A Flexible Architecture for Image Reconstruction in H.264/AVC Decoders

Adam Łuczak * Paweł Garstecki *

Abstract — **The H.264/AVC is the most recent standard of video compression. In this paper original and flexible architecture of image reconstruction block for H.264/AVC decoder is presented. Depending on application requirements the proposed design may be configured as low complexity simple unit with good performance or as fully pipelined construction with high performance. The architecture was implemented in *Verilog* HDL and synthesized and then tested on Xilinx *VirtexII* family device. The simulation results indicate that the implemented circuit is capable to process real-time video at clock close to the image sampling frequency.**

1 INTRODUCTION

The most recent method for efficient video compression provided by H.264/AVC standard [1, 2] significantly outperforms any previous technique in bitrate reduction. The high compression efficiency is achieved at the cost of high computational complexity [3]. The most popular are software implementations of AVC decoder, however there are known some software-hardware co-implementations as well [6].

In the case of software realizations processors require a few cycles to perform even simple stream operations due to generality of their architecture that is designed to support a wide range of applications. Therefore, high clock frequencies are required in order to decode video bitstream in real time. Then higher clock frequency results in increased power consumption which can be an obstacle for some areas of application e.g. mobile devices. A better solution for such appliances is efficient hardware implementation of AVC decoder, but pure hardware implementation is impeded by sophisticated compression algorithm with much irregularity within.

One of the most complex parts of decoding process is image reconstruction. In the case of software realizations it requires about 50% of computational power [4]. In this paper a flexible architecture for image reconstruction block which consists of inverse intra prediction and inverse transformation will be presented. Depending on user requirements this architecture can be configured to either high performance or extremely simple construction with small silicon area.

*Division of Multimedia Telecommunications and Radioelectronics , Poznan University of Technology, Poland, e-mail: [aluczak,pgarstec]@multimedia.edu.pl, tel.: +48 61 6652870, fax: +48 61 6652292.

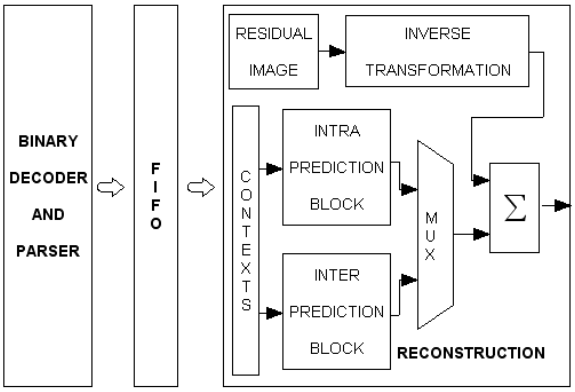

Figure 1: AVC decoder block diagram.

In order to achieve higher efficiency the proposed AVC decoder architecture was implemented as two separate modules: bitstream parser and reconstruction. The block diagram of the decoder is shown in fig. 1. This separation resulted in simplicity of the architecture of both modules and very low clock frequency. As an advantage of such division contexts (except few flags) are not used by parser thus they can be passed directly to the reconstruction part of the decoder. This paper presents the reconstruction part of the decoder only.

The proposed design uses bit-serial arithmetic to implement algorithms provided by the standard. In this paper it will be shown that such approach results in simplicity of the architecture and enables pipelining, which allows to reduce the total number of clock cycles required to perform macroblock reconstruction. However, serial approach involves special care for synchronization management due to various computational delays of different modules.

2 IMAGE RECONSTRUCTION

The block diagram of reconstruction block for AVC decoder is shown in fig. 1. This part of the decoder performs inverse transformation and prediction. The AVC standard defines three types of image prediction - Intra_4x4, Intra_16x16 and Inter.

The Inter prediction is not a part of this paper. Within one macroblock only one type of prediction may be applied.

2.1 Intra prediction

Intra prediction [1, 2] can be invoked for both – luminance and chrominance macroblocks. The samples' values are predicted from context which is previously reconstructed left, above and above-right spatially neighboring samples $L, K, \ldots I, X, A, \ldots, G, H$ of the same frame (fig. 2) in $Intra_{4x4}$ or $Intra_{16x16}$ modes. Prediction type selection is conditioned by the content of processed block. The Intra_4x4 pattern is suitable for more detailed areas and each 4x4 image component is predicted separately in this mode. For smooth areas the Intra_16x16 mode is more appropriate and all blocks of the luma macroblock are simultaneously predicted when applying it.

For the 4x4 luminance blocks, nine directions has been defined, including vertical, horizontal and DC prediction. In horizontal and vertical modes samples are predicted directly from context values, whereas each pixel from a block is estimated by weighted mean of up to three neighboring context samples when more sophisticated procedure is selected. In the case of $Intra_{16x16}$ prediction type four modes have been defined . Three of them (horizontal, vertical and DC) are very similar to those of $Intra_{4x4}$ except that for $Intra_{16x16}$ all 32 contexts are used. Additionally, plane prediction mode has been specified. In this case sample values are derived from a contexts-dependent equation of a plane [1].

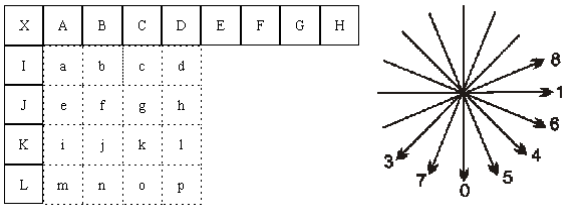

Figure 2: Intra 4x4 prediction block context and directions (a, ..., p - block samples, L, ..., H - context samples).

2.2 Inverse transformation

Transformation is used to convert samples from spatial to frequency domain. The AVC uses 4x4 integer transformation [1, 2, 5], similar to the 4x4 DCT. The decoder reverses this process by performing inverse transformation. The 1-dimensional

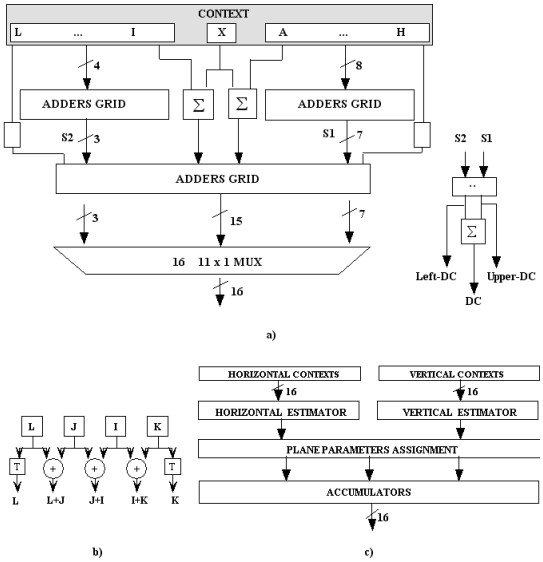

Figure 3: Intra 4x4 predictor architecture (a), adders grid (b) and plane predictor (c).

transformation is described by the following matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix}$$

This conversion is separable and can be performed in two steps - vertically and horizontally. Fast one dimensional algorithm requires seven additions and two binary shifts.

3 ARCHITECTURE

The proposed architecture uses bit-serial algorithms to implement methods defined by the AVC standard. The basic unit of the proposed design is a 4x4 reconstruction block that performs intra prediction and transformation of 4x4 image component. The prediction process is simultaneously performed with inverse transformation and a reconstructed image sample is a sum of both results.

The prediction module consists of three separate units: intra predictor, plane predictor and transformation block and block diagram of this entity is shown in fig. 3a. The main part of it is a full adders and delays grid which allows computing weighted sum of samples. Each of sixteen output values may be chosen from among nine of all generated sums (including DC) or two context values (in horizontal/vertical mode). This selection is provided by sixteen 11 to 1 binary multiplexers driven by prediction direction and output sample location. The inputs of this multiplexers are predicted plane and

DC mode values and $Intra_{16x16}$ and $Intra_{4x4}$ samples (all directions).

A separate entity of intra prediction unit is a plane predictor. This module assigns plane coefficients on the basis of which it predicts image samples. The block diagram of this module is shown in fig. 3c. The H and V elements compute weighted sum of sample differences which constitutes input to the plane parameters assignment block. This values are used to estimate image samples which are provided as simple accumulation of sample coordinates-derived values, which do not depend on actual context.

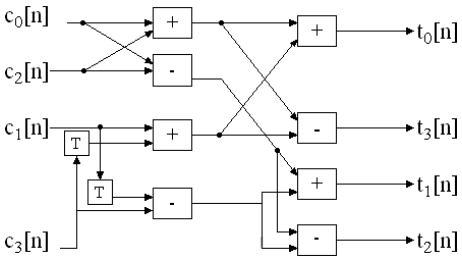

Figure 4: 1-D Integer Transform with serial arithmetic(c_i - input coefficients, t_i - output samples, T - delay).

The remaining element of presented architecture is inverse transformation block. The transformation process is performed first horizontally and then vertically and each step is realized by a single 1-D transform module. The architecture used to perform 1-D inverse transformation is shown in fig. 4. The module implemented with serial arithmetic requires only nine 1-bit adders and two 1-bit delays. Figure 5 presents 2-D transformation module construction based on 1-D blocks. This module performs transposition of samples as well. It should be noticed that in such implementation transposition is just suitable routing of wires connecting vertical (VT^{-1}) and horizontal (HT^{-1}) entities of the transformation block.

4 PERFORMANCE ANALYSIS

The presented architecture was implemented in *Verilog* hardware description language. Each prediction and transformation block requires about 160 cells gates when using bit-serial full adders. The comparison of area occupancy indispensable to implement transformation block in various arithmetic are presented in table 1. The obtained number of slices is a result of synthesis various architectures for VIRTEX II family devices. The proposed serial configuration requires 14 clock cycles

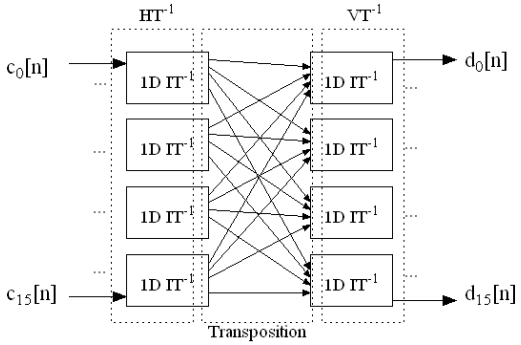

Figure 5: Block of 2-D Integer Transform (c_i - input transformation coefficients, d_i - output image samples).

to commit requested operations and the computational delay equals 6 cycles.

Architecture used	FPGA area
serial	86
distributed	128
parallel	368

Table 1: Number of slices necessary to implement inverse transform in various architectures (results obtained for VIRTEX II.

The main advantage of such architecture is that the proposed image reconstruction module may be configured in a few ways, depending on user requirements or application purpose. The reconstruction block may include one or up to four 4x4 modules. Each single module performs intra prediction, inverse quantization and transformation.

Some possible arrangements are presented in figure 6. The 4x4 reconstruction module reflects 4x4 block of luma macroblock component, thus output values of one entity are input contexts of other suitable elements. The upper contexts are stored in shift registers which are updated during processing. The computational delay (6 cycles) resulting from data processing by the transformation module is utilized to update context values.

Table 2 presents simulation results for reconstruction time of a single line and a whole macroblock for different configurations.

The most slowly performing configuration contains a single 4x4 reconstruction module and it requires 14 clock cycles per each 4x4 picture block. When using four modules, the reconstruction of one line (4x16 samples) is completed within 32 and of the whole macroblock within 76 clock cycles. This results in gain of 148 clock cycles per each macroblock. The table contains processing time per

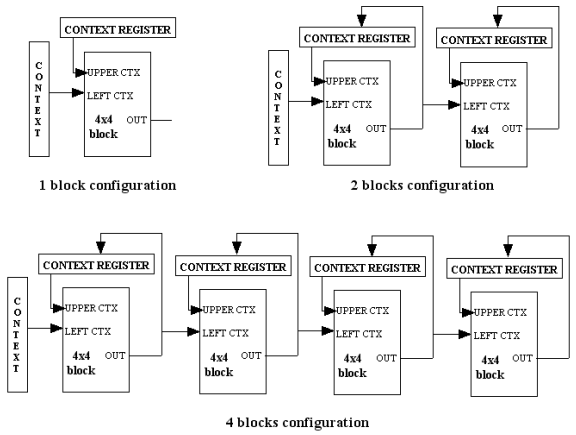

Figure 6: Reconstruction block configurations.

Number of 4x4 modules	Processing time [clock cycles]		
	4x4 blocks row of macroblock	macroblock (16 x 16)	per sample
1	56	224	0.875
2	40	160	0.625
4	32	74	0.289

Table 2: Decoding time for different configurations of reconstruction block.

one 8-bit image sample as well.

The result of such a module construction is pipelining which allows achieving high cycles reduction. In figure 7 processing chart for four block configuration is presented. Every 4x4 reconstruction module performs within fourteen clock cycles. Every next module starts to process data after six cycles delay.

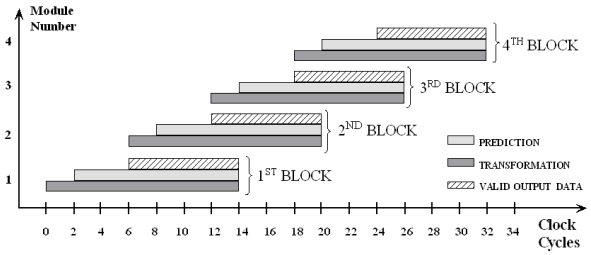

Figure 7: Reconstruction module performance (configuration with four 4x4 modules).

5 CONCLUSIONS AND FUTURE WORK

Area efficient and flexible hardware implementation of AVC image reconstruction block has been presented in this paper. It has been shown that serial arithmetic can be successfully adopted in devices which are requested to perform efficiently and have simple construction. The efficiency of proposed modules can be easily scaled depending on user/application requirements. In order to achieve higher computational efficiency of the whole reconstruction module the processing pipe has to be expanded by additional 4x4 reconstruction blocks.

Even the simplest configuration composed of a single 4x4 reconstruction module allows decoding SDTV video in real time.

The authors encouraged by good results of use of serial-arithmetic in video reconstruction plan to adopt this approach to motion estimation algorithm.

Acknowledgments

The research was provided as a part of a scientific project partially financially supported by The Ministry of Science and Information Society Technologies.

References

[1] ISO/IEC JTC 1/SC 29/WG 11 "ISO/IEC 14496 10 Advanced Video Coding 3^{rd} edition", Redmond, July 2004.

[2] T. Weigand, G.J. Sullivan, G. Bjøntegaard, A. Luthura, "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.D. 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] H.S. Malvar, A. HAllapuro, M. Karczewicz and L. Kerofsky, "Low-Complexity Transform and Quantization in H.264/AVC", IEEE. Trans. on Circuits and Systems for Video Technology, vol. 13, pp. 598–603, July 2003.

[6] S.H. Wang, W.H. Peng, Y. He, G.Y. Lin, S.C. Chang, C.N. Wang and T. Chiang, "A Software-Hardware Co-Implementation of MPEG4 Advanced Video Coding (AVC) Decoder with Block Level Pipelining", Journal of VLSI Processing Systems, Kluver 2003.