

Index

Note: Online information is listed by chapter and section number followed by page numbers (OL3.11-7). Page references preceded by a single letter with hyphen refer to appendices.

- 1-bit ALU, B-26–29. *See also* Arithmetic logic unit (ALU)
 - adder, B-27
 - CarryOut, B-28
 - for most significant bit, B-33
 - illustrated, B-29
 - logical unit for AND/OR, B-27
 - performing AND, OR, and addition, B-31, B-33
 - 32-bit ALU, B-29–38. *See also* Arithmetic logic unit (ALU)
 - defining in Verilog, B-35–38
 - from 31 copies of 1-bit ALU, B-34
 - illustrated, B-36
 - ripple carry adder, B-29
 - tailoring to MIPS, B-31–35
 - with 32 1-bit ALUs, B-30
 - 32-bit immediate operands, 112–113
 - 7090/7094 hardware, OL3.11-7
- A**
- Absolute references, 126
 - Abstractions
 - hardware/software interface, 22
 - principle, 22
 - to simplify design, 11
 - Accumulator architectures, OL2.21-2
 - Acronyms, 9
 - Active matrix, 18
 - add (Add), 64
 - add.d (FP Add Double), A-73
 - add.s (FP Add Single), A-74
 - Add unsigned instruction, 180
 - addi (Add Immediate), 64
 - Addition, 178–182. *See also* Arithmetic
 - binary, 178–179
 - floating-point, 203–206, 211, A-73–74
 - instructions, A-51
 - operands, 179
 - significands, 203
 - speed, 182
 - addiu (Add Imm.Unsigned), 119
 - Address interleaving, 381
 - Address select logic, D-24, D-25
 - Address space, 428, 431
 - extending, 479
 - flat, 479
 - ID (ASID), 446
 - inadequate, OL5.17-6
 - shared, 519–520
 - single physical, 517
 - unmapped, 450
 - virtual, 446
 - Address translation
 - for ARM cortex-A8, 471
 - defined, 429
 - fast, 438–439
 - for Intel core i7, 471
 - TLB for, 438–439
 - Address-control lines, D-26
 - Addresses
 - 32-bit immediates, 113–116
 - base, 69
 - byte, 69
 - defined, 68
 - memory, 77
 - virtual, 428–431, 450
 - Addressing
 - 32-bit immediates, 113–116
 - base, 116
 - displacement, 116
 - immediate, 116
 - in jumps and branches, 113–116
 - MIPS modes, 116–118
 - PC-relative, 114, 116
 - pseudodirect, 116
 - register, 116
 - x86 modes, 152
 - Addressing modes, A-45–47
 - desktop architectures, E-6
 - addu (Add Unsigned), 64
 - Advanced Vector Extensions (AVX), 225, 227
 - AGP, C-9
 - Algol-60, OL2.21-7
 - Aliasing, 444
 - Alignment restriction, 69–70
 - All-pairs N-body algorithm, C-65
 - Alpha architecture
 - bit count instructions, E-29
 - floating-point instructions, E-28
 - instructions, E-27–29
 - no divide, E-28
 - PAL code, E-28
 - unaligned load-store, E-28
 - VAX floating-point formats, E-29
 - ALU control, 259–261. *See also* Arithmetic logic unit (ALU)
 - bits, 260
 - logic, D-6
 - mapping to gates, D-4–7
 - truth tables, D-5
 - ALU control block, 263
 - defined, D-4
 - generating ALU control bits, D-6
 - ALUOp, 260, D-6
 - bits, 260, 261
 - control signal, 263
 - Amazon Web Services (AWS), 425
 - AMD Opteron X4 (Barcelona), 543, 544
 - AMD64, 151, 224, OL2.21-6
 - Amdahl's law, 401, 503
 - corollary, 49
 - defined, 49
 - fallacy, 556
 - and (AND), 64
 - AND gates, B-12, D-7
 - AND operation, 88
 - AND operation, A-52, B-6
 - andi (And Immediate), 64
 - Annual failure rate (AFR), 418
 - versus*.MTTF of disks, 419–420
 - Antidependence, 338
 - Antifuse, B-78
 - Apple computer, OL1.12-7

- Apple iPad 2 A1395, 20
 - logic board of, 20
 - processor integrated circuit of, 21
 - Application binary interface (ABI), 22
 - Application programming interfaces (APIs)
 - defined, C-4
 - graphics, C-14
 - Architectural registers, 347
 - Arithmetic, 176–236
 - addition, 178–182
 - addition and subtraction, 178–182
 - division, 189–195
 - fallacies and pitfalls, 229–232
 - floating-point, 196–222
 - historical perspective, 236
 - multiplication, 183–188
 - parallelism and, 222–223
 - Streaming SIMD Extensions and
 - advanced vector extensions in x86, 224–225
 - subtraction, 178–182
 - subword parallelism, 222–223
 - subword parallelism and matrix multiply, 225–228
 - Arithmetic instructions. *See also*
 - Instructions
 - desktop RISC, E-11
 - embedded RISC, E-14
 - logical, 251
 - MIPS, A-51–57
 - operands, 66–, 73
 - Arithmetic intensity, 541
 - Arithmetic logic unit (ALU). *See also*
 - ALU control; Control units
 - 1-bit, B-26–29
 - 32-bit, B-29–38
 - before forwarding, 309
 - branch datapath, 254
 - hardware, 180
 - memory-reference instruction use, 245
 - for register values, 252
 - R-format operations, 253
 - signed-immediate input, 312
 - ARM Cortex-A8, 244, 345–346
 - address translation for, 471
 - caches in, 472
 - data cache miss rates for, 474
 - memory hierarchies of, 471–475
 - performance of, 473–475
 - specification, 345
 - TLB hardware for, 471
 - ARM instructions, 145–147
 - 12-bit immediate field, 148
 - addressing modes, 145
 - block loads and stores, 149
 - brief history, OL2.21-5
 - calculations, 145–146
 - compare and conditional branch, 147–148
 - condition field, 324
 - data transfer, 146
 - features, 148–149
 - formats, 148
 - logical, 149
 - MIPS similarities, 146
 - register-register, 146
 - unique, E-36–37
 - ARMv7, 62
 - ARMv8, 158–159
 - ARPANET, OL1.12-10
 - Arrays, 415
 - logic elements, B-18–19
 - multiple dimension, 218
 - pointers *versus*, 141–145
 - procedures for setting to zero, 142
 - ASCII
 - binary numbers *versus*, 107
 - character representation, 106
 - defined, 106
 - symbols, 109
 - Assembler directives, A-5
 - Assemblers, 124–126, A-10–17
 - conditional code assembly, A-17
 - defined, 14, A-4
 - function, 125, A-10
 - macros, A-4, A-15–17
 - microcode, D-30
 - number acceptance, 125
 - object file, 125
 - pseudoinstructions, A-17
 - relocation information, A-13, A-14
 - speed, A-13
 - symbol table, A-12
 - Assembly language, 15
 - defined, 14, 123
 - drawbacks, A-9–10
 - floating-point, 212
 - high-level languages *versus*, A-12
 - illustrated, 15
 - MIPS, 64, 84, A-45–80
 - production of, A-8–9
 - programs, 123
 - translating into machine language, 84
 - when to use, A-7–9
 - Asserted signals, 250, B-4
 - Associativity
 - in caches, 405
 - degree, increasing, 404, 455
 - increasing, 409
 - set, tag size *versus*, 409
 - Atomic compare and swap, 123
 - Atomic exchange, 121
 - Atomic fetch-and-increment, 123
 - Atomic memory operation, C-21
 - Attribute interpolation, C-43–44
 - Automobiles, computer application in, 4
 - Average memory access time (AMAT), 402
 - calculating, 403
- ## B
- Backpatching, A-13
 - Bandwidth, 30–31
 - bisection, 532
 - external to DRAM, 398
 - memory, 380–381, 398
 - network, 535
 - Barrier synchronization, C-18
 - defined, C-20
 - for thread communication, C-34
 - Base addressing, 69, 116
 - Base registers, 69
 - Basic block, 93
 - Benchmarks, 538–540
 - defined, 46
 - Linpack, 538, OL3.11-4
 - multicores, 522–529
 - multiprocessor, 538–540
 - NAS parallel, 540
 - parallel, 539
 - PARSEC suite, 540
 - SPEC CPU, 46–48
 - SPEC power, 48–49
 - SPECrate, 538–539
 - Stream, 548
 - beq (Branch On Equal), 64
 - bge (Branch Greater Than or Equal), 125
 - bgt (Branch Greater Than), 125
 - Biased notation, 79, 200
 - Big-endian byte order, 70, A-43
 - Binary numbers, 81–82

- ASCII *versus*, 107
 - conversion to decimal numbers, 76
 - defined, 73
 - Bisection bandwidth, 532
 - Bit maps
 - defined, 18, 73
 - goal, 18
 - storing, 18
 - Bit-Interleaved Parity (RAID 3), OL5.11-5
 - Bits
 - ALUOp, 260, 261
 - defined, 14
 - dirty, 437
 - guard, 220
 - patterns, 220–221
 - reference, 435
 - rounding, 220
 - sign, 75
 - state, D-8
 - sticky, 220
 - valid, 383
 - ble (Branch Less Than or Equal), 125
 - Blocking assignment, B-24
 - Blocking factor, 414
 - Block-Interleaved Parity (RAID 4), OL5.11-5–5.11-6
 - Blocks
 - combinational, B-4
 - defined, 376
 - finding, 456
 - flexible placement, 402–404
 - least recently used (LRU), 409
 - loads/stores, 149
 - locating in cache, 407–408
 - miss rate and, 391
 - multiword, mapping addresses to, 390
 - placement locations, 455–456
 - placement strategies, 404
 - replacement selection, 409
 - replacement strategies, 457
 - spatial locality exploitation, 391
 - state, B-4
 - valid data, 386
 - blt (Branch Less Than), 125
 - bne (Branch On Not Equal), 64
 - Bonding, 28
 - Boolean algebra, B-6
 - Bounds check shortcut, 95
 - Branch datapath
 - ALU, 254
 - operations, 254
 - Branch delay slots
 - defined, 322
 - scheduling, 323
 - Branch equal, 318
 - Branch instructions, A-59–63
 - jump instruction *versus*, 270
 - list of, A-60–63
 - pipeline impact, 317
 - Branch not taken
 - assumption, 318
 - defined, 254
 - Branch prediction
 - as control hazard solution, 284
 - buffers, 321, 322
 - defined, 283
 - dynamic, 284, 321–323
 - static, 335
 - Branch predictors
 - accuracy, 322
 - correlation, 324
 - information from, 324
 - tournament, 324
 - Branch taken
 - cost reduction, 318
 - defined, 254
 - Branch target
 - addresses, 254
 - buffers, 324
 - Branches. *See also* Conditional branches
 - addressing in, 113–116
 - compiler creation, 91
 - condition, 255
 - decision, moving up, 318
 - delayed, 96, 255, 284, 318–319, 322, 324
 - ending, 93
 - execution in ID stage, 319
 - pipelined, 318
 - target address, 318
 - unconditional, 91
 - Branch-on-equal instruction, 268
 - Bubble Sort, 140
 - Bubbles, 314
 - Bus-based coherent multiprocessors, OL6.15-7
 - Buses, B-19
 - Bytes
 - addressing, 70
 - order, 70, A-43
- ## C
- C.mmp, OL6.15-4
 - C language
 - assignment, compiling into MIPS, 65–66
 - compiling, 145, OL2.15-2–2.15-3
 - compiling assignment with registers, 67–68
 - compiling while loops in, 92
 - sort algorithms, 141
 - translation hierarchy, 124
 - translation to MIPS assembly language, 65
 - variables, 102
 - C++ language, OL2.15-27, OL2.21-8
 - Cache blocking and matrix multiply, 475–476
 - Cache coherence, 466–470
 - coherence, 466
 - consistency, 466
 - enforcement schemes, 467–468
 - implementation techniques, OL5.12-11–5.12-12
 - migration, 467
 - problem, 466, 467, 470
 - protocol example, OL5.12-12–5.12-16
 - protocols, 468
 - replication, 468
 - snooping protocol, 468–469
 - snoopy, OL5.12-17
 - state diagram, OL5.12-16
 - Cache coherency protocol, OL5.12-12–5.12-16
 - finite-state transition diagram, OL5.12-15
 - functioning, OL5.12-14
 - mechanism, OL5.12-14
 - state diagram, OL5.12-16
 - states, OL5.12-13
 - write-back cache, OL5.12-15
 - Cache controllers, 470
 - coherent cache implementation techniques, OL5.12-11–5.12-12
 - implementing, OL5.12-2
 - snoopy cache coherence, OL5.12-17
 - SystemVerilog, OL5.12-2
 - Cache hits, 443
 - Cache misses
 - block replacement on, 457
 - capacity, 459

- Cache misses (*Continued*)
 - compulsory, 459
 - conflict, 459
 - defined, 392
 - direct-mapped cache, 404
 - fully associative cache, 406
 - handling, 392–393
 - memory-stall clock cycles, 399
 - reducing with flexible block placement, 402–404
 - set-associative cache, 405
 - steps, 393
 - in write-through cache, 393
- Cache performance, 398–417
 - calculating, 400
 - hit time and, 401–402
 - impact on processor performance, 400
- Cache-aware instructions, 482
- Caches, 383–398. *See also* Blocks
 - accessing, 386–389
 - in ARM cortex-A8, 472
 - associativity in, 405–406
 - bits in, 390
 - bits needed for, 390
 - contents illustration, 387
 - defined, 21, 383–384
 - direct-mapped, 384, 385, 390, 402
 - empty, 386–387
 - FSM for controlling, 461–462
 - fully associative, 403
 - GPU, C-38
 - inconsistent, 393
 - index, 388
 - in Intel Core i7, 472
 - Intrinsity FastMATH example, 395–398
 - locating blocks in, 407–408
 - locations, 385
 - multilevel, 398, 410
 - nonblocking, 472
 - physically addressed, 443
 - physically indexed, 443
 - physically tagged, 443
 - primary, 410, 417
 - secondary, 410, 417
 - set-associative, 403
 - simulating, 478
 - size, 389
 - split, 397
 - summary, 397–398
 - tag field, 388
 - tags, OL5.12-3, OL5.12-11
 - virtual memory and TLB integration, 440–441
 - virtually addressed, 443
 - virtually indexed, 443
 - virtually tagged, 443
 - write-back, 394, 395, 458
 - write-through, 393, 395, 457
 - writes, 393–395
- Callee, 98, 99
- Callee-saved register, A-23
- Caller, 98
- Caller-saved register, A-23
- Capabilities, OL5.17-8
- Capacity misses, 459
- Carry lookahead, B-38–47
 - 4-bit ALUs using, B-45
 - adder, B-39
 - fast, with first level of abstraction, B-39–40
 - fast, with “infinite” hardware, B-38–39
 - fast, with second level of abstraction, B-40–46
 - plumbing analogy, B-42, B-43
 - ripple carry speed *versus*, B-46
 - summary, B-46–47
- Carry save adders, 188
- Cause register
 - defined, 327
 - fields, A-34, A-35
- OLC 6600, OL1.12-7, OL4.16-3
- Cell phones, 7
- Central processor unit (CPU). *See also* Processors
 - classic performance equation, 36–40
 - coprocessor 0, A-33–34
 - defined, 19
 - execution time, 32, 33–34
 - performance, 33–35
 - system, time, 32
 - time, 399
 - time measurements, 33–34
 - user, time, 32
- Cg pixel shader program, C-15–17
- Characters
 - ASCII representation, 106
 - in Java, 109–111
- Chips, 19, 25, 26
 - manufacturing process, 26
- Classes
 - defined, OL2.15-15
 - packages, OL2.15-21
- Clock cycles
 - defined, 33
 - memory-stall, 399
 - number of registers and, 67
 - worst-case delay and, 272
- Clock cycles per instruction (CPI), 35, 282
 - one level of caching, 410
 - two levels of caching, 410
- Clock rate
 - defined, 33
 - frequency switched as function of, 41
 - power and, 40
- Clocking methodology, 249–251, B-48
 - edge-triggered, 249, B-48, B-73
 - level-sensitive, B-74, B-75–76
 - for predictability, 249
- Clocks, B-48–50
 - edge, B-48, B-50
 - in edge-triggered design, B-73
 - skew, B-74
 - specification, B-57
 - synchronous system, B-48–49
- Cloud computing, 533
 - defined, 7
- Cluster networking, 537–538, OL6.9-12
- Clusters, OL6.15-8–6.15-9
 - defined, 30, 500, OL6.15-8
 - isolation, 530
 - organization, 499
 - scientific computing on, OL6.15-8
- Cm*, OL6.15-4
- CMOS (complementary metal oxide semiconductor), 41
- Coarse-grained multithreading, 514
- Cobol, OL2.21-7
- Code generation, OL2.15-13
- Code motion, OL2.15-7
- Cold-start miss, 459
- Collision misses, 459
- Column major order, 413
- Combinational blocks, B-4
- Combinational control units, D-4–8
- Combinational elements, 248
- Combinational logic, 249, B-3, B-9–20
 - arrays, B-18–19
 - decoders, B-9
 - defined, B-5
 - don't cares, B-17–18
 - multiplexors, B-10
 - ROMs, B-14–16
 - two-level, B-11–14
 - Verilog, B-23–26

- Commercial computer development, OL1.12-4–1.12-10
- Commit units
 - buffer, 339–340
 - defined, 339–340
 - in update control, 343
- Common case fast, 11
- Common subexpression elimination, OL2.15-6
- Communication, 23–24
 - overhead, reducing, 44–45
 - thread, C-34
- Compact code, OL2.21-4
- Comparison instructions, A-57–59
 - floating-point, A-74–75
 - list of, A-57–59
- Comparisons, 93
 - constant operands in, 93
 - signed *versus* unsigned, 94–95
- Compilers, 123–124
 - branch creation, 92
 - brief history, OL2.21-9
 - conservative, OL2.15-6
 - defined, 14
 - front end, OL2.15-3
 - function, 14, 123–124, A-5–6
 - high-level optimizations, OL2.15-4
 - ILP exploitation, OL4.16-5
 - Just In Time (JIT), 132
 - machine language production, A-8–9, A-10
 - optimization, 141, OL2.21-9
 - speculation, 333–334
 - structure, OL2.15-2
- Compiling
 - C assignment statements, 65–66
 - C language, 92–93, 145, OL2.15-2–2.15-3
 - floating-point programs, 214–217
 - if-then-else, 91
 - in Java, OL2.15-19
 - procedures, 98, 101–102
 - recursive procedures, 101–102
 - while loops, 92–93
- Compressed sparse row (CSR) matrix, C-55, C-56
- Compulsory misses, 459
- Computer architects, 11–12
 - abstraction to simplify design, 11
 - common case fast, 11
 - dependability via redundancy, 12
 - hierarchy of memories, 12
 - Moore's law, 11
 - parallelism, 12
 - pipelining, 12
 - prediction, 12
- Computers
 - application classes, 5–6
 - applications, 4
 - arithmetic for, 176–236
 - characteristics, OL1.12-12
 - commercial development, OL1.12-4–1.12-10
 - component organization, 17
 - components, 17, 177
 - design measure, 53
 - desktop, 5
 - embedded, 5, A-7
 - first, OL1.12-2–1.12-4
 - in information revolution, 4
 - instruction representation, 80–87
 - performance measurement, OL1.12-10
 - PostPC Era, 6–7
 - principles, 86
 - servers, 5
- Condition field, 324
- Conditional branches
 - ARM, 147–148
 - changing program counter with, 324
 - compiling if-then-else into, 91
 - defined, 90
 - desktop RISC, E-16
 - embedded RISC, E-16
 - implementation, 96
 - in loops, 115
 - PA-RISC, E-34, E-35
 - PC-relative addressing, 114
 - RISC, E-10–16
 - SPARC, E-10–12
- Conditional move instructions, 324
- Conflict misses, 459
- Constant memory, C-40
- Constant operands, 72–73
 - in comparisons, 93
 - frequent occurrence, 72
- Constant-manipulating instructions, A-57
- Content Addressable Memory (CAM), 408
- Context switch, 446
- Control
 - ALU, 259–261
 - challenge, 325–326
 - finishing, 269–270
 - forwarding, 307
 - FSM, D-8–21
 - implementation, optimizing, D-27–28
 - for jump instruction, 270
 - mapping to hardware, D-2–32
 - memory, D-26
 - organizing, to reduce logic, D-31–32
 - pipelined, 300–303
- Control flow graphs, OL2.15-9–2.15-10
 - illustrated examples, OL2.15-9, OL2.15-10
- Control functions
 - ALU, mapping to gates, D-4–7
 - defining, 264
 - PLA, implementation, D-7, D-20–21
 - ROM, encoding, D-18–19
 - for single-cycle implementation, 269
- Control hazards, 281–282, 316–325
 - branch delay reduction, 318–319
 - branch not taken assumption, 318
 - branch prediction as solution, 284
 - delayed decision approach, 284
 - dynamic branch prediction, 321–323
 - logic implementation in Verilog, OL4.13-8
 - pipeline stalls as solution, 282
 - pipeline summary, 324
 - simplicity, 317
 - solutions, 282
 - static multiple-issue processors and, 335–336
- Control lines
 - asserted, 264
 - in datapath, 263
 - execution/address calculation, 300
 - final three stages, 303
 - instruction decode/register file read, 300
 - instruction fetch, 300
 - memory access, 302
 - setting of, 264
 - values, 300
 - write-back, 302
- Control signals
 - ALUOp, 263
 - defined, 250
 - effect of, 264
 - multi-bit, 264
 - pipelined datapaths with, 300–303
 - truth tables, D-14

- Control units, 247. *See also* Arithmetic logic unit (ALU)
 - address select logic, D-24, D-25
 - combinational, implementing, D-4–8
 - with explicit counter, D-23
 - illustrated, 265
 - logic equations, D-11
 - main, designing, 261–264
 - as microcode, D-28
 - MIPS, D-10
 - next-state outputs, D-10, D-12–13
 - output, 259–261, D-10
- Conversion instructions, A-75–76
- Cooperative thread arrays (CTAs), C-30
- Coprocessors, A-33–34
 - defined, 218
 - move instructions, A-71–72
- Core MIPS instruction set, 236. *See also* MIPS
 - abstract view, 246
 - desktop RISC, E-9–11
 - implementation, 244–248
 - implementation illustration, 247
 - overview, 245
 - subset, 244
- Cores
 - defined, 43
 - number per chip, 43
- Correlation predictor, 324
- Cosmic Cube, OL6.15-7
- Count register, A-34
- CPU, 9
- Cray computers, OL3.11-5–3.11-6
- Critical word first, 392
- Crossbar networks, 535
- CTSS (Compatible Time-Sharing System), OL5.18-9
- CUDA programming environment, 523, C-5
 - barrier synchronization, C-18, C-34
 - development, C-17, C-18
 - hierarchy of thread groups, C-18
 - kernels, C-19, C-24
 - key abstractions, C-18
 - paradigm, C-19–23
 - parallel plus-scan template, C-61
 - per-block shared memory, C-58
 - plus-reduction implementation, C-63
 - programs, C-6, C-24
 - scalable parallel programming with, C-17–23
 - shared memories, C-18
 - threads, C-36
- Cyclic redundancy check, 423
- Cylinder, 381
- D**
- D flip-flops, B-51, B-53
- D latches, B-51, B-52
- Data bits, 421
- Data flow analysis, OL2.15-11
- Data hazards, 278, 303–316. *See also* Hazards
 - forwarding, 278, 303–316
 - load-use, 280, 318
 - stalls and, 313–316
- Data layout directives, A-14
- Data movement instructions, A-70–73
- Data parallel problem decomposition, C-17, C-18
- Data race, 121
- Data segment, A-13
- Data selectors, 246
- Data transfer instructions. *See also* Instructions
 - defined, 68
 - load, 68
 - offset, 69
 - store, 71
- Datacenters, 7
- Data-level parallelism, 508
- Datapath elements
 - defined, 251
 - sharing, 256
- Datapaths
 - branch, 254
 - building, 251–259
 - control signal truth tables, D-14
 - control unit, 265
 - defined, 19
 - design, 251
 - exception handling, 329
 - for fetching instructions, 253
 - for hazard resolution via forwarding, 311
 - for jump instruction, 270
 - for memory instructions, 256
 - for MIPS architecture, 257
 - in operation for branch-on-equal instruction, 268
 - in operation for load instruction, 267
 - in operation for R-type instruction, 266
 - operation of, 264–269
 - pipelined, 286–303
 - for R-type instructions, 256, 264–265
 - single, creating, 256
 - single-cycle, 283
 - static two-issue, 336
- Deasserted signals, 250, B-4
- Debugging information, A-13
- DEC PDP-8, OL2.21-3
- Decimal numbers
 - binary number conversion to, 76
 - defined, 73
- Decision-making instructions, 90–96
- Decoders, B-9
 - two-level, B-65
- Decoding machine language, 118–120
- Defect, 26
- Delayed branches, 96. *See also* Branches
 - as control hazard solution, 284
 - defined, 255
 - embedded RISCs and, E-23
 - for five-stage pipelines, 26, 323–324
 - reducing, 318–319
 - scheduling limitations, 323
- Delayed decision, 284
- DeMorgan's theorems, B-11
- Denormalized numbers, 222
- Dependability via redundancy, 12
- Dependable memory hierarchy, 418–423
 - failure, defining, 418
- Dependencies
 - between pipeline registers, 308
 - between pipeline registers and ALU inputs, 308
 - bubble insertion and, 314
 - detection, 306–308
 - name, 338
 - sequence, 304
- Design
 - compromises and, 161
 - datapath, 251
 - digital, 354
 - logic, 248–251, B-1–79
 - main control unit, 261–264
 - memory hierarchy, challenges, 460
 - pipelining instruction sets, 277
 - Desktop and server RISCs. *See also* Reduced instruction set computer (RISC) architectures

- addressing modes, E-6
 - architecture summary, E-4
 - arithmetic/logical instructions, E-11
 - conditional branches, E-16
 - constant extension summary, E-9
 - control instructions, E-11
 - conventions equivalent to MIPS core, E-12
 - data transfer instructions, E-10
 - features added to, E-45
 - floating-point instructions, E-12
 - instruction formats, E-7
 - multimedia extensions, E-16–18
 - multimedia support, E-18
 - types of, E-3
 - Desktop computers, defined, 5
 - Device driver, OL6.9-5
 - DGEMM (Double precision General Matrix Multiply), 225, 352, 413, 553
 - cache blocked version of, 415
 - optimized C version of, 226, 227, 476
 - performance, 354, 416
 - Dicing, 27
 - Dies, 26, 26–27
 - Digital design pipeline, 354
 - Digital signal-processing (DSP) extensions, E-19
 - DIMMs (dual inline memory modules), OL5.17-5
 - Direct Data IO (DDIO), OL6.9-6
 - Direct memory access (DMA), OL6.9-4
 - Direct3D, C-13
 - Direct-mapped caches. *See also* Caches
 - address portions, 407
 - choice of, 456
 - defined, 384, 402
 - illustrated, 385
 - memory block location, 403
 - misses, 405
 - single comparator, 407
 - total number of bits, 390
 - Dirty bit, 437
 - Dirty pages, 437
 - Disk memory, 381–383
 - Displacement addressing, 116
 - Distributed Block-Interleaved Parity (RAID 5), OL5.11-6
 - div (Divide), A-52
 - div.d (FP Divide Double), A-76
 - div.s (FP Divide Single), A-76
 - Divide algorithm, 190
 - Dividend, 189
 - Division, 189–195
 - algorithm, 191
 - dividend, 189
 - divisor, 189
 - Divisor, 189
 - divu (Divide Unsigned), A-52. *See also*
 - Arithmetic
 - faster, 194
 - floating-point, 211, A-76
 - hardware, 189–192
 - hardware, improved version, 192
 - instructions, A-52–53
 - in MIPS, 194
 - operands, 189
 - quotient, 189
 - remainder, 189
 - signed, 192–194
 - SRT, 194
 - Don't cares, B-17–18
 - example, B-17–18
 - term, 261
 - Double data rate (DDR), 379
 - Double Data Rate RAMs (DDRRAMs), 379–380, B-65
 - Double precision. *See also* Single precision
 - defined, 198
 - FMA, C-45–46
 - GPU, C-45–46, C-74
 - representation, 201
 - Double words, 152
 - Dual inline memory modules (DIMMs), 381
 - Dynamic branch prediction, 321–323. *See also* Control hazards
 - branch prediction buffer, 321
 - loops and, 321–323
 - Dynamic hardware predictors, 284
 - Dynamic multiple-issue processors, 333, 339–341. *See also* Multiple issue
 - pipeline scheduling, 339–341
 - superscalar, 339
 - Dynamic pipeline scheduling, 339–341
 - commit unit, 339–340
 - concept, 339–340
 - hardware-based speculation, 341
 - primary units, 340
 - reorder buffer, 343
 - reservation station, 339–340
 - Dynamic random access memory (DRAM), 378, 379–381, B-63–65
 - bandwidth external to, 398
 - cost, 23
 - defined, 19, B-63
 - DIMM, OL5.17-5
 - Double Date Rate (DDR), 379–380
 - early board, OL5.17-4
 - GPU, C-37–38
 - growth of capacity, 25
 - history, OL5.17-2
 - internal organization of, 380
 - pass transistor, B-63
 - SIMM, OL5.17-5, OL5.17-6
 - single-transistor, B-64
 - size, 398
 - speed, 23
 - synchronous (SDRAM), 379–380, B-60, B-65
 - two-level decoder, B-65
 - Dynamically linked libraries (DLLs), 129–131
 - defined, 129
 - lazy procedure linkage version, 130
- ## E
- Early restart, 392
 - Edge-triggered clocking methodology, 249, 250, B-48, B-73
 - advantage, B-49
 - clocks, B-73
 - drawbacks, B-74
 - illustrated, B-50
 - rising edge/falling edge, B-48
 - EDSAC (Electronic Delay Storage Automatic Calculator), OL1.12-3, OL5.17-2
 - Eispack, OL3.11-4
 - Electrically erasable programmable read-only memory (EEPROM), 381
 - Elements
 - combinational, 248
 - datapath, 251, 256
 - memory, B-50–58
 - state, 248, 250, 252, B-48, B-50
 - Embedded computers, 5
 - application requirements, 6
 - defined, A-7
 - design, 5
 - growth, OL1.12-12–1.12-13
 - Embedded Microprocessor Benchmark Consortium (EEMBC), OL1.12-12

- Embedded RISCs. *See also* Reduced instruction set computer (RISC) architectures
- addressing modes, E-6
 - architecture summary, E-4
 - arithmetic/logical instructions, E-14
 - conditional branches, E-16
 - constant extension summary, E-9
 - control instructions, E-15
 - data transfer instructions, E-13
 - delayed branch and, E-23
 - DSP extensions, E-19
 - general purpose registers, E-5
 - instruction conventions, E-15
 - instruction formats, E-8
 - multiply-accumulate approaches, E-19
 - types of, E-4
- Encoding
- defined, D-31
 - floating-point instruction, 213
 - MIPS instruction, 83, 119, A-49
 - ROM control function, D-18–19
 - ROM logic function, B-15
 - x86 instruction, 155–156
- ENIAC (Electronic Numerical Integrator and Calculator), OL1.12-2, OL1.12-3, OL5.17-2
- EPIC, OL4.16-5
- Error correction, B-65–67
- Error Detecting and Correcting Code (RAID 2), OL5.11-5
- Error detection, B-66
- Error detection code, 420
- Ethernet, 23
- EX stage
- load instructions, 292
 - overflow exception detection, 328
 - store instructions, 294
- Exabyte, 6
- Exception enable, 447
- Exception handlers, A-36–38
- defined, A-35
 - return from, A-38
- Exception program counters (EPCs), 326
- address capture, 331
 - copying, 181
 - defined, 181, 327
 - in restart determination, 326–327
 - transferring, 182
- Exceptions, 325–332, A-33–38
- association, 331–332
 - datapath with controls for handling, 329
 - defined, 180, 326
 - detecting, 326
 - event types and, 326
 - imprecise, 331–332
 - instructions, A-80
 - interrupts *versus*, 325–326
 - in MIPS architecture, 326–327
 - overflow, 329
 - PC, 445, 446–447
 - pipelined computer example, 328
 - in pipelined implementation, 327–332
 - precise, 332
 - reasons for, 326–327
 - result due to overflow in add instruction, 330
 - saving/restoring stage on, 450
- Exclusive OR (XOR) instructions, A-57
- Executable files, A-4
- defined, 126
 - linker production, A-19
- Execute or address calculation stage, 292
- Execute/address calculation
- control line, 300
 - load instruction, 292
 - store instruction, 292
- Execution time
- as valid performance measure, 51
 - CPU, 32, 33–34
 - pipelining and, 286
- Explicit counters, D-23, D-26
- Exponents, 197–198
- External labels, A-10
- F**
- Facilities, A-14–17
- Failures, synchronizer, B-77
- Fallacies. *See also* Pitfalls
- add immediate unsigned, 227
 - Amdahl's law, 556
 - arithmetic, 229–232
 - assembly language for performance, 159–160
 - commercial binary compatibility importance, 160
 - defined, 49
 - GPUs, C-72–74, C-75
 - low utilization uses little power, 50
 - peak performance, 556
 - pipelining, 355–356
 - powerful instructions mean higher performance, 159
 - right shift, 229
- False sharing, 469
- Fast carry
- with “infinite” hardware, B-38–39
 - with first level of abstraction, B-39–40
 - with second level of abstraction, B-40–46
- Fast Fourier Transforms (FFT), C-53
- Fault avoidance, 419
- Fault forecasting, 419
- Fault tolerance, 419
- Fermi architecture, 523, 552
- Field programmable devices (FPDs), B-78
- Field programmable gate arrays (FPGAs), B-78
- Fields
- Cause register, A-34, A-35
 - defined, 82
 - format, D-31
 - MIPS, 82–83
 - names, 82
 - Status register, A-34, A-35
- Files, register, 252, 257, B-50, B-54–56
- Fine-grained multithreading, 514
- Finite-state machines (FSMs), 451–466, B-67–72
- control, D-8–22
 - controllers, 464
 - for multicycle control, D-9
 - for simple cache controller, 464–466
 - implementation, 463, B-70
 - Mealy, 463
 - Moore, 463
 - next-state function, 463, B-67
 - output function, B-67, B-69
 - state assignment, B-70
 - state register implementation, B-71
 - style of, 463
 - synchronous, B-67
 - SystemVerilog, OL5.12-7
 - traffic light example, B-68–70
- Flash memory, 381
- characteristics, 23
 - defined, 23
- Flat address space, 479
- Flip-flops
- D flip-flops, B-51, B-53
 - defined, B-51

- Floating point, 196–222, 224
 - assembly language, 212
 - backward step, OL3.11-4–3.11-5
 - binary to decimal conversion, 202
 - branch, 211
 - challenges, 232–233
 - diversity *versus* portability, OL3.11-3–3.11-4
 - division, 211
 - first dispute, OL3.11-2–3.11-3
 - form, 197
 - fused multiply add, 220
 - guard digits, 218–219
 - history, OL3.11-3
 - IEEE 754 standard, 198, 199
 - instruction encoding, 213
 - intermediate calculations, 218
 - machine language, 212
 - MIPS instruction frequency for, 236
 - MIPS instructions, 211–213
 - operands, 212
 - overflow, 198
 - packed format, 224
 - precision, 230
 - procedure with two-dimensional matrices, 215–217
 - programs, compiling, 214–217
 - registers, 217
 - representation, 197–202
 - rounding, 218–219
 - sign and magnitude, 197
 - SSE2 architecture, 224–225
 - subtraction, 211
 - underflow, 198
 - units, 219
 - in x86, 224
 - Floating vectors, OL3.11-3
 - Floating-point addition, 203–206
 - arithmetic unit block diagram, 207
 - binary, 204
 - illustrated, 205
 - instructions, 211, A-73–74
 - steps, 203–204
 - Floating-point arithmetic (GPUs), C-41–46
 - basic, C-42
 - double precision, C-45–46, C-74
 - performance, C-44
 - specialized, C-42–44
 - supported formats, C-42
 - texture operations, C-44
 - Floating-point instructions, A-73–80
 - absolute value, A-73
 - addition, A-73–74
 - comparison, A-74–75
 - conversion, A-75–76
 - desktop RISC, E-12
 - division, A-76
 - load, A-76–77
 - move, A-77–78
 - multiplication, A-78
 - negation, A-78–79
 - SPARC, E-31
 - square root, A-79
 - store, A-79
 - subtraction, A-79–80
 - truncation, A-80
 - Floating-point multiplication, 206–210
 - binary, 210–211
 - illustrated, 209
 - instructions, 211
 - significands, 206
 - steps, 206–210
 - Flow-sensitive information, OL2.15-15
 - Flushing instructions, 318, 319
 - defined, 319
 - exceptions and, 331
 - For loops, 141, OL2.15-26
 - inner, OL2.15-24
 - SIMD and, OL6.15-2
 - Formal parameters, A-16
 - Format fields, D-31
 - Fortran, OL2.21-7
 - Forward references, A-11
 - Forwarding, 303–316
 - ALU before, 309
 - control, 307
 - datapath for hazard resolution, 311
 - defined, 278
 - functioning, 306
 - graphical representation, 279
 - illustrations, OL4.13-26–4.13-26
 - multiple results and, 281
 - multiplexors, 310
 - pipeline registers before, 309
 - with two instructions, 278
 - Verilog implementation, OL4.13-2–4.13-4
 - Fractions, 197, 198
 - Frame buffer, 18
 - Frame pointers, 103
 - Front end, OL2.15-3
 - Fully associative caches. *See also* Caches
 - block replacement strategies, 457
 - choice of, 456
 - defined, 403
 - memory block location, 403
 - misses, 406
 - Fully connected networks, 535
 - Function code, 82
 - Fused-multiply-add (FMA) operation, 220, C-45–46
- ## G
- Game consoles, C-9
 - Gates, B-3, B-8
 - AND, B-12, D-7
 - delays, B-46
 - mapping ALU control function to, D-4–7
 - NAND, B-8
 - NOR, B-8, B-50
 - Gather-scatter, 511, 552
 - General Purpose GPUs (GPGPUs), C-5
 - General-purpose registers, 150
 - architectures, OL2.21-3
 - embedded RISCs, E-5
 - Generate
 - defined, B-40
 - example, B-44
 - super, B-41
 - Gigabyte, 6
 - Global common subexpression elimination, OL2.15-6
 - Global memory, C-21, C-39
 - Global miss rates, 416
 - Global optimization, OL2.15-5
 - code, OL2.15-7
 - implementing, OL2.15-8–2.15-11
 - Global pointers, 102
 - GPU computing. *See also* Graphics processing units (GPUs)
 - defined, C-5
 - visual applications, C-6–7
 - GPU system architectures, C-7–12
 - graphics logical pipeline, C-10
 - heterogeneous, C-7–9
 - implications for, C-24
 - interfaces and drivers, C-9
 - unified, C-10–12
 - Graph coloring, OL2.15-12

Graphics displays
 computer hardware support, 18
 LCD, 18

Graphics logical pipeline, C-10

Graphics processing units (GPUs), 522–529. *See also* GPU computing
 as accelerators, 522
 attribute interpolation, C-43–44
 defined, 46, 506, C-3
 evolution, C-5
 fallacies and pitfalls, C-72–75
 floating-point arithmetic, C-17, C-41–46, C-74
 GeForce 8-series generation, C-5
 general computation, C-73–74
 General Purpose (GPGPUs), C-5
 graphics mode, C-6
 graphics trends, C-4
 history, C-3–4
 logical graphics pipeline, C-13–14
 mapping applications to, C-55–72
 memory, 523
 multilevel caches and, 522
 N-body applications, C-65–72
 NVIDIA architecture, 523–526
 parallel memory system, C-36–41
 parallelism, 523, C-76
 performance doubling, C-4
 perspective, 527–529
 programming, C-12–24
 programming interfaces to, C-17
 real-time graphics, C-13
 summary, C-76

Graphics shader programs, C-14–15

Gresham's Law, 236, OL3.11-2

Grid computing, 533

Grids, C-19

GTX 280, 548–553

Guard digits
 defined, 218
 rounding with, 219

H

Half precision, C-42

Halfwords, 110

Hamming, Richard, 420

Hamming distance, 420

Hamming Error Correction Code (ECC), 420–421
 calculating, 420–421

Handlers
 defined, 449
 TLB miss, 448

Hard disks
 access times, 23
 defined, 23

Hardware
 as hierarchical layer, 13
 language of, 14–16
 operations, 63–66
 supporting procedures in, 96–106
 synthesis, B-21
 translating microprograms to, D-28–32
 virtualizable, 426

Hardware description languages. *See also*
 Verilog
 defined, B-20
 using, B-20–26
 VHDL, B-20–21

Hardware multithreading, 514–517
 coarse-grained, 514
 options, 516
 simultaneous, 515–517

Hardware-based speculation, 341

Harvard architecture, OL1.12-4

Hazard detection units, 313–314
 functions, 314
 pipeline connections for, 314

Hazards, 277–278. *See also* Pipelining
 control, 281–282, 316–325
 data, 278, 303–316
 forwarding and, 312
 structural, 277, 294

Heap
 allocating space on, 104–106
 defined, 104

Heterogeneous systems, C-4–5
 architecture, C-7–9
 defined, C-3

Hexadecimal numbers, 81–82
 binary number conversion to, 81–82

Hierarchy of memories, 12

High-level languages, 14–16, A-6
 benefits, 16
 computer architectures, OL2.21-5
 importance, 16

High-level optimizations, OL2.15-4–2.15-5

Hit rate, 376

Hit time
 cache performance and, 401–402

defined, 376

Hit under miss, 472

Hold time, B-54

Horizontal microcode, D-32

Hot-swapping, OL5.11-7

Human genome project, 4

I

I/O, A-38–40, OL6.9-2, OL6.9-3
 memory-mapped, A-38
 on system performance, OL5.11-2

I/O benchmarks. *See* Benchmarks

IBM 360/85, OL5.17-7

IBM 701, OL1.12-5

IBM 7030, OL4.16-2

IBM ALOG, OL3.11-7

IBM Blue Gene, OL6.15-9–6.15-10

IBM Personal Computer, OL1.12-7, OL2.21-6

IBM System/360 computers, OL1.12-6, OL3.11-6, OL4.16-2

IBM z/VM, OL5.17-8

ID stage
 branch execution in, 319
 load instructions, 292
 store instruction in, 291

IEEE 754 floating-point standard, 198, 199, OL3.11-8–3.11-10. *See also*
 Floating point
 first chips, OL3.11-8–3.11-9
 in GPU arithmetic, C-42–43
 implementation, OL3.11-10
 rounding modes, 219
 today, OL3.11-10

If statements, 114

I-format, 83

If-then-else, 91

Immediate addressing, 116

Immediate instructions, 72

Imprecise interrupts, 331, OL4.16-4

Index-out-of-bounds check, 94–95

Induction variable elimination, OL2.15-7

Inheritance, OL2.15-15

In-order commit, 341

Input devices, 16

Inputs, 261

Instances, OL2.15-15

Instruction count, 36, 38

Instruction decode/register file read stage

- control line, 300
- load instruction, 289
- store instruction, 294
- Instruction execution illustrations,
 - OL4.13-16-4.13-17
 - clock cycle 9, OL4.13-24
 - clock cycles 1 and 2, OL4.13-21
 - clock cycles 3 and 4, OL4.13-22
 - clock cycles 5 and 6, OL4.13-23, OL4.13-23
 - clock cycles 7 and 8, OL4.13-24
 - examples, OL4.13-20-4.13-25
 - forwarding, OL4.13-26-4.13-31
 - no hazard, OL4.13-17
 - pipelines with stalls and forwarding, OL4.13-26, OL4.13-20
- Instruction fetch stage
 - control line, 300
 - load instruction, 289
 - store instruction, 294
- Instruction formats, 157
 - ARM, 148
 - defined, 81
 - desktop/server RISC architectures, E-7
 - embedded RISC architectures, E-8
 - I-type, 83
 - J-type, 113
 - jump instruction, 270
 - MIPS, 148
 - R-type, 83, 261
 - x86, 157
- Instruction latency, 356
- Instruction mix, 39, OL1.12-10
- Instruction set architecture
 - ARM, 145-147
 - branch address calculation, 254
 - defined, 22, 52
 - history, 163
 - maintaining, 52
 - protection and, 427
 - thread, C-31-34
 - virtual machine support, 426-427
- Instruction sets, 235, C-49
 - ARM, 324
 - design for pipelining, 277
 - MIPS, 62, 161, 234
 - MIPS-32, 235
 - Pseudo MIPS, 233
 - x86 growth, 161
- Instruction-level parallelism (ILP), 354.
 - See also* Parallelism
- compiler exploitation, OL4.16-5-4.16-6
 - defined, 43, 333
 - exploitation, increasing, 343
 - and matrix multiply, 351-354
- Instructions, 60-164, E-25-27, E-40-42.
 - See also* Arithmetic instructions;
 - MIPS; Operands
 - add immediate, 72
 - addition, 180, A-51
 - Alpha, E-27-29
 - arithmetic-logical, 251, A-51-57
 - ARM, 145-147, E-36-37
 - assembly, 66
 - basic block, 93
 - branch, A-59-63
 - cache-aware, 482
 - comparison, A-57-59
 - conditional branch, 90
 - conditional move, 324
 - constant-manipulating, A-57
 - conversion, A-75-76
 - core, 233
 - data movement, A-70-73
 - data transfer, 68
 - decision-making, 90-96
 - defined, 14, 62
 - desktop RISC conventions, E-12
 - division, A-52-53
 - as electronic signals, 80
 - embedded RISC conventions, E-15
 - encoding, 83
 - exception and interrupt, A-80
 - exclusive OR, A-57
 - fetching, 253
 - fields, 80
 - floating-point (x86), 224
 - floating-point, 211-213, A-73-80
 - flushing, 318, 319, 331
 - immediate, 72
 - introduction to, 62-63
 - jump, 95, 97, A-63-64
 - left-to-right flow, 287-288
 - load, 68, A-66-68
 - load linked, 122
 - logical operations, 87-89
 - M32R, E-40
 - memory access, C-33-34
 - memory-reference, 245
 - multiplication, 188, A-53-54
 - negation, A-54
 - nop, 314
 - PA-RISC, E-34-36
 - performance, 35-36
 - pipeline sequence, 313
 - PowerPC, E-12-13, E-32-34
 - PTX, C-31, C-32
 - remainder, A-55
 - representation in computer, 80-87
 - restartable, 450
 - resuming, 450
 - R-type, 252
 - shift, A-55-56
 - SPARC, E-29-32
 - store, 71, A-68-70
 - store conditional, 122
 - subtraction, 180, A-56-57
 - SuperH, E-39-40
 - thread, C-30-31
 - Thumb, E-38
 - trap, A-64-66
 - vector, 510
 - as words, 62
 - x86, 149-155
- Instructions per clock cycle (IPC), 333
- Integrated circuits (ICs), 19. *See also*
 - specific chips
 - cost, 27
 - defined, 25
 - manufacturing process, 26
 - very large-scale (VLSIs), 25
- Intel Core i7, 46-49, 244, 501, 548-553
 - address translation for, 471
 - architectural registers, 347
 - caches in, 472
 - memory hierarchies of, 471-475
 - microarchitecture, 338
 - performance of, 473
 - SPEC CPU benchmark, 46-48
 - SPEC power benchmark, 48-49
 - TLB hardware for, 471
- Intel Core i7 920, 346-349
 - microarchitecture, 347
- Intel Core i7 960
 - benchmarking and rooflines of, 548-553
- Intel Core i7 Pipelines, 344, 346-349
 - memory components, 348
 - performance, 349-351
 - program performance, 351
 - specification, 345
- Intel IA-64 architecture, OL2.21-3
- Intel Paragon, OL6.15-8

- Intel Threading Building Blocks, C-60
 - Intel x86 microprocessors
 - clock rate and power for, 40
 - Interference graphs, OL2.15-12
 - Interleaving, 398
 - Interprocedural analysis, OL2.15-14
 - Interrupt enable, 447
 - Interrupt handlers, A-33
 - Interrupt-driven I/O, OL6.9-4
 - Interrupts
 - defined, 180, 326
 - event types and, 326
 - exceptions *versus*, 325–326
 - imprecise, 331, OL4.16-4
 - instructions, A-80
 - precise, 332
 - vectored, 327
 - Intrinsity FastMATH processor, 395–398
 - caches, 396
 - data miss rates, 397, 407
 - read processing, 442
 - TLB, 440
 - write-through processing, 442
 - Inverted page tables, 436
 - Issue packets, 334
- J**
- j (Jump), 64
 - jal (Jump And Link), 64
 - Java
 - bytecode, 131
 - bytecode architecture, OL2.15-17
 - characters in, 109–111
 - compiling in, OL2.15-19–2.15-20
 - goals, 131
 - interpreting, 131, 145, OL2.15-15–2.15-16
 - keywords, OL2.15-21
 - method invocation in, OL2.15-21
 - pointers, OL2.15-26
 - primitive types, OL2.15-26
 - programs, starting, 131–132
 - reference types, OL2.15-26
 - sort algorithms, 141
 - strings in, 109–111
 - translation hierarchy, 131
 - while loop compilation in, OL2.15-18–2.15-19
 - Java Virtual Machine (JVM), 145, OL2.15-16
 - jr (Jump Register), 64
 - J-type instruction format, 113
 - Jump instructions, 254, E-26
 - branch instruction *versus*, 270
 - control and datapath for, 271
 - implementing, 270
 - instruction format, 270
 - list of, A-63–64
 - Just In Time (JIT) compilers, 132, 560
- K**
- Karnaugh maps, B-18
 - Kernel mode, 444
 - Kernels
 - CUDA, C-19, C-24
 - defined, C-19
 - Kilobyte, 6
- L**
- Labels
 - global, A-10, A-11
 - local, A-11
 - LAPACK, 230
 - Large-scale multiprocessors, OL6.15-7, OL6.15-9–6.15-10
 - Latches
 - D latch, B-51, B-52
 - defined, B-51
 - Latency
 - instruction, 356
 - memory, C-74–75
 - pipeline, 286
 - use, 336–337
 - lbu (Load Byte Unsigned), 64
 - Leaf procedures. *See also* Procedures
 - defined, 100
 - example, 109
 - Least recently used (LRU)
 - as block replacement strategy, 457
 - defined, 409
 - pages, 434
 - Least significant bits, B-32
 - defined, 74
 - SPARC, E-31
 - Left-to-right instruction flow, 287–288
 - Level-sensitive clocking, B-74, B-75–76
 - defined, B-74
 - two-phase, B-75
 - lhu (Load Halfword Unsigned), 64
 - li (Load Immediate), 162
 - Link, OL6.9-2
 - Linkers, 126–129, A-18–19
 - defined, 126, A-4
 - executable files, 126, A-19
 - function illustration, A-19
 - steps, 126
 - using, 126–129
 - Linking object files, 126–129
 - Linpack, 538, OL3.11-4
 - Liquid crystal displays (LCDs), 18
 - LISP, SPARC support, E-30
 - Little-endian byte order, A-43
 - Live range, OL2.15-11
 - Livermore Loops, OL1.12-11
 - ll (Load Linked), 64
 - Load balancing, 505–506
 - Load instructions. *See also* Store instructions
 - access, C-41
 - base register, 262
 - block, 149
 - compiling with, 71
 - datapath in operation for, 267
 - defined, 68
 - details, A-66–68
 - EX stage, 292
 - floating-point, A-76–77
 - halfword unsigned, 110
 - ID stage, 291
 - IF stage, 291
 - linked, 122, 123
 - list of, A-66–68
 - load byte unsigned, 76
 - load half, 110
 - load upper immediate, 112, 113
 - MEM stage, 293
 - pipelined datapath in, 296
 - signed, 76
 - unit for implementing, 255
 - unsigned, 76
 - WB stage, 293
 - Load word, 68, 71
 - Loaders, 129
 - Loading, A-19–20
 - Load-store architectures, OL2.21-3
 - Load-use data hazard, 280, 318
 - Load-use stalls, 318
 - Local area networks (LANs), 24. *See also* Networks

- Local labels, A-11
 - Local memory, C-21, C-40
 - Local miss rates, 416
 - Local optimization, OL2.15-5.
 - See also Optimization implementing, OL2.15-8
 - Locality
 - principle, 374
 - spatial, 374, 377
 - temporal, 374, 377
 - Lock synchronization, 121
 - Locks, 518
 - Logic
 - address select, D-24, D-25
 - ALU control, D-6
 - combinational, 250, B-5, B-9–20
 - components, 249
 - control unit equations, D-11
 - design, 248–251, B-1–79
 - equations, B-7
 - minimization, B-18
 - programmable array (PAL), B-78
 - sequential, B-5, B-56–58
 - two-level, B-11–14
 - Logical operations, 87–89
 - AND, 88, A-52
 - ARM, 149
 - desktop RISC, E-11
 - embedded RISC, E-14
 - MIPS, A-51–57
 - NOR, 89, A-54
 - NOT, 89, A-55
 - OR, 89, A-55
 - shifts, 87
 - Long instruction word (LIW), OL4.16-5
 - Lookup tables (LUTs), B-79
 - Loop unrolling
 - defined, 338, OL2.15-4
 - for multiple-issue pipelines, 338
 - register renaming and, 338
 - Loops, 92–93
 - conditional branches in, 114
 - for, 141
 - prediction and, 321–323
 - test, 142, 143
 - while, compiling, 92–93
 - lui (Load Upper Imm.), 64
 - lw (Load Word), 64
 - lwc1 (Load FP Single), A-73
- ## M
- M32R, E-15, E-40
 - Machine code, 81
 - Machine instructions, 81
 - Machine language, 15
 - branch offset in, 115
 - decoding, 118–120
 - defined, 14, 81, A-3
 - floating-point, 212
 - illustrated, 15
 - MIPS, 85
 - SRAM, 21
 - translating MIPS assembly language into, 84
 - Macros
 - defined, A-4
 - example, A-15–17
 - use of, A-15
 - Main memory, 428. See also Memory
 - defined, 23
 - page tables, 437
 - physical addresses, 428
 - Mapping applications, C-55–72
 - Mark computers, OL1.12-14
 - Matrix multiply, 225–228, 553–555
 - Mealy machine, 463–464, B-68, B-71, B-72
 - Mean time to failure(MTTF), 418
 - improving, 419
 - versus AFR of disks, 419–420
 - Media Access Control (MAC) address, OL6.9-7
 - Megabyte, 6
 - Memory
 - addresses, 77
 - affinity, 545
 - atomic, C-21
 - bandwidth, 380–381, 397
 - cache, 21, 383–398, 398–417
 - CAM, 408
 - constant, C-40
 - control, D-26
 - defined, 19
 - DRAM, 19, 379–380, B-63–65
 - flash, 23
 - global, C-21, C-39
 - GPU, 523
 - instructions, datapath for, 256
 - layout, A-21
 - local, C-21, C-40
 - main, 23
 - nonvolatile, 22
 - operands, 68–69
 - parallel system, C-36–41
 - read-only (ROM), B-14–16
 - SDRAM, 379–380
 - secondary, 23
 - shared, C-21, C-39–40
 - spaces, C-39
 - SRAM, B-58–62
 - stalls, 400
 - technologies for building, 24–28
 - texture, C-40
 - usage, A-20–22
 - virtual, 427–454
 - volatile, 22
 - Memory access instructions, C-33–34
 - Memory access stage
 - control line, 302
 - load instruction, 292
 - store instruction, 292
 - Memory bandwidth, 551, 557
 - Memory consistency model, 469
 - Memory elements, B-50–58
 - clocked, B-51
 - D flip-flop, B-51, B-53
 - D latch, B-52
 - DRAMs, B-63–67
 - flip-flop, B-51
 - hold time, B-54
 - latch, B-51
 - setup time, B-53, B-54
 - SRAMs, B-58–62
 - unclocked, B-51
 - Memory hierarchies, 545
 - of ARM cortex-A8, 471–475
 - block (or line), 376
 - cache performance, 398–417
 - caches, 383–417
 - common framework, 454–461
 - defined, 375
 - design challenges, 461
 - development, OL5.17-6–5.17-8
 - exploiting, 372–498
 - of Intel core i7, 471–475
 - level pairs, 376
 - multiple levels, 375
 - overall operation of, 443–444
 - parallelism and, 466–470, OL5.11-2
 - pitfalls, 478–482
 - program execution time and, 417

- Memory hierarchies (*Continued*)
 - quantitative design parameters, 454
 - redundant arrays and inexpensive disks, 470
 - reliance on, 376
 - structure, 375
 - structure diagram, 378
 - variance, 417
 - virtual memory, 427–454
- Memory rank, 381
- Memory technologies, 378–383
 - disk memory, 381–383
 - DRAM technology, 378, 379–381
 - flash memory, 381
 - SRAM technology, 378, 379
- Memory-mapped I/O, OL6.9-3
 - use of, A-38
- Memory-stall clock cycles, 399
- Message passing
 - defined, 529
 - multiprocessors, 529–534
- Metastability, B-76
- Methods
 - defined, OL2.15-5
 - invoking in Java, OL2.15-20–2.15-21
 - static, A-20
- mfc0 (Move From Control), A-71
- mfhi (Move From Hi), A-71
- mflo (Move From Lo), A-71
- Microarchitectures, 347
 - Intel Core i7 920, 347
- Microcode
 - assembler, D-30
 - control unit as, D-28
 - defined, D-27
 - dispatch ROMs, D-30–31
 - horizontal, D-32
 - vertical, D-32
- Microinstructions, D-31
- Microprocessors
 - design shift, 501
 - multicore, 8, 43, 500–501
- Microprograms
 - as abstract control representation, D-30
 - field translation, D-29
 - translating to hardware, D-28–32
- Migration, 467
- Million instructions per second (MIPS), 51
- Minterms
 - defined, B-12, D-20
 - in PLA implementation, D-20
- MIP-map, C-44
- MIPS, 64, 84, A-45–80
 - addressing for 32-bit immediates, 116–118
 - addressing modes, A-45–47
 - arithmetic core, 233
 - arithmetic instructions, 63, A-51–57
 - ARM similarities, 146
 - assembler directive support, A-47–49
 - assembler syntax, A-47–49
 - assembly instruction, mapping, 80–81
 - branch instructions, A-59–63
 - comparison instructions, A-57–59
 - compiling C assignment statements into, 65
 - compiling complex C assignment into, 65–66
 - constant-manipulating instructions, A-57
 - control registers, 448
 - control unit, D-10
 - CPU, A-46
 - divide in, 194
 - exceptions in, 326–327
 - fields, 82–83
 - floating-point instructions, 211–213
 - FPU, A-46
 - instruction classes, 163
 - instruction encoding, 83, 119, A-49
 - instruction formats, 120, 148, A-49–51
 - instruction set, 62, 162, 234
 - jump instructions, A-63–66
 - logical instructions, A-51–57
 - machine language, 85
 - memory addresses, 70
 - memory allocation for program and data, 104
 - multiply in, 188
 - opcode map, A-50
 - operands, 64
 - Pseudo, 233, 235
 - register conventions, 105
 - static multiple issue with, 335–338
- MIPS core
 - architecture, 195
 - arithmetic/logical instructions not in, E-21, E-23
 - common extensions to, E-20–25
 - control instructions not in, E-21
 - data transfer instructions not in, E-20, E-22
 - floating-point instructions not in, E-22
 - instruction set, 233, 244–248, E-9–10
- MIPS-16
 - 16-bit instruction set, E-41–42
 - immediate fields, E-41
 - instructions, E-40–42
 - MIPS core instruction changes, E-42
 - PC-relative addressing, E-41
- MIPS-32 instruction set, 235
- MIPS-64 instructions, E-25–27
 - conditional procedure call instructions, E-27
 - constant shift amount, E-25
 - jump/call not PC-relative, E-26
 - move to/from control registers, E-26
 - nonaligned data transfers, E-25
 - NOR, E-25
 - parallel single precision floating-point operations, E-27
 - reciprocal and reciprocal square root, E-27
 - SYSCALL, E-25
 - TLB instructions, E-26–27
- Mirroring, OL5.11-5
- Miss penalty
 - defined, 376
 - determination, 391–392
 - multilevel caches, reducing, 410
- Miss rates
 - block size *versus*, 392
 - data cache, 455
 - defined, 376
 - global, 416
 - improvement, 391–392
 - Intrinsity FastMATH processor, 397
 - local, 416
 - miss sources, 460
 - split cache, 397
- Miss under miss, 472
- MMX (MultiMedia eXtension), 224
- Modules, A-4
- Moore machines, 463–464, B-68, B-71, B-72
- Moore's law, 11, 379, 522, OL6.9-2, C-72–73
- Most significant bit
 - 1-bit ALU for, B-33
 - defined, 74
 - move (Move), 139

- Move instructions, A-70–73
 coprocessor, A-71–72
 details, A-70–73
 floating-point, A-77–78
- MS-DOS, OL5.17-11
- mul.d (FP Multiply Double), A-78
- mul.s (FP Multiply Single), A-78
- mult (Multiply), A-53
- Multicore, 517–521
- Multicore multiprocessors, 8, 43
 defined, 8, 500–501
- MULTICS (Multiplexed Information and Computing Service), OL5.17-9–5.17-10
- Multilevel caches. *See also* Caches
 complications, 416
 defined, 398, 416
 miss penalty, reducing, 410
 performance of, 410
 summary, 417–418
- Multimedia extensions
 desktop/server RISCs, E-16–18
 as SIMD extensions to instruction sets, OL6.15-4
 vector *versus*, 511–512
- Multiple dimension arrays, 218
- Multiple instruction multiple data (MIMD), 558
 defined, 507, 508
 first multiprocessor, OL6.15-14
- Multiple instruction single data (MISD), 507
- Multiple issue, 332–339
 code scheduling, 337–338
 dynamic, 333, 339–341
 issue packets, 334
 loop unrolling and, 338
 processors, 332, 333
 static, 333, 334–339
 throughput and, 342
- Multiple processors, 553–555
- Multiple-clock-cycle pipeline diagrams, 296–297
 five instructions, 298
 illustrated, 298
- Multiplexors, B-10
 controls, 463
 in datapath, 263
 defined, 246
 forwarding, control values, 310
 selector control, 256–257
 two-input, B-10
- Multiplicand, 183
- Multiplication, 183–188. *See also* Arithmetic
 fast, hardware, 188
 faster, 187–188
 first algorithm, 185
 floating-point, 206–208, A-78
 hardware, 184–186
 instructions, 188, A-53–54
 in MIPS, 188
 multiplicand, 183
 multiplier, 183
 operands, 183
 product, 183
 sequential version, 184–186
 signed, 187
- Multiplier, 183
- Multiply algorithm, 186
- Multiply-add (MAD), C-42
- Multiprocessors
 benchmarks, 538–540
 bus-based coherent, OL6.15-7
 defined, 500
 historical perspective, 561
 large-scale, OL6.15-7–6.15-8, OL6.15-9–6.15-10
 message-passing, 529–534
 multithreaded architecture, C-26–27, C-35–36
 organization, 499, 529
 for performance, 559
 shared memory, 501, 517–521
 software, 500
 TFLOPS, OL6.15-6
 UMA, 518
- Multistage networks, 535
- Multithreaded multiprocessor
 architecture, C-25–36
 conclusion, C-36
 ISA, C-31–34
 massive multithreading, C-25–26
 multiprocessor, C-26–27
 multiprocessor comparison, C-35–36
 SIMT, C-27–30
 special function units (SFUs), C-35
 streaming processor (SP), C-34
 thread instructions, C-30–31
 threads/thread blocks management, C-30
- Multithreading, C-25–26
 coarse-grained, 514
 defined, 506
 fine-grained, 514
 hardware, 514–517
 simultaneous (SMT), 515–517
- multu (Multiply Unsigned), A-54
- Must-information, OL2.15-5
- Mutual exclusion, 121
- ## N
- Name dependence, 338
- NAND gates, B-8
- NAS (NASA Advanced Supercomputing), 540
- N-body
 all-pairs algorithm, C-65
 GPU simulation, C-71
 mathematics, C-65–67
 multiple threads per body, C-68–69
 optimization, C-67
 performance comparison, C-69–70
 results, C-70–72
 shared memory use, C-67–68
- Negation instructions, A-54, A-78–79
- Negation shortcut, 76
- Nested procedures, 100–102
 compiling recursive procedure showing, 101–102
- NetFPGA 10-Gigabit Ethernet card, OL6.9-2, OL6.9-3
- Network of Workstations, OL6.15-8–6.15-9
- Network topologies, 534–537
 implementing, 536
 multistage, 537
- Networking, OL6.9-4
 operating system in, OL6.9-4–6.9-5
 performance improvement, OL6.9-7–6.9-10
- Networks, 23–24
 advantages, 23
 bandwidth, 535
 crossbar, 535
 fully connected, 535
 local area (LANs), 24
 multistage, 535
 wide area (WANs), 24
- Newton's iteration, 218
- Next state
 nonsequential, D-24
 sequential, D-23

- Next-state function, 463, B-67
 - defined, 463
 - implementing, with sequencer, D-22–28
 - Next-state outputs, D-10, D-12–13
 - example, D-12–13
 - implementation, D-12
 - logic equations, D-12–13
 - truth tables, D-15
 - No Redundancy (RAID 0), OL5.11-4
 - No write allocation, 394
 - Nonblocking assignment, B-24
 - Nonblocking caches, 344, 472
 - Nonuniform memory access (NUMA), 518
 - Nonvolatile memory, 22
 - Nops, 314
 - nor (NOR), 64
 - NOR gates, B-8
 - cross-coupled, B-50
 - D latch implemented with, B-52
 - NOR operation, 89, A-54, E-25
 - NOT operation, 89, A-55, B-6
 - Numbers
 - binary, 73
 - computer *versus* real-world, 221
 - decimal, 73, 76
 - denormalized, 222
 - hexadecimal, 81–82
 - signed, 73–78
 - unsigned, 73–78
 - NVIDIA GeForce 8800, C-46–55
 - all-pairs N-body algorithm, C-71
 - dense linear algebra computations, C-51–53
 - FFT performance, C-53
 - instruction set, C-49
 - performance, C-51
 - rasterization, C-50
 - ROP, C-50–51
 - scalability, C-51
 - sorting performance, C-54–55
 - special function approximation
 - statistics, C-43
 - special function unit (SFU), C-50
 - streaming multiprocessor (SM), C-48–49
 - streaming processor, C-49–50
 - streaming processor array (SPA), C-46
 - texture/processor cluster (TPC), C-47–48
 - NVIDIA GPU architecture, 523–526
 - NVIDIA GTX 280, 548–553
 - NVIDIA Tesla GPU, 548–553
- O**
- Object files, 125, A-4
 - debugging information, 124
 - defined, A-10
 - format, A-13–14
 - header, 125, A-13
 - linking, 126–129
 - relocation information, 125
 - static data segment, 125
 - symbol table, 125, 126
 - text segment, 125
 - Object-oriented languages. *See also* Java
 - brief history, OL2.21-8
 - defined, 145, OL2.15-5
 - One's complement, 79, B-29
 - Opcodes
 - control line setting and, 264
 - defined, 82, 262
 - OpenGL, C-13
 - OpenMP (Open MultiProcessing), 520, 540
 - Operands, 66–73. *See also* Instructions
 - 32-bit immediate, 112–113
 - adding, 179
 - arithmetic instructions, 66
 - compiling assignment when in
 - memory, 69
 - constant, 72–73
 - division, 189
 - floating-point, 212
 - memory, 68–69
 - MIPS, 64
 - multiplication, 183
 - shifting, 148
 - Operating systems
 - brief history, OL5.17-9–5.17-12
 - defined, 13
 - encapsulation, 22
 - in networking, OL6.9-4–6.9-5
 - Operations
 - atomic, implementing, 121
 - hardware, 63–66
 - logical, 87–89
 - x86 integer, 152, 154–155
 - Optimization
 - class explanation, OL2.15-14
 - compiler, 141
 - control implementation, D-27–28
 - global, OL2.15-5
 - high-level, OL2.15-4–2.15-5
 - local, OL2.15-5, OL2.15-8
 - manual, 144
 - or (OR), 64
 - OR operation, 89, A-55, B-6
 - ori (Or Immediate), 64
 - Out-of-order execution
 - defined, 341
 - performance complexity, 416
 - processors, 344
 - Output devices, 16
 - Overflow
 - defined, 74, 198
 - detection, 180
 - exceptions, 329
 - floating-point, 198
 - occurrence, 75
 - saturation and, 181
 - subtraction, 179
- P**
- P+Q redundancy (RAID 6), OL5.11-7
 - Packed floating-point format, 224
 - Page faults, 434. *See also* Virtual memory
 - for data access, 450
 - defined, 428
 - handling, 429, 446–453
 - virtual address causing, 449, 450
 - Page tables, 456
 - defined, 432
 - illustrated, 435
 - indexing, 432
 - inverted, 436
 - levels, 436–437
 - main memory, 437
 - register, 432
 - storage reduction techniques, 436–437
 - updating, 432
 - VMM, 452
 - Pages. *See also* Virtual memory
 - defined, 428
 - dirty, 437
 - finding, 432–434
 - LRU, 434
 - offset, 429
 - physical number, 429
 - placing, 432–434

- size, 430
- virtual number, 429
- Parallel bus, OL6.9-3
- Parallel execution, 121
- Parallel memory system, C-36–41. *See also* Graphics processing units (GPUs)
 - caches, C-38
 - constant memory, C-40
 - DRAM considerations, C-37–38
 - global memory, C-39
 - load/store access, C-41
 - local memory, C-40
 - memory spaces, C-39
 - MMU, C-38–39
 - ROP, C-41
 - shared memory, C-39–40
 - surfaces, C-41
 - texture memory, C-40
- Parallel processing programs, 502–507
 - creation difficulty, 502–507
 - defined, 501
 - for message passing, 519–520
 - great debates in, OL6.15-5
 - for shared address space, 519–520
 - use of, 559
- Parallel reduction, C-62
- Parallel scan, C-60–63
 - CUDA template, C-61
 - inclusive, C-60
 - tree-based, C-62
- Parallel software, 501
- Parallelism, 12, 43, 332–344
 - and computers arithmetic, 222–223
 - data-level, 233, 508
 - debates, OL6.15-5–6.15-7
 - GPUs and, 523, C-76
 - instruction-level, 43, 332, 343
 - memory hierarchies and, 466–470, OL5.11-2
 - multicore and, 517
 - multiple issue, 332–339
 - multithreading and, 517
 - performance benefits, 44–45
 - process-level, 500
 - redundant arrays and inexpensive disks, 470
 - subword, E-17
 - task, C-24
 - task-level, 500
 - thread, C-22
- Paravirtualization, 482
- PA-RISC, E-14, E-17
 - branch vectored, E-35
 - conditional branches, E-34, E-35
 - debug instructions, E-36
 - decimal operations, E-35
 - extract and deposit, E-35
 - instructions, E-34–36
 - load and clear instructions, E-36
 - multiply/add and multiply/subtract, E-36
 - nullification, E-34
 - nullifying branch option, E-25
 - store bytes short, E-36
 - synthesized multiply and divide, E-34–35
- Parity, OL5.11-5
 - bits, 421
 - code, 420, B-65
- PARSEC (Princeton Application Repository for Shared Memory Computers), 540
- Pass transistor, B-63
- PCI-Express (PCIe), 537, C-8, OL6.9-2
- PC-relative addressing, 114, 116
- Peak floating-point performance, 542
- Pentium bug morality play, 231–232
- Performance, 28–36
 - assessing, 28
 - classic CPU equation, 36–40
 - components, 38
 - CPU, 33–35
 - defining, 29–32
 - equation, using, 36
 - improving, 34–35
 - instruction, 35–36
 - measuring, 33–35, OL1.12-10
 - program, 39–40
 - ratio, 31
 - relative, 31–32
 - response time, 30–31
 - sorting, C-54–55
 - throughput, 30–31
 - time measurement, 32
- Personal computers (PCs), 7
 - defined, 5
- Personal mobile device (PMD)
 - defined, 7
- Petabyte, 6
- Physical addresses, 428
 - mapping to, 428–429
 - space, 517, 521
- Physically addressed caches, 443
- Pipeline registers
 - before forwarding, 309
 - dependences, 308
 - forwarding unit selection, 312
- Pipeline stalls, 280
 - avoiding with code reordering, 280
 - data hazards and, 313–316
 - insertion, 315
 - load-use, 318
 - as solution to control hazards, 282
- Pipelined branches, 319
- Pipelined control, 300–303. *See also* Control
 - control lines, 300, 303
 - overview illustration, 316
 - specifying, 300
- Pipelined datapaths, 286–303
 - with connected control signals, 304
 - with control signals, 300–303
 - corrected, 296
 - illustrated, 289
 - in load instruction stages, 296
- Pipelined dependencies, 305
- Pipelines
 - branch instruction impact, 317
 - effectiveness, improving, OL4.16-4–4.16-5
 - execute and address calculation stage, 290, 292
 - five-stage, 274, 290, 299
 - graphic representation, 279, 296–300
 - instruction decode and register file read stage, 289, 292
 - instruction fetch stage, 290, 292
 - instructions sequence, 313
 - latency, 286
 - memory access stage, 290, 292
 - multiple-clock-cycle diagrams, 296–297
 - performance bottlenecks, 343
 - single-clock-cycle diagrams, 296–297
 - stages, 274
 - static two-issue, 335
 - write-back stage, 290, 294
- Pipelining, 12, 272–286
 - advanced, 343–344
 - benefits, 272
 - control hazards, 281–282
 - data hazards, 278

- Pipelining (*Continued*)
 - exceptions and, 327–332
 - execution time and, 286
 - fallacies, 355–356
 - hazards, 277–278
 - instruction set design for, 277
 - laundry analogy, 273
 - overview, 272–286
 - paradox, 273
 - performance improvement, 277
 - pitfall, 355–356
 - simultaneous executing instructions, 286
 - speed-up formula, 273
 - structural hazards, 277, 294
 - summary, 285
 - throughput and, 286
- Pitfalls. *See also* Fallacies
 - address space extension, 479
 - arithmetic, 229–232
 - associativity, 479
 - defined, 49
 - GPUs, C-74–75
 - ignoring memory system behavior, 478
 - memory hierarchies, 478–482
 - out-of-order processor evaluation, 479
 - performance equation subset, 50–51
 - pipelining, 355–356
 - pointer to automatic variables, 160
 - sequential word addresses, 160
 - simulating cache, 478
 - software development with multiprocessors, 556
 - VMM implementation, 481, 481–482
- Pixel shader example, C-15–17
- Pixels, 18
- Pointers
 - arrays *versus*, 141–145
 - frame, 103
 - global, 102
 - incrementing, 143
 - Java, OL2.15-26
 - stack, 98, 102
- Polling, OL6.9-8
- Pop, 98
- Power
 - clock rate and, 40
 - critical nature of, 53
 - efficiency, 343–344
 - relative, 41
- PowerPC
 - algebraic right shift, E-33
 - branch registers, E-32–33
 - condition codes, E-12
 - instructions, E-12–13
 - instructions unique to, E-31–33
 - load multiple/store multiple, E-33
 - logical shifted immediate, E-33
 - rotate with mask, E-33
- Precise interrupts, 332
- Prediction, 12
 - 2-bit scheme, 322
 - accuracy, 321, 324
 - dynamic branch, 321–323
 - loops and, 321–323
 - steady-state, 321
- Prefetching, 482, 544
- Primitive types, OL2.15-26
- Procedure calls
 - convention, A-22–33
 - examples, A-27–33
 - frame, A-23
 - preservation across, 102
- Procedures, 96–106
 - compiling, 98
 - compiling, showing nested procedure linking, 101–102
 - execution steps, 96
 - frames, 103
 - leaf, 100
 - nested, 100–102
 - recursive, 105, A-26–27
 - for setting arrays to zero, 142
 - sort, 135–139
 - strcpy, 108–109
 - string copy, 108–109
 - swap, 133
- Process identifiers, 446
- Process-level parallelism, 500
- Processors, 242–356
 - as cores, 43
 - control, 19
 - datapath, 19
 - defined, 17, 19
 - dynamic multiple-issue, 333
 - multiple-issue, 333
 - out-of-order execution, 344, 416
 - performance growth, 44
 - ROP, C-12, C-41
 - speculation, 333–334
 - static multiple-issue, 333, 334–339
 - streaming, C-34
 - superscalar, 339, 515–516, OL4.16-5
 - technologies for building, 24–28
 - two-issue, 336–337
 - vector, 508–510
 - VLIW, 335
- Product, 183
- Product of sums, B-11
- Program counters (PCs), 251
 - changing with conditional branch, 324
 - defined, 98, 251
 - exception, 445, 447
 - incrementing, 251, 253
 - instruction updates, 289
- Program libraries, A-4
- Program performance
 - elements affecting, 39
 - understanding, 9
- Programmable array logic (PAL), B-78
- Programmable logic arrays (PLAs)
 - component dots illustration, B-16
 - control function implementation, D-7, D-20–21
 - defined, B-12
 - example, B-13–14
 - illustrated, B-13
 - ROMs and, B-15–16
 - size, D-20
 - truth table implementation, B-13
- Programmable logic devices (PLDs), B-78
- Programmable ROMs (PROMs), B-14
- Programming languages. *See also* specific languages
 - brief history of, OL2.21-7–2.21-8
 - object-oriented, 145
 - variables, 67
- Programs
 - assembly language, 123
 - Java, starting, 131–132
 - parallel processing, 502–507
 - starting, 123–132
 - translating, 123–132
- Propagate
 - defined, B-40
 - example, B-44
 - super, B-41
- Protected keywords, OL2.15-21
- Protection
 - defined, 428
 - implementing, 444–446
 - mechanisms, OL5.17-9
 - VMs for, 424
- Protection group, OL5.11-5
- Pseudo MIPS
 - defined, 233

- instruction set, 235
 - Pseudodirect addressing, 116
 - Pseudoinstructions
 - defined, 124
 - summary, 125
 - Pthreads (POSIX threads), 540
 - PTX instructions, C-31, C-32
 - Public keywords, OL2.15-21
 - Push
 - defined, 98
 - using, 100
- Q**
- Quad words, 154
 - Quicksort, 411, 412
 - Quotient, 189
- R**
- Race, B-73
 - Radix sort, 411, 412, C-63–65
 - CUDA code, C-64
 - implementation, C-63–65
 - RAID, *See* Redundant arrays of inexpensive disks (RAID)
 - RAM, 9
 - Raster operation (ROP) processors, C-12, C-41, C-50–51
 - fixed function, C-41
 - Raster refresh buffer, 18
 - Rasterization, C-50
 - Ray casting (RC), 552
 - Read-only memories (ROMs), B-14–16
 - control entries, D-16–17
 - control function encoding, D-18–19
 - dispatch, D-25
 - implementation, D-15–19
 - logic function encoding, B-15
 - overhead, D-18
 - PLAs and, B-15–16
 - programmable (PROM), B-14
 - total size, D-16
 - Read-stall cycles, 399
 - Read-write head, 381
 - Receive message routine, 529
 - Receiver Control register, A-39
 - Receiver Data register, A-38, A-39
 - Recursive procedures, 105, A-26–27. *See also* Procedures
 - clone invocation, 100
 - stack in, A-29–30
 - Reduced instruction set computer (RISC)
 - architectures, E-2–45, OL2.21-5, OL4.16-4. *See also* Desktop and server RISCs; Embedded RISCs
 - group types, E-3–4
 - instruction set lineage, E-44
 - Reduction, 519
 - Redundant arrays of inexpensive disks (RAID), OL5.11-2–5.11-8
 - history, OL5.11-8
 - RAID 0, OL5.11-4
 - RAID 1, OL5.11-5
 - RAID 2, OL5.11-5
 - RAID 3, OL5.11-5
 - RAID 4, OL5.11-5–5.11-6
 - RAID 5, OL5.11-6–5.11-7
 - RAID 6, OL5.11-7
 - spread of, OL5.11-6
 - summary, OL5.11-7–5.11-8
 - use statistics, OL5.11-7
 - Reference bit, 435
 - References
 - absolute, 126
 - forward, A-11
 - types, OL2.15-26
 - unresolved, A-4, A-18
 - Register addressing, 116
 - Register allocation, OL2.15-11–2.15-13
 - Register files, B-50, B-54–56
 - defined, 252, B-50, B-54
 - in behavioral Verilog, B-57
 - single, 257
 - two read ports implementation, B-55
 - with two read ports/one write port, B-55
 - write port implementation, B-56
 - Register-memory architecture, OL2.21-3
 - Registers, 152, 153–154
 - architectural, 325–332
 - base, 69
 - callee-saved, A-23
 - caller-saved, A-23
 - Cause, A-35
 - clock cycle time and, 67
 - compiling C assignment with, 67–68
 - Count, A-34
 - defined, 66
 - destination, 83, 262
 - floating-point, 217
 - left half, 290
 - mapping, 80
 - MIPS conventions, 105
 - number specification, 252
 - page table, 432
 - pipeline, 308, 309, 312
 - primitives, 66
 - Receiver Control, A-39
 - Receiver Data, A-38, A-39
 - renaming, 338
 - right half, 290
 - spilling, 71
 - Status, 327, A-35
 - temporary, 67, 99
 - Transmitter Control, A-39–40
 - Transmitter Data, A-40
 - usage convention, A-24
 - use convention, A-22
 - variables, 67
- Relative performance, 31–32
 - Relative power, 41
 - Reliability, 418
 - Relocation information, A-13, A-14
 - Remainder
 - defined, 189
 - instructions, A-55
 - Reorder buffers, 343
 - Replication, 468
 - Requested word first, 392
 - Request-level parallelism, 532
 - Reservation stations
 - buffering operands in, 340–341
 - defined, 339–340
 - Response time, 30–31
 - Restartable instructions, 448
 - Return address, 97
 - Return from exception (ERET), 445
 - R-format, 262
 - ALU operations, 253
 - defined, 83
 - Ripple carry
 - adder, B-29
 - carry lookahead speed *versus*, B-46
 - Roofline model, 542–543, 544, 545
 - with ceilings, 546, 547
 - computational roofline, 545
 - illustrated, 542
 - Opteron generations, 543, 544
 - with overlapping areas shaded, 547
 - peak floating-point performance, 542
 - peak memory performance, 543
 - with two kernels, 547
 - Rotational delay. *See* Rotational latency
 - Rotational latency, 383

- Rounding, 218
 - accurate, 218
 - bits, 220
 - with guard digits, 219
 - IEEE 754 modes, 219
- Row-major order, 217, 413
- R-type instructions, 252
 - datapath for, 264–265
 - datapath in operation for, 266
- S**
- Saturation, 181
- sb (Store Byte), 64
- sc (Store Conditional), 64
- SCALAPAK, 230
- Scaling
 - strong, 505, 507
 - weak, 505
- Scientific notation
 - adding numbers in, 203
 - defined, 196
 - for reals, 197
- Search engines, 4
- Secondary memory, 23
- Sectors, 381
- Seek, 382
- Segmentation, 431
- Selector values, B-10
- Semiconductors, 25
- Send message routine, 529
- Sensitivity list, B-24
- Sequencers
 - explicit, D-32
 - implementing next-state function with, D-22–28
- Sequential logic, B-5
- Servers, OL5. *See also* Desktop and server RISCs
 - cost and capability, 5
- Service accomplishment, 418
- Service interruption, 418
- Set instructions, 93
- Set-associative caches, 403. *See also* Caches
 - address portions, 407
 - block replacement strategies, 457
 - choice of, 456
 - four-way, 404, 407
 - memory-block location, 403
 - misses, 405–406
 - n*-way, 403
 - two-way, 404
- Setup time, B-53, B-54
- sh (Store Halfword), 64
- Shaders
 - defined, C-14
 - floating-point arithmetic, C-14
 - graphics, C-14–15
 - pixel example, C-15–17
- Shading languages, C-14
- Shadowing, OL5.11-5
- Shared memory. *See also* Memory
 - as low-latency memory, C-21
 - caching in, C-58–60
 - CUDA, C-58
 - N-body and, C-67–68
 - per-CTA, C-39
 - SRAM banks, C-40
- Shared memory multiprocessors (SMP), 517–521
 - defined, 501, 517
 - single physical address space, 517
 - synchronization, 518
- Shift amount, 82
- Shift instructions, 87, A-55–56
- Sign and magnitude, 197
- Sign bit, 76
- Sign extension, 254
 - defined, 76
 - shortcut, 78
- Signals
 - asserted, 250, B-4
 - control, 250, 263–264
 - deasserted, 250, B-4
- Signed division, 192–194
- Signed multiplication, 187
- Signed numbers, 73–78
 - sign and magnitude, 75
 - treating as unsigned, 94–95
- Significands, 198
 - addition, 203
 - multiplication, 206
- Silicon, 25
 - as key hardware technology, 53
 - crystal ingot, 26
 - defined, 26
 - wafers, 26
- Silicon crystal ingot, 26
- SIMD (Single Instruction Multiple Data), 507–508, 558
 - computers, OL6.15-2–6.15-4
 - data vector, C-35
 - extensions, OL6.15-4
 - for loops and, OL6.15-3
 - massively parallel multiprocessors, OL6.15-2
 - small-scale, OL6.15-4
 - vector architecture, 508–510
 - in x86, 508
- SIMMs (single inline memory modules), OL5.17-5, OL5.17-6
- Simple programmable logic devices (SPLDs), B-78
- Simplicity, 161
- Simultaneous multithreading (SMT), 515–517
 - support, 515
 - thread-level parallelism, 517
 - unused issue slots, 515
- Single error correcting/Double error correcting (SEC/DEC), 420–422
- Single instruction single data (SISD), 507
- Single precision. *See also* Double precision
 - binary representation, 201
 - defined, 198
- Single-clock-cycle pipeline diagrams, 296–297
 - illustrated, 299
- Single-cycle datapaths. *See also* Datapaths
 - illustrated, 287
 - instruction execution, 288
- Single-cycle implementation
 - control function for, 269
 - defined, 270
 - nonpipelined execution *versus* pipelined execution, 276
 - non-use of, 271–272
 - penalty, 271–272
 - pipelined performance *versus*, 274
- Single-instruction multiple-thread (SIMT), C-27–30
 - overhead, C-35
 - multithreaded warp scheduling, C-28
 - processor architecture, C-28
 - warp execution and divergence, C-29–30
- Single-program multiple data (SPMD), C-22
- sll (Shift Left Logical), 64
- slt (Set Less Than), 64
- slti (Set Less Than Imm.), 64

- sltui (Set Less Than Imm.Unsigned), 64
- sltu (Set Less Than Unsig.), 64
- Smalltalk-80, OL2.21-8
- Smart phones, 7
- Snooping protocol, 468–470
- Snoopy cache coherence, OL5.12-7
- Software optimization
 - via blocking, 413–418
- Sort algorithms, 141
- Software
 - layers, 13
 - multiprocessor, 500
 - parallel, 501
 - as service, 7, 532, 558
 - systems, 13
- Sort procedure, 135–139. *See also*
 - Procedures
 - code for body, 135–137
 - full procedure, 138–139
 - passing parameters in, 138
 - preserving registers in, 138
 - procedure call, 137
 - register allocation for, 135
- Sorting performance, C-54–55
- Source files, A-4
- Source language, A-6
- Space allocation
 - on heap, 104–106
 - on stack, 103
- SPARC
 - annulling branch, E-23
 - CASA, E-31
 - conditional branches, E-10–12
 - fast traps, E-30
 - floating-point operations, E-31
 - instructions, E-29–32
 - least significant bits, E-31
 - multiple precision floating-point
 - results, E-32
 - nonfaulting loads, E-32
 - overlapping integer operations, E-31
 - quadruple precision floating-point
 - arithmetic, E-32
 - register windows, E-29–30
 - support for LISP and Smalltalk, E-30
- Sparse matrices, C-55–58
- Sparse Matrix-Vector multiply (SpMV),
 - C-55, C-57, C-58
 - CUDA version, C-57
 - serial code, C-57
 - shared memory version, C-59
- Spatial locality, 374
 - large block exploitation of, 391
 - tendency, 378
- SPEC, OL1.12-11–1.12-12
 - CPU benchmark, 46–48
 - power benchmark, 48–49
 - SPEC2000, OL1.12-12
 - SPEC2006, 233, OL1.12-12
 - SPEC89, OL1.12-11
 - SPEC92, OL1.12-12
 - SPEC95, OL1.12-12
 - SPECrate, 538–539
 - SPECratio, 47
- Special function units (SFUs), C-35, C-50
 - defined, C-43
- Speculation, 333–334
 - hardware-based, 341
 - implementation, 334
 - performance and, 334
 - problems, 334
 - recovery mechanism, 334
- Speed-up challenge, 503–505
 - balancing load, 505–506
 - bigger problem, 504–505
- Spilling registers, 71, 98
- SPIM, A-40–45
 - byte order, A-43
 - features, A-42–43
 - getting started with, A-42
 - MIPS assembler directives support,
 - A-47–49
 - speed, A-41
 - system calls, A-43–45
 - versions, A-42
 - virtual machine simulation, A-41–42
- Split algorithm, 552
- Split caches, 397
- Square root instructions, A-79
- sra (Shift Right Arith.), A-56
- srl (Shift Right Logical), 64
- Stack architectures, OL2.21-4
- Stack pointers
 - adjustment, 100
 - defined, 98
 - values, 100
- Stack segment, A-22
- Stacks
 - allocating space on, 103
 - for arguments, 140
 - defined, 98
 - pop, 98
 - push, 98, 100
 - recursive procedures, A-29–30
- Stalls, 280
 - as solution to control hazard, 282
 - avoiding with code reordering, 280
 - behavioral Verilog with detection,
 - OL4.13-6–4.13-8
 - data hazards and, 313–316
 - illustrations, OL4.13-23, OL4.13-30
 - insertion into pipeline, 315
 - load-use, 318
 - memory, 400
 - write-back scheme, 399
 - write buffer, 399
- Standby spares, OL5.11-8
- State
 - in 2-bit prediction scheme, 322
 - assignment, B-70, D-27
 - bits, D-8
 - exception, saving/restoring, 450
 - logic components, 249
 - specification of, 432
- State elements
 - clock and, 250
 - combinational logic and, 250
 - defined, 248, B-48
 - inputs, 249
 - in storing/accessing instructions,
 - 252
 - register file, B-50
- Static branch prediction, 335
- Static data
 - as dynamic data, A-21
 - defined, A-20
 - segment, 104
- Static multiple-issue processors, 333,
 - 334–339. *See also* Multiple issue
 - control hazards and, 335–336
 - instruction sets, 335
 - with MIPS ISA, 335–338
- Static random access memories (SRAMs),
 - 378, 379, B-58–62
 - array organization, B-62
 - basic structure, B-61
 - defined, 21, B-58
 - fixed access time, B-58
 - large, B-59
 - read/write initiation, B-59
 - synchronous (SSRAMs), B-60
 - three-state buffers, B-59, B-60
- Static variables, 102

- Status register
 - fields, A-34, A-35
 - Steady-state prediction, 321
 - Sticky bits, 220
 - Store buffers, 343
 - Store instructions. *See also* Load instructions
 - access, C-41
 - base register, 262
 - block, 149
 - compiling with, 71
 - conditional, 122
 - defined, 71
 - details, A-68–70
 - EX stage, 294
 - floating-point, A-79
 - ID stage, 291
 - IF stage, 291
 - instruction dependency, 312
 - list of, A-68–70
 - MEM stage, 295
 - unit for implementing, 255
 - WB stage, 295
 - Store word, 71
 - Stored program concept, 63
 - as computer principle, 86
 - illustrated, 86
 - principles, 161
 - Strcpy procedure, 108–109. *See also* Procedures
 - as leaf procedure, 109
 - pointers, 109
 - Stream benchmark, 548
 - Streaming multiprocessor (SM), C-48–49
 - Streaming processors, C-34, C-49–50
 - array (SPA), C-41, C-46
 - Streaming SIMD Extension 2 (SSE2)
 - floating-point architecture, 224
 - Streaming SIMD Extensions (SSE) and advanced vector extensions in x86, 224–225
 - Stretch computer, OL4.16-2
 - Strings
 - defined, 107
 - in Java, 109–111
 - representation, 107
 - Strip mining, 510
 - Striping, OL5.11-4
 - Strong scaling, 505, 517
 - Structural hazards, 277, 294
 - sub (Subtract), 64
 - sub.d (FP Subtract Double), A-79
 - sub.s (FP Subtract Single), A-80
 - Subnormals, 222
 - Subtraction, 178–182. *See also* Arithmetic
 - binary, 178–179
 - floating-point, 211, A-79–80
 - instructions, A-56–57
 - negative number, 179
 - overflow, 179
 - subu (Subtract Unsigned), 119
 - Subword parallelism, 222–223, 352, E-17
 - and matrix multiply, 225–228
 - Sum of products, B-11, B-12
 - Supercomputers, OL4.16-3
 - defined, 5
 - SuperH, E-15, E-39–40
 - Superscalars
 - defined, 339, OL4.16-5
 - dynamic pipeline scheduling, 339
 - multithreading options, 516
 - Surfaces, C-41
 - sw (Store Word), 64
 - Swap procedure, 133. *See also* Procedures
 - body code, 135
 - full, 135, 138–139
 - register allocation, 133
 - Swap space, 434
 - swc1 (Store FP Single), A-73
 - Symbol tables, 125, A-12, A-13
 - Synchronization, 121–123, 552
 - barrier, C-18, C-20, C-34
 - defined, 518
 - lock, 121
 - overhead, reducing, 44–45
 - unlock, 121
 - Synchronizers
 - defined, B-76
 - failure, B-77
 - from D flip-flop, B-76
 - Synchronous DRAM (SRAM), 379–380, B-60, B-65
 - Synchronous SRAM (SSRAM), B-60
 - Synchronous system, B-48
 - Syntax tree, OL2.15-3
 - System calls, A-43–45
 - code, A-43–44
 - defined, 445
 - loading, A-43
 - Systems software, 13
 - SystemVerilog
 - cache controller, OL5.12-2
 - cache data and tag modules, OL5.12-6
 - FSM, OL5.12-7
 - simple cache block diagram, OL5.12-4
 - type declarations, OL5.12-2
- ## T
- Tablets, 7
 - Tags
 - defined, 384
 - in locating block, 407
 - page tables and, 434
 - size of, 409
 - Tail call, 105–106
 - Task identifiers, 446
 - Task parallelism, C-24
 - Task-level parallelism, 500
 - Tebibyte (TiB), 5
 - Telsa PTX ISA, C-31–34
 - arithmetic instructions, C-33
 - barrier synchronization, C-34
 - GPU thread instructions, C-32
 - memory access instructions, C-33–34
 - Temporal locality, 374
 - tendency, 378
 - Temporary registers, 67, 99
 - Terabyte (TB) , 6
 - defined, 5
 - Text segment, A-13
 - Texture memory, C-40
 - Texture/processor cluster (TPC), C-47–48
 - TFLOPS multiprocessor, OL6.15-6
 - Thrashing, 453
 - Thread blocks, 528
 - creation, C-23
 - defined, C-19
 - managing, C-30
 - memory sharing, C-20
 - synchronization, C-20
 - Thread parallelism, C-22
 - Threads
 - creation, C-23
 - CUDA, C-36
 - ISA, C-31–34
 - managing, C-30
 - memory latencies and, C-74–75
 - multiple, per body, C-68–69
 - warps, C-27
 - Three Cs model, 459–461
 - Three-state buffers, B-59, B-60

- Throughput
 - defined, 30–31
 - multiple issue and, 342
 - pipelining and, 286, 342
 - Thumb, E-15, E-38
 - Timing
 - asynchronous inputs, B-76–77
 - level-sensitive, B-75–76
 - methodologies, B-72–77
 - two-phase, B-75
 - TLB misses, 439. *See also* Translation-lookaside buffer (TLB)
 - entry point, 449
 - handler, 449
 - handling, 446–453
 - occurrence, 446
 - problem, 453
 - Tomasulo's algorithm, OL4.16-3
 - Touchscreen, 19
 - Tournament branch predictors, 324
 - Tracks, 381–382
 - Transfer time, 383
 - Transistors, 25
 - Translation-lookaside buffer (TLB), 438–439, E-26–27, OL5.17-6. *See also* TLB misses
 - associativities, 439
 - illustrated, 438
 - integration, 440–441
 - Intrinsity FastMATH, 440
 - typical values, 439
 - Transmit driver and NIC hardware time *versus* receive driver and NIC hardware time, OL6.9-8
 - Transmitter Control register, A-39–40
 - Transmitter Data register, A-40
 - Trap instructions, A-64–66
 - Tree-based parallel scan, C-62
 - Truth tables, B-5
 - ALU control lines, D-5
 - for control bits, 260–261
 - datapath control outputs, D-17
 - datapath control signals, D-14
 - defined, 260
 - example, B-5
 - next-state output bits, D-15
 - PLA implementation, B-13
 - Two's complement representation, 75–76
 - advantage, 75–76
 - negation shortcut, 76
 - rule, 79
 - sign extension shortcut, 78
 - Two-level logic, B-11–14
 - Two-phase clocking, B-75
 - TX-2 computer, OL6.15-4
- ## U
- Unconditional branches, 91
 - Underflow, 198
 - Unicode
 - alphabets, 109
 - defined, 110
 - example alphabets, 110
 - Unified GPU architecture, C-10–12
 - illustrated, C-11
 - processor array, C-11–12
 - Uniform memory access (UMA), 518, C-9
 - multiprocessors, 519
 - Units
 - commit, 339–340, 343
 - control, 247–248, 259–261, D-4–8, D-10, D-12–13
 - defined, 219
 - floating point, 219
 - hazard detection, 313, 314–315
 - for load/store implementation, 255
 - special function (SFUs), C-35, C-43, C-50
 - UNIVAC I, OL1.12-5
 - UNIX, OL2.21-8, OL5.17-9–5.17-12
 - AT&T, OL5.17-10
 - Berkeley version (BSD), OL5.17-10
 - genius, OL5.17-12
 - history, OL5.17-9–5.17-12
 - Unlock synchronization, 121
 - Unresolved references
 - defined, A-4
 - linkers and, A-18
 - Unsigned numbers, 73–78
 - Use latency
 - defined, 336–337
 - one-instruction, 336–337
- ## V
- Vacuum tubes, 25
 - Valid bit, 386
 - Variables
 - C language, 102
 - programming language, 67
 - register, 67
 - static, 102
 - storage class, 102
 - type, 102
 - VAX architecture, OL2.21-4, OL5.17-7
 - Vector lanes, 512
 - Vector processors, 508–510. *See also* Processors
 - conventional code comparison, 509–510
 - instructions, 510
 - multimedia extensions and, 511–512
 - scalar *versus*, 510–511
 - Vectored interrupts, 327
 - Verilog
 - behavioral definition of MIPS ALU, B-25
 - behavioral definition with bypassing, OL4.13-4–4.13-6
 - behavioral definition with stalls for loads, OL4.13-6–4.13-8
 - behavioral specification, B-21, OL4.13-2–4.13-4
 - behavioral specification of multicycle MIPS design, OL4.13-12–4.13-13
 - behavioral specification with simulation, OL4.13-2
 - behavioral specification with stall detection, OL4.13-6–4.13-8
 - behavioral specification with synthesis, OL4.13-11–4.13-16
 - blocking assignment, B-24
 - branch hazard logic implementation, OL4.13-8–4.13-10
 - combinational logic, B-23–26
 - datatypes, B-21–22
 - defined, B-20
 - forwarding implementation, OL4.13-4
 - MIPS ALU definition in, B-35–38
 - modules, B-23
 - multicycle MIPS datapath, OL4.13-14
 - nonblocking assignment, B-24
 - operators, B-22
 - program structure, B-23
 - reg, B-21–22
 - sensitivity list, B-24
 - sequential logic specification, B-56–58
 - structural specification, B-21
 - wire, B-21–22
 - Vertical microcode, D-32

- Very large-scale integrated (VLSI) circuits, 25
- Very Long Instruction Word (VLIW)
defined, 334–335
first generation computers, OL4.16-5
processors, 335
- VHDL, B-20–21
- Video graphics array (VGA) controllers, C-3–4
- Virtual addresses
causing page faults, 449
defined, 428
mapping from, 428–429
size, 430
- Virtual machine monitors (VMMs)
defined, 424
implementing, 481, 481–482
laissez-faire attitude, 481
page tables, 452
in performance improvement, 427
requirements, 426
- Virtual machines (VMs), 424–427
benefits, 424
defined, A-41
illusion, 452
instruction set architecture support, 426–427
performance improvement, 427
for protection improvement, 424
simulation of, A-41–42
- Virtual memory, 427–454. *See also* Pages
address translation, 429, 438–439
integration, 440–441
mechanism, 452–453
motivations, 427–428
page faults, 428, 434
protection implementation, 444–446
segmentation, 431
summary, 452–453
virtualization of, 452
writes, 437
- Virtualizable hardware, 426
- Virtually addressed caches, 443
- Visual computing, C-3
- Volatile memory, 22
- W**
- Wafers, 26
defects, 26
dies, 26–27
yield, 27
- Warehouse Scale Computers (WSCs), 7, 531–533, 558
- Warps, 528, C-27
- Weak scaling, 505
- Wear levelling, 381
- While loops, 92–93
- Whirlwind, OL5.17-2
- Wide area networks (WANs), 24. *See also* Networks
- Words
accessing, 68
defined, 66
double, 152
load, 68, 71
quad, 154
store, 71
- Working set, 453
- World Wide Web, 4
- Worst-case delay, 272
- Write buffers
defined, 394
stalls, 399
write-back cache, 395
- Write invalidate protocols, 468, 469
- Write serialization, 467
- Write-back caches. *See also* Caches
advantages, 458
cache coherency protocol, OL5.12-5
complexity, 395
defined, 394, 458
stalls, 399
write buffers, 395
- Write-back stage
control line, 302
load instruction, 292
store instruction, 294
- Writes
complications, 394
expense, 453
handling, 393–395
memory hierarchy handling of, 457–458
schemes, 394
virtual memory, 437
write-back cache, 394, 395
write-through cache, 394, 395
- Write-stall cycles, 400
- Write-through caches. *See also* Caches
advantages, 458
defined, 393, 457
tag mismatch, 394
- X**
- x86, 149–158
Advanced Vector Extensions in, 225
brief history, OL2.21-6
conclusion, 156–158
data addressing modes, 152, 153–154
evolution, 149–152
first address specifier encoding, 158
historical timeline, 149–152
instruction encoding, 155–156
instruction formats, 157
instruction set growth, 161
instruction types, 153
integer operations, 152–155
registers, 152, 153–154
SIMD in, 507–508, 508
Streaming SIMD Extensions in, 224–225
typical instructions/functions, 155
typical operations, 157
- Xerox Alto computer, OL1.12-8
- XMM, 224
- Y**
- Yahoo! Cloud Serving Benchmark (YCSB), 540
- Yield, 27
- YMM, 225
- Z**
- Zettabyte, 6