A Bit-serial Implementation of Mode Decision Algorithm for AVC Encoders

Paweł Garstecki, Adam Łuczak and Marta Stępniewska
Division of Multimedia Telecommunications and Radioelectronics
Poznan University of Technology
Piotrowo 3a, 60-965 Poznan, Poland
Email: [pgarstec,aluczak,mstep]@multimedia.edu.pl
Telephone: +48 61 6652870 Fax: +48 61 6652292

***Abstract*— The paper presents a new and efficient architecture for H.264/AVC video encoder control. The architecture of mode decision and cost estimation module is implemented with the use of bit-serial arithmetic and provides pipelined processing of image blocks. The module is designed to support FPGA devices. It has been shown that the design is capable to perform at a very low clock speed, thus it is a suitable solution for wireless communications. The proposed modules have been implemented in Verilog HDL and synthesized for a Xilinx Virtex II family device.**

I. Introduction

The most recent H.264/AVC video compression standard [1] [2] defines a set of highly effective tools that enable achieving a very high compression gain in bitrate reduction in comparison to other methods. The proposed encoding algorithms, however, are characterized by high computational complexity and require high speed processors [3], [4].

The high complexity of the new methods does not facilitate popularisation of the new compression standard in some areas of application due to high power consumption. Some devices e.g. mobile devices do not have enough computational power and energy resources to encode and decode H.264/AVC video stream.

For such a case a pure hardware implementation of video codec seems to be more suitable. Most of the known solutions integrate a general purpose processor and a dedicated hardware acceleration block. However, this does not guarantee low power consumption, and application of such architectures to wireless communications is difficult as well.

Most of implementations of AVC codecs are using parallel arithmetic, due to simplicity of the design process and integration with a processor unit [5]. In this paper a pure hardware implementation of the analysis block of the AVC encoder is presented. The main functional blocks of the proposed structure are implemented in bit-serial arithmetic. This allows for constructing devices that are characterized by low power consumption and high processing efficiency. The presented block is developed to support FPGA devices that are composed of many simple configurable logic blocks (CLB). Processing based on a bit-serial arithmetic uses simple logic functions (e.g. bit-serial address, shift registers), thus its application to codecs' blocks allows for reduction of data busses, long connections and gaining simplicity of construction. This architecture is integrated with a reconstruction block of the AVC decoder presented in [6] as well.

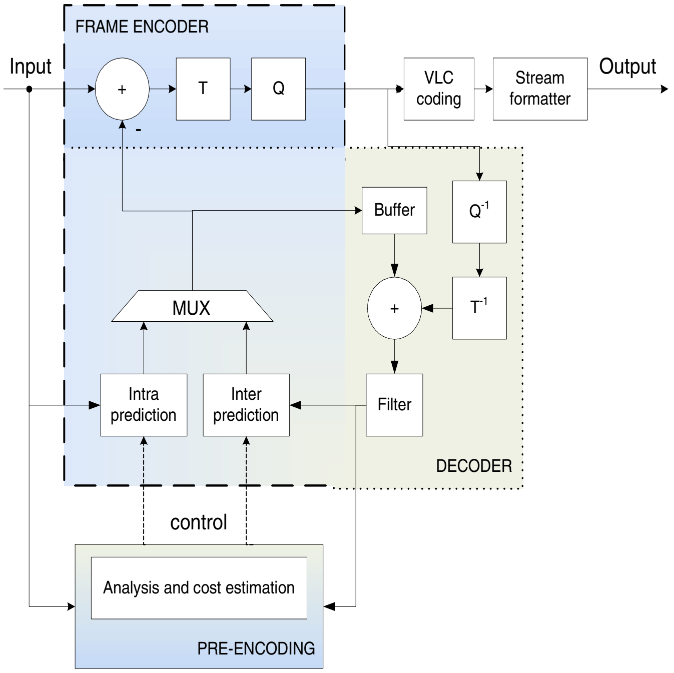
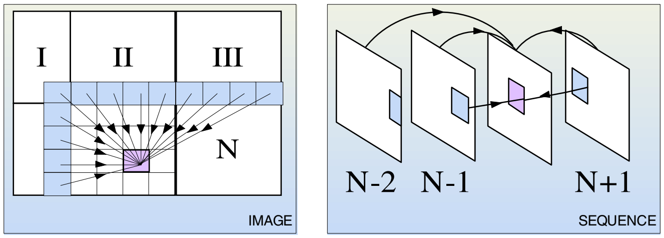


Fig. 1. The H.264/AVC encoder block diagram with a decoder reconstruction block (T- transformation, Q-quantization module).

II. The H.264/AVC Encoder

The general structure of the AVC encoder is shown in Fig. 1. In this diagram four major blocks of the encoder can be distinguished: a pre-coding block, a frame encoder, a reference pictures decoder and a stream formatter. The frame encoder block carries out prediction and residual data encoding, whereas the reference picture decoder creates a complementary feedback path. The inter-frame and intra-frame predictors are common parts of the frame encoder and decoder. The stream formatter encodes data using variable length codes and creates stream accordingly to AVC stream syntax and semantics. The last part is a pre-encoding block. This module



(a) Intra prediction

(b) Inter prediction

Fig. 2. Inter and Intra prediction of image blocks in H.264/Avc.

performs analysis prior to image encoding and selects macroblock prediction mode that is used in the encoding process.

A. Image encoding modes

Frame encoding in AVC is an adaptive process that may be invoked in several modes and it uses context adaptive algorithms to achieve high compression ratio. The AVC standard defines a set of tools that allow for reducing data redundancy significantly e.g. intra-frame (intra) and inter-frame (inter) prediction - similarly to the other hybrid-encoders. For each picture one of this methods is invoked.

In the intra (spatial) prediction mode samples of a macroblock are predicted from its neighboring samples of processed macroblock (Fig. 2(a)). In inter (temporal) mode all values are predicted from samples of reference frames (Fig. 2(b)).

Additionally, the AVC standard defines two types of intra prediction: 4x4 block and 16x16 macroblock prediction. Moreover, for each type several directions of prediction are defined: 9 directions for 4x4 prediction and 4 for 16x16 type. In the case of inter prediction four main modes are defined: $Inter_{16x16}$, $Inter_{16x8}$, $Inter_{8x16}$, $Inter_{8x8}$, however, the $Inter_{8x8}$ predicted block can be split into up to four independent 4x4 subblocks.

The encoder must select a proper prediction mode in order to achieve the required compression gain and the quality of resulting video. The most reliable method that allows for choosing optimal encoding scheme is to perform full encoding for all possible prediction modes. Finally, the best macroblock prediction scheme that ensures very good compression ratio and low image distortion is chosen. This is connected with rate control. The AVC encoder can perform in one of two available modes - constant bitrate mode (CBR) or variable bitrate mode (VBR; constant quality).

B. Cost estimation and compression efficiency

In order to achieve the best possible encoding gain a proper value of quantization parameter and a prediction mode need to be established. Hence, an additional very complex analysis has to be performed in the encoder. Due to high computational complexity of compression tools an implementation of analysis algorithm would require very high clock speed and considerable hardware area.

An effective method of prediction mode selection for each macroblock uses Lagrangian optimization [7], [8]. This method is based on a cost function. The encoding cost depends on a macroblock content and it is determined by compression efficiency and image distortion. The cost function is described by the following equation [9]:

$$Cost = D_{REC} + \lambda \cdot R_{REC} \qquad (1)$$

where D_{REC} denotes distortion and R_{REC} is a value that denotes a number of bits necessary to encode all required data.

The selection process resolves into the minimization of cost function (1). Depending on algorithm used each part of the above sum may be defined in a different way. The image distortion may be described by a sum of squared differences (SSD) between original and reconstructed image samples(2).

$$SSD = \sum_i e_i^2 \qquad (2)$$

where e_i is defined as follows:

$$e_i = s_{x,y,t} - \hat{s}_{x,y,t} \qquad (3)$$

and $s_{x,y,t}$ is an original image sample and $\hat{s}_{x,y,t}$ is a reconstructed image sample. Distortion can be also calculated with the use of a sum of absolute differences (SAD)

$$SAD = \sum_i |e_i| \qquad (4)$$

where e_i is defined in (3). This algorithm is much less complex and enables to obtain similar results therefore it seems to be more adequate. However, the rate-dependant part of the cost function requires additional computing and motion vectors' contribution must be taken into consideration.

An alternative and effective method of cost estimation calculates the energy proportional ratio for the prediction error signal. The energy computation is based on a discrete Hadamard transformation (HT) [10]. This method allows for approximating rate-dependant constraint R_{REC} as well and its complexity is not very high.

III. A BIT-SERIAL ARCHITECTURE PROPOSAL

The proposed architecture for coder control unit (pre-encoder block) is shown in Fig. 3. The presented unit implements encoding cost estimation and compression scheme selection algorithm that exploits Hadamard transformation. Inputs of this module are original image samples (org) and predicted values (pred) of any available prediction mode (Intra4x4, Intra8x8, Intra1x16 or Inter). The proposed encoder control block was implemented in the bit-serial arithmetic.

Processing is carried out for each macroblock in two stages: a macroblock encoding cost calculation (based on 4x4 block processing) and subsequently, a selection of the best-effort prediction modes for the entire macroblock. The first stage of computations is performed with the use of a set of analysis blocks shown in in Fig. 4. Input values of the presented module are samples of original image and predicted samples. This values are used to compute prediction error signal. The resulting error samples are processed in 16-point two-sdimensional Hadamard transformation unit. A 1-D Hadamard

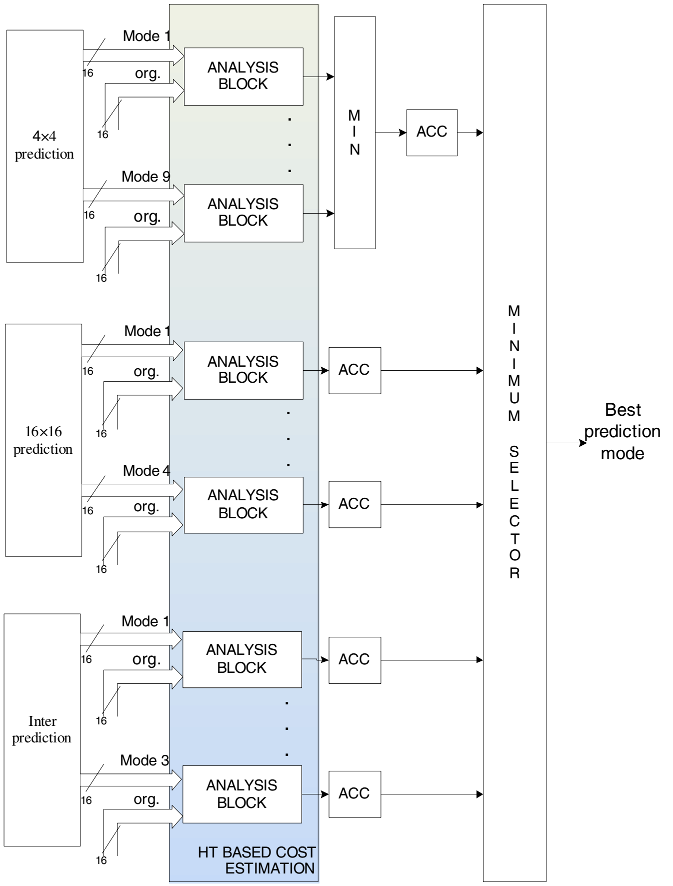


Fig. 3. An architecture proposal for Hadamard transformation based coder control in H.264/AVC encoder (ACC-accumulator, org - original image samples, $Mode_i$ - prediction samples).

transformation unit is shown in Fig. 5(a). The output values of transformation block are then used to calculate the sum of absolute values (SAD). The absolute value calculation module is shown in Fig. 5(b). It exploits a delay path and performs negation of a value depending on the sign bit. The resulting values for the whole macroblock and each prediction mode are accumulated and finally the decision is made and the best prediction mode (minimizing cost function 1) is selected. However, in the case of Intra 4x4 block prediction an additional decision process takes place. For each single 4x4 block within a macroblock the best prediction direction is chosen and in this way local minimization of the cost function is preformed. The analysis is carried out in 4x4 blocks and it involves units of reconstruction block of the decoder [6] (predictors and encoder loop path).

The analysis block shown in Fig. 4 is replicated for each analyzed prediction type to enable simultaneous calculations for all modes.

The encoder block presented in Fig. 1 uses the selected prediction mode to perform final prediction of coded macroblock. At the same time analysis for the next macroblock starts.

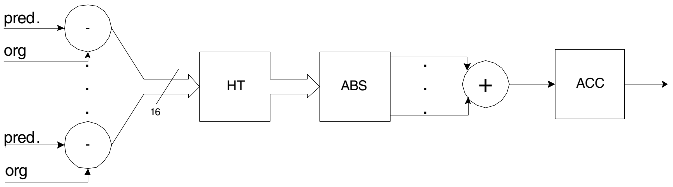
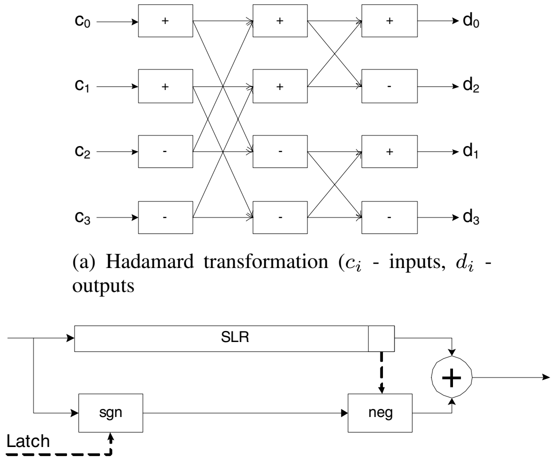


Fig. 4. Analysis block architecture (HT-Hadamard transformation module, ABS - absolute value block, ACC-accumulator).



(a) Hadamard transformation (c_i - inputs, d_i - outputs

(b) Absolute value computation unit(sgn- sign buffer, SLR - shift register, neg - negation)

Fig. 5. Analysis block components.

IV. Computations efficiency and simulation results

In the figure 6 a performance analysis for the macroblock processing is shown.

Let us assume that the analysis process starts at time zero. After two clock cycles delay the Hadamard transformation computation is invoked. The transformation block output values are 13 bits long, thus the absolute value computation requires 25 clock cycles. The processing of 4x4 image blocks is pipelined, however, processing of the next block must be delayed for 21 clock cycles, due to complementary operations. The computation time for all partial operations is shown in tab. I.

Computation time for a single 4x4 image block requires 36 clock cycles (please refer to Fig. 6). However, by exploiting

TABLE I

Processing time for a single 4x4 image block.

Module name	Processing time for a singe block [clock cycles]
Predictors	10
Hadamard transformation	13
Absolute value	12 (buffering) + 13
Final analysis and accumulation	21

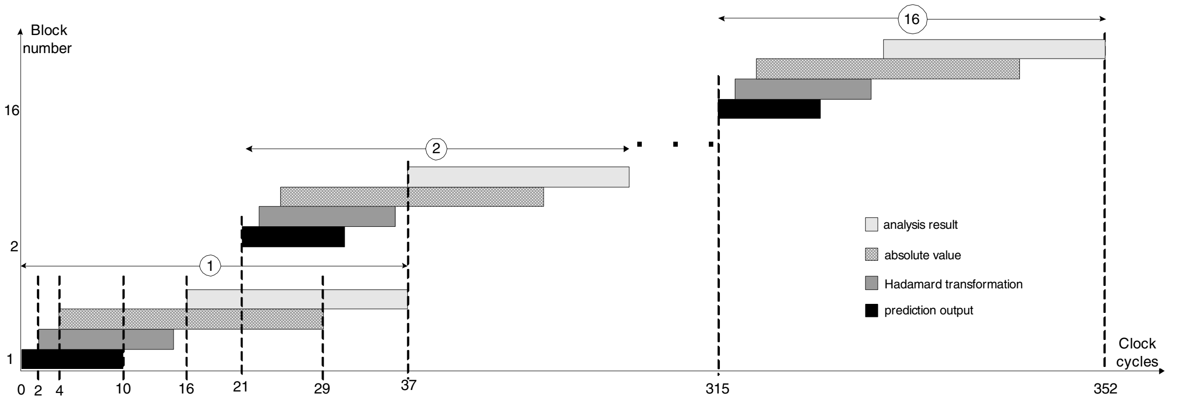


Fig. 6. The analysis block performance.

TABLE II
SYNTHESIS RESULTS OBTAINED FOR XILINX VIREX II DEVICE
(SYNTHESIS TOOL: ISE 6.3)

Block name	Area occupancy [FPGA slices]	Maximal frequency [MHz]
Hadamard transformation	89	221.12
Difference computation unit	16	
Absolute value computation unit	1	
Output adder	30	
Accumulator	18	
Analysis block	154	165.65
Coder control block	2710	160.2

pipelining the efficiency of the whole image macroblock processing time is very high. An analysis for the whole macroblock requires 21 single processing stages to be performed. A single cycle requires 37 clock cycles and the delay in the pipe equals 21, therefore the processing time for the whole macroblock (256 samples) is only $15 \cdot 21 + 37 = 352$ clock cycles. This results in processing efficiency equal about 1.37 clock cycle per sample. The processing without pipelining would require $16 \cdot 37 = 592$ clock cycles.

The analysis block was synthesized on Xilinx VirtexII family device and the synthesis results obtained in Xilinx ISE 6.3 are presented in tab. II. A single analysis block requires 154 FPGA slices and is capable to perform at clock speed equal 165.92 MHz. The whole control block area occupancy is about 2710 FPGA slices and the maximal frequency equals 160.2 MHz.

V. CONCLUSION

In this paper a new efficient architecture for AVC encoder control has been proposed. The modules have been implemented with the use of the bit-serial arithmetic. The presented architecture is fully pipelined and has a very regular structure. It is composed of 16 identical bit-serial processing pipes. The design can be easily implemented in ASIC and FPGA devices. Moreover, in the case of FPGA implementation it is effectively mapped onto FPGA configurable logic blocks (CLB).

In the proposed analysis scheme marcoblocks are processed by turns and the input image can be processed at clock frequency close to the sampling frequency (about 13MHz). This ensures low power consumption, hence, the design is suitable for mobile devices and other application where low power consumption is crucial.

It has been shown also that bit-serial architectures using pipelining allow for increasing significantly the processing efficiency of video compression algorithms implementations.

ACKNOWLEDGMENT

This work was supported by the public founds as a research project.

REFERENCES

[1] ISO/IEC JTC1/SC29/WG11, *ISO/IEC 14496 10 Advanced Video Coding*, 3rd ed., Redmond, July 2003.

[2] T. Wiegand, G. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the h.264/avc video coding standard.", *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, July 2003.

[3] M. Horovitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/avc baseline profile decoder complexity analysis", *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, pp. 704–716, July 2003.

[4] V. Lappalainen, A. Hallapuro, and T. Hämäläinen, "Complexity of optimized h.26l video decoder implementation", *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, pp. 717–725, July 2003.

[5] S. Wang, W. Peng, Y. He, G. Lin, and S. Chang, "A software-hardware co-implementation of mpeg4 advanced video coding (avc) decoder with block level pipelining", *Journal of VLSI Processing, Kluver*, 2003.

[6] P. Garstecki and A. Łuczak, "A flexible architecture for image reconstruction in h.264/avc decoders", in *Proc. of ECCTD 2005*, Cork, Ireland, August 2005.

[7] T. Wiegand and G. Sullivan, "Rate distortion optimization for video compression.", *IEEE Signal Processing Magazine*, vol. 15, pp. 74–90, November 1998.

[8] T. Wiegand and B. Girod, "Langrangian multiplier selection in hybrid video coder control.", in *Proc. of ICIP 2001*, Thessaloniki, Greece, October 2001.

[9] T. Wiegand, H. Schwarz, A. Koch, F. Kossentini and G. Sullivan, "Rate constrained coder control and comparison of video coding standards.", *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, pp. 688–704, July 2003.

[10] JVT of ISO/IEC MPEG and ITU-T VCEG, *"Text Description of Joint Model Reference Encoding Methods and Decoding Concealment Methods - ISO/IEC 14496-10 AVC amd.6"*, Hong Kong, January 2005.