

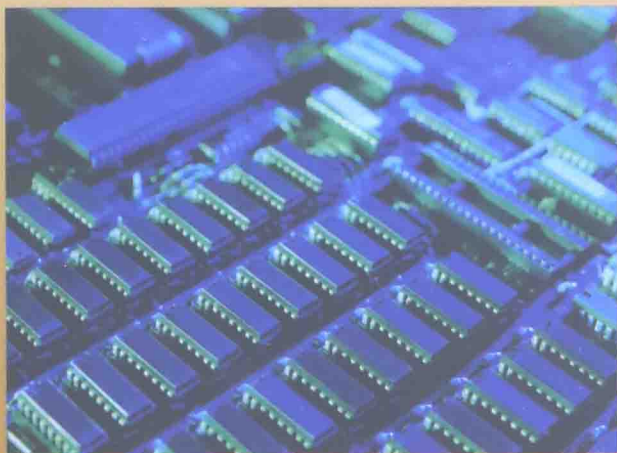


普通高等教育“十一五”国家级规划教材

可编程逻辑器件 及EDA技术

数字系统设计与SOPC技术

李景华 杜玉远 主编



東北大學出版社
Northeastern University Press



普通高等教育“十一五”国家级规划教材

可编程逻辑器件及 EDA 技术

——数字系统设计与 SOPC 技术

李景华 杜玉远 主编

TP332.1-43
07
2014

东北大学出版社

• 沈 阳 •

© 李景华 杜玉远 2014

图书在版编目 (CIP) 数据

可编程逻辑器件及 EDA 技术: 数字系统设计与 SOPC 技术 / 李景华, 杜玉远主编. —沈阳: 东北大学出版社, 2014.7

ISBN 978-7-5517-0708-4

I. ①可… II. ①李… ②杜… III. ①可编程逻辑器件—高等学校—教材 ②电子电路—电路设计—计算机辅助设计—高等学校—教材 IV. ①TP332.1 ②TN702

中国版本图书馆 CIP 数据核字 (2014) 第 160451 号

出 版 者: 东北大学出版社

地址: 沈阳市和平区文化路 3 号巷 11 号

邮编: 110004

电话: 024—83687331 (市场部) 83680267 (社务室)

传真: 024—83680180 (市场部) 83680265 (社务室)

E-mail: neuph @ neupress.com

http: // www. neupress. com

印 刷 者: 廊坊市新阳印刷有限公司

发 行 者: 东北大学出版社

幅面尺寸: 184mm × 228mm

印 张: 31.75

字 数: 693 千字

出版时间: 2014 年 9 月第 1 版

印刷时间: 2014 年 9 月第 1 次印刷

责任编辑: 王兆元

封面设计: 唯 美

责任校对: 闻 悦

责任出版: 唐敏志

ISBN 978-7-5517-0708-4

定 价: 88.00 元

前 言

随着电子技术的飞速发展,可编程逻辑器件及其设计技术正在快速进步,而可编程逻辑器件的应用数字系统设计技术也在不断进步和更新。

20 世纪末期,集成电路的制造技术处于深亚微米阶段,其特征尺寸为 $0.35\sim 0.18\mu\text{m}$ 。在经历短短几年之后,集成电路的制造技术已经发展到 $90\sim 65\text{nm}$ 的水平。这就意味着单片可编程逻辑器件可以集成几千万个 PLD 门。可编程逻辑器件和 ASIC 技术的融合,使系统在片 SOPC (System On Programmable Chip) 技术得以实现。所谓 SOPC 技术,就是将各种 MCU、存储器、各种接口单元以硬核或软核方式集成到可编程逻辑器件中。SOPC 作为数字系统设计的一个全新应用领域,需要融入最新的教学内容。因此,我们对本书第一版内容进行了必要的增减。

基于上述考虑,本书保留了第一版中可编程逻辑器件基础、数字系统设计、VHDL、典型数字系统设计方法及其实例等内容。

作者在总结 SOPC 技术实践的经验和体会的基础上,新增了 SOPC 技术的应用器件结构与工作原理、SOPC 的硬件设计和软件设计及其 IP 核应用技术等内容。特别详细讲解了 Quartus II 7.0 的使用方法, Nios II 系统的软件、硬件设计过程和设计实例。

此外,为了使读者更为充分地了解学习新兴的 EDA 设计工具和设计方法,在本书的最后一章还系统地介绍了 Altium 公司推出的一体化 EDA 设计工具 Altium Designer V6.5,并且以翔实的例子介绍了基于 NanoBoard-NB1 开发板的 EDA 设计实现方法。

本书可作为高等院校电子信息工程、计算机应用、通信工程、微电子、自动控制类专业的本科生和研究生学习数字系统设计课的教材,也可以作为电子系统设计工程师的技术参考书。

本书由李景华、杜玉远、李汇明编写。其中第 1、2、3、4 章由李汇明和李景华编写，第 5、6、7、8、9、10 章由杜玉远编写。全书由李景华和杜玉远负责内容编排、定稿与修改。本书在编写过程中参考了相关专家和学者的著作，同时也引用了一些公司的相关数据手册，在此深表谢意。

由于可编程逻辑器件和 EDA 技术发展迅速，加之编者水平有限，书中可能存在不足或错误，请读者批评指正。

编 者

2014 年 4 月

目 录

第 1 章 可编程器件和 EDA 技术概述	1
1.1 EDA 技术的主要特征	1
1.2 EDA 技术的设计方法	3
1.3 可编程逻辑器件简介	4
1.3.1 从 ASIC 到 FPGA/CPLD	4
1.3.2 CPLD 器件	7
1.3.3 FPGA 器件	13
1.4 可编程逻辑器件设计	28
1.4.1 可编程逻辑器件的设计流程	28
1.4.2 Xilinx 公司的 ISE 开发工具概述	29
1.4.3 Altera 公司的 Quartus II 开发工具概述	31
1.5 可编程逻辑器件选型	31
1.5.1 CPLD 选择的方法	31
1.5.2 FPGA 选择的方法	33
1.6 IP 核简介	34
1.7 EDA 技术的发展趋势	36
1.7.1 可编程逻辑器件的发展趋势	36
1.7.2 EAD 软件开发工具的发展趋势	36
1.7.3 设计输入方式的发展趋势	37
第 2 章 VHDL 硬件描述语言	39
2.1 HDL 简介	39
2.1.1 代表性的 HDL 语言	39
2.1.2 VHDL 程序结构	40
2.1.3 程序包	40
2.1.4 库	44
2.1.5 实体和结构体	45
2.1.6 配 置	49
2.2 VHDL 基本要素	50
2.2.1 标识符	50

2.2.2	数据对象	51
2.2.3	数据类型	53
2.2.4	用户自定义的数据类型	55
2.2.5	数据类型的转换	56
2.2.6	操作符	57
2.2.7	函数类属性	60
2.3	VHDL 的主要语句及应用	64
2.3.1	进 程	64
2.3.2	过程及其函数	65
2.3.3	顺序描述语句	68
2.3.4	信号赋值语句	73
2.3.5	COMPONENT 语句和 COMPONENT INSTANT 语句	75
2.3.6	GENERIC 语句和 GENERATE 语句	76
第 3 章	典型 VHDL 设计实例	79
3.1	组合逻辑电路设计	79
3.1.1	逻辑门电路设计	79
3.1.2	常用编码器设计	81
3.1.3	常用译码器设计	85
3.1.4	数据选择器设计	87
3.1.5	数据分配器设计	89
3.1.6	数值比较器设计	91
3.1.7	算术运算单元电路设计	93
3.2	时序逻辑电路设计	95
3.2.1	常用触发器设计	96
3.2.2	常用数码寄存器设计	98
3.2.3	常用计数器设计	102
3.3	有限状态机设计	105
3.3.1	有限状态机的建模	106
3.3.2	状态编码	109
3.3.3	Mealy 型状态机设计	111
3.3.4	Moore 型状态机设计	115
3.4	存储器设计	119
3.4.1	只读存储器(ROM)的设计	119
3.4.2	随机存储器(RAM)的设计	121
3.4.3	顺序存取存储器的设计	123
第 4 章	典型数字系统的设计	125
4.1	数字系统概述	125
4.2	数码管动态显示扫描电路原理及设计	125

4.2.1	数码管动态显示扫描电路原理	125
4.2.2	采用 VHDL 描述的动态显示扫描电路	126
4.3	乘法器的原理及设计	131
4.3.1	乘法器工作原理	131
4.3.2	采用 VHDL 描述的乘法器	132
4.4	除法器的原理及设计方法	135
4.4.1	除法器的工作原理	135
4.4.2	用 VHDL 描述的除法器	138
4.5	简易 CPU 工作原理及设计方法	141
4.5.1	简易 CPU 的工作原理	141
4.5.2	采用 VHDL 描述的 ALU	144
4.6	交通信号灯控制器原理及设计	148
4.6.1	交通信号灯控制器原理	148
4.6.2	交通信号灯的 VHDL 描述	151
4.7	数字频率计的原理及设计	156
4.7.1	数字频率计的原理	156
4.7.2	数字频率计的 VHDL 描述	159
4.8	数字信号发生器的原理及设计	164
4.8.1	数字信号发生器(DDS)的原理	164
4.8.2	数字信号发生器(DDS)的 VHDL 描述	166
第 5 章	Quartus II 7.0 开发系统	173
5.1	Quartus II 7.0 开发系统简介	173
5.1.1	Quartus II 7.0 开发系统的特性	173
5.1.2	Quartus II 7.0 开发系统的安装	174
5.1.3	Quartus II 7.0 开发系统的软件许可配置	178
5.1.4	Quartus II 7.0 开发系统的设计流程	179
5.2	设计输入	180
5.2.1	建立设计工程	181
5.2.2	原理图设计文件	184
5.2.3	VHDL 设计文件	189
5.2.4	设计约束文件	190
5.3	综合与编程	192
5.3.1	综合参数控制	193
5.3.2	RTL 查看器和状态机查看器	194
5.3.3	渐进式综合	196
5.3.4	多样化编程	203
5.4	设计仿真	205
5.4.1	仿真波形文件	205

5.4.2	仿 真	208
5.5	SignalTap II 逻辑分析器	209
5.5.1	设置和运行 SignalTap II 逻辑分析器	209
5.5.2	渐进式编译使用 SignalTap II 逻辑分析器	212
5.5.3	分析 SignalTap II 数据	213
5.6	设计实例	213
5.6.1	建立设计工程	213
5.6.2	建立源文件	214
5.6.3	编译设计	217
5.6.4	引脚锁定	218
5.6.5	仿真设计	221
5.6.6	编程和配置	224
第 6 章	SOPC 系统简介	225
6.1	概 述	225
6.1.1	SOC 简介	225
6.1.2	SOPC 技术	226
6.2	典型的 SOPC 系统处理器	226
6.2.1	Altera 公司的 Nios II 软核处理器	226
6.2.2	Xilinx 公司的 PowerPC 硬核处理器	228
6.2.3	Xilinx 公司的 MicroBlaze 软核处理器	229
6.2.4	Lattice 公司的 LatticeMico 32 软核处理器	231
6.3	典型的 SOPC 系统开发工具	233
6.3.1	Altera 公司的 SOPC 开发工具	233
6.3.2	Xilinx 公司的 SOPC 开发工具	237
6.3.3	Lattice 公司的 SOPC 开发工具	238
6.4	支持 Nios II 系统的 FPGA 器件	240
6.4.1	Cyclone 系列 FPGA 器件	240
6.4.2	Cyclone II 系列 FPGA 器件	246
6.4.3	Cyclone III 系列 FPGA 器件	253
6.4.4	Stratix II 系列 FPGA 器件	254
6.4.5	Stratix II GX 系列 FPGA 器件	261
6.5	支持 MicroBlaze 软核和 PowerPC 硬核的 FPGA 器件	264
6.5.1	Spartan-3 系列 FPGA 概述	264
6.5.2	Spartan-3 系列 FPGA 结构特性	265
6.5.3	Spartan-3 系列 FPGA 的 IOB 结构特性	266
6.5.4	Spartan-3 系列 FPGA 的 CLB 结构特性	266
6.5.5	Spartan-3 系列 FPGA 的 RAM 结构特性	268
6.5.6	Spartan-3 系列 FPGA 的时钟网络特性	269
6.5.7	Spartan-3 系列 FPGA 的布线资源特性	271

6.6 支持 LatticeMico32 软核处理器的 FPGA 器件	272
6.6.1 LatticeXP 系列 FPGA 概述	272
6.6.2 LatticeXP 系列的 PFU 和 PFF 结构特性	273
6.6.3 LatticeXP 系列的布线资源结构特性	275
6.6.4 LatticeXP 系列的 PLL 结构特性	276
6.6.5 LatticeXP 系列的嵌入式 RAM 块结构特性	277
6.6.6 LatticeXP 系列的 PIC 结构特性	278
第 7 章 Nios II 嵌入式处理器及总线接口	279
7.1 Nios II 嵌入式处理器	279
7.1.1 Nios II 系统概述	279
7.1.2 Nios II 嵌入式处理器结构	280
7.1.3 Nios II 嵌入式处理器 ALU	281
7.1.4 Nios II 嵌入式处理器复位	282
7.1.5 Nios II 嵌入式处理器异常和中断	283
7.1.6 Nios II 嵌入式处理器存储器和 I/O 组织	283
7.1.7 Nios II 嵌入式处理器 JTAG 调试模块	287
7.2 Nios II 嵌入式处理器编程结构	289
7.2.1 Nios II 嵌入式处理器通用寄存器	289
7.2.2 Nios II 嵌入式处理器控制寄存器	289
7.2.3 Nios II 嵌入式处理器工作模式	291
7.2.4 Nios II 嵌入式处理器异常处理	291
7.2.5 Nios II 嵌入式处理器异常原因确定	293
7.2.6 Nios II 嵌入式处理器异常返回	293
7.2.7 Nios II 嵌入式处理器中断处理	294
7.2.8 Nios II 嵌入式处理器的存储器和外设访问	294
7.2.9 Nios II 嵌入式处理器的复位	295
7.2.10 Nios II 嵌入式处理器的指令分类	296
7.3 Avalon 交换式总线	300
7.3.1 Avalon 总线基本术语	301
7.3.2 Avalon 总线传输	304
7.3.3 Avalon 总线从传输	305
7.3.4 Avalon 总线主传输	314
7.4 Avalon 总线的片外设备接口	319
7.4.1 从传输的 Avalon 三态信号	319
7.4.2 无延迟的 Avalon 三态从端口读传输	321
7.4.3 带固定延迟的 Avalon 三态从端口读传输	321
7.4.4 Avalon 三态从端口写传输	323
7.5 Avalon 总线地址对齐方式	325
7.5.1 地址对齐概述	325

7.5.2	地址对齐参数选择	325
7.5.3	动态总线宽度	326
第 8 章	Nios II 系统嵌入式外设	327
8.1	PIO 核	327
8.1.1	功能描述	327
8.1.2	SOPC Builder 中配置 PIO 核	329
8.1.3	PIO 核的编程模型	330
8.2	定时器核	338
8.2.1	功能描述	338
8.2.2	SOPC Builder 中配置定时器	339
8.2.3	定时器核的编程模型	341
8.3	PLL 核	344
8.3.1	功能描述	344
8.3.2	SOPC Builder 中配置 PLL 核	345
8.4	性能计数器核	347
8.4.1	功能描述	347
8.4.2	SOPC Builder 中配置性能计数器	348
8.4.3	性能计数器的编程模型	349
8.5	System ID 核	350
8.5.1	功能描述	350
8.5.2	SOPC Builder 中配置 System ID 核	351
8.5.3	System ID 核的编程模型	351
8.6	SDRAM 控制器核	351
8.6.1	功能描述	352
8.6.2	SOPC Builder 中配置 SDRAM 核	353
8.6.3	SDRAM 控制核的编程模型	355
8.6.4	SDRAM 应用	355
8.7	CFI 控制器核	356
8.7.1	功能描述	356
8.7.2	SOPC Builder 中配置 CFI 控制器核	357
8.7.3	CFI 控制器的编程模型	358
8.8	EPCS 控制器核	359
8.8.1	功能描述	359
8.8.2	SOPC Builder 中配置 EPCS 控制器核	360
8.8.3	EPCS 控制器的编程模型	361
8.9	FIFO 存储器核	362
8.9.1	功能描述	362
8.9.2	SOPC Builder 中配置 FIFO 存储器核	364

8.9.3	FIFO 存储器核的编程模型	367
8.10	SPI 核	369
8.10.1	功能描述	370
8.10.2	SOPC Builder 中配置 SPI 核	372
8.10.3	SPI 核的编程模型	374
8.11	UART 核	376
8.11.1	功能描述	376
8.11.2	SOPC Builder 中配置 UART 核	378
8.11.3	UART 核的编程模型	381
8.12	JTAG UART 核	384
8.12.1	功能描述	385
8.12.2	SOPC Builder 中配置 JTAG UART 核	386
8.12.3	JTAG UART 核的编程模型	388
8.13	DMA 核	390
8.13.1	功能描述	390
8.13.2	SOPC Builder 中配置 DMA 核	392
8.13.3	DMA 核的编程模型	393
第 9 章	Nios II 系统设计	396
9.1	Nios II 系统硬件设计	396
9.1.1	Nios II 系统硬件开发流程	396
9.1.2	Nios II 系统需求分析	397
9.1.3	Nios II 系统工程建立	397
9.1.4	Nios II 系统集成	399
9.1.5	Nios II 系统设计	409
9.1.6	Nios II 系统综合	411
9.1.7	Nios II 系统实现	412
9.2	Nios II 软件设计	415
9.2.1	Nios II 集成开发环境	415
9.2.2	Nios II 系统软件设计	416
9.2.3	Nios II 系统软件调试	418
9.3	Nios II 系统的引导	420
9.3.1	Nios II 系统的引导概述	420
9.3.2	Nios II 系统的引导	421
9.3.3	CFI Flash 引导程序	421
9.3.4	EPCS 引导程序	421
9.4	Flash 编程工具	422
9.5	Altera DE2 评估板	424
9.5.1	Altera DE2 评估板硬件资源	424

9.5.2	Altera DE2 评估板引脚的定义	426
9.5.3	Altera DE2 编程方法	437
9.6	基本 I/O 接口设计实例	438
9.6.1	设计需求分析	438
9.6.2	硬件设计	439
9.6.3	软件设计	442
9.7	定时器应用设计实例	444
9.7.1	设计需求分析	444
9.7.2	硬件设计	444
9.7.3	软件设计	448
第 10 章	一体化 EDA 开发工具	452
10.1	Altium Designer 6.X 简介	452
10.2	NanoBoard-NB1 性能简介	453
10.2.1	NanoBoard-NB1 基本配置	453
10.2.2	NanoBoard-NB1 安装	454
10.2.3	NanoBoard-NB1 主要接口资源	455
10.3	PCB 设计实例	459
10.3.1	建立 PCB 工程	459
10.3.2	原理图设计	460
10.3.3	PCB 设计	464
10.4	FPGA 设计实例	465
10.4.1	建立工程	465
10.4.2	建立设计原理图	466
10.4.3	导入设计约束	466
10.4.4	编译设计	473
10.4.5	在线调试	473
10.5	嵌入式系统设计实例	478
10.5.1	建立 FPGA 工程	478
10.5.2	建立嵌入式工程	480
10.5.3	编辑嵌入式软件	480
10.5.4	连接 FPGA 工程和嵌入式工程	482
10.5.5	调试嵌入式系统	482
10.6	简易计算器的设计与实现	483
10.6.1	设计需求分析	483
10.6.2	EPGA 工程的设计与实现	484
10.6.3	嵌入式软件的设计与实现	485
参考文献		495

第1章 可编程器件和 EDA 技术概述

EDA(Electronic Design Automation, 电子设计自动化)技术是以计算机为工作平台, 以融合了应用电子技术、计算机技术、智能化技术最新成果而研制成的电子 CAD 通用软件包为开发环境, 以电子系统设计为应用方向的电子产品自动化设计过程。它主要包含三方面的设计工作, 即 IC 设计、电子电路设计和 PCB 设计。本章主要介绍 EDA 技术的主要特征和发展方向, 常用的复杂可编程逻辑器件 CPLD 和现场可编程门阵列 FPGA 的工作原理, 并对常用的 EDA 工具和设计技术进行简单介绍。

1.1 EDA 技术的主要特征

20 世纪 90 年代以后, 电子系统已经从电路板级系统集成发展成为包括 ASIC、FPGA 和嵌入系统的多种模式, EDA 产业已经成为电子信息类产品的支柱产业。过去几十年内, IC 设计方法经历了从手工设计(Hand Design)、电路仿真(Circuit Simulation)、原理图输入(Schematic Capture)和逻辑仿真(Logic Simulation)、布局(Placement)和布线(Routing)到综合(Synthesis)几个阶段。近年来, 微电子技术以惊人的速度发展, 其工艺水平已达到纳米级。在一个芯片上可集成数百万乃至上千万个晶体管, 工作速度可达到 Gb/s 的数量级, 这为制造出规模更大、速度和信息容量更高的芯片系统提供了基础条件。集成电路设计技术的进步也对 EDA 技术提出了更高的要求, 大大地促进了 EDA 技术的发展。

以高级语言描述、系统仿真和综合技术为特征的 EDA 技术, 代表了当今电子设计技术的最新发展方向。EDA 设计技术的基本流程是设计者按照“自上而下”的设计方法, 对整个系统进行方案设计和功能划分。电子系统的关键电路一般用一片或几片专用集成电路(ASIC)实现, 采用硬件描述语言(HDL)完成系统行为级设计, 最后通过综合器和适配器生成最终的目标器件。这种被称为高层次的电子设计方法, 不仅极大地提高了系统的设计效率, 而且使设计者摆脱了大量的辅助性工作, 将精力集中于创造性的方案与概念的构思上。近年来的 EDA 技术主要有以下特点:

- (1) 采用行为级综合工具, 设计层次由 RTL 级上升到了系统级;
- (2) 采用硬件描述语言描述大规模系统, 使数字系统的描述进入抽象层次;
- (3) 采用 Floor Planning 技术, 使得复杂 IC 的描述规范化, 做到在逻辑综合早期设计阶段就考虑到物理设计的影响;
- (4) 将可测性电路结构做在可编程 ASIC 芯片上, 开发了扫描插入、内建自测试(Built-

in Self-Test)、边界扫描(JTAG)等可测性设计工具,并已集成到 EDA 系统中;

(5) 为带有嵌入式 CPU 核的 ASIC 设计提供软、硬件协同设计工具。

针对当今 ASIC 的特点(规模大而复杂、数字与模拟电路并存、硬件与软件设计并存产品上市速度快),建立并行设计工程框架结构的集成化、系统化设计环境,将不同公司的优秀工具集成为一个完整的 EDA 系统,各种 EDA 工具在该框架中可以并行使用。

下面介绍与 EDA 技术密切相关的几个概念。

1. “自上而下”的设计方法

电子设计的最初思路是选用标准集成电路“自底向上”地构造出一个新的系统,这样的设计方法不仅效率低、成本高,而且容易出错。

现代电子系统的设计理念是一种“自上而下”的设计方法。首先从系统设计入手,在顶层对系统进行功能的划分和结构的设计。这一级采用硬件描述语言对高层次的系统进行所谓行为级描述,在系统一级进行验证。然后,用综合优化工具生成具体门电路的网络表,其对应的物理实现通常采用印刷电路板或可编程专用集成电路。由于设计的主要仿真和调试过程是在高层次上完成的,因此有利于早期发现结构设计上的错误,避免设计工作的重复,同时减少了逻辑功能仿真的工作量,提高了设计的成功率。

2. ASIC

ASIC 是专用集成电路(Application Specific Integrated Circuit)的简称。它是随着电子产品的复杂度日益提高应运而生的。一个电子系统若由数万个中小规模的 IC 构成,会产生体积大、功耗大、可靠性差等诸多问题。解决这一问题的有效方法就是采用 ASIC 芯片进行设计。ASIC 按照设计方法的不同,可分为全定制 ASIC、半定制 ASIC 和可编程 ASIC。

全定制 ASIC 芯片是由设计者定义芯片上所有晶体管的几何图形和工艺规则,最后将设计结果交由 IC 厂家去进行掩模制造,做出产品。这种设计方法的优点是芯片可以获得最优的性能,即面积利用率高、速度快、功耗低;缺点是开发周期长,费用高,只适合大批量产品开发。目前模拟电路的设计通常采用该方法。

半定制 ASIC 芯片的版图设计方法分为门阵列设计法和标准单元设计法。这两种方法都是约束性的设计方法,其主要目的就是简化设计,以牺牲芯片性能为代价来缩短开发时间。该方法目前在通信领域仍有较大的应用空间。

可编程 ASIC,又称可编程逻辑器件。自 20 世纪 70 年代以来,经历了 PAL, GAL, CPLD, FPGA 几个发展阶段,其中 CPLD/FPGA 属高密度可编程逻辑器件,目前集成度已高达 200 万门/片~1000 万门/片,它将掩模 ASIC 集成度高的优点和可编程逻辑器件设计生产方便的特点结合在一起,特别适合于样品研制或小批量产品开发,使产品能以最快的速度上市,而当市场扩大时,它可以很容易地转由掩模 ASIC 实现,因此开发风险也大为降低,已成为现代高层次电子设计方法的实现载体。

3. 硬件描述语言

— 硬件描述语言 (Hardware Description Language, HDL) 用软件编程的方式来描述电子系统的逻辑功能、电路结构和连接形式, 与传统的门级描述方式相比, 它更适合大规模电子系统的设计。早期的硬件描述语言, 如 ABEL, HDL, AHDL, 由不同的 EDA 厂商开发, 互不兼容, 而且不支持多层次设计, 层次间翻译工作要由人工完成。为了克服以上不足, 1985 年美国国防部正式推出了高速集成电路硬件描述语言 VHDL, 1987 年 IEEE 采纳 VHDL 为硬件描述语言标准, 命名为 IEEE STD 1076—1987, 通常称为 87 版; 1993 年进一步修订后, 形成了更加完备的 IEEE STD 1076—1993 版本, 简称为 93 版; 最新的 VHDL 标准是 IEEE STD 1076—2001 版本。目前主流的设计工具可以很好地支持这些版本的 VHDL 设计。

4. EDA 系统框架结构

EDA 系统框架结构 (Framework) 是一套配置和使用 EDA 软件包的规范。目前主要的 EDA 系统都建立了框架结构, 如 Cadence 公司的 Design Framework, Mentor 公司的 Falcon Framework, 而且这些框架结构都遵守国际 CFI 组织制定的统一技术标准。框架结构能来自不同 EDA 厂商的工具软件进行优化组合, 集成在一个易于管理的统一的环境之下, 而且还支持任务之间、设计师之间以及整个产品开发过程中的信息传输与共享, 是并行工程和自上而下设计方法的实现基础。

1.2 EDA 技术的设计方法

EDA 技术的每一次进步, 都引起了设计层次上的飞跃。物理级设计主要指 IC 版图设计, 一般由半导体厂家完成, 对电子工程师没有太大的意义。电子工程师所关注的内容主要是电路级设计和系统级设计。

1. 电路级设计

电路级设计本质上是基于门级描述的单层次设计。电子工程师接受系统设计任务后, 首先确定设计方案, 并选择能实现该方案的合适元器件, 然后根据具体的元器件设计电路原理图。接着要进行第一次仿真, 其中包括数字电路的逻辑模拟、故障分析, 模拟电路的交直流分析、瞬态分析。在进行系统仿真时, 必须要有元件模型库的支持, 计算机上模拟的输入-输出波形代替了实际电路调试中的信号源和示波器。这一层次仿真主要是检验设计方案在功能方面的正确性。

仿真通过后, 根据原理图产生的电气连接网络表再进行 PCB 板的自动布局和布线。在制作 PCB 板之前还可以进行 PCB 后分析, 其中包括热分析、噪声及窜扰分析、电磁兼容分析、可靠性分析等, 并可将分析后的结果参数反标回电路图, 进行第二次仿真, 也称为后仿真。后仿真主要是检验 PCB 板在实际工作环境中的可行性。

电路级设计的所有工作 (包括设计输入、仿真和分析、设计修改等) 都是在基本逻辑门这一层次上进行的, 需要电子工程师全面地了解系统的功能特性和物理特性, 从而将开发

风险消灭在设计阶段。

2. 系统级设计

20 世纪 90 年代以来,电子信息类产品的开发明显呈现两个特点:一是产品复杂程度提高;二是产品上市时限紧迫。为了适应这种市场的需求,一种高层次的电子设计方法——系统级设计方法——应运而生。

高层次设计针对设计目标进行功能描述。设计者无需关心电路细节,而是将精力放在设计方案的制定和创新上,以高层次描述的形式输入计算机。高层次设计只是定义系统的行为特性,不涉及实现工艺,还可以在厂家综合库的支持下,利用综合优化工具,将高层次描述转换为针对某种工艺优化的网络表,可轻易实现工艺优化和系统升级。

系统级设计的流程如下。

- 系统工程师按照“自上而下”的方法进行系统划分。
- 采用 VHDL 语言状态图等方式描述系统,并编译成标准的 VHDL 文件。
- 进行代码级的功能仿真,检验系统功能设计的正确性。
- 对 VHDL 源代码进行综合优化处理,生成门级描述的网络表文件,这是将高层次描述转化为硬件电路的关键步骤。综合的过程要在相应的厂家综合库支持下才能完成。
- 利用产生的网络表文件进行适配前的时序仿真。
- 将综合后的网络表文件针对某一具体的目标器件进行适配,包括底层器件配置、逻辑分割、逻辑优化、布局布线。
- 根据适配后的仿真模型,进行适配后的时序仿真,因为已经得到所描述系统的实际硬件特性(如时延特性),所以仿真结果能比较精确地预期实现所描述系统的未来芯片的实际性能。如果仿真结果达不到设计要求,就需要修改 VHDL 源代码或选择不同速度和品质的器件,直至满足设计要求。
- 将适配产生的器件编程文件通过编程器或下载电缆载入目标芯片 FPGA 或 CPLD 中。

1.3 可编程逻辑器件简介

1.3.1 从 ASIC 到 FPGA/CPLD

1. ASIC

专用集成电路是设计者根据设计需求所设计的在特殊场合使用的集成电路,是相对标准集成电路而言的,它提供了设计的安全性和系统设计的整合效率。

标准集成电路是指具有标准的芯片功能、可以在市场上购买到的通用器件。例如,以下器件都属于标准集成电路:

- (1) 中央处理单元(CPU);