

硬件电路工程师从入门到提高丛书

# 可编程逻辑器件和 EDA 设计技术

姜雪松 张海风 编著



 机械工业出版社  
CHINA MACHINE PRESS

硬件电路工程师从入门到提高丛书

# 可编程逻辑器件和 EDA 设计技术

姜雪松 张海风 编著



机械工业出版社

目前,可编程逻辑器件和 EDA 设计技术的结合使得电子系统设计变得更加简单方便、灵活快速,因此掌握可编程逻辑器件和 EDA 设计技术已经成为通信与电子类技术人员的一项重要设计手段和技能。

本书从实际应用的角度出发,全面系统地介绍了可编程逻辑器件和 EDA 设计技术。本书第 1 部分主要介绍了可编程逻辑器件和 EDA 设计技术的基本知识,然后讨论了可编程逻辑器件的基本原理和 Xilinx 公司的主流 CPLD 和 FPGA 器件。第 2 部分主要介绍了 EDA 设计技术中的 VHDL 设计方法,然后重点讨论了两种流行的 EDA 开发工具: Altera 公司的 MAX + plusII 和 Xilinx 公司的 ISE。第 3 部分通过大型的应用实例来讨论可编程逻辑器件和 EDA 设计技术的结合应用。

本书内容丰富、技术新颖、实用性强,可以使读者快速、全面地掌握可编程逻辑器件和 EDA 设计技术。本书既可以作为高等院校电子通信类高年级本科生、研究生的教材或教学参考书,同时也可以作为从事电子系统设计的工程师和科研人员的技术参考书。

## 图书在版编目 (CIP) 数据

可编程逻辑器件和 EDA 设计技术/姜雪松等编著. —北京:机械工业出版社, 2005.9

(硬件电路工程师从入门到提高丛书)

ISBN 7-111-17314-7

I. 可... II. 姜... III. ①可编程逻辑器件②电子电路—电路设计: 计算机辅助设计 IV. ①TP332.1②TN702

中国版本图书馆 CIP 数据核字 (2005) 第 100249 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

责任编辑: 张俊红 版式设计: 冉晓华 责任校对: 李汝庚

封面设计: 陈 沛 责任印制: 杨 曦

济南新华印刷厂印刷

2006 年 1 月第 1 版·第 1 次印刷

787mm × 1092mm <sup>1</sup>/<sub>16</sub> · 23.75 印张 · 587 千字

0001—4000 册

定价: 38.00 元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

本社购书热线电话 (010) 68326294

封面无防伪标均为盗版

# 硬件电路工程师从入门到提高丛书

## 编 委 会

|     |     |     |     |
|-----|-----|-----|-----|
| 主 编 | 姜雪松 |     |     |
| 副主编 | 张俊红 | 张 凯 |     |
| 编 委 | 方华刚 | 姜立冬 | 蒋 亮 |
|     | 李晓凯 | 齐兆群 | 张 蓬 |
|     | 赵 鑫 | 叶 琅 | 许灵军 |

# 丛 书 序

随着我国经济建设的发展和科学技术的不断进步,以往有些硬件电路设计的书籍内容已经比较陈旧、落后,难以适应高等院校教学和硬件电路工程师的要求。特别是在电子学和通信技术发展神速、社会发展日新月异的今天,如何适应这种情况和要求,已经成为一个必须认真考虑的问题。

如今,我国已成为全球增长潜力最大的电子产品消费大国,同时也是全球最大的移动电话市场和第3大PC市场,未来5年还将成为全球第2大半导体市场。中国市场的商机令世界各国IT公司心动不已,竞相调整中国战略,纷纷加大投资。这种情况必将导致对硬件电路工程师的海量需求。以IC人才为例,据不完全统计,全国目前定位于IC设计的企业大约200多家,IC设计人员还不到4000人,大都是小作坊模式,每个企业只有两三人掌握某一方面芯片的专长。从总体上看,按未来几年的市场需求,每年所需IC设计人才保守估计在5万人左右,如果才能保证整个IC设计产业正常运作,人才需求量则高达20~50万。可见,提高硬件电路设计的人才教育,加强硬件电路工程师的人才储备,已经成为高等院校和各大IT公司的当务之急。

硬件电路设计是一门涉及到多门学科、实用性非常强的技术,因此硬件电路设计人员的培养需要进行大量的实践,而不仅仅是纸上谈兵。对于硬件电路设计人员的培养,除了需要培养具体的设计技术和设计技巧外,更为重要的是需要培养设计人员的创新意识。为此,组织一套理论严谨、内容新颖、实用性较强的硬件电路设计丛书,将会对我国的硬件电路设计人才的培养起到很大的推动作用。机械工业出版社的领导和编辑们独具慧眼,选题准确,决策果断,通过对硬件电路设计的相关选题进行层层筛选,最终选定了8个十分具有代表性的选题;同时组织了一批多年从事硬件电路设计、具有丰富实践经验的硬件电路设计工程师来进行编写,目的是保证这套丛书的质量和实用性。这套硬件电路工程师从入门到提高丛书包括:

- 《Verilog HDL与数字电路设计》
- 《VHDL与数字电路设计》
- 《可编程逻辑器件和EDA设计技术》
- 《印制电路板设计》
- 《Protel DXP电路设计入门与应用》
- 《HyperLynx仿真与PCB设计》
- 《DSP原理与应用》
- 《嵌入式系统原理与应用》

这套丛书从实际应用的角度出发,详细介绍了目前硬件电路设计的各个主要方面。这套丛书非常重视可读性,内容深入浅出,便于读者自学;同时也非常注重实践性,列举了典型的工程实例,体现了硬件电路设计书籍的实践性,从而可以使读者快速高效地掌握相关领域的知识。这套丛书面向所有的硬件电路工程师和立志于成为硬件电路工程师的相关专业人

员,既可以作为高等院校相关专业高年级本科生、研究生的教材或者教学参考书,同时也可以作为各类从事电子系统设计的科研人员硬件电路工程师的应用参考书。

最后,预祝机械工业出版社硬件电路工程师从入门到提高丛书取得成功,为我国硬件电路工程师的人才培养和发展贡献一份力量。同时对参与这套丛书工作的各位作者、出版社的领导和编辑们表示衷心的感谢,感谢你们为我国硬件电路工程师的人才培养和储备所作的努力!

硬件电路工程师从入门到提高丛书编委会

# 前 言

在电子系统设计领域中, 电子设计自动化 (Electronic Design Automation, EDA) 工具已经被越来越多的电子工程师所接受, 它必将取代人工设计方法成为主要的设计手段。目前, 可编程逻辑器件和 EDA 设计技术的结合大大改变了传统的系统设计方法, 传统的“固定功能模块+连线”的设计方法正在逐步退出历史舞台, 而基于芯片的设计方法正在成为电子系统设计的主流。可编程逻辑器件和 EDA 设计技术的结合使得电子系统设计变得更加简单方便、灵活快速, 因此掌握可编程逻辑器件和 EDA 设计技术已经成为从事电子系统设计的工程师和科研人员的一项重要设计手段和技能。

可编程逻辑器件广泛应用于计算机应用电路、数字电路、通信系统、工业自动控制、仪器仪表和集成电路设计等领域, 它能够将大量逻辑功能集成于一个单片 IC (集成电路) 之中。虽然专用集成电路 (ASIC) 也能够实现将大量逻辑功能集成于单片 IC 之中, 但是可编程逻辑器件具有更高的灵活性, 同时能够缩短开发周期, 加快产品的上市时间。可见, 可编程逻辑器件的出现大大改变了传统的电路系统设计方法, 从而形成了一种新的基于芯片的设计方法。通过基于芯片的设计方法, 可以大大减少系统设计中的元器件数量、缩小系统体积和降低功率消耗等, 从而提高了设计系统的性能和可靠性。

通常, EDA 设计技术就是以计算机硬件和系统软件为操作平台, 借鉴前人在电路和系统、数据库管理、图形学、图论和拓扑逻辑、编译原理、计算数学、人工智能和微电子学集成电路设计等领域中的最新科技成果而研制开发的商品化 EDA 通用支撑软件和应用软件包, 目的是帮助设计人员在计算机上完成数字系统的逻辑综合、布局布线、设计仿真和印制电路板设计等工作。在基于芯片的设计方法中, EDA 设计技术扮演了越来越重要的角色, 逐渐成为设计人员不可或缺的重要设计手段。如今, 无论是逻辑芯片设计还是电子系统设计, 所有的设计工作都需要在计算机上借助相应的 EDA 工具进行, 否则将会很难完成相应的设计任务。

可编程逻辑器件和 EDA 设计技术的结合为电子系统的设计带来了极大的方便, 它们已经逐渐成为设计人员进行硬件设计的最强有力工具。现在设计人员只要拥有一台计算机、一套相应的 EDA 开发工具和可编程逻辑器件, 那么基本上就可以实现电子系统的设计操作。可以毫不夸张地说, 任何的电子系统设计都离不开可编程逻辑器件和 EDA 开发工具。现在, 很多电子系统设计相关领域的各大公司和科研单位都采用可编程逻辑器件和 EDA 开发工具的结合来进行相应的设计, 这样可以大大缩短系统的设计周期, 以适应当今品种多、批量小的电子市场的需求, 提高产品的竞争能力。

本书将从实际应用的角度出发, 全面系统地介绍可编程逻辑器件和 EDA 设计技术。本书第 1 部分主要介绍了可编程逻辑器件和 EDA 设计技术的基本知识, 目的是使读者对其概念有一个清楚的认识; 接下来比较全面地讨论了可编程逻辑器件的基本原理, 然后以全球著名的可编程逻辑器件厂商 Xilinx 的产品为背景, 系统地介绍了 Xilinx 公司的主流 CPLD (复杂可编程逻辑器件) 和 FPGA (现场可编程门阵列) 器件。第 2 部分主要介绍了 EDA 设计

技术中的 VHDL (甚高速集成电路硬件描述语言) 设计方法, 然后重点讨论了两种流行的 EDA 开发工具: Altera 公司的 MAX + plusII 和 Xilinx 公司的 ISE, 目的是使读者掌握目前流行的 EDA 设计技术。第 3 部分通过大型的应用实例来讨论可编程逻辑器件和 EDA 设计技术的结合应用, 目的是使读者能够掌握可编程逻辑器件和 EDA 设计技术的实际应用。

本书内容丰富、技术新颖、实用性强, 可以使读者快速、全面地掌握可编程逻辑器件和 EDA 设计技术。本书既可以作为高等院校电子通信类高年级本科生、研究生的教材和教学参考书, 同时对于通信、雷达、程控交换、医疗器械、消费电子、工业控制和航空航天等各个领域的设计工程师和科研人员来说, 也是一本具有指导价值和实用价值的技术参考书。需要说明的是, 为了尽量保持各器件自身的固有特征, 书中部分图形符号和文字符号并未按国家标准做统一修改, 这点请读者注意。

本书由姜雪松和张海风工程师共同编写, 书中包含着作者多年使用可编程逻辑器件和 EDA 设计技术开发工程项目的经验总结。其中, 第 1 章~第 5 章由姜雪松编写, 第 6 章~第 9 章由张海风编写, 全书由姜雪松统稿。在本书编写的过程中, 葛树涛、夏钦东、王涛、曹建军、姜雪峰、蒋伟、尹斯星、杜平、丁海波和张海涛等参与了全书的校对和程序调试工作, 这里向他们表示由衷的感谢。另外, 作者在编写本书的过程中参考了不少专家和学者的著作以及 Xilinx 公司的数据手册, 在此表示深深的谢意!

由于可编程逻辑器件和 EDA 设计技术发展迅速, 各种相关新技术不断涌现, 同时限于作者的理论水平和实际开发经验, 书中难免存在一些不足之处或者错误, 愿望广大读者和相关专家批评指正。

作 者

# 目 录

## 丛书序

## 前言

## 第 1 章 可编程逻辑器件和 EDA 设计

### 技术 ..... 1

#### 1.1 可编程逻辑器件 ..... 1

##### 1.1.1 可编程逻辑器件的发展 ..... 1

##### 1.1.2 可编程逻辑器件的分类 ..... 3

#### 1.2 EDA 设计技术 ..... 8

##### 1.2.1 EDA 设计技术的发展历史 ..... 9

##### 1.2.2 EDA 设计技术的基本特征 ..... 10

##### 1.2.3 EDA 设计技术的工具 ..... 11

##### 1.2.4 EDA 设计技术的发展趋势 ..... 14

## 第 2 章 可编程逻辑器件的基本原理 ..... 15

### 2.1 可编程逻辑器件的基本结构 ..... 15

### 2.2 简单的可编程逻辑器件 ..... 16

#### 2.2.1 可编程只读存储器 (PROM) ..... 17

#### 2.2.2 可编程逻辑阵列 (PLA) ..... 18

#### 2.2.3 可编程阵列逻辑 (PAL) ..... 21

#### 2.2.4 通用阵列逻辑 (GAL) ..... 24

### 2.3 可编程逻辑器件: EPLD 和 CPLD ..... 30

#### 2.3.1 可擦除的可编程逻辑器件 (EPLD) ..... 30

#### 2.3.2 复杂的可编程逻辑器件 (CPLD) ..... 34

### 2.4 现场可编程门阵列 (FPGA) ..... 41

#### 2.4.1 查找表结构 ..... 42

#### 2.4.2 多路开关结构 ..... 43

#### 2.4.3 多级与非门结构 ..... 44

## 第 3 章 Xilinx 公司可编程逻辑器件的

### 结构原理 ..... 45

#### 3.1 XC9500 系列 CPLD 的结构 原理 ..... 45

##### 3.1.1 XC9500 系列 CPLD 的结构框图 ..... 46

##### 3.1.2 XC9500 系列 CPLD 的基本单元 ..... 47

##### 3.1.3 XC9500 系列 CPLD 的其他特性 ..... 54

##### 3.1.4 XC9500 系列 CPLD 的时序模型 ..... 56

### 3.2 CoolRunner XPLA3 系列 CPLD 的结构原理 ..... 57

#### 3.2.1 CoolRunner XPLA3 系列 CPLD 的 结构框图 ..... 59

#### 3.2.2 CoolRunner XPLA3 系列 CPLD 的 基本单元 ..... 59

#### 3.2.3 CoolRunner XPLA3 系列 CPLD 的 时序模型 ..... 64

### 3.3 CoolRunner-Ⅱ 系列 CPLD 的 结构原理 ..... 65

#### 3.3.1 CoolRunner-Ⅱ 系列 CPLD 的结构框图 ..... 67

#### 3.3.2 CoolRunner-Ⅱ 系列 CPLD 的 基本单元 ..... 67

#### 3.3.3 CoolRunner-Ⅱ 系列 CPLD 的 时序模型 ..... 70

### 3.4 Spartan-Ⅱ 和 Spartan-Ⅱ E 系列 FPGA 的结构原理 ..... 71

#### 3.4.1 Spartan-Ⅱ 和 Spartan-Ⅱ E 系列 FPGA 的结构框图 ..... 73

#### 3.4.2 Spartan-Ⅱ 和 Spartan-Ⅱ E 系列 FPGA 的组成部分 ..... 74

#### 3.4.3 延迟锁相环的应用 ..... 81

#### 3.4.4 块状 RAM 的应用 ..... 86

### 3.5 Virtex-Ⅱ 系列 FPGA 的结构 原理 ..... 91

#### 3.5.1 Virtex-Ⅱ 系列 FPGA 的结构框图 ..... 92

#### 3.5.2 Virtex-Ⅱ 系列 FPGA 的组成部分 ..... 93

## 第 4 章 VHDL 设计方法 ..... 102

### 4.1 VHDL 概述 ..... 102

### 4.2 VHDL 程序的结构 ..... 103

|                                           |     |                                   |     |
|-------------------------------------------|-----|-----------------------------------|-----|
| 4.2.1 实体说明                                | 103 | 6.2.4 元器件设计                       | 182 |
| 4.2.2 结构体                                 | 104 | 6.2.5 使用 PACE 创建管脚和面积约束           | 182 |
| 4.2.3 库和程序包                               | 107 | 6.3 添加时序约束                        | 185 |
| 4.2.4 配置                                  | 108 | 6.4 仿真行为模型                        | 188 |
| 4.3 VHDL 程序的元素                            | 110 | 6.5 使用 ModelSim 进行行为仿真            | 190 |
| 4.3.1 数据对象                                | 110 | 6.5.1 行为仿真                        | 190 |
| 4.3.2 数据类型                                | 112 | 6.5.2 布局布线后的仿真                    | 191 |
| 4.3.3 运算操作符                               | 116 | 6.6 原理图方式的设计输入                    | 191 |
| 4.4 VHDL 程序的描述语句                          | 119 | 6.6.1 创建生成一个原理图符号                 | 191 |
| 4.4.1 并行描述语句                              | 120 | 6.6.2 创建一个新的顶层原理图                 | 192 |
| 4.4.2 顺序描述语句                              | 126 | 6.6.3 例化 VHDL 模块                  | 192 |
| <b>第 5 章 EDA 开发工具——MAX + plus II</b>      | 135 | 6.6.4 原理图中连线                      | 193 |
| 5.1 MAX + plus II 开发工具简介                  | 135 | 6.6.5 为连线添加网络名                    | 193 |
| 5.1.1 MAX + plus II 开发工具的特点               | 135 | 6.6.6 为总线添加网络名                    | 195 |
| 5.1.2 MAX + plus II 对系统的配置要求              | 136 | 6.6.7 添加输入/输出管脚标记                 | 195 |
| 5.1.3 MAX + plus II 的安装和卸载                | 137 | 6.7 设计输入(FSM 状态机输入)               | 196 |
| 5.1.4 MAX + plus II 的功能模块                 | 142 | 6.7.1 状态机的建立和产生 VHDL 代码           | 197 |
| 5.2 MAX + plus II 的设计输入                   | 144 | 6.7.2 设计的功能仿真并产生测试平台              | 202 |
| 5.2.1 原理图设计输入                             | 145 | <b>第 7 章 CPLD/FPGA 的边界扫描与下载方式</b> | 205 |
| 5.2.2 文本设计输入                              | 150 | 7.1 概述                            | 205 |
| 5.3 MAX + plus II 的设计编译                   | 153 | 7.2 边界扫描测试                        | 206 |
| 5.3.1 可编程逻辑器件的设置                          | 153 | 7.2.1 边界扫描测试的结构                   | 206 |
| 5.3.2 设计项目的编译操作                           | 156 | 7.2.2 测试逻辑的控制                     | 212 |
| 5.4 MAX + plus II 的设计校验                   | 158 | 7.2.3 边界扫描测试的应用                   | 214 |
| 5.4.1 设计的仿真分析                             | 158 | 7.2.4 设计举例                        | 220 |
| 5.4.2 设计的定时分析                             | 164 | 7.3 Xilinx 器件的下载                  | 223 |
| 5.5 MAX + plus II 的器件编程                   | 165 | 7.3.1 Xilinx 器件的下载电缆              | 223 |
| <b>第 6 章 EDA 开发工具——ISE</b>                | 167 | 7.3.2 Xilinx 器件的下载方式              | 227 |
| 6.1 设计开始                                  | 168 | <b>第 8 章 I2C 总线的设计</b>            | 231 |
| 6.1.1 软件的安装                               | 168 | 8.1 I2C 总线的结构                     | 231 |
| 6.1.2 运行 ISE 软件                           | 171 | 8.1.1 I2C 总线的基本结构                 | 231 |
| 6.1.3 使用在线帮助                              | 176 | 8.1.2 双向传输的接口特性                   | 231 |
| 6.2 VHDL 的设计输入                            | 177 | 8.1.3 I2C 总线上的时钟信号                | 232 |
| 6.2.1 创建一个新的项目                            | 177 | 8.1.4 总线竞争的仲裁                     | 232 |
| 6.2.2 使用 Architecture Wizard 来创建一个 DCM 模块 | 178 | 8.2 I2C 总线的具体实现                   | 233 |
| 6.2.3 DCM 模块的声明和例化                        | 181 | 8.2.1 原理框图                        | 233 |

|                                            |            |                                           |            |
|--------------------------------------------|------------|-------------------------------------------|------------|
| 8.2.2 接口信号说明.....                          | 234        | 9.5.2 内存初始化.....                          | 293        |
| 8.2.3 I2C 总线设计的详细框图 .....                  | 234        | 9.5.3 DDR SDRAM 控制器的主状态机<br>说明.....       | 294        |
| 8.2.4 微处理器接口模块.....                        | 235        | 9.5.4 内存访问.....                           | 294        |
| 8.2.5 I2C 模块说明 .....                       | 237        | 9.5.5 程序说明.....                           | 294        |
| 8.3 程序说明 .....                             | 240        | 9.6 数据通道模块 .....                          | 320        |
| 8.3.1 顶层程序说明.....                          | 240        | 9.6.1 写数据通道.....                          | 322        |
| 8.3.2 I2C-CONTROL.VHD 程序说明 ..              | 246        | 9.6.2 读数据通道.....                          | 323        |
| 8.3.3 移位模块程序说明<br>(SHIFT.VHD) .....        | 264        | 9.6.3 程序说明.....                           | 323        |
| 8.3.4 计数器模块说明<br>(UPCNT4.VHD) .....        | 265        | 9.6.4 V2_DDR_IOB_8.VHD 程序<br>说明.....      | 331        |
| 8.3.5 微处理器接口模块<br>(UC_INTERFACE.VHD) ..... | 266        | 9.6.5 V2_DDR_IOB.VHD 模块<br>说明.....        | 335        |
| <b>第 9 章 DDR SDRAM 接口控制器.....</b>          | <b>274</b> | 9.6.6 SYNC_DQS2CLK_DRAM.VHD<br>模块说明 ..... | 339        |
| 9.1 概述.....                                | 274        | 9.6.7 RAM_8D.VHD 模块 .....                 | 347        |
| 9.2 设计要点 .....                             | 275        | 9.6.8 RAM_8D_1.VHD 模块 .....               | 351        |
| 9.3 顶层程序模块 .....                           | 277        | 9.7 DQS 选通模块 .....                        | 355        |
| 9.4 时钟模块 .....                             | 282        | 9.8 测试平台 .....                            | 360        |
| 9.5 控制模块说明 .....                           | 290        | <b>参考文献 .....</b>                         | <b>368</b> |
| 9.5.1 控制模块概述.....                          | 290        |                                           |            |

# 第 1 章 可编程逻辑器件和 EDA 设计技术

## 1.1 可编程逻辑器件

目前, 可编程逻辑器件广泛应用于计算机应用电路、数字电路设计、通信系统、工业自动控制、仪器仪表和集成电路设计等领域, 它的出现大大改变了传统的系统设计方法, 传统的“固定功能模块+连线”的设计方法正在逐步退出历史舞台, 而基于芯片的设计方法正在成为电子系统设计的主流。

### 1.1.1 可编程逻辑器件的发展

随着科学技术的发展, 集成电路的设计正朝着速度快、性能高、容量大、体积小和微功耗的方向发展, 这种发展必将导致集成电路的设计规模日益增大, 复杂程度日益增高。经过 40 多年的发展, 集成电路已经从小规模集成 (SSI)、中规模集成 (MSI)、大规模集成 (LSI) 发展到超大规模集成 (VLSI) 和特大规模集成 (ULSI), 每个芯片可以集成数百万个以上的晶体管。如今, 大规模和超大规模集成电路技术已经成为高科技研发领域的基础。

如果从具体的逻辑功能特点上来进行分类, 那么大规模和超大规模集成电路可以分为通用集成电路和专用集成电路两大类。例如, 常见的 74 系列、CC4000 系列、74HC 系列小规模数字集成电路和常用的大容量动态存储器等都属于通用集成电路, 它们的特点是逻辑功能比较简单, 并且固定不变。由于这些集成电路在组成复杂的电子系统中经常用到, 因此可以看出它们具有很强的通用性。

通常, 采用芯片厂商提供的中小规模集成电路可以组成任何复杂的电子系统, 但是为了减小系统电路的体积、重量、功耗和提高可靠性, 设计人员经常会把设计的系统直接做成一片大规模或超大规模集成电路, 这种为某种专门用途设计的集成电路就称为专用集成电路, 通常也称为 ASIC (Application Specific Integrated Circuit)。例如, 微处理器就是一种常见的专用集成电路, 它只是在某一类计算机中可以使用。可以看出, 这类集成电路的通用性比较差, 而且设计和制造它们的成本较高、周期较长。

一般来说, 专用集成电路又可以分为两种不同的类型, 分别是全定制集成电路 (Full-Custom IC) 和半定制集成电路 (Semi-Custom IC)。其中, 全定制集成电路从电路性能到构成和设计全部是针对某一个特殊的应用来进行设计的; 而半定制集成电路的电路性能也是专门针对于某一应用的, 但是其构成和设计则具有某种程度的通用性, 即把某些已经设计好的单元电路按照设计的需要加以连接而成。可以看出, 半定制集成电路由于设计和制造的成本较低和周期较短, 因此它得到了较为广泛的应用; 而全定制集成电路则经常用于某种性能要求较高、生产批量较大的场合。

对于设计人员来说, 常用的一种半定制集成电路是可编程逻辑器件, 通常也称为 PLD (Programmable Logic Device)。这里, PLD 虽然是作为一种通用集成电路来进行生产的, 但

是它的逻辑功能是由设计人员根据系统设计的具体要求通过相应的器件编程来实现的。另外,由于 PLD 的集成度很高,因此它可以满足大多数数字系统设计的需要。可见,设计人员通过相应的器件编程就可以把设计的系统集成在一片 PLD 上,而不再需要由厂商来设计和制造相应的专用集成电路了,这样便解决了专用集成电路的专用性和成本较高以及开发周期较长的主要矛盾。

从可编程逻辑器件的发展历史上看,可编程逻辑器件经历了 PROM、PLA、PAL、GAL、EPLD 到 CPLD 和 FPGA 的发展过程,它在结构、制造工艺、集成度、逻辑功能、速度和功耗上都有了很大的提高和改进。其中,CPLD 和 FPGA 由于集成度非常高,因此这两种可编程逻辑器件也常被称作高密度可编程逻辑器件。通常,可编程逻辑器件的发展历史如下所述:

1) 20 世纪 70 年代,采用熔丝编程的 PROM 和 PLA 器件可以称作是最早的可编程逻辑器件,它可以根据用户的需要写入相应的信息来完成一定的逻辑功能。但是由于熔丝烧断后不能再次接通,因此这时的器件编程是一次性的,写入后不能进行修改。

2) 20 世纪 70 年代末,MMI 公司率先推出了一种新的可编程逻辑器件——PAL,它也采用双极型工艺制作,熔丝编程方式。从逻辑特性上来看,PAL 器件不如 PLA 器件那样灵活和方便,但是它的成本较低、价格便宜、编程方便、编程器也较为便宜,另外它还具有保密位来防止非法读出,因此它在当时获得了广泛的应用。

3) 20 世纪 80 年代初,Lattice 公司推出了另外一种新型的可编程逻辑器件——GAL,它克服了 PAL 器件存在的缺点,应用起来更加灵活和方便。GAL 器件采用了电可擦除的 CMOS 工艺制作,采用电压信号可以擦除原来的信息而重新进行编程操作。另外,GAL 器件的输出端口设置了可编程的输出逻辑宏单元(Output Logic MacroCell, OLMC),设计人员通过编程可以将 OLMC 设置成不同的工作状态。这样可以采用同一型号的 GAL 器件来实现 PAL 器件的各种电路输出工作方式,大大提高了器件的通用性。

4) 20 世纪 80 年代中期,Xilinx 公司提出了现场可编程的概念,同时生产了世界上第 1 片 FPGA 器件。FPGA 器件的编程方式与 PAL 和 GAL 器件大为不同,它不是通过专门的编程器来完成的。通常,它的设计过程是利用一套专用的设计软件来完成的,最后生成一个用来对 FPGA 器件进行编程的文件。一般来说,FPGA 器件中的编程信息将会存储在专用的 RAM 中,这样上电后外部 RAM 中的编程信息将会传入到器件中的静态 RAM 中,从而实现相应的功能。

同一时期,Altera 公司推出了它的新一代可编程逻辑器件——EPLD,它采用 UVEPROM 和 E<sup>2</sup>PROM 工艺制作,同时集成度要比 PAL 和 GAL 器件高得多,其产品多半属于高密度的可编程逻辑器件。

5) 20 世纪 80 年代末,Lattice 公司又提出了在系统可编程技术,同时推出了一系列具有在系统可编程能力的 CPLD。与 EPLD 相比,CPLD 规模更大,结构更为复杂,能够完成较为复杂的系统功能设计。

6) 20 世纪 90 年代后,可编程逻辑器件进入到了飞速发展的阶段。在生产工艺方面,可编程逻辑器件的线宽越来越小,集成门数越来越多,功耗越来越低;在具体的器件编程方面,E<sup>2</sup>PROM 逐渐取代了 UVEPROM 工艺,同时 FPGA 器件的现场可编程技术和 CPLD 的在系统可编程技术使得可编程逻辑器件在使用上更加方便;在测试技术方面,可编程逻辑器件大多数均可采用边界扫描测试技术,大大加强了器件、电路板和系统的可测试性;在逻辑功能

上, 可编程逻辑器件内部嵌有微处理器, 大大提高了器件的逻辑功能和处理能力。

总之, 可编程逻辑器件的出现大大改变了传统的电路系统设计方法, 从而形成了一种新的基于芯片的设计方法。通过基于芯片的设计方法可以大大减少系统设计中的器件数量、缩小系统体积和降低功率消耗等, 从而提高了设计系统的性能和可靠性。

### 1.1.2 可编程逻辑器件的分类

前面提到过, 可编程逻辑器件经历了 PROM、PLA、PAL、GAL、EPLD 到 CPLD 和 FPGA 的发展过程, 各大公司的可编程逻辑器件层出不穷。通常, 由于历史原因和某些器件厂商的原因, 以往的可编程逻辑器件的命名十分不规范, 因此设计人员很难根据可编程逻辑器件的名称来进行严格的分类。在可编程逻辑器件应用领域, 它具有许多不同的分类方法, 没有一个相对统一的标准。

一般来说, 设计人员经常按照可编程逻辑器件的集成度和结构复杂度、基本结构、逻辑单元以及编程工艺来进行分类, 下面对其进行一下简单介绍。

#### 1. 按照集成度和结构复杂度进行分类

在可编程逻辑器件中, 集成度和结构复杂度是一项非常重要的技术指标, 因此根据集成度和结构复杂度进行分类是一种常用的分类方法。一般来说, 根据可编程逻辑器件的集成度和结构复杂度, 它可以分为低集成度可编程逻辑器件和高集成度可编程逻辑器件。这里将会提出这样一个问题: 低集成度和高集成度可编程逻辑器件的划分标准是什么呢?

通常, 业界往往以 PALCE22V10 或者 GAL22V10 的集成度为标准来划分低集成度和高集成度可编程逻辑器件, 即集成度大于 PALCE22V10 或者 GAL22V10 的可编程逻辑器件视为高集成度可编程逻辑器件, 否则为低集成度可编程逻辑器件。按照这样的划分标准, 可以看出 PROM、PLA、PAL 和 GAL 等器件属于低集成度可编程逻辑器件, 而 EPLD、CPLD 和 FPGA 等则属于高集成度可编程逻辑器件, 如图 1-1 所示。

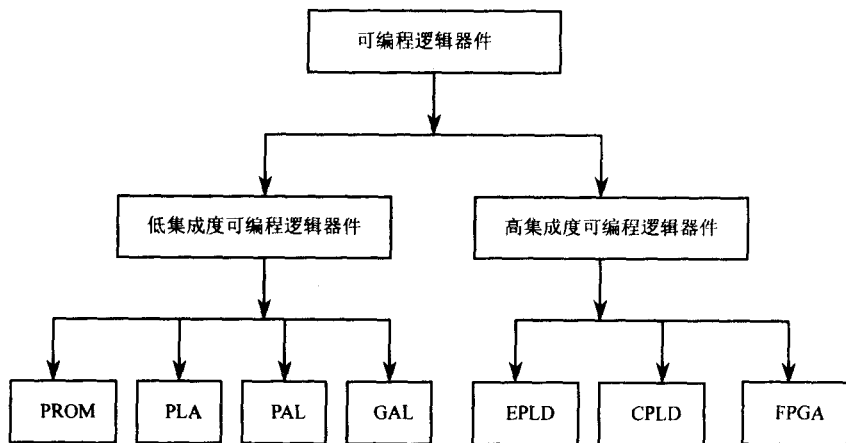


图 1-1 按照集成度和结构复杂度进行分类

由于低集成度可编程逻辑器件易于编程、开发成本较低、价格低廉, 同时对于开发软件的要求也很低, 因此在 20 世纪 80 年代得到了广泛的应用。但是随着科学技术的不断发展和进步, 低集成度可编程逻辑器件在集成度和性能方面的局限性变得非常明显, 而且它的寄存

器、I/O 引脚和时钟资源的数目有限,大大限制了设计的灵活性。随着 EPLD、CPLD 和 FPGA 高集成度可编程逻辑器件的出现,它们已经渐渐退出了历史舞台。

PROM (Programmable Read Only Memory) 是一种与阵列固定、或阵列可编程的与或阵列,通常它是由地址译码器、存储矩阵和输出缓冲器 3 个部分组成的。PROM 采用熔丝工艺编程,因此 PROM 的编程是一次性的,写入之后不可以进行擦除或重写操作。在具体的应用中,PROM 主要用来实现组合逻辑函数,特别是可以方便地实现多输出函数,因此它的应用场合还是相对较多的。随着制造工艺的不断发展,目前常用的 PROM 包括两种,分别是紫外线擦除可编程只读存储器(UltraViolet Erasable Programmable Read Only Memory, UVEPROM)和电擦除可编程只读存储器(Electrically Erasable Programmable Read Only Memory, E<sup>2</sup>PROM)。

PLA (Programmable Logic Array) 是一种与阵列可编程、或阵列也可进行编程的与或阵列。实际上,PLA 就是把 PROM 结构中的地址译码器改成乘积项发生器的一种可编程逻辑器件。PLA 曾经被认为是极有发展前途的可编程逻辑器件,但是由于器件的资源利用率低,现在已经基本退出了应用场合。

PAL (Programmable Array Logic) 是一种与阵列可编程、或阵列固定的与或阵列。与阵列可编程是指它产生的乘积项可以根据设计的需要来进行安排;或阵列固定是指 PAL 每个输出所分配的乘积项是固定不变的,不仅每个输出所具有的乘积项数目是固定的,而且固定数目的乘积项的位置也是固定的。由于 PAL 具有多种输出结构形式,因而它的型号很多,应用也较为广泛。

GAL (Generic Array Logic) 是一种在 PAL 器件的基础上发展起来的新型可编程逻辑器件,从基本结构上看,它仍然属于与或阵列结构的可编程逻辑器件。与 PAL 器件相比,它采用了 E<sup>2</sup>CMOS 工艺,可以进行电擦除和重写操作;另外 GAL 器件采用了一个可编程的逻辑宏单元输出(OLMC),通过对 OLMC 进行配置就可以得到多种形式的输出和反馈。由于 GAL 器件对 PAL 器件具有 100% 的兼容性,因此 GAL 器件几乎完全替代了 PAL 器件。

通常,可编程逻辑器件 PAL 和 GAL 非常适用于小规模的逻辑设计,例如选用一片或者几片简单的 PAL 或 GAL 芯片就可以代替印制电路板中的多片中小规模的通用 IC 芯片,从而实现缩减原来的组合电路和时序电路,目的是减少印制电路板中的连线,缩小电路板的体积和重量,同时提高其可靠性。

同样按照上面的划分标准,高集成度可编程逻辑器件主要包括 EPLD、CPLD 和 FPGA 等 3 种器件。由于这些高集成度可编程逻辑器件规模大、结构复杂,能够完成大多数比较复杂的电子系统设计,因此它们获得极其广泛的应用。目前,EPLD、CPLD 和 FPGA 不但可以解决设计系统的速度快、性能高、容量大、体积小、微功耗和高可靠性等问题,而且其开发周期短、开发软件成本低、芯片价格不断降低,因此它们获得了越来越广阔的应用市场。

20 世纪 80 年代中期,Altera 公司推出了一种新型的、电可擦除的可编程逻辑器件,它被称作 EPLD (Erasable Programmable Logic Device)。这种新型的可编程逻辑器件采用 UVEPROM 和 E<sup>2</sup>PROM 工艺制作,同时集成度要比 PAL 和 GAL 器件高得多,其产品多半属于高密度的可编程逻辑器件。通常 EPLD 的基本逻辑单位是宏单元,宏单元是由可编程的与或阵列、可编程寄存器和可编程 I/O 这 3 部分组成的。宏单元和整个器件的逻辑功能均由 EPROM 来定义和进行规划。

一般来说,世界著名的半导体器件公司,如 Altera、Xilinx、AMD、Lattice 和 Atmel 等公司均生产 EPLD 产品,但是结构差异很大。由于 EPLD 的内部互连能力较弱,在 20 世纪 80 年代末期受到了另外一种新兴的可编程逻辑器件 FPGA 的冲击,应用市场越来越小。直到 20 世纪 90 年代 EPLD 的改进器件 CPLD 出现以后,这种情况才有所改进。

CPLD (Complex Programmable Logic Device) 通常被称作复杂可编程逻辑器件,它是 EPLD 的改进形式,规模更大、结构更为复杂,因此它属于大规模集成电路的范围。一般情况下,CPLD 至少包含了 3 种结构:可编程逻辑宏单元、可编程 I/O 单元和可编程内部连线。此外,有些 CPLD 内部还集成了 RAM、FIFO 或双口 RAM 等存储器。其中,Lattice 公司的 ispLSI/pLSI1000 系列、AMD 公司的 MACH5 系列、Xilinx 公司的 XC9500 系列、Altera 公司的 MAX7000 系列是 CPLD 的代表产品。

通常,CPLD 在集成度和结构上呈现出来的特点是具有更大规模的与或阵列,增加了大量的逻辑宏单元和布线资源,触发器的数目也明显增多。另外,由于 CPLD 内部采用固定长度的金属线进行各种逻辑块的互连,因此其设计的电路具有时间可预测性,从而避免了分段式互连结构时序不可完全预测的缺点。可见,常见的数字电路如高速译码器、多位计数器、寄存器和移位寄存器、时序有限状态机、网络适配器和总线控制器等较大规模的逻辑设计均可以通过 CPLD 来实现。CPLD 由于具有同 FPGA 器件相似的集成度和易用性,而且它在速度和时序预测上还有一定的优势,因此,CPLD 能够在可编程逻辑器件技术的竞争中与 FPGA 器件并驾齐驱,成为两支领导可编程逻辑器件技术发展的力量之一。

近年来,各大器件厂商又推出了一些更大规模的 CPLD,从而使得 CPLD 的应用更加深入和广泛。Lattice 公司的 ispLSI/pLSI3256 集成度已经达到了 14000 个等效的 PLD 门,触发器数量也达到了 480 个;另外,ispLSI6000 系列的集成度达到了 25000 个等效的 PLD 门,它具有 320 个宏单元。Altera 公司推出的 MAX9000 系列最高集成度可以达到 24000 个等效的 PLD 门,逻辑宏单元数目可以达到 1024 个。因此,具有复杂算法的数字滤波器、傅里叶变换等数字信号处理单元的逻辑设计可以通过这些新型的具有更高集成度的 CPLD 来实现。

FPGA(Field Programmable Gate Array)通常被称作现场可编程门阵列,它最初是由 Xilinx 公司提出的。自从 20 世纪 80 年代中期 Xilinx 公司生产了世界上第 1 片 FPGA 器件后,这种新型的可编程逻辑器件便以其优越的性能获得了业界设计人员的青睐,应用迅速普及起来。与其他可编程逻辑器件相比,FPGA 器件的规模可以做得很大,一片芯片集成几十万乃至上百万的门是常有的事;另外它的逻辑功能单元绝不限于逻辑门,而且可以具有较为复杂的逻辑功能,从而使得整个芯片的功能大大加强。

FPGA 器件的功能由逻辑结构的配置数据决定,工作时这些配置数据存放在芯片内部的 SRAM 或者熔丝图上。FPGA 器件的编程方式与 PAL 和 GAL 器件大为不同,它不是通过专门的编程器来完成的。通常,它的设计过程是利用一套专用的设计软件来完成的,最后生成一个用来对 FPGA 器件进行编程的文件。FPGA 器件的配置数据可以存放在片外的 EPROM 或其他存储体上,设计人员可以控制芯片的加载过程并现场修改器件的逻辑功能,即所谓现场编程。可见,采用 FPGA 器件可以十分方便地实现相应的逻辑功能设计。

一般来说,Xilinx 公司 FPGA 器件的一般结构如图 1-2 所示。通过图 1-2 可以看出,一个 FPGA 器件是由 3 个部分组成的,分别是可编程输入/输出单元(Input/Output Block, IOB)、可编程逻辑单元(Configurable Logic Block, CLB)和可编程内部连接(Programmable

Interconnect, PI) 单元。其中, 可编程输入/输出单元的功能是用来提供内部逻辑阵列与外部的接口; 可编程逻辑单元的功能是用来提供设计人员需要的逻辑功能; 可编程内部连接单元的功能是用来进行 CLB 和 IOB 的内部连接, 从而实现所需要的连接功能。

通过上面的描述, 这里可以将 FPGA 器件的一般结构简单描述为: 可编程逻辑单元在器件中排列为阵列, 周围环绕着可编程内部连接单元, 可编程输入/输出单元则分布在四周的管脚上。由于 FPGA 器件的 3 个组成部分都是可以进行编程操作的, 因此改变 FPGA 器件的功能除了改变各 CLB 之间的连接外, 也可以通过改变各个 CLB 所实现的逻辑功能来实现。

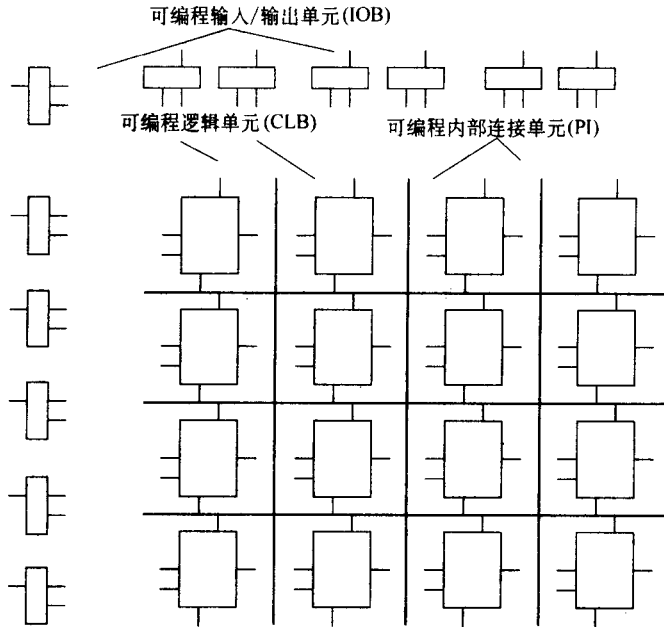


图 1-2 FPGA 器件的一般结构

目前, CPLD 和 FPGA 器件已经成为应用可编程逻辑器件的主流产品, 它们各自具有自己的特点和逻辑功能, 两者之间具有互补功能, 因此两者之间能够相互发展, 成为两支领导可编程逻辑器件技术发展的力量。通常, CPLD 和 FPGA 器件的差别主要体现在两个方面:

(1) CPLD 和 FPGA 器件内部逻辑单元的电路工艺不同 一般来说, CPLD 是从 GAL 和 EPLD 技术逐渐发展演变而来的, 它采用的关键技术是 EPROM 和 E<sup>2</sup>PROM 型的 CMOS 工艺, 因此器件可以长期保存编程数据, 同时也可以进行擦除或者重写操作。FPGA 器件采用的关键技术是基于 SRAM 的 CMOS 工艺, 因此它的单元电路逻辑需要上电进行配置, 掉电后配置数据丢失, 相应的芯片功能也随之消失。

由于 CPLD 和 FPGA 器件内部逻辑单元的电路工艺不同, 因此导致了它们的编程方式也大不相同: CPLD 一般采用在系统可编程 (In-System Programmability, ISP) 的方法来进行器件编程, 而 FPGA 器件一般则采用在线重配置 (In-Circuit Reconfigurability, ICR) 的方法来进行器件编程。

(2) CPLD 和 FPGA 器件内部连线的连接方式不同 通常, CPLD 采用了固定长度的金属线进行各种逻辑块的互连, 因此其设计的电路具有时间可预测性, 从而避免了分段式互连