HZ BOOKS

电子与电气工程丛书

PRENTICE HALL PTR

VHDL教程

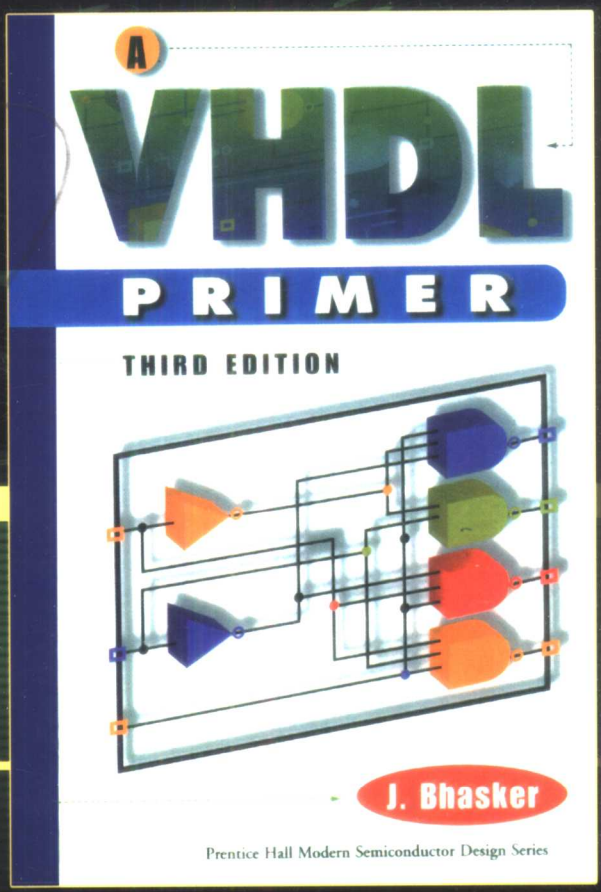
（原书第3版）

A VHDL Primer

(Third Edition)

（美）Jayaram. Bhasker 著

刘雷波　孟一聪　译



机械工业出版社
China Machine Press

电子与电气工程丛书

VHDL教程

（原书第3版）

A VHDL Primer

(Third Edition)

（美）Jayaram. Bhasker 著

刘雷波　孟一聪　译

机械工业出版社
China Machine Press

本书从VHDL语言的功能特性出发，介绍了VHDL语言的组成元素、描述风格、建模特征、测试平台的设计技巧等，并详细给出了一些经过作者验证的实例。本书的目的在于向广大的电子设计人员介绍VHDL语言的基本知识和使用它来设计数字系统硬件电路的方法，从而使设计者摆脱传统的人工设计方法的约束，使数字系统的设计水平上升到一个新的阶段。

本书适合作为计算机科学及其相关专业的教材或参考书，也可供工程技术人员参考。

本书版权登记号：图字：01-2005-1169

图书在版编目（CIP）数据

VHDL教程（原书第3版）/（美）巴斯克尔（Bhasker, J.）著；刘雷波，孟一聪译. －北京：机械工业出版社，2006.4

（电子与电气工程丛书）

书名原文：A VHDL Primer, Third Edition

ISBN 7-111-18524-2

Ⅰ. V…　Ⅱ. ① 巴…　② 刘…　③ 孟…　Ⅲ. 硬件描述语言，VHDL－程序设计　Ⅳ. TP312

中国版本图书馆CIP数据核字（2006）第012885号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：华　章

北京慧美印刷有限公司印刷 · 新华书店北京发行所发行

2006年4月第1版第1次印刷

787mm × 1092mm 1/16 · 15.25印张

定价：29.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译 者 序

VHDL（VHSIC Hardware Description Language）是一个详细且庞大的硬件描述语言。它在20世纪80年代后期由美国国防部开发，并成功地应用在软硬件系统的仿真、验证和设计综合等领域，对于设计自动化的发展起到了极大的促进和推动作用。

本书从VHDL语言的功能特性出发，介绍了VHDL语言的组成元素、描述风格、建模特征、测试平台的设计技巧等，并详细地给出了一些经过作者验证的实例来帮助读者更好地学习。本书最大的特点是：读者除了最好能够具备一些高级编程语言（如C、Pascal等）的知识外，不需具备VHDL语言的任何先验知识；读者对象主要是对VHDL语言感兴趣的软件和硬件设计师，而对软件和硬件方面没有特别的侧重和要求。本书的目的在于向广大的电子设计人员介绍VHDL语言的基本知识和使用它来设计数字系统硬件电路的方法，从而使设计者摆脱传统的人工设计方法的约束，使数字系统的设计水平上升到一个新的阶段。

本书由清华大学的刘雷波和香港科技大学的孟一聪共同翻译。参加翻译工作的还有清华大学微电子学研究所的史弋宇、张楠和赵晖等，对他们的辛勤工作表示感谢。同时也非常感谢机械工业出版社华章分社在组织出版和编辑工作中所给予的支持。

限于译者水平有限，中译本中难免有错误与不妥之外，恳请读者批评指正。

刘雷波

2005年于清华园青年公寓

译者简介

刘雷波于1999年毕业于清华大学电子工程系，获无线电技术与信息系统专业学士学位，2004年毕业于清华大学微电子学研究所，获得电子科学与技术专业博士学位。2004年至今在清华大学任教。讲授“VISI数字信号处理系统——设计与实现”和“数字集成电路分析与设计”课程。研究方向主要包括：图像编码理论、集成电路设计、数字信号处理等。已经发表论文10余篇，申请发明专利3项。

孟一聪于2002毕业于北京邮电大学电子工程系，获得电子工程专业学士学位，2005年毕业于清华大学电子工程系，获电路与系统专业硕士学位，现为香港科技大学电子工程专业的博士研究生。个人的研究方向包括：低功耗专用集成电路设计、生物电子等。

第3版前言

IEEE的STD_LOGIC_1164标准集合文件程序包自最新版本发布以来一直非常流行。本书的第3版使用该程序包替换原来的ATT_MVL程序包。附录E提供了IEEE STD_LOGIC_1164程序包的函数列表。

本书的第3版增加了许多实例以说明VHDL语言的一些其他实用功能。此外，本版还加入了用以描述I/O功能和测试平台的新章节，讨论了子程序的默认参数及生成语句的绑定实例等问题。附录F列出了另一个实用程序包——UTILS_PKG。本书用到了一些定义在UTILS_PKG程序包中的函数，而这些函数在STD_LOGIC_1164程序包中没有提供。

此外，本书对编排格式也做了一定的修改，更便于读者阅读且易于读懂。

J. Bhasker
1998年9月

第2版前言

正是因为具备综合能力才促使VHDL成为一门流行的硬件设计语言。随着使用综合工具进行硬件设计的人数的增加，使用VHDL语言的人也越来越多。时至今日，VHDL已经成为描述数字设计的IEEE标准和ANSI标准。

根据IEEE的规则，一个语言如果想继续成为标准就必须通过IEEE每5年一次的投票表决。VHDL在1987年首次成为标准语言，当时的标准发行号为IEEE Std 1076-1987。本书的第1版就是根据这个标准写的。1992年，IEEE重新投票，并最终决定继续将VHDL作为标准。1993年，经过仔细研究，IEEE发布了VHDL的最新标准IEEE Std 1076-1993。这个版本中加入了一些新的特征，某些结构体的语法更加一致，并解决了上一版本中很多有歧义的地方。本书的附录对这些修改一一做了综述。

本书的第2版是对IEEE Std 1076-1993中的VHDL语言进行描述。除了非常微小的修改以外，在VHDL语言的较早版本中描述的模型是严格向上兼容的，这些变化列于附录中。本书的第2版主要是在第1版的基础上，加入了一部分对新标准中新增加的特性和语法的说明和描述。第2版沿用了第1版的格式，在这一点上没有变化。第2版中也没有把相对第1版的变换和改进单独罗列出来，但是增加了一些实例并根据读者对第1版的反馈意见，进行相应修改、增加和删减了一部分内容。

致谢

非常感谢Paul Menchini先生，他帮助我审阅了这一版的全部内容，并提出了许多有价值的建议和意见。正是因为他的这些帮助，才使这一版改进的内容更加合理和成熟。

J. Bhasker

1994年4月

第1版前言

VHDL是用来描述数字系统模型的硬件描述语言。在获得了明确的时间信息后，VHDL就能够完整地描述数字系统的行为或结构。该语言支持层次化的系统建模，同时也支持自顶向下（top-down）和自底向上（bottom-up）的设计方法。VHDL可以在从高层结构级到底层门级的范围内描述系统及其子系统。VHDL仿真部分的语法定义非常精确，能够和该语言的其他部分（如实例建立部分等）实现无缝连接。因此，用VHDL设计的系统模型可以用VHDL仿真器来验证。

本书的目的是面向入门级水平的读者，读者可以不具备该语言的任何先验知识。但是，读者最好能够具备一些高级编程语言的知识，如最好了解C语言或者Pascal语言，并且对硬件设计有基本的了解。本书旨在面向对VHDL语言感兴趣的软件设计师和硬件设计师，对软件和硬件领域知识没有特别的侧重。

VHDL是一个庞大、冗余的语言，其中许多复杂结构体的语法定义最初不容易理解（VHDL曾被戏称为Very Hard Description Language，即非常难的描述语言）。但是，快速掌握VHDL中一些简单易用的部分还是可能的。本书的重点就是介绍VHDL中这些简单常用的特征，以便于读者可以很快地应用VHDL编写模型。实际上，这些特征对于完成一个高复杂度的模型设计已经足够了。

本书的初衷并不是想取代IEEE发布的标准VHDL语言参考手册（Standard VHDL Language Reference Manual），只是想采用基于实例的方法来说明和解释VHDL语言中复杂结构体的定义和使用方法，并以此作为IEEE官方标准的一个补充。本书的重点在于提供许多具体设计的实例以进一步解释VHDL中许多结构体的规化和语义。本书通常不提供结构体的完整语法，而只提供那些结构体最常用的用法，书中结构体的语法没有采用VHDL语言参考手册里的正式术语（Backus-Naur形式）编写，而是采用一种“自我解释”的风格编写。本书只是着眼于VHDL中最常用的部分，因此并没有涵盖整个VHDL语言。

本书内容的编排

第1章简单回顾了VHDL语言的发展历史，介绍VHDL的主要功能。第2章提供快速入门指导以演示VHDL的建模特征。后续的章节会对这里提到的内容进行深入讨论。第3章介绍了VHDL语言的重要组成元素，例如类型、子类型、对象和文字（literal）。此外，第3章还介绍了VHDL语言中预定义的操作符。

第4章介绍VHDL用于建模的行为级描述语言的风格。这一章介绍了多种实用的时序电路语句，并且说明了如何将这类语句应用于时序的行为级建模上。这种建模风格和其他高级编程语言的建模风格在语义上非常类似。第5章介绍了数据流建模风格。这一章介绍了并发信号赋值语句以及块语句，同时举例说明如何用这些语句对具有并发行为的设计建模。

第6章介绍了VHDL中结构级建模的设计风格。在这种建模风格里，设计往往被描述为一些连接在一起的有结构层次的元件（component）的集合。这一章还详细地介绍了元件例化语

句。第7章提出了实体－构造体对（entity-architecture pair）的概念，讲解了元件实例与驻留在不同库里的设计相互绑定的方法。这一章还介绍了如何通过类属（generic）将静态信息传递到设计中。

第8章介绍了子程序（subprogram）。子程序可以是函数或者是过程。此外，该章还介绍了子程序和操作符重载等重要内容。第9章介绍了VHDL语言定义的程序包和设计库的环境变量，同时还讲解了在某个库中的设计访问另一个库中的项目的方法。第10章给出诸如实体语句、别名（alias）、保护信号（guarded signal）以及属性（attribute）等VHDL的高级特征。

第11章介绍了仿真VHDL模型的方法学以及编写测试平台（test bench）的技巧。该章还给出了许多生成各种类型的时钟、波形的例子，并且举例说明了如何将这些用于测试设计中。第12章给出了一组易于理解的硬件建模的例子。这些例子包括如何对组合逻辑、同步逻辑建模，如何编写有限状态机等。

本书用粗体来表示VHDL中的保留字。读者可以在附录A里查到完整的保留字列表。书中大多数VHDL结构体没有使用标准VHDL语言参考手册采纳的正式术语来解释，而是以容易让读者理解的单词来解释的。此外，为了说明某个特定的特征，一些VHDL的结构体并没有被完全解释，而是根据需要只解释了其中的一部分。附录B列出了VHDL语言的完整语法。附录C列出了在第11章和第12章中用到的程序包的完整描述。

本书的VHDL结构体中，*楷体字*表示由模型设计者（程序设计员）提供的信息，例如：

entity *实体名称* **is**
　　[**port** (*端口列表*)]…

entity、**is**和**port**都是VHDL的保留字，而*实体名称*和*端口列表*表示由模型设计者提供的信息。方括号[ ]表示可由模型设计者选择填入的信息。一般来说，圆括号( )里表示的是和目前的讨论不相关的内容。本书提供的所有实例都已经用作者本地的VHDL系统验证过。

本书力求用实体来表示所有本书中提到的电路、系统、设计以及其他试图建模的任何东西。

阅读说明

第一次阅读本书的读者最好首先读第2章的入门指导。其他各章的安排是环环相扣的，通常后一章需要前一章的内容和知识作铺垫。因此，读者最好按照顺序阅读。但是，如果读者能够很好地理解第2章的指导，也可以跳过部分章节，直接阅读自己感兴趣的章节。同时，本书还提供完整讨论硬件建模实例的章节，这可以为读者的实际工作提供参考。在本书的末尾为方便读者阅读还提供了完整的参考文献和VHDL的语法的完整列表。读者如果愿意进一步弄清楚VHDL的语法和语义，还可以参考IEEE 发布的标准IEEE VHDL语言参考手册（IEEE Std 1076-1987）。

致谢

非常感谢在撰写本书过程中给予过帮助的同事们。尤其需要感谢Dinesh Bettadapur、Ray Voith、Joel Schoen、Sindhu Xirasagar、Gary Imken、Paul Harper、Oz Levia、Jeff Jones以及Guru Rao等人。他们帮我阅读了初稿，并提出了不少建设性的意见。这些帮助和意见使得本

书进一步得到完善。特别需要感谢Gary Imken先生，谢谢他有足够的耐心来回答我关于VHDL的特殊问题。

当然也要感谢我的夫人——Geetha女士，感谢她帮我审阅了本书的初稿，也感谢她为本书整个出版过程提供了正确的引导。

J. Bhasker

1991年10月

目　录

第1章 概 述

本章简单地回顾了VHDL的发展历史，并明确地指出了VHDL不同于其他硬件描述语言的功能。此外，本章还解释了VHDL中实体的概念。

1.1 什么是VHDL

VHDL是VHSIC Hardware Description Language（VHSIC硬件描述语言）的首字母缩写，而VHSIC是Very High Speed Integrated Circuits（超高速集成电路）的首字母缩写。VHDL是一个可以在不同抽象层次上对数字系统进行建模的硬件描述语言，其范围从算法层次到门电路层次。这里提到的数字系统的复杂程度各不相同，可以是一个简单的门电路，也可以是一个完整的数字电子系统，或者是介于二者之间的任何一种系统。数字系统也可以用层次化的方式描述。定时系统也可以按照同样的描述方式来显式地建模。 1

VHDL语言可以看作是下面几种语言的集成：

sequential language +

concurrent language +

net-list language +

timing specifications +

waveform generation language => VHDL

因此，这种语言具有的结构能够描述带定时或不带定时的数字系统的并发或时序行为。它也允许通过建立多个元件之间的连接来对系统建模。测试波形也可以采用相同的结构来产生。以上所有的结构可以结合起来，从而在一个单独的模型中提供一个全面的系统描述。

VHDL不仅定义了语法，而且对每一种语言结构都定义了非常明确的仿真语义。所以这种语言中的模型可以采用VHDL仿真器来验证。VHDL是一种强类型、冗余的语言。它继承了Ada编程语言的许多特点，尤其是Ada的时序语言部分。因为VHDL提供了广泛的建模功能，致使其常常难以理解。幸运的是，可以快速地掌握VHDL的一个核心子集，而不用学习VHDL中其他更加复杂的特性。这个子集对于大多数实际问题的建模是足够的。当然，完整的VHDL语言具有强大的功能，足以描述最复杂的芯片到完整的电子系统。

1.2 历史

对这种语言的需求是1981年在VHSIC项目中第一次产生的。在该项目中，许多美国公司已经在为国防部（Department of Defense，DoD）设计VHSIC芯片。当时大多数公司都使用不同的硬件描述语言来描述并设计他们的集成电路。因此，不同的供应商之间不能够有效地交换他们的设计。同样，不同的供应商提供给DoD的是采用不同硬件描述语言描述的芯片。芯片的再采购和复用也成为一大问题。因此，数字系统的设计、文档编制和验证都需要一种标准化的硬件描述语言。 2

1983年，为了开发一种语言版本，DoD与IBM、德州仪器（Texas Instruments）和Intermetrics这三家公司首次签定了合同。在1985年，制定和发布了VHDL 7.2版本。VHDL语

言的发展过程中，工业界的参与性很强，特别是那些正在开发VHSIC芯片的公司。在7.2版本发布之后，渴望使这种语言成为工业标准的需求越来越强烈。接着，1986年，这种语言被送到IEEE进行标准化。经过工业界、大学和DoD代表的共同努力，对语言作了重大增强，并于1987年12月正式成为IEEE标准。这个版本就是著名的IEEE Std 1076-1987。VHDL的官方描述出现在《IEEE标准VHDL语言参考手册（IEEE Standard VHDL Language Reference Manual）》中，可以从IEEE获得。这种语言也被美国国家标准协会（ANSI）认可。

根据IEEE的规则，IEEE标准每五年就必须重新投票以决定其是否继续保留为标准。后来的VHDL语言增加了许多新的特征，许多结构的语法变得更加统一，1987版本中的许多歧义都得以解决。新的版本为IEEE Std 1076-1993。本书中描述的VHDL语言就是基于这个标准的。对1987版本所作的主要修改在附录D中介绍。

从1988年9月起，DoD要求它的所有数字专用集成电路（Application-Specific Integrated Circuit，ASIC）提供者采用VHDL在行为级和结构级描述ASIC和它们的子元件。同时，也必须用VHDL语言来描述各个层次的验证ASIC芯片的测试平台。这一系列的政府需求在
3 “Military Standard 454（军事标准454）”中描述。

1987年以来，为了帮助增强模型的互操作性，对VHDL标准化程序包的需求越来越急迫。这是因为不同的CAE（computer-aided engineering，计算机辅助工程）供应商系统支持不同的程序包，会引起严重的模型互操作性问题。例如，使用的逻辑值就有46值逻辑、7值逻辑和4值逻辑等。后来成立了一个委员会来标准化这样一个程序包。这个委员会的成果是建立了9值逻辑程序包。这个程序包被称为STD_LOGIC_1164，后来经投票批准成为IEEE标准“IEEE Std 1164-1993”。本书中的一些例子就使用了这个程序包。

1.3 功能

下面是VHDL语言提供的与其他硬件描述语言不同的主要功能和特征。

- VHDL语言是芯片供应商和CAD工具用户的交流媒介。不同的芯片供应商可以提供给系统设计师各种元件的VHDL描述。CAD工具用户可以用VHDL语言获得在功能仿真的高层抽象的设计行为。
- VHDL语言可以作为不同CAD工具和CAE工具的通信媒介。例如，可以用一种图形捕捉程序来产生设计的VHDL描述，从而用作仿真程序的输入。
- VHDL语言支持层次化设计。也就是说数字系统可以建模为一系列相互连接的元件，每一个元件又可以建模为一系列相互连接的子元件。
- VHDL支持灵活的设计方法。自顶向下、自底向上或混合型。
- VHDL与特定工艺无关，但是支持特定工艺的特征。同时它还支持多种硬件工艺。例如，
4 可以定义新的逻辑类型和新的元件，也可以定义针对特定工艺的属性。由于其独立于工艺的特点，相同的模型可以综合成不同的供应商的库。
- VHDL支持同步和异步时序模型。
- 不同的数字模型技术，比如有限状态机描述、算法描述和布尔等式都可以用VHDL语言建模。
- VHDL是一种公用的、人类可读的和机器可读的语言，并且没有专利保护。
- VHDL语言是IEEE和ANSI的标准，所以采用这种语言描述的模型是可移植的。政府部

门对于维持这种语言作为标准也有很大的兴趣。因此想获得VHDL代码并且对代码进行二次开发就变得非常容易。

- VHDL语言支持三种不同的基本描述风格：结构、数据流和行为。同一个设计可以采用上面三种描述风格的任意组合。
- VHDL语言支持从抽象行为描述到非常细致的门级描述的多种抽象层次。然而，它不支持晶体管级以及晶体管级以下的模型描述。它允许设计采用单一的语言在多个层次上进行混合设计。
- VHDL语言支持任意大的设计建模，语言本身对设计规模没有限制。
- VHDL语言的元素，例如元件、函数、过程和程序包使得大规模设计建模变得更加容易。
- 可以使用相同的语言来写测试平台以对其他VHDL模型进行测试。
- VHDL语言中描述了正常的传输延迟、最小－最大延迟、建立和保持时间、时序约束和尖峰脉冲检测等。
- 模型中类属和属性的使用使得静态信息比如时序和布局信息的反向注释成为可能。
- 类属和属性对于描述参数化设计也非常有用。
- 模型不仅能描述设计功能，也能包括和用户定义属性相关的设计本身的信息，例如总的面积和速度。
- 一种普通的语言可以描述不同供应商的库元件。由于VHDL语言已经成为一种标准，能够解释VHDL的工具对多个供应商提供的模型的读取不存在任何困难。
- 对VHDL语言描述的模型可以采用仿真的方法进行验证，因为每个语言结构都定义了精确的仿真语义。
- 与某种综合描述风格一致的行为模型能够综合成门级描述。
- 定义新数据类型的能力使VHDL能在高的抽象层次上描述和仿真一个新设计，而在这个过程中，不用关心具体的实现细节。

5

1.4 硬件抽象

VHDL可以描述数字硬件器件模型。这种模型指定器件的外部视图以及一个或多个内部视图。器件的内部视图指定其功能或结构，而外部视图指定器件与其他模型的通信接口。图1-1说明了硬件器件和相应的软件模型。

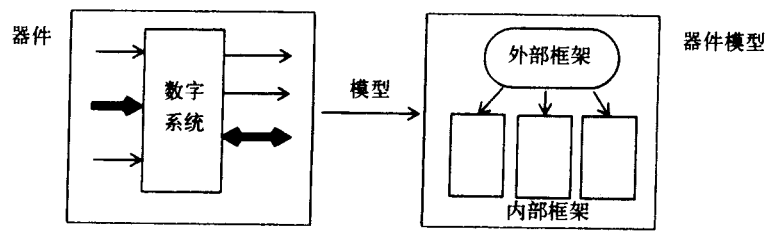


图1-1　器件与器件模型

器件到器件的模型映射是严格的一对多的。也就是说，硬件器件可以有多个器件模型。例如，高层抽象建模的器件可能没有把时钟信号作为它的输入，因为时钟信号可能并没有在描述中使用。同样，接口的数据传输也可以是整型数据，而不是逻辑值。VHDL中每一个器

6

件模型都被看作是一个唯一器件的独特描述，本书中称为实体。图1-2说明有多个器件模型的硬件器件的VHDL 框架结构，每个器件模型代表一个实体。即从VHDL的角度看，实体1到实体*N*代表*N*个不同的实体，而实际上它们都代表同一个硬件器件。

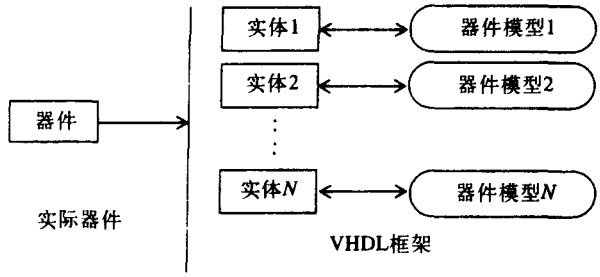


图1-2 器件的VHDL 框架结构

实体是实际器件的硬件抽象。每个实体采用一个模型描述，包括外部框架结构和一个或多个内部框架结构。同时，一个器件可以用一个或多个实体来描述。

第2章 教　程

本章是VHDL语言的快速入门，描述VHDL语言模型的主要特点。学完本章将能够写出简单的VHDL模型。

2.1 基本术语

VHDL是一种用来对数字系统建模的硬件描述语言。简单的数字系统可以只有一个逻辑门，复杂的可以是一个完整的电子系统。这里将数字系统的硬件抽象称为实体。当实体X在另外一个实体Y中使用时，实体X成为实体Y的一个元件。所以元件也是实体，这主要取决于其建模的层次。

为了描述实体，VHDL提供了以下五种不同类型的结构，称为设计单元：

1. 实体声明
2. 结构体
3. 配置声明
4. 程序包声明
5. 程序包体（内容）

8~9

实体由实体声明和至少一个结构体组成。实体声明描述实体的外部视图，例如：输入和输出信号名称。结构体包括实体的内部描述，例如：一系列相互连接的代表实体结构的元件，或是一系列代表实体行为的并行或时序语句。每一种类型的描述可以在不同的结构体中定义，也可以混合在一个结构体中定义。图2-1表示了一个实体和一种可能的模型。

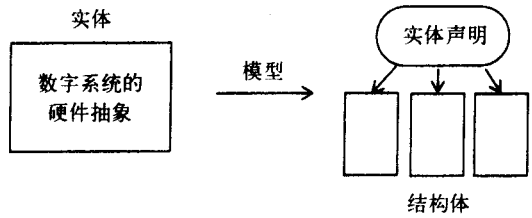


图2-1 实体及其模型

配置声明用来生成实体配置。从跟实体有关的结构体中选出一个结构体，与实体进行绑定。同时它也能将所选结构体中的元件与其他实体进行绑定。一个实体可以有多个不同的配置。

程序包声明集成了一系列相关的声明，包括类型声明、子类型声明和子程序声明，这些声明可以被两个或多个设计单元共享。一个程序包体包括程序包声明中声明的子程序的定义。

图2-2中有三个实体E1、E2和E3，实体E1有三个结构体E1_A1，E1_A2和E1_A3。结构体E1_A1是一个没有任何层次的纯行为模型。结构体E1_A2中有元件BX，而结构体E1_A3中有元件CX。实体E2有两个结构体E2_A1和E2_A2。其中，结构体E2_A1中有元件M1。实体E3有三个结构体E3_A1，E3_A2和E3_A3。注意每个实体只有一个单独的实体声明，但是可以有多

10

个结构体。

虚线代表在实体E1的配置中定义的绑定。图中有两种类型的绑定：结构体到实体的绑定和结构体中使用的元件到其他实体的绑定。例如，结构体E1_A3绑定到实体E1，结构体E2_A1绑定到实体E2。结构体E2_A1的元件M1绑定到实体E3。结构体E1_A3的元件CX绑定到实体E2。但是，也可以为实体E1选择另外一种具有以下绑定的不同配置：

√ 结构体E1_A2与其实体E1绑定；

√ 元件BX与实体E3绑定；

√ 结构体E3_A1与其实体E3绑定。

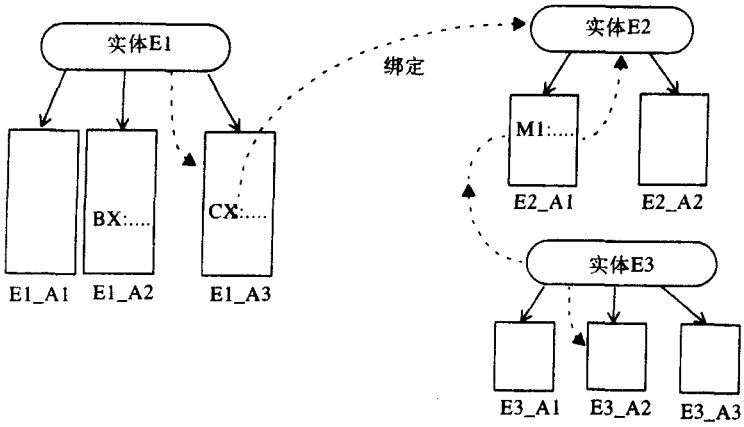


图2-2 实体E1的配置

一旦实体模型建立，需要一个VHDL系统来验证。典型的VHDL系统包括分析器和仿真器。
11 分析器读入单个文件中的一个或多个设计单元，经过语法验证和静态语义检查之后，把这些单元编译到一个设计库中。设计库是主机环境（也就是支持VHDL系统的环境），存储编译后的设计单元。

仿真器首先从设计库中读入编译后的描述，对实体进行仿真（实体通过实体-结构体对或配置来描述），然后进行下面的操作：

1. 细化
2. 初始化
3. 仿真

需要注意以下几个语法问题：VHDL语言对大小写不敏感，也就是说，对大写字母和小写字母相同对待（除了扩展标识符，字符串文字和字符文字）。例如，CARRY、CarrY和carrY都是指同一个名字。VHDL与Ada和Pascal非常相似，都是比较自由的。可以通过在文本前面加两个连续的短线（--）来加入注释。所有位于这两条短线和同一行结束位置之间的文本都属于注释部分。

本节介绍的术语将在后面的章节中详细介绍。

2.2 实体声明

实体声明定义实体名称和接口端口。*端口*是实体与其外部环境的其他模块进行通信的
12 信号。