HDL (hardware description language) and FPGA (field-programmable gate array) devices allow designers to quickly develop and simulate a sophisticated digital circuit, realize it on a prototyping device, and verify operation of the physical implementation. As the capacity of FPGA devices continues to grow, a device can accommodate an SoC (system on a chip) design, which integrates a processor, memory modules, I/O peripherals, and custom hardware accelerators into a single chip. This book uses a “learning by doing” approach and illustrates the FPGA and HDL development and design process by a series of examples in the SoC context.

The examples start with simple gate-level circuits, progress gradually through the RT (register-transfer) level modules, and lead to a functional embedded system with custom I/O peripherals and hardware accelerators. A simple SoC framework, FPro (abbreviated from the book title “FPGA Prototyping”), is introduced as a platform to integrate all the design examples together. An FPro system contains a Xilinx MicroBlaze MCS soft-core processor, a video subsystem, and the MMIO (memory-mapped I/O) subsystem that can incorporate custom I/O cores. Except for the processor, all components are designed and coded from scratch. All the hardware and software examples can be synthesized, compiled, and physically tested on the prototyping board.

Focus and audience

Focus The primary focus of this book is on developing efficient and reliable digital systems and effectively using HDL as a tool to describe the intended hardware. The HDL language itself is not the main subject and its coverage is limited to a
small synthesizable subset. The book uses about a dozen proven code templates to provide the skeletal structures of various types of circuits. These templates are general and can easily be integrated to construct a large, complex system. Although this approach limits the “freedom” of syntactic expression, it helps us steer our effort to develop an innovative and efficient hardware architecture.

After discussing the fundamentals in Part I, the book illustrates more complicated and sophisticated designs in the SoC context. Along the way, readers will learn many system-level concepts, including the derivation of a soft-core processor and IP (intellectual property) core based system, the partition and integration of software and hardware, and the development of custom I/O peripherals and hardware accelerators.

Although the book is intended for beginning designers, the examples follow strict design guidelines and prepare readers for future endeavors. The coding and design practice is “forward compatible,” which means that:

- The same practice can be applied to large design in the future.
- The same practice can aid other system development tasks, including simulation, timing analysis, verification, and testing.
- The same practice can be applied to ASIC technology and different types of FPGA devices.
- The code can be accepted by synthesis software from different vendors.

**Audience and prerequisites** The intended audience is students in an advanced digital design course as well as practicing engineers who wish to learn about FPGA- and HDL-based development. Readers need to have a basic knowledge of digital systems, usually a required course in electrical engineering and computer engineering curricula, and a working knowledge of the C/C++ language. Prior exposure to computer architecture, embedded system, and operating system is not necessary but will be helpful.

**Changes for the MicroBlaze MCS SoC Edition**

This book is the successor edition of *FPGA Prototyping with VHDL Examples: Xilinx Spartan 3 Version*. The most significant change is that the new edition presents the hardware in the SoC context and covers many system-level concepts. Instead of treating each module as an isolated entity, the book integrates them into a single coherent SoC platform that allows readers to explore both hardware and software “programmability” and develop complex and interesting embedded system projects. The major revisions in this edition are:

- Add four general-purpose peripheral modules: multi-channel PWM (pulse width modulation), I²C controller, SPI controller, and XADC (Xilinx analog-to-digital converter) controller.
- Introduce a music synthesizer constructed with a DDS (direct digital frequency synthesis) module and an ADSR (attack-decay-sustain-release) envelope generator.
- Expand the original video controller into a complete stream-based video subsystem that incorporates a video synchronization circuit, a test-pattern generator, an OSD (on-screen-display) controller, a sprite generator, and a frame buffer.
• Introduce basic concepts of software-hardware co-design with Xilinx MicroBlaze MCS soft-core processor.
• Provide an overview of the bus interconnect and interface circuit.
• Introduce basic embedded system software development.
• Suggest additional modules and peripherals for interesting and challenging projects.

Logistics

FPGA prototyping board  This book is prepared to be used with the Nexys 4 DDR FPGA prototyping board manufactured by Digilent Inc. It contains an Artix FPGA device and the needed I/O peripherals. All HDL codes and discussions in the book can be applied to this board directly. The less expensive Basys 3 board can be used as well. This board incorporates fewer I/O peripherals and contains a smaller FPGA device.

Most peripherals discussed in the book are de facto industrial standards and the corresponding HDL codes can be used for other FPGA boards as long as they provide adequate analog interface circuits and connectors. Another option is to use stand-alone I/O peripheral modules or to construct the circuits on a breadboard.

Software  The book uses the Xilinx Vivado WebPack edition for hardware development and the Xilinx SDK for software development. Both software packages are free and can be downloaded from Xilinx’s website.

PC accessories  The design examples involve interfaces to several PC peripheral devices, including a USB keyboard, a USB mouse, a VGA compatible monitor, and a powered speaker. These accessories are widely available and probably can be obtained from an old PC.

Book organization

The book is divided into four parts. Part I introduces the elementary HDL constructs and their hardware counterparts, and demonstrates the construction of a basic digital circuit with these constructs. It consists of six chapters:

• Chapter 1 describes the skeleton of an HDL program, the basic language syntax, and the logical operators. Gate-level combinational circuits are derived with these language constructs.
• Chapter 2 provides an overview of an FPGA device, prototyping board, and development flow.
• Chapter 3 introduces HDL’s relational and arithmetic operators and routing constructs. These correspond to medium-sized components, such as comparators, adders, and multiplexers. Module-level combinational circuits are derived with these language constructs.
• Chapter 4 presents the codes for memory elements and the construction of “regular” sequential circuits, such as counters and shift registers, in which the state transitions exhibit a regular pattern.
• Chapter 5 discusses the construction of a finite state machine (FSM), which is a sequential circuit whose state transitions do not exhibit a simple, regular pattern.
Chapter 6 presents the construction of an FSM with data path (FSMD). The
FSMD is used to implement the register-transfer (RT) methodology, in which
the system operation is described by data transfers and manipulations among
registers.
Chapter 7 covers the methods to infer FPGA’s internal memory modules,
which can then be used to construct buffers and lookup tables.

Part II introduces the hardware construction of an FPro system and the develop-
ment of embedded software. A basic “vanilla” FPro system, which contains a
timer core, a UART (universal asynchronous receiver and transmitter) core, a GPI
(general-purpose input) core, and a GPO (general-purpose output) core, is used to
illustrate the key concepts of the process. It consists of four chapters:

Chapter 8 introduces the SoC development and provides an overview of the
hardware organization and software structure of the FPro platform.
Chapter 9 discusses the software development for an embedded system and
the basic coding techniques to access low-level I/O cores.
Chapter 10 covers the FPro bus protocol and the bus interface circuit and
demonstrates the construction of basic GPI, GPO, and timer cores.
Chapter 11 presents the construction of a more sophisticated UART core and
the derivation of software device drivers.

Part III applies the techniques from Parts I and II to develop an array of I/O
cores for the peripherals on the Nexys 4 DDR prototyping board. The I/O cores
are constructed from scratch with custom hardware and device driver. Part III
consists of nine chapters:

Chapter 12 discusses the Xilinx device’s internal analog-to-digital converter
(XADC) and derives an interface circuits to retrieve the analog readings.
Chapter 13 presents the design of a multi-channel PWM core and demon-
strates its application for LED brightness adjustment and servo motor con-
trol.
Chapter 14 converts the seven-segment LED control circuit and the switch
debouncing circuit of Part I into I/O cores and integrates them into an FPro
system.
Chapter 15 provides an overview of the SPI protocol, covers the design of
an SPI controller core, and shows its operation with Nexys 4 DDR board’s
ADXL362 three-axis accelerometer.
Chapter 16 provides an overview of the I²C protocol, discusses the design of
an I²C controller core, and demonstrates its operation with Nexys 4 DDR
board’s ADT7420 temperature sensor.
Chapter 17 covers the design of a PS2 controller core, which can be connected
to a PS2 mouse or a PS2 keyboard, and discusses the device driver routines
to read and decode keyboard’s scan codes and to obtain and process mouse’s
movement information and button activities.
Chapter 18 discusses the construction of a DDFS (direct digital frequency syn-
thesis) controller core with amplitude and frequency modulation and demons-
trates its application as a music synthesizer.
Chapter 19 augments the music synthesizer with an ADSR (attack-decay-
sustain-release) envelope generator core, which can produce sound mimicking
various music instruments.
Part IV discusses the development of a stream-based video subsystem. The subsystem provides a framework to generate and mix multiple video sources into a single video data stream for display. It consists of four chapters:

- Chapter 20 introduces the concept of stream data processing and constructs a basic video system with a test-pattern generator, a color-to-grayscale conversion circuit, and a frame synchronization circuit.
- Chapter 21 provides an overview of the FPPro video subsystem framework and the FPPro video core structure and demonstrates the stream interface with a line buffer.
- Chapter 22 presents the design of a sprite circuit, which adds an overlay of small animated objects on the screen, and applies the technique for a mouse pointer core and a “Pac-Man ghost character” core.
- Chapter 23 discusses the design of an OSD (on-screen-display) controller core, which produces an overlay of text similar to the subtitles on a TV screen.
- Chapter 24 covers the design of a frame buffer, which maintains a bit map for one screen.

The book also includes an Appendix with four tutorials. The tutorials consist of the following:

- Develop, synthesize, and implement a digital circuit on the Nexys 4 DDR board with Vivado.
- Perform simulation of an HDL program with Vivado’s built-in simulator.
- Configure and instantiate Xilinx IP cores.
- Construct a basic FPPro system with a Xilinx microBlaze MCS IP core and develop software with the Xilinx SDK platform.

Companion Website

On an accompanying website (http://academic.csuohio.edu/chu_p), additional information is available, including the following materials:

- Errata
- HDL and C/C++ code listings and relevant files
- Links to synthesis and simulation software
- Links to reference materials

The book contains a number of color figures. They are shown as grayscale in the printed version. These figures can be found on the website as well.

Errata The book is self-prepared, which means that the author has produced all aspects of the text, including illustrations, tables, code listings, indexing, and formatting. As errors are always bound to happen, the accompanying website provides an updated errata sheet and a place to report errors.

P. P. Chu

Cleveland, Ohio
May, 2017