计算机专业人员书库

Windows 设备驱动程序（VxD与WDM）开发实务

武安河　周利莉　编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL: http://www.phei.com.cn

计算机专业人员书库

Windows 设备驱动程序（VxD 与 WDM）开发实务

武安河　周利莉　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

本书主要介绍了用 VtoolsD 开发 Windows 95/98/Me 下的非标准设备（指自己开发的，如数据采集卡等设备）驱动程序 VxD 的原理及编程方法。本书详细介绍了 VxD 的基本概念、VxD 程序、VxD 和 Win32 应用程序之间的通信，及 VxD 访问硬件设备，处理硬件中断，实现 DMA 操作，完成即插即用（PnP）功能的函数和类库，并详细介绍了 VtoolsD 开发工具的使用和如何用 VtoolsD 开发 Windows 下的 PCI 设备驱动程序 VxD。

本书还介绍了用 DriverWorks 开发 Windows 98/NT/2000 下的驱动程序 WDM 的原理及编程方法。

本书附有大量的编程实例，并附赠一张光盘，内含全部实例源代码便于读者学习和掌握。

本书是一本技术性较强的工具书和实用参考书。本书的适用对象是具有一定计算机硬件及 C++语言基础的计算机应用开发人员和高等院校学生。

图书在版编目（CIP）数据

Windows 设备驱动程序（VxD 与 WDM）开发实务/武安河，周利莉编著. -北京：电子工业出版社，2001.9
（计算机专业人员书库）
ISBN 7-5053-6924-5
Ⅰ. W... Ⅱ. ①武... ②周... Ⅲ. 窗口软件. Windows-驱动程序-程序设计 Ⅳ. TP316.7

中国版本图书馆 CIP 数据核字（2001）第 057632 号

丛 书 名：计算机专业人员书库
书　　名：Windows 设备驱动程序（VxD 与 WDM）开发实务
编 著 者：武安河　周利莉
责任编辑：施玉新　朱沭红
特约编辑：蔡祖瑛
印 刷 者：北京天宇星印刷厂
装 订 者：河北省涿州桃园装订厂
出版发行：电子工业出版社　URL:http://www.phei.com.cn
　　北京市海淀区万寿路 173 信箱　邮编 100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：19.75　字数：502 千字
版　　次：2001 年 9 月第 1 版　2001 年 9 月第 1 次印刷
书　　号：ISBN 7-5053-6924-5 / TP・3943
印　　数：6000 册　定价：36.00 元（含光盘）

出 版 说 明

信息产业的飞速发展迫切要求人们快速掌握新技术，并加以应用。选择适合自身的好书，少走弯路，从而节约时间，赢得主动就显得尤为重要。作为出版者，提供明确读者定位的作品，让读者对在什么情况下选择什么样的书有一个理性的而不是情绪化的选择，是我们的责任和义务。

秉着“专”和“精”的原则，我们推出这套《计算机专业人员书库》。

- ✓ 专——内容专，丛书面向的是有一定专业基础的计算机应用开发人士。
- ✓ 精——内容和作者精。在精心的前期选题内容组织安排下，挑选来自专业领域的资深技术专家和大学教授作为本套丛书的作者，保证丛书的质量。

出版高品位、高品质的图书是电子工业出版社计算机图书事业部的努力目标，您的意见是我们创造精品的动力源泉！如果您对本丛书有任何意见和想法，或有意撰写面向计算机专业人员的书籍，欢迎指点或垂询！

我们的 E-mail：jsj@phei.com.cn

电话：010-68216158。

电子工业出版社

计算机图书事业部

2001 年 09 月

前　言

在 Windows 环境下，如何开发设备驱动程序？这对大多数计算机应用开发人员来说可能是一个问题，因为现在国内有关这方面的资料和书籍非常少。

本书的读者

如果你是一名在校的大学生，所学的专业涉及《微机原理与应用》课程，那么这本书是你最好的参考书之一，她教你如何在 Windows 环境下实现对硬件的控制。

如果你是一名在职的计算机应用开发人员，正在或将要从事微机设备的开发工作，那么这本书是你最好的参考书之一，她教你如何编写 Windows 环境下的设备驱动程序。

什么是设备驱动程序？

从广义上来说，设备驱动程序就是控制硬件设备的一组函数。

在 Windows 环境下，如果要处理硬件中断，实现 DMA 操作，就一定要用到设备驱动程序，开发即插即用（PnP）设备（如 PCI 接口卡）更是这样。那么 Windows 环境下驱动程序究竟有多少种类？有三类。

1．VxD

VxD 是 Virtual Device Driver 的缩写，是 Microsoft 公司所提倡的新一代物理设备驱动程序。它运行在系统保护层（Ring 0 层）下，可以完成各种各样的系统物理层的访问。VxD 模式起源于 Windows 3.1 时代，一直到现在，它仍然在 Windows 95/98/Me 操作系统中起主导作用。

2．KMD

KMD 是 Kernel Mode Driver 的缩写，是在 Windows NT 下提出的管理、维护硬件运作的驱动程序模式。

3．WDM

WDM 是 Win32 Driver Model 的缩写，是 Microsoft 公司力推的全新的驱动程序模式。它的运行平台是 Windows 98/Me/NT/2000 操作系统。在不久的将来，在 Windows 平台上，WDM 将成为主流的驱动模式。

但在 Windows 98/Me 下，用户实在没有必要用 WDM 写驱动程序，因为其实际运作速度比较慢，而且代码变得十分长；再说，对于很多在 VxD 下就能很轻松解决的问题，使用 WDM 处理却很麻烦。

现在国内有关 VxD 方面的书籍，笔者见到的有三本，讲的基本上是虚拟设备驱动程序的原理与入门。若把它们用做开发设备驱动程序的参考书，其内容显然是不够的。

真正把 VxD 作为设备驱动程序来写的，国外有一本，书名是《Writing Windows VxDs and Device Drivers》，作者是 Karen Hazzah 先生。使用的工具是 VtoolsD，写得相当好。不过他写的时间比较早，使用的是 C 语言，而且它没说明工具的使用，作为 VxD 入门教程显然不太适合。

有关 WDM 方面的书籍就更少了，只见到过一本翻译的书。另外从网上看到一本英文的 WDM 书，书名是《Programming the Microsoft Windows Driver Model》，作者是 Walter Oney 先生。两本书的开发工具都是 DDK。

开发工具

开发设备驱动程序需要专门的开发工具，目前应用广泛的工具主要有两大类。

一类是 Microsoft 公司提供的 Windows DDK（Device Driver Kit）。它有 Windows 98 DDK 和 Windows 2000 DDK 两个版本。Windows 98 DDK 能够开发 Windows 95/98/Me/NT 下的 VxD、KMD 和 WDM 驱动程序。Windows 2000 DDK 能够开发 Windows 98/Me/NT/2000 下的 KMD 和 WDM 驱动程序。

另一类是 NuMega 公司提供的 VtoolsD 和 DriverWorks。

VtoolsD 开发包提供了对 VxD 编程的 C/C++类库支持，利用 VtoolsD 中的 QuickVxD 工具可以快速生成 VxD 的 C/C++代码框架，开发者可以在此基础上根据各自的需要添加自己的代码。早期的 VtoolsD 开发包名称为 VtoolsD for Windows 95（2.03 版本），提供了五个不同的工具。现在的 VtoolsD 开发包名称为 NuMega VtoolsD（3.0 版本），提供了三个不同的工具。相比较而言，使用 VtoolsD 要比使用 DDK 简单、快捷。

DriverStudio 是一个大的开发工具包，它包含 VtoolsD、SoftICE 和 DriverWorks 等开发工具，有 1.5 版本，2.00 版本和 2.01 版本。其中 DriverStudio2.01 版本中的 VtoolsD 版本的版本号为 3.05。DriverWorks 用于开发 KMD 和 WDM 驱动程序，但它需要 Windows 98 DDK 和 Windows 2000 DDK 的支持。

推荐使用 DriverWorks 开发 WDM，用它比使用 DDK 更简单、方便。

另外，开发设备驱动程序的调试工具也很重要，常用的有 NuMega 公司的 SoftICE，使用非常方便。

本书的内容

本书是关于设备驱动程序 VxD 和 WDM 的入门和提高级参考书。使用 C++语言，工具是 NuMega 公司提供的 VtoolsD 和 DriverWorks。。

本书有关 VxD 分为三部分：

第一部分主要讲述 VxD 程序的基本知识，包括 QuickVxD、Monitor 和 INF Editor 工具的使用。它包括第 1 至第 3 章、附录 A 和附录 B。

第二部分主要讲述 VxD 与应用程序之间的通信。它包括第 4 章和第 9 章。

第三部分主要讲述作为设备驱动程序 VxD 处理硬件的类库和函数。它包括第 5 至第 8 章、第 10 和第 11 章。

本书有关 WDM 部分主要讲述 WDM 程序的基本知识、入门和基本编程，包括第 12 至第 14 章。

另外本书还讲述了 SoftICE 的使用。它包括附录 C 和附录 D。

本书配套光盘

本书配套光盘含有书中所有实例的驱动程序和应用程序的全部源代码，以及生成的驱动程序和可执行的应用程序。

书中 VxD 的 19 个实例中，除 4 个为经修改的 VtoolsD 范例外，其他均为作者所创建。所有实例均在 Windows 98 操作系统上调试通过，也曾在 Windows 95/Me 操作系统上实验过部分程序，全部通过。

本书配套光盘的有关 WDM 部分共提供了三个 WDM 实例：一个为 Windows 2000 DDK 所开发的实例，其他两个是在 Windows 98 下用 DriverWorks 所开发的实例。

本书是作者教学和科研工作经验的总结。每章所附实例都是作者精心设计的，对理解

和掌握驱动程序的编程有非常大的帮助。相信本书对广大计算机爱好者和工程技术人员一定会有所帮助。

感谢解放军信息工程大学基础部刘昱旻主任和教研室领导王华、谭彦彬和朱可云的支持和帮助。

感谢阎玉祥老师和曾雷同志的支持和帮助。

另外在本书的编写过程中，得到了学生吴强、郭臣、饶玉萍及李明星的支持和帮助。

非常感谢国内的一些网站和网友们，感谢他们提供了所需的编程工具和资料。

由于作者的理论水平有限，书中难免出现差错和遗漏，敬请广大计算机应用开发人员批评指正。

作者　武安河

2001 年 6 月 写于解放军信息工程大学

目　录

第1章 基本知识

1.1 Windows的虚拟世界

Windows 95运行3种不同类型的应用程序：DOS应用程序、Win16应用程序和Win32应用程序。为了克服这些程序之间的不相容性，Windows在“虚拟环境中的虚拟机”上执行这些程序。在开发应用程序时，Windows程序员通常会忽略虚拟环境和真实环境之间的区别。对于大部分应用程序来说，虚拟环境就是真实环境。

而编写VxD则不同，因为VxD运行在管理层，即虚拟机之外。实际上，VxD是实现虚拟机的管理软件的一部分。这样VxD的编写者就需要更多地了解虚拟环境和物理环境的区别，以及Windows怎样创建虚拟机。对于那些需要利用一个应用程序虚拟环境中的资源来开发VxD的程序员来说，充分了解虚拟机尤为重要。

1.1.1 什么是虚拟机

虚拟机是由系统创建的假象；虚拟资源是硬件(有时是软件)资源的仿真。为了产生虚拟资源，仿真必须是完全的，以便在编写程序时硬件是真实的，而不是仿真的。例如，虚拟内存系统使用硬盘空间、系统软件、专门的处理器性能以及少量的物理内存来仿真具有大量物理内存的系统。对于运行在虚拟环境中的程序，仿真使得整个虚拟地址空间就像真实存在的物理内存空间一样。这样的内存系统叫做“虚拟化的”系统。

当一个系统虚拟了所有或者几乎所有的程序可访问的资源时，它就创造了一个“虚拟机”（VM）。程序可访问的资源包括寄存器、内存和辅助设备（如键盘和显示器等）。在Windows环境中创建虚拟机，其真实原因便是为了支持存在的DOS应用程序。对于一个DOS应用程序来说，总认定自己是惟一运行的应用程序，常常直接访问硬件，独占所有的系统内存和系统运行时间。但在Windows环境中，DOS应用程序并不是惟一运行的应用程序，因此Windows创建虚拟机供DOS应用程序运行。

虚拟机指的是一个任务具有自己的执行环境，它包括：

- 内存空间；
- I/O空间；
- 中断操作；
- 寄存器。

虚拟这样一个机器，需要专门的处理器支持，而且还需执行大量代码。80386(以及以后的兼容处理器)支持完成包括地址转换、所需内存分页、I/O地址捕获、指令捕获和中断捕获的复杂工作。

虚拟机的管理主要由虚拟机管理器(VMM)完成。VMM利用系统的硬件，不仅创建一个虚拟机，而是创建几个相互独立的虚拟机。每个虚拟机都有自己的虚拟执行环境。所有的Windows应用程序(Win32应用程序和Win16应用程序)运行在一个VM（Virtual Machine，

虚拟机）内，称为系统 VM，而每一个 DOS 应用程序运行在自己独立的 VM 内。每个虚拟环境可能与物理环境根本不同。

在 Windows 95 环境下，任务的执行单元被称为线程。一个 DOS VM 只有一个单独的线程。在系统 VM 中，所有的 Win16 程序分享一个单独的线程，而对于 Win32 程序来说，每个程序有自己的线程。Win32 程序可以是多线程的，由主线程运行创建其他的线程。

1.1.2 处理器模式

为了创建和维护虚拟机，VMM 开发了 80386 以及以后的兼容处理器的特殊性能。这些处理器有三种运行模式：实模式、保护模式和 V86 模式。Windows 95 利用其中的两种模式：保护模式和 V86 模式。

处理器模式决定了以下几个重要的执行性能：

- 处理器所能寻址的内存空间；
- 处理器如何实现程序的逻辑地址到总线的物理地址的转换；
- 处理器如何保护对内存和 I/O 端口的访问以及阻止某些指令的执行。

1. 实模式

80386 处理器开机或复位时，即进入实模式。在实模式下，可寻址的内存空间为 1MB。在实模式下，修改控制寄存器 CR0 中的 PE 位，使其置 1，可进入保护模式。在保护模式下，若将 CPU 状态标志寄存器中的虚拟方式标志 VM 置 1，可进入 V86 模式。

2. 保护模式

保护模式包含 32 位和 16 位两种保护模式。两者的最大区别在于可寻址的内存空间大小。16 位保护模式可寻址的内存空间为 16MB，而 32 位保护模式可寻址的内存空间为 4GB。对于物理内存而言，4GB 可能太大了，但对于提供虚拟内存的操作系统来说仍然有用。

再者，两者的最重要的区别在于段的大小。16 位保护模式的段被限定在 64KB，开发人员在编写大型程序时必须注意段的问题。而 32 位保护模式的段为 4GB，段如此之大，以至于对于 Windows 95 的 32 位保护模式来说，程序员是见不到段的，应用程序也从来不需要改变段。

在 Windows 95 环境中，对 32 位和 16 位两种保护模式来说，处理器实现程序的逻辑地址到物理地址的转换的方法是相同的。它分两个步骤：先查询选择器表将逻辑地址（由选择器和偏移量组成）转换成线性地址，再通过分页将线性地址转换成物理地址。

3. V86 模式

V86 模式的存在是仅仅为了仿真实模式。原先的 PC 机仅仅支持实模式，DOS 应用程序便是实模式下的应用程序。为了支持今天仍存在的 DOS 应用程序，X386 以后的处理器推出了 V86 模式。这种仿真允许 Windows 系统执行多任务 DOS 应用程序。

V86 模式可寻址的内存空间为 1MB，同实模式。程序的逻辑地址到物理地址的转换的方法是这样的：段左移 4 位加偏移量获得线性地址，线性地址到物理地址的转换是靠分页来完成的。分页对 DOS 应用程序来说完全是透明的。

为了防止由于 DOS 应用程序的出错而导致的系统崩溃，V86 模式也采取了一些保护机

制。在 V86 模式下运行的 DOS 应用程序，如果运行某些优先权指令，如访问某些 I/O 端口和禁止的存储缓冲区，便会产生异常事件，操作系统就会进行相应的处理。

1.1.3 Windows 的执行环境

Windows 95 体系结构支持 4 种不同的基本程序：管理程序、Win32 应用程序、Win16 应用程序和 DOS 应用程序。每一种程序都在不同的执行环境中运行。一个执行环境的描述包括处理器模式、优先权级别和比特位数，执行环境的描述见表 1-1。

表 1-1 Windows 的执行环境

程序类型	模 式	优先权级别	位 数	内存模式	VM
管理程序	保护	Ring 0	32	平面	VM 之外
Win32	保护	Ring 3	32	平面	系统 VM
Win16	保护	Ring 3	16	分段	系统 VM
DOS	V86	Ring 3	16	分段	独立 VM

管理程序在 Ring 0 层级别(最高的访问级别)的保护模式中运行，所以它们能够访问和控制实际的硬件环境。也就是说，管理程序在实际的机器上运行，而不是在虚拟机上运行。在所有构成 Windows 的组件中，只有 VMM 和 VxD 在管理程序环境中执行。

管理程序环境是 32 位的，这些程序可以访问 4GB 的虚拟内存。管理程序只使用两个选择器，每个选择器都可以访问 4GB 的虚拟内存。这两个选择器只是在属性上有所不同：一个被标记为 executable（可执行的）并被调入到 CS 中；另一个则被标记为 non-executable（不可执行的）并被调入到 DS、ES 和 SS 中。内存模式的这种类型(段只被调入一次并且以后再不调入)被称为平面模式，它使得分段对程序员来说几乎不可见。

Win32 应用程序和 Win16 应用程序都运行在 Ring 3 层级别(最低的访问级别)的保护模式下。Win32 应用程序采用平面内存模式，可以访问 4GB 的虚拟内存。Win16 应用程序采用一般的分段式内存模式(而不是平面模式)，只能访问 16MB 的虚拟内存。

DOS 应用程序并不运行在保护模式下，而是运行在 V86 模式下。V86 程序采用 8086 的翻译类型，没有选择器和描述器，段只是简单地变换成地址。V86 程序也运行在 Ring 3 层访问级别上，访问硬件资源和中断被隐藏和虚拟，内存模式是分段的。

1.1.4 如何实现虚拟环境

当在虚拟机中执行程序时，Windows 必须能够捕获并透明地“代替”以下功能：

- 对一个 I / O 端口查询；
- 对一个内存映射的外围设备查询；
- 可能导致传送到虚拟机之外的操作，如一个异常事件或中断。

1. 捕获 I / O 操作

不论保护模式还是 V86 模式，操作系统均可以通过捕获输入和输出指令来防止应用程序直接访问 I / O 映射的设备。Windows 95 使用 I / O 优先级(IOPL)和 I / O 允许映射(IOPM)来控制 VM 访问 I / O 地址。

在保护模式中，每个代码段都有一个相关的优先级描述器，它存储在描述器列表中。当在保护模式中执行一个 I / O 指令时，处理器将段的 IOPL 和当前可执行代码段的优先级(CPL)进行比较。如果 CPL＜IOPL，处理器则执行指令。如果 CPL≥IOPL，处理器使用 IOPM 作为保护的第二级。IOPM 是端口比特映射的列表：1 表示“拒绝访问”，0 表示“准许访问”。所以当 CPL≥IOPL 并且 IOPM＝0 时，处理器才执行指令；否则，处理器产生一个异常事件。

在 Windows 95 中，不论 SYS VM 还是 VM，IOPM 都是 I / O 的主要优先权机制。因为在系统 VM 中程序的 CPL>IOPL，所以 IOPM 就是所有虚拟机中 I / O 的主要优先权机制。在 V86 模式中，处理器总是参照 IOPM 而不管 IOPL 如何。

所以，通过操作 IOPM，Window 95 可以捕获对特殊端口的访问。Windows 95 使用这一特性来“虚拟”位于被捕获端口地址的物理设备。通过设备驱动程序跟踪对设备的访问，Windows 可以保持每个使用设备的 V M 的状态信息。

Windows 95 和它的标准组件 VxD 几乎可以捕获 PC 机上所有的标准 I/O 设备，不捕获非标准的 I / O 设备。表 1-2 列出了可捕获的端口位置。一个第三方的 VxD 可以捕获其他端口。

表 1-2 标准组件 VxD 捕获的 I/O 端口

端口地址	VxD	说明
00-0F/80-90/94-9F/C0-DF	VDMAD	DMA 控制器
20/21/A0/A1	VPICD	中断控制器
40-43	VTD	定时器
60/64	VKD	键盘
61	VSD	系统扬声器
2F8-2FF/3F8-3FF	SERIAL	串口
378-37F	LPT	并口
3B4/3B5/3BA/3C0-3CF/3D0-3DF	VDD	VGA

VMM 负责管理 IOPM。VxD 可以向 VMM 申请捕获特殊端口的服务，提出请求时，还要有安装捕获特殊端口的回调处理器。VMM 通过配置 IOPM 来响应这样的申请。当 VM 访问端口时，便产生异常事件。VMM 的异常处理器调用 VxD 的回调处理器。VxD 的回调处理器可作任何处理：VxD 可以忽略指令，可以执行指令，可以替换值(如 OUT 3F8，01H 可变为 OUT 3F8，81H)。

2. 捕获内存操作

尽管多数标准设备是为 I/O 捕获的，但仍有一些是为内存捕获的。Windows 95 主要根据页错误机制实现对存储器映射设备访问的虚拟化。要捕获对某个设备内存的访问，设备 VxD 会将页表中对应于物理内存的页标注为“不存在的”，并在 VMM 中注册它的页错误处理器。当在虚拟环境中运行的程序试图访问这个页时，就会产生一个页错误。VMM 的异常处理器会调用已注册的设备 VxD 的页错误处理器，VxD 页错误处理器决定产生与虚拟环境的请求相关联的动作。

虚拟视频设备(VDD)就使用这一机制来虚拟视频的帧缓冲区。当 DOS 的程序往位于逻

辑地址 B000：0000H 的视频缓冲区写数据时，输出并不显示在屏幕上，因为 VDD 捕获了它，并且将其放入物理内存的另外位置中。这样一来，写入视频缓冲区的数据只能在窗口上显示，并不在整个屏幕上显示。

3. 捕获中断和异常事件

除了捕获内存和 I / O 访问，Windows 还要获取某些具有优先级的指令。这些指令可能会绕过处理器的保护特性或者干扰虚拟机的完整性。优先级指令包括：影响处理器中断标志的指令(CLI，STI，POPF，IRET)或软件中断的指令(INT n)，以及那些调入描述器表的指令(LLDT，LGDT，LIDT)。Windows 获取这些指令以保证 VM 的完整性。例如对应于 INT 指令，Windows 开发了透明地截取 DOS 和 BIOS 调用的软中断。

在虚拟机中的程序运行在 Ring 3 层中。在这里执行一个有优先级的指令，就会产生一个异常事件。这时处理器会切换到 Ring 0 层，并且将控制交给相应的控制器。

进一步说，每个段都有一个相应的优先级描述器(DPL)。段的优先权等级决定大部分指令(例如 LLDT，LGDT)的优先权等级。另外一些指令（影响处理器中断标志的指令）则从 IOPL 中得到它们的优先权等级。例如在 Ring 3 层中的程序执行 STI 或 CLI 指令，只有 CPL＞IOPL 时，处理器才会触发一个异常事件。

VM86 环境和系统 VM 环境的主要区别之一就是涉及这些基于 I / O 的优先权。虽然 386 体系结构支持在两种情况下捕获 CLI 和 STI 指令，但 Windows 95 不能在 VM86 模式下捕获 CLI 和 STI 指令。VMM 故意为 DOS 应用程序设置为 CPL＝IOPL，所以 CLI 和 STI 不能产生异常事件。不过，Windows 应用程序（Win32 和 Win16）的 IOPL=0，所以 CLI 和 STI 将产生异常事件。VMM 异常处理器能禁止或允许系统 VM 的虚拟中断标志，但处理器的中断标志不受影响。

1.2 VxD 简介

VxD 是虚拟设备驱动程序(Virtual Device Driver)的缩写，中间的 x 表示某一类设备。如 VKD.VXD 表示键盘的驱动程序，VDMAD.VXD 表示系统 DMA 的驱动程序。不过作为设备驱动程序，一个 VxD 并不单纯要虚拟化设备。有些 VxD 虚拟化设备，有些 VxD 则只是设备驱动程序，并不做虚拟化设备的工作。还有的 VxD 并不影响任何设备，它们只是为其他的 VxD 或应用程序提供服务。

VxD 可以随 VMM 一起静态安装，也可以根据请求动态安装。在这两种情况下，VxD 与虚拟机管理器(VMM)紧密合作并共享执行程序。这种与操作系统的特殊关系使得 VxD 既不属于 Windows 应用程序，也不属于 DOS 应用程序。VxD 能够无限制地访问所有硬件设备，自由地检测操作系统的数据结构(如描述符和页表)，以及访问任何的内存位置。VxD 也能够捕获软件中断，捕获 I/O 端口和内存区域访问，甚至可以截获硬件中断。

虽然 Windows 和 DOS 应用程序也能够做低级任务(如捕获软件中断)，但是应用程序总要受到限制。例如，一个 Windows 应用程序可以捕获其他 Windows 应用程序获得的软件中断，但是不能捕获 DOS 应用程序获得的软件中断。而 VxD 不管中断的来源是哪里都可以捕获。

VxD 的运行平台是 Windows 3.1/95/98/Me 操作系统，本书中的 Windows 如无特别说明，

指的是 Windows 95/98/Me。

1.2.1 VxD 的安装

Windows 支持静态安装和动态安装 VxD。静态安装是在 Windows 初始化时安装 VxD 并一直保留在 Windows 中。如果一个 VxD 只被一个特定的应用程序所使用，或者其存在只是给某些应用程序提供服务，那么当这个 VxD 不用时它所占用的内存就浪费了，这种情况最好不要选择静态安装。

Windows 支持两种静态安装方法。第一种方法 Windows 3.X 也支持，就是在 SYSTEM.INI 文件中定义 device= XXX.VXD。另一种方法只适用于 Windows 9X，是在注册表中添加一个为静态 VxD 名字的键值（例如 XXX.VXD=pathname）。添加的位置在子键 \HKLM\System\CurrentControlSet\Services\ VxD 处。

动态安装的 VxD 并不在 Windows 初始化时自动安装，而是在一个应用程序或其他 VxD 的控制下安装和卸载。例如，即插即用 VxD 必须动态安装，因为 Windows 支持硬件在运行时进行删除和重新配置。支持这种硬件的 VxD 就可以在任何需要时安装和卸载。

动态安装的 VxD 和设备驱动程序一样也只被一个特定的应用程序所用。当这个应用程序需要使用这个设备时就安装 VxD，用 CreateFile()函数动态加载 VxD。当应用程序完成时，VxD 就被卸载，调用 CloseHandle()动态卸载 VxD。

静态安装和动态安装的 VxD 反映在 VMM 的信息设置上时有轻微的不同。有些信息只有静态 VxD 可获得，有些信息只有动态 VxD 可获得，不过大多数的信息两者都可获得。事实上，编写支持这两种安装方法的 VxD 很容易，只要反映这两者的信息设置就可以了。

1.2.2 VxD 的基本结构

虽然 VxD 使用 32 位平面内存模式，VxD 代码和数据仍然要组织成段的形式。VxD 使用以下类型的段：

- 实模式初始化段；
- 保护模式初始化段；
- 可分页段；
- 锁定段(不可分页段)；
- 静态段；
- 只调试段。

每一个类型段都有一个代码段和一个数据段，这样一个 VxD 可以有 12 个段。实模式代码和数据段是 16 位的，其他段都是 32 位的。

实模式初始化段代码在 VMM 进入到保护模式之前，就在 Windows 初始化过程中执行了。由于初始化很早，所以每一个静态安装的 VxD 都有机会来检查 Windows 安装前的实模式环境，然后决定是否应继续安装 VxD。在 AX 寄存器中返回退出代码时，VxD 通知 VMM 继续安装 VxD 的保护模式部分，或放弃安装 VxD，甚至放弃安装 Windows。

大多数 VxD 不需要一个实模式初始化程序，但是也有例外。

每一个静态安装的 VxD 的实模式部分执行完后，VMM 进入保护模式并给每一个静态安装的 VxD 一个机会来执行保护模式初始化段的代码。保护模式初始化代码也能返回一个错误代码来通知 VMM VxD 初始化失败。如果一个 VxD 报告初始化失败，VMM 就标记该