Design of Huffman Decoder FPGA Core

Ivan Mezei[1] and Rastislav Struharik[2]

***Abstract* – In this paper design of an Huffman decoder FPGA core that is part of an motion JPEG system is presented. Core is designed and verified using VHDL hardware description language. It is implemented and tested on an Virtex2 FPGA platform from Xilinx Inc.**

***Keywords* – Huffman decoding, motion JPEG, FPGA.**

I. INTRODUCTION

In this paper the development of a Huffman core that is to be implemented on Field Programmable Gate Array (FPGA) is presented. Core is a part of a project consisting of several cores that are part of an motion JPEG system [1]. The overall system is depicted in Fig. 1. It consists of Discrete Cosine Transformation (DCT), Quantizer and Entropy (Huffman) based encoder and decoder.

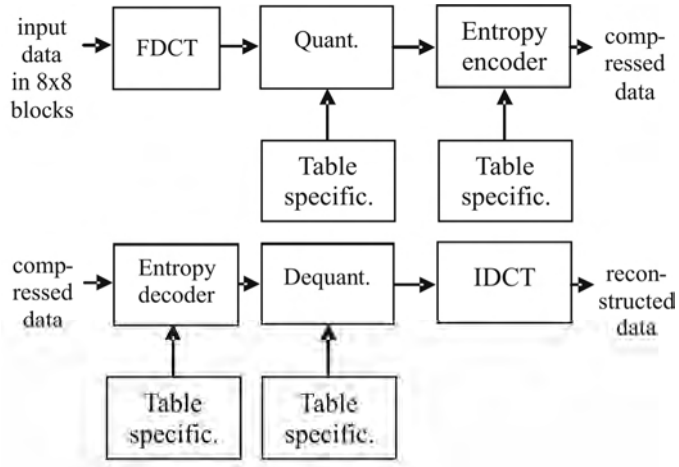

Fig. 1. Motion JPEG system

Motion JPEG is such coding system where all frames of motion pictures are coded as JPEG still pictures and then transmitted. Every picture is decomposed into luminance and chrominance components and after that in macro blocks that are 8x8 pixels sized.

It is well known that DCT is widely used in JPEG encoders [2] since it packs image data into almost optimal number of decorellated coefficients that can be compressed efficiently. Spectral energy of every transformed block is concentrated in top-left corner of macro block, meaning that in that area are coefficients that are significantly different then zero.

After quantization all coefficients that are close to zero become zero and only few coefficients remains non-zero concentrating around top-left corner.

Afterwards Huffman entropy encoding is usually done by run length encoding in zig-zag read order of transformed and quantized image data block thus producing compressed image data.

On the receiver side inverse steps are taken, and after Huffman decoding, run-length decoding, inverse zig-zag reconstruction of every block, dequantization and finally after inverse DCT image data block is decoded. This is not lossless decoding since compression is also not lossless, but this is not the problem in JPEG systems.

II. HUFFMAN DECODING

Theoretical background about Huffman coding is given in [3]. Application in JPEG compression systems can be found in [4-6]. In recent years there are several improvements in Huffman coding [2], but purpose of this paper is not to make improvements of Huffman coding algorithm but to make an core to be implemented on FPGA.

There are several possibilities for Huffman decoder architectures and various optimizations [7]. We decided to make serial sequential implementation with parallel extraction of decompressed data. Algorithm that explains how decoder works is depicted in Fig. 2.

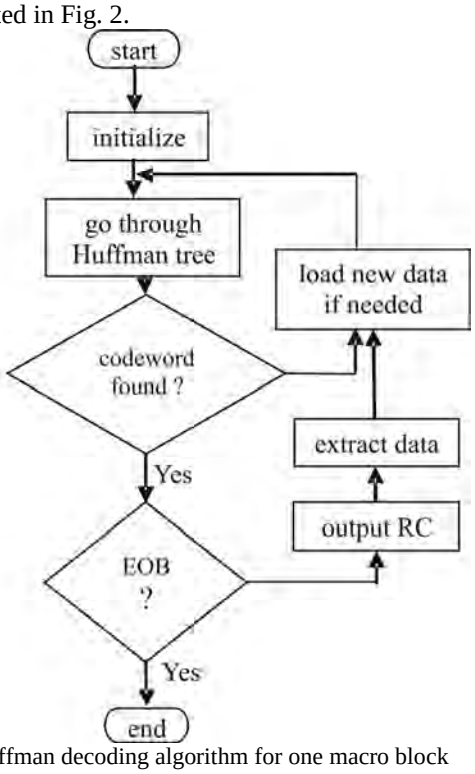

Fig. 2. Huffman decoding algorithm for one macro block

Compressed data bytes, that are input of decoder, are a sequence of Huffman codeword and additional bits in

[1]Ivan Mezei is with the Faculty of Technical Sciences, Trg Dositeja Obradovica 6, 21000 Novi Sad, Serbia, E-mail: imezei@uns.ns.ac.yu

[2]Rastislav Struharik is with the Faculty of Technical Sciences, Trg Dositeja Obradovica 6, 21000 Novi Sad, Serbia, E-mail: rasti@EUnet.yu

between. According to algorithm depicted in Fig. 2., decoder operates as follows. After initialization (loading of first compressed byte etc.), decoder goes through Huffman tree. Depending on input control signal, it uses appropriate luminance or chrominance table until it reaches valid codeword. Once codeword is found, RC coefficient can be sent to output of decoder. RC coefficient is concatenated **C**ategory of codeword and zero **R**un-length. According to category of codeword additional bits can be extracted and sent to output as decoded image data.

Using category and zero run-length, image data block can be reconstructed using inverse zig-zag and zero run-length inserting. Dequantizer and inverse DCT transformation complete the decoding process and decompressed image data is obtained as output of decoder.

III. IMPLEMENTATION

Huffman decoder core is to be implemented on Xilinx FPGA circuits. Manufacturer gave some guidelines for implementing decoder in FPGA [8]. There are different FPGA-based decoders to be used with JPEG decoding systems [9-10], and also commercial ones [11]. Our implementation is based on shift registers that serially shift out bit by bit as we go through Huffman tree based on compressed data from current macro block. After codeword have been found, additional bits are sent to output in parallel. Tables with luminance and chrominance RC coefficients are implemented as ROM memories. A small control unit controls how decoder operates.

Simplified block diagram of decoder core is shown in Fig. 3.

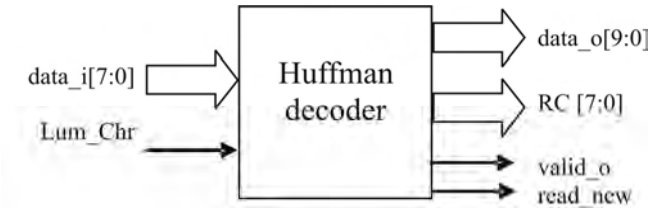

Fig. 3. Simplified decoder block

Huffman decoder core is completely designed and verified using VHDL hardware description language. Core is designed in 3 VHDL files with around 1000 lines of VHDL RTL code (half of which is luminance and chrominance coefficients table implementation). No time or spatial constraints are used.

Core is implemented and tested on 2v1000fg456-6 FPGA from Virtex2 family. Device utilization summary are shown in Table I. Decoder occupies around 5% of FPGA integrated circuit and estimated maximum frequency is around 90MHz. Possibly higher frequency could have been obtained by placing some constraints to critical signals. This was not needed since target clock frequency is 27MHz.

TABLE I
DEVICE UTILIZATION SUMMARY FOR VIRTEX2 FPGA

Device	2v1000fg456-6
Number of Slices	296 out of 5120 (5%)
Number of Slice Flip Flops	76 out of 10240 (0%)
Number of 4 input LUTs	554 out of 10240 (5%)
Number of BRAMs	4 out of 40 (10%)
Maximum Frequency	87.581MHz

Device utilization summary for one of the latest FPGAs from Xilinx (5vlx50ff1153-3 from Virtex 5 family) is shown in Table II. For this FPGA, decoder occupies below 1% of FPGA and estimated maximal frequency is around 176 Mhz.

Table II
DEVICE UTILIZATION SUMMARY FOR VIRTEX5 FPGA

Device	5vlx50ff1153-3
Number of Slice Registers	90 out of 28800 (0%)
Number of Slice LUTs	451 out of 28800 (1%)
Number of BRAMs	2 out of 48 (4 %)
Maximum Frequency	176.947 MHz

IV. CONCLUSION

In this paper design of an Huffman decoder FPGA core is presented. Core is successfully implemented and tested on Virtex2 FPGA family from Xilinx Inc. Design implementation summary for one of the latest families is reported also but testing is to be performed since we don't have that FPGA integrated circuit yet.

It is shown that estimated maximum clock frequency is around 90MHz for this implementation without any constrains applied. Actual frequency with which the core is tested is 27MHz.

V. REFERENCES

[1] P. Symes, *Video Compression Demystified*, McGraw-Hill, 2000.
[2] G. Lakhani, "Optimal Huffman Coding of DCT Blocks", IEEE Trans. on Circuits and Systems for Video Technology, vol. 14, no. 4, pp. 522-527, 2004.
[3] D.A. Huffman, "A Method for the Construction of Minimum-Redundancy Codes", Proc. of the IRE, vol. 40, no. 9, pp. 1098-1101, 1952.
[4] T. Acharya, A.K.Ray, *Image Processing – Principless and Applications*, Wiley, 2005.
[5] A. Cernasov, *Digital Video Electronics*, McGraw-Hill, 2004.
[6] G.K. Wallace, "The JPEG Still Picture Compression Standard", Communication of the ACM 34, pp. 30-44, 1991.
[7] C. Schneider, "A Parallel/Serial Trade-Off Methodology for Look-Up Table Based Decoders", Proc. DAC, pp. 498-503, 1997.
[8] L. Pillai, "Huffman coding", Application note 616, Xilinx Inc.
[9] Z. Yusof, Z. Aspar, I. Suleiman, "Field programmable gate array (FPGA) based baseline JPEG decoder", Proc. TENCON, vol. 3, pp. 218-220, 2000.
[10] M. Elbadri, R. Peterkin, V. Groza, D. Ionescu, and A. Saddik, "Hardware support of JPEG", Proc. Canadian Conference on Electrical and Computer Engineering, pp. 812-815, 2005.
[11] JPEG-D decoder core, www.cast-inc.com