WILLIAM STALLINGS

COMPUTER ORGANIZATION AND ARCHITECTURE

Designing for Performance

Pearson

Eleventh Edition

Computer Organization and Architecture

Designing for Performance

Eleventh Edition

Computer Organization and Architecture

Designing for Performance

Eleventh Edition

William Stallings

330 Hudson Street, New York, NY 10013

Senior Vice President Courseware Portfolio Management: *Marcia J. Horton*

Director, Portfolio Management: Engineering, Computer Science & Global Editions: *Julian Partridge*

Executive Portfolio Manager: *Tracy Johnson*

Portfolio Management Assistant: *Meghan Jacoby*

Managing Content Producer: *Scott Disanno*

Content Producer: *Amanda Brands*

R&P Manager: *Ben Ferrini*

Manufacturing Buyer, Higher Ed, Lake Side Communications, Inc. (LSC): *Maura Zaldivar-Garcia*

Inventory Manager: *Bruce Boundy*

Field Marketing Manager: *Demetrius Hall*

Product Marketing Manager: *Yvonne Vannatta*

Marketing Assistant: *Jon Bryant*

Cover Designer: *Black Horse Designs*

Cover Art: *Shuttersstock/Shimon Bar*

Full-Service Project Management: *Kabilan Selvakumar, SPi Global*

Printer/Binder: *LSC Communications, Inc.*

Library of Congress Cataloging-in-Publication Data

Names: Stallings, William, author.

Title: Computer organization and architecture : designing for performance / William Stallings.

Description: Eleventh edition. | Hoboken : Pearson Education, 2019. | Includes bibliographical references and index.

Identifiers: LCCN 0134997190 | ISBN 9780134997193

Subjects: LCSH: Computer organization. | Computer architecture.

Classification: LCC QA76.9.C643 S73 2018 | DDC 004.2/2—dc23 LC record available at **https://lccn.loc.gov/**

1 18

ISBN-10: 0-13-499719-0

ISBN-13: 978-0-13-499719-3

To Tricia my loving wife, the kindest and gentlest person

Contents

Preface xiii

About the Author xxii

Chapter 1 Basic Concepts and Computer Evolution 1
- **1.1 Organization and Architecture 2**
- **1.2 Structure and Function 3**
- **1.3 The IAS Computer 11**
- **1.4 Gates, Memory Cells, Chips, and Multichip Modules 17**
- **1.5 The Evolution of the Intel x86 Architecture 23**
- **1.6 Embedded Systems 24**
- **1.7 ARM Architecture 29**
- **1.8 Key Terms, Review Questions, and Problems 34**

Chapter 2 Performance Concepts 37
- **2.1 Designing for Performance 38**
- **2.2 Multicore, MICs, and GPGPUs 44**
- **2.3 Two Laws that Provide Insight: Ahmdahl's Law and Little's Law 45**
- **2.4 Basic Measures of Computer Performance 48**
- **2.5 Calculating the Mean 51**
- **2.6 Benchmarks and SPEC 59**
- **2.7 Key Terms, Review Questions, and Problems 66**

Chapter 3 A Top-Level View of Computer Function and Interconnection 72
- **3.1 Computer Components 73**
- **3.2 Computer Function 75**
- **3.3 Interconnection Structures 90**
- **3.4 Bus Interconnection 92**
- **3.5 Point-to-Point Interconnect 94**
- **3.6 PCI Express 99**
- **3.7 Key Terms, Review Questions, and Problems 107**

Chapter 4 The Memory Hierarchy: Locality and Performance 112
- **4.1 Principle of Locality 113**
- **4.2 Characteristics of Memory Systems 118**
- **4.3 The Memory Hierarchy 121**
- **4.4 Performance Modeling of a Multilevel Memory Hierarchy 128**
- **4.5 Key Terms, Review Questions, and Problems 135**

Chapter 5 Cache Memory 138
5.1 Cache Memory Principles 139
5.2 Elements of Cache Design 143
5.3 Intel x86 Cache Organization 165
5.4 The IBM z13 Cache Organization 168
5.5 Cache Performance Models 169
5.6 Key Terms, Review Questions, and Problems 173
Chapter 6 Internal Memory 177
6.1 Semiconductor Main Memory 178
6.2 Error Correction 187
6.3 DDR DRAM 192
6.4 eDRAM 197
6.5 Flash Memory 199
6.6 Newer Nonvolatile Solid-State Memory Technologies 202
6.7 Key Terms, Review Questions, and Problems 205
Chapter 7 External Memory 210
7.1 Magnetic Disk 211
7.2 RAID 221
7.3 Solid State Drives 231
7.4 Optical Memory 234
7.5 Magnetic Tape 240
7.6 Key Terms, Review Questions, and Problems 242
Chapter 8 Input/Output 245
8.1 External Devices 247
8.2 I/O Modules 249
8.3 Programmed I/O 252
8.4 Interrupt-Driven I/O 256
8.5 Direct Memory Access 265
8.6 Direct Cache Access 271
8.7 I/O Channels and Processors 278
8.8 External Interconnection Standards 280
8.9 IBM z13 I/O Structure 283
8.10 Key Terms, Review Questions, and Problems 287
Chapter 9 Operating System Support 291
9.1 Operating System Overview 292

9.2 Scheduling 303
9.3 Memory Management 309
9.4 Intel x86 Memory Management 320
9.5 ARM Memory Management 325
9.6 Key Terms, Review Questions, and Problems 330
Chapter 10 Number Systems 334
10.1 The Decimal System 335
10.2 Positional Number Systems 336
10.3 The Binary System 337
10.4 Converting Between Binary and Decimal 337
10.5 Hexadecimal Notation 340
10.6 Key Terms and Problems 342
Chapter 11 Computer Arithmetic 344
11.1 The Arithmetic and Logic Unit 345
11.2 Integer Representation 346
11.3 Integer Arithmetic 351
11.4 Floating-Point Representation 366
11.5 Floating-Point Arithmetic 374
11.6 Key Terms, Review Questions, and Problems 383
Chapter 12 Digital Logic 388
12.1 Boolean Algebra 389
12.2 Gates 394
12.3 Combinational Circuits 396
12.4 Sequential Circuits 414
12.5 Programmable Logic Devices 423
12.6 Key Terms and Problems 428
Chapter 13 Instruction Sets: Characteristics and Functions 432
13.1 Machine Instruction Characteristics 433
13.2 Types of Operands 440
13.3 Intel x86 and ARM Data Types 442
13.4 Types of Operations 445
13.5 Intel x86 and ARM Operation Types 458
13.6 Key Terms, Review Questions, and Problems 466
Appendix 13A Little-, Big-, and Bi-Endian 472
Chapter 14 Instruction Sets: Addressing Modes and Formats 476

14.1 Addressing Modes 477
14.2 x86 and ARM Addressing Modes 483
14.3 Instruction Formats 489
14.4 x86 and ARM Instruction Formats 497
14.5 Key Terms, Review Questions, and Problems 502
Chapter 15 Assembly Language and Related Topics 506
15.1 Assembly Language Concepts 507
15.2 Motivation for Assembly Language Programming 510
15.3 Assembly Language Elements 512
15.4 Examples 518
15.5 Types of Assemblers 523
15.6 Assemblers 523
15.7 Loading and Linking 526
15.8 Key Terms, Review Questions, and Problems 533
Chapter 16 Processor Structure and Function 537
16.1 Processor Organization 538
16.2 Register Organization 539
16.3 Instruction Cycle 545
16.4 Instruction Pipelining 548
16.5 Processor Organization for Pipelining 566
16.6 The x86 Processor Family 568
16.7 The ARM Processor 575
16.8 Key Terms, Review Questions, and Problems 581
Chapter 17 Reduced Instruction Set Computers 586
17.1 Instruction Execution Characteristics 588
17.2 The Use of a Large Register File 593
17.3 Compiler-Based Register Optimization 598
17.4 Reduced Instruction Set Architecture 600
17.5 RISC Pipelining 606
17.6 MIPS R4000 610
17.7 SPARC 616
17.8 Processor Organization for Pipelining 621
17.9 CISC, RISC, and Contemporary Systems 623
17.10 Key Terms, Review Questions, and Problems 625
Chapter 18 Instruction-Level Parallelism and Superscalar Processors 629

18.1 Overview 630
18.2 Design Issues 637
18.3 Intel Core Microarchitecture 646
18.4 ARM Cortex-A8 652
18.5 ARM Cortex-M3 658
18.6 Key Terms, Review Questions, and Problems 663
Chapter 19 Control Unit Operation and Microprogrammed Control 669
19.1 Micro-operations 670
19.2 Control of the Processor 676
19.3 Hardwired Implementation 686
19.4 Microprogrammed Control 689
19.5 Key Terms, Review Questions, and Problems 698
Chapter 20 Parallel Processing 701
20.1 Multiple Processors Organization 703
20.2 Symmetric Multiprocessors 705
20.3 Cache Coherence and the MESI Protocol 709
20.4 Multithreading and Chip Multiprocessors 718
20.5 Clusters 723
20.6 Nonuniform Memory Access 726
20.7 Key Terms, Review Questions, and Problems 730
Chapter 21 Multicore Computers 736
21.1 Hardware Performance Issues 737
21.2 Software Performance Issues 740
21.3 Multicore Organization 745
21.4 Heterogeneous Multicore Organization 747
21.5 Intel Core i7-5960X 756
21.6 ARM Cortex-A15 MPCore 757
21.7 IBM z13 Mainframe 762
21.8 Key Terms, Review Questions, and Problems 765
Appendix A System Buses 768
A.1 Bus Structure 769
A.2 Multiple-Bus Hierarchies 770
A.3 Elements of Bus Design 772
Appendix B Victim Cache Strategies 777
B.1 Victim Cache 778

B.2 Selective Victim Cache 780

Appendix C Interleaved Memory 782

Appendix D The International Reference Alphabet 785

Appendix E Stacks 788
E.1 Stacks 789

E.2 Stack Implementation 790

E.3 Expression Evaluation 791

Appendix F Recursive Procedures 795
F.1 Recursion 796

F.2 Activation Tree Representation 797

F.3 Stack Implementation 803

F.4 Recursion and Iteration 804

Appendix G Additional Instruction Pipeline Topics 807
G.1 Pipeline Reservation Tables 808

G.2 Reorder Buffers 815

G.3 Tomasulo's Algorithm 818

G.4 Scoreboarding 822

Glossary 826

References 835

Supplemental Materials

Index 844

Preface

What's New in the Eleventh Edition

Since the tenth edition of this book was published, the field has seen continued innovations and improvements. In this new edition, I try to capture these changes while maintaining a broad and comprehensive coverage of the entire field. To begin this process of revision, the tenth edition of this book was extensively reviewed by a number of professors who teach the subject and by professionals working in the field. The result is that, in many places, the narrative has been clarified and tightened, and illustrations have been improved.

Beyond these refinements to improve pedagogy and user-friendliness, there have been substantive changes throughout the book. Roughly the same chapter organization has been retained, but much of the material has been revised and new material has been added. The most noteworthy changes are as follows:

- **Multichip Modules:** A new discussion of MCMs, which are now widely used, has been added to **Chapter 1**.
- **SPEC benchmarks:** The treatment of SPEC in **Chapter 2** has been updated to cover the new SPEC CPU2017 benchmark suite.
- **Memory hierarchy:** A new chapter on memory hierarchy expands on material that was in the cache memory chapter, plus adds new material. The new **Chapter 4** includes:
  —Updated and expanded coverage of the principle of locality

  —Updated and expanded coverage of the memory hierarchy

  —A new treatment of performance modeling of data access in a memory hierarchy
- **Cache memory:** The cache memory chapter has been updated and revised. **Chapter 5** now includes:
  —Revised and expanded treatment of logical cache organization, including new figures, to improve clarity

  —New coverage of content-addressable memory

  —New coverage of write allocate and no write allocate policies

  —A new section on cache performance modeling.
- **Embedded DRAM:** **Chapter 6** on internal memory now includes a section on the increasingly popular eDRAM.
- **Advanced Format 4k sector hard drives:** **Chapter 7** on external memory now includes discussion of the now widely used 4k sector hard drive format.
- **Boolean algebra:** The discussion on Boolean algebra in **Chapter 12** has been expanded with new text, figures, and tables, to enhance understanding.
- **Assembly language:** The treatment of assembly language has been expanded to a full chapter, with more detail and more examples.
- **Pipeline organization:** The discussion on pipeline organization has been substantially expanded with new text and figures. The material is in new sections in **Chapters 16** (Processor Structure and Function), **17** (RISC), and **18** (Superscalar).
- **Cache coherence:** The discussion of the MESI cache coherence protocol in **Chapter 20** has been expanded with new text and figures.

Support of ACM/IEEE Computer Science and Computer Engineering Curricula

The book is intended for both an academic and a professional audience. As a textbook, it is intended as a one- or two-semester undergraduate course for computer science, computer engineering, and electrical engineering majors. This edition supports recommendations of the ACM/IEEE Computer Science Curricula 2013 (CS2013). CS2013 divides all course work into three categories: Core-Tier 1 (all topics should be included in the curriculum); Core-Tier-2 (all or almost all topics should be included); and Elective (desirable to provide breadth and depth). In the Architecture and Organization (AR) area, CS2013 includes five Tier-2 topics and three Elective topics, each of which has a number of subtopics. This text covers all eight topics listed by CS2013. **Table P.1** shows the support for the AR Knowledge Area provided in this textbook. This book also supports the ACM/IEEE Computer Engineering Curricula 2016 (CE2016). CE2016 defines a necessary body of knowledge for undergraduate computer engineering, divided into twelve knowledge areas. One of these areas is Computer Architecture and Organization (CE-CAO), consisting of ten core knowledge areas. This text covers all of the CE-CAO knowledge areas listed in CE2016. **Table P.2** shows the coverage.

Table P.1 Coverage of CS2013 Architecture and Organization (AR) Knowledge Area

IAS Knowledge Units	Topics	Textbook Coverage
Digital Logic and Digital Systems (Tier 2)	• Overview and history of computer architecture • Combinational vs. sequential logic/Field programmable gate arrays as a fundamental combinational sequential logic building block • Multiple representations/layers of interpretation (hardware is just another layer) • Physical constraints (gate delays, fan-in, fan-out, energy/power)	—**Chapter 1** —**Chapter 12**
Machine Level Representation of Data (Tier 2)	• Bits, bytes, and words • Numeric data representation and number bases • Fixed- and floating-point systems • Signed and twos-complement representations • Representation of non-numeric data (character codes, graphical data)	—**Chapter 10** —**Chapter 11**
Assembly Level Machine Organization (Tier 2)	• Basic organization of the von Neumann machine • Control unit; instruction fetch, decode, and execution • Instruction sets and types (data manipulation, control, I/O) • Assembly/machine language programming • Instruction formats • Addressing modes • Subroutine call and return mechanisms (cross-	—**Chapter 1** —**Chapter 8** —**Chapter 13** —**Chapter**

	reference PL/Language Translation and Execution) • I/O and interrupts • Shared memory multiprocessors/multicore organization • Introduction to SIMD vs. MIMD and the Flynn Taxonomy	14 —Chapter 15 —Chapter 19 —Chapter 20 —Chapter 21
Memory System Organization and Architecture (Tier 2)	• Storage systems and their technology • Memory hierarchy: temporal and spatial locality • Main memory organization and operations • Latency, cycle time, bandwidth, and interleaving • Cache memories (address mapping, block size, replacement and store policy) • Multiprocessor cache consistency/Using the memory system for inter-core synchronization/atomic memory operations • Virtual memory (page table, TLB) • Fault handling and reliability	—Chapter 4 —Chapter 5 —Chapter 6 —Chapter 7 —Chapter 9 —Chapter 20
Interfacing and Communication (Tier 2)	• I/O fundamentals: handshaking, buffering, programmed I/O, interrupt-driven I/O • Interrupt structures: vectored and prioritized, interrupt acknowledgment • External storage, physical organization, and drives • Buses: bus protocols, arbitration, direct-memory access (DMA) • RAID architectures	—Chapter 3 —Chapter 7 —Chapter 8
Functional Organization (Elective)	• Implementation of simple datapaths, including instruction pipelining, hazard detection, and resolution • Control unit: hardwired realization vs. microprogrammed realization	—Chapter 16 —Chapter

	• Instruction pipelining • Introduction to instruction-level parallelism (ILP)	**17** —**Chapter 18** —**Chapter 19**
Multiprocessing and Alternative Architectures (Elective)	• Example SIMD and MIMD instruction sets and architectures • Interconnection networks • Shared multiprocessor memory systems and memory consistency • Multiprocessor cache coherence	—**Chapter 20** —**Chapter 21**
Performance Enhancements (Elective)	• Superscalar architecture • Branch prediction, Speculative execution, Out-of-order execution • Prefetching • Vector processors and GPUs • Hardware support for multithreading • Scalability	—**Chapter 17** —**Chapter 18** —**Chapter 20**

Table P.2 Coverage of CE2016 Computer Architecture and Organization (AR) Knowledge Area

Knowledge Unit	Textbook Coverage
History and overview	**Chapter 1**—Basic Concepts and Computer Evolution
Relevant tools, standards and/or engineering constraints	**Chapter 3**—A Top-Level View of Computer Function and Interconnection
Instruction set architecture	**Chapter 13**—Instruction Sets: Characteristics and Functions **Chapter 14**—Instruction Sets: Addressing Modes and Formats **Chapter 15**—Assembly Language and Related Topics
Measuring performance	**Chapter 2**—Performance Concepts
Computer arithmetic	**Chapter 10**—Number Systems

	Chapter 11—Computer Arithmetic
Processor organization	**Chapter 16**—Processor Structure and Function **Chapter 17**—Reduced Instruction Set Computers (RISCs) **Chapter 18**—Instruction-Level Parallelism and Superscalar Processors **Chapter 19**—Control Unit Operation and Microprogrammed Control
Memory system organization and architectures	**Chapter 4**—The Memory Hierarchy: Locality and Performance **Chapter 5**—Cache Memory **Chapter 6**—Internal Memory Technology **Chapter 7**—External Memory
Input/Output interfacing and communication	**Chapter 8**—Input/Output
Peripheral subsystems	**Chapter 3**—A Top-Level View of Computer Function and Interconnection **Chapter 8**—Input/Output
Multi/Many-core architectures	**Chapter 21**—Multicore Computers
Distributed system architectures	**Chapter 20**—Parallel Processing

Objectives

This book is about the structure and function of computers. Its purpose is to present, as clearly and completely as possible, the nature and characteristics of modern-day computer systems.

This task is challenging for several reasons. First, there is a tremendous variety of products that can rightly claim the name of computer, from single-chip microprocessors costing a few dollars to supercomputers costing tens of millions of dollars. Variety is exhibited not only in cost but also in size, performance, and application. Second, the rapid pace of change that has always characterized computer technology continues with no letup. These changes cover all aspects of computer technology, from the underlying integrated circuit technology used to construct computer components

to the increasing use of parallel organization concepts in combining those components.

In spite of the variety and pace of change in the computer field, certain fundamental concepts apply consistently throughout. The application of these concepts depends on the current state of the technology and the price/performance objectives of the designer. The intent of this book is to provide a thorough discussion of the fundamentals of computer organization and architecture and to relate these to contemporary design issues.

The subtitle suggests the theme and the approach taken in this book. It has always been important to design computer systems to achieve high performance, but never has this requirement been stronger or more difficult to satisfy than today. All of the basic performance characteristics of computer systems, including processor speed, memory speed, memory capacity, and interconnection data rates, are increasing rapidly. Moreover, they are increasing at different rates. This makes it difficult to design a balanced system that maximizes the performance and utilization of all elements. Thus, computer design increasingly becomes a game of changing the structure or function in one area to compensate for a performance mismatch in another area. We will see this game played out in numerous design decisions throughout the book.

A computer system, like any system, consists of an interrelated set of components. The system is best characterized in terms of structure—the way in which components are interconnected, and function—the operation of the individual components. Furthermore, a computer's organization is hierarchical. Each major component can be further described by decomposing it into its major subcomponents and describing their structure and function. For clarity and ease of understanding, this hierarchical organization is described in this book from the top down:

- **Computer system:** Major components are processor, memory, I/O.
- **Processor:** Major components are control unit, registers, ALU, and instruction execution unit.
- **Control unit:** Provides control signals for the operation and coordination of all processor components. Traditionally, a microprogramming implementation has been used, in which major components are control memory, microinstruction sequencing logic, and registers. More recently, microprogramming has been less prominent but remains an important implementation technique.

The objective is to present the material in a fashion that keeps new material in a clear context. This should minimize the chance that the reader will get lost and should provide better motivation than a bottom-up approach.

Throughout the discussion, aspects of the system are viewed from the points of view of both architecture (those attributes of a system visible to a machine language programmer) and organization (the operational units and their interconnections that realize the architecture).

Example Systems

This text is intended to acquaint the reader with the design principles and implementation issues of contemporary operating systems. Accordingly, a purely conceptual or theoretical treatment would be inadequate. To illustrate the concepts and to tie them to real-world design choices that must be made, two processor families have been chosen as running examples:

- **Intel x86 architecture:** The x86 architecture is the most widely used for nonembedded computer systems. The x86 is essentially a complex instruction set computer (CISC) with some RISC features. Recent members of the x86 family make use of superscalar and multicore design principles. The evolution of features in the x86 architecture provides a unique case-study of the evolution of most of the design principles in computer architecture.
- **ARM:** The ARM architecture is arguably the most widely used embedded processor, used in cell phones, iPods, remote sensor equipment, and many other devices. The ARM is essentially a

reduced instruction set computer (RISC). Recent members of the ARM family make use of superscalar and multicore design principles.

Many, but by no means all, of the examples in this book are drawn from these two computer families. Numerous other systems, both contemporary and historical, provide examples of important computer architecture design features.

Plan of the Text

The book is organized into six parts:

- Introduction
- The computer system
- Arithmetic and logic
- Instruction sets and assembly language
- The central processing unit
- Parallel organization, including multicore

The book includes a number of pedagogic features, including the use of interactive simulations and numerous figures and tables to clarify the discussion. Each chapter includes a list of key words, review questions, and homework problems. The book also includes an extensive glossary, a list of frequently used acronyms, and a bibliography.

Instructor Support Materials

Support materials for instructors are available at the **Instructor Resource Center (IRC)** for this textbook, which can be reached through the publisher's Web site **www.pearson.com/stallings**. To gain access to the IRC, please contact your local Pearson sales representative via **www.pearson.com/replocator**. The IRC provides the following materials:

- **Projects manual:** Project resources including documents and portable software, plus suggested project assignments for all of the project categories listed subsequently in this Preface.
- **Solutions manual:** Solutions to end-of-chapter Review Questions and Problems.
- **PowerPoint slides:** A set of slides covering all chapters, suitable for use in lecturing.
- **PDF files:** Copies of all figures and tables from the book.
- **Test bank:** A chapter-by-chapter set of questions.
- **Sample syllabuses:** The text contains more material than can be conveniently covered in one semester. Accordingly, instructors are provided with several sample syllabuses that guide the use of the text within limited time. These samples are based on real-world experience by professors with the first edition.

Student Resources

For this new edition, a tremendous amount of original supporting material for students has been made available online. The **Companion Web Site**, at **www.pearson.com/stallings**, includes a list of relevant links organized by chapter and an errata sheet for the book. To aid the student in understanding the material, a separate set of homework problems with solutions are available at this site. Students can enhance their understanding of the material by working out the solutions to these problems and then checking their answers. The site also includes a number of documents and papers referenced throughout the text.

Projects and Other Student Exercises

For many instructors, an important component of a computer organization and architecture course is a project or set of projects by which the student gets hands-on experience to reinforce concepts from the text. This book provides an unparalleled degree of support for including a projects component in the course. The instructor's support materials available through the IRC not only includes guidance on how to assign and structure the projects but also includes a set of user's manuals for various project types plus specific assignments, all written especially for this book. Instructors can assign work in the following areas:

- **Interactive simulation assignments:** Described subsequently.
- **Research projects:** A series of research assignments that instruct the student to research a particular topic on the Internet and write a report.
- **Simulation projects:** The IRC provides support for the use of the two simulation packages: SimpleScalar can be used to explore computer organization and architecture design issues. SMPCache provides a powerful educational tool for examining cache design issues for symmetric multiprocessors.
- **Assembly language projects:** A simplified assembly language, CodeBlue, is used and assignments based on the popular Core Wars concept are provided.
- **Reading/report assignments:** A list of papers in the literature, one or more for each chapter, that can be assigned for the student to read and then write a short report.
- **Writing assignments:** A list of writing assignments to facilitate learning the material.
- **Test bank:** Includes T/F, multiple choice, and fill-in-the-blank questions and answers.

This diverse set of projects and other student exercises enables the instructor to use the book as one component in a rich and varied learning experience and to tailor a course plan to meet the specific needs of the instructor and students.

Interactive Simulations

An important feature in this edition is the incorporation of interactive simulations. These simulations provide a powerful tool for understanding the complex design features of a modern computer system. A total of 20 interactive simulations are used to illustrate key functions and algorithms in computer organization and architecture design. At the relevant point in the book, an icon indicates that a relevant interactive simulation is available online for student use. Because the animations enable the user to set initial conditions, they can serve as the basis for student assignments. The instructor's supplement includes a set of assignments, one for each of the animations. Each assignment includes several specific problems that can be assigned to students.

Acknowledgments

This new edition has benefited from review by a number of people, who gave generously of their time and expertise. The following professors provided a review of the entire book: Nikhil Bhargava (Indian Institute of Management, Delhi), James Gil de Lamadrid (Bowie State University, Computer Science Department), Debra Calliss (Computer Science and Engineering, Arizona State University), Mohammed Anwaruddin (Wentworth Institute of Technology, Dept. of Computer Science), Roger Kieckhafer (Michigan Technological University, Electrical & Computer Engineering), Paul Fortier (University of Massachusetts Darthmouth, Electrical and Computer Engineering), Yan Zhang (Department of Computer Science and Engineering, University of South Florida), Patricia Roden (University of North Alabama, Computer Science and Information Systems), Sanjeev Baskiyar (Auburn University, Computer Science and Software Engineering), and (Jayson Rock, University of Wisconsin-Milwaukee, Computer Science). I would especially like to thank Professor Roger Kieckhafer for permission to make use of some of the figures and performance models from his course lecture notes.

Thanks also to the many people who provided detailed technical reviews of one or more chapters: Rekai Gonzalez Alberquilla, Allen Baum, Jalil Boukhobza, Dmitry Bufistov, Humberto Calderón, Jesus Carretero, Ashkan Eghbal, Peter Glaskowsky, Ram Huggahalli, Chris Jesshope, Athanasios Kakarountas, Isil Oz, Mitchell Poplingher, Roger Shepherd, Jigar Savla, Karl Stevens, Siri Uppalapati, Dr. Sriram Vajapeyam, Kugan Vivekanandarajah, Pooria M. Yaghini, and Peter Zeno,

Professor Cindy Norris of Appalachian State University, Professor Bin Mu of the University of New Brunswick, and Professor Kenrick Mock of the University of Alaska kindly supplied homework problems.

Aswin Sreedhar of the University of Massachusetts developed the interactive simulation assignments.

Professor Miguel Angel Vega Rodriguez, Professor Dr. Juan Manuel Sánchez Pérez, and Professor Dr. Juan Antonio Gómez Pulido, all of University of Extremadura, Spain, prepared the SMPCache problems in the instructor's manual and authored the SMPCache User's Guide.

Todd Bezenek of the University of Wisconsin and James Stine of Lehigh University prepared the SimpleScalar problems in the instructor's manual, and Todd also authored the SimpleScalar User's Guide.

Finally, I would like to thank the many people responsible for the publication of the book, all of whom did their usual excellent job. This includes the staff at Pearson, particularly my editor Tracy Johnson, her assistant Meghan Jacoby, and project manager Bob Engelhardt. Thanks also to the marketing and sales staffs at Pearson, without whose efforts this book would not be in front of you.

About the Author

Dr. William Stallings

has authored 18 textbooks, and counting revised editions, over 70 books on computer security, computer networking, and computer architecture. In over 30 years in the field, he has been a technical contributor, technical manager, and an executive with several high-technology firms. Currently, he is an independent consultant whose clients have included computer and networking manufacturers and customers, software development firms, and leading-edge government research institutions. He has 13 times received the award for the best computer science textbook of the year from the Text and Academic Authors Association.

He created and maintains the Computer Science Student Resource Site at **ComputerScienceStudent.com**. This site provides documents and links on a variety of subjects of general interest to computer science students (and professionals). He is a member of the editorial board of *Cryptologia*, a scholarly journal devoted to all aspects of cryptology.

Dr. Stallings holds a PhD from MIT in computer science and a BS from Notre Dame in electrical engineering.

Acronyms

ACM	Association for Computing Machinery
ALU	Arithmetic Logic Unit
ANSI	American National Standards Institute
ASCII	American Standards Code for Information Interchange
BCD	Binary Coded Decimal
CD	Compact Disk
CD-ROM	Compact Disk Read-Only Memory
CISC	Complex Instruction Set Computer
CPU	Central Processing Unit
DRAM	Dynamic Random-Access Memory
DMA	Direct Memory Access
DVD	Digital Versatile Disk
EEPROM	Electrically Erasable Programmable Read-Only Memory
EPIC	Explicitly Parallel Instruction Computing
EPROM	Erasable Programmable Read-Only Memory
HLL	High-Level Language

I/O	Input/Output
IAR	Instruction Address Register
IC	Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
ILP	Instruction-Level Parallelism
IR	Instruction Register
LRU	Least Recently Used
LSI	Large-scale Integration
MAR	Memory Address Register
MBR	Memory Buffer Register
MESI	Modify-Exclusive-Shared-Invalid
MIC	Many Integrated Core
MMU	Memory Management Unit
MSI	Medium-Scale Integration
NUMA	Nonuniform Memory Access
OS	Operating System
PC	Program Counter

PCB	Process Control Block
PCI	Peripheral Component Interconnect
PROM	Programmable Read-Only Memory
PSW	Processor Status Word
RAID	Redundant Array of Independent Disks
RALU	Register/Arithmetic-Logic Unit
RAM	Random-Access Memory
RISC	Reduced Instruction Set Computer
ROM	Read-Only Memory
SCSI	Small Computer System Interface
SMP	Symmetric Multiprocessors
SRAM	Static Random-Access Memory
SSI	Small-Scale Integration
ULSI	Ultra Large-Scale Integration
VLIW	Very Long Instruction Word
VLSI	Very Large-Scale Integration

Part One Introduction

Chapter 1 Basic Concepts and Computer Evolution

1.1 Organization and Architecture

1.2 Structure and Function
- **Function**
- **Structure**

1.3 The IAS Computer

1.4 Gates, Memory Cells, Chips, and Multichip Modules
- **Gates and Memory Cells**
- **Transistors**
- **Microelectronic Chips**
- **Multichip Module**

1.5 The Evolution of the Intel x86 Architecture

1.6 Embedded Systems
- **The Internet of Things**
- **Embedded Operating Systems**
- **Application Processors versus Dedicated Processors**
- **Microprocessors versus Microcontrollers**
- **Embedded versus Deeply Embedded Systems**

1.7 ARM Architecture
- **ARM Evolution**
- **Instruction Set Architecture**
- **ARM Products**

1.8 Key Terms, Review Questions, and Problems

Learning Objectives

After studying this chapter, you should be able to:

- Explain the general functions and structure of a digital computer.
- Present an overview of the evolution of computer technology from early digital computers to the latest microprocessors.
- Present an overview of the evolution of the x86 architecture.
- Define embedded systems and list some of the requirements and constraints that various embedded systems must meet.

1.1 Organization and Architecture

In describing computers, a distinction is often made between *computer architecture* and *computer organization*. Although it is difficult to give precise definitions for these terms, a consensus exists about the general areas covered by each. For example, see [VRAN80], [SIEW82], and [BELL78a]; an interesting alternative view is presented in [REDD76].

Computer architecture refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. A term that is often used interchangeably with computer architecture is **instruction set architecture (ISA)**. The ISA defines instruction formats, instruction opcodes, registers, instruction and data memory; the effect of executed instructions on the registers and memory; and an algorithm for controlling instruction execution. **Computer organization** refers to the operational units and their interconnections that realize the architectural specifications. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O mechanisms, and techniques for addressing memory. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

For example, it is an architectural design issue whether a computer will have a multiply instruction. It is an organizational issue whether that instruction will be implemented by a special multiply unit or by a mechanism that makes repeated use of the add unit of the system. The organizational decision may be based on the anticipated frequency of use of the multiply instruction, the relative speed of the two approaches, and the cost and physical size of a special multiply unit.

Historically, and still today, the distinction between architecture and organization has been an important one. Many computer manufacturers offer a family of computer models, all with the same architecture but with differences in organization. Consequently, the different models in the family have different price and performance characteristics. Furthermore, a particular architecture may span many years and encompass a number of different computer models, its organization changing with changing technology. A prominent example of both these phenomena is the IBM System/370 architecture. This architecture was first introduced in 1970 and included a number of models. The customer with modest requirements could buy a cheaper, slower model and, if demand increased, later upgrade to a more expensive, faster model without having to abandon software that had already been developed. Over the years, IBM has introduced many new models with improved technology to replace older models, offering the customer greater speed, lower cost, or both. These newer models retained the same architecture so that the customer's software investment was protected. Remarkably, the System/370 architecture, with a few enhancements, has survived to this day as the architecture of IBM's mainframe product line.

In a class of computers called microcomputers, the relationship between architecture and organization is very close. Changes in technology not only influence organization but also result in the introduction of more powerful and more complex architectures. Generally, there is less of a requirement for generation-to-generation compatibility for these smaller machines. Thus, there is more interplay between organizational and architectural design decisions. An intriguing example of this is the reduced instruction set computer (RISC), which we examine in **Chapter 15**.

This book text examines both computer organization and computer architecture. The emphasis is perhaps more on the side of organization. However, because a computer organization must be designed to implement a particular architectural specification, a thorough treatment of organization requires a detailed examination of architecture as well.

1.2 Structure and Function

A computer is a complex system; contemporary computers contain millions of elementary electronic components. How, then, can one clearly describe them? The key is to recognize the hierarchical nature of most complex systems, including the computer [SIMO96]. A hierarchical system is a set of interrelated subsystems; each subsystem may, in turn, contain lower level subsystems, until we reach some lowest level of elementary subsystem.

The hierarchical nature of complex systems is essential to both their design and their description. The designer need only deal with a particular level of the system at a time. At each level, the system consists of a set of components and their interrelationships. The behavior at each level depends only on a simplified, abstracted characterization of the system at the next lower level. At each level, the designer is concerned with structure and function:

- **Structure:** The way in which the components are interrelated.
- **Function:** The operation of each individual component as part of the structure.

In terms of description, we have two choices: starting at the bottom and building up to a complete description, or beginning with a top view and decomposing the system into its subparts. Evidence from a number of fields suggests that the top-down approach is the clearest and most effective [WEIN75].

The approach taken in this book follows from this viewpoint. The computer system will be described from the top down. We begin with the major components of a computer, describing their structure and function, and proceed to successively lower layers of the hierarchy. The remainder of this section provides a very brief overview of this plan of attack.

Function

Both the structure and functioning of a computer are, in essence, simple. In general terms, there are only four basic functions that a computer can perform:

- **Data processing:** Data may take a wide variety of forms, and the range of processing requirements is broad. However, we shall see that there are only a few fundamental methods or types of data processing.
- **Data storage:** Even if the computer is processing data on the fly (i.e., data come in and get processed, and the results go out immediately), the computer must temporarily store at least those pieces of data that are being worked on at any given moment. Thus, there is at least a short-term data storage function. Equally important, the computer performs a long-term data storage function. Files of data are stored on the computer for subsequent retrieval and update.
- **Data movement:** The computer's operating environment consists of devices that serve as either sources or destinations of data. When data are received from or delivered to a device that is directly connected to the computer, the process is known as *input–output* (*I*/*O*), and the device is referred to as a *peripheral*. When data are moved over longer distances, to or from a remote device, the process is known as *data communications*.
- **Control:** Within the computer, a control unit manages the computer's resources and orchestrates the performance of its functional parts in response to instructions.

The preceding discussion may seem absurdly generalized. It is certainly possible, even at a top level of computer structure, to differentiate a variety of functions, but to quote [SIEW82]:

> There is remarkably little shaping of computer structure to fit the function to be performed. At the root of this lies the general-purpose nature of computers, in which all the functional specialization

> occurs at the time of programming and not at the time of design.

Structure

We now look in a general way at the internal structure of a computer. We begin with a traditional computer with a single processor that employs a microprogrammed control unit, then examine a typical multicore structure.

SIMPLE SINGLE-PROCESSOR COMPUTER

Figure 1.1 provides a hierarchical view of the internal structure of a traditional single-processor computer. There are four main structural components: