High-Performance and Time-Predictable Embedded Computing

RIVER PUBLISHERS SERIES IN INFORMATION SCIENCE AND TECHNOLOGY

Indexing: All books published in this series are submitted to the Web of Science Book Citation Index (BkCI), to CrossRef and to Google Scholar.

The "River Publishers Series in Information Science and Technology" covers research which ushers the 21st Century into an Internet and multimedia era. Multimedia means the theory and application of filtering, coding, estimating, analyzing, detecting and recognizing, synthesizing, classifying, recording, and reproducing signals by digital and/or analog devices or techniques, while the scope of "signal" includes audio, video, speech, image, musical, multimedia, data/content, geophysical, sonar/radar, bio/medical, sensation, etc. Networking suggests transportation of such multimedia contents among nodes in communication and/or computer networks, to facilitate the ultimate Internet.

Theory, technologies, protocols and standards, applications/services, practice and implementation of wired/wireless networking are all within the scope of this series. Based on network and communication science, we further extend the scope for 21st Century life through the knowledge in robotics, machine learning, embedded systems, cognitive science, pattern recognition, quantum/biological/molecular computation and information processing, biology, ecology, social science and economics, user behaviors and interface, and applications to health and society advance.

Books published in the series include research monographs, edited volumes, handbooks and textbooks. The books provide professionals, researchers, educators, and advanced students in the field with an invaluable insight into the latest research and developments.

Topics covered in the series include, but are by no means restricted to the following:

- Communication/Computer Networking Technologies and Applications
- Queuing Theory
- Optimization
- Operation Research
- Stochastic Processes
- Information Theory
- Multimedia/Speech/Video Processing
- Computation and Information Processing
- Machine Intelligence
- Cognitive Science and Brian Science
- Embedded Systems
- Computer Architectures
- Reconfigurable Computing
- Cyber Security

For a list of other books in this series, visit www.riverpublishers.com

High-Performance and Time-Predictable Embedded Computing

Editors

Luís Miguel Pinho

CISTER Research Centre, Polytechnic Institute of Porto, Portugal

Eduardo Quiñones

Barcelona Supercomputing Center, Spain

Marko Bertogna

University of Modena and Reggio Emilia, Italy

Andrea Marongiu

Swiss Federal Institute of Technology Zurich, Switzerland

Vincent Nélis

CISTER Research Centre, Polytechnic Institute of Porto, Portugal

Paolo Gai

Evidence Srl, Italy

Juan Sancho

ATOS, Spain

River Publishers

Published, sold and distributed by:
River Publishers
Alsbjergvej 10
9260 Gistrup
Denmark

River Publishers
Lange Geer 44
2611 PW Delft
The Netherlands

Tel.: +45369953197
www.riverpublishers.com

ISBN: 978-87-93609-69-3 (Hardback)
978-87-93609-62-4 (Ebook)

Contents

Preface

Nowadays, the prevalence of electronic and computing systems in our lives is so ubiquitous that it would not be far-fetched to state that we live in a cyber-physical world dominated by computer systems. Examples include pacemakers implanted within the human body to regulate and monitor heartbeats, cars and airplanes transporting us, smart grids, and traffic management.

All these systems demand more and more computational performance to process large amounts of data from multiple data sources, and some of them with guaranteed processing response times; in other words, systems required to deliver their results within pre-defined (and sometimes extremely short) time bounds. This timing aspect is vital for systems like planes, cars, business monitoring, e-trading, etc. Examples can be found in intelligent transportation systems for fuel consumption reduction in cities or railways, or autonomous driving of vehicles. All these systems require processing and actuation based on large amounts of data coming from real-time sensor information.

As a result, the computer electronic devices which these systems depend on are constantly required to become more and more powerful and reliable, while remaining affordable. In order to cope with such performance requirements, chip designers have recently started producing chips containing multiple processing units, the so-called multi-core processors, effectively integrating multiple computers within a single chip, and more recently the many-core processors, with dozens or hundreds of cores, interconnected with complex networks on chip. This radical shift in the chip design paved the way for parallel computing: rather than processing the data sequentially, the cooperation of multiple processing elements within the same chip allows systems to be executed concurrently, in parallel.

Unfortunately, the parallelization of the computing activities brought up many challenges, because it affects the timing behavior of the systems as well as the entire way people think and design computers: from the design of the hardware architecture, through the operating system up to the conceptualization of the end-user application. Therefore, although many-core processors are promising candidates to improve the responsiveness of these systems,

the interactions that the different computing elements may have within the chip can seriously affect the performance opportunities brought by parallel execution. Moreover, providing timing guarantees becomes harder, because the timing behavior of the system running within a many-core processor depends on interactions that are most of the time not known by the system designer. This makes system analysts struggle in trying to provide timing guarantees for such platforms. Finally, most of the optimization mechanisms buried deep inside the chip are geared only to increase performance and execution speed rather than providing predictable time behavior.

These challenges need to be addressed by introducing predictability in the vertical stack from high-level programming models to operating systems, together with new offline analysis techniques. This book covers the main techniques to enable performance and predictability of embedded applications. The book starts with an overview of some of the current many-core embedded platforms, and then addresses how to support predictability and performance in different aspects of computation: a predictable parallel programming model, the mapping and scheduling of real-time parallel computation, the timing analysis of parallel code, as well as the techniques to support predictability in parallel runtimes and operating systems.

The work reflected in this book was done in the scope of the European project P-SOCRATES, funded under the FP7 framework program of the European Commission. The project website (www.p-socrates.eu), provides further detailed information on the techniques presented here. Moreover, a reference implementation of the methodologies and tools was released as the UpScale Software Development Kit (http://www.upscale-sdk.com).

Luís Miguel Pinho
Eduardo Quiñones
Marko Bertogna
Andrea Marongiu
Vincent Nélis
Paolo Gai
Juan Sancho

February 2018

List of Contributors

Alessandra Melani, *University of Modena and Reggio Emilia, Italy*

Andrea Marongiu, *Swiss Federal Institute of Technology in Zürich (ETHZ), Switzerland; and University of Bologna, Italy*

Bruno Morelli, *Evidence SRL, Italy*

Claudio Scordino, *Evidence SRL, Italy*

Eduardo Quiñones, *Barcelona Supercomputing Center (BSC), Spain*

Errico Guidieri, *Evidence SRL, Italy*

Giuseppe Tagliavini, *University of Bologna, Italy*

Juan Sancho, *ATOS, Spain*

Luís Miguel Pinho, *CISTER Research Centre, Polytechnic Institute of Porto, Portugal*

Maria A. Serrano, *Barcelona Supercomputing Center (BSC), Spain*

Marko Bertogna, *University of Modena and Reggio Emilia, Italy*

Paolo Burgio, *University of Modena and Reggio Emilia, Italy*

Paolo Gai, *Evidence SRL, Italy*

Patrick Meumeu Yomsi, *CISTER Research Centre, Polytechnic Institute of Porto, Portugal*

Sara Royuela, *Barcelona Supercomputing Center (BSC), Spain*

Vincent Nélis, *CISTER Research Centre, Polytechnic Institute of Porto, Portugal*

List of Figures

List of Tables

List of Abbreviations

ADEOS	Adaptive Domain Environment for Operating Systems, a patch used by RTAI and Xenomai
ALU	Arithmetic logic unit
API	Application Programming Interface
APIC	Programmable timer on x86 machines
AUTOSAR	International consortium that defines automotive API
BCET	Best-Case Execution Time
BFS	Breadth First Scheduling
BOTS	Barcelona OpenMP task suite
BSD	Berkeley software license
CBS	Constant Bandwidth Server, a scheduling algorithm
CFG	Control Flow Graph
CMP	chip multi-processor
COTS	Commercial Off-The-Shelf
CPU	central processing unit
CUDA	Compute Unified Device Architecture
DAG	Direct Acyclic Graph
DDR	double data rate
DMA	direct memory access (engine)
DRAM	dynamic random-access memory
DSP	digital signal processor
DSU	debug system unit
EDF	Earliest Deadline First Scheduler, a scheduling algorithm
ELF	binary format that contain executables
eTDG	Extended Task Dependency Graph
EVT	Extreme Value Theory
FIFO	First-In First-Out
FLOPS	floating-point operations per second
FPGA	Field-Programmable Gate Array
GPGPU	General Purpose Graphics Processing Unit
GPL	General Public License

GPOS	General purpose Operating System
GPU	Graphics Processing Unit
GRUB	Greedy Reclamation of Unused Bandwidth, a scheduling algorithm
HAL	Hardware Abstraction Layer
HMI	Human Machine Interface
HPC	High Performance Computing
HRT	High Resolution Timers
ID	Identifier
IEC	International Electrotechnical Commission
IG	Interference Generator
IID	Independent and Identically Distributed
ILP	instruction-level parallelism
IRQ	Hardware Interrupt
ISA	Instruction Set Architecture
LHC	Large Hadron Collider
LIFO	Last-In First-Out
LL-RTE	Low-level runtime environment
LRU	Least recently used
MBPTA	Measurement-Based Probabilistic Timing Analysis
MCU	Microcontroller Unit
MEET	Maximum Extrinsic Execution Time
MIET	Maximum Intrinsic Execution Time
MPPA	Multi Purpose Processor Array
NOC	network-on-chip
NOP	No Operation
NUCA	non-uniform cache architecture
OpenCL	Open Computing Language
OpenMP	Open multi processing (programming model)
OpenMP DAG	OpenMP Direct Acyclic Graph
OS	Operating system
PCFG	Parallel Control Flow Graph
PCIe	peripheral component interconnect express
PE	processing element
PLRU	Pseudo-LRU
POSIX	Portable Operating System Interface for UNIX
PPC	PowerPC
P-SOCRATES	Parallel Software Framework for Time-Critical Many-core Systems

pWCET	Probabilistic Worst-Case Execution Time
RCU	Linux Read-Copy-Update technique
RM	Resource manager
RT	Real time
RT task	Real-Time task
RTE	Runtime environment
RTOS	Real time operative system
SDK	Software Development Kit
SMP	Symmetric Multi Processor
SMT	simultaneous multi-threading
SoC	System-on-Chip
SOM	system-on-module
SP	Stack pointer
TBB	Thread Building Blocks
TC	Task context
TDG	Task Dependency Graph
TDMA	Time Division Multiple Access
TLB	translation lookaside buffer
TLP	Thread-Level Parallelism
TSC	Task Scheduling Constraint
TSP	Task Scheduling Point
VLIW	Very Large Instruction Word
VPU	vector processing unit
WCET	Worst Case Execution Time
WFS	Work First Scheduling