Optimized architectures of CABAC codec for IA-32- , DSP- and FPGA-based platforms

Damian Karwowski, Marek Domański

Poznan University of Technology,
Chair of Multimedia Telecommunications and Microelectronics
dkarwow@multimedia.edu.pl
domanski@et.put.poznan.pl

Summary

Two optimized architectures of Context-based Adaptive Binary Arithmetic Coding (CABAC) are presented in the paper. These are: the software version of CABAC (dedicated to IA-32 and DSP platforms) and the hardware version of CABAC (dedicated to FPGA and ASIC platforms). The paper presents analysis of implementations for both versions of CABAC. The optimized software as well as the hardware version of CABAC was tested in terms of codec throughput.

1. Introduction

The paper concerns advanced adaptive entropy coding in contemporary hybrid video encoders. The state-of-the-art entropy coding algorithm used in video compression is the Context-based Adaptive Binary Arithmetic Coding (CABAC) [1, 3] that has been used in the new Advanced Video Coding (AVC) standard (ISO MPEG-4 AVC and ITU-T Rec. H.264) [1].

CABAC algorithm is distinguished by the highest compression ratio among entropy encoders used in video compression [3], but it was obtained at the cost of high computational complexity. Therefore, real-time processing of high quality video with CABAC algorithm is a very difficult problem even for today's high-performance digital media processors.

An important research problem is to propose the efficient architecture for CABAC codec with high computational performance. Two such architectures of CABAC are presented in this paper: the software version of CABAC and the hardware version of CABAC. These architectures were already presented in [12]. This paper makes continuation of that research and presents new no published yet experimental results on throughput of entropy codec.

2. CABAC entropy coding algorithm

CABAC technique represents the class of context-based adaptive arithmetic coding methods. In terms of coding efficiency, CABAC is the most powerful entropy coding technique used in compression of digital video. For that reason it has been adopted to the new worldwide video compression standard MPEG-4 AVC/H.264 [1] as an alternative (besides the variable-length coding) technique of entropy coding.

The block diagram of CABAC encoder has been presented in Figure 1.

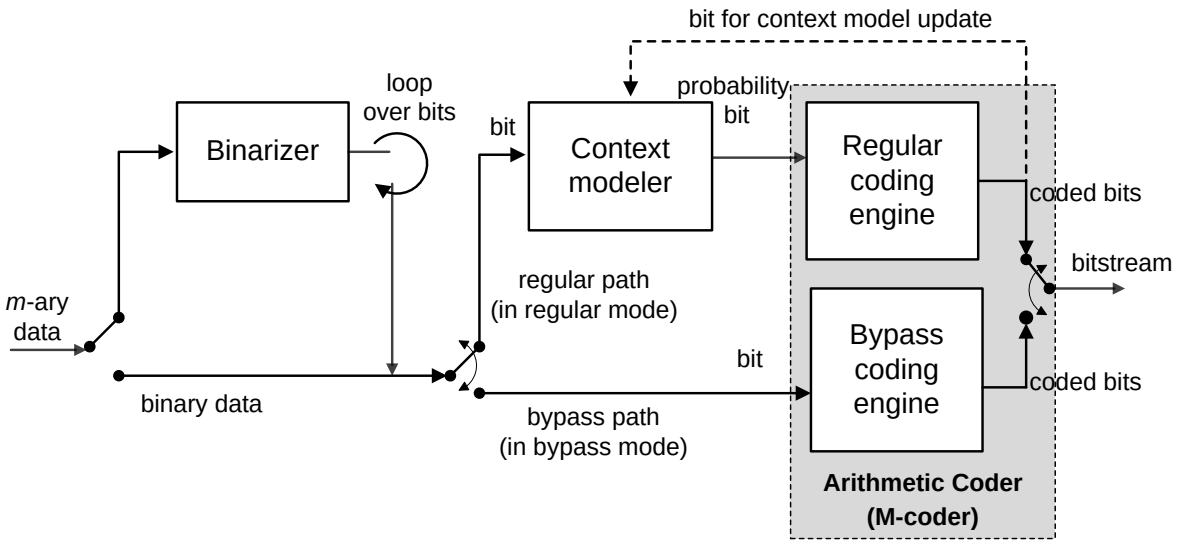

Figure 1. Block diagram of CABAC encoder.

There are three main functional blocks to be distinguished in CABAC. These are: binarizer of m-ary data, context modeler and the core of arithmetic codec [1, 3]. CABAC uses binary arithmetic coding, instead of m-ary arithmetic coding. In order to decrease the computational complexity of both the CABAC encoder and decoder, fast implementation of multiplication- and division-free binary arithmetic codec (the so-called M-codec) has been used [1, 3, 8, 9]. Due to application of binary arithmetic codec (instead of m-ary arithmetic codec) all non-binary valued data must be translated into sequence of binary symbols. This is realized by the binarizer at the first stage of coding. The binarization process has a huge impact on the number of binary symbols that are fed to binary arithmetic codec, which strongly influences the size of bitstream at the output of entropy codec as well as entropy codec complexity. Taking this into account, the binarization in CABAC has been adapted to statistics of m-ary data by the application of five different basic binarization schemes for coded syntax elements. These are: unary binarization, truncated unary binarization, k-th order Exp-Golomb binarization, fixed-length binarization and Huffman-based binarization [3]. The idea of binarization is very similar to variable-length coding (e. g. Huffman coding), however CABAC additionally uses efficient adaptive arithmetic coding in order to extra reduce the existing inter-binary symbol redundancy.

After the binarization, the resulted binary symbols are fed to arithmetic codec core that processes the input symbols according to the conditional probability of their occurrence in data stream. The conditional probabilities of binary symbols are estimated by the context modeler. The way of calculating these probabilities strongly influences the compression ratio of entropy encoder. Therefore, in order to accurately estimate the probabilities of symbols, total number of 399 pre-defined finite-state machines (FSM) has been defined in CABAC (this is only for the case of 4x4 transform), for all coded syntax elements, as presented in Figure 2. Such FSMs are referred to statistical models in the paper.

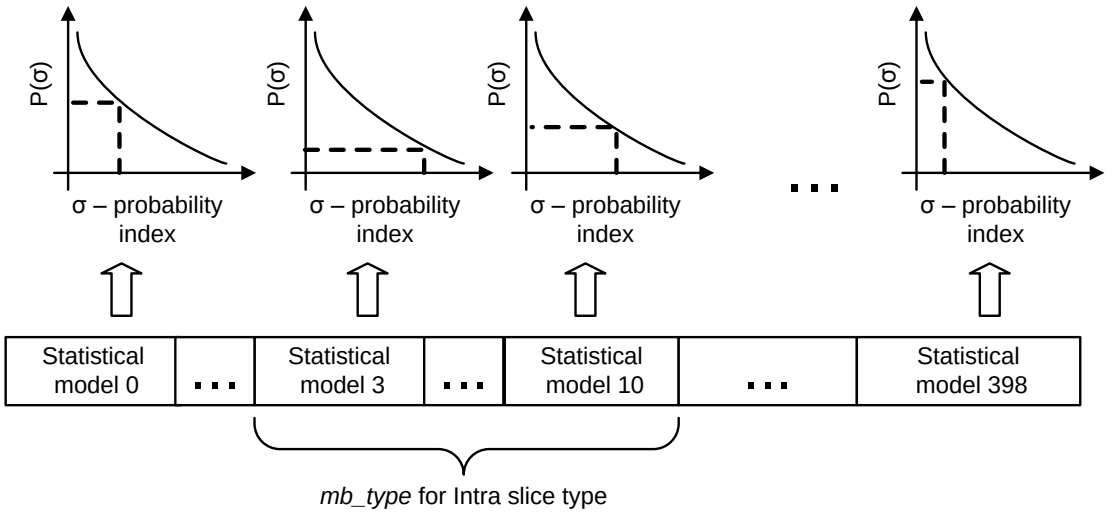

Figure 2. Statistical models in CABAC.

In CABAC, a given syntax element uses some sub-set of all 399 statistical models. The sub-sets of probability models for individual syntax element have been defined with respect to the statistics of the binary symbols in the binarized word. By coding of the binary symbol, a proper statistical model is chosen based on the values of syntax elements in the neighboring blocks (the left and the upper block in relation to the current block). This way, CABAC realizes two levels of adaptation to the current signal statistics: based on the syntax elements and its values in neighboring blocks. Finally, the chosen FSM calculates the probability of binary symbol that appeared in a given context. This probability is further used by the core of arithmetic codec. It is worth noticing that the data statistics estimation used in CABAC belongs to the most sophisticated mechanisms used in entropy encoders in video compression.

3. Features of CABAC codec and the research problem

As stated above, CABAC algorithm is distinguished by sophisticated mechanisms of data statistics estimation with multi-levels adaptation to the current signal statistics. Undoubtedly, these data modeling mechanisms belong to the most powerful, in terms of efficiency, that have ever been applied in video compression.

However, high compression ratio of CABAC has been achieved at the cost of significant increase of amount of computations that are needed to process a binary symbol of data. CABAC represents the context-based coding. When processing a binary symbol, the context data from neighboring blocks within an image are necessary. The coding path in CABAC strongly depends on these neighboring context data. It makes the CABAC algorithm very irregular and difficult to implement and test. Besides, the context-based coding and the application of adaptive arithmetic coding in CABAC impose sequential processing of symbols. Thus, the serial nature of CABAC is the source of enormous difficulties in exploiting the powerful multi-core technology. Taking all these into account the CABAC codec is a bottleneck in the case of processing of high quality video. Moreover, the real-time processing of high quality video with CABAC is still a great challenge for today's powerful multimedia processors.

All the facts mentioned above lead to an important research problem which is to propose the optimized architecture of high-speed CABAC codec that will allow for real-time processing of high definition (HD) video. Two optimized architectures of CABAC are presented in the paper. These are the software version of CABAC (dedicated to IA-32 and

DSP processors) and the hardware version of CABAC (dedicated to FPGA and ASIC platforms).

4. Optimized software version of CABAC

In order to limit the problem of high complexity of entropy processing module in MPEG-4 AVC codec, the optimized software version of CABAC codec (encoder and decoder) has been proposed. In optimized version of CABAC codec, all the functional blocks have been implemented from scratch and fully algorithmically optimized towards speed. The implementation has been made in C programming language. As stated above, the block of data statistics modeling makes an essential part of CABAC that determines its coding efficiency and computational complexity to a large extent. In the data statistics modeling block, the way of management of local context data is extremely important from the complexity point of view of entropy codec. Referring to this, the optimized software CABAC takes advantage of local context data buffer that enables very fast access to neighboring context data. This approach speeds up computations significantly.

The optimized software CABAC codec contains about 5200 lines of C program code with only 380 lines of program code for binary arithmetic codec core. It means that the core of arithmetic codec is a very small part of contemporary adaptive entropy codecs (about 7% in the case of CABAC implementation). It confirms that data statistics modeling module makes up the fundamental part of CABAC (more than 90% of whole implementation). What is interesting, this is not only characteristic of the presented CABAC implementation. In other well known and available implementations of CABAC (implementation of CABAC in x264 video codec [10], implementation of CABAC in JM 10.2 reference software [2]) the contribution of arithmetic codec core implementation in whole CABAC is similar and come to 12% (see Figure 3)

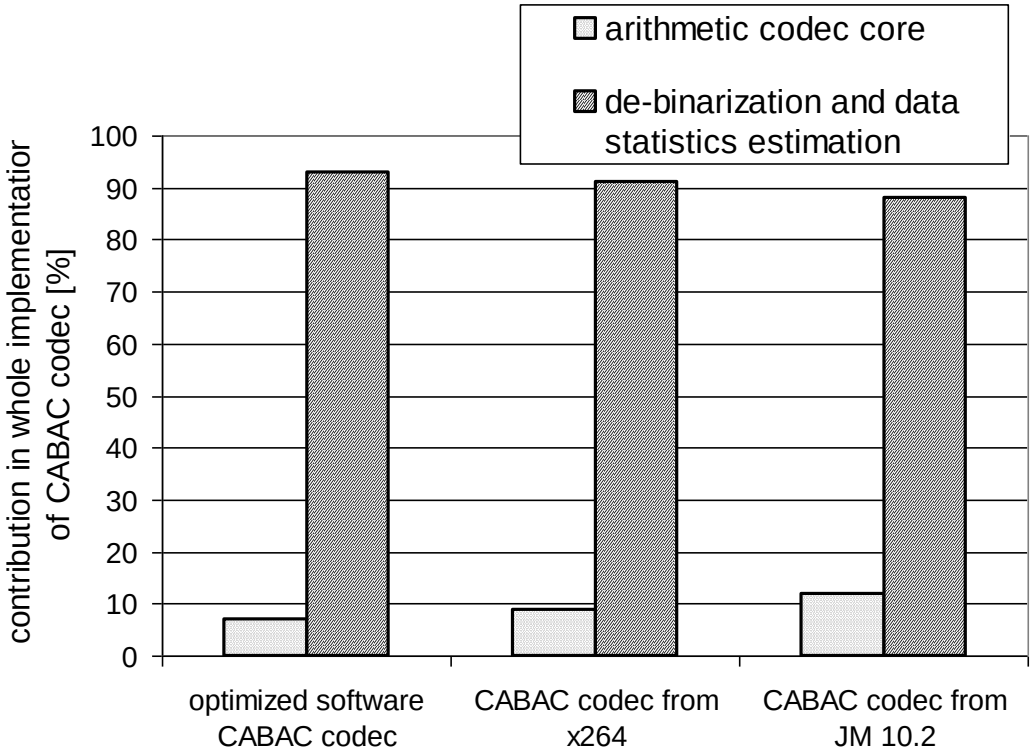

Figure 3. Contribution of arithmetic codec core in three implementations of CABAC.

The optimized software version of CABAC codec presented in this paper has been successfully activated in the original fast MPEG-4 AVC video decoder that had been elaborated in Poznań University of Technology in 2004 by multimedia team. The fast MPEG-4 AVC video decoder is highly algorithmically optimized for speed version of AVC decoder. The complexity of fast MPEG-4 AVC video decoder and the throughput of optimized software version of CABAC has been tested on two different platforms: IA-32 platform and DSP platform.

In the case of IA-32, tests have been done on Intel Core 2 Duo E6600 platform (2.4 GHz, 4MB of memory cache of Level 2) with 2 GB of RAM under the 32-bit Windows XP with Service Pack 2 operation system. Experiments have been done for the CITY, CREW, ICE, HARBOUR test video sequences (704x576 spatial resolution, 60 Hz of time resolution) for a wide range of target bitrates. Complexity of both the MPEG-4 AVC decoder and the CABAC decoder has been tested. Results are presented in Table 1.

Table 1. Complexity of fast MPEG-4 AVC decoder as well as the optimized CABAC decoder.

704x576 resolution seq.	Bitrate (kb/s)	Average total decoding time [ms/frame]	Average CABAC decoding time [ms/frame]	CABAC contribution in total decoding time [%]
	18385,44	39,02	14,57	37,34
	4136,96	26,69	4,26	15,96
	1228,59	23,7	2,01	8,48
City@60 Hz	499,99	20,26	1,42	7,01
	27569,97	37,60	21,75	57,85
	8482,42	24,59	8,22	33,43
	2488,56	18,16	3,44	18,94
Crew@60 Hz	949,46	14,95	2,17	14,52
	36606,40	48,59	26,16	53,84
	15861,45	35,29	13,16	37,29
	4850,05	26,32	5,00	19,00
Harbour@60 Hz	1692,99	22,28	2,54	11,40
	25437,44	32,14	19,06	59,30
	7466,42	17,19	6,43	37,41
	2739,93	13,38	3,21	23,99
Ice@60 Hz	1104,31	11,81	1,94	16,43

From experimental results it is clear that even algorithmically optimized CABAC decoder is a challenge for high speed IA-32 processors. In the case of 60 Hz sequences, the real-time CABAC decoding of bitstreams with bitrates up to approximately 20 Mb/s is possible.

In the case of DSP platform, experiments were done on high-performance TMS320DM642 digital signal processor (600 MHz clock frequency) [11]. Experiments revealed that real-time CABAC decoding is feasible only for bitstreams with bitrates below 10 Mb/s (for 30 Hz video sequence). What is interesting, the real-time MPEG-4 AVC decoding of bitstreams (with CABAC entropy decoder) was possible only for bitrates below 4 Mb/s. With regard to the CABAC decoder, about 75 clock cycles of the TMS320DM642 are needed to decode a binary symbol with optimized software CABAC, where about 60% of a binary symbol decoding time was needed for de-binarization and probability estimation. Thus, after the conditional probability estimation the actual arithmetic decoding absorbs about 40% of symbol decoding time.

5. Optimized hardware version of CABAC

Algorithmic optimizations that were done in the proposed software version of CABAC increase throughput of entropy codec. Nevertheless, real-time processing of high quality video with fast MPEG-4 AVC decoder and optimized CABAC is still very difficult. It is confirmed by experimental results presented in the previous section.

It is commonly known, that hardware platforms (Field Programmable Field Arrays - FPGA, Application Specific Integrated Circuit – ASIC) have significantly higher processing capabilities relative to IA-32 and DSP. It was the motivation to propose the hardware version of CABAC decoder.

A special, hardware version of CABAC decoder was designed with taking into consideration the features of hardware platforms. CABAC decoder was split into several modules that realize the following tasks: managing of local contexts, selecting the probability model, arithmetic decoding of binary symbols and de-binarizing the data. Because of several different binarization schemes used in CABAC, the task of de-binarization and control of syntax elements decoding was split into two modules: transform coefficient decoding and syntax elements decoding that operates the remaining syntax elements. The way of decoding and de-binarization of syntax elements was stored in ROM memory. The architecture of hardware version of CABAC decoder is presented in Figure 4.

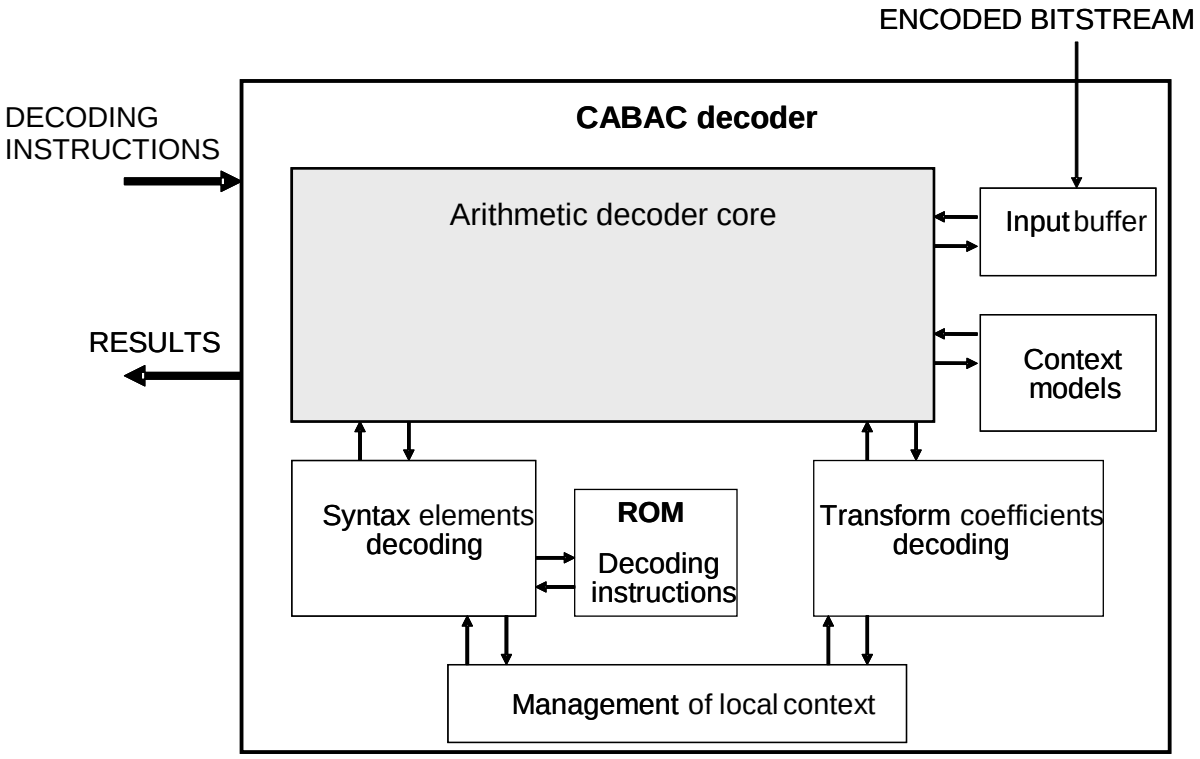

Figure 4. Architecture of hardware version of CABAC decoder.

The operating process of hardware CABAC decoder is the following. Entropy decoder receives a command of decoding the syntax element. ‘Management of local context’ module checks the values of a given syntax element in neighboring blocks and calculates the local context. Depending on the type of syntax element and its values in neighboring blocks (the local context) a proper probability model is chosen by the ‘Context models’ module. Having the conditional probability for a binary symbol, the ‘syntax elements decoding’ module and the ‘transform coefficients decoding’ module strobes the arithmetic decoder core and realizes

de-binarizing of data. The arithmetic decoder core works on encoded bitstream and realizes arithmetic decoding of data. After decoding results are sent to FIFO queue.

As it has been stated before, the serial nature of CABAC reduces significantly possibilities of doing computations in parallel. Nevertheless, proposed hardware architecture of CABAC decoder exploits some level of parallelism and task pipelining. The following computations can be partially done in parallel in optimized hardware CABAC decoder:

- Choosing the probability model;
- Renormalization of registers in arithmetic decoder core;
- Updating the probability model;
- Filling the input buffer with encoded bitstream.

After decoding the current binary symbol, operations for updating the probability model and computations for renormalization of registers in arithmetic decoder core are performed in parallel. Computations to select the probability model for the successive symbol are also started at the same time. After selecting the probability model for the new symbol, arithmetic decoding can be started if the registers renormalization process had been finished. In this way, by performing some of operations simultaneously in optimized entropy decoder, its computational performance has been significantly increased.

Presented hardware version of CABAC decoder has been implemented in the Verilog hardware description language (HDL) [4]. The optimized hardware CABAC decoder contains about 5500 lines of program code. It must be emphasized that the implementation of hardware CABAC was much more difficult and much more time-consuming, comparing to the software version of CABAC decoder (time needed to implement CABAC decoder in Verilog is about 2-3 times greater relative to the software CABAC). Hardware CABAC decoder was successfully synthesized on the high-performance Virtex series of FPGAs [5]: Virtex-2 Pro, Virtex-4, and Virtex-5 with Xilinx ISE software [6]. Numerical data that concerns the device occupation and clock frequency for CABAC decoder were presented in Table 2.

Table 2. Synthesis results for CABAC decoder.

FPGA device	Number of slices	Maximum clock frequency [MHz]
Virtex-2 Pro	2397	115
Virtex-4	2421	158
Virtex-5	1600	192
Virtex-6	1549	190

In practice, about 3 times higher clock frequency can be obtained when realizing the design as an application-specific integrated circuit (ASIC) in the same technology process [7]. It gives the maximum clock frequency of 600 MHz for the optimized hardware CABAC decoder.

Throughput of hardware CABAC decoder has also been measured. Tests were done on Virtex-5 platform with a set of a hundred thousands binary symbols. Experiments showed that hardware CABAC decoder processes a binary symbol in 7.5 clock cycles on average. It is a very good result comparing to 10 times greater number of clock cycles that is needed to decode a binary symbol with the optimized software CABAC working on high-performance TMS320DM642 signal processor. An ASIC device with hardware CABAC (and clock frequency of 600 MHz) enables real-time entropy processing of bitstreams with bitrates up to 80 Mb/s.

6. Conclusions

Two original architectures of CABAC entropy decoder were presented in the paper. These are: the software version of CABAC and the hardware version of CABAC. Both versions of CABAC decoder were algorithmically optimized towards speed with taking into consideration the features of target platform. Throughput of both versions of CABAC decoder was tested by performing series of experiments with test video sequences. Tests were done on IA-32, DSP and FPGA platforms. Maximum bitrates for encoded bitstreams that can be process with the proposed CABAC decoders in the real-time were pointed out in the paper. These are: bitrates up to 9-10 Mb/s for the software version of CABAC (at 600 MHz clock frequency) and up to 80 Mb/s for the hardware CABAC (at 600 MHz clock frequency) for 30 Hz time resolution of video sequence.

The paper revealed that the hardware version of CABAC is characterized by significantly higher throughput relative to the software version of CABAC. This is very important result of this paper. The number of clock cycles that are needed to process a binary symbol with the hardware CABAC decoder is about 10 times smaller in comparison to the software CABAC decoder.

References

1. ISO/IEC 14496-10, "Generic Coding of Audio-Visual Objects, Part 10: Advanced Video Coding", March 2006.

2. H.264/AVC software coordination site - http://bs.hhi.de/~suehring/tml.

3. D. Marpe, H. Schwarz, and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 7, pp. 620-636, July 2003.

4. IEEE Standard Hardware Description Language Based on the Verilog Hardware Description Language, IEEE Std 1364-1995, IEEE, 1995.

5. Virtex-5 FPGA Family Datasheet. Xilinx, Inc., San Jose, CA, 2007, http://www.xilinx.com/.

6. Xilinx ISE 9.2i software manuals, http://www.xilinx.com/.

7. I. Kuon, and J. Rose, "Measuring the Gap Between FPGAs and ASICs". IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., Vol. 26, No. 2, p. 203, February 2007.

8. D. Marpe, and T. Wiegand, "A highly Ecient Multiplication-Free Binary Arithmetic Coder and Its Application in Video Coding", IEEE International Conference on Image Processing (ICIP'03), Barcelona, Spain, September 2003.

9. D. Marpe, G. Marten, and H. L. Cycon, "A Fast Renormalization Technique for H.264/MPEG4-AVC Arithmetic Coding", 51st Internationales Wissenschaftliches Kolloquium Technische Universitat Ilmenau, September 2006.

10. x264 Project. http://www.videolan.org/developers/x264.html.

11. Texas Instrument, TMS320DM642 Video/Imaging Fixed-Point Digital Signal Processor. www.ti.com, July, 2002.

12. D. Karwowski, M. Domański, “High speed CABAC codecs for DSP- and FPGA-based platforms”, 1st International Conference on Image Processing & Communications, Bydgoszcz, Poland, September 2009.