



完整设计与
验证工程源代码

设计与验证 Verilog HDL

EDA先锋工作室
吴继华 王诚 编著

设计与验证并重，深入讲解Verilog语言精髓
理论与实践结合，快速掌握数字IC开发技术
经验与技巧集成，在学习和工作中少走弯路



人民邮电出版社
POSTS & TELECOM PRESS



设计与验证

Verilog HDL

■ EDA先锋工作室

吴继华 王诚 编著

人民邮电出版社

图书在版编目 (CIP) 数据

设计与验证——Verilog HDL / 吴继华, 王诚编著. —北京: 人民邮电出版社, 2006.8
ISBN 7-115-15041-9

I. 设... II. ①吴...②王... III. 硬件描述语言, VHDL—程序设计 IV. TP312
中国版本图书馆 CIP 数据核字 (2006) 第 082969 号

内 容 提 要

本书以实例讲解的方式对 HDL 语言的设计方法进行介绍。全书共分 9 章, 第 1 章至第 3 章主要介绍了 Verilog HDL 语言的基本概念、设计流程、语法及建模方式等内容; 第 4 章至第 6 章主要讨论如何合理地使用 Verilog HDL 语言描述高性能的可综合电路; 第 7 章和第 8 章重点介绍了如何编写测试激励以及 Verilog 的仿真原理; 第 9 章展望 HDL 语言的发展趋势。

本书配有一张光盘, 光盘中收录了书中示例的工程文件、设计源文件及说明文件等。另外为了配合读者进一步学习, 光盘中还提供了 Verilog 1995 和 Verilog 2001 这两个版本的 IEEE 标准文献, 读者可以从中查阅 Verilog 的语法细节。

本书围绕设计和验证两大主题展开讨论, 内容丰富, 实用性强, 可作为高等院校通信工程、电子工程、计算机、微电子和半导体等相关专业的教材, 也可作为硬件工程师和 IC 工程师的参考书。

设计与验证——Verilog HDL

-
- ◆ 编 著 EDA 先锋工作室 吴继华 王 诚
责任编辑 刘莎莎 李永涛
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京密云春雷印刷厂印刷
新华书店总店北京发行所经销
 - ◆ 开本: 787×1092 1/16
印张: 15
字数: 359 千字 2006 年 8 月第 1 版
印数: 1—5 000 册 2006 年 8 月北京第 1 次印刷

ISBN 7-115-15041-9/TP · 5576

定价: 32.00 元 (附光盘)

读者服务热线: (010) 67132692 印装质量热线: (010) 67129223



网站支持

EDA 先锋工作室是与人民邮电出版社紧密合作的一支电子设计领域专业书籍创作队伍。该工作室的成员都是电子、通信、半导体行业的资深研发人员。

为了配合本书的学习, EDA 先锋工作室在“EDA 专业论坛”(http://www.edacn.net)上开办了《设计与验证——Verilog HDL》的讨论园地。利用该园地, 作者联合业界专业人士长期在论坛上为读者答疑解惑, 讨论 EDA 工程经验与设计技巧, 并对书中所述问题加以引申, 借此与读者共同切磋, 互相提高。网站提供了本书光盘中所附资料的下载服务, 并介绍了 EDA 先锋工作室所编图书的出版动态。

EDA 先锋工作室非常重视您的批评和建议, 您可以通过电子函件以及网站反馈您的信息。

电子信箱 altera_book@edacn.net。

EDA 先锋工作室

主 编: 王 诚

副主编: 薛小刚 钟信潮

编 委: 李 楠 吴继华 庞 健 由武军 袁 园
周海涛 侯小辉 寿开宇 范丽珍 薛 宁
路 远 梁晓明 伊贵业 吴义涛 张世卓
张伟平 王书松 吴 蕾 胡安琪 吴卫旋
董振东 于春华

内容和特点

Verilog HDL 作为两大硬件描述语言之一, 拥有很大的用户群。据调查, 目前美国有 90% 左右的 IC 设计人员使用 Verilog, 在中国, 大概有 50% 左右的人在使用 Verilog。当前数字芯片设计行业正处于强劲上升时期, 风头盖过了几年前的软件设计业, 已经成为电子和 IT 类的高薪行业。大量高校毕业生和部分软件设计人员正在不断涌入这个领域。要想尽快在 IC 设计领域站稳脚跟, 就必须尽快掌握 HDL 语言的设计方法。

现在市场上关于 Verilog 的书籍大多数是介绍语法和建模的, 没有真正体现出理论性与实用性的结合。针对这种情况, 本工作室创作了本书。

全书共分 9 章, 各章内容简要介绍如下。

- 第 1 章: 介绍 HDL 的设计方法, Verilog 与 VHDL、C 等语言的区别, 以及 HDL 语言的设计与验证流程。
- 第 2 章: 介绍 Verilog 的语言基础。
- 第 3 章: 重点介绍 Verilog 的 3 种描述方法和不同的设计层次。
- 第 4 章: 介绍 RTL 建模的概念和一些常用电路的 Verilog 设计方法, 最后引申出 Verilog 语言的可综合子集。
- 第 5 章: 总结了常用的 RTL 同步设计原则, 逐一介绍了设计模块的划分、设计组合逻辑和时序逻辑时应该注意的问题, 以及优化 RTL 代码的方法等内容。
- 第 6 章: 介绍状态机的设计方法和技巧。
- 第 7 章: 介绍如何搭建测试平台, 对设计进行验证。
- 第 8 章: 详细描述了 Verilog 语言的语义和仿真原理, 是 Verilog 语言的精髓所在。
- 第 9 章: 总结并展望 HDL 和 HVL 的发展趋势。

读者对象

本书可作为高等院校通信工程、电子工程、计算机、微电子和半导体等相关专业的教材, 也可作为硬件工程师和 IC 工程师的参考书。

光盘使用说明

本书配套光盘中提供了书中示例的工程文件、设计源文件和说明文件, 示例按照章节编号和出现的先后顺序排列, 例如“Example-2-1”表示第 2 章中的第 1 个示例。

工程示例文件夹中包含该工程的项目文件、源文件、报告文件和生成结果等文件。

对于一些相对复杂的示例，说明文件中给出了示例的详细信息和操作指南，而对于一些简单的实例，则只给出了源代码。

另外，为了配合读者进一步学习，光盘中还提供了 Verilog 1995 和 Verilog 2001 这两个版本的 IEEE 标准文献，读者可以从中查阅 Verilog 的语法细节。

本书约定

为了方便读者阅读，本书还设计了 4 个小图标，这些图标的含义如下。



行家指点：用于介绍使用经验和心得，或罗列一些重要的概念。



注意事项：用于提醒读者应该注意的问题。



多学一招：用于介绍实现同一功能的不同方法。



操作实例：用于引出一个操作题目和相应的一组操作步骤。

本书第 1、4、5、6 章由王诚编写，第 2、3、7、8、9 章由吴继华编写。由于作者水平有限，书中难免会有疏漏，敬请读者批评指正。

感谢您选择了本书，也请您把对本书的意见和建议告诉我们。

EDA 先锋工作室网站 <http://www.EDACN.net>。

EDA 先锋工作室

2006 年 7 月

第 1 章 HDL 设计方法简介.....	1
1.1 设计方法的变迁.....	1
1.2 Verilog 语言的特点.....	2
1.2.1 Verilog 的由来.....	2
1.2.2 HDL 与原理图.....	2
1.2.3 Verilog 和 VHDL.....	3
1.2.4 Verilog 和 C 语言.....	4
1.3 HDL 的设计与验证流程.....	5
1.4 问题与思考.....	7
第 2 章 Verilog 语言基础.....	9
2.1 Top-Down 和 Bottom-Up.....	9
2.2 Verilog 的 3 种描述方法.....	10
2.2.1 实例.....	10
2.2.2 3 种描述方法.....	13
2.3 基本词法.....	14
2.4 模块和端口.....	15
2.5 编译指令.....	16
2.6 逻辑值与常量.....	17
2.6.1 逻辑值.....	17
2.6.2 常量.....	18
2.7 变量类型.....	19
2.7.1 线网类型.....	19
2.7.2 寄存器类型.....	19
2.7.3 变量的物理含义.....	20
2.7.4 驱动和赋值.....	20
2.8 参数.....	22
2.9 Verilog 中的并发与顺序.....	22
2.10 操作数、操作符和表达式.....	23
2.10.1 操作符.....	23
2.10.2 二进制数值.....	26
2.10.3 操作数.....	26

2.11 系统任务和系统函数	28
2.11.1 显示任务	28
2.11.2 文件输入/输出任务	28
2.11.3 其他系统任务和系统函数	29
2.12 小结	29
2.13 问题与思考	29
第 3 章 描述方式和设计层次	31
3.1 描述方式	31
3.2 数据流描述	31
3.2.1 数据流	31
3.2.2 连续赋值语句	31
3.2.3 延时	33
3.2.4 多驱动源线网	34
3.3 行为描述	36
3.3.1 行为描述的语句格式	36
3.3.2 过程赋值语句	40
3.3.3 语句组	43
3.3.4 高级编程语句	44
3.4 结构化描述	50
3.4.1 实例化模块的方法	52
3.4.2 参数化模块	53
3.5 设计层次	57
3.5.1 系统级和行为级	57
3.5.2 RTL 级	59
3.5.3 门级	60
3.5.4 晶体管级	60
3.5.5 混合描述	60
3.6 实例: CRC 计算与校验电路	60
3.6.1 CRC10 校验, 行为级	61
3.6.2 CRC10 计算电路, RTL 级	62
3.7 小结	64
3.8 问题与思考	64
第 4 章 RTL 概念与 RTL 级建模	65
4.1 RTL 与综合的概念	65
4.2 RTL 级设计的基本要素和步骤	65
4.3 常用的 RTL 级建模	67

4.3.1	阻塞赋值、非阻塞赋值和连续赋值	67
4.3.2	寄存器电路建模	68
4.3.3	组合逻辑建模	70
4.3.4	双向端口与三态信号建模	72
4.3.5	Mux 建模	73
4.3.6	存储器建模	74
4.3.7	简单的时钟分频电路	75
4.3.8	串并转换建模	77
4.3.9	同步复位和异步复位	77
4.3.10	使用 case 和 if...else 语句建模	81
4.3.11	可综合的 Verilog 语法子集	87
4.4	设计实例: CPU 读写 PLD 寄存器接口	87
4.5	小结	92
4.6	问题与思考	92

第 5 章 RTL 设计与编码指导93

5.1	一般性指导原则	93
5.1.1	面积和速度的平衡与互换原则	94
5.1.2	硬件原则	103
5.1.3	系统原则	105
5.2	同步设计原则和多时钟处理	107
5.2.1	同步设计原则	107
5.2.2	亚稳态	109
5.2.3	异步时钟域数据同步	111
5.3	代码风格	113
5.3.1	代码风格的分类	113
5.3.2	代码风格的重要性	113
5.4	结构层次设计和模块划分	114
5.4.1	结构层次化编码 (Hierarchical Coding)	114
5.4.2	模块划分的技巧 (Design Partitioning)	115
5.5	组合逻辑的注意事项	116
5.5.1	always 组合逻辑信号敏感表	116
5.5.2	组合逻辑反馈环路	117
5.5.3	脉冲产生器	118
5.5.4	慎用锁存器 (Latch)	119
5.6	时钟设计的注意事项	120
5.6.1	内部逻辑产生的时钟	120
5.6.2	Ripple Counter	121

5.6.3	时钟选择	121
5.6.4	门控时钟	121
5.6.5	时钟同步使能端	122
5.7	RTL 代码优化技巧	123
5.7.1	使用 Pipelining 技术优化时序	123
5.7.2	模块复用与资源共享	123
5.7.3	逻辑复制	125
5.7.4	香农扩展运算	127
5.8	小结	129
5.9	问题与思考	130
第 6 章	如何写好状态机	131
6.1	状态机的基本概念	131
6.1.1	状态机是一种思想方法	131
6.1.2	状态机的基本要素及分类	133
6.1.3	状态机的基本描述方式	133
6.2	如何写好状态机	134
6.2.1	评判 FSM 的标准	134
6.2.2	RTL 级状态机描述常用的语法	135
6.2.3	推荐的状态机描述方法	138
6.2.4	状态机设计的其他技巧	151
6.3	使用 Synplify Pro 分析 FSM	154
6.4	小结	157
6.5	问题与思考	157
第 7 章	逻辑验证与 Testbench 编写	159
7.1	概述	159
7.1.1	仿真和验证	159
7.1.2	什么是 Testbench	160
7.2	建立 Testbench, 仿真设计	161
7.2.1	编写仿真激励	162
7.2.2	搭建仿真环境	172
7.2.3	确认仿真结果	173
7.2.4	编写 Testbench 时需要注意的问题	175
7.3	实例: CPU 接口仿真	177
7.3.1	设计简介	177
7.3.2	一种 Testbench	178
7.3.3	另外一种 Testbench	182

7.4 结构化 Testbench	183
7.4.1 任务和函数	184
7.4.2 总线功能模型 (BFM)	184
7.4.3 测试套具 (Harness)	185
7.4.4 测试用例 (Testcase)	185
7.4.5 结构化 Testbench	186
7.5 实例: 结构化 Testbench 的编写	188
7.5.1 单顶层 Testbench	188
7.5.2 多顶层 Testbench	191
7.6 扩展 Verilog 的高层建模能力	192
7.7 小结	193
7.8 问题与思考	193

第 8 章 Verilog 语义和仿真原理195

8.1 从一个问题说起	195
8.2 电路与仿真	196
8.2.1 电路是并行的	196
8.2.2 Verilog 是并行语言	197
8.2.3 Verilog 仿真语义	197
8.3 仿真原理	198
8.3.1 Verilog 的仿真过程	198
8.3.2 仿真时间	202
8.3.3 事件驱动	203
8.3.4 进程	203
8.3.5 调度	204
8.3.6 时序控制 (Timing Control)	205
8.3.7 进程、事件和仿真时间的关系	205
8.3.8 Verilog 语言的不确定性	205
8.4 分层事件队列与仿真参考模型	206
8.4.1 分层事件队列	206
8.4.2 仿真参考模型	206
8.5 时序模型与延时	207
8.5.1 仿真模型 (Simulation Model)	207
8.5.2 时序模型 (Timing Model)	208
8.5.3 案例分析	208
8.5.4 在 Verilog 语言中增加延时	210
8.6 再谈阻塞与非阻塞赋值	213
8.6.1 本质	213

8.6.2 案例分析	216
8.7 如何提高代码的仿真效率	219
8.8 防止仿真和综合结果不一致	219
8.9 小结	220
8.10 问题与思考	220
第 9 章 设计与验证语言的发展趋势	221
9.1 设计与验证语言的发展历程	221
9.1.1 HDL 语言	221
9.1.2 C/C++和私有的验证语言	222
9.1.3 Accellera 和 IEEE 的标准化工作	222
9.2 硬件设计语言的发展现状和走向	223
9.2.1 HDL 的竞争	223
9.2.2 一些尝试	223
9.2.3 下一代的 Verilog 语言	223
9.2.4 SystemC	224
9.3 验证语言的发展现状和走向	225
9.3.1 验证方法	225
9.3.2 HVL 标准化进程	225
9.3.3 HVL 的新需求	226
9.4 总结和展望	226
9.5 小结	226
9.6 问题与思考	226
附录 Verilog 关键字列表	227

第1章 HDL 设计方法简介

本章重点介绍数字系统的建模和 HDL 语言的基本概念，并引入了主流的设计和验证流程。

本章主要内容如下：

- 设计方法的变迁；
- Verilog 语言的特点；
- HDL 的设计与验证流程。

1.1 设计方法的变迁

随着微电子设计技术的发展，数字集成电路已经从电子管、晶体管、中小规模集成电路、超大规模集成电路（VLSIC）逐步发展到今天的专用集成电路（ASIC）。人们在工作生活中用到的一些产品，如计算机、手机、数字电视等都运用了复杂的专用数字集成电路，而数字逻辑器件也从简单的逻辑门发展到了复杂的 SOC（System On Chip，片上系统），提供了对复杂系统的灵活支撑。

随着数字电路系统的不断发展，系统的逻辑复杂度与规模日益增加，数字系统的设计方法也随之不断演进。在早期简单的门逻辑设计阶段，电子辅助设计（EDA）工具的应用范围十分有限，工程师们习惯于使用卡诺图简化设计，然后通过面包板等实验系统验证设计；在系统相对复杂以后，工程师们又开始借助 EDA 工具通过原理图描述数字系统，原理图由元件库中的元件构成，使用 EDA 工具可以对原理图进行仿真并分析其性能；当数字系统发展到 ASIC 与可编程逻辑器件（PLD）设计阶段后，原理图不利于移植，维护起来费时费力等缺点逐步显现，这时一种抽象度更高、运用起来更灵活的设计方式——硬件描述语言（HDL， Hardware Description Language）应运而生。

使用 HDL 语言可以从算法、系统级（System Level）、功能模块级（Function Model Level）、行为级（Behavior Level）、寄存器传输级（RTL， Register Transfer Level）、门级（Gate Level）和开关级（Switch Level）等不同层次描述数字电路系统，然后通过 EDA 工具综合、仿真并实现该系统。可以说 HDL 语言的出现是数字系统设计方法的一个重大飞跃。

由于 EDA 工具的不断推陈出新，又引发了数字电路系统设计方法的另一个重大飞跃。当数字系统发展到 ASIC 和 PLD 设计阶段之后，人们需要直接描述 CMOS 的开关电路或门级电路。这种电路设计量庞大，仿真速度也非常慢，如果用开关级或门级方法描述当今系统门数量为千万门级的 FPGA（Field Programmable Gate Array，现场可编程门阵列），结果是不可想象的。这时人们就希望能够使用 HDL 语言直接从更高的层次描述电路，然后使用



EDA 工具自动将高层次的 HDL 电路描述解析到门级,从而大大缩短了设计与仿真的时间。这种通过 EDA 工具将高层次的电路描述解析到门级等低层次的电路描述的过程就叫做“综合”(Synthesize),或者称为逻辑综合。综合工具能将高层次的 HDL 语言、原理图等设计描述翻译成由与、或、非门等基本逻辑单元组成的门级连接(网表),并根据设计目标与要求(约束条件)优化所生成的逻辑连接,输出门级网表文件。目前最成熟的综合工具是 RTL 级综合工具,它可将 RTL 级描述翻译并优化为门级网表。综合工具的产生实现了数字电路系统设计方法的又一次伟大飞跃。

1.2 Verilog 语言的特点

本节综述 Verilog HDL 语言的特点,并将其与其他语言进行比较。

1.2.1 Verilog 的由来

Verilog 是 Verilog HDL 的简称。Verilog 语言最初于 1983 年由 Gateway Design Automation 公司开发,于 1995 年被认证为 IEEE 标准。Verilog 语言不仅定义了语法,而且还对每个语法结构都清晰定义了仿真语义,从而便于仿真调试。Verilog 语言继承了 C 语言的很多操作符和语法结构,对初学者而言易学易用。另外 Verilog 语言具有很强的扩展性,最新的 Verilog 2001 标准大大扩展了 Verilog 的应用灵活性。

另外一种流行的 HDL 语言是 VHDL (Very High Speed Integrated Circuit HDL, 超高速集成电路硬件描述语言),其发展初期得到了美国国防部的支持,并于 1987 年成为 IEEE 标准。VHDL 语言的特点是描述严谨。

为了加深读者对 Verilog 语言的理解,下面将其与其他几种电路描述方法进行对比。

1.2.2 HDL 与原理图

HDL 和原理图是两种最常用的数字硬件电路描述方法。原理图设计输入法在早期应用得比较广泛,它会根据设计要求,选用器件,绘制原理图,完成输入过程。这种方法的优点是直观,便于理解,元件库资源丰富;但是在大型设计中,这种方法的可维护性较差,不利于模块建设与重用,更主要的缺点是,当所选用的芯片升级换代后,所有的原理图都要作相应的改动。

目前进行大型工程设计时,最常用的设计方法是 HDL 设计输入法,其中影响最为广泛的 HDL 语言是 Verilog HDL 和 VHDL。它们的共同特点是利于自顶向下的设计,利于模块的划分与重用,可移植性好,通用性好,设计不因芯片工艺和结构的变化而变化,更利于向 ASIC 移植。

波形输入和状态机输入方法是两种常用的辅助设计输入方法。使用波形输入法时,只要绘制出激励波形和输出波形,EDA 软件就能自动根据响应关系进行设计或仿真。而使用状态机输入法时,设计者只需画出状态转移图,EDA 软件就能生成相应的 HDL 代码或者原理图。这两种设计方法往往与某种特定的设计工具相关,应用起来容易受到局限,而且效率和可维护性不高,仅仅在某些场合作为辅助的设计描述手段使用。



因此推荐初学者在描述和仿真数字电路时首选 HDL 语言方式，而在某些要求使用图形描述设计顶层的情况下才使用原理图，不要在设计顶层以外的其他层次使用原理图。另外不要依赖波形设计工具，因为简单的信号虽然用波形描述起来十分方便，但是复杂的测试激励几乎无法使用波形工具进行有效描述。

1.2.3 Verilog 和 VHDL

Verilog 和 VHDL 作为最流行的 HDL 语言，从设计能力上而言都能胜任数字电路系统的设计任务。

VHDL 最初被用作文档来描述数字硬件的行为，因此 VHDL 的描述性和抽象性更强，也就是说 VHDL 更适合描述更高层次（如行为级、系统级等）的硬件电路。

Verilog 最初是为更简捷、更有效地描述数字硬件电路和仿真而设计的，它的许多关键字和语法都继承了 C 语言的传统，因此易学易懂。

前面已经提到最流行的 HDL 语言是 Verilog 和 VHDL，后来在其基础上又发展出了许多抽象程度更高的硬件描述语言，如 SystemVerilog、Superlog、SystemC 和 CoWare C 等。这些高级 HDL 语言的语法结构更加丰富，更适合用于系统级、功能级等高层次的设计描述和仿真。HDL 语言适用层次示意图如图 1-1 所示，其中实线框表示适用程度较高，虚线框表示适用程度较低。

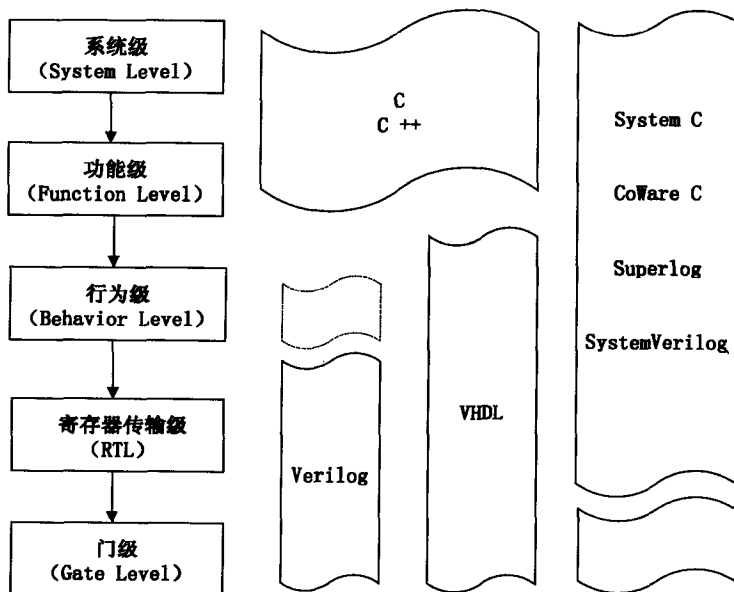


图1-1 HDL 语言适用层次示意图

本书推荐初学者先从 Verilog 学起。Verilog 语法的寄存器和线网两种数据类型定义清楚，时序与组合电路描述简洁，能够帮助初学者快速了解硬件设计的基本概念，非常容易上手，这也是 Verilog 成为最受欢迎的 HDL 语言的主要原因。

但是请读者明确，Verilog 与 VHDL 语言本身并没有什么优劣之分，而是各有所长。使用 HDL 语言描述数字硬件电路，其本质是将硬件电路抽象为语言这种表达形式，因此可以



说 HDL 是实际硬件电路与 EDA 工具之间的桥梁。选择何种语言作为桥梁本身并不重要，关键是如何有效地为真实电路建模，因此最重要的是建模的方法与思想。

1.2.4 Verilog 和 C 语言

Verilog 语言是根据 C 语言发明而来的，因此 Verilog 语言具备了 C 语言简洁易用的特点。Verilog 从 C 语言中借鉴了许多语法，例如预编译指令和一些高级编程语言结构等。

一、C 语言与 Verilog 的最大区别

C 语言与 Verilog 的最大区别在于 C 语言缺乏硬件描述的 3 个基本概念。

- 互连 (connectivity): 在硬件系统中，互连是一个非常重要的组成部分，而在 C 语言中，并没有直接可以用来表示模块间互连的变量；而 Verilog 的 wire 型变量配合一些驱动结构能有效地描述出网线的互连。
- 并发 (concurrency): C 语言天生是串行的，不能描述硬件之间的并发特性，C 语言编译后，其机器指令在 CPU 的高速缓冲队列中基本是顺序执行的；而 Verilog 可以有效地描述并行的硬件系统。
- 时间 (time): 运行 C 程序时，没有一个严格的时间概念，程序运行时间的长短主要取决于处理器本身的性能；而 Verilog 语言本身定义了绝对和相对的时间度量，在仿真时可以通过时间度量与周期关系描述信号之间的时间关系。

二、HDL 语言的本质

读者必须明确一点，就是硬件描述语言 (HDL) 同软件语言 (如 C、C++ 等) 是有本质区别的。Verilog 作为硬件描述语言，它的本质作用在于描述硬件。Verilog 虽然采用了 C 语言的形式，但是它的最终描述结果是芯片内部的实际电路。所以评判一段 HDL 代码优劣的最终标准是其描述并实现的硬件电路的性能 (包括面积和速度两个方面)。评价一个设计的代码水平较高，仅仅是说这个设计由硬件向 HDL 代码这种表现形式转换得更流畅、更合理，而一个设计的最终性能，在很大程度上取决于设计工程师所构想的硬件实现方案的合理性。

初学者，特别是从软件设计转行的初学者，片面追求代码的整洁、简短是错误的，是与评价 HDL 的标准背道而驰的。正确的编码方法是，首先要对所实现的硬件电路有一个清楚的认识，对该部分硬件的结构与连接了解得十分透彻，然后再用适当的 HDL 语句将其表达出来。

三、Verilog 与 C 的结合

Verilog 毕竟是硬件描述语言，它在抽象程度上比 C 语言要差一些，语法不如 C 灵活，在文件的输入与输出方面功能也明显不如 C。为了克服这些缺陷，Verilog 的设计者们发明了编程语言接口，也叫做 PLI。通过 PLI，可以在仿真器中实现 C 语言程序和 Verilog 程序间的互相通信，或者是在 Verilog 中调用 C 语言的函数库，从而大大扩展了 Verilog 语言的灵活性和高层次抽象的能力。开发时，一方面硬件设计者使用 Verilog 进行硬件建模，另一方面验证工程师却常常使用 C 语言来编写测试向量，然后通过 Verilog 的编程语言接口 (PLI) 将 Verilog 和 C 联系起来。

1.3 HDL 的设计与验证流程

HDL 的基本功能就是有效地描述并仿真硬件系统。这节我们抛开具体的 PLD 或 ASIC 设计流程,从 HDL 语言层次入手,分析典型的 HDL 设计与验证流程。HDL 的设计与仿真流程如图 1-2 所示,其中虚线框里的步骤可以依据项目的复杂程度而省略,实线框里的步骤为必须执行的步骤。

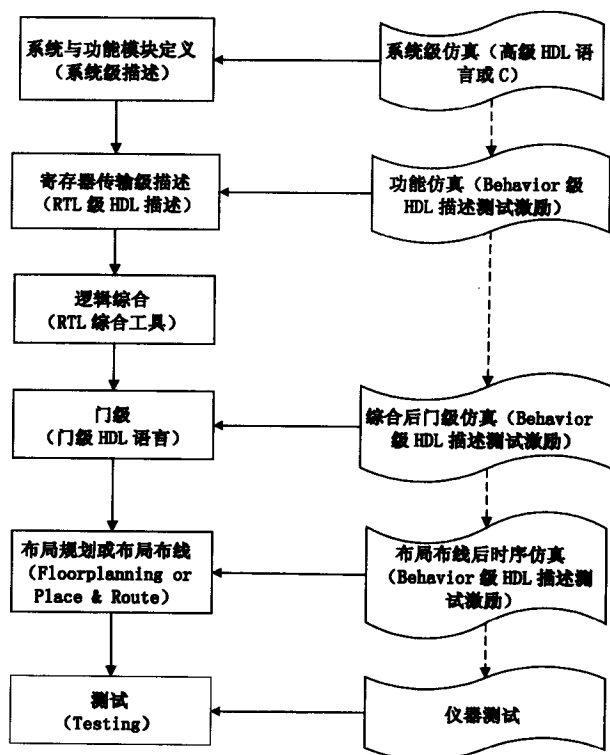


图1-2 HDL 的设计与仿真流程

下面具体讲一些关键的设计步骤与概念。

• 系统与功能模块定义 (系统与功能模块级)

在大型系统的设计与实现中,首先要进行详细的系统规划和描述。此时 HDL 描述侧重于对整体系统的规划和实现。系统级仿真的主要目标是对系统的整体功能和性能指标进行衡量。系统级设计与仿真多采用高级描述语言,如 C/C++、System C 和 System Verilog 等。系统级描述完成后,应该进一步将系统功能划分为可实现的具体功能模块,大致确定模块间的接口,如时钟、读写信号、数据流和控制信号等,并根据系统要求描述出每个模块或进程的时序约束,这个细化的过程被称为功能模块级设计。功能模块级仿真主要是考察每个子模块或进程的功能与基本时序。在系统级与功能模块级设计层次,必须整体权衡多种实现方案之间孰优孰劣,根据系统性能指标要求,从整体上优化实现方案,从而更有效地满足设计需求。