

可编程逻辑器件快速进阶丛书

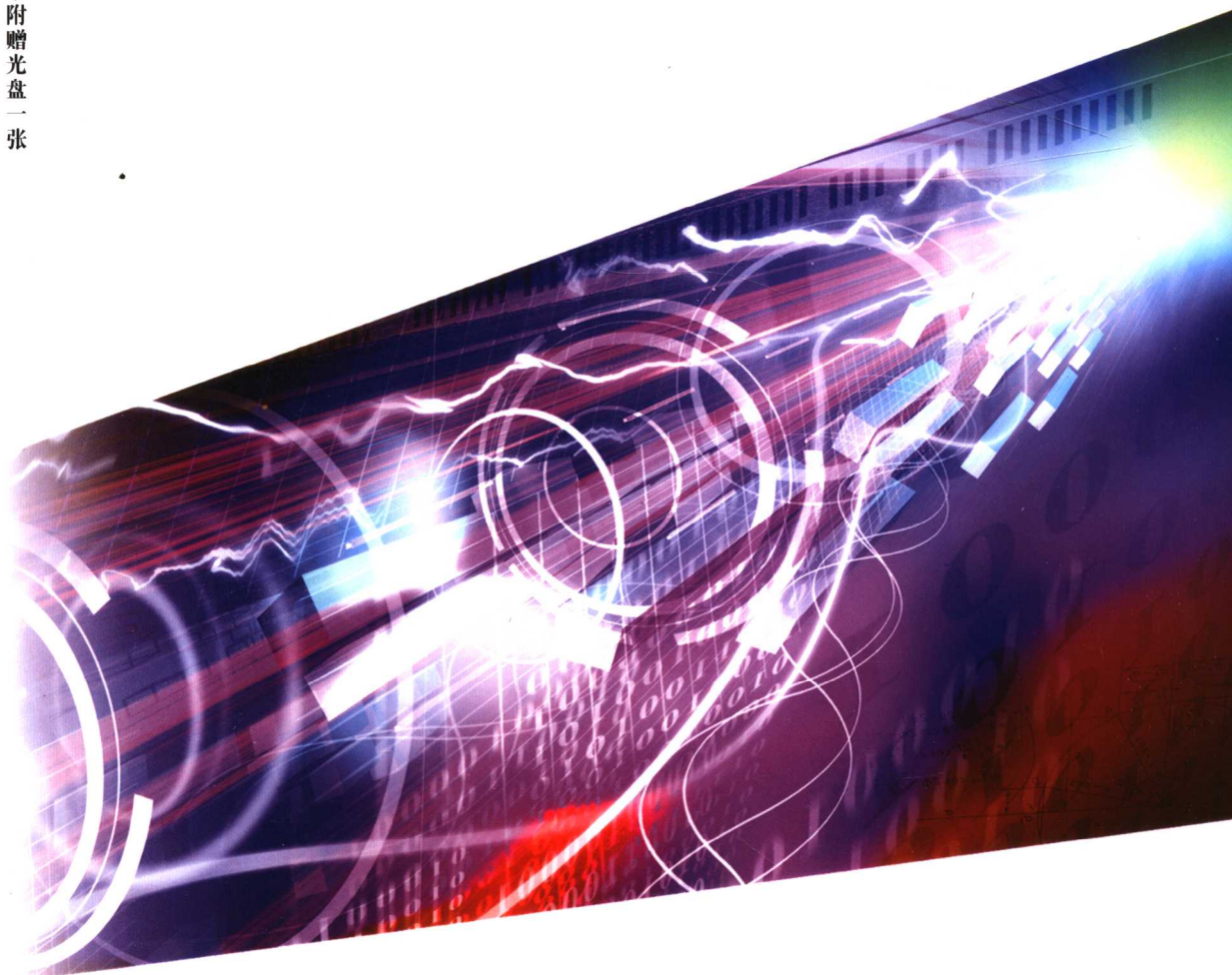
可编程逻辑器件开发软件

Quartus II



随书附赠光盘一张

郑亚民 董晓舟 编著



國防工業出版社

National Defense Industry Press

可编程逻辑器件快速进阶丛书

可编程逻辑器件开发 软件 Quartus II

郑亚民 董晓舟 编著

国防工业出版社

·北京·

内 容 简 介

本书围绕着 PLD 的设计这个主题,系统介绍了相关的基础知识和工具软件,给出了设计实例。本书的主要内容包括:可编程逻辑器件结构、Altera 公司最新系列器件的性能特点、VHDL 语言、Altera 公司新一代的 PLD 开发软件 Quartus II 使用入门与进阶、第三方工具软件、LogicLock 设计方法、DSP Builder 设计工具、SOPC Builder 与 Nios II 嵌入式处理器核的软硬件开发。

本书内容新颖全面、叙述简明清晰、结构层次分明,利用大量实例和图表说明问题,使读者易于接受。

本书既可以作为高年级本科生和研究生的 EDA 设计方法等课程的教材,也可以作为工程技术人员的参考资料。

为了方便读者,本书附光盘一张,其中包含了所有设计实例的源程序、工程等。

图书在版编目(CIP)数据

可编程逻辑器件开发软件 Quartus II / 郑亚民, 董晓
舟编著. —北京:国防工业出版社, 2006. 9
(可编程逻辑器件快速进阶丛书)
ISBN 7-118-04711-2

I. 可... II. ①郑... ②董... III. 可编程逻辑器件
—应用软件, Quartus 2 IV. TP332.1

中国版本图书馆 CIP 数据核字(2006)第 092423 号

※

国防工业出版社出版发行

(北京市海淀区紫竹院南路 23 号 邮政编码 100044)

北京市李史山胶印厂印刷

新华书店经售

*

开本 787×1092 1/16 印张 18½ 字数 427 千字

2006 年 9 月第 1 版第 1 次印刷 印数 1—4000 册 定价 36.00 元(含光盘)

(本书如有印装错误,我社负责调换)

国防书店:(010)68428422

发行邮购:(010)68414474

发行传真:(010)68411535

发行业务:(010)68472764

前 言

近几年来 PLD 技术的飞速发展,使得电子电路的设计方法发生了根本性变化,基于 PLD 的 EDA 技术在电子信息、通信、自动控制等领域的作用越来越重要。EDA 技术以现代电子计算机为设计工具和设计平台,以硬件描述语言为系统逻辑描述的主要手段,自动、高速地完成数字系统的逻辑综合、仿真模拟和布局布线等工作。使用基于 EDA 技术的 PLD 来开发数字系统,可以大大缩短设计时间,提高系统的可靠性。PLD 是目前电子设计领域中最具活力和发展前途的,它的影响丝毫不亚于 20 世纪 70 年代单片机的发明和使用。

本书第 1 章围绕着 PLD 设计这个主题,系统介绍了相关的基础知识和工具软件使用方法。第 2 章介绍了 PLD 的硬件基础知识,展示了 PLD 的发展脉络,分析了各个时期 PLD 的结构特点和应用局限,重点介绍了 CPLD/FPGA 器件的结构特点和应用优势,详细介绍了 Altera 公司最新系列器件的性能特点。第 3 章讲解了在 PLD 设计中必须使用到的硬件描述语言的基础知识,重点介绍了目前使用最为广泛的 VHDL 语言。由于篇幅所限,本章未对 VHDL 语言的语法作详尽的理论论述,而是结合许多实际例子说明了 VHDL 语言的基础应用,力求做到简明扼要、浅显易懂。第 4 章介绍了与 PLD 的设计密切相关的开发软件 Quartus II。Quartus II 是 Altera 公司新一代的 PLD 开发软件,适合大规模 CPLD/FPGA 的开发。本章详细地讲解了 Quartus II 软件的使用步骤、使用技巧和注意事项。结合实际设计例子,每一个设计步骤均配有图示,力求操作完整、详尽清晰,使读者能尽快入门。第 5 章是第 4 章的进阶,对 Quartus II 软件更深层次的应用作了详细讲解,重点介绍了各种技术细节,使读者对 Quartus II 软件有更深入的了解,并掌握更高层次的应用。第 6 章主要讲述 Quartus II 软件与第三方 EDA 工具软件的相互关系,重点介绍了 Synplify 和 ModelSim 软件的特点、设计流程及使用方法。第 7 章介绍了 LogicLock 技术在模块化设计中的实现过程。第 8 章和第 9 章分别讲述了 Altera 的系统级设计工具 DSP Builder 和 SOPC Builder 的使用方法,对 Nios II 嵌入式处理器系统的软、硬件设计过程进行了详细的描述。

本书的编写者一直跟踪 PLD 设计的发展,并将 PLD 设计方法用于科研和指导大学生电子设计竞赛,积累了一定的经验。在实践中,深切感受到市场上专门讲解 PLD 设计方面的书籍虽然有一些,但大部分纠缠于理论或操作的枯燥罗列,缺乏简单实用、清晰明了的步骤介绍,使读者难以入门。本书的叙述风格是避免枯燥的理论和步骤讲述,而以实例带出理论,以实例带出操作步骤和方法,力求使所述内容形象、实用,给读者展现一个清晰、完整的设计过程。从入门到进阶、从简单设计到复杂设计,层次分明。既适于入门级的读者,也适于有一定基础有待于提高的读者。本书既可以作为高年级本科生和研究生

EDA 设计方法等课程的教材,也可以作为工程技术人员的参考资料。

本书的编写由郑亚民、董晓舟完成。郝昶峰、侯续霞、郝春芳等同学对本书的例题及文字的输入作了部分工作,在此表示感谢。

由于作者水平有限,书中难免会有不足之处,敬请读者批评指正。

作者

目 录

第 1 章 概述	1
1.1 EDA 技术	1
1.1.1 EDA 技术发展	1
1.1.2 EDA 设计方法	2
1.1.3 EDA 设计层次	3
1.2 可编程逻辑器件概述	4
1.2.1 可编程逻辑器件发展	4
1.2.2 可编程逻辑器件常用开发软件	5
1.2.3 可编程逻辑器件设计流程	5
第 2 章 FPGA/CPLD 结构与应用	8
2.1 可编程逻辑器件基础	8
2.1.1 逻辑电路符号表示方法	8
2.1.2 可编程逻辑器件的分类	8
2.1.3 简单 PLD 原理	9
2.2 复杂可编程逻辑器件	12
2.2.1 CPLD 结构与原理	12
2.2.2 FPGA 结构与原理	15
2.2.3 CPLD 和 FPGA 特点总结	20
2.2.4 FPGA/CPLD 的发展趋势	21
2.3 Altera 新型器件简介	22
2.3.1 低成本 FPGA——Cyclone II	22
2.3.2 高性能、高密度 FPGA——Stratix II	26
2.3.3 低成本、低功耗 CPLD——MAX II	34
2.4 编程与配置	37
2.4.1 Altera FPGA 的配置方式	37
2.4.2 配置过程	38
2.4.3 Altera 配置器件	39
2.4.4 ByteBlaster II 下载电缆	41
2.4.5 Quartus II 软件支持	42
第 3 章 VHDL 基础	46
3.1 简介	46
3.1.1 VHDL 的出现	46
3.1.2 VHDL 的主要优点	46

3.2	VHDL 程序基本结构	47
3.2.1	实体描述	48
3.2.2	结构体描述	50
3.3	VHDL 的库和包	52
3.3.1	VHDL 库的种类和使用	52
3.3.2	程序包	52
3.3.3	库和程序包的引用	53
3.4	VHDL 的基本词法	53
3.4.1	标识符	53
3.4.2	数据对象	54
3.4.3	数据类型	56
3.4.4	运算符	58
3.4.5	VHDL 表达式	60
3.5	VHDL 的基本语法	60
3.5.1	并行描述语句	60
3.5.2	顺序描述语句	63
3.5.3	结构描述语句	69
3.6	简单逻辑电路的 VHDL 描述	71
3.6.1	译码器	71
3.6.2	三态门	72
3.6.3	电位型触发器	73
3.6.4	钟控型触发器	74
第 4 章	Quartus II 使用入门	76
4.1	Quartus II 简介	76
4.1.1	Quartus II 的设计流程	76
4.1.2	Quartus II 的设计特点	77
4.1.3	Quartus II 的图形用户界面	78
4.2	Quartus II 使用方法	79
4.2.1	设计输入	79
4.2.2	编译	85
4.2.3	仿真	87
4.2.4	配置器件	90
4.2.5	原理图输入方式	92
4.3	使用 Quartus II 设计数字逻辑	99
4.3.1	层次化设计方法——3 分频器设计	99
4.3.2	有限状态机设计——A/D 采样控制器设计	105
第 5 章	Quartus II 使用进阶	114
5.1	设计输入与约束	114
5.1.1	预备知识	115

5.1.2	设计输入	118
5.1.3	设计约束	146
5.2	编译	148
5.2.1	综合	150
5.2.2	布局布线	153
5.3	仿真与硬件验证	155
5.3.1	仿真	155
5.3.2	SignalTap II 在系统分析工具	158
5.4	Altera 的 IP Core	161
第 6 章	Quartus II 与第三方 EDA 工具	164
6.1	Quartus II 对第三方 EDA 工具的支持	164
6.1.1	Quartus II 支持的第三方 EDA 工具	164
6.1.2	Quartus II 与第三方 EDA 工具协同设计流程	165
6.2	Synplify/Synplify Pro 综合工具	165
6.2.1	Synplify Pro 驱动的设计流程	165
6.2.2	Quartus II 驱动的设计流程	173
6.3	ModelSim 仿真工具	175
6.3.1	使用 ModelSim 进行功能仿真	176
6.3.2	Altera 仿真库的指定	183
6.3.3	ModelSim 与 Quartus II 结合的仿真流程	183
第 7 章	LogicLock 设计方法	186
7.1	LogicLock 简介	186
7.1.1	LogicLock 区域	187
7.1.2	反标注布局布线信息	188
7.2	应用 LogicLock 方法设计流水线乘法器	189
7.2.1	开始前的准备	189
7.2.2	创建 LogicLock 区域	190
7.2.3	指定 LogicLock 区域的逻辑内容	194
7.2.4	反标注 LogicLock 区域	195
7.2.5	导出 LogicLock 约束	197
7.2.6	在顶层设计中使用导入 LogicLock 约束信息	198
7.2.7	查看 LogicLock 设计结果	201
第 8 章	DSP Builder 设计工具	203
8.1	DSP Builder 设计流程	203
8.1.1	Matlab/Simulink 简介	203
8.1.2	DSP Builder 设计流程	203
8.2	简单的 DSP Builder 设计实例	205
8.2.1	设计内容	205
8.2.2	建立 Simulink 模型	205

8.2.3	系统级仿真	213
8.2.4	使用 ModelSim 进行 RTL 仿真	216
8.2.5	自动完成综合适配流程	220
8.2.6	手动流程下的设计综合	221
8.2.7	使用 Quartus II 进行时序仿真	224
8.2.8	将 DSP Builder 设计作为模块调用	226
8.3	DSP Builder 设计规则	227
8.3.1	DSP Builder 命名规范	227
8.3.2	位宽设计规则	227
8.3.3	关于数据转换的说明	228
8.3.4	频率设计规则	229
8.3.5	Goto 和 From 模块	234
8.3.6	层次化设计	234
8.4	在 DSP Builder 中使用 IP 核	236
第 9 章	Nios II SOPC 嵌入式系统设计	238
9.1	Nios II 处理器简介	238
9.1.1	Nios II 软核处理器的特性	238
9.1.2	使用 Nios II 实现 SOPC 的特点	240
9.2	简单 SOPC 设计实例	243
9.2.1	SOPC 设计流程	243
9.2.2	设计内容及步骤	244
9.2.3	建立 Quartus II 工程	245
9.2.4	使用 SOPC Builder 创建 NiosII 硬件系统	246
9.2.5	编译 Nios II 系统并配置到目标器件	254
9.2.6	使用 Nios II IDE 开发软件系统	257
9.3	基于 HAL 的软件开发	262
9.3.1	使用 HAL 开发 Nios II 软件程序	263
9.3.2	HAL 设计实例	264
9.4	MicroC/OS-II 基础	268
9.5	用户自定义元件	272
9.5.1	用户自定义元件开发流程	272
9.5.2	自定义 PWM 元件	274
9.6	用户自定义指令	280
9.6.1	用户自定义指令开发流程	280
9.6.2	自定义指令实例	281
参考文献		287
光盘说明		288

第 1 章 概 述

1.1 EDA 技 术

1.1.1 EDA 技术发展

随着微电子设计技术与工艺的发展,电子器件的发展经历了从电子管、晶体管、小规模集成电路、中规模集成电路到大规模集成电路和超大规模集成电路各个阶段,整个电子器件的发展趋势是体积越来越小,集成度越来越高,功能越来越强大。现代集成电路发展到了今天的专用集成电路(ASIC, Application Specific Integrated Circuit),已经能够实现单片电子系统(SOC, System on a Chip)的功能。采用 ASIC 设计降低了产品的生产成本,提高了系统的可靠性,缩小了设计的物理尺寸,不足之处是一次性投资大,设计周期长,改版规模大,灵活性差。因而,近几年来出现了一种半定制电路,即可编程逻辑器件(PLD, Programmable Logic Device)。

使用 PLD 开发数字电路,可以大大缩短设计时间,提高系统的可靠性。PLD 是电子设计领域中最具活力和发展前途的一项技术,它的影响丝毫不亚于 20 世纪 70 年代单片机的发明和使用。那么 PLD 能做什么呢?毫不夸张地讲,PLD 能完成任何数字器件的功能,上至高性能的 CPU,下至简单的 74 系列电路,都可以用 PLD 来实现。PLD 如同一张白纸或是一堆积木,工程师可以通过传统的原理图输入法,或是硬件描述语言(HDL, Hardware Description Language)自由地设计一个数字系统。通过软件仿真,可以事先验证设计的正确性。在印制电路板(PCB, Print Circuits Board)完成以后,还可以利用 PLD 的在线修改能力,随时修改设计而不必改动硬件电路。

PLD 技术的飞速发展,使得电子电路的设计方法也发生了根本性变化,出现了所谓电子设计自动化(EDA, Electronic Design Automation)。EDA 就是以现代电子计算机为设计工具和设计平台,以硬件描述语言为系统逻辑描述主要手段,自动、高速地完成数字系统的逻辑综合、仿真模拟和布局布线等工作的设计技术和设计工具。

采用 EDA 技术进行数字系统设计是可编程逻辑器件与普通数字逻辑器件的本质差别。而利用硬件描述语言编程来描述系统硬件功能是该设计方法的重要特征。利用 EDA 技术,设计者只需要利用硬件描述语言编程完成对系统硬件功能的描述,就可以用计算机软件进行处理,代替人来完成数字系统的逻辑综合、仿真模拟和布局布线等功能。设计方法由手动设计变为计算机自动设计,大大提高了设计效率和设计质量,缩短了设计周期;不需改变设计硬件线路,只需改变软件设计就可以改正设计中出现的错误,既经济又方便。

显然,EDA 技术的发展与大规模集成电路设计技术、计算机辅助设计、可编程逻辑器件的发展同步。大致分为三个阶段。

1. 20 世纪 70 年代的 CAD 阶段

即计算机辅助设计(CAD, Computer Aided Design)阶段。出现了在计算机上进行电子系统电路图的输入、存储、修改及印制电路板版图设计等一些单独的软件工具,使人们摆脱了手工进行电子设计时的大量繁重、重复、单调计算与绘图工作,计算机取代人工进行电子系统的设计、分析与仿真。但是对于复杂的设计,当时的 EDA 技术不能提供系统级的仿真与综合,设计错误不能在开发初期发现,给设计修改带来了不便。

2. 20 世纪 80 年代的 CAE 阶段

即计算机辅助工程(CAE, Computer Aided Engineering)阶段。伴随着计算机和集成电路的发展,EDA 技术进入到计算机辅助工程 CAE 设计阶段。计算机采用各种软件仿真分析工具,将电路图输入、编译与连接、逻辑模拟、仿真分析、版图自动生成及各种单元库都集成在一个 CAE 系统中,对所设计的电子电路及系统进行功能验证或时序分析等,实现了电子系统或芯片从原理图输入到版图设计输出的全程设计自动化,节省了设计时间和设计费用,提高了设计效率。利用这些工具,设计师能在产品制作之前预知产品的功能与性能,能生成产品制造文件,使设计阶段对产品性能的分析前进了一大步。

3. 20 世纪 90 年代的 ESDA 阶段

即电子系统设计自动化(ESDA, Electronic System Design Automation)阶段。随着微电子技术的发展,速度更快、容量更大、功能更强的 PLD 的不断推出,对数字电子系统的设计提出了更高的要求。出现了以利用硬件描述语言、系统仿真和综合技术为特征的第三代 EDA 技术。其特点是在功能强大的 EDA 工具(包括系统行为级描述与结构级综合、系统仿真与测试验证、系统划分与指标分配、系统决策与文件生成等一整套设计工具)软件平台上,以系统级设计为核心,使用硬件描述语言进行系统设计,自动进行逻辑编译、仿真、优化、综合、布线、测试等工作,完成系统设计功能的硬件实现。使得设计者从繁杂的工作中解放出来,把精力集中在系统方案的设计上,是一种高效率的现代设计方法。

1.1.2 EDA 设计方法

1. 自下而上(bottom-up)的设计方法

传统的设计方法一般是根据系统的要求,首先画出整个系统的硬件流程图,再根据功能划分成不同的模块,手工画出一张张的电路原理图,然后根据原理图制作印制电路板,每个功能模块都调试通过后,再把各个模块连接起来进行系统的调试。完成硬件设计以后才能进行整个系统的仿真、调试,系统设计中的问题在调试后期才能发现。如果出现设计中没有考虑到的问题,就要在从底层重新设计,所以设计周期比较长。传统的设计过程是从底层开始,并在已有的功能模块的基础上来搭建高层次的模块直至整个系统,因此称为是自下而上的设计。

2. 自顶向下(top-down)的设计方法

现代 EDA 解决方案是一种采用自顶向下的设计方式。即按照数字系统的功能描述,把系统划分为若干个功能模块,然后再把每个模块化分为不同层次,从高层次到低层次逐步细化,称之为自顶向下的设计方式。自顶向下的设计方法有许多突出的优点:

1) 它克服了大规模电子系统高复杂度所带来的问题,系统可以层次式地划分为易于处理的子系统,再层次式地求精。

- 2) 各子系统可以给设计组中的成员同时设计,也加快了设计速度。
- 3) 设计错误可以在早期发现,极大地减少了设计的迭代次数。
- 4) 逻辑综合优化之前的设计工作和具体采用什么工艺生产芯片是无关的,因而设计的可移植性好,当要采用新的工艺时,可以直接从综合开始。
- 5) 自顶向下的设计方法增加了一次性设计成功的可能性。

3. 硬件描述语言的应用

对于简单的逻辑可以直接采用原始逻辑图或布尔方程输入法进行电路设计。但是对于复杂的系统设计,就必须依靠一种高层次的逻辑输入方式,这样就产生了硬件描述语言。利用硬件描述语言编程来表示逻辑器件及系统硬件的功能和行为,是 EDA 设计方法的一个重要特征。

硬件描述语言 HDL 是为了满足逻辑设计过程中的各种需求而设计的。首先,它能形象化地抽象表示电路的结构和行为,支持逻辑设计中层次与领域的描述。它把实际的硬件设计用语言的方式来描述,把复杂的电路设计用形象化的语言方式表示出来。它可以描述硬件电路的功能、信号连接关系以及定时关系,比电路原理图更能有效地表示电路特征。其次,支持电路描述由高层到低层的综合和转换,便于文档管理,易于理解 and 设计重用。最后, HDL 具有电路仿真与验证功能,可以保证设计的正确性。用户甚至不必编写任何测试向量便可以进行源代码级的调试。而且,设计者可以非常方便地比较各种方案之间的可行性及其优劣,而不需做任何实际的电路实验。

HDL 能够精确而且简明地描述数字电子系统,可用于从系统级到门级的描述,特别是能以非常抽象的形式反映出系统最本质的性能。HDL 可为系统级模拟提供方便的手段,使得系统实现之前就可以评价系统的性能。鉴于 HDL 具有以上诸多优点,只要开发者拥有 Pascal、C 等计算机高级语言的基础,便可以轻松地掌握 HDL,使硬件工作软件化。

硬件描述语言有多种,目前广泛应用的是 HDL 语言。VHDL 语言的全称是“超高速集成电路硬件描述语言”(Very High Speed Integrated Circuit Hardware Description Language),是硬件描述语言中的一种,对系统硬件的描述功能很强而语法规则又比较简单。

可以说,当今的数字系统设计离不开可编程逻辑器件和 EDA 设计工具,EDA 已经成为电子系统的主要设计手段。没有 EDA 技术的支持,要想完成复杂的电子系统和大规模集成电路的设计与制造是不可想象的。反过来,生产制造技术的不断进步又对 EDA 技术提出新的要求,大大推动了 EDA 软件和硬件描述语言的进步。

1.1.3 EDA 设计层次

一个完整的数字电子系统或集成电路从问题的提出到最后的物理实现,可以分为如下几个设计层次,即版图级、电路级、逻辑门级、寄存器传输级(RTL 级)、行为级、系统级。

1. 版图级

也称为物理级,是集成电路描述的最低层次。在版图级,以几何图形描述晶体管、MOS 管、二极管、电阻、连线等元件,利用器件的物理特性关系描述硬件的功能。在这一层次,系统的特性不仅与器件的互连方式有关,而且与器件的加工工艺有关。

2. 电路级

版图级以上的层次是电路级。在电路级,基本的元件是晶体管、MOS管、二极管和电阻电容等,系统的功能由电路的方程描述。

3. 逻辑门级

电路级上面的层次是逻辑门级,也简称为门级,这是数字系统的主要层次。门级设计中的基本单元是与门、或门、非门、三态门等各种门电路,以及少量的触发器。门电路的互连方式构成了门级的结构描述,逻辑图和布尔方程是这一层次的主要描述形式。门级描述也是硬件描述语言的最低层次。

4. 寄存器传输级

门级之上的层次是寄存器传输级,这一层次的描述较之门级描述更为抽象。在实时逻辑(RTL, Real Time Logic)级的设计过程中,需要大量采用触发器、寄存器、计数器、多路选择器、算术逻辑运算单元(ALU)等功能块,这些功能块的规模比基本门电路单元要大得多。RTL级设计的结构描述各功能模块的互连,反映其功能的是真值表和状态图。

5. 算法级

寄存器传输级之上的层次是行为级,又称为算法级,这一层次对系统功能的描述是抽象的算法模型,或者是控制流图和数据流图。

6. 系统级

描述电子系统功能的最高层次是系统级,在这一层次,系统行为的描述往往是一些性能指标,例如计算机的总线宽度、每秒执行指令的次数、数据的传输速率等。

一般地说,把门级以上各层次的设计称为前端设计,而把版图设计以后的工作称为后端设计。前端设计与工艺无关,而后端设计必须在某一确定的生产工艺下进行。

1.2 可编程逻辑器件概述

1.2.1 可编程逻辑器件发展

广义上讲,可编程逻辑器件(PLD)是指一切通过软件手段更改、配置器件内部连接结构和逻辑单元,完成既定设计功能的数字集成电路。从简单到复杂,可编程逻辑器件经历了一个很长的发展过程,目前已经能够支持在一个芯片上集成一个完整的系统。

早期使用较多的是可编程阵列逻辑(PAL, Programmable Array Logic)和通用可编程阵列逻辑(GAL, Generic Array Logic)。目前使用较多的可编程逻辑器件是现场可编程门阵列(FPGA, Field Programmable Gate Array)和复杂可编程逻辑器件(CPLD, Complex Programmable Logic Device)。现在CPLD和FPGA的芯片的容量远远大于PAL和GAL,更适合集成电路的设计。

1. PAL/ GAL

PAL/ GAL是早期可编程逻辑器件的发展形式,其特点是大多基于E²PROM工艺,结构较为简单,可编程逻辑单元多为与、或阵列,可编程单元密度较低,仅能适用于某些简单的数字逻辑电路。虽然PAL/ GAL密度较低,但是它们一出现即以其低功耗、低成本、高可靠性、软件可编程、可重复更改等特点引发了数字电路领域的巨大震动。目前仍然使

用 GAL 等可编程逻辑器件设计简单的数字逻辑电路,越来越多的 74 系列逻辑电路被 GAL 取代。GAL 等器件经历了近 20 年的发展,新一代的 GAL 以功能灵活、小封装、低成本、重复可编程、应用灵活等优点仍然在数字电路领域扮演着重要的角色。

2. FPGA/ CPLD

CPLD 是在 PAL、GAL 的基础上发展起来的。一般采用 E²PROM 工艺,也有少数采用 Flash 工艺,为乘积项结构即“与-或阵列”器件。CPLD 可实现的逻辑功能比 PAL、GAL 有了大幅度的提升,一般可以完成设计中较复杂、较高速度的逻辑功能,如接口转换、总线控制等。FPGA 一般采用 SRAM 编程工艺,由查找表组成可编程门,再构成阵列形式。FPGA 的集成度很高,其器件密度从数万系统门到数千万系统门不等,可以完成极其复杂的时序与组合逻辑电路功能,适用于高速、高密度的高端数字逻辑电路设计领域。

市场上 CPLD 和 FPGA 品种很多,使用较多的是 Altera、Xilinx 和 Lattice 这 3 个公司的产品,各公司都有不同型号的 CPLD 和 FPGA 产品,结构不同,使用的下载电缆线不一样,设计软件也不同。但共同特点是:都可以在系统进行编辑加载程序,不需要使用专用的编程器,它们都是直接将实验系统和计算机的并行口连接,通过运行软件对芯片进行下载,在实验室中就可将大量的数字电路设计集成到一个大芯片中。CPLD 和 FPGA 的主要不同是:CPLD 芯片一经上电加载即已完成编程,不必每次上电重新进行加载,也就是当程序烧入芯片后,只有下一次需要再修改程序时才需要对芯片重新进行加载,否则将不会改变先前所烧入的代码,类似于大的 GAL 芯片;而 FPGA 每次上电使用时,不管是否改变程序都要对芯片进行加载。

1.2.2 可编程逻辑器件常用开发软件

EDA 的核心是利用计算机自动完成电子系统的完整设计,因此 EDA 设计软件是必不可少的。目前,各种开发软件已经发展得相当完善,例如 Altera 公司上一代的 PLD 开发软件 Maxplus II,使用者众多,但是 Altera 已经停止开发 Maxplus II,而转向新一代 Quartus II 软件平台。Quartus II 适合更大规模 CPLD/FPGA 的开发。其他相关软件,例如 DSP Builder 软件是 Quartus II 与 Matlab/Simulink 的接口,利用 IP 核在 Matlab/Simulink 中快速完成数字系统的仿真和 FPGA 的最终实现;SOPC Builder 软件配合 Quartus II,可以完成 Nios II 软 CPU 的开发工作。

1.2.3 可编程逻辑器件设计流程

一般来说,完整的 FPGA/CPLD 设计流程包括电路设计与输入、功能仿真、综合、综合后仿真、实现、布线后仿真与验证、板级仿真验证与调试等步骤。具体阐述如下。

1. 电路设计与输入

常用的设计输入方法有硬件描述语言和原理图设计输入方法等。原理图设计的优点是直观、便于理解,元器件库资源丰富。但是在大型设计中,这种方法的可维护性较差,不利于模块构造与重用;主要的缺点是当所选用芯片升级换代后,所有的原理图都要做相应的改动。进行大型工程设计时最常用的设计方法是硬件描述语言设计输入法,其特点是自顶向下设计,利于模块的划分与复用,可移植性好,通用性好,设计不因芯片的工艺和结构的不同而变化,更利于向 ASIC 的移植。波形输入和状态机输入方法是两种常用的辅

助设计输入方法:使用波形输入法时,只要绘制出激励波形和输出波形,EDA 软件就能自动地根据响应关系进行设计;使用状态机输入法时,设计者只需要画出转态转移图,EDA 软件就能生成相应的 HDL 代码或者原理图,使用十分方便。但是需要指出的是,波形输入和状态机输入方法只能在某些特殊情况下缓解设计者的工作量,并不适合所有的设计。

2. 功能仿真

电路设计完成后,要用专用的仿真工具对设计进行功能仿真,验证电路功能是否符合设计要求。功能仿真有时也被称为前仿真,通过仿真能及时发现设计中的错误,加快设计进度,提高设计的可靠性。

3. 综合优化

综合优化(Synthesize)是指将硬件描述语言、原理图等设计输入翻译成由与、或、非门, RAM, 触发器等基本逻辑单元构成的逻辑连接(网表),并根据目标与要求(约束条件)优化所生成的逻辑连接,输出 edf 和 edn 等标准格式的网表文件,供 FPGA/CPLD 厂家的布局布线器件进行实现。通过设置参数制定优化准则,优化目标主要有面积和速度两个方面。一般根据设计需要选择面积或者速度优化。

4. 综合后仿真

综合完成后需要检查综合结果是否与原设计一致,叫做综合后仿真,可估计门延时带来的影响。但是只能估计门延时,不能估计线延时,仿真结果与布线后的实际情况还有一定的差距,并不十分准确。这种仿真的主要目的在于检查综合器的综合结果是否与设计输入一致。

5. 实现与布局布线

综合的本质是生成一些由与、或、非门, 触发器, RAM 等基本逻辑单元组成的逻辑网表,它与芯片实际的配置情况还有较大差距。此时应该使用 FPGA/CPLD 厂商提供的软件工具,根据所选芯片的型号,将综合输出的逻辑网表适配到具体 FPGA/CPLD 器件上,这个过程就叫做实现。因为只有器件开发商最了解器件的内部结构,所以实现步骤必须选用器件开发商提供的工具。在实现过程中最主要的是布局布线(PAR, Place And Route);所谓布局(Place)是指将逻辑网表中的硬件原语或者底层单元合理地适配到 FPGA 内部的固有硬件结构上,布局的优劣对设计的最终实现结果(在速度和面积两个方面)影响很大;所谓布线(Route)是指根据布局的拓扑结构,利用 FPGA 内部的各种连线资源,正确连接各个元件的过程。FPGA 的结构相对复杂,为了获得更好的实现结果,一般采用时序驱动的引擎进行布局布线,所以对于不同的设计输入,特别是不同的时序约束,获得的布局布线结果一般有较大差异。CPLD 结构相对简单得多,其资源有限而且布线资源一般为交叉连接矩阵,故 CPLD 的布局布线过程相对简单得多,一般被称为适配过程。

6. 时序仿真

将布局布线的延时信息反标注到设计网表中,所进行的仿真就叫做时序仿真。布局布线之后生成的仿真延时文件包含的延时信息最全,不仅包含门延时,还包含实际布线延时,所以布线后仿真最准确,能较好地反映芯片的实际工作情况。一般来说,布线后必须进行仿真,通过布局布线后仿真能检查设计时序与 FPGA 实际运行情况是否一致,确保设计的可靠性和稳定性,发现时序违规情况。

7. 板极仿真与验证

在有些高速设计情况下还需要用第三方的板极验证工具进行仿真与验证,分析高速设计的信号完整性、电磁干扰等电路特性。

8. 加载配置

如果编译、综合、布线/适配和行为仿真、功能仿真、时序仿真等过程都没有发现问题,即满足原设计的要求,则可以将由 FPGA/CPLD 布线/适配器产生的配置/下载文件通过编程器或下载电缆载入目标芯片 FPGA 或 CPLD 中。

9. 调试

最后步骤就是在线调试或者将生成的配置文件写入芯片中进行调试。示波器和逻辑分析仪(LA, Logic Analyzer)是主要调试工具。要求 FPGA 和 PCB 设计人员保留一定数量 FPGA 管脚作为测试管脚,在综合实现时再把这些输出信号锁定到测试管脚上,然后连接逻辑分析仪的探头进行观测。PCB 布线后测试管脚的数量就固定了,不能灵活增加,当测试脚不够用时会影响测试,如果测试脚太多又影响 PCB 布局布线。

对于简单的设计,使用 Quartus II 内嵌的 SignalTap II 进行在线逻辑分析可以较好地解决上述矛盾。SignalTap II 是一种 FPGA 在线片内信号分析工具,它的主要功能是在线、实时地读出 FPGA 的内部信号。

任何仿真或验证步骤出现了问题,就需要根据错误的定位返回到相应的步骤更改或者重新设计。

第 2 章 FPGA/CPLD 结构与应用

2.1 可编程逻辑器件基础

2.1.1 逻辑电路符号表示方法

由于 PLD 的特殊结构,用通用的逻辑门符号表示比较繁杂,因此一般用一种约定的符号来简化表示。接入 PLD 内部的与或阵列输入缓冲器电路,一般采用互补结构,图 2-1(a)、(b)表示当信号输入 PLD 后,分别以其同相信号和反相信号输出。

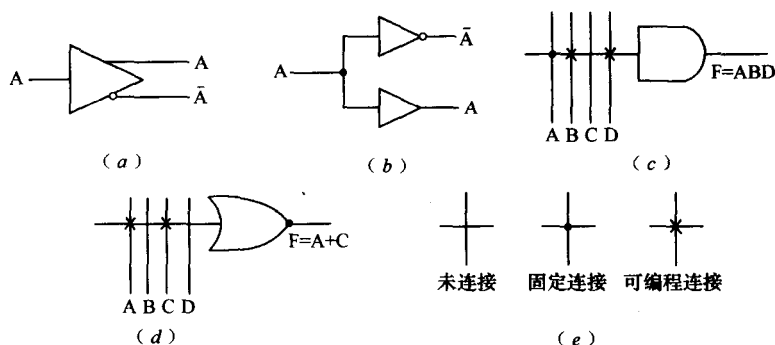


图 2-1 PLD 中各种门的简化图形表示

图 2-1(c)是 PLD 中与阵列的简化图形,表示可以选择 A、B、C、D 四个信号中的任一或全部输入与门。图 2-1(d)是 PLD 中或阵列的简化图形。

图 2-1(e)是阵列中的连接关系。十字交叉线表示此二线连接;交叉线交点上打黑点,表示是固定连接,即在 PLD 出厂时已连接;交叉线的交点上打叉,表示该点可编程,在 PLD 出厂后通过编程,其连接可随时改变。

2.1.2 可编程逻辑器件的分类

1. 按照集成度分类

芯片集成度较低,可用的逻辑门数量大约在 500 门以下,称为简单 PLD。早期的 PROM、PLA、PAL、GAL 等均属此类。

芯片集成度较高的为复杂 PLD,如现在大量使用的 CPLD、FPGA。

2. 按照编程结构分类

乘积项结构:其基本结构为“与-或阵列”,大部分简单 PLD 和 CPLD 都属于此类。

查找表结构:有简单的查找表组成可编程门,然后构成阵列,FPGA 属于此类。

3. 按照可编程性分类

熔断器型:这些熔断器就像平常的家用保险丝一样由一根特殊的导线构成。当有大