ELectronics EXpress

IEICE Electronics Express, Vol.8, No.22, 1863–1869

A 4-way parallel CAVLC design for H.264/AVC 4 Kx2 K 60 fps encoder

Huibo Zhong, Sha Shen, Yibo Fan[a)], and Xiaoyang Zeng

State Key Lab of ASIC and System, Fudan University

825 Zhangheng Road, Shanghai, 201203, China

a) *fanyibo@fudan.edu.cn*

Abstract: This paper presents a high performance design for Context-Based Adaptive Variable Length Coding (CAVLC) used in the H.264/AVC standard. A two-stage encoder is proposed to make the scan and encode stage work simultaneously. The scan engine scans four coefficients at each cycle. Parallel encoder for four "levels" and parallel encoder for four "Run_before" are adopted to accelerate the encode engine. Only 120 cycles at most are needed to process one MB. The proposed CAVLC encoder can support 4 Kx2 K@60 fps (frame per second) real-time encoding at 250 MHz and the gate count is about 32 k.

Keywords: H.264/AVC, entropy coding, CAVLC, level coding

Classification: Integrated circuits

References

[1] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264-ISO/IEC 14496-10 AVC)," March 2003.

[2] G. Bjontegaardand and K. Lillevold, "Context-adaptive VLC (CVLC) coding of coefficients," *JVT Document JVT-C028*, Fairfax, VA, 2002.

[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. Circuits Syst. II, Exp. Briefs*, vol. 53, no. 9, pp. 832–836, Sept. 2006.

[4] Y. Yi and B. C. Song, "High-speed CAVLC encoder for 1080p 60-Hz H.264 codec," *IEEE Signal Process. Lett.*, vol. 15, pp. 891–894, 2008.

[5] S. C. Hsia and W. S. Liao, "Fowward Computations for Context-Adaptive Variable-Length Coding Design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57 no. 8, pp. 637–641, Aug. 2010.

1 Introduction

H.264/AVC [1] is the latest international video coding standard. Two kinds of entropy coding methods are specified in H.264/AVC: Context-Based Adaptive Variable Length Coding (CAVLC) and Context-based Binary Arithmetic

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011

Coding (CABAC). Due to the context-adaptive feature, CAVLC yields a higher coding efficiency compared with conventional VLC coding. However, this context-adaptive feature introduces many data dependencies such as: 1). The symbols TotalCoeff, TrailingOne, Sign_trail, Level, TotalZero and Run_before cannot be obtained until the statistical process of each 4x4 block is finished [1, 2], which makes the scan and encode stage hard to be paralleled. 2). The bit-serial encoding process for all the symbols above makes it difficult to be speeded up by increasing parallelism or pipelining. 3). To encode TotalCoeff, nC should be determined at first, however, the value of nC is calculated according to the upper and left coded 4x4 block. 4). There is seven VLC tables for the levels in each 4x4 block where the table selection depends on the previous table number and the magnitude of the successive encoded Level. Because of the data dependencies in CAVLC, it is hard to propose a CAVLC encoder for high resolution application such as 4 Kx2 K or larger.

Typically, there are two methods to improve the speed of CAVLC encoder. One is to increase the operating frequency, and the other is to reduce the cycles of processing one MB. To increase the operating frequency, we can insert more pipeline stage. A two-stage architecture is adopted by many previous works to make the scan and encode work simultaneously. The general approach to cut down the cycles is to increase the parallelism. Varieties of VLSI architectures for CAVLC are proposed in [3, 4, 5]. Chen et al. [3] adopted a two-stage stage structure, but parallelism is not implemented in their work, which makes it require at most 500 cycles to process one MB and greatly restrict the throughput. A parallel structure for Luma (Y) and Chroma (Cr and Cr) encoding is proposed in [4], however, this parallel approach introduces large hardware cost. Yi et al. [5] also employed the two-stage structure. In addition, it scanned two coefficients at one cycle by using the parallel strategy. The cycles consumed by the scan and encode engine is not balanced. It needs NC + 4 cycles to process one 4x4 block where NC is equal to the nonzero coefficients in one 4x4 block. Once the NC becomes larger due to the bit rate, the cycles using for encode engine will increase.

In this paper, we focus on increasing the hardware parallelism for CAVLC. A two-stage architecture is also proposed in this paper. A paralleled scan engine which can scan four coefficients at one cycle is proposed. In the same way, a paralleled "Level" encoder is proposed to accelerate the speed of encoder engine. Four "Run_before" symbols are encoded simultaneously with "Level". With the methods above, only four cycles are needed by each stage of the CAVLC. Only 120 cycles at most are needed to process one MB.

2 Proposed architecture

Fig. 1 shows the proposed two-stage CAVLC encoder. The whole CAVLC encoder is comprised of scan engine and encode engine. In the scan stage, the scan address is generated by the scan engine according to the value of MB type and CBP code. The residual coefficients are fetched from the resid-

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011

ual buffer in the backward order. Then the scan engine counts the required statistics such as TrailingOne, TotalCoeff, TotalZero, and so on. The run-level symbols are extracted by the run-level detector and stored in the statistic buffer. The symbols obtained by the scan engine are encoded by the engine by looking up corresponding tables. In our work, the two stage of the CAVLC is well designed to consume the same cycles for each 4x4 block. To achieve the highest throughput, a scan engine which scans four coefficients at one cycle is proposed. It takes only four cycles to finish the parameter calculation of one 4x4 block. To achieve the same speed of the scan engine, some schemes have to be adopted to speed up the encode engine. The detail of the two engines is presented in the rest of this section.

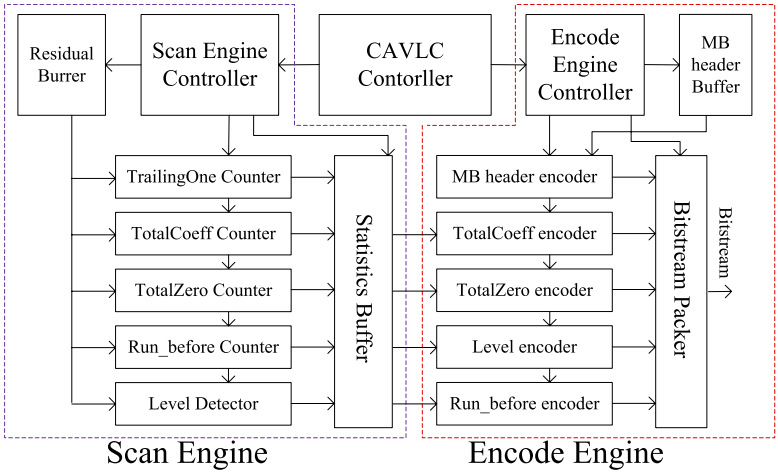

Fig. 1. Proposed CAVLC architecture

2.1 Scan engine

Scan engine is used to figure out the CAVLC parameters. There are totally 384 coefficients (for YUV = 4 : 2 : 0) inside one MB, so scan stage may become a bottleneck. A paralleled scan engine which can scan four coefficients at one cycle is proposed to raise the speed of scan engine. Firstly, eight state bits can be computed as follows:

$$\begin{cases} if\ \ coeff\_0 = 0, & s0 = 0 \\ else & s0 = 1 \\ if\ \ coeff\_1 = 0, & s1 = 0 \\ else & s1 = 1 \\ if\ \ coeff\_2 = 0, & s2 = 0 \\ else & s2 = 1 \\ if\ \ coeff\_3 = 0, & s3 = 0 \\ else & s3 = 1 \end{cases} \tag{1}$$

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011

$$\begin{cases} if\ \ coeff\_0 = \pm 1, & c0 = 1 \\ else & c0 = 0 \\ if\ \ coeff\_1 = \pm 1, & c1 = 1 \\ else & c1 = 0 \\ if\ \ coeff\_2 = \pm 1, & c2 = 1 \\ else & c2 = 0 \\ if\ \ coeff\_3 = \pm 1, & c3 = 1 \\ else & c3 = 0 \end{cases} \tag{2}$$

As shown in (1) and (2), s0∼s3 are the state bits indicating whether the coefficient equals to zero. c0∼c3 are the state bits indicating whether the coefficient equals to ±1. By the bits of these eight states, all required CAVLC symbols can be figured out. By judging the value of s0∼s3, TotalCoeff, TotalZero, Run_before can be obtained. To figure out TrailingOne, all the eight bits should be considered. To lower the design complexity, the four coefficients are divided into two parts: the low and the high. Each part is collected independently. Their results are summarized at the end. The detail of the architecture is shown in Fig. 2 (a).

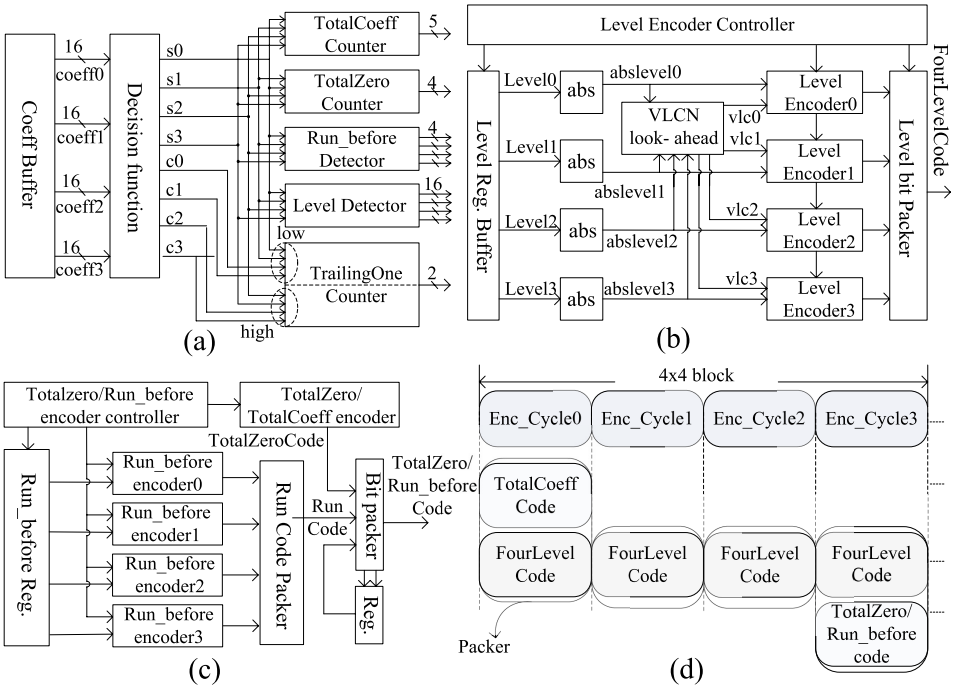

Fig. 2. Detail architecture of proposed CAVLC: (a) Parallel architecture for parameter computing. (b) Proposed parallel Level encoder. (c) Proposed parallel Run_before encoder. (d) Timing diagram of the Bitstream Packer

2.2 Encode engine

The encode engine encodes the data which are obtained by the scan engine. It is comprised of Level encoder, Run_before encoder, TotalCoeff and Trailin-

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011

gOne encoder, TotalZero encoder as shown in Fig. 1. TotalCoeff, TrailingOne and TotalZero are coded by looking up corresponding tables. The Level encoder and Run_before encoder are the key of the encode engine.

In H.264/AVC standard, seven VLC tables are used to define the codeword of Level symbols. The number of execution cycles required for encoding Level symbols depends on the number of nonzero coefficients in one 4x4 block. There may be at most 16 levels inside one 4x4 block. Therefore, the number of cycles required in coding stage may exceed that in scanning stage. Many previous works encode the Level symbols in sequential because the evaluation of each coefficient level depends on *vlc*, which denotes the index of the VLC table and is derived from the previously coded coefficient level as pseudo code (3) shows. Let $incVLC \in \{0, 3, 6, 12, 24, 48, 65535\}$. To achieve higher throughput, parallel Level encoding is necessary. The limit of data dependency can be effectively eliminated by using the vlc-look-ahead technique so that coefficients can be processed simultaneously.

$$\begin{aligned} &if\,(vlc = 0) \\ &\quad vlc++; \\ &if\,(abs(level) > incVLC[vlc]) \\ &\quad vlc++; \end{aligned} \tag{3}$$

Fig. 2 (b) shows the architecture of proposed Level coding element with VLCN look-ahead. In the proposed design, four Level symbols are fetched from the Level Register Buffer simultaneously, and four Level encoders are used for parallel encoding. Noted that the encode engine will consume the same cycles as the scan engine does if this structure is used. The controller of the Level encoder controls the four Level encoders according to the value of TotalCoeffs and TrailingOnes. VLCN look-ahead computes the four *vlc* codes for the corresponding level encoder according to the absolute value of the Level and previous *vlc*. The four Level encoders then encode the Level and generate level_prefix and level_suffix. And finally, the Level codeword packer packs all the four pairs of level_prefix and level_suffix into FourLevelCode.

In the same way, run_before encoder should also be processed parallel as the Level encoder does. Four Run_before symbols are fetched from Run_before buffer. Four Run_before codewords are produced by looking up the Run_Before&Zerosleft table. These codewords are packed with the TotalZero Code and stored in a register until all the levels are encoded. The detail of the Run_before encoder is shown in Fig. 2 (c). Then stored codeword is called TotalZero/Run_before code.

2.3 Bitstream packer

The codeword generated by the encoder should be packed into bitstream by the Bitstream Packer. As it has been noted, all the codeword should be packed within four cycles, so the timing of the Bitstream packer should be carefully designed. The detail timing diagram of the bitstream packer is illustrated in Fig. 2 (d). At Enc_cycle0 (the first cycle of the encode stage), the TotalCoeff code is packed with the first FourLevelCode. The TotalCoeff

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011

code can be obtained by looking up the VLC tables according to the value of TotalCoeff and TrailingOne. The FourLevelCode is generated as Fig. 2 (b) shows. At the following two cycles (Enc_cycle1 and Enc_cycle2), the next two FourLevelCode are packed. The TotalZero/Run_before code which is generated as Fig. 2 (c) shows is packed with the last FourLevelCode at Enc_cycle3 (the last cycle of encode stage).

3 Performance analysis and comparison

The proposed architecture has been implemented in Verilog HDL, synthesized using SMIC 0.13 um standard CMOS technology. The circuit occupies about 32 k gates at a core speed of 250 MHz. The comparison on hardware cost and processing speed of the proposed design with the existing design [3, 4, 5] is as Table I shows.

From Table I, we can see that the throughput of the proposed design is 120 cycles/MB, which is smaller than all [3, 4, 5] does. In practice, with the method using the coded block pattern (CBP) to determine whether the blocks need to be encoded or not, the average cycles will be smaller than the worst case. It takes about 90 cycles to encode one MB at average according to our experimental results. The hardware cost of our design is 32 k gate, which is larger than [3] and [5]. This is because more hardware parallelism is used in our design, such as four level encoders and four Run_before encoders.

Table I. Comparison of the proposed design with others

	Chen[3]	Yi[4]	Hsia[5]	Proposed
Technology(um)	0.18	0.09	0.18	0.13
Frequency	100M	227M	125M	250M
Gate count	23K	66K	15K	32K
speed(cycle/MB)	500	323	256	120
Max. Spec	1080P @30fps	1080P @60fps	1080P @30fps	4Kx2K @60fps
YUV format	4:2:0	4:4:4	4:2:2	4:2:0
Normalized Design Efficiency	0.40	0.25	0.67	1

In order to perform a fair comparison of various CAVLC design, a metric called *Design Efficiency* as defined in (4) is introduced. It jointly considers the pixels of Max. Spec (*Pixel*), video Format (*format*), gate count (*Gate*) and operating Frequency (*frequency*). The *format* is given by (5). The normalize *Design Efficiency* is illustrated in Table I. We can see that our design achieves the maximum *design Efficiency*. Moreover, our work can process 4 Kx2 K@60 fps video in real time.

$$Design\ Efficiency = \frac{Pixel * Format}{Gate * Frequency} \tag{4}$$

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011

$$Format = \begin{cases} 1.5, & YUV = 4:2:0 \\ 2, & YUV = 4:2:2 \\ 3, & YUV = 4:4:4 \end{cases} \quad (5)$$

4 Conclusion

This paper presents a high-throughput CAVLC encoder for the H.264/AVC system. Some parallel schemes are used to reduce the required cycles of processing one MB. With those schemes, a new CAVLC encoder architecture is proposed. Compared with other previous methods, this architecture has the following advantage: 1) it greatly reduces the cycles to process one MB, at most only 120 cycles are needed. 2) It achieves high coding speed with a comparatively less hardware cost. This high-performance CAVLC encoder can process 4 Kx2 K@60 fps video in real time.

Acknowledgments

This work is supported by State Key Lab of ASIC & System, Fudan University. Project Number: 11MS004.

DOI: 10.1587/elex.8.1863
Received August 08, 2011
Accepted October 14, 2011
Published November 25, 2011