Visual Basic 实时应用系统的实现方法

沈工　唐飞　毛永明　（哈尔滨工业大学　150001）

摘要：本文提出一种利用动态链接库(DLL)来及时处理硬件中断，从而实现 VB 应用程序对实时任务处理的方法。详细讨论了 VB 应用程序与 DLL 之间如何进行通信和数据交换的问题以及用 Visual C++ 编写 Windows 中断程序的方法。

关键词：VB　VC++　消息　动态链接库　中断

一、前言

关于如何进行 Windows 中断程序设计，已有一些文章发表，但都是用 BC++ 编写的。虽然 BC++ 也可以编写面向对象的 Windows 应用程序，但开发时间较长。

因此本文提出，用 VB 来完成系统的绝大部分功能，用 VC++ 编写中断程序并做成一个动态链接库(DLL)作为内核来及时处理硬件中断。该方法可以节约大量开发时间。要使该方法能够成功，必须解决 VB 应用程序与中断处理程序之间的通信和数据交换问题，本文对此进行了详细讨论，提出的方法具有普遍意义。由于用 BC++ 编写的中断程序不能移植到 VC++ 中，为此给出了如何用 VC++ 编写 Windows 中断程序的具体方法。

二、WINDOWS 的消息运行机制

Windows 是一个消息驱动式系统。Windows 对每一个输入事件都产生一个消息，并把这些消息收集在一个系统队列中，然后将它们连同其他一些消息一起放入一个应用程序队列中。应用程序队列是属于各个应用程序所有的先进先出队列，这就使得一些外部实时事件可能得不到及时处理。应用程序通过 GetMessage 函数来读取这些消息，并用 DispatchMessage 函数将它们传送给相应的窗口函数，其过程如图 1。

值得一提的是，虽然大多数消息都由 Windows 产生，但应用程序也可以产生自己的消息，并将它们放入其他应用程序的队列中。这就使得应用程序间的通信成为可能。

三、WINDOWS 环境下硬件中断原理及用 VC++ 的编程实现

从上面的分析可知，为了满足实时性的要求，就应该适时地避开 Windows 的消息循环而直接对中断进行处理。加入用户中断源后的 Windows 消息循环如图 2。实际应用证明，图 2 所示的消息循环是可行的，既能满足实时中断处理的要求，又不影响 Windows 系统的正常工作。考虑到不要让中断处理程序占用过多的 CPU 时间，仅把最紧迫的任务放在中断处理程序中处理，而把不很紧急的任务通过 PostMessage 函数向 VB 应用程序发送相应的消息，由 VB 应用程序处理。

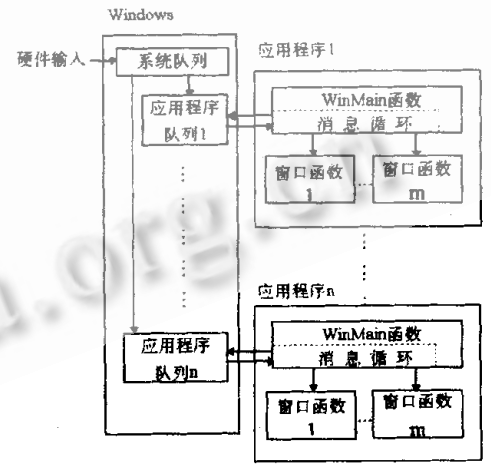

图 1

在 Windows 下的实中断实现模式和 DOS 环境下基本一致，具体过程可参见程序源代码。为提高编码效率，有些功能采用在 VC++ 语言中嵌入汇编语言的方式。

VC++ 虽然功能强大，但在编写 Windows 中断程序时却遇到了困难。这在其他语言(如 BC++)中也许是碰不到的。

编写中断程序最重要的一点是要获取及设置中断向量。常见的做法是用汇编语言中的数值回送操作符 SEG 和 OFFSET 来获取其段地址和偏移地址。但 VC++不接受 SEG 操作符。鉴于此，我们使用了 VC++1.5 中的 dos-setvect 函数来设置中断向量。dos-setvect 函数原型如下：

void--cdecl -dos-setvect(unsigned intnum, void(--cdecl --interrupt --far * handler)());

其中，intnum 为中断号，后一个参数为中断处理函数名。

当我们使用这个函数时，VC++编译器告诉我们，它不认识这个函数。打开包含这个函数的头文件 dos.h，我们会发现是这个函数原型前后的两条宏语句在作怪：#ifndef-WINDOWS、#endif。这两条语句说明这个函数不能应用在 Windows 应用程序中。但在这个函数的帮助中，微软公司明白地告诉我们，这个函数可应用于 Windows 应用程序。于是我们做了以下工作：

1.在 dos.h 文件中，把-dos -setvect 函数原型前后的 #ifndef-WINDOWS 和 #endif 用注释符(\)屏蔽掉。正是这两条指令使得该函数不能在 Windows 应用程序中使用。

2.将 llibce.lib 库加入到 linker Options 对话框中的 Libraries 框中。-dos-setvect 函数就在该库中，但在编译 Windows 应用程序的缺省状态下，是不连接该库的。

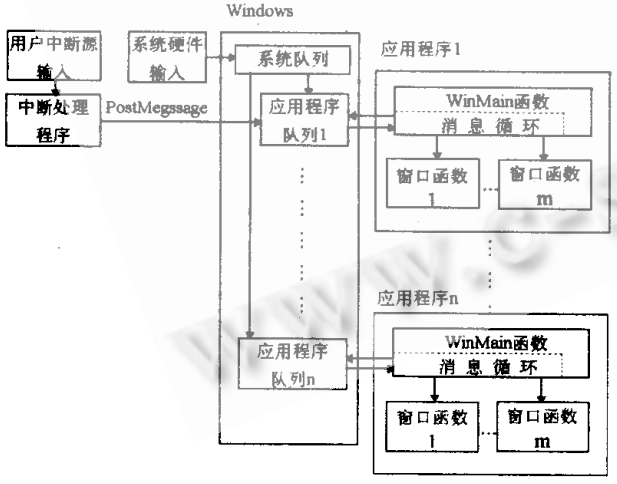

图 2

四、动态链接库

从某个角度看，动态链接库(DLL)只不过是运行程序内一群具备特定功能的子程序的组合。在一般情况下，编写程序时，无论所调用的函数来自一般的库还是 DLL，调用的方法并没有什么差别。

和一般的应用程序一样，DLL 中可以放置许多函数。每个函数都必须声明为远程调用(FAR)。DLL 的程序源代码中必须引入 windows.h 这个文件，借以辩别一些常量和数据类型的定义，并且读取其他有用信息。VC++的 PASCAL 保留字依然是用来定义参量传递和堆栈清除时所必须使用的协议，对于 DLL 而言，加上 PASCAL 以后，编译后的目标代码不仅运行起来比较快，同时也会比较小。

一个用 VC++编写的 DLL 主要包括以下几个部分：LibMain 函数、WEP 函数和用户自定义函数。其中 LibMain 函数是用来初始化 DLL 的，当 DLL 首次装入时，这个函数被调用；当所有调用这个 DLL 的程序结束或要释放时，Windows 就会调用 WEP 函数，看看是否必须进行某些必要的结束工作。对一般应用程序而言，这两个函数的格式大体相同。在后面的程序源代码中给出了这两个函数，读者可以直接引用，不必作任何修改。

DLL 的模块定义文件(.DLL)与通常的应用程序基本上是相同的。但要提醒注意的是，DLL 的模块定义文件中不应出现 STACKSIZE 这行语句，因为 DLL 没有自己的堆栈，而是使用调用它的应用程序的堆栈作为暂时的栖身之所。另外由于编写的是中断程序，因此 CODE 和 DATA 语句后的选择项最好为 PRELOADFIXED。

头文件(.h)中放的是用户自定义函数的函数原型。

用 VC++工作平台生成 DLL 是很方便的，只需在 New Project 对话框的 Project Type 框中选择 Windows Dynamic+Link Library (.DLL)标题，则编译后就会生成动态链接库(.DLL)。具体做法在许多 VC++书上都有介绍，读者可以参考。

五、VB 应用程序与动态链接库的通信和数据交换

VB 应用程序在调用 DLL 函数前首先要对其进行声明。DLL 函数或过程必须在全局模块或窗体(Form)层中声明才有效。当 DLL 函数或过程没有返回值时要声明为子程序 Sub 格式，若有返回值时应声明为函数 Function 格式。具体格式如下：

子程序 Sub 格式

Declare Sub DLL 过程名 Lib "DLL 库名"[Alias "别名"][(参数)]

函数 Function 格式

Declare Function DLL 函数名 Lib "DLL 库名"[Alias "别名"][(参数)]As 函数数据类型

如果已将生成的.DLL 文件拷贝到/windows/system 目录下,那么"DLL 库名"项中只要直接说明库名即可;如果.DLL 文件不在上述目录中,则必须指明.DLL 的全路径。

本程序通过一个全局数组 data-exchang()在 VB 应用程序和 DLL 之间传递数据。实现时只要把数组的首地址传递给 DLL 过程或函数即可。传递数组的首地址实际上是将数组中首元素以地址方式(非 Byval 方式)传递参数。具体实现可参见程序源代码。

从图 2 可以看到,当中断处理程序运行结束时,要通过 PostMessage 函数向 VB 应用程序队列发送一条消息,通过 VB 应用程序读走数据交换数组中的数据,并对一些不很紧急的任务进行处理。这在 C 语言中可以通过自定义消息来实现。但 VB 不能自定义消息,那么怎么来实现这个功能呢?先来简单分析一下 VB 的运行情况及事件(Event)、事件过程(Event Procedure)的实质。

既然 VB 应用程序是运行在 Windows 环境下的,那么必然也是靠消息来驱动的,只不过 VB 的语言设计者们已经把这些 Windows 系统定义的消息对应给了一个 VB 事件(Event),VB 程序员所需做的工作就是给这些事件附加代码,因而事件过程(Event Procedure)实质上就是消息处理函数。

前面已经讲了,VB 不能自定义消息,因而只能利用 Windows 系统定义的消息来完成我们的工作。在对 Windows 系统定义的消息进行分析后,决定选用 WM—MBUTTONDOWN 和 WM—MBUTTONUP 两条消息。当用户按下和松开鼠标中键时,分别发生上述两条消息。之所以选择这两条消息,是考虑到现在常见的鼠标都是双键或单键的,即使是三键鼠标,其驱动程序大多数也不支持中键,也就是说在一般使用中,中键实质上是多余的,不会因用户而发出它们。

在 VB 中选择 Form-MouseDown 事件过程来做这两条消息的处理函数。注意一定要把处理过程做在

```
if(Button = 4)Then
.....
End If
```

中。其中 Bubbon = 4 表示鼠标中键被按下。

在 DLL 中,中断处理函数的最后有下面两条语句

PostMessage (wm100Wnd, WM-MBUTTONDOWN, wParam, lParam);

PostMessage (wm100Wnd, WM-MBUTTONUP, wParam, lParam);

其中,wm100Wnd 是包含前述消息处理函数的 VB 应用程序窗口句柄,它可以在中断初始化函数中利用 GetActive Windows 函数获得。当 VB 调用中断初始化函数时,DLL 就可获得当前活动窗口的句柄,并把它储存在 DLL 的全局变量中,以供中断处理函数使用。lParam 含有光标的 X 和 Y 坐标,X 坐标在其低位字中,Y 坐标在其高位字中。wParam 参数在这里无用,可置为 0。

似乎仅有 PostMessage(wm100Wnd, WM-MBUTTONDOWS, wParam, lParam);一条语句就可以完全实现想要做的工作,事实上也确是这样。在 VB 应用程序接收到中断处理函数发出的 WM—MBUTTONDOWN 消息后,就会读走交换数组中的数据并对一些不很紧急的任务进行处理(可通过编程实现)。那为什么还要有 PostMessage(wm100Wnd, WM-MBUTTONUP, wParam, lParam);这条语句呢?要强调的是,这条语句必不可少!实际使用表明,如果没有这条语句,在执行完上述处理后,VB 应用程序窗口对第一次 Click 事件的反应将仅仅是激活窗口,使之能够响应以后的鼠标动作。如果中断的间隔很短,以致在这一段时间内根本不可能双击鼠标(在小于 0.5 秒的时间内,双击鼠标就很困难了,但这个时间对中断间隔来讲是很常见的),那么一旦启动中断,操作人员将不能通过鼠标对窗口进行任何操作,包括中止中断。这样利用中断的好处就一点也显示不出来了。需要指出,虽然此时窗口不能响应鼠标动作,但对键盘操作是能够响应的。

参考文献

[1] 程铁军、全相风等编,Windows 动态数据交换程序设计——用 Visual C++ & Microsoft C/C++,北京航空航天大学出版社,1995

[2] 魏彬,熊桂喜等编译,Windows 3.0 软件开发指南(三),清华大学出版社,1992

[3] 谭志斌,Windows 环境下硬件中断编程的实现,中国计算机用户,1996.3.4

(来稿时间:1997 年 4 月)