FPGA-Based Research on The Hardware Design of CAVLC Encoder

Jin Xu[a] Jianhua Mao[b]

Electronic Information college, Xi'an Polytechnic University, Xi'an, Shanxi, P.R. China
[a] Xujin0129@126.com, [b] maojianhua2007fang@126.com

Abstract

H.264/AVC video coding standard adopt CAVLC (Context-based Adaptive Variable Length Coding) in Baseline profile and Extended profile, this paper briefly introduces the principle of CAVLC encoding; In the hardware design, the Ping-Pong operation and structure of parallel processing are adopted in order to reduce CAVLC encoded clock cycle and improve the throughput. All hardware circuits are described by Verilog HDL, and are simulated by using Modelsim SE 6.5, and are verified by using ALTERA Cyclone II EP2C35F672C8 FPGA.

Keywords: H.264/AVC; CAVLC; FPGA

1. Introduction

H.264/AVC is a new video coding standard which is established by JVT which is constituted by ITU-T's VCEG and ISO/IEC's MPEG, and bit compression ratio is 39% of MPEG-4, 49% of H.263 and 64% of MPEG-2. H.264/AVC is a high compression ratio of the new standard, but the computation complexity is large, the software is difficult to satisfy the real-time requirement of High-Definition video. The process of H.264 encoding does not involve any floating-point operation, very suitable for the realization of the hardware circuit, this system makes full use of the parallel processing and Ping-Pong operation to achieve seamless buffering, and to improve the operating rate, and to achieve real-time requirements of HD video. CAVLC is one of the key technologies in H.264.

2. CAVLC Encoding Principle

CAVLC is based on the theory of Variable Length Coding, its principle is assigns shorter bit strings to symbols expected to be more frequent and longer bit strings to symbols expected to be less frequent, so that the average length of bit strings is close to the size of entropy. Base on the traditional Variable Length Coding, CAVLC introduces a Context-based Adaptive module to achieve higher compression ratio. The so-call Context-based Adaptive is for a coding parameter with a variety of table, according to the coded syntactic elements to select table dynamically.

The encoding of CAVLC consists of the following 5 parts:

- Coeff_token: it specifies the total number of non-zero transform coefficient levels and the number of trailing one transform coefficient level scan. A trailing one transform coefficient level is one of up to three consecutive non-zero transform coefficient levels having an absolute value equal to 1 at the end of a scan of non-zero transform coefficient levels.
- Trailing_ones_sign_flag: it specifies the sign of a trailing one transform coefficient level as follows:

- If Trailing_ones_sign_flag is equal to 0,the corresponding transform coefficient level is decoded as +1;
- Otherwise(Trailing_ones_sign_flag equal to 1),the corresponding transform coefficient level is decoded as -1;
- Level: it specifies the value of a non-zero transform coefficient levels;
- Total_zeros: it specifies the total number of zero-valued transform coefficient levels that are located before the position of the last non-zero transform coefficient level in a scan of transform coefficient levels;
- Run_before: it specifies the number of consecutive transform coefficient levels in the scan with zero value before a nonzero valued transform coefficient level .

The Level is most complex in CAVLC, and the Levels is divided into two parts: level_prefix and level_suffix. The suffixLength and LevelSuffixsSize are two variables in the process of encoding. The LevelSuffixsSize, which is the length of level_suffix, is unsigned integer. In general, the value of LevelSuffixsSize is equal to the value of suffixLength, but there are two exceptions:

- When the level_prefix is equal to 14, suffixLength is equal to 0, and LevelSuffixsSize is equal to 4;
- When the level_prefix is equal to 15, and the LevelSuffixsSize is equal to 12;

The suffixLength is based on the adaptive updating of context, updates of suffixLength is relevant to the value of suffixLength and the value of Level. Initialization of SuffixLength and update is as follows:

- The initialization of suffixLength:

$$\begin{cases} \text{SuffixLength} = 1; & \text{Totalcoeffs}>10 \text{ or Trailingones}<3: \\ \text{SuffixLength} = 0; & \text{Others;} \end{cases}$$

- Encoding for non-zero coefficients of maximum frequency;
- If the current value of non_zero is greater than a predefined threshold(Q), the suffixLength plus 1, namely:

$$\begin{cases} \text{Keep original value of suffixLength ;} & |\text{Level}| \leqslant Q; \\ \text{Original suffixLength plus 1;} & |\text{Level}| > Q; \quad (2) \end{cases}$$

The table of threshold is shown in Table 1. As shown in follow:

Table 1: The threshold value

Current SuffixLength	Q(threshold)
0	0
1	3
2	6
3	12
4	24
5	48
6	N/A

CAVLC, which is a basis 4×4 block transform coefficients of Zig-zag scan, is a algorithm of encoding. Non-zero coefficients of the block is small amplitude, mainly concentrated in the low frequency band, after Zig-zag scan, most of number are continuous zeros, using run-level coding, and coding by 5 semantic elements to achieve efficient lossless compression, as shown in Figure 1.

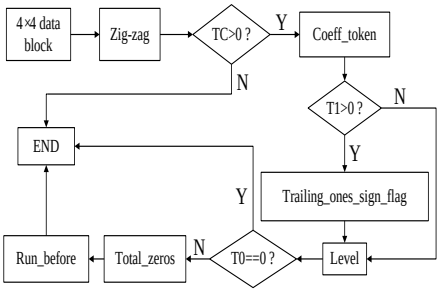

Figure 1: The process of CAVLC encoder

In Figure 1, TC, T1, T0 denote the number of non-zero coefficient, the number of 1 about tail number and the number of zero before the last non-zero coefficients respectively. As a result of the process of CAVLC encoding is serial, the software is easy to implement, but the implementation is slow speed and low efficiency. CAVLC encoding details is showed in reference [2], it would be superfluous to dwell on the paper.

3. The Realization of Hardware Circuit About CAVLC

The structure of CAVLC encoder is showed in Figure 2. In the entire encoding module, first to obtain the statistical information, which is to acquire the number of non-zero coefficients and the number of trailing coefficient before encoding the 4×4 blocks, at the same time through the NC module to obtain the value of NC, then use to choose the look-up table. In Figure 2, NA(the number of nonzero coefficients in a 4×4 block of the current block to the left), NB(the number of non-zero coefficients of a 4×4 block above the current block), NAEN and NBEN are control signal of NC module. After the zig-zag scan of 16 clock cycles, the coefficient of 4 x 4 block is read into FIFO serial, in this process, through the Level Detection module to record the non-zero in addition to Trailing_ones_sign_flag in the integer string, the coding sequence is reverse of the integer string after scanned, namely from right to left. Each time, the coefficients is read into the Trailing_ones counter, Coeff_token counter and the Run_before counter, then the modules will updating. After 16 clock cycles, through the serial registers to access the address of look-up table, then output the bit strings of look-up table. The hardware structure of CAVLC encoder is showed in Figure 2.

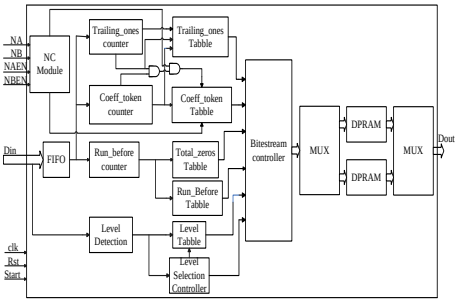

Figure 2: The hardware structure of CAVLC encoder

In this design, the codeword is obtained by the respective module, and then the codeword get into the module of Bitestream controller, in accordance with the H.264 standard, and output encoded codeword and length of codeword in proper sequence, then get into the multiplexer, realizing the ping-pong operation.

Ping-Pong operation is the characterized by the data stream without a pause between the input selection module and output selection module on the beat, then the data is processed. The module of ping-pong is operated as a whole, and from the end of this module, the output data stream is continuous, and from the head of this module, the input data stream is also continuous, and very suitable for data stream pipelined processing, so the ping-pong operation is often applied in the pipelined algorithms, and achieve a seamless processing of data buffer. The block diagram of Ping-Pong operation is showed in Figure 3.

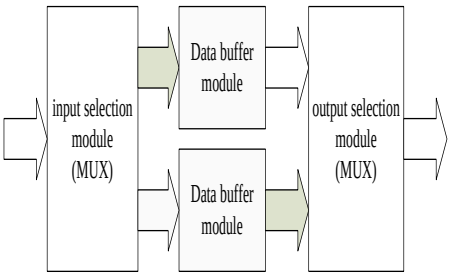

Figure 3: Block diagram of Ping-Pong operation

In the design, the Ping-Pong operation is to achieve seamless buffering, improve the data output rate. This module consists of two MUX and two DPRAM, through to the multiplexing of function module, in order to achieve the target which is improving the operating rate.

4. Simulation and Verification of CAVLC Encoder

Using the sequence of standard test: 0, 0, 5, 3, 2, -1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, and the NC value is 3, using the design of the CAVLC encoder to simulation by Modelsim SE 6.5, and the results as shown in Figure 4. In the figure, when trans_go is 1, it's said the output value is effective when the next clock cycle is come; when trans_over is 1, it's said the output value is end when the next clock cycle is come. The bitstreams of data_out port is the output of CAVLC encoded.

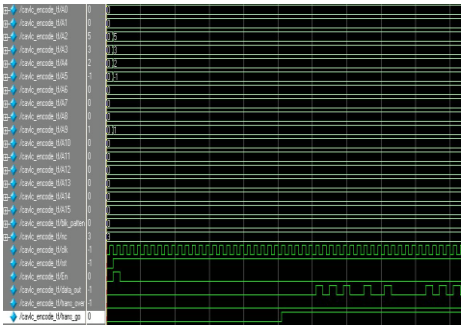

Figure 4: The simulation results

The output is the encoding through manual look-up table: 00001011000100101010101111, the results and the results of CAVLC encoder is same, thus verify the correctness of the CAVLC encoder.

5. Conclusion

This paper gives the analysis of CAVLC encoding algorithm in the H.264/AVC standard, and to solve this problem that the algorithm complexity is high and not easy to achieve require of real-time, and the paper put forward a high speed system for CAVLC encoding of H.264 standard. In this system, programming with Verilog HDL achieves the hardware circuit, and through FPGA to validate the correctness of the design. The highest of frequency is up to 148.67MHZ in this system, it can satisfy the requirement of real-time about HD video, and the design has a greater of significance in engineering practice.

6. References

[1] Houjie Bi, "A new generation of video coding standard—H.264/AVC[M]," *Beijing. Posts & Telecom press,* pp. 118-123, 2005.

[2] ITU-T, "H.264 Advanced Video Coding for Generic Audiovisual Services[S]," 2005

[3] Yuwen Xia, "digital system design course base on Verilog[M]," *Beijing. Beijing Aerospace University Press,* 2008

[4] Shuo Wu, "Implementation OF the Key Algorithms in H.264/AVC Base on FPGA[D]," *Xi'an. XIDIAN University,* 2011

[5] Ling Zhang, Fang Li, Wei He, "CAVLC and its FPGA realization for H.264[J]," *Computer Engineering and Applications, vol.44(17),* pp. 78-81,2008