

A

- abs function 113
- abs_diff function 113
- absolute values in vector
 - comparisons 249
- absolute values, returning 103, 113
- Accelerated Parallel Processing 351
- add_sat function 110
- addition assignment operators 96
- addition functions 110
- addition operator 96
- address spaces
 - constant and global, combined 88
 - kernel argument qualifiers 70
 - list of 88, 159
 - modifiers for 85
 - qualifiers for 88–90
 - size, determining 88
 - speed of accessing 88, 159
 - transferring data between 101
- addressing mode, for images 126–127
- addressing_mode parameter (clCreateSampler function) 125–126
- alignment, memory 90
- all function 97, 118
- allocating memory. *See* memory allocation
- AMD
 - Catalyst 354, 358
 - custom installation 354
 - download site 354
 - driver 358
 - driver compatibility 352
 - express installation 354
 - fglrx 358
 - fglrxinfo 358
 - installation script 359
 - license agreement 355
 - SDK 351, 362
 - APP. *See* Accelerated Parallel Processing
 - system requirements 352
 - AMD Aparapi. *See* Aparapi
 - AMD compiler 31
 - AMD development file variable 410
 - AMD SDK (software development kit) 14
 - animation in OpenGL 335–338
 - anti-aliasing, enabling for GLUT window 384
 - any function 97, 118
 - Aparapi
 - aparapi.dll location 198
 - documentation 198
 - drawbacks of 201
 - installing 198
 - Kernel class 198–200
 - license agreement for 197
 - methods, similarity to C kernel's 199
 - operating systems supported by 197
 - overview of 197
 - vs. C++ Wrapper API 197
 - web site for 198
 - work-items, configuring 200–201
 - AparapiItems.java file 201
 - AparapiRound.java file 198
 - Apple Inc., OpenCL development and 4–5
 - applications, host. *See* host applications
 - arguments for clGetDeviceIDs function 22–23
 - See also* kernel arguments
 - arithmetic functions 103
 - arithmetic operators 96, 161
 - See also* operators
 - arithmetic overflow 110–111
 - array function 216
 - arrays, reducing. *See* reduction
 - arrays, scalar
 - loading vectors from 101–102
 - vector storage in 102–103
 - assignment operators 96
 - async_work_group_copy function 165
 - async_work_group_strided_copy function 165
 - asynchronous data transfer 164–166
 - atomic multiply-and-add (MAD) operations 32
 - atomic operations 160–163
 - for mutex check-unlocking 163
 - testing 162
 - atomic_add function 244
 - atomic_cmpxchg function 163

atomic_inc function 244
 atomic.cl file 162
 automatic variables 407

B

Back 40 radix sort 256
 barrier commands 150
 barrier function 244
 address spaces, specifying 160
 in bitonic sort 253
 in reduction algorithm 228
 signature of 159
 See also fence functions
 basic_interop.c file 332–334
 big-endianness. *See* endianness
 bilinear interpolation
 128, 135–137
 Binaries data type 174
 binary buffer array,
 returning 33
 binary files, creating programs
 from 31
 binding vertex buffer objects
 (VBOs) 369
 bindings 197
 See also C++ Wrapper API
 bit reversal 312
 bitonic sort 244–253
 bitonic conversion 246–247
 bitonic merge 246, 251
 bitonic sets 245
 bitonic split operation 245
 coding 247–253
 creating bitonic
 sequences 246–247
 for loops in 253
 kernel-execution
 commands 250–251
 overview of 237
 sorting bitonic
 sequences 244–246
 sorting elements between
 vectors 249
 sorting vector elements
 with 248
 stages for 250
 work-group assignments 250
 See also radix sort
 bitselect function 116–118
 blocking argument (clEn-
 queueXX functions) 55
 box filters 342–343
 branch miss penalties,
 avoiding 233

bsort.cl file 251
 buckets, sorting digits into 254
 Buffer class
 constructor 216–217
 functions in 211
 buffer objects
 checking size/location of 52
 copying data between 60
 copying data to image
 object 60
 dimensionality,
 configuring 64
 host pointers 46
 host-accessible memory, con-
 straining to 46
 information parameters 52
 information, retrieving 52–54
 mapping, in C++ 187
 memory allocation 45–47
 pixel buffer objects
 (PBOs) 346–347
 properties, setting 45
 read/write permissions 45–46
 read/writing 54–58
 read/writing, in C++ 185–187
 rectangular data in
 56–57, 186
 sub-buffer objects,
 creating 47–48
 VBO data access with 333
 buffer_check application 52
 buffer_test application 57
 buffer_test.cpp file 186
 buffers, OpenGL
 creating from VBOs 326
 initializing 387–388
 resetting 390
 build errors, troubleshooting 34
 build function (Program
 class) 174
 build function signature 214
 build log
 compile errors,
 displaying 206
 reviewing 34–35, 175
 reviewing, in OpenGL
 372, 374
 reviewing, in PyOpenCL 214
 build method (CLProgram
 class) 206
 butterfly diagrams 310
 byte order 72–73

C

C++
 advantages of, over C
 167–168
 exception handling 170
 kernel arguments in 176–183
 profiling events in 192–193
 string classes 169
 vector classes 169
 vectors, dynamic allocation
 of 171
 See also object-oriented
 programming
 C++ Wrapper API
 advantages of 167–168
 cl::Buffer class 178–179
 cl::Image class 179–181
 cl::Memory class 177
 cl::string class 169
 cl::vector class 169
 CommandQueue class
 183–189
 Context class 172–173
 Device class 172–173
 Event class 189–194
 Kernel class 173–176
 Platform class 171–172
 Program class 173–176
 read-write commands
 185–187
 UserEvent class 192
 vs. Aparapi 197
 wait functions (Event
 class) 193
 callback functions
 associating events with
 142–143
 defined 141
 Event objects and 190
 example code 143–144
 GLUT and 384
 in C++ 190
 signature for 142, 190
 callback.cpp file 190
 card game analogy 8–10
 catch blocks 170
 ceil function 103
 CG method. *See* Conjugate
 Gradient method
 channel types for image
 objects 49
 char array allocation 20
 char data type 71
 charn data type 78, 242

- cl_callback macro 142
- cl_channel_order parameter (clCreateImage functions) 49
- cl_channel_order parameter (ImageFormat function) 180
- cl_channel_type parameter (clCreateImage functions) 49
- cl_channel_type parameter (ImageFormat function) 180
- cl_command_queue structures 40
See also command queues
- cl_command_queue_properties parameter (clCreateCommandQueue function) 40
- cl_context structure 28–29
- cl_context_properties parameter (clCreateContext function) 26, 324
- cl_context_properties structure, for OpenGL interoperability 27
- cl_context_reference_count parameter (clGetContextInfo function) 28
- cl_device_endian_little parameter (clGetDeviceInfo function) 73
- cl_device_fp_config data type 76
- cl_device_id array 22
- cl_device_image_support parameter (clGetDeviceInfo function) 124
- cl_device_info type parameters 23–24
- cl_device_preferred_vector_width_type parameter (clGetDeviceInfo function) 79
- cl_event data structures as user events 152 creating 142 wait lists and 145
See also events
- cl_event_info argument (clGetEventInfo function) 151–152
- cl_float data type 17
- cl_image_format argument (clCreateImage functions) 49
- cl_image_format structures 125
- cl_kernel data structure 36
See also kernels
- cl_khr_3d_image_writes extension 132
- cl_khr_fp64 extension 71
- cl_khr_icd 352
- cl_mem data structures 45, 124
See also memory objects
- cl_mem_flags parameter (clCreateBuffer function) 45
- cl_platform_id structure 18–22
See also platforms
- cl_platform_info data type 19
- CL_PLATFORM_NOT_FOUND_KHR error 361
- cl_program structure 31
See also programs
- cl_program_build_info parameter (clGetProgramBuildInfo function) 34
- cl_program_info parameter (clGetProgramInfo function) 33
- cl_sampler data structures 124
- cl_sampler objects 125, 128
See also samplers
- cl::Buffer class 178
- cl::Error class 170
- cl::Image objects 180
- cl::LocalSpaceArg objects 182
- cl::Memory class 177
- cl::string class 169, 173
- cl::vector class 169
- clamp function 105, 113
- clamping image color 127–128
- classes, C++
 - cl::Buffer 178–179
 - cl::Image 179–181
 - cl::Memory 177
 - CommandQueue 183–189
 - Context 172–173
 - Device 172–173
 - Event 189–194
 - for strings 169
 - for vectors 169
 - Kernel 173–176
 - Platform 171–172
 - Program 173–176
 - UserEvent 192*See also* specific classes
- classes, JavaCL 202–203
- classes, PyOpenCL 211
- CLBuffer class 209
- CLBuffer objects
 - as generic type 207
 - creating 207–208
- CLBuildException class 206
- clBuildProgram function 31–33, 74, 174
- clc compiler 31
- CLContext class 205
- clCreateBuffer function
 - host pointer parameter 46
 - signature for 45
- clCreateCommandQueue function
 - cl_queue_profiling_enable flag 153
 - parameters for 40
 - signature for 40
- clCreateContext function
 - callback function as argument 27
 - OpenGL, referencing with 324
 - signature for 26
- clCreateContextFromType function
 - C++ equivalent for 172
 - callback function as argument 27
 - signature for 26
- clCreateFromGLBuffer function 326
- clCreateFromGLRenderBuffer function 327
- clCreateFromGLTexture functions 327
- clCreateFromGLTexture2D function 346
- clCreateImage2D function 48–51
 - dimension and pitch arguments 50
 - signature for 124
- clCreateImage3D function 48–51
 - dimension and pitch arguments 50
 - signature for 124
- clCreateKernel function 37
- clCreateKernelsInProgram function 37–38
- clCreateProgramWithBinary function 30
 - Python equivalent 213
 - signature for 31

- clCreateProgramWithSource
 - function 30
 - Python equivalent 213
 - signature for 30
- clCreateSampler function
 - parameters for 125–126
 - signature for 125
- clCreateSubBuffer function 47
- clCreateUserEvent function 146
- clEnqueueAcquireGLObjects
 - function 328
- clEnqueueBarrier function 150
- clEnqueueCopyXX
 - functions 60
- clEnqueueMapXX functions 59
- clEnqueueMarker
 - function 148–149
- clEnqueueNDRangeKernel
 - function 62–63, 97
 - example application 100–101
 - example of 65
 - global_work_offset
 - argument 64
 - global_work_sizes
 - argument 64
 - local_work_size argument 66
 - non-blocking nature of 157
 - signature of 157
- clEnqueueReadBuffer
 - function 70, 141
- clEnqueueReleaseGLObjects
 - function 328
- clEnqueueTask function 40
- clEnqueueTask method
 - (CLKernel class) 208
- clEnqueueWaitForEvents
 - function 149–150
- clEnqueueXX functions
 - blocking argument 55
 - cl_events, creating 141–142
 - cl_mem argument 54
 - cl_uint argument 145
 - const argument 145
 - non-blocking nature of 144, 148
 - offset argument 56
 - origin argument 56
 - region argument 56
 - timing, comparing 155–156
 - when to call 144
- CLEvent object 208
- clFinish function 157, 328
- clGetContextInfo function 28
 - code example for 29–30
 - getInfo equivalent for 171
- clGetDeviceIDs function 22
- clGetDeviceInfo function
 - 23–24, 154
 - checking double data type
 - support 71
 - checking double data type
 - with 74
 - checking half data type
 - support 75
 - endianness, determining
 - with 73
 - getInfo equivalent for 171
 - IEEE-754 compliance
 - checking 76
 - preferred vector width, determining
 - with 79–80
- clGetEventInfo function
 - cl_event_info argument
 - parameters 151–152
 - getInfo equivalent for 171
 - signature of 151
- clGetEventProfilingInfo
 - function 153
 - example code 154
 - parameter values for 154
 - signature of 153
- clGetExtensionFunctionAddress
 - function 330
- clGetGLContextInfoKHR
 - function 330
- clGetGLObjectInfo
 - function 329–330
- clGetGLTextureInfo
 - function 329
- clGetKernelInfo function
 - 37–38, 171
- clGetKernelWorkGroupInfo
 - function 223
- clGetMemObjectInfo
 - function 52–54, 171
- clGetPlatformIDs function
 - 18–19
- clGetPlatformIds function 204
- clGetPlatformInfo function
 - 19–20
 - getInfo equivalent for 171
 - parameters for 20
- clGetProgramBuildInfo
 - function 33, 171
- clGetProgramInfo function 33
- clKernel class
 - kernel-execution
 - methods 208
 - setArg methods for 206
- close function 214
- clReleaseContext function
 - 29–30
- clReleaseKernel function 38
- clReleaseMemObject
 - function 125
- clReleaseSampler function 128
- clRetainContext function 29–30
- clRetainSampler function 128
- clSetEventCallback function
 - arguments for 142
 - C++ equivalent 190
 - signature for 142
 - when to call 144
- clSetKernelArg function 69, 92
 - cl_mem object and 125
 - invoking with primitive
 - data 92
 - NULL argument, invoking
 - with 91
 - signature for 44, 90
 - value parameter 44
- clSetUserEventStatus function
 - calling 147
 - signature of 146
- cluster computing. *See*
 - MapReduce
- clz function 118–119
- code. *See* host code
- color coordinates
 - beyond image size 126–127
 - clamping 127
 - formats for 126
- color intensity 126
- color normalization 126
- column vectors 267
- command events
 - creating 145
 - defined 145
 - in C++ 191
 - vs. user events 146
 - wait lists and 145–146, 191
 - See also* user events
- command profiling
 - configuring 153–155
 - example code 154
 - for data transfer 155–156
 - parameters for 154
 - timing values, returning 154
- command queues
 - allowing out of order kernel
 - processing 40
 - command operations 39
 - creating 183
 - creating, default 205
 - creating, in PyOpenCL 213

- command queues (*continued*)
 - execution order in 10
 - functions for enqueueing 54
 - in C++ 183–189
 - kernel distribution and 7
 - kernel execution
 - commands 40–41
 - overview of 39, 54
 - processing order in 40
 - returning 151
 - targeting devices in 39
 - See also* `cl_command_queue` structures; data structures
- command statuses 151
- command synchronization
 - blocking parameters and 148
 - example code 147
 - in C++ 191–192
 - overview of 145
 - wait functions 193
 - with barrier commands 150
 - with command events 145–146
 - with marker commands 148–149
 - with user events 146–148
 - with wait commands 149–150
- command types 151–152
- CommandQueue class
 - constructor 213
 - functions in 211
- CommandQueue objects
 - constructor 183
 - creating 183
 - functions for 188
- CommandQueue objects (PyOpenCL) 213
- commands
 - associating events with 141–142
 - defined 39
 - life-cycle of 151
 - timing, returning 154
- comments 404, 406
- comparison functions 105
- comparison functions (vectors) 118–122, 242
- `compile_shader` function 389
- compilers
 - build function of Program class 174–175
 - `clBuildProgram` function and 31–33
 - defined 14
 - OpenCL version, setting 32
 - options for 32
 - run-time 31
 - vendor-specific 31
- compiling applications
 - Linux, Nvidia device 360–361
 - Windows
 - AMD 356
 - Nvidia device 357–358
- compiling OpenCL 6
- compiling PyOpenCL 210
- complex numbers 300
- components, vector
 - assigning equal to other components 114
 - data types for 114–115
 - defined 78
 - hi/lo/even/odd selection method 83–84
 - initializing 80
 - leading zeros, counting 119
 - letter-indexing 82–83
 - maximum number of 81
 - normal, checking 120
 - number-indexing 81–82
 - selection methods for 80
 - syntax for 80
 - test functions 97
 - true/false returns 97
- compute units
 - defined 66
 - work-groups in 87
- concurrency 7
 - See also* parallel programming
- `configure_data_objects` function 332
- `configure_shared_data` function 335, 346
- `conj_grad.c` file 291
- Conjugate Gradient method
 - algorithm for 291
 - example code 291–293
- conjugate vectors 290
- constant memory 88–89
- constants
 - floating-point 108
 - transferring with compiler defines 232
- Context class
 - constructor 212
 - functions in 211
- Context objects 172–173
 - constructor 172
- Context objects (PyOpenCL) 212–213
- contexts
 - creating 26–28
 - creating, in PyOpenCL 212–213
 - creating, with JavaCL 204
 - defined 7, 25
 - for devices of given type 26
 - handle, returning 324
 - in card game analogy 25
 - information parameters 27–28
 - information, retrieving 28
 - Linux configuration 324–325
 - Mac OS configuration 325
 - necessity of 30
 - number of devices, returning 28
 - OpenGL 27
 - OpenGL, creating 332
 - OpenGL, referencing with 323–325
 - OpenGL, returning information on 330
 - platforms and 25–26
 - reference count 28–29
 - reference count, checking 30
 - reference count, returning 28–29
 - retrieving 33, 151
 - single-platform nature of 25
 - Windows configuration 324
 - See also* data structures
- contrast, image 135
- Cooley-Tukey algorithm
 - for four-element fast Fourier transform 310–312
 - for two-element fast Fourier transform 309–310
 - overview of 309
 - See also* fast Fourier transform
- coordinates, Euclidean 120
- coordinates, spherical 337–338
- `copySign` function 109
- `copyTo` method (CLBuffer class) 209
- cosine functions. *See* trigonometric functions
- `create_kernel.py` file 214
- `create_kernels.cpp` file 176
- `create_queue.py` file 213
- `create_some_context` function (pyopencl module) 213
- `createBuffer` function (JavaCL) 207

- createContext method (JavaCL)
 - arguments for 205
 - signature for 204
 - createDefaultQueue method (JavaCL) 205
 - createImage2D method (CLContext class) 208
 - createImage3D method (CLContext class) 208
 - createKernel method (JavaCL) 206
 - createKernels function signature 175
 - createKernels method (JavaCL) 206
 - createProgram methods (CLContext class) 205
 - createSubBuffer function (cl::Buffer class) 178
 - cross function 120, 122
 - cross products of vectors, returning 120–122
- D**
-
- darkening images 133–135
 - data partitioning 10
 - importance of 62
 - in matrix-vector multiplication 65
 - performance improvement with 158
 - profiling 157–158
 - data structures
 - card game analogy for 8–10
 - memory alignment 90
 - naming convention for 13
 - reference count, tracking 28
 - See also* specific data structure types
 - data transfer
 - asynchronous 164–166
 - enqueueing commands for 185–187
 - JavaCL commands for 209–210
 - JavaCL, example code for 209
 - overview of 155
 - profiling 155–156
 - status, checking 166
 - data transfer operations 101–102
 - data types
 - binaries 174
 - double 71–72
 - for shaders 378
 - in mask vectors 114–115
 - integer 109
 - OpenGL 369
 - operating on 18
 - primitive 17–18
 - scalar 70–71
 - similarity to C/C++ data types 18
 - sources 174
 - vector 77–85
 - See also* specific data types
 - data_size argument (clEnqueueXX functions) 56
 - data-parallelism 7
 - deallocating image objects 125
 - deallocating kernels 38
 - Dean-Ghemawat MapReduce paper 238
 - example tasks mentioned in 240
 - grep recommendation 242
 - See also* MapReduce
 - decrement operators 96
 - Deep Color image format 50
 - delaying events. *See* wait lists
 - denormals
 - disabling processing of 231
 - in float data type 74
 - support for 74
 - treating as zero 32, 74
 - dense matrices 260
 - dependency files 405
 - dependency statements, in a makefile 404
 - derivative work, in copyright law 202
 - determinant function 380
 - development systems. *See* hosts
 - Device class 211
 - device extensions 14
 - defined 24
 - testing 23–25
 - vs. platform extensions 24
 - device model
 - MapReduce and 240–242
 - math student analogy for 85–88
 - Device objects 172–173
 - accessing 171
 - checking time precision of 193
 - constructor 172
 - creating 172
 - creating Context objects from 172
 - example code 172
 - Device objects (PyOpenCL) 212
 - devices
 - address width, checking 72
 - command queues for 39
 - defined 22
 - endianness, determining 72–73
 - finding one with most compute units 204
 - host processor, identifying 23
 - identifying all 23
 - image object naming convention for 124
 - information parameters 23
 - information, retrieving 23–24
 - kernel distribution among 7
 - resolution, timing 154
 - returning information on 224
 - returning list of 204
 - targeting, in command queue 39
 - types 23
 - types, creating contexts by 26
 - with GPUs, identifying 23
 - See also* data structures
 - diagonals, matrix 259
 - differential equations 278–279
 - See also* sparse matrices
 - digital remastering 296
 - dimensionality 64
 - cl::Image objects and 180
 - matrix transposition and 261
 - returning 157
 - dimensions
 - in work-items 98–99
 - of image objects 50–51, 128
 - Direct3D devices 27
 - directFloats method (NIOUtils class) 210
 - disabling extensions 71
 - discrete Fourier transform
 - algorithm for 297
 - complex numbers 300
 - dot products and 298
 - equation for 301–302
 - example of 302–305
 - frequencies of interest 299
 - inverse 302
 - magnitude 307
 - mathematical notation 298
 - matrix for 302
 - OpenCL code for 305–306

discrete Fourier transform
(continued)
 properties of 306
 real-valued 304
 scaling 302
 sequences 298
 shifting property 306–308
 single-frequency vectors 299
 sinusoids 299–301
 stretching property 306, 308
 superposition property 306–307
 theory of 298–305
See also fast Fourier transform
 display function 334, 336–337, 348–349, 386, 389–390
 display modes (GLUT) 384
 distance function 120
 division operators 96
 dot function 120–121
 dot products
 discrete Fourier transforms and 298
 in matrix multiplication 262–263, 267
 one-dimensional vs. two-dimensional 341
 vector direction and 265
 vs. outer products 267
 dot products of vectors, returning 120–121
 double data type 71–72
 checking support for 74
 floating-point multiply-and-add (FMA) operation 75
 IEEE-754 format for 74
 requirements for 75
 rounding 75
 double_test.cl file 71
 doublen data type 78
 drawing modes (OpenGL) 375–376
 drawing shapes. *See* primitives

E

effects, image. *See* image filtering
 elements, matrix 259
 embarrassingly parallel tasks 240–242
See also MapReduce
 embedded OpenCL 412–414
 embossing images 344–345
 enabling extensions 71

endianness
 determining 72–73
 vector storage and 84–85
 engraving images. *See* embossing images
 enlarging images 138
See also scaling images
 enqueue_nd_range_kernel function 218
 enqueue_task function 218
 enqueueBarrier function (CommandQueue function) 194
 enqueueCopyBuffer function (CommandQueue class) 188
 enqueueCopyBufferToImage function (CommandQueue class) 188
 enqueueCopyImage function (CommandQueue class) 188
 enqueueCopyImageToBuffer function (CommandQueue class) 188
 enqueueing commands 54
 C++ functions for 183
 for read-write data transfer 185–187
 functions for 54
 in JavaCL 209–210
See also command queues
 enqueueMapBuffer function (CommandQueue class) 187–188
 enqueueMapImage function (CommandQueue class) 187–188
 enqueueMarker function (CommandQueue class) 194
 enqueueNDRange function (CLKernel class) 203, 208
 arguments for 208
 enqueueNDRangeKernel function (CommandQueue class) 184–185
 enqueueTask function (CLKernel class) 203
 enqueueTask function (CommandQueue class) 183
 enqueueUnmapMemObject function (CommandQueue class) 187–188
 enqueueWaitForEvents function (CommandQueue class) 194

environment variables
 355, 357–360
 advanced system setting 355
 AMDAPPSDKROOT 355–356, 359, 361
 AMDAPPSDKSAMPLESROOT 355
 dialog 355
 LD_LIBRARY_PATH 359
 NVSDKCOMPUTE_ROOT 357–358, 361
 PATH variable 360
 system property 355, 357
 system variable 355, 357
 user variable 355, 357
 environment variables in Windows 400
 equals sign (=), data transfer with 101
 erf/erfc function 106
 error checking
 in PyOpenCL 215
 removing 32
 error codes
 defined 169
 returning 170
 error parameter (clCreateContext function) 28
 error-checking routines 11
 Euclidean space 120
 Euler-Bernoulli equation 279
 event handling 384
 Event objects
 associating with commands 189
 callback functions and 190
 creating 190
 wait functions 193
 wait lists and 191
 event_t data structures 166
 events
 associating with callback functions 142–143
 associating with commands 141–142
 command
 synchronization 145–153
 command types,
 returning 151–152
 defined 140
 profiling. *See* profiling events
 reference count, returning 151
 retrieving information
 on 150–153
See also cl_event data structures; command events

exception handling 170
 execute method (Kernel class) 199–200
 execute_kernel function 333, 347–348
 exp/expm1 function 106
 exp2/exp10 function 106
 exponential functions 106
 extensions 14
 approved vs. unapproved 14
 enabling/disabling 71
 naming convention for 14, 20
 testing platforms for 20–22
 See also device extensions

F

$f(z)$, deriving 286
 factoring matrices 258, 269
 fast Fourier transform
 bit reversal 312
 butterfly diagrams 310
 constructing 309–312
 direction, setting 315
 float vectors and 315
 four-element 310–312
 frequency components 309
 history of 306
 loading data from global memory 314
 loops in 316
 merge process for 311
 nodes 310
 OpenCL code for 312–317
 superposition property 310
 two-element 309–310
 work-item
 synchronization 312
 See also Cooley-Tukey algorithm; discrete Fourier transform
 fast_distance function 120
 fast_length function 120
 fast_normalize function 120
 fdim function 105
 FEA (finite element analysis). *See* finite element analysis (FEA)
 fence functions 160
 See also barrier function
 fft.cl file 314
 file command (MinGW) 402
 filter_mode parameter (clCreateSampler function) 125, 128
 filtering images. *See* image filtering
 finite element analysis (FEA) 279
 finite math, requiring 32
 first-in, first-out (FIFO) principle 40
 first-person shooters 340
 flags parameter (cl::Buffer class) 178
 float data type 71
 IEEE-754 format for 73
 rounding 74
 values in 74
 float_config.c file 76
 float4 vector processing 338
 floating-point computing 73–77
 configuration parameters 76
 run-time exceptions 74
 floating-point constants 108
 floating-point multiply-and-add (FMA) 75
 floating-point processing
 comparison functions 105
 functions for 103–109
 operators for 95–97
 single-precision constants, requiring 32
 floating-point values, computing modulus 104
 floatn data type 78
 floor function 103
 FMA (floating-point multiply-and-add) 75
 fma function 103
 hardware processing, testing for 104
 speed improvement with 233
 vs. mad function 104
 fmax function 105
 fmin function 105
 fmod function 103–104
 for loops
 coding iterations separately 231
 in bitonic sort 253
 Fortran
 enabling in MinGW 402
 Fourier transform. *See* discrete Fourier transform; fast Fourier transform
 Fourier, Joseph 298
 FPS (first-person shooter) 340
 fract function 109
 fragment shaders 382–383
 compiling 331, 345
 creating 373
 example code 382
 functions of 382
 textures in 395–396
 uniforms and 395
 frameworks 14
 freglut project 366
 frequencies of interest 299
 frequency 297
 frequency analysis 296–298
 components 297
 frequency-domain signals 297
 magnitude 307
 of shifted sequences 307
 oscillation 297
 speed of 306
 stretching frequency 308
 time-domain signals 296
 frequency components 297
 frequency-domain sequences
 conversion. *See* discrete Fourier transform
 frequency-domain signals 297
 frexp function 109
 full_context.cpp file 172
 functions
 arithmetic 103
 callback 141–144, 190, 384
 clBuildProgram 31–33
 clCreateBuffer 45
 clCreateCommandQueue 40
 clCreateContext 26–28
 clCreateContextFromType 26–28
 clCreateImage2D 48, 50–51
 clCreateImage3D 48, 50–51
 clCreateKernel 37
 clCreateKernelsInProgram 36, 38
 clCreateProgramWithBinary 30–31
 clCreateProgramWithSource 30
 clCreateSubBuffer 47
 clEnqueueCopyXX 60
 clEnqueueNDRangeKernel 62–63, 65
 clEnqueueTask 40
 clEnqueueXX 55–58
 clGetContextInfo 28–30
 clGetDeviceIDs 22
 clGetDeviceInfo 23–24
 clGetKernelInfo 37–38

functions (*continued*)

- clGetMemObjectInfo 52–54
- clGetPlatformIDs 18–19
- clGetPlatformInfo 19–20
- clGetProgramBuildInfo 33–34
- clGetProgramInfo 33
- clReleaseContext 29–30
- clReleaseKernel 38
- clRetainContext 29–30
- clSetKernelArg 44–45
- comparison 105
- conventions for 95
- exponential 106
- floating-point 103–109
- floor 103
- getImageInfo 51
- inline 233
- integer 109–113
- logarithmic 106
- memory pointers in 95
- rounding 103–104
- select 116–118
- shuffle 114–116
- trigonometric 106
- See also* C++ functions; kernels
- fundamental frequency 299
- Fused Multiply and Add (fma) function 75

G

- Gaussian blur 343
- Gaussian function 343
- gcc command 400
- gcc compiler
 - build process 403
 - command-line operation 403
 - debugging data, including 403
 - directories, specifying 410
 - options for 403
 - preprocessing 404
 - producing object file with 404
- gcc executable 402–404
- gdb (GNU debugger) 403
- general-purpose GPU computing (GPGPU computing) 3
- geometric vector functions 120–122
- get_build_info function 214
- get_global_id function 98
- get_global_offset function 98–99
- get_global_size function 98

- get_group_id function 99
- get_image_channel_data_type function 133
- get_image_channel_order function 133
- get_image_depth function 133
- get_image_dim function 133
- get_image_height function 133
- get_image_width function 133
- get_info function (PyOpenCL) 212
- get_local_id function 99, 228
- get_local_size function 99
- get_num_groups function 99
- get_platforms function (pyopencl module) 212
- get_work_dim function 98
- get_work_id function 228
- getBestContext method (JavaCL) 204
- getBestDevice method (JavaCL class) 204
- getBuildInfo function 175
- getDevices function 171
- getGlobalId method 200
- getGroupId method 200
- getImageInfo function 51
- getImageInfo function (cl::Memory class) 181
- getInfo function 171
- getInfo function (cl::Memory class) 181
- getLocalId method 200
- getMaxComputeUnits method (CLDevice class) 204
- getProfilingInfo function 193
- getShaderInfoLog function 374
- GL Extension Wrangler Package (GLEW)
 - Linux installation 367
 - Mac OS installation 367
 - Windows installation 366
- GL Utility Toolkit (GLUT)
 - canvas region, setting 389
 - display modes 384
 - event-handling functions 384
 - fullscreen window, configuring 383
 - initialization functions 345, 383
 - keystroke function 385
 - Linux installation 367
 - mouse click function 384
 - OpenGL graphics display 387

- rendering the model 389–390
- resizing window 384, 386
- runtime, launching 383
- window display 385–386
- window, configuring 331, 383
- Windows installation 366
- glAttachShader function 373–374
- glBindAttribLocation function 373–374
- glBindBuffer function 368
 - access parameter values 369–370
 - signature of 369
- glBindTexture function 391
- glBindVertexArray function 371
- glBufferData function 336, 347, 368
 - example code 370
 - signature of 369
- glClear function 390
- glCompileShader function 372–373
- glCreateProgram function 372, 374
- glCreateShader function 372–373
- glDeleteBuffers function 368, 370
- glDeleteProgram function 373
- glDeleteTextures function 391
- glDisableVertexAttribArray function 371
- glDrawArrays function 336, 371, 377, 389–390
 - signature of 375
- glEnableVertexAttribArray function 371
- GLenum data type 369
- GLEW (GL Extension Wrangler Library). *See* GL Extension Wrangler Library (GLEW)
- glFinish function 328
- glGenBuffers function 368
- glGenTextures function 391
- glGenVertexArrays function 371
- glGetShaderInfoLog function 372
- glGetShaderiv function 372–374
- glIsBuffer function 368
- glLinkProgram function 373, 375
- global cache, reading data into 165–166

- global memory
 - copying sequential data to/from 165
 - data transfer to/from local memory 101, 165
 - defined 85, 88
 - qualifier for 89
 - sharing space with constant memory 88
 - speed of 88, 222
 - See also* texture memory
 - global size of work-items 222
 - global_work_offset argument (clEnqueueNDRangeKernel function) 64
 - global_work_sizes argument (clEnqueueNDRangeKernel function) 64
 - globalSize argument (enqueueNDRange function) 208
 - glPixelStorei function 391–392
 - glShaderSource function 372–373
 - GLsizei data type 369
 - GLSL (OpenGL Shading Language). *See* OpenGL Shading Language (GLSL)
 - glTexImage2D function 348, 391, 393–394
 - glTexParameterf function 391–392
 - interpolation method, setting 392
 - parameters for 392
 - GLuint data type 369
 - glUseProgram function 373, 375, 389
 - GLUT (GL Utility Toolkit). *See* GL Utility Toolkit (GLUT)
 - glut_intro.c file 385
 - glutCreateWindow function 383
 - glutDisplayFunc function 384–385
 - glutFullScreen function 383
 - glutIdleFunc function 385
 - glutInit function 383
 - glutInitDisplayMode function 383
 - glutInitWindowPosition function 383
 - glutInitWindowSize function 383
 - glutMainLoop function 385–386
 - glutMouseFunc function 384
 - glutPostRedisplay function 336–337
 - glutReshapeFunc function 384–385
 - glutTimerFunc function 385
 - glutVisibilityFunc function 385
 - glVertexAttribPointer function 336, 371
 - glViewport function 389
 - glxinfo command 367
 - GNU automatic variables 407
 - GNU build scripts. *See* makefiles
 - GNU tool access for Windows. *See* MinGW
 - Google MapReduce framework. *See* MapReduce
 - GPGPU (general-purpose GPU) computing 3
 - GPUs (graphics processing units). *See* graphics processing units (GPUs)
 - Gram-Schmidt method for vector orthogonalization 290–291
 - graphics processing units (GPUs)
 - devices with, identifying 23
 - in supercomputers 3
 - graphics rendering. *See* OpenGL
 - graphics textures. *See* textures
 - grayscale images 180
 - grep, implementing with MapReduce 242–244
-
- H**
-
- hadd function 110–111
 - half data type 75
 - halfn data type 78
 - Harwell-Boeing collection 281
 - See also* Matrix Market files
 - header file directories, identifying 32
 - Hello World! function 69–70
 - hello_kernel.cl file 69
 - hi/lo/even/odd method for vector component selection 83–84
 - HighColor image format 50
 - history of OpenCL 4–5
 - host applications
 - context management 25–26
 - defined 7
 - local memory and 91
 - OpenGL vs. OpenCL 365
 - primitive data types for 17–18
 - See also* data structures
 - host code
 - for matrix-vector multiplication 10–13
 - in separate text file 31
 - vs. kernel code 15
 - See also* source code
 - host memory
 - read/write accessibility 59
 - read/writing data to 55
 - host notification events
 - creating 141–143
 - example code 143–144
 - in C++ 189–191
 - process for 141
 - See also* events
 - host pointers from buffer objects 46
 - host_ptr parameter (cl::Buffer class) 178
 - host-accessible memory 46
 - hostbuf argument (PyOpenCL) 216
 - hosts
 - image object naming convention for 124
 - kernel distribution from 7
 - Householder transformation
 - example code 269
 - example of 270–276
 - inverse nature of 272
 - kernel for 273–274
 - matrices, finding 272
 - matrices, storing 273
 - vectors, finding 270–272
 - See also* QR decomposition
 - hyperbolic trigonometric functions 106–107
-
- I**
-
- I/O functions for Matrix Market files 282
 - ICD. *See* installable client driver
 - id_check.cl file 100
 - identity matrices 260, 285
 - IEEE-754 standard 73, 76–77
 - if statements
 - operators for 97
 - sorting scalars with 248
 - ilogb function 109

- Image class
 - constructor 216–217
 - functions in 211
- image coordinates
 - clamping 128
 - format for 132
 - interpolation 128
 - normalized 126, 129
- image filtering 341–345
 - embossing images 344–345
 - Gaussian blur 343
 - matrices and 341–342
 - noise removal 342–343
 - OpenCL coding for 345–349
 - sharpening images 344
 - with box filters 342–343
- image objects
 - 3D, writing to 132
 - addressing mode 126–127
 - advantages of 123
 - as kernel arguments 125, 129
 - bilinear interpolation 135–137
 - bit padding 180
 - channel data type vs. channel order 133
 - channel data type, returning 133
 - channel order, returning 133
 - channel order, specifying 180
 - channel types 49
 - channel types, setting 180
 - checking support for 124
 - clamping output color 127–128
 - contrast, changing 135
 - coordinate types 126
 - copying data between 60
 - copying data to buffer object 60
 - creating 48–51, 124–125
 - creating, from textures/renderbuffers 327, 346
 - creating, in JavaCL) 208
 - data types for 131
 - deallocating memory for 125
 - default modifier for 128
 - defined 124
 - depth, returning 133
 - dimensionality 128
 - dimensionality, configuring 64
 - dimensions 50–51
 - enlarging 138
 - formats 49–50
 - formats, specifying 180
 - grayscale 180
 - height, returning 133
 - in C++ 180–181
 - information parameters 51
 - information, retrieving 51, 181
 - interpolation 128
 - kernel processing of 126
 - naming conventions for 124
 - nearest-neighbor interpolation 135–136
 - normalized colors 126
 - pitch 50–51
 - PNG format, reading 134
 - processing, example code for 133–135
 - read functions 130–132
 - read/write access to 128
 - read/writing 54–58
 - read/writing, in C++ 185–187
 - rectangular data in 56–57, 186
 - row pitch 180
 - samplers and 124
 - scaling 135
 - size, returning as int2 vector 133
 - slice pitch 180
 - slices 50
 - texture memory and 123
 - two-dimensional and three-dimensional 48, 180
 - width, returning 133
 - write functions 132–133
 - zooming in on 128
- Image objects (PyOpenCL) 217
- image processing
 - functions 130–134
- image processing in embedded OpenCL 413–414
- ImageFormat class
 - constructor 217
- ImageFormat function 180
- import library 356
- increment operators 96
- index parameter (clSetKernelArg function) 44
- index spaces 64, 184
- indexes, inverted 240
- infinite numbers
 - disabling processing of 231
 - in float data type 74
- init_buffers function 345, 387
- init_cl function 345–346
- init_gl function 331, 345
- init_shaders function 345, 388
- init_textures function 345
- initializing
 - private memory 91
 - vectors 80
- inline functions 233
- input vectors
 - dot products and 341
 - sorting elements of 249
- installable client driver
 - 352–353, 358, 361, 363
 - default locations 354
- installing OpenCL
 - Linux 358–361
 - AMD device 358–359
 - Nvidia device 359–360
 - Mac OS 361–362
 - Windows 354–358
 - AMD device 354–355
 - Nvidia device 356–357
- installing OpenGL
 - on Linux 366–367
 - on Mac OS 367
 - on Windows 366
- int data type 71
- integer data type 109
- integer functions 109–113
 - addition 110
 - arithmetic overflow 110–111
 - multiplication 111–112
 - returning absolute values 113
- intensity, color 126
- interpolation for image objects 128
- intn data type 78
- intptr_t data type 71
- inverse discrete Fourier transform 302
- inverse function 380
- inverse square root function 106
- inverted indexes, with MapReduce 240
- involution, vector 268
- isequal function 118
- isfinite function 119
- isgreater function 118–119
- isgreaterequal function 118
- isinf function 119
- isless function 119
- islessequal function 119
- islessgreater function 119
- isnan function 119
- isnormal function 119–120

isnotequal function 118
 isordered function 119
 isunordered function 119

J

Java NIO buffers. *See* NIO buffers
 Java, enabling in MinGW 402
 JavaCL
 advantages of 201
 classes 202–203
 data transfer commands 209–210
 data transfer example
 code 209
 information parameters 204
 installing 202
 kernel creation 203–210
 license for 202
 overview of 202
 JavaCL class 204
 JavaCLProgram.java file 206
 JavaCLRoot.java file 209
 JOCL 197

K

kernel arguments
 address space modifiers 85, 181
 address space qualifiers 70, 88–90
 by value or by reference 70
 creation of 70
 for private memory 91–92
 image objects as 125, 129
 in C++ 176–183
 in local memory 91
 in local space 182–183
 in PyOpenCL 215–216
 JavaCL, setting 206–208
 local memory allocation 91, 182
 memory objects and 44
 memory objects as 181–182
 read/write settings 178
 requirement for 69
 samplers as 128
 setting 44–45
 Kernel class
 constructor 214
 functions in 211
 Kernel class (Aparapi) 198–200

kernel code
 buffering 30–31
 for matrix-vector multiplication 13
 vs. host code 15
 See also source code
 kernel distribution 7
 card game analogy for 8–10
 command queues and 10
 for matrix-vector multiplication 10–13
 partitioning 62
 to multiple devices 10
 Kernel objects
 constructor 175
 example code 176
 Kernel objects (PyOpenCL) 218
 kernel_init function 251
 kernel_merge function 251
 kernel_stage functions 251
 kernels
 .cl suffix for 69
 address space qualifiers 70
 command queues and 7
 creating 36–37, 170, 175–176
 creating, with JavaCL 203–210
 creating, with PyOpenCL 212–219
 creation of 7
 deallocating 38
 declaration for 69
 defined 7
 deployability of 36
 deploying to command queue 39
 enqueueing execution commands 40–41
 example of 69–70
 execution mode, setting in Aparapi 199
 execution time, profiling 157–158
 for matrix multiplication 264
 for reduction 227–228
 high-performance tips 231–233
 image processing by 126
 index space for 99
 information parameters 37
 information, retrieving 37–38
 local memory usage, returning 223
 loop execution 63
 memory allocation 37

multiple, processing 230–231
 performance,
 improving 231–233
 private memory
 initialization 91
 private memory usage,
 returning 223
 profiling, in C++ 193
 resource usage,
 determining 223
 returning information on 224
 returning void 69
 rounding functions, in Aparapi 198
 searching by name 38
 structure of 69
 syntax for 69
 testing 225
 vector width, multiple versions for 80
 vs. programs 30
 vs. work-items 63
 See also cl_kernel data structure; data structures
 kernels (image filtering) 342
 for embossing images 344
 for Gaussian blur 343
 for sharpening images 344
 Khronos Group 5

L

large numbers, multiplying and adding 112
 ldexp function 106
 leading zeros, counting in vector components 119
 least significant bits (LSB) in mask vectors 114–115
 least-significant digits (LSD) radix sort 254
 left-shift operator (112–113
 length function 120–121
 letter-indexing vector components 82–83
 libpng library 134
 libraries
 import library 358
 libamdocl64.so 358
 libcuda.so 358
 libOpenCL.so 352, 358, 360–362
 linker 352
 Mach-O 352
 oclUtils 358

libraries (*continued*)

- OpenCL.dll 352, 362
- OpenCL.lib 356, 358
- SDKUtil 356
- library files in makefile builds 411
- linear system solutions. *See* Conjugate Gradient method
- Linux
 - AMD device 358
 - AMD driver 358
 - bashrc 360
 - ICD names 352
 - lspci 352
 - Nvidia device 358
 - Nvidia driver 359
 - X server 359
- listPlatforms method (JavaCL class) 204
- little-endianness. *See* endianness
- local memory
 - avoiding conflicts 232
 - data transfer to/from global memory 101, 165
 - defined 88
 - in reduction algorithm 228
 - kernel argument configuration in 91
 - kernel arguments in 182–183
 - kernel usage of, returning 223
 - PyOpenCL allocation to 217
 - qualifier for 89
 - speed of 88
- local size of work-items 222
- local_work_size argument (clEnqueueNDRangeKernel function) 66
- LocalMemory class 217
- LocalMemory objects (PyOpenCL) 217
- localSize argument (enqueueNDRange function) 208
- LocalSpaceArg objects 182
- log/log1p function 106
- log2/log10 function 106
- logarithmic functions 106
- logb function 106
- long data type 71
- longn data type 78
- loops
 - in bitonic sort 253
 - index spaces 64
 - kernel execution and 63
 - unrolling 231

M

- Mac OS
 - chipset model 352
 - framework 352, 361–362
- macro declarations 404, 406–407
- macros, setting for private variables 232
- MAD (multiply-and-add) operations. *See* multiply-and-add (MAD) operations
- mad function 103–104
- mad_hi function 111–112
- mad_sat function 111–112
- mad_test.cl file 112
- mad24 function 111–112
- magnitude 307
- main function 378
- makefiles 362, 404–409
 - all rules 408
 - automatic variables 407
 - comments in 406
 - defined 404
 - deleting targets 408
 - dependencies 405
 - documentation for 409
 - environment variables in 410
 - example of 404, 409
 - executing shell statements 408
 - library files 411
 - macros in 406–407
 - make command, for multiple target files 408
 - phony targets 408–409
 - removing conflicts 408
 - rule processing 407
 - rules 405–406
 - rules, configuring builds with 407
 - shell statements in 406
 - statements in 404
 - structure of 404–407
 - tab characters in 406
- map method (CLBuffer class) 209
- map_copy application 60
- map_copy.cpp file 187
- map_flags argument (clEnqueueMapXX functions) 59
- mapping memory objects 58–59
 - in C++ 187–189
 - in PyOpenCL 219
- mapping textures 394–395
- MapReduce
 - combination of key-pairs 239
 - distributed grep task 242–244
 - embarrassingly parallel tasks 240, 242
 - history of 238
 - memory allocation 241
 - OpenCL device model and 240–242
 - processing independence of 239
 - processing model 238–239
 - word-count example 239
- marker commands 148–149
- mask vectors
 - component data type 114–115
 - defined 114
 - least significant bits (LSB) in 114–115
 - radix sort and 254–256
 - relational operators, creating with 118
 - select functions 116–118
 - shuffle functions 114–116
 - with two inputs 115
- math functions. *See* arithmetic functions
- math.h file 94
- matrices
 - as vectors 267–269
 - blocks, transposition of 260
 - column-major order 263, 379
 - defined 259
 - dense 260
 - diagonal 260
 - diagonals in 259
 - dot products and 341
 - elements in 259
 - factoring 258, 269
 - See also* QR decomposition
 - for Gaussian blur 343
 - functions for 379
 - identity 260, 285
 - in image filtering 342
 - initializing 379
 - notation for 259
 - positive-definite 285–286
 - Q matrix, finding 272
 - R matrix, finding 270–272
 - read/writing 57
 - row-major order 263
 - sparse. *See* sparse matrices

- matrices (*continued*)
 - square 259, 380
 - symmetric 260, 282
 - transformation 381
 - transposition, allocating mem-
ory for 261
 - transposition, example
code 260
 - upper triangular 270
 - vectors in 259
 - See also* QR decomposition;
shaders
 - Matrix Market files
 - banner 282
 - double values in 284
 - I/O functions for 282
 - indices, starting at 1 284
 - repository for 281
 - structure of 281
 - suffix for 281
 - symmetric matrices in 282
 - See also* sparse matrices
 - matrix multiplication 262–265
 - associative nature of 263
 - dot products and
262–263, 267
 - in OpenCL 263–265
 - in shader functions 380
 - kernel for 264
 - outer products 267–268
 - theory of 262–263
 - matrix transposition 259–262
 - matrix_mult.cl file 264
 - matrix-vector multiplication
10–13
 - data partitioning in 65
 - discrete Fourier transform
and 301
 - matrixCompMult function
379–380
 - matvec_mult function 13
 - matvec.c file 10–13
 - matvec.cl file 13
 - max function 105, 113
 - maxmag function 105
 - mean filters 342–343
 - memory address spaces. *See*
address spaces
 - memory alignment 90
 - memory allocation
 - for buffer objects 45–47
 - for build log 35
 - for cl_context structure 28
 - for kernels 37, 91
 - for MapReduce 241
 - for matrix transposition 261
 - for sub-buffer objects 54
 - for vertex buffer objects
(VBOs) 336
 - memory banks
 - avoiding conflicts 232
 - in reduction algorithm 227
 - memory objects
 - checking size/location of 52
 - data transfer between 59–62
 - deallocating 125
 - functions for mapping/
unmapping 59
 - hierarchy for 177
 - in PyOpenCL 216–217
 - information parameters 52
 - information, retrieving 181
 - kernel arguments and 44, 181
 - mapping 58–59
 - mapping, in C++ 187–189
 - properties, setting 45
 - read/write permissions 45–46
 - read/writing 54–58
 - read/writing, in C++ 185–187
 - See also* buffer objects
 - memory. *See* global memory;
host memory
 - MemoryObject class 211, 216
 - Method of Steepest Descent
algorithm for 288
 - example code 288–289
 - iterative nature of 285
 - residual vector,
determining 287
 - theory of 286–287
 - methods, JavaCL 202–203
 - min function 105, 113
 - MinGW
 - 64-bit builds with 402
 - compiling in 402
 - confirming installation 400
 - directory for installation 399
 - downloading 399
 - executables 401
 - executables, verifying 402
 - GNU compiler 402–404
 - GNU public license for 399
 - graphical installer 399
 - Hello World! application 402
 - installation options 399
 - installing in Windows
398–402
 - installing new tools 401
 - library definition
 - placement 411
 - mingw-get command 401
 - msys.bat file 401
 - PATH variable, setting 400
 - Perl scripts, enabling 401
 - repository catalogue,
downloading 399
 - root directory 401
 - subdirectories in 400
 - Unix commands in 401
 - See also* gcc compiler
- mingw32-gcc 402–404
 - minmag function 105
 - miplevel parameter (clCreate-
FromTextureXX
function) 327
 - mipmaps 327, 390, 392–393
 - See also* textures
 - mix function 105
 - mm_is_real function 282
 - mm_is_sparse function 282
 - mm_is_symmetric function 282
 - mm_read_mtx_crd_size
function 282
 - mobile devices, OpenCL
on 412–414
 - mod_round.cl file 104
 - modf function 109
 - modulo operator, avoiding
227, 232
 - modulus, returning 103–104
 - momentum, calculating 159
 - monotonic sets 245
 - See also* bitonic sort
 - most significant bits (MSB)
in select functions 116
 - testing 97
 - MSB (most significant bits). *See*
most significant bits (MSB)
 - msg argument (kernel
functions) 70
 - msys.bat file 401
 - .mtx files 285
 - See also* Matrix Market files
 - mul_hi function 111
 - mul24 function 111–112
 - multiplication functions
111–112
 - multiplication operator 96
 - multiply-and-add (MAD)
operations 175, 233
 - multiplying matrices. *See* matrix
multiplication
 - mutex.cl file 163

mutexes
 check-unlock procedure 163
 defined 163
 OpenGL synchronization
 with 328
 mutual exclusions. *See* mutexes

N

naming convention
 for data structures 13
 for extensions 14
 nan function 109
 nanoseconds 154
 National Institute for Standards
 and Technology (NIST)
 site 281
 ndarray objects 216
 nearest-neighbor
 interpolation 128, 135–136
 nextafter function 109
 NIO buffers 207
 data transfer methods 209
 identifying class of 207
 noise removal from images
 342–343
 normal numbers in float data
 type 74
 normal vectors 121
 normalize function 120–121
 normalized colors 126
 normalized coordinates
 126, 129
 normalized_coords parameter
 (clCreateSampler
 function) 125
 Not a Number (NaN) values 74
 notification events. *See* host noti-
 fication events
 num_entries parameter (clGet-
 PlatformIDs function) 19
 num_platforms parameter
 (clGetPlatformIDs
 function) 19
 number-indexing vector
 components 81–82
 numerical processing, in
 OpenGL 412–413
 numerical reduction. *See*
 reduction
 Nvidia
 development file variable 410
 download site 356
 GPU Computing SDK
 351, 356–357

license agreement 360
 run-time compiler 31
 SDK (software development
 kit) 14, 351, 362

O

object-oriented programming 167
See also C++
 offset argument (clEnqueueXX
 functions) 56
 offset argument (vloadn
 function) 102
 offset argument (vstoren
 function) 102
 op_test.cl file 96
 open function 214
 Open Graphics Language
 (OpenGL). *See* OpenGL
 OpenGL
 advantages of 5–8
 card game analogy for 8–10
 embedded. *See* embedded
 OpenGL
 history of 4–5
 installing. *See* installing
 OpenGL
 library name 411
 nonproprietary nature of 4
 on mobile devices 412–414
 portability of 6
 specifications 5
 OpenGL 1.1 standard 13
 full or embedded,
 checking 20
 OpenGL Working Group 5
 OpenGL
 advantages and
 disadvantages 322
 animation 335–338
 buffers, initializing 387–388
 buffers, resetting 390
 combining with GLUT
 387–390
 creating OpenGL context
 for 323–325
 data objects, creating
 332–333
 data structures 325
 data structures, as OpenGL
 data structures 322
 data types 369
 data types, returning 329
 development on host
 367–377
 drawing modes 375–376
 ensuring routine
 completion 328
 host application function 367
 initializing 345
 installing 366–367
 Linux installation 366–367
 Mac OS installation 367
 object information,
 returning 329–330
 OpenCL data structure
 initialization 345–346
 OpenCL initialization
 331–332
 OpenCL
 interoperability 322–323
 OpenCL interoperability,
 example of 331–334
 OpenCL kernel
 execution 333–334,
 337–338, 347
 OpenCL
 synchronization 328–329
 operation of 365
 overview of 364
 primitives 375–376
 rendering graphics 334
 rendering the model 389–390
 static rendering example 364
 viewport, setting 389
 Visual Studio header files 366
 vs. OpenGL applications 365
 window, creating with
 GLUT 383–386
 window, redrawing 348–349
 Windows installation 366
See also GL Utility Toolkit
 (GLUT); shaders
 OpenGL contexts 27
 creating 331–332
 device ID, returning 330
 displaying 27
 parameters for 27
 returning information on 330
 OpenGL Shading Language
 (GLSL) 377
 data types 378
 operators 379
 suffixes 378
See also shaders
 OpenGL textures. *See* textures
 opengl.dll file 366
 operations
 atomic 160–163
 for data transfer 101–102
 on data types 18

- operators
 - arithmetic 96
 - assignment 96
 - for shader functions 379
 - list of 96
 - on vectors and scalars 96
 - relational 97
 - relational, creating mask vectors with 118
 - See also* specific operators
 - optimizations, disabling 32
 - origin argument (clEnqueueXX functions) 56
 - orthogonalization, vector 290–291
 - oscillation 297
 - outer products
 - computing 268
 - vs. dot products 267
 - outerProduct function 379–380
 - overflow, arithmetic. *See* arithmetic overflow
- P**
-
- parallel processing 240, 242
 - parallel programming 7, 14
 - See also* MapReduce
 - parameters
 - cl_command_queue properties (clCreateCommandQueue) 40
 - cl_context_properties (clCreateContext function) 26
 - cl_context_reference_count (clGetContextInfo function) 28
 - cl_program_build_info (clGetProgramBuildInfo function) 34
 - cl_program_info (clGetProgramInfo function) 33
 - error (clCreateContext function) 28
 - for cl_device_info type 23
 - for clGetPlatformInfo function 20
 - num_entries (clGetPlatformIDs function) 19
 - num_platforms (clGetPlatformIDs function) 19
 - src_num (clCreateProgramWithSource function) 30
 - user_data (clCreateContext function) 28
 - void (clCreateContext function) 26
 - partitioning data. *See* data partitioning
 - Perl, enabling in MinGW 401
 - Petrov-Galerkin method 279
 - pinned memory 46
 - pitch of image objects 50–51
 - pixel buffer objects (PBOs) 346–347
 - pixel data memory objects. *See* image objects
 - Platform class 171–172, 211
 - platform extensions 14
 - testing 20
 - vs. device extensions 24
 - platform layer 14
 - Platform objects
 - accessing Device objects with 171
 - example code 172
 - Platform objects (PyOpenCL) 212
 - platform structures
 - creating 18
 - creating for every platform 19
 - setting maximum number of 19
 - Platform::get function 171
 - Platform::getInfo function 171
 - platforms
 - defined 10
 - devices associated with 23
 - extension testing 22
 - full or embedded standard, returning 20
 - information parameters 19
 - information, retrieving 19–20
 - initializing 18
 - JavaCL and 204
 - multiple contexts from 25–26
 - names, returning 20
 - vendors, returning 20
 - See also* cl_platform_id structure; data structures
 - PNG images, processing 134
 - polar coordinates, converting to rectilinear 106
 - portability of OpenCL 6
 - Portable Network Graphics (PNG) images, processing 134
 - positive-definite matrices 285–286
 - pow function 106
 - pown function 106
 - powr function 106
 - preferred vector widths 79–80
 - prefetch function 165–166
 - primitive data types 17–18
 - kernel arguments and 92
 - transferring with compiler defines 232
 - primitives 375
 - printStackTrace method (CLBuildException class) 206
 - private memory
 - accessing data as vectors 92
 - defined 88
 - initializing 91
 - kernel arguments for 91–92
 - kernel usage of, returning 223
 - PyOpenCL allocation to 217
 - qualifier for 90
 - size of 91
 - speed improvements with 232
 - usage considerations 88
 - private variables, reusing 232
 - processing image objects 133–135
 - profile_items.c file 157
 - profile_read.c file 155
 - profile.cpp file 193
 - profiling
 - commands 153–155
 - data partitioning 157–158
 - data transfer 155–156
 - kernels, in C++ 193
 - profiling events 140
 - configuration 153–155
 - example code 154
 - in C++ 192–193
 - Program class
 - constructors 213
 - functions in 211
 - Program objects
 - atomic MAD (multiply-and-add) operations, allowing 175
 - building 174–175
 - constructors 174
 - creating 174
 - creating, from text files 174

program objects (OpenGL)
 activating 373, 375
 attaching shaders to 374
 creating 374
 creating executable from 373
 deallocating 373

Program objects
 (PyOpenCL) 214

programs
 binary buffer array,
 returning 33
 build information
 parameters 34
 compiler options 32
 context, retrieving 33
 creating 30–31
 creating, from text files 30–31
 creating, in JavaCL 205
 creating, in PyOpenCL 214
 devices targeted by,
 returning 33
 information, retrieving 33–35
 kernel creation from 175–176
 multiple source files 35–36
 reference count, returning 33
 source code, returning as single string 33–34
 vs. kernels 30
See also `cl_program` structure;
 data structures

projections, vector 265–266

`ptrdiff_t` data type 71

PyOpenCL
 classes in 211
 compiling 210
 Context objects 212–213
 Device objects 212
 installing 210–211
 kernel arguments,
 setting 215–216
 kernel creation with 212–219
 kernel execution 218–219
 local memory allocation 217
 memory objects in 216–217
 naming conventions 212
 parameters given in
 tuples 218
 Platform objects 212
 script structure 211
 web site for 210

`pyopencl` class 211

Pythagorean Theorem 120

Python 210
See also PyOpenCL

Q

QR decomposition
 defined 270
 finding reflection vectors 270
 kernel for 273–276
 OpenCL
 implementation 273–276
 Q matrix, finding 272
 R matrix, finding 270–272
See also Householder
 transformation

`qr.cl` file 273–274

R

radix sort 254–256
 Back 40 implementation 256
 LSD (least-significant
 digits) 254
 number of passes by 254
 overview of 237, 254
`ushortn` vector and 255
 vector implementation
 of 254–256
See also bitonic sort

`radix_sort8.cl` file 255

rasterization 382

`rdft.cl` file 305

read function 214

read methods (`CLBuffer`
 class) 209

read_image functions 130–131

`read_mm.c` file 283

read-only memory. *See* constant
 memory

read/write data transfer 54–58
 enqueueing commands
 for 185–187
 in C++ 185–187
 profiling 155–156
 synchronizing 160
See also data transfer

readFile method (JavaCL) 206

real-valued sequences 304

rectangular data in memory
 objects 56–57, 186

rectilinear coordinates, convert-
 ing into polar 106

reducing images. *See* scaling
 images

reduction
 algorithm for 226–227
 barrier function and 228
 computational steps for 231
 defined 221
 kernel for 227–228
 local memory usage 228
 memory banks and 227
 speed, improving 228–229
 storage method for 227
 with vectors 228–229
 work-group 230–231

reduction of key-pairs. *See*
 MapReduce

`reduction_complete.cl` file 230

`reduction_scalar` function 226

`reduction_scalar.cl` file 227

`reduction_vector.cl` file 228

`reduction.cl` file 226

reference counts
 changing 29
 checking 29–30
 for `cl_context` structures 29
 for contexts 28
 for samplers, increasing/
 decreasing 128
 functions for 29
 returning 33, 151
 tracking 28

reflection, vector 266–267
 finding values for 268
 in OpenCL 269
See also Householder
 transformation

region argument (`clEnqueueXX`
 functions) 56

relational operators 97

remainder function 103–104

remainder of quotient,
 returning 103

`remquo` function 103

renderbuffers
 creating image objects
 from 327
 defined 325

rendering 3-D graphics. *See*
 OpenGL

rendering, OpenGL
 334, 389–390

`reshape` function 386

residual vector,
 determining 287

resizing images. *See* enlarging
 images

resolution, timing 154

retain counts. *See* reference
 counts

RGB image format 50

`rhadd` function 110–111

- rint function 103–104
 - Ritz-Galerkin method 279
 - rootn function 106
 - rotate function 112–113
 - round function 103
 - rounding away from zero 105
 - vs. rint function 104
 - rounding
 - doubles 75
 - floats 74
 - halfs 75
 - rounding functions 103–104
 - rounding functions (Aparapi) 198
 - row pitch in image objects 180
 - row vectors 267
 - row_pitch argument (clCreateImage functions) 50
 - rsqrt function 106
 - run method (Kernel class) 199
 - run_kernel.py file 219
 - run-time compilers 31
 - runtime 14
- S**
-
- sampler_t structures 129
 - samplers
 - as kernel arguments 128
 - as uniforms 395
 - bilinear interpolation 136–137
 - creating 125
 - defined 125
 - image objects and 124
 - interpolation type, setting 128
 - nearest-neighbor interpolation 135–136
 - properties, configuring 129–130
 - reference count, reducing 128
 - textures and 395
 - transferring to kernel 129
 - scalar arrays, loading vectors from 101–102
 - scalar data types 70–71
 - operations on 96
 - PyOpenCL and 217
 - reduction with 227–228
 - sorting 248
 - vector width, determining 79
 - scaling images 135
 - SD method . *See* Method of Steepest Descent
 - SDKs (software development kits)
 - device checking 351
 - Linux 352
 - Mac OS 352
 - Windows 352
 - header file 352, 356, 358, 362
 - vendors 14
 - searching text with
 - MapReduce 242–244
 - segmentation faults 144
 - select functions 116–118
 - example code 118
 - most significant bits (MSB) 116
 - relational operators and 118
 - select_test.cl file 118
 - semaphores. *See* mutexes
 - sequences 298
 - real-valued 304
 - shifting 307
 - setArg function
 - cl::LocalSpaceArg object, calling with 182
 - example code 182
 - signature for 215
 - setArg function (Kernel class)
 - arguments for 177
 - signature of 176
 - setArg method (JavaCL) 208
 - setArt method (CLKernel class) 206
 - setCallback function (Event class) 190
 - setExecutionMode method (Kernel class) 199
 - setKernelArg function 129
 - setSize method (Kernel class) 200
 - setStatus function (UserEvent class) 192
 - sgqrt function 106
 - shaders
 - attaching to program objects 373–374
 - build log, returning 372, 374
 - compiling 372–373, 389
 - components of 378
 - creating 372–373
 - data types 378
 - defined 364
 - deploying 388
 - deployment process 372
 - development 377–383
 - functions governing 372
 - information, retrieving 372
 - matrix functions 379
 - matrix initialization 379
 - matrix multiplication 380
 - operators for 379
 - overview of 364
 - singular nature of 373
 - source code, setting 372–373
 - structure of 378
 - suffixes 378
 - vs. kernels 322, 377, 379
 - See also* fragment shaders; vertex shaders
 - share groups. *See* contexts
 - sharpening images 344
 - shell statements, in a makefile 404
 - short data type 71
 - shortn data type 78
 - shuffle functions 114–116, 248
 - rearranging vector elements with 249
 - signatures for 114
 - sign function 109
 - signbit function 119
 - signed zero values, preventing 32
 - simple_image.cl file 134
 - sincos function 106
 - sine functions. *See* trigonometric functions
 - single-precision constants 32
 - sinusoids 299–301
 - size parameter (cl::Buffer class) 178
 - size_t data type 71
 - slice pitch in image objects 180
 - slice_pitch argument (clCreateImage functions) 50
 - smoothstep function 105
 - software development kits (SDKs). *See* SDKs (software development kits)
 - sorting algorithms. *See* bitonic sort
 - source code
 - for matrix-vector multiplication 10–13
 - in multiple files 35–36
 - single string, returning 33–34
 - See also* kernel code
 - Sources data type 174

- sparse matrices 260
 - Conjugate Gradient method 289–293
 - defined 280
 - importance of 278
 - Method of Steepest Descent 285–289
 - nonzero elements, reading 282–283
 - representing transmission tower 280
 - size of 281
 - size, returning 282
 - storage of 280–285
 - vs. dense matrices 280
 - See also* Matrix Market files
 - spatial filtering. *See* image filtering
 - specifications 5
 - sphere.c file 335–336
 - spherical coordinates 337–338
 - spinlocks 163
 - square matrices
 - converting to upper triangular matrices 270–276
 - defined 259
 - functions for 380
 - inverse, returning 380
 - src_num parameter (clCreateProgramWithSource function) 30
 - stalling commands. *See* command synchronization
 - static rendering 364
 - See also* OpenGL
 - std::string class 169, 173
 - std::vector class 169
 - steep_desc.c file 288
 - step function 105
 - stretching, definition of 308
 - string literals 89
 - string searching, with MapReduce 242–244
 - string_class macro 168
 - string_search function 244
 - string_search.cl file 243
 - strings, C++ classes for 169
 - sub_sat function 110
 - sub-buffer objects 47–48
 - creating, in C++ 178
 - memory allocation 54
 - size, defining 47
 - subtraction functions 110
 - subtraction operator 96
 - in vertex shaders 394
 - initializing 391
 - interpolation method, setting 392
 - mapping 394–395
 - minimizing 390, 392–393
 - mipmaps 327, 390, 392–393
 - overview of 390
 - pixel memory storage, setting 392
 - resizing 390
 - returning information on 329
 - zooming in/out from 392
- tgamma/lgamma function 106
- three_squares.c file 374, 387–389
- three_squares.frag file 382
- three_squares.vert 381
- Tianhe-1A supercomputer 3
- tick variable (display function) 337
- time-domain sequences, converting. *See* discrete Fourier transform
- time-domain signals 296
- timing. *See* profiling
- transferring data. *See* data transfer
- transformation matrices 381
- transformations 381
- transmission tower, represented by sparse matrix 280
- transpose function 379–380
- transpose.cl file 260
- transposition, matrix 260–262
 - data dimensionality of 261
 - example code 260
 - memory allocation 261
- trigonometric functions 106
- troubleshooting build errors 34
- trunc function 103
- try blocks 170

T

U

unrolling loops 231
 upper triangular matrices
 270–276
 upsample function 113
 user events
 creating 146
 defined 145
 example code 147
 in C++ 192
 on multiple devices 146
 status, setting 192
 status, updating 146
 vs. command events 146
 wait lists and 146–148, 192
See also command events
 user_data parameter (clCreate-
 Context function) 28
 user_event.c file 147
 user_event.cpp file 192
 UserEvent class constructor 192
 ushorthn data type 78, 255

V

variable, reusing private 232
 vec_reflect.cl file 269
 vec_step function 119
 vector data types
 for Program objects 174
 list of 78
 preferred vector width,
 determining 79–80
 valid values of n 78
 vector processing 6–7, 15
 vector processors 7
 vector reflection 266–267
 finding values for 268
 in OpenCL 269
See also Householder
 transformation
 vector_bytes.cl file 84
 vector_class macro 168
 vector-matrix multiplication. *See*
 matrix-vector multiplication
 vectors
 absolute values of
 comparisons 249
 as matrices 267–269
 C++ classes for 169
 column 267
 comparison functions
 105, 118–120, 242
 conjugate 290
 cross products,
 returning 120–122
 defined 6
 directions, determining 265
 dot products, returning
 120–121
 endianness and 84–85
 Euclidean distance,
 returning 120
 geometric coordinates 120
 geometric functions 120–122
 Gram-Schmidt Method of
 orthogonalization 290–291
 hi/lo/even/odd component
 selection 83–84
 in shader code 378
 initializing 80
 involution 268
 length, returning 120
 letter-indexing
 components 82–83
 linear dependence, and
 orthogonalization 290
 loading from scalar
 arrays 101–102
 mathematical 259
 maximum number of
 components 81
 memory access 84–85
 minimum/maximum values,
 setting 105
 normal 121
 normalizing 121
 number-indexing
 components 81–82
 numerical reduction
 with 228–229
 operators for 95–97
 orthogonalization of 290–291
 preferred widths,
 determining 79–80
 projections for 265–266
 projections, calculating 266
 reading from image
 objects 130–132
 row 267
 sorting elements between 249
 sorting elements in 248
 sorting elements of 254–256
 standardized processing in
 OpenCL 6–7
 storing to scalar arrays
 102–103
 subvectors, creating 81–82
 test functions 118–120
 true/false returns 97
 vs. arrays 77–78
See also components, vector;
 mask vectors
 vendor-specific library. *See*
 installable client driver
 vendors, returning 20
 vertex array objects (VAOs)
 creating 332–333, 371, 387
 defined 370
 functions for 371
 vertex buffer objects (VBOs)
 associating with vertex
 data 369
 binding 369
 buffer creation from 326
 creating 332–333, 369, 387
 deallocating memory 370
 defined 325, 368
 functions for 368
 initializing 333
 memory allocation 336
 unbinding 370
 vertex data, specifying
 335–336
 vertex shaders
 attributes for 381
 compiling 331, 345
 creating 373
 example code 381
 matrix-vector multiplication
 and 381
 overview of 380
 texture mapping 394–395
 vertex positions, setting 381
 vertices
 as float4 vectors 338
 attributes 370–372
 defined 364
 defining 368
 setting number of 377
 shapes formed from. *See*
 primitives
 spherical coordinates
 337–338
 vertex buffer objects
 (VBOs) 368–370
 viewport (OpenGL), setting 389
 Visual Studio OpenGL header
 files 366
 vloadn function 101–102
 void parameter (clCreateCon-
 text function) 26
 volume, calculating 159
 vstoren function 102–103

W

-
- wait commands 149–150
 - wait function (Event class) 193
 - wait lists
 - cl_event structures and 145
 - command events and 145, 191
 - configuring 145
 - defined 145
 - stalling multiple commands with 149–150
 - user events and 146–148, 192
 - wait_group_events function 165–166
 - waitForEvents function (Event class) 194
 - warnings, responding/suppressing 32
 - web link relationships, with MapReduce 240
 - web page access statistics, with MapReduce 240
 - wg_test.c file 224
 - wglGetCurrentContext function 324
 - wglGetCurrentDC function 324
 - when statements 97
 - while loops 231
 - Windows
 - Direct3D 356, 358
 - ICD names 352
 - MinGW 354
 - registry 352
 - registry key 352
 - Visual Studio 354, 356, 358
 - Windows environment variables 400
 - Wolfenstein 3D 340
 - work_dims argument (clEnqueueNDRangeKernel function) 64
 - work-groups
 - advantages of 65
 - assigning data to, in bitonic sort 250
 - compute units for 66
 - confirming successful processing 230
 - functions, list of 99
 - global functions vs. local functions 99
 - index space example 184
 - Java methods for returning information 200–201
 - MapReduce and 241–242
 - maximum size 223–224
 - memory storage 88
 - overview of 222
 - parallel execution limit on 223
 - reduction algorithm 226
 - relationship to work-items 87
 - returning information on 223–224
 - size, specifying 66
 - synchronization, working around 230–231
 - See also* work-items
 - work-item synchronization
 - overview of 159
 - with barriers and fences 159–160
 - with mutexes 163–164
 - work-groups and 160, 264
 - work-items
 - barriers for 159
 - comparison to for loop 99
 - compute unit constraint on 164
 - configuring, in Aparapi 200–201
 - defined 63
 - dimensionality, configuring 64
 - dimensions for 98–99
 - functions, list of 98
 - global functions vs. local functions 99
 - global ID 63, 98
 - global ID, specifying 64
 - global size 222
 - ID configuration, example code for 100–101
 - IDs, printing as float 100
 - index space example 184
 - Java methods for returning information 200–201
 - local ID 66
 - local memory, speed of 91
 - local size 222
 - maximum dimensions, finding 64
 - memory access, speed of 88
 - memory storage 88
 - mutexes for 163
 - numerical reduction of. *See* reduction
 - offset, returning 98–99
 - relationship to work-groups 87
 - size, returning 98, 157
 - size, setting 64, 208
 - spinlocks for 163
 - vectors, number to process 158
 - vs. kernels 63
 - See also* work-groups
 - write method (CLBuffer class) 209
 - write_image functions 132
 - writing to images 132–133

Z

-
- zeros, leading in vector components 119