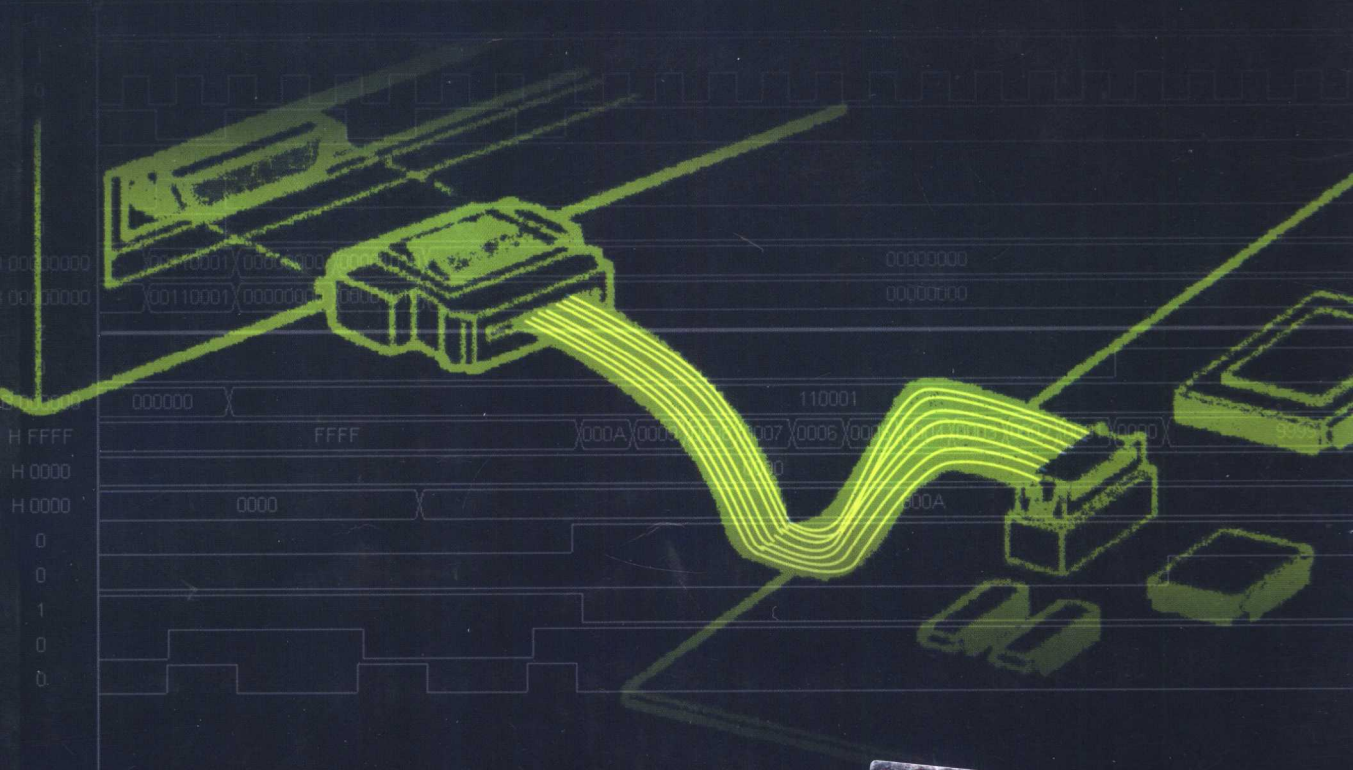
VHDL
与微机接口设计

赵世霞　杨 丰　刘揭生　编著



清华大学出版社

VHDL与微机接口设计

赵世霞　杨　丰　刘揭生　编著

清华大学出版社

北　京

内 容 简 介

本书是在教学实践的基础上编写的，主要目的是通过具体的实例来学习 VHDL 语言，并掌握微机接口电路的设计方法。

本书的内容分为 3 部分：第 1 部分是 VHDL 语言基础，介绍了 VHDL 语言的数据类型、基本语句、程序结构；第 2 部分是基于 VHDL 语言的微机接口电路设计，以常用的微机接口电路为例讲述了电路设计的方法，可以为设计大型复杂电路打下基础；第 3 部分是设计工具的使用，以 Altera 公司的 MAX+plus II 为例详细地讲述了软件的安装与使用，以图解的方式给出了详细的步骤和操作方法。

本书可以作为电子、计算机类大学生的实践类课程教学用书，也可作为各类电子技术人员的学习参考书。

图书在版编目（CIP）数据

VHDL 与微机接口设计/赵世霞，杨丰，刘揭生编著. —北京：清华大学出版社，2004.7
ISBN 7-302-08547-1

Ⅰ.V…　Ⅱ.①赵…　②杨…　③刘…　Ⅲ.①硬件描述语言，VHDL-高等学校-教材　②微型计算机-接口-高等学校-教材　Ⅳ.①TP312　②TP364.7

中国版本图书馆 CIP 数据核字（2004）第 037699 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　**邮　　编**：100084
社 总 机：010-62770175　**客户服务**：010-62776969
组稿编辑：欧振旭
文稿编辑：鲁秀敏
封面设计：秦　铭
版式设计：俞小红
印 装 者：北京国马印刷厂
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：20　**字数**：446 千字
版　　次：2004 年 7 月第 1 版　2004 年 7 月第1次印刷
书　　号：ISBN 7-302-08547-1/TP・6134
印　　数：1～5000
定　　价：28.00 元

前　言

现代电子技术的发展促进了传统设计方法的进步，掌握 VHDL 语言和计算机接口技术，使用 EDA 设计电子系统是电子类的大学生应具备的基本技能。微机接口是各大学普遍开设的计算机技术课程，但是我们的教学实验大部分是采用专用芯片，在内容和形式上基本是固定的。现在包括计算机在内的电子系统正朝着片上系统 SoC（system on chip）发展，为适应这种新的发展趋势，课程的知识结构需要更新，实验的手段也要有比较大的改观。VHDL 语言是面向硬件描述对象的语言，只有结合硬件设计对象来学习，才能更好地理解和掌握它，因此在学习方式上应与其他的高级语言有所不同，必须要面对硬件设计的实体。编写此书的目的是将 VHDL 语言的学习和微机接口的设计紧密地结合起来，在学习硬件描述语言 VHDL 的基础上，可亲自动手设计一个实际的微机接口芯片。从程序设计、仿真模拟、系统综合优化，目标电路代码装载到可编程芯片中，直至最后连接到微机总线上编写接口程序进行调试。这是一个综合多方面知识，融会贯通的全过程训练，可提高学习硬件的兴趣和信心。

本书是在教学实践的基础上编写的，书中的实例都经过实践检验。通过一段时间的教学实践，我们发现学习一种新的语言对于大多数学生难度都不会太大，但是对于设计并调试一个实际的应用系统来说，有相当一部分同学开始时都感到困惑和迷茫。因此，类似采用 EDA 设计的这种教学实验，我们必须要给学生开设，但一定要有针对性，实验也不能设计得太大或太繁杂。要让学生通过实验，能够得到一个完整的成果，而不是半成品或中间件，这样将有利于激发学习欲望。从开始心里没有底到逐步相信自己，一直到最后完成作品，每一步都有一种成功的喜悦。经过教学尝试，我们还认为，VHDL 语言与微机接口设计结合起来是合适的，难度适中，也可以促进学生对微机接口的学习更深入，不仅了解接口芯片外部的使用特性，而且更加清楚接口芯片内部的工作原理。在老师的指导下，充分发挥学生的创造性和综合运用知识的能力，从而达到从验证学习到研究型的转变，课程教学从以教师为主体到以学生为主体的转变。在学习 VHDL 基本语法和程序设计的基础上，利用本书和现有的微机接口实验装置，可以开设基于 VHDL 设计的各种微机接口实验。实验的平台可利用各学校现有的微机接口实验装置，不需要增加太大的投资。例如，我们就是利用原有的清华大学科教仪器厂的 TPC-H 实验装置，另外再配上一块 FPGA 的实验板，给学生开设实验的。在此特别感谢参加“SRT 项目”和“工程实践”的同学耿云川、陈彧和邹轶等所做的工作。

本书的内容有些是通过设计实践后总结出来的，也有的是经过摸索而得到的（例如该软件在不同操作系统的平台下装载芯片要安装配置文件的问题），将其介绍给大家，希望在学习时少走些弯路。

由于作者水平所限，本书在编写的过程中若有任何错漏，恳请广大读者批评指正。

作　者

2004 年 5 月于清华大学

目　录

第 1 部分　VHDL 语言基础知识

第 1 章　概述 2
1.1　电子器件的发展和现状 2
1.2　设计方法的发展 4
1.3　层次化的设计与 VHDL 的应用 5

第 2 章　VHDL 语言与程序结构 7
2.1　语言特点与设计流程 7
2.2　VHDL 程序基本结构 9
2.2.1　实体 9
2.2.2　结构体 12

第 3 章　VHDL 语言基础 16
3.1　标识符 16
3.2　数据对象（object） 17
3.3　数据类型 19
3.3.1　标准数据类型 20
3.3.2　用户自定义的数据类型 21
3.3.3　IEEE 标准数据类型 23
3.4　词法单元 24
3.5　运算操作符与表达式 25

第 4 章　VHDL 语言库的使用 29
4.1　库的作用与使用 29
4.2　库 29
4.3　程序包 30

第 5 章　VHDL 基本语法 33
5.1　并行语句 33
5.1.1　进程语句（process） 33
5.1.2　WAIT 语句 34
5.1.3　信号赋值语句 35

5.1.4 并行断言语句（assert）.....37
5.1.5 块语句（block）.....37
5.1.6 生成语句（generate）.....38
5.1.7 子程序（sub program）.....39
5.1.8 其他.....41
5.2 顺序语句.....42
5.2.1 变量和信号的赋值.....42
5.2.2 IF 语句.....43
5.2.3 CASE 语句.....44
5.2.4 LOOP 语句.....45
5.2.5 NEXT 语句.....46
5.2.6 EXIT 语句.....46
5.2.7 RETURN 语句.....47
5.2.8 NULL 语句.....47
5.2.9 REPORT 语句.....48
5.3 命名规则及注释.....48
5.4 VHDL 设计举例.....48
5.4.1 8 位移位寄存器的设计.....49
5.4.2 4 位微处理器的设计.....50
思考题.....63

第 2 部分 基于 VHDL 语言的微机接口电路设计

第 6 章 微机接口电路的设计.....66
6.1 设计并行接口芯片.....67
6.1.1 理解芯片的功能和结构.....67
6.1.2 了解芯片的工作方式与编程设置.....69
6.1.3 选择方案构思整体的实现思路.....71
6.1.4 实现相应的功能.....72
6.1.5 优化与功能取舍.....83
6.1.6 仿真模拟并装载芯片进行验证测试.....84
6.1.7 设计实例.....86
思考题.....153
6.2 设计串行接口芯片.....153
6.2.1 串行通信的基本概念与术语.....153
6.2.2 理解芯片并构思整体的实现思路.....160
6.2.3 用 VHDL 实现设计的思路.....166

6.2.4　设计实例 166
6.3　设计定时/计数器芯片 8253/8254 191
6.3.1　理解芯片的结构与功能 191
6.3.2　构思整体的实现思路 201
6.3.3　设计实例 202
6.4　设计 PS/2 键盘接口 225
6.4.1　PS/2 接口通信 225
6.4.2　键盘扫描码表 227
6.4.3　设计实例 229
思考题 244

第 3 部分　设计工具的使用

第 7 章　设计软件的使用 246
7.1　MAX+plus II 的使用 246
7.1.1　MAX+plus II 概况 246
7.1.2　软件的安装与认证 247
7.1.3　软件的使用 251
7.1.4　软件在使用中的常见问题 282
7.2　Quartus II 的使用 287
7.2.1　软件的安装 287
7.2.2　软件的使用 287
7.3　配置文件的安装 292
附录 A　PCI 接口卡的编程 296
A.1　使用 PCI 接口卡在 Windows 2000 下的编程 296
A.1.1　基本输入/输出函数 296
A.1.2　中断函数 297
A.2　使用 PCI 接口卡在 Windows 98 下的编程 298
A.2.1　替换基地址值的方法 298
A.2.2　使用中断时的编程 300
附录 B　ASCII 码表 306
附录 C　常用 DOS 功能调用命令 307
参考文献 309

第 1 部分

VHDL 语言基础知识

第1章 概 述

1.1 电子器件的发展和现状

电子技术的发展总是同电子器件的发展密切相关的，由于电子器件的不断更新换代，电子技术得到了飞速发展，当今信息技术被广泛应用在国民经济的方方面面。多媒体技术的普及、高速宽带网络的建设、数字电视的出现以及与我们日常生活息息相关的各种家用电器，都离不开微处理器、存储器和一些采用行业标准的专用芯片。

这些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几个阶段，其发展趋势是体积越来越小，集成度越来越高。随着半导体技术的迅速发展，设计与制造集成电路的任务已不完全由半导体厂商来独立承担，系统设计师们更愿意自己设计专用集成电路（ASIC）芯片，而且希望 ASIC 的设计周期尽可能短，最好在自己的实验室里就能设计并制造出适用的 ASIC 芯片，并且能够得到实际的应用。使用 ASIC 完成电子系统的设计，其优点是集成度高、保密性强、可大幅度地减少印刷电路板的面积和接插件、电路性能好并能降低装配和调试费用、更适合较大规模批量生产、降低生产成本。例如，现在微机中使用的显卡、网卡、数字电视等都是专用的 ASIC 芯片。采用 ASIC 设计的不足之处是一次性的投资大，设计周期一般比较长，不便于以后修改设计，因而近几年在 ASIC 领域出现了一种半定制电路，即现场可编程逻辑器件 FPLD。目前使用较多的可编程逻辑器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 gate array）。

早期使用的可编程器件有 PROM、EPROM、E2PROM、PAL、GAL 等，这些芯片的容量一般比较小，只能完成简单的数字逻辑功能，这些器件基本上是使用专用的编程器进行装载。CPLD 和 FPGA 的芯片是在早期使用的 PAL 和 GAL 基础上发展起来的，相比早期的可编程器件，现场可编程逻辑器件 FPLD 的容量大，器件的容量远远大于 PAL 和 GAL，更适合做时序和组合逻辑电路的设计，不但集成度高，而且逐渐向低功耗发展，芯片的使用电压有+5V、+3.3V 和+2.5 V 几种，有的芯片外部是+5V 而内部的工作电压则是+3.3V 或+2.5V。现在市场上 FPLD 的品种很多，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，例如，Xilinx 的 XC 系列；Altera 的 EPLD 系列、FPGA 系列；Lattice 的 ispLSI、ispGAL 等。由于各个公司的 FPLD 结构不同，使用的装载电缆线是不一样的，设计软件也不同，但共同的特点是：现场可编程逻辑器件 FPLD 都可以在系统进行编程加载程序，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装载，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，实现系统的微型化和可靠性。目前工程设计人员通常都采用这种方法进行电路设计。

1．CPLD 或 EPLD 芯片

这类器件的使用特点是：芯片一经上电加载即已完成编程，不必在每次上电时重新进行加载，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。例如，Lattice 的 ispGAL 芯片可以在系统进行编程加载程序，也可以与该公司的 ispLSI 芯片在同一个电路中加载程序。CPLD 在结构上主要由可编程逻辑宏单元 LMC（logic macro cell）围绕中心的可编程互连矩阵单元组成，其中 LMC 的逻辑结构比较复杂，并具有复杂的 I/O 单元互连结构，可以由用户根据设计的需要生成特定的电路结构，完成一定的功能。

2．FPGA 芯片

这类器件不像 CPLD 或 EPLD，每次上电使用时不管是否改变程序都要对芯片进行加载。现在 FPGA 芯片的容量一般比 CPLD 或 EPLD 的容量要大得多，更适合于做较大系统的复杂设计。FPGA 芯片通常包含 3 类可编程资源：可编程的逻辑功能块、可编程 I/O 块和可编程的内部互连。可编程的逻辑功能块是实现用户功能的基本单元，它们通常排成一个阵列，遍布于整个芯片中；可编程 I/O 块完成芯片上的逻辑与外部封装管脚的接口，常围绕着阵列排列于芯片四周；可编程的内部互连包括各种长度的连接线段和一些可编程连接开关，它们将各个可编程逻辑块或 I/O 块连接起来，构成特定功能的电路。

Xilinx 的 XC 系列器件采用的是现场可编程门阵列 FPGA，现场可编程门阵列是一种类似门阵列的结构，它的基本单元以阵列的形式排列在芯片上，但它不像门阵列那样由连线掩膜确定其最终的逻辑功能，而是将规则的连线阵列也已做好，其逻辑功能由各连线节点的控制开关的通断来确定。这些节点的控制开关的值有各种不同的控制方法，如静态随机存储器控制（SRAM）、反熔丝（antifuse based）控制以及由快闪烁存储器（Flash）控制等几大类。

SRAM 控制的 FPGA 器件是用静态随机存储器中存储的数值来控制芯片中可编程节点的通断，以实现芯片的设计功能。这种 FPGA 在使用时，需要在系统加电时首先进行功能初始化，将存储器的内容加载到芯片的控制器中。

反熔丝控制的 FPGA 器件是用反熔丝单元来控制可编程器件内的可编程节点的通断，使芯片中每一部分具有应有的逻辑功能，以实现器件的设计功能。反熔丝单元是一个被动的两端器件，通常情况下处于开路状态，在施加充足的电压后，能够永久地导通。用于现场可编程器件的反熔丝其占用芯片面积非常小，速度也很高，这方面的性能几乎可以与门阵列相比。但这种 FPGA 在编程时需要专用的编程设备，而且芯片功能一经编程确定后，不能再修改，反熔丝控制的 FPGA 器件价格最低。

Flash 控制的 FPGA 器件是用快闪烁存储器的数值来控制 FPGA 节点的通断，实现现场可编程目的。这种 FPGA 的工作特性与 SRAM-FPGA 相似，但与 SRAM-FPGA 相比，它的单元面积小。由于 Flash 具有不挥发性，使用时不必在每次上电时都重新进行功能加载，

所以使用是很方便的。

1.2 设计方法的发展

传统的手工设计方法一般是根据系统的要求，首先画出系统的硬件流程图，再根据功能划分成不同的模块，设计过程一般从底层开始，先要选择具体的元器件，用所选择的元器件进行各功能模块的逻辑电路设计，手工画出一张张的电路原理图，根据原理图制作印刷电路板，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。对整个系统的仿真、调试只能在完成硬件设计以后才能进行，系统设计中的问题在调试的后期才能发现，如果出现设计中没有考虑到的问题，就要再从底层重新设计，这样的设计周期一般较长。设计结果是若干张的电路原理图和信号的连接表，如果是一个大的系统，将是一大摞图纸，以后系统出现问题，查找修改起来都不直观。

上述过程是从底层开始，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。因此这种电子系统的传统的设计方法称为是自底至上（bottom-up）的设计。这里的底指的是设计树的末枝，设计过程必须从存在的基本单元模块出发，基本单元模块必须是已经设计成熟的单元，也可采用其他项目已开发好的标准单元。

由于电子器件的更新换代，在现代数字系统设计中，现场可编程器件 FPGA 和 CPLD 的使用越来越广泛，与此同时基于大规模可编程逻辑器件的 EDA（electronic design automation）硬件解决方案（EDA solution）也被广泛采用。这使得电子电路的设计方法也发生了根本性的变化，出现了电子电路设计自动化。

计算机应用的普及，自然产生了计算机辅助设计（CAD）。最早的电子 CAD 软件仅仅是一些绘图软件，包括绘制电路原理图、绘制印刷电路板图、绘制集成电路芯片板图以及一些简单的数值计算等。当出现了自动设计、验证和自动布局布线工具后，这类软件称之为第一代的 EDA 软件。后来又出现了第二代的 EDA 软件，它包括逻辑综合、仿真以及“自顶向下”的设计等。近年来又出现了第三代的 EDA 软件，称为电子系统设计自动化 EsDA，可以通过概念输入（框图、公式等）自动生成各种设计结果，包括 ASIC 芯片设计结果、电路原理图、PCB 板图以及软件等，并且可以进行机电一体化设计。与传统的设计方法不同，现代电子工程师们设计系统的过程是首先描述系统，然后用 EDA 工具在计算机上进行系统级仿真，设计适合自己用的 ASIC 芯片，用通用和专用芯片构成系统，进行功能模拟和带时延的仿真，布 PCB 板，对 PCB 板进行仿真，最终生产调试成功。

EDA，即电子设计自动化。所谓自动化是指利用计算机完成电子系统设计，现在已经逐渐成为电子系统的主要设计手段，尤其是采用可编程器件和软件仿真模拟方法的使用，给传统的电路设计方法带来了重大的变革，它使得设计工程师们从繁杂而零乱的工作中解放出来，而把着眼点放在电路的设计上，是一种节省时间而又高效率的现代设计理念。EDA 技术以计算机为工具，设计者只需要完成对系统功能的描述，就可以由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工作。其中模拟硬件电路

在实际工作时的时序关系是相当重要的，因为系统设计上的错误通过仿真模拟波形时就可以发现，而不是等到线路板调试时才发现错误，即使是在线路板调试时又发现错误，在外部连接线已经固定的情况下，只要对内部的软件设计进行改进，就可达到修改设计方案的目的，这种方法比起传统的电路设计方法进步多了，修改设计如同修改软件一样方便。

当今的硬件设计方法有几大优点：一是设计方法由手工设计变为自动设计，可以大大提高设计效率和设计质量，缩短设计周期；二是在系统设计和各个过程中可分别进行仿真，保证了设计的正确性，使得设计能够一次成功；三是能够根据实际需要来自行设计 ASIC 芯片。

可编程逻辑器件和 EDA 技术给硬件系统设计者提供了强有力的工具。如今，只要拥有一台计算机、一套相应的 EDA 软件和空白的可编程器件芯片，在实验室就可以完成数字系统的设计与生产。可以说，当今的数字系统设计离不开可编程器件和 EDA 设计工具。

1.3 层次化的设计与 VHDL 的应用

1．自顶向下（top-down）的设计方式

一般来说，EDA 解决方案是一种采用计算机自顶向下（top-down）的设计方式。这里的顶指的是设计树的树根，按照数字系统的功能描述，把系统划分为若干个功能模块，然后再把每个模块划分为不同的层次，由高层次到低层次逐步细化。这样的设计过程称之为自顶向下的设计方式。在底层设计时对逻辑进行必要的描述，并依赖特定的软件执行逻辑优化（logic optimization）与器件映射（device mapping），自顶向下设计的特点是每一层次划分时都要对某些目标进行优化，这些目标包括工作速度、芯片面积和芯片成本等。最后再使用由各芯片生产厂商提供的编译器执行布线（route）和网单优化（netlist optimization）。而直接采用原始逻辑图或布尔方程输入进行电路设计虽然对于简单的逻辑可以获得非常有效的结果。但是对于复杂的系统设计，应用它们就很容易产生错误，而必须依靠一种高层的逻辑输入方式，这样就产生了硬件描述语言 HDL（hardware description language）。所谓硬件描述语言，就是对实际的硬件设计用语言的方式来描述，能够把复杂的电路设计用形象化的语言方式表示出来，可以描述硬件电路的功能，信号连接关系以及定时关系的语言，它能比电路原理图更能有效地表示电路的特征。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，是该设计方法的一个重要特征。

硬件描述语言有 HDL、AHDL、Verilog HDL 和 VHDL 等。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）属于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范又比较简单。其中符合 IEEE-1076 标准的 VHDL 的应用成为 EDA 解决方案中的首选。VHDL 这种行为描述性语言将被广泛应用到新一代 EDA 硬件设计方案中，因此对 VHDL 的应用成为 EDA 解决方案的核心，更是整个电子逻辑系统设计的核心。

2．VHDL 语言的应用

VHDL 语言源于美国政府于 1980 年开始启动的超高速集成电路（very high speed

integrated circuits，vHSIC）计划。在这一计划的执行过程中，专家们认识到需要有一种标准的语言来描述集成电路的结构和功能。这样，vHSIC 的硬件描述语言（vHSIC hardware description language），即 VHDL 诞生了。很快，这一标准被美国电气和电子工程师协会（IEEE）所承认。

VHDL 语言作为高级硬件行为描述型语言，如今已经广泛被应用到 FPGA/CPLD 和 ASIC 中的设计。严格地讲，VHDL 是一种用来描述数字逻辑系统的“编程语言”。它通过对硬件行为的直接描述来实现对硬件的物理实现，代表了当今硬件设计的发展方向。VHDL 是为了满足逻辑设计过程中的各种需求而设计的。

第一，它是可以用来描述逻辑设计的结构，比如逻辑设计中有多少个子逻辑，而这些子逻辑又是如何连接的。除此之外，VHDL 并不十分关心一个具体逻辑依靠何种方式实现，而是把开发者的精力集中到逻辑所实现的功能上。

第二，VHDL 采用类似高级语言的语句格式完成对硬件行为的描述，具备更强的模块化能力，并拥有良好的可读性以及程序的移植性。另外，VHDL 淡化状态机，与或表达式等早一代硬件描述语言中的元素，用更类似于高级语言的表达式取代。这些也是为什么把 VHDL 称为“编程语言”的原因。

第三，VHDL 给出逻辑的模拟与调试为设计工作提供了最大的空间。VHDL 调试的过程是相当灵活的：一方面可以使用传统的调试方法，比如适用传统的波形激励或编写测试向量；另一方面，可以使用一些 VHDL 原码调试器，这类调试器可以大大加快 VHDL 程序调试的速度，因为它可以像调试软件一样单步跟踪调试每一条语句，并且可以设置断点，观察内部变量等。这些功能是传统的调试仿真方法所不具备的。这种调试器比较著名的有 Aldec 的 Active-HDL。拥有高效率的生成代码，能够节省大量的资源。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。

鉴于 VHDL 具有以上诸多优点，只要开发者具备一定的高级语言程序设计基础，拥有 Pascal、C 等计算机高级语言的基础，同时又了解一些基本数字电路的设计方法，在此基础上来学习 VHDL 程序设计应该是比较容易的，可以轻松地掌握 VHDL 使硬件工作软件化。现代电子系统设计人员应该把 VHDL 语言作为一种基础知识来学习，并要求能够熟练地使用 EDA 的设计工具。

在 EDA 解决方案中应用 VHDL 有助于缩短数字系统的开发周期。但还应该注意，除了靠编写 VHDL 程序简化逻辑之外，还需要选择合理的 HDL synthesis。不同公司的 HDL synthesis 所支持的语法并不相同，而且生成的代码效率也不同。经实验对比，相同的程序经过不同的编译器编译，芯片资源的消耗相差高达 10%。另外，由于知识产权和专利保护问题，目前国际上尚无统一的集成化开发工具可以完成从逻辑输入到下载所需的全部工作，这也给 VHDL 在 EDA 解决方案中的广泛应用带来了一定的困难。但是总的来讲，广大厂商均遵循 VHDL'87 这一通用标准并互相提供良好的软件接口，在某种程度上可以缓解该矛盾。随着 VHDL'93 标准的广泛应用以及第二代可编程逻辑器件的推出，VHDL 必将在未来的 EDA 解决方案中发挥不可替代的作用。

第 2 章　VHDL 语言与程序结构

2.1　语言特点与设计流程

1．VHDL 语言的特点

VHDL 语言目前主要是对数字电路设计的描述，对模拟电路的设计尚不能很好地表达。VHDL 语言在编程时要更加规范，程序结构要适合整个系统的硬件结构，要符合各模块的信号时序关系，以及数据流的走向。VHDL 语言的设计格式更是面向具体的硬件对象的语言，因此任何独立于硬件实体的程序设计是没有意义的。现在 EDA 设计代替了传统的手工设计，都是以 FPGA、CPLD、EPLD 等可编程器件作为系统中硬件的载体，大部分是以 VHDL 作为设计语言，并针对所使用的芯片来选择不同公司的软件在计算机上进行设计、综合。这种用程序设计完成的硬件结构可装载到对应的可编程器件中，进行仿真、模拟、验证。VHDL 的语言特点主要有：

（1）更加类似软件上的高级语言，具备更强的模块化能力并拥有良好的可读性以及程序的移植性；

（2）淡化状态机，与或表达式等早一代硬件描述语言中的元素，用更类似于高级语言的表达式取代；

（3）拥有高效率的生成代码，能够节省大量的资源。

2．VHDL 的设计流程

应用 VHDL 语言进行电子设计，首先要了解基本的设计流程。在 EDA 解决方案中采用 VHDL 要经过的流程如图 2.1 所示。

从图 2.1 可以看出，在分析系统指标后，就进入了设计阶段，首先是设计输入，可以采用的输入方式有 VHDL 语言、VHDL 语言与逻辑图混合输入或采用逻辑图输入，这一步通常都使用芯片生产商提供的开发工具，有关这方面的内容将在后面介绍。接下来是对所输入的源文件进行编译，即 HDL Synthesis。通常称为合成或逻辑合成，这一步通常由三个过程组成，分别是 HDL 语言合成（language synthesis 或 HDL compilation）、逻辑优化（optimization）、目标映射（technology mapping）。前两个过程很好理解，最后一步是为了适应不同公司的编译器而生成 EDIF（electronic design interchange format）的中间文件，也有的生成 AHDL、DSL、QDIF、XNF 等内部网单描述文件。有些公司的软件自带编译器，而也有些公司的软件不带编译器，这时需要使用第三方软件，比较著名的有 Exemplar 公司的 Leonardo Spectrum 和 Synplicity 公司的 Synplify 等。一般这类工具都采用 Behavior *Extracting* Synthesis Technology（B.E.S.T.）和 Synthesis Constraints Optimization Environment

（SCOPE），这两种技术可以提高 VHDL 逻辑合成的效率和可靠性。另外，这类工具在生成 EDIF 文件的同时还生成 VHDL 格式的网单，可以用于对逻辑功能调试（functional simulator）。这里的逻辑功能调试也就是图中的功能仿真，这是在芯片装载程序前的功能仿真模拟，它仅仅验证逻辑的正确性。在一般的 EDA 解决方案中，这一步一般采用编写测试向量或加激励波形的方法，只能对逻辑的输出信号进行模拟，而对于一些重要的内部信号则无能为力了。在采用 VHDL 后可以借助 HDL Synthesis 生成的 VHDL 格式的内部网单使用一些特殊的调试器对 VHDL 源程序进行类似于高级语言调试的单步跟踪调试。这样不仅可以观察重要的内部信号，而且可以清楚地看到程序执行的流程。一般还要经过时序模拟（timing simulatior）也就是图 2.1 中的时序分析，在这里可以对电路的工作频率、工作延时做定性的模拟，虽然这也还会跟实际情况有一定的差距，但还是建议开发者要进行这一步模拟。

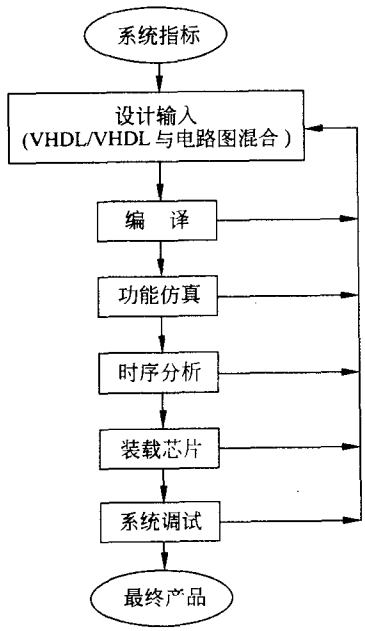


图 2.1 VHDL 的设计流程

对于这些步骤，一般均可借助由芯片生产商提供的开发软件完成。因此一定要在这里将各项功能都调试正确，接下来就是对芯片进行装载程序，在这里要用各个芯片厂商提供的编译器来生成可用于装载（download）的文件进行装载。例如，Altera 公司的软件生成的装载文件是 .POF，用软件菜单中的 Program 命令装载芯片；Lattice 公司的软件生成的装载文件是 .JED，用软件菜单中的 Fitter 命令装载芯片。每个公司的开发工具都具备这些功能。最后是对装载的芯片进行系统调试以验证设计的正确性。如果其中哪一步出现错误，都要重复前面的步骤，直到最终产品调试通过。

2.2　VHDL 程序基本结构

一个 VHDL 程序由 5 个部分组成，包括实体（ENTITY）、结构体（architecture）、配置（coxnfiguration）、包（package）和库（library）。实体和结构体两大部分组成程序设计的最基本单元。图 2.2 表示的是一个 VHDL 程序的基本组成。配置是用来从库中选择所需要的单元来组成该系统设计的不同规格的不同版本，VHDL 和 Verilog HDL 已成为 IEEE 的标准语言，使用 IEEE 提供的版本。包是存放每个设计模块都能共享的设计类型、常数和子程序的集合体。库是用来存放已编译的实体、结构体、包和配置。在设计中可以使用 ASIC 芯片制造商提供的库，也可以使用由用户生成的 IP 库。

下面主要介绍 VHDL 程序的基本组成部分实体和结构体。

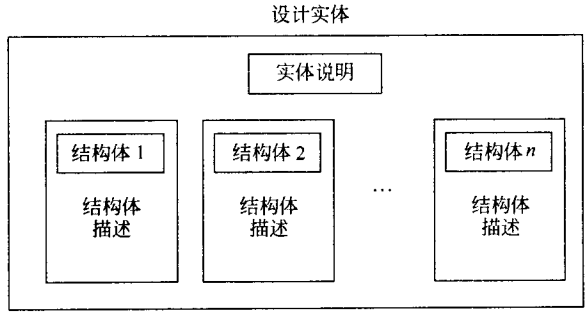


图 2.2　VHDL 程序的基本组成

2.2.1　实体

在用 VHDL 描述数字系统结构时，使用实体（ENTITY）结构。实体是 VHDL 程序设计中最基本的模块，可以单独编译并且可以并入设计库。ENTITY 所描述的是数字系统输入/输出接口，同时还定义一些全局常量以及与其他电路（程序模块或逻辑图模块）之间必要连接的拓扑结构，实体是用来描述一个元件或一个模块与整个系统中其他的元件或模块间的关系。但在 ENTITY 部分并不对电路的逻辑做任何描述，可以看成是一个所谓的逻辑“黑盒子”。很明显，VHDL 遵循 EDA 解决方案中自顶向下的设计原则，并能够保持良好的接口兼容性。

在一个设计中可以同时声明若干组 entity-achitecture，也就是说 entity-achitecture 是 VHDL 描述逻辑时的基本结构。设计的最顶层是顶级实体。如果设计分层次，那么在顶级实体中将包含较低级别的实体。

实体组织的一般格式是：

```
ENTITY 实体名 IS
    [ GENERIC (类属表); ]
    [ PORT(端口表); ]
```

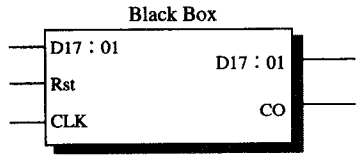
```
实体说明部分;

[BEGIN
    实体语句部分;]

END [ 实体名];
```

大写字母表示的是实体说明的框架，对于每一个实体说明都应该这样写，是不可缺少的。实际上 VHDL 是不区分大小写的。用大写表示是为了使程序结构更清晰而方便阅读。

图 2.3 是一个简单的器件，实体描述如下。




```
Entity black box is
    Port (
            Rst, clk: in      std_logic;
            D:        in      std_logic (7 downto 0);
信号名 ——→  Q:        buffer  std_logic (7 downto 0);
            C0:       out     std_logic
        );
End black box;
```

类型

模式

图 2.3 一个实际器件及其实体描述

从上面实体组织的格式看到，实体说明形象地描述了一个器件的外观视图，也就是从外部看这个器件时包括哪些端口。在实体说明的头部是类属说明和端口说明，用来设计实体和外部环境通信的对象、通信的格式约定和通信通道的大小等。

2.2.1.1 类属

1. 格式

```
GENERIC  [CONSTANT（名字表）]：[IN] 子类型标识 [：=静态表达式]
```

2. 功能

参数的类属设计是为实体和外部环境通信的集体静态信息提供通道。用来规定端口的大小、实体中元件的数目和实体的定时特性等相关的参数。

3. 实例

（1）对于各种译码器电路，如 2-4 译码器、3-8 译码器、4-16 译码器等，其特点是有 N 个输入，就有 2^N 个输出。而这些译码器除了端口数目不同之外，其他方面（如器件的连接）都是一样的，因此用类属进行统一的描述。

```
ENTITY decoder IS
```