

CONTENTS

Preface	<i>page</i> xv
Acknowledgments	xx

Part I Introduction

1 The digital abstraction	3
1.1 Digital signals	3
1.2 Digital signals tolerate noise	5
1.3 Digital signals represent complex data	9
1.3.1 Representing the day of the year	10
1.3.2 Representing subtractive colors	11
1.4 Digital logic functions	12
1.5 Verilog description of digital circuits and systems	14
1.6 Digital logic in systems	15
Summary	16
Bibliographic notes	17
Exercises	17
2 The practice of digital system design	21
2.1 The design process	21
2.1.1 Specification	22
2.1.2 Concept development and feasibility	23
2.1.3 Partitioning and detailed design	26
2.1.4 Verification	26
2.2 Digital systems are built from chips and boards	27
2.3 Computer-aided design tools	32
2.4 Moore's law and digital system evolution	33
Summary	35
Bibliographic notes	36
Exercises	36

Part II Combinational logic

3	Boolean algebra	41
3.1	Axioms	41
3.2	Properties	42
3.3	Dual functions	44
3.4	Normal form	45
3.5	From equations to gates	47
3.6	Boolean expressions in Verilog	49
	Summary	51
	Bibliographic notes	52
	Exercises	52
4	CMOS logic circuits	55
4.1	Switch logic	55
4.2	Switch model of MOS transistors	59
4.3	CMOS gate circuits	66
4.3.1	Basic CMOS gate circuit	66
4.3.2	Inverters, NANDs, and NORs	67
4.3.3	Complex gates	69
4.3.4	Tri-state circuits	72
4.3.5	Circuits to avoid	74
	Summary	75
	Bibliographic notes	75
	Exercises	76
5	Delay and power of CMOS circuits	79
5.1	Delay of static CMOS gates	79
5.2	Fan-out and driving large loads	82
5.3	Fan-in and logical effort	84
5.4	Delay calculation	87
5.5	Optimizing delay	89
5.6	Wire delay	92
5.7	Power dissipation in CMOS circuits	95
5.7.1	Dynamic power	96
5.7.2	Static power	97
5.7.3	Power scaling	97
	Summary	98
	Bibliographic notes	99
	Exercises	99

6	Combinational logic design	103
6.1	Combinational logic	103
6.2	Closure	104
6.3	Truth tables, minterms, and normal form	105
6.4	Implicants and cubes	108
6.5	Karnaugh maps	112
6.6	Covering a function	114
6.7	From a cover to gates	115
6.8	Incompletely specified functions	116
6.9	Product-of-sums implementation	118
6.10	Hazards	120
	Summary	122
	Bibliographic notes	123
	Exercises	123
7	Verilog descriptions of combinational logic	128
7.1	The prime number circuit in Verilog	128
7.1.1	A Verilog module	129
7.1.2	The <code>case</code> statement	130
7.1.3	The <code>case</code> statement	132
7.1.4	The <code>assign</code> statement	133
7.1.5	Structural description	134
7.1.6	The decimal prime number function	136
7.2	A testbench for the prime number circuit	137
7.3	Example: a seven-segment decoder	141
	Summary	146
	Bibliographic notes	146
	Exercises	147
8	Combinational building blocks	150
8.1	Multi-bit notation	150
8.2	Decoders	150
8.3	Multiplexers	155
8.4	Encoders	161
8.5	Arbiters and priority encoders	165
8.6	Comparators	170
8.7	Shifters	173
8.8	Read-only memories	173
8.9	Read-write memories	177
8.10	Programmable logic arrays	180
8.11	Data sheets	181
8.12	Intellectual property	182

Summary	183
Bibliographic notes	184
Exercises	184
9 Combinational examples	187
9.1 Multiple-of-3 circuit	187
9.2 Tomorrow circuit	189
9.3 Priority arbiter	192
9.4 Tic-tac-toe	193
Summary	201
Exercises	202
 Part III Arithmetic circuits	
10 Arithmetic circuits	207
10.1 Binary numbers	207
10.2 Binary addition	210
10.3 Negative numbers and subtraction	216
10.4 Multiplication	223
10.5 Division	226
Summary	230
Exercises	230
11 Fixed- and floating-point numbers	236
11.1 Representation error: accuracy, precision, and resolution	236
11.2 Fixed-point numbers	238
11.2.1 Representation	238
11.2.2 Operations	241
11.3 Floating-point numbers	243
11.3.1 Representation	243
11.3.2 Denormalized numbers and gradual underflow	245
11.3.3 Floating-point multiplication	245
11.3.4 Floating-point addition/subtraction	247
Summary	250
Bibliographic note	251
Exercises	251
12 Fast arithmetic circuits	255
12.1 Carry look-ahead	255
12.2 Booth recoding	260
12.3 Wallace trees	265
12.4 Synthesis notes	269

Summary	271
Bibliographic notes	272
Exercises	272
13 Arithmetic examples	275
13.1 Complex multiplication	275
13.2 Converting between fixed- and floating-point formats	276
13.2.1 Floating-point format	276
13.2.2 Fixed- to floating-point conversion	279
13.2.3 Floating- to fixed-point conversion	282
13.3 FIR filter	283
Summary	284
Bibliographic note	285
Exercises	285
 Part IV Synchronous sequential logic	
14 Sequential logic	291
14.1 Sequential circuits	291
14.2 Synchronous sequential circuits	293
14.3 Traffic-light controller	296
14.4 State assignment	299
14.5 Implementation of finite-state machines	300
14.6 Verilog implementation of finite-state machines	303
Summary	309
Bibliographic notes	309
Exercises	310
15 Timing constraints	314
15.1 Propagation and contamination delay	314
15.2 The D flip-flop	317
15.3 Setup- and hold-time constraints	318
15.4 The effect of clock skew	321
15.5 Timing examples	323
15.6 Timing and logic synthesis	324
Summary	326
Bibliographic notes	327
Exercises	327
16 Datapath sequential logic	331
16.1 Counters	331
16.1.1 A simpler counter	331

16.1.2	Up/down/load counter	333
16.1.3	A timer	336
16.2	Shift registers	338
16.2.1	A simple shift register	338
16.2.2	Left/right/load (LRL) shift register	339
16.2.3	Universal shifter/counter	340
16.3	Control and data partitioning	342
16.3.1	Example: vending machine FSM	342
16.3.2	Example: combination lock	348
	Summary	356
	Exercises	357
17	Factoring finite-state machines	360
17.1	A light flasher	360
17.2	Traffic-light controller	367
	Summary	378
	Exercises	378
18	Microcode	383
18.1	Simple microcoded FSM	383
18.2	Instruction sequencing	387
18.3	Multi-way branches	394
18.4	Multiple instruction types	397
18.5	Microcode subroutines	400
18.6	Simple computer	403
	Summary	406
	Bibliographic notes	408
	Exercises	411
19	Sequential examples	414
19.1	Divide-by-3 counter	414
19.2	SOS detector	415
19.3	Tic-tac-toe game	422
19.4	Huffman encoder/decoder	422
19.4.1	Huffman encoder	423
19.4.2	Huffman decoder	427
	Summary	430
	Bibliographic note	430
	Exercises	430

Part V Practical design

20	Verification and test	435
20.1	Design verification	435
20.1.1	Verification coverage	435
20.1.2	Types of tests	437
20.1.3	Static timing analysis	437
20.1.4	Formal verification	438
20.1.5	Bug tracking	438
20.2	Test	438
20.2.1	Fault models	439
20.2.2	Combinational testing	439
20.2.3	Testing redundant logic	440
20.2.4	Scan	441
20.2.5	Built-in-self-test (BIST)	442
20.2.6	Characterization	443
	Summary	444
	Bibliographic notes	444
	Exercises	445

Part VI System design

21	System-level design	449
21.1	System design process	449
21.2	Specification	450
21.2.1	Pong	451
21.2.2	DES cracker	452
21.2.3	Music player	454
21.3	Partitioning	456
21.3.1	Pong	456
21.3.2	DES cracker	457
21.3.3	Music synthesizer	457
	Summary	459
	Bibliographic notes	459
	Exercises	460
22	Interface and system-level timing	461
22.1	Interface timing	461
22.1.1	Always valid timing	461
22.1.2	Periodically valid signals	462
22.1.3	Flow control	463

22.2	Interface partitioning and selection	465
22.3	Serial and packetized interfaces	465
22.4	Isochronous timing	468
22.5	Timing tables	469
22.5.1	Event flow	470
22.5.2	Pipelining and anticipatory timing	471
22.6	Interface and timing examples	471
22.6.1	Pong	471
22.6.2	DES cracker	472
22.6.3	Music player	473
	Summary	476
	Exercises	476
23	Pipelines	479
23.1	Basic pipelining	479
23.2	Example pipelines	482
23.3	Example: pipelining a ripple-carry adder	484
23.4	Pipeline stalls	488
23.5	Double buffering	489
23.6	Load balance	494
23.7	Variable loads	494
23.8	Resource sharing	499
	Summary	500
	Bibliographic notes	500
	Exercises	501
24	Interconnect	504
24.1	Abstract interconnect	504
24.2	Buses	505
24.3	Crossbar switches	507
24.4	Interconnection networks	510
	Summary	512
	Bibliographic notes	512
	Exercises	513
25	Memory systems	515
25.1	Memory primitives	515
25.1.1	SRAM arrays	515
25.1.2	DRAM chips	517
25.2	Bit-slicing and banking memory	519
25.3	Interleaved memory	521
25.4	Caches	524
	Summary	528

Bibliographic notes	528
Exercises	528

Part VII Asynchronous logic

26 Asynchronous sequential circuits	533
26.1 Flow-table analysis	533
26.2 Flow-table synthesis: the toggle circuit	536
26.3 Races and state assignment	540
Summary	544
Bibliographic notes	545
Exercises	545
27 Flip-flops	548
27.1 Inside a latch	548
27.2 Inside a flip-flop	551
27.3 CMOS latches and flip-flops	553
27.4 Flow-table derivation of the latch	555
27.5 Flow-table synthesis of a D-flip-flop	557
Summary	559
Bibliographic notes	559
Exercises	559
28 Metastability and synchronization failure	563
28.1 Synchronization failure	563
28.2 Metastability	564
28.3 Probability of entering and leaving an illegal state	567
28.4 Demonstration of metastability	570
Summary	573
Bibliographic notes	574
Exercises	574
29 Synchronizer design	576
29.1 Where are synchronizers used?	576
29.2 Brute-force synchronizer	577
29.3 The problem with multi-bit signals	579
29.4 FIFO synchronizer	581
Summary	588
Bibliographic notes	589
Exercises	589

Appendix A: Verilog coding style	592
A.1 Basic principles	592
A.2 All state should be in explicitly declared registers	593
A.3 Define combinational modules so they are easy to read	594
A.4 Assign all variables under all conditions	596
A.5 Keep modules small	597
A.6 Large modules should be structural	599
A.7 Use descriptive signal names	599
A.8 Use symbolic names for subfields of signals	599
A.9 Define constants	600
A.10 Comments should describe intention and give rationale, not state the obvious	601
A.11 Never forget you are defining hardware	602
A.12 Read and be a critic of Verilog code	602
 References	 604
Index of Verilog modules	609
Subject index	611