电子信息与电气学科规划教材

Verilog HDL

数字系统设计与验证

乔庐峰 编著　王志功 审校

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY http://www.phei.com.cn

电子信息与电气学科规划教材

Verilog HDL数字系统设计与验证

乔庐峰　编著
王志功　审校

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

本书全面介绍如何使用Verilog HDL进行数字电路设计、仿真和验证。全书共分为Verilog HDL语法基础与基本电路单元设计、系统设计与验证和附录三个组成部分。本书以Verilog-1995和Verilog-2001标准为基础，重视电路仿真与验证，紧密结合设计实践，可以帮助读者掌握规范的电路设计方法。书中大量的例题可直接用于读者的设计实践，具有良好的参考价值。

本书适合通信工程、电子工程及相关专业的高年级本科生、硕士生作为教材使用，同时也可供进行集成电路设计和可编程逻辑器件设计的工程师参考使用。

图书在版编目（CIP）数据

Verilog HDL数字系统设计与验证/乔庐峰编著. -北京：电子工业出版社，2009.4
（电子信息与电气学科规划教材）
ISBN 978-7-121-08292-4

Ⅰ.V… Ⅱ.乔… Ⅲ.硬件描述语言，Verilog HDL-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆CIP数据核字（2009）第020091号

责任编辑：马 岚
印 刷：北京市海淀区四季青印刷厂
装 订：涿州市桃园装订有限公司
出版发行：电子工业出版社
北京市海淀区万寿路173信箱 邮编：100036
开 本：787 × 1092 印张：16.75 字数：472千字
印 次：2009年4月第1次印刷
印 数：4000册 定价：29.80元

凡所购买电子工业出版社的图书有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

Verilog HDL，通常也直接称其为Verilog，是一种通用的硬件描述语言。使用Verilog HDL可以对电子电路和系统的行为进行描述。基于这种描述，结合相关的EDA软件工具，可以进行电路的设计和仿真，并最终得到所期望的实际的电路与系统。

Verilog HDL出现于20世纪80年代初，由于其使用的方便性和实用性而逐渐被众多设计者所接受，影响力不断扩大，成为工业界的设计标准。美国电气和电子工程师协会（IEEE：Institute of Electrical and Electronics Engineers）于1995年12月制定了Verilog的国际标准IEEE 1364-1995。此后，IEEE在2001年又发布了更为完善和丰富的IEEE 1364-2001标准。这两个标准的发布极大地推动了Verilog在全球的发展，使之成为进行集成电路设计和可编程逻辑器件设计的应用最广泛的语言之一。目前，使用Verilog HDL（或另一种硬件描述语言VHDL）进行电路设计是从事相关领域科研和开发工作的工程师、设计师必须具备的基本能力。

笔者长期工作在教学、科研一线，使用Verilog设计了很多具有一定规模和复杂度的电路。在学习和使用Verilog过程中，也遇到过种种难题和困惑，这些难题和困惑随着专业知识的积累和广泛的交流学习，有的得到了解决，有的理清了思路。此后，在一些教学和讲座活动中，笔者将这些经验与很多人共同分享，不少学员都觉得颇有受益并建议在现有讲义的基础上编写一本相关的教材，于是便有了这本书。Verilog HDL语法内容比较庞杂，可以在算法级、RTL级、门级和晶体管级建立电路模型和进行电路仿真。本书从电路设计与验证的实际需要出发，重点介绍的是可用于电路综合、实现的RTL级语法和与电路仿真、验证密切相关的语法要点。在进行语法学习时，本书力求以简洁清晰的方式对语法要点进行文字说明，重点是提供典型例题加以辅助分析。书中的多数例题都具有典型性和代表性，部分例题直接源于工程设计实践，具有一定的参考价值。

全书共有16章，分成了两个组成部分，其中第一部分讨论语法基础与基本电路单元设计，第二部分讨论系统设计与验证。最后给出了5个附录。

第一部分包括第1章至第10章。第1章是引言，介绍了Verilog的产生、设计流程和基本语法要素。第2章至第6章介绍了Verilog的代码结构和Verilog中的常量、变量与数据类型、Verilog中的操作符/运算符、条件语句与循环语句、任务与函数等重要语法要素。在这几章中，Verilog基本语法的介绍力求准确、简明，与语法配套的例题包括了必要的顶层电路图、设计代码、电路综合结果、验证代码和仿真结果，以使读者得到完整而全面的理解。第7章简要介绍了用户定义的原语。第8章对数字电路中的状态机进行了全面分析，总结了3种常用状态机设计风格，并给出了典型例题加以对照分析。第9章介绍了常用的系统任务与编译预处理命令。系统任务部分内容枯燥，本书中通过给出典型例题并配合适当的文字说明，使这些内容更易于理解。在前面学习基本语法知识的基础上，第10章给出了大量数字电路设计实例，目的是强化基本语法知识中的关键点。

第二部分包括第11章到第16章。第11章专门讨论了数字电路设计中的时钟问题，详细分析了目前数字逻辑电路设计中常用的静态定时分析方法的原理，并讨论了多时钟并存时的时钟域划分、同步化设计等问题。第12章系统地介绍了Verilog设计验证技术。目前很多教材重视设计而忽视对代码的验证，本书中除了为主要的例题都编写了测试代码外，还在本章通过典型例题对验证方法进行了专门讨论。第13章给出了具有一定代表性的数字电路单元的设计代码、仿真代码以及综合和

仿真结果。第14章至第16章分别给出了通信中的异步复用电路、通用异步收发器电路和Viterbi译码器电路的完整设计方案，可供初学者和工程技术人员参考。

全书以附录的形式分别介绍了可编程逻辑器件的结构、发展和片上资源（见附录A），电路仿真工具ModelSim SE的使用方法（见附录B），Xilinx ISE集成开发平台的简明使用方法（见附录C），Altera Quartus II集成开发平台的简明开发流程（见附录D）和Verilog（IEEE Std-1364-1995）关键字（见附录E）。初学者通过对附录的学习可以初步掌握常用可编程逻辑器件开发平台的使用。

作者长期从事相关领域的教学、科研工作，在本书的编写上力求简洁、清晰，并注重了例题的完整性和针对性。本书所使用的仿真工具是ModelSim，主要电路综合工具为Synplify Pro/Synplify。

王志功教授对全书进行了审阅，并提出了修改意见，在此特别表示感谢。李永成和董时华等多位硕士研究生对本书部分章节内容的原始资料收集、汇总提供了帮助，并从读者的角度对部分内容提出了宝贵意见，在此表示感谢。

由于水平有限，书中难免有错误与疏漏之处，真诚希望广大读者批评指正。

电子邮件地址：qlf2000@21cn.com。

作者

2009年1月于南京

作者简介

乔庐峰 1993年7月毕业于南京解放军通信工程学院，2005年4月在东南大学获得工学博士学位。学科方向为电路与系统，长期从事通信领域超大规模集成电路设计和FPGA设计工作，在核心学术期刊和学术会议上发表论文三十余篇，编写教材两部，获部级科技进步奖两项，现为解放军理工大学通信工程学院副教授，电信交换教研室主任。

反侵权盗版声明

目　录

第一部分　语法基础与基本电路单元设计

第二部分　系统设计与验证

第一部分　语法基础与基本电路单元设计

第1章 引　言

1.1 Verilog HDL语言的产生与发展

Verilog HDL（Verilog HDL: Verilog Hardware Description Language）是一种硬件描述语言，可以对电子电路和系统的行为进行描述。基于这种描述，结合相关的软件工具，可以得到所期望的实际的电路与系统。

Verilog HDL从20世纪80年代初由GDA（Gateway Design Automation）公司最早推出，到现在被全球范围内的众多设计者所接受，已经经历了20多年的时间。

Verilog HDL（经常又称为Verilog）最初是GDA公司为其数字逻辑仿真器产品配套开发的硬件描述语言，用于建立硬件电路的模型。那时它只是一种专用语言，但随着这种仿真器产品及其后续版本Verilog-XL的出现和广泛应用，Verilog也因为其使用的方便性和实用性而逐渐被众多设计者所接受，影响力不断扩大。

1987年，著名的电子设计自动化（EDA: Electronic Design Automation）厂商Synopsys公司开始使用Verilog语言作为其综合工具的标准输入语言。

1989年，另一个著名的EDA厂商Cadence公司收购了GDA公司，然后把Verilog HDL进行了公开发布。随后，一个名为OVI（Open Verilog HDL International）的组织成立了，专门负责Verilog的发展和标准化推动工作。到了1993年，几乎所有专用集成电路设计厂商都开始支持Verilog，并且认为Verilog-XL是最好的电路仿真软件。同时，OVI推出2.0版本的Verilog规范。美国电气和电子工程师协会（IEEE: Institute of Electrical and Electronics Engineers）接受了将OVI的Verilog HDL 2.0作为IEEE标准的提案，并于1995年12月制定了Verilog的国际标准IEEE 1364-1995。此后，IEEE在2001年又发布了更为完善和丰富的IEEE 1364-2001标准。这两个标准的发布极大地推动了Verilog在全球的发展。

Verilog语言被广泛使用的基本原因在于它是一种标准语言，与设计工具和实现工艺无关，从而可以方便地进行移植和重用。Verilog语言的两个最直接的应用领域是可编程逻辑器件和专用集成电路（ASIC: Application Specific Integrated Circuits）的设计，其中可编程逻辑器件包括复杂可编程逻辑器件（CPLD: Complex Programmable Logic Devices）和现场可编程门阵列（FPGA: Field Programmable Gate Arrays）。一段Verilog代码编写完成后，用户可以使用Altera或Xilinx等厂商生产的可编程逻辑器件来实现整个电路，或者将其提交给专业的代工厂用于ASIC的生产，这也是目前许多复杂的商用芯片（例如微控制器）所采用的实现方法。

关于Verilog语言，最后需要说明的是，它不同于常规的顺序执行的计算机程序（program），Verilog从根本上讲是并发执行的，因此我们通常称之为Verilog代码（code），而不是Verilog程序。

1.2 设计流程

使用Verilog HDL语言的主要原因之一是，通过代码综合可以采用可编程逻辑器件（CPLD或FPGA）或ASIC来实现所需的电路。图1.1给出了采用Verilog进行电路设计的流程。如图1.1所示，

设计的第一个阶段是编写Verilog代码，编写后的代码保存为一个后缀为.v的文件。代码编写完毕后进入综合阶段。综合阶段的第一步是进行代码编译，代码编译过程把寄存器传输级（RTL: Register Transfer Level）的Verilog代码转换成门级网表；综合阶段的第二步是优化，主要是根据对电路工作速度和占用硬件资源大小的要求，对门级网表进行优化。在综合之后，可以对设计进行功能仿真。功能仿真通过后，布局布线工具可以在具体的PLD/FPGA器件上对各种电路单元进行布局布线工作，或者调用专用的电路单元库生成ASIC电路。完成布局布线后，可以进行电路的后仿真，仿真结果可以最大限度地接近真实的结果。完成电路的后仿真后，就可以采用PLD/FPGA或专用集成电路实现所需的设计功能了。

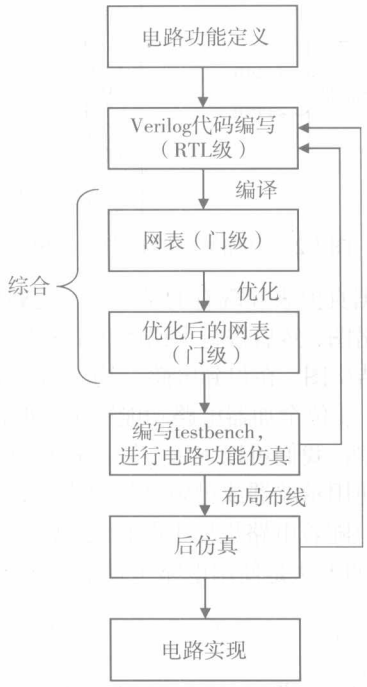


图1.1 Verilog设计流程图

目前有多种EDA工具支持采用Verilog进行电路综合、仿真以及实现。一些可编程器件生产厂商将使用Verilog进行电路设计所需的多种EDA工具集成为统一的开发平台提供给用户，进行针对本公司可编程器件产品的开发，从而使整个设计流程更加简捷和易于使用。目前比较常见的是Xilinx公司的ISE开发平台和Altera公司的Quartus II开发平台。这些平台中使用的综合工具和仿真工具通常由其他的专业EDA厂商提供，而这些EDA厂商除了为这些开发平台提供度身定制的工具外，还会推出具有标准接口的专业设计工具，供不同类型的用户配套使用。如Mentor Graphics公司的Leonardo Spectrum（综合工具），Synopsys公司的Design Compiler（综合工具），Synplicity公司的Synplify（综合工具）以及Model Technology公司的ModelSim（仿真工具）等。

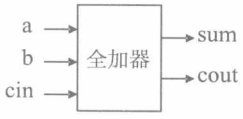
本书提供的设计实例都可以在Xilinx或Altera的CPLD/FPGA（见附录A）设计平台上实现，部分例题的综合使用了Synplify作为综合工具。

尽管在本书例题的实现和仿真中使用了不同种类的EDA工具，但我们还是尽力使电路的仿真波形在视觉效果上达到统一。对于所有例题，我们都使用ModelSim（见附录B）作为电路仿真工具，仿真波形也是从仿真结果窗口提取的，并且进行了简单处理。

1.3 Verilog HDL 在电路仿真中的应用

通过Verilog的发展历程可以看出，它最初的功能是建立电路模型和进行电路仿真。为了理解Verilog的作用，首先来看一个简单的数字逻辑电路设计过程。

我们的设计目标是实现一个图1.2所示的全加器电路，它包括两个输入信号a和b，cin是输入的进位信号，sum是输出的求和结果，cout是进位输出。图1.2中也给出了全加器的真值表。可以看出，当输入端出现奇数个1时，sum输出高电平，当输入端出现两个或两个以上高电平时，cout输出高电平。



a b cin	sum	cout
0 0 0	0	0
0 1 0	1	0
1 0 0	1	0
1 1 0	0	1
0 0 1	1	0
0 1 1	0	1
1 0 1	0	1
1 1 1	1	1

图1.2 全加器电路框图和真值表

明确设计目标之后，可以根据真值表和现有的基本数字逻辑电路单元，采用标准方法实现该逻辑功能。首先，根据真值表画卡诺图，然后进行卡诺图化简并选用逻辑单元进行组合，最终设计完成后得到的是图1.3所示的电路结构图。在得到电路结构图后，需要判断该电路的逻辑功能是否正确，设计过程中是否发生了错误。1位全加器电路功能简单，很容易判别其功能正确与否，对于功能复杂的电路该如何判断呢？当然，我们可以采用实际的逻辑电路器件将目标电路试搭出来，然后使用数字信号源加入输入信号，使用示波器或逻辑笔等仪表观察输出信号，由此来最终判断电路是否正确。这是一种可行的方法，但随着电路规模不断扩大，输入、输出信号关系复杂时，这种方法就变得难以接受了。此时，最好的方法是使用电路仿真工具在计算机上进行电路仿真。

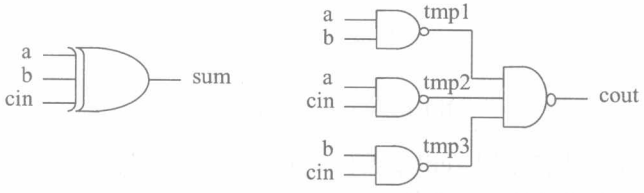


图1.3 全加器电路实现方法之一

使用计算机进行电路仿真至少包括3个环节的工作：建立用于电路仿真的模型，产生输入到被仿真电路的激励波形，输出可以观测的仿真结果。这3个基本环节中的前两个都可以使用Verilog来实现，下面给出具体的实现方法。

1.3.1 使用 Verilog 建立电路模型

下面是采用Verilog代码建立的1位全加器电路模型。

```
module full_adder(a, b, cin, sum, cout); // 定义一个名为full_adder的模块，它有5个端口信号
   input a;                              // 说明端口a为输入信号
   input b;                              // 说明端口b为输入信号
   input cin;                            // 说明端口cin为输入信号
   output sum;                           // 说明端口sum为输出信号
```

```
    output cout;                              // 说明端口 cout 为输出信号
    xor3 u1(sum, a, b, cin);                  // xor3 是用户定义的三输入异或门子电路模型
    nand2 u2(tmp1, a, b);                     // nand2 是用户定义的二输入与非门子电路模型
    nand2 u3(tmp2, a, cin);
    nand2 u4(tmp3, b, cin);
    nand3 u5(cout, tmp1, tmp2, tmp3);         // nand3 是用户定义的三输入与非门子电路模型
endmodule                                     // 模块结束
```

上面每行代码的右侧都加了注释，说明了该语句的含义，xor3、nand2 和 nand3 是采用 Verilog 定义的门电路模型。

我们可以直观地看出，这段代码所描述的逻辑功能与图 1.3 所给出的电路功能是直接对应的，这段代码就是为 1 位全加器建立的电路模型。

1.3.2 编写测试代码 testbench

为了对上述电路进行功能仿真，需要编写针对这一电路模型的测试代码，这种代码称为 testbench。下面是针对 1 位全加器编写的测试代码。

```
`timescale 1ns/1ns                            // 声明仿真时间单位为 1 ns，仿真精度为 1 ns
module full_adder_test;                       // 定义名为 full_adder_test 的测试模块
    reg a;                                    // 定义名为 a 的 reg 类型的变量
    reg b;                                    // 定义名为 b 的 reg 类型的变量
    reg cin;                                  // 定义名为 cin 的 reg 类型的变量
    wire sum;                                 // 定义名为 sum 的 wire 类型的变量
    wire cout;                                // 定义名为 cout 的 wire 类型的变量
    initial                                   // Verilog 内部定义的关键字
        begin
            a=0;  b=0;  cin=0;                // 在仿真时刻 0 时输入信号的值
        #100;                                 // 经过 100 ns 延迟
            a=0;  b=0;  cin=1;                // 在仿真时刻为 100 ns 时输入信号的值
        #100;                                 // 又经过 100 ns 延迟
            a=0;  b=1;  cin=0;                // 在仿真时刻为 200 ns 时输入信号的值
        #100;
            a=0;  b=1;  cin=1;
        #100;
            a=1;  b=0;  cin=0;
        #100;
            a=1;  b=0;  cin=1;
        #100;
            a=1;  b=1;  cin=0;
        #100;
            a=1;  b=1;  cin=1;
        end
    full_adder u1(a, b, cin, sum, cout);      // 调用被仿真电路的模型
endmodule
```

这段代码的功能是产生输入波形并将其加到被仿真电路的输入端。将采用 Verilog 建立的电路模型和编写的激励文件作为仿真器的输入文件，仿真器对其进行语法检查和编译后，可以得到图 1.4 所示的输出仿真波形。可以看出该电路的功能符合我们的设计目标。这样，采用 Verilog 建立电路的模型和针对该模型编写输入激励信号，通过电路仿真软件进行仿真，就可以得到仿真结果，从而降低了复杂数字电路的设计风险。需要说明的是，最终仿真结果与电路实现后的测试情况是否

一致，取决于所建立电路模型的精确程度和输入的测试波形是否覆盖了所有可能的应用情况，而这些都依赖于设计者对电路的理解程度和经验积累。

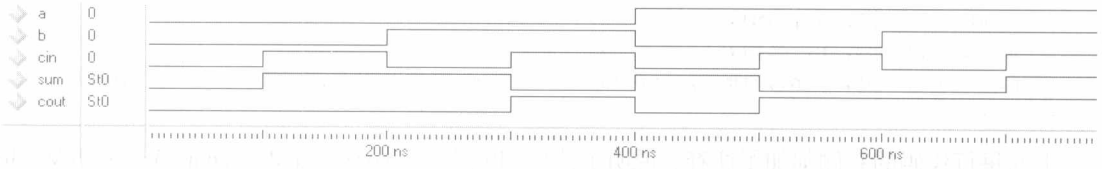


图 1.4 全加器电路模型仿真结果

Verilog 最初是作为一种建立电路模型并对电路功能进行仿真的语言而广泛使用的。通过建立电路模型和编写测试代码，设计者就可以对电路功能进行早期的验证。

1.4 Verilog HDL 在电路综合中的应用

对于初学者来说，非常关心的是如何采用 Verilog 描述电路功能，然后给予具体实现。在本书和很多 EDA 书籍中，将采用 Verilog 设计电路的过程称为建立电路模型的过程。电路综合过程就是将设计者建立的电路模型与基本逻辑电路单元库相结合，得到可以实现设计目标功能的电路的过程。

首先以全加器为例回顾一下常用的数字逻辑电路设计方法，如下所示：

1. 建立全加器的真值表
   根据需要实现的电路功能，可以建立图 1.2 所示的电路功能真值表。
2. 建立与真值表对应的卡诺图
   根据真值表，可以得到图 1.5 所示的卡诺图。
3. 进行卡诺图化简，得到积之和表达式，化简后可以得到最终的逻辑表达式。

$$sum = \bar{a}\bar{b}cin + a\bar{b}\bar{c}in + \bar{a}b\bar{c}in + abcin$$
$$= a \oplus b \oplus cin$$

$$cout = ab + bcin + acin$$

4. 根据现有的基本逻辑电路实现全加器电路。

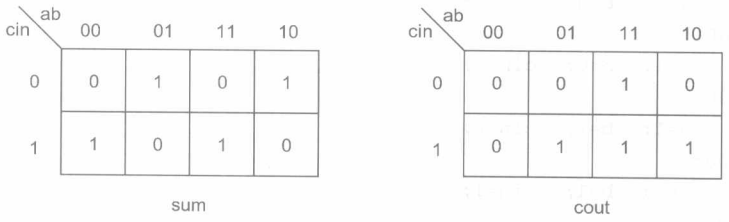


图 1.5 全加器卡诺图

在上面的设计过程中，可以发现以下问题：

1. 设计过程较为烦琐，当电路功能复杂时难以采用。
2. 布尔表达式的化简是一个至关重要的环节，它直接关系到最终电路实现时占用的硬件资源数量，从而影响到功耗、速度和可靠性等关键指标。如果以手工方式化简卡诺图或逻辑表达式，那么对于规模较大的设计来说工作量巨大。