Index

Symbols

\ (backslash)
  disallowed in filenames, 23
  for root directory, 16, 57

A

absolute pathnames, file objects, 177
access
  Direct Memory Access (DMA), 256
  file object, 177
  mount points and, 28
  mounted logical volumes, 24
  violations, memory and, 69
  virtual address space, 196-201
AcquireFileForNtCreateSection( ), 547-549
AcquireForCcFlush( ), 551
AcquireForLazyWrite( ), 289, 354
  example of, 553-554
AcquireForModWrite( ), 549-551
AcquireForReadAhead( ), 289, 352
administration
  event logging, 86-93
  status reports to, 66
aliasing, 185
allocating, 39–41
  access violations and, 69
  to Cache Manager, 248-249
  device objects, 140-141
  dispatcher object storage, 98
  ERSOURCE structures storage, 110
  for event log entries, 91-92
  event object storage, 101
  Executive spin lock storage, 96
  file objects, 176
  file streams, 263, 346
    changing file stream size, 487
  FSD design and, 364
  IRPs, 144-146, 150, 636-639
  lookaside lists and, 45
  MEM_ types for, 208
  multiple stack locations, 155
  termination handlers and, 84
  virtual address spaces, 206-209
  VPB structure, 369
  zones, 42
ANSI strings, converting, 47
APCs (asynchronous procedure calls), 10, 107
  APC_LEVEL, 10
  postprocessing IRPs, 150, 151, 154
APIs (application programming interfaces), 5
  as event log viewer, 87, 90
  MPR module, 61
arbitrary threads, 131-133
associated IRPs, 147, 647-650
asynchronous
  file object operations, 177
  I/O, 124, 458, 464-476
  IRP handling, 149-150
  paging I/O requests, 166

asynchronous *(continued)*
  procedure calls (see APCs)
  versus synchronous, 180-183
attach operation, filter driver, 622-634
attaching threads to processes, 200
attributes
  page frame, 202
  VAD structure, 205-206
  volume, modifying, 560-561

B

bad pages, 203
base address
  allocating memory at, 207-208
  determining, 52-54
batch oplocks, 574, 579-583
BCBs (buffer control blocks), 266-267, 316-318
block caching (see caching)
blue screen of death (BSOD), 55
boot file system driver, 191
boot loader startup routine, 186-187
boot sequences, 185-193
BootRecord structure, 186
breaking oplocks, 577-583
breakpoints, 54
BSOD (blue screen of death), 55
buffers
  buffer address, 134
  buffer cache, 246
  buffered I/O requests, 184, 533-534
  control blocks (BCBs), 266-267, 316-318
  data buffer, 152
  flush file buffers dispatch routines, 554-556
  multiple processes, same files, 215
  TLB (Translation Lookaside Buffer), 211, 213
  user-space buffer pointers, 183-185
bug, specifying and forwarding IRPs, 655-656
bugcheck calls, 55-56
BUILD.EXE utility, 736-740
byte offset, setting, 486
byte-range locks, 386-387, 536
  dispatch routines for, 562-571

C

caching, 235, 243
  Cache Manager, 18-19, 243-245
    allocating to, 248-249
    callback routines for, 289
    cleanup requests and, 328-332
    clients of, 258-260
    close requests, 332-333
    I/O Manager and, 348
    initializing, 190, 345
    interfaces of, 255-258, 271-293
    lazy writer component, 254
    logging and, 341
    VCB structure and, 373-374
    VMM interactions -with, 344-348
    when to read-ahead, 349-351
  CACHE_MANAGER_CALLBACKS structure, 552
  cache maps, 266
  data structures for, 260-267, 271-272
  DNLC implementation, 387
  during read/write operations, 248-254
  file size and, 267-269
  file streams, 245
  flushing the cache, 325-328
  initiating, 287-293
  names (MUP module), 64
  page coloring and, 202
  prerequisites for, 277-287
  private map structure, 266, 271, 289
  read-ahead (see reading, read-ahead functionality)
  resource acquisition, 274-276
  shared cache map structure, 266, 272
  system cache, 256
  termination of, 328-333
  TLB (Translation Lookaside Buffer), 211, 213
  VACB structure, 271-272
  virtual block caching, 246-248
  (see also memory)
call frames, 14
  call frame-based exception handlers, 72
  unwinding, 83-86
callback routines for Cache Manager, 289
cancelling
  IRPs, 150
  timer objects, 105

capitalization, converting, 48
CCB structure, 369, 384-385
CcCanIWrite( ), 300-303
CcCopyRead( ), 250, 293-297
CcCopyWrite( ), 254, 297-300
CcDeferWrite( ), 303-304
CcFastCopyRead( ), 293-297
CcFastCopyWrite( ), 297-300
CcFlushCache( ), 326-327, 332, 462, 551
CcGetDirtyPages( ), 342-343
CcGetFileObjectFromBcb( ), 318-319
CcGetFileObjectFromSectionPtrs( ), 339-340
CcGetLsnForFileObject( ), 344
CcInitializeCacheMap( ), 287-289, 345, 423, 436
CcIsThereDirtyData( ), 343-344
CcMapData( ), 306-308, 374
CcMdlRead( ), 257, 319-321
CcMdlReadComplete( ), 257, 321
CcMdlWriteComplete( ), 257, 323-324
CcPinMappedData( ), 308-310
CcPinRead( ), 310-311
  CcPreparePinWrite( ) versus, 314
CcPrepareMdlWrite( ), 257, 321-323
CcPreparePinWrite( ), 313-315
CcPurgeCacheSection( ), 335-337
CcReadAhead( ), 350
CcRepinBcbO, 316-317
CcScheduleReadAhead( ), 305-306, 350
CcSetAdditionalCacheAttributes( ), 341-342, 349, 353
CcSetDirtyPageThreshold( ), 337-338
CcSetDirtyPinnedData( ), 311-313
CcSetFileSizes( ), 268, 334-335
CcSetLogHandleForFile( ), 340-341
CcSetReadAheadGranularity( ), 304-305, 351
CcUninitializeCacheMap( ), 329-331, 528
CcUnpinDataC), 315-316
CcUnpinDataForThread( ), 315-316
CcUnpinRepinnedBcK), 317-318
CcZeroData( ), 338-339
characters
  character set for filenames, 23
  converting case of, 48
  UNC (Universal Naming Convention), 63
  Unicode, 46-49
checked build drivers, 738
CIFS protocol, 26
cleanup requests/routines, 328-332, 525-529
clients, Cache Manager, 258-260
client-side redirectors (see network redirectors)
CLOCK1_LEVEL, CLOCK2_LEVEL, 11
CloseHandle( ), 31
closing
  close requests/routines, 332-333, 529-531
  files, 31
  IRPs for, 165
  termination of caching, 328-333
CM, initializing, 190-191
code
  discarding after driver initialization, 38
  making pageable, 38
codes, event, 88
collided page fault, 231
comments, FSD design and, 363
committed memory, 206
Common Internet File System (CIFS), 26
CommonFCBHeader structure, 261
comparing Unicode strings, 47
completion routines, 160-161, 163-169, 650-661
  associated IRPs, 649-650
  invoking, 653-656
  IRQLs for, 170, 656-657
  specifying for IRPs, 651-653
compressed file streams, 484
COMPUTE_PAGES_SPANNED( ), 295
concatenating Unicode strings, 48
configuring
  CM (Configuration Manager), 190-191
  I/O subsystem components, 128
  WINDBG kernel, 745
connection-oriented vs. connectionless protocols, 26
consistency of data
  dispatch routines and, 425, 438
  distributed file systems and, 29
  I/O operations problems, 461-464
  network file systems and, 27
constants, Unicode strings as, 48
container objects, 398
  (see also directories)

CONTAINING_RECORD( ), 52-54
context control block (see CCB structure)
context, thread (see threads, thread context)
control information (see metadata)
control objects, 12
control requests (see system control requests)
controller objects, 123
converting
  ANSI and Unicode strings, 47
  string case, 48
copy interface, 256, 293-306
copy-on-write feature, 206
copying strings, 48
core, file system, 361
corruption of data, 65, 93
Count value, semaphores, 109
counting semaphores (see semaphore objects)
CPU privilege levels, 7-8
crash-dump files, analyzing, 746
crashing the system, 65
create/open dispatch routines, 397-424
  algorithm for, 406-407
  filter drivers and, 635-636
CreateFile( ), 30
critical region, 94
critical work requests, 352
customer code flag, 88

D

data
  caching (see caching)
  consistency of
    dispatch routines and, 425, 438
    distributed file systems and, 29
    I/O operations problems, 461-464
    network file systems and, 27
  corruption of, 65, 93
  encryption/decryption, 389
  flushing (see flushing)
  making pageable, 38
  pinning/unpinning, 257-258, 306-319
  reading (see reading)
  shared (see synchronization)
  writing (see writing)
data buffers, 152
data structures
  BootRecord structure, 186
  for caching, 260-267, 271-272
  determining base address, 52-54
  directory information, 505-507
  driver extension structure, 139
  ERESOURCE structure, 110-112
  for exceptions, 68
  file attribute modification, 486-489
  for file lock support, 567-568
  file object structure, 31
  file stream information, 481–486
  for file system drivers, 386-390
  of file system, 367-390
  interrupt spin locks for, 95-96
  of kernel, 12
  for linked lists, 49-52
  list of common, 135-180
  on-disk, 361, 367
  UNICODE_STRING, 46
  (see also objects)
DbgBreakPoint( ), 54, 56
DbgPrintO, 54
deadlock condition, 95
  create section requests and, 548
  fast mutex locks and, 106
  semaphore objects and, 108
deallocating
  IRPs, 163-169
  MEM_ types for, 209
  virtual address spaces, 206, 209-210
  (see also flushing)
debugging
  analyzing crash-dump files, 746
  breakpoints, 54
  bugcheck calls, 55-56
  checked build drivers, 738
  exceptions and, 72
  print statements for, 54
  WINDB.EXE debugger, 741-746
declaring Unicode string constants, 48
decryption, 389
deferred procedure calls (see DPCs)
delayed work requests, 352
delayed-write functionality (see writing, write-behind functionality)
depth, lookaside lists, 45

designing
  comments and, 363
  file systems, 360-365
  filter drivers, 663-667
detaching filter drivers from target, 661-663
detaching threads from processes, 200
\Device\UNC object type, 63
DeviceLock event object, 144
devices
  controlling, dispatch routines for, 584-599
  device IRQLs, 11
  device object extension, 140-141
  device objects, 139-144, 178
    attaching filter drivers to, 622-634
    create/open requests for, 421
    detaching filter drivers from, 661-663
    representing mini-FSD, 601
  interrupt spin locks and, 95-96
  IOCTL requests, handling, 596-599
  name in event identifier messages, 90
  physical device objects, 369
  unnamed device objects, 140
  volume device objects, 371-372
  VPB structure (see VPB structure)
DPS (see distributed file systems)
direct I/O method, 184
directories, 375-376
  change notification, 388-389
  control dispatch routines, 503-525
  directory entries, 375
  directory objects, create/open requests for, 422
  information data structures for, 505-507
  in-memory abstractions (see FCB structures)
  mount points, 28
  notify change directory request, 503-504, 509-518
  quotas, 387-388
  sharing, 24, 26-27
DIRQLs (device interrupt request levels), 11
dirty pages (see modified pages)
disabling read-ahead for file streams, 341-342
disk-based drivers (see local file system drivers)
dispatch routines, FSD
  asynchronous I/O, 464-476
  byte-range locks, 562-571
  cleanup routines, 525-529
  close routines, 529-531
  create/open routines, 397-424
  directory control routines, 503-525
  driver entry routines, 390-397
  file information routines, 476-503
  file system and device control, 584-599
  flush file buffers, 554-556
  invoking, pseudocode for, 472–473
  read routines, 424-437, 449-451
  volume information, 556-561
  write routines, 437-448, 449-451
DISPATCHLLEVEL, 10, 124
  DPC queue for, 12
dispatcher database, 12
dispatcher objects, 11, 98-100
  event objects, 100-103
  mutex objects, 105-108
  semaphore objects, 108-109
  timer objects, 103-105
dispatching exceptions, 71-74
distributed file systems, 27-29, 360
  (see also network file systems)
DMA (Direct Memory Access), 256
DNLC implementation, 387
documentation of system services, lack of, 6
doubly linked lists, 50-52
doubly-mapping, 185
downgrading oplocks (see sharing oplocks)
DPCs (deferred procedure calls), 10, 104, 133
  D,PC queue, 12
DriverEntry( ), 133
drivers
  allocating IRPs, 638-639
  checked build, 738
  development issues, 36-56
  driver extension structure, 139
  driver objects, 135-139, 178
  entry dispatch routines, 390-397
  file system (see file system drivers)
  filter drivers (see filter drivers)
  free build, 737-738
  initializing, 138-139, 191

drivers *(continued)*
  initializing fast I/O support, 277-282
  installing, 65-66
  interface layer, 361
  kernel-mode, building, 736-740
  layered, 123-124
  loading, 137-139
  name in event identifier messages, 90
  pageable, 37-39
  preparing to debug, 54-56
  reporting status of, 66
  synchronization between, 93-112
  target drivers, 622
  verify volume requests, 585, 592-596
dynamic name lookup cache, 387

E

encryption, 389
end-of-file position, modifying, 487
environment systems, 6
ERESOURCE objects, 110-112, 275
errors
  deadlock conditions, 95
  in finding target objects, 400
  logging (see events, logging)
  networking, 732-733
  page fault, 230
  in read-ahead attempts, 352
  reporting with top-level IRP component, 461
  system, crashing and, 66
  in write-behind attempts, 355
ETHREAD structure, 453
events
  event log viewers, 87
    finding message file, 90
    Registry key for, 366
  event objects, 100-103
  identifiers for, 87
  logging, 86-93
ExAcquireFastMutex( ), 106
ExAcquireFastMutexUnsafe( ), 106
ExAcquireResourceExclusiveLite( ), 111
ExAcquireResourceSharedLite( ), 111
ExAcquireSharedStarveExclusive( ), 111
ExAcquireSharedWaitForExclusive( ), 112
ExAllocateFromNPagedLookasideList( ), 45
ExAllocateFromPagedLookasideList( ), 45
ExAllocateFromZone( ), 43, 145
ExAllocatePool routines, 39, 234
  allocating zones with, 42
  ExAllocatePooK), 145
  ExAllocatePoolWithTag( ), 45, 233-234
  ZwAllocateVirtualMemory( ) versus, 208
except clause (see try-except construct)
EXCEPTION_ACCESS_VIOLATION exception, 69
EXCEPTION_EXECUTE_HANDLER value, 78
EXCEPTION_POINTERS structure, 79
exceptions
  call frame-based handlers, 72
  dispatching, 71-74
  exception filters, 77-81
  exception frame, 68
  exception frames, 15
  FSD design and, 363
  processing outcomes, 68-71
  structured exception handling, 290
  structured exception handling (SEH), 74-86
  termination handlers with, 86
  try-except construct, 76-81
  try-finally construct, 76-77, 81-86
  in user mode, 73
exclusive oplocks, 572-573, 579-583
ExDeleteResourceLite( ), 111
executable image file mappings, 219-220
execution context, 128, 130-134
Executive, 9, 15-19
  dispatcher objects (see dispatcher objects)
  initializing components of, 189-192
  RTLs (see RLTs)
  synchronization and, 93
Executive spin locks, 96
ExExtendZone( ), 44
ExFreeToZone( ), 43
ExInitializeFastMutex( ), 106
ExInitializeNPagedLookasideList( ), 45
ExInitializePagedlookasideList( ), 45
ExInitializeResourceLite( ), 111
ExInitializeResourceLite( ), 263
ExInitializeSListHead( ), 52
ExInitializeWorkItem( ), 469
ExInitializeZone( ), 43
ExInterlockedAllocateFromZone( ), 43
ExInterlockedFreeToZone( ), 43

ExInterlockedPopEntryList( ), 50
ExInterlockedPopEntrySList( ), 52
ExInterlockedPushEntryList( ), 50
ExInterlockedPushEntrySList( ), 52
ExQueryDepthSListHead( ), 52
ExQueueWorkItem( ), 351-352
ExReleaseFastMutex( ), 107
ExReleaseFastMutexUnsafe( ), 107
ExReleaseResourceForThreadLite( ), 111
extending zones, 44
ExTryToAcquireFastMutex( ), 107
ExTryToAcquireResourceExclusiveLite( ), 111

F

facility code, event, 88
fast I/O, 122, 277-282
  dangers of, 536-537
  dispatch table entry for path, 139
  evolution of, 532-535
  handling, 537-546
  I/O Manager and, 348
  pseudo, routines for, 546-552
  top-level IRP component for, 460
  writing custom routines, 545
FAST_IO_DISPATCH structure, 280
fast mutex objects, 105-107
FASTFAT file system, 246
  file resource acquisition hierarchy, 550
  file security, 368
FastIoCheckIfPossible( ), 538-541
"Fatal System Error" message, 56
faults (see page faults, handling)
FCB (file control block) structures, 262, 274-276, 368-369, 375-384
  CcUninitializeCacheMap( ) and, 331
  file size changes and, 268
field offset, calculating, 52
file servers, network, 118
  Cache Manager and, 259
file streams, 245
  allocating, 346
  allowing fast I/O on, 538-541
  attribute modification structures, 486-489
  byte-range lock requests for, 563
  caching, 245
  exclusive oplocks on, 572-573
  FCB structure for, 262, 268, 274-276
  file information dispatch routines, 476-503
  information data structures, 481-486
  manipulation functions for, 255, 334-344
  noncached requests and data consistency, 462–464
  opening, 282-287
  parsing paths of, 398-399
  quotas for, 387-388
  read-ahead for, 341-342, 349
  renaming, 477-479
  time attributes, 481
  truncating, 240-241
  write-behind and, 353
File System Control (see FSCTL interface requests)
file system drivers
  bypassing (see fast I/O)
  Cache Manager and, 258-259, 273-293
  cleanup requests to, 328-332
  close requests, 332-333
  data structures of, 386-390
  development issues, 36-56
  dispatch routines
    byte-range locks, 562-571
    cleanup, 525-529
    close, 529-531
    create, 397-424
    directory control, 503-525
    driver entry, 390-397
    file information, 476-503
    file system and device control, 584-599
    flush file buffers, 554-556
    read, 424-437, 449-451
    volume information, 556-561
    write, 437-448, 449-451
  functionality of, 21
  I/O Manager interface standard, 29
  interactions with VMM, 233-241
  interface of, 32-33
  mini-FSDs (file system recognizers), 599-614
  synchronization issues, 401-405
  top-level IRP components, 451-461
  verify volume requests and, 593-596

file system recognizers, 599-614
file systems
  controlling, dispatch routines for, 584-599
  data structures of, 367-390
  design of, 360-365
  distributed (distributed file systems)
  how they are used, 30-32
  in-memory data structures, 367, 380
  layout, logical volume, 23
  network/remote (see network file systems)
  Registry interactions, 365-367
  run-time library (FSRTL) routines, 454, 541-545
  types of, 22-29
  VMM and, 18
FILE_ device characteristics, 144
FILE_COMPLETE_IF_OPLOCKED flag, 576
FILE_OPBATCH_BREAK_UNDERWAY value, 578
FILE_OPLOCK_BROKEN_TO_LEVEL_2 value, 578
F1LE_OPLOCK_BROKEN_TO_NONE value, 578
FILE_WRITE_THROUGH flag, 325
filenames
  long (not 8.3 format), 484, 507
  network redirectors and, 60-64
  structures for, 483–484
  valid character set for, 23
files, 375-376
  byte-range locks (see byte-range locks)
  closing, 31
  crash-dump, analyzing, 746
  file information dispatch routines, 476-503
  file object structure, 31
  file objects, 123, 175-178, 179, 386
    byte offset, setting, 486
    byte-range lock requests for, 564
    closing, 332-333
    create/open requests for, 422
    fields of, Cache Manager and, 261-266
    waitability of, 178, 182
  flushing buffers, dispatch routines for, 554-556
  in-memory abstractions of (see FCB structures)
  log files (see logging)
  mapping, 215-217
    section objects, 219-223
    views into files, 219, 223
    for virtual block caching, 246, 249
    (see sharing memory)
  multiple linked, 375
  opening, 30, 58
  paths to (see pathnames)
  purging, 335-337
  reading data from, 31
  representation of in memory, 369-386
  sharing, 24, 26-27
  size of
    Cache Manager and, 267-269
    CcSetFileSizes( ), 334-335
    synchronizing changes to, 268
    top-level IRP component and, 459-461
  stub files, 621
filter drivers, 33-36, 118, 615-618
  attaching to targets, 622-632
    IRP routing after, 632-634
  Cache Manager and, 260
  create/open requests and, 635-636
  designing, 663-667
  detaching from targets, 661-663
  development issues, 36-56
  device objects and, 143, 178
  examples of, 619-622
  filter-driver device objects, 622
filters, exception, 77-81
finally clause (see try-finally construct)
flushing
  cache, 325-328
  file buffers, dispatch routines for, 554-556
  modified (dirty) pages, 224-229
  periodically (see writing, write-behind functionality)
  pinned buffers, 318
FO_CLEANUP_COMPLETE flag, 529
FO_FILE_FAST_IO_READ flag, 528
FO_FILE_MODIFIED flag, 528
FO_FILE_SIZE_CHANGED flag, 528
FO_SEQUENTIAL_ONLY flag, 351

FO_SYNCHRONOUSJO flag, 182
fork(), 206
format utility, 23
fragmentation of system memory, 41
free build drivers, 737-738
free pages, 203
FsContext field (file object), 261-264
FSCTL interface requests, 589-592
  data transfer methods, 585-588
  oplock requests, 577-583
  types of, 584-585
FSCTL_DISMOUNT_VOLUME function, 589-591
FSCTL_IS_PATHNAME_VALID function, 591
FSCTL_IS_VOLUME_MOUNTED function, 591
FSCTL_LOCK_VOLUME function, 589
FSCTL_MARK_VOLUME_DIRTY function, 591
FSCTL_OPBATCH_ACK_CLOSE_PENDING code, 583
FSCTL_OPLOCK_BREAK_ACKNOWLEDGE code, 582
FSCTL_QUERY_RETRIEVALJPOINTERS function, 591
FSCTLJJNLOCKJVOLUME function, 589
FSDs (see file system drivers)
FSRTL_COMMON_FCB_HEADER structure, 274, 283
FSRTL_COMMON_FCB_HEADER type, 261-262
FSRTL_FLAG_ACQUIRE_MAIN_RSRC_EX flag, 550
FSRTL_FLAG_ACQUIRE_MAIN_RSRC_SH flag, 550
FsRtlAcquireFileForModWrite( ), 550-551
FsRtlCopyRead( ), 537, 542-543, 545
FsRtlCopyWrite( ), 537, 542-545
FsRtlEnterFileSystem( ), 545-546
FsRtlExitFileSystem( ), 545-546
FsRtlNotifyCleanup( ), 518, 527
FsRtlNotifyFullReport( ), 527
FsRtlRegisterUncProvider( ), 64
FSRTL-supplied routines, 454, 541-545
functions
  copy interface-related, 256
  defined by I/O Manager, 157-159
  exception filter function, 78-81
  file stream manipulation, 255, 334-344
  (file system) run-time, 113
  MDL interface, 256-257
  names of, 363
  pinning interface, 258
  Unicode character manipulation, 47-48

G

GetExceptionCode( ), 79
GetExceptionInformation( ), 79
global
  DPC queue, 12
  name space for DFSs, 28
  PFN lock, 204
  root directory, 16, 57
  timer queue, 12
  variables, 140-141
granularity
  cache map, 271
  read-ahead, 304-305, 351

H

HAL (hardware abstraction layer), 8, 188
HalDisplayString( ), 56
handles, 17
  object, 134-135
  open handle count, 381-384
handling
  exceptions (see exceptions)
  fast I/O (see fast I/O)
  IRPs, 161-163
  page faults, 230-232
  termination (see termination handlers)
  traps (see traps)
  user-space buffer points, 183-185
hardware
  abstraction layer (HAL), 188
  HAL (hardware abstraction layer), 8
  hardware priority, 10
  independence of I/O subsystem, 126
  privilege levels, 7-8
headers
  IRP, 146-151
  object headers, 16-17
help, 750-751
  resources for further reading, 747-749
hierarchical storage management (HSM), 29, 260

hierarchy, drivers, 123-124
  inverted-tree format, 16
  logical volumes, 23-24
HIGH_LEVEL, 124
HIGHEST_LEVEL, 11
HSM (hierarchical storage management), 29, 260
  filter drivers for, 620-622
hypercritical work requests, 352
hyperspace area, 199

I

I/O errors (see errors)
I/O Manager, 19
  allocating IRPs, 636-638
  Cache Manager and, 348
  filter drivers (see filter drivers)
  functionality of, 119-122
  functions defined by, 157-159
  initializing components of, 191-192
  kernel-mode drivers interface standard, 29
  mounting logical volumes and, 371
  parsing object pathnames, 58-59
  pathnames from, 408
  reference count and, 381
  system service calls of, 32
  verify volume requests and, 593
I/O request packets (see IRPs)
I/O requests
  breaking into associated IRPs, 648
  in general, 180-185
  packets of (see IRPs)
  processing flow for (see filter drivers)
I/O Status Block, 174-175
I/O subsystem, NT, 117-119, 122-128
  (see also I/O Manager)
identifiers for events, 87
idle thread, 12
IDT (interrupt dispatcher table), 12
*ifdef statements around breakpoints, 54
image file mappings, 219-220
image section objects, 238-240
inheritance, priority, 14
initialization
  Cache Manager, 345
  discarding code after, 38
initialized state, 13
InitializeListHead( ), 50
initializing
  Cache Manager, 190
  cache operations, 255
  Configuration Manager, 190-191
  drivers
    fast I/O, 277-282
    routine for, 138-139
  ERESOURCE structures, 110
  event log entries, 91-92
  event objects, 101
  Executive components, 189-192
  file object fields, by Cache Manager, 261-266
  I/O Manager components, 191-192
  kernel, 189-192
  link list anchors, 50, 52
  spin locks, 97
  Unicode strings, 47
  VCB structure, 609-611
  zone headers, 43
in-memory data structures, 367, 380
InsertHeadList( ), 51
insertion strings in event identifier messages, 90
InsertTailList( ), 51
installing kernel-mode drivers, 65-66
integral subsystems, 7
Intel x86 MMU, 217
interactive debugging, 741-746
interface to file system drivers, 32-33
interfaces, Cache Manager, 255-258, 271-293
  routine synchronization, 274
  (see also under specific interface name)
intermediate drivers, 118
interrupts
  APCs (see APCs)
  interrupt dispatch table (see IDT)
  interrupt request levels (see IRQLs)
  interrupt service routines (see ISRs)
  interrupt spin locks, 95-96
  interruptibility of I/O subsystem, 124-125
inversion, priority, 14
inverted-tree format, 16
IoAcquireVpbSpinLock( ), 174
IoAllocateErrorLogEntry( ), 91-92
IoAllocateIrp( ), 122, 145, 146, 348, 636-638

loAllocateMdK), 121, 348
IoAttachDevice( ), 143, 372, 630-632
IoAttachDeviceByPointer( ), 143, 628-630, 632
IoAttachDeviceToDeviceStack( ), 630, 632
IoBuildAsynchronousFsdRequest( ), 146, 639-642
IoBuildDeviceIoControlRequest( ), 146, 609, 643-647
IoBuildSynchronousFsdRequest( ), 146, 642-643
IoCallDriver( ), 121, 122, 162, 348
IoCheckShareAccess( ), 424
IoCompleteRequest( ), 122, 155, 163-169, 651, 653-656
IoCreateDevice( ), 137, 140
IoCreateStreamFileObject( ), 331-332, 508
IoCreateSynchronizationEvent( ), 101
IOCTL requests
  building IRPs for, 645-647
  handling, 596-599
IoDetachDevice( ), 662
IoFreeIrp( ), 171
IoGetCurrentProcess( ), 200
IoGetDeviceObjectPointer( ), 627, 632
IoGetDeviceToVerify( ), 153, 593
IoGetRelatedDeviceObject(),627,632-634
IoGetTopLevelIrp( ), 455
Iolnitializelrp( ), 171, 638-639
IoInitializeTimer( ), 144
IoIsOperationSynchronous( ), 181-182, 465-466
IoMakeAssociatedIrp(), 146, 647-648
IoMarkIrpPending( ), 149, 160
IopCheckVpbMounted( ), 172
IopCloseFile( ), 165, 382, 525, 530
IopCompleteRequest( ), 167-168
IopDeleteFile( ), 382, 530-531
IopFreeIrpAndMdls( ), 165
IopInvalidDeviceRequest( ), 138
IopLoadDriver( ), 137
IoRaiseInformationalHardError( ), 153, 348, 461
IoReleaseVpbSpinLock( ), 174
IoRemoveShareAccess( ), 529
IoSetCompletionRoutine( ), 160, 651-653
IoSetDeviceToVerify( ), 593
IoSetHardErrorOrVeriFvDevice( ), 592
IoSetHardErrorOrVerifyDevice( ), 153
IoSetTopLevelIrp( ), 454-455
IoStartNextPacket( ), 143, 153
loStartNextPacketByKeyC), 153
IoStartPacket( ), 143, 153
IoVerifyVolume( ), 144, 172, 593
IoWriteErrorLogEntry( ), 91-93
IPIJLEVEL, 11
IRP_DEFER_IO_COMPLETION flag, 167
IRP_MJ_ codes, 158
IRP_MJ_CLEAN function, 381
IRP_MJ_CLEANUP function, 328, 526
IRP_MJ_CLOSE function, 332, 529-531
IRP_MJ_CREATE function, 381
IRP_MJ_FILE_SYSTEM_CONTROL function, 602-603
IRP_MJ_QUERYJNFORMATION type, 481-486
IRP_MJ_QUERY_VOLUME_INFORMATION function, 557-560
IRP_MJ_SET_INFORMATION type, 486-489
IRP_MJ_SET_VOLUME_INFORMATION function, 560-561
IRP_MN_LOAD_FILE_SYSTEM function, 585
IRP_MN_MOUNT_VOLUME function, 585
IRP_MN_NOTIFY_CHANGE_DIRECTORY type, 509-518
IRP_MN_QUERY_DIRECTORY type, 505-508
IRP_MN_UNLOCK_ functions, 566-567
IRP_MN_USER_FS_REQUEST function, 584
IRP_MN_VERIFY_VOLUME function, 585
IRP_NOCACHE flag, 248
IRP_PAGING_IO flag, 182
IRP_SYNCHRONOUS_IRP flag, 182
IRP_SYNCHRONOUS_PAGING_IO flag, 182
IrpContext structure, 466-469
IRPs (I/O request packets), 98, 122
  allocating, 144-146, 150
  associated vs. master, 647-650
  building, 636-650
  completion routines, 163-169, 649-650, 650-661
  handling asynchronously, 149-150
  I/O Status Block, 174-175
  key concepts, 169-172
  master versus associated, 147
  processing, 161-163

IRPs (*continued*)
  queuing, 132-133, 143
  reusing, 154-161
  routing, after filter driver
    attach, 632-634
  SetFileInformation, 269
  stack locations, 145, 154-161
  structure of, 146-154
  top-level component, 451-461
IRQLs (interrupt request levels), 10-11, 124
  for completion routines, 170, 656-657
  device (DIRQLs), 11
  Executive spin locks and, 96
  for PFN database lock, 204
  zone manipulation and, 43
IsListEmpty( ), 51
ISRs (interrupt service routines), 124
  arbitrary thread context, 133

K

KdBreakPoint( ), 54
KdPrintO, 54
KeAcquireSpinLock( ), 97
KeAcquireSpinLockAtDpcLevel( ), 97
KeAttachProcess( ), 200
KeBugCheck( ), 55, 73
KeBugCheckEx( ), 55
KeCancelTimer( ), 105
KeClearEvent( ), 103
KeDetachProcess( ), 200-201
KeEnterCriticalRegion( ), 546
KeEnterCriticalregion( ), 107
KeInitializeEventX), 101
KeInitializeMutex( ), 108
KeInitializeSemaphore( ), 109
KeInitializeSpinlock( ), 97
KeInitializeTimeEx( ), 104
KeInitializeTimer( ), 104
KeLeaveCriticalRegion( ), 106, 546
KeReadStateEvent( ), 103
KeReadStateMutex( ), 108
KeReadStateSemaphore( ), 109
KeReadStateTimer( ), 105
KeReleaseMutex( ), 108
KeReleaseSemaphore( ), 109
KeReleaseSpinLock( ), 97
KeReleaseSpinLockFromDpcLevel( ), 97
KeResetEvent( ), 101-103
kernel, 9-15
  initializing, 189-192
  memory for (see memory)
  objects of (see objects)
  spin locks, 94-98
kernel mode, 7-9
  building drivers for, 736-740
  determining if requestor mode, 148-149
  drivers for (see drivers)
  filter drivers (see filter drivers)
  kernel stack and, 45
  special file system implementations, 29
  threads of, 130
  VMM with, 235
kernel space, 198
kernel stack, 45-46
KeSetEvent( ), 101-103
KeSetTimeK), 105
KeSetTimerEx( ), 105
KeSynchronizeExecution( ), 95
KeWaitFor routines, 99
keys, Registry (see Registry)
KiDispatchException( ), 68, 71-74
KiInitializeKerneK), 189
KMODEJEXCEPTION_NOT_HANDLED
    error, 73
KSPINJ.OCK type, 51

L

LAN Manager
  IRPs and, 171
  oplocks (see opportunistic locking)
  (see also networking)
LAN Manager Network, 25
layered drivers, 123-124
layered FSD design, 361-362
layout, file system, 23
lazy-write (see writing, write-behind
    functionality)
libraries, run-time (see FSRTL-supplied
    routines; RTLs)
linked lists, 49-54
linking device objects, 142
list depth, lookaside lists, 45
lists of page frames, 203
loading
  drivers, 137-139
  Windows NT, 185-193

local file system drivers, 22-24, 27
local procedure calls (see LPC facility)
locality of reference, 350
locking
  byte-range, 386-387, 536
    dispatch routines for, 562-571
  DeviceLock event object, 144
  ERESOURCE (see ERESOURCE objects)
  by file system drivers, 233
  mutex objects, 105-108
  opportunistic, 388, 571-584
    bypassing FSD and, 536
    types of oplocks, 572-574
  read/write locks, 110-112
  spin locks, 94-98
  termination handlers and, 84
  VPB structure, 173
logging
  Cache Manager and, 341
  for fast recovery, 389
  obtaining dirty pages list, 342-343
logging events, 86-93
logical devices (see devices)
logical disks, 22
Logical Sequence Number (LSN), 312
logical volumes
  create/open requests for, 421
  device objects, 371-372
  disallowing concurrent operations to, 402-403
  file system layout of, 23
  managers of, 22-23
  mounting, 371-372
  quotas, 387-388
  verifying, 585, 592-596
  VPB structure, 172-174
long filenames, 484, 507
lookaside lists, 44-45
  allocating IRPs, 145
LPC facility, 18
LSN (Logical Sequence Number), 312

M

MACH operating system, 4
mapping
  cache map granularity, 271
  cache maps, 266
  files, 215-217
    section objects, 219-223
    views into files, 219, 223
    for virtual block caching, 246, 249
  mapped objects, 213, 216-217
  mapped page threads (see MPW threads)
  private cache map structure, 266, 271, 289
  shared cache map structure, 266, 272
master IRPs, 147, 647-650
MDLs (memory descriptor lists), 121, 234-235
  associated with IRPs, 146, 166, 168-169
  direct I/O method and, 184-185
  MDL interface, 256-257, 319-324
MEM_ (de)allocation types, 208-209
memory, 36-46
  access violations, 69
  allocating, 39-41
  allocating (see allocating)
  caching (see caching)
  checking for, 204
  committed versus reserved, 206
  descriptor lists (see MDLs)
  device object extension, 140-141
  file system recognizers and, 600
  fragmentation of, 41
  FSD design and, 364
  handling page faults, 230-232
  in-memory data structures, 367, 380
  kernel stack, 45-46
  lookaside lists, 44-45
  Management Unit (MMU), 210-211
  managing (see VMM)
  page frames, 201-204
  paged versus nonpaged, 95
  paged vs. nonpaged, 37-39
  physical, managing, 201-204
  purging physical, PPTs and, 218-219
  remote data storage, 28
  representing files in, 369-386
  shared, 213-224, 237
    page fault and, 231-232
  TLS (thread-local storage), 453-455, 457
  types of, 40
  virtual address space, 196-201
  zones, 41-44, 145
memory-mapped
  files (see sharing memory)
  I/O device registers, 210

messages
  for event identifiers, 87, 89-90
  how event log viewer finds, 90
  LPC facility for (see LPC facility)
metadata, 245, 376
  modification requests, 488
methods, 123
MiDispatchFauW), 230-232
MiEnsureAvailablePageOrWait( ), 204
mini-FSDs (see file system recognizers)
MiResolveDemandZeroFault( ), 232
MiResolvePageFileFault( ), 230-231
MiResolveProtoPteFault( ), 232
MiResolveTransitionFault( ), 231
MiWriteComplete( ), 228-229
MmAccessFault( ), 230
MmAllocateContiguousMemory( ), 41
MmAllocateNonCachedMemory( ), 41, 204
MmCanFileBeTruncated( ), 240-241
MmCheckCachedPageState( ), 347
MmCreateSection( ), 345
MmExtendSection( ), 547
MmFlushImageSection( ), 238-240
MmFlushSection( ), 346
MmGetSystemAddressForMdK), 185, 199, 234, 320
MmLockPagableCodeSection( ), 38
MmLockPagableCodeSectionByHandle( ), 38
MmLockPagableDataSection( ), 38
MmLockPagableDataSectionByHandle( ), 38
MmLockPageableCodeSection( ), 233
MmLockPageableDataSection( ), 233
MmMapViewInSystemCache( ), 346
MmPageEntireDriver( ), 38
MmPurgeSection( ), 347
MmQuerySystemSize( ), 236-237, 345
MmResetDriverPaging( ), 38
MmSetAddressRangeModified( ), 347
MMU (Memory Management Unit), 210-211, 217
MmUnlockPagableImageSection( ), 38
MmUnmapViewInSystemCache( ), 346
modified (dirty) pages, 203
  Cache Manager functions for, 342-344
  CcSetDirtyPinnedData( ), 311-313
  flushing, 224-229
  maximum number of, 337-338
  MmFlushImageSection( ) and, 239-240
  writer threads (see MPW threads)
modularity of I/O subsystem, 127-128
mounting
  IRP_MN_MOUNT_VOLUME for, 585
  logical volumes, 23, 371-372
  mount points, 28
  requests, IRPs for, 166
MPR module, 60-62, 729-735
MPW threads, 166, 225-229
multiple
  Multiple Provider Router (see MPR module)
  Multiple UNC Provider (see MUP module)
  network redirectors, 60
  physical disks (see logical volumes)
multiple linked files, 375
Multiple Provider Router (see MPR)
MULTIPLE_IRP_COMPLETE_REQUESTS error, 164
multiprocessors and I/O subsystem, 127
MUP module, 62-64
MUST_SUCCEED_POOL_EMPTY error, 40
mutex objects, 105-108

TV

name space
  distributed file system, 27-29
  mounted logical volumes, 23-24
  of Object Manager, 16, 56-60
names
  DNLC implementation, 387
  function, 363
  path (see pathnames)
  renaming file streams, 477-479
  UNC (Universal Naming Convention), 62-64
net command, 6l
network file systems, 24-27
  LAN Manager Network, 25
  (see also distributed file systems)
networking
  error codes, 732-733
  MPR for, 729-730
  network file servers, 118
    Cache Manager and, 259
  network provider DLL, 730-735
  network providers, 60

networking *(continued)*
  network redirectors, 24, 26, 118
    Cache Manager and, 259, 273-293
    handling filenames, 60-64
    pathnames supplied to, 408
  opportunistic locks, 388, 571-584
    bypassing FSD and, 536
  routing, 731
  transport protocols, 26
noise bits, 351
noncached I/O requests, 250-252
  data consistency and, 462-464
nonimage mappings, 219-220
nonpaged memory, 37-39
  allocating, 39-41
  spin locks and, 95
NonPagedPool type, 40
NonPagedPoolCacheAligned type, 40
NonPagedPoolCacheAlignedMustSucceed type, 40
NonPagedPoolMustSucceed type, 40
notification event objects, 101
notification timer objects, 104
notify change directory request, 503-504, 509-518
not-signaled state, 98
NPAddConnection( ), 61-62
NPGetCapsO, 733-735
NT (see Windows NT)
NtAllocateVirtualMemory( ), 207
NtCancelIoFile( ), 727
NtClose( ), 31, 530
NtCreateFile( ), 30, 672-681
NtCreateSection( ), 220, 345, 547
NtCurrentProcess( ), 207
NtDeleteFile( ), 725
NtDeviceIoControlFile( ), 723-725
NtFlushBuffersFile( ), 726
NTFS file system, 360
NTFS implementation, 245, 246
NtFsControlFile( ), 718-722
NtLockFile( ), 709-712
NtNotifyChangeDirectoryFile( ), 695-698
NtOpenFile( ), 681-683
NtQueryDirectoryFile( ), 689-694
NtQueryEaFile( ), 703-706
NtQueryInformationFile( ), 698-700
NtQueryVolumeInformationFile( ), 714-716
NtReadFile( ), 31, 120, 683-686
NtSetEaFile( ), 706-709
NtSetInformationFile( ), 700-703
NtSetVolumeInformationFile( ), 716-718
NtUnlockFile( ), 712-714
NtWriteFileO, 686-689

O

ObCreateObjectType( ), 191,525
ObDereferenceObject( ), 135, 508, 530
Object File System (OFS), 360
Object Manager, 15-17
  name space of, 16, 56-60
objects, 15, 123
  container objects, 398
  control objects, 12
  controller objects, 123
  device (see device objects)
  dispatcher objects, 11, 98-109
  driver (see drivers, driver objects)
  driver extension structure, 139
  ERESOURCE objects, 275
  event objects, 100-103
  file object (see files, file objects)
  mapped, 213, 216-217
  names for, 56
  NT Object Model, 123
  object handles (see handles)
  object table, 129
  overall relationships between, 178-180
  persistent, 375
  physical device objects, 369
  process objects, 128
  processes (see processes)
  reference count, 134
  section objects, 219-223
  semaphore objects, 108-109
  standard object headers, 16-17
  symbolic links, 57
  threads (see threads)
  timer objects, 103-105
  types of, 11-12
  volume device objects, 371-372
ObOpenObjectByPointer( ), 223
ObReferenceObjectByHandle( ), 134
ObReferenceObjectByPointeK), 530
OFS (Object File System), 360
on-disk data structures, 361, 367
open handle count, 381-384

opening
  CCB structure for, 384-385
  create/open dispatch routines, 397-424
  file streams, 282-287
  write-through requests during, 326
opening files, 30, 58
operating systems
  in general, 3-9
  I/O support (see I/O Manager)
  interactions with FSD, 363
  interface to file system driver, 32-33
  memory of, 197
  OS loader startup routine, 187-189
  parsing file stream paths, 398-399
  responding to exceptions, 67-68
opportunistic locks (oplocks), 388, 571-584
  bypassing FSD and, 536
  types of, 572-574
order
  IRP completion routines, 160
  resource acquisition, 549-551
  stack locations, 156
OS loader startup, 187-189
OS/2 subsystem, 7
overlapped I/O (see asychronous I/O)
owning threads, 110

P

packet-based I/O, 122
  (see also IRPs)
page color, 202
page faults, handling, 230-232
page file create requests, 423
page frames, 201-204
  database of (PFN database), 201-202, 211-212
  PPTs (prototype page tables), 202, 213, 217-219
page table entries (see PTEs)
page tables, 210-213
  entries in (see PTEs)
  prototype (see PPTs)
PAGE_ protection options, 208
pageable kernel-mode drivers, 37-39
paged memory, 37-39
  allocating, 39-41
  spin locks and, 95
PagedPool type, 40
PagedPoolCacheAligned type, 40
paging I/O requests, 166
  extended valid data length, 460
  noncached, 250-252
  synchronizing file size changes, 268-269
panic calls (see bugcheck calls)
parse methods, 16
parsing file stream paths, 398-399
parsing object pathnames, 57-59
partitions, 22
  mini-FSDs and, 609
passing messages (see messages)
PASSIVE_LEVEL, 10, 124
pathnames
  distributed file systems and, 27
  for file objects, 177
  file stream, parsing, 398-399
  for objects, 57
  supplied to FSD, 408
paths, fast I/O (see fast I/O)
PEB (Process Environment Block) structure, 129
pending IRPs, 160
performance
  call frame unwinding and, 85
  create/open routines and, 397
  exception conditions and, 85
  fast I/O, 122, 277-282, 348, 532, 534
  fast versus normal mutexes, 106
  file mapping and, 216
  file system recognizers and, 600
  logging for fast recovery, 389
  network provider, 731
  pinning data and, 258
periodic flushing (see writing, write-behind functionality
periodic timer objects, 104
persistent objects, 375
PFN database, 201-202, 211-212
physical addresses, translating virtual addresses to, 210-213
physical device object, 369
physical devices (see devices)
physical disks, 22
physical memory (see memory; VMM)
pinning interface, 257-258, 288, 306-319
  buffer control blocks, 266-267
placeholders in event identifier messages, 90

plug-and-play support, 139
pointers, 134-135
  for filter driver attach operation, 627
  to PTE/PPTE, 202
  user-space buffer, 183-185
pool allocation, 40-41
PopEntryList( ), 50
portability, 4, 126
POSIX subsystem, 7
POWER_LEVEL, 11
PPTEs (prototype page table entries), 217-218
PPTs (prototype page tables), 213, 217-219
  page faults and, 231-232
  pointer to, 202
  purging physical memory and, 218-219
PRCBs (processor control blocks), 12
preemptibility of I/O subsystem, 125-126
print statements for debugging, 54
priority
  IQRLs (see IRQLs)
  preempting threads, 125-126
  priority inheritance, 14
  priority inversion, 14
  thread execution, 13
private
  BCB (buffer control block), 266
  cache map structure, 266, 271, 289
PrivateCacheMap field (file object), 265-266
privileged mode (see kernel mode)
privileges, hardware levels of, 7-8
processes, 13-14, 128-134
  accessing memory, 196-197
  execution context, 128, 130-134
  PEB structure, 129
  Process Manager, 17
  process object structure, 128
  process objects, 128
  virtual address space of, 196-201
  (see also threads)
processors
  control blocks for (see PRCBs)
  processor control region, 12
PROFILE_LEVEL, 11
protection, virtual addresses, 206, 208
prototype page tables (see PPTs)
PsCreateSystemThread( ), 131
pseudo fast I/O routines, 546-552
pseudo file systems, 29
PTEs (page table entries), 212-213
  pointers to, 202
public BCB, 266
purging files, 335-337
purging physical memory, 218-219
PushEntryList( ), 50

Q

querying top-level IRP component value, 453-455
queues
  DPC queue, 12
  event log entries, 93
  of IRPs, 132-133, 143
  linked lists for, 49-54
  read-ahead requests in, 351-352
  timer queue, 12
quotas, 387-388

R

RAM (see memory)
raw file system driver, 191
ReadFileO, 31
reading
  caching during, 249-252
  exclusive oplocks, 572-573
  fast I/O requests (see fast I/O)
  file data, 31
  pinned data (see pinning interface)
  read byte-range locks, 562
  read dispatch routines, 424-437
    building IRPs for, 640-641
    ways to invoke, 449–451
  read/write locks, 110-112
  read-ahead functionality, 243-244, 295, 304-306, 349-352
    AcquireForReadAhead( ), 289
    callback example for, 552-554
    disabling for file streams, 341-342
    granularity of, 304-305, 351
  synchronizing file size changes with, 268
ready-to-run state, 13
real-time priority, 13
recognizers, file system (mini-FSDs), 599-614
recording in event log (see events, logging)

recurring timer objects, 104
redirectors (see network redirectors)
redirectors, network, 118
  Cache Manager and, 259, 273-293
  pathnames supplied to, 408
reference count, 134, 381-384
  device objects and, 142
  for page frames, 202
registering
  exception handlers, 76
  network provider DLL, 62
Registry
  configuring to load mini-FSD, 607-608
  file system interaction with, 365-367
  MPR, keys for, 729-730
reinitializing drivers, 191
relative pathnames, file objects, 177
ReleaseFileForNtCreateSection(), 547-549
ReleaseForCcFlushX), 551
ReleaseForModWrite( ), 549-551
ReleaseFromLazyWrite( ), 354
  example of, 553-554
ReleaseFromReadAhead( ), 352
remote
  data storage, 28
  file systems (see network file systems)
  resources, sharing, 62-64
RemoveEntryList( ), 51
RemoveHeadList( ), 51
RemoveTailList( ), 51
repinning/unpinning BCBs, 316-318
reporting driver status, 66
requestor mode, thread, 148-149
reserved bit, events, 88
reserved memory, 206
resources for further reading, 747-749
retrying instructions after exception, 68-70
return statement, 84
reusing IRPs, 154-161
root directory, 16, 57
routing, MPR for (see MPR)
RtlAnsiStringToUnicodeString( ), 47
RtlAppendUnicodeStringToString( ), 48
RtlAppendUnicodeToString( ), 48
RtlCopyUnicodeString( ), 48
RtlDispatchException( ), 72
RtlDowncaseUnicodeString( ), 48
RtlEqualUnicodeString( ), 47
RtlFreeUnicodeString( ), 48
RtlInitUnicodeString( ), 47
RtlPrefixUnicodeString( ), 48
RTLs (run-time libraries), 112-113
  (see also FSRTL-supplied routines)
RtlUnicodeStringToAnsiString( ), 47
RtlUpcaseUnicodeString( ), 48
RtlZeroMemory( ), 69
running state, 13
run-time libraries (see FSRTL-supplied
    routines; RTLs)

S

scheduling state, thread, 13
second chance processing, 73
section objects, 219-223
  discarding associated pages, 238-240
SectionObjectPointer field (file
    object), 264-265, 283-284
security
  dispatching user-mode exceptions, 73
  encryption/decryption, 389
  FASTFAT file system, 368
  Security Reference Manager, 18
  virus detection functionality, 619-620
segment data structure, 224
SEH (structured exception handling), 74-86
  avoiding, consequences of, 75
  Cache Manager with, 290
semaphore objects, 108-109
sequential-only flag, 351
Server Message Block (8MB) protocol, 483
servers, 24
service calls (see system service calls)
SetFilelnformation IRP, 269
SetLastError( ), 732
severity code, event, 88
SFilterAttachTarget( ) (example), 623-626
SFilterBetterFSDInterceptRoutineO
    (example), 661
SFilterDeviceExtension structure
    (example), 626
SFilterSampleCompletionRoutine( )
    (example), 656
SFsdAcqLazyWrite( ) (example), 553-554
SFsdAllocatelrpContextX)
    (example), 467-469
SFsdBreakPoint( ) (example), 54
SFsdCommonDeviceControl( )
    (example), 597-598

SFsdCommonDispatch( )
(example), 473-475
SFsdCommonRead( ) (example), 469–471
SFsdFastIoCheckIfPossible( )
(example), 539-540
SFsdFCB structure (example), 378-379
SFsdFileLockAnchor structure
(example), 567
SFsdFileLocklnfo structure (example), 567
SFsdHandleQueryPath( ) (example), 599
SFsdInitializeVCB( ) (example), 609-611
SFsdNtRequiredFCB structure
(example), 379-380
SFsdPostRequesK) (example), 471-473
SFsdRead( ) (example), 466
SFsdRelLazyWrite( ) (example), 553-554
sharing
data (see synchronization)
files/directories, 24, 26-27
memory, 213-224, 237
page faults and, 231-232
oplocks, 573-574, 579-583
resources with MUP module, 62-64
shared cache map structure, 266, 272
signaled state, 98
singly linked lists, 49-50
size
file streams, modifying, 487
of files (see files, size of)
truncate size, 330
SL_PENDING_RETURNED flag, 150
S-Lists, 52
SMB protocol, 483
software interrupts (see APCs)
source device objects, 622
special file system implementation, 29
spin locks, 94-98
for PFN database, 204
stacks
stack frames, unwinding, 71, 83-86
stack locations, 145, 154-161
allocating for multiple, 155
order of, 156
standard system services, 30-32
standby pages, 203
standby state, 13
starting up Windows NT (see system boot
sequence)
StartIo( ), 136
starvation, thread, 275
state
dispatcher objects, 98
event object, 102
page frame, 203
status reports, 66
STATUS_END_OF_FILE error, 268
STATUS_FILE_LOCK_CONFLICT error
code, 562
STATUS_FS_DRIVER_REQUIRED code, 602
STATUS_IN_PAGE_ERROR exception, 300
STATUS_INSUFFICIENT_RESOURCES
exception, 300
STATUS JNVALIDJJSERJBUFFER
exception, 299
STATUS_MORE_PROCESSING_REQUIRED
code, 164-165, 170
STATUSJVIORE_PROCESSING_REQUIRED
type, 657-661
STATUS_OPLOCK_BREAK_IN_PROGRESS
code, 578
STATUS_SEMAPHORE_LIMIT_EXCEEDED
exception, 109
STATUS_SHARING_VIOLATION error, 177
STATUS_UNEXPECTED_IO_ERROR
exception, 300
"STOP" message, 55
storage (see memory)
strings
with bugcheck code, 56
Unicode characters for, 46-49
structured exception handling (see SEH)
structures (see data structures)
stub files, 621
substrings, functions for, 48
subsystems, 4-7
I/O subsystem, 117-119,122-128
(see also I/O Manager)
interruptibility of, 124—125
modularity of, 127-128
portability of, 126
preemptibility of, 125-126
symbolic links, 57
synchronization, 93-112
Cache Manager interface routines, 274
create/open routines and, 401-405
determining if called, 465-466
dispatcher objects, 11,98-109
ERESOURCE-type primitives, 275

synchronization *(continued)*
of file object operations, 177
file size changes, 268
filter driver design and, 664
FSD design and, 364
of I/O, 124
building IRPs for, 642-643
STATUS_MORE_PROCESSING_REQUIRED and, 658-661
multiprocessors and, 127
of paging I/O requests, 166
spin locks, 94—98
synchronization event objects, 101
synchronization timer objects, 104
top-level IRP component and, 459
of VPB structure, 174
synchronous
versus asynchronous, 180-183
system boot sequence, 185-193
system cache, 256, 347
(see also caching)
system control requests, 30
system errors, 66
system failure, 65
quick recovery from, 389
system services
execution context and, 131
system services, list of, 671-728

T

target device objects, 622
attaching filter drivers to, 622-632
IRP routing after, 632-634
detaching filter drivers from, 661-663
target drivers, 622
terminated state, 13
termination handlers, 71, 81-84
exception handlers with, 86
termination of caching, 328-333
test-and-set instruction, 94
testing
executable image file mappings, 219
logical volumes, 585, 592-596
truncate operation acceptability, 240-241
threads, 13-14, 129-134
APCs and, 107
arbitrary threads, 131-133
asynchronous I/O and, 124
asynchronous processing, 464-476
byte-range locks and, 562
determining requestor mode, 148-149
event objects to synchronize, 100-103
execution context, 130-134
idle thread, 12
kernel stack, 45–46
MPW threads, 166, 225-229
owning threads, 110
preemptibility of, 125-126
process execution context, 128
semaphore objects and, 108-109
spin locks, 94-98
synchronizing, 93-112
thread context, 13-15, 129, 133-134
process address space and, 199
thread objects, 129
thread-local storage (TLS), 453-455, 457
trapping (see traps)
user-mode versus thread-mode, 130
zero page thread, 192
(see also processes)
time attributes, file streams, 481
TimeOut interval
waiting for dispatcher objects, 100
timer objects, 103-105
timer queue, 12
TLB (Translation Lookaside Buffer), 211, 213
TLS (thread-local storage), 453-455, 457
top-level IRP component, 451–461
setting and querying value of, 453–455
top-level writers, 459–460
translation
Lookaside Buffer (see TLB)
maps (see page tables)
of virtual addresses, 210-213
transport protocols, 26
traps, 14-15
trap frame, 67-68
trap frames, 14
trap handlers, 14, 67-68, 76
tree structure (see hierarchy, drivers)
truncate operation acceptability, testing, 240-241
try-except construct, 76-81
try-finally construct, 76-77, 81-86
try_return macro, 86

U

UNC (Universal Naming Convention)
  MUP module, 62-64
Unicode characters, 46-49
UNICODE_STRING structure, 46
unlock requests, 566-567, 570-571
  (see also locking)
unnamed device objects, 140
unpinning BCBs, 316-318
unpinning data, 258, 306, 315-316
unwinding stack frames, 71,83-86
user mode, 4-7
  determining if requestor mode, 148-149
  exceptions in, 73
  threads of, 130-131
  VMM with, 235
user space, 197
user stack
  switch to kernel mode and, 45
user-space buffer pointers, 183-185

V

VACB structure, 271-272
VADs (virtual address
    descriptors), 205-206, 216
validation, network provider DLL, 731
variable priority, 13
VCB structure, 373-375
  initializing, 609-611
VDM subsystem, 7
veneer, file system, 361
views into files, 219, 223
virtual addresses, 196, 204-213
  address space, 196-201
  control block (see VACB)
  descriptors for (VADs), 205-206
  manipulating, 205-210
  translating, 210-213
virtual block caching, 246-248
virtual devices (see devices)
virtual DOS machine (see VDM subsystem)
Virtual Memory Manager (see VMM)
virus detection functionality, 619-620
VMM (Virtual Memory Manager), 18,
    194-196
  Cache Manager and, 344-348
  file mapping, 215-217
  handling page faults, 230-232
  initialization of internal states, 189
  interactions with file system
    drivers, 233-241
  mapping in driver code, 137
  paging drivers, routines for, 38
  paging I/O requests, 166
  physical memory management, 201-204
  virtual addresses, 196, 204-213
vnodes (see FCB structures)
volume device objects, 371-372
volume information requests, 556-561
Volume Parameter Block (see VPB
    structure)
VPB structure, 140, 172-174, 179, 369,
    372-373
VPB structures
  routing IRPs after filter driver
    attach, 633

W

wait routines, 99
waiting state, 13
  file objects in, 178, 182
Win32 subsystem, 6-7
WINDBG.EXE debugger, 741-746
Windows NT
  boot sequence of, 185-193
  Cache Manager (see caching, Cache
    Manager)
  core architecture of, 4-9
  Event Log (see events, logging)
  Executive (see Executive)
  I/O Manager (see I/O Manager)
  I/O subsystem, 117-119, 122-128
  Kernel (see kernel)
  modes of, 4-9
  Object Manager (see Object Manager)
  Object Model, 123
  Registry (see Registry)
  system services, list of, 671-728
  Virtual Memory Manager (see VMM)
Windows on Windows, 7
wise characters (see Unicode characters)
WNetAddConnection( ), 6l
WNetAddConnection2( ), 6l
WNetSetlastErrorO, 732
worker threads (see threads)
WOW subsystem, 7

writing
  caching during, 252-254
  copy-on-write feature, 206
  in event log (see events, logging)
  exclusive oplocks, 572-573
  fast I/O requests (see fast I/O)
  pinned data (see pinning interface)
  read/write locks, 110-112
  synchronizing file size changes
    with, 268
  top-level writers, 459-460
  write-behind functionality, 243, 245,
    352-355
    AcquireForLazyWrite( ), 289, 354
    Cache Manager component for, 254
    call back example for, 552-554
    CcSetDirtyPinnedData( )
      and, 312-313
    FSRTL_FLAG2_DO_MODIFIED_
      WRITE flag, 262
    ReleaseFromLazyWrite( ), 354
  write byte-range locks, 563
  write dispatch routines, 437-448
    building IRPs for, 640-641
    ways to invoke, 449–451
  write throttling, 256
  write-through operations, 325-326
  (see also designing; reading)

Z

zero page thread, 192
zeroed pages, 203
  MiResolveDemandZeroFault( ), 232
zeroing file stream bytes, 338-339
zones, 41-44, 145
  disadvantages to, 44
  extending, 44
  ZONEJHEADER structure, 43
ZwAllocateVirtualMemory( ), 207-209
  as direct driver request, 234
ZwClose( ), 134, 530
ZwCreateSection( ), 220-223, 345
ZwFreeVirtualMemory( ), 209-210
ZwMapViewOfSection( ), 223
ZwOpenSection( ), 223
ZwUnmapViewOfSection( ), 223

About the Author

Rajeev Nagar has been working on operating systems (specifically storage management systems) for the past six years. He has designed and implemented kernel software for the Windows NT, AIX, HPUX, and SunOS platforms. His file system development work has included local, disk-based file systems, networked file systems, and distributed file systems. His undergraduate degree is in computer engineering, and he has a master's degree in computer science. Rajeev has implemented an OSF distributed file system client on the Windows NT platform, as well as other filter drivers for storage management products.

Colophon

A vulture is featured on the cover of *Windows NT File System Internals.* Vultures are divided into two famlies—New World vultures, a family that includes the majestic but near-extinct California condor, and Old World vultures. Both families are closely related to eagles and hawks, but, unlike their relatives, vultures are carrion eaters, not hunters. A vulture will rarely kill for food. Instead, they sit by and wait for another animal to die before starting to dine. Vultures often live in open country where herds of large mammals, such as cattle, can be found. They fly in slow circles, searching the ground for dead, sick, or injured animals. They also watch for running packs of jackals or hyenas, who often lead them to food. When food has been spotted, the vulture swoops down to the ground, and other circling vultures follow.

Both Old World and New World vultures have heads and necks that are almost bare, covered only by a thin layer of down. Many vultures have a thick ruff of feathers around their neck. These adaptations allow the vulture to place its head deep inside carcasses without soiling its plumage. The digestive enzymes of the vulture allow it to survive on decaying meat that would be toxic to other animals.

Although the modern view of vultures is often one of disgust and comtempt, some ancient cultures revered them as embodiments of immortality.

Edie Freedman designed the cover of this book, using a 19th-century engraving from the Dover Pictorial Archive. The cover layout was produced with Quark XPress 3-3 using the ITC Garamond font. Whenever possible, our books use Rep-Kover™, a durable and flexible lay-flat binding. If the page count exceeds Rep-Kover's limit, perfect binding is used.

The inside layout was designed by Nancy Priest and implemented in FrameMaker 5.0 by Mike Sierra. The text and heading fonts are ITC Garamond Light and Garamond Book. The illustrations that appear in the book were created in Macromedia Freehand 7.0 by Robert Romano. This colophon was written by Clairemarie Fisher O'Leary.