On Implementation of MPEG-2 like Real-Time Parallel Media Applications on MDSP SoC Cradle Architecture

Ganesh Yadav[1], R. K. Singh[2], and Vipin Chaudhary[1]

[1] Dept. of Computer Science, Wayne State University
{ganesh@cs.,vipin@}wayne.edu
[2] Cradle Technologies, rks@cradle.com

Abstract. In this paper we highlight the suitability of MDSP [3] architecture to exploit the data, algorithmic, and pipeline parallelism offered by video processing algorithms like the MPEG-2 for real-time performance. Most existing implementations extract either data or pipeline parallelism along with Instruction Level Parallelism (ILP) in their implementations. We discuss the design of MP@ML decoding system on shared memory MDSP platform and give insights on building larger systems like HDTV. We also highlight how the processor scalability is exploited. Software implementation of video decompression algorithms provides flexibility, but at the cost of being CPU intensive. Hardware implementations have a large development cycle and current VLIW dsp architectures are less flexible. MDSP platform offered us the flexibilty to design a system which could scale from four MSPs (Media Stream Processor is a logical cluster of one RISC and two DSP processors) to eight MSPs and build a single-chip solution including the IO interfaces for video/audio output. The system has been tested on CRA2003 board. Specific contributions include the multiple VLD algorithm and other heuristic approaches like early-termination IDCT for fast video decoding.

1 Introduction

Software programmable SoC architectures eliminate the need for designing dedicated hardware accelerators for each standard we want to work with. With the rapid evolution of standards like MPEG-2, MPEG-4, H.264, etc. such programmable systems are desirable. Building hardware accelerators for the new upcoming standards like H.264 becomes time critical if the evolution time between two successive standards is small e.g. MPEG-4 and H.264. The ability to implement these algorithms in software has many advantages: it is less expensive and more flexible for accommodating new algorithms and enhancements as they evolve. In order to meet the demands of higher-quality video applications, an SoC must provide, in addition to a high level of arithmetic processing power, a

[3] MDSP is the trademark of Cradle Technologies

sufficient degree of flexibility, integrate a powerful on-chip communication structure, and employ a well-balanced memory system to account for the growing amount of data to be handled.

On general purpose processors, decoder implementations are usually memory bottlenecked. An extension of a programmable core with dedicated modules, e.g. Trimedia [30], does not help when the functions that have been hard-wired change in a new version of a multimedia standard. Our solution combines both the data and algorithmic parallelization approach. We apply agreedy strategy to exploit performance. We apply static scheduling within MSP and use the co-worker model for dynamic scheduling of tasks on MSPs. Thus we primarily rely on spatial decomposition and then further exploit algorithmic parallelism. Software implementation allowed us to plug-in new intellectual contributions (like multiple VLD, faster IDCT; explained below) to make the implementation faster.

1.1 Contributions

Multiple VLD: We consider Multiple VLD as our most valuable contribution to the field. The multiple VLD algorithm is discussed in section 4.2. Verderber et. al. [28] reports a lookup table based VLD structure proposed by Lei and Sun [29] as the fastest known VLD decoder. We could improve this VLD algorithm further by 25-30 percent by our Multiple VLD algorithm. Our modification adds the capability to decode each codeword in a single cycle as well as multiple VLD symbols in a single access whenever allowed by the bit stream.

Early-termination IDCT: This idea is an offshoot of the MPEG-4 AC-DC prediction. The current block's first row or column coefficients are predicted based on the gradient of the higher DC value either in column or row direction. This serves as indication of the possibly flat areas in the horizontal or vertical direction. When gradient across columns is higher for the previous blocks, we do the row-wise 1D_IDCT first. This helps in termination of some of the IDCT calculations. The column-wise 1D_IDCT is done as a normal procedure. When gradient across rows is higher, early-termination is performed on the column-wise 1D_IDCT and row-wise 1D_IDCT is performed as a normal operation. We achieve a speedup of about 15-20 percent on the test bit-streams.

Software-only implementation on a chip sustaining 80 percent peak performance: The implementation presented in this paper is a complete MPEG-2 implementation of the standard (some portions are omitted for brevity) in software including the IO interfaces. We achieved a sustained performance of 80 percent of the peak processing power of MDSP engines.

The rest of the paper is organized as follows:We present design goals and parallelization opportunities in section 3. Section 4 covers a brief overview of the MDSP architecture, processor mapping,resource estimation and implementation strategy. Section 5 gives details on results and performance analysis.

2 Related Work

Lot of work [6, 8, 10, 12–16] has been done on parallelization of the MPEG-2 video decoding. Bilas et. al. [13] compare the coarse-grain parallelism at the GOP level and fine-grain parallelism at the slice level implemented on an SGI challenge multiprocessor. We used fine-grained parallelism at the slice level in our contribution. [11] presents a software solution on TI's Multimedia Video multi-Processor C80 and reports real-time results for the codec implementation. The MVP has a RISC processor, 4 DSP-like processors, DMA and video controller and large amount of cache. Ishiwata et. al. [17] reports the MEP customizable media processor core by the Toshiba. MEP is largely a hardware solution wherein the core can be customized for different applications like MPEG-2 to provide a hardware solution. It uses hardware accelerators for VLD, IDCT etc. Sriram and Hung [18] use data and instruction parallelism in their implementation on TI C6x along with the ILP offered by the VLIW.

3 MPEG-2 Video Decoder

An extensive description of the MPEG-2 could be found in [1–4]. MPEG-2 addresses the compression of various profiles and layers for bit-rates ranging from 4 to 100 Mbps and resolutions ranging from CIF to HDTV.

The decoder performs the following steps in the algorithm as shown in Fig. 1: (1) VLC decoding, VLD (2) De-quantization, IQ (3) Inverse DCT, IDCT - this step outputs the error (4) Motion compensation, MC (5) Add prediction and error.

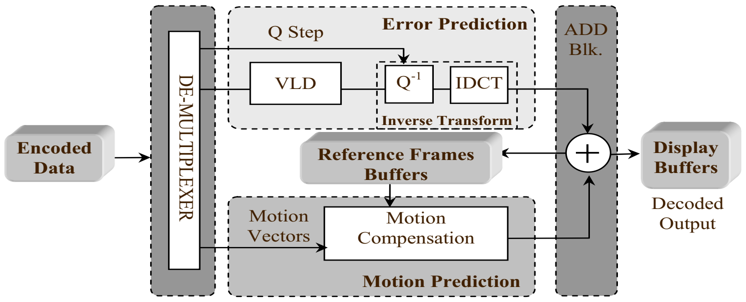

Fig. 1. Functional block diagram of the MPEG-2 Decoder

3.1 Design Goals

The design of decoder was done with the following goals in mind: *(1)* Minimal resources in terms of (a) processors (b) DRAM bandwidth (c) DRAM size and

(d) local memory size *(2)* Scalability in terms of (a) the ability to run on a variable number of processors, i.e. processor scalable (b) should exploit the on-chip memory, i.e. memory scalable *(3)* Reusability in terms of (a) have library of commonly used domain specific routines (b) design should support re-configuring of modules, i.e. plug & play (i) Selection of processor and (ii) Communication between processes (loosely coupled). These goals are common to most of the implementations on the MDSP architecture.

3.2 Parallelization Opportunities

The MDSP architecture allows us to exploit all kinds of parallelization opportunities provided by the MPEG-2 bit-stream.

Data Parallelism: Individual slices can be decoded independently and hence concurrently. Individual MSPs could be put to this task of decoding, called DecodeSliceTask(). Thus data parallelism is exploited at the MSP level as shown in Fig. 2 and 3.

Algorithmic Parallelism: Prediction error computation (VLD,IQ,IDCT) and Motion Compensated prediction (MC) can be done in parallel on different MBs. The decoded MB is obtained by adding MC data with the corresponding prediction error (see Fig. 2).

Pipelined Parallelism: The error prediction computation, which consists of (a) VLD (b) IQ and (c) IDCT, can be done in a pipe. VLD and IDCT are compute intensive operations. While one processor is decoding the VLC codes of block N another processor can compute the IDCT of the already decoded block N-1 as shown in Fig. 2. Thus, after an initial pipelline delay the above tasks run in parallel.

4 MDSP Architecture Overview and Processor Mapping

MDSP is an array of RISC and DSP processors that provide a seamless, scalable system solution to the full spectrum of video and multimedia related products. Four MSPs (4 PEs, 8 DSEs) are grouped in a single Compute Quad and share a common instruction cache (32 KB) and data memory (64 KB). The instruction cache is utilized by the PEs only. Each DSE has its own dedicated instruction memory of 512 instructions and 128 dedicated registers. A 4-way hardware multithreaded DMA engine (MTE) facilitates data pre-fetch from external DRAM. An IO quad facilitates the interface to the external world with its 2 PEs and 2 MTEs.

Each quad has 32 local and 64 global semaphores for synchronization between tasks and guarding critical, shared resources. The architecture is more amenable to high throughput when computation is effectively overlapped with

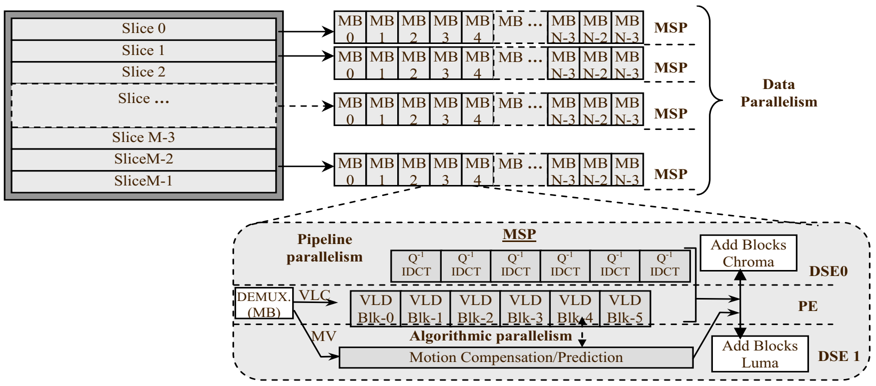

Fig. 2. Overview of various parallelisms existing in MPEG-2 Decoder

data communication. This can be done using double (or ping-pong) buffering strategies.

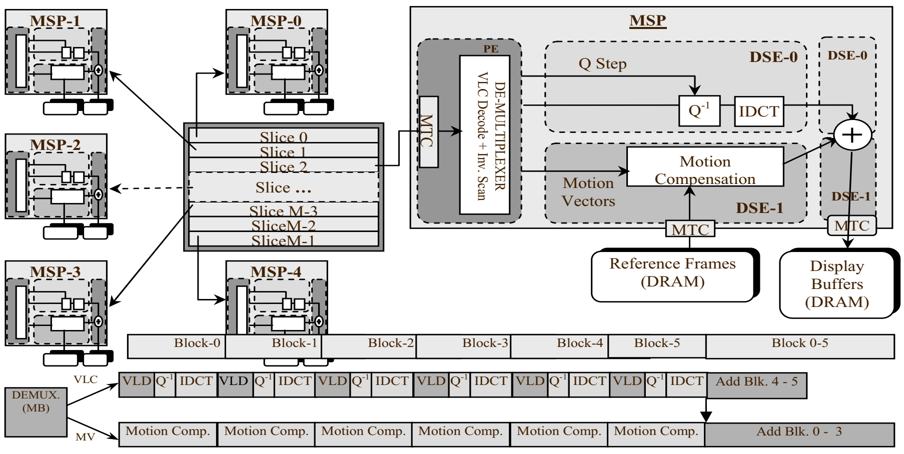

Fig. 3. Mapping MPEG-2 Video decoder on MDSP

4.1 Resource Estimation

Based on the profiling and hand-coding the compute intensive algorithms for DSEs,we estimated the decoder to take (a) 6-8 MSPs for supporting 15 Mbps video decoding (b) 32 KB local memory (c) 3.2 MB of DRAM (d) 300-400 MBps peak DRAM bandwidth (e) Average DRAM bandwidth required for I-pictures about 70 MBps and for B-pictures about 100 MBps. Fig. 3 shows the mapping on MDSP based on this estimation.

4.2 Implementation Strategy

The decoder implementation is divided into different tasks and typically works as a co-worker model. The following tasks are the main tasks that are implemented: (a) Detect/Bin Slice Task: detect slice pointers and put them in a bin, DetectSliceTask (b) DecodeSliceTask (c) FrameReorderTask and (d) ControllerTask

The controller task is the main task that is responsible for scheduling, allocation of tasks and updating shared global variables that facilitate communication between these tasks. The ControllerTask() is protected by a semaphore. As shown in Fig. 4, any MSP seeking a task has to wait till this semaphore is free. Once the semaphore is free, ControllerTask() allocates the next task to the MSP based on the status of the decoder.

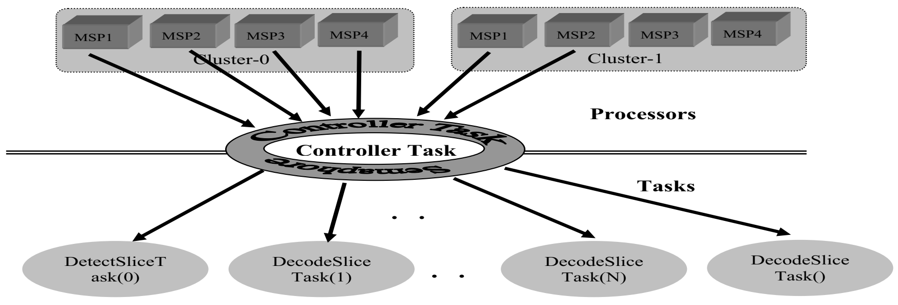

Fig. 4. Controller Task

Multiple VLD: The main idea in the multiple VLD implementations is to decode multiple symbols for every table lookup operation. The tables are packed in such a fashion that they carry multiple symbols whenever possible. This is done on a subset of most probable symbols with small number of bits. The symbols associated with larger number of bits of VLC code are put in separate tables. The smaller VLC codes are assigned to most frequent symbols in all of the video and image processing standards. This helps in packing more symbols together and also speeding the implementation. Fig. 5 shows an example of how multiple symbols can be packed. The field access unit present in each DSE facilitates easy extraction of multiple symbols. The speedup achieved by this approach varies from 35-40 percent on most bit-streams.

Implementing IDCT: We have implemented Chen's [9] IDCT algorithm. The algorithm uses floating-point arithmetic. It also uses the even-odd decomposition technique described in [26]. The implementation includes the saturation logic required for the MPEG-2 Video. The IDCT is implemented using the 4 MACs

Index	Run-0	Level-0	Run-1	Level-1	Valid Bit Count
…	…	…	…	…	…
01000100	1	1	1	1	8
01000101	1	1	1	-1	8
…	…	…	…	…	…
01010100	1	-1	1	1	8
01010101	1	-1	1	-1	8
…	…	…	…	…	…
01001000	1	1	0	1	7
01001001	1	1	0	1	7
01001010	1	1	0	-1	7
01001011	1	1	0	-1	7
…	…	…	…	…	…
01011000	1	-1	0	1	7
01011001	1	-1	0	1	7
01011010	1	-1	0	-1	7
01011011	1	-1	0	-1	7
…	…	…	…	…	…
01110100	0	3	0	1	8
01110101	0	3	0	-1	8
…	…	…	…	…	…
01111100	0	-3	0	1	8
01111101	0	-3	0	-1	8
…	…	…	…	…	…

Fig. 5. Multiple VLD implementation example (Table B-15 from MPEG-2 standard)

available on the DSE. The core loop performs 1-D IDCT in 55 Clock Cycles. The IDCT alongwith saturation takes 1600 clock cycles. With the early-termination algorithm the cycles reduce to 1300-1400. Since the MAC floating-point results can be truncated as well as rounded, we don't need to take special care of the IEEE precision requirements. This implementation has been verified to pass both the IEEE precision requirements for IDCT [2, 27, 7] as well as the dynamic range requirement in Corrigendum-2 of the MPEG-2 Video.

We have also done standalone experiments to implement the IDCT using the SIMD capabilities of the newer CRA3001 instruction set. The reference implementation is suggested in [22]. On these experiments we have got a speedup of 1.67.

Implementing Motion Compensation: Motion compensation is split across 2 DSEs. The MC DSE calculates the predictions for the Luma blocks and adds them to the IDCT error block from the IDCT DSE. Chroma prediction are also calculated on the MC DSE. However the addition of chroma predictions to the IDCT error is done by the IDCT DSE. The addition of the blocks on both the DSEs starts after the IDCT and Motion Prediction on the MB is finished by the respective DSEs. This achieves a better load-balance between the two DSEs within MSP and performance improvement, since both the DSEs do the same amount of work in terms of cycles. Each MSP replicates the above mapping.

4.3 General Design Strategy for Media Applications

While designing on multiprocessor architecture like MDSP it is important to understand whether the application is parallelizable. 1. Look for opportunities for parallelism, mainly data parallelism. 2. Compute the resource requirements in terms of cycles and bandwidth requirements 3. Allocate independent tasks to

M processors in a controlled manner 4. Keep all processors busy and processing units load-balanced. 5. Overlap compute with communicate using ping-pong buffering stratergies. The idea here is that while one of the ping-pong buffer is getting filled by the DMA engine, the processing units can work on the previously filled ping-pong buffer. 6. Speed up tasks like VLD (it can't be parallelized and other tasks in the pipe depend on it) so that they don't starve the other tasks in the pipe. 7. Invest in a good control procedure to reduce operating system overhead. The simplest stratergy is to statically allocate processors for particular tasks. Another is to implement a work task scheduler, as shown in Fig. 4. Alternatively, an SMP real-time kernel such as eCos can be used.

5 Performance Analysis

We refrain from comparing against implementations like TI C6x [18], Pentium (with MMX), HP PA(with MAX) [20] and UltraSPARC (with VIS) for the reason that the goal of MPEG-2 MP@ML is to achieve 30 fps real-time video, which we achieve in our implementation. Also the results specified in our implementations are on real-bitstreams provided by third party. Sriram and Hung [18] do a comparison between their implementation and the implementations specified above. However, these are biased towards 25 percent non-zero DCT coefficients and GOP structure of 2I, 20B, 8P frames of every 30 frames. This is not the case with real bit streams and sometimes not a good assumption as specified in [18]. Fig. 6 depicts the scalability of the implementation for SDTV sequences of NTSC and PAL formats. Thus, one would require 4-6 MSPs to decode bit-

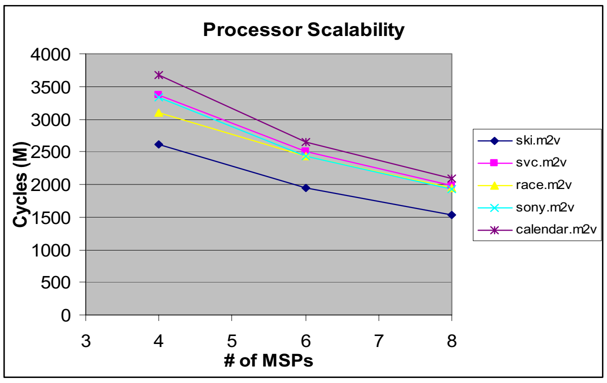

Fig. 6. Processor Scalability for different bit streams

streams ranging 5-15 mbps. On an average MDSP requires 1350M cycles for decoding 1 sec SDTV video. For HDTV we would require close to 28 MSPs assuming overheads. We expect the current Cradle chip CT3400 in production to meet these performance requirements.

6 Conclusions

Current implementation on CRA2003 does not exploit the SIMD/PIMAC capabilities offered by current chips like CT3400 in the MDSP family. These can be effectively utilized for motion compensation and IDCT algorithms as done in the implementations [22–25]. VLD algorithm is presently implemented on the PE, comparatively a slower processor to the DSE. The DSE's field access unit can be effectively utilised for further speeding up the VLD computation.

From our design experience on MDSP architecture we see that the architecture is processor scalable and suitable for video processing applications. Video processing applications are more amenable to data parallelism and are in turn processor scalable. Most of the implementations are bottlenecked for being processor scalable as well as memory scalable. We have found that it typically requires up to 4 MSPs to decode a MPEG-2 video sequence of 5 Mbps and up to 6 MSPs to decode a sequence of 15 Mbps. Also the processor scalable nature of the architecture gives in to code reuse which in turn would reduce the development cost and time. General purpose CPUs are very expensive in terms of dollar cost and solutions using just DSP multiprocessors require additional I/O components for a full solution for interfaces to external world. On the MDSP this could be done on the same chip using the IO quad provided with additional MTE units and software programmable IO. Exploiting peak sustained peformance out of VLIW DSP processors costs more development time as compared to MDSP. With MDSP we could achieve a sustained 80 percent of peak performance in our implementation.

Acknowledgements: We would like to thank members of the Cradle hardware and Cradle India Development Center team for their help with this work. In particular, we thank David Harrison for active discussion and feedback and Satish Gupta for his constant encouragement and support.

References

1. International Standard ISO/IEC 11172-2 Information Technology: Coding of Moving Pictures and associated audio for the storage media at 1.5 Mbps
2. International Standard ISO/IEC 13818-2 Information Technology - Generic Coding of moving pictures and associated audio. Information: Video
3. Barry Haskell, Atul Puri and Aurn Netravali: Digital Video : Introduction to MPEG2
4. Vasudev Bhaskaran and Konstanstions Konstantinides: Image and Video Compression Standards Algorithms and Architectures. Kulwer Academic Publicatons
5. T. Onoye, T. Masaki, Y. Morimoto, et.al.: Single chip implementation of MPEG-2 decoder for HDTV level pictures. IEICE Trans. Fundamentals, March 1996.
6. Aravind Bala, Darshat Shah,Wu-chi Feng, D.K. Panda: Experiences with Software MPEG-2 Video Decompression on an SMP PC.
7. V. Venkateswar: Precision Requirements for IDCT Implementations on 340I. Tech. Report, Semiconductor & Process Design Center, Texas Instruments, Feb. 1992.

8. K. Guttag, R.J. Gove, and J.R. Van Aken: A Single-Chip Multiprocessor for Multimedia: The MVP. IEEE Computer Graphics & Applications, Nov. 1992.
9. W.H. Chen, C.H. Smith, and S.C. Fralick: A Fast Computational Algorithm for the DCT. IEEE Trans. on Communications, Vol. COM-25, 1977.
10. Heng Liao, Andrew Wolfe: Available Parallelism in Video Applications. International Symposium on Microarchitecture, 1997
11. Murat Tekalp: Digital Video Processing. Prentice Hall PTR, 1995.
12. Fujitsu: MPEG2 Decoder with embedded SDRAM, RAMPEG MB87P2030.
13. Angelos Bilas, Jason Fritts, Jaswinder Pal Singh: Real-time parallel MPEG-2 decoding in software. Princeton University Technical Report TR-516-96.
14. W. Lee, J. Goldston, et. al.: Real-time MPEG video codec on single chip multiprocessor. Proc. of the SPIE, Digital Video Compression on Personal Computers: Algorithms and Technologies, Vol. 2187, Feb. 1994, pp. 32-42.
15. H. Jeschke, K. Gaedke, and P. Pirsch: Multiprocessor Performance for Real-Time Processing of Video Coding Applications. IEEE Trans. on Circuits and Systems for Video Technology, Vol. 2, pp. 221-230, 1992.
16. T. Akiyama, H. Aono, K. Aoki, et. al.: MPEG-2 Video Codec using Image Compression DSP, IEEE Trans. on Consumer Electronics, Vol. 40, pp. 466-472, 1994.
17. Shunichi Ishiwata, Tomoo Yamakage et. al.:A Single-Chip MPEG-2 Codec Based on Customizable Media Microprocessor, IEEE CICC 2002.
18. Sundararajan Sriram, Ching-Yu Hung:MPEG-2 video decoding on the TMS320C6X DSP architecture, IEEE Asilomar Conf. on Signals, Systems and Computers 1998.
19. Cradle Technologies, CRA2003 & CT3400 Hardware Architecture reference.
20. R. B. Lee et. al.:Real-time software MPEG video decoder on multimedia-enhanced PA 7100LC processor, Hewlett-Packard Journal, April 1995.
21. D. Ishii et. al.:Parallel variable length decoding with inverse quantization for software MPEG-2 decoders, Proc. of 1997 workshop on SiPS,1997.
22. Intel: Using MMX instructions in a fast IDCT algorithm for MPEG decoding, Intel Application Note AP-528.
23. Intel: Using MMX instructions to implement optimized motion compensation for MPEG1 video playback, Intel Application Note AP-529.
24. C. Fogg: Survey of software and hardware VLC architectures, SPIE Vol. 2186, Image and Video Compression, 1994.
25. C-G Zhou, L. Kohn, D. Rice, I. Kabir, A. Jabibi and X-P Hu, "MPEG video decoding with UltraSPARC visual instruction set", Compcon, Spring 1995
26. C-Y Huang and P. Landman: A compact IDCT design for MPEG video decoding, Proc. 1997 IEEE workshop on Signal Processing Systems (SiPS), Nov. 1997.
27. IEEE standard specification for the implementation of 8 by 8 IDCT, IEEE standard 1180-1190.
28. Matjaz Verderber, Andrej Zemva, Danjan Lampret: Hw/sw partitioned optimization and VLSI FPGA implementation of the MPEG-2 video decoder, SIGDA 2003.
29. Shaw-Min Lei, Ming-Ting Sun: An entropy coding system for digital HDTV applications, IEEE transactions on circuits and systems for video technology, vol. 1. pp 147-155, Mar. 1991.
30. Philips, TriMedia TM-1300 Media Processor Data Book, Sep. 2000.