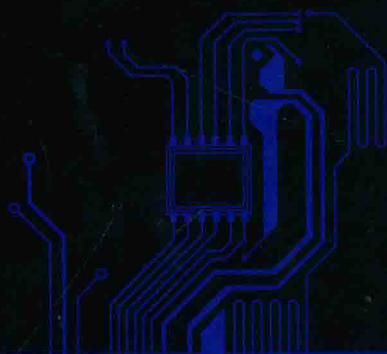




勇敢的芯 伴你玩转Altera FPGA

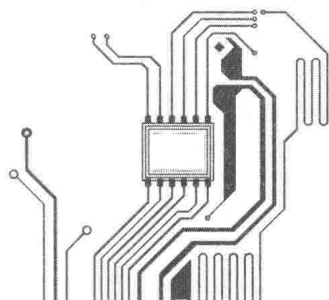
◎ 吴厚航 编著



- ◎ 提供FPGA板级设计说明、工具安装说明、入门实例、片内资源应用实例和综合实例，由浅入深地全面掌握FPGA的开发设计
- ◎ 可配套量身定制的开发实验平台，理论和实践相结合
- ◎ 适合Altera FPGA与Verilog HDL语言的入门学习



清华大学出版社



勇敢的芯 伴你玩转Altera FPGA

◎ 吴厚航 编著

清华大学出版社
北京

内 容 简 介

本书使用 Altera 公司的 Cyclone IV FPGA 器件,由浅入深地引领读者从板级设计、基础入门实例、FPGA 片内资源应用实例和综合进阶实例等方面,玩转 FPGA 逻辑设计。本书基于特定的 FPGA 实验平台,既有足够的理论知识深度作支撑,也有丰富的例程供实践学习,并且穿插了笔者在多年 FPGA 学习和开发过程中所积累的经验 and 技巧。

无论对于希望快速掌握 Verilog 语言进行 FPGA 开发的初学者,还是希望快速掌握基于 Altera Cyclone IV FPGA 进行开发的设计者,本书都是很好的选择。

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

勇敢的芯伴你玩转 Altera FPGA/吴厚航编著. —北京:清华大学出版社,2017

(电子设计与嵌入式开发实践丛书)

ISBN 978-7-302-47421-0

I. ①勇… II. ①吴… III. ①现场可编程门阵列—系统设计 IV. ①TP332.1

中国版本图书馆 CIP 数据核字(2017)第 129443 号

责任编辑:刘 星

封面设计:刘 键

责任校对:焦丽丽

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 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm 印 张:21.5

字 数:524 千字

版 次:2017 年 10 月第 1 版

印 次:2017 年 10 月第 1 次印刷

印 数:1~2500

定 价:59.00 元

产品编号:074826-01

前言

FPGA 技术在当前的电子设计领域越来越火热。它的成本虽然还是高高在上,但是它给电子系统所带来的不可限量的速度和带宽及其在灵活性、小型性方面的优势,越来越为对性能要求高、偏重定制化需求的开发者所青睐。因此,越来越多的电子工程师和电子专业在校学生希望能够掌握这门技术。而对一门电子技能的掌握,单凭读几本初级入门教材是很难达到的。笔者结合自身的学习经历,为广大学习者量身打造了基于低成本、高性价比的 Altera Cyclone IV FPGA 器件的硬件开发学习平台。基于该平台,配套本书的各种基本概念阐释和例程讲解,相信可以帮助大家快速掌握这门新技术。

全书共 10 章,各章主要内容如下:

第 1 章是基础中的基础,讲述可编程器件的基本概念及主要应用领域、相对传统技术的优势和开发流程。

第 2 章从 FPGA 开发平台的电路板设计入手,介绍 FPGA 板级硬件电路设计要点,以及本书配套开发平台的外围电路设计。

第 3 章从最基础的 0 和 1 开始回顾数字电路的基础,同时深入探讨读者所关心的可编程器件的内部架构和原理。

第 4 章讲述开发环境的搭建,包括 Altera FPGA 集成开发环境 Quartus II、仿真工具 ModelSim、文本编辑器 Notepad++ 以及下载器驱动和 UART 驱动安装,帮助读者快速解决学习路上遇到的最棘手的“软”问题。

第 5 章讲述 Verilog 的基本语法,包括语法的学习方法、可综合的语法子集以及代码风格与书写规范。

第 6 章和第 7 章完成最基本的工程创建、语法检查、仿真验证以及编译,甚至在线板级调试和代码固化,带领读者初步掌握基于 Altera Cyclone IV 的 FPGA 开发流程。

第 8 章介绍 13 个最基本的入门实例。

第 9 章通过 6 个实例帮助读者熟悉 FPGA 除逻辑资源以外的其他丰富资源,如 PLL 和可配置为 ROM、RAM、FIFO 的内嵌存储器,以及在线逻辑分析仪 SignalTap II 等。

第 10 章的 15 个例程,是对前面一些实例的集成整合,力图通过大量的实例实践,帮助读者熟练掌握 FPGA 的基本开发设计。

本书既有对基础理论知识的专门讲解,也有非常详细的实例演练和讲解,更多的是在实

Foreword

践中传递实用的设计方法与技巧,非常适合初学者。

本书配套例程的下载链接为 <http://pan.baidu.com/s/1i5LMUUD>。

本书配套开发平台的淘宝链接: <https://myfpga.taobao.com/>。

吴厚航(网名:特权同学)

2017年7月于上海

目录

第 1 章	FPGA 开发入门	1
1.1	FPGA 基础入门	1
1.2	FPGA 的优势在哪里	4
1.3	FPGA 应用领域	5
1.4	FPGA 开发流程	6
第 2 章	实验平台“勇敢的芯”板级电路详解	8
2.1	板级电路整体架构	8
2.2	电源电路	10
2.3	复位与时钟电路	12
2.3.1	关于 FPGA 器件的时钟	12
2.3.2	关于 FPGA 器件的复位	14
2.3.3	实验平台电路解析	14
2.4	FPGA 下载配置电路	15
2.5	SRAM 接口电路	17
2.6	ADC/DAC 芯片电路	18
2.7	UART 接口电路	19
2.8	RTC 接口电路	19
2.9	4×4 矩阵按键电路	20
2.10	VGA 显示接口电路	21
2.11	蜂鸣器、流水灯、数码管、拨码开关电路	22
2.12	超声波接口、外扩 LCD 接口电路	22
第 3 章	逻辑设计基础	24
3.1	0 和 1——精彩世界由此开始	24
3.2	表面现象揭秘——逻辑关系	26
3.3	内里本质探索——器件结构	30

Contents

第4章 软件安装与配置	35
4.1 软件下载和 license 申请	35
4.2 Quartus II 与 ModelSim-Altera 的安装	36
4.3 文本编辑器 Notepad++ 安装	37
4.4 Quartus II 中使用 Notepad++ 的关联设置	41
4.5 USB-Blaster 的驱动安装	43
4.5.1 Windows XP 系统 USB-Blaster 安装	43
4.5.2 在 Windows 7 系统安装 USB-Blaster	45
4.5.3 在 Windows 8 系统安装 USB-Blaster	47
4.6 串口芯片驱动安装	48
4.6.1 驱动安装	48
4.6.2 设备识别	48
第5章 Verilog 语法概述	50
5.1 语法学习的经验之谈	50
5.2 可综合的语法子集	52
5.3 代码风格与书写规范	57
第6章 基于仿真的第一个工程实例	72
6.1 新建工程	72
6.2 Verilog 源码文件创建与编辑	75
6.2.1 Verilog 源码文件创建	75
6.2.2 Verilog 源码文件编辑	75
6.3 Verilog 语法检查	76
6.4 ModelSim 仿真验证	77
6.4.1 Quartus II 基本设置	77
6.4.2 测试脚本创建与编辑	77
6.4.3 测试脚本关联设置	80
6.4.4 调用 ModelSim 仿真	82
第7章 基于板级调试的第二个工程实例	84
7.1 PWM 蜂鸣器驱动——功能概述	84
7.2 PWM 蜂鸣器驱动——引脚分配	86
7.3 PWM 蜂鸣器驱动——综合、映射与配置文件产生	88
7.4 Altera FPGA 配置方式	90
7.4.1 概述	90
7.4.2 配置方式	91
7.5 PWM 蜂鸣器驱动——FPGA 在线下载配置	92

7.6	PWM 蜂鸣器驱动——FPGA 配置芯片固化	94
7.7	PWM 蜂鸣器驱动——复位与 FPGA 重配置功能	98
第 8 章	基础入门实例	100
8.1	蜂鸣器开关实例	100
8.1.1	功能简介	100
8.1.2	代码解析	101
8.1.3	打开工程	101
8.1.4	下载配置操作	102
8.2	流水灯实例	103
8.2.1	功能简介	103
8.2.2	代码解析	104
8.2.3	下载配置	104
8.3	3-8 译码器实例	104
8.3.1	功能简介	104
8.3.2	代码解析	105
8.3.3	板级调试	106
8.4	按键消抖与 LED 开关实例	106
8.4.1	按键消抖原理	106
8.4.2	功能简介	107
8.4.3	代码解析	108
8.4.5	板级调试	110
8.5	经典模式流水灯实例	110
8.5.1	功能简介	110
8.5.2	代码解析	111
8.5.3	板级调试	113
8.6	基于 PLL 分频计数的 LED 闪烁实例	113
8.6.1	PLL 概述	113
8.6.2	功能简介	114
8.6.3	新建 IP 核文件	114
8.6.4	PLL 配置	116
8.6.5	模块化设计概述	120
8.6.6	模块化设计实践	121
8.6.7	代码解析	121
8.6.8	板级调试	123
8.7	数码管驱动实例	123
8.7.1	数码管驱动原理	123
8.7.2	功能概述	124
8.7.3	代码解析	125

8.7.4	板级调试	129
8.8	SRAM 读写测试实例	129
8.8.1	SRAM 读写时序解读	129
8.8.2	功能简介	132
8.8.3	代码解析	133
8.8.4	仿真设置	139
8.8.5	功能仿真	139
8.8.6	FPGA 在线配置	140
8.8.7	触发采样波形	140
8.9	UART loopback 测试	142
8.9.1	功能概述	142
8.9.2	代码解析	143
8.9.3	板级调试	150
8.10	VGA 驱动 ColorBar 显示实例	151
8.10.1	VGA 概述	151
8.10.2	功能简介	154
8.10.3	代码解析	155
8.10.4	板级调试	161
8.11	LCD 基本驱动实例	161
8.11.1	LCD 驱动时序	161
8.11.2	功能简介	163
8.11.3	代码解析	163
8.11.4	装配	167
8.11.5	板级调试	167
8.12	LCD 字符显示驱动实例	168
8.12.1	字符取模	168
8.12.2	ROM 初始化文档创建	171
8.12.3	新建源文件	172
8.12.4	ROM 配置	173
8.12.5	功能简介	176
8.12.6	代码解析	176
8.12.7	板级调试	180
8.13	矩阵按键扫描检测实例	181
8.13.1	键盘概述	181
8.13.2	矩阵按键工作原理	181
8.13.3	功能概述	182
8.13.4	代码解析	183
8.13.5	RTL Viewer	188
8.13.6	State Machine Viewer	190

8.13.7	Technology Map Viewer	191
8.13.8	板级调试	192
第9章	FPGA 片内资源应用实例	193
9.1	基于 SignalTap II 的超声波测距调试实例	193
9.1.1	超声波测距原理	193
9.1.2	功能简介	194
9.1.3	代码解析	194
9.1.4	硬件装配	197
9.1.5	SignalTap II 源文件创建	197
9.1.6	SignalTap II 配置	198
9.1.7	SignalTap II 调试	202
9.2	FPGA 片内 ROM 实例	202
9.2.1	功能概述	202
9.2.2	代码解析	203
9.2.3	ROM 初始化文档创建	205
9.2.4	新建 IP 核源文件	206
9.2.5	ROM 配置	208
9.2.6	功能仿真	211
9.2.7	FPGA 在线调试	211
9.3	FPGA 片内 RAM 实例	212
9.3.1	功能概述	212
9.3.2	代码解析	213
9.3.3	新建 IP 核源文件	215
9.3.4	RAM 配置	216
9.3.5	功能仿真	218
9.3.6	FPGA 在线调试	220
9.4	FPGA 片内 FIFO 实例	221
9.4.1	功能概述	221
9.4.2	代码解析	221
9.4.3	新建 IP 核源文件	224
9.4.4	FIFO 配置	225
9.4.5	功能仿真	228
9.4.6	FPGA 在线调试	229
9.5	FPGA 片内异步 FIFO 实例	230
9.5.1	功能概述	230
9.5.2	代码解析	231
9.5.3	新建 IP 核源文件	234
9.5.4	FIFO 配置	234

9.5.5 功能仿真	236
9.5.6 FPGA 在线调试	238
9.6 FPGA 片内 ROM、FIFO、RAM 联合实例	240
9.6.1 功能概述	240
9.6.2 代码解析	240
9.6.3 功能仿真	244
9.6.4 FPGA 在线调试	245
第 10 章 综合进阶实例	247
10.1 基于数码管显示的超声波测距回响脉宽计数实例	247
10.1.1 功能简介	247
10.1.2 代码解析	248
10.1.3 板级调试	250
10.2 基于均值滤波处理的超声波测距回响脉宽计数实例	250
10.2.1 功能简介	250
10.2.2 滤波算法与实现	251
10.2.3 代码解析	251
10.2.4 板级调试	253
10.3 基于进制换算的超声波测距结果显示实例	253
10.3.1 功能简介	253
10.3.2 距离计算公式实现	253
10.3.3 进制换算实现	254
10.3.4 代码解析	254
10.3.5 乘法器 IP 核创建、配置与例化	256
10.3.6 除法器 IP 核创建、配置与例化	259
10.3.7 板级调试	262
10.4 倒车雷达实例	263
10.4.1 应用背景	263
10.4.2 功能简介	263
10.4.3 代码解析	264
10.4.4 板级调试	266
10.5 基于 SRAM 批量读写的 UART bulk 测试实例	266
10.5.1 功能概述	266
10.5.2 代码解析	267
10.5.3 板级调试	275
10.6 基于数码管显示的 RTC 读取实例	277
10.6.1 RTC 芯片解析	277

10.6.2	功能简介	278
10.6.3	代码解析	279
10.6.4	板级调试	288
10.7	基于 UART 发送的 RTC 读取实例	289
10.7.1	功能简介	289
10.7.2	代码解析	289
10.7.3	板级调试	292
10.8	基于 UART 收发的 RTC 读写实例	293
10.8.1	功能简介	293
10.8.2	代码解析	293
10.8.3	板级调试	296
10.9	基于 UART 控制的 VGA 多模式显示实例	297
10.9.1	功能简介	297
10.9.2	代码解析	298
10.9.3	板级调试	298
10.10	基于 LED 显示的 D/A 输出驱动实例	299
10.10.1	D/A 芯片概述	299
10.10.2	功能简介	300
10.10.3	代码解析	300
10.10.4	板级调试	305
10.11	基于按键调整和数码管显示的 D/A 输出实例	306
10.11.1	功能简介	306
10.11.2	代码解析	306
10.11.3	板级调试	307
10.12	波形发生器实例	307
10.12.1	功能简介	307
10.12.2	代码解析	308
10.12.3	IP 核 CORDIC 配置与例化	312
10.12.4	板级调试	317
10.13	基于数码管显示的 A/D 采集实例	317
10.13.1	A/D 芯片接口概述	317
10.13.2	功能简介	318
10.13.3	代码解析	318
10.13.4	板级调试	322
10.14	A/D 和 D/A 联合测试实例	322
10.14.1	功能简介	322
10.14.2	代码解析	322

10.14.3	板级调试	324
10.15	RTC 时间的 LCD 显示和 UART 设置实例	325
10.15.1	功能简介	325
10.15.2	代码解析	325
10.15.3	板级调试	329
参考文献		330

FPGA 开发入门

本章导读

本章从 FPGA 的一些基本概念入手,将 ASIC、ASSP、ARM、DSP 与 FPGA 同台比对,同时对 FPGA 开发语言及主要厂商进行介绍;接着对 FPGA 技术在嵌入式应用中的优势和局限性进行讨论;最后简要论述 FPGA 的应用领域和开发流程。

1.1 FPGA 基础入门

1. FPGA 是什么

简单来说,FPGA 就是“可反复编程的逻辑器件”。图 1.1 展示了 Altera 公司的 Cyclone V SoC FPGA 器件,从外观上看,它和一般的 CPU 芯片没有太大差别。

FPGA 取自 Field Programmable Gate Array 这四个英文单词的首字母,译为“现场(Field)可编程(Programmable)逻辑阵列(Gate Array)”。1985 年,Xilinx 公司的创始人之一 Ross Freeman 发明了现场可编程门阵列(FPGA)。Freeman 发明的 FPGA 是一块全部由“开放式门”组成的计算机芯片。采用该芯片,工程师可以根据需要进行灵活编程,添加各种新功能,以满足不断发展的协议标准或规范,甚至可以在设计的最后阶段对芯片进行修改和升级。Freeman 当时就推测,这种低成本、高灵活性的 FPGA 将成为各种应用中定制芯片的替代品。也正是由于此项伟大的发明,让 Freeman 于 2009 年荣登美国发明家名人堂。

而至于 FPGA 到底是什么,能够干什么,又有什么过人之处,下面就把它和它的“师兄师弟”们进行比较,给出这些问题的答案。

2. FPGA、ASIC 和 ASSP

抛开 FPGA 不提,大家一定都很熟悉 ASIC 与 ASSP。ASIC,即专用集成电路(Application

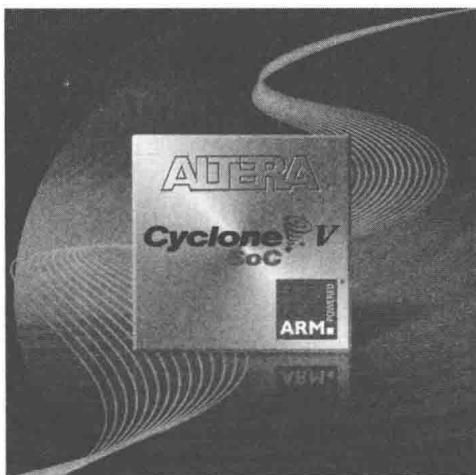


图 1.1 Altera 公司的 Cyclone V SoC FPGA 器件

Specific Integrated Circuit); 而 ASSP, 即专用标准产品 (Application Specific Standard Parts)。电子产品中, 它们无所不在, 还真是比 FPGA 的普及程度高。但是 ASIC 以及 ASSP 的功能相对固定, 是为了专一功能或专一应用领域而生, 希望对它们进行任何的功能和性能的改善往往是无济于事的。打个浅显的比喻, 如图 1.2 所示, 如果说 ASIC 或 ASSP 是布满铅字的印刷品, 那么 FPGA 就是可以自由发挥的白纸。

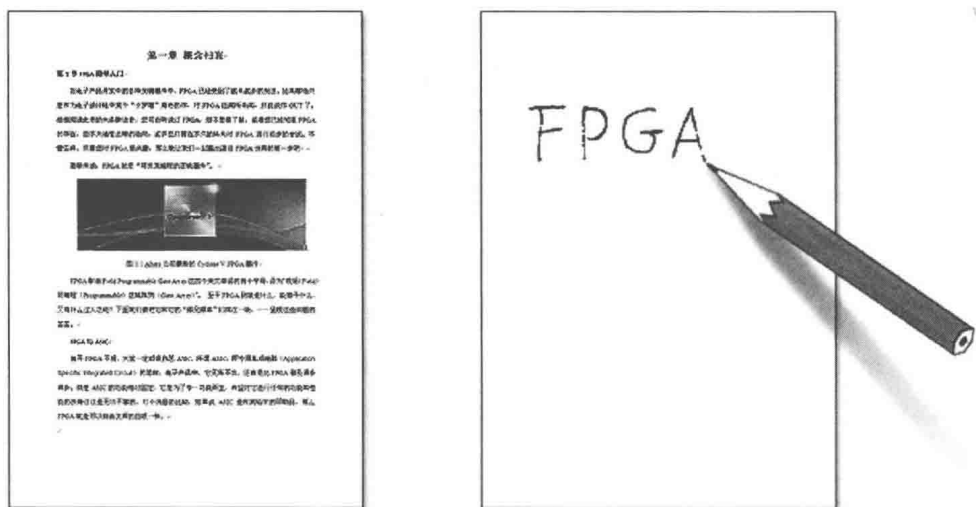


图 1.2 ASIC/ASSP 和 FPGA(如同印刷品和白纸)

使用了 FPGA 器件的电子产品, 在产品发布后仍然可以对产品设计作出修改, 极大地方便了产品的更新以及针对新的协议或标准作出的相应改进, 从而可以加速产品的上市时间, 并降低产品的失败风险和维护成本。相对于无法对售后产品设计进行修改的 ASIC 和 ASSP 来说, 这是 FPGA 特有的一个优势。由于 FPGA 可编程的灵活性以及近年来电子技术领域的快速发展, FPGA 也正在向高集成、高性能、低功耗、低价格的方向发展, 并且逐渐具备了与 ASIC 和 ASSP 相当的性能, 广泛地应用在各行各业的电子及通信设备中。

3. FPGA、ARM 和 DSP

与 ASIC 相比, FPGA、ARM 和 DSP 都具备与生俱来的可编程特性。或许身处开发第一线的底层工程师要说“不”了, 很多 ASIC 不是也开放了一些可配置选项, 实现“可编程”特性了吗? 是的, 但与 FPGA、ARM、DSP 能够“为所欲为”地任意操控整个系统而言, ASIC 的那点“可编程”性的确摆不上台面。当然, 换个角度来看, FPGA、ARM 和 DSP 都或多或少集成了一些 ASIC 功能, 而正是由于这些 ASIC 功能, 加上“可编程”特性, 使得它们相互区别开了, 并且各自独霸一方。

ARM(Advanced RISC Machines)是微处理器行业的一家知名企业, 设计了大量高性能、廉价、耗能低的 RISC 处理器、相关技术及软件。由 ARM 公司设计的处理器风靡全球, 大有嵌入式系统无处不 ARM 的趋势。通常所说的 ARM, 更多的是指 ARM 公司的处理器, 即 ARM 处理器, 如图 1.3 所示。ARM 通常包含一颗强大的处理器内核, 并且为这颗处理器量身配套了很多成熟的软件工具以及高级编程语言, 这也是它备受青睐的原因之一。当然, ARM 不只是一颗处理器, 因为在 ARM 内核处理器周边, 各种各样精于控制的外设比比皆是, 如 GPIO、PWM、ADC/DAC、UART、SPI、IIC 等。ARM 善于控制和管理, 在很多

工业自动化中大有用武之地。

DSP(Digital Singnal Processor,数字信号处理器)是一种独特的微处理器,有自己的完整指令系统,能够进行高速、高吞吐量的数字信号处理。DSP 只专注一件事,就是对各种语音、数据和视频做运算处理;或者也可以说,DSP 是为各种数学运算量身打造的。图 1.4 展示了 DSP 处理器。

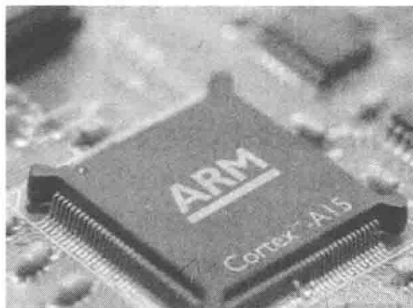


图 1.3 ARM 处理器



图 1.4 DSP 处理器

相比之下,套用近些年业内比较流行的一句广告词“All Programmable”来形容 FPGA(见图 1.5)再合适不过了。虽然 ARM 有很多外设,DSP 具备强大的信号运算能力,但在 FPGA 眼里,这些都不过是“小菜一碟”。这样说或许有些过了,但毫不夸张地讲,ARM 和 DSP 能做的,FPGA 也都能做;而 FPGA 可以做的,ARM 却不一定行,DSP 也不一定行。这就是在很多原型产品设计过程中,时不时有人会提出基于 FPGA 的方案了。在一些灵活性要求高、定制化程度高、性能要求也特别高的场合,FPGA 再合适不过了,甚至有时会是设计者唯一的选择。当然了,客观地来看,FPGA 固然强大,但它高高在上的成本、功耗和开发复杂性还是会让很多潜在的目标客户望而却步。而在这些方面,ARM 和 DSP 正好弥补了 FPGA 所带来的缺憾。

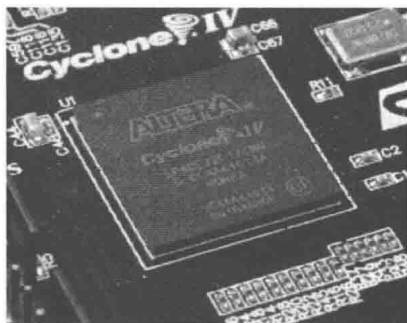


图 1.5 FPGA 器件

总而言之,在嵌入式系统设计领域,FPGA、ARM 和 DSP 互有优劣,各有所长。很多时候它们所实现的功能无法简单地相互替代,否则就不会见到如 TI 的 DaVinci 系列 ARM 中有 DSP、Xilinx 的 Zynq 或 Altera 的 SoC FPGA 中有 ARM 的共生现象了。FPGA、ARM 和 DSP,它们将在未来很长的一段时间内呈现三足鼎立的局面。

4. Verilog 与 VHDL

说到 FPGA,读者一定关心它的开发方式。FPGA 开发本质上就是一些逻辑电路的实现而已,因此早期的 FPGA 开发通过绘制原理图(和现在的硬件工程师绘制原理图的方式大体相仿)完成。而随着 FPGA 规模和复杂性的不断攀升,这种落后的设计方式几乎已经被大家遗忘了,取而代之的是能够实现更好的编辑性和可移植性的代码输入方式。

说到 FPGA 的设计代码,经过近三十年的发展,只有 Verilog 和 VHDL 二者最终脱颖而出,成为了公认的行业标准。对于这两种不同的语法,它们的历史渊源、孰优孰劣这里

就不提了。美国和中国台湾地区的逻辑设计公司大都以 Verilog 语言为主,国内目前学习和使用 Verilog 的人数也在逐渐超过 VHDL 的用户。从学习的角度来讲,Verilog 相对 VHDL 有着快速上手、易于使用的特点,因此博得了更多工程师的青睐。即便是从来没有接触过 Verilog 的初学者,只要凭着一二 C 语言的底子加上一些硬件基础,很快就可以熟悉 Verilog 语法。当然,仅仅是入门还是远远不够的,真正掌握 Verilog 必须花费很多时间和精力,加上一些项目的实践,才会慢慢对可编程逻辑器件的设计有更深入的理解和认识。

5. Altera、Xilinx 和 Lattice

相对于互联网的那些“暴发户”,半导体行业则更讲究历史底蕴,“今天丑小鸭,明天白天鹅”的故事要少得多,因此,两家历史最为久远的 FPGA 供应商 Altera 和 Xilinx 凭着一贯以来的专注,确保了它们在这个行业的统治地位。当然,很大程度上也是由于 FPGA 技术相对于一般的半导体产品有着更高的门槛,从器件本身到一系列配套的工具链,再到终端客户的技术支持——这一箩筐的麻烦事,让那些行业“大佬们”想想就头疼,更别提“插足捣腾”一下了。

目前,FPGA 器件的主流厂商 Altera 公司(已被 Intel 公司收购)和 Xilinx 公司的可编程逻辑器件占到了全球市场的 60% 以上。从明面上的“竞争对手”到今天暗地里还客气地互称“友商”,不难看出两家公司虽然有“明争暗斗”,但确实也不经意间地彼此促进,互相激励。的确,FPGA 的发展史充斥着这两家公司不断上演的“你方唱罢我登场”的情节,并且偶尔也会有第三者(如 Lattice 小弟)的“插足”戏份。经过多年发展,各方重新定位,Altera 和 Xilinx 已牢牢把持住象征统治地位的中高端市场,而 Lattice 也只能在低端市场中发挥作用。

不论是 Altera、Xilinx 还是 Lattice,甚至一些后来者,如国内 FPGA 厂商京微雅格,它们的 FPGA 器件内部结构虽然略有差异,但在开发流程、开发工具乃至原厂提供的各种支持上,都是“换汤不换药”。所以,这对用户而言,绝对是一个福音——只要掌握一套方法论,则对任何厂商的器件都可以游刃有余。

1.2 FPGA 的优势在哪里

若要准确评估 FPGA 技术能否满足开发产品的功能、性能以及其他各方面的需求,深入理解 FPGA 技术是至关重要的。在产品的整个生命周期中,如果产品功能必须进行较大的升级或变更,那么使用 FPGA 技术来实现就会有很大的优势。

在考虑是否使用 FPGA 技术来实现目标产品时,需要重点从以下几方面进行评估。

- 可升级性——产品在设计过程中,甚至将来产品发布后,是否有较大的功能升级需求? 是否应该选择具有易于更换的同等级、不同规模的 FPGA 器件?
- 开发周期——产品开发周期是否非常紧迫? 若使用 FPGA 开发,是否比其他方案具有更高的开发难度? 能否面对必须在最短的时间内开发出产品的挑战?
- 产品性能——对产品的数据速率、吞吐量或处理能力是否有特殊要求? 是否应该选择性能更好或速度等级更快的 FPGA 器件?