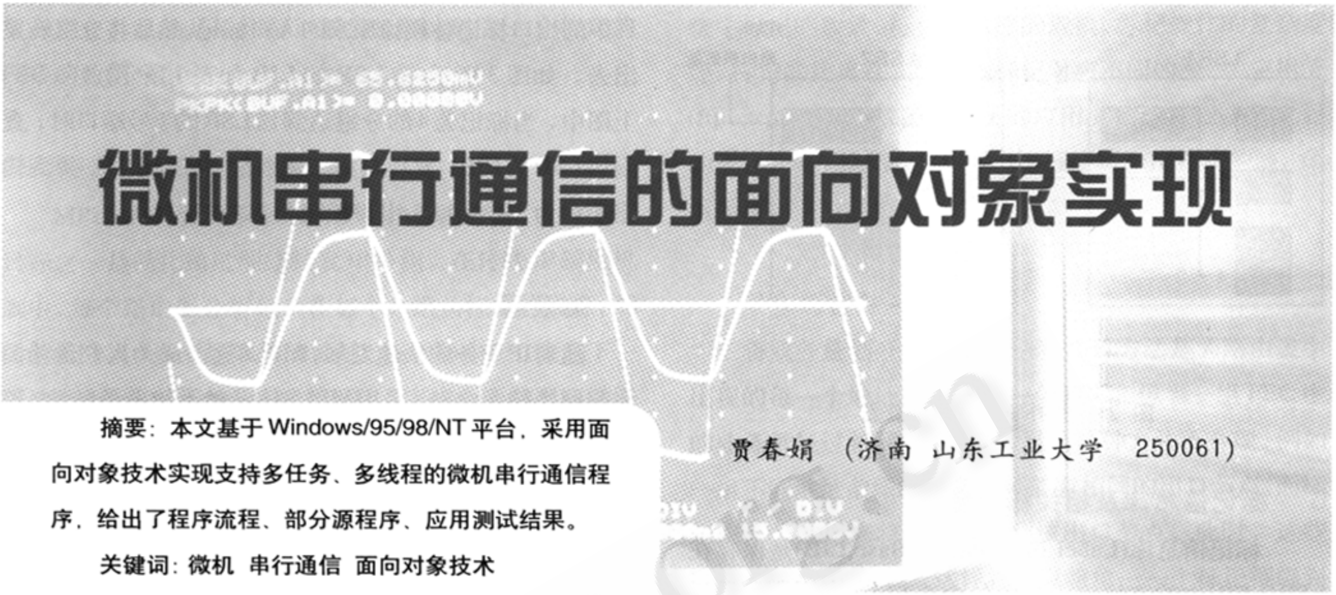

微机串行通信的面向对象实现

贾春娟 (济南 山东工业大学 250061)

摘要：本文基于Windows/95/98/NT平台，采用面向对象技术实现支持多任务、多线程的微机串行通信程序，给出了程序流程、部分源程序、应用测试结果。

关键词：微机 串行通信 面向对象技术

1 引言

微机监控系统中，传统的实现方式是基于DOS操作系统，常用中断或查询方式进行数据的发送与接收，是一种单任务的模式，CPU的利用率很低。Windows 3.x系统出现后，开始了多任务模式下通信程序开发，但是，采用的是传统的程序设计方法(即结构化程序设计)，程序设计时将数据和方法分离，其封装性、继承性都较差，很难重用。随着面向对象技术的出现，开发基于Windows 95/98/NT系统，支持多任务、多线程的面向对象的串行通信控制程序，是有现实意义的。

2 面向对象技术

面向对象技术是新一代软件工程的发展趋势，它在对对象进行抽象的基础上，通过类将不同层次的对象进行封装，利用类的继承性、多态性，则可以大大提高程序设计的效率和开放性。

面向对象技术的主要概念有：对象、类、方法和消息。类在定义数据的同时也定义了操作这些数据的方法；对象就是类的实例；向对象发出的服务请求称作消息，通过消息进行对象间的通信。封装使对象成为一些各司其职、互不干扰的独立单位；消息通信则为他们提供了唯一合法的动态联系途径，使他们的行为互相配合，构成一个有机整体。

面向对象技术的实现，主要有以下三种途径：

(1) 利用特殊的面向对象的编程语言，如：Smalltalk;

(2) 利用传统编程语言的面向对象的超集，如：C++、Object Pascal;

(3) 在原有编程环境下，开发具有面向对象机制的子程序或函数加调用。

3 串行通信的实现原理

串行通信程序的功能主要包括：一是从串口接收数据并送往接收缓冲区；二是将发送缓冲区的数据送往串口发送。从技术要求的角度，通信程序能支持多种速率通信，要求有高速数据处理能力，并且有良好的容错性。

3.1 串行通信程序中的类与线程

根据通信程序功能要求，及面向对象技术特点，程序设计时建立一个类：TuComm；两个线程：TreadCommThread、TWriteCommThread。下面分别介绍：

(1) TuComm类。用TuComm类对串口进行封装，其数据成员包括：串口号、数据位、停止位、波特率、以及串口与事项句柄。其主要的成员函数应包括：串口的启动、停止、数据的发送与接收、串口的初始化等。图1给出了TuComm类的定义。

(2)TReadCommThread线程。用TReadCommThread对串口的接收过程进行封装，其主要的数据成员包括：操作串口指针，接收停止指针，接收数据缓冲区，读数据事项，事项标志位等。其主要成员函数包括：读串口、设置读线程事项、置位读停止标志等。

(3)TWriteCommThread线程。用TWriteCommTh-read对串口的发送过程进行封装，其主要的数据成员包括：操作串口指针、发送数据缓冲区、发送停止标志等。其主要成员函数包括：写串口、置位停止标志等。

3.2 消息及事件的驱动过程

消息是面向对象技术中对象之间请求或相互协作的唯一途径，设计面向对象程序的另一个重要方面就是确定对象之间相互传递的消息。

```
Class TuComm
 {
 private:
    Char SzCommName  [40]; // 串口名称
    BYTE CommNo;              // 串口号
    BYTE ByteSize;            // 数据位
    BYTE StopBits;            // 停止位
    BYTE Parity;              // 奇偶校验
    DWORD BandRate;           // 波特率
    HANDLE hComm, hNewComm; // 串句柄
    HANDLE hPostEvent, hWriteEvent,
    hCloseEvent; // 事项句柄
    TReadCommThread *pReadThread; // 读线程
    TWriteCommThread *pWriteThread; // 写线程
 Public:
    Void DefReceiveData(BYTE *Buffer,DWORD dwlength);
    TuReceiveDataEvent OnReeceiveData;
    BOOL StartComm ();
    Void StopComm ();
    Void ReadPost;
    DWORD WriteComm (BYTE *pBuf,DWORD dwsize);
    Void InitCOmm ();
    Tucomm ();
    Tucomm ();
    }
```

图 1 TuComm 类的定义

在 TuComm 对象中，读、写线程对象间的同步就是靠事项来处理的。通过hPostEvent,hWriteEvent,hCloseEvent 三个事项，TuComm 对象和读、写线程对象相互协调，共同控制串行端口的通信进程。

4 程序设计及实现

本文采用前面提到的面向对象技术实现方法(2)，采用 C++ 面向对象的编程语言，使用 C++ Builder 4.0 开发工具，按上述设计原理进行编程实现。限于篇幅，将部分程序给出，图 2 为 TuComm 对象中 StartComm 方法的程序流程图，相应的源程序清单及注释列于表 1。

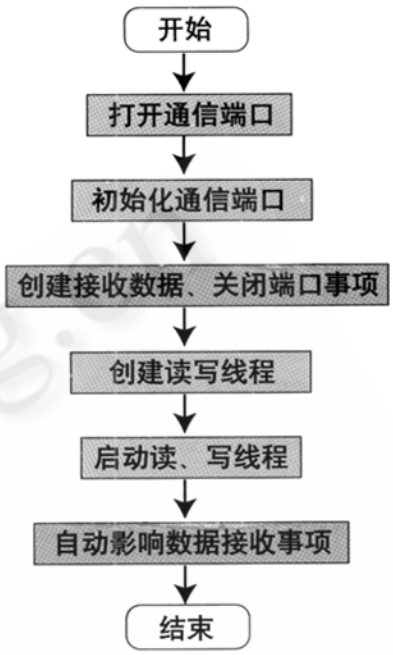

图 2 StartComm 方法的程序流程图

表 1 部分源程序清单

```
TUComm:: TUComm ()
{
  hComm=NULL;
  pReadThread=NULL;
  BaudRate=600;
  Parity=NOPARITY;
  ByteSize=8;
  StopBits=ONESTOPBIT;
  CommNO=3;
  sprintf(szCommName,"COM%d",CommNo);
  OnReceiveData=DefReceiveData;
}
//---------------------------------------------------
void TUComm::InitComm()
{
  DCB dcb;
  if (hComm !=NULL)
  {
    InitDCB(&dcb);
```

```
    dcb.Parity =Parity;
    dcb.BaudRate =BaudRate;
    dcb.ByteSize =ByteSize;
    dcb.StopBits =StopBits;
    }
}
//------------------
BOOL TUComm::StartComm()
{
COMMTIMEOUTS CommTimeOuts;
StopComm();
hNewComm =CreateFile(szCommName,
GENERIC_READ | GENERIC_WRITE,
      0,//exclusive access
      NULL,// no security attrs
      OPEN_EXISTING,
      FILE_ATTRIBUTE_NORMAL |
      FILE_FLAG_OVERLAPPED,NULL)
if (hNewComm !=INVALID_HANDLE_VALUE)
  hComm =hNewComm;
  if (hComm !=NULL)
  {
  SetCommMask (hComm,EV_RXCHAR);
  SetupComm(hCOmm,4096,4096);
  PurgeComm(hComm,PURGE_TXABORT | PURGE_
  RXABORT |
  PURGE_TXCLEAR |PURGE_RXCLEAR) ;
  CommTimeOuts.ReadIntervalTimeout=200;
//OxFFFFFFFF;
CommTimeOuts.ReadTotalTimeoutMultiplier=0;
CommTimeOuts.ReadTotalTimeoutConstant=0;
CommTimeOuts.WriteTotalTimeoutMultiplier=0;
CommTimeOuts.WriteTotalTimeoutConstant=0;
InitComm();
EscapeCommFunction (hComm,SETDTR);
hPostEvent=CreateEvent (0,TRUE,TRUE,NULL);
hCloseEvent=CreateEvent (0,TRUE,FALSE,NULL);
  pWriteThread=new WriteCommThread(this);
  pWriteThread->Priority=tpNormal;
  pReadThread=new TReadCommThread(this);
  pReadThread->Priority=tpNormal;
//tpLower;//tpNormal;//tpHigher;
  pReadThread->Resume();
  pWriteThread->Resume();
  }
  retuen (hComm !=NULL);
}
```

5 串行通信程序功能测试

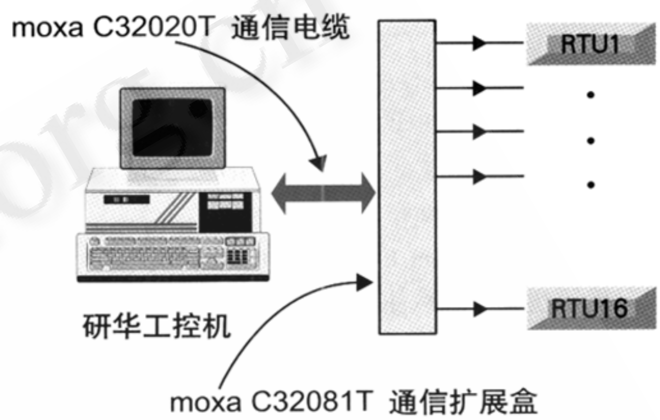

图 3 通信测试框图

如图3所示，将上述通信程序应用于电力监控SCADA系统中主机与多路RTU之间的通信。其中：主机采用研华工控机，CPU为PIII-450、64MRAM，并在主机中插入32路串行通信适配器(moxaC32010T)，通过moxa C32081T通信电缆接至16路仿真RTU相连，实现一台主机与16路RTU通信，RTU与主机之间的通信采用电力系统运动规约CDT或SC1801。测试结果显示，在RTU不断发送数据的情况下，主机的CPU负载率不超过10%。

6 结束语

实践证明，基于Windows 95平台并采用面向对象技术实现的串行通信程序，具有实时性好、数据获取准确，以及通信过程控制方便等特点；同时程序可移植性好、维护和扩充方便，因此，应用范围广泛。■

参考文献

1 徐新华 C++ Builder 3核心编程技术 北京希望电脑公司 1998.9

2 邵维忠 杨芙清 面向对象的系统分析 清华大学出版社 1998.12