

# INDEX

- \* (wildcards), 152, 761
- 0-1 knapsack problem, 553
- 1:1 mapping, 329–30
  - area/delay trade-offs, 329
  - PEs, 337
  - pitch matching, 330
  - topology matching, 329–30
- Absorbing boundary conditions (ABC), 702
- Abstract Physical Model (APM), 322
- Abstracted hardware resources, 234–36
- Accelerated PathFinder, 418–22
  - limiting search expansion, 419
  - multi-terminal nets and, 420, 421
  - parallelized, 215–16, 421
  - routing high-fanout nets first, 419
  - scaling sharing/history costs, 419
  - See also* PathFinder
- Accelerated simulated annealing, 415–18
  - communication bandwidth and, 416–17
  - distributed, 415
  - hardware-assisted, 418
  - parallelized, 416
  - See also* Simulated annealing
- Accelerating technology, 56–59
- Actel ProASIC3, 83
- Active Pages, 779–802
  - activation portion, 788
  - algorithmic complexity, 786–94
  - array-insert, 788–90
  - Central Processor, 782, 784–85, 788
  - configurations, 782
  - defect tolerance, 779, 799–801
  - DRAM hardware design, 780
  - execution with parameters, 787
  - hardware interface, 780
  - LCS, 791–94
  - multiplexing performance, 796
  - Page Processor, 781
  - performance results, 781–86
  - performance versus random processor defects, 800
  - processing time, 798
  - processor width performance, 796–97
  - processor-memory nonoverlap, 784–85
  - programming model, 781
  - related work, 801–2
  - speedup over conventional systems, 782–84
- Ad hoc testing, 96
- Adaptive Computing Systems (ACS), 57
- Adaptive lattice structures, 514
- Adaptive nulling, CORDIC algorithm and, 514
- Add/subtract FUs, 531, 532
- Adder trees
  - computation, 598
  - creation, 598
  - template-specific, 596
- Adders, 504
  - floating point implementation, 675–77
  - in reconfigurable dynamic ATR system, 609
- Address indirection, 178
- Advanced Encryption Standard (AES), 459, 775
- A\* heuristic, 373–374
- AIG, 285
- Algebraic layout specification, 352–60
  - calculation, 353
  - case study, 357–60
- Altera SignalTap, 271
- Altera Stratix, 19–23
  - block diagram, 19
  - DSP block, 21
  - LAB structure, 21
  - logic architecture, 19–21
  - logic element, 20
  - MultiTrack, 21–22
  - routing architecture, 21–23
- Altera Stratix-II, 68, 83, 300
  - configuration information, 68
  - horizontal/vertical routing, 308
- Alternative region implementations, 544, 549
  - heterogeneous, 550
  - number of, 550
  - obtaining, 550
  - parallel program partitioning, 557
  - sequential program partitioning, 549–50
  - See also* Hardware/software partitioning
- ALTOR, 313–14, 315

- AMD/Intel, 55–56
- Amdahl's Law, 62, 542
  - equation, 542
  - in hardware/software partitioning, 542, 543
  - solution space pruning, 544
- Amtel AT40K, 70
- Analytic peak estimation, 479–84
  - data range propagation, 482–84
  - LTI system, 479–82
  - See also* Peak estimation
- Analytic placement, 315
- AND gates, 133
- Angle approximation error, 522–23
- Annotations
  - absence of loop-carried memory dependence, 178–79
  - pointer independence, 178
- Antifuse, 17–18
  - use advantages, 18
- Application development, 435–38
  - challenges, 435
  - compute models, 93–107
  - system architectures, 107–25
- Application-specific computation unit, 603–4
- Application-specific integrated circuits. *See* ASICs
- Applications
  - arithmetic implementation, 448–52
  - characteristics and performance, 441–44
  - computational characteristics/ performance, 441–43
  - configure-once implementation, 445
  - embedded, 476
  - implementation strategies, 445–48
  - implementing with FPGAs, 439–52
  - RTR, 446–47
- Architectural space modeling, 816–26
  - efficiency, 817–25
  - raw density from architecture, 816–17
- Area flow, 280
- Area models, 485–96
  - high-level, 493–96
  - intersection mismatch, 823
  - for multiple-wordlength adder, 495
  - width mismatch, 819–21
- Area-oriented mapping, 280–82
- Arithmetic
  - BFP, 450
  - complexity, 442–43
  - distributed, 503–11
  - fixed-point, 448–49
  - implementation, 448–52
  - infinite-precision, 519
- Arithmetic logic units (ALUs), 5, 61, 114, 401
- Array processors, 48, 226–30, 790, 2191–222
- Array-insert algorithm, 788–90
  - processor and Active Pages computations, 790
  - simulation results, 790
- Arrays
  - block reconfigurable, 74–75
  - FPTAs, 745
  - local, 177
  - reconfigurable (RAs), 43
  - See also* FPGAs
- Artificial evolution, 727–29
- ASICs
  - cost, 440
  - debug and verification, 440–41
  - design time, 638
  - development, 440
  - general-purpose hardware implementation, 458
  - power consumption, 440
  - replacement, 2
  - time to market, 439–40
  - vendors, 754
  - verification, 637, 638
- Associativity, 799–800
- Asynchronous transfer mode (ATM)
  - networking, 755
- ATM adaptation layer 5 (AAL5), 758
- ATR, 591–610
  - algorithms, 592–94
  - dynamically reconfigurable designs, 594–600, 604–6
  - FOA algorithm, 592
  - with FPGAs, 591–610
  - implementation methods, 604–7, 608
  - implementations, 604–9
  - Mojave system, 604–6
  - Myrinet system, 606–7
  - reconfigurable computing models, 607–9
  - reconfigurable static design, 600–4
  - in SAR imagery, 591
  - SLD, 592–94
  - statically reconfigurable system, 606–7
- Automated worm detection, 766–67
- Automatic compilation, 162–75, 212–13
  - dataflow graphs, building, 164–69

- DFG optimization, 169–73
- DFG to reconfigurable fabric, 173–75
- hyperblocks, 164
- memory node connections, 175
- operation packing, 173–74
- pipelined scheduling, 174–75
- runtime netlist, 413–14
- scheduling, 174
- TDF, 212
- See also* C for spatial computing
- Automatic HW/SW partitioning, 175–76
- Automatic partitioning trend, 540–42
- Automatic Target Recognition. *See* ATR
- Back-pressure signal, 210
- Backtrack algorithm, 615–17
  - conflict analysis, 625
  - distributed control architecture, 620
  - efficiency, 616
  - FSM, 621–22
  - implementing, 619–24
  - implication circuit, 620
  - improved, 617–18, 626–27
  - improved, implementing, 624–27
  - nonchronological backtracking, 618
  - reconfigurable solver, 618–27
  - static variable ordering, 617
  - terminating conditions, 616
  - variable values, 619
- Basic blocks, 163
- Batcher bitonic sorter, 357–60
- BEE Platform Studio (BPS), 192, 193
  - BEE2 platform, 191–94
  - design flow, 194
  - I/O, 200
- Bellman–Ford algorithm, 386
- Bernoulli’s Law of Large Numbers, 835
- Bidirectional switches, 377
- Binary-level partitioning, 559
- Binding
  - flexible, 236–38
  - install time, 236–37
  - runtime, 237–38
- Bipartitioning, 312, 646
- Bitonic sorter, 357–60
  - ilv* combinator, 359
  - layout and behavior specification, 358
  - merger, 359
  - recursion and layout, 360
  - recursive structure, 357
- Bitops (bit operations), 808
- BLAS routines, 685
- Block floating point (BFP), 450
- Block reconfigurable arrays, 74–75
- BlockRAMs, 585, 708, 713, 766
  - caching modules, 715
  - dual-ported, 716
  - latency, 713
- Bloom filters, 762
  - payload scanning with, 762
  - SIFT used, 766
- Boolean expressions, 464
- Boolean operators, 465
- Boolean satisfiability (SAT), 613–35
  - algorithms, 615–18
  - applications, 614
  - backtrack algorithm, 615–17
  - backtrack algorithm improvement, 617–18
  - clauses, 614
  - CNF, 613
  - complete algorithms, 615
  - formulas, mapping, 634
  - formulation, 282, 613–14
  - incomplete algorithms, 615
  - parallel processing, 618–19
  - problem, 613
  - problem analysis, 618–19
  - test pattern generation, 614, 615
  - See also* SAT solvers
- Booth technique, 495
- BORPH, 197
- Bottom-up structure synthesis, 853
- Bottom-up technology, 855–58
  - crosspoints, 857–58
  - nanowires, 856–57
- Bulk Synchronous Parallelism (BSP), 118–19
- Butterflies, 688
- C++ language, 541
- C compiler flow, 163
- C compiler frontend, 163–64
  - CFG, 163
  - live variable analysis, 163
  - processing procedures, 164
- C for spatial computing, 155–80
  - actual control flow, 159–60
  - automatic compilation, 162–75
  - automatic HW/SW partitioning, 175–76
  - common path optimization, 161–62
  - data connections between operations, 157
  - full pushbutton path benefits, 155–56

- C for spatial computing (*cont.*)
  - hyperblocks, 164
  - if-then-else with multiplexers, 158–59
  - memory, 157–58
  - mixed operations, 157
  - partitioning, 155
  - programmer assistance, 176–80
- C language, 155–159, 171, 179, 541
- C-slow retiming, 390–93, 827
  - architectural change requirement, 395–96
  - benefits, 390
  - FPGA effects on, 391
  - interface, 391
  - latency improvement, 392
  - low-power environment effect, 392
  - memory blocks, 391
  - microprocessor application, 395
  - as multi-threading, 395–98
  - results, 392
  - as threaded design, 391
  - throughput, 392
  - See also* Retiming
- Caches
  - configurations, 83
  - virtually addressed, 397
- CAD
  - JHDL system, 255, 265–68
  - Mentor Graphics, 56
  - PipeRench tools, 34
  - runtime, 411
  - runtime processes, 238
  - Teramac for, 58
  - tools, 44–45, 66
- Cadence Xcite, 642
- Case studies
  - Altera Stratix, 19–23
  - Xilinx Virtex-II Pro, 23–26
- CDFG, *See* Control dataflow graphs
- Cellular automata (CA), 122–23, 702–3
  - folded, 123
  - two-dimensional, 122
  - well-known, 122
- Cellular programming evolutionary
  - algorithm, 738
- Central Limit Theorem, 835
- Centralized evolution, 736–37
- Chameleon architecture, 40–41
  - price/performance, 41
- Channel width, 430
- Checkpointing, 272
- Checksums, 847
- Chimaera architecture, 42–44
  - high-level user design language, 44
  - overview illustration, 43
  - RFUOPs, 43
  - VICs, 43–44
- Choice networks
  - creating, 285
  - mapping on, 286
- Church–Turing Thesis, 96
- Circuit combinators, 352
- Circuit emulation, 54–56, 637–68
  - AMD/Intel, 55–56
  - impacts, 56
  - in-circuit, 650
  - multi-FPGA, 641–44
  - single-FPGA, 640–41
  - system uses, 639–40
  - Virtual Wires, 56
  - VLE, 653–65
- Circuit graph
  - bidirectional switches, 377
  - de-multiplexers, 376–77
  - edges, 377
  - extensions, 376–77
  - model, 367
  - symmetric device inputs, 376
- Circuit layout
  - algebraic specification, 352–60
  - calculation, 353
  - deterministic, 352
  - explicit Cartesian specification, 351–52
  - no and totally explicit, 350
  - problem, 347–51
  - regularity, 319
  - specifying, 347–63
  - verification for parameterized designs, 360–62
- CLAP tool, benchmarks, 336
- Clause modules, 629–30
- Clearspeed SIMD array, 221
- Clock cycles
  - for circuit mapping, 649
  - latency, 507
  - $N/L$ , 510
  - packing operations into, 173–74
  - reducing number of, 506
- Clock frequency, 506
- Cloning, 54
- Clustering, 213, 227, 228, 304–6
  - benefits, 304
  - goals, 304
  - iRAC algorithm, 306

- mechanical, 423
- RASP system, 304–5
- T-VPack algorithm, 305
- VPack algorithm, 305
- CMOS scaling, 507
- CMX-2X, 60
- Coarse-grained architectures, 32–33
  - PipeRench, 32–34
- Codesign ladder, 541
- Coding phase, 582, 585–86
  - block diagram, 586
  - See also* SPIHT
- Col combinator, 354–55
- Columns, skipping, 602
- Common path, 161–62
- Common subexpression elimination (CSE), 171
- Communicating Sequential Processes (CSP), 93, 106
- Communication, 243–48
  - I/O, 247
  - intertask, 251
  - latency, 247
  - method calls, 244
  - point-to-point, 251
  - shared memory, 243–44
  - streams, 244–46
  - styles, 243–46
  - virtual memory, 246–47
- Compaction, 324, 337–44
  - HWOP selection, 338
  - optimization techniques, 338–42
  - phases, 337–38
  - regularity analysis, 338
- Compilation, 212–13
  - accelerating classical techniques, 414–22
  - architecture effect, 427–31
  - automatic, 162–75, 212–13
  - C, uses and variations, 175–80
  - fast, 411–32
  - incremental place and route, 425–27
  - multiphase solutions, 422–25
  - partitioning-based, 423
  - PathFinder acceleration, 418–22
  - runtime netlist, 411, 413–14, 432
  - simulated annealing acceleration, 415–18
  - slow, 411
  - for spatial computing, 155–80
- Compilation flow, 150–52
- Complete evolution, 736–38
  - centralized, 736–37
  - population-oriented, 737–38
  - See also* Evolvable hardware (EHW)
- Complete matching, 841
- Complex programmable logic devices (CPLDs), 292
- Component reuse, 198–200
  - signal-processing primitives, 198
  - tiled subsystems, 198–200
  - See also* Streaming FPGA applications
- Computations
  - data-centric, 110
  - data-dependent, 104
  - on dataflow graph, 99
  - density of, 826
  - deterministic, 95
  - feedforward, 389
  - fixed-point, 475–99
  - memory-centric, 779–802
  - models, 96
  - nondeterministic, 96
  - phased, 104
  - SCORE, 205
  - spatial, 157
  - stream, 203–17
- Compute bound algorithms, 443
- Compute models, 92–107
  - applications and, 94
  - challenges, 93–97
  - correctness reasoning, 95
  - data parallel, 105
  - data-centric, 105–6
  - dataflow, 98–103
  - in decomposing problems, 94–95
  - diversity, 92
  - functions, 97
  - multi-threaded, 93, 106
  - object-oriented, 98
  - objects, 97–98
  - parallelism existence, 95
  - SCORE, 74, 203–17
  - sequential control, 103–5
  - taxonomy, 93
  - transformation permissibility, 95
  - Turing–Complete, 97
- Compute units, 319
- Computing primitives, 95
- Concurrent statements, 144, 150
- Concurrent-error detection (CED), 846
- Configurable Array Logic (CAL), 53
- Configurable bitstreams, 16, 402–6
  - closed architecture, 402
  - configuration, generation, 401–9

- Configurable bitstreams (*cont.*)
  - control bits, 405
  - data generation software, 407–8
  - downloading mechanisms, 406–7
  - generation, 401–9
  - open, 408
  - sizes, 405, 406
  - tool flow, 408
  - underlying data structure, 402
- Configurable logic blocks (CLBs), 23, 325
  - complexity, 507
  - flip-flops, 508
  - multiple, 509
  - resource reduction, 508
  - XC6200, 741
- Configuration transfer time reduction, 80–82
  - architectural approaches, 81
  - compression, 81–82
  - data reuse, 82
- Configuration upsets, 849–50
- Configuration(s)
  - architectures, 66–76
  - block reconfigurable, 74–75
  - cache, 83
  - caching, 77
  - compression, 81–82
  - controller, 66, 73
  - cycles, number of, 68
  - data reuse, 82
  - data transfer, 67
  - grouping, 76
  - multi-context, 68–70
  - partially reconfigurable, 70–71
  - pipeline reconfigurable, 73–74
  - relocation and defragmentation, 71–73
  - scheduling, 77–79
  - security, 82–83
  - single-context, 67–68
  - swapping, 72
- Configure-once implementation, 445
- Configured switches, 216
- Conjunctive normal form (CNF), 291, 613
- Connection blocks, 8
  - detail, 10
  - island-style architecture with, 9
- Connection Machine, 221, 223
- Constant coefficient multipliers, 459, 495
- Constant folding, 169, 450–51
  - automated, 473
  - constant propagation, 463
  - implementations with/without, 451
  - in instance-specific designs, 456–57
  - in logical expressions, 464–66
- Constrained 2D placement, 335–6
- Content-addressable memories (CAMs), 444
- Context switching, 80
- Context-sensitive optimization, 340–42
  - superslices, 340, 341
  - See also* Compaction
- Control dataflow graphs (CDFGs), 319
  - conversion to forest of trees, 330
  - primitive operators, 332
  - sequence, 334–35
- Control flow, 159–60
  - implementation, 159
  - subcircuits, 160
  - See also* C for spatial computing
- Control flow graph (CFG), 163, 164
- Control nets, 322
- Controller design, 124, 194–98
  - with Matlab M language, 195–97
  - with Simulink blocks, 194–95
  - with Verilog, 197
  - with VHDL, 197
- Controllers
  - configuration, 66, 73
  - delay line, 195–96
  - FSM, 124
  - RaPiD, 39
  - sequential, 120
  - vector architecture, 120
- Coordinate systems, CORDIC, 520–21
- Coprocessors
  - independent, 36–40
  - scalar processor with, 117
  - streaming, 109–10
  - vector, 121–22
- CORDIC, 437, 513–35
  - adaptive lattice structures and, 514
  - adaptive nulling and, 514
  - alternatives, 513, 520
  - angle approximation error, 522–23
  - architectural design, 526–27
  - computation noise, 522
  - computational accuracy, 521–26
  - convergence, 527–28
  - coordinate systems, 520–21
  - datapath rounding error, 523–26
  - engine, 527, 534
  - in FFT, 514
  - folded architecture, 528–30
  - functions computed by, 521
  - implementation, 526–27

- input mapping, 527
- input sample, 525
- iterations, 516, 527
- Kalman filters and, 514
- micro-rotations, 526
- parallel linear array, 530–33
- PE, 532
- processing, 522
- quantization effects, 524
- realizations, 513–14
- result vector error, 523
- rotation mode, 514–17
- scaling, 517–19
- scaling compensation, 534
- shift sequences, 522
- as shift-and-add algorithm, 513
- unified description, 520–21
- variable format, 524
- vector rotation, 518
- vectoring mode, 514, 519–20, 525
- in VLSI signal processing, 514
- y-reduction mode, 519
- z-reduction mode, 517
- CORDIC processors
  - datapath format, 526
  - datapath width, 523
  - effective number of result bites, 525
  - FPGA implementation, 527–34
  - FPGA realizations, 523
  - full-range, 527, 528
  - with multiplier-based scaling compensation, 535
  - PE, 533
- COordinate Rotation DIgital Computer.
  - See* CORDIC
- Cosine, 437
- Cost function, 440
  - PathFinder, 368, 375
  - power-aware, 284
  - in simulated annealing, 306
- Coverification, 639–40, 650–51
  - flow between workstation and emulator, 665
  - performance, 650
  - simulation, 651
  - use of, 640
  - VLE interfaces for, 664–65
  - See also* Logic emulation systems
- CPU blocks, 15
- Cray supercomputers, 60
- Crosspoints, 857–58
  - diode, 859
  - nanowire-nanowire, 866
- Custom evolvable FPGAs, 743–45
  - axes, 744
  - POetic tissue, 743–45
  - See also* Evolvable hardware (EHW)
- Customizable instruction processors, 121, 461–62
- Cut enumeration-based algorithm, 287
- Cut generation, 279–80
- Cvt class, 266–68
  - GUI, 267
  - implementation, 266–67
- D-flip-flops, 596
- DA. *See* Distributed arithmetic
- DAOmap, 282–83
  - area improvement, 283
  - multiple cut selection passes, 283
- DAP, 221
- Data Encryption Standard (DES), 459
- Data nets, 321
- Data parallel, 119–22
  - application programming, 219–30
  - compute model, 105
  - languages, 222–23
  - SIMD, 120
  - SPMD, 120
  - system architecture, 119–22
- Data presence, 108–9, 110
- Data queuing, 756
- Data range propagation, 482–84
- Data-centric, 105–6, 110
- Data-dependent branching, 221
- Data-element size, 442–43
- Data-oriented specialization, 450–52
- Dataflow, 98–103
  - analysis-based operator size reduction, 172
  - direction, 321
  - dynamic streaming, 100–2
  - dynamic streaming, with peeks, 102
  - single-rate synchronous, 99
  - streaming, with allocation, 102–3
  - synchronous, 99–100
  - techniques, 93
- Dataflow graphs (DFGs), 78, 319
  - building, 164
  - circuit generation, 164
  - computation on, 99
  - control (CDFG), 319
  - DSP, 93
  - edges, 165
  - edges, building and ordering, 166–68

- Dataflow graphs (DFGs) (*cont.*)
  - implicit type conversions, 172
  - live variables at exits, 168–69
  - multirate, 100
  - muxes, building, 167
  - nodes, 165
  - operations in clock cycles, 173–74
  - optimization, 164
  - predicates, 167
  - scalar variables in memory, 169
  - single-rate static, 100
  - as “stepping stone,” 164–65
  - top-level build algorithms, 165–66
  - See also* DFG optimization
- Dataflow Intermediate Language (DIL), 34
- Dataflow single-rate synchronous, 99
- Datapath composition, 319–44
  - device architecture impact, 324–26
  - interconnect effect, 326
  - interface to module generators, 326–29
  - layout, 322–23
  - mapping, 329–33
  - regularity, 320–22
  - tool flow overview, 323–24
- Datapath pruning, 524
- Datapath rounding error, 523–26
- Datapaths
  - butterfly, 688
  - with explicit control, 195
  - FSM, 138–49
  - FSM communication with, 123–24
  - high-performance, 184
  - HWOPs, 320, 321–22
  - layout, 322–23
  - sharing, 109
  - SIMD, 815, 818
  - word-wide, 216
- dbC language, 224
- Deadlock, 96
- Deadlock prevention, 249
- Debug circuitry synthesis, 271–72
- Debugging
  - ASICs, 441
  - FPGAs, 440–41
  - JHDL, 270–72
- Decoders, 376–77, 862–63
- Dedicated-wire systems, 641
  - channel graph, 647
  - recursive bipartitioning, 646
  - routing problem, 646
  - See also* Multiplexed-wire systems
- Deep pipelining, 706
- Defect maps
  - with component-specific mapping, 836
  - model, 832
- Defect tolerance, 830–43
  - Active Pages and, 779, 799–801
  - associativity and, 800
  - concept, 830–32
  - defect map model, 832
  - global sparing, 836–37
  - local sparing, 838–39
  - with matching, 840–43
  - models, 831–32
  - nanoPLA, 869
  - perfect component model, 831, 837–38
  - with sparing, 835–39
  - substitutable resources, 832
  - testing, 835–36
  - yield, 832–35
- Defects
  - faults and, 830
  - lifetime, 848–49
  - rates, 829, 832
- Defragmentation, 71–73
  - device support, 77
  - software-based, 79–80
- Delay lines
  - controller, 195–96
  - synchronous, 194
  - VPR computation, 309–10
- Delay Optimal Mapping algorithm. *See* DAOmap
- Delay(s)
  - configurable, 187
  - as cost approximation, 375
  - delta, 150
- Delta delay, 150
- De-multiplexers, 376–77, 862–63
- Denial of service (DoS), 774
- Denormals, 673
- Depth-first search order, 585
- Derivative monitors, 490
- Deterministic Finite Automata, 103
- Device architecture, 3–27
- DFG. *See* Dataflow graphs
- DFG optimization, 169–73
  - Boolean value identification, 171
  - constant folding, 169
  - CSE, 171
  - dataflow analysis-based operator size reduction, 172
  - dead node elimination, 170–71
  - identity simplification, 170



- memory access optimization, 172
- redundant loads removal, 172–73
- strength reduction, 170
- type-based operator size reduction, 171–72
- See also* Dataflow graphs (DFGs)
- Digital signal processors (DSPs), 49, 93
- Direct memory access (DMA), 246
- Discrete cosine transform (DCT), 389, 479, 511
- Discrete Fourier transform (DFT)
  - output vector, 534
  - symmetries, 687
- Discrete wavelet transform (DWT), 567
  - architecture illustration, 575
  - architectures, 571–75
  - computational complexity, 572
  - engine runtime, 574
  - folded architecture, 571, 572
  - generic 2D biorthogonal, 573–74
  - partitioned, 572, 573
  - phase, 582
  - two-dimensional, 571
- Distributed arithmetic (DA), 503–11
  - algorithm, 504
  - application on FPGA, 511
  - FIR filters, 575
  - implementation, 504–7
  - LUT size and, 505
  - performance, improving, 508–11
  - reduced memory implementation, 507
  - theory, 503–4
  - two-bit-at-a-time reduced memory implementation, 509
- Division operation, 437
- Dijkstra's algorithm, 371
- Dot product, 506, 683–86
  - FPGA implementation, 685
  - maximum sustainable floating-point rate, 685
  - multiply-accumulate, 686
  - multiply-add, 686
  - performance, 685–87
- Downloading mechanisms, 406–7
- DRAMs
  - computational hardware, 786
  - dies, 831
  - hardware design, 780
  - high-density, 780
- Dtb class, 270
- Dynamic FPGAs, 600
- Dynamic Instruction Set Computer (DISC), 447
- Dynamic partial reconfiguration, 742–43
- Dynamic reconfiguration, 552
- Dynamic RPF, 29
- Dynamic scheduling, 240–41
  - frontier, 240–41
  - runtime information, 240, 241
  - See also* Scheduling
- Dynamic streaming dataflow, 101–2
  - with peeks, 102
  - primitives, 101
- Dynamic testbench, 269–70
- Dynamically linked libraries (DLLs), 235, 773
- Dynamically reconfigurable ATR system, 604–6
- Dynamically reconfigurable designs, 594–600
  - algorithm modifications, 594
  - FPGAs over ASICs, 595–96
  - image correlation circuit, 594–96
  - implementation method, 599–600
  - performance analysis, 596–97
  - template partitioning, 598–99
  - See also* ATR
- Edge mask display, 190
- Edges
  - building, 166–67
  - circuit graph model, 377
  - detection design driver, 185
  - liveness, 165
  - ordering, 167–68, 172, 173
- EDIF (Electronic Design Interchange Format), 407
- Effective area, 280
- Electric and magnetic field-updating algorithms, 700–1
- Embedded memory blocks (EMBs), mapping logic to, 291–92
- Embedded microprocessors, 197–98
- Embedded multipliers, 514
- Embedded multiply-accumulator (MACC) tiles, 514, 680
- EMB\_Pack algorithm, 292
- Epigenesis, 726
- Epigenetic axis, 727, 744
- Error checking, 233
- Error estimation, 485–96
  - fixed-point error, 486
  - high-level area models, 493–96

- Error estimation (*cont.*)
  - LTI systems, 487–89
  - noise model, 487–88
  - noise propagation, 488–89
  - nonlinear differentiable systems, 489–93
  - quantization, 711–12
  - simulation, 486
  - simulation-based methods, 487
- Evolution
  - artificial, 727–29
  - centralized, 736–37
  - complete, 736–38
  - extrinsic, 733
  - intrinsic, 734–35
  - open-ended, 738–39
  - population-oriented, 737–38
- Evolutionary algorithms (EAs), cellular programming, 738
- Evolutionary circuit design, 731, 733
- Evolutionary computation, 727
- Evolvable hardware (EHW), 729–46
  - as artificial evolution subdomain, 731
  - commercial FPGAs, 741–43
  - complete evolution, 736–38
  - custom, 743–45
  - digital platforms, 739–45
  - dynamic partial reconfiguration, 742–43
  - evolvable components, 739–40
  - extrinsic evolution, 733
  - future directions, 746
  - genome encoding, 731–32
  - intrinsic evolution, 734–35
  - JBits for, 743
  - living beings analogy, 729–30
  - off-chip, 732
  - on-chip, 732
  - open-ended evolution, 738–39
  - taxonomy, 733–39
  - virtual reconfiguration, 741–42
  - Xilinx XC6200 family, 740–41
- Exit nodes, 165
- Explicit layout
  - Cartesian, 351–52
  - no, 350
  - totally, 350
  - in VHDL, 351
- Explicit synchronization, 248
- Exploration
  - 0-1 knapsack problem, 553
  - complex formulations, 555–56
  - formulation with asymmetric communication, 553–55
  - parallel program partitioning, 558
  - sequential program partitioning, 552–57
  - simple formulation, 552–53
  - See also* Hardware/software partitioning
- Extended logic, 12–16
  - elements, 12–15
  - fast carry chain, 13–14
  - multipliers, 14–15
  - processor blocks, 15
  - RAM, 15
- Extreme subwavelength top-down lithography, 853
- Extrinsic EHW, 733
- F<sup>2</sup>PGA, 735
- Factoring, 515–16
- False alarm rate (FAR), 592
- Fast carry chain, 13–14
- Fast Fourier transform (FFT), 21, 389, 479
  - butterflies, 688
  - CORDIC algorithm and, 514
  - data dependencies, 692
  - FPGA implementation, 689–91
  - implementation factors, 692
  - parallel architecture, 689, 690
  - parallel-pipelined architecture, 690, 691
  - performance, 691–93
  - pipelined architecture, 689–91
  - radix-2, 687–88
- FDTD, 697–723
  - ABCs, 702
  - accelerating, 702
  - advantages on FPGA, 705–7
  - algorithm, 701–3
  - applications, 703–5
  - background, 697–701
  - breast cancer detection application, 703–4
  - as CA, 702–3, 723
  - as data and computationally intense, 702
  - deep pipelining, 706
  - field-updating algorithms, 700–1
  - fixed-point arithmetic, 706–7
  - flow diagram, 701
  - ground-penetrating radar application, 703
  - landmine detection application, 704
  - method, 697–707
  - model space, 698, 702, 712
  - parallelism, 705–6
  - PMLs, 702
  - reconfigurable hardware implementation, 704
  - spiral antenna model, 704, 705

- UPML, 702, 706
  - See also* Maxwell's equations
- FDTD hardware design case study, 707–23
  - 4 x 3 row caching model, 719
  - 4-slice caching design, 718
  - background, 707
  - data analysis, 709–12
  - dataflow and processing core
    - optimization, 716–18
  - expansion to three dimensions, 718–19
  - fixed-point quantization, 709–12
  - floating-point results comparison, 710, 711
  - hardware implementation, 712–22
  - managed-cache module, 717
  - memory hierarchy and interface, 712–15
  - memory transfer bottleneck, 715–16
  - model specifications, 711
  - parallelism, 720–21
  - performance results, 722
  - pipelining, 719–20
  - quantization errors, 710
  - relative error, 710, 711
  - relative error for different widths, 712
  - requirements, 707–8
  - results, 722
  - two hardware implementations, 721–22
  - WildStar-II Pro PFGA board, 708–9
- Feedforward correction, 844–45
  - memory, 845
  - TMR, 844
- FF. *See* Flip-flops
- Field effect transistors (FETs), 861
- Field Programmable Port Extender (FPX)
  - platform, 755, 756
  - applications developed for, 756
  - multiple copies, 770
  - physical implementation block diagram, 757
  - RAD circuits on, 770, 772
  - remote configuration on, 773
  - in WUGS, 756–57
- Field-programmable gate arrays. *See* FPGAs
- Field-programmable interconnect chips (FPICs), 643
- Field-programmable transistor arrays (FPTAs), 745
- Field-updating algorithms, 700–1
- FIFO, 37, 585
  - blocks, 586
  - buffers, 759
  - queues between operators, 108
  - streams, 847
  - token buffers, 102
- Fine-grained architectures, 30–32
- Finite-difference time-domain. *See* FDTD
- Finite-impulse response (FIR) filters, 21, 98, 389, 479
  - 4-tap, 510
  - 16-tap, 507, 508
  - distributed arithmetic, 575
  - general multipliers, 460
  - instance-specific multipliers, 460
  - mapping onto FPGA fabric, 507
  - SPIHT implementation and, 576
  - taps, 503
- Finite-precision arithmetic, 519
- Finite-State Machine with Datapath (FSMD), 112, 124
- Finite-state machines (FSMs), 112, 620, 621
  - coarse-grained, 125
  - communicating with datapaths, 123–24
  - controller, 124
  - datapath example, 138–49
  - states, 621–22
  - VHDL programming, 130
- Firewalls, 754
- First-in, first-out. *See* FIFO
- Fixed instructions, 815
- Fixed Order SPIHT, 578–80
  - basis, 579
  - order, 579
  - PSNR curve, 579
  - SPIHT comparison, 581
  - See also* SPIHT
- Fixed-frequency FPGAs, 394–95
- Fixed-Plus-Variable (F + V) computer, 48
- Fixed-point computation, 475–99, 706–7
  - analytic peak estimation, 479–84
  - FDTD algorithm, 708
  - peak value estimation, 478–85
  - precision analysis for, 475–99
  - relative error, 712
  - simulation-based peak estimation, 484
- Fixed-point error, 486
- Fixed-point number system, 448–49, 475–78
  - 2's complement, 709
  - data structure, 710
  - in embedded applications, 476
  - flexibility, 476
  - multiple wordlength paradigm, 476–77
  - reconfigurable logic, 476
- Fixed-point precision analysis, 575–78
  - final variable representation, 578

- Fixed-point precision analysis (*cont.*)
  - magnitude calculations, 576
  - variable representation, 577
  - See also* SPIHT
- FLAME, 327–28
  - design data model, 327–28
  - library specification, 328
  - Manager, 327
  - topology description, 328
- Flash memory, 17
- Flexible API for Module-based Environments. *See* FLAME
- Flexible binding, 236–38
  - fast CAD for, 238
  - install time binding, 236–37
  - preemption and, 242
  - runtime binding, 237–38
  - See also* Operating systems (OSs)
- FlexRAM, 801
- Flip-flops (FFs), 286, 597
  - CLB, 508
  - D, 5–6, 596
  - retiming and, 286
- Floating point, 449–50, 671–79, 706
  - adder block, 676
  - adder implementation, 675–77
  - adder layout, 676
  - application case studies, 679–92
  - denormals, 673
  - difficulty, 671–78
  - dot product, 683–86
  - FFT, 686–92
  - IEEE double-precision format, 672
  - implementation, 692
  - implementation considerations, 673–75
  - matrix multiply, 679–83
  - maximum sustainable rate, 685
  - multiplier block, 678
  - multiplier implementation and layout, 678
  - numbers, 672
  - summary, 692–94
- Floating region, 303
- Flow graphs, 78, 79
- FlowMap algorithm, 279, 282
- Focus of Attention (FOA) algorithm, 592
- Folded CA, 123
- Folded CORDIC architecture, 528–30
- Follow-on SAT solver, 627–33
  - characteristics, 628
  - clause modules, 629–30
  - compilation time reduction, 627–33
  - conflict analysis, 630
  - creation methodology, 632
  - global topology, 628
  - implementation issues, 631–32
  - main control unit, 630
  - optimized pipelined bus system, 628, 629
  - performance, 630–33
  - shared-wire global signaling, 628
  - structural regularity, 628
  - system architecture, 627–30
  - See also* Boolean satisfiability; SAT solvers
- Forward error correction (FEC), 755
- Forward propagation, 482–84
- FPGA fabrics, 14–15, 40–41
  - arbitrary-precision high-speed adder/subtractors support, 530
  - architectures, 30–34
  - dedicated paths, 511
  - footprint, 527
- FPGA placement, 297–98
  - alternative, 297–98
  - analytic, 315
  - challenge, 316
  - clustering, 304–6
  - designer directives, 302–4
  - device legality constraints, 300–1
  - difficulty, 275
  - general-purpose FPGAs, 299–316
  - homogeneous, 503
  - importance, 299
  - independence tool, 312
  - inputs, 299
  - legal, 300
  - optimization goals, 301–2
  - partition-based, 312–15
  - problem, 299–304
  - PROXI algorithm, 311–12
  - routability-driven algorithms, 301
  - routing architecture influence, 302
  - simulated annealing, 306–12
  - simultaneous routing, 311–13
  - timing-driven algorithms, 301
  - tools, 301
  - See also* FPGAs
- FPGAs, 1, 47
  - antifuse, 17–18
  - application implementation with, 439–52
  - arithmetic implementation, 448–52
  - ATR systems with, 591–610
  - backend phase, 151

- as blank hardware, 16
- case studies, 18–23
- circuit layout specification, 347–63
- clock rates, 441
- compilation flow, 151
- computing, CORDIC architectures for, 513–35
- configuration, 16–18
- configuration data transfer to, 67
- configuration memory systems, 2
- CORDIC processor implementation, 527–34
- cost, 440
- DA application on, 511
- debug and verification, 440–41
- dedicated processors, 15
- development, 440
- dynamic, 600
- efficiency of processors and, 825
- emulation system, 55
- evolvable, 725–46
- fabric, 15
- fixed-frequency, 394–95
- flash memory, 17
- flexibility, 87
- floating point for, 671–94
- general-purpose hardware implementation, 458
- island-style, 6, 7, 314
- K-gate, 600
- LUTs, 4–6, 279
- low-quality ASICs use, 1
- multi-context, 68–70
- network data processing with, 755–56
- number formats, 436
- partially reconfigurable, 70–71
- performance, 438
- power consumption, 440
- in reconfigurable computing role, 3
- routing resources, 348, 367
- scaling, 411, 412, 431–32
- SIMD computing on, 219–21
- single-context, 67–68
- SRAM, 16–17
- static, 600
- streaming application programming, 183–202
- strengths/weaknesses, 439–41
- testing after manufacture, 407
- time to market, 439–40
- volatile static-RAM (SRAM), 6
- See also* FPGA placement
- FPgrep, 761
- FPsed, 761
- FPX. *See* Field Programmable Extender platform
- Fractional fixed-point data, 523
- Fractional guard bits, 522
- FSM. *See* Finite-state machines
- FSM datapath, 138–49
  - adder representation, 144
  - concurrent statements, 144
  - control signal generation, 145–48
  - control signal generation illustration, 146
  - design illustration, 139
  - multiplexer representation, 144
  - multiplier representation, 144
  - next-state decoder, 149
  - registers, 144–45
  - sequential statement execution, 149
  - structural representation, 138–41
  - time-shared datapath, 141–44
- FSMD. *See* Finite-state machine with datapath
- Full-range CORDIC processors, 527, 528
  - input quadrant mapping, 528
  - micro-rotation engine, 529*See also* CORDIC
- Function blocks. *See* Logic blocks
- Functional blocks (FBs), 741
- Functional mapping algorithms, 277
- Functional Unit model, 41–43, 115–16
- Functions, 97
- GAMA, 331, 333
- Garp's nonsymmetrical RPF, 30–32, 40
  - configuration bits, 31
  - configurator, 32
  - number of rows, 30
  - partial array configuration support, 31*See also* Fine-grained architectures; RPF
- General computational array model, 807–14
  - implications, 809–14
  - instruction distribution, 810–13
  - instruction storage, 813–14
- General-purpose FPGA placement. *See* FPGA placement
- General-purpose programming languages (GPLs), 255, 256
- Generic 2D biorthogonal DWT, 573–74
- Genetic algorithms (GAs), 727–29
  - components, 729
  - crossover, 729
  - decoding, 728

- Genetic algorithms (*cont.*)
  - fitness evaluation, 728
  - genetic operators, 728–29
  - initialization, 728
  - mutation, 729
  - steps, 728–29
  - variable-length (VGA), 735
- Genome encoding, 731–32
  - fitness calculation, 732
  - high-level languages, 731
  - low-level languages, 732
- Genomes, 728
- Given's rotations, 514
- Global RTR, 446, 447
- Global sparing, 836–37
- Globally Asynchronous, Locally Synchronous (GALS) model, 109
- Glue-logic, 441
- Granularity, 30–34
  - coarse, 32–34, 546
  - dynamically determined, 547
  - fine-grained, 30–32, 546
  - heterogeneous, 546
  - manual partitioning, 546
  - parallel program partitioning, 557
  - region, 545
  - sequential program partitioning, 545–47
  - See also* Hardware/software partitioning
- Graph bipartitioning, 553
- GRASP, 618, 625, 632–33
- Greedy heuristics, 553–55
- Ground-penetrating radar (GPR), 703, 704, 711
- Group migration, 554
- Hard macros, 336, 424
- Hardware-Accelerated Identification of Languages (HAIL), 768
- Hardware-assisted simulated annealing, 418
- Hardware description languages (HDLs), 183, 235, 407, 541
- Hardware execution checkpoints, 272
- Hardware operators (HWOPs)
  - boundary dissolution, 337
  - compaction, 324
  - linear stripes, 335
  - mapping, 323
  - module generation, 323
  - multibit wide, 320
  - neighboring, 338
  - non-bit-sliced, 324
  - pitch, 321
  - pitch-matched, 322
  - placement, 324
  - regular structure, 320–21
  - selection for compaction, 338
  - swaps, 334
  - See also* Datapaths; HWOP placement
- Hardware protection, 250–51
- Hardware prototyping, 411, 412–13, 432
  - reasons for employing, 412
  - Taramac system and, 427
- Hardware/software partitioning, 539–59
  - alternative region implementation, 544, 549–50
  - exploration, 544, 552–57
  - FPGA technology and, 539
  - granularity, 544, 545–47
  - implementation models, 544, 550–52
  - of parallel programs, 557–58
  - partition evaluation, 544, 547–48
  - problem, 539–40
  - of sequential programs, 542–57
  - speedup following Amdahl's Law, 543
- Hash tables, 762
- HDL Coder, 183
- Heuristic search procedure, 496–97
- Heuristics, 553, 555
  - greedy, 553–55
  - neighborhood search, 556
  - nongreedy, 553–55
  - simulated annealing, 555–56
- Hierarchical annealing algorithm, 310–11
- Hierarchical composition, 125
- Hierarchical FPGAs, 313
- Hierarchical routing, 10–12
  - FPGA placements, 301
  - long wires, 11
- High-fanout nets, 419, 425
- High-level languages (HLLs), 44–45, 52, 401
  - enabling use of, 44–45
  - genome encoding, 731
- Huffman decoding, 233
- HWOP placement, 333–37
  - constrained two-dimensional, 335–36
  - linear, 333–35
  - simultaneous tree covering and, 334
  - styles, 333
  - two-dimensional, 336–37
- HWSystem class, 272
- Hyperblocks
  - basic block selection for, 166
  - building DFGs for, 164–69
  - formation, 168

- I/O, 247
  - bound algorithms, 443
  - performance, 443–44
- IDCT, 233
- IEEE double-precision floating-point
  - format, 672
- If-then-else, 158–59
- IKOS Logic Emulator, 630–31
- IKOS VirtualLogic SLI Emulator, 623
- Illinois Pular-based Optical Interconnect (iPOINT), 755
- Ilv combinator, 359
- Image correlation circuit, 594–96
- Image-processing design driver, 185–94
  - 2D video filtering, 187–91
    - horizontal gradient, 188, 189
  - mapping video filter to BEE2 FPGA platform, 191–94
  - RGB video conversion, 185–87
    - vertical gradient, 188, 189
  - See also* Streaming FPGA applications
- IMap algorithm, 281–82
- Implementation models
  - dynamic reconfiguration parameter, 552
  - parallel program partitioning, 557–58
  - parameters, 551–52
  - real-time scheduling, 558
  - sequential program partitioning, 550–52
  - See also* Hardware/software partitioning
- Implicit synchronization, 248–49
- Imprint lithography, 870–71
- Impulse project, 802
- In-circuit emulation, 639, 650
- Incremental mapping, 425–27
  - design clock cycle, 663
  - See also* Mapping
- Incremental partitioning, 661
- Incremental place and route, 425–77
- Incremental rerouting, 374–75
- Incremental routing, 661
- Independence tool, 312
- Induced architectural models, 814–16
  - fixed instructions, 815
  - shared instructions, 815–16
- Infinite-impulse response (IIR) filters, 21, 98, 479
- Install time binding, 236–37
- Instance-specific design, 411, 413, 432, 455–73
  - approaches, 457–58
  - architecture adaptation, 457
  - changing at runtime, 456
  - concept, 455
  - constant coefficient multipliers, 459
  - constant folding, 456–57
  - customizable instruction processors, 461–62
  - examples, 459–62
  - function adaptation, 457
  - implementation, 456
  - key-specific crypto-processors, 459–60
  - NIDS, 460–61
  - optimizations, 456–57
  - partial evaluation, 462–73
  - requirements, 456
  - taxonomy, 456–57
  - use examples, 457
- Instruction augmentation, 115–16
  - coprocessor model, 116
  - Functional Unit model, 115–16
  - instruction augmentation model, 116
  - manifestations, 115
- Instruction distribution, 810–13
  - assumptions, 811
  - wiring, 811
- Instruction Set Architecture (ISA)
  - processor models, 103
- Instruction-level parallelism, 796
- Instructions
  - array-wide, 814
  - base, 115
  - controller issuance, 113
  - fixed, 815
  - shared, 815–16
  - storage, 813–14
- Integer linear programming (ILP), 497, 553
- Integrated mapping algorithms, 284–89
  - integrated retiming, 286–87
  - MIS-pga, 288
  - placement-driven, 287–89
  - simultaneous logic synthesis, 284–86
  - See also* Technology mapping
- Integrated retiming, 286–87
- Interconnect
  - Altera Stratix MultiTrack, 21–22
  - connection block, 8–10
  - effect on datapath placement, 326
  - hierarchical, 10–12
  - nearest neighbor, 7–8
  - optimization, 110
  - programmability, 12
  - segmented, 8–10
  - sharing, 110
  - structures, 7–12
  - switch block, 8–10

- Internet key exchange (IKE), 775
- Internet Protocol Security (IPSec), 775
- Internet worms, 760
- Interslice nets, 322
- Intertask communication, 251
- Intraslice nets, 322
- Intrinsic evolution, 734–35
- Intrusion detection, 756, 762–67
- Intrusion detection and prevention system (IDPS), 763
- Intrusion detection system (IDS), 762
- Intrusion prevention, 756, 762–67
- Intrusion prevention system (IPS), 754, 763
- IP processing, 758
- iRAC clustering algorithm, 305–6
- Island-style FPGAs, 6, 7
  - with connect blocks, 9
  - partitioning, 314
- Isolation, 251
- Iterative mapping, 288
- Java, 541
- JBits, 408, 631
  - for evolving circuits, 743
  - JHDL with, 271
- JHDL, 88, 89, 255–72
  - advanced capabilities, 269–72
  - behavior synthesis, 270
  - CAD system, 255, 265–68
  - checkpointing, 272
  - circuit data structure, 257
  - as circuit design language, 264–65
  - debug circuitry synthesis, 271–72
  - debugging capabilities, 270–72
  - descriptions, 264
  - design process illustration, 257
  - dynamic testbenches, 269–70
  - as embedded design language, 256
  - hardware mode, 268–69
  - Logic Library, 270
  - module generators, 263
  - motivation, 255–57
  - open-source license, 272–73
  - placement attributes, 263
  - primitive instantiation, 257–59
  - primitives library, 257
  - programmatic circuit generation, 261–63
  - Sea Cucumber and, 270
  - simulation/hardware execution
    - environment, 268
  - as structural design language, 263–64
  - testbenches, 265–66
- JHDL classes
  - cvt, 266–68
  - dtb, 270
  - HWSysTem, 272
  - Logic, 259–61, 272
  - TechMapper, 260, 264, 272
- Johnson's algorithm, 809
- K*-input lookup tables (*K*-LUTs), 277
- K-Means clustering algorithm, 227, 228
- Kalman filters, 514
- Key-specific crypto-processors, 459–60
- Lagrangian multipliers and relaxation, 376
- Lambda Calculus model, 96
- Langmuir-Blodgett (LB) flow techniques, 857
- Language identification, 767–68
- Latency
  - BlockRAMs, 713
  - butterfly path, 694
  - C-slow retiming, 392
  - clock cycle, 507
  - communication, 247
- Lattice ECP2, 83
- Lava, 352
- LCS algorithm, 791–94
  - parallel execution, 791
  - simulation results, 793, 794
  - three-dimensional, 793–94
  - two-dimensional, 791–92
- Least significant bit (LSB), 321, 510
- Leiserson's algorithm, 384–86
- LEKO, 282
- LEON benchmark, 398
- Lifetime defects, 848–49
  - detection, 848–49
  - repair, 849
  - See also* Defects; Defect tolerance
- Linear placement, 333–35
- Linear time-invariant (LTI) systems, 479–82
  - analytic technique, 487–89
  - error sensitivity, 489
  - scaling with transfer functions, 481–82
  - transfer function calculation, 479–80
- Linear-feedback shift registers (LFSRs), 98
- Linearization, 490
- List of insignificant pixels (LIP), 569
- List of insignificant sets (LIS), 569, 570
- List of significant pixels (LSP), 569, 570, 586
- Lithographic scaling, 854–55



- Liveness edges, 165
- Local arrays, 177
- Local minima, 554
- Local RTR, 446–47, 448
- Local sparing, 838–39
- Location update chain, 417
- Logic, 3–6
  - duplication, 284
  - elements, 4–6
  - extended, 12–16
  - fast carry chain, 13–14
  - glue, 441
  - mapping to EMBs, 291–92
  - multivalued, 150
  - optimization, 342
  - programmability, 6
  - in RTL, 133
  - simultaneous synthesis, 284–86
  - unnecessary removal, 466
  - verification, 638
- Logic blocks, 5, 6, 13
- Logic class, 259–61
  - methods, 260–61
  - MUX example, 259–60
  - subroutines, 259
- Logic emulation systems, 411, 412–13, 432, 637–68
  - background, 637–39
  - case study, 653–65
  - complexity, 639
  - configuration illustration, 639
  - coverification, 639–40, 650–51
  - fast FPGA mapping, 652–53
  - FPGA-based, 637–39
  - FPGA-based, advantages, 667
  - future trends, 666–67
  - in-circuit emulation, 639, 650
  - issues, 650–51
  - logic analysis, 651
  - multi-FPGA, 641–44
  - processor-based, 666
  - single-FPGA, 640–41
  - types, 640–50
  - use of, 639–40, 651
  - VirtuaLogic VLE, 639, 653–65
- Logic fabric, 3–34, 14–15, 514
- Logic gates, 278
- Logic networks, 278
- Logic processors, 666–67
- LogicGen, 332–33
- Lookup table (LUT), 4–6, 264, 409, 503
  - 4-input, 507
  - DA implementation and, 505
  - defective, 838
  - exponential growth, 504
  - functionality, 403
  - inputs, 404
  - K*-input, 277
  - logic block illustration, 6
  - as logic “islands,” 404
  - mapping to, 289–90
  - as memory element, 403
  - memory size, 509
  - number per logic block, 5
  - outputs, 404
  - physical, 151
  - size, 5
  - synchronous, 510
- Loops
  - fission, 177
  - fusion, 177
  - interchange, 177
  - memory dependencies, 178–79
  - nest, 177
  - reversal, 177
- Loosely coupled RPF and processor
  - architecture, 41
- Lossless synthesis, 285
- Low-level languages, genome encoding, 732
- Low-temperature anneal, 311
- Low-voltage differential signaling (LVDS), 667
- LTI. *See* Linear time-invariant systems
- M*-tap filter, 509–10
- Macrocells, mapping to, 292
- Macros
  - hard, 336, 424
  - identification, 424
  - parameterizable, 493
  - soft, 336, 424
- Malware, 762
  - appearance, 764
  - propagation, 764
- Manual partitioning, 540, 546
- Mapping, 329–33
  - 1:1, 329–30
  - combined approach, 332–33
  - component-specific, 837
  - DA onto FPGAs, 507–8
  - dedicated-wire, 641
  - design, with multiple asynchronous clocks, 657–61
  - incremental, 425–27, 662–63

- Mapping (*cont.*)
  - LUT, 471–72
  - multi-FPGA emulator flow, 645
  - multiplexed-wire, 642
  - multiported memory, 657
  - N:1, 330–32
  - stages, 414
- Mapping algorithms, 277–93
  - area-oriented, 280–82
  - complex logic blocks, 290–91
  - DAOmap, 282–83
  - delay optimal, 283
  - FlowMap, 279, 282
  - functional, 277
  - for heterogeneous resources, 289–92
  - IMap, 281–82
  - integrated, 278, 284–89
  - iterative, 288
  - LEKO, 282
  - logic to EMBs, 291–92
  - LUTs of different input sizes, 289–90
  - macrocells, 292
  - matching formulation, 841
  - MIS-pga, 288
  - optimal-depth, 287
  - performance-driven, 282–83
  - placement-driven, 287–89
  - PLAmap, 292
  - power-aware, 283–84
  - PRAETOR, 280–81
  - structural, 277, 278–84
  - times, 837
- Markov Models, 78, 768
- Mask parameters, 184, 187
- MasPar, 221
- Master slices, 320, 321
- Matching
  - complete, 841
  - defect tolerance with, 840–43
  - fine-grained Pterm, 841–42
  - formulation, 841
  - maximal, 841
- MATLAB, 88, 195–97, 198
- Matrix multiply, 679–83
  - decomposition, 680
  - FPGA implementation, 680–81
  - implementation, 681
  - MACC operations, 680
  - maximum achievable performance versus
    - memory bandwidth, 683
    - memory accesses, 682
    - performance, 679, 682–83
    - performance of FPGAs and microprocessors, 684
- Maximal matching, 841
- Maximum magnitude phase, 582, 583–85
  - block diagram, 585
  - calculation, 583
  - See also* SPIHT
- Maxwell's equations, 697
  - curl, 698
  - discovery, 697
  - in rectangular coordinates, 699
  - as set of linear equations, 700
  - solving, 697
- Memory
  - access operations, 158
  - access optimization, 172
  - C for spatial computing, 157–58
  - CAM, 444
  - FDTD hardware implementation, 712–15
  - FPGA elements, 444
  - instruction, 814
  - nodes, 175
  - PE, 221
  - ports, 175, 444
  - retiming, 387
  - scalar variables in, 169
  - SDRAM, 760
  - shared, 124–25, 243–44
  - single pool, 104–5
  - total amount of, 444
  - virtual, 246–47
- Memory management unit (MMU), 246, 247
- Memory-centric computation, 779–802
  - algorithmic complexity, 786–94
  - parallelism, 794–99
  - performance results, 781–86
  - See also* Active Pages
- Message authentication code (MAC), 775
- Message passing, 124, 244
- Method calls, 244
- Microplacement, 342, 343
- Microprocessors, 439, 441
- MIS-pga algorithm, 288
- Modular robotics, 739
- Module generator interface, 326–29
  - data model, 327–28
  - flow, 327
  - intra-module layout, 328–29
  - library specification, 328
- Module generators
  - FLAME-based libraries, 327

- flexibility, 326
- PARAMOG library, 338
- Mojave ATR system, 594, 604–6
  - machine comparison, 606
  - photograph, 605
  - results, 604
  - used resources, 605
- Moore's Law, 637, 753
  - circuit density growth, 49
  - process scaling, 826
- MORPH project, 801
- Morton Scan Ordering, 584
- Most significant bit (MSB), 321, 493, 494, 510
- Multi-context devices, 68–70
  - benefits, 69
  - configuration bits, 69
  - drawbacks, 69–70
  - physical capacity, 69
- Multidomain signal transport, 658, 659, 660
  - requirement, 660
  - retimed, 660
- Multi-FPGA emulation, 641–44
  - as complex verification platforms, 641
  - constraints, 644
  - crossbar topology, 643
  - dedicated-wire mapping, 641, 642
  - design mapping, 644–45
  - high-level flow, 644
  - inter- and intra-FPGA connections, 647
  - inter-partition logic communication, 641
  - interconnection, 647
  - mapping flow, 645
  - mesh topology, 643
  - multiplexed-wire mapping, 642
  - partitioning approach, 645–46
  - placement approach, 645–46
  - routing approaches, 646–50
  - topologies, 641, 643
  - See also* Logic emulation systems
- Multi-SIMD coarse-grained array, 228
- Multi-terminal nets, 420, 421, 425
- Multi-threaded, 106, 123–25
  - FSMs with datapaths, 123–24
  - message passing, 124
  - model, 93
  - processors with channels, 124
  - shared memory, 124–25
- Multiple wordlength
  - adder formats, 494
  - optimization for, 478
  - paradigm, 476–77
- Multiplexed-wire systems, 642
  - circuit mapping, 649
  - incremental compilation, 662
  - inter- and intra-FPGA connections, 647
  - partitioning for, 646
  - routing, 648
  - utilization of wires, 648
  - See also* Dedicated-wire systems
- Multiplexers, 401
  - 2-input, 130–32, 403
  - 4-input, 134–35, 136–38, 404
  - FSM datapath, 144
  - if-then-else, 158–59
  - inputs, 403
  - logical equations, 133–34
  - primitive instantiation example, 258
  - pseudo, 377
- Multiplexing
  - factors, 796
  - nonactive memory and, 798
  - performance, 796
  - processor width versus, 797–99
- Multiplication function, 405
- Multipliers, 14–15
  - area estimation, 495
  - constant coefficient, 459, 495
  - embedded, 514, 712
  - floating point, 677–78
  - general cell, 466
  - instance-specific, 460
  - Lagrangian, 376
  - partial evaluation of, 466–70
  - shift-add, 467
- Multiply-accumulate (MACC) operations, 680
- Multiported memory mapping, 657
- Multiprocessing environments, 799
- Multivalued logic, 150
- Multiway partitioning, 313
- Muxes, building, 167
- Myrinet ATR system, 606–7
  - host, 606
  - photograph, 607
  - simulations, 607
- $N:1$  mapping, 330–32
- NanoPLA, 841
  - architecture, 864–70
  - basic logic block, 864–67
  - block illustration, 865
  - blocks, 867
  - defect tolerance, 869

- NanoPLA (*cont.*)
  - density benefits, 870
  - design mapping, 869
  - interconnect architecture, 867–69
  - memories, 869
  - tiling with edge I/O, 868
  - wired-OR planes, 867
- Nanoscale architecture, 853–73
  - bottom-up technology, 855–58
  - challenges, 858–59
  - CMOS pitch matching via tilt, 872
  - design alternatives, 870–72
  - imprint lithography, 870–71
  - interfacing, 871–72
  - lithographic scaling, 854–55
  - nanoPLA, 864–70
  - nanowire circuits, 859–62
  - restoration, 872
  - statistical assembly, 862–64
- Nanovia, 871
- Nanowire circuits, 859–62
  - inverter, 862
  - restoration, 860–62
  - wired-OR diode logic array, 859–60
- Nanowires, 856–57
  - addressing, 866
  - angled, 871
  - assembly, 857
  - decoder for, 863
  - doping profiles, 857–58
  - field effect controlled, 861
  - Langmuir–Blodgett alignment, 857
  - statistical selection, 863
  - switchable modules between, 858
- NBitAdder design, 262
- NBTI, 848
- NCHARGE API, 772
- Nearest-neighbor connectivity, 7–8
- Negotiated Analytic Placement (NAP)
  - algorithm, 315
- Negotiated Congestion Avoidance
  - algorithm, 369
- Negotiated congestion router, 367–72
  - algorithm, 370–71
  - first-order congestion, 368
  - iterative, 369
  - priority queue, 371
  - second-order congestion, 370
- Negotiated congestion/delay router, 372–73
- NetFPGA, 776
- Network Intrusion Detection System (NIDS), 460–61
- Network processing
  - build motivation, 753–54
  - complete system, 770–75
  - control and configuration, 771–72
  - control channel security, 774–75
  - data, with FPGAs, 755–56
  - dynamic hardware plug-ins, 773
  - hardware/software packet, 754–55
  - intrusion detection/prevention, 762–67
  - IP wrappers, 758
  - layered protocol wrapper
    - implementation, 759
  - partial bitfile generation, 773–74
  - payload processing with regular
    - expression scanning, 761–62
  - payload scanning with Bloom filters, 762
  - payload-processing modules, 760–61
  - protocol, 757–62
  - rack-mount chassis form factor, 770–71
  - with reconfigurable hardware, 753–57
  - reconfiguration mechanisms, 772–73
  - semantic, 767–70
  - system modularity, 756–57
  - TCP wrappers, 758–60
- Next-state decoder, 149
- Nodes
  - dead, elimination, 170–71
  - exit, 165
  - memory, connecting, 175
  - Seed, 291
- Noise injection, 490–93
- Noise model, 487–88
- Noise propagation, 488–89
- Nonchronological backtracking, 618
- Nondeterministic finite automata (NFA), 761
- Nonlinear differentiable systems, 489–93
  - derivative monitors, 490
  - hybrid approach, 489–93
  - linearization, 490
  - noise injection, 490–93
  - perturbation analysis, 489
- Nonrecurring engineering (NRE), 855
- Not a number (NaN), 449
- Number formats, 436
- Object-oriented model, 98
- Objects, 97–98
- On-demand scheduling, 239
- One-time programmable (OTP), 17
- Ontogenetic axis, 727, 744
- Ontogeny, 726

- Open Systems Interconnection (OSI)
  - Reference Model, 757
- Open-ended evolution, 738–39
- Operating system (OS)
  - abstracted hardware resources, 234–36
  - communication, 243–48
  - demands, 232
  - dynamic scheduling, 240–41
  - flexible binding, 236–39
  - on-demand scheduling, 239
  - preemption, 242
  - protection, 231, 249–51
  - quasi-static scheduling, 241
  - real-time scheduling, 241–42
  - roles, 231
  - scheduling, 239–42
  - security, 231
  - static scheduling, 239–40
  - support, 231–52
- Operations
  - C for spatial computing, 157
  - DFG, 173–74
  - MACC, 680
  - memory access, 158
  - packing into clock cycles, 173–74
- Operator size reduction, 171–72
  - dataflow analysis-based, 172
  - type-based, 171–72
- Optimization(s)
  - common path, 161–62
  - compaction, 338–42
  - context-sensitive, 340–42
  - decidable, 97
  - DFG, 164, 169–73
  - FPGA placement, 301–2
  - instance-specific, 456–57
  - interconnect, 110
  - logic, 342
  - memory access, 172
  - for multiple wordlength, 478
  - SPIHT, 586
  - undecidable, 97
  - wordlength, 485–97
  - word-level, 339–40
- Ordering edges, 167–68
  - absence, 173
  - existence, 173
  - false, removing, 172
- Packet inspection applications, 761
- Packet switches, 216
- Parallel compilation, VLE system, 665
- Parallel linear array, 531
  - based on Virtex-4 DSP48 embedded tile, 533
  - CORDIC, 530–33
- Parallel PathFinder, 377–79
- Parallel program partitioning, 557–58
  - alternative region implementations, 557
  - evaluation, 557
  - exploration, 558
  - granularity, 557
  - implementation models, 557–58
- Parallel programs, 540
  - data dependence, 102
  - data parallel, 105
  - data-centric, 105–6
  - multi-threaded, 106
  - sequentialization and, 104–5
  - synchronization, 248–49
- Parallelism, 99, 105, 118, 248
  - artificial, 105
  - bulk synchronous, 118–19
  - in compute models, 95
  - data, 95, 234, 442
  - FDTD, 705–6
  - FDTD hardware design case study, 720–21
  - in FFT computation, 689
  - instruction-level, 95, 234, 796
  - maximum possible, 236
  - memory-centric computation, 794–99
  - PathFinder qualities, 379
  - raw spatial, 219
  - task, 95
- Parameterizable macros, 493
- Parametric generation, 136–38
- PARAMOG module generator library, 338
- PARBIT tool, 773–74
- Partial evaluation, 462–73
  - accomplishing, 462
  - cell logic, 468–69
  - constant folding in logical expressions, 464–66
  - FPGA-specific concerns, 471–73
  - functional specialization, 468–70
  - geometric specialization, 470
  - LUT mapping, 471–72
  - motivation, 463
  - of multipliers, 466–70
  - optimized multiplication circuitry, 468
  - in practice, 464–66
  - process of specialization, 464
  - at runtime, 470–71

- Partial evaluation (*cont.*)
  - static resources, 472
  - true  $x$  value, 470
  - unnecessary logic removal, 466
  - verification of runtime specialization, 472–73
  - of XOR gate, 463
- Partial evaluators, 464
- Partially reconfigurable designs, 70–71
- Partition evaluation, 544, 547
  - design metric, 547
  - dynamic, 548
  - heterogeneous, 548
  - objective function, 547
  - parallel program partitioning, 557
  - sequential program partitioning, 544, 547–48
  - trade-off, 547–48
- Partition-based placement, 312–15
  - bipartitions, 312
  - hierarchical FPGAs, 313
  - multiway partitioning, 312
  - recursive partitioning, 313–14
  - See also* FPGA placement
- Partitioned DWT, 572, 573
- Partitioning, 155, 507
  - automatic HW/SW, 175–76
  - automatic, trend, 540–42
  - binary-level, 559
  - hardware/software, 539–59
  - incremental, 661
  - for island-style FPGAs, 314
  - manual, 540, 546
  - multi-FPGA, 645–46
  - for multiplexed-wire systems, 646
  - multiway, 312
  - recursive, 313–14
  - super-HWOP, 340
  - template, 598–99
- Partitions, 540
- PassAddOrConstant, 673, 674
- PATH algorithm, 310
- PathFinder, 216, 312, 365–80
  - accelerating, 418–22
  - applying  $A^*$  to, 373–74
  - for asymmetric architectures, 373
  - bidirectional switches, 377
  - circuit graph extensions, 376–77
  - circuit graph model, 367
  - communication bandwidth, 421
  - cost function, 368, 375
  - de-multiplexers, 376–77
  - distributed memory multiprocessor implementation, 378
  - enhancements/extensions, 374–77
  - implementation, 366
  - incremental rerouting, 374–75
  - in incrementally rerouting signals, 379
  - Lagrangian relaxation relationship, 376
  - Nair algorithm versus, 370
  - negotiated congestion router, 367–72
  - negotiated congestion/delay router, 372–73
  - parallel, 377–78
  - parallelized, 421
  - QuickRoute and, 379
  - resource cost, 375
  - SC-PathFinder, 366
  - in scheduling communication in computing graphs, 379
  - single-processor, 421
  - symmetric device inputs, 376
- Pattern matchers, 470–71
  - general bit-level, 471
  - instance-specific, 472
  - requirements, 470
- Pattern matching, 470
- Payload processing, 760–62
  - with Bloom filters, 762
  - modules, 760–61
  - with regular expression, 761–62
- PE. *See* Processing elements
- Peak estimation, 478–85
  - analytic, 479–84
  - simulation-based, 484
  - See also* Fixed-point computation
- Perfect component model, 831, 837–38
- Perfect matched layers (PMLs), 702
- Performance
  - Active Pages, 781–86
  - application, 441–44
  - computation, 441–43
  - coverification, 650
  - DA, 508–11
  - dot product, 685–86
  - FDTD hardware design case study, 722–23
  - FFT, 691–92
  - FPGA, 438
  - I/O, 443–44
  - matrix multiply, 682–83
  - multiplexing, 796
  - processor width, 796
- Performance-driven mapping, 282–83

- Perturbation analysis, 489
- Peutil.exe utility, 587
- Phased computations, 104
- Phased reconfiguration, 210–11
  - manager, 117
  - schedule, 215
- Phylogenetic axis, 727
  - POetic tissue, 744
  - subdivision, 735
- Phylogeny, 726
- Physical synthesis, 316
- PIM project, 801
- Pipe and Filter, 108
- Pipeline operators, 184
- Pipeline reconfigurable architecture, 73–74
- Pipelined scheduling, 174–75
- Pipelined SIMD/vector processing, 228–29
- Pipelining, 443
  - deep, 706
  - FDTD hardware design case study, 719–20
  - READ/CALCULATE/WRITE, 716
- PipeRench, 32–35
  - CAD tools, 34
  - DIL, 34
  - PEs, 33
  - physical stripe, 32
  - pipelined configuration, 32
  - virtual pipeline stages, 34
  - See also* Coarse-grained architectures; RPF
- Pipes, 99, 213
- Pitch matching, 330
- Placement directives, 302–4
  - fixed region, 303
  - floating region, 303
  - results, 304
  - See also* FPGA placement
- Placement-driven algorithms, 287–89
- PLAmap algorithm, 292, 869
- Plasma architecture, 427, 428
- POE model, 725–27
  - axes, 727
  - paradigms, 727
- POetic tissue, 743–45
- Pointer independence, 178
- Poly-phase filter bank (PFB), 200
- Population-oriented evolution, 737–38
- Port mapping, 133
- Power cost, 284
- Power estimation, 488–89
- Power-aware mapping, 283–84
- Power-based ranking, 284
- PRAETOR algorithm, 280–82
  - area reduction techniques, 281
  - See also* Mapping algorithms
- PRAM, 786
- Predicates, 167
- Preemption, 242
- Prefetching, 77
- Primary inputs (PIs), 278, 279
- Primary outputs (POs), 278, 279
- Primitive instantiation, 257–59
- Primitive instruction, 808
- PRISM, 53
- Probability of detection (PD), 592
- Processing elements (PEs), 29, 221–22, 225–26
  - data exchange, 221
  - index calculation, 227
  - memory, 221
  - resetting, 221
  - SIMD, 317
- Processor width
  - multiplying versus, 797–99
  - performance, 796–97
- Processors
  - with channels, 124
  - connecting with communication channels, 124
  - customizable instruction, 461–62
  - SIMD, 219
  - VLIW, 164
- Programmable Active Memories (PAM), 49–50
- Programmable chips, 2
- Programmable logic blocks (PLBs), 290–91
- Programmatic circuit generators, 261–63
- Programmer assistance (C compilation), 176–80
  - address indirection, 178
  - annotations, 178–79
  - control structure, 177–78
  - data size declaration, 178
  - large block integration, 179–80
  - local arrays, 177
  - loop fission and fusion, 177
  - loop interchange, 177
  - operator-level module integration, 179
  - useful code changes, 176–77
- Protection, 249–51
  - hardware, 250–51
  - task configuration, 251
- PROXI algorithm, 311–12
- Pterm matching, 841–42

- QRD-RLS (recursive least squares) filtering, 514
- Quartz system, 361
- Quasi-static scheduling, 241
- QuickRoute, 379
- Rack-mount chassis form factor, 770–71
- RAM
  - dedicated, 15
  - static (SRAM), 6, 15, 16–17, 767, 775
- Range propagation, 482–84
- Ranking, power-based, 284
- RaPiD, 36–40, 801
  - application design, 36
  - architecture block diagram, 37
  - datapath overview, 38
  - instruction generator, 39
  - PEs, 38
  - programmable controller, 39
  - programming, 39–40
  - stream generator, 37
  - VICs, 39
- RASP system, 304–5
- RAW project, 801
- Real-time scheduling, 241–42, 558
- Reconfigurable Application Specific Processor (RASP), 60
- Reconfigurable arrays (RAs), 43
- Reconfigurable Communications Processor (RCP), 41
- Reconfigurable computing architectures, 29–45
  - fabric, 30–34
  - impact on datapath composition, 324–26
  - independent RPF coprocessor, 36–40
  - processor + RPF, 40–44
  - RPF integration, 35–44
- Reconfigurable computing systems, 47–62
  - accelerating technology, 56–59
  - AMD/Intel, 55–56
  - CAL, 53
  - circuit emulation, 54–56
  - cloning, 54
  - early, 47–49
  - F + V, 48
  - future, 62
  - issues, 61–62
  - non-FPGA research, 61
  - PAM, 49–50
  - PRISM, 53
  - small-scale, 52–54
  - Splash, 51–52
  - supercomputing, 59–60
  - Teramac, 57–59
  - traditional processor/coprocessor arrangement, 48
  - VCC, 50–51
  - Virtual Wires, 56
  - XC6200, 53–54
- Reconfigurable functional units (RFUs), 41
  - processor pipeline with, 42
  - as RAs, 43
  - RFUOPs, 43
  - super-scalar processor with, 116
  - See also* RFU and processor architecture
- Reconfigurable image correlator, 602–3
- Reconfigurable Pipelined Datapaths. *See* RaPiD
- Reconfigurable processing fabric. *See* RPF
- Reconfigurable static design, 600–4
  - application-specific computation unit, 603–4
  - correlation task order, 601–2
  - design-specific parameters, 601
  - reconfigurable image correlator, 602–3
  - zero mask rows, 601–2
  - See also* ATR
- Reconfigurable supercomputing, 59–60
  - CMX-2X, 60
  - Cray, 60
  - Silicon Graphics, 60
  - SRC, 60
- Reconfiguration
  - configuration, 66–76
  - overhead, 65
  - phased, 210–11
  - phased manager, 117
  - process management, 76–80
  - RTR, 65, 446–47
  - virtual, 741–42
- Reconfiguration management, 65–83
  - configuration caching, 77
  - configuration compression, 81–82
  - configuration data reuse, 82
  - configuration grouping, 76
  - configuration scheduling, 77–79
  - configuration security, 82–83
  - configuration transfer time reduction, 80–82
  - context switching, 80
  - software-based relocation and defragmentation, 78–80
- Recursive partitioning, 313–14



- Recursive Pyramid Algorithm (RPA), 572
- Reflection, 269
- Register Transfer Level (RTL), 87, 129
  - logic organization, 133
  - VHDL description, 133–36
- Regular expression (RE), 761
- Regularity
  - circuit layout, 319
  - datapath composition, 320–22
  - importance, 344
  - inter-HWOP, 339
- Relocation, 71–73, 237
  - device support, 77
  - software-based, 79–80
  - support problem, 80
- Rent's Rule, 642
- Repipelining, 389–90
  - feedforward computations, 389
  - FPGA effects on, 391
  - latency cycles, 390
  - retiming derivation, 389
  - throughput improvement, 390
- Reprogrammable application devices (RADs), 756
- Resonant-tunneling diodes (RTDs), 872
- Resource cost, PathFinder, 375
- Retiming
  - adoption limitation factors, 398
  - area-time tradeoffs, 111
  - Bellman-Ford algorithm, 386
  - benefit, 388
  - constraint system, 385
  - correctness, 386
  - covering and, 286
  - design limitations, 387
  - effect, 287
  - FFs, 286
  - on fixed-frequency FPGAs, 394–95
  - FPGA effects on, 391
  - global set/reset constraint, 387
  - goal, 384
  - implementations, 393–94
  - with initial conditions, 387
  - integrated, 286–87
  - Leiserson's algorithm, 384–86
  - memories, 387
  - multiple clocks and, 387–88
  - operation, 383
  - problem and results, 388
  - sequential control, 110
  - as superlinear, 398
  - See also* C-slow retiming
- RFU and processor architecture, 41–42
  - datapath, 42
  - processor pipeline example, 42
- RGB data
  - conversion, 185–87
  - cycle alignment, 186
- RightSize, 493
- Rock's Law, 855
- Rollback, 845–48
  - communications, 847–48
  - detection, 846
  - recovery, 847
  - scheme, 849
  - for tolerating configuration upsets, 849–50
- Rotation
  - CORDIC, 515–18
  - Given's, 514
  - in matrix form, 515
  - micro-rotations, 526
  - as product of smaller rotations, 515
  - signal flow graph, 518
  - vector growth factor, 518
- Rotation mode, 514–17
  - micro-rotation extensions, 516
  - as z-reduction mode, 517
  - See also* CORDIC
- Routability-driven algorithms, 301
- Routing, 215–16
  - congestion, 302
  - FPGA resources, 348
  - global, 366
  - hierarchical, 10–12, 301
  - horizontal, 308
  - incremental, 661
  - multi-FPGA emulation, 646–50
  - multiplexed-wire systems, 648
  - nearest-neighbor, 7–8
  - negotiated congestion, 367–372
  - Pathfinder-style, 422
  - physical FPGA modifications for, 430
  - programmable resources, 12
  - SCORE, 215–16
  - search wave, 419
  - segmented, 8–10
  - simultaneous placement and, 311–13
  - solutions, 365–66
  - vertical, 308
  - VPR, 314, 372
- Rows
  - skipping, 602
  - zero mask, 601–2

- RPF and processor architectures, 40–44
  - Chimaera, 42–44
  - loosely coupled, 41
  - tightly coupled, 41–42
- RPFs, 29
  - architectures, 30–34
  - coarse-grained, 32–33
  - dynamic, 29
  - fine-grained, 30–32
  - independent coprocessor, 36–40
  - integration into traditional systems, 35–44
  - integration types, 35–36
  - locations in memory hierarchy, 35
  - RaPiD, 36–40
  - static, 29, 901
- RTL. *See* Register Transfer Level
- Rule tables, 738
- Runtime binding, 237–38
- Runtime netlist compilation, 213, 411, 413–14
  - dynamically compiled applications and, 414
  - requirement, 432
- Runtime reconfiguration (RTR), 65, 446–47
  - applications, 447
  - global, 446, 447
  - local, 446–47, 448
- Runtime Reconfigured Artificial Neural Network (RRANN), 447
- Runtime specialization, 472–73
- Sandia algorithm, 594
- SAR. *See* Synthetic Aperture Radar
- SAT solvers, 618–27
  - algorithms, 633
  - backtrack algorithm implementation, 619–24
  - differences among, 633
  - follow-on, 627–33
  - future research, 634–35
  - global topology, 621
  - HW/SW organization, 633
  - implementation issues, 631–33
  - improved backtrack algorithm implementation, 624–27
  - logic engine implementation, 633
  - performance, 630–31
  - problem analysis, 618–19
  - runtime performance, 623
  - simultaneous exploration of multiple states, 635
  - system architecture, 627–30
  - system-level design and synthesis methodologies, 634
  - See also* Boolean satisfiability
- Satisfiability (SAT)
  - Boolean, 282, 613–35
  - FPGA-based solvers, 413
  - problem, 413
- Sblocks, 742
- SC-PathFinder, 366
- Scaling
  - CORDIC algorithm, 517–19
  - CORDIC, compensation, 534
  - FPGA, 411, 412, 431–32
  - Moore’s Law process, 826
  - with transfer functions, 481–82
  - wordlength, 477
- Scheduling
  - configuration, 77–79
  - dynamic, 240–41
  - module-mapped DFG, 174
  - on-demand, 239
  - operating system, 239–42
  - pipelined, 174–75
  - preemption, 242
  - quasi-static, 241
  - real-time, 241–42, 558
  - SCORE, 213–15
  - static, 239–40
  - window-based, 79
- SCORE, 74, 203–217
  - application illustration, 204
  - back-pressure signal, 210
  - C++ integration and composition, 206–8
  - compilation, 212–13
  - compilation flow, 212
  - computations, 205
  - execution patterns, 208–12
  - fixed-size, 211–12
  - as higher-level programming model, 203
  - highlights, 217
  - operators, 205, 206, 207
  - phased reconfiguration, 210–11
  - platforms, 215
  - programming, 205–8
  - runtime, 203, 213–16
  - scalability, 203
  - scheduling, 213–15
  - sequential versus parallel, 211
  - standard I/O page, 211–12
  - stream support, 209–10
  - system architecture, 208–12

- TDF, 205–6
  - virtualization model, 213
- SCPlace algorithm, 310
- SDF, 88, 99–100, 184
- SDRAM memory, 760, 775
- Sea Cucumber, 270
- Search
  - alternative procedures, 497
  - heuristic procedure, 496–97
  - techniques, 496–97
- Search space, 728
- Second-Level Detection (SLD), 592–94
  - as binary silhouette matcher, 593
  - shape sum, 593
  - steps, 593–94
  - target models, 593
  - See also* ATR
- Semantic processing, 767–70
  - dataflow, 769
  - language identification, 767–68
  - of TCP data, 768–70
- Sensitivity list, 135
- Sequential control, 103–5, 110–18
  - with allocation, 104
  - compute task, 110
  - data dependencies, 110
  - data-dependent calculations, 104
  - Deterministic Finite Automata, 103
  - finite-state, 104
  - FSMD, 112
  - instruction augmentation, 115–16
  - phased computations, 104
  - phased reconfiguration manager, 117
  - processor, 114–15
  - single memory pool, 104–5
  - VLIW, 113–14
  - worker farm, 117–18
- Sequential program partitioning, 540
  - alternative region implementation, 544, 549–50
  - Amdahl's Law and, 542, 543
  - automatic, 175–76
  - exploration, 544, 552–57
  - granularity, 544, 545–47
  - ideal speedups, 543
  - implementation models, 550–52
  - manual, 176
  - partition evaluation, 544, 547–48
- Sequential Turing Machines, 103
- Sequentialization, 117
- Set Partitioning in Hierarchical Trees. *See* SPIHT
- Shared instructions, 815–16
- Shared memory, 124–25, 243–44
  - abstraction, 244
  - implementations, 243
  - pools, 124, 125
- Shared-wire global signaling, 628
- Signal-processing primitives, 198
- Signal-to-noise ratio (SNR), 486
- Signal-to-quantization-noise ratio (SQNR), 486
- Silicon Graphics supercomputers, 60
- SIMD (single-instruction multiple data), 120, 219–22
  - algorithm compilation, 226
  - ALU control, 826
  - array size, 224
  - bit-processing elements, 817
  - computing on FPGAs, 219–21
  - datapaths, 815, 818
  - dot-product machine, 220–21
  - extended architecture, 227
  - interprocessor communication model, 224
  - multiple engines, 226–28
  - with pipelined vector units, 229
  - processing architectures, 221–22
  - processing array, 221, 222
  - processors, 219
  - width mismatches, 820
  - width selections, 820
- SIMD/vector processing, 120–22
  - model, 229–30
  - multi-SIMD coarse-grained array, 228
  - multiple SIMD engines, 226–28
  - pipelined, 228–29
  - reconfigurable computers for, 223–26
  - SPMD model, 228
  - variations, 226–28
- Simulated annealing, 306–12
  - accelerating, 415–18
  - annealing schedule, 307
  - complexity, 556
  - cost function, 306
  - distributed, 415
  - hardware-assisted, 418
  - hierarchical algorithm, 310–11
  - key feature, 556
  - low-temperature anneal, 311
  - meta-heuristics, 497
  - move generator, 306
  - parallelized, 416
  - schedule, 307

- Simulated annealing (*cont.*)
  - simultaneous placement and routing, 311–12
  - strengths, 307
  - temperature schemes, 415
  - VPR/related algorithms, 307–11
- Simulated annealing placer, 836
- Simulation, 486
- Simulation-based peak estimation, 484
- Simulink
  - 2D video filtering, 187–91
  - component reuse, 198–200
  - control specification, 194–98
  - high-level algorithm designer, 188
  - image-processing design driver, 185–94
  - library browser, 196
  - mapping video filter to BEE2 platform, 191–94
  - Mask Editor, 198
  - mask parameters, 184
  - operator primitives, 183
  - pipeline operators, 184
  - programming streaming FPGA
    - applications in, 183–202
  - RGB video conversion, 185–87
  - RGB-to-Y diagram, 286
  - SDF, 184
  - subsystems, 184
  - System Generator, 184
  - top-level testbench, 192–93
- Simultaneous logic synthesis, 284–86
- Sine, 437
- Single-context FPGAs, 67–68
- Single-FPGA emulation, 640–41
- Single-instruction multiple data. *See* SIMD
- Single memory pool, 104–5
- Single program, multiple data. *See* SPMD
- Single-rate synchronous dataflow, 99
- Singular value decomposition (SVD), 514
- SLD. *See* Second-level Detection
- Small-scale reconfigurable systems, 52–54
- SMAP algorithm, 291
- Snapshots, 847
- SNORT, 445, 775
  - CPU time, 461
  - database, 761
  - intrusion detection, 753
  - intrusion filter for TCP (SIFT), 765
  - rule-based NID sensor, 763
- Sobel edge detection filter, 188, 191, 201
- Soft macros, 336, 424
- Sorter
  - case study, 357–60
  - with layout information removed, 362
  - recursion and layout, 360–61
  - recursive structure, 357
- Sparing
  - defect tolerance through, 835–39
  - global, 836–37
  - local, 838–39
  - row and column, 837
  - yield with, 834–35
- Spartan-3E, 530
- Spatial computations, 157
- Spatial computing, 155–80
- Spatial orientation trees, 569, 584
- Spatial simulated annealing, 215
- SPIHT, 565–88
  - architecture phases, 581–82
  - bitstream, 578, 579
  - coding algorithm, 570
  - coding engine, 568–71
  - coding phase, 582, 585–86
  - design considerations/modifications, 571–80
  - design overview, 581–82
  - design results, 587–88
  - DWT architectures, 567, 571–75
  - DWT phase, 582
  - engine runtimes, 588
  - Fixed Order, 578–80
  - fixed-point precision analysis, 575–78
  - hardware implementation, 580–86
  - image compression, 565–88
  - image quality, 568
  - LIP, 569
  - LIS, 569, 570
  - LSP, 569, 570, 586
  - maximum magnitude phase, 582, 583–85
  - Morton Scan Ordering, 584
  - optimization, 586
  - performance numbers, 587
  - spatial orientation trees, 569, 584
  - target hardware platform, 581
  - wavelet coding, 569
- Spiral antenna model, 704, 705
- Splash, 51–52
- SPMD (single program, multiple data), 120
  - in parallel processing clusters, 228
  - SIMD versus, 228
- Springtime PCI (SPCI) card, 664
- Square-root operation, 437
- SRC supercomputers, 60

- Standalone Board-level Evolvable System (SABLES), 745
- Static FPGAs, 600
- Static RPF, 29
- Static scheduling, 239–40
- Static-RAM (SRAM), 6, 15, 16–17, 814
  - analyzer, 767
  - cells, 17, 814
  - drawbacks, 17
  - parallel banks of, 775
- Straight-line code, 156
- Stream computations, 217
  - compilation, 212–13
  - execution patterns, 208–12
  - organization, 203–17
  - programming, 205–8
  - runtime, 213–16
  - system architecture, 107–110, 208–12
- Stream generator, 37
- Streaming dataflow, 107–10
  - with allocation, 102–3
  - data presence, 108–9
  - datapath sharing, 109
  - dynamic, 100–2
  - interconnect sharing, 110
  - streaming coprocessors, 109–10
- Streaming FPGA applications, 183–202
  - component reuse, 198–200
  - high-performance datapaths, 184
  - image-processing design driver, 185–94
- Streams, 37, 99, 244–46
  - abstraction, 245–46
  - input, 99
  - multirate, 100
  - persistence, 245–46
  - SCORE, 209–10
  - video, 185, 202
  - write, 206
- Structural mapping algorithms, 278–84
  - area-oriented, 280–82
  - cut generation, 279
  - DAOmap, 282–83
  - dynamic programming basis, 278–79
  - FlowMap, 279, 282
  - IMap, 281–82
  - LEKO, 282
  - performance-driven, 282–83
  - power-aware, 283–84
  - PRAETOR, 281–82
  - See also* Technology mapping
- Subsystems, 184
  - with configurable delays, 187
  - stream-based filtering, 190–91
  - tiled, 198–200
- Super-HWOP, 340–41
  - building, 342–43
  - microplacement, 342, 343
  - partitioning, 340
- Superslices, 340, 342
- Swap negotiation, 417
- Swappable logic units (SLU), 74
- SWIM project, 801
- Switch blocks
  - example architecture, 10
  - island-style architecture with, 9
- Switch boxes, 409
  - connectivity, 429
  - style and routability, 429
- Synchronization, 248–49
  - deadlock prevention, 249
  - explicit, 248
  - implicit, 248–49
  - thread-style, 248
- Synchronous Data Flow. *See* SDF
- Synopsys FPGA compiler, 393
- Synoptix, 493
- Synplicity Identify tool, 272
- Synthetic Aperture Radar (SAR)
  - ATR in, 591
  - Sandia real-time, 592
- System architectures, 107–25
  - bulk synchronization pattern, 118–19
  - cellular automata, 122–23
  - data parallel, 119–22
  - hierarchical composition, 125
  - multi-threaded, 123–25
  - sequential control, 110–18
  - streaming dataflow, 107–10
- System Generator library, 184
- SystemC, 205, 542
- Systolic image array pipeline, 603–4
- T-VPack algorithm, 305, 306
- Tail duplication, 164
- Task configuration protection, 251
- Task Description Format (TDF), 205–6
  - behavioral operator, 206
  - compositional operator, 208
  - operators, 208
  - as portable assembly language, 207
  - specification, 206, 207
- Taylor coefficients, 490
- Taylor expansion transformation, 490

- TCP processing, 758–60
  - block diagram, 760
  - circuit development, 759
  - semantic, 768–70
  - See also* Network processing
- Techmapper class, 260, 264, 272
- Technology mapping, 277–93
  - algorithms, 277
  - algorithms for heterogeneous resources, 289–92
  - functional algorithms, 277
  - integrated, 278, 284–89
  - in logic synthesis flow, 278
  - optimal solutions, 285
  - structural algorithms, 277, 278–84
- Templates
  - correlation between, 598
  - grouping example, 599
  - partitioning, 598–99
- Teramac, 57–59
  - applications, 58–59
  - features, 58
  - in hardware prototyping applications, 427
- Terasys Integrated Circuit, 221
- Terminal propagation, 314, 315
- Ternary content addressable memory (TCAM), 764
- Test pattern generation, 614, 615
- Testbenches
  - dynamic, 269–70
  - JHDL, 265–66
- Theoretical underpinnings, 807–27
- Tightly coupled RPF and processor architecture, 41–42
- Tiled subsystems, 198–200
- Timing-driven algorithms, 301, 302
- Topology matching, 329
- Transaction application protocol interface (TAPI), 664
- Transaction-based host-emulator
  - interfacing, 650–51
- Transfer functions
  - for nonrecursive systems, 480
  - scaling, 481–82
- Transformations, 555
- Transient faults, 830
  - feedforward correction, 844–45
  - rollback, 845–48
  - tolerance, 843–48
- Translation lookaside buffer (TLB), 247, 397
- Triple modular redundancy (TMR), 844–45, 849
- Triple-key DES, 83
- Truth tables, 4
- Turing Machine, 96
- Turing-Complete compute models, 97, 119
- Two-dimensional placement, 336–37
  - bin-based, 336
  - constrained, 335–36
- Two-dimensional video filtering, 187–91
- Uniaxial PML (UPML), 702, 706, 721
- User datagram protocol (UDP), 758
- Variable fixed-rate representation, 577
- Variable-length chromosome GAs (VGAs), 735
- Variables
  - live at exits, 168–69
  - scalar, in memory, 169
- Vector architectures, 120–21
  - functional units, 121
  - motivation, 120
  - sequential controller, 120
- Vector coprocessors, 121–22
- Vector functional units, 121, 229
- Vectoring mode, 519–20
  - convergence, 520
  - implementations, 519
  - range extension, 525
  - simulation, 519
  - as *y*-reduction mode, 519
  - See also* CORDIC
- Verilog, controller design with, 197
- Very High-Speed Integrated Circuit Hardware Description Language (VHDL), 87–88, 129–53
  - Active Pages, 782
  - concurrent statements, 144, 150
  - controller design with, 197
  - delta delay, 150
  - design development, 130
  - FSM datapath example, 138–49
  - gates, 130
  - hardware compilation flow and, 150–52
  - hardware descriptions, 153
  - hardware module description, 132–33
  - limitations, 153
  - multivalued logic, 150
  - parametric hardware generation, 136–38
  - popularity, 129
  - port mapping, 133

- ports, 133
- programming, 130–50
- RTL description, 133–36
- sequential, comparison, 149
- signals, 133
- structural description, 130–33
- submodules, 133
- syntax, 153
- Very long instruction word (VLIW), 61, 113–14, 795–97
  - computational elements, 795
  - processors, 164
  - of single multiply and add datapath, 113
  - time-slicing, 795
  - width, 797
- Virtual circuit identifier (VCI), 756
- Virtual Computer, 50–51
- Virtual instruction configurations (VICs), 29
  - Chimaera architecture, 43–44
  - RaPiD, 39
  - speculative execution, 43
- Virtual memory, 246–47
- Virtual path identifier (VPI), 756
- Virtual reconfiguration, 741–42
- Virtual Wires, 56
- Virtualized I/O, 72
- Virtually addressed caches, 397
- VirtuaLogic family, 642
- VirtuaLogic VLE emulation system, 639, 653–65
  - array boards, 653–55
  - case study, 653–65
  - design clock cycle, 656
  - design partitions, 655
  - emulation mapping flow, 654
  - emulator system clock speed, 665
  - incremental compilation of designs, 661–64
  - incremental mapping, 662
  - incremental partitioning, 661
  - incremental path identification, 661
  - incremental routing, 661
  - inter-FPGA communication, 656
  - interfaces for coverification, 664–65
  - intra-FPGA computation, 656
  - multidomain signal transport, 658, 659, 660
  - multiported memory mapping, 657
  - netlist comparison, 661
  - parallel FPGA compilation, 665
  - partitioning, 654
  - software flow, 654–57
  - specialized mapping techniques, 657
  - statically scheduled routing, 656
  - structure, 653–54
  - See also* Logic emulation systems
- Virus protection, 763–64
- VLSI, CORDIC algorithm in, 514
- VPack algorithm, 305
- VPR, 307–11
  - annealing schedule, 307
  - delay computation, 309–10
  - enhancements, 310
  - move generator, 307
  - range limit update, 307–8
  - recomputation, 310
  - router, 314, 372
- VStation family, 642
- Washington University Gigabit Switch (WUGS), 756
- Wavelets, 567
  - coding, 569
  - spatial orientation trees, 569
- Wildcards (\*), 152, 761
- WildStar-II Pro FPGA board, 708–9
  - block diagram, 709
  - features, 708
  - memory hierarchy levels, 713
  - Xilinx Virtex-II Pro FPGAs on, 722
- Window-based scheduling, 79
- Wire congestion, 312
- Wired-OR diode logic array, 859–60
- Wordlength
  - control over, 523
  - scaling, 477
- Wordlength optimization, 485–97
  - area models, 485–96
  - error estimation, 485–96
  - problem, 499
  - search techniques, 496–97
  - simulation-based methods, 487
- Word-level optimization, 339–40
- Word-wide datapaths, 216, 815
- Worker farms, 117–18
- Worm detection, 766–67
- Worm protection, 763–64
- Xilinx 6200 series FPGA, 53–54, 81
  - cell configuration, 732
  - CLBs, 741
  - EHW platforms and, 740–41
  - “open” bitstream, 408
  - wildcard registers, 81

## Xilinx

- ChipScope, 271
- Core Generator IP, 348
- EasyPath series, 842
- Embedded Development Kit (EDK), 197
- MicroBlaze, 194, 347
- Virtex 2000E FPGA, 581
- Virtex-4, 530, 533
- XC 4036EX FPGA, 632
- XC4VLX200 FPGA, 623
- XC4000 library, 596
- Xilinx Virtex-II Pro, 23–26, 68, 83, 530, 721
  - CLBs, 23–24
  - IBM PowerPC 405–D5 CPU cores, 25
  - logic architecture, 23–25
  - multiplier blocks, 24
  - routing architecture and resources, 25–26
  - SelectRAM+, 24, 25
  - on WildStar-II Pro board, 722
  - XC2VP100, 24
- XOR gates, 463, 464
- Y-reduction mode, 519
- YaMoR, 739
- Yield, 832–85
  - Law of Large Numbers impact, 835
  - perfect, 833–34
  - with sparing, 834–35
  - See also* Defect tolerance
- Z-reduction mode, 517
- Zero mask rows, 601–2