Digital System Designs and Practices

Using Verilog HDL and FPGAs

Ming-Bo Lin

Companion Website

WILEY

DIGITAL SYSTEM DESIGNS AND PRACTICES

Using Verilog HDL and FPGAs

Ming-Bo Lin
Department of Electronic Engineering
National Taiwan University of Science and Technology
Taipei, Taiwan

John Wiley & Sons (Asia) Pte Ltd

Visit our Home Page on www.wiley.com

Other Wiley Editorial Offices

John Wiley & Sons, Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

John Wiley & Sons Inc., 111 River Street, Hoboken, NJ 07030, USA

Jossey-Bass, 989 Market Street, San Francisco, CA 94103-1741, USA

Wiley-VCH Verlag GmbH, Boschstr. 12, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 42 McDougall Street, Milton, Queensland 4064, Australia

John Wiley & Sons Canada Ltd, 6045 Freemont Blvd, Mississauga, ONT, L5R 4J3, Canada

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Library of Congress Cataloging-in-Publication Data

Lin, Ming-Bo.
Digital system designs and practices: using Verilog HDL and FPGAs / Ming-Bo Lin.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-470-82323-1 (cloth)
1. Digital electronics. 2. Field programmable gate arrays. 3. Verilog (Computer hardware description language) I. Title.
TK7868.D5L535 2008
621.381–dc22 2008002506

ISBN 978-0-470-82323-1 (HB)

Typeset in 10.5/12pt Times by Thomson Digital, Noida, India.
Printed and bound in Singapore by Markono Print Media Pte Ltd, Singapore.
This book is printed on acid-free paper responsibly manufactured from sustainable forestry in which at least two trees are planted for each one used for paper production.

DIGITAL SYSTEM DESIGNS AND PRACTICES

Using Verilog HDL and FPGAs

To Alfred, Fanny, Alice and Frank
and in memory of my parents

PREFACE

With the advance of the semiconductor and communication industries, the use of system-on-a-chip (SoC) has become an essential technique to decrease product costs. It has become increasingly important for electrical engineers to develop a good understanding of the key stages of hardware description language (HDL) design flow based on cell-based libraries or field-programmable gate array (FPGA) devices. This book addresses the need for teaching such a topic based on Verilog HDL and FPGAs.

The objective of this book, *Digital System Designs and Practices: Using Verilog HDL and FPGAs*, is intended to be useful both as a text for students and as a reference book for practicing engineers or a self-study book for readers. For class use, each chapter includes many worked problems and review questions for helping readers test their understanding of the context. In addition, throughout the book, an abundance of examples are provided for helping readers realize the basic features of Verilog HDL and grasp the essentials of digital system designs as well.

The contents of this book stem largely from the course *FPGA System Designs and Practices*, given at our campus over the past few years. This course is an 'undergraduate-elective' and first-year graduate course. This book is structured so that it can be used as a sequence of courses, such as *Hardware Description Language*, *FPGA System Designs and Practices*, *Digital System Designs*, *Advanced Digital System Designs*, and others.

CONTENTS OF THIS BOOK

The contents of this book can be roughly divided into four parts. The first part includes Chapters 1 to 7 and introduces the basic features and capabilities of Verilog HDL. This part can also be used as a reference for Verilog HDL. The second part covers Chapters 8 to 10 and contains basic combinational and sequential modules. In addition, the various options for implementing a digital system are discussed in detail. The third part consists of Chapters 11 to 13 and examines the three closely related topics: design, synthesis and verification. The last part considers the register-transfer level (RTL) and system level design examples and the techniques of testing and testable design. This part is composed of Chapters 14 to 16.

Chapter 1 introduces the features and capabilities of Verilog HDL and gives a tutorial example to illustrate how to use it to model a design at various levels of abstraction and in various modeling styles. In addition, we also demonstrate the use of Verilog HDL to verify a design after the description of a design is completed.

Chapter 2 deals with how to model a design in structural style. In this style, a module is described as a set of interconnected components, which include modules, user-defined primitives (UDPs), gate primitives and switch primitives. In this chapter, we introduce the structural modeling at the gate and switch levels. The UDPs and modules are dealt with separately in Chapters 5 and 6.

Chapter 3 describes the essentials of dataflow modeling style. In this modeling style, the most basic statement is the continuous assignment, which consists of operators and operands in turn. The continuous assignment continuously drives a value onto a net and is usually used to model a combinational logic.

Chapter 4 is concerned with the behavioral modeling style, which provides users with the capability of modeling a design in a way like that of most high-level programming languages. In this modeling style, the most common statements include procedural assignments, selection statements and iterative (loop) statements. In addition, timing controls are also dealt with in detail.

Chapter 5 describes three additional behavioral ways provided by Verilog HDL which are widely used to model designs. These include tasks, functions and user-defined primitives (UDPs). Tasks and functions provide the ability to re-use the same piece of code from many places in a design and UDPs provide a means to model a design with a truth table. In addition, the predefined system tasks and functions are introduced. These system tasks and functions are useful when modeling, abstractly, a design in behavioral style or writing test benches for designs.

Chapter 6 discusses three closely related issues of hierarchical structural modeling. These include instantiations, `generate` statements and configurations. The instantiation is the mechanism through which the hierarchical structure is formed by modules being embedded into other modules. The `generate` statements can conditionally generate declarations and instantiations into a design. By using configurations, we may specify a new set of target libraries so as to change the mapping of a design without having to change the source description.

Chapter 7 deals with the additional features of Verilog HDL. These features include block constructs, procedural continuous assignments, specify blocks, timing checks and compiler directives.

Chapters 8 and 9 examine some basic combinational and sequential modules that are often used as basic building blocks to construct a complex design. In particular, these modules are the basic building blocks of a datapath when using the datapath and controller approach in a complex design.

Chapter 8 is concerned with the most commonly used combinational logic modules, which include encoders and decoders, multiplexers and demultiplexers, and magnitude comparators. In addition, a multiplexing-driven seven-segment light-emitting diode (LED) display system which combines the use of a decoder, as well as a multiplexer, is discussed in detail.

Chapter 9 examines several basic sequential modules that are widely used in digital systems. These include flip-flops, synchronizers, a switch-debouncing circuit, registers, data registers, register files, shift registers, counters (binary, BCD, Johnson), CRC generators and detectors, clock generators and pulse generators, as well as timing generators.

Chapter 10 describes various design options of digital systems. These options include application-specific integrated circuits (ASICs) and field-programmable devices. ASICs are devices that must be fabricated in IC foundries and can be designed with one of the following: full-custom, cell-based and gate-array-based approaches.

Field-programmable devices are the ones that can be personalized in laboratories and include programmable logic devices (PLDs), complex PLDs (CPLDs) and field-programmable gate arrays (FPGAs). In addition, the issues of interfacing two logic modules or devices with different logic levels and power-supply voltages are also dealt with in detail in this chapter.

The next three chapters consider three closely related issues: design, synthesis and verification. Chapter 11 introduces two useful techniques by which a system can be designed. These techniques include the finite-state machine (FSM) and register-transfer level (RTL) design approaches. The former may be described by using a state diagram or an algorithmic state machine (ASM) chart; the latter may be described by an ASM chart or by using the datapath and controller (DP+CU) paradigm. For a simple system, a three-step paradigm introduced in the chapter may be used to derive the datapath and controller of a design from its ASM chart. For complex systems, their datapaths and controllers are often derived from specifications in a state-of-the-art manner. An example of displaying four-digit data on a commercial dot-matrix liquid-crystal display (LCD) module is used to illustrate this approach. In this chapter, we also emphasize the concept that a hardware algorithm can usually be realized by using either a multiple-cycle or a single-cycle structure. The choice is based on the tradeoff among area (hardware cost), performance (operating frequency or propagation delay) and power consumption.

Chapter 12 is concerned with the principles of logic synthesis and the general architecture of synthesis tools. The function of logic synthesis is to transform an RTL representation into gate-level netlists. In order to make good use of synthesis tools, we need to provide the design environment and design constraints along with an RTL code and technology library. Moreover, we give some guidelines about how to write a good Verilog HDL code such that it can be accepted by most logic synthesis tools and can achieve the best compile times and synthesis results. These guidelines also include clock signals, reset signals and how to partition a design.

Verification is a necessary process that makes sure a design can meet its specifications both in function and timing. Chapter 13 deals with this issue in more detail and gives a comprehensive example based on FPGA design flow to illustrate how to enter, synthesize, implement and configure the underlying FPGA device of a design. Along with the design flow, static timing analyses are also given and explained. In addition, design verification through dynamic timing simulations, incorporating the delays of logic elements and interconnect, is introduced.

The next two chapters are concerned with more complex modules. Chapter 14 examines many frequently used arithmetic modules, including addition, multiplication, division, ALU, shift and two digital-signal processing (DSP) filters as well. Along with these arithmetic operations and their algorithms, we also re-emphasize the concept that a hardware algorithm can often be realized by using either a multiple-cycle or a single-cycle structure.

Chapter 15 describes the design of a small μC system, which is the most complex design example in the book. This system includes a general-purpose input and output (GPIO), timers and a universal asynchronous receiver and transmitter (UART) being connected by a system bus composed of an address bus and a data bus, as well as a control bus. The 16-bit CPU provides 27 instructions and 7 addressing modes.

The final chapter is concerned with the topic of testability and testable design. Testing is the only way to ensure that a system or a circuit may function properly. The goal of testing is to find any existing faults in a system or a circuit. In this chapter, we examine fault models, test vector generations, testable circuit design or design for testability. In addition, system-level testing, such as SRAM, a core-based system and system-on-a-chip (SoC), are also briefly dealt with.

Appendix A contains a complete syntax reference of Verilog HDL, including the keywords and formal definition of the Verilog-2001 standard in Backus-Naur Form (BNF).

SUPPLEMENTS

Two important and useful supplements are available for this book at the following URL: www.wiley.com/go/mblin. The first is student supplements, including source files of Verilog HDL examples in the book and the pdf files of lecture notes. The second is the instructor's supplements, containing figures, a solution manual and lecture notes in power-point files, in addition to the student supplements.

STUDENT PROJECTS

Many end-of-chapter problems may be assigned as student projects, in particular, the problems of Chapters 11, 14 and 15. Of course, many other chapters may also contain problems that may be used for the same purpose.

ACKNOWLEDGMENTS

Most material of this book has been taken from the course ET5009 offered at the National Taiwan University of Science and Technology over the past few years. My thanks go to the students of this course, who suffered through many of the experimental class offerings based on the draft of this book. Valuable comments from the participants of the course have helped in evolving the contents of this book and are greatly appreciated. Thanks to my mentor, Ben Chen, who is also a cofounder of the Chuan Hwa Book Company, who brought me into this colorful digital world about thirty years ago. In addition, he has also kindly allowed me to freely use figures, tables and even parts of material from my earlier Chinese Books, published by the Chuan Hwa Book Company, in this current book. Without this permission, it would have needed much more time to prepare the manuscript of this book. Finally but not least, I would like to thank my children, Alice and Frank, and my wife, Fanny, for their patience in enduring my absence from them during the writing of this book. I am also grateful to the publisher's staff for their support, encouragement and willingness to give prompt assistance during this book project.

Ming-Bo Lin
Taipei, Taiwan

CONTENTS