



Preface

Introduction

This book is designed for use in a computer organization or computer architecture course typically offered by computer engineering, computer science, electrical engineering, or information systems departments. Such a course would typically be at the junior or senior level, or at an advanced sophomore level. This book is also appropriate for bridge courses for graduate students in computer engineering, computer science, electrical engineering, or information systems.

There are no formal prerequisites for this book. Many students will have taken a course in digital circuits before using this book; although helpful, this is not absolutely necessary. I have included two chapters of background material on digital circuits and finite state machines for students lacking this background. Students do not need to know any specific programming language, but they should be able to follow an algorithm written in pseudo-code.

My Approach

In the past, I've taught the topics in this book using different texts. No text was perfect. Some started at too high a level of complexity, losing students in minutiae and details before they could grasp the fundamental concepts. Others covered material without cohesion between topics. These issues, and others, caused problems for my students. I wrote this book to address these problems and to improve student learning. It is my hope that I've successfully resolved some of the problems found in other texts.

Writing this book made me think about a lot more than the material it covers. I considered different ways to present the material, and different design methods. The methods I use in this book are primarily those that worked best for my students.

I like to include a lot of design work in my courses. Students learn by doing; they gain a better understanding of why things work the way they do. Accrediting boards, especially in engineering, require design work in the curriculum. This book includes a strong design component, starting with relatively simple designs and progressing to more complex designs.

Students seem to prefer a top-down approach to system design; however, studies show that they learn the material better when it is taught bottom-up. To balance these two, I first describe systems top-down. This gives students an overall perspective on the design process, since they know where each piece ultimately fits into the final design. However, I design the systems bottom-up, designing small parts of the system and then integrating them together to form the desired system. I feel that the bottom-up approach provides a better foundation for students to build on concepts learned earlier.

I like to illustrate concepts using simple examples. This allows me to introduce concepts without too many other details that confuse the presentation. For example, the book introduces CPU design with a 4-instruction, Very Simple CPU. After students have gone through this design, the book presents the more complex, 16-instruction Relatively Simple CPU that builds on the design techniques used for the Very Simple CPU while also introducing some more advanced techniques. This CPU is used in other design examples throughout the book.

This book uses the Relatively Simple CPU as a running theme. I start by specifying its instruction set architecture, and then design a simple computer using this CPU. The book then presents the design of the CPU itself, using both hardwired and microcoded control. Later in the book, I use this CPU to illustrate other topics, such as how interrupts are processed within the CPU. Such continuity minimizes the amount of extraneous material that students must learn, allowing them to concentrate on the concepts being taught.

For the topic of CPU design, specifically the control unit of the CPU, I use a finite state machine approach. By specifying the operations necessary to fetch, decode, and execute instructions as a straightforward sequence, I believe this gives students a better understanding of how the CPU processes instructions. It also offers the advantage of partitioning the design into two parts: specifying operations and implementing them.

Finally, I think students learn better when they can relate concepts to real-world systems. I've included real-world examples, both historical and current, in each chapter to meet this student need.

Scope of Coverage

This book covers a wide range of topics, from basic digital circuits through parallel processing. First, let's look at the topics covered in each chapter; then we'll describe possible paths through the text. This book is divided into three parts, each described below.

Digital Logic and Finite State Machines

The first part, *Digital Logic and Finite State Machines*, contains the first two chapters. Chapter 1, *Digital Logic Fundamentals*, introduces the basics of Boolean algebra and digital components, using both combinatorial and sequential logic. It introduces some of the more complex components used in the designs found throughout the text. This chapter includes a real-world section on programmable logic devices. Students use this material in their designs throughout the book.

Chapter 2 provides an *Introduction to Finite State Machines*. Such a chapter isn't typically found in a book on computer organization and architecture, but I felt it was important to have it in this book. The control unit of a microprocessor is a finite state machine; learning the basics of finite state machines makes it easier to understand how microprocessors work and why they work the way they do. Finite state machines are used extensively in the designs of CPUs in Chapters 6 and 7.

Computer Organization and Architecture

The second part of this book, *Computer Organization and Architecture*, covers the following eight chapters. Chapter 3 covers the *Instruction Set Architecture* (ISA) of a microprocessor. When you set out to design a microprocessor, you first determine the tasks it must perform, and then specify the instructions, internal registers, and other ISA components it needs to perform these tasks. The ISA is the first step in the design process.

The text looks at *Computer Organization* in Chapter 4. It examines how the processor connects to memory and input/output devices in a computer. This chapter also examines how physical memory is constructed. It provides a strong foundation on which the rest of this book is built. Two simple computer organizations are presented in this chapter. The first is based on the Relatively Simple CPU, an instructional aid used throughout this text; the other is a real-world design based on Intel's 8085 microprocessor.

Chapter 5 examines *Register Transfer Languages*. RTLs are used throughout the book to design microprocessors and other computer system components. This chapter presents the basic syntax of RTL and some typical designs. This gives students the knowledge needed to design systems found in this book, as well as systems beyond this text. This chapter also introduces VHDL, a hardware description language widely used in digital design.

The next five chapters examine the design of different portions of computer systems in detail. *CPU Design with Hardwired Control* is introduced in Chapter 6. First, this chapter presents the design of a Very Simple CPU from scratch. This illustrates some of the standard CPU design concepts without flooding the students with too many details. Then a more complex but still relatively simple CPU design is presented. The Relatively Simple CPU design uses many of the design techniques used for the Very Simple CPU and also introduces more advanced techniques.

A CPU can have one of two types of control units. Hardwired control was introduced in Chapter 6. In Chapter 7 this book presents *Microsequencer Control Unit Design*. This type of control is used in many advanced microprocessors available today. First, this chapter presents the design of a microsequencer for the Very Simple CPU developed earlier. As in the previous chapter, this introduces the fundamentals of microsequencers without too many extraneous details. Then this chapter presents the design of a microsequencer for the Relatively Simple CPU, again building on the foundation established by the earlier, simpler example while introducing more advanced design techniques. This chapter also examines the internal organization of a microcoded CPU, the Pentium microprocessor.

Chapter 8 examines *Computer Arithmetic*. It looks at different numeric formats and the algorithms to perform arithmetic operations on data in these formats. This chapter examines the hardware to implement these algorithms as well. This chapter also looks at specialized hardware to perform arithmetic operations and the IEEE 754 floating point standard used by all modern processors.

Chapter 9 covers the *Memory Organization* in computer systems. It introduces the memory hierarchy, and examines cache and virtual memory in detail. To illustrate their configuration in a typical computer system, this chapter examines the memory hierarchy of a computer with a Pentium microprocessor which runs Windows NT.

Input/Output Organization is covered in Chapter 10. It examines the basic input/output (I/O) functions, as well as some methods to improve I/O data transfers. Although this chapter covers interrupts, the section on interrupts may be covered earlier without any loss of continuity.

Advanced Topics

The last two chapters comprise the third part of the book, *Advanced Topics*. Chapter 11 covers *Reduced Instruction Set Computing*, or RISC processing. It introduces the rationale for RISC and its main features. As with interrupts in the previous chapter, the material on instruction pipelines in this chapter was written so that it may be covered earlier in the text without any loss of continuity. This chapter also introduces Intel's Itanium microprocessor.

Finally, Chapter 12 provides an *Introduction to Parallel Processing*. It presents the basic organizations and topologies of multiprocessor systems. It examines interprocessor communication and memory organization, and presents parallel algorithms for common functions.

Features

This book includes several features designed to make the material more accessible to students. Some of the features are listed below.

- *Practical Perspectives*. These perspectives help students understand why systems are designed the way they are, and show students how the concepts are

applied to real systems. Examples of these perspectives include why LED displays are designed active low, and how cache memory is organized in the Itanium microprocessor.

- *Historical Perspectives.* These perspectives describe components, systems, or events from the past. They help students gain an understanding by introducing timelines and important events in computer design. Examples of these perspectives include the timeline of Intel's early microprocessors and how the Pentium microprocessor got its name.
- *Real-World Examples.* Each chapter ends with an example of a real-world component or system, or a commonly used standard. This gives the students a better understanding of the concepts covered in the book. These sections examine the internal organization of several microprocessors, ranging from the 25-year-old 8085 to the latest Itanium. They also look at how the topics covered in the chapter are implemented in real-world systems, such as cache and virtual memory management in Pentium/Windows computers. Finally, these sections examine standards used in computer design, such as the IEEE 754 Floating Point Standard and the Universal Serial Bus Standard.

Paths Through the Book

The path an instructor uses for this book depends on the background of the students in the class. If the students have no background in digital logic, they should start at the beginning of the book. They would typically begin by covering Chapters 1, 2, 3, and 4. They could then study all of Chapter 5, or only the sections on RTL, and some or all of Chapter 6. The course can include Chapters 7 and 8, or exclude them without a loss of continuity. Most courses would then cover Chapters 9 and 10. Time permitting, the instructor could cover Chapters 11 and/or 12, or selected topics from these chapters.

If the students have already taken a course in digital design, they can skip Chapter 1 and probably Chapter 2. Some digital design courses do not cover finite state machines, or do so at a cursory level. If this is the case, Chapter 2 can be included in the course or assigned to students as a self-study topic or reference. From here, the course would normally cover Chapters 3 to 7. Chapter 8 is optional, depending on the preference of the instructor. The course would then cover Chapters 9, 10, and 11, and, as an option, some or all of Chapter 12.

Regardless of the path used, the instructor may wish to make some modifications. An instructor may wish to replace some of the real-world sections with other material. For example, Chapter 5 introduces VHDL. If the course using this book is a pre-requisite or co-requisite for a lab which uses Verilog, it would be appropriate to introduce material on Verilog rather than cover the VHDL section. Instructors should note that some of the real-world sections build on earlier sections. Chapters 3, 4, and 6 all use the 8085 microprocessor in their real-world sections, and the Pentium microprocessor introduced in Chapter 7 is used in the real-world computer discussed in Chapter 9.

A couple of topics could fit well in several places in the text. This is especially true for the material on interrupts in Chapter 10 and instruction pipelines in Chapter 11. I wrote this material to be self-contained, so that instructors can cover it along with earlier chapters without breaking the flow of the material. Instructors can cover these topics after Section 6.4, or with their present chapters in the book.

Supplements

There is a variety of supplemental material available for use with this book. Some are available to everyone; others are only available to instructors. All of the following supplements are available at the book's companion web site at <http://www.awl.com/carpinelli>.

Supplements Available to Everyone

- *Relatively Simple CPU Simulator*. This simulator allows students to enter a program, assemble it, and simulate its execution. The simulator animates the flow of data within the CPU to give students a better feel for how the CPU works. The simulator is written as a Java applet that can be run using any Java-enabled web browser on any computing platform. Additional simulators will be regularly added to the book's companion web site.
- *Figures*. All figures in this book have been converted to image files. Instructors may download these files for use in preparing their classroom presentations.

Supplements Available Only to Instructors

- *Solutions Manual*. A solutions manual containing solutions for every problem in this book is available. This manual is for qualified instructors only. It is available through your Addison Wesley sales representative or by sending an e-mail message to aw.cse@awl.com.
- *Instructor's Manual*. An instructor's manual is available for this book. It includes prerequisite topics, outcomes, and a summary for each chapter, as well as author's comments. As with the solutions manual, this manual is for qualified instructors only. It is available through your Addison Wesley sales representative or by sending an e-mail message to aw.cse@awl.com.

Acknowledgments

There is only one author listed for this book, but many people influenced its development and final form. Perhaps the greatest influence on this book is the students who have taken my undergraduate computer organization and architecture courses during the past 15 years. They have given me great feedback about what works and what does not when explaining the topics in this book. I've tried to use their feedback as much as possible in preparing this book because I trust their opinions. I especially thank the students who have tested preliminary versions of some chapters during the past few semesters.

Although I specified the basic functions of the Relatively Simple CPU simulator, it was implemented by four of my students, Aamish Kapadia, Ray Bobrowski, Leo Hendriks, and Benedicto Catalan. They did much more than simply code the simulator; they suggested and implemented several of its features. The simulator has more features and is much more user-friendly because of their efforts.

My colleagues at the New Jersey Institute of Technology were extremely helpful while preparing this book. Sol Rosenstark, who has written several books, provided invaluable advice from the beginning that made writing this book much easier. Tony Lambiase, Ken Sohn, and Joe Strano were always available to discuss the topics in this book, teaching and writing styles, and matters completely unrelated to this book. Our off-campus lunches broke many instances of writer's block.

I developed most, but not all, of the figures in this book. Thanks to Intel Corporation for granting permission to reprint several of their figures, and to Michelle Evans for arranging reprint permissions.

The publishing team at Addison Wesley has been terrific right from the start of this project. My editor, Susan Hartman Sullivan, and assistant editor, Lisa Kalner, provided the guidance and encouragement I needed to complete this book. As a first-time textbook author, I'm sure that I needed more help than more seasoned authors; I could not have completed this book without them. (And thanks for the iguana, Lisa.) Thanks also to Pat Mahtani, Michael Hirsch, Mary Boucher, Jennifer Pelland, Regina Hagen, and Joyce Cosentino of Addison Wesley, and Marilyn Lloyd of Pre-Press Company.

The reviewers for this book were very helpful. Their critiques and suggestions were an important factor in improving the manuscript, both in correcting errors and clarifying the presentation. The reviewers for this book are listed below.

- Murray R. Berkowitz (Towson University)
- Jose G. Delgado-Frias (University of Virginia)
- Ata Elahi (Southern Connecticut State University)
- Hassan Farhat (University of Nebraska)
- John C. Kelly, Jr. (North Carolina A&T State University)
- Rabi Mahapatra (Texas A&M University)
- Robert McIlhenny (California State University, Northridge)
- Paliath Narendran (University at Albany—SUNY)
- Yusuf Ozturk (San Diego State University)
- Ralph L. Place (Ball State University)
- John B. Zuckerman (Texas A&M University)

Finally, I'd like to thank my family and friends for their support and encouragement throughout the writing of this book. Thanks to my wife, Mary, for her love, understanding, and undying support. And thanks to my daughter, Maria, whose arrival during the writing of this book was a most welcome distraction.