山东省教育委员会“九五”立项教材

电子设计自动化

EDA

（修订本）

刘润华　单亦先　主编

石油大学出版社

山东省教育委员会“九五”立项教材

电子设计自动化

（修订本）

主　编　刘润华　单亦先
副主编　任旭虎　成谢锋
范爱平

石油大学出版社

图书在版编目(CIP)数据

电子设计自动化:EDA/刘润华等编. —东营:石油大学出版社,2001.8
山东省教育委员会"九五"立项教材
ISBN 7-5636-1100-2

Ⅰ. 电… Ⅱ. 刘… Ⅲ. 电子电路-计算机辅助设计 Ⅳ. TN702

中国版本图书馆 CIP 数据核字(2001)第 20468 号

电子设计自动化

(修订本)

刘润华 单亦先 主编

责任编辑:宋秀勇(电话 0546—8396155)
封面设计:孟卫东

出 版 者:石油大学出版社(山东 东营,邮编 257061)
网 址:http://sunctr.hdpu.edu.cn/~upcpress
电子信箱:upcpress@mail.hdpu.edu.cn
印 刷 者:青岛经济技术开发区华信包装印刷厂
发 行 者:石油大学出版社(电话 0546—8392139)

开 本:787×960 1/16 印张:18.5 字数:360 千字
版 次:2001 年 9 月第 1 版第 1 次印刷
印 数:1—4000 册
定 价:23.00 元

前 言

半个世纪以来，电子工程设计主要以手工设计和计算机辅助设计为主。随着电子器件和计算机软件的不断发展，逐步形成了电子设计自动化(Electronies Design Automation，简称EDA)。特别是在20世纪末，在系统编程技术的出现，才实现了真正意义上的电子设计自动化。EDA是将来电子设计的必经之路，也是电子技术教学必不可少的内容。

电子设计自动化主要包括两方面的内容，即电子系统仿真和电子系统设计，全部过程均在计算机上完成。EDA的仿真技术，为用户提供了全功能、全频带的分析仪器平台，实现了系统结构或电路特性的模拟以及电路参数的优化；EDA的设计技术，主要是指利用在系统可编程器件，包括在系统可编程模拟器件(ispPAC)和在系统可编程逻辑器件(ispPLD)，借助于开发软件，实现所要求的电子系统。

为了适应新世纪教学改革的需要，我们组织编写了《模拟电子技术》、《数字电子技术》和《电子设计自动化》等系列教材。在《模拟电子技术》中介绍了ispPAC，在《数字电子技术》中介绍了ispPLD等编程器件的硬件结构、原理和特性。本书为《电子设计自动化》，主要介绍了EDA设计方法、设计语言和开发软件以及设计实例。本书是在原山东省教育委员会“九五”立项教材《现代电子系统设计》(获省级教学成果奖)的基础上，经过删简、补充、修改后完成的，使得本书更加完善，更加有利于提高大学生的创新能力。

参加本书编写的有，石油大学的任旭虎(第1章)、单亦先(第2章)、刘润华(第3章)，山东大学的范爱平(第4章)，济南大学的成谢锋(第5章)，刘润华教授为本书主编，负责本书的策划、组织和定稿。

在本书的编写过程中，于云华、王心刚、董传岱、李震梅、刘立山等许多老师和研究生李波都做了大量工作。在此一并表示感谢。

本书可作为电类本科和研究生EDA课程的教材，也可作为电子技术课程设计的教材，还可与《模拟电子技术》与《数字电子技术》配套使用，作为电子电路仿真、ispPAC、ispPLD开发的上机教材或自学教材。

由于编者水平有限，加之时间仓促，书中错误和不足之处在所难免，恳请读者批评指正。

编　者

2001年7月

目 录

第1章 可编程逻辑设计语言

1.1 硬件描述语言 HDL 简介

1.1.1 为什么用硬件描述语言 HDL

HDL(Hardware Description Language)是随着集成电路系统化和高集成化的逐步发展而发展起来的,是一种用于数字系统设计和测试方法的描述语言。对于小规模的数字集成电路,我们通常可以用传统的设计方法,例如卡诺图、真值表、逻辑方程等方法来实现。但这种设计方法费时、费力,易出差错,因此在大规模数字系统设计中是不可取的。逻辑电路图和逻辑方程是传统的硬件描述方法,但随着系统复杂程度的增加,这两种描述变得过于复杂,不便于使用。同时在高于逻辑级的抽象层次上,这两种方法很难用简练的方式提供精确的描述,而且在自顶向下的设计方法中不能再把它当做通常的描述手段。而硬件描述语言(HDL)则逐渐成为满足以上要求的新方法。同时由于计算机和高速通讯设备的飞速发展,而对集成电路提出高集成度、系统化、微尺寸、微功耗的要求,因此,高密度逻辑器件和 HDL 便应运而生。

HDL 与高层次的软件程序设计语言类似,同时又具有以下不可比拟的优点:

(1) 在抽象层次上,可以对设计进行精确而简练的描述。

(2) 在不同层次上易于形成用于模拟和验证的设计描述。

(3) 在自动设计系统中(例如高层次综合工具)作为设计输入。

(4) 可以实现硬件和软件的联合设计,消除硬件和软件开发时间的间隔。

(5) 易于设计的修改,易于把修改的设计并入设计系统。

(6) 易于产生用户手册、服务手册等文件。

和通常的高级软件程序设计语言不同,HDL 的主要目的是用来编写设计文件并建立硬件器件的模拟模型。硬件系统的基本性质和硬件的设计方法决定了 HDL 的主要特征,HDL 的语言和语法的定义是为了能描述硬件的行为,它能自然描述硬件中进行的并行和非递归特性及时间关系。

HDL 在硬件设计领域的作用与 C 和 C++在软件设计领域的作用类似。在

软件设计领域，C 和 C++等高级语言早已取代低级汇编语言，同样的情况正在硬件设计领域发生，高级硬件设计语言(HDL)将逐步替代低级硬件描述方法，如逻辑状态表和逻辑电路图等。

1.1.2 各种 HDL 的浏览

自从 Iverson 于 1962 年提出 HDL 以来，已经出现许多 HDL。其中绝大多数是专有产品，包括 Silvar-lisco 公司的 HHDL(Hierarchial HDL)、Zycad 公司的 ISP、Gateway Design Automation 公司的 Verilog 以及 Mentor Graphics 公司的 BLM。高等学校和科研单位也有上百种 HDL，如 ABLE-HDL、MIMOLA 以及 SCHOLAR 等。此外，一些大型计算机制造商也有其内部使用的设计语言，例如 TIHDL 就是德克萨斯公司的硬件描述语言，在该公司内部及其客户中采用 TI-HDL 来描述设计。

某些 HDL 是从已有的软件程序设计语言发展起来的，例如：BLM、MIMOLA 和 SCHOLAR 是由 PASCAL 发展而来，而 Silicon Compiler 公司的 M 以及 Gateway 公司的 Verilog 则以 C 语言为基础。英国皇家信号和雷达研究所 1979 年为英国国防部开发了 ELLA，在 VHDL 出现之前广泛应用于欧洲，而 UDL/I 在日本以标准 HDL 的形式出现。多年以来，设计者一直使用这些专用的 HDL 来描述他们的军用或民用芯片设计。

美国国防部的项目有众多的承包人，他们使用过多的设计语言，使承包人甲的设计不能被承包人乙再次利用，这就造成了信息交换困难和设计维护困难。为了解决这个问题，美国国防部为他们的超高速集成电路计划(Very High Speed Integrated Circuit)提出了硬件描述语言 VHDL(VHSIC Hardware Description Language)，这个任务交给了德克萨斯仪器公司、IBM 公司和 Intermetrics 公司。1987 年 12 月，IEEE 接受 VHDL[Leib89]为标准 HDL，这就是今天我们所了解的 IEEE Std 1076～1987[LRM87]。此后又作了若干修改，增加了一些功能，新的标准版本记作 IEEE Std 1076～1993[LRM93]。严格地说，VHDL'93 和 VHDL'87 并不完全兼容(例如，增加了一些保留字并删去了某些属性)，但是，对 VHDL'87 的源码只作少许简单的修改就可以成为合法的 VHDL'93 代码。

在以下两节中，将详细介绍两种硬件描述语言：国内科研单位和高校使用广泛的硬件描述语言 ABEL-HDL 和具有良好应用前景的 IEEE 标准硬件描述语言 VHDL。

1.2　ABEL-HDL 硬件描述语言

ABEL-HDL 是美国 DATA I/O 公司开发的一种高级编译型可编程逻辑设计语言，它的特点是器件覆盖率高，几乎支持所有的 PLD 器件开发，用它不仅能完成设计输入，还提供了诸如语法规则检查、器件选择、逻辑优化、逻辑仿真、输出标准格式数据文件等处理功能。目前，ABEL 语言已经成为 PLD 设计者广泛使用的辅助设计软件之一。

1.2.1　ABEL-HDL 的基本元素

在用 ABEL-HDL 进行逻辑设计时，描述逻辑功能的源文件必须是符合 ABEL-HDL 语言语法规范的 ASCII 文本。

ABEL-HDL 源文件是由符合 ABEL-HDL 语言的基本符号构成的，这些符号必须满足一定的书写规则才能正确地描述逻辑功能。下面讨论 ABEL-HDL 语言的基本元素。

1. 合法的 ASCII 符

ABEL-HDL 源文件中可使用的合法 ASCII 符包括：

a～z(小写字母)　A～Z(大写字母)　0～9(数字)

!　@　#　$　?　|　&　*　(　)　_　=　+　[]　{}

;　:　'　"　～　\　|　,　<　>　/　.　^　%

2. 标识符

标识符用来标识器件、器件管脚或节点、集合、输入/输出信号、常量、宏及变量。所有这些标识符都必须遵循下述规则：

(1) 标识符最多 31 个字符长，必须以字母或下划线开头。

(2) 除第一个字符外，标识符可由大小写字母、数字、波浪线(～)和下划线(_)组成。

(3) 标识符不能使用空格，单词间分隔需用下划线。

(4) 标识符中不能使用句点，除非是一个合法的点后缀名。

(5) 标识符的命名不能与 ABEL 语言的关键字相同。

(6) 除保留标识符(关键字)外，标识符中同一字母的大小写表示不同的含义。例如，output 与 Output 为不同的标识符。

3. 关键字

关键字是一种特殊的标识符，被 ABEL 视为保留字，不能用来给器件、管脚、节点、常量、集合、宏定义及信号命名。在源文件中使用关键字时，仅表明这个

字所表示的某种功能。一旦关键字被用于错误场合，语言处理程序将标出错误。

硬件描述语言中的关键字不区分大小写，可以用大写、小写或混合字体输入，表1.2.1按字母顺序列出了 ABEL-HDL 的关键字。

表 1.2.1 ABEL-HDL 的关键字

关键字	说明	关键字	说明
1. Async _ reset	异步复位状态描述语句	16. Node	节点定义语句
2. Case _ Endcase	条件选择语句	17. Options	控制选项定义语句
3. Declarations	定义段语句	18. Pin	引脚定义语句
4. Device	器件定义语句	19. Property	特征定义语句
5. Equations	逻辑方程语句	20. State	状态描述定义语句
6. End	结束语句	21. State _ diagram	状态图语句
7. Functional _ block	功能模块定义语句	22. State _ register	状态寄存器说明语句
8. Fuses	熔丝状态定义语句	23. Sync _ reset	同步复位状态描述语句
9. Goto	无条件转移语句	24. Test _ vectors	测试语句
10. If _ Then _ Else	条件转移语句	25. Title	标题语句
11. Interface	功能模块接口定义语句	26. Trace	跟踪选项语句
12. Istype	属性定义语句	27. Truth _ table	真值表表头语句
13. Library	库引用语句	28. When _ Then _ Else	条件转移语句
14. Macro	宏定义语句	29. With _ Endwith	转移方程语句
15. Module	模块语句		

4. 常量与数值

(1) 常量、定值。常量、定值可用于 ABEL-HDL 的逻辑设计。常量值可用在赋值语句、真值表与测试向量中，有时还可赋给标识符以表示该标识符在整个模块中都有定值。常量值可为数值或非数值型特殊常量，在 ABEL-HDL 中的特殊常量如表1.2.2所示。

输入时在字母两边加圆点以表示其为特殊常量，特殊常量可用大写或小写字母。实际编写源文件时，为书写方便，常常用常量定义语句将 L、H 定义为 0 和 1，并将上述符号中的.C. 和.X. 等定义为 C 和 X。

(2) 数值。ABEL-HDL 中所有数值运算的精度都是 128 位，因此，有效数值范围是 0～$(2^{128}-1)$。数值可用五种形式中的任一种表示，其中四种是用不同的数制来表示数值，第五种是用字母符号表示数值。

当选用四种数制中的一种来表示数据时，须在该数据前标明所用数制的符号。表1.2.3列出了 ABEL-HDL 支持的四种数制及它们相应的符号，数制符号可用大写或小写输入。

表 1.2.2　ABEL-HDL 中的特殊常量

常量值	说　明
.C.	时钟输入(电平按低—高—低变化)
.K.	时钟输入(电平按高—低—高变化)
.U.	时钟上升沿
.D.	时钟下降沿
.F.	浮点输入或输出信号
.P.	寄存器预置
.X.	无关项
.Z.	三态高阻

表 1.2.3　ABEL-HDL 中的常用数制

基数名称	基数	符号
二进制	2	^ b
八进制	8	^ o
十进制	10	^ d
十六进制	16	^ h

若一个指定的数据前没有数制符号，则认为它使用默认数制，正常的默认数制为十进制。此外，数也可以用字符表示，即以字母的 ASCII 码作为数值。例如，'a'=97,'b'=^ h62 ,'abc'=^ h616263=6382203。

5. 字符串和注释

(1) 字符串。字符串是用单引号括起的一串 ASCII 码字符。字符串常用于模块语句、标题语句、控制选项语句和引脚、节点、属性的定义中。若要在字符串中使用单引号，要在它前面加一个反斜线“ \ ”，以便同字符串标志相区别。

(2) 注释。注释是使源文件易读的另一种方式。它常被用来说明源文件不易理解之处，注释不影响源文件的内容，也不参与编译。可以用两种方式插入注释：注释以双引号(")开始，以另一双引号或行结束标志结束，另外，注释不能用于关键字之间。

例如，Module Basic _ Logic; "gives the module name"

或　　Module Basic _ Logic; "gives the module name

处理程序是不对双引号后面的语句编译的，所以，利用双引号也可令文件中的某些行不执行，从而达到删去某行的目的，这在调试程序时，是经常采用的方法。

6. 运算符、表达式和等式

(1) 运算符。ABEL-HDL 表达式中可包含常量、信号名等对象。表达式将这些对象进行组合、比较或进行逻辑运算，以产生输出信号。这些运算(如加法、逻辑与等)在表达式中用运算符来表示。ABEL-HDL 的运算符可分为四种基本类型：逻辑运算符、算术运算符、关系运算符与赋值运算符，各种运算符如表1.2.4所示。

表 1.2.4 ABEL-HDL 的运算符

运算	运算符	示例	含义	运算优先级
逻辑运算	!	!A	对 A 逐位求反	1
	&	A&B	A 和 B 逐位相与	2
	#	A#B	A 和 B 逐位相或	3
	$	A$B	A 和 B 逐位异或	3
	!$	A!$B	A 和 B 逐位异或非	3
算术运算	-	-A	A 取负(以补码表示)	1
	-	A-B	A 减 B	3
	+	A+B	A 加 B	3
	*	A*B	A 乘 B	2
	/	A/B	A 除以 B(无符号整除)	2
	%	A%B	求模(求 A/B 的余数)	2
	<<	A<<B	A 左移 B 位	2
	>>	A>>B	A 右移 B 位	2
关系运算	==	A==B	如果 A 与 B 相等取值为真	3
	!=	A!=B	如果 A 不等于 B 取值为真	4
	>	A>B	如果 A 大于 B 取值为真	4
	<	A<B	如果 A 小于 B 取值为真	4
	>=	A>=B	如果 A 大于等于 B 取值为真	4
	<=	A<=B	如果 A 小于等于 B 取值为真	4
赋值运算	=	A=B	不带时钟的赋值(组合输出)	
	:=	A:=B	带时钟的赋值(寄存器输出)	
	? =	A? =B	组合型赋值(任意态)	
	?:=	A?:B	寄存器型赋值(任意态)	

① 逻辑运算符：逻辑运算符用于布尔方程。ABEL-HDL 使用的许多逻辑设计均采用标准逻辑运算符，多位逻辑运算是逐位进行的。如：4#6=6，即 100#110=110。

② 算术运算符：在 ABEL-HDL 语言中，算术运算符定义了表达式中两个对象之间的算术关系。移位运算符也属于算术运算符，因为左移一位就相当于乘以 2，右移一位就相当于除以 2。

注意：减号的用法不同时意义也不同，当它用在一个运算数前时，表示对这个运算数取二进制补码，当用于两个运算数之间时，表示将第二个运算数的二进

制补码与第一个运算数相加。除法是无符号整数除法，移位运算是逻辑无符号移位，移位运算中用零来补缺空位。

③ 关系运算符：关系运算符用来比较表达式中的两项。由关系运算符构成的表达式，输出一个布尔真值或假值。所有关系运算都是无符号运算。例如，表达式 !0>4 为真，因为 0 的二进制反码是1111，即无符号二进制数 15，而 15 是大于 4 的。本例中假定数的精度为 4 位二进制，实际上 ABE-HDL 使用的为 128 位二进制数，因此 0 的二进制反码应是 128 位均为 1 的二进制数。

逻辑真或假在 ABEL-HDL 内部用数值来表示。逻辑真表示为－1，取二进制补码后，相当于 128 位都为 1 的数。逻辑假为 0，相当于 128 位都为 0 的数。这就是说，一个结果取真或假的关系表达式可用在任何能使用数值或数值表达式的地方，其取值是－1 或 0。

使用运算符时，要考虑运算符的优先级关系，用括号可以改变表达式的运算顺序。每个运算符有相应的优先级，优先级最高为 1，最低为 4。

若同一表达式中有优先级相同的运算符，则按从左到右的顺序进行运算，圆括号可改变运算次序，圆括号内运算优先进行。

④ 赋值运算符：赋值运算符是一类用于布尔方程的特殊运算符，它是将表达式的运算结果通过等式赋给输出信号。有四种赋值运算符(两种组合型和两种寄存器型)。组合型赋值(＝或?＝)在表达式求值后立即赋予输出，没有延时。而寄存器赋值(:＝或?:＝)要等与输出有关的时钟脉冲到来后，才进行赋值。

注意：赋值运算符 :＝ 和 ?:＝ 只能用在引脚到引脚的寄存器方程中，在用点后缀的寄存器方程中要使用赋值运算符＝和? ＝。

(2) 表达式。表达式是标识符和运算符的组合，在表达式中，可以出现任何逻辑、算术以及关系运算，同时，表达式中的运算顺序由运算优先级别决定。当具有相同优先级别的多个运算出现在表达式中时，按从左到右的顺序运算，而当有括号内包含的运算时，最内层括号中的运算具有最高的优先级，其余以此类推。

(3) 等式。等式用于将表达式运算的结果赋于一个信号或一组信号。在 ABEL 语言中，一个信号可以被多次赋值，而其最终的输出结果是所有这些赋值的或，并不是最后一次的赋值，这是 ABEL 语言中等式的特殊特性。

7. 集合

在表达式和方程中可以使用集合。集合是作为一个独立单元进行操作的一组信号或常量，它用一个标识符来表示，其中的每一个信号或常量称为集合的元素。集合可以进行表1.2.4中除 * 、/、%、>>和<<以外的任何一种运算。集合将一组信号用一个名字来表示，从而简化了 ABEL 的逻辑设计和测试向量的描述。

(1) 集合的定义。定义一个集合的方法有枚举法、界限符法和两种方法的组合。例如下面三种方法均定义了一个具有 8 个元素(A7,A6,A5,A4,A3,A2,A1,A0)的集合,其集合名为 ADD 和 Select。

ADD=[A7,A6,A5,A4,A3,A2,A1,A0] 为枚举法

Select= [A7..A0] 为界限符法

ADD=[A7,A6,A5..A0] 为两种方法的组合

(2) 集合的赋值。可用数值或数值集合对集合赋值。如:

ADD=[1,0,1]; 或 ADD=5

用于集合赋值或比较的数要转换成二进制形式,并遵从下列规则:

① 如果该二进制数的有效位数多于集合中的元素个数,要从左边截去多余的位。

② 如果该二进制数的有效位数少于集合中的元素个数,要从左边用 0 补齐缺少的位。

例如,下面两式的赋值是等效的:

[A,B]=^b01 ; [A,B]=^b1

再如[A0,A1,A2]=3,则等效于 A0=0,A1=1,A2=1。

(3) 集合的运算。任何对集合的运算其实是对集合中的每个元素进行的。A-BEL-HDL 允许利用集合的序号访问集合中的元素。例如,SET=[A7,A6,A5,A4,A3,A2,A1,A0],则 SET[3]表示集合 SET 中的序号为 3 的元素,即 A4。

注意:序号 0 是集合中最左面的元素(与元素的排列顺序有关,而不是与元素的后缀或下标有关)。

除了少数运算符外,大多数运算符都适用于集合,集合按照布尔代数规则进行运算,参与运算的集合元素必须相等,如表达式"[A,B]+[A,B,C]"是非法的。

表1.2.5给出集合的运算规则,表中大写字母表示集合名称,小写字母表示集合中的元素,字母 k 和 n 为元素和集合的下标。集合的下标指出该集合含有多少元素,所以 Ak 指出集合 A 包含 k 个元素,al 表示集合 A 的第一个元素,ak-1表示集合 A 的第k-1个元素。

集合运算的方式与结果取决于其变量的类型。绝大多数运算符对一个集合的运算结果仍为一个集合。需注意的是关系运算符(==、!=、>、>=、<、<=)的结果为 TRUE(全 1)或 FALSE(全 0),且该数值的位数根据需要来截取。其长度由关系运算符所处场合的上下关系决定,不取决于变量的长度。