Contents lists available at ScienceDirect

International Journal of Electronics and Communications (AEÜ)

journal homepage: www.elsevier.com/locate/aeue

Regular paper

Efficient architecture of variable size HEVC 2D-DCT for FPGA platforms

Min Chen [a], Yuanzhi Zhang [b], Chao Lu [b,*]

[a] MulticoreWare, Inc., St. Louis, MO, USA
[b] Department of Electrical and Computer Engineering, Southern Illinois University, Carbondale, IL 62901, USA

ARTICLE INFO

Article history:
Received 3 October 2016
Revised 11 December 2016
Accepted 25 December 2016

Keywords:
H.265/HEVC
Two-dimensional discrete cosine transform (2D-DCT)
FPGA platform
Hardware architecture

ABSTRACT

This study presents a design of two-dimensional (2D) discrete cosine transform (DCT) hardware architecture dedicated for High Efficiency Video Coding (HEVC) in field programmable gate array (FPGA) platforms. The proposed methodology efficiently proceeds 2D-DCT computation to fit internal components and characteristics of FPGA resources. A four-stage circuit architecture is developed to implement the proposed methodology. This architecture supports variable size of DCT computation, including 4×4, 8×8, 16×16, and 32×32. The proposed architecture has been implemented in System Verilog and synthesized in various FPGA platforms. Compared with existing related works in literature, this proposed architecture demonstrates significant advantages in hardware cost and performance improvement. The proposed architecture is able to sustain 4 K@30 fps ultra high definition (UHD) TV real-time encoding applications with a reduction of 31–64% in hardware cost.

1. Introduction

Rapid advances in consumer electronics have resulted in a variety of emerging video coding applications. Typical examples include ultra-high definition (UHD) 4 K/8 K TV [1] or unmanned aerial vehicle (UAV) reconnaissance and surveillance [2,3], which demands aggressive video compression requirement. Despite the success in the last decade, video compression efficiency of H.264 standard cannot satisfy stringent requirements [4]. Alternatively, recently established H.265/HEVC standard has great potential to improve video compression efficiency by around 50%, while retaining the same video quality as H.264 [5,6]. As a result, HEVC has been viewed as one of the most promising standard to overcome these challenges [7,8].

Optimized coding efficiency in HEVC is attributed to increased computational complexity [9]. For example, Discrete Cosine Transform (DCT) and Inverse Discrete Cosine Transform (IDCT) are indispensable building blocks of HEVC hardware implementation [10]. Previous study has reported that DCT and IDCT computation in HEVC is estimated as 11% of total computational complexity in hardware implementations [6]. Due to computational similarity between DCT and IDCT, they can alter the coefficient matrix and share same circuit architecture. Large block sizes of DCT and IDCT (*i.e.*, 16×16 and 32×32) are supported in HEVC standard, while H.264 only accepts smaller block sizes (*i.e.*, 4×4 and 8×8). Large sizes of DCT and IDCT help to improve coding efficiency. For example, the use of 16×16 and 32×32 DCTs and IDCTs in HEVC reduces bit rate up to 10.1% [11]. However, the associated hardware cost rises significantly. For instance, a transpose buffer of $32 \times 32 \times 16$ bits is needed to store one-dimensional transform coefficients for 32×32 2D-DCT, while a transpose buffer of $8 \times 8 \times 16$ bits is sufficient for 8×8 2D-DCT. The hardware cost of transpose buffer will continue to increase in next-generation video coding standard, since it will include 64×64 and 128×128 DCT/IDCT operations [12]. Therefore, it is necessary to investigate efficient circuit architectures to reduce hardware implementation cost and computational complexity [13–15].

Nowadays, computational resources in FPGA are adequate to implement HEVC codecs, such as FPGA implementation of 4 K real-time HEVC decoder [16] and a full HD real-time HEVC main profile decoder [17]. The use of FPGA instead of application-specific integrated circuit (ASIC), shortens design time to market, and hence is a preferred approach for small volume production. Therefore, the study of HEVC FPGA implementation is gaining more and more attention. In order to satisfy real-time and high-efficiency coding in these emerging video applications, a few design methodologies and circuit architectures have been developed [11,18–24]. In [11], the authors presented an IDCT implementation with zero-column skipping technique to boost energy and area efficiency. However, the use of single-port static random access memory (SRAM) does not allow pipeline operation, hence the throughput of this design is impeded. The proposed architecture in [18] focuses on reduction

* Corresponding author.
E-mail addresses: chenm003@gmail.com (M. Chen), yzzhang@siu.edu (Y. Zhang), chaolu@siu.edu (C. Lu).

http://dx.doi.org/10.1016/j.aeue.2016.12.024
1434-8411/© 2016 Elsevier GmbH. All rights reserved.

of hardware utilization. A series of hardware minimization techniques was applied, including operation reordering, multiplications to shift-adds conversion, etc. In [19–21], the proposed designs utilized distributed arithmetic hardware to perform multiplications in 2D-DCT. These approaches are efficient for smaller DCT computation (*e.g.,* 8×8). The proposed architectures in [19–21] did not consider internal features and characteristics of FPGA platforms. In [22], a new algorithm and processing architecture for 2D-DCT were presented to achieve higher energy efficiency. Recently, the researchers in [23,24] proposed FPGA-based 2D-DCT with improved area-speed efficiency. [23,24] are initial trials to efficiently utilize features and dedicated components of FPGA platforms. However, the design strategy of allocating FPGA resources to fit DCT architectures are not elaborated in details.

These existing architectures are inefficient when implementing HEVC 2D-DCT in FPGA platforms, because larger DCTs (*e.g.,* 32×32) involves a great number of transpose buffers, cascaded additions and subtractions. When a design is synthesized towards FPGA platforms, many general-purpose logics (*i.e.,* Look-up Tables (LUTs)) are utilized. Thus, the critical path delay of a synthesized design is longer, and the maximum operation frequency is degraded. On the other hand, if a designer is aware of internal resources of FPGA, the resultant architecture may fit with FPGA components and features. Thus, the synthesized design efficiently utilizes FPGA resources, such as digital signal processor (DSP) blocks, bus width, and on-chip memory. An efficient hardware architecture should always make every effort to fit FPGA resources, which is the focus of this paper.

This paper makes the following contributions: (1) our proposed design methodology takes into account of hardware resources of FPGA platforms, and efficiently utilizes bus width, DSP blocks, BRAM blocks, and on-chip memory bandwidth. Thus, the required general programmable logics (*e.g.*, LUTs) are significantly reduced, and video processing throughput is largely improved. (2) A hardware architecture is proposed to support variable DCT sizes from 4×4 to 32×32, which can also be extended to larger DCT sizes (*e.g.*, 64×64 and 128×128). This architecture facilitates hardware sharing and reusing among different DCT sizes. The design details are described and illustrated through timing diagram. (3) The proposed architecture has been synthesized in various FPGA platforms. The benefits are presented through comparisons with existing designs in literature. The proposed architecture is able to sustain 4 K@30 fps ultra high definition (UHD) TV real-time encoding applications with a reduction of 31–64% in hardware cost.

The rest of this paper is organized as follows. Section 2 reviews basic DCT algorithm, hardware components and FPGA characteristics. Section 3 describes the proposed design methodology and system architecture. In Section 4, system implementation results are provided. An in-depth comparison with related design architectures in literature is presented. Finally, Section 5 concludes the paper.

2. Related work

2.1. Basic DCT algorithm

DCT is widely used in image coding and signal processing applications. DCT transforms images from spatial-domain into frequency-domain, and provides a more efficient representation of information. A 1D-DCT computation of an $N \times N$ block size can be expressed as [10,24]

$$Y(i,j) = \sum_{k=0}^{N-1} X(k,i) \times C(j,k) \tag{1}$$

Here X and Y are the input and output data matrix, respectively. C is an $N \times N$ transform matrix. N can be 4, 8, 16 or 32 in HEVC/H.265 standard. For simplicity, integer values are usually chosen in the transform matrix C in VLSI implementation. Thus, the hardware implementation achieves finite precision DCT approximation [9]. Thanks to the separability feature, a 2D-DCT is usually decomposed into two 1D-DCTs. 1D-DCT is firstly applied on individual row of input data matrix X, then, another 1D-DCT is applied to the results from the first 1D-DCT [18]. Two 1D-DCTs are connected through a transpose buffer, which temporarily stores the results of the first 1D-DCT. Because direct hardware implementation of matrix multiplication in 1D-DCT requires intensive computation, 1D-DCT based on even-odd decomposition techniques is widely accepted to minimize computational complexity [9].

2.2. Hardware components and features of FPGA platform

FPGA is one type of pre-fabricated integrated circuits designed for rapid prototyping and functional verification. Nowadays, FPGA platform consists of five main elements: DSP blocks, look-up tables (LUTs), flip-flops, random access memory (RAM) blocks and routing matrix. The DSP blocks, including pre-adder, multiplier, accumulator, etc., are dedicated for accelerating complex arithmetic computation. A look-up table is a collection of logic gates to implement any arbitrarily user defined Boolean function. A look-up table is composed of register arrays and works as a combinational logic of inputs. Users can program look-up tables to realize any combinational logic function. Each look-up table may connect with flip-flops, which are indispensable components of sequential logic modules. RAM block is an embedded storage element, whose type could be single-port, dual-port or quad-port. For example, a dual-port RAM enables simultaneously access (i.e., write or read) by two agents. Routing matrix is used to route signals and interconnect among FPGA processing resources.

Different FPGA companies implement these five main elements differently. For example, Zynq is one FPGA of Xilinx 7-series families. This FPGA has plentiful hardware resources, such as LUTs, DSP48s, and BRAMs [25,26]. The LUTs of Zynq FPGA can be configured as either six inputs with one output, or as five inputs with multiple separate outputs. Some LUTs could be configured as 64-bit distributed RAMs or as 32-bit shift registers. A DSP48 block consists of a 25-bit pre-adder, a 25×18 two's complement multiplier, and a 48-bit accumulator. A DSP48 block may be configured as a single-instruction-multiple-data (SIMD) arithmetic unit. DSP48 block is optimized for short critical path, and hence reaches clock frequency as high as 741 MHz. In addition, on-chip dual-port block RAMs (BRAM) with port width up to 72 bits are embedded in Zynq FPGA. This BRAM also supports asymmetric read or write operations with variable port width.

3. Proposed design methodology and circuit architecture

As been reviewed in Section 2, FPGA owns rich on-chip high performance resources. It is highly desirable to create HEVC architectures to fit FPGA components and characteristics. For example, 2D-DCT involves extensive matrix multiplications. Multiplications could be implemented either by LUTs or DSP blocks inside FPGA platforms. If a design is implemented using LUTs, due to distributed locations of LUT components and long wire routing, the resultant design will exploit lots of LUTs and suffer from slower system operation. In contrast, if dedicated DSP blocks are selected to implement multiplications, its operating frequency and hardware efficiency will be improved over LUT-based design scheme.

From the above discussion, it is clear that existing 2D-DCT designs in literature do not fully explore internal resources and characteristics of FPGA platforms. In this section, we propose a FPGA-friendly DCT design methodology and circuit architecture to mitigate the design challenges of future video coding applications.

3.1. Proposed methodology

Fig. 1 shows the proposed 2D-DCT algorithm, assuming targeted FPGA is Xilinx Zynq. Input data for DCT computation is a 9-bit 32 × 32 matrix. According to the separability property, a 2-D DCT is decomposed into two subsequent 1-D DCTs. Unlike the conventional approach [18] where input data are read and executed row by row, our proposed method performs two rows in parallel during the first 1D-DCT. The results (*i.e.,* 4 points) in each clock cycle are stored in BRAM. Because maximum port width of a BRAM in Zynq is up to 72 bits, and width of intermediate results from the first 1D-DCT is 16 bits, at most 4 points can be saved during each clock cycle. 16 clock cycles are required to accomplish two rows of data, and its output result during each clock cycle is a 2 × 2 matrix, as depicted in Fig. 1. Therefore, the output pattern of 4 points (*i.e.,* 2 × 2 matrix) fits with the port width of BRAMs in Zynq, which indicates computation throughput matches bandwidth of storage memory and hence no idle computation resource. Note this output 2 × 2 matrix will be transposed (also called re-ordered) before saving into two columns of a BRAM. A total of 16 BRAMs of each size 64 × 16 bits, are needed for a 32 × 32 2D-DCT computation. During the second 1D-DCT computation, each row of intermediate results is read from all 16 BRAMs and proceeded. Each clock cycle outputs 4 points (*i.e.,* 1 × 4 matrix) as shown in Fig. 1. Thus, throughput of two subsequent 1D-DCTs are balanced, avoiding the risk of operation stall.

To increase hardware utilization efficiency, on-chip DSP blocks are preferred to realize matrix multiplications. DCT transform using even-odd decomposition results in drastic reduction in the number of multiplications and additions, when compared with direct matrix multiplication [9]. Here we propose to map butterfly transform into DSP48 blocks in Zynq, where the pre-adder, multiplier and 48-bit accumulator efficiently collaborate to generate outputs. Butterfly transform can be applied several times in DCT. Each time an even data matrix splits into smaller even and odd parts, until down to 4 × 4 size. Even parts may be reused for different DCT sizes, but odd parts are prohibited. Table 1 illustrates how hardware resources for processing one pixel point varies with depth of butterfly transform. It is apparent that more levels of butterfly transforms require less number of multipliers. We propose to apply butterfly transform only once, instead of three times as in [6]. With the resource overhead of six more multipliers, the benefit of our design is to reuse these computation elements in parallel in smaller DCT sizes. For example, if the DCT size is 4 × 4, 16 multipliers can be reconfigured in parallel to boost computation performance, compared with conventional design which consists of only 10 multipliers. Overall, 128 DSP48 blocks in Zynq FPGA are enough to proceed 2D-DCT implementations.

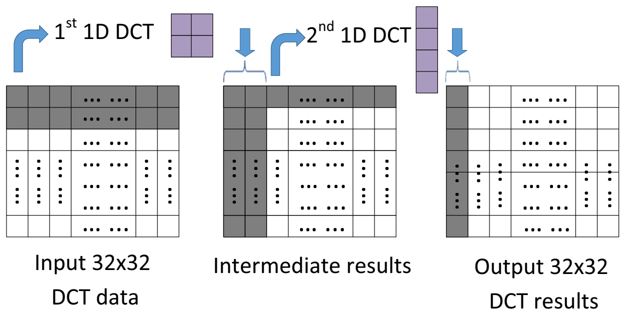


Fig. 1. Proposed 2D-DCT 32 × 32 algorithm with row and column seperation.

Table 1
Hardware resources for processing one pixel point vs. butterfly transform depth.

Butterfly Depth	# of Multipliers required in 1D-DCT	# of Adders required in 1D-DCT
0 (i.e., no butterfly transform)	32	31(bit width 16)
1	16	31(bit width 17)
2	12	31(bit width 18)
3	10	31(bit width 19)

3.2. Architecture design

Fig. 2 shows the proposed four-stage architecture to implement our design methodology. During the first stage, input DCT data are received from upstream module. Butterfly operation is executed and results are stored into a register buffer. During the second stage, data are fetched from the register buffer. These data are multiplied with coefficients, and the accumulated results are directly stored into BRAMs. Coefficients are obtained from a read only memory (ROM), whose address is sent by the control unit. So far, the computation of first 1D-DCT is done. The third stage reads data from BRAMs, executes one-level butterfly transform, and finally stores the results into a buffer register. The fourth stage reads data from the buffer register, and runs 1D-DCT again as described in the second stage with higher bit width to adapt larger dynamic range. As four pixels (64 bits) are processed every clock cycle, BRAMs are used as transpose buffers due to its port width is up to 72 bits. We only use 16 BRAMs and eliminate the use of any transpose register buffers in FPGA. During each clock cycle, the first 1D-DCT writes 2 × 2 reordered pixels into a BRAM, and the second 1D-DCT outputs 1 × 4 pixels as system output. In this architecture, the internal BRAM bandwidth and computation throughput of 2D-DCT match each other. The control unit generates the required read/write signals for BRAMs, coefficient addresses for ROM, and handshaking signals for input and output synchronization. Note the output results of 2D-DCT will be quantized in a separate quantization stage, so quantization process is not included in Fig. 2. The proposed architecture is applicable to IDCT computation after switch-

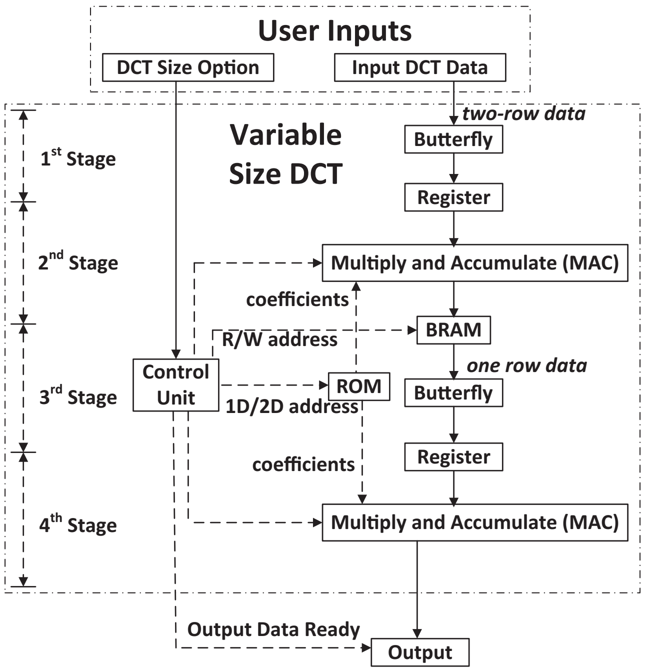


Fig. 2. FPGA four-stage architecture of the proposed 2D-DCT transformation.

ing multiplication coefficients tables and using a 16-bit input data matrix.

DCT size option (*i.e.,* 32 × 32, 16 × 16, 8 × 8, and 4 × 4 DCT/DST) and input DCT data are inputs of this architecture. If DCT size is less than 32 × 32, the architecture will be reconfigured to maximize throughputs and performance. For example, the control unit needs 16 clock cycles to write two rows of data in 32 × 32 DCT. While in 8 × 8 DCT mode, only 8 clock cycles are required to compute all of 8 × 8 DCT computation, and 16 clock cycles are needed to store results into BRAMs under the BRAM port width restriction, so our architecture will proceed double samples per cycle in 8 × 8 DCT to avoid waste computational resources. This reconfiguration property enables multiple transform sizes to be realized using the same architecture, thus, facilitating hardware sharing and reusing across different DCT block sizes.

Fig. 3 illustrates timing diagram of the proposed 2D-DCT architecture, where a 32 × 32 block size is chosen as an example. During the first stage, two rows of input data (64 points) feed into this architecture. Next, one level butterfly transform is executed and the results are stored in a register buffer. During the second stage, 1D-DCT computation occurs and takes 16 clock cycles. The resultant outputs (*i.e.,* 2 × 2 points) in each clock cycle are written into a BRAM. There are a total of 16 utilized BRAM blocks and 256 clock cycles needed for a 32 × 32 1D-DCT computation. During the next stage, 32 pixels are read from 16 BRAMs, which indicates 2 pixels per BRAM, to compute one level butterfly transform. The outputs are stored into a register. The last stage is responsible for the second 1D-DCT computation. Overall, this architecture takes 500 clock cycles to accomplish a 32 × 32 2D-DCT.

Fig. 4 shows architecture details of the first 1D-DCT stage, which proceeds two-row data in parallel. As one level butterfly transform is applied, the data of each row split into even and odd parts. For example, E00-E15 and O00-O15 are the results of each row after one level butterfly transform. These four groups of even or odd parts multiply with individual DCT coefficients, then, go through adding and rounding process. The 1D-DCT results of first-row and second-row input data are stored in the first and second columns of a BRAM, respectively. Through storing these 1D-DCT results in columns of BRAMs, no registers are needed to implement matrix transposition.

Fig. 5 shows how the proposed architecture corresponds to variable DCT sizes. Fifteen adders are organized as a tree structure. As outlined by the dash lines, a 4 × 4 DCT is embedded within an 8 × 8 DCT, which in turn is embedded in a 16 × 16 DCT and so on until a 32 × 32 DCT. Specially, in the HEVC standard, 4 × 4 blocks are transformed as either 4 × 4 DCT or 4 × 4 DST. The proposed architecture handles this 4 × 4 case by choosing different transform coefficient table. Based on the input DCT size option, this architecture in Fig. 5 adjusts its control unit and concurrently proceeds eight 4 × 4 1D-DCTs, four 8 × 8 1D-DCTs, two 16 × 16 1D-DCTs or one 32 × 32 1D-DCT. Note the multiplication coefficients are also updated, when the DCT size option varies with time. The proposed architecture can be easily modified for 2D-IDCT, where the input data are 16 bits and IDCT transform coefficient tables are used.

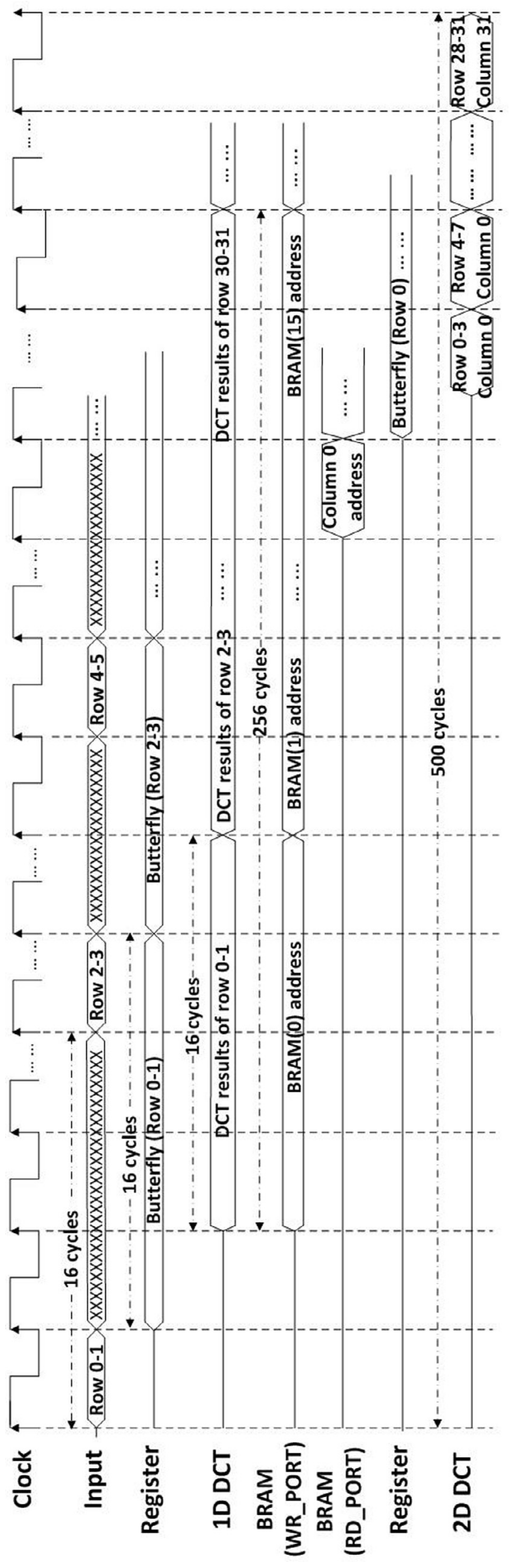


Fig. 3. Timing diagram of the proposed 32 × 32 2D-DCT architecture.

4. System implementation results and discussion

The proposed DCT architecture has been described in System Verilog. Synthesis has been conducted in various FPGA platforms, including Altera Stratix III, Cyclone II and Arria II GX, as well as Xilinx XC7VX330T and Zynq. The RTL compiler in [20] is no longer supported by FPGA vendor. To make a fair comparison with [20], our proposed design is synthesized in Cyclone II, which is the found oldest FPGA that RTL compiler still supports. DCT transform coefficients are stored in distributed RAMs or ROMs inside FPGA platforms. This proposed 2D-DCT architecture is generic and it is not limited to any fabrication technology. So the proposed

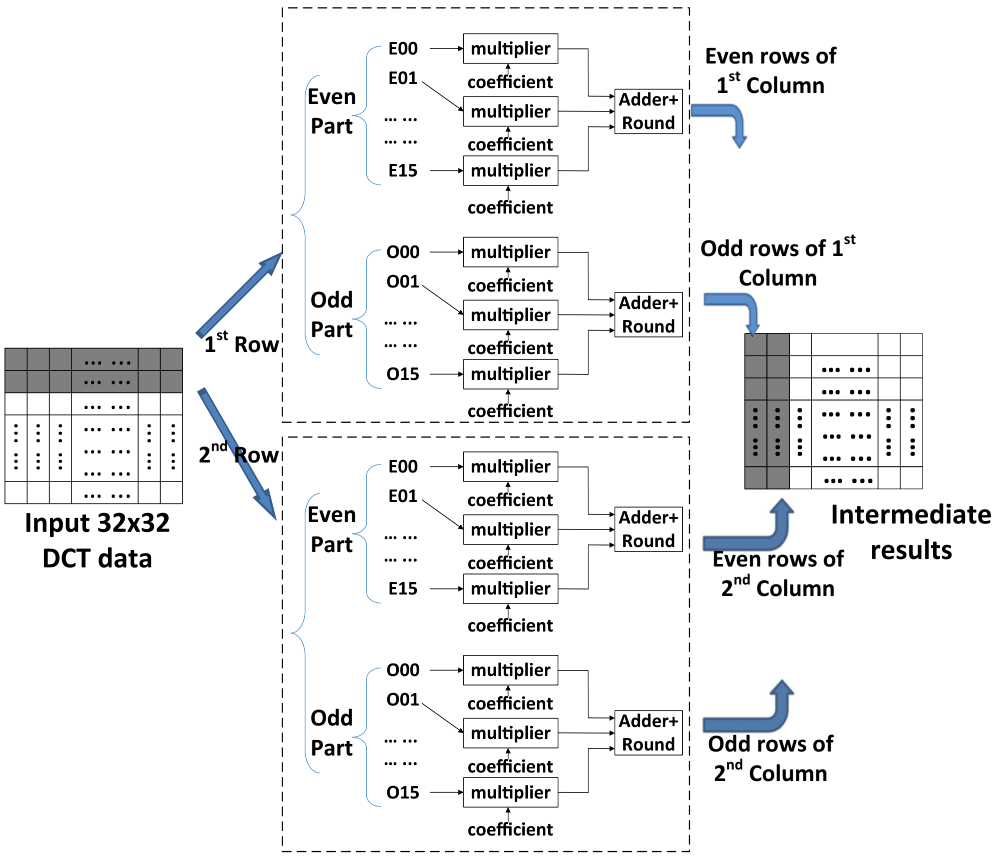


Fig. 4. First 1D-DCT architecture with BRAM data storage allocation.

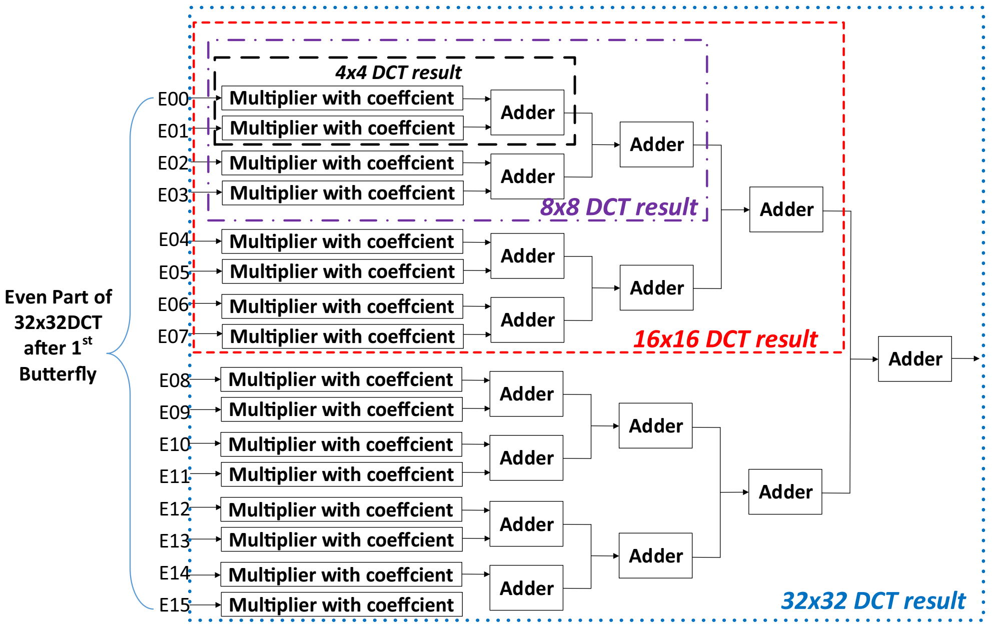


Fig. 5. Reconfiguration of DCT for variable block size.

architecture of 2D-DCT in this work is applicable to different technology or process nodes. It is well known that an advanced fabrication technology usually leads to shorter propagation delay and a higher clock frequency, but a constant number of 500 clock cycles is required to accomplish a 32×32 2D-DCT.

The related references in literature are included in Table 2, which lists key performance metrics of 2D-DCT architectures, such as FPGA model name, utilization of LUTs/ALMs and DSP blocks, the number of required clock cycle, clock frequency, and output throughput. Our proposed architecture enables variable block size

Table 2
FPGA Performance Results Summary.

	[18]	This work		[20]	This work	
FPGA	Stratix III			Altera Flex10K100	Cyclone II	
Supported DCT Size	16 × 16	16 × 16	4 × 4, 8 × 8, 16 × 16, 32 × 32	8 × 8	8 × 8	4 × 4, 8 × 8, 16 × 16, 32 × 32
# of LUT/ALM	16 K	1.4 K	5.2 K	2.5 K	2.5 K	10.4 K
# of DSP Block	0	16	32	0	64	128
# of Clock Cycle	18	60	500	128	7	500
Frequency (MHz)	27	283	206	10	116	94
Throughput (Mega Pixels/s)	204	577	421	5.53	237	192
	[21]	This work		[24]	This work	
FPGA	Xilinx XC7VX330T			Arria II GX		Xilinx Zynq
Supported DCT Size	8×8	8×8	4 × 4, 8 × 8, 16 × 16, 32 × 32	4 × 4, 8 × 8, 16 × 16, 32 × 32	4 × 4, 8 × 8, 16 × 16, 32 × 32	4 × 4, 8 × 8, 16 × 16, 32 × 32
# of LUT/ALM	3.1 K	1.7 K	5.6 K	7.3 K	5.0 K	5.8 K
# of DSP Block	0	64	128	128	32	128
# of Clock Cycle	15	7	500	N/A	500	500
Frequency (MHz)	256	239	177	200	138	222
Throughput (Mega Pixels/sec)	13	488	361	N/A	282	453

Table 3
Hardware Resource Results for Variable Size of 2D-DCT Computation.

DCT Block Size				[24] (ALM/Frequency)	This work (ALM or LUT/Frequency)		Performance Comparison at Arria II GX FPGA	
4 × 4	8 × 8	16 × 16	32 × 32	Arria II GX	Arria II GX	Xilinx Zynq	ALM number reduction	Clock freq. degradation
×	×	×	×	7269/200 MHz	5034/138 MHz	5806/222 MHz	31%	31%
	×	×	×	6928/200 MHz	4108/138 MHz	5726/222 MHz	41%	31%
		×	×	6821/200 MHz	3424/143 MHz	4733/225 MHz	50%	29%
			×	6792/200 MHz	2967/150 MHz	3898/237 MHz	56%	25%
×	×	×		5014/200 MHz	2586/179 MHz	3155/261 MHz	48%	11%
×	×			3436/200 MHz	2097/206 MHz	2478/289 MHz	39%	−3%
	×	×		4921/200 MHz	1781/185 MHz	2745/263 MHz	64%	8%

(4 × 4, 8 × 8, 16 × 16 and 32 × 32), while the references [18,20,21] only support one smaller block size (8 × 8 or 16 × 16). The references [18,20,21] do not utilize on-chip DSP blocks. Due to the use of on-chip DSPs, our proposed work results in much shorter critical path and significant improvement in terms of frequency and throughput. Due to [20] and this proposed work utilize different FPGA platforms (as well as different fabrication technology), the number of required clock cycle should be compared instead of frequency. Under the same supported DCT size (8 × 8), both works utilize the same amount of hardware resources (2.5 K LUT/ALM). However, the required number of clock cycle in this work is 7, which is equivalent to only 5.5% of that in [20]. Therefore, our proposed work is advantageous than [20]. In the world, Xilinx and Altera are two dominant FPGA companies. Their FPGA platforms are distinctive in terms of design tools, reconfigurable logic cells and system architecture. As a result, there is no intention to compare these key performance metrics of the proposed design between Arria II GX and Xilinx Zynq. The purpose of including Xilinx Zynq results in Tables 2 and 3 is to demonstrate that our proposed design is applicable to both primary FPGA manufacturers. The results in Tables 2 and 3 indicate that our proposed idea is generic and it is not limited to any specific FPGA company. Both the reference [24] and this work support variable DCT sizes. Using the same FPGA platform, our proposed architecture saves LUT resources by 32% and DSP blocks by 75%. As a result, our proposed architecture excels in hardware cost, while the clock frequency is a little degraded than [24]. Alternatively, if Xilinx Zynq is the implemented FPGA platform, our proposed architecture operates at a clock rate of 222 MHz and achieves a throughput of 453 M pixels/second. Note the required number of clock cycle in our proposed design is the same in both Arria II and Xilinx Zynq platforms.

Power Consumption is another important factor to compare among these designs. Yet, because there are no power consumption values available in references [18,21], we cannot include power consumption for a quantitative comparison in Table 2. Alternatively, we offer a comprehensive analysis as below. According to the statement of the primary FPGA Company Altera [27], the DSP blocks in modern FPGA platforms are very power efficient. These power-efficient DSP blocks enable the use of modern FPGA platforms in high definition video coding applications (e.g., our targeted application - HEVC encoder) [27]. Moreover, as reported in [28], the DSP blocks only account for 1% of total dynamic power consumption in Stratix III devices. 70% of power consumption are from user logic and signal routing. Therefore, the number of user logic (i.e., LUT/ALM) is a good indicator of energy consumption estimation. Table 2 exhibits that our proposed design consists of much less user logic than [18,21,24] under the same FPGA platform. Under different FPGA platforms, the number of user logic in the proposed work keeps the same as that in [20], but the required clock cycle is reduced largely. Hence, it is highly possible that our proposed design results in lower energy consumption due to significant savings in user logic and signal routing. Even though our proposed design uses 16 or 64 more DSP blocks than [18] or [21], the power consumption resulted from DSP blocks is negligible compared with that from user logic and signal routing. In all, considering the above two reasons, we estimate the use of DSP blocks in modern FPGA platforms probably does not lead to significant power consumption. Our proposed design is acceptable in handheld consumer electronics from an energy consumption point of view.

Moreover, we focus on Altera “Arria II GX” and Xilinx “Zynq” to thoroughly discuss system performance. Arria II GX and Zynq are based on eight-input adaptive-logic-modules (ALM8) and six-input look-up tables (LUT6), respectively. Table 3 summaries hardware resource results for variable size of 2D-DCT computation. Note even though Arria II and Zynq are implemented with 40 nm or 28 nm process respectively, the required number of clock cycle in our proposed design is the same (*i.e.,* 500 for a 32×32 2D-DCT in Table 2). Our proposed architecture in Zynq requires at least 15% more hardware resources than Arria II GX. Our proposed architecture demonstrates a higher clock frequency (*i.e.,* 222–289 MHz) at Xilinx Zynq, which is 30–38% faster than the system implementations at Arria II GX (*i.e.,* 138-206 MHz). This is because our architecture is designed inherently to best fit characteristics of Zynq platform, where the distributed RAMs/ROMs helps to improve operation speed and reduces the amount of LUTs for logic synthesis. Table 3 also demonstrate the benefit of this work over the reference [24]. Using the same FPGA platform, this work achieves 31–64% reduction in the number of ALMs, while the clock frequency overhead is no more than 31%.

Let us take a look at a 4 K@30fps UHD TV video encoding application. The minimum throughput to accomplish a 32×32 DCT is calculated as $3840 \times 2160 \times 30/(32 \times 32) = 243{,}000$ blocks/second. Since our proposed 2D-DCT architecture needs 500 clock cycles to complete one 32×32 block, therefore, our proposed architecture demands $243{,}000 \times 500 = 121.5$ million cycles/second, which is equivalent to a clock frequency of 121.5 MHz. As shown in Table 2, no matter the FPGA platform is Arria II GX or Xilinx Zynq, the clock rate of our synthesized solution reaches at least 138 MHz. This number indicates our proposed architecture is able to sustain 4 K@30 fps UHD TV real-time encoding applications, meanwhile achieving lower hardware cost.

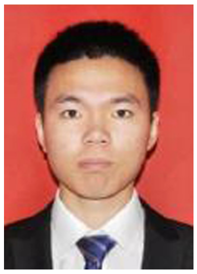
5. Conclusion

This paper presents a FPGA-friendly architecture design of variable size 2D-DCT for HEVC standard. 4×4, 8×8, 16×16 and 32×32 sizes of 2D-DCT are embedded in one architecture. This property enables multiple DCT sizes to share and reuse hardware resources. The proposed methodology efficiently proceeds 2D-DCT computation to fit internal components and characteristics of FPGA platforms. Details of circuit architecture and timing diagram are described in this work. The proposed architecture has been implemented in several FPGA platforms. Synthesis and simulation results demonstrate that the proposed architecture has great advantages in hardware cost, operating frequency and throughput, in contrast with prior works in literature. The proposed architecture is able to sustain 4 K@30 fps UHD TV real-time encoding applications with a reduction of 31–64% in hardware cost.

References

[1] Meuel H, Munderloh M, Ostermann J. Stereo mosaicking and 3D video for singleview HDTV aerial sequences using a low bit rate ROI coding framework. In: International conference on advanced video and signal based surveillance. p. 1–6.
[2] Bhaskaranand M, Gibson J. Low-complexity video encoding for UAV reconnaissance and surveillance. In: Military communications conference. p. 1633–8.
[3] Bhaskaranand M, Gibson J. Low complexity video encoding and high complexity decoding for UAV reconnaissance and surveillance. In: International symposium on multimedia. p. 163–70.
[4] Zhang Q, Chang H, Huang X, Huang L, Su R, Gan Y. Adaptive early termination mode decision for 3D-HEVC using inter-view and spatio-temporal correlations. Int J Electron Commun 2016;70(5):727–37.
[5] Bossen F, Bross B, Suhring K, Flynn D. HEVC complexity and implementation analysis. IEEE Trans Circuits Syst Video Technol December 2012;22(12):1685–96.
[6] Kalali E, Ozcan E, Yalcinkaya O, Hamzaoglu I. A low energy HEVC inverse transform hardware. IEEE Trans Consum Electron 2014;60(4):754–61.
[7] Kessentini A, Samet A, Ayed M, Masmoudi N. Performance analysis of inter-layer prediction module for H.264/SVC. Int J Electron Commun 2015;69(1):344–50.
[8] Samcovic A. Mathematical modeling of coding gain and rate-distortion function in multihypothesis motion compensation for video signals. Int J Electron Commun 2015;69(2):487–91.
[9] Budagavi M, Fuldseth A, Bjontegaard G, Sze V, Sadafale M. Core transform design in the high efficiency video coding (HEVC) standard. IEEE J Sel Top Signal Process 2013;7(6):1029–41.
[10] Rao KR, Yip P. Discrete cosine transform: algorithms, advantages, applications. Academic Press Inc; 1990.
[11] Tikekar M, Huang C, Sze V, Chandrakasan A. Energy and area efficient hardware implementation of HEVC inverse transform and dequantization. In: IEEE international conference on image processing (ICIP). p. 2100–4.
[12] JVET document, Joint video exploration team of ITU-T SG 16 WP 3 and ISP/IEC JTC 1/SC 29/WG 11 meeting.
[13] Meher P, Park S, Mohanty B, Lim K, Yeo C. Efficient integer DCT architectures for HEVC. IEEE Trans Circuits Syst Video Technol 2014;24(1):168–78.
[14] Zhu J, Liu Z, Wang D. Fully pipelined DCT/IDCT/Hadamard unified transform architecture for HEVC codec. In: IEEE international symposium on circuits and systems. p. 677–80.
[15] Budagavi M, Sze V. Unified forward+inverse transform architecture for HEVC. In: IEEE international conference on image processing. p. 209–12.
[16] Abeydeera M, Karunaratne M, Karunaratne G, Silva KD, Pasqual A. 4K real-time HEVC decoder on an FPGA. IEEE Trans Circuits Syst Video Technol January 2016;26(1):236–49.
[17] Engelhardt D, Moller J, Hahlbeck J, Stabernack B. FPGA implementation of a full HD real-time HEVC main profile decoder. IEEE Trans Consum Electron August 2014;60(3):476–84.
[18] Conceicao R, Souza J, Jeske R, Zatt B, Porto M, Agostini L. Low-cost and high throughput hardware design for the HEVC 16x16 2-D DCT transform. J Integr Circuits Syst 2014;9:25–35.
[19] Huang J, Parris M, Lee J, Demara R. Scalable FPGA architecture for DCT computation using dynamic partial reconfiguration. ACM transactions on embedded computing systems 2009;9(1).
[20] Yusof Z, Suleiman I, Aspar Z. Implementation of two dimensional forward DCT and inverse DCT using FPGA. Int Conf Electr Electron Technol 2000;3:242–5.
[21] Kitsos P, Voros N, Dagiuklas T, Skodras A. A high speed FPGA implementation of the 2D DCT for ultra-high definition video coding. In: International conference on digital signal processing. p. 1–5.
[22] Scrofano R, Jang J, Prasanna V. Energy-efficient discrete cosine transform on FPGAs. Eng Reconfigurable Syst Algorithms 2003:215–21.
[23] Atitallah A, Kadionik P, Ghozzi F, Nouel P, Masmoudi N, Marchegay P. Optimization and implementation on FPGA of the DCT/IDCT algorithm. In: International conference on acoustics speech and signal processing proceedings. p. 928–31.
[24] Pastuszak G. Hardware architectures for the H.265/HEVC discrete cosine transform. IET Image Process 2015;9(6):468–77.
[25] Xilinx FPGA document, <http://www.xilinx.com/support/documentation/white_papers/wp406-DSP-Design-Productivity.pdfg>.
[26] Xilinx 7 Series DSP48E1 Slice User Guide, <http://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf>.
[27] DSP blocks in Stratix series FPGAs, <https://www.altera.com/products/fpga/features/stx-dsp-block.html>.
[28] Blasinski H, Amiel F, Ea T, Impact of different power reduction techniques at architectural level on modern FPGAs. LASCAS; 2010.

Min Chen received the B.S. degree in Computer Science and Technology from Shenzhen University, China. He has more than 16 years of industry experience in video coding software/hardware co-design, and FPGA-based video codec. Now he is a H.265/HEVC software and system design engineer of Multicoreware Inc. at St. Louis, United States. His research interests include HEVC software and hardware video encoder design, VP9 video encoder/decoder, optimizations for multi-core codec architectures.



Yuanzhi Zhang received the B.S. and M.S. degrees in Electronic Engineering from Shandong University, China in 2011 and 2014, respectively. He is working towards his Ph.D. degree at Southern Illinois University Carbondale, IL, United States since 2015 August. His research interests include HEVC/H.265 video/image processing circuit and system optimization, low power SRAM VLSI design and methodology, and 3D-IC system design.

Chao Lu received the B.S. degree in electrical engineering from the Nankai University, Tianjin, China in 2004 and the M.S. degree in the Department of Electronic and Computer Engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2007. He obtained his Ph.D. degree at Purdue University, West Lafayette, Indiana, in 2012. From 2013 to 2015, He worked as a R&D circuit design engineer at Arctic Sand Technologies Inc. and Tezzaron Semiconductors. Now he works as an assistant professor in Electrical and Computer Engineering Department of Southern Illinois University Carbondale. His research interests include design of micro-scale energy harvesting systems, HEVC/H.265 video/image processor, power efficient memory design, and power management IC design for ultra-low power applications. Mr. Lu was the recipient of the Best Paper Award of the International Symposium on Low Power Electronics and Design (2007).