清华大学计算机系列教材

数字系统设计自动化

薛宏熙 边计年 苏 明

清华大学出版社

数字系统设计自动化

薛宏熙　边计年　苏　明

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书是高等学校计算机、电子工程等有关专业高年级学生和研究生的教科书，也是这一领域工程技术人员的参考书。本书是作者多年教学和科研工作的总结，力求做到深入浅出而又不失严格性。它为 EDA 工具的开发者提供理论基础，也为 EDA 工具的使用者提供专业知识。

全书共分 8 章：第 1 章介绍电子数字系统设计自动化(EDA)的各个领域；第 2 章介绍硬件描述语言 VHDL；第 3 章介绍 EDA 系统的框架结构；第 4 章介绍模拟技术；第 5 章介绍逻辑综合；第 6 章介绍高层次综合；第 7 章介绍故障诊断；第 8 章介绍形式验证的理论和方法。

图书在版编目(CIP)数据

数字系统设计自动化/薛宏熙等编著. —北京：清华大学出版社，1996

ISBN 7-302-02331-X

Ⅰ. 数… Ⅱ. 薛… Ⅲ. 数字系统-自动设计 Ⅳ. TP271.2

中国版本图书馆 CIP 数据核字(96)第 19380 号

出版者：清华大学出版社（北京清华大学校内，邮编 100084）

印刷者：北京国马印刷厂

发行者：新华书店总店北京科技发行所

开　本：787×1092 1/16　印张：26.5　字数：627 千字

版　次：1996 年 10 月 第 1 版　1996 年 10 月 第 1 次印刷

书　号：ISBN 7-302-02331-X/TP・1154

印　数：0001—6000

定　价：25.00 元

前　言

本书是为高等学校计算机、电子工程等有关专业的高年级学生和研究生编写的教科书，着重介绍数字系统设计自动化各个领域的基础理论和最新发展。

数字系统，包括数字集成电路及其所构成的网络，其中也包括计算机本身，一直处于飞速发展之中，影响到各行各业。不仅影响高新技术的各个领域，而且进入汽车、家电等日常生活的各个领域。由于VLSI规模和技术复杂度的急剧增长，一个芯片内已可集成一个小型的计算机系统，达几十万甚至几百万门，并且还在迅速增长，人工设计已成为不可能，必须依靠自动设计(EDA)技术。由于市场竞争的激烈，要求缩短VLSI的设计周期和降低设计成本，这一需求成为EDA技术的驱动力。近年来，EDA技术是一门非常活跃的学科，并且不断有更好的、比以往价格更为低廉的商品化软件问世。本书正是针对这些而围绕以下两个目标而写的：

(1)为EDA工具的开发者提供理论基础。

(2)为EDA工具的使用者提供专业知识。尤其是电子设计工程师需要了解这方面的知识，不能单纯依靠纸、笔进行设计。

80年代末至90年代初，EDA技术在硬件描述语言的标准化方面取得进展，VHDL是其代表。进入90年代，高层次综合成为本领域的研究热点，有大量的研究文章和新的算法发表，也有一些商品化软件问世。从用户的需求方面来说，要求EDA软件支持"自顶向下"的设计方法和支持"设计重用"。本书将反映这些最新进展。

作为一本教材，需要兼顾讲清基本原理和反映学科前沿两个方面。本书是作者多年教学和科研工作的总结，力求做到深入浅出、信息量大；易读而又不失严格性。写作目的既面向在校学生，也面向这一领域的工程技术人员。

为了使正文简练而又便于查阅，把VHDL的标准程序包放在附录A和附录B中。此外，本书使用的名词术语大量来自英文，中文译法并不统一。我们根据自己的理解和常见的用法加以选用，并将中英文名词对照表列在书末的附录C中，供读者参考。

全书共分8章：第1章是综述，全面介绍数字系统设计自动化的各个领域，对学习后续各章起引导作用。第2章介绍硬件描述语言VHDL。第3章介绍EDA系统的框架结构。第4章介绍模拟技术，它是当前验证设计是否正确的主要手段。第5章介绍逻辑综合。第6章介绍高层次综合。第7章介绍故障诊断。第8章介绍形式验证的理论和方法。

本书的编写工作由清华大学计算机系的下列同志承担：薛宏熙教授负责第1章、第2章和第5章；边计年副教授负责第4章、第7章和第8章；苏明博士负责第3章和第6章。由于作者水平所限，缺点和错误在所难免，恳请读者批评指正。

编　者

1995年10月

目　录

第 1 章　概论 …… 1
1.1　电子系统设计自动化技术发展的回顾 …… 1
1.2　从 EDA 的角度来观察 VLSI …… 2
1.2.1　VLSI 的分类 …… 2
1.2.2　芯片布图 …… 4
1.2.3　可编程逻辑器件 …… 6
1.3　数字系统自动设计的流程 …… 10
1.4　EDA 的主要领域 …… 12
1.4.1　硬件描述语言 …… 12
1.4.2　模拟验证 …… 15
1.4.3　综合技术 …… 17
1.4.4　测试诊断 …… 18
1.4.5　逻辑设计形式验证 …… 18
1.4.6　工程实现 …… 19
1.5　EDA 系统的构成 …… 20
参考文献 …… 21
第 2 章　硬件描述语言 VHDL …… 22
2.1　硬件描述语言 VHDL …… 22
2.2　VHDL 的基础知识 …… 23
2.2.1　设计实体和结构体的概念 …… 23
2.2.2　面向模拟器的某些特性 …… 25
2.2.3　结构和行为 …… 26
2.2.4　数据类型与对象 …… 29
2.2.5　各分立部分之间的联结 …… 32
2.2.6　VHDL 的主要构件 …… 33
2.2.7　设计库 …… 39
2.3　VHDL 的数据类型 …… 41
2.3.1　文字 …… 41
2.3.2　标量类型 …… 42
2.3.3　复合类型 …… 43
2.3.4　子类型 …… 47
2.3.5　属性 …… 47
2.3.6　预定义算符 …… 49

2.4 VHDL 行为描述 …… 52
2.4.1 进程语句 …… 52
2.4.2 行为模型的顺序性 …… 53
2.4.3 行为模型的并行(并发)性 …… 62
2.5 VHDL 的结构描述 …… 71
2.5.1 结构描述的基本特征 …… 71
2.5.2 规则结构 …… 79
2.5.3 配置指定 …… 82
2.5.4 默认值与无连接端口 …… 84
2.6 VHDL 对大规模设计的支持 …… 86
2.6.1 设计库的概念 …… 86
2.6.2 VHDL 中名字的可见性 …… 86
2.6.3 使用 library 语句访问其它库 …… 89
2.6.4 块语句 …… 90
2.6.5 设计中的数据共享 …… 90
2.6.6 结构描述和行为描述的混合使用 …… 93
2.7 VHDL 的一些高级特性 …… 94
2.7.1 重载 …… 94
2.7.2 用户定义的属性 …… 96
2.7.3 与信号相关的属性 …… 97
2.7.4 被保护的信号赋值语句 …… 100
2.7.5 断开指定 …… 101
2.7.6 空事项处理 …… 103
2.8 设计实例 …… 105
2.8.1 交通灯控制器 …… 105
2.8.2 创建技术说明书 …… 106
参考文献 …… 113
第 3 章 EDA 系统的框架结构 …… 115
3.1 概述 …… 115
3.1.1 EDA 系统框架结构的提出 …… 115
3.1.2 EDA 系统框架结构的概念 …… 117
3.1.3 EDA 系统框架结构的构成模型 …… 118
3.1.4 EDA 系统框架结构的特点 …… 118
3.2 数据模型与数据管理 …… 119
3.2.1 工程数据库及其管理系统 …… 119
3.2.2 EDA 系统中的数据模型 …… 121
3.2.3 EDA 系统中数据库的层次组织 …… 123
3.3 用户界面管理 …… 125

3.3.1 用户界面管理系统概述 …… 125
3.3.2 UIMS 的两种界面 …… 127
3.3.3 用户界面描述语言 …… 130
3.3.4 小结 …… 133
参考文献 …… 133
第 4 章 模拟 …… 135
4.1 模拟的目的和方法 …… 135
4.1.1 设计自动化与模拟验证 …… 135
4.1.2 模拟级别 …… 136
4.1.3 模拟系统的基本组成 …… 137
4.2 逻辑模拟模型 …… 137
4.2.1 电路网表 …… 138
4.2.2 信号状态值 …… 139
4.2.3 延迟模型 …… 144
4.2.4 元件模型 …… 146
4.3 逻辑模拟算法 …… 149
4.3.1 模拟过程 …… 149
4.3.2 事件表驱动模拟算法 …… 150
4.3.3 三值模拟算法与竞争冒险检测 …… 157
4.4 开关级模拟 …… 160
4.4.1 开关级电路模型 …… 160
4.4.2 计算节点信号状态的强度比较算法 …… 161
4.4.3 等效阻容网络算法 …… 165
4.4.4 信号延迟的计算 …… 167
4.4.5 门、功能块级和开关级的混合模拟处理 …… 171
4.5 高层次模拟 …… 171
4.5.1 VHDL 模拟系统的组成 …… 172
4.5.2 VHDL 内部模型的确立 …… 174
4.5.3 VHDL 模拟算法 …… 181
4.6 交互式模拟与调试 …… 186
4.6.1 高级图形调试器及 DEBUG 功能 …… 186
4.6.2 适应 DEBUG 功能的 VHDL 模型及算法 …… 190
4.6.3 交互式波形显示编辑工具 …… 192
参考文献 …… 195
第 5 章 逻辑综合 …… 197
5.1 逻辑综合的内容和方法 …… 197
5.2 布尔函数的立方体表示法 …… 200
5.3 立方体运算 …… 201

5.3.1 基本概念…… 202
5.3.2 相交和包含判断的具体实现…… 208
5.3.3 锐积运算…… 210
5.3.4 星积运算…… 216
5.4 多输出函数与单输出函数的阵列变换 …… 218
5.4.1 单输出函数的表示形式…… 219
5.4.2 阵列合并…… 219
5.4.3 阵列分离…… 220
5.5 单输出函数质立方体的计算 …… 220
5.5.1 锐积求质立方体…… 220
5.5.2 迭代星积求质立方体…… 221
5.5.3 广义星积求质立方体…… 222
5.6 单输出函数的自动综合 …… 225
5.6.1 选拔法求最小化覆盖…… 226
5.6.2 收缩算法求无冗余覆盖…… 229
5.7 多输出函数的自动综合 …… 230
5.7.1 收缩算法求无冗余覆盖…… 230
5.7.2 选拔法求最小化覆盖…… 232
5.7.3 判别质蕴涵项的 E 算法 …… 234
5.8 组合逻辑电路的变换 …… 236
5.8.1 多级逻辑电路转化为二级逻辑电路…… 236
5.8.2 二级逻辑电路转化为多级逻辑电路…… 238
5.9 时序逻辑电路的自动综合 …… 245
5.9.1 时序机的数学模型…… 246
5.9.2 完全规定时序机的状态最小化…… 247
5.9.3 不完全规定时序机的状态化简…… 254
5.9.4 时序机的状态分配…… 259
参考文献…… 260
第 6 章 高层次综合…… 262
6.1 高层次综合概述 …… 262
6.1.1 高层次综合的概念…… 262
6.1.2 高层次综合的优点…… 263
6.2 高层次综合的内容 …… 263
6.2.1 编译与转换…… 264
6.2.2 调度与分配…… 268
6.2.3 控制器综合…… 269
6.2.4 结果生成与反编译…… 269
6.2.5 高层次综合中的设计空间搜索…… 271

6.3 调度技术 …… 272
6.3.1 调度问题 …… 272
6.3.2 调度算法的分类 …… 274
6.3.3 ASAP 调度算法与 ALAP 调度算法 …… 275
6.3.4 列表调度算法 …… 278
6.3.5 调度中控制结构的处理 …… 281
6.3.6 调度中的功能单元库 …… 288
6.4 分配技术 …… 289
6.4.1 分配问题 …… 289
6.4.2 分配算法 …… 291
6.5 高层次综合中的优化技术 …… 295
6.5.1 具有分枝控制结构时操作的移动 …… 296
6.5.2 控制数据流图的结构变换 …… 297
6.6 小结 …… 300
参考文献 …… 301
第 7 章 故障诊断 …… 303
7.1 故障诊断与测试集 …… 303
7.1.1 测试与故障诊断 …… 303
7.1.2 故障模型 …… 304
7.1.3 测试向量与测试集 …… 304
7.1.4 故障的合并与测试集的压缩 …… 306
7.1.5 测试码的生成问题 …… 310
7.2 敏化路径法求组合电路的测试码 …… 311
7.2.1 单路径敏化法 …… 311
7.2.2 D 算法 …… 313
7.3 布尔差分法 …… 320
7.4 多故障的测试码生成 …… 325
7.4.1 多故障模型的 D 算法 …… 325
7.4.2 高阶布尔差分法 …… 326
7.5 时序电路的测试码生成 …… 328
7.5.1 同步时序电路的迭代展开 …… 328
7.5.2 扩展 D 算法 …… 329
7.5.3 异步时序电路的迭代展开 …… 332
7.6 故障模拟 …… 333
7.6.1 并行故障模拟 …… 334
7.6.2 演绎故障模拟 …… 335
7.6.3 同时故障模拟 …… 336
7.7 可测性设计 …… 337

7.7.1 可测性分析 …… 337
7.7.2 设置观察点和控制点 …… 341
7.7.3 组合电路的可测性电路结构 …… 342
7.7.4 扫描方式电路设计 …… 344
7.7.5 内建自测试设计 …… 346
参考文献 …… 350
第 8 章 形式验证 …… 351
8.1 形式验证的目的和基本方法 …… 351
8.1.1 形式验证的基本概念 …… 351
8.1.2 形式验证的基本方法 …… 352
8.2 基于符号处理的形式推理方法 …… 354
8.2.1 电路的描述 …… 354
8.2.2 公理系 …… 355
8.2.3 基于 FOL 定理证明系统的验证过程 …… 357
8.3 基于时序逻辑的验证 …… 366
8.3.1 时序逻辑简介 …… 366
8.3.2 用时序逻辑描述电路的时序关系 …… 369
8.3.3 利用状态迁移表的验证方法 …… 371
8.4 归纳断言法在逻辑验证中的应用 …… 376
8.4.1 归纳断言法简介 …… 376
8.4.2 一个寄存器传输语言及其公理定义 …… 378
8.4.3 验证实例 …… 381
8.5 提取行为表述式的验证方法 …… 383
8.5.1 验证用描述语言 ISPB 简介 …… 383
8.5.2 事件、历史序列和行为 …… 384
8.5.3 行为表达式 …… 385
8.5.4 由 ISPB 程序求行为表达式 …… 387
8.5.5 用行为表达式进行验证 …… 389
参考文献 …… 390
附录 A VHDL 预定义环境 …… 392
附录 B IEEE 多值逻辑系统标准包 …… 400
附录 C 英汉名词对照表 …… 404

第1章 概 论

1.1 电子系统设计自动化技术发展的回顾

电子线路可分为数字电路、模拟电路和数模混合电路，并且经历了从分立元件到集成电路的发展阶段，其中尤以数字集成电路的发展更为引人注目。如果不特别指明的话，当人们提到大规模集成电路(Large Scale Integrated Circuit，LSI)和超大规模集成电路(Very Large Scale Integrated Circuit，VLSI)时，通常是指数字集成电路。本书着重介绍数字集成电路自动设计的理论和方法。

VLSI的集成度目前已高达每片含几十万至几百万门，并且还在迅速提高。一个微处理器、甚至一个数字计算机系统完全可以集成在一片或数片VLSI之中。如果在小规模集成电路(Small Scale Integrated Circuit，SSI)时期，依靠手工设计还可勉强实现的话，时至今日，却已望尘莫及。可以断言，如果某人有了一个关于新型计算机的奇妙构想，但单纯依靠人力进行研制，那就很可能在它问世之前因此机已落后而夭折。

集成电路技术与其计算机辅助设计(Computer Aided Design，CAD)技术的发展是相伴而行的，相辅相成，相互促进。集成电路CAD技术在其发展过程中，同时还形成了计算机辅助制造(Computer Aided Manufacturing，CAM)、计算机辅助测试(Computer Aided Test，CAT)和计算机辅助工程(Computer Aided Engineering，CAE)等概念。经过发展融合，形成了电子系统设计自动化(Electronic Design Automation，EDA)这一概念。

从历史发展的过程来看，数字系统的复杂度及EDA工具的水平都是逐步提高的。初期阶段，数字系统的设计是分阶段进行的。在不同的阶段，由掌握不同专业知识的技术人员使用相应的EDA工具进行设计，前一阶段的成果(输出)就是后一阶段的工作依据(输入)。不同阶段的衔接，往往要靠人的介入。这一时期的EDA工具多数只能适应某一阶段的工作，并且首先从抽象级别较低的层次(接近物理实现)开始。最近几年的潮流是：从抽象级别较高(例如行为级)的层次开始，对电路作功能描述，自顶向下地跨越各个层次完成整个设计。

以美国为例，60年代至70年代出现了第一代CAD设备。其硬件以16位小型计算机为基础，软件功能主要是交互式图形编辑和设计规则检查，成为集成电路设计和版图设计的不可缺少的工具。80年代出现了第二代CAD系统。由于竞争激烈，IC CAD公司不再制造硬件，而选用高性能的且市场占有率高的32位工作站为工作平台，全力以赴开发软件。第二代CAD系统的功能大体上包括：原理图输入，模拟验证，逻辑综合，芯片布图和印制电路板布图等。这一时期的CAD系统，大都提供单元库的支持。由于单元库是经过精心设计和生产检验的，因而能保证设计质量和提高设计效率。80年代末至90年代初出现的EDA工具可视作第三代。其主要特点是：设计工作可以从高层次开始，使用标准化的硬件描述语言(例如VHDL)描述被设计电路的行为特性，自顶向下地跨越各个层次完

成整个设计。除此之外，第三代 EDA 工具还特别强调设计的可交流性、可再利用性和对大规模电路设计的支持。有人预言："未来的 VLSI 设计者是科学家而不是工程师"。意思是说：未来的 EDA 工具将高度自动化，设计者重点是概念设计，而大部分工程实现中的技术问题都可依靠 EDA 工具解决。

我国 EDA 工具的代表是"七五"期间开发的熊猫系统。它以 32 位工作站为硬件工作平台，以 UNIX 操作系统和 X-Window 为软件开发的基础，其功能大体上相当于国外的第二代 CAD 系统。熊猫系统经过"八五"期间的进一步完善和提高，已在国内广泛使用，用它已成功地设计了上百种电路，并且已进入国际市场。

1.2 从 EDA 的角度来观察 VLSI

VLSI 技术的迅速发展使得它自身成为一项和人类生活密切相关的重要产业。作为信息社会的支柱——计算机和通信，其硬件设备主要是集成电路。在电视、音响、洗衣机、汽车、玩具……等和我们日常生活相关的设备中，也无不大量使用集成电路。社会的需求带动了集成电路产业的发展，而集成电路产业的发展又带动了 EDA 产业的发展。当然，EDA 技术的发展又极大地推动了 VLSI 的设计和生产。

EDA 工具是为 VLSI 的设计、生产服务的，因而它必须适应 VLSI 技术的要求。现在我们就从 EDA 工具的角度来分析 VLSI 的特点。

1.2.1 VLSI 的分类

我们可以从不同的角度对 VLSI 进行分类。

1. 按工艺分类，最主要的有：

- **金属氧化物半导体**(Metal Oxide Semiconductor，MOS)
- **晶体管—晶体管逻辑**(Transistor-Transistor Logic，TTL)
- **发射极耦合逻辑**(Emitter Coupled Logic，ECL)

MOS 工艺最简单，功耗低。在同样生产技术条件下，其集成度最高，速度最慢。而 ECL 工艺复杂，功耗大。在同样生产技术条件下，其集成度最低，速度最快。TTL 介于二者之间。MOS 工艺又可以分为以下三种：

- pMOS—p 沟道 MOS
- nMOS—n 沟道 MOS
- CMOS—互补型 MOS

近年来，CMOS 集成电路最为流行。它成对地使用 pMOS 管和 nMOS 管，静态时仅其中一种管子导通，因而功耗小。此外，它还有速度快、抗干扰能力强、工作电压范围宽等优点。尽管 CMOS 工艺是 MOS 工艺中最复杂的一种，但还是比 TTL 和 ECL 工艺简单，并且 CMOS 工艺也已成熟，应用十分广泛。

各种工艺的加工方法虽然不同，但大体上都是在硅片上作如下加工：光刻、氧化、扩散、离子注入(图 1.1)等，每次加工大都需要一个掩膜。以光刻为例，掩膜覆盖在涂有感光胶的硅片上，其暴露部分因感光而被加固并被保留下来，其余部分则被刻蚀。因此，掩膜个

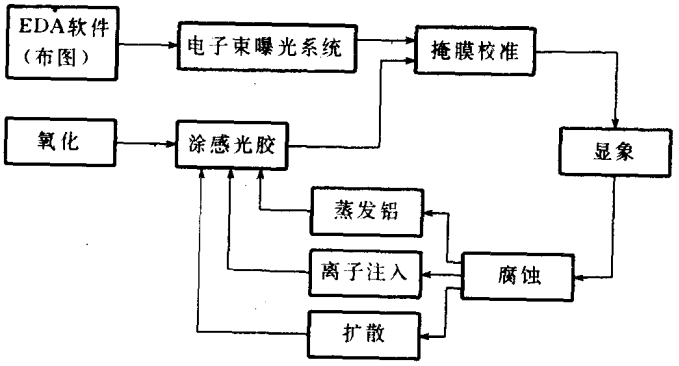

图 1.1 VLSI 芯片加工过程示意图

数大体上反映了加工的复杂程度(参阅表 1.1)。EDA 工具中的布图软件正是为了形成这些掩膜。

表 1.1 各种工艺复杂度的比较

项目 \ 工艺	nMOS	CMOS	TTL
每门占用芯片面积(μ^2)	1 000	1 500	30 000
每门延迟时间(ns)	1	0.7	0.2
每门静态功耗(mW)	0.5	0.001	10
掩膜数	6	8	7～8
扩散和离子注入次数	3	4～6	5

2. 按生产目的分类

- **通用集成电路**
- **专用集成电路**(Application Specific Integrated Circuit, ASIC)

生产厂家以供应市场为目的生产的芯片大多属于通用集成电路,例如微处理器芯片、存储器芯片、计算机外围电路芯片等。

为某个或某些用户专门用途而生产的芯片则属于专用集成电路。前者生产批量大,设计费用分摊在每个芯片上就不大。因此,对芯片的性能(和市场竞争力紧密相关)和芯片的利用率(和生产成本紧密相关)要求高,而对设计成本、设计周期的要求可以放宽。后者则着重于设计成本和设计周期,对 EDA 工具提出了更高的要求。

3. 按实现方式(设计风格)分类

用户在完成了电路设计之后,可选用以下不同的实现方式:

- **全定制**(Full-Custom)方式
- **半定制**(Semi-Custom)方式
- **现场可编程逻辑器件**(Programmable Logic Device,PLD)方式

当用户把自己的设计交给集成电路制造商,以加工订货的方式要求制造商供应成品,

这属于全定制或半定制方式。

当用户从市场上买来现场可编程逻辑器件，自己把电路“写入”该器件时，就是现场可编程方式。

全定制方式是基于晶体管级的芯片设计，仔细考虑每个管子的尺寸、位置及管子间的互连关系，其目标是密度高、速度快和功耗小。这种方式设计成本高且周期长。作为一种改进，EDA 工具提供标准单元库，库中有许多精心设计好的具有一定逻辑功能的标准单元。这些标准单元的逻辑功能及版图事前经过生产检验，具有较高的质量。把用户的原始设计转换成由标准单元构成的电路，很容易设计出整个电路的版图，效率得到提高。

半定制方式通常是指门阵列(Gate Array)方式。集成电路制造商预先准备好称作母片(Master Slice)的半成品(图 1.2)，母片上以一定间距成行、成列地排列着形状、大小相同的基本单元，基本单元通常是门或成对的 n 管和 p 管(CMOS 工艺)。以母片为基础，对用户的电路进行布图的方式称为门阵列方式。由于这种方式采用了规则的布图模式，有利于布线规则的形成；它是在半成品(母片)的基础上加工，因而生产周期短、生产成本低，80 年代开始流行。它的缺点是：由于基本单元之间保持固定的间距用于布线，必然存在某些地方走线稀疏(芯片面积利用率不高)；而另一些地方走线拥挤，甚至连线布不通。为了接通连线，还可能造成某些单元未被利用。

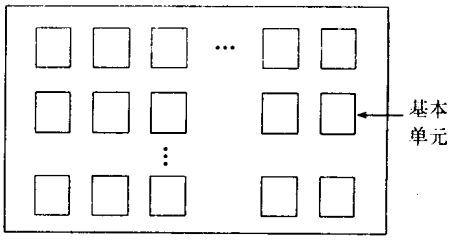

图 1.2 门阵列模式示意图

为了进一步缩短设计周期和降低设计成本，80 年代出现了现场可编程逻辑器件，90 年代上升趋势更猛。集成电路制造商向市场提供已封装完毕的芯片，其逻辑功能却可以由用户自己使用 EDA 工具“写入”。从生产厂家来看，可编程逻辑器件是通用器件，可以批量生产以降低成本。从用户角度看，自己可以将设计好的电路“写入”芯片，使之成为专用集成电路。有些可编程逻辑器件可多次“重写”，特别适合于新产品试制或小批量生产。可编程逻辑器件的缺点是：(1)芯片内部连线较长，速度相对较慢。(2)集成度相对较低。详细介绍见下节。

1.2.2 芯片布图

版图设计(layout)的主要任务是在电学性能的约束条件下，在芯片上安置所有单元电路(布局，placement)，并完成各单元之间的电学上的连结(布线，routing)，目标是芯片最小、连线总长度最短。集成电路自动版图设计主要模式有：

- **门阵列**(Gate Array)模式——前面已介绍过
- **多元胞**(Policell)模式
- **任意元胞**(Custom Cell)模式
- **可编程逻辑器件**(Programmable Logic Device，PLD)模式

多元胞模式也称作标准单元模式。标准单元是预先设计好的功能单元，其外形为等高而不等宽的矩形，按引出接点排列位置的不同分为单边单元和双边单元(图 1.3)。这些单

元块(元胞)的排列方式示于图 1.4，连线分配在单元行之间的水平通道区或单元行的两端及单元间的垂直通道区中，通道区的高度(或宽度)可根据布线设计的需要加以调整。在这种模式下，布图软件根据用户的逻辑图从标准单元库中调出单元块放于合适位置(布局)，然后自动布线，最后给出掩膜板数据。和门阵列模式相比，多元胞模式中芯片上所有单元均被利用，所以芯片面积较小。

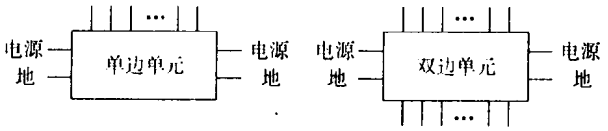

图 1.3　标准单元引出接点示意图

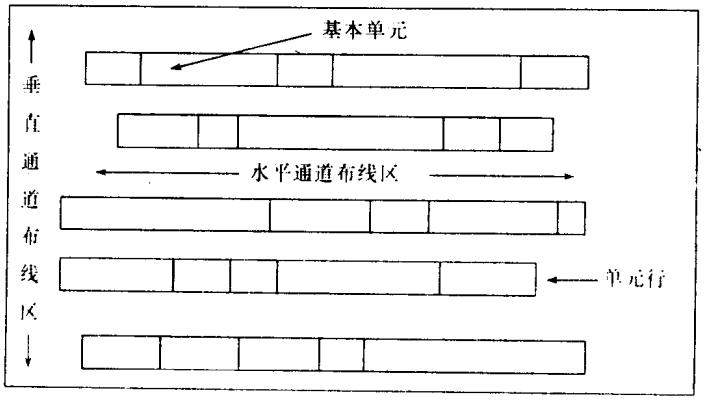

图 1.4　多元胞模式示意图

任意元胞模式与多元胞模式相比，其主要不同之点是放松了对单元块外形的限制——只要求其边框线是水平线或垂直线即可(图 1.5)。这种方法必然会提高芯片利用率和设计方法的灵活性，但使布局布线算法复杂，增加了布图软件的困难。

可编程逻辑器件在下一节中介绍。

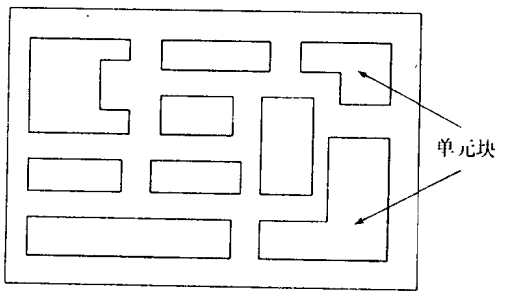

图 1.5　任意元胞模式示意图

1.2.3 可编程逻辑器件

图 1.6 表示可编程逻辑器件 PLD 的分类。我们将沿着 PLD 器件发展的历史轨迹，通过实例介绍其基本原理与实现方法。假定逻辑电路应实现的布尔函数为

$$y_1 = x_1 \cdot x_2 \cdot \bar{x}_3 + \bar{x}_2 \cdot x_3 + \bar{x}_1 \cdot x_3 \tag{1-1}$$

$$y_2 = x_1 \cdot x_2 \cdot x_3 + \bar{x}_1 \cdot \bar{x}_2 \tag{1-2}$$

令

$$p_1 = x_1 \cdot x_2 \cdot \bar{x}_3$$

$$p_2 = \bar{x}_2 \cdot x_3$$

$$p_3 = \bar{x}_1 \cdot x_3$$

$$p_5 = x_1 \cdot x_2 \cdot x_3$$

$$p_6 = \bar{x}_1 \cdot \bar{x}_2$$

则

$$y_1 = p_1 + p_2 + p_3$$

$$y_2 = p_5 + p_6$$

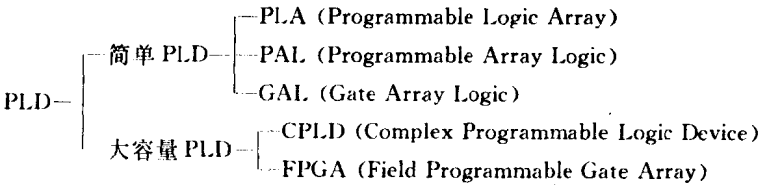

图 1.6 PLD 器件的分类

可以用硬连逻辑实现上述布尔函数(图 1.7)，也可以用可编程的规则矩阵实现上述布尔函数(图 1.8)。图 1.8 中有一个“与”阵列和一个“或”阵列，阵列中交叉点上若有一个小圆圈，表示该点被选中。因此，阵列中小圆圈的分布情况，表示了该阵列所实现的逻辑。可编程逻辑器件的含义是：可根据用户指定的逻辑，构造相应的阵列并加以实现。

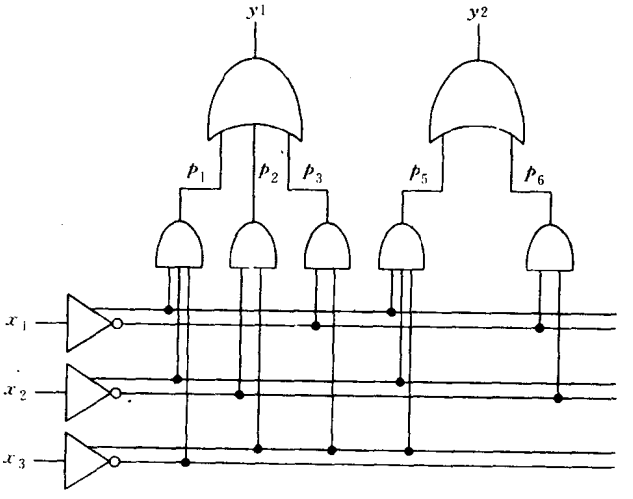

图 1.7 用硬连逻辑实现布尔函数

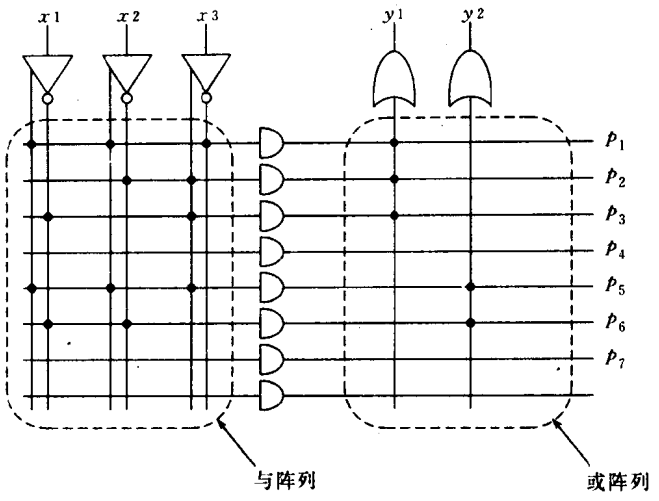

图 1.8　用可编程的规则矩阵实现布尔函数

如果需要量比较大，可选择由 IC 制造厂家用掩膜实现阵列逻辑；如果需要量很小，则可选择现场可编程逻辑器件，由用户自己实现阵列逻辑。从实现的方法来分，现场可编程技术可归类如下：

- **熔丝技术**——平时连通，加电可使熔断
- **反熔丝技术**——平时不连通，加电可使连通
- **电可写技术**

现场可编程逻辑器件又可分为一次写入不可改写的和可擦除后重写的两种；后者又可分为紫外光擦除和电擦除的两种。

图 1.8 所示电路仅能实现组合逻辑功能。在此基础上加入可编程的寄存器，既可实现组合逻辑功能又可实现时序逻辑功能，如图 1.9 所示。事实上，图 1.9 乃是简单可编程逻辑器件 GAL 的雏形。

简单可编程逻辑器件的缺点是：芯片内的逻辑与引脚有着密切的对应关系，当 IC 的制造工艺已相当进步，却因引脚数目限制了芯片内的逻辑功能时，这种电路结构的缺点就变得难以容忍了。

80 年代中期，出现了现场可编程门阵列(FPGA)，其基本结构示于图 1.10。图中每一逻辑模块都是可编程的同一结构的模块，简单地说，其规模类似于一个 GAL，用它可实现一定的逻辑功能。图中所示的布线区也是可编程资源，可按照用户的要求连接各逻辑模块和 I/O 模块。显然，FPGA 的集成度比 GAL 的集成度有很大的提高，可以在一个芯片内实现更复杂的逻辑。

FPGA 之所以能实现更复杂的逻辑，是因为可通过可编程的布线区实现逻辑模块之间的相互连接，并使最后形成的逻辑层次数不受限制。这种连接线段是可增加的，连接线段的数目事前不可预估，因而总延迟时间也不可预估，这是此种电路结构的缺点。

大容量 PLD 中，除了 FPGA 之外，还有一种称作复杂可编程逻辑器件 CPLD。它的逻辑模块规模更大，布线区也作了一些改进。现以 Altera 公司的产品为例，说明 CPLD 的结