第 24 卷第 5 期　　合肥工业大学学报(自然科学版)　　Vol. 24 No. 5
2001 年 10 月　　JOURNAL OF HEFEI UNIVERSITY OF TECHNOLOGY　　Oct. 2001

基于 Win95/98 实时控制系统快速响应中断的方法

周国祥,　郑兆全,　石　雷
(合肥工业大学 计算机与信息学院,安徽 合肥　230009)

摘　要: 在计算机工业控制领域,越来越多的基于 DOS 的实时控制系统和检测系统正在向 Windows 上移植,实现对中断的快速响应是实时控制系统和检测系统最基本的要求。文章介绍在 windows95/98 环境下,采用工具 VtoolsD 在 C++ 编程环境下编写 VxD(虚拟设备驱动)的一种简便有效的方法来实现对硬件中断的快速响应。

关键词: 实时控制系统; 中断; VxD; VtoolsD
中图分类号: TP273　　文献标识码: A　　文章编号: 1003-5060(2001)05-1006-04

A simple method of realizing quick response to interrupt in real-time control system based on Win95/98

ZHOU Guo-xiang,　ZHENG Zhao-quan,　SHI Lei
(School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China)

Abstract: In the control field of computer industry, more and more real-time control systems and test systems based on DOS is transplanted to Windows. A quick response to interrupt is the most basic requirement in real-time control systems and test systems. In this paper, a simple and efficient methods is introduced, which uses a tool VtoolsD and Vsiual C++ to program VxD (Virtual Device Driver), then a rapid response to hardware interrupt can be realized in the environment of Windows 95/98.

Key words: real-time control system; interrupt; VxD; VtoolsD

0　引　　言

在实时控制系统中一般采用中断方式来实现对外部信号的采集或向外部发出控制信号。在DOS 环境下,开发人员需要掌握相关硬件方面的知识,在中断向量表中写入中断服务程序的入口地址,用汇编语言进行简单编程即可在系统中实现中断功能。进入Windows 时代,该操作系统具有界面友好、资源丰富及操作方便等优点,越来越多的基于 DOS 的实时控制软件在向其移植或在其环境下开发。由于 Windows 系统本身不是为实时控制而设计,为了使系统稳定,更好的支持多任务,应用程序不能完全独占 CPU 资源,中断控制器被软件虚拟化。在 Windows 平台下,硬件中断均由 VPICD(Virtual Programmable Interrupt Controller Device)来管理(在本文中只讨论硬件中断),当中断发生时, VPICD

收稿日期: 2001-05-10
作者简介: 周国祥(1956-),男,安徽合肥人,合肥工业大学讲师.

将触发 VM 中默认的中断处理函数。用 VxD(Virtual Device Deriver)来重载中断处理函数[1,2]。对 VxD 编程不仅需要掌握汇编语言知识和相关硬件方面的知识, 还要对 Windows 内部机制有很深的了解, 这就给软件开发人员带来很多困难, 且要做许多工作。为此, 这里介绍一种简便有效的实现快速响应中断的方法, 利用工具 VtoolsD 用 C++ 编写 VxD, 而 VtoolsD 可以通过 www.vckbase.com 中下载获取。

1　实现中断响应的方法

QuickVxD 帮助你用 VtoolsD 封装的 C++ 类库来创建 VxD 程序的框架。打开 QuickVxD, 选择选项卡 Device Parameters, 在文本框 Device Name 处填写驱动硬件的名(实际上就是生成的文件名), 选择 Special Support 处的 Dynamically Loadable 以便随时加载 VxD。中断响应的具体方法为:

(1) 需要在程序中做一些硬件上的初始化和事后处理, 故在选项卡 Windows95 Control Messages 处, 选中:

SYS_DYNAMIC_DEVICE_INIT

SYS_DYNAMIC_DEVICE_EXIT

这时, 就可以生成目标文件了。

(2) 打开 VC++ 及 sample.mak(本文假设在 Device Name 处填写的是 sample), 一直按默认的选项打开, 再把 sample.cpp 添加入工程, 再打开 sample.h, 但不加入工程。将 Project Setting 中的 Output file name 和 Executable for debug session 中的*.exe 改为*.vxd。

(3) 在 VtoolsD 中, 处理中断的类是: VhardwareInt, 若要处理中断, 必须将其重载。先在 sample.h 中加入定义:

```
# define SAMPLE_IRQ          8//你的中断号, 这里设为 8
class SampleInt: public VHardware
{
public:
    SampleInt(): VHardwareInt(MY_IRQ, VPICD_OPT_CAN_SHARE, 0, 0){}
    virtual VOID OnHardwareInt(VMHANDLE);
};
```

要注意大小写, 否则会加载失败。在SampleDevice 中加入公共变量, 其类型就是预先定义的中断类的指针。如: pIRQ.

(4) 在 sample.cpp 中需要加入 3 个函数; 即 Sample::OnHardwareInt(VMHANDLE)、SampleDevice::OnSysDynamicDeviceInit()和 SampleDevice::OnSysDynamicDeviceExit()。当中断发生时函数 OnHardwareInt(VMHANDLE)就会响应, 若不重载, 则加载 VxD 时会失败。在 OnHardwareInt(VMHANDLE)结束前, 要加上 sendPhysicalEOI(), 以表示这次中断处理结束。

(5) 在重载 SampleDevice::OnSysDynamicDeviceInit()时, 要实现对中断的虚拟化。其步骤如下:

```
BOOL SampleDevice::OnSysDynamicDeviceInit()
{
    pIRQ = new SampleInt();
    if(pIRQ && pIRQ→hook())      //hook()为 VhardwareInt 类的成员函数, 用来实现//中断
                                 的虚拟化
{
    ……                          //其它的初始化代码
    pIRQ→physicalUnmask();       //将中断设为不屏蔽
```

```
    return TRUE;
}
else
    return FALSE;
}
```

在重载 SampleDevice::OnSysDynamicDeviceExit()时,要释放指针 pIRQ。

(6) 在应用程序中加载时,使用:

HANDLE hvxd = CreateFile("\\\\.\\sample.vxd",0,0,0, CREATE_NEW, FILE_FLAG_DELETE_ON_CLOSE,0);

当使用 VxD 结束时,必须将其释放,使用:

CloseHandle(hvxd);

框架完成之后,只需在必要的地方填写 C++ 代码即可。

2 应用程序和 VxD 的数据交换

在应用程序中常常需要和 VxD 交换数据并进行控制,这时可以利用 API 函数设备输入输出控制 DeviceIoControl 来解决问题,定义如下:

BOOL DeviceIoControl(HANDLE hDevice, DWORD dwIoControlCode, LPVOID lpvInBuffer, DWORD cbInbuffer, LPVOID lpvOutBuffer, DWORD cbOutBuffer, LPDWORD lpcbByteReturned, LPOVERLAPPED lpOverlapped)[3];

其中,hDevice 为应用程序中 VxD 的句柄,dwIoControlCode 为命令码,是用来传递到 VxD 中进行判断 VxD 应该做什么事的一个标志,lpvInBuffer 为应用程序传递给 VxD 数据的地址,cbInbuffer 为前面传递的数据的字节数,lpvOutBuffer 为 VxD 传递给应用程序的数据的地址,cbOutBuffer 为前者的字节数,lpcbByteReturnedwei,lpOverlapped 设置为 NULL 便可。作为对应,VtoolsD 中可以通过重载函数 SampleDevice::OnW32DeviceIOControl(PIOCTLPARAMS pDIOCParams){}处理。pDIOParams 用来传递相应的参数,定义如下:

```
typedef struct tagIOCTLParams[4,5]
{
    PCLIENT_STRUCT          dioc_pcrs;
    VMHANDLE                dioc_hvm;
    DWORD                   dioc_VxdDDB;
    DWORD                   dioc_IOCtlCode;
    PVOID                   dioc_InBuf;
    DWORD                   dioc_cbInBuf;
    PVOID                   dioc_OutBuf;
    DWORD                   dioc_cbOutBuf;
    PDWORD                  dioc_bytesret;
    OVERLAPPED*             dioc_ovrlp;
    DWORD                   dioc_hDevice;
    DWORD                   dioc_ppdb;
}
IOCTLPARAMS, * PIOCTLPARAMS;
```

这样就可以通过 switch(pDIOParams→dioc_ IOCtlCode) 来决定下一步的操作, 进行数据交换以及间接控制, 以便完成指定的任务。

3　应用实例

LMT 石油专用管二维漏磁高速探伤机检测系统, 是用来检测油管内部缺损情况的。该系统软件要完成一个重要工作就是及时采样检测结果信号, 并在屏幕上用波形来直观反应油管相对位置的缺陷及缺损程度。由于探伤速度比较快, 标记缺陷要求精确, 所以对采样的频率要求比较高(高达 8 Hz) , 而且探伤中途不能暂停。显然, 在 Windows 下用查询方式无法满足系统的要求。为此, 在系统检测程序中采用中断方式进行实时采样, 而中断响应正是利用 VxD 来实现的。检测系统主程序界面, 如图 1 所示。

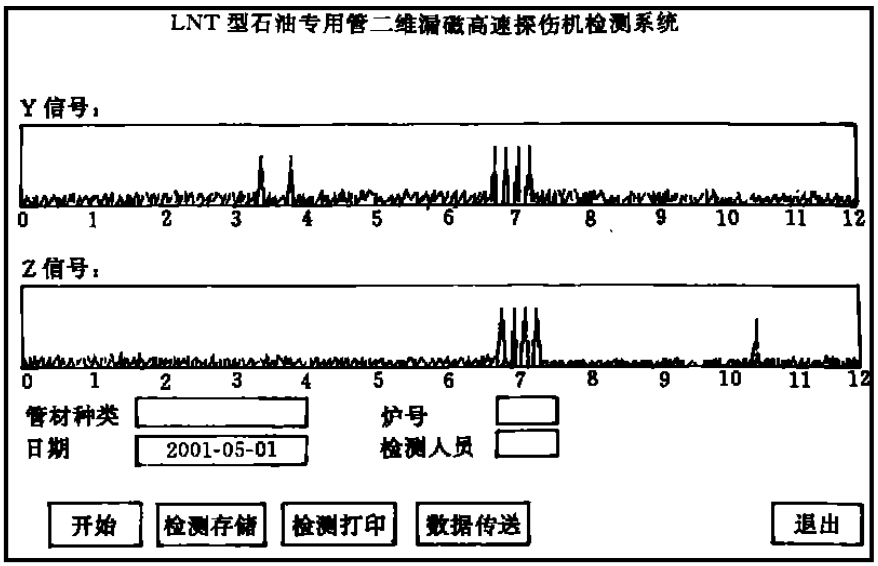

图 1　检测系统主程序界面图

系统运行表明, 中断实时采样完全满足了系统设计要求。

4　结束语

借助工具 VtoolsD, 用 C++ 编写 VxD 来实现对硬件中断的快速响应。实践表明该方法是完全可行的, 它使开发人员既避开了汇编语言和相关硬件方面的知识, 也无需对 Windows 内部机制有较深的了解, 从而为在资源丰富、操作友好的 Windows 下开发实时控制软件提供了方便, 并节省了时间。

[参　考　文　献]

[1]　孙守阁, 徐　勇. Windows 设备驱动程序技术内幕[M]. 北京: 清华大学出版社, 2000. 243- 261.
[2]　Scot W G S. Visual C++ 6.0 技术内幕[M]. 希望图书创作室译. 北京: 希望电子出版社, 1999. 673- 682.
[3]　Microsoft C. Microsoft developer network visual studio 6.0[M]. New York: Microsoft, 1999. 226- 234.
[4]　Vireo Software. VtoolsD Help[M]. New York: Vireo Software, 1998. 171- 177.
[5]　陈　坚. 实用 Visual C++ 编程大全[M]. 西安: 西安电子科技大学出版社, 2000. 450- 458.

(责任编辑　张秋娟)