Parallel Computer Organization and Design

Teaching fundamental design concepts and the challenges of emerging technology, this textbook prepares students for a career designing the computer systems of the future. In-depth coverage of complexity, power, reliability, and performance, coupled with treatment of parallelism at all levels, including ILP and TLP, provides the state-of-the-art training that students need. The whole gamut of parallel architecture design options is explained, from core microarchitecture to chip multiprocessors to large-scale multiprocessor systems. All the chapters are self-contained, yet concise enough that the material can be taught in a couple of semesters, making it perfect for use in senior undergraduate and graduate computer architecture courses. The book is also teeming with practical examples to aid the learning process, showing concrete applications of definitions. With simple models and codes used throughout, all material is accessible to a broad range of computer engineering/science students with only a basic knowledge of hardware and software.

Michel Dubois is a Professor in the Ming Hsieh Department of Electrical Engineering at the University of Southern California (USC) and part of the Computer Engineering Directorate. Before joining USC in 1984, he was a research engineer at the Central Research Laboratory of Thomson-CSF in Orsay, France. He has published more than 150 technical papers on computer architecture and has edited two books. He is a Fellow of the IEEE and of the ACM.

Murali Annavaram is an Associate Professor and Robert G. and Mary G. Lane Early Career Chair in the Ming Hsieh Department of Electrical Engineering at USC and part of the Computer Engineering Directorate, where he has developed and taught advanced computer architecture courses. Prior to USC, he spent 6 years at Intel researching various aspects of future CMP designs.

Per Stenström is a Professor of Computer Engineering at Chalmers University of Technology, Sweden. He has published two textbooks and over 100 technical papers. He has been a visiting scientist at Carnegie-Mellon, Stanford, and USC, and also was engaged in research at Sun Microsystems on its chip multi-threading technology. He is a Fellow of the IEEE and of the ACM, and is a member of the Royal Swedish Academy of Engineering Sciences and the Academia Europaea.

"Parallel computers and multicore architectures are rapidly gaining importance because the performance of a single core is not improving at the same historical level. Professors Dubois, Annavaram, and Stenstrom have created an easily readable book on the intricacies of parallel architecture design that academicians and practitioners alike will find extremely useful."

Shubu Mukherjee, Cavium, Inc.

"The book can help the readers to understand the principles of parallel systems crystally clear. A necessary book to read for the designers of parallel systems."

Yunji Chen, Institute of Computing Technology, Chinese Academy of Sciences

"All future electronic systems will comprise of a built-in microprocessor, consequently the importance of computer architecture will surge. This book provides an excellent tutorial of computer architecture fundamentals from the basic technology via processor and memory architecture to chip multiprocessors. I found the book very educationally flow and readable – an excellent instructive book worth using."

Uri Weiser, Technion

"This book really fulfils the need to understand the basic technological on-chip features and constraints in connection with their impact on computer architecture design choices. All computing systems students and developers should first master these single and multi core foundations in a platform independent way, as this comprehensive text does."

Mateo Valero, BSC

"After the drastic shift towards multi-cores that processor architecture has experienced in the past few years, the domain was in dire need of a comprehensive and up-to-date book on the topic. Michel, Murali, and Per have crafted an excellent textbook which can serve both as an introduction to multi-core and parallel architectures, as well as a reference for engineers and researchers."

Olivier Temam, INRIA, France

"*Parallel Computer Organization and Design* fills an urgent need for a comprehensive and authoritative yet approachable tutorial and reference text for advanced computer architecture topics. All of the key principles and concepts covered in Wisconsin's three-course computer architecture sequence are addressed in a well-organized, thoughtful, and pedagogically appealing manner, without overwhelming the reader with distracting trivia or an excess of quantitative data. In particular, the coverage of chip multiprocessors in Chapter 8 is fully up to date with the state of the art in industry practice, while the final chapter on quantitative evaluation – a true gem! – is a unique and valuable asset that will clearly set this book apart from its competition."

Mikko Lipasti, University of Wisonsin-Madison

"The book contains in-depth coverage of all the aspects of the computer systems. It is comprehensive, systematic, and in sync with the latest development in the field. The skillfully organized book uses self-contained chapters to allow the readers get a complete understanding of a topic without wandering through the whole book. Its content is rich, coherent and clear. Its questions are crafted to stimulate creative thinking. I recommend the book as a must read to all graduate students and young researchers and engineers designing the computers."

Lixin Zhang, Institute of Computing Technology, Chinese Academy of Sciences

"... parallel architectures are the key for high performance and high efficiency computing systems. This book tells the story of parallel architecture at all levels – from the single transistor to the full blown CMP – an unforgettable journey!"

Ronny Ronen, Intel

"Multicore chips have made parallel architectures ubiquitous and their understanding a necessity. This text provides a comprehensive treatment of parallel system architecture and the fundamentals of cache coherence and memory consistency in the most compact form to date. This is a perfect text for a one semester graduate course."

Lawrence Rauchwerger, Texas A&M University

"It is the best of today's books on the subject, and I plan to use it in my class. It is an up-to-date picture of parallel computing that is written in a style that is clear and accessible."

Trevor Mudge, Bredt Family Professor of Computer Engineering, University of Michigan

"Parallelism, at multiple levels and in many different forms, is now a necessity for all future computer systems, and the new generation of computer scientists and engineers have to master it. To understand the complex interactions among the hundreds of existing ideas, options, and choices, one has to categorize them, put them in order, and then synthesize them. That is precisely what Dubois, Annavaram, and Stenström do, in a magnificent way, in this extremely contemporary and timely book. I want to particularly stress the uniquely clear way in which the authors explain the hardest among these topics: coherence, synchronization, and memory consistency."

Manolis Katevenis, Professor of Computer Science, University of Crete

"This book is a truly comprehensive treatment of parallel computers, from some of the top experts in the field. Well grounded in technology yet remaining very accessible, it also includes important but often overlooked topics such reliability, power, and simulation."

Norm Jouppi, HP

"This text takes a fresh cut at traditional computer architecture topics and considers basic principles from the perspective of multi-core and parallel systems. The need for such a high quality textbook written from this perspective is overdue, and the authors of this text have done a good job in organizing and revamping topics to provide the next generation of computer architects with the basic principles they will need to design multi-core and many-core systems."

David Kaeli, Director of the NU Computer Architecture Research Laboratory, NEU

"An excellent book in an area that has long cried out for tutorial material – it will be an indispensable resource to students and educators in parallel computer architecture."

Josep Torrellas, University of Illinois

Parallel Computer Organization and Design

MICHEL DUBOIS
University of Southern California, USA

MURALI ANNAVARAM
University of Southern California, USA

PER STENSTRÖM
Chalmers University of Technology, Sweden

CAMBRIDGE UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9780521886758

First published 2012
Reprinted 2014

Printed in the United Kingdom by TJ International Ltd, Padstow, Cornwall

A catalog record for this publication is available from the British Library

Library of Congress Cataloging in Publication data
Dubois, Michel, 1953–
Parallel computer organization and design / Michel Dubois, Murali Annavaram, Per Stenström.
pages cm
Includes index.
ISBN 978-0-521-88675-8
1. Parallel computers. 2. Computer organization. I. Annavaram, Murali.
II. Stenström, Per. III. Title.
QA76.5.D754 2012
005.2′75 – dc23 2012010634

ISBN 978-0-521-88675-8 Hardback

Additional resources for this publication at www.cambridge.org/9780521886758

CONTENTS

PREFACE

Computer architecture is a fast evolving field, mostly because it is driven by rapidly changing technologies. We have all been accustomed to phenomenal improvements in the speed and reliability of computing systems since the mid 1990s, mostly due to technology improvements, faster clock rates, and deeper pipelines. These improvements have had a deep impact on society by bringing high-performance computing to the masses, by enabling the internet revolution and by fostering huge productivity gains in all human activities. We are in the midst of an information revolution of the same caliber as the industrial revolution of the eighteenth century, and few would deny that this revolution has been fueled by advances in technology and microprocessor architecture.

Unfortunately, these rapid improvements in computing systems may not be sustainable in future. Pipeline depths have reached their useful limit, and frequency cannot be cranked up for ever because of power constraints. As technology evolves and on-chip feature sizes shrink, reliability, complexity, and power/energy issues have become prime considerations in computer design, besides traditional measures such as cost, area, and performance. These trends have ushered a renaissance of parallel processing and parallel architectures, because they offer a clear path – some would say the only path – to solving all current and foreseeable problems in architecture. A widespread belief today is that, unless we can unleash and harness the power of parallel processing, the computing landscape will be very different very soon, and this dramatic change will have profound societal impacts. Thus interest in parallel architectures both in industry and in academia has turned from an engineering curiosity to an absolute necessity.

Over time, parallelism at all levels has gradually become the major approach to solve bottleneck problems in modern computer systems. Multiprocessor architectures, which provide scalable performance by simply connecting multiple processors together, have been the mainstay of high-end systems for decades. Multiprocessors exploit thread-level parallelism (TLP). They have been an enabling technology for large application domains with abundant threads, such as computer graphics, scientific/engineering computing, database management, and telecommunication services. Historically, microarchitectures have derived their superior performance from instruction-level parallelism (ILP) for years by advances in architecture and compiler technologies. Memory system architectures have evolved rapidly to keep up with the instruction throughput of processors by executing a large number of memory accesses concurrently while preserving correctness of execution. Interconnects and protocols are constantly improving to connect efficiently hundreds or thousands of processors as well as a few cores on chips clocked at gigahertz. Recently, the architecture of microprocessors has integrated system-level parallel architecture paradigms such as vector processing and multiprocessing. In this era of chip

multiprocessors, each microprocessor has multiple cores or CPUs, and each core can execute multiple threads concurrently.

Parallel architectures are hard to design and to program. Thus it is important to understand the unique problems caused by parallel architectures. This book provides a timely and comprehensive treatment of the design principles and techniques employed to exploit parallelism at instruction and thread levels. Furthermore it introduces reliability and power/energy as design targets. Previous books in computer architecture emphasize performance as the quintessential design concern for computer systems. However, while performance is still a major design criterion nowadays, other issues such as complexity, power, and reliability have emerged as first-rate design factors, and this new book in parallel computer architecture will include discussions on these factors.

The basic intention of this book is to explain how parallel architectures work and how to analyze the correct designs of today's parallel architectures, especially within technological constraints. We do not show performance data. We intentionally shy away from lengthy detailed descriptions of particular systems. Rather, the reader is encouraged to read published material in conferences and journals. Detailed bibliographies and historical perspective will be posted online. This leaves plenty of room in the book to explain design fundamentals. Students should be encouraged to think about, create, and realize their own designs. To achieve this level of practical knowledge and innovation, a thorough understanding of existing design practices and of the practical issues and limiting factors is fundamental. Nevertheless examples are used profusely throughout the book to illuminate the concepts and provoke readers into thinking on their own about the material. Moreover, two chapters (Chapter 8, on "Chip multiprocessors," and Chapter 9, on "Quantitative evaluations,") describe a number of machines and tools developed in industry and academia.

Exercises are an important part of the learning experience. The problems proposed after each chapter are "paper and pencil" problems. Some of them are very long and complex and could be broken up into subproblems. The main goal is to give students an opportunity to think hard and in depth about the design concepts exposed in each chapter while testing their ability to think abstractly.

The book is intended for both senior undergraduate and graduate students interested in computer architecture, including computer or electrical engineering students and computer science students. Additionally the book will also be of interest to practitioners and engineers in the computer industry. Because the book covers a wide range of architectures from microprocessors to multiprocessors and has basic as well as advanced research topics, it can be used in courses with various difficulty levels and themes by carefully selecting chapters and sections. Students will learn the hardware structures and components comprising multiprocessor architectures, the impact of technological trends on these architectures, and the design issues related to performance, power, reliability, and functional correctness. For example, the book can be used in a basic graduate course and, as a follow on, in an advanced, research course. The prerequisite is a basic computer architecture and organization course covering instruction sets and simple pipelined processor architecture. For example, exposure to a course on the 5-stage pipeline and its control mechanisms such as forwarding, stalling, and stage flushing in detail is most helpful

in order to understand the more complex hardware issues covered in the microarchitecture section. Basic topics on instruction sets and basic pipeline and memory concepts have been included in the book to make it self-contained. It is necessary to understand the working of a modern microarchitecture as it affects multiprocessor behavior. Furthermore, prior exposure to computer programming is, of course, necessary.

Book outline

The book is self-contained and we have made every attempt to make each chapter self-contained as well, even at the risk of being repetitious. It is organized in nine chapters. The first chapter (the introduction) gives a perspective on the field of computer architecture. The main components of this introduction are an overview of trends in processors, memories, and interconnects, a coverage of performance issues (mainly how to evaluate computer systems), and the impact of technology on future architectures.

Understanding the technological landscape in some level of detail is very important since so many design decisions in architecture today are guided by the impact of technology. Chapter 2 is a refresher on CMOS technology and the relevant issues connected with it. Some of this material can be skipped by students who have a background in VLSI design. It is mostly intended for computer science students who may have a very cursory knowledge of electrical engineering and CMOS technology. The knowledge of these key technology aspects is not a requirement for understanding the rest of the book, but it does help students understand why architecture is the way it is today and why some design decisions are made. This chapter is very different in nature from the rest of the book, as it is purely about technology.

Chapters 3, 4, and 6 describe the design of the basic building blocks of parallel systems: processors, memory, and interconnects. Chapter 3 covers microarchitectures. Instruction sets and basic machine organizations (such as the 5-stage pipeline) are briefly overviewed. In the process, the set of instructions and the basic ISA mechanisms adopted in the rest of the book are laid out. Special emphasis is given to exceptions and the way to treat them because exceptions have a great impact on how parallelism can be exploited in microarchitectures. A lot of this material can be skipped by students who already have some background in architecture. The major part of this chapter is the exploitation of instruction-level parallelism through various paradigms involving both hardware and software. At first, design issues of statically scheduled processors, including superscalar processors, which are extensions of the 5-stage pipeline, are presented. Since they have no mechanism to optimize the scheduling of instructions and take advantage of ILP, compiler technology is essential to their efficiency. Dynamic out-of-order (OoO) processors are able to re-schedule (after the compiler) instructions dynamically in large execution windows of hundreds of instructions. The evolution of OoO processor designs is presented step by step, starting with the Tomasulo algorithm and ending with speculative processors with speculative scheduling, the most advanced OoO architecture as of today. Out-of-order processors are at one end of the spectrum of processor architecture because their scheduling mechanism is dynamic. The problem with them is that their complexity and power consumption grow rapidly with the number of instructions executed in parallel. At the other end

of the parallel microarchitecture spectrum lie very long instruction word (VLIW) architectures. In such architectures, all decisions (including when to fetch, decode, and start instruction execution) are all made at compile time, which greatly reduces hardware complexity and power/energy consumption. Possibly architectures should adopt a compromise between the two extremes, and this was attempted in so-called EPIC (explicitly parallel instruction computing) architectures. Finally fine-grain parallelism is exploited in vector microarchitectures. Vector processing is efficient from both a performance and a power/energy point of view for multimedia and signal processing applications.

Chapter 4 is about the fundamental properties of memory hierarchies at the hardware level. Highly concurrent memory hierarchies are needed to feed parallel microarchitectures with data and instructions. This includes lockup-free cache design and software/hardware prefetching techniques. These techniques must ensure that memory behavior remains correct. Another factor important to the understanding of parallel architecture is the virtual memory system. Because of virtual memory, modern architectures must be capable of taking precise exceptions, and multiprocessors must include mechanisms to enforce the coherence of the structures supporting virtual memory in each processor (covered in Chapter 5).

Fundamentals of interconnection networks are the topic of Chapter 6. Interconnection networks connect system components (system area networks or SANs) or on-chip resources (on-chip networks or OCNs) in chip multiprocessors. Since allowing parallel access among components is critical to the performance of parallel architectures, the design of interconnection networks is critical to performance and power consumption. The design space is, however, huge. Chapter 6 provides a comprehensive overview of design principles for interconnection networks, including performance models, switching strategies, network topologies, routing algorithms, and the architecture of switches.

Chapters 5, 7, and 8 are dedicated to multiprocessors. In Chapter 5 the basic architectures and mechanisms of message-passing and shared-memory multiprocessors are exposed. At first the programming models and basic application programming interfaces are explained through program examples, which allows the reader to understand the types of mechanisms needed in the architecture. The basic architectural support required by message-passing architectures is presented in layers, from the various forms of message-passing primitives, to the basic protocol exchanges to implement them, to the basic hardware support to accelerate them. The balance of Chapter 5 focuses on the architectures of shared-memory systems. There are several possible computer organizations for shared-memory systems. One common denominator of these organizations is that, for economical reasons, multiprocessors must be built with off-the-shelf microprocessors, and these OTS microprocessors have each their own set of caches. Every processor in a shared-memory system and every core in a chip multiprocessor have private caches for instructions, for data and for virtual address translations. Therefore mechanisms must exist to maintain coherence among these structures. The chapter includes bus-based systems, and systems with distributed shared memory (cc-NUMAs and COMAs).

While the shared-memory coverage in Chapter 5 is about architectural mechanisms, Chapter 7 addresses logical properties of shared-memory multiprocessors, including synchronization, coherence, and the memory consistency model. There are close and subtle interactions

among these three features. Synchronization primitives and mechanisms are critical to correct execution of multi-threaded programs and must be supported in hardware. Coherence is needed between multiple copies of the same address in various caches and memory buffers. The ultimate correctness property of shared-memory systems is the memory consistency model, which dictates the possible dynamic interleavings of memory accesses. Concrete implementations of memory-consistency models are described in the contexts of both statically and dynamically scheduled processors. Chapter 7 is the most theoretical chapter in the book. However, no theoretical background is assumed.

Chapter 8 addresses chip multiprocessors (CMPs). Because of their tight integration and low latency communication capabilities, CMPs have the potential to enable new, easier, and more efficient programming models. In a CMP environment, CPUs are relatively inexpensive and can be used for all kinds of new, innovative modes of computation. This chapter covers such diverse topics as CMP architectures, core multi-threading, transactional memory, speculative thread parallelization, and assisted execution.

Finally, Chapter 9 focuses on quantitative evaluation methods for computer architecture designs. Most design decisions in computer architecture are based on a complex set of trade-offs between area, performance, power, and reliability. Hence, any design that intuitively improves on prior work must be thoroughly evaluated to quantify the improvement. As such, it is necessary for students and practitioners to understand quantitative methods for design space exploration. We cover a broad range of topics such as simulation methodologies, sampling techniques, and workload characterization approaches in this chapter.

Acknowledgments

This book took five years to write. It is derived from our experience teaching architecture to both undergraduate and graduate students and from the notes and exercises we developed over the years. Parallel computer architectures and parallel processing are here to stay and will play a key role in both the short and long terms. Thus we must continue to educate computer science and computer engineering students the best we can in both parallel architectures and programming. This book is our contribution towards this goal.

Michel Dubois

Over the years I have had the privilege to be influenced by many outstanding colleagues, unbeknownst to most of them. I would like to acknowledge my advisor at Purdue University, Fayé Briggs, who gave me confidence in my research abilities at the time I needed it most. The University of Minnesota and Purdue University provided me with the graduate education I needed to face the world. This was invaluable.

As a faculty member at USC I have learned a great deal from my Ph.D. students, and I hope they have learned from me too. They helped me build up and develop my ideas (some good and some bad) and my perspective on computer architecture. They are: Christoph Scheurich, Aydin Uresin, Jin-Chin Wang, Fong Pong, Luiz Barroso, Kangwoo Lee, Adrian Moga, Xiaogang Qiu,

Jaeheon Jeong, Jianwei Chen, and Jinho Suh. In particular, Jinho helped us greatly with the book.

I am of course forever indebted to my late parents, Solange and André, who brought me into this world and supported me when I decided to pursue graduate studies in the USA, even if that decision broke their heart. I also want to thank my wife, Lorraine, for her love and her constant support of my professional endeavors.

Finally I can speak for the three of us when I say that we owe a great deal to the team at Cambridge Press: Julie Lancashire, whose enthusiasm for this project was contagious from the get go; Sarah Matthews, who worked on many of the details of book production and sustained our enthusiasm during the whole process; and Irene Pizzie, who mercilessly corrected our grammar and worked very hard to maintain the consistency of the text throughout the whole book.

Murali Annavaram

When I accepted the invitation from Professor Michel Dubois to join hands on this worthy endeavor, little did I know the amount of time and energy a book of this magnitude consumes. However, after two years of working on this book it is gratifying to see that our collective effort resulted in a book that is greater than what each of us could bring to the table in isolation. Through this book I wanted to share the knowledge I gained about computer architecture over years of industrial and academic experience. Whatever I have contributed to this book I have learned from the masters in the field. In particular, the influence of Professor Edward S. Davidson is immeasurable. I miss his red and blue ink. Professor Yale Part has taught me how to teach, and his style is simply contagious. I would also like to acknowledge Professor Walid Najjar and Professor Farnam Jahanian for taking a chance with me.

In my industrial career no one exerted greater influence than John Shen. He truly was an amazing boss who knew how to herd the cats. There are others that also should be thanked: Ed Grochowski, the Zen master of creative thinking (and Pentium RTL); Bryan Black and his Austin gang for giving me 3D vision; Quinn Jacobson for trusting me with mobile systems; and Viji Srinivasan for the long discussions on prefetching.

The research I did at USC formed the foundation for some of the material in the book. The generous support I received from NSF and Nokia enabled me to pursue my research full throttle. This research work would not have been possible without the help of my wonderful graduate student group. They were the guinea pigs for the chapters I wrote and they provided solutions to the Exercises. Particular thanks are due to Jinho Suh, who never failed to amaze me with his all-round abilities, from statistics to Framemaker edits.

Bob and Nancy taught me the value of giving back to society that which I have learned. Kirti, Kalpesh, and the UM and CSU gang provided me with unparalleled support. The risk of mentioning some names results in the inevitability of leaving out others. There are many; you know who you are, and thank you.

Finally, to my family, who put up with my writing schedule to forgo some of their personal time, thank you.

Per Stenström

To a significant extent my contribution to this textbook is rooted in my research on parallel computer architecture over the years. I would never have pursued this route without the great inspiration of my advisor, Lars Philipson, who with his keen vision could see the potential of shared-memory multiprocessor technology as long ago as the early 1980s. More importantly, he enriched me with a sound view on science and gave me the confidence to form my own research program that, following in his footsteps, engaged a new generation of Ph.D. students that, under my supervision, gave me inspiration and taught me a lot. The lessons learned from the research projects over the years are condensed in this textbook. I am indebted to all my former Ph.D. students with whom I learned a great deal. Thanks go to: Mats Brorsson, Fredrik Dahlgren, Magnus Ekman, Håkan Grahn, Mafijul Islam, Thomas Lundqvist, Magnus Karlsson, Jim Nilsson, Ashley Saulsbury, Jonas Skeppstedt, Martin Thuresson, M. M. Waliullah, and FredrikWarg.

Apart from the "inner academic family," I would like to acknowledge the impact that so many people in the computer architecture community have had on my professional development over the years. My visits to Carnegie-Mellon University, Stanford University, the University of Southern California, and Sun Microsystems had a profound influence on my perspectives and views on what computer architecture is all about. Another important source of inspiration has been my participation and interaction with European colleagues in the formation of the network of excellence on high-performance and embedded architectures and compilers (HiPEAC). I wish I could list all the people I am deeply indebted to, but the list would be too long: thanks to all of you.

We have class-tested earlier drafts of this textbook at Chalmers and received lots of useful feedback to improve it. I am particularly indebted to the following people who provided feedback to us: Bhavishya Goel, Ben Juurlink, Johan Karlsson, Sally McKee, Filippo Del Tedesco, Andras Vajda, and M. M. Waliullah.

Finally, and indeed most importantly, I would like to thank my wife Carina and my daughter Sofia for all the love and support they have wholeheartedly given me in my endeavor of deepening our understanding of the principles behind parallel computer architectures.