

Contents

Preface	xxi
In the Classroom	xxv
Acknowledgments	xxvii
Contributors	xxix
About the Editors	xxxiii

CHAPTER 1 Introduction 1

*Charles E. Stroud, Lang-Terng (L.-T.) Wang, and
Yao-Wen Chang*

1.1 Overview of electronic design automation	2
1.1.1 Historical perspective	2
1.1.2 VLSI design flow and typical EDA flow	4
1.1.3 Typical EDA implementation examples	9
1.1.4 Problems and challenges	12
1.2 Logic design automation	13
1.2.1 Modeling	13
1.2.2 Design verification	14
1.2.3 Logic synthesis	17
1.3 Test automation	18
1.3.1 Fault models	19
1.3.2 Design for testability	21
1.3.3 Fault simulation and test generation	23
1.3.4 Manufacturing test	24
1.4 Physical design automation	25
1.4.1 Floorplanning	27
1.4.2 Placement	27
1.4.3 Routing	28
1.4.4 Synthesis of clock and power/ground networks	29
1.5 Concluding remarks	32
1.6 Exercises	33
Acknowledgments	35
References	35

CHAPTER 2 Fundamentals of CMOS design 39

Xingbao Chen and Nur A. Touba

2.1	Introduction	39
2.2	Integrated circuit technology	40
2.2.1	MOS transistor	41
2.2.2	Transistor equivalency	44
2.2.3	Wire and interconnect	46
2.2.4	Noise margin	48
2.3	CMOS logic	49
2.3.1	CMOS inverter and analysis	49
2.3.2	Design of CMOS logic gates and circuit blocks	52
2.3.3	Design of latches and flip-flops	55
2.3.4	Optimization techniques for high performance	57
2.4	Integrated circuit design techniques	58
2.4.1	Transmission-gate/pass-transistor logic	59
2.4.2	Differential CMOS logic	61
2.4.3	Dynamic pre-charge logic	62
2.4.4	Domino logic	63
2.4.5	No-race logic	67
2.4.6	Single-phase logic	70
2.5	CMOS physical design	71
2.5.1	Layout design rules	72
2.5.2	Stick diagram	75
2.5.3	Layout design	79
2.6	Low-power circuit design techniques	84
2.6.1	Clock-gating	85
2.6.2	Power-gating	85
2.6.3	Substrate biasing	87
2.6.4	Dynamic voltage and frequency scaling	88
2.6.5	Low-power cache memory design	89
2.7	Concluding remarks	92
2.8	Exercises	92
	Acknowledgments	95
	References	95

CHAPTER 3 Design for testability 97

Laung-Terng (L.-T.) Wang

3.1	Introduction	98
3.2	Testability analysis	100

3.2.1	SCOAP testability analysis	101
3.2.1.1	Combinational controllability and observability calculation.	102
3.2.1.2	Sequential controllability and observability calculation	103
3.2.2	Probability-based testability analysis	105
3.2.3	Simulation-based testability analysis	108
3.3	Scan design	109
3.3.1	Scan architectures	109
3.3.1.1	Muxed-D scan design	109
3.3.1.2	Clocked-scan design.	111
3.3.1.3	LSSD scan design.	113
3.3.2	At-speed testing	114
3.4	Logic built-in self-test	118
3.4.1	Test pattern generation	119
3.4.1.1	Exhaustive testing	121
3.4.1.2	Pseudo-random testing.	121
3.4.1.3	Pseudo-exhaustive testing.	125
3.4.2	Output response analysis.	129
3.4.2.1	Ones count testing	130
3.4.2.2	Transition count testing.	131
3.4.2.3	Signature analysis	131
3.4.3	Logic BIST architectures	135
3.4.3.1	Self-testing with MISR and parallel SRSG (STUMPS).	135
3.4.3.2	Built-in logic block observer (BILBO)	136
3.4.3.3	Concurrent built-in logic block observer (CBILBO).	138
3.4.4	Industry practices	138
3.5	Test Compression	139
3.5.1	Circuits for test stimulus compression	141
3.5.1.1	Linear-decompression-based schemes	141
3.5.1.2	Broadcast-scan-based schemes	145
3.5.2	Circuits for test response compaction.	149
3.5.2.1	Combinational compaction.	152
3.5.2.2	Sequential compaction.	156
3.5.3	Industry practices	159
3.6	Concluding remarks	161
3.7	Exercises.	162
	Acknowledgments.	165
	References	165

*Chung-Yang (Ric) Huang, Chao-Yue Lai, and
Kwang-Ting (Tim) Cheng*

4.1	Introduction	173
4.2	Computational complexity	175
4.2.1	Asymptotic notations.	177
4.2.1.1	O-notation	178
4.2.1.2	Ω -notation and Θ -notation	179
4.2.2	Complexity classes	180
4.2.2.1	Decision problems versus optimization problems	180
4.2.2.2	The complexity classes P versus NP	181
4.2.2.3	The complexity class NP-complete	182
4.2.2.4	The complexity class NP-hard	184
4.3	Graph algorithms	185
4.3.1	Terminology	185
4.3.2	Data structures for representations of graphs	187
4.3.3	Breadth-first search and depth-first search.	188
4.3.3.1	Breadth-first search	188
4.3.3.2	Depth-first search	190
4.3.4	Topological sort	192
4.3.5	Strongly connected component	193
4.3.6	Shortest and longest path algorithms	195
4.3.6.1	Initialization and relaxation	195
4.3.6.2	Shortest path algorithms on directed acyclic graphs	196
4.3.6.3	Dijkstra's algorithm	196
4.3.6.4	The Bellman-Ford algorithm	199
4.3.6.5	The longest-path problem	200
4.3.7	Minimum spanning tree.	200
4.3.8	Maximum flow and minimum cut	202
4.3.8.1	Flow networks and the maximum-flow problem	202
4.3.8.2	Augmenting paths and residual networks.	203
4.3.8.3	The Ford-Fulkerson method and the Edmonds-Karp algorithm	204
4.3.8.4	Cuts and the max-flow min-cut theorem	205
4.3.8.5	Multiple sources and sinks and maximum bipartite matching	207

4.4	Heuristic algorithms	208
4.4.1	Greedy algorithm	209
4.4.1.1	Greedy-choice property	210
4.4.1.2	Optimal substructure	211
4.4.2	Dynamic programming	211
4.4.2.1	Overlapping subproblems	213
4.4.2.2	Optimal substructure	214
4.4.2.3	Memoization	214
4.4.3	Branch-and-bound	215
4.4.4	Simulated annealing	217
4.4.5	Genetic algorithms	219
4.5	Mathematical programming	221
4.5.1	Categories of mathematical programming problems	221
4.5.2	Linear programming (LP) problem	222
4.5.3	Integer linear programming (ILP) problem	223
4.5.3.1	Linear programming relaxation and branch-and-bound procedure	224
4.5.3.2	Cutting plane algorithm	225
4.5.4	Convex optimization problem	226
4.5.4.1	Interior-point method	227
4.6	Concluding remarks	230
4.7	Exercises	230
	Acknowledgments	232
	References	232

CHAPTER 5 Electronic system-level design and high-level synthesis 235

Jianwen Zbu and Nikil Dutt

5.1	Introduction	236
5.1.1	ESL design methodology	236
5.1.2	Function-based ESL methodology	239
5.1.3	Architecture-based ESL methodology	241
5.1.4	Function architecture codesign methodology	243
5.1.5	High-level synthesis within an ESL design methodology	244
5.2	Fundamentals of High-level synthesis	246
5.2.1	TinyC as an example for behavioral descriptions	250
5.2.2	Intermediate representation in TinyIR	251
5.2.3	RTL representation in TinyRTL	253

5.2.4	Structured hardware description in FSM D	254
5.2.5	Quality metrics	257
5.3	High-level synthesis algorithm overview	261
5.4	Scheduling.	263
5.4.1	Dependency test.	263
5.4.2	Unconstrained scheduling	266
5.4.3	Resource-constrained scheduling	268
5.5	Register binding.	273
5.5.1	Liveness analysis	273
5.5.2	Register binding by coloring	277
5.6	Functional unit binding	281
5.7	Concluding remarks	289
5.8	Exercises	293
	Acknowledgments.	294
	References	294
CHAPTER 6	Logic synthesis in a nutshell	299
	<i>Jie-Hong (Roland) Jiang and Srinivas Devadas</i>	
6.1	Introduction	299
6.2	Data Structures for Boolean representation and reasoning	302
6.2.1	Quantifier-free and quantified Boolean formulas	303
6.2.2	Boolean function manipulation	308
6.2.3	Boolean function representation	309
6.2.3.1	Truth table	309
6.2.3.2	SOP	310
6.2.3.3	POS	311
6.2.3.4	BDD	312
6.2.3.5	AIG	321
6.2.3.6	Boolean network	323
6.2.4	Boolean representation conversion.	324
6.2.4.1	CNF <i>vs.</i> DNF.	324
6.2.4.2	Boolean formula <i>vs.</i> circuit.	326
6.2.4.3	BDD <i>vs.</i> Boolean network	326
6.2.5	Isomorphism between sets and characteristic functions	328
6.2.6	Boolean reasoning engines.	331
6.3	Combinational logic minimization	332
6.3.1	Two-level logic minimization	332

6.3.1.1	PLA implementation <i>vs.</i> SOP minimization	333
6.3.1.2	Terminology	334
6.3.2	SOP minimization	336
6.3.2.1	The Quine-McCluskey method	336
6.3.2.2	Other methods	340
6.3.3	Multilevel logic minimization	340
6.3.3.1	Logic transformations.	341
6.3.3.2	Division and common divisors	344
6.3.3.3	Algebraic division	344
6.3.3.4	Common divisors	350
6.3.3.5	Boolean division	356
6.3.4	Combinational complete flexibility.	357
6.3.5	Advanced subjects.	361
6.4	Technology mapping	362
6.4.1	Technology libraries	363
6.4.2	Graph covering.	365
6.4.3	Choice of atomic pattern set	366
6.4.4	Tree covering approximation.	367
6.4.5	Optimal tree covering	369
6.4.6	Improvement by inverter-pair insertion	370
6.4.7	Extension to non-tree patterns.	370
6.4.8	Advanced subjects.	371
6.5	Timing analysis	371
6.5.1	Topological timing analysis	374
6.5.2	Functional timing analysis	376
6.5.2.1	Delay models and modes of operation.	377
6.5.2.2	True floating mode delay	380
6.5.3	Advanced subjects.	383
6.6	Timing optimization	384
6.6.1	Technology-independent timing optimization	384
6.6.2	Timing-driven technology mapping	386
6.6.2.1	Delay optimization using tree covering	386
6.6.2.2	Area minimization under delay constraints	390
6.6.3	Advanced subjects.	391
6.7	Concluding remarks	392
6.8	Exercises	393
	Acknowledgments.	400
	References	400

*Laung-Terng (L.-T.) Wang, Xiaoqing Wen, and
Sbianling Wu*

7.1	Introduction	406
7.2	Scan design	408
7.2.1	Scan design rules	408
7.2.1.1	Tristate buses	408
7.2.1.2	Bidirectional I/O ports	409
7.2.1.3	Gated clocks	411
7.2.1.4	Derived clocks	412
7.2.1.5	Combinational feedback loops	412
7.2.1.6	Asynchronous set/reset signals	413
7.2.2	Scan design flow.	414
7.2.2.1	Scan design rule checking and repair	415
7.2.2.2	Scan synthesis	417
7.2.2.3	Scan extraction	422
7.2.2.4	Scan verification	422
7.3	Logic built-in self-test (BIST) design.	425
7.3.1	BIST design rules	425
7.3.1.1	Unknown source blocking	426
7.3.1.2	Re-timing	430
7.3.2	BIST design example	430
7.3.2.1	BIST rule checking and violation repair.	431
7.3.2.2	Logic BIST system design	431
7.3.2.3	RTL BIST synthesis	437
7.3.2.4	Design verification and fault coverage enhancement	438
7.4	RTL Design for testability	438
7.4.1	RTL scan design rule checking and repair.	440
7.4.2	RTL scan synthesis	441
7.4.3	RTL scan extraction and scan verification	442
7.5	Concluding remarks	443
7.6	Exercises	443
	Acknowledgments.	446
	References	446

Jiun-Lang Huang, Cheng-Kok Kob, and Stephen F. Cauley

8.1	Introduction	450
8.1.1	Logic simulation	451
8.1.2	Hardware-accelerated logic simulation	452
8.1.3	Circuit simulation	452
8.2	Logic simulation models	453
8.2.1	Logic symbols and operations	453
8.2.1.1	“1” and “0”	453
8.2.1.2	The unknown value u	453
8.2.1.3	The high-impedance state Z	453
8.2.1.4	Basic logic operations	454
8.2.2	Timing models	455
8.2.2.1	Transport delay	455
8.2.2.2	Inertial delay	456
8.2.2.3	Functional element delay model	457
8.2.2.4	Wire delay	457
8.3	Logic simulation techniques	459
8.3.1	Compiled-code simulation	460
8.3.1.1	Preprocessing	460
8.3.1.2	Code generation	461
8.3.1.3	Applications	462
8.3.2	Event-driven simulation	462
8.3.2.1	Zero-delay event-driven simulation	462
8.3.2.2	Nominal-delay event-driven simulation	463
8.4	Hardware-accelerated logic simulation	465
8.4.1	Types of hardware acceleration	467
8.4.2	Reconfigurable computing units	468
8.4.3	Interconnection architectures	470
8.4.3.1	Direct interconnection	470
8.4.3.2	Indirect interconnect	471
8.4.3.3	Time-multiplexed interconnect	472
8.4.4	Timing issues	474
8.5	Circuit simulation models	475
8.5.1	Ideal voltage and current sources	476
8.5.2	Resistors, capacitors, and inductors	476
8.5.3	Kirchhoff’s voltage and current laws	477
8.5.4	Modified nodal analysis	477

8.6	Numerical methods for transient analysis.	480
8.6.1	Approximation methods and numerical integration	480
8.6.2	Initial value problems	483
8.7	Simulation of VLSI interconnects.	485
8.7.1	Wire resistance	486
8.7.2	Wire capacitance	487
8.7.3	Wire inductance	489
8.7.4	Lumped and distributed models	491
8.7.5	Simulation procedure for interconnects	491
8.8	Simulation of nonlinear devices.	495
8.8.1	The diode.	496
8.8.2	The field-effect transistor.	498
8.8.3	Simulation procedure for nonlinear devices	502
8.9	Concluding remarks	504
8.10	Exercises.	506
	Acknowledgments.	509
	References	510
CHAPTER 9	Functional verification	513
	<i>Hung-Pin (Charles) Wen, Li-C. Wang, and Kwang-Ting (Tim) Cheng</i>	
9.1	Introduction	514
9.2	Verification hierarchy	515
9.2.1	Designer-level verification	517
9.2.2	Unit-level verification	518
9.2.3	Core-level verification	518
9.2.4	Chip-level verification	519
9.2.5	System/board-level verification	520
9.3	Measuring verification quality	520
9.3.1	Random testing.	520
9.3.2	Coverage-driven verification	522
9.3.3	Structural coverage metrics	524
9.3.3.1	Line coverage (<i>a.k.a.</i> statement coverage).	524
9.3.3.2	Toggle coverage.	524
9.3.3.3	Branch/path coverage	525
9.3.3.4	Expression coverage	526
9.3.3.5	Trigger coverage (<i>a.k.a.</i> event coverage).	528
9.3.3.6	Finite state machine (FSM) coverage	529
9.3.3.7	More on structural coverage.	530
9.3.4	Functional coverage metrics.	531

9.4	Simulation-based approach	532
9.4.1	Testbench and simulation environment development	533
9.4.2	Methods of observation points	535
9.4.3	Assertion-based verification	537
9.4.3.1	Assertion coverage and classification	538
9.4.3.2	Use of assertions	539
9.4.3.3	Writing assertions	540
9.5	Formal approaches	540
9.5.1	Equivalence checking	541
9.5.1.1	Checking based on functional equivalence	543
9.5.1.2	Checking based on structural search	543
9.5.2	Model checking (property checking)	547
9.5.2.1	Model checking with temporal logic	553
9.5.3	Theorem proving	556
9.6	Advanced research	561
9.7	Concluding remarks	563
9.8	Exercises	564
	Acknowledgments	570
	References	570
CHAPTER 10	Floorplanning	575
	<i>Tung-Chieh Chen and Yao-Wen Chang</i>	
10.1	Introduction	575
10.1.1	Floorplanning basics	575
10.1.2	Problem statement	577
10.1.3	Floorplanning model	577
10.1.3.1	Slicing floorplans	577
10.1.3.2	Non-slicing floorplans	578
10.1.4	Floorplanning cost	579
10.2	Simulated annealing approach	580
10.2.1	Simulated annealing basics	581
10.2.2	Normalized Polish expression for slicing floorplans	583
10.2.2.1	Solution space	585
10.2.2.2	Neighborhood structure	586
10.2.2.3	Cost function	588
10.2.2.4	Annealing schedule	590
10.2.3	B*-tree for compacted floorplans	593
10.2.3.1	From a floorplan to its B*-tree	594

10.2.3.2	From a B*-tree to its floorplan	594
10.2.3.3	Solution space	598
10.2.3.4	Neighborhood structure	598
10.2.3.5	Cost function	600
10.2.3.6	Annealing schedule	600
10.2.4	Sequence pair for general floorplans	600
10.2.4.1	From a floorplan to its sequence pair	600
10.2.4.2	From a sequence pair to its floorplan	601
10.2.4.3	Solution space	604
10.2.4.4	Neighborhood structure	604
10.2.4.5	Cost function	605
10.2.4.6	Annealing schedule	605
10.2.5	Floorplan representation comparison	605
10.3	Analytical approach	607
10.4	Modern floorplanning considerations	612
10.4.1	Soft modules	612
10.4.2	Fixed-outline constraint	615
10.4.3	Floorplanning for large-scale circuits	617
10.4.4	Other considerations and topics	622
10.5	Concluding remarks	625
10.6	Exercises	625
	Acknowledgments	631
	References	631
CHAPTER 11	Placement	635
	<i>Chris Chu</i>	
11.1	Introduction	635
11.2	Problem formulations	637
11.2.1	Placement for different design styles	637
11.2.1.1	Standard-cell placement	637
11.2.1.2	Gate array/FPGA placement	637
11.2.1.3	Macro block placement	637
11.2.1.4	Mixed-size placement	638
11.2.2	Placement objectives	638
11.2.2.1	Total wirelength	638
11.2.2.2	Routability	639
11.2.2.3	Performance	640
11.2.2.4	Power	640
11.2.2.5	Heat distribution	640
11.2.3	A common placement formulation	641

11.3	Global placement: partitioning-based approach	641
11.3.1	Basics for partitioning	642
11.3.1.1	Problem formulation.	642
11.3.1.2	The Fiduccia-Mattheyses algorithm	643
11.3.1.3	A multilevel scheme	645
11.3.2	Placement by partitioning	646
11.3.2.1	The basic idea	646
11.3.2.2	Terminal propagation technique	647
11.3.3	Practical implementations	648
11.3.3.1	The Capo algorithm	648
11.3.3.2	The Fengshui algorithm	649
11.4	Global placement: simulated annealing approach	649
11.4.1	The placement algorithm in TimberWolf	650
11.4.1.1	Stage 1	650
11.4.1.2	Stage 2	651
11.4.1.3	Annealing schedule	651
11.4.2	The Dragon placement algorithm	652
11.5	Global placement: analytical approach.	653
11.5.1	An exact formulation	653
11.5.2	Quadratic techniques.	655
11.5.2.1	Quadratic wirelength	655
11.5.2.2	Force interpretation of quadratic wirelength	658
11.5.2.3	Net models for multi-pin nets	659
11.5.2.4	Linearization methods.	661
11.5.2.5	Handling nonoverlapping constraints.	664
11.5.3	Nonquadratic techniques	668
11.5.3.1	Log-sum-exponential wirelength function.	669
11.5.3.2	Density constraint smoothing by bell-shaped function	670
11.5.3.3	Density constraint smoothing by inverse laplace transformation	672
11.5.3.4	Algorithms for nonlinear programs	672
11.5.4	Extension to multilevel	673
11.5.4.1	First choice	673
11.5.4.2	Best choice	674
11.6	Legalization	674
11.7	Detailed placement	675
11.7.1	The Domino algorithm.	675
11.7.2	The FastDP algorithm.	677

11.8	Concluding Remarks679
11.9	Exercises680
	Acknowledgments682
	References.682
CHAPTER 12	Global and detailed routing	687
	<i>Huang-Yu Chen and Yao-Wen Chang</i>	
12.1	Introduction688
12.2	Problem definition689
	12.2.1 Routing model.689
	12.2.2 Routing constraints691
12.3	General-purpose routing.692
	12.3.1 Maze routing.693
	12.3.1.1 Coding scheme694
	12.3.1.2 Search algorithm694
	12.3.1.3 Search space695
	12.3.2 Line-search routing695
	12.3.3 A*-search routing697
12.4	Global routing697
	12.4.1 Sequential global routing697
	12.4.2 Concurrent global routing699
	12.4.3 Steiner trees700
12.5	Detailed Routing704
	12.5.1 Channel routing704
	12.5.2 Full-chip routing710
12.6	Modern routing considerations715
	12.6.1 Routing for signal integrity.716
	12.6.1.1 Crosstalk modeling.716
	12.6.1.2 Crosstalk-aware routing.718
	12.6.2 Routing for manufacturability720
	12.6.2.1 OPC-aware routing721
	12.6.2.2 CMP-aware routing725
	12.6.3 Routing for reliability729
	12.6.3.1 Antenna-avoidance routing731
	12.6.3.2 Redundant-via aware routing.736
12.7	Concluding remarks738
12.8	Exercises740
	Acknowledgments.745
	References745

CHAPTER 13 Synthesis of clock and power/ground networks 751

Cheng-Kok Koh, Jitesh Jain, and Stephen F. Cauley

13.1	Introduction	751
13.2	Design considerations.	753
13.2.1	Timing constraints.	753
13.2.2	Skew and Jitter	755
13.2.3	<i>IR</i> drop and <i>L·di/dt</i> noise	760
13.2.4	Power dissipation	761
13.2.5	Electromigration	762
13.3	Clock Network design	763
13.3.1	Typical clock topologies.	763
13.3.2	Clock network modeling and analysis	770
13.3.3	Clock tree synthesis.	774
13.3.3.1	Clock skew scheduling	775
13.3.3.2	Clock tree routing	779
13.3.3.3	Zero-skew routing	781
13.3.3.4	Bounded-skew routing	793
13.3.3.5	Useful-skew routing	807
13.3.4	Clock tree optimization	811
13.3.4.1	Buffer insertion in clock routing	811
13.3.4.2	Clock gating.	816
13.3.4.3	Wire sizing for clock nets	819
13.3.4.4	Cross-link insertion.	826
13.4	Power/ground network design	829
13.4.1	Typical power/ground topologies	829
13.4.2	Power/ground network analysis	833
13.4.3	Power/ground network synthesis	836
13.4.3.1	Topology optimization	837
13.4.3.2	Power pad assignment	837
13.4.3.3	Wire width optimization	838
13.4.3.4	Decoupling capacitance	839
13.5	Concluding remarks	843
13.6	Exercises	843
	Acknowledgments.	846
	References	846

James C.-M. Li and Michael S. Hsiao

14.1	Introduction	851
14.2	Fault Collapsing	854
14.2.1	Equivalence fault collapsing	854
14.2.2	Dominance fault collapsing	858
14.3	Fault Simulation	861
14.3.1	Serial fault simulation	861
14.3.2	Parallel fault simulation	863
14.3.2.1	Parallel fault simulation	864
14.3.2.2	Parallel pattern fault simulation	866
14.3.3	Concurrent fault simulation	868
14.3.4	Differential fault simulation	871
14.3.5	Comparison of fault simulation techniques	874
14.4	Test Generation	876
14.4.1	Random test generation	876
14.4.1.1	Exhaustive testing	879
14.4.2	Theoretical Background: Boolean difference	880
14.4.2.1	Untestable Faults	881
14.4.3	Designing a stuck-at ATPG for combinational circuits	882
14.4.3.1	A naive ATPG algorithm	882
14.4.3.2	A basic ATPG algorithm	886
14.4.3.3	D algorithm	890
14.4.4	PODEM	895
14.4.5	FAN	900
14.5	Advanced Test Generation	902
14.5.1	Sequential ATPG: Time frame expansion	902
14.5.2	Delay fault ATPG	905
14.5.3	Bridging fault ATPG	908
14.6	Concluding Remarks	909
14.7	Exercises	910
	Acknowledgments	913
	References	913
	Index	919