ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

Design of a JPEG Compressor with an Efficient and Reconfigurable DCT Algorithm

Nekkanti Lakshmi Indira Rani [1], G.Barathi subhashini [2]

P.G Student, Department of Electronics and Communication Engineering, Mallareddy Engineering College Hyderabad, Telangana, India

Associate Professor, Department of Electronics and Communication Engineering, Mallareddy Engineering College Hyderabad, Telangana, India

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 pre processing. 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 . 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. We implement the JPEG compressions scheme with the DCT proposed above.

KEYWORDS: Algorithm-architecture code sign, DCT approximation, discrete cosine transform (DCT), high efficiency video coding (HEVC).

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 [1]–[3]. Recently, significant work has been done to derive approximate of 8-point DCT for reducing the computational complexity [4]–[9]. 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 [8] has proposed the signed DCT (SDCT) for 8 8 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 [7]. In the same vein, Bayer and Cintra [5], [6] 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 in [7], particularly for low- and high-compression ratio scenarios.

II. RELATED WORK

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 32 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 [11]. Besides, several image processing applications such as tracking [12] and simultaneous compression and

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

encryption [13] require higher DCT sizes. In this context, Cintra has introduced a new class of integer transforms applicable to several block-lengths [14]. Cintra *et al.* have proposed a new 16 16 matrix also for approximation of 16-point DCT, and have validated it experimentally [15]. 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 [16] and Potluri *et al.* have proposed a novel 8-point DCT approximation that requires only 14 addition [17]. On the other hand, 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 appropriate 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].

III. PROPOSED DCT APPROXIMATION

The elements of -point DCT matrix are given by:

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

For any even value of N we can find that

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

Dct matrix CN can be calculated as

$$\boldsymbol{C}_N = \frac{1}{\sqrt{2}}\boldsymbol{M}_N^{per}\boldsymbol{T}_N\boldsymbol{M}_N^{add} \quad - \quad (3)$$

Where Tn can be block spare matrix expressed by

$$\boldsymbol{T}_N = \begin{bmatrix} \boldsymbol{C}_{\frac{N}{2}} & \boldsymbol{0}_{\frac{N}{2}} \\ \boldsymbol{0}_{\frac{N}{2}} & \boldsymbol{S}_{\frac{N}{2}} \end{bmatrix} \quad - \quad (4)$$

Permutation matrx is expressed by

$$\boldsymbol{M}_N^{per} = \begin{bmatrix} \boldsymbol{P}_{N-1,\frac{N}{2}} & \boldsymbol{0}_{1,\frac{N}{2}} \\ \boldsymbol{0}_{1,\frac{N}{2}} & \boldsymbol{P}_{N-1,\frac{N}{2}} \end{bmatrix} \quad - \quad (5)$$

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

Finally matrix addition is defined by

$$\boldsymbol{M}_N^{add} = \begin{bmatrix} \boldsymbol{I}_{\frac{N}{2}} & \boldsymbol{J}_{\frac{N}{2}} \\ \boldsymbol{I}_{\frac{N}{2}} & -\boldsymbol{J}_{\frac{N}{2}} \end{bmatrix}, \quad - \quad (6)$$

When we closely look at (3) and (4), we note that C8 operates on sums of pixel pairs while S8 operates on differences of the same pixel pairs. Therefore, if we replace S8 with C8 by , we shall have two main advantages. Firstly, we shall have good compression performance due to the efficiency of C8 and secondly the implementation will be much simpler, scalable and reconfigurable. For approximation of S8 we have investigated two other low-complexity alternatives, and in the following we discuss here three possible options of approximation of S8.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

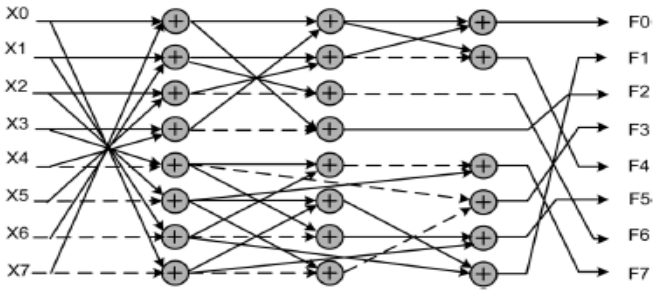

fig. 1. Signal flow graph (SFG) of ( ). Dashed arrows represent multiplications by 1.

We have not done exhaustive search of all possible solutions. So there could be other possible low-complexity implementation of S8. But other solutions are not expected to have the potential for reconfigurablity what we achieve by replacement of S8 by C8. Based on this third possible approximation of S8, we have obtained the proposed approximation as:

$$\hat{\boldsymbol{C}}_N = \frac{1}{\sqrt{2}} \boldsymbol{M}_N^{per} \begin{bmatrix} \hat{\boldsymbol{C}}_{\frac{N}{2}} & \boldsymbol{0}_{\frac{N}{2}} \\ \boldsymbol{0}_{\frac{N}{2}} & \hat{\boldsymbol{C}}_{\frac{N}{2}} \end{bmatrix} \boldsymbol{M}_N^{add}. \quad - (7)$$

As stated before, matrix is orthogonalizable. Indeed, for each we can calculate given by:

$$\boldsymbol{D}_N = \sqrt{\left(\hat{\boldsymbol{C}}_N \times \left(\hat{\boldsymbol{C}}_N\right)^t\right)^{-1}}, \quad - (8)$$

Where **(.)t** denotes matrix transposition. For data compression, we can use C_N^{orth} = DN x CN instead of CN since$(C_N^{orth})^{-1} = (C_N^{orth})^t$. since D_N is a diagonal matrix, it can be integrated into the scaling in the quantization process (without additional computational complexity). Therefore, as adopted in [4]–[8], the computational cost of C_N^{orth} is equal to that of . Moreover, the term of $1/\sqrt{2}$ in (9) can be integrated in the quantization step in order to have multipler less architecture. The procedure for the generation of the proposed orthogonal approximated DCT is stated in algorithm 1.

Algorithm 1 Generation of proposed DCT matrix

function PROPOSED DCT(N) ▷ N power of 2, $N \geq 8$
$N_0 \leftarrow \log_2(N/8)$ ▷ N_0 is the number of 8-sample blocks
$\boldsymbol{C}_{N/2^{N_0}} \leftarrow \lfloor 2\boldsymbol{C}_8 \rfloor$
while $N_0 > 0$ **do**
$\tilde{N} \leftarrow (N/2^{N_0-1})$
Calculate $\boldsymbol{M}_{\tilde{N}}^{per}, \boldsymbol{M}_{\tilde{N}}^{add}$
Calculate $\hat{\boldsymbol{C}}_{\tilde{N}}$
$N_0 \leftarrow N_0 - 1$
end while
Calculate $\boldsymbol{D}_N$
return $\hat{\boldsymbol{C}}_N, \boldsymbol{D}_N$
end function

III SCALABLE AND RECONFIGURABLE ARCHITECTURE FOR DCT COMPUTATION

The basic computational block of algorithm for the proposed DCT approximation $\mathbf{C_8}$, is given in [6]. The block diagram of the computation of DCT based $\mathbf{C_8}$ on is shown in Fig. 1. For a given input sequence the approximate DCT coefficients are obtained by $F = C^N . X^t$. An example of the block diagram of C^{16} 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

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

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.

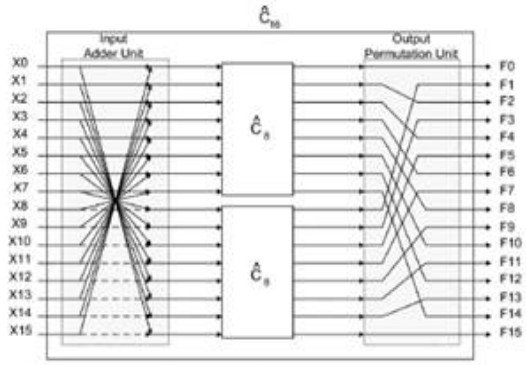

Fig. 2. Block diagram of the proposed DCT for $N = 16$ ($\hat{C}_{16}$).

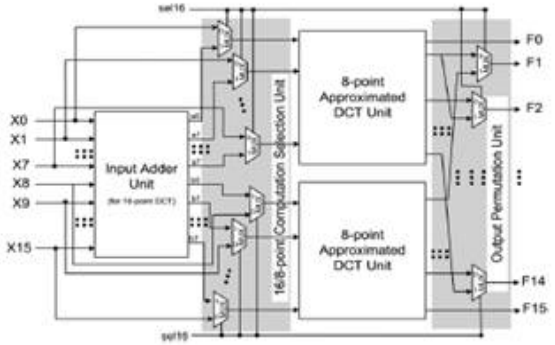

Fig. 3. Proposed reconfigurable architecture for approximate DCT of lengths $N = 8$ and 16.

to assess the computational complexity of proposed –point approximate DCT, 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 $\mathbf{M}_N^{per}$ has no computational cost and $\mathbf{M}_N^{add}$ require N 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 $N(\log_2 N - (1/4)$ 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 $\log_2 (N)T_A$ 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 [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 Diligent 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.

PROPOSED RECONFIGURABLE SCEME

As specified in the recently adopted HEVC [10], 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(7)] or [X(1), 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. 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, enables the computation of 16-point DCT and enables the computation of a pair of

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

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.

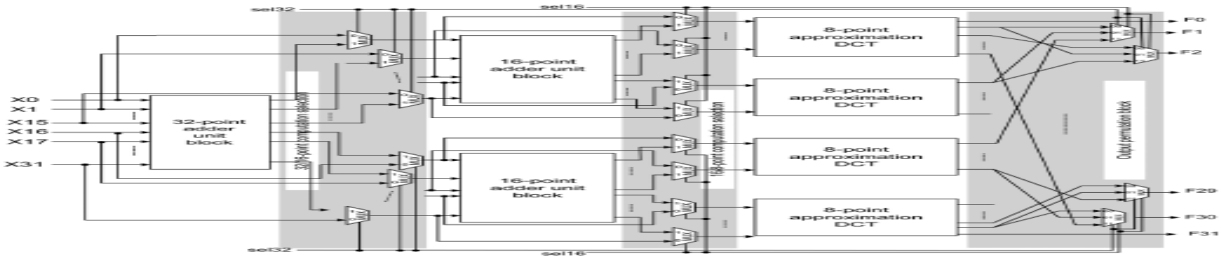

Fig.4proposed reconfigurable architecture.

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, 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. and are used as control signals to the 3:1 MUXes. Specifically, for 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.

IV. DCT AND IDCT IN JPEG

The type of data for information in the form of images, one of the most popular compression methods is JPEG. JPEG stands for Joint Photographic Expert Group. According to widely used in JPEG image included on the internet web pages. Use JPEG create a web page with a picture can be accessed faster than a web page with an image without compression.

Color image JPEG compression consists of five steps. This is shown in figure 5. The steps are: color space conversion, down sampling, 2-D DCT, quantization and entropy coding. Grayscale image compression uses only last three steps.

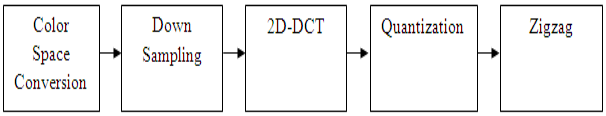

Fig.5JPEG compression steps of colour images.

JPEG stands for Joint Photographic Expert Group. According to widely used in JPEG image included on the internet web pages. Use JPEG create a web page with a picture can be accessed faster than a web page with an image without compression Color image JPEG compression consists.

Block diagram of entire system to be implemented in FPGA is shown in figure. Input data is inserted into the system every 8 bit sequentially. Actually, many DCT designs insert the input to the DCT in parallel. For example is 8 x 8 bit. This is ideal for DCT computing because it only consumes a clock cycle to insert data to 1D-DCT unit. With sequential manner, it takes 8 clock cycles to insert a set of data (8 points) to the DCT unit.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

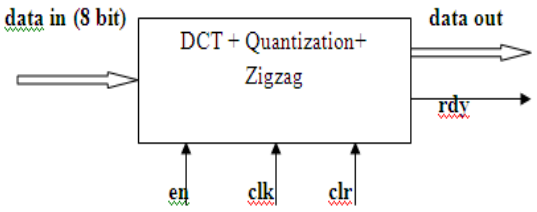

Fig.6Block representation of entaire system.

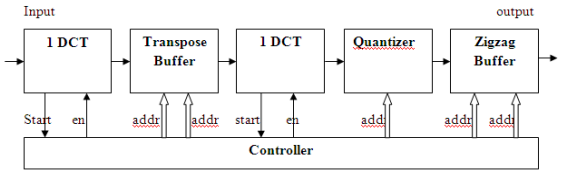

Fig.7 system architecture.

The 2D-DCT architecture, combined with zigzag and quantization used in this paper is shown in Fig. The 2DDCT module construction is modified from that also put the data sequentially into the module. Thus, the architecture of 2D-DCT was divided into two 1D DCT modules and one transpose buffer.

V. IMPLEMENTATION RESULTS

To get the matrix form of equation(1)we will use the following equation

$$T_{ij} = \begin{cases} \frac{1}{\sqrt{N}} & \text{if } i = 0 \\ \sqrt{\frac{2}{N}} \cos\left[\frac{(2j+1)i\pi}{2N}\right] & \text{if } i > 0 \end{cases}$$

Doing dct on an 8X8 block

Before we begin, it should be noted that the pixel values of a black and white image range from 0 to 255 in step 1, where pure block is represented by 0 and pure white is represented by 1 thus how a photo, illustration etc can be accurately represented by these 256 shades of grey. since the image comprises hundreds or even thousands of 8X8 block pixels ,the following description of what happens to one 8X8 block is a microcosm of the JPEG process.

Now let's start with a block of image pixel values. This particular block was choose from the very upper left hand corner of an image

$$Original = \begin{bmatrix} 154 & 123 & 123 & 123 & 123 & 123 & 123 & 136 \\ 192 & 180 & 136 & 154 & 154 & 154 & 136 & 110 \\ 254 & 198 & 154 & 154 & 180 & 154 & 123 & 123 \\ 239 & 180 & 136 & 180 & 180 & 166 & 123 & 123 \\ 180 & 154 & 136 & 167 & 166 & 149 & 136 & 136 \\ 128 & 136 & 123 & 136 & 154 & 180 & 198 & 154 \\ 123 & 105 & 110 & 149 & 136 & 136 & 180 & 166 \\ 110 & 136 & 123 & 123 & 123 & 136 & 154 & 136 \end{bmatrix}$$

because the DCT is designed to work on pixel values range from -128 to +127,the original block is "leveld off"by subtracting 128 from each entity this result in the following matrix

$$M = \begin{bmatrix} 26 & -5 & -5 & -5 & -5 & -5 & -5 & 8 \\ 64 & 52 & 8 & 26 & 26 & 26 & 8 & -18 \\ 126 & 70 & 26 & 26 & 52 & 26 & -5 & -5 \\ 111 & 52 & 8 & 52 & 52 & 38 & -5 & -5 \\ 52 & 26 & 8 & 39 & 38 & 21 & 8 & 8 \\ 0 & 8 & -5 & 8 & 26 & 52 & 70 & 26 \\ -5 & -23 & -18 & 21 & 8 & 8 & 52 & 38 \\ -18 & 8 & -5 & -5 & -5 & 8 & 26 & 8 \end{bmatrix}$$

We are now ready to perform the discrete cosine transform. which is accomplished by matrix multiplication

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

$$D = TMT'$$

In previous equation matrix M is first multiplied on the left by the DCT matrix T from the previous section, this transforms the rows. the columns are then transformed by multiplying on the right by the transpose of the dct matrix. this yields the following matrix.

1298	229	-482	-247	-250	-21	228	-83
178	678	-231	-370	128	-83	97	5
62	85	59	-26	-4	73	37	42
483	269	-197	-167	-69	-9	136	-112
242	209	-204	-89	114	183	-30	143
227	31	-76	-18	-35	65	-35	-13
-146	62	-25	9	72	22	21	1
-166	-175	124	122	-62	38	38	81

This block matrix now consists of 64DCT coefficients,cij,where I and j range from 0 to 7 the top left coefficients c00,correlates to the low frequencies of the original image block.

As we move away from c00 in all directions, the DCT coefficients correlates too higher and higher frequencies of the image block, where c77corresponds to the higher frequencies it is important to note that the human eye is most sensitive to low frequencies and result from the quantization step will reflect this fact.

If however, another level of quality and compression is desired, scalar multiples of the JPEG standard quantization matrix may be used. for a quality level grater than 50.the standard quantization matrix is multiplied by (100 quality level by 50).

Quantization is achieved by dividing each element in the transformed image matrix D by the corresponding elements in the quantization matrix, and then rounding to the nearest integer value. for the following step, quantization matrix is

$$C_{i,j} = round\left(\frac{D_{i,j}}{Q_{i,j}}\right)$$

16	4	-10	-3	-2	0	1	0
3	11	-3	-4	1	0	0	0
1	1	1	0	0	0	0	0
7	3	-1	-1	0	0	1	0
3	2	-1	-1	0	1	0	1
2	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0

The quantized matrix c is now ready for the final step of compression. Before storage, all coefficients of c are converted by an encoder to a stream of binary data .in depth coverage of the coding process is beyond the scope of the article. However we can point out one key aspect that the reader is sure to appreciate. After quantization it is quite common for most of the coefficients equal to zero. the advantage lies in the consolidation of relatively large runs of zeros, which compress very well.

The reason to use zigzag scanning is that it's reasonable to expect pixels in a block have more in common than pixels going across in a straight line. Having more in common means they can be compressed more.. the zig zag output is 16 4 3 1 11 -10 -3 -3 1 7 3 3 1 -4 -2 0 1 0 -1 2 2 -1 0 -1 -1 0 0 1 0 0 0 0 0 0 0 -1 -1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0.

Run-length encoding (RLE) is a very simple form of lossless data compression in which *runs* of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run.. the run length encoding output is 16 4 3 1 11 -10 -3 -3 1 7 3 3 1 -4 -2 0 1 0 -1 2 2 -1 0 -1 -1 0 0 1 0 127 7 -1 -1 0 127 7 1 1 0 127 8 1 0 127 9.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 9, September 2016

VI. CONCLUSION

In this paper, we have proposed a 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) at the cost of N additions 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 competing methods shows the effectiveness of the proposed approximation in terms of error energy, hardware resources consumption, and compressed image quality. 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 DCTs or four 8-point DCTs. The DCT algorithm mentioned above is used in the implementation of a JPEG compression scheme whose results are presented in the previous section.

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.

BIOGRAPHY

N L Indhira rani pursuing her Masters in the domain VLSI design in mallareddy engineering college (hyd).she completed her graduate from the college of GIET college of engineering (east Godavari) in the stream of E.C.E in the year 2013.