Preface

Since the fourth edition of *Digital Design*, the commercial availability of devices using digital technology to receive, manipulate, and transmit information seems to have exploded. Cell phones and handheld devices of various kinds offer new, competing features almost daily. Underneath the attractive graphical user interface of all of these devices sits a digital system that processes data in a binary format. The theoretical foundations of these systems have not changed much; indeed, one could argue that the stability of the core theory, coupled with modern design tools, has promoted the widespread response of manufacturers to the opportunities of the marketplace. Consequently, our refinement of our text has been guided by the need to equip our graduates with a solid understanding of digital machines and to introduce them to the methodology of modern design.

This edition of *Digital Design* builds on the previous four editions, and the feedback of the team of reviewers who helped set a direction for our presentation. The focus of the text has been sharpened to more closely reflect the content of a foundation course in digital design and the mainstream technology of today's digital systems: CMOS circuits. The intended audience is broad, embracing students of computer science, computer engineering, and electrical engineering. The key elements that the book focuses include (1) Boolean logic, (2) logic gates used by designers, (3) synchronous finite state machines, and (4) datapath controller design—all from a perspective of designing digital systems. This focus led to elimination of material more suited for a course in electronics. So the reader will not find here content for asynchronous machines or descriptions of bipolar transistors. Additionally, the widespread availability of web-based ancillary material prompted us to limit our discussion of field programmable gate arrays (FPGAs) to an introduction of devices offered by only one manufacturer, rather than two. Today's designers rely heavily on hardware description languages

(HDLs), and this edition of the book gives greater attention to their use and presents what we think is a clear development of a design methodology using the Verilog HDL.

MULTI-MODAL LEARNING

Digital Design supports a multimodal approach to learning. The so-called VARK characterization of learning modalities identifies four major modes by which humans learn: (V) visual, (A) aural, (R) reading, and (K) kinesthetic. In hindsight, we note that the relatively high level of illustrations and graphical content of our text addresses the visual (V) component of VARK; discussions and numerous examples address the reading (R) component. Students who exploit the availability of free simulators to work assignments are led through a kinesthetic (K) learning experience, including the positive feedback and delight of designing a logic system that works. The remaining element of VARK, the aural/auditory (A) experience, is left to the instructor. We have provided an abundance of material and examples to support classroom lectures. Thus, a course in digital design, using *Digital Design*, can provide a rich, balanced learning experience and address all the modes identified by VARK.

For those who might still question the presentation and use of HDLs in a first course in digital design, we note that industry has largely abandoned schematic-based design entry, a style which emerged in the 1980s, during the nascent development of CAD tools for integrated circuit (IC) design. Schematic entry creates a representation of functionality that is implicit in the layout of the schematic. Unfortunately, it is difficult for anyone in a reasonable amount of time to determine the functionality represented by the schematic of a logic circuit without having been instrumental in its construction, or without having additional documentation expressing the design intent. Consequently, industry has migrated to HDLs (e.g., Verilog) to describe the functionality of a design and to serve as the basis for documenting, simulating, testing, and synthesizing the hardware implementation of the design in a standard cell-based ASIC or an FPGA. The utility of a schematic depends on the careful, detailed documentation of a carefully constructed hierarchy of design modules. In the old paradigm, designers relied upon their years of experience to create a schematic of a circuit to implement functionality. In today's design flow, designers using HDLs can express functionality directly and explicitly, without years of accumulated experience, and use synthesis tools to generate the schematic as a by-product, automatically. Industry practices arrived here because schematic entry dooms us to inefficiency, if not failure, in understanding and designing large, complex ICs.

We note, again in this edition, that introducing HDLs in a first course in designing digital circuits is not intended to replace fundamental understanding of the building blocks of such circuits or to eliminate a discussion of manual methods of design. It is still essential for a student to understand *how hardware works*. Thus, we retain a thorough treatment of combinational and sequential logic devices. Manual design practices are presented, and their results are compared with those obtained with a HDL-based paradigm. What we are presenting, however, is an emphasis on *how hardware is designed*, to better prepare a student for a career in today's industry, where HDL-based design practices are dominant.

FLEXIBILITY

The sequence of topics in the text can accommodate courses that adhere to traditional, manual-based, treatments of digital design, courses that treat design using an HDL, and courses that are in transition between or blend the two approaches. Because modern synthesis tools automatically perform logic minimization, Karnaugh maps and related topics in optimization can be presented at the beginning of a treatment of digital design, or they can be presented after circuits and their applications are examined and simulated with an HDL. The text includes both manual and HDL-based design examples. Our end-of-chapter problems further facilitate this flexibility by cross referencing problems that address a traditional manual design task with a companion problem that uses an HDL to accomplish the task. Additionally, we link the manual and HDL-based approaches by presenting annotated results of simulations in the text, in answers to selected problems at the end of the text, and in the solutions manual.

NEW TO THIS EDITION

This edition of *Digital Design* uses the latest features of IEEE Standard 1364, but only insofar as they support our pedagogical objectives. The revisions and updates to the text include:

- Elimination of specialized circuit-level content not typically covered in a first course in logic circuits and digital design (e.g., RTL, DTL, and emitter-coupled logic circuits)
- Addition of "Web Search Topics" at the end of each chapter to point students to additional subject matter available on the web
- Revision of approximately one-third of the problems at the end of the chapters
- A printed solution manual for entire text, including all new problems
- Streamlining of the discussion of Karnaugh maps
- Integration of treatment of basic CMOS technology with treatment of logic gates
- Inclusion of an appendix introducing semiconductor technology

DESIGN METHODLOGY

This text presents a systematic methodology for designing a state machine to control the datapath of a digital system. Moreover, the framework in which this material is presented treats the realistic situation in which status signals from the datapath are used by the controller, i.e., the system has feedback. Thus, our treatment provides a foundation for designing complex and interactive digital systems. Although it is presented with an emphasis on HDL-based design, the methodology is also applicable to manual-based approaches to design.

JUST ENOUGH HDL

We present only those elements of the Verilog language that are matched to the level and scope of this text. Also, correct syntax does not guarantee that a model meets a functional specification or that it can be synthesized into physical hardware. So, we introduce students to a disciplined use of industry-based practices for writing models to ensure that a behavioral description can be synthesized into physical hardware, and that the behavior of the synthesized circuit will match that of the behavioral description. Failure to follow this discipline can lead to software race conditions in the HDL models of such machines, race conditions in the test bench used to verify them, and a mismatch between the results of simulating a behavioral model and its synthesized physical counterpart. Similarly, failure to abide by industry practices may lead to designs that simulate correctly, but which have hardware latches that are introduced into the design accidentally as a consequence of the modeling style used by the designer. The industry-based methodology we present leads to race-free and latch-free designs. It is important that students learn and follow industry practices in using HDL models, independent of whether a student's curriculum has access to synthesis tools.

VERIFICATION

In industry, significant effort is expended to verify that the functionality of a circuit is correct. Yet not much attention is given to verification in introductory texts on digital design, where the focus is on design itself, and testing is perhaps viewed as a secondary undertaking. Our experience is that this view can lead to premature "high-fives" and declarations that "the circuit works beautifully." Likewise, industry gains repeated returns on its investment in an HDL model by ensuring that it is readable, portable, and reusable. We demonstrate naming practices and the use of parameters to facilitate reusability and portability. We also provide test benches for all of the solutions and exercises to (1) verify the functionality of the circuit, (2) underscore the importance of thorough testing, and (3) introduce students to important concepts, such as self-checking test benches. Advocating and illustrating the development of a *test plan* to guide the development of a test bench, we introduce test plans, albeit simply, in the text and expand them in the solutions manual and in the answers to selected problems at the end of the text.

HDL CONTENT

We have ensured that all examples in the text and all answers in the solution manual conform to accepted industry practices for modeling digital hardware. As in the previous edition, HDL material is inserted in separate sections so that it can be covered or skipped as desired, does not diminish treatment of manual-based design, and does not dictate the sequence of presentation. The treatment is at a level suitable for beginning students who are learning digital circuits and a HDL at the same time. The text prepares

students to work on signficant independent design projects and to succeed in a later course in computer architecture and advanced digital design.

Instructor Resources

Instructors can download the following classroom-ready resources from the publisher's website for the text (www.pearsonhighered.com/mano):

- Source code and test benches for all Verilog HDL examples in the test
- All figures and tables in the text
- Source code for all HDL models in the solutions manual
- A downloadable solutions manual with graphics suitable for classroom presentation

HDL Simulators

The Companion Website identifies web URLs to two simulators provided by Synapti-CAD. The first simulator is *VeriLogger Pro*, a traditional Verilog simulator that can be used to simulate the HDL examples in the book and to verify the solutions of HDL problems. This simulator accepts the syntax of the IEEE-1995 standard and will be useful to those who have legacy models. As an interactive simulator, *Verilogger Extreme* accepts the syntax of IEEE-2001 as well as IEEE-1995, allowing the designer to simulate and analyze design ideas before a complete simulation model or schematic is available. This technology is particularly useful for students because they can quickly enter Boolean and *D* flip-flop or latch input equations to check equivalency or to experiment with flip-flops and latch designs. Students can access the Companion Website at www.pearsonhighered.com/mano.

Chapter Summary

The following is a brief summary of the topics that are covered in each chapter.

Chapter 1 presents the various binary systems suitable for representing information in digital systems. The binary number system is explained and binary codes are illustrated. Examples are given for addition and subtraction of signed binary numbers and decimal numbers in binary-coded decimal (BCD) format.

Chapter 2 introduces the basic postulates of Boolean algebra and shows the correlation between Boolean expressions and their corresponding logic diagrams. All possible logic operations for two variables are investigated, and the most useful logic gates used in the design of digital systems are identified. This chapter also introduces basic CMOS logic gates.

Chapter 3 covers the map method for simplifying Boolean expressions. The map method is also used to simplify digital circuits constructed with AND-OR, NAND, or NOR gates. All other possible two-level gate circuits are considered, and their method of implementation is explained. Verilog HDL is introduced together with simple examples of gate-level models.

Chapter 4 outlines the formal procedures for the analysis and design of combinational circuits. Some basic components used in the design of digital systems, such as adders and code converters, are introduced as design examples. Frequently used digital logic functions such as parallel adders and subtractors, decoders, encoders, and multiplexers are explained, and their use in the design of combinational circuits is illustrated. HDL examples are given in gate-level, dataflow, and behavioral models to show the alternative ways available for describing combinational circuits in Verilog HDL. The procedure for writing a simple test bench to provide stimulus to an HDL design is presented.

Chapter 5 outlines the formal procedures for analyzing and designing clocked (synchronous) sequential circuits. The gate structure of several types of flip-flops is presented together with a discussion on the difference between level and edge triggering. Specific examples are used to show the derivation of the state table and state diagram when analyzing a sequential circuit. A number of design examples are presented with emphasis on sequential circuits that use D-type flip-flops. Behavioral modeling in Verilog HDL for sequential circuits is explained. HDL Examples are given to illustrate Mealy and Moore models of sequential circuits.

Chapter 6 deals with various sequential circuit components such as registers, shift registers, and counters. These digital components are the basic building blocks from which more complex digital systems are constructed. HDL descriptions of shift registers and counter are presented.

Chapter 7 deals with random access memory (RAM) and programmable logic devices. Memory decoding and error correction schemes are discussed. Combinational and sequential programmable devices such as ROMs, PLAs, PALs, CPLDs, and FPGAs are presented.

Chapter 8 deals with the register transfer level (RTL) representation of digital systems. The algorithmic state machine (ASM) chart is introduced. A number of examples demonstrate the use of the ASM chart, ASMD chart, RTL representation, and HDL description in the design of digital systems. The design of a finite state machine to control a datapath is presented in detail, including the realistic situation in which status signals from the datapath are used by the state machine that controls it. This chapter is the most important chapter in the book as it provides the student with a systematic approach to more advanced design projects.

Chapter 9 outlines experiments that can be performed in the laboratory with hardware that is readily available commercially. The operation of the ICs used in the experiments is explained by referring to diagrams of similar components introduced in previous chapters. Each experiment is presented informally and the student is expected to design the circuit and formulate a procedure for checking its operation in the laboratory. The lab experiments can be used in a stand-alone manner too and can be accomplished by a traditional approach, with a breadboard and TTL circuits, or with an HDL/synthesis approach using FPGAs. Today, software for synthesizing an HDL model and implementing a circuit with an FPGA is available at no cost from vendors of FPGAs, allowing students to conduct a significant amount of work in their personal environment before using prototyping boards and other resources in a lab.

Circuit boards for rapid prototyping circuits with FPGAs are available at a nominal cost, and typically include push buttons, switches, seven-segment displays, LCDs, keypads, and other I/O devices. With these resources, students can work prescribed lab exercises or their own projects and get results immediately.

Chapter 10 presents the standard graphic symbols for logic functions recommended by an ANSI/IEEE standard. These graphic symbols have been developed for small-scale integration (SSI) and medium-scale integration (MSI) components so that the user can recognize each function from the unique graphic symbol assigned. The chapter shows the standard graphic symbols of the ICs used in the laboratory experiments.

ACKNOWLEDGMENTS

We are grateful to the reviewers of *Digital Design*, 5e. Their expertise, careful reviews, and suggestions helped shape this edition.

Dmitri Donetski, Stony Brook University
Ali Amini, California State University, Northridge
Mihaela Radu, Rose Hulman Institute of Technology
Stephen J Kuyath, University of North Carolina, Charlotte
Peter Pachowicz, George Mason University
David Jeff Jackson, University of Alabama
A. John Boye, University of Nebraska, Lincoln
William H. Robinson, Vanderbilt University
Dinesh Bhatia, University of Texas, Dallas

We also wish to express our gratitude to the editorial and publication team at Prentice Hall/Pearson Education for supporting this edition of our text. We are grateful, too, for the ongoing support and encouragement of our wives, Sandra and Jerilynn.

M. Morris Mano
Emeritus Professor of Computer Engineering
California State University, Los Angeles

Michael D. Ciletti
Emeritus Professor of Electrical and Computer Engineering
University of Colorado at Colorado Springs