国外电子信息精品著作(影印版)

# SystemVerilog 数字系统设计

# Digital System Design with SystemVerilog

Mark Zwolinski

科学出版社

国外电子信息精品著作(影印版)

Digital System Design with SystemVerilog

# SystemVerilog 数字系统设计

Mark Zwolinski

科学出版社

北京

图字:01-2012-2732

## 内 容 简 介

SystemVerilog是21世纪电子设计师必须掌握的最重要的语言之一,因为它是设计和验证复杂电子系统核心芯片的重要手段。本书是用SystemVerilog语言设计并验证数字系统的基本概念和具体方法。在介绍基本语法的基础上,阐述了如何用SystemVerilog构成数字电路、组件和系统,以及应该如何使用SystemVerilog搭建测试平台,并对设计进行验证。

本书既适合作电子、自动化和计算机专业本科生和研究生的教科书,也适合已经掌握Verilog和VHDL硬件描述语言的工程师使用。



**图书在版编目(CIP)数据**

SystemVerilog数字系统设计=Digital System Design with SystemVerilog:英文/(美)马克(Mark Z.)编著. —影印版. —北京:科学出版社, 2012

(国外电子信息精品著作)

ISBN:978-7-03-034380-2

Ⅰ.S… Ⅱ.马… Ⅲ.数字系统-系统设计-英文 Ⅳ.TP271

中国版本图书馆CIP数据核字(2012)第103154号

责任编辑:张宇 余丁 / 责任印制:张倩 / 封面设计:陈敬

科学出版社出版

北京东黄城根北街16号

邮政编码:100717

http://www.sciencep.com

源海印刷有限责任公司印刷

科学出版社发行 各地新华书店经销

*

2012年6月第 一 版 开本:787×1092 1/16

2012年6月第一次印刷 印张:25

字数:551 000

**定价:80.00元**

(如有印装质量问题,我社负责调换)

# 《国外电子信息精品著作》序

20世纪90年代以来，信息科学技术成为世界经济的中坚力量。随着经济全球化的进一步发展，以微电子、计算机、通信和网络技术为代表的信息技术，成为人类社会进步过程中发展最快、渗透性最强、应用面最广的关键技术。信息技术的发展带动了微电子、计算机、通信、网络、超导等产业的发展，促进了生命科学、新材料、能源、航空航天等高新技术产业的成长。信息产业的发展水平不仅是社会物质生产、文化进步的基本要素和必备条件，也是衡量一个国家的综合国力、国际竞争力和发展水平的重要标志。在中国，信息产业在国民经济发展中占有举足轻重的地位，成为国民经济重要支柱产业。然而，中国的信息科学支持技术发展的力度不够，信息技术还处于比较落后的水平，因此，快速发展信息科学技术成为我国迫在眉睫的大事。

要使我国的信息技术更好地发展起来，需要科学工作者和工程技术人员付出艰辛的努力。此外，我们要从客观上为科学工作者和工程技术人员创造更有利于发展的环境，加强对信息技术的支持与投资力度，其中也包括与信息技术相关的图书出版工作。

从出版的角度考虑，除了较好较快地出版具有自主知识产权的成果外，引进国外的优秀出版物是大有裨益的。洋为中用，将国外的优秀著作引进到国内，促进最新的科技成就迅速转化为我们自己的智力成果，无疑是值得高度重视的。科学出版社引进一批国外知名出版社的优秀著作，使我国从事信息技术的广大科学工作者和工程技术人员能以较低的价格购买，对于推动我国信息技术领域的科研与教学是十分有益的事。

此次科学出版社在广泛征求专家意见的基础上，经过反复论证、仔细遴选，共引进了接近30本外版书，大体上可以分为两类，第一类是基础理论著作，第二类是工程应用方面的著作。所有的著作都涉及信息领域的最新成果，大多数是2005年后出版的，力求“层次高、

内容新、参考性强”。在内容和形式上都体现了科学出版社一贯奉行的严谨作风。

当然，这批书只能涵盖信息科学技术的一部分，所以这项工作还应该继续下去。对于一些读者面较广、观点新颖、国内缺乏的好书还应该翻译成中文出版，这有利于知识更好更快地传播。同时，我也希望广大读者提出好的建议，以改进和完善丛书的出版工作。

总之，我对科学出版社引进外版书这一举措表示热烈的支持，并盼望这一工作取得更大的成绩。

中国科学院院士
中国工程院院士
2006 年 12 月

# Preface

## About This Book

When *Digital System Design with VHDL* was published, the idea of combining a text on digital design with one on a hardware description language seemed novel. At about the same time, several other books with similar themes were published. *Digital System Design with VHDL* has now been adopted by several universities as a core text and has been translated into Polish, Chinese, Japanese, and Italian. I had thought about writing *Digital System Design with Verilog*, but I had (and still have) some doubts about using Verilog as a teaching language despite its widespread use. Soon after the second edition of *Digital System Design with VHDL* was published, a new hardware description language appeared—SystemVerilog. This new language removed many of my doubts about Verilog and even offered some noticeable advantages over VHDL. So the success of the first book and the appearance of the new language convinced me that the time had come for a new edition.

This book is intended as a student textbook for both undergraduate and postgraduate students. The majority of Verilog and SystemVerilog books are aimed at practicing engineers. Therefore, some features of SystemVerilog are not described at all in this book. Equally, aspects of digital design are covered that would not be included in a typical SystemVerilog book.

Syllabuses for electrical, electronic, and computer engineering degrees vary between countries and between universities or colleges. The material in this book has been developed over a number of years for second- and third-year undergraduates and for postgraduate students. It is assumed that students will be familiar with the principles of Boolean algebra and combinational logic design. At the University of Southampton, UK, the first-year undergraduate syllabus also includes introductions to synchronous sequential design and programmable logic. This book therefore builds upon these foundations. It has often been assumed that topics such as SystemVerilog are too specialized for second-year teaching and are best left to final year or postgraduate courses. There are several good reasons why SystemVerilog

should be introduced earlier into the curriculum. With increasing integrated circuit complexity, there is a need for graduates with knowledge of SystemVerilog and the associated design tools. If left to the final year, there is little or no time for the student to apply such knowledge in project work. Second, conversations with colleagues from many countries suggest that today's students are opting for computer science or computer engineering courses in preference to electrical or electronic engineering. SystemVerilog offers a means to interest computing-oriented students in hardware design. Finally, simulation and synthesis tools and FPGA design kits are now mature and available relatively inexpensively to educational establishments on PC platforms.

## Structure of This Book

Chapter 1 introduces the ideas behind this book, namely the use of electronic design automation tools and CMOS and programmable logic technology. We also consider some engineering problems, such as noise margins and fan-out. In Chapter 2, the principles of Boolean algebra and combinational logic design are reviewed. The important matter of timing and the associated problem of hazards are discussed. Some basic techniques for representing data are discussed.

SystemVerilog is introduced in Chapter 3 through basic logic gate models. The importance of documented code is emphasized. We show how to construct netlists of basic gates and how to model delays through gates. We also discuss parameterized models. The idea of using SystemVerilog to verify models with testbenches is introduced.

In Chapter 4, a variety of modeling techniques are described. Combinational building blocks, buffers, decoders, encoders, multiplexers, adders, and parity checkers are modeled using a range of concurrent and sequential SystemVerilog coding constructs. The SystemVerilog models of hardware introduced in this chapter and in Chapters 5, 6, and 7 are, in principle, synthesizable, although discussion of exactly what is supported is deferred until Chapter 10. Testbench design styles are again discussed in Chapter 4. In addition, the IEEE dependency notation is introduced.

Chapter 5 introduces various sequential building blocks: latches, flip-flops, registers, counters, memory, and a sequential multiplier. The same style as Chapter 4 is used, with IEEE dependency notation, testbench design, and the introduction of SystemVerilog coding constructs.

Chapter 6 is probably the most important chapter of the book and discusses what might be considered the cornerstone of digital design: the design of finite state machines. The ASM chart notation is used. The design process from ASM chart to

D flip-flops and next state and output logic is described. SystemVerilog models of state machines are introduced.

In Chapter 7, the concepts of the previous three chapters are combined. The ASM chart notation is extended to include coupled state machines and registered outputs, and hence to datapath-controller partitioning. From this, we explain the idea of instructions in hardware terms and go on to model a very basic microprocessor in SystemVerilog. This provides a vehicle to introduce interfaces and packages.

The design of testbenches is discussed in more detail in Chapter 8. After recapping the techniques given in earlier chapters, we go on to discuss testbench architecture, constrained random test generation, and assertion-based verification.

SystemVerilog remains primarily a modeling language. Chapter 9 describes the operation of a SystemVerilog simulator. The idea of event-driven simulation is first explained, and the specific features of a SystemVerilog simulator are then discussed.

The other, increasingly important, role of SystemVerilog is as a language for describing synthesis models, as discussed in Chapter 10. The dominant type of synthesis tool available today is for RTL synthesis. Such tools can infer the existence of flip-flops and latches from a SystemVerilog model. These constructs are described. Conversely, flip-flops can be created in error if the description is poorly written, and common pitfalls are described. The synthesis process can be controlled by constraints. Because these constraints are outside of the language, they are discussed in general terms. Suitable constructs for FPGA synthesis are discussed. Finally, behavioral synthesis, which promises to become an important design technology, is briefly examined.

Chapters 11 and 12 are devoted to the topics of testing and design for test. This area has often been neglected, but is now recognized as being an important part of the design process. In Chapter 11, the idea of fault modeling is introduced. This is followed by test generation methods. The efficacy of a test can be determined by fault simulation.

In Chapter 12, three important design-for-test principles are described: scan path, built-in self-test (BIST), and boundary scan. This has always been a very dry subject, but a SystemVerilog simulator can be used, for example, to show how a BIST structure can generate different signatures for fault-free and faulty circuits.

We use SystemVerilog as a tool for exploring anomalous behavior in asynchronous sequential circuits in Chapter 13. Although the predominant design style is currently synchronous, it is likely that digital systems will increasingly consist of synchronous circuits communicating asynchronously with each other. We introduce the concept of the fundamental mode and show how to analyze and design

asynchronous circuits. We use SystemVerilog simulations to illustrate the problems of hazards, races, and setup and hold time violations. We also discuss the problem of metastability.

The final chapter introduces Verilog-AMS and mixed-signal modeling. Brief descriptions of digital-to-analog converters (DACs) and analog-to-digital converters (ADCs) are given. Verilog-AMS constructs to model such converters are given. We also introduce the idea of a phase-locked loop (PLL) here and give a simple mixed-signal model.

The Appendix briefly describes how SystemVerilog differs from earlier versions of Verilog.

At the end of each chapter a number of exercises have been included. These exercises are almost secondary to the implicit instruction in each chapter to simulate and, where appropriate, synthesize each SystemVerilog example. To perform these simulation and synthesis tasks, the reader may have to write his or her own testbenches and constraints files. The examples are available on the Web at zwolinski.org.

## How to Use This Book

Obviously, this book can be used in a number of different ways, depending on the level of the course. At the University of Southampton, I have been using the material as follows.

### Second Year of MEng/BEng in Electronic Engineering

Chapters 1 and 2 are review material, which the students would be expected to read independently. Lectures then cover the material of Chapters 3 through 7. Some of this material can be considered optional, such as Sections 5.3 and 5.7. Additionally, some constructs could be omitted if time is limited. The single-stuck fault model of Section 11.2 and the principles of test pattern generation in Section 11.3, together with the principles of scan design in Section 12.2, would also be covered in lectures.

### Third Year of MEng/BEng in Electronic Engineering

Students would be expected to independently re-read Chapters 4 to 7. Lectures would cover Chapters 8 to 13. Verilog-AMS, Chapter 14, is currently covered in a fourth-year module.

In all years, students need to have access to a SystemVerilog simulator and an RTL synthesis tool in order to use the examples in the text. In the second year, a group

design exercise involving synthesis to an FPGA would be an excellent supplement to the material. In the third year at Southampton, all students do an individual project. Some of the individual projects will involve the use of SystemVerilog.

## Web Resources

A Web site accompanies *Digital System Design with SystemVerilog* by Mark Zwolinski. Visit the site at zwolinski.org. Here you will find valuable teaching and learning material including all the SystemVerilog examples and links to sites with SystemVerilog tools.

# Acknowledgments

I would like to thank all those who pointed out errors in the VHDL versions of this book.

I would also like to thank everyone involved in the commissioning and preparation of this book: Bernard Goodwin and Elizabeth Ryan at Prentice Hall, Madhu Bhardwaj and Ben Kolstad at Glyph International, Susan Fox-Greenberg, who copy edited the text, Danielle Shaw for proof-reading, and Jack Lewis for indexing. Any errors are, however, my fault and not theirs!

Finally, I would like to thank several cohorts of students to whom I have delivered this material and whose comments have encouraged me to think about better ways of explaining these ideas.

# About the Author

***Mark Zwolinski*** is a full professor in the School of Electronics and Computer Science, University of Southampton, United Kingdom. He is the author of *Digital System Design with VHDL*, which has been translated into four languages and widely adopted as a textbook in universities worldwide. He has published over 120 refereed papers in technical journals and has been teaching digital design and design automation to undergraduate and graduate students for twenty years.

# Contents