Preface

Purpose

Embedded computing systems have grown tremendously in recent years, not only in their popularity, but also in their complexity. This complexity demands a new type of designer, one who can easily cross the traditional border between hardware design and software design. After investigating the availability of courses and textbooks, we felt a new course and accompanying textbook were necessary to introduce embedded computing system design using a unified view of software and hardware. This textbook portrays hardware and software not as different domains, but rather as two implementation options along a continuum of options varying in their design metrics, like cost, performance, power, size, and flexibility.

Three important trends have made such a unified view possible. First, integrated circuit (IC) capacities have increased to the point that both software processors and custom hardware processors now commonly coexist on a single IC. Second, quality compilers and program size increases have led to the common use of processor-independent C, C++, and Java compilers and integrated design environments (IDEs) in embedded system design, significantly decreasing the importance of the focus on microprocessor internals and assembly language programming that dominate most existing embedded system courses and textbooks. Third, synthesis technology has advanced to the point that synthesis tools have become commonplace in the design of digital hardware. Synthesis tools achieve nearly the same for hardware design as compilers achieve in software design: They allow the designer to describe desired functionality in a high-level programming language, and they then automatically generate an efficient custom-hardware processor implementation. The first trend makes the past separation of software and hardware design nearly impossible. Fortunately, the second and third trends enable their unified design, by turning embedded system design, at its highest level, into the problem of selecting and programming (for software), designing (for hardware), and integrating "processors."

Coverage

The first four chapters of this book strive to achieve the goal of presenting hardware and software in a unified way. These chapters stress that computations are carried out by processors. Many types of processors are available, including general-purpose processors

(software), custom single-purpose processors (hardware), standard single-purpose processors (peripherals), and so on. But nevertheless, they are all just processors, differing in their cost, power, performance, design time, flexibility, and so on, but essentially doing the same thing. Chapter 1 provides an overview of embedded systems and their design challenges. We introduce custom single-purpose processors in Chapter 2, emphasizing a top-down technique to digital design amenable to synthesis, picking up where many textbooks on digital design leave off. We introduce general-purpose processors and their use in Chapter 3, expecting this chapter to be mostly review for many readers, and ending by showing how to design a general-purpose processor using the techniques of Chapter 2. Chapter 4 describes numerous standard single-purpose processors (peripherals) common in embedded systems. Chapters 5 and 6 introduce memories and interfacing concepts, respectively, to complete the fundamental knowledge necessary to build basic embedded systems. Chapter 7 provides a digital camera example, showing how we can trade off among hardware, software, and peripherals to achieve implementations that vary in their power, performance, and size. These seven chapters form the core of this book.

Freed from the necessity of covering the nitty-gritty details of a particular microprocessor's internals and assembly language programming, this book includes coverage of some additional embedded systems topics. Chapter 8 describes advanced state machine computation models that are becoming popular when describing complex embedded system behavior. It also introduces the concurrent process model and real-time systems. Chapter 9 gives a basic introduction to control systems, enough to make students aware that a rich theory exists for control systems, and to enable students to determine when an embedded system is an example of a control system. Chapter 10 introduces a variety of popular IC technologies, from which a designer may choose for system implementation. Finally, Chapter 11 highlights various design technologies for building embedded systems, including discussion of hardware/software codesign, a user's introduction to synthesis (from behavioral down to logic levels), and the major trend toward design based on intellectual property (IP).

How to Use This Book

We use this book at the University of California, Riverside, in a one-quarter course called Introduction to Embedded Systems, which follows our introductory course on logic design, and which is taken by all computer science, computer engineering, and electrical engineering students at roughly the sophomore level. This early placement of the course in our curriculum represents our belief that an early unified view of hardware and software can be very beneficial to a student's mindset when later taking more specialized courses. The suggested placement of the course in an undergraduate curriculum is shown in Figure P.1. Our one-quarter course covers Chapters 1–7. We have a second quarter course on embedded systems that covers Chapters 8–12, supplemented with a textbook on real-time systems. A one-semester course might cover Chapters 1–7 plus two or three additional chapters of the instructor's choice.

We anticipate that in most electrical and computer engineering/science curricula, this textbook could be used in place of a processor-specific textbook in an existing course on microprocessor-based system design or microprocessor interfacing, as the lab components of

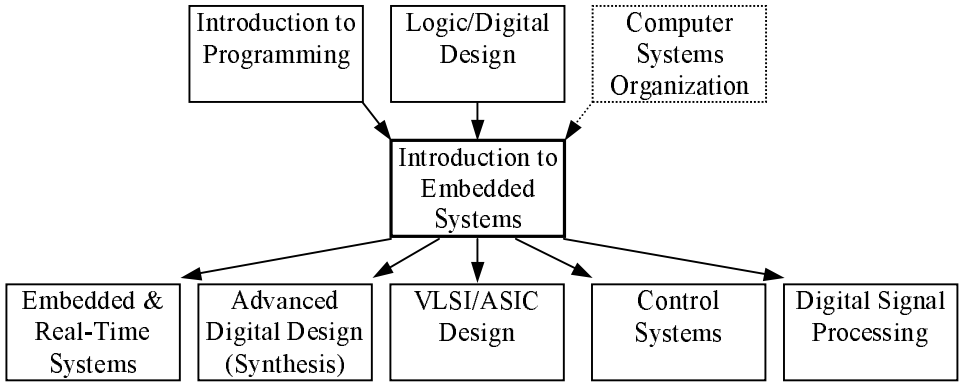

Figure P.1: Embedded systems design placement in a curriculum to create a unified view of hardware and software early.

those courses shift away from assembly-level programming to the use of more modern tools and to the integration of microprocessors and custom hardware (e.g., FPGAs). In other curricula, a new course on embedded systems may be necessary; we observe that numerous universities are introducing such courses, often converting a second course in digital design to a course on embedded systems (as we did at UCR). The book could also be used in a capstone senior design course as a text that brings together and organizes much of what students may have been exposed to already – such courses often do not even have a textbook. The book should also be useful at the graduate level for an introductory embedded systems course.

Laboratory

Ideally, a course using this book should have an accompanying laboratory. The ideal lab setup would include both software development on an embedded microprocessor or microcontroller platform and hardware development on an FPGA platform (or even in a simulation environment).

We intentionally created this book to be independent from any particular microprocessor. One reason is because embedded system tools and products are evolving rapidly, and thus we consider the ability to change lab environments without having to change textbooks an important one. A second reason is because the embedded system field has evolved sufficiently to warrant a book based on principles. However, a course with a hands-on lab may supplement this book with a processor-specific databook, which is typically low cost or even free, or with one of many commonly available "extended databook" processor-specific textbooks in wide use today.

Likewise, the book is independent of any particular hardware description language, synthesis tool, simulator, or FPGA. Supplements that describe the particular hardware environment, again usually available for free or at low cost, may be useful.

At UCR, our labs are based on the 8051 microcontroller and Xilinx FPGAs. We use the Keil C compiler for the microcontroller, Xilinx Foundation Express synthesis software for the FPGA, and a development board from Xess Corporation for prototyping –– the board contains both an 8051 and an FPGA. We also use an 8051 emulator and stand-alone 8051 chips from Philips.

We have provided extensive information on our lab setup and assignments on the book's Web page. Thus, while the book's microprocessor independence enables instructors to choose any lab environment, we have still provided instructors the option of obtaining extensive online assistance in developing an accompanying laboratory.

Additional Materials

A Web page has been established to be used in conjunction with the book: **http://www.cs.ucr.edu/esd**. This Web page contains supplementary material and links for each chapter. It also contains a set of lecture slides in Microsoft PowerPoint format; because the book itself was done entirely in Microsoft Word, the figures in the PowerPoint slides are PowerPoint drawings (rather than imported graphics), and thus can be modified as desired by instructors.

Furthermore, the Web page contains an extensive lab curriculum to accompany this textbook. Over 30 lab exercises, including detailed descriptions, schematics, and complete or partial solutions, can be found there. The exercises are organized by chapter, starting with very simple exercises and leading to progressively more complex ones. For example, Chapter 2's exercises start with a simple blinking light, and end with a soda machine controller and a calculator. Appendix A provides further information on our Web page.

Acknowledgments

We are grateful to numerous individuals for their assistance in developing this book. Sharon Hu of Notre Dame, Nikil Dutt of UC Irvine, and Smita Bakshi of UC Davis and Synplicity provided excellent reviewer feedback. Sharon Hu, Jan Madsen of the Technical University of Denmark, and Enoch Hwang of UC Riverside used early drafts of this book in their embedded systems courses. Susan Cotterell provided numerous contributions to the book, including several examples, much of the accompanying lab materials, and the Web site setup. Jason Villarreal and Kris Miller helped with proofreading at various stages. Jay Farrell of UC Riverside contributed much of the chapter on control systems. Karen Schechter converted our cover design idea into the initial 3-D scene. The generous donations of 8051 equipment from Philips Semiconductors and of FPGA equipment from Xilinx were a big assistance. Likewise, a National Science Foundation CAREER award supported some of this book's development. We thank Caroline Sieg at Wiley for overseeing the book's production and Madelyn Lesure for overseeing the cover design. Finally, we are deeply grateful to Bill Zobrist of Wiley, for believing in this project from its onset, for arranging the reviews of the book, and for overseeing various aspects of its production.

About the Authors

Frank Vahid is an Associate Professor in the Department of Computer Science and Engineering at the University of California, Riverside, which he joined in 1994. He is also a faculty member of the Center for Embedded Computer Systems at the University of California, Irvine. He received his B.S. in Computer Engineering from the University of Illinois, Urbana/Champaign, and his M.S. and Ph.D. degrees in Computer Science from the University of California, Irvine, where he was recipient of the Semiconductor Research Corporation Graduate Fellowship. He was an engineer at Hewlett Packard and has consulted for numerous companies, including NEC and Motorola. He is co-author of the graduate-level textbook *Specification and Design of Embedded Systems* (Prentice-Hall, 1994). He has been program chair and general chair for both the International Symposium on System Synthesis and for the International Symposium on Hardware/Software Codesign. He has been an active researcher in embedded system design since 1988, with more than 50 publications and several best paper awards, including an IEEE Transactions on VLSI best paper award in 2000. His research interests are in embedded system architectures, low-power design, and design methods for system-on-a-chip.

Tony Givargis is an Assistant Professor in the Department of Information and Computer Science and a member of the Center for Embedded Computer Systems at the University of California, Irvine. He received his B.S. and Ph.D. degrees from the University of California, Riverside, where he received the Department of Computer Science Best Thesis award and the UCR College of Engineering Outstanding Student award, and where he was recipient of the GAANN Graduate Fellowship, a MICRO fellowship, and a Design Automation Conference scholarship. As a consultant, he has developed numerous embedded systems for several companies, ranging from an irrigation management system to a GPS-guided, self-navigating automobile. He has published more than 20 research papers in the embedded systems field. His research interests include embedded and real-time system design, low power design, and processor/system-on-a-chip architectures.