Approximate DCT Design for Video Encoding Based on Novel Truncation Scheme

Heming Sun, *Member, IEEE*, Zhengxue Cheng, *Student Member, IEEE*, Amir Masoud Gharehbaghi, Shinji Kimura, and Masahiro Fujita, *Member, IEEE*

***Abstract*— This paper presents an energy- and area-efficient architecture for approximated discrete cosine transform (DCT). Due to the good compression ability, DCT is widely exploited in signal processing. However, it is computationally intensive especially for large transform sizes. In this paper, we have reduced the computation cost of DCT by truncating a couple of least significant bits (LSB), most significant bits (MSB), and zero columns. First, considering that the contribution of LSBs is weakened because of the final right shift operation, we have eliminated the computation process for some LSBs. For the addition of the remaining LSBs, a parallel carry propagation adder is proposed to reduce the calculation latency. Second, owing to the phenomenon that high-frequency components are quite small in natural scenes, a couple of MSBs are selectively truncated according to their positions. Third, quantization is taken into account for the system-level optimization. The quantized results of all-zero columns are utilized to skip the column transforms afterward. The experimental results show that at most 32% area consumption and 60% power consumption can be reduced compared with the originally accurate DCT, while the compression efficiency loss caused by the DCT approximation is negligible for High Efficiency Video Coding.**

***Index Terms*— DCT, approximate computing, HEVC, VVC, truncation.**

I. Introduction

DISCRETE cosine transform (DCT) is significantly important and widely utilized in signal compression. Through DCT, spatial signals can be converted to frequency domain, in which each signal is decomposed of components at different frequencies. For natural signals, most of the energy is concentrated in the low-frequency regions. Therefore, the low frequency components are remained while the signals at high frequency region could be sparsed for the compression. In the latest video compression standard High Efficiency Video Coding (HEVC) [1], DCT is adopted as an essential coding feature. The larger transform size has better energy concentration ability which can yield higher compression ratio. As depicted in [2], the introduction of large transforms can contribute to around 10% coding efficiency improvement for 1080p and larger videos. Therefore, in order to realize higher compression performance, HEVC supports larger DCT than previous standards such as H.264. As reported in [3], the largest DCT size can reach 32×32 in HEVC, while it is only 8×8 in H.264. However, in order to perform larger DCT, the computational complexity drastically increases. In the recent HEVC intra encoder [4], transforms account for about 53% logical gate counts. Therefore, a low-cost DCT architecture is desired.

Many low-cost designs [5]–[10] have been developed for H.264 transform. However, due to the fact that the transform size is different, those methods cannot be straightly applied in HEVC. In order to reduce the cost of DCT and inverse DCT (IDCT) in HEVC, many works have been exploited in [11]–[16]. They utilized Chen's algorithm [17] so that one 2N-point transform could be decomposed into even and odd parts, and then even part shares the same architecture with the N-point transform. By recursively decomposing the even parts, the number of multiplications and additions could be significantly reduced. However, despite of using Chen's algorithm, the remaining hardware cost is still high. Therefore, some approximated methods are incorporated to further reduce the cost. Meher *et al.* [18] presented a pruned transform architecture in which the least significant bits (LSB) are truncated. Potluri *et al.* [19] proposed an approximated 8-point DCT which only required 14 additions. Masera *et al.* [20] factorized a complete DCT to Walsh-Hadamard transform and given rotations which could be adaptively filtered to save the power consumption. Jridi *et al.* [21], [22] developed an orthogonal approximation architecture where the approximation for 2N-point DCT could be derived from N-point DCT. Bouguezel *et al.* [23] created a binary matrix to replace the original matrix to achieve a multiplier-less architecture. The above approximated methods can be categorized into two groups by the value of transform matrix. The works with modified matrix such as [19] and [21]–[23] are placed in the first category. In [19] and [23], the coefficients of the modified transform matrix are either ±1 or 0. In [21] and [22], the coefficients of the Odd part are altered to the same coefficients of the Even part. One major merit of the matrix modification is to simplify the calculation. However, it also results in significant coding efficiency degradation. For [22], the BD-psnr loss is more than 0.3dB. Therefore, in order to keep the coding performance, we have chosen to be in the

Manuscript received June 12, 2018; revised October 3, 2018, October 27, 2018, and November 7, 2018; accepted November 8, 2018. This work was supported in part by Grants-in-Aid for Scientific Research from JSPS and a research fund from NEC. This paper was recommended by Associate Editor G. Masera. *(Corresponding author: Heming Sun.)*

H. Sun is with the Waseda Research Institute for Science and Engineering, Tokyo 169-8555, Japan (e-mail: hemingsun@aoni.waseda.jp).

Z. Cheng is with the Graduate School of Fundamental Science and Engineering, Waseda University, Tokyo 169-8555, Japan (e-mail: zxcheng@asagi.waseda.jp).

A. M. Gharehbaghi and M. Fujita are with the VLSI Design and Education Center, University of Tokyo, Tokyo 113-0032, Japan (e-mail: amir@cad.t.u-tokyo.ac.jp; fujita@ee.t.u-tokyo.ac.jp).

S. Kimura is with the Graduate School of Information, Production and Systems, Waseda University, Kitakyushu 808-0135, Japan (e-mail: shinji_kimura@waseda.jp).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2018.2882474

second category which does not involve the transform matrix alteration.

In addition to the above works which utilizes the feature of approximation, some other designs took the advantage of zero-element skipping to save the hardware cost. Sun *et al.* [24] proposed a zero-skipping method for the memory read/write operations in the system of de-quantization and inverse transform. Tikekar *et al.* [2] skipped the computation of zero rows/columns based on the information of last non-zero flag. These methods are efficient for IDCT, however, they cannot be directly applied to DCT. That is because the inputs of IDCT are frequency-domain coefficients that are full of zero elements, therefore, the positions zero elements are already known before performing IDCT. However, for DCT, the inputs are spatial residuals that are regularly non-zero. Hence, the zero skipping for DCT requires the estimation of the zero element positions.

In the video encoding applications, both DCT and IDCT are amenable to adopt approximation. In this manuscript, we start the approximation from DCT due to its larger hardware cost in some encoding implementations. As presented in many real-time encoder architectures such as [4] and [25], DCT is usually used in both mode decision and reconstruction loop, while IDCT is generally only adopted in the reconstruction loop. That is because the lack of inverse transform only influences the distortion estimation which will not incur significant performance loss. As a result, in [4], DCT occupies 369K gates while IDCT consumes about 211K gates.

In this paper, we focus on an energy- and area-efficient DCT design. In order to maintain a good coding performance, we aim to manage the approximation without altering the transform matrix. The contributions of the paper are as follows.

1) LSB truncation (LT): We eliminate the carry propagation from the 6-th LSB to the 7-th LSB by truncating the lowest 6 bits. For the addition of remaining LSBs, a parallel processing scheme for the carry generation is presented.

2) MSB truncation (MT): An unfixed MSB truncation scheme according to the position of the transformed coefficient is developed.

3) Zero column truncation (ZCT): Quantized results are utilized to determine the activities of the column transform. Once there is an all-zero quantized column, the process for the following columns is skipped.

The rest of the paper is organized as follows. In Section II, the decomposition for DCT is demonstrated. The approximated schemes for LT, MT and ZCT are presented in Section III, IV and V respectively. In Section VI, we provide the extensibility to support various sizes of DCT. The results and analysis of coding efficiency and hardware cost are given in Section VII, followed by the conclusion and future work in Section VIII.

II. DCT Decomposition

A. Decomposition of 32-Point DCT

Taking 32-point DCT as an example, according to Chen's algorithm, it can be decomposed into three stages as shown in Fig. 1. In the first stage, a butterfly structure is employed. After that, the core computation can be divided into the Even and Odd part. There are 16 inputs and outputs for both parts. In fact, the Even part can be further decomposed into the even-even (EE) part and even-odd (EO) part. Finally, the results of the Even and Odd parts are permutated.

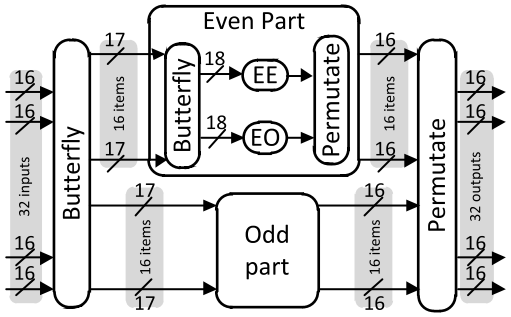

Fig. 1. Decomposition of 32-point DCT. Our proposals in this paper are focused on Odd and EO part.

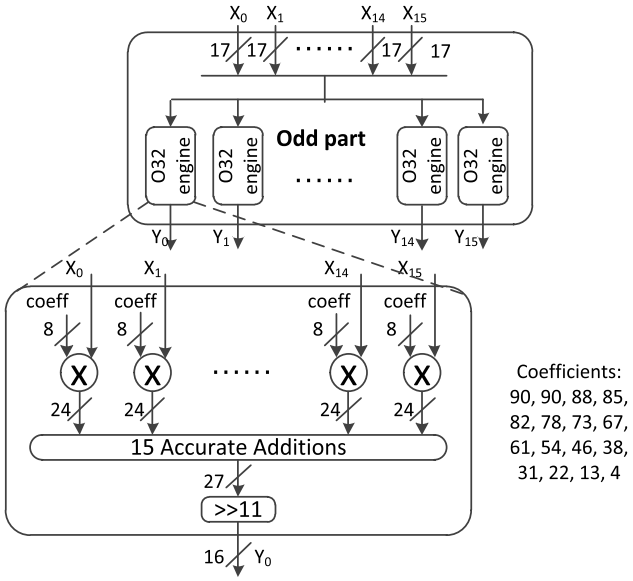

Fig. 2. The circuit for the Odd part of 32-point DCT.

When the input bit-width of 32-point DCT is 16, the input bit-width of the Odd part becomes 17 after the butterfly structure, and the circuit of the Odd part is shown in Fig. 2. For each input, it is multiplied by a constant coefficient defined by the HEVC transform matrix. The largest magnitude of a constant coefficient is 90, therefore, the corresponding product is 24-bit. 15 additions are required to calculate the summation of 16 products. Finally, an 11-bit right shift operation is executed to generate the 16-bit final result.

The circuit of the EO part is shown in Fig. 3, which is similar to that of the Odd part. The input bit-width becomes 18 after the butterfly structure. The constant coefficients are different from that of the Odd part. The largest magnitude is 90. Seven additions are required to calculate the summation of eight products. Finally, an 11-bit right shift operation is executed to generate the final results.

Between the Even and Odd part, the latter takes the majority of the hardware resources. As shown in Fig. 2, when calculating one result of the Odd part, 16 multiplications are required. Overall it used 256 multiplications to generate 16 results of

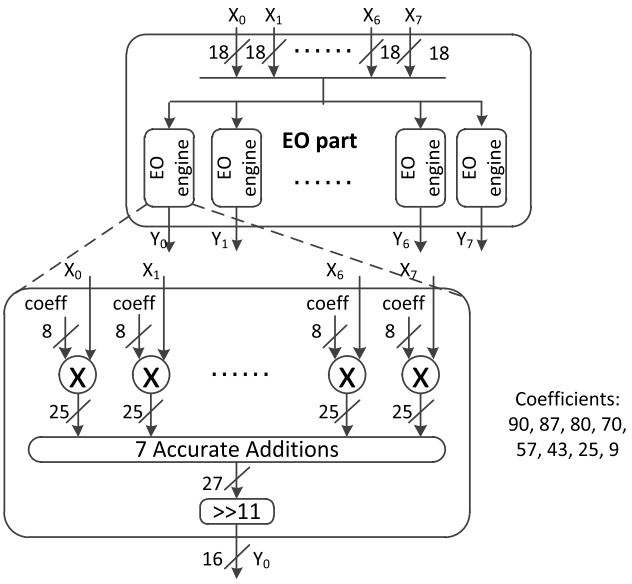

Fig. 3. The circuit for the EO part of 32-point DCT.

the Odd part. For the Even part, it is further decomposed to the EE and EO part, as shown in Fig. 1. For the EO part, it uses 64 multiplications to produce eight results as shown in Fig. 3. Similarly, for EE part, it also costs 64 multiplications. Overall, the Even part cost 128 multiplications, which is smaller than the Odd part. It is noted that in fact, the EE part can be further decomposed so that the number of demanded multiplications for EE part can be smaller than 64.

The architecture of 32-point DCT has been presented in the above. In fact, the architecture of the other transform sizes is very similar. For instance, for 16-point DCT, the Odd part resembles to the EO part of 32-point DCT, as shown in Fig. 3. The only difference is that the input becomes 17-bit and the final right shift turns into 10-bit rather than 11-bit.

B. Design Strategy

The structure of 32-point DCT is shown in Fig. 1. Our proposal is focused on Odd and EO part. The circuit of Odd and EO part is shown in Fig. 2-3, we can see that the essence of Odd and EO calculation can be summarized as two stages: sum-of-product (SOP) and right-shift operation. At first, because of the right-shift, some LSB operations in SOP have limited effect on the final result. Therefore, we propose LSB truncation for SOP part which is proposed in Section III. Secondly, the magnitude of the high-frequency component is usually small, so it is not necessary to use the entire bits. Some MSBs can be truncated. The MSB-oriented methods are given in Section IV. Finally, quantized transformed results of high-frequency components are usually zero, so the DCT calculation for these zero elements can be skipped, as shown in Section V.

III. Least Significant Bit Truncation (LT)

In the previous section, we have shown that the Odd part consumes more hardware resources than the Even part. Therefore, we start to reduce the hardware cost from the Odd part. From the final stage of Fig. 2, we can see that the 11-bit right shift is implemented for the Odd part. Therefore, the LSB's contribution to the final results is likely to be ignored due to the right shift. In order to eliminate the needless calculation, truncation is employed for a couple of LSBs.

A. Arithmetic-Free Scheme for Six-LSBs

We determined the number of LSB truncation bits based on an error analysis. As described in Section II, the Odd part is actually a SOP operation, each product can be written as $P_i = T_i X_i$ where T_i are the transform coefficients and X_i are the input. The final output (summation) can be calculated by the following equation where N is the number of products.

$$\mathrm{y} = \sum_{i=0}^{N-1} T_i X_i = \sum_{i=0}^{N-1} P_i \tag{1}$$

When truncating LSBs for the product term, an accurate P_i will become an approximated term $\hat{P}_i$. Therefore, the final output can be represented as

$$\hat{y} = \sum_{i=0}^{N-1} \hat{P}_i \tag{2}$$

For each product term, given that the precision of the approximated product is p1, thus the truncation error is bounded by 2^{-p1}. Therefore, the sum of error comes from the truncation is

$$E_{trun} = \sum_{i=0}^{N-1} (P_i - \hat{P}_i) \leq 2^{log_2^N - p1} \tag{3}$$

In addition to the above truncation error, there is another error coming from rounding. It is because the precision of the product is larger than the precision of the final output, so we have to round the value. Given that the precision of final output is p2, thus the rounding error is bounded by 2^{-p2-1}. Therefore, the overall error is given in Eq. (4).

$$\mathrm{E} = E_{trun} + E_{round} \leq 2^{log_2^N - p1} + 2^{-p2-1} \tag{4}$$

To achieve the last-bit accuracy, the maximum tolerant error is 2^{-p2} as reported in [26]. Therefore, we have to satisfy the condition in Eq. (5).

$$2^{log_2^N - p1} + 2^{-p2-1} \leq 2^{-p2} \tag{5}$$

To satisfy Eq. (5), p1 − p2 should be not smaller than $log_2^N + 1$, which means at least we need $log_2^N + 1$ more bits for the product compared with the final output. For 32-point DCT, N is 16, so 5 more bits are required for the precision of the product than the final output. In the original DCT, there are 11 more bits for the precision of the product than the final output. Therefore, we decide to truncate 6 LSBs for products in our method.

To explore the effect of LSB truncation on the output, the 24-bit production is divided into MSB and LSB part, which take (24-N) and N bits, respectively as shown in Fig. 4. $\mathrm{P_n}$ (n = 1, 2, 3 . . . 16) are 16 products and N is the number of truncated LSBs for each product. The summation of 16 truncated LSBs is calculated and marked as B[N+3:0]. The summation of the remaining MSBs are left shift by N-bit

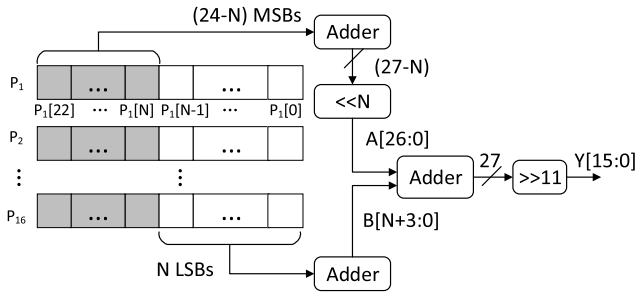

Fig. 4. Splitting the 24-bit product into (24-N) MSB and N LSB (N is the truncate bits). It is noted that the summation of the above adder is (27-N) bit rather than (28-N) bit because of the magnitudes of constant coefficients.

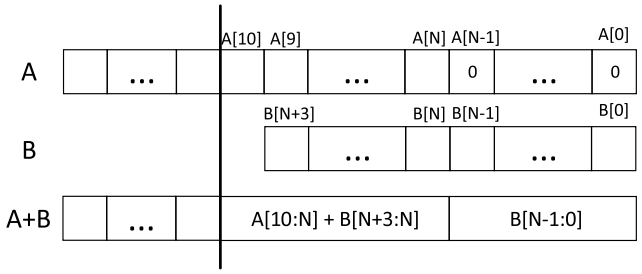

Fig. 5. The summation of A and B, A is the summation of (24-N) MSBs and B is the summation of N LSBs.

and marked as A[26:0]. The final result can be calculated by the following equation.

$$\mathrm{Y} = ((\mathrm{A}[26:0] + \mathrm{B}[\mathrm{N}+3:0]) \gg 11) \tag{6}$$

The summation of A and B is shown in Fig. 5. The sum will be right shifted for 11 bits to generate the final result. N LSBs of A are all zero. If the summation of A[10:N] and B[N+3:N] is larger than $(2^{11-\mathrm{N}}-1)$, there is a carry propagation to the 12th LSB which will be retained after 11-bit right shift. In this case, the truncation of B[N+3:N] will influence the final result.

The probability can be formulized in Eq. (8) where the denominator represents the overall cases (different values) of A[10:N] and B[N+3:N], while the numerator means the number of cases that the summation of A[10:N] and B[N+3:N] is larger than $2^{11-N}-1$. For the denominator, since the range of A[10:N] is from 0 to $2^{11-N}-1$, and the range of B[N+3:N] is from 0 to 2^4-1. So the number of overall combinational cases is $2^4 \times 2^{11-N}$. For the numerator, the analysis is divided in two situations. One situation is $2^{11-N}-1 \geq 15$, and the other situation is $2^{11-N}-1 < 15$.

In the first situation, if B[N+3:N] is 1, the summation of A[10:N] and B[N+3:N] will be larger than $2^{11-N}-1$ only when A[10:N] is $2^{11-N}-1$. If B[N+3:N] is 1, the summation of A[10:N] and B[N+3:N] will be larger than $2^{11-N}-1$ when A[10:N] is $2^{11-N}-1$ or $2^{11-N}-2$. In fact, for each specific value β of B[N+3:N], there are β cases that he summation of A[10:N] and B[N+3:N] will be larger than $2^{11-N}-1$. Therefore, overall there are $\sum_{i=0}^{15} i$ cases that satisfied the condition.

In the second situation, when B[N+3:N] is within the range of $[0, 2^{11-N}-1]$, for each specific value β of B[N+3:N], there are β cases that he summation of A[10:N] and B[N+3:N] will be larger than $2^{11-N}-1$. When B[N+3:N] is larger than $2^{11-N}-1$, the summation of A and B will always be larger than $2^{11-N}-1$ regardless of the value of A[10:N]. Therefore, the overall cases can be calculated in Eq. (7).

TABLE I

THE PROBABILITY THAT THE TRUNCATION WILL INFLUENCE THE FINAL RESULT FOR THE ODD PART

N	7	6	5	4
Prob. (%)	46.88	23.44	11.72	5.86

$$(\sum_{\beta=0}^{2^{11-N}-1} \beta) + \sum_{\beta=2^{11-N}}^{15} 2^{11-N}) = 2^{10-N} \times (31 - 2^{11-N}) \tag{7}$$

As a result, the probability could be written as Eq. (8).

$$P(N) = \begin{cases} \dfrac{\sum_{i=0}^{15} i}{2^4 \times 2^{11-N}} 2^{11-N} - 1 \geq 15 \\ \dfrac{2^{10-N} \times \left(31 - 2^{11-N}\right)}{2^4 \times 2^{11-N}} \\ \quad = \dfrac{31 - 2^{11-N}}{2^5} 2^{11-N} - 1 < 15 \end{cases} \tag{8}$$

The results in the case of N being from 4 to 7 are shown in Table I. When N is 6, the error probability is 23.44%.

In order to provide a more generic formula, we generalize Eq. (8) to Eq. (9). There are three parameters which will influence the probability, the first one is the number of inputs and the second one is how many bits are right shifted. In addition, the number of truncated LSBs will also affect the probability. We use M, S and N to represent the above three parameters, respectively.

Therefore, our problem is to calculate the probability that the summation of A[S-1:N] and B[N+$\log_2^{\mathrm{M}}$ − 1:N] is larger than $2^{S-N}-1$. The range of A[S-1:N] is from 0 to $2^{S-N}-1$, and the range of B[N+$\log_2^{\mathrm{M}}$ − 1:N] is from 0 to M-1. Therefore, the number of combinational cases of A[S-1:N] and B[N+$\log_2^{\mathrm{M}}-1$:N] is $M \times 2^{S-N}$, as shown in the denominator of Eq. (9). For the numerator, the analysis is divided to two situations, one is $2^{S-N}-1 \geq (M-1)$, and the other is $2^{S-N}-1 < (\mathrm{M}-1)$. In the first situation, the number of cases that the summation is larger than $2^{S-N}-1$ is $\sum_{i=0}^{\mathrm{M}-1} i$. In the second situation, the number of cases that the summation is larger than $2^{S-N}-1$ is equal to $2^{S-N-1} \times \left(2M - 1 - 2^{S-N}\right)$. Therefore, the probability can be written in Eq. (9).

$$P(M, N, S) = \begin{cases} \dfrac{\sum_{i=0}^{M-1} i}{M \times 2^{S-N}} \quad 2^{S-N} - 1 \geq M - 1 \\ \dfrac{2^{S-N-1} \times \left(2M - 1 - 2^{S-N}\right)}{M \times 2^{S-N}} \\ \quad = \dfrac{2M - 1 - 2^{S-N}}{2M} \quad 2^{S-N} - 1 < M - 1 \end{cases} \tag{9}$$

It is noted that each bit of A[10:0] and B[N+3:0] has equal probability to be 0 or 1. As shown in Fig. 5, A[10:0] and B[N+3:0] will decide the value of 11 LSBs. The histogram for the value of 11 LSBs is shown in Fig. 6. We can see that the probability is almost the same for each value within the range of 0 and 2047. Therefore, each bit of A[10:0] and B[N+3:0] is supposed to be 0 or 1 at 50% probability.

After truncating 6 LSBs for products, the bit-width of products are reduced from 24-bit to 18-bit. Correspondingly,

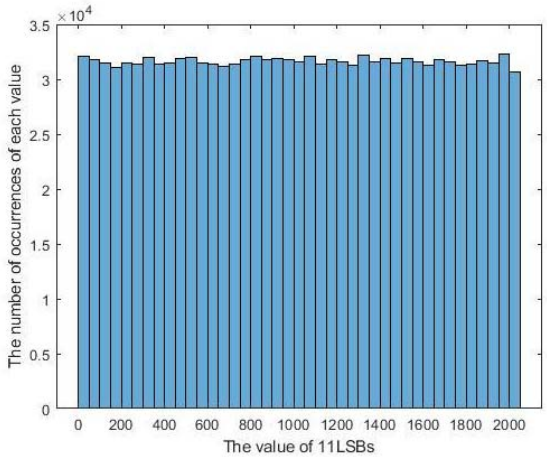

Fig. 6. The histogram of the value of 11LSBs.

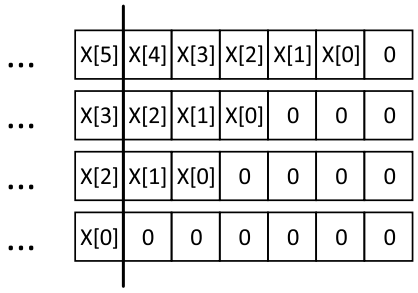

Fig. 7. Converting the multiplication with 90 to the addition of four items.

TABLE II
THE NUMBER OF TRUNCATED LSBS FOR THE INPUT WHEN MULTIPLYING BY DIFFERENT COEFFICIENTS

Coefficient	90	88	85	82	78	73	67
Truncated LSBs	3	2	4	2	4	3	5
Coefficient	61	54	46	38	31	22	13
Truncated LSBs	4	4	4	4	5	4	4

the right shift is reduced from 11-bit to 5-bit in the end. However, each product is the multiplier results of the inputs (X_n in Fig. 2) and constant coefficients. When generating product, the arithmetic operation of 6 LSBs is still required since the carry propagation to the 7-th LSB will also influence the final results. In order to further reduce the cost of the multiplication of inputs and constant coefficients, we truncate the LSBs of inputs and the number of truncated bits is dependent on the magnitude of coefficient. For example, when the coefficient is 90, the multiplication can be represented as the addition of four items as shown in Fig. 7. In the origin, the carry depends on the summation of the four items. If 3 LSBs are truncated, there will be no carry propagation to the 7-th LSB. For the other constant coefficients, the number of truncated LSBs for the inputs is listed in Table II. Note that when the constant coefficient is 4, the multiplication with the input is just left shift operation. Therefore, we do not truncate the LSBs of the inputs that are multiplied with 4.

B. Low-Latency Carry Estimation From Five-LSBs

After truncating the 6 LSBs, the bit-width of products is reduced from 24 to 18. Correspondingly, the right shift in the final step is reduced from 11 to 5. Therefore, for the addition of 16 products, the lowest 5 bits still have a limited effect on the final output, as only the carry value will be kept due to the right shift operation. Therefore, in this subsection, a low-latency carry estimation method for the lowest 5 bits is proposed.

TABLE III
TRUTH TABLE OF 1-bit OR OPERATION AND ADDITION

A[i]	B[i]	A[i]|B[i]	A[i] + B[i]
1	1	1	10
1	0	1	1
0	1	1	1
0	0	0	0

Suppose the 5-bit LSBs of two products are A[4:0] and B[4:0]. At first, the item of A[4]&B[4] is used for the carry estimation as shown in the following equation.

$$C1 = A[4] \,\&\, B[4] \tag{10}$$

If A[4]&B[4] is one, the carry propagation to the 6-th LSB is absolutely one which means that the precision rate (the fraction of retrieved cases that are relevant to the query) is 100%. However, for all the combinations of A and B, there are 496 cases that will lead to the carry propagation while there are only 256 cases when A[4]&B[4] is one. The recall rate (the fraction of the relevant cases that are successfully retrieved) is only 52% by only using A[4]&B[4] for the estimation. The definition of precision rate and recall rate can be obtained by the following equations.

$$\text{precision rate} = \frac{\textit{number of cases satisfy (10) and the accurate carry propagation to the 6-th LSB is 1}}{\textit{number of cases satisfy (10)}} \tag{11}$$

$$\textit{recall rate} = \frac{\textit{number of cases satisfy (10) and the accurate carry propagation to the 6-th LSB is 1}}{\textit{number of cases that accurate carry propagation to the 6-th LSB is 1}} \tag{12}$$

In order to increase the recall rate, the results of OR operation of A and B are used to predict the carry propagation as shown in the following equation.

$$C2 = (A|B == 5'b11111) \tag{13}$$

A|B is the bit-wise OR operation of A and B, e.g., A = 10101, B = 01110, then A|B = 11111. According to the truth table in Table III, we can obtain that if (A[i]|B[i]) is equal to 1, then (A[i]+B[i])≥1. Thus if (A | B==5'b11111), it means that $(A + B) \geq \sum_{i=0}^{4} 1 \cdot 2^i = 5'b11111$, which is very likely to produce a carry propagation to the 6-th LSB. Therefore, the final referee for the carry propagation is obtained by either of the above two situations, as shown in the following equation.

$$C = C1|C2 = \left(A \mid B == 5'b11111\right) | (A[4] \,\&\, B[4]) \tag{14}$$

The number of cases that satisfies the condition in Eq. (13) can be calculated by $\sum_{n=0}^{5} \left(\frac{5!}{(5-n)! \times n!} \times 2^n\right)$ which is equal

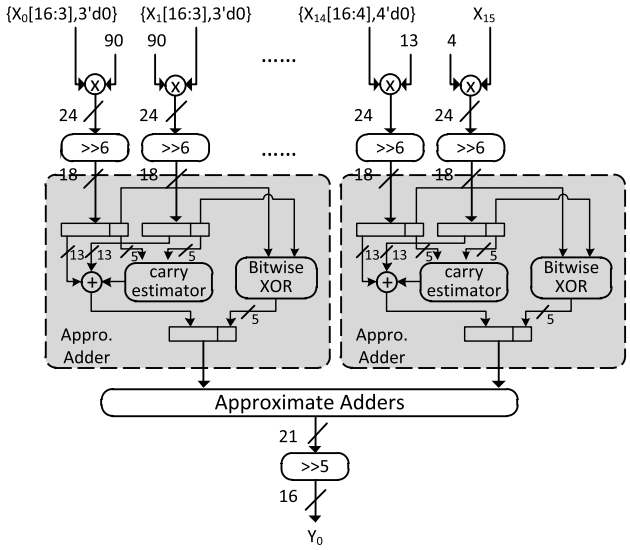

Fig. 8. Process of proposed LSB truncation.

to 243. However, among these 243 cases, there are 32 cases that the actual carry is not one. In addition, the number of intersections of the condition in Eq. (10) and Eq. (13) is calculated by $\sum_{n=0}^{4}\left(\frac{4!}{(4-n)!\times n!}\times 2^{n}\right)$ which is equal to 81. Overall 418 (256+243-81) cases are selected and 386 (418-32) cases are correct. Therefore, the recall rate is about 78% ($\frac{386}{496}$) and precision rate is about 92% ($\frac{386}{418}$). After estimating the carry propagation from 5-LSBs, the additions of the remaining bits are conducted by common accurate adders.

C. Whole Process of LSB Truncation

Taking Y_0 as an example, the whole process of LSB truncation is shown in Fig. 8. For Y_0, X_0 and X_1 multiplied with the constant coefficient 90. According to Table II, 3 most LSBs of X_0 and X_1 can be truncated. X_{14} is multiplied with 13, therefore, we truncate 4 most LSBs of X_{14}. For X_{15}, it is multiplied with 4, therefore, no truncation is performed. 16 products are right shift for 6 bits before addition. For the 16 right-shifted products, 15 proposed approximated adders are adopted to calculate the summation. For each addition, 5 LSBs of two addends are used to estimate the carry according to Eq. (14), and the estimated carry bit is used as the carry-in bit for the addition of high-order bits. For the 5 LSBs, the result is generated by the XOR operation, which can be processed in parallel with the operation of high-order bits. Finally, the summation is right shift for 5 bits to output the final result.

IV. Most Significant Bit Truncation (MT)

Since the dynamic range of high frequency components for transform is usually quite small, bit truncation for most significant bits (MSBs) is conducted in this section.

DCT results are the weighted summation of residual pixels which can be regarded as identically distributed random variables. Therefore, according to the central limit theorem, DCT results follow the Gaussian distribution. The mean is zero and the variance is proportional to the variance of pixels in the block according to [27]. When the variance of pixels in the

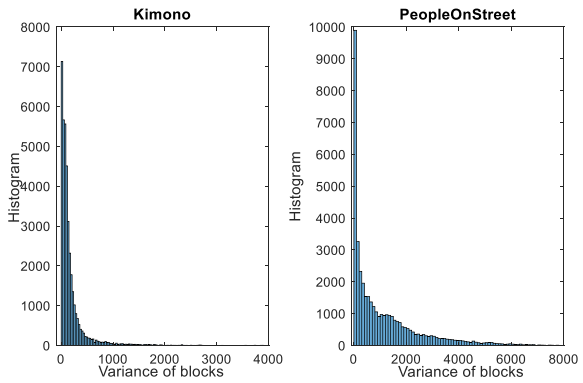

Fig. 9. Histograms for the variance of blocks.

block is changing, the probability density function (PDF) can be written as the following equation

$$\mathrm{p}\left(F_{x,y}\right)=\int_{0}^{\infty} p\left(F_{x,y}\,|\,\sigma^{2}\right) p\left(\sigma^{2}\right)\mathrm{d}\left(\sigma^{2}\right) \tag{15}$$

where $F_{x,y}$ is the DCT result, σ^2 is the variance of pixels in the block and $p\left(F_{x,y}\,|\,\sigma^{2}\right)$ is a Gaussian distribution whose PDF is represented in the following equation.

$$\mathrm{p}\left(F_{x,y}\,|\,\sigma^{2}\right)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{F_{x,y}^{2}}{2\sigma^{2}}} \tag{16}$$

For the variance of pixels, the distribution is observed as the exponential distribution as reported in [27]. However, the observation in [27] is based on 8×8 blocks, we demonstrate that the exponential distribution is also suitable for the variance of 32×32 blocks as shown in Fig. 9.

The PDF of the exponential distribution can be represented as

$$\mathrm{p}\,(\mathrm{x})=\begin{cases}\lambda e^{-\lambda x} & x\geq 0\\ 0 & x<0\end{cases} \tag{17}$$

Since the variance is non-negative, so Eq. (17) can be simplified as $\mathrm{p}\left(\sigma^{2}\right)=\lambda e^{-\lambda\sigma^{2}}$. Substituting Eq. (16) and Eq. (17) into Eq. (15), we can obtain $\mathrm{p}\left(F_{x,y}\right)$ in Eq. (18) which is exactly the PDF of Laplacian distribution.

$$\mathrm{p}\left(F_{x,y}\right)=\frac{\sqrt{2\lambda}}{2}e^{-\sqrt{2\lambda}|F_{x,y}|} \tag{18}$$

For Laplacian distribution, the probability for small values is much higher than that for large values. Therefore, we believe that it is feasible to do MSB truncation for DCT results.

When one row is processed in one clock cycle, it takes 32 clock cycles to compute 32 rows. In the N-th clock cycle, the output Y_n corresponds to the result of N-th row and (2∗n+1)-th column. The results of the low-frequency components are much larger than that of high-frequency components. Therefore, we do the experiment to verify the maximum required bits for each index of outputs. Three sequences *PeopleOnStreet*, *Kimono* and *RaceHorses* given in [29] are used for test, and the results are shown in Table IV. The coefficient of index 0 is the lowest frequency component in the Odd part so that the value is likely to be large. In order to prevent from the performance loss, no MSB truncation is applied for the index 0.

TABLE IV

THE NUMBER OF TRUNCATED MSBS FOR THE FINAL OUTPUTS OF THE ODD PART

Index of outputs (n)	0	1-2	3	4-8	9-12	13-15
# of truncated MSBs	0	2	3	4	5	6

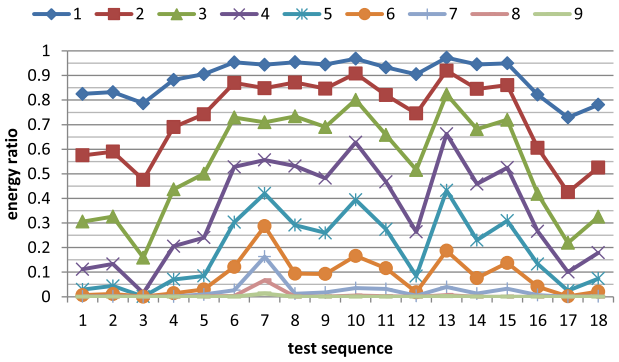

Fig. 10. Energy loss ratio in case of different retaining LSBs.

Since the training set only includes three sequences, therefore, in some extreme cases, the number of remaining bits after MT might not be enough. In order to guarantee the influence of the MSB truncation, we evaluate the energy loss of each bit for 18 test sequences. The evaluation is shown in the following equation.

$$\text{Energy}_{bit\_order} = \frac{\sum_{(|T_{ij}|>2^{bit\_order}-1)} \left(|T_{ij}| - (2^{bit\_order}-1)\right)^2}{\sum T_{ij}^2} \quad (19)$$

where T_{ij} is transformed coefficients, bit_order ranges from 1 to 15. $\left(|T_{ij}| - (2^{bit\_order}-1)\right)^2$ represents the energy that has to be represented by the MSBs larger than bit_order. In the other words, it can represent the energy loss if the MSBs larger than bit_order are truncated. The number of truncated MSBs for each index is shown in Table IV. Take index 13 of the Odd part as an example which truncated 6 MSBs. The energy loss ratio of remaining 9 LSBs is shown in Fig. 10. We can see that if only the most LSB is remained after MT, more than 80% energy is lost for all the 18 test sequences. The energy loss becomes smaller with more remaining LSBs. When 9 LSBs are kept, there is nearly no energy loss. Therefore, we believe that the truncation of 6 MSBs for the index 13 will not arouse evident energy loss. In addition to the energy loss, we also evaluate the MT's influence to the final coding efficiency as described in the experimental results and analyze the robustness in the following.

The number of truncated MSB for each index has been decided in Table IV. We first analyze the robustness of the MSB truncation for index 1. After that, for the following indexes, we give a statistical observation to show the robustness. According to the reference [27], the distribution of DCT results follows Laplacian distribution whose probability density function is shown in Eq. (20)

$$\text{f}\left(\text{x} \mid \mu, \text{b}\right) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right) \quad (20)$$

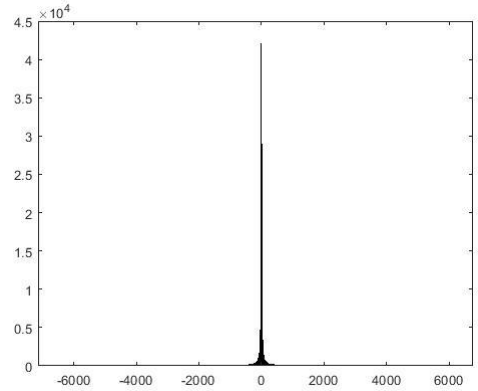

Fig. 11. The histogram of the DCT result of the index 1 for *PeopleOnStreet*.

where μ represents for the mean that is zero due to the unitary nature of DCT, and b stands for the variance. The corresponding inverse cumulative distribution function is given in Eq. (21).

$$F^{-1}(p) = \mu - b \times sgn\,(p-0.5) \times \ln(1-2|p-0.5|) \quad (21)$$

For the position of index 1, 2 MSBs are truncated thus the data range of [−8192,8191] can be kept. The corresponding covered probability can be calculated by Eq. (22) where b is the variance.

$$\text{P(b)} = \int_{x=-8192}^{8191} f\,(x|b) \quad (22)$$

In order to estimate the value of b, we analyze the DCT result for the sequence of *PeopleOnStreet*. The histogram is shown in Fig. 11 where the horizontal axis is the value and the vertical axis is the frequency. We can see that the distribution appears to be the Laplacian distribution. We use MATLAB to do the fitting and the estimated b is 17.58. After that, we can calculate $\int_{x=-8192}^{8191} f\,(x|17.58)$ that is more than 99.99%, which means that the data range after MSB truncation can cover 99.99% confidence interval.

In order to ensure the robustness against any input sequence, we estimate the value of b in the worst case. As described in [27], the spatial correlation plays an important role in determining the width of the distribution which is proportional to the magnitude of b. A block with smaller spatial correlation has the distribution with larger b. Therefore, we use the random noise as each pixel in order to make the spatial correlation approach 0. When pixels are random noise, the histogram of the transformed results is shown in Fig. 12. We can see that compared with Fig. 11, the distribution is more dispersed. We use MATLAB to fit the Laplace distribution and the estimated magnitude of b is 609.76. By substituting the result of b into Eq. (22), we can find that the result is still larger than 99.99%. Therefore, the truncation of 2 MSBs are trusty for the index 1.

For the following indexes, we also estimate the value of b for the sequence *PeopleOnStreet*, the results are shown in Fig. 13. We can see that the value of b becomes smaller for larger index. Therefore, it is reasonable that more MSBs could be truncated for larger indexes.

Although the data range after the MSB truncation can cover 99.99% confidence interval, there is still a little bit probability

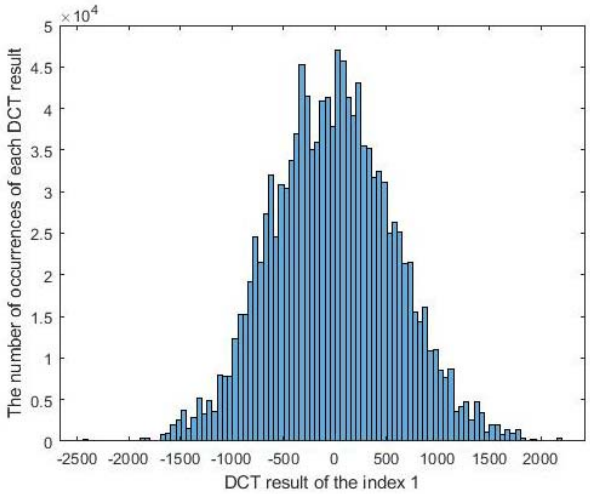

Fig. 12. The histogram of the DCT result of the index 1 for random noise inputs.

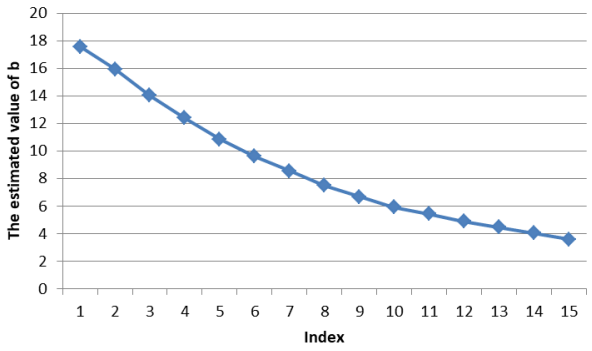

Fig. 13. The estimated value of b for different indexes.

that the MSB truncation will incorrectly truncate the sign bit thus lead to sign errors. The original output is 16-bit, marked as $O_{15}O_{14}\ldots O_1O_0$. Take index1 as an example, if 2MSBs are truncated, the value becomes $O_{13}O_{12}O_{11}\ldots O_1O_0$. Therefore, if the original output is larger than 8191, or smaller than -8192, there will be a sign error after the truncation. The magnitude of the error is related with the original value. For example, if the original value is within the range of [8192,16383], $O_{15}O_{14}O_{13}$ is 001. Before the truncation, O_{13} means 8192. However, after truncation, O_{13} becomes the sign bit thus it means -8192. Therefore, the magnitude of the error is 16384. Similarly, we can know the magnitude of error for the other ranges, as shown in Table V. The expectation of the error can be calculated by the following equation

$$\begin{aligned} \mathrm{E}\left[\text{error}\right] &= 16384 \times \left(\int_{8192}^{24575} f\left(x\right) + \int_{-24576}^{-8193} f\left(x\right) \right) \\ &\quad + 32768 \times \left(\int_{24576}^{32767} f\left(x\right) + \int_{-32768}^{-24577} f\left(x\right) \right) \\ &= 0.024 \end{aligned} \tag{23}$$

where f(x) is the probability density function shown in Eq. (20). We can calculate that the expectation is only 0.024, which is negligible.

It is noted that the MSB truncation for the final output is effective to the whole system. For example, when 6 MSBs are truncated, only 15 bits are required before the right shift. Therefore, the bit-width of the addition operations can also be reduced to 15.

TABLE V
THE MAGNITUDE OF ERROR AND RANGE FOR ALL THE ERROR CASES

$O_{15}O_{14}O_{13}$	Magnitude of error	Range
001	16384	[8192,16383]
010	16384	[16384,24575]
011	32768	[24576,32767]
110	16384	[-16384,-8193]
101	16384	[-24576,-16385]
100	32768	[-32768,-24577]

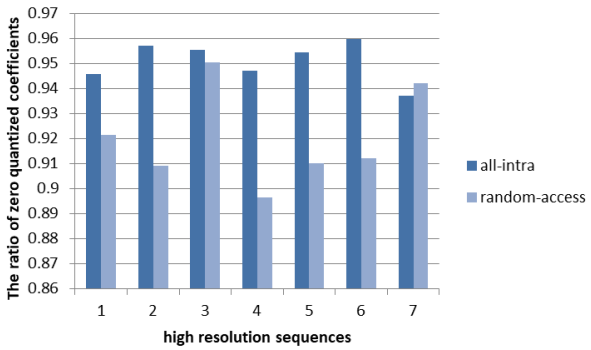

Fig. 14. The ratio of zero quantized coefficients for high-resolution sequences in case of *Maxbitrate*.

V. ZERO COLUMN TRUNCATION (ZCT)

In the video coding, quantization is executed after forward transform. Since the results of high-frequency transformed coefficients are usually very small for large transform sizes. Small high-frequency coefficients are most likely to become zero after quantization. We have analyzed the ratio of zero quantized coefficients for 32×32 at first.

In HEVC, *Maxbitrate* is defined in [1], which represents the upper limit for the number of bits per second. For each specific test sequence, more non-zero quantized coefficients are likely to occur in the case of *Maxbitrate*. Therefore, we test the results for the large-resolution sequences as shown in Fig. 14. The encoding configuration is "random-access, main" defined in [29], for each test sequence, we traverse the quantization parameter (QP) to find one which can achieve the *Maxbitrate*. We can see that there are considerable zero quantized coefficients. For each sequence, about 90% or even more quantized coefficients are zero.

The ratio of quantized coefficients has been ensured in the above. Moreover, once one column of quantized coefficients is all-zero, it is highly possible that the columns in the right side are all-zero or have small values. We evaluate the energy ratio as defined in the following equation.

$$\text{energy ratio} = \frac{\sum_{n>N} \left(Q_{mn}\right)^2}{\sum \left(Q_{ij}\right)^2} \tag{24}$$

where N represents the column number of the first all-zero column, Q_{ij} are the quantized coefficients of one 32×32, and Q_{mn} are the quantized coefficients that are the right side of the first all-zero column. The energy ratios of 18 test sequences are shown in Fig. 15. We can see that the results are very

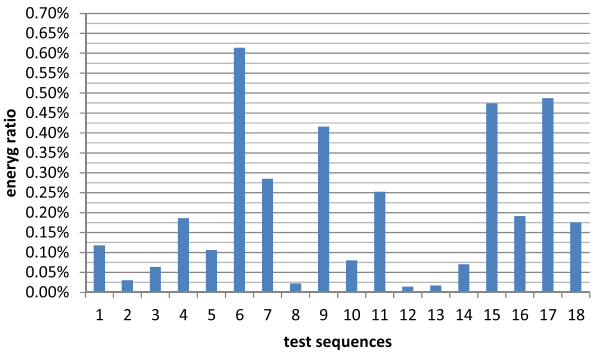

Fig. 15. The ratio of the energy of the first all-zero column and its rightward columns to the whole energy of the TU.

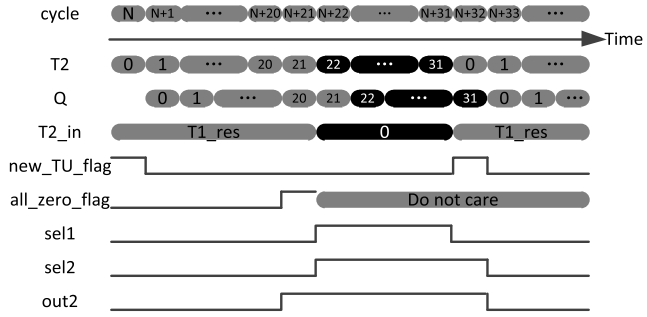

Fig. 16. Column transform (T2) and quantization (Q) are processed in a pipeline manner. The quantized results of the 20-th column are all-zero. Therefore, the inputs for T2 are set to 0 for the 22-th column and its following columns.

small, which means that the energy ratio after the first all-zero column is quite small. Therefore, we can skip the processing for the columns after the first all-zero column.

In the real implementation, the transform and quantization are usually executed in a pipeline manner, as shown in Fig. 16. Take transform unit (TU) 32×32 as an example, if the pixel parallelism is 32, for column 0, the column transform (T2) is processed in N-th cycle, and quantization (Q) is processed in the next clock cycle. The quantized results of the 20-th column are all-zero in (N+21)-th cycle, therefore, from (N+22)-th cycle, the inputs for T2 become zero for the 22-th column and its following columns of the same TU.

In order to achieve the pipeline timetable in Fig. 16, the circuit is shown in Fig. 17. A detector is used to check whether the quantized results are all-zero or not. The multiplexer 1 (MUX1) is used to decide the inputs for T2, while the multiplexer 2 (MUX2) is used to decide the selection signal for MUX1. When a new TU is processed, the selection signal (sel1) for the MUX1 is de-asserted, thus the results from transpose memory are used as inputs for T2. When there is an all-zero column after the quantization, the output of MUX2 becomes one. So the selection signal for the MUX1 becomes one in the next clock cycle. For the selection signal of MUX2, when the output of MUX2 is one, the output will keep this high state until the end of the current TU.

By using ZCT, the switching activities could be significantly reduced to save the power consumption. Note that this scheme is only applicable to T2 since the process of the row transform (T1) has already been finished before T2 starts.

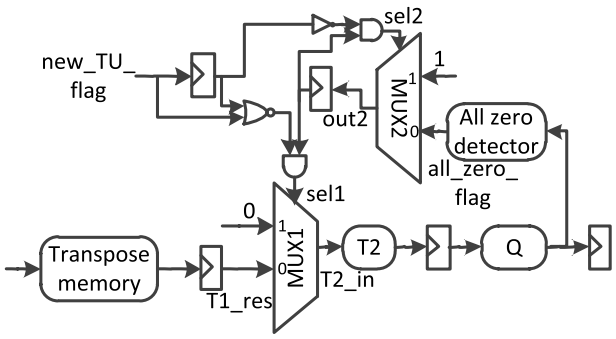

Fig. 17. The architecture when using ZCT. MUX1 is used to decide the inputs for T2, while MUX2 is used to decide the MUX1's selection signal.

TABLE VI

THE PROBABILITY THAT THE TRUNCATION WILL INFLUENCE THE FINAL RESULT FOR THE EO PART

N	8	7	6	5
Prob. (%)	43.75	21.88	10.94	5.47

TABLE VII

THE NUMBER OF TRUNCATED MSBS FOR THE FINAL OUTPUTS OF THE EO PART

Index of outputs (n)	0	1	2	3-4	5-7
# of truncated MSBs	0	2	4	5	6

VI. EXTENSIBILITY FOR VARIOUS SIZES OF DCT

The above methods are presented by the instance of the Odd part of 32-point DCT, and the input bit-width is 16 which are used in T2. However, the approximated schemes are not limited to the input bit-width and transform size. We show the extensibility in the following.

For the EO part of 32-point DCT, the architecture is shown in Fig. 3. For LT, the probability that the truncation will influence the final results is given in Table VI. We select 6 for the balance of coding efficiency and complexity reduction. For MT, the number of truncated MSBs is shown in Table VII. The index 0 is corresponding to the lowest-frequency component which has large dynamic range. Therefore, MT is not employed for index 0 in order to keep the coding efficiency. For the EE part of 32-point DCT, it can be further decomposed into even and odd parts and our methods are also applicable to the iterative decompositions. However, the hardware cost of the EE part is quite small in the overall. Therefore, we did not do the approximation for the EE part this time.

For T1, the input bit-width becomes 9. Correspondingly, the final right-shift bit number becomes 4 which is smaller than T2. Therefore, if LT is used, the probability that the truncation will influence the final result will achieve 50%. In order to keep the coding efficiency, LT is not adopted to T1. However, for the 4 LSBs, we still use approximate adders to calculate the summation. For MT, one complete row is processed and the largest magnitude for each position is tested as shown in Table VIII. From the results, we find that the number of possible truncated MSBs is quite limited compared with T2. Even for the highest-frequency component, only 3 MSBs are truncated. Therefore, the adoption of MT will not yield significant hardware cost reduction, while it bring

TABLE VIII
THE NUMBER OF TRUNCATED MSBS FOR THE FINAL OUTPUTS OF THE ODD PART IN ROW TRANSFORM

Index of outputs (n)	0	1-7	8-12	13-15
# of truncated MSBs	0	1	2	3

coding performance loss. Therefore, MT is not adopted for T1 in our proposal.

In the previous sections, we just perform the approximation for 32-point DCT. In order to show the applicability to other DCT computations in video coding, we integrate our LSB truncation method into more types and sizes of DCT. In the latest standard HEVC, there is only one DCT type (DCT-II) with the maximum size being 32×32. However, in the next generation standard versatile video coding (VVC) [28], the types of DCT are extended to DCT-II, DCT-V and DCT-VIII, with the maximum DCT size being extended to 128×128. Therefore, we apply the LSB truncation to VVC. The coding efficiency loss compared with the original is shown in Table XII in Section VII.A. We can see that the performance loss is smaller than 0.2% for the "all-intra" and "lowdelay" configuration, which is quite small. Therefore, we believe that our methods are applicable to other DCT computations.

VII. EXPERIMENTAL RESULTS

Our target is to reduce the hardware cost of DCT computation while keeping the coding efficiency. Since [18] and [20] have much better coding efficiency compared with the other references, so we just compare with [18] and [20] in terms of coding efficiency and hardware cost in this section.

A. Coding Efficiency Analysis

In this section, detailed comparisons are conducted to evaluate the coding efficiency of the approximation method. The proposed methods are integrated with HEVC Test Model (HM) 16.0 [30]. We have adopted the configuration "Intra, main" (AI), "Low delay, main" (LD) and "Random access, main" (RA) recommended in [29]. The performance loss is measured by BD-bitrate and BD-psnr which are calculated by Bjontegaard's method [31]. BD-bitrate represents the average bit rate difference in percent for the same PSNR, while BD-psnr means the average PSNR difference in dB for the same bit rate. Since our proposals are composed of LT, MT and ZCT, we analyze the performance of three proposals for AI, respectively, as shown in Table IX. When only LT is adopted, the BD-bitrate is about 0.06% on average. When LT and MT are adopted, the BD-bitrate becomes a little worse, decreases around 0.01%. If all the proposals are adopted, the average BD-bitrate is about 0.27%.

From the results, we can see that there is considerable variation in the results. We analyze three proposals respectively. For LT, according to our observation, there is a relationship between bit per pixel (bpp) and BD-bitrate. The bpp and BD-bitrate for each sequence is shown in Fig. 18. Blue and red curve represent bpp and BD-bitrate, respectively. We can see that Kimono (No. 3), BasketballDrive (No. 7) and Johnny (No. 17) have smaller bpp, while their BD-bitrates are much

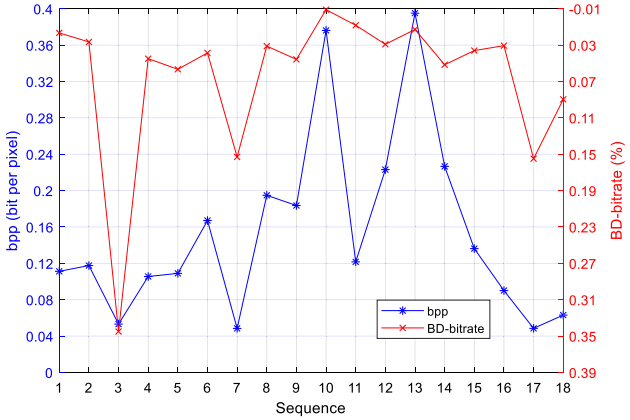

Fig. 18. Bit per pixel and BD-bitrate of LT for each sequence.

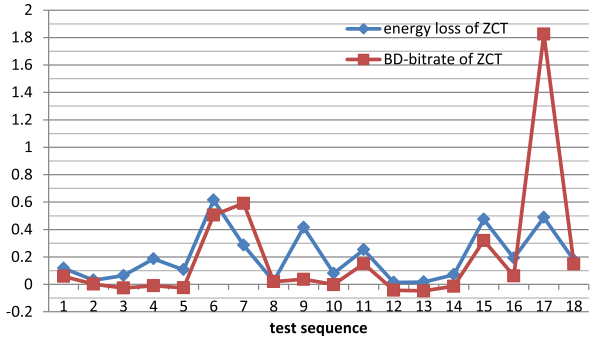

Fig. 19. The relationship between the energy loss and BD-bitrate aroused by ZCT.

larger than the others. In contrast, Partyscene (No. 10) and BQSquare (No. 13) have larger bpp, while their BD-bitrates are relatively small. It is because LT will influence the bit count, and those sequences with smaller bpp are more sensitive with the bit variation. As a result, LT will cause larger coding gain loss for the sequences with smaller bpp.

For MT, the variation of each sequence comes from different magnitude of high frequency components. Take BQMall as an example, BD-bitrate increase is 0.2014% which is much larger than the other sequences. It is because the texture of BQMall is complex, thus the magnitude of high frequency components is large. As a result, some MSBs of high frequency components are mistakenly truncated. If we truncated fewer MSBs for the high-frequency components, the coding gain will be improved. For example, if we truncate one fewer MSB for the 13 high-frequency components of Odd part and 6 high-frequency components of EO part, the BD-bitrate coming from MT is decreased from 0.2014% to 0.0751%.

For ZCT, we can see that for some sequences such as Johnny (No. 17), the BD-bitrate when performing ZCT is about 1.83%, which is larger than the other sequences. One of the reasons is that more energy is lost for Johnny when ZCT is adopted. By using Eq. (24), about 0.49% energy is lost for Johnny which is larger than most of the other sequences. The relationship between the energy loss and the BD-bitrate when performing the ZCT is shown in Fig. 19. We can see that there is a trend that the sequences with larger energy loss will suffer larger performance loss due to the ZCT, in general.

TABLE IX
THE CODING EFFICIENCY OF INDIVIDUAL PROPOSAL IN AI CASE

		LT		LT+MT		LT+MT+ZCT	
Class	Sequence	BD-bitrate (%)	BD-psnr (dB)	BD-bitrate (%)	BD-psnr (dB)	BD-bitrate (%)	BD-psnr (dB)
A	Traffic	0.0166	-9.09E-04	0.0257	-0.0014	0.0848	-0.0046
	PeopleOnStreet	0.0265	-0.0015	0.0313	-0.0018	0.0335	-0.002
B	Kimono	0.345	-0.0096	0.3124	-0.0087	0.2864	-0.0064
	ParkScene	0.0449	-0.0021	0.053	-0.0024	0.0432	-0.0019
	Cactus	0.0566	-0.0021	0.0547	-0.002	0.0297	-0.0015
	BQTerrace	0.0386	-0.0025	0.0354	-0.0021	0.5412	-0.0329
	BasketballDrive	0.153	-0.0045	0.1737	-0.005	0.7649	-0.0155
C	RaceHorsesC	0.0311	-0.002	0.0311	-0.002	0.0514	-0.0034
	BQMall	0.0457	-0.0029	0.2471	-0.0155	0.2842	-0.0177
	PartyScene	-0.009	8.06E-04	-0.009	8.04E-04	-0.0106	0.001
	BasketballDrill	0.0082	-4.34E-04	0.0082	-4.34E-04	0.1583	-0.0076
D	RaceHorses	0.0291	-0.0021	0.0291	-0.0021	-0.0131	5.18E-04
	BQSquare	0.0128	-0.001	0.0128	-0.001	-0.035	0.0042
	BlowingBubbles	0.0516	-0.003	0.0516	-0.003	0.0384	-0.0024
	BasketballPass	0.036	-0.0025	0.036	-0.0025	0.3564	-0.0214
E	FourPeople	0.0306	-0.0017	0.0306	-0.0017	0.0923	-0.0056
	Johnny	0.1549	-0.0063	0.1549	-0.0063	1.9802	-0.0808
	KristenAndSara	0.0897	-0.004	0.0897	-0.004	0.2385	-0.0128
Average		**0.0646**	**-0.0027**	**0.0760**	**-0.0034**	**0.2736**	**-0.0117**

For the final results, we can see that there are several sequences whose BD-bitrate is slightly better than the reference. It is because our proposals will truncate the transformed coefficients so that the number of encoded bits becomes smaller than the reference. Therefore, if the PSNR remains the same level or decreases at a lower rate compared to the number of encoded bits, the BD-bitrate will become a little better than the reference. Take RaceHorses as an example, for the result of QP37, the bitrate of proposal is 0.04% smaller than the origin, while the PSNR of the proposal is 0.024% lower than the origin. As a result, PSNR decreases slower than bitrate, so the BD-bitrate of the proposal is slightly better than the origin.

For the configuration LD and RA, the overall BD-bitrate is shown in Table X. The average BD-bitrate is about 0.21% and 0.05% for LD and RA respectively.

The comparison with the other works in terms of BD-bitrate is shown in Table XI. Compared with [18], the quality loss is slightly larger in the case of AI and LD, while a little bit smaller in the case of RA. Masera *et al.* [20] gave three modes for different coding efficiency. Among the three modes, the one with the best coding efficiency is shown in Table XI. Compared with [20], we can achieve smaller quality loss for all the configurations.

We apply the LSB truncation to Joint Exploration Test Model (JEM) 7.0 [32] which is the test model of VVC. The coding efficiency loss compared with the original is shown in Table XII. We can see that the performance loss is smaller than 0.2% for the "all-intra" and "lowdelay" configuration, which is quite small.

TABLE X
THE CODING EFFICIENCY IN LD AND RA CASES FOR LT+MT+ZCT

	LD		RA	
Sequence	BD-bitrate (%)	BD-psnr (dB)	BD-bitrate (%)	BD-psnr (dB)
Traffic			0.0615	-0.0017
PeopleOnStreet			-0.0092	2.00E-04
Kimono	0.8464	-0.0296	0.0967	-0.0027
ParkScene	0.0898	-0.0028	0.0313	-9.41E-04
Cactus	-0.0877	0.0021	0.0399	-0.0016
BQTerrace	0.3511	-0.0081	0.0078	2.48E-04
BasketballDrive	0.2636	-0.01	0.3387	-0.0056
RaceHorsesC	-0.1549	0.0065	-0.0561	0.0023
BQMall	0.2352	-0.01	0.155	-0.0062
PartyScene	-0.1462	0.0065	0.0361	-0.0016
BasketballDrill	0.5554	-0.0219	0.1759	-0.0074
RaceHorses	-0.0974	0.006	-0.2332	0.011
BQSquare	-0.243	0.0104	0.0732	-0.0034
BlowingBubbles	0.0064	-0.0013	-0.0053	5.78E-04
BasketballPass	0.4194	-0.0201	0.107	-0.0051
FourPeople	0.0711	-4.80E-04		
Johnny	1.048	-0.0179		
KristenAndSara	0.2348	-0.0077		
Average	**0.2120**	**-0.0061**	**0.0546**	**-0.0015**

TABLE XI
THE COMPARISON WITH OTHER WORKS IN TERMS OF QUALITY LOSS

Design	BD-bitrate		
	AI	LD	RA
[18]	0.1%	0.1%	0.1%
[20]	0.4%	0.3%	0.3%
Ours	**0.27%**	**0.21%**	**0.05%**

B. Hardware Cost Analysis

In this section, we analyze the cost of our proposed approximated hardware. We have used Verilog-HDL to implement the proposal and the design is synthesized with a TSMC 90nm cell library.

From the introduction in Section II, we can see that there are many multiplications with constant coefficients. It is noted that

TABLE XII
THE CODING EFFICIENCY IN AI AND LD CASES FOR VVC

Sequence	AI		LD	
	BD-bitrate (%)	BD-psnr (dB)	BD-bitrate (%)	BD-psnr (dB)
Traffic	0.0382	-0.0021		
PeopleOnStreet	-0.0020	0.0001		
Kimono	1.0128	-0.0359	0.6955	-0.0249
ParkScene	0.1167	-0.0050	0.0987	-0.0037
Cactus	0.0436	-0.0012	0.3567	-0.0074
BQTerrace	0.1276	-0.0067	0.0551	-0.0008
BasketballDrive	0.2869	-0.0073	0.8265	-0.0202
RaceHorsesC	0.0357	-0.0027	-0.0842	0.0059
BQMall	0.2034	-0.0128	-0.1785	0.0099
PartyScene	-0.0616	0.0046	-0.0334	0.0029
BasketballDrill	-0.3724	0.0181	0.3627	-0.0160
RaceHorses	-0.0806	0.0073	-0.1732	0.0097
BQSquare	0.0793	-0.0062	0.1376	-0.0074
BlowingBubbles	0.0742	-0.0073	-0.3841	0.0154
BasketballPass	0.1836	-0.0103	0.5476	-0.0247
FourPeople	0.0833	-0.0050	-0.0293	-0.0014
Johnny	0.3743	-0.0130	0.0093	-0.0035
KristenAndSara	0.2719	-0.0126	0.3295	-0.0078
Average	**0.1342**	**-0.0054**	**0.1585**	**-0.0046**

in the RTL level, we implement these multiplications directly (* operator in Verilog), and the multiplier with constant coefficient will be optimized by the Design Complier tool.

Same as [18], the working frequency is set to 187MHz in our experiment. Note that the maximum frequency of our design is 303MHz. Similar to the analysis of the coding efficiency, the following four configurations are used: 1) original accurate computation, 2) approximation using LT, 3) approximation using LT and MT and 4) all the approximations (LT, MT, and ZCT).

For the area consumption, for T2, original non-approximated hardware uses 149K gates. Approximation with LT reduced the gate count to 111K. Using MT approximation further reduced the gate count to 102K. ZCT approximation only contributes to reducing power consumption and has no effect on size of the hardware. For T1, we only adopt the approximated adders in LT which has no contribution to the area saving compared to the original non-approximated hardware. About 78K gates are consumed.

For the power consumption, we have measured the results by simulating and annotating the switching activities of each node of the gate-level netlist. A low-resolution sequence *RaceHorses* and high-resolution sequence *Kimono* are selected for the evaluation. For each sequence, four QPs are tested in the configuration of AI, RA and LD. Overall, the results for *RaceHorses* and *Kimono* for T2 are shown in Fig. 20 and Fig. 21, respectively. When ZCT is not adopted, the results are almost the same for the two test sequences. Original non-approximated hardware consumes about 68.4mW. When LT is adopted, the power consumption is reduced to 42.67mW. When MT is also utilized, additional 5mW can be saved. When ZCT is adopted, the power consumption varies for different configurations. The reason is that more power can be saved with more zero columns. Therefore, for the same

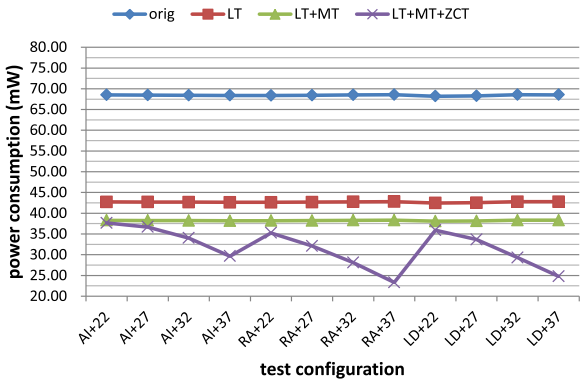

Fig. 20. The power consumption of *RaceHorses* under 12 different test configurations (e.g. AI+22 means "intra, main" and QP22).

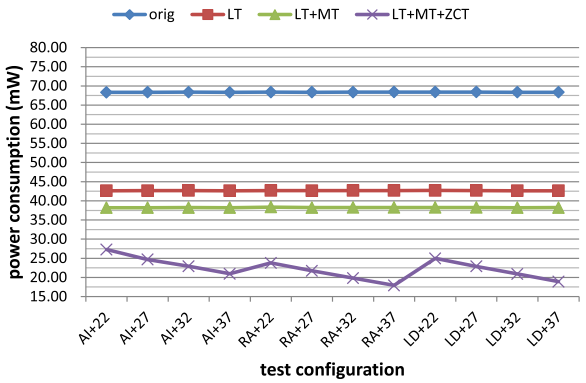

Fig. 21. The power consumption of *Kimono* under 12 different test configurations.

TABLE XIII
POWER AND AREA REDUCTION COMPARED WITH THE ORIGIN

	T1 (9-bit input)	T2 (16-bit input)
Original power consumption (mW)	28.52	68.40
Power consumption with proposals (mW)	22.34	26.96
Power reduction	**-21.67%**	**-60.59%**
Original gate counts (K)	77	149
Gate counts with proposals (K)	78	102
Area reduction	**-0%**	**-32%**

sequence, the power consumptions of RA/LD are smaller than that of AI, and the results of larger QP are smaller than that of smaller QP. For *Kimono*, there are more zero columns than *RaceHorses*. Therefore, the power consumptions of *Kimono* are smaller than *RaceHorses* in the same test configuration. As a result, the average power is 26.96mW. In addition, we have also measured the power consumption for T1 in the same configuration as T2. The average power consumption for the original non-approximated hardware is 28.52mW. When LT is employed, the average power is about 22.34mW. We summarize the results in Table XIII. For T1, 21.67% power consumption can be reduced. For T2, 60.59% power consumption and 32% area consumption can be reduced.

By observing the power results, we can see that QP does not affect the results of original DCT, LT and MT, while it affects the results of ZCT obviously. It is because quantization is executed after DCT, which means that the quantization

TABLE XIV

THE POWER REDUCTION AND BD-BITRATE FOR THREE TEST CONFIGURATIONS

Configuration	Power reduction (%)	BD-bitrate (%)
AI	-57.27	0.27
LD	-61.39	0.21
RA	-63.10	0.05

TABLE XV

THE HARDWARE COST COMPARISON FOR 32-POINT 1-D APPROXIMATED DCT WITH 16-bit INPUT

DESIGN	Technology	Frequency (MHz)	Gate (K)	Normalized area	Power* (mW)
[18]	90nm	187	103	0.551	17.63
[20]	90nm	250	167	0.668	13.50
OURS	**90nm**	**187**	**102**	**0.545**	**12.33**

* The result is obtained from the synthesis reports at 100MHz.

step does not directly influence the DCT process in a default manner. Therefore, for the original DCT, LT and MT, there is no substantial difference for the power results of different QPs.

However, for ZCT, the powers for different QPs are quite different, and the power for larger QP is smaller. It is because the quantization step directly influences the DCT process when using ZCT. As described in Section V, when detecting an all-zero column of quantized results, the DCT processes for all the columns in the right side are skipped. For larger QP, the all-zero column appears earlier, thus more columns of DCT process can be skipped. As a result, the power for larger QP becomes smaller.

Table XIV shows the compression performance along with power reduction for each test configuration respectively. For AI, the power reduction is 57.27% while the BD-bitrate is 0.27%. For LD, the power reduction is 61.39% while the BD-bitrate is 0.21%. For RA, the power reduction is 63.10% while the BD-bitrate is 0.05%. We can see that the variation of the power reduction for three test configurations is not so large, which comes from ZCT. Therefore, the averaged power reduction is used to compare with the other works in the following. It is noted that we use the same circuit for all the configurations, so there is no difference for the area reduction (32%) of three test configurations.

The comparison of the area and power consumption for the 32-point 1D approximated DCT is shown in Table XV. About gate count, our design costs 102K, while [18] and [20] consume 103K and 167K, respectively. Since the operating frequency in [20] is different, a normalized gate count is introduced as shown in Eq. (25).

$$\text{Normalized gate count} = \frac{gate\ count}{operating\ frequency} \quad (25)$$

About the power consumption, [18] gives the result at the 100MHz which consumes 17.63mW. For a fair comparison, we also synthesize our design and the work [20] at 100MHz, the power consumption from the synthesis report is shown in Table XV. We can see that our normalized gate count is 0.545, which is smaller than the other works. Besides, we achieve a lower power consumption of 12.33mW than other works.

TABLE XVI

THE COMPARISON WITH OTHER WORKS

Design	Power reduction (%)	Area reduction (%)	BD-bitrate (%)		
			AI	LD	RA
[18]	-23.91	-21	0.1	0.1	0.1
[20]	-31.6	N/A	0.4	0.3	0.3
PROPOSAL	**-60.59**	**-32**	**0.27**	**0.21**	**0.05**

The area and power reduction compared with the respective non-approximate version are shown in Table XVI. The non-approximate versions are different for ours and other works from two aspects. One difference is that [18] gives a reconfigurable architecture for the DCT size from 4×4 to 32×32, while we focus on the architecture for the largest DCT size 32×32. These two kinds of architectures are used in different parts of video coding. The former one is used for the reconstruction loop while the latter one is used for the mode decision, as shown in some video encoding implementations [33], [34]. The other difference is the synthesis condition. There are three kinds of libraries, i.e., slow, typical and fast, which will affect the synthesis results largely. We use the slow library while the other references did not explicitly mention the library for the synthesis. Therefore, in order to give a more fair comparison, we compare the area and power reduction starting from the respective non-approximate architectures, as given in Table XVI.

About the area reduction for the 32-point 1D DCT with 16-bit input, for our design, the gate counts for non-approximated version is 149K, while the gate count of the approximated version is 102K. Therefore, 32% area reduction can be achieved. For [18], the gate count for the non-approximated and approximated edition is 131K and 103K. Therefore, 21% area reduction can be reduced. For [20], the same circuits are used for non-approximated and approximated version, so there is no area reduction. Therefore, our approximated method can save more area consumption compared with the other works.

About the power reduction, 60.59% power can be reduced in our design as shown in Table XIII. For [18], for the non-approximated version, the 32-point DCT consumes 23.17mW, and it is reduced to 17.63mW for the approximated version. Around 23.91% power consumption is saved. Masera *et al.* [20] gave three modes which can support different power savings. Among the three modes, Mode 1 is the one which achieves a comparable coding efficiency compared with our design. Therefore, we compare with the results of mode 1 in [20]. Mode 1 (the approximated version) can save 31.60% power consumption compared to the mode 0 (non-approximated version). Therefore, our approximated method can save more power consumption compared with the other works.

VIII. CONCLUSION

In this paper, a low-cost approximated DCT is presented. At first, the arithmetic operations for 6 LSBs are completely removed. After that, a parallel circuit is presented to generate

the carry bit for the addition of the remaining LSBs. Secondly, unused MSBs are truncated to save the cost. More bits are truncated for higher-frequency components. Finally, the processing for selected columns with zero value is skipped to decrease the switching activities in order to save the power consumption. Overall, at most 32% area and 60% power consumption are reduced compared to the original accurate one, at the cost of 0.27%, 0.21% and 0.05% coding efficiency losses for AI, LD and RA, respectively. About the future work, we will try truncating regions based on zig-zag order rather than column order which suppress high and low-frequency components.

In Section VII.A, we have analyzed that video content influences the coding efficiency. Take ZCT as an example, there are some sequences such as RaceHorses whose BD-bitrate becomes better, while there are also some sequences such as Johnny whose BD-bitrate becomes worse. It is because using ZCT will reduce the number of encoded bits, while it will also lead to worse PSNR. Therefore, if the PSNR remains the same level or decreases at a lower rate than the number of encoded bits, the BD-bitrate will become better. Otherwise, the BD-bitrate will become worse. This difference comes from various video contents. To optimize the best BD-bitrate based on the video content dynamically, we plan to use deep learning technology. The input of deep learning network is video content and the output is the number of columns that should be truncated. By doing so, we can dynamically decide the truncation scheme to pursue a better coding efficiency.

REFERENCES

[1] *High Efficiency Video Coding*, document Rec. ITU-T H.265 and ISO/IEC 23008-2, Jan. 2013.

[2] M. Tikekar, C.-T. Huang, V. Sze, and A. Chandrakasan, "Energy and area-efficient hardware implementation of HEVC inverse transform and dequantization," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 2100–2104.

[3] G. J. 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.

[4] G. Pastuszak and A. Abramowski, "Algorithm and architecture design of the H.265/HEVC intra encoder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 210–222, Jan. 2016.

[5] H. Qi, Q. Huang, and W. Gao, "A low-cost very large scale integration architecture for multistandard inverse transform," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 7, pp. 551–555, Jul. 2010.

[6] G. A. Su and C.-P. Fan, "Low-cost hardware-sharing architecture of fast 1-D inverse transforms for H.264/AVC and AVS applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 12, pp. 1249–1253, Dec. 2008.

[7] G.-A. Su and C.-P. Fan, "Low-cost hardware-sharing architecture of fast 1-D inverse transforms for H.264/AVC and AVS applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 12, pp. 1249–1253, Dec. 2008.

[8] K. Wang *et al.*, "A reconfigurable multi-transform VLSI architecture supporting video codec design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 58, no. 7, pp. 432–436, Jul. 2011.

[9] H. Chang and K. Cho, "High-performance inverse transform circuit based on butterfly architecture for H.264 high profile decoder," in *Proc. IEEE Asia Pacific Conf. Circuits Syst.*, Dec. 2010, pp. 394–397.

[10] C. Peng, D. Yu, X. Cao, and S. Sheng, "A new high throughput VLSI architecture for H.264 transform and quantization," in *Proc. Int. Conf. ASIC*, Oct. 2007, pp. 950–953.

[11] H. Sun, D. Zhou, J. Zhu, S. Kimura, and S. Goto, "An area-efficient 4/8/16/32-point inverse DCT architecture for UHDTV HEVC decoder," in *Proc. IEEE Vis. Commun. Image Process. Conf.*, Dec. 2014, pp. 197–200.

[12] J. Zhu, Z. Liu, and D. Wang, "Fully pipelined DCT/IDCT/Hadamard unified transform architecture for HEVC Codec," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2013, pp. 677–680.

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

[14] M. Budagavi and V. Sze, "Unified forward+inverse transform architecture for HEVC," in *Proc. IEEE Int. Conf. Image Process.*, Sep./Oct. 2012, pp. 209–212.

[15] W. Zhao, T. Onoye, and T. Song, "High-performance multiplierless transform architecture for HEVC," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2013, pp. 1668–1671.

[16] S. Y. Park and P. K. Meher, "Flexible integer DCT architectures for HEVC," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2013, pp. 1376–1379.

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

[18] P. K. Meher, S. Y. Park, B. K. 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.

[19] U. S. Potluri, A. Madanayake, R. J. Cintra, F. M. 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, Reg. Papers*, vol. 61, no. 6, pp. 1727–1740, Jun. 2014.

[20] M. Masera, M. Martina, and G. Masera, "Adaptive approximated DCT architectures for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 12, pp. 2714–2725, Dec. 2017.

[21] M. Jridi, A. Alfalou, and P. K. Meher, "A generalized algorithm and reconfigurable architecture for efficient and scalable orthogonal approximation of DCT," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 2, pp. 449–457, Feb. 2015.

[22] M. Jridi and P. K. Meher, "Scalable approximate DCT architectures for efficient HEVC-compliant video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 8, pp. 1815–1825, Aug. 2017.

[23] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "Binary discrete cosine and Hartley transforms," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 4, pp. 989–1002, Apr. 2013.

[24] H. Sun, D. Zhou, S. Zhang, and S. Kimura, "A low-power VLSI architecture for HEVC de-quantization and inverse transform," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E99–A, no. 12, pp. 2375–2387, Dec. 2016.

[25] C.-C. Ju *et al.*, "A 0.5 nJ/pixel 4 K H.265/HEVC codec LSI for multi-format smartphone applications," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 56–67, Jan. 2016.

[26] F. de Dinechin, M. Istoan, and A. Massouri, "Sum-of-product architectures computing just right," in *Proc. IEEE Int. Conf. Appl.-Specific Syst., Architectures Processors (ASAP)*, Jun. 2014, pp. 41–47.

[27] E. Y. Lam and J. W. Goodman, "A mathematical analysis of the DCT coefficient distributions for images," *IEEE Trans. Image Process.*, vol. 9, no. 10, pp. 1661–1666, Oct. 2000.

[28] B. Bross, J. Chen, S. Liu, *Versatile Video Coding (Draft 2)*, document JVET-K1001, Oct. 2018.

[29] F. Bossen, *Common HM Test Conditions and Software Reference Configurations*, document JCTVC-L1100, Apr. 2013.

[30] *HEVC Test Model(HM-16.0)*. Accessed: Nov. 2018. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.0/

[31] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD-curves*, document VCEG-M33, Apr. 2001.

[32] *Joint Exploration Model (JEM-7.0)*. Accessed: Nov. 2018. [Online]. Available: https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/branches/HM-16.6-JEM-7.0-dev/

[33] S.-F. Tsai, C.-T. Li, H.-H. Chen, P.-K. Tsung, K.-Y. Chen, and L.-G. Chen, "A 1062Mpixels/s 8192Ã—4320p high efficiency video coding (H.265) encoder chip," in *Proc. Symp. VLSI Circuits*, Jun. 2013, pp. C188–C189.

[34] J. Zhu, Z. Liu, D. Wang, Q. Han, and Y. Song, "HDTV1080p HEVC intra encoder with source texture based CU/PU mode pre-decision," in *Proc. IEEE Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2014, pp. 367–372.

[35] H. Sun, Z. Cheng, A. M. Gharehbaghi, S. Kimura, and M. Fujita, "A low-cost approximate 32-point transform architecture," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

Authors' photographs and biographies not available at the time of publication.