Index

Numbers

1/z sorting
abutting span sorting, 1229-1230
AddPolygonEdges function, 1232-1233, 1238
vs. BSP-order sorting, 1226-1227
calculating 1/z value, 1220-1222
ClearEdgeLists function, 1236-1237
DrawSpans function, 1236
independent span sorting, 1230, 1231-1238, 1239-1241
intersecting span sorting, 1228-1229
PolyFacesViewer function, 1232
reliability, 1227
ScanEdges function, 1234-1236, 1238-1239
UpdateWorld function, 1237-1238

3-D animation

See also Hidden surface removal; 3-D drawing; 3-D polygon rotation
demo program; X-Sharp 3-D animation package.
demo programs
solid cube rotation program, 957-961, 962-963, 964-966, 967
3-D polygon rotation program, 939, 940-945, 948-949
12-cube rotation program, 972, 973-984, 985-987
depth sorting, 1000, 1001-1002
rotation
ConcatXforms function, 944
matrix representation, 938-939
multiple axes of rotation, 948
XformVec function, 943
rounding vs. truncation, 1002-1003
translation of objects, 937-938

3-D clipping
arithmetic imprecision, handling, 1240
line segments, clipping to planes, 1195-1197
overview, 1195
polygon clipping
BackRotateVector function, 1203
clipping to frustum, 1200, 1201-1206, 1206-1207
ClipToFrustum function, 1204
ClipToPlane function, 1199
optimization, 1207
overview, 1197-1200
PolyFacesViewer function, 1203
ProjectPolygon function, 1201
SetUpFrustum function, 1204
SetWorldspace function, 1204
TransformPoint function, 1203
TransformPolygon function, 1203
UpdateWorld function, 1205
viewspace clipping, 1207
ZSortObjects function, 1201

3-D drawing

See also BSP (Binary Space Partitioning) trees; Hidden surface removal;
Polygons, filling; Shading; 3-D animation.
backface removal
BSP tree rendering, 1160-1161
calculations, 955-957
motivation for, 954-955
and sign of dot product, 1140
solid cube rotation demo program, 957-961, 962-963, 964-966, 967
background surfaces, 1240
draw-buffers, and beam trees, 1187
and dynamic objects, 1100-1101
Gouraud shading, 1246-1250
lighting
  Gouraud shading, 1246-1250
  overlapping lights, 1247
  perspective correctness, 1248-1250
  rotational variance, 1249
  surface-based lighting, 1250-1256,
  1260-1262
  viewing variance, 1249
moving models in 3-D drawings,
  1212-1222
painter's algorithm, 1099, 1104-1105
perspective correctness problem,
  1276-1277
portals, and beam trees, 1188
projection
  dot products, 1141-1142
  overview, 937, 948
raycast, subdividing, and beam trees, 1187
reference materials, 934-935
rendering BSP trees
  clipping, 1158-1159
ClipWalls function, 1152-1155,
  1158-1159
DrawWallsBackToFront function, 1155-1156, 1160-1161
  overview, 1149
  reference materials, 1157
TransformVertices function, 1151-1152, 1158
UpdateViewPos function, 1151, 1157
UpdateWorld function,
  1156-1157, 1157
viewspace, transformation of
  objects to, 1158
wall orientation testing, 1160-1161
WallFacingViewer function, 1150-1151, 1161
span-based drawing, and beam
trees, 1187
transformation of objects, 935-936
triangle model drawing
  fast triangle drawing, 1263-1265
  overview, 1262-1263
  precision, 1265
  subdivision rasterization, 1266-1267,
  1267-1270
vertex-free surfaces, and beam
trees, 1187
visibility determination, 1099-1106
visible surface determination (VSD)
  beam trees, 1185-1189
  culling to frustum, 1181-1184
  overdraw problem, 1184-1185
  potentially visible set (PVS),
  precalculating, 1188-1189
3-D engine, Quake
BSP trees, 1276-1277
lighting, 1282-1283
model overview, 1276-1277
portals, 1279-1280
potentially visible set (PVS), 1278-1279
rasterization, 1282
world, drawing, 1280-1281
3-D math
cross products, 1139-1140
dot products
  calculating, 1135-1137
  calculating light intensity, 1137
  projection, 1141-1142
  rotation, 1143-1144
  sign of, 1140-1141
  of unit vectors, 1136
  of vectors, 1135-1136
matrix math
  assembly routines, 992, 996-999
  C-language implementations, 974-976
  normal vectors, calculating, 955-956
  rotation of 3-D objects, 938-939,
  943-944, 948
  transformation, optimized, 1172-1173, 1173-1174
vector length, 1135
3-D polygon rotation demo program
matrix multiplication functions, 943-944, 948
  overview, 939
  performance, 949
  polygon filling with clipping support, 940-943
  transformation and projection,
  944-945, 948
3-D solid cube rotation demo program
  basic implementation, 957-961, 962-963
  incremental transformations, 964-966
  object representation, 967
8-bit bus cycle-eater
  286 and 386 processors, 210
  8088 processor
effects on performance, 82
optimizing for, 83-85
overview, 79-82
and registers, 85

12-cube rotation demo program
limitations of, 986
optimizations in, 985-986
performance, 986
X-Sharp animation package, 972, 973-984, 984-985

16-bit checksum program
See also TCP/IP checksum program.
assembly implementation, 10-12, 17-18
C language implementation, 8-9, 15-16
overview, 8
redesigning, 9

16-color VGA modes
color paging, 628-629
DAC (Digital/Analog Converter), 626-628
palette RAM, 626
24-byte hi/lo function, 292-293
32-bit addressing modes, 256-258
32-bit division, 181-184, 1008
32-bit fixed-point arithmetic, optimizing, 1086-1089, 1090-1091, 1092-1093
32-bit instructions, optimizing, 1091
32-bit registers
See also Registers; VGA registers.
adding with LEA, 131
BSWAP instruction, 252
multiplying with LEA, 132-133
386 processor, 222
time vs. space tradeoff, 187
using as two 16-bit registers, 253-254

256-color modes
See also 320x400 256-color mode.
DAC settings, 629
mapping RGB model to, 1036, 1037-1038, 1039
resolution, 360x480 256-color mode, 619-620

386 processor
CMP instruction, 161, 306
code alignment, 215-218
cycle-eaters, 209-210
data alignment, 213-218
data transfer rates, 212
display adapter cycle-eater, 219-221
display memory wait states, 220
DRAM refresh cycle-eater, 219
effective address calculations, 129, 223-225
instruction fetching, 215-218
LEA vs. ADD instructions, 130
lookup tables, vs. rotating or shifting, 145-146
LOOP instruction vs. DEC/JNZ sequence, 139
memory access, performance, 223-225
new features, 221
POPF instruction, and interrupts, 226
protected mode, 208-209
stack pointer alignment, 218-219
system wait states, 210-212
320x240 256-color mode. See Mode X.
320x400 256-color mode
advantages of, 590-591
display memory organization, 591-593
line drawing, 600
page flipping demo program, 600-605
performance, 599-600
pixel drawing demo program, 593-598, 599-600

360x480 256-color mode
display memory, accessing, 621-622
Draw360x480Dot subroutine, 613-614
drawing speed, 618
horizontal resolution, 620
line drawing demo program, 615-618, 618-619
mode set routine (John Bridges), 609, 612, 620-621
on VGA clones, 610-611
Read360x480Dot subroutine, 614-615
256-color resolution, 619-620
vertical resolution, 619
386 native mode, 32-bit displacements, 187

Index
lookup tables, vs. rotating or shifting, 145-146

**LOOP** instruction vs. **DEC/JNZ** sequence, 139

memory access, performance, 223-225

**MUL** and **IMUL** instructions, 173-174

multiplication operations, increasing speed of, 173-174

new instructions and features, 222

Pentium code, running on, 411

protected mode, 208-209

rotation instructions, clock cycles, 185-186

system wait states, 210-212

32-bit addressing modes, 256-258

32-bit multiply and divide operations, 985

using 32-bit register as two 16-bit registers, 253-254

**XCHG** instruction, vs. **MOV**, 377, 832

386SX processor, 16-bit bus cycle-eater, 81

486 processor

AX register, setting to absolute value, 172

byte registers and lost cycles, 242-245

**CMP** instruction

operands, order of, 306

vs. **SCASW**, 161

copying bytes between registers, 172

and display adapter cycle-eater, 107

indexed addressing, 237-238

internal cache

effect on code timing, 246

optimization, 236

**LAHF** and **SAHF** instructions, 148

**LEA** instruction, vs. **ADD**, 130

**LODSB** instruction, 304

**LODSD** instruction, vs. **MOV/LEA** sequence, 171

lookup tables, vs. rotating or shifting, 145-146

**LOOP** instruction, vs. **DEC/JNZ** sequence, 139

**MOV** instruction, vs. **XCHG**, 377

n-bit vs. 1-bit shift and rotate instructions, 255-256

Pentium code, running on, 411

pipelining

address calculation, 238-240, 250

stack addressing, 241-242

rotation instructions, clock cycles, 185-186

stack-based variables, 184-184

32-bit addressing modes, 256-258

timing code, 245-246

using 32-bit register as two 16-bit registers, 253-254

**XCHG** instruction, vs. **MOV**, 377, 832

640x400 mode, mode set routine, 852-853

640x480 mode, page flipping, 836-837

8086 processor vs. 8088 processor, 79-81

8088 processor

**CMP** instruction, 161, 306

cycle-eaters

8-bit bus cycle-eater, 79-85

display adapter cycle-eater, 101-108

DRAM refresh cycle-eater, 95-99

overview, 78-79, 80

prefetch queue cycle-eater, 86-94

wait states, 99-101

display memory access, 220

effective address calculation options, 129

vs. 8086 processor, 79-81

**LAHF** and **SAHF** instructions, 148

**LEA** vs. **ADD**, 130

**LODSB** instruction, 304

lookup tables, vs. rotating or shifting, 145-146

**LOOP** instruction vs. **DEC/JNZ** sequence, 139

memory variables, size of, 83-85

stack-based variables, placement of, 184-184

8253 timer chip

and DRAM refresh, 95

reference material, 72

resetting, 43

system clock inaccuracies

long-period Zen timer, 53, 54

Zen timer, 43, 45-46, 48

timer 0

operation, 44

stopping, 54, 65

timer modes, 44, 45

timer operation, 43-45

undocumented features, 54, 65
Absolute value, setting AX register, 171
Abstraction, and optimization, 350-352, 345-346
Abutting span sorting, 1229-1230
AC (Attribute Controller), VGA addressing, 427-428
Color Select register, 628-629
Index register, 443, 555
Mode Control register, 575
Mode register
color paging, 628-629
256-color modes, 629
palette RAM registers, setting, 631-632
Pel Panning register, 574
registers, setting and reading, 583
screen blanking demo program, 556-557
Active edge table (AET), 744
Adapters, display. See Display adapter cycle-eater.
ADD instruction
   and Carry flag, 147-148
   vs. INC, 147-148, 219
   vs. LEA, 130, 170-171
AddDirtyRect function, 867-869
Addition, using LEA, 130, 131
AddObject function, 1001-1002
AddPolygonEdges function, 1232-1233, 1238
Addressable memory, protected mode, 221
Addressing modes
   486 processor
      indexed addressing, 237-238
      32-bit addressing modes, 256-258
   386 processor, 130-133, 222
   VGA, internal indexing, 427-428
Addressing pipeline penalty
   See also Pipeline stalls.
   486 processor, 238-240, 250
   Pentium processor, 400-403
AdvanceAET function
   complex polygons, 748-749
   monotone-vertical polygons, 769
AET (active edge table), 744
AGIs (Address Generation Interlocks), 400-403

See also Addressing pipeline penalty;
   Pipeline stalls.
Algorithms In C (book), 192, 196
Alignment
   Pentium processor
      non-alignment penalties, 376
      TCP/IP checksum program, 409
   REP STOS instruction, 735
   386 processor, 218
   286 processor
      code alignment, 215-218
      data alignment, 213-215
      stack pointer alignment, 218-219
   ALU and latch demo program, 453-457, 458-460
   ALUs (Arithmetic Logic Units), VGA
   ALU and latch demo program, 453-457, 458-460
   logical functions, 458
   operational modes, 458
   overview, 451-452
Ambient shading, 1023, 1025-1027
AND instruction, Pentium processor
   AGIs (Address Generation Interlocks), 401-402
   vs. TEST, 377
Animation
   See also Animation demo programs;
      Mode X; 3-D animation.
   apparent motion, 1064
   ball animation demo program, 431-441
   challenges in, 819-820
   on PCs, 795-796
   page flipping, flicker-free animation, 444-446
   speed, importance of, 1064
Animation demo programs
   Mode X animation, 924-925, 925-930
   page flipping animation
      assembly code, 825-830
      C code, 820-825
      split screen and page flipping, 830-837
   3-D polygon rotation
      matrix multiplication functions, 943-944, 948
      overview, 939
      performance, 949
polygon filling with clipping support, 940-943
transformation and projection, 944-945, 948
3-D solid cube rotation demo program
basic implementation, 957-961, 962-963
incremental transformations, 964-966
object representation, 967
Animation techniques
bit-plane animation
assembly implementation, 801-809, 810
limitations, 811-813
page flipping, 814
palette registers, 799-801
principles, 796-798
shearing, 813
dirty-rectangle animation
C implementation, 847-851, 863-869
description, 844-845
ordering rectangles, 873
overlapping rectangles, 872-873
vs. page flipping, 846, 862
performance, 873
system memory buffer size, 851
writing to display memory, 856-857
internal animation, 872
masked images, 871-872
Antialiasing, Wu’s algorithm, 776-779, 780-791, 791-792
Apparent motion, in animation, 1064
AppendRotationX function, 964, 975
AppendRotationY function, 964-965, 975
AppendRotationZ function, 965, 976
Appropriate technology, 775-776
Arithmetic flags. See Flags.
Arrays, sorting, 180-181
Aspect ratio, Mode X, 878
Assemblers
MASM (Microsoft Assembler), 187
optimizing assemblers, 71-72
TASM (Turbo Assembler), 71-72
Assembly language optimization
See also Clock cycles; Local optimization; Optimization.
data, placing limitations on, 274
instruction size vs. execution time, 90-92, 93
multi-bit rotations, 23-24
objectives, 28
optimizing instructions, 23-24
programmer’s responsibilities, 27-29
rearranging instructions, 418-419
reducing size of code, 416-418
stack addressing, 420
understanding data, importance of, 122
Assembly language programmers, vs. compilers, 154-156
Assembly language, transformation issues, 25-26
AT computer
display adapter cycle-eater, 107
286 processor, data transfer rates, 212
Attribute Controller, VGA. See AC (Attribute Controller), VGA.
Automatic variables, 184-185
AX register, setting to absolute value, 171

B
Backface culling. See Backface removal.
Backface removal
See also Hidden surface removal;
Visible surface determination.
BSP tree rendering, 1160-1161
calculations, 955-957
motivation for, 954-955
and sign of dot product, 1140
solid cube rotation demo program, 957-961, 962-963, 964-966, 967
Background surfaces, 1240
BackRotateVector function, 1203
Ball animation demo program, 431-441
Ball animation demo program, 431-441
Beam trees
improvement, attempts at, 1187-1188
overview, 1185
performance, 1186
potentially visible set (PVS),
precalculating, 1188-1189
Benchmarks, reliability of, 729
Biased perceptions, and optimization, 1080, 1085
Big endian format, 252
BIOS. See EGA BIOS; VGA BIOS.
Bit mask
bitmapped text demo program, 466-469, 470-471
and latches, 470
overview, 464-466
Bit Mask register
bit mask, controlling, 465
drawing solid text, 1040
setting inside a loop, 429
vs. write mode 3, 832, 844
BitMan, 1039-1041, 1042-1044
Bitmap organization, Mode X, 882-883
Bitmapped text
demo program using bit mask, 466-469, 470-471
reference material, 471
Bitmapped text demo program, 466-469, 470-471
Bitmaps
chunky, converting to planar, 504-505, 505-508
relocating, 516-517
temporary color effects, 509
Bit-plane animation
assembly implementation, 801-809, 810
limitations, 811-813
overview, 796
page flipping, 814
palette registers, 799-801
principles, 796-798
shearing, 813
“Black box” approach, and future of
programming, 725-726
Blocks. See Restartable blocks.
Borders (overscan), 555-556
BOUND instruction, 221
Boundary pixels, polygons
rules for selecting, 712
texture mapping, 1049-1052, 1065-1066, 1067
Bounding volumes, 1184
Boyer-Moore algorithm
assembly implementations, 271-274, 274-277
C language implementation, 269
overview, 263-265
performance, 266-268
test-bed program, 270
Branch prediction, Pentium processor,
377-378
Branching instructions
See also Branch prediction.
286 and 386 processors
non-word-alignment penalty, 216
and prefetch queue cycle-eater, 210
eliminating, 312-313
Pentium processor
branches within loops, 378
pairing in U-pipe, 405
x86 family CPUs, performance, 140
Bresenham’s line-drawing algorithm
basic algorithm
assembly implementation, 655-656, 671-677
C language implementation, 661-665, 665-671
description, 657-660
strengths and weaknesses, 660-661
run-length slice algorithm
assembly implementation, 698-704
C-language implementations, 688-692, 692-693
description, 683-684
implementation details, 685-687
integer-based implementation, 685-687
potential optimizations, 705
Bresenham’s run-length slice algorithm.
See Run-length slice algorithm.
Bridges, John
mode set routine, 360x480 256-color
mode, 609, 612, 620-621
256-color modes, undocumented, 879
Brute-force solutions, 193
BSP (Binary Space Partitioning) trees
2-D line representation, 1120
3-D rendering, 1162
beam trees
improvement, attempts at, 1187-1188
overview, 1185
performance, 1186
potentially visible set (PVS),
precalculating, 1188-1189
BSP compiler
BuildBSPTree function, 1125-1127
SelectBSPTree function, 1124-1125
BuildBSPTree function, 1125-1127
building, 1101-1104
BuildTree function, 1112
data recursion vs. code recursion,
1108-1113
description, 1098-1099, 1119
and dynamic objects, 1100-1101
Index 1305
edge sorting for hidden surface removal, 1220, 1226
inorder traversal, 1107-1113
leaves, storing polygons in, 1181
multiple BSP trees, sorting, 1227
optimizations, 1128-1129
performance, 1100, 1111-1113
potentially visible set (PVS)
  precalculating, 1188-1189
  world, drawing, 1280-1281
reference materials, 1114
rendering recursively
  backface removal, 1160-1161
  clipping, 1158-1159
ClipWalls function, 1152-1155
  1158-1159
DrawWallsBackToFront function, 1155-1156
  overview, 1149
  reference materials, 1157
TransformVertices function, 1151-1152, 1158
UpdateViewPos function, 1151, 1157
UpdateWorld function, 1156-1157, 1157
viewspace, transformation of objects to, 1158
wall orientation testing, 1160-1161
WallFacingViewer function, 1150-1151, 1161
SelectBSPTree function, 1124-1125
  splitting heuristic, 1128-1129
3-D engine, Quake
  overview, 1276-1277
  potentially visible set (PVS)
    management, 1278-1279
visible surface determination (VSD)
  beam trees, 1185-1189
  culling to frustum, 1181-1184
  overdraw problem, 1184-1185
  painter's algorithm, 1099-1106
  polygon culling, 1181-1184
  PVS, precalculating, 1188-1189
WalkBSPTree function, 1106
WalkTree function, 1109-1110
BSP compiler
BuildBSPTree function, 1125-1127
  overview, 1123
SelectBSPTree function, 1124-1125
BSP models, Quake 3-D engine, 1284
BSWAP instruction, 486 processor
  32-bit registers, using as two 16-bit registers, 253-254
  rotating pixel bits, 252
Bubble sort, 755
Buffer-filling routine, optimizations
  rearranging instructions, 418-419
  reducing size of code, 416-418
  stack addressing, 420
Buffers, internal
  in 16-bit checksum program, 15-16
  in search engine, 114-115
BuildBSPTree function, 1125-1127
BuildGET function, 768-769
BuildGETStructure function, 747-748
BuildMaps function, 353-355
BuildTree function, 1112
Bus access
  8088 processor, 81, 99-101
  Pentium processor, 377
Byte registers, 486 processor, 242-245
Byte-OUT instruction, 429
Byte-per-pixel mode. See Mode X.

C library functions
getc() function, 12, 14
memcmp() function, 116
memcpy() function, 116
memcpy() function, 1147-1148
memset() function, 727
optimization, 15
read() function, 12, 121
strstr() function, 115
Cache, internal. See Internal cache.
Cache lines, Pentium processor, 374
Calculations, redundant, and optimization, 682-683
Calculus and Analytic Geometry
  (book), 1135
CALL instruction
  486 processor, 241-242
  Pentium processor, 404
Carmack, John
  and id Software, 1118
overdraw, 1184-1186
subdivision rasterization, 1266-1267, 1267-1270
Carry flag
   DEC instruction, 148
   INC vs. ADD instructions, 147-148
   LOOP instruction, 148
   rotating bits through, 185
   in word count program (David Stafford), 317-319
Cats, shipping via air freight, 697-698
Cellmap class, 325-329, 333-335, 341-345
Cellmap wrapping, Game of Life, 331-332, 333-335, 336, 337-338
Cell_state method, 327, 334, 344
CGA (Color/Graphics Adapter)
   display adapter cycle-eater, 104
   VGA compatibility with, 430
Challenges
   Game of Life
      rules, 346, 350
      3-cell-per-word implementation
         (David Stafford), 351-352, 353-363, 365-365
   ScanBuffer routine, 305, 307-319
Change list, in Game of Life, 363-366
Chaplin, Michael, 776
Character/attribute map, VGA mode 3, 517
Chartreuse moose story, 399
Checksum programs. See 16-bit checksum program; TCP/IP checksum program.
Chunky bitmap conversion demo program, 505-508
Chunky bitmaps, converting to planar, 504-505, 505-508
Circular linked lists, 288-292
Clear_cell method, 327, 334, 343
ClearEdgeLists function, 1236-1237
Clements, Willem, 313-315
Client-server architecture, and QuakeWorld, 1291
Clipping
   See also Hidden surface removal (HSR); Visible surface determination (VSD).
      arithmetic imprecision, handling, 1240
      in BSP tree rendering, 1158-1159
      line segments, clipping to planes, 1195-1197
      masked copying, Mode X, 923
      overview, 1195
polygon clipping
   BackRotateVector function, 1203
      clipping to frustum, 1200, 1201-1206, 1206-1207
   ClipToFrustum function, 1204
   ClipToPlane function, 1199
      optimization, 1207
      overview, 1197-1200
   PolyFacesViewer function, 1203
   ProjectPolygon function, 1201
   SetUpFrustum function, 1204
   SetWorldspace function, 1204
   TransformPoint function, 1203
   TransformPolygon function, 1203
   UpdateViewPos function, 1202
   UpdateWorld function, 1205
      viewspace clipping, 1207
   ZSortObjects function, 1201
ClipToFrustum function, 1204
ClipToPlane function, 1199
Clock cycles
   See also Cycle-eaters.
      address calculation pipeline, 238-240
      branch prediction, 377-378
      byte registers and lost cycles, 242-245
      cross product floating point optimization, 1171, 1172
      and data alignment, 213-215
      data transfer rates, 81, 82
      dot product floating point optimization, 1170
      dual-pipe execution, 405
      effective address calculations
         286 and 386 processors, 223-225
         Pentium processor, 375-376
      8088 processor
         data transfer rates, 81, 82
         memory access, 82, 83-85
      floating point instructions, 1167-1170
      486 processor
         address calculation pipeline, 238-240, 250
         byte registers and lost cycles, 242-245
         indexed addressing, 237-238
         stack addressing, 241-242
         32-bit addressing modes, 256-258
   FXCH instruction, 1170
      indexed addressing, 237-238
instruction execution times, 86-93
lockstep execution, 390-394, 400-403
matrix transformation optimization, 1173
memory access, 82, 83-85
non-alignment penalties, 376
non-word-alignment penalty, 217
1/z value of planes, calculating, 1221
OUT instructions, 843, 1082-1083
Pentium processor
branch prediction, 377-378
cross product floating point optimization, 1171, 1172
dot product floating point optimization, 1170
effective address calculations, 375-376
floating point instructions, 1167-1168
FXCH instruction, 1170
initial pipe, effect of, 405
lockstep execution, 390-394, 400-403
matrix transformation optimization, 1173
non-alignment penalties, 376
pipelining, 1168-1170
prefix bytes, 376, 395, 407
prefix bytes, 376, 395, 407
vs. program size, 28
projection, floating point optimization, 1174
stack addressing, 241-242
string instructions, 82
system wait states, 211
32-bit addressing modes, 256-258
386 processor, effective address calculation, 223-225
286 processor
effective address calculation, 223-225
system wait states, 211
CMP instruction
operands, order of, 306
vs. SCASW, 161
CMPXCHG8B instruction, Pentium processor, 378
Code alignment
386 processor, 218
286 processor, 215-218
Code generator, for Game of Life (David Stafford), 351-352, 353-363, 363-365
Code recursion
vs. data recursion, 1108-1110
Euclid’s algorithm, 198-199
Collision detection demo program, 531-534
Color
adapter-dependent mapping, 1036
color perception research, 1035
reflective vs. emissive, 1035
Color Compare register, 531
Color cycling
bit-by-bit loading of DAC, 650-651
demo program, 643, 644-648, 648-649
interleaved loading of DAC, 649-650
loading DAC, 640-643
overview, 639-640
using page flipping, 650
using subset of DAC, 649
Color cycling demo program, 643, 644-648, 648-649
Color Don’t Care register, 534
Color Don’t Care register demo program, 535-537, 535
Color mapping demo program, EGA, 551-555
Color models. See RGB (red, green, blue) color model.
Color paging, 628-629
Color path, VGA
color paging, 628-629
DAC (Digital/Analog Converter), 626-628
palette RAM, 626
Color planes. See Planes, VGA.
Color Select register, 628-629
Color selection
EGA
overscan, 555-556
palette registers, 548-551, 551-555
screen blanking, 556-557
VGA, 557
ColorBarsUp subroutine, 604
Color-forcing demo program, 474-476
Color-patterned lines demo program, 509-515
Compiled DLLs, Quake 2, 1293
Compiler-based optimization
cautions for use of, 9
data recursion vs. code recursion, 1112-1113
in FindIDAaverage function, 159
Compilers
vs. assembly language programmers, 154-155
avoiding thinking like, 152, 154-155
bitblt compiler for Game of Life
(David Stafford), 351-352, 353-363, 363-365
handling of segments, 154
Complex polygons
defined, 710, 742
edges, keeping track of, 742-744, 753, 755, 756
polygon-filling programs, 745-752, 754

Computational Geometry, An
Introduction (book), 759-760
Computers
Graphics: Principles and
Practice (book), 660, 934, 1121
Computer Graphics (book), 1135, 1157
ConcatXforms function
assembly implementation, 997-999, 1019-1022
C-language implementation, 944, 976
CONSTANT_TO_INDEXED_REGISTER macro, 594
Coordinate systems
left-handed, 1140
right-handed, 935-937
Copy cells method, 327, 333
CopyDirtyRectangles function, 850
CopyDirtyRectangleToScreen function, 866-867
Copying
bytes between registers, 172
pixels, using latches (Mode X), 905-907, 908, 909-911
CopyRect subroutine, 871
CopyScreenToScreenMaskedX subroutine, 918, 919-921
CopyScreenToScreenX subroutine, 905-907, 908
CopySystemToScreenMaskedX subroutine, 916-918
CopySystemToScreenX subroutine, 908, 909-911
CosSin subroutine, 994-996, 999, 1013-1015
Count_neighbors method, 334-335
CPU reads from VGA memory, 526
CPUID instruction, Pentium
processor, 378
CreateAlignedMaskedImage function, 922-923
Cross products
calculating, 955-956, 1139-1140
floating point optimization, 1171, 1172
CRT Controller, VGA. See CRTC (CRT Controller), VGA.
CRTC (CRT Controller), VGA
addressing, 427-428
Line Compare register, 565
Overflow register, 565
shearing, 813-814
start address registers, setting, 583
Cycle-eaters
286 and 386 processors
data alignment cycle-eater,
213-215, 218
display adapter cycle-eater, 219-221
DRAM refresh cycle-eater, 219
overview, 209-210
prefetch queue cycle-eater, 211-212
system wait states, 210-212
data alignment cycle-eater
386 processor, 218
286 processor, 213-215
display adapter cycle-eater
286 and 386 processors, 219-221
8088 processor, 101-108
DRAM refresh cycle-eater
286 and 386 processors, 219
8088 processor, 95-99, 108
8-bit bus cycle-eater, 79-85, 108
8088 processor
display adapter cycle-eater, 101-108
DRAM refresh cycle-eater, 95-99, 108
8-bit bus cycle-eater, 79-85, 108
prefetch queue cycle-eater, 86-94, 108
wait states, 99-101
overview
286 and 386 processors, 209-210
8088 processor, 78-79, 80
prefetch queue cycle-eater
286 and 386 processors, 211-212
8088 processor, 86-94, 108
system wait states, 210-212
wait states, 99-101
Cycles. See Clock cycles; Cycle-eaters.
DAC (Digital/Analog Converter)
color cycling
- bit-by-bit loading, 650-651
color cycling demo program, 643, 644-648, 648-649
- interleaved loading, 649-650
- problems, 640-643
using subset of, 649
Data register, 642-643
index wrapping, 651
loading
- bit-by-bit loading, 650-651
directly, 642-643
- interleaved loading, 649-650
- via VGA BIOS, 641-642, 648
and Write Index register, 642-643, 651
Mask register, blanking screen, 651
Read Index register, 651-652
reading, 651-652
setting registers, 630, 631-632
in VGA color path, 626-628
Write Index register
- DAC index wrapping, 651
loading DAC, 642-643
DAC registers demo program, 632-635
Data alignment cycle-eater
- 386 processor, 218
- 286 processor, 213-215
Data bus, 8-bit
See also 8-bit bus cycle-eater.
Data manipulation instructions, and flags, 147
Data recursion
- vs. code recursion, 1108
- Euclid's algorithm, 200
- inorder tree traversal, 1108, 1109-1110, 1110
Data register, loading DAC, 642-643
DAC Rotate register
- barrel shifter, controlling, 463
- vs. CPU-based rotations, 489
- effect on ALUs, 452
Data rotation, VGA
- barrel shifter, 463-464
- bit mask, 464-471
- CPU vs. Data Rotate register, 489
Data transfer rates
display adapters, 220
8088 processor vs. 8086 processor, 81, 82
286 processor, 212
DDA (digital differential analyzer)
texture mapping
assembly implementation,
1069-1073, 1074
C implementation, 1053-1058
- disadvantages, 1052-1053, 1059
DrawTexturedPolygon, 1055-1056
- hardware dependence, 1053
- multiple adjacent polygons, 1068
- optimized implementation, 1069-1073, 1074
- orientation independence,
1065-1067, 1067
- performance, 1074
ScanOutLine function, 1058-1059, 1067, 1069-1073, 1074
SetUpEdge function, 1057-1058
StepEdge function, 1056-1057
- techniques, 1048-1051
DDJ Essential Books on Graphics Programming (CD), 1157
DEC instruction
- and Carry flag, 148
- memory accesses, 83
- vs. SUB, 219
DEC/JNZ sequence, 139
Delay sequences
loading palette RAM or DAC registers, 632
VGA programming, 558
DeleteNodeAfter function, 284
Depth sorting of nonconvex objects,
1000, 1001-1002
Diffuse shading, 1023-1025, 1025-1027
Digital differential analyzer. See DDA (digital differential analyzer).
Direct far jumps, 186
Direct memory access. See DMA.
Directed lighting, and shading, 1023, 1028
Directives
EVEN, 214
NOSMART, 72
Dirty-rectangle animation
demo program, C implementation,
847-851, 863-869
description, 844-845
ordering rectangles, 873
overlapping rectangles, 872-873
vs. page flipping, 846, 862
performance, 873
system memory buffer size, 851
writing to display memory, 856-857
Disk caches, 19
Display adapter cycle-eater
286 and 386 processors, 219-221
data transfer rates, 220
8088 processor
  graphics routines, impact on, 106
  optimizing for, 107
  overview, 101-104
  performance, impact on, 104
  read/write/modify operations, 107
  wait states, 99-101
Display memory
See also Bit mask; Display memory access.
  Mode X
  copying between memory locations, 905-907, 908
  copying from system memory, 908, 909-911
  masked copy from system memory, 916-918, 916
  masked copying between locations, 918-919, 919-921
  memory allocation, 903-904
  running code from, 104
  start address, changing, 857
VGA
  access times, 842-844
  360x480 256-color mode, 621-622
  320 x 400 256-color mode, 591-593, 605
Display memory access
See also Display memory;
Memory access.
display adapter cycle-eater, 101-103, 105, 107
and string instructions, 107
VGA access times, 842-844
wait states, 101-103, 220, 733
Display memory planes.
See Planes, VGA.
DIV instruction, 32-bit division, 1
81-184, 1008
Divide By Zero interrupt, 181
Divide-by-N timer mode, 45
Division, 32-bit, 181-184, 1008
DMA (direct memory access), and DRAM refresh, 95
"Don't care" planes, 535
DOS function calls
  overhead, 9
  and restartable blocks, 123
Dot products
  calculating, 1135-1137
  calculating light intensity, 1137
  floating point optimization, 1170, 1171
  line segments, clipping to planes, 1196-1197
  projection, 1141-1142
  rotation, 1143-1144
  sign of, 1140-1141
  of unit vectors, 1136
  of vectors, 1135-1136
Double-DDA texture mapping. See DDA
digital differential analyzer)
texture mapping.
D_PolysetRecursiveTriangle function, 1267-1270
Dr. Dobbs Journal, 1190
DRAM (dynamic RAM) refresh cycle-eater
286 and 386 processors, 219
8088 processor
  impact on performance, 97-98
  optimizing for, 98-99
  overview, 95-97
  vs. wait states, 100
  and 8253 timer chip, 95
  and Zen timer, 99
Draw360x480Dot subroutine, 613-614
DrawBackground function, 928
DrawBuffers, and beam trees, 1187
DrawBumperList function, 823
DrawEntities function, 849, 866
DrawGridCross subroutine, 808
DrawGridVert subroutine, 808-809
DrawHorizontalLineList function
  monotone-vertical polygons, filling, 765
  non-overlapping convex polygon assembly implementation, 734
  C implementation, 717, 720-721
  using memset() function, 727, 729
DrawHorizontallineList subroutine, 941-943
DrawHorizontallineSeg function
assembly implementation, 754
C implementation, 750-751
DrawHorizontalRun function, 692
DrawImage subroutine, 828

Drawing
See also Line-drawing algorithms;
Lines; 3-D drawing.
fill patterns, using latches, 453
pixel drawing
EVGADot function, 661-662, 669-670
optimization, 1074, 1086
painter's algorithm and overdraw problem, 1184
single-color drawing with write mode 3, 831-832
speeding up, 727-729
text
bitmapped text using bit mask, 466-469, 470-471
bitmapped text using write mode 3, 484-489, 489-490, 490-496
solid text using latches, 1039-1041, 1042-1044
using write mode 0, 832-833

DrawLine function, 785
DrawMasked subroutine, 870
DrawObject subroutine, 809-810
Draw_pixel function, 328, 330
DrawPObj ect function, 978-979, 1025-1027
DrawRect subroutine, 826-827
DrawSpans function, 1236
DrawSplitScreen function, 824
DrawTextString subroutine, 1043-1044
DrawTexturedPolygon function, 1055-1056
DrawVerticalRun function, 692
DrawVisibleFaces function, 961

Dynamic palette adjustment, 1039
Dynamic RAM. See DRAM (dynamic RAM) refresh.

EA (effective address) calculations
286 and 386 processors, 223-225
8088 processor, 129
486 processor
address calculation pipeline, 238-240
stack addressing, 241-242
Pentium processor, 375-376
320x400 256-color mode, 599-600

EBP register, 257

Edge triggered devices, 316
Edges vs. spans, sorted span hidden surface removal, 1215-1220
EGA BIOS, video function 10H, 550-551, 555
EGA (Enhanced Graphics Adapter)
color mapping, 548-551, 551-555
and display adapter cycle-eater, 104-108
mode 10H, 515-517, 518-521
palette registers, 549-550
registers, and high-level languages, 548
screens, capturing and restoring, 541-542, 543-547, 547-548
split screens
EGA bug, 573-574
horizontal panning, 574-575, 575-582, 583
overview, 563-565
registers, setting, 573
safety of, 585
split screen demo program, 565, 566-572, 572
text mode, 584
turning on and off, 565
8-bit bus cycle-eater
286 and 386 processors, 210
8088 processor
effects on performance, 82
optimizing for, 83-85
overview, 79-82
and registers, 85
8086 processor vs. 8088 processor, 79-81
8088 processor
CMP instruction, 161, 306
cycle-eaters
8-bit bus cycle-eater, 79-85
display adapter cycle-eater, 101-108
DRAM refresh cycle-eater, 95-99
overview, 78-79, 80
prefetch queue cycle-eater, 86-94
wait states, 99-101
display memory access, 220
vs. 8086 processor, 79-81
effective address calculation options, 129
LAHF and SAHF instructions, 148
LEA vs. ADD, 130
LODSB instruction, 304
lookup tables, vs. rotating or shifting,
145-146
LOOP instruction vs. DEC/JNZ
sequence, 139
memory variables, size of, 83-85
stack-based variables, placement of,
184-184
8253 timer chip
and DRAM refresh, 95
reference material, 72
resetting, 43
system clock inaccuracies
long-period Zen timer, 53, 54
Zen timer, 43, 45-46, 48
timer 0
operation, 44
stopping, 54, 65
timer modes, 44, 45
timer operation, 43-45
undocumented features, 54, 65
Emissive color, vs. reflective color, 1035
Enable Set/Reset register
setting drawing color, 666
specifying plane, 474
EnableSplitScreen function, 824
ENTER instruction
486 processor, 241-242
Pentium processor, 377
286 processor, 221
ENTER_display_mode function, 328, 362
Entities, Quake 3-D engine
BSP models, 1284
particles, 1287
polygon models, 1285-1286
sprites, 1287
subdivision rasterization, 1286
z-buffering, 1285-1286
EraseEntities function, 850, 867
Error accumulation, Wu antialiasing
algorithm, 778-779, 792
EU (Execution Unit)
286 and 386 processors
instruction execution times, 223-225
and prefetch queue, 210
8088 processor
8-bit bus cycle-eater, 80
prefetch queue cycle-eater, 86
wait states, 101
Euclid's algorithm
algorithm, 197
optimized assembly implementation,
200-202
recursive implementations, 198, 200
EVEN directive, 214
EVGADot function, 661-662, 669-670
EVGALine function
Bresenham's algorithm
assembly implementation, 671,
675-677
C-language implementation, 664-665, 665-668, 670-671
360x480 256-color mode line drawing program, 616-617
Execution times. See Clock cycles;
Instruction execution time.
Exit_display_mode function,
328, 329, 362

F

FADD instruction, Pentium processor,
1167-1170
Far jumps, to absolute addresses, 186-187
FDIV instruction, Pentium processor,
1167-1170
Fetch time
See also Instruction fetching.
286 and 386 processors, 210, 211
8088 processor, 86-93
Files
reading from
getc() function, 12, 14
read() function, 12
restartable blocks, 16
text, searching for. See Search engine.
Fill patterns, drawing using latches, 453
FillConvexPolygon function, 714-716, 720-721
FillMonotoneVerticalPolygon
function, 763-764
FillPolygon function
complex polygons, 746
monotone-vertical polygons, 767
FillRect subroutine, 869-870
FillRectangleX subroutine
document parallel processing, 888-891, 891-893
pixel-by-pixel plane selection, 885-887
plane-by-plane processing, 887-889
FillPatternX subroutine
assembly implementations
based on compiler optimization, 160
data structure reorganization, 163, 165-166
unrolled loop, 161, 162
C language implementation, 158
compiler optimization, 159
FindNodeBeforeValue function, 289
FindNodeBeforeValueNotLess
function, 286, 287
FindString function
Boyé-Moore algorithm, 269, 271-274, 274-277
overview, 175
scan-on-first-character approach, 176
scan-on-specified-character approach, 178
FixedDiv subroutine, 982, 993, 1010-1012
FIXED_MUL macro, 1016-1017
FixedMul subroutine, 981, 993-994, 1009-1010
Fixed-point arithmetic
vs. floating point, 985, 1206
vs. integer arithmetic, 730, 1065
32-bit fixed-point arithmetic, 1086-1089, 1090-1091, 1092-1093
Flags
and BSWAP instruction, 254
Carry flag, 147-148, 185, 317-319
INC vs. ADD, 147-148
and LOOP instruction, 148
and NOT instruction, 146-147
FLD instruction, Pentium processor, 1167-1170
Floating point optimization
clock cycles, core instructions,
1167-1168
cross product optimization, 1171, 1172
dot product optimization, 1170, 1171
FXCH instruction, 1169-1170
interleaved instructions, 1169-1170
matrix transformation optimization,
1172-1173, 1173-1174
overview, 1167-1170
pipelining, 1168-1170
projection to screen space, 1174
rounding control, 1174-1175
Floating-point calculations
vs. fixed-point calculations, 985, 1206
vs. integer calculations, 730
FMUL instruction
486 processor, 236
Pentium processor, 1167-1170
486 processor
AX register, setting to absolute value, 172
byte registers and lost cycles, 242-245
CMP instruction
operands, order of, 306
vs. SCASW, 161
copying bytes between registers, 172
and display adapter cycle-eater, 107
indexed addressing, 237-238
internal cache
effect on code timing, 246
optimization, 236
LAHF and SAHF instructions, 148
LEA instruction, vs. ADD, 131
LODSB instruction, 304
LODSD instruction, vs. MOV/LEA
sequence, 171
lookup tables, vs. rotating or shifting, 145-146

**LOOP** instruction, vs. **DEC/JNZ** sequence, 139

**MOV** instruction, vs. **XCHG**, 377

n-bit vs. 1-bit shift and rotate instructions, 255-256

Pentium code, running on, 411

pipelining
  address calculation, 238-240, 250
  stack addressing, 241-242

rotation instructions, clock cycles, 185-186

stack-based variables, 184-184

32-bit addressing modes, 256-258

timing code, 245-246

using 32-bit register as two 16-bit registers, 253-254

**XCHG** instruction, vs. **MOV**, 377, 832

FPU, Pentium processor
  clock cycles, core instructions, 1167-1168
  cross product optimization, 1171, 1172
  dot product optimization, 1170, 1171

**FXCH** instruction, 1169-1170

interleaved instructions, 1169-1170

matrix transformation optimization, 1172-1173, 1173-1174

overview, 1167-1170

pipelining, 1168-1170

projection to screen space, 1174

rounding control, 1174-1175

Frustum, clipping to, 1200, 1201-1206, 1206-1207

**FST** instruction, Pentium processor, 1167-1170

**FSUB** instruction, Pentium processor, 1167-1170

Function 13H, VGA BIOS, 459

Function calls, performance, 153


**FXCH** instruction, Pentium processor, 1169-1170

---

**G**

Game of Life
  abstraction and performance, 330-332, 345-346
  byte-per-cell implementation, 339-340, 341-345

C++ implementation
  basic, 324, 325-328
  optimized, 336, 337-338

cellmap-wrapped implementation, 331-332, 333-335, 336, 337-338

challenge to readers
  rules, 346, 350
  3-cell-per-word implementation
    (David Stafford), 351-352, 353-363, 365-366

change list, 363-366

performance analysis, 329-330, 332, 338, 340, 350

re-examining problem, 338-339, 363

rules, 324

3-cell-per-word implementation
  discussion, 363-365
  listing, 352-363

overview, 351-352

GC (Graphics Controller), VGA
  addressing, 427-428

architecture
  ALUs, 451-452
  barrel shifter, 463-464
  bit mask, 464-471
  latches, 452-453
  set/reset circuitry, 471-479

Bit Mask register
  bit mask, controlling, 465
  drawing solid text, 1040
  setting inside a loop, 429
  vs. write mode 3, 832, 844

Color Compare register, 531

Data Rotate register
  barrel shifter, controlling, 463
  vs. CPU-based rotations, 489
  effect on ALUs, 452

Enable Set/Reset register
  setting drawing color, 666
  specifying plane, 474

Graphics Mode register
  read mode 0, selecting, 525
  read mode 1, selecting, 531

Read Map register
  plane, selecting, for CPU reads, 526
  planes, specifying to be read, 542

Set/Reset register, 666
Gcd() function
  brute-force approach, 195
  Euclid's algorithm
    code recursive approach, 198
    data recursion approach, 200
    subtraction approach, 196
GCD (Greatest Common Denominator)
  problem
    brute-force approach, 193-196
    Euclid's algorithm, 197-200
    subtraction approach, 196-197
Gcd_recur() function, 199
Generality, vs. performance, 335
Gerold, David, 298
GET (global edge table), 744
Gec() function
  overhead, 14
  vs. read() function, 12
GetNextKey subroutine, 598, 605
GetUpAndDown function, 355
Global edge table (GET), 744
GLQuake, 1288-1290
Gouraud shading
  overview, 1246-1247
  perspective correction, 1248-1250
  problems with, 1247-1250
Graphics cards, and surface caching, 1261-1262
Graphics Controller, VGA. See GC
  (Graphics Controller), VGA.
Graphics Mode register
  read mode 0, selecting, 525
  read mode 1, selecting, 531
Graphics screen capture demo program, 543-545
Graphics screen restore demo program, 545-547
Graphics-to-text demo program, 518-521
Great Buffalo Sauna Fiasco, 137-138
GUIs, and future of programming profession, 725-726

Hardware dependence, DDA (digital differential analyzer) texture mapping, 1053
Hecker, Chris
  texture mapping insight, 1083
  underlying functionality of different approaches, 1189
Heinlein, Robert A., 1079-1080
Herbert, Frank, 1193
HGC (Hercules Graphics Card), 104
Hidden surface removal (HSR)
  backface removal, 954-957
  depth sorting, 1000, 1001-1002
  sorted spans approach
    abutting span sorting, 1229-1230
    AddPolygonEdges function, 1232-1233, 1238
    BSP order vs. 1/z order, 1220, 1226
    ClearEdgeLists function, 1236-1237
    DrawSpans function, 1236
      edge sorting, 1220-1222
      edges vs. spans, 1215-1220
      independent span sorting, 1230,
        1231-1238, 1239-1241
      intersecting span sorting, 1228-1229
      1/z sorting, 1220-1222, 1227-1231,
        1231-1238, 1239-1241
    overview, 1214-1215
    PolyFacesViewer function, 1232
      rotation instructions, clock cycles,
        185-186
    ScanEdges function, 1234-1236,
      1238-1239
    UpdateWorld function, 1237-1238
      High school graduates in Hawaii, 991-992
      Horizontal Pel Panning register, 442
      Horizontal resolution, 360x480 256-color mode, 620
      Horizontal smooth panning. See Panning.

id Software, 1118, 1190
Ideas, selling, 1193-1194
Illowsky, Dan, 187, 315
Image precedence. See
  Bit-plane animation.
IMUL instruction
  486 processor, 236
  or 386 processor, 173-174
INC instruction
  vs. ADD, 147-148, 219
  and Carry flag, 147-148
Incremental transformations of 3-D objects, 964
Independent span sorting
  AddPolygonEdges function, 1232-1233, 1238
  ClearEdgeLists function, 1236-1237
  DrawSpans function, 1236
    overview, 1230
  PolyFacesViewer function, 1232
  ScanEdges function, 1234-1236, 1238-1239
    texture mapping, 1238
  UpdateWorld function, 1237-1238
Index registers, VGA
  AC Index register, 443
    overview, 427-428
Indexing addressing, 237-238
Indirect far jumps, 186
Information, sharing, 1190, 1194
InitCellMap function, 361
InitializeCubes function, 980-981
InitializeFixedPoint function, 977
InitializeObjectList function, 1001
InitializePalette function, 1037
InitLinkedList function, 289
Inorder tree traversal
  code recursion vs. data recursion, 1107-1108
  data recursive implementation, 1108, 1109-1110, 1110
    performance, 1111-1113
INS instruction, 221
InsertNodeSorted assembly routine, 290
InsertNodeSorted function, 289
Instruction execution times
  See also Clock cycles; Zen timer.
  DRAM refresh cycle-eater, 97, 99
  8-bit bus cycle-eater, 82-85
    estimating, 93
  and instruction fetching, 225
  vs. instruction size, 90-92, 93, 211
  memory-addressing vs. register-only instructions, 223-225
  prefetch queue cycle-eater, 86-93
Instruction fetching
  See also Prefetch queue cycle-eater.
    code alignment, 215-218
    8088 processor, 86-93
    and instruction execution times, 225
Pentium processor, 374
    and system wait states, 211
286 processor, 215-218
    and wait states, 101
Instruction size, 32-bit addressing
    modes, 257
Instructions, assembly language
    optimizing, 23-24
Pentium processor
    pairable instructions, 388, 390-394
    V-pipe-capable instructions, 386-387
    size vs. execution time, 90-92, 93
Integer calculations, vs. fixed-point, 730, 1065
Integers, sorting, 180-181
Interleaved color cycling, 649-650
Interleaved operations, Pentium processor
  FXCH instruction and floating point operations, 1169-1170
  matrix transformation, 1172-1173,
    1173-1174
    overview, 394-395
  TCP/IP checksum program, 408
Internal animation, 872
Internal buffering
  See also Restartable blocks.
    in 16-bit checksum program, 15-16
    in search engine, 114-115
Internal cache
  486 processor
    effect on optimization, 236
    timing code, 246
Pentium processor
    instruction fetching, 374
    organization, 374-375
    paired instructions, 391, 396
Internal indexing, VGA, 427-429
Internet support
  Quake 2, 1293
  QuakeWorld, 1291
Interrupts
  DAC, loading, 643, 648
  Divide By Zero interrupt, 181
  and IRET instruction, 227
  and long-period Zen timer, 53, 66
  and page flipping, 446
  and POPF instruction, 226
  and Zen timer, 43, 45-46
Intersecting lines, 1121-1123
Intersecting span sorting, 1228-1229
Intuitive leaps, 1098
IRET instruction, vs. POPF instruction, 226-231
IRQ0 interrupts, and Zen timer, 45
IS_VGA equate, 572, 575

Jet Propulsion Lab, color perception research, 1035
JMP $+2$ instructions, 558, 632
JMP DWORD PTR instruction, 186-187
Jumps, to absolute addresses, 186-187

Kennedy, John, 171-172
Kent, Jim
   dynamic palette adjustment, 1039
   monotone-vertical polygons, filling, 760-761
Kissing, learning to, 281-282
Kitchen floor story, 261-262
Klerings, Peter, 350
Knuth, Donald, 323

LAHF instruction, 148
Large code model
   linking Zen timer, 71
   optimizing assemblers, 71-72
Latches
   and bit mask, 470
   and Color Don't Care register, 535-537, 535
   and CPU reads, 530
   drawing solid text, 1039-1041, 1042-1044
Mode X
   copying pixels, 905-907, 908, 909-911
   loading, with double copying process, 903
masked copying, 918-919, 919-921, 922-923
pattern fills, 899, 900-903, 903-904
overview, 452-453, 897-898
Latency, in QuakeWorld, 1291-1292
LEA instruction
   vs. ADD, 130, 170-171
   multiplication operations, 132-133, 172, 375-376
   32-bit registers
      addition, 131
      multiplication, 132-133
LEAVE instruction
   486 processor, 241-242
   Pentium processor, 377
   286 processor, 221
Level performance, 1213-1214
Life, Game of. See Game of Life.
Lighting
   See also Shading.
   Gouraud shading
      overview, 1246-1247
      perspective correction, 1248-1250
      problems with, 1247-1250
      intensity, calculating, 1137
      overlapping lights, 1247
      perspective correctness, 1248-1250
      in Quake 3-D engine, 1282-1283
      rotational variance, 1249
      surface-based lighting
         description, 1250-1251
         mipmapping, 1254-1255
         performance, 1251-1253
         surface caching, 1253-1256, 1260-1262
         two-pass approach, 1262
         viewing variance, 1249
Limits, transcending, in creative design, 1179-1180
Lindley, Bill, 854-855
LINE1 macro, 672-674
LINE2 macro, 674-675
Line Compare register, 565
Line segments
   clipping to planes, 1195-1197
   representation, 1195, 1196
Linear addressing, VGA, 430
Linear-time sorting, 1099
LineDraw function
   assembly implementation, 699-704,
C-language implementation, 688-691
Line-drawing algorithms
accumulated pixels approach (Jim MackrAz), 678
Bresenham's algorithms
basic line-drawing algorithm, 655-661, 661-665, 665-671, 671-677
run-length slice algorithm, 683-693, 698-704, 705
characteristics of, 656-657
run-length slice algorithm, 683-693, 698-704, 705
Wu antialiasing algorithm, 776-779,
Line-drawing demo program, 615-618, 618-619
LinInteCersectPlane function, 1142-1143
Lines
  drawing
  See also Line-drawing algorithms.
color-patterned lines demo program, 509-515
  320 See also Restartable blocks. 400
  256-color mode, 600
write mode 2, 509
intersecting, 1121-1123
parametric lines
clipping, 1121-1123
overview, 1119-1120
Linked lists
basic implementation, 283-285
circular lists, 288-292
dummy nodes, 285-287
head pointers, 284, 285
InsertNodeSorted assembly routine, 290
overview, 282
sentinels, 285-287
sorting techniques, 755
tail nodes, 286
test-bed program, 291
Little endian format, 252
Local optimization
  See also Assembly language
  optimization; Optimization.
bit flipping and flags, 146-147
defined, 140
incrementing and decrementing, 147-148
lookup tables, 145-146
unrolling loops, 143-145, 305, 312,
  377-378, 410
LOCK instruction, 377
Lockstep execution, Pentium processor,
  390-394, 400-403
LODSB instruction, 304, 312
LODSD instruction, 171
LODSW instruction, 312
Logical functions, ALU, 458
Logical height, virtual screen, 442
Logical width, virtual screen, 442
Long-period Zen timer
  See also Zen timer.
calling from C code, 69-72
and interrupts, 53
LZTEST.ASM listing, 66-67
LZTIME.BAT listing, 67-68
LZTIME2.ASM listing, 55-65
overview, 53
PS2 equate, 65-66
system clock inaccuracies, 43,
  45-46, 48
test-bed program, 66-69
TESTCODE listing, 69
ZTimerOff subroutine, 59-63
ZTimerOn subroutine, 58-59
ZTimerReport subroutine, 63-65
Lookup tables
CosSin subroutine, 994-996, 999
vs. rotating or shifting, 145-146
3-cell-per-word implementation, Game
  of Life, 365
word count program
  author's implementation, 303, 304
  David Stafford's implementation,
  309-311, 317-319
WC50 (Terje Mathisen), 307
LOOP instruction
  See also Loops.
vs. DEC/JNZ sequence, 139, 140-141
and flags, 148
Loops
  See also LOOP instruction.
  avoiding, 140
  and branch prediction, Pentium
    processor, 377-378
  unrolling, 143-145, 305, 312,
    377-378, 410
M

Mackraz, Jim, 678
Map Mask register
demo program, 472-473
drawing text, 833
optimizing Mode X, 1074
vs. Read Map register, 526
selecting planes for CPU writes, 443-444, 471-472
with set/reset circuitry, 474
write mode 1, 443
Map Mask register demo program,
472-473
Mask register, blanking screen, 651
Masked copying, Mode X
clocking, 923
between display memory locations,
918-919, 919-921
image and mask alignments,
generating, 922-923
performance, 924
system memory to display memory,
916-918, 916
Masked images, 871-872
MASM (Microsoft Assembler), 187
Math, 3-D
cross products, 1139-1140
dot products
calculating, 1135-1137
calculating light intensity, 1137
projection, 1141-1142
rotation, 1143-1144
sign of, 1140-1141
of unit vectors, 1136
of vectors, 1135-1136
matrix math
assembly routines, 992, 996-999
C-language implementations, 974-976
normal vectors, calculating, 955-956
rotation of 3-D objects, 938-939, 943-944, 948
transformation, optimized, 1172-1173, 1173-1174
3-D rotation, representation of, 938-939
Matrix math
assembly routines, 992, 996-999
C-language implementations, 974-976
normal vectors, calculating, 955-956
rotation of 3-D objects, 938-939, 943-944, 948
transformation, optimized, 1172-1173, 1173-1174
MDA (Monochrome Display Adapter), 104
MemchrO function, 116
MemcmpO function, 116
MemcpyO function, 1147-1148
Memory access
See also Display memory access.
clock cycles, bytes vs. words, 82, 83-85
DEC instruction, 83
and DRAM refresh, 98
8-bit bus cycle-eater, 82
performance, 286 and 386 processors, 223-225
prefetch queue cycle-eater, 86
system wait states, 210-213
and wait states, 106
Memory addressing, 221
Memory addressing modes, and arithmetic operations, 130-133
Memory allocation
display memory, 903-904
page flipping, 834
Memory locations, pushing and popping, 254-255
Memory variables
data alignment, 213-215
8088 processor, optimization, 83-85
Memory-addressing instructions, 223-225
MemsetO C library function, 727
Miles, John, 1081, 1093
Mipmapping, 1254-1255
Mode 12H (hi-res mode), 851-855
Mode 13H, 515, 590
Mode Control register, 575
Mode register
color paging, 628-629
256-color modes, 629
Mode X
See also X-Sharp 3-D animation package.
animation demo programs
page-flipped animation, 924-925, 925-930
3-D polygon rotation, 939, 940-945, 948
bitmap organization, 882-883
features, 878-879
FillRectangleX subroutine
four-plane parallel processing, 888-891, 891-893
pixel-by-pixel plane selection, 885-887
plane-by-plane processing, 887-889
four-plane parallel processing, 888-891, 891-893
latches
    copying pixels, 905-907, 908, 909-911
    loading, with double copying process, 903
overview, 897-898
pattern fills, 899, 900-903, 903-904
masked copying
    animation demo program, 924-925, 925-930
clipping, 923
between display memory locations, 918-919, 919-921
image and mask alignments,
generating, 922-923
performance, 924
system memory to display memory,
916-918, 916
memory allocation, 903-904
mode set routine, 880-881, 882
optimization, 1074
pattern fills, 899, 900-903, 903-904
pixel access and hardware planes, 1082
ReadPixelX subroutine, 884-885
vertical scanlines vs. horizontal, 1084-1086
WritePixelX subroutine, 883-884
ModelColor structure, 1035
ModelColorToColorIndex function, 1036, 1038
Mod-R/M byte, 257
Modular code
    and future of programming profession, 725-726
    optimizing, 153
Monotone-vertical polygons, filling, 760-761, 761-771, 771
MOV instruction, 236, 377, 832
MoveBouncer function, 824-825
MoveObject function, 929
MoveXSortedToAET function
    complex polygons, 749
    monotone-vertical polygons, 770
MOVSD instruction, 222, 386
MUL instruction, 97, 173-174
Multiplication
    increasing speed of, 173-174
    using LEA, 132-133, 172
    Multi-word arithmetic, 147-148

NEG EAX instruction, 222
Negation, two's complement, 171
Next1 function, 353
Next2 function, 353
Next_generation method, 327-328, 335, 336, 337-338, 344
Nonconvex objects, depth sorting, 1000, 1001-1002
Normal vectors
    building BSP trees, 1106
calculating, 955-956
direction of, 1140
Normals. See Normal vectors.
NOSMART assembler directive, 72
NOT instruction, 146-147, 147

Object collisions, detecting, 531-534
Object space, 935, 1135
Object-oriented programming, 725-726
Octant0 function
    360x480 256-color mode line drawing demo program, 615
    Bresenham's line-drawing algorithm, 662, 668-669
Octant1 function
    360x480 256-color mode line drawing demo program, 616
    Bresenham's line-drawing algorithm, 663, 668-669
Octants, and line orientations, 666-667

1/z sorting
  abutting span sorting, 1229-1230
  AddPolygonEdges function, 1232-1233, 1238
  vs. BSP-order sorting, 1226-1227
calculating 1/z value, 1220-1222
ClearEdgeLists function, 1236-1237
DrawSpans function, 1236
  independent span sorting, 1230, 1231-1238, 1239-1241
  intersecting span sorting, 1228-1229
PolyFacesViewer function, 1232
  reliability, 1227
ScanEdges function, 1234-1236, 1238-1239
  UpdateWorld function, 1237-1238

On-screen object collisions, detecting, 531-534
OpenGL API, GLQuake, 1288-1290
Operands, order of, 173-174
OPT2.ASM listing, 313-315

Optimization
  See also Assembly language
    optimization; Local optimization.
  32-bit registers, 187
  and abstraction, 330-332, 345-346
  and application parameters, 122
  assemblers, optimizing, 71-72
  avoiding thinking like a compiler, 152, 154-155
  and biased perceptions, 1080, 1085
  breakthrough level, 316
  BSP trees, 1128-1129
  buffer-filling routine, 416-420
  C library functions, 15
  compiler-based
    data recursion vs. code recursion, 1112-1113
    on vs. off, 9
data recursion, 1108-1113
data structures, 155-166
disk caches, 19
display adapter cycle-eater, 107
DRAM refresh, 98-99
8-bit bus cycle-eater, 83-85
fine-tuning existing code, 312-313
floating point operations
  clock cycles, core instructions, 1167-1168
cross product optimization,
  1171, 1172
dot product optimization, 1170, 1171
FXCH instruction, 1169-1170
interleaved instructions, 1169-1170
matrix transformation optimization,
  1172-1173, 1173-1174
overview, 1167-1170
pipelining, 1168-1170
projection to screen space, 1174
rounding control, 1174-1175

486 processor
  addressing pipeline penalty, 238-240, 243, 250-252
  internal cache, 236
  vs. Pentium processor, 378-379
  pushing and popping, 254-255
  reference materials, 236
  shift and rotate instructions, 255-256
  single cycle, importance of, 238
  stack addressing, 241-242
general rules, 223
generality, decreasing, 335
hardware efficiency, 1084-1086
knowing when to stop, 735
local optimization, 138-148
Mode X, 1074
modular code, 153
objectives and rules, 7-19, 156
pattern matching, 191-192, 202

Pentium processor
  and branch prediction, 378
code size and performance, 390
floating point operations, 1167-1175
interleaving operations, 394-395
pairing instructions, 390-394
pixel-drawing code, 1086
prefix bytes, 376, 395, 407
reference material, 374
superscalar execution, 384-396
vs. 386 and 486 processors, 378-379, 384

perspective on problem, changing, 315-316, 1084
pixel drawing, 1074
pointer advancement optimization,
  1086-1089, 1090-1091, 1092-1093
prefetch queue cycle-eater, 93
problem definition, changing, 332
rearranging instructions, 418-419
reducing size of code, 416-418
redundant calculations, 682-683
re-examining problem, 338-339
register variables, 338
restartable blocks, 118
sorting techniques, 755
stack addressing, 420
sufficient, 312
superscalar execution
  initial pipe, effect of, 405
  overview, 385-386
  pairable instructions, 388
  V-pipe-capable instructions, 386-387
texture-mapping optimization
  inner-loop optimizations, 1069-1073, 1074, 1081-1084
  instruction-by-instruction optimizations, 1086-1092
  pointer advancement optimization, 1086-1089, 1090-1091
  vertical scanlines, 1084-1086
32-bit fixed-point arithmetic, 1086-1089, 1090-1091, 1092-1093
32-bit instructions, 1091
386 processor, 378-379
time vs. space tradeoff, 187
transformation inefficiencies, 25-26
transformation matrices, 986
understanding data, importance of, 122, 175, 180, 305
understanding how things work, 726
unifying model, developing, 1110-1111
unrolling loops, 143-145, 410
using restartable blocks, 118
and VGA memory speed, 704-705
Optimized searching, 174-180
Optimizing assemblers, 71-72
OR instruction, 377
Orientation-independent texture mapping, 1065-1066, 1067
OUT instruction
  clock cycles, 1082-1083
  loading DAC, 640, 642-643
  loading palette RAM or DAC registers, 632
  performance, 444, 843
  word-OUT vs. byte-OUT, 429, 479
  vs. write mode 3, 483-484
OUTS instruction, 221
OUT_WORD macro, 566, 594
Overdraw problem, VSD
  and beam trees, 1185-1186
  painter's algorithm, 1184-1185
  sorted spans, 1215
Overflow register, split screen operation, 565
Overhead
  DOS function calls
    in 16-bit checksum program, 12
    in search engine, 121
  memcmp() function, 116
  strstr() function, 115
  of Zen timer, timing, 46, 72
Overlapping rectangles, in dirty-rectangle animation, 872-873
Overscan, 555-556, 641

P

Page flipping
  and bit-plane animation, 814
color cycling, 650
vs. dirty-rectangle animation, 846, 862
display memory start address, changing, 857
mechanics of, 833-836
memory allocation, 834, 903-904
overview, 444-446
single-page technique, 855-857
640×480 mode, 836-837
with split screen, 836-837
320×400 256-color mode, 600-605
timing updates, 835-836
VGA mode 12H (hi-res mode), 851-855
Page flipping animation demo programs
  Mode X, 924-925, 925-930
  split screen and page flipping, 820-825, 825-830, 836-837
  320×400 256-color mode, 600-605
Painter's algorithm
  See also 3-D animation, 3-D drawing, and BSP trees, 1099, 1104-1105
overdraw problem, 1184-1185
potentially visible set (PVS), precalculating, 1188-1189
Pairable instructions, Pentium processor, 388
Palette adjustment, dynamic, 1039
Palette RAM
  See also Palette registers.
color paging, 628-629
setting registers, 629-630, 631-632
VGA color path, 626
Palette registers
  See also Palette RAM.
EGA, 549-550
setting for bit-plane animation, 799-801, 811-813
Panning
  byte-by-byte vs. pixel-by-pixel, 574
overview, 441-442
  in split screens, 574-575, 575-582, 582-583
  in text mode, 442
PanRight subroutine, 582
Parametric lines
  clipping, 1121-1123
overview, 1119-1120
Particles, Quake 3-D engine, 1287
Pattern fills, 899, 900-903, 903-904
Pattern matching, 191-192, 202
PC compatibility, Zen timer, 48-49
Pel panning. See Panning.
Pel Panning register, 574, 583
Pentium processor
  AGIs (Address Generation Interlocks), 400-403
  alignment, 376
  branch instructions, pairing, 404-405
  branch prediction, 377-378
  bus, locking, 377
  cache lines, 374
  code size and performance, 390
data cache and paired instructions, 391
  display adapter cycle-eater, 107
EA (effective address) calculations, 375-376
floating point optimization
  clock cycles, core instructions, 1167-1168
cross product optimization, 1171, 1172
dot product optimization, 1170, 1171
FXCH instruction, 1169-1170
interleaved instructions, 1169-1170
matrix transformation optimization,
  1172-1173, 1173-1174
overview, 1167-1170
pipelining, 1168-1170
projection to screen space, 1174
rounding control, 1174-1175
FPU pipelining, 1168-1170
instruction fetching, 374
internal cache, 374-375, 396
LAHF and SAHF instructions, 148
LEA vs. ADD instructions, 131
LODSB instruction, 304
LOOP instruction vs. DEC/JNZ
  sequence, 139
MOV vs. XCHG instructions, 377
optimization
  pairing instructions, 390-394
  pixel-drawing code, 1086
reference material, 374
overview, 373-375
pipeline stalls
  FPU, 1168-1170
overview, 375
texture-mapping code, 1092
prefix bytes, 376, 395, 407
running Pentium code on
  386 or 486, 411
superscalar execution
  initial pipe, effect of, 405
  interleaving operations, 394-395
  internal cache, 396
  lockstep execution, 390-394, 400-403
overview, 384-386
pairable instructions, 388
prefix bytes, 395
register contention, 403-405
registers, small set, 395
U-pipe, 385-386
V-pipe, 385-386, 386-387
XCHG vs. MOV instructions, 377, 832
Pentium Processor Optimization Tools
  (book), 1148
Performance
  See also Assembly language
  optimization; Clock cycles; Cycle-eaters; Local optimization;
  Optimization; Zen timer.
and abstraction, 330-332, 345-346
beam trees, 1186
Boyer-Moore algorithm, 266-268
branching, 140
BSP (Binary Space Partitioning) trees,
1100, 1111-1113
bubble sort, 755
complex polygons, filling, 753
dirty-rectangle animation, 873
display adapter cycle-eater, 221
DRAM refresh, 97
function calls, 153
Game of Life
byte-per-cell implementation, 340
cellmap-wrapped implementation,
332, 338
challenge results, 351
general analysis, 329-330
and generality, 335
level performance, 1213-1214
lookup tables, vs. rotating or shifting,
145-146
masked copying, Mode X, 924
measuring, importance of, 34, 396
memory access, 223-225
OUT instruction, 444
OUT instructions, 843
PC-compatible computers, 48-49
polygon-filling implementations, 728
precalculated potentially visible set
(PVS), 1213-1214
profiling and 80x87 emulator, Borland
C++, 999
stack frames, 153
SuperVGA, with 486 processor, 842-844
texture mapping, 1074-1074
3-D polygon rotation demo
programs, 949
360x480 256-color mode, 618
320x400 256-color mode, 599-600
time-critical code, 13
vertical scanlines in texture mapping,
1084
video performance, 104
Wu antialiasing algorithm, 777-778
z-buffers, 1213
Perspective correction in texture
mapping, 1093
Perspective correctness problem,
Gouraud shading, 1248-1250
Perspective projection, 937, 1135
See also Projection.
Pipelining
486 processor
addressing pipeline penalty, 238-
240, 250
stack addressing, 241-242
FPU, Pentium processor, 1168-1170
Pitch angle, in polygon clipping, 1206
Pixel bits, rotating, 252
Pixel drawing
See also Pixels.
EVGADot function, 661-662, 669-670
optimization, 1074, 1086
painter's algorithm and overdraw
problem, 1184
Pixel intensity calculations, Wu's
antialiasing algorithm, 778-779
Pixel values, mapping to colors, 548-551,
551-555
Pixels
See also Boundary pixels, polygons;
Pixel drawing.
copying, using latches (Mode X), 905-
907, 908, 909-911
reading (320x400 256-color mode), 599
redrawing, display adapter
cycle-eater, 102
rotating bits, 252
writing (320x400 256-color mode),
599, 600
Plane mask, 1074
Plane-manipulation demo program,
476-478
Planes
clipping line segments to, 1195-1197
1/z value, calculating, 1221
representation, 1196
Planes, VGA
See also Bit-plane animation.
ALUs and latches, 451-453
and bit mask, 465
capturing and restoring screens, 541-
542, 543-547, 547-548
and Color Don't Care register, 534-535,
535-537
Index 1325
fonts, in text modes, 516
manipulating, 443-444, 476-478
and Map Mask register, 471-472
Mode X
  bitmap organization, 882-883
  four-plane parallel processing, 888-889, 891-893
  pixel-by-pixel plane selection, 885-887
  plane-by-plane processing, 887-889
Mode X pixel access, 1082
overview, 430
and Read Map register, 542
read mode 0, 525-526
and set/reset circuitry, 471-478
setting all to single color, 473-474
single-color drawing with write mode 3, 831-832
write mode 2, 502-504, 509
Pohl, Frederick, 1275
Pointer advancement optimization, 1086-1089, 1090-1091, 1092-1093
Pointer arithmetic, 171
Points, representation of, 1196
PolyFacesViewer function, 1203, 1232
Polygon clipping
  BackRotateVector function, 1203
  clipping to frustum, 1200, 1201-1206, 1206-1207
  ClipToFrustum function, 1204
  ClipToPlane function, 1199
  optimization, 1207
  overview, 1197-1200
PolyFacesViewer function, 1203
ProjectPolygon function, 1201
SetUpFrustum function, 1204
SetWorldspace function, 1204
TransformPoint function, 1203
TransformPolygon function, 1203
UpdateViewPos function, 1202
UpdateWorld function, 1205
viewspace clipping, 1207
ZSortObjects function, 1201
POLYGON.H header file
  complex polygons, 751
  monotone-vertical polygons, filling, 771
  non-overlapping convex polygons, 719-720
texture mapped polygons, 1054
3-D polygon rotation, 945-946
3-D solid cube rotation program, 965
X-Sharp 3-D animation package, 982-984
Polygon models, Quake 3-D engine, 1285-1286
Polygon-filling programs
  See also Polygons, filling.
complex polygons, 742-744, 745-752, 753, 754, 755-756
monotone-vertical polygons, 760, 761-771
non-overlapping convex polygons
  assembly implementations, 732-733, 733-734, 735-739
  C-language implementations, 713-720, 720-721, 729-732
PolygonIsMonotoneVertical function, 761
Polygons
  See also Texture mapping.
adjacent, and 1/z span sorting, 1230
backface removal, 954-957, 1160-1161
categories of, 710, 742, 759-760
clipping, 1158-1159
Gouraud shading, 1247
hidden surface removal, 1214-1222
normal vector, calculating, 955-956
projection in 3-D space, 937, 944-945, 948
representation, 1196
3-D polygon rotation demo program
  matrix multiplication functions, 943-944, 948
  overview, 939
  performance, 949
  polygon filling with clipping support, 940-943
transformation and projection, 944-945, 948
transformation to 3-D space, 935
unit normal, calculating, 1027-1028, 1137-1140
visibility, calculating, 955-956
visible surface determination (VSD)
  beam trees, 1185-1189
overdraw problem, 1184-1185
polygon culling, 1181-1184
potentially visible set (PVS),
precalculating, 1188-1189
visible surface determination
(VSD) culling to frustum, 1181-1184
wall orientation testing, BSP tree
rendering, 1160-1161

Polygons, filling
See also Polygon-filling programs;
Polygons; Texture mapping.
active edges, 742-744, 753, 755, 756
boundary pixels, selecting, 712
with clipping support, 940-943
complex polygons, 742
drawing, speeding up, 727-729
edge tracing
overview, 711-713
ScanEdge function, 716-717, 720-721, 730-732, 735-738
fitting adjacent polygons, 712-713
flat vs. pointed top, 720
integer vs. fixed-point arithmetic, 1065
in Mode X, 940-943
monotone-vertical polygons, 760-761, 771
nonconvex polygons, 755
non-overlapping convex polygons, 720-721
performance, comparison of implementations, 728
rasterization, 710-712
scan conversion, 710, 720-721
active edges, 721, 742-744, 753, 755, 756
C-language implementation, 713-717, 720-721
defined, 710
zero-length segments, 721

Polyhedrons
hidden surfaces, 955, 1000, 1001-1002
representation of, 962
3-D solid cube rotation demo program
basic implementation, 957-961, 962-963
incremental transformations, 964-966
object representation, 967
POPF instruction, 226, 226-231
Popping, memory locations vs. registers, 254-255
Portable code, and future of
programming profession, 725-726
Portals
and beam trees, 1188
in Quake 3-D engine, 1279-1280
Potentially visible set (PVS)
vs. portals, 1279-1280
precalculating, 1188-1189, 1213-1214
Quake 3-D engine, 1278-1279
Precalculated results
BSP trees and potentially visible set
(PVS), 1188-1189
lookup tables, 146
Precision
long-period Zen timer, 53
rounding vs. truncation, 1002-1003
Zen timer, 48, 52
Prefetch queue
286 and 386 processors, 225
Prefetch queue cycle-eater
286 and 386 processors, 210
instruction execution times, 87-93
optimizing for, 93
overview, 86
system wait states, 210
and Zen timer, 88, 92
Prefix bytes
Pentium processor, 376, 395, 407
and stack-based variables, 184
Prefixes. See Prefix bytes.
Principles of Interactive Computer Graphics (book), 934
Problems, quick responses to, 1166
Profiling, and 80x87 emulator, Borland C++, 999
Program size vs. clock cycles, 28
Programmer’s Guide to PC Video Systems
(book), 651
Projection
defined, 1135
floating point optimization, 1174
LineIntersectPlane function,
1142-1143
overview, 937, 948
XformAndProjectPoly function, 944-945
rotation without matrices, 1143-1144
using dot product, 1141-1142
ProjectPolygon function, 1201
Proportional text, 489
Protected mode
    addressable memory, 221
486 processor
    addressing calculation pipeline, 239
    indexed addressing, 237-238
    general tips, 140
over view, 208-209
    32-bit addressing modes, 256-258
PS2 equate, long-period Zen timer, 65-66
PS/2 computers, 54, 66
PUSH instruction, 222, 241-242, 404
PUSHA instruction, 221
Pushing, memory locations vs. registers, 254-255
PZTEST.ASM listing, Zen timer, 49
PZTIME.BAT listing, Zen timer, 51
PZTIMER.ASM listing, Zen timer, 35-42

Q
QLife program, 352-363
QSCAN3.ASM listing, 309-311
Quake 2, 1293
Quake
    surface caching, 1253-1256, 1260-1262
    surface-based lighting
        description, 1250-1251
        mipmapping, 1254-1255
        performance, 1251-1253
    surface caching, 1253-1256,
        1260-1262
texture mapping, 1261-1262
3-D engine
    BSP trees, 1276-1277
    lighting, 1282-1283
    model overview, 1276-1277
    portals, 1279-1280
    potentially visible set (PVS), 1278-1279
    rasterization, 1282
    world, drawing, 1280-1281
and visible surface determination (VSD), 1181
QuakeWorld, 1291-1292

R
Radiosity lighting, Quake 2, 1293
Rasterization of polygons
    See also Pol ygons, filling.
    boundary pixels, selecting, 712
efficient implementation, 711
in Quake 3-D engine, 1282
Rate of divergence, in 3-D drawing, 937
Raycast, subdividing, and beam trees, 1187
RCL instruction, 185-186
RCR instruction, 185-186
Read360x480Dot subroutine, 614-615
Read() C library function
    vs. getc() function, 12
overhead, 121
Read Index register, 651-652
Read Map register
demo program, 526-530
planes, specifying to be read, 542
read mode, 0, 526
Read Map register demo program, 526-530
Read mode, 0, 521
Read mode 1
    Color Don’t Care register, 534
    overview, 525-526
    vs. read mode 0, 521
    selecting, 525
Read/write/modify operations, 107
Read-after-write register contention, 404
ReadPixel subroutine, 598, 599
ReadPixelX subroutine, 884-885
Real mode. See 386 processor.
Real mode
    addressing calculation pipeline, 239
    32-bit addressing modes, 256-258
Rectangle fill, Mode X
    four-plane parallel processing, 888-891, 891-893
    pixel-by-pixel plane selection, 885-887
    plane-by-plane processing, 887-889
Recursion
    BSP trees
        building BSP trees, 1101-1104
data recursive inorder traversal, 1107-1113
    visibility ordering, 1104-1106
code recursion
  vs. data recursion, 1108-1110
Euclid’s algorithm, 198-199
compiler-based optimization,
  1112-1113
data recursion
  vs. code recursion, 1108-1110
compiler-based optimization,
  1112-1113
Euclid’s algorithm, 200
inorder tree traversal, 1108-1110
performance, 1111-1113
performance, 1111-1113
Reference materials
3-D drawing, 934-935
3-D math, 1135
bitmapped text, drawing, 471
Bresenham’s line-drawing algorithm, 660
BSP trees, 1114, 1157
circle drawing, 626
color perception, 625
8253 timer chip, 72
486 processor, 236
 parametric line clipping, 1121
Pentium processor, 374, 1148
SVGA programming, 626
VGA registers, 583
ReferenceZTimerOff subroutine, 41
ReferenceZTimerOn subroutine, 40
Reflections, in GLQuake, 1290
Reflective color, vs. emissive color, 1035
Register contention, Pentium processor,
  403-405
Register-only instructions, 223-225
Registers
  See also 32-bit registers; VGA registers.
AX register, 171
copying bytes between, 172
EGA palette registers, 549-550
8-bit bus cycle-eater, 85
486 processor
  addressing pipeline penalty, 238-240, 250
  byte registers and lost cycles, 242-245
  indexed addressing, 237-238
  pushing or popping, vs. memory locations, 254-255
scaled, 256-258
stack addressing, 241-242
32-bit addressing modes, 256-258
prefetch queue cycle-eater, 94
and split screen operations, 573
and stack frames, 153
VGA architecture, 427-429
Relocating bitmaps, 516-517
Rendering BSP trees
backface removal, 1160-1161
clipping, 1158-1159
ClipWalls function, 1152-1155,
  1158-1159
DrawWallsBackToFront function,
  1155-1156, 1160-1161
overview, 1149
reference materials, 1157
TransformVertices function, 1151-
  1152, 1158
UpdateViewPos function, 1151, 1157
UpdateWorld function, 1156-
  1157, 1157
viewspace, transformation of objects to, 1158
wall orientation testing, 1160-1161
WallFacingViewer function, 1150-
  1151, 1161
RenderMan Companion (book), 742
REP MOVVS instruction, 148
REP MOVSW instruction, 82, 105, 220
REP SCASW instruction, 166
REP STOS instruction, 727, 735
REPNZ SCASB instruction
  vs. Boyer-Moore algorithm, 267-268,
    271, 274
in string searching problem, 121-122,
  174-175, 262-263
REPZ CMPS instruction
  vs. Boyer-Moore algorithm, 267-268,
    271, 274
in string searching problem, 121-122,
  174-175, 262-263
Restartable blocks
in 16-bit checksum program, 16
optimizing file processing, 118
performance, 122
in search engine, 117-118
size of, 114, 121

Index 1329
Results, precalculating
See also lookup tables.
BSP trees and potentially visible set (PVS), 1188-1189
RET instruction, 241-242
Reusable code, and future of programming profession, 725-726
RGB (red, green, blue) color model mapping to 256-color mode, 1036, 1037-1038, 1039
overview, 1034-1035
Richardson, John, 316
Right-handed coordinate system, 935-937
ROL instruction, 185-186
Roll angle, in polygon clipping, 1206
ROR instruction, 185-186
Rotate instructions
hand assembling, 255-256
n-bit vs. 1-bit, 255-256
286 processor, 222
RotateAndMovePObject function, 977-978
Rotation, 3-D animation
ConcatXforms function, 944
matrix representation, 938-939
multiple axes of rotation, 948
using dot product, 1143-1144
XformVec function, 943
Rotational variance, 1249
Rotations, bitwise
vs. lookup tables, 145-146
multi-bit vs. single-bit, 185-186
Rounding vs. truncation
in 3-D animation, 1002-1003
floating point optimization, 1174-1175
texture mapping, 1066-1067
Run-length slice algorithm
assembly implementation, 698-704
C-language implementations, 688-692, 692-693
description, 683-684
implementation details, 685-687
integer-based implementation, 685-687
potential optimizations, 705
Ruts, mental, staying out of, 1147-1148
monotone-vertical polygons, 770

ScanOutline function
assembly implementation, 1069-1073, 1074
C-language implementation, 1058-1059, 1067-1069

SCASW instruction, 161

Screen blanking
demo program, 556-557
using DAC Mask register, 651
Screen blanking demo program, 556-557
Screen capture programs, 541-548
Screen redraws, and display adapter
cycle-eater, 101, 102
Screen refresh rate, 619

Screenspace
defined, 1135
and normals of polygons, 1137-1138
projecting to, BSP tree rendering, 1159
uses for, 967

SEARCH.C listing, 118-121

Search engine
See also Searching.
Boyer-Moore algorithm, 263-277
design considerations, 114
execution profile, 121

FindString function, 175, 176, 178, 269
optimization, 174-180
restartable blocks, 117-118
search space and optimization, 122, 175
search techniques, 115-116, 175

SearchForString function, 118

Search engine
See also Searching.
Boyer-Moore algorithm, 263-277
design considerations, 114
execution profile, 121

Shading
See also Lighting; 3-D drawing.
ambient shading, 1023
diffuse shading, 1023-1024
directed light sources, 1028
effects, 360x480 256-color mode, 618
overall shading, calculating, 1025
of polygons, 1025-1026, 1027-1029

Shearing
cause of, 813
in dirty-rectangle animation, 846
page flipping, 814
sheep, 1063
Shift instructions, 222, 255-256
Shifting bits, vs. lookup tables, 145-146

ShowBounceCount function, 823-824
ShowPage subroutine
masked copying animation, Mode X, 929-930
page flipping animation, 827
Show_text function, 329, 363
SHR instruction, 88-91, 97
SIB byte, 257
640x400 mode, mode set routine, 852-853
640x480 mode, page flipping, 836-837
16-bit checksum program
See also TCP/IP checksum program.
assembly implementation, 10-12, 17-18
C language implementation, 8-9, 15-16
overview, 8
redesigning, 9
16-color VGA modes
color paging, 628-629
DAC (Digital/Analog Converter), 626-628
palette RAM, 626
Small code model, linking Zen timer, 70
Software patents, 1194
Sorted span hidden surface removal
abutting span sorting, 1229-1230
AddPolygonEdges function, 1232-1233, 1238
BSP order vs. 1/2 order, 1220, 1226
ClearEdgeLists function, 1236-1237
DrawSpans function, 1236
edge sorting, 1220-1222
edges vs. spans, 1215-1220
independent span sorting, 1230, 1231-1238, 1239-1241
intersecting span sorting, 1228-1229
1/2 sorting, 1220-1223, 1227-1231, 1231-1238, 1239-1241
overview, 1214-1215
PolyFacesViewer function, 1232
ScanEdges function, 1234-1236, 1238-1239
UpdateWorld function, 1237-1238
1/2 sorting for hidden surface removal, 1220-1222
and optimization, 755
z-buffers, 1212-1213
SortObjects function, 1002
Span-based drawing, and beam trees, 1187
Specular reflection, 1023
Split screens
EGA bug, 573-574
horizontal panning, 574-575, 575-582, 583
overview, 563-565
page flipping, 640x480 mode, 836-837
registers, setting, 573
safety of, 585
split screen demo program, 565, 566-572, 572
text mode, 584
turning on and off, 565
SplitScreenDown subroutine, 572
SplitScreenUp subroutine, 572
Spotlights
Gouraud shading, 1247
shading implementation, 1028
Sprites
masked images, 871-872
Quake 3-D engine, 1287
Square wave timer mode, 44
Stack addressing
address pipeline effects, 241-242
assembly language optimization, 420
Stack frames, performance, 153
Stack pointer alignment, 218-219
Stack-based variables, placement of, 184-185
Stacks, POPF vs. IRET, 226-231
Stafford, David
25-byte sorting routine, 180-181
Game of Life implementation, 351-352, 353-363, 365-365
ScanBuffer assembly routine, word count program, 309-311, 317-319
24-byte hi/lo function, 292-293
Start Address High and Low registers, 834-836
State machines
3-cell-per-word implementation, Game of Life, 363-366
word count program, 315
StepEdge function, 1056-1057
STOSB instruction, 236
String instructions, 107
String searching. See Search engine; Searching.
StrstrO function, 115
SUB instruction, 219
Subdivision rasterization, 1266-1267, 1267-1270, 1286
Superscalar execution
initial pipe, effect of, 405
interleaving operations, 394-395
lockstep execution, 390-394, 400-403
overview, 384-386
register contention, 403-405
V-pipe-capable instructions, 386-387
SuperVGA, 104, 107, 842-844
Surface caching
hardware interactions, 1260-1262
surface-based lighting, 1253-1256
in VQuake, 1288
Surface-based lighting
description, 1250-1251
mipmapping, 1254-1255
performance, 1251-1253
surface caching, 1253-1256, 1260-1262
texture mapping, 1261-1262
System clock
inaccuracies
long-period Zen timer, 53, 54
Zen timer, 43, 45-46, 48
timer 0, 8253 chip, 44, 54
System memory, Mode X
copying to display memory, 908, 909-911
masked copy to display memory, 916-918, 916
System wait states, 210-213

Text
Table-driven state machines, 316-319
Tail nodes, in linked lists, 286
TASM (Turbo Assembler), 71-72
TCP/IP checksum program
basic implementation, 406
dword implementation, 409
interleaved implementation, 408
unrolled loop implementation, 410
Test function, 358, 365
TEST instruction, 377, 401-402
Texels
Gouraud shading, 1247
mipmapping, 1254-1255
Text, drawing
bitmapped text demo program
using bit mask, 466-469, 470-471
using write mode 3, 484-489, 489-490, 490-496
high-speed text demo program, using write mode 3, 490-496
solid text demo program, using latches, 1039-1041, 1042-1044
using write mode 0, 832-833
Text mode
display adapter cycle-eater, 104
horizontal resolution, 620
panning, 443
split screen operations, 584-585
Text pages, flipping from graphics to text, 517
TEXT_UP macro, 454, 459
TextUp subroutine, 829
Texture mapping
See also DDA (digital differential analyzer) texture mapping.
boundary pixels, polygons, 1049-1052, 1066, 1067
C implementation, 1053-1058
independent span sorting, 1238
onto 2-D transformed polygons, 1050
onto 3-D transformed polygons, 1051
onto untransformed polygon, 1049
optimization
inner-loop optimizations, 1069-1073, 1074, 1081-1084
instruction-by-instruction optimizations, 1086-1092
pointer advancement optimization, 1086-1089, 1090-1091
vertical scanlines, 1084-1086
orientation independence, 1065-1066, 1067
overview, 1048
Pentium pipeline stalls, 1092
perspective correction, 1093
surface-based lighting, 1261-1262
vertical scanlines, 1084-1086
32-bit addressing modes, 256-258
32-bit division, 181-184, 1008
32-bit fixed-point arithmetic, optimizing, 1086-1089, 1090-1091, 1092-1093
32-bit instructions, optimizing, 1091
32-bit registers
See also Registers; VGA registers.
adding with LEA, 131
BSWAP instruction, 252
multiplying with LEA, 132-133
386 processor, 222
time vs. space tradeoff, 187
using as two 16-bit registers, 253-254

3-D animation
See also Hidden surface removal; 3-D drawing; 3-D polygon rotation
demo program; X-Sharp 3-D animation package.
demo programs
solid cube rotation program, 957-961, 962-963, 964-966, 967
3-D polygon rotation program, 939, 940-945, 948-949
12-cube rotation program, 972, 973-984, 985-987
deep sorting, 1000, 1001-1002
rotation
ConcatXforms function, 944
matrix representation, 938-939
multiple axes of rotation, 948
XformVec function, 943
rounding vs. truncation, 1002-1003
translation of objects, 937-938

3-D clipping
arithmetic imprecision, handling, 1240
line segments, clipping to planes, 1195-1197
overview, 1195
polygon clipping
BackRotateVector function, 1203
clipping to frustum, 1200, 1201-1206, 1206-1207
ClipToFrustum function, 1204
ClipToPlane function, 1199
optimization, 1207

overview, 1197-1200
PolyFacesViewer function, 1203
ProjectPolygon function, 1201
setUpFrustum function, 1204
SetWorldspace function, 1204
TransformPoint function, 1203
TransformPolygon function, 1203
UpdateViewPos function, 1202
UpdateWorld function, 1205
viewspace clipping, 1207
ZSortObjects function, 1201

3-D drawing
See also BSP (Binary Space Partitioning) trees; Hidden surface removal; Polygons, filling; Shading; 3-D animation.
backface removal
BSP tree rendering, 1160-1161
calculations, 955-957
motivation for, 954-955
and sign of dot product, 1140
solid cube rotation demo program, 957-961, 962-963, 964-966, 967
background surfaces, 1240
draw-buffers, and beam trees, 1187
dynamic objects, 1100-1101
Gouraud shading, 1246-1250
lighting
Gouraud shading, 1246-1250
overlapping lights, 1247
perspective correctness, 1248-1250
rotational variance, 1249
surface-based lighting, 1250-1256,
viewing variance, 1249
moving models in 3-D drawings, 1212-1222
painter's algorithm, 1099, 1104-1105
perspective correctness problem, 1248-1250
portals, and beam trees, 1188
projection
dot products, 1141-1142
overview, 937, 948
raycast, subdividing, and beam trees, 1187
reference materials, 934-935
rendering BSP trees  
backface removal, 1160-1161  
clipping, 1158-1159  
ClipWalls function, **1152-1155**, 1158  
DrawWallsBackToFront function, **1155-1156**, 1160-1161  
overview, 1149  
reference materials, 1157  
TransformVertices function, **1151-1152**, 1158  
UpdateViewPos function, **1151**, 1157  
UpdateWorld function, **1156-1157**, 1157  
viewspace, transformation of  
objects to, 1158  
wall orientation testing, 1160-1161  
WallFacingViewer function, **1150-1151**, 1161  
span-based drawing, and  
beam trees, 1187  
transformation of objects, 935-936  
triangle model drawing  
fast triangle drawing, 1263-1265  
overview, 1262-1263  
precision, 1265  
subdivision rasterization, 1266-1267, **1267-1270**  
vertex-free surfaces, and  
beam trees, 1187  
visibility determination, 1099-1106  
visible surface determination (VSD)  
beam trees, 1185-1189  
culling to frustum, 1181-1184  
overdraw problem, 1184-1185  
polygon culling, 1181-1184  
potentially visible set (PVS), precalculating, 1188-1189  
3-D engine, Quake  
BSP trees, 1276-1277  
lighting, 1282-1283  
model overview, 1276-1277  
portals, 1279-1280  
potentially visible set (PVS), 1278-1279  
rasterization, 1282  
world, drawing, 1280-1281  
3-D math  
cross products, 1139-1140  
dot products  
calculating, 1135-1137  
calculating light intensity, 1137  
projection, 1141-1142  
rotation, 1143-1144  
sign of, 1140-1141  
of unit vectors, 1136  
of vectors, 1135-1136  
matrix math  
assemble routines, 992, **996-999**  
C-language implementations, **974-976**  
normal vectors, calculating, 955-956  
rotation of 3-D objects, 938-939, **943-944**, 948  
transformation, optimized, 1172-1173, **1173-1174**  
vector length, 1135  
3-D polygon rotation demo program  
matrix multiplication functions, **943-944**, 948  
overview, 939  
performance, 949  
polygon filling with clipping  
support, **940-943**  
transformation and projection, **944-945**, 948  
3-D solid cube rotation demo program  
basic implementation, **957-961**, 962-963  
incremental transformations, **964-966**  
object representation, 967  
386 native mode, 32-bit  
displacements, 187  
386 processor  
alignment, stack pointer, 218-219  
CMP instruction, 161, 306  
cycle-eaters, 209-210  
data alignment, 213, 218  
and display adapter cycle-eater, 107  
display adapter cycle-eater, 219-221  
double word alignment, 218  
DRAM refresh cycle-eater, 219  
effective address calculations, 129, 223-225  
LEA instruction, 130-133, 172  
LODSD vs. MOV/LEA sequence, 171  
lookup tables, vs. rotating or shifting,
145-146

**LOOP** instruction vs. **DEC/JNZ** sequence, 139

memory access, performance, 223-225

**MUL** and **IMUL** instructions, 173-174

multiplication operations, increasing speed of, 173-174

new instructions and features, 222

Pentium code, running on, 411

protected mode, 208-209

rotation instructions, clock cycles, 185-186

system wait states, 210-212

32-bit addressing modes, 256-258

32-bit multiply and divide operations, 985

using 32-bit register as two 16-bit registers, 253-254

**XCHG** vs. **MOV** instructions, 377, 832

386SX processor, 16-bit bus cycle-eater, 81

360×480 256-color mode

display memory, accessing, 621-622

**Draw360×480Dot** subroutine, 613-614

drawing speed, 618

horizontal resolution, 620

line drawing demo program, 615-618, 618-619

mode set routine (John Bridges), 609, 612, 620-621

on VGA clones, 610-611

**Read360×480Dot** subroutine, 614-615

256-color resolution, 619-620

vertical resolution, 619

320×400 256-color mode

advantages of, 590-591

display memory organization, 591-593

line drawing, 600

page flipping demo program, 600-605

performance, 599-600

pixel drawing demo program, 593-598, 599-600

320×240 256-color mode. See **Mode X**.

Time perception, subjectivity of, 972

Time-critical code, 13

Timer 0, 8253 timer chip

operation, 44

stopping, 54, 65

Timer modes, 44, 45

**TIMER_INT** BIOS routine, 44

Timers

*See also* 8253 timer chip; Long-period Zen timer; Zen timer.

divide-by-N mode, 45

square wave mode, 44

Timeslicing delays, 446

Timing intervals

long-period Zen timer, 53

Zen timer, 45

Transformation inefficiencies, 25-26

Transformation matrices. *See Matrices; Matrix math.*

Transformation of 3-D objects

defined, 1135

floating point optimization, 1172-1173, 1173-1174

incremental transformations, 964

steps in, 935-936

**TransformPolygon** function, 1203

Translation in 3-D space, 937-938

Treuenfels, Anton, 756

Triangle model drawing

fast triangle drawing, 1263-1265

overview, 1262-1263

precision, 1265

subdivision rasterization, 1266-1267, 1267-1270

Triangles, and rotational variance, 1249-1250

Trinity, 1294

Truncation errors, in 3-D animation, 1002-1003

Truncation vs. rounding

floating point optimization, 1174-1175

texture mapping, 1066-1067

TSRs, and DAC, loading, 643, 648

Turbo Profiler, and 80×87 emulator, Borland C++, 999

12-cube rotation demo program

limitations of, 986

optimizations in, 985-986

performance, 986
X-Sharp animation package, 972, 973-984, 984-985
24-byte hi/lo function, 292-293
286 processor
CMP instruction, 161, 306
code alignment, 215-218
cycle-eaters, 209-210
data alignment, 213-215
data transfer rates, 212
display adapter cycle-eater, 219-221
display memory wait states, 220
DRAM refresh cycle-eater, 219
effective address calculations, 129, 223-225
instruction fetching, 215-218
LEA vs. ADD instructions, 130
lookup tables, vs. rotating or shifting, 145-146
LOOP instruction vs. DEC/JNZ sequence, 139
memory access, performance, 223-225
new features, 221
POPF instruction, and interrupts, 226
protected mode, 208-209
stack pointer alignment, 218-219
system wait states, 210-212
256-color modes
See also 320x400 256-color mode.
DAC settings, 629
mapping RGB model to, 1036, 1037-1038, 1039
resolution, 360x480 256-color mode, 619-620
Two-pass lighting, 1262
Two’s complement negation, 171

UpdateWorld function, 1205, 1237-1238
U-pipe, Pentium processor
branch instructions, 404-405
overview, 385-386
pairable instructions, 388

V

Variables, word-sized vs. byte-sized, 82, 83-85
Vectors
cross product, 1139-1140
cross-products, calculating, 955-956
dot product, 1135-1137
length equation, 1135
optimization of, 986
unit vectors, dot product, 1136-1137
VectorsUp function
Bresenham’s line-drawing algorithm, 664-665
360x480 256-color mode line drawing program, 617-618
Verite Quake, 1287-1280
Vertex-free surfaces, and beam trees, 1187
Vertical blanking, loading DAC, 641
Vertical resolution, 360x480 256-color mode, 619
Vertical scanlines, in texture mapping, 1084-1086
Vertical sync pulse
loading DAC, 641, 648
and page flipping, 444-446, 835-836
split screens, 573
VGA BIOS
DAC (Digital/Analog Converter)
loading, 641-642, 648
setting registers, 630, 631-632
vs. direct hardware
programming, 458-459
function 13H, 459
and nonstandard modes, 854-855
palette RAM, setting registers, 629-630, 631-632
reading from DAC, 652
text routines, in 320x400 256-color mode, 592

Unifying models, and optimization, 1110-1111
Unit normal of polygons, calculating, 1027-1028, 1137-1140
Unit vectors, dot product, 1136-1137
Unrolling loops, 143-145, 305, 312, 377-378, 410
UpdateViewPos function, 1202

Index 1337
and VGA registers, 458
VGA clones
  potential incompatibilities, 446-447
  360×480 256-color mode, 610-611
VGA color path
  color paging, 628-629
  DAC (Digital/Analog Converter), 626-628, 630, 631-632
  palette RAM, 626, 629-630, 631-632
VGA compatibility, 446-447, 610-611
VGA memory
  Color Don't Care register, 535-537, 535
  CPU reads, 520, 526
VGA modes
  bit-plane animation, 811
  color compare mode, 531-534, 531
  mode 0, set/reset circuitry, 471-472, 474-479
  mode 12H (hi-res mode), page flipping, 851-855
  mode 13H
    converting to 320×400 256-color mode, 593
    overview, 515
    resolution, 590
Mode X
  bitmap organization, 882-883
  copying pixels using latches, 905-907, 908, 909-911
  features, 878-879
  four-plane parallel processing, 888-891, 891-893
  masked copying, 916-918, 916, 918-919, 919-921
  memory allocation, 903-904
  mode set routine, 880-881, 882
  pattern fills, 899, 900-903, 903-904
  pixel-by-pixel plane selection, 885-887
  plane-by-plane processing, 887-889
ReadPixelIX subroutine, 884-885
WritePixelIX subroutine, 883-884
  and page flipping, 444-445
read mode 1
  Color Don't Care register, 534
  overview, 525-526, 531
  selecting, 525
  and set/reset circuitry, 478
640×400 mode set routine, 852-853
split screen operations, 584-585
text mode, panning, 443
320×400 256-color mode
  advantages, 590-591
  converting mode 13H to, 593
display memory organization, 591-593
  page flipping demo program, 600-605
and virtual screens, 441
write mode 0, drawing text, 832-833
write mode 1, overview, 444
write mode 2
  chunky bitmaps, converting to planar, 504-505, 505-508
  mechanics, 502
  overview, 501-502
  selecting, 504
write mode 3
  vs. Bit Mask register, 844
drawing bitmapped text, 484-489, 489-490, 490-496
  overview, 483-484, 496
  single-color drawing, 831-832
  vs. write mode 0, 490
VGA registers
AC Index register, bit 5 settings, 443
Bit Mask register
  bit mask, controlling, 465
drawing solid text, 1040
  setting inside a loop, 429
  vs. write mode 3, 832, 844
Color Compare register, in read mode 1, 531
Color Don't Care register, in read mode 1, 534
Color Select register, color paging, 628-629
Data register, loading DAC, 642-643
Data Rotate register
  barrel shifter, controlling, 463
  vs. CPU-based rotations, 489
effect on ALUs, 452
Enable Set/Reset register
  setting drawing color, 666
  specifying plane, 474
Graphics Mode register
  read mode 0, selecting, 525
  read mode 1, selecting, 531
  and high-level languages, 548
Horizontal Pel Panning register, 442
internal indexing, 427-429
Line Compare register, split screen operation, 565
Map Mask register
drawing text, 833
optimizing Mode X, 1074
vs. Read Map register, 526
selecting planes for CPU writes,
443-444, 471-472
with set/reset circuitry, 474
write mode 1, 444
Mask register, blanking screen, 651
Mode Control register, pel panning in split screen, 575
Mode register
color paging, 628-629
256-color modes, 629
Overflow register, split screen operation, 565
palette RAM registers, setting, 631-632
Pel Panning register, 574, 583
Read Index register, 651-652
Read Map register
plane, selecting, for CPU reads, 526
planes, specifying to be read, 542
Set/Reset register, setting drawing color, 666
setting, 504, 558
setting and reading, 582
Start Address High and Low registers,
834-836
and VGA BIOS, 458
Write Index register
DAC index wrapping, 651
loading DAC, 642-643
VGA (Video Graphics Adapter)
ALU and latch demo program, 453-457, 458-460
architecture, 426-429
ALUs, 451-452
barrel shifter, 463-464
bit mask, 464-471
latches, 452-453
set/reset circuitry, 471-479
ball animation demo program, 431-441
CGA compatibility, 430
delay sequences, 558
display memory, 446
fill patterns, drawing, 453
GC (Graphics Controller), architecture,
451-453, 463-479
I/O access times, 842-844
linear addressing, 430
memory access times, 842-844
overview, 426
page flipping, 444-446
panning, 441-443
performance, with 486
processor, 842-844
potential incompatibilities, 446-447
registers, internal indexing, 425-429
screens, capturing and restoring, 541-542, 543-547, 547-548
split screens
horizontal panning, 574-575, 575-582, 583
overview, 563-565
registers, setting, 573
safety of, 585
split screen demo program, 565,
566-572, 572
text mode, 584
turning on and off, 565
25 MHz clock and 28 MHz clock,
switching between, 620-621
virtual screens
overview, 430
panning, 441-443
Video function 10H, EGA BIOS,
550-551, 555
Viewing angle, and BSP tree rendering,
1157-1158
Viewing variance, 1249
Viewsplace
defined, 1135
and normals of polygons, 1137-1138
in 3-D transformation, 935
transformation to, BSP rendering, 1158
uses for, 967
Viewspace clipping, 1207
Virtual screens
overview, 430
panning, 441-443
Visibility determination
See also Visible surface determination.
and BSP trees, 1099-1106
Visibility of polygons, calculating, 955-956
Visible surface determination (VSD)
beam trees, 1185-1189
culling to frustum, 1181-1184
overdraw problem, 1184-1185
polygon culling, 1181-1184
and portals, 1279-1280
potentially visible set (PVS),
precalculating, 1188-1189
V-pipe, Pentium processor
branch instructions, 404-405
overview, 385-386
V-pipe-capable instructions, 386-387
VQuake, 1287-1280
VSD. See Visible surface determination (VSD).

W

Wait30Frames function, 854
Wait states
display memory wait states
8088 processor, 101-103
286 processor, 220
vs. DRAM refresh, 100
overview, 99
system memory wait states, 210-213
WaitForVerticalSyncEnd subroutine, 569, 579-580
WaitForVerticalSyncStart subroutine, 569, 579
WalkBSPTree function, 1106
WalkTree function
code recursive version, 1108
data recursive version, 1109-1110
Wall orientation testing, BSP tree rendering, 1160-1161
WC word counting program (Terje Mathisen), optimization, 250-252, 306, 319
Williams, Rob, 174
Winnie the Pooh orbiting Saturn, 1047
WinQuake, 1280
Word alignment, 286 processor
code alignment, 215-218
data alignment, 213-215
stack pointer alignment, 218-219
Word count program
edge triggered device, 316
fine-tuning optimization, 312-313
initial C implementation, 299
lookup table, 303, 304, 317-319
ScanBuffer assembly routine
author's implementation, 301-302
Stafford, David's, 309-311, 317-319
Willem Clements' implementation, 313-315
as state machine, 315
theoretical maximum performance, 319
Word-OUT instruction, 429
Word-sized variables, 8088 processor
memory access, 82
optimization, 83-85
World, drawing, in Quake 3-D engine, 1280-1281
Worldspace
defined, 1135
in 3-D transformation, 935
uses for, 967
Write Index register
DAC index wrapping, 651
loading DAC, 642-643
Write mode 0
drawing text, 832-833
vs. write mode 2, 503
Write mode 1
overview, 444
vs. write mode 3, 490
Write mode 2
chunky bitmaps, converting to planar,
vs. Bit Mask register, 844
character/attribute map, 517
drawing bitmapped text, 484-489,
489-490, 490-496
drawing solid text, 1039-1041.
1042-1044
graphics, preserving on switch to, 515-517, 518-521
overview, 483-484, 496
single-color drawing, 831-832
vs. write mode 1, 490
Write mode 3 demo program, 484-489, 489-490, 490-496
Write modes, VGA
and set/reset circuitry, 478
text, drawing, 484, 490, 496
Write-after-write register contention, 404
WritePixel subroutine, 597, 599
WritePixelX subroutine, 883-884
Writing pixels
320×400 256-color mode, 599, 600
Wu antialiasing algorithm
assembly implementation, 787-791
C-language implementation, 780-786
description, 776-778, 791-792
error accumulation, 778-779, 792
performance, 777-778
pixel intensity calculations, 778-779
Wu, Xiaolin. See Wu antialiasing algorithm.

X

X86 family CPUs
See also 8088 processor.
32-bit division, 181-184, 1008
branching, performance, 140
copying bytes between registers, 172
interrupts, 9
limitations for assembly programmers, 27
lookup tables, vs. rotating or shifting, 145-146
LOOP instruction vs. DEC/JNZ sequence, 139
machine instructions, versatility, 128
memory addressing modes, 129-133
overview, 208
transformation inefficiencies, 26
XCHG instruction, 377, 832
X-clipping, in BSP tree rendering, 1159
XformAndProjectObject function, 974
XformAndProjectPoints function, 960
XformAndProjectPoly function, 944-945
XformVec function
assembly implementation, 996-997,

1017-1019
C implementation, 943, 976
XLAT instruction
in Boyer-Moore algorithm, 274-277
byte registers, 243
with lookup table, 304
XOR instruction, vs. NOT, 147
X-Sharp 3-D animation package
AppendRotationX function, 975
AppendRotationY function, 964-965, 975
AppendRotationZ function, 965, 976
code archives on diskette, 985
ConcatXforms function
assembly implementation, 997-999, 1019-1022
C implementations, 944, 976
CosSin subroutine, 994-996, 999, 1013-1015
DDA (digital differential analyzer)
texture mapping
assembly implementation, 1069-1073, 1074
C implementation, 1053-1058
disadvantages, 1052-1053, 1059
DrawTexturedPolygon, 1055-1056
hardware dependence, 1053
multiple adjacent polygons, 1068
optimized implementation, 1069-1073, 1074
orientation independence, 1065-1067, 1067
performance, 1074
ScanOutline function, 1058-1059, 1067
SetUpEdge function, 1057-1058
StepEdge function, 1056-1057
techniques, 1048-1051
DrawObject function, 978-979
ambient and diffuse shading support, 1025-1027
FixedDiv subroutine, 982, 993, 1010-1012
FIXED_MUL macro, 1016-1017
FixedMul subroutine, 981, 993-994, 1009-1010
InitializeCubes function, 980-981
InitializeFixedPoint function, 977
matrix math, assembly routines, 992, 996-999
ModelColorToColorIndex function, 1036, 1038
older processors, support for, 1007-1008, 1008-1023
OVERVIEW, 984-985
POLYGON.H header file, 982-984
RGB color model
  mapping to 256-color mode, 1036, 1037-1038, 1039
  overview, 1034-1035
RotateAndMovePObj ect function, 977-978
XformAndProjectPObj ect function, 974
XformVec function
  assembly implementation, 996-997, 1017-1019
  C implementation, 976
XSortAET function
  complex polygons, 748
  monotone-vertical polygons, 769

Y

Yaw angle, in polygon clipping, 1206
Y-clipping, in BSP tree rendering, 1159

Z

Z-buffers
  performance, 1213
Quake 3-D engine, 1285-1286
  vs. sorted spans, 1215
sorting moving models, 1212-1213
Z-clipping, in BSP tree rendering, 1158
Zen timer
  See also Long-period Zen timer.
  calling, 48
  calling from C code, 69-72
  and DRAM refresh, 99
  and interrupts, 43
  interval length, 45
  overhead of, timing, 46, 72
  PC compatibility, 48-49
  precision, 48, 52
  prefetch queue cycle-eater, 88, 92
PS/2 compatibility, 66
PZTEST.ASM listing, 49
PZTIME.BAT listing, 51
PZTIMER.ASM listing, 35-42
ReferenceZTimerOff subroutine, 41
ReferenceZTimerOn subroutine, 40
reporting results, 47
starting, 43
stopping, 46
system clock inaccuracies, 43, 45-46, 48
test-bed program, 48-52
TESTCODE listing, 50
timing 486 code, 245-246
ZTimerOff subroutine, 38-41, 46-47
ZTimerOn subroutine, 37-38, 43
ZTimerReport subroutine, 41-42, 47-48
Zero-wait-state memory, 211
Z-order display, masked images, 872
Z-sorting, for hidden surface removal, 1220-1222
ZSortObjects function, 1201
ZTimerOff subroutine
  long-period Zen timer, 59-63
Zen timer, 38-41, 46-47
ZTimerOn subroutine
  long-period Zen timer, 58-59
Zen timer, 37-38, 43
ZTimerReport subroutine
  long-period Zen timer, 63-65
Zen timer, 41-42, 47-48