VLSI IMPLEMENTATION OF INTEGER DCT ARCHITECTURES FOR HEVC IN FPGA TECHNOLOGY

Kota Gopal1 ,K Shekar2
1: Asst Professor, Department of ECE, Kshatriya College of Engineering, Armoor, TS, India
2: Student OUCE, OU, TS, India

Abstract

High Efficiency Video Coding (HEVC) inverse transform for residual coding uses 2-D 4x4 to 32x32 transforms with higher precision as compared to H.264/AVC's 4x4 and 8x8 transforms resulting in an increased hardware complexity. In this paper, an energy and areaefficient VLSI architecture of an HEVC-compliant inverse transform and dequantization engine is presented. We implement a pipelining scheme to process all transform sizes at a minimum throughput of 2 pixel/cycle with zero-column skipping for improved throughput. We use datagating in the 1-D Inverse Discrete Cosine Transform engine to improve energy-efficiency for smaller transform sizes. A high-density SRAM-based transpose memory is used for an area-efficient design. This design supports decoding of 4K Ultra-HD (3840x2160) video at 30 frame/sec. The inverse transform engine takes 98.1 kgate logic, 16.4 kbit SRAM and 10.82 pJ/pixel while the dequantization engine takes 27.7 kgate logic, 8.2 kbit SRAM and 1.10 pJ/pixel in 40 nm CMOS technology. Although larger transforms require more computation per coefficient, they typically contain a smaller proportion of non-zero coefficients. Due to this trade-off, larger transforms can be more energy efficient

Keywords- HEVC, Inverse Discrete Cosine Transform, Transpose Memory, Data Gating.

Introduction

The High Efficiency Video Coding (HEVC) standard is the most recent joint video project of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) standardization organizations, working together in a partnership known as the Joint Collaborative Team on Video Coding (JCTVC) [1]. The first edition of the HEVC standard is expected to be finalized in January 2013, resulting in an aligned text that will be published by both ITU-T and ISO/IEC. Additional work is planned to extend the standard to support several additional application scenarios, including extended-range uses with enhanced precision and color format support, scalable video coding, and 3-D/stereo/multiview video coding. In ISO/IEC, the HEVC standard will become MPEG-H Part 2 (ISO/IEC 23008-2) and in ITU-T it is likely to become ITU-T Recommendation H.265. Video coding standards have evolved primarily through the development of the well-known ITUT and ISO/IEC standards. The ITU-T produced H.261 [2] and H.263 [3], ISO/IEC produced MPEG-1 [4] and MPEG-4 Visual [5], and the two organizations jointly produced the H.262/MPEG-2 Video [6] and H.264/MPEG-4 Advanced Video Coding (AVC) [7] standards.

The two standards that were jointly produced have had a particularly strong impact and have found their way into a wide variety of products that are increasingly prevalent in our daily lives. Throughout thisevolution, continued efforts have been made to maximize compression capability and improve other characteristics such as data loss robustness, while considering the computational resources that were practical for use in products at the time of anticipated deployment of each standard.

The Discrete cosine transform (DCT) plays a vital role in video compression due to its nearoptimal de correlation efficiency [1]. Several variations of integer DCT have been suggested in the last two decades to reduce the computational complexity. The new H.265/High Efficiency Video Coding (HEVC) standard has been recently finalized and poised to replace H.264/AVC [8]. Some hardware architectures for the integer DCT for HEVC have also been proposed for its realtime implementation. Ahmed et al. [9] decomposed the DCT matrices into sparse submatrices where the multiplications are avoided by using the lifting scheme. Shenet al used the multiplier less multiple constant multiplication (MCM) approach for four-point and eight-point DCT, and have used the normal multipliers with sharing techniques for 16 and 32-point DCTs. Park et al. [11] have used Chen's factorization of DCT where the butterfly operation has been implemented by the processing element with only shifters, adders, and multiplexors. Budagavi and Sze [12] proposed a unified structure to be used for forward as well as inverse transform after the matrix decomposition One key feature of HEVC is that it supports DCT of different sizes such as 4, 8, 16, and 32. Therefore, the hardware architecture should be flexible enough for the computation of DCT of any of these lengths. The existing designs for conventional DCT based on constant matrix multiplication (CMM) and MCM can provide optimal solutions for the computation of any of these lengths, but they are not reusable for any length to support the same throughput processing of DCT of different transform lengths. Considering this issue, we have analyzed the possible implementations of integer DCT for HEVC in the context of resource requirement and reusability, and based on that, we have derived the proposed algorithm for hardware implementation. We have designed scalable and reusable architectures for 1-D and 2-D integer DCTs for HEVC that could be reused for any of the prescribed lengths with the same throughput of processing irrespective of transform size.

HEVC Coding Design and Feature Highlights

The HEVC standard is designed to achieve multiple goals, including coding efficiency, ease of transport system integration and data loss resilience, as well as implementability using parallel processing architectures. The following subsections briefly describe the key elements of the design by which these goals are achieved, and the typical encoder operation that would generate a valid bit stream

A. Video Coding Layer The video coding layer of HEVC employs the same hybrid approach (inter-/intrapicture prediction and 2-D transform coding) used in all video compression standards since H.261. Fig. 1 depicts the block diagram of a hybrid video encoder, which could create a bitstream conforming to the HEVC standard An encoding algorithm producing an HEVC compliant bit stream would typically proceed as follows. Each picture is split into block-shaped regions, with the exact block partitioning being conveyed to the decoder. The first picture of a video sequence (and the first picture at each clean random access point into a video sequence) is coded using only intra picture prediction (that uses some prediction of data spatially from region-toregion within the same picture, but has no dependence on other pictures). For all remaining pictures of a sequence or between random access points, inter picture temporally predictive coding modes are typically used for most blocks. The encoding process for inter picture prediction consists of choosing motion data comprising the selected reference picture and motion vector (MV) to be applied for predicting the samples of each block. The encoder

and decoder generate identical inter picture prediction signals by applying motion compensation (MC) using the MV and mode decision data,

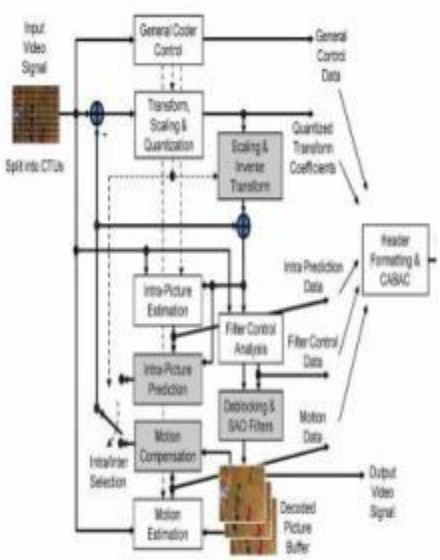

Fig. 1. Typical HEVC video encoder (with decoder modeling elements shaded in light gray).

Algorithm for Hardware Implementation of Integer DCT for HEVC: In the Joint Collaborative Team-Video Coding (JCT-VC), which manages the standardization of HEVC, Core Experiment 10 (CE10) studied the design of core transforms over several meeting cycles. The eventual HEVC transform design involves coefficients of 8- bit size, but does not allow full factorization unlike other competing proposals. It however allows for both matrix multiplication and partial butterfly implementation. In this section, we have used the partial-butterfly algorithm of for the computation of integer DCT along with its efficient algorithmic A. Key Features of Integer DCT for HEVC The N-point integer DCT 1 for HEVC given by [14] can be computed by a partial butterfly approach using a (N/2)-point DCT and a matrix–vector product of (N/2)×(N/2) matrix with an (N/2)-point vector as

$$\begin{bmatrix} y(0) \\ y(2) \\ \cdot \\ \cdot \\ \cdot \\ y(N-4) \\ y(N-2) \end{bmatrix} = \mathbf{C}_{N/2} \begin{bmatrix} a(0) \\ a(1) \\ \cdot \\ \cdot \\ \cdot \\ a(N/2-2) \\ a(N/2-1) \end{bmatrix}$$

And

$$\begin{bmatrix} y(1) \\ y(3) \\ \cdot \\ \cdot \\ \cdot \\ y(N-3) \\ y(N-1) \end{bmatrix} = \mathbf{M}_{N/2} \begin{bmatrix} b(0) \\ b(1) \\ \cdot \\ \cdot \\ \cdot \\ b(N/2-2) \\ b(N/2-1) \end{bmatrix}$$

where

$$a(i) = x(i) + x(N-i-1)$$
$$b(i) = x(i) - x(N-i-1)$$

for i=0,1,⋯,N/2−1.X=*x(0),x(1),⋯,x(N−1)+ is the input vector andY=*y(0),y(1),⋯,y(N−1)+ isN-point DCT of X. CN/2 is (N/2)-point integer DCT kernel matrix of size (N/2)×(N/2).MN/2 is also a matrix of size (N/2)×(N/2) and its (i, j)th entry is defined as

B Hardware Oriented Algorithm

Direct implementation of (1) requiresN2 /4 + MULN/2 multiplications,N2 /4+N/2 + ADDN/2 additions, and 2 shifts where MULN/2and ADDN/2are the number of multiplications and additions/subtractions of (N/2)-point DCT, respectively. Computation of (1) could be treated as a CMM problem [15]–[17]. Since the absolute values of the coefficients in all the rows and columns of matrix Min (1b) are identical, the CMM problem can be implemented as a set ofN/2 MCMs that will result in a highly regular architecture and will have low-complexity implementation. The kernel matrices for four-, eight-, 16-, and 32- point integer DCT for HEVC are given in [14], and 4- and eightpoint integer DCT

are represented, respectively, as Based on (1) and (2), hardware oriented algorithms for DCT computation can be derived in three stages as in Table I. For 8-, 16-, and 32- point DCT, even indexed coefficients of *y(0),y(2),y(4),⋯y(N−2)+ are computed as 4-, 8-, and 16-point DCTs of *a(0),a(1),a(2),⋯a(N/2−1)+,respectively, according to (1a). In Table II, we have listed the arithmetic complexities of the reference algorithm and the MCM-based algorithm for four-, eight-, 16-, and 32-point DCT. Algorithms for Inverse DCT (IDCT) can also be derived in a similar way.

Proposed Architectures for Integer DCT Computation A. Proposed Architecture for Four-Point Integer DCT: The proposed architecture for four-point integer DCT is shown in Fig. 1(a). It consists of an input adder unit (IAU), a shift-add unit (SAU), and an output adder unit (OAU) The IAU computesa(0),a(1),b(0), andb(1) according to STAGE-1 of the algorithm as described in Table I. The computations of ti,36 and ti,83 are performed by two SAUs according to STAGE-2 of the algorithm The computation oft0,64 andt1,64 does not consume any logic since the shift operations could be rewired in hardware. The structure of SAU is shown in Fig. 1(b). Outputs of the SAU are finally added by the OAU according to STAGE-3 of the algorithm B. Proposed Architecture for Integer DCT of Length8and Higher Length DCTs: The generalized architecture for N-point integer DCT based on the proposed algorithm is shown in Fig. 2. It consists of four units, namely the IAU, (N/2)-point integer DCT unit, SAU, and OAU. The IAU computesa(i) and b(i) for i =0,1, ..., N/2−1 according to STAGE-1 of the algorithm of Section II-B. The SAU provides the result of multiplication of input sample with DCT coefficient by STAGE-2 of the algorithm.

Finally, the OAU generates the output of DCT from a binary adder tree of log 2N−1 stages. Fig. 3(a)–(c), respectively, illustrates the structures of IAU, SAU, and OAU in the case of eight-point integer DCT. Four SAUs are required to compute ti,89, ti,75, ti,50, and ti,18 for i =0,1,2,and3 according to STAGE-2 of the algorithm. The outputs of SAUs are finally added by two-stage adder tree according to STAGE-3 of the algorithm. Structures for 16- and 32-point integer DCT can also be obtained similarly. RESULTS AND DISCUSSIONS A. Synthesis Results of1-D Integer DCT We have coded the architecture derived from the reference algorithm of as well as the proposed architectures for different transform lengths in VHDL, and synthesized by Synopsys Design Compiler using TSMC 90-nm General Purpose (GP) CMOS Library. The word length of input samples are chosen to be 16 bits.

The area, computation time, and power consumption (at 100-MHz clock frequency). It is found that the proposed architecture involves nearly 14% less areadelay product (ADP) and 19% less energy per sample (EPS) compared to the direct implementation of reference algorithm, in average, for integer DCT of lengths 4, 8, 16, and 32. Additional 19% saving in ADP and 20% saving in EPS are also achieved by the pruning scheme with nearly the same throughput rate. The pruning scheme is more effective for higher length DCT since the percentage of total area occupied by the SAUs increases as DCT length increases, and hence more adders are affected by the pruning scheme. A. Comparison With the Existing Architectures We have named the proposed reusable integer DCT architecture before applying pruning as reusable architecture-1 and that after applying pruning as reusable architecture-2. The processing rate of the proposed integer DCT unit is 16 pixels per cycle considering 2-D folded structure since 2-D transform of 32×32 block can be obtained in 64 cycles.

In order to support 8Kultrahigh definition (UHD) (7680×4320) at 30 frames/s and 4:2:0 YUV format that is one of the applications of HEVC [20], the proposed reusable architectures should work at the operating frequency faster than 94 MHz (7680×4320×30×1.5/16). The computation times of 5.56 ns and 5.27 ns for reusable architectures- 1 and 2, respectively (obtained from the synthesis without any timing constraint) are enough for this application. Also, the computation time less than 5.358 ns is needed to support 8K UHD at 60 frames/s, which can be achieved by slight increasein silicon area when we synthesize the reusable architecture-1 with the desired timing constraint. Existing architectures for HEVC forN= 32 in erms of gate count that is normalized by area of 2-inputNANDgate, maximum operating frequency, processing rate, throughput, and supporting video format. The proposed reusable architecture-2 requires larger area but offers much higher throughput. Also, the proposed architectures involve less gate counts, as well as higher throughput, C. Synthesis Results of 2-D Integer DCT We also synthesized the the folded and fullparallel structures for 2-D integer DCT. We have listed total gate counts, processing rate, throughput, power consumption, andEPS in Table VII. We set the operational frequency to 187 MHz for both cases to support UHD at 60 frames/s. The 2-D fullparallel structure yields 32 samples in each cycle after initial latency of 32 cycles providing double the throughput of the folded structure. However, the full-parallel architecture consumes 1.69 times more power than the folded architecture since it has two 1-D DCT units and nearly the same complexity of transposition buffer while the throughput of full-parallel design is double the throughput of folded design. Thus, the fullparallel design involves 15.6% less EPS

Conclusions

In this paper, we presented a very lowcomplexity DCT approximation obtained via pruning. The resulting approximate transform requires only 10 additions and possesses performance metrics comparable with state-ofthe-art methods, including the recent architecture presented in [24]. By means of computational Volume 2, Issue 5 JUNE 2015 IJOEET simulation, VLSI hardware realizations, and a full HECV implementation, we demonstrated the practical relevance of our method as an image and video codec

REFERENCES

[1] B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, High Efficiency Video Coding (HEVC) Text Specification Draft 9, document JCTVC-K1003, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCTVC), Oct. 2012.

[2] Video Codec for Audiovisual Services at px64 kbit/s, ITU-T Rec. H.261, version 1: Nov. 1990, version 2: Mar. 1993.

[3] Video Coding for Low Bit Rate Communication, ITU-T Rec. H.263, Nov. 1995 (and subsequent editions). [4] Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/s—Part 2: Video, ISO/IEC 11172-2 (MPEG-1), ISO/IEC JTC 1, 1993.

[5] Coding of Audio-Visual Objects—Part 2: Visual, ISO/IEC 14496-2 (MPEG-4 Visual version 1), ISO/IEC JTC 1, Apr. 1999 (and subsequent editions).

[6] Generic Coding of Moving Pictures and Associated Audio Information— Part 2: Video, ITU-T Rec. H.262 and ISO/IEC 13818- 2 (MPEG 2 Video), ITU-T and ISO/IEC JTC 1, Nov. 1994.

[7] Advanced Video Coding for Generic AudioVisual Services, ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC), ITU-T and ISO/IEC JTC 1, May2003 (and subsequent editions).

[8] H. Samet, "The quadtree and related hierarchical data structures," Comput. Survey, vol. 16, no. 2, pp. 187–260, Jun. 1984.

[9] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard,"IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003. [10] S. Wenger, "H.264/AVC over IP," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 645–656, Jul. 2003.

[11] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments,"IEEE Trans. Circuits Syst