设备驱动程序编程
——从 DOS、Windows3.X 到 Windows95

薛宏涛 沈林成 常文森 (国防科技大学自动控制系 410073)
杨 光 (长沙铁道学院土木建筑学院 410075)

摘要:本文简要分析了 DOS、Windows3.X 及 Windows95 下设备驱动程序编制的原理,提出了一种在 Windows95 下设备驱动程序的编制方法。

关键词:设备驱动程序 中断服务程序 多线程编程 Windows95 串口通信程序

一、引言

现在,Windows95 以其方便友好的界面和强大的功能已逐渐普及并取代了原来的 DOS 及 Windows3.X,但原来的许多硬件设备驱动程序都是基于 MS-DOS 及 Windows3.X 编制的,这些设备驱动程序许多不能直接在 Windows95 下运行。而 Windows95 下设备驱动程序的编制方法及风格与 MS-DOS 及 Windows3.X 有很大的不同,因此设备驱动程序的编制要从 DOS、Windows3.X 的编程模式向 Windows95 转化。

二、DOS、Windows3.X 下的设备驱动程序编制

MS-DOS 作为一个单任务环境,每次通常只允许一个程序运行,但 TSR 能够加载并驻留在内存中,对其他程序提供服务。这个特性使得 TSR 十分适合提供设备驱动支持,因此,大量的 MS-DOS 程序依靠 TSR 作为设备驱动程序。Windows3.X 没有建立对于文件 I/O 的支持,它依靠 DOS 读出和写入数据,因而,Windows3.X 程序也能访问 MS-DOS TSR,故 Windows3.X 下的设备驱动程序编制和 DOS 下的方法基本上是相同的。

DOS、Windows3.X 下的设备驱动程序通过直接使用 BIOS 或 DOS 系统的基本调用,改变系统的中断向量表,从而在系统接到中断时转去执行用户编写的中断处理程序,由系统作保存现场、恢复断点运行等整个中断机制的维护。其设备驱动程序编制比较简单,操作要点如下:

·保护好原中断向量。Borland C++ 的 getect()函数可以把这个 far 指针存在函数的参数中;

·将自己的中断服务程序的地址放到中断向量表中,setvect()函数可完成;

·在程序退出系统之前,恢复原来的中断向量。再次使用 setvect()函数,用前面保存的原中断向量为参数;

·使用 DOS 系统调用允许产生中断。如用 int86();

·编制中断处理程序。通过 inport()、outport()函数直接对端口进行读、写操作。

三、Windows95 下设备驱动程序编制

Windows95 的文件系统完全抛弃了 MS-DOS 系统中断调用这一机械的方法,不再依赖于 MS-DOS,采用了层次化的 32 位文件系统设计。从程序员编程角度看,只需使用 WIN32 API,比 DOS 中断方式编程更安全,易于移植及富于健壮性等,但 Windows95 下的编程复杂程度也大大增加了,下面介绍一种在 Windows95 下设备驱动程序编制的思想及实现方法。

利用 Windows95 支持的多线程机制,可以模仿 DOS 下中断服务程序的编制思想,在内存维护一个"硬件监视线程",相当于 DOS 下潜伏于内存中的"中断服务程序"。由主线程对硬件监视线程进行创建、启动、撤消等维护工作,其要点是:程序一开始先由主线程作一些必要的初始化工作,然后建立及启动硬件监视线程,由硬件监视线程监视硬件的情况。当发生硬件中断事件时,硬件监视线程会检测到中断事件,然后通过消息传递的方法报告"主线程",使主线程处理消息。之后,若主线程不再需要接收消息了,则撤销硬件监视线程。如图 1 所示。

这就涉及到了多线程的同步问题,同步是为了避免多线程对共享资源的访问冲突而采取的协调多线程运行的方法。WIN32 提供了多种同步方法,如等待函数、事件、互斥、信号、临界区等同步对象以及复用 I/O 技术。

这里只介绍等待函数、事件对象和复用 I/O 技术，其他方法参见[2]。

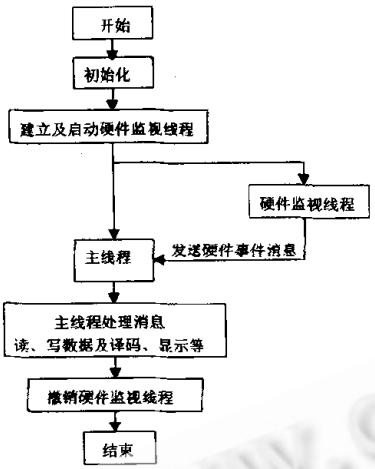

图 1

1. 等待函数

WIN32 API 使用等待函数来等待一个可等待对象(如事件、互斥、信号)变为信号状态。如 WaitForSingleObject()，若等待对象没有信号，则调用线程进入等待状态，直到等待条件满足则返回，调用线程继续执行。

2. 事件对象

可以通过 SetEvent()、ResetEvent()设置、复位该事件信号状态的一种对象，用来协调多线程的执行。用 CreateEvent()、CloseHandle()建立和销毁事件对象。

3. 复用 I/O

若使用 FILE-FLAG-OVERLAPPED 标志创建硬件设备，则该设备的调用函数采用复用 I/O 技术。采用复用 I/O 技术的函数在它等待费时的操作完成之前可以立即返回，即使操作还没有完成。

四、程序示例

这里以 SUMMAGRID III(简称 SG3)数字化仪为例，分别编写了 DOS(或 Windows3.X)及 Windows95 下的设备驱动程序。SG3 数字化仪连到 COM2，缺省的地址为 2F8H，中断号 IRQ 为 3，该数字化仪以每帧 8 个字节向计算机报告数据，其中第一个字节为状态字节，其状态位为"1"有效，后七个字节为数据，其状态位为"0"有效。

DOS 及 Windows3.X 下的程序在 Borland C++3.1 下编译通过，Windows95 下的程序在 Visual C++5.0 下编译通过，两个程序分别已在 Windows3.1 和 Windows95 下的"地形导航地理信息系统"中使用。

五、Windows95 下调试程序遇到的问题及解决办法

在程序的实现过程中，遇到的第一个问题就是不能正确打开串口，即 CreateFile()函数返回一个无效的设备句柄，而此时检查"控制面板"的"系统"中串 2(COM2)的设备运转情况是正常的(Windows95 并没有报告任何硬件冲突)，这就引起了疑问，令人不解。后来想到，每一个 WIN32 函数在调用失败后均调用 SetError()函数返回一个错误码供查错参考，于是在 CreateFile()函数失败后调用 GetError()函数检查错误码，经跟踪调试发现错误码为 31，查系统文档知，错误码 31 代表"连到系统的一个设备没有正常工作"，但为什么 COM2 没有正常工作呢？检查了硬件连线没有问题，而且在 Windows95 的虚拟 DOS 环境下运行数字化仪的检测程序(数字化仪自带的)是正常的，能够读出及跟踪游标的位置坐标。后来想到是不是 CMOS 参数在作怪，果然，查到 CMOS 中的通信口 2 的设置是 2F8/IRQ3，这与网卡的中断号 3 是冲突的，修改了 CMOS 设置，把通信口 2 的设置改为 2E8/IRQ10，再将"控制面板"的"系统"中通信口 2 的设置也改为 2E8/IRQ10，重新启动后则通信口 2 工作正常(网络也能正常工作)，此时通信口 2 的代号改为 COM4(或者修改网卡的中断号 3 为其他不与其他硬件发生冲突的中断号也可)。

(注：原文附有程序实例，篇幅太长予以删略，有兴趣的读者可与作者联系。)

参考文献

[1] 欣禹，李莉等译，Microsoft Win32 程序员参考大全(二)，1995.4

[2] 方旭等编，Borland C++3.1 实用大全，1994.8，北京航空航天大学出版社

[3] 王齐，"Windows95 中的串行通信"，微电脑世界，1997.3

(来稿时间：1998 年 4 月)