Introduction to Logic Circuits & Logic Design with Verilog

Introduction to Logic Circuits & Logic Design with Verilog

1st Edition

Brock J. LaMeres

Springer

Brock J. LaMeres
Department of Electrical & Computer Engineering
Montana State University
Bozeman, MT, USA

ISBN 978-3-319-53882-2 ISBN 978-3-319-53883-9 (eBook)
DOI 10.1007/978-3-319-53883-9

Library of Congress Control Number: 2017932539

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The purpose of this new book is to fill a void that has appeared in the instruction of digital circuits over the past decade due to the rapid abstraction of system design. Up until the mid-1980s, digital circuits were designed using *classical* techniques. Classical techniques relied heavily on manual design practices for the synthesis, minimization, and interfacing of digital systems. Corresponding to this design style, academic textbooks were developed that taught classical digital design techniques. Around 1990, large-scale digital systems began being designed using hardware description languages (HDLs) and automated synthesis tools. Broad-scale adoption of this *modern design* approach spread through the industry during this decade. Around 2000, hardware description languages and the modern digital design approach began to be taught in universities, mainly at the senior and graduate level. There were a variety of reasons that the modern digital design approach did not penetrate the lower levels of academia during this time. First, the design and simulation tools were difficult to use and overwhelmed freshman and sophomore students. Second, the ability to implement the designs in a laboratory setting was infeasible. The modern design tools at the time were targeted at custom integrated circuits, which are cost and time prohibitive to implement in a university setting. Between 2000 and 2005, rapid advances in programmable logic and design tools allowed the modern digital design approach to be implemented in a university setting, even in lower level courses. This allowed students to learn the modern design approach based on HDLs and prototype their designs in real hardware, mainly field programmable gate arrays (FPGAs). This spurred an abundance of textbooks to be authored teaching hardware description languages and higher levels of design abstraction. This trend has continued until today. While abstraction is a critical tool for engineering design, the rapid movement toward teaching only the modern digital design techniques has left a void for freshman and sophomore level courses in digital circuitry. Legacy textbooks that teach the classical design approach are outdated and do not contain sufficient coverage of HDLs to prepare the students for follow-on classes. Newer textbooks that teach the modern digital design approach move immediately into high-level behavioral modeling with minimal or no coverage of the underlying hardware used to implement the systems. As a result, students are not being provided the resources to understand the fundamental hardware theory that lies beneath the modern abstraction such as interfacing, gate level implementation, and technology optimization. Students moving too rapidly into high levels of abstraction have little understanding of what is going on when they click the “compile & synthesize” button of their design tool. This leads to graduates who can model a breadth of different systems in an HDL, but have no depth into how the system is implemented in hardware. This becomes problematic when an issue arises in a real design, and there is no foundational knowledge for the students to fall back on in order to debug the problem.

This new book addresses the lower level foundational void by providing a comprehensive, bottoms-up, coverage of digital systems. This book begins with a description of lower level hardware including binary representations, gate-level implementation, interfacing, and simple combinational logic design. Only after a foundation has been laid in the underlying hardware theory is the Verilog language introduced. The Verilog introduction gives only the basic concepts of the language in order to model, simulate, and synthesize combinational logic. This allows the students to gain familiarity with the language and the modern design approach without getting overwhelmed by the full capability of the language. This book then covers sequential logic and finite state machines at the structural level. Once this secondary foundation has been laid, the remaining capabilities of Verilog are presented that allow sophisticated, synchronous systems to be modeled. An entire chapter is then dedicated to examples of sequential system modeling, which allows the students to learn by example. The second part of this textbook introduces the details of programmable logic, semiconductor memory, and arithmetic circuits. This book culminates with a discussion of computer system design, which incorporates all of the knowledge gained in the previous chapters. Each component of a computer system is described with an accompanying Verilog implementation, all while continually reinforcing the underlying hardware beneath the HDL abstraction.

Written the Way It Is Taught

The organization of this book is designed to follow the way in which the material is actually learned. Topics are presented only once sufficient background has been provided by earlier chapters to fully understand the material. An example of this *learning-oriented* organization is how the Verilog language is broken into two chapters. Chapter 5 presents an introduction to Verilog and the basic constructs to model combinational logic. This is an ideal location to introduce the language because the reader has just learned about combinational logic theory in Chap. 4. This allows the student to begin gaining experience using the Verilog simulation tools on basic combinational logic circuits. The more advanced constructs of Verilog such as sequential modeling and test benches are presented in Chap. 8 only after a thorough background in sequential logic is presented in Chap. 7. Another example of this learning-oriented approach is how arithmetic circuits are not introduced until Chap. 12. While technically the arithmetic circuits in Chap. 12 are combinational logic circuits and could be presented in Chap. 4, the student does not have the necessary background in Chap. 4 to fully understand the operation of the arithmetic circuitry so its introduction is postponed.

This incremental, *just-in-time* presentation of material allows the book to follow the way the material is actually taught in the classroom. This design also avoids the need for the instructor to assign sections that move back-and-forth through the text. This not only reduces course design effort for the instructor but allows the student to know where they are in the sequence of learning. At any point, the student should know the material in prior chapters and be moving toward understanding the material in subsequent ones.

An additional advantage of this book's organization is that it supports giving the student hands-on experience with digital circuitry for courses with an accompanying laboratory component. The flow is designed to support lab exercises that begin using discrete logic gates on a breadboard and then move into HDL-based designs implemented on off-the-shelf FPGA boards. Using this approach to a laboratory experience gives the student experience with the basic electrical operation of digital circuits, interfacing, and HDL-based designs.

Learning Outcomes

Each chapter begins with an explanation of its learning objective followed by a brief preview of the chapter topics. The specific learning outcomes are then presented for the chapter in the form of concise statements about the measurable knowledge and/or skills the student will possess by the end of the chapter. Each section addresses a single, specific learning outcome. This eases the process of assessment and gives specific details on student performance. There are 600+ exercise problems and concept check questions for each section tied directly to specific learning outcomes for both formative and summative assessment.

Teaching by Example

With over 200 worked examples, concept checks for each section, 200+ supporting figures, and 600+ exercise problems, students are provided with multiple ways to learn. Each topic is described in a clear, concise written form with accompanying figures as necessary. This is then followed by annotated worked examples that match the form of the exercise problems at the end of each chapter. Additionally, concept check questions are placed at the end of each section in this book to measure the student's general understanding of the material using a concept inventory assessment style. These features provide the student multiple ways to learn the material and build an understanding of digital circuitry.

Course Design

This book can be used in multiple ways. The first is to use the book to cover two, semester-based college courses in digital logic. The first course in this sequence is an *introduction to logic circuits* and covers Chaps. 1, 2, 3, 4, 5, 6, and 7. This introductory course, which is found in nearly all accredited electrical and computer engineering programs, gives students a basic foundation in digital hardware and interfacing. Chapters 1, 2, 3, 4, 5, 6 and 7 only cover relevant topics in digital circuits to make room for a thorough introduction to Verilog. At the end of this course, students have a solid foundation in digital circuits and are able to design and simulate Verilog models of concurrent and hierarchical systems. The second course in this sequence covers *logic design* using Chaps. 8, 9, 10, 11, 12, and 13. In this second course, students learn the advanced features of Verilog such as procedural assignments, sequential behavioral modeling, system tasks, and test benches. This provides the basis for building larger digital systems such as registers, finite state machines, and arithmetic circuits. Chapter 13 brings all of the concepts together through the design of a simple 8-bit computer system that can be simulated and implemented using many off-the-shelf FPGA boards.

This book can also be used in a more accelerated digital logic course that reaches a higher level of abstraction in a single semester. This is accomplished by skipping some chapters and moving quickly through others. In this use model, it is likely that Chap. 2 on numbers systems and Chap. 3 on digital circuits would be quickly referenced but not covered in detail. Chapters 4 and 7 could also be covered quickly in order to move rapidly into Verilog modeling without spending significant time looking at the underlying hardware implementation. This approach allows a higher level of abstraction to be taught but provides the student with the reference material so that they can delve in the details of the hardware implementation if interested.

All exercise and concept problems that do not involve a Verilog model are designed so that they can be implemented as a multiple choice or numeric entry question in a standard course management system. This allows the questions to be automatically graded. For the Verilog design questions, it is expected that the students will upload their Verilog source files and screenshots of their simulation waveforms to the course management system for manual grading by the instructor or teaching assistant.

Instructor Resources

Instructors adopting this book can request a solution manual that contains a graphic-rich description of the solutions for each of the 600+ exercise problems. Instructors can also receive the Verilog solutions and test benches for each Verilog design exercise. A complementary lab manual has also been developed to provide additional learning activities based on both the 74HC discrete logic family and an off-the-shelf FPGA board. This manual is provided separately from the book in order to support the ever-changing technology options available for laboratory exercises.

Bozeman, MT, USA　　　　Brock J. LaMeres

Acknowledgments

Dr. LaMeres is eternally grateful to his family for their support of this project. To JoAnn, your love and friendship makes everything possible. To Alexis, your kindness and caring brings joy to my heart. To Kylie, your humor and spirit fills me with laughter and pride. Thank you so much.

Dr. LaMeres would also like to thank the 400+ engineering students at Montana State University that helped proof read this book in preparation for the first edition.

Contents