COMPILERS AND OPERATING SYSTEMS FOR LOW POWER

COMPILERS AND OPERATING SYSTEMS FOR LOW POWER

Edited by
LUCA BENINI
University of Bologna

MAHMUT KANDEMIR
The Pennsylvania State University

J. RAMANUJAM
Louisiana State University

Kluwer Academic Publishers
Boston/Dordrecht/London

Contents

List of Figures

List of Tables

Contributing Authors

Nevine AbouGhazaleh	University of Pittsburgh, USA
Andrea Acquaviva	University of Bologna, Italy
Alex Arenas	Universitat Rovira i Virgili, Spain
Itsujiro Arita	Kyushu Institute of Technology, Japan
Luca Benini	University of Bologna, Italy
Ricardo Bianchini	Rutgers University, USA
Bruno Bougard	IMEC, Belgium
Francky Catthoor	IMEC, Belgium
Bruce Childers	University of Pittsburgh, USA
Paolo D'Alberto	University of California–Irvine, USA
Michael Franz	University of California–Irvine, USA
Rajesh Gupta	University of California–Irvine, USA
Taliver Heath	Rutgers University, USA
Lei He	University of California–Los Angeles, USA
Jerry Hom	Rutgers University, USA
Ulrich Kremer	Rutgers University, USA
Jesus Labarta	Universitat Politecnica de Catalunya, Spain
Weiping Liao	University of California–Los Angeles, USA
Suet-Fei Li	University of California–Berkeley, USA
Rami Melhem	University of Pittsburgh, USA
M. Angels Moncusí	Universitat Rovira i Virgili, Spain
Daniel Mossé	University of Pittsburgh, USA
Alexandru Nicolau	University of California–Irvine, USA
Eduardo Pinheiro	Rutgers University, USA
Jan Rabaey	University of California–Berkeley, USA
Bruno Riccó	University of Bologna, Italy
Toshinori Sato	Kyushu Institute of Technology, Japan
K.C. Shashidhar	IMEC, Belgium
Greg Stitt	University of California–Riverside, USA
Roy Sutton	University of California–Berkeley, USA
Enrique V. Carrera	Rutgers University, USA
Frank Vahid	University of California–Riverside, USA
Arnout Vandecappelle	IMEC, Belgium
Alexander Veidenbaum	University of California–Irvine, USA

Preface

In the last ten years, power dissipation has emerged as one of the most critical issues in the development of large-scale integrated circuits, and electronic systems in general. Technology scaling is not the only cause for this trend: in fact, we are moving toward a world of pervasive electronics, where our cars, houses, and even our environment and our bodies will be linked in a finely-knit network of communicating electronic devices capable of complex computational tasks materializing a vision of "ambient intelligence," the ultimate goal of embedded computing. Today, power consumption is probably the main obstacle in the realization of this vision: current electronic systems still require too much power to perform critical ambient intelligence tasks (e.g., voice processing, vision, wireless communication). For this reason, power, or energy (i.e., power-performance ratio) minimization is now aggressively targeted in all the phases of electronic system design.

While early low-power (or energy-efficient) design focused on technology and hardware optimization, it is now clear that software power optimization is an equally critical goal. Most of complex integrated systems are highly programmable. In fact, the new millennium has seen the rapid diffusion of embedded processor cores as the basic computational workhorse for large-scale integrated systems on silicon, and today we are witnessing the rebirth of multiprocessor architectures, fully integrated on a single silicon substrate. It is therefore obvious that the power consumption of integrated systems dominated by core processors and memories is heavily dependent on the applications they run and the middleware supporting them.

In general, we can view the software infrastructure as layered in applications and run-time support middleware (often called "operating system"). Applications control the user-level functionality of the system, but they interface to the SoC platform via hardware abstraction layers provided by the middleware. Software energy minimization can be tackled with some hope of success only if both application-level software and middleware are both optimized form maximum energy efficiency. The Compilers and Operating Systems for Low Power (COLP) Workshop aims at creating a forum that brings together researchers operating in both application-level energy optimization and low-power operat-

ing systems. The main objective of this initiative is to create opportunities for cross-fertilization between closely related areas that can greatly benefit from a tighter interaction. Papers presented at COLP are work-in-progress and are selected based on their potential for stimulating thoughts and creative discussions.

This book is the result of a careful (and sometimes painful) process of selection and refinement of the most significant contributions to the 2001 edition of COLP. The editors have first selected the papers based both on reviewer evaluations and on feedback from the audience at the oral presentation. They have then solicited an extended version of the papers, in a format more suitable for archival publications. The extended versions have then been reviewed by the editors to ensure consistency. The results of this "distillation" process have been collected in this book, which we hope will bring the reader a wealth of fresh and valuable ideas for further research as well as technology transfer.

Organization

The book is divided into twelve chapters. The first six chapters focus on low energy operating systems, or more in general, energy-aware middleware services. The following five chapters are centered on compilation and code optimization. Finally, the last chapter takes a more general viewpoint on mobile computing.

Chapter 1, entitled "Low Power Operating System for Heterogeneous Wireless Communication Systems" is contributed by Suet-Fei Li, Roy Sutton, and Jan Rabaey, from UC Berkeley. The chapter describes an ultra-low overhead operating system for wireless microsensors and compares it with more traditional embedded operating systems.

Chapter 2, "Low Power Approach in a Modified Dual Priority Scheduling for Hard Real-Time Systems" (by M. Angels Moncusí, A. Arenas, and J. Labarta from Universitat Rovira i Virgili and Universitat Politecnica de Catalunya) deal with task scheduling, one of the most classical problems in real-time operating systems, and investigates a novel dual- priority algorithm with high energy efficiency.

The third chapter, contributed by N. Nevine AbouGazelah, D. Mossé, R. Melhem, and B. Childers (from University of Pittsburgh) entitled "A Restricted Model for the Optimal Placement of Power Management Points in Real Time Applications" deals with an important issue at the boundary between applications and operating systems, namely the optimal insertion of systems calls that dynamically change the supply voltage (and operating frequency) during the execution of an application.

The fourth chapter, by A. Acquaviva, L. Benini and B. Riccó (Università di Bologna), is entitled "Energy Characterization of Embedded Real-Time Oper-

ating Systems." The chapter describes a methodology for characterizing the energy cost of most primitives and function calls in embedded operating systems.

Chapter 5, by E. Pinheiro, R. Bianchini, E. Carrera and T. Heath (Rutgers University), is entitled "Load Balancing and Unbalancing for Power and Performance in cluster-Based Systems" and it deals with an important emerging topic, namely low-energy multiprocessors. The chapter gives a fresh look at load balancing issues in cluster-based systems when energy constraints are tight.

Chapter 6 closes the first group. It is entitled "Energy Management of Virtual Memory on Diskless Devices" (by J. Hom and U. Kremer, Rutgers University) and it deals with virtual memory, one of the basic hardware abstraction layers provided by standard operating systems.

The next chapter, entitled "Propagating Constants Past Software to Hardware Peripherals in Fixed-Application Embedded Systems," contributed by G. Stitt and F. Vahid, discusses how propagating application-level constant to hardware improves both power and form factor, leading to up to 2-3 times reductions in peripheral size.

In Chapter 8, entitled "Constructive Timing Violation for Improving Energy Efficiency," T. Sato and I. Arita present a technique that relies on a fault-tolerance mechanism and speculative execution to save power. Their technique, called *constructive timing violation*, guarantees that the timing constraints for critical paths are not violated.

In the next chapter, entitled "Power Modeling and Reduction of VLIW Processors," the authors W. Liao and L. He present an in-depth study of power behavior of a VLIW architecture, and develop an infrastructure which can be used for architecture-based as well as compiler studies.

Chapter 10, entitled "Low Power Design of Turbo Decoder Module with Exploration of Power-Performance Tradeoffs," demonstrates how a systematic data transfer and storage exploration methodology helps characterize energy and performance behavior of Turbo Coding. Vandecappelle, Bougard, Shashidbar, and Catthoor also discuss the cycle budget-energy tradeoff.

In the next chapter, "Static Analysis of Parameterized Loop Nests for Energy Efficient Use of Data Caches," P. D'Alberto, A. Nicolau, A. Veidenbaum, and R. Gupta demonstrate that the compiler analysis of loop with regular access patterns can reveal useful information for optimizing power.

Finally, in Chapter 12, entitled "A Fresh Look at Low-Power Mobile Computing," M. Franz presents a technique that allows large portions of applications to be offloaded to a base station for execution.

We believe that, with the proliferation of power-constrained devices, energy optimizations will become even more important in the future. Consequently, it is hard to imagine that architectural and circuit-level optimizations alone will

provide the required level of energy efficiency for demanding applications of next generation computing. The research papers presented here do not only demonstrate state-of-the-art, but they also prove that, to obtain the best energy/performance characteristics, compiler, system software, and architecture must work together.

Acknowledgments

This book grew out of the Workshop on Compilers and Operating Systems, 2001 (COLP 01). We acknowledge the active contribution of the program committee of COLP 01: Eduard Ayguade, R. Chandramouli, Bruce Childers, Marco Cornero, Rudi Eigenmann, Manish Gupta, Rajiv Gupta, Mary Jane Irwin, Uli Kremer, Rainer Leupers, Diana Marculescu, Enric Musoll, Anand Sivasubramaniam, Mary Lou Soffa, Vamsi K. Srikantam, Chau-Wen Tseng, Arnout Vandecappelle, and N. Vijaykrishnan. In addition, we thank the following reviewers for their thoughtful reviews of the initial submissions to the workshop: Bharadwaj Amrutur, Eui Young Chung, Anoop Iyer, Miguel Miranda, Phillip Stanley-Marbell, Emil Talpes, Chun Wong, and and Peng Yang. The feedback from the audience at the COLP 01 workshop is greatly appreciated.

We sincerely thank Alex Greene and Melissa Sullivan, and the editorial team at Kluwer for their invaluable help, enthusiasm and encouragement throughout this project. We gratefully acknowledge the support of the U.S. National Science Foundation through grants CCR–9457768, CCR–0073800, and CCR–0093082 during this project.

LUCA BENINI, MAHMUT KANDEMIR, J. RAMANUJAM