IOSR Journal Of VLSI And Signal Processing (IOSR-JVSP)
Volume 6, Issue 3, Ver. Ii (May. -Jun. 2016), Pp 82-90
E-ISSN: 2319 – 4200, P-ISSN No. : 2319 – 4197
Www.Iosrjournals.Org

Reconfigurable Architecture and an Algorithm for Scalable And Efficient Orthogonal Approximation of Dct

Divyarani[1], Ashwath Rao[2]
[1]*ECE department, Sahyadri College of Engineering and Management, India*
[2]*Associative Professor ECE department, Sahyadri College of Engineering and Management, India*

Abstract: *This proposed paper presents architecture of generalized recursive function to generate approximation of orthogonal function DCT with an approximate length N could be derived from a pair of DCTs of length (N/2) at the cost of N additions for input preprocessing. Approximation of DCT is useful for reducing its computational complexity without impact on its coding performance. Most of the existing design for approximation of the DCT target only the small transform lengths DCT, and some of them are non-orthogonal. Proposed method is highly scalable for hardware and software implementation of DCT of higher lengths, and it can make use of the present approximation of 8-point DCT to obtain DCT approximation of any power of two length, N>8. It is shown that proposed design involves lower arithmetic complexity compared with the other existing design. One uniquely interesting feature of the proposed method is that it could be composed for the calculation of a 32-point DCT or for parallel calculation of two 16-point DCTs or four 8-point DCTs. The proposed method is found to offer many advantages in terms of hardware regularity , modularity and complexity. The design is implemented in Xillinx IES 10.1 design suite and synthesized using Cadence Encounter.*

Keywords*: Algorithm-architecturecodesign, DCT approximation, discrete cosine transform, high efficiency video coding.*

I. Introduction

The DCT is popularly used in image and video compression. The main purpose of the approximation algorithms is to eliminate multiplications which consume more power and computation time. The use of approximation is important for higher-size DCT since the computational difficuties of the DCT grows nonlinearly. Haweel [8] has proposed the signed DCT for 8 X 8 blocks where the basis vector elements are given by their sign, i.e,±1. Bouguezel-Ahmad-Swamy have proposed many methods. They have given a good estimation of the DCT by replacing the basis vector elements by 0, ±1/2, ±1 [7]. In the paper [5], [6] Bayer and Cintra have proposed two transforms derived from 0 and ±1 as elements of transform kernel, and have proved that their methods gives better than design in [7], particularly for low- and high-compression ratio scenarios. Modern video coding standards such as high efficiency video coding [10] uses DCT of larger block sizes (up to 32×32) in order to perform higher compression ratio. But, the extension of the design strategy used in H264 AVC for larger transform sizes is not possible [11]. Besides, in many image processing applications such as tracking [12] and simultaneous compression and encryption [13] needs higher DCT sizes. In this case, Cintra has introduced a new class of integer transforms applicable to many block-lengths [14]. Cintra *et al.* have proposed a new 16 X 16 matrix also for approximation of 16-point DCT, and have validated it experimentally [15]. Two new transforms have been proposed for 8-point DCT approximation: Cintra *et al.* have developed a less-complexity 8-point DCT approximation based on integer functions [16] and Potluri *et al.* have proposed a new 8-point DCT approximation that uses only 14 additions [17]. On the other hand, Bouguezel *et al.* have proposed two designs wchich is multiplication-free approximate form of DCT. The first method is for length N = 8 , 16 and 32; and is mainly based on the relevant extension of integer DCT [18]. Also, by using the sequency-ordered Walsh-Hadamard transform proposed in [4] a systematic method for developing a binary version of higher-size DCT is developed. This transform is a permutated method of the WHT which gives all the benifits of the WHT.

A scheme of approximation of DCT should have the following features:

i) It should have low computational complexity.
ii) It should have low error energy to give compression performance near to exact DCT, and preferably should be orthogonal.
iii) It should applicable for higher lengths of DCT to support modern video coding standards, and other applications like surveillance, tracking, encryption and simultaneous compression.

Some of the existing methods are deficient in terms of scalability [18], generalization for higher sizes [15], and orthogonality [14]. This proposed design try to maintain orthogonality in the approximation of DCT for two reasons. Firstly, if the transform is orthogonal, then find its inverse, and the kernel matrix of this inverse transform is obtained by transposing the kernel matrix of the forward transform. This feature of inverse transform could be used to compute the inverse and forward DCT by similar computing structures. Moreover, in case of orthogonal transforms, similar fast algorithms are relevant to both inverse and forward transforms [19], [20]. This paper proposes an algorithm to derive approximate form of DCTs which satisfy all the three features. This paper obtain the proposed approximate form of DCT by recursive decomposition of sparse DCT matrix. It shows that proposed method involves lower arithmetic complexity than the existing DCT approximation algorithms. The proposed DCT approximation form of different lengths are orthogonal, and result in lower error-energy compared to the existing system. The decomposition process gives generalization of a proposed transform for higher-size DCTs and proposed algorithm is easily scalable for hardware and software implementation of higher lengths DCT. Based on the proposed algorithm, paper has proposed a fully scalable, reconfigurable, and parallel architecture for the computation of approximate DCT. One unique feature of proposed design is that the structure for the computation of 32-point DCT could be used for the parallel exicution of two 16-point DCTs or four 8-point DCTs. The proposed method is more usefull than the existing methods in terms of hardware complexity and energy compaction.

II. Proposed System Design And Methodology

The elements of -point DCT matrix C_N are given by:

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

where $0 \le i, j \le N-1$, $\epsilon_0 = 1/\sqrt{2}$, and $\epsilon_i = 1$ for i>0. The DCT given by (1) is referred to as exact DCT in order to disting- uish it from approximated forms of DCT. For $k \in [0,(N/2) - 1]$ and $i = 2k$, for any even value of N it can find that

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

since $\epsilon_{2k} = \epsilon_k$, (2) can be rewritten as:

$$c(2k,j) = \epsilon_k \sqrt{\frac{2}{N}} \cos(2j+1)k\pi N \quad (3)$$

Hence, on the right-hand side of (3) the cosine transform kernel corresponds to N/2-point DCT and its elements can be assumed to be $\sqrt{2c}$ (k, j), for $0 \le j \le (N/2) - 1$. Hence, the first N/2 elements of even rows of DCT matrix of size N × N corresponds to the N/2-point DCT matrix. Accordingly, the recursive decomposition of C_N can be performed as detailed in (4)–(8). Using odd/even symmetries of its row vectors, DCT matrix C_N can be represented by the following matrix product

$$C_N = \frac{1}{\sqrt{2}} M_N^{per} \; T_N \; M_N^{add} \quad (4)$$

where T_N is a block sparse matrix expressed by:

$$T_N = \begin{bmatrix} C_{\frac{N}{2}} & 0_{\frac{N}{2}} \\ 0_{\frac{N}{2}} & S_{\frac{N}{2}} \end{bmatrix} \quad (5)$$

where $0_{N/2}$ is ((N/2) × (N/2)) zero matrix. The submatrix $S_{N/2}$ consists of odd rows of the first N/2 columns of $\sqrt{(2)}C_N$. M_N^{per} is a permutation matrix expressed by:

$$M_N^{per} = \begin{bmatrix} P_{N-1,\frac{N}{2}} & 0_{1,\frac{N}{2}} \\ 0_{1,\frac{N}{2}} & P_{N-1,\frac{N}{2}} \end{bmatrix} \quad (6)$$

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

$$P_{N-1,\frac{N}{2}}^{(i)} = \begin{cases} 0_{1,\frac{N}{2}}, & if\ i = 1,3,\dots,N-1 \\ I_{\frac{N}{2}\left(\frac{i}{2}\right)}, & if\ i = 0,2,\dots,N-2 \end{cases} \quad (7)$$

where $I_{N/2}(i/2)$ is (i/2) th row vector of the ((N / 2) × (N / 2)) identity matrix and M_N^{add} is defined by:

$$M_N^{add} = \begin{bmatrix} I_{\frac{N}{2}} & J_{\frac{N}{2}} \\ I_{\frac{N}{2}} & -J_{\frac{N}{2}} \end{bmatrix} \quad (8)$$

where $J_{N/2}$ is an ((N/2) × (N/2)) matrix having all ones on the anti-diagonal and zeros elsewhere.

To reduce the computational complexity of DCT, the computational cost of matrices presented in (4) is required to be assessed. Since M_N^{per} does not involve any arithmetic or logic operation, and M_N^{add} requires N/2 additions and N/2 subtractions, they contribute very less to the total arithmetic complexity and cannot be

reduced further. Therefore, for reducing the computational complexity of N-point DCT, it needs to approximate T_N in (5). Let $\hat{C}_{N/2}$ and $\hat{S}_{N/2}$ denote the approximation matrices of $C_{N/2}$ and $S_{N/2}$,respectively. To find these approximated submatrices it need to take the smallest size of DCT matrix to stop the approximation procedure to 8, since 4-point DCT and 2-pointDCT can be implemented by adders only. Correspondingly, a good C_N approximation , where N is an integral power of two, N $\geq$ 8, leads to a proper approximations of C_8 and S_8 . For approximation of C_8, choose the 8-point DCT given in [6] since that presents the best trade-off between quality of the reconstructed image and the number of required arithmetic operators. The trade-off analysis given in [6] shows that approximating C_8 by $\hat{C}_8 = \lfloor 2C_8 \rfloor$ where $\lfloor . \rfloor$ denotes the rounding-off operation outperforms the current state-of-the-art of 8-point approximation methods.

From (4) and (5), observe that C_8 operates on sums of pixel pairs while S_8 operates on differences of the same pixel pairs. Therefore, by replacing $\hat{S}_8$ by $\hat{C}_8$, there are two main advantages. Firstly, there is a good compression performance due to the efficiency of $\hat{C}_8$ and secondly the implementation will be much simpler, scalable and reconfigurable. For approximation of S_8 this paper has investigated two other low-complexity alternatives, and in the following paper will discuss three possible options of approximation of S_8:

i) The first one is to approximate S_8 by null matrix, which implies all even-indexed DCT coefficients are takenas zero. The transform obtained from this approximation is far from the exact values of even-indexed DCT coefficients, and odd coefficients do not have any other information.

ii) The second solution is gien by approximating S_8 by 8 × 8 matrix where each row contains one 1 and other all elements are zeros The approximate transform in this case is nearer to the exact DCT when compared to the solution obtained by null matrix.

iii) The third solution consists of approximation S_8 of by $\hat{C}_8$ .Since as well as S_8 are submatrices of C_{16} and operate on matrices generated by differences and sum of pixel pairs at a distance of 8, approximation of S_8 by $\hat{C}_8$ has attractive computational properties: good compression efficiency, orthogonality since $\hat{C}_8$ is orthogonalizable, and regularity of the signal-flow graph, other than scope for reconfigurable implementation and scalability.

Based on this third possible approximation of S_8 , this paper has obtained the proposed approximation of as:

$$\hat{C}_N = \frac{1}{\sqrt{2}} M_N^{per} \begin{bmatrix} \hat{C}_{\frac{N}{2}} & 0_{\frac{N}{2}} \\ 0_{\frac{N}{2}} & \hat{C}_{\frac{N}{2}} \end{bmatrix} M_N^{add} \quad (9)$$

As state d before, matrix $\hat{C}_N$ is orthogonalizable. Indeed, for each $\hat{C}_N$ we can calculate D_N given by:

$$D_N = \sqrt{\left(\hat{C}_N \times \left(\hat{C}_N\right)^t\right)^{-1}} \quad (10)$$

where $(.)^t$ denotes matrix transposition. For data compression, use $C_N^{orth} = D_N \times \hat{C}_N$ instead of $\hat{C}_N$ since $(C_N^{orth})^{-1} = (C_N^{orth})^t$. Since D_N is a diagonal matrix, it can be integrated into scaling in quantization process. Therefore, as adopted in [4]–[8], the computational cost of is equal to that of Moreover, the term of in (9) can be integrated in the quantization step in order to get multiplerless architecture. The design for the generation of the proposed orthogonal approximated DCT is stated in Algorithm 1.

Algorithm 1 for proposed DCT matrix

function PROPOSED DCT(N) ⋯ N power of 2, N ≥ 8
  $N_O \leftarrow log_2(N/8)$ ⋯ N_0 is the number of 8-sample blocks
  $\hat{C}_{N/2^{N_0}} \leftarrow \lfloor 2C_8 \rfloor$
  while $N_O > 0$ **do**
    $\tilde{N} \leftarrow (N/2^{N_O} - 1)$
    Calculate $M_{\tilde{N}}^{per}$, $M_{\tilde{N}}^{add}$ ⋯Eq(6),(8)
    Calculate $\hat{C}_{\tilde{N}}$ ⋯ Eq(9)
    $N_O \leftarrow N_O - 1$
  end while
  Calculate $\boldsymbol{D_N}$ ⋯Eq(10)
  return $\hat{\boldsymbol{C}}_{\boldsymbol{N}}$ **,** $\boldsymbol{D_N}$
end function

III. Scalable and Reconfigurable Architecture For Dct Computation

This section discuss about the proposed scalable architecture for the computation of approximate DCT of N = 16 and 32. Paper has derived the theoretical estimate of its hardware complexity and discuss the reconfiguration scheme.

A. Proposed Scalable Design

The block diagram of the computation of DCT based on $\hat{C}_8$ is shown in Fig. 1. For a given input sequence $\{X(n)\}$, n $\in [0, N-1]$, the approximate DCT coefficients are obtained by F = $\hat{C}_N \cdot X^t$. An example of the block diagram of $\hat{C}_{16}$ is illustrated in Fig. 2, where two units for the computation of $\hat{C}_8$ along with an input adder unit and output permutation unit are used. The functions of these two blocks are shown respectively in (8) and (6).Note that structures of 16-point DCT of Fig. 2 could be used to obtain the DCT of higher sizes.

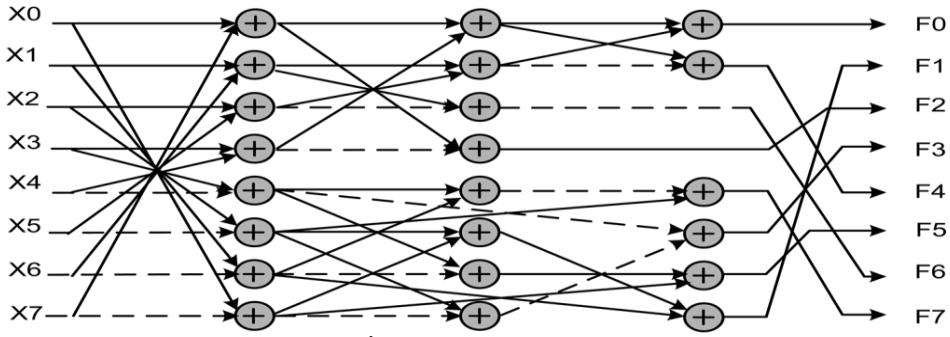


Fig. 1 Signal flow graph of $\hat{C}_8$. Dashed arrows represent multiplications by -1

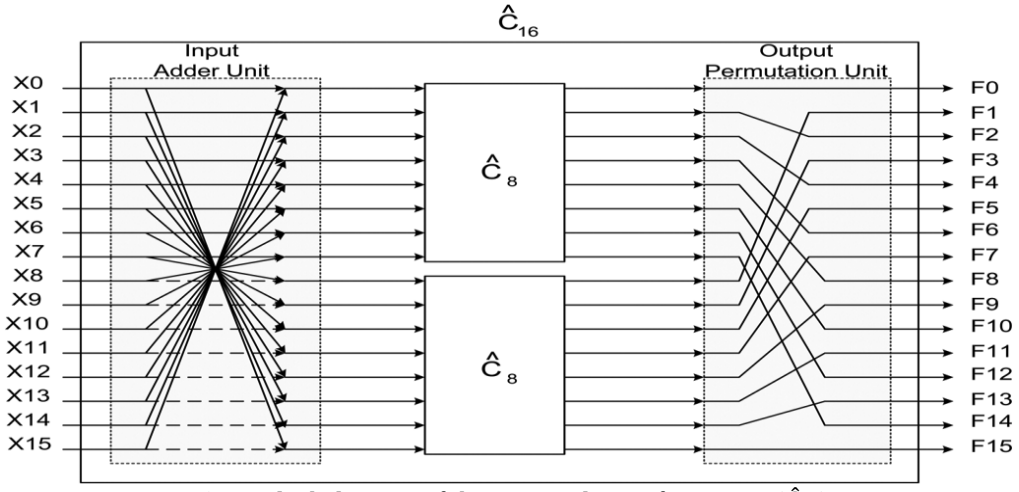


Fig. 2 Block diagram of the proposed DCT for N = 16 ($\hat{C}_{16}$)

Table I: Requirement of Arithmetic operations For Several Approximation Algorithms

N	Method	Arithmetic operation	
		Add	Shift
8	Proposed	22	0
	BDCT [4]	24	0
	BAS [18]	24	4
	BC [6]	22	0
16	Proposed	60	0
	BDCT [4]	64	0
	BAS [18]	64	8
	BC [15]	72	0
32	Proposed	152	0
	BDCT [4]	160	0
	BAS [18]	160	16
64	Proposed	368	0
	BDCT [4]	384	0

B. Complexity Comparison

To obtain the computational complexity of proposed *N*–point approximate DCT, it required to determine the executing cost of matrices given in (9). As in Fig. 1 the 8-point DCT approximation uses 22 additions. Since M_N^{per} has no computational cost and M_N^{add} requires N additions for N –point DCT, the overall arithmetic complexity of 16-point, 32-point, and 64-point approximate DCT are 60, 152, and 368 additions, respectively. More generally, the arithmetic complexity of N-point DCT is equal to $N(log_2 N - (1/4))$ additions. Since the structures for the computation of DCT of different lengths are scalable and regular, the

computational time for N DCT coefficients can be found using $log_2(N)T_A$ where T_A is the addition-time. The number of arithmetic operations used in proposed approximate DCT of different lengths are shown in Table I. It can be found that the proposed method requires the less number of additions, and does not use any shift operations. Note that shift operation needs only rewiring during hardware implementation and does not involve any combinational components.

Using an integrated logic analyze pipelined and non-pipelined designs of different methods are developed, synthesized and validated and validation is done using the Digilent EB of Spartan6-LX45. This method used 8-bit inputs, and have allowed the increase of output size . For the 8-point transform of Fig. 1, there is 11-bit and 10-bit outputs. By adding registers in the input and output stages along with registers after each adder stage the pipelined design are obtained, while the no pipeline registers are used within the non-pipelined designs. The synthesis results obtained from XST synthesizer are presented in Table II. It shows that pipelined designs gives higher maximum operating frequency and shows that the proposed design involves nearly 7%, 6%, and 5% less area when compared to BDCT method for N equal to 16, 32, and 64, respectively. Both pipelined and non-pipelined designs involve the same number of LUTs. Mainly, for different transform lengths,the proposed designs are reusable.

C. Proposed Reconfiguration Scheme

DCT of different lengths required to be used in video coding applications. Therefore the proposed method should be reused for the DCT of different lengths instead of using separate structures for different lengths. The reconfigurable architecture for the implementation of approximated 16-point DCT is shown in Fig. 3. It consists of three computing units, namely two 8-point DCT approximation units and a 16-point input adder unit that generates *a(i)* and *b(i), i* $\in$ [1: 7]. The input to the first 8-point DCT approximation unit is fed through 8 MUXes that select either [*a*(0), *a*(1), . . . ,*a*(7)] or [*X*(0), *X*(1), . . . , *X*(7)], depending on whether it is used for 16-point DCT calculation or 8-point DCT calculation. Similarly, the input to the second 8-point DCT unit (Fig. 3) is fed through 8 MUXes that select either [*b*(0), *b*(1), . . ,*b*(7)] or [*X*(8), *X*(9), . . ,*X*(15)], depending on whether it is used for 16-point DCT calculation or 8-point DCT calculation. On the other hand, the output permutation unit uses 14 MUXes to select and re-order the output depending on the size of the selected DCT. *Sel*16 is used as control input of the MUXes to select inputs and to perform permutation according to the size of the DCT to be computed. Specifically, *Sel*16 = 1 enables the computation of 16-point DCT and *Sel*16 = 0 enables the computation of a pair of 8-point DCTs in parallel. Consequently, the architecture of Fig. 3 allows the calculation of a 16-point DCT or two 8-point DCTs in parallel.

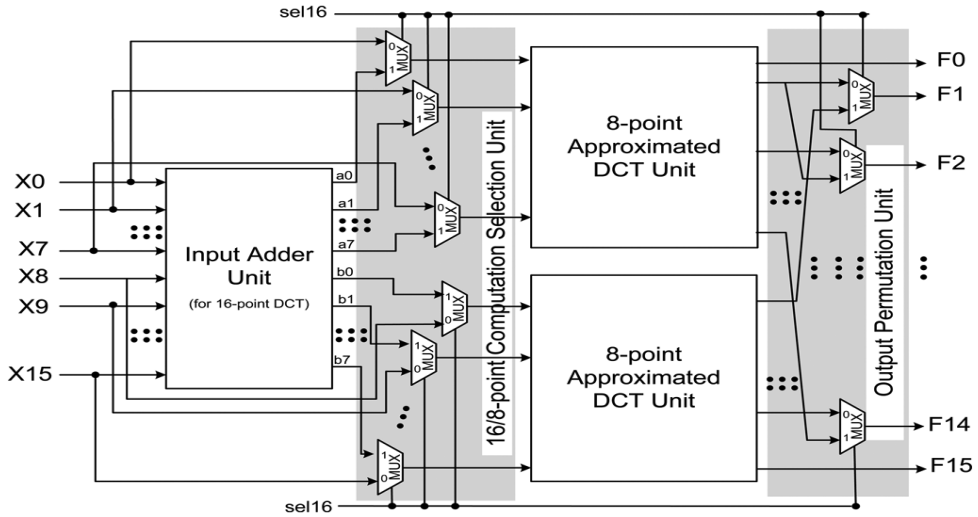


Fig. 4 Proposed reconfigurable architecture for approximate DCT of lengths N = 8 , 16 and 32.

A reconfigurable design for the computation of 32, 16, and 8-point DCTs is presented in Fig. 4. It performs the calculation of a 32-point DCT or two 16-point DCTs in parallel or four 8-point DCTs in parallel. The architecture is composed of 32-point input adder unit, two 16-point input adder units, and four 8-point DCT units. The reconfigurability is achieved by three control blocks composed of 64 2:1 MUXes along with 30 3:1 MUXes. The first control block decides whether the DCT size is of 32 or lower. If *Sel*32 = 1, the selection of input data is done for the 32-point DCT, otherwise, for the DCTs of lower lengths. The second control block decides whether the DCT size is higher than 8. If *Sel*16 = 1 the length of the DCT to be computed is higher than 8 otherwise, the length is 8. The third control block is used for the output permutation unit which re-orders the

output depending on the size of the selected DCT. *Sel*32 and *Sel*16 are used as control signals to the 3:1 MUXes. Specifically, for $\{Sel32, Sel16\ \}_2$ equal to {00}, {01} or {11} the 32 outputs correspond to four 8-point parallel DCTs, two parallel 16-point DCTs, or 32-point DCT, respectively.

IV. Experimental Validation

The proposed system simulation results as follows:

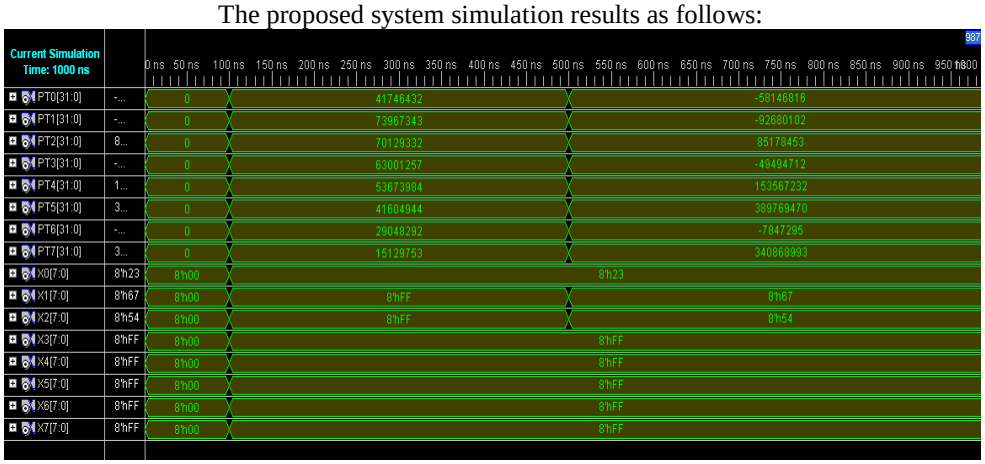


Fig. 5 8-point DCT output.

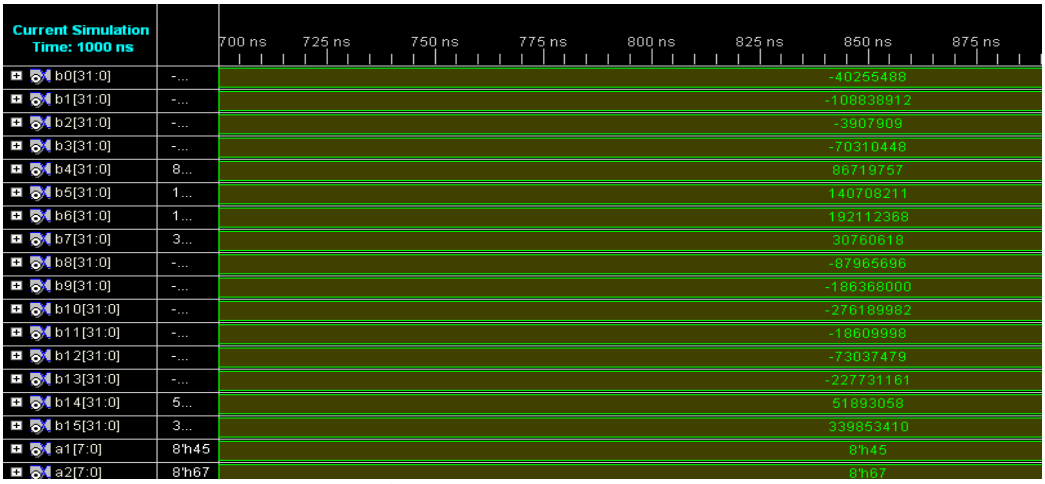
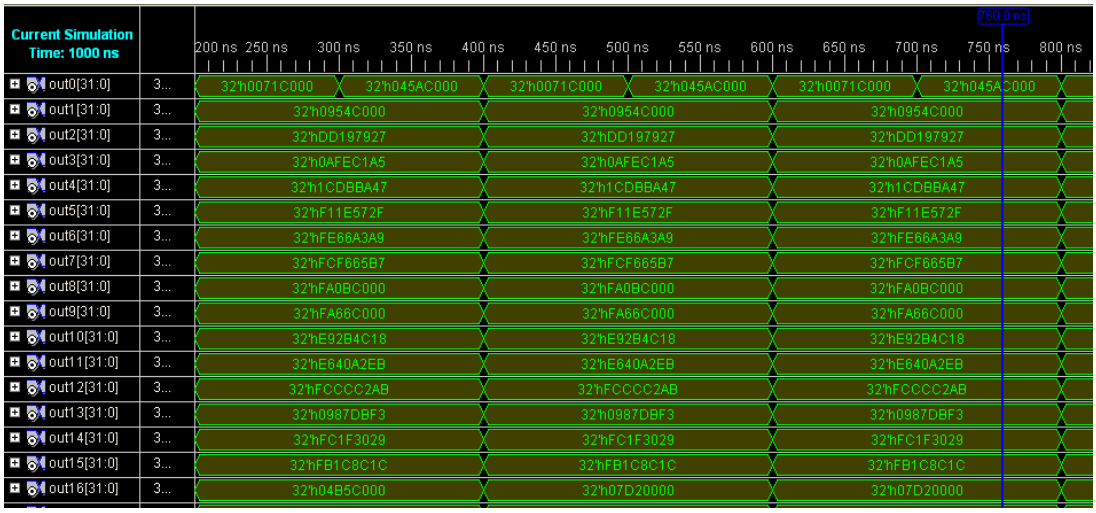


Fig. 6 16-point DCT output



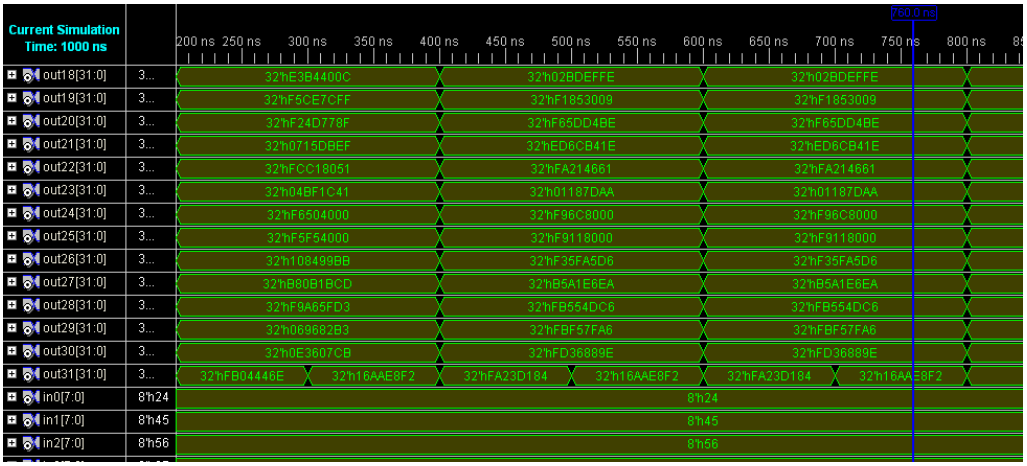


Fig. 7 32-point DCT output

The proposed system synthesized result using Cadence Encounter:

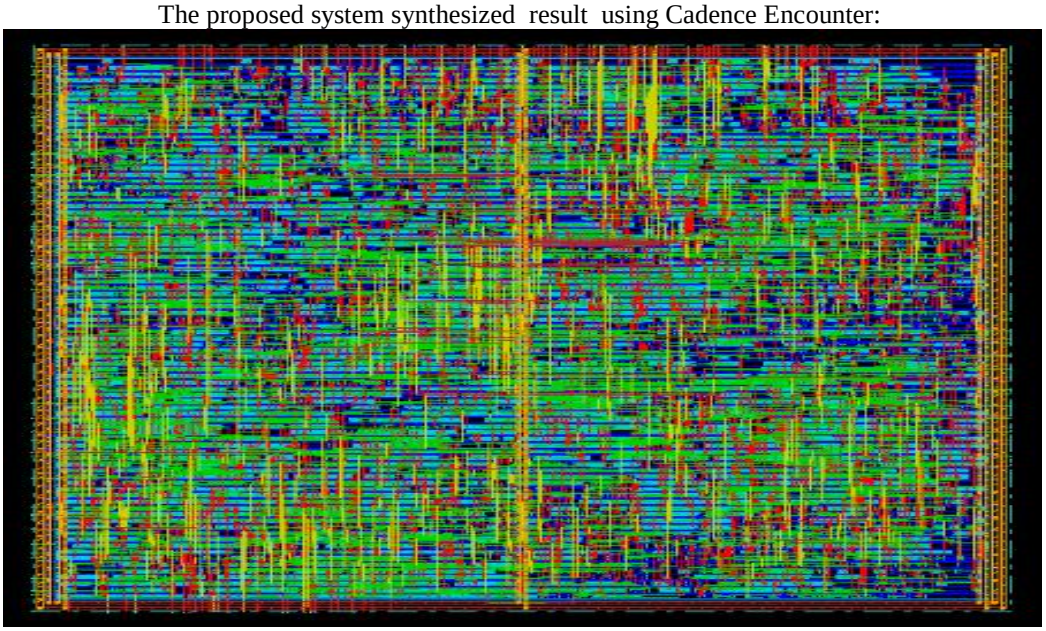


Fig. 8 GDS report of 8-point DCT

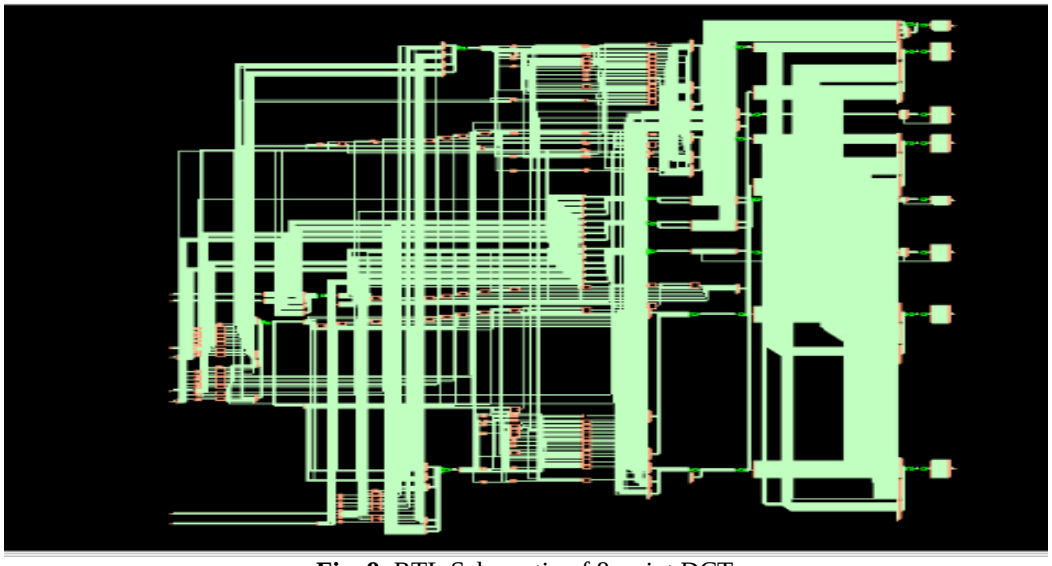


Fig. 9 RTL Schematic of 8-point DCT

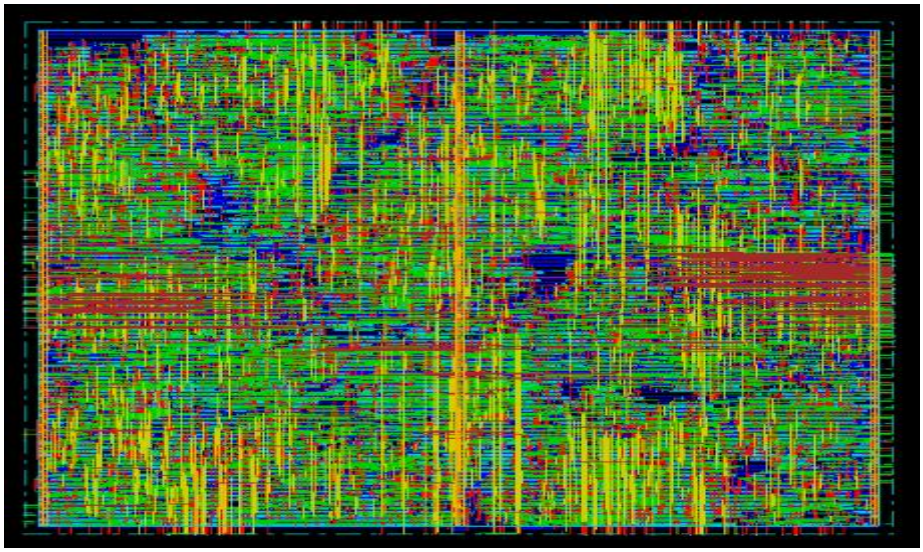


Fig. 10 GDS report of 16-point DCT

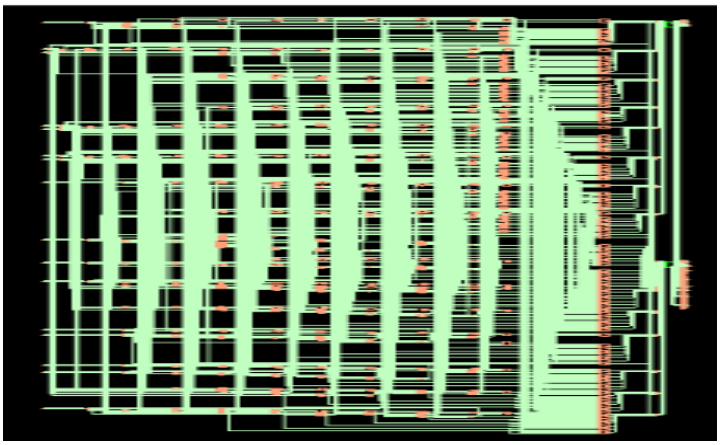


Fig. 11 RTL Schematic of 16-point DCT

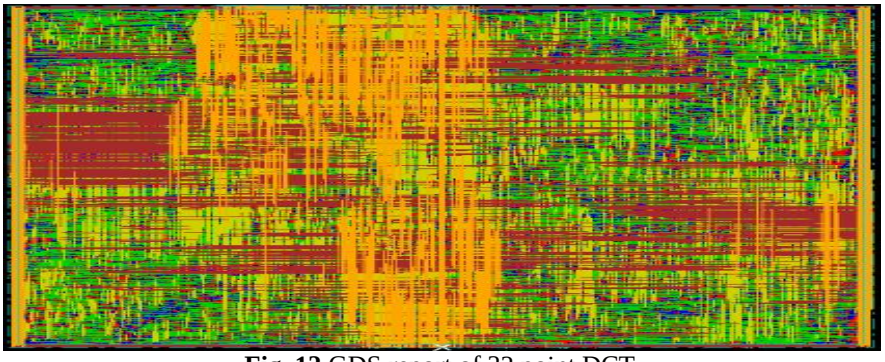


Fig. 12 GDS report of 32 point DCT

V. Conclusions

This paper has proposed a recursive algorithm to obtain orthogonal approximation of DCT. The value of length *N* could be obtained from a DCT pair of length (*N/2*) at the cost of *N* additions for input preprocessing. In this proposed architecture it contains many advantages like regularity, structural simplicity, lower-computational complexity, and scalability. Comparison with existing competing methods shows the effectiveness of error energy, hardware resources consumption, and compressed image quality to obtain better quality in proposed architecture. Paper has also proposed a fully scalable architecture which is reconfigurable for approximate DCT computation where the computation of 32-point DCT could be configured for two 16-point DCTs or four 8-point DCTs placed in parallel.

Acknowledgment

I hereby take this opportunity to express my gratitude and thankfulness to all those concerned in helping me in working on this project. I would like to thank my project guide Mr. Ashwath Rao, Associative professor of Electronics and Communication Engineering department for his continuous encouragement while working on the project.

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] C. Loeffler, A. Lightenberg, and G. S. Moschytz, "Practical fast 1-D DCT algorithm with 11 multiplications," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, May 1989, pp. 988–991.
[3] M. Jridi, P. K. Meher, and A. Alfalou, "Zero-quantised discrete cosine transform coefficients prediction technique for intra-frame video encoding," *IET Image Process.*, vol. 7, no. 2, pp. 165–173, Mar. 2013.
[4] 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.
[5] 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.
[6] 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.
[7] S. Bouguezel, M. Ahmad, and M. N. S. Swamy, "Low-complexity 8× 8 transform for image compression," *Electron. Lett.*, vol. 44, no. 21, pp. 1249–1250, Oct. 2008.
[8] T. I. Haweel, "A new square wave transform based on the DCT," *Signal Process.*, vol. 81, no. 11, pp. 2309–2319, Nov. 2001.
[9] V. Britanak, P.Y.Yip, and K. R. Rao, Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations. London, U.K.: Academic, 2007.
[10] G. J. Sullivan, J.-R. Ohm,W.-J.Han, and T.Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
[11] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, 2012.
[12] X. Li, A. Dick, C. Shen, A. van den Hengel, and H. Wang, "Incremental learning of 3D-DCT compact representations for robust visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 4, pp. 863–881, Apr. 2013.
[13] A. Alfalou, C. Brosseau, N. Abdallah, andM. Jridi, "Assessing the performance of a method of simultaneous compression and encryption of multiple images and its resistance against various attacks," *Opt. Express*, vol. 21, no. 7, pp. 8025–8043, 2013.
[14] R. J. Cintra, "An integer approximation method for discrete sinusoidal transforms," *Circuits, Syst., Signal Process.*, vol. 30, no. 6, pp. 1481–1501, 2011.
[15] F. M. Bayer, R. J. Cintra, A. Edirisuriya, and A. Madanayake, "A digital 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.
[16] R. J. Cintra, F. M. Bayer, and C. J. Tablada, "Low-complexity 8-point DCT approximations based on integer functions," *Signal Process.*, vol. 99, pp. 201–214, 2014.
[17] U. S. Potluri, A.Madanayake, R. J. Cintra, F.M. Bayer, S. Kulasekera, andA. 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.
[18] 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.
[19] K. R. Rao and N. Ahmed, "Orthogonal transforms for digital signal processing," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 1976, vol. 1, pp. 136–140.
[20] 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.
[21] "USC-SIPI image database,"Univ. Southern California, Signal and Image Processing Institute [Online]. Available: http://sipi.usc.edu/database/, 2012