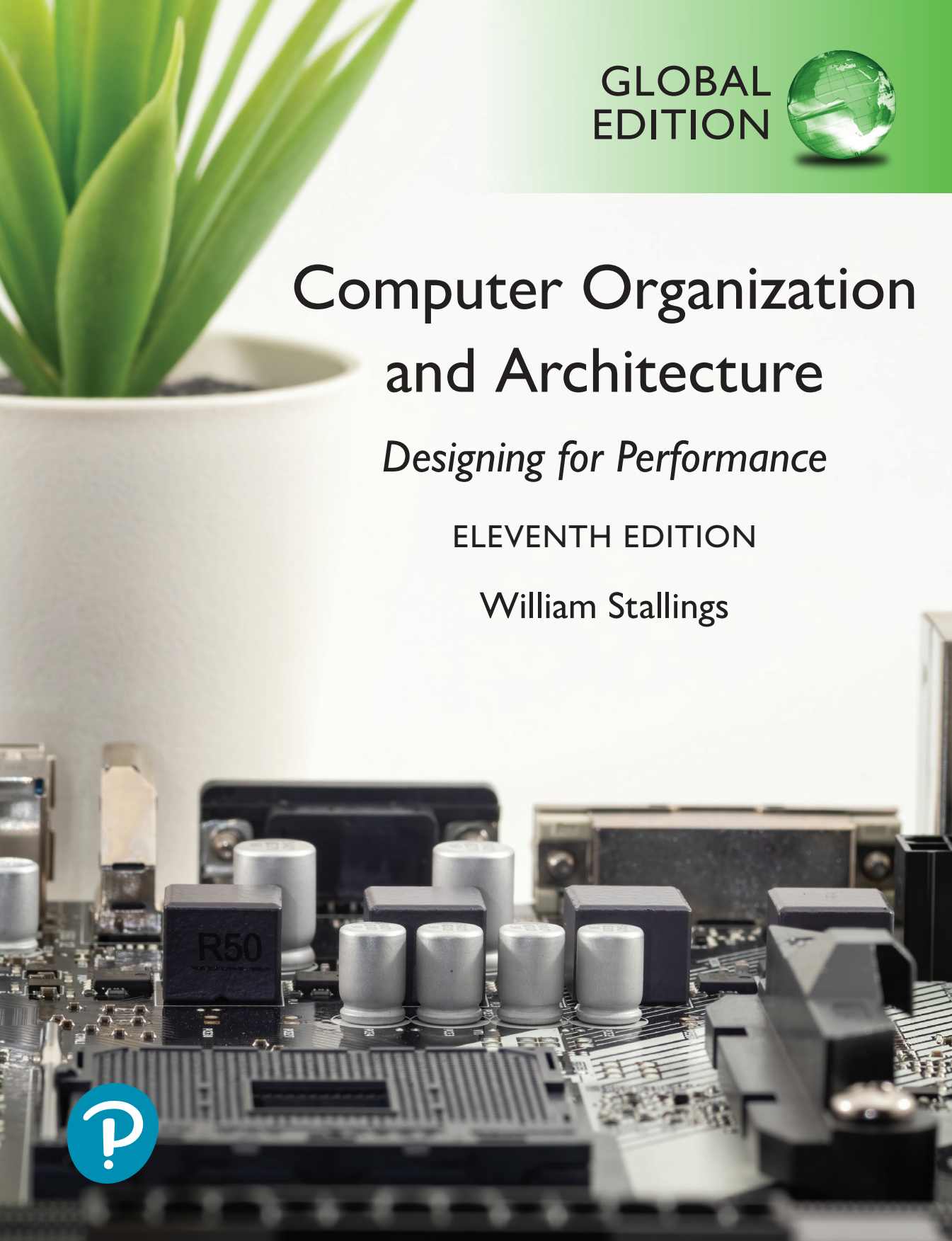
GLOBAL EDITION

Computer Organization and Architecture

Designing for Performance

ELEVENTH EDITION

William Stallings



Digital Resources for Students

Your eBook provides 12-month access to digital resources on the Companion Website. Refer to the preface in the textbook for a detailed list of resources.

Follow the instructions below to register for the Companion Website for William Stallings' *Computer Organization and Architecture*, Eleventh Edition, Global Edition.

1. Go to www.pearsonglobaleditions.com
2. Enter the title of your textbook or browse by author name.
3. Click Companion Website.
4. Click Register and follow the on-screen instructions to create a login name and password.

ISSSTY-FLAIL-EMERY-DUVET-BLUNT-RISES

Use a coin to scratch off the coating and reveal your access code.
Do not use a sharp knife or other sharp object as it may damage the code.

Use the login name and password you created during registration to start using the online resources that accompany your textbook.

IMPORTANT:

This access code can only be used once. This subscription is valid for 12 months upon activation and is not transferable.

For technical support, go to https://support.pearson.com/getsupport

Computer Organization and Architecture

Designing for Performance

Eleventh Edition

This page is intentionally left blank

Computer Organization and Architecture

Designing for Performance

Eleventh Edition

Global Edition

William Stallings

330 Hudson Street, New York, NY 10013

Cover Art: *zairiazmal/Shutterstock*

Pearson Education Limited
KAO Two
KAO Park
Hockham Way
Harlow
CM17 9SR
United Kingdom

and Associated Companies throughout the world

Visit us on the World Wide Web at: www.pearsonglobaleditions.com

Please contact https://support.pearson.com/getsupport/s/contactsupport with any queries on this content.

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

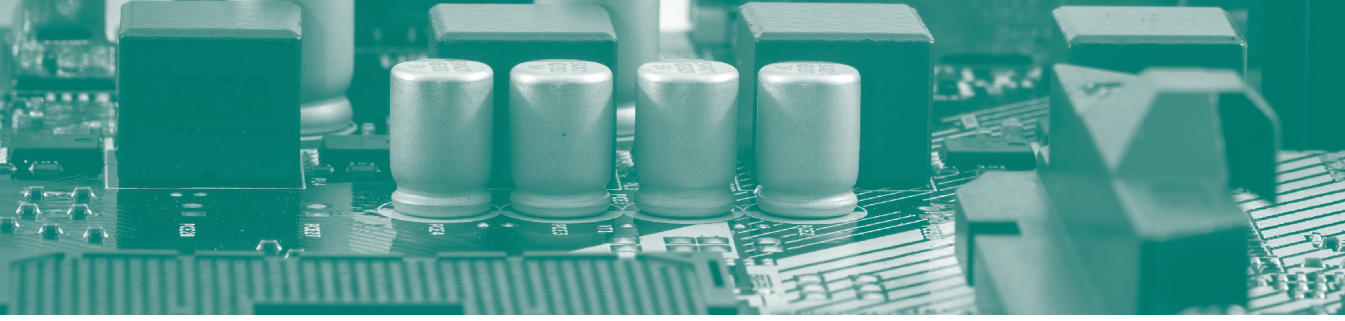
ISBN 10: 1-292-42010-3 (print)
ISBN 13: 978-1-292-42010-3 (print)
ISBN 13: 978-1-292-42008-0 (uPDF eBook)

To Tricia
my loving wife, the kindest
and gentlest person

This page is intentionally left blank

CONTENTS

This page is intentionally left blank



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

IAS Knowledge Units	Topics	Textbook Coverage
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-reference PL/Language Translation and Execution) • I/O and interrupts • Shared memory multiprocessors/multicore organization • Introduction to SIMD vs. MIMD and the Flynn Taxonomy	—Chapter 1 —Chapter 8 —Chapter 13 —Chapter 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 • Instruction pipelining • Introduction to instruction-level parallelism (ILP)	—Chapter 16 —Chapter 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.pearsonglobaleditions.com. 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.pearsonglobaleditions.com, includes a list of relevant links organized by chapter. 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. Students can subscribe to the Companion Web Site by using the access codes provided on the inside front cover.

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.

ACKNOWLEDGMENTS FOR THE GLOBAL EDITION

Pearson would like to acknowledge and thank the following for their work on the Global Edition.

Contributor

Asral Bahari Jambek (Universiti Malaysia Perlis, Malaysia)
Carl Barton (Birkbeck University of London)
Sutep Tongngam (NIDA, Thailand)

Reviewers

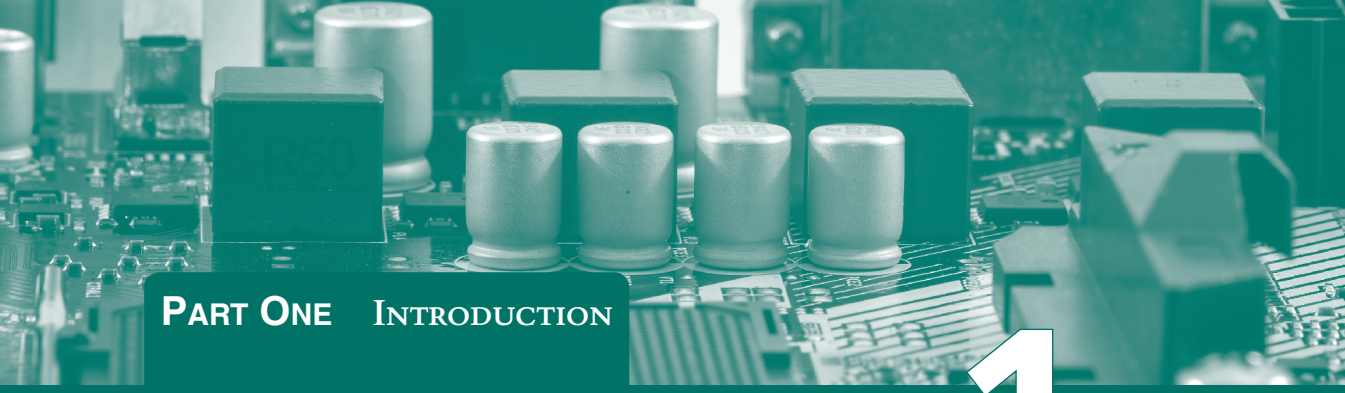
Rana Ejaz Ahmed (American University of Sharjah, UAE)
Ritesh Ajoodha (University of the Witwatersrand, Johannesburg)
Asral Bahari Jambek (Universiti Malaysia Perlis, Malaysia)
Patricia E.N. Lutu (University of Pretoria, South Africa)
Sezer Gören Uğurdağ (Yeditepe Üniversitesi, Turkey)

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.



PART ONE INTRODUCTION

CHAPTER 1

BASIC CONCEPTS AND COMPUTER EVOLUTION

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 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:

- **Central processing unit (CPU):** Controls the operation of the computer and performs its data processing functions; often simply referred to as **processor**.
- **Main memory:** Stores data.
- **I/O:** Moves data between the computer and its external environment.
- **System interconnection:** Some mechanism that provides for communication among CPU, main memory, and I/O. A common example of system

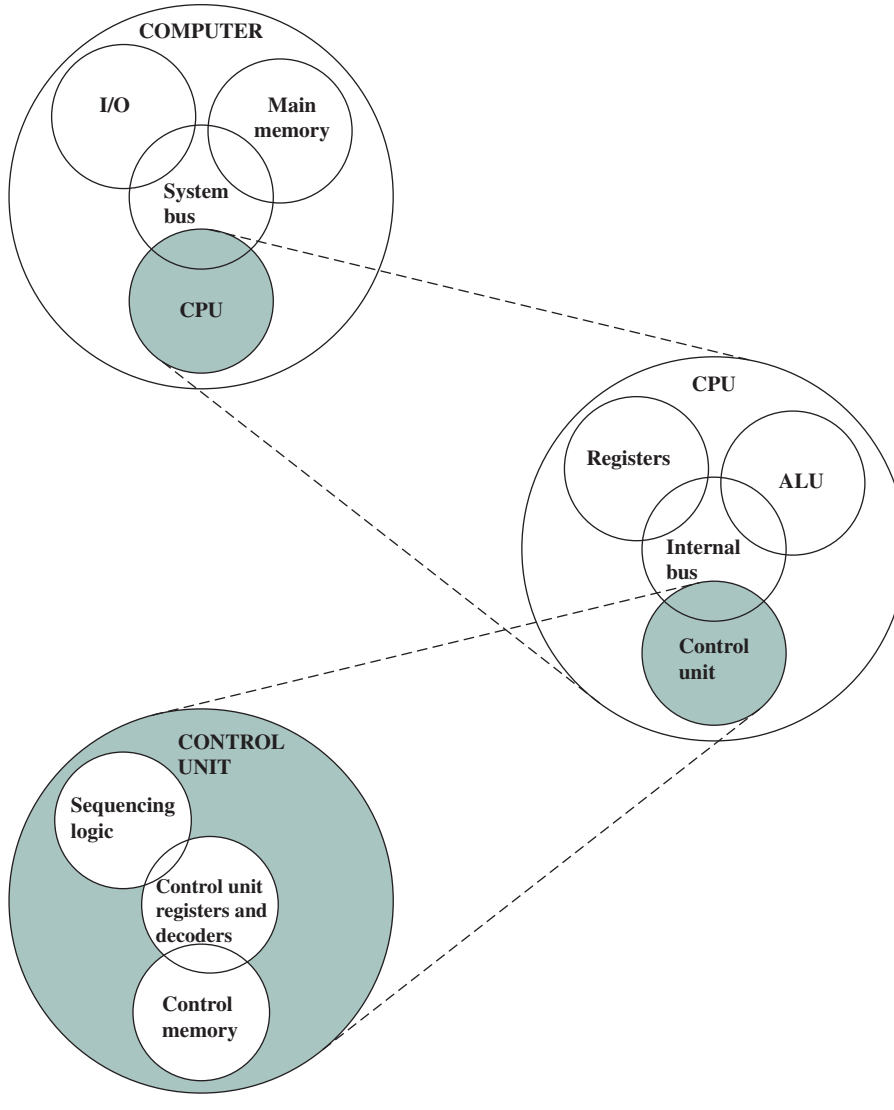


Figure 1.1 The Computer: Top-Level Structure

interconnection is by means of a **system bus**, consisting of a number of conducting wires to which all the other components attach.

There may be one or more of each of the aforementioned components. Traditionally, there has been just a single processor. In recent years, there has been increasing use of multiple processors in a single computer. Some design issues relating to multiple processors crop up and are discussed as the text proceeds; Part Five focuses on such computers.

Each of these components will be examined in some detail in Part Two. However, for our purposes, the most interesting and in some ways the most complex component is the CPU. Its major structural components are as follows:

- **Control unit:** Controls the operation of the CPU and hence the computer.
- **Arithmetic and logic unit (ALU):** Performs the computer's data processing functions.
- **Registers:** Provides storage internal to the CPU.
- **CPU interconnection:** Some mechanism that provides for communication among the control unit, ALU, and registers.

Part Three covers these components, where we will see that complexity is added by the use of parallel and pipelined organizational techniques. Finally, there are several approaches to the implementation of the control unit; one common approach is a *microprogrammed* implementation. In essence, a microprogrammed control unit operates by executing microinstructions that define the functionality of the control unit. With this approach, the structure of the control unit can be depicted, as in Figure 1.1. This structure is examined in Part Four.

MULTICORE COMPUTER STRUCTURE As was mentioned, contemporary computers generally have multiple processors. When these processors all reside on a single chip, the term *multicore computer* is used, and each processing unit (consisting of a control unit, ALU, registers, and perhaps cache) is called a *core*. To clarify the terminology, this text will use the following definitions.

- **Central processing unit (CPU):** That portion of a computer that fetches and executes instructions. It consists of an ALU, a control unit, and registers. In a system with a single processing unit, it is often simply referred to as a *processor*.
- **Core:** An individual processing unit on a processor chip. A core may be equivalent in functionality to a CPU on a single-CPU system. Other specialized processing units, such as one optimized for vector and matrix operations, are also referred to as cores.
- **Processor:** A physical piece of silicon containing one or more cores. The processor is the computer component that interprets and executes instructions. If a processor contains multiple cores, it is referred to as a **multicore processor**.

After about a decade of discussion, there is broad industry consensus on this usage.

Another prominent feature of contemporary computers is the use of multiple layers of memory, called *cache memory*, between the processor and main memory.

Chapter 4 is devoted to the topic of cache memory. For our purposes in this section, we simply note that a cache memory is smaller and faster than main memory and is used to speed up memory access, by placing in the cache data from main memory, that is likely to be used in the near future. A greater performance improvement may be obtained by using multiple levels of cache, with level 1 (L1) closest to the core and additional levels (L2, L3, and so on) progressively farther from the core. In this scheme, level n is smaller and faster than level $n + 1$.

Figure 1.2 is a simplified view of the principal components of a typical multicore computer. Most computers, including embedded computers in smartphones and tablets, plus personal computers, laptops, and workstations, are housed on a motherboard. Before describing this arrangement, we need to define some terms. A **printed circuit board (PCB)** is a rigid, flat board that holds and interconnects chips and other electronic components. The board is made of layers, typically two to ten, that interconnect components via copper pathways that are etched into

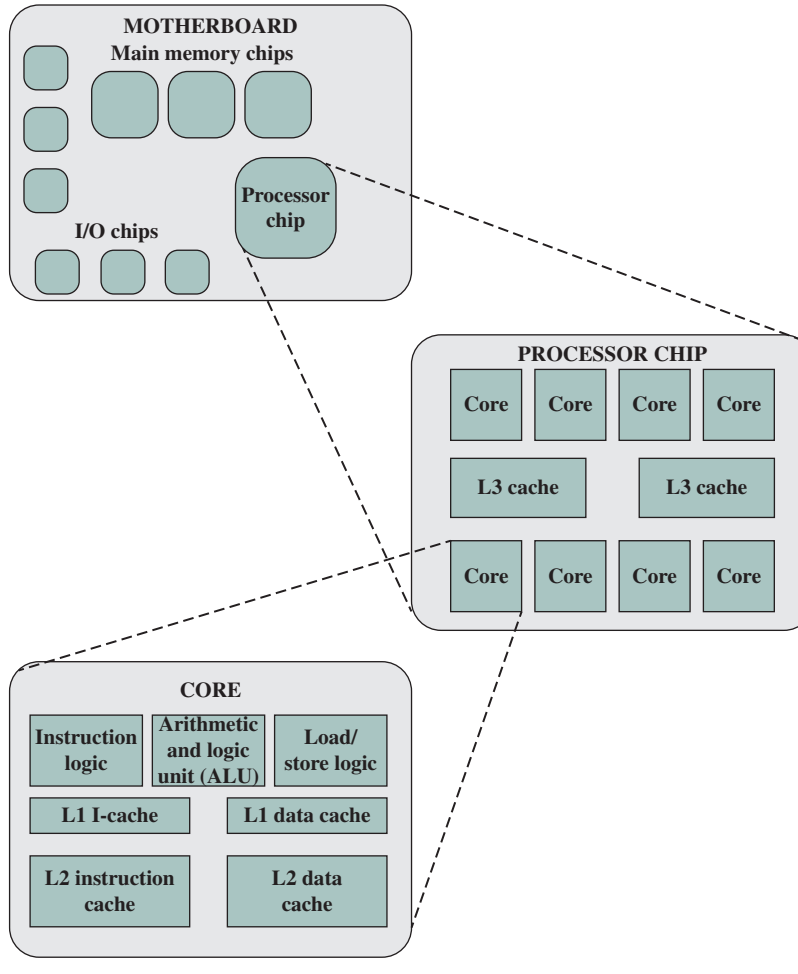


Figure 1.2 Simplified View of Major Elements of a Multicore Computer

the board. The main printed circuit board in a computer is called a system board or **motherboard**, while smaller ones that plug into the slots in the main board are called expansion boards.

The most prominent elements on the motherboard are the chips. A **chip** is a single piece of semiconducting material, typically silicon, upon which electronic circuits and logic gates are fabricated. The resulting product is referred to as an **integrated circuit**.

The motherboard contains a slot or socket for the processor chip, which typically contains multiple individual cores, in what is known as a *multicore processor*. There are also slots for memory chips, I/O controller chips, and other key computer components. For desktop computers, expansion slots enable the inclusion of more components on expansion boards. Thus, a modern motherboard connects only a few individual chip components, with each chip containing from a few thousand up to hundreds of millions of transistors.

Figure 1.2 shows a processor chip that contains eight cores and an L3 cache. Not shown is the logic required to control operations between the cores and the cache and between the cores and the external circuitry on the motherboard. The figure indicates that the L3 cache occupies two distinct portions of the chip surface. However, typically, all cores have access to the entire L3 cache via the aforementioned control circuits. The processor chip shown in Figure 1.2 does not represent any specific product, but provides a general idea of how such chips are laid out.

Next, we zoom in on the structure of a single core, which occupies a portion of the processor chip. In general terms, the functional elements of a core are:

- **Instruction logic:** This includes the tasks involved in fetching instructions, and decoding each instruction to determine the instruction operation and the memory locations of any operands.
- **Arithmetic and logic unit (ALU):** Performs the operation specified by an instruction.
- **Load/store logic:** Manages the transfer of data to and from main memory via cache.

The core also contains an L1 cache, split between an instruction cache (I-cache) that is used for the transfer of instructions to and from main memory, and an L1 data cache, for the transfer of operands and results. Typically, today's processor chips also include an L2 cache as part of the core. In many cases, this cache is also split between instruction and data caches, although a combined, single L2 cache is also used.

Keep in mind that this representation of the layout of the core is only intended to give a general idea of internal core structure. In a given product, the functional elements may not be laid out as the three distinct elements shown in Figure 1.2, especially if some or all of these functions are implemented as part of a microprogrammed control unit.

EXAMPLES It will be instructive to study some real-world examples that illustrate the hierarchical structure of computers. Let us take as an example the motherboard for a computer built around two Intel Quad-Core Xeon processor chips.