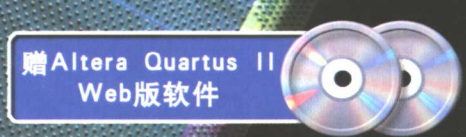
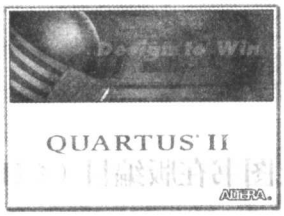
Altera公司推荐FPGA/CPLD 培训教材

Altera FPGA/CPLD设计（高级篇）

EDA先锋工作室 吴继华 王诚 编著

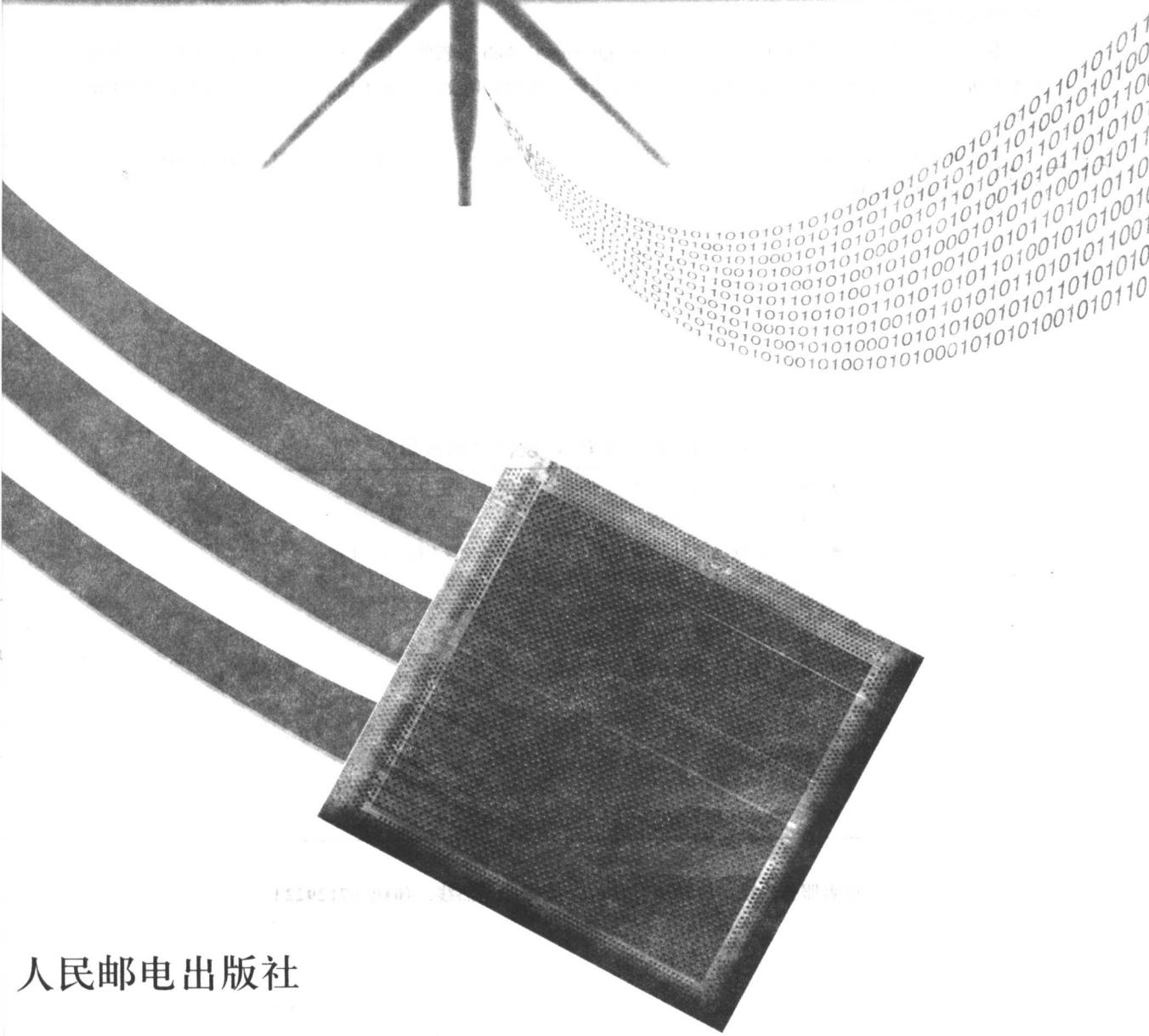


人民邮电出版社
POSTS & TELECOM PRESS



Altera FPGA/CPLD设计（高级篇）

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

Altera FPGA/CPLD 设计. 高级篇 / 吴继华，王诚编著. —北京：人民邮电出版社，2005.7
ISBN 7-115-13500-2

Ⅰ. A... Ⅱ. ①吴...②王... Ⅲ. 可编程序逻辑器件 Ⅳ. TP332.1

中国版本图书馆 CIP 数据核字（2005）第 063471 号

内 容 提 要

本书结合作者多年工作经验，深入地讨论了 Altera FPGA/CPLD 的设计、优化技巧。在讨论 FPGA/CPLD 设计指导原则的基础上，介绍了 Altera 器件的高级应用；引领读者学习逻辑锁定设计工具，详细讨论了时序约束与静态时序分析方法；结合实例讨论如何进行设计优化，介绍了 Altera 的可编程器件的高级设计工具与系统级设计技巧。

本书附带两张光盘：光盘 1 中收录了 Altera Quartus II Web 版软件，读者可以安装使用；光盘 2 中收录了本书所有实例的完整工程、源代码、详细操作步骤和使用说明文件，便于读者边学边练，提高实际应用能力。

本书可作为高等院校通信工程、电子工程、计算机、微电子与半导体等专业的教材，也可作为硬件工程师和 IC 工程师的实用工具书。

Altera FPGA/CPLD 设计（高级篇）

◆ 编　著　EDA 先锋工作室　吴继华　王　诚
责任编辑　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：22
字数：532 千字　　2005 年 7 月第 1 版
印数：1 – 6 000 册　　2005 年 7 月北京第 1 次印刷

ISBN 7-115-13500-2/TP・4708

定价：45.00 元（附 2 张光盘）

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

关 于 本 书

内容和特点

FPGA/CPLD、DSP 和 CPU 被称为未来数字电路系统的 3 块基石，也是目前硬件设计研究的热点。与传统电路设计方法相比，FPGA/CPLD 具有功能强大，开发过程投资小、周期短，可反复编程修改，保密性能好，开发工具智能化等特点，特别是随着电子工艺的不断改进，低成本 FPGA/CPLD 器件推陈出新，这一切促使 FPGA/CPLD 成为当今硬件设计的首选方式之一。可以说 FPGA/CPLD 设计技术是当今高级硬件工程师与 IC 工程师的必备技能。

我国可编程逻辑器件设计技术落后于国外，目前立足工程实践，系统地介绍最新 FPGA/CPLD 设计工具的中文书籍较为贫乏。在这种情况下，为了满足广大工科在校生了解业界流行的高效 FPGA/CPLD 设计技术的需要，提高硬件工程师与 IC 工程师的工程实践技巧，我们编写了《Altera FPGA/CPLD 设计（基础篇）》和《Altera FPGA/CPLD 设计（高级篇）》。

《Altera FPGA/CPLD 设计（高级篇）》（以下简称“本书”）涵盖了 Altera 主流 FPGA/CPLD 硬件结构与特点，详尽地讨论了 Quartus II 与第三方 EDA 工具的设计方法，系统地阐述了 Altera 可编程逻辑设计优化技术。

本书共 7 章，各章内容简要介绍如下。

- 第 1 章　探讨了可编程逻辑设计的基本原则和常用思想与技巧，并详细地讨论了 Altera 推荐的 Coding Style。
- 第 2 章　分别介绍了 Altera 器件的时钟管理、片内存储器、数字信号处理、片外高速存储器、差分接口与 DPA、高速串行收发器等高级硬件特性与应用方法。
- 第 3 章　重点介绍 LogicLock 设计方法。
- 第 4 章　介绍时序分析的基本概念与常用约束方法的基础上，讨论了高级时序分析的技巧。
- 第 5 章　介绍资源利用率优化、I/O 时序优化、最高频率优化等设计优化的实用技术，并讨论了如何使用 DSE 进行优化的方法。
- 第 6 章　介绍 Tcl 脚本、HardCopy、Nios II 处理器、DSP Builder 等高级软工具的使用方法。
- 第 7 章　重点讨论了信号完整性、电源设计、功耗分析与热设计、SERDES 与高速系统设计等系统级设计技巧。

本书的主要特点介绍如下。

- 全面系统：涵盖了 Altera 软、硬件设计技术，基础与高级设计工具，全面系统地论述了 Altera 可编程设计技术。
- 实用价值高：本书的作者都有丰富的 FPGA/CPLD、数字 ASIC 设计经验，本书立足于工程实践的需要，对工程设计有显著的指导意义。
- 内容新颖：本书的作者长期工作在可编程逻辑设计的最前沿，与 FPGA 器件制造公司与 EDA 软件设计公司联系紧密，所以有幸能够在第一时间内使用最新版本的 FPGA/CPLD 设计工具。书中涉及的所有工具均根据较新资料撰写，使图书介绍的内容新颖。

- 剖析深刻：书中对 FPGA/CPLD 设计的基本原理、方法有较为详尽的论述，对各种设计工具的介绍并不局限于操作方法，而是结合作者多年的工作经验与心得，从较深的层面对各个工具的特点进行剖析。

读者对象

本书可作为高等院校通信工程、电子工程、计算机、微电子与半导体学等理工专业的教材，也可作为硬件工程师和 IC 工程师的实用工具书。

附盘内容

配套光盘提供了书中所有示例的完整工程文件、设计源文件和说明文件。

每个工程示例都包括了该工程的项目文件、源文件、报告文件和生成结果等文件，读者可以用 Quartus II 或相应的软件直接打开。设计源文件根据设计输入类型分为源代码或原理图等。请读者将设计源文件拷贝到计算机硬盘上，并按照书中的操作步骤自行操作练习。示例说明文件包含了示例的详细信息和操作指南。

另外，经 Altera 公司特别授权，光盘中收录了 Altera Quartus II Web 版软件、相关器件手册和技术文档。Altera Quartus II Web 版软件支持部分 MAX® II、Stratix™ II、Stratix、Cyclone™、APEX™ 20KE、APEX II、ACEX™、Excalibur™、FLEX® 10KE、FLEX 10K®、FLEX 10KA、FLEX 6000、MAX 7000S、MAX 7000B、MAX 7000AE 和 MAX 3000A 器件。请读者按照光盘中的说明文件安装 Quartus II Web 版软件，并访问 www.altera.com.cn 网址，进入 Home > Support > Design Software > Licensing > Quartus II Web Edition Licensing 链接，申请软件的 License。在此，我们对 Altera 公司的强力支持表示真挚的感谢！

本书约定

为了方便读者阅读，书中设计了 4 个小图标，它们代表的含义如下。

 行家指点：用于介绍使用经验和心得，或罗列重要的概念。

 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、3 章由王诚编写，第 2、5、6 章由吴继华编写，第 4、7 章由吴继华、王诚合作编写。

Altera 应用工程经理郭晶先生、李健先生，资深系统工程师林斌先生，资深高速技术专家韦俊伟先生对全书进行了审校。Altera 大中华区销售经理赵典锋先生，中国区销售经理钟屹先生，客户经理王刚先生、陈卫中先生，亚太区应用工程总监罗炜亮先生对本书提出了许多建设性意见，并给予作者多方面的帮助。Altera 亚太区市场总监梁乐观先生，市场经理刘芳女士，积极参与本书出版工作的组织与协调，在此一并表示衷心的感谢。在这里要特别感

谢 Altera 副总裁李彬先生在百忙之中亲自为本书撰写序言。感谢所有关心并支持本书的同仁佳友！

感谢您选择了本书，如果您对书中内容有任何困惑和建议，请与我们联系。

本书互联网支持：EDACN 专业论坛，http://www.edacn.net；

电子函件：altera_book@edacn.net（作者），liyongtao@ptpress.com.cn（责任编辑）。

如果您需要得到 Altera 更全面的服务与技术支持，请访问 http://www.altera.com.cn。

EDA 先锋工作室

2005 年 3 月

序

Altera 公司的总部位于美国加州的圣何塞，并在全球的 14 个国家中拥有近 2000 名员工。作为可编程单芯片系统（SOPC）方案的先行者，Altera 将可编程逻辑技术、软件工具、IP 和技术服务结合在一起，为全球约 14000 个客户提供极具价值的可编程系统解决方案。

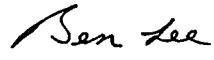
自 20 年前发明世界上第一个可编程逻辑器件开始，Altera 公司秉承了创新的传统。新产品系列将可编程逻辑的内在优势——灵活性、产品及时面市、更高级性能以及集成化结合在一起，专为满足当今大范围的系统需求而开发设计。

Altera 可编程解决方案包括：

- 业内先进的 FPGA、CPLD 和结构化 ASIC 技术；
- 全面内嵌的软件开发工具；
- 优秀的 IP 内核；
- 可定制嵌入式处理器；
- 现成的开发包。

我理解，如今的 FPGA/CPLD 设计工程师们正苦于没有完备的设计方法学来指导，没有系统的设计技术帮助他们将设计做得最优化。加上 FPGA/CPLD 属于前沿技术，大篇的英文资料也让国内工程师无法迅速掌握最新的技术动态和设计技巧。

在这里，我向广大读者郑重推荐《Altera FPGA/CPLD 设计（基础篇）》和《Altera FPGA/CPLD 设计（高级篇）》这两本书。这两本书不仅介绍了 Altera 传统的 PLD 技术，还介绍了 Altera 的可编程片上系统（System On a Programmable Chip）的设计思想，非常成功的嵌入式处理器 Nios 和 Nios II，以及 Altera 领先的结构化 ASIC 技术——HardCopy。这两本书与众不同之处是，它通过介绍 Altera 的器件和设计工具（Quartus II），引申出可编程逻辑器件的设计思想和高级设计技巧。同时，在书中包含了丰富的设计实例，使读者能够在完成书中理论学习的同时，通过实践深入掌握，养成良好的设计习惯。



李彬（Ben Lee）
Altera 亚太区副总裁
Altera International Limited
2005 年

工作室简介

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国内外著名电子、通信、半导体行业的资深研发人员、技术支持、市场销售、信息咨询和管理人员。

该工作室的宗旨为：联合国内外 EDA 设计人才，培养 EDA 设计专业队伍，推动我国 EDA 技术的发展。该工作室的主要工作范围为：创作 EDA 相关技术丛书，培养国内 EDA 设计专业人才，电子产品设计研发。EDA 先锋工作室擅长的技术领域有：FPGA/CPLD 设计，ASIC 设计，高速 PCB 设计，嵌入式系统设计等。EDA 先锋工作室愿意与各界有识之士开展积极的合作！

网站支持

为了配合学习本书，EDA 先锋工作室在“EDA 专业论坛”（http://www.edacn.net）上开办了《Altera FPGA/CPLD 设计（基础篇）》与《Altera FPGA/CPLD 设计（高级篇）》的讨论园地，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，藉此与读者共同切磋、互相提高。网站提供本书光盘中附带例子等相关资料的下载服务，并介绍 EDA 先锋工作室所编写图书的出版动态。

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。

电子函件：altera_book@edacn.net；网站地址：http://www.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

目　　录

第 1 章　可编程逻辑设计指导原则 1
1.1　可编程逻辑基本设计原则 1
1.1.1　面积和速度的平衡与互换原则 1
1.1.2　硬件原则 11
1.1.3　系统原则 13
1.1.4　同步设计原则 17
1.2　可编程逻辑常用设计思想与技巧 19
1.2.1　乒乓操作 19
1.2.2　串并转换 21
1.2.3　流水线操作 22
1.2.4　异步时钟域数据同步 23
1.3　Altera 推荐的 Coding Style 27
1.3.1　Coding Style 的含义 27
1.3.2　结构层次化编码（Hierarchical Coding） 27
1.3.3　模块划分的技巧（Design Partitioning） 29
1.3.4　组合逻辑的注意事项 30
1.3.5　时钟设计的注意事项 33
1.3.6　全局异步复位资源 39
1.3.7　判断比较语句 case 和 if...else 的优先级 39
1.3.8　使用 Pipelining 技术优化时序 40
1.3.9　模块复用与 Resource Sharing 40
1.3.10　逻辑复制 42
1.3.11　香农扩展运算 44
1.3.12　信号敏感表 46
1.3.13　状态机设计的一般原则 47
1.3.14　Altera Megafunction 资源的使用 49
1.3.15　三态信号的设计 49
1.3.16　加法树的设计 50
1.4　小结 52
1.5　问题与思考 52
第 2 章　Altera 器件高级特性与应用 53
2.1　时钟管理 53
2.1.1　时序问题 53
2.1.2　锁相环应用 60
2.2　片内存储器 69

2.2.1 RAM 的普通用法 69
2.2.2 RAM 用做移位寄存器 73
2.2.3 RAM 实现固定系数乘法 74
2.3 数字信号处理 75
2.3.1 DSP 块资源 75
2.3.2 工具支持 79
2.3.3 典型应用 79
2.4 片外高速存储器 80
2.4.1 存储器简介 80
2.4.2 ZBT SRAM 接口设计 83
2.4.3 DDR SDRAM 接口设计 85
2.4.4 QDR SRAM 接口设计 99
2.4.5 DDR2、QDR II 和 RLDRAM II 100
2.4.6 软件支持和应用实例 100
2.5 高速差分接口和 DPA 102
2.5.1 高速差分接口的需求 102
2.5.2 器件的专用资源 102
2.5.3 动态相位调整电路（DPA） 109
2.5.4 软件支持和应用实例 112
2.6 高速串行收发器 115
2.7 小结 117
2.8 问题与思考 117
第 3 章 LogicLock 设计方法 119
3.1 LogicLock 设计方法简介 119
3.1.1 LogicLock 设计方法的目标 120
3.1.2 LogicLock 设计流程 122
3.1.3 LogicLock 设计方法支持的器件族 122
3.2 LogicLock 区域 122
3.2.1 Region 的类型与常用属性值 123
3.2.2 Region 的创建方法 124
3.2.3 Region 的层次结构 129
3.2.4 指定 Region 的逻辑内容 130
3.3 LogicLock 的约束注意事项 132
3.3.1 约束优先级 132
3.3.2 规划 LogicLock 区域 133
3.3.3 向 LogicLock 区域中布置器件特性 133
3.3.4 虚拟引脚（Virtual Pins） 134
3.4 反标注布线信息 135

3.4.1 导出反标注布线信息 136
3.4.2 导入反标注布线信息 138
3.5 LogicLock 设计方法支持的 Tcl Scripts 138
3.6 Quartus II 基于模块化的设计流程 139
3.7 小结 149
3.8 问题与思考 149
第 4 章 时序约束与时序分析 151
4.1 时序约束与时序分析基础 151
4.1.1 周期与最高频率 152
4.1.2 利用 Quartus II 工具分析设计 154
4.1.3 时钟建立时间 157
4.1.4 时钟保持时间 158
4.1.5 时钟输出延时 158
4.1.6 引脚到引脚的延迟 159
4.1.7 Slack 159
4.1.8 时钟偏斜 160
4.1.9 Quartus II 时序分析工具和优化向导 160
4.2 设置时序约束的常用方法 161
4.2.1 指定全局时序约束 162
4.2.2 指定个别时钟约束 166
4.3 高级时序分析 174
4.3.1 时钟偏斜 174
4.3.2 多时钟域 176
4.3.3 多周期约束 176
4.3.4 伪路径 183
4.3.5 修正保持时间违例 185
4.3.6 异步时钟域时序分析 186
4.4 最小化时序分析 187
4.5 使用 Tcl 工具进行高级时序分析 188
4.6 小结 189
4.7 问题与思考 189
第 5 章 设计优化 191
5.1 解读设计 191
5.1.1 内部时钟域 192
5.1.2 多周期路径和伪路径 193
5.1.3 I/O 接口的时序要求 194
5.1.4 平衡资源的使用 194
5.2 设计优化的基本流程和首次编译 195

5.2.1 设计优化基本流程 195
5.2.2 首次编译的约束和设置 196
5.2.3 查看编译报告 198
5.3 资源利用优化 200
5.3.1 设计代码优化 201
5.3.2 资源重新分配 201
5.3.3 解决互连资源紧张的问题 203
5.3.4 逻辑综合面积优化 203
5.3.5 网表面积优化 207
5.3.6 寄存器打包 209
5.3.7 Quartus II 中的资源优化顾问 211
5.4 I/O 时序优化 211
5.4.1 执行时序驱动的编译 211
5.4.2 使用 IOE 中的触发器 212
5.4.3 可编程输入输出延时 215
5.4.4 使用锁相环对时钟移相 217
5.4.5 其他 I/O 时序优化方法 218
5.5 最高时钟频率优化 219
5.5.1 设计代码优化 219
5.5.2 逻辑综合速度优化 225
5.5.3 布局布线器设置 227
5.5.4 网表优化和物理综合 228
5.5.5 使用 LogicLock 对局部进行优化 233
5.5.6 位置约束、手动布局和反标注 234
5.5.7 Quartus II 中的时序优化顾问 235
5.6 使用 DSE 工具优化设计 236
5.6.1 为什么需要 DSE 236
5.6.2 什么是 DSE，如何使用 236
5.7 如何减少编译时间 238
5.8 设计优化实例 239
5.9 小结 242
5.10 问题与思考 243
第 6 章 Altera 其他高级工具 245
6.1 命令行与 Tcl 脚本 245
6.1.1 命令行脚本 246
6.1.2 Tcl 脚本 250
6.1.3 使用命令行和 Tcl 脚本 254
6.2 HardCopy 流程 255

6.2.1 结构化 ASIC 255
6.2.2 HardCopy 器件 258
6.2.3 HardCopy 设计流程 260
6.3 基于 Nios II 处理器的嵌入式系统设计 263
6.3.1 Nios II 处理器系统 263
6.3.2 Avalon 交换结构 266
6.3.3 使用 SOPC Builder 构建系统硬件 269
6.3.4 Nios II IDE 集成开发环境 272
6.3.5 Nios II 系统典型应用 278
6.4 DSP Builder 工具 281
6.4.1 DSP Builder 设计流程 281
6.4.2 与 SOPC Builder 一起构建系统 284
6.5 小结 285
6.6 问题与思考 285
第 7 章 FPGA 系统级设计技术 287
7.1 信号完整性及常用 I/O 电平标准 287
7.1.1 信号完整性 287
7.1.2 单端标准 292
7.1.3 差分标准 296
7.1.4 伪差分标准 299
7.1.5 片上终端电阻 299
7.2 电源完整性设计 300
7.2.1 电源完整性 300
7.2.2 同步翻转噪声 301
7.2.3 非理想回路 304
7.2.4 低阻抗电源分配系统 307
7.3 功耗分析和热设计 311
7.3.1 功耗的挑战 311
7.3.2 FPGA 的功耗 311
7.3.3 热设计 313
7.4 SERDES 与高速系统设计 315
7.4.1 SERDES 的基本概念 316
7.4.2 Altera Stratix GX 和 Stratix II 中 SERDES 的基本结构 319
7.4.3 典型高速系统应用框图举例 324
7.4.4 高速 PCB 设计注意事项 329
7.5 小结 331
7.6 问题与思考 331
附录 配套光盘使用说明 333

第1章　可编程逻辑设计指导原则

本章旨在探讨可编程逻辑设计的一些基本规律。FPGA/CPLD 的设计规律与方法是一个非常大的课题，在此不可能面面俱到，希望通过本章提纲携领的粗浅介绍，引起读者们的注意。如果大家能在日后的工作实践中，不断积累，有意识地用 FPGA/CPLD 的基本设计原则、设计思想作为指导，将取得事半功倍的效果！

本章主要内容如下：

- 可编程逻辑基本设计原则；
- 可编程逻辑常用设计思想与技巧；
- Altera 推荐的 Coding Style。

1.1　可编程逻辑基本设计原则

可编程逻辑设计有许多内在规律可循，总结并掌握这些规律对于较深刻地理解可编程逻辑设计技术非常重要。本章从 FPGA/CPLD 的基本概念出发，总结出 4 个基本设计原则，这些指导原则范畴非常广，希望读者不仅仅是学习它们，更重要的是理解它们，并在今后的工作实践中充实、完善它们。

(1) 面积和速度的平衡与互换原则。

面积和速度的平衡与互换原则提出了 FPGA/CPLD 设计的两个基本目标，并探讨了这两个目标的对立统一的矛盾关系。

(2) 硬件原则。

硬件原则重点在于提醒读者转化软件设计的思路，理解 HDL 语言设计的本质。

(3) 系统原则。

系统原则希望读者能够通过全局、整体上把握设计，从而提高设计质量，优化设计效果。

(4) 同步设计原则。

同步设计原则是设计时序稳定的基本要求，也是高速 PLD 设计的通用法则。

1.1.1　面积和速度的平衡与互换原则

这里“面积”是指一个设计所消耗 FPGA/CPLD 的逻辑资源数量：对于 FPGA，可以用所消耗的触发器（FF）和查找表（LUT）来衡量；对于 CPLD，常用宏单元（MC）衡量。用设计所占用的等价逻辑门数来衡量设计所消耗 FPGA/CPLD 的逻辑资源数量也是一种常见

的衡量方式。“速度”指设计在芯片上稳定运行时所能够达到的最高频率，这个频率由设计的时序状况决定，与设计满足的时钟周期、PAD to PAD Time、Clock Setup Time、Clock Hold Time 和 Clock-to-Output Delay 等众多时序特征量密切相关。面积（Area）和速度（Speed）这两个指标贯穿着 FPGA/CPLD 设计的始终，是设计质量评价的终极标准。这里我们就讨论一下设计中关于面积和速度的基本原则：面积和速度的平衡与互换。

面积和速度是一对对立统一的矛盾体。要求一个设计同时具备设计面积最小，运行频率最高，这是不现实的。科学的设计目标应该是在满足设计时序要求（包含对设计最高频率的要求）的前提下，占用最小的芯片面积，或者在所规定的面积下，使设计的时序余量更大，频率更高。这两种目标充分体现了面积和速度的平衡思想。关于面积和速度的要求，我们不应该简单地理解为工程师水平的提高和设计完美性的追求，而应该认识到它们是和产品的质量和成本直接相关的。如果设计的时序余量比较大，运行的频率比较高，则意味着设计的健壮性更强，整个系统的质量更有保证；另一方面，设计所消耗的面积更小，则意味着在单位芯片上实现的功能模块更多，需要的芯片数量更少，整个系统的成本也随之大幅度削减。

作为矛盾的两个组成部分，面积和速度的地位是不一样的。相比之下，满足时序、工作频率的要求更重要一些，当两者冲突时，采用速度优先的准则。

面积和速度的互换是 FPGA/CPLD 设计的一个重要思想。从理论上讲，一个设计如果时序余量较大，所能跑的频率远远高于设计要求，那么就能通过功能模块复用减少整个设计消耗的芯片面积，这就是用速度的优势换面积的节约；反之，如果一个设计的时序要求很高，普通方法达不到设计频率，那么一般可以通过将数据流串并转换，并行复制多个操作模块，对整个设计采取“乒乓操作”和“串并转换”的思想进行处理，在芯片输出模块处再对数据进行“并串转换”。从宏观上看，整个芯片满足了处理速度的要求，这相当于用面积复制换取速度的提高。面积和速度互换的具体操作技巧很多，比如模块复用、“乒乓操作”、“串并转换”等，需要大家在日后工作中不断积累。下面举例说明如何使用“速度换面积”和“面积换速度”。

【例1-1】 如何使用“速度的优势换取面积的节约”？

在 WCDMA（宽带码分多址）系统中，使用到了快速哈达码（FHT）运算，FHT 由步相同的算法完成，如图 1-1 所示。

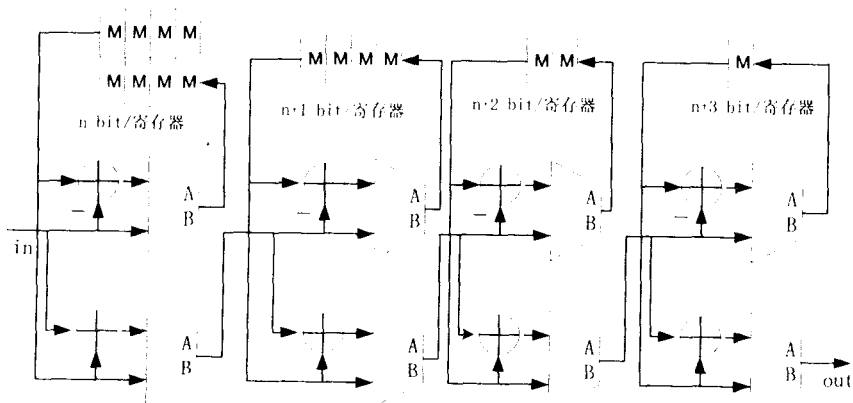


图1-1　FHT 原理图

FHT 的单步算法如下：

$$Out[2i] = In[2i] + In[2i+8]; i = 0-7;$$
$$Out[2i+1] = In[2i+1] - In[2i+1+8]; i = 0-7$$

考虑流水线式数据处理的要求，最自然的设计方法就是设计不同端口宽度的 4 个单步 FHT，并用将这 4 个单步模块串联起来，从而完成数据流的流水线处理。该 FHT 实现方式的代码如下：

```
    //该模块是 FHT 的顶层，调用 4 个不同端口宽度的单步 FHT 模块，完成整个 FHT 算法
    module
fhtpart(Clk,Reset,FhtStarOne,FhtStarTwo,FhtStarThree,FhtStarFour,
            I0,I1,I2,I3,I4,I5,I6,I7,I8,
            I9,I10,I11,I12,I13,I14,I15,
            Out0,Out1,Out2,Out3,Out4,Out5,Out6,Out7,Out8,
            Out9,Out10,Out11,Out12,Out13,Out14,Out15);
    input Clk;    //设计的主时钟
    input Reset;   //异步复位
    input FhtStarOne,FhtStarTwo,FhtStarThree,FhtStarFour; //4 个单步算法的时序控制信号
    input [11:0] I0,I1,I2,I3,I4,I5,I6,I7,I8;
    input [11:0] I9,I10,I11,I12,I13,I14,I15;                    //FHT 的 16 个输入
    output [15:0] Out0,Out1,Out2,Out3,Out4,Out5,Out6,Out7;
    output [15:0] Out8,Out9,Out10,Out11,Out12,Out13,Out14,Out15;  //FHT 的 16 个输出

    //第 1 次 FHT 单步运算的输出
    wire [12:0] m0,m1,m2,m3,m4,m5,m6,m7,m8,m9;
    wire [12:0] m10,m11,m12,m13,m14,m15;

    //第 2 次 FHT 单步运算的输出
    wire [13:0] mm0,mm1,mm2,mm3,mm4,mm5,mm6,mm7,mm8,mm9;
    wire [13:0] mm10,mm11,mm12,mm13,mm14,mm15;

    //第 3 次 FHT 单步运算的输出
    wire [14:0] mmm0,mmm1,mmm2,mmm3,mmm4,mmm5,mmm6,mmm7,mmm8,mmm9;
    wire [14:0] mmm10,mmm11,mmm12,mmm13,mmm14,mmm15;

    //第 4 次 FHT 单步运算的输出
    wire [15:0] Out0,Out1,Out2,Out3,Out4,Out5,Out6,Out7,Out8,Out9;
    wire [15:0] Out10,Out11,Out12,Out13,Out14,Out15;
```

```
    //第 1 次 FHT 单步运算
    fht_unit1 fht_unit1(Clk,Reset,FhtStarOne,
              I0,I1,I2,I3,I4,I5,I6,I7,I8,
              I9,I10,I11,I12,I13,I14,I15,
              m0,m1,m2,m3,m4,m5,m6,m7,m8,
              m9,m10,m11,m12,m13,m14,m15
              );

    //第 2 次 FHT 单步运算
    fht_unit2 fht_unit2(Clk,Reset,FhtStarTwo,
              m0,m1,m2,m3,m4,m5,m6,m7,m8,
              m9,m10,m11,m12,m13,m14,m15,
              mm0,mm1,mm2,mm3,mm4,mm5,mm6,mm7,mm8,
              mm9,mm10,mm11,mm12,mm13,mm14,mm15
              );
    //第 3 次 FHT 单步运算
    fht_unit3 fht_unit3(Clk,Reset,FhtStarThree,
              mm0,mm1,mm2,mm3,mm4,mm5,mm6,mm7,mm8,
              mm9,mm10,mm11,mm12,mm13,mm14,mm15,
              mmm0,mmm1,mmm2,mmm3,mmm4,mmm5,mmm6,mmm7,mmm8,
              mmm9,mmm10,mmm11,mmm12,mmm13,mmm14,mmm15
              );
    //第 4 次 FHT 单步运算
    fht_unit4 fht_unit4(Clk,Reset,FhtStarFour,
              mmm0,mmm1,mmm2,mmm3,mmm4,mmm5,mmm6,mmm7,mmm8,
              mmm9,mmm10,mmm11,mmm12,mmm13,mmm14,mmm15,
              Out0,Out1,Out2,Out3,Out4,Out5,Out6,Out7,Out8,
              Out9,Out10,Out11,Out12,Out13,Out14,Out15
              );
endmodule
```

单步 FHT 运算如下（仅仅举例第 4 步的模块）：

```
module fht_unit4(Clk,Reset,FhtStar,
          In0,In1,In2,In3,In4,In5,In6,In7,In8,
          In9,In10,In11,In12,In13,In14,In15,
          Out0,Out1,Out2,Out3,Out4,Out5,Out6,Out7,Out8,
          Out9,Out10,Out11,Out12,Out13,Out14,Out15
          );
```