International Journal of VLSI System Design and Communication Systems

ISSN 2322-0929
Vol.05, Issue.06,
June-2017,
Pages:0482-0488

www.ijvdcs.org

A Novel Approach to Implement Scalable DCT Architecture for Timing Critical Applications

MUDIDANA VENKATESH[1], V. KEERTHI KIRAN[2]
[1]PG Scholar, Baba Institute of Technology and Science, Bakkannapalem,Madhurawada, Visakhapatnam, AP, India,
E-mail: m.venkatesh255@gmail.com.
[2]Assistant Professor, Baba Institute of Technology and Science, Bakkannapalem,Madhurawada, Visakhapatnam, AP, India,
E-mail: keerthikiran123@gmail.com.

Abstract: Approximation of discrete cosine transform (DCT) is useful for reducing its computational complexity without significant impact on its coding performance. 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. Proposed algorithm is highly scalable for hardware as well as software implementation of DCT of higher lengths, and it can make use of the existing approximation of 8-point DCT to obtain approximate DCT of any power of two length, .We demonstrate that the proposed approximation of DCT provides comparable or better image and video compression performance than the existing approximation methods. It is shown that proposed algorithm involves lower arithmetic complexity compared with the other existing approximation algorithms. We have presented a fully scalable reconfigurable parallel architecture for the computation of approximate DCT based on the proposed algorithm. One uniquely interesting feature of the proposed design is that it could be configured for the computation of a 32-point DCT or for parallel computation of two 16-point DCTs or four 8-point DCTs with a marginal control overhead. The proposed architecture is found to offer many advantages in terms of hardware complexity, regularity and modularity.

Keywords: DCT, Orthogonal, XILINX, Verilog.

I. INTRODUCTION

The discrete cosine transform (DCT) is popularly used in image and video compression. Since the DCT is computationally intensive, several 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 consume most of the power and computation time, and to obtain meaningful estimation of DCT as well. Haweel has proposed the signed DCT (SDCT) for 8 X8 blocks where the basis vector elements are replaced by their sign, i.e, 1. Bouguezel-Ahmad-Swamy (BAS) have proposed a series of methods. They have provided a good estimation of the DCT by replacing the basis vector elements by 0, 1/2, 1. In the same vein, Bayer and Cintra have proposed two transforms derived from 0 and 1 as elements of transform kernel, and have shown that their methods perform better than the method, particularly for low- and high-compression ratio scenarios. The need of approximation is more important for higher-size DCT since the computational complexity of the DCT grows nonlinearly. On the other hand, modern video coding standards such as high efficiency video coding (HEVC) [10] uses DCT of larger block sizes (up to 32X 32) in order to achieve higher compression ratio. But, the extension of the design strategy used in H264 AVC for larger transform sizes, such as 16-point and 32-point is not possible. Besides, several image processing applications such as tracking and simultaneous compression and encryption require higher DCT sizes.

In this context, Cintra has introduced a new class of integer transforms applicable to several block-lengths. Cintra have proposed a new 16 X16 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 addition . On the other hand, Bouguezel 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 appropriate extension of integer DCT. 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. This transform is a permutated version of the WHT which approximates the

DCT very well and maintains all the advantages of the WHT. A scheme of approximation of DCT should have the following features:

1. It should have low computational complexity.
2. It should have low error energy in order to provide compression performance close to the exact DCT, and preferably should be orthogonal.
3. It should work for higher lengths of DCT to support modern video coding standards, and other applications like tracking, surveillance, and simultaneous compression and encryption.

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, generalization for higher sizes, and orthogonality. We intend tomaintain 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 structures.

II. DISCRETE COSINE TRANSFORM

The DFT is not the only transform that is widely used in applications Published standards for image and video coding (compression) make use of the DCT

1. JPEG (1989)
2. MPEG1, 2, and 4
   - MPEG1(1992):video CD players, storage and retrieval of moving pictures and audio on storage media
   - MPEG2 (1994): HDTV, DVD, standard for Digital TV (cable)
   - MPEG3, originally targeted for HDTV, was incorporated into MPEG2
   - MPEG4 (late 1998): standard for multimedia applications, targeted for wireless video
3. H.261, H.263
   - H.261 (circa 1993): video conferencing
   - H.263 (circa 1995): wireless video

These standards provide instructions for decoding the signal, but there is often considerable freedom for encoding the signal.For more information go to http://drogo.cselt.it/mpeg/.

A. Compression

Two classes of compression algorithms try to reduce the number of bits required to represent a signal.

1. lossless: compression ratios around 2-3:1 for data files
2. lossy: compression ratios up to 1000:1 for video

For wireless video, need compression ratios up to 1000:1. Can get near lossless video compression at 8:1 with little degradation. Compression algorithms work by removing redundancy in the signal. In video signals, the redundancy can be of three forms.

- statistical: (e.g. Huffman codes, arithmetic, Lempel-Ziv)
- spatial: (e.g. vector quantization, DCT, subband coders, wavelets)
- temporal: (e.g. motion compensation)

Wavelet compression is used in JPEG-2000, MPEG4, and H.263+ Transform coders decompose a frame into blocks, typically 8 x 8. In MPEG2, they are called macroblocks and divide the frame into luminance (intensity) and chrominance (color) images (YUV).

- luminance image: one 16 x 16 macroblock or four 8 x 8 macroblocks (Y)
- chrominance image: two 8 x 8 blocks (UV)

A 2-D DCT of each block is computed and the transform coefficients are quantized. Quantized coefficients are coded losslessly. The choice of quantization affects the transmission rate and distortion.

Advantages of the DCT (relative to the DFT)

- real-valued
- better energy compaction (much of the signal energy can be represented by only a few coefficients)
- coefficients are nearly uncorrelated
- experimentally observed to work well

B. 2-D DCT

$$X_c[k_1,k_2]=\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}4x[n_1,n_2]\cos\left(\frac{\pi(2n_1+1)k_1}{2N_1}\right)\cos\left(\frac{\pi(2n_2+1)k_2}{2N_2}\right)$$

$$x[n_1,n_2]=\frac{1}{N_1N_2}\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1}c[k_1]c[k_2]X_c[k_1,k_2]\cos\left(\frac{\pi(2n_1+1)k_1}{2N_1}\right)\cos\left(\frac{\pi(2n_2+1)k_2}{2N_2}\right)$$

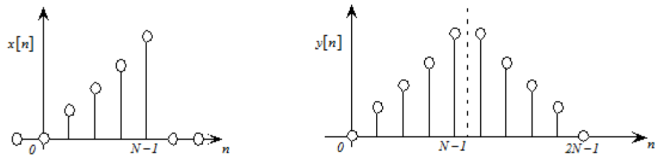
$$c[k]=\begin{cases}1/2, & k=0\\ 1, & k\neq 0\end{cases}$$

1. The DFT is related to the Fourier Series coefficients of a periodically extended sequence.
2. The DCT is related to the Fourier Series coefficients of a symmetrically extended sequence.
3. The 2-D DCT is a separable transform. It can be evaluated using a row-column decomposition. For a 8 x8 DCT, we need 16 1-D DCTs. Calling 16 functions may lead to unacceptable overhead for 8-point DCTs

C. 1-D DCT

$$X_c[k]=\sum_{n=0}^{N-1}2x[n]\cos\left(\frac{\pi(2n+1)k}{2N}\right)$$

Define the symmetric extension of $x[n]$ as $y[n]=x[n]+x[2N-1-n]$ for $n=0,1,...,N-1.$



Now consider the 2N-point DFT of $y[n]$.

$$Y[k] = \sum_{n=0}^{2N-1} y[n] e^{-j\frac{2\pi}{2N}kn}$$
$$= \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{2N}kn} + \sum_{n=N}^{2N-1} x[2N-1-n] e^{-j\frac{2\pi}{2N}kn}$$
$$= \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{2N}kn} + \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{2N}k(2N-1-n)}$$
$$= \sum_{n=0}^{N-1} x[n] \left\{ e^{-j\frac{2\pi}{2N}kn} + e^{j\frac{2\pi}{2N}kn} e^{j\frac{2\pi}{2N}k} \right\}$$
$$= e^{j\frac{\pi}{2N}k} \cdot \sum_{n=0}^{N-1} x[n] \left\{ e^{-j\frac{\pi}{2N}k} e^{-j\frac{2\pi}{2N}kn} + e^{j\frac{2\pi}{2N}kn} e^{j\frac{\pi}{2N}k} \right\}$$
$$= e^{j\frac{\pi}{2N}k} \cdot \sum_{n=0}^{N-1} 2x[n] \cos\left(\frac{\pi k}{2N}(2n+1)\right)$$

D. ALGORITHM #1 FOR 1-D DCT

1. Set $y[n] = x[n] + x[2N-1-n]$
2. Calculate $Y[k]$ using a 2N-point DFT
3. Set $X_c[k] = exp\left(-j\frac{\pi}{2N}k\right)Y[k]$, for $k = 0, 1, \ldots, N-1$.

This requires N+N $\log_2$(2N) complex multiplies. Another algorithm can be developed that requires fewer multiplies by using a shorter DFT.

E. REVIEW OF 1-D DECIMATION-IN-TIME (DIT) FFT

Consider the 1-D DFT

$$Y[k] = \sum_{n=0}^{2N-1} y[n] W_{2N}^{nk}$$

Divide the sum into two components, one over the even samples and one over the odd samples.

$$Y[k] = \sum_{r=0}^{N-1} y[2r] W_{2N}^{2rk} + \sum_{r=0}^{N-1} y[2r+1] W_{2N}^{(2r+1)k}, \quad k = 0,1,\ldots,2N-1$$
$$= \sum_{r=0}^{N-1} y[2r] W_N^{rk} + W_{2N}^{k} \sum_{r=0}^{N-1} y[2r+1] W_N^{rk}, \quad k = 0,1,\ldots,2N-1$$
$$= G[k] + W_{2N}^{k} H[k]$$

Note that

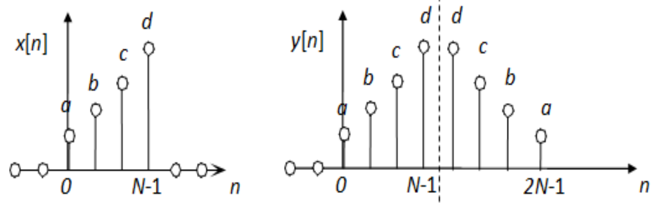
$$W_{2N}^{2rk} = \exp(-2\pi\frac{2rk}{2N}) = \exp(-2\pi\frac{rk}{N}) = W_N^{rk}$$

G[k] and H[k] are N-point DFTs:

Set

$$g[n] = y[2n], \quad n = 0, 1, \ldots, N-1$$

$$h[n] = y[2n+1], \quad n = 0, 1, \ldots, N-1$$



$$g[n] = \{a, c, d, b\}$$

$$h[n] = \{b, d, c, a\} = g[N-1-n]$$
$$\Rightarrow H[k] = W_N^{-k} G[((N-k))_N] = W_N^{-k} G^*[k]$$

Therefore,

$$Y[k] = G[k] + e^{j\frac{\pi k}{2N}} e^{-j\frac{\pi k}{N}} G^*[k], \quad k = 0, 1, \ldots, 2N-1$$
$$Y[k] = G[k] + e^{-j\frac{\pi k}{2N}} G^*[k], \quad k = 0, 1, \ldots, 2N-1$$
$$X_c[k] = \text{Re}\left(e^{-j\frac{\pi k}{2N}}\left(G[k] + e^{-j\frac{\pi k}{2N}} G^*[k]\right)\right), \quad k = 0, 1, \ldots, 2N-1$$
$$= \text{Re}\left(e^{-j\frac{\pi k}{2N}} G[k] + e^{-j\frac{2\pi k}{2N}} G^*[k]\right)$$
$$= \text{Re}\left(e^{-j\frac{3\pi k/2}{2N}}\left(e^{j\frac{\pi k/2}{2N}} G[k] + e^{-j\frac{\pi k/2}{2N}} G^*[k]\right)\right)$$
$$X_c[k] = 2\cos\left(\frac{3\pi k}{4N}\right)\cos\left(\frac{\pi k}{4N}\right)\text{Re}\left(G[k]\right), \quad k = 0,1,\ldots, N-1$$

F. Algorithm #2 for 1-D DCT

1. Set
$$g[n] = x[2n], \quad n = 0,1,\ldots, \frac{N}{2} - 1$$
2. $$g[N-1-n] = x[2n+1], \quad n = 0,1,\ldots, \frac{N}{2} - 1$$
3. Calculate the N-point DFT of $g[n]$
4. Set
5. $$X_c[k] = 2\cos\left(\frac{3\pi k}{4N}\right)\cos\left(\frac{\pi k}{4N}\right)\text{Re}\left(G[k]\right)$$

The coefficients in front of G_{real}[k] can be pre-computed for different values of k for a given N.

III. DCT APPROXIMATION

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

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

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 distinguish it from approximated forms of DCT. For $k \in [0, (N/2) - 1]$ and $i = 2k$, for any even value of N we can

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

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

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

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

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

Where T_N is a block sparse matrix expressed by:

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

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

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

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

$$P^{(i)}_{N-1,\frac{N}{2}} = \begin{cases} 0_{1,\frac{N}{2}} & if \quad i = 1,3,5,\ldots,N-1 \\ I_{\frac{N}{2}}\left(\frac{i}{2}\right) & if\ i = 0,2,4,\ldots,N-2 \end{cases} \qquad (7)$$

Where $I_{N/2}(i/2)$ is the $(i/2)$ th row vector of the $((N/2) \times (N/2))$ identity matrix. Finally, the last matrix in (4), M_N^{add} is defined by:

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

Where $J_{N/2}$ is an $((N/2) \times (N/2))$ 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 little to the total arithmetic complexity and cannot be reduced further. Therefore, for reducing the computational complexity of -point DCT, we need 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 sub matrices we take the smallest size of DCT matrix to terminate the approximation procedure to 8, since 4-point DCT and 2-pointDCT can be implemented by adders only. Consequently, a good approximation of C_N , where N is an integral power of two, for $N \ge 8$, leads to a proper approximations of C_8 and S_8. For approximation of C_8 we can choose the 8-point DCT since that presents the best trade-off between the number of required arithmetic operators and quality of the reconstructed image. The trade-off analysis shows that approximating C_8 by $\hat{C}_8 = \lfloor 2C_8 \rceil$ by where denotes the rounding-off operation outperforms the current state-of-the-art of 8-point approximation methods.

When we closely look at (4) and (5), we note that C_8 operates on sums of pixel pairs while S_8 operates on differences of the same pixel pairs. Therefore, if we replace $\hat{S}_8$ by $\hat{C}_8$, we shall have two main advantages. Firstly, we shall have 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 we have investigated two other low-complexity alternatives, and in the following we discuss here three possible options of approximation of S_8:

1. The first one is to approximate S_8 by null matrix, which implies all even-indexed DCT coefficients are assumed to be zero. The transform obtained by this approximation is far from the exact values of even-indexed DCT coefficients, and the odd coefficients do not contain any information.
2. The second solution is obtained by approximating S_8 by an 8X8 matrix where each row contains one 1 and all other elements are zeros. Here, elements equal to 1 correspond to the maximum of elements of the exact DCT in each row. The approximate transform in this case is closer to the exact DCT than the solution obtained by null matrix.
3. iii) The third solution consists of approximation of S_8 by $\hat{C}_8$ Since C_8 as well as S_8 are sub matrices of C_{16} and operate on matrices generated by sum and differences of pixel pairs at distance of 8, approximation of S_8 by $\hat{C}_8$ has attractive computational properties: regularity of the signal-flow graph, orthogonality since $\hat{C}_8$ isorthogonalizable,and good compression efficiency, other than scalability and scope for reconfigurable implementation.

We have not done exhaustive search of all possible solutions. So there could be other possible low-complexity implementation of $\hat{S}_8$. But other solutions are not expected to have the potential for reconfigurablity what we achieve by replacement of by $\hat{S}_8$. Based on this third possible approximation of , we have obtained the proposed approximation of $\hat{C}_N$ as C_N:

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

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

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

where $(\cdot)^t$ denotes matrix transposition. For data compression, we can use $\boldsymbol{C}_N^{orth} = \boldsymbol{D}_N \times \hat{\boldsymbol{C}}_N$ instead of $\hat{\boldsymbol{C}}_N$ since $(\boldsymbol{C}_N^{orth})^{-1} = (\boldsymbol{C}_N^{orth})^t$. Since is a diagonal matrix, it can be integrated into the scaling in the quantization process (without additional computational complexity). Therefore, as adopted, the computational cost of $\boldsymbol{C}_N^{orth}$ is equal to that of $\hat{C}_N$. Moreover, the term can be integrated in the quantization step in order to have multiplier less architecture. The procedure for the generation of the proposed orthogonal approximated DCT is stated in Algorithm 1.

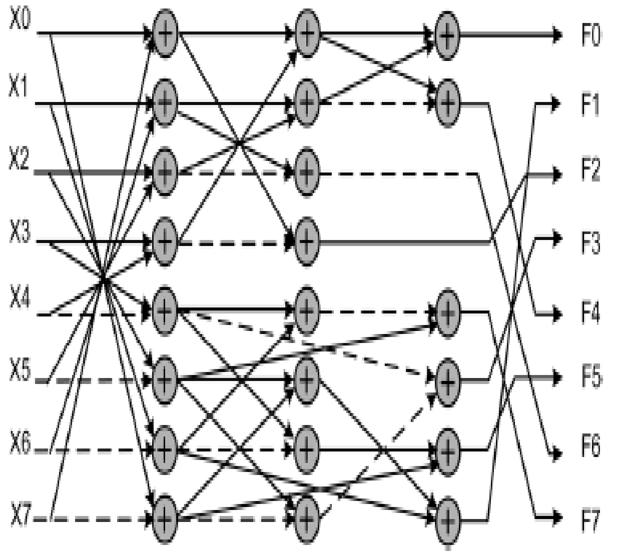


Fig 1. Signal flow graph (SFG) of $(\hat{\boldsymbol{C}}_8)$. Dashed arrows represent multiplications.
by 1.

Scalable and Reconfigurable Architecture for DCT Computation

In this section, we discuss the proposed scalable architecture for the computation of approximate DCT of $N = 16$and 32.We have derived the theoretical estimate of its hardware complexity and discuss the reconfiguration scheme.

A. Proposed Scalable Design

The basic computational block of algorithm for the proposed DCT approximation, $\hat{\boldsymbol{C}}_8$ is given. The block diagram of the computation of DCT based on $\hat{\boldsymbol{C}}_8$ is shown in Fig1.For a given input sequence $\{\boldsymbol{X}(n)\}, n \in [0, N-1]$, the approximate DCT coefficients are obtained by $\boldsymbol{F} = \hat{\boldsymbol{C}}_N \cdot \boldsymbol{X}^t$. An example of the block diagram of $\hat{\boldsymbol{C}}_{16}$ is illustrated in Fig. 2, where two units for the computation of $\hat{\boldsymbol{C}}_8$ 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.

B. Complexity Comparison

To assess the computational complexity of proposed N–point approximate DCT $N(\log_2 N - (1/4))$, we need to determine the computational cost of matrices quoted in (9). As shown in Fig. 1 the approximate 8-point DCT involves 22 additions. Since has $\boldsymbol{M}_N^{per}$ no computational cost and $\boldsymbol{M}_N^{add}$ requires additions for N –point DCT, the overall arithmetic complexity of 16-point, 32-point, and 64-point DCT approximations are 60, 152, and 368 additions, respectively. More generally, the arithmetic complexity of -point DCT is equal to 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 $\log_2(N)T_A$ where T_A 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, 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.

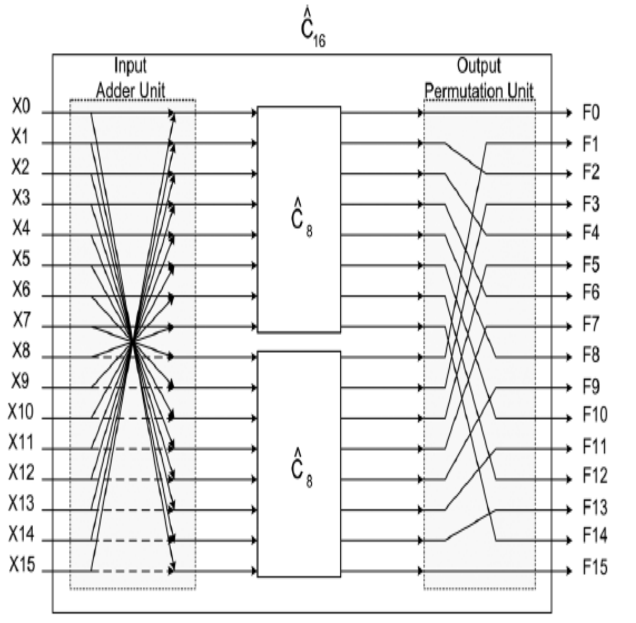


Fig2.Block diagram of the proposed DCT for $N = 16\ (\hat{\boldsymbol{C}}_{16})$.

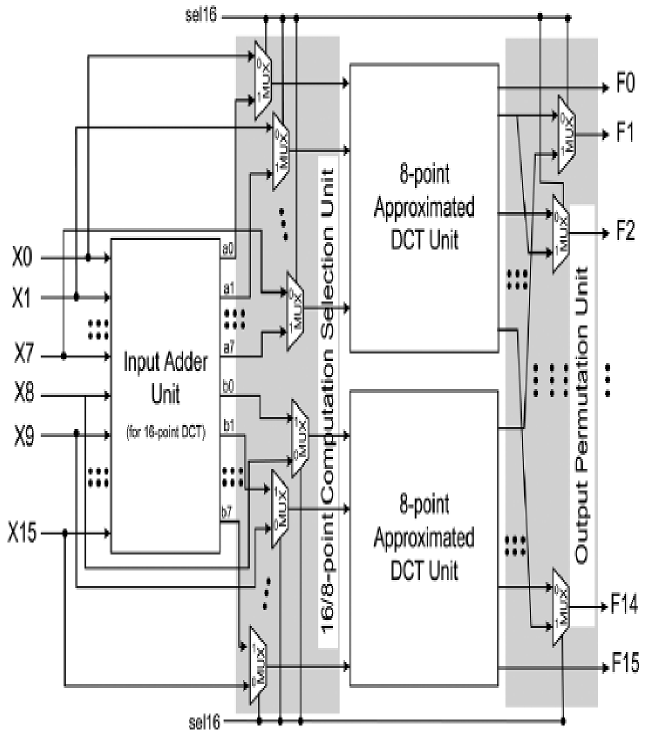

sel16
F0
F1
F2
F14
F15
X0
X1
X7
X8
X9
X15
Input Adder Unit (for 16-point DCT)
a0
a1
a7
b0
b1
b7
MUX
16/8-point Computation Selection Unit
8-point Approximated DCT Unit
8-point Approximated DCT Unit
Output Permutation Unit
sel16

Fig3.Proposed reconfigurable architecture for approximate DCT of lengths $N = 8$ and 16.

C. Proposed Reconfiguration Scheme

As specified in the recently adopted HEVC, DCT of different lengths such as $N = 8$., 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 $b(i),\ i \in [1 : 7]$, . The input to the first 8-point DCT approximation unit is fed through 8 MUXes that select either $[a(0), a(1), \ldots, a(7)]$ or $[X(0), X(1), \ldots, X(7)]$, 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 or , 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.

IV. RESULTS

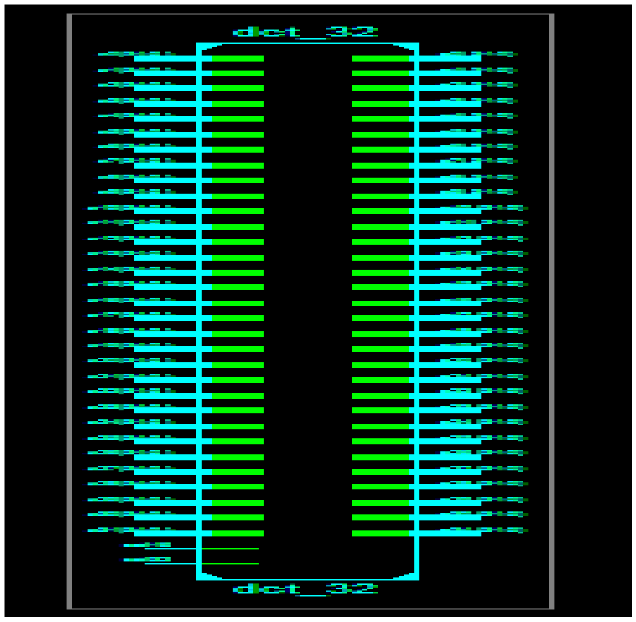


Fig4. RTL Schematic.

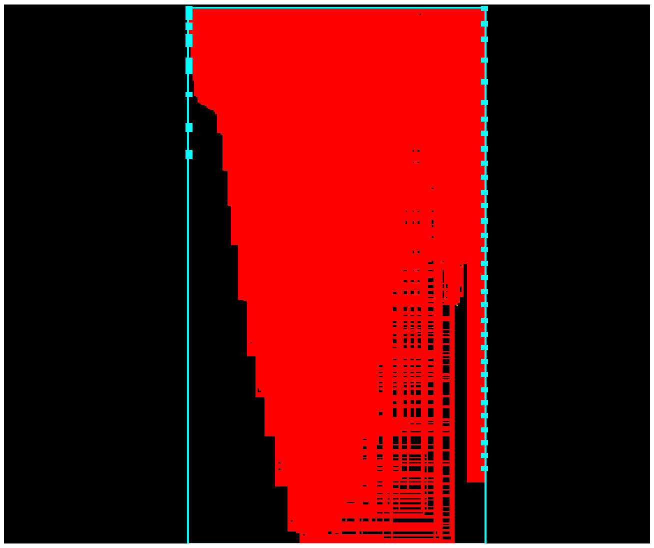
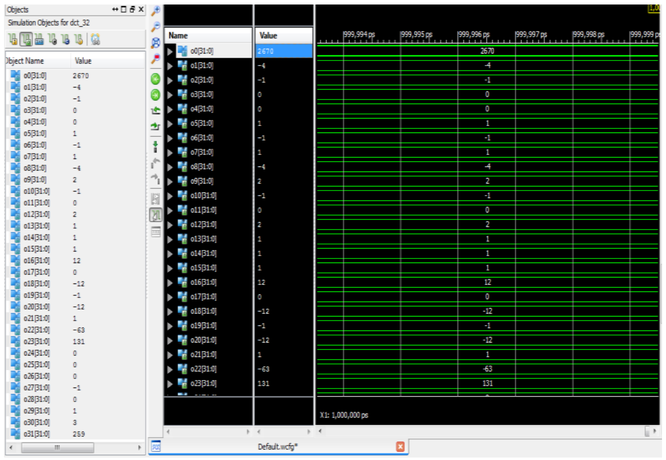


Fig5. Technological schematic



Objects

Simulation Objects for dct_32

Object Name	Value
o0[31:0]	2670
o1[31:0]	-4
o2[31:0]	-1
o3[31:0]	0
o4[31:0]	0
o5[31:0]	1
o6[31:0]	-1
o7[31:0]	1
o8[31:0]	-4
o9[31:0]	2
o10[31:0]	-1
o11[31:0]	0
o12[31:0]	2
o13[31:0]	1
o14[31:0]	1
o15[31:0]	1
o16[31:0]	12
o17[31:0]	0
o18[31:0]	-12
o19[31:0]	-1
o20[31:0]	-12
o21[31:0]	1
o22[31:0]	-63
o23[31:0]	131
o24[31:0]	0
o25[31:0]	0
o26[31:0]	0
o27[31:0]	-1
o28[31:0]	0
o29[31:0]	1
o30[31:0]	3
o31[31:0]	259

Name	Value
o0[31:0]	2670
o1[31:0]	-4
o2[31:0]	-1
o3[31:0]	0
o4[31:0]	0
o5[31:0]	1
o6[31:0]	-1
o7[31:0]	1
o8[31:0]	-4
o9[31:0]	2
o10[31:0]	-1
o11[31:0]	0
o12[31:0]	2
o13[31:0]	1
o14[31:0]	1
o15[31:0]	1
o16[31:0]	12
o17[31:0]	0
o18[31:0]	-12
o19[31:0]	-1
o20[31:0]	-12
o21[31:0]	1
o22[31:0]	-63
o23[31:0]	131

X1: 1,000,000 ps

Default.wcfg*

Fig6. Waveforms.

V. CONCLUSION

In this paper, we have proposed carry select adder logic for recursive algorithm to obtain orthogonal approximation of DCT where approximate DCT of length N could be derived from a pair of DCTs of length $(N/2)$ instead of N additions for input preprocessing. The proposed carry select logic for the approximated DCT has several advantages, such as of regularity, structural simplicity, lower-computational complexity, and scalability. Along with these we have another advantage, that is latency (delay) is reduced by 18.67% then the previous addition techniques. We have also proposed a fully scalable reconfigurable architecture for approximate DCT computation where the computation of 32-point DCT could be configured for parallel computation of two 16-point DCT's or four 8-point DCT's. This can also be extended to N point DCT by using $(N/2)$ point DCT, with reduced latency.

VI. 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-DDCT 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 8x 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

[22] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, 2004.