READINGS IN
Computer Architecture

EDITED BY
Mark D. Hill
Norman P. Jouppi
Gurindar S. Sohi

Readings in Computer Architecture

Mark D. Hill
University of Wisconsin-Madison

Norman P. Jouppi
Compaq Western Research Laboratory

Gurindar S. Sohi
University of Wisconsin-Madison

Morgan Kaufmann Publishers
San Francisco, California

Senior Editor Denise E. M. Penrose
Director of Production and Manufacturing Yonie Overton
Production Editor Sarah Burgundy
Assistant Editor Marilyn Uffner Alan
Editorial Coordinator Meghan Keeffe
Cover Design Eileen Wagner
Text Design, Composition, and Pasteup Susan M. Sheldrake, ShelDragon Graphic Design
Copyeditor Kathy Finch
Indexer Steve Rath
Printer Victor Graphics

Advice, Praise, and Errors: Any correspondence related to this publication or intended for the authors should be addressed to the Editorial and Sales Office of Morgan Kaufmann Publishers, Dept. CAR APE. Information regarding errors found in the original material is encouraged; electronic mail can be sent to carbugs@mkp.com. Please check the errata page at *http://www.mkp.com/architecture-readings* to see if the bug has already been reported.

Morgan Kaufmann Publishers
Editorial and Sales Office
340 Pine Street, Sixth Floor
San Francisco, CA 94104-3205
USA

Telephone	415/392-2665
Facsimile	415/982-2665
Email	mkp@mkp.com
WWW	*http://www.mkp.com*
Order toll free	800/745-7323

Printed in the United States of America

04 03 02 01 00 5 4 3 2 1

Library of Congress Cataloging-in-Publication Data
Hill, Mark D. (Mark Donald)
Readings in computer architecture / Mark D. Hill, Norman P. Jouppi, Gurindar Sohi.
p. cm.
Includes bibliographical references (p.
ISBN 1-55860-539-8
1. Computer architecture. I. Jouppi, Norman P. (Norman Paul) II. Sohi, Gurindar. III. Title
QA76.9.A73H55 2000
004.2'2—dc21 99-44480
CIP

Readings in Computer Architecture

To Sue, Nicole, and Gregory
To Lili, Mark, and Matthew
To Marilyn and Jacinth

Preface

In theory, theory and practice are the same.
In practice, they're different.
—Attribution unknown

We are pleased to present you with this collection of readings in computer architecture. Computer architecture can be considered the science and art of selecting and interconnecting hardware components to create a computer that meets functional, performance, and cost goals. Computer architecture has a rich history of practice whose mastery aids those interested in moving the field forward.

Computer architecture continues to flourish in a dynamic environment. One can think of it as an interface with hardware implementation technologies below and application and system software above. From below, exponential semiconductor advances continue to provide a greater number of faster transistors, while at the same time altering tradeoffs both on and between chips (e.g., on-chip wires can exceed gate delays). Architects use the additional transistors in new ways to make the compound growth of a computer system's capacity exceed the rate at which transistors are getting faster.

From above, the applications that use architectures change. Classic applications have evolved, while new application domains emerge as advances make computing cost-effective to new areas. Application areas include scientific computing (e.g., quantum dynamics to vehicle crash simulation), business data processing (e.g., accounting to data mining), personal productivity tools (e.g., word processing and spreadsheets), global connectivity (e.g., electronic mail to telecommuting), and emerging applications (e.g., personal digital assistants to virtual reality).

It is to the students, researchers, and practitioners of this dynamic field of computer architecture that we offer this new collection of readings. The rest of this preface discusses why we are introducing this reader now, how we select and introduce papers, and how one might use this reader.

Why Have a Computer Architecture Reader Now?

Why a reader now, when the last widely successful readers are decades old, there are good textbooks in the field, and the Web is expanding as a source for technical information? In our view, the last widely successful collection of readings in computer architecture was from Bell and Newell in 1971 [2] updated by Siewiorek in 1982 [6]. These readers were excellent, but much has happened since they were published, and they are currently out of print.

Another reader that complements this one is the recently published collection of selected papers from the first twenty-five years of the *International Symposium on Computer Architecture* (ISCA) edited by Sohi in 1998 [8]. Our reader differs from the ISCA reader in two important ways. First, we have drawn papers from many sources, not just from the ISCA. Second, we have contributed introductory material that puts multiple papers in context. In contrast, the ISCA reader has an introduction by each of the original author groups that describes how their paper came about, and, sometimes, their view on its future impact. We find these insights valuable and recommend the ISCA reader to you as well.

Why have a reader when there are good textbooks in the field? The answer to this question has deep roots in the history of science. A *primary source* is a work written by the people who did the work (discovery, invention, etc.) and when they did it. A *secondary source* is a work written after the primary source and usually by different authors. Textbooks are the vehicle usually used to introduce people to scientific or engineering disciplines. Textbooks are secondary sources designed for teaching. They summarize, synthesize, and interpret work in the intellectual model (*paradigm*) that prevails when they are written. They are indispensable for giving people a reasonably recent snapshot of the state of the art. Noteworthy textbooks in computer architecture include Almasi and Gottlieb [1], Culler, Singh, and Gupta [3], Hennessy and Patterson [4], and Stone [9].

Students and professionals who wish to contribute to advancing a field, however, must eventually move beyond textbooks to primary sources. New primary sources provide the most recent thinking that will not appear in textbooks for one or more years. In computer architecture, the pressure on "bleeding-edge" primary sources is so great that conference papers have more impact on intellectual progress than do journal papers (which have a longer delay between submission and appearance).

This reader contains many old primary sources. We see three reasons why people should read both old and new primary sources. First, the experience of reading old primary sources trains people to read new ones. Second, it makes them aware of the process of discovering. Primary sources over time give a "motion picture" rather than a "snapshot," enabling people to see how the movie progresses and how they might continue moving the movie forward. Finally, it allows people to see ideas in the context of multiple paradigms instead of just within the current paradigm. Students in the early 1980s, for example, could take whole courses on computer architecture with no mention of out-of-order execution, which was previously invented but out of vogue. We encourage readers interested in how science advances to read the landmark primary source by Thomas Kuhn [5].

Why have a reader when people can just use the Web? The primary value of this reader is our editorial judgment for selecting papers (discussed in the following section) and background we provide in introducing them and placing them in context. This value is not diminished by the existence of the World Wide Web.

A secondary value of a reader is the coalescing into one hard copy the hard copies of many papers that are time consuming to obtain. This value remains present for the many pre-Web papers we have selected (and is enhanced as post-Web researchers are more reluctant to do the leg work to obtain hard copies the traditional way). This value is significantly diminished, however, for post-Web papers.

Our approach is to turn the Web into an advantage. This hard copy reader is supplemented with a Web component at URL:

http://www.mkp.com/architecture-readings.

This Web page is modeled after this hard copy reader. Its ten sections follow the ten chapters of this reader. Each section contains pointers to recent papers and some text to introduce them and put them in context. In particular, the papers represent the best of the state of the art (e.g., a new microprocessor paper). Furthermore, our Web component will be updated, at least, annually to make this reader a living document that responds more rapidly than is possible with hard copy editions. Thus, the Web allows us to dynamically extend the value of our editorial judgment in a way not practical before the Web.

What's in this Reader and Why?

This reader focuses on what we believe are and will continue to be the critical issues facing computer architects. These include issues in *technology, evaluation methods, instruction set design, instruction level parallelism, dataflow/multithreading, memory systems, input/output systems, single-instruction multiple data parallelism,* and *multiple-instruction multiple data parallelism.* The down side of our focus, however, is that many important areas of inquiry are not covered (e.g., computer-aided design and computer arithmetic).

After limiting our focus, we were still left with the challenging task of selecting fifty-odd papers to fill a single bound volume from thousands of computer architecture papers that have been published. (For this reason, we hope that you do not judge us too harshly if we left out a few of your favorite papers.) We have used several guidelines for selecting papers. We looked for papers that:

- are primary sources
- have lasting impact
- are readable
- are an illustrative case study
- contribute insight to the modern reader

We did not apply these guidelines rigidly, in part, because they can contradict one another (e.g., some primary sources are inscrutable to the modern reader). One consequence of our emphasis on primary sources is that we exclude most survey papers, including influential ones (e.g., Smith's cache survey [7]).

Furthermore, the papers selected (and our original material) underwent two rounds of thoughtful review and discussion with a dozen researchers from academia and industry. They—see the acknowledgments at the end of this preface—helped to refine the selected papers to even better reflect what the community considers important as well as what students should know. We, the editors, resolved conflicts in the advice we received and accept all responsibility for the final version.

We have organized this reader into ten chapters. Each chapter is introduced by some original material

that places the papers in context, introducing each and, in many cases, providing additional content pertinent to the chapter topic but beyond the scope of the included papers.

Chapter 1, Classic Machines: Technology, Implementation, and Economics; examines the foundation of computer architecture. This chapter begins a discussion of classic machines from the ENIAC to Cray's machines with an emphasis on how implementation technologies influenced architecture. It then discusses Gordon Moore's amazing technology predictions from the 1960s (e.g., Moore's Law) and how they lead to microprocessor-based computers. Included papers discuss IBM 360, CDC 6600, Cray 1, Moore's predictions, and early Intel microprocessors.

Chapter 2, Methods, is not about computer architecture, per se, but about some of the methods used to evaluate and refine architectures. Without these methods, the progress we have come to expect would not be possible. This chapter discusses the scientific method and the three major classes of methods that computer architects use: analytic modeling, simulation, and system monitoring. Techniques are illustrated with case studies from memory hierarchy evaluation. Included papers present Amdahl's Law, a system-monitoring study of the VAX-11/780, and trace-driven simulation algorithms of evaluating alternative caches.

Chapter 3, Instruction Sets, discusses the interface between software and hardware that is so critical, in that it enables software to run on multiple generations of hardware. This chapter examines how changes in implementation technology and compiler technology have changed instruction set tradeoffs, including putting the complex instruction set computer (CISC) versus reduced instruction set computer (RISC) debate of the 1980s into perspective. Included papers discuss instruction sets as compiler targets, RISC, CISC, the most widely used instruction set (Intel 80386), and ideas on predication pertinent to emerging instruction sets such as IA-64.

Chapter 4, Instruction Level Parallelism (ILP), examines issues critical to accelerating processor execution rates beyond the increases provided by faster transistors and bit-level parallelism. This chapter examines issues for executing multiple instructions in parallel, which include hazards, precise exceptions, speculative execution, branch prediction, and explicitly parallel architectures, such as very long instruction word (VLIW). Included papers discuss the IBM 360/91, IBM RS/6000, MIPS R10000, branch prediction, precise exceptions, out-of-order execution, and a survey of instruction level parallelism issues.

Chapter 5, Dataflow and Multithreading, discusses these two approaches to increasing parallelism and tolerating latency. Dataflow has had considerable intellectual impact, and multithreading appears poised to significantly impact practice. This chapter explains how classic dataflow pushes limits by eliminating the program counter, whereas multithreading more conventionally (and more practically) uses multiple program counters. Included papers discuss foundational dataflow work, the tagged-token dataflow approach, an early multithreaded computer (HEP), and recent simultaneous multithreading ideas.

Chapter 6, Memory Systems, examines caches and virtual memory, two critical aspects of the memory hierarchy that can largely determine sustained computer performance. Because textbooks cover both concepts in detail, both discussions begin with early concepts and then examine selected more-recent concepts that are illustrated in the papers. Included are cache papers that discuss the first data cache proposals (Wilkes), the first commercial cache (IBM 380/85), nonblocking caches, snooping cache coherence, and victim caches/stream buffers. Included virtual memory papers present the first virtual memory system (Atlas), a 1980s virtual memory system from VAX-11/780, and a case study of some of the interactions between caches and virtual memory.

Chapter 7, I/O: Storage, Networks, and Graphics, examines systems needed to make computers interact with the outside world. In particular, this chapter discusses Input/Output (I/O) economics, presents a case study of personal computer (PC) I/O systems options, and introduces included papers. Included papers discuss historical I/O systems, disk modeling, redundant arrays of inexpensive disks (RAID), Ethernet local area network, routing in interconnection networks, and a case study of graphics support.

Chapter 8, Single Instruction Multiple Data (SIMD) Parallelism, discusses an approach to parallel computing where a single thread of control directs the manipulation of multiple data operations. SIMD's perceived utility has waxed and waned several times over the past decades. Even though interest in SIMD has currently waned, readers should be familiar with it because it is likely to wax again in the future. This chapter introduces basic SIMD concepts and included papers. Included papers discuss the original SIMD/MIMD taxonomy, a SIMD machine that issued

dependent operations (BSP), and massive processing in memory.

Chapter 9, Multiprocessors and Multicomputers, discusses computing systems in which multiple processors can operate independently. This is called *multiple instruction multiple data* (MIMD) parallelism. Such systems have long been the "future" of computing, but now they have finally earned substantial commercial success. This chapter discusses parallel software, shared-memory multiprocessors (processors joined via the memory system), multicomputers (processors joined via the I/O system), and selected future trends. Included shared-memory multiprocessor papers discuss an early prototype machine (CMU C.mmp), memory consistency models, cache coherence, a recent scalable example (Stanford DASH), and a *cache-only memory architecture* (COMA). Multicomputer papers discuss an early multicomputer (Caltech Cosmic Cube) and shared memory on multicomputers.

Finally, Chapter 10, Recent Implementations and Future Prospects, revisits some of the issues for modern machines that Chapter 1 raised for classic machines. Included papers discuss Intel Pentium, Intel Pentium Pro, and future microprocessor trends.

How to Use this Book

We have prepared this book with the hope that it will be valuable to both professionals and students. Both groups can read these primary sources and our original commentary to learn more about the rich history of computer architecture and to better see how to move the field forward.

For instructors, we see three beneficial ways to use our reader. First, it can be used as a supplement to a primary textbook. At the University of Wisconsin, for example, we have a primary graduate architecture course that focuses on uniprocessors. Most recent instructors have used Hennessy and Patterson [4] and a custom reader of papers. We anticipate some instructors will use this reader and some recent papers from the Web (at this reader's Web site and elsewhere) to replace their current custom reader. Wisconsin also has a secondary graduate architecture course that focuses on parallel processing. It has recently used Culler, Singh, and Gupta [3] and a custom reader. As above, one could replace the custom reader with this reader and supplementary Web papers.

A second approach fits schools having a first course that uses a textbook and a second course whose orientation toward projects or case studies favors a custom reader. Once again, this reader and Web papers could replace the custom reader.

A third model is for graduate courses that eschew textbooks and just use readers (as occurs for some offerings of Wisconsin's courses). This approach requires considerable skill and effort from faculty to tie together disjoint ideas. Using this reader and selected other papers, possibly Web only, will ease the instructor's burden, because we make considerable effort to tie our included papers together.

Acknowledgments

We wish to thank the many people that have contributed ideas, comments, and criticism to this reader, especially Arvind, MIT; Douglas Clark, Princeton; Matthew Farrens, UC-Davis; Josh Fisher, Hewlett-Packard; Kourosh Gharachoraloo, Compaq; James Goodman, University of Wisconsin-Madison; John Hennessy, Stanford; Wen-Mei Hwu, University of Illinois at Champaign-Urbana; Dave Nagle, Carnegie Mellon; Rishiyur Nikhil, Compaq; Yale Patt, University of Michigan; Constantine Polychronopoulos, University of Illinois at Champaign-Urbana; Todd Rockoff, Advantest; Balaram Sinharoy, IBM T. J. Watson Research Center; Jim Smith, University of Wisconsin-Madison; Mike Smith, Harvard; Mark Smotherman, Clemson; Dan Sorin, University of Wisconsin-Madison; Mateo Valero, Universitat Politcnica de Catalunya; David Wood, University of Wisconsin-Madison; and Cheng-Zhong Xu, Wayne State. Of course, all responsibility for what is said or included in this reader falls on us.

We also want thank the wonderful staff at Morgan Kaufmann for their talents, efforts, and encouragement, especially Denise Penrose, Marilyn Alan, Sarah Burgundy, and Meghan Keeffe.

References

[1] G. S. Almasi and A. Gottlieb. *Highly Parallel Computing*. Menlo Park, CA: Benjamin/Cummings, 1994.

[2] C. G. Bell and A. Newell. *Computer Structures: Readings and Examples*. New York: McGraw-Hill, 1971.

[3] D. Culler, J. P. Singh, and A. Gupta. *Parallel Computer Architecture: A Hardware/Software Approach*, San Francisco, CA: Morgan Kaufmann, 1998.

[4] J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 1996.

[5] T. S. Kuhn. *The Structure of Scientific Revolutions*, 2nd ed. Chicago, IL: Univ. of Chicago Press, 1970.

[6] D. P. Siewiorek, C. G. Bell, and A. Newell. *Computer Structures: Principles and Examples*. New York: McGraw-Hill, 1982.

[7] A. J. Smith. Cache Memories. *ACM Computing Surveys*, 14(3):473–530, 1982.

[8] G. S. Sohi. *25 Years of the International Symposia on Computer Architecture: Selected Papers*. New York, NY: ACM Press, 1998.

[9] H. S. Stone. *High-Performance Computer Architecture*, 3rd ed. Reading, MA: Addison-Wesley, 1993.

Included Papers

Chapter 1: Classic Machines: Technology, Implementation, and Economics

G. M. Amdahl, G. A. Blaauw, and F. P. Brooks, Jr., "Architecture of the IBM System/360," *IBM Journal of Research and Development*, Apr. 1964.

J. E. Thornton, "Parallel operation in the Control Data 6600," *Fall Joint Computers Conference*, vol. 26, pp. 33–40, 1961.

R. M. Russell, "The CRAY-1 computer system," *Communications of the ACM*, 21(1):63–72, 1978.

J. S. Kolodzey, "CRAY-1 computer technology," *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, CHMT-4(2), pp. 181–187, June 1981.

G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, pp. 114–117, Apr. 1965.

S. Mazor, "The history of the microcomputer—Invention and evolution," *Proceedings of the IEEE*, pp. 1601–1607, Dec. 1995.

Chapter 2: Methods

G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," *AFIPS Conference Proceedings*, pp. 483–485, Apr. 1967.

M. D. Hill and A. J. Smith, "Evaluating associativity in CPU caches," *IEEE Transactions on Computers*, C-38(12):1612–1630, 1989.

J. S. Emer and D. W. Clark, "A characterization of processor performance in the VAX-11/780," *Proceedings of the Eleventh International Symposium on Computer Architecture*, Ann Arbor, MI, pp. 301–310, June 1984.

Chapter 3: Instruction Sets

W. A. Wulf, "Compilers and computer architecture," *IEEE Computer*, 14(7):41–48, 1981.

G. Radin, "The 801 minicomputer," *Proceedings of the Symposium on Architectural Support for Programming Languages and Operating Systems*, pp. 39–47, Mar. 1982.

D. A. Patterson and D. R. Ditzel, "The case for the reduced instruction set computer," *ACM Computer Architecture News*, 8(6):25–33, 15 Oct. 1980.

R. P. Colwell, C. Y. Hitchcock III, E. D. Jensen, H. M. Brinkley Sprunt, and C. P. Kollar, "Instruction Sets and Beyond: Computers, complexity, and controversy," *IEEE Computer*, 18(9), 1985.

J. Crawford, "Architecture of the Intel 80386," *Proceedings of the ICCD*, pp. 155–160, Oct. 1986.

S. A. Mahlke, R. E. Hank, J. E. McCormick, D. I. August, and W. W. Hwu, "A comparison of full and partial predicated execution support for ILP processors," *Proceedings of the 22nd Annual Symposium on Computer Architecture* pp. 138–150, June 1995.

Chapter 4: Instruction Level Parallelism (ILP)

D. W. Anderson, F. J. Sparacio, and R. M. Tomasulo, "The IBM System/360 model 91: Machine philosophy and instruction-handling," *IBM Journal of Research and Development*, Jan. 1967.

J. E. Smith and A. R. Pleszkun, "Implementing precise interrupts in pipelined processors," *IEEE Transactions on Computers*, C-37(5):562–573, May 1988.

J. E. Smith, "A study of branch prediction strategies," *Proceedings of the Eighth Annual Symposium on Computer Architecture*, pp. 135–148, May 1981.

T.-Y. Yeh and Y. N. Patt, "Two-level adaptive training branch prediction," *Proceedings of the 24th Annual Workshop on Microprogramming (MICRO-24)*, Albuquerque, NM, pp. 55–60, Dec. 1991.

Y. N. Patt, W. W. Hwu, and M. Shebanow, "HPS, a new microarchitecture: Introduction and rationale," *Proceedings of the 18th Annual Workshop on Microprogramming*, Pacific Grove, CA, pp. 103–108, Dec. 1985.

G. S. Sohi and S. Vajapeyam, "Instruction issue logic for high-performance, interruptable pipelined processors," *Proceedings of the 14th Annual Symposium on Computer Architecture* pp. 27–34, June 1987.

G. F. Grohoski, "Machine organization of the IBM RISC System/6000 processor," *IBM Journal of Research and Development*, 34(1):37–58, 1990.

K. C. Yeager "The Mips R10000 superscalar microprocessor," *IEEE Micro*, 16(2):28–40, 1996.

B. R. Rau and J. A. Fisher, "Instruction-level parallel processing: History, overview, and perspective," *The Journal of Supercomputing*, 7(1):9–50, 1993. Reprinted in Rau and Fisher (eds.), *Instruction-Level Parallelism*, Norwell, MA: Kluwer, 1993.

Chapter 5: Dataflow and Multithreading

J. B. Dennis and D. P. Misunas, "A preliminary architecture for a basic data-flow processor," *Proceedings of the 2nd Annual Symposium on Computer Architecture, Computer Architecture News*, 3(4):126–132, 1974.

Arvind and R. S. Nikhil, "Executing a program on the MIT tagged-token dataflow architecture," *IEEE Transactions on Computers*, 39(3):300–318, 1990.

B. J. Smith, "Architecture and applications of the HEP mul-

tiprocessor computer system," *Proceedings of the International Society for Optical Engineering*, 241–248, 1981.

D. M. Tullsen, S. J. Eggers, J. S. Emer, H. M. Levy, J. L. Lo, and R. L. Stamm, "Exploiting choice: Instruction fetch and issue on an implementable simultaneous multithreading processor," *Proceedings of the 23rd Annual Symposium on Computer Architecture*, pp. 191–202, May 1996.

Chapter 6: Memory Systems

M. V. Wilkes, "Slave memories and dynamic storage allocation," *IEEE Transactions on Electronic Computers*, EC-14(2):270–271, 1965.

J. S. Liptay, "Structural aspects of the System/360 model 85, part II: The cache," *IBM Systems Journal*, 7(1):15–21, 1968.

D. Kroft, "Lockup-free instruction fetch/prefetch cache organization," *Proceedings of the Eighth Symposium on Computer Architecture* pp. 81–87, May 1981.

J. R. Goodman, "Using cache memory to reduce processor-memory traffic," *Proceedings of the Tenth International Symposium on Computer Architecture*, Stockholm, Sweden, pp. 124–131, June 1983.

N. P. Jouppi, "Improving direct-mapped cache performance by the addition of a small fully-associative cache and prefetch buffers," *Proceedings of the 17th Annual Symposium on Computer Architecture, Computer Architecture News*, 18(2):364–373, 1990

T. Kilburn, D. B. G. Edwards, M. J. Lanigan, and F. H. Sumner, "One-level storage system," *IRE Transactions*, EC-11(2):223–235, 1962.

D. W. Clark and J. S. Emer, "Performance of the VAX-11/780 translation buffer: Simulation and measurement," *ACM Transactions on Computer Systems*, 3(1):31–62, 1985.

W.-H. Wang, J.-L. Baer, and H. M. Levy, "Organization and performance of a two-level virtual-real cache hierarchy," *Proceedings of the 16th Annual International Symposium on Computer Architecture*, Jerusalem, pp. 140–148, June 1989.

Chapter 7: I/O: Storage Systems, Networks, and Graphics

M. Smotherman, "A sequencing-based taxonomy of I/O systems and review of historical machines," *ACM Computer Architecture News* 17(5):5–15, Sept. 1989.

Storage Systems

C. Ruemmler and J. Wilkes, "An introduction to disk drive modeling," *IEEE Computer* 27(3):17–28, 1994.

D. A. Patterson, G. Gibson, and R. H. Katz, "A case for redundant arrays of inexpensive disks (RAID)," *Proceedings of the ACM SIGMOD Conference*, Chicago, IL, June 1988.

Networks

R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks." *Communications of the ACM*, 19(7):395–404.

L. M. Ni and P. K. McKinley, "A survey of wormhole routing techniques in direct networks," *IEEE Computer*, 26(2):62–76, 1993.

Graphics

K. Akeley, "Reality engine graphics," *SIGGRAPH '93 Proceedings*, pp. 109–116.

Chapter 8: Single-Instruction Multiple Data (SIMD) Parallelism

M. J. Flynn, " Very high-speed computing systems," *Proceedings of the IEEE* , vol. 54, no. 12, Dec. 1966.

D. J. Kuck and R. A. Stokes, "The Burroughs scientific processor (BSP)," *IEEE Transactions on Computers*, C-31(5):363–376, 1982.

M. Gokhale, B. Holmes, and K. Iobst, "Processing in memory: The Terasys massively parallel PIM array," *IEEE Computer*, 28(4):23–31, 1995.

Chapter 9: Multiprocessors and Multicomputers

W. A. Wulf and S. P. Harbison, "Reflections in a pool of processors/An experience report on C.mmp/Hydra," *Proceedings of the National Computer Conference (AFIPS)*, June 1978.

L. Lamport, "How to make a multiprocessor computer that correctly executes multiprocess programs," *IEEE Transactions on Computers*, C-28(9):690–691, 1979.

L. M. Censier and P. Feautrier, "A new solution to coherence problems in multicache systems," *IEEE Transactions on Computers*, C-27(12):1112–1118, 1978.

D. Lenoski, J. Laudon, K. Gharachorloo, W.-D. Weber, A. Gupta, J. Hennessy, M. Horowitz, and M. Lam, "The Stanford Dash multiprocessor," *IEEE Computer*, 25(3):63–79, 1992.

E. Hagersten, A. Landin, and S. Haridi "DDM—A cache-only memory architecture," *IEEE Computer*, 25(9):44–54, 1992.

C. L. Seitz, "The cosmic cube," *Communications of the ACM*, pp. 22–33, Jan. 1985.

K. Li and P. Hudak, "Memory coherence in shared virtual memory systems," *ACM Transactions on Computer Systems*, 7(4):321–359, 1989.

Chapter 10: Recent Implementations and Future Prospects

D. Alpert and D. Avnon, "Architecture of the Pentium microprocessor," *IEEE Micro*, 13(3):11–21, 1993.

D. Papworth, "Tuning the Pentium Pro microarchitecture," *IEEE Micro*, 16(2):8–15, 1996.

M. Slater, "The microprocessor today," *IEEE Micro*, 16(6):32–44, 1996.

A. Yu, "The future of microprocessors," *IEEE Micro*, 16(6):46–53, 1996.

Contents

CHAPTER 1

Classic Machines: Technology, Implementation, and Economics

1.1 Introduction

The technology for implementation of computing systems has varied widely over the last one hundred years. Computers and their architecture at any point in time have been a consequence of the capabilities and limitations of the technology of their day. In this section, we illustrate this principle with a brief tour of computer history from a technological standpoint. It is essential to have a good understanding of technology, the trend of its advancement, and its limitations in order to have a good understanding of future developments in computers and their architecture.

1.2 Early Computer Technology

During the first third of the twentieth century, technology was limited to mechanical and electromechanical calculators.[1] During the 1930s, several experimenters began building electronic calculating machines with vacuum tubes. From 1937 through 1942, Atanasoff and Berry at Iowa State University constructed a computer (the ABC) using binary circuits built with vacuum tubes and a capacitive drum memory. This machine had almost become fully operational when the demands of World War II caused its development to stop. The next significant vacuum-tube computer development was the ENIAC, which became operational in 1945 at the Moore School of Electrical Engineering of the University of Pennsylvania.

1.2.1 ENIAC Computer Technology

The ENIAC had over 18,000 vacuum tubes. Vacuum tubes require a heater element in order to displace electrons from their cathodes. This requires a significant amount of power. The heater element also has a tendency to burn out after a period of operation. Thus, based on the total power consumed by all the tubes and the expected time between tube failures, there is a maximum practical size of a vacuum tube computer. At the time of ENIAC, many vacuum tubes had lifetimes of only a few thousand hours. Worse yet, tube failures were spread over the lifetime of the tube as described by probability distributions. Given these handicaps, one would think it would be impossible to build a computer with 18,000 tubes given these characteristics.

[1] Information about these early calculators and computers up until 1965 can be found in the excellent reference of Williams [26].