Index

A
A operators, 481
A quantifiers, 476
Absolute transition delays, 149–150
Abstraction
levels of, 44–46
for properties, 466
acc_next_cell routine, 146
acc_next_driver routine, 115
acc_next_port routine, 146
acc_set_value routine, 146
Acceptance states and strings, 359, 362–363
Actions in assertions, 234
Active events, 22, 82
Address maps, CAM for, 190–194
Address space for pseudorandom test
   generators, 229
ADDs (algebraic decision diagrams), 421–422
AF operator in CTL, 480
Affirmative assertions, 232–233
AG operator in CTL, 480
Alert on detection method, 121
Alert on output transition method, 121
Alert task, 186
Alerts, 185–186
   levels of, 186
   self-checking codes for, 169
Algebraic decision diagrams (ADDs), 421–422
Algorithmic abstraction level, 44
ALU (Arithmetic Logic Unit), modules
   for, 35
always blocks
   at signs in, 47
   profiling statistics for, 123
Ancestor vertices in depth-first searches, 375
AND gates and operators
   for Boolean functions, 343–344
   in clock gating, 41–42
   in RTL coding, 36
   for sequences, 254, 484
   update events with, 84–85
Architectural function and specifications
   for pseudorandom test generators, 228
   in test plans, 217–222
Arithmetic Logic Unit (ALU), modules
   for, 35
Arrays
   instantiating, 45, 51
   macro files for, 54
   mapping to, 188–189
   as recognizable components, 47
   tracing, 58–59, 308–311
Asserting cuts, 441
Assertions, 232
   built-in, 247–248
   components of, 233–234
   concurrent, 249–250
      multiple clocks for, 258
      sequence connectives for, 253–258
      sequence constructors for,
      250–253
      sequences for, 259
   system functions for, 258
   container, 245–247
   defining, 232–233
   immediate, 248–249
   interval constraints for, 242–243
   one-hot and one-cold signals in,
   237–238
   parity in, 236
   problems, 280–286
   sequential, 238–241
   signal ranges as, 235
Index

Assertions, continued
- on signals
  - change patterns, 241–242
  - membership, 236–237
  - stylized comment, 248
  - summary, 279
  - in symbolic simulation, 446
  - for test benches, 194
  - time windows for, 242
  - unclocked timing, 243–245
  - unknown values in, 236

Assessment, response. See Response assessment

Assignments
- continuous, 18–20
- nonblocking. See Nonblocking assignments
- in race problems, 39
- in search-based algorithms, 434, 437

Asynchronous logic, synthesizability of, 57

Asynchronous memory arrays, 47

Asynchronous stimuli application, 159–160


At signs (@) in always blocks, 47

Automatons
- finite-state. See Finite-state automatons and languages
  - language containment with, 487–489
  - property checking in, 485–486
  - for sequential specification, 467–468

AX operator in CTL, 479

B

Back edges in depth-first searches, 375

Back ends in simulators, 74–75

Back tracing algorithm for memory, 309–310

Backtracking in search-based algorithms, 435–437

Backward debugging, 301–302

Backward faulty state reachability analysis, 500–501

Backward traversal of states, 500

Base clocks, 151

Batch simulation mode, 127

BCP (Boolean constraint propagation), 437

BDDs. See Binary decision diagrams (BDDs)

Behavioral abstraction level, 44–45

Behavioral models, 53

Benchmarks in system level verification, 216

BFS (breadth-first searches), 378–380

Bifurcation in debugging, 300–301

Binary decision diagrams (BDDs), 388–390
  - Boolean operations with, 390–395, 399–404
  - construction algorithms for, 395–397
  - edge-attributed, 412–414
  - efficiency from, 524
  - in formal method-based verification, 12
  - reduction operation for, 397–398
  - restriction operations for, 398
  - shared, 412–413
  - size of, 405–406, 410–411, 416, 524
  - variables in, 397, 404–411
  - zero-suppressed, 414–417

Binary moment diagrams (BMDs), 423–424

Binary test case reduction algorithm, 291–292

Binary trees, 434–438

Binate clauses, 434

BIST (Built-In Self Test) structure, 224

Bit operations, converting to bus, 48–50

Bit ranges, 188

Bit streams, applying, 139

Black boxing SCCs, 105

Block coverage, 261–263

BMDs (binary moment diagrams), 423–424

Boolean constraint propagation (BCP), 437

Boolean domains vs. sequence domains, 250–251

Boolean operations, 342
  - with binary decision diagrams, 390–395, 399–404
  - characteristic, 351–353
  - incompletely specified, 349–350
Index

in LTL formulas, 475
operators for, 353–358
for properties, 467
pseudo, 420–422
representation of, 342–346
symmetric, 346–348
Boolean satisfiability, 346, 430–431
implication graphs and learning in,
438–442
resolvent algorithms for, 431–434
search-based algorithms for, 434–438
Bottlenecks, 78, 111
Bottom-up process, 44
Bound properties, 446
Braces ({}), for set members, 332
branch routine, 436
Branching instructions with
pseudorandom test generators,
230
Branching time models, 474, 476
Breadth-first searches (BFS), 378–380
Break points, 128
Buchi automata, 363
complemented, 488
fairness constraints in, 482
language containment in, 507
language emptiness in, 512–513
Bugs
debugging process. See Debugging
process
in formal method-based verification,
13, 17
latency of, 213
priority of, 217, 296
in simulation-based verification,
10–11
tracking systems for, 295–296
Built-in assertions, 247–248
Built-In Self Test (BIST) structure, 224
Built-in system functions, 258
Bus contention
in four-state simulation, 64
in two-state simulation, 109
Bus functional model, 199–203
Bus operations
coding for, 37
converting to bit, 48–50

C
C/C++ language
array mapping to, 188–189
compiling into, 77
for self-checking codes, 173–174
for simulations, 51, 63–64
Cache units, testing, 225
CAM (content addressable memory),
190–194
Canonicity
in binary decision diagrams, 409
in formal method-based verification,
12
of reduced OBDDs, 395
of representation, 12, 388
Capacitance in gate delay, 118
Capacity of simulators, 111–112
Cardinality of sets, 332
Cartesian products, 333–334
Case splitting, 447
Categories of functionality, 218–222
Cause and effect, assertions for, 242
Cells
initialization in, 54–55, 145
in library files, 54, 113
Central processing unit (CPU) design,
modules for, 35
Centralized resources, 54–55
Change delta, assertions for, 242
Characteristic functions
Boolean, 351–353
in symbolic finite-state machines, 495
for symbolic simulation, 450–453
Check-in tests, 318
Check pointing, 128, 292–294
Child vertices in depth-first searches, 375
Chronological backtracking in
search-based algorithms, 437
Circuit reduction, 289–290
Circuits in verification paradigm, 6
Circular queues, 81, 239–240
Classic cofactor for Boolean functions, 355
Clauses
for Boolean functions, 343
in Boolean satisfiability, 430, 434
in search-based algorithms, 435
Clock domain analysis
back ends for, 75

Clock domains
crossing, 43
partitioning algorithm for, 94

Clock gating
in cycle-based simulation, 61
in verification, 41–42

Clock generation and synchronization, 148
absolute transition delays in, 149–150
clock independence and jitter, 152–153
delta delay in, 153–154
explicit and toggle methods, 148–149
multipliers and divider ins, 151–152
organization of, 154–155
in test harnesses, 141
time unit and resolution in, 150
time zero clock transitions in, 150

Clock trees
in cycle-based simulation, 60
processing, 95

Clocks for concurrent assertions, 258
CMOS constructs, 36
CNF (conjunctive normal form), 343–346

Co-simulation, 161
with encapsulated models, 125–126
with reference models, 174–178

Code coverage, 260–261
blocks, 261–263
expressions, 263–266
instrumentation for, 267–269
ordering tests in, 270
paths, 263
performance and methodology for, 269–270
sequences, 267
in simulation-based verification, 10
statements, 261
states, 266–267
toggles, 267
transitions, 267

Code generation
in cycle-based simulators, 96–100
simulators for, 75
for verification, 25–27

Code profiling, 52

Coding guidelines
cycle-based simulation, 59–62
debugability, 57–59
functional correctness, 26–27
structural checks, 36–38
summary, 38
syntactical checks, 27–36
hardware simulation/emulation, 62–64
linter design, 66–67
portability and maintainability, 52–56
problems, 67–71
simulation performance. See Simulation
summary, 67
synthesizability, 57
timing correctness, 39–43
tool compatibility, 56–59
two-state and four-state simulation, 64–66

Cofactors
generalized
for Boolean functions, 355–356
for computational improvements, 527–529
operations, 353
Combinational equivalence, 388, 424–425
Combinational expressions, 234
Combinational loops
coding for, 36–37
compiling, 102–105
cycle-based simulators for, 92–93
event-driven simulators for, 86–87

Command files, 113, 127–128
Command line for compilation, 114
Command registers, 219
Comment assertions, 248
Comments, 47, 55
Commercial assertions, 247–248
Communication channels in cycle-based simulators, 96
Comparison mechanisms
in expression coverage, 265
in verification paradigm, 6
Compatibility, tool, 26, 56–59
<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compilation</td>
<td>565</td>
</tr>
<tr>
<td>combinational loops, 102–105</td>
<td></td>
</tr>
<tr>
<td>command line for, 114</td>
<td></td>
</tr>
<tr>
<td>hardware, 108</td>
<td></td>
</tr>
<tr>
<td>into high-level language, 77–78</td>
<td></td>
</tr>
<tr>
<td>incremental, 126–127</td>
<td></td>
</tr>
<tr>
<td>Compile time</td>
<td></td>
</tr>
<tr>
<td>in data dumping, 299</td>
<td></td>
</tr>
<tr>
<td>of simulators, 111–112</td>
<td></td>
</tr>
<tr>
<td>Compiled code, 75</td>
<td></td>
</tr>
<tr>
<td>event-driven simulators, 101</td>
<td></td>
</tr>
<tr>
<td>versus interpreted, 111</td>
<td></td>
</tr>
<tr>
<td>library file images, 113</td>
<td></td>
</tr>
<tr>
<td>simulation systems, 78</td>
<td></td>
</tr>
<tr>
<td>Compilers, 74–79</td>
<td></td>
</tr>
<tr>
<td>Complements of sets, 334</td>
<td></td>
</tr>
<tr>
<td>Completely specified Boolean functions,</td>
<td></td>
</tr>
<tr>
<td>349</td>
<td></td>
</tr>
<tr>
<td>Completely specified finite-state</td>
<td></td>
</tr>
<tr>
<td>automatons, 361</td>
<td></td>
</tr>
<tr>
<td>Completely unreduced decision diagrams,</td>
<td></td>
</tr>
<tr>
<td>415</td>
<td></td>
</tr>
<tr>
<td>Component level optimization, 51–52</td>
<td></td>
</tr>
<tr>
<td>Component machines, 363</td>
<td></td>
</tr>
<tr>
<td>Composite operators, 265</td>
<td></td>
</tr>
<tr>
<td>Computation tree logic. See CTL</td>
<td></td>
</tr>
<tr>
<td>(computation tree logic)</td>
<td></td>
</tr>
<tr>
<td>Computational improvements, 524</td>
<td></td>
</tr>
<tr>
<td>early quantification, 525–527</td>
<td></td>
</tr>
<tr>
<td>generalized cofactors, 527–529</td>
<td></td>
</tr>
<tr>
<td>Computed tables for binary decision</td>
<td></td>
</tr>
<tr>
<td>diagrams, 401</td>
<td></td>
</tr>
<tr>
<td>Computer farms, 319</td>
<td></td>
</tr>
<tr>
<td>Concurrent assertions, 249–250</td>
<td></td>
</tr>
<tr>
<td>multiple clocks for, 258</td>
<td></td>
</tr>
<tr>
<td>sequence connectives for, 253–258</td>
<td></td>
</tr>
<tr>
<td>sequence constructors for, 250–253</td>
<td></td>
</tr>
<tr>
<td>sequences for, 259</td>
<td></td>
</tr>
<tr>
<td>system functions for, 258</td>
<td></td>
</tr>
<tr>
<td>Concurrent processes, 17–19</td>
<td></td>
</tr>
<tr>
<td>cond_signal function, 98–99</td>
<td></td>
</tr>
<tr>
<td>cond_wait function, 98</td>
<td></td>
</tr>
<tr>
<td>Conditional cases, overlapping, 32–34</td>
<td></td>
</tr>
<tr>
<td>Conditional statements, 263</td>
<td></td>
</tr>
<tr>
<td>Conditions for assertions, 232</td>
<td></td>
</tr>
<tr>
<td>Cone tracing in debugging, 306–308</td>
<td></td>
</tr>
<tr>
<td>Configuration of verification environment,</td>
<td></td>
</tr>
<tr>
<td>196–199</td>
<td></td>
</tr>
<tr>
<td>Configuration specification files, 317</td>
<td></td>
</tr>
<tr>
<td>Conflicted clauses in search-based</td>
<td></td>
</tr>
<tr>
<td>algorithms, 435</td>
<td></td>
</tr>
<tr>
<td>Conflicts in implication graphs, 440–441</td>
<td></td>
</tr>
<tr>
<td>Conjunctive normal form (CNF), 343–346</td>
<td></td>
</tr>
<tr>
<td>Connection rules, 34–35</td>
<td></td>
</tr>
<tr>
<td>Connectives, sequence, 253–258</td>
<td></td>
</tr>
<tr>
<td>Connectivity traces, 117</td>
<td></td>
</tr>
<tr>
<td>Consensus in Boolean satisfiability, 431</td>
<td></td>
</tr>
<tr>
<td>Constants</td>
<td></td>
</tr>
<tr>
<td>macros for, 54</td>
<td></td>
</tr>
<tr>
<td>in signal range assertions, 235</td>
<td></td>
</tr>
<tr>
<td>Constraints</td>
<td></td>
</tr>
<tr>
<td>in equivalence checking, 429–430</td>
<td></td>
</tr>
<tr>
<td>fairness. See Fairness constraints</td>
<td></td>
</tr>
<tr>
<td>in symbolic simulation, 447–450</td>
<td></td>
</tr>
<tr>
<td>Construction algorithms, 395–397</td>
<td></td>
</tr>
<tr>
<td>Container assertions, 245–247</td>
<td></td>
</tr>
<tr>
<td>container operator, 257</td>
<td></td>
</tr>
<tr>
<td>Content addressable memory (CAM), 190–194</td>
<td></td>
</tr>
<tr>
<td>Continuous assignments, 18–20</td>
<td></td>
</tr>
<tr>
<td>Continuous time in model checking, 469</td>
<td></td>
</tr>
<tr>
<td>Control checks in assertions, 247</td>
<td></td>
</tr>
<tr>
<td>Controllable distributions, 227</td>
<td></td>
</tr>
<tr>
<td>Converting scalar operations to vector,</td>
<td></td>
</tr>
<tr>
<td>48–51</td>
<td></td>
</tr>
<tr>
<td>Counter-example generation, 501–502</td>
<td></td>
</tr>
<tr>
<td>Scountones function, 258</td>
<td></td>
</tr>
<tr>
<td>Coverage, 259–260</td>
<td></td>
</tr>
<tr>
<td>code. See Code coverage</td>
<td></td>
</tr>
<tr>
<td>functional, 273–278</td>
<td></td>
</tr>
<tr>
<td>item coverage and cross-coverage, 278–279</td>
<td></td>
</tr>
<tr>
<td>measures of, 260–261, 320</td>
<td></td>
</tr>
<tr>
<td>parameters, 260, 270–272</td>
<td></td>
</tr>
<tr>
<td>problems, 280–286</td>
<td></td>
</tr>
<tr>
<td>summary, 279</td>
<td></td>
</tr>
<tr>
<td>Coverage areas, 270</td>
<td></td>
</tr>
<tr>
<td>cpu_apply_vector task, 158</td>
<td></td>
</tr>
<tr>
<td>CPU (Central processing unit) design,</td>
<td></td>
</tr>
<tr>
<td>modules for, 35</td>
<td></td>
</tr>
<tr>
<td>CRC (cyclic redundant code), 50</td>
<td></td>
</tr>
<tr>
<td>Cross-coverage, 278–279</td>
<td></td>
</tr>
<tr>
<td>Cross-domain events, 105–106</td>
<td></td>
</tr>
</tbody>
</table>
Cross edges in depth-first searches, 375
CTL (computation tree logic)
  fairness constraints in, 483, 520–523
  fix-point computation in, 513–520
  formula checking in, 489–493
  in model checking, 477–482
  relative expressiveness of, 483–484
  symbolic computation, 513–523
Cubes for Boolean functions, 342
Curly braces ({} for set members, 332
Cut assertion in implication graphs, 441
Cut points
  in equivalence checking, 426
  in formal method-based verification, 12
Cycle-based simulation, 9, 59–62, 88
  back ends for, 75
  clock domain analysis in, 93–95
  clock trees in, 60, 95
  code generation and simulation
    control in, 96–100
    combinational loop compilation for, 102–105
    for combinational loop detection, 92–93
  vs. event-driven, 59–62, 110–111
  execution order in, 95–96
  leveling in, 88–92
  limitations of, 10
  performance of, 100
  with symbolic simulators, 445
Cycle delay operator, 484
Cycle-level accuracy, 175
Cyclic redundant code (CRC), 50

Data dependency for pseudorandom test
generators, 229
Data dumping in simulation, 297–300
Data-in registers, 219
Data-out registers, 219
Data structures
  for hardware concurrency, 18–19
  tracing, 58
Data types in performance, 51
Davio expansions, 417–418
De Morgan’s law, 344
Debugability, 56–59
  assertions for, 232
  of simulators, 111–112
Debuggers
  functionality in, 312–315
  in test plans, 223
Debugging process, 287–288
  data dumping in, 297–300
  error tracking systems in, 295–296
  failure capture in, 288–289
  isolation in. See Isolation in debugging
  problems, 322–330
  saving debug sessions, 292–294, 313
  scope reduction in, 289–294
  signal traces for, 167
  simulators, 114–116
  summary, 321
Decision diagrams, 387–388
  binary. See Binary decision diagrams
    (BDDs)
  binary moment, 423–424
  Boolean satisfiability. See Boolean
  satisfiability
equivalence checking based on,
  424–430
  ordered functional, 417–419
  problems, 458–464
  and pseudo Boolean functions,
  420–422
  summary, 457
Decision variables in implication graphs,
  438–440
Declaration of module ports, 55
Declarative assertions, 249
Decomposition
  of directed graphs, 377

DACs (directed acyclic graphs)
  binary decision diagrams. See Binary
decision diagrams (BDDs)
  components of, 103–105, 377–378
  implication graphs as, 439
Dangling nets, 51
Data, memory for, 78
Data blocks in functional coverage,
  273–274
Data checks in assertions, 247

D
Index 567

in hierarchical verification, 214
Decrement signals, assertions for, 242
Defining
  assertions, 232–233
  sets, 333
Delay models, 116–119
Delay operator, 484
Delay simulators, zero vs. unit-delay, 109–110
Delays
  in clock gating, 41, 118
  in clock generation, 148–150
  in cycle-based simulation, 60
  propagation, 120–121
  in synchronization, 153–154
  synthesizability of, 57
  with update events, 84–85
Delta delay, 153–154
Dependencies for pseudorandom test generators, 229
Depth-first searches and sorts, 89–93, 373–378
Descendant vertices in depth-first searches, 375
Design
  error types of, 4
  size reductions for, 291
  update and maintenance of, 315–318
Design code, separating from checking code, 171–172
Design memory, 190, 194
Design profiling, 122–123
Design state dumping, 162–167
Design views in debugging isolation, 311–312
Detection of bugs
  self-checking codes for, 169
  signal traces for, 167
DFS (depth-first searches and sorts), 89–93, 373–378
Diagrams, decision. See Binary decision diagrams (BDDs); Decision diagrams
diff command, 167
Difference of functions, 356
Directed edges
  in depth-first searches, 375
for finite-state automata, 372
Directed graphs (digraphs)
  binary decision diagrams. See Binary decision diagrams (BDDs)
  components of, 103–105, 377–378
  in cycle-based simulators, 89
  for finite-state automata, 359, 372
  implication graphs as, 439
Directed tests, 9, 227
Discrete time in model checking, 469
Disjoint sets, 333
Disjunctive normal form (DNF), 343–346
Display code in performance, 51
$display task, 59, 165, 299–300
Distributed event management, 105–106
Distributive laws of Boolean algebra, 344–345
Dividers, clock, 151–152
DNF (disjunctive normal form), 343–346
Domain analysis
  back ends for, 75
  cycle-based simulators, 93–95
Domain-crossing glitches, 43
Dominator vertices in implication graphs, 439
Don’t care values, 29–30, 349–350
Drivers
  bus, coding for, 37
  tracing, 306–308
Duality principle, 344
$dump_nodes task, 163–164
$dumpall task, 166
$dumpfile statement, 299
$dumpflush task, 166
Dumping
  design state, 162–167
  memory, 186–190
  node values, 298
  in performance, 51
  selective, 58, 115–116
  simulation data, 297–300
$dumplimit task, 166
$dumpmemb task, 190
$dumpmemh task, 190
$dumpvar task, 116, 299
$dumpvars task, 166
Duty cycles in clocks, 150
Dynamic programming for binary decision diagrams, 402
Dynamic table mapping, 190
Dynamic variable ordering in binary decision diagrams, 408

E
E operator
  checking, 491–493
  in CTL, 477, 481
E quantifier, 476
Early quantification, 525–527
Ease of use for code coverage metrics, 261
ECC (error correction code)
  transformations, 50–51
ECOs (engineering change orders), 12
Edge-attributed BDDs, 412–414
Edges
  in binary moment diagrams, 424
  in breadth-first searches, 379
  in depth-first searches, 375
  for finite-state automatons, 372
Editing files, 317
EF operator in CTL, 480
Efficiency from BDDs, 524
EG operator
  checking, 491–494
  in CTL, 480
Elaborators, 74–75
Elements of partitions, 335
else directives, 197–198
Embedded sequential state, 28–32
Empty clauses in Boolean satisfiability, 433
Empty languages, 488–489, 508
Emulation, hardware, 62–64, 106–108, 125–126
Emulation code, 75
Enable signals, mutual exclusion of, 37
Encapsulated models, co-simulation with, 125–126
End-time groups, 91
ended operator, 258
endif directives, 197
Ending states in finite-state automatons, 362
endsequence keyword, 252
Engineering change orders (ECOs), 12
Environment modeling, symbolic simulation for, 447
Equivalence
  finite-state machine, 370–371
  function, 387–388
  set, 335–338
  state, 365–370
Equivalence checking, 4, 465
  decision diagrams for, 424–430
  in formal method-based verification, 12
  problems, 458–464
  specifications for, 466
  summary, 457
  in symbolic computation, 502–506
Error correction code (ECC)
  transformations, 50–51
Error injectors, 184–185
Error tracking systems, 295–296
Errors
  alert mechanisms for, 185–186
  assertions for. See Assertions
  in coding guidelines, 27
  counter-examples for, 501–502
  in formal method-based verification, 12
  linters for, 66–67
  in property automata, 485
  self-checking code for, 170
  in simulation-based verification, 9
  in specifications, 6, 17
  types of, 4
Evaluation, event, 83–85
Even parity in assertions, 236
Event-based logic, synthesizability of, 57
Event-counting constructs, 39
Event-driven simulators, 10, 80
  compiled, 101
  vs. cycle-based, 59–62, 110–111
  event-driven scheduling algorithms, 87–88
  event propagation in, 85–86
  performance of, 100
  scheduling semantics for, 82–83
  with symbolic simulators, 445
time advancement and oscillation
  detection in, 86–87
  timing wheels/event managers in,
  80–82
  update and evaluation events in,
  83–85
Event managers, 80–82
Events
  in hybrid simulators, 105–106
  leveled for zero-delay simulation,
  101–102
  propagation of, 85–86
  in RTL coding, 36
  in simulation-based verification, 9–10
  update and evaluation, 83–85
EX operator
  checking, 490
  in CTL, 479
Exception handling, 78
Execution in zero time, 19
Execution order in cycle-based simulators,
  95–96
Existential operation, 357–358
Existential problem, 357
Exit subroutines for pseudorandom test
  generators, 230
Expansions for ordered functional
decision diagrams, 417
Expected results for pseudorandom test
  generators, 229
Expected values for nodes, 301
Explicit connections, 34
Explicit methods for clocks, 148–149
Explicit model checking, 494
Expression coverage, 263–266
Expressions in assertions, 234, 259
Expressiveness, relative, 483–484

F
Failure capture, 288–289
Fair formulas, 493
Fair paths, 520–521
Fair state sets, 493
Fairness constraints
  CTL model checking with, 520–523
  in model checking, 482–483
  property checking with, 493–494
  in symbolic computation, 507–513
  Fairness properties, 467
  False loops in cycle-based simulation,
  59–60
  Fanins with topological sorts, 91
  Fanouts
    in event propagation, 85–86
    for finite-state automata, 372
    with topological sorts, 91
  Faulty state reachability analyses
    backward, 500–501
    forward, 499–500
$display statement, 167
FF configuration coding, 37–38
Field programmable gate array (FPGA)
  hardware simulators, 10, 62
  back ends for, 75
  characteristics of, 106–107
File I/O in test bench interfaces, 195
File structure for simulators, 113–114
Files
  editing, 317
  in modules, 53
  revision control for, 315–318
Final states in finite-state automata, 359
Finite-state automata and languages,
  359–363, 369
  breadth-first searches in, 378–380
  depth-first searches in, 373–378
  finite-state machine equivalence in,
    370–371
  graph algorithms for, 371–373
  product automata and machines,
    363–365
  state equivalence and machine
    minimization in, 365–370
Finite-state machines, 361
  coding, 47–48
  in design, 311–312
  equivalence in, 370–371
  in finite-state automata, 359
  and Kripke structures, 471–474
  state traversal in, 495–501
first_match operator, 255
Fix-point computations, 513–520
Fix-point operators for CTL model, 492
Flipping variables, 435
Floating point units (FPUs), modules for, 35
Flow
  in formal method-based verification, 14–15
  in simulation-based verification, 11
force constructs
  for node value changes, 115
  in RTL coding, 36
Forests in depth-first searches, 374
fork/join constructs, 36
Formal verification, 8, 12–15
  limitations of, 17
  preliminaries, 331–332
  Boolean functions. See Boolean operations
  finite-state automatons and languages. See
  Finite-state automatons and languages
  problems, 381–386
  sets and operations, 332–334
  summary, 380–381
  versus simulation-based verification, 15–17
Forward debugging, 301–302
Forward edges in depth-first searches, 375
Forward traversal of states, 498–500
Four-state constructs, synthesizability of, 57
Four-state representation, 188–189
Four-state simulators, 64–66, 108–109, 123–125
FPGA-based hardware simulators, 10, 62
  back ends for, 75
  characteristics of, 106–107
FPUs (floating point units), modules for, 35
Free variables
  in search-based algorithms, 436
  in symbolic simulation, 442–443
Frequencies, clock, 151–152
Front ends in simulators, 74
full_case directive, 29–31
Full-delay models, 117
Functional correctness rules, 26–27
  structural checks, 36–38
summary, 38
  syntactical checks. See Syntactical checks
Functional coverage, 260–261, 273–278
  in simulation-based verification, 10
  of specifications, 17
Functional dependency, 453
Functional representation in symbolic
  finite-state machines, 496
Functionality
  debugger, 312–315
  extracting, 217–222
  prioritizing, 222–223
Functionally equivalent functions, 387–388
Functions
  Boolean. See Boolean operations
  inlining, 62–63
  mapping, 334
  profiling statistics for, 123
Future events, 23, 82, 84

G
Gate arrays, instantiation of, 51
Gate netlists
  in abstraction levels, 44–45
  in hardware simulation, 62
Gated clocks, 41–42
Gates
  in bus operations, 37
  delays in, 41, 117–118, 120–121
Generalized cofactors
  for Boolean functions, 355–356
  for computational improvements, 527–529
get_init_value function, 146
Glitches, 110
  clock gating, 41–42
  domain-crossing, 43
  unit-delay simulators for, 110
Golden response, 167–169
  co-simulation with reference models, 174–178
  for self-checking codes, 169–174
Graph algorithms for finite-state automatons, 371–373
Gray code, assertions for, 241
Grouping
   equivalent states, 368–369
   points in verification, 16
   signals in assertions, 237

H
Hamming code, assertions for, 241
Hanging situations, 244–245
Hardware compilation, 108
Hardware Description Language (HDL), 25
code types of, 35
design files, 113–114
wrappers, 125–126
Hardware simulators and emulators, 77
coding guidelines for, 62–64
encapsulated models, 125–126
operation of, 106–108, 111–112
in system level verification, 215
Hardy, Godfrey H., 332
Hasse diagrams, 339
HDL (Hardware Description Language), 25
code types of, 35
design files, 113–114
wrappers, 125–126
Header files, 55
Height of binary decision diagrams, 405
Heuristics in binary decision diagrams, 407
Hierarchical paths and access, 54
   for port connections, 35
   in test bench interfaces, 195
Hierarchical signal tracing, 58–59
Hierarchical verification, 214–217
Hierarchy of assertions, 233
High impedance in two-state simulation, 109
High-level code, 75, 77
High-priority bugs, 217
Higher levels of abstraction, 44–46
$hold timing checks, 119
Host parameters in failure capture, 288
Hosts for simulators, 77–78, 111
Human error, 4
Hybrid simulators, 100–101

combinational loop compiling, 102–105
compiled event-driven, 101
distributed event management in, 105–106
hardware, 106–108
levelled event processing for zero-delay simulation, 101–102
Hybrid traversal method, 500

I
I/O ports
   in golden files, 168
   in test bench interfaces, 195
if-then-else statements, 29
$ifdef directives
   for assertions, 232
   for checking code, 172
   for dumping, 115–116, 163, 299
   for hierarchical signal tracing, 58
   for model activation, 53–54
   for verification environment configuration, 197–198
Images
   of functions, 334
   of present states, 498
Immediate assertions, 248–249
Impedance in two-state simulation, 109
Implementation
   vs. specifications, 2
   verification of, 4
IMPLEMENTATION_MODEL setting, 54
Implication graphs, 438–442
implication sequence operator, 255–256, 484
Implicit model checking, 494–495
Implicitly embedded sequential state, 28–32
Implied variables, 439–440
Inactive events, 22, 82
include files, 54
Incompletely specified Boolean functions, 349–350
Incompletely specified finite-state automatons, 361
Increment signals, assertions for, 242
Incremental compilation, 126–127
Indentation, 55
Independence, clock, 152–153
Indexed vectors, 158–159
Indirect measures, 320–321
Inertial delay, 121
infer routine, 436–438
Inferred latch phenomenon, 29
Infinite paths in fairness constraint, 482–483, 493
initial constructs
profiling statistics for, 123
in RTL initialization, 143
Initial states
in counter-examples, 501
in test plans, 218, 223
Initialization, 142–143
of cells, 54–55
of PLI, 145–147
for pseudorandom test generators, 230
of RTL, 143–145
in test harnesses, 141
at time zero, 147
Initializers in co-simulation models, 176
Input assignments in search-based algorithms, 434
Input constraints in symbolic simulation, 447–450
Input files for simulators, 113
Input oriented verification, 8
Input sequences in counter-examples, 501
Input stimuli in test harnesses, 141
Input symbols in finite-state automatons, 359
Inputs
initialization of, 143
subsets of, 212
in test plans, 218
$inset function, 258
Instantiations
gate array, 51
with initialization, 145
sequences, 259
Instruction-level accuracy, 175
Instructions
fetching, 76
memory for, 78
for stimulus generation, 161–162
Instrumentation for code coverage, 267–269
Intellectual property (IP) procurement, 111
Interactive mode
debugging in, 297
for simulations, 117, 127–128
Interconnect delays, 117
Interconnections in system level verification, 215
Interface processors, 108
Interfaces, minimizing, 51
Intermediate languages, 75, 77
Interpreted code, 75–77
Interpreted simulation systems, 78, 111
Interpreters for bus functional devices, 199–200
Interrupt handling, 78
intersect sequence operator, 254, 484
Intersections of sets, 333
Interval constraints for assertions, 242–243
Invariants in symbolic simulation, 447
Inversion attributes in edge-attributed BDDs, 413
IP (intellectual property) procurement, 111
Irredundancy conditions, 398
Isolation in debugging, 300
debugger functionality in, 312–315
design views in, 311–312
diagram tracing in, 302–304
forward and backward debugging in, 301–302
load, driver, and cone tracing in, 306–308
memory and array tracing in, 308–311
reference values, propagation, and bifurcation in, 300–301
time framing in, 304–306
Isomorphic subgraphs, 420, 424
ISS programs, 215
$isunknown function, 258
ITE operations, 399–404
Item coverage, 278–279
Iteration times in property checking, 14
J–L

Jitter and clock independence, 152–153
Joins for partially ordered sets, 340–342
Joint Test Action Group (JTAG) boundary scans, 224

Kernels, simulation, 78–79
Kripke structures
  for CTL formulas, 489–493
  for symbolic checking, 519
  for time, 470–474

L-automata, 507
Language containment
  in property checking, 487–489
  in symbolic computation, 507–513
Languages of finite-state automata, 362
Large tests, programmed code for, 161
Latch loops, coding, 37
Latches
  configuration coding for, 37–38
  gated clocks with, 42
Latency of bugs, 213
Lattices, 341
LDSTUB instruction, 217
Leaf nodes
  in binary decision diagrams, 389
  in binary moment diagrams, 423
Legal operations, pseudorandom generation of, 227
Lemmas in property checking, 14
Leveled event processing, 101–102
Leveling in cycle-based simulators, 88–92
Levels
  abstraction, 44–46
  alert, 186
  optimization, 115
Libraries
  initialization of, 145
  precompiled, 127
Linear temporal logic (LTL)
  in model checking, 475–477
  relative expressiveness of, 483–484
Linear time models, 474
Linked lists, 80–81
Linters
  designing and using, 66–67
  purpose of, 27
  in simulation-based verification, 9
Literals
  for Boolean functions, 343, 432–433
  in search-based algorithms, 437
Liveness properties, 466
Load tracing, 306–308
Loading memory, 186–190
Local canonicity in binary decision diagrams, 409
Local variables for sequence constructors, 257
lock_mutex function, 99
Logic circuitry in top-level modules, 35
Lookup tables, 107
Loops
  coding for, 36–37
  compiling, 102–105
  in cycle-based simulation, 59–60, 92–93, 102–105
  in event-driven simulators, 86–87
  in hardware simulation, 63
  problems with, 35–36
  for pseudorandom test generators, 230
  tracing, 59
  in tracing diagrams, 304
  zero time, 310
Low level optimization, 51–52
Lower-level constructs in RTL coding, 36
LTL (linear temporal logic)
  in model checking, 475–477
  relative expressiveness of, 483–484

M

Machine code, 77
Machine equivalence, 370–371
Machine minimization, 365–370
Macros
  for arrays, 54
  for directives, 198
SCC, 104–105
Maintenance
  of design, 315–318
  golden files for, 169
Makefiles, 113
Mapping
- arrays, 188–189
- CAM for, 190–194
- nodes in equivalence checking, 425–429
- between sets, 334
Market share costs from faulty design, 2
Mathematics in formal verification, 332
Maximum values in signal range
- assertions, 235
Maxterms for Boolean functions, 343, 346
MDDs (multivalue decision diagrams), 420
Mealy machines, 359, 362
Measures, coverage, 260–261, 320
Meets for partially ordered sets, 340–342
Membership in sets, 332
Memory
- checking contents of, 170
- dumping. See Dumping
- in equivalence checking, 428
- for formal verification, 16
- initializing, 142, 145, 195
- for instructions, 78
- loading, 186–190
- for pseudorandom test generators,
  229–230
- sparse and CAM, 190–194
- tracing, 308–311
- for vectors, 155
Memory arrays
- macro files for, 54
- as recognizable components, 47
- tracing, 58–59
Memory management units (MMUs), 35
Merge transformations, 402
Meshe for hardware simulators, 106
Methodologies
- formal method-based verification, 12–17
- simulation-based verification, 8–11,
  15–17
Midcycle PLI, 64
Minimal coverage by set elements, 339
Minimization algorithms, 365–370
Minimum support for Boolean functions,
  342
Minimum values in signal range
- assertions, 235
Minterms for Boolean functions, 343, 346
Mission-critical applications, 223
MMUs (memory management units), 35
Model checking, 4, 465–466
- assertions in, 484
- computation trees in, 474
- fairness constraint in, 482–483
- in formal method-based verification, 13
- problems, 531–537
- properties in, 466–467
- property checking. See Properties
- propositional temporal logic in,
  474–482
- specifications in, 466–468
- summary, 531
- symbolic computation in. See
  Symbolic computation
- temporal structure in, 468–474
- tools for, 529–530
Module-level assertions, 233
Module-level verification, 214, 216
Module port declarations, 55
Modules, 35
- dumping out nodes in, 164
- files in, 53
Monitor events, 23, 82
$monitor task, 23, 82, 165
$monitoroff task, 165
$monitoron task, 165
Monitors for parameter coverage, 272
Monotonic subsets, 513
Moore machines, 359, 362
Muller automata, 507
Multi-level representation for Boolean
functions, 343
Multiple clocks, 151, 258
Multiple events, 80–82
Multiple messages, linters for, 66–67
Multiple-output Boolean functions, 351
Multiple writes and reads in debugging,
  310–311
$multiplication task, 173
Multipliers, clock, 151–152
Multivalued decision diagrams (MDDs), 420
Mutual exclusion of bus enable signals, 37
Index

N
Naming conventions, 55
Narrow pulse filtering, 120–121
Native code, 75, 77
Netlists
  in abstraction levels, 44–45
  in hardware simulation, 62
Next-state functions, 48
  Boolean, 351
  in equivalence checking, 428, 430
NMOS constructs, 36
$nochange timing checks, 119
Nodes
  in binary decision diagrams, 388–395, 406
  in binary moment diagrams, 423
  expected values for, 301
  mapping, in equivalence checking, 425–429
  in shared BDDs, 412
  sharing, 406, 412
  in symbolic simulation, 447
  in tracing diagrams, 303
Nonblocking assignments
  in clock gating, 41
  for clock synchronization, 153–154
  in execution, 82
  in nondeterminism, 20–22
  processing order of, 22, 82–83
  for race problems, 40
  for transition delay, 149
Nonchronological backtracking, 437
Nondeterminism, 19–22
Nondeterministic finite-state automata, 361
Nonterminal nodes in binary moment diagrams, 423
NOR constructs in RTL coding, 36
Normal mode in test plans, 224–225
NOT constructs in RTL coding, 36

O
OBDDs (ordered binary decision diagrams), 392–395
Observability assertions for, 232
of bugs, 213
Odd parity in assertions, 236
OFDDs (ordered functional decision diagrams), 417–419
Off-line mode for self-checking codes, 170
OMI (open model interface), 125
On sets, 349–350
On-the-fly mode for self-checking codes, 170
One-cold signals, 237–238
One-hot signals, 237–238, 247
One-to-one functions, 334
$onehot function, 258
Open model interface (OMI), 125
Operands
  for properties, 467
  unequal width, 27–28
Optimization levels, 51–52, 115
OR gates and operators
  for Boolean functions, 343–344
  in clock gating, 41–42
  for sequence constructors, 254, 484
Ordered binary decision diagrams (OBDDs), 392–395
Ordered functional decision diagrams (OFDDs), 417–419
Ordered multivalued decision diagrams, 420
Ordering
  in cycle-based simulators, 89
  of execution in nondeterminism, 19
  nodes in binary decision diagrams, 392
  tests in code coverage, 270
  of variables, 32, 404–411
Ordering relations for sets, 338
Oscillation detection, 86–87
Output files for simulators, 113
Output oriented verification, 8
Outputs
  initialization of, 143
  in test plans, 218
Overconstraining, 14
Overflow in signal range assertions, 235
Overlapping conditional cases, 32–34
Overlapping tests, 319
OVL format, 118
Parallel-case directives, 33
Parameter coverage, 260, 270–272
Parameter sets, 228
Parameterization in symbolic simulation, 449, 453–457
Parent vertices in depth-first searches, 375
Parity in assertions, 236
Partially ordered sets, 339–342
Partitioned subcircuits, 107
Partitioners, 108
Partitions
set, 334–337
state, 369
for transition relations, 527
$pass_states_for_compare routine, 177
$past function, 258
Paths
in binary moment diagrams, 423
in breadth-first searches, 379
coverage of, 263
in CTL, 476–477
in decision diagrams, 416
in fairness constraints, 482–483, 493–494, 520–522
in graph algorithms, 372
in temporal logic, 475–478
in time models, 474
Performance
for code coverage, 269–270
of compilation, 78
of interpreted code, 76–77
optimization levels in, 51–52, 115
of simulation process. See Simulation of simulators, 78, 100, 110–112, 114–116
of zero-delay simulation, 87–88
$period timing checks, 119
Phase-drifting, 153
Phase lock loops (PLLs), 151, 154
PLIs (programming language interfaces), 51
initializing, 145–147
midcycle, 64
PLLs (phase lock loops), 151, 154
Pluses directives
for dumping, 58, 116, 163
for parameters, 127
PMOS constructs, 36
Points in verification, 16
Portability and maintainability, 52
centralized resources, 54–55
high-level code, 77
interpreted code, 76
project code layout, 52–54
RTL design file format, 55–56
rules for, 26
simulators, 112
summary, 56
Ports
connections to, 34–35
declaration of, 55
in golden files, 168
in test bench interfaces, 195
Posets, 339–342
Positive expansions, 417
Postprocessing mode
debugging in, 297
node value dumping in, 162
Postscan circuits in equivalence checking, 428
Power-on tests, 219
Pragmas for coverage tools, 263
Pre-images of present states, 498
Precompiled libraries, 127
Precompiled object code, 125
Predecessor variables in binary decision diagrams, 406
Predicate transformers, 515
Predicated events, 84
Preferred design constructs, 35–36
Preprocessing for memory contents, 170
Prerun methods, 174
Prescan circuits in equivalence checking, 428
Primitive gates, synthesizability of, 57
Priority
of bugs, 217, 296
of events, 82
of functionality, 222–223
Priority encoders, 33
Procedural assertions, 248
Index 577

Proceedural blocks, 18–19
Proceedural statements, 18, 21
Processor arrays
   in hardware simulators, 106–107
   in simulation-based verification, 10
Product finite-state machines, 363–365
Profiling for simulators, 122–123
Program construction for pseudorandom
test generators, 230
Programmed stimuli, 161–162
Programming language interfaces (PLIs),
   51
      initializing, 145–147
      midcycle, 64
Progress tracking in test plans, 223–226
Project code layout, 52–54
Propagaton
   of assignments in search-based
      algorithms, 437
   in debugging isolation, 300–301
   of events, 85–86
   gate delays from, 120–121
Properties
   checking, 4, 484
      in automata, 485–486
      CTL formulas, 489–493
      with fairness constraints, 493–494
      in formal method-based
         verification, 13–14
      language containment in, 487–489
   in model checking, 466–467
   in symbolic simulation, 446–447
   verifying, 142
Property automaton, 467
Propositional properties, 467
Propositional temporal logic
   CTL, 475–477
   LTL, 477–482
Pseudo Boolean functions, 420–422
Pseudorandom errors, 184
Pseudorandom test generators, 227–228
   program construction for, 230
   register and memory allocation for,
      229–230
   self-checking mechanisms for,
      230–231
   in simulation-based verification, 9
   user interface for, 228–229
PULLDOWN constructs, 36
PULLUP constructs, 36
Pulse filtering, narrow, 120–121
Pure literal rules
   in Boolean satisfiability, 432
   in search-based algorithms, 437
Q
   Quantification, early, 525–527
   Quest operators, 265
Queues
   circular, 239–240
   event, 80–81, 83, 86
Quotient sets, 335
R
   Race problems
      causes of, 39–40
      in nondeterminism, 20
      unit-delay simulators for, 110
   Random initialization routines, 144
   Random number generators, 184
   $random task, 144
   Random tests. See Pseudorandom test
generators
   Random vectors, 159
   Ranges of functions, 334
   READ command, 200–201
   Read operations
      cache units for, 225
      in debugging, 310–311
      race problems from, 39–40
      tracing, 117
   $readmemb task, 145, 186–187
   $readmemh task, 145, 186–187
   Reason parts in implication graphs, 440
   Recognizable components, 46–48
   Recomilation, incremental, 126
   Recovery timing checks, 119
   Reduced MDDs, 420
   Reduced OBDDs (ROBDDs), 12, 395,
      424–425
   Reduction operations
      for binary decision diagrams, 397–398
      for binary moment diagrams, 424
Redundancy, 5
  forms of, 7
  limitations of, 6
Redundant codes, 51
Redundant nodes, 402
Reed-Muller form
  for Boolean functions, 355
  for ordered functional decision diagrams, 417, 419
Reference clocks, 165, 181
reference_compare semaphore, 177
Reference models
  co-simulation with, 174–178
  verifying, 195
Reference outputs
  in simulation-based verification, 9
  in verification paradigm, 6
reference_resume semaphore, 177
Reference traces, 167
Reference values in debugging isolation, 300–301
Refinement process
  for descriptions, 2
  in property checking, 487
Registers
  disk, 219
  for pseudorandom test generators, 229–230
Regression suites, 319
Regression tests, 318–319
Relational representation in symbolic finite-state machines, 495
Relations in sets, 334–338
Relative expressiveness of CTL and LTL, 483–484
Release operations
  for code delivery, 319–320
  in LTL, 476
Release products, 320
repeat statements, 151
Report generators in linters, 67
Reproducible tests, 227
Repudiative assertions, 233
Resolved clauses in search-based algorithms, 435
Resolvent algorithm for Boolean satisfiability, 431–434
Respin costs from faulty design, 2
Response assessment, 162
  design state dumping, 162–167
  golden response, 167–169
  co-simulation with reference models, 174–178
  for self-checking codes, 169–174
  temporal specifications in, 178–183
  in test harnesses, 141
$\$restart task, 293–294
RESTORE command, 220
Restriction operations for binary decision diagrams, 398
Reverse ordering for binary decision diagrams, 397
Revision control, 11, 315–318
ROBDDs (reduced OBDDs), 12, 395, 424–425
Robin automata, 507
Roots
  in binary decision diagrams, 389
  of depth-first searches, 373
RTL_compare semaphore, 177
RTL models, 36
  clock generation in, 154–155
  in design, 311–312
  file format for, 55–56
  initialization, 143–145
  levelization in, 91
RTL_resume semaphore, 177
rtran construct, 124
Runtime operations
  batch mode arguments, 127
  data dumping selection, 299
  equivalence checking, 428
  simulator options, 113

S
Safety properties, 466
Sampling in dumping, 165
SAT (satisfiability) techniques, 430–431
  in formal method-based verification, 12
  search-based algorithms, 434–438
Satisfiability problem for Boolean functions, 346, 430–431
  implication graphs and learning in, 438–442
Index

resolvent algorithms for, 431–434
search-based algorithms for, 434–438
$save task, 293
Saving debug sessions, 292–294, 313
SBDDs (shared BDDs), 412–413
Scalar operations vs. vector, 48–51
SCCs (strongly connected components), 93
in directed graphs, 103–105
finding, 377–378
Scheduling
for event-driven simulators, 82–83, 87–88
nondeterminism in, 19–22
semantics, 22, 82–83
Schematic views, 311–312
Scope
circuit reduction, 289–290
of design profiling, 122–123
of design state dumping, 162–163
of errors and warnings, 186
test case reduction, 291–292
Scoreboarding, 8, 223
SDF (Standard Delay File) format, 118
Searches
for Boolean satisfiability, 434–438
breadth-first, 378–380
depth-first, 89–93, 373–378
for SCCs, 377–378
Sector registers, 219
Seeds for pseudorandom test generators, 227
select_branch routine, 436–438
Selection probability distribution, 229
Selective dumping, 58, 115–116
Self-checking routines
golden response for, 169–174
for pseudorandom test generators, 229–231
sem_post routine, 177
sem_wait routine, 177
Semaphores, 177
Semiformal verification, 8
Sensitivity lists
in procedural blocks, 18
variables on, 32
sequence keyword, 252
Sequences
for concurrent assertions, 259
connectives for, 253–258
constructors for, 250–253
coverage of, 267
for temporal behavior, 484
Sequential assertions, 238–241
Sequential elements and components
configuration coding for, 38
in cycle-based simulation, 61
functional equivalence of, 425
reset coding for, 38
Sequential specification, 467–468
Sequential states, implicitly embedded, 28–32
Sets, 332–334
partially ordered, 339–342
partitions of, 334–337
relations in, 334–338
$setup timing checks, 119
$setuphold timing checks, 119
Severity of alerts, 186
Shannon cofactor
for binary decision diagrams, 390
for Boolean functions, 354–355
for ordered functional decision diagrams, 417
Shannon theorem for Boolean functions, 353–355
Shared BDDs (SBDDs), 412–413
Shift register files, 238–239
Sifting algorithm, 409–410
Signals
and assertions, 194, 232
archives, 233
change patterns, 241–242
membership, 236–237
properties, 247
ranges, 235
in debugging, 115, 167
dumping, 164–165, 298–299
in golden files, 169
hierarchical, 58–59
for self-checking codes, 169
on sensitivity lists, 18
strength of, 36
Signature files, 126
Simulation
control units in, 74, 79
controlling, in cycle-based simulators, 96–100
data dumping in, 297–300
engines in, 74, 78–79
performance in, 78
higher levels of abstraction, 44–46
interface minimization, 51
low level/component level optimization, 51–52
rules for, 26
simulator recognizable components, 46–48
summary, 52
vector vs. scalar operations, 48–51
simulation-based verification, 8–11
assertions in, 232
vs. formal verification, 15–17
Simulation cycles, 22, 82
Simulation time in scope reduction, 289
Simulator consoles, 127–128
Simulator recognizable components, 46–48
Simulators, 73–79
cosimulation with encapsulated models, 125–126
cycle-based. See Cycle-based simulation
debugging, 114–116
design profiling for, 122–123
event-driven. See Event-driven simulators
file structure for, 113–114
hardware, 111–112
hybrid. See Hybrid simulators
incremental compilation, 126–127
interpreted vs. compiled, 111
operations and applications, 112–113
performance of, 78, 100, 110–112, 114–116
problems in, 130–136
simulator consoles, 127–128
summary, 129
taxonomy and comparison, 108–112
timing verification for, 116–122
two-state and four-state, 64–66, 108–109, 123–125
in verification, 9
zero vs. unit-delay, 109–110
Simultaneous read and writes, race problems from, 39–40
Size
binary decision diagrams, 405–406, 410–411, 416, 524
golden files, 169
$s$-skew timing checks, 119
Smooth operators, 357
Sparse memory modeling, 190–194
Spatial neighborhood in data dumping, 297–298
Special processors in hardware simulation, 62
 Specifications, 2
architectural, 217–222
assertions as, 232
for coverage, 273
errors in, 6, 17
functional coverage of, 17
in levels of abstraction, 44
in model checking, 466–468
sequential, 467–468
temporal, 178–183
for test plans, 217
Standard Delay File (SDF) format, 118
State diagrams
for finite-state automata, 359, 371–373
Kripke structures for, 470–474
State explosions, 365
State machine-based approach to functionality, 218
State space approach to test scenarios, 219–220
State transitions
in symbolic finite-state machines, 495
time in, 469–470
Statement coverage, 261
States
coverage of, 266–267
dumping, 162–167
equivalence of, 365–370
in finite-state automata, 359
initialization of, 142
mapping, 371
partitions of, 369
Index

saving, 292–294
in temporal logic, 475–478
traversing, 495–501
Static errors, 9
Static variable ordering in binary decision diagrams, 408
Status registers, 219
Stimuli
generation of, 155–159
asynchronous stimuli, 159–160
programmed stimuli, 161–162
in system level verification, 216
Street automata, 507
Strength constructs
in four-state constructs, 124
in RTL coding, 36
String acceptance, 359, 362
$strobe task, 23, 82
Strongly connected components (SCCs), 93
in directed graphs, 103–105
finding, 377–378
Structural checks, 36–38
Structural models, 53
Stuck-at faults, 356
Stylized comment assertions, 248
Subsets
of inputs, 212
of specifications, 466
synthesizable, 56–57
Subtraction of sets, 334
Successor variables in binary decision diagrams, 406
Sums of products, 342–343
Support variables for Boolean functions, 342
Swap operations, 408–409
SWIFT interface, 125
Symbolic computation, 494–495
counter-example generation in,
501–502
for CTL model checking
with fairness constraints, 520–523
fix-point computation in, 513–520
equivalence checking in, 502–506
finite-state machine representation
and state traversal, 495–501
language containment and fairness constraints in, 507–513
problems, 531–537
summary, 531
transitive closure by, 508–512
Symbolic simulation, 442–446
characteristic functions for, 450–453
input constraints in, 447–450
parameterization process in, 449,
453–457
problems, 458–464
summary, 457
verification in, 446–447
Symmetric Boolean functions, 346–348
Symmetric property for sets, 338
Symptoms in debugging process, 288
Synchronization
clock. See Clock generation and synchronization
clock domains, 43
in co-simulation models, 176
for hardware simulators, 107
Synchronous memory arrays, 47
Synchronous timing specifications,
178–180
Syntactical checks, 27
connection rules, 34–35
implicitly embedded sequential state,
28–32
operands of unequal width, 27–28
overlapping conditional cases, 32–34
preferred design constructs, 35–36
Synthesizability, 26–27, 57
System assertions
concurrent. See Concurrent assertions immediate, 248–249
System functions for concurrent assertions, 258
System level verification, 214–216
System tasks
in cycle-based simulators, 96
dumping, 166
T
t_tnodeinfo structure, 187–188
Tape-out criteria, 320–321
Target objectives in pseudorandom tests,
Tasks
- in cycle-based simulators, 96
- inlining, 62–63
- profiling statistics for, 123
- for self-checking codes, 174

Temporal behavior
- in assertions, 247, 250
- sequences for, 484

Temporal logic. See CTL (computation tree logic); LTL (linear temporal logic)

Temporal specifications in response assessment, 178–183

Temporal structure in model checking, 468–474

Temporal windows in data dumping, 298–299

Terminal nodes in binary decision diagrams, 389

Test benches
- anatomy of, 137–142
- bus functional model, 199–203
- clock generation and synchronization. See Clock generation and synchronization
- initialization mechanisms in, 142–147
- interface for, 195–196
- problems, 204–209
- response assessment. See Response assessment
- in simulation-based verification, 9
- stimulus generation for, 155–159
- asynchronous stimuli, 159–160
- programmed stimuli, 161–162
- summary, 204
- synthesizability of, 57
- verification environment
  - configuration in, 196–199
  - verification utility for. See Verification utility

Test cases, 142
- creating, 223
- reducing, 291–294

Test generators, pseudorandom. See Pseudorandom test generators

Test harnesses, 141

Test items in test plans, 223

Test mode in test plans, 224

Test patterns in verification paradigm, 6

Test plans, 8
- architectural specifications in, 217–222
- prioritizing functionality in, 222–223
- problems, 280–286
- progress tracking in, 223–226
- summary, 279
- test cases for, 223

Test scenarios, 211–213
- hierarchical verification in, 214–217
- pseudorandom test generators, 227–231
- in system level verification, 216
- test plans in. See Test plans

Testability units, 224–226

$test$plusargs directive
- for dumping, 116, 163, 299
- for parameters, 127
- tf_getp task, 173
- tf_nodeinfo task, 176, 187, 190
- tf_put task, 115, 176

Theorem-proving approach in property checking, 14

Threads for reference models, 177

3-SAT techniques, 431

throughout operator, 256, 484

Time. See also Synchronization; Temporal behavior
- in assertions, 243–245
- in data dumping, 298
- in debugging isolation, 304–306
- in event-driven simulators, 86–87

Time slots in a timing wheel, 82

Time stamps
- of errors and warnings, 186
- in golden files, 168

Time-to-market costs from faulty design, 2

Time windows for assertions, 242

Time zero
- clock transitions at, 150
- glitches at, 42–43
- initialization at, 147
- loop constructs at, 310

Timing checks
- disabling, 117
- profiling statistics for, 123
- for simulators, 116–122
Index

Timing correctness
- clock gating, 41–42
- domain-crossing glitches, 43
- race problems, 39–40
- rules for, 26–27
- summary, 43
- time zero glitches, 42–43
Timing properties, 119
Timing wheels, 80–82
Toggles
- for clock generation and synchronization, 148–149
- coverage of, 267
Tool compatibility, 26, 56–59
Top-down process, 44
Top-level functional blocks, 52
Top-level module components, 35, 53
Topological sorts, 89–93
Tori, 106
Total transition relations, 471
Traceability support, 115
Tracing
- array, 58–59, 308–311
- debugger functionality for, 312–313
- diagram, 302–304
- hierarchical signals, 58–59
- load, driver, and cone, 306–308
Tracks, disk, 219
TRAN constructs, 36, 124
Transactors for bus functional devices, 199–200
Transducers in finite-state automata, 359
Transformations for error correction code, 50–51
Transistor-level models, synthesizability of, 57
Transition functions and relations
- in early quantification, 526–527
- in finite-state automata, 359–360
- partitioning, 527
- in symbolic finite-state machines, 495–496
- total, 471
Transitions
- assertions for, 242
- coverage of, 267
- delays in, 149–150
- in timing specifications, 179–182
Transitive closure
- in finite-state automata, 372
- by symbolic computation, 508–512
Transport delay, 121
Traversing
- states, 495–501
- vertices, 373
Tree edges in depth-first searches, 375
Trees
- binary, 434–438
- clock
  - in cycle-based simulation, 60
  - processing, 95
- in depth-first searches, 374–375
- in model checking, 468–474
tri0 data type, 124
tri1 data type, 124
Trigger expressions in assertions, 234
trigger_function function, 99
TRIREG constructs
- in four-state constructs, 124
- in RTL coding, 36
Tristate devices, 37
Two-dimensional queues, 80–81
2-SAT techniques, 430–431
Two-state simulation, 64–66, 108–109, 117, 123–125

U
U operator
- in CTL, 477
- in LTL, 475
UDPs (user-defined primitives), 36
UIPs (unique implication points), 439–440
Unate variable rule, 432
Unbound properties, 446–447
Unclocked timing assertions, 243–245
Underconstraining, 14
Underflow in signal range assertions, 235
Union of sets, 333
Unique implication points (UIPs), 439–440
Unique tables for binary decision diagrams, 401
Unit clause rules
- in Boolean satisfiability, 433
- in search-based algorithms, 437
Unit-delay models, 117
Unit-delay simulators, 109–110
Unit level verification, 214, 216
Unit literals in Boolean satisfiability, 433
Universal operator, 357
Unknown values in assertions, 236
unlock_mutex function, 99
Unreferenced variables in performance, 51
Unrolled circuits in symbolic simulation, 445
Unsynthesizable constructs, 64
Update events, 83–85
Update of design, 315–318
User-defined mapping in equivalence checking, 426
User-defined nodes in tracing diagrams, 303
User-defined primitives (UDPs), 36
User errors in formal verification, 17
User interface for pseudorandom test generators, 228–229
User tasks
  in cycle-based simulators, 96
  for self-checking codes, 174
util_check_cache task, 183
util_dump_reg module, 183
$util_memory_loader task, 187
utility.v file, 183

V
Valid states in test plans, 218
Validation, event, 85
Value change dump (VCD) format, 116, 165–166
Variables
  assertion conditions for, 232
  in binary decision diagrams, 397, 404–411
  for Boolean functions, 342
  compare order of, 32
  flipping, 435
  in golden files, 168
  in implication graphs, 438–440
  initialization at time zero, 147
  interactive commands for, 128
  in Kripke structures, 471
  in race problems, 39
  in search-based algorithms, 436
  on sensitivity lists, 32
  for sequence constructors, 257
  in symbolic simulation, 442–443, 447
  unreferenced, 51
VCD (value change dump) format, 116, 165–166
Vector Boolean functions, 351–353
Vectors
  indexed, 158–159
  limitations of, 161
  memory for, 155
  random, 159
  vs. scalars, 48–51
  in verification, 8–9, 16
Verification
  coverage. See Coverage
  design updates and maintenance in, 315–318
  environment configuration for, 196–199
  formal. See Formal verification
  languages for, 5
  refinement, 487
  regression tests in, 318–319
  release mechanisms in, 319–320
  simulation-based, 8–11
    assertions in, 232
    vs. formal verification, 15–17
  summary, 321
  tape-out criteria in, 320–321
  transformation process, 5
Verification utility, 183–184
  assertion routines, 194
  error and warning alert mechanisms, 185–186
  error injectors, 184–185
  memory loading and dumping mechanisms, 186–190
  sparse memory and CAM, 190–194
Verilog/VHSIC Hardware Description Language (VHDL), 5
Version control, 315–318
Vertices
  for Boolean functions, 343
  in breadth-first searches, 378–380
  in depth-first searches, 373–378
  for finite-state automatons, 372
in implication graphs, 439–440
Views  
  in debugging isolation, 311–312  
  of profiling, 122  
  for verification environment, 197
Violations, detecting, 67
Virtual machines, 75
Visited infinitely states, 362
vpi_iterate construct, 115

W
Wait constructs, 36
Warnings  
  with implied sequential states, 32  
  linters for, 66–67  
  mechanisms for, 185–186
Waveform viewers, 165, 167
Waveform views in design, 311–313
Weight in binary moment diagrams, 424
Width  
  of binary decision diagrams, 405  
  of operands, 27–28
$width timing checks, 119
within operator, 256, 484
Word ranges, 188

Wrappers  
  for bus functional devices, 199–200  
  HDL, 125–126
Write operations  
  in debugging, 310–311  
  race problems from, 39–40  
  tracing, 117
Write signals, 201

X–Z
X operator  
  in CTL, 477  
  in LTL, 475
ZBDDs (zero-suppressed BDDs), 414–417
Zero-delay models, 116–117
Zero-delay simulation, 109–110  
  leveled event processing for, 101–102  
  performance of, 87–88
Zero-suppressed BDDs (ZBDDs), 414–417
Zero time  
  clock transitions at, 150  
  execution at, 19  
  glitches at, 42–43  
  loop constructs at, 310