Development of Multi-Core Video Decoding Platforms based on High-Level Architecture Simulations

Florian Seitner, Michael Bleyer (Faculty Mentor) and Margrit Gelautz (Faculty Mentor)
Institute of Software Technology and Interactive Systems
Vienna University of Technology
Vienna, Austria
Email: {seitner, bleyer, gelautz}@ims.tuwien.ac.at

Abstract — *The high computational demands of state-of-the-art video coding standards pose serious challenges on strongly resource-restricted architectures. For reaching the performance specifications, specialized multi-core architectures for video processing are becoming more and more popular. In this work, we introduce an high-level simulator for supporting the development of such decoding platforms. Our system combines all available information such as hardware measurements, profilings and human expertise. Based on this input, the behaviour of the final architecture running a parallel video decoder is estimated. Using this information, adaptations of the current hardware or software design can be done. The simulator shall aid in developing efficient and application-optimized decoding systems.*

I. Introduction

The H.264 video standard [1] is currently used in a wide range of video-related areas such as video content distribution and television broadcasting. Compared to preceding standards such as MPEG-2 and MPEG-4 SP/ASP, improved coding efficiency is reached by introducing more advanced pixel processing algorithms (e.g. quarter-pixel motion estimation) as well as by the use of more sophisticated algorithms for predicting syntax elements from neighbouring macroblocks (e.g. context-adaptive VLC). These new coding tools result in significantly increased CPU and memory loads required for decoding the video stream. In environments of limited processing power such as embedded systems, the high computational demands pose a serious challenge for practical H.264 implementations. Multi-core systems provide an elegant and power-efficient solution to overcome these performance limitations.

The design of such a specialized multi-processor decoding architecture is a non-trivial task. For exploiting the processing power of a multi-core system most efficiently, an equal workload between the cores must be achieved. Apart from the system's usage this also influences the required buffer sizes between the cores for compensating differences in the workload. However, the significant workload differences in typical video decoding systems make this balancing a challenging task.

Figure 1 visualizes the structure of the H.264 decoding process. The computational complexity of a macroblock's parsing and deblocking functions is strongly bitrate dependent. While the parsing complexity typically raises with the macroblock's number of bits and syntax elements, the deblocking filter is applied more aggressively for low bitrates. For the pixel-based decoding tasks (e.g. intra prediction and motion compensation) a large variety of possible macroblock coding modes exists. The coding options such as the prediction type (i.e. for H.264 skipped, intra and inter prediction) and the macroblock partitioning influence the decoding complexity significantly.

For multi-core video decoding systems, predicting the run-time behaviour is not straight forward. Differences in the workload, algorithmic dependencies and stalls due to resource limitations (e.g. size restrictions of communication buffers) must be considered. Furthermore, video decoding platforms are often based on heterogeneous architectures for addressing the execution behaviour of the individual decoding tasks more efficiently. For example, the highly conditional parsing and entropy decoding functions require processors with efficient branch execution. Depending on which processor a decoding task is executed, differences in the decoder run-time will occur.

The dynamics and heterogeneity of video decoding systems often result in the following questions:

- Can we reach the specified decoding requirements on a specified multi-core architecture?
- Which architecture is required to handle a certain set of videos (i.e. a set of streams that represents the common input characteristics of an application such as DVB-T)?
- What is the optimal decoder hardware and software partitioning for this architecture?

These questions have a major impact on the architecture decisions and should be addressed before implementing a video decoding system.

For solving these questions, assumptions about the architecture requirements regarding the computational decoding complexity are typically made. They allow us to decide on the hardware components and the software

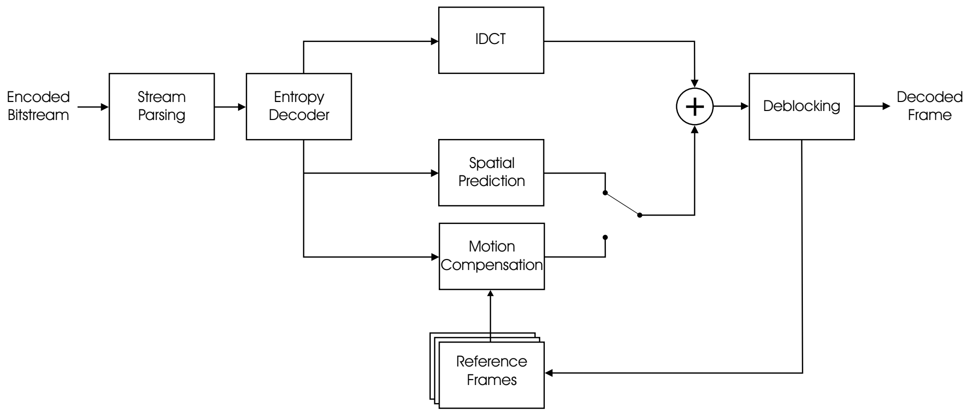

Figure 1: The H.264 decoding process. After parsing and entropy decoding the syntax elements of a macroblock, a spatial or temporal prediction of this macroblock is computed. This prediction is added to the inverse transformed (IDCT) residual information and a deblocking filter removes blocking artefacts introduced during the quantisation step.

structure of our decoder. Complexity estimation techniques are therefore of prime importance. In recent years, advanced techniques for estimating the run-time of a program have evolved.

Static algorithm analysis and path analysis techniques have been introduced in [2]. These techniques analyse an algorithm's definition (e.g. its source code) for estimating the upper and lower run-time bounds of a program. For considering the impact of the input data on the program execution, dynamic profiling methods [3] have been developed. These methods observe the program's execution behaviour during the run-time. This allows us to measure the complexity of the decoding functions for individual input data. Hardware simulations and HW/SW-Codesign methods [4] provide accurate run-time information but require labour intensive adaptation of the hardware and decoder software before first run-time estimations are possible.

All these approaches above can provide us with information about the complexity requirements of our video decoding system. However, interpretation of the complexity information in the context of a multi-core system is not straight-forward. Algorithmic dependencies and resource limitations result in run-time constraints between the processing units. For highly dynamic multi-core decoder systems, making predictions about the parallel decoding system's behaviour is hardly possible.

In our work, we introduce a simulator for supporting the development process of multi-core video decoding systems. It estimates the basic parameters such as the execution time, the memory transfers and the power consumption of the decoder system. Instead of running the partitioned decoding tasks and using exactly specified interfaces for connecting them together, only an abstract information about the macroblocks' runtime (e.g. decoding complexity) is required. This is a major advantage of our system, since the system designer typically gets this information without building the complete hardware architecture or without partitioning the software of the decoder. Architecture evaluations are possible before the decoder architecture is effectively built. Additionally, the Partition Assessment Tool (PAT) allows software design explorations for supporting the partitioning of the decoder software.

Acknowledgments

This work has been supported by ON DEMAND Microelectronics AG [5] and the Austrian Federal Ministry of Transport, Innovation, and Technology under the FIT-IT Project 812429.

References

[1] ITU-T and ISO/IEC. *Advanced video coding for generic audiovisual services (ITU Rec. H.264 — ISO/IEC 14496-10)*. ITU-T and ISO/IEC, March 2005.

[2] Peter P. Puschner and Christian Koza. Calculating the maximum execution time of real-time programs. *Journal of Real-Time Systems*, 1(2):159–176, 1989.

[3] Susan L. Graham, Peter B. Kessler, and Marshall K. McKusick. gprof: a call graph execution profiler. In *SIGPLAN Symposium on Compiler Construction*, pages 120–126, 1982.

[4] Peter Voigt Knudsen and Jan Madsen. Pace: A dynamic programming algorithm for hardware/software partitioning. In *Proceedings of the Int. Workshop on Hardware-Software Co-Design*, pages 85–92, 1996.

[5] ON DEMAND Microelectronics. http://www.odmsemi.com, 2008.