An Optimized Hardware Implementation of 4-point Adaptive Multiple Transform design for post-HEVC

A. Kammoun*†, S. Ben Jdidia*, F. Belghith*, W. Hamidouche†, J. F. Nezan†, and N. Masmoudi*
* National School of Engineering Sfax, University of Sfax, Tunisia
e-mail: Ahmed.Kammoun@insa-rennes.fr
† IETR UMR 6164 / INSA Rennes, France
e-mail: Wassim.Hamidouche@insa-rennes.fr

***Abstract*—Under the exploration of the future video coding standard a new transform design called Adaptive Multiple Transform (AMT) has been proposed. This design involves five DCT/DST-based transform types known as: DCT-II, DCT-VIII, DCT-V, DST-I and DST-VII. This work, proposes multiplierless hardware architectures of size 4 for all considered transform types. These architectures are implemented in FPGA benefitting from both correlation and symmetry properties of the matrices coefficients. This paper presents and compares two different architecture aspects without and with involving state-machines to evaluate their effect on the proposed hardware design. The experimental and synthesis results show that the two methods, supporting all five transform types, require less than 3% of the offered FPGA device area and provide respectively 318 MHz and 285 MHz as maximum operation frequency. With adding the pipelining operation, the proposed designs can support real time coding of 4Kp30 and 2Kp60 videos, respectively. Moreover, the implementation based on state-machines offers about 45% operations number and hardware area reduction.**

I. Introduction

The High Efficiency Video Coding (HEVC) [1] is the latest video coding standard defined by the International Telecommunication Union/Telecommunication (ITU-T) and the International Organization for Standardization (ISO) in 2013. HEVC provides a bit-rate reduction up to 50% with respect to Advanced Video Coding (AVC) standard [2] for the same subjective video quality [3], [4].

The exponential increase of the users demand for high video quality applications stresses the need of designing a new video coding standard. Recently, to face these new challenges, the ITU-T and ISO collaboration has created a new team called Joint Video Exploration Team (JVET) charged to investigate a potential future standard with higher coding performance than HEVC. Therefore, through their several meetings [5], a new software called Joint Exploration test Model (JEM), based on HEVC reference HM16.6 [6], has been established to develop and test the proposed technical contributions. The JEM software provided about 25% to 30% coding gain compared to HEVC [7], [8].

This gain comes from a summation and a consequence of several algorithmic improvements such as the new approach of transform called Adaptive Multiple Transform (AMT) which involves new transform types from Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST) family. A detailed description of all tools is provided in document [9]. Actually, this coding gain was enabled at the expense of higher complexity (up to x10) at both encoder and decoder sides compared to HEVC (HM) [7]. Therefore, this high complexity becomes a real issue for the development of a new video coding standard. This study focuses on the transform module as one of the most time consuming operation in the JEM software especially with involving four new transform types of DCT/DST family. This paper proposes a simplified algorithm with optimized hardware architectures of all transform types predefined of size 4.

The rest of the paper is organized as follows. The new transforms used in the JEM software as well as the existing hardware implementations of DCT-based transforms are described in Section II. Section III introduces the proposed hardware architectures of the new transform types of size 4. The experimental and synthesis results are provided and discussed in Section IV. Finally, Section V concludes this paper.

II. Related work

A. Adaptive Multiple Transform Design

The HEVC standard is based on the the well-known DCT type II as the main transform function and the DST type VII for Intra code blocks of size 4×4. In the the JEM exploration software, the use of trigonometric transforms has been extended with the Adaptive Multiple Transform (AMT) that includes DCT-II, DCT-V, DCT-VII, DST-I and DST-VII transforms. The AMT algorithm is applied at the processing unit level (inter or intra prediction residual block) A specific *CU-level* flag is added to signal whether single or multiple transforms is used. If the *CU-level* flag is equal to 0, the classic HEVC transforms (DCT-II and DST-VII) are applied, otherwise two additional flags are added to signal the horizontal and vertical transforms used for the current CU [9].

For both Inter and Intra CU blocks, the JEM encoder encodes the CU with all transforms within the selected set and then chooses the one that minimizes the rate distortion cost. Related to their magnitude characteristics, the combinations of these transform types contribute efficiently and improve the flexibility of the transform design [10]. However, the fact that five transform types will be excessively evaluated for each CU, comes with the cost of higher computation complexity. This can be an issue for real time implementation especially that most of the involved transform types do not have a fast implementation [11].

B. Fast DCT/DST implementations

Several previous video coding standards are based on the DCT-II type transform. Therefore, many works proposed an efficient hardware implementations for DCT-II type transform with simplified decomposition methods. Shen et al [12] presented

a unified Very-Large-Scale Integration (VLSI) architecture for 4, 8, 16, and 32 point Inverse Integer Core Transforms (IICT). A multiplierless technique was applied to the 4- and 8-point IDCTs, and regular multipliers and hardware sharing were applied to the 16- and 32-point IICTs. To reduce the required hardware resources, the memory was transposed using the Static Random Access Memory (SRAM) module.

Paramud et al. [13] presented an efficient and reusable architectures for the implementation of DCT-II for different lengths using constant matrix multiplication. Moreover, the proposed architecture can be pruned to reduce the complexity of implementation substantially with only a marginal effect on the coding performance for both folded and full-parallel 2-D DCT implementations.

Ahmed et al. [14] proposed a dynamic N-point DCT for HEVC designed all inverse transform sizes ($4 \times 4, 8 \times 8, 16 \times 16$ and 32×32). The hardware architecture is partially folded in order to save the area and improve the speed up of the design. The proposed architecture reached as maximum frequency of 150 MHz which enables to support real time of 1080p30 video coding [14] .Ahmed et al [15] proposed a unified FPGA architecture of Inverse DCT-II for all transform unit sizes through sparse matrices decomposition eliminating multiplication operations. To further reduce hardware area, they exploited resources reuse and involved multiplexers to provide multiple coefficients for large sizes. The proposed architecture enables to reach an operational frequency up to 284 MHz.

Moreover, some works in the literature proposed hardware implementations of other DCT/DST transform types. Authors in [16] proposed an optimized hardware cost of DST-VII and DCT-VIII types with 5 multiplications and 11 additions for 4×4 instead of respectively 12 and 16 in the original design. Work in [17] also minimized the number of operations of the DCT and DST transform types compared to the original but with using multiplications which are heavily resources consuming operations.

At the best of our acknowledgement, there is no contributions for a low complexity architecture sufficient enough to hold all these five transform types together. Therefore, we propose in this paper a multiplierless and low cost consumption architecture for 4×4 size of AMT transform types to further preserve the hardware area while maintaining the desired performance.

III. Hardware architectures of the AMT transform types

The JEM codec is based on transform basis functions to calculate multiplication matrices coefficients. Hence as a first step, we extracted these matrices to be used as an input for the hardware implementation. In the following, we will present the proposed decomposition algorithms of each transform type for size 4×4 with their associated architectures benefiting from the matrices correlation and symmetry. The general equation for all transform types is:

$$Dst\_1D = M4 \cdot Src \tag{1}$$

where Src [src0...src3] represents the residual vector which is the input of the 1D-transform unit and Dst_1D [dst0...dst3] is the corresponding output one.

A. DCT-II transform

The only difference, with respect to HEVC DCT-II transform, consists in the coefficients values while the symmetric property of the matrix remains unchanged. Therefore, proceeding with the same decomposition method proposed in [15], the size 4 of DCT-II matrix will be computed as follows: In equation (1), M4 can be described as:

$$M4 = P4r \cdot M4.1 \cdot P4.1 \tag{2}$$

where M4= $\begin{pmatrix} 256 & 256 & 256 & 256 \\ 334 & 139 & -139 & -334 \\ 256 & -256 & -256 & 256 \\ 139 & -334 & 334 & -139 \end{pmatrix}$

P4.1 = $\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$ and P4r = $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

in order to obtain the bloc-diagonal matrix as follows:

M4.1= $\begin{pmatrix} 256 & 256 & 0 & 0 \\ 256 & -256 & 0 & 0 \\ 0 & 0 & -139 & -334 \\ 0 & 0 & 334 & -139 \end{pmatrix}$

$\begin{pmatrix} 256 & 256 \\ 256 & -256 \end{pmatrix} \oplus \begin{pmatrix} -139 & -334 \\ 334 & -139 \end{pmatrix}$

where $\oplus$ is direct sum operator.

M4.11= $\begin{pmatrix} 256 & 256 \\ 256 & -256 \end{pmatrix}$ and M4.22= $\begin{pmatrix} -139 & -334 \\ 334 & -139 \end{pmatrix}$

M4.11 requires 2 additions and 2 shifts operations. However M4.22= 128 x $\begin{pmatrix} -1 & -3 \\ 3 & -1 \end{pmatrix}$+$\begin{pmatrix} -11 & 50 \\ -50 & -11 \end{pmatrix}$and of course with replacing these coefficients by their equivalences it will require then 16 additions and 14 shifts. Fig. 1 illustrates the 4 point DCT-II hardware architecture.

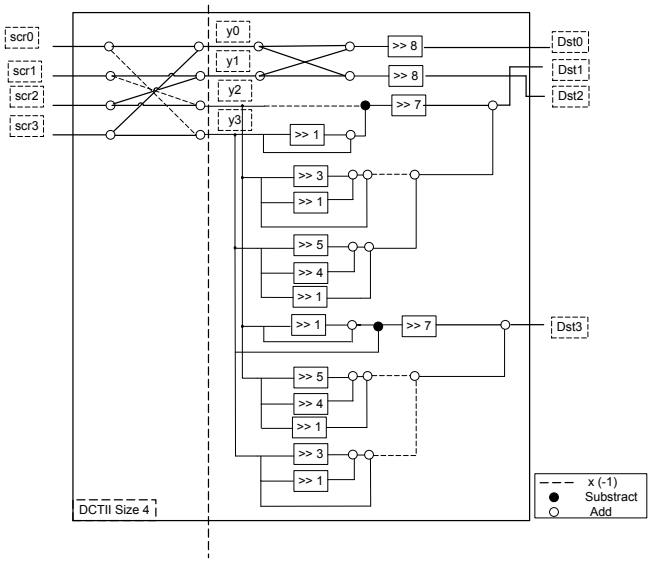

Fig. 1. Proposed hardware architecture for DCT-II size 4

B. DST-I transform

The previous decomposition principle can be applied since the DST-I matrix coefficients present the same symmetric properties than DCT-II.

$$\text{M4}=\begin{pmatrix} 190 & 308 & 308 & 190 \\ 308 & 190 & -190 & -308 \\ 308 & -190 & -190 & 308 \\ 190 & -308 & 308 & -190 \end{pmatrix}$$

Through the same sparse matrices we obtain the following bloc-diagonal matrix:

$$\text{M4.1}=\begin{pmatrix} 190 & 308 & 0 & 0 \\ 308 & -190 & 0 & 0 \\ 0 & 0 & -190 & -308 \\ 0 & 0 & 308 & -190 \end{pmatrix}$$

We can notice that coefficients are similar in both submatrices, that's why it would be the same block architecture with appropriate signs.

$$\begin{pmatrix} 190 & 308 \\ 308 & -190 \end{pmatrix} = 190 \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} + 118 \cdot \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

This equation requires 12 additions and 12 shifts as number of operations . Fig. 2 presents the proposed architecture of DST-I.

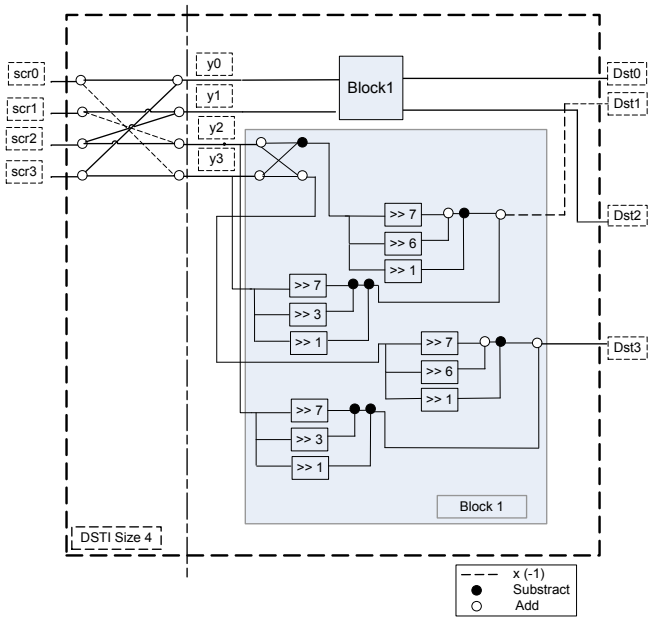

Fig. 2. Proposed hardware architecture for DST-I size 4

C. DST-VII transform

$$\text{M4}=\begin{pmatrix} 117 & 219 & 296 & 336 \\ 296 & 296 & 0 & -296 \\ 336 & -117 & -296 & 219 \\ 219 & -336 & 296 & -117 \end{pmatrix}$$

Noticing that the 1^{st}, 3^{rd} and 4^{th} rows have the same coefficients but in different order and signs, the idea consists in designing one block that can be used three times in parallel with the appropriate coefficients order and signs. This equation is an example of the decomposition adopted for the first output:

$$\begin{aligned} dst\_0 = 128 \cdot [&src\_0 + 2(src\_1 + src\_2 + src\_3) + src\_3] \\ &+ 8 \cdot [-src\_0 - 5(src\_1 - src\_2) - 6src\_3] \\ &- 3 \cdot [src\_0 - src\_1]\,. \end{aligned} \tag{3}$$

The DST-VII transform architecture presented in Fig. 3 details how additions and shift operations are used to compute the DST-VII transform while eliminating all multiplications.

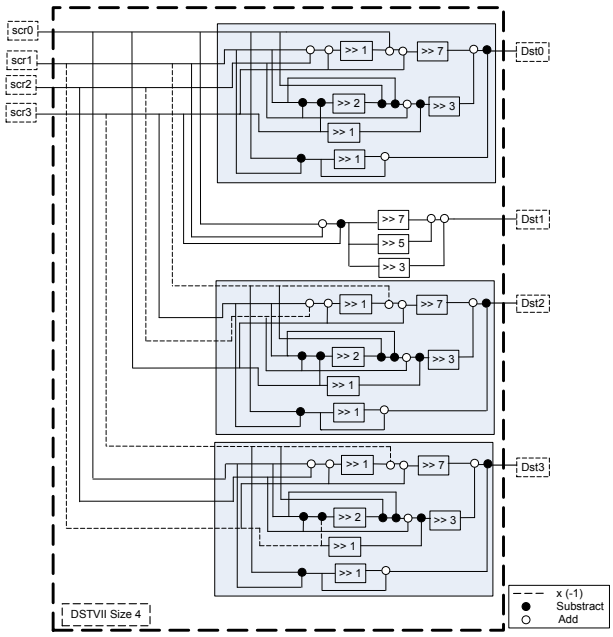

Fig. 3. Proposed hardware architecture for DST-VII size 4

D. DCT-VIII transform

$$\text{M4}=\begin{pmatrix} 336 & 296 & 219 & 117 \\ 296 & 0 & -296 & -296 \\ 219 & -296 & -117 & 336 \\ 117 & -296 & 336 & -219 \end{pmatrix}$$

Compared to the DST-VII matrix, DCT-VIII one has the same coefficients but in inverse order for each row. Then, with only inversing the inputs order and assigning the appropriate coefficients signs we can easily benefit from DST-VII (size 4) architecture to implement the DCT-VIII transform type with no additional computational complexity.

E. DCT-V transform

$$\text{M4}=\begin{pmatrix} 194 & 274 & 274 & 274 \\ 274 & 241 & -86 & -349 \\ 274 & -86 & -349 & 241 \\ 274 & -349 & 241 & -86 \end{pmatrix}$$

Coefficients redundancies in $M4$ allowed us to propose an architecture, presented in Fig. 4. This architecture consists in one block charged to provide the 1^{st} output. This block requires 7 additions and 6 shifts. A second block (*block2-DCT5*) is used three times in parallel modifying only the inputs order to obtain the other outputs using 14 additions and 7 shifts.

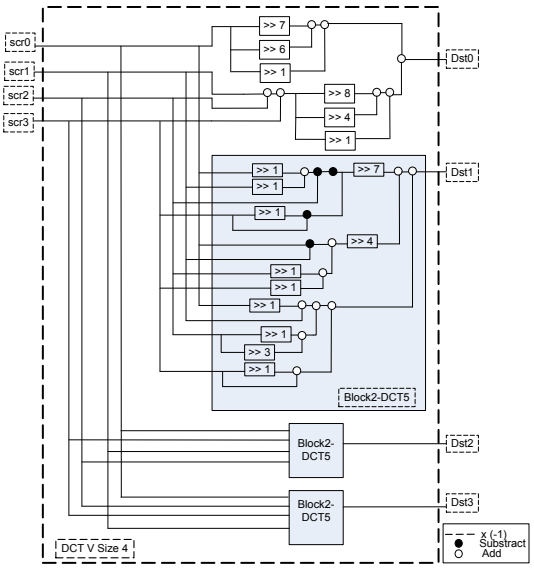

Fig. 4. Proposed hardware architecture for DCT-V size 4

IV. Experimental and synthesis results

The five transform types architectures were specified in VHDL language using *Modelsim* software tool.

A. Simulation results

1) State-machine based architecture:

A unified circuit that encompasses the different types for size 4 is designed. In DST-VII (Fig. 3), DCT-VIII- and DCT-V (Fig. 4) architectures, there was a component that has been used three times at once. Taking advantage of this property, we noticed that introducing a state-machine managing those blocks operation sequentially, according to a definite control unit, would provide a considerable computational complexity reduction compared to the first proposed implementation method. Table I shows the performance of both solutions (no state-machine 1^{st} method and with state-machine 2^{nd} method) in terms of required operations (adders and shifts) and clock cycles. We can notice that the number of operations is mainly related to the values of the coefficients matrices. It also gives us a clear idea how using state machines (2^{nd}) enables about 45% reduction in computational complexity at the expense of higher clock cycles.

TABLE I. Performance of the 1^{st} (no state machine) and 2^{nd} (with state machine) implementation methods

	Adders		Shifts		Clock cycles	
	1^{st} meth.	2^{nd} meth.	1^{st}	2^{nd}	1^{st}	2^{nd}
DCT-II	24	24	16	16	4	4
DST-I	28	16	24	12	5	7
DST-VII	46	18	21	9	5	9
DCT-VIII	0	0	0	0	6	10
DCT-V	49	21	27	13	5	9
Total	147	79	88	50	-	-
Reduction	–	46 %	–	44%	–	–

Table II presents a comparison between the original design using multiplication operations and the multiplierless proposed architectures under Stratix-III EP3SL340F1760C4 FPGA device using the software tool *Quartus II 9.0*. The synthesis results show that the proposed methods offer a wide logic elements optimization. The low reserved registers number of the original solution is due to the lack of intermediate computations by shift and addition operations. On the other hand, the decrease in the number of operations (Table I, 2^{nd} method) was reflected automatically to the hardware cost due to the reuse of the hardware resources offered through state machines. Table II shows also that the two proposed designs have reached respectively 318 MHz and 285 MHz as maximum processing frequencies.

TABLE II. Synthesis results of the proposed architectures

	original	**1^{st} method**	**2^{nd} method**
Pins	303 (41%)	295 (40%)	295 (40%)
ALUTS	10116 (5%)	6842 (3%)	3802 (1%)
Registers	994 ($<$ 1%)	3603 (1%)	2219 ($<$ 1%)
Frequence	356 MHz	318 MHz	285 MHz

2) Discussion:

As the design architecture (1^{st} method) provides parallel output generation, we can reach further optimization by adding the pipelining operation. This latter is more interesting when it concerns computing of multiple rows of size 4 (4x4 blocks size as an example). Table III shows the clock cycles required for computing 1D 4x4 blocks size of each transform type. Of course, the more rows are computed the more interesting pipeline becomes, especially for future works (other transform sizes) as the AMT adopts asymmetric block sizes (4x8 , 4x16, 4x32, 4x64..).

TABLE III. Clock cycles of pipelined 4x4 blocks

	DCT-II	DST-I	DST-VII	DCT-VIII	DCT-V
Clock cycles 1^{st}	11	12	12	13	12
Clock cycles 2^{nd}	11	22	30	30	30

The required frequency to compute 4K videos at 30 frames per second (fps) is 202 Mhz (13*3840*2160*30 / 4*4). Therefore, the first proposed design, reaching an operational frequency up to 318 Mhz, can easily support 4K coding assuming only 4x4 block sizes. With involving other larger sizes and more computational complexity we may have different results. On the other hand, although the state machine architecture provided computational and area optimization, it can't really benefit from the pipelining operation because of data dependencies and lack of total parallel output generation. As an example, DST-I requires 22 clock cycles to compute 4x4 block instead of 28 (7*4) without pipeline. This can support 2K resolution video processing at 60 fps.

V. Conclusion

The hardware implementation AMT design involving DCT-II, DST-I, DST-VII, DCT-VIII and DCT-V transform types is has been investigated in his paper. A multiplierless implementation of these transforms for size 4 is presented. Concerning the hardware architectures, two aspects have been introduced and compared in this study. The first one is based on eliminating the multiplication operations (with adders and shifters) using symetric and sparse matrices. The second one provides a more optimized computational complexity using state-machines to preserve the hardware resources, while presenting some additional clock cycles.

As future works, this study can be extended for the other larger sizes where these methods can be combined in order to provide an efficient tradeoff between hardware cost and required clock cycles.

REFERENCES

[1] H. I. Recommendation, "High Efficiency Video Coding (HEVC)," *23008-2 MPEG-H Part 2*, 2013.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[3] J. R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the Coding Efficiency of Video Coding Standards x2014;Including High Efficiency Video Coding (HEVC)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669–1684, Dec 2012.

[4] T. K. Tan, R. Weerakkody, M. Mrak, N. Ramzan, V. Baroncini, J.-R. Ohm, and G. J. Sullivan, "Video Quality Evaluation Methodology and Verification Testing of HEVC Compression Performance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 76–90, Jan. 2016. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7254155

[5] Joint-Video-Exploration-Team, "Several jvet meetings," https://jvet.hhi.fraunhofer.de/.

[6] C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. J. Sullivan, " High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Update 4 of Encoder Description," *Joint Collaborative Team of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11 on Video Coding TCSVT.2003.815165,22nd Meeting*, Oct 2015.

[7] E. Alshina, A. Alshin, K. Choi, and M. Park, " Performance of JEM 1 tools analysis," *Document JVET-B0044 3rd 2nd JVET Meeting:San Diego, CA, USA*, Febrary,2016.

[8] N. Sidaty, W. Hamidouche, P. Philippe, and O.Deforges, "Emerging Video Coding Performance: 4K Quality Monitoring," in *International Conference on Quality of Multimedia Experience (QoMEX 2017)*, Erfurt, Germany, 2017.

[9] J. Chen, E. Alshina, G. J. Sullivan, J.-R. Ohm, and J. Boyce., "Algorithm Description of Joint Exploration Test Model 5 (JEM 5)," *Output document: JVET-E1001-v2 of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 5th Meeting:*, pp. 12–20, Janaury 2017.

[10] X. Zhao, J. Chen, M. Karczewicz, L. Zhang, X. Li, and W-J.Chien, "Enhanced multiple transform for video coding," *Data Compression Conference (DCC)*, pp. 73–82, March 2016.

[11] T. Biatek, V. Lorcy, P. Castel, and P. Philippe, "Low-complexity adaptive multiple transforms for post-hevc video coding," in *2016 Picture Coding Symposium (PCS)*, Dec 2016, pp. 1–5.

[12] Shen, S., Shen, W. .and Fan, Y., Zeng, and X., "A unified 4/8/16/32-point integer IDCT architecture for multiple video coding standards," *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 788–793, 2012.

[13] P. K. Meher, S. Y. Park, B. K. Mohanty, K. S. Lim, and C. Yeo, "Efficient Integer DCT Architectures for HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 1, pp. 168–178, Jan 2014.

[14] A. Ahmed and M. Shahid, "N Point DCT VLSI Architecture for Emerging HEVC Standard," *VLSI Design*, pp. 1–13, 2012.

[15] A. Kammoun, F. Belghith, H. Loukil, and N. Masmoudi, "An Optimized and Unified Architecture Design for H.265/HEVC 1-D Inverse Core Transform," *IEEE IPAS 2016, International Image Processing Applications and Systems Conference*, November 2016.

[16] R.-K. Chivukula and Y.-A. Reznik, "Fast Computing of Discrete Cosine and Sine Transforms of Types VI and VII," *Proceedings of the SPIE*, vol. 8135, no. 5, Sep 2011.

[17] M. Puschel and J.-M.-F. Moura, "Algebraic Signal Processing Theory: Cooley-Tukey Type Algorithms for DCTs and DSTs," *IEEE Transactions on Signal Processing*, vol. 22, no. 12, April, 2008.