

# HW/SW PARTITIONED OPTIMIZATION AND VLSI-FPGA IMPLEMENTATION OF THE MPEG-2 VIDEO DECODER

Matjaž Verderber, Andrej Žemva

University of Ljubljana, Faculty of Electrical Engineering, Ljubljana, Slovenia

**Key words:** MPEG-2 video decoder, FPGA implementation, optimization, ISO/IEC 13818-2, power consumption, speeding-up, inverse discrete cosine transform-IDCT, variable length decoding-VLD, Huffman coding, embedded system

**Abstract:** In this paper, we have proposed optimized real-time MPEG-2 video decoder. The decoder has been implemented in one FPGA device as a HW/SW partitioned system. We made timing/power-consumption analysis and optimization of the MPEG-2 decoder. On the basis of the achieved results, we decided for hardware implementation of the IDCT and VLD algorithms. Remaining parts were realized in software with 32-bit RISC processor. MPEG-2 decoder (RISC processor, IDCT core, VLD core) has been described in high-level Verilog/VHDL hardware description language and implemented in Virtex 1600E FPGA. Finally, the decoder has been tested on the Flextronics prototyping board.

## Strojno in programsko razdeljena optimizacija in FPGA implementacija MPEG-2 video dekoderja

**Ključne besede:** MPEG-2 video dekodek, FPGA implementacija, optimizacija, ISO/IEC 13818-2, poraba moči, pohitritev, inverzna diskretna kosinusna transformacija-DCT, dekodiranje s spremenljivo dolžino-VLD, Huffmanovo kodiranje, vgrajeni sistem

**Izvleček:** V članku smo predstavili optimiziran MPEG-2 video dekodek namenjen dekodiranju v realnem času. Dekodek smo realizirali v enem FPGA vezju kot kombinacijo programske in strojne rešitve. Opravili smo analizo in optimizacijo hitrosti delovanja ter porabe moči MPEG-2 dekoderja. Na podlagi rezultatov smo se odločili, da časovno in energijsko zahtevne algoritme (inverzna kosinusna transformacija ter dekodiranje s spremenljivo dolžino) realiziramo direktno v strojni opremi, ostale dele dekoderja pa realiziramo z RISC mikroprocesorjem. Celoten MPEG-2 dekodek (RISC procesor, IDCT jedro, VLD jedro) smo opisali v visokonivojskih jezikih Verilog in VHDL in ga implementirali v Virtex 1600 programirljivem FPGA vezju. Vezje dekoderja smo nato preizkusili v realnem testnem okolju na FPGA prototipnem sistemu proizvajalca Flextronics.

### 1. Introduction

MPEG-2 video standard is an important standard for video compression today /1/. MPEG-2 coding/decoding algorithm can be found in different applications such as digital video broadcast, DVD, cable TV, graphics/image processing cards, set top boxes, digital cameras, SDTV and HDTV. Due to different profiles and levels of the MPEG-2 video standard, every application has specific ratio between performance and quality. Since modern multimedia applications increase both aspects of the MPEG-2 compression, it is necessary to achieve best performance in terms of real-time operation and reduced hardware cost.

The MPEG-2 standard includes several compression techniques such as variable length coding (VLC), discrete cosine transform (DCT), quantization and motion compensation. It was shown that some of these parts can be optimized with parallel structures and efficiently implemented in the hardware-software partitioned system. Recently, several MPEG-2 decoders have been developed either as software based applications /2/ or hardware based ASIC custom chips /3/.

A parallel decoder for the MPEG-2 standard implemented on a shared memory multiprocessor was presented in /2/. The primary goal of this approach was to provide an all-

software memory solution for real-time high quality video decoding and to investigate the important properties as they pertain to multiprocessor systems.

In /3/, a VLSI chip for real-time decoding of MPEG-2 video was developed as hardware based ASIC device with un-optimized decoding functions.

In literature, we haven't found any description of combined hardware-software implementation of the MPEG-2 decoder on a single chip. The reason is probably the lack of technology that provides efficient hardware/software implementation. With the recent advantages in technology from leading manufactures of the programmable devices (Xilinx-Microblaze /4/, Altera-Nios /5/), the proposed design gain importance. The VLC decoding algorithm and IDCT algorithm are implemented with fast parallel architectures directly in hardware. Hardware part is described in VHDL/Verilog and implemented together with the RISC processor in a single Virtex 1600E FPGA device. Decoding of the coded bitstream, inverse quantization, motion-compensation are running in software on 32-bit RISC processor which is implemented in FPGA and uses Linux as the operating system. This is so due to the non-existence of efficient parallel architectures for these algorithms. This partitioning was selected in order to achieve better timing properties and lower power consumption.

With the proposed parallel optimization in hardware, up to 40% improvement is achieved compared to the complete software implementation. The main benefit of our MPEG-2 decoder is leveraging best characteristics from hardware and software based solutions. The proposed MPEG-2 video decoder was tested on the Flextronics FPGA based industrial board /6,7/ which serves as a testing environment and forms the intermediate step in VLSI chip design cycle.

This paper is organized as follows: the next section describes timing optimization and optimization of the power consumption of the decoder. The implementation of the MPEG-2 decoder is presented in Section 3. Finally, Section 4 completes the paper with a conclusion and future work.

## 2. Optimization of the MPEG2 Video Decoder

As illustrated in Figure 1, coded pictures are first variable length (Huffman) decoded and their type is determined from the header information. Pictures are inverse scanned and inverse quantized. Inverse quantization is performed in two steps. Data are first divided with a quantization-step matrix and then scaled with a scale factor.

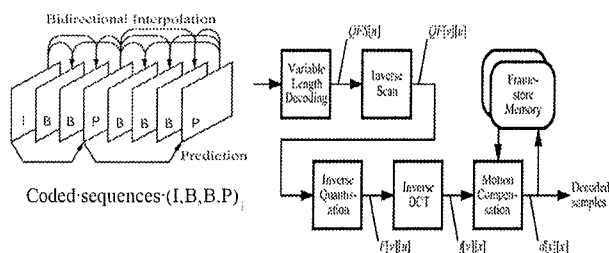


Figure 1: Structure of the ISO/IEC 13818-2 MPEG-2 video decoder

MPEG-2 standard defines 3 types of the coded frames (I, P, B pictures). Intra-coded pictures I take advantages of the picture's spatial correlation. They do not reference any other pictures in the coded bit stream. They provide fast random access with only moderate compression. I pictures are only DCT decoded while no motion compensation is performed. Inter-coded pictures P, B consider also temporal correlation between successive pictures (motion compensation). P pictures are decoded using motion-compensated prediction from the past I or P pictures which are stored in the Frame-Store-Memory. The compression for P pictures is better than for I pictures, and P pictures can be used as reference points for additional motion compensation. Bidirectionally predicted pictures B provide the highest degree of compression. They are decoded using motion-compensated prediction from either past and/or future I or P pictures. Since B pictures are not used in the prediction of other B or P pictures, such pictures accommodate more distortion and hence yielding more compression than I or P pictures.

The ISO/IEC 13818-2 (MPEG-2 video part) standard defines only the frame for different algorithms included in the standard /8/. Therefore, there are various possibilities to optimize particular parts of the decoder.

In order to deal with timing and power-consumption optimization, we have developed a software support in visual C++ programming language. A screenshot of the software tool is shown in Figure 2.



Figure 2: Program for MPEG-2 software simulation

This software allows us to simulate input sequences with different settings (profile/level, bit-rate, I/P frame distance, number of frames in GOP, frame/field pictures and all other coder/decoder parameters). The complete analysis can be performed and timing properties of particular algorithms included in the standard can be simulated. For timing analysis, elapsed time of the decoding parts in milli-seconds is measured for every decoded sequence. Finally, the speed of the program execution is evaluated. Based on the obtained results, we have estimated equivalent decoding speed for real-time decoding.

### 2.1. Timing Optimization

Figure 3 shows a computational load of the decoding functions in MPEG-2 decoder.

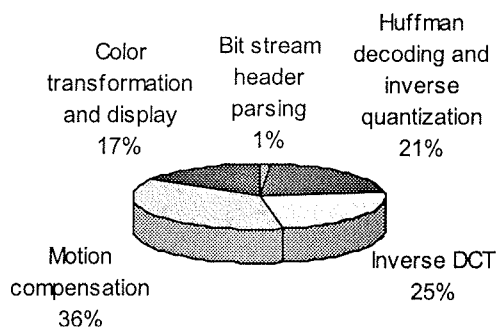


Figure 3: Computational power distribution for MPEG-2 decoding

The data in the Figure 3 are based on simulations of source input format (SIF) (352x244, 25Hz) resolution pictures coded at 4 Mbits/s containing two B-pictures for every P-picture. As illustrated in Figure 3, the most computational expensive parts of the decoder are motion compensation, inverse DCT and VLD decoding.

The main idea of our decoder is to exploit advantages of the parallel structures which can be efficiently implemented in hardware. Hardware implementation of DCT and VLD decoder promises better results compared to software based algorithms. The key point of a parallel hardware structure is reduced number of operation and an ability to work in parallel. On the contrary, hardware implementation of motion compensation does not gain performance versus software based solution. In our decoder, we have therefore built fast hardware cores for IDCT and VLD decoder.

The basic computation element in a DCT-based system is the transformation of an  $N \times N$  image block from the spatial domain to the DCT domain and vice versa. For the image compression standards,  $N$  is usually 8. From a hardware or software implementation point of view, an  $8 \times 8$  block size does not impose significant memory requirements. Furthermore, the computational complexity of an  $8 \times 8$  DCT is manageable on most computing platforms. Mathematical formulation for the 2D discrete cosine transform is shown in (1).

$$F(u, v) = \frac{C_u C_v}{4} \sum_{x=0}^7 \sum_{y=0}^7 f(x, y) \cos\left(\frac{(2x+1)\mu\pi}{16}\right) \cos\left(\frac{(2y+1)\nu\pi}{16}\right)$$

$$C_u, C_v = \frac{1}{\sqrt{2}}(u, v = 0) \dots \text{otherwise} \quad 1$$

$$\begin{aligned} x, y & \text{ coordinates in pixel domain}(0 \dots 7) \\ u, v & \text{ coordinates in DCT domain}(0 \dots 7) \end{aligned} \quad (1)$$

The separable nature of the 2D DCT is exploited by performing the 1D DCT on the eight rows and then the 1D DCT on the eight columns /9/. Several fast algorithms are available for calculation of the 8-point 1D DCT. In our decoder, scaled version of Chen has been used /10/. It was selected due to the minimum required number of additions and multiplication. Structure of Chen DCT is illustrated in Figure 4.

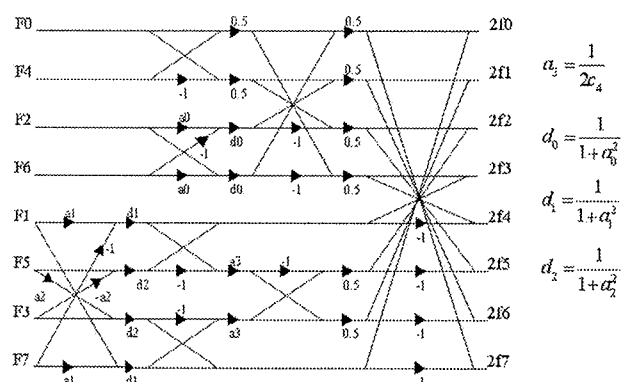


Figure 4: Modified Chen 1D inverse DCT structure

For variable length decoder (VLD), a lookup table based decoder structure proposed by Lei and Sun /11/ has been used. It is one of the fastest known variable length decoders today /1/. It decodes each codeword in a single cycle regardless of its length. The block diagram of the Lei-Sun VLD is illustrated in Figure 5.

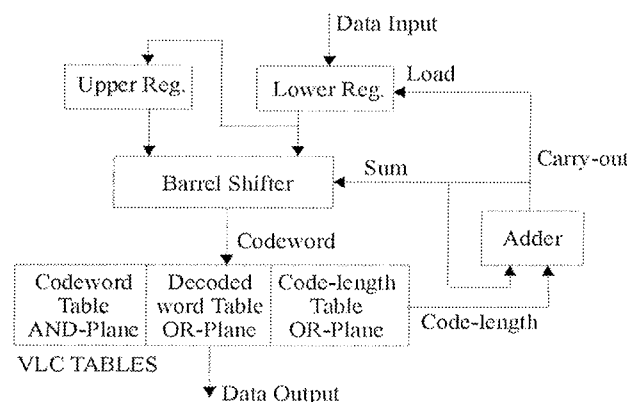


Figure 5: Block diagram of the Lei-Sun variable length decoder

Coded data are always concatenated together and segmented into 16-bit words. When the 16-bit lower and upper register are fully loaded, the decoding process starts. At each cycle, the output of the barrel shifter is matched in parallel with all the entries in the codeword lookup table. The barrel shifter operates like a sliding window on the contents of the two registers. When a match is found, the lookup table outputs the corresponding source symbol and the length of the decoded codeword. The barrel shifter is then shifted to the beginning of the next codeword. As the 4-bit adder overflows 16, this indicates that the upper register has been fully decoded. After the content of the lower register is transferred into the upper register, the VLC decoder loads a new 16-bit segment into the lower register, and operations continue.

Once the IDCT and VLD had been modeled in VHDL, we have performed timing simulation of MPEG-2 decoder. IDCT 1-D Chen structure is described in VHDL at RTL level. Our simulations have shown a delay of 100 ns for 1-D IDCT and 1.6  $\mu$ s for 2-D 8x8 IDCT targeting the Xilinx FPGAs. In the VLD decoder, every coded symbol can be decoded in only one cycle. Since the prototyping evaluation board operates at 25 MHz, 40 ns delay for each decoded data is required.

Our results show up to 40% improvement of speed for MPEG-2 decoding compared to software based solution. The results before and after optimization are presented in Figure 6.

These results have been obtained with the software decoding speed of 45 MHz. We have calculated equivalent decoding speed for the real-time decoding rate (40 ns for 25 Hz frame rate) which was 72 MHz. As seen in Figure 6, there are great deviations from the average decoding times.

This is a consequence of the irregular CPU tasks and can be ignored.

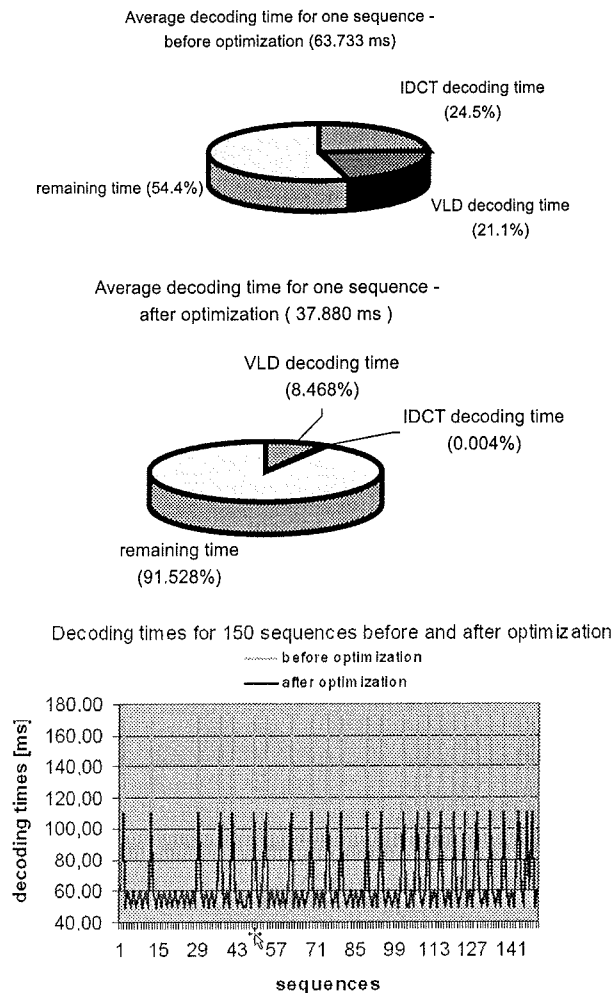


Figure 6: Results of the timing simulation before and after optimization

## 2.2. Power Consumption Analysis

At this point, we summarize results of the detailed energy conscious study made by Henkel and Li /12/. The study is based on the idea that an application specific hardware can be more energy efficient due to the higher utilization. They made an energy dissipation study by distinguishing five different cases of HW/SW partitioned MPEG-2 encoder (Figure 7). As a reference case a complete software implementation was used. Then, four candidate hardware parts were synthesized (quantization-case 1, two dimensional DCT-case 2, one dimensional DCT-case 3, quantization and two dimensional DCT-case 4) and analyzed. The encoder was intended to run on an architecture composed of the CPU, instruction and data cache, a main memory and an application specific hardware (ASIC). Since every encoder also includes a duplicated decoder, results can be easily adopted for the MPEG-2 decoder. As illustrated in Table 1, each of the selected co-designs in cases 1 to 4 yields an energy savings up to 48% compared to the reference case of software based solution.

| Case                                               | Energy [ mJ] |       |        |
|----------------------------------------------------|--------------|-------|--------|
|                                                    | SW           | HW    | total  |
| <b>Case 0:</b> All SW Solution                     | 74.32        | N/A   | 74.32  |
| <b>Case 1:</b> Quant. in HW, rest in SW.           | 49.33        | 14.09 | 63.42  |
| <b>Case 2:</b> 2-D DCT in HW, rest in SW.          | 27.16        | 0.709 | 27.869 |
| <b>Case 3:</b> 1-D DCT in HW, rest in SW           | 49.51        | 0.156 | 49.666 |
| <b>Case 4:</b> Quant. and 2-D DCT in HW rest in SW | 16.99        | 22.27 | 39.26  |

Table 1: Energy dissipation for selected co-designs

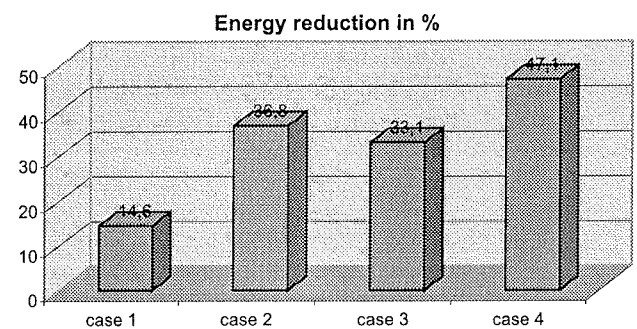


Figure 7: Results of the case study for energy-conscious HW/SW partitioning

## 3. VLSI-FPGA Implementation of the MPEG-2 Video Decoder

### 3.1. System Environment

Our HW/SW partitioned MPEG-2 decoder has been tested on the Flextronics /6/ FPGA based prototyping board (Figure 8).

The heart of the board is the Xilinx Virtex 1600E FPGA /13/. Several peripheral devices and connectors (ethernet, VGA, LCD, GSM, UART, IDE, PS2) serve as interfaces from the FPGA to the external world. 32 MByte flash,

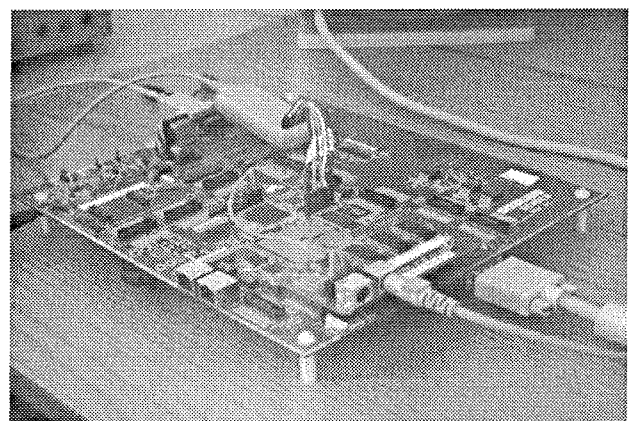


Figure 8: Flextronics prototyping board

64 MByte SDRAM, and 4 MByte SRAM allow implementation of the complex FPGA applications. For the MPEG-2 decoder, we are using flash memory (loaded input sequences), SRAM, UART (display of the MPEG-2 decoding messages) and VGA output to the display.

Every hardware core is first described in Verilog or any other high level hardware description language (HDL). We have used Synplify tool /14/ for circuit synthesis and Xilinx ISE tool /15/ for target implementation. Once the hardware is designed, there are two options how to port software applications on the board. The first is to use Linux operating system. On the RISC processor and various peripheral hardware cores (ethernet, ps2, uart etc), we can port derivative of Linux kernel (uClinux) which is intended for microcontrollers without Memory Management Units (MMUs). The second option is to use the monitor program which is loaded into the RAM of the RISC controller. The latter method is used during the development cycle. When an application meets the requirements, it is compiled for the Linux operating system.

### 3.2 FPGA Implementation

Our HW/SW partitioned decoder is composed of three parts: a 32-bit RISC processor, IDCT and VLD hardware cores (Figure 9).

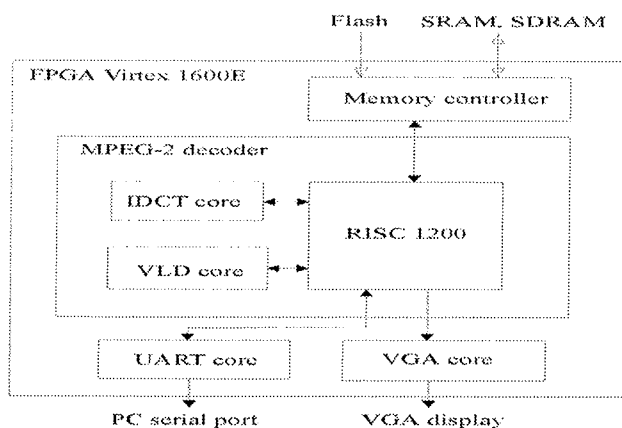


Figure 9: Block diagram of the implemented MPEG-2 decoder

The decoded frames are displayed on the analog monitor. During the MPEG-2 decoding process, we can also see possible decoding errors and warnings ("buffer overrun" for example). For this purpose, we have implemented the VGA Controller core (display of decoded frames) and UART communication core into the FPGA. For the memory accesses from RISC to external memory, the FPGA contains memory controller core.

The RISC processor is based on OpenRISC 1000 /7/ design and is a 32-bit scalar RISC with Harvard microarchitecture, 5 stage integer pipeline, virtual memory support (MMU) and basic DSP capabilities. Default caches are 1-way direct-mapped 8KB data cache and 1-way direct-mapped 8 KB instruction cache, each with 16-byte line

size. Both caches are physically tagged. Block diagram is shown in Figure 10.

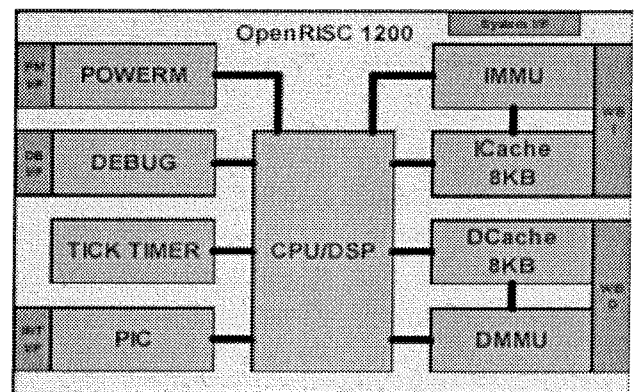


Figure 10: Architecture of the RISC controller

Supplemental facilities include a debugging unit for real-time debugging, a high resolution tick timer, a programmable interrupt controller and power management support. When implemented in a typical 0.18u 6LM process, it provides over 300 dhrystone 2.1 MIPS at 300 MHz and 300 DSP MAC 32x32 operations. OR1200 is intended for embedded, portable and networking applications.

IDCT core follows the Chen IDCT structure described in Section 2.1. The core is described with logic functions at the RTL level. It works with row-column decompositions since 2-D DCT can be computed with 1-D DCT. The simplified architecture is shown in Figure 11.

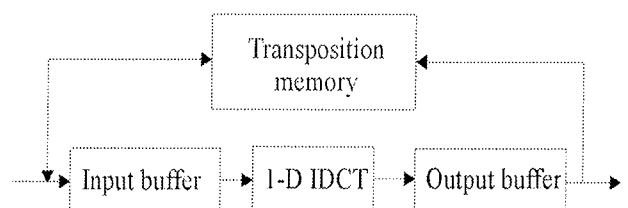


Figure 11: Simplified IDCT core architecture

The architecture of the VLD decoder is described in Figure 5. From the implementation point of view, this core is more complex compared to IDCT core. Due to the need of proper internal logic synchronization, the most complex part is controller of variable length decoding steps. There is also a problem of difficult implementation and routing of the VLD table circuit. MPEG-2 standard specifies 11 different variable length code tables with up to 17 bits wide codewords. Hardware implemented VLD decoding tables require several FPGA look up tables (LUTs), since every Virtex LUT is only 4bit wide. This problem was solved building large (over 12 bits wide) VLD tables with a use of smaller VLD tables.

### 3.3 Implementation Results

Virtex 1600E device (Figure 12) features a flexible, regular architecture that comprises an 72x108 array of confi-

gurable logic blocks (CLBs) surrounded by programmable input/output blocks (IOBs), all interconnected by a hierarchy of fast, versatile routing resources. CLBs interconnect through a general routing matrix (GRM). The GRM comprises an array of routing switches located at the intersections of horizontal and vertical routing channels. Virtex 1600E also includes 589824 bits of dedicated memory (BlockRAM) and Clock DLLs for clock-distribution delay compensation. For Clock distribution, FPGA uses fast specially dedicated global clocks (GCLKs, GCLKIOBs) or local routing resources GRM. The VersaRing I/O interface provides additional routing resources around the periphery of the device. This routing improves I/O routability and facilitates pin locking.

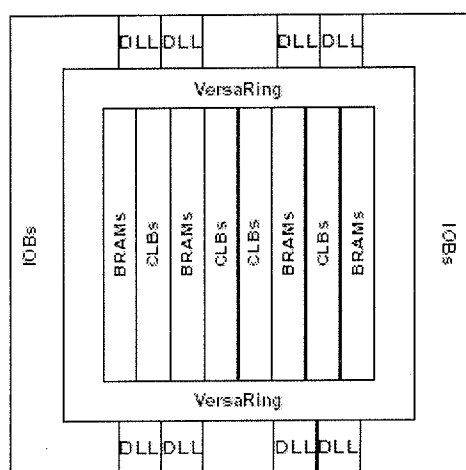


Figure 12: Virtex 1600E architecture

Each CLB is organized in two slices and provides the functional elements for constructing logic. IOBs provide the interface between package pins and the CLBs. Table 2 shows implementation results of the MPEG-2 decoder in the Virtex 1600E FPGA.

| Number of: | RISC1200 | IDCT | VLD |
|------------|----------|------|-----|
| SLICES     | 22%      | 8%   | 10% |
| BlockRAMs  | 12%      | 0    | 0   |
| IOBs       | 90%      | 20%  | 6%  |
| GCLKs      | 75%      | 75%  | 25% |
| GCLKIOBs   | 90%      | 25%  | 25% |

Table 2: Utilization of the FPGA

Results in the Table 2 have been obtained with separate implementation of the particular modules (RISC, IDCT, VLD). The most important data is the number of the SLICES (basic FPGA structure) and BlockRAMs. In the common implemented MPEG-2 decoder, this resources are merged together. Entire MPEG-2 decoder utilizes 40% of the Virtex device. Other resources (IOBs, GCLKs, global clock IOBs-GCLKIOBs) are shared, so the decoder utilizes 90% IOBs, 75% GCLKs and 90% GCLKIOBs. As we can see in Table 2, there is about 60% free of space for other applications.

## 4. Conclusion

In this paper, we have proposed an optimized MPEG-2 video decoder. We have considered optimization of speed and power consumption. Both, timing and the power consumption optimization show correlated results by achieving optimized MPEG-2 decoder. We have shown that 40% higher decoding speed and 36% lower power consumption can be obtained with combined hardware and software implementation. From a power-consumption point of view, it would be reasonable to implement quantization in hardware, but this guaranties any improvements of the decoding speed.

We have presented a modern implementation method where the complex embedded system (MPEG-2 decoder) can be efficiently HW/SW partitioned and optimized. In the future work, we will concentrate on timing optimization and HW/SW implementation of the next generation standard (MPEG-4).

## Literature

- /1/ Vasudev Bhaskaran, Konstantinos Konstantidies "Image and Video Compression Standards, Algorithms and Architectures", Kluwer Academic Publishers, Boston, 1997.
- /2/ Eishi Morimatsu, Kiyoshi Sakai, Koichi Yamashita, ... "Development of a VLSI Chip for Real Time MPEG-2 Video Decoder", Proceedings of the 1995 International Conference on Image Processing, IEEE, 1995.
- /3/ Angelos Bilas, Jason Fritts, Jaswinder Pal Singh, "Real-time Parallel MPEG-2 Decoding in Software", Proceedings of the 11<sup>th</sup> International Parallel Processing Symposium, IEEE 1997.
- /4/ [http://www.xilinx.com/xlnx/xil\\_prodcat\\_product.jsp?title=microblaze](http://www.xilinx.com/xlnx/xil_prodcat_product.jsp?title=microblaze)
- /5/ <http://www.altera.com/literature/lit-nio.html>
- /6/ <http://www.flextronicssemi.com/semi/flexsemi.nsf>
- /7/ <http://www.opencores.org/>
- /8/ ISO/IEC 13818-2: 1995 MPEG video standard, ITU-T H.262 Recommendation.
- /9/ G. Panneerselvam, P.J.W. Graumann, L.E. Turner, "Implementation of Fast Fourier Transform and Discrete Cosine Transform in FPGAs", FPL'95 conference.
- /10/ W.C Chen, C.h Smith and S.C. Fralick, "A fast Computational Algorithm for the Discrete Cosine Transform", IEEE Transactions on Communications, Vol. COM-25, No. 9, pp.1004-1009, Sept.1997.
- /11/ Shaw-Min Lei and Ming-Ting Sun, "An Entropy Coding System for Digital HDTV Applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, no. 1, pp. 147 - 155, March 1991.
- /12/ Jörg Henkel, Yanbing Li, "Energy-Conscious HW/SW Partitioning of Embedded Systems, A Case Study on an MPEG-2 Encoder", Proceedings of the 6<sup>th</sup> International Workshop on Hardware/Software Codesign, Seattle, 1998.
- /13/ The Programmable Logic Data Book, 2002.
- /14/ <http://www.synplicity.com/products/synplifypro/index.html>
- /15/ [http://www.xilinx.com/ise/ise\\_promo/finish\\_faster.htm](http://www.xilinx.com/ise/ise_promo/finish_faster.htm)

Matjaž Verderber, Andrej Žemva  
Univerza v Ljubljani, Fakulteta za elektrotehniko,  
Ljubljana, Slovenija