Towards Low Power Approximate DCT Architecture for HEVC Standard

Zdenek Vasicek, Vojtech Mrazek and Lukas Sekanina
Brno University of Technology, Faculty of Information Technology, IT4Innovations Centre of Excellence
Brno, Czech Republic
Email: {vasicek, imrazek, sekanina}@fit.vutbr.cz

***Abstract*—Video processing performed directly on IoT nodes is one of the most performance as well as energy demanding applications for current IoT technology. In order to support real-time high-definition video, energy-reduction optimizations have to be introduced at all levels of the video processing chain. This paper deals with an efficient implementation of Discrete Cosine Transform (DCT) blocks employed in video compression based on the High Efficiency Video Coding (HEVC) standard. The proposed multiplierless 4-input DCT implementations contain approximate adders and subtractors that were obtained using genetic programming. In order to manage the complexity of evolutionary approximation and provide formal guarantees in terms of errors of key circuit components, the worst and average errors were determined exactly by means of Binary decision diagrams. Under conditions of our experiments, approximate 4-input DCTs show better quality/power trade-offs than relevant implementations available in the literature. For example, 25% power reduction for the same error was obtained in comparison with a recent highly optimized implementation.**

I. Introduction

Small embedded systems connected to Internet of Things (IoT) are expected to be a new infrastructure of the information society. These systems range from simple smart sensors to advanced embedded applications requiring considerable computing resources. Video processing performed directly on IoT nodes is one of the most performance as well as energy demanding applications. In order to support real-time coding and decoding of high definition video on low cost devices, energy consumption optimization has to be introduced at all levels of the video processing system.

One of the most frequently performed and thus energy demanding operations is the Discrete Cosine Transform (DCT) block. In commonly optimized DCT blocks, bit width of all operations is reduced as much as possible, multipliers are replaced with additions, subtractions and shifts, and less important logic is pruned in order to reduce power consumption [1]. As video processing is, in principle, an error resilient application, DCT can further be approximated. Various approaches to the DCT approximation will be reported in Section II.

This paper deals with a deep optimization and approximation of the DCT block employed in the state of the art High Efficiency Video Coding (HEVC) standard [2]. In addition to the common optimization techniques, we propose to approximate adders and subtractors of a multiplierless DCT in such a way that the resulting error is kept under a predefined threshold. As the error of addition/subtraction is computed formally, without applying circuit simulation, it is always guaranteed that the target error is never exceeded. The error calculation procedure is based on relaxed equivalence checking using Binary Decision Diagrams (BDDs). BDDs are also used to analyze power consumption during the approximation process.

The approximate adders and subtractors are generated using a genetic programming-based approximation method. The so-called functional gate-level approximation is, in fact, performed in which accurate logic circuits are gradually simplified while an acceptable error is tolerated. Various approximate implementations of DCT showing good trade-offs between the quality of video processing and power consumption were obtained. These implementations were employed in reference HEVC implementation in order to compare them with available relevant implementations and determine their impact on the quality of video processing on selected benchmark video sequences. For example, 25% power reduction for the same error was obtained in comparison with a recent highly optimized implementation [1].

The rest of the paper is organized as follows. Section II briefly surveys relevant approximate circuit design approaches. Section III is devoted to the principles of DCT and its efficient implementation. The proposed approximation method for adders and subtractors is presented in Section IV. After presenting the experimental setup in Section V, results are reported and discussed in Section VI. Conclusions are given in Section VII.

II. Related Work

Efficient implementations of signal processing blocks in general and DCT in particular have been developed for decades (see the introduction to DCT in Section III). This effort has recently led to establishing a new High Efficiency Video Coding standard. Profiling results of HEVC given in [3] indicate that there is a good potential for improving energy and implementation efficiency especially of forward as well as inverse DCT blocks. These improvements can, in fact, be considered as approximations introduced at various levels of the algorithm and its implementation. They primarily include employing the integer data representation, bit width reduction, multiplierless multiplication and coefficient approximation [3], [1].

978-3-9815370-8-6/17/$31.00

A. DCT as a Test Problem for Approximate Computing

However, with the development of the approximate computing paradigm, more radical approximations have been proposed. DCT blocks of JPEG and MPEG often serve as one of several test circuits used to evaluate the quality of general purpose approximation methods which usually perform the optimization and approximation at the gate level; see, for example, SALSA [4], ASLAN [5] and the logic isolation based approximation method [6]. In paper [7], several simplified transistor-level implementations of a 1 bit full adder were introduced and used in approximate adders that are employed in DCT of the JPEG implementation. Raha et al. proposed a power efficient video encoder in which all the adders and subtractors of the motion estimation and DCT blocks were replaced by their approximate configurable versions. The method enabled to automatically adjust a degree of hardware approximation dynamically based on the video characteristics [8]. A very specific optimization approach exploiting the sensitivity error analysis and error variance propagation in the DCT graph was developed in [9]. With respect to the available budget, the optimal number of 1 bit adders that should be approximated in all adders and multipliers was computed using the mixed integer nonlinear problem solver. The determined 1 bit adders were then replaced by their approximate versions taken from [7].

With respect to the approach developed in this paper, the aforementioned methods show two main drawbacks. Firstly, they do not deal with DCT intended for HEVC. Secondly, the papers show that DCT can be approximated, but the results are not compared against other DCT approximation methods.

B. Approximate Adders

As the proposed approach is based on approximate adders, this subsection briefly surveys this subarea of approximate computing. Adders are approximated by either general-purpose approximation methods or problem-specific methods. In the former case, the adders serve as one of many circuit classes that can be approximated by methods such as [4], [6] or multiobjective genetic programming-based methods [10]. Problem-specific methods exploit the structure of conventional adders. Another class of circuits are quality configurable adders (e.g. GeAR adders) which allow for a dynamic control of the error-power trade-off [11].

Four types of approximate adders are considered and evaluated in [12]: (1) Speculative adders in which the carry is speculated for each sum bit using only one or several bits. (2) Segmented adders, where an n-bit adder is divided into k-bit sub-adders and the carry is then generated by using different methods. (3) Carry-select adders in which multiple sub-modules are used to compute the sum for different carry values, and the result is determined according to the carry of a sub-module. (4) Approximate 1 bit full adders where the full adder is approximated at the transistor level and used as a building block of more complex adders [7].

C. Error Calculation

Almost all circuit approximation methods compute the error by circuit simulation. An open question is how many test vectors have to be applied in order to obtain trustworthy results. Only a few papers deal with formal analysis of candidate approximate circuits to establish the error and provide formal guarantees on the error bound. Checking the worst error can be based on satisfiability (SAT) solving as demonstrated in [13]. Binary decision diagrams (BDDs) were used to obtain the average arithmetic error, worst error, error rate [14] and average Hamming distance [15]. However, the formal methods are currently inapplicable for determining some types of errors for certain classes of circuits, e.g. the average error for non-trivial multipliers.

III. DCT and its approximation

A. Principle of DCT

For a given input vector $\mathbf{X} = [\mathbf{x_0}, \mathbf{x_1}, \ldots, \mathbf{x_{N-1}}]^\mathbf{T}$, its corresponding DCT output $\mathbf{Y} = [\mathbf{y_0}, \mathbf{y_1}, \ldots, \mathbf{y_{N-1}}]^\mathbf{T}$ is given by $\mathbf{Y} = \mathbf{C_N X}$, where $\mathbf{C_N}$ denotes the N-point DCT kernel. In general, $\mathbf{C_N}$ consists of real numbers. The elements $c_{ij} \in \mathbf{C_N}$ are defined as $c_{ij} = \frac{A}{\sqrt{N}} \cos[\frac{\pi}{N}(j + \frac{1}{2})i]$, where $i, j = 0, \ldots, N-1$. A is equal to 1 and $\sqrt{2}$ for $i = 0$ and $i > 0$ respectively.

HEVC standard utilizes 4-point, 8-point, 16-point and 32-point DCT in its forward (video coder) as well as inverse (video decoder) form. In order to simplify the complexity of hardware circuits computing the output of DCT, HEVC scales the coefficients by a power of two and rounds them to the integer value. The scaling and rounding was optimized to preserve the main advantages and properties of DCT such as the orthogonality of basis vectors, equal norm of basis vectors and good compression efficiency on the one hand as well as the low complexity of hardware circuits on the other hand [16].

Let us restrict ourself to a 4-point 1D forward transform whose output is defined as follows:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \mathbf{C_4} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad (1)$$

The output of DCT could be determined by means of straightforward matrix multiplication. This approach, however, requires to perform N^2 multiplications and $N(N-1)$ additions. The costly matrix multiplication could be avoided by utilizing the symmetry properties of basis vector. A decomposition technique known as Even-Odd decomposition [16] could be employed to reduce the computational complexity of DCT. The 4-point DCT given by Eq. 1 can then be rewritten to

$$\begin{bmatrix} y_0 \\ y_2 \end{bmatrix} = \mathbf{C_2^o} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 64 & 64 \\ 64 & -64 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$
$$\begin{bmatrix} y_1 \\ y_3 \end{bmatrix} = \mathbf{C_2^e} \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} 83 & 36 \\ 36 & -83 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \quad (2)$$

where $a_0 = x_0 + x_3$, $a_1 = x_1 + x_2$, $b_0 = x_0 - x_3$, $b_1 = x_1 - x_2$. This gives us a hardware architecture shown in Fig. 1. In the first part (I.), the partial sums a_i and b_i are calculated using two w-bit adders and two w-bit subtractors, both producing a $(w+1)$-bit signed integer value. In the second part (II.), multiplication is performed. Finally, the multiplied values are summed up by two adders and two subtractors (part III.).

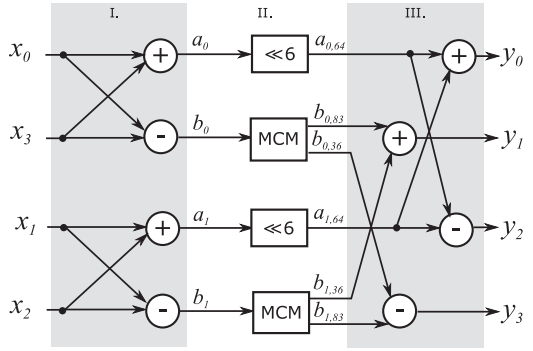

Fig. 1: Architecture of 4-point integer DCT.

In order to reduce the complexity of adder (subtractor) computing the output y_0 (y_2), the multiplication by factor 64 which is performed by means of a simple logic shift may be postponed and executed after summing (subtracting) the partial sums a_0 and a_1. This simple modification helps to reduce the bitwidth of adder (subtractor) by six. Both components operate with two $(w+1)$-bit integers.

As the coefficients utilized in Eq. 2 are constant, we can avoid the usage of costly multipliers and employ a much efficient approach – multiplierless multiple constant multiplier (MCM) which determines the value of $36 \cdot b_i$ and $83 \cdot b_i$ efficiently. MCM is a hardware block that exclusively consists of additions, subtractions, and shifts.

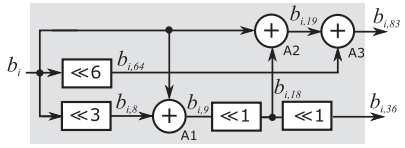

Fig. 2: Multiplierless multiple constant multiplier (83, 36)

The most compact implementation of MCM for 83 and 36 is shown in Fig. 2. The MCM has a single input denoted b_i and two outputs $b_{i,83}$ and $b_{i,36}$. According to Fig. 1, b_i is an $(w+1)$-bit signed integer. The MCM consists of three adders arranged in such a way that they have a minimal possible bitwidth. One $(w+4)$-bit signed adder (A1) which adds $(w+1)$-bit and $(w+4)$-bit signed values is shared between the subcircuit outputting $83 \cdot b_i$ and the subcircuit outputting $36 \cdot b_i$. The remaining two adders (i.e. $(w+6)$-bit A2 and $(w+7)$-bit A3) are employed to determine $83 \cdot b_i$.

B. The Proposed Approximation of DCT

DCT can be approximated by reducing the number of utilized adders (subtractors) and/or employing approximate adders (subtractors). In this paper, we investigate the impact of both approaches.

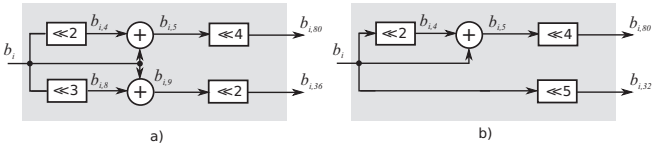

Fig. 3: Two variants of approximate MCM (83, 36)

The number of components can be reduced, for example, by approximating the coefficients of the matrix $\mathbf{C_2^e}$ shown in Eq. 2. It means, in fact, that we replace the MCM blocks employed in the architecture shown in Fig. 1 with different ones. The selection of the proper coefficients is nontrivial as it is necessary to consider not only the distance between the original and approximate coefficients, but also the number of components and the bit-width of the components. In this case, we exhaustively enumerated all possible MCM implementations with two coefficients that are within the range of +-16. We obtained three architectures representing the best trade-off among the considered parameters. Two of them are shown in Fig. 3. In the first case (Figure 3a), coefficient 83 is replaced with coefficient 80. This small difference helps to reduce the number of adders by one. In addition to that, the bit-width of the adder and subtractor combining the outputs of MCM block in DCT can be reduced by 2 because the shift by 2 can be postponed. When the coefficient 32 is used instead of the coefficient 36, we can remove another adder (see Figure 3b). This helps to reduce the bit-width by 4.

The third architecture, as proposed in [1], utilizes MCM with coefficient 64 and 32. i.e. the multiplication is replaced by the shifts. The bit-width is reduced by 5.

IV. Design of approximate adders and subtractors

Several approaches have been proposed to approximate arithmetic circuits. In this work, we employed Cartesian Genetic Programming (CGP) [17] because it can easily handle constraints given on candidate circuits, the method is naturally multi-objective and high-quality approximate circuits have already been obtained with it [18].

The standard CGP is a branch of genetic programming which represents candidate designs using directed acyclic graphs. A candidate circuit is modeled using an array of programmable nodes. In our case, 2-input Boolean functions are considered. The circuit utilizes n_i primary inputs and n_o primary outputs. Feedback connections are not enabled.

The primary inputs and the outputs of the nodes are labeled $0, 1 \ldots N$ and considered as addresses which the node inputs can be connected to. A candidate solution is represented in the so-called chromosome (which is, in fact, a netlist) by $N - n_i$ triplets (x_1, x_2, ψ) determining for each node its function ψ ($\psi \in \Gamma$ where Γ is the set of available functions) and input connections. The last part of the chromosome contains n_o integers specifying the nodes where the primary outputs are connected to.

CGP employs a simple search method in which the initial population P contains at least one implementation of the accurate circuit (adder or subtractor) and a few circuits generated using mutation of the accurate circuit. The next step consists of the evaluation of candidate circuits using the fitness function. Each member of P then receives the so-called fitness score and the highest-scored individual becomes a new parent of the next population. From this parent, λ candidate solutions are generated using mutation. The termination criterion is given by the maximum number of iterations.

A. Error metrics

In order to design signed approximate adders and subtractors exhibiting suitable parameters, the worst-case arithmetic error and average-case arithmetic error need to be kept under some level. In our approach, we employed the worst-case arithmetic error as a design constrain and average-case arithmetic error as a design objective.

The error metrics are determined using BDDs as follows. For each candidate solution, we create a virtual circuit that is subsequently represented using BDDs. The virtual circuit (shown in Fig. 4) takes a $2n$-bit input vector x and produces absolute difference between the $(n+1)$-bit output of accurate adder (subtractor) $f(x)$ and approximate adder (subtractor) $f'(x)$. Subtraction in virtual circuit is calculated using $m = n+2$ full-adders with first carry-in set to 1 and inverting each bit of the subtrahend. The absolute value is computed using $m-1$ half-adders and $m-1$ XOR gates.

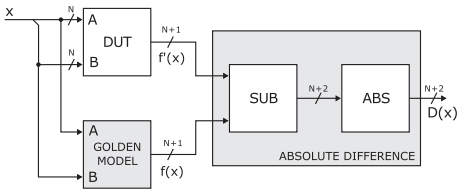

Fig. 4: Virtual circuit used for arithmetic error analysis

The difference $D(x)$ is represented by m-bit binary vector which corresponds with a natural number. It holds that $D(x) = \sum_{0 \leq i < m} d_i(x) \cdot 2^i$. Then, Algorithm 1 is used for the worst case error analysis.

Algorithm 1: worst-case error analysis

Input: BDD representation of the virtual circuit (d)
Output: The maximum arithmetic error (ε_{max})

```
1 ε_max ← 0, μ ← true;
2 for i ∈ {m − 1, m − 2, . . . , 0} do
3     if satisfiable(μ ∧ d_i) then
4         μ ← μ ∧ d_i; ε_max ← ε_max + 2^i;
5 return ε_max;
```

The average-case arithmetic error can be obtained by m calls of SATcount operation (one per each bit of D) which determines the number of input assignments that evaluates d_i to one.

Algorithm 2: average-case error analysis

Input: BDD representation of the virtual circuit (d)
Output: The average arithmetic error (ε_{avg})

```
1 ε_avg ← 0;
2 for i ∈ {m − 1, m − 2, . . . , 0} do
3     ε_avg ← ε_avg + 2^{i−2n} · satcount(d_i);
4 return ε_avg;
```

B. Fitness function

In this paper, the following fitness function is utilized:

$$fitness(A) = -\begin{cases} Pwr(A) & \text{if } \varepsilon_{max}(A) \leq E \\ \infty & \text{otherwise} \end{cases},$$

where A represents a candidate circuit, E the maximum acceptable error level and $Pwr(A)$ is the power consumption of A. In order to estimate the power consumption of a candidate circuit, we adopted a method based on the switching activity estimation introduced in [10].

For each candidate circuit, a virtual circuit is constructed and represented using BDDs. The construction of BDD starts with a candidate circuit denoted as DUT in Fig. 4. At the end of this step, BDD for each output bit of $f'(x)$ is available. At this point, transition probabilities are determined for each gate of DUT. These probabilities are employed to determine the switching activity and subsequently the power consumption. Then, the rest of the virtual circuit is created and worst-case error ε_{max} is analyzed using the Algorithm 1.

V. Experimental setup

We will evaluate the impact of the 4-point DCT (with bit-width-optimized adders) approximations on the quality of video processing and power consumption. The accurate implementation of 4-point DCT (denoted as A1) requires five different adders and two different subtractors (see the discussion in Section III-A). In addition to A1, three approximate architectures are considered: architecture A2 which utilizes MCM from Fig. 3a, A3 employing MCM from Fig. 3b and finaly A4 which replaces MCM with logic shifts. As the MCMs presented in Fig. 3 contain shifts that can be moved after the adders and subtractors situated in the third part of DCT architecture, six additional adders and three subtractors needs to be designed. In total, we need to implement 16 different circuits (11 adders and 5 subtractors) to construct four considered DCT architectures.

The goal of CGP is to design various approximate implementations of each circuit. We considered seven error levels defining the maximum acceptable worst-case error, in particular $E \in \{0.1\%, 0.2\%, 0.5\%, 1\%, 2\%, 5\%, 10\%\}$. Note that 100% corresponds with $\varepsilon_{max} = 2^{w+1}$, where w is the circuit's bit-width. For each E, 20 independent CGP runs were executed with parameters: $\lambda = 5$, $\Gamma = \{BUF, AND, OR, XOR, NAND, NOR, XNOR\}$. CGP is

TABLE I: Average time needed to perform error analysis for w-bit adders/subtractors

	$w = 4$	$w = 8$	$w = 12$	$w = 16$
simulation	4.5 μs	1.9 ms	682.4 ms	140.90 s
BDD ε_{max}	10.3 μs	3.5 ms	127.9 ms	1.38 s
Speedup	0.43×	0.54×	5.33×	102.30×
BDD ε_{avg}	14.0 μs	4.6 ms	312.7 ms	2.93 s
Speedup	0.32×	0.42×	2.18×	48.09×

terminated when 10^6 generations are exhausted or when its duration exceeded 120 minutes.

We obtained 140 different solutions for each circuit and 2240 solutions in total. For each E and each circuit, we identified a single solution exhibiting the best trade-off between the power consumption and average-case error (the exact average-case error was calculated using Algorithm 2). Then, the chosen circuits were utilized to create 7 approximate variants for each of four considered DCT architectures.

We obtained 32 different implementations of DCT whose parameters were subsequently evaluated. Firstly, we implemented these architectures in Verilog, synthesized in ABC and measured their power consumption. Secondly, we implemented them in C language and employed in reference implementation of HEVC[1] where we replaced the code performing the 4-point DCT with our implementations of DCT.

VI. Results

The results of performance analysis of the BDD-based method calculating the error of approximate adders (subtractors) are summarized in Table I. We measured the average time needed to perform the whole error analysis when employed in evolutionary loop. We run CGP for 10 minutes to obtain statistically significant results.

The proposed BDD-based technique is compared with an optimized approach based on exhaustive simulation. As evident, the BDD-based method performs better for more complex circuits (adders having at least 12-bits, i.e. 24 inputs). This result was expected as the simulation algorithm is able to evaluate circuit response for up to 256 different input vectors in one pass using modern CPUs. For example, for a 16-bit adder, the achieved speedup is greater than 100 in the case of the worst-case error analysis. The average-case error analysis is approximately two times slower compared to the worst-case error analysis. Despite this, the speedup greater than 48 was achieved for 16-bit adder when compared with the exhaustive simulation.

Figure 6 shows results of evolutionary approximation of one circuit instance – 16-bit signed adder. According to the boxplots size, the proposed evolutionary method provides relative stable results independently of the number of executed evolutionary runs. It is evident that the power consumption

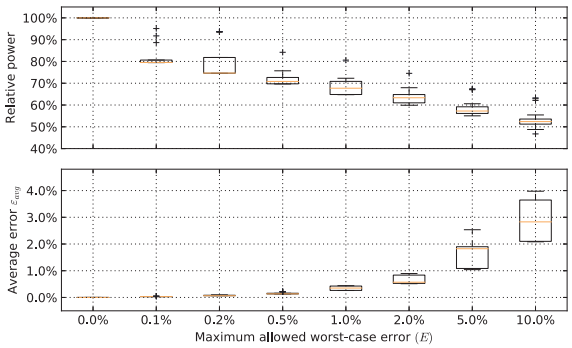

Fig. 6: Parameters of evolved approximate 16-bit adders for various error levels. For each E, 20 solutions are considered.

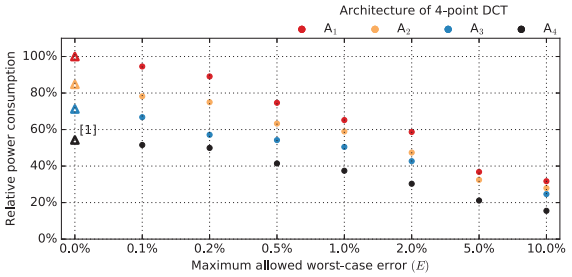

Fig. 7: Parameters of various DCT architectures constructed using the evolved adders and subtractors. The triangles represent the implementations employing the accurate components.

decreases when error E is increasing. Naturally, the average-case error follows the opposite trend.

Parameters of all (i.e. 32) obtained implementations of 4-point DCT are summarized in Figure 7. The architectures utilizing accurate adders and subtractors (symbol A1, $E = 0$) exhibit the highest power consumption. The power consumption decreases with increasing number of adders removed from MCM as well as decreasing bit-width of adders and subtractors placed in the third part of DCT architecture. As a consequence of that, architecture A4 shows a 45% reduction in power consumption compared to A1. When we compare the power consumption of seven approximate versions of each architecture, we can see that power consumption decreases with increasing E. For $E = 10\%$, the architectures exhibit very similar power consumption.

The impact of the approximate DCT on the quality of obtained video sequences was evaluated on six common test video sequences[2]. Results for three of them are shown in Figure 5. For each video sequence, we measured the peak signal-to-noise ratio (PSNR) between the original frames and frames encoded and subsequently decoded by HEVC codec. To evaluate PSNR, 100 video frames were used. When we compare parameters of the architectures utilizing the accurate adders and subtractors (see triangles in Figure 5), it is evident

[1]Reference software for ITU-T H.265 high efficiency video coding available at https://www.itu.int/rec/T-REC-H.265.2

[2]YUV video sequences available at http://trace.eas.asu.edu/yuv/

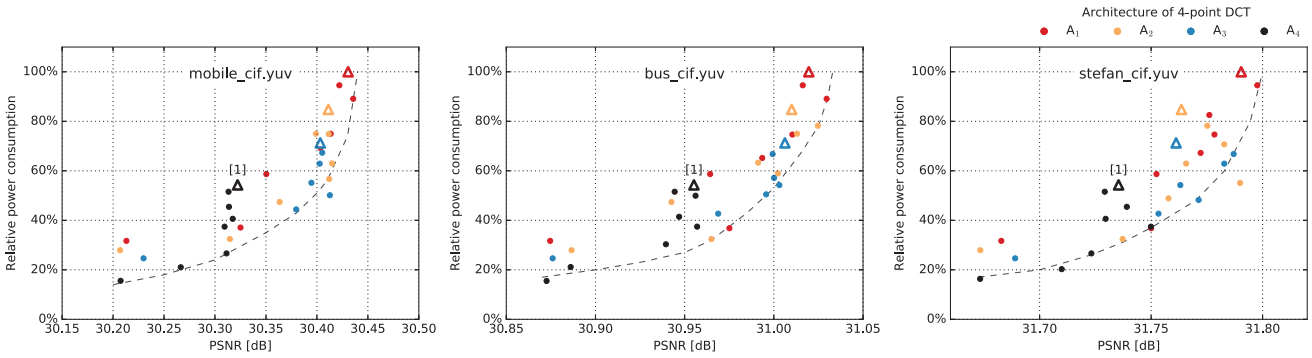

Fig. 5: The quality expressed in terms of average PSNR between original and encoded video sequences and relative power consumption evaluated for three common video sequences. Each point corresponds with one implementation of 4-point DCT. The triangles represent the implementations employing the accurate components.

that A4 proposed in [1] provides the worst results. In all cases, A2 provides better results than A3 and A3 provides better results than A4. The difference in quality between A2 and A3 is very small, however, A3 is able to achieve more than 30% power reduction. When approximate adders and subtractors are introduced to DCT, the dependency between the power consumption and quality is more complex. It happens, for example, that A1 with approximate adders/subtractors exhibiting 0.1% worst-case error provides better PSNR then the exact A1. We hypothesize that this phenomenon is related to the construction of DCT coefficient values in the reference HEVC implementation.

VII. Conclusions

We proposed a new method for optimization and approximation of the DCT block employed in the HEVC standard. The proposed multiplierless DCT implementations contain approximate adders and subtractors that were obtained using CGP. In order to manage the complexity of evolutionary approximation, the worst and average errors were determined by means of BDDs. Under conditions of our experiments, evolved implementations show better quality/power trade-offs than relevant implementations available in the literature. Our future work will be devoted to applying the proposed approximation method on other blocks of HEVC in order to find even better quality/power trade-offs.

Acknowledgments

This work was supported by Czech science foundation project GA16-17538S.

References

[1] M. Jridi and P. Meher, "A scalable approximate dct architectures for efficient HEVC compliant video coding," *IEEE Trans. Circuits Syst. Video Technol*, pp. 1–10, 2016.

[2] G. J. Sullivan, J. R. Ohm *et al.*, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol*, vol. 22, no. 12, pp. 1649–1668, 2012.

[3] F. Bossen, B. Bross *et al.*, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol*, vol. 22, no. 12, pp. 1685–1696, 2012.

[4] S. Venkataramani, A. Sabne, V. J. Kozhikkottu, K. Roy, and A. Raghunathan, "SALSA: systematic logic synthesis of approximate circuits," in *The 49th Annual Design Automation Conference 2012, DAC'12*. ACM, 2012, pp. 796–801.

[5] A. Ranjan, A. Raha, S. Venkataramani, K. Roy, and A. Raghunathan, "ASLAN: Synthesis of approximate sequential circuits," in *Proc. Conference on Design, Automation and Test in Europe*, ser. DATE'14. EDA Consortium, 2014, pp. 1–6.

[6] S. Jain, S. Venkataramani, and A. Raghunathan, "Approximation through logic isolation for the design of quality configurable circuits," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 612–617.

[7] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Trans. on CAD of Integr. Circuits and Systems*, vol. 32, no. 1, pp. 124–137, 2013.

[8] A. Raha, H. Jayakumar, and V. Raghunathan, "A power efficient video encoder using reconfigurable approximate arithmetic units," in *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, 2014, pp. 324–329.

[9] F. S. Snigdha, D. Sengupta, J. Hu, and S. S. Sapatnekar, "Optimal design of JPEG hardware under the approximate computing paradigm," in *2016 53nd ACM/EDAC/IEEE Design Automation Conf. (DAC)*, 2016, pp. 1–6.

[10] R. Hrbacek, V. Mrazek, and Z. Vasicek, "Automatic design of approximate circuits by means of multi-objective evolutionary algorithms," in *Proc.of the 11th Int. Conf. on Design and Technology of Integrated Systems in Nanoscale Era*. IEEE, 2016, pp. 239–244.

[11] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "A low latency generic accuracy configurable adder," in *Proc. 52nd Annual Design Automation Conference*. ACM, 2015, pp. 86:1–86:6.

[12] H. Jiang, J. Han, and F. Lombardi, "A comparative review and evaluation of approximate adders," in *Proc. 25th Edition on Great Lakes Symposium on VLSI*. ACM, 2015, pp. 343–348.

[13] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: Modeling and analysis of circuits for approximate computing," in *2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2011, pp. 667–673.

[14] M. Soeken, D. Grosse, A. Chandrasekharan, and R. Drechsler, "BDD minimization for approximate computing," in *21st Asia and South Pacific Design Automation Conf. ASP-DAC 2016*. IEEE, 2016, pp. 474–479.

[15] Z. Vasicek and L. Sekanina, "Evolutionary design of complex approximate combinational circuits," *Genetic Programming and Evolvable Machines*, vol. 17, no. 2, pp. 1–24, 2016.

[16] 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.

[17] J. F. Miller, *Cartesian Genetic Programming*. Springer-Verlag, 2011.

[18] Z. Vasicek and L. Sekanina, "Evolutionary approach to approximate digital circuits design," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 432–444, 2015.