

# CONTENTS IN DETAIL

<b>ACKNOWLEDGMENTS</b>	<b>xix</b>
------------------------	------------

<b>PREFACE</b>	<b>xxi</b>
----------------	------------

<b>INTRODUCTION</b>	<b>xxv</b>
---------------------	------------

Why Good Programming is Important . . . . .	.xxvi
Learning to Code is Only a Starting Place . . . . .	xxvii
Importance of Low-Level Knowledge . . . . .	xxviii
Who Should Read This Book? . . . . .	xxviii
What Are Computers? . . . . .	.xxix
What Is Computer Programming? . . . . .	.xxix
Coding, Programming, Engineering, and Computer Science . . . . .	xxxii
The Landscape. . . . .	xxxiii
What's in This Book . . . . .	xxxv

<b>1</b>	
<b>THE INTERNAL LANGUAGE OF COMPUTERS</b>	<b>1</b>

What Is Language? . . . . .	2
Written Language . . . . .	2
The Bit . . . . .	3
Logic Operations . . . . .	3
Boolean Algebra. . . . .	4
De Morgan's Law . . . . .	5
Representing Integers Using Bits. . . . .	6
Representing Positive Numbers . . . . .	6
Binary Addition. . . . .	8
Representing Negative Numbers. . . . .	10
Representing Real Numbers. . . . .	14
Fixed-Point Representation . . . . .	14
Floating-Point Representation . . . . .	15
The IEEE Floating-Point Standard . . . . .	17
Binary-Coded Decimal System . . . . .	18
Easier Ways to Work with Binary Numbers . . . . .	18
Octal Representation . . . . .	18
Hexadecimal Representation . . . . .	19
Representing the Context . . . . .	20
Naming Groups of Bits . . . . .	20
Representing Text. . . . .	22
The American Standard Code for Information Interchange . . . . .	22
The Evolution of Other Standards . . . . .	24
Unicode Transformation Format 8-bit . . . . .	24
Using Characters to Represent Numbers . . . . .	25
Quoted-Printable Encoding. . . . .	26
Base64 Encoding . . . . .	26
URL Encoding . . . . .	27

Representing Colors . . . . .	27
Adding Transparency . . . . .	29
Encoding Colors . . . . .	30
Summary . . . . .	31

## **2** **COMBINATORIAL LOGIC** **33**

The Case for Digital Computers . . . . .	34
The Difference Between Analog and Digital . . . . .	35
Why Size Matters in Hardware . . . . .	36
Digital Makes for More Stable Devices . . . . .	37
Digital in an Analog World . . . . .	38
Why Bits Are Used Instead of Digits . . . . .	40
A Short Primer on Electricity . . . . .	41
Using Plumbing to Understand Electricity . . . . .	41
Electrical Switches . . . . .	44
Building Hardware for Bits . . . . .	47
Relays . . . . .	47
Vacuum Tubes . . . . .	50
Transistors . . . . .	51
Integrated Circuits . . . . .	52
Logic Gates . . . . .	53
Improving Noise Immunity with Hysteresis . . . . .	54
Differential Signaling . . . . .	55
Propagation Delay . . . . .	57
Output Variations . . . . .	58
Building More Complicated Circuits . . . . .	60
Building an Adder . . . . .	60
Building Decoders . . . . .	63
Building Demultiplexers . . . . .	64
Building Selectors . . . . .	65
Summary . . . . .	67

## **3** **SEQUENTIAL LOGIC** **69**

Representing Time . . . . .	70
Oscillators . . . . .	70
Clocks . . . . .	71
Latches . . . . .	71
Gated Latches . . . . .	73
Flip-Flops . . . . .	74
Counters . . . . .	77
Registers . . . . .	78
Memory Organization and Addressing . . . . .	79
Random-Access Memory . . . . .	82
Read-Only Memory . . . . .	83
Block Devices . . . . .	85
Flash Memory and Solid State Disks . . . . .	88
Error Detection and Correction . . . . .	88
Hardware vs. Software . . . . .	90
Summary . . . . .	91

<b>4</b>	<b>COMPUTER ANATOMY</b>	<b>93</b>
Memory . . . . .		94
Input and Output . . . . .		96
The Central Processing Unit . . . . .		97
Arithmetic and Logic Unit . . . . .		97
Shiftiness . . . . .		99
Execution Unit . . . . .		100
Instruction Set . . . . .		102
Instructions . . . . .		102
Addressing Modes . . . . .		104
Condition Code Instructions . . . . .		105
Branching . . . . .		105
Final Instruction Set . . . . .		106
The Final Design . . . . .		109
The Instruction Register . . . . .		109
Data Paths and Control Signals . . . . .		109
Traffic Control . . . . .		110
RISC vs. CISC Instruction Sets . . . . .		113
Graphics Processing Units . . . . .		114
Summary . . . . .		115

<b>5</b>	<b>COMPUTER ARCHITECTURE</b>	<b>117</b>
Basic Architectural Elements . . . . .		118
Processor Cores . . . . .		118
Microprocessors and Microcomputers . . . . .		119
Procedures, Subroutines, and Functions . . . . .		120
Stacks . . . . .		122
Interrupts . . . . .		125
Relative Addressing . . . . .		128
Memory Management Units . . . . .		130
Virtual Memory . . . . .		132
System and User Space . . . . .		133
Memory Hierarchy and Performance . . . . .		133
Coprocessors . . . . .		135
Arranging Data in Memory . . . . .		136
Running Programs . . . . .		137
Memory Power . . . . .		138
Summary . . . . .		139

<b>6</b>	<b>COMMUNICATIONS BREAKDOWN</b>	<b>141</b>
Low-Level I/O . . . . .		142
I/O Ports . . . . .		142
Push My Buttons . . . . .		144
Let There Be Lights . . . . .		146
Lights, Action, . . . . .		147
Bright Ideas . . . . .		148
2 <sup>n</sup> Shades of Gray . . . . .		149

Quadrature . . . . .	150
Parallel Communication . . . . .	152
Serial Communication . . . . .	152
Catch a Wave . . . . .	154
Universal Serial Bus . . . . .	156
Networking . . . . .	156
Modern LANs . . . . .	158
The Internet . . . . .	158
Analog in the Digital World . . . . .	160
Digital-to-Analog Conversion . . . . .	160
Analog-to-Digital Conversion . . . . .	162
Digital Audio . . . . .	165
Digital Images . . . . .	173
Video . . . . .	174
Human Interface Devices . . . . .	176
Terminals . . . . .	176
Graphics Terminals . . . . .	177
Vector Graphics . . . . .	178
Raster Graphics . . . . .	180
Keyboard and Mouse . . . . .	181
Summary . . . . .	181

## 7

### ORGANIZING DATA

**183**

Primitive Data Types . . . . .	184
Arrays . . . . .	185
Bitmaps . . . . .	187
Strings . . . . .	188
Compound Data Types . . . . .	189
Singly Linked Lists . . . . .	191
Dynamic Memory Allocation . . . . .	195
More Efficient Memory Allocation . . . . .	196
Garbage Collection . . . . .	197
Doubly Linked Lists . . . . .	198
Hierarchical Data Structures . . . . .	199
Storage for the Masses . . . . .	203
Databases . . . . .	204
Indices . . . . .	206
Moving Data Around . . . . .	206
Vectored I/O . . . . .	210
Object-Oriented Pitfalls . . . . .	211
Sorting . . . . .	212
Making a Hash of Things . . . . .	213
Efficiency vs. Performance . . . . .	215
Summary . . . . .	216

## 8

### LANGUAGE PROCESSING

**217**

Assembly Language . . . . .	217
High-Level Languages . . . . .	219

Structured Programming . . . . .	220
Lexical Analysis . . . . .	221
State Machines . . . . .	223
Regular Expressions . . . . .	224
From Words to Sentences . . . . .	226
The Language-of-the-Day Club . . . . .	228
Parse Trees . . . . .	228
Interpreters . . . . .	231
Compilers . . . . .	232
Optimization . . . . .	234
Be Careful with Hardware . . . . .	236
Summary . . . . .	236

## 9

### **THE WEB BROWSER 237**

Markup Languages . . . . .	238
Uniform Resource Locators . . . . .	239
HTML Documents . . . . .	240
The Document Object Model . . . . .	242
Tree Lexicon . . . . .	243
Interpreting the DOM . . . . .	244
Cascading Style Sheets . . . . .	244
XML and Friends . . . . .	248
JavaScript . . . . .	251
jQuery . . . . .	253
SVG . . . . .	254
HTML5 . . . . .	255
JSON . . . . .	255
Summary . . . . .	256

## 10

### **APPLICATION AND SYSTEM PROGRAMMING 259**

Guess the Animal Version 1: HTML and JavaScript . . . . .	262
Application-Level Skeleton . . . . .	263
Web Page Body . . . . .	263
The JavaScript . . . . .	264
The CSS . . . . .	267
Guess the Animal Version 2: C . . . . .	267
Terminals and the Command Line . . . . .	267
Building the Program . . . . .	268
Terminals and Device Drivers . . . . .	268
Context Switching . . . . .	269
Standard I/O . . . . .	271
Circular Buffers . . . . .	272
Better Code Through Good Abstractions . . . . .	273
Some Mechanics . . . . .	274
Buffer Overflow . . . . .	275
The C Program . . . . .	275
Training . . . . .	281
Summary . . . . .	282

## **11 SHORTCUTS AND APPROXIMATIONS 283**

Table Lookup . . . . .	284
Conversion . . . . .	284
Texture Mapping . . . . .	285
Character Classification . . . . .	288
Integer Methods . . . . .	290
Straight Lines . . . . .	292
Curves Ahead . . . . .	298
Polynomials . . . . .	301
Recursive Subdivision . . . . .	301
Spirals . . . . .	301
Constructive Geometry . . . . .	304
Shifting and Masking . . . . .	311
More Math Avoidance . . . . .	313
Power Series Approximations . . . . .	313
The CORDIC Algorithm . . . . .	313
Somewhat Random Things . . . . .	318
Space-Filling Curves . . . . .	319
L-Systems . . . . .	320
Going Stochastic . . . . .	322
Quantization . . . . .	323
Summary . . . . .	333

## **12 DEADLOCKS AND RACE CONDITIONS 335**

What Is a Race Condition? . . . . .	336
Shared Resources . . . . .	337
Processes and Threads . . . . .	337
Locks . . . . .	339
Transactions and Granularity . . . . .	340
Waiting for a Lock . . . . .	341
Deadlocks . . . . .	341
Short-Term Lock Implementation . . . . .	342
Long-Term Lock Implementation . . . . .	343
Browser JavaScript . . . . .	343
Asynchronous Functions and Promises . . . . .	346
Summary . . . . .	350

## **13 SECURITY 351**

Overview of Security and Privacy . . . . .	352
Threat Model . . . . .	352
Trust . . . . .	353
Physical Security . . . . .	355
Communications Security . . . . .	356
Modern Times . . . . .	357

Metadata and Surveillance . . . . .	359
The Social Context . . . . .	359
Authentication and Authorization . . . . .	361
Cryptography . . . . .	362
Steganography . . . . .	362
Substitution Ciphers . . . . .	363
Transposition Ciphers . . . . .	365
More Complex Ciphers . . . . .	366
One-Time Pads . . . . .	367
The Key Exchange Problem . . . . .	367
Public Key Cryptography . . . . .	368
Forward Secrecy . . . . .	369
Cryptographic Hash Functions . . . . .	369
Digital Signatures . . . . .	370
Public Key Infrastructure . . . . .	370
Blockchain . . . . .	371
Password Management . . . . .	371
Software Hygiene . . . . .	372
Protect the Right Stuff . . . . .	372
Triple-Check Your Logic . . . . .	373
Check for Errors . . . . .	373
Minimize Attack Surfaces . . . . .	373
Stay in Bounds . . . . .	374
Generating Good Random Numbers Is Hard . . . . .	375
Know Thy Code . . . . .	376
Extreme Cleverness Is Your Enemy . . . . .	378
Understand What’s Visible . . . . .	378
Don’t Overcollect . . . . .	379
Don’t Hoard . . . . .	379
Dynamic Memory Allocation Isn’t Your Friend . . . . .	379
Garbage Collection Is Not Your Friend Either . . . . .	381
Data as Code . . . . .	382
Summary . . . . .	384

## 14

<b>MACHINE INTELLIGENCE</b>	<b>385</b>
Overview . . . . .	386
Machine Learning . . . . .	388
Bayes . . . . .	389
Gauss . . . . .	390
Sobel . . . . .	393
Canny . . . . .	398
Feature Extraction . . . . .	399
Neural Networks . . . . .	401
Using Machine Learning Data . . . . .	406
Artificial Intelligence . . . . .	407
Big Data . . . . .	409
Summary . . . . .	412

<b>15</b>		
<b>REAL-WORLD CONSIDERATIONS</b>		<b>413</b>
The Value Proposition . . . . .		414
How We Got Here . . . . .		416
A Short History . . . . .		416
Open Source Software . . . . .		419
Creative Commons . . . . .		420
The Rise of Portability . . . . .		420
Package Management . . . . .		421
Containers . . . . .		422
Java . . . . .		422
Node.js . . . . .		424
Cloud Computing . . . . .		424
Virtual Machines . . . . .		425
Portable Devices . . . . .		425
The Programming Environment . . . . .		425
Are You Experienced? . . . . .		426
Learning to Estimate . . . . .		426
Scheduling Projects . . . . .		426
Decision Making . . . . .		427
Working with Different Personalities . . . . .		428
Navigating Workplace Culture . . . . .		429
Making Informed Choices . . . . .		430
Development Methodologies . . . . .		430
Project Design . . . . .		432
Writing It Down . . . . .		432
Fast Prototyping . . . . .		432
Interface Design . . . . .		433
Reusing Code or Writing Your Own . . . . .		436
Project Development . . . . .		436
The Talk . . . . .		437
Portable Code . . . . .		439
Source Control . . . . .		440
Testing . . . . .		440
Bug Reporting and Tracking . . . . .		441
Refactoring . . . . .		441
Maintenance . . . . .		441
Be Stylish . . . . .		442
Fix, Don't Re-create . . . . .		443
Summary . . . . .		444
<b>INDEX</b>		<b>445</b>