A New Algorithm to Derive Hardware Efficient Integer Discrete Cosine Transform for HEVC

Boyu Qin and Jiajia Chen*
Nanjing University of Aeronautics and Astronautics, Nanjing, China
boyu.qin@outlook.com
*Corresponding author email: jiajia_chen@nuaa.edu.cn

***Abstract*—HEVC (High Efficiency Video Coding) is a popular video coding standard. With the rapid development of multimedia technology, people pursue not only the high-definition of video, but also the portability and miniaturization of devices, which brings great challenges to the low-power design of video encoding chips. To provide solutions to address both compression quality and hardware efficiency, a measure for evaluating the hardware cost in integer DCT unit of HEVC is proposed in this paper, which helps to determine the most hardware efficient transform matrix. In addition, Genetic Algorithm is used to solve the multi-objective optimization to derive the solution for better coding performance. Experiments show that the transform matrix derived and its hardware implementation has advantages in both hardware cost and coding performance. Compared with the most competitive methods in recent years, the hardware cost of the proposed method has been successfully reduced by at least 15.35% of chip area and 4.91% of power consumption.**

Keywords—discrete cosine transform, High Efficiency Video Coding, video coding

I. INTRODUCTION

Discrete cosine transform (DCT) is an important operation in digital signal processing [1]. It is widely used for image and video compression because of its decorrelation and energy compaction properties. Integer discrete cosine transform (Int-DCT) is one of the approximations of exact DCT to achieve a lower computing cost and eliminate drifting error [2], which makes it commonly employed in recent video coding standards, such as H.264/Advanced Video Coding (AVC) [3] and high efficiency video coding (HEVC) [4]. The Int-DCT matrices of size 4×4 , 8×8 , 16×16 and 32×32 are applied for two-dimensional transforms in the context of block-based motion-compensated video compression in HEVC standard [5].

There are some Int-DCT implementations proposed in the literature by using different methods to determine Int-DCT coefficients. A simplified Int-DCT matrix was proposed in [6] by only using 0, 1, and –1 as coefficients, which can be implemented by adders. In [7], Cintra proposed an approximation method based on *m*-th order dyadic rational approximating function. This method can adapt to different approximation precision by setting different *m*. A number of Int-DCT implementations are proposed by optimizing the computing process. Fong *et.al* proposed RICT in [8] which has recursive structure that an order-2N transform can be derived from an order-2(*N*–1) transform. Another representative method was proposed in [9] to truncate some least significant bits, most significant bits, and zero columns to achieve a lower hardware cost. An area- and power-efficient DCT architecture that based on time-multiplexing reconfigurable multipliers and sporadic logarithmic shifters was proposed in [10]. In [10], Genetic Algorithm (GA) was applied to search out the most efficient double base number system (DBNS) for each coefficient in Int-DCT matrix. Although GA can efficiently perform global search, it may not converge to the global optimal solution. In this paper, we propose a new measure to evaluate hardware cost. We add constraint to the DBNS representations to reduce the search space. Because of the reduced search space, the Exhaustive Search is a better choice to ensure the global optimal solution for the lowest hardware cost.

In recent years, a number of new Int-DCT are proposed by developing Int-DCT matrices with good trade-off between hardware cost and coding performance. Hardware Efficient Int-DCT (HEICT) [11] is one of them, which uses a weighted sum approach to figure out this multi-objective problem. Although these weights are adjustable for different application scenarios, how to determine specific weightage for each property is yet to be discussed. In this paper, we propose a hardware optimization method and a multi-objective optimization method which include two new contributions:

1. We optimize the Int-DCT implementation by a new method to find out the most efficient double base number system (DBNS) solution for the lowest hardware cost.
2. We solve the Int-DCT coefficients approximation as a multi-objective optimization problem by using GA.

This paper is organized as follows. Section II presents the related works adopted in our method. Section III presents the proposed method optimizing Int-DCT implementation and the multi-objective optimized solution that achieves a good trade-off between hardware consumption and coding performance. In Section IV, the proposed algorithm is compared with four state-of-art methods, followed by the conclusions in Section V.

II. RELATED WORKS ADOPTED IN OUR METHOD

The $N\times N$ 2-dimensional DCT is required in HEVC, which is computed by applying 1-D transforms in the horizontal and vertical directions [12]. 1-D DCT can be expressed as [5]:

$$w_i = \sum_{j=0}^{N-1} u_j c_{ij} \quad (1)$$

where $i = 0,\ldots,N-1$. u_j are the input samples and w_i are the transformed coefficients. Elements c_{ij} of the DCT transform matrix **C** are defined as

$$c_{ij} = \frac{A}{\sqrt{N}} \cos\left[\frac{\pi}{N}\left(j+\frac{1}{2}\right)i\right] \quad (2)$$

where $i, j = 0,\ldots,N-1$. A is equal to 1 and $2^{1/2}$ for $i = 0$ and $i > 0$ respectively.

In order to achieve a lower hardware cost and eliminate drifting error [2], the finite precision Int-DCT matrix **d** is applied in HEVC, rather than the original DCT matrix.

The Int-DCT in HEVC can be implemented as a matrix multiplication between the Int-DCT matrix and the residual signal. To reduce the hardware cost, the reconfigurable multiplier (RM) based method proposed in [13] is used in our design to implement the matrix multiplication, which is shown in Fig. 1. Each row represents an RM that corresponds to a row of coefficients in Int-DCT matrix. The *i*-th RM will be configured as **d**(*i*,*j*) at the *j*-th cycle, and then multiples with *x*(*j*). The multiplications between the *i*-th coefficients in Int-DCT matrix and the residual signal will be accumulated to get *y*(*i*) after *N* cycles.

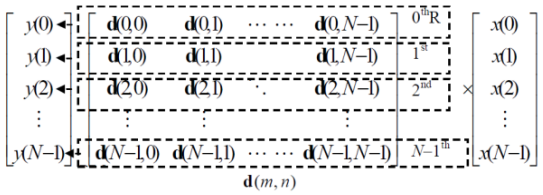

Fig. 1. The RM based method to implement the matrix multiplication

Based on the works of [14], double base number system (DBNS) has been proven to be efficient to implement add-shift digital circuits. The coefficients in Int-DCT matrix **d** using double base number system can be presented as

$$n = \sum_{i=1}^{l} s_i p^{a_i} q^{b_i} \qquad s_i \in \{0,1\}, a_i, b_i \geq 0 \quad (3)$$

where p and q are reciprocal integers. Each positive integer n has a number of different DBNS representations. To further reduce the complexity, the DBNS representations are limited by using $l = 2$ in our design. In such case, (3) can be simplified to

$$n = p^{a_1} q^{b_1} + p^{a_2} q^{b_2} \qquad a_1, b_1, a_2, b_2 \geq 0 \quad (4)$$

In DBNS applications, 2 and 3 are usually chosen as base numbers, i.e. $p,q \in \{2,3\}$. However, some integers cannot be represented as (4) by limiting $l = 2$ while $p,q \in \{2,3\}$. Therefore, we add 5 and 7 as base numbers in our design. Due to the simplicity of implemented 2^a multiplications as left-shift operations in digital circuits, we constrain $p = 2, q \in \{3,5,7\}$. The DBNS that we apply is

$$n = 2^{a_1} q^{b_1} + 2^{a_2} q^{b_2} \qquad a_1, b_1, a_2, b_2 \geq 0,\ q \in \{3,5,7\} \quad (5)$$

Our verification experiments show that every positive integer in Int-DCT matrix can be presented as (5).

Fig. 2 shows the hardware architecture used in our design. When the input data is x, the Pre-processing Block will generate $q^b x$ i.e. x, $3x$, $5x$, $7x$…. Because the term number of DBNS representations is two, $q^{b_1} x$ and $q^{b_2} x$ will be multiplied with 2^{a_1} and 2^{a_2} by using direct left-shift operations in Shift Block. Therefore, RegA and RegB are corresponding to $2^{a_1} q^{b_1} x$ and $2^{a_2} q^{b_2} x$. The sum of RegA and RegB represents a multiplication between one of Int-DCT matrix elements and the input data.

III. The Proposed Method

A. The Proposed Hardware Optimization Method

The hardware architecture of one of the RMs is shown in Fig. 3. As an example, this RM is one of the sixteen RMs in 16-point Int-DCT implementation. Due to the unique number property of Int-DCT [5], the number of unique coefficients on each odd row is the same, and there are 8 unique coefficients in each odd row. Because each row of the matrix corresponds to one RM, these 8 constants will be configured for the RM according to the control logic. This RM will be configured as multiplications between the input and 90, 87, 80, 70, 57, 43, 26, 9, which are truncated to 8-bit. Other types of RMs in 16-point Int-DCT implementations have 4 or 2 constants, corresponding to different rows of the Int-DCT matrix.

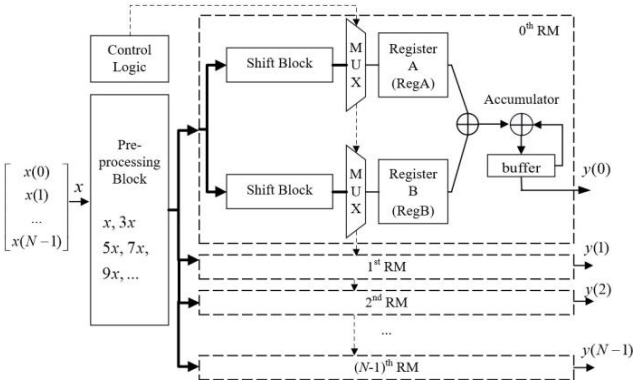

Fig. 2. The Int-DCT implementation based on RM

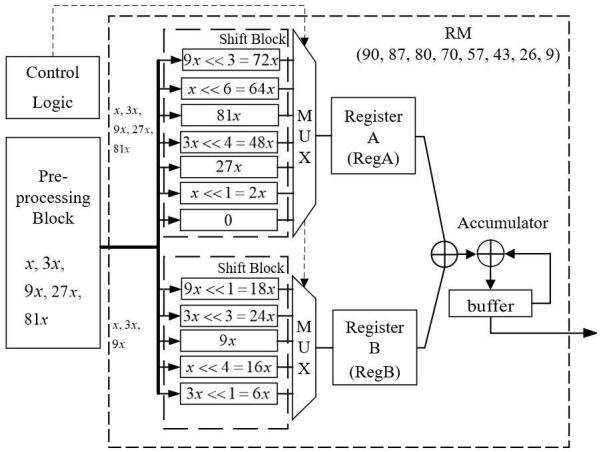

Fig. 3. Hardware architecture of RM

One problem when implementing RM is that the DBNS representations of the same integer are not unique. For example, 90 has eight different DBNS representations as follows, where $<<x$ represents left-shift by x-bit.

$$\begin{aligned}
90 &= 10 + 80 = 5 << 1 + 5 << 4 \\
90 &= 80 + 10 = 5 << 4 + 5 << 1 \\
90 &= 18 + 72 = 9 << 1 + 9 << 3 \\
90 &= 72 + 18 = 9 << 3 + 9 << 1 \\
90 &= 36 + 54 = 9 << 2 + 27 << 1 \\
90 &= 9 + 81 = 9 << 0 + 81 << 0 \\
90 &= 54 + 36 = 27 << 1 + 9 << 2 \\
90 &= 81 + 9 = 81 << 0 + 9 << 0
\end{aligned} \tag{6}$$

Different DBNS representations lead to different values of RegA and RegB, which influences the hardware architecture of Shift Block in Fig. 3. Taking the RM in Fig. 3 for example, it has eight constants while each constant has 5 to 10 different DBNS representations. Thus, this RM has a total of more than 5^8 different implementations. In this paper, we propose that the more DBNS terms shared between different constants, the fewer state values RegA and RegB need to be configured. The state values of RegA correspond to all the first terms of DBNS representations that are used in the RM, while the state values of RegB correspond to the second terms. Therefore, if one DBNS term can be shared between multiple DBNS representations, the number of different terms will be reduced. In such case, RegA and RegB require fewer state values, leading to a simplified Shift Block. As for the complexity of the entire RM, it is dependent on multiple factors:

1. The number of state values of RegA and RegB: The more shared state values leads to the lower hardware cost.

2. The number of $q^b x$. If one of the $q^b x$ can be commonly used in different DBNS representations, the hardware cost of Pre-processing Block will be reduced.

3. Bit-width of RegA and RegB: $q^b x$ have different bit-width after Pre-processing Block and different left-shift bits. Thus, the bit-width of RegA and RegB need to be determined by the maximum bit-width of all state values, which also influence the bit-width of the adder and the accumulator.

To address these factors, we propose a measure to evaluate the hardware cost of different DBNS representations as follows:

$$Bc = q^b x \text{ bit-width} + \text{left-shift bits} \tag{7}$$

With (7), the sum of Bc of all state values in an RM is defined as a measure of hardware cost of total RM. This measure is an efficient, because a specific state value has its lowest Bc only when both the bit-width of $q^b x$ and left-shift bits are the lowest. When RegA and RegB have the fewest state values, the RM achieves the lowest Bc with the maximum sharing between state values.

With this measure, the method to find out the best DBNS representations of each RM is proposed as follows:

1. List all DBNS representations of each constant and combine them as one solution of the RM.

2. Calculate Bc of each solution and choose the one with the lowest Bc.

3. If more than one solution can achieve the lowest Bc, compare their number of state values, the number of $q^b x$ and bit-width of RegA and RegB, to choose the better solution.

Fig. 4 shows the optimized design of Fig. 3 by using the proposed method. The numbers of state values of RegA and RegB are respectively reduced from 7 and 5 to 5 and 4, while Bc of this RM is also reduced from 141 to 102.

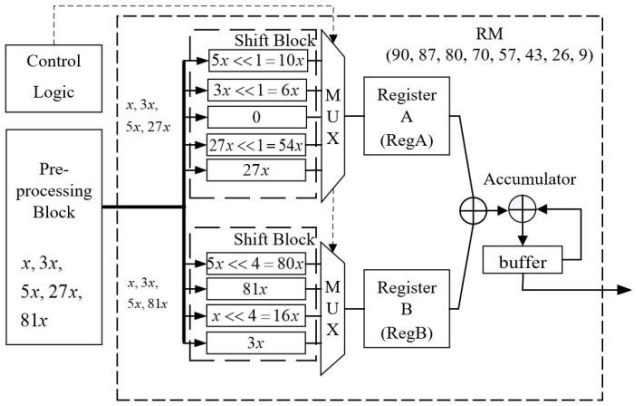

Fig. 4. The optimized hardware architecture of RM

B. Multi-objective optimization using Genetic Algorithm

Both hardware cost and coding performance should be carefully considered in Int-DCT approximation. In this section, we consider the following three measures as the evaluation values of coding performance: (i) the DCT distortion [15] (ii) the coding gain [16] (iii) the transform efficiency [17].

For N-point Int-DCT matrix, the DCT distortion is defined as [15]

$$d_2 = 1 - \frac{1}{N}\left\| \text{diag}\left(T_{DCT} \cdot T_{Nx}^{\mathrm{T}}\right)\right\|_2^2 \tag{8}$$

where diag(**X**) represents the main diagonal of matrix **X**. T_{Nx} denotes the normalized Int-DCT matrix while T_{DCT} represents the infinite-precision DCT matrix. DCT distortion is a measure of the signal energy that is deferred from the individual DCT sub-bands [15]. The lower the DCT Distortion is, the closer the approximated Int-DCT matrix is to the original infinite-precision DCT matrix.

When a transform matrix is orthogonal, its energy compression property can be evaluated by the coding gain which is defined as [18]

$$\mathrm{G}_{TC} = \frac{\frac{1}{N}\sum_{k=0}^{N-1}\sigma_k^2}{\left(\prod_{k=0}^{N-1}\sigma_k^2\right)^{\frac{1}{N}}} \tag{9}$$

where σ_k^2 represent the variance of the k-th transformed coefficients. The infinite-precision DCT matrix is an orthogonal matrix that can be evaluated by (9). However, sometimes the Int-DCT matrix is not orthogonal. In such circumstance, we adopted the unified coding gain [16] which can be computed by

$$C_g = 10 \cdot \log_{10}\left[\prod_{k=1}^{N}\frac{1}{\left(A_k \cdot B_k\right)^{\frac{1}{N}}}\right] \tag{10}$$

where $A_k = \mathrm{sum}\left[\left(\mathbf{h}_k^{\bullet} \cdot \mathbf{h}_k\right) \circ \mathbf{R}_{\mathrm{X}}\right]$, sum(·) is the sum of the elements of its matrix argument, operator ∘ denotes the elementwise matrix product, $B_k = \|\mathbf{g}_k\|_2^2$ and $\|\cdot\|_2^2$ return the Euclidean norm. $\mathbf{h}_k$ and $\mathbf{g}_k$ are the k-th row of $\mathbf{C}_N$ and $\mathbf{C}_N^{\bullet}$, respectively. $\mathbf{R}_{\mathrm{X}}$ is the autocorrelation matrix of input data.

The transform efficiency is a measure of the decorrelation ability of the transform, which is defined as [17]

$$\eta = \frac{\sum_{m=1}^{N} \left| r_{m,m}^{(\mathbf{X})} \right|}{\sum_{m=1}^{N} \sum_{n=1}^{N} \left| r_{m,n}^{(\mathbf{X})} \right|} \cdot 100 \qquad (11)$$

where $r_{m,n}^{(\mathbf{X})}$ denote the (m,n)-th elements of the covariance matrix $\mathbf{X}$, which can be obtained by $\mathbf{X} = \mathbf{C}_N \cdot \mathbf{R}_{\mathbf{x}} \cdot \mathbf{C}_N^{\bullet}$.

The DCT Distortion measures the closeness between Int-DCT matrix and infinite-precision DCT matrix while the coding gain and transform efficiency measure the coding performance. Therefore, together with *Bc*, the Int-DCT design is a multi-objective optimization problem. It is impossible to find an Int-DCT matrix with the best for all measures. Therefore, we adopted Genetic Algorithm in this work to search for the quasi-optimal solution. It firstly takes some randomly generated individuals as the initial population. The next generation is produced by reproduction and selection process. The fitness function measures the adaptability of each individual to the living environment, which represents the quality of each solution. The *Bc*, DCT distortion, coding gain and transform efficiency are the fitness functions in our proposed method. Compared with other multi-objective optimization methods which have been used in Int-DCT design, the optimization method we proposed can quickly find out the Pareto solution even in a complicated search space with a lot of variables.

IV. Experimental Results and Discussion

A. Results of the Proposed Hardware Optimized Matrix

In Section III, an optimization method of Int-DCT implementation based on reconfigurable multipliers is proposed. In this example, we started the initial Int-DCT matrix solution by the 16-point core transform matrix $\mathbf{c}_i$ which defined in HEVC standard [4]. Due to the unique number property [5] that the number of unique elements in a DCT matrix of size $2^M \times 2^M$ equals to $2^M - 1$, the unique number of 16-point Int-DCT matrix is 15, which leads to a search space of size 3^{15}. By using the method that we proposed in Section III, the Int-DCT matrix with the lowest *Bc* can be obtained.

This derived matrix is denoted as Proposed1, and we have implemented this matrix and original matrix in HEVC using the architecture presented in Section III. The hardware is described by Verilog and synthesized on Xilinx xc7a35tftg256-1 by Xilinx Vivado Design Suite v19.1 with the supply power at 1.2V. The clock frequency was set to 80MHz. The proposed cost measure *Bc*, circuit areas in the number of LUT slices, and power in mW are presented in Table I.

TABLE I. *BC* and Hardware Cost on FPGA

Method	*Bc*	#LUT	Power/mW
Proposed1	856	1174	114
The matrix in HEVC	1198	1384	120

The results show that the proposed cost measure is closely correlated with the actual hardware cost. Proposed1 can achieve a lower area by 15.17% over the matrix in HEVC standard. For total power consumption, the proposed design reduces the power cost by 5%. The results verify that the proposed hardware optimization method can search out a lower cost solution.

B. Results of the Proposed Multi-objective Optimized Matrix

In Section III, a multi-objective optimization method by using GA with *Bc*, DCT distortion, coding gain, transform efficiency as objectives is proposed. The search space is set the same as Section IV.A and the solution is denoted as Proposed2. These four measures are compared and the results are shown in Table II.

TABLE II. The Proposed Matrices Performance in Serachspace

Method	*Bc*	DCT Distortion	Coding Gain	Transform Efficiency
Proposed1	100.00%	82.16%	91.05%	88.98%
Proposed2	99.96%	98.74%	99.96%	100.00%

From Table II, we can see that although the *Bc* of Proposed2 is larger than Proposed1, it still lower than 99.96% of the solutions in search space, which leads to an efficient hardware implementation. More importantly, Proposed2 gets a better performance in DCT distortion, coding gain and transform efficiency than Proposed1. In general, the multi-objective optimization method we proposed can clearly search out qualified solution that can achieve a good trade-off between hardware cost and coding performance.

C. Comparisons with Other Algorithms

The above proposed Int-DCT implementations, Proposed1 and Proposed2, are compared with three relevant works proposed in [5], [8] and [11]. CT is the core transform that is applied in HEVC standard [5]. RICT is an Int-DCT algorithm that was proposed by Fong *et.al* in [8], which used the recursive property to simplify hardware architecture. HEICT is a hardware efficient DCT by using weighted sum approach to solve the multi-objective problem [11]. All 16-point designs are described in Verilog and simulated using the same condition as Section IV.A. The number of LUTs (#LUT), the number of Flip Flops (#FF) and the power dissipations are presented in Table III.

TABLE III. Hardware Performance Comparisons on FPGA for 16-point Int-DCT

Method	#LUT	#FF	Power/mW
CT [5]	3862	587	174
RICT [8]	2528	2268	166
HEICT [11]	1479	1126	122
Proposed1	1174	1076	114
Proposed2	1252	1110	116

From Table III, we can clearly see that Proposed1, the solution with lowest *Bc*, achieves the lowest hardware area and power consumption of these competitive algorithms by reduced at least 20.62% of #LUT and 6.55% of power consecution respectively. Furthermore, the solution of our multi-objective algorithm, Proposed2, can still save 15.35% of area and 4.91% of power over the other designs.

V. CONCLUSION

A new hardware optimization and multi-objective optimization methods of Int-DCT in HEVC is presented in this paper. The proposed measure to evaluate the hardware cost contributes to searching out the best DBNS representation, which leads to a reduction of hardware cost. The proposed multi-objective optimization method based on Genetic Algorithm generates a solution with good trade-off between coding performance and hardware cost. The experimental results show that the proposed design can save at least 15.35% of hardware area and 4.91% of power consumption, compared with recently published algorithms.

ACKNOWLEDGEMENT

This work was supported by Jiangsu Natural Science Foundation under grant 1004-PAC19009 and Grant 56YAH18043 at Nanjing University of Aeronautics and Astronautics, Nanjing, China.

REFERENCES

[1] Rao, K. Ramamohan, and Ping Yip. *Discrete cosine transform: algorithms, advantages, applications*. Academic press, 2014.

[2] C. Fong, Q. Han and W. Cham, "Recursive Integer Cosine Transform for HEVC and Future Video Coding Standards," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 2, pp. 326-336, Feb. 2017.

[3] Advanced Video Coding for Generic Audiovisual Services, document ITU-T Rec. H.264, International Telecommunication Union, 2009.

[4] High efficiency video coding, document ITU-T Rec. H.265, International Telecommunication Union, 2013.

[5] M. Budagavi, A. Fuldseth, G. Bjøntegaard, V. Sze and M. Sadafale, "Core Transform Design in the High Efficiency Video Coding (HEVC) Standard," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 1029-1041, Dec. 2013.

[6] R. J. Cintra and F. M. Bayer, "A DCT Approximation for Image Compression," *IEEE Signal Processing Letters*, vol. 18, no. 10, pp. 579-582, Oct. 2011.

[7] R. J. Cintra. "An Integer Approximation method for Discrete Sinusoidal Transforms," *Circuits Systems Signal Processing*, vol.20, no.6, pp. 1481-1501, Dec. 2011.

[8] C. Fong, Q. Han and W. Cham, "Recursive Integer Cosine Transform for HEVC and Future Video Coding Standards," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 2, pp. 326-336, Feb. 2017.

[9] H. Sun, Z. Cheng, A. M. Gharehbaghi, S. Kimura and M. Fujita, "Approximate DCT Design for Video Encoding Based on Novel Truncation Scheme," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 4, pp. 1517-1530, Apr. 2019.

[10] J. Chen, Y. Wang, J. Zhao and S. Rahardja, "A new area and power efficient DCT circuits using sporadic logarithmic shifters," *IEICE Electronics Express*, vol. 16, no. 14, pp. 20190317-20190317, Jun. 2019.

[11] J. Chen, S. Liu, G. Deng and S. Rahardja, "Hardware Efficient Integer Discrete Cosine Transform for Efficient Image/Video Compression," *IEEE Access*, vol. 7, pp. 152635-152645, Oct. 2019.

[12] G. J. Sullivan, J. Ohm, W. Han and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[13] J. Chen and C. Chang, "High-Level Synthesis Algorithm for the Design of Reconfigurable Constant Multiplier," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 12, pp. 1844-1856, Dec. 2009.

[14] V. S. Dimitrov, G. A. Jullien and W. C. Miller, "Theory and applications of the double-base number system," *IEEE Transactions on Computers*, vol. 48, no. 10, pp. 1098-1106, Oct. 1999.

[15] Wien M, Sun S, "ICT comparison for adaptive block transforms," ITU VCEG-L12, Eibsee, Germany. Jan. 2001.

[16] J. Katto, and Y. Yasuda, "Performance evaluation of subband coding and optimization of its filter coefficients," *Journal of Visual Communication and Image Representation*, vol. 2, no. 4, pp. 303–313, Dec. 1991.

[17] Cham, and W-K, "Development of integer cosine transforms by the principle of dyadic symmetry," *IEE Proceedings I (Communications, Speech and Vision)*, vol. 136, no. 4 pp. 276-282, Aug. 1989.

[18] N. S. Jayant, and P. Noll. *Digital coding of waveforms*. Englewood Cliffs NJ: Prentice-Hall, 1984.