TELKOMNIKA, Vol.17, No.5, October 2019, pp.2457~2464
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v17i5.12815

HEVC 2D-DCT architectures comparison for FPGA and ASIC implementations

Ainy Haziyah Awab*[1], Ab Al-Hadi Ab Rahman[2], Mohd Shahrizal Rusli[3], Usman Ullah Sheikh[4], Izam Kamisian[5], Goh Kam Meng[6]
[1,2,3,4,5]School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia
[6]Faculty of Engineering and Technology, Tunku Abdul Rahman University College, Kuala Lumpur, Malaysia
*Corresponding author, e-mail: ainy.haziyah@gmail.com

Abstract

This paper compares ASIC and FPGA implementations of two commonly used architectures for 2-dimensional discrete cosine transform (DCT), the parallel and folded architectures. The DCT has been designed for sizes 4x4, 8x8, and 16x16, and implemented on Silterra 180nm ASIC and Xilinx Kintex Ultrascale FPGA. The objective is to determine suitable low energy architectures to be used as their characteristics greatly differ in terms of cells usage, placement and routing methods on these platforms. The parallel and folded DCT architectures for all three sizes have been designed using Verilog HDL, including the basic serializer-deserializer input and output. Results show that for large size transform of 16x16, ASIC parallel architecture results in roughly 30% less energy compared to folded architecture. As for FPGAs, folded architecture results in roughly 34% less energy compared to parallel architecture. In terms of overall energy consumption between 180nm ASIC and Xilinx Ultrascale, ASIC implementation results in about 58% less energy compared to the FPGA.

Keywords*: ASIC, DCT, FPGA, HEVC, low energy*

Copyright © 2019 Universitas Ahmad Dahlan. All rights reserved.

1. Introduction

The latest video coding standard for video compression was developed by Joint Collaborative Team-Video Coding (JCT-VC) is known as high-efficiency video coding (HEVC). HEVC is the replacement of the previous standard that is Advanced Video Coding (AVC/H.264) [1]. The architecture of HEVC is based on block-based hybrid video coding approach [2]. HEVC uses variable transform unit (TU) sizes for DCT that is 4x4, 8x8, 16x16 and 32x32, and also the discrete sine transforms (DST) with size 4x4. HEVC achieves double rate better video compression efficiency compared to AVC/H.264 standard [1, 3]. However, the TU sizes for DCT in AVC/H.264 is only 8x8, as implemented in [4-7]. The function of DCT is to reduce the redundancies by transforming the spatial domain into the spectral domain and it is widely used in image and video compression technology. In this paper, the main goal is to design the 2D-DCT architecture for transform sizes of 4x4, 8x8, and 16x16 and evaluate the most suitable architectures for FPGA and ASIC platforms.

Meher et al. [8] proposed reusable architecture of integer DCT which provides the same throughput with 32 output coefficient per cycle for all TU, but produces a higher gate count. It also proposed folded and parallel HEVC 2D-DCT architectures for 8K and 4K video applications. The work by Basiri et al. [9] propose a multiplier unit with configurable carry save adder (CSA) tree implemented in 32-point 1D integer DCT architecture. The 1D-DCT architecture also uses the parallel and folded design. The 32x32-point parallel architecture gives good improvement by using 45nm CMOS TSMC library. Another work by Mehul Tiketar at el. [10], which utilizes a multiplierless multiple constant multiplication (MCM) instead of regular multipliers to reduce area overhead and applied data-gating to improve the power efficiency. The folded and parallel architectures, and other specialized design techniques described in these papers are implemented in the present paper. Apart from that, separability architecture has implemented for variable-length DCT HEVC where it uses a register and transposition memory as the block structure in 2D-DCT [11].

Received July 29, 2018; Revised March 22, 2019; Accepted April 25, 2019

There is relatively limited work for DCT implementation on FPGA [12]. One such work is given in [13] that implements and explores the design space of the full HEVC DCT. The design covers all valid DCT sizes and also the 4x4 DST [14, 15]. However, it implements at high-level and includes various architectural optimizations such as actor merging, pipelining, etc [16]. Based on [17], the gap between the FPGAs and ASICs are measured on area, performance and power consumption. In term of dynamic power consumption, FPGAs achieve approximately 14 times more than ASICs. While this gap is generally well known, suitable DCT architectures for FPGA or ASIC has not been studied in detail. Most of the works in literature implements on ASIC technologies where it is shown that a parallel architecture results in highest performance, with tradeoff on area and power. Folded architecture on the hand has shown to be able to reduce size and power at the expense of performance. Due to the high performance parallel architecture in ASIC, energy efficiency is also generally better. However, for FPGA a different case is expected due to the unpredictable placement and routing compared to ASIC, especially for large size transforms. Thus the present paper provides experimental results on comparing energy efficiency for small (4x4 and 8x8) and large size (16x16) transforms for FPGA and ASIC implementations. This paper is organized as follows. In section 2, the theory of DCT are described. In section 3, the parallel and folded architectures are presented. In section 4, the results of ASIC and FPGA are assessed in terms of energy per block, maximum frequency, throughput, power consumption, area and gate count. Section 5 concludes the paper.

2. DCT Theory

The (1) and (2) is the basic equation of N-point one-dimensional (1D) DCT transform as defined in [18, 19]:

$$y(k) = a(k)\sum_{n=0}^{N-1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right], 0 \le k \le N-1 \quad (1)$$

$$a(0) = \sqrt{\frac{1}{N}}; \quad a(k) = \sqrt{\frac{2}{N}}, 1 \le k \le N-1 \quad (2)$$

where $x(n)$ is the input data and $y(k)$ is the output data? For *N=4*, the equation 1D-DCT can be written in matrix form as given in (3), where c is the constant matrix:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad (3)$$

One of the properties of 2D-DCT is separable it in two ways, a column-wise 1D-DCT and followed by a row-wise 1D-DCT or vice-versa [2]. Figure 1 shows the example for row and column process of 4x4 2D-DCT [9]. It starts with the row process first, each row of the input matrix the 1D-DCT is performed and the intermediate results are stored in transposition buffer matrix row by row. Next, the 1D-DCT is performed again column by column from the transposition buffer matrix for the column process. The results of the column process are required for 2D-DCT.

The elements of the forward transform matrix are denoted in the HEVC standard [2, 20]. The smaller size of transform matrices can be derived from the 32x32 matrix as shown in (4).

$$d_{i,j}^{N} = d_{i(32/N),j}^{32} \qquad where\ i,j = 0, \dots\dots, N-1 \quad (4)$$

Let $y_{4x4\ DCT}$ denote the 4x4 transform matrix. Based on [2, 21], the elements of $y_{4x4\ DCT}$ can obtain as follows:

$$y_{4x4\ DCT} = \begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix} \quad (5)$$

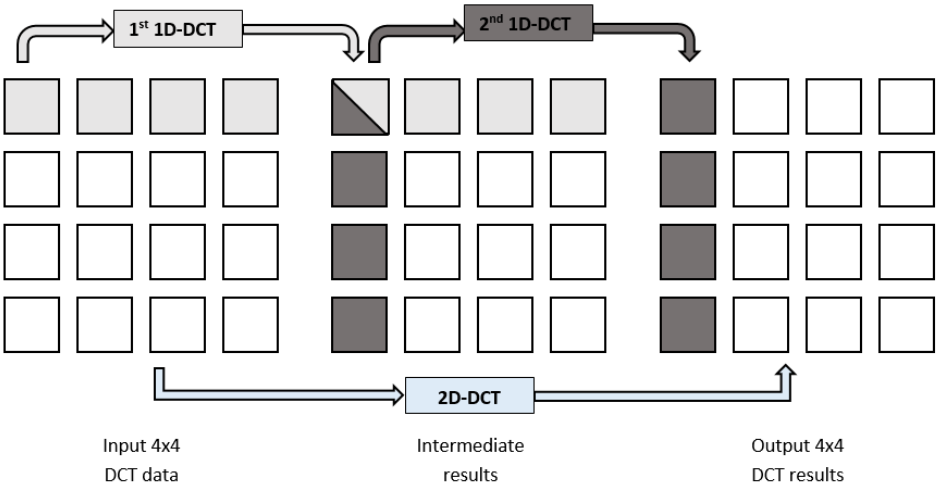

Figure 1. Example for row and column process of 4x4 2D-DCT

By follow the symmetry property of the transform matrix, the number of necessary computations can be reduced by broken down (5) into Even and Odd part [22]. For the even matrix, the 0th and 2nd line of the horizontal and vertical is selected for row and column respectively. For the odd matrix, the row and column are selected from the 1st and 3rd line of the horizontal and vertical. The calculation of Even and Odd part for the 4x4 matrix used in this work can be computed in matrix form as shown in (6) and (7), and the output 1D-DCT in (8).

$$\begin{bmatrix} Even_0 \\ Even_1 \end{bmatrix} = \begin{bmatrix} 64 & 64 \\ 64 & -64 \end{bmatrix} \begin{bmatrix} x_0 \\ x_2 \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} Odd_0 \\ Odd_1 \end{bmatrix} = \begin{bmatrix} 83 & 36 \\ 36 & -83 \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} \quad (7)$$

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} Even_0 + Odd_0 \\ Even_1 + Odd_1 \\ Even_1 - Odd_1 \\ Even_0 - Odd_0 \end{bmatrix} \quad (8)$$

By using the same method, the equations for 8x8, 16x16 and 32x32 transform block also can be derived [2].

3. 2D-DCT Architecture

This section describes the 2D-DCT for the 4x4 TU block size. Basically, this work is designing the 4x4 1D-DCT by using the combination of four 4-point DCT, where each N-point module is located horizontally. The structure of 1D-DCT is depicted in Figure 2. The complete 1D-DCT for 4x4 TU consists of 16 input/output signal. For the larger TU size such as 8x8, 16x16 and 32x32, it consists of 64, 256 and 1024 input/output signal respectively. Note that a serializer-deserializer (SERDES) modules can be used to stream the inputs and outputs. In addition, for 8x8, 16x16, and 32x32, it needs eight 8-point, sixteen 16-point DCT, and thirty two 32-point DCT respectively to perform the complete 1D-DCT. The HEVC 2D-DCT architecture in this work are parallel and folded 2D-DCT. The details on the architectures can be found in [8-10] and [23-26]. The architectures are also discussed briefly in this section.

3.1. Parallel Architecture

Figure 2 shows the parallel 4x4 2D-DCT architecture. It consists of two 1D-DCT modules, where the first 1D-DCT module is used to perform the row process and the other 1D-DCT module is used to perform the column process. In this architecture, the register-based

transpose memory is not used. All the input signal is fed simultaneously into the first 1D-DCT module and 1D-DCT module will do some operation of the adder, subtractor, and multiplication. The results from first 1D-DCT that is the row process will directly transpose to the next 1D-DCT module that is column process values. The second 1D-DCT module will do the same process as the first one. The results of the 2D-DCT module are obtained from the column process in the second 1D-DCT module.

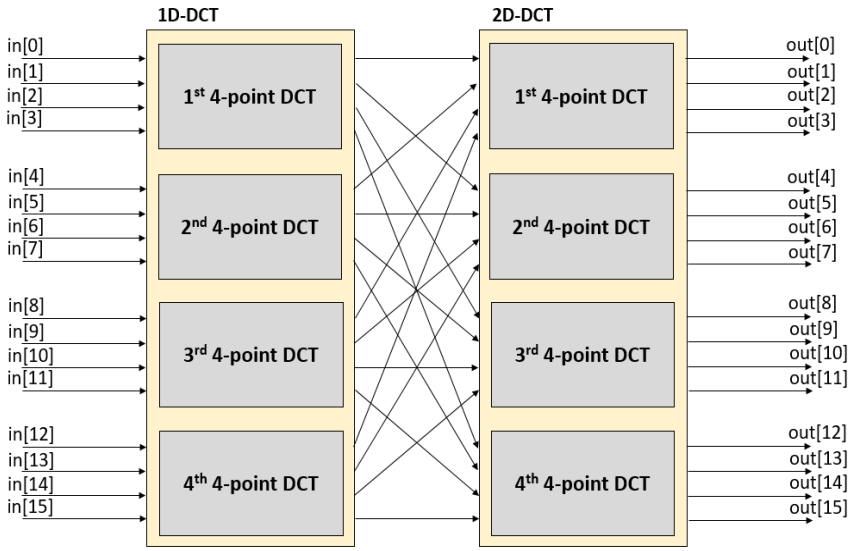

Figure 2. The structure of parallel 4x4 2D-DCT architecture

3.2. Folded Architecture

The folded 4x4 2D-DCT architecture is shown in Figure 3. This architecture consists of one 1D-DCT module, a register-based transpose memory, and some multiplexer. The block of 1D-DCT is used to perform both processes of row and column. The multiplexer is used as a control signal in this circuit. All the input signal is fed simultaneously into the multiplexer.

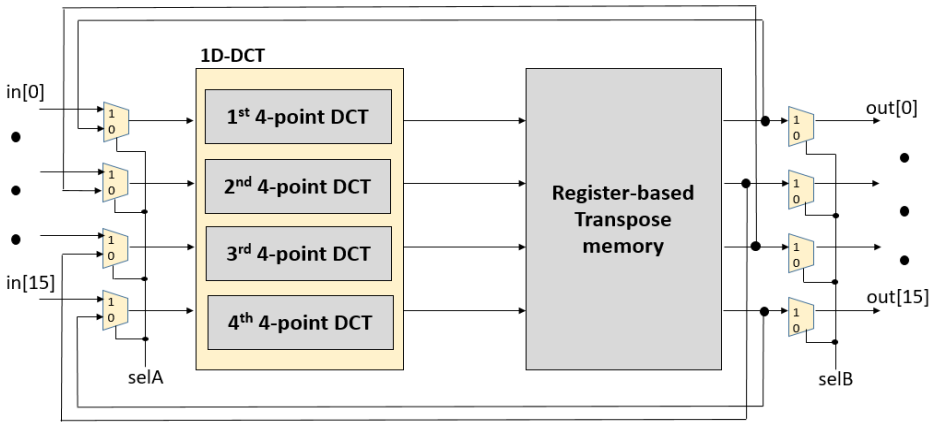

Figure 3. The structure of folded 4x4 2D-DCT architecture

Figure 4 is the state diagram of the control signal for folded architecture. The value of *N* for 4x4 transform is 16. For the first cycle at state S0, the signal selA is set as high and the input signal is fed to the multiplexer and 1D-DCT module, it will perform the row process. At state S1, the register-based transpose memory receives a signal from 1D-DCT and it will store the

intermediate results of successive row ($N_i = 16$). It will remain at the same state if the value of *N* is not enough otherwise it will go the next state S2. At state S2, the signal selB is low and it will go back to the input part for the second cycle. For the second cycle at state S3, the signal selA is low and selB is high. The results of the register-based transpose memory are fed as input to the multiplexer as well as the 1D-DCT module and both of them will do the same process as previous for the next successive column ($N_j = 16$). The results of the register-based transpose memory from the second cycles are defined as the results of the 2D-DCT module. The state diagram also can be reused for the larger size by change the value of N, it depends on the TU size.

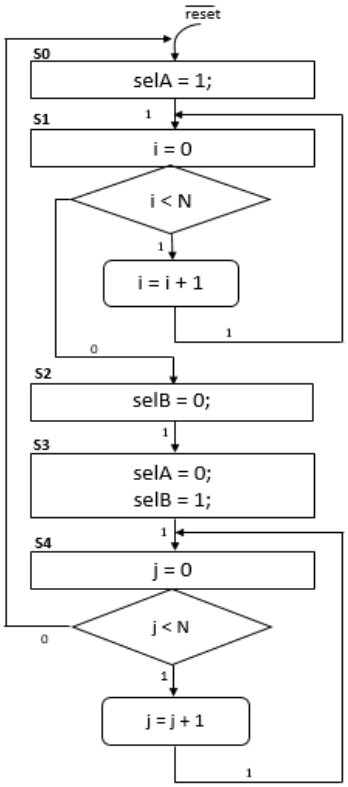

Figure 4. State diagram of control signal for folded architecture

4. Results and Analysis

The 2D-DCT architecture for 4x4, 8x8 and 16x16 TU size has been designed using Verilog HDL and implemented in Silterra 180nm technology process for ASIC and Xilinx Kintex Ultrascale for FPGA. The simulation results obtained from Xilinx Vivado for FPGA is compared to the one obtained from Synopsys DC compiler to ensure the results are correct. SERDES modules have been used to serialize and deserialize the parallel input and output. The word length of each input/output pixels of 1D-DCT and 2D-DCT is 16 bits. This section discusses energy per block and other performance on ASIC and FPGA designs for both parallel and folded architectures.

4.1. Energy per Block

The graph in Figure 5 and Figure 6 shows the plot of energy per block in FPGA and ASIC respectively. Energy is calculated using the formula E=Pt, where P is the total power, and t is the time to process a single block. The energy per block is increased slightly in parallel with an increase in TU size. In FPGA, it can be seen that there is minimal difference for small and medium sized blocks. For large 16x16 block however, parallel architecture consumes 150.22nJ, while folded architecture consumes 98.84nJ. This results in roughly 34% less energy for folded architecture compared to parallel architecture. The main reason for this is due to

the significantly more wiring in parallel architecture, which is generally known to have a negative effect on its performance and power for FPGAs [17].

As shown in Figure 6 for ASIC implementation, a parallel architecture for all TU size yields lower energy compared to the folded architecture. Similar to FPGAs, for small and medium sized transforms, the results are almost similar. However, for large 16x16, it can be seen that the parallel architecture consumes 2.61nJ while the folded architecture consumes 3.72nJ. This results in roughly 30% less energy for the parallel architecture compared to the folded architecture. Another interesting observation is the energy comparison between ASIC and FPGA. It can be seen that implementation on Silterra 180nm results in roughly 150x less energy compared to implementation on Xilinx Kintex Ultrascale (using 14nm technology). This is mainly attributed to the intrinsically high power of FPGAs compared to ASICs.

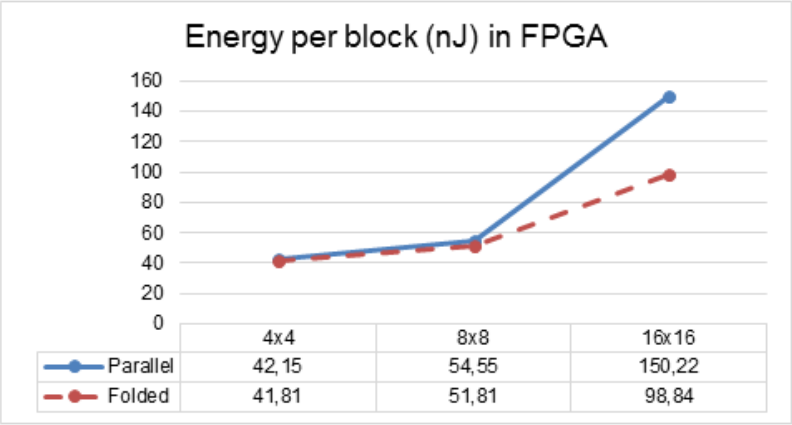

Figure 5. Energy per block in FPGA

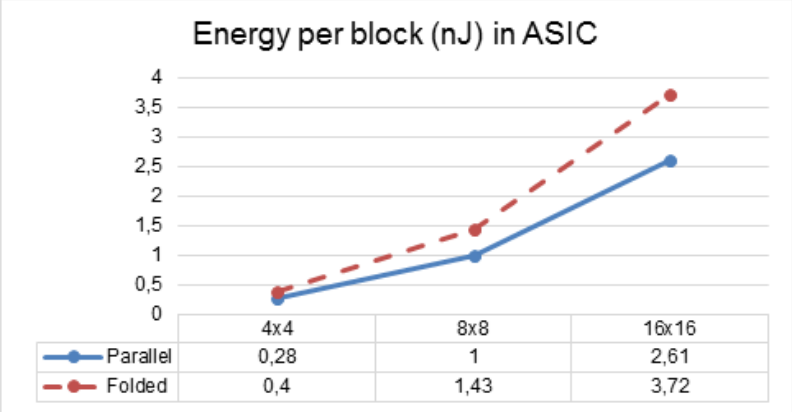

Figure 6. Energy per block in ASIC

4.2. Other performance on ASIC and FPGA

Table 1 shows the results of others performance comparison in terms of maximum frequency, throughput, power, area and gate count. For the ASIC implementation, the clock period in this work is set at 20ns. The maximum frequency has been obtained using the folded architecture, which is twice of the parallel architecture. This is because the critical path is roughly twice longer in the parallel architecture. Therefore, in terms of throughput, parallel architectures have higher throughput compared to the folded architecture. Note that clock cycle latency is almost similar between the two architectures since a SERDES is used on the inputs

and outputs. For a 4x4 block, the clock cycle latency is 35 clock cycles for parallel architecture, and 39 clock cycles for folded architecture.

In terms of power for ASIC, the folded architecture of 4x4, 8x8 and 16x16 consumes 1.4 times more power consumption than the parallel architecture. Besides, the total core area of parallel architecture is about twice bigger than folded architecture, due to the more registers were used in parallel architecture. Total gate count of 16x16 block size is 7 times more than 8x8 transform size for both architecture due to many blocks (adders, multiplication, and shifters) were used in the design.

For FPGAs, an interesting observation from the results is the maximum frequency is higher on the parallel architecture. Furthermore, power consumption is also higher on the parallel architecture. The parallel architecture also has higher throughput and the 16x16 block size in FPGA is roughly 2 times more throughput compared to ASIC. The resources used by these design also reported in Table 1 which include the Look-Up Table (LUT), Flip-Flop (FF) and DSP. In terms of area, the ASIC is that the resultant circuit is permanently drawn into silicon whereas in FPGAs the circuit is made by connecting a number of configurable blocks. This is difficult to compare more details about the area between them. As mentioned, this is possibly due to the significantly more wiring on the parallel architecture. Because of this, the folded architecture generally results in better energy efficiency in FPGAs.

Table 1. The Results of the Two Architecture and Sizes in ASIC and FPGA

Platform	Architecture	Parallel			Folded		
	Transform size	4x4	8x8	16x16	4x4	8x8	16x16
ASIC	Max Frequency (MHz)	53.53	51.49	35.30	100.81	97.18	65.19
	Throughput (Gpixel/s)	0.81	3.30	9.04	0.81	3.11	8.34
	Power (W)	3.307m	11.650m	30.503m	4.699m	16.766m	43.478m
	Area $(mm)^2$	0.17	1.45	11.08	0.09	0.89	6.31
	Gate count	17K	145K	1108K	9K	89K	631K
FPGA	Max Frequency (MHz)	128.75	86.19	69.45	67.23	59.91	38.42
	Throughput (Gpixel/s)	2.06	5.52	17.78	0.54	1.92	4.92
	Power (W)	0.493	0.638	1.757	0.489	0.606	1.156
	LUT	991	5049	123418	1036	4158	30276
	FF	587	2141	8286	850	3177	12413
	DSP	48	368	1368	20	184	1240

5. Conclusion

In this paper, a comparison study has been performed for 2-D HEVC DCT for ASIC and FPGA implementations. The aim is to determine suitable architectures for these implementation platforms. Furthermore, overall energy efficiency comparison between FPGAs and ASICs have also been determined. The study includes the design and implementation of two commonly used 2-D DCT architectures which are the parallel and folded. Three DCT sizes have been designed and compared, which are the 4x4, 8x8, and 16x16. Results show that parallel architecture is most suitable for ASIC due to a more predictable instance placement and routing; while the significantly more wiring in parallel architecture results in relatively poor performance in FPGAs. Results also show that using the Silterra 180nm technology achieves roughly 58x less energy compared to using the Xilinx Kintex Ultrascale at 14nm technology. Future work is to complete the HEVC DCT for size 32x32 and 4x4 DST.

Acknowledgements

The authors would like to acknowledge Universiti Teknologi Malaysia (UTM) for providing the facilities and funding for this research (Research University Grant (GUP) vote numbers 17H41 and 14J48). The authors would also like to acknowledge the support of the Ministry of Higher Education (MOHE) under the Fundamental Research Grant Scheme (FRGS) with the reference code FRGS/1/2016/TK04/TARUC/02/1.

References

[1] Sullivan GJ, Ohm JR, Han WJ, Wiegand T. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE T Circ Syst Vid*. 2012; 22(12): 1649-1668.

[2] Budagavi M, Fuldseth A, Bjøntegaard G, Sze V, Sadafale M. Core transform design in the high efficiency video coding (HEVC) standard. *IEEE Journal of Selected Topics in Signal Processing*. 2013; 7(6): 1029–1041.

[3] Sze V, Budagavi M, Sullivan GJ. High efficiency video coding (HEVC). In: Integrated Circuit and Systems, Algorithms and Architectures. Springer. 2014: 1–375.

[4] NM Zabidi, AAH Ab Rahman. VLSI Design of a Fast Pipelined 8x8 Discrete Cosine transform, *International Journal of Electrical and Computer Engineering (IJECE)*. 2017; 7(3): 1430-1435.

[5] KA Wahid, M Martuza, M Das, C McCrosky. Efficient hardware implementation of 8x8 integer cosine transforms for multiple video codecs. *Journal of Real-Time Processing*. 2011: 1-8.

[6] A Madanayake et al. Low-Power VLSI Architectures for DCT/DWT: Precision vs Approximation for HD Video, Biomedical, and Smart Antenna Applications. *IEEE Circuits and Systems Magazine*. 2015; 15(1): 25-47.

[7] M Fu, GA Jullien, VS Dimitrov, M Ahmadi. *A low-power DCT IP core based on 2D algebraic integer encoding*. Proceedings of the International Symposium on Circuits Systems (ISCAS). 2004; 2: 765-768.

[8] Meher PK, Park SY, Mohanty BK, Lim KS, Yeo C. Efficient Integer DCT Architectures for HEVC*. IEEE T Circ Syst Vid*. 2014; 24(1): 168-178.

[9] Mohamed Asan Basiri M. and Noor Mahammad. Sk. *High Performance Integer DCT Architectures for HEVC. 2017* 30th International Conference on VLSI Design and 2017 16th International Conference on Embedded Systems (VLSID). Hyderabad. 2017: 121-126.

[10] M Tikekar, CT Huang, V Sze. A Chandrakasan. *Energy and area-efficient hardware implementation of HEVC inverse transform and dequantization.* 2014 IEEE International Conference on Image Processing (ICIP). Paris. 2014: 2100-2104.

[11] NC Vayalil, J Haddrill, Y Kong. *An Efficient ASIC Design of Variable-Length Discrete Cosine Transform for HEVC.* 2016 European Modelling Symposium (EMS). Pisa. 2016: 229-233.

[12] Chen M, Zhang YZ, Lu C. Efficient architecture of variable size HEVC 2D-DCT for FPGA platforms. *Aeu-Int J Electron* C. 2017: 73: 1-8.

[13] KZ Yion, AAH Ab Rahman. Exploring the Design Space of HEVC Inverse Transforms with Dataflow Programming. *Indonesian Journal of Electrical Engineering and Computer Science*. 2017; 6(1): 104-109.

[14] J Nan, N Yu, W Lu, D Wang. *A DST Hardware Structure of HEVC*. 2015 2nd International Conference on Information Science and Control Engineering. Shanghai. 2015: 546-549.

[15] Masera M, Martina M, Masera G. *Area efficient DST architectures for HEVC*. 13th Conference on Ph.D. Research in Microelectronics and Electronics (PRIME). 2017: 101-104.

[16] H Amer, AAH Ab Rahman, I Amer, M Mattavelli. *Methodology and Technique to Improve Throughput of FPGA-based CAL dataflow Programs: Case Study of the RVC MPEG-4 SP Intra Decoder*. 2011 IEEE Workshop on Signal Processing Systems (SiPS). 2011: 186-191.

[17] Kuon I, Rose J. Measuring the gap between FPGAs and ASICs. *IEEE T Comput Aid D*. 2007; 26(2): 203-215.

[18] NC Vayalil, J Haddrill, Y Kong. *An Efficient ASIC Design of Variable-Length Discrete Cosine Transform for HEVC*. European Modelling Symposium (EMS). Pisa. 2016: 229-233.

[19] M Jridi, A Alfalou. *A low-power, high-speed DCT architecture for image compression: Principle and implementation.* 2010 18th IEEE/IFIP International Conference on VLSI and System-on-Chip. Madrid. 2010: 304-309.

[20] K Tong et al*. A fast algorithm of 4-point floating DCT in image/video compression*. 2012 International Conference on Audio, Language and Image Processing. Shanghai. 2012: 872-875.

[21] Hong L, He WF, He GH, Mao ZG. Area-efficient HEVC IDCT/IDST architecture for 8Kx4K video decoding. *IEICE Electron Expr*. 2016; 13(6).

[22] AE Ansari, A Mansouri, A Ahaitouf. *An efficient VLSI architecture design for integer DCT in HEVC standard*. 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA). Agadir. 2016: 1-5.

[23] W Zhao, T Onoye, T Song. *High-performance multiplierless transform architecture for HEVC*. 2013 IEEE International Symposium on Circuits and Systems (ISCAS2013). Beijing. 2013: 1668-1671.

[24] M Martuza, K Wahid. *A cost effective implementation of 8×8 transform of HEVC from H.264/AVC*. 2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE). Montreal. 2012: 1-4.

[25] E Kalali, AC Mert, I Hamzaoglu. A computation and energy reduction technique for HEVC Discrete Cosine Transform. *IEEE Transactions on Consumer Electronics*. 2016; 62(2): 166-174.

[26] Masera M, Martina M, Masera G. Adaptive Approximated DCT Architectures for HEVC. *IEEE T Circ Syst Vid*. 2017; 27(12): 2714-25.