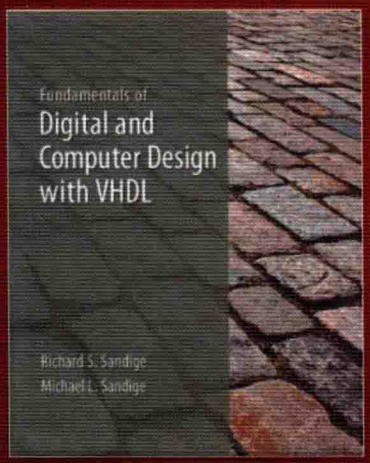
Mc Graw Hill Education

清华版双语教学用书



数字逻辑与计算机设计

—— VHDL语言描述

Fundamentals of Digital and Computer Design with VHDL

Richard S. Sandige
Michael L. Sandige 著

罗 嵘 编译

清华大学出版社

清华版双语教学用书

数字逻辑与计算机设计

—— VHDL语言描述

Fundamentals of Digital and Computer Design with VHDL

Richard S. Sandige
Michael L. Sandige 著

罗 嵘 编译

清华大学出版社
北京

Richard S. Sandige, Michael L. Sandige
Fundamentals of Digital and Computer Design with VHDL
ISBN: 9780073380698

图书在版编目(CIP)数据

数字逻辑与计算机设计: VHDL 语言描述/(美)桑迪格(Sandige, R. S.),(美)桑迪格(Sandige, M. L.)著; 罗嵘编译. —北京: 清华大学出版社, 2015
清华版双语教学用书
ISBN 978-7-302-38683-4

Ⅰ. ①数… Ⅱ. ①桑… ②桑… ③罗… Ⅲ. ①VHDL 语言-程序设计-双语教学-教材-英、汉 Ⅳ. ①TP301.2

中国版本图书馆 CIP 数据核字(2014)第 283778 号

责任编辑: 盛东亮
封面设计: 何凤霞
责任校对: 白 蕾
责任印制: 刘海龙

出版发行: 清华大学出版社
网 址: http://www.tup.com.cn, http://www.wqbook.com
地 址: 北京清华大学学研大厦 A 座 邮 编: 100084
社 总 机: 010-62770175 邮 购: 010-62786544
投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn
质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn
印 刷 者: 清华大学印刷厂
装 订 者: 三河市新茂装订有限公司
经 销: 全国新华书店
开 本: 203mm×260mm **印 张**: 47 **字 数**: 1319 千字
版 次: 2015 年 1 月第 1 版 **印 次**: 2015 年 1 月第1 次印刷
印 数: 1～2500
定 价: 99.00 元

产品编号: 051699-01

To
my loving wife
Edie

译 者 序

当收到清华大学出版社盛东亮编辑的邀请，编译这本书的目录及书中各节的导读和相关术语时，我也正在翻译这本书的全文，所以就欣然允诺，担任了中文翻译版和双语版两个版本的翻译工作。

通过翻译这本书，我开始真正体会到这本书非常适合电子信息类专业本科生，它由浅入深、循序渐进地把设计一个简单计算机的方法展开论述，利用 VHDL 语言和可编程逻辑器件完成一个基本的计算机系统的设计，便于学生们学习和理解。因此，我欣然接受了邀请，编译此书以便学生和想了解计算机硬件设计的工程师们阅读英文原版书籍。

本书由 Richard S. Sandige 和 Michael L. Sandige 所著。前者是加州理工州立大学的一名退休名誉教授，他在美国多所大学从事过教学工作，还在多家公司从事过研发工作，具有丰富的教学和技术开发经验。后者则非常热爱游戏的研发，目前是 WildTangent 公司的首席工程师。两位作者将本书分成数字设计和计算机设计两部分内容，数字设计部分包括第 1 章至第 9 章的内容，计算机设计包括第 10 章至第 25 章的内容。此外，还撰写了 5 个附录，涵盖了与前面 25 章的内容对应的 34 个实验，以及实验中涉及的软件和硬件开发板的使用手册等内容。

罗 嵘

2015 年 1 月于清华大学

Preface

This book is intended for an introductory digital design course for students at the freshman level; it is also intended for an introductory computer design course with assembly language programming for students at the sophomore level or above. The material in the book is suitable for students who study computer engineering, computer science, and electrical engineering. This book uses a spiral teaching approach by introducing a design problem and then, in the same or a later chapter, either (1) reemphasizing the same concepts when a different design problem is presented, or (2) working the same problem using a different technique. This is done to increase the likelihood of retention.

There is no prerequisite for the book; however, computer familiarity and/or a first programming course usually helps students to learn VHDL and assembly language programming.

KEY FEATURES OF THIS TEXT

- Generic VHDL code is taught and used throughout the book so that different companies' VHDL tools can be used if desired.
- Classical and modern VHDL designs are taught to provide a balance throughout the course.
- A Karnaugh Map Explorer program is provided as an interactive tool to teach students to use Karnaugh maps with 2, 3, and 4 variables.
- Students are taught how to design VBC1 (Very Basic Computer 1) and VBC1-E (an enhanced version of VBC1), which are 4-bit educational computers. Both computers can be downloaded into a field programmable gate array (FPGA) on a development board and programmed in assembly language.
- An editor/assembler/simulator program called EASY1 is provided to teach students how to program VBC1 and VBC1-E.
- A memory loader software program is provided to teach students how to design a loader for instruction memory.
- One or more experiments are provided for every chapter and keyed by number; many have a recommended pre-lab assignment, related to either writing assembly language programs or obtaining simulations for VHDL designs.
- Homework problems are keyed to every section of the book.

CHAPTER AND TOPICAL OVERVIEW

In the digital design section of this book, the following topics are covered in Chapters 1 through 9:

- VHDL (Very High Speed Integrated Circuit Hardware Description Language) is introduced in Chapter 1 for combinational logic circuits.
- Thirty-four experiments are provided to allow students to "learn by doing." Experiments 1A through 9B provide practice for students to learn how to design VHDL circuits for digital

designs in the laboratory. The experiment numbers are keyed to the chapter numbers. The experiments may be used for homework assignments or special projects.

- Students are introduced in Chapter 2 to a 7-segment display that uses a letter display system that they can design and build with an FPGA board via VHDL to display the high (H) or low (L) level of a slide switch. The Karnaugh Map Explorer program is also provided as an interactive tool to teach students to use Karnaugh maps with 2, 3, and 4 variables.
- A graphical design method is introduced for manually designing logic circuits in NAND/NAND and NOR/NOR form in Chapter 3 and then followed up with equivalent designs using VHDL. Decoders and multiplexers (MUXs) are introduced and manual methods are used to design circuits with both decoders and MUXs. These methods are followed with equivalent designs using VHDL. Function and logic hazards are presented and students are shown how to eliminate logic hazards with logic hazard cover terms.
- Combinational logic circuit design with VHDL is presented in Chapter 4 with complete coverage of dataflow, behavioral, and structural design styles. VHDL examples that include both scalar and vector inputs and outputs are provided.
- Set-reset (S-R) latches, D latchs, and D flip-flops are designed both classically and with VHDL in Chapter 5. In this chapter students learn how to design D edge-triggered flip-flops. Experiment 5A provides hands-on learning experience in the gate-level design of a D latch and a positive edge-triggered D flip-flop with a CLR (clear) input. Experiment 5B provides a similar hands-on experience of an 8-bit register and a positive edge-triggered D flip-flop with a PRE (preset) input.
- Finite state machine design is divided into simple and complex state machines. Simple state machines are basic counters without an external input to change the counting sequence. Complex state machines have an external input(s) to change the counting sequence. Simple state machines are presented in Chapter 6. An algorithmic equation method for simple state machines is presented to show students how to design simple state machines manually. This method works for practically any size state machine but it is tedious because students must learn how to write *D* excitation equations. Students learn how to use the arithmetic method, which is especially useful when generating a slower clock frequency (or frequency divider). In addition, a present-state/next-state (PS/NS) tabular method is introduced that allows students to write the VHDL equations for basic counters using a process that eliminates the hassle of obtaining *D* excitation equations. A new counter design description called a *counting* or *state sequence diagram* is introduced alongside the traditional state diagram.
- In Chapter 7 various computer circuits are presented including three-state outputs, data bus sharing, adder and subtractor design, ripple carry adders, and carry look-ahead adders. Experiments 7A and 7B provide hands-on design experience of special combinational logic systems related to Chapter 7, that is, a simple (single-bit) error detection system and a 4-bit simple adder-subtractor system.
- Circuit implementation techniques presented in Chapter 8 show the implementation of programmable logic devices—that is, PROMs, PLAs, PALs, GALs and LUTs. This chapter also contains a brief introduction to the positive logic convention and direct polarity indication and how to convert between the two representations. A modular design technique is presented to show how to design MUX and DMUX trees. Designing and testing of a LUT design system is provided in Experiment 8.
- Complex state machine design with VHDL is presented in Chapter 9 using the two-process PS/ NS method. The first process, called the *synchronous process*, generates the flip-flops, and the second process, called the *combinational process*, decodes the next-state functions and any Moore and Mealy outputs that may be present in the design. State machine encoding styles that are used for complex programmable logic devices (CPLDs) and FPGAs are also presented in this chapter. Experiment 9A provides hands-on design experience in designing a one-hot up/down counter system using a flat design approach. Experiment 9B

provides hands-on design experience in designing a 10-state counter system using a hierarchal design approach. A synchronizer circuit is introduced that improves the reliability of complex state machine designs by reducing the possibility of metastability. To complete the discussion, two additional state machine design methods are presented—the two-assignment PS/NS method and the hybrid PS/NS method.

In the computer design section of this book, the following topics are covered in Chapters 10 through 17:

- After a brief introduction to Harvard and Princeton (von Neumann)-type computer architectures, students are introduced to a very basic Harvard-type computer called VBC1 (Very Basic Computer 1) in Chapter 10. Students learn the programmer's register model, the instruction set architecture, and the format for writing assembly language for VBC1. Students get their first introduction to writing assembly language programs for VBC1 in Experiment 10. Appendix D provides a tutorial for the editor/assembler/simulator 1 that we call EASY1.
- In Chapter 11, the assembly language form, transfer function form, and machine code form are presented for all the instructions for VBC1. Programming examples and techniques are presented for VBC1.
- The experiments accompanying Chapter 12 through Chapter 17 allow students to construct, design, and implement VBC1 via an FPGA over a period of six weeks if one experiment is performed per week. The number of experiments per week can be decided on by the instructor teaching the laboratory class. Recommended pre-lab assignments are provided.
- Chapter 12 presents the design of input/output (I/O) circuits for VBC1 and deals mainly with bus steering circuits, loadable D registers, driving light-emitting diodes (LEDs) and driving 7-segment displays.
- The design of instruction memory (for storing programs), a loading program counter (for loading instruction memory), and a debounced circuit (for single stepping through instruction memory) is presented in Chapter 13. An introduction to gated clock circuits and how to remove them is also emphasized in this chapter.
- Chapter 14 presents the design of a multiplexed display system for VBC1. To provide additional experience, students learn how to design a word display system in this chapter.
- The design of the instruction decoder for VBC1 is presented in Chapter 15 because each instruction must be decoded to automatically provide instruction execution.
- The design of the arithmetic logic unit (ALU) for VBC1 is covered in Chapter 16. In addition to an expanded ALU design, students also learn how to design the following circuits: shifter circuits, barrel shifter circuits, and shift register circuits via VHDL.
- Chapter 17 covers the final design for VBC1, which includes the design of the running program counter that allows VBC1 to run at a specified clock frequency. Experiment 17L is designed to provide the capability to automatically load programs into the instruction memory of VBC1 via a memory loader program. Up to this point the instruction memory was either loaded manually via slide switches or preloaded via initialization of the instruction memory in the VHDL code for VBC1.

In the computer design section of this book, the following topics are covered in Chapters 18 through 25:

- In Chapter 18, the assembly language form, transfer function form, and machine code form are presented for all the instructions for VBC1-E. This includes modified IN and OUT instructions (four ports each), additional instructions for a data memory (STORE and FETCH), additional arithmetic and logic instructions (SUB, NOT, AND, OR, and XNOR), additional shift and rotate instructions (SR1, SL0, SL1, RR, and RL), additional control instructions (JMP, JMPR, and HALT), additional software interrupt instructions (INT and

IRET), and a hardware interrupt capability. Two assembler directives (BIPROC and EQU) are also included in the EASY1-E assembler for VBC1-E. Experiment 18 provides students with hands-on experience with writing and simulating assembly language programs for both VBC1 and VBC1-E.

- The experiments accompanying Chapters 19 through Chapter 25 allow students to construct, design, and implement VBC1-E via an FPGA over a period of seven weeks (if one experiment is performed per week). The number of experiments per week can be decided on by the instructor teaching the laboratory class.
- In Chapter 19 students learn how to expand the I/O design of VBC1 to make VBC1-E. This includes redesigning the instruction decoder to handle the expanded I/O design for the IN and OUT instructions.
- Data memory did not exist in VBC1 so Chapter 20 covers the design of a simple data memory with four storage locations for VBC1-E.
- In Chapter 21, students learn how to enhance the design of the ALU for VBC1 to include new arithmetic and logic instructions, and new shift and rotate instructions. This includes redesigning the instruction decoder to handle the additional instructions. The design for the new control instructions JMP, JMPR, and HALT are presented, and the instruction decoder is redesigned to handle these new instructions.
- Chapter 22 is a very short chapter showing how to design a circuit to prevent program execution during manual loading. Students often find manual loading of VBC1-E to be distracting, which led to this chapter in the book.
- VBC1-E has additional storage locations and Chapter 23 covers the modification of the instruction memory to include the additional storage locations.
- VBC1-E has new software interrupt instructions and Chapter 24 covers the design of the necessary circuits to handle the instructions INT and IRET. The instruction decoder is also redesigned to handle these new instructions.
- Chapter 25 covers the final design for VBC1-E, which includes the design of the hardware interrupt capability. Experiment 25L is designed to provide the capability to automatically load programs into the instruction memory of VBC1-E via a memory loader program.

Information on CAD Tools and FPGA Boards

For the digital design portion of this book, circuits and systems are presented using both a classical methodology—that is, manual calculations—and VHDL designs. For the computer design portion of this book, VHDL is used to design VBC1 and VBC1-E. This approach encourages students to design their own digital systems and/or games once they learn how easy it is to design with VHDL. Students are not restricted to generating circuits on a small circuit board, where they must place the IC (integrated circuit) packages and wire them together. The FPGA (field programmable gate array) chips that are used on modern digital boards remove this arduous task. Xilinx® ISE® WebPACK is the primary CAD (computer-aided design) tool used in this book. The ISE WebPACK is available to students and instructors via the Xilinx web site at http://www.xilinx.com/support/download/index.htm.

A programmable logic FPGA chip can be reprogrammed over and over, which is an ideal way for students to learn by trial and error. Designs with VHDL code are simulated to verify that the VHDL code works. If there is an error in the simulation of their VHDL code, students can simply find and fix the error or errors and rerun the simulation. When the simulation is correct, the correct bit pattern for the VHDL code can be downloaded into the FPGA chip. Students can then observe their design working in hardware.

The Digilent® Company manufactures and distributes two very popular FPGA boards. Their web site is http://www.digilentinc.com. The boards are (1) the BASYS 2™ board, which contains a Spartan 3E FPGA manufactured by Xilinx, and (2) the NEXYS 2™ board, which also contains a Spartan 3E FPGA. The cost of the BASYS 2 board is about $49; the cost of the

NEXYS 2 board is about $99. If the book is used for a digital design course, either the BASYS 2 or NEXYS 2 board can be used. If the book is used for a computer design course, either the BASYS 2 or NEXYS 2 board can be used to design a stripped-down version of VBC1-E, for an added cost of about $55 for a few additional peripheral modules. The NEXYS 2 board has additional flexibility that allows an add-on board called the FX2 MIB (module interface board) to be used. This add-on board allows a full-blown version of VBC1-E to be designed with additional peripheral modules. The cost of the FX2 MIB is about $20 and the additional peripheral modules are about $40.

Information for Laboratory Experiments

Appendix A contains 34 laboratory experiments. To perform Experiments 1B through 25L requires the use of a BASYS 2 or NEXYS 2 board. These boards are quite popular and will be upgraded as newer FPGAs become available. The experiments may be performed on a different FPGA board from another vender, provided that the FPGA board has the required input/output capability, or the VHDL code is modified to match the input/output capability of the different FPGA board. The Altera® DE1 and DE2 boards can also be used to perform the experiments using Quartus-II software, because generic VHDL code is used throughout the book. The I/O for the DE1 and DE2 boards are slightly different than that of the BASYS 2 and NEXYS 2 boards, thus requiring changes to be made in the generic VHDL code to accommodate the differences in I/O capability.

Recommended pre-lab assignments are included so students can learn how to modify test bench code or write test bench code to simulate their VHDL designs. Appendix B provides students with help in how to modify test bench code.

ACKNOWLEDGMENTS

We would like to thank the many reviewers who made valuable comments during the development and writing phases of this book. We hope the final product meets with their approval.

Bharat Bhuva	Vanderbuilt University
Suresh Borkar	Illinois Institute of Technology
C. Hwa Chang	Tufts University
Katherine Compton	University of Wisconsin–Madison
Steve Crist	Western New England College
Nila Y. Desai	Sardar Vallabhbhai National Institution of Technology, India
Rahul Dubey	DA-IICT (Dhirubhai Ambani Institute of Information and Communication Technology), Gujarat India
Mark Faust	Portland State University
Maria Garazaran	University of Illinois–Urbana
Roger Haggard	Tennessee Technological University
Ronald Hayne	The Citadel
M. Nazrul Islam	Old Dominion University
Anura Jayasumana	Colorado State University
Brock LaMeres	Montana State University
Yong Li	University of Wisconsin–Platteville
Yufeng Lu	Bradley University
Aleksander Malinowski	Bradley University
Chad Mano	Utah State University
James C. Maxted	University of Iowa

David G. Meyer	Purdue University
Venkatesan Muthukumar	University of Nevada–Las Vegas
Shahin Nazarian	University of Southern California
Patricia Nava	University of Texas–El Paso
Haluk Ozemek	San Jose State University
James K. Peckol	University of Washington
Reginald Perry	Florida State University
Arvind Rajawat	Maulana Azad National Institute of Technology, Bhopal India
Jackie Rice	University of Lethbridge
Salam Salloum	California State Polytechnic University–Pomona
Martha Sloan	Michigan Technological University
James Stine	Oklahoma State University
Somanath Tripathy	Indian Institute of Technology, Patna India
Kenneth S. Vastola	Rensselaer Polytechnic University
Earl Wells	University of Alabama–Huntsville
Phillip Wilsey	University of Cincinnati

We would also like to provide a special acknowledgment to Scott Marshall, one of Richard's students, who developed and wrote Experiments 17L and 25L. Scott also wrote the memory loader program for VBC1-L and VBC1-EL. Scott did this for Digilent's BASYS 2 and NEXYS 2 boards and also for Altera's DE1 and DE2 boards. The memory loader program provides the means to automatically download the instruction bit patterns into the instruction memory of VBC1 and VBC1-E without using switches or initializing instruction memory at startup.

We would also like to thank the many dedicated and studious students at California State Polytechnic University in San Luis Obispo, in both electrical and computer engineering. Many students took an interest in our work, making suggestions and corrections as they completed the digital and computer design courses taught from our notes while the book was being developed. Without these caring students—who are too many to mention by name—it would not have been possible to write this book. Thank you.

Without the wonderful caring help of the following people at McGraw-Hill, this book would not have made it through the publication process: Raghu Srinivasan, global publisher; Peter Massar, senior sponsoring editor; Lorraine Buczek, developmental editor; and Jane Mohr, project manager. We really appreciate all your help in making this book become a reality.

We would like to thank the Digilent Company for supplying the BASYS 2 and NEXYS 2 boards, providing the technical assistance in using them, and providing the photos in Experiments 1A and 19. The people who helped us are Clint Cole, Gene Apperson, Jim O'Dell, Joe Harris, Joshua Pederson, Fiona Cole, Stephanie Roberts, Norman MacDonald, and Roy Bean.

We would like to thank the Altera® Corporation for supplying the DE1 and DE2 boards and for providing the technical assistance in using them. The people who helped us are Blair Fort and Ralene Marcoccia.

Richard would like to personally thank the Xilinx Company for allowing him to attend in-house courses on the ISE software and to work at Xilinx over the summer with their software and hardware engineers. The people who helped him are Rina Raman, Peter Alfke, and Paul Hartke. Richard would also like to thank Ken Chapman who reviewed the manuscript very early in the development of the book and made important comments. Ken is the designer of the Xilinx PicoBlaze 8-bit microcontroller. There were many more engineers who provided excellent advice and help while Richard worked at Xilinx, and sincere thanks goes out to all of them.

Last, but not least, we would like to thank the faculty in the electrical engineering department and CPE program at Cal Poly who had a part in making this book better. The people who helped us are Jim Harris, Michael Cirovic, Art MacCarley, Albert Liddicoat, Xiao-Hua (Helen) Yu, Tina Smilkstein, David Braun, Dennis Derickson, Fred DePiero, John Oliver, Wayne Pilk-

ington, Lynne Silvovsky, Bryan Mealy, John Saghri, and Hugh Smith. Some of these people taught the courses using our notes and others encouraged or discouraged us from doing certain things, which we believe made the book better.

Online Resources

A number of valuable resources are available to students and instructors at the following website: www.mhhe.com/sandige. Students may download the programs used in the text, such as the Karnaugh map explorer program, the EASY1 program, and the Memory Loader program. PowerPoint slides, solutions to the end-of-chapter problems, stimulus processes for pre-labs, and solutions to the laboratory experiments are available to instructors only.

No book is totally error free. If you find an undiscovered error, please e-mail your comments to richard@sandige.com. An up-to-date list of discovered errors will be available to all readers on the book's website.

Richard Sandige
Michael Sandige

About the Authors

Richard Sandige is a Professor Emeritus at California Polytechnic State University (Cal Poly) at San Luis Obispo, California. He taught from 1998 to 2010 at Cal Poly in the CPE (computer engineering) program. He received a PhD in electrical engineering from Texas A&M University in 1978, an MS in electrical engineering from West Virginia University in 1969, and a BS in electrical engineering from West Virginia University in 1963. He taught from 1989 to 1998 at the University of Wyoming in the Electrical Engineering Department. He worked at Hewlett-Packard (HP) Company from 1979 to 1989, where he was on the team that designed the first HP desktop computer. He taught at Texas A&M University from 1973 to 1979 in the Engineering Technology Department while working on his PhD. He taught at Southwest Virginia Community College from 1970 to 1973 in the Electronics Department. He worked at Conductron Missouri (Electronic Branch of McDonnell Douglas) from 1969 to 1970 on flight simulators. He taught at West Virginia Institute of Technology from 1967 to 1969 while working on his MS. He served in the United States Air Force from 1963 to 1966, working on the research and development of intercontinental ballistic missiles, after receiving his commission as Second Lieutenant in the Air Force Reserve Officer Training Corps at West Virginia University. While working at the University of Wyoming, he served as an assistant editor for the *IEEE Transactions on Education*. Dr. Sandige has more than 20 publications in professional journals and has published four textbooks and one lab manual.

Mike Sandige is a software architect and developer with a history of and interest in game and game technology development. He received a BS in electrical engineering and computer science in 1990 from the University of Colorado, Boulder. He started in the game industry as an independent contractor, working as the sole or lead programmer on several titles from 1987 through 1994. In 1994, he co-founded Cinematronics LLC, with the charter to make games for Microsoft Windows, an emerging platform for games. As vice president of product development, he worked on games, and as the company grew, he managed and directed the engineering staff. One of the more popular games he developed during this time was the "Space Cadet" pinball game that shipped with Windows 95. Cinematronics became a successful game development studio, leading to acquisition by Maxis (the makers of "The Sims") in 1996. In 1997, Mike Sandige joined Eclipse Entertainment, a small startup, as vice president of research and development. He worked on game engine architecture, rendering technology and helping make Eclipse Entertainment's Genesis3D game engine a leading product. He joined WildTangent when it acquired Eclipse Entertainment in 1999 to leverage the 3D engine team and technology into a new web browser–based technology, the "WildTangent Web Driver." He architected the technology and managed the technology team, as well as assisting with the development of a few of the many games that used the engine. Mike has remained with WildTangent for more than 10 years, helping the company with different technical challenges as it transitioned from a game and technology developer to a game distributer; he currently serves as a principal engineer. Mike Sandige has credits on more than 20 games and other published software titles.

Contents