucial role in digital image processing, it is also very important for efficient transmission and storage of images. When we compute the number of bits per image resulting from typical sampling rates and quantization methods, we find that image compression is needed. Therefore development of efficient techniques for image compression has become necessary.One of the most important process in numerous applications in science and engineering, from lossy compression of audio and images to spectral methods for the numerical solution of partial differential equations is Discrete Cosine Transform (DCT).The discrete cosine transform (DCT) helps separate the image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality).Approximation of discrete cosine transform (DCT) is useful for reducing its computational complexity without significant impact on its coding performance. Various techniques and algorithms are developed for the fast,efficient and scalable approximation of DCT. The DCT can be used to convert the signal (spatial

information) into numeric data (frequency or spectral information) so that the image's information exists in a quantitative form that can be manipulated for compression. The DCT transforms a signal from a spatial representation into a frequency representation. In an image, most of the energy will be concentrated in the lower frequencies, so if we transform an image into its frequency components and throw away the higher frequency coefficients, we can reduce the amount of data needed to describe the image without sacrificing too much image quality.

II. BLOCK DIAGRAM

A 2D DCT is obtained by converting the image in to its corresponding matrix and the moving them into the reconfigurable logical structure for finding 1D DCT. The transpose of the output is then taken and it is again processed with 1D DCT giving us the 2D DCT which has more energy compaction. The Xilinx System Generator for DSP is a plug-in to Simulink that enables designers to develop high-performance DSP systems for Xilinx FPGAs. Designers can design and simulate a system using MATLAB, Simulink, and Xilinx library of bit/cycle-true models. The tool will then automatically generate synthesizable Hardware Description Language (HDL) code mapped to Xilinx pre-optimized algorithms. In multimedia applications, embedded watermarks should be invisible, robust, and have a high capacity.

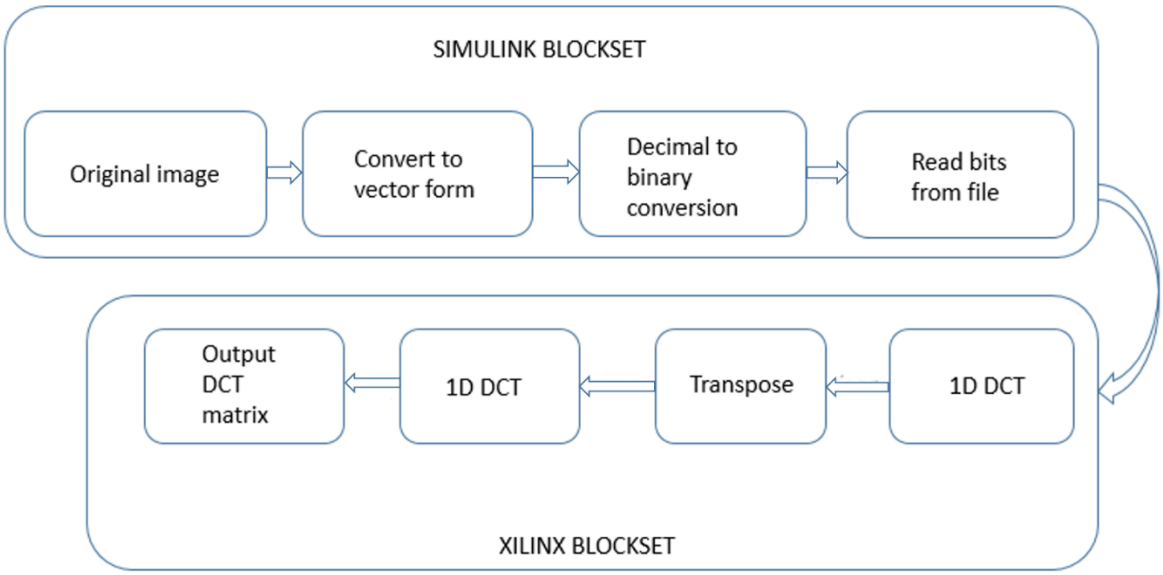

Fig 3.1 Block diagram of proposed architecture

SIMULINK BLOCK SET

In the VLSI implementation of applying DCT to an image which convert to vector form, then the entire decimal signal are convert to binary signals which means bit form. The group of bits stored in a file and using the Simulink block sets read an image in a bit by bit format.

XILINX BLOCK SET

A discrete cosine transform (DCT) expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies. In particular, a DCT is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. The DCTs are generally related to Fourier Series coefficients of a periodically and symmetrically extended sequence whereas DFTs are related to Fourier Series coefficients of a periodically extended sequence. DCTs are equivalent to DFTs 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), whereas in some variants the input and/or output data are shifted by half a sample. The discrete cosine transform (DCT) helps separate the image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality).

The general equation for a 1D (*N* data items) DCT is defined by the following equation:

$$F(u) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \Lambda(i).cos\left[\frac{\pi.u}{2.N}(2i+1)\right] f(i)$$

The general equation for a 2D (*N* by *M* image) DCT is defined by the following equation:

$$F(u,v) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \left(\frac{2}{M}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \Lambda(i).\Lambda(j).cos\left[\frac{\pi.u}{2.N}(2i+1)\right] cos\left[\frac{\pi.v}{2.M}(2j+1)\right] .f(i,j)$$

In general, a lot of multiplication operation have to be carried out in order to find DCT of an image or signal. Many fast algorithms have been developed in order to find DCT easily. The main objective of the approximation algorithms is to get rid of multiplications which consume most of the power and computation-time, and to obtain meaningful estimation of DCT as well.DCT as explained before is commonly used in data compression applications internationally due to good de-correlation properties. Its widespread use can be attributed to the energy compaction quality of the cosine transform. Transform coding is based on the premise that the pixels in an image exhibit certain levels of correlation with the neighboring pixels. Consequently, these correlations can be exploited to predict the value of a pixel from its respective neighbor. A transformation is, therefore, defined to map this correlated data into uncorrelated coefficients. The information content of an individual pixel is relatively small i.e. to a large extent visual contribution of a pixel can be predicted with the help of the neighbor. The DCT of data can be found directly given the data values, but if the data is more, say about 100 or 500 it is not possible to find the DCT/IDCT on the whole. Hence we divide the given data elements into sets of 8 which is most commonly done. The normal implementations of 1D-DCT is using multipliers and adder-subtractor in the design as given in. The 1D-DCT of a data set of 8 elements can be found out using the matrix shown below,

$$\begin{bmatrix} y0 \\ y1 \\ y2 \\ y3 \\ y4 \\ y5 \\ y6 \\ y7 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} c4 & c4 & c4 & c4 & c4 & c4 & c4 & c4 \\ c1 & c3 & c5 & c7 & -c7 & -c5 & -c3 & -c1 \\ c2 & c4 & c6 & -c2 & -c2 & -c6 & c6 & c2 \\ c3 & -c7 & -c1 & -c5 & -c5 & c1 & c7 & -c3 \\ c5 & -c1 & c7 & c3 & -c3 & -c7 & c1 & -c5 \\ c6 & -c2 & c2 & -c6 & -c6 & c2 & -c2 & c6 \\ c7 & -c5 & c3 & -c1 & c1 & -c3 & c5 & -c7 \end{bmatrix} \begin{bmatrix} x0 \\ x1 \\ x2 \\ x3 \\ x4 \\ x5 \\ x6 \\ x7 \end{bmatrix}$$

Where x(0) to x(7) represent the 8 data elements and Cx= Cos(πx/16). In this implementation it needs the use of 64 multipliers for multiplying the data values with the cosine values, 56 adders for adding the result after multiplication, and shifters that perform arithmetic right shifts to divide by 2. The major concern is the number of adders and multipliers that are used and the power consumed by the design. This matrix can be further simplified as in, which focuses more on reducing the number of adders and multipliers to decrease the net power consumed by the design and to decrease the number of gates in the design. The figure below shows the butterfly representation of the 1D DCT/IDCT. x0 to x7 represents the input data. The input data is first added or subtracted and then multiplied with the cosine values. The result after multiplying the data with the cosine values is added between them and then divided by two which gives the final output y0 to y7 of 1D DCT/IDCT.

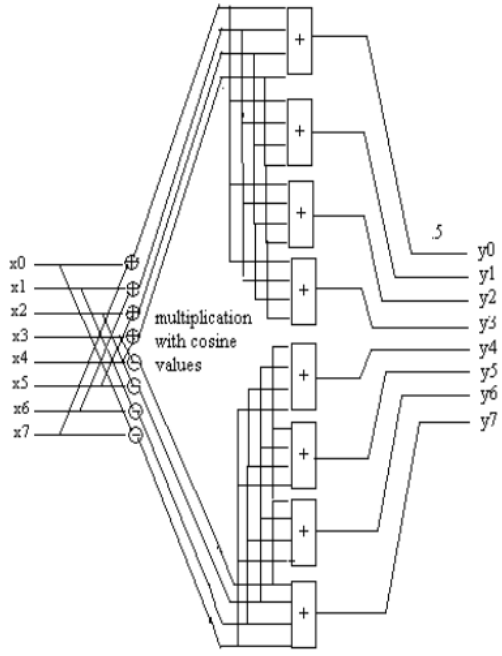

Fig. 3.2 Butterfly diagram for 1D DCT

For 2D DCT the output obtained after performing 1D DCT i.e. y0 to y7 is fed to a Transpose Memory. The Transpose memory can hold 64 bits or 8bytes of data. The data written into the transpose memory is set as the input of another 1D DCT block. The second input will be the seventh bit of all the data in the memory and so on.

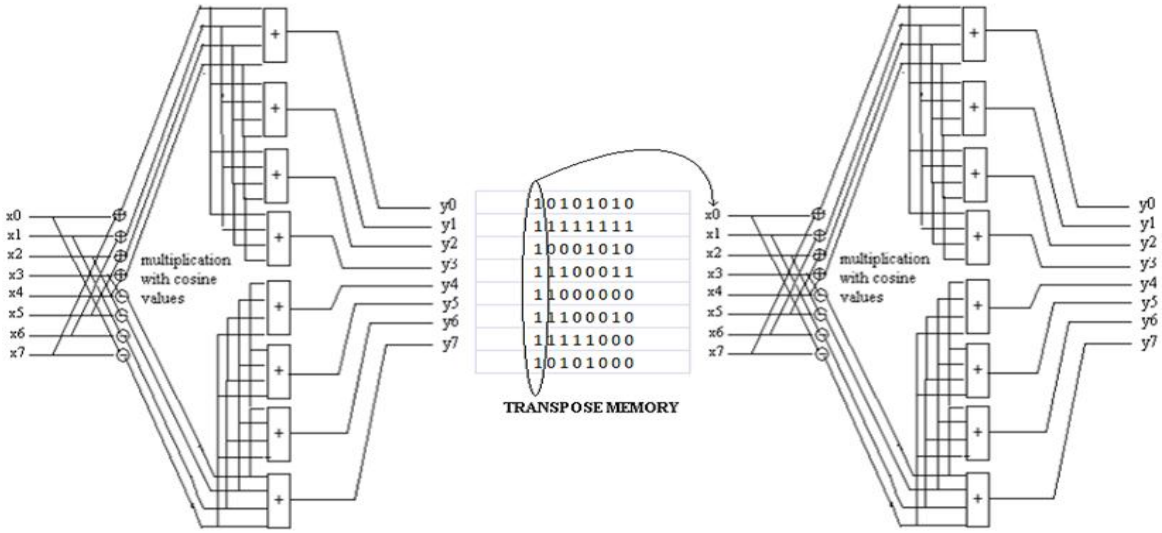

Fig. 3.3 Butterfly diagram for 2D DCT

The reconfigurable structure used in this paper is scalable, orthogonal and is efficient in carrying out DCT of an image which can be a 32point, two parallel 16point or 4 parallel 8 point DCTs and does not use any multiplication or shifting operations making them much simple and easier to implement.

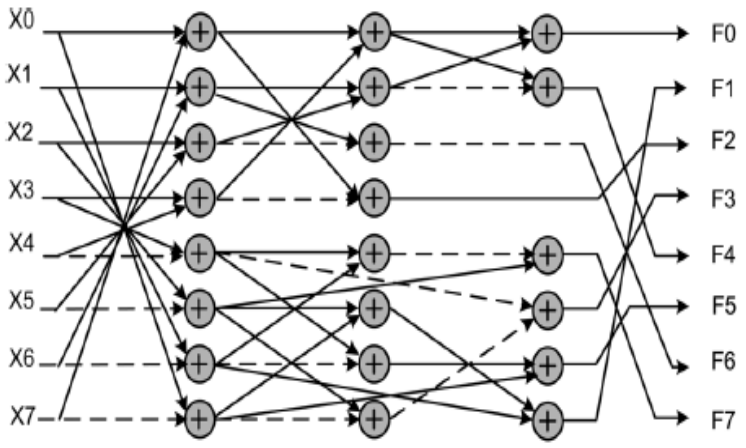

Fig 3.4 Signal Flow graph of 8 point DCT [2]

The structure for 8 point DCT contains 22 adders. The structure is simple and fast enough to find 8 point dct without any multiplication or shifting operation that costs much space and computational complexity.

The structure for 8 point DCT was further modified and implemented with 14 adders which reduced the complexity and increased the speed and efficiency in computation to a large extent.

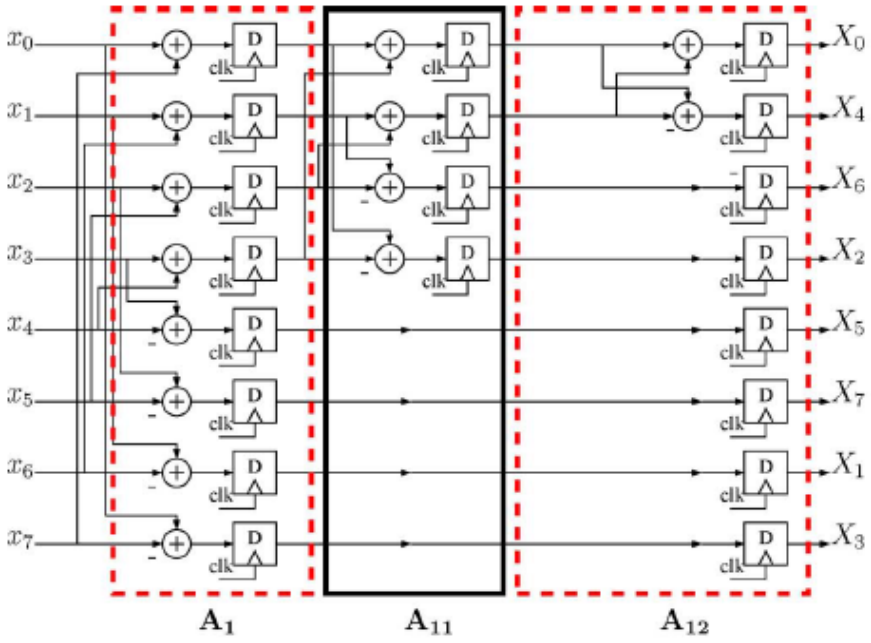

Fig 3.5 Modified structure for 8 point DCT approximation

The 16 point DCT can be obtained from two 8 point DCT structure. The inputs are given to an input adder unit, and then passed to the 8 point DCT structure. The output is then given to an output permutation structure that provide output based on the select line which determines whether the DCT should be done is 8 point DCT or 16 point DCT.The reconfigurable structure is able to compute 16 point DCT or two parallel 8 point DCTs. When the select line sel16 = 1, the structure computes 16 point DCT and when the select line sel16=0 the structure is processed accordingly to compute two parallel 8 point DCT.

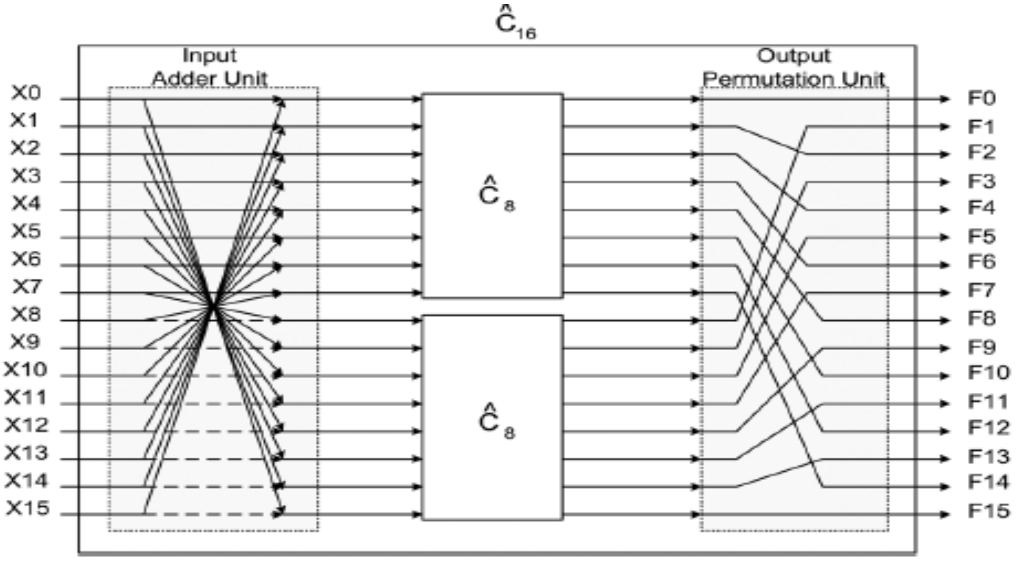

Fig 3.6 Block diagram for 16 point DCT [2]

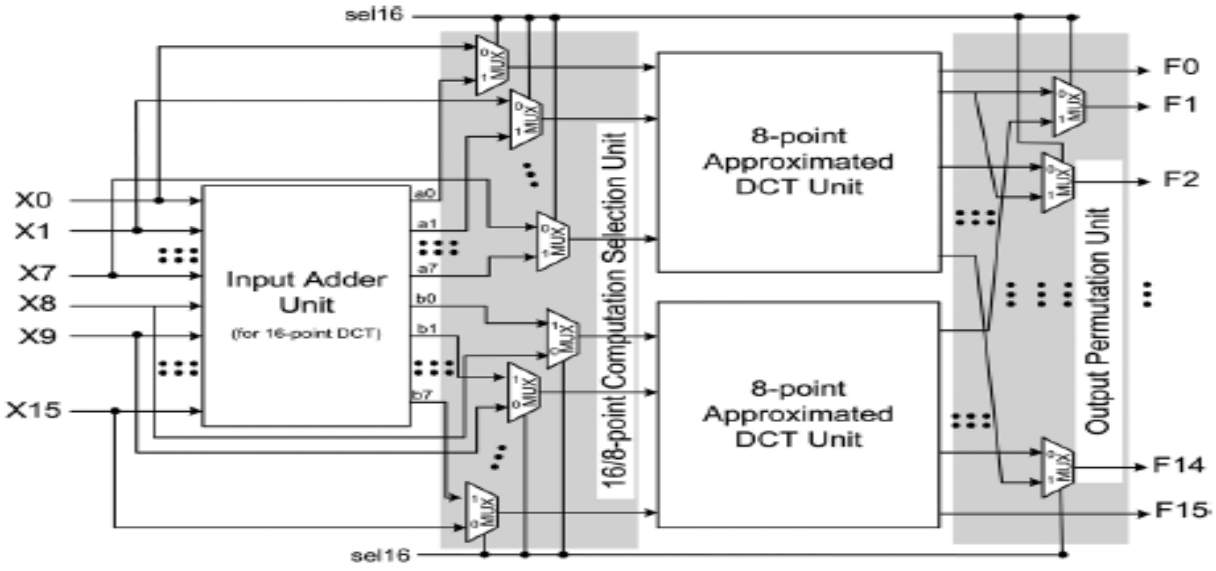

Fig. 3.7Reconfigurable architecture for approximate DCT of lengths N=8, 16 [2]

The 32 point DCT structure is derived from the 16 point DCTstructure. The reconfigurable structure contains a 32 point input adder unit followed by two 16 point input adder units and then four 8 point DCT units.

Depending on which N point DCT is required, the select line is provided correspondingly. The structure is configured so as to compute 32 point DCT when sel32=1,sel16=1 ,two parallel 16 point DCT when sel32=0,sel16=1 and four parallel 8 point DCT when sel32=0,sel16=0. The output permutation block selects output after determining the type of the DCT to be computed and provide the output selecting the correct permutation from computed outputs.

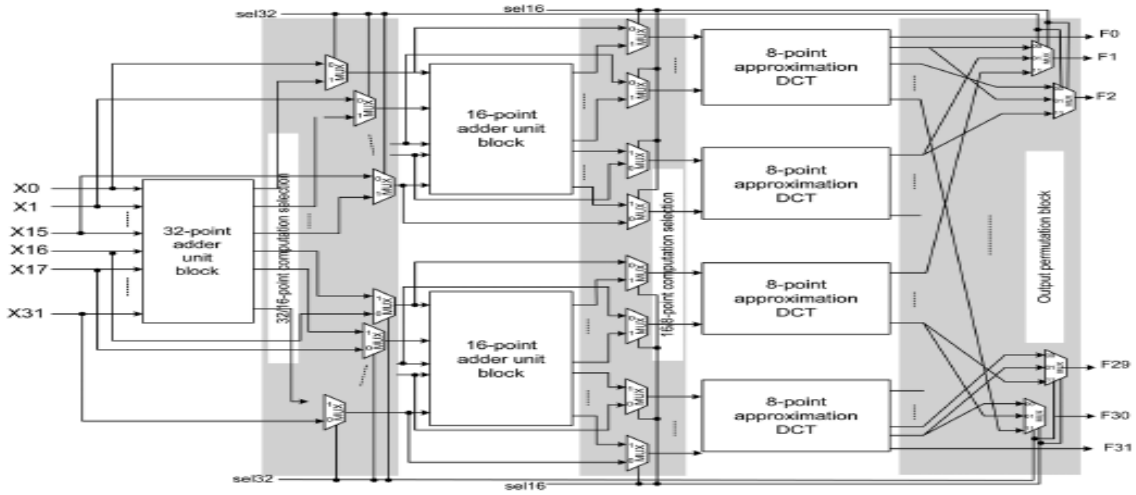

Fig 3.8Reconfigurable architecture for approximate DCT of lengths N=8, 16 and 32 [2]

The structure computes and provides a one dimensional DCT or 1D DCT. 1D DCT gives us energy compaction but performing a two dimensional DCT is much more efficient and is helpful in image compression. A 2D DCT

can be performed by taking the transpose of the output 1D DCT matrix and computing 1D DCT again.The transpose of the matrix can be calculated in as follows.The structure of the proposed 4 x 4 transposition buffer is shown in Fig. It consists of 16 registers arranged in 4 rows and 4 columns. (N x N) transposition buffer can store N values in any one column of registers by enabling them by one of the enable signals ENi for i = 0, 1,…..N -1. One can select the content of one of the rows of registers through the MUXs. During the first N successive cycles, the DCT module receives the successive columns of (N x N) block of input for the computation of STAGE-1, and stores the intermediate results in the registers of successive columns in the transposition buffer. In the next N cycles, contents of successive rows of the transposition buffer are selected by the MUXs and fed as input to the 1-D DCT module. N MUXs are used at the input of the 1-D DCT module to select either the columns

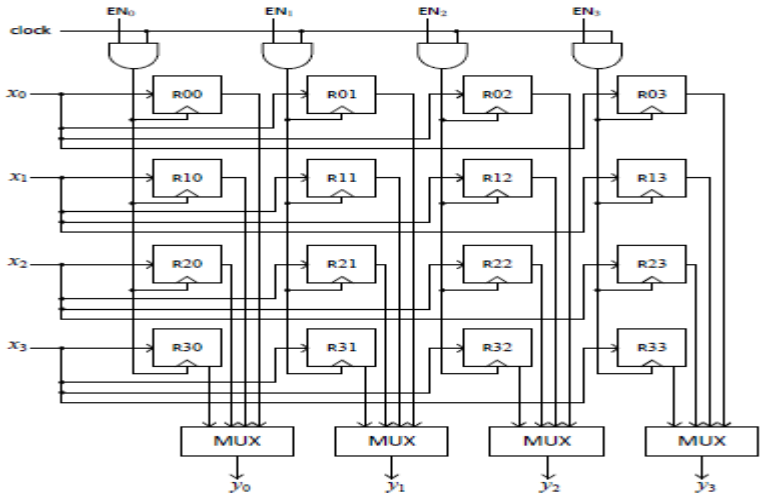

Fig 3.8Structure of the transposition buffer for input size 4 x 4

By combining four transposition buffers of input size 4x4 we can develop structure of transposition buffer for input size 8x8 . By combining four transposition buffers of input size 8x8 we can develop structure of transposition buffer for input size 16x16 . By combining four transposition buffers of input size 16x16 we can develop structure of transposition buffer for input size 32x32 .Input select lines, enables and flags are declared and provided so that the outputs for each of the section is selected correspondingly.

IMPLEMENTATION

The architecture is coded for 32 point, 16 point and 8 point DCT and implemented in Verilog. The image for which the DCT is to be computed is converted in to its corresponding hex values using MATLAB and is then given to the coded architecture. The input of the Verilog design will be a 32x32 matrix containing 1024 values. Enables and clock pulses for the flip flops are given as inputs too. The structure will receive input and compute 32 point or two parallel 16 point or four parallel 8 point DCTs according to the select lines. The output matrix is the computed one dimensional DCT matrix and is then taken transpose using the transpose architecture that was coded. The output matrix after transposing is again fed in to the DCT architecture to compute the two

dimensional DCT matrix. The two dimensional DCT is much more efficient and energy compact and is very useful in image transformation.

Codes are written for 8 point DCT architecture using 22 adders as well as 14 adders and the entire structure for computing 2D DCT with 32,16 and 8 point are implemented separately. Using a sample image, simulations are run and the logical utilization for both the implementations are evaluated.

The device utilization summary obtained is as follows:

Logic utilization	Existing system	Implemented system
No. of slices	20920	14796
No. of slice FFs	3478	17542
Total no. of 4 input LUTs	35664	19432
No. of bonded IOBs	646	1066
Delay (ns)	4.496	4.283

CONCLUSION

Codes for computing two dimensional DCT was written in Verilog. The codes were simulated and tested. The output DCT matrix obtained showed much energy compaction and is assumed to provide much advantage in post processes in image compression. The modified structure obtained an additional hardware efficiency of 12.93% and reduction in delay by 4.73% than the existing system.

REFERENCES

[1] F. M. Bayer and R. J. Cintra, "DCT-like transform for image compression requires 14 additions only," *Electron.Lett.*, vol. 48, no. 15, pp. 919–921, Jul. 2012

[2] Maher Jridi, Ayman Alfalou and Pramod KumarMeher, "A Generalized Algorithm and Reconfigurable Architecture for Efficient and Scalable Orthogonal Approximation of DCT", *IEEE Trans. Circuits Syst. I Reg.Papers*, vol. 62, no.2 Feb 2015

[3] R. J. Cintra and F. M. Bayer, "A DCT approximation for image compression," *IEEE Signal Process. Lett.*, vol. 18, no. 10, pp. 579–582, Oct. 2011

[4] S. Bouguezel, M. Ahmad, and M. N. S. Swamy, "Low-complexity 8x8 transform for image compression," *Electron. Lett.*, vol. 44, no. 21, pp. 1249–1250, Oct. 2008.

[5] U. S. Potluri, A.Madanayake, R. J. Cintra, F.M. Bayer, S. Kulasekera, and A. Edirisuriya, "Improved 8-point approximate DCT for image and video compression requiring only 14 additions," *IEEE Trans. CircuitsSyst. I, Reg. Papers*, vol. 61, no. 6, pp. 1727–1740, Jun. 2014.

[6] F. M. Bayer, R. J. Cintra, A. Edirisuriya, and A. Madanayake, "A digita hardware fast algorithm and FPGA-based prototype for a novel 16-point approximate DCT for image compression applications," *Meas. Sci. Technol.*, vol. 23, no. 11, pp. 1–10, 2012.

[7] Z. Mohd-Yusof, I. Suleiman, and Z. Aspar, "Implementation of two dimensional forward DCT and inverse DCT using FPGA," in *Proc.TENCON 2000*, vol. 3, pp. 242–245.

[8] Thomas and Moorby : The Verilog Hardware Description Language,Springer US,5th edition,2002