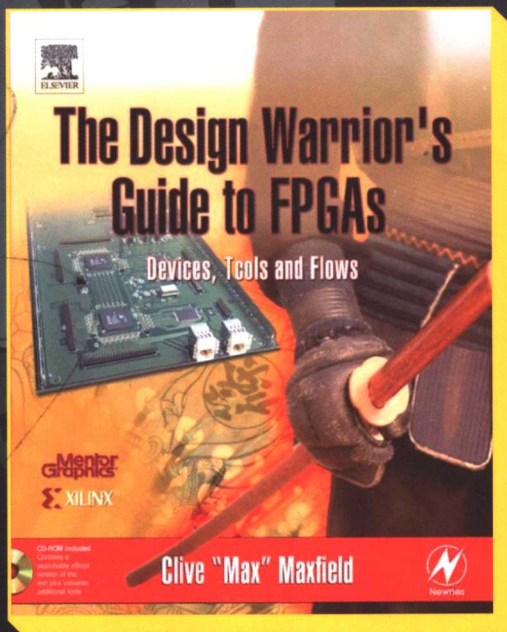


# FPGA设计指南

## 器件、工具和流程

The Design Warrior's Guide to FPGAs  
Devices, Tools and Flows

[美] Clive "Max" Maxfield 著  
杜生海 邢闻 译



TP332.1/50

2007

**TURING**

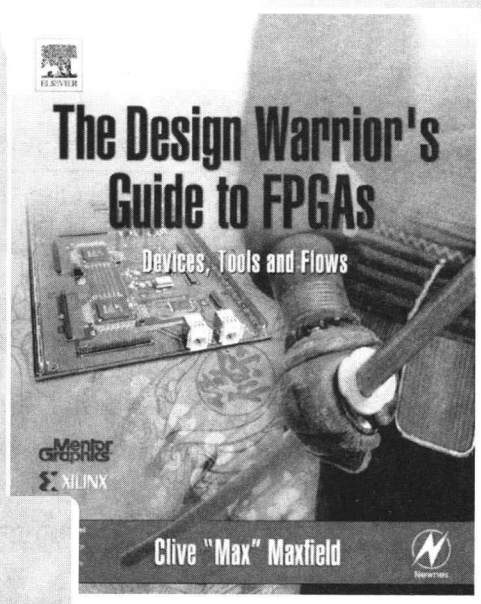
图灵电子与电气工程丛书

# FPGA设计指南

## 器件、工具和流程

The Design Warrior's Guide to FPGAs  
Devices, Tools and Flows

[美] Clive “Max” Maxfield 著  
杜生海 邢闻 译



人民邮电出版社  
北京

## 图书在版编目 (CIP) 数据

FPGA 设计指南: 器件、工具和流程 / (美) 马克斯菲尔德 (Maxfield, C.) 著; 杜生海, 邢闻译. —北京: 人民邮电出版社, 2007.12

(图灵电子与电气工程丛书)

ISBN 978-7-115-16862-7

I. F… II. ①马…②杜…③邢… III. 可程序编辑器件—系统设计 IV. TP332.1

中国版本图书馆 CIP 数据核字 (2007) 第 146283 号

## 内 容 提 要

本书用简洁的语言向读者展示了什么是FPGA、FPGA如何工作、如何对FPGA编程以及FPGA设计中遇到的各种概念、器件和工具,如传统的基于HDL/RTL的仿真和逻辑综合、最新的纯C/C++设计捕获和综合技术以及基于DSP的设计流程。另外,本书还涉及大量丰富的、工程师所需的技术细节。

本书适用于使用FPGA进行设计的工程师、进行嵌入式应用任务开发的软件工程师以及高等院校电气工程专业的师生。

图灵电子与电气工程丛书

## FPGA 设计指南: 器件、工具和流程

- 
- ◆ 著 [美] Clive “Max” Maxfield  
译 杜生海 邢 闻  
责任编辑 朱 巍
  - ◆ 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>  
北京铭成印刷有限公司印刷  
新华书店总店北京发行所经销
  - ◆ 开本: 700 × 1000 1/16  
印张: 22  
字数: 432 千字 2007 年 12 月第 1 版  
印数: 1-4 000 册 2007 年 12 月北京第 1 次印刷

著作权合同登记号 图字: 01-2006-2739 号

ISBN 978-7-115-16862-7/TN

定价: 49.00 元

读者服务热线: (010)88593802 印装质量热线: (010)67129223

# 版权声明

*The Design Warrior's Guide to FPGAs*

by Clive "Max" Maxfield

ISBN 0-7506-7604-3

Copyright © 2004, Mentor Graphics Corporation and Xilinx, Inc. All rights reserved.

Authorized Simplified Chinese translation edition published by the Proprietor.

ISBN: 981-259-632-1

Copyright © 2007 by Elsevier (Singapore) Pte Ltd. All rights reserved.

**Elsevier (Singapore) Pte Ltd.**

3 Killiney Road

#08-01 Winsland House I

Singapore 239519

Tel: (65)6349-0200

Fax: (65)6733-1817

First Published 2007

2007年初版

Printed in China by POSTS & TELECOM PRESS under special arrangement with Elsevier (Singapore) Pte Ltd. This edition is authorized for sale in China only, excluding Hong Kong SAR and Taiwan.

Unauthorized export of this edition is a violation of the Copyright Act. Violation of this Law is subject to Civil and Criminal Penalties.

本书简体中文版由Elsevier (Singapore) Pte Ltd.授权人民邮电出版社在中华人民共和国境内出版发行。

本版仅限于在中华人民共和国境内（不包括中国香港特别行政区和台湾地区）  
及标价销售。未经许可之出口，视为违反著作权法，将受法律之制裁。

## 致 谢

很早以前我就想写一本FPGA方面的书了，所以当Elsevier公司（据我所知，它是世界上最大的英语图书出版商之一）的出版人Carol Lewis给予我这样的机会时，我很高兴。

然而，有个小问题，我已经把生命中近10年的大部分时间用于工作了，现在还要把晚间和周末都消磨在写书中。某种程度上，这减少了我陪伴家人和朋友的时间。因此，我很高兴Mentor Graphics和Xilinx公司能够为本书的创作提供赞助，这使得我能够在白天写作，而晚间和周末仍然属于我自己。

作为一名专业工程师，我也不喜欢看实际上是介绍某种专门技术的书籍。所以令我高兴的是，两个赞助者都明确了本书的内容不应以Mentor或Xilinx为中心，而应该包括所有我认为有用的信息（不计个人好恶的）。

如果不与人协作，一个人是无法完成这样一本书的。在此期间，我接受了大量的帮助和建议，无法在此一一提及。但是，我想在此感谢所有帮助我的在Mentor和Xilinx公司工作的人，他们为我花费了很多时间，并给我提供了大量信息。还要感谢Gartner DataQuest公司的Gray Smith 和Daya Nadamuni，以及EETimes杂志的Richard Goering，他们一直在花时间回复我的电子邮件（这些邮件都带着令人惧怕的标题“Just one more little question...”）。

在此，我还想感谢以下公司对我提供帮助的人，这些公司是0-In、AccelChip、Actel、Aldec、Altera、Altium、Axis、Cadence、Carbon、Celoxica、Elanix、InTime、Magma、picoChip、QuickLogic、QuickSilver、Synopsys、Synplicity、The MathWorks、Hier Design和Verisity。<sup>①</sup>我还要感谢Launchbird Design Systems的Tom Hawkins给予我的热情帮助，他在开源设计工具方面的洞察力使我受益匪浅。同样，GigaTest实验室的Eric Bogatin与我分享了他在电路板级信号完整性方面的知识。

最后，当然也很重要的一点，再次感谢我的出版人——Elsevier公司的Carol Lewis，他允许我从我的*Designus Maximus Unleashed*一书中摘取出附录B的内容，而且允许我从我的另一本书*Bebop to the Boolean Boogie Second Edition*中摘取出附录C的内容。

---

<sup>①</sup> 如果有遗漏，我将十分抱歉（请通知我，我会将其加入到本书的下一版中）。

# 前 言

本书内容新颖奇特，但并不是一本典型的技术流派的书（作为作者，我知道这一点）。我之所以这么说，是因为本书旨在满足非常广泛的、各类读者的兴趣需要。本书的主要读者是目前正在使用现场可编程门阵列（FPGA）进行设计的合格的工程师，或计划在不久的将来这么做的人。本书涉及大量丰富的、工程师所喜爱的技术细节，如多种不同的设计流程、工具和概念。此外本书还涵盖了一系列技术层次相对低的主题，如基本概念。

我之所以如此编排本书，是因为目前人们对FPGA产生了浓厚的兴趣，尤其是还没有使用过或认真考虑过FPGA的人。最早的FPGA器件在它们所支持的等效逻辑门数量和所提供的性能方面相对有限，所以任何“严肃”（巨大、复杂、高性能）的设计都是由专用集成电路（ASIC）或专用标准部件（ASSP）自动实现的。然而，设计和制造ASIC和ASSP极其费时且造价高昂，并且最终设计是固定的，不开发器件的新版本就无法对其修改。

相比之下，创建一个FPGA设计的成本要比设计ASIC或ASSP低得多，在FPGA中实现设计改动更为容易，这样这种设计的上市速度就更快。特别令人感兴趣的是，最近面世的新FPGA架构中包括了数百万计的等效逻辑门、嵌入式处理器和超高速互连。这些器件允许FPGA被用于那些目前还仅是ASIC和ASSP的应用。

与FPGA器件中嵌入式处理器有关的设计，需要硬件工程师和软件工程师的协作。很多情况下，软件工程师可能不太熟悉这些器件中的一些与硬件有关的基本设计考虑。因此，除了硬件工程师，本书也旨在满足为这些器件承担开发嵌入式应用任务的软件工程师们的需求。

本书还适用于高等院校电气工程专业的学生，为EDA和FPGA公司工作的销售人员、市场人员，以及市场分析师和杂志编辑等相关读者。这些读者可以从基本概念和附录介绍的材料中发现对自己有帮助的底层技术细节。

最后，我要把它写成我最喜欢读的那类书。（此时此刻，我很愿意阅读本书——在开始这项工作时我泰然自若——因为那时我已经有了一些关于如何写作本书的线索……如果你领会我的意思。）说实话，我自己也不喜欢阅读技术书籍，因为它们往往令人头痛。因此，我在写作时，更愿意把复杂的主题分散到基本概念（“它

们从何而来”和“我们为什么这样做”)以及使人们产生兴趣的技术细节之中。相比从前精力旺盛的时候,这样做增添了我的优势,同时我还可以通过阅读自己的著作(总能有些美好的期待)从中获得惊喜和快乐。

Clive “Max” Maxfield

# 目 录

|                                   |    |  |    |
|-----------------------------------|----|--|----|
| <b>第1章 概论</b> .....               | 1  | 3.6.2 Micromatrix和Micromosaic            | 26 |
| 1.1 什么是FPGA                       | 1  | 3.6.3 门阵列                                | 27 |
| 1.2 FPGA为什么令人感兴趣                  | 1  | 3.6.4 标准单元器件                             | 28 |
| 1.3 FPGA的用途                       | 2  | 3.6.5 结构化ASIC                            | 29 |
| 1.4 本书内容                          | 3  | <b>3.7 FPGA</b> .....                    | 30 |
| 1.5 本书不包括什么                       | 4  | 3.7.1 FPGA平台                             | 32 |
| 1.6 读者对象                          | 4  | 3.7.2 FPGA-ASIC 混合                       | 33 |
| <b>第2章 基本概念</b> .....             | 5  | 3.7.3 FPGA厂商如何设计芯片                       | 34 |
| 2.1 FPGA的核心                       | 5  | <b>第4章 FPGA结构的比较</b> .....               | 35 |
| 2.2 简单的可编程功能                      | 5  | 4.1 一点提醒                                 | 35 |
| 2.3 熔丝连接技术                        | 5  | 4.2 一些背景信息                               | 35 |
| 2.4 反熔丝技术                         | 7  | 4.3 反熔丝与SRAM与其他                          | 36 |
| 2.5 掩模编程器件                        | 8  | 4.3.1 基于SRAM的器件                          | 36 |
| 2.6 PROM                          | 9  | 4.3.2 以SRAM为基础器件的安全<br>问题和解决方案           | 37 |
| 2.7 基于EPROM的技术                    | 10 | 4.3.3 基于反熔丝的器件                           | 37 |
| 2.8 基于EEPROM的技术                   | 12 | 4.3.4 基于EPROM的器件                         | 39 |
| 2.9 基于闪存的技术                       | 12 | 4.3.5 基于E <sup>2</sup> PROM/FLASH的<br>器件 | 39 |
| 2.10 基于SRAM的技术                    | 12 | 4.3.6 FLASH-SRAM混合器件                     | 39 |
| 2.11 小结                           | 13 | 4.3.7 小结                                 | 40 |
| <b>第3章 FPGA的起源</b> .....          | 15 | 4.4 细粒、中等微粒和粗粒结构                         | 40 |
| 3.1 相关的技术                         | 15 | 4.5 MUX与基于LUT的逻辑块                        | 41 |
| 3.2 晶体管                           | 15 | 4.5.1 基于MUX的结构                           | 41 |
| 3.3 集成电路                          | 16 | 4.5.2 基于LUT的结构                           | 42 |
| 3.4 SRAM/DRAM和微处理器                | 16 | 4.5.3 基于MUX还是基于LUT                       | 43 |
| 3.5 SPLD和CPLD                     | 17 | 4.5.4 3、4、5或6输入LUT                       | 44 |
| 3.5.1 PROM                        | 18 | 4.5.5 LUT与分布RAM与SR                       | 44 |
| 3.5.2 PLA                         | 20 | 4.6 CLB、LAB与slices                       | 45 |
| 3.5.3 PAL和GAL                     | 22 | 4.6.1 Xilinx 逻辑单元                        | 45 |
| 3.5.4 其他可编程选择                     | 22 | 4.6.2 Altera逻辑部件                         | 46 |
| 3.5.5 CPLD                        | 23 | 4.6.3 slicing和dicing                     | 46 |
| 3.5.6 ABEL、CUPL、PALASM、<br>JEDEC等 | 24 | 4.6.4 CLB和LAB                            | 47 |
| 3.6 专用集成电路(门阵列等)                  | 25 | 4.6.5 分布RAM和移位寄存器                        | 47 |
| 3.6.1 全定制                         | 26 |  |    |



|                                    |           |                                   |           |
|------------------------------------|-----------|-----------------------------------|-----------|
| 4.7 快速进位链 .....                    | 48        | 6.5 专业FPGA和独立EDA提供商 .....         | 74        |
| 4.8 内嵌RAM .....                    | 48        | 6.6 使用专门工具的FPGA设计<br>顾问 .....     | 75        |
| 4.9 内嵌乘法器、加法器、MAC等 .....           | 49        | 6.7 开源、免费和低成本的设计<br>工具 .....      | 75        |
| 4.10 内嵌处理器核(硬的和软的) .....           | 50        | <b>第7章 FPGA与ASIC设计风格 .....</b>    | <b>77</b> |
| 4.10.1 硬微处理器核 .....                | 50        | 7.1 引言 .....                      | 77        |
| 4.10.2 软微处理器核 .....                | 52        | 7.2 编码风格 .....                    | 77        |
| 4.11 时钟树和时间管理器 .....               | 52        | 7.3 流水线和逻辑层次 .....                | 77        |
| 4.11.1 时钟树 .....                   | 52        | 7.3.1 什么是流水线 .....                | 77        |
| 4.11.2 时钟管理器 .....                 | 53        | 7.3.2 电子系统中的流水线 .....             | 78        |
| 4.12 通用I/O .....                   | 55        | 7.3.3 逻辑层次 .....                  | 79        |
| 4.12.1 可配置I/O标准 .....              | 56        | 7.4 异步设计实践 .....                  | 80        |
| 4.12.2 可配置I/O阻抗 .....              | 56        | 7.4.1 异步结构 .....                  | 80        |
| 4.12.3 核与I/O电压 .....               | 57        | 7.4.2 组合回路 .....                  | 80        |
| 4.13 吉比特传输 .....                   | 57        | 7.4.3 延迟链 .....                   | 81        |
| 4.14 硬IP、软IP和固IP .....             | 58        | 7.5 时钟考虑 .....                    | 81        |
| 4.15 系统门与实际的门 .....                | 59        | 7.5.1 时钟域 .....                   | 81        |
| 4.16 FPGA年 .....                   | 60        | 7.5.2 时钟平衡 .....                  | 81        |
| <b>第5章 FPGA编程(配置) .....</b>        | <b>62</b> | 7.5.3 门控时钟与使能时钟 .....             | 81        |
| 5.1 引言 .....                       | 62        | 7.5.4 PLL和时钟调节电路 .....            | 82        |
| 5.2 配置文件 .....                     | 62        | 7.5.5 跨时钟域数据传输的<br>可靠性 .....      | 82        |
| 5.3 配置单元 .....                     | 62        | 7.6 寄存器和锁存器考虑 .....               | 82        |
| 5.4 基于反熔丝的FPGA .....               | 63        | 7.6.1 锁存器 .....                   | 82        |
| 5.5 基于SRAM的FPGA .....              | 64        | 7.6.2 具有“置位”和“复位”<br>输入的触发器 ..... | 82        |
| 5.5.1 迅速的过程欺骗了眼睛 .....             | 65        | 7.6.3 全局复位和初始化条件 .....            | 83        |
| 5.5.2 对嵌入式(块)RAM、分布<br>RAM编程 ..... | 65        | 7.7 资源共享(时分复用) .....              | 83        |
| 5.5.3 多编程链 .....                   | 66        | 7.7.1 使用它或者放弃它 .....              | 83        |
| 5.5.4 器件的快速重新初始化 .....             | 66        | 7.7.2 其他内容 .....                  | 83        |
| 5.6 使用配置端口 .....                   | 66        | 7.8 状态机编码 .....                   | 84        |
| 5.6.1 FPGA作为主设备串行下载 .....          | 67        | 7.9 测试方法学 .....                   | 84        |
| 5.6.2 FPGA作为主设备并行下载 .....          | 68        | <b>第8章 基于原理图的设计流程 .....</b>       | <b>85</b> |
| 5.6.3 FPGA作为从设备并行下载 .....          | 69        | 8.1 往昔的时光 .....                   | 85        |
| 5.6.4 FPGA作为从设备串行下载 .....          | 70        | 8.2 EDA初期 .....                   | 86        |
| 5.7 使用JTAG端口 .....                 | 70        | 8.2.1 前端工具,如逻辑仿真 .....            | 86        |
| 5.8 使用嵌入式处理器 .....                 | 71        | 8.2.2 后端工具如版图设计 .....             | 89        |
| <b>第6章 谁在参与游戏 .....</b>            | <b>73</b> | 8.2.3 CAE + CAD = EDA .....       | 90        |
| 6.1 引言 .....                       | 73        | 8.3 简单的原理图驱动ASIC设计<br>流程 .....    | 90        |
| 6.2 FPGA和FPAA提供商 .....             | 73        |                                   |           |
| 6.3 FPNA 提供商 .....                 | 73        |                                   |           |
| 6.4 全线EDA提供商 .....                 | 74        |                                   |           |

|                                |                              |     |
|--------------------------------|------------------------------|-----|
| 8.4 简单(早期)的原理图驱动FPGA           | 10.2 基于ASIC的SVP方法            | 114 |
| 设计流程                           | 10.2.1 门级SVP(由快速综合产生)        | 115 |
| 8.4.1 映射                       | 10.2.2 门级SVP(由基于增益的综合产生)     | 115 |
| 8.4.2 包装                       | 10.2.3 团簇SVP                 | 117 |
| 8.4.3 布局布线                     | 10.2.4 基于RTL的SVP             | 117 |
| 8.4.4 时序分析和布局布线后仿真             | 10.3 基于FPGA的SVP              | 119 |
| 8.5 平坦的原理图与分层次的原理图             | 10.3.1 交互式操作                 | 120 |
| 8.5.1 沉闷的扁平原理图                 | 10.3.2 增量式布局布线               | 121 |
| 8.5.2 分等级(基于模块)的原理图            | 10.3.3 基于RTL的FPGA SVP        | 121 |
| 8.6 今天的原理图驱动设计流程               | <b>第11章 基于C/C++等语言的设计流程</b>  | 122 |
| <b>第9章 基于HDL的设计流程</b>          | 11.1 传统的HDL设计流程存在的问题         | 122 |
| 9.1 基于原理图流程的问题                 | 11.2 C对C++与并行执行对顺序执行         | 124 |
| 9.2 基于HDL设计流程的出现               | 11.3 基于SystemC的设计流程          | 125 |
| 9.2.1 不同的抽象层次                  | 11.3.1 什么是SystemC以及它从哪里来     | 125 |
| 9.2.2 早期基于HDL的ASIC设计流程         | 11.3.2 SystemC 1.0           | 125 |
| 9.2.3 早期基于HDL的FPGA设计流程         | 11.3.3 SystemC 2.0           | 126 |
| 9.2.4 知道结构的FPGA流程              | 11.3.4 抽象级                   | 127 |
| 9.2.5 逻辑综合与基于物理的综合             | 11.3.5 基于SystemC设计流程的可选方案    | 127 |
| 9.3 图形设计输入的生活                  | 11.3.6 要么喜爱它,要么讨厌它           | 129 |
| 9.4 绝对过剩的HDL                   | 11.4 基于增强型C/C++的设计流程         | 129 |
| 9.4.1 Verilog HDL              | 11.4.1 什么是增强型C/C++           | 129 |
| 9.4.2 VHDL和VITAL               | 11.4.2 可选择的增强型C/C++设计流程      | 131 |
| 9.4.3 混合语言设计                   | 11.5 基于纯C/C++的设计流程           | 132 |
| 9.4.4 UDL/I                    | 11.6 综合的不同抽象级别               | 134 |
| 9.4.5 Superlog 和 SystemVerilog | 11.7 混合语言设计和验证环境             | 136 |
| 9.4.6 SystemC                  | <b>第12章 基于DSP的设计流程</b>       | 138 |
| 9.5 值得深思的事                     | 12.1 DSP简介                   | 138 |
| 9.5.1 担心,非常担心                  | 12.2 可选择的DSP实现方案             | 139 |
| 9.5.2 串行与并行多路复用器               | 12.2.1 随便选一个器件,不过不要让我看到是哪种器件 | 139 |
| 9.5.3 小心锁存器                    | 12.2.2 系统级评估和算法验证            | 139 |
| 9.5.4 聪明地使用常量                  | 12.2.3 在DSP内核中运行的软件          | 140 |
| 9.5.5 资源共用考虑                   |                              |     |
| 9.5.6 还有一些不可忽视的内容              |                              |     |
| <b>第10章 FPGA设计中的硅虚拟原型</b>      |                              |     |
| 10.1 什么是硅虚拟原型                  |                              |     |

|                                       |     |   |     |
|---------------------------------------|-----|---|-----|
| 12.2.4 专用DSP硬件 .....                  | 141 | 14.2.1 模块化设计 .....                        | 169 |
| 12.2.5 与DSP相关的嵌入式<br>FPGA资源 .....     | 143 | 14.2.2 增量设计 .....                         | 169 |
| 12.3 针对DSP的以FPGA为中心的<br>设计流程 .....    | 144 | 14.2.3 存在的问题 .....                        | 170 |
| 12.3.1 专用领域语言 .....                   | 144 | 14.3 总有其他办法 .....                         | 171 |
| 12.3.2 系统级设计和仿真环境 .....               | 145 | <b>第15章 高速设计与其他PCB设计</b>                  |     |
| 12.3.3 浮点与定点表示 .....                  | 146 | <b>注意事项</b> .....                         | 172 |
| 12.3.4 系统/算法级向RTL的转换<br>(手工转换) .....  | 146 | 15.1 开始之前 .....                           | 172 |
| 12.3.5 系统/算法级向RTL的转换<br>(自动生成) .....  | 147 | 15.2 我们都很年轻, 因此 .....                     | 172 |
| 12.3.6 系统/算法级向C/C++的<br>转换 .....      | 148 | 15.3 变革的时代 .....                          | 173 |
| 12.3.7 模块级IP环境 .....                  | 150 | 15.4 其他注意事项 .....                         | 175 |
| 12.3.8 别忘了测试平台 .....                  | 150 | 15.4.1 高速设计 .....                         | 175 |
| 12.4 DSP与VHDL/Verilog混合<br>设计环境 ..... | 151 | 15.4.2 信号完整性分析 .....                      | 175 |
| <b>第13章 基于嵌入式处理器的<br/>设计流程</b> .....  | 153 | 15.4.3 SPICE与IBIS .....                   | 176 |
| 13.1 引言 .....                         | 153 | 15.4.4 起动功率 .....                         | 176 |
| 13.2 硬核与软核 .....                      | 154 | 15.4.5 使用内部末端阻抗 .....                     | 176 |
| 13.2.1 硬核 .....                       | 154 | 15.4.6 串行或并行处理数据 .....                    | 177 |
| 13.2.2 微处理器软核 .....                   | 156 | <b>第16章 观察FPGA的内部节点</b> .....             | 178 |
| 13.3 将设计划分为硬件和<br>软件部分 .....          | 157 | 16.1 缺乏可见性 .....                          | 178 |
| 13.4 硬件和软件的世界观 .....                  | 159 | 16.2 使用多路复用技术 .....                       | 179 |
| 13.5 利用FPGA作为自身的<br>开发环境 .....        | 160 | 16.3 专用调试电路 .....                         | 180 |
| 13.6 增强设计的可见性 .....                   | 161 | 16.4 虚拟逻辑分析仪 .....                        | 180 |
| 13.7 其他一些混合验证方法 .....                 | 161 | 16.5 虚拟线路 .....                           | 181 |
| 13.7.1 RTL (VHDL或Verilog) .....       | 162 | 16.5.1 问题描述 .....                         | 181 |
| 13.7.2 C/C++、SystemC等 .....           | 162 | 16.5.2 虚拟线路解决方案 .....                     | 183 |
| 13.7.3 硬件模拟器中的物理芯片 .....              | 163 | <b>第17章 IP</b> .....                      | 185 |
| 13.7.4 指令集仿真器 .....                   | 163 | 17.1 IP的来源 .....                          | 185 |
| 13.8 一个相当巧妙的设计环境 .....                | 165 | 17.2 人工优化的IP .....                        | 185 |
| <b>第14章 模块化设计和增量设计</b> .....          | 167 | 17.2.1 未加密的RTL级IP .....                   | 186 |
| 14.1 将设计作为一个大的模块<br>进行处理 .....        | 167 | 17.2.2 加密的RTL级IP .....                    | 186 |
| 14.2 将设计划分为更小的模块 .....                | 168 | 17.2.3 未经布局布线的网表级IP .....                 | 186 |
|                                       |     | 17.2.4 布局布线后的网表级IP .....                  | 186 |
|                                       |     | 17.3 IP核生成器 .....                         | 187 |
|                                       |     | 17.4 综合资料 .....                           | 187 |
|                                       |     | <b>第18章 ASIC设计与FPGA设计<br/>之间的移植</b> ..... | 189 |
|                                       |     | 18.1 可供选择的设计方法 .....                      | 189 |
|                                       |     | 18.1.1 只做FPGA设计 .....                     | 189 |
|                                       |     | 18.1.2 FPGA之间的转换 .....                    | 189 |
|                                       |     | 18.1.3 FPGA到ASIC的转换 .....                 | 190 |

|                                  |     |                                   |     |
|----------------------------------|-----|-----------------------------------|-----|
| 18.1.4 ASIC到FPGA的转换 .....        | 191 | 19.7 混合设计 .....                   | 218 |
| <b>第19章 仿真、综合、验证等设计</b>          |     | 19.7.1 HDL语言到C语言的转换 .....         | 218 |
| <b>工具</b> .....                  | 193 | 19.7.2 代码覆盖率 .....                | 219 |
| 19.1 引言 .....                    | 193 | 19.7.3 性能分析 .....                 | 220 |
| 19.2 仿真(基于周期、事件<br>驱动等) .....    | 193 | <b>第20章 选择合适的器件</b> .....         | 221 |
| 19.2.1 什么是事件驱动逻辑<br>仿真器 .....    | 193 | 20.1 丰富的选择 .....                  | 221 |
| 19.2.2 事件驱动逻辑仿真器发展<br>过程简述 ..... | 195 | 20.2 要是选型工具就好了 .....              | 221 |
| 19.2.3 逻辑值与不同逻辑值系统 .....         | 196 | 20.3 工艺 .....                     | 222 |
| 19.2.4 混合语言仿真 .....              | 197 | 20.4 基本资源和封装 .....                | 223 |
| 19.2.5 其他延迟格式 .....              | 198 | 20.5 通用I/O接口 .....                | 223 |
| 19.2.6 基于周期的仿真器 .....            | 201 | 20.6 嵌入式乘法器、RAM等 .....            | 224 |
| 19.2.7 选择世界上最好的逻辑<br>仿真器 .....   | 202 | 20.7 嵌入式处理器核 .....                | 224 |
| 19.3 综合(逻辑/HDL综合与<br>物理综合) ..... | 203 | 20.8 吉比特I/O能力 .....               | 224 |
| 19.3.1 逻辑/HDL综合技术 .....          | 203 | 20.9 可用的IP .....                  | 224 |
| 19.3.2 物理综合技术 .....              | 203 | 20.10 速度等级 .....                  | 225 |
| 19.3.3 时序重调、复制及二次<br>综合 .....    | 204 | 20.11 轻松的注解 .....                 | 226 |
| 19.3.4 选择世界上最好的综合<br>工具 .....    | 206 | <b>第21章 吉比特收发器</b> .....          | 227 |
| 19.4 时序分析(静态与动态) .....           | 206 | 21.1 引言 .....                     | 227 |
| 19.4.1 静态时序分析 .....              | 206 | 21.2 差分对 .....                    | 228 |
| 19.4.2 统计静态时序分析 .....            | 207 | 21.3 多种多样的标准 .....                | 229 |
| 19.4.3 动态时序分析 .....              | 207 | 21.4 8bit/10bit编码等 .....          | 230 |
| 19.5 一般验证 .....                  | 208 | 21.5 深入收发器模块内部 .....              | 231 |
| 19.5.1 验证IP .....                | 208 | 21.6 组合多个收发器 .....                | 233 |
| 19.5.2 验证环境和创建testbench .....    | 210 | 21.7 可配置资源 .....                  | 234 |
| 19.5.3 分析仿真结果 .....              | 211 | 21.7.1 逗号检测 .....                 | 234 |
| 19.6 形式验证 .....                  | 211 | 21.7.2 差分输出摆幅 .....               | 234 |
| 19.6.1 形式验证的不同种类 .....           | 212 | 21.7.3 片内末端电阻 .....               | 234 |
| 19.6.2 形式验证究竟是什么 .....           | 212 | 21.7.4 预加重 .....                  | 234 |
| 19.6.3 术语及定义 .....               | 213 | 21.7.5 均衡化 .....                  | 235 |
| 19.6.4 其他可选的断言/属性<br>规范技术 .....  | 214 | 21.8 时钟恢复、抖动和眼图 .....             | 236 |
| 19.6.5 静态形式验证和动态形式<br>验证 .....   | 216 | 21.8.1 时钟恢复 .....                 | 236 |
| 19.6.6 各种语言的总结 .....             | 217 | 21.8.2 抖动和眼图 .....                | 237 |
|                                  |     | <b>第22章 可重配置计算</b> .....          | 239 |
|                                  |     | 22.1 可动态重配置逻辑 .....               | 239 |
|                                  |     | 22.2 可动态重配置互连线 .....              | 239 |
|                                  |     | 22.3 可重配置计算 .....                 | 240 |
|                                  |     | <b>第23章 现场可编程节点阵列</b> .....       | 243 |
|                                  |     | 23.1 引言 .....                     | 243 |
|                                  |     | 23.2 算法评估 .....                   | 244 |
|                                  |     | 23.3 picoChip公司的picoArray技术 ..... | 245 |

|   |            |
|---|------------|
| 23.3.1 一个理想的picoArray应用:<br>无线基站 .....    | 246        |
| 23.3.2 picoArray设计环境 .....                | 247        |
| 23.4 QuickSilver公司的ACM技术 .....            | 247        |
| 23.4.1 设计混合节点 .....                       | 249        |
| 23.4.2 系统控制器节点、输入输出<br>节点及其他节点 .....      | 249        |
| 23.4.3 空间与时间分割 .....                      | 250        |
| 23.4.4 在ACM上创建和运行程序 .....                 | 251        |
| 23.4.5 还有更多的内容 .....                      | 252        |
| 23.5 这就是硅, 但与我们知道的<br>并不相同 .....          | 252        |
| <b>第24章 独立的设计工具 .....</b>                 | <b>253</b> |
| 24.1 引言 .....                             | 253        |
| 24.2 ParaCore Architect .....             | 253        |
| 24.2.1 产生浮点处理功能模块 .....                   | 254        |
| 24.2.2 产生FFT功能模块 .....                    | 254        |
| 24.2.3 基于网络的接口 .....                      | 255        |
| 24.3 Confluence系统设计语言 .....               | 256        |
| 24.3.1 一个简单的例子 .....                      | 256        |
| 24.3.2 还有更多的功能 .....                      | 258        |
| 24.3.3 免费评估版本 .....                       | 259        |
| 24.4 你是否具有这种工具 .....                      | 259        |
| <b>第25章 创建基于开源的设计流程 .....</b>             | <b>260</b> |
| 25.1 如何白手起家创办一家FPGA<br>设计工作室 .....        | 260        |
| 25.2 开发平台: Linux .....                    | 260        |
| 25.3 验证环境 .....                           | 262        |
| 25.3.1 Icarus Verilog .....               | 263        |
| 25.3.2 Dinotrace和GTKWave .....            | 263        |
| 25.3.3 Covered代码覆盖率工具 .....               | 263        |
| 25.3.4 Verilator .....                    | 263        |
| 25.3.5 Python .....                       | 264        |
| 25.4 形式验证 .....                           | 264        |
| 25.4.1 开源模型检查 .....                       | 265        |
| 25.4.2 基于开源的自动推断 .....                    | 265        |
| 25.4.3 真正的问题是什么 .....                     | 266        |
| 25.5 访问公共IP元件 .....                       | 266        |
| 25.5.1 OpenCores .....                    | 266        |
| 25.5.2 OVL .....                          | 267        |
| 25.6 综合与实现工具 .....                        | 267        |
| 25.7 FPGA开发板 .....                        | 267        |
| 25.8 综合材料 .....                           | 267        |
| <b>第26章 FPGA未来的发展 .....</b>               | <b>269</b> |
| 26.1 一种担忧 .....                           | 269        |
| 26.2 下一代结构和技术 .....                       | 269        |
| 26.2.1 十亿晶体管级器件 .....                     | 269        |
| 26.2.2 超快速I/O .....                       | 270        |
| 26.2.3 超快速配置 .....                        | 270        |
| 26.2.4 更多的硬IP .....                       | 271        |
| 26.2.5 模拟与混合信号器件 .....                    | 271        |
| 26.2.6 ASMBL与其他结构 .....                   | 272        |
| 26.2.7 不同的结构粒度 .....                      | 272        |
| 26.2.8 ASIC结构中的嵌入式<br>FPGA内核 .....        | 273        |
| 26.2.9 ASIC和FPGA结构中嵌入<br>FPNA内核或者相反 ..... | 273        |
| 26.2.10 基于MRAM的器件 .....                   | 273        |
| 26.3 设计工具 .....                           | 273        |
| 26.4 期待意外的发生 .....                        | 274        |
| <b>附录A 信号完整性简介 .....</b>                  | <b>275</b> |
| <b>附录B 深亚微米延迟效应 .....</b>                 | <b>285</b> |
| <b>附录C 线性移位寄存器 .....</b>                  | <b>299</b> |
| <b>术语表 .....</b>                          | <b>310</b> |
| <b>索引 .....</b>                           | <b>326</b> |

# 第 1 章 概 论

## 1.1 什么是FPGA

现场可编程门阵列 (FPGA) 是由可配置 (可编程) 逻辑块组成的数字集成电路 (IC), 这些逻辑块之间用可配置的互连资源。设计工程师可以对这类器件进行 (编程) 配置来完成各种各样的任务。

由于实现方式不同, 有些FPGA只能编程一次, 而有些FPGA可以重复多次编程。自然, 只能编程一次的器件被称为一次性可编程 (OTP) 器件。

FPGA名称中的“现场可编程”是指编程“在现场”进行。(与那些内部功能已被制造商固化的器件正相反。) 这意味着FPGA可能是在实验室中配置的, 也可能是在对已应用于外部世界的电子系统中某个设备功能的改进。如果一个器件能够在某个高层系统有编程的能力, 它就是在系统可编程 (ISP) 器件。

## 1.2 FPGA为什么令人感兴趣

数字集成电路 (IC) 有很多种类型, 包括所谓的“jelly-bean”逻辑 (指由一些简单、固定的逻辑功能组成的小元件)、存储器件和微处理器。但是, 我们特别感兴趣的是可编程逻辑器件 (PLD)、专用集成电路 (ASIC)、专用标准部件 (ASSP), 当然, 还有FPGA。

1

就我们讨论的这部分内容而言, PLD这个术语包括简单可编程逻辑器件 (SPLD) 和复杂可编程逻辑器件 (CPLD)。

第2章和第3章将更详细地讨论不同类型的PLD、ASIC和ASSP, 眼下我们只需要了解, PLD是一种内部架构已经由制造商确定、但可由工程师通过在现场配置 (编程) 来实现多种不同功能的器件。但是, 和FPGA相比, 这种器件包含的逻辑门相对有限, 它们可以用来实现的功能也更少、更简单。

与PLD对应的是ASIC和ASSP, 它们拥有数百万计的逻辑门, 可以用来产生令人难以置信的强大复杂功能。ASIC和ASSP基于相同的设计流程和制造工艺, 都是定制设计来满足专门的应用。它们之间仅有的不同就在于, ASIC是专为某个特定公司的使用而设计制造的, 而ASSP的目的是出售给多个顾客。(今后, 在我们使用ASIC一词时, 它可能是指ASSP, 除非另外指出或此处的解释不符合上下文。)

尽管ASIC在尺寸 (晶体管的数量)、复杂性和性能实现上都是最佳的, 但是设

设计和制造它是一个很耗费时间和代价昂贵的过程，还有一个不足，就是最终设计是固定的，不开发新版本，就无法调整。

2 FPGA因而弥补了PLD和ASIC之间的空白。因为它可以像PLD那样在现场编程，同时具有数百万的逻辑门<sup>①</sup>，可以用来实现很大、很复杂的功能，而这是从前只有使用ASIC才可以实现的。

FPGA设计的成本要大大低于ASIC设计的成本（尽管大规模生产时ASIC部件肯定要比FPGA便宜很多），同时，在FPGA中对设计修改要容易很多，而且这种设计的产品上市时间会早很多。FPGA催生了许多富有创新意识的小公司（当然FPGA也为很多大的系统设计公司所使用），因为它给“小规模开发”提供了便利。这意味着某个工程师或小工程团队可以在基于FPGA的测试平台上实现他们的软件或硬件想法，而不必承担数额巨大的不可重现实工程（NRE）成本或采购昂贵的ASIC软件开发工具。因此，据估计在2003年只有1500家到4000家ASIC设计公司<sup>②</sup>和5000家ASSP设计公司（这些数字近年来正在显著下降）；相反，同年估计有大约45万家FPGA设计公司<sup>③</sup>。

3

### 1.3 FPGA的用途

20世纪80年代中期，FPGA刚出现时，大部分用来实现粘合逻辑<sup>④</sup>、中等复杂度的状态机和相对有限的数据处理任务。在20世纪90年代早期，FPGA的规模和复杂度开始增加，那时它们的主要市场在通信和网络领域，这都涉及大量数据的处理。后来，到了20世纪90年代末，FPGA在消费、汽车和工业领域的应用经历了爆炸式增长。

FPGA经常用于制作ASIC设计的原型或是提供一个硬件平台来验证一个新算法的物理层实现。但是，FPGA的低开发成本和短上市时间意味着越来越多公司使用它制造最终的产品。（实际上一些主要的FPGA厂商就拥有直接与ASIC在某些特定市场上竞争的器件。）

到了21世纪早期，已经可以买到有数百万门容量的高性能FPGA。这些器件中有

① 在FPGA概念中，哪些器件包含“逻辑门”这个概念有些模糊。这将在第4章详细讨论。

② 该数字不太确切，这取决于所指的具体对象。

③ 这些数字难以确定的另外一个原因是，人们在到底什么是“设计开始”上不容易达成一致。例如，在ASIC设计中，“设计开始”应该包括中途取消的设计，还是仅仅考虑那些最终完成流片的设计？对于FPGA，由于FPGA的可重配置的特性，这个问题几乎要变得没有意义了。或许还应该讲一讲下面的事，在给我介绍了一个以FPGA为主的工业评论网站之后，一名来自FPGA厂商的代表补充说：“其实这里给出的数字不是非常准确。”当我问为什么时，他露出了狡黠的笑容：“主要是因为我们不能给他提供很好的数据！”

④ 粘合逻辑是指用来连接（“粘合”）大逻辑块、功能块或器件的相对少量的简单逻辑（以及作为它们之间的接口）。

的内嵌了微处理器核、高速输入/输出 (I/O) 接口和类似的模块。这样, 今天的FPGA几乎可以用来实现任何东西, 包括通信设备和由软件定义无线电; 雷达、影像和其他数字信号处理 (DSP) 的应用; 直至包含硬件和软件的片上系统 (SoC)<sup>①</sup>。

4

说得更具体一些, FPGA正在蚕食4个主要的市场: ASIC和定制硅、DSP、嵌入式微处理器应用和物理层通信芯片。另外, FPGA还创建一个属于自己的新市场: 可重配置计算 (RC) 技术。

- ASIC和定制硅: 正如前面部分提到的, 今天的FPGA正逐渐用来实现各种设计, 而从前这些设计仅能由ASIC和定制硅来完成。
- 数字信号处理: 高速DSP传统上是用数字信号处理器来实现的, 实际上这是一种特殊剪裁的微处理器。但是, 今天的FPGA可以包含内嵌的乘法器、专用计算例程和大量的片上RAM等所有DSP操作所需的特性。这些特性再加上FPGA提供的并行性, 其结果就是比最快的DSP芯片还要快上500倍乃至更多。
- 嵌入式微处理器: 小的控制功能传统上是用称作微控制器的专用嵌入式微处理器处理的。这些低成本器件由围绕在处理器核外的片上程序和指令存储器、定时器和I/O组成。但是, 随着FPGA的价格不断下降, 甚至最小的器件都足以实现一个集成有可选定制I/O功能的软处理器核。结果就是FPGA对嵌入控制应用越来越具有吸引力。
- 物理层通信: FPGA长期以来用于实现物理层通信芯片和网络协议层互连的粘合逻辑。今天的高端FPGA拥有了多种高速收发器, 这意味着通信和网络功能可以合并到同一设备中。
- 可重配置计算技术: 这是指由FPGA提供的固有的并行性和可重配置性来实现软件算法的“硬件加速”。许多公司正在建立大型的以FPGA为基础的可重配置计算引擎, 来完成从硬件仿真到密码分析, 再到开发新药物的任务。

5

## 1.4 本书内容

任何从事过电子设计或电子设计自动化领域的人都知道, 近年来这些领域正变得越来越复杂, FPGA也不例外。

早些时候 (在20世纪80年代中期) 还不那么复杂, 当时FPGA才刚刚出现。最早的FPGA只包含数百个简单的逻辑门, 设计这些元件的流程——主要是基于原理图——也容易理解和使用。相比之下, 今天的FPGA极其复杂, 设计工具、流程、工艺多得令人眼花缭乱。

<sup>①</sup> 尽管片上系统 (SoC) 一词倾向于表示在单个器件上的完整电子学系统, 现状是人们无一例外地还需要额外的元件。因此, 更准确的表达可能是片上子系统 (SSoC) 或片上部分系统 (PoaSoC)。



本书的开篇介绍了基本的概念和目前常见的几种FPGA的架构和器件。然后研究各种设计工具和流程，具体应用哪种取决于设计工程师的需要。接着，除了了解FPGA的内部组成之外，本书也考虑了将器件与同一个电路板内的系统其余部分集成时的各种问题，包括对最近才出现的吉比特接口的讨论。

6

## 1.5 本书不包括什么

本书内容不针对特定的FPGA厂商或特殊的FPGA器件，因为新的特性和芯片类型出现得很快，以至于在本书出版之前（有时是在我撰写完相关文字以前），这里的相关内容已经过时了。

与之类似，本书尽可能地（在这样做有意义的范围内）不提及具体EDA厂商或他们的工具名称，因为这些厂商经常彼此并购，改变其公司名字——换句话说，让这些名字变得奇形怪状——或者变更他们的设计和分析工具的名称。类似地，事情发展得如此迅速，这个行业很少说“A工具有这个特性，而B工具没有”，因为可能仅仅几个月，B工具已经得到增强，而A工具已经被淘汰了。

由于上述原因，本书主要介绍不同风格的FPGA器件和多种设计工具的概念以及流程，但不涉及哪个FPGA厂商支持哪种架构，哪个EDA厂商和工具支持哪种专门特性的问题，这留给读者去研究（在第6章列出了有用的Web站点）。

7

## 1.6 读者对象

本书面向大范围的读者，包括小型FPGA设计顾问、大系统机构中的硬件和软件设计工程师、向FPGA领域转型的ASIC设计者、开始使用FPGA的DSP设计者、高等院校相关专业学生、EDA和FPGA公司的销售人员、市场人员和其他人员、评论专栏和杂志社的编辑们。

8