ISSN: 2395-1680 (ONLINE)
DOI: 10.21917/ijme.2022.0220

ARCHITECTURAL DESIGN AND OPTIMIZATION OF DISTRIBUTED ARITHMETIC BASED 2-D DISCRETE COSINE TRANSFORM

Shrikanth Shirakol and S.S. Kerur
Department of Electronics and Communication Engineering, SDM College of Engineering and Technology, India

Abstract

DCT is immensely used in Multimedia applications because it provides high energy compaction. The proposed architectural design of 1D-DCT employs an efficient computational technique, Distributed arithmetic and is synthesized using front end VLSI technique. The motive of utilising Distributed arithmetic is to have multiplier-less architecture that reduces the Area-delay product in comparison to the multiplier-based design by retaining the same structural regularities. The symmetric property of the DCT kernel matrix is applied to develop the proposed architecture which reduces the requirements of a number of multiplications by almost 50%. The 1D-DCT architecture is extended to 2D-DCT using only N 1D-DCT modules, while the conventional row-column decomposition method requires 2N 1D-DCT modules. The proposed 2D-DCT architecture is designed and implemented on a 65nm LX110T device of Vertex-5 FPGA and its performance evaluation is carried out. The debug and verification process has been carried out using the Virtual input-output technique. The results show improvement in delay and area consumption in contrast with existing models.

Keywords:

Dimensional Discrete cosine transform (1D-DCT), Distributed arithmetic (DA), Multiply and accumulate (MAC), Field programmable gate array (FPGA)

1. INTRODUCTION

In the current world, there is substantial importance for the proficient design and implementation of image transforms due to its high requirement in multimedia applications such as Digital voice and picture communication, Compression, Advanced Driver Assisted Systems (ADAS), Surveillance, Computer vision, Virtual and Augmented Reality, drones, Biomedical signal processing. DCT is used to decompose the spatial frequency of sound, image and video in terms of various cosines. It de-correlates the information and yields the transform coefficients. These coefficients can be encoded autonomously retaining the compression efficiency. Compared to Discrete Fourier Transform, DCT has only real coefficients to calculate the transformed values and provide high energy compaction which is very much essential in Image processing applications.

In DSP applications, more focus is on compressing the data i.e. image, data, speech and video. Image compression is a procedure which curtails the size of an image input without diminishing the quality of the image. There is an image transform technique called Discrete Cosine Transform which makes image compression efficient. DCT produces real values per data point. When compared to DFT, DCT has low complexity involved. FPGA is recommended for implementing DSP functions due to its low-cost, high-end DSP optimized block sets, high-level design approaches, reconfigurability, parallel resources, and increased bandwidth.

Meher et al. [1] introduced resource and power-efficient architectures for implementing integer Discrete Cosine Transforms (DCTs) of various lengths for use in High-Efficiency Video Coding (HEVC). Parallel architectures for 1-D integer DCTs of varying lengths can be derived using an efficient constant matrix multiplication approach. The structure could be reused for DCTs with lengths of 4, 8, 16, and 32 with a throughput of 32 DCT coefficients per cycle regardless of the transform size. Additionally, they proposed energy-efficient topologies for folded and full-parallel 2-D DCT implementations. It is discovered that the proposed architecture supports UHD.

Chatterjee et al. [2] came up with a real-valued DCT that cuts down on the hardware cost and processing time by significantly lowering the complexity and length of intermediate data. However, it still has the same coding speed as the integer DCT. More than that, a hardware efficient data flow model is also shown for the 2D-DCT architecture. This model shows that a transpose memory of 15-bit data depth is enough to process 9-bit vestigial data. To tackle the problem of lack of synchronisation among the multiple rotation angles of the CORDIC, Hai Huang et al. developed a CORDIC-based fast method for power-of-two-point DCT/IDCT and deduced its corresponding efficient VLSI implementation [3].

Nam and Lee [4] presented that the N×N DCT, where $N = 2m$, can be determined by just using N 1-D DCTs and additions, rather than $2N$ 1-D DCTs as in the traditional row-column methodology. As a result, the total number of multiplications necessary for the suggested technique is just half that required for the row-column methodology, and it is also fewer than that of most other fast algorithms, but the number of additions is approximately similar to that of others. It is further demonstrated that for hardware implementation of the proposed algorithm with parallelism, only $0.5N$ 1-D DCT modules are required.

Garrido et al. [5] stated that Future video coding (FVC) which is named a versatile video coding standard (VVC) in a later stage. Although the standardization process is still in its beginning phases, it is projected to eventually replace HEVC. The adaptive multiple core transform is one of the advancements given, which uses five different types of 2-D discrete sine/cosine transforms (DCT-II, DCT-V, DCT-VIII, DST-I, and DST-VII) with design flexibility ranging from 4X4 to 64X64 transform unit sizes.

The literature reveals that 1D-DCT architecture is often optimised by computational techniques to improvise the performance. One such computational technique presented in various articles is Distributed Arithmetic. The paper has focused on DA, which replaces multipliers by shifting and adding operations hence computational speed increases and also improves the hardware utilization [6]-[8].

Several compact models for DCT architectural design have been studied using various VLSI approaches such as CORDIC, Parallel architectures, orthogonal approximation, recursive sparse matrix decomposition, symmetry, lifting schemes, multiplier-less designs, and so on [9]-[27]. The proposed design can be applied to image input through moving window architecture till the whole image values are traversed [28].

The author could comprehend and integrate the concepts of the work through a review of the literature which broadly focuses on DSP and VLSI systems [29]-[33].

The Proposed work considers a technique called the modified row-column decomposition method exploiting symmetricity of the DCT basis function. The 1D-DCT architecture is extended to 2D-DCT employing the proposed 1-D DCT. The comparative analysis of the performance metrics is carried out within the techniques employed, and with existing models. 8×8 DCT kernel matrix is considered in the work, as it is most predominant in the JPEG application. The same work can be extended for higher kernel dimensions.

2. DESIGN OF 1-D DCT USING DA

RTL architectural design process includes the integration of millions of transistors on a single chip. In a conventional implementation, the basic definition of DCT is used. The flowchart in Fig.1 illustrates the steps of the design of 1-D DCT.

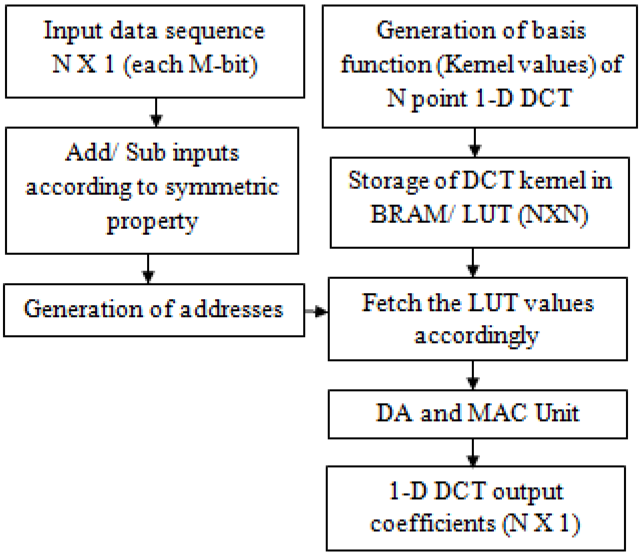
Input data sequence N X 1 (each M-bit)
Add/ Sub inputs according to symmetric property
Generation of addresses
Generation of basis function (Kernel values) of N point 1-D DCT
Storage of DCT kernel in BRAM/ LUT (NXN)
Fetch the LUT values accordingly
DA and MAC Unit
1-D DCT output coefficients (N X 1)

Fig.1. Design methodology of 1-D DCT using the modified row-column decomposition method

Eq.(1) gives the Basic definition of 1-D DCT [12] where $x(n)$ is input, $Y(k)$ is output, $\alpha(k)$ is constant, and n and k are input and output indexes.

$$Y(k)=\alpha(k)\sum_{n=0}^{N-1}x(n)\cos\left(\frac{\pi(2n+1)k}{2N}\right) \quad (1)$$

where $0\le k\le N-1$

The kernel matrix for 1-D DCT, is realized autonomously to construct the DCT basis function matrix.

$$C_x=\cos\left(\frac{x\pi}{16}\right) \quad (2)$$

$$x=(2n+1)k \quad (3)$$

The Eq.(2) and Eq.(3) are applied to construct the DCT basis kernel matrix [30]. The computed value of coefficients is denoted as C_1=0.981, C_2=0.924, C_3=0.831, C_4=0.707, C_5=0.556, C_6=0.383, C_7=0.195.

$$\begin{bmatrix} y0\\ y1\\ y2\\ y3\\ y4\\ y5\\ y6\\ y7 \end{bmatrix} = \begin{bmatrix} c4 & c4 & c4 & c4 & c4 & c4 & c4 & c4\\ c1 & c3 & c5 & c7 & -c7 & -c5 & -c3 & -c1\\ c2 & c6 & -c6 & -c2 & -c2 & -c6 & c6 & c2\\ c3 & -c7 & -c1 & -c5 & -c5 & c1 & c7 & -c3\\ c4 & -c4 & -c4 & c4 & c4 & -c4 & -c4 & c4\\ c5 & -c1 & c7 & c3 & -c3 & -c7 & c1 & -c5\\ c6 & -c2 & -c2 & -c6 & -c6 & c2 & -c2 & c6\\ c7 & -c5 & c3 & -c1 & c1 & -c3 & c5 & c7 \end{bmatrix} * \begin{bmatrix} x0\\ x1\\ x2\\ x3\\ x4\\ x5\\ x6\\ x7 \end{bmatrix}$$

The DCT coefficients $Y(k)$ for k=0 and k=1 have expressed in terms of kernel values C_x in the Eq.(4) and Eq.(5).

$$k=0,\ Y(0) = (c_0\times x_0) + (c_0\times x_1) + (c_0\times x_2) + (c_0\times x_3) + (c_0\times x_4) + (c_0\times x_5) + (c_0\times x_6) + (c_0\times x_7) \quad (4)$$

$$k=1,\ Y(1) = (c_1\times x_0) + (c_3\times x_1) + (c_5\times x_2) + (c_7\times x_3) + (-c_7\times x_4) + (-c_5\times x_5) + (-c_3\times x_6) + (-c_1\times x_7) \quad (5)$$

The conventional method requires 64 multiplication and 56 Addition/Subtraction operations. The performance improvisation in the proposed architecture is achieved by applying the symmetric property of matrices. By employing the symmetric property of the DCT basis function, the number of multiplications can be reduced to 32 but it requires 8 extra Addition/Subtraction operations, which is reasonable against the reduction of multipliers.

Kernel matrix with symmetric property is given by

$$\begin{bmatrix} y0\\ y2\\ y4\\ y6\\ y1\\ y3\\ y5\\ y7 \end{bmatrix} = \begin{bmatrix} c4 & c4 & c4 & c4 & 0 & 0 & 0 & 0\\ c2 & c6 & -c6 & -c2 & 0 & 0 & 0 & 0\\ c4 & -c4 & -c4 & c4 & 0 & 0 & 0 & 0\\ c6 & -c2 & c2 & -c6 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & c1 & c3 & c5 & c7\\ 0 & 0 & 0 & 0 & c3 & -c7 & -c1 & -c5\\ 0 & 0 & 0 & 0 & c5 & -c1 & c7 & c3\\ 0 & 0 & 0 & 0 & c7 & -c5 & c3 & -c1 \end{bmatrix} * \begin{bmatrix} x0+x7\\ x1+x6\\ x2+x5\\ x3+x4\\ x0-x7\\ x1-x6\\ x2-x5\\ x3-x4 \end{bmatrix}$$

After the application of symmetric property, the DCT coefficients $Y(k)$ for k=0 and k=1 have expressed in terms of kernel values C_x, in the Eq.(6) and Eq.(7).

$$k=0,\ Y(0) = [c_0\times (x_0+x_7)] + [c_0\times(x_1+x_6)] + [c_0\times (x_2+x_5)] + [c_0\times (x_3+x_4)] \quad (6)$$

$$k=1,\ Y(1) = [c_2\times (x_0+x_7)] + [c_6\times (x_1+x_6)] + [-c_6\times(x_2+x_5)] + [-c_2\times (x_3+x_4)] \quad (7)$$

There are various techniques to implement the proposed design in an optimized way. The main goal is to achieve computational optimization. Multipliers are one of computationally intense digital circuits. The proposed design is implemented by employing the technique called Distributed Arithmetic (DA). Calculating inner products with DA is a fast, non-multiplication-based method, unlike the existing multiplier-based techniques. The DA provides efficient computations in terms of hardware utilization. The DA differs in the following ways; it generates various partial products by addition and shift operations [6]-[8]. It uses a lookup table and shift registers. The procedure to employ DA in the DCT architecture is explained further. The following denotations are used in the architecture.

C= [C_1, C_2…C_N] is matrix of constant values.

X= [X_1, X_2…X_N] is matrix of input variable.

Each C_N, X_N are M-bits and N-bits respectively.

X_k = {b_{k1}, b_{k2} … $b_{k(N-1)}$} be a N-bits scaled number.

$$X_k = \sum_{n=1}^{N-1} b_{kn} 2^{-n} \tag{4}$$

$$\left[\sum_{k=1}^{k=K} c_k b_{kn}\right] = f_n\left(b_n, b_{2n} ....b_{kn}\right) \tag{5}$$

Pre-calculated values can be stored in LUT of 2^k words using Eq.(6). The computed LUT values in Table.1 are generated and are stored in the RAM.

$$Y\left(k\right) = \sum_{n=1}^{N-1}\left[\sum_{k=1}^{k=K} c_k b_{kn}\right] 2^{-n} \tag{6}$$

Each integer value is stored in a vector of bit width equal to 2 for magnitude. Since the maximum integer value in the LUT is 2, Two magnitude bits are sufficient to store all the integer values. Similarly, each fractional value is stored in a vector of bit width equal to 14. One sign bit is used for both integer and fractional parts. The LUT is calculated for only 4 decimal points. Hence, 14 magnitude bits are sufficient to store all the fractional values. The total size of LUT is given by (8*16*17 =2176 bits) where 8 is for 8-point DCT, 16 is for addresses and 17 is for each location holding a value.

The proposed architecture of 1-D DCT with 8 input lines consists of Adder/Subtraction unit, Address Generator (Even and odd), LUT and MAC unit. By utilizing the symmetric property of the DCT basis function matrix, the architecture is greatly optimised due to the reduction of the multiplication operations, with the trade-off of 8 addition/subtraction units. Further Distributed Arithmetic technique optimizes the hardware utilization. There are 8 input signals of 8 bit each and 8 output signals, each output signals have 3 parts (sign, integer and fraction part). The Q-format is used in this case, which provides almost accurate results in comparison with IEEE 754 format.

The 1-D DCT design employs the symmetric property of the kernel matrix, which requires the addition and subtraction of input accordingly. The proposed architecture uses an 8×8 kernel matrix and the following additions/subtractions are carried out in the foremost step of the design.

S_0=X_0+X_7; S_2=X_1+X_6; S_4=X_2+X_5; $S6$=X_3+X_4

D_1=X_0-X_7; D_3=X_1-X_6; D_5=X_2-X_5; $D7$=X_3-X_4

The sum and difference resulting are represented in DA based 4-bit scaled number format, which is subsequently used as an address to fetch the contents from the LUT. Two different types of addresses are generated: Even and Odd addresses. Address for even output lines is applied to the even index DCT's LUT. Address for odd output lines is applied to the odd index DCT's LUT. Addresses are generated as follows:

First address = {S_0[MSB], S_1[MSB], S_2[MSB], S_3[MSB]};

Second address = {S_0[MSB-1], S_1[MSB-2], S_2[MSB-3], S_3[MSB-4]};

Last address = {S_0[LSB], S_1[LSB], S_2[LSB], S_3[LSB]};

Similarly odd addresses are calculated by D_4 to D_7. The width of each address is 4-bit hence totally 16 different combinations can be formed. The generated address is applied to LUT serially initially from first address to the last address, to fetch the pre-calculated values.

Table.1. LUT for 8 Point 1D-DCT

	Y0	**Y2**	**Y4**	**Y6**	**Y1**	**Y3**	**Y5**	**Y7**
0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000
0001	+0.7071	-0.9239	+0.7071	-0.3827	+0.1951	-0.5556	+0.8315	-0.9808
0010	+0.7071	-0.3827	-0.7071	+0.9239	+0.5556	-0.9808	+0.1951	+0.8315
0011	+1.4142	-1.3066	+0.0000	+0.5412	+0.7507	-1.5364	+1.0266	-0.1493
0100	+0.7071	+0.3827	-0.7071	-0.9239	+0.8315	-0.1951	-0.9808	-0.5556
0101	+1.4142	-0.5412	+0.0000	-1.3066	+1.0266	-0.7507	-0.1493	-1.5364
0110	+1.4142	+0.0000	-1.4142	+0.0000	+1.3871	-1.1759	-0.7857	+0.2759
0111	+2.1213	-0.9239	-0.7071	-0.3827	+1.5822	-1.7315	+0.0458	-0.7049
1000	+0.7071	+0.9239	+0.7071	+0.3827	+0.9808	+0.8315	+0.5556	+0.1951
1001	+1.4142	+0.0000	+1.4142	+0.0000	+1.1759	+0.2759	+1.3871	-0.7857
1010	+1.4142	+0.5412	+0.0000	+1.3066	+1.5364	-0.1493	+0.7507	+1.0266
1011	+2.1213	-0.3827	+0.7071	+0.9239	+1.7315	-0.7049	+1.5822	+0.0458
1100	+1.4142	+1.3066	+0.0000	-0.5412	+1.8123	+0.6364	-0.4252	-0.3605
1101	+2.1213	+0.3827	+0.7071	-0.9239	+2.0074	+0.0808	+0.4063	-1.3413
1110	+2.1213	+0.9239	-0.7071	+0.3827	+2.3679	-0.3444	-0.2301	+0.4710
1111	+2.8284	+0.0000	+0.0000	+0.0000	+2.5630	-0.9000	+0.6014	-0.5098

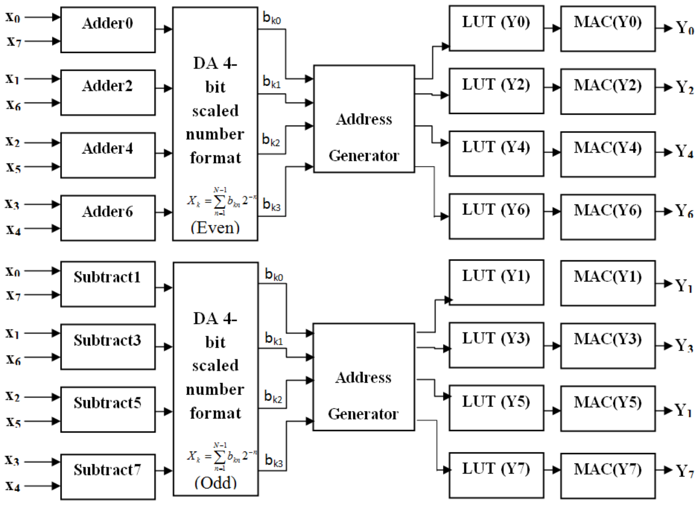

Fig. 2. Datapath of proposed architecture of 8 input line 1-D DCT employing DA and MAC unit

MAC unit in Fig.3, mainly consists of adder/subtractor, shift register and left shift operation. The content of the shift register is multiplied by 2 using the left shift operation (multiplier less). The obtained result is added or subtracted from data which is extracted from the LUT and stored in the shift register. Finally, the value stored in the shift register is read after certain clock cycles. The (In case of 8-bit input, the value of shift register will be read after 9 clock cycles).

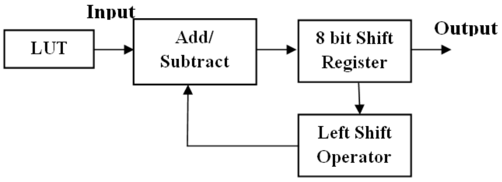

Fig.3. MAC unit used in 1-D DCT architecture

3. DESIGN OF 2-D DCT USING 1-D DCT

The two Dimensional DCT sequence Y_{mn} where m and n range from 0 to N-1 is given in Eq.(7) for a given data sequence x_{ij} where i and j range from 0 to N-1 [4].

$$y_{mn}=\frac{4}{n^2}u(m)u(n)\sum_{i=0}^{N-1}\sum_{i=0}^{N-1}\cos\left(\frac{(2i+1)m}{2N}\right)\pi\cos\left(\frac{(2i+1)n}{2N}\right)\pi \quad (7)$$

$$\text{where, } U(n)=U(m)=\begin{cases}\frac{1}{\sqrt{2}} & m=n=0\\ 1 & Otherwise\end{cases}$$

Neglecting the scale factor $4/N^2u(m)u(n)$ for convenience, and by defining a de-normalized form of Y_{mn},

$$y_{mn}=\frac{1}{2}\left\{\begin{matrix}\sum_{i=0}^{N-1}\sum_{i=0}^{N-1}X_{ij}\cos\frac{(2i+1)m+(2j+1)n}{2N}\pi+\\ \sum_{i=0}^{N-1}\sum_{i=0}^{N-1}X_{ij}\cos\frac{(2i+1)m-(2j+1)n}{2N}\pi\end{matrix}\right\} \quad (8)$$

The preceding relationship can be utilized to obtain the N×N DCT for N=8. The DCT of N×N can be divided into two transforms. Each term can be described in terms of N 1-D DCTs by arranging and modifying the data in such a way that the N×N DCT can be generated from N distinct 1-D DCTs.

As previously stated, the criterion for the basis function of the transforms in Eq.(8) to be analogous to the 1-D DCT in Eq.(1) is that $2im+m\pm2j+1$ should be represented as $2i+1$ scaled by ρ, where ρ is an integer greater than or equal to one. In order to meet this requirement, we obtain

$$2i+1 = 2i\rho+\rho\text{mod}2N \quad (9a)$$

$$2j+1 = 2N-(2i\rho+\rho)\text{mod}2N \quad (9b)$$

where ρ positive odd integer in the range 1:N-1. Due to the cosine function, an even value of ρ produces the same kernel. When ρ is larger than N-1, the value of j equals one of the values produced by the ρ in the range 1:N-1.

For a specific value of ρ, {$j(\rho;a)$; i ranging from 0 to N-1} is the sequence of j generated by Eq.(9a), and {$j(\rho;b)$; i ranging from 0 to N-1} is the sequence of j generated through Eq.(9b). Thus, by clustering the two-dimensional state vector {x_{ij}; i,j ranging from 0 to N-1} into N one-dimensional DCT sequences {$x_{ij}(\rho;a)$ and $x_{ij}(\rho;b)$}, the 1D transforms in Eq.(8) may be represented as a sum of one-dimensional DCT. The denotation of one-dimensional data vectors is done by R_ρ^a and R_ρ^b respectively. The following clustering of input sequences is generated for 2-D DCT design using N 1-D DCTs.

$$R_\rho^a=\{x_{ij}(\rho;a);\ i \text{ ranging from 0 to } N\text{-1},\ j(\rho;a)=\rho_i+(\rho-1)/2 \bmod \text{N}\} \quad (10a)$$

$$R_\rho^b=x_{ij}(\rho;b);\ i \text{ ranging from 0 to } N\text{-1},\ j(\rho;b)=N-1-\rho_i+(\rho-1)/2 \bmod \text{N}\} \quad (10b)$$

The inputs are grouped using (10) in order to match the DCT kernel matrix of 1D DCT. The output Y_{mn} of *Eq.*((8) is resolved into two components $\alpha_{\rho l}$ and $\beta_{\rho l}$ whose DCT kernel matches that of 1-D DCT. $\alpha_{\rho l}$ is computed for even value of n. The proposed architectural design of 1D-DCT is applied for realizing $\alpha_{\rho l}$.

$$\alpha_{\rho l}=\sum_{i=0}^{N-1}\left(x_{ij}(\rho,a)+x_{ij}(\rho,b)\right)\cos\left(\frac{(2i+1)\omega}{2N}\right)\pi \quad (11a)$$

where, $\beta_{\rho l}$ is computed for odd value of n. The 1-D DCT architecture discussed is used for realization. The quotient $q_{\rho i}$ is computed as $(\rho_i+(\rho-1)/2)/N$, to adjust the sign of $\beta_{\rho l}$.

$$\beta_{\rho l}=\sum_{i=0}^{N-1}(-1)^{q_{\rho i}}\left(x_{ij}(\rho,a)+x_{ij}(\rho,b)\right)\cos\left(\frac{(2i+1)\omega}{2N}\right)\pi \quad (11b)$$

where ω is $m\pm n\rho$ for ω in the range 0 to N-1. If $m\pm n\rho$ is out of the range then new value ω is computed using ω mod 2N. Sign convention follows the trigonometric cosine function. Employing 1-D DCT, $\alpha_{\rho l}$ and $\beta_{\rho l}$ are computed and consequently are added/subtracted accordingly to compute the final output Y_{mn} as shown in the proposed Two-dimensional DCT architecture of Fig.4 and Fig.5.

$$y_{mn}=0.5\sum_{\rho=1(odd)}^{N-1}\left(\alpha_{\rho l+}+\alpha_{\rho l-}\right)\text{for } n\to even \quad (12a)$$

$$y_{mn}=0.5\sum_{\rho=1(odd)}^{N-1}\left(\beta_{\rho l+}+\beta_{\rho l-}\right)\text{for } n\to odd \quad (12b)$$

The postfix +, – to $\alpha_{\rho l}$ and $\beta_{\rho l}$ indicates $\omega=m+n\rho$ and $\omega=m-n\rho$ respectively. The architecture shown in Fig.4 is for even value of n. The inputs are grouped as $R_\rho{}^a$ as in Eq.(10a). $\alpha_{\rho l}$ in Eq.(11a) realized 1-D DCT is applied with intermediate value. The final output Y_{mn} is computed by adding/subtracting $\alpha_{\rho l}$'s.

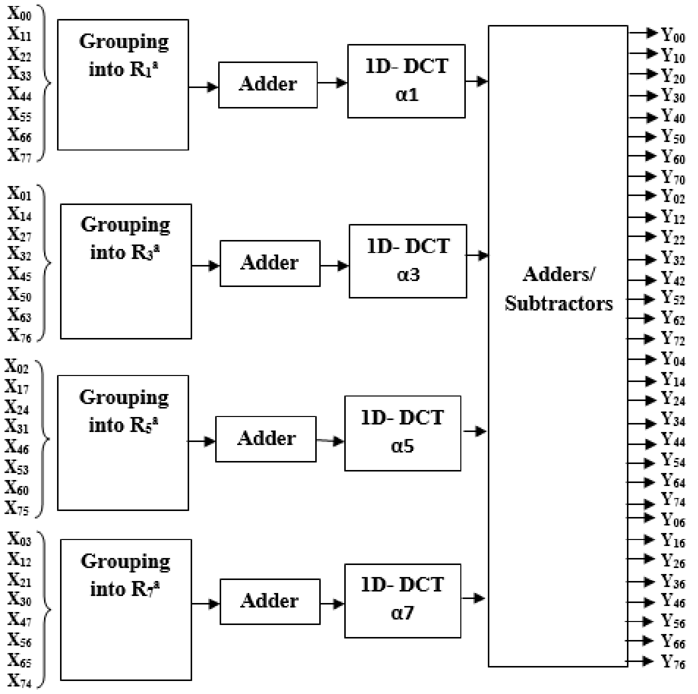
X00 X11 X22 X33 X44 X55 X66 X77
Grouping into R1a
Adder
1D- DCT α1
X01 X14 X27 X32 X45 X50 X63 X76
Grouping into R3a
Adder
1D- DCT α3
X02 X17 X24 X31 X46 X53 X60 X75
Grouping into R5a
Adder
1D- DCT α5
X03 X12 X21 X30 X47 X56 X65 X74
Grouping into R7a
Adder
1D- DCT α7
Adders/ Subtractors
Y00 Y10 Y20 Y30 Y40 Y50 Y60 Y70
Y02 Y12 Y22 Y32 Y42 Y52 Y62 Y72
Y04 Y14 Y24 Y34 Y44 Y54 Y64 Y74
Y06 Y16 Y26 Y36 Y46 Y56 Y66 Y76

Fig.4. Datapath of the proposed 2D-DCT architecture (*n* is even)

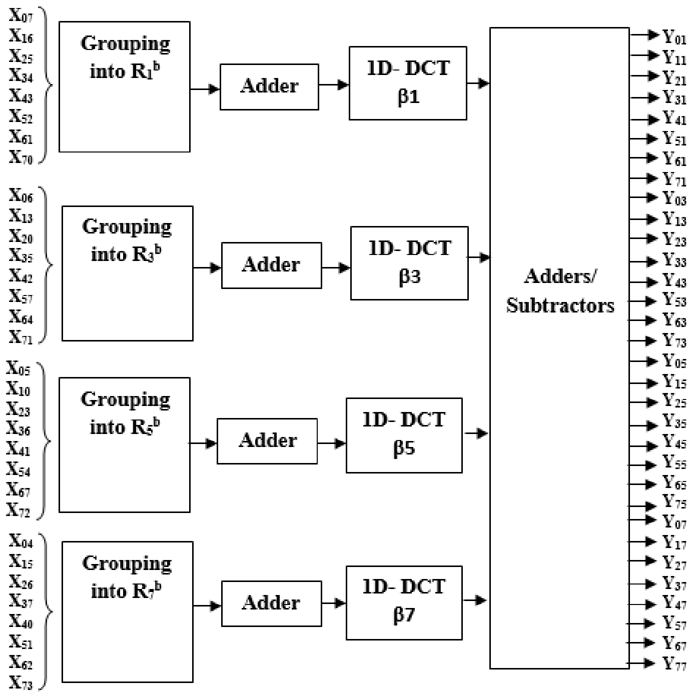
X07 X16 X25 X34 X43 X52 X61 X70
Grouping into R1b
Adder
1D- DCT β1
X06 X13 X20 X35 X42 X57 X64 X71
Grouping into R3b
Adder
1D- DCT β3
X05 X10 X23 X36 X41 X54 X67 X72
Grouping into R5b
Adder
1D- DCT β5
X04 X15 X26 X37 X40 X51 X62 X73
Grouping into R7b
Adder
1D- DCT β7
Adders/ Subtractors
Y01 Y11 Y21 Y31 Y41 Y51 Y61 Y71
Y03 Y13 Y23 Y33 Y43 Y53 Y63 Y73
Y05 Y15 Y25 Y35 Y45 Y55 Y65 Y75
Y07 Y17 Y27 Y37 Y47 Y57 Y67 Y77

Fig.5. Datapath of the proposed 2D-DCT architecture (*n* is odd)

The architecture shown in Fig.5 is for odd value of *n*. The inputs are grouped as $R_\rho{}^b$ in Eq.(10b). $\beta_{\rho l}$ in Eq.(11b) realized 1-D DCT is applied with intermediate value. The final output Y_{mn} is computed by adding/subtracting $\beta_{\rho l}$'s. The clustering of the input vectors becomes important while using 1D DCT as sub-block in the design of 2D DCT. This process must be handled with extreme caution.

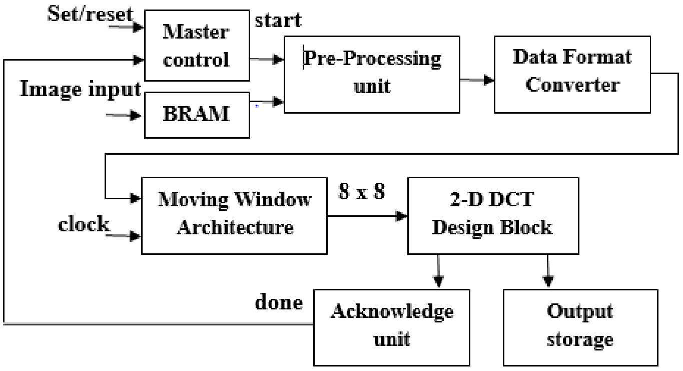

Fig.6. Control Logic model of proposed 2-D DCT architecture

The Fig.6 depicts the control logic model for the proposed work. Considering two-dimensional input data as image of the appropriate dimension is stored in the Block RAM of FPGA (LX110T). The set signal from the Master control initiates the processing and data conversion of the image data. The moving window architecture feeds 8×8 part of the image data at once to the 2-D DCT block. The 2-D DCT computes the results and acknowledgement signal is generated for receiving the next set of inputs.

4. RESULTS AND DISCUSSIONS

The results for the proposed 1D and 2-D DCT architecture are presented in this portion. The state-of-the-art tool is used for implementing the proposed design on the FPGA hardware. As the number of input and output ports is massive, the Chip Scope pro tool package is utilised, which has a variety of modules that may be added to the HDL model to record inputs and outputs directly from the FPGA hardware. Integrated Controller (ICON) and Virtual Input/Output (VIO) modules are laboured to facilitate providing input and display the output. In real-time, this machine can monitor and drive signals into the design. The inputs can be provided virtually and outputs are observed on the monitor screen, which are generated by the FPGA board.

Table.2. Root mean square deviation of the output values of proposed 1-D DCT design

Sample Inputs (*N*=8)	Output Coefficients		Error E_i
	Proposed Architecture	Manual Calculation	
64	202.202	202.2325	0.0305
56	44.581	44.5756	0.0054
105	-79.729	-79.7083	0.0207
133	-15.221	-15.1996	0.0214
92	17.675	17.6777	0.0027
66	21.723	21.7277	0.0047
34	10.803	10.8206	0.0176
22	-5.927	-5.9068	0.0202

RMSD= $\sqrt{\dfrac{\sum_{i=1}^{N} E_i^2}{N}}$ =	0.04

The similar deviation in mean square error is observed in 2D-DCT output. The proposed architectures are implemented on Vertex-5 XC5VLX110T FPGA board, which is built on 65nm technology node. The performance metrics for 1D-DCT (*N*=8) are computed using the state-of-the-art tool and are tabulated in Table.3. Also, Fig.7-Fig.9 shows the performance metrics computed for 2D-DCT for the dimensions 2×2, 4×4 and 8×8.

Table.3. Performance Analysis of proposed 1D-DCT for N=8

Performance Metrics	Parameters	Results
Delay Analysis (ns)	Logic Delay (ns)	2.448 (25.1%)
	Routing Delay (ns)	7.308 (74.9%)
	Total Delay	9.756
Power Analysis (mW)	Quiescent power	1033.63
	Dynamic power	14.51
	Total power	1048.14
Device Utilization (#)	Slice Registers	311 (1%)
	Slice LUT's	2,055 (3%)
	Occupied Slices	957 (5%)

The graphs show that higher values of *N* necessitate a greater investment in resources. In other words, as the dimension increases, so does the possibility of parallelised execution. When dealing with larger dimension input data, the possibility for performance enhancement is greater.

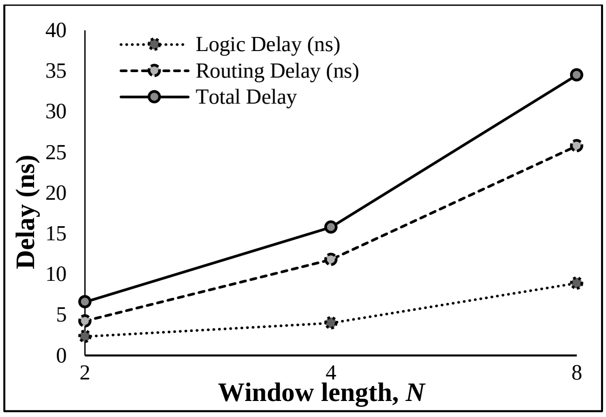

Fig.7. Delay analysis of proposed 2-D DCT architecture

1D-DCT and 2D-DCT architectures are aimed to achieve efficiency in the performance metrics such as Area, Power, Propagation delay, Latency, Mean square error, PSNR, Precision, etc. The proposed architecture is aimed to have high precision and optimized Area-delay product (ADP). High precision is certain due to usage of Q-format wherein fractional parts of the values are computed separately. Optimized ADP is achieved by performing multiplication using DA and MAC units. The vertex FPGA is a 65nm technology node. The comparative analysis with existing technique becomes extremely narrow due to the parameters considered and performance metrics aimed to achieve. An attempt is made to compare the results with the architectures considering the similar parameters.

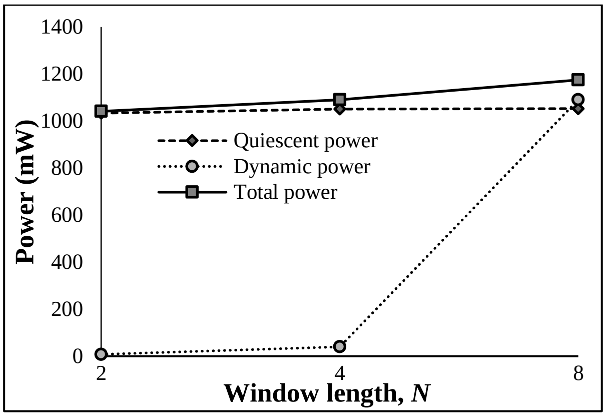

Fig.8. Power analysis of proposed 2-D DCT architecture

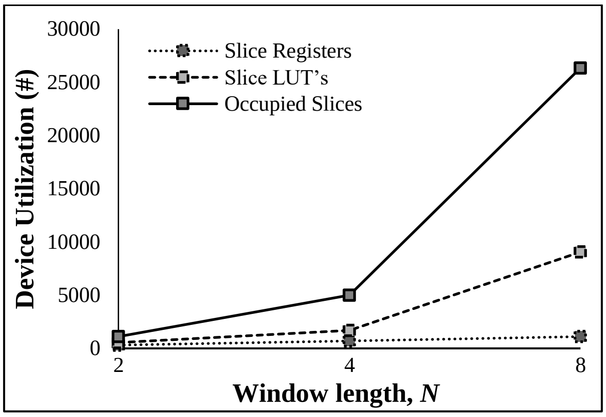

Fig.9. Device utilization summary of proposed 2-D DCT design

Megalingam et al. [16] utilizes 2094 LUTs for 1-D DCT design whereas the proposed 1D-DCT design consumes 2055. Urbi et al. [17] presented a 1D-DCT architecture with a propagation delay of 17.9ns. The proposed architecture provides more than 80% improvement with 9.6ns. Three decimal precision is considered for representing fractional parts, which yields optimum output with negligible mean square error. Most of the DCT algorithms in the literature utilize multiplier blocks in the computationally intense architecture. The proposed architecture outwits DA and MAC units in the design.

5. CONCLUSION

The work presents high precision 1D-DCT and 2D-DCT architectures. The proposed model implements 2D-DCT using N 1D-DCT modules, unlike the conventional row-column decomposition method where 2N 1D-DCT modules are required. Distributed arithmetic units and MAC units are employed to accomplish multiplier-less multiplication by shift and add operations. The computational speed of the proposed model is increased with less hardware requirement, which results in a favourable Area-Delay product. The proposed architecture is implemented on the XUPV5-LX110T device of 65nm technology node vertex-5 FPGA. The obtained results illustrate that the proposed model shows significant improvement in LUT utilization and more than 25% improvement in time delay consumption. Mean square error is negligibly small due to high precision computation which provides accuracy up to 3 decimal points. The proposed architecture can be further extended to higher dimensions.

REFERENCES

[1] Pramod Kumar Meher, Sang Yoon Park and Basant Kumar Mohanty, "Efficient Integer DCT Architectures for HEVC", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 24, No. 1, pp. 1-13, 2014.

[2] Subiman Chatterjee and Kishor Sarawadekar, "An Optimized Architecture of HEVC Core Transform using Real-Valued DCT Coefficients", *IEEE Transactions on Circuits and Systems-II, Express Briefs*, Vol. 65, No. 12, pp. 1-16, 2018.

[3] Hai Huang and Liyi Xiao, "CORDIC based Fast Algorithm for Power of Two Point DCT and its Efficient VLSI Implementation", *Microelectronics Journal*, Vol. 45, pp. 1480-1488, 2014.

[4] Nam Ik Cho and Sang Uk Lee, "Fast Algorithm and Implementation of 2-D Discrete Cosine Transform", *IEEE Transactions on Circuits and Systems*, Vol. 38, No. 3, pp. 297-305, 1991.

[5] Matias J. Garrido, Fernando Pescador, M. Chavarrias, P.J. Lobo and Cesar Sanz, "A High Performance FPGA-Based Architecture for the Future Video Coding Adaptive Multiple Core Transform", *IEEE Transactions on Consumer Electronics*, Vol. 64, No. 1, pp. 1-14, 2018.

[6] Sungwook Yu and Earl E. Swartzlander, "DCT Implementation with Distributed Arithmetic", *IEEE Transaction on Computers*, Vol. 50, No. 9, pp. 1-15, 2001.

[7] Yuk-Hee Chan and Wan-Chi Siu, "On the Realization of Discrete Cosine Transform using the Distributed Arithmetic", *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, Vol. 39, No. 3, pp. 1-13, 1992.

[8] Yung-Pin Lee, Thou-Ho Chen, Liang-Gee Chen, Mei-Juan Chen and Chung-Wei Ku, "A Cost-Effective Architecture for 8*8 Two-Dimensional DCT/IDCT using Direct Method", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 7, No. 3, pp. 1-14, 1997.

[9] Darren Slawecki and Weiping Li, "DCT/IDCT Processor Design for High Data Rate Image Coding", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 2, No. 2, pp. 45-56, 1992.

[10] Jie Liang and Trac D. Tran, "Fast Multiplier Less Approximations of the DCT with the Lifting Scheme", *IEEE Transactions on Signal Processing*, Vol. 49, No. 12, pp. 1-19, 2001.

[11] Tian Sheuan Chang, Chin Sheng Kung and Chein Wei Jen, "A Simple Processor Core Design for DCT/IDCT", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No. 3, pp. 1-15, 2000.

[12] N. Ahmed, T. Natarajan and K.R. Rao, "Discrete Cosine Transform", *IEEE Transactions on Computers*, Vol. 23, No. 1, pp. 90-93, 1974.

[13] Maher Jridi and Ayman Alfalou, "A Generalized Algorithm and Reconfigurable Architecture for Efficient and Scalable Orthogonal Approximation of DCT", *IEEE Transactions on Circuits and Systems-I: Regular Papers*, Vol. 62, No. 2, pp. 1-20, 2015.

[14] Ashfaq Ahmed, Muhammad Usman Shahid and Ata Ur Rehman, "N Point DCT VLSI Architecture for Emerging HEVC Standard", *VLSI Design*, Vol. 2012, pp. 1-8, 2012.

[15] Jianfeng Zhang, Paul Chow and Hengzhu Liu, "FPGA Implementation of Low-Power and High-PSNR DCT/IDCT Architecture based on Adaptive Recoding CORDIC", *Proceedings of International Conference on Field Programmable Technology*, pp. 7-9, 2015.

[16] Rajesh Kannan Megalingam, V. Vineeth Sarma, B. Venkat Krishnan and M. Rahul Srikumar, "Novel Low Power, High Speed Hardware Implementation of 1D DCT/IDCT using Xilinx FPGA", *Proceedings of International Conference on Computer Technology and Development*, pp. 1-12, 2009.

[17] Urbi Sharma, Tarun Verma and Raju Jain, "VLSI Architecture for DCT Based on High Quality DA", *International journal of Engineering and Technical Research*, Vol. 2, No. 6, pp. 1-13, 2014.

[18] S. Indumati and M. Sailaja, "Optimization of ECAT through DA-DCT", *IOSR Journal of Electronics and Communication Engineering*, Vol. 3, No. 1, pp. 1-14, 2012.

[19] K. Maharatna, A.S. Dhar and Swapna Banerjee, "A VLSI Array Architecture for Realization of DFT, DHT, DCT and DST", *Signal Processing*, Vol. 81, pp. 1813-1822, 2001.

[20] P. Subramanian and A. Sagar Chaitanya Reddy, "VLSI Implementation of Fully Pipelined Multiplierless 2D DCT/IDCT Architecture for JPEG", *Proceedings of IEEE International Conference on Signal Processing*, pp. 401-404, 2010.

[21] Vijay Kumar Sharma, K.K. Mahapatra and Umesh C. Pati, "An Efficient Distributed Arithmetic based VLSI Architecture for DCT", *Proceedings of IEEE International Conference on Devices and Communications*, pp. 1-13, 2011.

[22] M. Mohamed Asan Basiri and S.K. Noor Mahammad, "Multi-Mode Parallel and Folded VLSI Architectures for 1D-Fast Fourier Transform", *Integration*, Vol. 55, pp. 43-56, 2016.

[23] C. Loeffler, A. Ligtenberg and G.S. Moschytz, "Practical Fast 1-D DCT Algorithms with 11 Multiplications", *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pp. 988-991, 1999.

[24] S. Haroon-Ur-Rashid and J. Basart, "An Optimized DCT Based Hardware Design for FPGA Implementation of High-Altitude Images", *Proceedings International Conference on Engineering, Sciences and Technology*, pp. 1-13, 2004.

[25] Chinna V. Gowdar and M.C. Parameshwara, "Design of Energy Efficient Approximate Multipliers for Image Processing Applications", *ICTACT Journal on Microelectronics*, Vol. 7, No. 1, pp. 1057-1061, 2021.

[26] M. Lakshmi Kiran, K. Nikhileswar and K. Venkata Ramanaiah, "FPGA Implementation of CSD Based NN Image Compression Architecture", *ICTACT Journal on Microelectronics*, Vol. 6, No. 4, pp. 1052-1055, 2021.

[27] N.I. Cho and S.U. Lee, "DCT Algorithms for VLSI Parallel Implementations", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 38, pp. 121-127, 1990.

[28] Shrikanth K. Shirakol, Veerayya Hiremath and S.S. Kerur, "FPGA Based Implementation of Digital Filters for Image

Denoising", *Proceedings of International Conference on Smart sensors Measurements and Instrumentation*, pp. 1-8, 2021.

[29] Altera Corporation, "Implementing FIR Filters and FFTs with 28-nm Variable-Precision DSP Architecture", Available at https://www.intel.com/content/dam/support/jp/ja/programmable/support-resources/bulk-container/pdfs/literature/wp/wp-01140-fir-fft-dsp.pdf, 2010.

[30] Bob Broderson, "Energy Efficiency of various Embedded platforms", *Proceedings of International Conference on Wireless Power Transfer and Management for Medical Applications*, pp. 341-349, 2013.

[31] Uwe Meyer-Baese, "Digital Signal Processing with Field Programmable Gate Arrays", Springer, 2007.

[32] Roger Woods, John McAllister, Gaye Lightbody and Ying Yi, "*FPGA-based Implementation of Signal Processing Systems*", John Wiley and Sons, 2008.

[33] Donald G. Bailey, "*Design for Embedded Image processing on FPGAs*", John Wiley and Sons, 2011.