IMPLEMENTATION OF HYBRID DCT/DPCM VIDEO-ENCODER IN THE DM642 EVM FOR EDUCATIONAL PURPOSES

J. Huska, P. Kulla

Department of Radioelectronics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Ilkovičova 3, 812 19 Bratislava
e-mail: huska@kre.elf.stuba.sk, ppkulla@elf.stuba.sk

Summary There is presented implementation of hybrid DCT/DPCM video-encoder in the powerful TI's media processor TMS320DM642 embed in evaluation module DM642 EVM. Code Composer Studio 3 is used as working space. Video-encoder is implemented with use of C code so it is easily understandable. The easiness is very important because the implemented video-encoder will be used for educational purposes. The structure of the encoder is lightly recognizable for students frequenting the courses Analog and Digital Television, Digital Image Processing and Encoding so some routines could be lightly modified.

1. INTRODUCTION

We have decided to implement hybrid DCT/DPCM with motion compensation video-encoder into working space Code Composer Studio 3 used for developing the applications for Texas Instrument's DSP TMS320DM642. The request for that became from the suggestion to charge students with the basics of video compression at a higher level. It is difficult for the students to understand correctly the nature of the lossy video compression in the speed of a semester. With using the live demonstration running on one of the newest DSP is attractive and inspires the interest in taught issue.

2. MPEG-2 VIDEO ENCODER MODEL

It is known that hybrid DPCM/DCT with motion compensation video encoder is the core of MPEG-2 which is wide-spread thanks to its use in digital video broadcasting (DVB) and DVD-video. For elimination of interframe redundancy it uses differential coding with motion compensation, the transform coding based on discrete cosine transform (DCT) is used for reducing the intraframe redundancy.

Statistical symbol redundancy is removed by variable lenght coding (VLC). The structure of the MPEG-2 video encoder is shown in Fig. 1.

3. THE WORKSTATION COMPOSITION

The core of MPEG-2 video-encoder is implemented in DM642 EVM. The DM642 EVM is a development board that enables evaluation of a design with the DM642 Digital Media Processor which is based on TI's successful family of C64xx DSPs. It is designed to work with Texas Instruments Code Composer Studio software tools connected through a JTAG emulator. Board features include [4]:

- 32MB of SDRAM,
- 4MB of linear flash memory,
- 2 video decoders,
- 1 video encoder,
- RS-232 UARTs, 100Mb ethernet,
- AIC23 stereo codec,
- Numerous video inputs and outputs,
- Support for HDTV data rates.

For use of the video processing card DM642 EVM was created the workstation based on personal computer (P4 2.4GHz, 512MB RAM) as shown

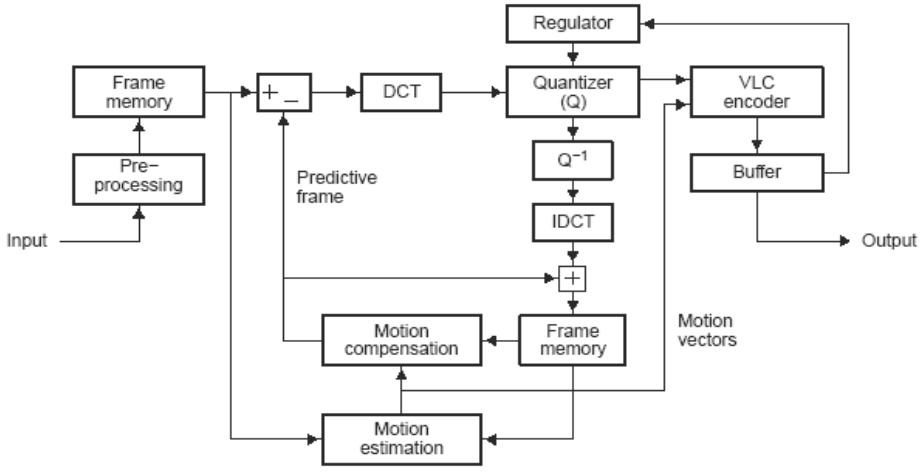

Fig. 1 MPEG-2 video-encoding block diagram, [3]

in Fig. 2. Supplementation of the JTAG boundary to host computer, required by DM642EVM, is done by XDS510 PCI card. Video-sequences are acquired by a camera in PAL standard connected directly to DM642 EVM video input. The processed video-sequence could be viewed on the connected preview screen to corresponding DM642 EVM video output.

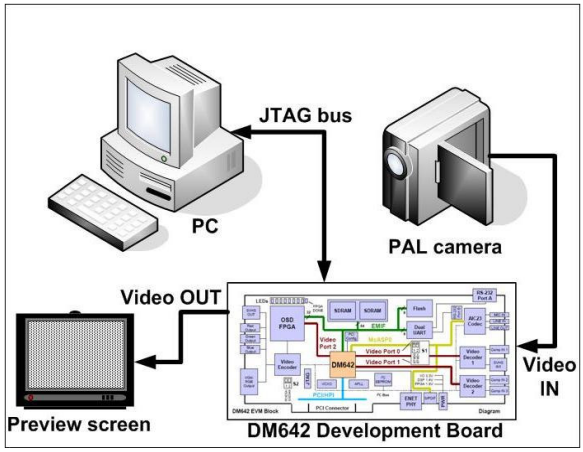

Fig. 2 Composition of the workstation for video processing

4. IMPLEMENTATION OF MPEG-2 VIDEO-ENCODER

Firstly we created universal project template in Code Composer Studio for video processing. It contains only routines for DSP configuration and video capture/video display, so it spares time when new application is developed. Blank project template consists of these three tasks:

- video input task (captures frame),
- video processing task (processes frame),
- video display task (displays frame).

The tasks are embedded in main function which performs all initializations. Algorithm is inserted into video processing task (Fig. 3). It is divided into two parts:

- control routine (frame reordering, control),
- encoding routine (I, P and B frame coding).

Control routine controls whole video-encoding process:

- saves current captured frame to off-chip frame memory,
- reorders the frames for encoding (Tab. 1),
- assigns attributes to frames (I, P or B).

Tab. 1 Video-encoding steps

capture/ display order	I1	B2	B3	P4	B5	B6	P7	B8	B9
coding steps	*P7	*B8	*B9	I1	B2	B3	P4	B5	B6
reference	*P4	*P7, I1	*P7, I1	-	I1, P4	I1, P4	I1	P4, P7	P4, P7

Comment: * stands for frame from previous GOP (Group of Pisctures)

The implementation of the own video encoding is segmented into three subroutines according to associated frame type by control routine (in regard to GOP structure) to current encoded frame:

- I-frame encoding,
- P-frame encoding,
- B-frame encoding.

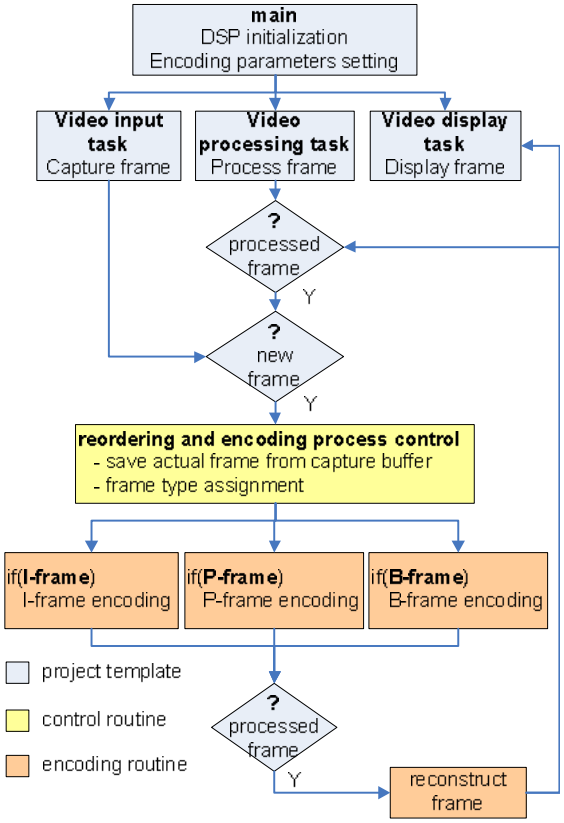

Fig. 3 Flow diagram of the video-encoder implementation

The frame is divided into slices in these encoding routines (slice consists of M macroblocks MB). The size of a MB slice, M, is only restricted by the available L1D (level 1 data cache) size. The bigger of M, the better EDMA performance could be expected for data throughput. There are used several loops in which is the frame encoding process broken in order to avoid the huge cache miss penalty and CPU stalling, [2]. M macroblocks (MB slice) are processed at a time in each loop instead of a single macroblock. The MB slice is not flush out of L1D until the frame encoding loop is over. The loops are:

- formatting raw data into MB structure,
- forward DCT with quantization,
- RLC and VLC coding,
- dequantization with inverse DCT,
- deformatting MB into raw data,
- motion estimation,
- motion compensation.

The algorithm for I-, P- and B-frame encoding is created with combination of these loops. For

simplicity the encoding scheme in a frame encoding does not meet all the MPEG-2 standard specifications. The disagreements to standard are specified in the corresponding sections.

Encoding process in I-frame encoding is the simplest among the frame types, all the macroblocks are intra coded. The flow diagram is shown in Fig. 4. Each block within the MB is DCT coded and the coefficients are divided by the quantizer steps from quantizer matrix *QMatr(w)*. The quantizer steps in *QMatr(w)* are derived from the multiplication of the original quantisation weighting matrix $QMatr_{ori}(w)$ and the quantizer parameter $q \in <1,31>$ as defined in equation (1).

$$Qmatr(w)=round(2 \cdot q \cdot QMatr_{ori}(w)/32) \qquad (1)$$

The quantization parameter is a constant value set during encoding parameters setting. The quantizer step size is different for different coefficients. The only exception is the DC coefficients, which are treated differently. This is because the eye is sensitive to large areas of luminance and chrominance errors; then the accuracy of each DC value should be high and fixed. The quantizer step size for the DC coefficient is fixed to eight. Since in the quantisation weighting matrix the DC weighting element is eight, the quantizer parameter for the DC coefficient is always 1, irrespective of the quantisation parameter used for the remaining AC coefficients.

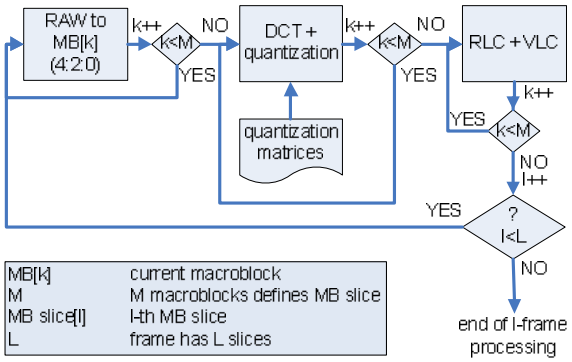

Fig. 4 I-frame encoding flow diagram

As in I-frame, each P-frame is divided into MB slices. Coding of P-frames is more complex than of I-frames, since motion compensated blocks are constructed. Than the difference between motions compensated macroblock and the current macroblock is partitioned into blocks, and then DCT transformed and coded. The flow diagram is in Fig. 5. The difference between implemented algorithm and the standard MPEG-2 rests in use of only one type of a macroblock – motion compensated. In a standard MPEG-2 [1, 3, 5] is used appropriate type of MB (intra or inter coded) upon the error between predictive MB and original MB.

In contrary to P-frame encoding, in B-frames the difference between motion compensated block and current block is not coded. Only motion vectors are saved and consecutive frame reconstruction refers to displacing the macroblock to the new position given by motion vector. Upon error between motion compensated macroblock and original macroblock is used forward, backward or both motion vectors. In a selection of macroblock position given by motion vector. Upon error between motion compensated macroblock and original macroblock is used forward, backward or

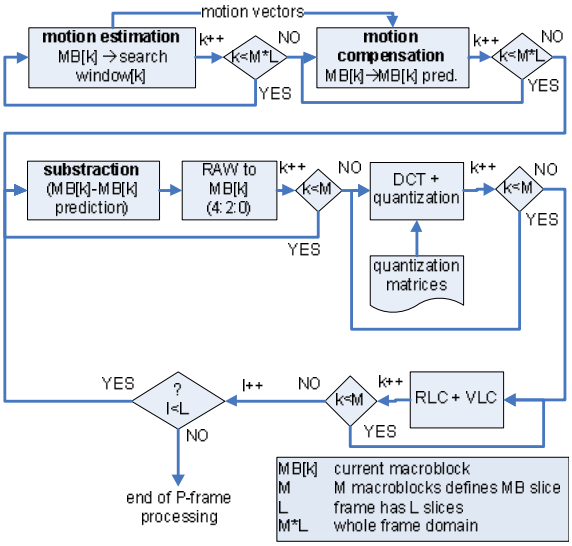

Fig. 5 P-frame encoding flow diagram

both motion vectors. In a selection of macroblock type rests the disagreement with the MPEG-2 standard. In fact, it should be coded the difference macroblock for motion compensated one which has bigger error than the threshold, [1, 3, 5]. As it was stated before, the disagreements with the standard are due to better understanding of lossy video compression basics. The flow diagram of B-frame encoding is shown in Fig. 6.

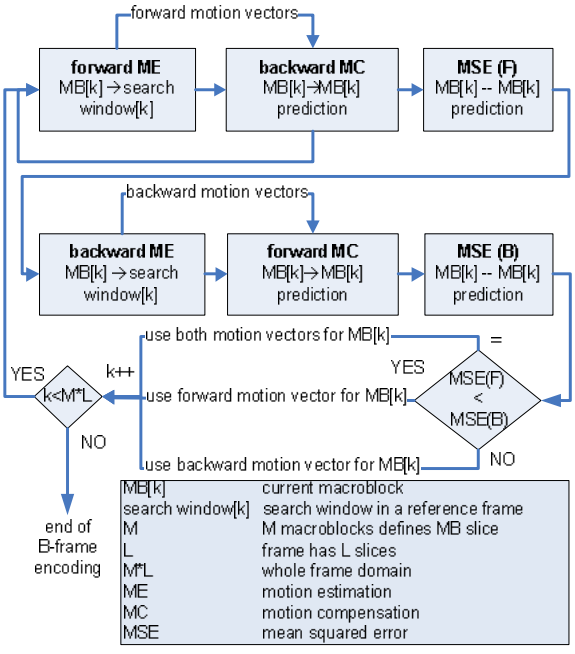

Fig. 6 B-frame encoding flow diagram

Frame reconstruction consists of inverse processing steps that are equivalent to encoding steps for corresponding frame type.

5. CONCLUSION

The core of MPEG-2 video encoder was implemented in DM642 EVM. The implementation of video encoder should be used as demonstration of lossy video compression in a teaching process related to video compression or digital television.

Thanks to the separation of encoding steps could be easily done modifications of the encoding process. With help of described video encoder implementation could be easily demonstrated the fundamentals of video compression as differential coding with and without motion compensation or influence of various motion estimation algorithms on predictive frame quality. At last in figure Fig. 7 is shown the functionality of the video encoder implementation on the example of the first four frames of GOP=IBBPBBPBB.

Acknowledgement

This contribution has been supported by the Slovakia Ministry of Education under VEGA Grant No.G-1/3107/06 and VTP Project No. 102/VTP/ 2000.

a) I-frame (first GOP frame)

b) first B-frame (second GOP frame)

c) second B-frame (third GOP frame)

d) first P-frame (fourth GOP frame)

Fig. 7 Example of the first four frames of GOP=IBBPBBPBB for quantizer parameter q=20 defined in equation (1) , used 3-step motion estimation algorithm

REFERENCES

[1] Ghanbari, M.: *Standard Codecs: Image Compression to Advanced Video Coding, Institution of Electrical Engineers, 2003.*

[2] Cheng Peng: *Video Encoding Optimization on TMS320DM64x/C64x, Texas Instruments Application Report SPRAA63, October 2004.*

[3] ISO/IEC 13818-2.

[4] www.spectrumdigital.com: *Evaluation Module (EVM) for the TMS320DM642Quick Start Installation Guide, 2005*

[5] Huska, J.: *Digitálne spracovanie a kódovanie videosekvencií v systémoch DVB – písomná časť rigoróznej skúšky, KRE FEI STU v Bratislave, 2006*