

INDEX

Numbers

- 16-bit hardware addresses (PCI), 471
- 16-bit PCI registers, 476
- 16-bit ports, 230
 - string functions for, 232
- 32-bit addresses
 - PCI bus I/O and memory space, 473
- 32-bit PCI registers, 483, 485-488
- 32-bit ports, 230
 - string functions for, 232
- 64-bit addresses
 - accessing PCI bus memory space, 473
- 64-bit programmable decoder, 485
- 64-bit regions and PCI registers, 483
- 8-bit ports, 230
 - reading/writing, 230
 - string functions for, 232

A

- access
 - blocking open requests, 168
 - cloning devices on open, 169-171
 - concurrent (see race conditions)
 - to device files, 164-171
 - to drivers, 59
 - to expansion board memory, 238-247
 - PCI configuration space, 480-483
 - restricting
 - to simultaneous users, 167
 - via capabilities, 137
 - to user space in Linux 2.0, 173-175
- access_ok(), 135
- active queue heads, 342

- add_timer(), 201-203, 207
- __add_wait_queue(), 287, 292
- add_wait_queue_exclusive(), 146, 179
- add_wait_queue(), 179, 287, 292
- Address Resolution Protocol (see ARP)
- address types, 371
- addresses
 - bus (see bus addresses)
 - hardware (see hardware addresses)
 - PCI, 471-474
 - for peripheral boards, 473
 - Plug and Play, 496
 - resolving, 455-458
- Adelson-Velski-Landis (AVL) tree, 515
- alias directive (modprobe), 308
- aliases for device names, 69
- alignment, data, 299
- alloc_bootmem_low_pages(), 221, 225
- alloc_bootmem_low(), 221, 225
- alloc_bootmem_pages(), 221, 225
- alloc_bootmem(), 221, 225
- alloc_kiovec(), 396, 422
 - map_user_kiobuf and, 399
- alloc_skb(), 454, 468
- allocate_resource structure, 41
- allocating
 - DMA buffers, 402-404
 - major device numbers, 57-61
 - memory, 36, 73-75
 - at boot time, 221-223
 - determining how much, 211
 - kmalloc for, 208-211

We'd like to hear your suggestions for improving our indexes. Send email to index@oreilly.com.

Index

- allocating, memory (continued)
 - by page, 214-217
 - vmalloc for, 217-220
- ports, 36-41
- resources in Linux 2.4, 40
- socket buffers, 449, 454
- allocator module, 223
- Alpha architecture
 - I/O memory management support, 411
 - porting and, 233
- alpha_machine_vector structure, 494
- analyzing crash dumps, 125
- applications vs. kernel modules, 16-21
- arch directory, 517
- ARM architecture
 - layout of boot code, 510
 - PCI DMA interface support, 411
 - porting and, 233
- ARP (Address Resolution Protocol)
 - Ethernet and, 455
 - IFF_NOARP flag and, 432, 438
 - overriding, 456
- asm directory, 17
- <asm/atomic.h> header file, 285, 291
- <asm/bitops.h> header file, 284, 291
- <asm/byteorder.h> header file, 298, 304
- <asm/current.h> header file, 21
- <asm/dma.h> header file, 414, 416, 423
- <asm/io.h> header file, 249, 422
 - accessing I/O ports, 230
 - converting between bus/virtual addresses, 404
- <asm/ioctl.h> header file, 130
- <asm/irq.h> header file, 262, 267
- <asm/msr.h> header file, 183, 205
- <asm/page.h> header file, 297, 303, 372, 376
- <asm/pcibios.h> header file, 502
- <asm/pgtable.h> header file, 218, 377
- <asm/processor.h> header file, 497
- <asm/sbus.h> header file, 412
- <asm/segment.h> header file, 95
- <asm/semaphore.h> header file, 76, 95
- <asm/system.h> header file, 228, 249
- <asm/types.h> header file, 295
- <asm/uaccess.h> header file, 78, 95, 135, 177
- <asm/unaligned.h> header file, 299, 304
- assembly language dump of code, 116
- asynchronous DMA, 401
- asynchronous notification, 159-162
 - backward compatibility issues, 173
 - drivers and, 161
- asynchronous running of task queues, 191
- atomic_add_and_test(), 286
- atomic_add(), 286, 291
- atomic bit operations, 284
 - backward compatibility issues, 289
- atomic_dec_and_test(), 286, 291
- atomic_dec(), 286, 291
- atomic_inc_and_test(), 286
- atomic_inc(), 286, 291
- atomic integer operations, 285
- atomic_read(), 286
- atomic_set(), 286
- atomic_sub_and_test(), 286
- atomic_sub(), 286, 291
- atomic_t data type, 285
- atomic.h header file, 285, 291
- autoconf.h header file, 316
- autodetecting parameter values, 42
- autoirq_report(), 260
- autoirq_setup(), 260
- automatic
 - device parameters detection, 43
 - driver configuration, 43
 - IRQ number detection, 258-262
 - shared interrupts and, 276
 - module loading/unloading, 305-311
- AVL (Adelson-Velski-Landis) tree, 515

B

- b_end_io(), 339, 368
 - clustered I/O, 341
 - “make request” function and, 346
- backward compatibility
 - access to user space, 173-175
 - asynchronous notification, 173
 - block drivers, 364-366
 - capabilities, 175
 - compiling for multiprocessor systems, 48
 - demand-loading capability, 318
 - DMA (direct memory access), 420
 - exporting symbols, 48-50

- backward compatibility (continued)
 - file_operations structure, 91-93
 - fsync method, 173
 - hardware management, 248
 - interrupt handling, 288
 - memory management, 418-420
 - programming interface, 223
 - module configuration parameters, 50
 - module usage count, 93
 - networking, 464-466
 - peripheral buses, 502
 - resource management, 47
 - seeking, 176
 - select method in Linux version 2.0, 175
 - semaphore support, 94
 - task queues/timing issues, 204
 - user space, access to, 94
 - wait queues, 172
- barrier(), 228, 249
- base address registers, 485-488
- base module parameter, 237
- base name, device, 356
- bdops (see block_device_operations structure)
- bfd (binary format description) library and ksymbols, 116
- BH (see bottom halves)
- bh->b_end_io(), 339, 368
 - clustered I/O, 341
 - “make request” function and, 346
- bibliography, 527
- __BIG_ENDIAN symbol, 298, 304
- big-endian byte order, 298
- bigphysarea patch, 222
- binary format description (bfd) library and ksymbols, 116
- binary formats, 513
- binfmt_elf.c file, 513
- bit operations, 284
 - backward compatibility issues, 289
- bit specifications, 236
- bit splitting and minor numbers, 69
- bitfields, defining ioctl commands, 130, 177
- bitops.h header file, 284, 291
- bits, clearing, 264
- blk_cleanup_queue(), 323, 366
- BLK_DEFAULT_QUEUE macro, 324, 367
- blk_dev global array, 324, 364, 367
- blk_dev_struct structure, 324
- blk_init_queue(), 323, 366
 - initializing device-specific queues, 343
- blk_ioctl(), 351, 368, 518
 - backward compatibility issues, 365
- blk_queue_headactive(), 342, 368
- blk_queue_make_request(), 346, 368
- blk_size global array, 324, 367
 - sizes array and, 357
- blkdev_dequeue_request(), 338, 368
 - end_request() and, 340
- blkdev_entry_next_request(), 337, 368
- blkdev_next_request(), 337, 368
- blkdev_prev_request(), 337, 368
- blkdev_release_request(), 338, 368
- blkdev.h header file, 323, 366
- BLKELVGET command, 351
- BLKELVSET command, 351
- BLKFLSBUF command, 350
- BLKFRAGET command, 350
- BLKFRASET command, 350
- BLKGETSIZE command, 349, 361
- blk.h header file, 328-330, 367
 - clustered requests and, 340
 - declaring DEVICE_NR first, 361
 - how macros and functions work, 339
- BLKPG command, 350
- blkpg.c file, 518
- blkpg.h header file, 351
- BLKRAGET command, 350
- BLKRASET command, 350
- BLKROGET command, 350
- BLKROSET command, 350
- BLKRRPART command, 350, 361
- BLKSECTGET command, 350
- BLKSECTSET command, 350
- blksize_size global array, 324, 367
- BLKSSZGET command, 350
- block_dev.c file, 513
- block_device_operations structure, 322
 - backward compatibility issues, 364
 - I/O operations, 323
 - removable devices, 352
- block drivers, 7
 - arrays for information about, 324
 - backward compatibility, 364-366
 - generic hard disk support, 356

Index

- block drivers (continued)
 - handling requests, 330-348
 - interrupt-driven, 362-364
 - io_request_lock and, 338
 - ioctl method and, 349-352
 - <linux/blk.h> header file (see blk.h header file)
 - loading/unloading, 321-354
 - mounting devices, 348
 - multiqueue, 342-345
 - partitionable devices and, 355-362
 - raw I/O capability, 397
 - registering/unregistering, 322-328
 - removable block devices, 352-354
 - vs. char drivers, 321
- block_fsync method, 158, 328
- blocking I/O operations, 141-153
 - blocking open requests, 168
 - testing, 153
- BogoMips value, 188
- books
 - Linux kernel, 527
 - Unix design/internals, 528
- booting
 - acquiring a dedicated buffer at, 221
 - allocating memory while, 221-223
 - kernels, 507-509
 - (non)modularized drivers and, 434
 - PCI and, 474
 - what happens before, 509-511
- bootmem.h header file, 221, 225
- bottom halves
 - BH mechanism, 271
 - of interrupt handlers, 269-274
 - marking, 272
 - task queues, 190, 197
 - tasklets and, 198-200, 270
 - writing, 273
- bounce buffers, 406
 - architectures not supporting, 411
 - streaming DMA mappings and, 409
- bridge subdirectory, 516
- bridges for PCI systems, 471
 - ignored by pcidata module, 482
- BSS segments, 379
- buffer cache and request structure, 335
- buffer_head structure, 332
 - fields for, 335
 - performing clustered I/O, 341
- buffer.c file, 513
- buffering and interrupt-driven I/O, 278
- buffers
 - buffer overruns, 112
 - DMA, 402-404
 - for printk(), 100
 - in request queues, 336
 - socket (see socket buffers)
 - user-space and raw I/O, 397-400
- bugs (see debugging; troubleshooting)
- bus addresses, 372
 - converting between virtual addresses and, 404
 - dma_addr_t type and, 406
 - DMA-based hardware and, 404
- bus architecture, 470-505
 - backward compatibility issues, 502
 - device-specific directories, 523
 - ISA interface, 494-496
 - PC/104 and PC/104+, 496
 - PCI interface, 470-494
- bus_to_virt(), 404, 422
- busy loops, 186
- busy waiting implementation, 186
- byte order
 - PCI registers and, 475, 480
 - portability and, 298
- byteorder.h header file, 298, 304
- bzImage file, 510

C

- caches, lookaside, 211-214
 - backward compatibility issues, 223
- caching problems for devices, 228, 385
- call_usermodehelper(), 311, 320
- CAP_DAC_OVERRIDE capability, 137
 - single-user access to devices, 168
- CAP_NET_ADMIN capability, 137
- CAP_SYS_ADMIN capability, 137
- CAP_SYS_MODULE capability, 137
- CAP_SYS_RAWIO capability, 137
- CAP_SYS_TTY_CONFIG capability, 137
- capabilities
 - restricted operations and, 137
 - testing for, using request_module, 306

- capability.h header file, 137, 178
- capable(), 137, 178
- Card Select Number (CSN), 496
- cardctl program, 3
- carrier signals, 451
- cdrom_device_info structure, 520
- cdrom.c file, 520
- CFLAGS variable (make), 23
- change_bit(), 284, 291
- change_mtu method, 441
 - improving performance using socket buffers, 449
- channels, DMA, 413-415
- char drivers, 6, 54-96
 - defining mechanism of, 54
 - version numbers, 55-62
 - vs. block drivers, 321
- check_disk_change(), 354, 369
- check_media_change method, 353
 - backward compatibility issues, 364
- check_mem_region(), 53, 250
 - backward compatibility issues, 47
 - working with I/O memory, 40, 239
- check_region(), 52, 250
 - backward compatibility issues, 47
 - working with I/O ports, 38, 229
- CHECKSUM_ symbols, 449
- checksums
 - adding to symbol names, 314
 - building, 317
- circular buffers, 279
 - implementing interrupt handlers, 264-266
 - for printk(), 100
- claim_dma_lock(), 416, 424
- class PCI register, 476
- classes, module, 6-8
- cleanup_module(), 16, 50
 - error handling and, 31
 - network drivers and, 434
 - releasing ports, 39
 - unregistering items, 34
 - using unique names instead of, 34
- clear_bit(), 284, 291
- clear_dma_ff(), 417, 424
- CLEAR_INTR macro, 329
- clearing bits on interface board, 264
- cli(), 252
- clock cycles, counting, 182
- clock ticks (see jiffies value)
- cloning devices on open requests, 169-171
- close method, 72
 - accessing data within partitions, 360
 - adding VMA operations, 386
 - after cloning devices on open, 171
 - for single-open devices, 165
 - vm_operations_struct structure, 381
 - (see also release method)
- closing network interface, 443-445
- clustered requests, 340
- code, delaying execution of, 186-189
- coding style, 23
- collisions, device, 36, 38
- command numbers, ioctl, 130-133
- command-line parsing, 507
- command-oriented drivers, 140
- compiler optimizations, 227
- concurrency, 20, 278-288
 - controlling transmission, 446
 - multiqueue block drivers and, 345
- concurrent access (see race conditions)
- conditional compilation, avoiding, 90
- CONFIG_DEVFS_FS, 85
 - portability issues and, 90
- CONFIG_MODVERSIONS(), 316, 320
- CONFIG_PCI(), 477, 503
- CONFIG_SMP configuration option, 48
- config.h header file, 316, 320, 477, 503
- configuration space, PCI, 473, 480-483
- configuration transactions, PCI, 473
- configuring
 - DMA controller, 415-418
 - drivers, 42-44
 - network devices, 441
 - PCI registers, 475-479
- consistent DMA mappings, 406
 - setting up, 407
- console_loglevel variable, 98
 - debugging system hangs, 118
- console.c file, 518, 522
- consoles
 - drivers/char directory and, 518
 - frame buffer consoles, 522
 - selecting for messages, 99
 - wrong font on, 140

Index

- constructor function
 - (kmem_cache_create), 212
- controlling access (see access)
- controlling-by-write, 140
- converting virtual addresses, 404
- __copy_from_user, 79, 96
- copy_from_user(), 79
 - memcpy_tofs and, 94
 - vmalloc() and, 218
- __copy_to_user, 79, 96
- copy_to_user(), 79
 - memcpy_fromfs and, 94
 - using put_user() instead of, 136
- copying, cross-space, 78
- core files, 120
- core-file (gdb command), 121
- core/skbuff.c file, 516
- counter registers, 182
- CPU modalities (levels), 19
- __cpu_to_le32 macro, 298, 304
- crash dump analyzers, 125
- CRC (cyclic redundancy check) algorithm
 - and module version control, 314
- create_bounce(), 348
- create_module system call, 9
 - using vmalloc() and, 218
- create_proc_read_entry(), 106
- cross compilation and platform
 - dependency, 27
- cross-space copying, 78
- CSN (Card Select Number), 496
- CURRENT_DEV macro, 329, 332
- current_nr_sectors field, 332
- current process, 21, 52
- current time, retrieving, 184
- current.h header file, 21
- currenttime file (jit module), 185
- CURRENT(), 330, 368
 - accessing fields in request structure, 332
- custom
 - data types, 296
 - ioctl methods for networking, 458
 - task queues, 198
- cycles_t type, 183

D

- data
 - explicitly sizing, 295
 - physical packet transport, 429, 445-450
 - protecting from race conditions, 279
 - transferring
 - for block driver requests, 332-334
 - with DMA, 401-418
 - using ioctl method, 131
 - unaligned, portability and, 299
- data structures, portability of, 299
- data types
 - for explicitly sizing data, 295
 - interface-specific, 296
 - loose typing for I/O functions, 297
 - mixing different, 294
 - portability and, 293-297
 - standard C types, 293
- dataalign program, 300
- datasize program, 293
- dcache.c file, 513
- dd utility and scull driver example, 73
- deadlocks
 - avoiding, 77
 - detecting with IKD, 124
- deallocating (see allocating)
- debugging, 97-127
 - using a debugger, 120-127
 - using Dynamic Probes, 127
 - using gdb, 120-122
 - using IKD (integrated kernel debugger), 124
 - implementing debug levels, 102
 - interrupt handling, 267
 - with ioctl method, 108
 - using kdb kernel debugger, 122-124
 - using kgdb, 125
 - using Linux Trace Toolkit (LTT), 127
 - locked keyboard, 118
 - module loading, 24
 - modules, 113-118
 - by printing, 97-103
 - with /proc filesystem, 103-107
 - by querying, 103-108
 - race conditions, 278-288
 - system faults, 110-120

- debugging (continued)
 - system hangs, 118
 - using User-Mode Linux, 126
 - by watching in user space, 108-110
 - (see also troubleshooting)
- DECLARE_TASK_QUEUE, 191, 198, 206
- DECLARE_TASKLET, 199, 206, 270, 290
- DECLARE_TASKLET_DISABLED, 199, 206
- DECLARE_WAIT_QUEUE_HEAD, 141, 143
 - jjq module and , 193
- decoders, programmable, 485
- decoding oops messages, 113-118
- DEFAULT_CONSOLE_LOGLEVEL, 98
- DEFAULT_MESSAGE_LOGLEVEL, 98
- del_timer_sync(), 202, 207
 - avoiding race conditions, 203
 - backward compatibility issues, 205
- del_timer(), 202, 207
- delay.h header file, 188, 206
- delaying execution of code, 186-188
- delete_module system call, 34
- demand-loading modules, 305-311
 - slave/master modules example, 309
- dentry field (file structure), 68
 - backward compatibility issues, 93
- depmod program, 319
- dereferencing
 - invalid pointers, 111-118
 - I/O pointers, not recommended, 240
 - memory addresses, 294
 - physical addresses, 240
- destructor function
 - (kmem_cache_create), 212
- dev_alloc_skb(), 449, 454, 468
- dev_id pointer, 254, 267
 - installing shared handlers, 275
- dev_kfree_skb(), 454, 468
- dev_mc_list structure, 462
- /dev nodes, 6
 - assigning, 57
 - char devices and, 55
 - /dev/random device, 255
 - /dev/urandom device, 255
 - dynamic major number allocation, 58
 - removing, 61
- dev structure and device initialization, 432
- dev_t type (Unix), 62
- dev_table.c file, 521
- dev_tint(), backward compatibility issues
 - for, 465
- development kernels, 11
- devfs (device filesystem), 56, 85-91
 - advantages of, 85
 - dual-mode initialization, 88
 - flags, 87
 - portability issues and, 90
 - DEVFS_FL_AUTO_DEVNUM flag, 87
 - DEVFS_FL_AUTO_OWNER flag, 87
 - DEVFS_FL_DEFAULT flag, 87
 - DEVFS_FL_HIDE flag, 87
 - DEVFS_FL_NO_PERSISTENCE flag, 87
 - DEVFS_FL_NONE flag, 87
 - DEVFS_FL_SHOW_UNREG flag, 87
 - devfs_fs_kernel.h header file, 96
 - devfs_get_flags(), 87
 - devfs_mk_dir(), 86
 - devfs_register(), 86
 - devfs_set_flags(), 87
 - devfs_unregister(), 86
- device control operations, 5
- device entry points, filesystem for, 85-91
- device files, 55
 - controlling access, 164-171
 - deleting, 61
- device filesystem (see devfs)
- DEVICE_INTR symbol, 329, 367
- device memory (see I/O memory)
- DEVICE_NAME symbol, 329, 367
- DEVICE_NO_RANDOM symbol, 329
- DEVICE_NR symbol, 329, 367
 - minor_shift value and, 356
- DEVICE_OFF macro, 329
- DEVICE_ON macro, 329
- DEVICE_REQUEST symbol, 329
- device-dependent symbols, 328-330
- deviceID PCI register, 476
- devices
 - assigning virtual addresses to, 242
 - autodetecting parameters of, 43
 - base name of, 356
 - block (see block drivers)
 - caching problems, 228, 385
 - character (see char drivers)
 - classes of, 6-8
 - cloning on open requests, 169-171

Index

- devices (continued)
 - collisions between, 36
 - creating using devfs, 86, 88
 - DMA and, 401-418
 - file operations on, 63-66
 - hardware management, 226-250
 - hot-pluggable, handling, 489-493
 - identifying type with ls command, 55
 - interrupts (see interrupt handlers)
 - names of, 56
 - aliases for, 69
 - dynamic major number allocation, 58
 - removing, 61
 - network (see network drivers)
 - partitionable, 355-362
 - accessing data within partitions, 360
 - PCI (see PCI)
 - reading and writing, 78-84
 - reading data from, 157
 - removable, 352-354
 - removing using devfs, 86
 - seeking, 163
 - single-open, 165
 - single-user access to, 167
 - truncating on open, 71
 - version (see version numbering)
 - writing control sequences to, 140
 - writing data to, 157
- devices.c file, 513
- digital I/O ports, 235-238
- direct memory access (see DMA)
- directly mapped I/O memory, 240
- directories of kernel headers, 17
- directory entry (file structure), 68
 - backward compatibility issues, 93
- disable_dma(), 417, 424
- disable_irq_nosync(), 267, 290
 - backward compatibility issues, 289
- disable_irq(), 267, 290
 - backward compatibility issues, 289
 - shared handlers and, 276
- disabling interrupts, 267
 - using save_flags/restore_flags, 252
- disassemble command (gdb), 121
- disassembled code and ksymoops, 116
- disk changes, 352-354
- disk files vs. open files, 67
- disk geometry, retrieving, 351
- dma_addr_t type, 406
- DMA (direct memory access), 401-418
 - allocating buffers for, 402-404
 - backward compatibility issues, 420
 - configuring controller, 415-418
 - dedicated buffers at boot time, 221
 - __get_dma_pages() and, 215, 223
 - __GFP_DMA flag and, 209
 - for ISA memory, 413-418
 - PCI devices and, 404-412
 - dealing with difficult hardware, 405
 - DMA mappings (see DMA mappings)
 - hardware dependencies for, 411
 - simple example of, 411
 - registering usage, 414
 - ring buffers, 402
- DMA mappings, 405-410
 - consistent, 406
 - setting up, 407
 - scatter-gather, 409
 - streaming, 406
 - setting up, 407-409
- dma_spin_lock, 416
- DMAC (DMA controller), 413
- DMA-capable memory zone, 210
 - SLAB_CACHE_DMA flag and, 212
- dma.h header file, 414, 416, 423
- dmesg command, 115
- do_basic_setup(), 508
- do_gettimeofday(), 185, 206
- do_initcalls(), 508
- do_ioctl method, 441, 458
- do_IRQ(), 263
- do_map_pgoff(), 514
- do_timer(), 193
 - BH mechanism and, 272
- down_interruptible(), 77, 95
- down(), 77
- dquot.c file, 513
- driver modules, 7
- drivers
 - adding new, 56-61
 - asynchronous notification and, 161
 - character (see char drivers)
 - choosing ioctl numbers for, 130
 - command-oriented, 140
 - configuring, 42-44

- drivers (continued)
 - device names (see devices, names of)
 - file operations, 63-66
 - FireWire, 8
 - I2O, 8
 - input/output buffers and, 148
 - interrupt-driven, 362-364
 - mechanism of (see mechanism, driver)
 - monitoring with preprocessor, 101-103
 - network drivers, 425-469
 - probing for IRQ numbers, 261
 - removing (see unloading modules)
 - SCSI, 7
 - security issues, 9
 - USB (see USB drivers)
 - user-space, 45
 - version (see version numbering)
 - writing, using devfs, 85-91
- drivers/block directory, 518
- drivers/cdrom directory, 520
- drivers/char directory, 518
- drivers/i2c directory, 524
- drivers/ide directory, 519
- drivers/input directory, 523
- drivers/md directory, 519
- drivers/media directory, 523
- drivers/mtd directory, 524
- drivers/net directory, 521
- driver-specific symbols, 328-330
- drivers/scsi directory, 520
- drivers/sound directory, 521
- drivers/video directory, 522
- dump analyzers, 125
- Dynamic Probes debugging tool, 127

- E**
- EBUSY error, 168
- edge-triggered vs. level-triggered interrupt lines, 274, 495
- EISA (Extended ISA) buses, 497
- elevator.o file, 519
- ELF sections
 - avoiding #ifdefs, 508
 - changes to kernel compilation, 509
- embedded systems, different ld scripts needed for , 510
- enable_dma(), 417, 424
- enable_irq(), 267, 290
 - backward compatibility issues, 289
 - shared handlers and, 276
- enabling interrupts, 267
 - using save_flags/restore_flags, 252
- end_request(), 330, 368
 - DEVICE_NO_RANDOM symbol and, 329
 - interrupt-driven block drivers and, 362
 - splitting up multibuffer requests, 339
- end_that_request_first(), 340, 368
- end_that_request_last(), 340, 368
- endless loops, preventing, 118
- end-of-file
 - poll method and, 156
 - seeking relative to, 163
- enet_statistics structure, Linux 2.0, 465
- entropy pool and SA_SAMPLE_RANDOM flag, 255
- errno.h header file, 31
- error codes, 31
- errors
 - handling in init_module(), 30-32
 - read/write, 80
 - strace command to debug, 110
- /etc/hosts file, 428
- /etc/modules.conf file, 307, 319
- /etc/networks file, 428
- /etc/syslog.conf file, 100
 - avoiding performance problems, 103
- ETH_ALEN macro, 444, 468
- eth_header method, 440
- ETH_P_IP macro, 457, 468
- eth_type_trans(), 469
 - overriding ARP, 456
- ether_setup(), 432, 468
 - setting up interface information, 436-439
- etherdevice.h header file, 468
- Ethernet, 429
 - address resolution, 455-458
 - ARP and, 455
 - non-Ethernet headers, 457
- ethernet subdirectory, 516
- exclusive sleep, 146
- exclusive waits, 146
- exec.c file, 513
- execution modes, 19
- execve(), 511

Index

- `__exit` attribute, 35
- exit system call, 512
- expansion board memory, 238-247
- experimental kernels, 11
- expires field (timer_list structure), 201
- EXPORT_NO_SYMBOLS macro, 29, 51
 - in Linux 2.0, 48
- EXPORT_SYMBOL macro, 30, 50-51
- EXPORT_SYMBOL_NOVERS macro, 29, 51
- EXPORT_SYMTAB macro, 29, 51
- exporting symbols, 29, 317
 - in Linux 2.0, 48-50
- Extended ISA (EISA) buses, 497
- external buses, 499-502
 - directories for, 524
- F**
- `f_dentry` pointer, 68
 - backward compatibility issues, 93
- `f_flags` field (file structure), 67
 - `O_NONBLOCK` flag, 134, 148
- `f_mode` field (file structure), 67
- `f_op` pointer, 68
- `f_pos` field (file structure), 67, 91
 - `read_proc/get_info()` and, 105
- `F_SETFL` command, 134, 161
 - `fcntl` system call and, 159
- `F_SETOWN` command, 161
 - `fcntl` system call and, 159
- facilities, (un)registering in
 - `init_module()`, 29-32
- fast interrupt handlers, 262-264
 - backward compatibility issues, 288
- `fasync_helper()`, 162, 179
- `fasync` method, 65
 - asynchronous notification and, 161
 - backward compatibility issues, 173
- `fasync_struct` structure, 161
- faults (see system faults)
- `faulty_write()`
 - `klogd` and, 113
 - `ksymoops` and, 115
- `fb_info` structure, 522
- `fbmem.c` file, 522
- `fc_setup()`, 437
- `fcntl` system call
 - `F_SETOWN/F_SETFL` commands, 159
 - vs. `ioctl` method, 134
- `fcntl.h` header file, 148
- `fdatasync` system call, 158
- FDDI networks, configuring interfaces, 437
- `fddi_setup()`, 437
- `fdisk` program, 355-362
- fiber channel devices, initializing, 437
- FIFO (first-in-first-out) devices, 55
 - `poll` method and, 156
- `fifo.c` file, 513
- file flags, 67
- file handling and `fs` directory, 513
- file modes, 67
- `file_operations` structure, 57, 63-66, 68
 - backward compatibility issues, 91-93
 - declaring using tagged initialization, 66
 - `mmap` method and, 384
- file structure, 63, 66
- File System header (`fs.h`), 95
- `file.c` file, 513
- `filemap.c` file, 514
- filesystem modules, 8
- filesystem nodes, 4
 - block drivers accessed by, 7
 - names, device (see devices, names of)
- `filp` pointer, 67
 - in `ioctl` method, 129
 - mounting block drivers, 348
 - in read/write methods, 78
 - retrieving inode pointers from, 93
- `filp->f_op`, 68
 - implementing multiple fops, 70
 - initializing, 89
- `filp->private_data`
 - initializing, 89
- `FIOASYNC` command, 134
- `FIOCLEX` command, 134
- `FIONBIO` command, 134
- `FIONCLEX` command, 134
- FireWire drivers, 8
- firmware, PCI-aware, 474
- first-in-first-out (FIFO) devices, 55
 - `poll` method and, 156
- flags
 - `devfs`, 87

- flags (continued)
 - file, 67
 - flash memory, executing kernel from, 510
 - flush method, 65
 - backward compatibility issues, 93
 - close system call and, 73
 - flushing pending output, 158
 - font, incorrect on console, 140
 - fops pointers, 63
 - as argument to register_chrdev, 56
 - implementing multiple, 70
 - fops->open, 70
 - forcing module load, 24
 - fork system call, 512
 - fragmentation, 403
 - frame buffer video devices directory, 522
 - free command, 85
 - free_dma(), 414, 423
 - free_irq(), 253, 289
 - when to call, 255
 - free_kiovec(), 396, 422
 - free_pages(), 215, 225
 - free_page(), 215, 225
 - fs directory, 513
 - fs.h header file, 95, 177, 322, 366
 - asynchronous notification and, 161
 - block driver commands and, 349
 - blocking/nonblocking operations, 148
 - file structure and, 66
 - kdev_t type and, 62
 - register_chrdev(), 56
 - fsync_dev method, 328
 - flushing all partitions, 359
 - fsync method, 65, 158
 - backward compatibility issue, 173
 - functions
 - accessing memory in Linux 2.0, 173-175
 - calling from modules/applications, 17
 - disassembling with gdb, 121
 - inserting schedule() calls in, 118
- G**
- gcc compiler
 - g option, 121
 - inline assembly code, 184
 - O flag, 22
 - SPARC platforms and, 27
 - Wall flag, 23
 - gdb debugger, 120-122
 - kgdb patch and, 125
 - gendisk_head, 358, 369
 - gendisk_struct structure, 356, 369
 - adding to global list, 358
 - removing from global list, 360
 - General Public License (GPL), 12
 - generic hard disk support, 356
 - genhd.c file, 518
 - genhd.h header file, 356, 369
 - __GENKSYMS__, 320
 - genksyms program, 317
 - geographical addressing, 473
 - lack of in ISA devices, 494
 - MCA buses and, 497
 - NuBus and, 499
 - Plug and Play, 496
 - SBus and, 498
 - geometry, disk, 351
 - get_cycles(), 183
 - __get_dma_pages(), 215, 225
 - get_dma_residue(), 417, 424
 - get_fast_time(), 185, 206
 - __get_free_page(), 215, 225
 - advantage of using, 217
 - __get_free_pages(), 215, 225
 - get_free_pages(), 208
 - allocating memory using, 298
 - limitations on memory allocation, 403
 - mmap method and, 392
 - returning virtual addresses, 217
 - get_info(), 104-107
 - get_kernel_syms system call, 24
 - get_page(), 387
 - backward compatibility issues, 419
 - get_stats method, 441, 459
 - get_unaligned(), 299, 304
 - __get_user(), 136, 178
 - get_user(), 136, 178
 - Linux 2.0 version, 174
 - get_zeroed_page(), 215, 225
 - getdents system call, 513
 - GFP_ATOMIC flag, 209, 224
 - page-oriented allocation functions, 215
 - preparing for allocation failure, 215
 - GFP_BUFFER flag, 209

Index

- __GFP_DMA flag, 209, 224
 - memory zones and, 210
 - page-oriented allocation functions, 215
- __GFP_HIGHMEM flag, 210, 224
 - memory zones and, 210
 - page-oriented allocation functions, 215
- GFP_HIGHUSER flag, 209
- GFP_KERNEL flag, 36, 208, 224
 - page-oriented allocation functions, 215
- GFP_USER flag, 36, 209
- GKSMP symbol, 317
- global
 - memory areas, 55
 - message enabling/disabling, 101
- goto statement, 30
- GPL (General Public License), 12
- gpm mouse server, 45, 119
- group, device, 59

H

- handle_IRQ_event(), 263
- handle_scancode(), 518
- hard_header method, 440, 457
 - backward compatibility issues, 466
 - building packets with ARP query results, 455
- hard_header_parse method, 442
- hard_start_transmit method, 445
- hard_start_xmit method, 440, 445
 - backward compatibility issues, 464
- HARDRESET command, 132
- hardsect_size global array, 324, 367
- hardware (see devices)
- hardware abstractions (PCI), 493
- hardware addresses, 437
 - assigning, 444
 - changing, using set_mac_address method, 441
 - multicasting and, 460-464
 - used with PCI peripherals, 471-474
- hardware headers
 - adding before transmitting packets, 454
 - backward compatibility issues, 466
 - building, 440
 - encapsulating information, 457
 - overriding ARP, 456

- hardware memory barriers, 228, 249
 - backward compatibility issues, 248
- HAVE_DEVLIST, backward compatibility issues for, 466
- HDIO_GETGEO command, 351
- hdreg.h header file, 351
- head pointers and circular buffers, 280
- header_cache method, 442
- header_cache_update method, 442
- header files, 17
 - include directory and, 517
 - managing symbol visibility, 29
 - removing conditional compilation, 90
- headers, Ethernet (see Ethernet)
- headers, non-Ethernet, 457
- helper programs, running, 311
- hex values of oops messages, 114
- hiding global symbols, 29
 - in Linux 2.0, 48
- high memory, 372
 - request queues and, 348
- high memory zone, 210
- high RAM addresses, reserving, 223
- highmem.c file, 515
- highmem.h header file, 374
- HIPPI drivers, preparing fields for, 437
- hippi_setup(), 437
- host adapters, plugging into core system, 520
- host numbers, 428
- hosts.c file, 520
- hot-pluggable devices, handling, 489-493
- hung system, 118
- HZ (time frequency) symbol, 181, 297

I

- i_rdev field (inode structure), 61
- I2O drivers, 8
- IA-64 architecture
 - PCI DMA interface support, 411
 - porting and, 233
 - /proc/interrupts file, snapshot of, 257
- IDE device drivers, directory for, 519
- if_ether.h header file, 468
- ifconfig command
 - net_device structure and, 435

- ifconfig command (continued)
 - opening/closing interfaces, 443
- `#ifdef` constructs
 - avoiding with `devfs`, 90
 - avoiding with `init` calls, 508
- `IFF_` symbols, 438, 462
- `IFF_NOARP` flag, 432
- `if.h` header file, 438, 458, 467
- `ifreq` structure, 458
- IKD (integrated kernel debugger)
 - patch, 124
- `IMMEDIATE_BH` bottom half, 272
 - writing a BH bottom half, 273
- immediate queue, 193, 197, 206
 - BH mechanism and, 272
 - writing a BH bottom half, 273
- `in_interrupt()`, 192, 206
 - vs. `intr_count` global variable, 205
- `inb_p()`, 232, 249
- `inb()`, 230, 249
- `include/asm` directory (see entries under `<asm/>`)
- `include` directory, 517
- infinite loops, preventing, 118
- `inflate.c` file, 517
- `__init` attribute, 35
- `init` calls and `#ifdef` constructs, 508
- `INIT_LIST_HEAD` macro, 301
- `init_module()`, 16, 29-32, 50
 - error handling in, 30-32
 - `EXPORT_NO_SYMBOLS` macro and, 29
 - hiding global symbols, 48
 - unregistering facilities from, 30
 - using unique names instead of, 34
- `init` process, 511
- `INIT_REQUEST()`, 330, 368
 - splitting up multibuffer requests, 339
- `init` scripts and loading/unloading
 - modules, 60
- `init` thread, 507
- `init_timer()`, 201, 207
- `__initdata` attribute, 35
- `init.h` header file, 35, 50
- initialization functions and boot-time
 - memory allocation, 221
- initializing
 - kernel data structures, 507
 - modules, 29-32
 - explicitly naming functions for, 34
 - network devices, 432
 - semaphores, 76
- `initrd` utility, 360
- inline assembly code (example), 183
- inline functions, 22
 - for accessing I/O ports, 230
- `inl()`, 231, 249
- inode pointer
 - backward compatibility issues, 91
 - in `ioctl` method, 129
 - retrieving from `filp` pointer, 93
- inode structure
 - accessing device numbers, 61, 69, 95
 - mounting block drivers, 348
- `inode->i_rdev`, 61, 69, 95
- `inode.c` file, 513
- input buffers, driver, 148
- input files, enabling asynchronous
 - notification from, 159
- input management, directory for, 523
- input module, 28
- input pins, 226, 235
 - reading values from parallel port, 238
- `input_register_device()`, 523
- `input_register_handler()`, 523
- `input.c` file, 523
- `input.h` header file, 504
- `insb()`, 232, 249
- `insl()`, 232, 249
- `insmod` program, 6, 24
 - assigning parameter values, 42
 - backward compatibility issues, 319
 - dynamically allocating major numbers, 60
 - `-f` switch, 24
 - `modprobe` program vs., 28
 - module loading and security, 309
 - testing modules using, 16
 - version control in modules, 314
 - `vmalloc()` and, 218
- installing interrupt handlers, 253-264
- `insw()`, 232, 249
- `int` data type, 294
- integrated kernel debugger (IKD)
 - patch, 124
- `inter_module_get_request()`, 313, 319

Index

- `inter_module_get()`, 312, 319
- `inter_module_put()`, 313, 319
- `inter_module_register()`, 312, 319
- `inter_module_unregister()`, 312, 319
- interactive kernel debugger (kdb), 122-124
- interface buses, 496-502
- interface flags for `net_device` structure, 438
- interface-specific data types, 296
- intermodule communication, 311-314
- Internet sites about Linux kernels, xv
- interrupt handlers, 251-292
 - using arguments with, 267
 - autodetecting IRQ numbers, 258-262, 276
 - backward compatibility issues, 288
 - BH mechanism, 271
 - bottom halves of handlers, 269-274
 - enabling/disabling interrupts, 252, 267
 - fast vs.slow, 262-264
 - backward compatibility issues, 288
 - implementing, 264-268
 - installing, 253-264
 - at device open, 255
 - shared handlers, 275
 - for network drivers, 450
 - preparing parallel ports for, 253
 - `/proc` files for, 256
 - race conditions, 278-288
 - circular buffers for, 279
 - lock variables for, 284-286
 - spinlocks for, 281-283
 - running shared handlers, 276
 - sharing interrupts, 274-278
 - tasklets, 270
 - on x86 architecture, 263
- interrupt mode and asynchronous execution, 191
- interrupt numbers, 254
 - used as arguments, 267
 - probing using kernel facility, 259
- interrupt request lines (see IRQs)
- Interrupt Service Routine (ISR), 181
- interrupt-driven operation, 278
 - block drivers, 362-364
- `interrupt.h` header file, 199, 206, 259, 272, 290
- `interruptible_sleep_on_timeout()`, 142, 178
 - delaying code execution, 187
- `interruptible_sleep_on()`, 142, 178
 - avoiding race conditions, 286
 - implementation of, 144
 - vs. `wait_event` macro, 145
- interrupts, code, 77
- interrupts
 - PCI, 488
 - timer, 181
- `interrupts` file, 256, 289
 - shared interrupts and, 277
- intervals of time, 181-184, 297
- `intptr_t` type (C99 standard), 294
- `intr_count` global variable, 205
- `inw()`, 230, 249
- `_IO()` macro, 131, 177
- I/O, 158
 - accessing, PCI and, 483-488
 - asynchronous notification, 159-162
 - blocking, 141-153
 - blocking/nonblocking, 148
 - buffers for, 148
 - flushing pending, 158
 - interrupt-driven, 278
 - ISA devices and, 494
 - pausing, 232
 - remapping specific regions of, 389
 - space for, in PCI buses, 473
 - string operations, 231
 - transferring data with DMA, 401-418 (see also reading; writing)
- I/O memory, 39-41, 226, 238-247
 - directly mapped, 240
 - page tables and, 239
 - software-mapped, 242
- I/O ports, 36-41, 226, 229-234
 - allocating, 39
 - digital, 235-238
 - inline functions for accessing, 230
 - parallel (see parallel ports)
- I/O registers vs. RAM, 227-229
- I/O registry, accessing, 38
- `io_request_lock`, 338, 368
 - backward compatibility issues, 366
 - multiqueue block drivers and, 343
 - performing clustered I/O, 341
- I/O request queues (see request queues)
- `iobuf.h` header file, 396, 422

- `_IOC()` macro, 177
- `_IOC_TYPEBITS` macro, 131, 177
- `_IOC_NR()` macro, 131, 177
- `_IOC_READ` macro, 131, 177
- `_IOC_NONE` macro, 131, 177
- `_IOC_DIRBITS` macro, 177
- `_IOC_TYPE()` macro, 131, 177
- `_IOC_NRBITS` macro, 131, 177
- `_IOC_SIZEBITS` macro, 131, 177
- `_IOC_WRITE` macro, 131, 177
- `_IOC_SIZE()` macro, 131, 177
- `_IOC_DIR()` macro, 131, 177
- `ioctl` method, 64, 129-141
 - accessing specific information for partitions, 361
 - using bitfields to define commands, 130
 - block devices and, 349-352
 - changing `read_ahead` values, 326
 - command numbers, choosing, 130-133
 - controlling devices without, 140
 - controlling I/O channel, 128
 - customizing for networking, 458
 - debugging with, 108
 - extra argument of, 134-139
 - implementing `ioctl` commands, 138
 - network devices and, 441
 - predefined commands of, 133
 - using scalar values to define commands, 133
 - TIOCLINUX command, 99
 - type checking disabled, 129
- `ioctl.c` file, 513
- `ioctl.h` header file, 130, 177
 - setting up command numbers, 131
- `ioctl-number.txt` file, 130
- `io.h` header file (asm), 249, 422
 - accessing I/O ports, 230
 - converting between bus/virtual addresses, 404
- `io.h` header file (linux), 250
- `iomem` file, 39, 53
- `iomem_resource` structure, 41
- `ioperm()`, 231
- `iopl()`, 231
- `ioport_resource` structure, 41
- `ioport.h` header file, 38, 52, 229, 250
 - resource ranges and, 40
- `ioports` file, 37, 53
- `_IOR()` macro, 131, 177
- `_IOW()` macro, 131, 177
- `_IOWR()` macro, 131, 177
- `ioremap_nocache()`, 242, 250
- `ioremap()`, 217-219, 225, 250
 - accessing I/O memory, 239
 - backward compatibility issues, 248
 - ISA memory range, 243
 - software-mapped I/O memory and, 242
- `IORESOURCE_IO` flag, 484
- `IORESOURCE_MEM` flag, 484
- `IORESOURCE_PREFETCH` flag, 484
- `IORESOURCE_READONLY` flag, 484
- `ionmap()`, 217, 225, 250
 - backward compatibility issues, 248
 - software-mapped I/O memory and, 242
- `iovec` structures, 84
- IP numbers
 - assigning, 427-429
 - resolving to physical addresses, 455-458
- `ip_summed` field (`sk_buff`), 449, 453
- `ipc` directory, 517
- `ipv4/ipv6` subdirectories, 516
- `irq` argument (interrupt number), 254, 267
- `IRQ_WAITING` status bit, setting, 264
- `irq.h` header file, 262, 267
- IRQs (interrupt request lines), 253
 - autodetecting (probing) numbers for, 258-262
 - shared interrupts and, 276
 - level-triggered vs. edge-triggered, 274, 495
 - PCI devices and, 488
 - statistics on, 257
- ISA bus master DMA, 413
- ISA devices, 494-496
 - DMA for, 413-418
 - EISA (Extended ISA) buses, 497
 - identifying I/O regions, 36
 - interrupt sharing and, 274, 495
 - pausing I/O, 232
 - Plug-and-Play specification, 496
 - probing, 38
 - programming techniques, 495
 - VLB (VESA Local Bus) devices, 498
- ISA memory
 - accessing, 244

Index

ISA memory (continued)
 below 1 MB, 243-245
 DMA for, 413-418
 nopage method and, 389
 probing for, 245-247
isa_readb and related functions, 245
ISDN drivers and lookaside caches, 211-214
ISR (Interrupt Service Routine), 181

J

jiffies value
 in busy waiting implementation, 186
 kernel timers and, 201
 no solution for short delays, 188
 retrieving current time, 184
 at timer interrupt, 182
 trans_start field and, 442
 variable syntax, 205
jq (Just In Queue) module, 193
 timer usage example, 202
jq_print_tq(), 193
jit (Just In Time) module
 current time, retrieving, 185
 delaying code execution, 186
jitbusy program, 186
Just In Queue (jq) module, 193
 timer usage example, 202
Just In Time (jit) module
 current time, retrieving, 185
 delaying code execution, 186

K

kbd_mode -a command, 119
kcore file, 120
kdataalign program, 300
kdatasize module, 294
kdb kernel debugger, 122-124
kdev_t_no_nr(), 62
kdev_t type, 62
 extracting physical device number, 329
kdev_t.h header file, 62
keep directive (modprobe), 308
KERN_ALERT macro, 98
KERN_CRIT macro, 98
KERN_DEBUG macro, 98
KERN_EMERG macro, 98
KERN_ERR macro, 98

KERN_INFO macro, 98
KERN_NOTICE macro, 98
KERN_WARNING macro, 98
kernel directory, 512
kernel headers, 17
kernel I/O buffers, 396-400
kernel I/O vectors, 396
kernel lockups, detecting, 124
kernel logical addresses (see logical addresses)
kernel sources, 527
kernel space, 19
 transferring to/from user space, 78-84
kernel stack debugger (IKD feature), 124
__KERNEL__ symbol, 22, 50
 explicitly sizing data, 295
 kernel header files and, 17
__KERNEL_SYSCALLS__, 511
kernel timers, 200-203
KERNEL_VERSION macro, 25, 47
kernel_version variable, 52
kernel virtual addresses (see virtual addresses)
kernelnd program, backward compatibility issues for, 318
kernelnd.h header file, backward compatibility issues for, 319
KERNELDIR variable and version dependency, 25
kernel.h header file, 98, 228, 249
kernels
 allocating memory at boot time, 221-223
 books about Linux, 527
 booting, 507-509
 with initrd, 360
 capabilities and restricted operations, 137
 concurrency in, 20
 connecting network drivers to, 430-434
 current process and, 21
 developmental (experimental), 11
 filesystem modules, 8
 flash memory, executing from, 510
 handling system faults (see system faults)
 IKD (integrated kernel debugger)
 patch, 124
 initial boot sequence, 507
 introduction to, 1-14

- kernels (continued)
 - kgdb patch and, 125
 - kiobufs, 396-400
 - kmod facility and, 305
 - linked lists, 300-302
 - loading modules into (see loading modules)
 - messages (see messages)
 - module version control, 314-318
 - multicasting support, 461
 - probing interrupt numbers with, 259
 - race conditions and, 76-78
 - request queues, finding, 343
 - running task queues, 191
 - security (see security)
 - splitting role of, 4-6
 - symbol table, 27-29
 - klogd and, 114
 - system hangs, 118
 - time intervals in, 181-184
 - tracing programs, 108-110
 - using conventional data types, 295
 - version numbering, 10
 - web sites about, xv
 - keventd process, 192, 195
 - backward compatibility issues, 204
 - call_usermodehelper and, 311
 - keyboard, debugging when locked, 118
 - keyboard.c file, 518
 - kfree_skb(), 454, 468
 - kfree(), 36, 52, 224
 - defined in slab.c file, 514
 - scull driver example and, 73
 - kgcc package, 22
 - kgdb patch, 125
 - khttpd subdirectory, 516
 - kill_fasync(), 162, 179
 - kiobuf_init(), 396, 422
 - kiobufs, 396-400
 - kiovecs, 396
 - klogd daemon
 - c flag, 98
 - debugging modules with, 113
 - decoding oops messages, 113
 - f option, 101
 - logging messages, 100
 - obtaining clean oops messages, 115
 - p option, 114
 - kmalloc.c file, 211, 514
 - kmalloc(), 36, 52, 208-211, 224
 - defined in slab.c file, 514
 - flags argument, 208-210
 - limitations on memory allocation, 403
 - performance degradation issues, 216
 - returning virtual addresses, 217-219
 - scull driver example and, 73
 - size argument, 211
 - vs. vmalloc(), 217-219
 - kmap(), 374, 421
 - backward compatibility issues, 420
 - kmem_cache_alloc, 213, 225
 - kmem_cache_create, 212, 224
 - kmem_cache_destroy, 213, 224
 - kmem_cache_free, 213, 225
 - kmem_cache_t, 212, 224
 - kmod facility, 305
 - loading modules, 310
 - user-mode helper programs and, 311
 - kmod.c file, 512
 - kmod.h header file, 306, 319
 - backward compatibility issues, 319
 - kmsg file, 100
 - kswapd thread, 515
 - ksymoops utility, 114-118
 - obtaining clean oops messages, 115
 - ksyms command, 27
 - ksyms file, 27, 30, 53
 - ksymoops and, 115
 - kunmap(), 374, 421
 - backward compatibility issues, 420
- ## L
- layered modularization, 28
 - LCRASH utility, 126
 - ld scripts and boot code layout, 510
 - ld -r command, 23
 - __le32_to_cpu macro, 298, 304
 - least significant bit and partitionable devices, 355
 - LEDs, soldering to output pins, 237
 - levels
 - debugging, 102
 - message priority (see loglevels)
 - levels (modalities), CPU, 19

Index

- level-triggered vs. edge-triggered interrupt lines, 274, 495
- lib directory, 517
- libraries, 17
- license, Linux, 12
- line disciplines, implementing, 521
- link state, changes in, 451
- linked lists, 300-302
- Linux
 - license terms, 12
 - version numbering, 10
- linux directory, 17
- Linux Documentation Project web site, xv
- Linux Kernel Crash Dumps (LKCD), 126
- Linux Trace Toolkit (LTT), 127
- LINUX_VERSION_CODE macro, 25, 52
- <linux/autoconf.h> header file, 316
- <linux/blk.h> header file (see blk.h header file)
- <linux/blkdev.h> header file, 323, 366
- <linux/blkpg.h> header file, 351
- <linux/bootmem.h> header file, 221, 225
- <linux/capability.h> header file, 137, 178
- <linux/config.h> header file, 316, 320, 477, 503
- <linux/delay.h> header file, 188, 206
- <linux/devfs_fs_kernel.h> header file, 87, 96
- <linux/errno.h> header file, 31
- <linux/etherdevice.h> header file, 468
- <linux/fcntl.h> header file, 148
- <linux/fs.h> header file, 95, 177, 322, 366
 - asynchronous notification and, 161
 - block driver commands and, 349
 - blocking/nonblocking operations, 148
 - file structure and, 66
 - kdev_t type and, 62
 - register_chrdev(), 56
- <linux/genhd.h> header file, 356, 369
- <linux/hdreg.h> header file, 351
- <linux/highmem.h> header file, 374
- <linux/if_ether.h> header file, 468
- <linux/if.h> header file, 438, 458, 467
- <linux/init.h> header file, 35, 50
- <linux/input.h> header file, 504
- <linux/interrupt.h> header file, 199, 206, 259, 272, 290
- <linux/io.h> header file, 250
- <linux/iobuf.h.h> header file, 396, 422
- <linux/ioctl.h> header file, 177
 - setting up command numbers, 131
- <linux/ioport.h> header file, 38, 52, 229, 250
 - resource ranges and, 40
- <linux/kdev_t.h> header file, 62
- linux-kernel mailing list, 13
- <linux/kernel.h> header file, 98, 228, 249
- <linux/kerneld.h> header file
 - backward compatibility issues, 319
- <linux/kmod.h> header file, 306, 319
 - backward compatibility issues, 319
- <linux/list.h> header file, 144, 300-302, 304
- <linux/malloc.h> header file, 224
- <linux/mm.h> header file, 209, 224, 380, 421
- <linux/module.h> header file, 24, 51, 66
 - version.h header file and, 25
- <linux/modversions.h> header, 315, 320
- <linux/netdevice.h> header file, 431, 467
- <linux/param.h> header file, 181, 205
- <linux/pci.h> header file, 405, 422, 477, 503
 - accessing configuration space, 480
 - detecting size of PCI regions, 486
 - pci_ops structure and, 493
- <linux/poll.h> header file, 154, 179
- <linux/proc_fs.h> header file, 104
- <linux/scatterlist.h> header file, 410
- <linux/sched.h> header file, 52, 178, 205, 289, 291
 - interrupt request line functions, 253
 - jiffies value and, 182
 - kernel directory and, 512
 - wait queue code information, 147
- <linux/skbuff.h> header file, 445, 452, 468
- <linux/sockios.h> header file, 458, 469
- <linux/spinlock.h> header file, 166, 180, 281, 290
- <linux/symtab_begin.h> header file, 51
- <linux/symtab_end.h> header file, 51
- <linux/time.h> header file, 206
- <linux/timer.h> header file, 201, 207
- <linux/tqueue.h> header file, 190, 192, 206
- <linux/types.h> header file, 295, 303
- <linux/uio.h> header file, 84

- <linux/usb.h> header file, 504
 - <linux/version.h> header file, 25, 52
 - <linux/vmalloc.h> header file, 217, 225
 - <linux/wait.h> header file, 144, 178
 - list_add(), 301, 304
 - list_add_tail(), 301, 304
 - list_del(), 301, 304
 - list_empty(), 301, 304
 - testing request queues with, 344
 - list_entry(), 301, 304
 - list_head data structure, 300-302
 - list_splice(), 301, 304
 - list.h header file, 144, 300-302, 304
 - lists, linked, 300-302
 - __LITTLE_ENDIAN symbol, 298, 304
 - little-endian byte order, 298, 475, 480
 - LKCD (Linux Kernel Crash Dumps), 126
 - ll_rw_blk.c file, 518
 - llseek method, 64, 92, 163
 - in Linux version 2.0, 176
 - loading block drivers, 321-354
 - loading modules, 24
 - on demand, 305-311
 - slave/master modules example, 309
 - dynamically assigned device numbers, 59
 - for network drivers, 430
 - version dependency and, 24
 - LocalTalk devices, setting up fields for, 437
 - lock_kiovec(), 396, 422
 - lock method, 65
 - lock variables, 284-286
 - locked keyboard, debugging, 118
 - lockup detector (IKD), 124
 - loff_t (long offset), 64, 67, 91
 - LOG_BUF_LEN circular buffer, 100
 - logging messages, 100
 - logical addresses, 372
 - loglevels (message priorities), 15, 97-99
 - long data type, 294
 - long delays, 186-188
 - lookaside caches, 211-214
 - backward compatibility issues, 223
 - loopback interface, 426
 - IFF_LOOPBACK flag, 438
 - loop.c file, 519
 - loops
 - busy, 186
 - endless, 118
 - software, 188
 - loops_per_second value, 188
 - low memory, 372
 - lp.c file, 518
 - ls command, identifying device type, 55
 - lseek method, 64
 - in Linux version 2.0, 176
 - syntax in Linux 2.0, 92
 - ltalk_setup(), 437
 - LTT (Linux Trace Toolkit), 127
 - LVM (logical volume manager) drivers
 - drivers/md directory, 519
 - “make request” function and, 346
- ## M
- M68k architecture
 - layout of boot code, 510
 - no support for PCI bus, 411
 - porting and, 233
 - MAC (Medium Access Control)
 - addresses, 437
 - resolving, 455-458
 - set_mac_address method and, 441
 - machine-specific registers, 183
 - magic SysRq key, 119
 - mailing list, linux-kernel, 13
 - major device numbers, 56-61
 - dynamic allocation of, 57-61
 - MAJOR macro, 62, 95
 - major_name value (gendisk_struct), 356
 - MAJOR_NR symbol, 328, 367
 - “make request” function, 346-348
 - __make_request(), 346
 - make utility
 - building a makefile, 23
 - KERNELDIR variable and, 25
 - makefiles, 22
 - adding version control with, 315
 - exporting versioned symbols, 317
 - install rules for, 26
 - SPARC architecture and, 27
 - malloc.h header file, 224
 - mangling symbol names, 314-317
 - map_user_kiobuf(), 399, 422
 - maplist array (kiobuf), 396, 400
 - mapper program, 391

Index

- mapping memory (see memory management)
- mapping registers, 405
 - architectures not supporting, 411
 - scatterlists and, 409
- mark_bh(), 272, 290
- marking bottom halves, 272
- max_readahead global array, 325, 367
 - backward compatibility issues, 365
- max_sectors global array, 326, 367
- max_segments global array, 326
- mb(), 228, 249
- MCA (Micro Channel Architecture)
 - buses, 497
- mdelay(), 188, 206
- mechanism, driver
 - defining, 54
 - policy versus, 2
- media, directory for, 523
- Medium Access Control addresses (see MAC addresses)
- mem.c file, 518
- memcpy_fromfs(), 94, 96
- memcpy_fromio(), 241, 250
- memcpy_tofs(), 94, 96
- memcpy_toio(), 241, 250
- memory
 - accessing
 - from expansion boards, 238-247
 - in Linux 2.0, 173-175
 - in PCI buses, 473, 483-488
 - allocating, 73-75
 - at boot time, 221-223
 - with kmalloc, 208-211
 - by page, 214-217
 - performance degradation issues, 216
 - with vmalloc, 217-220
 - circular buffers, 279
 - free, information on, 85
 - global areas, 55
 - high, 372
 - how much to allocate, 211
 - ISA memory range, 243-245
 - limitations on, 372
 - lookaside caches, 211-214
 - low, 372
 - managing allocation, 36
 - page size and portability, 297
 - persistence, 55
 - verifying user-space addresses, 135
 - vs. I/O registers, 227-229
- memory barriers, 228
 - backward compatibility issues, 248
 - performance issues, 229
- memory management, 4
 - accessing pages not in memory, 387-389
 - backward compatibility issues, 418-420
 - DMA (direct memory access), 401-418
 - fragmentation, 403
 - handling map region changes, 387-389
 - kernel source file directory, 514
 - memory mapping/remapping, 373-375
 - accessing pages not in memory, 387-389
 - handling region changes, 387-389
 - kiobufs, 396-400
 - mmap method, 382-395
 - PCI regions, 485
 - RAM, 390-394
 - specific I/O regions, 389
 - virtual addresses, 394
 - mmap method, 382-395
 - PCI and, 483-488
 - theory of, 370-382
 - VMA (virtual memory areas), 378-382
- memory map arrays, 374
- memory maps, components of, 379
- memory zones, 210
- memory.c file, 515
- memory-is-prefetchable bit, 483
- memory-mapped registers (see I/O memory)
- memset_io(), 241, 250
- messages
 - globally enabling/disabling, 101
 - logging, 100
 - oops messages, 111-118
 - priorities (loglevels) of, 15, 97-99
- mice, 119
 - asynchronous notification, 161
- Micro Channel Architecture (MCA)
 - buses, 497
- minor device numbers, 56, 61, 69
- MINOR macro, 62, 95
- minor_shift value (gendisk_struct), 356

- MIPS processor
 - directly mapped memory, 240
 - inline assembly code and, 183
 - layout of boot code, 510
 - PCI DMA interface support, 411
 - porting and, 233
- MIPS64 architecture, support for PCI DMA
 - interface, 411
- misc directory, 525
 - installing drivers in, 26
- misc-modules/export.c file, 49
- MKDEV macro, 62, 95
- mknod command, 57
- mlock system call, 46
- mlock.c file, 514
- mm directory, 514
- mmap_avl.c file, 515
- mmap method, 65, 382-395
 - using remap_page_range, 384-386
 - remapping virtual addresses with, 394
 - scullp driver and, 391-394
 - usage count and, 386
 - vm_area_struct structure and, 380
- mmap.c file, 514
- mm.h header file, 209, 224, 380, 421
- mm/kmalloc.c file, 211, 514
- mm/slab.c file, 211, 514
- MOD_DEC_USE_COUNT macro, 33, 51
- MOD_IN_USE macro, 33, 51
- MOD_INC_USE_COUNT macro, 33, 51
- mod_timer(), 202, 207
 - avoiding race conditions, 203
- modalities (levels), CPU, 19
- modes
 - device modes, 59
 - file modes, 67
- modprobe program, 319
 - assigning parameter values, 42
 - directives, 308
 - insmod program vs., 28
 - loading modules, 307
 - request_module() and, 306
 - security issues for module names, 309
 - version control in modules, 314
- modularization
 - kmod facility, 305
 - layered, 28
 - network drivers, 434
- MODULE_AUTHOR macro, 44, 51
- MODULE_DESCRIPTION macro, 44, 51
- module_exit(), 35, 50
- module_init(), 35, 50
- __module_kernel_version symbol, 24
- module parameters, 43
 - backward compatibility issues, 50
- MODULE_PARM_DESC macro, 43, 51
- MODULE_PARM macro, 42, 51
 - backward compatibility issues, 50
- MODULE_SUPPORTED_DEVICE macro, 44, 51
- MODULE symbol, 22
- module.c file, 512
- module.h header file, 24, 51, 66
 - version.h header file and, 25
- modules, 6
 - applications vs., 16-21
 - classes of, 6-8
 - communicating between, 311-314
 - current process and, 21
 - debugging, 113-118
 - exporting symbols, 29, 317
 - in Linux 2.0, 48-50
 - filesystem, 8
 - header files of, 17
 - initializing, 29-32
 - explicitly naming functions for, 34
 - interrupts (see interrupt handlers)
 - license terms, 12
 - loading/unloading, 16, 61, 305-311
 - with dynamically assigned device numbers, 59
 - insmod program and, 24
 - for network drivers, 430, 434
 - slave/master modules example, 309
 - usage count and, 33, 313
 - using init scripts, 60
 - version dependency and, 24
 - (see also cleanup_module())
 - partition detection in, 357-360
 - platform dependency, 27
 - probing for hardware (see probing)
 - requesting the loading of, 306
 - security (see security)
 - stacking, 28
 - usage count, 33, 313

Index

- modules, usage count (continued)
 - backward compatibility issues, 93
 - version control, 314-318
 - version dependency, 24-26
 - modules file, 34, 51
 - ksymoos and, 114
 - modutils package
 - exporting symbols, 29
 - misc directory and, 26
 - MODVERSIONS, 320
 - modversions.h header file, 315, 320
 - monitoring, preprocessor for, 101-103
 - most significant bit, 253
 - partitionable devices and, 355
 - mounting block drivers, 348
 - mremap system call, 387
 - remapping specific I/O regions, 390
 - msr.h header file, 183, 205
 - MTU, network devices and, 441
 - multicasting, 460-464
 - IFF_MULTICAST flag and, 439
 - multiprocessor systems
 - backward compatibility issues, 48
 - multiqueue block drivers, 342-345
 - mutex semaphores, 76
 - mutual exclusion mode (semaphores), 76
- ## N
- n_tty.c file, 518
 - namei.c file, 513
 - names, device (see devices, names of)
 - namespace pollution, 18
 - native DMA, 413-418
 - natural alignment of data items, 300
 - nbd.c file, 519
 - nbtest program, 153
 - net_device_stats structure, 433
 - backward compatibility issues, 465
 - fields in, 459
 - net_device structure, 430, 435-443
 - device methods of, 440-442
 - ether_setup and, 432, 436-439
 - hidden fields, 436-443
 - interface flags for, 438
 - interface information, 436-439
 - unusual devices, assigning fields for, 437
 - utility fields for, 442
 - visible fields, 435
 - net directory, 516
 - net_init.c file, 436
 - netdevice.h header file, 431, 467
 - netif_carrier_off(), 451, 467
 - backward compatibility issues, 465
 - netif_carrier_ok(), 452, 467
 - netif_carrier_on(), 451, 467
 - backward compatibility issues, 465
 - netif_rx(), 467
 - packet reception and, 450
 - netif_start_queue(), 444, 467
 - backward compatibility issues, 464
 - netif_stop_queue(), 444, 467
 - backward compatibility issues, 464
 - controlling transmission concurrency, 446
 - netif_wake_queue(), 467
 - backward compatibility issues, 464
 - restarting packet transmission, 446
 - netsyms.c file, 516
 - network drivers, 7, 425-469
 - connecting to kernel, 430-434
 - initializing devices, 432, 521
 - interrupt handlers for, 450
 - link state, changes in, 451
 - loading/unloading modules for, 430, 434
 - methods of, 440-442
 - modularized vs. non-modularized, 434
 - opening/closing network
 - interface, 443-445
 - socket buffers (see socket buffers)
 - statistics on, 459
 - networking, 6
 - backward compatibility issues, 464-466
 - __NO_VERSION__ symbol, 26, 52
 - nonblocking operations, 148
 - poll method, 154-159
 - select method, 154-159
 - testing, 153
 - non-modularized network drivers, 434
 - nonpreemption and concurrency, 20
 - nopage method, 382
 - backward compatibility issues, 419
 - mapping memory with, 387-389
 - mapping RAM to user space, 391-394

nopage method (continued)
 mremap system call with, 387
 preventing extension of mapping, 390
 remapping virtual addresses with, 394
 normal memory zone, 210
 NR_IRQS symbol, 262
 NuBus, 499
 NULL pointers, invalid
 dereferencing, 111-113
 NUM macro, splitting minor numbers, 69
 numbering versions (see version numbering)

O
 O_NDELAY flag (f_flags field), 148
 O_NONBLOCK flag (f_flags field), 67, 134, 148
 read/write methods and, 157
 O_RDONLY flag (f_flags field), 67
 O_SYNC flag (f_flags field), 67
 objdump utility, 118
 disassembling module functions, 122
 octets vs. bytes, 426
 oops messages, 61, 111-118
 decoding, 113-118
 resolving hex values of, 114
 open method, 65, 68-72
 accessing data within partitions, 360
 adding VMA operations, 386
 blocking, 168
 checking for disk changes, 354
 cloning devices in response to, 169-171
 initializing file pointers, 89
 mounting block drivers, 348
 for network devices, 440, 443
 private_data and, 68
 requesting DMA channels, 414
 restricting simultaneous users and, 167
 for single-open devices, 165
 vm_operations_struct structure, 381
 open.c file, 513
 opening network interface, 443-445
 optimizations, compiler, 227
 options directive (modprobe), 308
 outb_p(), 232
 outb(), 230, 249
 outl(), 231, 249

output buffers, driver, 148
 output pins, 226, 235
 soldering LEDs to, 237
 outsb(), 232, 249
 outsl(), 232, 249
 outsw(), 232, 249
 outw(), 230, 249
 overriding ARP, 456

P

__pa(), 372, 421
 backward compatibility issues, 420
 packages, upgrading, 10
 PACKET_BROADCAST flag, 453
 PACKET_HOST flag, 453
 PACKET_MULTICAST flag, 453
 PACKET_OTHERHOST flag, 453
 packets
 multicasting, 460-464
 transmission/reception of, 429, 445-450
 page_address(), 374, 421
 page_alloc.c file, 514
 Page Directory (PGD) page table, 375
 page faults caused by invalid pointers, 111
 Page Mid-level Directory (PMD) page table, 375
 PAGE_SHIFT symbol, 297, 303
 page size and portability, 297
 PAGE_SIZE symbol, 297, 303
 mmap method and, 383
 page_table_lock, 378
 backward compatibility issues, 419
 remapping virtual addresses, 395
 page tables, 375-378
 building
 using nopage, 387-389
 using remap_page_range, 384
 I/O memory and, 239
 remapping virtual addresses, 394
 page.h header file, 297, 303, 372, 376
 page-oriented allocation functions, 214-217
 panic.c file, 512
 Parallel Line Internet Protocol (PLIP)
 using Ethernet headers, 456
 interrupt handling differences, 450
 overriding ARP, 457

Index

- parallel port driver modules, stacking, 28
- parallel ports, 235-238
 - disabling interrupts, 268
 - preparing for interrupt handling, 253
 - running shared interrupt handlers, 276
 - stacking driver modules, 28
- parameters
 - assigning values, 42
 - device, 43
 - module, 43
 - backward compatibility issues, 50
- param.h header file, 181, 205
- parport device driver, 518
- parse_options(), 507
- partial data transfers
 - read method, 80
 - write method, 82
- partitionable devices, 355-362
 - accessing data within partitions, 360
 - detecting partitions
 - with initrd, 360
 - in modules, 357-360
 - generic hard disk support for, 356
- path directive (modprobe), 308
- pausing I/O, 232
- PC parallel interface, 235-238
- PC/104 and PC/104+ bus architectures, 496
- pci_alloc_consistent(), 407, 422
- PCI_BASE_ADDRESS_ symbols, 483-486
- pci_bus structure, 494, 503
- pci_dev_driver(), 491
- pci_dev structure, 404, 477, 503
 - backward compatibility issues, 502
 - reading configuration variables, 481
- pci_device_id structure, 491, 503
 - ID fields for, 492
- PCI_DMA_BIDIRECTIONAL symbol, 408, 422
- PCI_DMA_FROMDEVICE symbol, 407, 422
 - bounce buffers and, 409
- PCI_DMA_NONE symbol, 408, 422
- pci_dma_supported(), 405, 422
- pci_dma_sync_sg(), 410, 423
- PCI_DMA_TODEVICE symbol, 407, 422
 - bounce buffers and, 409
- pci_driver structure, 491-493, 503
 - backward compatibility issues, 503
 - handling hot-pluggable devices, 490
- pci_enable_device(), 478
- pci_find_class(), 478, 504
- pci_find_device(), 478, 504
- pci_find_slot(), 478
- pci_free_consistent(), 407, 422
- pci_insert_device(), 491
- PCI_INTERRUPT_ symbols, 488
- pci_map_sg(), 410, 423
- pci_map_single(), 408, 423
- pci_module_init(), 490, 504
- pci_ops structure, 493
- PCI (Peripheral Component Interconnect)
 - addressing, 471-474
 - base address registers, 485-488
 - configuration registers, 475-479
 - configuration space, 473, 480-483
 - device configuration snapshot, 481
 - DMA and, 404-412
 - dealing with difficult hardware, 405
 - DMA mappings (see DMA mappings)
 - hardware dependencies for, 411
 - simple example of, 411
 - drivers, alternative to, 476
 - drivers/pci directory, 523
 - geographical addressing, 473
 - hardware abstractions, 493
 - hot-pluggable devices, 489-493
 - interface of, 470-494
 - interrupts, 488
 - I/O resources, 484
 - using ioremap(), 218
 - remap_page_range and, 389
- pci_present(), 477, 503
- pci_read_config_functions, 480, 504
- pci_register_driver(), 490, 504
- pci_remove_device(), 491
- pci_resource_end(), 484
- pci_resource_flags(), 484
- pci_resource_start(), 484
- pci_set_dma_mask(), 405
- pci_sync_single(), 409, 423
- pci_unmap_sg(), 410, 423
- pci_unmap_single(), 408, 423
- pci_unregister_driver(), 491, 504
- pci_write_config_functions, 481, 504
- pcibios.h header file, 502
- pcidata module, 482

- pcidump program, 482
- pci.h header file, 405, 422, 477, 503
 - accessing configuration space, 480
 - detecting size of PCI regions, 486
 - pci_ops structure and, 493
- pciregions module, 486
- PDEBUG/PDEBUGG symbols, 102
- pending output, flushing, 158
- performance
 - allocating socket buffers, 449
 - avoiding device collisions, 36
 - clustering requests and, 340
 - debugger use, 120
 - degrading by allocating too much memory, 216
 - managing system resources, 35-41
 - memory barriers and, 229
 - mmap method, 384
 - namespace pollution, 18
 - output buffers and, 148
 - PCI vs. ISA, 470
 - printk to debug, 103
 - raw I/O limitations to, 397
 - using request queues (see request queues)
 - string operations and, 231
- peripheral bus architecture (see bus architecture)
- Peripheral Component Interconnect (see PCI)
- peripheral memory, 238-247
- perorr() vs. strace command, 110
- persistence of memory, 55
- PG_locked flag, 374
- PG_reserved flag, 374
- pgd_offset(), 377
- PGD (Page Directory) page table, 375
- pgd_val(), 377
- pgtable.h header file, 218, 377
- physical addresses, 372
 - mapping virtual addresses to, 375
- pins 9/10 of parallel connector, 253
 - generating interrupts, 265
- platform dependency, 11
 - bit operations and, 284
 - kmalloc flags and, 209
 - for modules, 27
 - porting and, 232-234
 - /proc/stat file, 257
- platform-specific directories, 524
- PLIP (Parallel Line Internet Protocol)
 - using Ethernet headers, 456
 - interrupt handling differences, 450
 - overriding ARP, 457
- Plug-and-Play (PnP) specification, 496
- pm.c file, 512
- pmd_offset(), 377
- PMD (Page Mid-level Directory) page table, 375
- pmd_val(), 377
- PnP (Plug-and-Play) specification, 496
- pointers and invalid dereferencing, 111-118
- Point-to-Point Protocol (PPP) and interrupt handling differences, 450
- policy, driver, 2-4
 - controlling devices by printing and, 140
- poll method, 64, 154-159
 - data structures of, 159
- poll_table_entry structure, 158
- poll_table structure, 154, 158
- poll_wait(), 154, 179
- POLLERR flag, 155
- poll.h header file, 154, 179
- POLLHUP flag, 155
- POLLIN flag, 155
- POLLOUT flag, 155
- POLLPRI flag, 155
- POLLRDBAND flag, 155
- POLLRDNORM flag, 155
- POLLWRBAND flag, 155
- POLLWRNORM flag, 155
- portability, 297-302
 - data types and, 293-297
 - devfs (device filesystem), 90
 - porting and, 232-234
- ports, 36-41, 229-234
 - accessing different sizes, 230
 - allocating, 39
 - avoiding collisions, 37
 - parallel (see parallel ports)
 - platform dependency and, 232-234
- post-install directive (modprobe), 308
- post-remove directive (modprobe), 308
- PowerPC architecture
 - page tables not used in, 377

Index

- PowerPC architecture (continued)
 - PCI DMA interface support, 411
 - porting and, 233
- PPP (Point-to-Point Protocol) and interrupt
 - handling differences, 450
- pread method, 79, 91
 - lseek method and, 164
- precision, temporal, 185
- predefined
 - ioctl method commands, 133
 - task queues, 192-198
- preemption and concurrency, 20
- prefetchable bit, 483
- prefixes, 18, 44
- pre-install directive (modprobe), 308
- preprocessor, using to monitor
 - driver, 101-103
- pre-remove directive (modprobe), 308
- printing
 - controlling devices by, 140
 - to debug code, 97-103
 - from gdb debugger, 121
 - interface-specific data, 296
 - partition information, 359
 - _t data items, 296
- printk.c file, 512
- printk(), 15, 52
 - circular buffers for, 100
 - current pointer and, 21
 - debugging with, 97-100, 103
 - logging messages from, 100
 - loglevel strings for, 98
 - turning debug messages on/off, 101
- priority
 - asynchronous notification and, 159-162
 - immediate queue, 193, 197
 - memory allocation, 36, 208
 - message (see loglevels)
- private_data field (file structure), 68, 147
- privileged operations, 137
- probe_irq_off(), 259, 289
- probe_irq_on(), 259, 289
- probe method, 491
- Probes, Dynamic, 127
- probing, 36-41
 - backward compatibility issues, 466
 - for IRQ numbers, 258-262
 - shared interrupts and, 276
 - for ISA memory, 245-247
 - for network devices, 432
- proc_dir_entry
 - create_proc_read_entry() and, 106
 - proc_register_dynamic() and, 107
- /proc filesystem
 - creating
 - /proc entries, 106
 - read-only /proc files, 104
 - debugging with, 103-107
 - installing an interrupt handler, 256
 - removing /proc entries, 107
 - shared interrupts and, 277
 - vs. ioctl method, 108
- /proc/bus/pci file
 - backward compatibility issues, 503
 - browsing configuration space, 481
 - visibility of hardware addresses, 471
- /proc/bus/pci/devices file, 474
- /proc/devices file, 58
- processes
 - access to multiple, 167
 - avoiding race conditions with
 - spinlocks, 166, 281-283
 - kernel timers for, 200-203
 - opening devices for each process, 165
 - requiring, 192
 - sleeping, 141-148
 - race conditions and, 286-288
 - task queues for, 189-200
 - wait queues and, 141-147
 - waking up (see waking up processes)
- processor.h header file, 497
- processor-specific registers, 182-184
- proc_fs.h header file, 104
- /proc/interrupts file, 256, 289
 - shared interrupts and, 277
- /proc/iomem file, 39, 53
- /proc/ioports file, 37, 53
- /proc/kcore file, 120
- /proc/kmsg file, 100
- /proc/ksyms file, 27, 53
 - ksymoops and, 115
 - module version support and, 315
 - searching for registration functions, 30
- /proc/modules file, 34, 51
 - ksymoops and, 114

- /proc/pci file
 - backward compatibility issues, 503
 - browsing configuration space, 482
 - visibility of hardware addresses, 471
- /proc/pcidata file, 482
- /proc/pciregions file
 - browsing configuration space, 486
- proc_register(), 107
- proc_register_dynamic(), 107
- /proc/slabinfo file, 213
- /proc/stat file, 257, 289
- /proc/sys/kernel/printk file, reading
 - console loglevel with, 99
- producer/consumer algorithm, 279
- programmable decoders, 485
- programming drivers (see writing, drivers)
- programs, obtaining, 12
- protect method, 381
- proto_ops structure, 516
- pte_offset(), 377
- pte_page(), 378
- pte_present(), 378
- pte_val(), 377
- PTRS_PER_PGD macro, 377
- PTRS_PER_PMD macro, 377
- PTRS_PER_PTE macro, 377
- put_unaligned(), 299, 304
- __put_user(), 136, 178
- put_user(), 136, 178
 - Linux 2.0 version, 174
- pwrite method, 79, 91
 - lseek method and, 164

Q

- quantum (memory area), 73
 - race conditions and, 76
 - reading/writing one at a time, 85
- querying to debug, 103-108
- queue heads, active, 342
- queue_task_irq_off(), 204
- queue_task_irq(), 204
- queue_task(), 191, 206
 - rescheduling tasks, 192
 - running custom task queues, 198
 - scheduler queue and, 195
 - timer queue and, 196
 - vs. queue_task_irq, 204

- queues
 - initializing/cleaning up, 323
 - request (see request queues)
 - scheduler queue, 192, 194-196
 - task (see task queues)
 - timer (see entries under tq_; timer queue)
 - wait (see wait queues)

R

- race conditions, 20
 - avoiding, with wait_event macros, 142, 179
 - interrupt handling and, 278-288
 - introduction to, 76-78
 - kernel timers and, 203
 - single-processor vs. SMP systems, 166
- RAID drivers
 - drivers/md directory, 519
 - “make request” function and, 346
- RAM
 - probing ISA memory for, 246
 - remapping, 390-394
 - reserving high RAM addresses, 223
 - vs. I/O registers, 227-229
- random numbers, 255
- ranges, resource, 40
- raw I/O and user-space buffers, 397-400
- rd.c file, 519
- rdtsc/rdtscl functions, 183, 205
- read_ahead global array, 325, 367
- read_lock_bh(), 283, 291
- read_lock_irqsave(), 283, 291
- read_lock_irq(), 283, 291
- read_lock(), 283, 291
- read method, 64, 78-81
 - arguments to, 79
 - code for, 81
 - configuring DMA controller, 415
 - f_pos field (file structure) and, 67, 91
 - get_info() and, 104
 - lseek method and, 163
 - poll method and, 157
 - read_proc() and, 104
 - return values, rules for interpreting, 80
 - strace command and, 109

Index

- read method (continued)
 - syntax in Linux 2.0, 92
- read_proc(), 104-107
 - connecting to /proc hierarchy, 106
- read_unlock_bh(), 283, 291
- read_unlock_irqrestore(), 283, 291
- read_unlock_irq(), 283, 291
- read_unlock(), 283, 291
- read_write.c file, 513
- readb(), 240, 250
- readdir method, 64
- reader-writer spinlocks, 283
- reading
 - blocking I/O, 141-153
 - blocking/nonblocking operations, 148
 - poll method, 154-159
 - select method, 154-159
 - testing, 153
 - from a device, 78-81
- readl(), 240, 250
- readq(), 241
- readv method, 66, 84
- read/write instructions, reordering, 227
- read/write position, changing, 64
- readw(), 240, 250
- rebuild_header method, 440
 - backward compatibility issues, 466
- reception of packets, 429, 448-450
 - multicasting, 460-464
- reentrancy, 20, 118, 147
- register_blkdev(), 322, 366
- register_cdrom(), 520
- register_chrdev(), 56-58, 95
 - vs. register_blkdev(), 322
- register_disk(), 369
 - accessing data within partitions, 360
 - backward compatibility issues, 366
 - printing partition information, 359
 - reading generic disk partition table, 358
 - registering devices, 327
- register_framebuffer(), 522
- register_netdev(), 467
- REGISTER_SYMTAB macro, 49
- register_syntab(), 48, 51
- registering
 - block drivers, 322-328
 - DMA usage, 414
 - facilities in init_module, 29-32
 - network drivers, 430
 - ports, 38
- registers
 - I/O, 227-229
 - mapping, 405
 - scatterlists and, 409
 - PCI configuration, 475-479
 - processor-specific, 182-184
- release_dma_lock(), 416, 424
- release_irq(), 276
- release_mem_region(), 53, 250
 - backward compatibility issues, 47
 - working with I/O memory, 40, 239
- release method, 65, 72
 - blocking open and, 169
 - syntax in Linux 2.0, 92
 - unmounting block devices, 349
 - (see also close method)
- release_region(), 52, 250
 - backward compatibility issues, 47
 - working with I/O ports, 38, 229
- remap_page_range(), 384-386, 421
 - limitations in dealing with RAM, 390
 - mapping addresses returned by ioremap, 395
- remapping
 - I/O regions, 389
 - PCI regions, 485
 - RAM, 390-394
 - virtual addresses, 394
- removable block devices, 352-354
- remove method, 492
- remove_proc_entry(), 107
- __remove_wait_queue, 287, 292
- remove_wait_queue(), 179, 287, 292
- reordering read/write instructions, 227
- repatch program, 527
- request_dma(), 414, 423
- request function
 - backward compatibility issues, 364
 - basic design of, 330-334
 - buffer cache and, 336
 - interrupt-driven devices and, 362
 - io_request_lock and, 338
 - multiqueue block drivers and, 343
 - register_disk and, 359
 - registering block devices, 323

- request function (continued)
 - splitting up multibuffer requests, 339
 - transferring data, 332
 - request_irq(), 253, 289
 - installing shared handlers, 275
 - when to call, 255
 - request_mem_region(), 53, 250
 - backward compatibility issues, 47
 - working with I/O memory, 40, 239
 - request_module(), 306, 319
 - inter_module_get_request() and, 313
 - loading modules, 307
 - modprobe program and, 306
 - security issues for module names, 309
 - request queues, 324
 - active queue heads and, 342
 - blk.h header file and, 328-330
 - block drivers not using, 345-348
 - buffers in, 336
 - defining, 343
 - initializing device-specific, 343
 - introduction to, 330-331
 - I/O request locks (see io_request_lock)
 - manipulating, 337
 - multiqueue block drivers and, 342-345
 - request_queue structure, 335
 - __request_region(), 41
 - request_region(), 52, 250
 - backward compatibility issues, 47
 - working with I/O ports, 38, 229
 - request structure, 332
 - buffer cache and, 335
 - releasing back to kernel, 338
 - requesting interrupts (see interrupt handlers)
 - requests, block driver, 330-348
 - blocking, 168
 - clustered, 340
 - handling data transfer, 332-334
 - interrupt-driven devices and, 362
 - partitionable devices and, 361
 - requeuing/rescheduling tasks, 192
 - reserved pages, remapping, 390-394
 - reserving high RAM addresses, 223
 - resetup_one_dev(), 366
 - resolution, time, 185
 - resolving Ethernet addresses, 455-458
 - resource ranges, 40
 - resources
 - allocating in Linux 2.4, 40
 - managing, 35-41
 - backward compatibility for, 47
 - PCI, 484
 - restore_flags(), 252
 - restricting access (see access)
 - resume method, 492
 - revalidate method, 353
 - backward compatibility issues, 364
 - register_disk and, 359
 - ring buffers, DMA, 402
 - RISC processor and inline assembly
 - code, 183
 - rmb(), 228, 249
 - rmmod program, 6, 34
 - dynamically allocating major numbers, 60
 - testing modules using, 16
 - ROM, probing ISA memory for, 246
 - route utility, 429
 - Rules.make file, 26
 - platform dependency and, 27
 - run_task_queue(), 191, 206
 - running custom task queues, 198
 - runtime errors, strace for, 110
 - RW_LOCK_UNLOCKED, 283
 - rwlock_t type, 283, 291
- ## S
- S390 architecture
 - no support for PCI bus, 411
 - porting and, 234
 - SA_INTERRUPT flag, 254, 289
 - fast vs. slow interrupt handling, 262
 - SA_SAMPLE_RANDOM flag, 255, 289
 - SA_SHIRQ flag, 255, 289
 - installing shared handlers, 275
 - SAK (Secure Attention Key) function, 119
 - save_flags(), 252
 - sball driver (example), 321-369
 - adding raw I/O capability, 397-400
 - sballr driver (example), 397-400
 - SBus (Sun-designed bus), 498
 - drivers/sbus directory, 524
 - performing DMA mappings on, 412
 - sbus.h header file, 412

Index

- scatter-gather DMA mappings, 409
- scatterlist structure, 410, 423
- scatterlist.h header file, 410
- scatterlists, mapping, 409
- sched.h header file, 52, 178, 205, 289, 291
 - capable() and, 137
 - interrupt request line functions, 253
 - jiffies value and, 182
 - kernel directory and, 512
 - wait queue code information, 147
- schedule_task(), 192, 195, 206
 - backward compatibility issues, 204
- schedule_timeout(), 188
- scheduler queue (tq_scheduler), 192, 194-196
 - backward compatibility issues, 204
- schedule(), 145, 179, 512
 - delaying execution of code, 187
 - exclusive waits and, 146
 - preventing endless loops with, 118
 - reentrant functions and, 147
- screen layouts, kernel support for, 522
- SCSI drivers, 7
 - drivers/scsi directory, 520
- scsi_ioctl.c file, 520
- scsi_module.c file, 520
- scsi_register_module(), 520
- scsi.c file, 520
- scull driver (example), 54-94, 101, 131, 135-139
- scullc driver (example), 213
- scullp driver (example), 216
 - mapping RAM to user space, 391-394
- scullpipe devices (examples), 150-153
- scullv driver (example), 219-220, 394
- Secure Attention Key (SAK) function, 119
- security, 9
 - module loading and, 309
- seeking a device, 163
 - in Linux version 2.0, 176
- segment.h header file, 95
- select method, 154-159
 - in Linux version 2.0, 175
 - poll method and, 64
- selection.c file, 518
- sema_init(), 76, 95
 - sysdep.h header file and, 94
- semaphore.h header file, 76, 95
- semaphores, 76-78
 - backward compatibility issues, 94
 - detecting deadlocks with IKD, 124
 - incrementing value of, 77
 - initializing, 76
 - not used in interrupt handlers, 279
 - protecting critical code regions, 151
 - vs. spinlocks, 166
- set_bit(), 284, 291
- set_config method, 441
- set_current_state(), 287, 291
 - backward compatibility issues, 288
- set_dma_addr(), 416, 424
- set_dma_count(), 417, 424
- set_dma_mode(), 416, 424
- SET_FILE_OWNER macro, 93
- SET_INTR macro, 329
- set_mac_address method, 441
- set_mb(), 229
- SET_MODULE_OWNER macro, 66, 95, 467
 - backward compatibility issues, 465
 - net_device structure and, 433
- set_multicast_list method, 441, 461-464
 - interface flags and, 439
- set_rmb(), 229
- set_wmb(), 229
- setconsole program (example), 99
- setterm program, 140
- setup_arch(), 507
- sg_dma_address(), 410, 423
- sg_dma_len(), 410, 423
- sharing interrupts, 274-278
- short delays, 188-189
- short driver (example), 237
 - accessing I/O memory, 241
 - BH implementation, 273
 - going to sleep and avoiding race conditions, 286
 - implementing
 - interrupt handlers, 264-266
 - probing in the driver, 261
 - installing an interrupt handler, 255
- shutting down modules (see unloading modules)
- SIGIO signal, 160
- signal handling, 151
 - down_interruptible() and, 77

- signal.c file, 512
- single-open devices, 165
- SIOCDEVPRIVATE commands, 458, 469
- SIOCSIFADDR command, 458
- SIOCSIFMAP command, 458
- size of block devices, 324
- sizing data explicitly, 295
- sk_buff structure
 - fields for, 452
 - receiving packets, 448
 - transmitting packets, 445
- skb_headroom(), 455, 468
- skb_pull(), 455, 468
- __skb_push(), 454, 468
 - backward compatibility issues, 465
- skb_push(), 454, 468
- __skb_put(), 454, 468
- skb_put(), 454, 468
- skb_reserve(), 455, 468
- skb_tailroom(), 454, 468
- skbuff.h header file, 445, 452, 468
- skull driver (example), 22-44
- SLAB_CACHE_DMA flag, 212, 224
- SLAB_CTOR_ATOMIC flag, 212, 224
- SLAB_CTOR_CONSTRUCTOR flag, 213, 224
- SLAB_HWCACHE_ALIGN flag, 212, 224
- SLAB_NO_REAP flag, 212, 224
- slab.c file, 211, 514
- sleep_on_timeout(), 142, 178
 - delaying execution, 187
- sleep_on(), 142, 178
 - avoiding race conditions, 286
- sleeping processes, 141-148
 - avoiding race conditions, 286-288
- SLOW_DOWN_IO statement, 249
- slow interrupt handlers, 262-264
 - backward compatibility issues, 288
- __SMP__ symbol, 22, 50
- SMP systems
 - backward compatibility issues, 48
 - concurrency in the kernel, 20
 - kernel headers and, 22
 - module version control and, 314
 - race conditions and, 76-78
 - running tasklets on, 198-200
 - spinlocks to avoid race conditions, 166
 - writing reentrant code, 147
- snapshot of PCI configuration, 481
- snull driver (example), 426-457
- sock_ioctl(), 458
- socket buffers, 445, 452-455
 - allocating, 449, 454
 - functions acting on, 454
- socket.c file, 516
- sockios.h header file, 458, 469
- soft lockup detector (IKD), 124
- softirq.c file, 512
- softnet implementation and backward
 - compatibility, 464
- software loops, 188
- software memory barriers, 228, 249
- software versions (see version numbering)
- software-mapped I/O memory, 242
- sound cards, drivers for, 521
- sound_install_audiodrv(), 521
- SPARC architecture
 - defining disable_irq/enable_irq as
 - pointers, 268
 - high memory, 210
 - I/O memory management support, 411
 - platform dependency and, 27
 - porting and, 234
 - SBus, 498
 - performing DMA mappings on, 412
- SPARC64 platform
 - data alignment, 300
 - directly mapped memory, 240
 - gdb debugger and, 121
 - objdump utility and, 118
 - oops messages and, 116
- special files, 55
- spin_is_locked(), 282, 290
- spin_lock_bh(), 282, 290
- spin_lock_init(), 166, 180, 281, 290
- spin_lock_irqsave(), 281, 290
 - avoiding deadlocks with, 282
- spin_lock_irq(), 281, 290
- spin_lock(), 167, 180, 281, 290
- spin_trylock(), 282, 290
- spin_unlock_bh(), 282, 290
- spin_unlock_irqrestore(), 282, 290
- spin_unlock_irq(), 282, 290
- spin_unlock_wait(), 282, 290
- spin_unlock(), 167, 180, 282, 290
- spinlock_t type, 166, 180, 281, 290, 367

Index

- spinlock.h header file, 166, 180, 281, 290
 - spinlocks, 281-283
 - dma_spin_lock, 416
 - io_request_lock, 338
 - page_table_lock, 378
 - reader-writer, 283
 - vs. semaphores, 166
 - xmit_lock, 443, 446
 - spull driver (example), 355-364
 - device methods for, 360
 - stack meter (IKD feature), 124
 - stacking modules, 28
 - standard C data types, 293
 - start_kernel(), 507-509
 - stat file, 257, 289
 - static symbols, 18
 - statistics
 - on caches, 213
 - on interrupts, 257
 - on network interfaces, 433, 441, 459
 - sti(), 252
 - stop method, 440, 443
 - strace command, 108-110
 - streaming DMA mappings, 406
 - setting up, 407-409
 - string operations, 231
 - struct page pointer, 373-375
 - backward compatibility issues, 419
 - struct timeval pointer, 185, 205
 - subsystem deviceID PCI register, 476
 - subsystem vendorID PCI register, 476
 - sunrpc subdirectory, 516
 - Super-H architecture
 - no support for PCI bus, 411
 - porting and, 234
 - supervisor mode, 19
 - suser(), 175
 - suspend method, 492
 - swap_state.c file, 515
 - swapfile.c file, 515
 - swapout method, 382
 - switch statement, with ioctl, 129, 133
 - symbols
 - driver-specific, 328-330
 - exporting, 29, 317
 - in Linux 2.0, 48-50
 - hiding global, 29
 - mangling symbol names, 314-317
 - static, declaring as, 18
 - symbol table, 27-29
 - klogd and, 114
 - module version control and, 315
 - symtab_begin.h header file, 51
 - symtab_end.h header file, 51
 - sync method, 382
 - synchronization (see lock method; race conditions)
 - sys_create_module(), 24
 - sys_delete_module system call, 33
 - sys_syslog(), 98
 - sysctl_net.c file, 516
 - sysdep.h header file, 26
 - backward compatibility issue, 47-50
 - sema_init() and, 94
 - SET_FILE_OWNER macro and, 93
 - wait queues in Linux 2.0/2.2, 172
 - syslogd daemon
 - logging messages, 100
 - performance problems with, 103
 - sysrq.txt file, 119
 - <sys/sched.h> header file
 - capable() and, 137
 - system calls, 24
 - invoked by init thread, 511
 - system faults
 - changing message loglevels after, 99
 - debugging, 110-120
 - handling, kernels vs. applications, 19
 - system hangs, 118
 - precautions when reproducing, 120
 - system resources
 - allocating in Linux 2.4, 40
 - managing, 35-41
 - backward compatibility for, 47
 - system.h header file, 228, 249
 - System.map file
 - klogd and, 114
 - kysmoops and, 114
- ## T
- _t data types, 296
 - tagged initialization format, 63
 - avoiding flush method, 93
 - declaring file_operations structure, 66

- tail pointers and circular buffers, 280
- take_over_console(), 522
- TASK_EXCLUSIVE flag, 146
- TASK_INTERRUPTIBLE flag, 145, 287, 291
- task_queue, 191, 206
- task queues, 189-200
 - backward compatibility issues, 204
 - data structures of, 190
 - declaring, 191
 - declaring custom, 198
 - driver timeline, 193
 - predefined, 192-198
 - requeuing/rescheduling tasks, 192
 - running, 191
- TASK_RUNNING flag, 145, 287, 291
- TASK_UNINTERRUPTIBLE flag, 291
- tasklet_disable(), 200, 207
- tasklet_enable(), 200, 207
- tasklet_kill(), 200, 207
- tasklet_schedule(), 199, 206, 270, 290
 - BH mechanism and, 272
- tasklets, 198-200, 270
 - scheduling, 199
- tcpdump program, 430
- terminals, selecting for messages, 99
- test_and_change_bit(), 285, 291
- test_and_clear_bit(), 285, 291
- test_and_set_bit(), 285, 291
- test_bit(), 284, 291
- testing (non)blocking operations, 153
- “thundering herd” problem, 146
- time, 181-207
 - delaying execution of code, 186-189
 - HZ (time frequency), 181, 297
 - kernel timers, 200-203
 - sleeping processes, 286-288
 - time intervals in the kernel, 181-184, 297
- time.c/timer.c files, 512
- time.h header file, 206
- timeouts
 - backward compatibility issues, 204
 - of kernel timers, 201
 - scheduling, 188
 - setting up short-term, 187
 - transmission (see transmission timeouts)
- TIMER_BH bottom half, 272
- timer interrupts, 181
- timer_list structure, 201
- timer queue element structure, 190
- timer queue (tq_timer), 193, 196, 206
 - BH mechanism and, 272
- timer.h header file, 201, 207
- timers, 200-203
- timestamp counter (TSC), 183
- TIOCLINUX command, 99
- to_kdev_t(), 62
- token ring networks, setting up interfaces
 - for, 437
- top-half vs. bottom-half handlers, 269
- tq_immediate queue, 193, 197, 206
 - BH mechanism and, 272
 - writing a BH bottom half, 273
- tq_scheduler queue, 192, 194-196
 - backward compatibility issues, 204
- tq_struct structure, 190
- tq_timer(), 193, 196, 206
 - BH mechanism and, 272
- TQUEUE_BH bottom half, 272
- tqueue.h header file, 190, 192, 206
- tr_configure(), 437
- tracing programs, 108-110
 - Linux Trace Toolkit (LTT), 127
- transistor-transistor logic (TTL) levels, 235
- transmission concurrency, controlling, 446
- transmission of packets, 429, 445-448
 - multicasting, 460-464
- transmission timeouts, 433, 447
 - tx_timeout method and, 440
 - watchdog_timeo field and, 442
- traversal of linked lists, 302
- troubleshooting, 97
 - porting problems, 232-234
 - race conditions, 278-288
 - system hangs, 118
 - wrong font on console, 140
 - (see also debugging)
- truncating devices on open, 71
- TSC (timestamp counter), 183
- TTL (transistor-transistor logic) levels, 235
- tunelp program, 3
- tx_timeout method, 440, 447
- TYPE macro, splitting minor numbers, 69
- types.h header file (asm), 295
- types.h header file (linux), 295, 303

Index

U

- u8, u16, u32, u64 data types, 295, 303
- uaccess.h header file, 78, 95, 135, 177
- uClinix port
 - different ld scripts needed for, 510
 - mmnommu directory, 515
- udelay(), 188, 206
- uint8_t/uint32_t types, 295
- uintptr_t type (C99 standard), 294
- uio.h header file, 84
- unaligned data, 299
- unaligned.h header file, 299, 304
- uniqueness of ioctl command numbers, 130
- universal serial bus drivers (see USB drivers)
- Unix design books, 528
- unix subdirectory, 516
- unloading modules, 16, 34, 61
 - on demand, 305-311
 - for network drivers, 434
 - usage count and, 33, 313
 - (see also cleanup_module())
- unlock_kiovec(), 396, 422
- unmap_kiobuf(), 399, 422
- unmap method, 381
- unregister_blkdev(), 322, 366
- unregister_cdrom(), 520
- unregister_chrdev(), 61, 95
- unregister_netdev(), 467
- unregistering
 - block drivers, 322-328
 - facilities, 30
- unsigned type, 230
 - platform dependencies and, 232
- up(), 77, 95
- urandom device, 255
- usage count, 386
 - accessing data within partitions, 360
 - decremented by release method, 72
 - incremented by open method, 68
 - maintained by block drivers, 323
 - maintaining via owner field, 71
 - modules, 33, 313
 - backward compatibility issues, 93
 - nopage method and, 392
- usb_deregister(), 500, 505
- usb_driver structure, 500, 505
- usb_register(), 500, 505
- USB (universal serial bus) drivers, 7, 500
 - call_usermodehelper and, 311
 - directory for, 524
 - lookaside caches, 211-214
 - stacking on usbcore/input modules, 28
 - writing, 500-502
- usbcore module, 28
- usb.h header file, 504
- __USE_OLD_SELECT__ preprocessor symbol, 176
- __USE_OLD_SYMTAB__, 49
- user mode, 19
 - helper programs, running, 311
- user space, 19
 - access to, in Linux 2.0, 173-175
 - accessing I/O ports from, 230
 - capabilities/restrictions in, 137
 - changes in access to, 94
 - entering via init process, 511
 - explicitly sizing data in, 295
 - mapping RAM to, 390-394
 - reentrant functions and, 147
 - retrieving datum from, 136
 - transferring to/from kernel space, 78-84
 - watching programs run in, 108-110
 - writing drivers in, 45
- user virtual addresses, 371
- User-Mode Linux, 126
- users, restricting access to
 - simultaneous, 167
- UTS_RELEASE macro, 25

V

- __va(), 372, 421
 - backward compatibility issues, 420
- validating
 - block driver requests, 330
 - disk changes, 353
- variables, declaring as volatile, 279
- vector operations (readv/writv), 84
- vendorID PCI register, 476
- verify_area(), 173-175
- VERIFY_ symbols, 135, 178
- version dependency, 24-26
 - module version control, 314-318

- version numbering, 10
 - char drivers, 55-62
 - major device numbers, 56-61
 - minor device numbers, 56, 61, 69
 - versioned symbols, 315
 - enabling module version control, 316
 - exporting, 317
 - version.h header file, 25, 52
 - VESA Local Bus (VLB) devices, 498
 - vfree(), 217, 225
 - backward compatibility issues, 248
 - video_device structure, 523
 - video devices, directory for, 522
 - video_register_device(), 523
 - video/videodev.c file, 523
 - virt_to_bus(), 404, 422
 - backward compatibility issues, 420
 - virt_to_page(), 374, 421
 - backward compatibility issues, 418
 - mapping memory with nopage, 389
 - virtual addresses, 372
 - assigning to devices, 242
 - mapping to physical addresses, 375
 - remapping, 394
 - Sbus peripherals and, 498
 - vmalloc and related functions, 217-220
 - virtual memory areas (VMAs), 378-382
 - main fields in vm_area_struct, 380
 - VLB (VESA Local Bus) devices, 498
 - vm_area_struct structure, 380
 - backward compatibility issues, 419
 - VM_IO flag, 381
 - vm_operations_struct structure, 381
 - vm_private_data field (vm_area_struct), 393
 - backward compatibility issues, 419
 - VM_RESERVED flag, 381
 - VMA_OFFSET macro, 387
 - VMALLOC_VMADDR(), 395
 - vmalloc.c file, 514
 - vmalloc.h header file, 217, 225
 - vmalloc(), 217-220, 225, 394
 - vs. kmalloc(), 217-219
 - VMAs (virtual memory areas), 378-382
 - main fields in vm_area_struct, 380
 - vmlinux kernel image, 510
 - vmscan.c file, 515
 - volatile, declaring variables as, 279
 - vremap() in Linux 2.x only, 223, 248
 - vsprintf.c file, 517
 - vt.c file, 518
- ## W
- wait_event_interruptible(), 142, 179, 288, 292
 - wait_event(), 142, 179, 288, 292
 - vs. interruptible_sleep_on(), 145
 - wait_queue_head_t, 178
 - new in Linux version 2.3.1, 172
 - poll table entries and, 158
 - sleeping/waking up processes, 141-143
 - working with advanced applications, 144
 - wait_queue_t type, 144, 179
 - poll table entries and, 158
 - wait queues, 141-147
 - avoiding race conditions, 287
 - defined type for, 178
 - delaying code execution, 187
 - in Linux versions 2.0/2.2, 172
 - manipulating, 144
 - poll table entries and, 158
 - putting processes into, 179
 - wait.h header file, 144, 178
 - wake_up_interruptible_sync(), 143, 178
 - wake_up_interruptible(), 143, 178
 - wake_up_sync(), 143, 178
 - wake_up(), 143, 178
 - resuming execution of code, 188
 - waking up processes, 142
 - exclusive waits and, 146
 - functions used for, 178
 - release method and, 169
 - Wall flag (gcc), 23, 296
 - watchdog_timeo field (net_device structure), 442, 447
 - watching programs in user space, 108-110
 - web sites related to Linux kernels, xv
 - wmb(), 228, 249
 - wppage method, 382
 - backward compatibility issues, 419
 - wrapper functions, compiling under 2.0
 - headers, 92
 - write_lock_bh(), 283, 291
 - write_lock_irqsave(), 283, 291
 - write_lock_irq(), 283, 291

Index

write_lock(), 283, 291
write method, 64, 78-80
 code for, 83
 configuring DMA controller, 415
 f_pos field (file structure) and, 67, 91
 input/output buffers and, 148
 llseek method and, 163
 poll method and, 157
 return values, rules for interpreting, 82
 select method and, 157
 strace command and, 109
 syntax in Linux 2.0, 92
write_unlock_bh(), 283, 291
write_unlock_irqrestore(), 283, 291
write_unlock_irq(), 283, 291
write_unlock(), 283, 291
writeb(), 240, 250
writel(), 240, 250
writeq(), 241
writev method, 66, 84
writew(), 240, 250
writing, 97
 blocking I/O, 141-149
 blocking/nonblocking operations, 148
 control sequences to devices, 140
 to a device, 78-80, 82-84
 drivers
 using devfs, 85-91
 reentrant code, 147
 in user space, 45
 version numbering, 10
 watching user-space programs
 run, 108-110
 writer's role in, 2-4
 interrupt handler bottom halves, 273
 interrupt handlers, 264-268
 makefiles, 22
 (see also debugging)

X

x86 architecture
 interrupt handling on, 263
 limitations of platform, 510
 PCI DMA interface support, 411
 porting and, 233

x/i (examine instructions) command, 121
xtime variable, 185

Z

zImage file, 510