

CONTENTS

Chapter 1

Welcome Aboard 1

- 1.1 What We Will Try To Do 1
- 1.2 How We Will Get There 2
- 1.3 A Computer System 3
- 1.4 Two Very Important Ideas 5
- 1.5 Computers as Universal Computational Devices 6
- 1.6 How Do We Get the Electrons to Do the Work? 8
 - 1.6.1 The Statement of the Problem 9
 - 1.6.2 The Algorithm 10
 - 1.6.3 The Program 10
 - 1.6.4 The ISA 11
 - 1.6.5 The Microarchitecture 12
 - 1.6.6 The Logic Circuit 12
 - 1.6.7 The Devices 13
 - 1.6.8 Putting It Together 13
- Problems 13

Chapter 2

Bits, Data Types, and Operations 17

- 2.1 Bits and Data Types 17
 - 2.1.1 The Bit as the Unit of Information 17
 - 2.1.2 Data Types 18
- 2.2 Integer Data Types 19
 - 2.2.1 Unsigned Integers 19
 - 2.2.2 Signed Integers 19
- 2.3 2's Complement Integers 21
- 2.4 Binary-Decimal Conversion 23
 - 2.4.1 Binary to Decimal Conversion 23
 - 2.4.2 Decimal to Binary Conversion 24
- 2.5 Operations on Bits—Part I: Arithmetic 25
 - 2.5.1 Addition and Subtraction 25
 - 2.5.2 Sign-extension 26
 - 2.5.3 Overflow 27

- 2.6 Operations on Bits—Part II: Logical Operations 29
 - 2.6.1 The AND Function 29
 - 2.6.2 The OR Function 30
 - 2.6.3 The NOT Function 30
 - 2.6.4 The Exclusive-OR Function 31
 - 2.6.5 Examples 31
- 2.7 Other Representations 32
 - 2.7.1 Floating Point Data Type 32
 - 2.7.2 ASCII Codes 36
 - 2.7.3 Hexadecimal Notation 36
- Problems 38

Chapter 3

Digital Logic Structures 47

- 3.1 The Transistor 47
- 3.2 Logic Gates 49
 - 3.2.1 The NOT Gate (or, Inverter) 49
 - 3.2.2 OR and NOR Gates 50
 - 3.2.3 AND and NAND Gates 52
 - 3.2.4 DeMorgan's Law 54
 - 3.2.5 Larger Gates 55
- 3.3 Combinational Logic Structures 55
 - 3.3.1 Decoder 56
 - 3.3.2 Mux 56
 - 3.3.3 Full Adder Circuit 57
 - 3.3.4 Logical Completeness 60
- 3.4 Basic Storage Elements 60
 - 3.4.1 The R-S Latch 61
 - 3.4.2 The Gated D Latch 62
 - 3.4.3 A Register 62
- 3.5 The Concept of Memory 64
 - 3.5.1 Address Space 64
 - 3.5.2 Addressability 64
 - 3.5.3 A 2²-by-3-bit Memory 65
- 3.6 The Data Path of the LC-2 67
 - Problems 68

Chapter 4

The Von Neumann Model 75

- 4.1 Basic Components 75
 - 4.1.1 Memory 75
 - 4.1.2 Processing Unit 77
 - 4.1.3 Input and Output 78
 - 4.1.4 Control Unit 78
 - 4.1.5 Summary: The LC-2 as an Example of the Von Neumann Model 79
- 4.2 Instruction Processing 79
 - 4.2.1 The Instruction 80
 - 4.2.2 The Instruction Cycle 82
 - 4.2.3 Examples 84
- 4.3 Changing the Sequence of Execution 84
- 4.4 Stopping the Computer 85
 - Problems 87

Chapter 5

The LC-2 91

- 5.1 The ISA: Overview 91
 - 5.1.1 Memory Organization 92
 - 5.1.2 Registers 92
 - 5.1.3 The Instruction Set 93
 - 5.1.4 Opcodes 93
 - 5.1.5 Data Types 93
 - 5.1.6 Addressing Modes 95
 - 5.1.7 Condition Codes 95
- 5.2 Operate Instructions 95
- 5.3 Data Movement Instructions 97
 - 5.3.1 Immediate Mode 98
 - 5.3.2 Direct Mode 98
 - 5.3.3 Indirect Mode 99
 - 5.3.4 Base+Offset Mode 99
 - 5.3.5 An Example 99
- 5.4 Control Instructions 101
 - 5.4.1 Conditional Branches 101
 - 5.4.2 An Example 102
 - 5.4.3 Two Methods for Loop Control 104
 - 5.4.4 Example: Adding a Column of Numbers Using a Sentinel 105
 - 5.4.5 The TRAP Instruction 106
- 5.5 Another Example: Counting Occurrences of a Character 107

5.6 The Data Path Revisited 110

- 5.6.1 Basic Components of the Data Path 110
- 5.6.2 The Instruction Cycle 113
 - Problems 114

Chapter 6

Programming 119

- 6.1 Problem Solving 119
 - 6.1.1 Systematic Decomposition 119
 - 6.1.2 The Three Constructs: Sequential, Conditional, Iterative 120
 - 6.1.3 LC-2 Control Instructions to Implement the Three Constructs 121
 - 6.1.4 The Character Count Example from Chapter 5, Revisited 123
- 6.2 Debugging 127
 - 6.2.1 Debugging Operations 127
 - 6.2.2 Examples: Use of the Interactive Debugger 128
 - Problems 135

Chapter 7

Assembly Language 139

- 7.1 Assembly Language Programming—Moving Up a Level 139
- 7.2 An Assembly Language Program 140
 - 7.2.1 Instructions 141
 - 7.2.2 Pseudo-ops (Assembler Directives) 144
 - 7.2.3 Example: The Character Count Example of Section 5.5, Revisited 145
- 7.3 The Assembly Process 147
 - 7.3.1 Introduction 147
 - 7.3.2 A Two-Pass Process 147
 - 7.3.3 The First Pass: Creating the Symbol Table 148
 - 7.3.4 The Second Pass: Generating the Machine Language Program 149
- 7.4 Beyond Assembly of a Single Assembly Language Program 149
 - 7.4.1 The Executable Image 150
 - 7.4.2 More than One Object File 151
 - Problems 151

Chapter 8

I/O 157

- 8.1 I/O Basics 157
 - 8.1.1 Device Registers 157
 - 8.1.2 Memory-Mapped I/O Versus Special Input/Output Instructions 158
 - 8.1.3 Asynchronous Versus Synchronous 158
 - 8.1.4 Interrupt-Driven Versus Polling 160
- 8.2 Input from the Keyboard 160
 - 8.2.1 Basic Input Registers (the KBDR and the KBSR) 160
 - 8.2.2 The Basic Input Service Routine 160
- 8.3 Output to the Monitor 162
 - 8.3.1 Basic Output Registers (the CRTDR and the CRTSR) 162
 - 8.3.2 The Basic Output Service Routine 162
 - 8.3.3 Example: Keyboard Echo 163
- 8.4 A More Sophisticated Input Routine 163
- 8.5 Interrupt-Driven I/O 165
 - 8.5.1 Element 1: The Interrupt Signal 166
 - 8.5.2 Element 2: The Test for Interrupts 167

Chapter 9

TRAP Routines and Subroutines 171

- 9.1 LC-2 TRAP Routines 171
 - 9.1.1 Introduction 171
 - 9.1.2 The TRAP Mechanism 172
 - 9.1.3 The TRAP Instruction 173
 - 9.1.4 The RET Instruction 174
 - 9.1.5 An Example 176
 - 9.1.6 TRAP Routines for Handling I/O 177
 - 9.1.7 TRAP Routine for Halting the Computer 179
 - 9.1.8 Saving and Restoring Registers 179
 - 9.2 Subroutine Calls>Returns 181
 - 9.2.1 The JSR/RET Mechanism 182
 - 9.2.2 The JSR and JSRR Instructions 183
 - 9.2.3 An Example 184
 - 9.2.4 Another Subroutine: Writing a Character String to the Monitor 186
 - 9.2.5 Library Routines 187
- Problems 192

Chapter 10

And, Finally . . . 195

- 10.1 The Stack—A Very Important Storage Structure 195
 - 10.1.1 The Stack—An Abstract Data Type 195
 - 10.1.2 Two Example Implementations 196
 - 10.1.3 Implementation in Memory 197
 - 10.1.4 The Complete Picture 201
- 10.2 Arithmetic Using a Stack 201
 - 10.2.1 The Stack as Temporary Storage 201
 - 10.2.2 An Example 203
 - 10.2.3 OpAdd, OpMult, and OpNeg 203
- 10.3 Data Type Conversion 210
 - 10.3.1 Example: The Bogus Program: $2 + 3 = e$ 210
 - 10.3.2 ASCII to Binary 211
 - 10.3.3 Binary to ASCII 214
- 10.4 Our Final Example: The Calculator 215
 - Problems 221

Chapter 11

Introduction to Programming in C 223

- 11.1 Our Objective 223
 - 11.2 Bridging the Gap 224
 - 11.3 Translating High-Level Language Programs 226
 - 11.3.1 Interpretation 226
 - 11.3.2 Compilation 227
 - 11.3.3 Pros and Cons 227
 - 11.4 The C Programming Language 227
 - 11.4.1 The C Compiler 228
 - 11.5 A Simple Example 230
 - 11.5.1 The Function `main` 232
 - 11.5.2 Formatting, Comments, and Style 233
 - 11.5.3 The C Preprocessor 233
 - 11.5.4 Input and Output 235
- Problems 238

Chapter 12

Variables and Operators 241

- 12.1 Introduction 241
- 12.2 Variables 241

- 12.2.1 Three Basic Data Types: **int**, **char**, **double** 243
- 12.2.2 Identifiers 246
- 12.2.3 Scope: Globals and Locals 247
- 12.2.4 More Examples 249
- 12.2.5 Symbol Table 250
- 12.2.6 Allocating Space for Variables 250
- 12.2.7 Literals, Constants, and Symbolic Values 253
- 12.3 Operators 254
 - 12.3.1 Expressions and Statements 255
 - 12.3.2 The Assignment Operator 256
 - 12.3.3 Arithmetic Operators 257
 - 12.3.4 Precedence and Associativity 258
 - 12.3.5 Bitwise Operators 259
 - 12.3.6 Logical Operators 260
 - 12.3.7 Relational Operators 262
 - 12.3.8 A Simple Example 263
 - 12.3.9 C's Special Operators 263
 - 12.3.10 Tying It Together 267
- 12.4 A More Comprehensive Example 268
- 12.5 Summary 270
 - Problems 271

Chapter 13

Control Structures 275

- 13.1 Conditional Constructs 276
 - 13.1.1 The **if** Statement 276
 - 13.1.2 The **if-else** Statement 278
 - 13.1.3 The **switch** Statement 282
 - 13.1.4 An Example Program 284
- 13.2 Iteration Constructs 285
 - 13.2.1 The **while** Statement 286
 - 13.2.2 The **for** Statement 288
 - 13.2.3 The **do-while** Statement 292
 - 13.2.4 The **break** and **continue** Statements 294
- 13.3 Completing the Picture: C Syntax 295
 - 13.3.1 Declarations 295
 - 13.3.2 Statements 296
- 13.4 Problem Solving Using C 296
 - 13.4.1 Problem 1: Approximating the Value of π 296
 - 13.4.2 Problem 2: Finding Prime Numbers Less than 100 298
 - 13.4.3 Problem 3: Detecting a Sequence of Text 301
- Problems 304

Chapter 14

Functions 311

- 14.1 Introduction 311
- 14.2 High-level Programming Structure 312
- 14.3 Functions in C 313
 - 14.3.1 The Declaration 315
 - 14.3.2 The Call 316
 - 14.3.3 The Definition 316
 - 14.3.4 The Return Value 316
- 14.4 Another Example 317
- 14.5 The Run-time Stack 317
 - 14.5.1 The Activation Record 318
 - 14.5.2 Activation Records During Execution 320
- 14.6 Implementing Functions in C 322
 - 14.6.1 The Call 323
 - 14.6.2 Starting the Callee Function 324
 - 14.6.3 Ending the Callee Function 325
 - 14.6.4 Returning to the Caller Function 325
 - 14.6.5 Tying It All Together 326
- 14.7 Problem Solving: Case Conversion 327
 - Problems 330

Chapter 15

Debugging 335

- 15.1 Introduction 335
- 15.2 Types of Errors 336
 - 15.2.1 Syntactic Errors 336
 - 15.2.2 Semantic Errors 337
 - 15.2.3 Algorithmic Errors 339
- 15.3 Debugging Techniques 339
 - 15.3.1 Ad-hoc Techniques 340
 - 15.3.2 The Source-level Debugger 341
- Problems 342

Chapter 16

Recursion 347

- 16.1 Introduction 347
- 16.2 What Is Recursion? 348
- 16.3 A High-level Example: Binary Search 349
- 16.4 Towers of Hanoi 350
- 16.5 A Detailed Example in C 354
- 16.6 Another Detailed Example in C 358
 - Problems 359

Chapter 17

Pointers and Arrays 365

- 17.1 Introduction 365
- 17.2 Pointers 366
 - 17.2.1 Declaring Pointer Variables 367
 - 17.2.2 Operators for Pointers 368
 - 17.2.3 Some Examples Using Pointer Variables 370
 - 17.2.4 The Syntax Demystified 374
- 17.3 Arrays 374
 - 17.3.1 Declaring Arrays and Accessing Elements 375
 - 17.3.2 Examples Using Arrays 376
 - 17.3.3 Arrays as Parameters 379
 - 17.3.4 Strings in C 380
 - 17.3.5 The Relationship Between Arrays and Pointers 383
 - 17.3.6 More Examples 384
 - 17.3.7 Common Pitfalls with Arrays in C 387
- Problems 389

Chapter 18

I/O in C 393

- 18.1 Introduction 393
- 18.2 A Brief Note About the C Standard Library 393
- 18.3 I/O, One Character at a Time 394
 - 18.3.1 `putchar` 394
 - 18.3.2 `getchar` 395
 - 18.3.3 Buffered I/O 395
- 18.4 Slightly More Sophisticated I/O 397
 - 18.4.1 `printf` 397
 - 18.4.2 `scanf` 399
 - 18.4.3 Variable Argument Lists 401
- 18.5 I/O from Files 402
 - Problems 404

Chapter 19

Data Structures 407

- 19.1 Introduction 407
- 19.2 Structures 408
 - 19.2.1 The Basics of Structures in C 408
 - 19.2.2 Arrays and Pointers with Structures 411

- 19.3 A Foray into Dynamic Allocation 413
- 19.4 The Linked List 416
 - 19.4.1 What Is a Linked List? 416
 - 19.4.2 An Example Using Linked Lists 417
- 19.5 Conclusion 425
 - Problems 425

Appendix A

The LC-2 ISA 429

- A.1 Overview 429
- A.2 Notation 430
- A.3 The Instruction Set 430

Appendix B

From LC-2 to IA-32 451

- B.1 LC-2 Features and Corresponding IA-32 Features 452
 - B.1.1 Instruction Set 452
 - B.1.2 Memory 457
 - B.1.3 Internal State 459
- B.2 The Format and Specification of IA-32 Instructions 462
 - B.2.1 Prefix 462
 - B.2.2 Opcode 463
 - B.2.3 ModR/M Byte 463
 - B.2.4 SIB Byte 464
 - B.2.5 Displacement 464
 - B.2.6 Immediate 465
- B.3 An Example 466

Appendix C

The Microarchitecture of the LC-2 471

- C.1 Overview 471
- C.2 The State Machine 473
- C.3 The Data Path 475
- C.4 The Control Structure 475
- C.5 Memory-mapped I/O 481
- C.6 Interrupt Control 481
 - C.6.1 Initiating an Interrupt 482
 - C.6.2 Returning from an Interrupt, RTI 485
- C.7 Control Store 486

Appendix D

The C Programming Language 489

- D.1 Overview 489
- D.2 C Conventions 489
 - D.2.1 Source Files 489
 - D.2.2 Header Files 489
 - D.2.3 Comments 490
 - D.2.4 Literals 490
 - D.2.5 Formatting 492
 - D.2.6 Keywords 492
- D.3 Identifiers, Types, and Declarations 492
 - D.3.1 Identifiers 492
 - D.3.2 Basic Data Types 493
 - D.3.3 Type Qualifiers 494
 - D.3.4 Storage Class 495
 - D.3.5 Derived Types 495
 - D.3.6 Structures 496
- D.4 Declarations 497
 - D.4.1 Variable Declarations 497
 - D.4.2 Function Declarations 498
 - D.4.3 **typedef** 498
- D.5 Operators 499
 - D.5.1 Assignment Operator 499
 - D.5.2 Arithmetic Operators 500
 - D.5.3 Bitwise Operators 500
 - D.5.4 Logical Operators 500
 - D.5.5 Relational Operators 501
 - D.5.6 Special Operators 501
 - D.5.7 Conditional Expression 501
 - D.5.8 Pointer, Array, Structure Operators 502

- D.5.9 Miscellaneous Operators 502
- D.5.10 Precedence 502
- D.6 Expressions and Statements 503
 - D.6.1 Expressions 503
 - D.6.2 Statements 503
- D.7 Control Flow 503
 - D.7.1 **if** 503
 - D.7.2 **if-else** 504
 - D.7.3 **switch** 504
 - D.7.4 **while** 506
 - D.7.5 **for** 506
 - D.7.6 **do-while** 507
 - D.7.7 **break** 507
 - D.7.8 **continue** 508
 - D.7.9 **return** 508
- D.8 Standard Library Functions 509
 - D.8.1 I/O Functions 509
 - D.8.2 String Functions 510
 - D.8.3 Math Functions 511
 - D.8.4 Utility Functions 511

Appendix E

Useful Tables 513

- E.1 Conversion Specifications for C I/O 513
- E.2 ASCII Codes 514
- E.3 Commonly Used Numerical Prefixes 515

Index 517