**学号：1730401025**

**北京工商大学**

《嵌入式》实验报告

**院（系）计算机与信息工程学院**

**专 业 计算机应用技术**

**学生姓名 李庄子**

**成 绩**

**指导老师**

**2017 - 2018 学年第 1 学期**

# 作业一

**要求：**使用if…then…else结构编写一个判断命令行所传入参数大小的程序

将所输入数值存放在位置参数$1中

若$1>100,则输出：the number is greater than 100

若$1<10,则输出： the number is smaller than 10

否则输出： the number is between 10 and 100

# 运行结果：

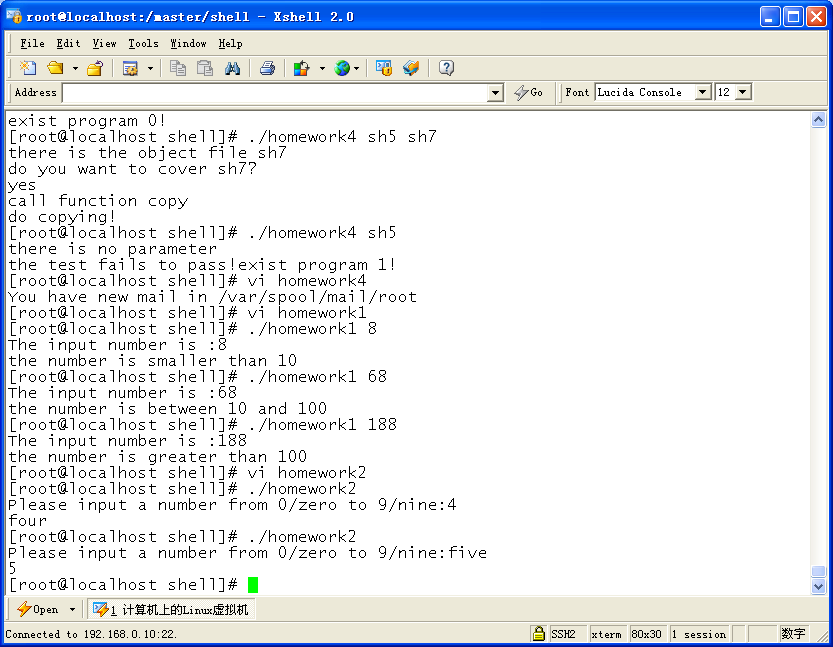
# 

# 作业二

**要求：**使用case结构编写一个实现0~9中英文数字转换的程序。

即输入0，显示zero，。。。

**运行结果：**



# 作业三

**要求：**请用until循环计算1到100的和。

# 运行结果：

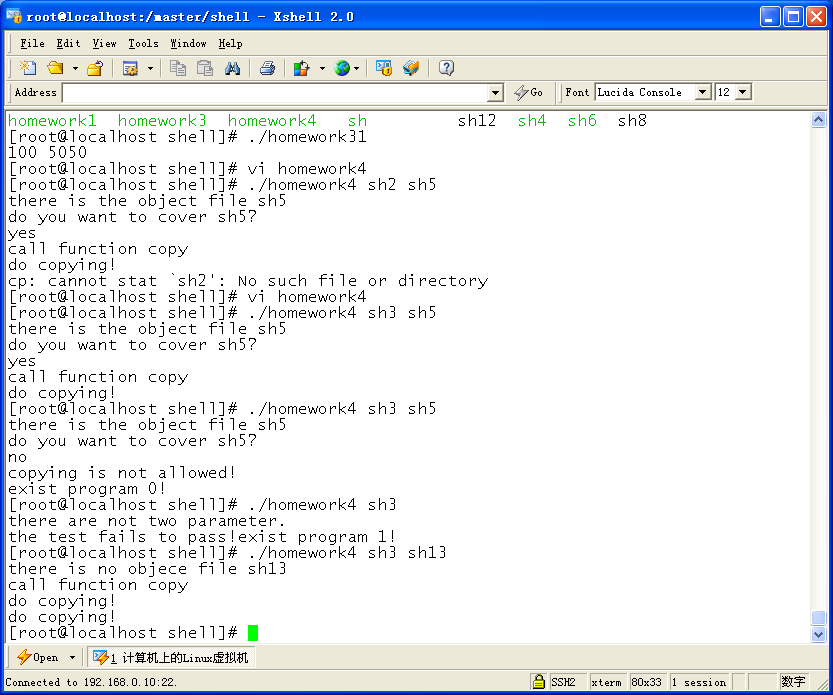
# 

# 作业四

**要求：**编写一个简易版本的cp命令，有两个参数，源文件和目标文件，如果目标文件已经存在，则ᨀ示用户，并让他选择是否继续执行复制，如果用户选择yes则继续，否则不继续

提示： 1、使用if结构 2、程序流程: 测试参数是否为两个（[ “$#” –ne 2] ），若测试不通过退出程序(exit 1)——测试目标文件是否已存在（[ -e “$2” ]）若不存在则执行copy（cp $1 $2 ），若已存在则询问是否覆盖——从标准输入读取用户响应（read answer）并判断响应是否为yes（[ “$answer” !=yes ]）,若不等于yes则输出提示：copy不允许，并退出程序（exit 0），若等于yes则执行copy。

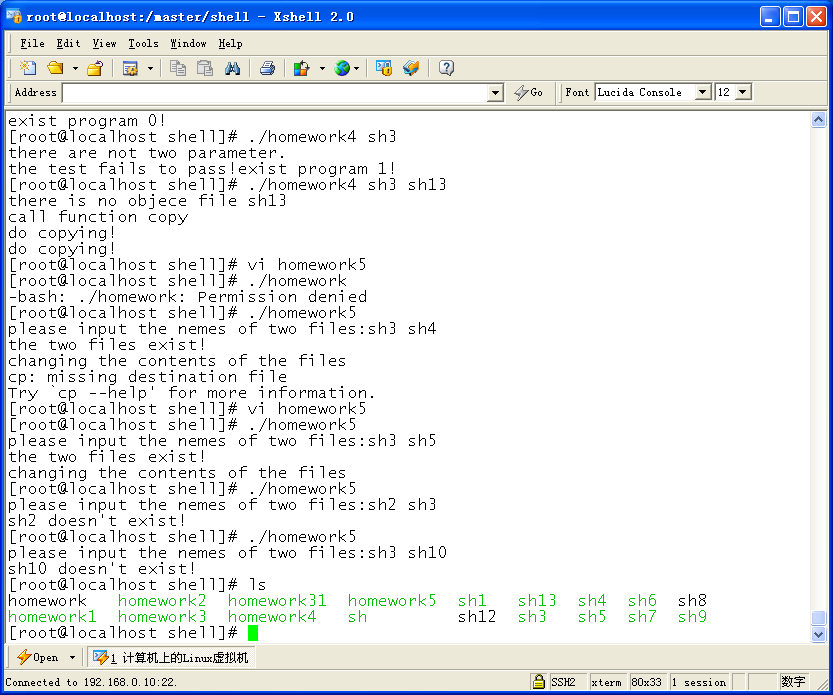
运行结果：



# 作业五

**要求：**编写一个Shell脚本，从键盘上接收两个文件名，如果两个文件都存在则交换两个文件的内容，否则应给出出错信息。

**运行结果：**



# 微型摄像头原理以及实现

# 实验目的

嵌入式系统（Embedded system），是一种“嵌入机械或电气系统内部、具有专属功能的计算机系统”，由于控制功能单一却重要，通常要求及时应对的实时计算性能。被嵌入的系统通常是包含数位硬件和机械部件的完整设备，举例而言汽车的防打滑煞车就是这样的一个功能集成。相反，通用计算机如个人计算机则设计灵活，可以全能处理各式各样的运算情况，以满足广大终端用户不同的需要。现在常见的很多设备都采用简单的嵌入式系统来控制，以达到最高效。

现代嵌入式系统通常是基于微控制器（如含集成内存和/或外设接口的中央处理单元）的；但普通微处理器（使用外部存储芯片和外设接口电路）也很常见，特别是在较复杂的系统中。通用型处理器、专门进行某类计算的处理器、为手持应用订制设计的处理器等，都可能应用到嵌入式系统。常见的典型专用处理器有数字信号处理器。

嵌入式系统的关键特性是专用于处理特定的任务，因此工程师能对其进行优化，以降低产品的体积和成本，提升可靠性和性能。

为了更进一步的理解嵌入式，本小组进行微型摄像头开发实验。

# 实验要求

本实验配有完整的文档，实验器材，程序，可利用官方提供的程序和文档，对摄像头进行命令的发送与数据接收。

# 实验原理

本次实验主要是串口通信原理，串口通信是指外设和计算机间，通过数据信号线 、地线、控制线等，按位进行传输数据的一种通讯方式。这种通信方式使用的数据线少，在远距离通信中可以节约通信成本，但其传输速度比并行传输低。串口通信（Serial Communications）的概念非常简单，串口按位（bit）发送和接收字节。尽管比按字节（byte）的并行通信慢，但是串口可以在使用一根线发送数据的同时用另一根线接收数据。它很简单并且能够实现远距离通信。比如IEEE488定义并行通行状态时，规定设备线总长不得超过20米，并且任意两个设备间的长度不得超过2米；而对于串口而言，长度可达1200米。典型地，串口用于ASCII码字符的传输。通信使用3根线完成，分别是地线、发送、接收。由于串口通信是异步的，端口能够在一根线上发送数据同时在另一根线上接收数据。其他线用于握手，但不是必须的。串口通信最重要的参数是波特率、数据位、停止位和奇偶校验。对于两个进行通信的端口，这些参数必须匹配。

# 实验结果

PTC08 摄像头上电初始化流程

（1）上电

（2）延时 2.5 秒

（3）复位指令

（4）设置拍照图片大小指令 （可选，如更改后需要复位一下才会生效）

（5）设置拍照图片压缩率指令（可选）

拍照片的过程

（1）发送清空图片缓存指令

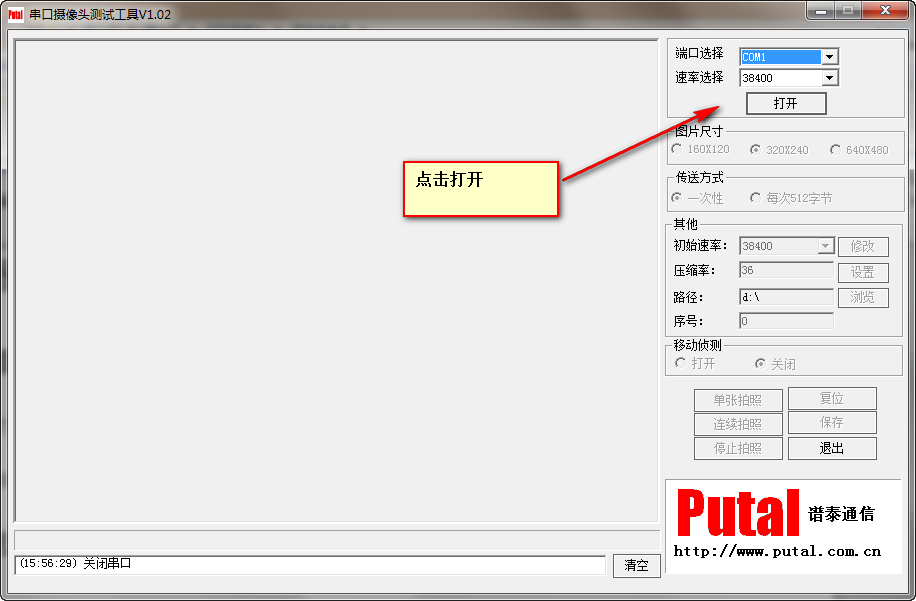
（2）发送拍照指令

（3）发送读取所拍图片长度指令

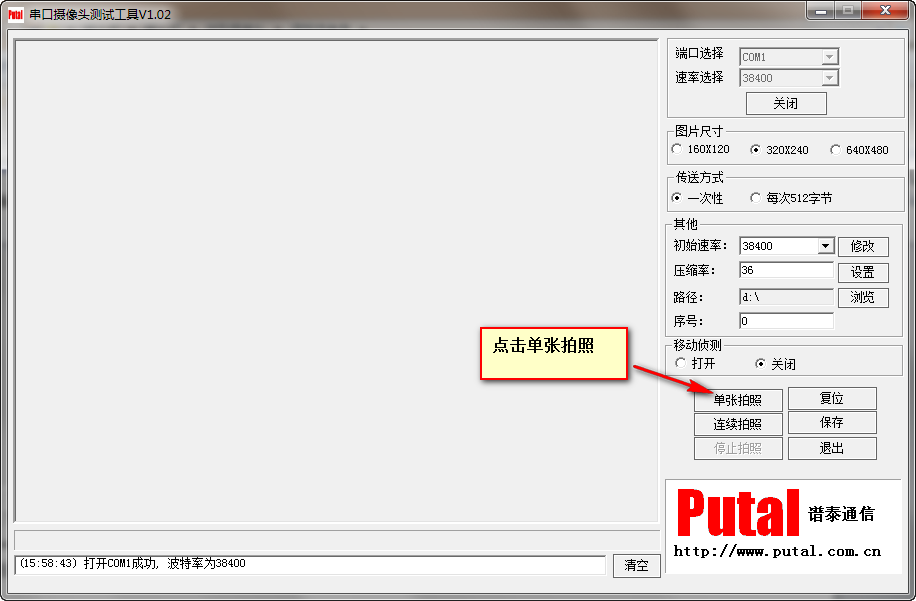
（4）发送读取所拍图片数据指令

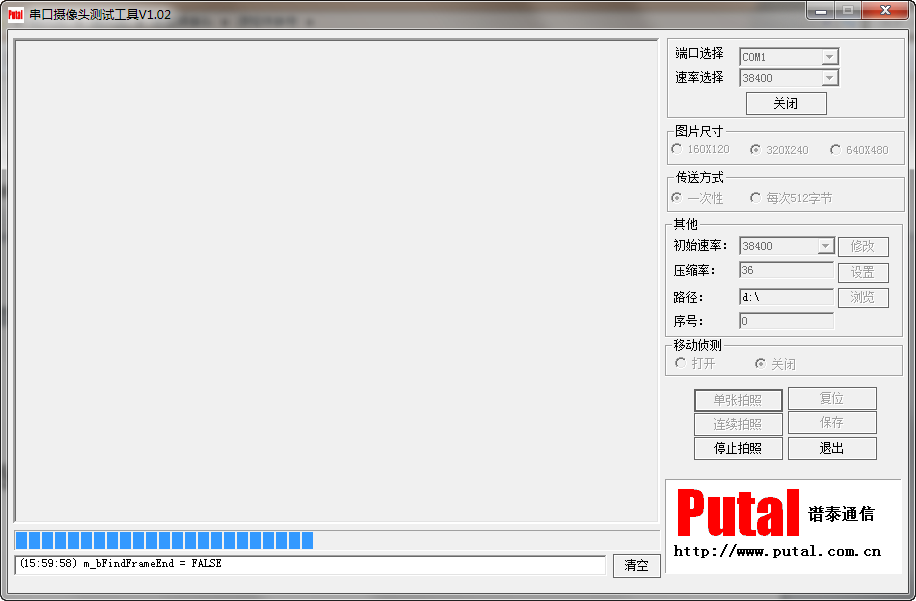
## 拍摄照片

* 打开串口



* 拍摄照片



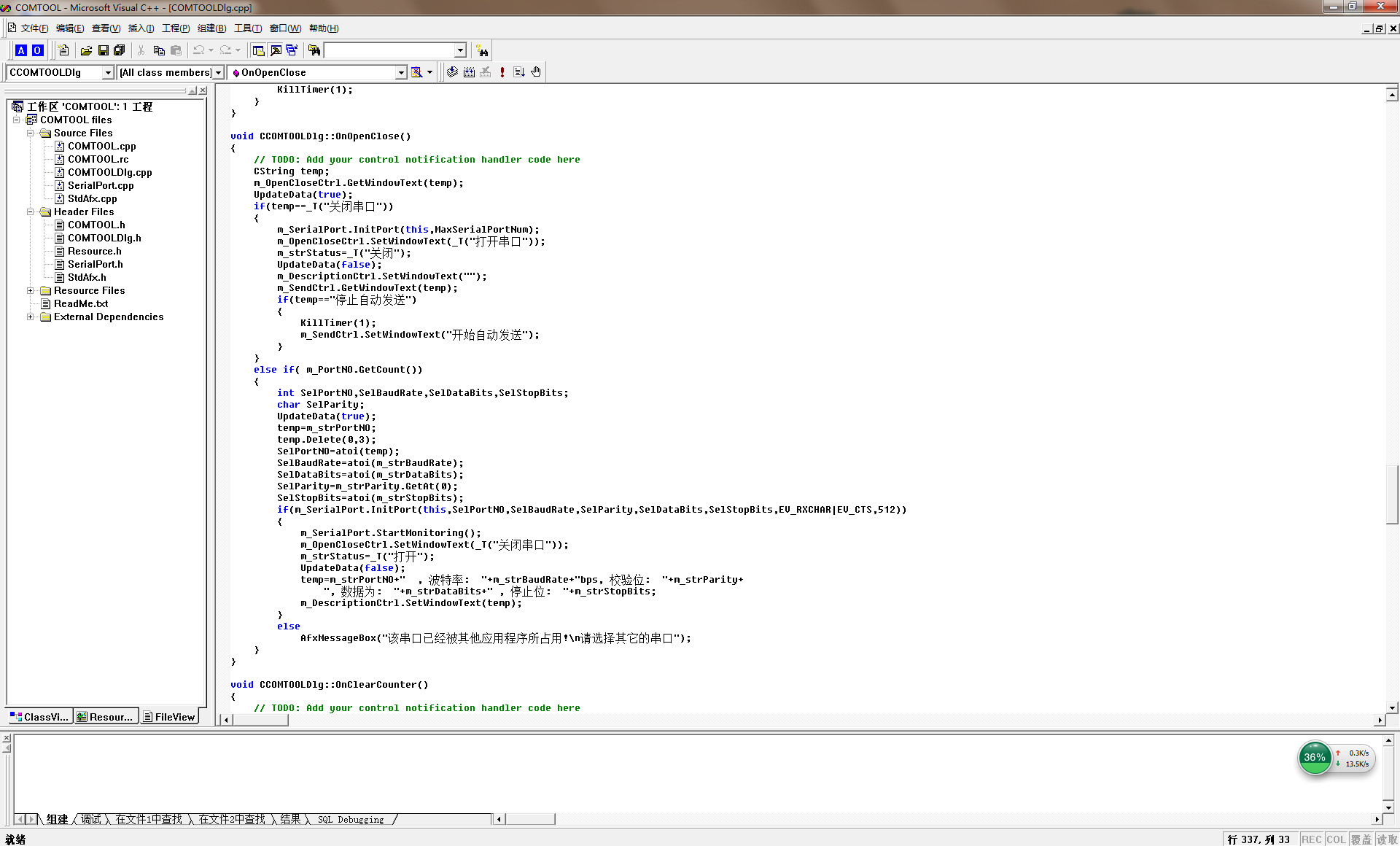


## 

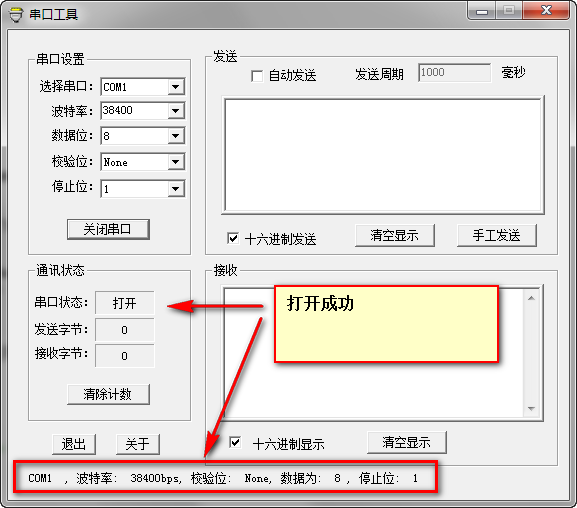
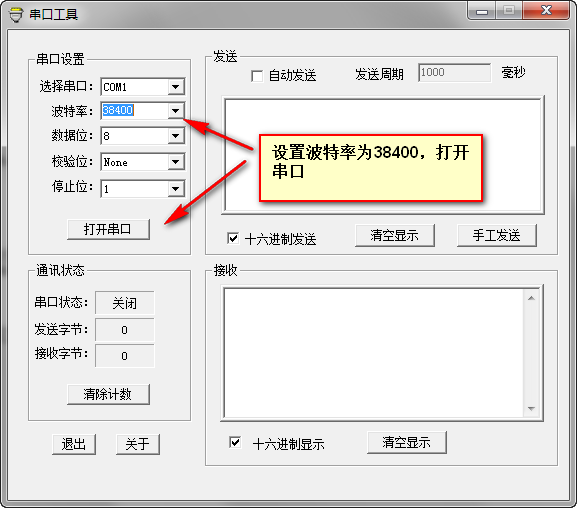
## 发送命令

上一次实验，我们进行了拍照实验，当我们点击拍照按钮时，实际上程序向摄像头模组发送了多条命令，讲数据接收后又进行了图片合成操作，下面来演示下如何与摄像头模组通讯。

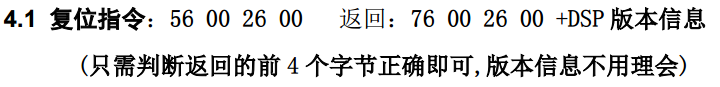
* 编译源程序

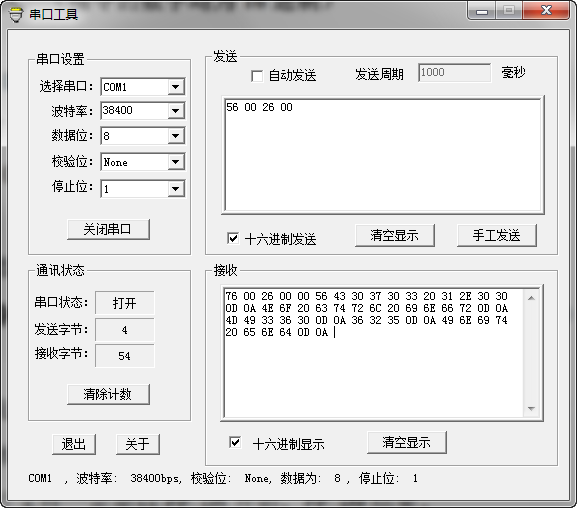


* 运行

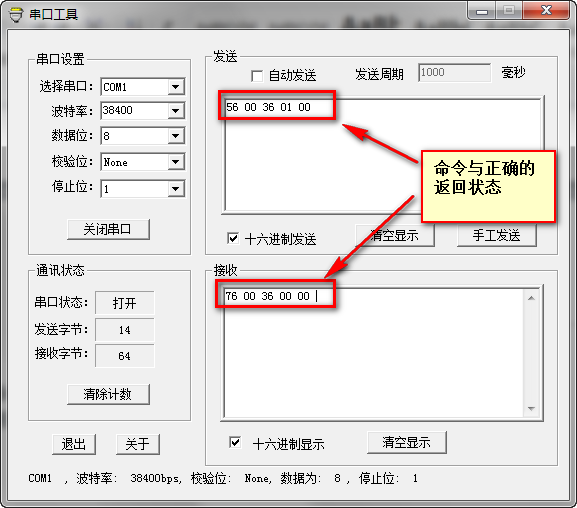


* 按照官方文档，发送复位命令





* 拍照指令



* 读所拍图片长度指令



返回：76 00 34 00 04 00 00 XX YY XX YY -------图片数据长度，XX 为高位字节，YY 为低位字节

* 读取所拍图片数据指令



# 关键程序代码

BOOL CCOMTOOLDlg::OnInitDialog()

{

CDialog::OnInitDialog();

// Add "About..." menu item to system menu.

// IDM\_ABOUTBOX must be in the system command range.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

CString strAboutMenu;

strAboutMenu.LoadString(IDS\_ABOUTBOX);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// Set the icon for this dialog. The framework does this automatically

// when the application's main window is not a dialog

SetIcon(m\_hIcon, TRUE); // Set big icon

SetIcon(m\_hIcon, FALSE); // Set small icon

// TODO: Add extra initialization here

m\_SendPeriodCtrl.EnableWindow(m\_bAutoSend);

m\_OpenCloseCtrl.SetWindowText(\_T("打开串口"));

m\_DescriptionCtrl.SetWindowText("");

if(m\_bHexS)

GetDlgItem(IDC\_SendEdit)->ModifyStyle(0,ES\_UPPERCASE,0);

else

GetDlgItem(IDC\_SendEdit)->ModifyStyle(ES\_UPPERCASE,0,0);

if(m\_bHexR)

GetDlgItem(IDC\_ReceiveEdit)->ModifyStyle(0,ES\_UPPERCASE,0);

else

GetDlgItem(IDC\_ReceiveEdit)->ModifyStyle(ES\_UPPERCASE,0,0);

CString temp;

//显示波特率

for(int i=0;i<sizeof(BaudRate)/sizeof(int);i++)

{

temp.Format("%d",BaudRate[i]);

m\_BaudRate.AddString((LPCTSTR)temp);

}

temp.Format("%d",9600);

m\_BaudRate.SetCurSel(m\_BaudRate.FindString(0,temp));

//显示奇偶校验

for (i=0;i<ParitySelNum;i++)

{

m\_Parity.AddString((LPCTSTR) Parity[i]);

}

m\_Parity.SetCurSel(m\_Parity.FindString(0,\_T("None")));

//显示停止位

for(i=0;i<sizeof(StopBits)/sizeof(int);i++)

{

temp.Format("%d",StopBits[i]);

m\_StopBits.AddString((LPCTSTR)temp);

}

temp.Format("%d",1);

m\_StopBits.SetCurSel(m\_StopBits.FindString(0,(LPCTSTR)temp));

//显示数据位

for(i=0;i<sizeof(DataBits)/sizeof(int);i++)

{

temp.Format("%d",DataBits[i]);

m\_DataBits.AddString((LPCTSTR)temp);

}

temp.Format("%d",8);

m\_DataBits.SetCurSel(m\_DataBits.FindString(0,(LPCTSTR)temp));

//显示串口设置

for(i=1;i<=MaxSerialPortNum-1;i++)

{

if(m\_SerialPort.InitPort(this,i))

{

temp.Format("COM%d",i);

m\_PortNO.AddString((LPCTSTR)temp);

}

}

if(m\_PortNO.GetCount())

{

m\_SerialPort.InitPort(this,MaxSerialPortNum);

m\_PortNO.SetCurSel(0);

}

return TRUE; // return TRUE unless you set the focus to a control

}

void CCOMTOOLDlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialog::OnSysCommand(nID, lParam);

}

}

void CCOMTOOLDlg::OnOpenClose()

{

// TODO: Add your control notification handler code here

CString temp;

m\_OpenCloseCtrl.GetWindowText(temp);

UpdateData(true);

if(temp==\_T("关闭串口"))

{

m\_SerialPort.InitPort(this,MaxSerialPortNum);

m\_OpenCloseCtrl.SetWindowText(\_T("打开串口"));

m\_strStatus=\_T("关闭");

UpdateData(false);

m\_DescriptionCtrl.SetWindowText("");

m\_SendCtrl.GetWindowText(temp);

if(temp=="停止自动发送")

{

KillTimer(1);

m\_SendCtrl.SetWindowText("开始自动发送");

}

}

else if( m\_PortNO.GetCount())

{

int SelPortNO,SelBaudRate,SelDataBits,SelStopBits;

char SelParity;

UpdateData(true);

temp=m\_strPortNO;

temp.Delete(0,3);

SelPortNO=atoi(temp);

SelBaudRate=atoi(m\_strBaudRate);

SelDataBits=atoi(m\_strDataBits);

SelParity=m\_strParity.GetAt(0);

SelStopBits=atoi(m\_strStopBits);

if(m\_SerialPort.InitPort(this,SelPortNO,SelBaudRate,SelParity,SelDataBits,SelStopBits,EV\_RXCHAR|EV\_CTS,512))

{

m\_SerialPort.StartMonitoring();

m\_OpenCloseCtrl.SetWindowText(\_T("关闭串口"));

m\_strStatus=\_T("打开");

UpdateData(false);

temp=m\_strPortNO+" , 波特率: "+m\_strBaudRate+"bps, 校验位: "+m\_strParity+

", 数据为: "+m\_strDataBits+" , 停止位: "+m\_strStopBits;

m\_DescriptionCtrl.SetWindowText(temp);

}

else

AfxMessageBox("该串口已经被其他应用程序所占用!\n请选择其它的串口");

}

}

void CCOMTOOLDlg::HexStringFilter(CString &str)

{

BOOL bOK;

for(int i=0;i<str.GetLength();)

{

bOK=((str.GetAt(i)>='0')&&(str.GetAt(i)<='9'))||

((str.GetAt(i)>='A')&&(str.GetAt(i)<='F'))||

((str.GetAt(i)>='a')&&(str.GetAt(i)<='f'));

if(!bOK)

str.Delete(i);

else i++;

}

}

char CCOMTOOLDlg::CombineHexChar(char CharH,char CharL)

{

char result;

CString temp;

if(CharH>='0'&&CharH<='9') result=(CharH-'0');

else if(CharH>='a'&&CharH<='f') result=(CharH-'a'+10);

else if(CharH>='A'&&CharH<='F') result=(CharH-'A'+10);

else result=0;

result<<=4;

if(CharL>='0'&&CharL<='9') result+=(CharL-'0');

else if(CharL>='a'&&CharL<='f') result+=(CharL-'a'+10);

else if(CharL>='A'&&CharL<='F') result+=(CharL-'A'+10);

else result+=0;

return result;

}

CString CCOMTOOLDlg::DevideHexChar(char HexChar)

{

CString result=\_T("");

int temp=(HexChar&0xF0)>>4;

if(temp<10)

result+=(temp+'0');

else

result+=(temp+'A'-10);

temp=HexChar&0x0F;

if(temp<10)

result+=(temp+'0');

else

result+=(temp+'A'-10);

return result;

}

# 心得体会

嵌入式系统的应用程序开发同在PC 机上开发应用程序的区别不是很大。对于Windows CE系统而言，Microsoft已经提供了较为完善的开发工具。特别是.NET Compact work的使用，使得基于Windows CE.NETCompactwork的应用程序有了跨平台性。开发人员可以使用Windows 的C# 语言直接在PC 上进行http://CE.NET应用程序的开发和模拟调试，也可将目标系统同PC 机相连，进行联机调试。现在有很多系统支持J2ME(JAVA的嵌入式系统版本) ，这使得JAVA 在嵌入式系统应用开发中占有较大的优势。另外，作为专业的嵌入式系统软件开发人员，还需要充分了解面向对象技术和设计模式等方面的知识。

本实验从拿到题目到实验结束，每个人都很积极的找资料、想方法，分工明确，配合紧密，我们才能在这么短的时间内高质量的完成实验。每个人切身体会到了什么是合作精神，什么是吃苦精神，这是我们无比珍贵的财富。最后感谢段老师的耐心指导！