

PREFACE

This book is intended for an introductory course in digital logic design, which is a basic course in most electrical and computer engineering programs. A successful designer of digital logic circuits needs a good understanding of basic concepts and a firm grasp of computer-aided design (CAD) tools. The purpose of our book is to provide the desirable balance between teaching the basic concepts and practical application through CAD tools. To facilitate the learning process, the necessary CAD software is included as an integral part of the book package.

The main goals of the book are (1) to teach students the fundamental concepts in classical manual digital design and (2) illustrate clearly the way in which digital circuits are designed today, using CAD tools. Even though modern designers no longer use manual techniques, except in rare circumstances, our motivation for teaching such techniques is to give students an intuitive feeling for how digital circuits operate. Also, the manual techniques provide an illustration of the types of manipulations performed by CAD tools, giving students an appreciation of the benefits provided by design automation. Throughout the book, basic concepts are introduced by way of examples that involve simple circuit designs, which we perform using both manual techniques and modern CAD-tool-based methods. Having established the basic concepts, more complex examples are then provided, using the CAD tools. Thus our emphasis is on modern design methodology to illustrate how digital design is carried out in practice today.

TECHNOLOGY AND CAD SUPPORT

The book discusses modern digital circuit implementation technologies. The emphasis is on programmable logic devices (PLDs), which is the most appropriate technology for use in a textbook for two reasons. First, PLDs are widely used in practice and are suitable for almost all types of digital circuit designs. In fact, students are more likely to be involved in PLD-based designs at some point in their careers than in any other technology. Second, circuits are implemented in PLDs by end-user programming. Therefore, students can be provided with an opportunity, in a laboratory setting, to implement the book's design examples in actual chips. Students can also simulate the behavior of their designed circuits on their own computers. We use the two most popular types of PLDs for targeting of designs: complex programmable logic devices (CPLDs) and field-programmable gate arrays (FPGAs).

Our CAD support is based on Altera Quartus II software. Quartus II provides automatic mapping of a design into Altera CPLDs and FPGAs, which are among the most widely used PLDs in the industry. The features of Quartus II that are particularly attractive for our purposes are:

- It is a commercial product. The version included with the book supports all major features of the product. Students will be able to easily enter a design into the CAD

system, compile the design into a selected device (the choice of device can be changed at any time and the design retargeted to a different device), simulate the functionality and detailed timing of the resulting circuit, and if laboratory facilities are provided at the student's school, implement the designs in actual devices.

- It provides for design entry using both hardware description languages (HDLs) and schematic capture. In the book, we emphasize the HDL-based design because it is the most efficient design method to use in practice. We describe in detail the IEEE Standard Verilog HDL language and use it extensively in examples. The CAD system included with the book has a Verilog compiler, which allows the student to automatically create circuits from the Verilog code and implement these circuits in real chips.
- It can automatically target a design to various types of devices. This feature allows us to illustrate the ways in which the architecture of the target device affects a designer's circuit.
- It can be used on most types of popular computers. The version of Quartus II provided with the book runs on computers using Microsoft Windows. However, through Altera's university program the software is also available for other machines, such as SUN or HP workstations.

A Quartus II CD-ROM is included with each copy of the book. Use of the software is fully integrated into the book so that students can try, firsthand, all design examples. To teach the students how to use this software, the book includes three, progressively advanced, hands-on tutorials.

SCOPE OF THE BOOK

Chapter 1 provides a general introduction to the process of designing digital systems. It discusses the key steps in the design process and explains how CAD tools can be used to automate many of the required tasks.

Chapter 2 introduces the basic aspects of logic circuits. It shows how Boolean algebra is used to represent such circuits. It also gives the reader a first glimpse at Verilog, as an example of a hardware description language that may be used to specify the logic circuits.

The electronic aspects of digital circuits are presented in Chapter 3. This chapter shows how the basic gates are built using transistors and presents various factors that affect circuit performance. The emphasis is on the latest technologies, with particular focus on CMOS technology and programmable logic devices.

Chapter 4 deals with the synthesis of combinational circuits. It covers all aspects of the synthesis process, starting with an initial design and performing the optimization steps needed to generate a desired final circuit. It shows how CAD tools are used for this purpose.

Chapter 5 concentrates on circuits that perform arithmetic operations. It begins with a discussion of how numbers are represented in digital systems and then shows how such numbers can be manipulated using logic circuits. This chapter illustrates how Verilog can be used to specify the desired functionality and how CAD tools provide a mechanism for developing the required circuits. We chose to introduce the number representations at this point, rather than in the very beginning of the book, to make the discussion more mean-

ingful and interesting, because we can immediately provide examples of how numerical information may be processed by actual circuits.

Chapter 6 presents combinational circuits that are used as building blocks. It includes the encoder, decoder, and multiplexer circuits. These circuits are very convenient for illustrating the application of many Verilog constructs, giving the reader an opportunity to discover more advanced features of Verilog.

Storage elements are introduced in Chapter 7. The use of flip-flops to realize regular structures, such as shift registers and counters, is discussed. Verilog-specified designs of these structures are included. The chapter also shows how larger systems, such as a simple processor, may be designed.

Chapter 8 gives a detailed presentation of synchronous sequential circuits (finite state machines). It explains the behavior of these circuits and develops practical design techniques for both manual and automated design.

Asynchronous sequential circuits are discussed in Chapter 9. While this treatment is not exhaustive, it provides a good indication of the main characteristics of such circuits. Even though the asynchronous circuits are not used extensively in practice, they should be studied because they provide an excellent vehicle for gaining a deeper understanding of the operation of digital circuits in general. They illustrate the consequences of propagation delays and race conditions that may be inherent in the structure of a circuit.

Chapter 10 is a discussion of a number of practical issues that arise in the design of real systems. It highlights problems often encountered in practice and indicates how they can be overcome. Examples of larger circuits illustrate a hierarchical approach in designing digital systems. Complete Verilog code for these circuits is presented.

Chapter 11 introduces the topic of testing. A designer of logic circuits has to be aware of the need to test circuits and should be conversant with at least the most basic aspects of testing.

Chapter 12 presents a complete CAD flow that the designer experiences when designing, implementing, and testing a digital circuit.

Appendix A provides a complete summary of Verilog features. Although use of Verilog is integrated throughout the book, this appendix provides a convenient reference that the reader can consult from time to time when writing Verilog code.

Appendices B, C, and D contain a sequence of tutorials on the Quartus II CAD tools. This material is suitable for self-study; it shows the student in a step-by-step manner how to use the CAD software provided with the book.

Appendix E gives detailed information about the devices used in illustrative examples.

WHAT CAN BE COVERED IN A COURSE

All the material in the book can be covered in 2 one-quarter courses. A good coverage of the most important material can be achieved in a single one-semester, or even a one-quarter, course. This is possible only if the instructor does not spend too much time teaching the intricacies of Verilog and CAD tools. To make this approach possible, we organized the Verilog material in a modular style that is conducive to self-study. Our experience in teaching different classes of students at the University of Toronto shows that the instructor

may spend only 2 to 3 lecture hours on Verilog, concentrating mostly on the specification of sequential circuits. The Verilog examples given in the book are largely self-explanatory, and students can understand them easily. Moreover, the instructor need not teach how to use the CAD tools, because the Quartus II tutorials in Appendices B, C, and D are suitable for self-study.

The book is also suitable for a course in logic design that does not include exposure to Verilog. However, some knowledge of Verilog, even at a rudimentary level, is beneficial to the students, and it is a great preparation for a job as a design engineer.

One-Semester Course

A natural starting point for formal lectures is Chapter 2. The material in Chapter 1 is a general introduction that serves as a motivation for why logic circuits are important and interesting; students can read and understand this material easily.

The following material should be covered in lectures:

- Chapter 2—all sections.
- Chapter 3—sections 3.1 to 3.7. Also, it is useful to cover sections 3.8 and 3.9 if the students have some basic knowledge of electrical circuits.
- Chapter 4—sections 4.1 to 4.7 and section 4.12.
- Chapter 5—sections 5.1 to 5.5.
- Chapter 6—all sections.
- Chapter 7—all sections.
- Chapter 8—sections 8.1 to 8.9.

If time permits, it would also be very useful to cover sections 9.1 to 9.3 and section 9.6 in Chapter 9, as well as one or two examples in Chapter 10.

One-Quarter Course

In a one-quarter course the following material can be covered:

- Chapter 2—all sections.
- Chapter 3—sections 3.1 to 3.3.
- Chapter 4—sections 4.1 to 4.5 and section 4.12.
- Chapter 5—sections 5.1 to 5.3 and section 5.5.
- Chapter 6—all sections.
- Chapter 7—sections 7.1 to 7.10 and section 7.13.
- Chapter 8—Sections 8.1 to 8.5.

A MORE TRADITIONAL APPROACH

The material in Chapters 2 and 4 introduces Boolean algebra, combinational logic circuits, and basic minimization techniques. Chapter 2 provides initial exposure to these topics using

only AND, OR, NOT, NAND, and NOR gates. Then Chapter 3 discusses the implementation technology details, before proceeding with the synthesis techniques and other types of gates in Chapter 4. The material in Chapter 4 is appreciated better if students understand the technological reasons for the existence of NAND, NOR, and XOR gates, and the various programmable logic devices.

An instructor who favors a more traditional approach may cover Chapters 2 and 4 in succession. To understand the use of NAND, NOR, and XOR gates, it is necessary only that the instructor provide a functional definition of these gates.

VERILOG

Verilog is a complex language, which some instructors feel is too hard for beginning students to grasp. We fully appreciate this issue and have attempted to solve it. It is not necessary to introduce the entire Verilog language. In the book we present the important Verilog constructs that are useful for the design and synthesis of logic circuits. Many other language constructs, such as those that have meaning only when using the language for simulation purposes, are omitted. The Verilog material is introduced gradually, with more advanced features being presented only at points where their use can be demonstrated in the design of relevant circuits.

The book includes more than 150 examples of Verilog code. These examples illustrate how Verilog is used to describe a wide range of logic circuits, from those that contain only a few gates to those that represent digital systems such as a simple processor.

SOLVED PROBLEMS

The chapters include examples of solved problems. They show how typical homework problems may be solved.

HOMEWORK PROBLEMS

More than 400 homework problems are provided in the book. Answers to selected problems are given at the back of the book. Solutions to all problems are available to instructors in the *Solutions Manual* that accompanies the book.

LABORATORY

The book can be used for a course that does not include laboratory exercises, in which case students can get useful practical experience by simulating the operation of their designed circuits by using the CAD tools provided with the book. If there is an accompanying laboratory, then a number of design examples in the book are suitable for laboratory experiments.

Instructors can access the Solutions Manual and the PowerPoint slides (containing all figures in the book) at:

www.mhhe.com/brownverilog

ACKNOWLEDGMENTS

We wish to express our thanks to the people who have helped during the preparation of the book. Kelly Chan helped with the technical preparation of the manuscript. Dan Vranesic produced a substantial amount of artwork. He and Deshanand Singh also helped with the preparation of the solutions manual. Tom Czajkowski helped in checking the answers to some problems. Jonathan Rose provided helpful suggestions for improving the treatment of timing issues. The reviewers, William Barnes, New Jersey Institute of Technology; Thomas Bradicich, North Carolina State University; James Clark, McGill University; Stephen DeWeerth, Georgia Institute of Technology; Clay Gloster, Jr., North Carolina State University (Raleigh); Carl Hamacher, Queen's University; Vincent Heuring, University of Colorado; Yu Hen Hu, University of Wisconsin; Wei-Ming Lin, University of Texas (Austin); Wayne Loucks, University of Waterloo; Chris Myers, University of Utah; Vojin Oklobdzija, University of California (Davis); James Palmer, Rochester Institute of Technology; Gandhi Puvvada, University of Southern California; Teodoro Robles, Milwaukee School of Engineering; Tatyana Roziner, Boston University; Rob Rutenbar, Carnegie Mellon University; Eric Schwartz, University of Florida; Wen-Tsong Shiue, Oregon State University; Charles Silio, Jr., University of Maryland; Scott Smith, University of Missouri (Rolla); Arun Somani, Iowa State University; Bernard Svihel, University of Texas (Arlington); and Zeljko Zilic, McGill University provided constructive criticism and made numerous suggestions for improvements.

We are grateful to the Altera Corporation for providing the Quartus II system, especially to Misha Burich. The support of McGraw-Hill people has been exemplary. We truly appreciate the help of Michael Hackett, Brenda Rolwes, Darlene Schueller, April Southwood, Kris Tibbetts, Judi David, and Michael Weitz.

Stephen Brown and Zvonko Vranesic