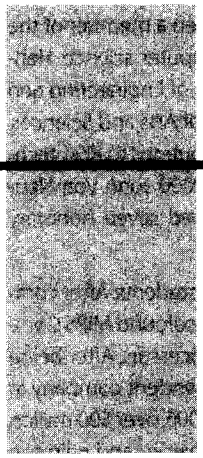

Computer Architecture
A Quantitative Approach

Fourth Edition

John L. Hennessy is the president of Stanford University, where he has been a member of the faculty since 1977 in the departments of electrical engineering and computer science. Hennessy is a Fellow of the IEEE and ACM, a member of the National Academy of Engineering and the National Academy of Science, and a Fellow of the American Academy of Arts and Sciences. Among his many awards are the 2001 Eckert-Mauchly Award for his contributions to RISC technology, the 2001 Seymour Cray Computer Engineering Award, and the 2000 John von Neumann Award, which he shared with David Patterson. He has also received seven honorary doctorates.

In 1981, he started the MIPS project at Stanford with a handful of graduate students. After completing the project in 1984, he took a one-year leave from the university to cofound MIPS Computer Systems, which developed one of the first commercial RISC microprocessors. After being acquired by Silicon Graphics in 1991, MIPS Technologies became an independent company in 1998, focusing on microprocessors for the embedded marketplace. As of 2006, over 500 million MIPS microprocessors have been shipped in devices ranging from video games and palmtop computers to laser printers and network switches.

David A. Patterson has been teaching computer architecture at the University of California, Berkeley, since joining the faculty in 1977, where he holds the Pardee Chair of Computer Science. His teaching has been honored by the Abacus Award from Upsilon Pi Epsilon, the Distinguished Teaching Award from the University of California, the Karlstrom Award from ACM, and the Mulligan Education Medal and Undergraduate Teaching Award from IEEE. Patterson received the IEEE Technical Achievement Award for contributions to RISC and shared the IEEE Johnson Information Storage Award for contributions to RAID. He then shared the IEEE John von Neumann Medal and the C & C Prize with John Hennessy. Like his co-author, Patterson is a Fellow of the American Academy of Arts and Sciences, ACM, and IEEE, and he was elected to the National Academy of Engineering, the National Academy of Sciences, and the Silicon Valley Engineering Hall of Fame. He served on the Information Technology Advisory Committee to the U.S. President, as chair of the CS division in the Berkeley EECS department, as chair of the Computing Research Association, and as President of ACM. This record led to a Distinguished Service Award from CRA.

At Berkeley, Patterson led the design and implementation of RISC I, likely the first VLSI reduced instruction set computer. This research became the foundation of the SPARC architecture, currently used by Sun Microsystems, Fujitsu, and others. He was a leader of the Redundant Arrays of Inexpensive Disks (RAID) project, which led to dependable storage systems from many companies. He was also involved in the Network of Workstations (NOW) project, which led to cluster technology used by Internet companies. These projects earned three dissertation awards from the ACM. His current research projects are the RAD Lab, which is inventing technology for reliable, adaptive, distributed Internet services, and the Research Accelerator for Multiple Processors (RAMP) project, which is developing and distributing low-cost, highly scalable, parallel computers based on FPGAs and open-source hardware and software.

Computer Architecture
A Quantitative Approach

Fourth Edition

John L. Hennessy
Stanford University

David A. Patterson
University of California at Berkeley

With Contributions by

Andrea C. Arpaci-Dusseau
University of Wisconsin–Madison

Remzi H. Arpaci-Dusseau
University of Wisconsin–Madison

Krste Asanovic
Massachusetts Institute of Technology

Robert P. Colwell
R&E Colwell & Associates, Inc.

Thomas M. Conte
North Carolina State University

José Duato
Universitat Politècnica de València and *Simula*

Diana Franklin
California Polytechnic State University, San Luis Obispo

David Goldberg
Xerox Palo Alto Research Center

Wen-mei W. Hwu
University of Illinois at Urbana–Champaign

Norman P. Jouppi
HP Labs

Timothy M. Pinkston
University of Southern California

John W. Sias
University of Illinois at Urbana–Champaign

David A. Wood
University of Wisconsin–Madison

Morgan Kaufmann Publishers

An imprint of Elsevier

Computer Architecture: A Quantitative Approach, 4ed
Hennessy and Patterson

Morgan Kaufmann Publishers
An Imprint of Elsevier
500 Sansome Street, Suite 400, San Francisco, CA 94111

Original ISBN: 978-0-12-370490-0

First Printed in India 2007
Reprinted 2007
Reprinted 2008 (thrice)
Reprinted 2009 (thrice)
Reprinted 2010

Indian Reprint ISBN: 978-81-312-0726-0

This edition is for sale in Bangladesh, Bhutan, India, Maldives, Nepal, Pakistan, Sri Lanka and designated countries in South-East Asia through Elsevier (Singapore) Pte. Ltd. Sale and purchase of this book outside of these countries is unauthorised by the publisher.

Published by Elsevier, a division of Reed Elsevier India Private Limited.

Registered Office: Gate No. 3, Shed A-1, 2 Industrial Area, Kalkaji, New Delhi-110 019.

Sales and Publishing Office: 14th Floor, Building No. 10B, DLF Cyber City, Phase-II, Gurgaon-122 002, Haryana, India.

Printed and bound at Rajkamal Electric Press, Plot No. 2, Phase IV, Kundli (Haryana).

To Andrea, Linda, and our four sons

Foreword

by Fred Weber, President and CEO of MetaRAM, Inc.

I am honored and privileged to write the foreword for the fourth edition of this most important book in computer architecture. In the first edition, Gordon Bell, my first industry mentor, predicted the book's central position as the definitive text for computer architecture and design. He was right. I clearly remember the excitement generated by the introduction of this work. Rereading it now, with significant extensions added in the three new editions, has been a pleasure all over again. No other work in computer architecture—frankly, no other work I have read in any field—so quickly and effortlessly takes the reader from ignorance to a breadth and depth of knowledge.

This book is dense in facts and figures, in rules of thumb and theories, in examples and descriptions. It is stuffed with acronyms, technologies, trends, formulas, illustrations, and tables. And, this is thoroughly appropriate for a work on architecture. The architect's role is not that of a scientist or inventor who will deeply study a particular phenomenon and create new basic materials or techniques. Nor is the architect the craftsman who masters the handling of tools to craft the finest details. The architect's role is to combine a thorough understanding of the state of the art of what is possible, a thorough understanding of the historical and current styles of what is desirable, a sense of design to conceive a harmonious total system, and the confidence and energy to marshal this knowledge and available resources to go out and get something built. To accomplish this, the architect needs a tremendous density of information with an in-depth understanding of the fundamentals and a quantitative approach to ground his thinking. That is exactly what this book delivers.

As computer architecture has evolved—from a world of mainframes, minicomputers, and microprocessors, to a world dominated by microprocessors, and now into a world where microprocessors themselves are encompassing all the complexity of mainframe computers—Hennessy and Patterson have updated their book appropriately. The first edition showcased the IBM 360, DEC VAX, and Intel 80x86, each the pinnacle of its class of computer, and helped introduce the world to RISC architecture. The later editions focused on the details of the 80x86 and RISC processors, which had come to dominate the landscape. This latest edition expands the coverage of threading and multiprocessing, virtualization

and memory hierarchy, and storage systems, giving the reader context appropriate to today's most important directions and setting the stage for the next decade of design. It highlights the AMD Opteron and SUN Niagara as the best examples of the x86 and SPARC (RISC) architectures brought into the new world of multiprocessing and system-on-a-chip architecture, thus grounding the art and science in real-world commercial examples.

The first chapter, in less than 60 pages, introduces the reader to the taxonomies of computer design and the basic concerns of computer architecture, gives an overview of the technology trends that drive the industry, and lays out a quantitative approach to using all this information in the art of computer design. The next two chapters focus on traditional CPU design and give a strong grounding in the possibilities and limits in this core area. The final three chapters build out an understanding of system issues with multiprocessing, memory hierarchy, and storage. Knowledge of these areas has always been of critical importance to the computer architect. In this era of system-on-a-chip designs, it is essential for every CPU architect. Finally the appendices provide a great depth of understanding by working through specific examples in great detail.

In design it is important to look at both the forest and the trees and to move easily between these views. As you work through this book you will find plenty of both. The result of great architecture, whether in computer design, building design or textbook design, is to take the customer's requirements and desires and return a design that causes that customer to say, "Wow, I didn't know that was possible." This book succeeds on that measure and will, I hope, give you as much pleasure and value as it has me.

Contents

Preface

Why We Wrote This Book

Through four editions of this book, our goal has been to describe the basic principles underlying what will be tomorrow's technological developments. Our excitement about the opportunities in computer architecture has not abated, and we echo what we said about the field in the first edition: "It is not a dreary science of paper machines that will never work. No! It's a discipline of keen intellectual interest, requiring the balance of marketplace forces to cost-performance-power, leading to glorious failures and some notable successes."

Our primary objective in writing our first book was to change the way people learn and think about computer architecture. We feel this goal is still valid and important. The field is changing daily and must be studied with real examples and measurements on real computers, rather than simply as a collection of definitions and designs that will never need to be realized. We offer an enthusiastic welcome to anyone who came along with us in the past, as well as to those who are joining us now. Either way, we can promise the same quantitative approach to, and analysis of, real systems.

As with earlier versions, we have strived to produce a new edition that will continue to be as relevant for professional engineers and architects as it is for those involved in advanced computer architecture and design courses. As much as its predecessors, this edition aims to demystify computer architecture through an emphasis on cost-performance-power trade-offs and good engineering design. We believe that the field has continued to mature and move toward the rigorous quantitative foundation of long-established scientific and engineering disciplines.

This Edition

The fourth edition of *Computer Architecture: A Quantitative Approach* may be the most significant since the first edition. Shortly before we started this revision, Intel announced that it was joining IBM and Sun in relying on multiple processors or cores per chip for high-performance designs. As the first figure in the book documents, after 16 years of doubling performance every 18 months, sin-

gle-processor performance improvement has dropped to modest annual improvements. This fork in the computer architecture road means that for the first time in history, no one is building a much faster sequential processor. If you want your program to run significantly faster, say, to justify the addition of new features, you're going to have to parallelize your program.

Hence, after three editions focused primarily on higher performance by exploiting instruction-level parallelism (ILP), an equal focus of this edition is thread-level parallelism (TLP) and data-level parallelism (DLP). While earlier editions had material on TLP and DLP in big multiprocessor servers, now TLP and DLP are relevant for single-chip multicores. This historic shift led us to change the order of the chapters: the chapter on multiple processors was the sixth chapter in the last edition, but is now the fourth chapter of this edition.

The changing technology has also motivated us to move some of the content from later chapters into the first chapter. Because technologists predict much higher hard and soft error rates as the industry moves to semiconductor processes with feature sizes 65 nm or smaller, we decided to move the basics of dependability from Chapter 7 in the third edition into Chapter 1. As power has become the dominant factor in determining how much you can place on a chip, we also beefed up the coverage of power in Chapter 1. Of course, the content and examples in all chapters were updated, as we discuss below.

In addition to technological sea changes that have shifted the contents of this edition, we have taken a new approach to the exercises in this edition. It is surprisingly difficult and time-consuming to create interesting, accurate, and unambiguous exercises that evenly test the material throughout a chapter. Alas, the Web has reduced the half-life of exercises to a few months. Rather than working out an assignment, a student can search the Web to find answers not long after a book is published. Hence, a tremendous amount of hard work quickly becomes unusable, and instructors are denied the opportunity to test what students have learned.

To help mitigate this problem, in this edition we are trying two new ideas. First, we recruited experts from academia and industry on each topic to write the exercises. This means some of the best people in each field are helping us to create interesting ways to explore the key concepts in each chapter and test the reader's understanding of that material. Second, each group of exercises is organized around a set of case studies. Our hope is that the quantitative example in each case study will remain interesting over the years, robust and detailed enough to allow instructors the opportunity to easily create their own new exercises, should they choose to do so. Key, however, is that each year we will continue to release new exercise sets for each of the case studies. These new exercises will have critical changes in some parameters so that answers to old exercises will no longer apply.

Another significant change is that we followed the lead of the third edition of *Computer Organization and Design (COD)* by slimming the text to include the material that almost all readers will want to see and moving the appendices that

some will see as optional or as reference material onto a companion CD. There were many reasons for this change:

1. Students complained about the size of the book, which had expanded from 594 pages in the chapters plus 160 pages of appendices in the first edition to 760 chapter pages plus 223 appendix pages in the second edition and then to 883 chapter pages plus 209 pages in the paper appendices and 245 pages in online appendices. At this rate, the fourth edition would have exceeded 1500 pages (both on paper and online)!
2. Similarly, instructors were concerned about having too much material to cover in a single course.
3. As was the case for *COD*, by including a CD with material moved out of the text, readers could have quick access to all the material, regardless of their ability to access Elsevier's Web site. Hence, the current edition's appendices will always be available to the reader even after future editions appear.
4. This flexibility allowed us to move review material on pipelining, instruction sets, and memory hierarchy from the chapters and into Appendices A, B, and C. The advantage to instructors and readers is that they can go over the review material much more quickly and then spend more time on the advanced topics in Chapters 2, 3, and 5. It also allowed us to move the discussion of some topics that are important but are not core course topics into appendices on the CD. Result: the material is available, but the printed book is shorter. In this edition we have 6 chapters, none of which is longer than 80 pages, while in the last edition we had 8 chapters, with the longest chapter weighing in at 127 pages.
5. This package of a slimmer core print text plus a CD is far less expensive to manufacture than the previous editions, allowing our publisher to significantly lower the list price of the book. With this pricing scheme, there is no need for a separate international student edition for European readers.

Yet another major change from the last edition is that we have moved the embedded material introduced in the third edition into its own appendix, Appendix D. We felt that the embedded material didn't always fit with the quantitative evaluation of the rest of the material, plus it extended the length of many chapters that were already running long. We believe there are also pedagogic advantages in having all the embedded information in a single appendix.

This edition continues the tradition of using real-world examples to demonstrate the ideas, and the "Putting It All Together" sections are brand new; in fact, some were announced after our book was sent to the printer. The "Putting It All Together" sections of this edition include the pipeline organizations and memory hierarchies of the Intel Pentium 4 and AMD Opteron; the Sun T1 ("Niagara") 8-processor, 32-thread microprocessor; the latest NetApp Filer; the Internet Archive cluster; and the IBM Blue Gene/L massively parallel processor.

Topic Selection and Organization

As before, we have taken a conservative approach to topic selection, for there are many more interesting ideas in the field than can reasonably be covered in a treatment of basic principles. We have steered away from a comprehensive survey of every architecture a reader might encounter. Instead, our presentation focuses on core concepts likely to be found in any new machine. The key criterion remains that of selecting ideas that have been examined and utilized successfully enough to permit their discussion in quantitative terms.

Our intent has always been to focus on material that is not available in equivalent form from other sources, so we continue to emphasize advanced content wherever possible. Indeed, there are several systems here whose descriptions cannot be found in the literature. (Readers interested strictly in a more basic introduction to computer architecture should read *Computer Organization and Design: The Hardware/Software Interface,* third edition.)

An Overview of the Content

Chapter 1 has been beefed up in this edition. It includes formulas for static power, dynamic power, integrated circuit costs, reliability, and availability. We go into more depth than prior editions on the use of the geometric mean and the geometric standard deviation to capture the variability of the mean. Our hope is that these topics can be used through the rest of the book. In addition to the classic quantitative principles of computer design and performance measurement, the benchmark section has been upgraded to use the new SPEC2006 suite.

Our view is that the instruction set architecture is playing less of a role today than in 1990, so we moved this material to Appendix B. It still uses the MIPS64 architecture. For fans of ISAs, Appendix J covers 10 RISC architectures, the 80x86, the DEC VAX, and the IBM 360/370.

Chapters 2 and 3 cover the exploitation of instruction-level parallelism in high-performance processors, including superscalar execution, branch prediction, speculation, dynamic scheduling, and the relevant compiler technology. As mentioned earlier, Appendix A is a review of pipelining in case you need it. Chapter 3 surveys the limits of ILP. New to this edition is a quantitative evaluation of multithreading. Chapter 3 also includes a head-to-head comparison of the AMD Athlon, Intel Pentium 4, Intel Itanium 2, and IBM Power5, each of which has made separate bets on exploiting ILP and TLP. While the last edition contained a great deal on Itanium, we moved much of this material to Appendix G, indicating our view that this architecture has not lived up to the early claims.

Given the switch in the field from exploiting only ILP to an equal focus on thread- and data-level parallelism, we moved multiprocessor systems up to Chapter 4, which focuses on shared-memory architectures. The chapter begins with the performance of such an architecture. It then explores symmetric and distributed-memory architectures, examining both organizational principles and performance. Topics in synchronization and memory consistency models are

next. The example is the Sun T1 ("Niagara"), a radical design for a commercial product. It reverted to a single-instruction issue, 6-stage pipeline microarchitecture. It put 8 of these on a single chip, and each supports 4 threads. Hence, software sees 32 threads on this single, low-power chip.

As mentioned earlier, Appendix C contains an introductory review of cache principles, which is available in case you need it. This shift allows Chapter 5 to start with 11 advanced optimizations of caches. The chapter includes a new section on virtual machines, which offers advantages in protection, software management, and hardware management. The example is the AMD Opteron, giving both its cache hierarchy and the virtual memory scheme for its recently expanded 64-bit addresses.

Chapter 6, "Storage Systems," has an expanded discussion of reliability and availability, a tutorial on RAID with a description of RAID 6 schemes, and rarely found failure statistics of real systems. It continues to provide an introduction to queuing theory and I/O performance benchmarks. Rather than go through a series of steps to build a hypothetical cluster as in the last edition, we evaluate the cost, performance, and reliability of a real cluster: the Internet Archive. The "Putting It All Together" example is the NetApp FAS6000 filer, which is based on the AMD Opteron microprocessor.

This brings us to Appendices A through L. As mentioned earlier, Appendices A and C are tutorials on basic pipelining and caching concepts. Readers relatively new to pipelining should read Appendix A before Chapters 2 and 3, and those new to caching should read Appendix C before Chapter 5.

Appendix B covers principles of ISAs, including MIPS64, and Appendix J describes 64-bit versions of Alpha, MIPS, PowerPC, and SPARC and their multimedia extensions. It also includes some classic architectures (80x86, VAX, and IBM 360/370) and popular embedded instruction sets (ARM, Thumb, SuperH, MIPS16, and Mitsubishi M32R). Appendix G is related, in that it covers architectures and compilers for VLIW ISAs.

Appendix D, updated by Thomas M. Conte, consolidates the embedded material in one place.

Appendix E, on networks, has been extensively revised by Timothy M. Pinkston and José Duato. Appendix F, updated by Krste Asanovic, includes a description of vector processors. We think these two appendices are some of the best material we know of on each topic.

Appendix H describes parallel processing applications and coherence protocols for larger-scale, shared-memory multiprocessing. Appendix I, by David Goldberg, describes computer arithmetic.

Appendix K collects the "Historical Perspective and References" from each chapter of the third edition into a single appendix. It attempts to give proper credit for the ideas in each chapter and a sense of the history surrounding the inventions. We like to think of this as presenting the human drama of computer design. It also supplies references that the student of architecture may want to pursue. If you have time, we recommend reading some of the classic papers in the field that are mentioned in these sections. It is both enjoyable and educational

to hear the ideas directly from the creators. "Historical Perspective" was one of the most popular sections of prior editions.

Appendix L (available at *textbooks.elsevier.com/0123704901*) contains solutions to the case study exercises in the book.

Navigating the Text

There is no single best order in which to approach these chapters and appendices, except that all readers should start with Chapter 1. If you don't want to read everything, here are some suggested sequences:

- *ILP:* Appendix A, Chapters 2 and 3, and Appendices F and G
- *Memory Hierarchy:* Appendix C and Chapters 5 and 6
- *Thread-and Data-Level Parallelism:* Chapter 4, Appendix H, and Appendix E
- *ISA:* Appendices B and J

Appendix D can be read at any time, but it might work best if read after the ISA and cache sequences. Appendix I can be read whenever arithmetic moves you.

Chapter Structure

The material we have selected has been stretched upon a consistent framework that is followed in each chapter. We start by explaining the ideas of a chapter. These ideas are followed by a "Crosscutting Issues" section, a feature that shows how the ideas covered in one chapter interact with those given in other chapters. This is followed by a "Putting It All Together" section that ties these ideas together by showing how they are used in a real machine.

Next in the sequence is "Fallacies and Pitfalls," which lets readers learn from the mistakes of others. We show examples of common misunderstandings and architectural traps that are difficult to avoid even when you know they are lying in wait for you. The "Fallacies and Pitfalls" sections is one of the most popular sections of the book. Each chapter ends with a "Concluding Remarks" section.

Case Studies with Exercises

Each chapter ends with case studies and accompanying exercises. Authored by experts in industry and academia, the case studies explore key chapter concepts and verify understanding through increasingly challenging exercises. Instructors should find the case studies sufficiently detailed and robust to allow them to create their own additional exercises.

Brackets for each exercise (<chapter.section>) indicate the text sections of primary relevance to completing the exercise. We hope this helps readers to avoid exercises for which they haven't read the corresponding section, in addition to providing the source for review. Note that we provide solutions to the case study

exercises in Appendix L. Exercises are rated, to give the reader a sense of the amount of time required to complete an exercise:

[10] Less than 5 minutes (to read and understand)

[15] 5–15 minutes for a full answer

[20] 15–20 minutes for a full answer

[25] 1 hour for a full written answer

[30] Short programming project: less than 1 full day of programming

[40] Significant programming project: 2 weeks of elapsed time

[Discussion] Topic for discussion with others

A second set of alternative case study exercises are available for instructors who register at *textbooks.elsevier.com/0123704901*. This second set will be revised every summer, so that early every fall, instructors can download a new set of exercises and solutions to accompany the case studies in the book.

Supplemental Materials

The accompanying CD contains a variety of resources, including the following:

- Reference appendices—some guest authored by subject experts—covering a range of advanced topics
- Historical Perspectives material that explores the development of the key ideas presented in each of the chapters in the text
- Search engine for both the main text and the CD-only content

Additional resources are available at *textbooks.elsevier.com/0123704901*. The instructor site (accessible to adopters who register at *textbooks.elsevier.com*) includes:

- Alternative case study exercises with solutions (updated yearly)
- Instructor slides in PowerPoint
- Figures from the book in JPEG and PPT formats

The companion site (accessible to all readers) includes:

- Solutions to the case study exercises in the text
- Links to related material on the Web
- List of errata

New materials and links to other resources available on the Web will be added on a regular basis.

Helping Improve This Book

Finally, it is possible to make money while reading this book. (Talk about cost-performance!) If you read the Acknowledgments that follow, you will see that we went to great lengths to correct mistakes. Since a book goes through many printings, we have the opportunity to make even more corrections. If you uncover any remaining resilient bugs, please contact the publisher by electronic mail (*ca4bugs@mkp.com*). The first reader to report an error with a fix that we incorporate in a future printing will be rewarded with a $1.00 bounty. Please check the errata sheet on the home page (*textbooks.elsevier.com/0123704901*) to see if the bug has already been reported. We process the bugs and send the checks about once a year or so, so please be patient.

We welcome general comments to the text and invite you to send them to a separate email address at *ca4comments@mkp.com*.

Concluding Remarks

Once again this book is a true co-authorship, with each of us writing half the chapters and an equal share of the appendices. We can't imagine how long it would have taken without someone else doing half the work, offering inspiration when the task seemed hopeless, providing the key insight to explain a difficult concept, supplying reviews over the weekend of chapters, and commiserating when the weight of our other obligations made it hard to pick up the pen. (These obligations have escalated exponentially with the number of editions, as one of us was President of Stanford and the other was President of the Association for Computing Machinery.) Thus, once again we share equally the blame for what you are about to read.

John Hennessy ■ *David Patterson*

Acknowledgments

Although this is only the fourth edition of this book, we have actually created nine different versions of the text: three versions of the first edition (alpha, beta, and final) and two versions of the second, third, and fourth editions (beta and final). Along the way, we have received help from hundreds of reviewers and users. Each of these people has helped make this book better. Thus, we have chosen to list all of the people who have made contributions to some version of this book.

Contributors to the Fourth Edition

Like prior editions, this is a community effort that involves scores of volunteers. Without their help, this edition would not be nearly as polished.

Reviewers

Krste Asanovic, Massachusetts Institute of Technology; Mark Brehob, University of Michigan; Sudhanva Gurumurthi, University of Virginia; Mark D. Hill, University of Wisconsin–Madison; Wen-mei Hwu, University of Illinois at Urbana–Champaign; David Kaeli, Northeastern University; Ramadass Nagarajan, University of Texas at Austin; Karthikeyan Sankaralingam, Univeristy of Texas at Austin; Mark Smotherman, Clemson University; Gurindar Sohi, University of Wisconsin–Madison; Shyamkumar Thoziyoor, University of Notre Dame, Indiana; Dan Upton, University of Virginia; Sotirios G. Ziavras, New Jersey Institute of Technology

Focus Group

Krste Asanovic, Massachusetts Institute of Technology; José Duato, Universitat Politècnica de València and Simula; Antonio González, Intel and Universitat Politècnica de Catalunya; Mark D. Hill, University of Wisconsin–Madison; Lev G. Kirischian, Ryerson University; Timothy M. Pinkston, University of Southern California

Appendices

Krste Asanovic, Massachusetts Institute of Technology (Appendix F); Thomas M. Conte, North Carolina State University (Appendix D); José Duato, Universitat Politècnica de València and Simula (Appendix E); David Goldberg, Xerox PARC (Appendix I); Timothy M. Pinkston, University of Southern California (Appendix E)

Case Studies with Exercises

Andrea C. Arpaci-Dusseau, University of Wisconsin–Madison (Chapter 6); Remzi H. Arpaci-Dusseau, University of Wisconsin–Madison (Chapter 6); Robert P. Colwell, R&E Colwell & Assoc., Inc. (Chapter 2); Diana Franklin, California Polytechnic State University, San Luis Obispo (Chapter 1); Wen-mei W. Hwu, University of Illinois at Urbana–Champaign (Chapter 3); Norman P. Jouppi, HP Labs (Chapter 5); John W. Sias, University of Illinois at Urbana–Champaign (Chapter 3); David A. Wood, University of Wisconsin–Madison (Chapter 4)

Additional Material

John Mashey (geometric means and standard deviations in Chapter 1); Chenming Hu, University of California, Berkeley (wafer costs and yield parameters in Chapter 1); Bill Brantley and Dan Mudgett, AMD (Opteron memory hierarchy evaluation in Chapter 5); Mendel Rosenblum, Stanford and VMware (virtual machines in Chapter 5); Aravind Menon, EPFL Switzerland (Xen measurements in Chapter 5); Bruce Baumgart and Brewster Kahle, Internet Archive (IA cluster in Chapter 6); David Ford, Steve Kleiman, and Steve Miller, Network Appliances (FX6000 information in Chapter 6); Alexander Thomasian, Rutgers (queueing theory in Chapter 6)

Finally, a special thanks once again to Mark Smotherman of Clemson University, who gave a final technical reading of our manuscript. Mark found numerous bugs and ambiguities, and the book is much cleaner as a result.

This book could not have been published without a publisher, of course. We wish to thank all the Morgan Kaufmann/Elsevier staff for their efforts and support. For this fourth edition, we particularly want to thank Kimberlee Honjo who coordinated surveys, focus groups, manuscript reviews and appendices, and Nate McFadden, who coordinated the development and review of the case studies. Our warmest thanks to our editor, Denise Penrose, for her leadership in our continuing writing saga.

We must also thank our university staff, Margaret Rowland and Cecilia Pracher, for countless express mailings, as well as for holding down the fort at Stanford and Berkeley while we worked on the book.

Our final thanks go to our wives for their suffering through increasingly early mornings of reading, thinking, and writing.

Contributors to Previous Editions

Reviewers

George Adams, Purdue University; Sarita Adve, University of Illinois at Urbana–Champaign; Jim Archibald, Brigham Young University; Krste Asanovic, Massachusetts Institute of Technology; Jean-Loup Baer, University of Washington; Paul Barr, Northeastern University; Rajendra V. Boppana, University of Texas, San Antonio; Doug Burger, University of Texas, Austin; John Burger, SGI; Michael Butler; Thomas Casavant; Rohit Chandra; Peter Chen, University of Michigan; the classes at SUNY Stony Brook, Carnegie Mellon, Stanford, Clemson, and Wisconsin; Tim Coe, Vitesse Semiconductor; Bob Colwell, Intel; David Cummings; Bill Dally; David Douglas; Anthony Duben, Southeast Missouri State University; Susan Eggers, University of Washington; Joel Emer; Barry Fagin, Dartmouth; Joel Ferguson, University of California, Santa Cruz; Carl Feynman; David Filo; Josh Fisher, Hewlett-Packard Laboratories; Rob Fowler, DIKU; Mark Franklin, Washington University (St. Louis); Kourosh Gharachorloo; Nikolas Gloy, Harvard University; David Goldberg, Xerox Palo Alto Research Center; James Goodman, University of Wisconsin–Madison; David Harris, Harvey Mudd College; John Heinlein; Mark Heinrich, Stanford; Daniel Helman, University of California, Santa Cruz; Mark Hill, University of Wisconsin–Madison; Martin Hopkins, IBM; Jerry Huck, Hewlett-Packard Laboratories; Mary Jane Irwin, Pennsylvania State University; Truman Joe; Norm Jouppi; David Kaeli, Northeastern University; Roger Kieckhafer, University of Nebraska; Earl Killian; Allan Knies, Purdue University; Don Knuth; Jeff Kuskin, Stanford; James R. Larus, Microsoft Research; Corinna Lee, University of Toronto; Hank Levy; Kai Li, Princeton University; Lori Liebrock, University of Alaska, Fairbanks; Mikko Lipasti, University of Wisconsin–Madison; Gyula A. Mago, University of North Carolina, Chapel Hill; Bryan Martin; Norman Matloff; David Meyer; William Michalson, Worcester Polytechnic Institute; James Mooney; Trevor Mudge, University of Michigan; David Nagle, Carnegie Mellon University; Todd Narter; Victor Nelson; Vojin Oklobdzija, University of California, Berkeley; Kunle Olukotun, Stanford University; Bob Owens, Pennsylvania State University; Greg Papadapoulous, Sun; Joseph Pfeiffer; Keshav Pingali, Cornell University; Bruno Preiss, University of Waterloo; Steven Przybylski; Jim Quinlan; Andras Radics; Kishore Ramachandran, Georgia Institute of Technology; Joseph Rameh, University of Texas, Austin; Anthony Reeves, Cornell University; Richard Reid, Michigan State University; Steve Reinhardt, University of Michigan; David Rennels, University of California, Los Angeles; Arnold L. Rosenberg, University of Massachusetts, Amherst; Kaushik Roy, Purdue University; Emilio Salgueiro, Unysis; Peter Schnorf; Margo Seltzer; Behrooz Shirazi, Southern Methodist University; Daniel Siewiorek, Carnegie Mellon University; J. P. Singh, Princeton; Ashok Singhal; Jim Smith, University of Wisconsin–Madison; Mike Smith, Harvard University; Mark Smotherman, Clemson University; Guri Sohi, University of Wisconsin–Madison; Arun Somani, University of

Washington; Gene Tagliarin, Clemson University; Evan Tick, University of Oregon; Akhilesh Tyagi, University of North Carolina, Chapel Hill; Mateo Valero, Universidad Politécnica de Cataluña, Barcelona; Anujan Varma, University of California, Santa Cruz; Thorsten von Eicken, Cornell University; Hank Walker, Texas A&M; Roy Want, Xerox Palo Alto Research Center; David Weaver, Sun; Shlomo Weiss, Tel Aviv University; David Wells; Mike Westall, Clemson University; Maurice Wilkes; Eric Williams; Thomas Willis, Purdue University; Malcolm Wing; Larry Wittie, SUNY Stony Brook; Ellen Witte Zegura, Georgia Institute of Technology

Appendices

The vector appendix was revised by Krste Asanovic of the Massachusetts Institute of Technology. The floating-point appendix was written originally by David Goldberg of Xerox PARC.

Exercises

George Adams, Purdue University; Todd M. Bezenek, University of Wisconsin–Madison (in remembrance of his grandmother Ethel Eshom); Susan Eggers; Anoop Gupta; David Hayes; Mark Hill; Allan Knies; Ethan L. Miller, University of California, Santa Cruz; Parthasarathy Ranganathan, Compaq Western Research Laboratory; Brandon Schwartz, University of Wisconsin–Madison; Michael Scott; Dan Siewiorek; Mike Smith; Mark Smotherman; Evan Tick; Thomas Willis.

Special Thanks

Duane Adams, Defense Advanced Research Projects Agency; Tom Adams; Sarita Adve, University of Illinois at Urbana–Champaign; Anant Agarwal; Dave Albonesi, University of Rochester; Mitch Alsup; Howard Alt; Dave Anderson; Peter Ashenden; David Bailey; Bill Bandy, Defense Advanced Research Projects Agency; L. Barroso, Compaq's Western Research Lab; Andy Bechtolsheim; C. Gordon Bell; Fred Berkowitz; John Best, IBM; Dileep Bhandarkar; Jeff Bier, BDTI; Mark Birman; David Black; David Boggs; Jim Brady; Forrest Brewer; Aaron Brown, University of California, Berkeley; E. Bugnion, Compaq's Western Research Lab; Alper Buyuktosunoglu, University of Rochester; Mark Callaghan; Jason F. Cantin; Paul Carrick; Chen-Chung Chang; Lei Chen, University of Rochester; Pete Chen; Nhan Chu; Doug Clark, Princeton University; Bob Cmelik; John Crawford; Zarka Cvetanovic; Mike Dahlin, University of Texas, Austin; Merrick Darley; the staff of the DEC Western Research Laboratory; John DeRosa; Lloyd Dickman; J. Ding; Susan Eggers, University of Washington; Wael El-Essawy, University of Rochester; Patty Enriquez, Mills; Milos Ercegovac; Robert Garner; K. Gharachorloo, Compaq's Western Research Lab; Garth Gibson; Ronald Greenberg; Ben Hao; John Henning, Compaq; Mark Hill, University

of Wisconsin–Madison; Danny Hillis; David Hodges; Urs Hoelzle, Google; David Hough; Ed Hudson; Chris Hughes, University of Illinois at Urbana–Champaign; Mark Johnson; Lewis Jordan; Norm Jouppi; William Kahan; Randy Katz; Ed Kelly; Richard Kessler; Les Kohn; John Kowaleski, Compaq Computer Corp; Dan Lambright; Gary Lauterbach, Sun Microsystems; Corinna Lee; Ruby Lee; Don Lewine; Chao-Huang Lin; Paul Losleben, Defense Advanced Research Projects Agency; Yung-Hsiang Lu; Bob Lucas, Defense Advanced Research Projects Agency; Ken Lutz; Alan Mainwaring, Intel Berkeley Research Labs; Al Marston; Rich Martin, Rutgers; John Mashey; Luke McDowell; Sebastian Mirolo, Trimedia Corporation; Ravi Murthy; Biswadeep Nag; Lisa Noordergraaf, Sun Microsystems; Bob Parker, Defense Advanced Research Projects Agency; Vern Paxson, Center for Internet Research; Lawrence Prince; Steven Przybylski; Mark Pullen, Defense Advanced Research Projects Agency; Chris Rowen; Margaret Rowland; Greg Semeraro, University of Rochester; Bill Shannon; Behrooz Shirazi; Robert Shomler; Jim Slager; Mark Smotherman, Clemson University; the SMT research group at the University of Washington; Steve Squires, Defense Advanced Research Projects Agency; Ajay Sreekanth; Darren Staples; Charles Stapper; Jorge Stolfi; Peter Stoll; the students at Stanford and Berkeley who endured our first attempts at creating this book; Bob Supnik; Steve Swanson; Paul Taysom; Shreekant Thakkar; Alexander Thomasian, New Jersey Institute of Technology; John Toole, Defense Advanced Research Projects Agency; Kees A. Vissers, Trimedia Corporation; Willa Walker; David Weaver; Ric Wheeler, EMC; Maurice Wilkes; Richard Zimmerman.

John Hennessy ■ *David Patterson*