



Fundamentals of
**Digital and
Computer Design
with VHDL**

Richard S. Sandige
Michael L. Sandige

This page intentionally left blank

Fundamentals of Digital and Computer Design with VHDL

Richard S. Sandige

Professor Emeritus, California Polytechnic State University

Michael L. Sandige

Principal Engineer, WildTangent





FUNDAMENTALS OF DIGITAL AND COMPUTER DESIGN WITH VHDL

Published by McGraw-Hill, a business unit of The McGraw-Hill Companies, Inc., 1221 Avenue of the Americas, New York, NY 10020. Copyright © 2012 by The McGraw-Hill Companies, Inc. All rights reserved. Printed in the United States of America. No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written consent of The McGraw-Hill Companies, Inc., including, but not limited to, in any network or other electronic storage or transmission, or broadcast for distance learning.

Some ancillaries, including electronic and print components, may not be available to customers outside the United States.



This book is printed on recycled, acid-free paper containing 10% postconsumer waste.

1 2 3 4 5 6 7 8 9 0 QDB/QDB 1 0 9 8 7 6 5 4 3 2 1

ISBN 978-0-07-338069-8

MHID 0-07-338069-5

Vice President & Editor-in-Chief: *Marty Lange*

Vice President & Director of Specialized Publishing: *Janice M. Roerig-Blong*

Global Publisher: *Raghothaman Srinivasan*

Senior Sponsoring Editor: *Peter E. Massar*

Senior Marketing Manager: *Curtis Reynolds*

Developmental Editor: *Lorraine K. Buczek*

Lead Project Manager: *Jane Mohr*

Design Coordinator: *Brenda A. Rowles*

Cover Designer: *Studio Montage, St. Louis, Missouri*

Cover Image: © R.A. - *Fotolia.com*

Buyer: *Nicole Baumgartner*

Media Project Manager: *Balaji Sundararaman*

Compositor: *Lachina*

Typeface: *10/12 Times LT Std*

Printer: *Quad Graphics*

All credits appearing on pages are considered to be an extension of the copyright page.

Library of Congress Cataloging-in-Publication Data

Sandige, Richard S.

Fundamentals of digital and computer design with VHDL / Richard S. Sandige, Michael L. Sandige.
p. cm.

Includes index.

ISBN-13: 978-0-07-338069-8 (alk. paper)

ISBN-10: 0-07-338069-5 (alk. paper)

1. Digital electronics. 2. Electronic digital computers—Design and construction—Data processing. 3. VHDL (Computer hardware description language) I. Sandige, Michael L. II. Title.

TK7868.D5S253 2012

621.39'2—dc23

2011033683

To
my loving wife
Edie

Brief Contents

Preface	xiii
About the Authors	xx
1 Boolean Algebra, Boolean Functions, VHDL, and Gates	1
2 Number Conversions, Codes, and Function Minimization	37
3 Introduction to Logic Circuit Analysis and Design	67
4 Combinational Logic Circuit Design with VHDL	94
5 Bistable Memory Device Design with VHDL	125
6 Simple Finite State Machine Design with VHDL	156
7 Computer Circuits	184
8 Circuit Implementation Techniques	210
9 Complex Finite State Machine Design with VHDL	227
10 Basic Computer Architectures	279
11 Assembly Language Programming for VBC1	292
12 Designing Input/Output Circuits	316
13 Designing Instruction Memory, Loading Program Counter, and Debounced Circuit	335
14 Designing Multiplexed Display Systems	357
15 Designing Instruction Decoders	379
16 Designing Arithmetic Logic Units	398
17 Completing the Design for VBC1	416
18 Assembly Language Programming for VBC1-E	425
19 Designing Input/Output Circuits for VBC1-E	458
20 Designing the Data Memory Circuit for VBC1-E	471
21 Designing the Arithmetic, Logic, Shift, Rotate, and Unconditional Jump Circuits for VBC1-E	482
22 Designing a Circuit to Prevent Program Execution During Manual Loading for VBC1-E	493
23 Designing Extended Instruction Memory for VBC1-E	496
24 Designing the Software Interrupt Circuits for VBC1-E	504
25 Completing the Design for VBC1-E	516
A Laboratory Experiments	528
B Obtaining Simulations via the VHDL Test Bench Program	675
C FPGA Pin Connections—Handy Reference	683
D EASY1 Tutorial	687
E Three Methods for Loading Instructions into Memory	701
Index	705

Contents

Preface xiii
About the Authors xx

Chapter 1

Boolean Algebra, Boolean Functions, VHDL, and Gates 1

- 1.1 Introduction 1
 - 1.2 Basics of Boolean Algebra 1
 - 1.2.1 Venn Diagrams 2
 - 1.2.2 Black Boxes for Boolean Functions 3
 - 1.2.3 Basic Logic Symbols 4
 - 1.2.4 Boolean Algebra Postulates 7
 - 1.2.5 Boolean Algebra Theorems 8
 - 1.2.6 Proving Boolean Algebra Theorems 9
 - 1.3 Deriving Boolean Functions from Truth Tables 10
 - 1.3.1 Deriving Boolean Functions Using the 1s of the Functions 10
 - 1.3.2 Deriving Boolean Functions Using the 0s of the Functions 11
 - 1.3.3 Deriving Boolean Functions Using Minterms and Maxterms 12
 - 1.4 Writing VHDL Designs for Simple Gate Functions 15
 - 1.4.1 VHDL Design for a NOT Function 15
 - 1.4.2 VHDL Design for an AND Function 17
 - 1.4.3 VHDL Design for an OR Function 18
 - 1.4.4 VHDL Design for an XOR Function 19
 - 1.4.5 VHDL Design for a NAND Function 21
 - 1.4.6 VHDL Design for a NOR Function 22
 - 1.4.7 VHDL Design for an XNOR Function 24
 - 1.4.8 VHDL Design for a BUFFER Function 26
 - 1.4.9 VHDL Design for any Boolean Function Written in Canonical Form 27
 - 1.5 More about Logic Gates 30
 - 1.5.1 Equivalent Gate Symbols 30
 - 1.5.2 Functionally Complete Gates 31
 - 1.5.3 Equivalent Gate Circuits 32
 - 1.5.4 Compact Description Names for Gates 32
 - 1.5.5 International Logic Symbols for Gates 32
- Problems 34

Chapter 2

Number Conversions, Codes, and Function Minimization 37

- 2.1 Introduction 37
 - 2.2 Digital Circuits versus Analog Circuits 37
 - 2.2.1 Digitized Signal for the Human Heart 37
 - 2.2.2 Discrete Signals versus Continuous Signals 38
 - 2.3 Binary Number Conversions 38
 - 2.3.1 Decimal, Binary, Octal, and Hexadecimal Numbers 38
 - 2.3.2 Conversion Techniques 40
 - 2.4 Binary Codes 45
 - 2.4.1 Minimum Number of Bits for Keypads and Keyboards 45
 - 2.4.2 Commonly Used Codes: BCD, ASCII, and Others 45
 - 2.4.3 Modulo-2 Addition and Conversions between Binary and Reflective Gray Code 48
 - 2.4.4 7-Segment Code 51
 - 2.4.5 VHDL Design for a Letter Display System 52
 - 2.5 Karnaugh Map Reduction Method 54
 - 2.5.1 The Karnaugh Map Explorer 55
 - 2.5.2 Using a 2-Variable K-Map 56
 - 2.5.3 Using a 3-Variable K-Map 58
 - 2.5.4 Using a 4-Variable K-Map 60
 - 2.5.5 Don't-Care Outputs 61
- Problems 63

Chapter 3

Introduction to Logic Circuit Analysis and Design 67

- 3.1 Introduction 67
- 3.2 Integrated Circuit Devices 67
- 3.3 Analyzing and Designing Logic Circuits 69
 - 3.3.1 Analyzing and Designing Relay Logic Circuits 69
 - 3.3.2 Analyzing IC Logic Circuits 70
 - 3.3.3 Designing IC Logic Circuits 71
- 3.4 Generating Detailed Schematics 74

- 3.5 Designing Circuits in NAND/NAND and NOR/NOR Form 76**
- 3.6 Propagation Delay Time 78**
- 3.7 Decoders 79**
 - 3.7.1 Designing Logic Circuits with Decoders and Single Gates 82**
- 3.8 Multiplexers 85**
 - 3.8.1 Designing Logic Circuits with MUXs 87**
- 3.9 Hazards 88**
 - 3.9.1 Function Hazards 88**
 - 3.9.2 Logic Hazards 89**
- Problems 91**

Chapter 4

Combinational Logic Circuit Design with VHDL 94

- 4.1 Introduction 94**
- 4.2 VHDL 94**
- 4.3 The Library Part 95**
- 4.4 The Entity Declaration 96**
- 4.5 The Architecture Declaration 97**
 - 4.5.1 Comments about a Dataflow Design Style 98**
 - 4.5.2 Comments about a Behavioral Design Style 98**
 - 4.5.3 Comments about a Structural Design Style 98**
- 4.6 Dataflow Design Style 99**
- 4.7 Behavioral Design Style 102**
- 4.8 Structural Design Style 106**
- 4.9 Implementing with Wires and Buses 112**
- 4.10 VHDL Examples 116**
 - 4.10.1 Design with Scalar Inputs and Outputs 117**
 - 4.10.2 Design with Vector Inputs and Outputs 118**
 - 4.10.3 Common VHDL Constructs 120**
- Problems 121**

Chapter 5

Bistable Memory Device Design with VHDL 125

- 5.1 Introduction 125**
- 5.2 Analyzing an S-R NOR Latch 125**
 - 5.2.1 Simple On/Off Light Switch 125**
 - 5.2.2 Circuit Delay Model for an S-R NOR Latch 127**
 - 5.2.3 Characteristic Table for an S-R NOR Latch 128**
 - 5.2.4 Characteristic Equation for an S-R NOR Latch 129**
 - 5.2.5 PS/NS Table for an S-R NOR Latch 129**
 - 5.2.5 Timing Diagram for an S-R NOR Latch 130**
- 5.3 Analyzing an S-R NAND Latch 132**
 - 5.3.1 Circuit Delay Model for an S-R NAND Latch 132**

- 5.3.2 Characteristic Table for an S-R NAND Latch 132**
- 5.3.3 Characteristic Equation for an S-R NAND Latch 133**
- 5.3.4 PS/NS Table for an S-R NAND Latch 133**
- 5.3.5 Timing Diagram for an S-R NAND Latch 133**
- 5.4 Designing a Simple Clock 134**
- 5.5 Designing a D Latch 137**
 - 5.5.1 Gated S-R Latch Circuit Design 137**
 - 5.5.2 D Latch Circuit Design with S-R Latches 138**
 - 5.5.3 D Latch Circuit Design via the Characteristic Table for a D Latch 139**
 - 5.5.4 Timing Diagram for a D Latch 140**
 - 5.5.5 Creating a Clock via a D Latch 141**
 - 5.5.6 Creating an 8-bit D Latch 142**
- 5.6 Designing D Flip-Flop Circuits 143**
 - 5.6.1 Designing Master–Slave D Flip-Flop Circuits 143**
 - 5.6.2 Designing D Flip-Flop Circuits with S-R NAND Latches 146**
 - 5.6.3 Timing Diagram for Positive Edge-Triggered D Flip-Flop 149**
- Problems 150**

Chapter 6

Simple Finite State Machine Design with VHDL 156

- 6.1 Introduction 156**
- 6.2 Synchronous Circuits 156**
- 6.3 Creating D-type Flip-Flops in VHDL 157**
- 6.4 Designing Simple Synchronous Circuits 158**
- 6.5 Counter Design Using the Algorithmic Equation Method 159**
- 6.6 Nonconventional Counter Design Using the Algorithmic Equation Method 167**
- 6.7 Counter Design Using the Arithmetic Method 170**
- 6.8 Frequency Division (Slowing Down a Fast Clock Frequency) 171**
- 6.9 Counter Design Using the PS/NS Tabular Method 174**
- 6.10 Nonconventional Counter Design Using the PS/NS Tabular Method 177**
- Problems 178**

Chapter 7

Computer Circuits 184

- 7.1 Introduction 184**
- 7.2 Three-State Outputs and the Disconnected State 184**

- 7.3 Data Bus Sharing for a Microcomputer System** 187
- 7.4 More about XOR and XNOR Symbols and Functions** 190
 - 7.4.1** Odd and Even Functions 191
 - 7.4.2** Single-Bit Error Detection System 192
 - 7.4.3** Comparators and Greater Than Circuits 194
- 7.5 Adder Design** 197
 - 7.5.1** Designing a Half Adder Module 197
 - 7.5.2** Designing a Full Adder Module 198
- 7.6 Designing and Using Ripple-Carry Adders and Subtractors** 200
- 7.7 Propagation Delay Time for Ripple-Carry Adders** 203
- 7.8 Designing Carry Look-Ahead Adders** 203
- 7.9 Propagation Delay Time for Carry Look-Ahead Adders** 206
 - Problems 206

Chapter 8

Circuit Implementation Techniques 210

- 8.1 Introduction** 210
- 8.2 Programmable Logic Devices** 210
 - 8.2.1** PROMs and LUTs 212
 - 8.2.2** PLAs 213
 - 8.2.3** PALs or GALs 213
 - 8.2.4** Designing with PROMs or LUTs 214
 - 8.2.5** Designing with PLAs 215
 - 8.2.6** Designing with PALs or GALs 216
- 8.3 Positive Logic Convention and Direct Polarity Indication** 217
 - 8.3.1** Signal Names 217
 - 8.3.2** Analyzing Equivalent Circuits for the PLC and the DPI Systems 218
- 8.4 More about MUXs and DMUXs** 221
 - 8.4.1** Designing MUX Trees 223
 - 8.4.2** Designing DMUX Trees 223
 - Problems 224

Chapter 9

Complex Finite State Machine Design with VHDL 227

- 9.1 Introduction** 227
- 9.2 Designing with the Two-Process PS/NS Method** 228
- 9.3 Explanation of CPLDs and FPGAs and State Machine Encoding Styles** 231
- 9.4 Summary of Finite State Machine Models** 234

- 9.5 Designing Compact Encoded State Machines with Moore Outputs** 235
- 9.6 Designing One-Hot Encoded State Machines with Moore Outputs** 237
- 9.7 Designing Compact Encoded State Machines with Moore and Mealy Outputs** 241
- 9.8 Designing One-Hot Encoded State Machines with Moore and Mealy Outputs** 243
- 9.9 Using the Algorithmic Equation Method to Design Complex State Machines** 245
- 9.10 Improving the Reliability of Complex State Machine Designs** 251
- 9.11 Additional State Machine Design Methods** 255
 - 9.11.1** Two-Assignment PS/NS Method 256
 - 9.11.2** Hybrid PS/NS Method 259
 - Problems 262

Chapter 10

Basic Computer Architectures 279

- 10.1 Introduction** 279
- 10.2 Generic Data-Processing System or Computer** 279
- 10.3 Harvard-Type Computer and RISC Architecture** 280
- 10.4 Princeton (von Neumann)-Type Computer and CISC Architecture** 282
- 10.5 Overview of VBC1 (Very Basic Computer 1)** 283
- 10.6 Design Philosophy of VBC1** 283
- 10.7 Programmer's Register Model for VBC1** 286
- 10.8 Instruction Set Architecture for VBC1** 287
- 10.9 Format for Writing Assembly Language Programs** 289
 - Problems 290

Chapter 11

Assembly Language Programming for VBC1 292

- 11.1 Introduction** 292
- 11.2 Instruction Set for VBC1** 292
- 11.3 The IN Instruction** 293
- 11.4 The OUT Instruction** 296
- 11.5 The MOV Instruction** 298
- 11.6 The LOADI Instruction** 300
- 11.7 The ADDI Instruction** 301
- 11.8 The ADD Instruction** 303
- 11.9 The SR0 Instruction** 304
- 11.10 The JNZ Instruction** 306

- 11.11 Programming Examples and Techniques for VBC1 308**
 - 11.11.1** Unconditional Jump 308
 - 11.11.2** Labels 308
 - 11.11.3** Loop Counter 309
 - 11.11.4** Program Runs Amuck 310
 - 11.11.5** Subtraction Instruction 310
 - 11.11.6** Multiply Instruction 312
 - 11.11.7** Divide Instruction 312
 - Problems 312**

Chapter 12

Designing Input/Output Circuits 316

- 12.1** Introduction 316
- 12.2** Designing Steering Circuits 316
- 12.3** Designing Bus Steering Circuits 318
- 12.4** Designing Loadable Register Circuits 319
- 12.5** Designing Input Circuits 321
 - 12.5.1** Designing an Input Circuit Driven by Four Slide Switches 323
- 12.6** Designing Output Circuits 324
 - 12.6.1** Designing an Output Circuit to Drive Four LEDs 325
 - 12.6.2** Designing an Output Circuit to Drive a 7-Segment Display 326
 - 12.6.3** A Closer Look at the Circuitry for Display 0 328
- 12.7** Combining Input and Output Circuits to Form a Simple I/O System 329
- 12.8** Alternate VHDL Design Styles 332
 - Problems 333**

Chapter 13

Designing Instruction Memory, Loading Program Counter, and Debounced Circuit 335

- 13.1** Introduction 335
- 13.2** Designing an Instruction Memory 335
 - 13.2.1** Coding Alterations for Instruction Memory 337
 - 13.2.2** Initializing Instruction Memory for VBC1 at Startup 339
- 13.3** Designing a Loading Program Counter 342
- 13.4** Designing a Debounced One-Pulse Circuit 345
- 13.5** Design Verification for a Debounced One-Pulse Circuit 348
 - Problems 355**

Chapter 14

Designing Multiplexed Display Systems 357

- 14.1** Introduction 357
- 14.2** Multiplexed Display System for Four 7-Segment LED Displays 357
- 14.3** Designing a Multiplexed Display System Using VHDL 360
 - 14.3.1** Designing Module 1: A 4-to-1 MUX Array 360
 - 14.3.2** Designing Module 2: A HEX Display Decoder 361
 - 14.3.3** Designing Module 3: A 2-bit Counter and a Frequency Divider 362
 - 14.3.4** Designing Module 4: A 2-to-4 Decoder 364
- 14.4** Complete Design of a Multiplexed Display System Using a Flat Design Approach 364
- 14.5** Complete Design of a Multiplexed Display System Using a Hierarchical Design Approach 367
- 14.6** Designing a Word Display System Using a Flat Design Approach 372
 - Problems 377**

Chapter 15

Designing Instruction Decoders 379

- 15.1** Introduction 379
- 15.2** Purpose of the Instruction Decoder 379
- 15.3** Instruction Decoder Truth Tables for the IN, OUT, and MOV Instructions 380
- 15.4** Designing an Instruction Decoder for the IN Instruction 382
- 15.5** Designing an Instruction Decoder for the OUT and MOV Instructions 383
- 15.6** Instruction Decoder Truth Table for the LOADI Instruction 384
- 15.7** Instruction Decoder Truth Table for the ADDI Instruction 385
- 15.8** Instruction Decoder Truth Table for the ADD Instruction 386
- 15.9** Instruction Decoder Truth Table for the SR0 Instruction 387
- 15.10** Designing an Instruction Decoder for the SR0 Instruction 388
- 15.11** Instruction Decoder Truth Table for the JNZ Instruction 389
- 15.12** Designing an Instruction Decoder for the JNZ Instruction 391
- 15.13** Designing an Instruction Decoder for VBC1 393
 - Problems 393**

Chapter 16

Designing Arithmetic Logic Units 398

- 16.1 Introduction 398
- 16.2 Utilization of the Arithmetic Logic Unit 398
- 16.3 Designing the LOADI Instruction Part of the ALU 399
- 16.4 Designing the ADDI Instruction Part of the ALU 400
- 16.5 Designing the ADD Instruction Part of the ALU 401
- 16.6 Designing the SR0 Instruction Part of the ALU 401
- 16.7 Designing an ALU for VBC1 402
- 16.8 Additional Circuit Designs with VHDL 403
 - 16.8.1 Designing Additional ALU Circuits 403
 - 16.8.2 Designing Shifter Circuits 406
 - 16.8.3 Designing Barrel Shifter Circuits 409
 - 16.8.4 Designing Shift Register Circuits 412
- Problems 414

Chapter 17

Completing the Design for VBC1 416

- 17.1 Introduction 416
- 17.2 Designing a Running Program Counter 416
- 17.3 Combining a Loading and a Running Program Counter 419
- 17.4 Designing a Run Frequency Circuit and a Speed Circuit 421
- 17.5 Designing Circuits to Provide a Loader for Instruction Memory for VBC1 423
- Problems 424

Chapter 18

Assembly Language Programming for VBC1-E 425

- 18.1 Introduction 425
- 18.2 Instruction Summary 425
- 18.3 Input, Output, and Interrupt Instructions 427
- 18.4 Data Memory Instructions 432
- 18.5 Arithmetic and Logic Instructions 434
- 18.6 Shift and Rotate Instructions 437
- 18.7 Jump, Jump Relative, and Halt Instructions 439
- 18.8 More about Interrupts and Assembler Directives 443
- 18.9 Complete Instruction Set Summary for VBC1-E 448
- Problems 449

Chapter 19

Designing Input/Output Circuits for VBC1-E 458

- 19.1 Introduction 458
- 19.2 Designing the Input Circuit for VBC1-E 458
- 19.3 Instruction Decoder Truth Table for the Modified IN Instruction for VBC1-E 460
- 19.4 Designing the Output Circuit for VBC1-E 462
- 19.5 Instruction Decoder Truth Table for the Modified OUT Instruction for VBC1-E 464
- 19.6 Designing an Instruction Decoder for the Modified IN and OUT Instructions for VBC1-E 466
- 19.7 Designing an Instruction Decoder for the LOADI, ADDI, and JNZ Instructions for VBC1-E 467
- Problems 468

Chapter 20

Designing the Data Memory Circuit for VBC1-E 471

- 20.1 Introduction 471
- 20.2 Designing the Data Memory for VBC1-E 471
- 20.3 Designing Circuits to Select the Registers and Data for VBC1-E 475
- 20.4 Instruction Decoder Truth Tables for the STORE and FETCH Instructions for VBC1-E 475
- 20.5 Designing an Instruction Decoder for the STORE and FETCH Instructions for VBC1-E 478
- 20.6 Designing an Instruction Decoder for the MOV Instruction for VBC1-E 479
- Problems 480

Chapter 21

Designing the Arithmetic, Logic, Shift, Rotate, and Unconditional Jump Circuits for VBC1-E 482

- 21.1 Introduction 482
- 21.2 Designing the Arithmetic and Logic Instructions Part of the ALU for VBC1-E 482
- 21.3 Designing the Instruction Decoder for the Arithmetic and Logic Instructions for VBC1-E 484
- 21.4 Designing the Shift and Rotate Instructions Part of the ALU for VBC1-E 485
- 21.5 Designing the Instruction Decoder for the Shift and Rotate Instructions for VBC1-E 486
- 21.6 Designing the JMP and JMPR Circuits for VBC1-E 488
- 21.7 Designing the Instruction Decoder for the JMP and JMPR Instructions for VBC1-E 489
- Problems 490

Chapter **22**

Designing a Circuit to Prevent Program Execution During Manual Loading for VBC1-E 493

- 22.1 Introduction 493
- 22.2 Designing a Circuit to Modify Manual Loading for VBC1-E 493
- 22.3 Modifying the Instruction Decoder for Manual Loading for VBC1-E 495
 - Problems 495

Chapter **23**

Designing Extended Instruction Memory for VBC1-E 496

- 23.1 Introduction 496
- 23.2 Modifying the Instruction Memory to Add Extended Instruction Memory for VBC1-E 496
- 23.3 Modifying the Running Program Counter Circuit for VBC1-E 500
- 23.4 Modifying the Proper Address Circuit for VBC1-E 501
- 23.5 Modifying the Loading Program Counter Circuit for VBC1-E 501
- 23.6 Modifying the JMPR Circuit for VBC1-E 502
 - Problems 502

Chapter **24**

Designing the Software Interrupt Circuits for VBC1-E 504

- 24.1 Introduction 504
- 24.2 Designing the Modified Circuit for the Running Program Counter and the Select Circuit for VBC1-E 504
- 24.3 Designing the Circuit to Store PCPLUS1 for VBC1-E 509
- 24.4 Instruction Decoder Truth Tables for the INT and IRET Instructions for VBC1-E 510
- 24.5 Designing the Instruction Decoder for the INT and IRET Instructions for VBC1-E 511
 - Problems 513

Chapter **25**

Completing the Design for VBC1-E 516

- 25.1 Introduction 516
- 25.2 Designing a Debounced One-Pulse Trigger Interrupt Circuit and Modifying the RPC Circuit for VBC1-E 516
- 25.3 Designing Circuits for Displaying the Signal *RETA* for VBC1-E 521
- 25.4 Designing Circuits to Provide a Loader for Instruction Memory for VBC1-E 525
 - Problems 525

Appendices

A Laboratory Experiments 528

- Experiment 1A:** Designing and Simulating Gates 528
- Experiment 1B:** Completing the Design Cycle 534
- Experiment 2:** Designing and Testing a Keypad Encoder System 539
- Experiment 3:** Designing and Testing a Check Gates System 542
- Experiment 4:** Designing and Testing a Custom Decimal Display Decoder System 546
- Experiment 5A:** Designing and Testing a D Latch and a D Flip-Flop with a CLR Input 549
- Experiment 5B:** Designing and Testing an 8-bit Register and a D Flip-Flop with a PRE Input 553
- Experiment 6A:** Designing and Testing a Simple Counter System—A One-Hot Up Counter with 8 Bits 558
- Experiment 6B:** Designing and Testing a Simple Counter System—A Gray Code Counter with 2 Bits 562
- Experiment 6C:** Designing and Testing a Simple Nonconventional Counter System—A Robot Eye Circuit 565
- Experiment 6D:** Designing and Testing a Simple Nonconventional Counter—A Smiley Face Circuit 569
- Experiment 7A:** Designing and Testing a Simple Error Detection System Using a Flat Design Approach 572
- Experiment 7B:** Designing and Testing a 4-bit Simple Adder-Subtractor System Using a Hierarchical Design Approach 577

- Experiment 8:** Designing and Testing a LUT Design System Using a Flat Design Approach 580
 - Experiment 9A:** Designing and Testing a One-Hot Up/Down Counter System Using a Flat Design Approach 584
 - Experiment 9B:** Designing and Testing a 10-State Counter System Using a Hierarchical Design Approach 589
 - Experiment 10:** Working with EASY1 (Editor/Assembler/Simulator) for VBC1 593
 - Experiment 11:** Writing and Simulating Programs for VBC1 with EASY1 598
 - Experiment 12:** Designing and Testing VBC1 (Data Path Unit) 600
 - Experiment 13:** Designing and Testing VBC1 (Instruction Memory Unit) 605
 - Experiment 14:** Designing and Testing VBC1 (Monitor System) 609
 - Experiment 15:** Designing and Testing VBC1 (Instruction Decoder) 613
 - Experiment 16:** Designing and Testing VBC1 (Arithmetic Logic Unit) 617
 - Experiment 17:** Designing and Testing VBC1 (Final Hardware Design for VBC1) 621
 - Experiment 17L:** Designing a Loader for Instruction Memory for VBC1 626
 - Experiment 18:** Writing Assembly Language Programs and Running Them on VBC1 632
 - Experiment 19:** Designing and Testing VBC1-E (IN, OUT, and Unchanged Instructions) 635
 - Experiment 20:** Designing and Testing VBC1-E (MOV and Data Memory Instructions) 640
 - Experiment 21:** Designing and Testing VBC1-E (Almost All Instructions) 645
 - Experiment 22:** Designing and Testing VBC1-E (Modified Manual Loading) 651
 - Experiment 23:** Designing and Testing VBC1-E (Add Extended Instruction Memory) 654
 - Experiment 24:** Designing and Testing VBC1-E (INT and IRET Instructions) 658
 - Experiment 25:** Designing and Testing VBC1-E (Final Hardware Design for VBC1-E) 663
 - Experiment 25L:** Designing a Loader for Instruction Memory for VBC1-E 668
- B** Obtaining Simulations via the VHDL Test Bench Program 675
 - B.1** Introduction 675
 - B.2** Example 1—Combinational Logic Design (project: AND_3) 675
 - B.3** Example 2—Synchronous Sequential Logic Design (project: DFF) 679
- C** FPGA Pin Connections—Handy Reference 683
 - C.1** BASYS 2 Board 683
 - C.2** NEXYS 2 Board 684
 - C.3** Memory Loader I/O Pin Connections for the FPGAs on the BASYS 2 and NEXYS 2 Board 685
 - C.4** FX2 MIB (Module Interface Board)—Add-on Board for NEXYS 2 686
- D** EASY1 Tutorial 687
 - D.1** Introduction 687
 - D.2** EASY1 Screen or GUI 687
 - D.3** EASY1 Layout 687
 - D.4** How to Use EASY1 689
 - D.5** Example 1—A Simple Input/Output Program 689
 - D.6** Example 2—Input/Output Program Modified to Run Continuously 695
 - D.7** Example 3—A Simple State Machine Program 696
 - D.8** Example 4—A Complex State Machine Program 696
 - D.9** Example 5—Generating Time Delays 698
 - D.10** Using EASY1 to Generate Machine Code for VBC1 699
- E** Three Methods for Loading Instructions into Memory 701
 - E.1** Loading Memory Manually 701
 - E.2** Initializing Memory at Startup 702
 - E.3** Loading Memory via the Memory Loader Program 703
- Index 705