

THE ART OF DIGITAL DESIGN

**An Introduction To
Top-Down Design**

**DAVID WINKEL
FRANKLIN PROSSER**

THE ART OF DIGITAL DESIGN

an introduction to top-down design

DAVID WINKEL
University of Wyoming

FRANKLIN PROSSER
Indiana University

PRENTICE-HALL, INC., Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

Winkel, David.

The art of digital design.

Bibliography: p. 489

Includes index.

1. Electronic digital computers—Design and construction. 2. Minicomputers—Design and construction. I. Prosser, Franklin, joint author. II. Title.

TK7888.3.W56 621.3819'58'2 79-18460

ISBN 0-13-046607-7

© 1980 by PRENTICE-HALL, INC.,
Englewood Cliffs, New Jersey 07632

*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.*

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

Editorial/production supervision
and interior design by Linda Mihatov
Cover design by Edsal Enterprises
Manufacturing buyer: Gordon Osbourne

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
PRENTICE HALL OF CANADA, LTD., *Toronto*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
PRENTICE-HALL OF SOUTHEAST ASIA PTE. LTD., *Singapore*
WHITEHALL BOOKS LIMITED, *Wellington, New Zealand*

Preface

This is an introduction to the art of digital hardware design. The design of computer hardware, once the exclusive province of the electrical engineer, is now of vital interest to the computer scientist, whereas techniques for the systematic solution of problems—the computer scientist's specialty—are of increasing importance to the electrical engineer.

Traditional approaches to design, which evolved prior to the integrated circuit, place great emphasis on the hardware devices themselves. In earlier times, this was natural, since the devices were so expensive that hardware costs controlled design. This led to the development of many complex methods of logic minimization, state assignment, handling asynchronous circuits, and so on, that are now little used due to the low cost and great power and flexibility of modern digital components. The complexity of traditional design methods actually can interfere with the designer's ability to create a straightforward, understandable, and correct design!

When we first studied digital design, we consulted traditional textbooks and practiced traditional design methods, but we found that these methods did not really give us much help in solving complex digital hardware problems. A vast body of vital design knowledge was missing from the books, and there was a great lack of systematic methodology for dealing with a digital problem as a system. Eventually, we realized that

the traditional emphasis was misplaced. The difficult part of digital design is not choosing or assembling the hardware devices, but rather is understanding the problem and developing a systematic solution for the system architecture and its control. For this book, we have looked closely at each design technique, and have been ruthless in eliminating methods that do not contribute substantially to the goal of clear and correct design.

Here is our thesis: We must approach hardware design problems from the top, remaining aloof from hardware commitments as long as we can. We must thoroughly understand the problem and must let the problem requirements guide us to suitable hardware, rather than allow premature hardware selections to force us into unsuitable design decisions.

With the realization that the human cost of designing and maintaining a digital system far exceeds the cost of the materials, digital designers are developing a new philosophy toward the design process. This new design approach, which this book exploits, insists that our design tools must materially assist the designer to understand, solve, and document complex hardware and software problems—if necessary, at the cost of additional hardware. The designer's mind must be uncluttered by unnecessary detail. To one experienced in modern computer programming, this has a familiar sound. Software design methodology has undergone drastic improvements in recent years. The software world accepts as valid and powerful such concepts as structured programming and top-down design. This book is a contribution in the same spirit to the field of digital hardware design.

COURSE LEVEL

This book is for self-study and for classroom use. It should benefit the student or professional unsophisticated in digital hardware, yet it provides an opportunity for old hands to come to grips with some modern trends in basic design. For background, we assume that the reader has an elementary knowledge of computer problem solving in a high-level language and has an elementary exposure to the structure of computers and the use of assembly language. The student should be familiar with number systems such as binary, octal, and hexadecimal, and with number representations, particularly two's complement. We assume no prior knowledge of electronics or hardware other than Ohm's Law and simple formulas for series and parallel resistances.

In a college curriculum, this book is suitable for a first course in digital design. The course may be at the middle or upper undergraduate level in computer science or electrical engineering. The text is close to the spirit of the ACM Curriculum 78 course CS4 (Introduction to Computer Organization),[†] and covers over 85 percent of the material suggested

[†]*Communications of the ACM* 22, No. 3 (March 1979), pages 147-165.

therein. For electrical engineers, this text is a modernization of the traditional first course in digital design. The text emphasizes the solution of design problems rather than the study of hardware. It uses modern philosophies advocated in the IEEE proposed courses DL-1 and DL-2.[†]

In sum, the computer science student should feel at home with the structured approach and will be delighted to find that he or she can successfully design and understand complex hardware. The electrical engineering student will gain insight into modern, systematic design style and will develop an understanding of the computer scientist's emphasis on structure.

PLAN OF THE BOOK

This book first provides a thorough foundation for design, emphasizing basic hardwired design. It then introduces microprogramming and microprocessor-based design and prepares the student for further study of these topics. Throughout, we apply the principles of top-down design.

The book has four parts. Three of the parts form a sequence leading systematically from logic fundamentals through digital design with microcomputers, with emphasis on solving digital problems using hardwired structures of the complexity of medium- and large-scale integration (MSI and LSI). The ordering of the topics is from primitive to complex, the natural way. The fourth part is a collection of information on digital technology that the student may read whenever appropriate. Part IV does not depend on the previous parts of the book.

Part I develops the tools for digital design. In these four chapters, we present the theory and formalism required for systematic digital design. Chapter 1 covers the theory of logical expressions—Boolean algebra, truth tables, and useful simplification techniques. Chapter 2 presents the conversion between logic expressions and hardware diagrams, using elementary small-scale integration circuits. This chapter introduces the crucial distinction between voltage and logic, and develops the mixed-logic method for drafting physical circuits from logic expressions. Chapter 3 develops a collection of basic design tools: useful combinational building blocks such as the multiplexer, decoder, and arithmetic logic unit. The treatment emphasizes the systematic uses of the building blocks. Chapter 4 introduces the theory of circuits with memory—the sequential circuits. After establishing the necessary theory and use of basic sequential elements such as flip-flops, we describe standard sequential building blocks such as storage registers, shift registers, random access memories, and read only memories.

Part II explores the art of digital design at the hardwired MSI and LSI level. The theme of these five chapters is that a designer must understand

[†]IEEE Computer 10, No. 12, December 1977, pages 76–79.

the problem before becoming committed to chips and wires—top-down design. Chapter 5 introduces the structure of a digital hardware problem solution—the architecture and the control. We present a hardware flow-chart method for expressing control algorithms that allows clear, systematic synthesis of state generators and output signals. Chapter 6 is a series of digital design examples that form a framework for showing systematic solutions of common design situations at the MSI level. In Ch. 7, we execute a large-scale design—a complete minicomputer—using top-down style. Chapter 8 translates the minicomputer design into hardware, using the principle of deferring the hardware decisions. Chapter 9 introduces a limited asynchronous design technique by implementing a Teletype interface to the minicomputer.

Part III bridges the gap between hardware and software. Here we consider the uses of modern LSI technology—microprogramming, bit slices, microprocessors. Chapter 10 discusses microprogramming and its widespread impact on computer design. As an illustration, we show how to convert the minicomputer of Part II into a microprogrammed version. Chapter 11 is an introduction to the use of conventional microprocessors and microcomputers in digital design. Part III is intended to serve as a broad introduction rather than an in-depth treatment of these topics.

In its single chapter, Part IV contains material on digital technology. The topics in Ch. 12 are independent of previous material; the student may explore the treatment whenever he needs specific hardware information. Representative topics are power systems and power distribution, reading integrated circuit data sheets, handling pull-up resistors, clocks, noise problems, and line driving.

SUGGESTIONS FOR COURSE CONTENT

This book provides abundant material for a semester or two-quarter course at the undergraduate level, and adequate material for a year's study. For more compressed schedules, here are some guidelines for selecting course topics. First, Ch. 12 (Part IV) provides a "buffer" in that the instructor can adjust the degree of coverage of hardware technology without disturbing the main text.

We assume that students who pursue the study of digital design will take subsequent courses that deal in depth with microprogramming and microprocessor-based design. In such event, a compressed schedule using this book might cover Parts I and II, and, if time permits, Ch. 10.

Where the instructor wishes to include all of Part III, a shortened sequence might also use Chs. 1-5; several of the design examples in Ch. 6; Chs. 7 and 8, excluding the treatment of the "operate" microinstruction; and, time permitting, Ch. 9. The remarks at the beginning of Ch. 6 will help in determining which design examples to include.

Laboratory experience is a vital part of any study of digital design. In our courses, taught for several years at the University of Wyoming and Indiana University, the students construct and study the complete mini-computer of Chs. 7, 8, and 9. We believe that lengthy experiments on individual gates, flip-flops, and registers are unnecessary; the laboratory time is better spent working with the design and implementation of more complex systems. Students will master individual components in a natural way while studying the structure of larger systems. In any event, we encourage instructors to provide a digital laboratory to accompany the course, for, after all, no design works until it is built!

We gratefully acknowledge the valuable contributions of our colleague, Daniel Friedman, who served enthusiastically as “reviewer without portfolio.”

An Instructor’s Manual is available for this book, giving suggestions for teaching the course and laboratory, and providing answers to the end-of-chapter exercises.

DAVID WINKEL

FRANKLIN PROSSER

Contents

Preface

ix

Part I

TOOLS FOR DIGITAL DESIGN

1 Describing Logic Equations

3

The Need for Abstraction, Formalism, and Style 4

Style 4; *Abstraction* 5; *Formalism* 5

Logic in Digital Design 6

Logical Constants 6; *Logical Variables* 7; *Truth Tables* 7; *Logical Operators and Truth Tables* 8

Elements of Boolean Algebra 11

Basic Manipulations 11; *Equations from Truth Tables* 14; *Truth Tables from Equations* 18; *Condensing Truth Tables* 19; *Don't-Care Outputs in Truth Tables* 21

Karnaugh Maps 22

Building K-maps 24; *Simplifying with K-maps* 25; *K-map Simplification Blunders* 28; *Other Ways of Reading K-maps* 28

Conclusion 29

Exercises 29

2	Implementing Logic in Hardware	36
	Representing TRUE and FALSE with Physical Devices	36
	Representing AND, OR, and NOT	38
	<i>Mixed Logic</i>	39
	Building and Reading Circuits	51
	<i>Analyzing Mixed-Logic Circuits</i>	51; <i>Synthesizing Mixed-Logic Circuits</i>
	<i>Reading Other Types of Circuit Diagrams</i>	57; <i>Other Common Logic Functions</i>
	<i>60</i>	
	Exercises	62
3	Building Blocks for Digital Design	68
	Integrated Circuit Complexity	69
	Combinational Building Blocks	69
	<i>Combinational and Sequential Circuits</i>	69; <i>The Multiplexer</i>
	<i>The Demultiplexer</i>	75; <i>The Decoder</i>
	<i>The Encoder</i>	80;
	<i>The Comparator</i>	81; <i>A Universal Logic Implementer</i>
	<i>84; The Full Adder</i>	86; <i>The Arithmetic Logic Unit</i>
	<i>89</i>	
	Data Movement	91
	<i>The Bus</i>	92
	Exercises	97
4	Building Blocks with Memory	100
	The Time Element	101
	<i>Hazards</i>	101; <i>Circuits with Feedback</i>
	<i>103</i>	
	Sequential Circuits	104
	<i>Unclocked Sequential Circuits</i>	105; <i>Clocked Sequential Circuits</i>
	<i>109</i>	
	The Basic Clocked Building Blocks	112
	<i>The JK Flip-Flop</i>	112; <i>The D Flip-Flop</i>
	<i>116</i>	
	Register Building Blocks	118
	<i>Data Storage</i>	118; <i>Counters</i>
	<i>118; Shift Registers</i>	123; <i>Three-State Outputs</i>
	<i>125; Bit Slices</i>	126
	Large Memory Arrays	126
	<i>Random Access Memory</i>	126; <i>Read-Only Memory</i>
	<i>131; Programmable Logic Array</i>	134; <i>Nonaddressable Memories</i>
	<i>137</i>	
	Timing Devices	138
	<i>The Single Shot</i>	139; <i>The Delay Line</i>
	<i>140</i>	
	Conclusion	141
	Exercises	141

Part II

THE ART OF DIGITAL DESIGN

5	Design Methods	149
	Elements of Design Style	150
	<i>Top-Down Design</i>	150; <i>Separation of Controller and Architecture</i>
	<i>151;</i>	
	<i>Control-Algorithm-Driven Design</i>	153

Algorithmic State Machines	154				
<i>States and Clocks</i>	155; <i>ASM Chart Notations</i>	156			
Implementing Algorithmic State Machines	160				
<i>Traditional Synthesis from an ASM Chart</i>	160; <i>The Multiplexer Controller Method</i>	163; <i>The One-Hot Method</i>	167		
Design Pitfalls	170				
<i>Clock Skew</i>	171; <i>Asynchronous Inputs and Races</i>	172; <i>Asynchronous ASMs</i>	174; <i>Sidestepping the Pitfalls</i>	175; <i>Debugging Synchronous Systems</i>	175
Conclusion	176				
<i>Summary of Design Guidelines</i>	177				
Exercises	177				

6 Practicing Design

182

Design Example 1: Single Pulser	183												
<i>Algorithmic Solution for the Single Pulser</i>	183; <i>A Combined Architecture-Algorithm Solution</i>	185; <i>A Single-Pulser Building Block</i>	186; <i>Generalizing the Single Pulser</i>	187									
Design Example 2: System Clock	188												
<i>Problem Statement</i>	188; <i>Digesting the Problem</i>	188; <i>The System Clock Algorithm</i>	189; <i>Implementing the Circuit</i>	190; <i>Critique</i>	191								
Design Example 3: Serial Bit Clock	192												
<i>Approaching the Problem</i>	192; <i>Initial Architecture</i>	193; <i>Control Algorithm</i>	194; <i>Implementing the ASM</i>	195									
Design Example 4: Serial-Parallel Data Conversions	197												
<i>Specifying the Problem</i>	198; <i>Building the $P \rightarrow S$ Converter</i>	200; <i>Building The $S \rightarrow P$ Converter</i>	202; <i>Closing</i>	205									
Design Example 5: Adder with Accumulator	205												
<i>Problem Statement</i>	205; <i>Digesting the Problem</i>	206; <i>Preliminary Guess at the Architecture</i>	206; <i>The Control Algorithm</i>	208; <i>Detailed Architecture</i>	210; <i>ASM Synthesis</i>	210							
Design Example 6: Simple Combination Lock	212												
<i>Problem Statement</i>	212; <i>Digesting the Problem</i>	212; <i>Developing the Algorithm</i>	213; <i>Elegant Combination Lock ASM</i>	214; <i>Architecture</i>	217; <i>Implementing the Control</i>	219							
Design Example 7: Black Jack Dealer	221												
<i>Rules of Play for Dealer</i>	222; <i>Problem Statement</i>	222; <i>Digesting the Problem</i>	222; <i>Initial Black Jack Dealer ASM</i>	223; <i>Saving States</i>	225; <i>Algorithm Errors</i>	227; <i>Races, Again</i>	227; <i>Process Synchronization</i>	229; <i>Single Pulser Revisited</i>	230; <i>The Final Black Jack Dealer ASM</i>	231; <i>The Final Black Jack Dealer Architecture</i>	231; <i>Implementing the Control Algorithm</i>	234; <i>Summing up</i>	236
Exercises	237												

7 Designing a Minicomputer

243

PDP-8I Specifications	244	
<i>PDP-8 Memory Addressing</i>	245; <i>PDP-8I Instructions</i>	250

LD15 Architecture	255
<i>Principal Elements of the Architecture</i>	<i>256; Data Paths 258</i>
Preliminary Sketch of LD15 Control	264
<i>Fetch and Execute</i>	<i>265; Investigating Fetch 265</i>
Developing the LD15 ASM Chart	270
<i>Conventions</i>	<i>270; Memory Control 271; State F1 275; State F2 278;</i>
<i>State F3 278; State F4 280; State F5 281; State F6 282; State F7 282;</i>	
<i>State IDLE 282; Execute Processing 285; State E0: Single Instruction</i>	
<i>Switch 285; State E0: Manual Operation Detection 287; Execute Phase:</i>	
<i>Manual Commands 288; Execute Phase: Memory Accessing Instructions 289.</i>	
<i>Execute Phase: IOT Instruction 289; State E0: Operate Microinstructions 294</i>	
Conclusion	301
Exercises	305

8 Building the Minicomputer 310

Preliminaries	310
<i>Auxiliary Variables</i>	<i>310; ASM Chart Location Labels 312</i>
The Data Routing System	313
<i>Data Multiplexer Select Signals</i>	<i>313; ALU Operations 315; Register Load</i>
<i>Signals 317; Link Bit Architecture and Control 318</i>	
State Sequencing System	320
<i>State Generator</i>	<i>320</i>
Special Systems	322
<i>Operate Instruction Priority Control</i>	<i>322; IOP Signal Enabler 326;</i>
<i>Interrupt System Control 327; Manual System 327; Logic Equations 328</i>	
Memory System	328
<i>LD15 Memory Access</i>	<i>328; Memory Unit 329</i>
Finished!	334
Exercises	335

9 Interfacing to the Minicomputer 339

The Teletype	339
PDP-8 Input-Output Protocol	341
<i>From TTY to PDP-8 (DA03)</i>	<i>341; From PDP-8 to TTY (DA04) 342;</i>
Requirements of the LD15-TTY Interface	342
<i>Interface Flags and Interrupts</i>	<i>343; Receive Section (DA03) 343;</i>
<i>Transmit Section (DA04) 343</i>	
Preliminary Interface Architecture	344
<i>The UART</i>	<i>344; Incorporating the UART 346</i>
Interface Control Algorithm	346
<i>Event Clocking</i>	<i>346; Synchronizing Signals to the LD15 and the UART 348;</i>
<i>Receive Circuit Algorithm 348; Transmit Circuit Algorithm 350; Interrupt</i>	
<i>Generation 351</i>	
Implementing the TTY Interface	352

Polishing the LD15 Input-Output	353
<i>Maintaining LD15 Control over the Status Inputs</i>	353; <i>Attaching Several Devices</i>
<i>355</i>	
A Final Word for Part II	356
Exercises	356

Part III

BRIDGING THE HARDWARE-SOFTWARE GAP

10	Microprogrammed Design	361
	Classical Microprogramming	362
	Classical Microprogramming with Modern Technology	365
	<i>Microprogramming with Multiple Qualifiers Per State</i>	366; <i>One Qualifier Per State</i>
	<i>369; Single-Qualifier, Single-Address Microcode</i>	372; <i>Comparison of the Microprogramming Approaches</i>
	<i>377</i>	
	Moving Toward Programming	377
	A Logic Engine	378
	<i>Microcontroller Features</i>	379; <i>Stand-Alone Support System</i>
	<i>379; Designing the Logic Engine Microcontroller</i>	380; <i>Bit Slices</i>
	<i>383; Logic Engine Assembly Language</i>	387; <i>Restoring Mixed-Logic Capabilities</i>
	<i>389</i>	
	A Microprogrammed LD15	392
	Summing Up	395
	Exercises	396
11	Microcomputers in Digital Design	399
	The Computer as a Device Controller	400
	<i>Enter the Microcomputer</i>	401
	The Microcomputer in Digital Design	401
	Microcomputer Components	405
	<i>Processor (MPU)</i>	405; <i>Memory</i>
	<i>406; Data Movement</i>	406; <i>The Support Apparatus</i>
	<i>406</i>	
	Microcomputer Data Flow	407
	<i>Memory-Mapped Input-Output</i>	408; <i>Separate Input-Output</i>
	<i>409; Device Addressing</i>	409; <i>Bus Interface Hardware</i>
	<i>410; Bus Protocols</i>	411
	Microcomputer Input-Output	411
	<i>Programmed Input-Output</i>	412; <i>Direct Memory Access Input-Output</i>
	<i>412; Interrupt-Driven Input-Output</i>	415
	Design Example 1: Wire-Wrap Controller—Pure Software Control	417
	<i>Architecture</i>	418; <i>Controller Specifications</i>
	<i>419; Control Algorithm</i>	420; <i>Wrapping Up</i>
	<i>425</i>	
	Design Example 2: Terminal Multiplexer—Hybrid Hardware-Software Control	425
	<i>Specifications</i>	426; <i>Structure of the Terminal Multiplexer</i>
	<i>428; Microcomputer Control Algorithm</i>	431; <i>Checking Out the System</i>
	<i>435; Alternative Approaches</i>	436
	Conclusion	437
	Exercises	438

12 Meeting the Real World	445
On Transistors and Gates 446	
<i>Charge Flow 446; Metallic Conduction 446; Insulators 446; Semiconductors 447; Diode 447; Bipolar Transistors 448; MOS Transistors 450; Gates from Transistors 451</i>	
Bipolar Logic Families 452	
<i>RTL: Resistor-Transistor Logic 452; TTL: Transistor-Transistor Logic 452; ECL: Emitter-Coupled Logic 454</i>	
Unipolar Logic Families 455	
<i>p-MOS and n-MOS 455; CMOS 455</i>	
Three-State and Open Collector Outputs 456	
<i>Open Collector 456; Three-State Outputs 458</i>	
Integrated Circuit Data Sheets 459	
<i>Electrical Data 460; Input and Output Loadings 462; Noise Margins 463</i>	
The Schmitt Trigger 464	
A Power-On Reset Circuit 466	
Oscillators 467	
<i>Schmitt Trigger Oscillator 467; The 555 Oscillator 468; Crystal-Controlled Oscillators 469; The System Clock 469</i>	
Switch Debouncing 469	
Lamp Drivers 471	
The Optical Coupler 472	
A 20-mA Current Loop Interface for a Teletype 473	
Unused Gate Inputs 475	
Power Supplies 475	
Power Distribution 477	
<i>Losses in Power Distribution Systems 477; Combatting Inductance in Power Distribution Systems 479; Power Supply Bypassing 480</i>	
Noise 481	
<i>Crosstalk 481; Reflections 482; Summary 485</i>	
A Selection of SSI and MSI Integrated Circuits 486	
Bibliography	489
Index	493

part I

***Tools for
Digital Design***

Describing Logic Equations

Before you begin this journey into digital design, it is important that you understand the philosophy that will guide your study. If you have not read the preface, do so now before you go on. The preface discusses the issues that give rise to the need for good style and structure in digital design. Also, the preface contains an outline of the book, which will give you a perspective on where you are heading and how you will get there. It is particularly important that you understand this book's approach to the details of digital hardware. The overriding emphasis is to let the problem solution dictate the hardware, rather than allowing premature hardware commitments to coerce the solution. This conscious suppression of hardware detail during most of the design pays big dividends. Chips and wires and power supplies are still important—they are vital to success, and you will need a good background in many areas of digital technology in order to become an accomplished designer—but too often in digital design the hardware has dominated the solution to the problem. To head off this common malady, we have placed in Part IV of this book almost all the technology needed for your introduction to digital design. This information does not depend on the material in Parts I through III. Read the topics in Part IV as you require or desire them while you are progressing through the first three parts of the text.