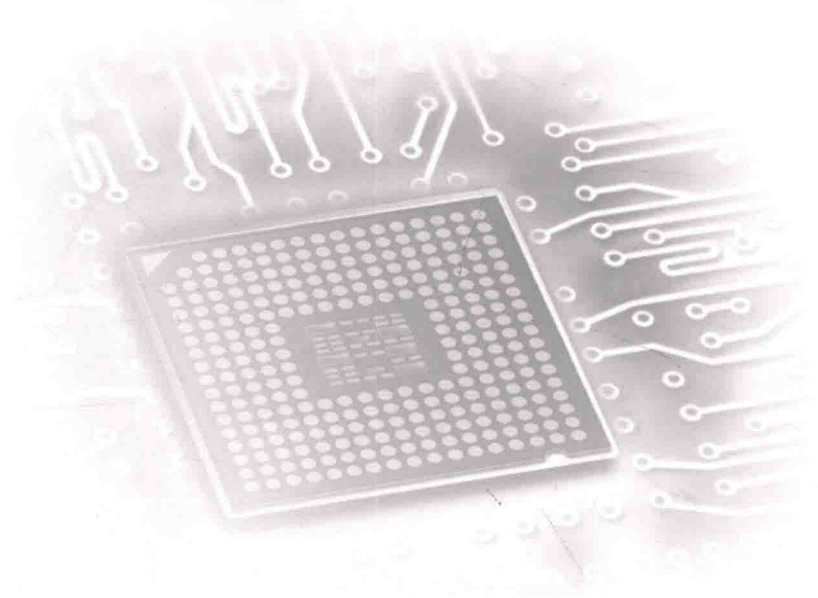




普通高校“十三五”规划教材



# Verilog硬件描述语言与设计

李洪革 李 峭 何 锋 等 编著



北京航空航天大学出版社  
BEIHANG UNIVERSITY PRESS



普通高校“十三五”规划教材

# Verilog 硬件描述语言与设计

李洪革 李峭 何锋 等编著

北京航空航天大学出版社

## 内 容 简 介

本书是电子信息工程、计算机科学与技术、自动化等电子、电气类一级学科的 EDA 教学必备基础教材,全书从硬件描述语言 Verilog HDL 简介入手,重点阐述了硬件描述语言的基础语法、高级语法和与之匹配的硬件电路设计基础、高级电路设计案例等;除了对 Verilog HDL 语法基础详细阐述外,对逻辑电路、时序综合和状态机等复杂电路设计问题也进行了介绍。本书根据国家全日制电子信息类教学大纲要求匹配了对应的实验实习,并对复杂数字系统也进行了案例讲解。全书共 11 章,主要包含 Verilog HDL 语言基础、逻辑电路结构、状态机与时序综合、验证等高级主题的内容。

本书可作为普通高等学校、科研院所电子信息工程、电气工程、计算机等相关专业的本科生或研究生的教材,还可作为上述领域工程技术人员的参考书。

### 图书在版编目(CIP)数据

Verilog 硬件描述语言与设计 / 李洪革等编著. --

北京:北京航空航天大学出版社,2017.2

ISBN 978-7-5124-2142-4

I. ①V… II. ①李… III. ①硬件描述语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2017)第 002516 号

版权所有,侵权必究。

### Verilog 硬件描述语言与设计

李洪革 李峭 何锋 等编著

责任编辑 张冀青

\*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) <http://www.buaapress.com.cn>

发行部电话:(010)82317024 传真:(010)82328026

读者信箱:goodtextbook@126.com 邮购电话:(010)82316936

北京时代华都印刷有限公司印装 各地书店经销

\*

开本:787×1092 1/16 印张:21.25 字数:544 千字

2017 年 3 月第 1 版 2017 年 3 月第 1 次印刷 印数:3 000 册

ISBN 978-7-5124-2142-4 定价:45.00 元

# 前 言

电子信息工程是现代消费电子、工业电子发展的科学基础。电子信息科学的发展依赖并推动着微电子、信息、通信和计算机科技的发展,随着集成化制造技术的迅猛发展,电子技术的核心——集成电路,已经成为影响国民经济发展的基石。2010年以来,我国集成电路进口贸易额高达2 000亿元,已经成为超过进口石油的第一大进口商品。基于此,国务院继18号文件后又推出《国家集成电路产业发展推进纲要》,以及北京市推出了《北京市进一步促进软件产业和集成电路产业发展若干政策》(京政发[2014]6号)。其间,国家中长期科学和技术发展规划纲要中公布了16项重大科技专项,前两项即是集成电路设计和制造专项课题。由此可见微电子工业在科学发展中的地位和作用。

随着我国近年来消费类电子产业的迅猛发展,该领域的科学、工程技术专业人才也显得更加短缺和重要。硬件描述语言是现代电子信息工程开发的基础工具,相关的图书先后有北京航空航天大学夏宇闻教授的《Verilog 数字系统设计教程》,王金明教授等编著的《数字系统设计与 Verilog HDL》等。由硬件描述语言衍生的FPGA应用开发等工具书更是受到读者的欢迎。除此之外,讲述某类工程数字系统中的模块代码的书籍也有一定的读者群。由于硬件描述语言与传统的高级程序语言所实现的最终目标存在本质区别,所以硬件设计人员不能仅仅考虑所描述数字系统的逻辑功能,更重要的是要考虑所实现集成化系统的物理性能。高级数字电路/系统的物理性能通常包含电路面积、功耗、速度、时滞和吞吐率等多方面的指标。

本教材以最新的IEEE Verilog—2009为基础讲解Verilog HDL语言语法和设计方法,结合当前电子信息工程、微电子科学与工程、信号与通信工程等实际需求,在符合全日制教学大纲课时规定的基础上重新进行了编写。本教材满足EDA基础类教学和工程技术需求。具体内容如下:

第1章讲述硬件描述语言的历史起源和工程应用发展概况等。

第2章介绍Verilog硬件描述语言基础,其中包含基本语法、系统任务与编译指令等。

第3章介绍Verilog HDL语法中的数据类型、端口声明、表达式以及IEEE 1995和IEEE 2005标准对比。

第4章介绍设计建模和用户自定义原语(UDP)规则等,其中包括逻辑建模、数据流建模、模块与层次和用户自定义原语等。

第5章介绍行为描述,主要包括行为建模、过程赋值语句、行为语句以及任务和函数。

第6章介绍Verilog HDL中测试平台和仿真,其中具体为测试平台、波形生成、数据显示与文件访问以及典型仿真验证实例。

第7章讲述基本数字电路采用Verilog HDL的设计方法和案例,以组合逻辑电路和时序逻辑电路两个部分进行介绍。

第8章介绍有限状态机的基础概念、类型和设计方法、编码风格、优化设计等。

第9章介绍时序、逻辑综合与验证。本章从时序概念、延迟种类、时序检查、延迟反标入手讨论逻辑综合的方法、优化和必要性;基于工程实例介绍验证方法。

第 10 章介绍仿真器使用方法和基础模块、复杂逻辑模块等设计案例,便于初学者上机实习。

第 11 章讨论并分析复杂逻辑系统的代码设计和仿真验证,为设计师系统、完整设计复杂案例提供支撑。

本书由李洪革构思执笔,李峭、何锋等五位教师、工程师执笔参编。编写分工如下:李洪革编写第 1、2、3、8 章,10.1、10.3 节;李峭编写第 4、5、11 章;何锋编写第 6、7 章和 10.2 节;中电研究所纪宇工程师编写 9.1 和 9.2 节;Marvell Technology Beijing Ltd. 工程师杨奇桦编写 9.3 节,多名研究生参与了书稿图片的制作和校对工作。

全书凝结了作者十余年 Verilog 数字系统设计的工作经验,并吸收、总结多位学者最新的研究成果。在该书的编写过程中,得到了多方面的支持与帮助。张有光教授、夏宇闻教授百忙中审阅并提出宝贵意见。国家集成电路人才培养基地——北京航空航天大学电子信息工程学院的领导和师生,一直对本书给予大力的支持和帮助。本书还获得北京航空航天大学校规划教材的支持。北京航空航天大学出版社对本书的出版提供了直接而热情的帮助。在此谨向所有为本书的编写、出版给予鼓励和帮助的社会各界人士表示衷心的感谢!

尽管作者对书稿进行了多次修改和推敲,但由于集成化系统设计的先进性和快速发展的特点,且作者学识有限,书中的错误和不当在所难免,恳请使用本书的师生和社会各界人士给予批评、指正。

李洪革

2016 年 9 月

honggeli@buaa.edu.cn

本书免费提供程序源代码和教学课件,读者可发邮件至 [goodtextbook@126.com](mailto:goodtextbook@126.com) 申请索取。若有其他帮助,请拨打 010-82317036 联系本书编辑。

# 目 录

第 1 章 电子系统与硬件描述语言	1
1.1 电子系统的集成化	1
1.2 硬件描述语言与设计方法	3
1.3 数字电路/系统实现	7
1.4 集成化设计发展趋势	11
1.5 数字集成应用前景	14
习 题	14
第 2 章 硬件描述语言基础	16
2.1 Verilog HDL 语言概况	17
2.1.1 模 块	17
2.1.2 模块名	18
2.1.3 模块组成	18
2.2 基本语法	19
2.2.1 标识符	20
2.2.2 关键字	20
2.2.3 操作符	21
2.2.4 数字声明	21
2.2.5 注释与空白符	22
2.3 系统任务与编译指令	23
2.3.1 字符串	23
2.3.2 系统任务	23
2.3.3 编译指令	25
习 题	26
第 3 章 语法与要素	28
3.1 数据类型	28
3.1.1 数 值	28
3.1.2 线网类型	29
3.1.3 变量声明	31
3.1.4 寄存器类型	32
3.1.5 阵 列	33
3.1.6 标量与矢量	34
3.1.7 参 数	35
3.2 端 口	36
3.2.1 端口命名	36

3.2.2 端口声明 .....	37
3.2.3 连接方式 .....	38
3.3 表达式 .....	41
3.3.1 运算表达 .....	41
3.3.2 操作符 .....	42
3.3.3 操作数 .....	48
3.4 标准主要差别 .....	52
习    题 .....	57
<b>第4章 建模与用户原语</b> .....	<b>59</b>
4.1 基础建模 .....	59
4.1.1 门级建模 .....	59
4.1.2 开关级建模 .....	61
4.2 数据流建模 .....	64
4.2.1 连续赋值语句 .....	64
4.2.2 数据流建模实例 .....	65
4.3 模块与层次 .....	66
4.3.1 模块划分 .....	66
4.3.2 带参数模块 .....	68
4.3.3 层次命名 .....	71
4.4 用户定义原语(UDP) .....	72
4.4.1 UDP 的含义 .....	72
4.4.2 组合逻辑 UDP .....	73
4.4.3 时序电路 UDP .....	73
习    题 .....	75
<b>第5章 行为描述</b> .....	<b>77</b>
5.1 行为级建模 .....	77
5.1.1 过程块 .....	78
5.1.2 语句块 .....	79
5.1.3 时序控制 .....	82
5.2 过程赋值语句 .....	84
5.2.1 阻塞赋值语句 .....	85
5.2.2 非阻塞赋值语句 .....	86
5.2.3 过程连续赋值语句 .....	89
5.3 行为语句 .....	90
5.3.1 条件语句 .....	90
5.3.2 多路分支语句 .....	91
5.3.3 循环语句 .....	93
5.3.4 其他语句 .....	96
5.3.5 生成块 .....	98

5.4 任务和函数 .....	101
5.4.1 任 务 .....	101
5.4.2 函 数 .....	103
习 题 .....	107
<b>第 6 章 测试、仿真和验证 .....</b>	<b>109</b>
6.1 测试平台 .....	109
6.1.1 测试向量 .....	110
6.1.2 测试模块 .....	111
6.2 波形生成 .....	113
6.2.1 值序列 .....	113
6.2.2 重复信号 .....	116
6.2.3 时钟的建立 .....	117
6.3 数据显示与文件访问 .....	120
6.3.1 数据显示 .....	120
6.3.2 文件访问 .....	123
6.3.3 从文本文件中读取向量 .....	127
6.3.4 向文本文件中写入向量 .....	128
6.4 典型仿真验证实例 .....	129
6.4.1 3-8 译码器 .....	129
6.4.2 序列检测器 .....	131
6.4.3 时钟分频器 .....	132
习 题 .....	133
<b>第 7 章 基础逻辑电路 .....</b>	<b>137</b>
7.1 组合电路设计 .....	137
7.1.1 门级结构设计 .....	137
7.1.2 连续赋值语句设计 .....	141
7.1.3 过程块语句设计 .....	143
7.1.4 组合电路不完全描述 .....	145
7.1.5 典型组合电路设计实例 .....	147
7.2 时序电路设计 .....	150
7.2.1 时序电路设计方法 .....	150
7.2.2 时序电路单元 .....	152
7.2.3 时序电路不同描述风格 .....	156
7.2.4 时序电路设计常见错误 .....	159
7.2.5 典型时序电路设计实例 .....	162
习 题 .....	163
<b>第 8 章 有限状态机及设计 .....</b>	<b>165</b>
8.1 状态机基础 .....	165
8.1.1 基本概念 .....	165



8.1.2	状态机分类 .....	167
8.2	状态机设计 .....	173
8.2.1	状态机描述方法 .....	173
8.2.2	状态机状态编码 .....	185
8.2.3	状态机优化设计 .....	190
8.2.4	状态机容错和设计准则 .....	192
习    题	.....	195
<b>第 9 章</b>	<b>时序、综合及验证</b> .....	<b>196</b>
9.1	时序与延迟 .....	196
9.1.1	时序概念 .....	196
9.1.2	延迟模型 .....	200
9.1.3	延迟种类 .....	200
9.1.4	路径延迟建模 .....	203
9.1.5	时序检查 .....	206
9.1.6	延迟反标 .....	207
9.2	逻辑综合 .....	209
9.2.1	概    念 .....	209
9.2.2	逻辑综合过程 .....	209
9.2.3	代码可综合设计 .....	212
9.3	验证方法 .....	214
9.3.1	验    证 .....	214
9.3.2	测试验证种类 .....	216
9.3.3	测试验证平台实例 .....	220
习    题	.....	228
<b>第 10 章</b>	<b>仿真实验</b> .....	<b>231</b>
10.1	硬件描述语言仿真器 .....	231
10.1.1	ModelSim 仿真 .....	231
10.1.2	逻辑综合后仿真 .....	237
10.2	Verilog 基础模块设计 .....	241
10.2.1	组合逻辑建模 .....	241
10.2.2	时序逻辑建模 .....	247
10.3	复杂逻辑设计 .....	256
10.3.1	阻塞赋值和非阻塞赋值 .....	256
10.3.2	任务与函数 .....	258
10.3.3	有限状态机 .....	262
习    题	.....	266
<b>第 11 章</b>	<b>设计案例</b> .....	<b>270</b>
11.1	异步 FIFO 设计 .....	270
11.1.1	实验目的与实验要求 .....	270

---

11.1.2 基于最高两位判决的异步 FIFO 设计	271
11.1.3 基于四象限判决的异步 FIFO 设计	276
11.2 全双工 UART 接口设计	283
11.2.1 实验目的与实验要求	283
11.2.2 UART 通信协议	283
11.2.3 UART 发送器的实现	286
11.2.4 UART 接收器的设计	289
11.2.5 UART 接收器的仿真测试	300
11.3 循环码编译码器设计	305
11.3.1 实验目的与实验要求	305
11.3.2 (7, 3)循环码	305
11.3.3 (7, 3)循环码的编码器	307
11.3.4 (7, 3)循环码的译码器	311
习 题	320
附 录	324
参考文献	327

# 第 1 章

## 电子系统与硬件描述语言

随着电子工业中微电子制造技术的微纳化,电子产品的高度集成化、低功耗化已经成为主流,硅基微电子元器件成为电子系统性能的决定因素。早在 20 世纪 80 年代初,数字集成电路设计工程技术人员为了应对日益复杂化的集成电路设计而开发了基于高级程序语言的形式化自动设计方法,从而颠覆了 60—70 年代广泛使用的人工逻辑综合的设计法。由此诞生了用于描述电子电路(系统)的高级语言,称其为硬件描述语言。相关的硬件设计——EDA 工具的使用,也成为业界发展的必然,其中,电路设计所必需的硬件描述语言成为人们关注的重点。

### 1.1 电子系统的集成化

1946 年 2 月 14 日,世界上第一台计算机 ENIAC 在美国宾夕法尼亚大学诞生。这部机器使用了 18800 个真空管,机器长 50 英尺(1 英尺=0.3048 米),宽 30 英尺,占地 1500 平方英尺,重达 30 吨。它的计算速度可实现每秒 5000 次的加法运算。该机器标志着“电子”计算机的真正到来!1947 年 12 月 16 日,贝尔实验室的 William Shockley、John Bardeen、Walter Brattain 成功地制造出第一个点接触式晶体管,由此开启了电子系统晶体管器件的时代。1958 年 9 月 12 日,德州仪器公司的 Jack Kilby 试验成功了第一块硅基晶体管的集成电路,标志着电子系统的集成化的开启。1965 年戈登·摩尔(Gordon Moore)在 *Electronics Magazine* 杂志中预测:未来,一个芯片上的晶体管数量大约每年翻一倍(10 年后修正为每 18 个月),即所谓的“摩尔定律”。

以 Intel 公司为例,1968 年 7 月,罗伯特·诺伊斯(Robert Noyce)和戈登·摩尔(Gordon Moore)从仙童(Fairchild)半导体公司辞职,创立了一个新的企业,即英特尔(Intel)公司,英文名 Intel 为 integrated electronics“(集成电子设备)”的缩写。电子系统在几十年中,已经从初期的晶体管分立器件发展到功能集成化再到系统集成。民用消费类电子产品、工业汽车电子产品,甚至空天电子系统的进步真实地再现了电子技术发展的过程。在 20 世纪 50—60 年代,电子系统都是以分立器件为核心而组建的,如消费类电子产品,就是处于器件离散、结构独立以及功能分立的状态。此时各系统的器件分立简单,板级结构复杂,导致故障率高,体积庞大,性能有限,难以实现高速大量信息数据处理和交互,系统的维护和升级也受到了严格的限制。到 70 年代,以 Intel 公司为代表的集成电路在电子系统中已经占有一席之地。随着电子系统的功能复杂化,1978 年 Intel 公司标志性地把 8088 微处理器销售给 IBM 个人计算机事业部,武装了 IBM 新产品 IBM PC 的中枢大脑。16 位的 8088 微处理器含有 2.9 万个晶体管,运行频率为 5 MHz、8 MHz 和 10 MHz。8088 微处理器成功推动 Intel 进入了“财富 500 强”企业排名。进入 80 年代后,Intel 公司发布了 286、386、484 等多种微处理器并成功应用到个人计算机(PC)。286 处理器集成了 13.4 万个晶体管,实现了第一款 16 位、运行频率可达 12.5 MHz 的处理器。386 处理器首次在 x86 架构下实现了 32 位系统,集成了 27.5 万个晶体

管,运行频率可达 40 MHz。90 年代前后,微电子技术已经发展到了超大规模集成的阶段,高集成度 ASIC 芯片的出现,大大提高了信息处理的能力,而且减小了系统的质量,降低了能耗,提高了可靠性。以 Intel 奔腾处理器为代表的集成器件,包含了 300 万个晶体管,采用 Intel 0.8  $\mu\text{m}$  的工艺技术。进入 2000 年以后,Intel 公司的 Pentium 4 采用 90 nm 的制造工艺,采用了 31 级流水线设计,配备了 16 KB 的一级缓存和多达 1 MB 的二级缓存,带有超线程技术的 Pentium 4 是 Intel 的一个卖点。Pentium 4 处理器实现了最高达 3.4 GHz 的工作频率。它代表着单核处理器的最高水平。今天,Intel 公司的产品如酷睿 i7-6700K-4 采用第二代 FinFET 14 nm 制造工艺,晶体管数量达到 2.28 亿个,核心超线程 4.0 GHz 主频,8 MB 三级缓存,支持双通道 DDR3/DDR4 内存(1600 MHz 或 2133 MHz),功耗 95 W。

现代电子信息产业始于硅谷,其硅技术先驱者包括诺依斯(N. Noyce)、摩尔(R. Moore)、布兰克(J. Blank)、克莱尔(E. Kliner)等,他们在离开肖克利实验室后成立了仙童公司,“仙童”作为第一批硅谷的半导体厂商,为整个芯片及 IT 产业培养了大量人才,全美有超过 200 家高科技公司都与仙童公司有或多或少的关系。乔布斯曾经说过:“仙童半导体就像是成熟的蒲公英,一遇东风,这种创新精神的种子就随风四处飘扬了。”与仙童公司有关系的著名企业包括 Intel、AMD、LSI、National Semi.、Xilinx、ATMEL 等。其他和仙童公司相关的电子信息类公司如图 1.1 所示。以 Intel、AMD 为代表的半导体公司巨头引领着硅技术工程产业化发展。然而,在硅技术集成度以“摩尔定律”的规律飞速发展的同时,自动化电子信息集成化设计方法逐渐成为产业发展的必然。

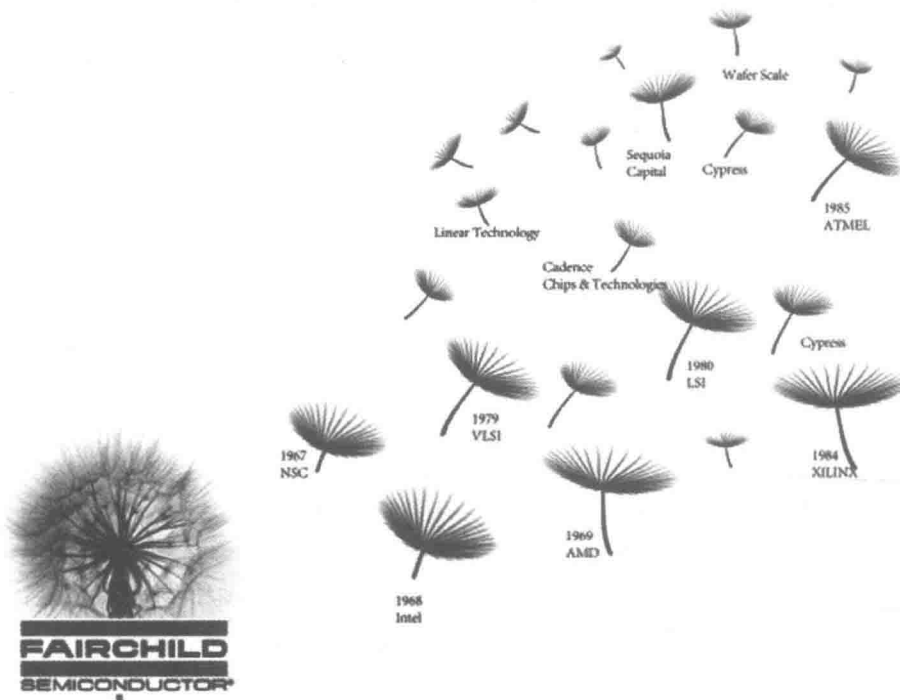


图 1.1 其他和仙童公司相关的电子信息类公司

Carver Mead 等人于 1980 年发表的《超大规模集成电路系统导论》(*Introduction to VLSI Systems*)标志着电子设计自动化发展时代的到来。这一篇具有重大意义的论文提出了通过高

级编程语言进行芯片设计的新思想。这种自动化设计方法在进行集成电路逻辑仿真、功能验证和布局布线等方面极大地减轻了设计师的劳动强度,从而为高复杂度芯片设计提供了可能。时至今日,以硬件描述语言为代表的自动化设计方法已经成为电子信息产业发展的基础。

## 1.2 硬件描述语言与设计方法

20世纪60—70年代,尽管集成电路制造取得了飞速发展,然而,当时的集成电路设计工程师只能采用代工厂提供的专用电路图来进行手工设计。对于相对复杂的数字逻辑电路,设计师从原理设计、功能设计、电路设计到版图设计,一般需要一年以上的设计周期,其中仅仅版图布局布线环节,工程师就要花费数周的时间才能完成。随着大规模集成电路的研发,80年代初系统集成可达数十万逻辑门,而其功能的仿真也很难通过传统的面板测试法验证设计的系统,在此基础上,后端工程师开始寻找通过电子设计自动化(EDA)的方法将手工设计转变为计算机辅助。前端的工程师也希望使用一种标准的语言来进行硬件设计,以提高设计的复杂度和可靠性,基于此,硬件描述语言(Hardware Description Languages, HDL)应运而生。美国国防部制定了一套电子电路规范标准文档VHSIC(Very High Speed Integrated Circuit),对上述VHSIC改良的VHDL语言在1982年正式诞生。1983年,Gateway设计自动化公司的菲尔·莫比(Phil Moorby)牵头研发了Verilog硬件描述语言。1990年,GDA公司被Cadence公司收购。1990年初,开放Verilog国际(Open Verilog International, OVI)组织(即现在的Accellera)成立。1992年,该组织申请将Verilog纳入国际电气和电子工程师协会IEEE标准。最终,Verilog成为了国际电气和电子工程师协会IEEE 1364—1995标准,即通常所说的Verilog—1995。Verilog HDL语言更接近于高级语言C,设计人员更容易理解和掌握。VHDL语言描述较复杂,其设计风格类似于PASCAL,其特点对系统设计则更有优势。Verilog HDL的IEEE 1364—2001标准(也称为Verilog—2001标准)与IEEE 1364—1995标准相比有显著的提高。2005年,用于描述系统级设计的SystemVerilog获批成为电气和电子工程师协会IEEE 1800—2005标准。为了提升Verilog的设计能力,2009年Verilog融合了SystemVerilog,成为了新的电气和电子工程师协会IEEE 1800—2009标准的Verilog硬件描述语言。因此,在数字集成电路设计(特别是超大规模集成电路的计算机辅助设计)的电子设计自动化领域中,Verilog HDL是一种用于描述、设计、仿真、验证数字电子系统的硬件描述语言。

20世纪80年代中期,工程师已经开始普遍采用HDL进行数字电路的逻辑验证,但仍延续手工方法将逻辑功能设计转化为相互连接的逻辑门表示的电路图,而手工设计大大延长了产品的研发周期。80年代后期,Synopsys公司开发了Design Compiler(简称DC)的逻辑自动综合工具,综合工具的诞生对数字电路的设计方法产生了巨大的影响。工程师可以使用HDL在寄存器传输级(Register Transfer Level, RTL)对电路进行功能描述。通过DC综合工具,设计师只需说明数据在寄存器移动和处理的过程,以及构成逻辑电路及其连线是由自动综合工具从RTL描述中抽取出来的即可,无需手工转化电路的门级网表。自动综合工具的诞生完全解放了设计师在逻辑门电路布局中的手工劳动,使设计师更专注于电路性能、结构的提升。

Verilog HDL 在电子信息集成化设计领域被广泛使用,其语法特点如下:

- 可实现基于底层数字逻辑门的设计,如逻辑与、或、非门等;
- 可实现基于行为描述的高层次设计,如条件选择语句、循环语句等;
- 可实现多种建模的混合描述风格;
- 可实现层次结构化设计的编码风格;
- 使用高级语言的高层次行为描述,以便抽象、简化底层的复杂逻辑门电路;
- 可以完成系统逻辑功能的仿真、验证,还可以基于物理器件参数设置延迟、时序、逻辑综合等;
- 可实现并发执行功能,能完全模拟硬件电路的工作过程;
- 用户可以使用自定义用户原语(UDP)和 MOS 器件,具有更强的仿真使用的灵活性;
- 支持电路由高层次行为描述到低层次逻辑门的逻辑综合。

设计方法学在计算机领域已经成为一门学科而被接受,因此,高级程序设计语言的设计方法已不可忽视。硬件描述语言的编写开发必须以工程化的思想为指导,运用标准的设计方法进行设计。高级语言的设计方法通常包含面向计算、面向过程和面向对象等。目前,Verilog HDL是一种面向过程的结构化程序设计方法,该方法的典型思想是:自顶向下、逐步细化。面向过程的语言结构是按电路功能划分成若干个基本模块,这些模块形成一个树形结构,各模块间关系尽可能简单,功能独立。数字电路的结构化设计由于采用了模块分化与功能分解,自顶向下分而治之的策略,因此,可将一个复杂的问题分解为若干子问题,各个子问题分别由不同的工程人员解决,从而提高了设计速度且便于电路调试,为数字系统的开发和维护铺平了道路。

在程序语言结构化设计思想的指导下,数字电路 Verilog 编程的步骤如下:

① 需求分析。需求分析是 Verilog HDL 程序设计中必不可少的环节。需求分析是指设计师理解、归纳、整理客户的性能需求,基于上述性能需求提出解决问题的策略方法,明确电路设计的总任务。

② 系统设计。这一部分可以分为两步:一是总体设计,即按照电路的设计要求,把总任务分解成为一些功能相对独立的子任务,最终达到每个子目标只专门完成某单一的逻辑功能的目的;二是模块设计,即按照各独立的子目标,给出各自算法完成代码设计。

③ 算法、模块和可综合设计的实现。算法是具体的解决步骤,该步骤实际上是对某些给定的数据按照一定的次序进行有限步的运算且能够求出问题的解。算法要做到易读、易懂,自身必须具有良好的结构,而良好的结构是仅用数据流、选择和循环三种基本结构组合而成的。硬件描述语言除算法、模块等逻辑设计外,还包含所设计模块的可综合化以及综合后的时序约束是否满足。Verilog HDL 仿真阶段支持不可综合的代码设计,但却无法实现电路结构,因此,需要认真对待。

④ 测试验证。代码程序编好之后,难免会出现各种各样的错误,在认真检查代码编写的语法错误后,编写可检测电路功能的测试分支,即测试平台(testbench)。测试平台模块是与电路代码相独立的,不需要完成物理实现,只是检验电路的逻辑和时序功能。电路验证除测试分支的部分外,还有对电路设计的形式化验证、代码覆盖率以及自检测验证等。

⑤ 编写程序使用与维护的文档。内容包括程序功能介绍、使用说明、参数含义等。对于有价值的程序,写出使用和维护说明等文档资料是很有必要的。

Verilog HDL 结构化自顶向下设计的流程如图 1.2 所示。项目经理根据需求分析提出相关的设计要素,提出顶层设计模块,并对下层模块进行分解。逻辑模块设计工程师执行相应的各模块设计,提出各自的设计思想、实现算法并完成测试工作。物理层设计工程师则在上述基础上完成代码的综合、综合后仿真、布局布线以及最终的测试验证。

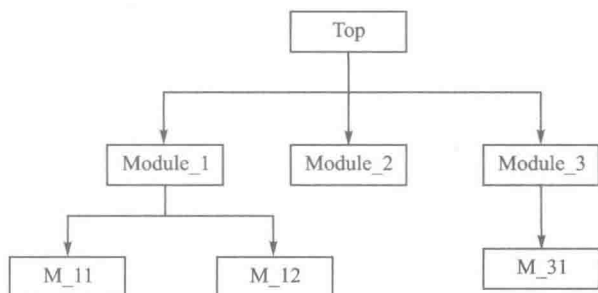


图 1.2 Verilog HDL 结构化自顶向下设计流程

基于硬件描述语言的数字电路自动化设计方法和步骤已经在前面介绍过,下面介绍 Verilog HDL 硬件电路的设计流程。图 1.3 描述了 FPGA/ASIC 数字电路设计的典型流程。在设计流程中,系统设计师首先制定所设计电路的技术指标并对功能需求进行细节描述。从系统和抽象的角度对电路功能、指标、接口及总体结构进行描述。系统分析设计阶段只考虑系统的功能而不关注具体电路结构,采用的工具一般是 C/C++、SystemC/SystemVerilog 或 MATLAB 等。当系统功能仿真满足总体设计的性能要求后,硬件设计工程师使用 HDL 语言对系统进行行为级描述,其间主要完成电路逻辑功能、物理功能的实现,并进行性能的分析 and 解决其他高层次的问题。

系统级电路的行为描述是设计中重要的一环,为提高对硬件描述语言的可理解性,一般根据其功能划分为数个功能模块和子模块并完成可综合(Synthesizable)的语法描述。这种按功能需求层层分割电路单元的方法就是所谓的层次化设计(Hierarchical Design)。对于系统的行为级描述和综合化设计,设计师依赖于 EDA 工具厂商所提供的各种工具软件。在逻辑功能的仿真阶段,FPGA 设计一般使用 Mentor Graphics 公司的 ModelSim 或者是 FPGA 开发平台自带的功能仿真平台。而 ASIC 设计工程师一般更喜欢 NC-Verilog/Verilog-XL。对于逻辑功能的仿真,仿真器并不考虑实际逻辑门或连线所产生的时间延迟、门延迟、传输延迟等信息,而是使用单位延迟的数学模型来粗略估算电路的逻辑行为。尽管逻辑功能仿真不能得到精确物理时序等结果,但已经基本满足电路逻辑功能设计的正确性验证。为实现对电路模块的功能验证,基于 HDL 语言的测试平台是必要的。其中,必须考虑所有可能影响设计功能的输入信号的组合,以便发现错误的逻辑功能描述。上述仿真验证过程中,错误修改与实际的设计经验有重要的关系,初学者往往要通过大量的实验验证总结经验。

对于 FPGA 设计,当完成电路功能验证后就可以使用相关的 FPGA 设计软件平台进行芯片设计。其平台主要有 Xilinx 公司的 ISE 开发平台和 Altera 公司的 Quartus II 平台。其设计流程主要包括功能仿真、逻辑综合、时序约束、布局布线和配置约束等几个步骤。设计可在任意开发平台下全部开发完成,无需第三方工具软件的支持,也无需集成电路物理层或器件布局布线的专业知识。

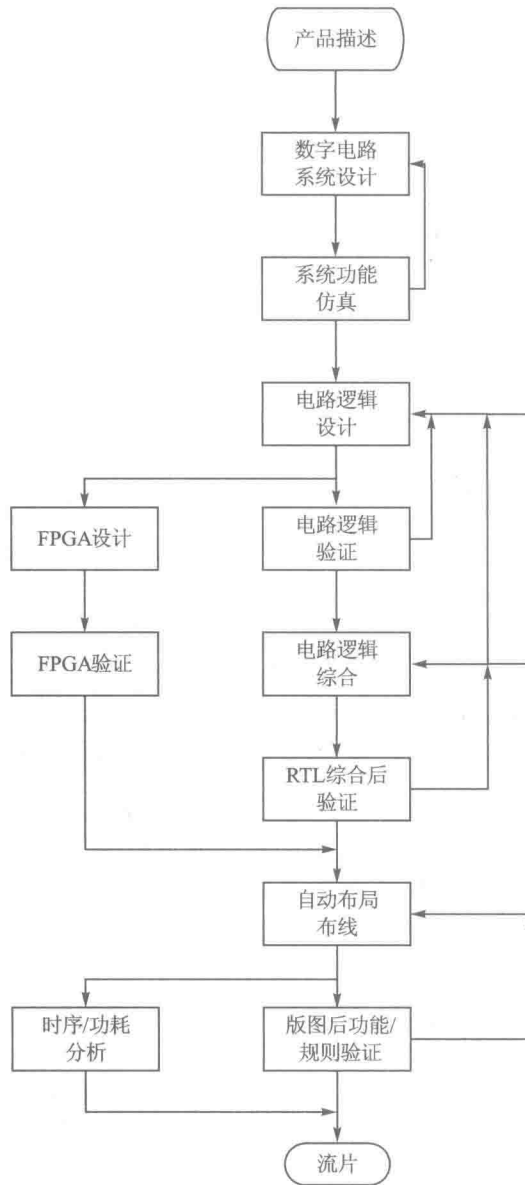


图 1.3 FPGA/ASIC 数字电路设计流程

采用 ASIC 设计方法通过电路逻辑功能验证后,后端的工作往往更复杂也更关键。设计工作的第二阶段是逻辑综合(Logic Synthesis),此阶段依靠综合工具来实现。综合过程必须选择预计流片工厂的逻辑单元库作为逻辑电路的物理单元。单元库也可以从第三方单元库供货商处获取,一般很少使用。一般而言,单元库包含的逻辑信息有以下几项:

① Cell Schematic,用于电路综合,以便产生逻辑电路的网表(Netlist)。

② Timing Model,描述各逻辑门精确时序模型,设计时提取逻辑门内寄生电阻、电容进行仿真,从而建立各逻辑门的实际延迟参数。其中包含门延迟、输入/输出延迟和连线延迟等。此数据用于综合后功能仿真以验证电路动态时序。



③ Routing Model,描述各逻辑门在进行连线时的限制,作为布线时的参考。

综合工具在完成从代码到网表的转化过程中,其中心工作就是如何获得最优化的逻辑网表。根据设定的综合约束,综合工具最终得到最为接近的结果。一般的约束条件有面积、时序和功耗,这三项约束条件是互相制约的关系,设计时应折中考虑以获得最优结果。

经过综合工具综合后得到的门级逻辑网表还要再进行第二次逻辑功能仿真,此仿真要附加反标(Back-Annotation)到测试平台的时间延迟的文件,以检验电路的逻辑功能和时序约束两个方面。在综合后仿真时,一般只考虑门延迟参数,而连线延迟是不考虑的(由于无法预计实际连线的长度及使用的金属层)。时序变异是综合后经常出现的错误,其中包含建立时间和保持时间的问题,还有电磁干扰、脉冲干扰等现象。

布局布线主要完成三项工作:版图规划、布局和布线。此部分工作也必须使用代工厂的物理库的配合才可以进行,同时,代工厂的标准单元物理库必须与综合阶段的逻辑库相一致才可以。由于各模块之间互连线较长,从而产生较大的连线延迟,而模块内的逻辑门间连线较短,因此连线延迟也较小。在深亚微米甚至纳米工艺中,其连线延迟将占主导地位。布局后的功能仿真是 ASIC 设计中最重要的一环,经过布局布线后的电路,除重复验证是否仍符合原始逻辑功能设计外,还要考虑物理实现时门延迟和连线延迟等影响下电路功能是否正常。与逻辑门级的功能验证基本相同,当发现错误时,需要修改上一级数据甚至原始的硬件描述语言代码。经过布局布线工具所产生的标准延迟格式(SDF)文件,提供了详实的物理层次延迟参数,通过反标后,仿真器能精确估算数字电路的电气行为,并可表明发生时序错误的时间点。经过反标后的仿真验证可以发现逻辑功能和时序约束的问题,对后仿时出现的问题需要修改综合约束条件甚至原始代码。

对于 ASIC 设计工程师而言,前端设计要求对 HDL 有良好的理解和对设计工作的全面把握;后端设计则要求对所使用物理单元库的物理特性充分理解,对工具充分掌握以及对流程严格操作。布局布线后仿尽管通过,但基于代工厂的设计规则验证和电气特性验证是流片前必需的步骤。版图验证主要包含设计规则检查(DRC)及版图与网表对比的检查(LVS)。在设计中,既可以采用 Cadence 公司的 Assure 工具软件,也可以使用 Mentor 的 Calibre 进行验证,深亚微米工艺一般用于后者。此时的规则检测一般来说不会有太多错误,少量手工修复即可。如果有大量错误的话,则需要返工重做自动布局布线 APR。LVS 主要验证网表与版图的一致性,是否存在短路、断路等错误。在做 LVS 前,需要把布局布线后的网表文件转换成 Spice 网表文件,使用 Calibre 的 v2lvs 命令并配合 Spice 标准库。

以上是整个 FPGA/ASIC 设计流程的简单描述,而在实际设计中会涉及到许多未提及的问题,其中包括电路性能优化、时序分析、功耗分析、可测试性设计、功能一致性验证及静态时序分析等,这些问题将在相关的章节中讨论。

### 1.3 数字电路/系统实现

随着大规模数字集成的到来,传统的单元集成正在被系统集成所代替,即整个系统完全集成到单一芯片上,从而提高了系统的性能。数字系统的集成化实现方法主要包含现场可编程门阵列(Field Programmable Gate Array, FPGA)和专用集成电路(Application Specific Integrated Circuits, ASIC)。