RESUME's wavelet-based scalable video decoder

Hendrik Eeckhaut Mark Christiaens Harald Devos Philippe Faes
Dirk Stroobandt
Hendrik.Eeckhaut@elis.UGent.be
Parallel information systems - ELIS - Ghent University - Belgium
http://www.elis.ugent.be/resume

Abstract

In the RESUME project we developed a real-time FPGA prototype of a fully scalable, wavelet based video decoder which overcomes the complexity and bandwidth issues associated with scalable video.

1 Introduction

The RESUME-project (Reconfigurable Embedded Systems for Use in Scalable Multimedia Environments) explored the benefits of using reconfigurable hardware for the implementation of scalable multimedia applications by building an FPGA implementation of a real-time, scalable, wavelet-based video decoder. The term *scalable video* refers to a coding scheme that can easily accommodate changes in quality of service (QoS) without the need for transcoding. A scalable video stream can be decoded at varying frame rates, resolutions and image qualities by simply skipping dispensable parts in the video stream, only decoding those parts that will contribute to the displayed video. The algorithmic structure of the RESUME scalable video coder and decoder (codec) is shown in Figure 1 and is described in [2].

Such a scalable video codec has advantages for both the server (the provider of the content) and the clients. On the one hand the server scales well since it has to produce only one encoded video stream that can be broadcasted to all clients, irrespective of their QoS requirements. On the other hand the client (or the network) can easily adapt the decoding parameters to its needs. This way it is possible to optimize the use of the network, display, the required processing power, the required memory, ...

Scalability has a lot of advantages but comes at a cost. The decoding algorithm is computationally complex and really stresses the system bandwidth as it replaces the block-based DCT-approach with frame-based wavelets. This has a tremendous impact on the hardware architecture.

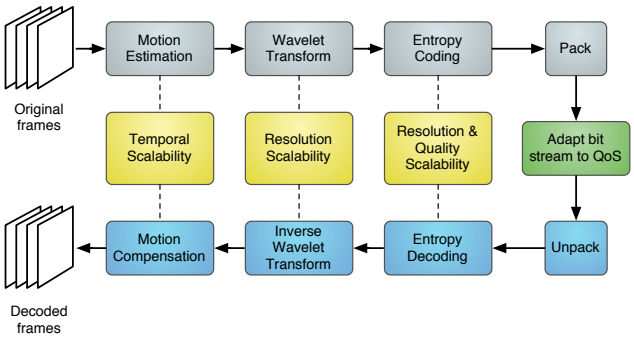

Figure 1. High-level overview of the codec.

2 Hardware implementation

We implemented the complete decoding pipeline of our custom wavelet-based scalable video codec on a PCI development board equipped with a Stratix FPGA S60 and 256 MiB of DDR SDRAM memory. The FPGA board is plugged into a standard PC with two monitors, one dedicated to displaying the decoded video, the other to interact with the system (Figure 2). The design goals were real-time, lossless decoding of CIF-sequences (352×288 pixels) at 25 frames per second.

Implementing a complete video codec is a complex undertaking that requires careful planning. We applied the following methodology. First the entire software code base was cleaned-up and we made sure that the algorithms used were properly understood. We chose to use a HW/SW-codesign approach leaving as much of the algorithm as possible in SW running on a CPU while implementing the time-critical parts in reconfigurable HW. The hardware design was implemented using Altera's SOPC (System-On-a-Programmable-Chip) Builder for component-based system integration. For each of the steps in the decoding pipeline we developed custom components to deliver the required hardware acceleration (Figure 3). Many compo-

Figure 2. Photograph of our wavelet-based scalable video decoder in action.

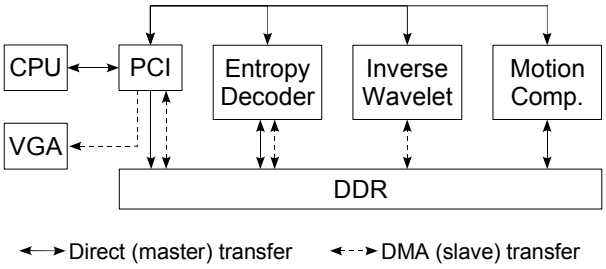

Figure 3. Simplified hardware overview.

nents have substantially different maximum clock frequencies. To accommodate this, the design is subdivided into multiple clock domains. Using SOPC Builder this can be achieved fairly easily by assigning different clocks to each of the components.

Most components of the design were highly optimized to achieve real-time decoding. The wavelet entropy decoder, very similar to AVC's CABAC, was exhaustively elaborated until it produced one decoded symbol per clock cycle by applying multilevel speculation and pipelining [2]. We also developed an automatic inverse discrete wavelet transform (IDWT) generator based on polyhedral techniques (loop transformations) [1]. This resulted in a line-based IDWT especially tailored to the specific access pattern of the external on-board DDR-memory. For all development (hardware and software) we applied a write-tests-first strategy and used well founded engineering techniques as code reuse, refactoring, regression testing and continuous integration to continuously guard the quality of the entire design.

The major bottlenecks in the design were the limited bandwidth of the PCI-bus and the DDR-memory. Trading in the conventional block-based DCT-transform for the wavelet-transform results in calculations on larger data blocks: wavelet sub bands instead of macro blocks. The larger intermediate results no longer fit in the internal memories of the FPGA and force off-chip buffering. Of course, this results in large bandwidth requirements between the external memory and the FPGA. To alleviate this problem, much effort has been put in optimizing this communication through the use of DMA transfers in efficient burst mode.

Now that we have a design capable of lossless decoding, we can investigate the options of reconfiguration with smaller or slower designs to adapt the hardware resource usage to lower quality requirements. This way we can effectively link the video scalability to real hardware scalability.

3 Conclusions

To our knowledge, we present the first complete hardware design of a real-time, fully scalable wavelet-based video decoder. This result was only possible by using a cutting edge methodology. Firstly, the design was performed using state-of-the-art tools such as Altera's SOPC Builder (for a component-based design), the WRaP-IT/URUK tool-suite (for automatic loop transformations) and CLooGVHDL (for automatically generating VHDL from loop nests). Secondly, some components in the design, such as the inverse discrete wavelet transform and the arithmetic decoder, are highly optimized. Compared to a software implementation at 2 GHz (AMD64 3500+) the FPGA-implementation at approximately 55 MHz is 3 times faster. Finally, the quality of the entire design was carefully guarded using a regression framework based on the best practices from the software world (using Apache Maven and Continuum).

Acknowledgment

This research is supported by the I.W.T. Vlaanderen, grant 020174, the F.W.O., grant G.0021.03, GOA project 12.51B.02 of Ghent University and the Altera university program. P. Faes is supported by a PhD grant from the I.W.T. Vlaanderen.

References

[1] H. Devos, K. Beyls, M. Christiaens, J. Van Campenhout, E. H. D'Hollander, and D. Stroobandt. Finding and applying loop transformations for generating optimized FPGA implementations. *Transactions on High Performance Embedded Architectures and Compilers, LNCS 4050*, 1(1):151–170, 2007.

[2] H. Eeckhaut, M. Christiaens, H. Devos, and D. Stroobandt. Implementing a hardware-friendly wavelet entropy codec for scalable video. In *Proceedings of SPIE: Wavelet Applications in Industrial Processing III*, volume 6001, pages 169–179, October 2005.