



普通高等教育电气工程与自动化(应用型)“十二五”规划教材

**EDA Technology
and Application**

EDA 技术与应用

◎ 陈海宴 主 编

◎ 游余新 郑玉珍 副主编



机械工业出版社
CHINA MACHINE PRESS

普通高等教育电气工程与自动化（应用型）“十二五”规划教材

EDA 技术与应用

主 编 陈海宴

副主编 游余新 郑玉珍



机械工业出版社

本书根据课堂教学的要求,深入浅出地对 EDA 技术、Verilog HDL (硬件描述语言)、可编程逻辑开发应用及相关知识做了系统的介绍,使读者能初步了解和掌握 EDA 的基本内容及实用技术。

全书共分 10 章,内容涉及 EDA 的基本知识、可编程逻辑器件的结构和工作原理、Quartus II 软件开发应用、Verilog HDL 语法知识、设计的层次与常用模块设计、宏功能模块设计、可综合设计与优化、系统仿真与 ModelSim 软件使用、数字设计实例、C/C++ 语言开发可编程逻辑器件等。书中的例子均给出了介绍、程序代码和仿真结果。使用 Quartus II 软件平台,通过了 ModelSim 的仿真测试。各章都配有一定数量的习题。

本书内容翔实,语言通俗易懂,可以帮助初学者在短时间内学习 EDA 技术和用 Verilog HDL 进行硬件电路的设计,并进一步拓展读者的视野到可综合的 C/C++ 设计,可作为通信、电子、自动化、计算机等相关专业的教材,也可作为电子设计和开发人员学习 EDA 技术和 Verilog HDL 的参考用书。

本书配有免费电子课件,欢迎选用本书作教材的老师发邮件到 jinacmp@163.com 索取,或登录 www.cmpedu.com 注册下载。

图书在版编目(CIP)数据

EDA 技术与应用/陈海宴主编. —北京:机械工业出版社,2012.4
普通高等教育电气工程与自动化(应用型)“十二五”规划教材
ISBN 978-7-111-37682-8

I. ①E… II. ①陈… III. ①电子电路—电路设计:计算机辅助设计—高等学校—教材 IV. ①TN702

中国版本图书馆 CIP 数据核字(2012)第 041168 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

策划编辑:吉玲 责任编辑:吉玲 王荣 刘丽敏

版式设计:霍永明 责任校对:闫玥红

封面设计:张静 责任印制:杨曦

北京双青印刷厂印刷

2012 年 5 月第 1 版第 1 次印刷

184mm×260mm·16.25 印张·408 千字

标准书号:ISBN 978-7-111-37682-8

定价:33.00 元



凡购本书,如有缺页、倒页、脱页,由本社发行部调换

电话服务

网络服务

社服务中心:(010) 88361066

门户网: <http://www.cmpbook.com>

销售一部:(010) 68326294

教材网: <http://www.cmpedu.com>

销售二部:(010) 88379649

读者购书热线:(010) 88379203

封面无防伪标均为盗版

普通高等教育电气工程与自动化（应用型）“十二五”规划教材

编审委员会委员名单

主任委员：刘国荣

副主任委员：

张德江	梁景凯	张元	袁德成	焦斌
吕进	胡国文	刘启中	汤天浩	黄家善
钱平	王保家			

委员（按姓氏笔画排序）：

丁元明	马修水	王再英	王军	叶树江
孙晓云	朱一纶	张立臣	李先允	李秀娟
李海富	杨宁	陈志新	周渊深	尚丽萍
罗文广	罗印升	罗兵	范立南	娄国焕
赵巧娥	项新建	徐建英	郭伟	高亮
韩成浩	蔡子亮	樊立萍	穆向阳	

前 言

目前,EDA 技术已经成为现代电子设计领域的基本手段,涵盖印制电路板 (PCB) 设计、可编程逻辑器件开发、专用集成芯片设计以及系统验证等诸多领域。硬件描述语言 (HDL) 是 EDA 技术中的一个重要组成部分,可应用于除 PCB 设计外的各个领域。Verilog HDL 和 VHDL 是两个主流 HDL。Verilog HDL 要比 VHDL 简单,而且 Verilog HDL 与 C 语言语法风格类似,更容易被在校大学生和初学者接受。利用 Verilog HDL 设计数字逻辑电路和数字系统,是电子电路设计领域的一次革命性变化,也是 21 世纪的 IC 设计工程师所必须掌握的专业知识。

全书内容分为 10 章:第 1 章为 EDA 技术概述,第 2 章为可编程逻辑器件基础,第 3 章为 Quartus II 开发软件应用,第 4 章为 Verilog HDL 的基本语法,第 5 章为 Verilog 设计的层次与常用模块设计,第 6 章为宏功能模块设计,第 7 章为可综合设计与优化,第 8 章为系统仿真与 ModelSim 软件使用,第 9 章为数字设计实例,第 10 章为 C/C++ 语言开发可编程逻辑器件,读者可参考学习。书中的部分图是国外软件生成的,因此图形符号与国内的会有所差别,请读者注意相互对比。

本书体系完整、层次清晰、通俗易懂、学练结合,在以下几个方面具有一定的特色:

(1) 内容编排层次清晰、循序渐进,以 Verilog HDL 开发为主线符合学习和应用规律。

(2) Verilog HDL 从 C 语言中继承了多种操作符和结构,从形式上看,Verilog HDL 和 C 语言有很多相似之处,它的核心子集非常易于学习和使用。

(3) 遵循硬件应用系统开发的基本步骤和思路,详细讲解 Quartus II 和 ModelSim 开发工具的使用,具备很强的指导性和可读性。

(4) 阐述如何进一步提高设计抽象层次,进行 C/C++ 语言的可综合设计,进行架构分析和验证,无缝地从 C/C++ 到 RTL 到 FPGA 的板级下载流程。

(5) 实例注重教学实效,突出 EDA 课程实践性强的特点,提高读者实践动手能力和工程设计能力。

本书可以帮助初学者在短时间内学习 EDA 技术和用 Verilog HDL 进行硬件电路的设计,可作为通信、电子、自动化、计算机等相关专业的教材,也可作为电子设计和开发人员学习 EDA 技术和 Verilog HDL 参考用书。

参加本书编写的有陈海宴、游余新、郑玉珍、鲁奔、李照龙、吴亚萍、曾小星、黄蒙、李治、王芬芬、王斌、杨琳娟、田芳、杨明、哈森其其格、卢东华、孟繁荣、李华和付朋辉等。

哈尔滨工业大学(威海)的戴伏生教授审阅了本书的大纲,并提出了宝贵的意见,在此表示感谢。同时,也要感谢在本书的编写过程中给予我们支持的许多专家和同行。

鉴于编者水平有限,书中难免存在疏漏和错误之处,恳请专家和广大读者批评指正。有兴趣的读者,可以发送电子邮件到 chenhy736@sina.com,与编者进一步交流。

编 者

目 录

前言

第 1 章 EDA 技术概述 1

- 1.1 EDA 技术简介 1
- 1.2 EDA 技术的发展和优势 2
 - 1.2.1 EDA 技术的发展 2
 - 1.2.2 EDA 技术的优势 3
- 1.3 硬件描述语言 (HDL) 4
 - 1.3.1 原理图设计方法 4
 - 1.3.2 HDL 的设计方法 4
 - 1.3.3 HDL 设计方法与传统原理图设计方法的比较 5
- 1.4 综合 6
- 1.5 基于 HDL 的设计方法 7
- 1.6 EDA 工程的设计流程 8
- 1.7 EDA 集成开发工具简介 10
- 1.8 IP 核 12
- 1.9 小结 12
- 1.10 习题 13

第 2 章 可编程逻辑器件基础 14

- 2.1 可编程逻辑器件概述 14
 - 2.1.1 可编程逻辑器件的发展过程 14
 - 2.1.2 可编程逻辑器件的分类 15
- 2.2 PROM、PLA、PAL 和 GAL 的基本结构 16
 - 2.2.1 逻辑电路符号的表示方法 16
 - 2.2.2 PLD 器件的基本结构 17
 - 2.2.3 PROM 的基本结构 17
 - 2.2.4 PLA 的基本结构 17
 - 2.2.5 PAL 和 GAL 的基本结构 18
 - 2.2.6 PROM、PLA、PAL 和 GAL 电路的结构特点 20
- 2.3 CPLD 的基本结构和工作原理 21
 - 2.3.1 CPLD 的基本结构 21
 - 2.3.2 Altera 公司的 CPLD 23
- 2.4 FPGA 的结构和工作原理 26
 - 2.4.1 FPGA 的基本结构 27
 - 2.4.2 Altera 公司的 FPGA 29
- 2.5 CPLD/FPGA 的应用选型 33

2.6 小结 34

2.7 习题 34

第 3 章 Quartus II 开发软件应用 36

- 3.1 Quartus II 软件设计流程 36
- 3.2 Quartus II 软件安装 36
- 3.3 创建工程文件 43
 - 3.3.1 建立工程 43
 - 3.3.2 建立设计文件 48
 - 3.3.3 原理图输入方法 50
 - 3.3.4 文本输入设计方法 52
 - 3.3.5 编译 53
- 3.4 约束输入 53
 - 3.4.1 器件选择 53
 - 3.4.2 引脚分配及验证 54
 - 3.4.3 使用 “Assignment Editor” 和 “Settings” 对话框 54
- 3.5 综合和仿真 58
 - 3.5.1 使用 Quartus II 的集成综合 58
 - 3.5.2 使用 Quartus II 的仿真器进行仿真设计 59
- 3.6 下载配置 62
 - 3.6.1 JTAG 模式 62
 - 3.6.2 AS 模式 62
- 3.7 实例: 3 线-8 线译码器设计与仿真 63
 - 3.7.1 实例简介 63
 - 3.7.2 实例目的 63
 - 3.7.3 实例内容 63
- 3.8 小结 68
- 3.9 习题 68

第 4 章 Verilog HDL 的基本语法 69

- 4.1 Verilog 简介 69
 - 4.1.1 Verilog HDL 的发展过程 69
 - 4.1.2 Verilog HDL 与 C 语言的比较 70
- 4.2 Verilog HDL 设计举例 72
- 4.3 Verilog 模块的结构 74
- 4.4 Verilog HDL 的要素与表达式 78
 - 4.4.1 注释 79
 - 4.4.2 常量 79

4.4.3 变量	80	5.5.2 或非门电路	121
4.4.4 操作符	82	5.5.3 异或门电路	122
4.4.5 字符串、关键字、标识符	89	5.5.4 三态门电路	122
4.5 赋值语句	89	5.5.5 编码器	123
4.5.1 连续赋值	90	5.5.6 译码器	124
4.5.2 过程赋值	91	5.5.7 BCD-七段显示译码器	126
4.5.3 连续赋值和过程赋值的不同	94	5.5.8 2选1数据选择器	128
4.6 块语句	95	5.5.9 4选1数据选择器	129
4.6.1 顺序语句块 (begin-end)	95	5.5.10 数值比较器	130
4.6.2 并行语句块 (fork-join)	96	5.5.11 总线缓冲器	131
4.6.3 起始时间和结束时间	97	5.6 基本时序电路设计	132
4.7 条件语句	97	5.6.1 触发器	132
4.7.1 if-else 语句	97	5.6.2 寄存器	133
4.7.2 case 语句	98	5.6.3 计数器	134
4.7.3 比较 if-else 嵌套与 case 语句	100	5.6.4 串-并转换器	135
4.8 循环语句	100	5.7 加法器设计	136
4.8.1 for 语句	100	5.7.1 并行加法器	137
4.8.2 forever 语句	101	5.7.2 流水线加法器	137
4.8.3 repeat 语句	101	5.8 乘法器设计	139
4.8.4 while 语句	102	5.8.1 并行乘法器	139
4.9 过程语句	102	5.8.2 查找表乘法器	140
4.9.1 initial 语句	102	5.9 乘累加器设计	142
4.9.2 always 语句	104	5.10 小结	143
4.10 任务与函数	106	5.11 习题	143
4.10.1 任务	107	第6章 宏功能模块设计	144
4.10.2 函数	109	6.1 算术运算模块库	144
4.11 预编译指令	110	6.1.1 算术运算模块库模块列表	144
4.11.1 宏定义语句 ('define、'undef)	110	6.1.2 乘法器模块设计举例	144
4.11.2 文件包含语句 ('include)	111	6.1.3 计数器模块设计举例	148
4.11.3 时间尺度 ('timescale)	113	6.2 逻辑门库	151
4.11.4 条件编译指令 ('ifdef、'else、 'endif)	113	6.2.1 逻辑门库宏模块列表	151
4.12 小结	114	6.2.2 3线-8线译码器模块设计举例	152
4.13 习题	114	6.3 I/O 模块库	154
第5章 Verilog 设计的层次与常用 模块设计	116	6.4 存储器模块库	155
5.1 Verilog 设计的层次	116	6.4.1 存储区模块库宏模块及功能 描述	155
5.2 行为描述	116	6.4.2 参数化 RAM 模块设计举例	156
5.3 数据流描述	117	6.5 小结	157
5.4 结构描述	117	6.6 习题	157
5.4.1 Verilog 内置门元件	117	第7章 可综合设计与优化	158
5.4.2 门元件的调用	118	7.1 可综合设计	158
5.5 基本组合逻辑电路设计	119	7.1.1 综合的概念及其过程	158
5.5.1 与非门电路	120	7.1.2 可综合模型的设计	159
		7.1.3 综合结果的验证	161

7.2 Verilog HDL 设计优化	162	9.1.1 卷积码的编码工作原理	206
7.2.1 公因子和公因子表达式	162	9.1.2 卷积码的 Verilog 实现	207
7.2.2 算术表达式优化	163	9.1.3 卷积码的 ModelSim 仿真	208
7.2.3 运算符优化	163	9.2 通用异步收发器的 Verilog HDL 设计 与验证	209
7.2.4 循环语句的优化	163	9.2.1 通用异步收发器的规范	209
7.3 面积与速度的折中	164	9.2.2 电路结构设计	210
7.3.1 速度换面积	164	9.2.3 UART 控制电路模块的代码设计 与分析	212
7.3.2 面积换速度	165	9.2.4 发送电路的代码设计与仿真 分析	216
7.4 有限状态机设计	166	9.2.5 接收电路的代码设计与仿真	220
7.4.1 有限状态机的设计步骤	166	9.2.6 UART 系统仿真	225
7.4.2 有限状态机编码方式	167	9.2.7 UART 自动测试 Testbench	228
7.4.3 用 Verilog HDL 设计可综合的 状态机的指导原则	167	9.3 小结	231
7.4.4 状态机的 3 种设计风格	168	9.4 习题	231
7.5 小结	175		
7.6 习题	175		
第 8 章 系统仿真与 ModelSim 软件 使用	176	第 10 章 C/C++ 语言开发可编程 逻辑器件	232
8.1 系统任务与函数	176	10.1 基于 C/C++ 的硬件设计方法	232
8.2 用户自定义原语	180	10.2 硬件设计的 C++ 数据类型	233
8.3 应用 Testbench 仿真验证	181	10.2.1 ac_int 型	233
8.3.1 基本结构	181	10.2.2 ac_fixed 型	233
8.3.2 验证过程	182	10.3 C/C++ FIR 滤波器设计	234
8.3.3 验证的全面性与代码覆盖率 分析	185	10.3.1 直接型 FIR 滤波器	234
8.4 应用 ModelSim 软件仿真	190	10.3.2 奇对称 FIR 滤波器	235
8.4.1 软件简介	190	10.3.3 转置型 FIR 滤波器	235
8.4.2 ModelSim 软件的安装过程	191	10.4 C++ 滤波器的可编程逻辑实现及 验证	236
8.4.3 使用 ModelSim 进行设计仿真	192	10.4.1 C++ FIR 滤波器的实现	236
8.4.4 在 Quartus II 中直接调用 ModelSim	196	10.4.2 FIR 滤波器的验证	244
8.5 实例: 4 位全加器设计及 ModelSim 仿真	200	10.5 小结	245
8.5.1 实例简介	200	10.6 习题	247
8.5.2 实例目的	200		
8.5.3 实例内容	200	附录 Quartus II 支持的 Verilog 结构	248
8.6 小结	205	附录 A Quartus II 对 Verilog 的支持 情况 1	248
8.7 习题	205	附录 B Quartus II 对 Verilog 的支持 情况 2	249
第 9 章 数字设计实例	206	参考文献	250
9.1 卷积编码 Verilog HDL 设计	206		

第 1 章 EDA 技术概述

本章主要讲述 EDA 技术的概念、发展历史和技术优势,介绍 EDA 技术中非常重要的几个方面: HDL、仿真、综合,自顶向下 (Top-Down) 及自底而上 (Bottom-Up) 的设计方法,以及 EDA 的设计流程和集成开发工具等如 Quartus II 软件的特点,最后介绍 IP 核的概念和特点。

1.1 EDA 技术简介

电子设计自动化 (Electronic Design Automation, EDA) 技术是指利用计算机完成电子系统的设计,以计算机和微电子技术为先导的先进技术,汇集了计算机图形学、拓扑学、逻辑学、微电子工艺与结构学以及计算数学等多种计算机应用学科的最新成果。EDA 技术是电子设计技术的发展趋势,利用 EDA 工具可以代替设计者完成电子系统设计中的大部分工作,设计人员只需完成对系统功能的描述,就可以由计算机软件进行处理,得到设计结果,而且修改设计如同修改软件一样方便,可以极大地提高设计效率。

广义的 EDA 技术应用于半导体工艺设计自动化、可编程器件设计自动化、电子系统设计自动化、印制电路板设计自动化、仿真与测试等领域。狭义的 EDA 技术是指以大规模可编程逻辑器件或专用集成芯片为设计载体,以硬件描述语言为系统逻辑描述的主要表达方式,以计算机、大规模可编程逻辑器件或专用集成芯片的开发软件及实验开发系统为设计工具,自动完成用软件方式描述的电子系统到硬件系统的逻辑编译、逻辑简化、逻辑分割、逻辑综合及优化、布局布线、逻辑仿真,直至完成对特定目标芯片的适配编译、逻辑映射、编程下载等工作,最终形成集成电子系统或专用集成芯片的一门多学科融合的新技术。本书将主要介绍应用于大规模可编程逻辑器件 CPLD/FPGA 的 EDA 技术。

随着微电子技术以惊人的速度发展,其工艺水平已达到深亚微米级,并仍在追随 Gordon Moore (摩尔) 定律,每 18 个月设计规模翻一番,在一个芯片上可集成数千万乃至上亿万只晶体管,工作速度可达到 Gbit/s 级,人们已经能够把一个完整的电子系统集成在一个芯片上,即 SOC (System On Chip)。可编程逻辑器件 (Programmable Logic Device, PLD) 的出现极大地改变了传统的电子系统设计方法。PLD 自 20 世纪 70 年代后开始发展,经历了可编程逻辑阵列 (Programmable Logic Array, PLA)、通用阵列逻辑 (Generic Array Logic, GAL)、现场可编程门阵列 (Field Programmable Gate Array, FPGA) 和复杂可编程逻辑器件 (Complex Programmable Logic Device, CPLD) 等阶段,PLD 的广泛使用不仅简化了电路设计、降低了研制成本、提高了系统可靠性,而且给数字系统的设计和实现过程带来了革命性变化。电子系统的设计方法从 CAD (Computer Aided Design)、CAE (Computer Aided Engineering) 到 EDA,设计的自动化程度越来越高,设计的复杂性也越来越强。

EDA 技术是现代电子设计的有效手段,如果没有 EDA 技术的支持,要完成超大规模集成电路的设计和制造的复杂度是不可想象的,当然,EDA 技术也是随着电子技术的发展而不断进步的。

1.2 EDA 技术的发展和优势

随着计算机技术的发展,从 20 世纪 60 年代中期开始,人们就不断开发出各种计算机辅助设计工具来帮助设计人员进行电子系统的设计。电路理论和半导体工艺水平的提高,也对 EDA 技术的发展起了巨大的推进作用,使 EDA 作用范围从 PCB (Printed Circuit Board, 印制电路板) 设计延伸到电子线路和集成电路设计,直至整个系统的设计,使 IC 芯片系统应用、电路制作和整个电子系统生产过程都集成在一个环境之中。

1.2.1 EDA 技术的发展

一般认为 EDA 技术发展大致分为以下 3 个阶段。

1. CAD 阶段

20 世纪 70 年代,随着中小规模集成电路的开发应用,越来越多不同外形的元器件被用到印制电路板上;每片集成电路包含的元器件也从几十、几百到几千甚至几万。传统的手工制图设计印制电路板和集成电路的方法已无法满足设计的精度和效率要求。因此工程师们开始进行二维平面图形的计算机辅助设计,以便解脱繁杂、机械的版图设计工作,这就产生了第一代 EDA 工具,即 CAD (Computer Aided Design) 软件。这一阶段的特点是一些单独的工具软件,主要实现 PCB 布线设计、电路模拟、逻辑模拟及版图的绘制等,通过使用计算机,将设计人员从大量烦琐重复的计算和绘图工作中解脱出来。常用的 Protel 和 Altium Designer,以及用于电路模拟的 SPICE 软件和后来产品化的 IC 版图编辑与设计规则检查系统等软件,都是这个阶段的产品。

CAD 工具存在的问题主要有两个方面:第一,由于各个工具软件是由不同的公司和专家开发的,只能解决一个领域的问题,完成一个电子系统的设计需要轮流使用不同的软件,设计效率较低;第二,缺乏系统级的设计考虑,对于复杂电子系统的设计,不能提供系统级的仿真与综合,设计错误如果在开发后期才被发现,将给修改工作带来极大不便。

2. CAE 阶段

随着集成电路规模的不断扩大,以及电子系统设计的逐步复杂,电子设计 CAD 的工具随之发展和完善,人们在集成电路与电子系统设计方法学以及设计工具集成化方面取得了许多成果,进入 CAE (Computer Aided Engineering) 阶段。在这个阶段,各种设计工具如原理图输入、编译与连接、逻辑模拟、测试码生成、版图自动布局以及各种单元库已经齐全,由于采用了统一数据管理技术,因而能够将各个工具集成为一个 CAE 系统,按照设计方法学制定的设计流程,可以实现从设计输入到版图输出的全程自动化。设计者能在产品制作之前预知产品的功能与性能,能生成产品制造文件,在设计阶段对产品性能的分析前进了一大步。多数 CAE 系统还集成了 PCB 自动布局布线软件及热特性、噪声、可靠性等分析软件,进而可以实现电子系统设计自动化。

如果说自动布局布线的 CAD 工具代替了设计工作中绘图的重复劳动,那么具有自动综合能力的 CAE 工具则代替了设计者的部分工作,对保证电子系统的设计、制造出最佳的电子产品起着关键的作用。但是,大部分从原理图出发的电子设计工具软件仍然不能适应复杂电子系统的设计要求,而具体化的元器件图形又制约着优化设计。

3. EDA 阶段

传统的数字电子系统设计采用“自底向上”（Bottom-Up）搭积木的方法，将具有固定功能的通用芯片如 74 系列 TTL 器件或 4000 系列 CMOS 器件搭建成系统，缺乏灵活性，不易实现大型系统的设计，且设计过程中产生大量的设计文档，不易管理。

20 世纪 90 年代以后，EDA 技术提供了一种“自顶向下”（Top-Down）的全新设计方法。首先从系统设计入手，在顶层进行功能框图的划分和结构设计，在框图一级进行仿真、调试。用硬件描述语言对高层次的系统行为进行描述，在系统一级进行验证。然后用综合优化工具生成具体门电路的网表，其对应的物理实现级可以是印制电路板或专用集成电路。由于设计的主要仿真和调试过程是在较高层次上完成的，有利于早期发现结构设计上的错误，避免设计工作的浪费，同时也减少了逻辑功能仿真的工作量，提高了设计的一次成功率。

同时，设计师逐步从使用硬件转向设计硬件，从电路级电子产品开发转向系统级电子产品开发，相应地对电子系统的设计工具提出了更高的要求。这个阶段出现了以高级语言描述、系统仿真和综合技术为特征的第三代 EDA 技术，以系统级设计为核心，包括系统行为级描述与结构级综合，系统仿真与测试验证，系统划分与指标分配，系统决策与文件生成等一整套的 EDA 工具，不仅极大地提高了系统的设计效率，而且使设计人员摆脱了大量的辅助性及基础性工作，将精力集中于创造性的方案与概念的构思上，属于高层次的电子设计方法。

1.2.2 EDA 技术的优势

与传统的数字电子系统设计方法相比较，EDA 技术运用硬件描述语言（Hardware Description Language, HDL）进行电子系统设计，具有很多优势，具体表现为：

- 1) 以软件设计的方式来设计硬件，提高了设计的自由度，减少所需芯片种类和数量，使整个系统可集成在一个芯片上，体积小、功耗低、可靠性高。
- 2) HDL 比传统的电路图设计更适合于描述大规模、功能复杂的数字系统，标准化的语言便于设计的复用、交流、修改和文档的管理保存。
- 3) 采用 HDL 设计电子系统时，与具体的器件无关，可以在不同的 PLD 器件上实现，设计者拥有完全的自主知识产权。
- 4) 用软件方式设计的系统到硬件系统物理实现之间的转换是由 EDA 工具软件自动完成的，降低对设计者硬件知识和硬件经验的要求。
- 5) 在 EDA 设计过程中可用软件进行各个阶段的仿真，保证设计过程的正确性，降低设计成本，缩短设计周期。
- 6) EDA 技术可以实现系统现场编程、在线升级，为产品更新换代提供极大便利。
- 7) EDA 工具采用标准化和开放性的框架结构，与硬件平台无关的用户界面可以实现各种 EDA 工具间的优化组合，实现资源共享，有利于大规模、有组织的设计开发工作。
- 8) EDA 工具软件配有丰富的库，如元器件图形符号库、元器件模型库、工艺参数库、标准单元库、电路模块库以及 IP 库等，适用于电子系统设计的各个阶段。

EDA 技术在仿真、时序分析、集成电路自动测试、高速印制电路板设计及开发操作平台的扩展等方面取得新的突破，向着功能强大、简单易学、使用方便的方向发展。一方面，EDA 设计正从主要着眼于数字逻辑向模拟电路和数-模混合电路的方向发展，开发工具要具有混合信号处理能力；另一方面，在硅集成电路制造工艺已进入超深亚微米（Very Deep

Sub-Micron, VDSM) 阶段, 可编程逻辑器件向高密度、高速度、宽频带方向发展, 随着芯片集成度提高, 单个芯片内集成通用微控制器/微处理器核心 (MCU/MPU Core)、专用数字信号处理器核心 (DSP Core)、存储器核心 (Memory Core)、嵌入式软件/硬件、数-模混合器件、RF 处理器等, EDA 技术与上述器件间的物理与功能界限已日益模糊, 系统描述方式需简化、高效化和统一化。随着 EDA 技术的不断成熟, 软件和硬件的概念将日益模糊, 使用单一的高级语言 (如 C/C++/SystemC) 直接设计整个系统将成为发展趋势。

1.3 硬件描述语言 (HDL)

1.3.1 原理图设计方法

设计一个数字逻辑系统时, 传统的做法是设计一张电路图, 电路图中由电路符号表示基本设计单元, 线表示信号的连接。电路符号通常取自构造电路图的零件库中, 例如标准逻辑器件 (如 74 系列等) 的符号取自标准逻辑零件符号库, 专用集成电路 (Application Specific Integrated Circuit, ASIC) 所需符号可以取自 ASIC 库的专用宏单元, 这就是传统的原理图设计方法。为了能够对设计进行验证, 设计者必须通过搭建硬件平台例如电路板来进行验证。

1.3.2 HDL 的设计方法

随着电子系统设计的集成度、复杂度越来越高, 传统的原理图设计方法已经不能满足设计的要求, 因此需要借助当今先进的 EDA 工具, 使用一种描述语言, 对数字电路和数字逻辑系统能够进行形式化的描述, 这就是硬件描述语言。设计者利用 HDL 来描述自己的设计思想, 利用 EDA 工具进行仿真, 并自动综合到门级电路, 最后由 ASIC 或 FPGA 实现功能。例如设计一个 2 输入与门, 传统的方法可能从标准器件库中调用一个 74 系列的器件, 但在硬件描述语言中, 可以用 “&” 的形式来描述一个与门, “C = A&B” 就是一个 2 输入与门的描述, 而 “and” 就是一个与门器件。

常见的硬件描述语言包括 VHDL、Verilog HDL、AHDL、System Verilog 和 System C 等, 但在 IEEE 工业标准中, 主要有 VHDL 和 Verilog HDL, 这是当前最流行的硬件描述语言, 得到几乎所有主流 EDA 工具的支持。VHDL 发展较早, 始于美国国防部的超高速集成电路计划, 目的是给出一种与工艺无关、支持大规模系统设计的标准方法和手段, 其语法严格, 是一种全方位的硬件描述语言, 包括系统行为级、寄存器传输级和逻辑门级多个设计层次, 支持结构、数据流、行为 3 种描述形式的混合描述, 自顶向下或自底向上的电路设计过程都可以用 VHDL 来完成, 是数字电路设计的方法之一。

Verilog HDL 是在 C 语言的基础上发展起来的, 语法较自由, 具有简洁、高效、易用的特点, Verilog HDL 最初是由 Gateway Design Automation 公司于 1983 年为其模拟器产品开发的硬件建模语言, 于 1995 年成为 IEEE 标准。Verilog HDL 用于从算法级 (Algorithm Level)、寄存器传送级 (Register Transfer Level)、门级 (Gate Level) 到版图级 (Layout Level) 等各个层次的数字系统建模, 设计的规模可以是任意的, Verilog HDL 不对设计的规模大小施加任何限制。不同层次的描述方式如表 1-1 所示。

Verilog HDL 可以采用 3 种不同方式或混合方式对设计建模, 包括: 行为描述方式即使用过程化结构建模; 数据流方式即使用连续赋值语句方式建模; 结构化方式即使用门和模块

实例语句描述建模。此外，Verilog HDL 提供了编程语言接口，通过该接口可以在模拟、验证期间从设计外部访问设计，包括模拟的具体控制和运行，完整的 HDL 足以对从最复杂的芯片到完整的电子系统进行描述。

表 1-1 Verilog HDL 各层次描述方式

设计层次	行为描述	结构描述
行为级	系统算法	系统逻辑框图
RTL 级	数据流图、真值表、状态机	寄存器、ALU、ROM 等分模块描述
门级	布尔方程、真值表	逻辑门、触发器、锁存器构成的逻辑图
版图级	几何图形	图形连接关系

Verilog HDL 不仅定义了语法，而且对每个语法结构都定义了清晰的模拟、仿真语义。因此，用这种语言编写的模型能够使用 Verilog 仿真器进行验证。Verilog HDL 从 C 语言中继承了多种操作符和结构，提供了扩展的建模能力，核心子集非常易于学习和使用。Verilog HDL 作为标准化的硬件设计语言，设计时独立于器件，可以很容易地把完成的设计移植到不同厂家的不同芯片中去，信号参数也很容易改变。Verilog HDL 设计与工艺无关，使得设计者在功能设计、逻辑验证阶段可以不必过多考虑门级与工艺实现的具体细节，只是根据系统设计时对芯片的需要，施加不同的约束条件，即可设计出实际电路，具有很强的移植能力。

VHDL 与 Verilog HDL 都可以在不同层次上进行电路描述，并且最终都要转换成门电路级才能被布线器或适配器接受。与 VHDL 相比，Verilog HDL 最大的优点是易学易用，编程风格灵活简洁，在美国许多著名高校都以 Verilog HDL 作为主要授课内容。

1.3.3 HDL 设计方法与传统原理图设计方法的比较

HDL 和传统的原理图输入方法的关系就好比是高级语言和汇编语言的关系。HDL 的可移植性好，使用方便，易于共享和复用，但效率不如原理图；原理图输入的可控性好，效率高，比较直观，但设计大规模 CPLD/FPGA 时显得很烦琐，移植性差。HDL 更适合大规模数字系统的设计，例如设计一个 32 位的加法器，利用传统图形输入软件需要输入 500 ~ 1000 个门，而利用 HDL 只需要用 “ $A = B + C$ ” 即可表达，在实际的 PLD 设计中，通常建议采用原理图和 HDL 结合的方法来设计。需要注意的是，HDL 描述的毕竟是硬件电路，包含许多硬件特有的结构和特点，例如电路具有并行性，程序在调试时不能采用单步执行等调试手段等。

用硬件描述语言（HDL）开发可编程逻辑器件的流程一般可分为文本编辑、功能仿真、逻辑综合、布局布线、时序仿真和编程下载等阶段。任何文本编辑器都可以进行文本编辑，通常 VHDL 的源程序保存为 .vhd 文件，Verilog HDL 的源程序保存为 .v 文件。功能仿真需要将文件调入 HDL 仿真软件进行，主要检查逻辑功能是否正确，而不检查电路的时序，也称为前仿真，简单的设计可以不进行功能仿真。逻辑综合是把 HDL 的描述综合成最简化的布尔表达式和信号的连接关系，并会生成 .edf (edif) 的 EDA 工业标准文件。将 .edf 文件调入 PLD 厂商提供的软件中进行布局布线，即把设计好的逻辑映射到 PLD/FPGA 内。时序仿真需要利用在布局布线中获得的精确参数，用仿真软件验证电路的时序，也称为后仿真。

确认仿真无误后就可以编程下载，将编程文件下载到可编程逻辑器件中。

总的说来，硬件描述语言（HDL）是用来设计电子系统硬件电路的计算机语言，它用软件编程的方式来描述电子系统的逻辑功能、信号连接和时序关系，采用形式化方式描述数字电路、设计数字逻辑系统。硬件描述语言是 EDA 技术的重要组成部分，也是 EDA 技术发展的高级阶段的一个重要标志。

1.4 综合

EDA 技术可以在不同层次上进行数字逻辑系统设计，如图 1-1 所示。综合（Synthesis）是将较高层次的设计描述自动转化为较低层次描述的过程。

1. 综合的任务

综合的任务是根据设计目标与要求将高级语言、原理图等设计输入翻译成由与、或、非逻辑门，存储器或触发器等基本逻辑单元所组成的逻辑连接（网表），供 CPLD/FPGA 厂商的布局布线器进行实现。综合分为行为综合、逻辑综合和版图综合或结构综合。行为综合是指从算法表示、行为描述转换到寄存器传输级（RTL）；逻辑综合（RTL 综合）是指将 RTL 级描述转换到逻辑门级（包括触发器）；版图综合或结构综合是从逻辑门表示转换到版图表示，或转换到 CPLD/FPGA 器件的配置网表表示。

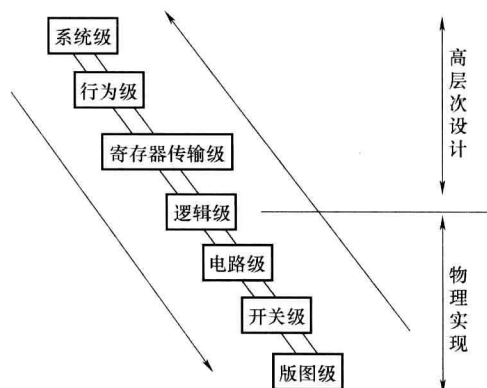


图 1-1 EDA 设计层次级别

2. 综合器的功能

综合器的功能就是将设计描述与给定硬件结构用某种网表文件的方式联系起来，显然，综合器是设计描述与硬件实现之间的一座桥梁。RTL 综合器是 EDA 技术实施电路设计中完成电路简化、算法优化、硬件结构细化的计算机软件，是将硬件描述语言转化为硬件电路的重要工具。

3. 综合的过程

RTL 综合器在把 HDL 源程序转化成硬件电路时一般经过以下两个步骤：首先，对 HDL 源码进行处理分析，产生一个与实现技术无关的通用原理图；然后根据设计要求执行优化算法、化简状态和布尔方程，使之满足各种约束条件，按半导体工艺要求，采用相应的工艺库，把优化的布尔描述映射到实际的逻辑电路网表。RTL 综合器的输出文件一般是网表文件，可以是用于电路设计数据交换和交流的工业标准化格式的文件，或者是直接用 HDL 表达的标准格式网表文件，也可以是对应 FPGA/CPLD 器件厂商的网表文件。

4. 比较硬件综合器与软件编译器

硬件综合器与软件编译器的作用是不同的，软件语言设计与硬件语言设计的目标流程如图 1-2 所示。用软件语言如 C 或汇编语言编写的源程序经过编译器产生机器可执行的代码流，而设计硬件电路时，HDL 编写的源程序经过综合器产生电路网表文件，才能下载到可编程逻辑器件中，实现系统功能。

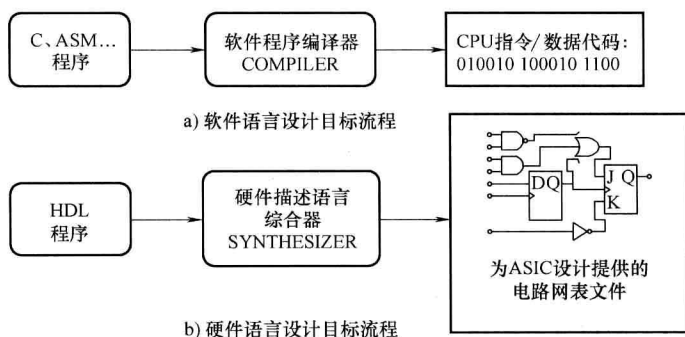


图 1-2 软件编译器与硬件综合器的作用

1.5 基于 HDL 的设计方法

在基于 EDA 技术的设计中，通常有两种设计思路，一种是自底向上（Bottom-Up）的设计方法，另一种是自顶向下（Top-Down）的设计方法。

1. 自底向上的设计方法

数字逻辑系统传统的设计方法通常采用搭积木的方式，将各种标准芯片如 74/54 系列的 TTL 器件或 4000/4500 系列的 CMOS 器件加上外围电路构成模块，由这些模块进一步形成各种功能电路，进而构成系统，是一种自底向上的设计方法，如图 1-3 所示。自底向上的设计方法好比用一砖一瓦建造金字塔，效率低，容易出错且不易修改。

2. 自顶向下的设计方法

可编程逻辑器件和 EDA 技术的发展提出了更符合人们逻辑思维习惯的自顶向下设计方法，使人们可以立足于 PLD 芯片，自己定义器件的内部逻辑和引脚，通过芯片设计来实现各种数字逻辑功能。由于引脚定义的灵活性，可以减轻原理图和印制电路板设计的工作量和难度，增加了设计的自由度，提高设计效率，同时也减少了所需芯片的数量，减小了系统体积，降低了功耗，提高了系统可靠性。自顶向下的设计方法如图 1-4 所示，从系统级入手，在顶层进行功能划分和结构设计，用 HDL 语言对高层次的系统进行行为描述。这样按照从

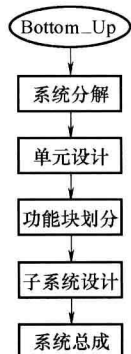


图 1-3 自底向上（Bottom-Up）的设计方法

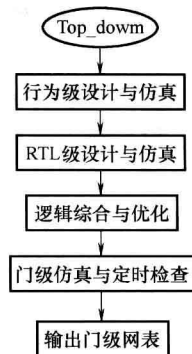


图 1-4 自顶向下（Top-Down）设计方法

上到下的顺序, 在各个层次上进行设计和仿真, 有利于早期发现结构设计上的错误, 提高设计成功率。

在自顶向下的设计过程中, 有些步骤可以由 EDA 工具软件自动完成, 如逻辑综合, 有些步骤 EDA 工具只是提供操作平台。系统的设计需要经过“设计—验证—修改—再验证”的反复过程, 直到能够完全实现要求的逻辑功能和性能。当然设计过程中也不绝对是自顶向下, 有时也需要用自底向上的方法, 在系统划分的基础上, 先进行底层单元设计, 再逐步向上进行功能块和子系统的设计。

3. 基于 HDL 的自顶向下设计方法的实施步骤

1) 按照自顶向下的设计方法进行系统划分。

2) 输入 HDL 源代码, 这是高层次设计中最为普遍的输入方式。当然也可以采用图形输入方式 (如符号图、状态图等), 图形输入方式具有直观、容易理解的优点。

3) 将以上的设计输入编译成标准的 HDL 文件。对于大型设计, 应该进行代码级的功能仿真, 主要是检验系统逻辑功能设计的正确性。因为对于大型设计, 综合和适配可能需要花费数小时, 所以在综合前对源代码仿真, 就可以大大减少设计重复的次数和时间, 对于简单的设计, 可以略去功能仿真。

4) 利用综合器对 HDL 源代码进行综合优化处理, 生成门级描述的网表文件, 这是将高层次 HDL 描述转化为硬件电路的关键步骤。综合优化是针对可编程逻辑器件的某一产品系列进行的, 所以综合的过程要在相应厂商综合库的支持下才能完成。综合后, 可利用产生的网表文件进行适配前的时序仿真, 仿真过程不涉及具体器件的硬件特性。

5) 利用适配器将综合后的网表文件针对某一具体的目标器件进行逻辑映射操作, 包括底层器件配置、逻辑分割、逻辑优化和布局布线。适配完成后, 产生多项设计结果, 以过程文件的形式生成, 例如适配报告文件, 说明芯片内部资源利用情况和设计的布尔方程描述情况等, 还有适配后的仿真模型和器件编程文件等。根据适配后的仿真模型, 可以进行适配后的时序仿真, 因为已经得到器件的实际硬件特性 (如时延特性), 所以仿真结果能比较精确地预期未来芯片的实际性能。如果仿真结果达不到设计要求, 就需要修改 HDL 源代码或选择不同速度性能的器件, 直至满足设计要求。

6) 将适配器产生的编程文件通过编程器或下载电缆载入到目标芯片 FPGA 或 CPLD 中。对于大批量产品开发, 只要更换相应的厂商综合库, 就可以很容易地由 ASIC 的形式实现。

1.6 EDA 工程的设计流程

基于可编程逻辑器件的 EDA 工程设计典型的设计流程主要包括设计准备、设计输入、设计处理、器件编程和设计验证等 5 个基本步骤, 如图 1-5 所示。

1. 设计准备阶段

此阶段主要完成系统设计、设计方案论证和器件选择等内容。对于低密度 PLD, 可以进行书面逻辑设计, 将电路的逻辑功能直接用逻辑方程、真值表状态图或原理图等方式进行描述, 然后根据整个电路输入、输出端数以及所需要的资源 (门、触发器数目) 选择能满足设计要求的器件系列和型号。对于高密度

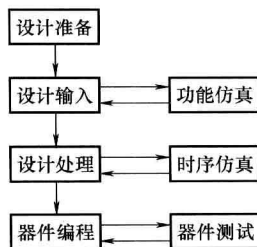


图 1-5 EDA 工程设计流程

PLD, 系统方案的选择通常采用自顶向下的设计方法。首先在顶层进行功能框图的划分和结构设计, 然后再逐级设计底层的结构。一般描述系统总功能的模块放在最上层, 称为顶层设计; 描述系统某一部分功能的模块放在下层, 称为底层设计。底层模块还可以再向下分层。系统方案的设计工作和器件的选择都可以在计算机上完成。选择器件时除了应考虑器件的引脚数、资源外, 还要考虑其速度、功耗以及结构特点, 通过对不同芯片进行平衡、比较, 确定最佳方案。

2. 设计输入阶段

设计输入就是设计者将所设计的系统或电路以开发软件要求的某种形式表示出来, 并送入计算机的过程。设计输入有多种方式, 常用的有原理图输入、硬件描述语言输入和波形输入等, 也可以采用文本、图形混合的输入方式。当目标系统不是很庞大时, 原理图输入是一种最直接的输入方式, 易读性强, 便于电路的调整, 容易实现仿真。所画的电路原理图与传统的器件连接方式基本相同, 编辑器中有许多现成的单元可以利用, 也可以自己根据需要设计元器件, 有助于提高工作效率。但随着设计规模增大, 原理图输入的设计易读性迅速下降, 很难搞清电路的实际功能, 改变电路结构十分困难, 移植性差, 文档管理和交流都很困难, 不利于团队合作, 因此不适于较大或较复杂的系统。硬件描述语言是用文本方式描述设计, 是 EDA 工程中最普遍使用的输入方式。它分为普通的硬件描述语言和行为描述语言。普通硬件描述语言有 ABEL-HDL、CUPL 等, 它们支持逻辑方程、真值表、状态机等逻辑表达方式; 行为描述语言是指高层硬件描述语言 VHDL 和 Verilog HDL, 有许多突出的优点, 如语言的公开可利用性, 便于组织大规模系统的设计, 具有很强的逻辑描述和仿真功能, 输入效率高, 可以非常方便地在不同的设计输入库之间转换, 可移植性好, 通用性好, 设计与芯片工艺及结构无关。波形输入法适合用于时序逻辑和有重复性的逻辑函数设计, 主要用于建立和编辑波形设计文件以及输入仿真向量和功能测试向量, EDA 工具软件可以根据用户定义的输入/输出波形自动生成逻辑关系。

3. 设计处理阶段

设计处理是 EDA 工程设计中的核心环节, 从设计输入完成以后到编程文件产生的整个逻辑综合、优化、布线和适配过程通常称为设计处理, 由计算机自动完成, 设计者只能通过设置参数来控制其处理过程。在编译过程中, 软件对设计输入文件进行逻辑化简和综合, 逻辑综合得到的网表文件通过适配器对具体的目标器件进行逻辑映射, 转换成实际的电路, 具体操作包括底层器件的配置、逻辑分割、逻辑优化和布线, 最后产生用于编程的下载文件。需要注意的是, HDL 描述的硬件系统要经过逻辑综合后最终转换成硬件电路, 如果纯粹以软件工程思想编写代码, 可能会造成某些语句不能综合成实际电路, 或形成的电路效率低下, 性能指标不佳等问题, 因此设计者需要对 EDA 工具的逻辑综合和优化过程有一定的了解。编程文件是可供器件编程下载使用的数据文件, 对于阵列型 PLD 来说, 编程文件是熔丝图文件, 即 JEDEC (简称 JED) 文件或 POF 格式文件, 对于 FPGA 来说, 编程文件是位流数据文件, 有 SOF、JAM、BIT 等格式的文件。有时为了提高电路的性能和效率, 可以采用第三方 EDA 软件进行逻辑综合, 如 Mentor Graphics 公司的 Precision RTL Plus 和 Synopsys 公司的 Synplify, 最后再用器件商提供的适配器进行适配。

4. 设计验证阶段

设计验证是包括功能仿真和时序仿真, 是对所设计电路功能的验证, 可以在设计输入和设计处理过程中同时进行。功能仿真是在设计输入完成以后的逻辑功能验证, 又称前仿真,