Hardware Implementation of HEVC CABAC Binarizer

Duyen Hai Pham[*], Jeonhak Moon[*] , Seongsoo Lee[*][★]

Abstract

This paper proposes hardware architecture of HEVC (high efficiency video coding) CABAC (context-based adaptive binary arithmetic coding) binarizer. The proposed binarizer was designed and implemented as an independent module that can be integrated into HEVC CABAC encoder. It generates each bin string of each syntax element in a single cycle. It consists of controller module, TU (truncated unary binarization) module, TR (truncated Rice binarization) module, FL (fixed length binarization) module, EGK (k-th order exp-Golomb coding) module, CALR (coeff_abs_level_remaining) module, QP Delta (cu_qp_delta_abs) module, Intra Pred (intra_chroma_pred_mode) module, Inter Pred (inter_pred_idc) module, and Part Mode (part_mode) module. The proposed binarizer was designed in Verilog HDL, and it was implemented in 45 nm technology. Its operating speed, gate count, and power consumption are 200 MHz, 1,678 gates, and 50 uW, respectively.

Key words: HEVC, CABAC, binarizer, hardware, implementation

I. Introduction

HEVC (high-efficiency video coding) [1]-[4] is the next-generation international standard of video codec succeeding H.264/AVC (advanced video coding). This standard delivers the same quality as H.264/AVC at close to 50% of the bitrate, has ability to support higher resolutions reach 8K. It achieves better quality with reduced bandwidth costs, which enables to deliver HD (high definition) or UHD (ultra high definition) contents to mobile devices over limited bandwidth.

CABAC (context-based adaptive binary arithmetic coding) is an entropy coding technique used in H.264/AVC and also in HEVC with high coding efficiency. It performs three main functions:

binarization, context modeling, and arithmetic coding, describes as the Fig. 1. Binarization generates binary symbol, called as bin, from the syntax elements. Context modeling calculates the probability of the each bin based on the neighboring information and last coded bins. Finally, binary arithmetic encoding compresses the bins into bits based on the estimated probability.

Binarizer is one of core blocks in CABAC encoder. This paper proposes the architecture of binarizer for HEVC CABAC encoder. It was designed in Verilog HDL and was implemented in 45 nm technology.

II. Hardware Architecture

HEVC CABAC binarization has several basic binarization formats such as unary (U), truncated unary (TU), truncated Rice (TR), kth-order Exp-Golomb (EGK), and fixed length (FL). Almost all syntax elements exploit these basic binarization formats, but some syntax elements such as coeff_abs_level_remaining (CALR) and cu_qp_delta_abs (QP Delta) exploit two or more combinations of basic binarization. Some syntax elements such as inter_pred_idc (Inter Pred Mode), intra_chroma_pred_mode (Intra Pred Mode), and part_mode (Part

∗ School of Electronic Engineering, Soongsil University, sslee@ssu.ac.kr, 010-9182-3835
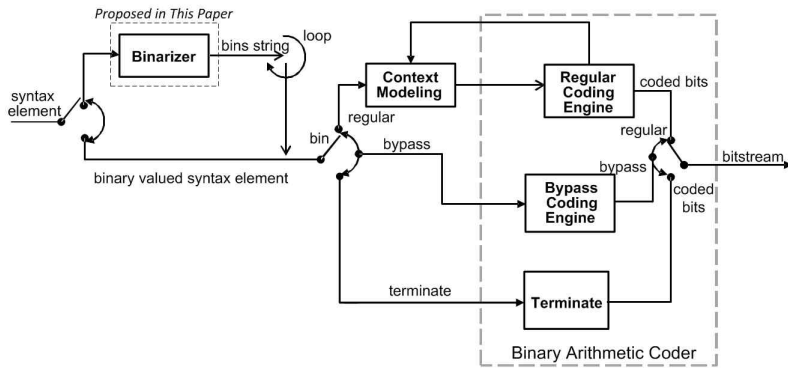★ Corresponding author

Fig.   1.   Block diagram of HEVC-CABAC encoder.

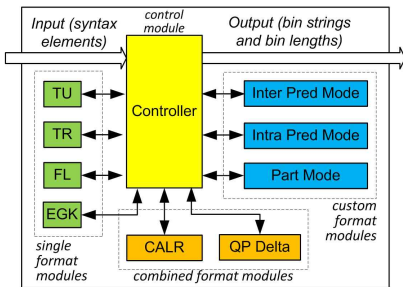Fig.   2.   Architecture of the proposed binarizer.
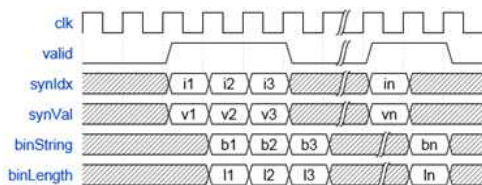
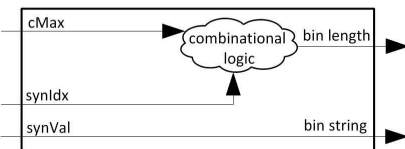Fig.   3.   Timing chart of the proposed binarizer.

Fig.   4.   FL module.

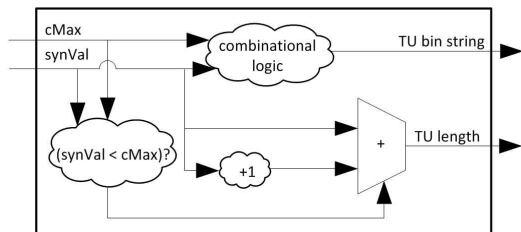| prefix/val	bin string						
0	0						
1	1	0					
2	1	1	0				
3	1	1	1	0			
4	1	1	1	1	0		
5 (cMax=5)	1	1	1	1	1		
...	...						
binIdx	0	1	2	3	4	5	6

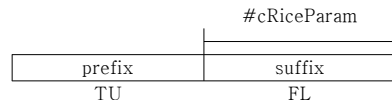Fig.   5.   TU Binarization.

Fig.   6.   TU module.

Fig.   7.   TR Binarization.

Mode) use custom binarization formats.

Comparing with H.264/AVC, the basic binarization formats are almost same, but combination and custom binarization processes are different. The proposed binarizer supports all the basic and custom binarization format in HEVC. It gets syntax element as an input and puts a bin string as an output.

1. Top Architecture

The proposed binarizer consists of four groups of submodules such as controller module, single format modules, combined format modules, and custom format modules. Its inputs are *synVal* and *synIdx* which determine the specific binarization process for corresponding syntax element. Its outputs are bin
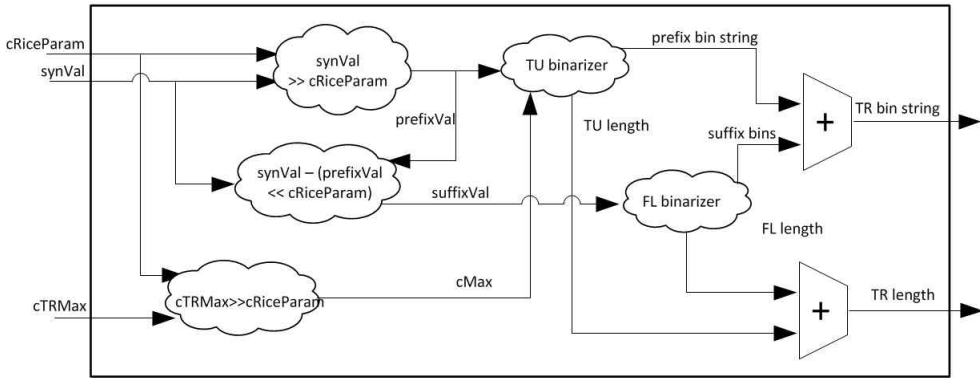
Fig. 8. TR module.

string and bin length of each syntax element.

The controller module gets the *synVal* and *synIdx* from peripheral bus. It categorizes the type of binarization format of each syntax element based on *synIdx* and assigns it to corresponding submodule.

Single format modules are TU, TR, FL, and EGK modules, which process their corresponding single binarization formats. Combined format modules are CALR (*coeff_abs_level_remaining*) and QP Delta (*cu_qp_delta_abs*) modules, which invoke single format modules and process their corresponding combined binarization formats.

Cunstom format modules are Inter Pred Mode (*inter_pred_idc*), Intra Pred Mode (*intra_chroma_pred_mode*), and Part Mode (*part_mode*) modules. They process custom binarization formats, and they are implemented as look-up tables.

The hardware architecture of the proposed binarizer is shown in Fig. 2. Its latency is one cycle, as shown in the timing chart of Fig. 3.

2. FL Module

Fixed length binarization is performed by using a fixed length unsigned integer bin string which represents a binary value of syntax element value. The length of bin string is equal to *Ceil(Log2(cMax+1))*. Bin string starts from most significant bit (MSB) to least significant bit (LSB). FL module is designed as shown in Fig. 4.

3. TU Module

Truncated unary binarization is a reduction form of unary scheme which generates a bin string of "1" and follow a terminating "0" bit at the end of bin string for the case of *synVal* < *cMax*. If the *synVal* reaches to *cMax*, terminating "0" is not presented in the TU bin string. The length is calculated from *synVal* + 1 (or *synVal* when *synVal* equals to *cMax*). Fig. 5 explains TU binarization. TU module is designed as shown in Fig. 6.

4. TR Module

Truncated Rice binarization consists of two parts, i.e. prefix and suffix (when it present) bin string. The prefix part invokes TU binarization. The suffix part invokes FL binarization as shown in Fig. 7. The length of suffix part is equal to the value of *cRiceParam*. *prefixVal* = *synVal>>cRiceParam* and *suffixVal=synVal−((prefixVal)<<cRiceParam)*. When *cRiceParam*=0, TR binarization becomes TU binarization. TR binarization is used in CALR syntax element for prefix bin string. TR module is designed as shown in Fig. 8.

5. EGK Module

k-th Exp-Golomb binarization is variable length codes. The bin length of code words increases exponentially with their value. Each codeword consists of prefix and suffix parts. This codeword is lead by N bit separators of "1" or "0", and it is followed by N bit suffix codewords which contains information.

Table 1. EGK binarization.

syntax element value（synVal）	CodeNum
0	0
1	1
−1	2
2	3
−2	4
3	5
−3	6
...	...
$(-1)^{k+1}$Ceil(k/2)	k

prefix format		
k=0	k=1	k=2
0	0x	0xx
10x	10xx	10xxx
110xx	110xxx	110xxxx
1110xxx	1110xxxx	1110xxxxx
...	...	...

k	Code Num	prefix format	bin string
0	0	0	0
	1	10x	10**0**
	2		10**1**
	3	110xx	110**00**
	4		110**01**
	5		110**10**
	6		110**11**
	...	...	...
1	0	0x	0**0**
	1		0**1**
	2	10xx	10**00**
	3		10**01**
	4		10**10**
	5		10**11**
	...	...	...
2	0	0xx	0**00**
	1		0**01**
	2		0**10**
	3		0**11**
	4	10xxx	10**000**
	5		10**001**
	6		10**010**
	7		10**011**
	8		10**100**
	9		10**101**
	10		10**110**
	11		10**111**
	...	...	...

EGK binarization represents signed number as shown in Table 1. *synVal* is divided into two parts by the k-order variable. *prefixVal* is added by one and after that using MSB position logic detects the most significant one bit existed in the *prefixVal*
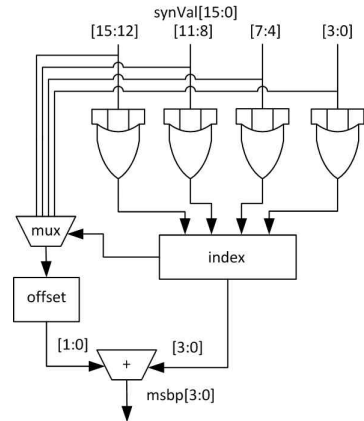

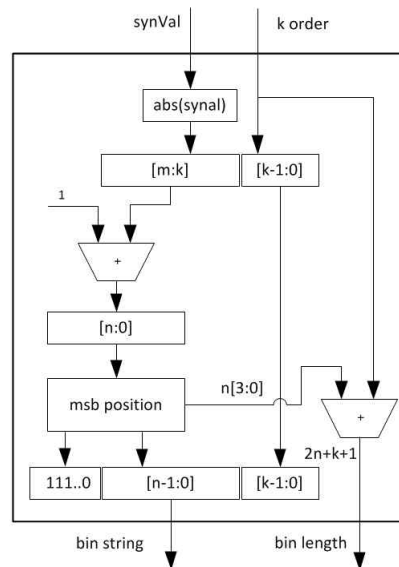Fig. 9. MSB position logic in EGK module.

Fig. 10. EGK module.

which is used to determine the number of "1" bins for prefix bin string. Suffix bins represented by *suffixVal* added at the end of bin string.

MSB position logic is designed as shown in Fig. 9. Index variable indicates the highest position of sub-string contains "1" bit, the range is 0, 4, 8, 12. An offset variable indicates the position of "1" present in the sub-string with the range from 0..3. *N* position is the sum of index and offset. *N* is also used to calculated the length of EGK bin string. EGK module is designed as shown in Fig. 10.
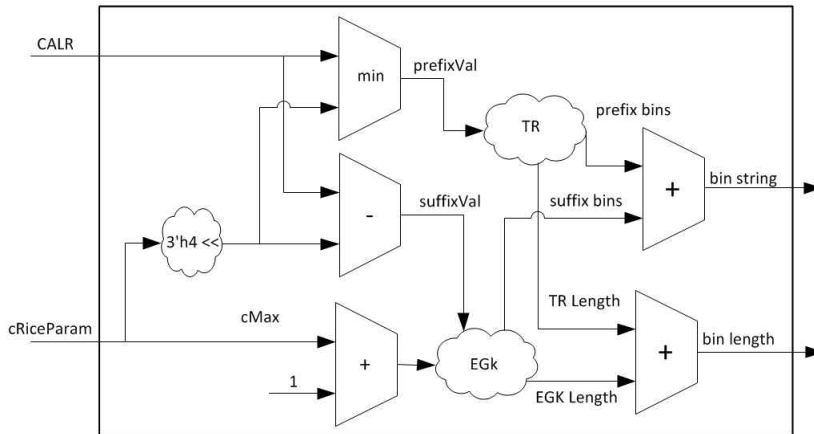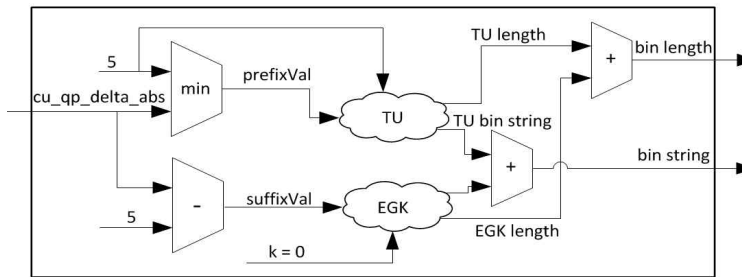
Fig. 11. CALR module.

Fig. 12. QP Delta module.

6. CALR Module

CALR module invokes TR binarization and EGK binarization for generating bin string. $prefixVal$=Min($cMax$, $coeff\_abs\_level\_remaining$[n]) and $suffixVal$=$coeff\_abs\_level\_remaining$[n]-cMax. CALR module is designed as shown in Fig. 11.

7. QP Delta Module

The bin string is a concatenation of a prefix bin string and a suffix bin string (when it present). $prefixVal$=Min($cu\_qp\_delta\_abs$, 5) which will invoke the TU binarzation with $cMax$=5. $suffixVal$=$cu\_qp\_delta\_abs$−5. It invoke the EGK binarization with 0th k-order. QP Delta module is designed as shown in Fig. 12.

8. Lookup Tables for Intra Pred Module, Inter Pred Module, and Part Mode Module

Intra Pred module, Inter Pred module, and Part Mode module can be easily designed with lookup tables [1].

III. Experiments and Results

The proposed binarizer was designed in Verilog HDL. It was simulated and verified using 12 test vectors, i.e. 4 standard test sequences (class A, B, C, and D) with 3 encoder configurations (low delay, random access, and all intra) as shown in Table 2. It has passed in all tests.

The designed binarizer was implemented in 45 nm technology. CAD tool was supported by IC Design Education Center (IDEC). As shown in Table 3, its operating speed, gate count, and power consumption are 200 MHz, 1,678 gates, and 50 uW, respectively. Its critical path delay is 4.763 ns. The latency of the designed binarizer is only one cycle.

Table 2. Test sequences.

Sequence	Class A	Class B	Class C	Class D
Size	2560x 1600	1920x 1080	832x 480	416x 240
Frame Rate	30	24	50	50
Frames	10	10	10	10
Bit Depth	8	8	8	8
Encoder	3 Configurations (low delay, random access, and all intra)			

Table 3. Synthesis Results.

Technology	45nm
Operating Frequency	200 MHz
Gate Counts	1,678 gates
Critical Path Delay	4.763 ns
Dynamic Power	50 uW
Leakage Power	3.25 uW

IV. Conclusions

In this paper, a binarizer for HEVC CABAC encoder was proposed. It was designed in Verilog HDL with 10 modules to cover 4 basic binarization modes and their combined modes. It was implemented in 45 nm technology. It can encode one bin string of each syntax element per cycle. Its operating speed, gate count, and power consumption are 200 MHz, 1,678 gates, and 50 uW, respectively.

References

[1] B. Bross, W. Han, J. Ohm, G. Sullivan, and T. Wiegand, "JCTVC-L1003_v34: High efficiency video coding (HEVC) text specification draft 10," Joint Collaborative Team on Video Coding (JCT-VC), Jan. 2013.

[2] HEVC software repository HM-11 reference model at https://hevc.hhi.fraunhofer.de/svn/svn_ HEVCSoftware/branches/HM-11.0-dev/

[3] S. Han, W. Nam, and S. Lee, "Design of Low-Area HEVC Core Transform Architecture", Journal of IKEEE, vol. 17, no. 2, pp. 119-128, Jun. 2013.

[4] J. Lee and S. Lee, "8×8 HEVC Inverse Core Transform Architecture Using Multiplier Reuse", Journal of IKEEE, vol. 17, no. 4, pp. 570-578, Dec. 2013.

BIOGRAPHY

Duyen Hai Pham (Student Member)

2007 : BS degree in Physics, Soongsil University.
2013~Now : MS candidate in Electronic Engineering, Soongsil University.
<Main Interest> HEVC, Multimedia SoC Design

Jeonhak Moon (Student Member)

2005 : BS degree in Electronic Engineering, Soongsil University.
2008 : MS degree in Electronic Engineering, Soongsil University.
2008~Now : PhD candidate in Electronic Engineering, Soongsil University.
<Main Interest> HEVC, Multimedia SoC Design

Seongsoo Lee (Life Member)

1991 : BS degree in Electronic Engineering, Seoul National University.
1993 : MS degree in Electronic Engineering, Seoul National University.
1998 : PhD degree in Electrical Engineering, Seoul National University.
1998~2000 : Research Associate, University of Tokyo
2000~2002 : Research Professor, Ewha Womans University
2002~Now : Associate Professor in School of Electronic Engineering, Soongsil University
<Main Interest> HEVC, Low-Power SoC Design, Multimedia SoC Design, Battery Management