

## PREFACE

---

With the maturity and availability of hardware description language (HDL) and synthesis software, using them to design custom digital hardware has become a mainstream practice. Because of the resemblance of an HDL code to a traditional program (such as a C program), some users believe incorrectly that designing hardware in HDL involves simply writing syntactically correct software code, and assume that the synthesis software can automatically derive the physical hardware. Unfortunately, synthesis software can only perform transformation and local optimization, and cannot convert a poor description into an efficient implementation. Without an understanding of the hardware architecture, the HDL code frequently leads to unnecessarily complex hardware, or may not even be synthesizable.

This book provides in-depth coverage on the systematical development and synthesis of efficient, portable and scalable register-transfer-level (RT-level) digital circuits using the VHDL hardware description language. RT-level design uses intermediate-sized components, such as adders, comparators, multiplexers and registers, to construct a digital system. It is the level that is most suitable and effective for today's synthesis software.

RT-level design and VHDL are two somewhat independent subjects. VHDL code is simply one of the methods to describe a hardware design. The same design can also be described by a schematic or code in other HDLs. VHDL and synthesis software will not lead automatically to a better or worse design. However, they can shield designers from low-level details and allow them to explore and research better architectures.

The emphasis of the book is on *hardware* rather than *language*. Instead of treating synthesis software as a mysterious black box and listing “recipe-like” codes, we explain the relationship between the VHDL constructs and the underlying hardware structure and illustrate how to explore the design space and develop codes that can be synthesized into efficient cell-level implementation. The discussion is independent of technology and can

be applied to both ASIC and FPGA devices. The VHDL codes listed in the book largely follow the IEEE 1076.6 RTL synthesis standard and can be accepted by most synthesis software. Most codes can be synthesized without modification by the free “demo-version” synthesis software provided by FPGA vendors.

**Scope** The book focuses primarily on the design and synthesis of RT-level circuits. A subset of VHDL is used to describe the design. The book is not intended to be a comprehensive ASIC or FPGA book. All other issues, such as device architecture, placement and routing, simulation and testing, are discussed exclusively from the context of RT-level design.

**Unique features** The book is a hardware design text. VHDL and synthesis software are used as tools to realize the intended design. Several unique features distinguish the book:

- Suggest a coding style that shows a clear relationship between VHDL constructs and hardware components.
- Use easy-to-understand conceptual diagrams, rather than cell-level netlists, to explain the realization of VHDL codes.
- Emphasize the reuse aspect of the codes throughout the book.
- Consider RT-level design as an integral part of the overall development process and introduce good design practices and guidelines to ensure that an RT-level description can accommodate future simulation, verification and testing needs.
- Make the design “technology neutral” so that the developed VHDL code can be applied to both ASIC and FPGA devices.
- Follow the IEEE 1076.6 RTL synthesis standard to make the codes independent of synthesis software.
- Provide a set of synthesis guidelines at the end of each chapter.
- Contain a large number of non-trivial, practical examples to illustrate and reinforce the design concepts, procedures and techniques.
- Include two chapters on realizing sequential algorithms in hardware (known as “register transfer methodology”) and on designing control path and data path.
- Include two chapters on the scalable and parameterized designs and coding.
- Include a chapter on the synchronization and interface between multiple clock domains.

**Book organization** The book is basically divided into three major parts. The first part, Chapters 1 to 6, provides a comprehensive overview of VHDL and the synthesis process, and examines the hardware implementation of basic VHDL language constructs. The second part, Chapters 7 to 12, covers the core of the RT-level design, including combinational circuits, “regular” sequential circuits, finite state machine and circuits designed by register transfer methodology. The third part, Chapters 13 to 16, covers the system issues, including the hierarchy, parameterized and scalable design, and interface between clock domains. More detailed descriptions of the chapters follow.

- Chapter 1 presents a “big picture” of digital system design, including an overview on device technologies, system representation, development flow and software tools.
- Chapter 2 provides an overview on the design, usage and capability of a hardware description language. A series of simple codes is used to introduce the basic modeling concepts of VHDL.
- Chapter 3 provides an overview of the basic language constructs of VHDL, including lexical elements, objects, data types and operators. Because VHDL is a strongly typed language, the data types and operators are discussed in more detail.

- Chapter 4 covers the syntax, usage and implementation of concurrent signal assignment statements of VHDL. It shows how to realize these constructs by multiplexing and priority routing networks.
- Chapter 5 examines the syntax, usage and implementation of sequential statements of VHDL. It shows the realization of the sequential statements and discusses the caveats of using these statements.
- Chapter 6 explains the realization of VHDL operators and data types, provides an in-depth overview on the synthesis process and discusses the timing issue involved in synthesis.
- Chapter 7 covers the construction and VHDL description of more sophisticated combinational circuits. Examples show how to transform conceptual ideas into hardware, and illustrate resource-sharing and circuit-shaping techniques to reduce circuit size and increase performance.
- Chapter 8 introduces the synchronous design methodology and the construction and coding of synchronous sequential circuits. Basic “regular” sequential circuits, such as counters and shift registers, in which state transitions exhibit a regular pattern, are examined.
- Chapter 9 explores more sophisticated regular sequential circuits. The design examples show the implementation of a variety of counters, the use of registers as fast, temporary storage, and the construction of pipelined combinational circuits.
- Chapter 10 covers finite state machine (FSM), which is a sequential circuit with “random” transition patterns. The representation, timing and implementation issues of FSMs are studied with an emphasis on its use as the control circuit for a large, complex system.
- Chapter 11 introduces the register transfer methodology, which describes system operation by a sequence of data transfers and manipulations among registers, and demonstrates the construction of the data path (a regular sequential circuit) and the control path (an FSM) used in this methodology.
- Chapter 12 uses a variety of design examples to illustrate how the register transfer methodology can be used in various types of problems and to highlight the design procedure and relevant issues.
- Chapter 13 features the design hierarchy, in which a system is gradually divided into smaller parts. Mechanisms and language constructs of VHDL used to specify and configure a hierarchy are examined.
- Chapter 14 introduces parameterized design, in which the width and functionality of a circuit are specified by explicit parameters. Simple examples illustrate the mechanisms used to pass and infer parameters and the language constructs used to describe the replicated structures.
- Chapter 15 provides more sophisticated parameterized design examples. The main focus is on the derivation of efficient parameterized RT-level modules that can be used as building blocks of larger systems.
- Chapter 16 covers the effect of a non-ideal clock signal and discusses the synchronization of an asynchronous signal and the interface between two independent clock domains.

**Audience** The intended audience for the book is students in advanced digital system design course and practicing engineers who wish to sharpen their design skills or to learn the effective use of today’s synthesis software. Readers need to have basic knowledge of digital systems. The material is normally covered in an introductory digital design course,

which is a standard part in all electrical engineering and computer engineering curricula. No prior experience on HDL or synthesis is needed.

Verilog is another popular HDL. Since the book emphasizes hardware and methodology rather than language constructs, readers with prior Verilog experience can easily follow the discussion and learn VHDL along the way. Most VHDL codes can easily be translated into the Verilog language.

**Web site** An accompanying web site ([http://academic.csuohio.edu/chu\\_p/rtl](http://academic.csuohio.edu/chu_p/rtl)) provides additional information, including the following materials:

- Errata.
- Summary of coding guidelines.
- Code listing.
- Links to demo-version synthesis software.
- Links to some referenced materials.
- Frequently asked questions (FAQ) on RTL synthesis.
- Lecture slides for instructors.

**Errata** The book is “self-prepared,” which means the author has prepared all materials, including the illustrations, tables, code listing, indexing and formatting, by himself. As the errors are always bound to happen, the accompanying web site provides an updated errata sheet and a place to report errors.

P. P. CHU

*Cleveland, Ohio*  
*January 2006*