GLOBAL EDITION

Digital Systems

Principles and Applications

TWELFTH EDITION

Neal S. Widmer • Gregory L. Moss • Ronald J. Tocci

Pearson

TWELFTH EDITION

GLOBAL EDITION

Digital Systems

Principles and Applications

Neal S. Widmer
Purdue University

Gregory L. Moss
Purdue University

Ronald J. Tocci
Monroe Community College

Pearson

Harlow, England • London • New York • Boston • San Francisco • Toronto • Sydney • Dubai • Singapore • Hong Kong
Tokyo • Seoul • Taipei • New Delhi • Cape Town • Sao Paulo • Mexico City • Madrid • Amsterdam • Munich • Paris • Milan

Editor-in-Chief: Andrew Gilfillan
Product Manager: Anthony Webster
Program Manager: Holly Shufeldt
Project Manager: Rex Davidson
Editorial Assistant: Nancy Kesterson
Team Lead Project Manager: Bryan Pirrmann
Team Lead Program Manager: Laura Weaver
Project Manager, Global Edition: Sudipto Roy
Senior Acquisitions Editor, Global Edition: Sandhya Ghoshal
Senior Project Editor, Global Edition: Daniel Luiz
Project Editor, Global Edition: Rahul Arora
Manager, Media Production, Global Edition: M. Vikram Kumar
Manufacturing Controller, Production, Global Edition: Angela Hawksbee
Director of Marketing: David Gesell
Senior Product Marketing Manager: Darcy Betts
Field Marketing Manager: Thomas Hayward
Procurement Specialist: Deidra M. Skahill
Creative Director: Andrea Nix
Art Director: Diane Y. Ernsberger
Cover Designer: Lumina Datamatics, Inc.
Full-Service Project Management: Philip Alexander/Integra Software Services, Pvt, Ltd.

Pearson Education Limited
Edinburgh Gate
Harlow
Essex CM20 2JE
England

and Associated Companies throughout the world

Visit us on the World Wide Web at:
www.pearsonglobaleditions.com

Authorized adaptation from the United States edition, entitled Digital Systems, 12th edition, ISBN 978-0-134-22013-0, by Ronald Tocci, Neal Widmer, and Greg Moss, published by Pearson Education © 2017.

ISBN 10: 129-2-16200-7
ISBN 13: 978-1-292-16200-3

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

10 9 8 7 6 5 4 3 2 1
14 13 12 11 10

Printed and bound in Vivar, Malaysia.
Typeset in Times Europa LT Std Roman by Integra Software Services, Pvt, Ltd.

PREFACE

This book is a comprehensive study of the principles and techniques of modern digital systems. It teaches the fundamental principles of digital systems and covers thoroughly both traditional and modern methods of applying digital design and development techniques, including how to manage a systems-level project. The book is intended for use in two- and four-year programs in technology, engineering, and computer science. It can also be used for High School STEM education courses in these topical areas. Although a background in basic electronics is helpful, most of the material requires no electronics training. Portions of the text that use electronics concepts can be skipped without adversely affecting the comprehension of the logic principles.

What's New in This Edition?

The following list summarizes the improvements in the twelfth edition of *Digital Systems*. Details can be found in the section titled "Specific Changes" on page 6.

- Every *section* of every chapter now has a short list of expected outcomes for that section.
- Chapter 1 has been revised extensively in response to feedback from users.
- New material on troubleshooting prototype circuits using systematic fault isolation techniques applied to digital logic circuits has been added to Section 4-13.
- Quadrature Shaft Encoders used to obtain absolute shaft position serve as a real example of flip-flop applications, and timing limitations.
- More material has been added to better explain the behavior of VHDL data objects and how they are updated in sequential processes.
- Throughout the text, obsolete technology has been deleted or abbreviated to provide only content appropriate to modern systems. More modern examples are used as needed.
- Some new problems have been added and outdated problems have been removed.

General Features

In industry today, getting a product to market very quickly is important. The use of modern design tools, CPLDs, and FPGAs allows engineers to progress from concept to functional silicon very quickly. Microcontrollers have taken over many applications that once were implemented by digital circuits, and DSP has been used to replace many analog circuits. It is amazing that microcontrollers, DSP, and all the necessary glue logic can now be consolidated onto a single FPGA using a hardware description language with advanced development tools. Today's students must be exposed to these modern tools, even in an introductory course. It is every educator's responsibility to find the best way to prepare graduates for the work they will encounter in their professional lives.

The standard SSI and MSI parts that have served as "bricks and mortar" in the building of digital systems for over 40 years are now obsolete and becoming less available. Many of the techniques that have been taught over that time have focused on optimizing circuits that are built from these outmoded devices. The topics that are uniquely suited to applying the old technology *but do not contribute to an understanding of the new technology* are being de-emphasized. From an educational standpoint, however, these small ICs do offer a way to study simple digital circuits, and the wiring of circuits using breadboards is a valuable pedagogic exercise. They help to solidify concepts such as binary inputs and outputs, physical device operation, and practical limitations, using a very simple platform. Consequently, we have chosen to continue to introduce the conceptual descriptions of digital circuits and to offer examples using conventional standard logic parts. For instructors who continue to teach the fundamentals using SSI and MSI circuits, this edition retains those qualities that have made the text so widely accepted in the past. Many hardware design tools even provide an easy-to-use design entry technique that will employ the functionality of conventional standard parts with the flexibility of programmable logic devices. A digital design can be described using a schematic drawing with pre-created building blocks that are equivalent to conventional standard parts, which can be compiled and then programmed directly into a target PLD with the added capability of easily simulating the design within the same development tool.

We believe that graduates will actually apply the concepts presented in this book using higher-level description methods and more complex programmable devices. The major shift in the field is a greater need to understand the description methods, rather than focusing on the architecture of an actual device. Software tools have evolved to the point where there is little need for concern about the inner workings of the hardware but much more need to focus on what goes in, what comes out, and how the designer can describe what the device is supposed to do. We also believe that graduates will be involved with projects using state-of-the-art design tools and hardware solutions.

This book offers a strategic advantage for teaching the vital topic of hardware description languages to beginners in the digital field. VHDL is undisputedly an industry standard language at this time, but it is also very complex and has a steep learning curve. Beginning students are often discouraged by the rigorous requirements of various data types, and they struggle with understanding edge-triggered events in VHDL. Fortunately, Altera offers AHDL, a less demanding language that uses the same basic concepts as VHDL but is much easier for beginners to master. So, instructors can opt to use AHDL to teach introductory students or VHDL for more advanced classes. This edition offers more than 40 AHDL examples, more than 40 VHDL examples, and many examples of simulation testing. All of these design files are available on the website (http://www.pearsonglobaleditions.com/tocci).

Altera's software development system is Quartus II. The material in this text does not attempt to teach a particular hardware platform or the details of using a software development system. We have chosen to show what this tool can do, rather than train the reader how to use it.

Many laboratory hardware options are available to users of this book. Complete development boards are available that offer the normal types of inputs and outputs like logic switches, pushbuttons, clock signals, LEDs, and 7-segment displays. Many boards also offer standard connectors for readily available computer hardware, such as a standard keyboard, computer mouse, VGA video monitor, COM ports, audio in/out jacks, plus two 40-pin general-purpose I/O ribbon connectors that allow connection to any digital peripheral hardware.

Our approach to HDL and PLDs gives instructors several options:

1. The HDL material can be skipped entirely without affecting the continuity of the text.
2. HDL can be taught as a separate topic by skipping the material initially and then going back to the last sections of Chapters 3, 4, 5, 6, 7, and 9 and then covering Chapter 10.
3. HDL and the use of PLDs can be covered as the course unfolds—chapter by chapter—and woven into the fabric of the lecture/lab experience.

Among all specific hardware description languages, VHDL is clearly the industry standard and is most likely to be used by graduates in their careers. We have always felt that it is a bold proposition, however, to try to teach VHDL in an introductory course. The nature of the syntax, the subtle distinctions in object types, and the higher levels of abstraction can pose obstacles for a beginner. For this reason, we have included Altera's AHDL as the recommended introductory language for freshman and sophomore courses. We have also included VHDL as the recommended language for more advanced classes or introductory courses offered to more mature students. We do not recommend trying to cover both languages in the same course. Sections of the text that cover the specifics of a language are clearly designated with a color bar in the margin. The HDL code figures are set in a color to match the color-coded text explanation. The reader can focus only on the language of his or her choice and skip the other. Obviously, we have attempted to appeal to the diverse interests of our market, but we believe we have created a book that can be used in multiple courses and will serve as an excellent reference after graduation.

Chapter Organization

Many instructors opt to not use the chapters of a textbook in the sequence in which they are presented. This book was written so that, for the most part, each chapter builds on previous material, but it is possible to alter the chapter sequence somewhat. The first part of Chapter 6 (arithmetic operations) can be covered right after Chapter 2 (number systems), although this will lead to a long interval before the arithmetic circuits of Chapter 6 are encountered. Much of the material in Chapter 8 (IC characteristics) can be covered earlier (e.g., after Chapter 4 or 5) without creating any serious problems.

This book can be used either in a one-term course or in a two-term sequence. In a one-term course, limits on available class hours might require omitting some topics. Obviously, the choice of deletions will depend on factors such as program or course objectives and student background. Sections

FIGURE P1 Letters denote categories of problems, and asterisks indicate that corresponding solutions are provided at the end of the text.

PROBLEMS

SECTION 9-1

B 9-1. Refer to Figure 9-3. Determine the levels at each decoder output for the following sets of input conditions.
(a)*All inputs LOW
(b)*All inputs LOW except E_3 = HIGH
(c) All inputs HIGH except $\overline{E}_1 = \overline{E}_2$ = LOW
(d) All inputs HIGH

B 9-2.* What is the number of inputs and outputs of a decoder that accepts 128 different input combinations?

* Answers to problems marked with an asterisk can be found in the back of the text.

in each chapter that deal with troubleshooting, PLDs, HDLs, or microcomputer applications can be deferred to an advanced course.

PROBLEM SETS This edition includes six categories of problems: basic (B), challenging (C), troubleshooting (T), new (N), design (D), and HDL (H). Undesignated problems are considered to be of intermediate difficulty, between basic and challenging. Problems for which solutions are printed in the back of the text or on the website (http://www.pearsonglobaleditions.com/tocci) are marked with an asterisk (see Figure P1).

PROJECT MANAGEMENT AND SYSTEM-LEVEL DESIGN Several real-world examples are included in Chapter 10 to describe the techniques used to manage projects. These applications are generally familiar to most students studying electronics, and the primary example of a digital clock is familiar to everyone. Many texts talk about top-down design, but this text demonstrates the key features of this approach and how to use the modern tools to accomplish it.

SIMULATION FILES This edition also includes simulation files that can be loaded into Multisim®. The circuit schematics of many of the figures throughout the text have been captured as input files for this popular simulation tool. Each file has some way of demonstrating the operation of the circuit or reinforcing a concept. In many cases, instruments are attached to the circuit and input sequences are applied to demonstrate the concept presented in one of the figures of the text. These circuits can then be modified as desired to expand on topics or create assignments and tutorials for students. All figures in the text that have a corresponding simulation file on the website are identified by the icon shown in Figure P2.

Specific Changes

The major changes in the topical coverage are listed here.

- **Chapter 1.** Chapter 1 has been revised extensively in response to feedback from users. The significance of how Digital Systems will impact innovations of the future is emphasized.

  New material focuses on interpretation of terminology and introduction to concepts used throughout the text. Basic concepts of binary

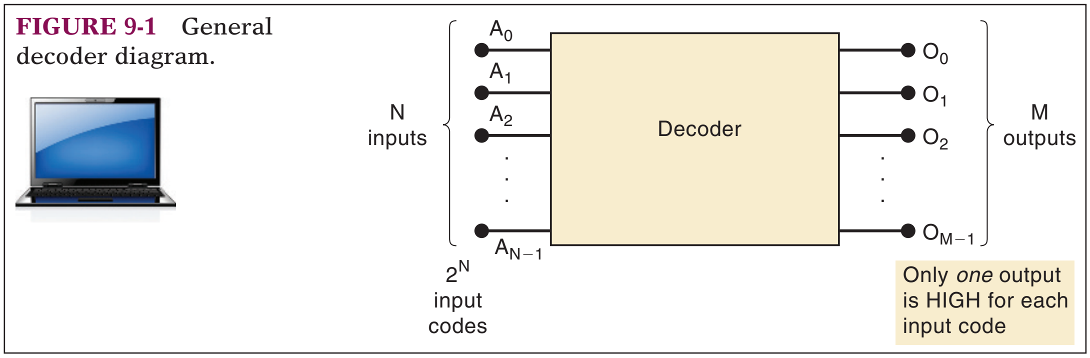


FIGURE P2 The icon denotes a corresponding simulation file on the Web.

signals are introduced and explained through examples. New material on periodic cycles and measurements on digital waveforms is presented, setting the stage for understanding these issues in later chapters. The basics of digital signals and sampling are explained at a very introductory level.

This chapter in the 11th edition had material that has now become very outdated since its publication. Some of the historic analogies used in that edition were ineffective. The revisions have replaced or eliminated these.

- **Chapter 2.** The Gray Code is used to introduce the concept of a quadrature encoder: a device that produces a 2-bit Gray Code sequence capable of discerning the direction and angular rotation of a shaft.
- **Chapter 3.** New problems at the end of this chapter focus on logic circuits common to automobiles.
- **Chapter 4.** The material introducing PLD programming and development software has been updated and improved. The section on troubleshooting has been expanded to teach structured problem solving as it applies to hardware debugging of traditional prototyped digital circuits. The VHDL material has been enhanced to explain some subtle but very important aspects of data objects in this language. The role of the "PROCESS" is also more thoroughly covered improving the foundation that Chapter 5 builds on.
- **Chapter 5.** High-speed digital systems are easily affected by timing limitations of the circuitry. New material in this chapter explains the adverse effects caused when setup and hold time requirements are violated by explaining meta-stability. A teaching example that can be reproduced in the laboratory environment has been added. The focus is on the many applications of D flip-flops but it is presented in the context of a quadrature shaft encoder that must reliably and repeatedly keep track of absolute shaft position as it is rotated back and forth over many cycles. Design techniques from Chapter 4 are employed to design a circuit that should meet the system's needs. The initial circuit's marginal performance demonstrates what happens when real-timing constraints are not taken into account. A way to correct this problem is presented using even more applications of D flip-flops.
- **Chapter 6.** An Example from the 11th edition used some features of Quartus software that have since become obsolete. The example has been modified to align with more recent updates of Quartus.
- **Chapter 7.** Very few and minor changes were made to Chapter 7.
- **Chapter 8.** The section on the obsolete Emitter Coupled Logic (ECL) was deleted along with other minor updates.

- **Chapter 9.** The concept of Time Division Multiplexing is added to provide an example of how many digital signals are able to share a common data pathway. A simple system is presented that can easily be reproduced in a laboratory exercise.
- **Chapter 10.** No changes were made in Chapter 10.
- **Chapter 11.** No changes were made in Chapter 11.
- **Chapter 12.** The coverage of floating gate MOSFETS, the technology behind flash memory, is enhanced.
- **Chapter 13.** This chapter has been generalized with references to older series of CPLDs and FPGAs abbreviated.

Retained Features

This edition retains all of the features that made the previous editions so widely accepted. It utilizes a block diagram approach to teach the basic logic operations without confusing the reader with the details of internal operation. All but the most basic electrical characteristics of the logic ICs are withheld until the reader has a firm understanding of logic principles. In Chapter 8, the reader is introduced to the internal IC circuitry. At that point, the reader can interpret a logic block's input and output characteristics and "fit" it properly into a complete system.

The treatment of each new topic or device typically follows these steps: the principle of operation is introduced; thoroughly explained examples and applications are presented, often using actual ICs; short review questions are posed at the end of the section; and finally, in-depth problems are available at the end of the chapter. These problems, ranging from simple to complex, provide instructors with a wide choice of student assignments. These problems are often intended to reinforce the material without simply repeating the principles. They require students to demonstrate comprehension of the principles by applying them to different situations. This approach also helps students to develop confidence and expand their knowledge of the material.

The material on PLDs and HDLs is distributed throughout the text, with examples that emphasize key features in each application. These topics appear at the end of each chapter, making it easy to relate each topic to the general discussion earlier in the chapter or to address the general discussion separately from the PLD/HDL coverage.

The extensive troubleshooting coverage is spread over Chapters 4 through 12 and includes presentation of troubleshooting principles and techniques, case studies, 17 troubleshooting examples, and 46 *real* troubleshooting problems. When supplemented with hands-on lab exercises, this material can help foster the development of good troubleshooting skills.

This edition offers more than 220 worked-out examples, more than 660 review questions, and more than 640 chapter problems/exercises. Some of these problems are applications that show how the logic devices presented in the chapter are used in a typical microcomputer system. Answers to a majority of the problems immediately follow the Glossary. The Glossary provides concise definitions of all terms in the text that have been highlighted in boldface type.

An IC index is provided at the back of the book to help readers locate easily material on any IC cited or used in the text. The back endsheets provide tables of the most often used Boolean algebra theorems, logic gate summaries, and flip-flop truth tables for quick reference when doing problems or working in the lab.

Supplements

An extensive complement of teaching and learning tools has been developed to accompany this textbook. Each component provides a unique function, and each can be used independently or in conjunction with the others.

WEB RESOURCES

- **Quartus II Web Version software from Altera.** This development system software is available from Altera.
- **Design files from the textbook figures.** More than 40 design files in each language are presented in figures throughout the text. Students can load these into the Altera software and test them.
- **Circuits from the text rendered in Multisim®.** Students can open and work interactively with approximately 100 circuits to increase their understanding of concepts and prepare for laboratory activities. The Multisim circuit files are provided for use by anyone who has Multisim software.

INSTRUCTOR RESOURCES

- ***Online Instructor's Resource Manual.*** This manual contains worked-out solutions for all end-of-chapter problems in this textbook.
- ***Online PowerPoint® presentations.*** Figures from the text, in addition to Lecture Notes for each chapter, are available.
- ***Online TestGen.*** A computerized test bank is available.

To access supplementary materials online, instructors need to request an instructor access code. Go to **www.pearsonglobaleditions.com/tocci**, where you can register for an instructor access code. Within 48 hours after registering, you will receive a confirming e-mail, including an instructor access code. Once you have received your code, go to the site and log on for full instructions on downloading the materials you wish to use.

Acknowledgments

We are grateful to all those who evaluated the eleventh edition and provided answers to an extensive questionnaire:

Their comments, critiques, and suggestions were given serious consideration and were invaluable in determining the final form of the twelfth edition.

We also are greatly indebted to Professor Frank Ambrosio, Monroe Community College, for his usual high-quality work on the *Instructor's*

Resource Manual; and Professor Daniel Leon-Salas, Purdue University, for his technical review of topics and many suggestions for improvements.

A writing project of this magnitude requires conscientious and professional editorial support, and Pearson came through again in typical fashion. We thank the staff at Pearson for their help to make this publication a success.

And finally, we want to let our wives, children, and grandchildren know how much we appreciate their support and their understanding. We hope that we can eventually make up for all the hours we spent away from them while we worked on this revision.

Neal S. Widmer
Ronald J. Tocci
Gregory L. Moss

Acknowledgments for the Global Edition

Pearson would like to thank the following people for their work on the content of the Global Edition:

Contributors:

Moumita Mitra Manna, University of Calcutta

Ankita Pramanik, Indian Institute of Engineering Science and Technology, Shibpur

Reviewers:

Chih-Wei Liu, National Chiao Tung University

Hung-Ming Chen, National Chiao Tung University

Ankita Pramanik, Indian Institute of Engineering Science and Technology, Shibpur

CONTENTS

CHAPTER 1 Introductory Concepts 22

CHAPTER 2 Number Systems and Codes 56

CHAPTER 3 Describing Logic Circuits 88

CHAPTER 4 Combinational Logic Circuits 156

CHAPTER 10 Digital System Projects Using HDL 764

CHAPTER 11 Interfacing with the Analog World 814

To you, Cap, for loving me for so long; and for the million and one ways you brighten the lives of everyone you touch.

—RJT

To my wife and best friend, Kris, who has sacrificed the most to complete this work. To our children John and Brooke, Brad and Amber, Blake and Tashi, Matt and Tamara, Katie and Matthew, and to our grandchildren Jersey, Judah, and the two we have yet to meet, who are in production along with this book.

—NSW

To my expanding family, Marita, David, Ryan, Christy, Jeannie, Taylor, Micah, Brayden, and Lorelei.

—GLM

CHAPTER 1

INTRODUCTORY CONCEPTS

OUTLINE

CHAPTER OUTCOMES

Upon completion of this chapter, you will be able to:

- Distinguish between analog and digital representations.
- Describe how information can be represented using just two states (1s and 0s).
- Cite the advantages and drawbacks of digital techniques compared with analog.
- Describe the purpose of analog-to-digital converters (ADCs) and digital-to-analog converters (DACs).
- Recognize the basic characteristics of the binary number system.
- Convert a binary number to its decimal equivalent.
- Count in the binary number system.
- Identify typical digital signals.
- Identify a timing diagram.
- State the differences between parallel and serial transmission.
- Describe the property of memory.
- Describe the major parts of a digital computer and understand their functions.
- Distinguish among microcomputers, microprocessors, and microcontrollers.

INTRODUCTION

In today's world, the term *digital* has become part of our everyday vocabulary because of the dramatic way that digital circuits and digital techniques have become so widely used in almost all areas of life: computers, automation, robots, medical science and technology, transportation, telecommunications, entertainment, space exploration, and on and on. You are about to begin an exciting educational journey in which you will discover the fundamental principles, concepts, and operations that are common to all digital systems, from the simplest on/off switch to the most complex computer.

This chapter will introduce many of the underlying concepts that you will encounter as you learn more about your digital world. As new terms and concepts are presented, you will be directed to the chapters later in the text that expand and clarify the points. We want you to realize just how deeply digital systems impact your life. Then we want you to wonder how they work and how you might use digital systems to make the future better.

Let's go through a typical example of starting a day. The alarm clock wakes me up and I look at the time of day displayed on big bright

seven-segment LEDs (see Chapter 9). The digital alarm has compared the time of day with my alarm setting and when they were equal it activated the alarm (see Chapter 10). The alarm is "latched" on until I reset it with "off" or "snooze"(see Chapter 5 for latches). I go to the bathroom and decide to weigh myself before showering. The bathroom scales respond to the tap of my toe by awaking from its sleep mode, clearing the digital display and waiting for me to step on. It measures my weight and displays it in pounds. After a few seconds, it goes back to sleep. I grab my cordless shaver from the charger. A digital circuit inside the shaver has been controlling the charging cycle. I pick up my electric toothbrush. It can operate in three modes or "states" depending on how many times I push the button (see state machines in Chapter 7). It also keeps track of how long I brush and signals every 30 seconds in a 2 minute brush cycle. This is all controlled by a digital system inside the toothbrush hand-piece. I flip on the closet light. It has an energy saver feature that turns it off in case I forget, thanks to a small digital circuit in the light bulb (see interfacing in Chapter 8). I walk into my bedroom and turn the lights on low using the dimmer switch. The dimmer switch is an old analog circuit, but the new LED light bulbs can still be dimmed by it! This is because of a digital circuit inside the LED light bulb that controls the LEDs (see pulse width modulation in Chapter 11). I disconnect my cell phone from its charger. What a digital miracle I am holding in my hand!

I have not left the bedroom and already my life has been touched by seven digital systems. We could continue but you get the idea. Digital systems are everywhere around you and new applications are constantly being developed. All of the digital systems in the world are built from a surprisingly small number of basic circuits or building blocks. There are many instances of each block in most systems but only a few different blocks. This book will introduce you to those basic digital circuits and help you to understand the purpose, role, capabilities, and limitations of each one. Then you can use your innovation skills and the knowledge from this book to meet the next new demand.

1-1 INTRODUCTION TO DIGITAL 1s AND 0s

OUTCOMES

Upon completion of this section, you will be able to:

- Correlate new terms with their definition.
- Identify two states and assign a digit to each.
- Correlate each state with its representation in a given circuit.
- Recognize which state will activate a device in a given system.
- Identify the state of a digital signal under various physical conditions.
- Assign proper names to signals in a digital system.

Digital systems deal with things that are in one of two distinct states. The easiest example is anything that is either on or off. If you look at many devices today, you will find that the on/off switch is a single push button with the symbol shown in **Figure 1-1**. This icon represents a 1 and a 0, the numerical digits used to describe the two states in a digital system. We use numeric digits 0 and 1 to represent the two states off and on, respectively. Since there are only two digits, we call them **binary digits,** or **bits.** It is often said that digital systems are just a bunch of 1s and 0s and that is pretty

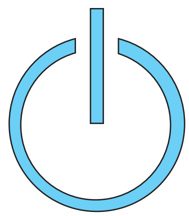


FIGURE 1-1 The ubiquitous on/off symbol.

accurate. When we organize groups of numeric digits, we can create number systems and number systems are very powerful ways to represent things. As can be seen from all the digital systems around us, a lot can be done with just two possible states when circuits that can represent these two states are strategically organized.

Let's try to identify some things that must be categorized in one of two states in a system that is familiar to everyone: the automobile. The doors are either locked or unlocked. There is no such thing as being partially locked. We could also say a door is either open or closed. Now we know that a door can be partially opened, but in an automotive system the important thing to know is when the door is completely closed and safely latched. One state is considered to be closed and latched, while the other state is anything from slightly ajar to wide open. The parking brake is either set (engaged to any degree) or it is not set (completely disengaged). The engine is either running (at any speed) or it is not running. A button on the trunk lid is either pressed or not pressed. On some cars, opening the trunk when the engine is running requires the parking break to be set, the doors unlocked, and the trunk button to be pressed. When the engine is not running the trunk can be opened whenever the trunk button is pressed and the doors are unlocked. Digital circuits observe the state of each component and make a "logical" decision to either open or not open the trunk. For this reason, these conditions are often referred to as **logic states**.

After the two states of a system component are defined, one of the digital values (1 or 0) is assigned to each state. For example, on a Ford perhaps a door that is open may be assigned a 1 (closed = 0), but on a Lexus a door that is open may be assigned a state of 0 (closed = 1). In Chapter 3, we will discuss naming conventions for digital signals that help avoid confusion regarding the meaning of 1s and 0s in any system.

How are the states of 1 and 0 represented electrically in a digital system? The answer depends on the technology of the electrical system but the simplest answer is that a 0 is generally represented by a low voltage (close to 0 V) and a 1 is generally represented by a higher voltage. Consider, as an example, common electrical circuits in a home and in an automobile. In electrical systems, a voltage must be applied to a complete circuit to cause current to flow through the active device and "turn it on." **Figure 1-2(a)** demonstrates a light bulb in your home which requires 110 V AC (alternating current) to turn the light on. When no voltage is applied (0 volts AC), the light is off. Any light bulb in your car requires 12 V DC (direct current) to turn the light on and 0 V DC to turn it off, as demonstrated in Figure 1-2(b). The two systems are very similar but the technology of the systems differs. Consequently, the representations of a HIGH state (i.e., higher voltage) must match the system. In these simple wiring examples, the HIGH voltage is either connected to or disconnected from the lamp. A more accurate model of a digital logic circuit reflects that the output is always connected to either the source of the high voltage (HIGH state) or the source of the low voltage (LOW state). Figures 1-2(c) and (d) illustrate how this would look

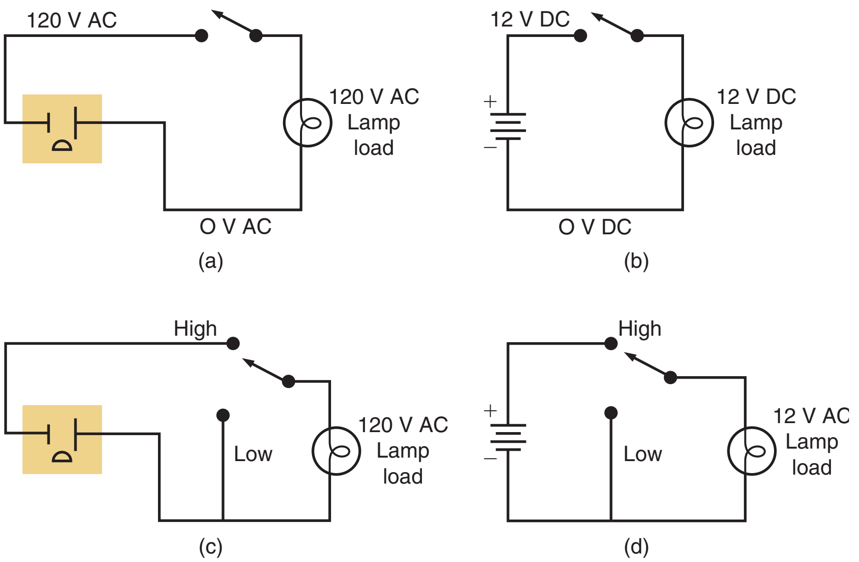


FIGURE 1-2 (a) Typical 120 V AC house wiring; (b) typical 12 V DC automotive wiring; (c) 120 V AC model of a logic circuit; (d) 12 V AC model of a logic circuit.

for a simple light circuit. Chapter 8 will thoroughly explain why digital logic circuits operate like Figures 1-2(c) and (d) rather than like simple electrical wiring in your home or car, as depicted in Figures 1-2(a) and (b). The main point is that a 0 is typically represented by the LOW voltage or value near 0 V. The state designated as 1 is typically represented by a HIGH voltage and the value of that voltage depends on the technology of the system. These values of HIGH and LOW are often referred to as **logic levels.**

Some digital devices are activated by applying a HIGH, while others are activated by applying a LOW. **Figure 1-3** demonstrates these two scenarios for a simple light circuit. Notice that in Figure 1-3(a) the switch supplies the HIGH by connecting the voltage source which supplies current from the battery to the light and activates the light. In Figure 1-3(b), the switch supplies the LOW by connecting the return path from the light to the battery in order to activate the light. In Chapter 3, we will further investigate this concept of a device being active-HIGH or active-LOW.

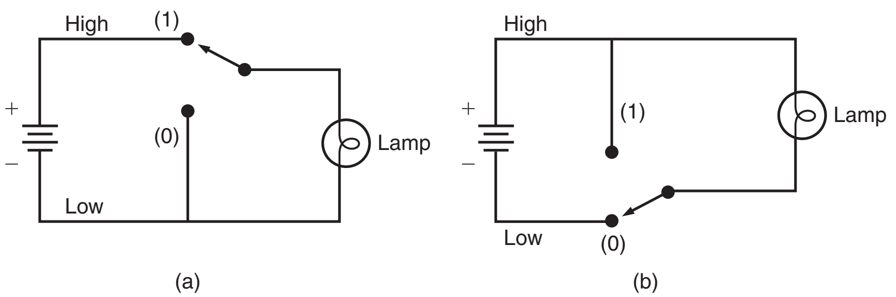


FIGURE 1-3 (a) Applying HIGH turns the lamp ON; (b) applying LOW turns the lamp ON.

Sensors that serve as inputs to digital systems also can be wired in many different ways. For example, consider a circuit that can determine if the key to a car has been inserted into the ignition switch. As we are often reminded, this piece of information is used to sound an alarm if the car door is opened when the key is still in the ignition. **Figure 1-4** demonstrates two possible ways to wire this switch and the affect each method has on the meaning of the digital output level. In Figure 1-4(a), the contacts are open, producing a LOW when no key is present. When the key is inserted, as in Figure 1-4(b), it pushes contact points to the +12 V position, producing a

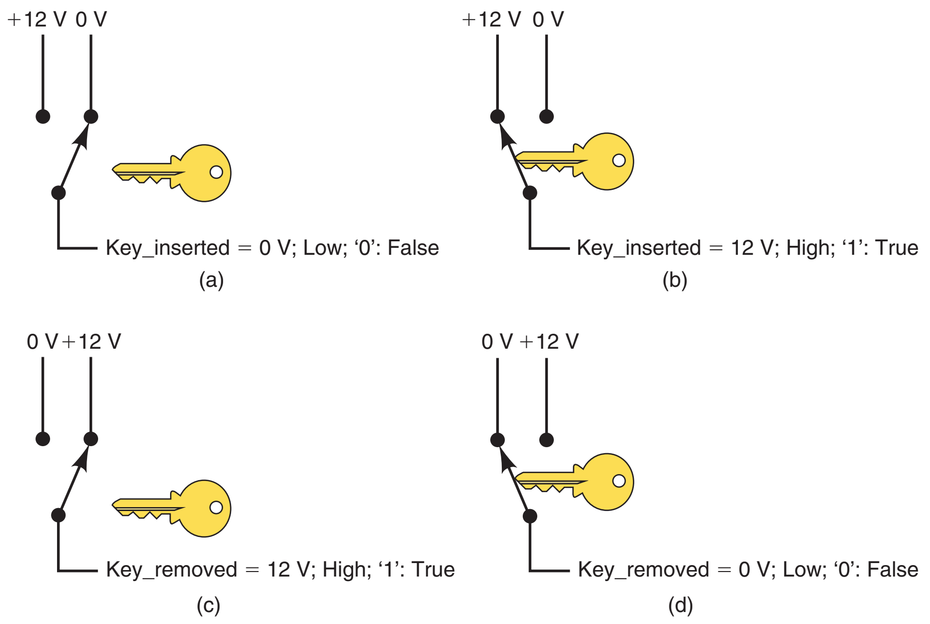


FIGURE 1-4 Physical conditions, logic levels, and signal labels: (a) false that key is inserted, (b) true that key is inserted, (c) true that key is removed, (d) false that key is removed.

HIGH at the output. A good label for the output signal from this circuit would be *key_inserted* because the logic level of HIGH represents the state of 1 or true. *Key_inserted* is true when the output is HIGH. Contrast this circuit with Figure 1-4(c) in which the switch contacts are wired in the opposite way. In this case, inserting the key produces a LOW (Figure 1-4(d)) and removing the key produces a HIGH (Figure 1-4(c)). A good label for this signal is *key_removed* because it is true that the key is removed when the output is HIGH. The name of the signal describes a physical condition which should be true when the level is HIGH or 1. Chapters 3 and 4 will expand on these concepts using HIGHs and LOWs to activate/deactivate other circuits. This is fundamental to understanding all digital systems.

Now that we know that 1s are represented by a HIGH voltage and 0s by LOW voltage, all that remains is defining how high the voltage must be to be considered a 1 and how low a voltage must be to be considered a 0. The answer to this question also depends on the technology used to implement the digital system. Electronic digital systems have gone through many changes as technology has advanced. But the principles of representing 1s and 0s remain the same. In all systems, a defined range of higher voltages is acceptable as a HIGH (1). Another defined range of lower voltages is acceptable as a LOW (0). In between is a range of voltages that is considered neither HIGH nor LOW. Voltages in this range are considered invalid. **Figure 1-5**

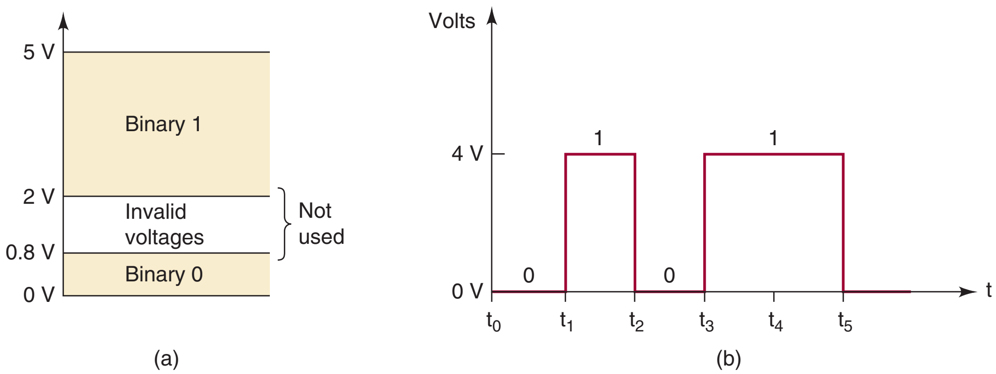


FIGURE 1-5 Logic levels and timing (a) typical voltage ranges for a given technology of digital circuits. (b) a graph of signal levels changing over time.

demonstrates this concept for 5-volt logic systems that were based on bipolar transistor technology. Figure 1-5a indicates that in order for circuits using this technology to recognize the input as a '1' it must be a voltage greater than two but less than five. The input voltage must be less than 0.8 V to recognize it as a '0'. In the evolution of digital systems, various technologies such as electromechanical switches (relays), vacuum tubes, bipolar transistors, and MOSFET transistors have been used to implement digital logic circuits, each with their own characteristic definition of how to represent a 1 and a 0.

It is quite common and often necessary to depict the activity of a logic level over time. We called this a **timing diagram**. Figure 1-5(b) represents a typical digital waveform for the voltage ranges defined in part (a). The time axis is labeled at specific points in time, t_1, t_2, … t_5. Notice that the HIGH voltage level between t_1 and t_2 is at 4 V. In digital systems, the exact value of a voltage is not important. A HIGH voltage of 3.7 V or 4.3 V would represent the exact same information. Likewise, a LOW voltage of 0.3 V represents the same information as 0 V. This points out a significant difference between analog and digital systems. In an analog system, the exact voltage is important. For example, if the analog voltage coming from a sensor is proportional to temperature, then 3.7 V would represent a different value of temperature than 4.3 V. In other words, the voltage carries significant information in the analog system. Circuits that can preserve exact voltages are much more complicated than digital circuits that simply need to recognize a voltage in one of two ranges. Digital circuits are designed to produce output voltages that fall within the prescribed 0 and 1 voltage ranges such as those defined in Figure 1-5. Likewise, digital circuits are designed to respond predictably to input voltages that are within the defined 0 and 1 ranges. What this means is that a digital circuit will respond in the same way to all input voltages that fall within the allowed 0 range; similarly, it will not distinguish between input voltages that lie within the allowed 1 range. To illustrate, Figure 1-6 represents a typical digital circuit with input v_i and output v_o. The output is shown for two different input signal waveforms. Note that v_o is the same for both cases because the two input waveforms, while differing in their exact voltage levels, are at the same binary levels.

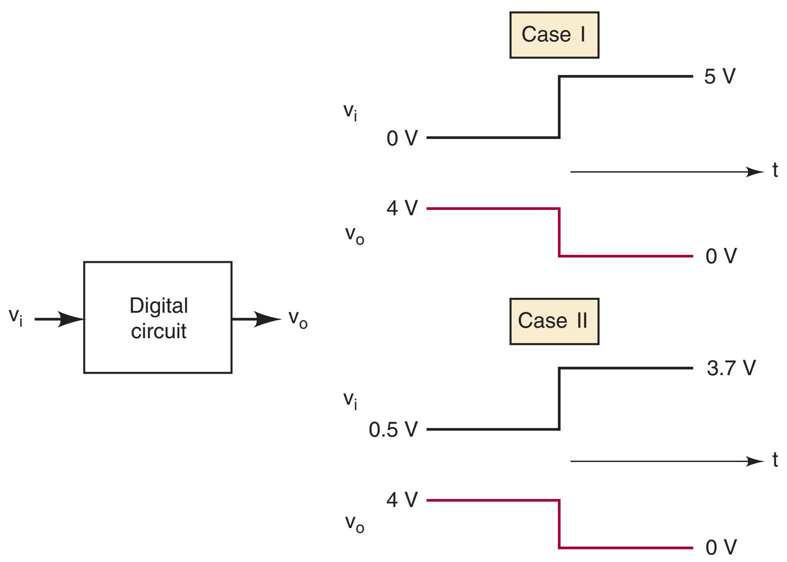


FIGURE 1-6 A digital circuit responds to an input's binary level (0 or 1) and not to its actual voltage.

OUTCOME ASSESSMENT QUESTIONS*

1. What are the two numeric digits used to represent states in a digital system?
2. What are the two terms used to represent the two logic levels?
3. What is the abbreviation for binary digit?
4. Which binary digit value is typically represented by low (near-zero) voltage?
5. What voltage represents the binary digit value of 1?
6. Which logic level is typically assigned a value of 1?
7. What is the logic level produced in Figure 1-4(a) when the key is removed?
8. According to Figure 1-5, what is the lowest voltage that would be recognized as a logic 1?
9. According to Figure 1-5, what is the highest voltage that would be recognized as a logic 0?
10. According to Figure 1-5, how would a voltage of 1.0 V be recognized?

1-2 DIGITAL SIGNALS

OUTCOMES

Upon completion of this section, you will be able to:

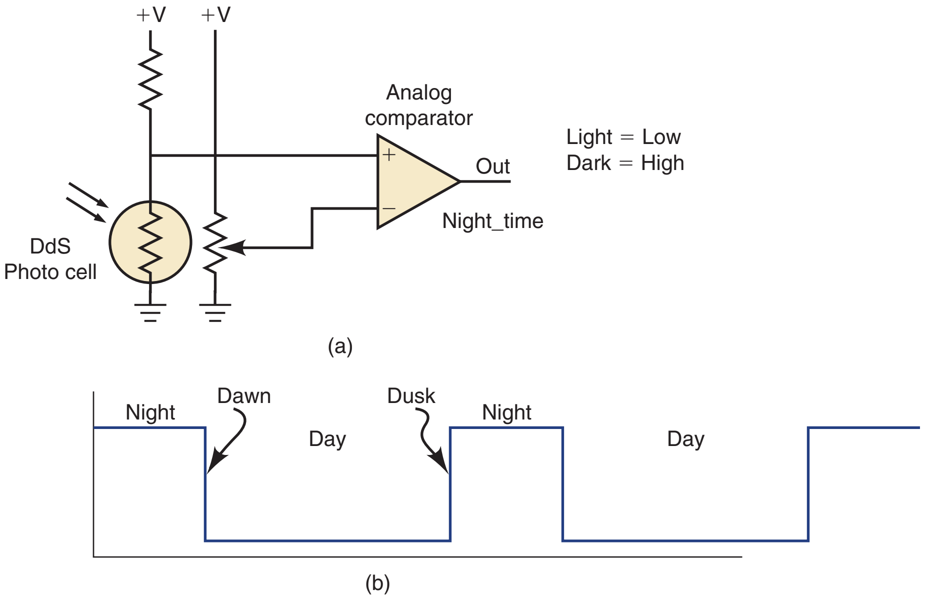
- Determine if a waveform is periodic or not.
- Measure period and frequency.
- Measure duty cycle.
- Identify events and classify edges as rising or falling.
- Recognize valid/invalid inputs.
- Recognize a timing diagram.

Suppose we have a light sensor that is intended to turn on the streetlights at night. An example of a circuit that could perform this task is shown in **Figure 1-7(a)**. Chapter 8 will explain more about analog comparators. This circuit's output will produce a logic 1 when no light is present (darkness). It outputs a logic 0 (0 V) when a certain level of light is present. The signal that comes from the sensor should be labelled with a signal name. It will always be either a 1 (HIGH) or a 0 (LOW) but it should be named something that informs the user about the physical condition represented by the signal. For example, if this sensor is intended to control a street lamp, the name of the output signal should be something like "night_time". When the signal is "1" it is true that it is nighttime. When the output is "0" we can say that it is false that it is nighttime. Chapter 3 will expand on these labelling techniques.

When a circuit like this is placed in service, it will output a 1 at night and a 0 during the day. At some point around dawn, it will change from a 1 to a 0. Around dusk, it will change from a 0 to a 1. This transition between the two states is called an **edge**. *At dawn, when the signal proceeds from HIGH to LOW, it is considered a **falling edge**, or **negative edge***. Graphing the logic state over time tells us something about the operation of the system. **Figure 1-7(b)** shows the graph over time of the output of the light sensor.

*Answers to outcome assessment questions are found at the end of the chapter in which they occur.

FIGURE 1-7 (a) Darkness sensor; (b) a timing diagram of the output.



Need for Timing

Digital circuits have inputs that are in one of two states: 1 or 0. The outputs are also either producing a 1 or a 0. In the previous section, we learned that 1s and 0s are represented by prescribed voltages and that voltage changes on the inputs result in changes in the output voltage. It can be very helpful to show the relationship between changes at the input and changes at the output in order to demonstrate the operation of the system. This means the logic states must be observed over time. Timing diagrams show the relationship, over time, between many digital "signals." It is very important that you understand timing diagrams and can relate them to physical events in a digital circuit. For example, assume there is a circuit represented by the block diagram in **Figure 1-8** that detects the "edge" at dawn, waits 10 minutes, and then turns off the streetlamp. Figure 1-8(b) is a timing diagram which shows the input to the circuit as well as the output. From this diagram, we can determine the relationship between the two signals. Notice the curvy arrows. They are used to indicate the cause-and-effect relationship between input and output signals.

FIGURE 1-8 Timing diagram with input and output.