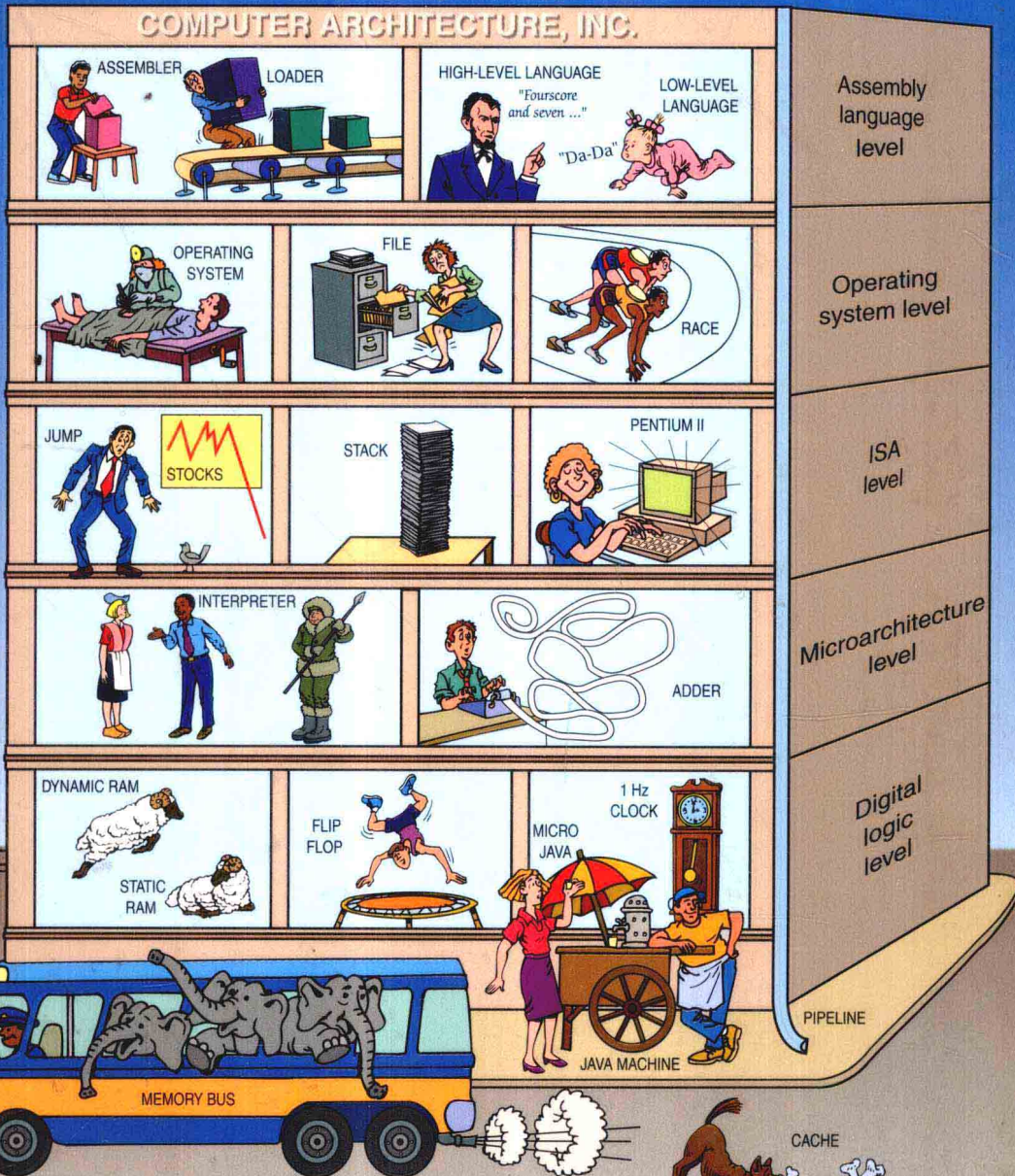


Fourth Edition

# STRUCTURED COMPUTER ORGANIZATION

Andrew S. Tanenbaum

UltraSPARC



# **STRUCTURED COMPUTER ORGANIZATION**

FOURTH EDITION

**ANDREW S. TANENBAUM**

*Vrije Universiteit  
Amsterdam, The Netherlands*

With contributions from  
**JAMES R. GOODMAN**

*University of Wisconsin  
Madison, WI*

**PRENTICE HALL**

UPPER SADDLE RIVER, NEW JERSEY 07458

Library of Congress Cataloging-in-Publication Data

Tanenbaum, Andrew S.

Structured computer organization /

Andrew S. Tanenbaum. — 4th ed.

p. cm.

Includes bibliographical references and index.

ISBN: 0-13-095990-1

1. Computer programming. 2. Computer organization. I. Title.

QA76.6.T38 1999

004.2'2—dc21

98-42130

CIP

Publisher: **ALAN APT**

Development editor: **SONDRA CHAVEZ**

Editor-in-chief: **MARCIA HORTON**

Production editor: **IRWIN ZUCKER**

Managing editor: **EILEEN CLARK**

Manufacturing buyer: **PAT BROWN**

Copy editor: **MARTHA WILLIAMS**

Director of production and manufacturing: **DAVID W. RICCARDI**

Composition and interior design: **ANDREW S. TANENBAUM**

Cover concept: **ANDREW S. TANENBAUM**

Cover director: **ANN FRANCE**

Cover illustrator: **DON MARTINETTI, DM GRAPHICS, INC.**

Editorial assistant: **TONI HOLM**

© 1999, 1990, 1984, 1976 by Prentice Hall, Inc.

Upper Saddle River, NJ 07458

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

The author and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

**TRADEMARK INFORMATION:** Windows NT is a registered trademark of Microsoft Corporation. UltraSPARC and Java are trademarks of Sun Microsystems, Inc.. Pentium II is a registered trademark of Intel Corporation. UNIX is a registered trademark of The Open Group. PostScript is a registered trademark of Adobe Systems Inc.

Printed in the United States of America

10 9 8 7 6 5 4 3 2

**ISBN 0-13-095990-1**

Prentice-Hall International (UK) Limited, *London*

Prentice-Hall of Australia Pty. Limited, *Sydney*

Prentice-Hall of Canada, Inc., *Toronto*

Prentice-Hall Hispanoamericana, S. A., *Mexico*

Prentice-Hall of India Private Limited, *New Delhi*

Prentice-Hall of Japan, Inc., *Tokyo*

Pearson Education Asia Pte. Ltd., *Singapore*

Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

*To Suzanne, Barbara, Marvin, Bram, and the memory of Sweetie  $\pi$*

# PREFACE

The first three editions of this book were based on the idea that a computer can be regarded as a hierarchy of levels, each one performing some well-defined function. This fundamental concept is as valid today as it was when the first edition came out, so it has been retained as the basis for the fourth edition. As in the first three editions, the digital logic level, the microarchitecture level, the instruction set architecture level, the operating system machine level, and the assembly language level are all discussed in detail (although we have changed some of the names to reflect modern practice).

Although the basic structure has been maintained, this fourth edition contains many changes, both small and large, that bring it up to date in the rapidly changing computer industry. For example, all the code examples, which were in Pascal, have been rewritten in Java, reflecting the popularity of Java in the computer world. Also, the example machines used have been brought up to date. The current examples are the Intel Pentium II, the Sun UltraSPARC II, and the Sun picoJava II, an embedded low-cost hardware Java chip.

Multiprocessors and parallel computers have also come in widespread use since the third edition, so the material on parallel architectures has been completely redone and greatly expanded, now covering a wide range of topics, from multiprocessors to COWs.

The book has become longer over the years (although still not as long as some other popular books on the subject). Such an expansion is inevitable as a subject develops and there is more known about it. As a result, when the book is used for a course, it may not always be possible to finish the book in a single course (e.g.,

in a trimester system). A possible approach would be to do all of Chaps. 1, 2, and 3, the first part of Chap. 4 (up through and including Sec. 4.4), and Chap. 5 as a bare minimum. The remaining time could be filled with the rest of Chap. 4, and parts of Chaps. 6, 7, and 8, depending on the interest of the instructor.

A chapter-by-chapter rundown of the major changes since the third edition follows. Chapter 1 still contains an historical overview of computer architecture, pointing out how we got where we are now and what the milestones were along the way. The enlarged spectrum of computers that exist is now discussed, and our three major examples (Pentium II, UltraSPARC II, and picoJava II) are introduced.

In Chapter 2, the material on input/output devices has been updated, emphasizing the technology of modern devices, including RAID disks, CD-Recordables, DVD, and color printers, among many others.

Chapter 3 (digital logic level) has undergone some revision and now treats computer buses and modern I/O chips. The major change here is additional material on buses, especially the PCI bus and the USB bus. The three new examples are described here at the chip level.

Chapter 4 (now called the microarchitecture level) has been completely rewritten. The idea of using a detailed example of a microprogrammed machine to illustrate the ideas of data path control has been retained, but the example machine is now a subset of the Java Virtual Machine. The underlying microarchitecture has been correspondingly changed. Several iterations of the design are given, showing what trade-offs are possible in terms of cost and performance. The last example, the Mic-4, uses a seven-stage pipeline and provides an easy introduction to how important modern computers, such as the Pentium II, work. A new section on improving performance has been added, focusing on the most recent techniques such as caching, branch prediction, (superscalar) out-of-order execution, speculative execution, and predication. The new example machines are discussed at the microarchitecture level.

Chapter 5 (now called the instruction set architecture level) deals with what many people refer to as “machine language.” The Pentium II, UltraSPARC II and Java Virtual Machine are used as the primary examples here.

Chapter 6 (operating system machine level) has examples for the Pentium II (Windows NT) and UltraSPARC II (UNIX). The former is new and has many features that are worth looking at, but UNIX is still a reliable workhorse at many universities and companies and is well worth examining in detail as well due to its simple and elegant design.

Chapter 7 (assembly language level) has been brought up to date by using examples from the machines we have been studying. New material on dynamic linking has been added as well.

Chapter 8 (parallel computer architectures) has been completely rewritten from the third edition. It now covers both multiprocessors (UMA, NUMA, and COMA) in detail, as well as multicomputers (MPP and COW).

The bibliography has been extensively revised and brought up to date. Well over two-thirds the references refer to works published after the third edition was published. Binary numbers and floating-point numbers have not undergone much change recently, so the appendices are largely the same as in the previous edition.

Finally, some problems have been revised and many new problems have been added since the third edition. Accordingly, a new problem solutions manual is available from Prentice Hall. It is available *only* to faculty members, who can request a free copy from their Prentice Hall representative.

A Web site for this book is available. PostScript files for all the illustrations used in the book are available electronically. They can be fetched and printed, for example, for making overhead sheets. In addition, a simulator and other and software tools are there too. The URL for this site is

<http://www.cs.vu.nl/~ast/sco4/>

The simulator and software tools were produced by Ray Ontko. The author wishes to express his gratitude to Ray for producing these extremely useful programs.

A number of people have read (parts of) the manuscript and provided useful suggestions or have been helpful in other ways. In particular, I would like to thank Henri Bal, Alan Charlesworth, Kourosh Gharachorloo, Marcus Goncalves, Karen Panetta Lentz, Timothy Mattson, Harlan McGhan, Miles Murdocca, Kevin Normoyle, Mike O'Connor, Mitsunori Ogihara, Ray Ontko, Aske Plaat, William Potvin II, Nagarajan Prabhakaran. James H. Pugsley, Ronald N. Schroeder, Ryan Shoemaker, Charles Silio, Jr., and Dale Skrien for their help, for which I am most grateful. My students, especially Adriaan Bon, Laura de Vries, Dolf Loth, Guido van 't Noordende, have also helped debug the text. Thank you.

I would especially like to thank Jim Goodman for his many contributions to this book, especially to Chaps. 4 and 5. The idea of using the Java Virtual Machine was his, as were the microarchitectures for implementing it. Many of the advanced ideas were due to him. The book is far better for his having put in so much effort.

Finally, I would like to thank Suzanne for her patience for my long hours in front of my Pentium. From my point of view the Pentium is a big improvement over my older 386 but from hers, it does not make much difference. I also want to thank Barbara and Marvin for being great kids and Bram for always being quiet when I was trying to write.

Andrew S. Tanenbaum

# CONTENTS

## PREFACE

xvi

## 1 INTRODUCTION

1

### 1.1 STRUCTURED COMPUTER ORGANIZATION 2

1.1.1 Languages, Levels, and Virtual Machines 2

1.1.2 Contemporary Multilevel Machines 4

1.1.3 Evolution of Multilevel Machines 8

### 1.2 MILESTONES IN COMPUTER ARCHITECTURE 13

1.2.1 The Zeroth Generation—Mechanical Computers (1642-1945) 13

1.2.2 The First Generation—Vacuum Tubes (1945-1955) 16

1.2.3 The Second Generation—Transistors (1955-1965) 19

1.2.4 The Third Generation—Integrated Circuits (1965-1980) 21

1.2.5 The Fourth Generation—Very Large Scale Integration (1980-?) 23

### 1.3 THE COMPUTER ZOO 24

1.3.1 Technological and Economic Forces 25

1.3.2 The Computer Spectrum 26

### 1.4 EXAMPLE COMPUTER FAMILIES 29

1.4.1 Introduction to the Pentium II 29

1.4.2 Introduction to the UltraSPARC II 31

1.4.3 Introduction to the picoJava II 34

### 1.5 OUTLINE OF THIS BOOK 36



**2 COMPUTER SYSTEMS ORGANIZATION****39**

- 2.1 PROCESSORS 39
  - 2.1.1 CPU Organization 40
  - 2.1.2 Instruction Execution 42
  - 2.1.3 RISC versus CISC 46
  - 2.1.4 Design Principles for Modern Computers 47
  - 2.1.5 Instruction-Level Parallelism 49
  - 2.1.6 Processor-Level Parallelism 53
- 2.2 PRIMARY MEMORY 56
  - 2.2.1 Bits 56
  - 2.2.2 Memory Addresses 57
  - 2.2.3 Byte Ordering 58
  - 2.2.4 Error-Correcting Codes 61
  - 2.2.5 Cache Memory 65
  - 2.2.6 Memory Packaging and Types 67
- 2.3 SECONDARY MEMORY 68
  - 2.3.1 Memory Hierarchies 69
  - 2.3.2 Magnetic Disks 70
  - 2.3.3 Floppy Disks 73
  - 2.3.4 IDE Disks 73
  - 2.3.5 SCSI Disks 75
  - 2.3.6 RAID 76
  - 2.3.7 CD-ROMs 80
  - 2.3.8 CD-Recordables 84
  - 2.3.9 CD-Rewritables 86
  - 2.3.10 DVD 86
- 2.4 INPUT/OUTPUT 89
  - 2.4.1 Buses 89
  - 2.4.2 Terminals 91
  - 2.4.3 Mice 99
  - 2.4.4 Printers 101
  - 2.4.5 Modems 106
  - 2.4.6 Character Codes 109
- 2.5 SUMMARY 113

## 3 THE DIGITAL LOGIC LEVEL

117

- 3.1 GATES AND BOOLEAN ALGEBRA 117
  - 3.1.1 Gates 118
  - 3.1.2 Boolean Algebra 120
  - 3.1.3 Implementation of Boolean Functions 122
  - 3.1.4 Circuit Equivalence 123
- 3.2 BASIC DIGITAL LOGIC CIRCUITS 128
  - 3.2.1 Integrated Circuits 128
  - 3.2.2 Combinational Circuits 129
  - 3.2.3 Arithmetic Circuits 134
  - 3.2.4 Clocks 139
- 3.3 MEMORY 141
  - 3.3.1 Latches 141
  - 3.3.2 Flip-Flops 143
  - 3.3.3 Registers 145
  - 3.3.4 Memory Organization 146
  - 3.3.5 Memory Chips 150
  - 3.3.6 RAMs and ROMs 152
- 3.4 CPU CHIPS AND BUSES 154
  - 3.4.1 CPU Chips 154
  - 3.4.2 Computer Buses 156
  - 3.4.3 Bus Width 159
  - 3.4.4 Bus Clocking 160
  - 3.4.5 Bus Arbitration 165
  - 3.4.6 Bus Operations 167
- 3.5 EXAMPLE CPU CHIPS 170
  - 3.5.1 The Pentium II 170
  - 3.5.2 The UltraSPARC II 176
  - 3.5.3 The picoJava II 179
- 3.6 EXAMPLE BUSES 181
  - 3.6.1 The ISA Bus 181
  - 3.6.2 The PCI Bus 183
  - 3.6.3 The Universal Serial Bus 189

- 3.7 INTERFACING 193
  - 3.7.1 I/O Chips 193
  - 3.7.2 Address Decoding 195

- 3.8 SUMMARY 198

## **4 THE MICROARCHITECTURE LEVEL 203**

- 4.1 AN EXAMPLE MICROARCHITECTURE 203
  - 4.1.1 The Data Path 204
  - 4.1.2 Microinstructions 211
  - 4.1.3 Microinstruction Control: The Mic-1 213
- 4.2 AN EXAMPLE ISA: IJVM 218
  - 4.2.1 Stacks 218
  - 4.2.2 The IJVM Memory Model 220
  - 4.2.3 The IJVM Instruction Set 222
  - 4.2.4 Compiling Java to IJVM 226
- 4.3 AN EXAMPLE IMPLEMENTATION 227
  - 4.3.1 Microinstructions and Notation 227
  - 4.3.2 Implementation of IJVM Using the Mic-1 232
- 4.4 DESIGN OF THE MICROARCHITECTURE LEVEL 243
  - 4.4.1 Speed versus Cost 243
  - 4.4.2 Reducing the Execution Path Length 245
  - 4.4.3 A Design with Prefetching: The Mic-2 253
  - 4.4.4 A Pipelined Design: The Mic-3 253
  - 4.4.5 A Seven-Stage Pipeline: The Mic-4 260
- 4.5 IMPROVING PERFORMANCE 264
  - 4.5.1 Cache Memory 265
  - 4.5.2 Branch Prediction 270
  - 4.5.3 Out-of-Order Execution and Register Renaming 276
  - 4.5.4 Speculative Execution 281
- 4.6 EXAMPLES OF THE MICROARCHITECTURE LEVEL 283
  - 4.6.1 The Microarchitecture of the Pentium II CPU 283
  - 4.6.2 The Microarchitecture of the UltraSPARC-II CPU 288
  - 4.6.3 The Microarchitecture of the picoJava II CPU 291
  - 4.6.4 A Comparison of the Pentium, UltraSPARC, and picoJava 296
- 4.7 SUMMARY 298

## **5 THE INSTRUCTION SET ARCHITECTURE LEVEL 303**

- 5.1 OVERVIEW OF THE ISA LEVEL 305
  - 5.1.1 Properties of the ISA Level 305
  - 5.1.2 Memory Models 307
  - 5.1.3 Registers 309
  - 5.1.4 Instructions 311
  - 5.1.5 Overview of the the Pentium II ISA Level 311
  - 5.1.6 Overview of the the UltraSPARC II ISA Level 313
  - 5.1.7 Overview of the Java Virtual Machine 317
- 5.2 DATA TYPES 318
  - 5.2.1 Numeric Data Types 319
  - 5.2.2 Nonnumeric Data Types 319
  - 5.2.3 Data Types on the Pentium II 320
  - 5.2.4 Data Types on the UltraSPARC II 321
  - 5.2.5 Data Types on the Java Virtual Machine 321
- 5.3 INSTRUCTION FORMATS 322
  - 5.3.1 Design Criteria for Instruction Formats 322
  - 5.3.2 Expanding Opcodes 325
  - 5.3.3 The Pentium II Instruction Formats 327
  - 5.3.4 The UltraSPARC II Instruction Formats 328
  - 5.3.5 The JVM Instruction Formats 330
- 5.4 ADDRESSING 332
  - 5.4.1 Addressing Modes 333
  - 5.4.2 Immediate Addressing 334
  - 5.4.3 Direct Addressing 334
  - 5.4.4 Register Addressing 334
  - 5.4.5 Register Indirect Addressing 335
  - 5.4.6 Indexed Addressing 336
  - 5.4.7 Based-Indexed Addressing 338
  - 5.4.8 Stack Addressing 338
  - 5.4.9 Addressing Modes for Branch Instructions 341
  - 5.4.10 Orthogonality of Opcodes and Addressing Modes 342
  - 5.4.11 The Pentium II Addressing Modes 344
  - 5.4.12 The UltraSPARC II Addressing Modes 346
  - 5.4.13 The JVM Addressing Modes 346
  - 5.4.14 Discussion of Addressing Modes 347
- 5.5 INSTRUCTION TYPES 348
  - 5.5.1 Data Movement Instructions 348

- 5.5.2 Dyadic Operations 349
- 5.5.3 Monadic Operations 350
- 5.5.4 Comparisons and Conditional Branches 352
- 5.5.5 Procedure Call Instructions 353
- 5.5.6 Loop Control 354
- 5.5.7 Input/Output 356
- 5.5.8 The Pentium II Instructions 359
- 5.5.9 The UltraSPARC II Instructions 362
- 5.5.10 The picoJava II Instructions 364
- 5.5.11 Comparison of Instruction Sets 369
- 5.6 FLOW OF CONTROL 370
  - 5.6.1 Sequential Flow of Control and Branches 371
  - 5.6.2 Procedures 372
  - 5.6.3 Coroutines 376
  - 5.6.4 Traps 379
  - 5.6.5 Interrupts 379
- 5.7 A DETAILED EXAMPLE: THE TOWERS OF HANOI 383
  - 5.7.1 The Towers of Hanoi in Pentium II Assembly Language 384
  - 5.7.2 The Towers of Hanoi in UltraSPARC II Assembly Language 384
  - 5.7.3 The Towers of Hanoi in JVM Assembly Language 386
- 5.8 THE INTEL IA-64 388
  - 5.8.1 The Problem with the Pentium II 390
  - 5.8.2 The IA-64 Model: Explicitly Parallel Instruction Computing 391
  - 5.8.3 Predication 393
  - 5.8.4 Speculative Loads 395
  - 5.8.5 Reality Check 396
- 5.9 SUMMARY 397

## **6 THE OPERATING SYSTEM MACHINE LEVEL 403**

- 6.1 VIRTUAL MEMORY 404
  - 6.1.1 Paging 405
  - 6.1.2 Implementation of Paging 407
  - 6.1.3 Demand Paging and the Working Set Model 409
  - 6.1.4 Page Replacement Policy 412
  - 6.1.5 Page Size and Fragmentation 414
  - 6.1.6 Segmentation 415
  - 6.1.7 Implementation of Segmentation 418

- 6.1.8 Virtual Memory on the Pentium II 421
- 6.1.9 Virtual Memory on the UltraSPARC 426
- 6.1.10 Virtual Memory and Caching 428
- 6.2 VIRTUAL I/O INSTRUCTIONS 429
  - 6.2.1 Files 430
  - 6.2.2 Implementation of Virtual I/O Instructions 431
  - 6.2.3 Directory Management Instructions 435
- 6.3 VIRTUAL INSTRUCTIONS FOR PARALLEL PROCESSING 436
  - 6.3.1 Process Creation 437
  - 6.3.2 Race Conditions 438
  - 6.3.3 Process Synchronization Using Semaphores 442
- 6.4 EXAMPLE OPERATING SYSTEMS 446
  - 6.4.1 Introduction 446
  - 6.4.2 Examples of Virtual Memory 455
  - 6.4.3 Examples of Virtual I/O 459
  - 6.4.4 Examples of Process Management 470
- 6.5 SUMMARY 476

## **7 THE ASSEMBLY LANGUAGE LEVEL 483**

- 7.1 INTRODUCTION TO ASSEMBLY LANGUAGE 484
  - 7.1.1 What Is an Assembly Language? 484
  - 7.1.2 Why Use Assembly Language? 485
  - 7.1.3 Format of an Assembly Language Statement 488
  - 7.1.4 Pseudoinstructions 491
- 7.2 MACROS 494
  - 7.2.1 Macro Definition, Call, and Expansion 494
  - 7.2.2 Macros with Parameters 496
  - 7.2.3 Advanced Features 497
  - 7.2.4 Implementation of a Macro Facility in an Assembler 498
- 7.3 THE ASSEMBLY PROCESS 498
  - 7.3.1 Two-Pass Assemblers 498
  - 7.3.2 Pass One 499
  - 7.3.3 Pass Two 502
  - 7.3.4 The Symbol Table 505

- 7.4 LINKING AND LOADING 506
  - 7.4.1 Tasks Performed by the Linker 508
  - 7.4.2 Structure of an Object Module 511
  - 7.4.3 Binding Time and Dynamic Relocation 512
  - 7.4.4 Dynamic Linking 515
- 7.5 SUMMARY 519

## **8 PARALLEL COMPUTER ARCHITECTURES 523**

- 8.1 DESIGN ISSUES FOR PARALLEL COMPUTERS 524
  - 8.1.1 Communication Models 526
  - 8.1.2 Interconnection Networks 530
  - 8.1.3 Performance 539
  - 8.1.4 Software 545
  - 8.1.5 Taxonomy of Parallel Computers 551
- 8.2 SIMD COMPUTERS 554
  - 8.2.1 Array Processors 554
  - 8.2.2 Vector Processors 555
- 8.3 SHARED-MEMORY MULTIPROCESSORS 559
  - 8.3.1 Memory Semantics 559
  - 8.3.2 UMA Bus-Based SMP Architectures 564
  - 8.3.3 UMA Multiprocessors Using Crossbar Switches 569
  - 8.3.4 UMA Multiprocessors Using Multistage Switching Networks 571
  - 8.3.5 NUMA Multiprocessors 573
  - 8.3.6 Cache Coherent NUMA Multiprocessors 575
  - 8.3.7 COMA Multiprocessors 585
- 8.4 MESSAGE-PASSING MULTICOMPUTERS 586
  - 8.4.1 MPPs—Massively Parallel Processors 587
  - 8.4.2 COWs—Clusters of Workstations 592
  - 8.4.3 Scheduling 593
  - 8.4.4 Communication Software for Multicomputers 598
  - 8.4.5 Application-Level Shared Memory 601
- 8.5 SUMMARY 609

## **9 READING LIST AND BIBLIOGRAPHY 613**

### **9.1 SUGGESTIONS FOR FURTHER READING 613**

- 9.1.1 Introduction and General Works 613
- 9.1.2 Computer Systems Organization 614
- 9.1.3 The Digital Logic Level 615
- 9.1.4 The Microarchitecture Level 616
- 9.1.5 The Instruction Set Architecture Level 617
- 9.1.6 The Operating System Machine Level 617
- 9.1.7 The Assembly Language Level 618
- 9.1.8 Parallel Computer Architectures 618
- 9.1.9 Binary and Floating-Point Numbers 620

### **9.2 ALPHABETICAL BIBLIOGRAPHY 620**

## **A BINARY NUMBERS 631**

- A.1 FINITE-PRECISION NUMBERS 631
- A.2 RADIX NUMBER SYSTEMS 633
- A.3 CONVERSION FROM ONE RADIX TO ANOTHER 635
- A.4 NEGATIVE BINARY NUMBERS 637
- A.5 BINARY ARITHMETIC 640

## **B FLOATING-POINT NUMBERS 643**

- B.1 PRINCIPLES OF FLOATING POINT 644
- B.2 IEEE FLOATING-POINT STANDARD 754 646

## **INDEX 653**



# 1

## INTRODUCTION

A digital computer is a machine that can solve problems for people by carrying out instructions given to it. A sequence of instructions describing how to perform a certain task is called a **program**. The electronic circuits of each computer can recognize and directly execute a limited set of simple instructions into which all its programs must be converted before they can be executed. These basic instructions are rarely much more complicated than

Add 2 numbers.

Check a number to see if it is zero.

Copy a piece of data from one part of the computer's memory to another.

Together, a computer's primitive instructions form a language in which it is possible for people to communicate with the computer. Such a language is called a **machine language**. The people designing a new computer must decide what instructions to include in its machine language. Usually, they try to make the primitive instructions as simple as possible, consistent with the computer's intended use and performance requirements, in order to reduce the complexity and cost of the electronics needed. Because most machine languages are so simple, it is difficult and tedious for people to use them.

This simple observation has, over the course of time, led to a way of structuring computers as a series of abstractions, each abstraction building on the one below it. In this way, the complexity can be mastered and computer systems can