Lecture Notes in Computer Science 1474
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Frank Mueller Azer Bestavros (Eds.)

Languages, Compilers, and Tools for Embedded Systems

ACM SIGPLAN Workshop LCTES'98
Montreal, Canada, June 19-20, 1998
Proceedings

Springer

Series Editors

Gerhard Goos, Karlsruhe University, Germany
Juris Hartmanis, Cornell University, NY, USA
Jan van Leeuwen, Utrecht University, The Netherlands

Volume Editors

Frank Mueller
Humboldt Universität Berlin, Institut für Informatik
Unter den Linden 6, D-10099 Berlin, Germany
E-mail: mueller@informatik.hu-berlin.de

Azer Bestavros
Boston University, Department of Computer Science
111 Cummington Street, Boston, MA 02215, USA
E-mail: best@bu.edu

Cataloging-in-Publication data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Languages, compilers, and tools for embedded systems : proceedings / ACM SIGPLAN Workshop LCTES '98, Montreal, Canada, June 19 - 20, 1998. Frank Mueller ; Azer Bestavros (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Singapore ; Tokyo : Springer, 1998
(Lecture notes in computer science ; Vol. 1474)
ISBN 3-540-65075-X

CR Subject Classification (1991): D.4.7, C.3, D.3, D.2

ISSN 0302-9743
ISBN 3-540-65075-X Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author
SPIN 10638685 06/3142 – 5 4 3 2 1 0 Printed on acid-free paper

Preface

From 19th to the 20th of June the ACM SIGPLAN 1998 Workshop on Languages, Compilers, and Tools for Embedded Systems (LCTES'98) was held in Montreal, Canada. LCTES'98 is a reincarnation of the Workshop on Languages, Compilers, and Tools for Real-Time Systems, which was held three times in previous years. The change of focus from real-time systems to embedded systems came as a response to the growing importance of embedded systems. Embedded systems research is rapidly growing since the number of embedded processors (and thereby installed systems) already exceeds the number of general-purpose processors (personal computers and workstations) today. The shift of focus from real-time systems to embedded systems also reflects the more general nature of embedded systems, which encompasses a multitude of aspects in addition to real-time systems. These aspects include embedded languages, memory restrictions, special-purpose processors and peripheral devices, and power consumption, just to name a few.

In the past, custom kernels, non-standard languages, vendor-specific device interfaces and custom hardware were often employed for building embedded systems. Today, there is an emerging trend to exploit off-the-shelf hardware and to enhance standard software to meet embedded requirements, ranging from real-time extensions of common programming languages and operating systems to appropriate tools for embedded programmers.

For both real-time and embedded systems, electrical engineers and computer scientists are active in addressing many similar problems, but with different backgrounds and approaches. LCTES is intended to expose researchers and developers from the areas of embedded systems and real-time systems to relevant work and interesting problems in the other area and provide a forum where they can interact. The range of topics covered by LCTES reaches from experimental work over commercial systems to applied theory.

LCTES'98 featured two invited talks and presentations of seven long and twelve short refereed papers. The papers were selected out of 54 submissions. Twelve referees prepared written multiple reviews for each submission. The recommendations of the referees determined an initial set of papers selected for acceptance. The members of the program committee were then given the option to resolve differences in opinion for the papers they refereed based on the written reviews. After the resolution process, the 19 papers included in the workshop proceedings were selected for presentation and publication.

Summary of the Papers

The accurate estimation of Worst Case Execution Time (WCET) plays an important role in the design of correct and efficient embedded software. WCET is typically overestimated because of insufficient compiler path analysis and timing analysis. Existing methods require annotations or static path analysis, which establish upper bounds on loop iterations and identify some infeasible paths. Static timing analysis are too simple; fixed penalties are usually charged by cache misses and pipeline hazards. In their paper entitled "Integrating path and timing analysis using instruction-level simulation techniques", Thomas Lundqvist and Per Stenström of Chalmers University of Technology, Sweden, propose a new method to estimate WCET on contemporary processors with complex pipelines and multilevel hierarchies. The new method integrates path analysis and timing analysis based on an instruction level simulator, which handles unknown data values. The simulator automatically performs path analysis while simulating the program. For some classes of programs, this approach has specific advantages compared to other approaches as it naturally integrates path and timing analysis. The WCET predictions can be very precise.

Another important functionality of compilers targeted at embedded software systems is the prediction of cache performance. In their paper entitled "On predicting data cache behavior for real-time systems", Christian Ferdinand and Reinhard Wilhelm of the University of the Saarland, Germany, present analysis techniques for determining memory locations within a cache that do not get replaced during the program's execution. This gives an upper bound on the number of cache misses. The technique used is similar to dataflow analysis to determine the fixed point state of the cache. This state is analyzed to provide the memory blocks that would never get replaced. To do this, the authors present representations for the components of the cache and transfer functions that map cache states based on memory references.

In their paper entitled "Automatic accurate time-bound analysis for high-level languages", Yanhong A. Liu and Gustavo Gomez of Indiana University, USA, present an approach to automate timing analysis and tuning of embedded programs. This involves initially transforming the program functions to add timing functions at the source level. These are optimized symbolically with respect to specific values of the input parameters. There are some further transformations proposed to improve accuracy, in the presence of conditionals. An implementation has been carried out for a subset of the Scheme language and a comparison of calculated and measured execution times for some programs is presented.

The construction of execution environments, in which real-time and non-real-time application components coexist is important. In their paper "Extending RT-Linux to support flexible hard real-time systems with optional components", A. Terrasa, A. Espinosa and A. Garcia-Fornes, of the Universidad Politecnica de Valencia, Spain, present their RT-Linux solution to this problem, which is based on the well-known partitioning of a task into an initial part, an optional part, and a final part.

Another aspect of cache behavior that impacts real-time embedded software systems is cache reloading. In their paper entitled "Limited preemptible scheduling to embrace cache memory in real-time systems", Sheayun Lee et. al. of Seoul National University, Korea, propose a technique that allows preemption only at certain points in

tasks scheduled using fixed priority preemptive scheduling. By analyzing cache behavior, these preemption points are chosen to minimize cache reload times when resuming the preempted process. In order to determine the useful cache blocks in a program, all tasks are considered in isolation. However, due to periodic tasks or due to shared code, there may be cache blocks which are useful for other tasks or other incarnations of the same task.

Distributed systems are entering the stage of real-time systems. In "A uniform reliable multicast protocol with guaranteed response times", Laurent George and Pascale Minet from INRIA, France, extend a well-known broadcast algorithm to a multicast protocol in a distributed environment. Their goal includes tolerating failures of nodes and omissions of messages while still guaranteeing end-to-end response times. The work assumes a variant of multiprocessor earliest-deadline-first scheduling under a number of restrictions.

An important functionality for embedded software designers is the ability to fine-tune the timing behavior of their programs. In their paper entitled "A tool to assist in fine-tuning and debugging embedded real-time systems", Gaurav Arora and David B. Stewart of the University of Maryland, USA, present a tool that can be used late in the development cycle of embedded software to adjust the system to meet its timing requirements. The tool takes data sampled from actual executions (via some monitor tool) and correlates it with the specification and/or the modeled behavior. The tool enables adjustments in the code/timing parameters to be specified and effects thereof to be predicted.

The use of distributed architectures for embedded systems has been hampered by the unavailability of appropriate programming environments for such architectures. In their paper entitled "Debugging distributed implementations of modal process systems", Ken Hines and Gaetano Borriello of the University of Washington in Seattle, USA, discuss some of the debugging techniques enabled by the modal process model of Chou and Borriello. This modal process model enables the use of distributed architectures for embedded systems because it alleviates the problem of additional costs in their design, implementation, and maintenance.

The advent of the Java Virtual Machine promises many benefits to embedded software development. In their paper entitled "Using Inferno to run Java on small devices", C. F. Yurkoski, L. R. Rau, and B. K. Ellis, of Bell Labs, USA, present an implementation of Java on the Inferno operating system. Inferno is similar to Java in that it uses a virtual machine for its basic program execution engine. However, the Inferno VM is superior to the Java VM from a performance viewpoint – and thus more suitable for embedded systems. In this paper, the authors discuss a variety of approaches that were considered for supporting Java on Dis (the Inferno VM). The approach finally taken was to translate Java bytecode to Dis instructions.

The work of Michael Weiss et. al. of the Open Group Research Institute, USA, takes an alternative approach to Java bytecode compilation, ensuring compatibility and the possibility of dynamic class loading. In their paper entitled "TurboJ, a Java bytecode-to-native compiler", the authors present a bytecode to native code compiler for Java (JVM code). Their approach is to coexist with the JVM, allowing JVM bytecode and native code to be executed. They translate class files into C files and new versions of

class files, thereby performing a variety of optimizations. The C files are compiled and linked into a shared library to be used by the JVM at run time.

Another approach to minimizing the timing unpredictability caused by cache systems is presented in a paper entitled "Cache-sensitive pre-runtime scheduling", by Daniel Kästner and Stephan Thesing of the University of the Saarland, Germany. The paper presents an interesting technique that attempts to avoid switching between tasks that have incompatible cache contents. The paper presents an integration of cache-behavior prediction into a pre-runtime scheduling schema. Tasks are split into segments. The static cache-analysis predicts (interprets) the cache behavior of every segment which is used in the defined scheduling algorithm.

Recently, dynamic reactive systems have been studied where tasks are only triggered by external events. The work titled "Priority assignment for embedded reactive real-time systems" by Felice Balarin from Cadence Berkeley Labs, USA, presents a theoretical framework to assign priorities to precedence-related tasks. It extends an existing optimal priority assignment algorithm to limit the number of possible priority orderings examined for an optimal solution – the limit is reduced from an exponential function in the number of tasks, to a quadratic.

Scheduling and schedulability analysis are key issues for real-time, embedded computing systems, especially when the deadlines imposed on processes are hard. In their paper entitled "Mapping an embedded hard real-time systems SDL specification to an analyzable task network – A case study", Thomas Kolloch and Georg Färber of TU München, Germany, present a method to transform an SDL specification of a system into a precedence graph that can be analyzed for real-time schedulability. It presents a mine-control system as a case study to illustrate the process.

The deadlines imposed on embedded system processes are often specified to guarantee specific end-to-end delays. In their paper entitled "End-to-end optimization in heterogeneous distributed real-time systems", Seonho Choi of Bowie State University, USA, describes a method to merge a bunch of scheduling tests for local systems (CPUs and networks) into a big end-to-end schedulability test in such a way that a linear programming optimization algorithm can be used to find some optimal local/intermediate deadlines.

Compilers and tools developed for engineering embedded systems must rely on a canonical description of the underlying machinery to be able to provide meaningful analysis (*e.g.*, timing and schedulability analysis). Toward that goal, the paper entitled "Machine descriptions to build tools for embedded systems", by Norman Ramsey and Jack W. Davidson of the University of Virginia Charlottesville, USA, presents techniques for specifying descriptions of machines that can be used by a variety of tools. These tools include compilers, simulators, assemblers, linkers, debuggers, and profilers. The authors describe in detail the notation used for representing machine instructions, registers, memory, etc.

Compiler optimizations for embedded systems often aim at goals that differ from those for general-purpose architectures. In "Non-local instruction scheduling with limited code growth" by Keith D. Cooper and Philip J. Schielke from Rice University, USA, two techniques for global instruction scheduling without increasing the code size are presented. One technique is novel, the other is an extension of earlier work (with a

counterexample for the previous method). The methods are applied (via simulation) to VLIW architectures to show that 6–7% execution time can be saved and the code size can be reduced by 11%.

Embedded systems are typically subjected to stringent constraints on memory size. Thus, techniques and tools that assist in the partitioning and overlaying of data structures to minimize memory needs are pivotal for such limited-memory systems. In their paper entitled "An efficient data partitioning method for limited memory embedded systems", Anantharaman Sundaram and Santosh Pande of the University of Cincinnati, USA, propose a framework for efficient data partitioning of arrays on limited memory embedded systems based on the concept of footprint, which is associated with array references in programs. The idea behind footprinting is to partition arrays into local and remote sections of memory. The main goal is to place the data that is referenced most frequently in the local memory to minimize the number of remote references.

In their paper entitled "A design environment for counterflow pipeline synthesis", Bruce R. Childers and Jack W. Davidson of the University of Virginia, USA, outline a hardware/software codesign approach for special purpose embedded architectures. First, the kernel (*i.e.*, the part associated with high execution frequency) of an application is identified, then a search in the design space of the pipeline is performed to determine the number/functionality of stages. The intent is to optimize the timing of the pipeline for this piece of code given timings for each functional unit. Heuristics are used to guide the search by taking data dependencies and timings into account. Results show a 1.3 to 2 times speedup over comparable general-purpose designs.

Embedded systems use special-purpose operating systems, *e.g.*, micro kernels with limited, application-specific functionality and fast response to events. The paper titled "Efficient user-level I/O in the ARX real-time operating system" by Yangmin Seo, Jungkeun Park, and Seongsoo Hong from Seoul National University, Korea, discusses a variation on the implementation of UNIX signals in the ARX real-time operating system. The focus is on propagating kernel-level interrupts to user space efficiently.

Workshop History and Future

LCTES was preceded by three workshops on Languages, Compilers, and Tools for Real-Time Systems (LCT-RTS). The proceedings of the previous workshops are available online. In addition, the final program with links to the authors of the current workshop (LCTES'98) is also online. Finally, the next LCTES workshop is currently being planned for 1999, and online information will be made available during the Fall of 1998.

http://www.cs.indiana.edu/~liu/lctes99/
LCTES 1999 Call for Papers (Atlanta, Georgia, USA)

http://www.informatik.hu-berlin.de/~mueller/lctes98
LCTES 1998 Program (Montreal, Quebec, Canada)

http://www.cs.pitt.edu/~gupta/lct-rts97.html
LCT-RTS 1997 Online Proceedings (Las Vegas, Nevada, USA)

http://www.cs.umd.edu/projects/TimeWare/sigplan95
LCT-RTS 1995 Online Proceedings (La Jolla, California, USA)

http://www.cs.umd.edu/users/pugh/sigplan_realtime_workshop/lct-rts94
LCT-RTS 1994 Online Proceedings (Orlando, Florida, USA)

LCTES Program Committee

- Gul Agha, University of Illinois, USA
- Azer Bestavros, Boston University, USA
- Rajiv Gupta, University of Pittsburgh, USA
- John Gough, Queensland University of Technology, Australia
- Wolfgang Halang, University of Hagen, Germany
- Annie Liu, Indiana University, USA
- Thomas Marlowe, Seton Hall University, USA
- Frank Mueller, Humboldt University Berlin, Germany
- Manas Saksena, Concordia University, Canada
- Andy Wellings, University of York, UK
- David Whalley, Florida State University, USA
- Reinhard Wilhelm, University of the Saarland, Germany

Acknowledgments

The workshop chairs would like to acknowledge the following people:

- the invited speakers, Bran Selic and Marc Campbell, for their voluntary contributions to the workshop;
- Manas Saksena for his local support;
- Richard Gerber and Manas Saksena for their support of the workshop program;
- Alfred Hofmann for his work as a contact person to Springer LNCS;
- Thomas Ball for his programs to facilitate the review process;
- Jack Davidson and the ACM crew, Donna Baglio, Carole Mann, Judy Osteller, and David Mann, for their organizational support;
- the ACM SIGPLAN executive committee for their endorsement of LCTES;
- and last but not least the eager members of the program committee and the anonymous external reviewers.

The Co-Chairs,

June 1998 Frank Mueller and Azer Bestavros

Table of Contents

Refereed Papers