串行通讯之
.NET SerialPort
异步写数据

Hanford

2016 年 12 月 05 日

目 录

第 1 章　说明

1　为什么需要异步写数据？

如下图所示，以波特率 300 打开一个串口。

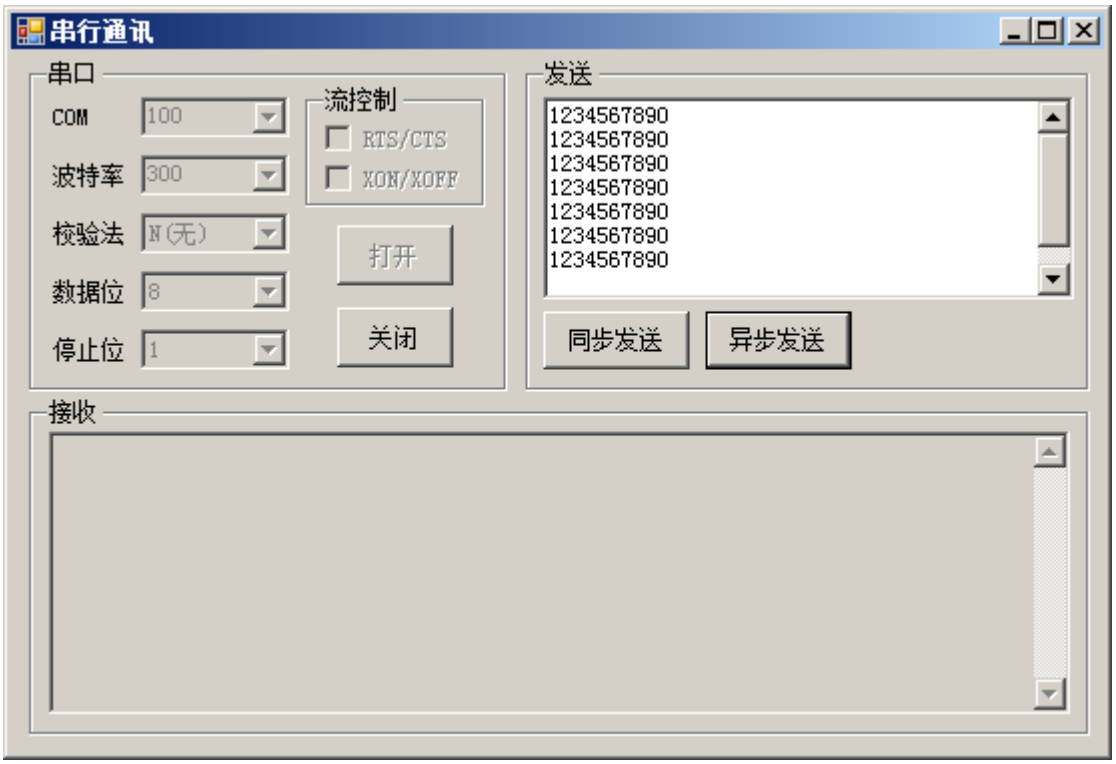

图 1

单击“同步发送”按钮，则数据未发送完之前写数据的函数不会返回。波特率 300，每秒大概能发送 25 个字符，发送 500 个字符就需要 20 秒。这 20 秒之内，整个程序将处于假死状态。

单击“异步发送”按钮，就不会出现假死状态。

2　异步写数据的代码

异步写数据的代码如下：

```
private void btnWriteAsync_Click(object sender, EventArgs e)
{//异步写
    byte[] byt = System.Text.Encoding.Default.GetBytes(txtSend.Text);
    if(byt!=null && byt.Length > 0)
    {
        IntPtr hComm = GetCommHandle(m_sp);
        UInt32 w = 0;
        m_ov.hEvent = IntPtr.Zero;
        m_ov.Internal = IntPtr.Zero;
        m_ov.InternalHigh = IntPtr.Zero;
        m_ov.Offset = 0;
        m_ov.OffsetHigh = 0;
        WriteFile(hComm, byt, (UInt32)byt.Length, ref w, ref m_ov);
    }
}
```

要点为：

1）GetCommHandle 函数获取.NET SerialPort 对象的串口句柄 hComm；

2）调用 WriteFile 函数，异步写数据。

以下是结构 OVERLAPPED 的声明、函数 WriteFile 的声明、函数 GetCommHandle 的实现：

```
[StructLayout(LayoutKind.Sequential,Pack=4)]
public struct OVERLAPPED
{
    public IntPtr Internal;
    public IntPtr InternalHigh;
    public UInt32 Offset;
    public UInt32 OffsetHigh;
    public IntPtr hEvent;
}
[DllImport("kernel32.dll", SetLastError = true
        , CallingConvention = CallingConvention.Winapi)]
private static extern UInt32 WriteFile(IntPtr hFile, byte[] lpBuffer
                                , UInt32 nNumberOfBytesToWrite
                                , ref UInt32 lpNumberOfBytesWritten
                                , ref OVERLAPPED lpOverlapped);

protected System.IO.Ports.SerialPort m_sp =
                                    new System.IO.Ports.SerialPort();
protected OVERLAPPED      m_ov;

static IntPtr GetCommHandle(System.IO.Ports.SerialPort sp)
```

```
{//获得串口句柄，供 Win32 API 使用
      IntPtr hComm = IntPtr.Zero;
      if (sp != null)
      {
            object stream = typeof(System.IO.Ports.SerialPort).GetField("internalSerialStream", System.Reflection.BindingFlags.NonPublic | System.Reflection.BindingFlags.Instance).GetValue(sp);
            var handle = (Microsoft.Win32.SafeHandles.SafeFileHandle)stream.GetType().GetField("_handle", System.Reflection.BindingFlags.NonPublic | System.Reflection.BindingFlags.Instance).GetValue(stream);
            hComm = handle.DangerousGetHandle();
      }
      return hComm;
}
```

3 源代码

本文的代码已上传至 git 服务器：

https://github.com/hanford77/Exercise

https://git.oschina.net/hanford/Exercise

在目录 SerialPort\c# 目录内。