AN AREA REDUCTION SCHEME FOR THE H.264/AVC CAVLC ENCODER

Choudhury A. Rahman and Wael Badawy
Laboratory for Integrated Video Systems (LIVS), University of Calgary
Calgary, Alberta, Canada T2N 1N4
{carahman, badawy}@ucalgary.ca

ABSTRACT

This paper presents an approach to reduce the area used for the implementation of look up tables (LUTs) in the H.264/AVC Context-based Adaptive Variable Length Coding (CAVLC) encoder. Replacing the LUTs of the *coeff_token* by an algorithm that closely follows the codeword features and can be easily implemented with simple computational elements, the proposed scheme shows that an area savings of more than 40% can be achieved with a bit-rate increase of less than 0.8%. The proposed approach is very effective to reduce the area for low bit-rate applications such as mobile video applications which have simple/low to moderate motion characteristics.

Index Terms— Context-based Adaptive Variable Length Coding (CAVLC), Entropy coding, H.264/AVC, VLSI implementation.

1. INTRODUCTION

The video coding standard approved by ITU-T as Recommendation H.264 and ISO/IEC as International Standard 14496-10 (MPEG-4 Part 10) Advanced Video Coding (AVC) [1] has been finalized in May 2003. This new standard H.264/AVC is designed for application in the areas such as broadcast, interactive or serial storage on optical and magnetic devices such as DVDs, video-on-demand or multimedia streaming, multimedia messaging etc. over ISDN, DSL, Ethernet, LAN, wireless and mobile networks. Some new features of the standard enable enhanced coding efficiency such as motion estimation with quarter pixel accuracy, multiple reference frames and variable block sizes for inter prediction. Moreover, the intra prediction has also lifted the coding performance. The advanced entropy coding method (CAVLC and CABAC) used in the standard further improves the performance.

The CAVLC entropy coding method is used for coding quantized transform coefficients of the residual images. It is highly context adaptive that results in very good coding efficiency. There are several architectures proposed so far for the CAVLC encoder [2-4]. The main drawback of these architectures is that they all require larger storage space, so their implementation cost is higher. The requirement for larger storage space comes from the LUTs for storing VLC tables. In this paper, we propose an area savings approach by which the costly LUTs can be replaced by simple computational elements. This is done by replacing the VLC tables with an algorithm that generates the VLC codewords. The algorithm offers regularity and follows the characteristics of the codewords as closely as possible so that the tradeoff does not cost much. The experimental results show that the algorithm is very effective in reducing the area of implementation especially for low bit-rate case with very minimal increase in bit-rate.

The rest of the paper is organized as follows: Section 2 discusses the algorithm of the Context-based Adaptive Variable Length Coding (CAVLC). Section 3 presents the proposed approach to reduce the area used for the implementation of the look up tables (LUTs). Performance analysis is shown in section 4. Finally, section 5 concludes the paper.

2. THE CAVLC ENTROPY CODING

In the proposed H.264/AVC standard, two entropy coding methods are included, namely context-based adaptive variable length coding (CAVLC) [1,5] and context-based adaptive binary arithmetic coding (CABAC) [6]. In the CAVLC entropy coding, only the quantized transform coefficients (4x4 blocks of luma and 2x2 blocks of chroma) of the residual images are coded. Other information, i.e., macroblock (MB) header, is coded using Exp-Golomb codes. The transform coefficients are first scanned in a zigzag fashion. The zigzag scanning is done to produce long runs of zeros. CAVLC uses multiple VLC tables. The VLC tables are switched depending on previously coded elements. This result in improved coding efficiency compared to using a single VLC table. The CAVLC encoder encodes the following elements of a block of transform coefficients in order.

1. *coeff_token:* Encodes the number of nonzero coefficients and trailing ones (maximum 3, the rest are coded as normal coefficient). There are three

VLC tables and one FLC table to choose from for the luma block depending on the number of nonzero coefficients of the left and upper block of the current block and one VLC table for the chroma block.

2. *sign_trail:* Sign of the trailing ones (0 for positive and 1 for negative ones) in reverse order.

3. *levels:* Encodes the nonzero coefficients excluding the trailing ones in reverse order. Each level is encoded by one of the seven VLC tables depending on the magnitude of each successive encoded level.

4. *total_zeros:* Encodes the total number of zeros occurring after the first nonzero coefficient in reverse order by separate VLC tables for luma and chroma.

5. *run_before:* Encodes the number of zeros preceding each nonzero coefficient in reverse zigzag order by a VLC table.

3. THE PROPOSED APPROACH

One of the difficulties in VLSI implementation of the CAVLC coder is the VLC tables. Traditionally, the VLC tables are implemented with memory (LUT) because the VLC codes do not follow any mathematical relationships. Since a lot of VLC tables are used for CAVLC coding, the required storage space for hardware implementation is large. In this paper, we propose an area savings approach by replacing the VLC tables in a way so that they can be implemented with simple computational elements such as adders, subtractors and MUXs and thereby eliminating the requirement of huge memory. The tradeoff of the approach is the bit-rate. Nevertheless, the experimental results show that the proposed approach is very effective for low bit-rate applications such as mobile video applications where, an area savings of more than 40% can be achieved with a bit-rate increase of less than 0.8%.

The implementation of the *coeff_token* VLC and FLC tables are shown in this paper. There are three VLC tables and one FLC table to choose from for the luma blocks (table I) and the choice depends on the number of total coefficients in the left block (N_L) and the upper block (N_U) of the current block. A parameter N is calculated for this purpose following the pseudo code shown in Fig. 1. For chroma blocks, there is only one VLC table.

The VLC tables are biased from small to large number of total nonzero coefficients in a block so that *coeff_token* can be coded with smallest possible codeword. The FLC is a 6 bit code. The least significant 2 bit represents the trailing ones and the rest represents the total number of nonzero coefficient in the current block.

Table I
Choice of the *coeff_token* table

Value of N	Table
$N < 2$	VLC 1
$2 \le N < 4$	VLC 2
$4 \le N < 8$	VLC 3
$N \ge 8$	FLC

if *upper block* and *left block* is available

$$N = \left(\frac{N_L + N_U}{2} \right)$$

else if *upper block* is available

$$N = N_U$$

else if *left block* is available

$$N = N_L$$

else

$$N = 0$$

Fig.1. Pseudo code for calculating N from N_L and N_U.

Table	Pseudo Code
VLC 1 / Chroma DC	*if ((TotalCoeff - Trailing Ones) == 0){* *if (Trailing Ones < 2){* *Code = 1;* *Length = 1 + Trailing Ones; }* *else{* *Code = Trailing Ones;* *Length = 4; }* *}else{* *Code = 4 + Trailing Ones;* *Length = (TotalCoeff - Trailing Ones) + 5; }*
VLC 2	*if ((TotalCoeff - Trailing Ones) == 0){* *if (Trailing Ones < 2){* *Code = 2 + Trailing Ones;* *Length = 2; }* *else{* *Code = Trailing Ones;* *Length = 3; }* *}else{* *Code = 4 + Trailing Ones;* *Length = (TotalCoeff - Trailing Ones) + 4; }*
VLC 3	*if ((TotalCoeff - Trailing Ones) < 2){* *if (TotalCoeff == Trailing Ones){* *Code = 8 + Trailing Ones;* *Length = 4; }* *else{* *Code = 12 + Trailing Ones;* *Length = 4; }* *}else{* *Code = 4 + Trailing Ones;* *Length = (TotalCoeff - Trailing Ones) + 2; }*
FLC	*Length = 6;* *if (Trailing Ones == 0 &&* *(TotalCoeff - Trailing Ones) == 0)* *Code = 3;* *else* *Code = {(TotalCoeff– 1)[3:0], trailing_ones}*

Fig.2. Pseudo code for of the proposed algorithm for generating the VLC tables.

Fig. 2 shows the pseudo code of the proposed scheme for generating the VLC tables for the *coeff_token*. The

algorithm offers a regularity that can be easily implemented with computational elements for efficient VLSI implementation. The desired feature of the codes, i.e., shorter codewords for the most probable symbols, is also tried to maintain as closely as possible especially for low bit-rate case, i.e., small number of nonzero coefficients, in order to make the algorithm suitable for mobile video applications where area savings is relatively a big concern.

The block diagram of the proposed algorithm is shown in Fig. 3. It has 3 blocks. The Code generator generates the code for the VLC and the Length Generator generates the length of the VLC code. The Controller block generates the control signals for the two generators.

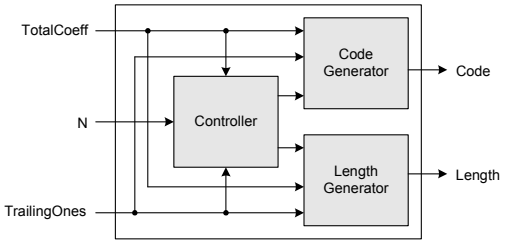

Fig.3. The block diagram of the proposed architecture.

The detail of the proposed architecture is shown in Figs.4, 5 and 6. The 'Code' generator is built with an adder, a subtractor and an output MUX controlled by the selector signal C2. The input MUX controlled by the selector signal C1, is replaced by logic gates during synthesis because of having constant inputs. On the other hand, the 'Length' generator is composed of 2 adders, 1 subtractor and an output MUX. The two MUXs (controlled by the selector signals L1 and L2), are again replaced with logic gates during synthesis for the same reason.

4. PERFORMANCE ANALYSIS

The proposed architecture for *coeff_token* is coded in Verilog HDL, simulated and synthesized by ModelSim 6.0 and Xilinx ISE development tools 6.2, respectively. The target device for synthesis was the Xilinx Virtex II FPGA (2v3000fg676-4). Table II shows the synthesis results.

Table II
Synthesis results of the proposed architecture

CLB Slices	4 input LUTs	Bonded IOBs	Gate Count
26	45	21	315

The total gate count of the architecture is 315. This gives an area savings of around 43% than the implementation reported in [2]. Table III shows the comparison.

Table III
Comparison with the proposed design with others

	Proposed	Chien *et. al.* [2]	Chen *et. al.* [3]
Gate Count	315	554	864

The JM 11.0 reference software [7] is used in the simulation for evaluating the performance of the proposed algorithm. Table IV shows the result with QCIF main profile (I-B-P-B-P) frame sequences and table V shows the result with QCIF baseline profile (I-P-P-P) frame sequences. Both the tables show that at quantization parameter (QP) value 30, the overall bit-rate increase is less than 0.8%.

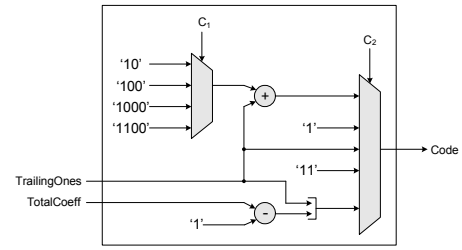

Fig.4. Architecture detail of the Code Generator block.

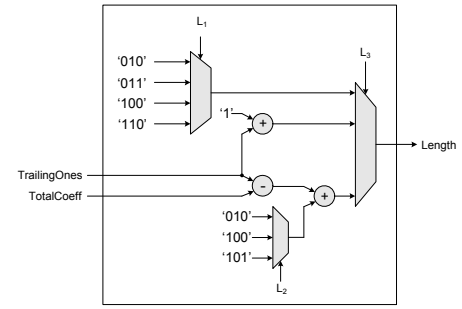

Fig.5. Architecture detail of the Length Generator block.

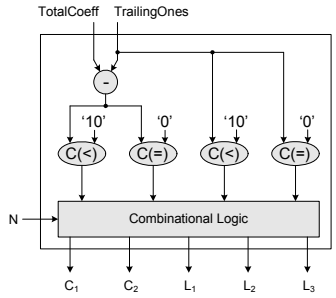

Fig.6. Architecture of the Controller unit.

Table IV
Experimental results with QCIF main profile (I-B-P-B-P) frame sequences

	Bit-rate increase (%)			
	I	P	B	Overall
Foreman	0.75	0.12	0.15	0.22
Mobile	1.59	0.06	-0.21	0.30
Carphone	0.53	0.07	0.10	0.13
Claire	0.43	0.19	0.29	0.29
Container	1.30	0.29	0.57	0.77
Hall	0.85	0.18	0.07	0.37
Miss America	0.69	0.31	-0.36	0.22
News	0.91	0.22	0.05	0.38
Salesman	0.55	0.31	-0.15	0.37
Silent	0.46	0.19	0.29	0.28

Table V
Experimental results with QCIF baseline profile (I-P-P-P) frame sequences

	Bit-rate increase (%)		
	I	P	Overall
Foreman	0.75	0.12	0.22
Mobile	1.63	-0.10	0.23
Carphone	0.53	0.23	0.29
Claire	0.44	0.47	0.45
Container	1.30	0.21	0.74
Hall	0.85	0.04	0.31
Miss America	0.68	0.06	0.21
News	0.76	0.08	0.30
Salesman	0.47	0.43	0.44
Silent	0.55	0.33	0.38

Fig. 7 shows the bit-rate increase of three different sequences (Container, Salesman and Miss America) that reported the highest, mediocre and lowest overall bit-rate increase in table V for baseline profile (I-P-P-P) frame sequences. The figure also shows that for moderate to high compression ratio, i.e., low bit-rate applications such as mobile video applications, the bit-rate increase is relatively much smaller. As a result, the area savings for such applications is much beneficial. Fig. 8 shows the plot of overall bit-rate increase of different frames sequences at QP = 30. This figure also justifies the tradeoff for typical handheld applications, i.e., phone conferencing, news broadcasting etc., which have simple/low to moderate motion characteristics.

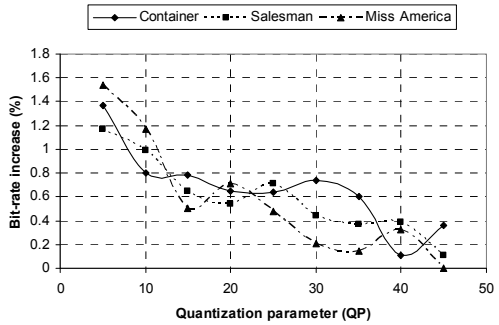

Fig. 7. Bit-rate increase of 3 QCIF baseline profile (IPPP) frame sequences at different QP values.

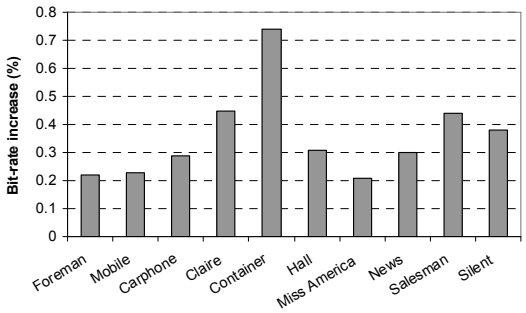

Fig. 8. Overall bit-rate increase of different frames sequences at QP = 30.

5. CONCLUSION

This paper presents an area savings approach by replacing the LUTs for the VLC tables of a CAVLC coder in a way so that they can be implemented with simple computational elements and thereby eliminating the requirement of huge memory. The proposed algorithm follows the desired feature of the codes as closely as possible especially for the low bit-rate case so that the tradeoff between bit-rate increase and area savings proves its feasibility. The experimental results show that the proposed approach is very effective for mobile video (low bit-rate) applications where, an area savings of more than 40% can be achieved with a bit-rate increase of less than 0.8%.

6. ACKNOWLEDGEMENTS

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canadian Foundation for Innovations (CFI), Micronet R&D Canada, and iCORE for supporting this research.

7. REFERENCES

[1] "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050r1, May 2003.

[2] C. D. Chien, K. P. Lu, Y. H. Shih, and J. I. Guo, "A high performance CAVLC encoder design for MPEG-4 AVC/H.264 video coding applications," in the Proc. of IEEE ISCAS 2006, pp. 3838–3841.

[3] T. C. Chen, Y. W. Huang, C. Y. Tsai, B. Y. Hsieh, and L. G. Chen, "Architecture design of context-based adaptive variable-length coding for H.264/AVC," IEEE Trans. On CAS-II, vol. 53, issue. 9, Sept. 2006, pp. 832–836

[4] Y. K. Lai, C. C. Chou, and Y. C. Chung, "A simple and cost effective video encoder with memory-reducing CAVLC," in the Proc. of IEEE ISCAS 2005, vol. 1, pp. 432–435.

[5] Iain E.G. Richardson, H.264 and MPEG-4 Video Compression, Wiley Publishers, December 2003, ISBN 0470848375.

[6] D. Marpe, H. Schwarz, and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," IEEE Trans. Circuits Syst. Video Tech., vol. 13, issue. 7, pp. 620–636, July 2003.

[7] Joint Video Team (JVT) reference software, version 11.0, http://iphome.hhi.de/suehring/tml/download/jm11.0.zip