

# Index

*Note:* Page numbers in *italics* indicate figures, tables and text boxes; page numbers preceded by “e” refer to online material.

- 0, 8, 22. *See also* LOW, FALSE
  - 1, 8, 22. *See also* HIGH, TRUE
  - 32-bit datapath, 386
  - 32-bit instructions, 329
  - 64-bit architecture, 360
  - 74xx series logic, 533.e1–533.e5
    - parts
      - 2:1 mux (74157), 533.e4
      - 3:8 decoder (74138), 533.e4
      - 4:1 mux (74153), 533.e4
      - AND (7408), 533.e3
      - AND3 (7411), 533.e3
      - AND4 (7421), 533.e3
      - counter (74161, 74163), 533.e4
      - FLOP (7474), 533.e1, 533.e3
      - NAND (7400), 533.e3
      - NOR (7402), 533.e3
      - NOT (7404), 533.e1
      - OR (7432), 533.e3
      - register (74377), 533.e4
      - tristate buffer (74244), 533.e4
      - XOR (7486), 533.e3
  - #define, 541.e5–541.e6
  - #include, 541.e6–541.e7. *See also*
    - Standard libraries
- A**
- ABI. *See* Application Binary Interface (ABI)
  - Abstraction, 4–5
    - digital. *See* Digital abstraction
  - Accumulator, 367
  - Acorn Computer Group, 296, 472
  - Acorn RISC Machine, 350
  - Active low, 74–75
  - A/D conversion, 531.e31–531.e32
  - Ad hoc* testing, 452
  - ADCs. *See* Analog-to-digital converters (ADCs)
  - ADD, 297, 536
  - Adders, 239–246
    - carry propagate, 240
    - carry-lookahead, 241
    - full, 56, 240
    - half, 240
    - HDL for, 184, 200, 450
    - prefix, 243
    - ripple-carry, 240
  - Addition, 14–15, 17–18, 235, 239–246, 297. *See also* Adders
    - binary, 14–15
    - floating point, 259
    - signed binary, 15–17
  - Address. *See also* Memory
    - physical, 509–513
    - translation, 509–512
    - virtual, 508. *See also* Virtual memory
  - Addressing modes, ARM, 336
    - base, 336
    - immediate, 336
    - PC-relative, 336
    - register, 336
  - Advanced High-performance Bus (AHB), 531.e54
  - Advanced Micro Devices (AMD), 296
  - Advanced microarchitecture, 456–470
    - branch prediction. *See* Branch prediction
    - deep pipelines. *See* Deep pipelines
    - heterogeneous multiprocessors. *See* Heterogeneous multiprocessors
    - homogeneous multiprocessors. *See* Homogeneous multiprocessors
    - micro-operations. *See* Micro-operations
    - multiprocessors. *See* Multiprocessors
    - multithreading. *See* Multithreading
    - out-of-order processor. *See* Out-of-order processor
    - register renaming. *See* Register renaming
    - single instruction multiple data. *See* Single instruction multiple data (SIMD)
    - superscalar processor. *See* Superscalar processor
  - Advanced Microcontroller Bus Architecture (AMBA), 531.e54
  - Advanced RISC Machines, 472
  - AHB. *See* Advanced High-performance Bus (AHB)
  - AHB-Lite bus, 531.e54–531.e55
  - Altera FPGA, 274–279
  - ALU. *See* Arithmetic/logical unit (ALU)
  - ALU Decoder, 398–400
  - ALUControl, 248–250, 392, 395
  - ALUOp, 398

- ALUResult*, 392–397
  - ALUSrc*, 396
  - AMAT. *See* Average memory access time (AMAT)
  - AMBA. *See* Advanced Microcontroller Bus Architecture (AMBA)
  - AMD. *See* Advanced Micro Devices (AMD)
  - AMD64, 368
  - Amdahl, Gene, 492
  - Amdahl's Law, 492
  - American Standard Code for Information Interchange (ASCII), 315–316, 541.e8, 541.e27–541.e28
  - Analog I/O, 531.e25–531.e32
    - A/D conversion, 531.e31–531.e32
    - D/A conversion, 531.e25–531.e28
    - Pulse-width modulation (PWM), 531.e28–531.e31
  - Analog-to-digital converters (ADCs), 531.e25, 531.e27, 531.e31–531.e32
  - Analytical engine, 7–8
  - AND gate, 20–22, 179
    - chips (7408, 7411, 7421), 533.e3
    - truth table, 20, 22
    - using CMOS transistors, 32–33
  - AND, 303–304
  - AND-OR (AO) gate, 46
  - Anode, 27
  - Antidependence, 464
  - Application Binary Interface (ABI), 320
  - Application-specific integrated circuits (ASICs), 533.e9
  - Architectural state, 338, 364
    - for ARM, 385–386
  - Architecture, 295
    - assembly language, 296
    - instructions, 297–298
    - operands, 298–303
    - compiling, assembling, and loading, 339
      - assembling, 342–343
      - compilation, 340–341
      - linking, 343–344
      - loading, 344–345
      - memory map, 339–340
    - evolution of ARM architecture, 350
    - 64-bit architecture, 360
    - digital signal processors (DSPs), 352–356
    - floating-point instructions, 357–358
  - power-saving and security instructions, 358
  - SIMD instructions, 358–360
  - Thumb instruction set, 351–352
- ARM instructions, 295–369, 535–540
  - branch instructions, 308–309, 539
  - condition flags, 306–308, 540
  - data-processing instructions, 303–306, 535–537
    - logical instructions, 303–304
    - multiply instructions, 305–306, 537
    - shift instructions, 304–305
  - formats
    - addressing modes, 336
    - branch instructions, 334
    - data-processing instructions, 329–333
    - interpreting, 336–337
    - memory instructions, 333–335
    - stored program, 337–338
  - instruction set, 295
  - memory instructions, 301–303, 313–317, 538
  - miscellaneous instructions, 345–346, 539
- ARM Microcontroller Development Kit (MDK-ARM), 297
- ARM microprocessor, 385
  - data memory, 385–388
  - instruction memory, 385–388
  - multicycle, 406–425
  - pipelined, 425–433
  - program counter, 385–388
  - register file, 385–388
  - single-cycle, 390–406, 443–456
  - state elements of, 385–388
- ARM processors, 470
- ARM registers, 299–300
  - program counter, 308, 338, 386–387
  - register file, 386–387
  - register set, 299–300
- ARM single-cycle HDL, 443–456
  - building blocks, 449–452
  - controller, 443
  - datapath, 443
  - testbench, 452–456
- ARM7, 472, 473
- ARM9, 474
- ARM9E, 472
- ARMv3 architecture, 472
- ARMv4 instruction set, 295, 539
- ARMv7 instruction, 472
- Arrays, 313–317, 541.e23–541.e29
  - accessing, 313–317, 541.e23
- machine language, 329
  - addressing modes, 336
  - branch instructions, 334–335
  - data-processing instructions, 329–333
  - interpreting, 336–337
  - memory instructions, 333–334
  - stored program, 337–338
- odds and ends, 345
  - exceptions, 347–350
  - loading literals, 345–346
  - NOP, 346
- programming, 303
  - branching, 308–309
  - conditional statements, 309–312
  - condition flags, 306–308
  - function calls, 317–329
  - getting loopy, 312–313
  - logical and arithmetic instructions, 303–306
  - memory, 313–317
- x86 architecture, 360
  - big picture, 368
  - instruction encoding, 364–367
  - instructions, 364
  - operands, 362–363
  - peculiarities, 367–368
  - registers, 362
  - status flags, 363–364
- Arguments, 317–319, 541.e15
  - pass by reference, 541.e22
  - pass by value, 541.e22
- Arithmetic
  - ARM instructions, 303–306
  - circuits, 239–255
  - C operators, 541.e11–541.e13
  - HDL operators, 185
- Arithmetic/logical unit (ALU), 248–251, 392
  - implementation of, 249
  - in processor, 392–430
- ARM architecture, evolution of, 296, 350
  - 64-bit architecture, 360
  - digital signal processing (DSP) instructions, 352–356
  - floating-point instructions, 357–358
  - power-saving and security instructions, 358
- SIMD instructions, 358–360
- Thumb instruction set, 351–352

- bytes and characters, 315–317, 541.e27–541.e29
  - comparison or assignment of, 541.e28
  - declaration, 314–317, 541.e23
  - indexing, 314–317, 541.e23–541.e27
  - initialization, 541.e23–541.e24
  - as input argument, 541.e24–541.e25
  - multi-dimension, 541.e26–541.e27
  - ASCII. *See* American Standard Code for Information Interchange (ASCII)
  - ASICs. *See* Application-specific integrated circuits (ASICs)
  - ASR, 304
  - Assembler, 339, 541.e44
  - Assembling, 342–343
  - Assembly language, ARM, 295–350, 535–540
    - instructions, 297–350, 535–540
    - operands, 297–303
    - translating high-level code to, 339–345
    - translating machine language to, 337
  - Assembly language, x86. *See* x86
  - Associativity
    - in Boolean algebra, 62, 63
    - in caches, 493, 498–500
  - Astable circuits, 119
  - Asymmetric multiprocessors. *See* Heterogeneous multiprocessors
  - Asynchronous circuits, 120–123
  - Asynchronous resettable flip-flops
    - definition, 116
    - HDL, 194–196
  - Asynchronous serial link, 531.e17, 531.e17. *See also* Universal Asynchronous Receiver Transmitter (UART)
  - AT Attachment (ATA), 531.e61–531.e62
  - Average memory access time (AMAT), 491, 504
- B**
- B, 308–309, 334–336, 396–397
  - Babbage, Charles, 7
  - Banked registers, 348–349
  - Base addressing, 336
  - Baud rate, 531.e17–531.e19
  - BCD. *See* Binary coded decimal (BCD)
  - BCM2835, 531.e3, 531.e4–531.e5, 531.e8, 531.e9, 531.e19
  - timer, 531.e23
  - Behavioral modeling, 173–174
  - Benchmarks, 389
  - BEQ, 309
  - Biased exponent, 257
  - BIC (bit clear), 303–304
  - big.LITTLE, 469
  - Big-endian memory, 303
  - Big-endian order, 178
  - Binary addition, 14–15. *See also* Adders, Addition
  - Binary coded decimal (BCD), 258
  - Binary encoding, 125–126, 129–131
    - for divide-by-3 counter, 129–131
    - for traffic light FSM, 125–126
  - Binary numbers
    - signed, 15–19
    - unsigned, 9–11
  - Binary to decimal conversion, 10, 10–11
  - Binary to hexadecimal conversion, 12
  - Bipolar junction transistors, 26
  - Bipolar motor drive, 531.e50
  - Bipolar signaling, 531.e18
  - Bipolar stepper motor, 531.e51, 531.e52–531.e53
    - AIRPAX LB82773-M1, 531.e51, 531.e51
    - direct drive current, 531.e52
  - Bistable element, 109
  - Bit, 8
    - dirty, 506
    - least significant, 13, 14
    - most significant, 13, 14
    - sign, 16
    - use, 502
    - valid, 496
  - Bit cells, 264–269
    - DRAM, 266–267
    - ROM, 268–270
    - SRAM, 267
  - Bit swizzling, 188
  - Bitline, 264
  - Bitwise operators, 177–179
  - BL (branch and link), 318
  - Block, 493
  - Block offset, 500–501
  - Block size (*b*), 493, 500–501
  - Blocking and nonblocking assignments, 199–200, 205–209
  - BLT. *See* Branch if less than (BLT)
  - BlueSMiRF silver module, 531.e42–531.e43, 531.e42
  - Bluetooth wireless communication, 531.e42–531.e43
    - BlueSMiRF silver module, 531.e42–531.e43
    - classes, 531.e42
  - BNE, 310
  - Boole, George, 8
  - Boolean algebra, 60–66
    - axioms, 61
    - equation simplification, 65–66
    - theorems, 61–64
  - Boolean equations, 58–60
    - product-of-sums form, 60
    - sum-of-products form, 58–60
  - Boolean logic, 8. *See also* Boolean algebra, Logic gates
  - Boolean theorems, 61–64
    - associativity, 63
    - combining, 62
    - commutativity, 63
    - complements, 62
    - consensus, 62, 64
    - covering, 62
    - De Morgan's, 63–64
    - distributivity, 63
    - idempotency, 62
    - identity, 62
    - involution, 62
    - null element, 62
  - Branch if less than (BLT), 334–335
  - Branch instructions, 308–309
    - ARM instructions, 539, 539
  - Branch misprediction penalty, 438, 459
  - Branch prediction, 459–461
  - Branch target address (BTA), 334–335
  - Branch target buffer, 459
  - Branching, 308–309, 334–336
    - conditional, 309
    - unconditional, 309
  - Breadboards, 533.e18–533.e19
  - BTA. *See* Branch target address (BTA)
  - Bubble, 20, 63
    - pushing, 63–64, 71–73
  - Bubble, in pipeline, 435–436
  - Buffers, 20
    - lack of, 117
    - tristate, 74–75
  - Bugs, 175
    - in C code, 541.e45–541.e49
  - Bus, 56
    - tristate, 75

- Bus interfaces, 531.e54–531.e57
    - AHB-Lite, 531.e54–531.e55
    - memory and peripheral interface example, 531.e55–531.e57
  - Bypassing, 432. *See also* Forwarding
  - Byte, 13–14, 315–317. *See also* Characters
    - least significant, 13–14
    - most significant, 13–14
  - Byte offset, 495
  - Byte-addressable memory, 301–302
    - big-endian, 302–303
    - little-endian, 303
- C**
- C programming, 541.e1–541.e49
    - common mistakes. *See* Common mistakes in C
    - mistakes in C
    - compiler. *See* Compiler, i\_Hlt414277118n C
    - conditional statements. *See* Conditional statements
    - control-flow statements. *See* Control-flow statements
    - data types. *See* Data types
    - executing a program, 541.e4
    - function calls. *See* Function calls
    - loops. *See* Loops
    - operators. *See* Operators
    - simple program, 541.e3–541.e4
    - standard libraries. *See* Standard libraries
    - variables. *See* Variables in C
  - Caches, 489–508
    - address fields
      - block offset, 500–501
      - byte offset, 495
      - set bits, 495
      - tag, 495
    - advanced design, 503–507
    - evolution of, in ARM, 507
    - multiple level, 504
    - organizations, 502
      - direct mapped, 494–498
      - fully associative, 499–500
      - multi-way set associative, 498–499
    - parameters
      - block, 493
      - block size, 493, 500–501
      - capacity (C), 492–493
      - degree of associativity (N), 499
      - number of sets (S), 493
    - performance of
      - hit, 490–492
      - hit rate, 491–492
      - miss, 480–492, 505
        - capacity, 505
        - compulsory, 505
        - conflict, 498, 505
        - penalty, 500
      - miss rate, 491–492
        - reducing, 505–506
      - miss rate *vs.* cache parameters, 505–506
    - replacement policy, 502–503
    - status bits
      - dirty bit (D), 506
      - use bit (U), 502
      - valid bit (V), 496
    - write policy, 506–507
      - write-back, 506–507
      - write-through, 506–507
  - CAD. *See* Computer-aided design (CAD)
  - Callee, 317
  - Callee save rule, 324
  - Callee-saved registers, 323
  - Caller save rule, 324
  - Caller-saved registers, 323
  - Canonical form. *See* Sum-of-products (SOP) form, Product-of-sums (POS) form
  - Capacitors, 28
  - Capacity, of cache, 492–493
  - Capacity miss, 505
  - Carry propagate adder (CPA). *See* Carry-lookahead adder (CLA); Prefix adders; Ripple-carry adder
  - Carry-lookahead adder (CLA), 241–243, 242
  - case statement, in HDL, 201–203. *See also* Switch/case statement
  - casez, case?, in HDL, 205
  - Cathode, 27
  - Cathode ray tube (CRT), 531.e36. *See also* VGA (Video Graphics Array) monitor
    - horizontal blanking interval, 531.e36
    - vertical blanking interval, 531.e36
  - Character LCDs, 531.e33–531.e36
  - Characters (char), 315–317, 541.e8, 541.e27
    - arrays. *See also* Strings
    - C type, 541.e27
  - Chips, 28
    - multiprocessors, 468
  - Chopper constant current drive, 531.e51
  - Circuits
    - 74xx series. *See* 74xx series logic application-specific integrated (ASICs), 533.e9
    - astable, 119
    - asynchronous, 120, 122–123
    - combinational. *See* Combinational logic
    - definition of, 55
    - delay, 88–92
    - glitches in, 92–95
    - multiple-output, 68
    - priority, 68
    - sequential. *See* Sequential logic
    - synchronous, 122–123
    - synchronous sequential, 120–123
    - synthesized, 176, 179, 181
    - timing, 88–95, 141–151
  - CISC. *See* Complex Instruction Set Computer (CISC) architectures
  - CLBs. *See* Configurable logic blocks (CLBs)
  - Clock cycles per instruction (CPI), 390
  - Clock period, 142, 390
  - Clock skew, 148–151
  - Clustered multiprocessors, 470
  - cmd field, 330, 535, 537
  - CMOS. *See* Complementary Metal-Oxide-Semiconductor Logic (CMOS)
  - CMP, 402
  - Combinational composition, 56
  - Combinational logic, 174
    - design, 55–106
      - Boolean algebra, 60–66
      - Boolean equations, 58–60
      - building blocks, 83–88, 239–255
      - delays, 88–92
      - don't cares, 81–82
      - Karnaugh maps (K-maps), 75–83
      - multilevel, 66–73
      - precedence, 58
      - timing, 88–95
      - two-level, 69

- X (contention). *See* Contention (X)
  - X (don't cares). *See* Don't care (X)
  - Z (floating). *See* Floating (Z)
  - HDLs. *See* Hardware description languages (HDLs)
  - Combining theorem, 62
  - Command line arguments, 541. e44–541.e45
  - Comments
    - in ARM assembly, 297–298
    - in C, 297–298, 541.e5
    - in SystemVerilog, 180
    - in VHDL, 180
  - Common mistakes in C, 541.e45–541.e49
  - Comparators, 246–248
  - Comparison
    - in hardware. *See* Comparators; Arithmetic/logical unit (ALU)
    - processor performance, 424–425
    - using ALU, 251
  - Compiler, in C, 339–345, 541.e4–541.e5, 541.e43–541.e44
  - Complementary Metal-Oxide-Semiconductor gates (CMOS), 26–34
  - Complements theorem, 62
  - Complex instruction set computer (CISC) architectures, 298, 361, 458
  - Complexity management, 4–7
    - digital abstraction, 4–5
    - discipline, 5–6
    - hierarchy, 6–7
    - modularity, 6–7
    - regularity, 6–7
  - Compulsory miss, 505
  - Computer-aided design (CAD), 71, 129
  - Concurrent signal assignment statement, 179, 183–184, 193, 200–206
  - cond* field, 306–307, 330, 535
  - Condition flags, 306–308
    - ARM instructions, 540, 540
  - Condition mnemonics, 307
  - Conditional assignment, 181–182
  - Conditional branches, 308–309
  - Conditional Logic, 398–400, 413–415
  - Conditional operator, 181–182
  - Conditional signal assignments, 181–182
  - Conditional statements, 309
    - in ARM assembly
      - if, 309–310
      - if/else, 310–311
      - switch/case, 311–312
    - in C, 541.e17–541.e18
      - if, 541.e17–541.e18
      - if/else, 541.e17
      - switch/case, 541.e17–541.e18
    - in HDL, 194, 201–205
      - case, 201–203
      - casez, case?, 205
      - if, if/else, 202–205
  - Configurable logic blocks (CLBs), 275, 533.e7. *See also* Logic elements (LEs)
  - Conflict miss, 505
  - Consensus theorem, 62, 64
  - Constants
    - in ARM assembly, 300–301. *See also* Immediates
    - in C, 541.e5–541.e6
  - Contamination delay, 88–92. *See also* Short path
  - Contention (x), 73–74
  - Context switching, 467
  - Continuous assignment statements, 179, 193, 200, 206
  - Control hazard, 432, 437–440
  - Control signals, 91, 249
  - Control unit, 386. *See also* ALU Decoder, Main Decoder
    - of multicycle ARM processor, 413–423
    - of pipelined ARM processor, 430
    - of single-cycle ARM processor, 397–401
  - Control-flow statements
    - conditional statements. *See* Conditional statements
    - loops. *See* Loops
  - CoreMark, 389
  - Cortex-A7 and -A15, 475
  - Cortex-A9, 475
  - Counters, 260–261
    - divide-by-3, 130
  - Covering theorem, 62
  - CPA. *See* Carry propagate adder (CPA)
  - CPI. *See* Clock cycles per instruction (CPI); Cycles per instruction (CPI)
  - Critical path, 89–92, 402
  - Cross-coupled inverters, 109, 110
    - bistable operation of, 110
  - CRT. *See* Cathode ray tube (CRT)
  - Current Program Status Register (CPSR), 306, 324, 347
  - Cycle time. *See* Clock period
  - Cycles per instruction (CPI), 390, 424
  - Cyclic paths, 120
  - Cyclone IV FPGA, 275–279
- D**
- D flip-flops. *See* Flip-flops
  - D latch. *See* La\_Hlt414277505tches
  - D/A conversion, 531.e25–531.e28
  - DACs. *See* Digital-to-analog converters (DACs)
  - DAQs. *See* Data Acquisition Systems (DAQs)
  - Data Acquisition Systems (DAQs), 531. e62–531.e63
    - myDAQ, 531.e62–531.e63
  - Data hazard, 432–436
    - HDL for, 455
  - Data memory, 387–388
  - Data segment, 340
  - Data sheets, 533.e9–533.e14
  - Data types, 541.e21–541.e35
    - arrays. *See* Arrays
    - characters. *See* Characters (char)
    - dynamic memory allocation. *See* Dynamic memory allocation (malloc, free)
    - linked list. *See* Linked list
    - pointers. *See* Pointers
    - strings. *See* Strings
    - structures. *See* Structures (struct)
    - typedef, 541.e31–541.e32
  - Datapath
    - multicycle ARM processor, 406–413
      - B instruction, 412–413
      - LDR instruction, 407–410
      - STR instruction, 411–412
    - pipelined ARM processor, 428–430
    - single-cycle ARM processor, 390
      - B instruction, 396–397

- Datapath (*Continued*)
    - LDR instruction, 391–394
    - STR instruction, 394–396
  - Data-processing instructions, 536
    - ARM instructions, 329–333, 396–397, 535–537
    - encodings, 536
  - DC motors, 531.e43, 531.e44–531.e48
    - H-bridge, 531.e44, 531.e45
    - shaft encoder, 531.e43–531.e44
  - DC transfer characteristics, 24–26. *See also* Direct current (DC) transfer characteristics, Noise margins
  - DDR. *See* Double-data rate memory (DDR)
  - De Morgan, Augustus, 63
  - De Morgan's theorem, 63–64
  - DE-9 cable, 531.e19
  - Decimal numbers, 9
  - Decimal to binary conversion, 11
  - Decimal to hexadecimal conversion, 13
  - Decode stage, 425
  - Decoders
    - definition of, 86–87
    - HDL for
      - behavioral, 202–203
      - parameterized, 219
    - logic using, 87–88
    - Seven-segment. *See* Seven-segment display decoder
  - Deep pipelines, 457
  - Delaymicros function, 531.e24
  - Delays, logic gates. *See also* Propagation delay
    - in HDL (simulation only), 188–189
  - DeleteUser function, 541.e33
  - Dennard, Robert, 266
  - Destination register (rd or rt), 393, 409
  - Device driver, 531.e3, 531.e6–531.e8
  - Device under test (DUT), 220
  - Dhrystone, 389
  - Dice, 28
  - Dielectric, 28
  - Digital abstraction, 4–5, 7–9, 22–26
  - Digital circuits. *See* Logic
  - Digital signal processors (DSPs), 352–356, 469
  - Digital system implementation, 533. e1–533.e35
    - 74xx series logic. *See* 74xx series logic
    - application-specific integrated circuits (ASICs), 533.e9
    - assembly of, 533.e17–533.e20
    - breadboards, 533.e18–533.e19
    - data sheets, 533.e9–533.e14
    - economics, 533.e33–533.e35
    - logic families, 533.e15–533.e17
    - packaging, 533.e17–533.e20
    - printed circuit boards, 533.e19–533. e20
    - programmable logic, 533.e2–533.e9
  - Digital-to-analog converters (DACs), 531.e25–531.e28
  - DIMM. *See* Dual inline memory module (DIMM)
  - Diodes, 27–28
    - p-n junction, 28
  - DIPs. *See* Dual-inline packages (DIPs)
  - Direct current (DC) transfer characteristics, 24, 25
  - Direct mapped cache, 494–498, 495
  - Direct voltage drive, 531.e51
  - Dirty bit (*D*), 506
  - Discipline
    - dynamic, 142–151. *See also* Timing analysis
    - static, 142–151. *See also* Noise margins
  - Discrete-valued variables, 7
  - Distributivity theorem, 63
  - Divide-by-3 counter
    - design of, 129–131
    - HDL for, 210–211
  - Divider, 254–255
  - Division
    - circuits, 254–255
  - Do/while loops, in C, 541.e19–541.e20
  - Don't care (*X*), 69, 81–83, 205
  - Dopant atoms, 27
  - Double, C type, 541.e8–541.e9
  - Double-data rate memory (DDR), 268, 531.e60–531.e61
  - Double-precision formats, 258
  - DRAM. *See* Dynamic random access memory (DRAM)
  - DSPs. *See* Digital signal processors (DSPs)
  - Dual inline memory module (DIMM), 531.e60
  - Dual-inline packages (DIPs), 28, 533.e1, 533.e17
  - Dynamic branch predictors, 459
  - Dynamic data segment, 340
  - Dynamic discipline, 142–151. *See also* Timing analysis
  - Dynamic memory allocation (malloc, free), 541.e32–541.e33
    - in ARM memory map, 340
  - Dynamic power, 34
  - Dynamic random access memory (DRAM), 266–267, 487–490, 519, 531.e58, 531.e60, 531.e61
- ## E
- EasyPIO, 531.e6
  - Economics, 533.e33
  - Edge-triggered flip-flop. *See* Flip-flops
  - EEPROM. *See* Electrically erasable programmable read only memory (EEPROM)
  - EFLAGS register, 363
  - Electrically erasable programmable read only memory (EEPROM), 270
  - Embedded I/O (input/output) systems, 531.e3–531.e32
    - analog I/O, 531.e25–531.e32
    - A/D conversion, 531.e31–531. e32
    - D/A conversion, 531.e25–531. e28
    - digital I/O, 531.e8–531.e11
    - general-purpose I/O (GPIO), 531. e8–531.e11
    - interrupts, 531.e32
    - LCDs. *See* Liquid Crystal Displays (LCDs)
    - microcontroller peripherals, 531. e32–531.e53
    - motors. *See* Motors
    - serial I/O, 531.e11–531.e23. *See also* Serial I/O
    - timers, 531.e23–531.e24
    - VGA monitor. *See* VGA (Video Graphics Array) monitor
  - Enabled flip-flops, 115–116
  - Enabled registers, 196–197. *See also* Flip-flops
  - EOR (XOR), 303–304
  - EPROM. *See* Erasable programmable read only memory (EPROM)

Equality comparator, 247  
 Equation minimization  
   using Boolean algebra, 65–66  
   using Karnaugh maps. *See* Karnaugh maps (K-maps)  
 Erasable programmable read only memory (EPROM), 270, 533.e6  
 Ethernet, 531.e61  
 Exceptions, 346–350  
   banked registers, 348–349  
   exception-related instructions, 349–350  
   exception vector table, 347–348  
   execution modes and privilege levels, 347  
   handler, 340, 349  
   start-up, 350  
 Execution time, 389  
`exit`, 541.e41  
 Extended instruction pointer (EIP), 362  
*ExtImm*, 408

## F

factorial function call, 326  
   stack during, 327  
 Factoring state machines, 134–136  
 False, 8, 20, 35, 58, 60, 74, 111, 112, 113, 116, 124, 196  
 Fast Fourier Transform (FFT), 352  
 FDIV. *See* Floating-point division (FDIV)  
 FFT. *See* Fast Fourier Transform (FFT)  
 Field programmable gate arrays (FPGAs), 274–279, 531.e14, 531.e38, 531.e63, 533.e7–533.e9  
   driving VGA cable, 531.e38  
   in SPI interface, 531.e13–531.e16  
 File manipulation, in C, 541.e38–541.e40  
 Finite state machines (FSMs), 123–141, 209–213, 413, 417  
   complete multicycle control, 424  
   deriving from circuit, 137–140  
   divide-by-3 FSM, 129–131, 210–211  
   factoring, 134–136, 136  
   in HDL, 209–213  
   LE configuration for, 277–279  
   Mealy FSM, 132–134  
   Moore FSM, 132–134  
   snail/pattern recognizer FSM, 132–134, 212–213  
   state encodings, 129–131. *See also* Binary encoding, One-cold encoding, One-hot encoding  
   state transition diagram, 124, 125  
   traffic light FSM, 123–129  
 Fixed-point numbers, 255–256  
 Flags, 250  
 Flash memory, 270. *See also* Solid state drive (SSD)  
 Flip-flops, 114–118, 193–197. *See also* Registers  
   back-to-back, 145, 152–157, 197. *See also* Synchronizers  
   comparison with latches, 118  
   enabled, 115–116  
   HDL for, 451. *See also* Registers  
   metastable state of. *See* Metastability  
   register, 114–115  
   resettable, 116  
   scannable, 262–263  
   shift register, 261–263  
   transistor count, 114, 117  
   transistor-level, 116–117  
 Float, C type, 541.e6–541.e9  
   print formats of, 541.e36–541.e37  
 Floating (Z), 74–75  
   in HDLs, 186–188  
 Floating output node, 117  
 Floating point division (FDIV) bug, 175  
 Floating-gate transistor, 270. *See also* Flash memory  
 Floating-point division (FDIV), 259  
 Floating-point instructions, ARM, 357–358  
 Floating-point numbers, 256–258  
   addition, 259  
   formats, single- and double-precision, 258  
   in programming. *See* Double, C type; Float, C type  
   rounding, 259  
   special cases  
     infinity, 258  
     NaN, 258  
 Floating-Point Status and Control Register (FPSCR), 358  
 Floating-point unit (FPU), 259  
 For loops, 312–313, 541.e20

Format conversion (`atoi`, `atol`, `atof`), 541.e41–541.e42  
 Forwarding, 432–435. *See also* Hazards  
 FPGAs. *See* Field programmable gate arrays (FPGAs)  
 FPU. *See* Floating-point unit (FPU)  
 FPSCR. *See* Floating-Point Status and Control Register (FPSCR)  
 Frequency shift keying (FSK), 531.e42 and GFSK waveforms, 531.e42  
 Front porch, 531.e37  
 FSK. *See* Frequency shift keying (FSK)  
 FSMs. *See* Finite state machines (FSMs)  
 Full adder, 56, 182, 184, 200, 240  
   using `always/process` statement, 200  
 Fully associative cache, 499–500  
*funct* field, 330, 333  
 Function calls, 317, 541.e15–541.e16  
   additional arguments and local variables, 328–329  
   arguments, 319, 541.e15  
   leaf, 324–326  
   multiple registers, loading and storing, 322  
   naming conventions, 541.e16  
   with no inputs or outputs, 318, 541.e15  
   nonleaf, 324–326  
   preserved registers, 322–324  
   prototypes, 541.e16  
   recursive, 326–328  
   return, 318–319, 541.e15  
   stack, use of, 320–322. *See also* Stack  
 Furber, Steve, 473  
 Fuse-programmable ROM, 269–270

## G

Gates  
 AND, 20, 22, 128  
   buffer, 20  
   multiple-input, 21–22  
 NAND, 21, 31  
 NOR, 21–22, 111, 128  
 NOT, 20  
 OR, 21  
   transistor-level. *See* Transistors  
 XNOR, 21  
 XOR, 21

General-purpose I/O (GPIO), 531.  
 e8–531.e11  
 switches and LEDs example, 531.e8  
 Generate signal, 241, 243  
 Genwaves function, 531.e27  
 Glitches, 92–95  
 Global data segment, 340  
 GPIO. *See* General-purpose I/O (GPIO)  
 Graphics accelerators, 469  
 Graphics processing units (GPUs), 460  
 Gray, Frank, 76  
 Gray codes, 76  
 Ground (GND), 22  
 symbol for, 31

**H**

Half adder, 240, 240  
 Hard disk, 490–491. *See also* Hard drive  
 drive  
 Hard drive, 490, 508. *See also* Hard disk, Solid state drive (SSD), Virtual memory  
 Hardware description languages (HDLs), 443–456. *See also* SystemVerilog, VHSIC Hardware Description Language (VHDL)  
 2:1 multiplexer, 452  
 adder, 450  
 capacity, 505  
 combinational logic, 174, 198  
 bitwise operators, 177–179  
 blocking and nonblocking assignments, 205–209  
 case statements, 201–202  
 conditional assignment, 181–182  
 delays, 188–189  
 data memory, 455  
 data types, 213–217  
 history of, 174–175  
 if statements, 202–205  
 internal variables, 182–184  
 numbers, 185  
 operators and precedence, 184–185  
 reduction operators, 180–181  
 immediate extension, 451  
 instruction memory, 455–456  
 modules, 173–174

parameterized modules, 217–220  
 processor building blocks, 449–452  
 register file, 450  
 resettable flip-flop, 451  
 resettable flip-flop with enable, 452  
 sequential logic, 193–198, 209–213  
 simulation and synthesis, 175–177  
 single-cycle ARM processor, 443–456  
 structural modeling, 190–193  
 testbench, 220–224, 452–453  
 top-level module, 454  
 Hardware handshaking, 531.e18  
 Hardware reduction, 70–71. *See also* Equation minimization  
 Hazard unit, 432–435  
 Hazards. *See also* Hazard unit  
 control hazards, 432, 437–440  
 data hazards, 432–436  
 pipelined processor, 431–441  
 read after write (RAW), 431, 464  
 solving  
 control hazards, 437–440  
 forwarding, 432–434  
 stalls, 435–436  
 write after read (WAR), 464  
 write after write (WAW), 465  
 H-bridge control, 531.e45  
 HDL. *See* Hardware description languages (HDLs), SystemVerilog; VHSIC Hardware Description Language (VHDL)  
 Heap, 340  
 Heterogeneous multiprocessors, 469–470  
 Hexadecimal numbers, 11–13  
 Hexadecimal to binary and decimal conversion, 11, 12  
 Hierarchy, 6  
 HIGH, 23. *See also* 1, ON  
 High-level programming languages, 303, 541.e2  
 compiling, assembling, and loading, 339–345  
 translating into assembly, 300  
 High-performance microprocessors, 456  
 Hit, 490  
 Hit rate, 491  
 Hold time constraint, 142–148  
 with clock skew, 149–151  
 Hold time violations, 145, 146, 147–148, 150–151

Homogeneous multiprocessors, 468–469  
 Hopper, Grace, 340

## I

I/O. *See* Input/output (I/O) systems  
 IA-32 architecture. *See* x86  
 IA-64, 368  
 ICs. *See* Integrated circuits (ICs)  
 Idempotency theorem, 62  
 Identity theorem, 62  
 Idioms, 177  
 if statements  
 in ARM assembly, 309–310  
 in C, 541.e17  
 in HDL, 202–205  
 if/else statements, 310, 541.e27  
 in ARM assembly, 310–311  
 in C, 541.e17–541.e18  
 in HDL, 202–205  
 ILP. *See* Instruction level parallelism (ILP)  
 IM. *See* Instruction memory  
 imm8 field, 330–331  
 imm12 field, 333  
 imm24 field, 334  
 Immediate addressing, 336  
 Immediate extension, 451  
 Immediates, 300–301, 330–332, 345–346. *See also* Constants  
 Implicit leading one, 257  
 Information, amount of, 8  
 Initializing  
 arrays in C, 541.e23–541.e24  
 variables in C, 541.e11  
 Input/Output (I/O) systems, 531.  
 e1–531.e64  
 device driver, 531.e3, 531.e6–531.e8  
 embedded I/O systems. *See* Embedded I/O (input/output) systems  
 I/O registers, 531.e3  
 memory-mapped I/O, 531.e1–531.e3  
 personal computer I/O systems. *See* Personal computer (PC) I/O systems  
 Input/output elements (IOEs), 275  
 Institute of Electrical and Electronics Engineers (IEEE), 257–258



- Instruction encoding, x86, 364–367, 366
  - Instruction formats, ARM, 328
    - addressing modes, 336
    - branch instructions, 334–335
    - data-processing instructions, 329–333
    - interpreting, 336–337
    - memory instructions, 333–335
    - stored program, 337–338
  - Instruction formats, x86, 364–367
  - Instruction level parallelism (ILP), 465, 467, 468
  - Instruction memory, 387, 427, 455
  - Instruction register (IR), 407, 414
  - Instruction set, 295
    - for ARM, 386
  - Instruction set. *See also* Architecture
  - Instructions, x86, 360–368
  - Instructions, ARM, 295–360, 535–540
    - branch instructions, 308–309, 539
    - condition flags, 306–308, 540
    - data-processing instructions, 535
    - logical, 303–304, 536–537
    - memory instructions, 301–303, 313–317, 333–334, 538
    - miscellaneous instructions, 539
    - multiply instructions, 305–306, 537
    - shift instructions, 304–305
  - Instructions per cycle (IPC), 390
  - Integrated circuits (ICs), 533.e17
  - Intel. *See* x86
  - Intel processors, 360
  - Intel x86. *See* x86
  - Interrupts, 347, 531.e32
  - Invalid logic level, 186
  - Inverters, 20, 119, 178. *See also* NOT gate
    - cross-coupled, 109, 110
    - in HDL, 178, 199
  - An Investigation of the Laws of Thought* (Boole), 8
  - Involution theorem, 62
  - IOEs. *See* Input/output elements (IOEs)
  - IPC. *See* Instructions per cycle (IPC)
  - IR. *See* Instruction register (IR)
  - IRWrite, 407, 414
- J**
- Java, 303. *See also* Language
- K**
- Karnaugh, Maurice, 75
  - Karnaugh maps (K-maps), 75–84, 93–95, 126
    - logic minimization using, 77–83
    - prime implicants, 65, 77–81, 94–95
    - seven-segment display decoder, 79–81
      - with “don’t cares”, 81–82
  - Kilobit (Kb/Kbit), 14
  - Kilobyte (KB), 14
  - K-maps. *See* Karnaugh maps (K-maps)
- L**
- LAB. *See* Logic array block (LAB)
  - Land grid array, 531.e58
  - Language. *See also* Instructions
    - assembly, 296–303
    - machine, 329–338
    - mnemonic, 297
  - Last-in-first-out (LIFO) queue, 320. *See also* Stack
  - Latches, 111–113
    - comparison with flip-flops, 109, 118
    - D, 113, 120
    - SR, 111–113, 112
      - transistor-level, 116–117
  - Latency, 157–160, 425, 435
  - Lattice, silicon, 27
  - LCDs. *See* Liquid crystal displays (LCDs)
  - LDR, 301–303, 313–317, 333–334, 391–394, 538
    - critical paths for, 402
  - Leaf function, 324
  - Leakage current, 34
  - Least recently used (LRU) replacement, 502–503
    - two-way associative cache with, 502–503, 503
  - Least significant bit (lsb), 13, 14
  - Least significant byte (LSB), 13, 14, 301
  - LEs. *See* Logic elements (LEs)
  - Level-sensitive latch. *See* La\_Hlt414277542tches: D
  - LIFO. *See* Last-in-first-out (LIFO) queue
  - Line options, compiler and command, 341–343, 541.e43–541.e45
  - Linked list, 541.e33–541.e34
  - Linker, 340–341
  - Linking, 339
  - Linux, 531.e23–531.e24
  - Liquid crystal displays (LCDs), 531.e33–531.e36
  - Literal, 58, 96
    - loading, 345–346
  - Little-endian bus order in HDL, 178
  - Little-endian memory addressing, 303
  - Load register instruction (LDR), 301–302
  - Loading literals, 345–346
  - Loads, 344–345
    - base addressing of, 336
  - Local variables, 328–329
  - Locality, 488
  - Logic
    - bubble pushing, 71–73
    - combinational. *See* Combinational logic
    - families, 25–26, 533.e15–533.e17, 533.e15, 533.e17
    - gates. *See* Gates
    - hardware reduction, 70–71
    - multilevel. *See* Multilevel
      - combinational logic
    - programmable, 533.e2–533.e9
    - sequential. *See* Sequential logic
    - transistor-level. *See* Transistors
    - two-level, 69
  - Logic array block (LAB), 276
  - Logic arrays, 271–280. *See also* Field programmable gate arrays (FPGAs), Programmable logic arrays (PLAs)
    - transistor-level implementation, 279–280
  - Logic elements (LEs), 275–279
    - of Cyclone IV, 276–277
    - functions built using, 277–279
  - Logic families, 25–26, 533.e15–533.e17, 533.e15, 533.e17
    - compatibility of, 26
    - logic levels of, 25
    - specifications, 533.e15, 533.e17
  - Logic gates, 19–22, 179, 533.e2
    - AND. *See* AND gate
    - AND-OR (AO) gate, 46
      - with delays in HDL, 189
    - multiple-input gates, 21–22
    - NAND. *See* NAND gate
    - NOR. *See* NOR gate

- Logic gates (*Continued*)
    - NOT. *See* NOT gate
    - OR. *See* OR gate
    - OR-AND-INVERT (OAI) gate, 46
    - XNOR. *See* XNOR gate
    - XOR. *See* XOR gate
  - Logic levels, 22–26
  - Logic simulation, 175–176
  - Logic synthesis, 176–177, 176
  - Logical instructions, 303–304
  - Logical shifter, 251
  - Lookup tables (LUTs), 270, 275–276
  - Loops, 312–313, 541.e19–541.e20
    - in ARM assembly
      - for, 312–313
      - while, 312
    - in C
      - do/while, 541.e19–541.e20
      - for, 541.e20
      - while, 541.e19
  - Lovelace, Ada, 338
  - LOW, 23. *See also* 0, FALSE
  - Low Voltage CMOS Logic (LVCMOS), 25
  - Low Voltage TTL Logic (LVTTTL), 25
  - lsb. *See* Least significant bit (lsb)
  - LSB. *See* Least significant byte (LSB)
  - LSL, 304
  - LSR, 304
  - LUTs. *See* Lookup tables (LUTs)
  - LVCMOS. *See* Low Voltage CMOS Logic (LVCMOS)
  - LVTTTL. *See* Low Voltage TTL Logic (LVTTTL)
- M**
- MAC. *See* Multiply-accumulate (MAC)
  - Machine code. *See* Machine language
  - Machine language, 329
    - addressing modes, 336
    - branch instructions, 334–335
    - data-processing instructions, 329–333
    - interpreting, 336–337
    - memory instructions, 333–335
    - stored program, 337–338, 338
    - translating to assembly language, 337
  - Magnitude comparator, 247
  - Main Decoder, 398–400, 400
  - Main FSM, 413–423, 423
  - main function in C, 541.e3
  - Main memory, 489–491
  - malloc function, 541.e32
  - Mantissa, 257
  - Master-slave flip-flop. *See* Flip-flops
  - Masuoka, Fujio, 270
  - math.h, C library, 541.e42–541.e43
  - Max-delay constraint. *See* Setup time constraint
  - Maxterms, 58
  - MCUs. *See* Microcontroller units (MCUs)
  - Mealy machines, 123, 123, 132–134
    - state transition and output table, 134
    - state transition diagrams, 133
    - timing diagrams for, 135
  - Mean time between failure (MTBF), 153–154
  - Medium-scale integration (MSI) chips, 533.e2
  - MemWrite, 394, 397
  - Memory, 313. *See also* Memory arrays
    - access time, 491
    - addressing modes, 363
    - area and delay, 267–268
    - big-endian, 302
    - byte-addressable, 301–303
    - bytes and characters, 315–317
    - HDL for, 272, 273, 455–456
    - hierarchy, 490
    - little-endian, 303
    - logic using, 270–271
    - main, 490
    - operands in, 301–303
    - physical, 509
    - ports, 265–266
    - protection, 515. *See also* Virtual memory
    - types, 266–270
      - DDR, 268
      - DRAM, 266–267
      - flash, 270
      - register file, 268
      - ROM, 268–270
      - SRAM, 266
    - virtual, 490. *See also* Virtual memory
  - Memory address computation, 419
  - data flow during, 419
  - Memory and peripheral interface, 531.e55–531.e57
  - Memory arrays, 264–271. *See also* Memory
    - bit cell, 264–270
    - HDL for, 272, 273, 455–456
    - logic using, 270–271
    - organization, 264–265
  - Memory hierarchy, 490–491
  - Memory instructions, 301–303, 313–317, 333–334, 391–394
    - encodings, 333–334, 538
  - Memory interface, 487–488
  - Memory map, ARM, 339–340, 531.e2
  - Memory performance. *See* Average Memory Access Time (AMAT)
  - Memory protection, 515
  - Memory systems, 487
    - ARM, 507–508
    - performance analysis, 491–492
    - x86, 531.e3
  - Memory-mapped I/O, 531.e1–531.e3, 531.e7
    - address decoder, 531.e1, 531.e2
    - communicating with I/O devices, 531.e2
    - hardware, 531.e2, 531.e2, 531.e3
  - MemtoReg, 396, 397
  - Metal-oxide-semiconductor field effect transistors (MOSFETs), 26
    - switch models of, 30
  - Metastability, 151–157
    - metastable state, 110, 151
    - resolution time, 151–152, 154–157
    - synchronizers, 152–154
  - Microarchitecture, 296, 385, 388–389.
    - See also* Architecture advanced. *See* Advanced microarchitecture architectural state. *See* Architectural state description of, 385–389 design process, 386–388 evolution of, 470–476 HDL representation, 443–456 generic building blocks, 449–452 single-cycle processor, 444–449 testbench, 452–456 multicycle processor. *See* Multicycle ARM processor

- performance analysis, 389–390.
    - See also Performance analysis
  - pipelined processor. See Pipelined ARM processor
  - real-world perspective, 470–476
  - single-cycle processor. See Single-cycle ARM processor
  - Microcontroller, 531.e3, 531.e25
  - Microcontroller peripherals, 531.e32–531.e53
    - Bluetooth wireless communication, 531.e42–531.e43
    - character LCD, 531.e33–531.e36
      - control, 531.e35–531.e36
      - parallel interface, 531.e33
    - motor control, 531.e43–531.e53
    - VGA monitor, 531.e36–531.e42
  - Microcontroller units (MCUs), 531.e3
  - Micro-operations (micro-ops), 458–459
    - designers, 456
    - high-performance, 456
  - Microprocessors, 3, 13, 295
    - architectural state of, 338
  - Millions of instructions per second, 425
  - Min-delay constraint. See Hold time constraint
  - Minterms, 58
  - Miss, 490–492, 505
    - capacity, 505
    - compulsory, 505
    - conflict, 498, 505
  - Miss penalty, 500
  - Miss rate, 491–492
    - and access times, 492
  - Misses
    - cache, 490
      - capacity, 505
      - compulsory, 505
      - conflict, 505
    - page fault, 509–510
  - ModR/M byte, 366
  - Modularity, 6
  - Modules, in HDL
    - behavioral and structural, 173–174
    - parameterized modules, 217–220
  - Moore, Gordon, 30
  - Moore machines, 123, 132
    - state transition and output table, 134
    - state transition diagrams, 133
    - timing diagrams for, 135
  - Moore's law, 30
  - MOS transistors. See Metal-oxide-semiconductor field effect transistors (MOSFETs)
  - MOSFET. See Metal-oxide-semiconductor field effect transistors (MOSFETs)
  - Most significant bit (msb), 13, 14
  - Most significant byte (MSB), 13, 14, 301, 302
  - Motors
    - DC, 531.e43, 531.e44–531.e47
    - H-bridge, 531.e45–531.e46, 531.e45, 531.e46
    - servo, 531.e44, 531.e48–531.e49
    - stepper, 531.e44, 531.e49–531.e53
  - MOV, 301
  - MPSSE. See Multi-Protocol Synchronous Serial Engine (MPSSE)
  - msb. See Most significant bit (msb)
  - MSB. See Most significant byte (MSB)
  - MSI chips. See Medium-scale integration (MSI) chips
  - MTBF. See Mean time between failure (MTBF)
  - Multicycle ARM processor, 406
    - control, 413–421
    - datapath, 407–413
      - B instruction, 412–413
      - data-processing instructions, 412
      - LDR instruction, 407–410
      - STR instruction, 411–412
    - performance, 421–425
  - Multicycle microarchitectures, 388
  - Multilevel combinational logic, 69–73.
    - See also Logic
  - Multilevel page tables, 516–518
  - Multiple-output circuit, 68–69
  - Multiplexers, 83–86
    - definition of, 83–84
    - HDL for
      - behavioral model of, 181–183
      - parameterized N-bit, 218–219
      - structural model of, 190–193
    - logic using, 84–86
    - symbol and truth table, 83
  - Multiplicand, 252–253
  - Multiplication. See Multiplier
  - Multiplier, 252–253
    - HDL for, 253
  - Multiply instructions, 305–306, 537, 537
  - Multiply and multiply-accumulate instructions, 355–356
  - Multiply-accumulate (MAC), 352, 356
  - Multiprocessors, 468–470
    - chip, 468
    - heterogeneous, 469–470
    - homogeneous, 468
  - Multi-Protocol Synchronous Serial Engine (MPSSE), 531.e63
  - Multithreaded processor, 467
  - Multithreading, 467–468
  - Mux. See Multiplexers
  - myDAQ, 531.e62–531.e63
- ## N
- NAND (7400), 533.e3
  - NAND gate, 21
    - CMOS, 31–32
  - Nested if/else statement, 311, 541.e18
  - Newton computer, 472
  - Nibbles, 13–14
  - nMOS transistors, 28–31, 29–30
  - Noise margins, 23–26, 23
    - calculating, 23–24
  - Nonarchitectural state, 386, 388
  - Nonblocking and blocking assignments, 199–200, 205–209
  - Nonleaf function calls, 324–326
  - Nonpreserved registers, 322–323, 326
  - NOP, 346, 431
  - NOR gate, 21–22, 63, 533.e3
    - chip (7402), 533.e3
    - CMOS, 32
    - pseudo-nMOS logic, 33
    - truth table, 22
  - Not a number (NaN), 258
  - NOT gate, 20
    - chip (7404), 533.e3
    - CMOS, 31
  - Noyce, Robert, 26
  - Null element theorem, 62
  - Number conversion
    - binary to decimal, 10–11
    - binary to hexadecimal, 12
    - decimal to binary, 11, 13
    - decimal to hexadecimal, 13
    - hexadecimal to binary and decimal, 11, 12
    - taking the two's complement, 16

Number systems, 9–19  
 binary, 9–11, 10–11  
 comparison of, 18–19, 19  
 estimating powers of two, 14  
 fixed-point, 255, 255–256  
 floating-point, 256–259  
 addition, 259, 260  
 special cases, 258  
 hexadecimal, 11–13, 12  
 negative and positive, 15  
 sign/magnitude, 15–16  
 signed, 15–18  
 two's complement, 16–18  
 unsigned, 9–11

## O

Odds and ends, 345  
 exceptions, 346–350  
 loading literals, 345–346  
 NOP, 346  
 OFF, 26, 30  
 Offset, 302, 392, 408  
 Offset indexing, ARM, 314  
 ON, 26, 30  
 One-bit dynamic branch predictor, 460  
 One-cold encoding, 130  
 One-hot encoding, 129–131  
 One-time programmable (OTP), 533.e2  
*op* field, 330  
 Opcode. *See op* field  
 Operands  
 ARM, 298  
 constants/immediates, 300–301  
 memory, 301–303  
 registers, 299  
 register set, 300  
 x86, 362–363, 363  
 Operation code. *See op* field  
 Operators  
 in C, 541.e11–541.e14  
 in HDL, 177–185  
 bitwise, 177–181  
 precedence, 185  
 reduction, 180–181  
 table of, 185  
 ternary, 181–182  
 OR gate, 21  
 OR-AND-INVERT (OAI) gate, 46

ORR (OR), 303–304  
 OTP. *See* One-time programmable (OTP)  
 Out-of-order execution, 466  
 Out-of-order processor, 463–465  
 Output dependence, 465  
 Overflow  
 with addition, 15  
 detection, 250–251  
 Oxide, 28

## P

Packages, chips, 533.e17–533.e18  
 Page fault, 509  
 Page number, 511  
 Page offset, 511  
 Page table, 510–513  
 Pages, 509  
 Paging, 516  
 Parallel I/O, 531.e11  
 Parallelism, 157–160  
 Parity gate. *See* XOR gate  
 Partial products, 252  
 Pass by reference, 541.e22  
 Pass by value, 541.e22  
 Pass gate. *See* Transmission gates  
 PC. *See* Program counter (PC)  
 PC Logic, 400  
 PCB. *See* Printed circuit boards (PCBs)  
 PCI. *See* Peripheral Component Interconnect (PCI)  
 PCI express (PCIe), 531.e60  
 PC-relative addressing, 335, 336  
 PCSrc, 394, 395–396, 440  
 PCWrite, 410  
 Perfect induction, proving theorems  
 using, 64–65  
 Performance analysis, 389–390  
 multicycle ARM processor, 422–424  
 pipelined ARM processor, 425–428  
 processor comparison, 424  
 single-cycle ARM processor, 402  
 Performance Analysis, 389–390.  
*See also* Average Memory Access Time (AMAT)  
 Peripheral Component Interconnect (PCI), 531.e59–531.e60  
 Peripherals devices. *See* Input/output (I/O) systems

Personal computer (PC) I/O systems, 531.e57–531.e64  
 data acquisition systems, 531.e62–531.e63  
 DDR3 memory, 531.e60–531.e61  
 networking, 531.e61  
 PCI, 531.e59–531.e60  
 SATA, 531.e61–531.e62  
 USB, 531.e59, 531.e63–531.e64  
 Phase locked loop (PLL), 531.e39  
 Physical memory, 509  
 Physical page number (PPN), 511  
 Physical pages, 509  
 Pipelined ARM processor, 425–428  
 abstract view of, 427  
 control unit, 430  
 datapath, 428–429  
 description, 425–428  
 hazards, 431–441  
 performance analysis, 441–443  
 throughput, 426  
 Pipelined microarchitecture. *See* Pipelined ARM processor  
 Pipelining, 158–160  
 PLAs. *See* Programmable logic arrays (PLAs)  
 Plastic leaded chip carriers (PLCCs), 533.e17  
 Platters, 508  
 PLCCs. *See* Plastic leaded chip carriers (PLCCs)  
 PLDs. *See* Programmable logic devices (PLDs)  
 PLL. *See* Phase locked loop (PLL)  
 pMOS transistors, 28–31, 29  
 Pointers, 541.e21–541.e23, 541.e25, 541.e28, 541.e30, 541.e32  
 POS. *See* Product-of-sums (POS) form  
 Positive edge-triggered flip-flop, 114  
 Post-indexed addressing, ARM, 314  
 Power consumption, 34–35  
 Power-saving and security instructions, 358  
 PPN. *See* Physical page number (PPN)  
 Prefix adders, 243–245, 244  
 Prefix tree, 245  
 Pre-indexed addressing, ARM, 314  
 Preserved registers, 322–324, 323  
 Prime implicants, 65, 77  
 Printed circuit boards (PCBs), 533.e19–533.e20  
 printf, 541.e35–541.e37

- Priority
    - circuit, 68–69
    - encoder, 102–103, 105
  - Procedure calls. *See* Function calls
  - Processor performance comparison, 442
    - multicycle ARM processor, 424
    - pipelined ARM processor, 442
    - single-cycle processor, 405
  - Processor-memory gap, 489
  - Product-of-sums (POS) form, 60
  - Program counter (PC), 308, 338, 387, 394
  - Programmable logic arrays (PLAs), 67, 272–274, 533.e6–533.e7
    - transistor-level implementation, 280
  - Programmable logic devices (PLDs), 533.e6
  - Programmable read only memories (PROMs), 269, 271, 533.e2–533.e6
  - Programming
    - in ARM, 303
    - arrays. *See* Arrays
    - branching. *See* Branching
    - in C. *See* C programming
    - conditional statements, 309–312
    - condition flags, 306–308
    - constants. *See* Constants; *Immediates*
    - function calls. *See* Function calls
    - getting loopy, 312–313
    - logical and arithmetic instructions, 303–306
    - loops. *See* Loops
    - memory, 313–317
    - shift instructions, 304–305
  - PROMs. *See* Programmable read only memories (PROMs)
  - Propagate signal, 241
  - Propagation delay, 88–92. *See also* Critical path
  - Pseudoinstructions, 346
  - Pseudo-nMOS logic, 33–34, 33
    - NOR gate, 33
    - ROMs and PLAs, 279–280
  - Pulse-Width Modulation (PWM), 531.e28–531.e31
    - analog output with, 531.e30–531.e31
    - duty cycle, 531.e28
    - signal, 531.e28
  - PWM. *See* Pulse-Width Modulation (PWM)
- Q**
- Quiescent supply current, 34
- R**
- Race conditions, 119–120, 120
    - rand, 541.e40–541.e41
  - Random access memory (RAM), 266–268, 271, 272
  - Raspberry Pi, 531.e3–531.e4, 531.e5, 531.e6, 531.e32, 531.e48–531.e49
  - RAW hazard. *See* Read after write (RAW) hazard
  - Rd field, 330
  - Read after write (RAW) hazard, 431, 464. *See also* Hazards
  - Read only memory (ROM), 266, 268–270
    - transistor-level implementation, 279–280
  - Read/write head, 508
  - ReadData bus, 393, 394
  - Receiver gate, 22
  - Recursive function calls, 326–328
  - Reduced instruction set computer (RISC) architecture, 298, 458
  - Reduction operators, 180–181
  - Register file (RF)
    - ARM register descriptions, 299
    - HDL for, 449
    - in pipelined ARM processor (write on falling edge), 428
    - schematic, 268
    - use in ARM processor, 387
  - Register renaming, 465–467
  - Register set, 300. *See also* Register file (RF)
  - Registers. *See* ARM registers; Flip-flops; x86 registers
    - loading and storing, 322
    - preserved and nonpreserved, 322–324
  - RegSrc, 402
  - Regularity, 6
  - RegWrite, 393, 433
  - Replacement policies, 516
  - Resettable flip-flops, 116
  - Resettable registers, 194–196
  - Resolution time, 151–152. *See also* Metastability
    - derivation of, 154–157
  - Return value, 317
  - RF. *See* Register file (RF)
  - Ring oscillator, 119, 119
  - Ripple-carry adder, 240, 240–241, 243
  - RISC architecture. *See* Reduced instruction set computer (RISC) architecture
  - Rising edge, 88
  - Rm field, 330
  - Rn field, 330
  - ROM. *See* Read only memory (ROM)
  - ROR, 304
  - rot field, 330–331
  - Rotations per minute (RPM), 531.e44
  - Rotators, 251–252
  - Rounding modes, 259
  - RPM. *See* Rotations per minute (RPM)
  - RS-232, 531.e18
- S**
- Sampling, 141
  - Sampling rate, 531.e25
  - SATA. *See* Serial ATA (SATA)
  - Saturated arithmetic, 353
  - Scalar processor, 461–463, 460
  - Scan chains, 262–263
    - scanf, 541.e38
  - Scannable flip-flop, 262–263
  - Schematics, rules of drawing, 31, 67
  - SCK. *See* Serial Clock (SCK)
  - SDI. *See* Serial Data In (SDI)
  - SDO. *See* Serial Data Out (SDO)
  - SDRAM. *See* Synchronous dynamic random access memory (SDRAM)
  - Segment descriptor, 367
  - Segmentation, 367
  - Selected signal assignment statements, 182
  - Semiconductors, 27
    - industry, sales, 3
  - Sequencing overhead, 143–144, 149, 160, 442
  - Sequential building blocks. *See* Sequential logic

- Sequential logic, 109–161, 259–263
  - counters, 260
  - finite state machines. *See* Finite state machines (FSMs)
  - flip-flops, 114–118. *See also* Registers
  - latches, 111–113
    - D, 113
    - SR, 111–113
  - registers. *See* Registers
  - shift registers, 261–263
  - timing of. *See* Timing analysis
- Serial ATA (SATA), 531.e62
- Serial Clock (SCK), 531.e12
- Serial communication, with PC, 531.e20
- Serial Data In (SDI), 531.e12
- Serial Data Out (SDO), 531.e12
- Serial I/O, 531.e11–531.e23
  - SPI. *See* Serial peripheral interface (SPI)
  - UART. *See* Universal Asynchronous Receiver Transmitter (UART)
- Serial Peripheral Interface (SPI), 531.
  - e11, 531.e12–531.e17
  - connection between PI and FPGA, 531.e14
  - ports
    - Serial Clock (SCK), 531.e12
    - Serial Data In (SDI), 531.e12
    - Serial Data Out (SDO), 531.e12
  - register fields in, 531.e13
  - slave circuitry and timing, 531.e15
  - waveforms, 531.e12
- Servo motor, 531.e44, 531.e48–531.e49
- Set bits, 495
- Setup time constraint, 142, 145–147
  - with clock skew, 148–150
- Seven-segment display decoder, 79–82
  - with don't cares, 82–83
  - HDL for, 201–202
- Shaft encoder, 531.e43, 531.e47–531.
  - e48, 531.e48
- Shift instructions, 304–305, 305
- Shift registers, 261–263
- Shifters, 251–252
- Short path, 89–92
- Sign bit, 16
- Sign extension, 18
- Sign/magnitude numbers, 15–16, 256
- Signed binary numbers, 15–19
- Signed multiplier, 217
- Silicon dioxide (SiO<sub>2</sub>), 28
- Silicon lattice, 27
- SIMD. *See* Single instruction multiple data (SIMD)
- SIMD instructions, 358–360
  - simple function, 318
- Simple programmable logic devices (SPLDs), 274
- Simulation waveforms, 176
  - with delays, 189
- Single instruction multiple data (SIMD), 460, 472
- Single-cycle ARM processor, 390, 444
  - Conditional Logic, 447–448
  - control, 397–401
  - controller, 445
  - datapath, 390, 448–449
    - B instruction, 396–397
    - data-processing instructions, 395–396
    - LDR instruction, 391–394
    - STR instruction, 394–396
  - Decoder, 446
  - instructions, 402
  - performance, 402–405
- Single-cycle microarchitecture, 388
- Single-precision formats, 258. *See also* Floating-point numbers
- Skew. *See* Clock skew
- Slash notation, 56
- Slave latch, 114. *See also* Flip-flops
- Small-scale integration (SSI) chips, 533.
  - e2
- Solid state drive (SSD), 490. *See also* Flash memory, Hard drive
- SOP. *See* Sum-of-products (SOP) form
- Spatial locality, 488, 500–502
- Spatial parallelism, 157–158
- SPEC, 389
- SPECINT2000, 424
- SPI. *See* Serial Peripheral Interface (SPI)
- Squashing, 465
- SR latches, 111–113, 112
- SRAM. *See* Static random access memory (SRAM)
- srand, 541.e40–541.e41
- Src2 field, 330, 333
- SSI chips. *See* Small-scale integration (SSI) chips
- Stack, 320–329. *See also* Function calls
  - during recursive function call, 326–328
  - preserved registers, 322–324
- stack frame, 322, 328
- stack pointer (SP), 320
- storing additional arguments on, 328–329
- storing local variables on, 328–329
- Stalls, 435–436. *See also* Hazards
- Standard libraries, 541.e35–541.e43
  - math, 541.e42–541.e43
  - stdio, 541.e35–541.e40
    - file manipulation, 541.e38–541.e40
    - printf, 541.e35–541.e37
    - scanf, 541.e38
  - stdlib, 541.e40–541.e42
    - exit, 541.e41
    - format conversion (atoi, atol, atof), 541.e41–541.e42
    - rand, srand, 541.e40–541.e41
    - string, 541.e43
- State encodings, FSM, 129–131, 134.
  - See also* Binary encoding, One-cold encoding, One-hot encoding
- State machine circuit. *See* Finite state machines (FSMs)
- State variables, 109
- Static branch prediction, 459
- Static discipline, 24–26
- Static power, 34
- Static random access memory (SRAM), 266, 267, 519
- Status flags, 363. *See also* Condition flags
- stdio.h, C library, 541.e35–541.e40.
  - See also* Standard libraries
- stdlib.h, C library, 541.e40–541.e42.
  - See also* Standard libraries
- Stepper motors, 531.e44, 531.e49–531.
  - e53
    - bipolar stepper motor, 531.e49, 531.e50–531.e52
    - half-step drive, 531.e50, 531.e51
    - two-phase-on drive, 531.e50, 531.e51
    - wave drive, 531.e52–531.e53
- Stored program, 337–338
- STR, 394–396
- string.h, C library, 541.e43
- Strings, 316–317, 541.e28–541.e29.
  - See also* Characters (char)
- Structural modeling, 173–174, 190–193
- Structures (struct), 541.e29–541.e31

- SUB, 297
  - Substrate, 28–29
  - Subtraction, 17, 246, 297
  - Subtractor, 246–247
  - Sum-of-products (SOP) form, 58–60
  - Superscalar processor, 461–463
  - Supervisor call (SVC) instruction, 349
  - Supply voltage, 22. *See also*  $V_{DD}$
  - SVC. *See* Supervisor call (SVC) instruction
  - Swap space, 516
  - switch/case statements
    - in ARM assembly, 311–312
    - in C, 541.e17–541.e18
    - in HDL. *See* case statement, in HDL
  - Symbol table, 342, 343
  - Symmetric multiprocessing (SMP), 468.
    - See also* Homogeneous multiprocessors
  - Synchronizers, 152–154, 152–153
  - Synchronous circuits, 122–123
  - Synchronous dynamic random access memory (SDRAM), 268
    - DDR, 268
  - Synchronous logic, design, 119–123
  - Synchronous resettable flip-flops, 116
  - Synchronous sequential circuits, 120–123, 122. *See also* Finite state machines (FSMs)
    - timing specification. *See* Timing analysis
  - SystemVerilog, 173–225. *See also* Hardware description languages (HDLs)
    - accessing parts of busses, 188, 192
    - bad synchronizer with blocking assignments, 209
    - bit swizzling, 188
    - blocking and nonblocking assignment, 199–200, 205–208
    - case statements, 201–202, 205
    - combinational logic using, 177–193, 198–208, 217–220
    - comments, 180
    - conditional assignment, 181–182
    - data types, 213–217
    - decoders, 202–203, 219
    - delays (in simulation), 189
    - divide-by-3 FSM, 210–211
    - finite state machines (FSMs), 209–213
    - Mealy FSM, 213
    - Moore FSM, 210, 212
    - full adder, 184
      - using always/process, 200
      - using nonblocking assignments, 208
    - history of, 175
    - if statements, 202–205
    - internal signals, 182–184
    - inverters, 178, 199
    - latches, 198
    - logic gates, 177–179
    - multiplexers, 181–183, 190–193, 218–219
    - multiplier, 217
    - numbers, 185–186
    - operators, 185
    - parameterized modules, 217–220
      - $N:2^N$  decoder, 219
      - $N$ -bit multiplexers, 218–219
      - $N$ -input AND gate, 220
    - priority circuit, 204
      - using don't cares, 205
    - reduction operators, 180–181
    - registers, 193–197
      - enabled, 196
      - resettable, 194–196
    - sequential logic using, 193–198, 209–213
    - seven-segment display decoder, 201
    - simulation and synthesis, 175–177
    - structural models, 190–193
    - synchronizer, 197
    - testbench, 220–224
      - self-checking, 222
      - simple, 221
      - with test vector file, 223–224
    - tristate buffer, 187
    - truth tables with undefined and floating inputs, 187, 188
    - z's and x's, 186–188, 205
- T**
- Tag, 495
  - Taking the two's complement, 16–17
  - Temporal locality, 488, 493–494, 497, 502
  - Temporal parallelism, 158–159
  - Temporary registers, 299
  - Ternary operators, 181, 541.e13
  - Testbench, 452–456
  - Testbenches, HDLs, 220–224
    - self-checking, 221–222
    - simple, 220–221
    - with testvectors, 222–224
  - Text Segment, 340, 344
  - Thin small outline package (TSOP), 533. e17
  - Thread level parallelism (TLP), 467
  - Threshold voltage, 29
  - Throughput, 157–160, 388, 425, 468
  - Thumb instruction set, 351–352
  - Timers, 531.e23–531.e24
  - Timing
    - of combinational logic, 88–95
      - delay. *See* Contamination delay; Propagation delay
      - glitches. *See* Glitches
    - of sequential logic, 141–157
      - analysis. *See* Timing analysis
      - clock skew. *See* Clock skew
      - dynamic discipline, 141–142
      - metastability. *See* Metastability
      - resolution time. *See* Resolution time
      - time
      - system timing. *See* Timing analysis
  - Timing analysis, 141–151
    - calculating cycle time. *See* Setup time constraint
    - with clock skew. *See* Clock skew
    - hold time constraint. *See* Hold time constraint
    - max-delay constraint. *See* Setup time constraint
    - min-delay constraint. *See* Hold time constraint
    - multicycle processor, 424
    - pipelined processor, 441
    - setup time constraint. *See* Setup time constraint
    - single-cycle processor, 405
  - TLB. *See* Translation lookaside buffer (TLB)
  - TLP. *See* Thread level parallelism (TLP)
  - Transistors, 26–34
    - bipolar, 26
    - CMOS, 26–33
    - gates made from, 31–34
    - latches and flip-flops, 116–117
    - MOSFETs, 26

- Transistors (*Continued*)
    - nMOS, 28–34, 29–33
    - pMOS, 28–34, 29–33
      - pseudo-nMOS, 33–34
      - ROMs and PLAs, 279–280
      - transmission gate, 33
  - Transistor-Transistor Logic (TTL), 25–26, 533.e15–533.e16
  - Translating and starting a program, 339
  - Translation lookaside buffer (TLB), 514–515
  - Transmission Control Protocol and Internet Protocol (TCP/IP), 531.e61
  - Transmission gates, 33
  - Transmission lines, 533.e20–533.e33
    - characteristic impedance ( $Z_0$ ), 533.e30–533.e31
      - derivation of, 533.e30–533.e31
    - matched termination, 533.e22–533.e24
    - mismatched termination, 533.e25–533.e28
    - open termination, 533.e24–533.e25
    - reflection coefficient ( $k_r$ ), 533.e31–533.e32
      - derivation of, 533.e31–533.e32
    - series and parallel terminations, 533.e28–533.e30
      - short termination, 533.e25
      - when to use, 533.e28
  - Transparent latch. *See* Latches: D
  - Traps, 347
  - Tristate buffer, 74–75, 187
    - HDL for, 186–187
    - multiplexer built using, 84–85, 91–93
  - True, 8, 20–22, 58–59, 70, 74, 111–112, 116, 129, 176, 180, 205
  - Truth tables, 20
    - ALU decoder, 399, 404
    - with don't cares, 69, 81–83, 205
    - multiplexer, 83
    - seven-segment display decoder, 79
    - SR latch, 111, 112
    - with undefined and floating inputs, 187–188
  - TSOP. *See* Thin small outline package (TSOP)
  - TTL. *See* Transistor-Transistor Logic (TTL)
  - Two's complement numbers, 16–18
  - Two-bit dynamic branch predictor, 460
  - Two-cycle latency of LDR, 435
  - Two-level logic, 69
  - typedef, 541.e31–541.e32
- U**
- UART. *See* Universal Asynchronous Receiver Transmitter (UART)
  - Unconditional branches, 308, 309
  - Undefined instruction exception, 347
  - Unicode, 315
  - Unit under test (UUT), 220
  - Unity gain points, 24
  - Universal Asynchronous Receiver Transmitter (UART), 531.e17–531.e23
    - hardware handshaking, 531.e18
  - Universal Serial Bus (USB), 270, 531.e18, 531.e59
    - USB 1.0, 531.e59
    - USB 2.0, 531.e59
    - USB 3.0, 531.e59
  - Unsigned multiplier, 217, 252–253
  - Unsigned numbers, 18
  - Upton, Eben, 531.e4
  - USB. *See* Universal Serial Bus (USB)
  - USB links, 531.e63–531.e64
    - FTDI, 531.e63
    - UM232H module, 531.e64
  - Use bit ( $U$ ), 502
- V**
- Valid bit ( $V$ ), 496
  - Variables in C, 541.e7–541.e11
    - global and local, 541.e9–541.e10
    - initializing, 541.e11
    - primitive data types, 541.e8–541.e9
  - $V_{CC}$ , 23. *See also* Supply voltage,  $V_{DD}$
  - $V_{DD}$ , 22, 23. *See also* Supply voltage
  - Vector processor, 460
  - Verilog. *See* SystemVerilog
  - Very High Speed Integrated Circuits (VHSIC), 175. *See also* VHSIC Hardware Description Language (VHDL)
  - VGA (Video Graphics Array) monitor, 531.e36–531.e42
    - connector pinout, 531.e37
    - driver for, 531.e39–531.e42
  - VHDL. *See* VHSIC Hardware Description Language (VHDL)
  - VHSIC. *See* Very High Speed Integrated Circuits (VHSIC)
  - VHSIC Hardware Description Language (VHDL), 173–175
    - accessing parts of busses, 188, 192
    - bad synchronizer with blocking assignments, 209
    - bit swizzling, 188
    - blocking and nonblocking assignment, 199–200, 205–208
    - case statements, 201–202, 205
    - combinational logic using, 177–193, 198–208, 217–220
    - comments, 180
    - conditional assignment, 181–182
    - data types, 213–217
    - decoders, 202–203, 219
    - delays (in simulation), 189
    - divide-by-3 FSM, 210–211
    - finite state machines (FSMs), 209–213
      - Mealy FSM, 213
      - Moore FSM, 210, 212
    - full adder, 184
      - using always/process, 200
      - using nonblocking assignments, 208
    - history of, 175
    - if statements, 202
    - internal signals, 182–184
    - inverters, 178, 199
    - latches, 198
    - logic gates, 177–179
    - multiplexer, 181–183, 190–193, 218–219
    - multiplier, 217
    - numbers, 185–186
    - operators, 185
    - parameterized modules, 217–220
      - $N:2^N$  decoder, 219
      - $N$ -bit multiplexers, 218, 219
      - $N$ -input AND gate, 220, 220



priority circuit, 204  
     reduction operators, 180–181  
     using don't cares, 205  
 reduction operators, 180–181  
 registers, 193–197  
     enabled, 196  
     resettable, 194–196  
 sequential logic using, 193–198,  
     209–213  
 seven-segment display decoder,  
     201  
 simulation and synthesis, 175–177  
 structural models, 190–193  
 synchronizer, 197  
 testbench, 220–224  
     self-checking, 222  
     simple, 221  
     with test vector file, 223–224  
 tristate buffer, 187  
 truth tables with undefined and  
     floating inputs, 187, 188  
 z's and x's, 186–188, 205  
 Video Graphics Array (VGA). *See* VGA  
     (Video Graphics Array) monitor  
 Virtual address, 509  
     space, 515  
 Virtual memory, 490, 508–518  
     address translation, 509–512  
     cache terms comparison, 510  
     memory protection, 515  
     multilevel page tables, 516–518  
     page fault, 509–510  
     page number, 511  
     page offset, 511  
     pages, 509  
     page table, 512–513

    replacement policies, 516  
     translation lookaside buffer (TLB),  
         514–515  
     write policy, 506–507  
 Virtual page number (VPN), 512  
 Virtual pages, 509  
 V<sub>SS</sub>, 23

## W

Wafers, 28  
 Wait for event (WFE) instruction, 358  
 Wait for interrupt (WFI) instruction, 358  
 Wall, Larry, 20  
 WAR hazard. *See* Write after read  
     (WAR) hazard  
 WAW hazard. *See* Write after write  
     (WAW) hazard  
 Weak pull-up, 33  
 Weird number, 18  
 WFE. *See* Wait for event (WFE)  
     instruction  
 WFI. *See* Wait for interrupt (WFI)  
     instruction  
 while loops, 312, 541.e19  
 White space, 180  
 Whitmore, Georgiana, 7  
 Wi-Fi, 531.e61  
 Wilson, Sophie, 472  
 Wire, 67  
 Wireless communication, Bluetooth,  
     531.e42–531.e43  
 Wordline, 264

Write after read (WAR) hazard, 464.  
     *See also* Hazards  
 Write after write (WAW) hazard,  
     464–465  
 Write policy, 506–507  
     write-back, 506–507  
     write-through, 506–507

## X

X. *See* Contention (x); Don't care (X)  
 x86  
     architecture, 360–368, 362  
         big picture, 368  
         branch conditions, 366  
         instruction encoding, 364–367  
         instructions, 364–367  
         memory addressing modes,  
             363  
         operands, 362–363  
         peculiarities, 368  
         registers, 362  
         status flags, 363  
 Xilinx FPGA, 275  
 XNOR gate, 21–22  
 XOR gate, 21

## Z

Z. *See* Floating (Z)