

串口使用与测量第一次报告

一.实验目的

- 1.熟练使用 Linux 下 io 函数 read、write 和 epoll 等
- 2.熟练 RS485 串口的信号特点
- 3.熟练处理流式通信数据
- 4.理解 485 总线的冲突问题

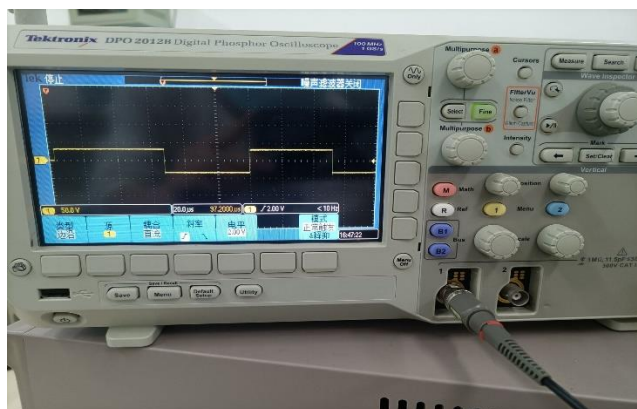
二.实验过程

RS485 信号测量

本实验需要将 A 板与 B 板通过 RS485 接口连接并进行通信。需要使用杜邦线连接两单片机板的 RS485 接口，通过计算机向 B 板写入发送的数据（A 板序列号+学号），数据将通过串口发给 B 板，再经 RS485 接口发给 A 板，发送后 A 板将发回密码，B 板将密码通过串口发送到计算机，需要在计算机上读出密码。

1.测量 A 板波特率

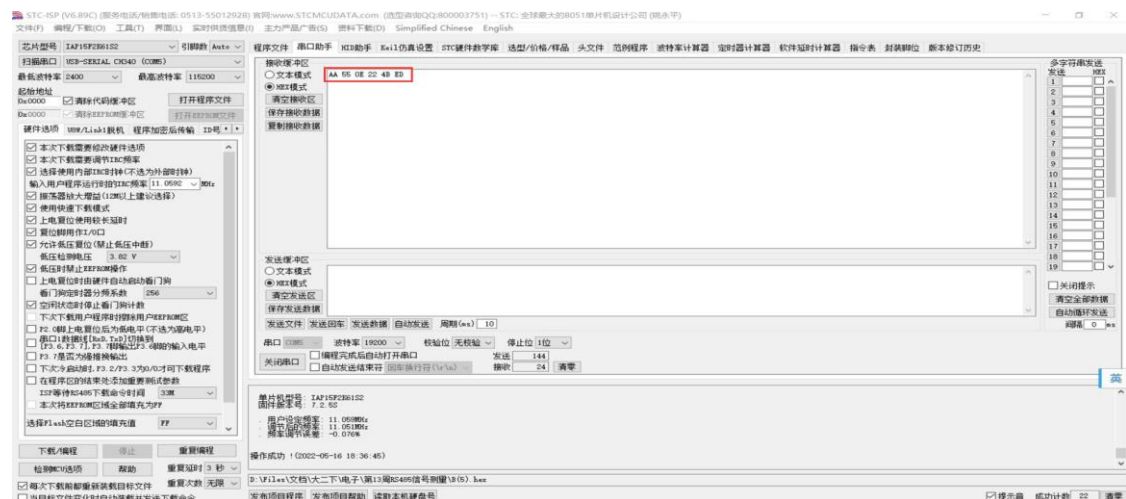
将 A 板 RS485 接口接到示波器，测量如下：



波特率大约为 $1/50\mu s$ ，故可确认波特率为 19200。将 B 板接示波器，控制摇杆调整至波特率为 19200。

2.测量序列号

将 A 板与 B 板的 RS485 接口通过杜邦线连接，按下 A 板 K3，在串口调试助手可以收到 A 板的序列号为：0e224bed。



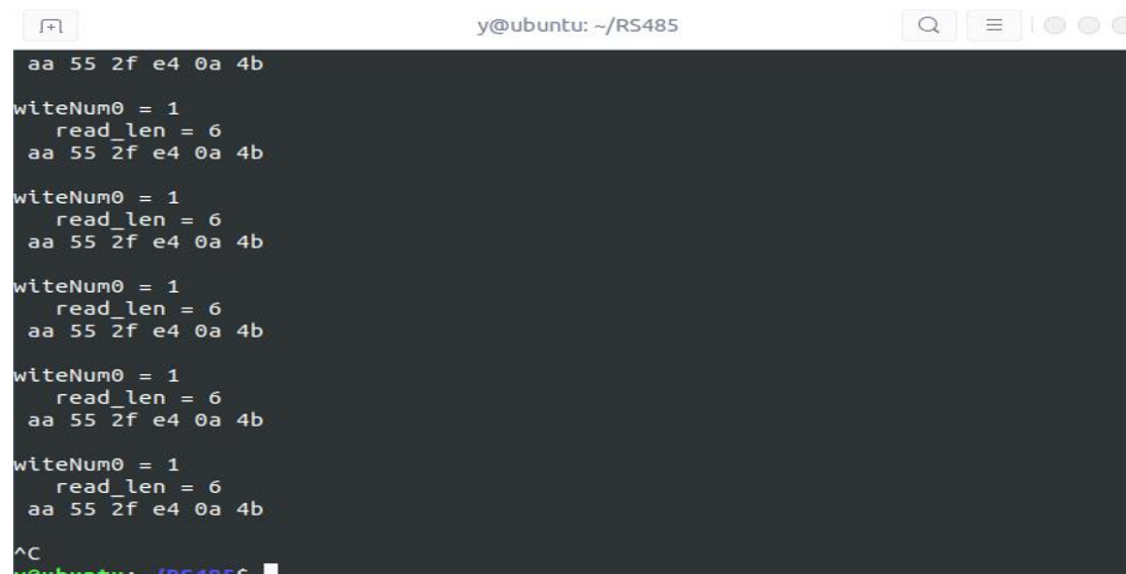
此处由于时间有限，直接使用串口调试助手得到 A 板的序列号，可用示波器得到 A 板的序列号，使用示波器按波形读取序列号时，需要注意停止位。

3.读密码

在 linux 平台下的前次实验的读取串口程序中，设置缓冲区，循环中向串口写入 AA55+序列号+学号



运行程序，可以收到 A 板发送的密码：



密码为 2fe40a4b。

4.提交

将序列号及密码提交。

```
命令提示符
Microsoft Windows [版本 10.0.19042.1586]
(c) Microsoft Corporation。保留所有权利。

C:\Users\18332>curl "132.232.98.70:6363/check485?id=202004061409&v=0E224BED&s=2FE40A4B"
OK
C:\Users\18332>
```

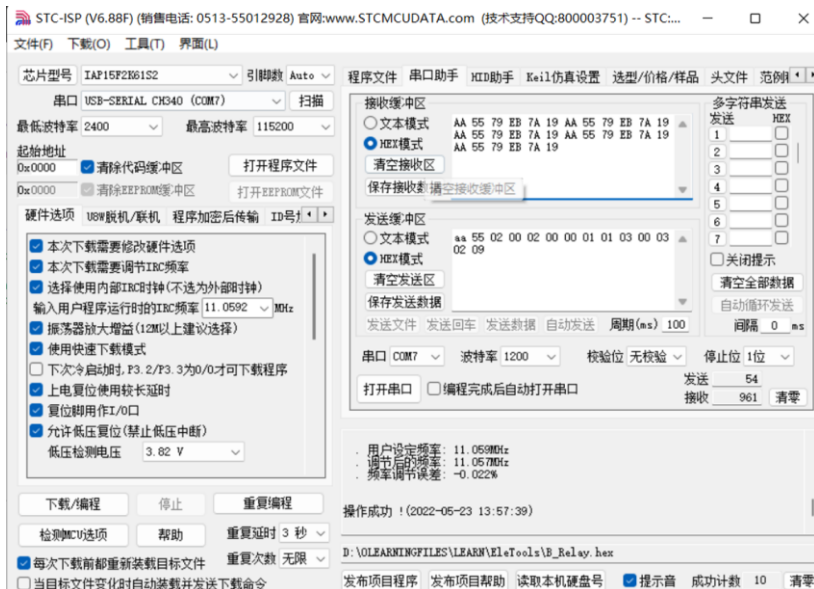
RS485 总线数据收发

RS485 总线有两条线信号线，能够传输一个逻辑信号。计算机标准的 UART 串口有 RX、TX 收发两条线，因此能够同时进行数据的接收和发送。而 RS485 只有一个逻辑信号，因此同一时刻只能有一个主体进行数据发送（因此叫做半双工通信串口）。

本实验需要连接两单片机板的 RS485 接口，A 板将先发送序列号。使用计算机连接 B 板串口，发送学号，然后 B 板将发送给 A 板，此后 A 板将通过 RS485 接口发送密码，B 板需要接收密码，取出密码发回，然后 A 板将发送新的密码。重复该过程至 A 板不再发送新的密码。

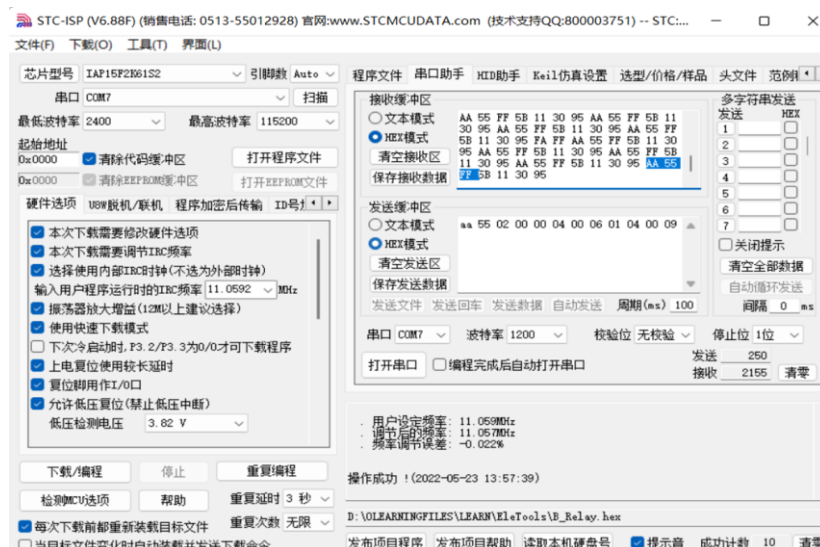
1.测量 A 板序列号

先使用串口调试助手接收 A 板的序列号，序列号为 79eb7a19



2.接收密码，分析密码格式

先使用串口调试助手发送学号，接收一次 A 板的密码，从而判断 A 板发送密码的格式。得到的密码格式为 aa 55 ff 5b 11 30 95，即 aa 55 后开始四个字节即为密码，中间只有一个字节的填充字符。



已知密码格式，接下来编写程序完成 B 板的学号发送，密码的处理及密码的发送工作。

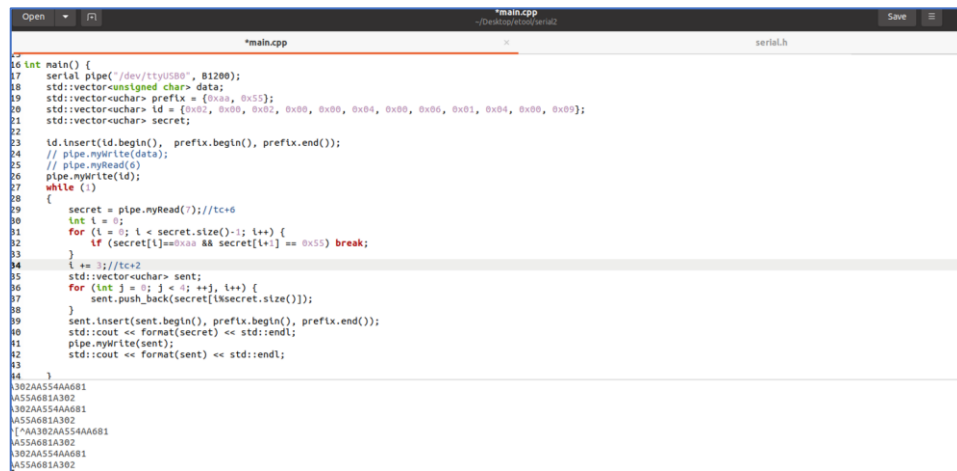
C++语言实现

```
int main() {
    serial pipe("/dev/ttyUSB0", B1200);
    std::vector<unsigned char> data;
    std::vector<uchar> prefix = {0xaa, 0x55};
    std::vector<uchar> id = {0x02, 0x00, 0x02, 0x00, 0x00, 0x00, 0x04, 0x00, 0x06, 0x01, 0x04, 0x00, 0x09};
    std::vector<uchar> secret;

    id.insert(id.begin(), prefix.begin(), prefix.end());

    pipe.myWrite(id);
    while (1)
    {
        secret = pipe.myRead(7);
        int i = 0;
        for (i = 0; i < secret.size()-1; i++) {
            if (secret[i]==0xaa && secret[i+1] == 0x55) break;
        }
        i += 3;
        std::vector<uchar> sent;
        for (int j = 0; j < 4; ++j, i++) {
            sent.push_back(secret[i+secret.size()]);
        }
        sent.insert(sent.begin(), prefix.begin(), prefix.end());
        std::cout << format(secret) << std::endl;
        pipe.myWrite(sent);
        std::cout << format(sent) << std::endl;
    }
}
```

运行结果：



最终得到的密码为 a681a302，提交：



C 语言实现

主要部分如下：

```
EpollInit(fd); //初始化终端事件触发函数epoll,设置要监听的事件及相关参数等
unsigned char buf[]={0xAA,0x55,0x02, 0x00, 0x02, 0x00, 0x00, 0x04, 0x00, 0x06, 0x01, 0x04, 0x00, 0x09};
if(write(fd,buf,14)<=0) printf("write failed!\n");
int count=0;
int start=4; //开始的字节
unsigned char temp;
while(1){
    rl = ComRead(tmp,1); //读取数据
    printf(" %02x", tmp[0]);
    //识别前导AA 55
    if(count>=2){
        count++;
        if(count>=start) buf[count-start+2]=tmp[0];
        if(count-start==3) { //读到4个字节
            write(fd,buf,0);
            count=0; //清0
        }
        if(tmp[0]==0x55&&temp==0xAA) //找到aa55, count开始计数
            count=2;
        temp=tmp[0]; //上一个字节
    }
    close(epid);
    close(fd);
    return 0;
}
```

读取到 aa55 后开始计数，start 设置为 4，即 count=4，第四个字节而开始向缓冲区 buf 写入密码字节，写够 4 字节后向串口写入数据。由于这种实现需要一个字节一个字节读取数据，耗时很长，最后没有使用该程序，但测试时运行三分钟左右，得到的密码提交为 38 号密码，表明程序可以正确运行。

三.实验总结

通过实验进一步熟悉了串口通信的方法，以及使用 write，read 等函数进行串口数据进行读写。了解了 RS485 接口的使用，通过 RS485 实现了两板的通信和数据传输。认识了 RS485 的半双工通信模式，在程序运行时，有些产生的密码是错误的，即发生了冲突得到了错误数据。对于程序的编写，能够通过 c 语言

程序实现功能,但是效果不好,还需要进一步学习。最终的实现还是参考他人的c++代码完成,后续还需要加深理解,提高代码的性能以及编写相关代码的能力。