

**EDA** 工 | 程 | 技 | 术 | 丛 | 书 |



Xilinx大学合作计划推荐用书



XILINX FPGA : DEVELOPMENT AND APPLICATION (SECOND EDITION)

# Xilinx FPGA 开发实用教程 (第2版)

徐文波 田耘 编著  
Xu Wenbo Tian Yun



清华大学出版社



XILINX FPGA: DEVELOPMENT AND APPLICATION (SECOND EDITION)

# Xilinx FPGA 开发实用教程 (第2版)

本书第1版自2008年出版以来，屡次重印，获得了读者的诸多好评。近几年，FPGA技术发展突飞猛进，FPGA的设计理念不断发展，相关开发工具也持续更新；为满足广大读者的学习需求，作者倾力修订了本书。

第2版系统地讲述了Xilinx FPGA的开发技术，全书内容涵盖Xilinx器件概述、Verilog HDL开发基础与进阶、Xilinx FPGA电路原理与系统设计、基于ISE Foundation的逻辑设计、时序分析、逻辑开发专题、基于EDK的嵌入式系统设计、基于System Generator的DSP系统设计、数字信号处理专题以及SERDES技术专题。

本书融汇了两位作者的一线教学与工程开发经验，理论结构清晰完整，语言叙述深入浅出，应用示例极其丰富，非常适合于从事Xilinx系统FPGA开发的工程师、相关专业的研究生以及高年级本科生使用。

## 附赠光盘

附赠光盘中给出了书中所有实例的源码和相关的工程文件，软件版本为ISE Design Suite 13.2。

## 关于作者

**徐文波** 获得信息工程专业学士学位（北京邮电大学，2005年）、信号与信息处理专业博士学位（北京邮电大学，2010年），现执教于北京邮电大学信息与通信工程学院，长期从事信号处理理论及实际开发的教学与研究工作，发表多篇相关学术论文。

**田耘** 获得电子信息工程专业学士学位（北京邮电大学，2006年）、信号与信息处理专业硕士学位（北京邮电大学，2009年），一直从事信号处理领域中的FPGA技术开发，著有多部FPGA相关技术图书。



CD-ROM

清华大学出版社数字出版网站

**WQ Book** 书文局泉  
www.wqbook.com

上架指导：计算机/可编程序逻辑器件

ISBN 978-7-302-28643-1



9 787302 286431 &gt;

定价：54.50元

EDA 工 | 程 | 技 | 术 | 丛 | 书 |



XILINX FPGA: DEVELOPMENT AND APPLICATION (SECOND EDITION)

# Xilinx FPGA 开发实用教程

(第2版)

徐文波 田耘 编著  
Xu Wenbo Tian Yun

清华大学出版社  
北京

## 内 容 简 介

本书系统地论述了 Xilinx FPGA 开发方法、开发工具、实际案例及开发技巧, 内容涵盖 Xilinx 器件概述、Verilog HDL 开发基础与进阶、Xilinx FPGA 电路原理与系统设计、基于 ISE Foundation 的逻辑设计、时序分析、逻辑开发专题、基于 EDK 的嵌入式系统设计、基于 System Generator 的 DSP 系统设计、数字信号处理专题以及 SERDES 技术专题共 10 章。各章均以实战开发为目的, 结合最新版本的软硬件特征, 覆盖了 FPGA 的各主要应用领域。配套光盘中包含了书中所有的实例代码, 便于读者快速动手实践。书中融汇了作者多年的工程开发经验, 希望能够极力帮助读者提高工程开发能力。

本书适合作为电子信息工程、通信工程、自动化、计算机科学与技术等相关专业的高年级本科生及研究生的教学用书, 也可以作为从事 FPGA 设计工作的工程师的参考图书。

本书封面贴有清华大学出版社防伪标签, 无标签者不得销售。

版权所有, 侵权必究。侵权举报电话: 010-62782989 13701121933

### 图书在版编目(CIP)数据

Xilinx FPGA 开发实用教程 / 徐文波, 田耘编著. --2 版. --北京: 清华大学出版社, 2012. 7

EDA 工程技术丛书

ISBN 978-7-302-28643-1

I. ①X… II. ①徐… ②田… III. ①可编程逻辑器件—系统开发—教材 IV. ①TP332. 1

中国版本图书馆 CIP 数据核字(2012)第 074825 号

责任编辑: 盛东亮

封面设计: 李召霞

责任校对: 白 蕾

责任印制: 王静怡

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者: 北京密云胶印厂

经 销: 全国新华书店

开 本: 185mm×260mm 印 张: 32.25 字 数: 762 千字  
(附光盘 1 张)

版 次: 2008 年 11 月第 1 版 2012 年 7 月第 2 版 印 次: 2012 年 7 月第 1 次印刷

印 数: 9001~12000

定 价: 54.50 元

---

产品编号: 043592-01

# 序

赛灵思(Xilinx)公司作为可编程器件(PLD)的领导厂商,占有超过 50% 的市场份额,为客户提供可编程逻辑芯片(CPLD、FPGA 和 PROM)、软件设计工具、不同等级的知识产权核(IP Core)以及系统级的完整解决方案。

随着工艺和设计水平的不断提高,FPGA 在数字系统中所扮演的角色也从逻辑胶合者提升到处理核心。从 2006 年起,赛灵思公司的 FPGA 就涵盖了逻辑应用、数字信号处理以及嵌入式三大应用领域。到目前为止,赛灵思已成为完整的解决方案提供者。例如,量产的 Spartan 6 系列 FPGA 采用 45nm 工艺,广泛应用在中低规模系统中,如机器视觉、机顶盒以及广泛的多媒体处理等;量产的 Virtex-5/6 系列 FPGA 分别采用 65nm、40nm 工艺,主要面向高端应用,如高速互联网络、无线通信、宽带接入以及汽车工业等。28nm 的 7 系列 FPGA(Artix、Kintex 和 Virtex 三个低、中、高系列)也已逐渐量产,进一步提升 FPGA 系统设计能力。此外,Zynq-7000 可扩展处理平台(EPP)将 ARM Cortex-A9 双核处理器系统与可编程逻辑紧密集成在一起,为业界带来革命性的创新解决方案。

基于赛灵思公司的领先技术,更多的工程师和研究人员已加入到赛灵思 FPGA 的开发队伍中。在过去四年中,赛灵思公司通过大学和开源社区 OpenHard,开展了三届开源硬件创新大赛以及多个网络研讨会,但切入点都比较零散,不能形成系统化知识体系。《Xilinx FPGA 开发实用教程》第一版于 2008 年出版,弥补了上述不足,帮助广大技术人员、在校的研究生和高年级本科生尽快掌握 Xilinx FPGA 的开发流程,连续印刷 4 次,深受读者欢迎。由于 FPGA 技术发展迅速,因此作者更新了原书内容,并以最新的 ISE13.x 版本和量产的 6 系列器件为例进行介绍,更加符合 FPGA 发展趋势。

整体而言,本书具有以下三项特色:首先,从逻辑设计、数字信号处理、嵌入式系统设计和高速连接四个方面系统地介绍了 Xilinx FPGA 的开发与应用,条理清晰、思路明确,符合 FPGA 目前和未来的发展趋势;其次,较为详细地介绍了 Xilinx FPGA 的开发技巧,融入了作者的工程开发经验,对于初学者和工程开发人员来讲都具有较强的可读性;最后,极为全面地介绍了赛灵思公司的 ISE、System Generator 以及 EDK 开发软件,非常系统和完整。

因此,对于在校研究生、高年级本科生及从事 FPGA 开发的工程师来说,本书是一本较为理想的 EDA 教材和工程工具书,我郑重地将其推荐给大家!希望通过本书的出版,使更多的读者掌握赛灵思 FPGA 的开发技能,更好地促进 FPGA 开发技术的普及和推广。

赛灵思(Xilinx)公司中国区大学计划经理

谢凯年博士

2012 年 6 月

# 前言

2008年10月,作者有幸聆听了Xilinx公司全球CTO Ivo Bolsens先生在清华大学所作的题为“FPGA: The future platform for transforming, transporting and computing”的演讲,感触颇深。Ivo Bolsens先生指出了FPGA的三大应用领域:数字处理中的信号变换、高速交换中的数据收发以及求解中的复杂计算。作者带着感慨基于当时的ISE 9.1软件版本,编写了《Xilinx FPGA开发实用教程》一书,受到读者青睐,多次重印。经过3年多的发展,Xilinx公司的软、硬件均有大幅升级,ISE软件已升级到13.x版本,FPGA已经发展到28nm的“7”系列芯片。因此作者在第1版的基础上,结合ISE软件和FPGA特征,重新整理了原稿,删除了部分冗余、陈旧的内容,形成了此次的修订版版本。

同第1版一样,修订版版本中的全部内容都是作者实际项目开发经验和Xilinx公司各类文档、书籍的结合体,全部信息几乎都可以从Xilinx网站以及Google上找到渊源,不过我们仍然向您推荐本书,因为网络的信息是分散的、杂乱的,且正确性不是100%的,本书各章内容的安排是从大量的实践中总结出来的,循序渐进,条理清楚,且都经过作者验证,我们的目的就是从Ivo Bolsens先生的观点出发,结合项目开发,将网络上尽可能多的相关信息以相对较高的质量组合起来。

本书适合电子、通信以及计算机等相关专业的研究生和高年级本科生使用,同时也适合于从事Xilinx系列FPGA设计和开发的工程师。毫无疑问,市场上已经有很多关于FPGA设计的书籍,我们也不认为本书是其中最重要的一本,但我们意识到FPGA开发一定要结合芯片特点以及提供商的诸多建议和协议,只有这样才能真正掌握其开发之道。

在第1版中,由于未配备光盘,缺少实际的工程和电子版本代码,不利于快速学习,广大读者多次给作者指出这一不足。因此,在修订版本中,我们将全书所有内容都移植在小巧的S6 CARD板卡(基于Spartan 6 LX9的开发板,和身份证大小一样,通过USB供电和调试,无须下载线缆)上,并将所有的工程文件附在光盘上,为读者提供通用的验证平台。S6 CARD板卡的详细信息可参考与非网相关介绍(<http://www.openhw.org/shop/index.php?action=product&id=248>)。

全书各章由徐文波、田耘共同完成编写。此外,在成文过程中参考了较多的书籍、论文和网络文献,在此向广大作者表示深深谢意。

FPGA技术博大精深且发展迅猛,不可能通过一本书进行全方位的详细介绍,更多还需要读者自己动手实践。由于作者水平有限,FPGA技术发展迅速,书中难免存在不妥之处,敬请广大读者指正。

作 者

2012年5月

# 目录

<b>第1章 Xilinx器件概述</b>	<b>1</b>
1.1 可编程逻辑器件基础	1
1.1.1 可编程逻辑器件的基本情况	1
1.1.2 可编程逻辑器件的发展历史	2
1.1.3 PLD开发工具	3
1.1.4 典型FPGA开发流程	3
1.2 Xilinx FPGA芯片	6
1.2.1 FPGA的工作原理	6
1.2.2 Xilinx FPGA芯片结构	7
1.2.3 软核、硬核及固核	13
1.2.4 Xilinx主流FPGA	14
1.3 Xilinx软件工具	18
1.3.1 ISE Foundation软件	18
1.3.2 EDK开发工具	20
1.3.3 System Generator DSP工具	20
1.3.4 ChipScope Pro	20
1.3.5 PlanAhead	21
1.4 本书案例验证平台——S6 CARD开发板	22
1.4.1 S6 CARD开发板的组成与功能	22
1.4.2 S6 CARD板卡引脚约束说明	25
本章小结	26
<b>第2章 Verilog HDL开发基础与进阶</b>	<b>27</b>
2.1 Verilog HDL语言	27
2.1.1 Verilog HDL语言的历史	28
2.1.2 Verilog HDL的主要功能	28
2.1.3 Verilog HDL和VHDL的区别	29
2.1.4 Verilog HDL设计方法	29
2.2 Verilog HDL基本程序结构	30
2.3 Verilog HDL语言的数据类型和运算符	32
2.3.1 标志符	32
2.3.2 数据类型	32

# 目录

2.3.3 模块端口 .....	34
2.3.4 常量集合 .....	34
2.3.5 运算符和表达式 .....	36
2.4 Verilog HDL 语言的描述语句 .....	40
2.4.1 结构描述形式 .....	41
2.4.2 数据流描述形式 .....	42
2.4.3 行为描述形式 .....	42
2.4.4 混合设计模式 .....	50
2.5 Verilog HDL 建模与调试技巧 .....	50
2.5.1 双向端口的使用和仿真 .....	51
2.5.2 阻塞赋值与非阻塞赋值 .....	53
2.5.3 输入值不确定的组合逻辑电路 .....	55
2.5.4 数学运算中的扩位与截位操作 .....	56
2.5.5 利用块 RAM 来实现数据延迟 .....	57
2.5.6 测试向量的生成 .....	59
2.6 Verilog HDL 常用程序示例 .....	60
2.6.1 数字电路中基本单元的 FPGA 实现 .....	61
2.6.2 基本时序处理模块 .....	66
2.7 Xilinx 器件原语的使用 .....	71
本章小结 .....	74
 第 3 章 Xilinx FPGA 电路原理与系统设计 .....	75
3.1 FPGA 配置电路 .....	75
3.1.1 Xilinx FPGA 配置电路 .....	75
3.1.2 Xilinx FPGA 常用的配置引脚 .....	77
3.1.3 Xilinx FPGA 配置电路分类 .....	78
3.2 JTAG 电路的原理与设计 .....	80
3.2.1 JTAG 电路的工作原理 .....	80
3.2.2 Xilinx JTAG 下载线 .....	82
3.3 FPGA 的常用配置电路 .....	85
3.3.1 主串模式——最常用的 FPGA 配置模式 .....	86
3.3.2 SPI 串行 Flash 配置模式 .....	91
3.3.3 从串配置模式 .....	97

# 目录

3.3.4 主字节宽度并行配置模式 .....	98
3.3.5 JTAG 配置模式 .....	101
3.3.6 System ACE 配置方案 .....	102
3.4 iMPACT 软件使用 .....	107
3.4.1 iMPACT 软件 .....	107
3.4.2 iMPACT 中的 JTAG 配置操作 .....	109
3.4.3 iMPACT 中的 Xilinx PROM 配置操作 .....	112
3.4.4 iMPACT 中的 SPI Flash 配置操作 .....	114
3.4.5 FPGA 配置失败的常见问题 .....	117
3.5 从配置 PROM 中读取用户数据 .....	118
3.5.1 从 PROM 中引导数据 .....	118
3.5.2 硬件电路设计方法 .....	119
3.5.3 软件操作流程 .....	121
本章小结 .....	122
 第 4 章 基于 ISE Foundation 的逻辑设计 .....	123
4.1 ISE 套件 .....	123
4.1.1 ISE 的特点 .....	123
4.1.2 ISE 的功能 .....	124
4.1.3 ISE 的安装 .....	125
4.1.4 ISE 的用户界面 .....	125
4.2 基于 ISE 的设计输入 .....	126
4.2.1 新建工程 .....	126
4.2.2 代码输入 .....	128
4.2.3 代码模板的使用 .....	130
4.2.4 Xilinx IP Core 的原理与应用 .....	131
4.3 ISE 基本操作 .....	135
4.3.1 基于 Xilinx XST 的综合 .....	135
4.3.2 基于 ISim 的仿真 .....	137
4.3.3 基于 ISE 的实现 .....	140
4.3.4 基于目标和策略的设计方法 .....	143
4.3.5 基于 SmartGuide 的设计方法 .....	146
4.3.6 比特文件的生成 .....	149

# 目录

4.3.7 基于 IMPACT 的芯片配置 .....	154
4.3.8 功耗分析以及 XPower 的使用 .....	155
4.4 约束 .....	158
4.4.1 约束文件 .....	158
4.4.2 UCF 文件的语法说明 .....	159
4.4.3 引脚和区域约束语法 .....	161
4.4.4 时序约束语法 .....	163
4.5 调试利器——ChipScope Pro .....	167
4.5.1 ChipScope Pro 工作原理 .....	167
4.5.2 ChipScope Pro 操作流程 .....	169
4.5.3 ChipScope Pro 开发实例 .....	171
4.6 ISE 与第三方 EDA 软件 .....	179
4.6.1 ModelSim 软件的使用 .....	179
4.6.2 ModelSim 和 ISE 的联合开发流程 .....	183
4.6.3 MATLAB 软件的使用 .....	183
4.6.4 ISE 与 MATLAB 的联合使用 .....	184
4.6.5 MATLAB、ModelSim 和 ISE 联合开发实例 .....	186
本章小结 .....	194
 第 5 章 时序分析 .....	195
5.1 时序分析的作用和原理 .....	195
5.1.1 时序分析的作用 .....	195
5.1.2 静态时序分析原理 .....	196
5.1.3 时序分析的基础知识 .....	197
5.2 Xilinx FPGA 中的时钟资源 .....	203
5.2.1 全局时钟资源 .....	203
5.2.2 第二全局时钟资源 .....	206
5.3 ISE 时序分析器 .....	207
5.3.1 时序分析器的特点 .....	207
5.3.2 时序分析器的文件类型 .....	208
5.3.3 时序分析器的调用与用户界面 .....	208
5.3.4 提高时序性能的手段 .....	214
本章小结 .....	218

# 目录

<b>第 6 章 逻辑开发专题 .....</b>	<b>219</b>
<b>6.1 Verilog HDL 设计进阶 .....</b>	<b>219</b>
6.1.1 面向硬件的程序设计思维 .....	219
6.1.2 “面积”和“速度”的转换原则 .....	223
6.1.3 同步电路的设计原则 .....	224
<b>6.2 Xilinx FPGA 芯片底层单元的使用 .....</b>	<b>227</b>
6.2.1 Xilinx 全局时钟网络的使用 .....	227
6.2.2 CMT 时钟管理模块的使用 .....	228
6.2.3 Xilinx 内嵌块存储器的使用 .....	233
6.2.4 硬核乘加器的使用 .....	240
<b>6.3 代码风格 .....</b>	<b>242</b>
6.3.1 代码风格的含义 .....	242
6.3.2 代码书写风格 .....	242
6.3.3 通用设计代码风格 .....	246
6.3.4 Xilinx 专用设计代码风格 .....	255
<b>6.4 UART 接口开发实例 .....</b>	<b>259</b>
6.4.1 串口接口与 RS-232 协议 .....	259
6.4.2 串口通信控制器的 Verilog HDL 实现 .....	261
6.4.3 RS-232 设计板级调试 .....	274
<b>本章小结 .....</b>	<b>284</b>
<b>第 7 章 基于 EDK 的嵌入式系统设计 .....</b>	<b>285</b>
<b>7.1 可配置嵌入式系统(EDK) .....</b>	<b>285</b>
7.1.1 基于 FPGA 的可编程嵌入式开发系统 .....	285
7.1.2 Xilinx 公司的解决方案 .....	286
<b>7.2 Xilinx 嵌入式开发系统组成 .....</b>	<b>287</b>
7.2.1 片内微处理器软核 MicroBlaze .....	287
7.2.2 PLB 总线系统结构 .....	290
7.2.3 IP 核以及设备驱动 .....	295
<b>7.3 EDK 软件 .....</b>	<b>301</b>
7.3.1 EDK 设计的实现流程 .....	301
7.3.2 EDK 的文件管理架构 .....	303

# 目录

7.4 XPS 软件典型操作 .....	307
7.4.1 XPS 的启动 .....	308
7.4.2 利用 BSB 创建新工程 .....	308
7.4.3 XPS 的用户界面 .....	314
7.4.4 XPS 的目录结构与硬件平台 .....	318
7.4.5 在 XPS 加入 IP Core .....	319
7.4.6 XPS 工程的综合与实现 .....	322
7.5 SDK 软件典型操作 .....	325
7.5.1 SDK 的用户界面 .....	325
7.5.2 SDK 的典型操作 .....	327
7.5.3 IP 外设的 API 函数查阅和使用方法 .....	335
7.5.4 GPIO 外设开发实例 .....	337
7.5.5 其他外设开发实例 .....	340
本章小结 .....	340
 第 8 章 基于 System Generator 的 DSP 系统设计 .....	341
8.1 System Generator 的特点与安装 .....	341
8.1.1 System Generator 的主要特点 .....	342
8.1.2 System Generator 的安装和配置 .....	343
8.2 System Generator 的使用基础 .....	344
8.2.1 System Generator 开发流程 .....	344
8.2.2 Simulink 的应用 .....	346
8.3 基于 System Generator 的 DSP 系统设计 .....	348
8.3.1 System Generator 的应用 .....	348
8.3.2 System Generator 中的信号类型 .....	360
8.3.3 自动代码生成 .....	360
8.3.4 编译 MATLAB 设计生成 FPGA 代码 .....	362
8.3.5 子系统的建立与 ISE 调用 .....	364
8.4 基于 System Generator 的硬件协仿真 .....	368
8.4.1 硬件协仿真平台的特点与平台安装 .....	368
8.4.2 硬件协仿真的基本操作 .....	369
8.4.3 共享存储器的操作 .....	375
8.5 System Generator 的高级应用 .....	377

# 目录

8.5.1 导入外部的 HDL 程序模块 .....	377
8.5.2 设计在线调试 .....	383
8.5.3 系统中的多时钟设计 .....	386
8.5.4 FPGA 设计的高级技巧 .....	388
本章小结 .....	392
<b>第 9 章 数字信号处理专题 .....</b>	<b>393</b>
9.1 数字信号 .....	393
9.1.1 数字信号的产生 .....	393
9.1.2 采样定理 .....	394
9.1.3 数字系统的主要性能指标 .....	395
9.1.4 A/D 转换的字长效应 .....	395
9.2 常用 DSP IP Core 及其应用 .....	397
9.2.1 DDS 模块 IP Core 的应用 .....	397
9.2.2 FFT 算法 IP Core 的应用 .....	400
9.2.3 Cordic 算法 IP Core 的应用 .....	411
9.2.4 FIR 滤波器 IP Core 的应用 .....	414
9.3 多速率滤波器的 FPGA 实现 .....	428
9.3.1 多速率信号处理的意义 .....	428
9.3.2 多速率信号滤波器的基本操作 .....	428
9.3.3 CIC 滤波器的 FPGA 实现 .....	434
9.3.4 HB 滤波器的 FPGA 实现 .....	445
本章小结 .....	447
<b>第 10 章 SERDES 技术专题 .....</b>	<b>448</b>
10.1 高速数据连接功能 .....	448
10.1.1 高速数据传输 .....	448
10.1.2 Xilinx 公司高速连接功能的解决方案 .....	449
10.2 实现吉比特高速串行 I/O 的相关技术 .....	449
10.2.1 吉比特高速串行 I/O 的特点和应用 .....	449
10.2.2 吉比特串行 I/O 系统的组成 .....	451
10.2.3 吉比特串行 I/O 的设计要点 .....	455
10.3 Rocket I/O 收发器原理与开发 .....	457

# 目录

10.3.1 Rocket I/O 硬核组成与工作原理 .....	457
10.3.2 GTP 硬核组成与工作原理 .....	459
10.3.3 GTP Wizard 开发实例 .....	473
10.4 PCI-Express G1 端点接口设计 .....	488
10.4.1 PCI Express G1 技术 .....	488
10.4.2 Xilinx PCI Express G1 端点模块 .....	489
10.4.3 PCI Express G1 端点接口实例解读 .....	496
本章小结 .....	499
参考文献 .....	500

# 第1章 Xilinx 器件概述

FPGA(Field Programmable Gate Array)即现场可编程门阵列,属于可编程逻辑器件的一种,在20世纪90年代获得突飞猛进的发展,经过近30年的发展,到目前已成为实现数字系统的主流平台之一。本章主要介绍FPGA概念、芯片结构、工作原理、开发流程以及Xilinx公司的主要可编程芯片,为读者提供FPGA系统设计的基础知识。

## 1.1 可编程逻辑器件基础

### 1.1.1 可编程逻辑器件的基本情况

可编程逻辑器件(Programmable Logic Device,PLD)起源于20世纪70年代,是在专用集成电路(Application Specific Integrated Circuit,ASIC)的基础上发展起来的一种新型逻辑器件,是当今数字系统设计的主要硬件平台,其主要特点就是完全由用户通过软件进行配置和编程,从而完成某种特定的功能,且可以反复擦写。在修改和升级PLD时,不需额外地改变PCB电路板,只是在计算机上修改和更新程序,使硬件设计工作成为软件开发工作,缩短了系统设计的周期,提高了实现的灵活性并降低了成本,因此获得了广大硬件工程师的青睐,形成了巨大的PLD产业规模。

目前常见的PLD产品有:编程只读存储器(Programmable Read Only Memory,PROM)、现场可编程逻辑阵列(Field Programmable Logic Array,FPLA)、可编程阵列逻辑(Programmable Array Logic,PAL)、通用阵列逻辑(Generic Array Logic,GAL)、可擦除的可编程逻辑器件(Erasable Programmable Logic Array,EPLA)、复杂可编程逻辑器件(Complex Programmable Logic Device,CPLD)和现场可编程门阵列(Field Programmable Gate Array,FPGA)等类型。PLD器件从规模上又可以细分为简单PLD(SPLD)、复杂PLD(CPLD)以及FPGA。它们内部结构的实现方法各不相同。

可编程逻辑器件按照颗粒度可以分为3类:①小颗粒度(如“门海(sea of gates)”架构);②中等颗粒度(如FPGA);③大颗粒度(如

CPLD)。按照编程工艺可以分为 4 类：①熔丝(Fuse)和反熔丝(Antifuse)编程器件；②可擦除的可编程只读存储器(UEPROM)编程器件；③电信号可擦除的可编程只读存储器(EEPROM)编程器件(如 CPLD)；④SRAM 编程器件(如 FPGA)。在工艺分类中，前 3 类为非易失性器件，编程后，配置数据保留在器件上；第 4 类为易失性器件，掉电后配置数据会丢失，因此在每次上电后需要重新进行数据配置。

目前，主流 FPGA 器件均为 SRAM(Static RAM)工艺，包括最新一代的 28nm 工艺器件(Xilinx 公司的 7 系列芯片和 Altera 公司的 5 系列芯片)；只有深空宇航领域的 FPGA 大部分为反熔丝工艺(如 Xilinx 公司的 QPRO 系列)。本书内容以通用的 SRAM 工艺 FPGA 为主进行介绍。

在 PLD 技术高度发达的今天，不同类型、不同公司的 PLD 器件都获得了一定的发展，形成了独有特色(例如高性能、低功耗以及低成本等不同特点)，呈现出百花绽放的局面。

### 1.1.2 可编程逻辑器件的发展历史

可编程逻辑器件的发展可以划分为 4 个阶段，即从 20 世纪 70 年代初期到中期为第 1 阶段，20 世纪 70 年代中期到 80 年代中期为第 2 阶段，20 世纪 80 年代后期到 90 年代后期为第 3 阶段，20 世纪 90 年代后期到目前为第 4 阶段。

第 1 阶段的可编程器件只有简单的可编程只读存储器(PROM)、紫外线可擦除只读存储器(EPROM)和电可擦只读存储器(EEPROM)3 种，由于结构的限制，它们只能完成简单的数字逻辑功能。

第 2 阶段出现了结构上稍微复杂的可编程阵列逻辑(PAL)和通用阵列逻辑(GAL)器件，正式称为 PLD，能够完成各种逻辑运算功能。典型的 PLD 由“与”、“非”阵列组成，用“与或”表达式来实现任意组合逻辑，所以 PLD 能以乘积和形式完成大量的逻辑组合。

第 3 阶段 Xilinx 和 Altera 分别推出了与标准门阵列类似的 FPGA 以及类似于 PAL 结构的扩展性 CPLD，提高了逻辑运算的速度，具有体系结构和逻辑单元灵活、集成度高以及适用范围宽等特点，兼容了 PLD 和通用门阵列的优点，能够实现超大规模的电路，编程方式也很灵活，成为产品原型设计和中小规模(一般小于 10 000)产品生产的首选。这一阶段，CPLD、FPGA 器件的制造工艺和产品性能都获得了长足的发展，达到了 0.18 $\mu$ m 工艺和系数数百万门的规模。

第 4 阶段出现了 SoPC(System on Programmable Chip，可编程的片上系统)和 SoC(System on Chip，片上系统)技术，是 PLD 和 ASIC 技术融合的结果，涵盖了实时化数字信号处理技术、高速数据收发器、复杂计算以及嵌入式系统设计技术的全部内容。目前 Xilinx 和 Altera 最新的 FPGA 产品，制造工艺已达到 28nm，系统门数已接近千万门。而且，这一阶段的逻辑器件不仅内嵌了硬核高速乘法器、Gbits 差分串行接口、ARM A9 内核、PCI-E 硬核以及 SDRAM 控制器等硬核加速单元，还包括丰富的软核 IP Core 资源、MicroBlaze 和 NiosII，实现了软件需求和硬件设计的完美结合、高速与灵活性的完美结合，使其不仅超越了 ASIC 器件的性能和规模，也超越了传统意义上 FPGA 的概念，使 PLD 的应用范围从单片扩展到系统级。目前，基于 PLD 的片上可编程系统

(Programmable System on Chip, PSoC)仍在不断向前发展中。

### 1.1.3 PLD 开发工具

基于高复杂度 PLD 器件的开发,在很大程度上要依靠电子设计自动化(Electronic Design Automation, EDA)来完成。PLD 的 EDA 工具以计算机软件为主,将典型的单元电路封装起来构成固定模块并形成标准的硬件开发语言(如 HDL 语言)供设计人员使用。设计人员应考虑如何将可组装的软件库和软件包搭建出满足需求的功能模块甚至完整的系统。PLD 开发软件需要自动地完成逻辑编译、化简、分割、综合及优化、布局布线、仿真以及对于特定目标芯片的适配编译和编程下载等工作。典型的 EDA 工具中必须包含两个特殊的软件包,即综合器和适配器。综合器的功能就是针对给定的硬件系统组件,将设计者在 EDA 平台上完成的针对某个系统项目的 HDL、原理图或状态图形描述,进行编译、优化、转换和综合。

随着开发规模的级数性增长,就必须减短 PLD 开发软件的编译时间并提高其编译性能以及提供丰富的知识产权(IP)核资源供设计人员调用。此外,PLD 开发界面的友好性以及操作的复杂程度也是评价其性能的重要因素。目前,在 PLD 产业领域中,各个芯片提供商的 PLD 开发工具已成为影响其成败的核心成分。只有全面做到芯片技术领先、文档完整和 PLD 开发软件优秀,相应的芯片才能获得用户的认可。

由于 PLD 软件是用户开发 PLD 芯片的唯一手段,它和芯片、用户开发技巧一起构成决定设计性能的 3 大要素,因此 PLD 软件在开发中的重要性是不言而喻的。一个完美的 PLD 开发软件应当具备下面 5 点:

- (1) 准确地将用户设计转换为电路模块;
- (2) 能够高效地利用器件资源;
- (3) 能够快速地完成编译和综合;
- (4) 提供丰富的 IP 资源;
- (5) 用户界面友好、操作简单。

Xilinx 公司的 ISE、Altera 公司的 Quartus II 是业界公认的优秀的集成 PLD 开发软件。此外,仿真软件 ModelSim、新兴的设计与验证工具 MATLAB 等诸多第三方 EDK 开发软件也满足上述要求。

### 1.1.4 典型 FPGA 开发流程

FPGA 的设计流程就是利用 EDA 开发软件和编程工具对 FPGA 芯片进行开发的过程。FPGA 的开发流程一般如图 1-1 所示,包括电路设计、设计输入、功能仿真、综合优化、综合后仿真、实现、布线后仿真、板级仿真以及芯片编程与调试等主要步骤。

#### 1. 电路功能设计

在系统设计之前,首先要进行方案论证、系统设计和 FPGA 芯片选择等准备工作。系统工程师根据任务要求,如系统的指标和复杂度,对工作速度和芯片本身的各种资源、

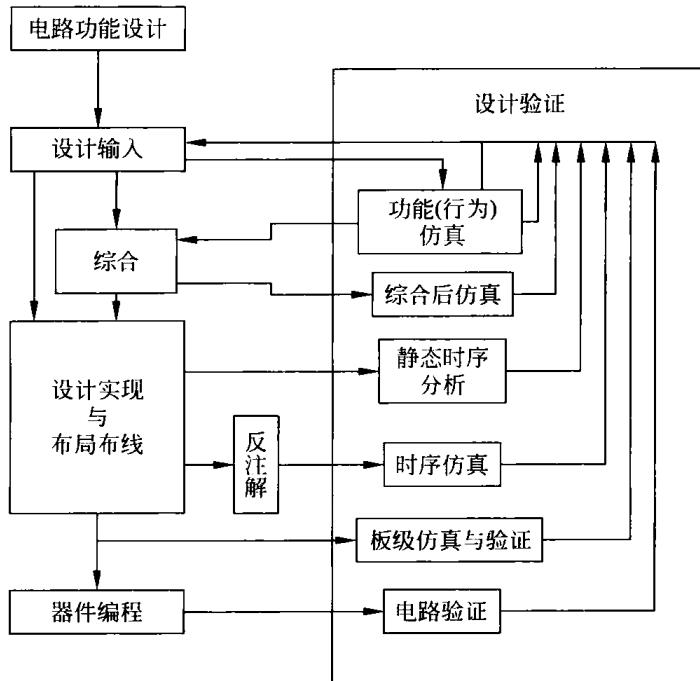


图 1-1 FPGA 开发的一般流程

成本等方面进行权衡,选择合理的设计方案和合适的器件类型。一般都采用自顶向下的设计方法,把系统分成若干个基本单元,然后再把每个基本单元划分为下一层次的基本单元,一直这样做下去,直到可以直接使用 EDA 元件库为止。

## 2. 设计输入

设计输入是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,并输入给 EDA 工具的过程。常用的方法有硬件描述语言(HDL)和原理图输入方法等。原理图输入方式是一种最直接的描述方式,在可编程芯片发展的早期应用比较广泛,它将所需的器件从元件库中调出来,画出原理图。这种方法虽然直观并易于仿真,但效率很低,且不易维护,不利于模块构造和重用。更主要的缺点是可移植性差,当芯片升级后,所有的原理图都需要作一定的改动。目前,在实际开发中应用最广的就是 HDL 语言输入法,利用文本描述设计,可以分为普通 HDL 和行为 HDL。普通 HDL 有 ABEL、CUR 等,支持逻辑方程、真值表和状态机等表达方式,主要用于简单的小型设计。而在大中型工程中,主要使用行为 HDL,其主流语言是 Verilog HDL 和 VHDL。这两种语言都是美国电气与电子工程师协会(IEEE)的标准,其共同的突出特点有:语言与芯片工艺无关,利于自顶向下设计,便于模块的划分与移植,可移植性好,具有很强的逻辑描述和仿真功能,而且输入效率很高。本书第 2 章将会介绍 Verilog HDL 设计的基本方法。

## 3. 功能仿真

功能仿真,也称为前仿真,是在编译之前对用户所设计的电路进行逻辑功能验证,此时的仿真没有延迟信息,仅对初步的功能进行检测。仿真前,要先利用波形编辑器和

HDL 等建立波形文件和测试向量(即将所关心的输入信号组合成序列),仿真结果将会生成报告文件和输出信号波形,从中便可以观察各个节点信号的变化。如果发现错误,则返回设计修改逻辑设计。常用的工具有 Model Tech 公司的 ModelSim、Synopsys 公司的 VCS 和 Cadence 公司的 NC-Verilog 以及 NC-VHDL 等软件。功能仿真不仅是 FPGA 开发过程中的必需步骤,也是系统设计中最关键的一步。

#### 4. 综合

所谓综合,就是将较高级抽象层次的描述转化成较低层次的描述。综合优化根据目标与要求优化所生成的逻辑连接,使层次设计平面化,供 FPGA 布局布线软件进行实现。就目前的层次来看,综合优化(Synthesis)是指将设计输入编译成由与门、或门、非门、RAM、触发器等基本逻辑单元组成的逻辑连接网表,而并非真实的门级电路。真实具体的门级电路需要利用 FPGA 制造商的布局布线功能,根据综合后生成的标准门级结构网表来产生。为了能转换成标准的门级结构网表,HDL 程序的编写必须符合特定综合器所要求的风格。由于门级结构、RTL 级的 HDL 程序的综合是很成熟的技术,所有的综合器都可以支持到这一级别的综合。常用的综合工具有 Synplicity 公司的 Synplify/Synplify Pro 软件以及各个 FPGA 厂家自己推出的综合开发工具。

#### 5. 综合后仿真

综合后仿真检查综合结果是否和原设计一致。在仿真时,把综合生成的标准延时文件反标注到综合仿真模型中去,可估计门延时带来的影响。但这一步骤不能估计线延时,因此和布线后的实际情况还有一定的差距,并不十分准确。目前的综合工具较为成熟,对于一般的设计可以省略这一步,但如果在布局布线后发现电路结构和设计意图不符,则需要回溯到综合后仿真来确认问题所在。在功能仿真中介绍的软件工具一般都支持综合后仿真。

#### 6. 实现与布局布线

实现是将综合生成的逻辑网表配置到具体的 FPGA 芯片上,布局布线是其中最重要的过程。布局将逻辑网表中的硬件原语和底层单元合理地配置到芯片内部的固有硬件结构上,并且往往需要在速度最优和面积最优之间作出选择。布线根据布局的拓扑结构,利用芯片内部的各种连线资源,合理正确地连接各个元件。目前,FPGA 的结构非常复杂,特别是在有时序约束条件时,需要利用时序驱动的引擎进行布局布线。布线结束后,软件工具会自动生成报告,提供有关设计中各部分资源的使用情况。由于只有 FPGA 芯片生产商对芯片结构最为了解,所以布局布线必须选择芯片开发商提供的工具。

#### 7. 时序仿真与验证

时序仿真,也称为后仿真,是指将布局布线的延时信息反标注到设计网表中来检测有无时序违规(即不满足时序约束条件或器件固有的时序规则,如建立时间、保持时间等)现象。时序仿真包含的延迟信息最全,也最精确,能较好地反映芯片的实际工作情况。由于不同芯片的内部延时不一样,不同的布局布线方案也给延时带来不同的影响。

因此在布局布线后,通过对系统和各个模块进行时序仿真,分析其时序关系,估计系统性能,以及检查和消除竞争冒险是非常有必要的。在功能仿真中介绍的软件工具一般都支持综合后仿真。

## 8. 板级仿真与验证

板级仿真主要应用于高速电路设计中,对高速系统的信号完整性、电磁干扰等特征进行分析,一般以第三方工具进行仿真和验证。

## 9. 芯片编程与调试

设计的最后一步就是芯片编程与调试。芯片编程是指产生使用的数据文件(位数据流文件,Bitstream File),然后将编程数据下载到 FPGA 芯片中。其中,芯片编程需要满足一定的条件,如编程电压、编程时序和编程算法等方面。逻辑分析仪(Logic Analyzer, LA)是 FPGA 设计的主要调试工具,但需要引出大量的测试引脚,且 LA 价格昂贵。目前,主流的 FPGA 芯片生产商都提供了内嵌的在线逻辑分析仪(如 Xilinx 的 ChipScope、Altera 的 SignalTapII 以及 SignalProbe)来解决上述矛盾,它们只需要占用芯片少量的逻辑资源,因此具有很高的实用价值。

## 1.2 Xilinx FPGA 芯片

### 1.2.1 FPGA 的工作原理

如前所述,FPGA 是在 PAL、GAL、EPLD、CPLD 等可编程器件的基础上进一步发展的产物。它是作为 ASIC 领域中的一种半定制电路而出现的,既解决了定制电路的不足,又克服了原有可编程器件门电路有限的缺点。

由于 FPGA 需要被反复烧写,它实现组合逻辑的基本结构不可能像 ASIC 那样通过固定的与非门来完成,而只能采用一种易于反复配置的结构,查找表可以很好地满足这一要求。目前,主流 FPGA 都采用了基于 SRAM 工艺的查找表结构,也有一些军品和宇航级 FPGA 采用 Flash 或者熔丝与反熔丝工艺的查找表结构。通过烧写文件改变查找表内容的方法来实现对 FPGA 的重复配置。

根据数字电路的基本知识可以知道,对于一个  $n$  输入的逻辑运算,不管是与或非运算还是异或运算等,最多只可能存在  $2^n$  种结果。所以如果事先将相应的结果存放于一个存储单元,就相当于实现了与非门电路的功能。FPGA 的原理也是如此,它通过烧写文件去配置查找表的内容,从而在相同的电路情况下实现了不同的逻辑功能。

查找表(Look-Up-Table)简称为 LUT,LUT 本质上就是一个 RAM。目前主流的 FPGA 中多使用 4 或 6 输入的 LUT,所以每一个 LUT 可以看成一个有 4(或 6)位地址线的  $16 \times 1$ (或  $64 \times 1$ )的 RAM。当用户通过原理图或 HDL 语言描述了一个逻辑电路以后,PLD/FPGA 开发软件会自动计算逻辑电路的所有可能结果,并把真值表(即结果)事先写入 RAM,这样,每输入一个信号进行逻辑运算就等于输入一个地址进行查表,找出地址对应的内容,然后输出即可。

下面给出一个4与门电路的例子来说明LUT实现逻辑功能的原理。

**例1-1** 给出一个使用LUT实现4输入与门电路的真值表,如表1-1所示。

表1-1 4输入与门的真值表

实际逻辑电路		LUT的实现方式	
a, b, c, d 输入	逻辑输出	RAM地址	RAM中存储的内容
0 0 0 0	0	0 0 0 0	0
0 0 0 1	0	0 0 0 1	0
...	...	...	...
1 1 1 1	1	1 1 1 1	1

可以看出,LUT具有和逻辑电路相同的功能。实际上,LUT具有更快的执行速度和更大的规模。

由于基于LUT的FPGA具有很高的集成度,其器件密度从数万门到数千万门不等,可以完成极其复杂的时序逻辑电路与组合逻辑电路,所以适用于高速、高密度的高端数字逻辑电路设计领域。其组成部分主要有可编程输入/输出单元、基本可编程逻辑单元、内嵌SRAM、丰富的布线资源、底层嵌入功能单元、内嵌专用单元等,主要设计和生产厂家有Xilinx、Altera、Lattice等厂商。

如前所述,FPGA是由存放在片内的RAM来设置其工作状态的,因此工作时需要对片内RAM进行编程。用户可根据不同的配置模式,采用不同的编程方式。FPGA有如下几种配置模式:

- (1) 并行模式:并行PROM、Flash配置FPGA。
- (2) 主从模式:一片PROM配置多片FPGA。
- (3) 串行模式:串行PROM配置FPGA。
- (4) 外设模式:将FPGA作为微处理器的外设,由微处理器对其编程。

目前,FPGA市场占有率最高的两大公司Xilinx和Altera生产的FPGA都是基于SRAM工艺的,需要在使用时外接一个片外存储器以保存程序。上电时,FPGA将外部存储器中的数据读入片内RAM,完成配置后,进入工作状态;掉电后FPGA恢复为白片,内部逻辑消失。这样FPGA不仅能反复使用,还无须专门的FPGA编程器,只需通用的EPROM、PROM编程器即可。Xilinx、Actel公司还提供反熔丝技术的FPGA,只能下载一次,具有抗辐射、耐高低温、低功耗和速度快等优点,在军品和航空航天领域中应用较多,但这种FPGA不能重复擦写,开发初期比较麻烦,费用也比较昂贵。Lattice是ISP(In-System Programmability,在系统可编程)技术的发明者,在低成本、低功耗PLD应用上有一定的特色。早期的Xilinx产品一般不涉及军品和宇航级市场,但目前已经有Q Pro-R等多款产品进入该类领域。

## 1.2.2 Xilinx FPGA芯片结构

目前,FPGA芯片仍是基于查找表技术的,但其概念和性能已经远远超出查找表技术的限制,并且整合了常用功能的硬核模块(如块RAM、时钟管理和DSP)。图1-2所示

为 Xilinx 公司 Spartan-2 系列 FPGA 的内部结构图,从中可以看出 FPGA 芯片主要由 6 部分组成:可编程输入输出单元、基本可编程逻辑单元、完整的时钟管理、嵌入块式 RAM、丰富的布线资源、内嵌的底层功能单元和内嵌专用硬件模块(注: Virtex 5 后续的 Virtex 系列、Spartan 6 后续的 Spartan 系列都为 6 输入的 LUT,之前的 Spartan 2/3、Virtex 2/4 系列均为 4 输入 LUT)。

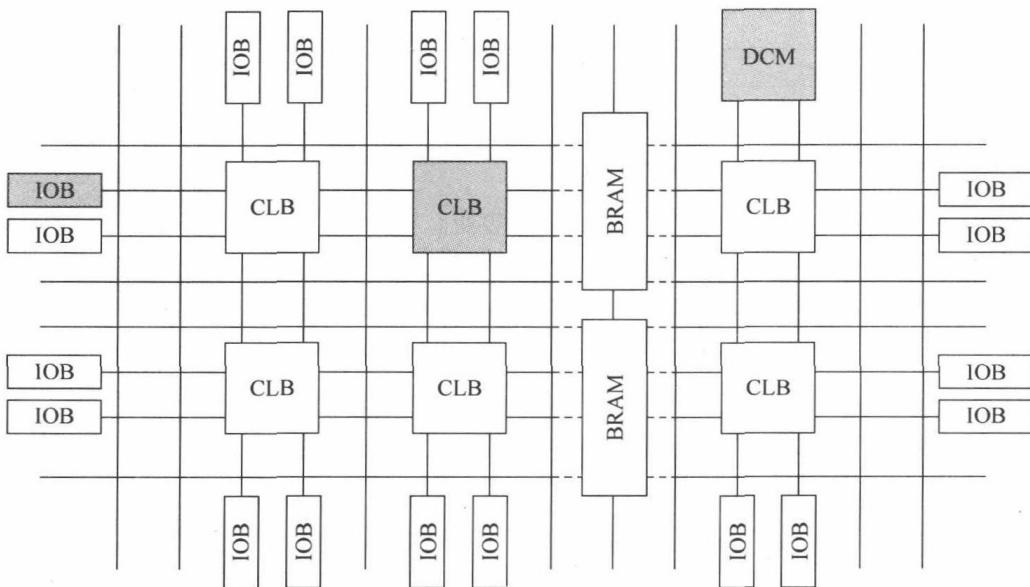


图 1-2 FPGA 芯片的内部结构

注:图 1-2 只是一个示意图,实际上每一个系列的 FPGA 都有其相应的内部结构,由于应用场合不同,所以内部结构会有局部不同

每个模块的功能如下。

### 1. 可编程输入输出单元( IOB )

可编程输入/输出单元简称 I/O 单元,是芯片与外界电路的接口部分,完成不同电气特性下对输入/输出信号的驱动与匹配要求,其示意结构如图 1-3 所示。为了便于管理和适应多种电气标准,FPGA 的 IOB 被划分为若干个组(bank),每个 bank 的接口标准由其接口电压  $V_{cco}$  决定,一个 bank 只能有一种  $V_{cco}$ ,但不同 bank 的  $V_{cco}$  可以不同。只有相同电气标准的端口才能连接在一起,  $V_{cco}$  电压相同是接口标准的基本条件。

FPGA 内的 I/O 按组分类,每组都能够独立地支持不同的 I/O 标准。通过软件的灵活配置,可适配不同的电气标准与 I/O 物理特性,可以调整驱动电流的大小,可以改变上拉电阻与下拉电阻。目前,I/O 口的频率也越来越高,一些高端的 FPGA 通过 DDR(或 DDR2)寄存器技术可以支持高达 2Gb/s 的数据速率。

外部输入信号可以通过 IOB 模块的存储单元输入到 FPGA 的内部,也可以直接输入 FPGA 内部。当外部输入信号经过 IOB 模块的存储单元输入到 FPGA 内部时,其保持时间(hold time)的要求可以降低,通常默认为 0。

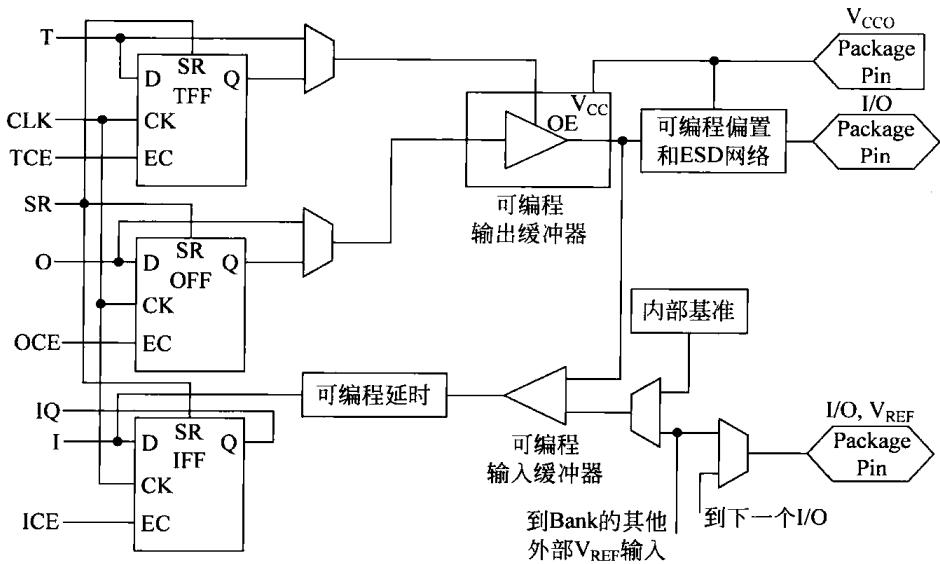


图 1-3 典型的 IOB 内部结构示意图

## 2. 可配置逻辑块(CLB)

CLB 是 FPGA 内的基本逻辑单元。CLB 的实际数量和特性会依器件的不同而不同，但是每个 CLB 都包含一个可配置开关矩阵，此矩阵由 4 或 6 个输入、一些选型电路（多路复用器等）和触发器组成。开关矩阵是高度灵活的，可以对其进行配置以便处理组合逻辑、移位寄存器或 RAM。在 Xilinx 公司的 FPGA 器件中，CLB 由多个（一般为 4 或 2 个）相同的 Slice 和附加逻辑构成，如图 1-4 所示。每个 CLB 模块不仅可以用于实现组合逻辑、时序逻辑，还可以配置为分布式 RAM 和分布式 ROM。

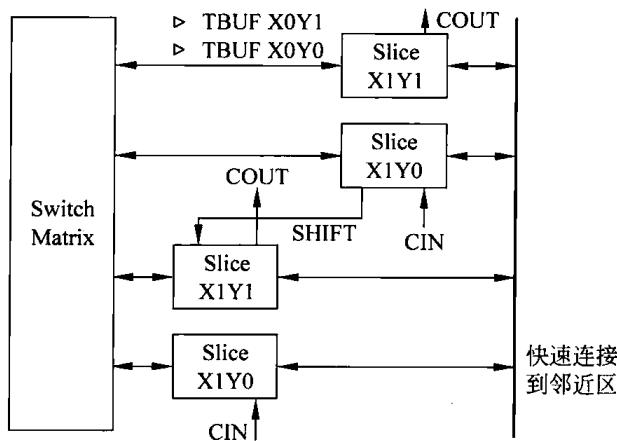


图 1-4 典型的 CLB 结构示意图

Slice 是 Xilinx 公司定义的基本逻辑单位。基于 4 输入 LUT 的传统 Slice 内部结构如图 1-5 所示，一个 Slice 由两个 4/6 输入的查找表函数、进位逻辑、算术逻辑、存储逻辑和函数复用器组成。算术逻辑包括一个异或门(XORG)和一个专用与门(MULTAND)，

一个异或门可以使一个 Slice 实现 2bit 全加操作,专用与门用于提高乘法器的效率; 进位逻辑由专用进位信号和函数复用器(MUXC)组成,用于实现快速的算术加减法操作; 4 输入函数发生器用于实现 4 输入 LUT、分布式 RAM 或 16 比特移位寄存器(目前,基于 65nm 工艺的 FPGA 一般都采用 6 输入查找表,可以实现 6 输入 LUT 或 64 比特移位寄存器); 进位逻辑包括两条快速进位链,用于提高 CLB 模块的处理速度。

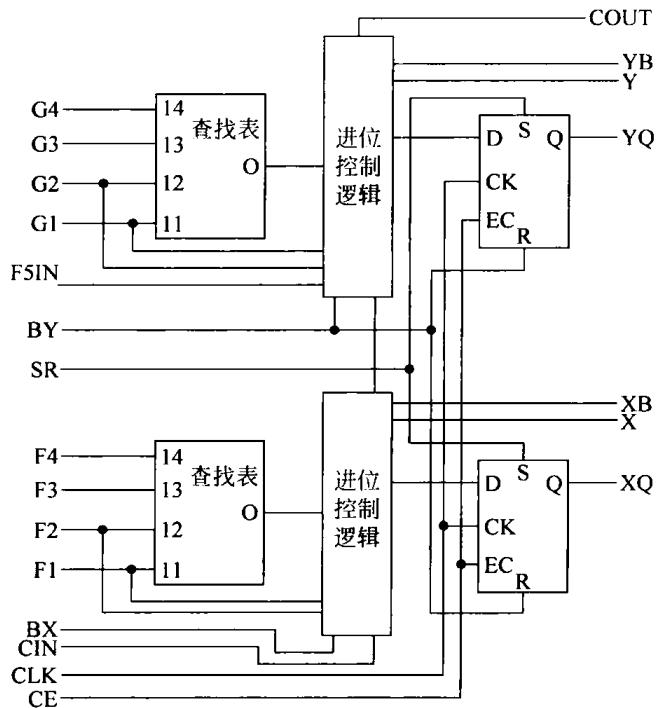


图 1-5 典型的 4 输入 Slice 结构示意图

基于 6 输入 LUT 的传统 Slice 内部结构如图 1-6 所示,其包括 4 个 6 输入的 LUT 和 8 个寄存器,因此新一代的 6 输入 Slice,其逻辑能力从资源上讲是“老”器件的 4 倍(因为 6 输入 LUT 的逻辑能力是  $2^6$ ,4 输入 LUT 的逻辑能力是  $2^4$ )。

### 3. 时钟管理模块

业内大多数 FPGA 均提供数字时钟管理(Xilinx 的全部 FPGA 均具有这种特性)。Xilinx 推出最先进的 FPGA 提供数字时钟管理和模拟相位环路锁定。相位环路锁定能够提供精确的时钟综合,且能够降低抖动,并实现过滤功能。

### 4. 嵌入式块 RAM(BRAM)

大多数 FPGA 都具有内嵌的块 RAM,这大大拓展了 FPGA 的应用范围和灵活性。块 RAM 可被配置为单端口 RAM、双端口 RAM、内容地址存储器(CAM)以及 FIFO 等常用存储结构。RAM、FIFO 是比较常见的概念,此处不再赘述。CAM 存储器在其内部的每个存储单元中都有一个比较逻辑,写入 CAM 中的数据会和内部的每一个数据进行比较,并返回与端口数据相同的所有数据的地址,因而在路由的地址交换器中有广泛的应用。除了块 RAM,还可以将 FPGA 中的 LUT 灵活地配置成 RAM、ROM 和 FIFO 等结构。在实际应用中,芯片内部块 RAM 的数量也是选择芯片的一个重要因素。

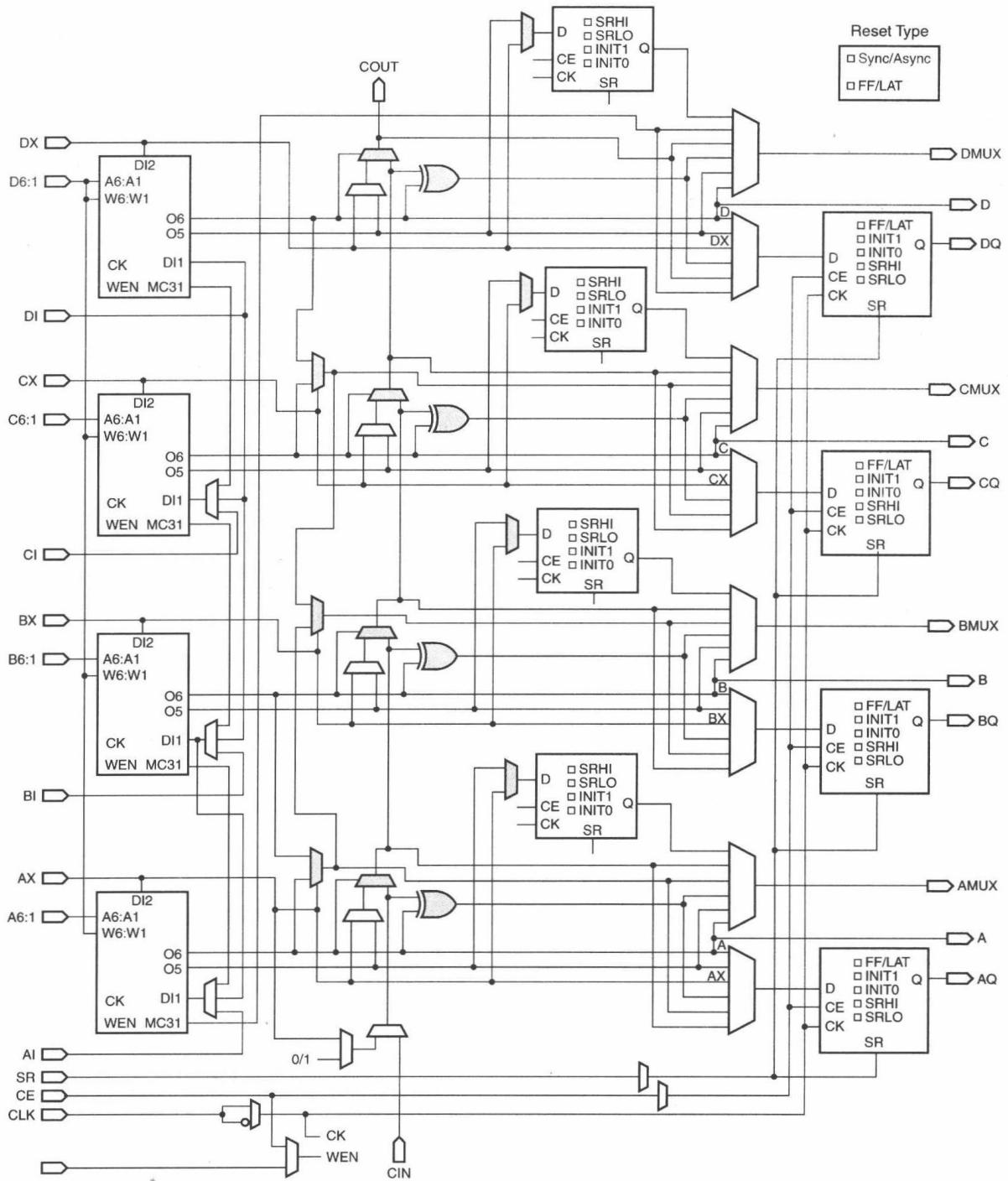


图 1-6 基于 6 输入 LUT 的 Slice 结构图

目前,6 输入 LUT 器件中的单片块 RAM 的容量为 36K 比特,即位宽为 36 比特、深度为 1024,可以根据需要改变其位宽和深度,但要满足两个原则:首先,修改后的容量(位宽×深度)不能大于 36K 比特;其次,位宽最大不能超过 72 比特。当然,可以将多片块 RAM 级联起来形成更大的 RAM,此时只受限于芯片内块 RAM 的数量,而不再受上面两条原则约束。

4 输入 LUT 器件中的单片块 RAM 的容量为 18K 比特,即位宽为 18 比特、深度为 1024,在实际应用中也需要后面两个原则:首先,修改后的容量(位宽×深度)不能大于 18K 比特;其次,位宽最大不能超过 36 比特。

## 5. 丰富的布线资源

布线资源连通 FPGA 内部的所有单元,而连线的长度和工艺决定着信号在连线上的驱动能力和传输速度。Xilinx FPGA 芯片内部有着丰富的布线资源,根据工艺、长度、宽度和分布位置的不同而划分为 4 种不同的类别。第一类是全局布线资源,用于芯片内部全局时钟和全局复位/置位的布线;第二类是长线资源,用以完成芯片 Bank 间的高速信号和第二全局时钟信号的布线;第三类是短线资源,用于完成基本逻辑单元之间的逻辑互连和布线;第四类是分布式的布线资源,用于专有时钟、复位等控制信号线。

低成本 FPGA 和高性能 FPGA 最大的区别就在于布线资源数量的不同,高端 FPGA 总具有最好的布线资源。例如,Virtex 6 XC6VLX75T 和 Spartan 6 XC6SLX75T 器件相比,虽然二者逻辑资源 Slice 数目一样,但 XC6VLX75T 具有更多的布线资源。在复杂设计中,XC6SLX75T 往往要利用 LUT 来完成逻辑布线,从而使得设计速度降低并且占用更多的逻辑资源。因此,在高速设计中推荐使用 Virtex 或者 Kintex 系列;在低速设计中推荐使用 Spartan 和 Artix 系列。

从本质上讲,布线资源的使用方法和设计的结果有密切、直接的关系。但在实际中面对一个固定的芯片,设计者不需要直接选择布线资源,布局布线器可自动地根据输入逻辑网表的拓扑结构和约束条件选择布线资源来连通各个模块单元。

## 6. 底层内嵌功能单元

内嵌功能模块主要指 DLL(Delay Locked Loop)、PLL(Phase Locked Loop)、DSP 和 CPU 等软处理核(Embedded Processor)。正是由于集成了丰富的内嵌功能单元,从而使得单片 FPGA 成了系统级的设计工具,具备了软硬件联合设计的能力,逐步向 SoC 平台过渡。

DLL 和 PLL 具有类似的功能,可以完成时钟高精度、低抖动的倍频和分频,以及占空比调整和移相等功能。Xilinx 老一代 4 输入 LUT FPGA 芯片上只集成了 DLL,6 输入 LUT FPGA 芯片上同时集成了 PLL 和 DLL。PLL 和 DLL 可以通过 IP 核生成的工具方便地进行管理和配置。典型的 DLL 结构如图 1-7 所示。

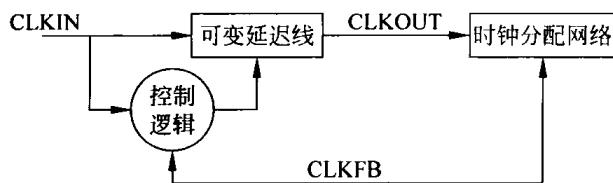


图 1-7 典型的 DLL 模块示意图

## 7. 内嵌专用硬核

内嵌专用硬核是相对底层嵌入的软核而言的,指 FPGA 处理能力强大的硬核(Hard Core),等效于 ASIC 电路。为了提高 FPGA 性能,芯片生产商在芯片内部集成了一些专用的硬核。例如,为了提高 FPGA 的乘法速度,主流的 FPGA 中都集成了专用乘法器;为了适用通信总线与接口标准,很多高端的 FPGA 内部都集成了串并收发器(SERDES),可以达到数十吉比特/秒的收发速度。

例如,Xilinx 公司的高端产品不仅集成了 Power PC 或者 ARM A9 系列高性能 CPU,还内嵌了 DSP Core 模块,其相应的系统级设计工具是 EDK 和 Platform Studio,并以此提出了片上系统 SoC 的概念。通过 Power PC、Miroblaze、Picoblaze 等处理器平台,能够开发标准的 DSP 处理器及其相关应用,达到 SoC 的开发目的。

### 1.2.3 软核、硬核及固核

IP(Intelligent Property)核是具有知识产权核的集成电路芯核总称,是经过反复验证过的、具有特定功能的宏模块,与芯片制造工艺无关,可以移植到不同的半导体工艺中。到了 SoC 阶段,IP 核设计已成为 ASIC 电路设计公司和 FPGA 提供商的重要任务,也是其实力体现。对于 FPGA 开发软件,其提供的 IP 核越丰富,用户的设计就越方便,其市场占有率就越高。目前,IP 核已经变成系统设计的基本单元,并作为独立设计成果被交换、转让和销售。

从 IP 核的提供方式上,通常将其分为软核、硬核和固核三类。从完成 IP 核所花费的成本来讲,硬核代价最大;从使用灵活性来讲,软核的可复用性最高。在 FPGA 领域中,最常用的是软核和硬核这两类 IP。

#### 1. 软核

软核在 EDA 设计领域指的是综合之前的寄存器传输级(RTL)模型;具体在 FPGA 设计中指的是对电路的硬件语言描述,包括逻辑描述、网表和帮助文档等。软核只经过功能仿真,需要经过综合以及布局布线才能使用。其优点是灵活性高、可移植性强,允许用户自配置;缺点是对模块的预测性较低,在后续设计中存在发生错误的可能性,有一定的设计风险。软核是 IP 核应用最广泛的形式。

Xilinx 公司一直以来提供的软核 IP 资源比其他厂家都更为丰富,全部集成在开发组件 Core Generator 中,本书将在 4.2.4 节对其进行详细说明。

#### 2. 固核

固核在 EDA 设计领域指的是带有平面规划信息的网表;具体在 FPGA 设计中可以看做带有布局规划的软核,通常以 RTL 代码和对应具体工艺网表的混合形式提供。将 RTL 描述结合具体标准单元库进行综合优化设计,形成门级网表,再通过布局布线工具即可使用。和软核相比,固核的设计灵活性稍差,但在可靠性上有较大提高。目前,固核也是 IP 核的主流形式之一。

### 3. 硬核

硬核在 EDA 设计领域指经过验证的设计版图；在 FPGA 设计中具体指布局和工艺固定、经过前端和后端验证的设计，设计人员不能对其进行修改。不能修改的原因有两个：首先是系统设计对各个模块的时序要求很严格，不允许打乱已有的物理版图；其次是保护知识产权的要求，不允许设计人员对其进行任何改动。硬核的优势也很明显：首先，不占用 FPGA 内部的任何逻辑资源和存储单元；其次，运行速度可以达到芯片标称值，且不影响其周边逻辑的布局布线。

目前，所有的 Xilinx FPGA 内部都继承了 CDM（时钟管理单元）、DSP 单元以及 BRAM 等硬核资源，高端芯片中还包括 SERDES、PCIE、SDRAM 控制器、ARM/Power PC 以及 ADC 等硬核模块。

## 1.2.4 Xilinx 主流 FPGA

世界上第一款 FPGA 芯片由 Xilinx 公司于 1984 年推出。之后，从 XC330、XC4000 到 Spartan-6 系列、Virtex-6 和最新的 Artix-7、Kintex-7 以及 Virtex-7 系列，Xilinx 公司一直保持着 FPGA 领域的全球领先地位。目前，XC 3000、XC 4000、Spartan、Spartan-XL、Viterx、Virtex-E、Spartan-2 和 Viterx-2 系列已经基本被淘汰或者正逐渐退出市场，Spartan-6 和 Viterx-6 以后的 FPGA 产品为其目前的主流产品。

### 1. Spartan 类

Spartan 系列适用于普通的工业、商业等领域，目前主流的芯片包括 Spartan-3A、Spartan-3A DSP 以及 Spartan-6 等种类。Spartan-6 系列芯片不仅在逻辑容量方面有较大提升，还增强了大量的内嵌专用乘法器和专用块 RAM 资源，具备实现复杂数字信号处理和片上可编程系统的能力。

#### 1) Spartan-3A 和 Spartan-3A DSP 系列

Spartan-3A 在 Spartan-3 和 Spartan-3E 平台的基础上，整合了各种创新特性帮助客户极大地削减了系统总成本。利用独特的器件 DNA ID 技术，实现业内首款 FPGA 电子序列号；提供了经济、功能强大的机制来防止发生篡改、克隆和过度设计的现象；并且具有集成式看门狗监控功能的增强型多重启动特性。支持商用 Flash 存储器，有助于削减系统总成本。其主要特性为：

- (1) 采用 90nm 工艺，密度高达 74 880 个逻辑单元。
- (2) 工作时钟范围为 5~320MHz。
- (3) 领先的连接功能平台，具有最广泛的 I/O 标准（26 种，包括新的 TMDS 和 PPDS）支持。
- (4) 利用独特的 Device DNA 序列号实现业内首个功能强大的防克隆安全特性。
- (5) 五个器件，具有高达 1.4M 的系统门和 502 个 I/O。
- (6) 灵活的功耗管理。

Spartan-3A 系列产品的主要技术特征如表 1-2 所示。

表 1-2 Spartan-3A 系列 FPGA 主要技术特征

型 号	系 统 门数	Slice 数 目	分 布 式 RAM 容 量(Kb)	块 RAM 容 量(Kb)	专 用 乘 法 器 数	DCM 数 目	最 大 可 用 I/O 数	最 大 差 分 I/O 对 数
XC3S50A	50k	864	11	54	3	2	144	64
XC3S200A	200k	2016	28	288	16	4	248	112
XC3S400A	400k	4032	56	360	20	4	311	142
XC3S700A	700k	6624	92	360	20	8	372	165
XC3S1400A	1400k	12 672	176	576	32	8	502	227

Spartan-3A DSP 平台提供了最具成本效益的 DSP 器件,其架构的核心就是 XtremeDSP DSP48A Slice,还提供了性能超过 30GMAC/s、存储器带宽高达 2196Mb/s 的新型 XC3SD3400A 和 XC3SD1800A 器件。新型 Spartan-3A DSP 平台是成本敏感型 DSP 算法和需要极高 DSP 性能的协处理应用的理想之选。其主要特征如下所示:

- (1) 采用 90nm 工艺,密度高达 74 880 个逻辑单元;
- (2) 内嵌的 DSP48A 可以工作到 250MHz;
- (3) 采用结构化的 SelectRAM 架构,提供了大量的片上存储单元;
- (4)  $V_{CCAU}$  的电压支持 2.5V 和 3.3V,对于 3.3V 的应用简化了设计;
- (5) 低功耗效率,Spartan-3A DSP 器件具有很高的信号处理能力,可达到 4.06 GMACs/mW。

Spartan-3A DSP 系列产品的主要技术特征如表 1-3 所示。

表 1-3 Spartan-3A DSP 系列 FPGA 主要技术特征

型 号	系 统 门数	Slice 数 目	分 布 式 RAM 容 量(Kb)	块 RAM 容 量(Kb)	专 用 乘 法 器 数	DCM 数 目	最 大 可 用 I/O 数	最 大 差 分 I/O 对 数
XC3S1800A	1800k	18 770	260	1512	84	8	519	227
XC3S3400A	3400k	26 856	373	2268	126	8	469	213

## 2) Spartan-6 系列

第六代 Spartan 系列产品 Spartan-6 FPGA 基于公认的低功耗 45nm、9-金属铜层、双栅极氧化层工艺技术,提供了高级功耗管理技术、150 000 个逻辑单元、硬核 PCI Express 模块、硬核 DRAM 存储器、250MHz DSP Slice 和 3.125Gb/s 低功耗收发器,为成本敏感型应用提供了最佳的低风险、低成本、低功耗和高性能均衡。

其主要特征如下:

- (1) 采用 45nm 工艺,密度高达 150 000 个逻辑单元;
- (2) 1 个 Slice 内部具有 4 个 6 输入查找表和 8 个触发器;
- (3) 单个块 RAM 的大小为 36Kb;
- (4) 内嵌的 DSP48 Silce 可以工作到 250MHz;
- (5) LXT 系列为低成本串行收发器,最高速率可达 3.125Gb/s;
- (6) 具有 SDRAM 硬核控制器;
- (7)  $V_{CCAU}$  的电压支持 2.5V 和 3.3V,对于 3.3V 的应用简化了设计。

Spartan-6 系列产品的主要技术特征如表 1-4 所示,其中 LX 为面向逻辑开发的器件,LXT 集成了低成本收发器,极大地增加了 Spartan-6 应用范围。

表 1-4 Spartan-6 系列 FPGA 主要技术特征

型 号	Spartan-6 Slice	块 RAM 容量(Kb)	DRAM 硬核控制器	PCI-E 硬核	DSP48E Slice	GTP	I/O bank 数目	最大可用 I/O 数
XC6SLX4	600	216	0	0	8	0	4	132
XC6SLX9	1430	576	2	0	16	0	4	200
XC6SLX16	2278	576	2	0	32	0	4	232
XC6SLX25	3758	936	2	0	38	0	4	266
XC6SLX45	6822	2088	2	0	58	0	4	358
XC6SLX75	11 662	3096	4	0	132	0	6	408
XC6SLX100	15 822	4824	4	0	180	0	6	480
XC6SLX150	23 038	4824	4	0	180	0	6	576
XC6SLX25T	3758	936	2	1	38	2	4	250
XC6SLX45T	6822	2088	2	1	58	4	4	296
XC6SLX75T	11 662	3096	4	1	132	8	6	348
XC6SLX100T	15 822	4824	4	1	180	8	6	498
XC6SLX150T	23 038	4824	4	1	180	8	6	540

## 2. Virtex 类

Virtex 系列是 Xilinx 的高端产品,也是业界的顶级产品,Xilinx 公司正是凭借 Virtex 系列产品赢得市场,从而获得 FPGA 供应商的领军地位。可以说 Xilinx 以其 Virtex 系列 FPGA 产品引领现场可编程门阵列行业,主要面向电信基础设施、汽车工业、高端消费电子等应用。目前,主流芯片包括 Virtex-5 和 Virtex-6,其 Slice 结构都是 6 输入的 LUT 结构,包含 4 个 6 输入查找表和 8 个触发器。

### 1) Virtex-5 系列

Virtex-5 系列是 Xilinx 65nm 一代的 FPGA 产品,计划提供 4 种新型平台,每种平台都在高性能逻辑、串行连接功能、信号处理和嵌入式处理性能方面实现了最佳平衡。现有的 3 款平台为 LX、LXT 以及 SXT。LX 针对高性能逻辑进行了优化,LXT 针对具有低功耗串行连接功能的高性能逻辑进行了优化,SXT 针对具有低功耗串行连接功能的 DSP 和存储器密集型应用进行了优化。其主要特点如下:

- (1) 采用了最新的 65nm 工艺,结合低功耗 IP 块将动态功耗降低了 35%;此外,还利用 65nm 三栅极氧化层技术保持低静态功耗;
- (2) 利用 65nm ExpressFabric 技术,实现了真正的 6 输入 LUT,并将性能提高了 2 个速度级别;
- (3) 内置有用于构建更大型阵列的 FIFO 逻辑和 ECC 的增强型 36Kb Block RAM 带有低功耗电路不通顺,可以关闭未使用的存储器;
- (4) 逻辑单元多达 330 000 个,可以实现无与伦比的高性能;
- (5) I/O 引脚多达 1200 个,可以实现高带宽存储器/网络接口,1.25Gb/s LVDS;

- (6) 低功耗收发器多达 24 个,可以实现  $100\text{Mb/s} \sim 3.75\text{Gb/s}$  高速串行接口;
- (7) 核电压为  $1\text{V}, 550\text{MHz}$  系统时钟;
- (8)  $550\text{MHz}$  DSP48E Slice 内置有  $25 \times 18$  MAC, 提供 352GMACS 的性能,能够在将资源使用率降低 50% 的情况下,实现单精度浮点运算;
- (9) 利用内置式 PCIe 端点和以太网 MAC 模块提高面积效率;
- (10) 更加灵活的时钟管理管道(Clock Management Tile),结合了用于进行精确时钟相位控制与抖动滤除的新型 PLL 和用于各种时钟综合的数字时钟管理器(DCM);
- (11) 采用了第二代 sparse chevron 封装,改善了信号完整性,并降低了系统成本;
- (12) 增强了器件配置,支持商用 Flash 存储器,从而降低了成本。

Virtex-5 系列产品的主要技术特征如表 1-5 所示。

表 1-5 Virtex-5 系列 FPGA 主要技术特征

型 号	Virtex-5 Slice	分布 式 RAM 容量(Kb)	块 RAM 容量(Kb)	以太网 MAC	DSP48E Slice	Rocket I/O	I/O bank 数目	最大可用 I/O 数
XC5VLX30	4800	320	1152	0	32	0	13	400
XC5VLX50	7200	480	1728	0	48	0	17	560
XC5VLX85	12 950	840	3456	0	48	0	17	560
XC5VLX110	17 280	1120	4608	0	64	0	23	800
XC5VLX220	34 560	2280	6912	0	128	0	23	800
XC5VLX330	51 840	3520	10 368	0	192	0	23	1200
XC5VLX30T	4800	320	1296	4	32	8	12	360
XC5VLX50T	7200	480	2160	4	48	12	15	450
XC5VLX85T	12 960	840	3888	4	48	12	15	450
XC5VLX110T	17 280	1120	5328	4	64	16	20	680
XC5VLX220T	34 560	2280	7632	4	128	16	20	680
XC5VLX330T	51 840	3420	11 664	4	192	24	27	980
XC5VSX35T	5440	520	3024	4	192	8	12	360
XC5VSX50T	8160	760	4750	4	288	12	15	480
XC5VSX95T	14 720	1520	8784	4	640	16	18	640

## 2) Virtex-6 系列

Virtex-6 是目前 Xilinx 公司最新量产的旗舰产品,比 Virtex-5 功耗降低多达 50%,成本降低多达 20%。该系列产品进行了最合适的组合优化,包括灵活性、硬内核 IP、收发器功能以及开发工具支持,从而可以帮助客户满足市场需求,在追求更高带宽的同时,适应不断演化的标准以及苛刻的性能要求。在无线/有线通信、广播、航空航天以及国防等领域中有着广泛应用。

Virtex-6 FPGA 系列包括三个面向应用领域而优化的 FPGA 平台,分别提供了不同的特性和功能组合来更好地满足不同客户应用的需求:

- (1) Virtex-6 LXT FPGA——专门应用于需要高性能逻辑、DSP 以及基于低功耗 GTX  $6.5\text{Gb/s}$  串行收发器的串行连接能力。
- (2) Virtex-6 SXT FPGA——专门应用于需要超高性能 DSP 以及基于低功耗 GTX

6.5Gb/s 串行收发器的串行连接能力。

(3) Virtex-6 HXT FPGA——专门应用于需要最高的串行连接能力,多达 64 个 GTH 串行收发器可提供高达 11.2Gb/s 带宽。

Virtex-6 系列产品的主要技术特征如表 1-6 所示。

表 1-6 Virtex-6 系列 FPGA 主要技术特征

型 号	Virtex-6 Slice	DSP48E Slice	块 RAM 容量(Kb)	以太网 MAC	PCI-E 硬核	Rocket I/O	I/O bank 数目	最大可用 I/O 数
XC6VLX75T	11 640	288	5616	4	1	12	9	360
XC6VLX130T	20 000	480	9504	4	2	20	15	600
XC6VLX195T	31 200	640	12 384	4	2	20	15	600
XC6VLX240T	37 680	768	14 976	4	2	24	18	720
XC6VLX365T	56 880	576	14 976	4	2	24	18	720
XC6VLX550T	85 920	864	22 752	4	2	36	30	1200
XC6VLX760	118 560	864	25 920	0	0	0	30	1200
XC6VSX315T	49 200	1344	25 344	4	2	24	18	720
XC6VSX475T	74 400	2016	38 304	4	2	36	21	840
XC6VHX250T	39 360	576	18 144	4	4	48	8	320
XC6VHX255T	39 360	576	18 576	2	2	24	12	480
XC6VHX380T	59 760	864	27 648	4	4	48	18	720
XC6VHX565T	88 560	864	32 832	4	4	48	18	720

## 1.3 Xilinx 软件工具

EDA 软件对最终设计的性能有着很大的影响。Xilinx 所有器件的开发都需要通过 Xilinx 公司的软件 ISE Design Suite 来完成。ISE Design Suite 涉及了 FPGA 设计的各个应用领域,包括逻辑开发、数字信号处理以及嵌入式系统开发等,主要包括 ISE Foundation、嵌入式开发套件(EDK)、System Generator DSP 开发工具、ChipScope Pro 分析仪、PlanAhead 设计和分析工具等组成部分,其完整的开发功能如图 1-8 所示。

### 1.3.1 ISE Foundation 软件

ISE Foundation 软件是 Xilinx 公司推出的 FPGA/CPLD 集成开发环境,不仅包括逻辑设计所需的一切,还具有大量简便易用的内置式工具和向导,使得 I/O 分配、功耗分析、时序驱动设计收敛、HDL 仿真等关键步骤变得容易而直观。

ISE 的主要功能包括设计输入、综合、仿真、实现和下载,涵盖了 FPGA 开发的全过程,从功能上讲,其工作流程无须借助任何第三方 EDA 软件。

(1) 设计输入: ISE 提供的设计输入工具包括用于 HDL 代码输入和查看报告的 ISE 文本编辑器(ISE Text Editor),用于原理图编辑的工具 ECS(Engineering Capture System),用于生成 IP Core 的 Core Generator,用于状态机设计的 StateCAD 以及用于约

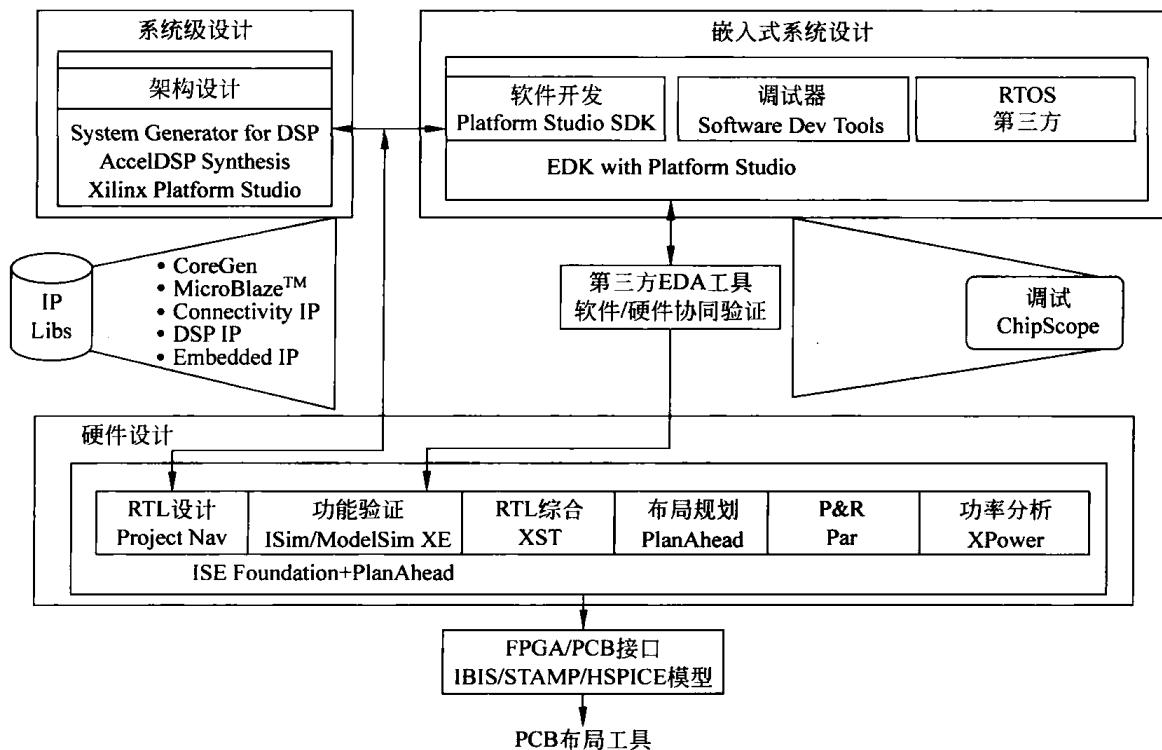


图 1-8 ISE Design Suite 的完整功能

束文件编辑的 Constraint Editor 等。

(2) 综合：ISE 的综合工具不但包含了 Xilinx 自身提供的综合工具 XST，同时还可以内嵌 Mentor Graphics 公司的 LeonardoSpectrum 和 Synplicity 公司的 Synplify，实现无缝链接。

(3) 仿真：ISE 本身自带了一个具有图形化波形编辑功能的仿真工具 HDL Bencher，同时又提供了使用 Model Tech 公司的 Modelsim 进行仿真的接口。

(4) 实现：此功能包括了翻译、映射、布局布线等，还具备时序分析、引脚指定以及增量设计等高级功能。

(5) 下载：下载功能包括了 BitGen，用于将布局布线后的设计文件转换为位流文件；还包括了 ImPACT，功能是进行设备配置和通信，控制将程序烧写到 FPGA 芯片中去。

使用 ISE 进行 FPGA 设计的各个过程可能涉及的设计工具如表 1-7 所示。

表 1-7 ISE 设计工具表

设计输入	综合	仿真	实现	下载
HDL 文本编辑器	XST		Translate	
ECS 原理图编辑器	FPGA Express	HDL Bencher	MAP	BitGen
StateCAD 状态机编辑器	(Synplify)	(ModelSim)	Place and Route	IMPACT
Core Generator	(LeonardoSpectrum)		Xpower	
Constraint Editor				

### 1.3.2 EDK 开发工具

嵌入式系统包括硬件开发和软件开发两部分,因此嵌入式开发工具是指可生成硬件平台,并能编辑、编译、链接、加载和调试高级编程语言(通常是 C 或 C++),最终将其运行在处理器引擎上的综合开发工具。EDK 就是 Xilinx 公司推出的基于 FPGA 的嵌入式开发工具,包括了嵌入式硬件平台开发工具(Platform Studio)、嵌入式软件开发工具(Platform Studio SDK)、嵌入式 IBM PowerPC 硬件处理器核、Xilinx MicroBlaze 软处理器核、开发所需的技术文档和 IP,为设计嵌入式可编程系统提供了全面的解决方案。

EDK 套件中还包括了最新的 IP 内核以优化系统设计。同时还包括了 SPI、DDR2/DDR3、DMA、PS2 和支持 SGMII 的三模式以太网 MAC 等外设,FlexrayTM 外设选项,以及用于 DMA 的 PCI Express 驱动支持。改进后的多端口存储器控制器以及存储器接口生成器(MIG)工具为存储密集应用提供了更为强大和丰富的接口选择。此外,对软核处理器 MicroBlaze 的内核 IP 进一步优化和更新,从而可以提供更大的缓存接口灵活性。

### 1.3.3 System Generator DSP 工具

目前,FPGA 芯片早就脱离扮演胶合逻辑角色的阶段,已成为数字信号处理系统的核心器件。在芯片内,不仅包含了逻辑资源,还有多路复用器、存储器、硬核乘加单元以及内嵌的处理器等设备,并且还具备高度并行计算的能力,使得 FPGA 已成为高性能数字信号处理的理想器件,特别适合于完成数字滤波、快速傅里叶变换等。

System Generator 是 Xilinx 公司的系统级建模工具,在很多方面扩展了 MathWorks 公司的 Simulink 平台,提供了适合硬件设计的数字信号处理(DSP)建模环境,加速、简化了 FPGA 的 DSP 系统级硬件设计。System Generator 提供了系统级设计能力,允许在相同的环境内进行软、硬件仿真、执行和验证,并不需要书写 HDL 代码。此外,System Generator 工具还能完成高级提取,自动编译生成 FPGA 代码,也可通过低级的提取,对 FPGA 的底层资源进行访问,从而实现高效率 FPGA 设计构建。目前,基于 System Generator 的设计方法已在复杂系统实现中展现了强大的潜能,必将成为未来流行的 FPGA 开发技术之一。

### 1.3.4 ChipScope Pro

逻辑分析仪(Logic Analyzer)是 FPGA 调试阶段不可缺少的工具,但是传统逻辑分析仪有两个弊端:首先价格昂贵;其次需要使用大量探头,不仅不合实际且操作麻烦。Xilinx 公司为了解决用户的这两个难题,推出了在线逻辑分析仪(ChipScope Pro),通过软件方式为用户提供稳定和方便的解决方案。该在线逻辑分析仪不仅具有逻辑分析仪的功能,而且成本低廉、操作简单,因此具有极高的实用价值。ChipScope Pro 既可以独立使用,也可以在 ISE 集成环境中使用,非常灵活,为用户提供方便和稳定的逻辑分析解决方案,支持 Spartan 和 Virtex 全系列 FPGA 芯片。

ChipScope Pro 将逻辑分析器、总线分析器和虚拟 I/O 小型软件核直接插入到用户的设计当中,可以直接查看任何内部信号或节点,包括嵌入式硬或软处理器。信号在操作系统速度下或接近操作系统速度下被采集,并从编程接口中引出,再将采集到的信号通过 ChipScope Pro 逻辑分析器进行分析,从而为设计解放了更多的引脚。利用 FPGA 的可重编程性能,可以在几分钟或几小时内确定设计问题并修改设计;内置的软件逻辑分析器可以用来识别设计问题并进行调试,包括高级触发、过滤和显示选项,无须重新综合即可改变探针指向;可利用远程控制(从办公室到实验室,或在全球范围内)通过互联网连接进行调试;此外还包括 Agilent 科技推出的、用于实现功能强大的验证功能的逻辑分析器可选配件,可以探测包括从 FPGA 内部到板上任何地方的交叉互联信号。

### 1.3.5 PlanAhead

PlanAhead 工具简化了综合与布局布线之间的设计步骤,能够将大型设计划分成较小的、更易于管理的模块,并集中精力优化各个模块。此外,还提供了一个直观的环境,为用户提供原理图、平面布局规划或器件图,可快速确定和改进设计的层次,以便获得更好的结果和更有效地使用资源,从而获得最佳的性能和更高的利用率,极大地提升了整个设计的性能和质量。PlanAhead 的主要功能如下。

#### 1. 轻松实现引脚规划的 PinAhead 技术

PlanAhead 包含 PinAhead 技术,可以帮助用户更好地处理引脚分配的复杂性问题。PinAhead 提供了一个以全自动或半自动方式将 I/O 端口分配到物理封装引脚上的环境。

#### 2. 整合了 ExploreAhead

ExploreAhea 是一种实现探索工具,通过管理多个实现运行。ExploreAhead 允许用户根据他们指定的策略或者作为工厂默认方法发售的预定策略,执行多个实现操作。在 Linux 环境下,ExploreAhead 具有在远程主机上运行设计的能力。

#### 3. 可完成基于模块的增量设计

PlanAhead 提供了层次化、模块化的增量设计方法,让设计者只需改变一部分设计,而保持其他部分的完整性,从而缩短了设计时间。即使是在经常改动的情况下,它也能让用户保持所需的性能。

#### 4. 提高设计的信号完整性

PlanAhead 提供了检查加权平均 SSO(WASSO)分析限制的功能。这使得设计者能够更轻松地限制 FPGA 输出处的触地反弹数量,并能够防止发生 FPGA 引起的其他器件的操作失误。

## 5. 支持部分重配置

PlanAhead 简化了针对部分重配置的、功能强大但复杂的设计流程。部分重配置是一种独特的方法,可以在静态部分仍然工作的情况下改变设计的动态部分。部分重配置可以让用户减小设计的尺寸、重量、成本和功耗。

## 6. 基于 TimeAhead 的延时估计

TimeAhead 是一种灵活的、集成到 PlanAhead 中的时序分析器,它让用户在进行布局和布线之前就可以估计布线延迟。采用基于 PlanAhead 模块的方法,可在完成布局和布线的同时,提高时序估计的准确度。

## 1.4 本书案例验证平台——S6 CARD 开发板

为了让读者更好地理解本书内容,我们专门将本书全部例子在板卡 S6 CARD 上完成,并在光盘中给出完整的工程文件,便于读者自行练习。S6 CARD 板卡相关信息见光盘说明。

### 1.4.1 S6 CARD 开发板的组成与功能

S6 CARD 开发板以 Spartan-6 系列的 XC6SLX9-TQ144 芯片为核心,供电、下载与调试都通过板卡自身的 USB 接口完成,扩展了 LED、GPIO、UART 以及 USB-JTAG 电路,正面结构如图 1-9 所示。该板卡体积小巧,和身份证大小相同,便于携带。

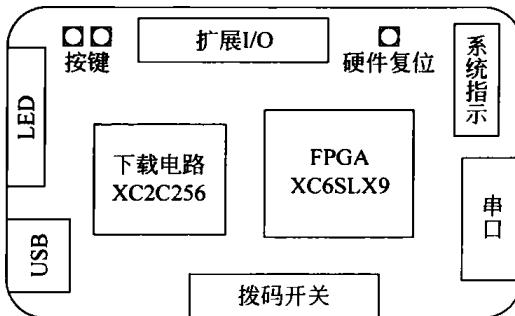


图 1-9 S6 CARD 板卡结构图

其中,FPGA 的配置芯片使用了 32Mb 的 SPI Flash M25P32 容量较大。关于 SPI Flash 的加载原理将在 3.4.4 节详细说明。

#### 1. LED 驱动电路

LED 是最基本的电路组件,给高电平就发光,且发光的程序和驱动电流有关。板卡的 LED 电路如图 1-10 所示。

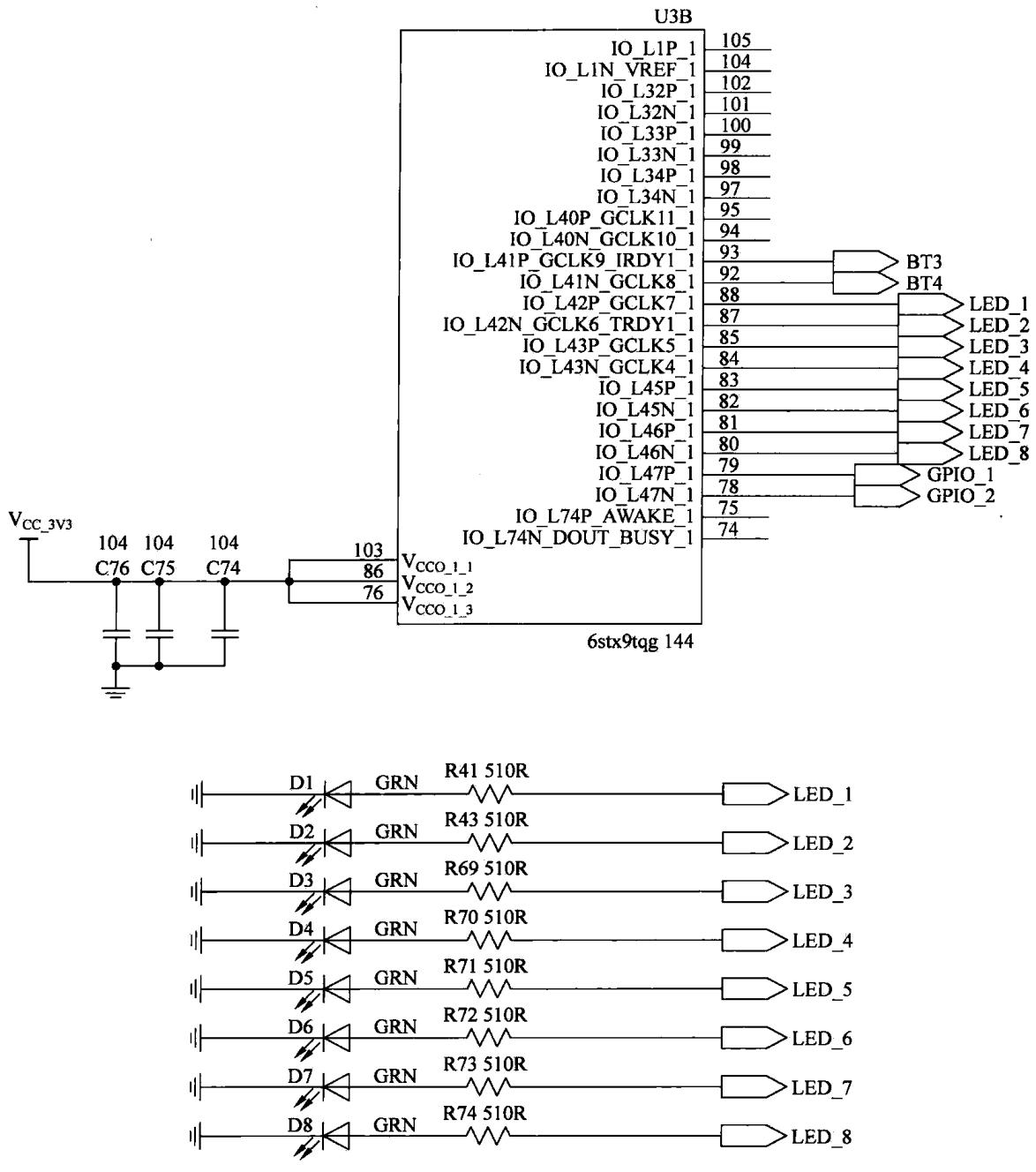


图 1-10 S6 CARD LED 电路

## 2. 按钮和拨码开关电路

LED、按钮以及拨码开关本质上属于同一类设备，LED 为输出设备，而按钮和拨码开关属于输入设备。按钮为瞬时输入设备，仅在按下时维持一个固定输入，松开则返回到固定的逻辑相反状态。S6 CARD 的按钮电路如图 1-11 所示，按钮按下时，FPGA 相应引脚为低电平。

拨码开关为长时固定状态输入，拨到哪个状态就一直维持该状态的输入。S6 CARD 的拨码开关电路如图 1-12 所示，拨码到“ON”时，FPGA 相应引脚为 0，否则为 1。

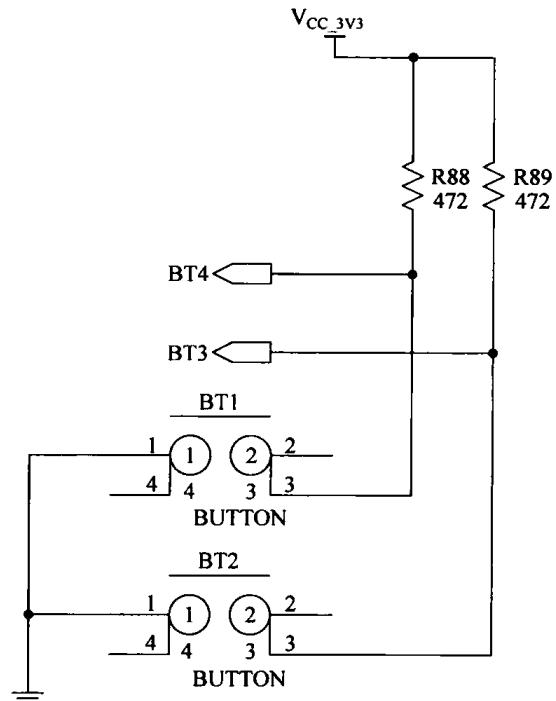


图 1-11 S6 CARD 按钮电路

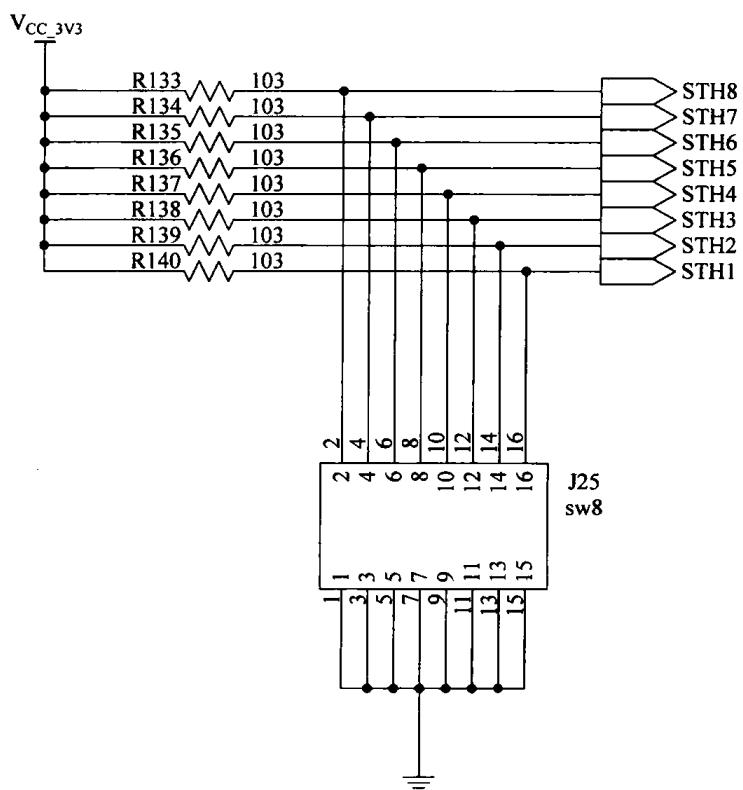


图 1-12 S6 CARD 拨码开关电路

### 3. UART 驱动电路

S6 板卡保留了 UART232 的“公头”(DTE)接口,相应的电路如图 1-13 所示。本书 6.4 节将详细介绍 UART232 工作原理。

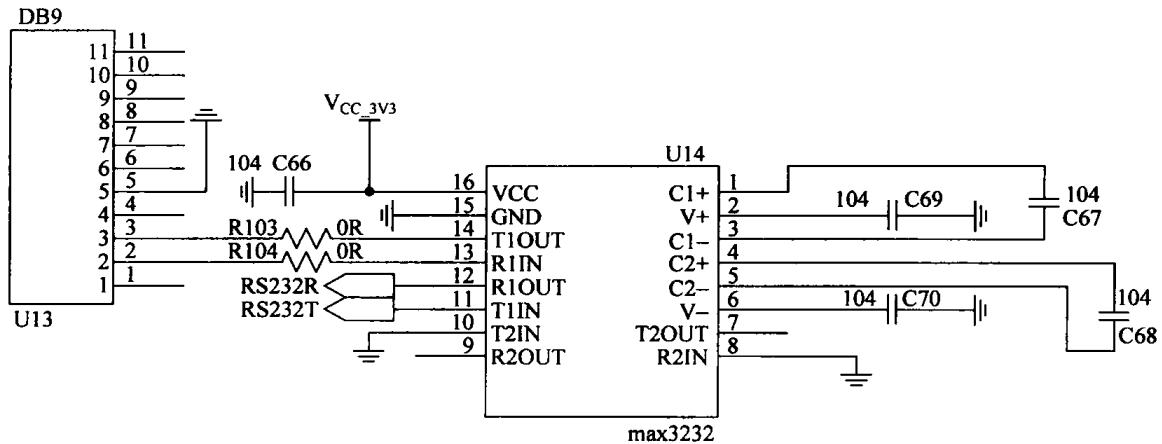


图 1-13 S6 CARD UART232 电路

### 4. 扩展引脚电路

S6 CARD 板卡预留了 20 个扩展接口,其中 2 个电源、2 个接地,16 个通用数据线,具体如图 1-14 所示。

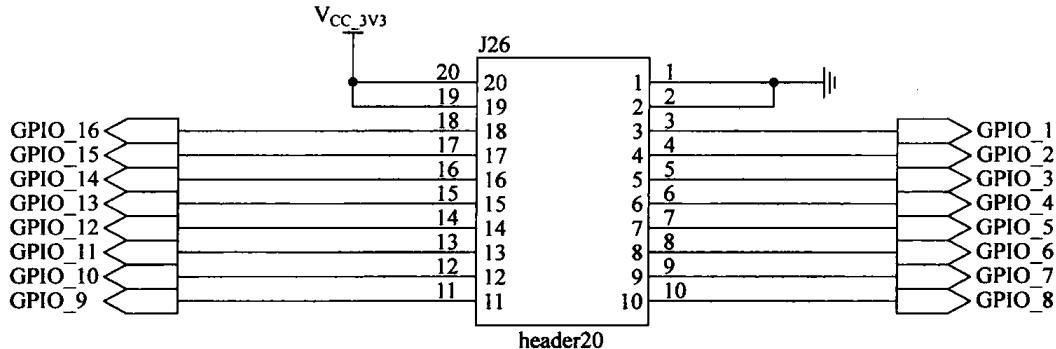


图 1-14 S6 CARD 扩展 I/O 电路

#### 1.4.2 S6 CARD 板卡引脚约束说明

为了便于读者自行练习,下面给出 S6 CARD 板卡所有的 UCF 约束。

```
# # 系统时钟引脚
NET "sys_clk_50MHz" LOC = "P56" | IOSTANDARD = LVCMOS33 ;
# # 按钮引脚
NET "button_1" LOC = "P92" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "button_2" LOC = "P93" | IOSTANDARD = LVTTL | PULLDOWN ;
```

```

## LED 引脚
NET "LED< 7 >" LOC = "P88" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 6 >" LOC = "P87" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 5 >" LOC = "P85" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 4 >" LOC = "P84" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 3 >" LOC = "P83" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 2 >" LOC = "P82" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 1 >" LOC = "P81" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED< 0 >" LOC = "P80" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
## 串口引脚
NET "rxd" LOC = "P126" | IOSTANDARD = LVTTL ;
NET "txd" LOC = "P127" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = SLOW ;
## 开关引脚
NET "Switch< 7 >" LOC = "P120" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 6 >" LOC = "P119" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 5 >" LOC = "P118" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 4 >" LOC = "P117" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 3 >" LOC = "P116" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 2 >" LOC = "P115" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 1 >" LOC = "P114" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Switch< 0 >" LOC = "P112" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
## 扩展接口引脚
NET "Gpio< 15 >" LOC = "P33" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 14 >" LOC = "P34" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 13 >" LOC = "P35" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 12 >" LOC = "P40" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 11 >" LOC = "P41" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 10 >" LOC = "P43" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 9 >" LOC = "P44" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 8 >" LOC = "P45" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 7 >" LOC = "P46" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 6 >" LOC = "P47" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 5 >" LOC = "P48" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 4 >" LOC = "P50" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 3 >" LOC = "P51" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 2 >" LOC = "P55" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 1 >" LOC = "P78" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "Gpio< 0 >" LOC = "P79" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;

```

## 本章小结

本章首先介绍了 FPGA 器件的基本概念和发展历史,然后详细介绍了 FPGA 器件的工作原理、芯片结构以及组成部件,并在此基础上讨论了 FPGA 的开发流程,给出传统的开发方法以及 SoC 阶段的设计方法。列举了 Xilinx 公司的主流 FPGA 芯片——这些芯片被广泛地应用在数字系统设计中,熟悉了解其性能指标和参数是 Xilinx 芯片开发人员所必须达到的要求。概括叙述了 Xilinx FPGA 开发软件的组成和具体功能。最后,简要介绍了本书的实验平台 S6 CARD。

# 第2章 Verilog HDL 开发基础与进阶

10 年前,相对于原理图输入设计方法,基于硬件描述语言(Hardware Description Language, HDL)的设计方式是一种非常先进的设计方法,但随着EDA工具和FPGA芯片的发展,如今,HDL已成为FPGA开发者应该掌握的基本技能,是设计者掌握SoPC和系统级设计方法的基础。HDL是以文本形式来描述数字系统硬件结构和行为的语言,用它可以表示逻辑电路图、逻辑表达式,还可以表示数字逻辑系统所完成的逻辑功能。本书以Verilog HDL语言为主,首先对其进行简单介绍,然后详细地解释其语法标准,并针对几个专题深入地讨论,最后给出常用程序实例。关于VHDL和System C的使用可参考相关文献,限于篇幅,本书不对它们展开叙述。

## 2.1 Verilog HDL 语言

Verilog HDL 和 VHDL 是目前世界上最流行的两种硬件描述语言(HDL),均为 IEEE 标准,被广泛地应用于基于可编程逻辑器件的项目开发。二者都是在 20 世纪 80 年代中期开发出来的,前者由 Gateway Design Automation(GDA)公司(该公司于 1989 年被 Cadence 公司兼并)开发,后者由美国军方研发。

HDL 语言以文本形式来描述数字系统的硬件结构和行为,是一种用形式化方法来描述数字电路和系统的语言,可以从上层到下层来逐层描述自己的设计思想。即用一系列分层次的模块来表示复杂的数字系统,并逐层进行验证仿真,再把具体的模块组合由综合工具转化成门级网表,接着再利用布局布线工具把网表转化为具体电路结构的实现。目前,这种自顶向下的方法已被广泛使用。概括地讲,HDL 语言包含以下主要特征:

- (1) HDL 语言既包含一些高级程序设计语言的结构形式,同时也兼顾描述硬件线路连接的具体结构。
- (2) 通过使用结构级行为描述,可以在不同的抽象层次描述设计。HDL 语言采用自顶向下的数字电路设计方法,主要包括 3 个领域中的 5 个抽象层次,如表 2-1 所示。

(3) HDL 语言是并行处理的,具有同一时刻执行多任务的能力。这和一般高级设计语言(例如 C 语言)串行执行的特征是不同的。

(4) HDL 语言具有时序的概念。一般的高级编程语言是没有时序概念的,但在硬件电路中从输入到输出总是有延时存在的,为了描述这一特征,需要引入时延的概念。HDL 语言不仅可以描述硬件电路的功能,还可以描述电路的时序。

### 2.1.1 Verilog HDL 语言的历史

1983 年, GDA 公司的 Philip Moorby 首创了 Verilog HDL。后来 Moorby 成为 Verilog HDL-XL 的主要设计者和 Cadence 公司的第一合伙人。1984—1986 年,Moorby 设计出第一个关于 Verilog HDL 的仿真器,并提出了用于快速门级仿真的 XL 算法,使 Verilog HDL 语言得到迅速发展。1987 年 Synopsys 公司开始使用 Verilog HDL 行为语言作为综合工具的输入。1989 年 Cadence 公司收购了 GDA 公司,Verilog HDL 成为 Cadence 公司的私有财产。1990 年初,Cadence 公司把 Verilog HDL 和 Verilog HDL-XL 分开,并公开发布了 Verilog HDL。随后成立的 OVI (Open Verilog HDL International)组织负责 Verilog HDL 的发展并制定有关标准,OVI 由 Verilog HDL 的使用者和 CAE 供应商组成。1993 年,几乎所有 ASIC 厂商都开始支持 Verilog HDL,并且认为 Verilog HDL-XL 是最好的仿真器。同时,OVI 推出 2.0 版本的 Verilog HDL 规范,IEEE 则将 OVI 的 Verilog HDL 2.0 作为 IEEE 标准的提案。1995 年 12 月,IEEE 制定了 Verilog HDL 的标准 IEEE1364-1995。目前,最新的 Verilog 语言版本是 2000 年 IEEE 公布的 Verilog 2001 标准,其大幅度地提高了系统级和可综合性能。

### 2.1.2 Verilog HDL 的主要功能

Verilog HDL 既是一种行为描述语言,也是一种结构描述语言。如果按照一定的规则和风格编写代码,就可以将功能行为模块通过工具自动转化为门级互连的结构模块。这意味着利用 Verilog 语言的功能,就可以构造一个模块间的清晰结构来描述复杂的大规模设计,并对所需的逻辑电路进行严格的设计。

下面是 Verilog 语言的主要功能:

- (1) 可描述顺序执行或并行执行的程序结构;
- (2) 用延迟表示式或事件表达式来明确地控制过程的启动时间;
- (3) 通过命名的事件来触发其他过程里的激活行为或停止行为;
- (4) 提供了条件和循环等程序结构;
- (5) 提供了可带参数且非零延续时间的任务程序结构;
- (6) 提供了可定义新的操作符的函数结构;
- (7) 提供了用于建立表达式的算术运算符、逻辑运算符和位运算符;
- (8) 提供了一套完整的表示组合逻辑基本元件的原语;
- (9) 提供了双向通路和电阻器件的描述;
- (10) 可建立 MOS 器件的电荷分享和衰减模型;

(11) 可以通过构造性语句精确地建立信号模型。

表 2-1 给出 Verilog HDL 的表述能力。

表 2-1 Verilog HDL 语言能力总结

描述级别	抽象级别	功能描述	物理模型
行为级	系统级	用语言提供的高级结构能够实现所设计模块外部性能的模型	芯片、电路板和物理划分的子模块
	算法级	用语言提供的高级功能能够实现算法运行的模型	部件之间的物理连接, 电路板
	RTL 级	描述数据如何在寄存器之间流动和如何处理、控制这些数据流动的模型	芯片、宏单元
逻辑级	门级	描述逻辑门和逻辑门之间连接的模型	标准单元布图
电路级	开关级	描述器件中三极管和存储节点以及它们之间连接的模型	晶体管布图

注意: Verilog HDL 语言有一个重要特征: 和 C 语言风格有很多的相似之处, 学习起来比较容易。

### 2.1.3 Verilog HDL 和 VHDL 的区别

Verilog HDL 和 VHDL 都是用于逻辑设计的硬件描述语言。VHDL 在 1987 年成为 IEEE 标准, Verilog HDL 则在 1995 年才成为 IEEE 标准, 这是因为前者是美国军方组织开发的, 而后者则是从民间公司转化而来, 要成为国际标准就必须放弃专利。相比而言, Verilog HDL 具有更强的生命力。

Verilog HDL 和 VHDL 的相同点在于: 都能形式化地抽象表示电路的行为和结构; 支持逻辑设计中层次与范围的描述; 可以简化电路行为的描述; 具有电路仿真和验证机制; 支持电路描述由高层到低层的综合转换; 与实现工艺无关; 便于管理和设计重用。

但 Verilog HDL 和 VHDL 又有各自的特点, 由于 Verilog HDL 推出较早, 因而拥有更广泛的客户群体、更丰富的资源。Verilog HDL 还有一个优点就是容易掌握, 如果具有 C 语言学习的基础, 很快就能够掌握。而 VHDL 需要 Ada 编程语言基础, 一般需要半年以上的专业培训才能够掌握。传统观点认为 Verilog HDL 在系统级抽象方面较弱, 不太适合特大型的系统。但经过 Verilog 2001 标准的补充之后, 系统级表述性能和可综合性能有了大幅度提高。当然, 这两种语言也仍处于不断完善的过程中, 都在朝着更高级描述语言的方向前进。

### 2.1.4 Verilog HDL 设计方法

#### 1. 自下而上的设计方法

自下而上的设计是传统的设计方法, 是从基本单元出发, 对设计进行逐层划分的过程。这种设计方法与用电子元件在模拟实现板上建立一个系统的步骤有密切的关系。

该设计方法的优点与缺点如下：

1) 优点

设计人员对这种设计方法比较熟悉；实现各个子模块所需的时间较短。

2) 缺点

对系统的整体功能把握不足；由于必须先对多个子模块进行设计，因此实现整个系统功能所需时间长；另外，对设计人员之间相互协作也有较高的要求。

## 2. 自上而下的设计方法

自上而下的设计是从系统级开始，把系统划分为基本单元，然后再把基本单元划分为下一层次的基本单元，直到可用 EDA 元件实现为止。该设计方法的优点与缺点如下：

1) 优点

在设计周期开始就做好了系统分析；由于设计的主要仿真和调试过程是在高层完成的，所以能够在早期发现结构设计上的错误，避免了设计工作的浪费，方便了系统的划分和整个项目的管理，可减少设计人员劳动，避免了重复设计。

2) 缺点

得到的最小单元不标准，且制造成本高。

## 3. 混合的设计方法

复杂数字逻辑电路和系统设计过程，通常以上两种设计方法的结合。设计时需要考虑多个目标的综合平衡。在高层系统用自上而下的设计方法实现，而使用自下而上的方法从库元件或以往设计库中调用已有的设计单元。混合设计方法兼有以上两种方法的优点，并且可使用先进的矢量测试方法。

在实际工程开发中，往往采取混合设计方法，首先从高层开始按照功能划分相对独立、规模较大的功能子模块，然后每个模块内部再根据各个底层模块实现难度以及时序要求等方面综合考虑，在高层需求的前提下，从底层模块一级一级向上开发，再连接成整个功能子模块。最后，将子模块有效组合成整个设计。

## 2.2 Verilog HDL 基本程序结构

用 Verilog HDL 描述的电路设计就是该电路的 Verilog HDL 模型，也称为模块，是 Verilog 的基本描述单位。模块描述某个设计的功能或结构以及与其他模块通信的外部接口，一般来说一个文件就是一个模块，但并不绝对如此。模块是并行运行的，通常需要一个高层模块通过调用其他模块的实例来定义一个封闭的系统，包括测试数据和硬件描述。一个模块的基本架构如下：

```
module module_name (port_list)
    //声明各种变量、信号
    reg          //寄存器
    wire         //线网
    parameter    //参数
    input        //输入信号
```

## 有关此电子图书的说明

本人由于一些便利条件，可以帮您提供各种中文电子图书资料，且质量均为清晰的 PDF 图片格式，质量要高于网上大量传播的一些超星 PDG 的图书。方便阅读和携带。只要图书不是太新，文学、法律、计算机、人文、经济、医学、工业、学术等方面 的图书，我都可以帮您找到电子版本。所以，当你想要看什么图书时，可以联系我。我的 QQ 是：**89039855**，大家可以在 QQ 上联系我。

此 PDF 文件为本人亲自制作，请各位爱书之人尊重个人劳动，敬请您不要修改此 PDF 文件。因为这些图书都是有版权的，请各位怜惜电子图书资源，不要随意传播，否则，这些资源更难以得到。