Design and Implementation of Image Compression Encoder using Orthogonal Approximation DCT

Vanitha R, K S Gurumurthy

Department of ECE, REVA Institute of Technology, Bengaluru-64, Karnataka, India

Abstract— *Image Compression is usually carried out using discrete cosine transform (DCT) because compressed image using DCT will take less memory to store the image and quality of the image will be good compared JPEG and HEVC. But, in this work an attempt is made to achieve compression using Approximation DCT (ADCT). ADCT is useful for reducing its computational complexity without affecting its coding performance. It provides better image and video compression compared to the DCT. ADCT is orthogonal and it has lower structural complexity compared to DCT. The unique feature of the ADCT is that it could be configured for the computation of the 32 point ADCT or for parallel computation of two16 point ADCTs or four 8 points ADCTs. It has many advantages in terms of orthogonality, structural simplicity and lower computational complexity. The proposed ADCT is implemented using Verilog and Simulated by ModelSim and synthesized by Xilinx ISE 9.1i. Results are compared with 16 point ADCT with 16 point DCT implementation. The target device is XC5vtx330t-2ff1738. The 16 point ADCT implementation results in a saving of 28.37% IOBs and 63% of LUTs, compared to existing 16 point DCT implementation.*

Keywords— approximation discrete cosine transform (ADCT), compression, discrete cosine transform DCT) , high efficiency video coding (HEVC, structural complexity.

I. INTRODUCTION

Image or video compression reduces the size of image without losing its quality. The reconstructed image should be same as the original image. Compacting is important for storage and transmission of images. JPEG (Joint Photographic Expert Group) is most widely used compression method. Compare to JPEG and high efficiency video coding (HEVC) ADCT is better because after image crimping it will reduce the bandwidth and consumes less memory to store the image and also good quality of image will be obtained.

Discrete cosine transform (DCT)

DCT transforms an image or signal from spatial domain to frequency domain. It removes high frequency components. It express a sum of finite sequence of a data points in terms of a sum of cosine functions oscillating at different frequencies. If we use cosine functions instead of sine functions the compression will become critical.

The one Dimensional DCT can be expressed as:

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

Where

$$\Lambda(i) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for} \xi = 0 \\ 1 & \text{otherwise} \end{cases}$$

The inverse one Dimensional DCT is $F^{-1}(u)$.The two Dimensional DCT is given by:

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

Where

$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for} \xi = 0 \\ 1 & \text{otherwise} \end{cases}$$

DCT is required to enhance memory or storage capacity. It will utilize the channel bandwidth efficiently. It is used in image compression and video compression. It provides higher compression ratio for video and images. It is widely used for Image compression. In DCT we need to use IDCT to reconstruct the original image. In Data compression there are two types. One technique achieves compression without the loss of any information. . It is usually used where a loss of information is a major damage. The other technique will incur little amount of loss of information. It is used for video and audio where lost information will not have any effect on the reconstructed image. It goes unnoticed by most users. DCT is non-orthogonal. It is scalable and reconfigurable. Its structure is complex. It has high computational cost.

Apporximate discrete cosine transform (ADCT)

It is a lossless compression technique. ADCT can be upgraded to accommodate more number of points. The coefficients of N – point ADCT are given by:

$$c(i,j) = \epsilon_i \sqrt{\frac{2}{N}} \cos \frac{(2j+1)i\pi}{2N} \qquad (3)$$

With $0 \leq$ i , j $\leq$ N – 1, $\epsilon_{0 = 1/}\sqrt{2}$ and ϵ_i = 1 , for i > 0.
For K € [0, (N / 2) - 1] and i = 2k, for any even value of N we can find that ,

$$c(2k,j) = \epsilon_{2k} \sqrt{\frac{2}{N}} \cos \frac{(2j+1)2k\pi}{2N} \qquad (4)$$

Since $\epsilon_{2k} = \epsilon_k$, equation 6 can be rewritten as,

$$c(2k,j) = \epsilon_k \sqrt{\frac{2}{N}} \cos (2j+1)k\pi N \qquad (5)$$

ADCT matrix C_N is given by:

$$\boldsymbol{C}_N = \frac{1}{\sqrt{2}} \boldsymbol{M}_N^{per} \boldsymbol{T}_N \boldsymbol{M}_N^{add} \qquad (6)$$

T_N is given by:

$$\boldsymbol{T}_N = \begin{bmatrix} \boldsymbol{C}_{\frac{N}{2}} & \boldsymbol{0}_{\frac{N}{2}} \\ \boldsymbol{0}_{\frac{N}{2}} & \boldsymbol{S}_{\frac{N}{2}} \end{bmatrix} \qquad (7)$$

$\boldsymbol{M}_N^{per}$ permutation matrix is represented by:

$$\boldsymbol{M}_N^{per} = \begin{bmatrix} \boldsymbol{P}_{N-1,\frac{N}{2}} & \boldsymbol{0}_{1,\frac{N}{2}} \\ \boldsymbol{0}_{1,\frac{N}{2}} & \boldsymbol{P}_{N-1,\frac{N}{2}} \end{bmatrix} \qquad (8)$$

Where $0_{1,N/2}$ is a row of N / 2 zeros and $P_{N-1,N/2}$ is a (N – 1) X (N / 2) matrix defined by its row vectors as

$$\boldsymbol{P}_{N-1,\frac{N}{2}}^{(i)} = \begin{cases} \boldsymbol{0}_{1,\frac{N}{2}} & if \quad i = 1,3,5,\ldots,N-1 \\ \boldsymbol{I}_{\frac{N}{2}}\left(\frac{i}{2}\right) & if\, i = 0,2,4,\ldots,N-2 \end{cases} \qquad (9)$$

Where $I_{N/2}$(i / 2) is the (i / 2)th row vector of the ((N / 2) X (N / 2)) identiy matrix.
Since $\boldsymbol{M}_N^{add}$ is given by

$$\boldsymbol{M}_N^{add} = \begin{bmatrix} \boldsymbol{I}_{\frac{N}{2}} & \boldsymbol{J}_{\frac{N}{2}} \\ \boldsymbol{I}_{\frac{N}{2}} & -\boldsymbol{J}_{\frac{N}{2}} \end{bmatrix} \qquad (10)$$

Where $J_{N/2}$ is an ((N / 2) X (N / 2)) matrix having all 1s as anti-diagonal elements and 0s elsewhere. $\boldsymbol{M}_N^{per}$ contributes very little to the total artthmetic compexity. Approximating T_N given in equation (8), the overhead on the processor for N point ADCT is reduced. The $\hat{\boldsymbol{C}}_{N2}$ and $\hat{\boldsymbol{S}}_{N2}$ denote the approximation of $\hat{\boldsymbol{C}}_{N2}$ and $\hat{\boldsymbol{S}}_{N2}$ respectively. The smallest size of ADCT matrix is taken to find approximated submatrices. A good approximation of C_N for N > 8 leads to a proper approximation of C_8 and S_8.
It is orthogonal. It has structural simplicity as it allows only two mathematical operations namely addition and subtraction. Therefore it has lower computational complexity. It is reconfigurable. The objective of the paper is to implement image compression using 8 point /16point /32 point ADCT technique. Initially it is proposed to verify the technique using MATLAB and later on it is proposed to implement the same on Xilinx FPGA. The objective of the project is also to compare ADCT implementation with that of DCT implementation from the device utilization view point.
The rest of the paper is organised as:
Unit II gives literature survey. The implementation of the proposed ADCT is discussed in unit III. The simulation are given in unit IV. Conclusion arrived from this work is covered in unit V.

II. LITERATURE SURVEY

A. M. Shams et al., [1] have proposed 2-D DCT architecture to improve both area and power simultaneously. To reduce area this architecture is designed with tristate buffer. In this architecture adder is replaced by a low power reversible Vedic adder to improve the overall performance. The DCT encoder consumes less power as reversible gates are used.
R. J. Cintra et al., [2] have proposed twelve approximations for 8-point DCT based on integer functions. The approximation should have very low arithmetic complexity, orthogonality property and low computational complexity. The approximations are achieved without multiplier. It requires only additions and bit shifting operations. They reported high performance and low complexity.
U. S. Potluri et al., [3] have proposed 8-point ADCT technique wherein, only 14 additions are used for shrinking Image and Video. Multiplications are not used here. The proposed transform exhibits low computational complexity. It is comparable to the latest DCT approximations in terms of both algorithm complexity and peak SNR.
U. S. Potluri et al., [4] introduced a hardware compression technique to generate a butterfly structure. It requires less number of additions. In this work a new DA architecture called NEDA is proposed for reducing the cost metrics of power and area.
S. Bouguezel et al., [5] have proposed Walsh – Hadmard transform to develop binary version of a given transform. This method is successfully applied for developing a binary discrete cosine transform (BDCT). In this method only addition operation has been used. This paper shows the ability of the BDCT in approximating the DCT very well.
R. J. Cintra et al., [6] have proposed an approximate method for the evaluation of discrete transform and analysed a class of integer transforms for discrete Fourier, Hartley and cosine transforms. These transform are based on dyadic rational

approximation methods. The approximate transforms have low multiplicative complexity and the orthogonal.

III. IMPLEMENTATION

The image compression process has 3 stages as shown in figure 1. The first stage is image transformation. The input image is decomposed into 16 x 16 blocks of pixels from left to right, top to bottom, the DCT is applied to each block.

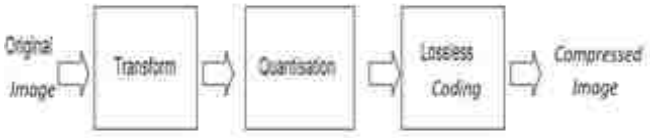

Fig.1: Image compression stages

The second stage is quantization. Each block is compressed through quantization and it is encoded in the third stage. So the channel bandwidth will get reduced and the image takes less space to store. The original image will not be retained. To get the original image we need to do decompression using inverse discrete cosine transform (IDCT). After IDCT the original image will be obtained. The decompression block diagram is shown in figure 2.

Fig.2: Image decompression stages

8 point ADCT signal flow graph (SFG) is shown in Figure 3. As it is seen in the figure, 8 point ADCT involves 22 additions. For 16, 32 and 64 ADCT technique requires 60,152 and 368 additions. It does not require any shift operations. It requires addition and subtraction operations. During hardware implementation shift operation requires only rewiring. It does not require any combinational components. It has lower computational complexity as it requires few operations. It has structural simplicity because we will use only addition and subtraction operations.

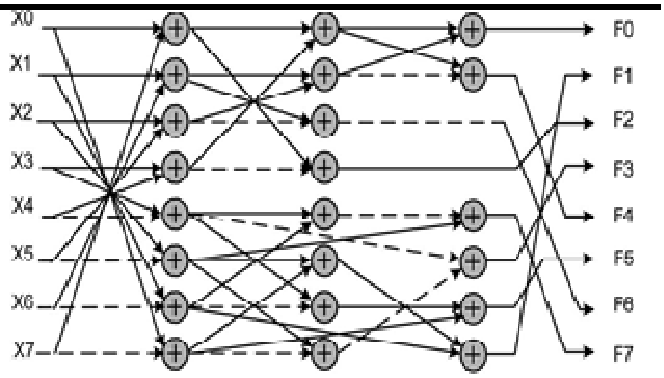

Fig.3: Signal flow graph for 8 point ADCT

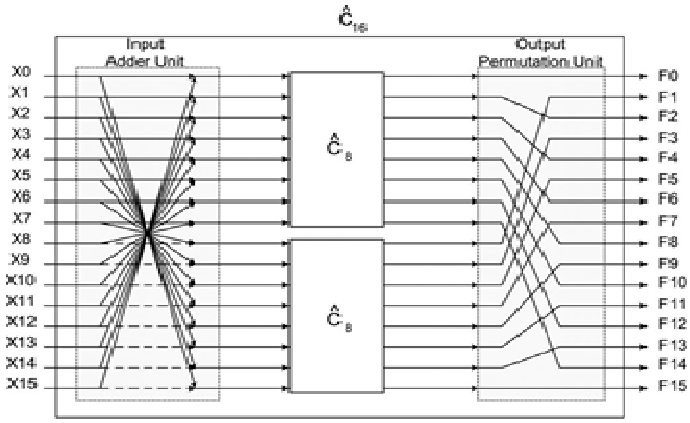

Fig.4: Signal flow graph for 16 point ADCT

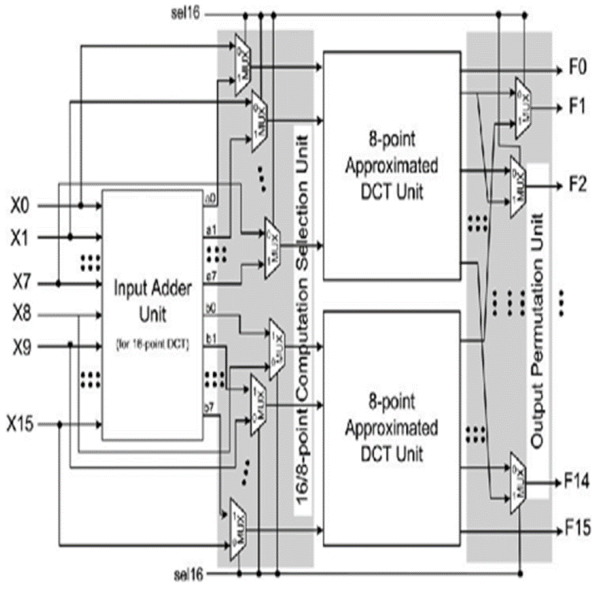

Fig.5: Single block which works as 8 and 16.point ADCT

Signal flow graph for 16 point ADCT is shown in Figure 4. The block diagram consists of two units such as input adder

units and output permutation units. For the computation of two $\hat{C}_8$ the input and output units are used. The 32 point ADCT can be obtained by merging a pair of 16 point ADCTs with an input unit and output unit.

Reconfigurable block diagram for ADCT of lengths N = 8, 16 is shown in Figure 5. The architecture consists of 3 computing units such as two 8 point approximated ADCT units, 16 point input adder unit. The two 8 point ADCT units and a 16 point adder unit that generates the a(i) and b(i), i € [1:7]. The 8 MUXs are used to give inputs for first 8 point ADCT that select either [a(0), a(1),…..,a(7)] or [X(0), X(1),…..,X(7], depending upon whether it is used for 16 point or 8 point ADCT calculation. Similarly, the input to the second 8 point ADCT unit is given with the help of 8 MUXs that select either [b(0),b(1),…..,b(7)] or [X(8),X(9),…..,X(15)],depending upon whether it is used for 16 point ADCT or 8 point ADCT calculation. All MUXs are 2:1. The output block utilizes 14 MUXs to select and route the output depending upon the size of the selected ADCT. The line- Sel16 is used as control input of the MUXs to select and to compute the output according to the size of the ADCT. Sel16 = 1 used for computation of 16 point ADCT and Sel16 = 0 used for the computation of 8 point ADCT in parallel.

Reconfigurable architecture for 32 point ADCT of lengths N = 8, 16 and 32 is shown in Figure 6. The architecture consists of 32 point, 16 point adder units and 8 point ADCT.

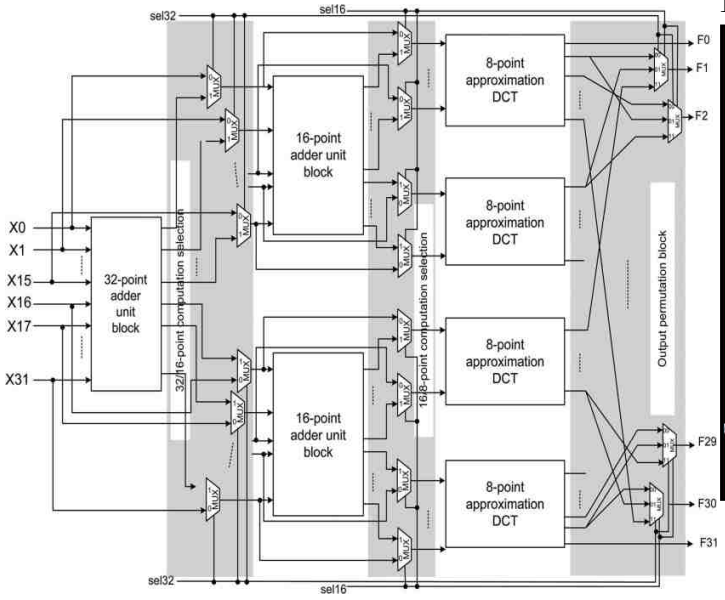

Fig.6: Reconfigurable Architecture for ADCT of lengths N = 8, 16 and 32

The reconfigurability is achieved by three control blocks consisting of sixty four 2:1 MUXs and thirty 3:1 MUXs. The first block decides whether ADCT size is thirty two or lower. If sel32 = 1 the input data is selected for 32 point ADCT otherwise for the ADCTs of lower lengths. The second block decides whether the ADCT size is higher than 8. If sel16 = 1 the length of the ADCT is higher than the 8 otherwise ADCT length is 8.The third block is the output Permutation block which reorders the output depending on the size of the selected ADCT. Sel32 and Sel16 used as the control signals for the 3:1 MUXs. If Sel32 = 0 and Sel16 = 0 used for the computation of four 8 point parallel ADCTs. Sel32 = 0 and sel16 = 1 used for two 16 point parallel ADCTs. Sel32 = 1 and sel16 = 1 used for the computation of 32 point ADCT.

The 8, 16 and 32 point ADCT encoders have been coded using Verilog they are simulated using ModelSim. For area comparison 16 point ADCT and 16 DCT encoders have been implemented Xilinx Spartan ISE. Hardware implementation have been verified using Xilinx 9.1i.

IV. SIMULATION RESULT

For simulation when clock is high the value will get changed but when clock is low the previous value will retain until clock gets high. Using ModelSim tool ADCT coefficients are obtained for 8 point to 32 point ADCT. Simulated results are obtaines using Modelsim. This are shown in Figure 7, 8 and 9 respectively. The reconstructed images are shown in Figure 11, 12 and 13 respectively.

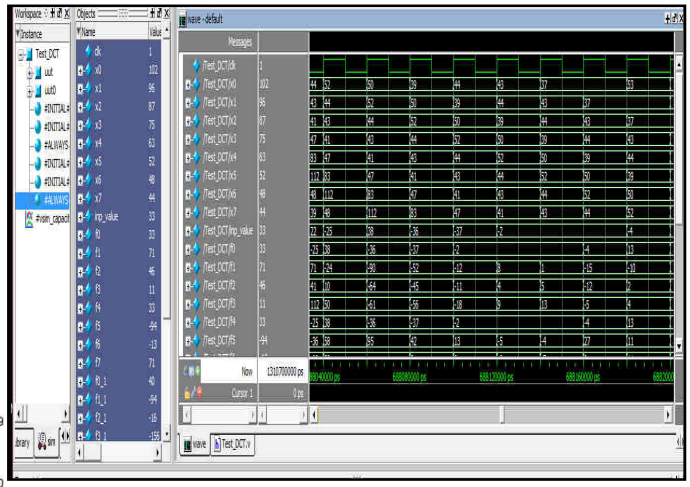

Fig.7: 8 point ADCT coefficeints obtained by simulation

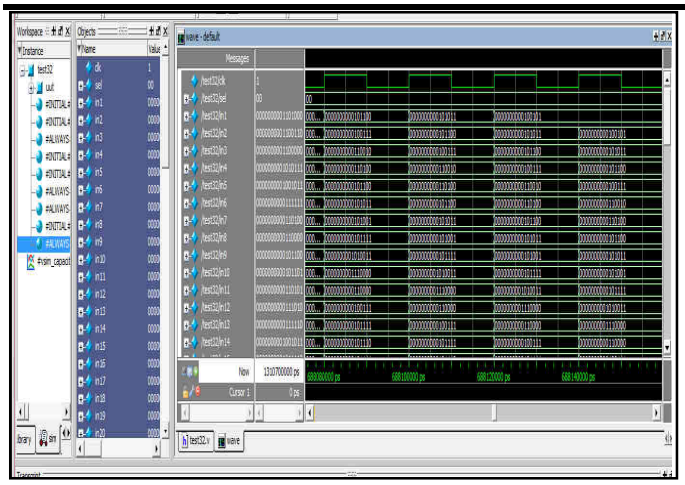

Fig.8: 16 point ADCT coefficeints obtained by simulation

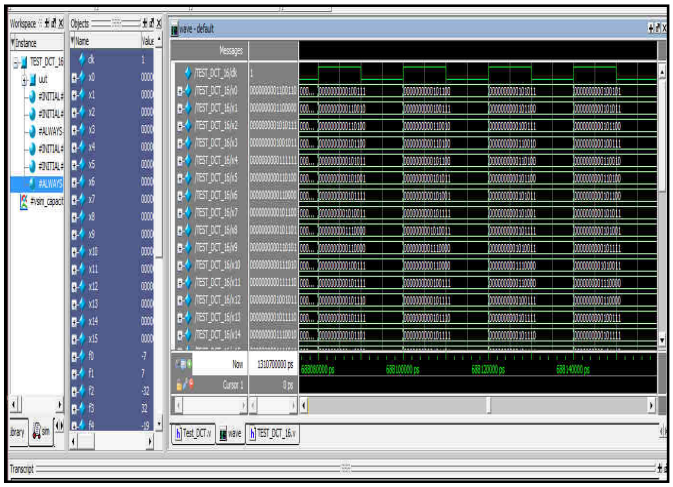

Fig.9: 32 point ADCT coefficeints obtained by simulation

Fig.10: Input image used for all the simulations

Fig.11: Output image obtained for 8 point ADCT

Fig.12: Output image obtained for 16 point ADCT

Fig.13: Output image obtained for 32 point ADCT

To compare the structural complexity of ADCT architecture with that of exact DCT, DCT architecture is also implemented

for length N = 16.The 16 point DCT simulation result, input image, output image and IDCT image are shown in Figure 14, 10, 15, and 16 respectively. Simulation result of 16 point DCT shown in figure 14.

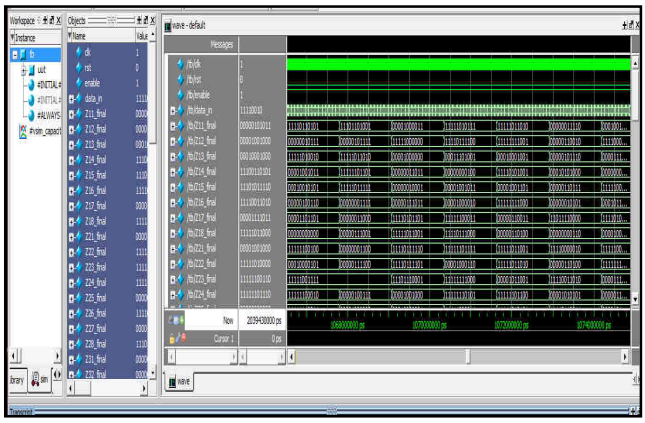

Fig.14: Simulation result of 16 point DCT

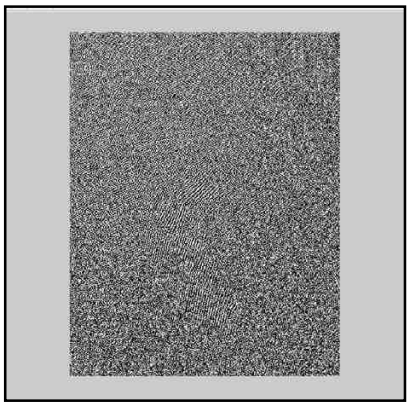

Fig.15: 16 point DCT output image

Fig.16: IDCT image

Snap shots of synthesis results obtained from ISE TOOL for 8/16 /32 point ADCT and 16 point DCT are shown in Figure 17, 18, 19, 20 and 21 respectively .Synthesis results are compared for 16 point ADCT implementation with 16 point DCT implementation.

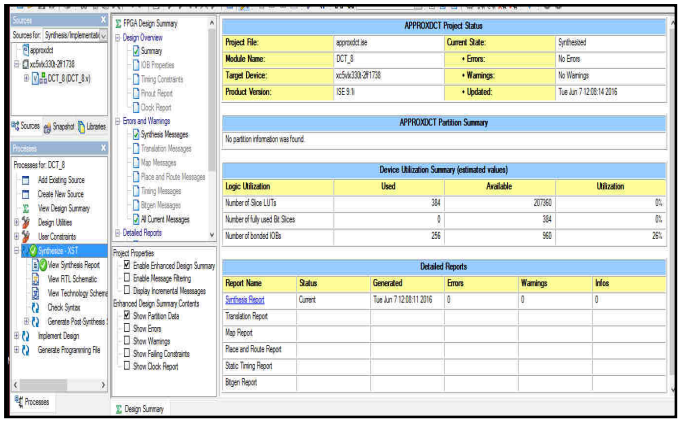

Fig.17: synthesis result for 8 point ADCT

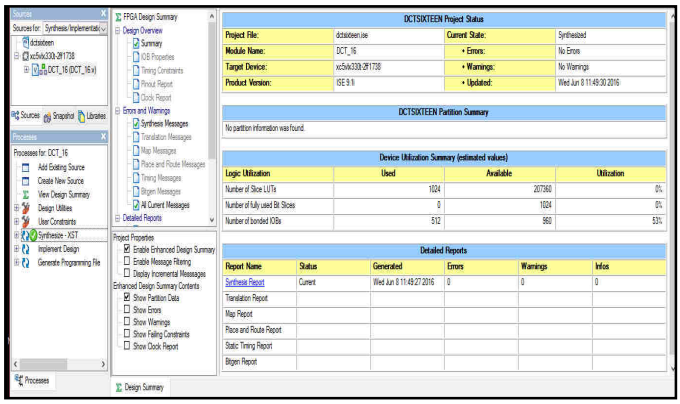

Fig.18: Synthesis result for 16 point ADCT

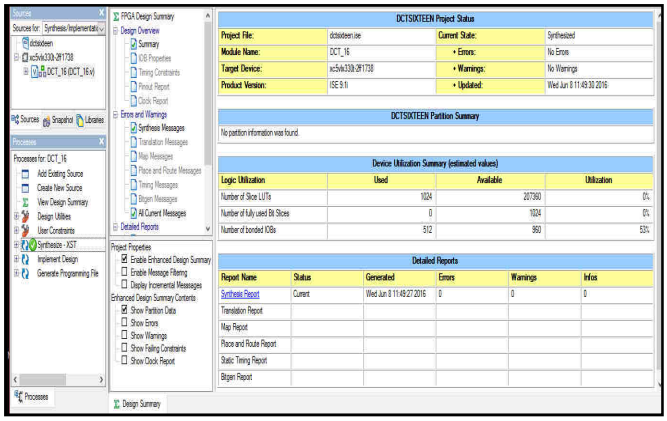

Fig.19: Synthesis result for reconfigurable 16 point ADCT

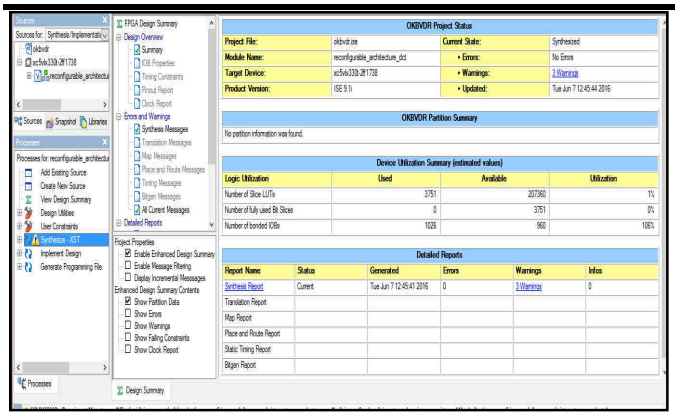

Fig. 20: Synthesis result for reconfigurable 32 point ADCT

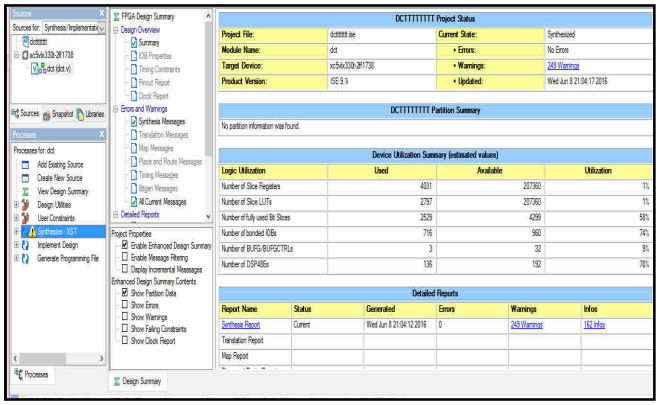

Fig. 21: Synthesis report for 16 point DCT

The comparison of the resources used by the target device is given in the table 1. It is shown that ADCT implementation of length N = 16 consumes less resources compared to DCT implementation of length N = 16.

Table 1: Device usage summary

	Available	Used for ADCT	Used for DCT	% Saved
LUTs	207360	1024	2797	63
IOBs	960	512	716	28.37

V. CONCLUSION

Compression is required to save the channel bandwidth while transmitting images. It is also required to save the memory space while storing. There are several compression techniques available. Among them, DCT technique is popular. But DCT technique is non orthogonal with high structural and computational complexity. To overcome these drawbacks, improvised DCT namely ADCT technique is proposed in this paper. ADCT architectures for lengths N = 8, 16 and 32 have been modelled using Verilog and same are verified. In this work a reconfigurable architecture for ADCT of lengths N =8, 16 and 32 is proposed, implemented and verified. In existing DCT technique the reconstruction of the image is done using IDCT. But, for proposed ADCT it is not required. DCT uses adders, subtractors, multipliers and shifters. Proposed ADCT is orthogonal and it has lower computational complexity and structural simplicity. It uses only adders and subtractors. In future the proposed ADCT technique discussed in this paper may be extended to 64 point ADCT. By doing ASIC implementation of these architectures using Cadence tool, power can be estimated and a low power design can be attempted.

REFERENCES

[1] A. M. Shams, A. Chidanandan, W. Pan, and M. A. Bayoumi, "NEDA: A low-power high-performance DCT architecture," IEEE Trans.Signal Process., vol. 54, no. 3, pp. 955–964, 2006.

[2] R. J. Cintra, F. M. Bayer, and C. J. Tablada, "Low-complexity 8-pointDCT approximations based on integer functions," Signal Process, vol.99, pp. 201–214, 2014.

[3] 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. Circuits Syst. I, Reg. Papers, vol. 61, no. 6, pp. 1727–1740, Jun. 2014.

[4] S. Bouguezel, M. Ahmad, and M. N. S. Swamy, "A novel transform for image compression," in Proc. 2010 53rd IEEE Int. Midwest Symp. Circuits Syst. (MWSCAS), pp. 509–512.

[5] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "Binary discrete cosine and Hartley transforms," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 60, no. 4, pp. 989–1002, Apr. 2013.

[6] R. J. Cintra, "An integer approximation method for discrete sinusoidal transforms," Circuits, Syst., Signal Process., vol. 30, no. 6, pp.1481–1501, 2011.