POLITECNICO DI TORINO
Repository ISTITUZIONALE

Approximate Arai DCT Architecture for HEVC

Original
Approximate Arai DCT Architecture for HEVC / Renda, Giovanni; Masera, Maurizio; Martina, Maurizio; Masera, Guido. - STAMPA. - 1:(2017), pp. 133-136. (Intervento presentato al convegno New Generation of CAS tenutosi a Genova (ITA) nel 7-9 settembre) [10.1109/NGCAS.2017.38].

Availability:
This version is available at: 11583/2689773 since: 2017-11-07T14:14:01Z

Publisher:
IEEE

Published
DOI:10.1109/NGCAS.2017.38

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

17 April 2024

Approximate Arai DCT Architecture for HEVC

Giovanni Renda, Maurizio Masera, Maurizio Martina and Guido Masera
Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy 10129
Email: giovanni.renda@studenti.polito.it, maurizio.masera@polito.it, maurizio.martina@polito.it, guido.masera@polito.it

***Abstract*—This work describes an approximate DCT architecture for the High Efficiency Video Coding (HEVC) standard. Since the standard requires to support multiple block sizes, architectures based on exact implementation require a relevant amount of hardware resources, namely multipliers and adders. This work aims to reduce the amount of hardware resources while keeping the rate-distortion performance nearly optimal. To achieve this goal, this work exploits an exact factorization of the DCT of size N = 8, which is then extended to obtain approximate DCTs of size N = 16 and N = 32. Simulation and implementation results prove that the proposed approximate solution features a complexity reduction with respect to exact one of more than 43% with an average rate-distortion performance loss of 4.74% for the worst-case (all-intra) configuration.**

I. Introduction

Transform coding is an important feature of the High Efficiency Video Coding (HEVC) standard [1], which allows to improve coding efficiency by removing redundancy between residuals after the intra/inter prediction stage. To achieve higher coding efficiency, transform coding in HEVC has been improved at the expense of much higher complexity with respect to H.264/AVC [2]. Indeed the HEVC standard specifies Discrete Cosine Transform (DCT) block sizes from 4 × 4 up to 32 × 32 [3]. Moreover, the rate-distortion optimization leads to an increased complexity at the encoder side [4], which puts severe throughput requirements on the design of the DCT module of an HEVC encoder.

Therefore, several hardware architectures to compute the variable-size DCT in HEVC have been proposed in the last years. Dias *et al.* [5] exploited a 2D systolic array to implement the DCT as matrix-vector multiplication, thus supporting multiple standards. On the other hand, Meher *et al.* [6] designed an efficient integer DCT architecture for HEVC by relying on the odd-even decomposition of the DCT matrix and by reusing the core $N/2$-point DCT for the even computation of the N-point DCT. Moreover, to achieve high throughput, such an architecture includes and additional $N/2$-point DCT unit, so that it computes $32/N$ N-point DCTs concurrently. However, these approaches require a lot of hardware resources as they implement exactly the DCT matrix specified by the HEVC standard [3]. For this reason, approximation has been introduced as a new paradigm to efficiently compute the DCT in video coding applications, by trading complexity for rate-distortion performance loss [7]. Several approximations of the 8-point DCT have been derived by manipulating the coefficients and by simplifying the DCT matrix. A collection of these methods is available in [8]. To extend the transform size from 8 to 32, Jridi *et al.* [9] proposed a generalized algorithm and a reconfigurable hardware architecture. In particular, their solution relies on factorizing the DCT matrix by using the odd-even decomposition and processing both the even and the odd parts with the approximate 8-point DCT proposed by Cintra and Bayer [10], which requires 22 additions only. However, this approach results in poor rate-distortion performance because of the rough approximation of the core 8-point DCT.

The aim of this work is to explore the design space generated by the adoption of an exact low-complexity factorization of the 8-point DCT to be used as core module in the generalized algorithm proposed in [9]. Among the different DCT factorizations described in [11], the one proposed by Arai *et al.* [12] has been chosen in this work, because it needs 5 multiplications and 29 additions only. Thus, this work implements the 8-point Arai-based DCT and exploits it as the core unit for the reconfigurable architecture proposed in [9]. This solution allows to investigate different trade-offs between rate-distortion performance and hardware cost by changing the number of bits used to represent the internal multiplication constants.

The paper is organized as follows. Section II briefly overviews the generalized DCT algorithm for HEVC proposed in [9] and shows the proposed low-complexity fixed-point DCT based on the Arai factorization [12]. The proposed architecture is shown in Section III while Section IV shows the results of the rate-distortion performance analysis and the hardware details. Finally, Section V concludes the paper.

II. Approximate DCT Algorithm

This Section recalls the generalized algorithm proposed in [9], which is used to approximate the 16-point and the 32-point DCT by employing the inner core 8-point DCT. Moreover, the DCT factorization proposed in [12] is briefly summarized.

A. Generalized DCT Algorithm

According to [11], the DCT matrix $\mathbf{C}_N$ is defined as:

$$[\mathbf{C}_N]_{i,j} = \epsilon_i \sqrt{\frac{2}{N}} \cos\frac{\pi(2j+1)i}{2N} \quad 0 \leq i,j \leq N-1, \quad (1)$$

where $\epsilon_0 = 1/\sqrt{2}$ and $\epsilon_i = 1$ for $i > 0$.

By applying the odd-even decomposition, the DCT matrix in (1) can be rewritten in the following form:

$$\mathbf{C}_N = \frac{1}{\sqrt{2}} \cdot \mathbf{P}_N \cdot \left[\begin{array}{cc} \mathbf{C}_{\frac{N}{2}} & \mathbf{0}_{\frac{N}{2}} \\ \mathbf{0}_{\frac{N}{2}} & \mathbf{S}_{\frac{N}{2}} \end{array} \right] \cdot \mathbf{B}_N, \quad (2)$$

where $\mathbf{C}_{\frac{N}{2}}$ is the $N/2$-order DCT matrix and $\mathbf{S}_{\frac{N}{2}}$ is composed of the first $N/2$ coefficients of the odd rows of $\sqrt{2} \cdot \mathbf{C}_N$. The $\mathbf{P}_N$ matrix is the alternating permutation matrix defined by Φ_N, which assigns the $\phi_N(k)$-th input to the k-th output:

$$\Phi_N = \begin{pmatrix} k \\ \phi_N(k) \end{pmatrix} \quad k = 0, \ldots, N-1, \tag{3}$$

with

$$\phi_N(k) = \begin{cases} \lfloor \frac{k}{2} \rfloor & k \text{ even} \\ \lfloor \frac{k}{2} \rfloor + \frac{N}{2} & k \text{ odd} \end{cases}, \tag{4}$$

and $\mathbf{B}_N$ is the input butterfly matrix, which is defined using $\mathbf{I}_N$ and $\mathbf{J}_N$, the N-order identity and anti-diagonal identity matrices, respectively:

$$\mathbf{B}_N = \begin{bmatrix} \mathbf{I}_{\frac{N}{2}} & \mathbf{J}_{\frac{N}{2}} \\ \mathbf{I}_{\frac{N}{2}} & -\mathbf{J}_{\frac{N}{2}} \end{bmatrix}. \tag{5}$$

The DCT approximation proposed in [9] modifies (2) by substituting the inner $\mathbf{C}_{\frac{N}{2}}$ and $\mathbf{S}_{\frac{N}{2}}$ matrices with the approximated $\hat{\mathbf{C}}_{\frac{N}{2}}$:

$$\hat{\mathbf{C}}_N = \frac{1}{\sqrt{2}} \cdot \mathbf{P}_N \cdot \begin{bmatrix} \hat{\mathbf{C}}_{\frac{N}{2}} & \mathbf{0}_{\frac{N}{2}} \\ \mathbf{0}_{\frac{N}{2}} & \hat{\mathbf{C}}_{\frac{N}{2}} \end{bmatrix} \cdot \mathbf{B}_N. \tag{6}$$

This recursion applies for N = 32 and 16, while for N = 8, $\hat{\mathbf{C}}_8$ is the one proposed in [10]. This matrix has been generated by rounding the original DCT matrix in (1) as $\hat{\mathbf{C}}_8 = \lfloor 2 \cdot \mathbf{C}_8 \rceil$, thus showing null multiplicative complexity, since the only constituent elements are 0, 1 or -1. Moreover, it is worth noting that $\hat{\mathbf{C}}_N$ is orthogonalizable. Therefore, for each $\hat{\mathbf{C}}_N$ it is possible to compute $\mathbf{D}_N$ as:

$$\mathbf{D}_N = \sqrt{\left(\hat{\mathbf{C}}_N \times \left(\hat{\mathbf{C}}_N\right)^T\right)^{-1}}, \tag{7}$$

which can be integrated in the quantization process of the video encoder, thus not introducing any additional computation when calculating the DCT.

B. Low-Complexity Arai-Based Factorization

The 8-point DCT factorization proposed by Arai *et al.* [12] has been considered in this current work, because of its very low complexity. It is derived by only computing the real part of the first eight output coefficients of the 16-point Discrete Fourier Transform (DFT) factorized using the Winograd algorithm [13]. The DCT inputs $x(k)$ are connected to the DFT inputs $X(k)$ according to the following mapping:

$$X(k) = \begin{cases} x(k) & 0 \leq k \leq 7 \\ x(15-k) & 8 \leq k \leq 15 \end{cases}, \tag{8}$$

while the output DCT coefficients $y(n)$ are calculated by applying the final normalization:

$$y(n) = \frac{2 \cdot \epsilon_n \cdot \Re[Y(n)]}{\cos \frac{n\pi}{16}} \tag{9}$$

where $Y(n)$ is the DFT output and $\epsilon_0 = 1/\sqrt{2}$ and $\epsilon_n = 1$ for $n > 0$.

TABLE I
REAL AND INTEGER VALUES OF α_i FOR N_q=8.

Coefficient	Real value	Integer Value N_q=8
α_1, α_3	$\cos \frac{4\pi}{16}$	$181x = x + (((3x << 4) - 3x) << 2)$ $3x = (x << 2) - x$
α_2	$\cos \frac{2\pi}{16} - \cos \frac{6\pi}{16}$	$138x = (5x + (x << 6)) << 1$ $5x = x + (x << 2)$
α_4	$\cos \frac{2\pi}{16} + \cos \frac{6\pi}{16}$	$334x = ((8x - x) + (5x << 5)) << 1$ $5x = x + (x << 2) \quad 8x = x << 3$
α_5	$\cos \frac{6\pi}{16}$	$97x = x + (((x << 2) - x) << 5)$

TABLE II
ACCURACY MEASURES AND ARITHMETIC COMPLEXITY OF DIFFERENT 8-POINT DCT APPROXIMATIONS.

Algorithm	ε	MSE	C_g	η	Mult	Add
Exact DCT	0	0	8,83	93,99	64	56
HM-16.12 [15]	0,0020	0,0009	8,83	93,82	22	28
CB-2011 [10]	1,7945	0,9800	8,18	87,42	0	22
Arai [12] N_q=4	0,0529	0,0693	8,98	92,83	5	29
N_q=5	0,0137	0,0093	8,97	93,65		
N_q=6	0,0025	0,0010	8,88	93,94		
N_q=7	0,0030	0,0020	8,86	93,81		
N_q=8	0,0006	0,0002	8,84	93,97		

In this work, we propose an hardware-oriented implementation of the Arai-based DCT, where all the internal multiplications have been substituted with add-and-shift blocks [14]. Indeed, the sinusoidal factors (α_i) defined by the Winograd algorithm (see Table I) have been scaled on N_q fractional bits, thus generating a space of DCT approximations which trade accuracy for hardware complexity. As it can be observed, $0 < \alpha_i < 2$ for i = 1,. . . ,5, thus only one bit is required to represent the integer part. In order to analyze the effectiveness of each approximation, the matrix proximity metrics and the transform-related measures defined in [8], [11], namely the error energy (ε), the Mean Square Error (MSE), the transform coding gain (C_g) and the transform efficiency (η) metrics have been calculated and compared with the exact 8-point DCT, the integer DCT used in HEVC [15] and the low complexity approximation proposed in [10] and exploited in [9]. Table II reports these accuracy measures and the arithmetic complexity in terms of number of multiplications and additions as well. As shown in the Table, using more than N_q = 4 bits to represent the internal coefficients, the fixed-point DCT based on the Arai factorization approximates very well the exact DCT and the one employed in the reference software of the HEVC standard [15]. Moreover, it is observed that the CB-2011, which was adopted in [9], shows the minimum arithmetic complexity while providing worse accuracy measures. The integer values of the sinusoidal coefficients of the Arai-based DCT have been calculated as $\lfloor \alpha_i \cdot 2^{N_q} \rceil$. Table I lists the real and the integer values of each coefficient, as well as the add-and-shift implementation with N_q = 8 [14]. Moreover, it is worth noting that the final normalization in (9) does not affect the structure of the factorization. Therefore, it has been integrated in the quantization step, thus not requiring any additional multiplication in the transform stage.

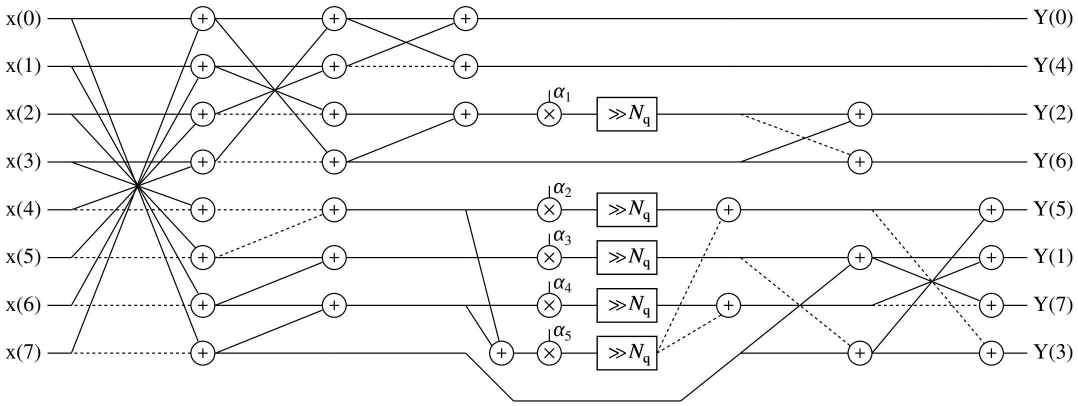

Fig. 1. Proposed 8-point DCT architecture. Dashed lines represent inputs to be subtracted.

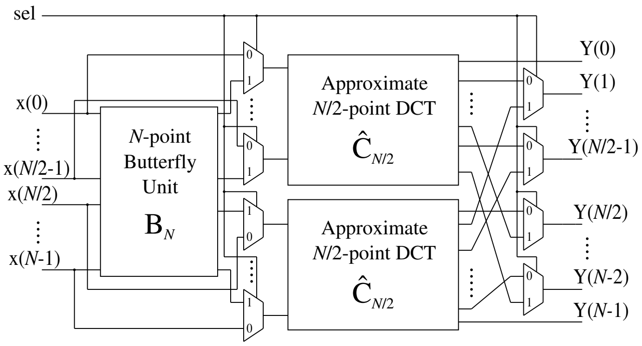

Fig. 2. Reconfigurable architecture for DCT of size N = 16 and 32, which uses the core approximated 8-point DCT architecture [9].

III. Proposed Architecture

The proposed 8-point DCT architecture based on the Arai factorization is depicted in Fig. 1. It is composed of 29 adders/subtracters and 5 integer multipliers, which are followed by right-shift of N_q bits. It is worth noting that N_q is fixed, therefore the shift operation is implemented by simple wiring, thus not incurring additional hardware overhead. Also, since α_i are constants, multipliers are simplified as adders and wired shift operations (see Table I).

The adopted reconfigurable architecture which implements the generalized algorithm proposed in [9] is reported in Fig. 2. This recursive structure applies for $\hat{\mathbf{C}}_{32}$ and $\hat{\mathbf{C}}_{16}$, where the inner $\hat{\mathbf{C}}_8$ is the 8-point DCT architecture of Fig. 1. By adopting this strategy, the architecture is able to concurrently process 32 samples which are grouped according with the transform size. To this purpose, the architecture makes use of two approximate $N/2$-point DCT units plus the additional hardware of the N-point butterfly unit, which implements $\mathbf{B}_N$, and of the banks of multiplexers used to reconfigure the architecture to support variable-size computation. Specifically, the first bank between the butterfly unit and the approximate DCTs, is used to skip the computation of the butterflies when sel=0, i.e. two $N/2$-point DCTs have to be executed in parallel. When sel=1 the results produced by the butterfly unit become the inputs of the two approximate $N/2$-point DCT cores. On the other hand, the multiplexers at the output implement the permutation defined by the $\mathbf{P}_N$ matrix when sel=1; otherwise they select the outputs of the two $N/2$-point DCTs without reordering.

This architecture has been designed to work in a Folded structure. As proposed in [6], the Folded implementation reuses the one-dimensional DCT unit for computing the two-dimensional DCT. Indeed, the DCT block is fed with 32 samples taken row-wise from the $32/N$ input data blocks of size $N \times N$ in the first N cycles. Then, the output of the DCT block is scaled according to [3] and stored row-wise in a transposition buffer. During the following N cycles, successive columns are read from the transposition buffer and fed to the DCT unit, which produces the final output DCT coefficients. The whole computation needs $2N$ clock cycles to compute $N \times N$ results independently of the DCT size N, thus resulting in a constant throughput of 16 samples/cycle. Since the DCT block in Fig. 2 is used for both the row-wise and column-wise computations, its inputs and outputs are represented with 16 bits, as specified in [3], while the internal signals have been sized in order to avoid overflow.

IV. Implementation Results

As observed in Section II, different N_q values define a design space, which has been explored both in terms of coding efficiency and hardware complexity. In order to assess the rate-distortion performance of the modified encoder [16], the Arai-based factorization of the DCT has been integrated into the HEVC reference software model HM-16.12 [15]. Only the forward transform of sizes from 8 to 32 has been changed, whereas the 4-point DCT and the decoder implements the original HEVC transform, as specified in [3]. Simulations have been performed on all the sequences taken from classes A, B, C, D, E and F according to three encoding configurations, namely All-Intra, Low-Delay and Random-Access main. Quantization parameters equal to 22, 27, 32 and 37 have been used according to [17]. The computational resources were provided by HPC@POLITO (http://www.hpc.polito.it). Table III reports the Bjøntegaard Delta rate (BD-rate) metric [18], averaged on all the sequences of the same class and measured

TABLE III
BD-RATE [%] COMPARISON OF THE PROPOSED DCT IN HEVC WITH RESPECT TO THE WORK IN [9].

Class	[9]	N_q=4	N_q=5	N_q=6	N_q=7	N_q=8
			All-Intra			
A	14.82	9.89	9.61	9.59	9.52	9.53
B	9.71	6.89	6.76	6.74	6.70	6.70
C	3.77	2.19	2.10	2.10	2.07	2.06
D	3.76	2.15	2.07	2.06	2.03	2.02
E	7.72	5.59	5.50	5.49	5.46	5.45
F	2.15	1.39	1.31	1.31	1.27	1.27
All	7.07	4.74	4.61	4.60	4.56	4.56
			Low-Delay			
A	-	-	-	-	-	-
B	6.84	5.44	5.46	5.44	5.43	5.43
C	3.96	2.72	2.72	2.68	2.68	2.68
D	2.87	1.55	1.54	1.52	1.56	1.49
E	5.40	4.15	4.19	4.29	4.26	4.10
F	3.63	2.62	2.63	2.67	2.62	2.57
All	4.61	3.36	3.37	3.38	3.37	3.32
			Random-Access			
A	12.14	7.55	7.50	7.41	7.42	7.39
B	8.17	6.29	6.28	6.27	6.23	6.23
C	2.67	1.66	1.55	2.14	2.14	2.11
D	2.98	2.12	2.08	2.11	2.09	2.05
E	-	-	-	-	-	-
F	2.67	1.91	1.86	1.89	1.84	1.83
All	5.92	4.00	3.95	3.93	3.92	3.90

between curves obtained by encoding the video sequences with the modified algorithm and with the original partial-butterfly approach used in the HM. The table compares the proposed solutions for $4 \leq N_q \leq 8$ with the method employed in [9]. As expected, the coding efficiency degrades when lowering the number of bits used to represent α_i, thus showing an average loss of about 4.74% in the worst-case configuration (All-Intra). However, the rate-distortion loss is lower than the one achieved in [9] independently of N_q. On the other hand, Table IV lists the clock frequency (f_{CK}), the gate count and the power consumption (P) of the proposed architectures when synthesized using a 90-nm standard cell library. As expected, architectures with small N_q achieve higher clock frequencies and reduced power consumption and gate count, thus showing similar hardware complexity as the work in [9] while providing improved rate-distortion performance. When used in the folded structure and synthesized for the same frequency of [6] (f_{CK}=187 MHz), the proposed architectures show gate counts of 116.9 K and 107.7 K for N_q equal to 8 and 4, respectively. Thus, they feature a complexity reduction ranging from 43% to 48% with respect to the implementation of the HEVC non-approximated DCT (208 K) [6].

V. CONCLUSION

This paper has proposed a reconfigurable approximate DCT architecture for HEVC, which exploits the Arai factorization to reduce the hardware cost of the core 8-point DCT. The rate-distortion analysis and the hardware synthesis results show that the proposed implementations outperform the one presented in [9] by providing similar complexity reduction with better rate-distortion performance and reduce the complexity with respect to the exact DCT in [6] at the cost of small quality loss.

TABLE IV
COMPARISON OF RECONFIGURABLE DCT ARCHITECTURES FOR HEVC.

Design		f_{CK} (MHz)	Gates	P (mW)
Meher *et al.* [6]		187	131.0 K	-
Jridi *et al.* [9]		322	30.6 K	7.21
Proposed	N_q=4	324	29.2 K	6.63
	N_q=5	318	30.8 K	6.91
	N_q=6	314	33.5 K	8.07
	N_q=7	300	34.5 K	7.90
	N_q=8	285	38.4 K	8.53

REFERENCES

[1] G. Sullivan, J. 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.

[2] F. Bossen, B. Bross, K. Shring, and D. Flynn, “HEVC Complexity and Implementation Analysis,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec 2012.

[3] M. Budagavi, A. Fuldseth, G. Bjontegaard, V. Sze, and M. Sadafale, “Core Transform Design in the High Efficiency Video Coding (HEVC) Standard,” *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1029–1041, Dec 2013.

[4] M. Masera, L. Re Fiorentin, M. Martina, G. Masera, and E. Masala, “Optimizing the Transform Complexity-Quality Tradeoff for Hardware-Accelerated HEVC Video Coding,” in *Proc. 2015 Conference on Design and Architectures for Signal and Image Processing*, Sept 2015, pp. 1–6.

[5] T. Dias, N. Roma, and L. Sousa, “High performance multi-standard architecture for DCT computation in H.264/AVC High Profile and HEVC codecs,” in *Proc. 2013 Conference on Design and Architectures for Signal and Image Processing*, Oct 2013, pp. 14–21.

[6] P. Meher, S. Y. Park, B. Mohanty, K. S. Lim, and C. Yeo, “Efficient Integer DCT Architectures for HEVC,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 1, pp. 168–178, Jan 2014.

[7] A. Madanayake, R. Cintra, V. Dimitrov, F. Bayer, K. Wahid, S. Kulasekera, A. Edirisuriya, U. Potluri, S. Madishetty, and N. Rajapaksha, “Low-Power VLSI Architectures for DCT/DWT: Precision vs Approximation for HD Video, Biomedical, and Smart Antenna Applications,” *IEEE Circuits Syst. Mag.*, vol. 15, no. 1, pp. 25–47, Firstquarter 2015.

[8] U. Sadhvi Potluri, A. Madanayake, R. Cintra, F. Bayer, S. Kulasekera, and A. Edirisuriya, “Improved 8-Point Approximate DCT for Image and Video Compression Requiring Only 14 Additions,” *IEEE Trans. Circuits Syst. I*, vol. 61, no. 6, pp. 1727–1740, Jun 2014.

[9] M. Jridi, A. Alfalou, and P. Meher, “A Generalized Algorithm and Reconfigurable Architecture for Efficient and Scalable Orthogonal Approximation of DCT,” *IEEE Trans. Circuits Syst. I*, vol. 62, no. 2, pp. 449–457, Feb 2015.

[10] R. J. Cintra and F. M. Bayer, “A DCT Approximation for Image Compression,” *IEEE Signal Process. Lett.*, vol. 18, no. 10, pp. 579–582, Oct 2011.

[11] V. Britanak, P. C. Yip, and K. R. Rao, *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations*. Academic Press, 2007.

[12] Y. Arai, T. Agui, and M. Nakajima, “A Fast DCT-SQ Scheme for Images,” *The Transactions of the IEICE*, vol. E 71, no. 11, pp. 1095–1097, Nov 1988.

[13] S. Winograd, “On Computing the Discrete Fourier Transform,” *Mathematics of computation*, vol. 32, no. 141, pp. 175–199, 1978.

[14] A. Dempster and M. MacLeod, “Use of minimum-adder multiplier blocks in FIR digital filters,” *IEEE Trans. Circuits Syst. II*, vol. 42, no. 9, pp. 569–577, Sep 1995.

[15] Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, *HM-16.12 Reference Software*. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.12/

[16] G. Renda and M. Masera. (2017, May) Approximate DCT Architecture for HEVC: Modified HM-16.12 Software and VHDL Code. [Online]. Available: http://personal.det.polito.it/maurizio.masera/downloads.html

[17] F. Bossen, *Common test conditions and software reference configurations*, document JCTVC-L1100, Geneva, CH, Jan 2013.

[18] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD Curves*, document VCEG-M33, ITU-T SG16/Q6, Austin, TX, Apr 2001.