基于 C# 的串口通信系统的研究与设计

周 阳 周美娇 黄 波 刘金利
(上海理工大学 光电信息与计算机工程学院 上海 200082)

摘 要：针对工业控制中 PC 对执行机构的控制与数据处理等问题，采用 RS232 串口线连接 PC 端上位机和 CPU 板的结构来设计一个串口通信系统。该系统的上位机采用. NET Framework 的 Windows Forms 模块，并利用其常用控件的方法、属性和事件，实现了 PC 下载编译文件至 CPU 板、发送功能指令、接收应答数据，并实时显示通信过程。经实际验证，该系统通信功能正常，且界面友好，具有功能强、操作方便的优点，有较强的实用性。

关键词：C#；串口通信；控件；文件下载；RS232
中图分类号：TP311.52 **文献标识码**：A **国家标准学科分类代码**：520.4060

The research and design of serial communication system based on C#

Zhou Yang Zhou Meijiao Huang Bo Liu Jinli
(School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200082, China)

Abstract: In industrial control, for the problem of controlling the actuator directly by PC and data processing, we design the serial communication system by using RS232 serial cable to connect PC and CPU board. By using the Windows Forms module of . NET Framework and the properties、events and methods of its commonly used controls, we design the PC software. It can realize the goal of downloading compiled files from PC to CPU board, sending function commands, receiving response data, and it has real-time display for communication process. Practical use verifies that the serial communication of this system is correct, It has a friendly interface, advantages of strong function and easy operation, and good practicality.

Keywords: C#; serial communication; controls; download files; RS232

1 引 言

在现代工业控制中，既要根据实时情况发出控制指令，使现场设备完成操作，又要准确获取现场数据，对其进行分析，从而实现集中管理。

一般的控制系统中，下位机为满足不同的控制需求，要更新不同版本的驱动程序，单步控制、调试执行机构，并采集数据进行分析。为增强控制的灵活性、智能性及人机交互的友好性，设计 PC 端上位机，通过 RS232 串口线连接 CPU 板。在这样的控制系统中，工作人员仅通过上位机中按键就能控制整个系统，包括更新驱动程序；控制各执行机构运行，这也便于对各部件的调试与维护；同时又能实时查看数据进行分析。

利用串口进行通信也具有这样的好处：传送距离长且成本低，有足够的带宽，易实现自定义的协议，并且数据传输可靠、线路简单、使用灵活、维护方便[1-2]。

是唯一专门为. NET Framework 设计开发的语言，其简单、类型安全的特点极大增加了开发效率，在表达形式上，它不仅保持了 C 语言的特点，而且增加了很多创新点，极大缩短了开发周期[3]。

因此，利用串口、C# 本身的优势，在 Visual Studio 2010 的开发环境下，进行串口通信系统的设计，从而提高整个控制系统的智能化程度。

2 系统总体设计方案

串口通信中，PC 端软件作为上位机、单片机作为下位机组成的数据通信系统，是最可靠、最简单、灵活的通信方式[4]。

在上述串口通信系统中，常包括：PC 端上位机，RS232 串口线，MCU 及外围电路，相应的执行机构，典型的系统结构如图 1 所示。

图 1 串口通信系统

收稿日期：2014-11

该系统中，以设计 PC 端上位机为主，其包括 2 个界面：初始化界面和通信界面。初始化界面中实现对串口参数的选择与配置、串口的打开与关闭；通信界面中实现编译文件的选择与下载、按键控制执行机构，并包含一个显示区域。

为实现串口通信，除了必要的初始化工作，还需制定一个良好的通信协议，PC 端和下位机 MCU 均以固定的格式发送数据，这样能对接收到的数据进行校验，保证数据收发的正确性[5]。当然，编译文件的下载也是在此基础上实现的，首先将文件中的所有数据读出并存储好，再根据已制定好的协议将数据发送给下位机。为达到操作的友好性，上位机要有必要的报错功能，将通信过程中有错的应答过程如实显示，便于用户修改。

3　实验方法与控件应用

3.1　实验方法与开发平台

该串口通信系统中的上位机，是在 Visual Studio 2010 的开发环境下，利用. NET Framework 的 Windows Forms 模块中的控件进行设计的。

Visual Studio 2010 是微软公司推出的开发环境，是目前最流行的 Windows 平台应用程序开发环境，它同时带来了 NET Framework 4. 0、Microsoft Visual Studio 2010 CTP，并且支持面向 Windows7 的应用程序[6]。

Visual Studio 2010 是. NET 开发的首选工具，它可自动执行编译源代码的步骤；允许 UI 元素的简单拖放设计；具备智能检测错误功能，在输入代码时给出合适的推荐代码；可以显示、导航项目中的元素，这些元素可以使 C＃源文件，也可以是其他图像或声音文件。

3.2　控件介绍与应用

3.2.1　控件 OpenFileDialog

该控件是一个需要预先配置的对话框，配置好后，便可提供 Windows 操作系统中的“打开文件”对话框相同的操作和功能。

AddExtension 属性的设置确定是否自动在文件名中添加扩展名。

FileName 属性是存储在“文件”对话中选定文件的名称，包括完整的路径。

Filter 属性中设置的字符串，决定着对话框的“另存为文件的类型”或“文件类型”框中出现的选择内容，若当前文件筛选器未包含有效文件扩展名，则添加 DefaultExt 属性中的指定的扩展名。

InitialDirectory 属性可用下列源之一进行设置：以前在程序中使用的路径、从持久性源读取的路径、标准 Windows 系统和用户路径、与当前应用程序相关的路径。

Multiselect 属性设置为 True 可选择打开多个文件。

设计过程中，对涉及到的属性设置好后，再利用 ShowDialog 方法，便可实现既定功能。

3.2.2　控件 SerialPort

SerialPort 控件用于控制串行文件资源，提供同步 I/O 和事件驱动的 I/O、对引脚和中断状态的访问，以及对串行驱动程序属性的访问，支持以下格式的编码：ASCIIEncoding、UTF8Encoding、Unicode Encoding、UTF32Encoding 以及 mscorlib. dll 中定义的代码页小于 50000 或者为 54936 的所有编码。

该控件在整个系统中发挥着至关重要的作用，通信功能的实现都是以该控件为基础的，其使用方法如下所述：

1）参数类属性

运用串口进行通信，首要工作就是配置好串口的 5 个基本参数：PortName、BaudRate、DataBits、Parity、StopBits。除属性 PortName 均可常采用各自的默认值，但要注意，其值要与单片机的串口模块设置成一样的参数，这样才能进行正常通信[1]。

2）控制类属性

BreakState 属性确定中断信号的状态。若要进入中断状态，将其设置为 True。当传输被挂起并且线路被置于中断状态时，中断信号状态发生且直到其被释放时才停止；

DiscardNull 属性确定 Null 字节在端口和接收缓冲区之间传输时是否被忽略。在正常情况下，此值应设置为 False，若为 True，则会使 UTF32 和 UTF16 编码字节产生意外结果；

Encoding 确定传输前后文本转换的字节编码，默认为 ASCIIEncoding；

Handshake 属性确定端口数据传输的握手协议。握手时，将指示连接到 SerialPort 对象的设备在缓冲区中至少有(ReadBufferSize-1024)个字节时停止发送数据。当缓冲区中字节数不大于 1024 时，将指示设备重新开始发送数据。如果设备在大于 1024 个字节块中发送数据，可能导致缓冲区溢出；

若将 Handshake 属性设置为 RequestToSendXOnOff 并将 CtsHolding 设置为 False，则不会发送 XOff 字符；

ReceivedBytesThreshold 属性设置 DataReceived 事件发生前内部输入缓冲区中的字节数；

NewLine 属性定义 ReadLine 和 WriteLine 方法的行尾，默认为换行符；

DtrEnable 属性设置在串行通信过程中启用数据终端就绪(DTR)信号的值。在 XON/XOFF 软件握手、请求发送/可以发送(RTS/CTS)硬件握手和调制解调器通信的过程中，通常启用 DTR；

RtsEnable 属性设置串行通信中是否启用请求发送(RTS)信号；

ReadBufferSize、WriteBufferSize 属性分别设置输入、输出缓冲区的大小，设置的值小于各自的默认值都会被忽略；

3）状态、数据类属性

IsOpen 属性是根据其逻辑值指示 SerialPort 对象的打开或者关闭的状态;

BytesToRead、BytesToWrite 属性用于获取接收和发送缓冲区的字节数,缓冲区包括串行驱动程序的缓冲区和 SerialPort 对象自身的内部缓冲;

DsrHolding 属性获取数据设置就绪(DSR)信号的状态;

4)事件

SerialPort 控件最重要的事件就是 DataReceived 事件。当接受缓冲区的字节数达到 ReceivedBytesThreshold 属性的设置值时,就会自动触发该事件,进行数据的接收[3,7]。

3.2.3　控件 ProgressBar

该控件常用于帮助用户了解等待一项进程完成所需的时间,在窗体上只能水平放置。通常,在水平条中显示数目的矩形来指示进程的进度。

HasChildren 属性用于确定控件是否包含一个或多个子控件。

Step 属性用于确定一个值,该值对应每次调用 PerformStep 方法在进度栏中填充的量。

Style 属性用于确定在进度栏上指示进度的方式,包括:以增加分段快的数量来指示进度;以增加平滑连续的栏来指示进度;通过字幕方式在 ProgressBar 中连续滚动一个块来指示进度。

3.2.4　控件 ComboBox

该控件用于在下拉组合框中显示数据,它分两个部分显示:顶部一个允许用户键入的文本框,下部是一个列表框。

AutoComplete 类属性能够创建带自动填充功能的 ComboBox,可将所输入的字符串前缀与所维护源中的所有字符串的前缀进行比较,以此自动完成输入字符的填写。

SelectionStart 属性确定组合框中选定文本的起始索引。

4　系统软件设计

4.1　软件设计方案

本系统的主要功能是,利用串口线 RS232 进行数据发送与接收,这是通过控件 SerialPort 实现的。同时,利用控件 OpenFileDialog 来选择、打开文件,将文件中的数据发送至 MCU,并由 MCU 写入 Flash 中。根据系统功能,可得出如图 2 所示的软件设计流程。

根据图 2,软件设计的流程为:先配置好串口的参数,随后打开串口;在确保串口已经打开的基础上,可以进行功能模块的选择,即用户选择是下载文件还是发送指令或数据,在完成一次操作后,可继续进行其他操作,直到退出该系统软件[1,8]。

在系统执行任意一个功能时,若通信过程中数据收发有错误,会立即终止当前过程,并将具体的错误信息显示给用户。

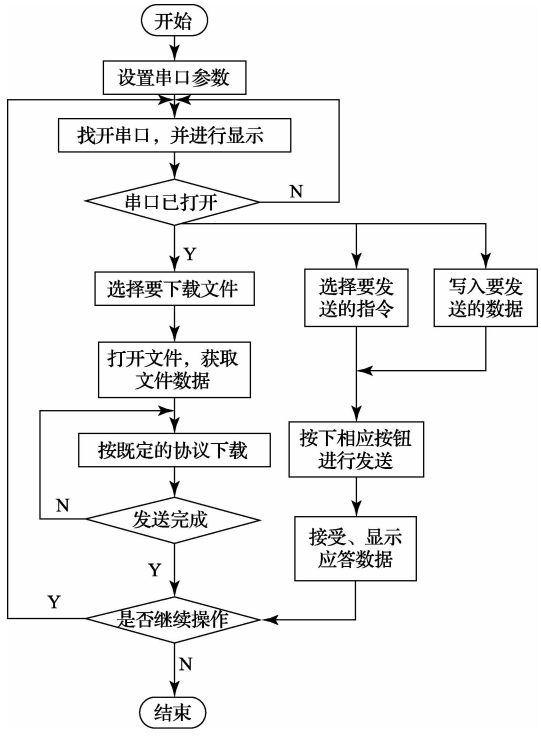

图 2　系统功能结构

4.2　软件设计实现

4.2.1　选择、打开文件

使用对话框类控件,应按照以下步骤:从 Windows Form 类别中找到 OpenFileDialog 控件,并拖放到 Windows Form 设计器的底部(此时必须引用命名空间 System.Windows.Forms);把一些属性设置为启用/禁闭可选功能,并设置对话框的状态;调用 ShowDialog 方法,显示对话框,等待并响应用户的输入;若用户按下 OK 按钮,则关闭对话框,比较对话框的结果和 DialogResult.OK 之后,通过查询特定的属性值,获取用户的输入值。

当启动应用程序时,可以把文件名作为一个参数,传递归来的文件名用来打开文件,从而可以获取文件中的数据。

修改 UARTCommunication 构造函数中的执行代码,对所传递的文件名使用一个字符串:

```
public UARTCommunication(string filename)
{
        InitializeComponent();
        if (filename ! = null)
        {
           this.filename = filename;
           OpenFile();
        }
}
```

在文件 Program.cs 中修改 Main()方法的执行代码,

以传递一个参数：

```
static void Main(string[] args)
{
    string filename = null;
    if (args.Length ! = 0)
        filename = args[0];
    Application.EnableVisualStyles();
    Application.Run(
        newUARTCommunication(filename));
}
```

另外，还必须执行 OpenFile()方法，打开一个文件，获取并存储文件中的数据：

```
protected void OpenFile()
{
    try
    {
        filepath = opf.FileName;
        arrayFileData=
            File.ReadAllBytes(filepath);
    //数组 arrayFileData 已定义
}
catch (IOException ex)
{
        MessageBox.Show(ex.Message,
        "UART Communication",
        MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);
    }
}
```

这里用 File 类读取文件，要添加 System.IO 命名空间。另外，使用 OpenFile()方法，没有编写打开文件的调用代码，这样可使 OpenFile()在程序的其他部分再次使用[7]。

文件过滤器定义了用户可以选择的文件类型，增加了选择文件的快速性，简单的过滤器字符串为：Text Documents (*.txt) | *.txt | All Files | *.*。

4.2.2　接收、发送数据

按照软件设计的流程图，利用控件 SerialPort 收发数据时，首先要做的是配置好串口的 5 个参数。在此，可以将常用的：8 个数据位、1 个停止位、无校验位、波特率为 115 200设置为默认参数，也可在上述控件 ComboBox 的 Items 属性里添加一些值，供用户在特殊情况下选择，以增强通信系统的友好性[9-11]。

对于属性 PortName 的设置则稍复杂些，因为每台计算机所能使用的 COM 口不一定是相同的。若每次使用该通信系统时，都让用户去设备管理器中查看占用的 COM 口，会显得繁琐。对于这一问题，可以采用自动获取 COM 口的方法来为 PortName 配置参数：

```
Foreach(string com in System.IO.Ports.SerialPort.
GetPortNames())
    this.comboBoxPortName.Items.Add(com);
    comboBoxPortName.SelectedIndex = comindex ;
    comindex += 1;
    serialport.PortName = comboBoxPort.Text;
```

至此，已将串口的参数配置好，再将串口打开，便可以进行数据发送、接收的工作。对于 SerialPort 控件，打开或者关闭串口还是比较简单的，如下所示：

```
serialport.Open(); //打开串口
serialport.Close(); //关闭串口
```

Serial 类调用重载的 Write 和 WriteLine 方法发送数据，都是将数据写入串口输出缓冲区，其中 WriteLine 可发送字符串并在字符串末尾加入换行符[9-12]。读取串口缓冲区的方法有许多，其中除了 ReadExisting 和 ReadTo，其余的方法都是同步调用，线程被阻塞直到缓冲区有相应的数据或大于 ReadTimeOut 属性设定的时间值后，引发 ReadExisting 异常。

接收数据则采用 DataReceived 事件，在接收到了属性 ReceivedBytesThreshold 中设置的字符个数或接收到了文件结束字符并将其放入了输入缓冲区时被触发；要合理的设置 ReceivedBytesThreshold 的值，若接收的是定长的数据，则将 ReceivedBytesThreshold 设为接收数据的长度，若接收数据的结尾是固定的字符或字符串则可采用 ReadTo 的方法或在 DataReceived 事件中判断接收的字符是否满足条件[13-14]。

4.2.3　委托、多线程的利用

委托是一种可以把引用存储为函数的类型，可实现进程间的数据传递。委托的声明非常类似于函数，但不带函数体，且要使用 delegate 关键字。委托的声明指定了一个函数名，其中包含一个返回类型和参数列表。在定义了委托后，就可以声明该委托类型的变量，接着把这个变量初始化为与委托有相同签名的函数使用，随后就可以使用委托变量调用这个函数。

由于 DataReceived 事件在辅线程被引发，当收到完整的一条数据，返回主线程处理或在窗体上显示时，属于跨线程的处理。在该通信系统中，包含了接收数据并显示的功能，因此可采用控件异步委托的方法 Control.BeginInvoke 或者同步委托的方法 Invoke。本系统采用的是方法 Invoke：

```
this.textBoxHM650Display.Invoke(
new MethodInvoker(
delegate
{
    //用户添加其他处理程序，得到要显示的数据
this.textBox.AppendText(receiveData);
}));
```

在进行程序下载时,由于数据量比较大,执行文件下载函数的时间就比较长。若不另开一个新线程来执行此操作,则上位机的界面会出现类似于"死机"的现象,直到文件下载完成才会恢复正常。为避免出现该问题,需要使用多线程来优化该通信系统。

线程是进程的一个执行单元,是操作系统分配给CPU时间的基本单元。使用多线程可以同时完成多个任务,可以使程序的响应速度更快,让占用大量处理时间的任务或当前没有进行处理的任务定期将处理时间让给别的任务,也可随时停止任务,并为每个任务配置优先级以优化程序性能[15]。

在使用多线程时要添加命名空间System.Threading,一个Thread实例管理一个线程。通过简单实例化一个Thread对象,便创建一个线程,然后通过Thread对象提供的方法对线程进行管理。

新建一个线程,只需将其声明并为其提供想成起始点处的方法委托,再用Thread.Start()方法启动该线程,具体步骤如下:

首先是声明:Thread a;

其次是实例化:a = new Thread(new ThreadStart(b));其中,b为新建过程中执行的过程名;

最后则是调用Thread.Start()方法启动该线程a.Start()。

在本系统中,进行程序下载时,程序是如下设计的:

```
Thread aa = new Thread(senddata); // senddata()包含整个程序的下载协议
if (serialport.IsOpen)
{
aa.Start();
//senddata2(); // senddata2()仅为程序数据的下载
}
else
{
MessageBox.Show("Please open your port!");
}
```

4.2.4　检验报错

定时器的使用是为了增强本系统的报错功能。在接收应答数据时,若接收到的数据是错的,则可直接显示,供用户纠正。但是,若上位机一直接收不到应答指令,即应答数据,则整个文件下载过程无法继续进行,系统一直处于等待状态,且不会报错。在这种情况下,若使用定时器,则每次定时时间到后去检查是否接收到数据,若没有则强制终止当前过程,并报错没有接收到数据。该功能的实现只需添加Tick事件,再设置合适的定时时间即可。

5　实验结果与分析

本文中所设计的串口通信系统的界面如图3和4所示。经调试,用RS232串口线将PC端上位机和CPU板相连,将串口打开后,两者便可进行正常的通信。

图3　串口通信系统界面1

图4　串口通信系统界面2

为CPU板能自动控制执行机构,首先要做的是下载驱动程序。因此,先点击"浏览"按键,在PC上选择好所要下载的编译文件,然后点击"下载"按键,上位机便按照既定协议发送数据,下载文件时会将应答过程实时显示,进度条则实时显示文件下载的进度。

点击不同的功能按键,上位机则会发送对应的数据至CPU板,MCU处理数据后边控制执行机构,例如可控制电机的启停、转向、转速。执行机构运行后,会将相应的数据发送给上位机,并在显示区域进行显示,便于工作人员分析处理,体现了人机交互的友好性。

本系统还包括报错功能,在调试过程中,已验证了这一功能。在任何操作过程中,若上位机发送错误数据,下位机会将报错并返回给上位机,若上位机未接收到返回指令或者接收到错误数据,也会报错,一旦报错则终止当前动作,以保证通信过程的正确性。

6　结　　论

串口通信在远程数据采集、设备通信等方面广泛应用。针对CPU板需更新驱动程序,单步控制、调试执行机构,收发数据并实时显示等要求,利用OpenFileDialog和SerialPort这两个重要控件进行PC端上位机的设计,并利用RS232串口线连接CPU板,以此构成一个完整的串口通信系统。经实际运行,该通信系统能实现所有功能,具有良好的人机交互界面,是符合设计要求的。

对于类似的串口通信系统，仅需修改通信协议、增加或修改一些功能按键，便可通过 RS232 串口线控制不同 CPU 板所连接的执行机构，具有较强的移植性，有很强的实用价值。

参考文献

[1] WANG W Q. Implementation of Serial Port Communication System Based on SerialPort [J]. Science Mosaic, 2011(5):21-23.

[2] LU H F, JIANG CH Y, YANG X G. Key Technical Research of On-line Monitor System Based on Serial Communication [J]. Chinese Journal of Scientific Instrument, 2006, 27(3):2043-2044.

[3] ZHAI X SH, WANG B X, FAN M. Programming of serial communication based on visual C # [J]. Electronic Science and Technology, 2011, 24(2): 24-26.

[4] JIANG T, ZHANG J P. Design and implement for user-control software of data acquisition system based on C#[J]. Electronic Test, 2009(9):58-60.

[5] LI X, LIAO W. Serial communication protocols of commonly used equipment and their applications[J]. Process Automation Instrumentation, 2011, 32(10): 82-86.

[6] KARLI W, CHRISTIAN N. Beginning visual C # 2005[M]. BeiJing: Tsinghua University Press, 2006.

[7] FAN SH R. Visual C # 2008 controls usage examples Comments [M]. BeiJing: Tsinghua University Press, 2009.

[8] XIAO L L, LV W ZH, LI X F, et al. The design of pyroelectric coefficient testing system based on serial communication [J]. Chinese Journal of Scientific Instrument, 2005, 26(8):367-368.

[9] WANG A M, LIU G J, L X. Real-time monitoring system based on serial communication between PC and AI regulator using C # [J]. Industrial Control Computer, 2006, 19(7):62-63.

[10] XU L L, YU Z R. Communication between SR23 and PC based on serial port in C #. NET[J]. Computer and Modernization, 2011(5):107-109.

[11] DING J M, QU X H, SONG L M. Application of serial port in industry field multi-data-measuring[J]. Journal of Electronic Measurement and Instrument, 2005, 19(2):83-87.

[12] ZHANG J CH. Research and implementation of serial communication strategy based on C # [J]. Journal of LiaoCheng University, 2010, 23(1):100-103.

[13] WU Y F. Implementation of serial communication between PC and PLC by serial port [J]. Internet Fortune, 2009(2):156-157.

[14] HAN ZH H, ZHAO ZH L, DING ZH W. Implementation of serial communication between PC and IC read-writer based on C # serialport class[J]. Science Mosaic, 2007(3):79-81.

[15] WU X L, LIU CH SH. Programming of serial communication design and implement based on multi-thread [J]. Control Engineering of China, 2004, 11(2): 171-173.

作者简介

周阳，在读研究生。主要研究方向是嵌入式软件开发。E-mail: usstyang8023@163.com