УДК 621.382

V.G. Abakumov[1], Dr. Sc., T. Ádám[2], Ph. D., B. Formanek[2], A. Kane[2], Ph. D.,
P.V. Popovych[1]

DSP Implementation of a Video Bitrate Transcoder

Исследованы решения, с помощью которых возможно реализовать транскодер видеопотока. Предложено использовать для этой цели решение на основе цифровых сигнальных процессоров. Проведены измерения параметров быстродействия для выбранных процессоров общего назначения и специализированного микроконтроллера, на которых реализован транскодер.

Solutions are researched with which help it is possible to implement video bitrate transcoder. Digital signal processors are offered for this purpose. Measurements of implemented transcoder performance parameters are carried out for selected general purpose processors and specialized microcontroller.

Introduction

Communication networks place bandwidth constraints on video transmission. Original video content usually compressed at a high bit rate to keep video quality close to the original. The network bandwidth limits require the video data to be converted to lower bit rate by real-time video transcoding before transmission. Video transcoding algorithms use information from input compressed video streams to simplify computation and to improve video quality. In this paper, we propose a digital signal processor (DSP) implementation of a low complexity open loop MPEG-2 video transcoder, working entirely in the frequency domain. Open-loop transcoders are computationally efficient, mainly used in systems with real-time requirements.

When choosing an implementation platform for next generation products, many factors are evaluated, such as performance, power consumption, cost, ease of development and another necessary feature is overall system flexibility [1].

1. Implementation platforms

There are different platforms with different features. ASICs (Application Specific Integrated Circuit) or ASSPs (Application Specific Standard Product)

are ICs (Integrated Circuit) customized for a particular use, rather than intended for general-purpose use, FPGAs (Field Programmable Gate Array) are integrated circuits designed to be configured by the customer or designer after manufacturing, DSPs (Digital Signal Processor) are specialized microprocessors with optimized architecture for fast operational needs of digital signal processing and CPUs the more and more powerful general purpose microprocessors.

1.1. ASIC

ASIC-based solutions offered high performance with the lowest power consumption and per unit cost. However, ASICs presented several problems. One is their (increasingly) high development cost in time (two or more years to product launch) and money. To recoup design costs requires prohibitively high volumes [2].

Another problem with ASICs is the lack of flexibility. With their long design cycles, ASICs aren't able to respond effectively to rapidly shifting customer needs.

1.2. FPGA

FPGAs provide a great deal of flexibility and today's FPGAs use cutting edge 45nm silicon technology and offer a wide array of speeds and capacity options. While not as flexible as a CPUs, FPGAs can be programmed to meet the exact needs of the system application. The feature set can be aligned with what the system designer needs and it can be implemented much faster than the typical two-year ASIC cycle. However, FPGAs by themselves can be quite expensive to deploy, and can be very difficult to program [3].

FPGAs usually used at a lower level in a system architecture where the computational complexity lower but data rate is high and processing speed very important. In digital television network equipments FPGAs used for Transport Stream remultiplexing, PID (Packet IDentifier) filtering, PCR (Program Clock Reference) correction or to provide complementary accelerator support to video encoder or decoder DSPs.

1.3. DSP

Originally the main difference between DSPs and other SIMD capable CPUs was that the DSPs were self-contained processors with their own signal

processing optimized instruction set, and generally operated in internal RAM driven by DMA transfers.

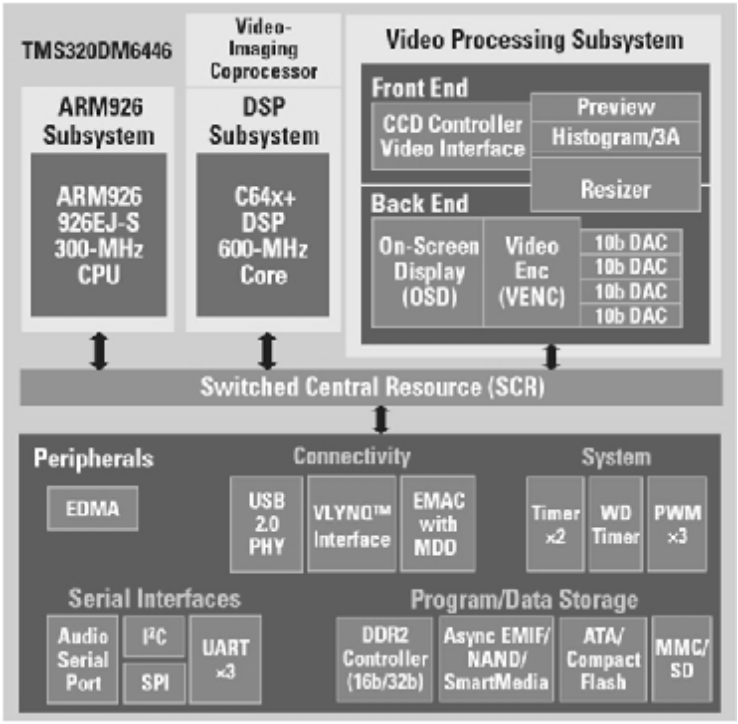

Fig. 1. Heterogeneous multicore DSP with video accelerators [4]

Modern DSPs on the other hand combine the features of low-power DSPs with features traditionally associated with general-purpose microprocessors, such as privilege modes, large general purpose register file, external memory access and memory protection. To issue and execute multiple instructions per clock cycle DSPs use VLIW (Very Long Instruction Word) techniques - in contrast to superscalar architecture of general purpose processors - to execute instructions in deterministic order and time frame.

Many new DSP families are focused on certain types of digital signal processing applications. While the DSP part of these ICs are more general purpose, they offer integrated specialized fixed function (although configurable) accelerators, for example for common video processing tasks, as shown on Fig. 1. These devices are able to offer a balance between ASIC-like cost and power and the flexibility of programmable DSPs [2].

Traditional DSP code development flow involved validating a C language model for correctness on a host PC and then porting that C code to hand coded DSP assembly language. This was both time consuming and error prone. Modern DSP development tool-set contains optimizing C/C++ compiler so the

whole application can reside in a C/C++ framework that is simpler to maintain, support, and upgrade.

1.4. General purpose microprocessors

For flexibility reasons, many system designs stick with off-the-shelf CPUs such as an x86 variant processor running on a standard server or desktop motherboard. Other CPU centric solutions deploy more embedded solutions using multi-function CPUs such as an ARM variant or embedded version of PowerPC processor. Modern general purpose processors furthermore have DSP like instruction set extension usually in the form of SIMD (Single Instruction, Multiple Data) vector instructions.

Today, many modern embedded CPUs are actually faster than low-cost fixed-point DSPs. But in signal processing applications, embedded general purpose CPUs typically can't compete with DSP processors on power and cost efficiency, and they usually lack the specialized on-chip integration and development tools needed for signal processing applications.

2. Video transcoding

Generally, there exist three transcoding architectures for homogeneous bit-rate transcoding: cascaded decoder and an encoder, closed-loop transcoder and open-loop transcoder. Homogeneous transcoding performs conversion between video bitstreams of the same standard, bit-rate transcoding changes only the video bit-rate with fixed spatial and temporal resolution.

The most straightforward transcoding architecture is to cascade a decoder and an encoder directly. In this architecture, the incoming source video stream is fully decoded, and then the decoded video re-encoded into the target video stream with desirable bit-rate or format. It is computationally very expensive, but often used way of video transcoding [5].

More computationally efficient solution is the closed-loop encoder that is a concatenation of a decoder and a simplified encoder. Rather than performing full-scale motion estimation, as in a standalone video encoder, the encoder reuses the motion vectors along with other information extracted from the input video bitstream. Thus, the motion estimation, which usually accounts for significant part of the encoder computation [6], is omitted.

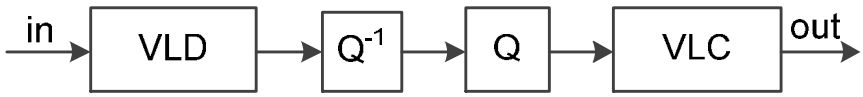

Fig. 2. Open-loop transcoder: VLD - Variable Length Decoder, Q – Quantizer, Q^{-1} – Dequantizer, VLC – Variable Length Coder

To address another major source of computational complexity, DCT (Discrete Cosine Transform), open-loop transcoding is used, since open-loop transcoding operates directly on the DCT coefficients. In open-loop transcoders, shown on Fig. 2, the process of video coding is reversed until the quantization step, a new quantizer value is calculated for lower bitrate, then the DCT coefficients requantized with this new quantizer value - without inverse DCT transformation -, and the rest of the video coding process is executed again with the new DCT coefficient values.

Disadvantage of open-loop transcoders is that the prediction feedback loop is broken, hence the name open-loop, that causes encoder/decoder predictor mismatch, called drift error and it may cause severe degradation to the video quality.

Actually there is the same idea behind all three transcoding architectures. To reduce bitrate, a higher quantizer step size is determined, so the amount of information contained in each picture will be lower, which means lower bitrate for the entire video stream.

2.1. DSP implementation

DSPs and SIMD instructions are effective for applications that are highly parallelizable and require execution of the same operation over and over again; they are less effective for applications with less uniform computational demands. Motion estimation, DCT and inverse DCT are well-suited for DSP execution: they require many identical, for example, multiply-accumulate operations that can be run in parallel. If the application requires frequent decision making and branches, however, DSP or SIMD may not be a good fit [7].

By simplifying the video transcoding architectures, processing stages with high computational requirements are left out of the transcoder design, what remains is code with frequent decision making and branches. This type of application can be difficult to implement efficiently on a traditional DSP architectures.

Some of the newer, more complex codecs (such as H.264) also require computationally demanding portions of the code to be finely interleaved with decision-making code. To help address this challenge, modern DSP and SIMD designs include conditional instruction execution to reduce the need for branches.

DSP application development starts with a C/C++ model on a host PC. With modern DSP development tools, nearly the same C/C++ code can be run on DSP devices as on general purpose CPUs. Also today DSP ICs are SoC (System on Chip) designs so they have many interfaces a PC has, for example Ethernet network controller. That enables the comparison of the same transcoding algorithm running on general purpose processors and on DSPs with the same source material from IP network.

3. Experimental results

We analyzed our implementation of open-loop MPEG-2 video transcoder [8] on three different general purpose processors and a DSP with the same input MPEG-2 SD video stream.

Table 1. Frame rate with different processors

	Core2Duo 2GHz	Athlon64 2GHz	P4 HT 2.8GHz	C6437 600MHz
t_F, us	4007	8269	5072	29954
fps	249.56	120.93	197.16	33.38
t_{Fn}, us	4007	8269	7100.8	8986.2
fps_n	249.56	120.93	140.83	111.28

In the first row of Table 1 the average frame time t_F in microseconds shown that was measured while transcoding 200 video frames from the input video stream. In the second row we can see the average number of frames (fps) that can be transcoded in one second.

Because not every processors we tested run on the same clock speed, the third row of Table 1 shows normalized values of the average transcoding time t_{Fn} for a theoretical 2GHz P4 and a theoretical 2GHz DSP, with the normalized frame per second (fps_n) values in the fourth row.

Conclusion

The experimental results show that C/C++ code running on a DSP has comparable speed clock-for-clock to older superscalar general purpose processors. However modern processors are much faster and the highest clock frequency of modern DSPs are only in the 1GHz range.

Running the same C/C++ code on DSP as on general purpose processor is the first step in DSP development, with optimized C/C++ code and using some optimized assembly code we expect achieving better DSP performance.

References

1. *Markandey V.,* Seshan N., Bardia P. Streaming multimedia challenges DSP design // EE Times, Sept. 2001. – http://www.eetimes.com/electronics-news/4163935/Streaming-multimedia-challenges-DSP-design

2. *Zhang O., Hua* Jia Y., Fu Zhao Y. Using DaVinci for video surveillance // DSP Design Line, April 2008. - http://www.eetimes.com/design/signal-processing-dsp/4017632/Using-DaVinci-for-video-surveillance

3. *Rash J.* Viewpoint: The importance of FPGA-to-ASIC solutions to accelerate CPU-based protocols // EE Times Design, March 2010. - http://www.eetimes.com/design/programmable-logic/4033359/Viewpoint-The-importance-of-FPGA-to-ASIC-solutions-to-accelerate-CPU-based-protocols

4. Texas Instruments. DaVinci™ Technology Overview, Sept. 2008. - http://focus.ti.com/lit/ml/sprb189b/sprb189b.pdf

5. *Ahmad I.,* Xiaohui Wei, Yu Sun, Ya-Qin Zhang. Video transcoding: an overview of various techniques and research issues // Multimedia IEEE Transactions. – 2005. – Vol. 7, № 5. – P. 793-804.

6. *Xin, J.,* Lin C.-W., Sun M.-T. Digital Video Transcoding // Proceedings of the IEEE. – 2005. – Vol. 93, № 1. – P. 84-97.

7. BDTI. Massively parallel processors for DSP // DSP Design Line, Jun. 2007. - http://www.eetimes.com/design/signal-processing-dsp/4017498/Massively-parallel-processors-for-DSP-part-1

8. *Formanek B.*, Ádám T. Rate control in MPEG-2 video bitrate transcoder // Proc. of the 10th International Carpathian Control Conference. – 2009. – P. 297-300.

Ключевые слова: видеотранскодер, без обратной связи, битрейт, квантование, цифровой сигнальный процессор, приложение, код C/C++.

[1] *Национальный технический университет Украины «КПИ»*

[2] *Университет г. Мишкольц, Венгрия*