基于 Java 的串口通信应用编程

丁振凡[1],王小明[1],邓建明[2],周 斌[2]
(1.华东交通大学,江西 南昌 330013;
2.南昌铁路局,江西 南昌 330001)

摘 要:介绍了 Java 实现串口通信编程的技术处理。着重就串口通信的连接、数据缓冲区资源的多线程访问控制以及数据读取中的超时控制等问题进行了详细讨论,有效地实现了主机与下位单片机之间的数据传递。该通信方式已用于基于工控机的绝缘电阻检测应用中。

关键词:Java;串口通信;多线程;comm 包

中图分类号:TP391 **文献标识码**:A **文章编号**:1674-7720(2012)13-0084-03

Research of Java_based programming for serial communcation

Ding Zhenfan[1], Wang Xiaoming[1], Deng Jianming[2], Zhou Bin[2]
(1.East China Jiaotong University, Nanchang 330013, China;
2.Nanchang Railway Bureau, Nanchang 330001, China)

Abstract: Introduce the Java programming process to implements serial comunication. detailly discuss about the connection of serial communication, and the multithread contorll for the access of data Buffer resource, and the expires time control in the data reading. effectively realize the data transmission between host and single chip microcomputer. This mode have used in insulation resistance detection application for Industrial control machine.

Key words: Java; serial communcation; multi thread; comm package

嵌入式系统或传感器网络的很多应用都需要通过 PC 机与嵌入式设备或传感器节点进行通信。其中,最常用的接口就是 RS-232 串口。串口通信可以是上位机与下位机之间的直接串口通信,也可以是在串口上连接无线通信模块,通过串口进行无线通信。RS-232-C 是在 1970 年由美国电子工业协会(EIA)联合贝尔系统、调制解调器厂家及计算机终端生产厂家共同制定的用于串行通信的标准。RS-232 是一个全双工的通信协议,它可以同时进行数据接收和发送的工作。

Java 实现串口通信的编程方式通常采用 SUN 发布的串口通信 API,它是以独立 jar 包形式提供的一个标准扩展。其中包含 3 个文件:comm.jar 提供了通信用的 java API;win32com.dll 提供了串口通信的本地驱动接口;javax.comm.properties 是这个驱动的类配置文件。Java 读写串口过程主要是调用 javax.comm 包中的 API 函数。在 javax.comm 包中,串口的读写操作是数据流形式,串口初始化后,通过 CommPort 类的 getInputStream()和 getOutputStream()方法即可分别取得端口的输入流和输出流。

串口通信应用程序有两种模式:一种是实现 SerialPortEventListener 接口,通过监听串口事件并作相应处理;另一种就是建立一个独立的接收线程负责数据的接收。本文采用的是后一种方式。

本文的串口通信是上位工控机与下位单片机之间的通信。工控机通过触摸屏方式来使用系统,下位单片机连接绝缘检测笔实现绝缘电阻的检测,并将检测过程的结果发送给上位机。本文仅介绍主机方的 Java 串口通信编程处理技术,如图 1 所示的虚线上方部分,包含消息接收线程、Swing 事件驱动应用界面、消息缓冲区、消息分析处理程序。其中:①在图形应用界面中通过用户的操作来触发事件实现与单片机的通信;②主机通过串口向单片机发送启动检测的消息;③单片机在收到消息后,将启动检测,并将传感器获取的数据通过串口发送给主机作为响应;④数据接收

线程将收到的数据放到一个缓存中；⑤消息分析处理程序从缓存中获取数据并进行分析处理。主机和单片机间每次通信传送 1 B。

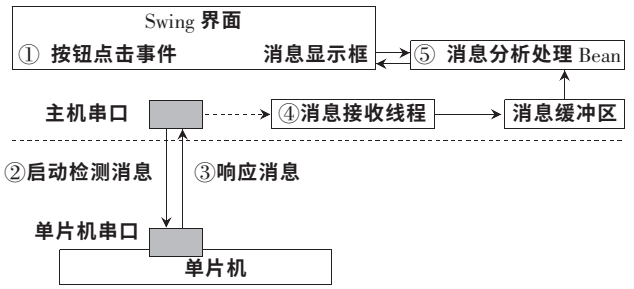

图 1 Java 串口通信的消息处理过程

1 串口缓冲区的控制

串口缓冲区(SerialBuffer 类)实现从串口接收到的一个完整消息的封装，本系统的消息按协议设计为 11 B，其中包含消息的起始标记、识别标识和数据字节。消息缓冲区是消息接收线程和消息分析处理 Bean 之间的桥梁，只有在接收到一条完整的消息后才可以进行消息的分析解析。串口缓冲区安排有 3 个重要属性：

(1) Content 属性：存放 11 B 的消息；

(2) Available 属性：标识消息是否可用；

(3) LengthNeeded 属性：统计收到的消息字节长度。

该类还定义了两个重要方法：(1) public synchronized byte[] GetMsg()：从缓冲区读取消息；(2) public synchronized void putbyte(int c)：写一个字节到缓冲区。方法定义中均含有 synchronized 关键词，也就是要使用这两个方法必须取得缓冲区的对象锁，从而实现对缓冲区这个共享资源的访问互斥操作。

2 消息接收线程

消息接收线程(ReadSerial 类)循环从串口读取数据并将其存放到消息缓冲区中，串口无数据或缓冲区满时它将处于资源等待状态。以下为线程的 run 方法代码：

```
public void run() {
    try {
        while (true) {
            int c = ComPort.read();
                                    //从串口读 1 B
            ComBuffer.putbyte(c);
                                    //将数据放入消息缓冲区
        }
    } catch (IOException e) { }
}
```

3 消息分析处理 Bean

消息分析处理 Bean(SerialBean 类)是 Swing 界面处理程序对串口进行操作访问的调用接口。其中封装有 3 个方法：Initialize 方法实现串口的初始化；ReadPort ()方法从消息缓冲区读消息并进行分析；WritePort (byte[] Msg)方法写消息到串口。串口初始化只执行 1 次，包括如下工作：

(1)打开串口

portId=CommPortIdentifier.getPortIdentifier(串口名)；

serialPort=(SerialPort)portId.open ("串口所有者名称",超时等待时间)；

(2)获取串口的输入/输出流

in=serialPort.getInputStream()；

out=serialPort.getOutputStream()；

(3)设置串口参数

serialPort.setSerialPortParams(9 600, SerialPort.DATABITS_8, SerialPort.STOPBITS_1, SerialPort.PARITY_NONE)；

(4)创建消息缓冲区对象

SB=new SerialBuffer()；

(5)创建并启动消息接受线程

RT=new ReadSerial(SB, in)；

RT.start()；

4 串口通信驱动程序的装载

串口驱动程序的装载是实现串口初始化的前提，如果程序在 Spring STS 开发环境中运行，会从 JDK 的运行环境装载驱动程序。这种情况下，必须事先将 comm.jar 拷贝到 JDK 安装目录的 jre\lib\ext 目录下，将 win32com.dll 拷贝到 JDK 安装目录的 jre\bin 目录下，将 javax.comm.properties 拷贝到 JDK 安装目录的 jre\lib 目录[2]下。

如果将开发的应用导出为独立的可运行的 JAR 文件，则必须将以上的 3 个文件安排到应用工程的 src 所在目录路径下，并在程序中用如下程序代码进行装载。

```
Import javax.comm.CommDriver;
import javax.swing.JOptionPane;
public class CurrentStatus {
public static SerialBean SB ;        //消息分析处理 Bean
public static void init(int n){          //n 为串口编号
    String driverName="com.sun.comm.Win32Driver";
    CommDriver driver=null;
    try {
        System.loadLibrary("win32com");    //装载 DLL
        driver=(CommDriver)Class.forName
        (driverName).newInstance();
    }
    catch(Exception e1){ e1.printStackTrace();}
    driver.initialize();                    //驱动程序初始化
    SB=new SerialBean(n);
    if (SB.Initialize()==-1)
        JOptionPane.showMessageDialog(null,"串口初始化错误");
}
}
```

这样，在程序中可通过执行 init 方法实现具体串口

的通信初始化。例如：

```
CurrentStatus.init(1);                    //初始化串口 1
```

5 在图形界面中实现通信调用

在电气设备的绝缘电阻检测应用中，图 2 为应用界面。当用户点击某个检测项对应的按钮时，通过注册按钮点击事件触发执行代码，将通过消息分析处理 Bean 的 WritePort 方法给串口发送检测命令，并利用循环等待读取来自单片机的检测结果。由于单片机在检测过程中将发送系列检测结果，因此，要循环读取数据，直到超过一定时间无数据可读或接收到结束标志的消息为止。具体工作过程如图 3 所示。其中，数据库的访问处理采用 Spring 的 JdbcTemplate 类提供的功能实现。

图 2 应用检测界面

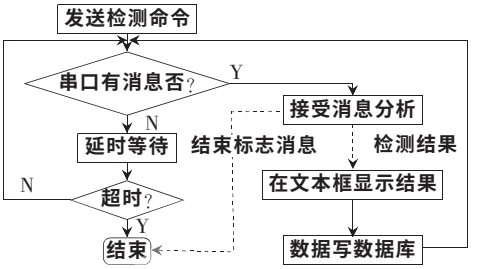

图 3 一次检测的消息接收过程

本文介绍了 Java 实现串口通信编程的典型编程处理要点，可有效地实现上位工控机与单片机之间的数据通信。系统通过多线程及对消息缓冲区资源的访问控制、延时等待控制等措施，保证了上下位机之间通信的可靠传递。实际应用中要根据具体的消息格式来组织通信过程中数据的分析处理。

参考文献

[1] 吴金锋，刘伟平，黄红斌.Java 串口通信数据采控系统的设计与实现[J].微计算机信息，2010(10).

[2] 唐未香.Java 程序与 ZigBee 串口通讯的实现[J].福建电脑，2010(5).

(收稿日期：2012-03-12)

作者简介：

丁振凡，男，1965 年生，教授，硕士生导师，主要研究方向：语义 Web，分布式计算，计算机辅助教学。