# Appendix I
# Synthesizable and Non-Synthesizable Verilog Constructs

The list of synthesizable and non-synthesizable Verilog constructs is tabu-lated in the following Table

| Verilog Constructs | Used for | Synthesizable construct | Non-Synthesizable Construct |
|---|---|---|---|
| module | The code inside the module and the endmodule consists of the declarations and functionality of the design | Yes | No |
| Instantiation | If the module is synthesizable then the instantiation is also synthesizable | Yes | No |
| initial | Used in the test benches | No | Yes |
| always | Procedural block with the reg type assignment on LHS side. The block is sensitive to the events | Yes | No |
| assign | Continuous assignment with wire data type for modeling the combinational logic | Yes | No |
| primitives | UDP's are non-synthesizable whereas other Verilog primitives are synthesizable | Yes | No |
| force and release | These are used in test benches and non-synthesizable | No | Yes |
| delays | Used in the test benches and synthesis tool ignores the delays | No | Yes |
| fork and join | Used during simulation | No | Yes |
| ports | Used to indicate the direction, input, output and inout. The input is used at the top module | Yes | No |
| parameter | Used to make the design more generic | Yes | No |
| time | Not supported for the synthesis | No | Yes |

(continued)

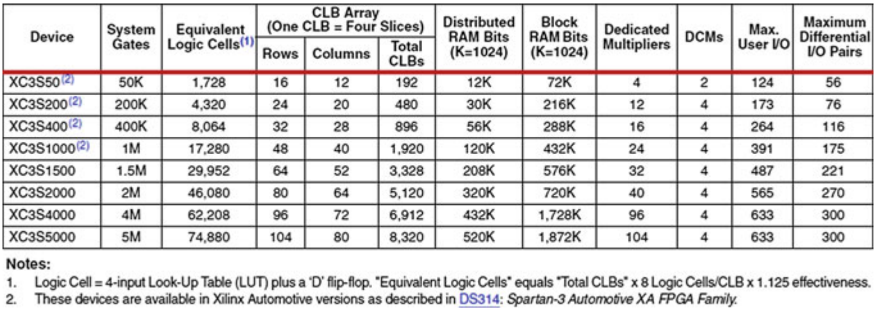
© Springer India 2016
V. Taraate, *Digital Logic Design Using Verilog*,
DOI 10.1007/978-81-322-2791-5

(continued)

| | | | |
|---|---|---|---|
| real | Not supported for synthesis | No | Yes |
| functions and task | Both are synthesizable. Provided that the task does not have the timing constructs | Yes | No |
| loop | The for loop is synthesizable and used for the multiple iterations. | Yes | No |
| Verilog Operators | Used for arithmetic, bitwise, unary, logical, relational etc are synthesizable | Yes | No |
| Blocking and non-blocking assignments | Used to describe the combinational and sequential design functionality respectively | Yes | No |
| if-else, case, casex, casez | These are used to describe the design functionality depending on the priority and parallel hardware requirements | Yes | No |
| Compiler directives (‘ifdef,‘undef, ‘define) | Used during synthesis | Yes | No |
| Bits and part select | It is synthesizable and used for the bit or part select | Yes | No |

# Appendix II
# Xilinx Spartan Devices

- Xilinx Spartan 3 Devices



| Device | System Gates | Equivalent Logic Cells[1] | CLB Array (One CLB = Four Slices) | | | Distributed RAM Bits (K=1024) | Block RAM Bits (K=1024) | Dedicated Multipliers | DCMs | Max. User I/O | Maximum Differential I/O Pairs |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Rows | Columns | Total CLBs | | | | | | |
| XC3S50[2] | 50K | 1,728 | 16 | 12 | 192 | 12K | 72K | 4 | 2 | 124 | 56 |
| XC3S200[2] | 200K | 4,320 | 24 | 20 | 480 | 30K | 216K | 12 | 4 | 173 | 76 |
| XC3S400[2] | 400K | 8,064 | 32 | 28 | 896 | 56K | 288K | 16 | 4 | 264 | 116 |
| XC3S1000[2] | 1M | 17,280 | 48 | 40 | 1,920 | 120K | 432K | 24 | 4 | 391 | 175 |
| XC3S1500 | 1.5M | 29,952 | 64 | 52 | 3,328 | 208K | 576K | 32 | 4 | 487 | 221 |
| XC3S2000 | 2M | 46,080 | 80 | 64 | 5,120 | 320K | 720K | 40 | 4 | 565 | 270 |
| XC3S4000 | 4M | 62,208 | 96 | 72 | 6,912 | 432K | 1,728K | 96 | 4 | 633 | 300 |
| XC3S5000 | 5M | 74,880 | 104 | 80 | 8,320 | 520K | 1,872K | 104 | 4 | 633 | 300 |

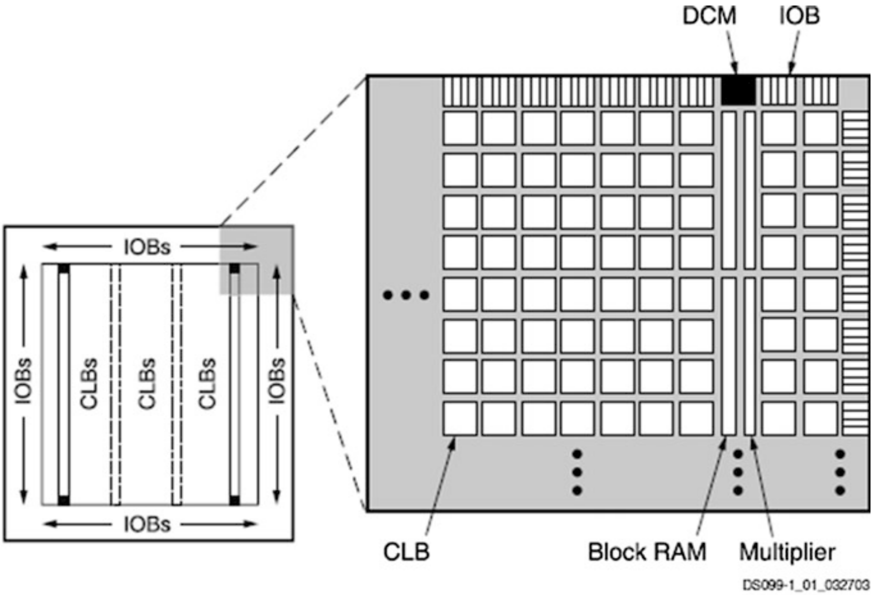
**Notes:**

1. Logic Cell = 4-input Look-Up Table (LUT) plus a 'D' flip-flop. "Equivalent Logic Cells" equals "Total CLBs" x 8 Logic Cells/CLB x 1.125 effectiveness.
2. These devices are available in Xilinx Automotive versions as described in DS314: *Spartan-3 Automotive XA FPGA Family.*

© Springer India 2016

V. Taraate, *Digital Logic Design Using Verilog*,
DOI 10.1007/978-81-322-2791-5

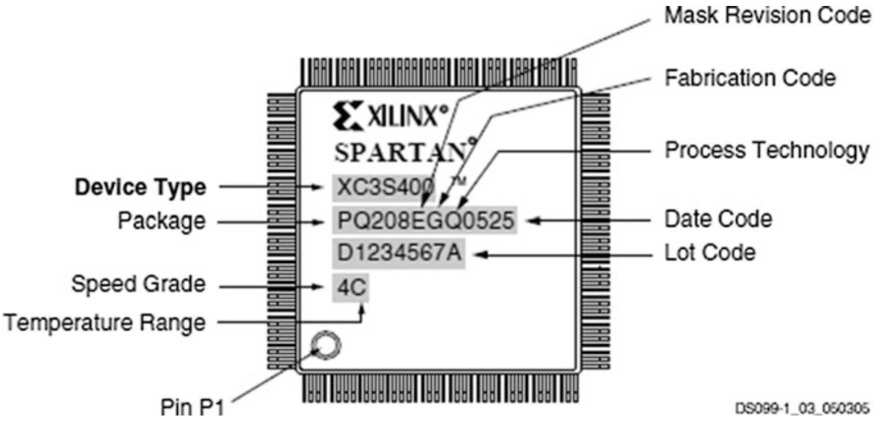
- Spartan 3 Family Architecture

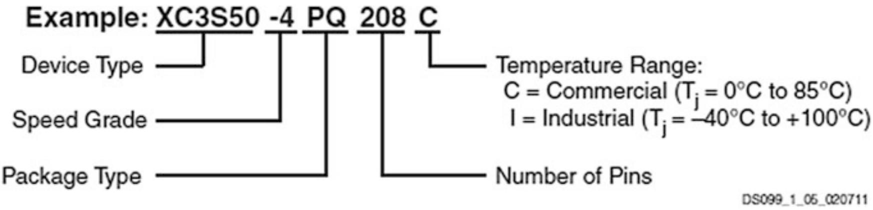


**Notes:**

1. The two additional block RAM columns of the XC3S4000 and XC3S5000 devices are shown with dashed lines. The XC3S50 has only the block RAM column on the far left.

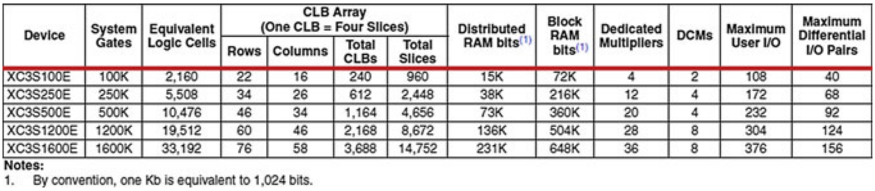
- Xilinx Spartan 3 Package information for Part no XC3S400-4PQ208C




Example: XC3S50 -4 PQ 208 C
Device Type
Speed Grade
Package Type
Temperature Range:
C = Commercial ($T_j$ = 0°C to 85°C)
I = Industrial ($T_j$ = –40°C to +100°C)
Number of Pins
DS099_1_05_020711

For more information please use the following link http://www.xilinx.com/support/documentation/data_sheets/ds099.pdf.

- Xilinx FPGA Spartan 3E Devices

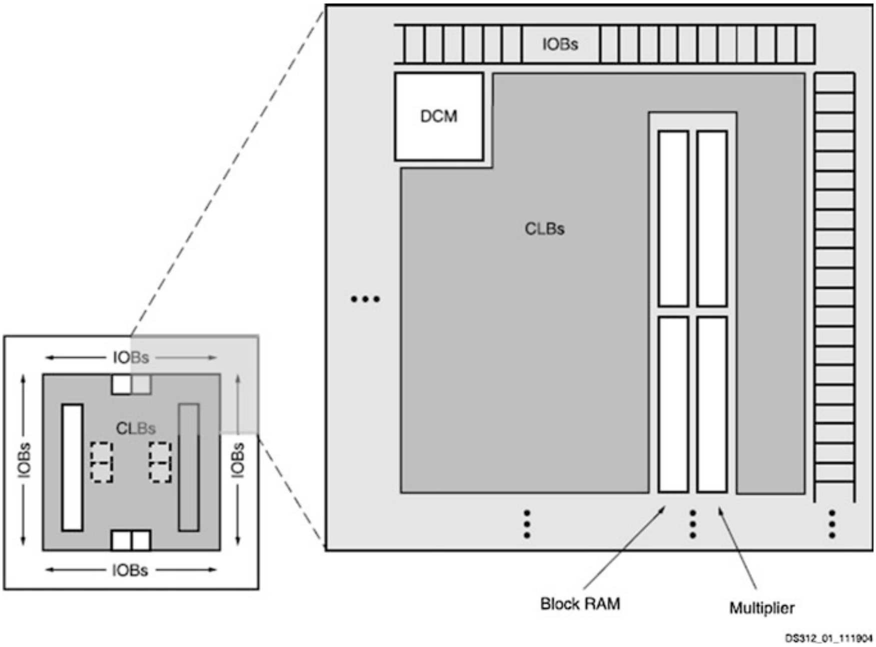


| Device | System Gates | Equivalent Logic Cells | CLB Array (One CLB = Four Slices) | | | | Distributed RAM bits[1] | Block RAM bits[1] | Dedicated Multipliers | DCMs | Maximum User I/O | Maximum Differential I/O Pairs |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Rows | Columns | Total CLBs | Total Slices | | | | | | |
| XC3S100E | 100K | 2,160 | 22 | 16 | 240 | 960 | 15K | 72K | 4 | 2 | 108 | 40 |
| XC3S250E | 250K | 5,508 | 34 | 26 | 612 | 2,448 | 38K | 216K | 12 | 4 | 172 | 68 |
| XC3S500E | 500K | 10,476 | 46 | 34 | 1,164 | 4,656 | 73K | 360K | 20 | 4 | 232 | 92 |
| XC3S1200E | 1200K | 19,512 | 60 | 46 | 2,168 | 8,672 | 136K | 504K | 28 | 8 | 304 | 124 |
| XC3S1600E | 1600K | 33,192 | 76 | 58 | 3,688 | 14,752 | 231K | 648K | 36 | 8 | 376 | 156 |

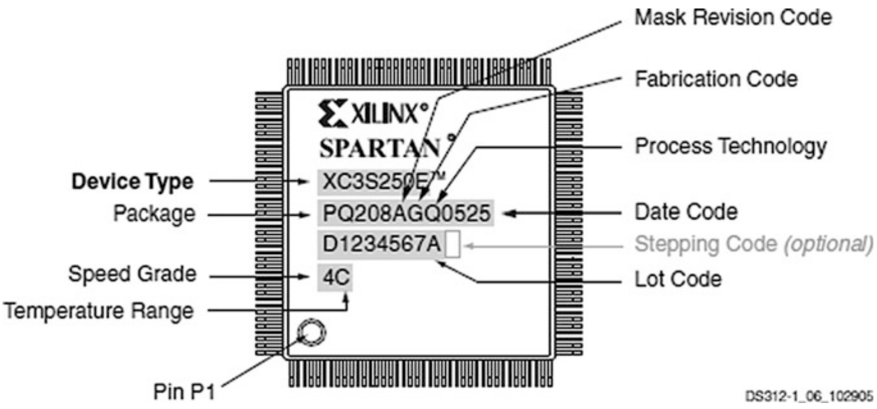
**Notes:**

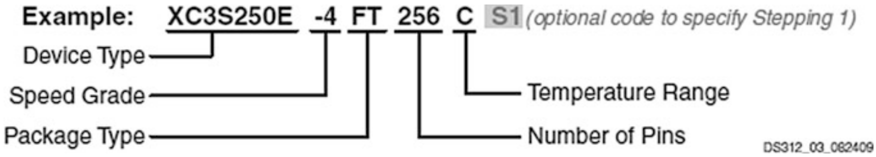
[1] By convention, one Kb is equivalent to 1,024 bits.

- Xilinx Spartan 3E Architecture



- Xilinx Spartan 3E package information





For more information please use the following link http://www.xilinx.com/support/documentation/data_sheets/ds312.pdf.

# Appendix III
# Design For Testability

**The Design For Testability (DFT) and its necessity is discussed in summarized**

In the practical ASIC design, the DFT is used to find out various kinds of faults in the design. For FPGA designs this step is excluded. The necessity of DFT is for early detection of the faults in the design using scan chain insertions. The functional abstraction of defects is called as fault and the abstraction of the fault is the system level error. Physical testing is carried out after manufacturing of chip to understand the fabrication-related issues or faults.

The defects in the design can be physical or electrical. Physical defects are due to silicon or defective oxide. Electrical defects are short, open, transistor defects and changes in the threshold voltage.

Few of the faults in the design are following

1. Stuck at faults: Stuck at one or Stuck at zero
2. Memory faults or pattern-sensitive faults
3. Bridging faults
4. Cross point faults
5. Delay faults

Testing process is the process of test pattern generation, test pattern application and output evaluation.

Generally, the test flow includes the following:

1. Identify the target faults
2. Test generation
3. Fault Simulation
4. Testability
5. DFT

- Design For Testability (DFT)
  The DFT is required to reduce the defect level in the design. Consider the following design; in this design it is not possible to give the test input so design is not testable. The DFT uses the concept of controllability and observabilty. The key steps are

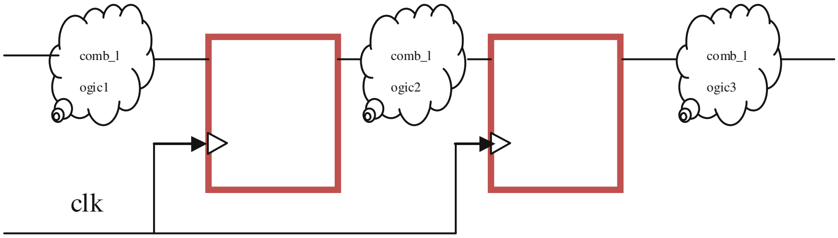
© Springer India 2016
V. Taraate, *Digital Logic Design Using Verilog*,
DOI 10.1007/978-81-322-2791-5

1. RTL design
2. Simulation
3. Synthesis
4. Insert scan chain
5. Layout

If every data input of the register need to be forced to the known value during the test, then the design is controllable.

Observability indicates the ability to observe the node at primary output. The de-sign needs to be controllable and observable.

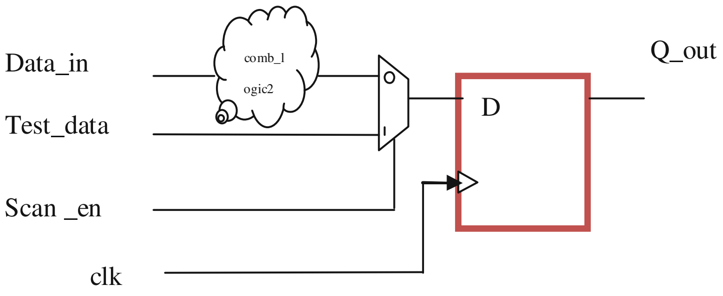
As shown in the following design, the design input of comb_logic1 is controllable and the output from comb_logic3 is observable. But comb_logic1 and comb_logic2 are not observable. So for detection of faults, it is essential to make comb_logic1, comb_logic2, and comb_logic3 controllable as well as observable.



- The basic DFT techniques are: Ad-HOC DFT and Structured DFT. The structured DFT includes the scan-based DFT which is again classified as MUX -based DFT and level–sensitive, element-based DFT. An-other structured DFT technique is MBIST and LBIST. JTAG is used for boundary scan.

Basic MUX-based technique is described below.

- MUX-based scan cell
  The MUX-based scan cell is shown below and it has additional inputs as Test_data, Scan_en. The MUX is inserted at the input of the D flip-flop and during testing Scan_en=1 the D input is Test_data. During normal operation, the Scan_en=0 and Data_in can pass through the combinational logic to the D input. Thus, the following cell works both in the test and normal modes. The clk can be scan_clk during the test mode.



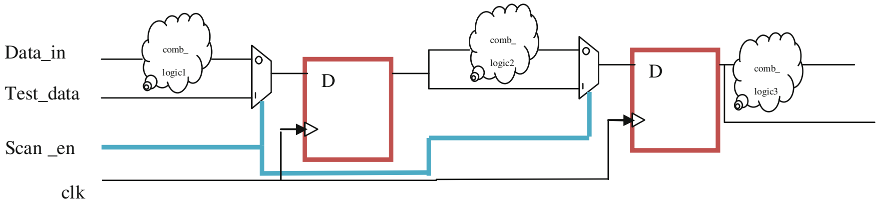
- MUX-based scan chain:

  Normally used method is insertion of scan by using MUX logic. MUX-based scan cell shown in the above figure is used to replace the sequential elements from the design. Depending on the requirements the design team decides whether to use partial scan method or full scan method.

  In the partial scan method few of the sequential elements are replaced by the MUX based scan cell. In the full scan method, all the sequential elements are re-placed by the MUX-based scan cell.

  Due to scan insertion, the area and timing of the design has significant impact. Scan insertion increases the area of the design and due to added MUX-based logic even it affects on the timing of the design. The following example shows the scan chain using MUX-based scan cells.

  Most of the time, the partial scan is recommended if area and timing is the constraint but this reduces overall fault coverage. If full scan is used then it increases area and has significant impact on timing but this improves overall fault coverage.



- Scan Design rules
  Following are few of the scan design rules need to be considered:

1. Generated clocks in the design: There should not be generated clocks in the design as they are not controllable
2. Combinational feedback loop: There should not be any combinational loop in the design as it creates issues in the timing analysis and hence it is essential to break the combinational loop
3. Gated clocks: Gated clocks need to be avoided as they are not controllable
4. Asynchronous Control signals: There should not be any internally generated asynchronous control signals
5. Do not mix the positive and negative edge triggered flip-flops
6. Avoid use of latches in the design
7. If shift registers are used then do not replace them by using scan enabled flip-flops but only ensure the enable control
8. Do not use the clock input as data
9. Bypass the memories during DFT

# Index

**Symbols**
#0delay assignments, 80
$display, 229
$finish, 229
$monitor, 80, 229
$strobe, 80
1 line to 2 or (1:2) decoder, 61
14 nanometer (nm), 261
100 K gates, 300
2 line to 4 or (2:4) decoder, 66
2:1 MUX, 54
20 nanometer (nm), 261
4 line to 16 or (4:16) decoder, 67
4:1 MUX, 57
40 nanometer (nm), 261
50% duty cycle, 266

**A**
Acknowledgment or notification, 329
Active, 80
Active event queue, 80
Active power, 364
Adders, 38
Addition, 179
AHB, 386
Algorithms, 386
Altera, 242
ALU, 392
ALU architecture, 172
always, 11, 222, 369
Analyze, 264
AND, 173
AND logic, 32
Antifuse, 234
Arbiters, 172, 391
Architecture, 3, 237, 259, 289
Area, 369
Area minimization, 30
Area optimization, 246
Area utilization, 98
Arithmetic logic unit (ALU), 172
Arithmetic operations, 11, 39
ASIC chip designs, 2
ASIC design, 197
ASIC library, 389
ASIC porting, 383
Asynchronous, 322
Asynchronous counters, 138
Asynchronous design, 249
Asynchronous path, 295
Asynchronous pulse generator, 169, 248
Asynchronous reset, 109, 156, 157
Asynchronous reset ‘reset_n’, 200
Attribute, 308

**B**
Barrel shifter, 192
Basic cell or base cell, 257
Baud rate control block., 396
Begin-end, 150
Behavior, 9
Behavioral, 7
Bidirectional, 10
Bidirectional IO, 395
Bidirectional shift register, 130
Binary counter, 114
Binary encoding, 244
Binary-to-Gray, 49, 342
Bit-stream, 238
Bitwise operations, 16
Blocking (=), 10, 80, 217
Blocking assignments, 146, 243
Block level, 389
Bluetooth, 382
Boolean algebra, 2
Bottom-up, 3, 300
BRAM, 235
Buses, 397

© Springer India 2016
V. Taraate, *Digital Logic Design Using Verilog*,
DOI 10.1007/978-81-322-2791-5

**C**
Cadence RTL Compiler, 259
Capture flip-flop, 282
case, 57, 369
case construct, 92
case-endcase, 57, 93
Case equality, 97
Case inequality, 97
Cell library, 263
Characterize, 312
Check_design, 265, 318
Check_timing, 318
Checker, 187, 228
Chip level, 389
CLB, 232, 235
clk initalization, 229
clk generator, 229
Clock balancing, 370
Clock buffer, 370
Clock definitions, 286
Clock domain, 161
Clock domain crossing (CDC), 250, 322
Clocked-based logic, 104
Clocked logic, 192
Clock gating, 251, 363, 364, 366, 390
Clock gating structure, 161
Clocking boundary, 341
Clock path group, 285
Clock skew, 242, 387
Clock to ‘q’ delay, 280
Clock tree, 363, 364, 387
Clock tree synthesis, 260
CMOS, 360
CMOS logic, 2
Code converters, 49
Coding guidelines, 79
Combinational logic, 10, 27
Combinational loop, 245
Combinational path, 286
Combinational path group, 284
Combinational shifters, 192
Comparators, 46
Compile, 267
Compile-characterize, 302
Compiler, 314
Computational blocks, 386
Concentration and replication, 18
Concurrent, 10
Concurrent execution, 161
Conditional assignments, 55
Configuration data, 239
Consolidated control signal, 334
Constant folding, 272
Constants, 10
Constraints, 260
Continuous assignment, 82, 92
Control and timing unit, 392
Control path, 165, 392
Control signals, 325
Coverage goals, 388
CPLD, 230
CPU, 256
create_clock, 265, 287
Cumulative delay, 160
Current simulation time, 80
Current_state, 198
Cycle accurate, 187
Cycles, 390
Cycle stealing, 154

**D**
Data arrival time, 278, 281
Database, 263
Data buffers, 396
Data integrity, 322
Data path, 280, 364, 392
Data path synchronizer, 340
Data propagation, 390
Data rate, 258
Data required time, 281
DCM, 235, 242
DDR, 241
DDR II, 382
DDR III, 382
Dead zone code, 273
Debug, 303
Decoder, 58
Decrement, 179
default, 93, 217, 369
Defining hierarchy, 390
Delay operators, 185
Deserializer, 391
Design compiler, 263
Design constraints, 5, 300
Design environment, 302
Design implementation, 252
Design object, 264, 301
Design partitioning, 274, 306, 346, 386
Design performance, 79, 162, 163, 386
Design rule constraints, 300
Design rule library, 261
Design rules, 302
Design specification, 257
DesignWare, 262
Device utilization summary, 239
DFT, 259, 389
DFT friendly RTL, 390
Differential IOs, 387

Different phases, 324
DLL, 239, 242
DMA controller, 391
DRC violations, 315
Drivers, 228
Drive strength, 302
DSP, 192, 382
DSP algorithms, 389
DSP blocks, 235
DSP filtering, 389
DUV, 226
Dynamic, 278
Dynamic power, 251, 360
Dynamic voltage and frequency scaling, 365

**E**
EDA, 238, 387
EDA tool, 4, 300, 364
Edge triggered, 107
EDIF, 238
Efficient synthesis, 215
Effort level, 252
Efforts levels, 267
Eight-bit parameterized counter, 130
Elaborate, 264
Empty and full flag, 341
Enable, 67, 104
Encoder, 68
Encoding, 174, 291
Encounter from Cadence., 261
End point, 284
Equality operators, 11
Ethernet, 382
Even parity, 187
Events, 390
Exclusive OR, 34

**F**
Fabrication techniques, 258
False path, 295, 327
Fast debugging, 215
Faults, 263
Feasibility study, 385
FFT, 386
FIFO, 386, 396
FIFO memory buffer, 338, 345
FIR, 386
Flash memory, 233
Flip-flop, 103, 107
Floor planning, 260
Four as to one MUX, 56
FPD, 230
FPGA, 230, 382
FPGA designs, 228
FPGA prototyping, 382
Frequency synthesis, 242
FSM, 197, 244, 314
FSM coding, 200
FSM control, 340
Full adder, 39, 98, 181
Full-case, 174
Full-case statement, 91
Full-custom, 256
Full subtractors, 41
Functional and timing proven., 386
Functional design specifications, 3
Functionality, 257
Functional model, 3
Functional simulation, 238
Functions, 172, 184

**G**
Gated clock, 161
Gate level netlist, 3, 260, 278, 303
GDSII, 360
GDSII file, 261
General purpose, 387
Geometric, 260
Glitches, 111, 157, 199
Glitches or hazards, 343
Glitch free, 216, 244
Global clock buffers, 247
Glue logic, 274, 306
Gray counter, 123, 210, 342
Gray encoding, 341
Gray-to-binary, 50, 341
group, 307
group_path, 310

**H**
Half adder, 38
Half subtractor, 41
Handshaking, 338
Handshaking mechanism, 391
Handshaking signals, 329
Hazards, 111, 390
HDL, 370
Hierarchical design, 302
Hierarchies, 290
High impedance, 11
High speed, 257
High-speed interfaces, 241
High speed IOs, 387
Hold, 160

Hold time, 260
Hold time violations, 278, 346
Hold violation, 280, 315

**I**
I2C, 382
IC Compiler from Synopsys, 261
IEEE 1364-2005, 80
IEEE 1801, 371
if-else, 57, 369
if-else construct, 92
IIR, 386
Inactive, 80
Increment, 179
Incremental compilation, 309
initial, 11, 220, 222
Input and output delay, 267
Input argument, 185
Input register path group, 284
Input string, 185
Input to reg path, 284
Instantiation, 8
Instruction register and decoder logic, 392
Inter-assignment delays, 222
Interconnect, 232
Interface, 258
Internally generated clock, 157
Intra-assignment delays, 224
IO blocks (IOB), 235, 238
IO high performance standards, 235
IO interfaces, 387
IPs, 258, 382
Isolation, 371
Isolation cell, 365, 371, 390
Isolation control, 371

**J**
Johnson counter, 127

**L**
Latch-based designs, 163
Latches, 103, 217
Late arrival, 332
Late arrival signal, 291
Latency, 249, 297, 327, 340, 360
Launch flip-flop, 282
Layout, 257
Leakage current, 360
Legal converging, 329
Level shifter, 371, 374, 390
Level synchronizers, 325
Level-to-pulse, 329
Level triggered, 154
Libraries, 5, 302
Library models, 366
Link library, 263
Linting tool, 250
Load, 302
Logical equality, 97
Logical flattening, 309
Logical inequality, 97
Logical operators, 11
Logic capacity, 232
Logic cells, 238
Logic density, 232
Logic duplication, 246, 291
Logic gates, 27
Longer runtime, 305
Loss of correlation, 347
LUTs, 239

**M**
Macrocells, 238
Macros, 258
Map, 262
Map_effort, 309
Master mode, 233
Master-slave flip-flops, 368
Maximum area, 257
Maximum operating frequency, 280
Mealy, 198
Mealy level to pulse, 200
Memory storage element, 107
Metastability, 248, 324
Micro-architecture, 3, 172, 259, 289, 300, 388
Microprocessors, 256
Minimization techniques, 27
Minimum bus width, 369
The min, max corner analysis, 287
Min or max, 267
Missing ‘else’ clause, 95
Modeling levels, 390
Monitor, 80, 228
Moore, Gordon, 2, 198
Multibit adders and subtractors, 44
Multi-bit signals, 338
Multi-Cycle Path (MCP), 295, 331, 343
Multi phase clock, 165
Multiple “always” block, 150
Multiple $V_{th}$, 364
Multiple clock domain, 163
Multiple clocks, 274
Multiple control signals, 332
Multiple Driver Assignment, 102
Multiple driver error, 102
Multiple drivers, 245
Multiple power domain, 382
Multiplex decoding, 390

Multiplex encoding, 292
Multiplexers, 53, 98, 367
Multiplier, 235, 386
MUX, 53
MUX Synchronizer, 331

**N**
NAND logic, 33
NBA, 80
NBA queue, 88, 150
Negative clock skew, 283
Negative edge, 104
Negative level sensitive D latch, 106
Nested if-else, 244
Netlist, 238
Nets, 308
Next_state, 198
Next state logic, 199
Nonblocking, 10, 217, 243
Non-blocking (<=), 80
Non-blocking assignments, 80, 150, 189
Nonconverging, 328
Nonsynthesizable, 219
Non-synthesizable constructs, 11
NOR logic, 28

**O**
Odd parity, 187
One-hot encoding, 212, 244
Opcode, 172
Operand, 181
Operand Isolations, 364
Optimization, 5, 184
Optimization constraints, 300
Optimize, 262
Optimized netlist, 263
OR, 173
OR logic, 28
Oscillatory behavior, 85
Output register path, 284
Output to reg path, 284

**P**
Packaging, 258
Packets, 386
Parallel and multiplexing logic, 173
Parallel execution, 386
Parallel input parallel output logic., 132
Parallelism, 391
Parallel logic, 57, 157
Parameterized binary and gray counter, 123
Parameters, 10
Parasitic (RC), 260
Parasitic capacitance, 361
Parentheses, 273
Parity checker, 189
Parity detectors, 48
Parity generator, 187
Partitioning of design, 172
Path groups, 315
Performance, 232
Performance constraints, 257
Performance improvement, 177
Phase Shifting, 242
Photo lithography, 261
Physical verification, 261
Pipelined design, 391
Pipelined processor, 389
Pipelined register, 163
Pipelined stage, 313
Pipelining, 154, 161, 177, 290, 390
PIPO registers, 132
PLA, 230
Place and route, 252, 389
PLL, 242, 382
Port interfaces, 306
Ports, 308
Positive clock skew, 282
Positive edge, 104
Positive level sensitive latch, 104
Positive slack, 290
The post-synthesis verification, 246
Power, 184, 238, 259
Power compiler, 366
Power domains, 371
Power gating, 365
Power management, 363
Power planning, 260
Power rails, 371
Power Shut-Off (PSO), 365
Power state tables, 371
Power switches, 371
Pre layout STA, 260
Prime Time, 278
Priorities, 308
Priority encoders, 69
The priority encoding, 292
Priority logic, 57, 92, 157
Procedural assignments, 83
Procedural Block “always”, 146
Process, 302
Processing algorithms, 389
Process node, 360
Processor, 392
Program and stack pointer, 392
Program Language Interface (PLI), 11
Programmable interconnects, 238
Programmable logic devices (PLD), 230, 382

Programmable Switch, 232
PROM, 230
Propagation delay, 280, 313
Protocol, 172, 386, 391
Prototype, 230, 382
Pulse stretcher, 329
Pulse synchronizers, 330

**R**
Race around conditions, 87
Race condition, 85
RAM, 140
Random test, 388
Read, 264
Readability, 79, 198
Read empty logic, 352
Read synchronization, 352
Reduction operators, 19
References, 308
Reg, 83
Register balancing, 290, 291, 313, 390
Register duplications, 290
Register retiming, 390
Registered inputs and outputs, 174
Register logic, 199
Register output logic, 160
Register-to-register path, 174
Register-to-register timing path, 313
Reg to reg path, 284
Relational operator, 18
Report_constraints, 315
Report_constraints_all, 315
Report_timing, 310
Reset deassertion, 109, 346
Reset recovery, 157
Reset synchronizer, 346
Reset tree, 156
Resource sharing, 47, 98, 174, 246
Retention, 371
Retention cell, 373, 390
Retention control, 373
Reusability, 79
Ring counter, 125, 226
Ripple counter, 140
Robust verification, 390
ROM, 140
Round robin, 393
RTL, 3, 259, 306
RTL code, 9
RTL design, 284, 358
RTL synthesis, 3

**S**
SATA, 382
Scan insertions, 263
SDC, 287, 295, 327
Sdc commands, 300
SDF, 246
Search_path, 263
Semiconductor, 383
Semi-custom, 256
Sensitivity, 10
Sensitivity list, 81, 245
Sequence detector, 212
Sequential, 10
Sequential logic, 10, 103
Serial data, 396
Serializer, 391
Set_clock_latency, 287
Set_clock_uncertainty, 287
Set_dont_touch, 305
Set_input_delay, 287, 288
Set_output_delay, 287
Set and reset, 370
Set-don't_touch_network, 369
Setup, 160
Setup time, 278
Setup violation, 310
Shift operators, 20
Shift register, 127, 130, 146, 240
Short-circuit power, 362
Sign operands, 13
Silicon wafer, 257
Simulation, 220
Simulation and synthesis mismatch., 90
Simulator, 5, 390
Single port RAM, 241
Skew, 266, 334
Slack, 279, 310
Slave mode, 233
Slowest path in the design, 154
SOC, 172, 359, 382
SOC components, 391
SOC design, 228
SOC validation, 387
SPI, 382
SPLD, 230
SRAM, 233
SRAM memory cell, 233
SRPG, 366
STA, 238, 278
Standard cell, 256, 360
State encoding, 215

State machines, 274
State register, 199
State transition, 200
State transition table, 210
Static, 278, 397
Stratified event queue, 80
Stratified event queuing, 146
Stray capacitance, 361
Structural, 7
Structural design, 7
Structured ASIC, 232
Stuck at fault coverage, 389
Subroutine, 184
Subtraction, 179
Subtractors, 41
Switches, 371
Switching activity, 362
Switching power, 161
Switch level design, 3
Symbol_library, 263
Synchronization, 156, 391
Synchronization failure, 324
Synchronized asynchronous resets, 250
Synchronizers, 274
Synchronous, 114, 198
Synchronous design, 114, 248, 286
Synchronous reset, 111, 157
Synopsys, 367
Synopsys_dc.setup, 263
Synopsys DC, 300
Synopsys Design Compiler, 259
Synopsys PT, 278, 300
Synthesis, 199, 238, 259, 387
Synthesizable, 7
Synthesizable constructs, 392
Synthesizable RTL, 383
Synthesized logic, 200
Synthesizer, 259
System C, 388
System Verilog, 388

**T**
Target library, 262
Task, 172, 184
Technology constraints, 258
Technology library, 262
Temperature, 302
Testability, 260
Testbench, 5, 219, 390
Test plan, 388
Test vectors, 388
Three-bit down counter, 118
Three-bit up counter, 114
Three-bit up-down counter, 120
Three-procedural block FSM, 200
Throughput, 360, 386
Time borrowing, 154
Time budgeting, 154
Time control, 185
Time to market, 257
Timing analysis, 198, 277, 303
Timing analyzer, 286
Timing closure, 278
Timing goals, 286
Timing issues, 247
The timing optimization, 308
Timing or area, 238
Timing parameters, 287
Timing paths, 260
Timing performance, 238
Timing proven IPs, 383
Timing sequence, 327
Timing summary, 315
Timing violation, 160, 174, 260, 280
Toggle flip-flop, 200
Toggle synchronizer, 330
Too tight constraints, 278
Top-down, 3, 300
Top level, 274
Transactions, 390
Transistor, 2
Transitions in the state diagram, 217
Translate, 262
Transport and inertial, 390
Tri-state, 37, 102, 240
Two-level synchronizer, 250
Two stage level synchronizer, 109, 157

**U**
UART, 382, 396
Unified Power Format (UPF), 370, 389
Unintended latches, 88
Unintentional combinational loops, 85
Universal logic, 35, 55
Unknown, 11
USB, 382
User constraints, 315

**V**
Vendor-specific power formats, 370
Verification, 259, 343
Verification methodologies, 5
Verification planning, 388
Verilog, 2
Verilog IEEE standards, 6
Verilog RTL, 3
Video decoders, 386
Virtex, 239

Virtual clock, 266
Voltage, 302
Voltage level, 374

**W**
Weight factor, 310
Wire, 82
Working directory, 263
Write command, 267
Write full logic, 352
Write synchronization, 352

**X**
Xilinx, 242
XILINX Spartan, 239
XNOR, 34
XOR, 34, 173
XOR logic, 55