Hardware/Software Co-Optimization for Low-Energy Embedded Systems

Proposal of DEPS (Dynamic Energy/Performance Scaling) Framework

Hiroaki Takada
Center for Embedded Computing Systems
Graduate School of Information Science, Nagoya Univ.
C3-1 (631), Furo-cho, Chikusa-ku, Nagoya, 464-8603 Japan
Email: hiro@ertl.jp

I. INTRODUCTION

We, a joint research team of Nagoya Univ., Kyushu Univ., Toshiba Corp., and Ritsumeikan Univ., conducted a project titled "Hardware/Software Co-Optimization for Low-Energy Embedded Systems" from FY2005 to FY2010. The goal of the project is to develop technologies and methodologies for reducing energy consumption of an embedded system dramatically while guaranteeing the required QoS.

Application software executed in an embedded system is usually known in design time, we take the approach to fully exploit the characteristics of the application software for reducing its energy consumption.

Because many embedded systems require real-time properties, our concrete goal is defined to minimize the average energy consumption of a system while guaranteeing its real-time property (worst case response times, in more concrete). We focus on the energy consumption of the processors and the memory system of an embedded system.

Optimizing both hardware and software of an embedded system in a consistent manner is a promising approach for reducing its energy consumption.

II. DEPS FRAMEWORK

The concept of DEPS (Dynamic Energy and Performance Scaling is proposed as the central approach of our project [1,2].

A. Concept of DEPS

Though DVFS (Dynamic Voltage and Frequency Scaling) is widely adopted approach to reduce energy consumption of processors, the operating voltage of processors recently becomes quite low and the effectiveness of DVFS is reduced.

On the other hand, modern high performance processors are getting more and more complex and their energy efficiency is degraded. This is because deep pipelining, cache memories, branch predictor, and other complex mechanism for faster processing consume large energy.

From these observations, the following generalization of DVFS emerges as a promising approach. When processing request is large, a high performance and low energy-efficient processor (or configuration of processor) is used. Otherwise, a low performance and high energy-efficient processor (or configuration of processor) is used instead. Here, the term "configuration of processor" includes not only the operating voltage and frequency of the processor, but also the size and configuration of the cache memories and turning on/off of the branch predictor.

We name this approach as dynamic energy and performance scaling, or DEPS in short.

B. Overview of the DEPS Framework

For realizing the DEPS framework, we investigated on the following four topics.

1) 1. Hardware technologies to choose optimal tradeoff point between energy consumption and performance and IP cores, including processors, implementing them: We designed a multi-performace processor (MPP) core which can change the operating voltage/frequency and the size and way count of the cache memory, and implemented a SoC with multiple MPP cores [3,4].

2) Techniques to examine the characteristics of application software and tools implementing them: In order to control DEPS mechanism precisely, the feature of application softwre should be known. We investigated on the techniques to examine the characteristics of application software by analyzing a large amount of the execution traces of the software. Execution traces including the information on energy consumption are gathered using an instruction set simulator and a tool to estimate consumed energy from the traces [3]. Then, the colleced traces are analized with the technique called execution trace mining [5].

3) Static optmizing methods to determine the optimal system configuration in design time and tools implementing them: For embedded systems, in which executed application software is known in design time, the system configuration should be determined in design time for reducing runtime overhead. For this purpose, we investigated on the methods to optimally determine the assignment of tasks to processors, the

This work was supported by Core Research for Evolutional Science and Technology (CREST) of the Japan Science and Technology (JST).

allocation of execution time budget to tasks, and layout of memory areas [6,7,8].

4) Dynamic optimizing methods to determine the optimal system configuration in runtime and a real-time operating system (RTOS) implementing them: In runtime, the execution times of the tasks are usually smaller than their maximum values, with which static optimization has been executed, and some slack times occur as the result. We developed an RTOS, which calculate the slack times and determine the optimal configuration of processors considering them.

C. Software Development Environment for DEPS

We developed a software development environment for DEPS framework, which is to automate the optimization of embedded system as far as possible.

Figure 1 shows the overview of our software development environment. The optimization is done in three phases.

The first phase is the intra-task optimization. The characteristics of each task are analyzed from execution traces obtained by an instruction set simulator. In this phase, checkpoints at which the configuration of the processor can be changed are inserted in a task. Memory allocation within a task is also determined in this phase.

The second phase is the inter-task optimization. Based on the results obtained at the first phase, the characteristic and behavior of application software consisting of multiple tasks are analyzed and optimized. In this phase, task assignment to processor is determined and execution time budget is distributed to tasks. Inter-task partition of scratch-pad memory is also done in this phase.

The last phase is the runtime optimization phase. The RTOS achieves maximal energy savings using two management tables generated at the previous phase.

III. Other Research Topics and Results

In addition to the DEPS framework described above, we investigated on several research topics in the project. Some of the results are as follows.

- requirements analysis method considering energy consumption
- algorithm theory for low energy consumption [9]
- optimization of voltage margin
- hybrid memory
- memory architecture to compensate for delay variation [10]
- gated flip-flop [11]

Acknowledgment

I would like to express my gratitude to all members of our research project from Nagoya Univ., Kyushu Univ., Toshiba Corp, and Ritsumeikan Univ.

References

[1] G. Zeng, H. Tomiyama, and H. Takada, "A Generalized Framework for Energy Savings in Hard Real-Time Embedded Systems," IPSJ Trans. on System LSI Design Methodology, vol.2, pp.167-179 (2009).

[2] H. Takase et al., "An Integrated Optimization Framework for Reducing the Energy Consumption of Embedded Real-Time Applications," Int'l Symposium on Low Power Electronics and Design, to appear (2011).

[3] T. Ishihara and M. Goudarzi, "System-Level Techniques for Estimating and Reducing Energy Consumption in Real-Time Embedded Systems," Int'l SoC Design Conference, pp.67-72 (2007).

[4] T. Ishihara et al., "AMPLE: An Adaptive Multi-Performance Processor for Low-Energy Embedded Applications," IEEE Symposium on Application Specific Processors, pp.83-88 (2008).

[5] T. Tatematsu et al., "Checkpoints Extraction Using Execution Traces for Intra-Task DVFS in Embedded Systems," IEEE 6th Int'l Symposium on Electronic Design, Test and Applications, pp. 19-24 (2011).

[6] Y. Ishitobi, T. Ishihara, and H. Yasuura, "Code and Data Placement for Embedded Processors with Scratchpad and Cache Memories," Journal of Signal Processing Systems (2008).

[7] H. Takase, H. Tomiyama, and H. Takada, "Partitioning and Allocation of Scratch-Pad Memory for Priority-Based Preemptive Multi-Task Systems," Design, Automation and Test in Europe (2010).

[8] L. Gauthier and T. Ishihara, "Optimal Stack Frame Placement and Transfer for Energy Reduction Targeting Embedded Processors with Scratch-Pad Memories," IEEE Workshop on Embedded Systems for Real-Time Multimedia, pp.116-125 (2009).

[9] T. Yokoyama et al., "Analyzing and Optimizing Energy Efficiency of Algorithms on DVS Systems: A First Step towards Algorithmic Energy Minimization," Asia and South Pacific Design Automation Conference, pp.727-732 (2009).

[10] M. Goudarzi, T. Ishihara, "SRAM Leakage Reduction by Row/Column Redundancy under Random Within-die Delay Variation," IEEE Trans. on VLSI Systems (2009).

[11] T. Okuhira and T. Ishihara, "Unification of Multiple Gated Flip-Flops for Saving the Power Consumption of Register Circuits, Int'l Conference on Embedded Systems and Intelligent Technology, vol.1, pp.115 (2010).

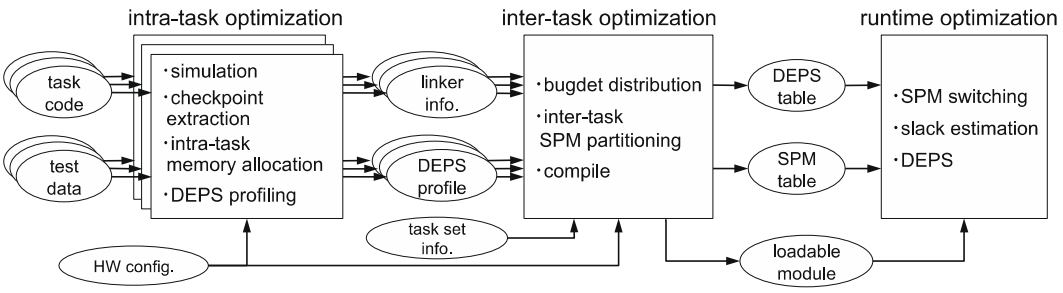

Figure 1. Overview of the Software Development Environment