第22卷第6期
2019年6月

软件工程　SOFTWARE ENGINEERING

Vol.22 No.6
Jun. 2019

文章编号：2096-1472(2019)-06-22-04　　DOI:10.19644/j.cnki.issn2096-1472.2019.06.006

基于Websocket的串口控制系统的研究

张宏桥，蒋　虎，曾晓华，段凌飞
(湘南学院电子信息与电气工程学院，湖南 郴州 423000)

摘　要：随着计算机技术的不断发展，物联网设备的远控技术在实现上变得更加快捷。其中，串口是当前嵌入式数据传输技术的一种主流通讯方式，它有着协议简单，支持全双工传输等特点。本文针对常用的RS232串口，基于Websocket协议，使用支持跨平台的网络库libevent和串口库Serial Communication Library，研究并实现了Websocket与RS232之间的数据中转。接着基于Web前端HTML5标准的Websocket特性及相关UI框架，实现了Web与RS232之间数据的异地操作与传输，将串口数据实时绘制为网页图表并最终验证了该系统设计的有效性与可扩展性。

关键词：物联网；远程控制；串口通讯；Websocket
中图分类号：TP311　**文献标识码**：A

Research on the Serial Port Control System Based on Websocket

ZHANG Hongqiao,JIANG Hu,ZENG Xiaohua,DUAN Lingfei
(*College of Electronic Information and Electrical Engineering,Xiangnan University,Chenzhou* 423000,*China*)

Abstract: With the development of computer technology,it becomes more efficient to implement the remote control techniques on the devices of Internet of Things.The serial port is a main method of data transmission for embedded devices,with the characteristics of simple protocol and good support for full duplex transmission.Aiming at the serial port type of RS232,this paper studies the research and implementation of data transfer between Websocket and RS232 by using the cross-platform network library of libevent and the serial port library of Serial Communication Library.Then,based on the Websocket characteristics of HTML5 and Echarts Library of Web UI framework,it realizes the remote operation and data transmission between Web and RS232,and renders the serial port data on a web page graph in real time.Finally,the validity and extensibility of the proposed system are verified.

Keywords: Internet of Things;remote control;serial port communication;Websocket

1　引言(Introduction)

随着计算机技术的发展，物联网设备远程操控的实现变得更加快捷。串口是目前嵌入式数据传输的主流方式之一，有着协议简单，线路连接方便等特点[1]。同时，在人机交互领域，Web前端技术也愈发成熟与稳定，终端使用支持HTML5的浏览器访问服务器页面，即可实时控制节点并获取硬件数据[2]。目前主流的前端UI设计趋于模块化与代码的精简化，其本身又具有跨平台的优势，使得前端的多端发布更加容易[3]。本文利用H5标准的Websocket协议，使用C/C++设计了一种跨平台的网页与串口数据交互的中转服务端，可以在网页中对本地或者远程的多个硬件串口进行操作，同时允许用户可以便捷地对前端UI进行定制。

本文设计的数据操作框架如图1所示。基于Websocket协议与串口编程接口，将硬件数据上送至Web控制端，同时从Web控制端将相应的操作指令下发至数据服务端以完成具体操作。

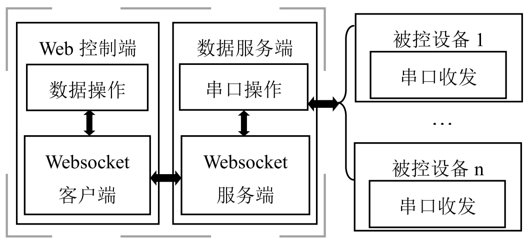

图1 系统结构图

Fig.1 System structure

基金项目：湘南学院2017年度校级实验室开放项目(编号31).

各模块解释如下：(1)被控设备：即下位机设备，通过RS232协议进行数据交互。(2)数据服务端：系统框架的核心，对下管理并操作多个串口数据，对上接收并执行Web控制端下发的控制指令，并在指令完成后向控制端返回相应的结果。(3)Web控制端：人机交互界面，使用Web框架实现，通过Websocket连接数据服务端并间接操作串口，允许控制端与服务端的异地分布。

2 系统相关技术(System technology)

2.1 串口操作

大部分的设备模块均支持一个以上的串口来进行通讯[4]。本文基于C/C++的编程环境，针对常用的RS232串口进行操作。在PC级别的操作系统上，串口操作一般基于自身的应用程序编程接口以文件的方式驱动串口。本文使用支持Windows、Linux，以及Mac操作系统环境的Serial Communication Library库进行串口操作，相关功能函数如下：

信息查询：vector<serial::PortInfo>comInfo=serial::list_ports()，其中comInfo为包含串口名称等字符串信息的vector向量。

设置与打开串口：serial::Serial(strComName,nBaud,tagTimeOut)，其中strComName为string类型的串口名称，nBaud为整数类型的波特率，tagTimeOut为serial::Timeout结构体类型的读写超时配置。该构造函数默认调用open成员函数，如果打开失败则会抛出异常信息。

关闭串口：Serial::close()，关闭当前实例对应的串口。

状态查询：Serial::isOpen()，返回当前实例对应串口的打开状态。

读取串口：Serial::read(pBuffer,nLen)，从当前串口读缓冲区中读取指定整数长度nLen的数据到字节指针pBuffer指向的内存中。

写入串口：Serial::write(pBuffer,nLen)，将字节指针pBuffer指向的内存数据，写到串口写缓冲区中，长度为指定的整数nLen。

2.2 Websocket数据传输

Websocket是W3C标准下的一种基于Tcp的应用层协议，利用该协议可以在Web层面快速稳定的进行数据流交互[5]。协议的基本操作包含安全握手，数据类型与数据流控制等内容，关键的通讯状态如下：

准备连接状态：服务端建立Tcp监听服务，等待客户端发起连接。

握手验证状态：Websocket基于HTTP协议进行握手验证，浏览器的网页地址格式为ws://服务主机地址:端口(默认80)，对于HTTPS(HTTP+SSL)的加密验证，地址格式为wss://服务主机地址:端口(默认443)。当客户端申请握手时，在协议头的“Sec-WebSocket-Key”字段里添加由Base64编码的握手文本，当服务端收到该握手信息后，提取该Base64秘钥，并在其之后加上一段特殊字符串"258EAFA5-E914-47DA-95CA-C5AB0DC85B11"，接着进行SHA-1与BASE64的操作，生成应答信息，最终赋给应答协议头中的“Sec-WebSocket-Accept”字段并返回给客户端校验，以确认接下来的数据交互是Websocket协议访问，而非HTTP。

数据传输状态：Websocket支持多种长度与消息类型的数据传输。常用的数据格式有文本、二进制、PINGPONG等格式，此外还可自定义数据类型。数据长度上支持短(小于126字节)、中(小于65536字节)、长(小于2^64字节)三种长度的数据传输。

由于底层的Tcp是面向流的传输，可以通过协议定义的控制字来解决Tcp传输数据的粘包、分包等问题，具体的帧头功能解释详见表1。

表1 Websocket协议帧头说明

Tab.1 Protocol header description of Websocket

帧头标识	长度	功能说明
数据终止	1位	判断本帧数据是否为一个完整数据包的最后帧
保留	3位	保留位
数据类型	4位	数据类型，范围0到15，服务端和客户端之间的消息类型可以不一样
是否加密	1位	当前数据帧是否使用循环异或加密
数据长度	7位	若小于126，则为当前帧的长度
扩展长度	4、8字节	若“数据长度”等于126，用4字节表示当前帧长度 若“数据长度”等于127，用8字节表示当前帧长度
加密掩码	0、4字节	若“是否加密”为0，则没有加密掩码 若“是否加密”为1，则用4字节存放加密掩码
传输数据	X字节	被传输的有效数据

本文使用支持跨平台的libevent网络库，基于上述的Websocket协议原理实现相关服务。实现时需注意如下问题：(1)长数据分帧方式的不同，即数据有可能在Websocket层被分为多帧发送；也可能在Websocket层使用一帧发送，却在Tcp层被分为了多帧。(2)服务端与客户端在数据加密上的区别，一般情况下，服务端向客户端发送非加密数据，客户端向服务端发送掩码异或加密的数据。

2.3 Web端数据传输与UI设计

目前主流的H5浏览器均支持Websocket相关的Javascript编程接口，相关的功能函数与事件如表2所示。

表2 浏览器Websocket相关函数与事件

Tab.2 Websocket functions and events of browser

函数或事件名称	功能	说明
WebSocket(Url)	连接函数	尝试连接Url指定的服务端地址，并返回操作对象
send	发送函数	向服务器发送数据
onopen	打开事件	连接建立时触发
onclose	关闭事件	连接关闭时触发
onerror	错误事件	连接/数据异常时触发
onmessage	接收事件	接收到数据时触发，接收的数据存放在event.data里

在目前H5的Web标准下，可以使用各类Javascript库高效地实现Web数据的渲染与交互。如JQuery库有着编码方便、性能高、插件丰富等特点；EasyUI库具有美观的界面插件，专注于用较少的代码开发UI。Echarts则是图表显示功能较为强大的Javascript功能库，可以方便地集成到其他模块当中[6]。本文基于Echarts库，将Websocket收到的串口数据实时绘制在浏览器图表上。

3 系统设计与实现(System design and implementation)

本文采用循环指令队列实现异步串口操作，支持通过一个控制端同时操作多个串口，控制流程如图2所示，其中“源ID”指的是下发当前指令所用的Websocket上下文环境。

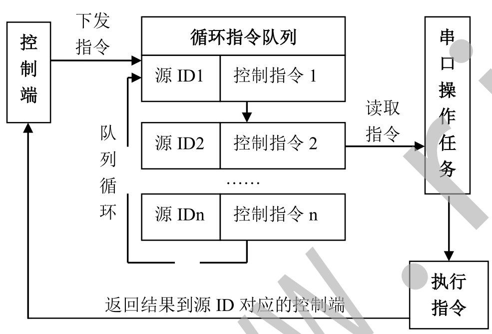

图2 循环指令队列

Fig.2 Queue of loop instructions

控制指令采用字符串格式，服务端接收指令并异步执行串口操作，最终返回给控制端相应的操作结果，具体命令与响应的定义如下文所述。

查询所有串口：当控制端下发查询命令后，数据服务端执行串口遍历相关功能，并返回串口列表。

表3 查询串口列表

Tab.3 List of serial ports

功能	格式	说明
控制命令	com list:	查询所有串口
服务响应	Com List:串口1|…|串口n	串口列表

打开串口：控制端下发打开串口命令，指定要打开串口的名称、波特率等信息，数据服务端先检查指定的串口是否存在，若存在则检查当前串口状态并尝试以指定的配置打开串口，最终异步响应执行结果。

表4 打开串口

Tab.4 Open serial port

功能	格式	说明
控制命令	com open:波特率,串口号	打开串口
服务响应	Com Open:串口号 not exist	串口不存在
	Com Open:串口号opened already	串口已经打开
	Com Open:串口号 opened failed	串口打开失败
	Com Open:串口号 opened ok	串口打开成功

关闭串口：控制端下发关闭串口命令，指定要关闭串口的名称，数据服务端先检查指定的串口是否存在，如果存在且串口已经打开，则尝试关闭串口，最终异步响应执行结果。

表5 关闭串口

Tab.5 Close serial port

功能	格式	说明
控制命令	com close:波特率,串口号	关闭串口
服务响应	Com Close:串口号 not exist	串口不存在
	Com Close:串口号 closed	串口关闭成功

查询串口状态：控制端下发串口查询命令，指定要查询串口的名称，数据服务端先检查指定的串口是否存在，如果存在则检查串口的开关状态，最终异步响应查询结果。

表6 串口状态查询

Tab.6 Check serial port state

功能	格式	说明
控制命令	com state:串口号	状态查询
服务响应	Com State:串口号 not exist	串口不存在
	Com State:串口号 opened	串口已经打开
	Com State:串口号 closed	串口已经关闭

写入串口数据：控制端下发写串口命令，指定所写的数据类型、串口号，以及要写入的数据。数据服务端先检查指定的串口是否存在，如果存在且打开则执行写入操作，最终异步响应写入结果。其中，数据类型为0或1，0表示文本型数据，1表示用Base64编码的二进制数据。

表7 写入串口

Tab.7 Write serial port

功能	格式	说明
控制命令	com write:类型,串口号:写入数据	数据写入
服务响应	Com Write:串口号 not exist	串口不存在
	Com Write:串口号 closed	串口关闭
	Com Write:串口号 data error	数据解析异常
	Com Write:串口号 ok	串口写成功
	Com Write:串口号 failed	串口写失败

读取串口数据：控制端下发读串口命令，指定要读取的长度、串口号。数据服务端先检查指定的串口是否存在，如果存在且打开则执行读取操作，最终异步响应读取结果。其中，实际读取的数据长度小于等于指定的读取长度，数据以Base64编码的形式返回。

表8 读取串口

Tab.8 Read serial port

功能	格式	说明
控制命令	com read:预读取长度,串口号	数据读取
服务响应	Com Read:串口号 not exist	串口不存在
	Com Read:串口号 closed	串口关闭
	Com Read:串口号:Base64数据	串口读到数据
	Com Read:串口号 failed	串口读失败

4 系统功能验证(System function verification)

本文使用被控设备向PC串口发送随时间变化的正弦信号，Web控制端以20ms的间隔下发读命令给数据服务端，服务端接收串口数据并上送给控制端，最终使用Echarts把数据以图表的形式渲染出来，串口的参数以网页URL的形式传入。本文使用同一局域网内的两台PC机进行测试，其中一台运行数据服务器并与下位机用串口连接，下位机不断向数据服务端发送正弦数值，另外一台在浏览器中运行Web控制端网页，获取串口数据并显示，工作流程如图3所示。

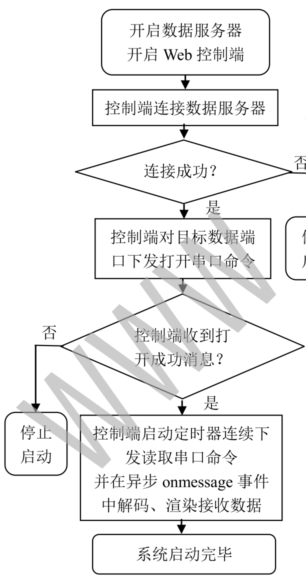

图3 系统工作流程

Fig.3 System working flowchart

图4显示了数据中转服务器启动后绑定了通用IP地址0.0.0.0，以及指定的端口8881，并打印出本地串口列表。图5显示了在同一网络环境的另外一台电脑上用浏览器直接打开本地网页文件，即数据控制端，其启动后尝试连接数据服务器，并以9600的配置打开名称为“COM6”的串口，最终将硬件数据实时渲染为可视化的图形。

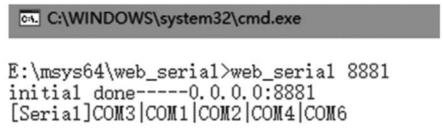

图4 数据服务端初始化

Fig.4 Data server initialization

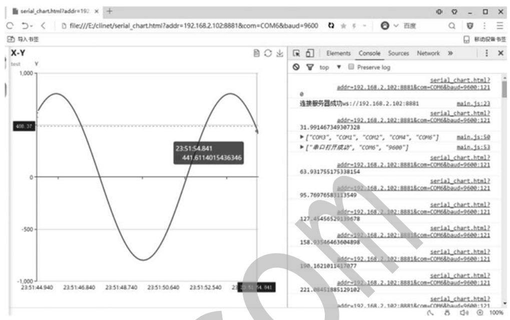

图5 Web控制端数据绘制

Fig.5 Data rendering on Web controller

5 结论(Conclusion)

本文使用支持跨平台的RS232串口库Serial Communication Library，以及网络编程库libevent，基于Websocket协议实现了网页与串口之间的数据传输。然后利用了Web网页的Websocket特性，以及Echarts网页前端UI库，实现了将串口波形数据渲染到网页图表上的WEB界面。最终验证了该系统的有效性与可扩展性，为嵌入式上位机的实现提供了一种新的思路。

参考文献(References)

[1] 张胜,李瑞民.物理隔断计算机网络间的实时通信设计[J].电视技术,2016,40(6):127-130.

[2] 谢佳柏,陈贤祥,胡欣宇,等.基于低功耗蓝牙和WebSocket的物联网数据网关[J].仪表技术与传感器,2016(1):76-78.

[3] 章锐,陈树勇,刘道伟,等.基于ECharts的电网Web可视化研究及应用[J].电测与仪表,2017,54(19):59-66.

[4] 陈旭辉,杨红云.基于ＳＴＭ３２的多串口并行传输系统设计[J].计算机测量与控制,2019,27(1):166-170.

[5] 钱宇虹.使用Websocket和Servlet实现服务器定点推送[J].软件工程,2016,19(10):30-33.

[6] 冀潇,李杨.采用ECharts可视化技术实现的数据体系监控系统[J].计算机系统应用,2017,26(6):72-76.

作者简介：

张宏桥(1989-)，男，硕士，助理实验师.研究领域：软件开发，计算机控制.

蒋　虎(1998-)，男，学士生.研究领域：软件开发，嵌入式开发.

曾晓华(1976-)，女，硕士，高级实验师.研究领域：系统仿真，嵌入式开发.

段凌飞(1983-)，男，硕士，讲师.研究领域：嵌入式开发，计算机控制.