International Journal for Modern Trends in Science and Technology
Volume: 03, Issue No: 09, September 2017
ISSN: 2455-3778
http://www.ijmtst.com

An Efficient 16 Point Reconfigurable DCT Architecture Using Improved 8 Point DCT

P. Bose Babu[1] | M.S.Lakshmi[2]

[1]Assistant Professor, Department of ECE, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India.
[2]PG Scholar, Department of ECE, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India.

To Cite this Article
P. Bose Babu and M.S.Lakshmi, "An Efficient 16 Point Reconfigurable DCT Architecture Using Improved 8 Point DCT", International Journal for Modern Trends in Science and Technology, Vol. 03, Issue 09, September 2017, pp.114-120

ABSTRACT

In this paper we present a reconfigurable 32 Approximation of discrete cosine transform (DCT) using 32 adder unit and 16 bit adder unit 8 point additions with less number of calculations. Most of the existing algorithms for approximation of the DCT target only the DCT of small transform lengths, and some of them are non-orthogonal. This paper presents a generalized recursive algorithm to obtain orthogonal approximation of DCT where an approximate DCT of length could be derived from a pair of DCTs of length at the cost of additions for input preprocessing. We perform recursive sparse matrix decomposition and make use of the symmetries of DCT basis vectors for deriving the proposed approximation algorithm. The DCT is employed in a multitude of compression standards due to its remarkable energy compaction properties. Multiplier-free approximate DCT transforms have been proposed that offer superior compression performance at very low circuit complexity. Such approximations can be realized in digital VLSI hardware using additions and subtractions only, leading to significant reductions in chip area and power consumption compared to conventional DCTs and integer transforms.

Key words: *Approximate DCT, low-complexity algorithms, Algorithm-architecture co design, DCT-approximation, discrete cosine transform(DCT).*

I. INTRODUCTION

The Discrete Cosine Transform(DCT) is popularly used in image and video compression. Since the DCT is computationally intensive, several types algorithms have been proposed in the literature to compute it efficiently. Recently, significant work has been done to derive approximate of 8-point DCT for reducing the computational complexity .The main objective of the approximation algorithms is to get rid of multiplications which consumes most of the power and computation time ,and to obtain meaningful estimation of DCT as well. The need of approximation is more important for higher size DCT since the computational complexity of the DCT grows non linearly. On the other hand ,modern video coding standards such as a high efficiency video coding (HEVC) uses DCT of larger block sizes (up to 32 32) in order to achieve higher compression ratio.But,the extension of the design strategy used in H264AVC for larger transform sizes,such as 16-point and 32-point is not possible[11].Besides, several image processing applications such as tracking and simultaneous compression and encryption[13] require higher DCT sizes.In this context,Cintra has introduced a new class of integer transforms applicable to several block lengths.Cintra et al.have proposed a new 16 matrix also for approximation of 16-point DCT,and have validated it experimentally. Recently ,two new transforms have been proposed for 8-point DCT approximation: Cintra et al.have proposed a low

complexity 8-point approximate DCT based on integer functions and Potluri et al.have proposed a novel 8-point DCT approximation that requires only 14 additions. On the otherhand ,Bouguezel et al.have proposed two methods for multiplication free approximate form of DCT.The first method is for length 16 and 32; and is based on the approximate extension of integer DCT[18].Also, a systematic method for developing a binary version of high size DCT(BDCT) by using the Sequency-Ordered Walsh-Hadamard transform (SO-WHT) is proposed in[4] .This transform is a permutated version of the WHT which approximate the DCT very well and maintains all the advantages 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 in order to provide compression and
iii) Performance close to the exact DCT preferably should be orthogonal.

But the existing DCT algorithms do not provide the best of all the above three requirements. Some of the existing methods are deficient in terms of scalability[18],generalization for higher sizes[15],and orthogonality.We intend to maintain orthogonality in the approximate DCT for two reasons.Firstly,if the transform is orthogonal,we can always find its inverse and the kernel matrix of the inverse transform is obtained by just transposing the kernel matrix of the forward transform.This feature of inverse transform could be used to compute the forward and inverse DCT by similar computing and that stryctures.Moreover in case of that orthogonal transforms,similar fast algorithms are applicable to both forward and inverse transforms.

In this context,the discrete cosine transform (DCT) is an essential mathematical tool in both image and video coding.Indeed the DCT was demonstrated to provide good energy compaction for natural images,which can be described by first-order Markov signals.Two dimensional (2-D) version of the 8-point DCT.

It was adopted in several imaging standards such as additionally new compression scheme such as the High Efficiency Video Coding(HEVC) employs DCT like integer transforms operating at various block sizes ranging from 4x4 to 32x32 pixels. The distinctive characterstic of HEVC is its capability of achieving high compression performance at approximately half the bit rate required by H264/AVC with the same image quality.Also HEVC was demonstrated to be especially effective for high-resolution video applications.Therefore,low complexity DCT like approximations may benefit future video codecs including emerging HEVC/H.265 systems.

Several efficient algorithms were developed and a noticeable literature is available.Although fast algorithms can significantly reduce the computational complexity of computing the DCT,floating point operations are still required. Despite their accuracy,floating point operations are expensive in terms of circuitry complexity and power consumption.Therefore,minimizing the number of floating point operationsis a sought property in a fast algorithm .One way of circumventing this issue is by means of approximate transforms.

II. EXISTING METHOD

Using its separable property, an (N × N)-point 2-D integer DCT could be computed by the row-column decomposition technique in two distinct stages.

1) STAGE-1: N -point 1-D integer DCT is computed for each column of the input matrix of size (N × N) to generate an intermediate output matrix of size (N × N).

2) STAGE-2: N -point 1-D DCT is computed for each row of the intermediate output matrix of size (N × N) to generate desired 2-D DCT of size (N × N).

1.1 FOLDED STRUCTURE FOR 2-D INTEGER DCT

The folded structure for the computation of (N × N)-point 2-D integer DCT is shown in Fig. 5(a). It consists of one N - point 1-D DCT module and a transposition buffer. The structure of the proposed 4 × 4 transposition buffer is shown in Fig. 5(b). It consists of 16 registers arranged in four rows and four columns. (N × 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 MUXes

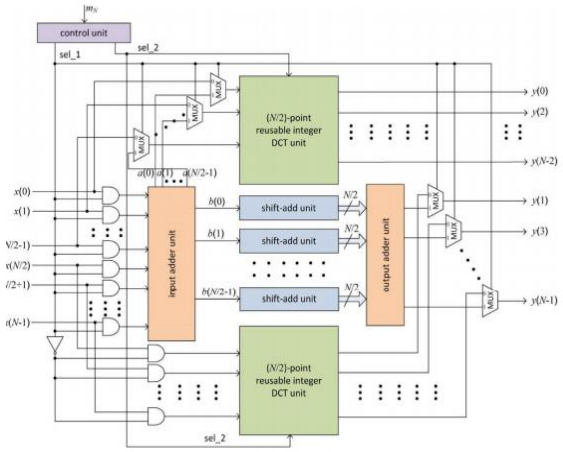

Fig. 1 Reusable architecture of integer DCT.

During the first N successive cycles, the DCT module receives the successive columns of (N ×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 MUXes and fed as input to the 1-D DCT module. N MUXes are used at the input of the 1-D DCT module to select either the columns from the input buffer (during the first N cycles) or the rows from the transposition buffer (during the next N cycles).

1.2. FULL-PARALLEL STRUCTURE FOR 2-D INTEGER DCT

The full-parallel structure for (N × N)-point 2-D integer DCT is shown in Fig. 6(a). It consists of two N -point 1-D DCT modules and a transposition buffer. The structure of the 4 × 4 transposition buffer for full-parallel structure is shown in Fig. 6(b). It consists of 16 register cells (RC) [shown in Fig. 6(c)] arranged in four rows and four columns. N × N transposition buffer can store N values in a cycle either row wise or column-wise by selecting the inputs by the MUXes at the input of RCs. The output from RCs can also be collected either row-wise or column-wise. To read the output from the buffer, N number of (2N − 1):1 MUXes [shown in Fig. 6(d)] are used, where outputs of the ith row and the ith column of RCs are fed as input to the ith MUX. For the first N successive cycles, the ith MUX provides output of N successive RCs on the ith row. In the next N successive cycles, the ith MUX provides output of N successive RCs on the ith column. By this arrangement, in the first N cycles, we can read the output of N successive columns of RCs and in the next N cycles, we can read the output of N successive rows of RCs. The transposition buffer in this case allows both read and write operations concurrently. If for the N cycles, results are read and stored column-wise now, then in the next N successive cycles, results are read and stored in the transposition buffer row-wise. The first 1-D DCT module receives the inputs column-wise from the input buffer. It computes a column of intermediate output and stores in the transposition buffer. The second 1-D DCT module receives the rows of the intermediate result from the transposition buffer and computes the rows of 2-D DCT output row-wise

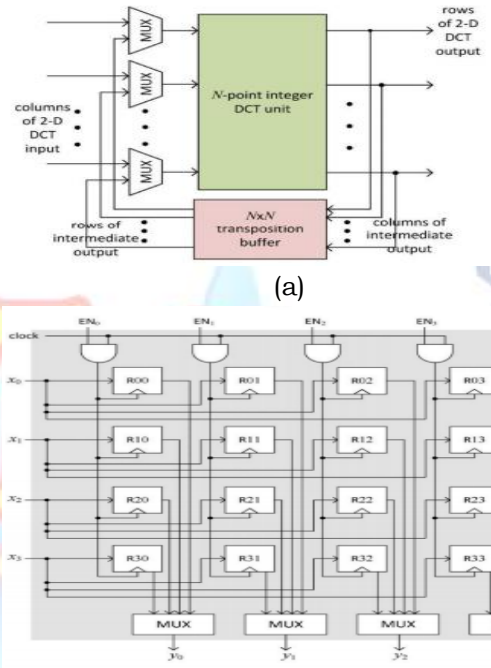

Fig. 2. Folded structure of (N × N)-point 2-D integer DCT. (a) Folded 2-D DCT architecture. (b) Structure of the transposition buffer for input size 4× 4.

The basic computational block of algorithm for the proposed DCT approximation, is given in [6]. The block diagram of the computation of DCT based on is shown in Fig. 1. For a given input sequence the approximate DCT coefficients are obtained by . An example of the block diagram of is illustrated in Fig. 2, where two units for the computation of are used along with an input adder unit and output permutation unit. 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 extended to obtain the DCT of higher sizes. For example, the structure for the computation of 32-point DCT could be obtained by combining a pair of 16-point DCTs with an input adder block and output permutation block

1.3 Complexity Comparison

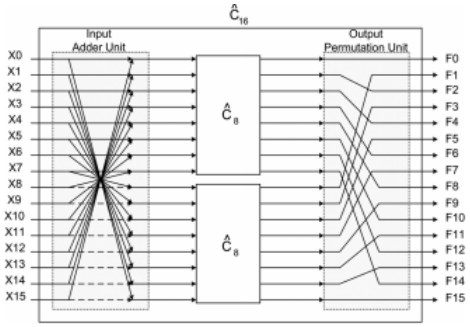

Fig. 3. Block diagram of the proposed DCT for N=16.

N	Method	Arithmetic operations	
		Add	Shift
8	Proposed=BC [6]	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

TABLE I ASSESSMENT OF REQUIRED ARITHMETIC OPERATIONS FOR SEVERAL APPROXIMATION ALGORITH

and 64-point DCT approximations 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. Moreover, since the structures for the computation of DCT of different lengths are regular and scalable, the computational time for N DCT coefficients can be found to be $\log_2(N)Ta$ where is the addition-time. The number of arithmetic operations involved in proposed DCT approximation of different lengths and those of the existing competing approximations are shown in Table I. It can be found that the proposed method requires the lowest number of additions, and does not require any shift operations. Note that shift operation does not involve any combinational components, and requires only rewiring during hardware implementation. But it has indirect contribution to the hardware complexity since shift-add operations lead to increase in bit-width which leads to higher hardware complexity of arithmetic units which follow the shift-add operation. Also, we note that all considered approximation methods involve significantly less computational complexity over that of the exact DCT algorithms. According to the Loeffler algorithm [2], the exact DCT computation requires 29, 81, 209, and 513 additions along with 11, 31, 79, and 191 multiplications, respectively for 8, 16, 32, and 64-point DCTs.

Pipelined and non-pipelined designs of different methods are developed, synthesized and validated using an integrated logic analyzer. The validation is carried out by using the Digilent EB of Spartan6-LX45. We have used 8-bit inputs, and we have allowed the increase of output size (without any truncations). For the 8-point transform of Fig. 1, we have 11-bit and 10-bit outputs. The pipelined design are obtained by insertion of registers in the input and output stages along with registers after each adder stage, 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 provide significantly higher maximum operating frequency (MOF). It also shows that the proposed design involves nearly 7%, 6%, and 5% less area compared to the BDCT design for equal to 16, 32, and 64, respectively. Note that both pipelined and non-pipelined designs involve the same number of LUTs since pipeline registers do not require additional LUTs. For 8-point DCT, we have used the approximation proposed in [6] which forms the basic computing block of the proposed method. Also, we underline that all designs have the same critical path; and accordingly have the same MOFs. Most importantly, the proposed designs are reusable for different transform lengths.

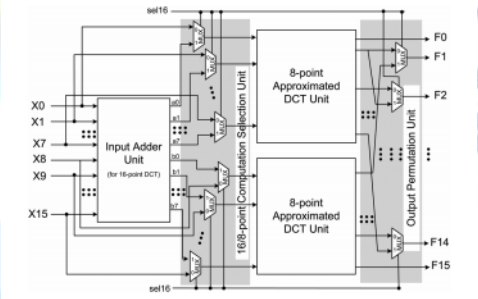

Fig. 4. Proposed reconfigurable architecture for approximate DCT of lengths n=8 and 16.

III. PROPOSED MODEL

DCT of different lengths such as 16, 32 are required to be used in video coding applications. Therefore, a given DCT architecture should be potentially reused for the DCT of different lengths instead of using separate structures for different lengths. We propose here such reconfigurable DCT structures which could be reused for the computation of DCT of 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 approximated DCT units and a 16-point

input adder unit that generates a(i) and b(i) ,i=[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(2)……a(7)]or[x(0),x(1),x(2)……..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(2)……b(7)] or [x(8),x(9),x(10)……..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.SEL16 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, SEl16=1 enables the computation of 16-point DCT and SEL16=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.

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 SEl32=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 the length of the DCT to be computed is higher than 8 (DCT length of 16 or 32), 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 Sel16 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. Note that the throughput is of 32 DCT coefficients per cycle irrespective of the desired transform size.

2.1 8 POINT DCT DESIGN:

We review the mathematical description of the selected 8-point DCT approximations. All discussed methods here consist of a transformation matrix that can be put in the following format:

We aim at deriving a novel low-complexity approximate DCT. For each end, we propose a search over the 8 8 matrix space in order to find candidate matrices that possess low computation cost. Let us define the cost of a transformation matrix as the number of arithmetic operations required for its computation. One way to guarantee good candidates is to restrict the search to matrices whose entries do not require multiplication operations. Thus we have the following optimization problem:

We propose digital computer architectures that are custom designed for the real time implementation of the fast algorithms described in section II. The proposed architecture employs two parallel realizations of DCT approximation blocks as shown in Fig.1. The 1-D approximate DCT blocks implement a particular fast algorithm chosen from the collection described earlier in the paper. The row and column-wise transforms can be any of the DCT approximations detailed in the paper. In other words, there is no restriction for both row and column-wise transforms to be the same. However, for simplicity, we adopted identical transforms for both steps between the approximate DCT blocks a real-time row-parellel transposition buffer circuit is required. Such block ensures data ordering for converting the row-transformed data from the first DCT approximation circuit to a transposed format as required by the column transform circuit. The transposition buffer block is detailed in Fig.2.

The digital architecture of the discussed approximate DCT algorithms were given hardware signal flow diagrams as listed below:

1) proposed novel algorithm and architecture shown in Fig.3(a);
2) BAS-2008 architecture shown in Fig.3(b);
3) BAS-2011 architecture shown in Fig.3(c);
4) CB-2011 architecture shown in Fig.3(d);
5) Modified CB-2011 architecture shown in Fig.3(e);
6) Architecture for the algorithm in [40] shown in Fig.3(f)

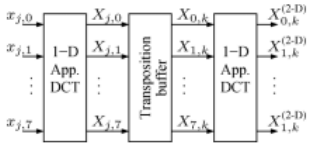

Fig. 5.Two-dimensional approximate transform by means of 1-D approximate transform.

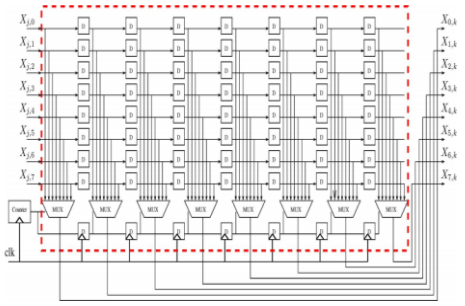

Fig. 6. Details of the transposition buffer block.

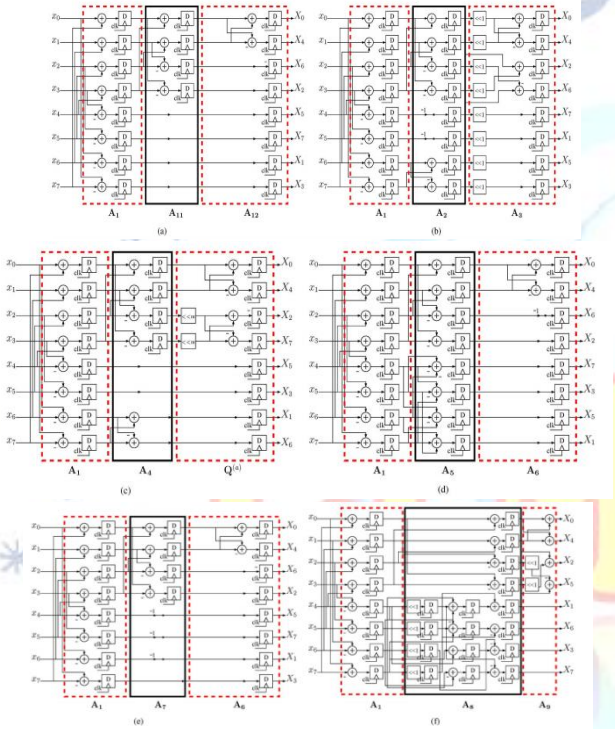

Fig. 7.Digital architecture for considered DCTapproximations. (a) Proposed approximate transform , (b) BAS-2008 approximate DCT , (c) BAS-2011 approximate DCT where , (d) CB-2011 approximate DCT , (e) Modified CB-2011 approximate DCT , (f) Approximate DCT in [40] .

IV. Output wave forms

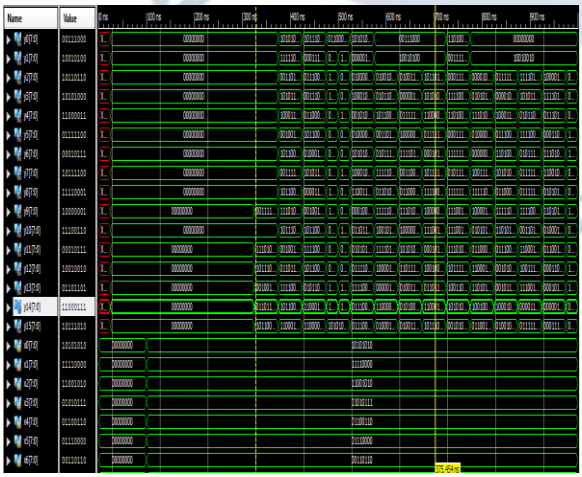

Fig 8:output waveform

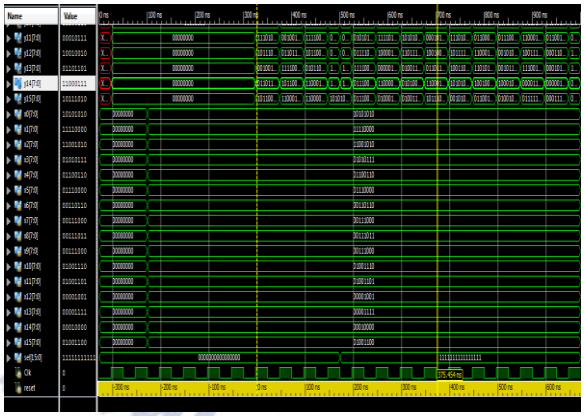

Fig 9:output waveform

As we say that we are performing 16 point reconfigurable using no multiplications ,it depends on sel line if sel goes low 8 point DCT is performed or else 16 point DCT is performed.

V. Conclusion

We have proposed a recursive algorithm to obtain orthogonal approximation of DCT where approximate DCT of length 16could be derived from a pair of DCTs length 8 at the cost of additions 16 for input preprocessing .The proposed approximated DCT has several advantages,such as of regularity,structural simplicity,lower computational complexity and scalability.Comparison with recently proposed computing methods shows the effectiveness of the proposed approximation in terms of error energy, hardware resources consumption,and compressed image quality.

References

[1] A.M Shams ,A.Chidanandan,W.pan and M.A.Boyoumi,"NEDA: A low power high-performance,DCT-architecture,"IEEE trans.Signal-process.vol.54,no.3,pp.955964,2006.

[2] C.loeffler,A.lihttenberg,and,G.S.Moschytz,"practica l fast 1-DDCT algorithm with 11 multiplications,"inproc, int,conf. acoust. speech signal process(ICASSP),may 1989,PP.988-991.

[3] M.jridi,P.K.Meher,andA.alfalou,"zero-quantiseddiscr eate cosine transform coeffients prediction technique for intra-frame video encoding,"IET image process,vol 7,no 2,pp.165-173,Mar 2013.

[4] S.Bouguezel,M.O.Ahmead and M.N.S.Swamy,"Binary descretecosinr and hatrtly transformations, IEEE trans circuits SystI.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 4 additions only,"electronlet.vol 48,no 5,pp.919-921,jul-2012.

[6] R.j.Cintra and F.M.Bayer ," A DCT approximation for image compression,"IEEE signal process,let,vol.18,no.10,pp,579-582,oct-2011.

[7] Narasimha,M:Peterson,A(june 1978),"on the computation of the discreate cosine transform ",IEEE transactions on Comminicatins.26(6):934-936.doi;0.109/TCOM.1978.1094144.

[8] Makhoul,j(febriary 1980). "A fast cosine transform in one and two dimensions ",IEEE transactions on acoustics sppech and signal processing.28(1):27-34.DOI:10.1109/TASSP.19801163351.

[9] Sorensen,H:jones ,D.heideman,M:Burrus,C.(june 1987),"real-values fast fourier transform algorithms",IEEE transactions on acoustics,Speech and signal processing,35(6):849-863,doi:10.1109/TASSP.1987.1165220.

[10] Arai,Y:Agui,T.Nakaajima,M.(November 1988),"A fast DCT-SQ scheme for images",IEICE transactions .71(11):1095-1097

[11] Plonka,G.tasche,M.(January 2005)."fast and numerically stable alogotrithms for discrete cosine transforms".Linear algebra and its applications.394(1):309-345.doi:10.1016/j.laa.2004-07-015.

[12] DuVetteril,M(april 1990)."fast fouriortransforms:A tutorial review and state of the art",'Signal processing .19(4):259-299.doi:10.1016/0615-1684(90090158-U.