Digital Design

With an Introduction to the Verilog HDL

FIFTH EDITION

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

PEARSON

Upper Saddle River Boston Columbus San Franciso New York Indianapolis London Toronto Sydney Singapore Tokyo Montreal Dubai Madrid Hong Kong Mexico City Munich Paris Amsterdam Cape Town

سر شناسه : مانو، ام موریت، ۱۹۲۷ - م. Mano, M. Morris عنوان و نام پدیدآور Digital design M. Morris Mano 1 وضعيت ويراست : ويراست بنجم. : تهران: صفار: اشرافی، ۱۳۹۱ ۲۰۱۳ م. مشخصات نشر - ۱۶۴ ص: مصور، جدول. مشخصات نلاهري ISBN 978-0-13-277420-8 وضعیت فهرست نویسی : -انگلیسی. يادداشت كامپيونرهاي رفمي مدارها موضوع مدارهاي منطقي عوضوع موضوع مدارهاي مجتمع رقمي شناسه افزوده چىلتى، مايكل دى. شناسه افزوده Cilletti, Michael D TKYAAA T ATSO ITS رده بندی کنگره رده بندی دیویی شماره کتابشناسی ملی : نام کتاب : Digital Design Michael D. Ciletti - M.Morris Mano : طرح جلد : فرهاد كمالي حروفچینی : معرفت ليتوگرافى : گنج تايگان : گنج شایگان ۱ ۵۵۴۰۲۴۷۸ **شمارگا**ن : ۵۵۰ نیخه ۱۷۰۰۰ ریال

تأليف

چاپخانه

نوبت چاپ : اول- بهار ۱۳۹۱ : انتشارات صفّار - اشراقی ناشر

مرکز پخش 💎 : خیابان انقلاب- روبروی دبیرخانه دانشگاه تهران بازارچه کتاب ط

انتشارات اشراقی (۶۶۴۰۸۴۸۷ خیابان انقلاب- روبروی دبیرخانه دانسگاه تهران بازارچه کتاب- طبعه زبرین

یخش کتاب بینش ۱ ۶۶۴۹۶۲۹۹ كتابغروشي مراديان (۶۶۲۱۵۲۱۰ كتابفروشي صفا FFRYAAFF)

شانک : ۸-۲۲۷۷۴۰ - ۱۲-۰ - ۸۷۸

www.saffarpublishing.com

تلفن گويا: ۹۹۲-۹۹۷

ISBN 978-0-13-277420-8

www.Eshraghi.ir

این اثر، مشمول قانون حمایت مؤلفان و مصنفان و هنرمندان مصوب ۱۳۴۸ است. هرکس تمام یا قسمتی از این اثر را بدون اجازه مؤلف (ناشر) نشر. یا پخش یا عرضه کند مورد پیگرد قانونی قرار خواهد گرفت.

Vice President and Editorial Director, ECS:

Marcia J. Horton

Executive Editor: Andrew Gilfillan

Vice-President, Production: Vince O'Brien

Executive Marketing Manager: Tim Galligan

Marketing Assistant: Jon Bryant

Permissions Project Manager: Karen Sanatar

Senior Managing Editor: Scon Disanno

Production Project Manager/Editorial Production

Manager: Greg Dalles

Cover Designer: Jayne Conte Cover Photo: Michael D. Ciletti Composition: Jouve India Private Limited Full-Service Project Management: Jouve India Private Limited Printer/Binder: Edwards Brothers Typeface: Times Ten 10/12

Copyright © 2013, 2007, 2002, 1991, 1984 Pearson Education, Inc., publishing as Prentice Halls One Lake Street. Upper Saddle River, New Jersey 07458. All rights reserved, Manufactured in the United States of America. This publication is protected by Copyright, and permission should be obtained from the publisher prior to any problem of reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission(s) to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle Wiver, New Jersey 07458.

Many of the designations by manufacturers and seller to distinguish their products are channed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark chann, the designations have been printed in initial caps or all caps.

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

Verilogger Pro and SynaptiCAD are trademarks of Synaptic AD Inc. Blacksburg, VA 24062-0608.

The author and publisher of this book have used their best close in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of the furgishing, performance, or use of these programs.

About the cover: "Spider Rock in Canyon at Chelley," Chinic, Arizona, USA, January 2011, Photograph courtesy of mdc Images, LLC (www.mdcilettiphotography.com). Used by permission.

Library of Congress Cataloging-in-Publication Data

Mano, M. Morris, 1927-

Digital design; with an introduction to the verilog hdl. M. Morris Mano, Michael D. Ciletti. - 5th ed.

n em

Includes index.

ISBN-13: 978-0-13-277420-

ISBN-10:0-12-277420-8

1. Electronic digital Suputers — Circuits, 2. Logic circuits, 3. Logic design, 4. Digital integrated circuits, 1. Ciletti, Michael D. H. Title.

FK7888.3.M343-2011

621,39°5 - dc23

2011039094



Contents

Preface

1	Digi	tal Systems and Binary Numbers		1
	1.1	Digital Systems	1	
	1.2	Binary Numbers	3	
	1.3	Number-Base Conversions	6	
	1.4	Octal and Hexadecimal Numbers	8	
	1.5	Complements of Numbers	10	
	1.6	Signed Binary Numbers	14	
	1.7	Binary Codes	18	
	1.8	Binary Storage and Registers	27	
	1.9	Binary Logic	30	
2	Bool	ean Algebra and Logic Gates		38
	2.1	Introduction	38	
	2.2	Basic Definitions	38	
	2.3	Axiomatic Definition of Boolean Algebra	40	
	2.4	Basic Theorems and Properties of Boolean Algebra	43	
	2.5	Boolean Functions	46	
	2.6	Canonical and Standard Forms	51	
	2.7	Other Logic Operations	58	
	2.8	Digital Logic Gates	60	
	2.9	Integrated Circuits	66	

ix

3	Gate	-Level Minimization		
	3.1	Introduction	73	
	3.2	The Map Method	73	
	3.3	Four-Variable K-Map	80	
	3.4	Product-of-Sums Simplification	84	
	3.5	Don't-Care Conditions	88	
	3.6	NAND and NOR Implementation	90	
	3.7	Other Two-Level Implementations	97	
	3.8	Exclusive-OR Function	103	
	3.9	Hardware Description Language	108	
4	Com	binational Logic	*	125
	4.1	Introduction	125	
	4.2	Combinational Circuits	125	
	4.3	Analysis Procedure	126	
	4.4	Design Procedure	129	
	4.5	Binary Adder-Subtractor	133	
	4.6	Decimal Adder	144	
	4.7	Binary Multiplier	146	
	4.8	Magnitude Comparator	148	
	4.9	Decoders	150	
	4.10	Encoders	155	
	4.11	Multiplexers	158	
	4.12	HDL Models of Combinational Circuits	164	
5	Sync	hronous Sequential Logic		190
	5.1	Introduction	190	
	5.2	Sequential Circuits	190	
	5.3	Storage Elements: Latches	193	
	5.4	Storage Elements: Flip-Flops	196	
	5.5	Analysis of Clocked Sequential Circuits	204	
	5.6	Synthesizable HDL Models of Sequential Circuits	217	
	5.7	State Reduction and Assignment	231	
	5.8	Design Procedure	236	
6	Registers and Counters			255
_	6.1	Registers	255	
	6.2	Shift Registers	258	
	6.3	Ripple Counters	266	
	6.4	Synchronous Counters	271	
	6.5	Other Counters	278	
	6.6	HDL for Registers and Counters	283	
		<u>.</u>		

7.1				Contents	vii
7.2 Random-Access Memory 7.3 Memory Decoding 7.4 Error Detection and Correction 7.5 Read-Only Memory 7.6 Programmable Logic Array 7.7 Programmable Array Logic 7.8 Sequential Programmable Devices 7.8 Sequential Programmable Devices 8 Design at the Register Transfer Level 8.1 Introduction 8.2 Register Transfer Level Notation 8.3 Register Transfer Level in HDL 8.4 Algorithmic State Machines (ASM) 8.5 Design Example (ASM) Chart) 8.6 HDL Description of Design Example 8.7 Sequential Binary Multiplier 8.8 Control Logic 8.9 HDL Description of Buary Multiplier 8.10 Design with Multiplexer 8.11 Race-Free Design (Software Pace Conditions) 8.12 Latch-Free Design (Why Waste Silicon?) 8.13 Other Language Fe. tures 9 Laboratory Faperiments with Standart ICs and FPGAs 9.1 Introduction to Experiments with Standart ICs and FPGAs 9.2 Experiment 2: Digital Logic Gates 9.4 Experiment 4: Combinational Circuits 9.5 Experiment 5: Code Converters 9.6 Experiment 6: Design with Multiplexer 9.7 Experiment 6: Design with Multiplexer 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 8: Flip-Flops 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 11: Shift Registers 9.14 Experiment 11: Shift Registers 9.15 Experiment 11: Shift Registers 9.16 Experiment 11: Shift Registers 9.17 Experiment 11: Shift Registers 9.18 Experiment 11: Shift Registers 9.19 Experiment 11: Shift Registers 9.10 Experiment 11: Shift Registers 9.11 Experiment 11: Shift Registers 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 12: Serial Addition 9.15 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit	7	Mem	ory and Programmable Logic		299
7.3 Memory Decoding 7.4 Error Detection and Correction 312 7.5 Read-Only Memory 315 7.6 Programmable Logic Array 32 7.7 Programmable Array Logic 3.5 7.8 Sequential Programmable Devices 322 8 Design at the Register Transfer Level 8.1 Introduction 3.5 8.2 Register Transfer Level Notation 3.5 8.3 Register Transfer Level in HDL 3.5 8.4 Algorithmic State Machines (ASM) 3.5 Design Example (ASMD Chart) 3.7 8.6 HDL Description of Design Example 3.7 Sequential Binary Multiplier 3.9 8.8 Control Logic 3.96 8.9 HDL Description of oil any Multiplier 3.91 8.10 Design with Multiplexen 3.11 Race-Free Design (Software Pace Conditions) 4.12 Latch-Free Design (Software Pace Conditions) 4.13 Other Language Fe, tures 9 Laboratory Experiments with Standarl ICs and FPGAs 4.3 Experiment 2: Digital Logic Gates 9.4 Experiment 2: Digital Logic Gates 9.5 Experiment 3: Simplification of Boolean Functions 4.8 Experiment 5: Code Converters 9.5 Experiment 5: Code Converters 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.1 Experiment 18: Sinift Registers 9.1 Experiment 19: Sequential Circuits 9.2 Experiment 19: Sequential Circuits 9.3 Experiment 19: Sequential Circuits 9.4 Experiment 19: Sequential Circuits 9.5 Experiment 19: Sequential Circuits 9.6 Experiment 19: Sequential Circuits 9.7 Experiment 19: Sequential Circuits 9.8 Experiment 19: Sequential Circuits 9.9 Experiment 19: Sequential Circuits 9.1 Experiment 11: Shift Registers 9.1 Experiment 12: Experiment 11: Memory Unit		7.1	Introduction	299	
7.3 Memory Decoding 7.4 Error Detection and Correction 7.5 Read-Only Memory 7.6 Programmable Logic Array 7.7 Programmable Logic Array 7.8 Sequential Programmable Devices 7.8 Sequential Programmable Devices 8 Design at the Register Transfer Level 8.1 Introduction 8.2 Register Transfer Level Notation 8.3 Register Transfer Level in HDL 354 8.4 Algorithmic State Machines (ASMs) 8.5 Design Example (ASMD Chart) 8.6 HDL Description of Design Example 8.7 Sequential Binary Multiplier 8.8 Control Logic 8.9 HDL Description of Binary Multiplier 8.10 Design with Multiplexer 8.11 Race-Free Design (Software Race Conditions) 8.12 Latch-Free Design (Software Race Conditions) 8.12 Latch-Free Design (Software Race Conditions) 8.12 Latch-Free Design (Software Race Conditions) 8.13 Other Language Fe, tures 9 Laboratory Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 2: Digital Logic Gates 9.4 Experiment 2: Digital Logic Gates 9.5 Experiment 3: Simplification of Boolean Functions 9.6 Experiment 5: Code Converters 9.7 Experiment 5: Code Converters 9.8 Experiment 5: Code Converters 9.9 Experiment 7: Adders and Subtractors 9.1 Experiment 8: Flip-Flops 9.2 Experiment 8: Flip-Flops 9.3 Experiment 7: Adders and Subtractors 9.4 Experiment 8: Flip-Flops 9.5 Experiment 8: Flip-Flops 9.6 Experiment 7: Adders and Subtractors 9.7 Experiment 19: Sequential Circuits 9.8 Experiment 19: Sequential Circuits 9.9 Experiment 19: Sequential Circuits 9.1 Experiment 11: Shift Registers 9.1 Experiment 11: Shift Registers 9.1 Experiment 11: Strip Registers 9.1 Experiment 11: Shift Registers 9.1 Experiment 11: Strip Registers 9.1 Experiment 12: Serial Addition 9.14 Experiment 12: Serial Addition				300	
7.4 Error Détection and Correction 7.5 Read-Only Memory 7.6 Programmable Logic Array 7.7 Programmable Logic Array 7.7 Programmable Array Logic 7.8 Sequential Programmable Devices 8 Design at the Register Transfer Level 8.1 Introduction 8.2 Register Transfer Level Notation 8.3 Register Transfer Level in HDL 354 8.4 Algorithmic State Machines (ASM) 8.5 Design Example (ASMD Chart) 8.6 HDL Description of Design Example 8.8 Control Logic 8.9 HDL Description of Duany Multiplier 8.10 Design with Multiplexen 8.11 Race-Free Design (Myh Waste Silicon?) 8.12 Latch-Free Design (Myh Waste Silicon?) 8.13 Other Language Features 9 Laboratory Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 5: Code Converters 9.6 Experiment 6: Design with Multiplexers 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.1 Experiment 7: Adders and Subtractors 9.2 Experiment 7: Adders and Subtractors 9.3 Experiment 8: Flip-Flops 9.4 Experiment 8: Flip-Flops 9.5 Experiment 8: Flip-Flops 9.6 Experiment 19: Sequential Circuits 9.7 Experiment 19: Sequential Circuits 9.8 Experiment 19: Sequential Circuits 9.9 Experiment 11: Shift Registers 9.1 Experiment 11: Serial Addition 9.14 Experiment 11: Serial Addition 9.14 Experiment 11: Memory Unit				3 <i>07</i>	
7.5 Read-Only Memory 7.6 Programmable Logic Array 7.7 Programmable Logic 7.8 Sequential Programmable Devices 8 Design at the Register Transfer Level 8.1 Introduction 8.2 Register Transfer Level Notation 8.3 Register Transfer Level in HDL 8.4 Algorithmic State Machines (ASM) 8.5 Design Example (ASMD Chart) 8.6 HDL Description of Design Example 8.7 Sequential Binary Multiplier 8.8 Control Logic 8.9 HDL Description of briany Multiplier 8.10 Design with Multiplexen 8.11 Race-Free Design (Software Pace Conditions) 8.12 Latch-Free Design (Software Pace Conditions) 8.13 Other Language Fe. tures 9 Laboratory Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments with Standard ICs and FPGAs 9.2 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 5: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 7: Adders and Subtractors 9.9 Experiment 7: Adders and Subtractors 9.1 Experiment 7: Adders and Subtractors 9.2 Experiment 7: Adders and Subtractors 9.3 Experiment 7: Adders and Subtractors 9.4 Experiment 8: Flip-Flops 9.5 Experiment 8: Flip-Flops 9.6 Experiment 7: Sequential Circuits 9.7 Experiment 8: Flip-Flops 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 11: Shift Registers 9.11 Experiment 11: Shift Registers 9.12 Experiment 11: Shift Registers 9.13 Experiment 11: Strift Registers 9.14 Experiment 11: Strift Registers 9.15 Experiment 11: Strift Registers 9.16 Experiment 11: Strift Registers 9.17 Experiment 11: Strift Registers 9.18 Experiment 11: Strift Registers 9.19 Experiment 11: Strift Registers 9.19 Experiment 11: Strift Registers 9.10 Experiment 11: Strift Registers 9.11 Experiment 11: Strift Registers 9.12 Experiment 11: Strift Registers				312	
7.6 Programmable Logic Array 7.7 Programmable Array Logic 7.8 Sequential Programmable Devices 325 8 Design at the Register Transfer Level 351 8.1 Introduction 8.2 Register Transfer Level Notation 351 8.3 Register Transfer Level Notation 351 8.4 Algorithmic State Machines (ASMs) 363 8.5 Design Example (ASMD Chart) 371 8.6 HDL Description of Design Example 381 8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Binary Multiplier 402 8.10 Design with Multiplexer 311 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Software Race Conditions) 422 8.13 Other Language Fe, tures 426 9 Laboratory Experiments with Standart ICs and FPGAs 9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 4: Combinational Circuits 450 9.6 Experiment 4: Combinational Circuits 450 9.7 Experiment 7: Adders and Subtractors 452 9.8 Experiment 7: Adders and Subtractors 453 9.9 Experiment 7: Adders and Subtractors 457 9.10 Experiment 10: Counters 461 9.11 Experiment 11: Shift Registers 463 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 11: Shift Registers 463 9.14 Experiment 11: Strift Registers 467 9.14 Experiment 11: Strift Registers 467 9.14 Experiment 11: Strift Registers 467				315	
7.7 Programmable Array Logic 7.8 Sequential Programmable Devices 8 Design at the Register Transfer Level				321	
8 Design at the Register Transfer Level 351 8.1 Introduction 8.2 Register Transfer Level Notation 351 8.3 Register Transfer Level in HDL 354 8.4 Algorithmic State Machines (ASMS) 363 8.5 Design Example (ASMD Chart) 371 8.6 HDL Description of Design Example 381 8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Binary Multiplier 402 8.10 Design with Multiplexen 411 8.11 Race-Free Design (Software Pace Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Fe. tures 426 9 Laboratory Fx. P. iments with Standar I ICs and FPGAs 438 9.1 Introduction to Experiments 438 9.2 E. periment 1: Binary and Decimal Numbers 443 9.3 Exp. riment 2: Digital Logic Gates 446 9.4 Exp. riment 2: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 19: Sequential Circuits 460 9.11 Experiment 11: Shift Registers 463 9.13 Experiment 11: Shift Registers 463 9.14 Experiment 11: Shift Registers 463 9.13 Experiment 11: Shift Registers 467 9.14 Experiment 11: Memory Unit 467			Programmable Array Logic	325	
Transfer Level 351 8.1 Introduction 8.2 Register Transfer Level Notation 351 8.3 Register Transfer Level in HDL 354 8.4 Algorithmic State Machines (ASM) 363 8.5 Design Example (ASMD Chart) 371 8.6 HDL Description of Design Example 381 8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Danny Multiplier 402 8.10 Design with Multiplexer 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Features 426 9 Laboratory Fareliments with Standari ICs and FPGAs 438 9.1 Introduction to Experiments 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 7: Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467				329	
Transfer Level 351 8.1 Introduction 8.2 Register Transfer Level Notation 351 8.3 Register Transfer Level in HDL 354 8.4 Algorithmic State Machines (ASM) 363 8.5 Design Example (ASMD Chart) 371 8.6 HDL Description of Design Example 381 8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Danny Multiplier 402 8.10 Design with Multiplexer 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Features 426 9 Laboratory Fareliments with Standari ICs and FPGAs 438 9.1 Introduction to Experiments 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 7: Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467	8	Desi	an at the Register		
8.2 Register Transfer Level Notation 3.51 8.3 Register Transfer Level in HDL 3.54 8.4 Algorithmic State Machines (ASMS) 3.63 8.5 Design Example (ASMD Chart) 3.71 8.6 HDL Description of Design Example 3.81 8.7 Sequential Binary Multiplier 3.91 8.8 Control Logic 3.96 8.9 HDL Description of Binary Multiplier 4.02 8.10 Design with Multiplexer 4.11 8.11 Race-Free Design (Software Race Conditions) 4.22 8.12 Latch-Free Design (Why Waste Silicon?) 4.25 8.13 Other Language Fe, tures 4.26 9 Laboratory Experiments with Standard ICs and FPGAs 4.38 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 4.43 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 4.6 9.7 Experiment 4: Combinational Circuits 4.6 9.8 Experiment 5: Code Converters 9.8 Experiment 5: Code Converters 9.9 Experiment 6: Design with Multiplexers 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 4.60 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit 4.67	_			•	351
8.2 Register Transfer Level Notation 3.51 8.3 Register Transfer Level in HDL 3.54 8.4 Algorithmic State Machines (ASMS) 3.63 8.5 Design Example (ASMD Chart) 3.71 8.6 HDL Description of Design Example 3.81 8.7 Sequential Binary Multiplier 3.91 8.8 Control Logic 3.96 8.9 HDL Description of Binary Multiplier 4.02 8.10 Design with Multiplexer 4.11 8.11 Race-Free Design (Software Race Conditions) 4.22 8.12 Latch-Free Design (Why Waste Silicon?) 4.25 8.13 Other Language Fe, tures 4.26 9 Laboratory Experiments with Standard ICs and FPGAs 4.38 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 4.43 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 4.6 9.7 Experiment 4: Combinational Circuits 4.6 9.8 Experiment 5: Code Converters 9.8 Experiment 5: Code Converters 9.9 Experiment 6: Design with Multiplexers 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 4.60 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit 4.67		Q 1	Introduction	351	
8.3 Register Transfer Level in HDL 8.4 Algorithmic State Machines (ASMs) 8.5 Design Example (ASMD Chart) 8.6 HDL Description of Design Example 8.7 Sequential Binary Multiplier 8.8 Control Logic 8.9 HDL Description of Binary Multiplier 8.10 Design with Multiplexer 8.11 Race-Free Design (Software Race Conditions) 8.12 Latch-Free Design (Why Waste Silicon?) 8.13 Other Language Fe tures 9 Laboratory Experiments with Standar I ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit 9.14					
8.4 Algorithmic State Machines (ASM) 363 8.5 Design Example (ASMD Chart) 371 8.6 HDL Description of Design Example 381 8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Binary Multiplier 402 8.10 Design with Multiplexers 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Fe, tures 426 9 Laboratory Experiments 438 9.1 Introduction to Experiments 443 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 463 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
8.5 Design Example (ASMD Chart) 8.6 HDL Description of Design Example 8.7 Sequential Binary Multiplier 8.8 Control Logic 8.9 HDL Description of Brany Multiplier 8.10 Design with Multiplexers 8.11 Race-Free Design (Software Race Conditions) 8.12 Latch-Free Design (Why Waste Silicon?) 8.13 Other Language Fe, tures 9 Laboratory Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit 9.15					
8.6 HDL Description of Design Example 381 8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Binary Multiplier 402 8.10 Design with Multiplexer 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Fe. tures 426 9 Laboratory Fxperiments with Standard ICs and FPGAs 438 9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
8.7 Sequential Binary Multiplier 391 8.8 Control Logic 396 8.9 HDL Description of Binary Multiplier 402 8.10 Design with Multiplexer 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Fe. tures 426 9 Laboratory Fareliments with Standard ICs and FPGAs 438 9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
8.8 Control Logic 8.9 HDL Description of B nary Multiplier 8.10 Design with Multiplexers 8.11 Race-Free Design (Software Race Conditions) 8.12 Latch-Free Design (Why Waste Silicon?) 8.13 Other Language Features 9 Laboratory Experiments with Standart ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit					
8.9 HDL Description of Binary Multiplier 402 8.10 Design with Multiplexer 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Features 9 Laboratory Experiments with Standar ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 48 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit					
8.10 Design with Multiplexers 411 8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Features 426 9 Laboratory Experiments with Standard ICs and FPGAs 438 9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467			HDL Description of Rinary Multiplier		
8.11 Race-Free Design (Software Race Conditions) 422 8.12 Latch-Free Design (Why Waste Silicon?) 425 8.13 Other Language Features 9 Laboratory Falenments with Standard ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit 426 438 438 438 438 438 438 438 43					
8.12 Latch-Free Design (Why Waste Silicon?) 8.13 Other Language Features 9 Laboratory Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit			Race-Free Design (Software Race Conditions)		
9 Laboratory Experiments with Standard ICs and FPGAs 9.1 Introduction to Experiments 9.2 Experiment 1: Binary and Decimal Numbers 9.3 Experiment 2: Digital Logic Gates 9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 9.6 Experiment 5: Code Converters 9.7 Experiment 6: Design with Multiplexers 9.8 Experiment 7: Adders and Subtractors 9.9 Experiment 8: Flip-Flops 9.10 Experiment 9: Sequential Circuits 9.11 Experiment 10: Counters 9.12 Experiment 11: Shift Registers 9.13 Experiment 12: Serial Addition 9.14 Experiment 13: Memory Unit					
9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467	9	Labo	ratory Experiments		
9.1 Introduction to Experiments 438 9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					438
9.2 Experiment 1: Binary and Decimal Numbers 443 9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467				430	
9.3 Experiment 2: Digital Logic Gates 446 9.4 Experiment 3: Simplification of Boolean Functions 448 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.4 Experiment 3: Simplification of Boolean Functions 9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467			Experiment 1: Binary and Decimal Numbers		
9.5 Experiment 4: Combinational Circuits 450 9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467			Experiment 2: Digital Logic Gates		
9.6 Experiment 5: Code Converters 452 9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467			experiment 3: Simplification of Boolean Functions		
9.7 Experiment 6: Design with Multiplexers 453 9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.8 Experiment 7: Adders and Subtractors 455 9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.9 Experiment 8: Flip-Flops 457 9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.10 Experiment 9: Sequential Circuits 460 9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.11 Experiment 10: Counters 461 9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.12 Experiment 11: Shift Registers 463 9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.13 Experiment 12: Serial Addition 466 9.14 Experiment 13: Memory Unit 467					
9.14 Experiment 13: Memory Unit 467					

9.16 9.17 9.18 9.19	Experiment 15: Clock-Pulse Generator Experiment 16: Parallel Adder and Accumulator Experiment 17: Binary Multiplier Verilog HDL Simulation Experiments and Rapid Prototyping with FPGAs	473 475 478 480	488
10.1	Rectangular-Shape Symbols	488	
10.2	Qualifying Symbols	491	
10.3	Dependency Notation	493	
10.4	Symbols for Combinational Elements	495	
10.5	Symbols for Flip-Flops	497	
10.6	Symbols for Registers	499	
10.7	Symbols for Counters	502	
10.8	Symbol for RAM	504	
	X		507
•			307
	Sebut of the man		521
			539
_			

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 handfield 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 Dieval Design builds on the previous four editions, and the feedback of the team of eviewers 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 in ended 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 webbased 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

Preface

(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 hirdsign, we note that the relatively high level of illustrations and graphical content of our terouddresses the visual (V) component of VARK: discussions and numerous examples address the reading (R) component. Students who exploit the availability of free smallatory 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 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 industrates havely 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 HD s (cos., 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 byproduct, 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
- Streamining 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.

PUST 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.

VERTFICATION

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 unruduce 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 class room presentation

HDL Simulators

The Companion Website identifies web URLs to we 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 llops 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 chapter 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 enror 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 in portant 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. Then expensise, 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 Nebtaska, 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