SCALABLE VLSI ARCHITECTURE FOR HIGH-PERFORMANCE INTEGER TRANSFORM IN UHD HEVC VIDEO PROCESSING

D.Pravallika[1], D. Sumalatha[1], B. Vara Lakshmi[1]

[1]Department of Electronics and Communication Engineering

[1]Sree Dattha Institute of Engineering and Science, Sheriguda, Hyderabad, Telangana

Abstract

This research presents a versatile VLSI architecture for computing the N-point Discrete Cosine Transform (DCT), a critical component in High-Efficiency Video Coding (HEVC). The HEVC standard supports multiple block sizes for DCT computation, ranging from 4 × 4 to 32 × 32, necessitating a flexible and efficient hardware design. To optimize area utilization and processing speed, particularly for large video sequences, this study proposes a partially folded hardware architecture. The design strategically decomposes DCT matrices into sparse submatrices, significantly reducing the number of required multiplications. Moreover, in a pioneering approach, the architecture eliminates multiplications entirely by employing the lifting scheme, enhancing computational efficiency while minimizing hardware complexity. This innovation not only streamlines hardware implementation but also ensures seamless real-time processing for 1080P HD video codecs, achieving an operational frequency of 150 MHz. The proposed design effectively balances computational throughput and hardware efficiency, making it a practical solution for modern video processing applications.

Keywords: Bit-plane matrix, HEVC, Integer Transform, Ultra High Definition, VLSI Architecture, DCT, Video Coding.

1. Introduction

With rapid technological advancements, hardware size is shrinking while storage capacity continues to expand. This evolution has led to the widespread use of high-end video applications in daily life, including movie streaming, video conferencing, and high-definition video recording. Modern smartphones and other compact devices now support a range of multimedia applications that were once considered impractical. The increasing demand for high-quality video processing has made the development of highly efficient video coders essential. However, achieving high efficiency in video compression comes with the challenge of increased computational complexity. As devices become more powerful and multifunctional, the need for optimized video coding techniques becomes even more critical. The trade-off between efficiency and complexity necessitates innovative solutions that balance performance and processing power. Advanced compression algorithms and machine learning-driven video encoding methods are being explored to meet these demands while ensuring seamless real-time video experiences. The growing reliance on cloud computing and edge computing further enhances video processing capabilities, allowing for real-time applications with minimal latency. High-efficiency video coding (HEVC) and its successors, such as Versatile Video Coding (VVC), have significantly improved compression ratios, reducing bandwidth and storage requirements. These advancements are vital for supporting ultra-high-definition (UHD) and 4K video applications, enabling smoother streaming and higher-quality video content on various devices. As video technology continues to evolve, balancing efficiency with computational feasibility remains a key challenge, driving research into novel encoding strategies and hardware optimizations to support the next generation of multimedia experiences.As pointed out in [1, 2], several blocks of video codecs, including the transform stage [3], motion estimation and entropy coding [4], are responsible for this high complexity. As an example the discrete-cosine-transform (DCT), that is used in several standards for image and video compression, is a computation intensive operation. In particular, it requires a large number of additions and multiplications for direct implementation. HEVC, the brand new and yet-to-release video coding standard, addresses high efficient video coding. One of the tools employed to improve coding efficiency is the DCT with different transform sizes. As an example, the 16-point DCT of HEVC is shown in [5]. In video compression, the DCT is widely used because it compacts the image energy at the low frequencies, making easy to discard the high frequency components. To meet the requirement of real-time processing, hardware implementations of 2-D DCT/inverse DCT (IDCT) are adopted, for example, [6].

2.literature Survey

The 2-D DCT/ IDCT can be implemented with the 1-D DCT/IDCT and a transpose memory in a row-column decomposition manner. In the direct implementation of DCT, float-point multiplications have to be tackled, which cause precision problems in hardware. Hence, we propose a Walsh-Hadamard transformbased DCT implementation [7]. Then, inspired by the DCT factorizations proposed in [8, 9], we factorize the remaining rotations into simpler steps through the lifting scheme [10]. The resulting lifting scheme-based architecture, inspired by [11–13], is simplified, exploiting the techniques proposed in [9, 14] to achieve a multiplierless implementation. Other techniques can be employed to achieve multiplierless solutions, such as the ones proposed in [8, 15–18], but they are not discussed in this work. In this work, the proposed multisize DCT architecture supports all the block sizes of HEVC and is proposed for the real-time processing of 1080P HD video sequences.

III. EXISTING SYSTEM

HEVC was developed with the goal of providing twice the compression efficiency of the previous standard, H.264 / AVC. Although compression efficiency results vary depending on the type of content and the encoder settings, at typical consumer video distribution bit rates HEVC is typically able to compress video twice as efficiently as AVC. Types of compressions There are two types of compressions Digital identity to the original image. Only achieve a modest amount of compression Lossless compression involves with compressing data, when decompressed data will be an exact replica of the original data. This is the case when binary data such as executable are compressed. Discards components of the signal that are known to be redundant. Signal is therefore changed from input.

III. PROPOSED SYSTEM

SIGNED BIT MATRIX-BASED TRANSFORM ALGORITHM:

In order to narrow the bit width of intermediate transformed data, we propose the bit decomposition algorithm which decomposes the integer transform matrix into several SBT matrices. Let di,j be the element in the ith row and jth column in the N × N integer transform matrix DN , i.e., DN = (di,j). If di,j is positive, the binary expression of di,j is (bK−1,i,j ,..., b1,i,j b0,i,j)2, bk,i,j ∈ {0, 1}. Then, there is the relation

$$d_{i,j} = \sum_{k=0}^{K-1} \left(\mathrm{sgn}(d_{i,j}) b_{k,i,j} 2^{k} \right), \; b_{k,i,j} \in \{0, 1\} \qquad (1)$$

where K is the number of binary significant bits of element di,j , bk,i,j denotes the kth bit of di,j, and sgn(∗) is the sign indication function that returns 1 for the positive value and −1 for the negative value, i.e., sgn(x)={ 1, $x > 0$ −1, $x < 0$. Thus (1) can also be written as

$$d_{i,j} = \sum_{k=0}^{K-1} (b_{k,i,j} 2^{k}), \; b_{k,i,j} \in \{0, \mathrm{sgn}(d_{i,j})\} . \qquad (2)$$

Equation (2) is the signed integer binarization. If all elements di,j in integer transform matrix DN are binarized and all the kth bits of all binary di,j , bk,i,j , 0 ≤ i, j < N, construct the kth bit plane, which is expressed in the form of N × N SBT matrix BN,k = (bk,i,j), there is Equation (2) is the signed integer binarization. If all elements di,j in integer transform matrix DN are binarized and all the kth bits of all binary di,j , bk,i,j , 0 ≤ i, j < N, construct the kth bit plane, which is expressed in the form of N × N SBT matrix BN,k = (bk,i,j), there is

$$D_N = \sum_{k=0}^{K-1} (B_{N,k} 2^{k}) \qquad (3)$$

Where K is the binary bit width of the maximum element in matrix DN , i.e., K = log2 max(di,j) . The matrix BN,k containing elements of 0 or ±1 is just the SBT matrix. K SBT matrices are totally generated in the matrix decomposition.

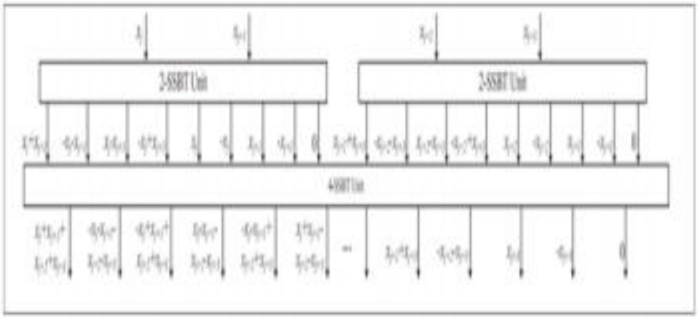


Fig. 1 . Hierarchical structure of SBT.

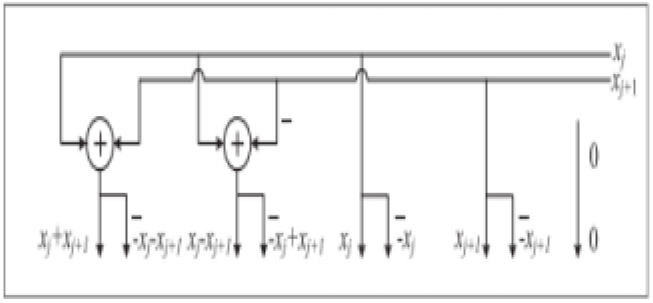


Fig. 2. Adder reuse circuit of 2-SSBT.

The number of all the possible element combination situations of 2-SSBT is 32 = 9. The nine element combinations are (1, 1), (1, −1), (−1, 1), (−1, −1), (1, 0), (−1, 0), (0, 1), (0, −1), and (0, 0). As for the 2-SSBT vector, all possible addition operations for −→b l k,i · −→x l can be expressed as:

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \\ 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_j \\ x_{j+1} \end{pmatrix} = \begin{pmatrix} x_j + x_{j+1} \\ x_j - x_{j+1} \\ -x_j + x_{j+1} \\ -x_j - x_{j+1} \\ x_j \\ -x_j \\ x_{j+1} \\ -x_{j+1} \\ 0 \end{pmatrix}.$$

The inner circuit design of the 2-SSBT unit is described in Fig. 4.1. According to the relationship between M-SSBT and M/2- SSBT, the SBT can be implemented in a hierarchical way. An M-SSBT can be implemented through jointing two M/2- SSBTs. The hierarchical structure of 4- SSBT as an example is illustrated in Fig.2 where the output of two 2-SSBT units is input to a 4- SSBT unit for 4-SSBT computation. The inner circuit design of the 4-SSBT circuit is also shown in Fig. 3. It can be summarized that the number of adders, #ADDER, used in the proposed adder reuse scheme for SBT can be calculated according to the expression

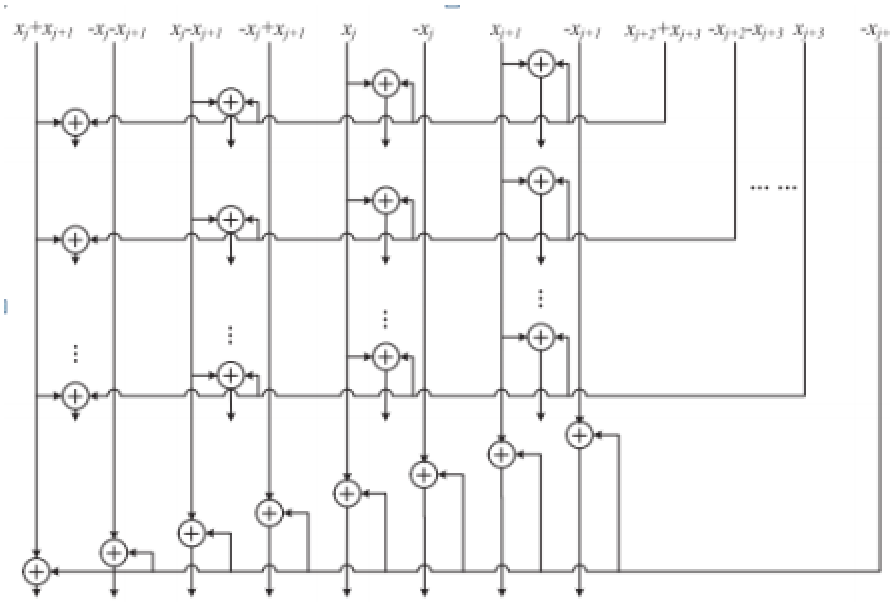


Fig. 3 Part of the adder reuse circuit of 4-SSBT

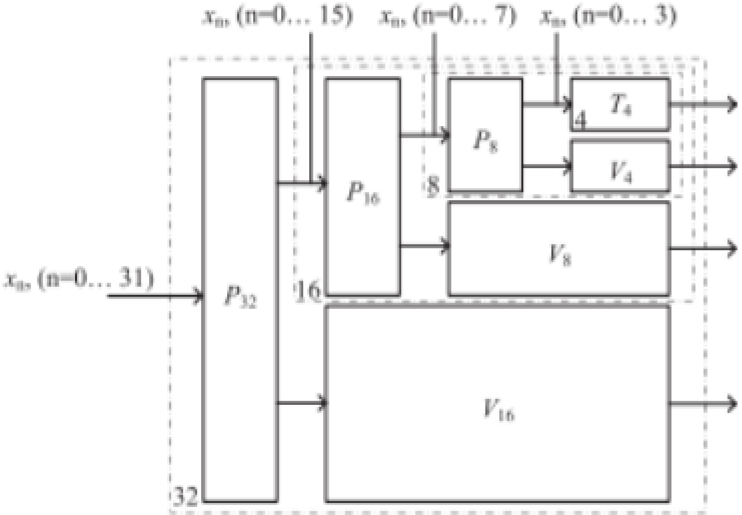


Fig. 4 One-dimensional 32 × 32 transform top-level architecture.

5. SIMULATION RESULTS

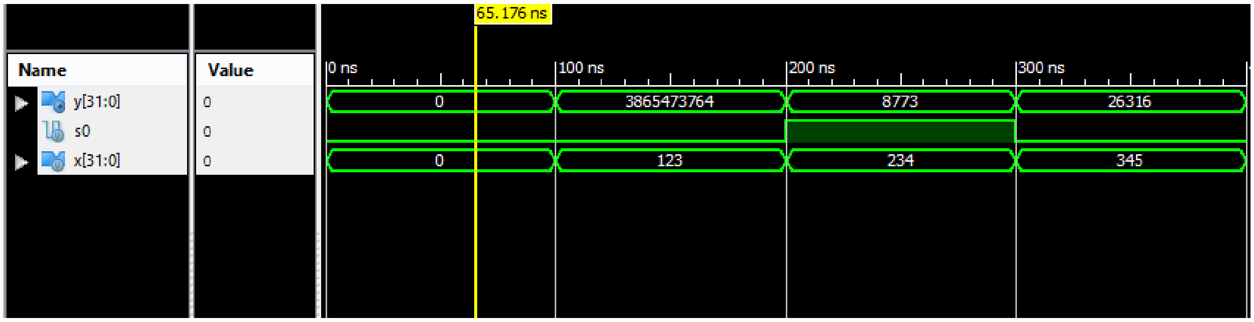


Fig 5.1 waveforms

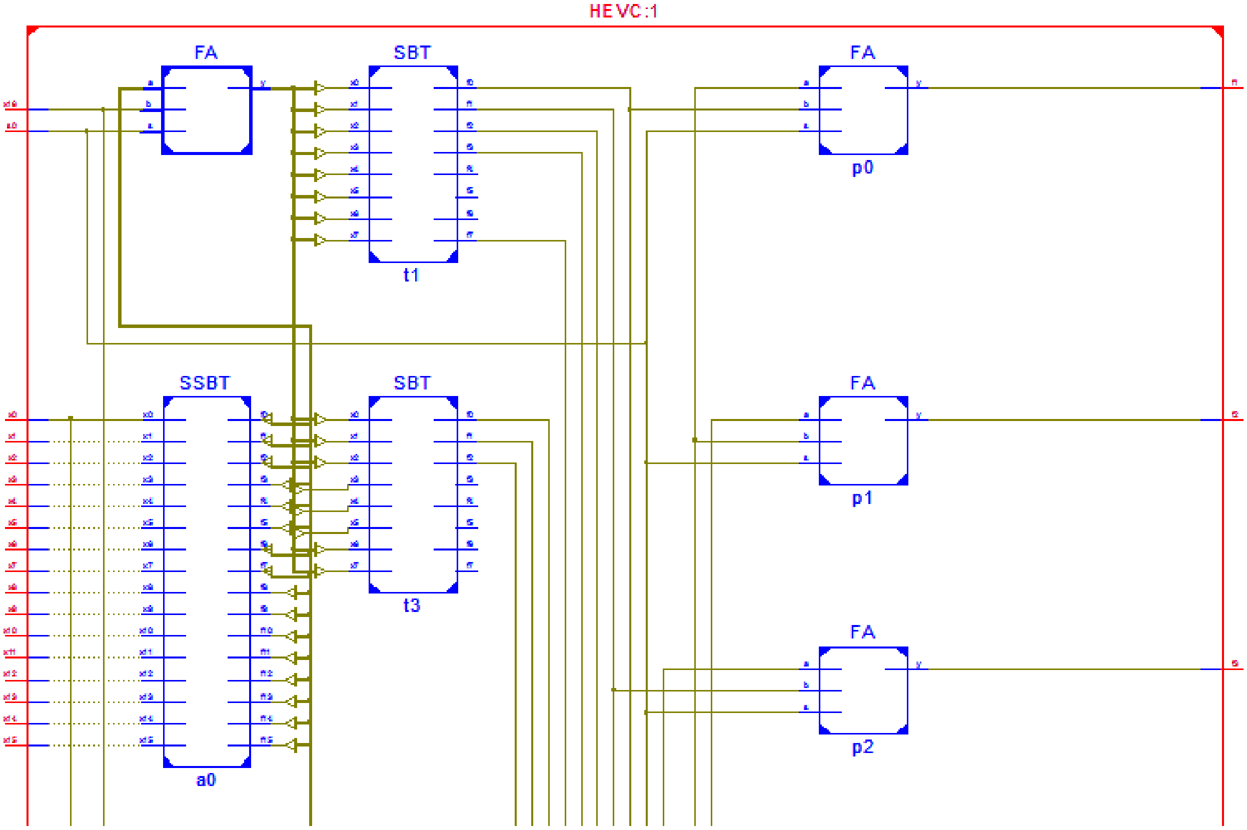
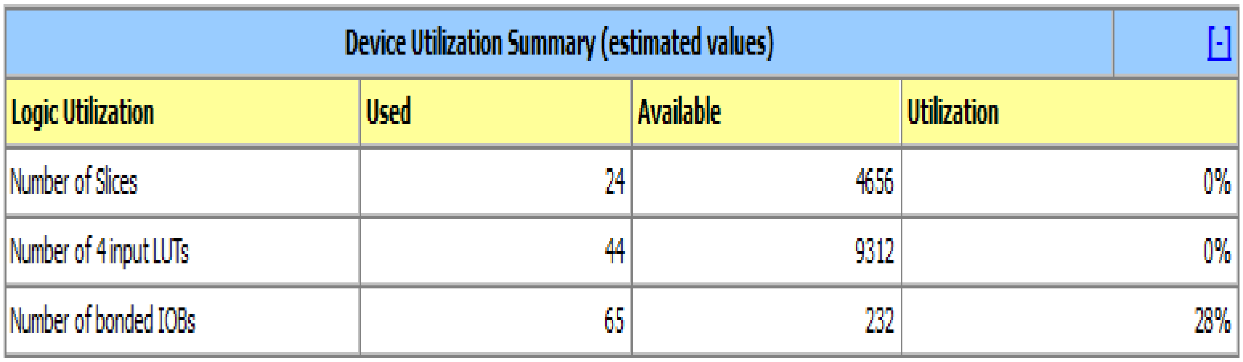
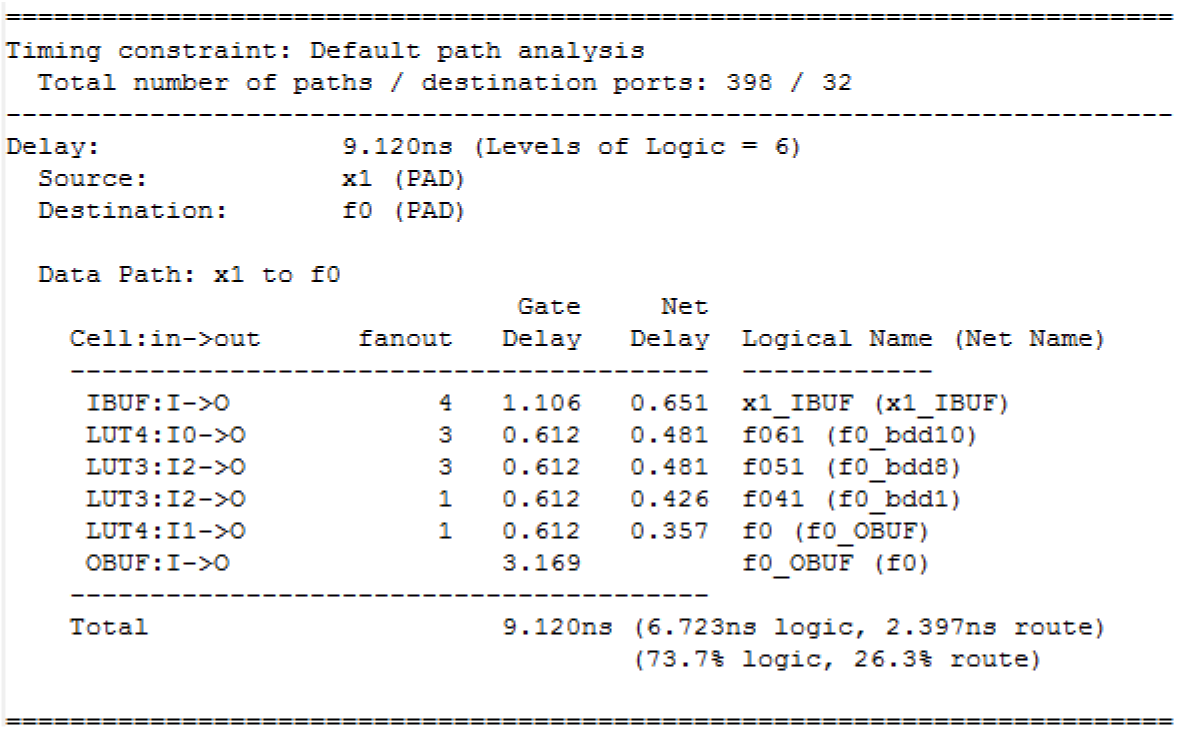


Fig 5.2 RTL schematic



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	24	4656	0%
Number of 4 input LUTs	44	9312	0%
Number of bonded IOBs	65	232	28%

Fig 5.3 Design Summary



```
========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 398 / 32
------------------------------------------------------------------------
Delay:               9.120ns (Levels of Logic = 6)
  Source:            x1 (PAD)
  Destination:       f0 (PAD)

  Data Path: x1 to f0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             4   1.106   0.651  x1_IBUF (x1_IBUF)
     LUT4:I0->O            3   0.612   0.481  f061 (f0_bdd10)
     LUT3:I2->O            3   0.612   0.481  f051 (f0_bdd8)
     LUT3:I2->O            1   0.612   0.426  f041 (f0_bdd1)
     LUT4:I1->O            1   0.612   0.357  f0 (f0_OBUF)
     OBUF:I->O                 3.169          f0_OBUF (f0)
    ----------------------------------------
    Total                      9.120ns (6.723ns logic, 2.397ns route)
                                       (73.7% logic, 26.3% route)

========================================================================
```

Fig 5.4 Time Summary

CONCLUSION

A fast integer transform VLSI engineering based inadequate SBT is proposed for continuous ultra HD video coding acclimating to the HEVC standard. Considering the bit width impact on circuit delay, the bit width of the number change framework is upgraded in the proposed VLSI engineering. The whole number change lattice with high-piece width components is deteriorated into a few SBT lattices with low-piece width components based on the proposed lattice bit-plane deterioration strategy. The change engineering with the SBT calculation can work more productively for the lowpiece width calculations. The circuit reuse system is especially proposed for the SBT to lessen the number of adders of the VLSI architecture. The proposed transform hardware architecture can process video data with higher speed and proper area compared with previous work. A large number of adders for the SBT is saved using the proposed circuit reuse strategy

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao, “On image processing and a discrete cosine transform,” IEEE Trans. Comput., vol. C-23, no. 1, pp. 90–93, Jan. 1974.

[2] G. K. Wallace, “JPEG still image data compression standard,” Commun. ACM, vol. 34, no. 4, pp. 30–44, Apr. 1991.

[3] “Generic coding of moving pictures and associated audio information Part 2: Video ITU-T and ISO/IEC JTC1,” in ITU-T Recommendation H.262, ISO/IEC 13818-2 (MPEG-2), Nov. 1994

[4] “Video coding for low bit-rate communication Version 1,” in ITU-T Recommendation H.263, Nov. 1995.

[5] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003.

[6] L. Yu, S. Chen, and J. Wang, “Overview of AVS-video coding standards,” Signal Process., Image Commun., vol. 24, no. 4, pp. 247–262, Apr. 2009.

[7] G. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, “Overview of the high efficiency video coding (HEVC) standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[8] W.-H. Chen, C. H. Smith, and S. C. Fralick, “A fast computational algorithm for the discrete cosine transform,” IEEE Trans. Commun., vol. COM-25, no. 9, pp. 1004–1009, Sep. 1977.

[9] A. Ahmed, M. U. Shahid, and A. ur Rehman, “N-point DCT VLSI architecture for emerging HEVC standard,” VLSI Design, vol. 2012, 2012, Art. no. 752024.

[10] Y. Arai, T. Agui, and M. Nakajima, “A fast DCT-SQ scheme for images,” Trans. IEICE, vol. E71, no. 11, pp. 1095–1097, Nov. 1988.