Received: April 19, 2023. Revised: June 28, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

An Improved VLSI Architectural Design of Discrete Cosine Transform Based on the Loeffler-DCT Algorithm

Shrikanth Shirakol[1*] **S S Kerur[1]**

[1]*SDM College of Engineering and Technology, Dharwad, VTU, Belagavi, India*
* Corresponding author's Email: shrikanthks09@gmail.com

Abstract: The Discrete Cosine Transform (DCT) is a basic transform block used in Adaptive Multicore Transform (AMT), which is a core of High Efficiency Video Coding (HEVC) and Versatile Video Coding (VVC) standards. AMT uses artificial intelligence technique to decide on the transform output. The suggested One-Dimensional DCT architecture is conceived by 8-point structure Loeffler-DCT technique and synthesized using a floating-point Arithmetic. By preserving the structural regularity as the Loeffler-based design, the optimized floating-point architecture consumes optimum space and delay thereby increasing the precision. The work focuses on obtaining high precision output without compromising on ADP (Area-Delay-Product). Due to the fact that the floating-point multiplier unit is developed using shift-add operations, the results reveal the accomplishment of better resolution output while maintaining Root Mean Square deviation as low as 0.0062. A total of 117 additions, 66 shifts are employed in the suggested architecture. The proposed 1D-DCT is used as a sub-block in developing the architecture of 2D-DCT using only 50% of the 1D-DCT subblocks that is needed for the conventional technique. The 8 X 8 2-D DCT, is computed using only 8 1-D DCT's and additions, instead of using 16 1-D DCT's, as in the traditional row-column method. The model is tested with standard images, which resulted in better PSNR and MSE compared to the standard method. In comparison to the state-of-the-art on DCT, the proposed method obtains a root mean square error that is negligible up to three decimal places resulting in improvement of PSNR by 17%, with a maximum clock frequency of 496MHz has been achieved. The proposed design strikes a balance between trade-off parameters such as area, speed, and precision.

Keywords: Discrete cosine transform (DCT), Floating point multiplier (FPM), Field programmable gate array.

1. Introduction

Digital signal processors are one of the most rapidly increasing technologies in the coming decades due to their diverse applications in numerous fields, such as digital speech processing, image analysis, and Artificial Intelligence and Machine Learning. It is possible to establish the extent to which two images are identical in image processing by comparing their hues, grey-scales, and depth. Hence, digital signal processors are utilised in numerous applications. Like other processors, a good DSP processor should have the fastest speed, the highest code density, and the lowest power requirement. Several DSP application-specific processors prioritise speed over other relevant criteria like space and energy. Owing to the computationally intensive nature of DSP algorithms, techniques that improve the DSP system's performance are required. VLSI architectures for DSP functions offer optimization potential. Multiplication is a crucial computational operation in DSP functions like the FFT, DCT, FIR Filters, etc. In today's technologically advanced society, the ability to manage data is essential. So, the implementation of a high-end architectural design has the potential to have a significant impact on the Signal processing [1, 4].

The authors conducted a preliminary experimental analysis on various transforms to analyse the error introduced when an input sample with varied sizes is fed into the forward transform followed by the inverse transform. Fig. 1 shows the

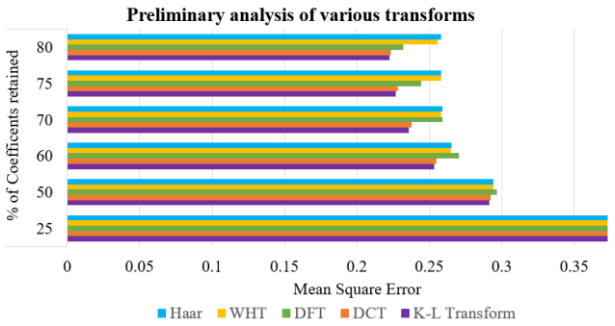

Figure. 1 Comparative analysis of various transforms on Mean square error with amount of coefficient retained for reconstruction

analysis of various transforms where the mean square error is plotted across the size of the input samples. The DCT and KL transforms resulted in the least mean square error for a higher number of samples. KLT is input-dependent, unlike DCT, making DCT the superior option.

The paper focuses on implementing 1D-DCT on FPGA using efficient computational techniques and to integrate the developed 1D-DCT to implement 2D-DCT. Existing models discussed in the literature focus primarily on enhancing area, delay, power, and accuracy. Due to the trade-off between these metrics, the design must make concessions in one or more of them. The advantage of the proposed design is that it achieves high precision through optimal resource utilization and accomplishes a balance in the metrics considered. Several VLSI architectures for DCT are proposed in the literature that use effective computational techniques to improve speed, area, power, and accuracy [2, 3].

The paper's contributions are architectural improvisations at the macro and micro levels. Macro level Optimization: The suggested work employs an efficient loeffler technique that employs the theoretically fewest multiplier blocks within it, which are substituted with floating point multiplier blocks to produce high precision output. Micro level optimization: The floating-point multiplier uses add-shift blocks, resulting in multiplier-less multiplication. These improvements result in optimal hardware resource efficiency, optimal speed, low mean square error, good PSNR, and accomplishes a balance in the metrics considered.

The structure of the paper is as follows: The second section focuses on the relevant literature survey on existing models. The third section provides extensive detail regarding the computational techniques used in the work. The fourth section focuses on the proposed architectural design of DCT. The experimental findings and their inferences are discussed in section 5. Section 6 concludes the paper.

2. Existing models

With just 11 multiplications and 29 additions, C loeffler developed a method for implementing 1D-DCT for 8 points. A novel class of DCT algorithms were proposed that are practical and quick [5]. The proposed architecture considers optimizing the loeffler-DCT technique and efforts are made to reduce the computational liability further.

With just 11 multiplications and 29 additions, C loeffler developed a method for implementing 1D-DCT for 8 points. A novel class of DCT algorithms were proposed that are practical and quick [5]. The proposed architecture considers optimizing the loeffler-DCT technique and efforts are made to reduce the computational liability further.

Anjana and Samuel presented a method for implementing a floating-point multiplier using a Vedic multiplier on DCT. A high-speed FPM is developed with the help of Vedic mathematics [6]. This paper concentrates on the implementation of a Vedic multiplier chosen for its consistent structure, but it leads to a steady increase in latency and size. FPMs contribute more to digital signal processors' delay paths.

R. L. Chung suggested a unique loeffler DCT design based on a coordinate rotation digital computer (CORDIC) mechanism. The described design was fabricated utilizing a UMC 0.18-m CMOS process with an 8.04K gate count and 100 MHz operating frequency [7]. However, Loeffler based DCT employs CORDIC which uses fixed-point arithmetic, which can lead to quantization problems in calculations. Over the course of several iterations, these mistakes may begin to distort the final outcome.

Deivakani suggested a method for implementing DCT that utilizes a recursive algorithm in general. The recurrent sparse matrix has been disintegrated by utilizing the vector symmetry from the DCT basis to build a flexible and scalable approximation approach for enforcing the longest software and hardware by employing an 8-point approximation to obtain a DCT [8]. However, recurrent sparse matrix take up more space than dense matrices since they have to keep track of both the non-zero values and their associated indices. The extra space required for storing matrix elements of higher dimensions can become a serious issue in particular applications.

Masera developed an area-efficient fixed-point architecture for the implementation of the discrete cosine transform (DCT) of different sizes in HEVC. The proposed method has reduced rate-distortion losses and achieved significant complexity savings compared to existing implementations. Two families of architectures for the 2D-DCT are designed: folded

and full-parallel [9]. The proposed method uses Integer DCT approximation which yields higher mean square error deviations.

Thiruveni provided a digital implementation of an approximate 16-point DCT architecture based on the modified gate diffusion input (MGDI) approach. The simulation result indicates that area, power, and latency are reduced by 20%, 16%, and 7%, respectively [10]. The suggest method uses low complexity DCT kernel matrix due to which the obtained results deviated from the standard coefficient. Such architecture is not preferred in the high precision requirement applications.

The N-point 1D-Integer DCT architecture suggested by M. A. Basri and N. M. Sk comprises a signed adjustable carry save adder tree-based multiplier unit. The parallel integer DCT delivers an improvement over the odd-even decomposition-based design in terms of worst path delay [11]. However, the Parallel architecture uses 2N 1D-DCT blocks for NxN 2D-DCT computation there by increasing the hardware requirements. The suggested folded architecture uses N 1D-DCT blocks.

DCT is frequently employed in digital signal processing because it approaches the statistically ideal Karhunen-Loeve transform (KLT) for inseparably linked signals. An efficient 2-D discrete cosine transform calculation was given by Sang Uk Lee in a study (DCT). Instead of utilising N/2 1-D DCTs as in the typical row-by-column method, it is demonstrated that N X N DCT, where N = 2m, can be computed using only N/4 1-D DCTs and additions and multiplications requirement. The row-column technique has twice as many multiplications as the algorithm that was suggested. It's even faster than that of other fast algorithms, which have far more multiplication operations [12, 13]. The suggested technique employs only N/2 1D-DCT blocks but 1D-DCT architecture consumes 32 multiplications, which are optimized using a new distributed architecture (NEDA). Even though the design complexity of 1D-DCT is reduced but it needs more arithmetic computation.

The existing models have majorly focused on employing efficient computational techniques, namely coordinate rotation digital computer (CORDIC) mechanism [14-16], which approximates the cosine term appearing in the DCT definition thereby amplifying the mean square error in the output coefficients. CORDIC mechanism is suited for area optimization requirements. The pipeline architectures discussed in the articles [17] results in delay overhead. The trade-off exists between area and speed performances in the pipeline architectures.

Akman proposed 2D-DCT design which reduces calculation complexity based on erroneous calculations in the steps, which can be ignored in the quantizing step [21]. Although the technique proposed reduces complexity in computation but yields high MSE and low PSNR compared to the conventional methods.

Darji and Solanki suggested an architecture which eliminated the floating-point multiplier to reduce hardware complexity. It employs MCM (multiple constant multiplication) [22]. However, there exists a trade-off between eliminating floating point multiplication with accuracy.

Maher and Srikanthan demonstrated that parallel topologies for 1D integer DCT of varying lengths can be derived from matrix multiplication schemes utilizing minimal adders. The suggested 2D DCT architecture makes use of a unique transposition buffer that, without altering the dimension of the transposition buffer, gives twice the throughput of existing solutions [25]. However, the N point 1D-DCT architecture involves the N/2 point 1D-DCT blocks. This iterative structure increases with increases as the size of N.

P. Garg and K. Suneja proposed 1D-DCT design using floating point units. The outcomes indicate improved precision and computational time [26]. However, the method employed uses 32 multiplications for 8 input data set.

To summarize, a review of the available literature demonstrates the requirement for design, to balance the trade-off between accuracy, processing time, and space utilization. The suggested method aims to enable an effective investigation of the design space to find the best compromises among the parameters under consideration. The proposed study employs Loeffler DCT, which is proven in the literature to be computationally efficient. Furthermore, floating-point units are used in it to improve precision. The floating-point units are designed with care to use minimal time with a slight area utilization overhead.

3. Efficient computational blocks

The Proposed 1-D DCT architecture results in high precision model due to the usage of efficient computational blocks floating point arithmetic. Fig. 2 depicts the architecture of floating-point adder/ subtractor. Exponent comparison, pre-aligning, adding, normalising, and round off are the five phases of the traditional floating-point addition algorithm. The phases for computing addition or subtraction (X ± Y) for the given floating-point integers X = (s1, e1, m1) and Y = (s2, e2, m2) are described in detail. The

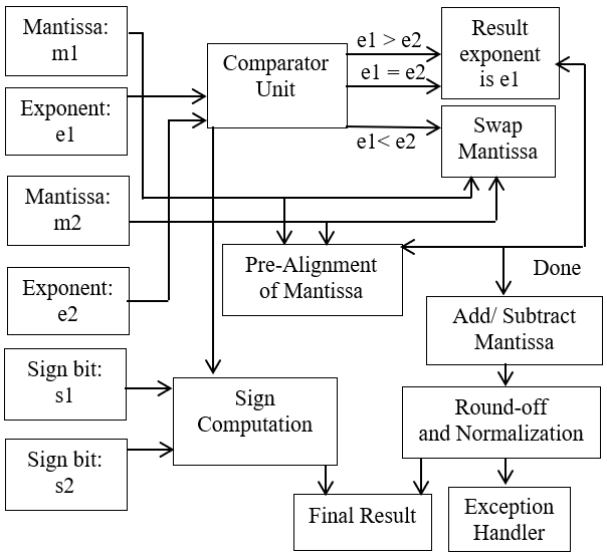

Figure. 2 Architecture of Floating-point Adder-Sub

phases for computing addition or subtraction (X ± Y) for the given floating-point integers X = (s1, e1, m1) and Y = (s2, e2, m2) are described in detail.

The architecture has comparator unit which compare the exponents of the two numbers X and Y which are to be added/subtracted. If exponent of X, e1 is less than exponent of Y, e2 (e1< e2), swap position of mantissas (m1 and m2). Assign the bigger exponent as the result's provisional exponent. Align the mantissas in second step by moving the smaller mantissa to the right by (|e1-e2|) bits. The arranged mantissas of both X and Y are now added or subtracted to get an intermediate result of the mantissa using pre-alignment unit.

Normalization process is carried out to represent the result in IEEE standard floating-point representation. If the tentative result contains leading zeros, shift the result to the left and strip the number of leading zeros from the exponent. Shift operation on floating point representation is carried out as mentioned. For n-left shift, add n-value to the exponent part of floating-point binary number. For n-right shift, subtract n-value to the exponent part of floating-point binary number.

Exception Handler unit handles certain exception that may occur in the algorithm. A flag is set by default, and the calculation proceeds. Exceptions are made in the following situations: Overload (when the rounded quantity is big to represent). Infinity is assigned for the result; Under-load (when the rounded quantity is too small to represent); Divide by zero, Uncertain results, Invalid. This flag is set for NAN.

The FPM needs a normal multiplier design block as a sub-block to facilitate the process of floating-point multiplication. There is immense literature on efficient multiplier design. The proposed architecture utilizes serial multiplier design [2] shown in Fig. 3. It comprises of an accumulator to store the multiplier, intermediate results and final product, control unit to carry on the process. The multiplier architecture is extended to accommodate 16-bit inputs to fit to the need of the proposed DCT architecture.

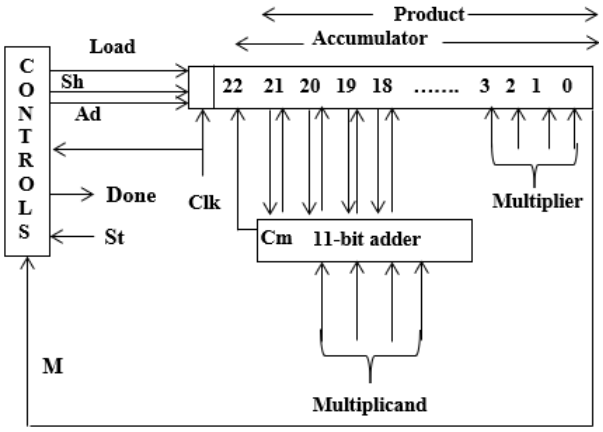

Figure. 3 Architecture of 11-bit serial multiplier using Shift-Add operations [2]

Multiplying by shifting and adding is all that is required here. As soon as a partial product is generated, it is added and saved in the accumulator. Adding two 11-bit values needs a multiplicand of 11 bits, a multiplier of 11 bits, and 22-bit result storage. The product register acts as an accumulator, adding up the incomplete products. If the multiplier's LSB bit is set to '1,' the multiplicand is pushed to the accumulator preceded by a right shift; if the multiplier's LSB bit is set to '0,' the product is only shifted. This iteration is repeated until the value reaches the 11th bit. There are several control signals that indicate when the process is complete: add, shift, load, start, and done.

The proposed architecture uses half precision IEEE 754 floating point standards, where 10-bits are allocated for mantissa (M), 5-bits for exponent (E) and 1-bit for sign (S) as shown in Eq. (1).

$$V = (-1)^s * (1 + M) * 2^b;\ \ b = E - 15 \quad (1)$$

Fig. 4 depicts the architectural design of floating-point multiplier unit. The architecture is modified to adjust with the required specification for the proposed design of 1D-DCT. The FPM block contains Sign bit calculator, Unsigned Adder-Subtractor, 11-bit multiplier, and Normalizer. The sign bit is computed using the rules of multiplication of signed numbers. When two numbers are multiplied, the result is a negative quantity when any of the

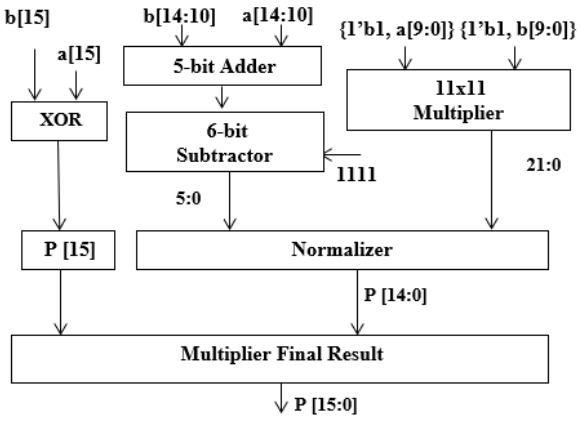

Figure. 4 Architecture of Floating-point multiplier unit

Table 1. Floating-point representation of scaling factors

Scaling factor	Decimal value	IEEE 754 Half precision representation
cos(3π/8)	0.3826	0011011000011111
sin(3π/8)	0.9238	0011101101100010
cos(π/16)	0.9807	0011101111010111
sin(π/16)	0.1950	0011001000111101
cos(3π/16)	0.8314	0011101010100111
sin(3π/16)	0.5555	0011100001110010

multiplied values is negative. When two numbers have the same polarity, the sign bit is always positive. A basic XOR gate will suffice the requirement and hence used in the construction of a sign bit computing unit.

The unsigned adder-subtractor: responsible of adding the first input's exponent to the other input's exponent and deducting the bias value (1b) from the summation output (i.e. e1 (Exponent) + e2 (Exponent) − Bias). Here, a 5-bit adder and a 6-bit Subtractor are utilised, and bias value is 15 (2^{k-1}-1, where k is number of exponent bits) for the half precision standards. Unsigned multiplier unit controls the process of unsigned multiplication of mantissas. The least significant 10 bits of each of the input's mantissa is supplied as an input this unit. One additional MSB bit with logic '1' is appended for both the inputs while supplying to suffice the demand of standard representation of floating-point number i.e., 1. Mantissa. The performance of the multiplier is chosen in such a way that it has no negative effect on the overall performance of the DCT design. It is necessary to normalize the result of the mantissa multiplication so that it begins with the number one to the left of the decimal point (a leading '1'). Considering that the inputs are normalized numbers, the partial product has the first bit (bit 21) set to one, effectively starting the resultant multiplication product with a decimal point. The result would yield 1 bit (MSB bit) from sign bit calculator unit, 5-bit (exponent bits) from unsigned adder-subtractor unit, and the rest 10 bits (Mantissa) through normalized serial multiplier block. Overall, the authors could multiply the two floating point integers using a shift-add operation rather than a multiplication. This significantly improves the performance of the DCT unit.

4. Architectural design of loeffler DCT

The name DCT comes from the fact that the N*N transform matrix Y are produced as a function of cosines. The DCT is comparable to DFT, and in terms of compression, the DCT outperforms the DFT. In the Eq. (2), the N-point DCT is defined as follows [4] for an input stream {x (n)}, n ϵ [0, N −1],

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

Where $\alpha(0)=1/\sqrt{2}$ and $\alpha(k)=1$ if $k \neq 0$;

The proposed work focuses on Loeffler 1D-DCT design [5], which entails 11 multiplications and 29 additions in the design. The proposed architectural design of 1D-DCT, Loeffler 1D-DCT is implemented using Floating point arithmetic to accomplish higher precision. The care has been taken to reduce the computationally intense operations such as multiplication operations are replaced by shift-add processes. These yield optimum results compared to most of the existing techniques in the literature. It has a few scaling factors that are expressed in terms of sine and cosine values. Floating point format is utilised to express the scaling factors in Eq. (2). Table 1 lists scaling factors in corresponding floating-point integers.

The proposed architectural design of 1D-DCT employing Floating point arithmetic units is shown in Fig. 5. The design utilizes the symmetric property of DCT basis function matrix. The architecture can be analysed in four stages for simplicity. During the first stage of procedure, the inputs are added and subtracted accordingly. Assume x0, x1, x2, x3, x4, x5, x6 and x7 as inputs to DCT block. The first stage results in the intermediate values S0, S1,…., S7, C1,…,C3 and further stages proceed as illustrated.

Stage 1:S0=x0+x7,S1=x1+x6,S2=x2+x5, S3=x4+x3
Stage 1: S4=x7-x0, S5=x6-x1, S6=x5-x2, S7=x4-x3
Stage 2:C0=S0+S3,C1=S2+S1,C2=S2-S1,C3=S3-S0
Stage 3: Y0=C0+C1

Some of the outputs need additional stage of computation whose data-path is depicted in the

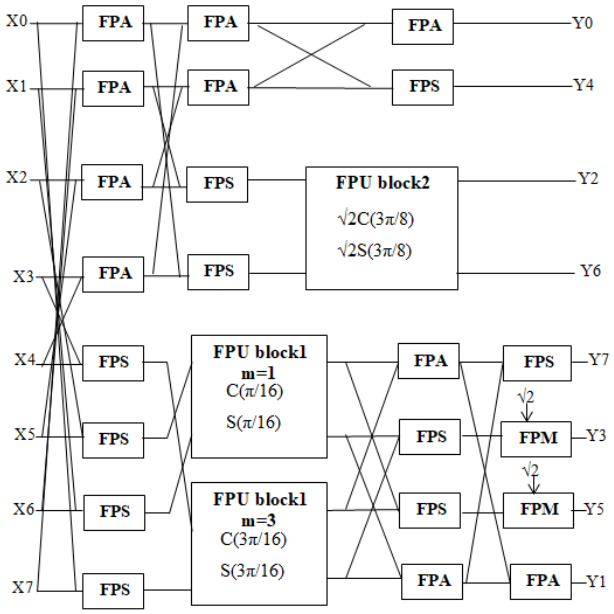

Figure. 5 Proposed Architectural design of Loeffler 1D DCT with FPU

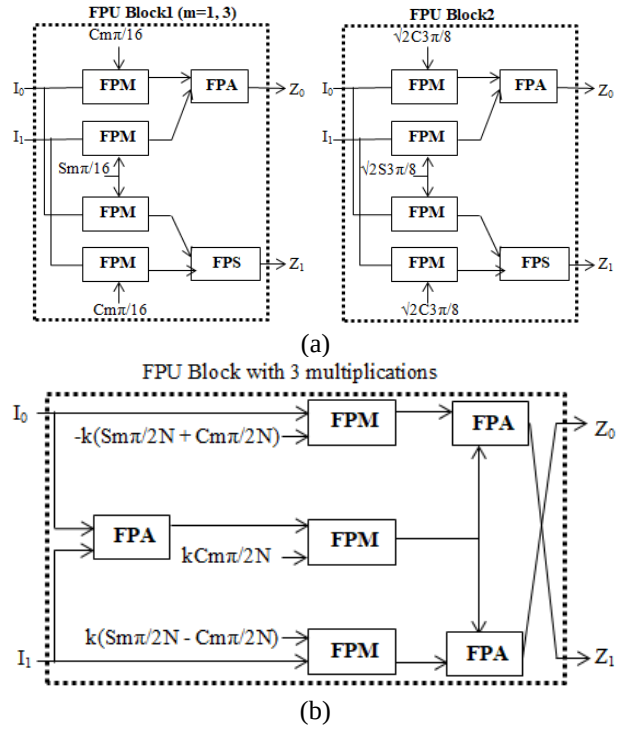

Figure. 6 Architectural design of: (a) FPU block used in 1D DCT design (method 1), and (b)FPU block with reduced number of multiplications (method 2)

architectural design of Loeffler 1D- DCT shown in Fig. 5. The proposed architecture follows 8X8 Loeffler 1D-DCT design with inclusion of Floating-point adders, subtractors and multipliers in order to achieve high precision output. Loeffler proposed in [5] to derive DCT coefficients in four phases, as seen in Fig. 5. The first stage is performed by four FPA and four FPS blocks, while the second stage is composed of two FPA, two FPS, and two FPU (multiplier, adder, and Subtractor) blocks. Each FPU block consumes 4 FPM but can be decreased to 3 FPM using constant adjustments. The third stage is made up of three FPA, three FPS, and one FPU block. The fourth level stage employs two FPM blocks to accomplish $\sqrt{2}$ multiplication, along with one FPA and one FPS. Eqs. (3) and (4) defines the FPU block shown in architecture. The two inputs are considered as I0, I1 and outputs are Z0, Z1. The same equations are applied to construct FPU blocks 1 and 2.

$$Z0 = I0.cos\left(\frac{m\pi}{2N}\right) + I1.sin\left(\frac{m\pi}{2N}\right) \quad (3)$$

$$Z1 = -I0.sin\left(\frac{m\pi}{2N}\right) + I1.cos\left(\frac{m\pi}{2N}\right) \quad (4)$$

Where m=1 or 3; N=8. Fig. 6 (a) illustrates the internal structure of an FPU block. The structure follows Eqs. (3) and (4) using floating point blocks. In the architecture, trigonometric functions cosine and sine are represented as C and S respectively. $\cos(m\pi/16) = C_m\pi/16$; m=1 or 3; $\sin(m\pi/16) = S_m\pi/16$; m=1 or 3. The FPU block can be further optimised with only 3 multiplications and 3 additions requirement as suggested by Loeffler modified fast DCT computation algorithm. The Eqs. (5) and (6) are the modified versions of Eqs. (3) and (4) respectively.

$$Z0 = \left(-ksin\left(\frac{m\pi}{2N}\right) + kcos\left(\frac{m\pi}{2N}\right)\right) I1 + kcos\left(\frac{m\pi}{2N}\right)(I0 + I1) \quad (5)$$

$$Z1 = \left(-ksin\left(\frac{m\pi}{2N}\right) - kcos\left(\frac{m\pi}{2N}\right)\right) I0 + kcos\left(\frac{m\pi}{2N}\right)(I0 + I1) \quad (6)$$

The modified block diagram is shown in Fig. 6b. With this, the proposed Loeffler 1D-DCT design requires 11 multiplications and 29 additions as compared to 14 and 26 [5]. The inverse DCT employs the same algorithmic structure as the forward DCT, but in a reverse context. The architecture for inverse transform is not shown to avoid the duplication.

5. 2D-DCT using N 1D-DCT

The two-dimensional DCT sequence Y_{mn}; where m and n ranges from 0 to N-1is given in Eq. (7) for a given data sequence xij where i and j ranges from 0 to N-1 [4,13].

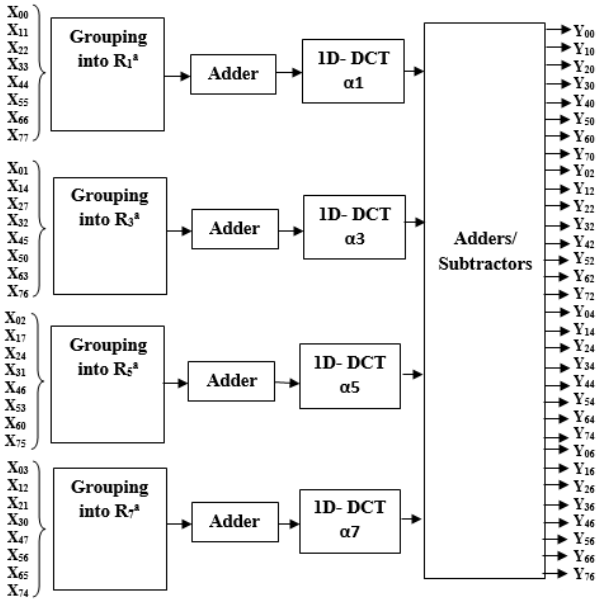

Figure. 7 Datapath of the 2D-DCT architecture (even)

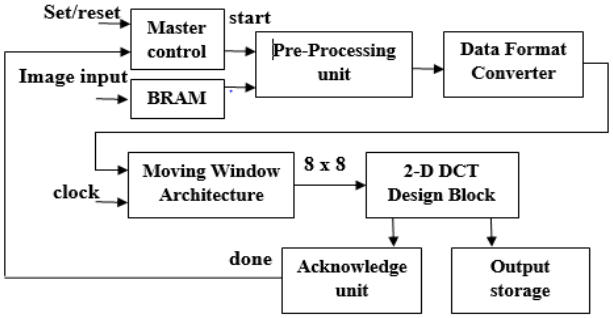

Figure. 8 Control Logic of proposed 2D-DCT architecture

$$Ymn = \frac{4}{N^2}u(m)u(n) * \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{ij} cos\frac{(2i+1)m}{2N}\pi \;\; cos\frac{(2i+1)n}{2N}\pi$$

$$where\ u(n) = u(m) = \begin{cases} \frac{1}{\sqrt{2}}, & m = n = 0 \\ 1, & Otherwise \end{cases} \quad (7)$$

The proposed architectural design of 1D-DCT is applied for realizing $\alpha_{\rho l}$ Eq. (8).

$$\alpha_{\rho l} = \sum_{i=0}^{N-1}\left(x_{ij(\rho,a)} + x_{ij(\rho,b)}\right)\cos\left(\frac{(2i+1)\omega}{2N}\right)\pi \quad (8)$$

$\beta_{\rho l}$ is computed for odd value of n using (9). The quotient $q_{\rho i}$ is computed as (ρi + (ρ-1)/2)/N, to adjust the sign of $\beta_{\rho l}$.

$$\beta_{\rho l} = \sum_{i=0}^{N-1}(-1)^{q_{\rho i}}\left(x_{ij(\rho,a)} + x_{ij(\rho,b)}\right) cos\left(\frac{(2i+1)\omega}{2N}\right)\pi \quad (9)$$

Where ω is m±nρ for ω in the range 0 to N-1. If m±nρ is out of the range, then new value ω is computed using ω mod 2N. The cosine function of trigonometry dictates the sign convention. Using 1-D DCT, α_ρl and β_ρl are computed using Eqs. (10a), (10b), and then they are added and subtracted in the appropriate manner to compute the final output, Y_{mn}, as illustrated in the suggested architecture of 2-D DCT shown in Fig. 7.

$$y_{mn} = \frac{1}{2}\sum_{\rho=1\ (even)}^{N-1}\left(\alpha_{\rho l+} + \alpha_{\rho l-}\right) \quad (10a)$$

$$y_{mn} = \frac{1}{2}\sum_{\rho=1\ (odd)}^{N-1}\left(\beta_{\rho l+} + \beta_{\rho l-}\right) \quad (10b)$$

The postfix +, – to α_ρl and β_ρl indicates ω=m+nρ and ω=m-nρ respectively. Several architectures for designing 2D-DCT using 1D-DCT are available in the literature. The traditional method necessitates 2N 1-D DCT blocks that employ the row-column decomposition method. The architecture suggested in [13] is used here, which employs only N 1D-DCT blocks. Fig. 7 depicts the architecture of 2D-DCT (the even part), whereas the odd part is similar to the even except the input groups and minor sign changes.

The 2D-DCT architecture is designed for N=8. The image dataset for higher dimensions is split into the multiple blocks of 8X8 and fed into the 2D-DCT architecture. This process is carried out by Moving window architecture. Fig. 8 depicts the arrangement and control logic for processing an image input to the proposed design. For the functional verification purpose, the 2D-DCT output coefficients are processed and fed as an input the inverse transformation of 2D-DCT. The structure of the inverse DCT remains same as the DCT is a separable and orthogonal transform. The results are matched with original input data, which are discussed in the next section.

6. Results and discussions

The Authors have made use of Xilinx Synthesis Tool (XST) to design and verify the proposed architecture. The hardware implementation of the proposed DCT design is carried out on 28nm, 40nm, 65nm technology node devices of Vertex, Kintex FPGA device family. The Chip Scope pro tool suite, which offers a variety of modules that can be added to the HDL model to capture inputs and outputs directly from the FPGA hardware, is used since the number of input and output ports is vast. An effort has

Table 2. Precision comparison and RMSD computation

Sample Input sequence	1D- DCT output		Precision
	Manual Calculation	Proposed architecture	
9	16.2635	16.26	0.0035
4	-2.9341	-2.93	-0.0041
1	3.537	3.53	0.007
2	5.8307	5.82	0.0107
8	2.1213	2.12	0.0013
7	-0.5625	-0.56	-0.0025
8	-0.6997	-0.69	-0.0097
7	1.7542	1.75	0.0042
			RMSD = 0.0062

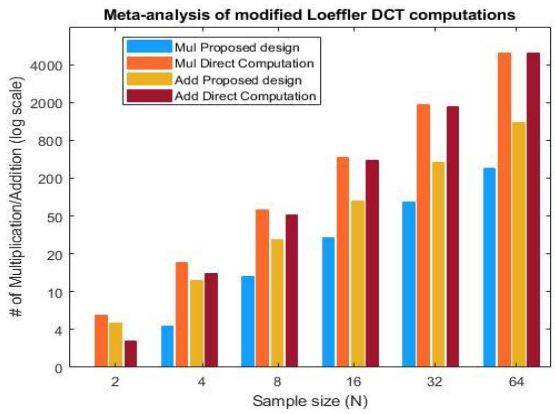

Figure. 9 Analysis of computations involved in the conventional and proposed method
*Multipliers are replaced by shift-add operation block

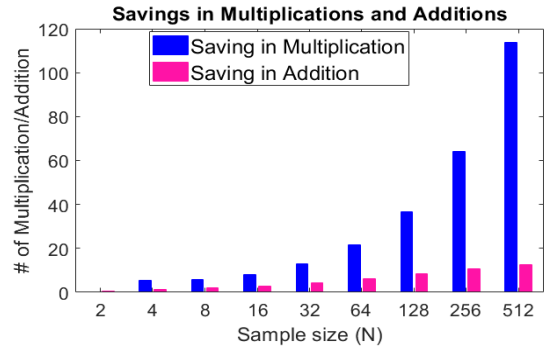

Figure. 10 Saving achieved in number of computations

been made to make it easier to provide input and to display output via VIO (Virtual Input/Output) modules. This tool can monitor and influence the design in real time. The performance metric considered for the evaluation of precision of DCT is root mean square deviation (RMSD). Eq. (11) defines RMSD of the output of DCT architecture to the manually calculated output values.

$$RMSD = \sqrt{\frac{\sum_{i=1}^{N} Ei^2}{N}} \quad (11)$$

Where $Ei = (Op - Om)$; Op is output of proposed DCT; Om is output computed manually. N is input sequence length. For experimentation, input length of 8 is selected. The RMSD resulted for the experiment is 0.0062 as shown in Table 2, which is reasonable considering the complexity involved in the computation of DCT coefficients, making the proposed architecture high precision.

The process of reading and writing the inputs and outputs to the floating-point architectures (FPU) are quite complex and time consuming. The given set of input sequence is to be initially converted to half precision IEEE 754 standard format and the same is to be done while reading the outputs. When considering eight 16-bit inputs, a total of 8*16-bit = 128-bit input and output must be handled. This is not possible with the FPGAs where usually 32 input pins and 32 outputs pins are provided. The state-of-the-art tool is employed for realizing the intended design on the FPGA. All 128 bits of input is provided virtually using Chipscope pro VIO and the outputs are displayed virtually on the monitor through the FPGA.

Fig. 9 provides the analysis on number of computations required for the conventional and proposed method. As the number of computations in conventional method increases exponentially with respect to increase in the number of samples, the logarithmic scale is considered across the y-axis.

The Eqs. (12) and (13) are used to derive the values of the respective parameters for proposed 1D-DCT.

$$\text{No. of Multiplication, } M1 = \left(\frac{N}{2}\right) log_2^N - 1 \quad (12)$$

$$\text{No. of Addition, } A1 = (N) log_2^N + \left(\frac{N}{4}\right)^2 + 1 \quad (13)$$

The plot proves that proposed method performance increases with increases in the sample size. Fig. 10 shows the saving achieved in the number of multiplications and additions using the proposed method, which is computed by taking the ratio of proposed to conventional method. The multiplication is computationally intense compared to additions and hence the results are favourable.

Fig. 11 shows the obtained RTL schematic of proposed 1D-DCT (method 1) for N=8 from Xilinx synthesis tool. The RTL design overview of 1D-DCT shows that 11 FPMs and 29 FPS/FPAs are employed in the proposed 1D-DCT architecture. Table 3 illustrates the comparative analysis of the proposed DCT with the existing method. The loeffler model uses the least multiplier. The multipliers are replaced by Shift-add in order to achieve the favourable results.

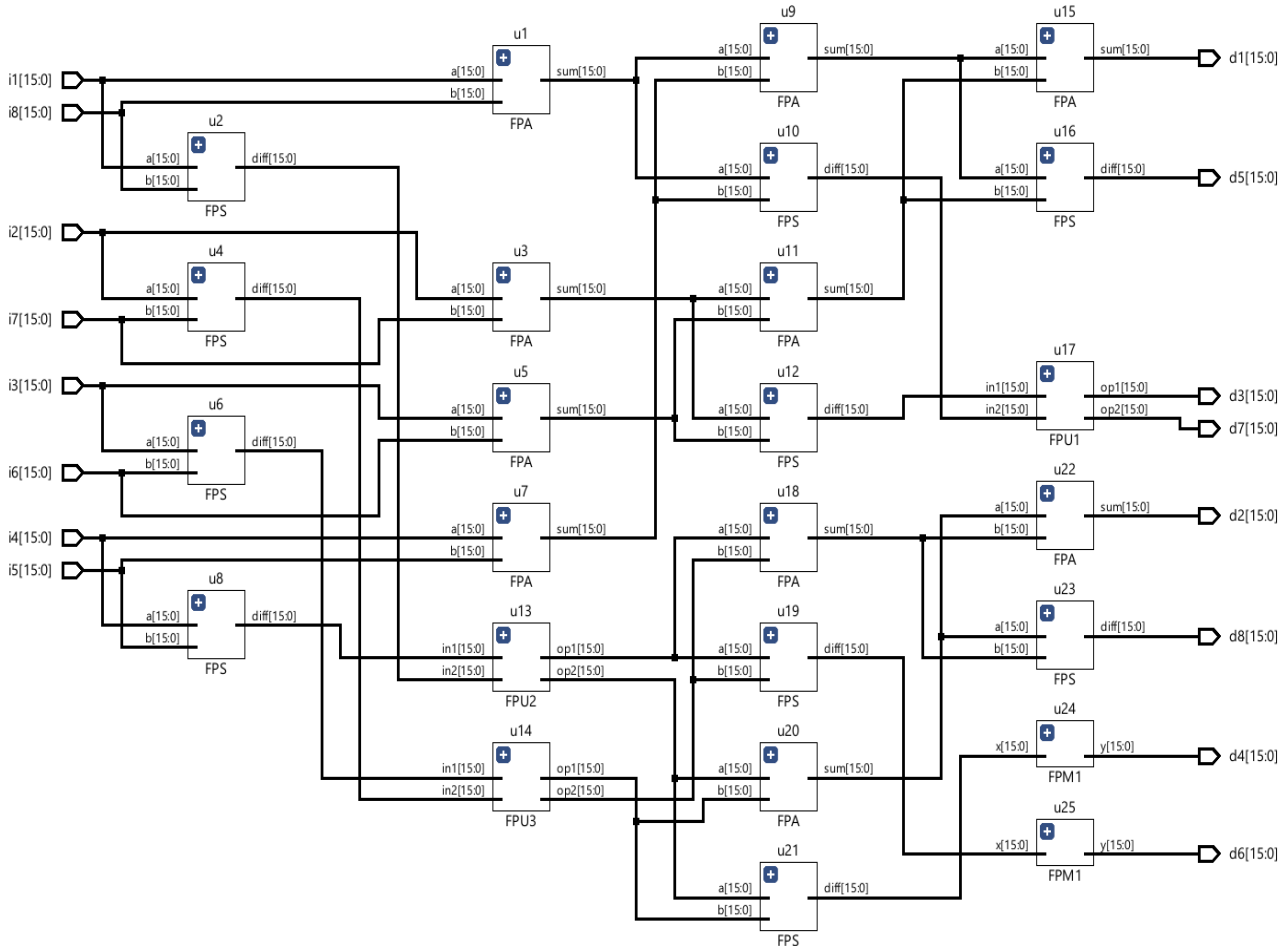

Figure. 11 Register Transfer Logic (RTL) netlist schematic of proposed 1D-DCT

Table 3: Comparative analysis of proposed DCT (N=8) with existing models

Reference	**Transform**	**Computation method used**	**Technology**	**Hardware Utilization**	**Computation time (ns)**	**Max. Frequency (MHz)**	**RMSD**	**PSNR**
Chung [7]	2D-DCT	Recursive CORDIC	180nm	8400	-	100	0.06	31.54
Lee [16]	1D-DCT	CORDIC	130nm	22400	-	-	-	31.45
Sun [18]	2D-DCT	CORDIC-Loeffler DCT	130nm	44920	15.08	-	-	31.92
Meher [25]	2D Int DCT	Matrix Vector Prod. Unit (MVPU)	90nm	37750	25.6	380	-	-
P. Garg [26]	1D-DCT	Approx. Floating-point adder	28nm	6446	28.05	-	0.04	27.92
Singhadia [27]	2D-DCT	Transform Symmetricity	40nm	39625	-	149.35	-	-
Proposed design	1D and 2D-DCT	Loeffler DCT with FPU and Shift-add multiplication	28nm	8731 (1D) 34924(2D)	18.3	496.2	0.0064	37.24*
			40nm	8778(1D) 35712(2D)	22.7	410		

*50% DCT coefficients unitized for reconstruction; '- ': Values not mentioned in the literature

The analysis shows that the balance in the trade-off factors such as Hardware utilisation, Time, MSE and PSNR is achieved. Without utilizing more hardware, the proposed design could achieve higher accuracy with optimum computational delay.

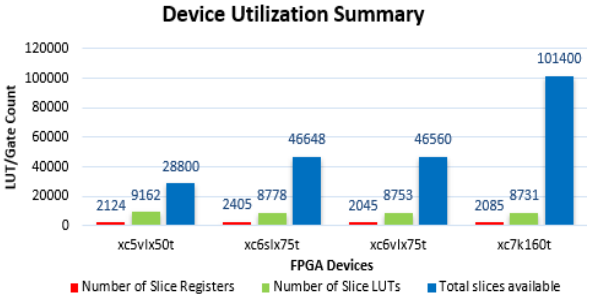

Figure. 12 Device utilization summary of the 1D-DCT design implemented on various FPGA devices

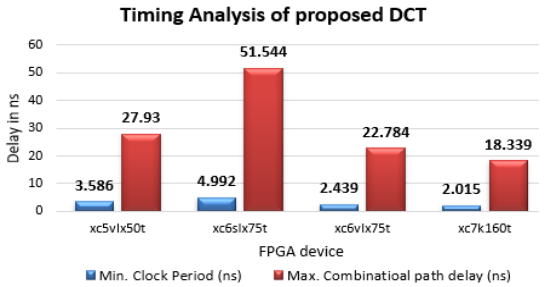

Figure. 13 Timing analysis of the 1D-DCT design implemented on various FPGA devices

Table 4. Computational complexity comparison of proposed 1D-DCT (N=8) with existing models

Reference	Multipliers	Adders	Shift	Storage units
Chung [7]	Recursive algorithm	108	96	64
Lee [16]	CORDIC	120	92	24
Sun [18]	CORDIC	104	82	12
Sungwook [23]	NEDA	32	88	40
Singhadia [27]	Transform Symmetry	800	416	0
Proposed design	FPU with Shift-add	117	66	8

Figure. 14 Reconstructed images by feeding the DCT coefficients to the inverse DCT: (a) retaining 50% of the coefficients, and (b) retaining 25% of the coefficients

The proposed architecture is designed using HDL and is implemented on various FPGA hardware with varied technology nodes to evaluate the performance. The device utilization summary is shown in Fig. 12 depicts that the number of slice registers and LUTs utilized by the design on the FPGA devices- 65nm (xc5vlx50t), 45nm (xc6slx75t), 40nm (xc6vlx75t) and 28nm (xc7k160t). The average gate count (LUT) consumption is 8.5K LUTs.

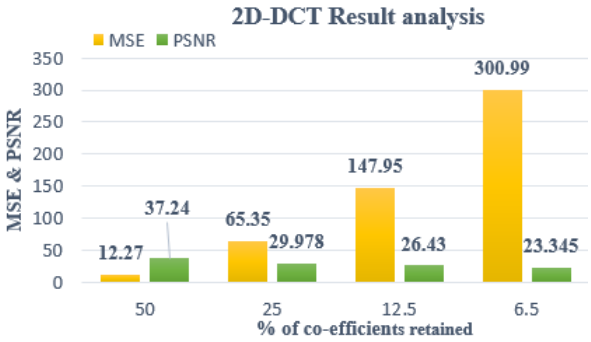

Figure. 15 Comparison of MSE and PSNR

The timing analysis of the proposed DCT is carried out on various technology node FPGAs and is shown in Fig. 13. It reveals that the maximum combinational path delay resulted in the range of 18ns to 51ns. The maximum operating frequency of the clock obtained is 496MHz on xc7k160t FPGA.

Table 4 displays the computational complexity of numerous existing approaches that have been studied in the literature. The number of multipliers and adders needed for each of the architecture is tabulated. It is evident that the proposed architecture uses optimum number of multipliers and adders in comparison with existing efficient techniques. Fig. 14 Shows the sample input image data compared with the reconstructed image output by retaining 50% and 25% DCT coefficients. The results comprehend the functionality of the proposed architecture.

Mean square error and PSNR are computed for the 2D-DCT design with varied percentage of retention of the DCT coefficients is provided in Fig. 15.

Although the proposed architecture employs a similar design structure to the Loeffler-based DCT design presented in the literature, the proposed design accomplishes greater PSNR of 37.24dB as compared with 31.92dB in CORDIC loeffler based DCT [8], without compromising much on speed and area. Utilizing floating point arithmetic units contributes to its high precision construction. Using shift-add operations rather than multipliers improves hardware utilization and reduces the combination path delay.

7. Conclusion

The work presents a high precision 1D-DCT architectures employing floating point arithmetic unit. The proposed model implements 1D-DCT using 8X8 Loeffler architecture which utilizes 14 multiplications-26 additions (method 1) and 11 multiplications-29 additions (method 2). The floating-point arithmetic unit consists of serial

multiplier blocks, which are made up of shift-add operations rather than true multiplication operations. The proposed method uses 117 additions, 66 shifts and 8 storage units of 16-bits, which shows improvements in comparison with existing models. The Computational time and hardware utilization are found to be optimum considering the computationally intense floating-point units. The proposed architecture is implemented on various FPGA devices discussed. Because of the high precision computation, which enables accuracy up to three decimal points, the Root Mean square error deviation is negligibly small. The experimental results demonstrate that the proposed model significantly enhanced the required number of computations and precision. In addition, the PSNR has improved by 17%, and the maximum clock frequency attained is 496MHz. The proposed design strikes a balance between trade-off parameters such as area, speed, and precision. The 2D-DCT is tested using a sample image and results of MSE and PSNR are found to be favourable. Further, the proposed architecture can be extended to higher lengths of input sequence.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

The First author handled the responsibility of the conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization. The supervision and project administration have been done by Second author.

Acknowledgments

The research work was supported by the Research Centre of Department of ECE, SDM College of Engineering and Technology, Dharwad.

References

[1] U. Meyer-Baese, "Digital Signal Processing with Field Programmable Gate Arrays", *Springer Berlin, Heidelberg*, 2014.

[2] C. Roth, L. John, and B. Lee, "Digital Systems Design Using Verilog", *Cengage Learning, Boston*, 2015.

[3] C. Roth and L. Kinney, "Fundamentals of Logic Design", *Cengage Learning, Stamford*, 2014.

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

[5] C. Loeffler, A. Ligtenberg, and G. Moschytz, "Practical fast 1-D DCT algorithms with 11 multiplications", In: *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, Glasgow, UK, pp. 988-991, 1989.

[6] S. Anjana, C. Pradeep, and P. Samuel, "Synthesize of High-Speed Floating-point Multipliers Based on Vedic Mathematics", *Procedia Computer Science*, Vol. 46, pp. 1294–1302, 2015.

[7] R. Chung, C. Chen, C. Chen, P. Abu, and S. Chen, "VLSI Implementation of a Cost-Efficient Loeffler DCT Algorithm with Recursive CORDIC for DCT-Based Encoder", *Electronics*, Vol. 10, No. 7, p. 862, 2021.

[8] M. Deivakani, S. Kumar, N. Kumar, E. Raj, and V. Ramakrishna, "VLSI Implementation of Discrete Cosine Transform Approximation Recursive Algorithm", *Journal of Physics: Conference Series*, Vol. 1817, No. 1, p. 012017, 2021.

[9] M. Masera, G. Masera, and M. Martina, "An Area-Efficient Variable-Size Fixed-Point DCT Architecture for HEVC Encoding", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 30, No. 1, pp. 232–242, 2020.

[10] M. Thiruveni and D. Shanthi, "Efficient VLSI Architecture for 16-Point Discrete Cosine Transform", In: *Proc. of the National Academy of Sciences, India Section A: Physical Sciences*, Vol. 90, No. 1, pp. 27–37, 2018.

[11] M. Basiri and N. M. Sk, "High Performance Integer DCT Architectures for HEVC", In: *Proc. of 30th International Conference on VLSI Design and 16th International Conference on Embedded Systems (VLSID)*, Hyderabad, India, pp. 121-126, 2017.

[12] N. Cho and S. 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.

[13] S. Shirakol and S. Kerur, "Architectural Design and Optimization of Distributed Arithmetic based 2-D Discrete Cosine Transform", *ICTACT Journal on Microelectronics*, Vol. 08, No. 01, pp. 1275–1282, 2022.

[14] S. Inguva and J. Seventiline, "LH-CORDIC: Low Power FPGA Based Implementation of CORDIC Architecture", *International Journal of Intelligent Engineering and Systems*, Vol. 12, No. 2, pp. 305–314, 2019, doi: 10.22266/ijies2019.0430.30.

[15] J. Zhang, P. Chow, and H. Liu, "FPGA Implementation of low-power and high-PSNR DCT/IDCT architecture based on adaptive

recoding CORDIC", In: *Proc. of International Conference on Field Programmable Technology (FPT)*, Queenstown, New Zealand, pp. 128-135, 2015.

[16] M. Lee, J. Yoon, and J. Park, "Reconfigurable CORDIC-Based Low-Power DCT Architecture Based on Data Priority", *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, Vol. 22, No. 5, pp. 1060–1068, 2014.

[17] N. M. Zabidi and A. A. Rahman, "VLSI Design of a Fast Pipelined 8x8 Discrete Cosine Transform", *International Journal of Electrical and Computer Engineering (IJECE)*, Vol. 7, No. 3, pp. 1430-1435, 2017.

[18] C. Sun, S. Ruan, B. Heyne, and J. Goetze, "Low-power and high-quality CORDIC-based Loeffler DCT for signal processing", *IET Circuits, Devices & Systems*, Vol. 1, No. 6, pp. 453-461, 2007.

[19] Z. Wu, J. Sha, Z. Wang, L. Li, and M. Gao, "An improved scaled DCT architecture", *IEEE Transactions on Consumer Electronics*, Vol. 55, No. 2, pp. 685–689, 2009.

[20] S. Yu and E. E. Swartziander, "DCT implementation with distributed arithmetic", *IEEE Transactions on Computers*, Vol. 50, No. 9, pp. 985–991, 2001.

[21] A. Akman and S. Cekli, "Efficient 2D DCT architecture based on approximate compressors for image compression with HEVC intra-prediction", *Journal of Real-Time Image Processing*", Vol. 20, No. 22, p. 1861, 2023.

[22] V. S. B, V. Solanki, and A. D. Darji, "Design of Hardware Efficient Approximate DCT Architecture", In: *Proc. of 36th International Conference on VLSI Design and 22nd International Conference on Embedded Systems (VLSID)*, Hyderabad, India, pp. 145-150, 2023.

[23] M. Barbareschi, S. Barone, A. Bosio, J. Han, and M. Traiola, "A Genetic-algorithm-based approach to the Design of DCT Hardware Accelerators", *ACM Journal on Emerging Technologies in Computing Syst*ems, Vol. 18, No. 3, pp. 1-25, 2022.

[24] M. S. Madni and C. Vijaya, "Hand Gesture Recognition Using Semi Vectorial Multilevel Segmentation Method with Improved ReliefF Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 447-457, 2021, doi: 10.22266/ijies2021.0630.37.

[25] P. K. Meher, S. Lam, T. Srikanthan, D. H. Kim, and S. Y. Park, "Area-Time Efficient Two-Dimensional Reconfigurable Integer DCT Architecture for HEVC", *Electronics*, Vol. 10, No. 5, p. 603, 2021.

[26] P. Garg and K. Suneja, "Hardware design of high speed 1-D DCT module using approximate floating-point adder", In: *Proc. of 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, pp. 623-625, 2020.

[27] A. Singhadia, M. Mamillapalli, and I. Chakrabarti, "Hardware-Efficient 2D-DCT/IDCT Architecture for Portable HEVC-Compliant Devices", *IEEE Transactions on Consumer Electronics*, Vol. 66, No. 3, pp. 203-212, 2020.

[28] S. K. Shirakol, V. Hiremath, and S. S. Kerur, "FPGA-Based Implementation of Digital Filters for Image Denoising", In: *Proc. of Smart Sensors Measurements and Instrumentation, Lecture Notes in Electrical Engineering*, Vol 750, No. 1, pp. 155–166, 2021.