

Appendix A: List of Worked Examples

EXAMPLE 2.1. CONVERTING DECIMAL TO DECIMAL.....	12
EXAMPLE 2.2. CONVERTING BINARY TO DECIMAL	13
EXAMPLE 2.3. CONVERTING OCTAL TO DECIMAL.....	13
EXAMPLE 2.4. CONVERTING HEXADECIMAL TO DECIMAL	14
EXAMPLE 2.5. CONVERTING DECIMAL TO BINARY	15
EXAMPLE 2.6. CONVERTING DECIMAL TO OCTAL.....	16
EXAMPLE 2.7. CONVERTING DECIMAL TO HEXADECIMAL	17
EXAMPLE 2.8. CONVERTING BINARY TO OCTAL	18
EXAMPLE 2.9. CONVERTING BINARY TO HEXADECIMAL	18
EXAMPLE 2.10. CONVERTING OCTAL TO BINARY	19
EXAMPLE 2.11. CONVERTING HEXADECIMAL TO BINARY.....	19
EXAMPLE 2.12. CONVERTING OCTAL TO HEXADECIMAL	20
EXAMPLE 2.13. CONVERTING HEXADECIMAL TO OCTAL	20
EXAMPLE 2.14. SINGLE-BIT BINARY ADDITION	21
EXAMPLE 2.15. MULTIPLE-BIT BINARY ADDITION	21
EXAMPLE 2.16. SINGLE-BIT BINARY SUBTRACTION.....	22
EXAMPLE 2.17. MULTIPLE-BIT BINARY SUBTRACTION.....	22
EXAMPLE 2.18. FINDING THE RANGE OF AN UNSIGNED NUMBER	24
EXAMPLE 2.19. DECIMAL VALUES THAT A 4-BIT, SIGNED MAGNITUDE CODE CAN REPRESENT.....	25
EXAMPLE 2.20. FINDING THE RANGE OF A SIGNED MAGNITUDE NUMBER.....	26
EXAMPLE 2.21. FINDING THE DECIMAL VALUE OF A SIGNED MAGNITUDE NUMBER.....	26
EXAMPLE 2.22. DECIMAL VALUES THAT A 4-BIT, ONE'S COMPLEMENT CODE CAN REPRESENT.....	27
EXAMPLE 2.23. FINDING THE RANGE OF A 1'S COMPLEMENT NUMBER	28
EXAMPLE 2.24. FINDING THE DECIMAL VALUE OF A 1'S COMPLEMENT NUMBER.....	28
EXAMPLE 2.25. DECIMAL VALUES THAT A 4-BIT, TWO'S COMPLEMENT CODE CAN REPRESENT.....	29
EXAMPLE 2.26. FINDING THE RANGE OF A TWO'S COMPLEMENT NUMBER.....	30
EXAMPLE 2.27. FINDING THE DECIMAL VALUE OF A TWO'S COMPLEMENT NUMBER.....	30
EXAMPLE 2.28. FINDING THE TWO'S COMPLEMENT CODE OF A DECIMAL NUMBER.....	31
EXAMPLE 2.29. TWO'S COMPLEMENT ADDITION.....	32
EXAMPLE 3.1. CALCULATING I_{CC} AND I_{GND} WHEN SOURCING MULTIPLE LOADS.....	49
EXAMPLE 3.2. CALCULATING I_{CC} AND I_{GND} WHEN BOTH SOURCING AND SINKING LOADS.....	50
EXAMPLE 3.3. DETERMINING IF SPECIFICATIONS ARE VIOLATED WHEN DRIVING ANOTHER GATE AS A LOAD.....	72
EXAMPLE 3.4. DETERMINING THE OUTPUT CURRENT WHEN DRIVING MULTIPLE GATES AS THE LOAD	73
EXAMPLE 3.5. DETERMINING THE OUTPUT CURRENT WHEN DRIVING A PULL-UP RESISTOR AS THE LOAD	74
EXAMPLE 3.6. DETERMINING THE OUTPUT CURRENT WHEN DRIVING A PULL-DOWN RESISTOR AS THE LOAD	75
EXAMPLE 3.7. DETERMINING THE OUTPUT CURRENT WHEN DRIVING AN LED WHERE HIGH = ON	76
EXAMPLE 3.8. DETERMINING THE OUTPUT CURRENT WHEN DRIVING AN LED WHERE HIGH = OFF	77
EXAMPLE 4.1. PROVING DEMORGAN'S THEOREM OF DUALITY USING PROOF BY EXHAUSTION.....	84
EXAMPLE 4.2. CONVERTING BETWEEN POSITIVE AND NEGATIVE LOGIC USING DUALITY.....	85
EXAMPLE 4.3. USING THE COMMUTATIVE PROPERTY TO UNTANGLE CROSSED WIRES.....	89
EXAMPLE 4.4. USING THE ASSOCIATIVE PROPERTY TO ADDRESS FAN-IN LIMITATIONS.....	90
EXAMPLE 4.5. USING THE DISTRIBUTIVE PROPERTY TO REDUCE THE NUMBER OF LOGIC GATES IN A CIRCUIT.....	91
EXAMPLE 4.6. PROVING THE ABSORPTION THEOREM USING PROOF BY EXHAUSTION.....	92
EXAMPLE 4.7. PROVING OF THE UNITING THEOREM.....	93
EXAMPLE 4.8. CONVERTING A SUM OF PRODUCTS FORM INTO ONE THAT USES ONLY NAND GATES.....	95
EXAMPLE 4.9. CONVERTING A PRODUCT OF SUMS FORM INTO ONE THAT USES ONLY NOR GATES	96
EXAMPLE 4.10. USING DEMORGAN'S THEOREM IN ALGEBRAIC FORM (1).....	97
EXAMPLE 4.11. USING DEMORGAN'S THEOREM IN ALGEBRAIC FORM (2).....	97
EXAMPLE 4.12. DETERMINING THE LOGIC EXPRESSION FROM A LOGIC DIAGRAM.....	100
EXAMPLE 4.13. DETERMINING THE TRUTH TABLE FROM A LOGIC DIAGRAM.....	101
EXAMPLE 4.14. DETERMINING THE DELAY OF A COMBINATIONAL LOGIC CIRCUIT	102

EXAMPLE 4.15. CREATING A CANONICAL SUM OF PRODUCTS LOGIC CIRCUIT USING MINTERMS.....	104
EXAMPLE 4.16. CREATING A MINTERM LIST FROM A TRUTH TABLE.....	105
EXAMPLE 4.17. CREATING EQUIVALENT FUNCTIONAL REPRESENTATIONS FROM A MINTERM LIST	106
EXAMPLE 4.18. CREATING A PRODUCT OF SUMS LOGIC CIRCUIT USING MAXTERMS	108
EXAMPLE 4.19. CREATING A MAXTERM LIST FROM A TRUTH TABLE.....	109
EXAMPLE 4.20. CREATING EQUIVALENT FUNCTIONAL REPRESENTATIONS FROM A MAXTERM LIST	110
EXAMPLE 4.21. CREATING EQUIVALENT FORMS TO REPRESENT LOGIC FUNCTIONALITY	111
EXAMPLE 4.22. MINIMIZING A LOGIC EXPRESSION ALGEBRAICALLY	113
EXAMPLE 4.23. USING A K-MAP TO FIND A MINIMIZED SUM OF PRODUCTS EXPRESSION (2-INPUT)	118
EXAMPLE 4.24. USING A K-MAP TO FIND A MINIMIZED SUM OF PRODUCTS EXPRESSION (3-INPUT)	119
EXAMPLE 4.25. USING A K-MAP TO FIND A MINIMIZED SUM OF PRODUCTS EXPRESSION (4-INPUT)	120
EXAMPLE 4.26. USING A K-MAP TO FIND A MINIMIZED PRODUCT OF SUMS EXPRESSION (2-INPUT).....	121
EXAMPLE 4.27. USING A K-MAP TO FIND A MINIMIZED PRODUCT OF SUMS EXPRESSION (3-INPUT).....	122
EXAMPLE 4.28. USING A K-MAP TO FIND A MINIMIZED PRODUCT OF SUMS EXPRESSION (4-INPUT).....	123
EXAMPLE 4.29. DERIVING THE MINIMAL SUM FROM A K-MAP.....	125
EXAMPLE 4.30. USING DON'T CARES TO PRODUCE A MINIMAL SOP LOGIC EXPRESSION.....	126
EXAMPLE 4.31. ELIMINATING A TIMING HAZARD BY INCLUDING NONESSENTIAL PRODUCT TERMS.....	131
EXAMPLE 5.1. DEFINING VHDL ENTITIES	153
EXAMPLE 5.2. MODELING LOGIC USING CONCURRENT SIGNAL ASSIGNMENTS AND LOGICAL OPERATORS	159
EXAMPLE 5.3. MODELING LOGIC USING CONDITIONAL SIGNAL ASSIGNMENTS.....	161
EXAMPLE 5.4. MODELING LOGIC USING SELECTED SIGNAL ASSIGNMENTS.....	163
EXAMPLE 5.5. MODELING LOGIC USING DELAYED SIGNAL ASSIGNMENTS (INERTIAL DELAY MODEL).....	164
EXAMPLE 5.6. MODELING LOGIC USING DELAYED SIGNAL ASSIGNMENTS (TRANSPORT DELAY MODEL).....	166
EXAMPLE 5.7. MODELING LOGIC USING STRUCTURAL VHDL (EXPLICIT PORT MAPPING)	167
EXAMPLE 5.8. MODELING LOGIC USING STRUCTURAL VHDL (POSITIONAL PORT MAPPING)	168
EXAMPLE 6.1. 2-TO-4 ONE-HOT DECODER—LOGIC SYNTHESIS BY HAND	176
EXAMPLE 6.2. 3-TO-8 ONE-HOT DECODER—VHDL MODELING USING LOGICAL OPERATORS	177
EXAMPLE 6.3. 3-TO-8 ONE-HOT DECODER—VHDL MODELING USING CONDITIONAL AND SELECT SIGNAL ASSIGNMENTS	178
EXAMPLE 6.4. 7-SEGMENT DISPLAY DECODER—TRUTH TABLE.....	179
EXAMPLE 6.5. 7-SEGMENT DISPLAY DECODER—LOGIC SYNTHESIS BY HAND.....	180
EXAMPLE 6.6. 7-SEGMENT DISPLAY DECODER—MODELING USING LOGICAL OPERATORS	181
EXAMPLE 6.7. 7-SEGMENT DISPLAY DECODER—MODELING USING CONDITIONAL AND SELECTED SIGNAL ASSIGNMENTS	182
EXAMPLE 6.8. 4-TO-2 BINARY ENCODER—LOGIC SYNTHESIS BY HAND.....	183
EXAMPLE 6.9. 4-TO-2 BINARY ENCODER—VHDL MODELING	184
EXAMPLE 6.10. 2-TO-1 MULTIPLEXER—LOGIC SYNTHESIS BY HAND	185
EXAMPLE 6.11. 4-TO-1 MULTIPLEXER—VHDL MODELING	186
EXAMPLE 6.12. 1-TO-2 DEMULTIPLEXER—LOGIC SYNTHESIS BY HAND.....	187
EXAMPLE 6.13. 1-TO-4 DEMULTIPLEXER—VHDL MODELING	188
EXAMPLE 7.1. PUSH-BUTTON WINDOW CONTROLLER—WORD DESCRIPTION	219
EXAMPLE 7.2. PUSH-BUTTON WINDOW CONTROLLER—STATE DIAGRAM.....	220
EXAMPLE 7.3. PUSH-BUTTON WINDOW CONTROLLER—STATE TRANSITION TABLE.....	221
EXAMPLE 7.4. SOLVING FOR THE NUMBER OF BITS NEEDED FOR BINARY STATE ENCODING.....	223
EXAMPLE 7.5. PUSH-BUTTON WINDOW CONTROLLER—STATE ENCODING	225
EXAMPLE 7.6. PUSH-BUTTON WINDOW CONTROLLER—NEXT STATE LOGIC	226
EXAMPLE 7.7. PUSH-BUTTON WINDOW CONTROLLER—OUTPUT LOGIC	227
EXAMPLE 7.8. PUSH-BUTTON WINDOW CONTROLLER—LOGIC DIAGRAM	228
EXAMPLE 7.9. SERIAL BIT SEQUENCE DETECTOR (PART 1).....	229
EXAMPLE 7.10. SERIAL BIT SEQUENCE DETECTOR (PART 2).....	230
EXAMPLE 7.11. SERIAL BIT SEQUENCE DETECTOR (PART 3).....	231
EXAMPLE 7.12. VENDING MACHINE CONTROLLER (PART 1).....	232
EXAMPLE 7.13. VENDING MACHINE CONTROLLER (PART 2).....	233
EXAMPLE 7.14. VENDING MACHINE CONTROLLER (PART 3).....	234
EXAMPLE 7.15. 2-BIT BINARY UP COUNTER (PART 1).....	236
EXAMPLE 7.16. 2-BIT BINARY UP COUNTER (PART 2).....	237
EXAMPLE 7.17. 2-BIT BINARY UP/DOWN COUNTER (PART 1).....	238
EXAMPLE 7.18. 2-BIT BINARY UP/DOWN COUNTER (PART 2).....	239
EXAMPLE 7.19. 2-BIT GRAY CODE UP COUNTER (PART 1).....	240

EXAMPLE 7.20. 2-BIT GRAY CODE UP COUNTER (PART 2).....	241
EXAMPLE 7.21. 2-BIT GRAY CODE UP/DOWN COUNTER (PART 1).....	242
EXAMPLE 7.22. 2-BIT GRAY CODE UP/DOWN COUNTER (PART 2).....	243
EXAMPLE 7.23. 3-BIT ONE-HOT UP COUNTER (PART 1)	244
EXAMPLE 7.24. 3-BIT ONE-HOT UP COUNTER (PART 2).....	245
EXAMPLE 7.25. 3-BIT ONE-HOT UP/DOWN COUNTER (PART 1).....	246
EXAMPLE 7.26. 3-BIT ONE-HOT UP/DOWN COUNTER (PART 2).....	247
EXAMPLE 7.27. 3-BIT ONE-HOT UP/DOWN COUNTER (PART 3).....	248
EXAMPLE 7.28. DETERMINING THE NEXT STATE LOGIC AND OUTPUT LOGIC EXPRESSION OF AN FSM.....	251
EXAMPLE 7.29. DETERMINING THE STATE TRANSITION TABLE OF AN FSM.....	252
EXAMPLE 7.30. DETERMINING THE STATE DIAGRAM OF AN FSM.....	253
EXAMPLE 7.31. DETERMINING THE MAXIMUM CLOCK FREQUENCY OF AN FSM.....	256
EXAMPLE 8.1. BEHAVIOR OF SEQUENTIAL SIGNAL ASSIGNMENTS WITHIN A PROCESS.....	268
EXAMPLE 8.2. BEHAVIOR OF CONCURRENT SIGNAL ASSIGNMENTS OUTSIDE A PROCESS.....	268
EXAMPLE 8.3. VARIABLE ASSIGNMENT BEHAVIOR.....	269
EXAMPLE 8.4. USING IF/THEN STATEMENTS TO MODEL COMBINATIONAL LOGIC.....	271
EXAMPLE 8.5. USING CASE STATEMENTS TO MODEL COMBINATIONAL LOGIC.....	273
EXAMPLE 8.6. BEHAVIORAL MODELING OF A RISING EDGE TRIGGERED D-FLIP-FLOP USING ATTRIBUTES.....	277
EXAMPLE 8.7. CREATING A VHDL TEST BENCH.....	279
EXAMPLE 8.8. USING REPORT STATEMENTS IN A VHDL TEST BENCH.....	280
EXAMPLE 8.9. USING ASSERT STATEMENTS IN A VHDL TEST BENCH.....	281
EXAMPLE 8.10. BEHAVIORAL MODELING OF A D-FLIP-FLOP USING THE RISING_EDGE() FUNCTION.....	285
EXAMPLE 8.11. WRITING TO AN EXTERNAL FILE FROM A TEST BENCH (PART 1).....	294
EXAMPLE 8.12. WRITING TO AN EXTERNAL FILE FROM A TEST BENCH (PART 2).....	295
EXAMPLE 8.13. WRITING TO AN EXTERNAL FILE FROM A TEST BENCH (PART 3).....	296
EXAMPLE 8.14. WRITING TO STD_OUT FROM A TEST BENCH (PART 1).....	297
EXAMPLE 8.15. WRITING TO STD_OUT FROM A TEST BENCH (PART 2).....	298
EXAMPLE 8.16. READING FROM AN EXTERNAL FILE IN A TEST BENCH (PART 1).....	298
EXAMPLE 8.17. READING FROM AN EXTERNAL FILE IN A TEST BENCH (PART 2).....	299
EXAMPLE 8.18. READING FROM AN EXTERNAL FILE IN A TEST BENCH (PART 3).....	300
EXAMPLE 8.19. READING SPACE-DELIMITED DATA FROM AN EXTERNAL FILE IN A TEST BENCH (PART 1).....	300
EXAMPLE 8.20. READING SPACE-DELIMITED DATA FROM AN EXTERNAL FILE IN A TEST BENCH (PART 2).....	301
EXAMPLE 8.21. READING SPACE-DELIMITED DATA FROM AN EXTERNAL FILE IN A TEST BENCH (PART 3).....	302
EXAMPLE 9.1. BEHAVIORAL MODEL OF A D-LATCH IN VHDL.....	309
EXAMPLE 9.2. BEHAVIORAL MODEL OF A D-FLIP-FLOP IN VHDL.....	310
EXAMPLE 9.3. BEHAVIORAL MODEL OF A D-FLIP-FLOP WITH ASYNCHRONOUS RESET IN VHDL.....	311
EXAMPLE 9.4. BEHAVIORAL MODEL OF A D-FLIP-FLOP WITH ASYNCHRONOUS RESET AND PRESET IN VHDL.....	312
EXAMPLE 9.5. BEHAVIORAL MODEL OF A D-FLIP-FLOP WITH SYNCHRONOUS ENABLE IN VHDL.....	313
EXAMPLE 9.6. PUSH-BUTTON WINDOW CONTROLLER IN VHDL—DESIGN DESCRIPTION.....	314
EXAMPLE 9.7. PUSH-BUTTON WINDOW CONTROLLER IN VHDL—ENTITY DEFINITION.....	314
EXAMPLE 9.8. PUSH-BUTTON WINDOW CONTROLLER IN VHDL—ARCHITECTURE.....	317
EXAMPLE 9.9. PUSH-BUTTON WINDOW CONTROLLER IN VHDL—SIMULATION WAVEFORM.....	318
EXAMPLE 9.10. PUSH-BUTTON WINDOW CONTROLLER IN VHDL—EXPLICIT STATE CODES.....	318
EXAMPLE 9.11. SERIAL BIT SEQUENCE DETECTOR IN VHDL—DESIGN DESCRIPTION AND ENTITY DEFINITION.....	319
EXAMPLE 9.12. SERIAL BIT SEQUENCE DETECTOR IN VHDL—ARCHITECTURE.....	320
EXAMPLE 9.13. SERIAL BIT SEQUENCE DETECTOR IN VHDL—SIMULATION WAVEFORM.....	320
EXAMPLE 9.14. VENDING MACHINE CONTROLLER IN VHDL—DESIGN DESCRIPTION AND ENTITY DEFINITION.....	321
EXAMPLE 9.15. VENDING MACHINE CONTROLLER IN VHDL—ARCHITECTURE.....	322
EXAMPLE 9.16. VENDING MACHINE CONTROLLER IN VHDL—SIMULATION WAVEFORM.....	323
EXAMPLE 9.17. 2-BIT BINARY UP/DOWN COUNTER IN VHDL—DESIGN DESCRIPTION AND ENTITY DEFINITION.....	323
EXAMPLE 9.18. 2-BIT BINARY UP/DOWN COUNTER IN VHDL—ARCHITECTURE (THREE PROCESS MODEL).....	324
EXAMPLE 9.19. 2-BIT BINARY UP/DOWN COUNTER IN VHDL – SIMULATION WAVEFORM.....	325
EXAMPLE 9.20. 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE UNSIGNED.....	326
EXAMPLE 9.21. 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE INTEGER.....	327
EXAMPLE 9.22. 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE STD_LOGIC_VECTOR (1).....	328
EXAMPLE 9.23. 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE STD_LOGIC_VECTOR (2).....	329
EXAMPLE 9.24. 4-BIT BINARY UP COUNTER WITH ENABLE IN VHDL.....	330

EXAMPLE 9.25. 4-BIT BINARY UP COUNTER WITH LOAD IN VHDL.....	331
EXAMPLE 9.26. RTL MODEL OF AN 8-BIT REGISTER IN VHDL.....	332
EXAMPLE 9.27. RTL MODEL OF A 4-STAGE, 8-BIT SHIFT REGISTER IN VHDL.....	333
EXAMPLE 9.28. REGISTERS AS AGENTS ON A DATA BUS—SYSTEM TOPOLOGY.....	334
EXAMPLE 9.29. REGISTERS AS AGENTS ON A DATA BUS—RTL MODEL IN VHDL.....	335
EXAMPLE 9.30. REGISTERS AS AGENTS ON A DATA BUS—SIMULATION WAVEFORM.....	336
EXAMPLE 10.1. CALCULATING THE FINAL DIGIT LINE VOLTAGE IN A DRAM BASED ON CHARGE SHARING.....	338
EXAMPLE 10.2. BEHAVIORAL MODEL OF A 4x4 ASYNCHRONOUS READ-ONLY MEMORY IN VHDL.....	363
EXAMPLE 10.3. BEHAVIORAL MODEL OF A 4x4 SYNCHRONOUS READ-ONLY MEMORY IN VHDL.....	364
EXAMPLE 10.4. BEHAVIORAL MODEL OF A 4x4 ASYNCHRONOUS READ/WRITE MEMORY IN VHDL.....	366
EXAMPLE 10.5. BEHAVIORAL MODEL OF A 4x4 SYNCHRONOUS READ/WRITE MEMORY IN VHDL.....	367
EXAMPLE 12.1. DESIGN OF A HALF ADDER.....	386
EXAMPLE 12.2. DESIGN OF A FULL ADDER.....	386
EXAMPLE 12.3. DESIGN OF A FULL ADDER OUT OF HALF ADDERS.....	388
EXAMPLE 12.4. DESIGN OF A 4-BIT RIPPLE CARRY ADDER (RCA).....	389
EXAMPLE 12.5. TIMING ANALYSIS OF A 4-BIT RIPPLE CARRY ADDER.....	390
EXAMPLE 12.6. DESIGN OF A 4-BIT CARRY LOOK AHEAD ADDER (CLA)—OVERVIEW.....	391
EXAMPLE 12.7. DESIGN OF A 4-BIT CARRY LOOK AHEAD ADDER (CLA)—ALGEBRAIC FORMATION.....	392
EXAMPLE 12.8. TIMING ANALYSIS OF A 4-BIT CARRY LOOK AHEAD ADDER.....	393
EXAMPLE 12.9. STRUCTURAL MODEL OF A FULL ADDER IN VHDL USING TWO HALF ADDERS.....	394
EXAMPLE 12.10. STRUCTURAL MODEL OF A 4-BIT RIPPLE CARRY ADDER IN VHDL.....	395
EXAMPLE 12.11. VHDL TEST BENCH FOR A 4-BIT RIPPLE CARRY ADDER USING NESTED FOR LOOPS.....	396
EXAMPLE 12.12. STRUCTURAL MODEL OF A 4-BIT CARRY LOOK AHEAD ADDER IN VHDL.....	397
EXAMPLE 12.13. 4-BIT CARRY LOOK AHEAD ADDER—SIMULATION WAVEFORM.....	398
EXAMPLE 12.14. BEHAVIORAL MODEL OF A 4-BIT ADDER IN VHDL.....	399
EXAMPLE 12.15. DESIGN OF A 4-BIT SUBTRACTOR USING FULL ADDERS.....	400
EXAMPLE 12.16. CREATING A PROGRAMMABLE INVERTER USING AN XOR GATE.....	400
EXAMPLE 12.17. DESIGN OF A 4-BIT PROGRAMMABLE ADDER/SUBTRACTOR.....	401
EXAMPLE 12.18. PERFORMING LONG MULTIPLICATION ON DECIMAL NUMBERS.....	402
EXAMPLE 12.19. PERFORMING LONG MULTIPLICATION ON BINARY NUMBERS.....	403
EXAMPLE 12.20. DESIGN OF A SINGLE-BIT MULTIPLIER.....	403
EXAMPLE 12.21. DESIGN OF A 4-BIT UNSIGNED MULTIPLIER.....	404
EXAMPLE 12.22. TIMING ANALYSIS OF A 4-BIT UNSIGNED MULTIPLIER.....	404
EXAMPLE 12.23. MULTIPLYING AN UNSIGNED BINARY NUMBERS BY TWO USING A LOGICAL SHIFT LEFT.....	405
EXAMPLE 12.24. ILLUSTRATING HOW AN UNSIGNED MULTIPLIER INCORRECTLY HANDLES SIGNED NUMBERS.....	406
EXAMPLE 12.25. PROCESS TO CORRECTLY HANDLE SIGNED NUMBERS USING AN UNSIGNED MULTIPLIER.....	407
EXAMPLE 12.26. PERFORMING LONG DIVISION ON DECIMAL NUMBERS.....	408
EXAMPLE 12.27. PERFORMING LONG MULTIPLICATION ON BINARY NUMBERS.....	409
EXAMPLE 12.28. DESIGN OF A 4-BIT UNSIGNED DIVIDER USING A SERIES OF ITERATIVE SUBTRACTORS.....	410
EXAMPLE 12.29. DIVIDING 1111_2 (15_{10}) BY 0111_2 (7_{10}) USING THE ITERATIVE SUBTRACTION ARCHITECTURE.....	411
EXAMPLE 12.30. DIVIDING AN UNSIGNED BINARY NUMBERS BY TWO USING A LOGICAL SHIFT RIGHT.....	412
EXAMPLE 13.1. MEMORY MAP FOR A 256x8 MEMORY SYSTEM.....	422
EXAMPLE 13.2. EXECUTION OF AN INSTRUCTION TO “LOAD REGISTER A USING IMMEDIATE ADDRESSING”.....	425
EXAMPLE 13.3. EXECUTION OF AN INSTRUCTION TO “LOAD REGISTER A USING DIRECT ADDRESSING”.....	426
EXAMPLE 13.4. EXECUTION OF AN INSTRUCTION TO “STORE REGISTER A USING DIRECT ADDRESSING”.....	427
EXAMPLE 13.5. EXECUTION OF AN INSTRUCTION TO “ADD REGISTERS A AND B”.....	428
EXAMPLE 13.6. EXECUTION OF AN INSTRUCTION TO “BRANCH ALWAYS”.....	429
EXAMPLE 13.7. EXECUTION OF AN INSTRUCTION TO “BRANCH IF EQUAL TO ZERO”.....	430
EXAMPLE 13.8. TOP LEVEL BLOCK DIAGRAM FOR THE 8-BIT COMPUTER SYSTEM.....	432
EXAMPLE 13.9. INSTRUCTION SET FOR THE 8-BIT COMPUTER SYSTEM.....	433
EXAMPLE 13.10. MEMORY SYSTEM BLOCK DIAGRAM FOR THE 8-BIT COMPUTER SYSTEM.....	434
EXAMPLE 13.11. CPU BLOCK DIAGRAM FOR THE 8-BIT COMPUTER SYSTEM.....	438
EXAMPLE 13.12. STATE DIAGRAM FOR LDA_IMM.....	444
EXAMPLE 13.13. SIMULATION WAVEFORM FOR LDA_IMM.....	445
EXAMPLE 13.14. STATE DIAGRAM FOR LDA_DIR.....	446
EXAMPLE 13.15. SIMULATION WAVEFORM FOR LDA_DIR.....	447
EXAMPLE 13.16. STATE DIAGRAM FOR STA_DIR.....	448

EXAMPLE 13.17. SIMULATION WAVEFORM FOR STA_DIR	449
EXAMPLE 13.18. STATE DIAGRAM FOR ADD_AB.....	450
EXAMPLE 13.19. SIMULATION WAVEFORM FOR ADD_AB.....	451
EXAMPLE 13.20. STATE DIAGRAM FOR BRA.....	452
EXAMPLE 13.21. SIMULATION WAVEFORM FOR BRA.....	453
EXAMPLE 13.22. STATE DIAGRAM FOR BEQ.....	454
EXAMPLE 13.23. SIMULATION WAVEFORM FOR BEQ WHEN TAKING THE BRANCH ($Z = 1$).....	455
EXAMPLE 13.24. SIMULATION WAVEFORM FOR BEQ WHEN THE BRANCH IS NOT TAKEN ($Z = 0$)	456

Suggested Readings

1. Wakerly JF (2006) Digital design: principles and practice, 4th edn. Pearson Education, Upper Saddle River, NJ
2. Kang S, Leblebici Y, Kim C (2015) “Combinational MOS logic circuits” in CMOS digital integrated circuits: analysis and design, 4th edn. McGraw-Hill Education, New York, NY
3. Cady FM (2007) Software and hardware engineering. Oxford University Press, Upper Saddle River, NJ
4. Ciletti MD (2003) Modeling, synthesis, and rapid prototyping with Verilog HDL. Prentice Hall, Upper Saddle River, NJ
5. Mano MM, Ciletti MD (2012) Digital design – with an introduction to the Verilog HDL, 5th edn. Pearson, Upper Saddle River, NJ
6. Yalamanchili S (1997) VHDL starter’s guide. Prentice Hall, Upper Saddle River, NJ
7. IEEE Standard VHDL Language Reference Manual (2009) in IEEE Std 1076-2008 (Revision of IEEE Std 1076-2002), Jan. 26 2009

Index

A

Absorption, 92
Abstraction, 143
AC specifications. *See* Switching characteristics
Adder/subtractor circuit, 400
Adders
 in VHDL, 393
Addition, 21, 385
AND gate, 40
Anti-fuse, 347
Associative property, 89
Asynchronous memory, 345
Axioms, 82
 logical negation, 82
 logical precedence, 83
 logical product, 82
 logical sum, 83
 logical values, 82

B

Base, 7
Base conversions, 11
 binary to decimal, 12
 binary to hexadecimal, 18
 binary to octal, 17
 decimal to binary, 15
 decimal to decimal, 11
 decimal to hexadecimal, 16
 decimal to octal, 15
 hexadecimal to binary, 19
 hexadecimal to decimal, 14
 hexadecimal to octal, 20
 octal to binary, 18
 octal to decimal, 13
 octal to hexadecimal, 19
Binary addition. *See* Addition
Binary number system, 9
Binary subtraction. *See* Subtraction
Bipolar junction transistor (BJT), 65
Bistable, 196
Boolean algebra, 81
Boolean algebra theorems, 83
Borrows, 22
Break-before-make switch behavior, 215
Buffer, 39
Byte, 10

C

Canonical product of sums, 106
Canonical sum of products, 103

Capacity, 341
Carry, 21
Carry look ahead adders (CLA), 390
Charge sharing, 357
Classical digital design flow, 146
CMOS. *See* Complementary metal oxide semiconductor (CMOS)
CMOS gates
 inverter, 58
 NAND gate, 59
 NOR gate, 62
Combinational logic analysis, 99
Combining, 93
Commutative property, 88
Complementary metal oxide semiconductor (CMOS), 4, 56
 operation, 57
Complements, 87
Complete sum, 125
Complex programmable logic device (CPLD), 375
Computer system design, 417, 425, 427, 428, 432, 433, 438, 439, 442, 444
 addressing modes, 423
 arithmetic logic unit (ALU), 419
 central processing unit, 419
 condition code register, 419
 control unit, 419
 data memory, 418
 data path, 419
 direct addressing, 423
 example 8-bit system, 432
 control unit, 442
 CPU, 438
 data path, 439
 detailed instruction execution, 444
 instruction set, 432
 memory system, 433
 general purpose registers, 419
 hardware, 417
 immediate addressing, 423
 indexed addressing, 424
 inherent addressing, 424
 input output ports, 418
 instruction register, 419
 instructions, 417
 branches, 428
 data manipulations, 427
 loads and stores, 425
 memory address register, 419
 memory map, 422
 memory mapped system, 420
 opcodes, 422

Computer system design (*cont.*)
 operands, 422
 program, 417
 counter, 419
 memory, 418
 registers, 419
 software, 417, 422
 Configurable logic block (CLB), 376
 Conjunction (\wedge), 82
 Converting between bases. *See* Base conversions
 Converting between positive and negative logic, 84
 Counters, 236, 325
 designing by hand, 236
 modeling in VHDL, 325
 Covering, 92
 Cross-coupled inverter pair, 195

D

Data sheet, 55
 7400 DC operating conditions, 69
 DC specifications, 45
 I_{IH-max} , 47
 I_{IL-max} , 47
 I_{I-max} , 47
 I_{OH-max} , 46
 I_{OL-max} , 46
 I_{O-max} , 46
 I_q (quiescent current), 48
 NM_H , 47
 NM_L , 47
 V_{IH-max} , 47
 V_{IH-min} , 47
 V_{IL-max} , 47
 V_{IL-min} , 47
 V_{OH-max} , 45
 V_{OH-min} , 45
 V_{OL-max} , 45
 V_{OL-min} , 45
 Decimal number system, 9
 Decoders, 175
 DeMorgan's Theorem, 93
 DeMorgan's Theorem of Duality, 83
 Demultiplexer design by hand, 187
 Demultiplexer modeling in VHDL, 188
 Demultiplexers, 187
 Design abstraction, 143
 Design domains, 144
 behavioral, 144
 physical, 144
 structural, 144
 Design levels, 144
 algorithmic, 144
 circuit, 144
 gate, 144
 register transfer, 144
 system, 144

Design simplicity, 3
 D-flip-flop, 207
 Digit, 9
 Digit notation, 9
 Digital design flow, 146
 Diodes, 75
 7400 DIP pinout, 69
 Discrete components, 56
 Disjunction (\vee), 82
 Distinguished one cells, 125
 Distributive property, 91
 Division, 408
 by powers of 2, 412
 signed, 412
 unsigned, 408
 using iterative subtractions, 409
 D latch, 206
 Don't cares (X), 125
 Double pole, double throw (DPDT) switch, 214
 Double pole, single throw (DPST) switch, 214
 Driving loads, 71
 Driving resistive loads, 73
 Dual in-line package (DIP), 69
 Duality, 83
 Dynamic hazard, 130
 Dynamic random access memory (DRAM), 355

E

Electrical signaling, 1
 Electrically erasable programmable read only memory (EEPROM), 350
 Encoders, 183
 Erasable programmable read only memory (EPROM), 348
 Essential prime implicant, 125

F

Field programmable gate array (FPGA), 375
 Finite state machines (FSM), 219
 behavioral modeling in VHDL, 314
 binary state encoding, 222
 design examples by hand, 229
 design process, 228
 final logic diagram, 227
 gray code state encoding, 223
 introduction, 219
 next state logic, 225
 one-hot state encoding, 224
 output logic, 226
 reset condition, 249
 state diagram, 219
 state memory, 222
 state transition table, 221
 state variables, 225
 synthesis by hand, 221

FLASH memory, 351
 NAND-FLASH, 351
 NOR-FLASH, 351
 Floating-gate transistor, 348
 Forward current (I_F), 75
 Forward voltage (V_F), 75
 Full adders, 386
 Functionally complete sets, 98
 Fuse, 347

G

Gajski and Kuhn's Y-chart, 144
 Gates, 37
 Generic array logic (GAL), 373
 Glitches, 129

H

Half adders, 386
 Hard array logic (HAL), 374
 Hazards, 129
 Hexadecimal number system, 10
 History of HDLs, 140

I

Idempotent, 87
 Identity theorem, 86
 Input/output blocks (IOBs), 381
 Integrated circuit, 56
 Inverter, 40
 Involution, 88

K

Karnaugh map (K-map), 113

L

Large scale integrated circuit (LSI) logic, 175
 Leading zero, 9
 Least significant bit (LSB), 10
 Light emitting diodes (LEDs), 75
 Logic block (LE), 376
 Logic expression, 38
 Logic families, 56
 Logic function, 38
 Logic HIGH, 44
 Logic levels, 44
 Logic LOW, 44
 Logic minimization, 112
 Logic symbol, 37
 Logic synthesis, 103
 Logic value, 45
 Logic waveform, 39
 Look-up table (LUT), 377

M

Mask read-only memory (MROM), 346
 Maxterm list (Π), 108
 Maxterms, 106
 Mealy machine, 220
 Medium scale integrated circuit (MSI) logic, 175
 Memory map model, 341
 Metal oxide semiconductor field effect transistor (MOSFET), 56
 Metastability, 196
 Minimal sum, 123, 125
 Minimization, 112
 of logic algebraically, 112
 of logic using K-maps, 116
 Minterm list (Σ), 104
 Minterms, 103
 Modern digital design flow, 146
 Moore machine, 220
 MOSFET. *See* Metal oxide semiconductor field effect transistor (MOSFET)
 Most significant bit (MSB), 10
 Multiplexer design by hand, 185
 Multiplexer modeling in VHDL, 186
 Multiplexers, 185
 Multiplication, 402
 by powers of 2, 405
 combinational multiplier, 403
 shift and add approach, 402
 signed, 405
 unsigned, 402

N

NAND-debounce circuit, 216
 NAND gate, 41
 Negation (\neg), 82
 Negative logic, 45
 Nibble, 10
 NMOS, 57
 Noise, 2
 Noise margin HIGH (MN_H), 47
 Noise margin LOW (MN_L), 47
 Non-volatile memory, 342
 NOR gate, 41
 NPN, 65
 Null element, 86
 Numerals, 7

O

Octal number system, 10
 Ohm's law, 73
 One-hot binary encoder design by hand, 183
 One-hot binary encoder modeling in VHDL, 183
 One-hot decoder design by hand, 176

One-hot decoder modeling in VHDL, 176
 One's complement numbers, 26
 OR gate, 41
 Output DC specifications. *See* DC specifications
 Output logic macrocell (OLMC), 373

P

7400 Part numbering scheme, 68
 Place and route, 147
 PMOS, 57
 PNP, 65
 Positional number system, 7
 Positional weight, 11
 Positive logic, 45
 Postulates, 82
 Power consumption, 4
 Power supplies, 48
 I_{CC} , 48
 I_{GND} , 48
 V_{CC} , 48
 Prime implicant, 117
 Product of sums (POS) form, 94
 Programmable array logic (PAL), 372
 Programmable interconnect points (PIPs), 380
 Programmable logic array (PLA), 371
 Programmable read only memory (PROM), 347
 Proof by exhaustion, 83

Q

Quiescent current (I_q), 48

R

Radix, 7
 Radix point, 8
 Random access memory (RAM), 342
 Range
 one's complement numbers, 27
 signed magnitude numbers, 25
 two's complement numbers, 29
 unsigned numbers, 23
 Read cycle, 341
 Read only memory (ROM), 342, 343
 Read/write (RW) memory, 342
 Ripple carry adders (RCA), 388
 Ripple counter, 213

S

7-Segment decoder design by hand, 179
 7-Segment decoder modeling in VHDL, 180
 Semiconductor memory, 341
 7400 Series logic families, 67
 Sequential access memory, 342
 Sequential logic analysis, 250
 Sequential logic timing, 211
 Shift register, 218
 Signaling, 1

Signed magnitude numbers, 24
 Signed numbers, 24
 Simple programmable logic device (SPLD), 375
 Single pole, double throw (SPDT) switch, 214
 Single pole, single throw (SPST) switch, 214
 Sinking current, 46, 47
 Small scale integrated circuit (SSI) logic, 175
 Sourcing and sinking multiple loads, 50
 Sourcing current, 46
 Sourcing multiple loads, 50
 SR latch, 198, 201
 SR latch with enable, 204
 Static 0 hazard, 130
 Static 1 hazard, 130
 Static random access memory (SRAM), 352
 Subtraction, 22, 400
 Sum of products (SOP) form, 94
 Switch debouncing, 213
 Switching characteristics, 51
 t_f (fall time), 51
 t_{PHL} (propagation delay HIGH to LOW), 51
 t_{PLH} (propagation delay LOW to HIGH), 51
 t_r (rise time), 51
 t_t (transition time), 51
 Synchronous memory, 345

T

Technology mapping, 147
 Timing hazards, 129
 Toggle flop (T-flop), 212
 Trailing zero, 9
 Transistor-transistor logic (TTL), 65
 Transmitter/receiver circuit, 44
 Truth table formation, 38
 TTL operation, 65
 Two's complement arithmetic, 31
 Two's complement numbers, 29

U

Uniting, 93
 Unsigned numbers, 23

V

Verification, 145
 Verilog HDL, 141
 Very large scale integrated circuit (VLSI) logic, 175
 VHDL, 139
 VHDL behavioral modeling techniques, 315, 318, 319,
 325–327, 329, 330
 adders, 393
 counters, 325
 using type INTEGER, 326
 using type STD_LOGIC_VECTOR, 327
 using type UNSIGNED, 325
 with enables, 329
 with loads, 330
 D-flip-flops, 310

- D-latches, 309
 - finite state machines, 314
 - design examples, 319
 - explicit state encoding using subtypes, 318
 - three process model, 315
 - user-enumerated state encoding, 315
 - modeling agents on a bus, 334
 - modeling memory, 362
 - modeling registers, 332
 - modeling shift registers, 333
 - RTL modeling, 332
 - VHDL constructs, 149, 155, 164, 166, 168, 265, 266, 279, 280, 291
 - architecture, 149, 153
 - assignment operator (\leftarrow), 156
 - attributes, 276
 - case statements, 272
 - component declaration, 155
 - concatenation operator, 158
 - concurrent signal assignments, 158
 - concurrent signal assignments with logical operators, 159
 - conditional signal assignments, 160
 - constant declaration, 154
 - data types, 150
 - delayed signal assignments, 164
 - inertial, 164
 - transport, 164
 - entity, 149
 - entity definition, 153
 - for loops, 275
 - if/then statements, 270
 - libraries and packages, 152
 - logical operators, 156
 - loop statements, 274
 - numerical operators, 157
 - operators, 155
 - packages, 149
 - process, 265
 - sensitivity list, 265
 - wait statement, 266
 - relational operators, 157
 - selected signal assignments, 161
 - sequential signal assignments, 267
 - shift operators, 157
 - signal declaration, 154
 - structural design, 166
 - component declaration, 155
 - component instantiation, 166
 - explicit port mapping, 166
 - port mapping, 166
 - positional port mapping, 168
 - test benches, 278
 - assert statements, 280
 - reading/writing external files, 291
 - report statements, 279
 - variables, 269
 - while loops, 275
 - VHDL data types
 - array, 152
 - bit, 150
 - bit_vector, 151
 - boolean, 150
 - character, 150
 - integer, 150
 - natural, 152
 - positive, 152
 - real, 150
 - std_logic, 282
 - std_logic_vector, 282
 - std_ulogic, 282
 - std_ulogic_vector, 282
 - string, 151
 - time, 151
 - user-defined enumerated, 151
 - VHDL packages, 282, 283, 285, 287
 - MATH_COMPLEX, 291
 - MATH_REAL, 289
 - NUMERIC_BIT, 288
 - NUMERIC_BIT_UNSIGNED, 289
 - NUMERIC_STD, 286
 - conversion functions, 287
 - type casting, 287
 - NUMERIC_STD_UNSIGNED, 288
 - standard, 149
 - STD_LOGIC_1164, 282
 - resolution function, 283
 - type conversions, 285
 - STD_LOGIC_ARITH, 302
 - STD_LOGIC_SIGNED, 303
 - STD_LOGIC_TEXTIO, 291
 - STD_LOGIC_UNSIGNED, 303
 - TEXTIO, 291
 - Volatile memory, 342
- W**
- Weight, 11
 - Word, 10
 - Write cycle, 341
- X**
- X-don't cares, 125
 - XNOR gate, 43
 - XOR gate, 42
 - XOR/XNOR gates in K-maps, 126
- Y**
- Y-chart, 144