基于 BCW 和 VB 的 Windows 串行通信的实现

路红英　岳玉霞　高存宝　（北方交通大学自动化所　100044）

摘要：本文讨论了有关 Windows 环境下串行通信的实现方法，并给出了基于 BCW 和 Visual Basic 通信控件的具体实例。

关键词：串行通信　Windows　BCW　VB 通信控件

由于 Windows 界面美观，操作容易，在其平台上编写各种各样的应用程序也越来越受到广大用户的青睐。而在实际的开发过程中，我们往往需要它能够具备与外围设备进行通信的能力。除了最常用的打印机外，一些外围设备如调制解调器、传真机和通过串行口或并行口与计算机相连接的设备，均需通过 Windows 所提供的通信接口函数与计算机应用程序进行通信。特别是在计算机测控软件中，这种需要就更为突出。那么如何在 Windows 中对计算机的通信资源进行操作呢？本文给出两种方法：即利用 Windows 现成的 API 函数和利用 VB 特有的通信控件。

一、串行通信的基本机制

常用的 PC 串行通信程序大多利用 BIOS INT14H 中断，以查询 I/O 方式完成异步串行通信。

Windows 系统提供中断方式驱动的串行通信驱动程序 COMM.DRV。通信程序无需直接对串行端口进行操作，而是通过驱动程序这一编程接口进行间接操作。

图 1　用户与 Windows 通信模块的关系

Windows 操作系统中，用户与 Windows 的关系见图 1，串行通信采用“事件通知”方式支持数据按块传送。进行通信时，Windows 开辟一个用户定义的输入输出缓冲区，每接收一个字符就产生一个低级硬件中断，串行驱动程序立即取得控制权，并将字符放入输入数据缓冲区，然后将控制权返还正在运行的应用程序。如果输入数据缓冲区满了，驱动程序用当前定义的流控机制通知发送方停止发送数据。发送数据也采用类似的处理方式。应用程序将需要发送的数据放入输出数据缓冲区，然后就由 Windows 来负责传送，每发送一个字符就产生一个低级硬件中断。这一切均在后台进行，无需用户介入，用户看到的只是输入输出缓冲区。实现串行通信在 Windows 环境下的操作，用户只需进行下面几个步骤，如图 2 所示。

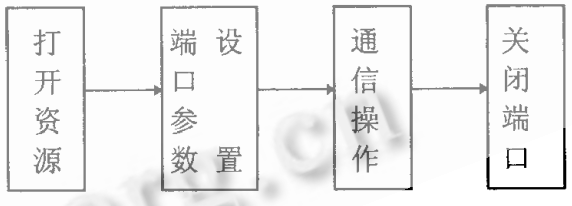

图 2　串行通信操作基本流程

首先，要打开需要进行通信的端口，并给通信的接收数据缓冲区和发送数据缓冲区分配一定大小的内存区；接着，就要对通信的端口参数进行设置，这里主要是对通信设备控制模块（DCB）进行设置，包括波特率、数据格式以及流控设置等信息；然后就可以对端口进行读写等操作了；在结束串行通信之后，最终关闭打开的通信端口。

二、基于 BCW 的 Windows 通信

由于 Windows 本身提供了完备的 API 接口，因此在编制 Windows 通信程序时完全可以利用这些 API 函数，

Windows 中与通信有关的 API 函数如表 1 所示。

下面结合 BCW 介绍通信函数的具体应用。

串行通信的 Windows 应用程序的实现分为两个部分，即初始化串行口和串行口消息处理。

表 1 Windows 中主要的通信函数

功能	编程接口函数
打开通信资源	OpenComm( )
串行通信资源配置	BuildCommDCB( ) SetCommState( )
读写操作	ReadComm( ) WriteComm( )
通信错误检查	GetCommError( )
关闭通信事件	CloseComm( )

初始化串行口的主要工作包括：选定串行口，设置输入输出缓冲区大小以打开一个串行通信端口，调用 OpenComm( )函数来完成。设置串行口的设备控制块，进而设置串行口的各项参数，分别由 BuildCommDCB( )函数和 SetCommState( )函数来完成。BuildCommDCB 函数只是把标准的 DCB 字符串转换成 DCB 中的适当字节及位设置。SetCommState 函数真正设置通信端口属性。最后还要允许串行口消息 WM-COMMNOTIFY 投递到消息队列中。需要的话，可通过 SetCommEventMask( )等函数允许响应需要的串行口事件。

串行口消息处理就比较简单了，一般是调用 WriteComm( )或 ReadComm( )函数发送或接收数据，以响应串行口消息。

另外，在通信完成以后还应关闭串行口，这通过调用 CloseComm( )函数来完成。

三、特有通信控件的 Windows 通信

Visual Basic(以下简称 VB)的灵活性，很大程度上表现为它的控件(VBX)的可扩展性，即它定义了 VBX 的外部标准，可以使得许多独立的软件公司编写出基于 VB 的 VBX 控件。其中就有 Crescent Software 公司为 Microsoft 公司所写，并包含在 VB 套装软件中的 MSCOMM. VBX。该通信控件通过串行口传送、接收数据，为用户的应用程序提供了串行通信的基础。使用 VB 中提供的这个预定义对象(通信控件)，通过改变对象属性，向对象发送消息，以及为对象事件编写响应代码，可以方便地完成用户应用程序之间的串行通信，而不必考虑其实现细节。该控件为“事件驱动”，只响应“OnComm”事件，该事件可对已发生的事件或错误进行处理，与 CommEvent 属性紧密相关。使用该控件前，必须保证 MSCOMM. VBX 已被正确安装于当前 Windows 的 \SYSTEM 子目录下。

VB 通信控件提供了 27 种属性，一个通信事件(OnComm 事件)及两个通信函数，具有丰富的功能。利用 VB 提供的通信控件来编制通信程序，关键是准确地理解并设置其属性。通信控件的主要属性有：

CommPort	设置串行口	Name	对象名
InBufferCount	输入缓冲区读入字节数	OutBufferCount	输出缓冲区写入字节数
InBufferSize	输入缓冲区字节长度	OutBufferSize	输出缓冲区字节长度
Input	读入数据	PortOpen	打开和关闭串行口
InputLen	读入数据长度	Settings	设置速率，校验，数
Output	输出数据		据位数，停止位个数

这些属性有些是在设计阶段不能设置的(PortOpen, InBufferCount, Input, InputLen, Output, OutBufferCount)。这些属性的用法在下面例子中可以看到。

OnComm 事件是通信控件提供的唯一的事件，在该事件过程中，可以对已发生的通信事件或错误进行处理，它与 CommEvent 属性紧密相关。无论什么时候 CommEvent 属性值改变时，都会发生 OnComm 事件，表明有一个通信事件或错误发生。

在 VB 通信控件的许多属性中，CommEvent 属性是一个很重要的属性，它返回当前通信事件或错误，该属性在设计时不可用，运行时只读。只要有通信事件或错误发生，VB 通信控件就会触发一个 OnComm 事件，程序即进入了 OnComm 事件的处理过程。与此同时，CommEvent 属性就有了一个对应于当前发生的通信事件或错误的属性值(数字化的代码)。为了确定触发 OnComm 事件的通信事件或错误，可参照 CommEvent 属性表。例如，属性值 = 1001，属性名称为 MSCOMM-ER-BREAK，表示一个 BREAK 信号被接收；属性值 = 2，属性名称为 MSCOMM-EV-RECEIVE，表示接收 Rthreshold 个字符等。

下面给出一个上位机和下位机之间的通信实例，系统中，下位机负责数据采集，上位机进行现场数据处理。其中上位机的通信软件清单如下：

程序 1 给出了上位机通信程序初始化连接程序。

上位机控件清单：

控 件	名 称	属性
Form	Form1	Caption = "上位机通信"
MSComm	Comm1	
Timer	Timer1	Interval = 1000

程序 1 窗体及通信初始化连接程序

```
Sub Form1-Load( )
    Form1.SHOW '显示 Form1
    Comm1.CommPort = 1 '应用 COM1 口
    Comm1.Settings = "9600, N, 8, 1" '波特率 9600, 无奇偶校验位, '8 位数据位, 1 位停止位
    Comm1. InputLen = 0 ' Input 将读取接收缓冲区的全部内容
    Comm1. InBufferSize = PACKLENGTH '输入缓冲区长度设置
    Comm1.OutBufferSize = 2 '输出缓冲区长度设置
    Comm1.PortOpen = True '打开串行口
    Comm1.SThreshold = 1 '当发送缓冲区为空时触发 OnComm 事件
    Comm1. Rthreshold = PACKLENGTH - 1 '接收缓冲区字符个数达到 PACKLENGTH 时触发 OnComm 事件
End Sub
```

示例程序中，上位机向下位机发送是在定时事件 Timer1-Timer( )中完成的，每间隔一定的时间(1 秒)，Windows 获得控制权，执行定时事件 Timer1-Timer( )，上位机向下位机发送"OK"信号。程序 2 给出了上位机的这部分程序。下位机接收到发送来的"OK"信号后，向上位机发送一长度为 PACKLENGTH 的数据包，于是上位机的通信对象 Comm1 的 OnComm 事件被触发，程序进入 Comm1-OnComm( )事件过程。若正常接收，CommEvent 属性被置为 MSCOMM-EV-RECEIVE，表明数据包被接收。程序 3 给出上位机接收部分的程序。为了便于观察接收到的数据包，这里将接收的数据放到一个文本框中以便显示。

程序 2 定时发送程序

```
Sub Timer1. Timer( )
    Comm1. OutBufferCount = 0 '清空输出缓冲区
    outstring $ = " OK "
    Comm1. Output = outstring $ '发送"OK"命令到串行口
End Sub
```

程序 3 通信接收及处理子程序

```
Sub Comm1-OnComm( )
  Select Case Comm1. CommEvent
    'Error(错误)
    Case MSCOMM-ER-BREAK
    Case MSCOMM-ER-CDT0
    Case MSCOMM-ER-CTST0
    Case MSCOMM-ER-DSRT0
    Case MSCOMM-ER-FRAME
    Case MSCOMM-ER-OVERRUN
    Case MSCOMM-ER-RXOVER
    Case MSCOMM-ER-RXPARITY
    Case MSCOMM-ER-TXFULL
    'Events(事件)
    Case MSCOMM-EV-CD
    Case MSCOMM-EV-CTS
    Case MSCOMM-EV-DSR
    Case MSCOMM-EV-RING
    Case MSCOMM-EV-SEND
    Case MSCOMM-EV-RECEIVE
      mystring $ = Comm1. Input
      text1. text = " "
      text1. text = mystring $ '显示接收的字符
  End Select
End Sub
```

参考文献

[1] 于恒兰 兰旭"Windows 环境下串行异步通信程序的设计技术"电脑编程技巧与维护 1996.5

[2] 谢庆国 刘英敏"基于 Windows 的实时通信的实现"中国计算机用户 1997.2

[3] 舒继武 袁光锋"Visual Basic 3.0 中的通信及应用"微型机与应用 1996.8

(来稿时间：1998 年 5 月)