Preface

This book is unique in its treatment in that it presents digital logic design from the perspective of computer architecture, starting at the beginning with 1's and 0's and leading through to the design of a microprocessor.

We believe that building a microprocessor is a special rite of passage for engineering and computer science students. The inner workings of a processor seem almost magical to the uninitiated yet prove to be straightforward when carefully explained. Digital design in and of itself is a powerful and exciting subject. Assembly language programming unveils the inner language spoken by the processor. Microarchitecture is the link that brings it all together.

The first two versions of this increasingly popular text cover the MIPS and ARM architectures. As one of the original Reduced Instruction Set Computing architectures, MIPS is clean and exceptionally easy to understand and build. MIPS remains an important architecture, as it has inspired many of the subsequent architectures, including RISC-V. The ARM architecture has exploded in popularity over the past several decades because of its efficiency and rich ecosystem. More than 50 billion ARM processors have been shipped, and more than 75% of humans on the planet use products with ARM processors.

Over the past decade, RISC-V has emerged as an increasingly important architecture, both pedagogically and commercially. As the first widely used open-source computer architecture, RISC-V offers the simplicity of MIPS with the flexibility and features of modern processors.

Pedagogically, the learning objectives of the MIPS, ARM, and RISC-V editions are identical. The RISC-V architecture has a number of features, including extendibility and compressed instructions, that contribute to its efficiency but add a small amount of complexity. The three microarchitectures are also similar, with MIPS and RISC-V architectures sharing many similarities. We expect to offer MIPS, ARM, and RISC-V editions as long as the market demands.

FEATURES

Side-by-Side Coverage of SystemVerilog and VHDL

Hardware description languages (HDLs) are at the center of modern digital design practices. Unfortunately, designers are evenly split between the two dominant languages, SystemVerilog and VHDL. This book introduces HDLs in Chapter 4 as soon as combinational and sequential logic design has been covered. HDLs are then used in Chapters 5 and 7 to design larger building blocks and entire processors. Nevertheless, Chapter 4 can be skipped and the later chapters are still accessible for courses that choose not to cover HDLs.

This book is unique in its side-by-side presentation of SystemVerilog and VHDL, enabling the reader to learn the two languages. Chapter 4 describes principles that apply to both HDLs, and then provides language-specific syntax and examples in adjacent columns. This side-by-side treatment makes it easy for an instructor to choose either HDL and for the reader to transition from one to the other, either in a class or in professional practice.

RISC-V Architecture and Microarchitecture

Chapters 6 and 7 offer in-depth coverage of the RISC-V architecture and microarchitecture. RISC-V is an ideal architecture because it is a real architecture shipped in an increasing number of commercial products, yet it is streamlined and easy to learn. Moreover, because of its popularity in the commercial and hobbyist worlds, simulation and development tools exist for the RISC-V architecture.

Real-World Perspectives

In addition to the real-world perspective in discussing the RISC-V architecture, Chapter 6 illustrates the architecture of Intel x86 processors to offer another perspective. Chapter 9 (available as an online supplement) also describes peripherals in the context of SparkFun's RED-V RedBoard, a popular development board that centers on SiFive's Freedom E310 RISC-V processor. These real-world perspective chapters show how the concepts in the chapters relate to the chips found in many PCs and consumer electronics.

Accessible Overview of Advanced Microarchitecture

Chapter 7 includes an overview of modern high-performance microarchitectural features, including branch prediction, superscalar, and out-of-order operation, multithreading, and multicore processors. The

treatment is accessible to a student in a first course and shows how the microarchitectures in the book can be extended to modern processors.

End-of-Chapter Exercises and Interview Questions

The best way to learn digital design is to do it. Each chapter ends with numerous exercises to practice the material. The exercises are followed by a set of interview questions that our industrial colleagues have asked students who are applying for work in the field. These questions provide a helpful glimpse into the types of problems that job applicants will typically encounter during the interview process. Exercise solutions are available via the book's companion and instructor websites.

ONLINE SUPPLEMENTS

Supplementary materials are available online at ddcabook.com or the publisher's website: https://www.elsevier.com/books-and-journals/book-companion/9780128200643. These companion sites (accessible to all readers) include the following:

- Links to video lectures
- Solutions to odd-numbered exercises
- Figures from the text in PDF and PPTX formats
- Links to professional-strength computer-aided design (CAD) tools from Intel®
- Instructions on how to use PlatformIO (an extension of Visual Studio Code) to compile, assemble, and simulate C and assembly code for RISC-V processors
- Hardware description language (HDL) code for the RISC-V processor
- Intel's Quartus helpful hints
- Lecture slides in PowerPoint (PPTX) format
- Sample course and laboratory materials
- List of errata

The instructor site (accessible to instructors who register at https://inspectioncopy.elsevier.com) includes the following:

- Solutions to all exercises
- Laboratory solutions

EdX MOOC

This book also has a companion Massive Open Online Course (MOOC) through EdX. The course includes video lectures, interactive practice

problems, and interactive problem sets and labs. The MOOC is divided into two parts: Digital Design (ENGR 85A) and Computer Architecture (ENGR85B) offered by HarveyMuddX (on EdX, search for "Digital Design HarveyMuddX" and "Computer Architecture HarveyMuddX"). EdX does not charge for access to the videos but does charge for the interactive exercises and certificate. EdX offers discounts for students with financial need.

HOW TO USE THE SOFTWARE TOOLS IN A COURSE

Intel's Quartus Software

The Quartus software, either Web or Lite Edition, is a free version of Intel's professional-strength Quartus™ FPGA design tools. It allows students to enter their digital designs in schematic or using either the SystemVerilog or the VHDL hardware description language (HDL). After entering the design, students can simulate their circuits using the ModelSim™-Intel FPGA Edition or Starter Edition, which is available with Intel's Quartus software. Quartus also includes a built-in logic synthesis tool that supports both SystemVerilog and VHDL.

The difference between the Web or Lite Edition and the Pro Edition is that the Web or Lite Edition supports a subset of the most common Altera FPGAs. The free versions of ModelSim degrade performance for simulations with more than 10,000 lines of HDL, whereas the professional version of ModelSim does not.

PlatformIO

PlatformIO, which is an extension of Visual Studio Code, serves as a software development kit (SDK) for RISC-V. With the explosion of SDKs for each new platform, PlatformIO has streamlined the process of programming and using various processors by providing a unified interface for a large number of platforms and devices. It is available as a free download and can be used with SparkFun's RED-V RedBoard, as described in the labs provided on the companion website. PlatformIO provides access to a commercial RISC-V compiler and allows students to write both C and assembly programs, compile them, and then run and debug them on SparkFun's RED-V RedBoard (see Chapter 9 and the accompanying labs).

Venus RISC-V Assembly Simulator

The Venus Simulator (available at: https://www.kvakil.me/venus/) is a web-based RISC-V assembly simulator. Programs are written (or copy/pasted) in the Editor tab and then simulated and run in the Simulator tab. Registers and memory contents can be viewed as the program runs.

LABS

The companion site includes links to a series of labs that cover topics from digital design through computer architecture. The labs teach students how to use the Quartus tools to enter, simulate, synthesize, and implement their designs. The labs also include topics on C and assembly language programming using PlatformIO and SparkFun's RED-V RedBoard.

After synthesis, students can implement their designs using the Altera DE2, DE2-115, DE0, or other FPGA board. The labs are written to target the DE2 or DE-115 boards. These powerful and competitively priced boards are available from de2-115.terasic.com. The board contains an FPGA that can be programmed to implement student designs. We provide labs that describe how to implement a selection of designs on the DE2-115 board using the Quartus software.

To run the labs, students will need to download and install Intel's Quartus Web or Lite Edition and Visual Studio Code with the PlatformIO extension. Instructors may also choose to install the tools on lab machines. The labs include instructions on how to implement the projects on the DE2/DE2-115 board. The implementation step may be skipped, but we have found it of great value. We have tested the labs on Windows, but the tools are also available for Linux.

RVfpga

RISC-V FPGA, also referred to as RVfpga, is a free two-course sequence that can be completed after learning the material in this book. The first course shows how to target a commercial RISC-V core to an FPGA, program it using RISC-V assembly or C, add peripherals to it, and analyze and modify the core and memory system, including adding instructions to the core. This course uses the open-source SweRVolf system-on-chip (SoC) (https://github.com/chipsalliance/Cores-SweRVolf), which is based on Western Digital's open-source commercial SweRV EH1 core (https://www.westerndigital.com/company/innovations/risc-v). The course also shows how to use Verilator, an open-source HDL simulator, and Western Digital's Whisper, an open-source RISC-V instruction set simulator (ISS). RVfpga-SoC, the second course, shows how to build an SoC based on SweRVolf using building blocks such as the SweRV EH1 core, interconnect, and memories. The course then guides the user in loading and running the Zephyr operating system on the RISC-V SoC. All necessary software and system source code (Verilog/SystemVerilog files) are free, and the courses may be completed in simulation, so no hardware is required. RVfpga materials are freely available with registration from the Imagination Technologies University Programme: https://university.imgtec.com/rvfpga/.

BUGS

As all experienced programmers know, any program of significant complexity undoubtedly contains bugs. So, too, do books. We have taken great care to find and squash the bugs in this book. However, some errors undoubtedly do remain. We will maintain a list of errata on the book's webpage.

Please send your bug reports to ddcabugs@gmail.com. The first person to report a substantive bug with a fix that we use in a future printing will be rewarded with a $1 bounty!

ACKNOWLEDGMENTS

We appreciate the hard work of Steve Merken, Nate McFadden, Ruby Gammell, Andrae Akeh, Manikandan Chandrasekaran, and the rest of the team at Morgan Kaufmann who made this book happen. We love the art of Duane Bibby, whose cartoons enliven the chapters.

We thank Matthew Watkins, who contributed the section on Heterogeneous Multiprocessors in Chapter 7, and Josh Brake, who contributed to Chapter 9 on Embedded I/O Systems. We greatly appreciate the work of Mateo Markovic and Geordie Ryder, who reviewed the book and contributed to the exercise solutions. Numerous other reviewers also substantially improved the book. They include Daniel Chaver Martinez, Roy Kravitz, Zvonimir Bandic, Giuseppe Di Luna, Steffen Paul, Ravi Mittal, Jennifer Winikus, Hesham Omran, Angel Solis, Reiner Dizon, and Olof Kindgren. We also appreciate the students in our courses at Harvey Mudd College and UNLV who have given us helpful feedback on drafts of this textbook. And last, but not least, we both thank our families for their love and support.