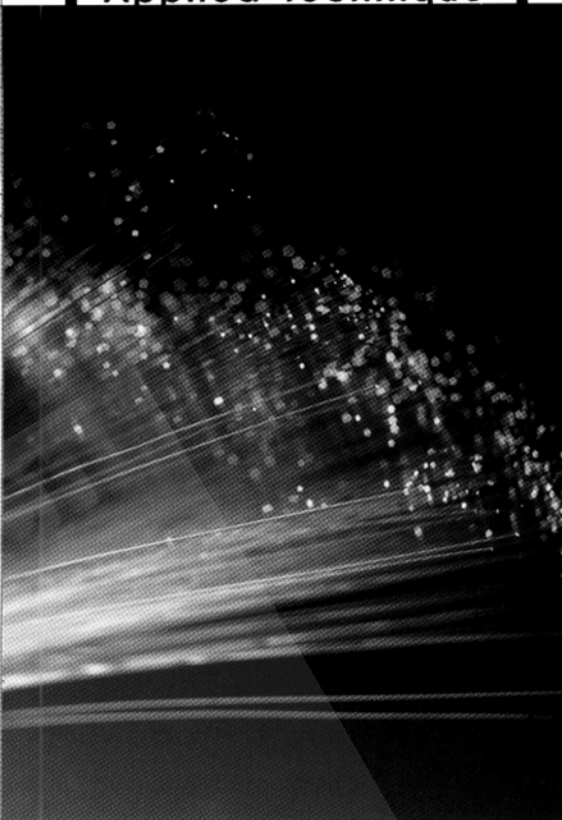




基于PC可视化环境的串行通信方案

詹国华 张量 袁贞明 徐建军 (杭州师范学院信息工程学院 310036)

摘要: 文章对PC微机可视化环境的串行通信进行研究,介绍了利用PC微机串行端口实现通信的几种实

用方案,如VB或Delphi等可视化编程语言中利用Windows API函数实现串行通信,或利用MSComm

通信控件实现串行通信,或拓展编写动态链接库实现串行通信,并给出了具体的实现范例。

关键词: 串行通信 API MSComm DLL

1 引言

随着信息技术的发展以及计算机网络的广泛使用,计算机通信技术日臻成熟。在众多通信实现方案中,异步串行通信作为一种灵活、方便、可靠的通信手段,有着广泛的应用,尤其是在工业控制领域,微机与微机、微机与外设、微机与分布式下位

机等都可以通过RS-232C串行端口互连通信,以实现控制和传输数据等目的。随着基于PC微机的Windows操作系统的普及应用,特别是Windows 95的推出,越来越多的PC微机用户转向了可视化的Windows操作系统。但Windows应用软件的开发却并非易事,特别是涉及到对底层的控制,在

Windows可视化环境下怎样利用串行端口实现通信,是目前受到普遍关注的一个研究课题,一般认为可利用Windows的通信API函数或利用Visual Basic标准通信控件MSComm来实现,本文探讨了几种串行通信方案。

2 利用Windows API函数实现串行通信

Windows环境下,利用API函数对串口控制,可实现丰富的功能,应用广泛,适合于编写较为复杂的底层通信程序。实现串行通信一般可分为以下几个步骤:

(1) 初始化串行端口: 首先用CreateFile函数打开指定串口,置参数中访问类型为GENERIC_READ | GENERIC_WRITE,共享模式为0,创建标志为OPEN_EXISTING,模板句柄为NULL。

(2) 配置串口: 可使用CommConfig Dialog函数以对话框方式设置波特率、数据位、奇偶校验、停止位和流控制方式,并且可以恢复缺省值。

(3) 超时控制: 为了防止意外情况使串口读写无休止地等待,Windows提供了超时设置,这是由COMMTIMEOUTS结构定义的,通过调用GetCommTimeOuts函数获得当前的设置,用SetCommTimeOuts函数来完成设置。

(4) 数据传输: 使用 ReadFile、WriteFile 函数读写串行端口, 读写操作可分四种方式: 查询方式、同步 I/O、异步 (后台) I/O 和事件驱动 I/O。

(5) 关闭串口: 程序的终止可以自动关闭串口, 也可用函数 CloseHandle 来关闭串口, 以便释放所独占资源。

在 Windows 95 系统中, 串行口和串行通信驱动程序是通过一个数据结构进行配置的, 这个数据结构被称为设备控制块 (Device Control Block), 简称 DCB。

2.1 VB 中利用 API 函数实现串行通信

由于控件的易用性, 现在使用 Visual Basic 控制串口几乎都以控件为实例, 很少有介绍利用 Windows API 函数进行控制串口的, 这样使得我们无法更好的利用串口资源, 更由于使用 Windows API 函数实现的功能比使用控件更多 (控件其实也是调用了 Windows API 函数, 当然不可能把所有的控制串口的信息放入该控件中), 但是使用 Windows API 函数较为复杂。

在 Visual Basic 中利用 Windows API 函数来实现一般的串行通信需要用到下列一些函数: CreateFile 打开串口; BuildCommDCB 创建 DCB; GetCommState 获取当前 DCB; SetCommState 设置 DCB; SetupComm 设置输入输出缓冲区; ReadFile 读取串口信息; WriteFile 向串口写信息; apiGetCommTimeouts 获取当前超时设置; apiSetCommTimeouts 超时设置; CloseHandle 关闭串口。

2.2 Delphi 中利用 API 函数实现串行通信

在 Delphi 中利用 Windows API 函数控制串口, 其思路和一般过程类似于在 Visual Basic 中的编程, 都是 (1) 打开串口 (2) 配置串口 (3) 进行通信 (4) 关闭串口。

一般而言, 串行通信需要利用多线程技术实现, 而 Delphi 比 Visual Basic 更容易实现多线程编程。线程是进程内部执行的路径, 是操作系统分配 CPU 时间的基本实体。每个进程都由单线程开始完成应用程序的执行。其主要的处理逻辑可以表述如下: 进程一开始先由主线程做一些必要的初始化工作, 然后主线程根据需要在适当时候建立通信监视线程监视通信口, 当指定的串行口事件发生时, 向主线程发送 M_COMMNOTIFY 消息, 主线程对其进行处理。注意: 多线程同时执行, 将会引起对共享资源的冲突。为避免冲突, 就要用同步多线程对共享资源进行访问。通过 CreateEvent() 创建事件对象, 使用 SetEvent() 或 PulseEvent() 函数将事件对象设置成信号同步。在应用程序中, 利用 WaitForSingleObject() 函数等待同步的触发, 等到指定的事件被其他线程设置为有信号时, 才继续向下执行程序。

在读取串口信息时, 一个比较好的解决办法是调用 ClearCommError() 函数, 来确定在一次读操作中在缓冲区中等待被读的字节数。使用该函数时还可以检测错误并且清除错误条件。在通信中, 超时是个很重要的考虑因素, 因为在数据接收过程中由于某种原因突然中断或停止, 如果不采取超时控制机制, 将会使得 I/O 线程被挂起或无限阻塞。串行通信中的超时设置分为两步,

首先设置 COMMTIMEOUTS 结构的五个变量, 然后调用 SetCommTimeouts() 设置超时值。我们可以使用 GetCommTimeouts() 函数得到系统初始值。

3 VB 标准控件 MSComm 实现串行通信

使用 Windows API 函数进行串行通信, 虽然能够实现的功能较多, 但对编程者要求较高, 其操作也较复杂。而使用控件给我们提供了方便。Visual Basic 中为我们提供了用于串行通信的标准控件 MSComm, 它提供了功能完善的串行数据的发送和接收功能。

MSComm 控件有事件驱动和程序查询两种处理方式, 前者由 MSComm 控件的 OnComm 事件捕获并处理通信错误及事件, 而后者通过检查 CommEvent 属性值来判断事件和通信错误。

MSComm 控件的主要属性有:

- (1) CommPort: 设定或传回通信连接端口代号, 程序必须指定所要使用的串行端口号码, Windows 系统会用该通信端口与外界通信, 程序也可以从该属性读回所使用的连接端口号。(注意: MSComm 控件的端口号最大值为 16)。
- (2) Settings: 设定初始化参数。以字符串的形式设定或传回联机速度、同位检查、数据位、停止位四个参数。其格式为: "BBBB,P,D,S", MSComm 控件的默认值为 "9600,N,8,1", 最高联机速度视其传输芯片而定, 如: 传输芯片为 16C450, 其最高联机速度为 19200 bps, 而 16C550 芯片因含有缓冲区, 其最高联机速度可达 115200 bps。正确的数据位为 4, 5, 6, 7, 8 (默认), 正确的停止位为 1 (默认), 1.5 或 2。(注意: 使用 RS-232C 通信双方, Settings 必须完全一致)。
- (3) PortOpen: 设定或传回通信连接端口的状态。
- (4) Input: 从输入缓冲区传回并移除字符。采用 FIFO (First In First Out) 机制。
- (5) Output: 将一个字符写入传输缓冲区, 是否发送给对方视传送门槛属性的设定 (SThreshold)。
- (6) HandShaking: 指定通信两方的握协议。即流量控制。
- (7) Rthreshold: 设定或传回引发接收事件的字符数, 若为 0 (默认), 则无论缓冲区中有多少字符均不引发接收事件。
- (8) Sthreshold: 若设为 0 (默认) 则数据传输事件不会产生 OnComm 事件中的传送事件。
- (9) CommEvent: 只要有通信错误或事件发生时都会产生 OnComm 事件。
- (10) InputMode: 设定或传回 Input 属性取回的数据的型态。(0: 文字形式; 1: 二进制形式)。

注意, 在使用 MSComm 控件时由于它不是标准控件, 需在 "工程" → "部件" 中添加。

除了在 Visual Basic 中利用该控件控制串口外, 还可以在 Delphi 中通过导入该控件进行串口编程, 须注意的是: 在早期版本的 Delphi 中无法使用 MSComm 控件, 例如在 Delphi 3.0 中 (下面讨论的是在 Delphi 5.0 中用该控



件控制串口)。还有在Delphi中没有提供串行通信控件,需要采用Microsoft的MSComm控件,MSComm控件是VB中的OCX控件,首先需要将其添加到Delphi中,选择菜单“Component”→“Import ActiveX Control”,在“Import ActiveX”页内选择“Microsoft Comm Control”,点击“Install”安装,安装后在“ActiveX”组件板中出现MSComm图标(为一部电话机),即可被使用。有一点要注意,在Object Inspector中MSComm控件的Input和Output属性以及PortOpen等属性是不可见的,但它们仍然存在,前两个属性的类型是OleVariant(Ole万能变量)。

在Delphi中对该控件的属性和方法的使用基本与在Visual Basic中一样,下面给出一接收数据的语句样例(MSComm控件名为:MSComm1)

// 串口的打开以及初始化,使用时可放入合适位置,如窗口建立或某一按钮块中,

```
MSComm1.CommPort := 2;           // 串口号, 2为 Com2 口
MSComm1.Settings := '9600,n,8,1'; // 设置波特率, 校验位, 数据位, 停止位
MSComm1.PortOpen := true;         // 打开串口
MSComm1.InBufferCount := 0;       // 清空接收缓冲区
MSComm1.InputLen := 0;            // Input 读取整个缓冲区内容
MSComm1.RThreshold := 1;         // 每次接收到字符即产生 OnComm 事件
```

```
MSComm1.DTREnable := true;       // 数据终端准备好
MSComm1.RTSEnable := true;       // 请求发送

// 数据的接收, 处理, 使用时放入该控件所在窗体的 MSComm1Comm 中
if MSComm1.CommEvent = 2 then
begin
```

```
readstr := MSComm1.Input; // readstr 为 Olevariant 变量
// 加入对收到的数据进行处理模块
end;
```

```
MSComm1.PortOpen := false; // 关闭串口
```

对于使用 Visual Basic 的 MSComm 控件对串口进行控制,只需知道 MSComm 的属性和事件的用法即可,但是 MSComm 控件给我们提供的功能有限,当我们所要设计的串行通信较为复杂,更全面的对串口进行控制时,就要用到 Windows API 函数对串口控制,前面已具体讨论了使用 Windows API 函数控制串口。

在 Visual Basic 中,当传送字符中含有超过 ASCII 128 以上的字符时,使用 MSComm 默认的数据型态时将无法正确传送,这是由于 MSComm 在使用文字模式时,Visual Basic 会把 ASCII 128 以上的字符认为是双字节字符的一部分,当接收到超过 ASCII 128 以上的字符时,会等待下一个字符来组合成一个文字,此时必须采用二进制模式来传送字符。这是一个容易被疏忽的问题。

MSComm 控件的通信功能实现,实际上是调用了 API 函数,而 API 函数是由 Comm.drv 解释并传给设备驱动程序执行的,VB 程序开发者只需知道 MSComm 控件的属性和事件的用法即可以实现串行通信。

4 编写自己的动态链接库实现串行通信

使用 Windows API 函数来控制串口,使程序功能更强大,但编写时较为复杂,我们可将所需的功能编写为自己的动态链接库,提供给不同的程序使用,实现代码重用,方便调用者,还可节省资源,使使用者在用 Windows API 函数控制串口时,和使用 Visual Basic 的 MSComm 控件一样方便,但功能更完善。

4.1 利用 Visual Basic 编写动态链接库

在 Visual Basic 中只能编写 ActiveX DLL，而无法编写出真正的 DLL 文件。下面以一个实例来简单说明如何编写关于控制串口的 ActiveX DLL 文件，该实例只是简单的实现串口的读写用以串口的通信。新建工程时选择 ActiveX DLL。在编写动态链接库时应注意：仅供动态链接库本身使用的变量，函数等必须以 Dim 或 Private 定义，而提供给调用程序使用的接口函数必须以 Public 定义。

在 Visual Basic 中对 ActiveX DLL 的调用，首先需将该文件复制到 C:\WINDOWS\SYSTEM 目录下，（假设实例保存的文件名为 Mycomm.dll；其工程名称为 MycommF 在“工程”→“工程属性”中设置；类模块名称为 MycommCls）然后选择菜单“工程”→“引用”，在弹出的对话框中选择 MycommF 项（若没有，可单击浏览，在对话框中选择 Mycomm.dll 文件即可）确定即可。变量的定义：有两种方法可定义对该类型的引用。（1）Dim Mycomm1 As New MycommCls。（2）Dim Mycomm2 As New MycommF。MycommCls 我们以第一种方式定义。具体的使用：

```
Dim Handle as Long           '定义串口句柄
Mycomm1.OpenThePort ("com2") '打开串口 2
Mycomm1.WriteStr (Handle, "WriteString") '向串口写信息，返回
True 为成功
Mycomm1.ReadStr (Handle)     '从串口读取信息，返回
值为串口信息
Mycomm1.CloseCOM (Handle)    '关闭串口，返回 True 为成功
```

上述仅提供了以上几种函数，当然，我们可以为动态链接库添加更多的功能，以提供给用户。注意，在 Visual Basic 中动态链接库的使用方法与 Windows API 函数的使用方法一样，用到的每个函数都需进行申明。

4.2 利用 Delphi 编写动态链接库

使用 Delphi 所编写的动态链接库可以为不同的语言所调用。利用 Delphi 的多线程编写简单的优势和它所编写的 DLL 文件不被调用语言环

境所限制，我们可以用 Delphi 编写出功能更为强大的串行通信函数库，或者编写出提供特定功能的串行通信函数库。根据上面在 Delphi 中利用 Windows API 控制串口中提到的方法，先编写出串行通信程序，然后将其转化为 DLL 文件。或直接建立 DLL 文件，然后根据在 Delphi 中利用 Windows API 控制串口中提到的方法在文件中实现串行通信。值得注意的是在编写完各类通信函数后，应提供外部调用接口，即在 DLL 文件中有 Exports 导出表。注意，在使用 Windows API 函数来编写自己的动态链接库控制串口时，当我们不再需要使用串口资源时，应及时地将串口资源释放，即用 CloseHandle (Hcomm) 函数来释放所占用的串资源。

5 总结

在使用 PC 串口实现串行通信时，可以利用 Windows 提供的 API 函数，也可以利用 Visual Basic 中的标准控件 MSComm。利用 API 函数时能实现较为复杂的功能，应用面更为广泛，并能实现更为底层的操作，但同时对编程者的要求也更高，编写过程也更为复杂；而使用控件则方便、简单，但其能实现功能也有限。这样就有了第三种实现方案：利用 API 函数编写自己的动态链接库，其优点是利用 API 函数编写出控件无法实现的功能，而其使用则与使用控件一样的方便。

参 考 文 献

- 1 范逸之，Visual Basic 与 RS232 串行通信控制 [M]，北京中国青年出版社，2000，第 1 版。
- 2 美，Dan Appleman，京京翻译组，Visual Basic 5.0 Win32 API 开发人员指南 [M]，机械工业出版社，1998，第 1 版。
- 3 季雪岗、王晓辉、张宏林等，Delphi 编程疑难详解 [M]，人民邮电出版社，2000，第 1 版。

