

HZ BOOKS  
华章科技

国内首本

基于SystemVerilog验证语言的经典之作

资深IC设计工程师及专业社区

联袂力荐

钟文枫◎编著



THE ART OF VERIFICATION WITH SYSTEMVERILOG

SystemVerilog

与功能验证

面向对象的设计语言——SystemVerilog引入到IC设计验证领域，将极大提高验证的效率，成为未来的主流



机械工业出版社  
China Machine Press

2017  
北京航空航天大学出版社

北京市东城区东直门内大街2号

THE ART OF VERIFICATION WITH SYSTEMVERILOG

# SystemVerilog 与功能验证

钟文枫◎编著



TP312

Z731



机械工业出版社  
China Machine Press



本书重点介绍硬件设计描述和验证语言 SystemVerilog 的基本语法及其在功能验证上的应用；书中以功能验证为主线，讲述基本的验证流程、高级验证技术和验证方法学，以 SystemVerilog 为基础结合石头、剪刀、布的应用实例，重点阐述了如何采用 SystemVerilog 实现随机激励生成、功能覆盖率驱动验证、断言验证等多种高级验证技术；最后，通过业界流行的开放式验证方法学 OVM 介绍如何在验证平台中实现可重用性。

封底无防伪标均为盗版

版权所有，侵权必究

本书法律顾问 北京市展达律师事务所

### 图书在版编目 (CIP) 数据

SystemVerilog 与功能验证/钟文枫编著. —北京：机械工业出版社，2010. 8

ISBN 978-7-111-31373-1

I. S… II. 钟… III. 硬件描述语言, SystemVerilog - 程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2010) 第 140823 号

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑：张少波

北京诚信伟业印刷有限公司印刷

2010 年 10 月第 1 版第 1 次印刷

185mm × 260mm · 14 印张

标准书号：ISBN 978-7-111-31373-1

定价：36.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：(010) 88378991；88361066

购书热线：(010) 68326294；88379649；68995259

投稿热线：(010) 88379604

读者信箱：hzjsj@hzbook.com

在读者深入阅读本书之前，我先对本书的主要结构和内容做个简要的介绍，以便不同背景的读者能够有选择地阅读，快速获取自己需要的知识。

## 本书的结构和内容

本书分为 11 章，系统论述了基于 SystemVerilog 的功能验证方法，重点关注以下三个方面的内容。

- 功能验证在整个 FPGA/ASIC 设计流程中的作用及主要的验证技术和方法学。
- SystemVerilog 的语法结构和在功能验证上的应用及基本原理。
- 如何采用 SystemVerilog 搭建验证平台。

第 1 章从 FPGA/ASIC 整个设计流程的角度介绍功能验证的地位和作用、验证的基本流程、验证的主要技术和方法学，最后引入硬件验证语言 SystemVerilog。

第 2 章介绍 SystemVerilog 相对于 Verilog 增加的数据类型、SystemVerilog 的编程结构、流程控制和方法（任务/函数）等基本语法。

第 3 章介绍 fork...join 结构、并行进程之间的通信方法：mailbox/semaphore/event 同步和互斥机制。

第 4 章介绍面向对象编程语言、类的基本概念、对象的创建、赋值与复制；如何采用类来封装事务处理器等验证组件和搭建验证平台。

第 5 章介绍虚接口，以及如何通过虚接口实现类的对象和设计模块的连接，实现事务处理器的可重用。

第 6 章介绍随机激励生成，其中重点讨论基于对象的约束随机激励产生机制、约束定义以及动态控制激励随机生成。

第 7 章介绍面向对象编程语言中的继承和多态。

第 8 章介绍覆盖率在验证流程中的作用、SystemVerilog 功能覆盖率的语法，包括覆盖组、覆盖点和交叉覆盖点，以及如何实现一个覆盖率驱动验证平台。

第 9 章介绍断言在验证流程中的作用、断言的采用策略、SystemVerilog 断言子集的语法结构以及如何通过 bind 结构实现断言与设计分离。

第 10 章介绍验证重用以及 OVM 验证方法学的核心技术：基于 Factory 的对象生成机制、动态参数配置、激励生成与验证架构分离以及测试用例在验证架构的顶层。

第 11 章介绍 SystemVerilog 和 C 语言的接口：DPI，重点介绍如何在 SystemVerilog 层面定义输入方法接口（SystemVerilog 调用外部 C 程序）和输出方法接口（SystemVerilog 程序输出供外部 C 调用），最后介绍 DPI 在验证中的作用。



从第4~9章,我们以石头、剪刀、布的仲裁器作为实例,并将各章中涉及的 SystemVerilog 重要语法和验证技术应用到验证平台搭建中,附有大量源代码供读者参考和练习。

## 如何阅读本书

本书的目标读者是 FPGA/ASIC 设计工程师和验证工程师、相关专业的在校本科生、研究生和老师。若具有一定的硬件描述语言 (Verilog 或者 VHDL) 和面向对象编程语言 (如 C++) 的基础,将有助于对本书的阅读。

想了解验证技术概况和验证方法学的读者,可以直接阅读第1章和第10章;对于普通读者,建议从第1章阅读到第6章;再根据自己的学习和工作需要,学习第7章以后的内容,这部分内容适合验证工程师和有一定验证经验的读者。有 Verilog 基础的读者,可以粗略浏览一下第2章的新增数据类型, SystemVerilog 其他语法结构与 Verilog 基本类似;没有 Verilog 基础的读者,这一章需要认真阅读;想了解覆盖率驱动验证、断言或者 DPI 的读者可以直接阅读对应内容的第8章、第9章和第11章。

学习一门新语言的第一步是学会读懂代码,本书为每个例子提供了详尽的解释,并且每个例子都可以在仿真平台上运行。第二步是能够将学到的语法应用到自己的项目中,编写自己的程序。第三步是学会调试代码,调试代码是最艰苦和最具挑战性的工作。和大多数编程语言一样, SystemVerilog 中基于面向对象编程结构的执行一般是不消耗物理时间的,是动态生成和析构的,为此在关键部位嵌入调试代码 (如 \$display) 将是最有效的方法。

EDA 联盟网站 [www.edaunion.com](http://www.edaunion.com) 是专业的技术论坛,为广大的工程师和读者提供了一个技术讨论的平台;各位读者可以在验证板块发帖讨论和交流自己学习过程中遇到的问题,分享自己的学习体会。本书的所有例子也可以从论坛中下载,若发现任何错误请告知我们。希望本书能够为对 SystemVerilog 和功能验证技术感兴趣的读者提供一个入门的指南。

## 致谢

本书在编写过程中,充分利用和吸收了业界最前沿的技术和信息,在此对书中参考引用文献和书籍的作者表示衷心的感谢。由于编写时间仓促,本人学识有限,若书中存在纰漏请各位读者给予谅解并指正,我将虚心听取并及时更正,并欢迎进行技术切磋和探讨 (联系方式: [edaunion.book@gmail.com](mailto:edaunion.book@gmail.com))。

本书的完成有赖于朋友们的支持,其中赵立、萧路和王国平三位资深工程师审阅了全稿,并给出了很多有建设性的意见。对他们辛勤的工作,致以最高的敬意和最真挚的感谢。

在此,我还要特别感谢重庆邮电大学的邓亚平教授、吴慧莲教授、鲜继清教授和张宗琪教授对我在学业上的帮助和支持;感谢郑建宏教授提供了 TD-SCDMA 终端芯片项目的研发平台和锻炼机会;感谢入行以来帮助过我的朋友俞洋、钟信潮、王诚、薛小刚、庄永军和傅骏诚;感谢华为海思的宾兵,让我有幸加入了一个伟大的集体,参与了 GPON 项目;感谢明导电子乐于分享经验和技术的同事。

最后,感谢我敬爱的父母!

钟文枫

2010年7月15日于上海



一本打开的书，  
一扇开启的门，  
通向科学圣殿的阶梯，  
托起一流人才的基石。



逻辑与计算机设计基础 (英文版·第3版)

作者: (英)马诺, (美)凯姆著  
书号: 7-111-23517-0 定价: 45.00元



数字逻辑基础与Verilog设计, 原书第2版

作者: [加] Stephen Brown 等译者: 夏宇闻等  
书号: 7-111-22182-1 定价: 65.00元  
■ Stephen Brown 曾四次获得电机工程、计算机工程  
和计算机科学课程的最佳教学奖  
■ 内容权威, 该领域经典教材



数字设计原理与实践, 原书第4版

作者: [美] John F. Wakerly 译者: 林生等  
书号: 7-111-20666-8 定价: 75.00元  
■ 经过多年教学锤炼的经典教科书  
■ 作者严谨学术风范与丰富经验的完美结合, 具有  
独到的“功底”



专业成就人生  
立体服务大众

www.hzbook.com

填写读者调查表 加入华章书友会  
获赠精彩技术书 参与活动和抽奖

尊敬的读者：

感谢您选择华章图书。为了聆听您的意见，以便我们能够为您提供更优秀的图书产品，敬请您抽出宝贵的时间填写本表，并按底部的地址邮寄给我们（您也可通过www.hzbook.com填写本表）。您将加入我们的“华章书友会”，及时获得新书资讯，免费参加书友会活动。我们将定期选出若干名热心读者，免费赠送我们出版的图书。请一定填写书名书号并留全您的联系信息，以便我们联络您，谢谢！

书名：

书号：7-111-( )

姓名：	性别： <input type="checkbox"/> 男 <input type="checkbox"/> 女	年龄：	职业：
通信地址：		E-mail：	
电话：	手机：	邮编：	

1. 您是如何获知本书的：

朋友推荐  书店  图书目录  杂志、报纸、网络等  其他

2. 您从哪里购买本书：

新华书店  计算机专业书店  网上书店  其他

3. 您对本书的评价是：

技术内容	<input type="checkbox"/> 很好	<input type="checkbox"/> 一般	<input type="checkbox"/> 较差	<input type="checkbox"/> 理由_____
文字质量	<input type="checkbox"/> 很好	<input type="checkbox"/> 一般	<input type="checkbox"/> 较差	<input type="checkbox"/> 理由_____
版式封面	<input type="checkbox"/> 很好	<input type="checkbox"/> 一般	<input type="checkbox"/> 较差	<input type="checkbox"/> 理由_____
印装质量	<input type="checkbox"/> 很好	<input type="checkbox"/> 一般	<input type="checkbox"/> 较差	<input type="checkbox"/> 理由_____
图书定价	<input type="checkbox"/> 太高	<input type="checkbox"/> 合适	<input type="checkbox"/> 较低	<input type="checkbox"/> 理由_____

4. 您希望我们的图书在哪些方面进行改进？

---



---

5. 您最希望我们出版哪方面的图书？如果有英文版请写出书名。

---



---

6. 您有没有写作或翻译技术图书的想法？

是，我的计划是\_\_\_\_\_  否

7. 您希望获取图书信息的形式：

邮件  信函  短信  其他\_\_\_\_\_

请寄：北京市西城区百万庄南街1号 机械工业出版社 华章公司 计算机图书策划部收  
邮编：100037 电话：(010) 88379512 传真：(010) 68311602 E-mail: hzjsj@hzbook.com



# SystemVerilog | 目 录

## 前言

<b>第 1 章 功能验证技术与方法学概要</b> .....	1
1.1 功能验证与验证平台 .....	1
1.1.1 专用芯片设计流程 .....	1
1.1.2 什么是验证 .....	2
1.1.3 验证平台可以做些什么 .....	3
1.1.4 功能验证流程 .....	5
1.2 验证技术和验证方法学 .....	8
1.2.1 黑盒、白盒与灰盒验证 .....	8
1.2.2 验证技术 .....	9
1.2.3 验证存在的挑战 .....	13
1.2.4 验证方法学 .....	13
1.2.5 断言验证 .....	15
1.2.6 覆盖率驱动验证 .....	16
1.3 硬件验证语言 .....	21
1.3.1 Open Vera .....	21
1.3.2 e 语言 .....	21
1.3.3 PSL .....	22
1.3.4 SystemC .....	22
1.3.5 SystemVerilog .....	22
<b>第 2 章 数据类型与编程结构</b> .....	24
2.1 数据类型 .....	24
2.1.1 两态数据类型 .....	25
2.1.2 枚举类型和用户自定义类型 .....	26
2.1.3 数组与队列 .....	28
2.1.4 字符串 .....	36
2.1.5 结构体和联合体 .....	37
2.1.6 常量 .....	39

2.1.7	文本表示 .....	41
2.1.8	操作符和表达式 .....	43
2.2	过程语句 .....	45
2.2.1	赋值语句 .....	45
2.2.2	控制结构 .....	46
2.3	函数和任务 .....	52
2.3.1	函数和任务的区别 .....	52
2.3.2	子程序定义 .....	53
2.3.3	子程序参数 .....	53
2.3.4	子程序返回 .....	56
2.3.5	自动存储 .....	56
2.4	编程结构 .....	57
2.4.1	模块 .....	57
2.4.2	接口 .....	59
2.4.3	过程块和语句块 .....	60
2.4.4	数据对象 .....	62
2.4.5	程序块 .....	62
2.4.6	简单的验证架构 .....	63
2.5	数据的生命周期和作用域 .....	64
2.6	数据类型转换 .....	65
2.6.1	静态类型转换 .....	66
2.6.2	动态类型转换 .....	66
<b>第3章</b>	<b>并发进程与进程同步 .....</b>	<b>68</b>
3.1	fork...join .....	68
3.1.1	三种并发方式 .....	69
3.1.2	进程与变量 .....	72
3.1.3	进程控制 .....	73
3.2	mailbox .....	74
3.2.1	mailbox 的基本操作 .....	75
3.2.2	参数化 mailbox .....	77
3.2.3	mailbox 应用实例 .....	77
3.3	semaphore .....	78
3.3.1	semaphore 的基本操作 .....	79
3.3.2	semaphore 应用实例 .....	80

3.4 event .....	81
3.4.1 事件触发 .....	81
3.4.2 等待事件 .....	81
3.4.3 事件的触发属性 .....	81
<b>第4章 面向对象编程入门 .....</b>	<b>83</b>
4.1 过程编程语言与面向对象编程语言 .....	83
4.2 类 .....	84
4.2.1 类的基本概念 .....	85
4.2.2 构造函数 .....	87
4.2.3 静态属性与方法 .....	89
4.2.4 this 操作符 .....	91
4.2.5 对象的赋值与复制 .....	91
4.2.6 块外声明 .....	94
4.3 石头、剪刀、布仲裁器实例（基于类的验证平台） .....	95
4.3.1 验证环境顶层 .....	96
4.3.2 验证组件 .....	99
<b>第5章 虚接口 .....</b>	<b>107</b>
5.1 虚接口的基本概念及应用 .....	107
5.1.1 虚接口的基本概念 .....	107
5.1.2 虚接口的应用 .....	109
5.2 端口模式和时钟控制块 .....	113
5.2.1 端口模式 .....	113
5.2.2 时钟控制块 .....	114
<b>第6章 随机测试 .....</b>	<b>118</b>
6.1 激励产生 .....	118
6.1.1 什么是随机 .....	119
6.1.2 潜在问题 .....	119
6.2 随机生成机制 .....	120
6.2.1 随机系统函数 .....	120
6.2.2 randcase/randsequence .....	121
6.3 基于对象的随机生成 .....	122
6.3.1 随机变量 .....	123
6.3.2 约束定义 .....	124



18	6.3.3 随机方法	130
18	6.3.4 随机使能控制	131
18	6.3.5 约束的动态修改	134
18	6.4 标准随机函数	134
88	6.5 随机激励的应用	135
<b>第7章 继承与多态</b> 137		
18	7.1 继承和多态的基本概念	137
28	7.2 继承与子类	137
58	7.2.1 类的继承与重写	138
88	7.2.2 子类对象与父类对象的赋值	141
10	7.2.3 构造函数调用	142
10	7.3 虚方法与多态	144
40	7.3.1 虚方法	145
20	7.3.2 多态	148
80	7.4 虚类和参数化类	148
101	7.4.1 虚类	148
101	7.4.2 参数化类	149
101	7.5 约束重写	150
109	7.6 数据的隐藏与封装	151
<b>第8章 功能覆盖率</b> 153		
111	8.1 覆盖率	153
111	8.1.1 目标覆盖率	153
111	8.1.2 代码覆盖率	154
118	8.1.3 功能覆盖率	154
911	8.2 SystemVerilog 的功能覆盖率	155
911	8.2.1 覆盖组 (covergroup)	155
150	8.2.2 覆盖点 (coverpoint)	157
150	8.2.3 交叉覆盖点 (cross)	159
151	8.3 覆盖率驱动验证平台	162
<b>第9章 断言</b> 167		
151	9.1 断言的概念及作用	167

9.2	SVA .....	169
9.2.1	SVA 的语法层次结构 .....	170
9.2.2	SVA 应用实例 .....	173
9.2.3	bind .....	175
<b>第 10 章</b>	<b>验证重用与验证方法学 .....</b>	<b>178</b>
10.1	验证重用中存在的问题 .....	178
10.2	验证方法学 OVM .....	179
10.3	OVM 的四大核心技术 .....	180
10.3.1	基于 Factory 的验证平台动态构建 .....	181
10.3.2	动态的配置机制 .....	183
10.3.3	测试用例在验证架构的顶层 .....	184
10.3.4	激励产生与验证架构分离 .....	185
<b>第 11 章</b>	<b>SystemVerilog 与 C 语言的接口 .....</b>	<b>187</b>
11.1	什么是 DPI .....	187
11.2	DPI 的应用 .....	188
11.2.1	方法的导入 .....	188
11.2.2	方法的导出 .....	190
11.2.3	DPI 的数据类型映射 .....	191
11.2.4	DPI 的具体应用 .....	192
<b>附录 A</b>	<b>覆盖率内置参数和方法列表 .....</b>	<b>193</b>
<b>附录 B</b>	<b>断言重复操作符和序列操作符列表 .....</b>	<b>195</b>
<b>附录 C</b>	<b>QuestaSim 简要介绍 .....</b>	<b>198</b>
<b>附录 D</b>	<b>常用术语中英文对照 .....</b>	<b>205</b>
<b>参考文献</b>	.....	<b>207</b>
<b>后记</b>	.....	<b>208</b>

# 源代码索引 | SystemVerilog

源代码 2-1	枚举类型实例: enum_example. sv	26
源代码 2-2	用户自定义类型实例: typedef_example. sv	28
源代码 2-3	多维数组和压缩数组实例: array_example. sv	30
源代码 2-4	动态数组实例: dy_array_example. sv	31
源代码 2-5	关联数组实例: as_array_example. sv	33
源代码 2-6	队列实例: queue_example. sv	35
源代码 2-7	结构体实例: struct_example. sv	38
源代码 2-8	类型参数化实例: para_type_example. sv	40
源代码 2-9	赋值语句实例: assign_example. sv	46
源代码 2-10	if 条件选择语句实例: if_example. sv	47
源代码 2-11	case 条件选择语句实例: case_example. sv	47
源代码 2-12	for 循环语句实例: for_example. sv	48
源代码 2-13	while 循环语句实例: while_example. sv	49
源代码 2-14	do...while 循环语句实例: dowhile_example. sv	49
源代码 2-15	repeat 循环语句实例: repeat_example. sv	50
源代码 2-16	forever 循环语句实例: forever_example. sv	51
源代码 2-17	foreach 循环语句实例: foreach_example. sv	51
源代码 2-18	ref 引用端口类型实例: ref_example. sv	54
源代码 2-19	模块实例: module_example. sv	58
源代码 2-20	接口实例: interface_example. sv	59
源代码 2-21	静态变量实例: static_auto_example. sv	65
源代码 3-1	fork...join 语句实例: fork_example. sv	70
源代码 3-2	fork...join_any 语句实例: fork_any_example. sv	71
源代码 3-3	fork...join_none 语句实例: fork_none_example. sv	71
源代码 3-4	disable 语句实例: disable_example. sv	73
源代码 3-5	mailbox 应用实例: mailbox_example. sv	77
源代码 3-6	semaphore 应用实例: semaphore_example. sv	80
源代码 3-7	event 应用实例: event_example. sv	82
源代码 4-1	简单以太包的类实例: ether_packet. sv	84



源代码 4-2	类的构造函数实例: new_construct_example. sv	88
源代码 4-3	类的静态变量实例: static_var_class. sv	89
源代码 4-4	类的静态方法实例: static_method_class. sv	90
源代码 4-5	this 操作符实例: this_class_example. sv	91
源代码 4-6	类的浅复制实例: shallow_copy. sv	92
源代码 4-7	类的自定义深复制实例: deep_copy. sv	93
源代码 4-8	仲裁器设计 (石头、剪刀、布): rps_dut. sv	95
源代码 4-9	验证环境顶层 (石头、剪刀、布): top_class_based. sv	96
源代码 4-10	时钟复位模块 (石头、剪刀、布): rps_clock_reset. sv	97
源代码 4-11	验证环境库文件 (石头、剪刀、布) ——激励单元片段: rps_env_pkg. sv	98
源代码 4-12	验证环境库文件 (石头、剪刀、布) ——激励生成器片段: rps_env_pkg. sv	99
源代码 4-13	事务驱动器 (石头、剪刀、布): rps_driver. sv	100
源代码 4-14	监控器 (石头、剪刀、布): rps_monitor. sv	101
源代码 4-15	基于类的验证环境 (石头、剪刀、布): rps_env. sv	102
源代码 4-16	验证环境库文件 (石头、剪刀、布) ——记分板片段: rps_env_pkg. sv	103
源代码 5-1	虚接口例子: virtual_interface_example. sv	108
源代码 5-2	定义虚接口 (石头、剪刀、布): interface. sv	109
源代码 5-3	基于虚接口的事务驱动器 (石头、剪刀、布): rps_driver. sv	110
源代码 5-4	基于虚接口的监控器 (石头、剪刀、布): rps_monitor. sv	111
源代码 5-5	基于虚接口的验证环境 (石头、剪刀、布): rps_env. sv	111
源代码 5-6	基于虚接口的验证顶层 (石头、剪刀、布): rps_tb_top. sv	112
源代码 5-7	端口模式实例: interface_mode. sv	113
源代码 6-1	randcase 实例: randcase_example. sv	121
源代码 6-2	randsequence 实例: randsequence_example. sv	122
源代码 6-3	基于类的随机变量实例: class_random_example. sv	123
源代码 6-4	随机约束块实例: constraint_example. sv	125
源代码 6-5	inside 操作实例: inside_example. sv	125
源代码 6-6	dist 操作实例: dist_example. sv	126
源代码 6-7	foreach 操作实例: foreach_random_example. sv	128
源代码 6-8	solve...before 操作实例: solve_before_example. sv	129
源代码 6-9	随机变量使能模式实例: rand_mode_example. sv	132
源代码 6-10	随机约束使能模式实例: constraint_mode_example. sv	133
源代码 6-11	标准随机函数实例: std_randomize_example. sv	135
源代码 6-12	基于类的随机激励实例 (石头、剪刀、布): rps_env_pkg. sv	135
源代码 7-1	类的继承实例: class_extend_example. sv	139
源代码 7-2	类的重写实例 1: class_override_example. sv	139

源代码 7-3	基类与派生类实例: base_derived_example. sv	141
源代码 7-4	super 操作实例: super_example. sv	142
源代码 7-5	构造函数链实例: new_chain_example. sv	143
源代码 7-6	类的重写实例 2: base_override_example. sv	144
源代码 7-7	虚方法与多态实例: virtual_poly_example. sv	145
源代码 7-8	参数化类实例: parameterized_class. sv	149
源代码 7-9	约束块重写实例: constraint_override_example. sv	150
源代码 8-1	功能覆盖组实例: covergroup_example. sv	156
源代码 8-2	功能覆盖点实例: coverpoint_example. sv	156
源代码 8-3	功能分组柜实例: bin_example. sv	158
源代码 8-4	功能交叉覆盖点实例: cross_example. sv	159
源代码 8-5	功能覆盖模块 (石头、剪刀、布): rps_coverage. sv	162
源代码 8-6	基于功能覆盖率验证环境 (石头、剪刀、布): rps_env. sv	165
源代码 9-1	断言序列实例: sequence_example. sv	171
源代码 9-2	断言属性实例: property_example. sv	172
源代码 9-3	断言模块 (石头、剪刀、布): rps_sva. sv	173
源代码 9-4	基于断言的验证顶层 (石头、剪刀、布): rps_tb_top. sv	174
源代码 9-5	bind 操作实例: bind_example. sv	175
源代码 11-1	外部 Hello 程序 1 (基于 C): example 1/hello_c. c	190
源代码 11-2	DPI 导入方法实例: example 1/hello. v	190
源代码 11-3	外部 Hello 程序 2 (基于 C): example 2/hello_c. c	191
源代码 11-4	DPI 导出方法实例: example 2/hello. v	191

## 功能验证技术与方法学概要

摩尔定律指出，集成电路可容纳的晶体管数目约每隔 18 个月便会增加一倍，性能也将提升一倍。随着半导体制造工艺的改进，大规模 SoC (System on Chip, 片上系统) 和多核设计的出现，专用集成电路设计的复杂度以指数形式增长，这使得验证工作成为芯片设计流程中的瓶颈，数据表明，接近 70% ~ 80% 的设计时间花费在功能验证中。目前，专用集成电路已经可以达到上亿门，设计上复杂度的提高迫切需要在功能验证方面有新的技术和方法学。

本章将从芯片设计流程入手，讨论功能验证在整个流程中的位置及其所涵盖的内容；介绍业界目前流行的各种验证技术和验证方法学，以及这些技术的优点和应用场景；最后，简要介绍业界常用验证语言 Vera、E、PSL、SystemC 和 SystemVerilog 的来历和特点。

### 1.1 功能验证与验证平台

在芯片设计过程中，验证是一个覆盖面比较广的课题，其中主要包括功能验证、物理验证、时序验证等内容。本书讨论的重点是功能验证，针对芯片设计对象的行为功能进行验证，以保证设计能够按照设计规范实现其应有的功能。在本章，我们会介绍功能验证中存在的多种验证技术，其中通过验证语言、传统的硬件描述语言或者 C 等其他语言，搭建验证平台 (testbench, 或译为测试平台) 是业界最为重要的验证手段。

验证平台 (testbench) 需要提供更多的自动化机制来提高每一个测试用例 (test case) 的功能覆盖率和减少创建测试用例的时间。然而，我们也要折中考虑到验证平台的复杂度和投入在开发验证平台的时间。一方面，验证平台的开发不应该成为专用集成电路开发的“制约因素”；另一方面，一定程度提高验证平台复杂性可以减少创建多个测试用例所消耗的时间，为此这是一个很难的抉择。

#### 1.1.1 专用芯片设计流程

图 1-1 举例说明了一个 ASIC (专用集成电路) 从制定设计规范到投片的设计流程。

目前，整个流程中关键的部分在于功能验证。在图 1-1 中功能验证只作为一个独立的模块，但它是一个相当复杂的过程——包括定义测试用例，创建测试环境，运行测试用例，保证所有要求的用例被覆盖到。

验证活动在设计规范完成后就可以开始，而且持续到版图完成，在很多情况下，验证可能超出了版图完成的阶段。下一节，我们将进一步详细介绍功能验证和验证流程。



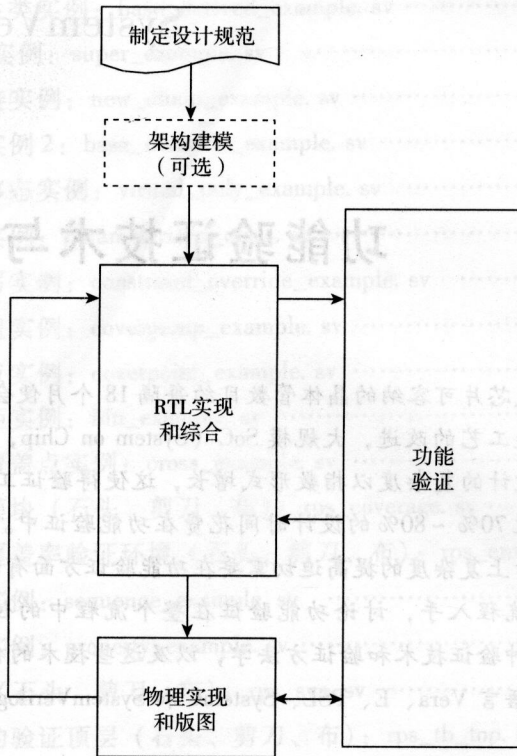


图 1-1 基本的 ASIC 设计流程

### 1.1.2 什么是验证

验证是确保设计和预定的设计期望一致（吻合）的过程。设计期望通常是通过一个或者多个设计规范来定义的。对于专用集成电路设计，在不同的阶段存在如下形式的验证。

- 1) 寄存器传输级（Register Transfer Level, RTL）的功能验证。
- 2) 门级的仿真，为了验证综合后网表和期望的功能是否一致。
- 3) 形式验证（等价性检查）来确保门级网表和 RTL 代码的一致性。
- 4) 时序验证，为了验证设计能否在特定的频率上运行，通常采用静态验证工具。

在本书下面，我们提到的验证（verification）特指寄存器传输级（RTL）的功能验证。其他形式的验证都不在讨论范围内。

功能验证在专用集成电路设计流程中关注设计的行为，几乎所有的功能都可以在 RTL 层次被验证。

图 1-2 展示了一个高层次的功能验证视图。一个专用集成电路设计可以被看成拥有一系列输入和输出的集合，设计的输出是基于设计的输入和当前的状态。在功能验证的过程中，工程师在被测设计（Design Under Test, DUT）外搭建验证平台。验证平台被用来应用一个或者多个测试激励，并将激励发送到设计的输入中。激励可以通过验证平台产生，或者通过手动创建。最后，输出将进行比较，看结果是否正确。结果检查可以通过验证平台、脚本或者手工来实现。