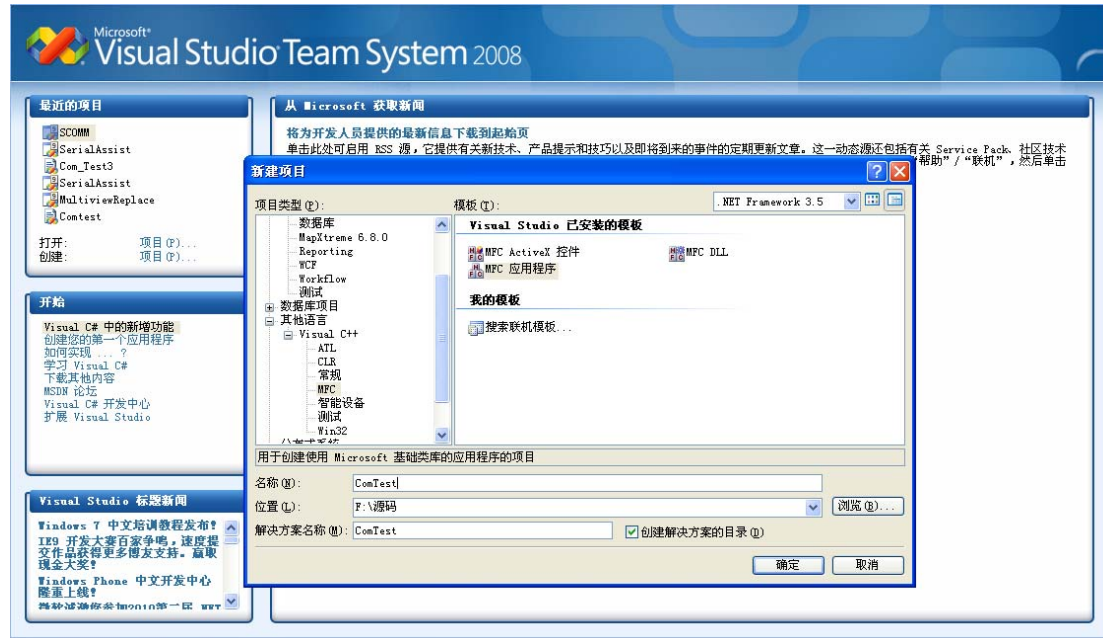


VS2008 上位机串口通信简单例程

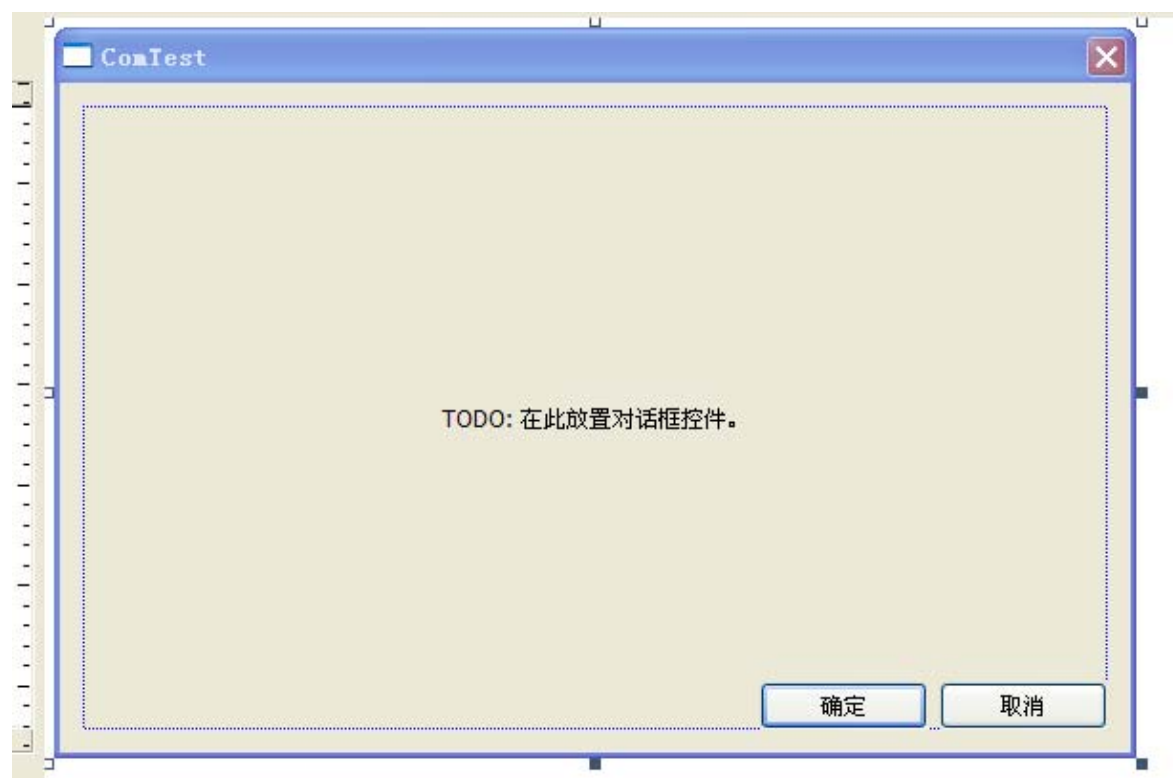
首先，在 vs2008 环境下创建 MFC 运用程序



设置项目名称为 ComTest（这个地方随意命名，根据个人习惯），点击确定后，**点击下一步**出现如下界面：



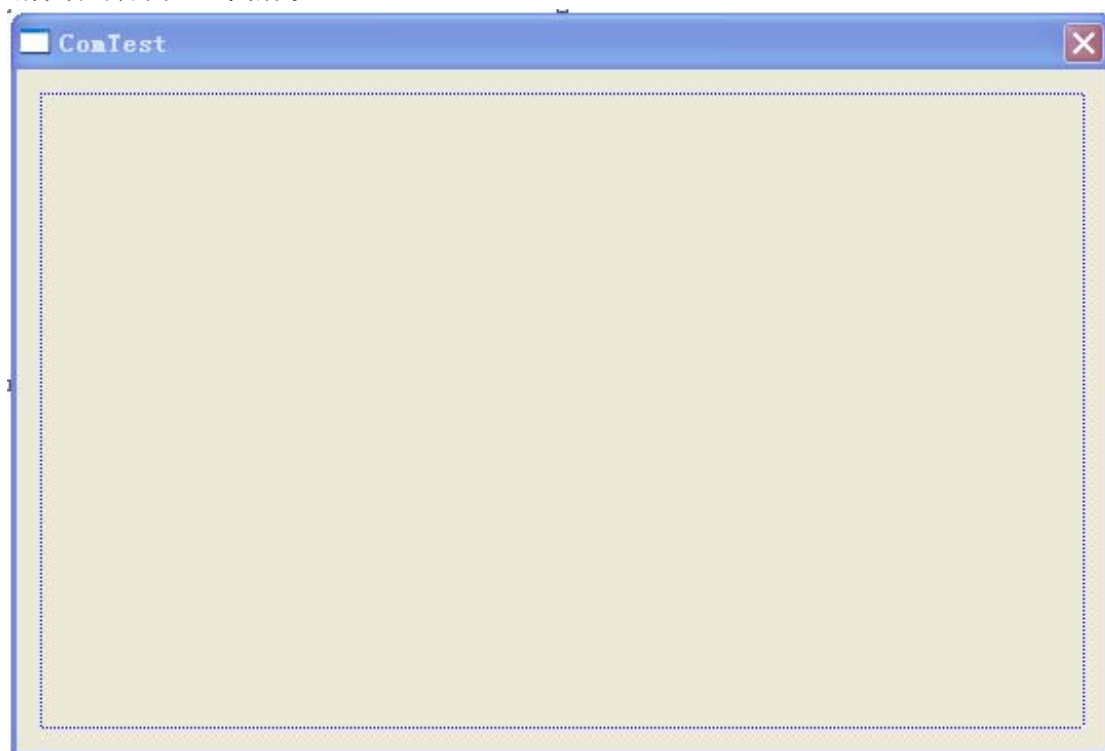
选择“基于对话框”模式然后直接点击完成即可（其他选项按默认方式），点击完成后出现如下界面：



解决资源管理器中自动给你生成好代码目录（可点击菜单栏“视图”选项打开解决方案资源管理器），如下图所示



我们再次回到对话框编辑窗口, 删除自动生产的控件(静态文本控件、确定和取消按钮控件), 删除后的界面如下图所示:

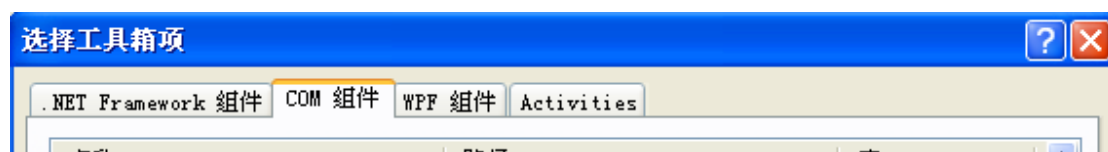


然后我们根据工具栏来选择添加我们所需要的控件, 在添加控件之前我们先来把串口通信控件加入到工具箱中, 因为默认的工具箱是不带 MS 串口通信控件的。添加方法如下:

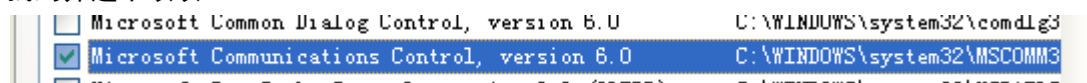
在工具箱界面点击鼠标右键出现如下界面:



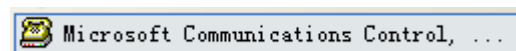
然后点击**选择项**出现如下界面, 然后选择“COM 组件”



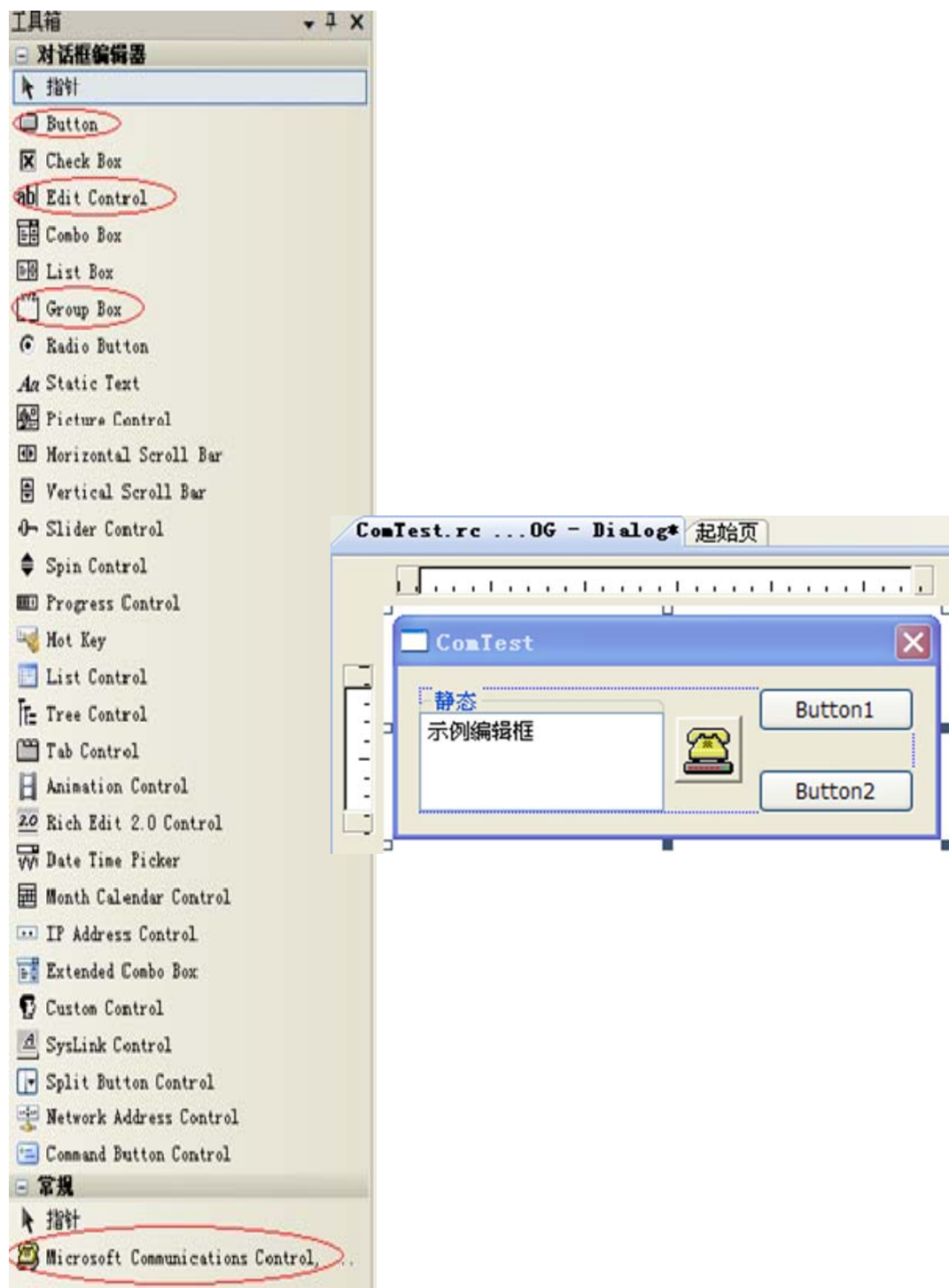
找到并选中该项



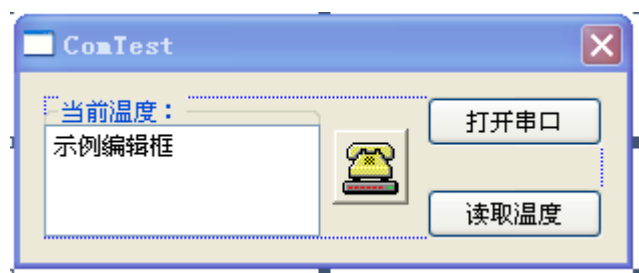
最后点击确定键，就成功添加 MS 串口通讯控件了，工具箱中就会出现串口控件图标了



然后我们回到对话框编辑界面，添加控件并调整对话框大小，最终如图所示

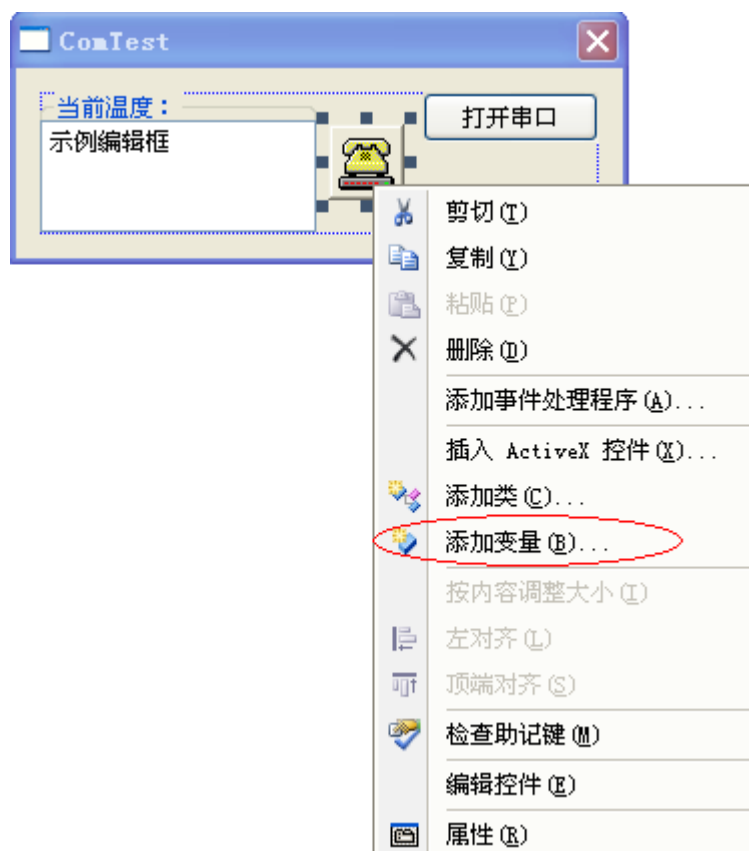


然后我们修改各控件的属性，得到如下效果



基本界面到此算是完工了，下面将进入代码阶段，没有基础的同学可能看起来会比较吃力，建议多看几遍，一步一来。

首先为 MS 串口控件添加一个变量，在对话框编辑窗口中，鼠标右键点击 MS 串口控件。如下图所示



然后将出现如下对话框:

添加成员变量向导 - ComTest

欢迎使用添加成员变量向导

访问 (A): public ☒ 控件变量 (Q)

变量类型 (V): Cmscomm1 控件 ID (I): IDC_MSCOMM1 类别 (T): Control

变量名 (N): m_Com 控件类型 (Y): OCX 最大字符数 (X):

最小值 (U): 最大值 (E):

.h 文件 (F): mscomm1.hcpp 文件 (P): mscomm1.cpp ...

注释 (M) (// 不需要 表示法):

完成 取消

变量名取名为 m_Com, 点击完成, 代码向导将会自动为你生产相关代码框架。

接下来同样的方法为两个按钮控件: 打开串口、读取温度 控件添加变量 m_OpenSerial、m_ReadData (变量命名规则网上有很多, 具体方法可根据自己喜好, 也可请自行查阅命名规范)。

变量类型 (V): CButton 控件 ID (I): IDC_BUTTON1 类别 (T): Control

变量名 (N): m_OpenSerial 控件类型 (Y): BUTTON 最大字符数 (X):


变量类型 (V): CButton 控件 ID (I): IDC_BUTTON2 类别 (T): Control

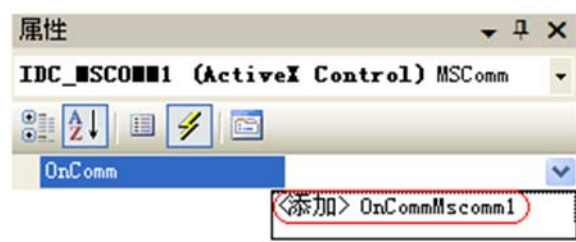
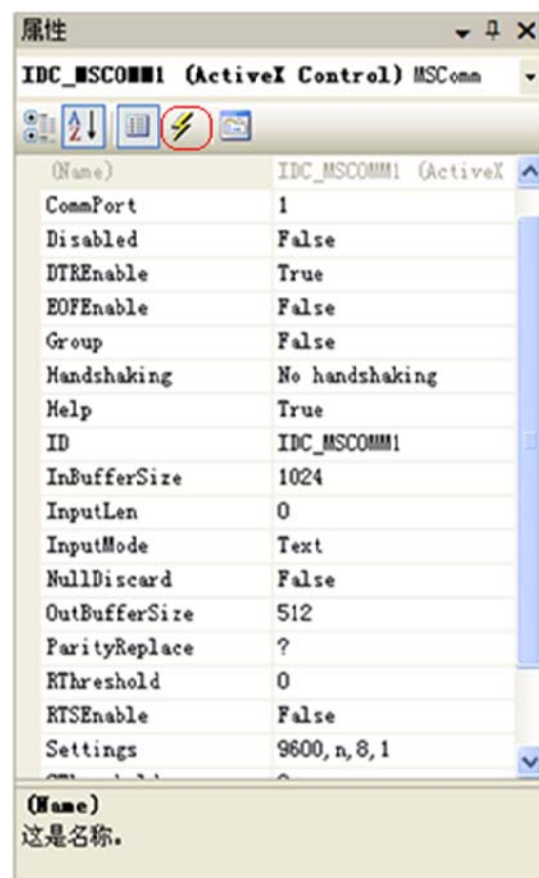
变量名 (N): m_ReadData 控件类型 (Y): BUTTON 最大字符数 (X):

同样为 文本编辑框 控件添加变量 m_ReceiveData

变量类型 (V): CString 控件 ID (I): IDC_EDIT2 类别 (T): Value

变量名 (N): m_ReceiveData 控件类型 (Y): EDIT 最大字符数 (X):

我们再次回到对话框编辑界面，鼠标左键点击**串口控件**按钮，然后在对应的属性栏 点击**控件事件按钮** 



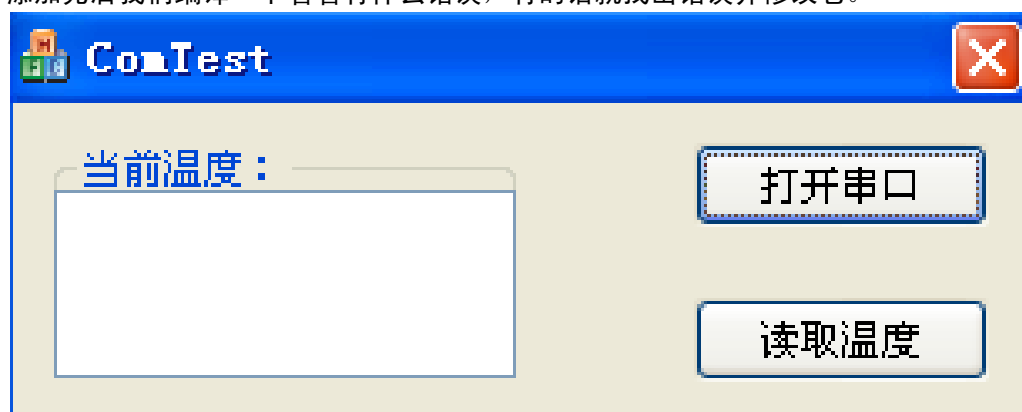
然后点击<添加>OnCommMscomm1 就会自动生成好代码框架，如下图所示



然后我们在// TODO: 在此处添加消息处理程序代码下添加如下代码:

```
// TODO: 在此处添加消息处理程序代码
VARIANT variant1; //定义VARIANT型变量, 用于存放接收到的数据
ColeSafeArray safearray; //定义safearray型变量
LONG len, k; //定义长整型变量len, k
BYTE rxdata[2048]; //定义BYTE型数组
CString stremp1; //定义一个字符串
if(m_Com.get_CommEvent() == 2) //判断引起OnComm时间的原因
{
    //如果是接收到特定个字节数, 则读取接收到的数据
    variant1 = m_Com.get_Input(); //把接收到的数据存放到VARIANT型变量里
    safearray = variant1; //VARIANT型变量转换为ColeSafeArray型变量
    len = safearray.GetOneDimSize();
    for(k=0; k<len; k++)
    {
        safearray.GetElement(&k, rxdata+k);
        //得到接收到的数据放到BYTE型数组rxdata里
    }
    for(k=0; k<len; k++)
    {
        BYTE bt = (*(unsigned char*)(rxdata+k)); //读取AD转换的高字节
        if((k%2) == 0)
            if((k+1) < len)
            {
                gllen++; //全局的变量, 对接收到的转换结果的个数进行计算
                int temp = bt*4 + (*(unsigned char*)(rxdata+k+1)) >> 6;
                //高低字节合并成实际的转换结果, 注意转换结果是左对齐
                stremp1.Format(_T("%2.2f"), (2.56*temp/1024));
                //计算成实际温度值
                m_ReceiveData += stremp1;
                m_ReceiveData += " °C\r\n"; //字符串加单位°C后换行
            }
    }
}
SetDlgItemText(IDC_EDIT1, m_ReceiveData); //更新文本控件的显示
```

添加完后我们编译一下看看有什么错误, 有的话就找出错误并修改它。



我的代码没问题, 编译后出现上述界面, 接下来我们继续代码的编写

下面要编写的是对话框的初始函数 `BOOL CComTestDlg::OnInitDialog()`
我们在 `BOOL CComTestDlg::OnInitDialog()` 中找到 `// TODO: 在此添加额外的初始化代码`
然后在下面添加如下对话框函数初始化代码

`// TODO: 在此添加额外的初始化代码`

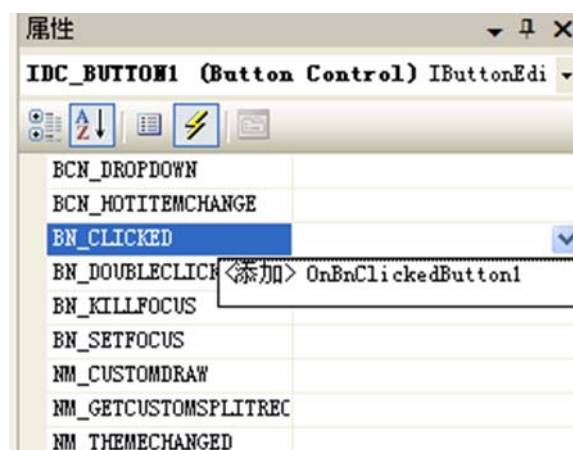
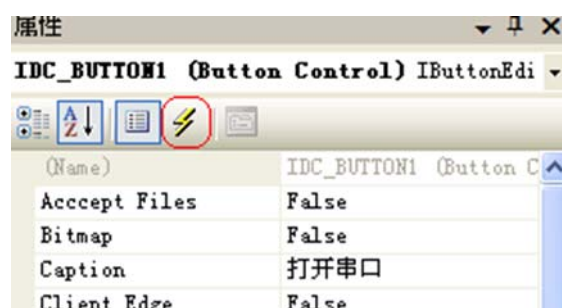
```
    gllen = 0;  
    //记录转换次数全局变量清零  
    m_Com.put_CommPort(2);  
    //选择串口号2(笔记本没有串口, 用的虚拟串口)  
    m_Com.put_PortOpen(TRUE);  
    //打开串口  
    m_Com.put_RThreshold(2);  
    //收到两个字节引发OnComm事件  
    m_Com.put_InputMode(1);  
    //输入模式选为二进制  
    m_Com.put_Settings(_T("9600,n,8,1"));  
    //设置串口参数, 波特率, 无奇偶校验, 位停止位, 位数据位
```

然后再次编译一下看看有没有错误, 出现错误就把它修改过来。

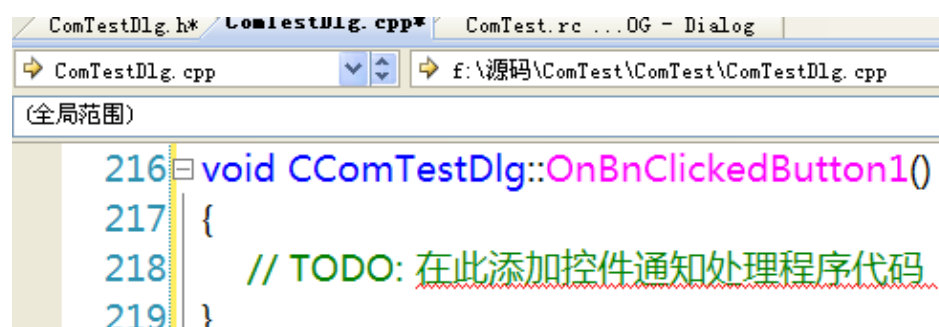
接下来我们来编写按钮控件的相关事件函数

回到对话框编辑界面, 鼠标左键点击 打开串口 按钮控件, 在其属性窗口点击事件图标

 , 然后在事件中选择 <添加> OnBnClickedButton1



点击添加后, 代码自动生成框架, 如下所示



然后我们找到// TODO: 在此添加控件通知处理程序代码, 在其下面添加如下代码


```
// TODO: 在此添加控件通知处理程序代码
if(! m_Com.getPortOpen())//判断串口是否已经打开
{
    m_Com.put_PortOpen(TRUE); //如果串口是关闭的 ,则打开
串口
    if (m_Com.getPortOpen())//如果串口打开成功
    {
        MessageBox(_T("串口初始化完毕"),_T("提示")); //提示
串口成功初始化
    }
    else MessageBox(_T("串口被占用"),_T("错误")); //如果串口
已经被打开 , 消息框提醒错误
    m_OpenSerial.SetWindowText(_T("关闭串口")); //按钮显
示状态改变
}
else
{
    m_Com.put_PortOpen(FALSE); //如果已经打开串口 , 则关
闭串口
    m_OpenSerial.SetWindowText(_T("打开串口")); //按钮显
示状态改变
}
```

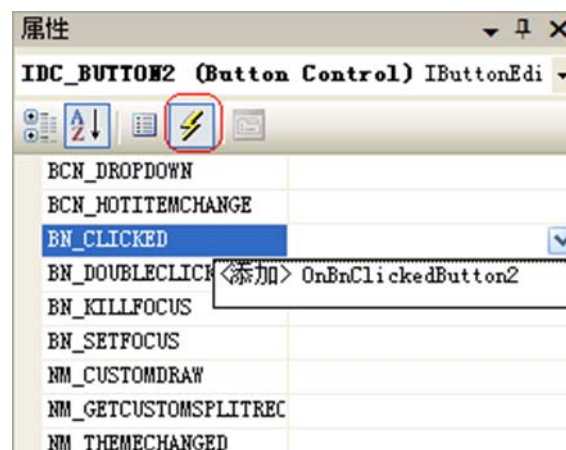
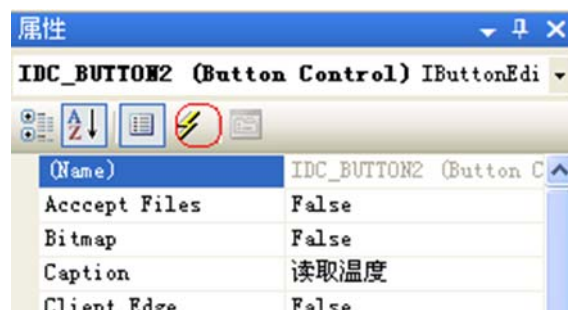
到此, 按钮控件 打开串口 的代码已经编写完了, 我们可以编译运行一下, 检查错误、看看实际效果



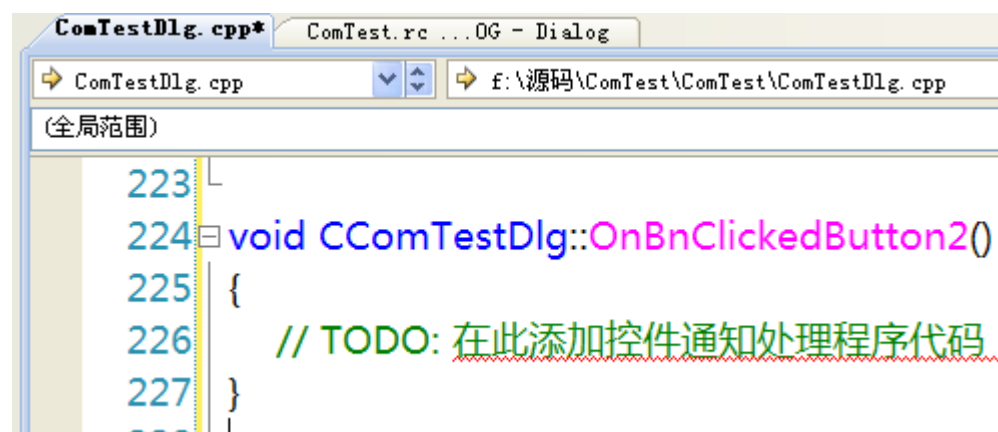
我们可以看出, 当我们点击打开串口按钮是 会弹出提示对话框, 提示串口初始化完毕, 然后 按钮的显示状态也随之改变, 如上图红圈中所示

至此我们的上位机基本上是完成了,但是还缺一个与单片机通讯的指令发送,细心的同学可能已经发现了,不是还有一个按钮控件的代码没有编写么,接下来我们进行上位机编写的最后一步,向单片机发送读取温度的指令,通过该指令读取单片机控制的温度传感器测得的温度值!

我们回到对话框编辑界面,鼠标左键点击 读取温度 按钮控件,在其属性窗口点击事件图标 , 然后在事件中选择<添加>OnBnClickedButton2



点击添加以后, 出现如下界面

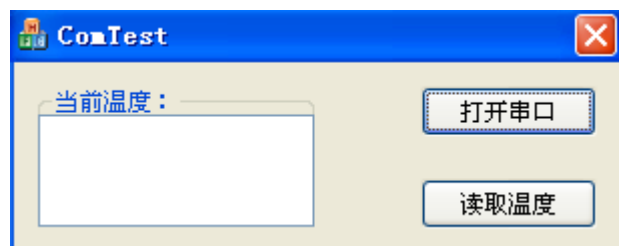


然后我们找到// TODO: 在此添加控件通知处理程序代码, 在其下面添加如下代码

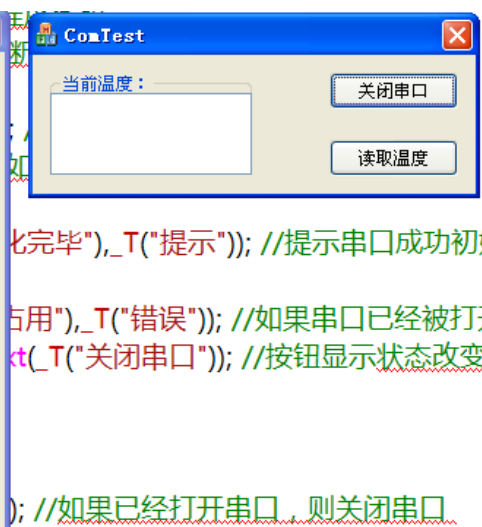
// TODO: 在此添加控件通知处理程序代码

```
CByteArray m_SendArray; //定义数组变量
m_SendArray.RemoveAll(); //发送前先清空数组
m_SendArray.SetSize(1); //设定数组维数为
m_SendArray.SetAt(0,0xff); //给数组赋值, 即 (与单片机通讯
的读取温度指令)
m_Com.put_Output (COleVariant(m_SendArray)); //强制转换
后才能发送
```

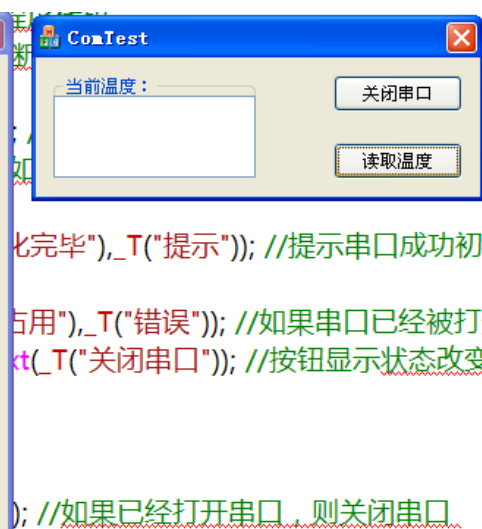
先编译一下, 看看有没有错误, 有错误的话找到错误并修改它, 本代码没问题, 编译运行后如下图所示



我们来测试一下, 打开一个串口调试助手(模拟单片机的串口), 设置好波特率, 我们来模拟一下通过串口上位机与单片机的通讯, 首先设置好参数, 如下图所示

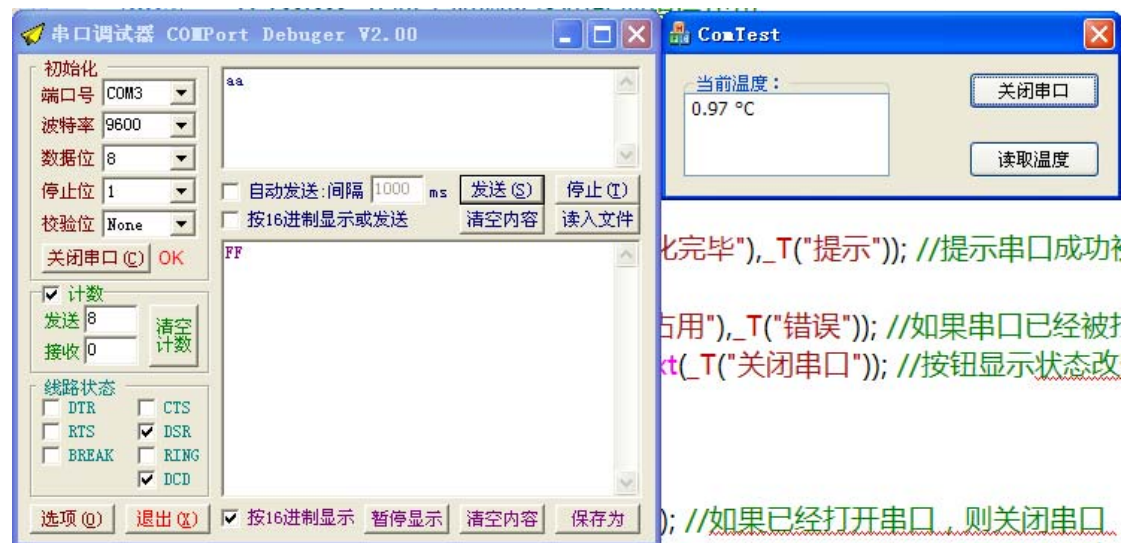


我们先通过上位机软件(即本教程所编写的上位机软件)来向单片机发送读取温度值的指令, 我们点击上位机的读取温度按钮, 理论上将会发送 0xff 指令到串口调试器(模拟单片机), 我们来看看实际运行效果, 如下图所示



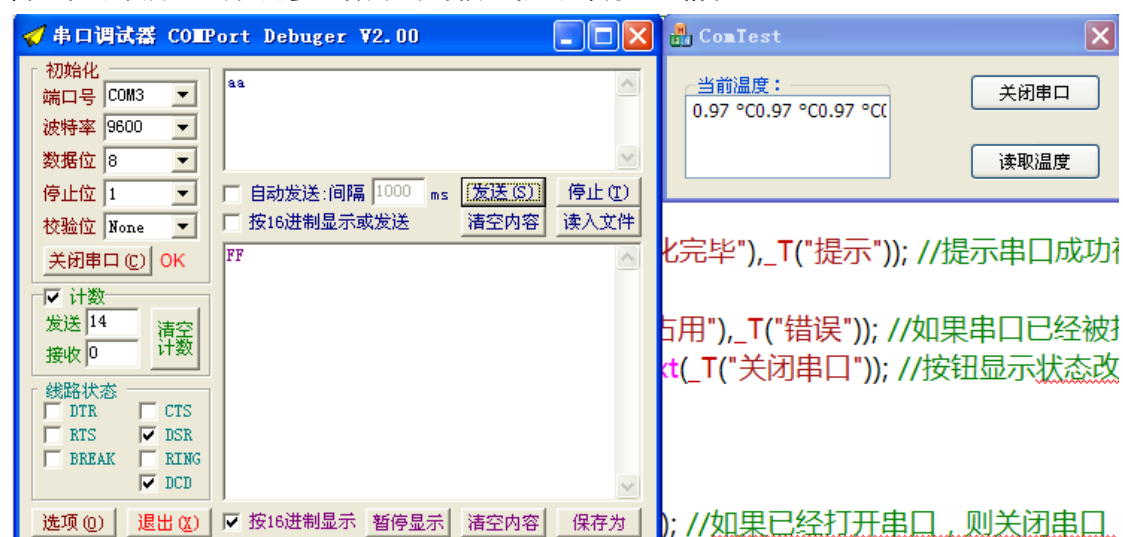
已经收到 FF 指令, 说明上位机向单片机读取指令, 单片机成功接收, 因为数据格式是按十六进制显示的, 所以 0xff 前面的 0x 就省略了

我们再来模拟单片机把温度值发送给上位机，比如我们发送 aa 温度数据指令到上位机

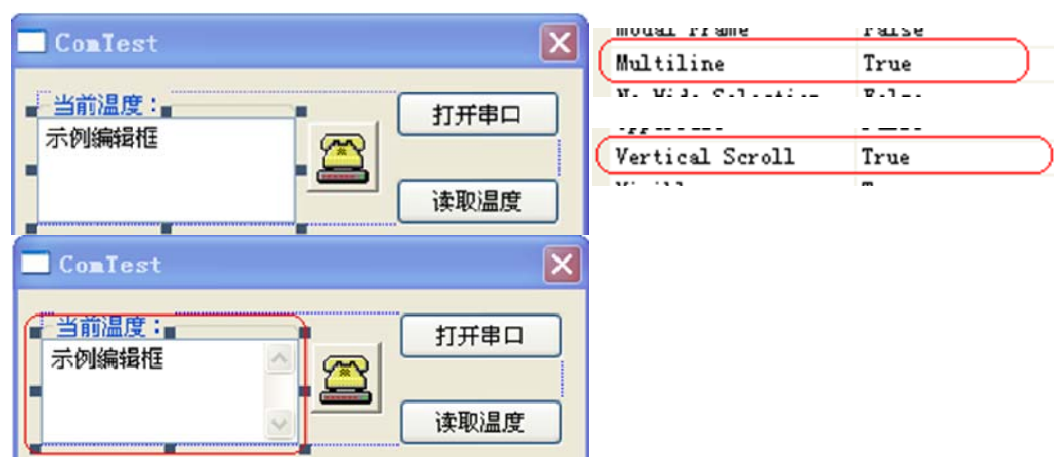


解算出来的温度是 0.97 摄氏度，这里的解算方法和单片机具体的数据格式设置有关。

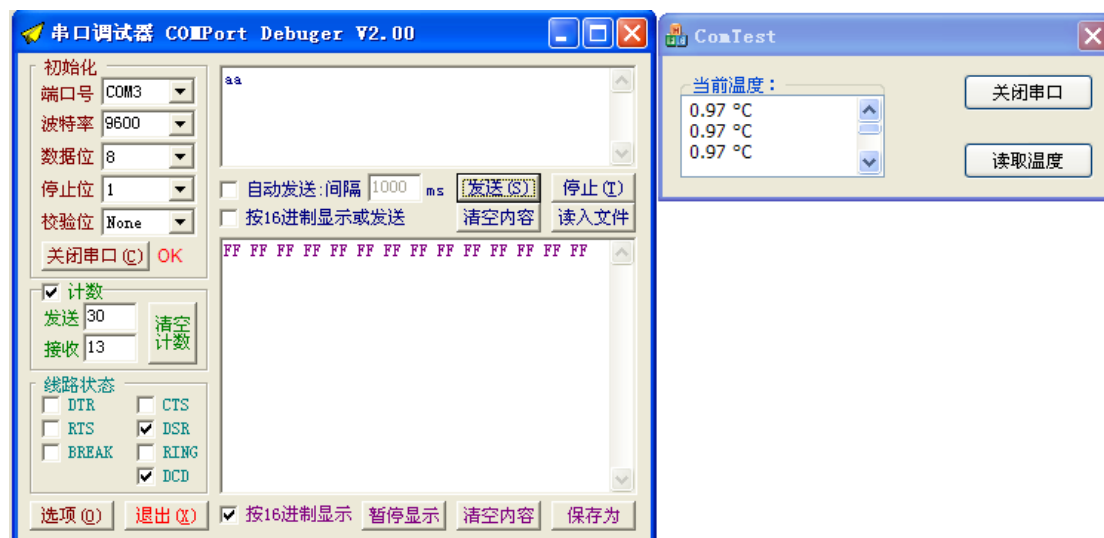
到此一个简单的串口软件已经制作完毕，我们多测试几组数据，就会发现上位机的数据接收窗口很不友好，当发送多组数据的时候，就会出现如下情况



后面的数据都看不到了，这时候我们只需要回到对话框编辑界面，鼠标左键点击文本编辑框控件，在其属性栏里更改 Multiline 选项，将其参数改为 Ture，然后更改 Vertical Scroll 选项，将其参数改为 Ture，效果如下图所示



这时候我们来测试发送多组数据时的情况，如下图所示



不管有多少组数据，我们都可以通过滚动条来查看数据！

后续章节预告：

《单片机与温度传感器温度的硬件电路设计调试及单片机程序编写》

制做者: bbsview

2012-6-13