Transactions on Machine Intelligence 2(2) (2019) 99-107

ISSN Online: 2821-1693

Transactions on Machine Intelligence

Journal Homepage: https://tmachineintelligence.ir/index.php/TMI

A Low-Cost 16×16 and 32×32 DCT Architectures for HEVC Application Using Configurable Constant Multipliers

R. Younesi[1], S. M. J. Rastegar Fatemi[1], M. Rastegarpour[2,*]

[1] Department of Electrical Engineering, Saveh Branch, Islamic Azad University, Saveh, Iran
[2] Department of Computer Engineering, Saveh Branch, Islamic Azad University, Saveh, Iran

ARTICLE INFO

Article History:
Received 2 February 2019
Received in revised form 6 May 2019
Accepted 19 June 2019
Available online 19 June 2019

Keywords:
DCT; HEVC; Multiple Constant Multiplication; VLSI; Low-Cost

ABSTRACT

This paper introduces a novel area-efficient Discrete Cosine Transform (DCT) architecture designed for the High Efficiency Video Coding (HEVC) standard, a recently introduced international video compression standard. The DCT architecture is capable of performing 16 and 32-point DCT computations, which are essential for HEVC applications. Given the high complexity associated with larger transform sizes in the HEVC standard, the focus of this paper is primarily on developing efficient solutions for these larger point DCT transform sizes. The key innovation lies in leveraging commonality in constant multiplications through the use of configurable constant multipliers, aiming to reduce the area cost and hardware utilization. Consequently, the proposed architecture exhibits a significant reduction in the number of adder and shift blocks compared to existing architectures. Experimental results, conducted for the 90-nm technology node, indicate a remarkable 42% reduction in area consumption compared to other architectures. Furthermore, the proposed DCT architecture is demonstrated to support real-time 4K video resolution.

1. INTRODUCTION

High Efficiency Video Coding (HEVC) is an emerging video coding standard that is widely used for transmitting or storing high and ultra-high resolution video contents. This standard outperforms it predecessor AVC/ H.264 in terms of coding efficiency about two times with almost the same video quality. This emerging video coding was jointly developed by the ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG) and currently has a widespread application in multimedia streaming, broadcast television, multimedia content storage, and mobile communication. The higher coding efficiency in HEVC standard results in higher design complexity in comparison to the previous standards, adverse affecting on devices which have the resource and power limitations. Moreover, since the HEVC standard uses different higher point transform sizes, the hardware implementation faces several challenges that need to be carefully addressed.

Discrete Cosine Transform (DCT) is one of the most complicated transforms which is widely used in transform stage of HEVC encoder and the other applications of digital image and video processing [1,2]. In nature, the DCT

* Corresponding Author: m.rastgarpour@iau-saveh.ac.ir
Department of Computer Engineering, Saveh Branch, Islamic Azad University, Saveh, Iran

http://dx.doi.org/10.47176/TMI.2019.99

transform requires a large number of floating point multipliers resulting in a high area occupation and power consumption on final hardware, which cannot be tolerated, especially in resource constraint internet of thing (IoT) devices. As the transform size of DCT increases, the number of computation and hardware cost also increase quadratically. It should be noted that Unlike the AVC/ H.265, the HEVC uses 4×4 to 32×32 block sizes. Higher block sizes require more hardware resources and their implementation incur higher complexity and consume more power compared to the lower point DCTs.

High Efficiency Video Coding (HEVC) has emerged as a prominent video coding standard, widely utilized for the transmission and storage of high and ultra-high-resolution video content. It surpasses its predecessor, AVC/H.264, by approximately doubling the coding efficiency while maintaining almost the same video quality. Developed jointly by the ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG), HEVC finds extensive applications in multimedia streaming, broadcast television, multimedia content storage, and mobile communication.

Despite its coding advantages, the HEVC standard introduces higher design complexity, impacting devices with resource and power limitations. The utilization of various higher-point transform sizes in HEVC poses challenges for hardware implementation. The Discrete Cosine Transform (DCT), a complex transform widely used in the HEVC encoder's transform stage and other digital image and video processing applications, presents a particular challenge. The inherently high hardware requirements and power consumption of DCT, especially with larger transform sizes, become a critical concern, particularly for resource-constrained Internet of Things (IoT) devices.

Unlike its predecessor AVC/H.265, HEVC adopts block sizes ranging from 4×4 to 32×32. The implementation of larger block sizes requires more hardware resources, leading to increased complexity and higher power consumption compared to lower-point DCTs. As the transform size of DCT increases, both the computation workload and hardware costs escalate quadratically, demanding careful consideration in resource-constrained scenarios.

To address the hardware cost challenges associated with the Discrete Cosine Transform (DCT) architecture in the HEVC standard, many researchers have focused on developing area-efficient solutions, particularly for 16 and 32-point transform sizes. Park et al. [3] proposed 16×16 and 32×32 DCT architectures utilizing shift and adder blocks instead of extensive multipliers to reduce hardware costs. Through hardware sharing techniques, they created an architecture capable of processing 30 frames per second of a 4K video format. Budagavi et al. [4] introduced a unified forward and inverse transform unit by leveraging the symmetry within the transform defined in the HEVC standard.

Jridi et al. [5] presented scalable approximate DCT architectures for HEVC, introducing a reconfigurable Multiple Constant Multiplication (MCM) algorithm [6] supporting both forward and inverse transforms. Their approach achieved enhanced compressed image quality, especially for larger sizes, compared to other approximate DCTs, while maintaining arithmetic complexity comparable to predecessors due to its configurable structure. Chang et al. developed a 4×4 and 8×8 inverse DCT architecture for multiple coding standards, employing hardware sharing to support more transform modes and yield fewer normalized gate counts than existing architectures.

In this manuscript, we present a novel, space-efficient Discrete Cosine Transform (DCT) architecture designed for application in High Efficiency Video Coding (HEVC), accommodating transform sizes of 16 and 32 points. Our approach leverages an approximate integer DCT, as opposed to real-valued DCT coefficients, effectively reducing hardware costs and mitigating encoder-decoder inconsistencies. To implement the integer constant multiplication essential to the HEVC standard's DCT, we capitalize on hardware usability and enhance hardware sharing through the utilization of configurable multiple constant multipliers (MCM).

Experimental findings on the proposed DCT architecture, conducted on TSMC 90-nm technology, reveal a reduced number of arithmetic circuits, including adder and shift blocks, compared to both the direct implementation of the DCT algorithm and alternative architectures. Notably, the proposed architecture demonstrates diminished area and gate count requirements, along with lower computation delay, outperforming other designs. Consequently, this architecture proves suitable for computing DCT of larger sizes in devices with resource and power constraints.

The subsequent sections of this paper are structured as follows: Section 2 provides an overview of the fundamental DCT algorithm and architecture. Following that, Section 3 introduces our DCT architecture, incorporating configurable constant multipliers. Section 4 delves into the experimental results, and the concluding remarks are presented in the final section.

2. OVERVIEW OF INTEGER DCT

An N-point 1-D DCT coefficient Y_i over the input samples x_i can originally be expressed as (1), where $i = 0, 1, .., N-1$ and c_{ij} is elements of the DCT transform matrix C which is defined as (2), where d is equal to 1 and $\sqrt{2}$ for $i = 0$ and $i > 0$ respectively.

$$Y_i = \sum_{j=0}^{N-1} c_{ij} x_j \tag{1}$$

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

According to (2), c_{ij} represents an $N \times N$ transform matrix with real-valued elements. Realizing infinite precision real-valued transform matrix elements will result in a significant area overhead and power consumption of final implementation. Furthermore, to avoid encoder-decoder mismatch and drift caused by manufacturers realizing DCTs with different floating point representations, the core transform matrices of HEVC decoder are defined in [7] as the approximation to the real-valued DCT matrix. Using a matrix representation, one can alternatively define DCT computation of (1) as $[Y_N]^T = [T_N] * [X_N]^T$, where T_N is an $N \times N$ transform matrix, X_N and Y_N are N-point input samples and output coefficients, respectively. The core transform matrix T_N for different N-point DCTs on the basis of integer approximation is defined by HEVC core transform [7]. According to the core transform specified in [7], the 4-point and 8-point transform kernels can be defined as (3) and (4).

$$T_4 = \begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix} \tag{3}$$

$$T_8 = \begin{bmatrix} 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\ 89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 \\ 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\ 75 & -18 & -89 & -50 & 50 & 89 & 18 & -75 \\ 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\ 50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 \\ 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\ 18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 \end{bmatrix} \tag{4}$$

Using even-odd decomposition strategy, an N-point DCT kernel matrix can be decomposed into an $(N/2)$-point even and $(N/2)$-point odd part, where the even part can be further decomposed into the lower point kernels. In this strategy, the $(N/2)$-point even parts represent the DCT computation of the $(N/2)$-point, while the odd part can be realized by multiplication of an $(N/2)\times(N/2)$ matrix. According to the even-odd decomposition, the 8-point 1D-DCT computation which is defined as $[Y_8]^T = [T_8] * [X_8]^T$, can be decomposed as expressed in (5), where T_4 is a transform matrix of 4-point DCT as defined by (3), O_4 represents the matrix multiplication used in 4-point odd block as (7), and a_i and b_i are the even and odd outputs of butterfly block, which can be calculated according to (6). It is worth pointing out that the $T_{N/2}$ and $O_{N/2}$ matrices can be specified by using the even and odd rows of the first half of the basis vectors in matrix T_N, respectively.

$$[y_0, y_2, y_4, y_6] = T_4 \times [a_0, a_1, a_2, a_3] \quad (5)$$

$$[y_1, y_3, y_5, y_7] = O_4 \times [b_0, b_1, b_2, b_3]$$

$$a(i) = x(i) + x(N-i+1) \quad (6)$$

$$b(i) = x(i) - x(N-i+1)$$

$$O_4 = \begin{bmatrix} 89 & 75 & 50 & 18 \\ 75 & -18 & -89 & -50 \\ 50 & -89 & 18 & 75 \\ 18 & -50 & 75 & -89 \end{bmatrix} \quad (7)$$

Similarly, the larger N-point DCTs can be computed by using a lower $(N/2)$-point DCT, an $(N/2)$-point odd block, and N-point butterfly structure. The general architecture of N-point DCT using even-odd decomposition is shown in Fig .1. Direct realization of algorithm presented in (5)-(7) is referred to a reference algorithm in the rest of the paper.

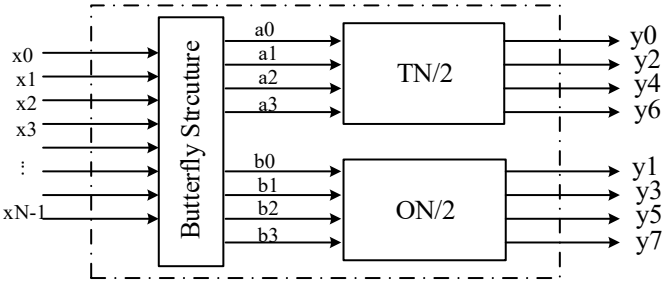

Fig.1. General architecture of N-point DCT using even-odd decomposition strategy [7]

3. PROPOSED AREA-EFFICIENT DCT ARCHITECTURE

The reference algorithm described in Section 2 requires $N^2/4 + M(N/2)$ multiplications, $N^2/4 + N/2 + A(N/2)$ additions, and two shifts, where $M(N/2)$ and $A(N/2)$ represents the number of multiplications and adders for $(N/2)$- point DCT, respectively. To lower the number of arithmetic circuits required to implement 16 and 32-point DCTs, we exploit the redundancy in the adder and shifter blocks required for constant multiplications by using the reconfigurable constant multipliers. As a matter of fact, the additions required for different constants are time-multiplexed to reuse the same adders in the final circuit. This distinguishing feature increases the reusability and hardware sharing in the topology of constant multipliers as well as reducing the hardware cost of final implementation, albeit at the expense of minor increase in latency. The configurable multipliers use the multiplexers (Mux) and a control signal to share the same adder and shift blocks. Fig. 2 shows a configurable multiplier, where an input signal x is multiplied by a set of constants $\{C_1, C_1, \dots, C_N\}$ according to the value of control signal sel. By selecting an appropriate value for "sel," the multipliers can be configured based on a specific constant. Fig. 2(b) provides an illustration of a configurable multiplier, exemplified with the multiplication of the input signal by constants 11 and 21. In Fig. 2(a), the configurable constant multiplier is shown to consist of two adders, three shift blocks, and one multiplexer. It's noteworthy that configurable constant multiplication is achieved by integrating various constant multiplications with the goal of minimizing area consumption.

An alternative to configurable constant multiplication is parallel constant multiplication, depicted in Fig. 2(b). Unlike configurable constant multiplication, the parallel multiplication architecture reduces hardware latency by increasing the number of arithmetic units. However, it does so at the expense of a higher count of adder and shift blocks, eliminating the need for multiplexers. It is important to mention that, in 90-nm technology, the area consumed by a multiplexer is approximately three times less than that of an adder block with the same bit width. Consequently, the configurable constant multiplication is anticipated to have lower area consumption and hardware cost than the parallel approach, owing to the hardware reusability when realizing DCT architectures.

The 2D-DCT architecture can be implemented through a row-column approach by employing two 1D-DCT blocks and a transposition butterfly. The unfolded 2D-DCT architecture is illustrated in Fig. 3. Initially, the first 1D-DCT processes the rows of the 32×32 input block, and the intermediate output generated is stored in a transposition memory. Once all rows of the input block are processed, the column of the transposition memory is then fed into the second 1D-DCT, producing 32 output coefficients.

It's important to highlight that during the processing of the second 1D-DCT to generate 2D-DCT outputs, the first 1D-DCT computation for another input block can be concurrently performed, and its intermediate results can be stored column-wise in the transposition memory. This mechanism enhances the throughput of the final implementation, albeit at the expense of introducing an additional 1D-DCT architecture.

Our primary focus lies in the development of the odd component for 16-point and 32-point 1D-DCT architectures. The even components can be derived by substituting lower-point DCTs. As depicted in Fig. 4, the proposed architecture for the 16-point DCT comprises an 8-point 1D-DCT, a 16-point butterfly block, and an 8-point odd block. This novel 8-point odd block incorporates 8 multiplexers, 8 configurable constant multipliers, and 7 adders. The adder tree in Fig. 4 can be implemented using conventional adders or other high-performance, low-power adders [8]. The throughput of the architecture is two coefficients per cycle. The 8-point DCT unit generates eight coefficient outputs, and the remaining eight outputs are produced by the proposed odd block using configurable constant multiplication. Consequently, the computation of a 16-point DCT requires eight clock cycles. The multiplexers within the odd block are controlled by the C1 signal, which selects one of the odd outputs of the butterfly block. The upper half of the multiplexers is fed with b1, b3, b4, b7, while the lower half is fed with b1, b2, b5, b6. The constants of the configurable multipliers are determined based on the odd matrix block of O8, defined by the odd rows of the first half of the transform matrix T16. Fig. 5 illustrates the eight configurable multipliers necessary for implementing the proposed 16-point DCT. Signal C2, responsible for selecting the appropriate constants in runtime, controls all constant multipliers.

By configuring C1 and C2 through control units, sixteen outputs of the proposed 16-point DCT can be generated accordingly. The same approach is applicable to the proposed 32-point DCT using a 16-point DCT and a 16-point odd block. The embedded 16-point odd block in the 32-point DCT requires 16 multiplexers, 16 constant multipliers, and 15 adders. Furthermore, the throughput of an N-point DCT remains consistent at two coefficients per cycle, irrespective of the transform size.

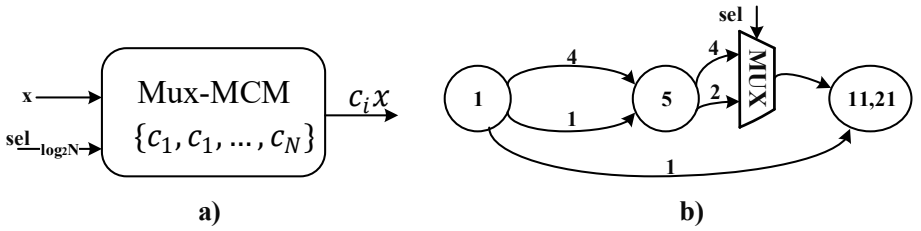

Fig.2. a) Structure of configurable multiplier; b) Example of configurable multiplication with 11 and 21 constants

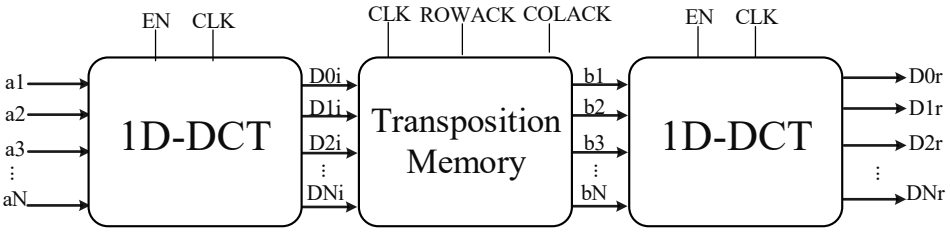

Fig.3. Architecture of unfolded 2D-DCT using two 1D-DCTs based on row-column approach

4. EXPERIMENTAL RESULTS

We implemented the proposed 16-point and 32-point DCT architectures using VHDL language and synthesized them with Synopsys Design Compiler® in the 90-nm technology node. ModelSim software facilitated the verification process, and experimental results were obtained using a personal computer equipped with an Intel Core-i7 processor and 8 gigabytes of DRAM. The algorithm presented in [9] was employed for the implementation of configurable constant multipliers. Table I presents the results, indicating a noteworthy reduction in the number of arithmetic circuits, including adder and shift blocks, for the proposed 16-point and 32-point DCTs compared to the reference algorithm. Additionally, the proposed architecture demonstrates a decrease in area overhead for both 16 and 32-point DCTs.

Further analysis in Table II highlights that the proposed architecture for the 32-point DCT does not require multiplication. The number of adder and shifter blocks needed for implementing the proposed 32-point DCT is lower than that of the architectures in [10–14]. Table III provides a comprehensive comparison between the proposed 2D-DCT architecture and other architectures presented in [12, 13, 15], focusing on gate count and throughput. The proposed DCT architecture operates at a frequency of 370 MHz with a gate count of 124K. Notably, the proposed architecture exhibits a significant reduction in area consumption, averaging around 42%, compared to the architectures in [12, 13, 15]. Consequently, it can efficiently compute a 4K video format with a resolution of 3840×2160 at 60 frames per second (fps).

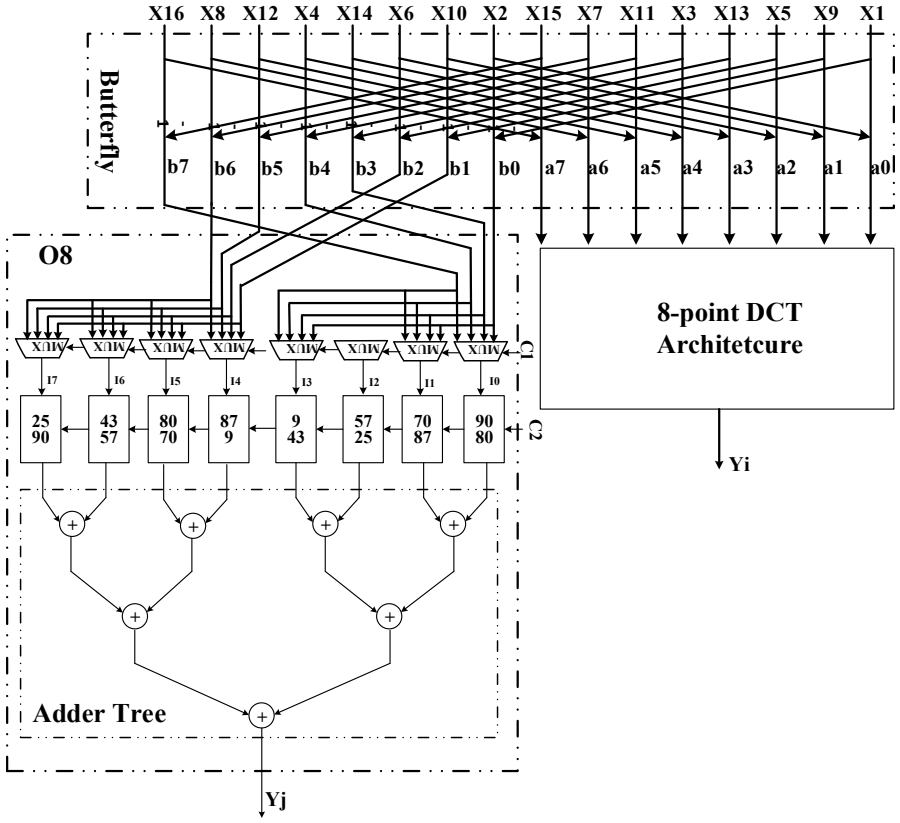

Fig.4. The proposed area-efficient 16-point DCT architecture

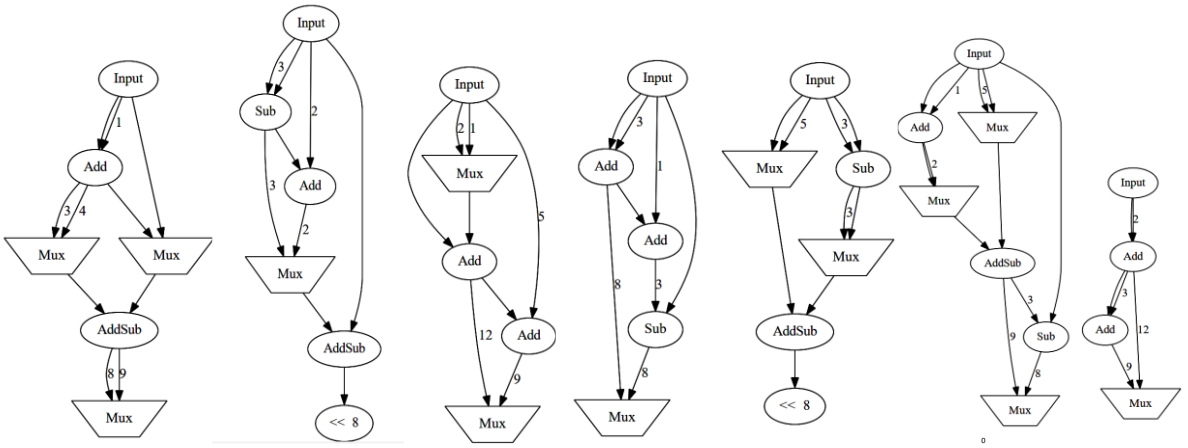

Fig.5. The proposed eight configurable constant multipliers required for realizing the proposed 16-point DCT Architecture

Table 1. Number of arithmetic circuits required for computing 16 and 32-point DCTs

Architecture	Design	Multiplier	Add/Sub	Shift	Multiplexer	Area (µm^2)
Reference Algorithm	DCT-16	86	100	2	0	88451
	DCT-32	342	372	2	0	332889
Proposed Architecture	DCT-16	0	74	66	57	42330
	DCT-32	0	160	139	210	97076

Table 2. Comparison of number of arithmetic circuits required for 32-point DCT

Design	Multiplier	Add/Sub	Shift	Multiplexer
Partial Butterfly [10]	340	372	2	0
Fong [11]	336	400	0	0
CSD-CSE Based [14]	0	764	306	0
Proposed Architecture	0	160	139	210

Table 3. Comparison of 2D-DCT architectures

Design	Tech	Gate Count	Frequency	Throughput	Supporting Video format
Park [15]	90-nm	312 K	168 MHz	16	7680×4320@ 30 fps
Masera [13]	90-nm	243 K	250 MHz	12.84	7680×4320@ 60 fps
Ahmed [12]	90-nm	149 K	150 MHz	1.0/2.6/6.4/7.5≈1.64	2048×1080@ 60 fps
Proposed	90-nm	124 K	370 MHz	2	3840×2160@ 60 fps

5. CONCLUSION

We have devised area-efficient 16 and 32-point DCT architectures using reconfigurable multiplier blocks. The primary innovation in this paper is the adoption of configurable multipliers instead of parallel multipliers. This strategic choice enhances hardware reusability and diminishes hardware costs. Through the integration of multiplexers, we augment hardware reusability by sharing adder and shift blocks, allowing for the configuration of an integrated multiplier to handle multiple constants. This distinctive feature empowers the DCT architecture to reduce both hardware costs and the number of adder and shift blocks necessary for its implementation.

The experimental results underscore the effectiveness of the proposed architecture, demonstrating a 42% reduction in area cost compared to existing architectures. This positions the proposed architecture as highly suitable

for resource-limited High-Efficiency Video Coding (HEVC) applications, such as smart surveillance systems. Furthermore, the flexibility of this architecture extends to supporting the inverse DCT (IDCT) and processing various combinations of transform sizes a direction that could be explored in future iterations of this work.

Transparency Statement

The data supporting this study are available upon reasonable request to the corresponding author, subject to ethical and confidentiality considerations.

Acknowledgments

We would like to express our gratitude to all individuals who contributed to this project.

Declaration of Interest

The authors declare that they have no competing interests.

Funding

This research received no specific grant from any funding agency, commercial, or not-for-profit sectors.

REFERENCES

[1] Shabani A, Timarchi S, Mahdavi H. Power and area efficient CORDIC-Based DCT using direct realization of decomposed matrix. *Microelectronics Journal* 2019;91:11–21. https://doi.org/10.1016/j.mejo.2019.07.008.

[2] Shabani A, Timarchi S. Low-power DCT-based compressor for wireless capsule endoscopy. *Signal Processing: Image Communication* 2017;59:83–95. https://doi.org/10.1016/j.image.2017.03.003.

[3] Park JS, Nam WJ, Han SM, Lee S. 2-D large inverse transform (16x16, 32×32) for HEVC (High Efficiency Video Coding). *Journal of Semiconductor Technology and Science* 2012;12:203–11. https://doi.org/10.5573/JSTS.2012.12.2.203.

[4] Budagavi M, Sze V. Unified forward+inverse transform architecture for HEVC. In: *Proceedings - International Conference on Image Processing, ICIP*; 2012. p. 209–12. https://doi.org/10.1109/ICIP.2012.6466832.

[5] Jridi M, Meher PK. Scalable approximate DCT architectures for efficient HEVC-compliant video coding. *IEEE Transactions on Circuits and Systems for Video Technology* 2017;27:1815–25. https://doi.org/10.1109/TCSVT.2016.2556578.

[6] Voronenko Y, Püschel M. Multiplierless multiple constant multiplication. *ACM Transactions on Algorithms* 2007;3:11--es. https://doi.org/10.1145/1240233.1240234.

[7] Budagavi M, Fuldseth A, Bjontegaard G, Sze V, Sadafale M. Core transform design in the high efficiency video coding (HEVC) Standard. *IEEE Journal on Selected Topics in Signal Processing* 2013;7:1029–41. https://doi.org/10.1109/JSTSP.2013.2270429.

[8] Hassanzadeh A, Shabani A. Low Power Parallel Prefix Adder Design Using Two Phase Adiabatic Logic. *Journal of Electrical and Electronic Engineering* 2015;3:181. https://doi.org/10.11648/j.jeee.20150306.11.

[9] Tummeltshammer P, Hoe JC, Puschel M. Time-multiplexed multiple-constant multiplication. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 2007;26:1551–63.

[10] Bossen F, Sühring K. Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11 2015:1–30. Available: https://hevc.hhi.fraunhofer.de/svn/svn HEVCSoftware/tags/HM-8.0/.

[11] Fong CK, Han Q, Cham WK. Recursive Integer Cosine Transform for HEVC and Future Video Coding Standards. *IEEE Transactions on Circuits and Systems for Video Technology* 2017;27:326–36. https://doi.org/10.1109/TCSVT.2015.2513664.

[12] Ahmed A, Shahid MU, Rehman AU. N point DCT VLSI architecture for emerging HEVC standard. *VLSI Design* 2012;2012. https://doi.org/10.1155/2012/752024.

[13] Masera M, Martina M, Masera G. Adaptive Approximated DCT Architectures for HEVC. *IEEE Transactions on Circuits and Systems for Video Technology* 2017;27:2714–25. https://doi.org/10.1109/TCSVT.2016.2595320.

[14] Darji AD, Makwana RP. High-performance multiplierless DCT architecture for HEVC. In: *19th International Symposium on VLSI Design and Test, VDAT*; 2015. p. 1–5. https://doi.org/10.1109/ISVDAT.2015.7208051.

[15] Park SY, Meher PK. Flexible integer DCT architectures for HEVC. In: *Proceedings - IEEE International Symposium on Circuits and Systems*; 2013. p. 1376–9. https://doi.org/10.1109/ISCAS.2013.6572111.