

Contents

Foreword xxxi

Introduction xxxiii

Part I 1

Chapter 1 The Best Optimizer Is between Your Ears 3

The Human Element of Code Optimization 5

Understanding High Performance 6

When Fast Isn't Fast 6

Rules for Building High-Performance Code 7

Know Where You're Going 8

Make a Big Map 8

Make Lots of Little Maps 8

Know the Territory 12

Know When It Matters 13

Always Consider the Alternatives 14

Know How to Turn On the Juice 16

Where We've Been, What We've Seen 19

Where We're Going 19

Chapter 2 A World Apart 21

The Unique Nature of Assembly

Language Optimization 23

Instructions: The Individual versus
the Collective 23

Assembly Is Fundamentally Different 25

Transformation Inefficiencies 25

Self-Reliance 27

<i>Knowledge</i>	27
The Flexible Mind	28
<i>Where to Begin?</i>	30

Chapter 3	Assume Nothing	31
	Understanding and Using the Zen Timer	33
	The Costs of Ignorance	34
	The Zen Timer	35
	<i>The Zen Timer Is a Means, Not an End</i>	42
	<i>Starting the Zen Timer</i>	43
	Time and the PC	43
	Stopping the Zen Timer	46
	Reporting Timing Results	47
	Notes on the Zen Timer	48
	A Sample Use of the Zen Timer	49
	The Long-Period Zen Timer	53
	<i>Stopping the Clock</i>	54
	Example Use of the Long-Period Zen Timer	66
	Using the Zen Timer from C	69
	<i>Watch Out for Optimizing Assemblers!</i>	71
	<i>Further Reading</i>	72
	<i>Armed with the Zen Timer, Onward and Upward</i>	72

Chapter 4	In the Lair of the Cycle-Eaters	75
	How the PC Hardware Devours Code Performance	77
	Cycle-Eaters	78
	The Nature of Cycle-Eaters	78
	<i>The 8088's Ancestral Cycle-Eaters</i>	79
	The 8-Bit Bus Cycle-Eater	79
	<i>The Impact of the 8-Bit Bus Cycle-Eater</i>	82
	<i>What to Do about the 8-Bit Bus Cycle-Eater?</i>	83
	The Prefetch Queue Cycle-Eater	86
	<i>Official Execution Times Are Only Part of the Story</i>	87
	<i>There Is No Such Beast as a True Instruction Execution Time</i>	88
	<i>Approximating Overall Execution Times</i>	93
	<i>What to Do about the Prefetch Queue Cycle-Eater?</i>	93
	<i>Holding Up the 8088</i>	94

	Dynamic RAM Refresh: The Invisible Hand	95
	<i>How DRAM Refresh Works in the PC</i>	95
	<i>The Impact of DRAM Refresh</i>	97
	<i>What to Do About the DRAM Refresh Cycle-Eater?</i>	98
	Wait States	99
	The Display Adapter Cycle-Eater	101
	<i>The Impact of the Display Adapter Cycle-Eater</i>	104
	<i>What to Do about the Display Adapter Cycle-Eater?</i>	107
	<i>Cycle-Eaters: A Summary</i>	108
	<i>What Does It All Mean?</i>	108
Chapter 5	Crossing the Border	111
	Searching Files with Restartable Blocks	113
	<i>Searching for Text</i>	114
	Avoiding the String Trap	115
	Brute-Force Techniques	115
	Using memchr()	116
	<i>Making a Search Restartable</i>	117
	Interpreting Where the Cycles Go	121
	<i>Knowing When Assembly Is Pointless</i>	122
	Always Look Where Execution Is Going	123
Chapter 6	Looking Past Face Value	125
	How Machine Instructions May Do More Than You Think	127
	<i>Memory Addressing and Arithmetic</i>	128
	Math via Memory Addressing	130
	<i>The Wonders of LEA on the 386</i>	131
	Multiplication with LEA Using Non-Powers of Two	132
Chapter 7	Local Optimization	135
	Optimizing Halfway between Algorithms and Cycle Counting	137
	<i>When LOOP Is a Bad Idea</i>	138
	The Lessons of LOOP and JCXZ	139
	<i>Avoiding LOOPS of Any Stripe</i>	140

Local Optimization 140
Unrolling Loops 143
 Rotating and Shifting with Tables 145
 NOT Flips Bits—Not Flags 146
 Incrementing with and without Carry 147

Chapter 8 Speeding Up C with Assembly Language 149

Jumping Languages When You Know
It'll Help 151
 Billy, Don't Be a Compiler 152
Don't Call Your Functions on Me, Baby 153
Stack Frames Slow So Much 153
Torn Between Two Segments 154
 Why Speeding Up Is Hard to Do 154
Taking It to the Limit 155
 A C-to-Assembly Case Study 156

Chapter 9 Hints My Readers Gave Me 167

Optimization Odds and Ends from
the Field 169
 Another Look at LEA 170
 The Kennedy Portfolio 171
 Speeding Up Multiplication 173
 Optimizing Optimized Searching 174
 Short Sorts 180
 Full 32-Bit Division 181
 Sweet Spot Revisited 184
 Hard-Core Cycle Counting 185
 Hardwired Far Jumps 186
 Setting 32-Bit Registers: Time versus Space 187

Chapter 10 Patient Coding, Faster Code 189

How Working Quickly Can Bring Execution
to a Crawl 191
 The Case for Delayed Gratification 192
The Brute-Force Syndrome 193
 Wasted Breakthroughs 196

Recursion 199
Patient Optimization 200

Chapter 11	Pushing the 286 and 386	205
	New Registers, New Instructions, New Timings, New Complications	207
	Family Matters	208
	Crossing the Gulf to the 286 and the 386	208
	In the Lair of the Cycle-Eaters, Part II	209
	<i>System Wait States</i>	210
	<i>Data Alignment</i>	213
	Code Alignment	215
	<i>Alignment and the 386</i>	218
	<i>Alignment and the Stack</i>	218
	<i>The DRAM Refresh Cycle-Eater: Still an Act of God</i>	219
	<i>The Display Adapter Cycle-Eater</i>	219
	New Instructions and Features: The 286	221
	New Instructions and Features: The 386	222
	<i>Optimization Rules: The More Things Change...</i>	223
	<i>Detailed Optimization</i>	223
	popf and the 286	225

Chapter 12	Pushing the 486	233
	It's Not Just a Bigger 386	235
	<i>Enter the 486</i>	236
	Rules to Optimize By	236
	<i>The Hazards of Indexed Addressing</i>	237
	<i>Calculate Memory Pointers Ahead of Time</i>	238
	Caveat Programmor	241
	<i>Stack Addressing and Address Pipelining</i>	241
	<i>Problems with Byte Registers</i>	242
	<i>More Fun with Byte Registers</i>	244
	<i>Timing Your Own 486 Code</i>	245
	The Story Continues	246

Chapter 13	Aiming the 486	247
	Pipelines and Other Hazards of the High End	249

	<i>486 Pipeline Optimization</i>	250
	BSWAP: More Useful Than You Might Think	252
	Pushing and Popping Memory	254
	Optimal 1-Bit Shifts and Rotates	255
	32-Bit Addressing Modes	256
Chapter 14	Boyer-Moore String Searching	259
	Optimizing a Pretty Optimum Search Algorithm	261
	String Searching Refresher	262
	The Boyer-Moore Algorithm	263
	Boyer-Moore: The Good and the Bad	266
	Further Optimization of Boyer-Moore	274
	<i>Know What You Know</i>	277
Chapter 15	Linked Lists and Unintended Challenges	279
	Unfamiliar Problems with Familiar Data Structures	281
	Linked Lists	282
	Dummies and Sentinels	285
	Circular Lists	288
	Hi/Lo in 24 Bytes	292
Chapter 16	There Ain't No Such Thing as the Fastest Code	295
	Lessons Learned in the Pursuit of the Ultimate Word Counter	297
	Counting Words in a Hurry	298
	<i>Which Way to Go from Here?</i>	302
	Challenges and Hazards	305
	<i>Blinding Yourself to a Better Approach</i>	306

	<i>Watch Out for Luggable Assumptions!</i>	306
	The Astonishment of Right-Brain Optimization	307
	Levels of Optimization	312
	<i>Optimization Level 1: Good Code</i>	312
	Level 2: A New Perspective	315
	<i>Level 3: Breakthrough</i>	316
	<i>Enough Word Counting Already!</i>	319
Chapter 17	The Game of Life	321
	The Triumph of Algorithmic Optimization in a Cellular Automata Game	323
	Conway's Game	324
	<i>The Rules of the Game</i>	324
	Where Does the Time Go?	329
	The Hazards and Advantages of Abstraction	330
	Heavy-Duty C++ Optimization	336
	Bringing In the Right Brain	338
	<i>Re-Examining the Task</i>	338
	<i>Acting on What We Know</i>	340
	<i>The Challenge That Ate My Life</i>	346
Chapter 18	It's a Wonderful Life	347
	Optimization beyond the Pale	349
	Breaking the Rules	350
	Table-Driven Magic	351
	Keeping Track of Change with a Change List	363
	<i>A Layperson's Overview of QLIFE</i>	366
Chapter 19	Pentium: Not the Same Old Song	369
	Learning a Whole Different Set of Optimization Rules	371
	The Return of Optimization as Art	372

	The Pentium: An Overview	373
	<i>Crossing Cache Lines</i>	373
	<i>Cache Organization</i>	374
	Faster Addressing and More	375
	Branch Prediction	377
	Miscellaneous Pentium Topics	378
	<i>486 versus Pentium Optimization</i>	378
	<i>Going Superscalar</i>	379
Chapter 20	Pentium Rules	381
	How Your Carbon-Based Optimizer Can Put the “Super” in Superscalar	383
	An Instruction in Every Pipe	384
	V-Pipe-Capable Instructions	386
	Lockstep Execution	390
	Superscalar Notes	394
	<i>Register Starvation</i>	395
Chapter 21	Unleashing the Pentium’s V-pipe	397
	Focusing on Keeping Both Pentium Pipes Full	399
	Address Generation Interlocks	400
	Register Contention	403
	<i>Exceptions to Register Contention</i>	404
	Who’s in First?	405
	Pentium Optimization in Action	406
	<i>A Quick Note on the 386 and 486</i>	411
Chapter 22	Zenning and the Flexible Mind	413
	Taking a Spin through What You’ve Learned	415
	Zenning	415

Part II 421

Chapter 23 Bones and Sinew 423

At the Very Heart of Standard PC

Graphics 425

The VGA 426

An Introduction to VGA Programming 427

At the Core 427

Linear Planes and True VGA Modes 430

Smooth Panning 441

Color Plane Manipulation 443

Page Flipping 444

The Hazards of VGA Clones 446

Just the Beginning 447

The Macro Assembler 447

Chapter 24 Parallel Processing with the VGA 449

Taking on Graphics Memory Four Bytes
at a Time 451

VGA Programming: ALUs and Latches 451

Notes on the ALU/Latch Demo
Program 458

Chapter 25 VGA Data Machinery 461

The Barrel Shifter, Bit Mask, and
Set/Reset Mechanisms 463

VGA Data Rotation 463

The Bit Mask 464

The VGA's Set/Reset Circuitry 471

Setting All Planes to a Single Color 473

Manipulating Planes Individually 476

Notes on Set/Reset 478

A Brief Note on Word OUTs 479

Chapter 26	VGA Write Mode 3	481
	The Write Mode That Grows on You	483
	A Mode Born in Strangeness	483
	A Note on Preserving Register Bits	496
Chapter 27	Yet Another VGA Write Mode	499
	Write Mode 2, Chunky Bitmaps, and Text-Graphics Coexistence	501
	Write Mode 2 and Set/Reset	501
	<i>A Byte's Progress in Write Mode 2</i>	502
	<i>Copying Chunky Bitmaps to VGA Memory Using Write Mode 2</i>	504
	<i>Drawing Color-Patterned Lines Using Write Mode 2</i>	509
	When to Use Write Mode 2 and When to Use Set/Reset	515
	Mode 13H—320×200 with 256 Colors	515
	Flipping Pages from Text to Graphics and Back	515
Chapter 28	Reading VGA Memory	523
	Read Modes 0 and 1, and the Color Don't Care Register	525
	Read Mode 0	525
	Read Mode 1	531
	When all Planes "Don't Care"	534
Chapter 29	Saving Screens and Other VGA Mysteries	539
	Useful Nuggets from the VGA Zen File	541
	Saving and Restoring EGA and VGA Screens	541
	16 Colors out of 64	548
	Overscan	555

	A Bonus Blanker	556
	Modifying VGA Registers	558
Chapter 30	Video Est Omnis Divisa	561
	The Joys and Galling Problems of Using Split Screens on the EGA and VGA	563
	How the Split Screen Works	563
	<i>The Split Screen in Action</i>	565
	<i>VGA and EGA Split-Screen Operation Don't Mix</i>	572
	Setting the Split-Screen-Related Registers	573
	The Problem with the EGA Split Screen	573
	Split Screen and Panning	574
	<i>The Split Screen and Horizontal Panning: An Example</i>	575
	Notes on Setting and Reading Registers	582
	Split Screens in Other Modes	584
	How Safe?	585
Chapter 31	Higher 256-Color Resolution on the VGA	587
	When Is 320×200 Really 320×400?	589
	Why 320×200? Only IBM Knows for Sure	590
	320×400 256-Color Mode	590
	<i>Display Memory Organization in 320×400 Mode</i>	591
	<i>Reading and Writing Pixels</i>	593
	Two 256-Color Pages	600
	Something to Think About	605
Chapter 32	Be It Resolved: 360×480	607
	Taking 256-Color Modes About as Far as the Standard VGA Can Take Them	609
	Extended 256-Color Modes: What's Not to Like?	610
	360×480 256-Color Mode	611
	How 360×480 256-Color Mode Works	619

*480 Scan Lines per Screen: A Little Slower,
But No Big Deal* 619
360 Pixels per Scan Line: No Mean Feat 620
Accessing Display Memory in 360×480 256-Color Mode 621

Chapter 33	Yogi Bear and Eurythmics Confront VGA Colors	623
	The Basics of VGA Color Generation	625
	VGA Color Basics	626
	<i>The Palette RAM</i>	626
	<i>The DAC</i>	626
	<i>Color Paging with the Color Select Register</i>	628
	<i>256-Color Mode</i>	629
	<i>Setting the Palette RAM</i>	629
	<i>Setting the DAC</i>	630
	If You Can't Call the BIOS, Who Ya Gonna Call?	631
	An Example of Setting the DAC	632

Chapter 34	Changing Colors without Writing Pixels	637
	Special Effects through Realtime Manipulation of DAC Colors	639
	Color Cycling	639
	The Heart of the Problem	640
	<i>Loading the DAC via the BIOS</i>	641
	<i>Loading the DAC Directly</i>	642
	A Test Program for Color Cycling	643
	Color Cycling Approaches that Work	649
	Odds and Ends	651
	<i>The DAC Mask</i>	651
	<i>Reading the DAC</i>	651
	<i>Cycling Down</i>	652

Chapter 35	Bresenham Is Fast, and Fast Is Good	653
	Implementing and Optimizing Bresenham's Line-Drawing Algorithm	655
	The Task at Hand	656
	Bresenham's Line-Drawing Algorithm	657
	<i>Strengths and Weaknesses</i>	660
	An Implementation in C	661
	<i>Looking at EVGALine</i>	665
	<i>Drawing Each Line</i>	668
	<i>Drawing Each Pixel</i>	669
	Comments on the C Implementation	670
	Bresenham's Algorithm in Assembly	671
Chapter 36	The Good, the Bad, and the Run-Sliced	679
	Faster Bresenham Lines with Run-Length Slice Line Drawing	681
	Run-Length Slice Fundamentals	683
	Run-Length Slice Implementation	685
	Run-Length Slice Details	687
Chapter 37	Dead Cats and Lightning Lines	695
	Optimizing Run-Length Slice Line Drawing in a Major Way	697
	Fast Run-Length Slice Line Drawing	698
	<i>How Fast Is Fast?</i>	704
	<i>Further Optimizations</i>	705

Chapter 38	The Polygon Primeval	707
	Drawing Polygons Efficiently and Quickly	709
	Filled Polygons	710
	<i>Which Side Is Inside?</i>	710
	How Do You Fit Polygons Together?	712
	Filling Non-Overlapping Convex Polygons	713
	Oddball Cases	721
Chapter 39	Fast Convex Polygons	723
	Filling Polygons in a Hurry	725
	Fast Convex Polygon Filling	726
	<i>Fast Drawing</i>	727
	<i>Fast Edge Tracing</i>	730
	The Finishing Touch: Assembly Language	732
	<i>Maximizing REP STOS</i>	735
	Faster Edge Tracing	735
Chapter 40	Of Songs, Taxes, and the Simplicity of Complex Polygons	739
	Dealing with Irregular Polygonal Areas	741
	Filling Arbitrary Polygons	742
	<i>Active Edges</i>	742
	Complex Polygon Filling: An Implementation	750
	<i>More on Active Edges</i>	753
	<i>Performance Considerations</i>	753
	Nonconvex Polygons	755
	<i>Details, Details</i>	755
Chapter 41	Those Way-Down Polygon Nomenclature Blues	757
	Names Do Matter when You Conceptualize a Data Structure	759
	Nomenclature in Action	760

Chapter 42	Wu'ed in Haste; Fried, Stewed at Leisure 773
	Fast Antialiased Lines Using Wu's Algorithm 775
	Wu Antialiasing 776
	Tracing and Intensity in One 778
	Sample Wu Antialiasing 782
	<i>Notes on Wu Antialiasing</i> 791
Chapter 43	Bit-Plane Animation 793
	A Simple and Extremely Fast Animation Method for Limited Color 795
	Bit-Planes: The Basics 796
	<i>Stacking the Palette Registers</i> 799
	Bit-Plane Animation in Action 801
	Limitations of Bit-Plane Animation 811
	Shearing and Page Flipping 813
	Beating the Odds in the Jaw- Dropping Contest 814
Chapter 44	Split Screens Save the Page Flipped Day 817
	640×480 Page Flipped Animation in 64K...Almost 819
	A Plethora of Challenges 819
	A Page Flipping Animation Demonstration 820
	<i>Write Mode 3</i> 831
	<i>Drawing Text</i> 832
	<i>Page Flipping</i> 833
	<i>Knowing When to Flip</i> 835
	Enter the Split Screen 836
Chapter 45	Dog Hair and Dirty Rectangles 839
	Different Angles on Animation 841
	Plus ça Change 842

	VGA Access Times	842
	Dirty-Rectangle Animation	844
	<i>So Why Not Use Page Flipping?</i>	846
	Dirty Rectangles in Action	846
	Hi-Res VGA Page Flipping	851
	Another Interesting Twist on Page Flipping	855
Chapter 46	Who Was that Masked Image?	859
	Optimizing Dirty-Rectangle Animation	861
	Dirty-Rectangle Animation, Continued	862
	Masked Images	871
	Internal Animation	872
	<i>Dirty-Rectangle Management</i>	872
	Drawing Order and Visual Quality	873
Chapter 47	Mode X: 256-Color VGA Magic	875
	Introducing the VGA's Undocumented "Animation-Optimal" Mode	877
	What Makes Mode X Special?	878
	Selecting 320×240 256-Color Mode	879
	Designing from a Mode X Perspective	885
	Hardware Assist from an Unexpected Quarter	889
Chapter 48	Mode X Marks the Latch	895
	The Internals of Animation's Best Video Display Mode	897
	Allocating Memory in Mode X	903
	Copying Pixel Blocks within Display Memory	905
	<i>Copying to Display Memory</i>	908
	Who Was that Masked Image Copier?	911

Chapter 49	Mode X 256-Color Animation 913	
	How to Make the VGA Really Get up and Dance 915	
	Masked Copying 915	
	<i>Faster Masked Copying</i> 918	
	<i>Notes on Masked Copying</i> 923	
	Animation 924	
	Mode X Animation in Action 924	
	Works Fast, Looks Great 930	
Chapter 50	Adding a Dimension 931	
	3-D Animation Using Mode X 933	
	References on 3-D Drawing 934	
	The 3-D Drawing Pipeline 935	
	<i>Projection</i> 937	
	<i>Translation</i> 937	
	<i>Rotation</i> 938	
	A Simple 3-D Example 939	
	<i>Notes on the 3-D Animation Example</i> 948	
	An Ongoing Journey 949	
Chapter 51	Sneakers in Space 951	
	Using Backface Removal to Eliminate Hidden Surfaces 953	
	One-sided Polygons: Backface Removal 954	
	<i>Backface Removal in Action</i> 957	
	Incremental Transformation 964	
	A Note on Rounding Negative Numbers 966	
	Object Representation 967	
Chapter 52	Fast 3-D Animation: Meet X-Sharp 969	
	The First Iteration of a Generalized 3-D Animation Package 971	

	This Chapter's Demo Program	972
	A New Animation Framework: X-Sharp	984
	Three Keys to Realtime Animation	
	Performance	985
	<i>Drawbacks</i>	986
	<i>Where the Time Goes</i>	987
Chapter 53	Raw Speed and More	989
	The Naked Truth About Speed in	
	3-D Animation	991
	Raw Speed, Part I: Assembly Language	992
	Raw Speed, Part II: Look it Up	999
	<i>Hidden Surfaces</i>	1000
	<i>Rounding</i>	1002
	Having a Ball	1003
Chapter 54	3-D Shading	1005
	Putting Realistic Surfaces on Animated	
	3-D Objects	1007
	Support for Older Processors	1007
	Shading	1023
	<i>Ambient Shading</i>	1023
	<i>Diffuse Shading</i>	1023
	Shading: Implementation Details	1027
Chapter 55	Color Modeling in	
	256-Color Mode	1031
	Pondering X-Sharp's Color Model in an	
	RGB State of Mind	1033
	A Color Model	1034
	A Bonus from the BitMan	1039
Chapter 56	Pooh and the Space	
	Station	1045
	Using Fast Texture Mapping to Place Pooh	
	on a Polygon	1047

	Principles of Quick-and-Dirty Texture Mapping	1048
	<i>Mapping Textures Made Easy</i>	1049
	<i>Notes on DDA Texture Mapping</i>	1052
	Fast Texture Mapping: An Implementation	1053
Chapter 57	10,000 Freshly Sheared Sheep on the Screen	1061
	The Critical Role of Experience in Implementing Fast, Smooth Texture Mapping	1063
	Visual Quality: A Black Hole ... Er, Art	1064
	Fixed-Point Arithmetic, Redux	1064
	Texture Mapping: Orientation Independence	1065
	Mapping Textures across Multiple Polygons	1068
	<i>Fast Texture Mapping</i>	1068
Chapter 58	Heinlein's Crystal Ball, Spock's Brain, and the 9-Cycle Dare	1077
	Using the Whole-Brain Approach to Accelerate Texture Mapping	1079
	Texture Mapping Redux	1080
	<i>Left-Brain Optimization</i>	1081
	<i>A 90-Degree Shift in Perspective</i>	1084
	That's Nice—But it Sure as Heck Ain't 9 Cycles	1086
	<i>Don't Stop Thinking about Those Cycles</i>	1091
	Texture Mapping Notes	1092
Chapter 59	The Idea of BSP Trees	1095
	What BSP Trees Are and How to Walk Them	1097

BSP Trees	1098
<i>Visibility Determination</i>	1099
<i>Limitations of BSP Trees</i>	1100
Building a BSP Tree	1101
<i>Visibility Ordering</i>	1104
Inorder Walks of BSP Trees	1107
<i>Know It Cold</i>	1109
<i>Measure and Learn</i>	1111
Surfing Amidst the Trees	1113
<i>Related Reading</i>	1114

Chapter 60 **Compiling BSP Trees 1115**

Taking BSP Trees from Concept to Reality	1117
Compiling BSP Trees	1119
<i>Parametric Lines</i>	1119
<i>Parametric Line Clipping</i>	1121
<i>The BSP Compiler</i>	1123
Optimizing the BSP Tree	1128
BSP Optimization: an Undiscovered Country	1129

Chapter 61 **Frames of Reference 1131**

The Fundamentals of the Math behind 3-D Graphics	1133
<i>3-D Math</i>	1134
<i>Foundation Definitions</i>	1134
The Dot Product	1135
<i>Dot Products of Unit Vectors</i>	1136
Cross Products and the Generation of Polygon Normals	1137
Using the Sign of the Dot Product	1140
Using the Dot Product for Projection	1141
<i>Rotation by Projection</i>	1143

Chapter 62	One Story, Two Rules, and a BSP Renderer	1145
	Taking a Compiled BSP Tree from Logical to Visual Reality	1147
	BSP-based Rendering	1148
	The Rendering Pipeline	1157
	<i>Moving the Viewer</i>	1157
	<i>Transformation into Viewspace</i>	1158
	<i>Clipping</i>	1158
	<i>Projection to Screenspace</i>	1159
	<i>Walking the Tree, Backface Culling and Drawing</i>	1160
	Notes on the BSP Renderer	1162
Chapter 63	Floating-Point for Real-Time 3-D	1163
	Knowing When to Hurl Conventional Math Wisdom Out the Window	1165
	Not Your Father's Floating-Point	1167
	Pentium Floating-Point Optimization	1167
	<i>Pipelining, Latency, and Throughput</i>	1168
	<i>FXCH</i>	1169
	The Dot Product	1170
	The Cross Product	1171
	Transformation	1172
	Projection	1174
	Rounding Control	1174
	A Farewell to 3-D Fixed-Point	1175
Chapter 64	Quake's Visible-Surface Determination	1177
	The Challenge of Separating All Things Seen from All Things Unseen	1179

VSD: The Toughest 3-D Challenge of All	1180
The Structure of Quake Levels	1181
Culling and Visible Surface Determination	1181
<i>Nodes Inside and Outside the View Frustum</i>	1183
Overdraw	1184
The Beam Tree	1185
3-D Engine du Jour	1186
<i>Subdividing Raycast</i>	1187
<i>Vertex-Free Surfaces</i>	1187
<i>The Draw-Buffer</i>	1187
<i>Span-Based Drawing</i>	1187
<i>Portals</i>	1188
Breakthrough!	1188
Simplify, and Keep on Trying New Things	1189
Learn Now, Pay Forward	1190
References	1190

Chapter 65	3-D Clipping and Other Thoughts	1191
	Determining What's Inside Your Field of View	1193
	3-D Clipping Basics	1195
	<i>Intersecting a Line Segment with a Plane</i>	1195
	Polygon Clipping	1197
	<i>Clipping to the Frustum</i>	1200
	<i>The Lessons of Listing 65.3</i>	1206
	Advantages of Viewspace Clipping	1207
	Further Reading	1208

Chapter 66	Quake's Hidden-Surface Removal	1209
	Struggling with Z-Order Solutions to the Hidden Surface Problem	1211
	Creative Flux and Hidden Surfaces	1212

<i>Drawing Moving Objects</i>	1212
<i>Performance Impact</i>	1213
<i>Leveling and Improving Performance</i>	1213
Sorted Spans	1214
<i>Edges versus Spans</i>	1215
Edge-Sorting Keys	1220
<i>Where That 1/Z Equation Comes From</i>	1221
<i>Quake and Z-Sorting</i>	1221
<i>Decisions Deferred</i>	1222

Chapter 67	Sorted Spans in Action	1223
	Implementing Independent Span Sorting for Rendering without Overdraw	1225
	Quake and Sorted Spans	1226
	Types of 1/z Span Sorting	1228
	<i>Intersecting Span Sorting</i>	1228
	<i>Abutting Span Sorting</i>	1229
	<i>Independent Span Sorting</i>	1230
	1/z Span Sorting in Action	1230
	<i>Implementation Notes</i>	1239

Chapter 68	Quake's Lighting Model	1243
	A Radically Different Approach to Lighting Polygons	1245
	<i>Problems with Gouraud Shading</i>	1247
	<i>Perspective Correctness</i>	1248
	<i>Decoupling Lighting from Rasterization</i>	1250
	<i>Size and Speed</i>	1251
	<i>Mipmapping To The Rescue</i>	1254
	<i>Two Final Notes on Surface Caching</i>	1255

Chapter 69	Surface Caching and Quake's Triangle Models	1257
	<i>Letting the Graphics Card Build the Textures</i>	1261
	<i>The Light Map as Alpha Texture</i>	1262
	<i>Drawing Triangle Models Fast</i>	1263
	<i>Trading Subpixel Precision for Speed</i>	1265
	<i>An Idea that Didn't Work</i>	1265

An Idea that Did Work 1266
More Ideas that Might Work 1270

Chapter 70 Quake: A Post-Mortem and a Glimpse into the Future 1273

Lighting 1282
Dynamic Lighting 1283
BSP Models 1284
Polygon Models and Z-Buffering 1285
The Subdivision Rasterizer 1286
Sprites 1287
Particles 1287

How We Spent Our Summer Vacation:

After Shipping Quake 1287

Verite Quake 1287
GLQuake 1288
WinQuake 1290
QuakeWorld 1291
Quake 2 1293

Afterword 1297

Index 1299