УДК 621.382.8

IP-блок кодирования CAVLC для видеокодека H.264/AVC

И.А. Беляев

ОАО НПЦ «ЭЛВИС», ioganess@yandex.ru

***Аннотация* — В данной работе рассматривается аппаратная реализация (IP-блок) алгоритма кодирования CAVLC (Context-Adaptive Variable Length Coding), используемого в стандарте сжатия видео H.264/AVC. IP-блок реализован в виде RTL-модели на языке Verilog. Приводится краткое описание алгоритма CAVLC. Описывается архитектура IP-блока кодирования CAVLC. Приводятся характеристики IP-блока и осуществляется его сравнение с аналогами.**

***Ключевые слова* — IP-блок, CAVLC, H.264, AVC.**

I. Введение

H.264/AVC – один из новейших стандартов сжатия цифрового видео [1]. Впервые опубликованный в 2003 году, он начинает набирать популярность как один из лучших по степени сжатия [3]. В то же время технологии, сделавшие H.264 столь сильным в плане сжатия, сделали его более затратным в плане количества вычислений по сравнению с предыдущими стандартами [4].

В задаче кодирования видеоинформации можно выделить три главные, составляющие суть процесса подзадачи: формирование предсказания кадра, трансформация данных кадра и энтропийное кодирование (рис.1).

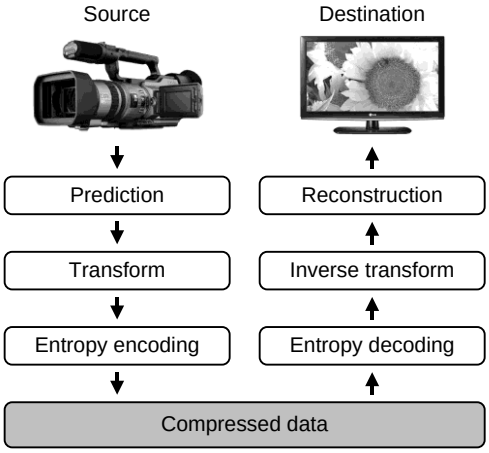

Рис. 1. Основные этапы кодирования и декодирования видео

Проблема первых двух подзадач заключается в большом количестве операций сложения и умножения, проблема последней – в оперировании кодами переменной длины. Упаковка потока таких кодов в фиксированные по длине слова памяти связана с постоянными операциями сдвига и проверки переполнения текущего слова. А каждая проверка на переполнение является командой условного ветвления и, в случае выполнения условия, приводит к перезагрузке программного конвейера и потере нескольких тактов. Кроме того необходим анализ кодируемых данных, что также приводит к командам сравнения, последующим условным командам и задержкам конвейера.

И если проблема формирования предсказания и трансформации данных кадра хорошо решается использованием векторного/матричного процессора, то для решения проблемы энтропийного кодирования необходимы специализированные вычислительные блоки, ориентированные на быстрое выполнение одной конкретной задачи.

В стандарте H.264 используется два вида энтропийного кодирования: кодирование контекстно-адаптивными кодами переменной длины (CAVLC) и арифметическое кодирование (CABAC). В данной статье рассматривается аппаратная реализация IP-блока кодирования CAVLC.

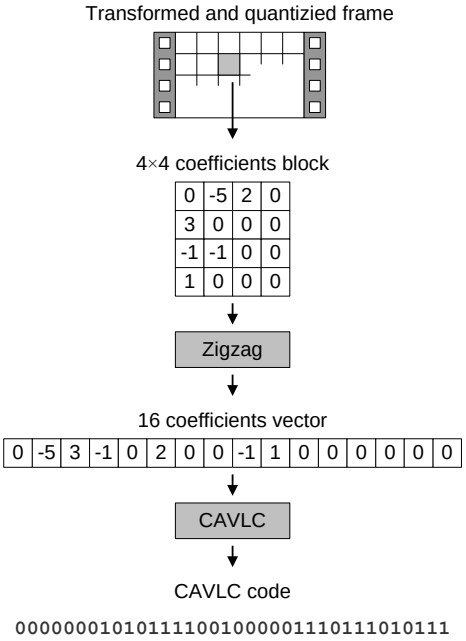

Рис. 2. Процедура кодирования CAVLC

II. КОДИРОВАНИЕ CAVLC В СТАНДАРТЕ H.264/AVC

В кодере H.264 метод CAVLC [2, 5] имеет дело с просканированными по зигзагу блоками 4×4, 2×4 или 2×2 квантованных коэффициентов. Таким образом, на вход CAVLC подаётся вектор из 16, 8 или 4 элементов. На выходе CAVLC получается битовая строка, соответствующая кодируемому вектору (рис.2).

Обычно после предсказания, преобразования и квантования блоки коэффициентов становятся весьма разреженными, т.е. содержат много нулевых коэффициентов. Для компактного представления серий нулевых коэффициентов используется схема «серия-значение», которая в случае CAVLC подразумевает отдельное кодирование ненулевых коэффициентов и отдельное кодирование серий нулей перед ними (рис.3).

0	-5	3	-1	0	2	0	0	-1	1	0	0	0	0	0	0

Рис. 3. Схема «серия-значение»

Завершающими элементами вектора ненулевых коэффициентов обычно являются числа ±1. В CAVLC они называются остаточными единицами (TrailingOnes) и кодируются особым образом. Остаточных единиц не может быть больше трёх. В случае, если более трёх завершающих ненулевых коэффициентов равны ±1, остаточными единицами считаются только 3 последних из них (рис.4).

0	-5	3	-1	0	2	0	0	-1	1	0	0	0	0	0	0

Рис. 4. Остаточные единицы

Битовая строка, получаемая в результате кодирования блока коэффициентов методом CAVLC, состоит из следующих синтаксических элементов:

coeff_token – код для общего количества ненулевых коэффициентов (TotalCoeff) и количества остаточных единиц (TrailingOnes)

trailing_ones_sign_flag – код для знака остаточной единицы (TrailingOne)

level_prefix – первая часть кода ненулевого коэффициента (кроме остаточных единиц)

level_suffix – вторая часть кода ненулевого коэффициента (кроме остаточных единиц)

total_zeros – код для общего количества нулей, предшествующих всем ненулевым коэффициентом (не считаются завершающие нули, стоящие после последнего ненулевого коэффициента)

run_before – код для серии нулей, предшествующей ненулевому коэффициенту

III. АРХИТЕКТУРА IP-БЛОКА КОДИРОВАНИЯ CAVLC

Предлагаемая архитектура IP-блока является универсальной в том смысле, что IP-блок способен кодировать блоки квантованных коэффициентов любого возможного размера (4×4, 2×4 или 2×2) в любом, определяемом программистом порядке. Таким образом, в отличие от [7, 8], снимается ограничение на использование какой-либо одной дискретизации цветности (4:4:4, 4:2:2 или 4:2:0) или какого-либо определённого профиля и уровня стандарта H.264. Также предлагаемый IP-блок способен осуществлять зигзаг-преобразование входных данных (комбинационно, без дополнительных тактов) и вычислять параметр nC для кодируемого блока.

A. Структура IP-блока

Структура предлагаемого IP-блока кодирования CAVLC приведена на рис.5.

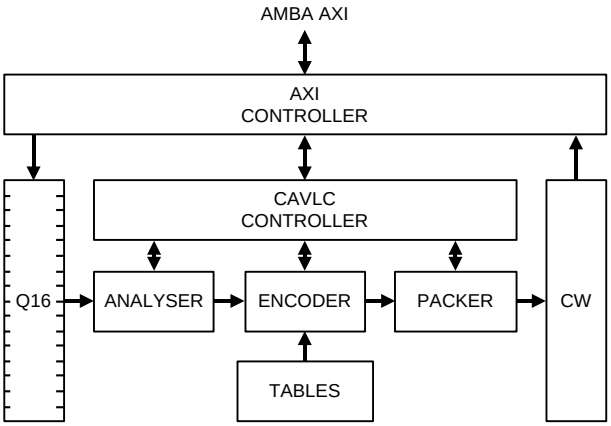

Рис. 5. Структура IP-блока кодирования CAVLC

IP-блок состоит из следующих элементов:

- AXI CONTROLLER – контроллер 256-разрядной шины AMBA AXI. IP-блок является slave-устройством. По шине AXI master-устройство может осуществлять управление IP-блоком, записывать входные данные и считывать выходные.
- CAVLC CONTROLLER – блок управления. Содержит управляющие регистры и формирует управляющие сигналы.
- Q16 – 256-разрядный регистр входных данных. Вмещает массив из 16 16-разрядных коэффициентов.
- ANALYSER – анализатор входных данных. Осуществляет анализ регистра Q16, определяет ненулевые коэффициенты, остаточные единицы и длины серий нулей.
- ENCODER – кодер CAVLC. Ставит в соответствие элементам coeff_token, total_zeros и run_before коды из таблиц. Формирует коды для элементов trailing_ones_sign_flag, level_prefix и level_suffix.
- TABLES – таблицы кодов CAVLC.
- PACKER – упаковщик кодов CAVLC. Соединяет получаемые от блока ENCODER коды в одно кодовое слово.
- CW – 256-разрядный регистр выходных данных.

B. Управление IP-блоком

Управление IP-блоком осуществляется посредством записи и чтения его внутренних регистров, доступных через интерфейс AMBA AXI. Состав программно-доступных регистров IP-блока приведён в табл. 1.

Таблица 1

Программно-доступные регистры IP-блока

Название	**Описание**
CSR	Регистр управления и состояния
nC	Параметр nC
CW_TOTAL_LEN	Итоговая длина выходного кодового слова (бит)
CW_EXTBITS	Внешняя битовая строка для вставки в выходное кодовое слово
CW_EXTBITS_LEN	Длина внешней битовой строки
Q16	Регистр входных данных
CW	Регистр выходных данных

C. Режимы работы IP-блока

IP-блок может работать в четырёх режимах, или выполнять четыре команды.

- Сброс

Режим сброса приводит к установке всех управляющих кодированием регистров IP-блока в их начальное состояние.

- Кодирование

В этом режиме IP-блок осуществляет кодирование входных данных. Кодируется (coeff_num_m1 + 1) младших 16-разрядных коэффициентов из регистра Q16. Выходные данные записываются в регистр CW. В случае заполнения всех 256 бит регистра CW в CSR устанавливается флаг готовности кодового слова.

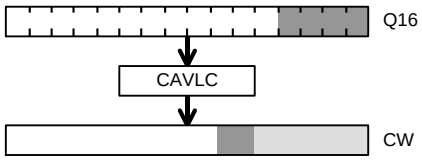

Рис. 6. Режим кодирования

- Вставка в кодовое слово внешних бит

В этом режиме в регистр CW добавляются CW_EXTBITS_LEN младших бит регистра CW_EXTBITS. Учитывая особенности сжатых данных H.264, этот режим может быть полезен для добавления служебной информации в поток кодов CAVLC.

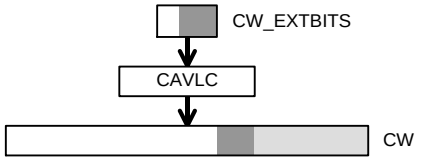

Рис. 7. Режим вставки в кодовое слово внешних бит

- Выгрузка незавершённого кодового слова

В этом режиме в регистр CW выталкиваются остатки предыдущего кода из блока PACKER и выставляется флаг готовности кодового слова. Этот режим необходим для завершения кодирования определённой части данных кадра.

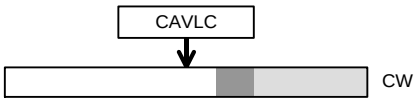

Рис. 8. Режим выгрузки незавершённого кодового слова

D. Конвейер IP-блока

Конвейер обработки данных IP-блока состоит из 12 фаз (рис.9). Описание фаз конвейера приведено в табл. 2. Регистр входных данных не конвейеризован в силу большого размера, поэтому новые входные данные могут быть записаны в IP-блок только после полного завершения кодирования предыдущих (фаза c0).

Таблица 2

Описание фаз конвейера IP-блока

Фаза	**Длит. (тактов)**	**Описание**
a0	1	Анализ входных данных
a1	1	
a2	1	
t0	1	Кодирование coeff_token и total_zeros (если необходимо)
t1	1	
c0	0 .. 16	Кодирование ненулевых коэффициентов (level_prefix и level_suffix). Кодирование и предварительная упаковка длин серий нулей (run_before)
c1	0 .. 16	
c2	0 .. 16	
c3	0 .. 1	Отправка total_zeros и run_before на упаковку
p0	1 .. 18	Упаковка кодов в выходное кодовое слово
p1	1 .. 18	
p2	1 .. 18	

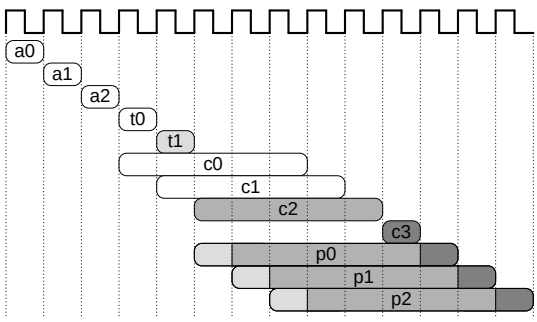

Рис. 9. Конвейер IP-блока

Особенностью архитектуры IP-блока, аналогичной [7, 9], является отсутствие временных затрат на обработку нулевых коэффициентов (табл. 3). Такты тратятся только на кодируемые элементы. Однако, в

отличие от [9], быстрее происходит запись входных данных, а также total_zeros и все run_before упаковываются вместе за 1 такт.

Быстродействия предлагаемого IP-блока повышено за счёт введения межблочной конвейеризации. Без межблочной конвейеризации IP-блок готов к обработке следующего блока коэффициентов только после завершения фазы p3. В этом случае затраты на кодирование одного блока 4×4 коэффициентов составляют от 8 до 25 тактов. С межблочной конвейеризацией IP-блок готов к обработке следующего блока коэффициентов уже после фазы a1 (если нет ненулевых коэффициентов) или c0. Тогда затраты на кодирование одного блока 4×4 составят от 2 до 19 тактов.

Таблица 3

Временные затраты (количество тактов) на кодирование одного блока коэффициентов

	[7]	**[9]**	**IP-блок**
Запись входных данных для блока 4×4	16	4	1
Кодирование coeff_token	1	0..1	1
Кодирование ненулевых коэффициентов (trailing_ones, level_prefix, level_suffix)	TotalCoeff	TotalCoeff	TotalCoeff
Кодирование total_zeros	1	1	1
Кодирование run_before	TotalCoeff	1	0

Таблица 4

Сравнение аппаратных затрат и производительности с аналогами

	[6]	**[7]**	**[8]**	**[10]**	**IP-блок**
Технологический процесс, нм	180	180	180	180	65
Площадь, кол-во вентилей	9,7K	23,6K	13,1	66K	30K
Рабочая частота, МГц	125	100	133	200	500
Производительность, кадров в секунду для HD 1080 (1920×1080) 4:2:0 при частоте 100 МГц	24	30	23	60	26 .. 237 (78 при QP=20)

IV. Реализация IP-блока кодирования CAVLC

Предлагаемая архитектура IP-блока была реализована в виде RTL-модели на языке Verilog. Для верификации IP-блока была написана модель кодера CAVLC на языке Matlab. Верификация RTL-модели осуществлялась в среде Cadence IUS путём сравнения выходного кода IP-блока с эталонным, полученным из модели на языке Matlab.

Логический синтез RTL-модели проводился с помощью Synopsys Design Compiler на библиотеке элементов TSMC с проектными нормами 65 нм. Полученные характеристики IP-блока и их сравнение с аналогами приведены в табл. 4.

V. Заключение

Занимая небольшую площадь, предлагаемый IP-блок позволяет обеспечить высокую скорость выполнения процедуры кодирования CAVLC, что может быть полезно в системах, ориентированных на сжатие видео по стандарту H.264/AVC в режиме реального времени.

Литература

[1] ITU-T Recommendation H.264 // ISO/IEC 14496-10 (03.2010).

[2] Overview of the H.264/AVC video coding standard / Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, Ajay Luthra // IEEE Transaction on Circuits and Systems for Video Technology. 2003. Vol. 13. No. 7. P. 560–576.

[3] Nejat Kamaci, Yucel Altunbasak. Perfomance comparison of the emerging H.264 video coding standard with the existing standards // Multimedia and Expo 2003 International Conference. 2003. Vol. 1. P. 345–352.

[4] Jörn Ostermann, Jan Bormans, Peter List. Video coding with H.264/AVC: tools, performance, and complexity // IEEE Circuits and Systems Magazine. First quarter 2004. P. 8–28.

[5] Ян Ричардсон. Видеокодирование. H.264 и MPEG-4 – стандарты нового поколения. М.: Техносфера. 2005. С. 261–269.

[6] A high performance CAVLC encoder design for MPEG-4 AVC/H.264 video coding applications / Chih-Da Chien, Keng-Po Lu, Yi-Hung Shih, Jiun-In Guo // IEEE International Symposium on Circuits and Systems. 2006. P. 3838–3841.

[7] Dual-block-pipelined VLSI architecture of entropy coding for H.264 baseline profile / Tung-Chien Chen, Yu-Wen Huang, Chuan-Yong Tsai, Bing-Yu Hsieh, Liang-Gee Chen // Proc. International Symposium on VLSI Design, Automation and Test (VLSI-DAT). 2005. P. 271–274.

[8] Hu Hongqi, Sun Jingnan, Xu Jiadong. High performance architecture design of CAVLC encoder in H.264/AVC // Congress on Image and Signal Processing. 2008. P. 613–616.

[9] Min-Chi Tsai, and Tian-Sheuan Chang. High performance context adaptive variable length coding encoder for MPEG-4 AVC/H.264 video coding // IEEE Conference on Circuits and Systems. 2006. P. 586–589.

[10] Yongseok Yi, Byung Cheol Song. High-speed CAVLC encoder for 1080p 60-Hz H.264 codec // IEEE Signal Processing Letters. 2008. Vol. 15. C. 891–894.