MAVD: MPEG-2 Audio Video Decode system on MDSP™

Ganesh Yadav, R K Singh, and Vipin Chaudhary

Abstract — *We have implemented a software only MPEG-2 Audio Video Decode (MAVD) system on the Cradle MDSP™ architecture and we highlight the suitability of MDSP™ architecture to exploit the data, algorithmic, and pipeline parallelization offered by Video processing algorithms like the MPEG2 Video for real-time performance and efficient partitioning of System, Audio and Video Processing on a single chip multiprocessor. 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 MPEG2 video decoding system and MPEG-2 Stereo Decode System on this shared memory MDSP™ platform. We also highlight how the processor scalability is exploited as part of the design on this architecture. Although simultaneous audio-video decode on general-purpose processors provides flexibility, they are not cost-effective. Most of the media processors exploit hardware acceleration in part or full to alleviate the high-throughput demands put by these algorithms; thereby making them inflexible for other applications. With the flexibility offered by the Cradle platform we could design a video decoder that could scale from four MSPs (Media Stream Processor that is a 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 Cradle's internal CRA20.03 evaluation board. Specific contributions include the multiple VLD algorithm and other heuristic approaches like early-termination IDCT for fast video decoding.*

Index Terms — MPEG-2 Audio Video, Multiple VLD, Multiprocessor DSP, System-on-Chip.

I. INTRODUCTION

The availability of software programmable System-on-Chip (SoC) architectures like MDSP™ [38-40] eliminates the need of having dedicated hardware accelerator boards and additional glue logic to build a full system solution for each standard we want to work with. With the rapid evolution of standards like MPEG-2 [17-19, 27], MPEG-4[20], H.264 [28], 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 on programmable processors fully in software has many advantages: it is less expensive and more flexible for accommodating new algorithms and enhancements as they evolve. This flexibility is offered by the MDSP™ architecture along with reduced time to market (less than 50% of ASIC cycle) as compared to ASIC based solutions.

Most existing implementations extract either data or pipeline parallelism along with Instruction Level Parallelism offered by VLIW in their implementations. On general-purpose processors, the MPEG2 implementations are usually memory bottlenecked. Our solution combines the data, algorithmic and pipeline parallelization and is a greedy strategy (applies static scheduling) to exploit performance. However, the static scheduling scheme allows mapping any given module (audio, video decode/render) on to any of the MSP or group of MSPs and is not tied to a particular MSP. A unique advantage of a software implementation is that with new intellectual contributions (like multiple VLD, faster IDCT; explained below) we could just plug-in the newer modifications to make the implementations faster.

A. Intellectual Contributions

Multiple VLD: The main idea in the multiple VLD implementations is to decode multiple symbols in one 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 symbols and 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-audio and image processing standards. This helps to pack multiple such symbols together to make a lookup table. We could improve the performance of the overall VLD operation with our algorithm by 55-70% over normal lookup based approaches. Verderber *et. al.* [2] report that a lookup table based VLD structure proposed by Lei and Sun [3] is the fastest known VLD decoder today. With our modifications applied to this VLD algorithm it can be improved further by 45-50% in software. Our modification adds the capability to decode each codeword in a single

Ganesh Yadav and Vipin Chaudhary are with Department of Computer Science, Wayne State University, 5143 Cass Avenue, Detroit, MI, 48202 USA (e-mail: ganesh@ cs.wayne.edu, vipin@wayne.edu).

R. K. Singh is with Cradle Technologies, Inc. 82-103 Pioneer Way, Mountain View, 94041 USA.

0-7803-8526-8/04/$20.00 ©2004 IEEE.

cycle as well as multiple VLD symbols in a single cycle whenever allowed by the bit-stream.

Early-termination IDCT: This idea is an offshoot of the MPEG-4 AC-DC prediction. The MPEG-4 AC-DC prediction predicts the current block first row or first column coefficients based on the gradient of the DC value either in column direction or row direction is higher. This essentially checks whether the given MB has vertical edges or horizontal edges or gradient profile in either of this direction. So when the DCT is applied, most of the coefficients across row or column become zeroes in areas with these types of gradient profiles. This helps in the early-termination of the IDCT. When we see that the gradient across the 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. Similar process is carried out if the gradient direction is across rows. In that case early-termination is performed on the column-wise 1D_IDCT and row-wise 1D_IDCT is performed as a normal operation. In addition, one can skip the 1-D IDCT whenever the coefficients in a row are zeroes. Using this method we achieved a speedup of about 15-20% on the test bit-streams.

Software-only implementation on a Chip sustaining 80% 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. It achieves a sustained performance of 80% of the peak.

B. Design Goals Achieved

The following design goals were achieved by the implementation of the MPEG2 video decoder:

(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)* having library of commonly used domain specific routines
- (b) design that should support re-configuring of modules, *i.e.,* Plug & Play
  - *(i)* Selection of processor and
  - *(ii)* Communication between processes (loosely coupled).

These goals are not specific to this implementation and would be common to most of the implementations on the MDSP™ architecture.

II. Related Work

MPEG-2 implementations [14-15, 21, 23-24, 35, 37] are studied on different platforms including DSP, SMP PCs and general purpose processors. Various MPEG-2 video parallelization approaches [9, 13-14, 26, 29, 31, 33] are studied and cost-benefit analysis of different strategies is carried out. [31] presents a data-parallel approach to MPEG-2 video decoding. [13] presents a performance analysis on VLIW/DSP architecture from the perspective of code compaction and optimizations. [33] talks about two parallelizing approaches at the coarse-grain GOP level and fine-gain slice level. In our contribution we use fine-grained slice level parallelism approach. Various architecture and design strategies [1-2, 5-8, 25, 31-32, 34, 36] have been tried to map the MPEG-2 application. [1] presents a multi-core SoC for the implementation of Multimedia applications on a single chip. [2] presents a Hardware-Software partitioned approach to MPEG-2 video decoding. [5] presents multimedia application design on the VLIW multiprocessor Imagine core. [6] studies multimedia application development on the vector media processors. [7] carries out a performance and cost-benefit analysis study of multimedia applications on VLIW, superscalar and VLIW architectures. [8,34] present reconfigurable architectures for mapping multimedia applications. Ishiwata et. al. [34] reports the MEP customizable media processor core by 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. [25, 32] present multiprocessor solutions to the MPEG-2 video decoding problem. [36] presents a single chip ASIC LSI for MPEG-2 video decode. Sriram and Hung [37, 46] use data and instruction parallelism in their implementation on TI C6x along with the ILP offered by the VLIW. [46] presents a MPEG Audio/video performance analysis on the TI DSP. Also multimedia applications are studied on general purpose processors [42-43, 45]. Most of the general purpose processors used for PCs now have multimedia extensions [10-11, 41] for speeding up these applications. Also in most of the media applications memory becomes the bottleneck rather than the compute power of the processors. [12] presents a hard-wired memory pre-fetching technique to increase the data bandwidth throughput of the memory system.

III. MDSP ARCHITECTURE

The system is implemented on CRA20.03 chip of the MDSP™ family. CRA20.03 is Cradle's internal evaluation board. Fig. 1 presents the Cradle's chip architecture. MDSP is an *array of RISC and DSP processors* (MSPs) that provide a seamless, scalable system solution to the full spectrum of video and multimedia related products. An MSP consists of a RISC engine, PE and two fast DSP engines, DSEs and a Memory Transfer Engine (MTE) to facilitate data pre-fetch from external DRAM. Four such MSPs are grouped as a single cluster that shares a common instruction (32 Kbytes) and data memory (64 Kbytes). Each DSE has a dedicated instruction memory and 128 registers. They also share a common high-speed local bus. The processors are loosely coupled with common instruction and data memory. The processors can be programmed individually or in-group to exploit the available parallelism in a given application for maximum throughput. Current chip in the family of MDSP™, CT3400 has one compute quad and one IO quad and runs at 230 MHz. The chip is designed to provide a hardware platform that can be fully programmed to meet the demanding need of video and audio processing, image processing, graphics, communications and other similar applications.

Apart from handling the high processing requirements of the multimedia applications, MDSP provides a sufficient degree of flexibility – processors are loosely coupled, integrates a powerful on-chip communication structure; shared memory communication and synchronization is achieved using local and global hardware semaphores; and a well-balanced memory structure that provides a large amount of on-chip memory for growing data demands and helps reduce the bandwidth constraints on off-chip memory.

On a wide range of *media processing* applications, a single chip MDSP processor achieves 80% sustained of its peak performance. This performance is achieved by casting these applications as *data parallel* applications along with *pipelined parallelism* (applications structured as streams of data passing through computation kernels). MDSP like other processors [1,5-6, 25, 34] targets the four key attributes of Media applications, including signal processing, image and video processing, and graphics.

High Compute and Data Bandwidth Requirements: Media applications are compute intensive and require up to tens of billions of operations per second to achieve real-time performance. A Compute Quad in MDSP has multiple high compute Digital Signal Engines (DSEs) which perform arithmetic and Multiply-accumulate operations. They also have SIMD [10-11, 41] and Parallel Integer Multiply Accumulate (PIMAC) capabilities. The multithreaded DMA engines help in pre-fetching/transfer of the large amount of data. Thus, the high compute requirement is achieved by the high throughput DSEs and data bandwidth is handled by the MTE.

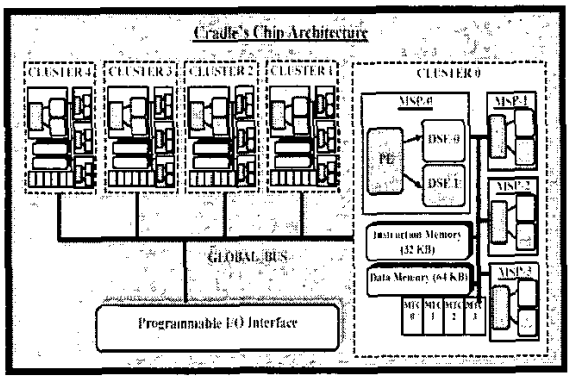

Fig. 1. Cradle's Chip Architecture

High Computation to Communication Ratio: Pipelined Parallelism exposes locality of media processing applications, allowing implementations to minimize global memory usage. Thus, pipelined-parallel programs tend to achieve a high computation to memory ratio: most media applications perform large amount of compute operations for each data memory reference. Large amount of cycles are spent computing the data fetched and comparatively very less number of cycles are spent on fetching the data to the local memory from DRAM. MDSP provides a 64K local memory for pre-fetching data in memory. This local data is passed through compute kernels achieving high compute to communicate ratio. Memory latency is hidden by using programming techniques like ping-pong buffering. Ping-pong buffering allows the compute kernels to operate on the data already in the local memory while the DMA fetches next set of data. This effectively overlaps computation with communication. Also, because of high computation to communication ratio, memory does not become the bottleneck.

Locality of Computations with Less Global Data Reuse: The typical data reference pattern in media applications requires a single read and write per global data element. Little global reuse means that traditional caches are largely ineffective in these applications. Intermediate results are usually produced at the end of a computation stage and consumed at the beginning of the next stage.

IV. IMPLEMENTATION

We have implemented a full-fledged MPEG-2 Audio Video Decode (MAVD) system on this architecture. The system consists of Transport/Program Stream (TS/PS) bit stream demux, Video Decoder and Renderer, Audio Decoder and Renderer. The IO quad (a portion of the MDSP™ architecture responsible for IO) implements interfaces for video/audio-out and bit stream management. Fig. 2 shows the system mapping on the MDSP™ platform.

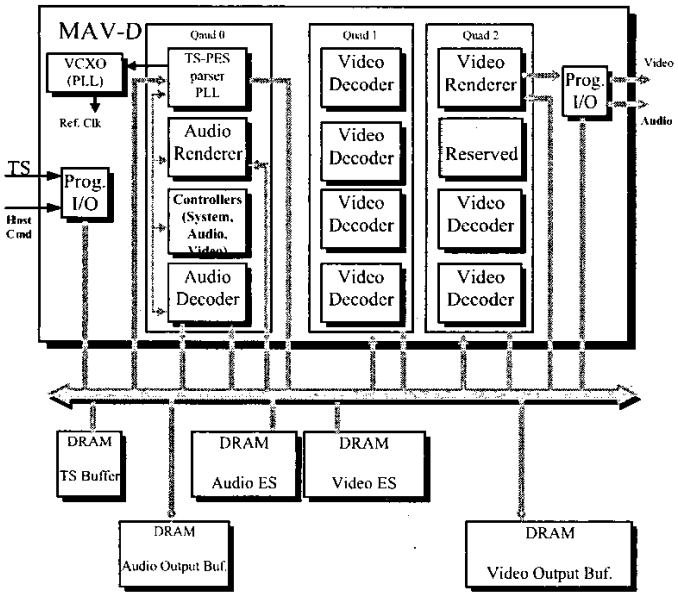

Fig. 2. MAVD mapping on MDSP CRA20.03

A. Resource Estimation and the Implementation Strategy

The following resource estimation was done based on the profiling of the C code and hand coding the compute intensive algorithms for the DSEs. The resource estimation is done for a MPEG-2 A/V system on the MDSP™ architecture. System controllers include MPEG-2 System controller and PLL controller and renderers include audio and video renderers. Fig. 3 shows the system block diagram along with the memory and data bandwidth requirements for MPEG-2 Video.

System Controllers & Renderers: The system controllers & renderes require two MSPs. One MSP is required for handling system components and 1 MSP is required for audio/video rendering. They together would require 4 kbytes of local memory.

Video Decoder: MPEG-2 MP@ML video decoder is estimated to require 6-8 MSPs for supporting 15 Mbits/sec video decoding. It requires 32 Kbytes local memory and 3.2 Mbytes of DRAM. The peak DRAM bandwidth estimated for the video decoder is 300-400 MB/sec peak. Average DRAM bandwidth estimated for I-pictures is about 70 MB/sec and for B-pictures is about 100 MB/sec.

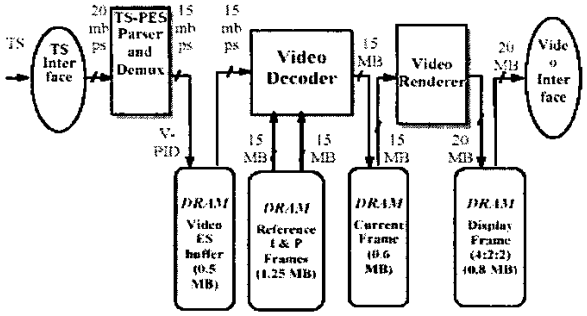

Fig. 3. MPEG-2 Video System Block Diagram

MPEG-2 Stereo Audio Decoder: The Audio decoder is estimated to take one MSP for MPEG-2 Stereo Decode. It requires 5 Kbytes local memory and 80 Kbytes of DRAM. The data bandwidth is estimated at 30 MB/sec peak and 10 MB/sec average DRAM bandwidth.

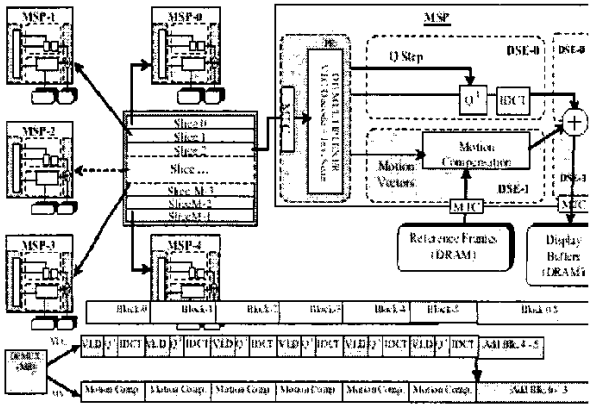

Fig. 4. MPEG-2 Video Decoder Mapping

External memory requirement for the whole system is estimated at 4 Mbytes of DRAM with 500 Mbytes/sec access speeds. The whole system diagram is presented here for demonstrating a solution capability of MDSP™ (see Fig. 2).

Fig. 4 depicts the mapping of the MPEG-2 video decoder on a portion of MDSP and shows data, pipeline and algorithmic parallelisms. Data parallelism is exploited at the slice level and pipeline parallelism is achieved at the block level within a Macroblock. Algorithmic parallelism is achieved between the VLD,

Inverse Quantization, Inverse DCT and Motion Compensation.

Index	Run-0	Level-0	Run-1	Level-1	Valid Bit Count
...	...	...	...	...	...
0100 0100	1	1	1	1	8
0100 0101	1	1	1	-1	8
...	...	...	...	...	...
0100 1000	1	-1	1	1	8
0100 1001	1	-1	1	-1	8
...	...	...	...	...	...
0110 0000	1	1	0	2	7
0110 0001	1	1	0	2	7
0110 0010	1	1	0	-1	7
0110 0011	1	1	0	-1	7
...	...	...	...	...	...
0110 1000	1	-1	0	1	7
0110 1001	1	-1	0	2	7
0110 1010	1	-1	0	-1	7
0110 1011	1	-1	0	-1	7
...	...	...	...	...	...
0111 0100	0	3	0	1	8
0111 0101	0	3	0	-1	8
...	...	...	...	...	...
0111 1100	0	-3	0	1	8
0111 1101	0	-3	0	-1	8
...	...	...	...	...	...

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

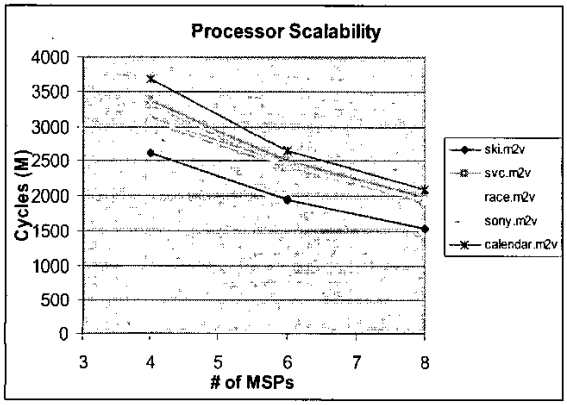

Fig. 6. Processor Scalability of MPEG-2 video decoder

Fig. 6 shows the scalability of the implementation on various MPEG-2 bit streams ranging from 3-15 Mbps. The system works as a co-worker model, where each MSP is active (unlike master-slave) and asks for more work from the controller task when it has finished its share of work ,*e.g.*, decoding of slice. Audio decode is done on a single MSP. Synchronization between tasks and guarding critical regions is achieved through hardware semaphores.

We have also implemented the multiple variable length decode algorithm (see Fig. 5) implemented in most video applications on MDSP. Multiple lookup tables are created (each entry is 32 bit) containing multiple VLC symbols packed in each entry along with the sign of the levels. In our implementations we were able to decode multiple symbols per cycle. The implementation is optimal and targeted towards memory constrained embedded systems. We have confirmed this algorithm in our implementations of H261/3, MPEG-2/4 and achieved multifold speedup improvements against algorithms, which can decode at the symbol rate only. This limits the decoding throughput capability of these algorithms. Most parallel decoding approaches use the length of the first code-word to detect the second code-word in parallel. By a single table lookup operation we detect multiple code-words without any detection mechanisms.

V. Conclusion

We have presented MPEG-2 Audio Video Decode System on the MDSP platform. While designing on multiprocessor architecture like Cradle's MDSP™ it is important to understand whether the application is parallelizable. We need to look for opportunities for data, algorithmic, and pipeline parallelization. From our design experience on Cradle's architecture we see that the architecture is processor scalable and suitable for video processing applications that are more amenable to data parallelism. Also the processor scalable nature of the Cradle's architecture gives in to code reuse which in turn would reduce the development cost and time. On the MDSP™ a single chip solution could be developed using the IO quad provided with additional MTE units and software programmable IO for developing interfaces like I^2C, PCI, Video-in/Video-out for NTSC/PAL standards. Finally we have achieved a sustained 80% of peak performance in our implementation.

References

[1] Hans-Joachim Stolberg et. al: "HiBRID-SoC – A Mutli-Core System-On-Chip Architecture for Mutlitmedia Signal Processing Application", SIGDA 2003.

[2] Matjaz Verderber, Andrej Zemva,Ddanjan Lampret, "Hw/Sw partitioned optimization and VLSI FPGA implementation of the MPEG-2 video decoder", SIGDA 2003.

[3] 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, March 1991.

[4] D. Ishii et. al., "Parallel variable length decoding with inverse quantization for software MPEG-2 decoders", Proceedings of 1997 workshop on signal processing systems (SiPS), 1997.

[5] John Owens et. al. : "Media Processing Applications on the Imagine Stream Processor", Proc. IEEE International Conference on Computer Design: VLSI in Computers and Processors.

[6] Kozyrakis: "Scalable Vector Media-processors for Embedded Systems", PhD thesis, University of California at Berkeley, 2002.

[7] C. Kozyrakis and D. Patterson: "Vector vs. superscalar and VLIW architectures for embedded multimedia benchmarks", In Proceedings of the 35th Annual IEEE/ACM International Symposium on Microarchitecture, November 2002.

[8] Bove,V.M. and Watlington, J.A. Cheops: "A Reconfigurable Data-Flow System for Video Processing" IEEE Transactions on Circuits and Systems for Video Technology (April 5, 1995), pp. 140-149.

[9] Liao, Heng and Wolfe, Andrew: "Available Parallelism in Video Applications", In Proceedings of the International Symposium on Microarchitecture (December, 1997), pp. 321-329.

[10] Peleg, Alex and Weiser, Uri: "MMX Technology Extension to the Intel Architecture", In IEEE Micro (August, 1996), pp. 42-50.

[11] Tremblay, Marc, et. al. : 'VIS Speeds New Media Processing", In IEEE Micro (August, 1996), pp. 10-20.

[12] Jason Fritts: "Multi-Level Memory Prefetching for Media and Stream Processing", Proceedings of the 2002 International Conference on Multimedia and Expo.

[13] Tsvetomir P. Petrov: "Code Compaction and Parallelization for VLIW/DSP Chip Architectures", Masters Thesis, MIT, June 1999.

[14] Han chen et. al.: "A Parallel Ultra-High Resolution MPEG-2 Video Decoder for PC Cluster Based Tiled Display Systems", IEEE Proc. of the International Parallel and Distributed Symposium, 2002.

[15] Jui-Hua Li et. al. : "An Efficient Video Decoder for MPEG-2 MP@ML", Proc. of the IEEE International Conference on ASAP, 1997.

[16] International Standard ISO/IEC 11172-2 Information Technology: Coding of Moving Pictures and associated audio for the storage media at 1.5 Mbits/s.

[17] International Standard ISO/IEC 13818-2 Information Technology - Generic Coding of moving pictures and associated audio. Information: Video.

[18] Barry G. Haskell, Atul Puri and Aurn N, Netravali: Digital Video : Introduction to MPEG-2.

[19] Vasudev Bhaskaran and Konstanstions Konstantinides: Image and Video Compression Standards Algorithms and Architectures. Kulwer Academic Publicatons.

[20] ISO/IEC 14496-2 – Information Technology - Coding of Audio-Visual Objects: Visual, 1999.

[21] T. Onoye, T. Masaki, Y. Morimoto, Y. Sato, I. Shirakawa, and K. Matsumura: "Single chip implementation of MPEG2 decoder for HDTV level pictures", IEICE Trans. Fundamentals, vol. E79-A, no. 3, pp. 330-338, March 1996.

[22] H. Mizuno, H. Kobayashi, T. Onoye, and I. Shirakawa: "Performance estimation at architecture level for embedded systems", IEICE Trans. Fundamentals, vol. E85-A, no. 12, pp. 2636-2644, Dec. 2002.

[23] T. Masaki, Y. Morimoto, Y. Sato, T. Onoye, and I. Shirakawa: "Single-chip VLSI decoder for MPEG2 MP@HL", Proc. Synthesis and System Integration of Mixed Technologies, pp. 211-218, Aug. 1995.

[24] Aravind Bala, Darshat Shah,Wu-chi Feng, D.K. Panda: "Experiences with Software MPEG-2 Video Decompression on an SMP PC", Department of Computer and Information Science Ohio State University,

[25] K. Guttag, R.J. Gove, and J.R. Van Aken: "A Single-Chip Multiprocessor for Multimedia: The MVP", IEEE Computer Graphics & Applications, pp. 53--64, Nov., 1992.

[26] Heng Liao, Andrew Wolfe: "Available Parallelism in Video Applications", International Symposium on Microarchitecture, 1997.

[27] Murat Tekalp: Digital Video Processing. Prentice Hall PTR, 1995.

[28] Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, Draft ITU-T recommendation and final Draft of International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC), Document number JVT-G050r1.doc, May 2003.

[29] Angelos Bilas, Jason Fritts, Jaswinder Pal Singh: "Real-time parallel MPEG-2 decoding in software". Princeton University Technical Report TR-516-96.

[30] W. Lee, J. Goldston, R. J. Gove, and Y. Kim: "Real-time MPEG video codec on single chip multiprocessor", Proceedings of the SPIE, Digital Video Compression on Personal Computers: Algorithms and Technologies, Vol. 2187, feb. 1994, pp. 32-42.

[31] Shahriar M. Akramullah, Ishfaq Ahmad, and Ming L. Liou. "A Data-Parallel Approach for Real-Time MPEG-2 Video Encoding", Journal of Parallel and Distributed Computing, 1995 Nov. 1.30(2), pp. 129-146.

[32] 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.

[33] T. Akiyama, H. Aono, K. Aoki, K.W. Ler, B. Wilson, T. Araki et. al.: "MPEG2 Video Codec using Image Compression DSP", IEEE Trans. on Consumer Electronics, Vol. 40, pp. 466--472, 1994.

[34] Shunichi Ishiwata, Tomoo Yamakage, et. al. : "A Single-Chip MPEG-2 Codec Based on Customizable Media Microprocessor" IEEE CICC 2002.

[35] Les Kohn, Greg Efland "A one-chip MPEG-2 codec makes DVD authoring on a PC possible", BYTE, December 1997.

[36] H. Mizosoe, K. Maeda, Y. Kubo, Y. Tsusru, K. Koruki "An advanced multimedia processing LSI suitable for HDTV applications" International Conference on Consumer Electronics(ICCE) 2001, pp 96-97.

[37] Sundararajan Sriram, Ching-Yu Hung, "MPEG-2 video decoding on the TMS320C6X DSP architecture", IEEE Asilomar Conf. on Signals, Systems, and Computers 1998.

[38] Cradle Technologies, "CRA2003 architecture reference, version 4.0".

[39] Cradle Technologies, "CRA3001 architecture reference"

[40] Cradle Technologies, "3400 Hardware Architecture reference, version 4.0", DT-00100-002.

[41] R. B. Lee et. al., "Real-time software MPEG video decoder on multimedia-enhanced PA 7100LC processor", Hewlett-Packard Journal April 1995.

[42] Intel, "Using MMX instructions in a fast IDCT algorithm for MPEG decoding", Intel Application Note AP-528

[43] Intel, "Using MMX instructions to implement optimized motion compensation for MPEG1 video playback", Intel Application Note AP-529.

[44] C. Fogg, "Survey of software and hardware VLC architectures", SPIE Vol. 2186, Image and Video Compression, 1994.

[45] 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.

[46] D. Hocevar, S. Sriram, C-Y. Hung, "Performance modeling for system design: an MPEG A/V decoder example", ISCAS, June 1998.