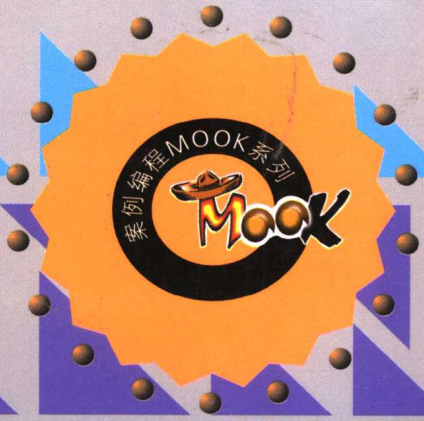
案例编程MOOK系列

Visual C/C++

编程精选集锦

网络与通信分册



《电脑编程技巧与维护》杂志社 主编

科学出版社
www.sciencep.com

案例编程Book系列

Visual C/C++编程精选集锦

网络与通信分册

《电脑编程技巧与维护》杂志社　主编

科学出版社

北　京

内 容 简 介

Visual C/C++ 作为功能强大的可视化应用程序开发工具，是计算机界公认的优秀应用开发工具。Microsoft 的基本类库 MFC 使得开发 Windows 应用程序变得很容易，适合作各种系统软件、应用软件、网络软件、游戏软件等开发平台。

根据 Visual C/C++ 的不同应用对象，将精选的 190 个实例分为数据库及图形图像分册、网络与通信分册、关键技术精解分册出版。本书为网络与通信分册。全书本着实用第一的原则，紧紧围绕主题展开，循序渐进，由浅入深地介绍了使用 Visual C/C++ 进行应用程序开发的思想方法与编程技巧。

本书的特色体现如下几点：第一，每一章都是通过一个个的实例来介绍 Visual C/C++ 应用编程方法和技巧，避免枯燥、空洞的理论，并且每一个实例都具有很强的实用性和代表性。第二，所选的每一个实例都是从事 Visual C/C++ 应用编程人员的经验总结，具有很强的实用性，其中很多编程技巧可供借鉴。第三，每一个实例的程序源代码都是经过上机调试通过，给程序开发人员移植源代码带来了方便，从而加快应用编程的步伐。第四，对老版本经典实例进行点评，选取一些老版本开发环境的经典实例加以点评分析，使之能够起到触类旁通的作用。

本书适用于有一定 Visual C/C++ 应用基础的编程人员和应用开发人员，对初学 Visual C/C++ 编程的读者也有一定的参考价值。

图书在版编目（CIP）数据

Visual C/C++编程精选集锦（网络与通信分册）/《电脑编程技巧与维护》杂志社主编. —北京：科学出版社，2003

（案例编程 MOOK 系列）

ISBN 7-03-011498-1

I. V… II. 电… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字（2003）第 036157 号

策划编辑：孟战龙/责任编辑：万国清

责任印制：吕春珉/封面设计：三函设计

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

http://www.sciencep.com

新蕾印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2003 年 5 月第 一 版 开本:B5(720×1000)

2003 年 5 月第一次印刷 印张:32

印数:1—5 000 字数:613 000

全套定价：150.00 元

（共三册 光盘另加 5.00 元）

（如有印装质量问题，我社负责调换〈路通〉）

出版说明

MOOK 又称杂志书，是杂志(Magazine)和图书(Book)相结合的新型出版物，它集合了二者的优点，既有图书的内容深入全面、更具专业性和权威性，又有杂志的贴近生活、轻松、休闲、通俗易懂、时效性强的特点。

MOOK 代表了一种出版趋势，即图书的杂志化和杂志的图书化，在这里可以发表专家学者的真知灼见，它营造的是一种轻松、休闲的氛围，专家学者用读者更易于理解的语言，阐释理论、知识和信息。

“案例编程 MOOK 系列”是科学出版社策划出版的一套 MOOK 丛书，包括《Visual C/C++ 编程精选集锦》(三册)、《Visual Basic 编程精选集锦》、《Java 编程精选集锦》、《Delphi 编程精选集锦》、《ASP 编程精选集锦》共七本。每本书有几十个编程实例，并依不同的编程应用分成了若干章、条目清晰可查、使用极为方便。这套书编入了《电脑编程技巧与维护》杂志近一、两年发表的精彩编程文章，并根据读者要求，组织收入了更具价值的编程案例。这套书遵循 MOOK 的特点：讲究内容的深入以及专业性和权威性，同时兼顾轻松、通俗易懂、时效性强等特点。实例的讲解是先给出设计目标，然后介绍实现目标的基本思想和方法，最后详细给出其核心程序的源代码，对程序的关键部分进行讲解并给出程序的运行效果。

“案例编程 MOOK 系列”的与众不同，在于它的理念。这里少了教科书上的说教，少了尘世浮华的喧嚣。这里带给你的是一份清新、纯粹的体验感受。把它作为生活的一部分，用心经营，仔细体会。“案例编程 MOOK 系列”的与众不同，在于它的思想。汇集众多顶级程序员、项目经理的成功经验，告诉你最新的创意和最实用的方法。力求为你建造一个真正的知识整合，思想交流的平台。你可以学习编程高手的诀窍、丰富你的编程技巧，你可以与高手切磋编程实战技巧，你可以积累更多的经验。

思想、智慧、观念、技巧无处不在……

前　言

Visual C/C++ 作为功能强大的可视化应用程序开发工具，是计算机界公认的优秀应用开发工具。Microsoft 的基本类库 MFC 使得开发 Windows 应用程序变得很容易，适合作各种系统软件、应用软件、网络软件、游戏软件等开发平台。

《Visual C/C++ 编程精选集锦》精选了《电脑编程技巧与维护》杂志近一二年发表的精彩编程文章，并根据读者要求，组织收入了更具价值的编程案例。《电脑编程技巧与维护》杂志是为从事电脑编程、系统应用和人员创办的专业性和实用性都很强的技术刊物，它从 1994 年创刊八年多以来，始终遵循着“实用第一，智慧密集”的办刊宗旨，紧紧跟踪计算机软硬件技术发展和应用趋势，不断求变创新。针对软件开发过程中许多要点和技巧问题，着重提供各类解决方案。对电脑编程人员来说，程序开发能力的提高，除了对语言和算法的学习，还要集思广益，充分借鉴别人的长处，深入透彻地理解其中的精髓，然后溶入到自己的设计能力中去，这样无论是对于自身和整体都有莫大的提高，这正是编写这套书的初衷。

根据 Visual C/C++ 的不同应用对象，将精选的 190 个实例分为数据库及图形图像分册、网络与通信分册、关键技术精解分册出版。网络与通信分册分为 4 章，第 1 章 Visual C/C++ 在硬件编程中的应用，介绍 Visual C/C++ 硬件编程中的一些方法和技巧；第 2 章网络与通信应用编程，介绍 Visual C/C++ 在网络编程方面的应用实例和技巧；第 3 章计算机维护应用编程，介绍 Visual C/C++ 在计算机维护中应用的实例和技巧；第 4 章计算机安全应用编程，介绍计算机维护的一些实例。

本书为网络与通信分册。全书每一章都本着实用第一的原则，紧紧围绕一个主题展开，循序渐进，由浅入深地介绍了使用 Visual C/C++ 进行应用程序开发思想方法与编程技巧。

本书的特色体现如下几点：第一，每一章都是通过一个个的实例来介绍 Visual C/C++ 应用编程方法和技巧，避免枯燥、空洞的理论，并且每一个实例都具有很强的实用性和代表性。在实例的讲解上一般都是先给设计目标，接着介绍实现该目标的基本思想和方法，然后详细给出其核心程序的源代码，对程序的关键部分进行讲解并给出程序的运行效果。第二，所选的每一个实例都是从事 Visual C/C++ 应用编程人员的经验总结，具有很强的实用性，其中很多编程技巧可供借鉴。第三，每一个实例的程序源代码都是经过上机调试通过，给程序开发人员移植源代码带来了方便，从而加快应用编程的步伐。第四，对老版本经典实例进行点评，选取一些老版本开发环境的经典实例加以点评分析，使之能够起到触类旁通的作用。

本书是《电脑编程技巧与维护》资源的二次开发，浓缩了 Visual C/C++ 程序设计的精华，其目的是提升 Visual C/C++ 程序开发能力，把应用 Visual C/C++ 进行编程的心得体会、经验与大家共享。本书定位于有 Visual C/C++ 应用基础上的编程人员和应用开发人员，对初学 Visual C/C++ 编程的新手也有一定的参考价值。本书内容全面、概念清晰、层次分明，例题典型而实用，但不足甚至疏漏之处在所难免，恳请广大读者批评指正。

《电脑编程技巧与维护》杂志社

目　录

第1章　Visual C 在硬件编程中的应用

第2章　网络与通信应用编程

第3章　计算机维护应用编程

第4章　计算机安全应用编程

第 1 章

Visual C 在硬件编程中的应用

实例 1

Visual C++6.0 中利用图像扫描控件控制扫描仪

扫描仪的发展虽然只有十几年的历史，但因其采用封闭的光学扫描环境，受周围环境的影响小，图像稳定，扫描精度高，再加上计算机技术的飞速发展和图形环境的日益普遍，扫描仪迅速成为计算机不可缺少的图文输入工具之一，被广泛地应用于图形、图像处理的各个领域。

在实际中，许多人需要自己编写图像处理分析应用程序，如果在这种程序中加入扫描功能，就可以使程序更完美。如何控制扫描仪并使扫描、处理图像实现操作一体化呢，例如扫描图像并在捕获图像扫描结束后，直接进行图像处理或其他处理。本人在用 Dexxa Flatbed Scanner 扫描仪进行网络数据传送编程时，简单地利用 Kodak 图像扫描控件实现了这种功能。程序实现了对安装的扫描仪应用程序的调用，并在扫描结束的消息处理函数中，加入了图像处理和网络程序函数，从而实现了扫描到网络传送的一体化。

本文利用一个基于对话框的程序，包括一个扫描开始按钮和一个取消按钮，简单地介绍了在Visual C++6.0 中实现图像扫描控件控制扫描仪的方法。在按下扫描开始按钮后，即可自动调用安装了的扫描应用程序界面，扫描完成后，进入了消息处理函数中进行其他的操作。

一、在程序中嵌入图像扫描控件

首先，在 Visual C++ 6.0 中新建一个 MFC AppWizard(exe)项目文件，并命名为 scanner，在 AppWizard 第一项中选基于对话框项，第二项中选 Active Controls 复选框，其他缺省。这样建立了一个以对话框为主窗口的应用程序。

然后，在 Resource View 中新建对话框，ID 改为 IDD_SCANNER_DIALOG，Caption 为 scanner。进入对话框，加入图像扫描控件。有两种方法，分别如下所示：

方法一，在对话框中，按右键，将弹出一菜单项，在菜单项中选 Insert ActiveX Control，稍等片刻会弹出一个 ActiveX Control 的列表框，从列表框中选 Kodak 图像扫描控件，按 OK 键即完成了添加控件任务。

方法二，在主菜单选 Project/Add to Project/Components and Controls/Register Active Controls 之后，将列出许多控件，在控件中选出 Kodak 图像扫描控件，再单击 Insert 键，即完成添加任务。

此时在 ClassView 中可以看到新添加的 CImgScan 类，查看类的源文件 imgscan.cpp 可深刻了解控件的属性和使用方法。其中 Get(…)函数用来访问该属性的当前值，Set(…)函数则用来设置该属性的新值。

二、设置 Kodak 图像扫描控件

激活图像扫描控件的属性对话框，将其 ID 名改为 IDC_SCANCTRL2，其主要属性见

图 1-1。

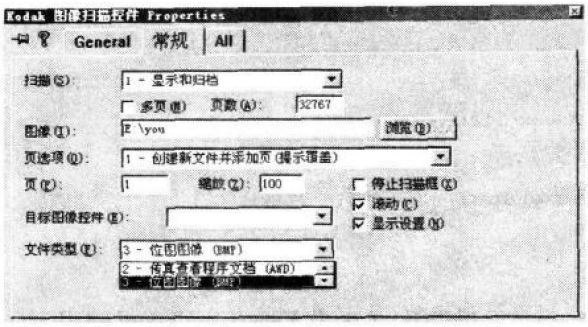


图 1-1

如图 1-1，常规属性中，其中扫描(S)：中有六个可选项，它们是：0—只有显示，1—显示和归档，2—只归档，3—显示并使用文件模板，4—只有使文件模板，5—只传真。

图像(I)选项为扫描文件归档存储路径和名称。注意如果计算机没有 E 盘，应改为相应的存储路径；本例 you 为扫描后存储的 BMP 文件名，以供处理或显示用。

页选项有 7 个可选项，包括：0—创建新文件并添加页，1—创建新文件并添加页(提示覆盖)，2—在现有文件中附加页，3—在现有文件中插入页，4—覆盖现有文件中的页，5—将页覆盖到现有文件中(提示覆盖)，6—覆盖所有页(提示覆盖文件)。

文件类型项目可选为 1—TIF 图像文档(TIFF)，2—传真查看程序文档(AWD)以及 3—图像(BMP)。

图 1-1 中值是本例所用值，实际使用中可根据需要灵活设定，更进一步的了解可查找有关帮助。

三、编制

1) 对类 CImgScan 实例化，在菜单项中选 View，进入 ClassWizard，在成员变量(Member Valiables)的 Control IDs 中选中 IDC_SCANCTL2，按 Add Variable)，添加变量 m_scaner，类型 CImgScan。

2) 在对话框添加开始扫描按钮控件，控件 ID 为 IDC_STARTS，加入如下所示代码，则按下该键将调用系统安装的扫描应用程序。

```
void CScannerDlg:: OnStarts()
{
    if(! m_scaner.ScannerAvailable())
      {
      AfxMessageBox("can't available scanner");
      exit(0);
      }
      else if(m_scaner.OpenScanner())
```

```
    {
        AfxMessageBox("can't open scanner");
        exit(0);
    }
    else  if(m_scaner.StartScan())
    {
AfxMessageBox("failed to start scanner");    exit(0);
    }
}
```

3) 加入扫描结束消息处理函数，从菜单 View/ClassWizard 选中 Message Maps 的 Object IDs：IDC_SCANCTRL2，再在右边选中 ScanDone，按 Add Function 添加消息处理函数，程序如下所示，按 Edit Code 进入程序可进行程序编写。本例中只加入一句提示语句，以便在扫描图像结束后标识捕获到了扫描结束消息。当然根据你的应用程序的需要可加入相应的处理函数。

```
BEGIN_EVENTSINK_MAP(CScannerDlg, CDialog)
    //{{AFX_EVENTSINK_MAP(CScannerDlg)
ON_EVENT(CScannerDlg, IDC_SCANCTRL2, 2/* ScanDone */, OnScanDoneScanctrl2, VTS_NONE)
    //}}AFX_EVENTSINK_MAP
END_EVENTSINK_MAP()
void CScannerDlg::OnScanDoneScanctrl2()
{
  AfxMessageBox("scanning finished");
// 该语句用于本例提示扫描结束消息捕获到，实际应用中去掉
//在此添加图像处理函数，其中图像文件为 E:\you
//在此添加网络发送函数或其他函数
}
```

4) 在对话框添加停止扫描按钮控件，控件 ID 为 IDC_STOPS，加入如下所示代码，则按下该键将退出应用程序。

```
void CScannerDlg::OnStops()
{
    m_scaner.StopScan();
    m_scaner.CloseScanner();
}
```

在编写处理图像或其他需要使用扫描图像的应用程序时，利用本文的方法加入扫描功能，再在扫描结束 ScanDone 消息处理函数中，加入处理子函数，可以使你的程序更加完美。

（游明星）

实例 2

也谈开发硬件中断虚拟驱动程序

中断，是 CPU 与计算机外部设备进行通信的有效手段。例如数据采集卡，ADC 转换结束，就发一个“转换结束”的硬件中断信号，要求 CPU 读取数据。硬件中断编程，在 DOS 操作系统下并不复杂。目前，计算机普遍由 DOS 系统转向了 Windows 98 系统，由于操作系统接管了硬件资源，使得一般的应用程序不能直接操作硬件资源，因此，开发 Windows 98 环境下的虚拟设备驱动程序(Virtual Device Driver：VxD)成了许多外部设备(如数据采集卡)开发中的主要工作之一，具有重要意义。

使用 VToolsD 软件包，结合 Visual C++，用C/C++ 语言可以高效快速地开发 VxD，编程者根据 VtoolsD 提供的范例，能够很快掌握实现硬件中断 VxD 的一般方法。所以本文不准备讨论 Windows 98 中断处理机制和编写硬件中断 VxD 的一般方法，需要的读者请看 VtoolsD 自带的帮助，笔者关注于 VxD 编程工作中遇到的一些实际问题。

一、VxD 与应用程序间通信

由于开发硬件中断 VxD 的目的在于控制硬件和提供硬件和应用程序的接口，所以实现 VxD 与应用程序之间的通信成了 VxD 开发中的关键。

1. 从 Win32 程序与 VxD 通信

一般的 Win32 应用程序工作在 ring3 级，而 VxD 作为操作系统的扩展工作于 ring0 级，要实现 ring3 级到 ring0 级的通信，首先必须获得所要控制的 VxD 句柄，这可以使用一个 Win32 API 函数 CreateFile。CreateFile 用于获取一个静态或动态 VxD 的句柄。对于动态 VxD 而言，CreateFile 首先会装载它(如果没有被装载的话)，如果装载成功，操作系统将会发出 SYS_DYNAMIC_DEVICE_INT 消息，通知 VxD 进行动态初始化，而后获得它的句柄。

Win32 API 函数 DeviceIOControl 是 Win32 程序与 VxD 通信的关键。任何时候，在 Win32 程序中调用 DeviceIOControl，相应的 VxD 就会收到系统发来的 W32DEVICEIOCONTROL 消息，VtoolsD 在生成的框架中用 OnW32DeviceControl 函数响应这个消息：OnW32DeviceControl 函数只有一个 IOCTRPARAMS 结构的参数，该结构中的 dioc_IOCtrlCode 成员对应于 DeviceIOControl 函数中的 dwIoControlCode 参数(Win32 程序向 VxD 传递的消息号)；编程者通过重载这个函数，使 VxD 根据不同的消息号作出相应的响应，就可以实现 Win32 App 到 VxD 的通信。

下面是一个例子，Win32 App 通过 VxD 获得 ADC 转换的数据。

在 Win32 App 中：

```
#define PASS_DATA  (0x8000 + 1) //定义数据传输的消息号
……
hVxD = CreateFile("\\\\.\\adc.vxd",0,0,0,CREATE_NEW,
FILE_FLAG_DELETE_ON_CLOSE,0);//动态加载 VxD 并获取 VxD 句柄
DeviceIOControl(hVxD,PASS_DATA,(LPVOID)ibuf,sizeof(ibuf),(LPVOID)obuf,sizeof(obuf),
&cbReturned,NULL);
//与 adc.vxd 的事件过程 OnW32DeviceIoControl 交换数据
//ibuf 是 Win32 程序传递给 VxD 的数据缓冲地址,如果不需要可以给 NULL
//obuf 是 VxD 返回数据所存放的缓冲地址
//VxD 传回的实际数据总量放在 cbReturned 中
CloseHandle(hVxD); //卸载 VxD
```

在 VxD 中：

```
#define PASS_DATA  (0x8000 + 1)
DWORD ADCDevice :: OnW32DeviceIoControl (PIOCTLPARAMS pDIOCParams)
{
      switch(pDIOCParams->dioc_IOCtlCode)
      {
……
            case  PASS_DATA:
memcpy(pDIOCParams -> dioc_OutBuf,adc_data,4096);
//每次从 VxD 向 App 传输 4096 个(4K)字节
return 0;
            ……
}
return 0;
}
```

值得注意的是，为了避免和 Visual C 编译系统中一些已定义的变量发生冲突，VtoolsD 建议自定义的消息号大于 32768(0x8000)。

2. 从 VxD 与 Win32 程序通信

从 VxD 与 Win32 程序通信最简单的办法就是使用 SHELL_PostMessage 服务。SHELL_PostMessage 服务可以用来从 VxD 中向 Win32 应用程序发送消息，其定义为：

BOOL SHELL_PostMessage(HANDLE hWnd, DWORD uMsg, WORD wParam, DWORD lParam, PPostMessage_HANDLER pCallback, PVOID refData)

其中 hWnd 是处理该消息的 Win32 程序窗口句柄，在使用 SHELL 服务前必须将此消息处理窗口句柄传给 VxD（可以使用从 Win32 App 到 VxD 通信的方法，即 DeviceIoControl 机制来传递）。uMsg 是自定义的消息号，pCallback，refData 用于回调函数，如果不需要，全部设置为

NULL。

例如上例中 VxD 与 Win32 App 数据传输可以采取这样一种思路：每采集 4K 数据，在 VxD 里用 SHELL_postMessage()传递消息给 Win32 App，例如：

```
SHELL_PostMessage(appWnd, WM_BUFFER_FULL, 0, 0, 0, NULL);
```

/*WM_BUFFER_FULL 是自定义消息，appWnd 是消息接受窗口句柄，Win32 App 相应的窗口接到消息后，就可以用 DeviceIOControl 来取数*/

值得注意的是，SHELL_PostMessage 服务不能用于 Win32 控制台程序(Win32 Console Application)，也不能在中断处理时调用；另外，由于 SHELL_PostMessage 发送给应用程序的消息可能因处于线程的消息循环中造成一定的延时，所以建议创建一个专门的线程等待该消息。

二、动态虚拟化中断号

VtoolsD 提供 VhardwareInt 硬件中断类处理硬件中断，该类构造函数的第一个参数就是要虚拟化的中断号，在许多示例中都是采用一个事先确定的中断号，但也可以将这个参数改为一个变量，由应用程序决定要虚拟哪一个未用的中断号。具体做法：注意不在 OnSysDynamicDeviceInit()里建立中断类实例，应该使用 DeviceIoControl 机制发个自定义消息给 VxD，通知要虚拟的中断号，并建立中断类实例。

```
DWORD ADCDevice:: OnW32DeviceIoControl(PIOCTLPARAMS pDIOCParams)
{
      switch(pDIOCParams->dioc_IOCtlCode)
      {
case INITIRQ:   //动态虚拟中断号消息
  IRQNumbear =  *(int *)pDIOCParams->dioc_InBuf; //获得要虚拟的中断号
pMyIRQ = new MyHwInt(); //建立硬件中断类实例
if(pMyIRQ  &&  pMyIRQ->hook()) //挂钩中断号
{
      ......
}
......
```

三、使用 SoftICE 调试 VxD

VxD 编好以后，一般都要经过调试。SoftICE 是一个核心调试器，是调试 VxD 的利器。SoftICE 甚至和 VtoolsD 集成到 NuMega DriverStudio 中一起发行，成为开发驱动程序的首选工具。一般资料上甚少介绍用 SoftICE 调试 VxD 的具体方法，本文把我们的经验介绍给大家。其实，使用 SoftICE 调试 VxD 只要遵循一定的顺序，一步一步地进行就可以了：

1) 用 Visual C 创建 VxD。

2) 使用 SoftICE 的 Loader 加载 VxD 文件。分为两步：第一步用 Open 打开一个 VxD 文件，注意，如果用 Visual C 生成 VxD，会同时生成一个.PDB 文件，该文件包涵了 VxD 的源文件和调试信息，所以要确保 PDB 文件和 VxD 文件在同一目录下(或者在 Loader 可以搜寻到的目

录下)；第二步单击 Load 按键，Loader 将自动把 .PDB 转换成 .NMS 调试符号文件，并加载调试符号文件和源文件。

3) 加载完成后，就可以开始调试 VxD 了。首先用 Ctrl+D 呼出 SoftICE，用“file *”看已经加载的源文件，用“file+文件名”选择要调试的文件，就可以看到 VxD 的源代码。用 SRC 命令使显示在源代码和汇编代码间切换。

4) 使用 BPX 命令设置断点，对应代码高亮表示断点设置成功。VxD 运行到断点处会自动呼出 SoftICE，停到断点处。

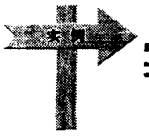
5) 启动调用 VxD 的 Win32 应用程序。SoftICE 其他几个常用命令：BC—清除断点；O—往端口写一个字节；I—从端口读入一个字节；GENINT—产生一个中断；T—Trace 跟踪执行指令；G—执行程序；EXIT—退出。

四、关于实时控制的问题

众所周知，Windows 95/98 系统并不是一个实时操作系统，它通过 VPICD(Virtual Programmable Interrupt Controller Device)管理所有硬件中断事件。我们编写的硬件中断 VxD 从 VPICD 手中获得中断的处理权。整个 Windows 硬件中断响应过程比 DOS 复杂得多，Windows 的 ISR(Interrtupt Service Routine)时间可以是 DOS 的 ISR 的 20 倍。不过通过编写合适的 VxD，我们还是可以获得比较满意的准实时效果的。有资料表明，一个中断 VxD 在 486 /66 计算机上能处理频率为 10KHz 的中断信息，不会漏掉一个中断。另外在处理硬件中断时，最好不要运行其他软件。

以上技术，已在笔者编写的 ISA 多道脉冲幅度分析器虚拟驱动程序中实现。编译和调试工具：VtoolsD3.0、Visual 6.0、SoftICE4.05 for Windows 9X。

(张雄飞　刘　平)



实例 3

利用 DirectX 实现对游戏操纵杆的编程

DirectX 是 Microsoft 的提供的一组编程接口，开发者可以轻而易举地创建包括高性能的平面和三维图形、声音混合与倒播，及 Internet 多媒体播放支持体系的多种应用程序。DirectX 编程技术为硬件制造商、应用程序开发商和用户之间提供了一个一致的接口。从而在充分利用现有硬件特性的基础上，减少了安装和设置的复杂性。基于 DirectX 的应用程序和游戏，在得到了利用硬件加速的特性以外，还获得了 Windows 平台的易操作性和通信联网能力。DirectX 包含一系列组件，这些组件都是基于 COM(组件对象模型)的编程接口，主要有以下五个部分：

1)DirectDraw 主要用于二维图形和动画的显示。

2)DirectSound 是声响合成和声卡的接口，用于数字音频的合成与回放。

3)DirectPlay 是一组开发多用户游戏的接口。

4)Direct3D 提供了一组完全的 3D 图形系统接口。

5)DirectInput 是一组支持基于 Windows 硬件输入的 API 和驱动程序。

由于以上的特点，DirectX 逐渐成为 Windows 环境下开发游戏和其他高性能图形程序的首选工具，在军事、国防、仿真、虚拟现实等许多领域都有应用。本文主要是利用其 DirectInput 部分实现在飞行模拟程序中用游戏操纵杆对飞机的飞行控制。

一、DirectInput 简介

1. 支持设备

一般说来，DirectInput 支持与 PC 机相连的任何输入设备，当然不包括那些没有 DirectInput 驱动程序的设备。通常我们常用的三种设备是：键盘、鼠标和游戏操纵杆。

2. 性能

DirectInput 方式胜过传统的 Windows 机制，比传统的 Windows 机制快。它可以在保持设备独立性的前提下直接访问设备，从而避开了 Windows。

3. 输入数据

根据应用程序的需要，DirectInput 可用于检索两种形式的输入数据：即时存取数据和缓冲存取数据。即时存取数据可描述在需要数据的时刻输入设备的状态。而缓冲存取方式利用一个队列来描述输入设备的状态变化，例如激活按钮和坐标变化就分别对应于上述的两种不同方式。