

# Appendix A: List of Worked Examples

---

EXAMPLE 2.1 DEFINING VHDL ENTITIES .....	17
EXAMPLE 3.1 SOP LOGIC CIRCUIT: VHDL MODELING USING LOGICAL OPERATORS .....	25
EXAMPLE 3.2 3-TO-8 ONE-HOT DECODER: VHDL MODELING USING LOGICAL OPERATORS .....	26
EXAMPLE 3.3 7-SEGMENT DISPLAY DECODER: TRUTH TABLE .....	27
EXAMPLE 3.4 7-SEGMENT DISPLAY DECODER: LOGIC SYNTHESIS BY HAND .....	28
EXAMPLE 3.5 7-SEGMENT DISPLAY DECODER: VHDL MODELING USING LOGICAL OPERATORS .....	29
EXAMPLE 3.6 4-TO-2 BINARY ENCODER: LOGIC SYNTHESIS BY HAND .....	30
EXAMPLE 3.7 4-TO-2 BINARY ENCODER: VHDL MODELING USING LOGICAL OPERATORS .....	31
EXAMPLE 3.8 4-TO-1 MULTIPLEXER: VHDL MODELING USING LOGICAL OPERATORS .....	32
EXAMPLE 3.9 1-TO-4 DEMULTIPLEXER: VHDL MODELING USING LOGICAL OPERATORS .....	33
EXAMPLE 3.10 SOP LOGIC CIRCUIT: VHDL MODELING USING CONDITIONAL SIGNAL ASSIGNMENTS .....	35
EXAMPLE 3.11 3-TO-8 ONE-HOT DECODER: VHDL MODELING USING CONDITIONAL SIGNAL ASSIGNMENTS .....	36
EXAMPLE 3.12 7-SEGMENT DISPLAY DECODER: VHDL MODELING USING CONDITIONAL SIGNAL ASSIGNMENTS .....	37
EXAMPLE 3.13 4-TO-2 BINARY ENCODER: VHDL MODELING USING CONDITIONAL SIGNAL ASSIGNMENTS .....	38
EXAMPLE 3.14 4-TO-1 MULTIPLEXER: VHDL MODELING USING CONDITIONAL SIGNAL ASSIGNMENTS .....	39
EXAMPLE 3.15 1-TO-4 DEMULTIPLEXER: VHDL MODELING USING CONDITIONAL SIGNAL ASSIGNMENTS .....	40
EXAMPLE 3.16 SOP LOGIC CIRCUIT: VHDL MODELING USING SELECTED SIGNAL ASSIGNMENTS .....	42
EXAMPLE 3.17 3-TO-8 ONE-HOT DECODER: VHDL MODELING USING SELECTED SIGNAL ASSIGNMENTS .....	43
EXAMPLE 3.18 7-SEGMENT DISPLAY DECODER: VHDL MODELING USING SELECTED SIGNAL ASSIGNMENTS .....	44
EXAMPLE 3.19 4-TO-2 BINARY ENCODER: VHDL MODELING USING SELECTED SIGNAL ASSIGNMENTS .....	45
EXAMPLE 3.20 4-TO-1 MULTIPLEXER: VHDL MODELING USING SELECTED SIGNAL ASSIGNMENTS .....	46
EXAMPLE 3.21 1-TO-4 DEMULTIPLEXER: VHDL MODELING USING SELECTED SIGNAL ASSIGNMENTS .....	47
EXAMPLE 3.22 MODELING LOGIC USING DELAYED SIGNAL ASSIGNMENTS (INERTIAL DELAY MODEL) .....	48
EXAMPLE 3.23 MODELING LOGIC USING DELAYED SIGNAL ASSIGNMENTS (TRANSPORT DELAY MODEL) .....	49
EXAMPLE 4.1 MODELING LOGIC USING STRUCTURAL VHDL (EXPLICIT PORT MAPPING) .....	54
EXAMPLE 4.2 MODELING LOGIC USING STRUCTURAL VHDL (POSITIONAL PORT MAPPING) .....	55
EXAMPLE 4.3 DESIGN OF A HALF ADDER .....	56
EXAMPLE 4.4 DESIGN OF A FULL ADDER .....	57
EXAMPLE 4.5 DESIGN OF A FULL ADDER OUT OF HALF ADDERS .....	58
EXAMPLE 4.6 DESIGN OF A 4-BIT RIPPLE CARRY ADDER (RCA) .....	59
EXAMPLE 4.7 STRUCTURAL MODEL OF A FULL ADDER IN VHDL USING TWO HALF ADDERS .....	60
EXAMPLE 4.8 STRUCTURAL MODEL OF A 4-BIT RIPPLE CARRY ADDER IN VHDL .....	61
EXAMPLE 5.1 BEHAVIOR OF SEQUENTIAL SIGNAL ASSIGNMENTS WITHIN A PROCESS .....	68
EXAMPLE 5.2 BEHAVIOR OF CONCURRENT SIGNAL ASSIGNMENTS OUTSIDE A PROCESS .....	68
EXAMPLE 5.3 VARIABLE ASSIGNMENT BEHAVIOR .....	69
EXAMPLE 5.4 USING IF/THEN STATEMENTS TO MODEL COMBINATIONAL LOGIC .....	71
EXAMPLE 5.5 USING CASE STATEMENTS TO MODEL COMBINATIONAL LOGIC .....	73
EXAMPLE 5.6 BEHAVIORAL MODELING OF A RISING EDGE TRIGGERED D-FLIP-FLOP USING ATTRIBUTES .....	77
EXAMPLE 6.1 BEHAVIORAL MODELING OF A D-FLIP-FLOP USING THE RISING_EDGE() FUNCTION .....	84
EXAMPLE 6.2 BEHAVIORAL MODEL OF A 4-BIT ADDER IN VHDL .....	87
EXAMPLE 7.1 CREATING A VHDL TEST BENCH .....	100
EXAMPLE 7.2 VHDL TEST BENCH FOR A 4-BIT RIPPLE CARRY ADDER USING NESTED FOR LOOPS .....	101
EXAMPLE 7.3 USING REPORT STATEMENTS IN A VHDL TEST BENCH .....	103
EXAMPLE 7.4 USING ASSERT STATEMENTS IN A VHDL TEST BENCH .....	104
EXAMPLE 7.5 WRITING TO AN EXTERNAL FILE FROM A TEST BENCH (PART 1) .....	105
EXAMPLE 7.6 WRITING TO AN EXTERNAL FILE FROM A TEST BENCH (PART 2) .....	106
EXAMPLE 7.7 WRITING TO AN EXTERNAL FILE FROM A TEST BENCH (PART 3) .....	107
EXAMPLE 7.8 WRITING TO STD_OUT FROM A TEST BENCH (PART 1) .....	108
EXAMPLE 7.9 WRITING TO STD_OUT FROM A TEST BENCH (PART 2) .....	109
EXAMPLE 7.10 READING FROM AN EXTERNAL FILE IN A TEST BENCH (PART 1) .....	109
EXAMPLE 7.11 READING FROM AN EXTERNAL FILE IN A TEST BENCH (PART 2) .....	110

EXAMPLE 7.12 READING FROM AN EXTERNAL FILE IN A TEST BENCH (PART 3) .....	111
EXAMPLE 7.13 READING SPACE-DELIMITED DATA FROM AN EXTERNAL FILE IN A TEST BENCH (PART 1) .....	111
EXAMPLE 7.14 READING SPACE-DELIMITED DATA FROM AN EXTERNAL FILE IN A TEST BENCH (PART 2) .....	112
EXAMPLE 7.15 READING SPACE-DELIMITED DATA FROM AN EXTERNAL FILE IN A TEST BENCH (PART 3) .....	113
EXAMPLE 8.1 BEHAVIORAL MODEL OF A D-LATCH IN VHDL .....	117
EXAMPLE 8.2 BEHAVIORAL MODEL OF A D-FLIP-FLOP IN VHDL .....	118
EXAMPLE 8.3 BEHAVIORAL MODEL OF A D-FLIP-FLOP WITH ASYNCHRONOUS RESET IN VHDL .....	119
EXAMPLE 8.4 BEHAVIORAL MODEL OF A D-FLIP-FLOP WITH ASYNCHRONOUS RESET AND PRESET IN VHDL .....	120
EXAMPLE 8.5 BEHAVIORAL MODEL OF A D-FLIP-FLOP WITH SYNCHRONOUS ENABLE IN VHDL .....	121
EXAMPLE 8.6 RTL MODEL OF AN 8-BIT REGISTER IN VHDL .....	122
EXAMPLE 8.7 RTL MODEL OF A 4-STAGE, 8-BIT SHIFT REGISTER IN VHDL .....	123
EXAMPLE 8.8 REGISTERS AS AGENTS ON A DATA BUS: SYSTEM TOPOLOGY .....	124
EXAMPLE 8.9 REGISTERS AS AGENTS ON A DATA BUS: RTL MODEL IN VHDL .....	124
EXAMPLE 8.10 REGISTERS AS AGENTS ON A DATA BUS: SIMULATION WAVEFORM .....	125
EXAMPLE 9.1 PUSH-BUTTON WINDOW CONTROLLER IN VHDL: DESIGN DESCRIPTION .....	128
EXAMPLE 9.2 PUSH-BUTTON WINDOW CONTROLLER IN VHDL: ENTITY DEFINITION .....	128
EXAMPLE 9.3 PUSH-BUTTON WINDOW CONTROLLER IN VHDL: ARCHITECTURE .....	131
EXAMPLE 9.4 PUSH-BUTTON WINDOW CONTROLLER IN VHDL: SIMULATION WAVEFORM .....	131
EXAMPLE 9.5 PUSH-BUTTON WINDOW CONTROLLER IN VHDL: EXPLICIT STATE CODES .....	132
EXAMPLE 9.6 SERIAL BIT SEQUENCE DETECTOR IN VHDL: DESIGN DESCRIPTION AND ENTITY DEFINITION .....	133
EXAMPLE 9.7 SERIAL BIT SEQUENCE DETECTOR IN VHDL: ARCHITECTURE .....	134
EXAMPLE 9.8 SERIAL BIT SEQUENCE DETECTOR IN VHDL: SIMULATION WAVEFORM .....	134
EXAMPLE 9.9 VENDING MACHINE CONTROLLER IN VHDL: DESIGN DESCRIPTION AND ENTITY DEFINITION .....	135
EXAMPLE 9.10 VENDING MACHINE CONTROLLER IN VHDL: ARCHITECTURE .....	136
EXAMPLE 9.11 VENDING MACHINE CONTROLLER IN VHDL: SIMULATION WAVEFORM .....	137
EXAMPLE 9.12 2-BIT BINARY UP/DOWN COUNTER IN VHDL: DESIGN DESCRIPTION AND ENTITY DEFINITION .....	137
EXAMPLE 9.13 2-BIT BINARY UP/DOWN COUNTER IN VHDL: ARCHITECTURE (THREE PROCESS MODEL) .....	138
EXAMPLE 9.14 2-BIT BINARY UP/DOWN COUNTER IN VHDL: SIMULATION WAVEFORM .....	139
EXAMPLE 10.1 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE UNSIGNED .....	144
EXAMPLE 10.2 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE INTEGER .....	145
EXAMPLE 10.3 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE STD_LOGIC_VECTOR (1) .....	146
EXAMPLE 10.4 4-BIT BINARY UP COUNTER IN VHDL USING THE TYPE STD_LOGIC_VECTOR (2) .....	147
EXAMPLE 10.5 4-BIT BINARY UP COUNTER WITH ENABLE IN VHDL .....	148
EXAMPLE 10.6 4-BIT BINARY UP COUNTER WITH LOAD IN VHDL .....	149
EXAMPLE 11.1 BEHAVIORAL MODEL OF A 4 × 4 ASYNCHRONOUS READ ONLY MEMORY IN VHDL .....	156
EXAMPLE 11.2 BEHAVIORAL MODEL OF A 4 × 4 SYNCHRONOUS READ ONLY MEMORY IN VHDL .....	157
EXAMPLE 11.3 BEHAVIORAL MODEL OF A 4 × 4 ASYNCHRONOUS READ/WRITE MEMORY IN VHDL .....	159
EXAMPLE 11.4 BEHAVIORAL MODEL OF A 4 × 4 SYNCHRONOUS READ/WRITE MEMORY IN VHDL .....	160
EXAMPLE 12.1 MEMORY MAP FOR A 256 × 8 MEMORY SYSTEM .....	168
EXAMPLE 12.2 EXECUTION OF AN INSTRUCTION TO “LOAD REGISTER A USING IMMEDIATE ADDRESSING” .....	171
EXAMPLE 12.3 EXECUTION OF AN INSTRUCTION TO “LOAD REGISTER A USING DIRECT ADDRESSING” .....	172
EXAMPLE 12.4 EXECUTION OF AN INSTRUCTION TO “STORE REGISTER A USING DIRECT ADDRESSING” .....	173
EXAMPLE 12.5 EXECUTION OF AN INSTRUCTION TO “ADD REGISTERS A AND B” .....	174
EXAMPLE 12.6 EXECUTION OF AN INSTRUCTION TO “BRANCH ALWAYS” .....	175
EXAMPLE 12.7 EXECUTION OF AN INSTRUCTION TO “BRANCH IF EQUAL TO ZERO” .....	176
EXAMPLE 12.8 TOP-LEVEL BLOCK DIAGRAM FOR THE 8-BIT COMPUTER SYSTEM .....	178
EXAMPLE 12.9 INSTRUCTION SET FOR THE 8-BIT COMPUTER SYSTEM .....	179
EXAMPLE 12.10 MEMORY SYSTEM BLOCK DIAGRAM FOR THE 8-BIT COMPUTER SYSTEM .....	180
EXAMPLE 12.11 CPU BLOCK DIAGRAM FOR THE 8-BIT COMPUTER SYSTEM .....	184
EXAMPLE 12.12 STATE DIAGRAM FOR LDA_IMM .....	191
EXAMPLE 12.13 SIMULATION WAVEFORM FOR LDA_IMM .....	192
EXAMPLE 12.14 STATE DIAGRAM FOR LDA_DIR .....	193
EXAMPLE 12.15 SIMULATION WAVEFORM FOR LDA_DIR .....	194
EXAMPLE 12.16 STATE DIAGRAM FOR STA_DIR .....	195
EXAMPLE 12.17 SIMULATION WAVEFORM FOR STA_DIR .....	196
EXAMPLE 12.18 STATE DIAGRAM FOR ADD_AB .....	197
EXAMPLE 12.19 SIMULATION WAVEFORM FOR ADD_AB .....	198

---

EXAMPLE 12.20 STATE DIAGRAM FOR BRA .....	199
EXAMPLE 12.21 SIMULATION WAVEFORM FOR BRA .....	200
EXAMPLE 12.22 STATE DIAGRAM FOR BEQ .....	201
EXAMPLE 12.23 SIMULATION WAVEFORM FOR BEQ WHEN TAKING THE BRANCH ( $Z = 1$ ) .....	202
EXAMPLE 12.24 SIMULATION WAVEFORM FOR BEQ WHEN THE BRANCH IS NOT TAKEN ( $Z = 0$ ) .....	203

# Index

---

## A

Abstraction, 4  
Adders  
  in VHDL, 59

## C

Capacity, 153  
Classical digital design flow, 8  
Computer system design, 163  
  addressing modes, 169  
  arithmetic logic unit (ALU), 165  
  central processing unit, 165  
  condition code register, 165  
  control unit, 165  
  data memory, 164  
  data path, 165  
  direct addressing, 170  
  example 8-bit system, 177  
    control unit, 188  
    CPU, 184  
    data path, 185  
    detailed instruction execution, 190  
    instruction set, 178  
    memory system, 179  
  general purpose registers, 165  
  hardware, 163  
  immediate addressing, 169  
  inherent addressing, 170  
  input output ports, 164  
  instruction register, 165  
  instructions, 163  
    branches, 174  
    data manipulations, 173  
    loads and stores, 170  
  memory address register, 165  
  memory mapped system, 166  
  memory map, 167  
  opcodes, 169  
  operands, 169  
  program, 163  
  program counter, 165  
  program memory, 164  
  registers, 165  
  software, 163, 169  
Counters, 143  
  modeling in VHDL, 143

## D

Demultiplexer design by hand, 32, 39, 46  
Demultiplexer, 32, 39, 46

Design abstraction, 4  
Design domains, 5  
  behavioral domain, 5  
  physical domain, 5  
  structural domain, 5  
Design levels, 5  
  algorithmic level, 5  
  circuit level, 5  
  gate level, 5  
  register transfer level, 5  
  system level, 5  
Digital design flow, 8

## F

Finite state machines (FSM)  
  behavioral modeling in VHDL, 127  
Full adders, 56

## G

Gajski and Kuhn's Y-chart, 5

## H

Half adders, 56  
History of HDLs, 1

## M

Memory map model, 153  
Modern digital design flow, 8  
Multiplexer design by hand, 31, 38, 45  
Multiplexers, 31, 38, 45

## N

Nonvolatile memory, 154

## O

One-hot binary encoder design by hand, 29  
One-hot binary encoder modeling in VHDL, 29  
One-hot decoder modeling in VHDL, 26, 35, 42

## P

Place and route, 8

## R

Random access memory (RAM), 155  
Read cycle, 153  
Read only memory (ROM), 154  
Read/write (RW) memory, 154  
Ripple carry adders (RCA), 58

**S**

Semiconductor memory, 153  
 Sequential access memory, 155  
 7-Segment decoder design by hand, 27  
 7-Segment decoder modeling in VHDL, 28

**T**

Technology mapping, 8

**V**

Verification, 6  
 Verilog HDL, 2  
 VHDL behavioral modeling techniques
 

- adders, 59
- counters, 143
  - using type INTEGER, 144
  - using type STD\_LOGIC\_VECTOR, 145
  - using type UNSIGNED, 143
  - with enables, 148
  - with loads, 149
- D-flip-flops, 118
- D-latches, 117
- finite state machines, 127
  - explicit state encoding using subtypes, 132
  - three process model, 129
  - user-enumerated state encoding, 129
- modeling agents on a bus, 123
- modeling memory, 155
- modeling registers, 122
- modeling shift registers, 122

 VHDL constructs, 16
 

- architecture, 16, 17
- assignment operator (<=), 21
- attributes, 76
- case statements, 71
- component declaration, 19
- concatenation operator, 24
- concurrent signal assignments, 24
- concurrent signal assignments with logical operators, 25
- conditional signal assignments, 34
- constant declaration, 19
- data types, 13
- delayed signal assignments, 48
  - inertial, 48
  - transport, 48
- entity, 16
- entity definition, 17
- for loops, 75
- if/then statements, 70
- libraries and packages, 17
- logical operators, 22
- loop statements, 74
- numerical operators, 23
- operators, 21
- packages, 16
- process, 65
  - sensitivity list, 65
  - wait statement, 66

- relational operators, 23
- selected signal assignments, 41
- sequential signal assignments, 67
- shift operators, 23
- signal declaration, 18
- structural design, 53
  - component declaration, 19
  - component instantiation, 53
  - explicit port mapping, 53
  - port mapping, 53
  - positional port mapping, 55
- test benches, 99
  - assert statements, 103
  - reading/writing external files, 89
  - report statements, 102
- variables, 68
- while loops, 75

 VHDL data types
 

- array, 15
- bit, 13
- bit\_vector, 14
- boolean, 13
- character, 13
- integer, 14
- natural, 15
- real, 14
- std\_logic, 81
- std\_logic\_vector, 81
- std\_ulogic, 81
- std\_ulogic\_vector, 81
- string, 14
- time, 14
- user-defined enumerated, 15

 VHDL packages, 81
 

- MATH\_COMPLEX, 95
- MATH\_REAL, 93
- NUMERIC\_BIT, 92
- NUMERIC\_BIT\_UNSIGNED, 93
- NUMERIC\_STD, 85
  - conversion functions, 88
  - type casting, 88
- NUMERIC\_STD\_UNSIGNED, 92
- standard, 16
- STD\_LOGIC\_1164, 81
  - resolution function, 82
  - type conversions, 84
- STD\_LOGIC\_ARITH, 95
- STD\_LOGIC\_SIGNED, 96
- STD\_LOGIC\_TEXTIO, 89
- STD\_LOGIC\_UNSIGNED, 96
- TEXTIO, 89

 Volatile memory, 154
**W**

Write cycle, 153

**Y**

Y-chart, 5