Michael Beigl
Paul Lukowicz (Eds.)

LNCS 3432

Systems Aspects in Organic and Pervasive Computing – ARCS 2005

18th International Conference on Architecture of Computing Systems Innsbruck, Austria, March 2005, Proceedings

Springer

Michael Beigl Paul Lukowicz (Eds.)

Systems Aspects in Organic and Pervasive Computing – ARCS 2005

18th International Conference on
Architecture of Computing Systems
Innsbruck, Austria, March 14-17, 2005
Proceedings

Springer

Volume Editors

Michael Beigl
University of Karlsruhe
Telecooperation Office (TecO)
Vincenz-Priessnitz-Strasse 1
76131 Karlsruhe, Germany
E-mail: michael@teco.edu

Paul Lukowicz
University for Health Science, Medical Informatics and Technology
Innsbruck, Austria
E-mail: paul.lukowicz@umit.at

Library of Congress Control Number: 2005921801

CR Subject Classification (1998): C.2, C.5.3, D.4, D.2.11, H.3.5, H.4, H.5.2

ISSN 0302-9743
ISBN 3-540-25273-8 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 11398196 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3432

Lecture Notes in Computer Science

For information about Vols. 1–3322

please contact your bookseller or Springer

Vol. 3452: F. Baader, A. Voronkov (Eds.), Logic Programming, Artificial Intelligence, and Reasoning. XI, 562 pages. 2005. (Subseries LNAI).

Vol. 3436: B. Bouyssounouse, J. Sifakis (Eds.), Embedded Systems Design. XV, 492 pages. 2005.

Vol. 3432: M. Beigl, P. Lukowicz (Eds.), Systems Aspects in Organic and Pervasive Computing - ARCS 2005. X, 265 pages. 2005.

Vol. 3418: U. Brandes, T. Erlebach (Eds.), Network Analysis. XII, 471 pages. 2005.

Vol. 3416: M. Böhlen, J. Gamper, W. Polasek, M.A. Wimmer (Eds.), E-Government: Towards Electronic Democracy. XIII, 311 pages. 2005. (Subseries LNAI).

Vol. 3415: P. Davidsson, B. Logan, K. Takadama (Eds.), Multi-Agent and Multi-Agent-Based Simulation. X, 265 pages. 2005. (Subseries LNAI).

Vol. 3414: M. Morari, L. Thiele (Eds.), Hybrid Systems: Computation and Control. XII, 684 pages. 2005.

Vol. 3412: X. Franch, D. Port (Eds.), COTS-Based Software Systems. XVI, 312 pages. 2005.

Vol. 3411: S.H. Myaeng, M. Zhou, K.-F. Wong, H.-J. Zhang (Eds.), Information Retrieval Technology. XIII, 337 pages. 2005.

Vol. 3410: C.A. Coello Coello, A. Hernández Aguirre, E. Zitzler (Eds.), Evolutionary Multi-Criterion Optimization. XVI, 912 pages. 2005.

Vol. 3409: N. Guelfi, G. Reggio, A. Romanovsky (Eds.), Scientific Engineering of Distributed Java Applications. X, 127 pages. 2005.

Vol. 3408: D.E. Losada, J.M. Fernández-Luna (Eds.), Advances in Information Retrieval. XVII, 572 pages. 2005.

Vol. 3406: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing. XVII, 829 pages. 2005.

Vol. 3404: V. Diekert, B. Durand (Eds.), STACS 2005. XVI, 706 pages. 2005.

Vol. 3403: B. Ganter, R. Godin (Eds.), Formal Concept Analysis. XI, 419 pages. 2005. (Subseries LNAI).

Vol. 3401: Z. Li, L. Vulkov, J. Waśniewski (Eds.), Numerical Analysis and Its Applications. XIII, 630 pages. 2005.

Vol. 3398: D.-K. Baik (Ed.), Systems Modeling and Simulation: Theory and Applications. XIV, 733 pages. 2005. (Subseries LNAI).

Vol. 3397: T.G. Kim (Ed.), Artificial Intelligence and Simulation. XV, 711 pages. 2005. (Subseries LNAI).

Vol. 3396: R.M. van Eijk, M.-P. Huget, F. Dignum (Eds.), Agent Communication. X, 261 pages. 2005. (Subseries LNAI).

Vol. 3395: J. Grabowski, B. Nielsen (Eds.), Formal Approaches to Software Testing. X, 225 pages. 2005.

Vol. 3394: D. Kudenko, D. Kazakov, E. Alonso (Eds.), Adaptive Agents and Multi-Agent Systems III. IIX, 313 pages. 2005. (Subseries LNAI).

Vol. 3393: H.-J. Kreowski, U. Montanari, F. Orejas, G. Rozenberg, G. Taentzer (Eds.), Formal Methods in Software and Systems Modeling. XXVII, 413 pages. 2005.

Vol. 3391: C. Kim (Ed.), Information Networking. XVII, 936 pages. 2005.

Vol. 3390: R. Choren, A. Garcia, C. Lucena, A. Romanovsky (Eds.), Software Engineering for Multi-Agent Systems III. XII, 291 pages. 2005.

Vol. 3389: P. Van Roy (Ed.), Multiparadigm Programming in Mozart/OZ. XV, 329 pages. 2005.

Vol. 3388: J. Lagergren (Ed.), Comparative Genomics. VIII, 133 pages. 2005. (Subseries LNBI).

Vol. 3387: J. Cardoso, A. Sheth (Eds.), Semantic Web Services and Web Process Composition. VIII, 147 pages. 2005.

Vol. 3386: S. Vaudenay (Ed.), Public Key Cryptography - PKC 2005. IX, 436 pages. 2005.

Vol. 3385: R. Cousot (Ed.), Verification, Model Checking, and Abstract Interpretation. XII, 483 pages. 2005.

Vol. 3383: J. Pach (Ed.), Graph Drawing. XII, 536 pages. 2005.

Vol. 3382: J. Odell, P. Giorgini, J.P. Müller (Eds.), Agent-Oriented Software Engineering V. X, 239 pages. 2005.

Vol. 3381: P. Vojtáš, M. Bieliková, B. Charron-Bost, O. Sýkora (Eds.), SOFSEM 2005: Theory and Practice of Computer Science. XV, 448 pages. 2005.

Vol. 3379: M. Hemmje, C. Niederee, T. Risse (Eds.), From Integrated Publication and Information Systems to Information and Knowledge Environments. XXIV, 321 pages. 2005.

Vol. 3378: J. Kilian (Ed.), Theory of Cryptography. XII, 621 pages. 2005.

Vol. 3377: B. Goethals, A. Siebes (Eds.), Knowledge Discovery in Inductive Databases. VII, 190 pages. 2005.

Vol. 3376: A. Menezes (Ed.), Topics in Cryptology – CT-RSA 2005. X, 385 pages. 2004.

Vol. 3375: M.A. Marsan, G. Bianchi, M. Listanti, M. Meo (Eds.), Quality of Service in Multiservice IP Networks. XIII, 656 pages. 2005.

Vol. 3374: D. Weyns, H.V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems. X, 279 pages. 2005. (Subseries LNAI).

Vol. 3372: C. Bussler, V. Tannen, I. Fundulaki (Eds.), Semantic Web and Databases. X, 227 pages. 2005.

Vol. 3370: A. Konagaya, K. Satou (Eds.), Grid Computing in Life Science. X, 187 pages. 2005. (Subseries LNBI).

Vol. 3369: V.R. Benjamins, P. Casanovas, J. Breuker, A. Gangemi (Eds.), Law and the Semantic Web. XII, 249 pages. 2005. (Subseries LNAI).

Vol. 3368: L. Paletta, J.K. Tsotsos, E. Rome, G.W. Humphreys (Eds.), Attention and Performance in Computational Vision. VIII, 231 pages. 2005.

Vol. 3366: I. Rahwan, P. Moraitis, C. Reed (Eds.), Argumentation in Multi-Agent Systems. XII, 263 pages. 2005. (Subseries LNAI).

Vol. 3365: G. Mauri, G. Păun, M.J. Pérez-Jiménez, G. Rozenberg, A. Salomaa (Eds.), Membrane Computing. IX, 415 pages. 2005.

Vol. 3363: T. Eiter, L. Libkin (Eds.), Database Theory - ICDT 2005. XI, 413 pages. 2004.

Vol. 3362: G. Barthe, L. Burdy, M. Huisman, J.-L. Lanet, T. Muntean (Eds.), Construction and Analysis of Safe, Secure, and Interoperable Smart Devices. IX, 257 pages. 2005.

Vol. 3361: S. Bengio, H. Bourlard (Eds.), Machine Learning for Multimodal Interaction. XII, 362 pages. 2005.

Vol. 3360: S. Spaccapietra, E. Bertino, S. Jajodia, R. King, D. McLeod, M.E. Orlowska, L. Strous (Eds.), Journal on Data Semantics II. XI, 223 pages. 2005.

Vol. 3359: G. Grieser, Y. Tanaka (Eds.), Intuitive Human Interfaces for Organizing and Accessing Intellectual Assets. XIV, 257 pages. 2005. (Subseries LNAI).

Vol. 3358: J. Cao, L.T. Yang, M. Guo, F. Lau (Eds.), Parallel and Distributed Processing and Applications. XXIV, 1058 pages. 2004.

Vol. 3357: H. Handschuh, M.A. Hasan (Eds.), Selected Areas in Cryptography. XI, 354 pages. 2004.

Vol. 3356: G. Das, V.P. Gulati (Eds.), Intelligent Information Technology. XII, 428 pages. 2004.

Vol. 3355: R. Murray-Smith, R. Shorten (Eds.), Switching and Learning in Feedback Systems. X, 343 pages. 2005.

Vol. 3353: J. Hromkovič, M. Nagl, B. Westfechtel (Eds.), Graph-Theoretic Concepts in Computer Science. XI, 404 pages. 2004.

Vol. 3352: C. Blundo, S. Cimato (Eds.), Security in Communication Networks. XI, 381 pages. 2005.

Vol. 3351: G. Persiano, R. Solis-Oba (Eds.), Approximation and Online Algorithms. VIII, 295 pages. 2005.

Vol. 3350: M. Hermenegildo, D. Cabeza (Eds.), Practical Aspects of Declarative Languages. VIII, 269 pages. 2005.

Vol. 3349: B.M. Chapman (Ed.), Shared Memory Parallel Programming with Open MP. X, 149 pages. 2005.

Vol. 3348: A. Canteaut, K. Viswanathan (Eds.), Progress in Cryptology - INDOCRYPT 2004. XIV, 431 pages. 2004.

Vol. 3347: R.K. Ghosh, H. Mohanty (Eds.), Distributed Computing and Internet Technology. XX, 472 pages. 2004.

Vol. 3346: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 249 pages. 2005. (Subseries LNAI).

Vol. 3345: Y. Cai (Ed.), Ambient Intelligence for Scientific Discovery. XII, 311 pages. 2005. (Subseries LNAI).

Vol. 3344: J. Malenfant, B.M. Østvold (Eds.), Object-Oriented Technology. ECOOP 2004 Workshop Reader. VIII, 215 pages. 2005.

Vol. 3343: C. Freksa, M. Knauff, B. Krieg-Brückner, B. Nebel, T. Barkowsky (Eds.), Spatial Cognition IV. Reasoning, Action, and Interaction. XIII, 519 pages. 2005. (Subseries LNAI).

Vol. 3342: E. Şahin, W.M. Spears (Eds.), Swarm Robotics. IX, 175 pages. 2005.

Vol. 3341: R. Fleischer, G. Trippen (Eds.), Algorithms and Computation. XVII, 935 pages. 2004.

Vol. 3340: C.S. Calude, E. Calude, M.J. Dinneen (Eds.), Developments in Language Theory. XI, 431 pages. 2004.

Vol. 3339: G.I. Webb, X. Yu (Eds.), AI 2004: Advances in Artificial Intelligence. XXII, 1272 pages. 2004. (Subseries LNAI).

Vol. 3338: S.Z. Li, J. Lai, T. Tan, G. Feng, Y. Wang (Eds.), Advances in Biometric Person Authentication. XVIII, 699 pages. 2004.

Vol. 3337: J.M. Barreiro, F. Martin-Sanchez, V. Maojo, F. Sanz (Eds.), Biological and Medical Data Analysis. XI, 508 pages. 2004.

Vol. 3336: D. Karagiannis, U. Reimer (Eds.), Practical Aspects of Knowledge Management. X, 523 pages. 2004. (Subseries LNAI).

Vol. 3335: M. Malek, M. Reitenspieß, J. Kaiser (Eds.), Service Availability. X, 213 pages. 2005.

Vol. 3334: Z. Chen, H. Chen, Q. Miao, Y. Fu, E. Fox, E.-p. Lim (Eds.), Digital Libraries: International Collaboration and Cross-Fertilization. XX, 690 pages. 2004.

Vol. 3333: K. Aizawa, Y. Nakamura, S. Satoh (Eds.), Advances in Multimedia Information Processing - PCM 2004, Part III. XXXV, 785 pages. 2004.

Vol. 3332: K. Aizawa, Y. Nakamura, S. Satoh (Eds.), Advances in Multimedia Information Processing - PCM 2004, Part II. XXXVI, 1051 pages. 2004.

Vol. 3331: K. Aizawa, Y. Nakamura, S. Satoh (Eds.), Advances in Multimedia Information Processing - PCM 2004, Part I. XXXVI, 667 pages. 2004.

Vol. 3330: J. Akiyama, E.T. Baskoro, M. Kano (Eds.), Combinatorial Geometry and Graph Theory. VIII, 227 pages. 2005.

Vol. 3329: P.J. Lee (Ed.), Advances in Cryptology - ASIACRYPT 2004. XVI, 546 pages. 2004.

Vol. 3328: K. Lodaya, M. Mahajan (Eds.), FSTTCS 2004: Foundations of Software Technology and Theoretical Computer Science. XVI, 532 pages. 2004.

Vol. 3327: Y. Shi, W. Xu, Z. Chen (Eds.), Data Mining and Knowledge Management. XIII, 263 pages. 2005. (Subseries LNAI).

Vol. 3326: A. Sen, N. Das, S.K. Das, B.P. Sinha (Eds.), Distributed Computing - IWDC 2004. XIX, 546 pages. 2004.

Vol. 3325: C.H. Lim, M. Yung (Eds.), Information Security Applications. XI, 472 pages. 2005.

Vol. 3323: G. Antoniou, H. Boley (Eds.), Rules and Rule Markup Languages for the Semantic Web. X, 215 pages. 2004.

Preface

The key challenge for future computer system is dealing with complexity. On one hand this involves internal system complexity which has increased exponentially over recent years. Here the main objectives are to maintain system reliability and to keep the design and maintenance effort manageable, while at the same time continuing to provide new functionality and increasing system performance. This has been the focus of so-called autonomous computing, which aims to bring self-configuration and repair to a wide range of computing systems.

On the other hand future computer systems are more and more becoming integrated into the fabric of everyday life and thus have to deal with the complexities of the real world. They will become smaller, more appropriate for their use, integrated into everyday objects, and often virtually or physically invisible to the users. They will also be deployed in a much higher quantity and penetrate many more application areas than traditional notions of computer systems. This requires computer systems to be adaptable within a much wider range of possible tasks, subjected to much harsher conditions.

To provide such features and functionality, computer devices will become tinier yet still increase in system complexity; they must consume less power, while still supporting advanced computation and communications, such that they are highly connected yet still operate as autonomous units. Pervasive and ubiquitous computing research addresses such issues by developing concepts and technology for interweaving computers into our everyday life. The principal approach is to enhance system functionality and adaptability by recognizing context and situations in the environment.

Organic computing deals with high system complexity by drawing analogies from complex biological systems, with the human-centered goal of self-organization. It addresses both internal system complexity of conventional systems and the complexity involved in pervasive environments dealing with the real world. Thus organic computing investigates the design and implementation of self-managing systems that are self-configuring, self-optimizing, self-healing, self-protecting, context aware, and anticipatory. It touches upon a number of exciting research topics including ultra-low power consumption, scalability and complexity of devices and systems, self-awareness, adaptive networking, and smart behavior of systems.

Many papers submitted to the Architecture of Computing Systems Conference (ARCS) address these aspects of adaptable, self-organizing systems. For computer system hardware, reconfigurable hardware allows us to optimize the system performance based on the application context, relieving software developers from detailed consideration of the inherently inflexible hardware constraints. Adaptive methods for managing resources and tasks enable (embedded) microprocessor systems to be both real-time aware but also very low in their power consumption. In software, middleware agents are able to cope with changes in application and environment, thus still providing a minimum of functionality even under difficult and changing conditions.

Adaptive ad hoc communication networks and context-aware pervasive systems and applications provide the functionality mostly visible to the end user of such systems. To achieve this extent of adaptivity a large variety of methods were used – many of them borrowed from nature. The papers in this book present a good profile of such novel methods and their application in the area of computing systems.

This year the ARCS conference selected 18 papers from a competitive field of 52 submissions from 12 countries. All papers accepted for presentation were peer reviewed and discussed in the first step in an online discussion among members of the international Program Committee. In the PC Meeting then the final decisions were made based on these reviews and the online discussions. Care was taken to avoid any conflict of interest by handing out papers and discussion papers only to PC members not involved in or related to the work.

We would like to take the opportunity to thank the numerous people who supported us in organizing the paper program and the conference: the Program Committee members for their efforts in reviewing many papers; Uwe Brinkschulte for supporting us by serving as the Workshops and Tutorials Chair; and the General and Program Chairs, Christian Müller-Schloer and Theo Ungerer, for sharing their experience with us and helping us to organize the paper program and the conference.

We extend our gratitude to several organizations that provided financial and organizational support for the ARCS conference. Volker Schanz from the ITG provided the legal framework and the ARCS Fachausschuss, the organizational body of the conference. Financial and organizational support came from the APS+PC group, which organized and funded a special session with several interesting invited talks. Donations also came from our benefactor, Siemens AG. We would also like to thank Christian Decker and Michael Biebl for their help during the electronic submission process, and the University for Health Sciences, Medical Information and Technology in Innsbruck, Austria for hosting the conference.

January 2005

Michael Beigl
Program Chair ARCS 2005

Paul Lukowicz
General Chair ARCS 2005

Executive Committee

Conference Chair:	Paul Lukowicz	University for Health Sciences, Medical Informatics and Technology, Austria
Program Chair:	Michael Beigl	University of Karlsruhe, Germany
Workshops and Tutorials Chair:	Uwe Brinkschulte	University of Karlsruhe, Germany

Program Committee

Shigeru Ando	University of Tokyo, Japan
Nader Bagherzadeh	University of California, Irvine, USA
Frank Bellosa	University of Erlangen, Germany
Uwe Brinkschulte	University of Karlsruhe, Germany
Roy Campbell	University of Illinois at Urbana-Champaign, USA
Alois Ferscha	University of Linz, Austria
Paul Havinga	University of Twente, Netherlands
Wolfgang Karl	University of Karlsruhe, Germany
Jürgen Kleinöder	Universität Erlangen-Nürnberg, Germany
Rudolf Kober	Siemens AG, München, Germany
Spyros Lalis	University of Thessaly, Greece
Erik Maehle	Universität zu Lübeck, Germany
Christian Müller-Schloer	University of Hannover, Germany
Joe Paradiso	MIT Media Laboratory, USA
Burghardt Schallenberger	Siemens AG, Germany
Hartmut Schmeck	University of Karlsruhe, Germany
Albrecht Schmidt	LMU Munich, Germany
Karsten Schwan	Georgia Institute of Technology, USA
Rainer G. Spallek	Technische Universität Dresden, Germany
Peter Steenkiste	Carnegie Mellon University, USA
Yoshito Tobe	Tokyo Denki University, Japan
Hide Tokuda	Keio University, Japan
Theo Ungerer	University of Augsburg, Germany
Klaus Waldschmidt	University of Frankfurt, Germany
Lars Wolf	University of Braunschweig, Germany
Martina Zitterbart	University of Karlsruhe, Germany

Organization

The Architecture of Computing Systems (ARCS) Conference was organized by the ITG (Informationstechnische Gesellschaft – Information Technology Society) and the Special Interest Group on Computer and Systems Architecture of the GI (Gesellschaft für Informatik – German Informatics Society), supported by OCG (Austrian Computer Society), OVE/GIT (Austrian Electrotechnical Association) and electrosuisse (ITG), and held in cooperation with ACM.

Sponsor

Siemens AG, Munich, Germany

Table of Contents

Adaptation, Power Consumption and Scheduling

Adaptation and Agents

Adaptation and Services

Application Adaptable Systems

Pervasive Computing and Communication

Energy Management for Embedded Multithreaded Processors with Integrated EDF Scheduling

Sascha Uhrig and Theo Ungerer

Institute of Computer Science, University of Augsburg,
86159 Augsburg, Germany
Tel.: +498215982353, Fax.: +498215982359
{uhrig,ungerer}@informatik.uni-augsburg.de

Abstract. This paper proposes a new hardware-based energy management technique for future embedded multithreaded processors with integrated Earliest Deadline First (EDF) real-time scheduling. Our energy management technique controls frequency reduction and dynamic voltage scaling depending on the deadlines, the Worst Case Execution Times (WCET), and the real execution times. Hard real-time capability can be guaranteed for aperiodic threads and for threads with deadlines shorter than their period. Our evaluations show that energy consumption can be reduced up to about $\frac{2}{3}$ of a comparable software-based algorithm.

Keywords: energy management, energy-aware program execution, real-time scheduling, multithreading, EDF scheduling

1 Introduction

The reduction of energy consumption is an important research field because of the rapidly growing number of battery-powered mobile and embedded devices. Hard real-time is often an essential requirement for such systems. This paper focuses on energy management in embedded processor cores in combination with real-time applications. The aim is to reduce the total energy consumption by optimizing power consumption without delaying the completion of the real-time threads.

In CMOS devices, the power consumption is proportional to the square of the supply voltage and linear to the frequency:

$$P_{cmos} = aC_L V_{DD}^2 f,$$

where a is the activity of the circuit, C_L is the output load capacity, V_{DD} the supply voltage, and f the frequency. Obviously, power consumption can be reduced dynamically by decrementing supply voltage and clock frequency. Unfortunately, supply voltage depends on clock frequency and, using lower frequency, the processor's performance is reduced too. Hence, in real-time systems, we have to control frequency in a way which does not harm the real-time behavior of the system.

M. Beigl and P. Lukowicz (Eds.): ARCS 2005, LNCS 3432, pp. 1–17, 2005.

We developed a multithreaded Java microcontroller – called Komodo microcontroller – with hardware-integrated real-time scheduling schemes [1, 2] for application in embedded real-time systems and ubiquitous devices. The Komodo microcontroller is able to perform a thread switch without any overhead. Thus, instructions of active threads are executed in an overlapped fashion inside the core pipeline; the EDF scheduler hardware ensures that the thread with the earliest deadline is the thread with the highest priority. Due to hardware multithreading, instructions of other threads are executed within latency cycles of the thread with the highest priority without interfering with its execution (latency bridging).

We investigate mechanisms to minimize energy consumption using hardware-based energy management techniques that are made possible by a multithreaded processor core with integrated EDF scheduling. In particular, we show that energy saving techniques like frequency reduction and voltage scaling can be controlled more efficient by the integrated EDF energy management than using conventional operating system methods. Our hardware-integrated energy management algorithm chooses automatically in each processor cycle the frequency and voltage level that is currently required to perform a real-time application without any miss of deadline.

The next two sections show state-of-the-art energy saving mechanisms and related work. Section 4 presents the extensions for hardware-based energy management within the processor-integrated EDF scheduler and in section 5 we evaluate our approach. Section 6 concludes the paper.

2 State-of-the-Art Energy Saving Mechanisms

Commercial processors use a number of techniques for saving energy like pipeline gating, several suspend or sleep modes, and reduction of frequency and supply voltage. Intel's XScale [3], Transmeta's Crusoe [4] and the MSP430 [5] from Texas Instruments work with software-controlled techniques of frequency reduction and voltage scaling.

We describe shortly the energy saving features of the XScale and the Crusoe processors, because we use their electrical properties (voltages and frequency rates) for simulating our hardware-based energy management. Both processors are able to run at several frequencies using different supply voltages. A change of frequency requires among other tasks to complete all outstanding memory accesses, to set the external SDRAM to self-refresh mode, and to disable the interrupt controller. Most tasks are done automatically, but, nevertheless, they need time for execution. The whole process of changing frequency requires up to 500μs in the case of the XScale. Using the Crusoe processor, the time required for a supply voltage change depends on the distance of the two voltage levels. The maximum value is about 896μs in the default configuration.

Pipeline Gating [6] is a technique for selectively disconnecting parts of the processor, especially pipeline stages. So the energy consumption can be reduced by uncoupling unnecessary parts of the pipeline without concerning any other component. In contrast, frequency and voltage scaling affect the whole circuit.

3 Related Work on Real-Time Energy Management

Different directions of research targeting real-time applications are present: energy management controlled by the application, the operating system, or by the hardware itself. Application-based power management requires special power control sequences within the application's program code. Shin et al. [7] present a technique for automatic insertion of power controlling code based on a WCET analysis before runtime. The suggested mechanism is feasible for hard real-time systems.

In contrast to application-based techniques, other approaches focus on frequency and voltage reduction controlled by the operating system, especially by its thread scheduler. Pillai et al. [8] present several energy-aware scheduling schemes similar to the EDF scheduling scheme for low-power embedded real-time operating systems. Jejurikar et al. [9] focus on the problem of task synchronization in combination with energy-aware task scheduling. Pouwelse et al. [10, 11] describe a hybrid approach, which is based on an extended Linux OS with a so-called *energy priority scheduling*. The parameters for the scheduler are given by the application.

A theoretical approach for an energy saving technique using EDF scheduling is presented by Krishna et al. [12, 13]. Their energy management is based on an offline thread schedule, the online schedule, an offline and an online function, which describe the amount of work to do. Aydin et al. [14] additionally use a speculative speed adjustment for periodic real-time tasks.

All presented techniques are based on a single-threaded processor core and a software-based energy management. Energy management investigations concerning multithreaded processors pertain simultaneous multithreading and are made by [15, 16]. Energy management of a multithreaded single-issue processor with integrated Guaranteed Percentage (GP) hardware real-time scheduling was evaluated by ourselves [17, 18].

All existing processors and research approaches (except our GP energy management) suffer from the inefficiency of software control: Calculating the optimal frequency and the supply voltage by software requires a software overhead. Additionally, most control techniques assume a continuous frequency control which is not realistic. In real processors, frequency is selected by binary clock multipliers and dividers, i.e. only discrete frequency levels are possible. A more efficient solution is a hardware-based energy management, i.e. the processor core decides to run at the optimal frequency and voltage level by itself and is able to readjust frequency and voltage during thread execution.

Another drawback of existing energy management techniques in combination with real-time scheduling is the often used assumption, that the deadline of each thread has to be equal to its period. Krishna et al. and Aydin et al. additionally require an offline thread "execution" for determining the *amount of work* function and the offline schedule itself for the energy management.

4 Hardware-Based Energy Management Mechanism

4.1 Thread Model

For our energy management technique we permit arbitrary activation of threads with the constraints that all threads are independent and that a thread will only be restarted after its completion, i.e. at most one instance of each thread is active at a time. In the case of periodic threads, we do not make the assumption that their deadlines are equal to their periods.

For the realization of our proposed energy management technique, several characteristics of the execution of a thread are necessary. Fig. 1 illustrates the required values which are measured in execution cycles. The figure is divided into two scheduling areas: the upper area describes the *regular thread scheduling* which is similar to Krishna's offline scheduling, with the difference that it is generated online by the knowledge of the WCETs and the deadlines of the already completed and all actually active threads. The lower area mirrors the scheduling depending on the real runtime behavior of the threads, i.e. the *runtime scheduling.* In addition to these two schedulers a third scheduler, not shown in the figure, called *execution scheduler* is present. It is responsible for the selection of the thread executed within the multithreaded processor pipeline in the current clock cycle. Because of the latency bridging, the scheduling decision temporarily alternates between different threads.

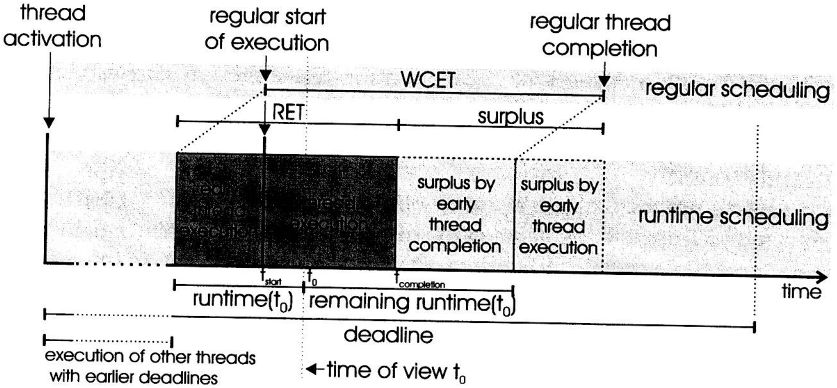

Fig. 1. Characteristics during the execution of a thread.

The *deadline* and the *WCET* are given by the application and stored as constants within the energy management unit. The *surplus* are the remaining cycles from thread completion to the regular (planned) completion of the thread assuming that all previous threads have exhausted their *WCET* too. The *runtime(t_0)* represents the amount of execution cycles the current thread has executed up to time t_0. In general, due to the multithreaded execution and the *surplus* of the previous thread, an *early thread execution* takes place and thus, the *runtime(t_{start})*

is greater than zero at the *regular start of execution*. At thread completion, the *runtime($t_{completion}$)* is equal to the real execution time (*RET*). The *remaining runtime(t_0)* is the number of cycles the thread will run from time t_0 (assuming its WCET), i.e., the difference between the *WCET* and the *runtime(t_0)*. The surplus is the sum of the surplus by early thread completion and the surplus by early thread execution (surplus of the previous thread).

4.2 Methodology

The idea behind the hardware-based energy management mechanism is that the active threads rarely need the time calculated as WCET for the actual execution as it is reported in [19]. Thus frequency can be reduced such that all threads terminate as late as possible but not later than the time predicted by the schedulability analysis (depending on the WCETs). As a consequence, the supply voltage can be adapted to a level corresponding to the throttled frequency, which may lead to a tremendous energy saving. Because of the direct relationship between the selected clock frequency and the required supply voltage, determining the optimal clock frequency is the real challenge.

Using a software-based solution, frequency and voltage selection is only possible at the time of a thread suspend or activation (intertask DVS) or at dedicated points during thread execution (intratask DVS). In contrast to a software-based version, our hardware-based energy management is able to observe the progression (in execution cycles) of all active threads continuously. Thus, clock frequency and supply voltage can be adapted dynamically during the thread's execution to approximate the optimal execution speed.

At the time of a thread suspend the presented energy saving mechanism registers the number of execution cycles remaining to the regular thread suspend, i.e. the surplus. Due to the surplus of the just suspended thread the execution of the thread directly following can be slowed down. The optimal frequency $f_{reduced}$ can be calculated by the formula

$$f_{reduced} = \frac{WCET}{surplus + WCET} * f_{max},$$

where f_{max} is the maximum frequency of the processor, $WCET$ is the WCET of the new thread, and *surplus* is the surplus of the just suspended thread. If the processor is working at the calculated optimal frequency $f_{reduced}$ and the new thread requires its complete WCET, its execution completes exactly at the time planned by the schedulability analysis. If the new thread does not need its WCET for execution it offers a surplus to the following thread. Usually only fixed frequency levels are provided by the processor. So the optimal frequency cannot be selected and a frequency higher than the optimal one has to be chosen. As result, the really required energy is higher than the theoretical necessary energy.

4.3 Implementation

To realize the EDF energy management the following set of five hardware registers are required for each hardware thread slot: