

**课 程 实 验 报 告**

**课程名称： 射频识别技术原理及应用**

**专业班级： 物联网工程1601**

**学 号： U201614890**

**姓 名： 徐光磊**

**指导教师： 甘早斌**

**报告日期：**

**计算机科学与技术学院**

**目 录**

[**1 实验一 低频读写器实验** 4](#_Toc5870208)

[**1.1 实验目的** 4](#_Toc5870209)

[**1.2 实验内容及结果** 4](#_Toc5870210)

[**1.3 实验体会与总结** 7](#_Toc5870214)

[**1.4 核心源码说明** 7](#_Toc5870215)

[**2 实验二 高频读写器实验ISO14443A** 10](#_Toc5870216)

[**2.1 实验目的** 10](#_Toc5870217)

[**2.2 实验内容及结果** 10](#_Toc5870218)

[**2.3 实验体会与总结** 22](#_Toc5870224)

[**2.4 核心源码说明** 22](#_Toc5870225)

[**3 实验三 高频读写器实验ISO15693** 24](#_Toc5870226)

[**3.1 实验目的** 24](#_Toc5870227)

[**3.2 实验内容及结果** 24](#_Toc5870228)

[**3.3 实验体会与总结** 33](#_Toc5870237)

[**3.4 核心源码说明** 33](#_Toc5870238)

[**4 实验四 超高频读写器实验** 34](#_Toc5870239)

[**4.1 实验目的** 34](#_Toc5870240)

[**4.2 实验内容及结果** 34](#_Toc5870241)

[**4.3 实验体会与总结** 39](#_Toc5870248)

[**4.4 开发实例源码** 39](#_Toc5870249)

[**5 实验五 RFID综合应用实验** 40](#_Toc5870250)

[**5.1 需求分析** 40](#_Toc5870251)

[**5.2 系统详细设计** 40](#_Toc5870252)

[**5.3 系统实现与系统测试** 40](#_Toc5870253)

[**5.4 总结** 40](#_Toc5870254)

[**5.5 系统源代码** 40](#_Toc5870255)

**1 实验一 低频读写器实验**

**1.1 实验目的**

通过本次实验了解博创科技 RFID 读写器的结构组成，熟悉各个模块的功能，掌握 试验箱的连接和操作方法。掌握串口命令参数的意义和设置方式。

了解低频读写器的基本原理，学会如何使用实训软件对低频读写器进行读卡操作（验证性实验）。

学习和掌握在低频读写器的编程操作，对标签进行读操作，了解低频读写器的工作 机理，并完成一个示例程序。

**1.2 实验内容及结果**

1.2.1 完成低频读写器的标签读取试验

在实验一开始时，先需要把实验箱中的串口连接线、电源连接线完成连接，其中串口连接线直接连接在电脑的主板上即可。将本次开发所需要的125K模块的电源按钮打开，同时按选择按钮选择到125K模块，即125K模块旁的绿灯亮起。



图1.1

如图1.1所示，在完成开发箱的链接后，打开RFID教学实训系统中，在连接操作中配置上正确的125K读写，连接端口为COM1，频率为9600，点击连接按钮即可球球连接。若右侧输出“初始化COM1成功”，则说明成功连接上了读卡器设备。

将125K卡靠近进行刷卡，可以在主界面的通讯记录中看到了刷卡的记录，带有时间、数据报文（卡ID）等信息。

1.2.2 熟悉低频 LF-125K 开发实例，完成低频读写器编程实验，熟悉和了解低频读写器API 函数；分析应用 demo 存在的问题，进而完善应用 demo 的功能。完善的功能包括： 记录保存进出的历史记录、停留时间。

在Qt中利用QtRecord等相关组件和SQLite数据库进行通信，利用QtSerialPort进行和串口的通信过程。最终实现了一个比较完善的门禁系统，可以记录用户的历史进出信息和每次出入门禁的停留时间。

在代码的编写上，主要的界面基于Qt的图形化编程界面，可以快速构建出QTable、按钮、选择项等元件，连接上槽函数后，即可对程序进行回调响应，调用指定的函数。在数据库方面，主要通过调用QtRecord来对SQLite发起SQL命令请求，对数据库进行增删改查操作。

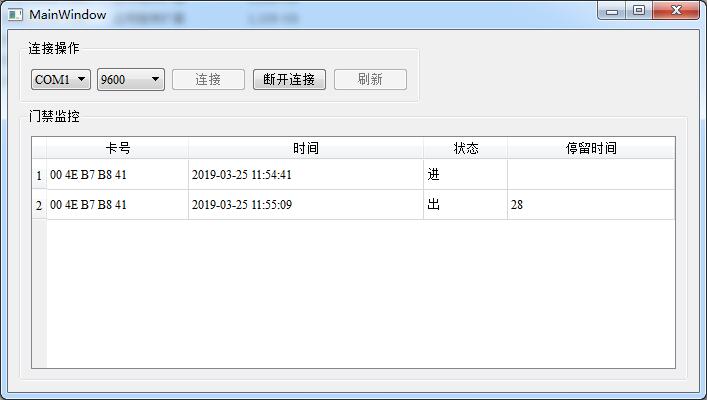


图1.2

如图1.2所示，在连接成功后，连续重复刷同一张卡，则可以看到对应的进系统和出系统信息，同时记录下了当下的时间，出门禁的记录表项中还记录下了停留时间。

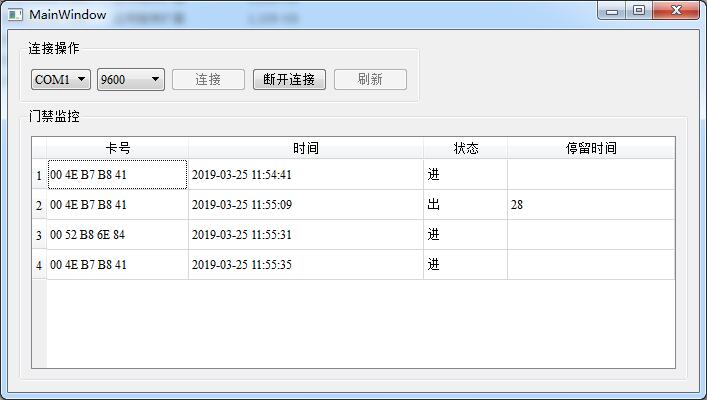


图1.3

如图1.3所示，进行多张卡的混合刷卡情况下，程序能够进行正常的连续记录。

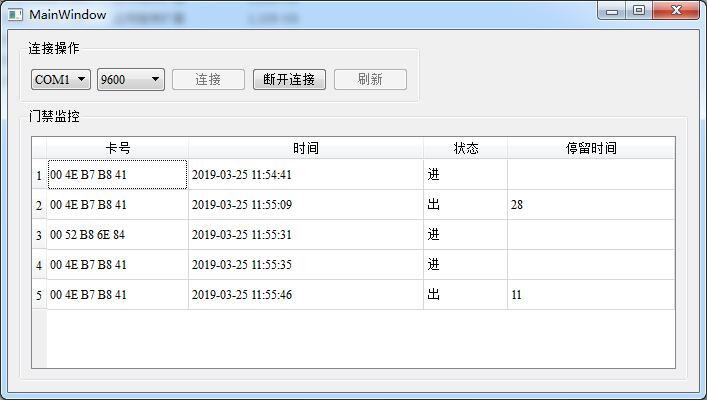


图1.4

如图1.4所示，在多张卡的混合刷卡情况下，离开程序后，重新进入，仍然能够查看到SQLite数据库中存储的门禁进出信息。

1.2.3 思考题

1、通过试验箱，反复循环读取十张低频电子标签。在读取过程中可能会遇到哪些问 题或发生哪些现象，并分析遇到的这些问题或现象的原因；

答：在这个过程中，发现的问题有，某一次中读卡器没有成功识别出低频电子标签，需要重新刷卡才能正确读取。由于这个问题不好复现，

2、在利用低频读写器模拟门禁系统中，如何获取读写器发送过来的卡号？请写出相应的函数体（含注释），并说明函数的调用方法。

答：通过调用QSerialPort构造函数新建一个serialPort变量，通过这个变量，来调用setPortName，setBaudRate等函数进行初始化，检测到数据通过的时候，就可以调用serialPort->readAll()来返回在串口中读取得到的数据。

相关代码如下：

//新建一个QSerialPort对象，构造函数中传入this，返回值为一个QSerialPort对象

this->serialPort = new QSerialPort(this);

//传参为端口名，可以对QSerialPort进行初始化

serialPort->setPortName(name);

//传参为波特率，可以对QSerialPort完成初始化

serialPort->setBaudRate(baud.toInt(),QSerialPort::AllDirections);

//没有参数，返回值为串口中待传输的数据长度

if(serialPort->bytesAvailable() < 5)

return;

//没有参数，返回值串口中所有待传输的数据，以QByteArray的形式返回

QByteArray data = serialPort->readAll();

**1.3 实验体会与总结**

通过本次实验我学会了通过试验箱对低频标签的读取过程，能够利用Qt基于串口通信进行基础的模拟GUI软件进行开发，能够调用低频读写器的基础API函数和Qt数据库操作相关的函数指令。

**1.4 核心源码说明**

Mainwindow.cpp

if(m125dll->LF125K\_FrameAnalysis((uint8 \*)(data.data())) == 0)

{

QString tagId = CharStringtoHexString(tr(" "),data.data(),data.length());

QString time = CurrentDateTime();

int index = model->findRecord(tagId);

if(index >= 0 )

{

QString text = model->record(index).value(2).toString();

if(text == tr("进")){

QString oldtime = model->record(index).value(1).toString();

qDebug()<<oldtime;

QDateTime start = QDateTime::fromString(oldtime, "yyyy-MM-dd hh:mm:ss");

QDateTime end = QDateTime::fromString(time, "yyyy-MM-dd hh:mm:ss");

uint stime = start.toTime\_t();

uint etime = end.toTime\_t();

QString period = QString::number(etime-stime);

model->addRecordWithPeriod(tagId,time,tr("出"),period);

// model->addRecord(tagId,time,tr("出"));

}

else

model->addRecord(tagId,time,tr("进"));

}

else {

model->addRecord(tagId,time,tr("进"));

}

}

在监听到刷卡信息后，获取当前的刷卡的卡号和当前的时间，从数据库中查询最近的同卡号的相关信息。若最近一条信息是出或无信息，则在数据库中添加一条入的信息。若最近一条是入，则在数据库中添加一条出的信息，同时在信息中加入停留信息的表项。

Recordtablemodel.cpp

int RecordTableModel::addRecordWithPeriod(QString &tagId, QString &time, QString status, QString period)

{

QSqlRecord record;

record.append(QSqlField(header.at(0), QVariant::String));

record.append(QSqlField(header.at(1), QVariant::String));

record.append(QSqlField(header.at(2), QVariant::String));

record.append(QSqlField(header.at(3), QVariant::String));

record.setValue(0, QVariant(tagId));

record.setValue(1, QVariant(time));

record.setValue(2, QVariant(status));

record.setValue(3, QVariant(period));

insertRecord(-1, record);

return rowCount();

}

在这个文件中主要负责和数据相关的读建表操作，举一个实现完全的函数来作为例子。通过调用QSqlRecord来执行一条SQL插入表项指令，指令中的数值有tag id，时间，出入状态，停留时间。

**2 实验二 高频读写器实验ISO14443A**

**2.1 实验目的**

通过本次实验了解高频读写器的基本原理，学会如何使用高频读写器，掌握 串口命令参数的意义和设置方式。

阅读和了解 ISO14443A 协议的主要内容，进一步加深对 S50 卡的存储结构和 ISO14443A 协议的理解，掌握 ISO14443A 协议的常用命令的含义和用法。

通过高频读写器的实验，掌握对 S50 卡各个扇区数据的读写方法，并熟悉高 频读写器（ISO14443A）API 函数。

**2.2 实验内容及结果**

2.2.1 完成 ISO14443A 协议下标签寻卡、唤醒、休眠实验



图2.1

如图所示，在连接了传感器开发板和调试PC机后，在RFID读写Demo中设置为连接COM1端口，在19200波特率下进行通信，连接后尝试进行搜寻RFID 13.56M卡片。

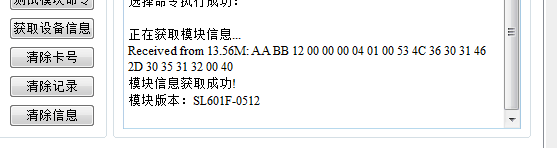


图2.2

如图所示，可以看到在交互信息中提示了正确执行了相关命令，模块信息获取成功，说明连接成功。

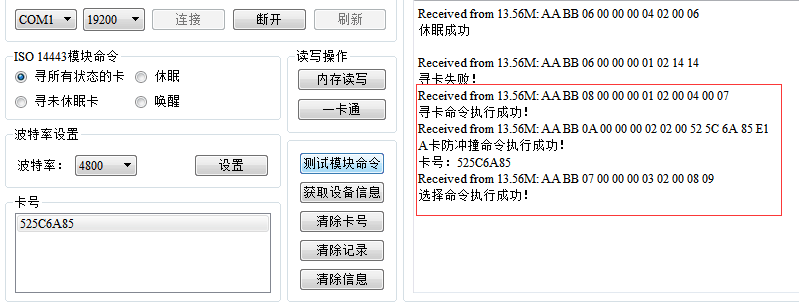


图2.3

如图所示，在ISO 14443命令面板中执行了“寻所有状态的卡”，在交互信息中输出了寻卡命令、A卡防冲突、选择命令执行成功，说明能够正确完成寻找卡片的过程。

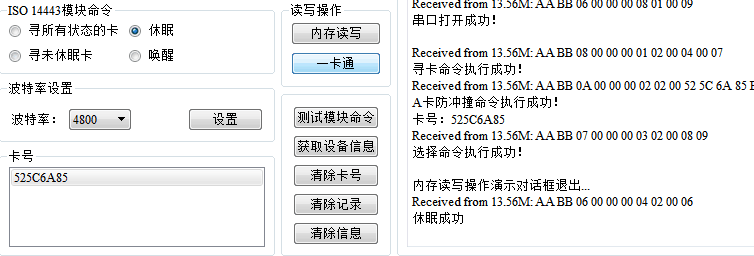


图2.4

如图所示，对正在测试的卡片执行了休眠命令，然后再执行寻未休眠卡命令。可以看到交互信息中输出的是，休眠成功。说明对于卡片的休眠命令生效了，预期的效果为使得卡片不能够被通过一般的方法进行寻找。

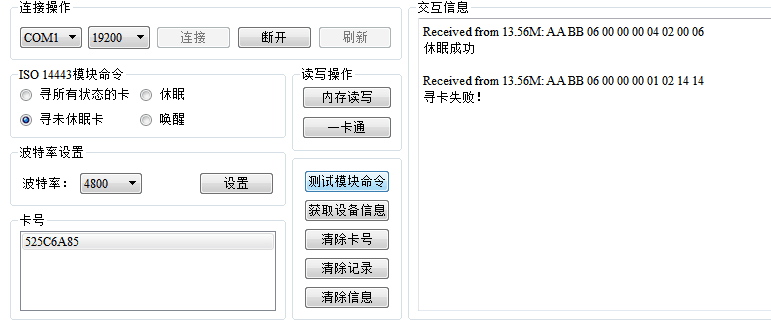


图2.5

如图所示，之前对正在测试的卡片执行了休眠命令，现在再执行寻未休眠卡命令。可以看到交互信息中输出的是，寻卡失败。说明对于卡片的休眠命令生效了，使得卡片不能够被通过一般的方法进行寻找。

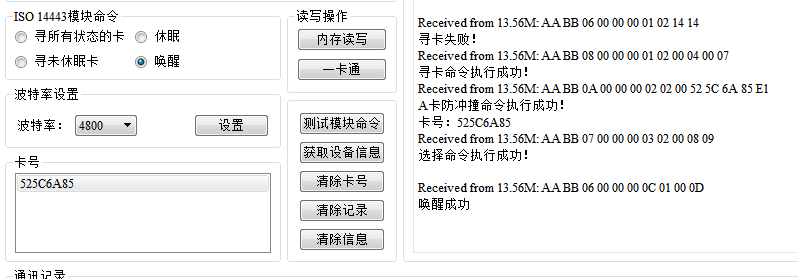


图2.6

如图所示，对指定的卡片执行唤醒命令，可以看到在交互信息面板中输出了唤醒成功的信息。在唤醒成功之后，则可以通过普通的寻卡命令进行搜索此卡。

2.2.2 完成 ISO14443A 协议下标签内存读写实验

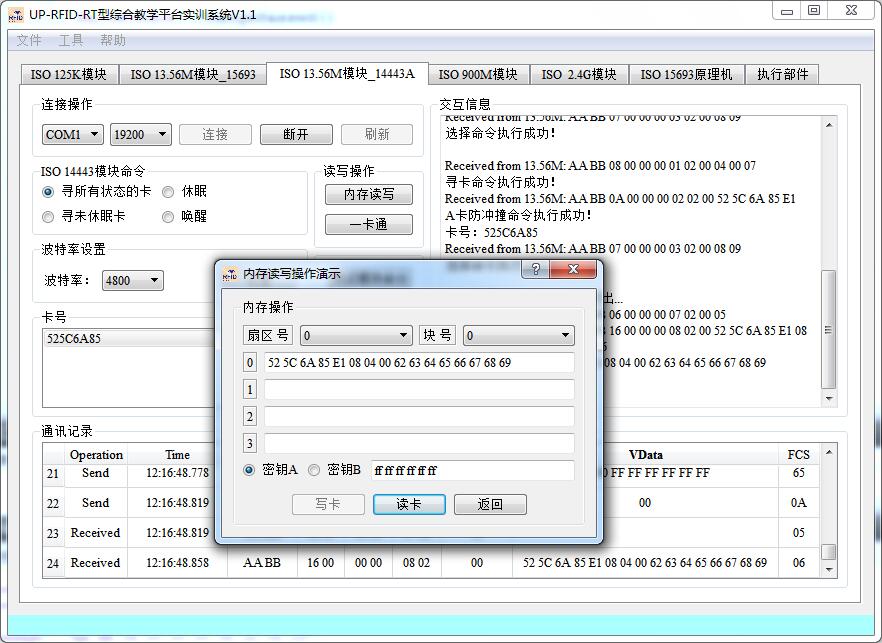


图2.7

如图所示，对于指定的卡片进行内存读写操作，通过RFID Demo中的内存读写管理，尝试读取0号扇区0号块中的卡片序列号数据，可以看到正确返回了读取得到的数据。

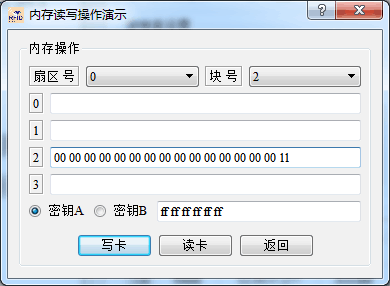


图2.8

如图所示，因为0号扇区0号数据块无法写入，更换块号尝试对卡片进行写入（原始数据为全0）。

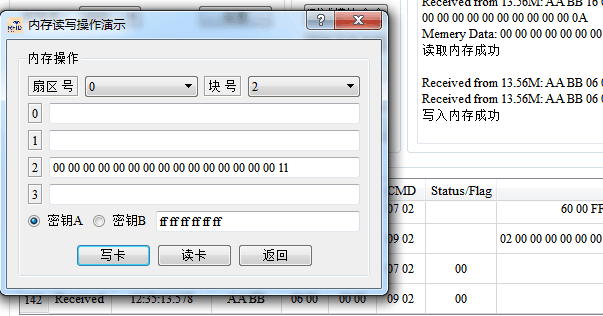


图2.9

如图所示，在完成写入之后，看到了在交互信息模块中，输出了写入内存成功的消息。在之后的读取中，可以看到对于内存的写入内容被读取命令输出出来了。

2.2.3 完成 ISO14443A 协议下标签一卡通实验

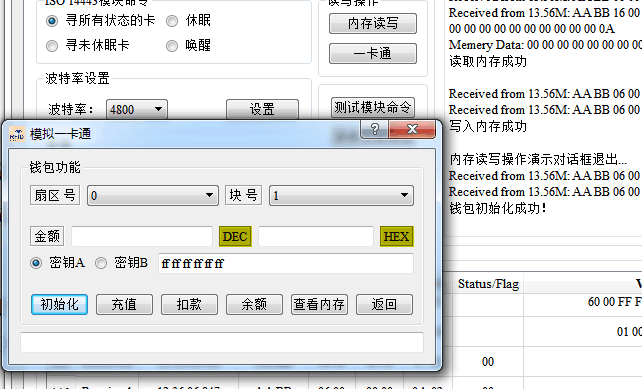


图2.10

如图所示，执行了RFID Demo中的模拟一卡通测试程序。可以对卡片进行初始化、充值、扣款、查看余额、查看内存等功能。图中的交互信息中输出了钱包初始化成功的消息。

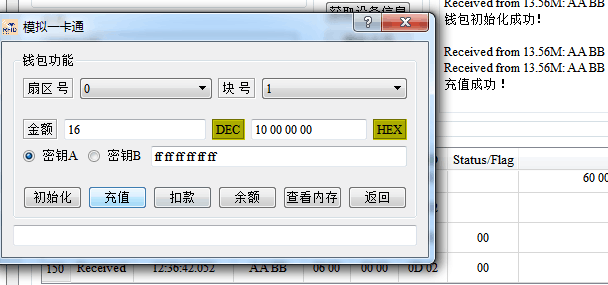


图2.11

如图所示，输入金额16元，再点击了充值按钮，在交互信息面板中，成功输出了充值成功的消息。

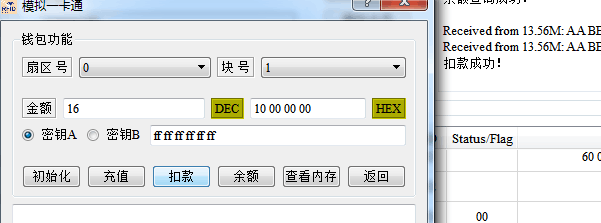


图2.12

如图所示，在充值了之后，进行16元的扣款操作，可以看到在交互信息面板中成功输出了扣款成功的信息。在之后的查看余额命令中，输出的金额为0，说明充值、扣款、查看余额的功能都是相同的。

2.2.4 熟悉和了解高频 HF1356M 14443A 开发实例，掌握高频读写器（14443A） API 函数，并通过编程实现对 S50 卡的指定内存进行数据读写、充值、减值



图2.13

如图所示，在开发实例的一卡通管理系统中，进入的第一个界面为输入管理员的账户密码来进入管理界面。

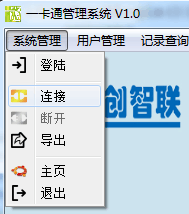


图2.14

如图所示，在左上角的控制栏中有一个系统管理，在每次登陆之后，需要点击连接来完成软件和传感器串口通讯的过程。

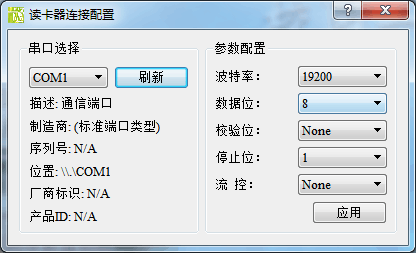


图2.15

如图所示，为读卡器的连接配置，默认配置为COM1端口通信，通过19200波特率，8位数据位的标准进行通信。这部分的数据将用来初始化编程中的串口通信模块。

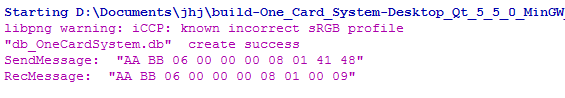


图2.16

如图所示，在完成读卡器的初始化命令后，程序输出了发送的信息和接受的信息，说明双方成功进行了通信过程。

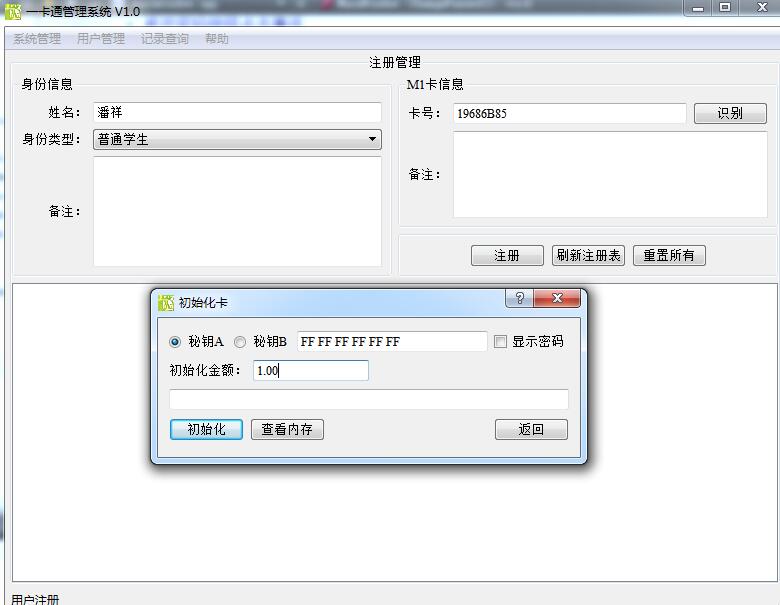


图2.17

如图所示，在管理的界面中，先添加一个新卡新用户，在初始化界面中，可以同时进行初始化和余额赋值。注意在秘钥的选择中，可以直接选择秘钥A，FFFFFFFFFFFF。

C:\Users\28529\OneDrive\RFID实验\第二次实验\10.png

如图所示，在程序的输出界面中，看到了Qt程序执行了插入数据的SQL语句。可见新建用户成功。

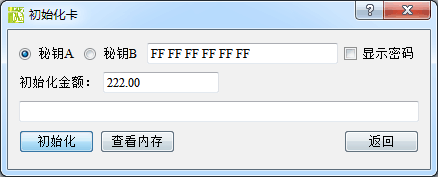
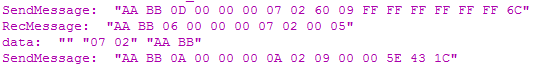


图2.18

如图所示，可以通过初始化卡来为卡初始化一个金额。



如图所示，在完成了初始化之后，可以看到程序在通讯过程中，成功发送和接受的信息、对应的数据报文。

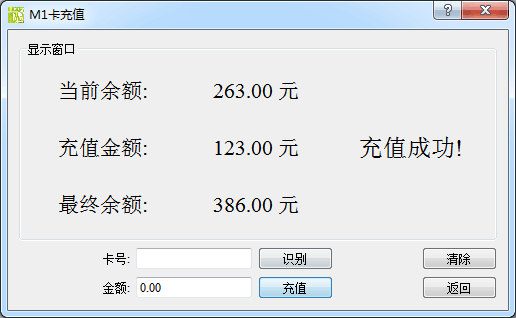


图2.19

如图所示，在充值的界面中，程序可以正常的对M1卡进行任意金额的充值，并最后输出一个求和的最终余额。

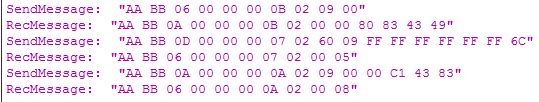


图2.20

如图所示，在充值之后，程序进行了正确的输出，可以看到在通信过程的发送和接受报文。



图2.21

如图所示，在记录查询的选项卡中，可以进行各种记录的查询，本质上就是在对SQLite数据库进行查询操作。



图2.22

如图所示，在消费模拟界面中，可以对卡中的余额进行写操作，扣去费用。

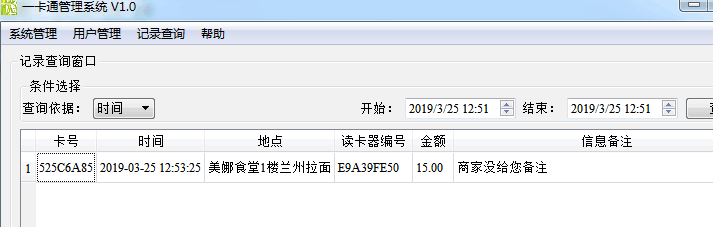


图2.23

如图所示，在记录查询界面中，可以正确查询到前面的消费记录，说明消费的过程已经被成功写入SQLite数据库中。

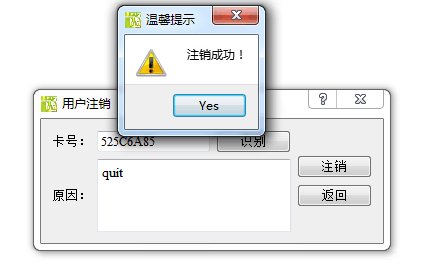


图2.24

如图所示，可以对已经注册的老用户老卡号，进行注销操作，在注销之后预计会在注销记录中进行记录。

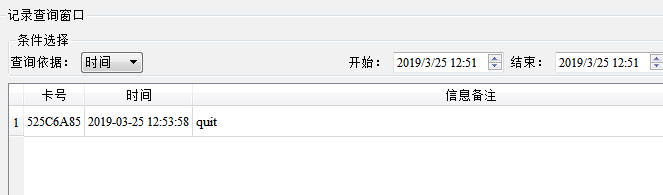


图2.25

如图所示，在记录查询界面中，成功查询到了注销的记录，则说明注销的程序逻辑运行正确。

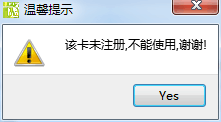


图2.26

如图所示，在卡被注销之后，尝试进行充值，程序进行了提示，“该卡未注册，不能使用”，说明注销的信息数据被成功地写入到了卡片内部。

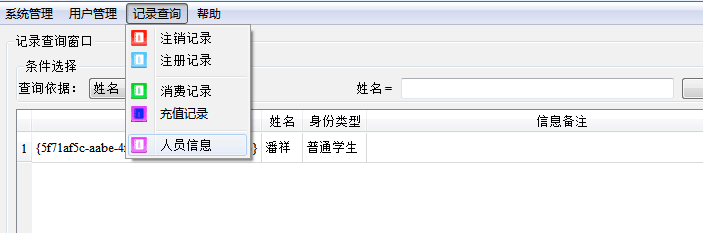


图2.27

如图所示，在记录查询中的人员查询中，可以查询到在记录中的所有用户，包括其卡号、姓名、身份类型。

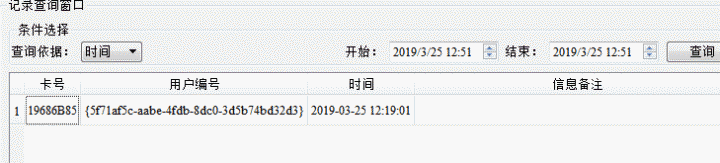


图2.28

如图所示，在记录查询中的新建用户记录中，可以看到人员记录查询中的那名同学的记录时间。

2.2.5 思考题

1、S50 卡共有 16 个扇区，请问第 4 扇区的绝对块地址号是多少？请详细说 明计算的方法和依据。

答：16个扇区的64个块按绝对地址编号为0~63，所以第4扇区的绝对块地址号为64/16\*3=12。

2、S50 卡第 1 扇区第 0 块是否可读写？为什么？

答：可以，只有每个扇区的3号数据块和0号扇区的0号数据块不可写

3、“S50 卡共有 16 个扇区，每个扇区由 4 块组成，第 4 块为控制块，其余 三块为数据块，都可用于存储数据”。这句话正确吗？如果不正确，请改正。

答：不正确，0号扇区的0号数据块是用来存储卡的序列号的。应该改正为S50卡除了0号扇区以外的每个扇区其他三块都为数据块，都可用于存储数据，0号扇区的第一块是用来存储卡的序列号。

4、S50 卡的数据块用于存储数据时，可以有哪几种用途？

答：可以用于存储序列号、数据、密码在内的各种二进制数据。

5、如何将一张空白的S50卡初始化成电子钱包

答：将金额初始化为0。

**2.3 实验体会与总结**

在这次实验中，我们主要学习了13.56M这个在生活中使用频率最高的RFID通信标准的使用过程，包括和使用编程语言和其进行无线通信的过程。通过RFID Demo软件来熟悉对于13.56M的RFID卡的通信流程、内存读写、休眠等过程。同时完成阅读和修改一个较大的消费卡模拟系统的代码，使其能够有正常的运行逻辑。主要学会的有以下四点：

1、通过试验箱，学会对 ISO14443A 协议下标签进行寻卡、唤醒、休眠操作， 掌握寻卡、唤醒、休眠的命令代码、命令功能和命令数据包构成，并分析唤醒、 休眠命令对标签寻卡所产生的影响。

2、通过试验箱，学会对 ISO14443A 协议下标签指定内存的数据读写。掌握块 3 存取控制位对本扇区其他块的读写操作控制逻辑。掌握密码 A 和密码 B 的 加载方法以及对不同块的读写控制逻辑。

3、学会通过试验箱对 ISO14443A 协议下标签进行加值减值操作；

4、掌握高频读写器API函数的调用方法，并能够通过编程实现对ISO14443A 协议下标签数据的读写操作以及加值减值操作。

**2.4 核心源码说明**

本次实验基于给定的例子程序进行功能性的修改。



图2.29

如图2.29所示，在注销逻辑运行的过程中，原始代码同时在注册记录表中删除了用户的相关信息。则将对应的删除表项操作注释掉。

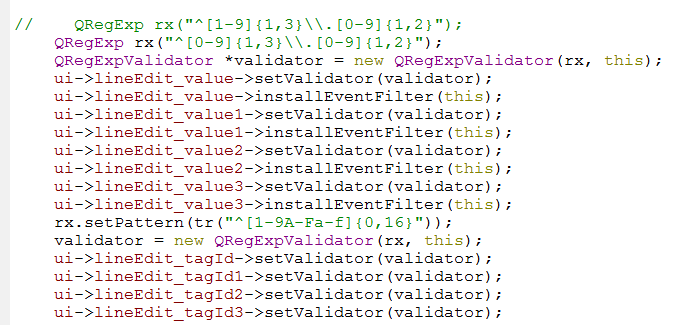


图2.30

如图2.30所示，在原始的金额文本框中匹配的正则表达式撰写错误，会导致在操作过程中无法在数字的整数部位输入0，将正则表达式修改为“^[1-9]{1,3}\\.[0-9]{1,2}”，从而实现支持数字0的输入。这个正则表达式，在三个不同的输入框中出现了，都需要进行修改。

**3 实验三 高频读写器实验ISO15693**

**3.1 实验目的**

通过本次实验了解高频读写器的基本原理，学会如何使用高频读写器，掌握 系统命令参数的意义和设置方式。

进一步加深对ISO15693协议下标签的存储结构以及ISO15693协议的理解。 通过读写器试验箱，掌握对 ISO15693 协议下标签读写操作以及ISO15693协议 标签存储结构的功能，并熟悉高频读写器API函数。

**3.2 实验内容及结果**

3.2.1完成 ISO15693 协议下的单标签和多标签手工寻卡和自动寻卡；



图3.1

如图所示，在RFID Demo中进行电脑软件和读卡器硬件的连接。设置为连接的端口为COM1，传输的波特率为19200，在硬件板上，需要调整打开13.56M模块的电源开关，同时将switch开关切换到13.56模块上，使其绿灯亮起。

成功连接后可以看到在交互信息中成功输出了初始化COM1成功，和接收到的数据报文。

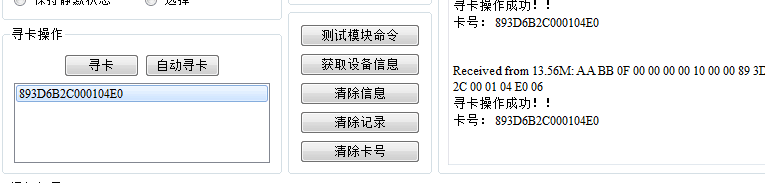


图3.2

如图所示，寻卡操作中有寻卡和自动寻卡，点击了自动寻卡后，程序就会多开一个线程来定时地尝试读取卡的数据，也就是自动循环进行单次寻卡的操作。

3.2.2根据标签内存地址，完成 ISO15693 协议下标签指定地址的数据读写实

验；

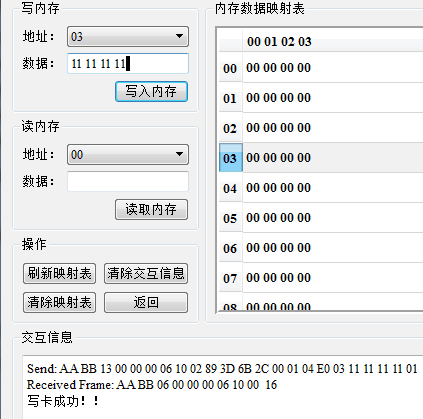


图3.3

如图所示，在内存读写测试程序中，对指定的03地址区域写入11 11 11 11的数据，可以看到在交互信息面板中显示了发送和接受到的数据报文，并提示写卡成功。

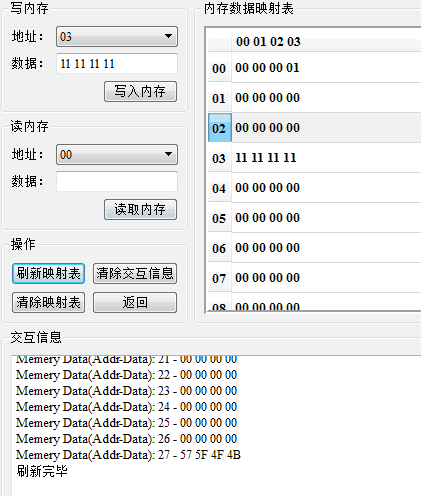


图3.4

如图所示，根据标签的内存地址，可以在读写内存的子模块中，指定一个地址来进行读取和写入。设置写地址的目标为03，写入11 11 11 11，可以看到刷新了映射表后，03位置的数据刷新为了11 11 11 11，说明成功完成了写入。

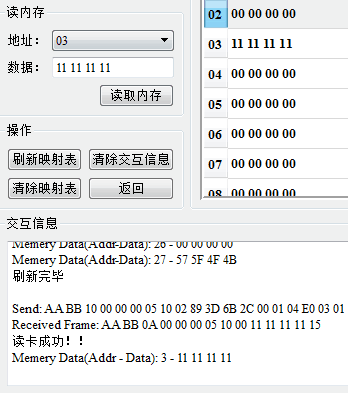


图3.5

如图所示，指定03地址进行读内存操作，输出的数据为11 11 11 11，这既说明了写入的结果是正确的，也说明了读取到了正确的结果。

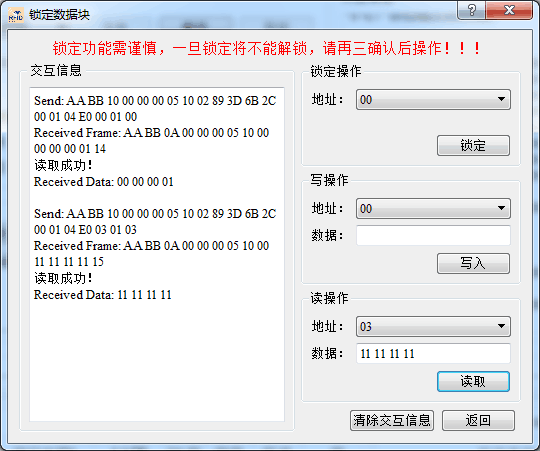


图3.6

如图所示，在数据块读写锁定的操作模块中，可以通过指定数据块的地址，来对其进行读写或锁定，图中对03块进行读取，得到的数据为11 11 11 11，和之前写入的数据相同，运行正常。

3.2.3根据标签内存地址，完成 ISO15693 协议下标签指定地址范围的内存数据读取实验；

图3.6

如图所示，通过指定标签的内存地址范围，可以进行多个数据块的内存数据块进行读取。

3.2.4 ISO15693 协议的命令，完成标签静默状态设置、重置到准备状态、标签 选择命令实验；

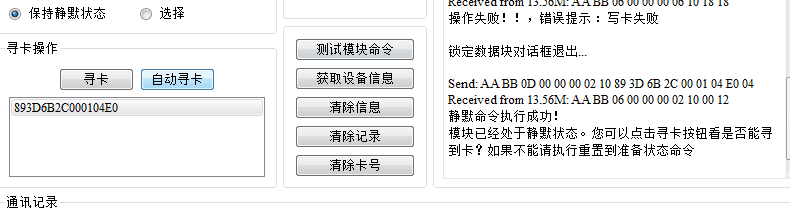


图3.7

如图所示，选择保持静默状态命令，进行测试模块命令。可以看到在交互信息面板中成功输出了静默命令执行成功的信息。

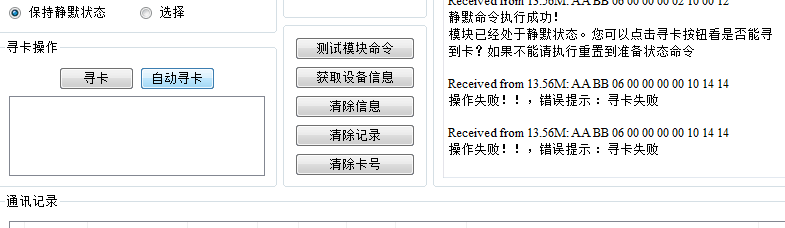


图3.8

如图所示，在静默成功之后，在进行寻卡，会看到交互信息面板输出寻卡失败的提示，由此可知成功进行了休眠。

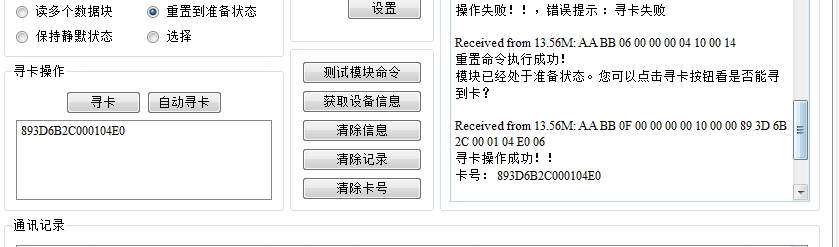


图3.9

如图所示，在测试模块命令重置到准备状态之后，交互信息面板就成功输出重置命令执行成功的结果，重置之后则可以正常的进行寻卡操作。

3.2.5完成 ISO15693 协议下标签 DSFID、AFI 的读写和块安全位的读取实验；

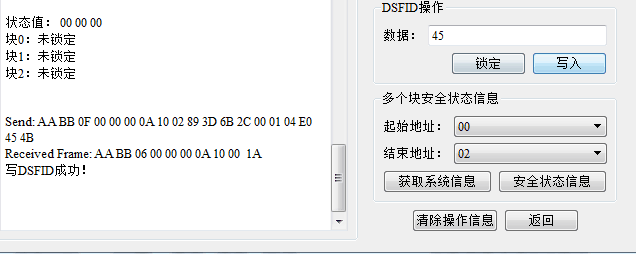


图3.10

如图所示，在特征位操作界面中可以进行DSFID、AFI操作，可以直接指定一个数据进行写入，参考实验书的数据，测试写入45，交互信息面板输出了写DSFID成功的信息。

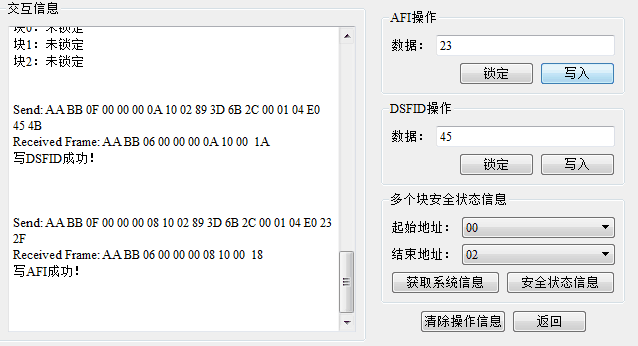


图3.11

如图所示，参考实验书的数据，AFI操作写入数据23，交互信息面板中输出写AFI成功的提示信息。

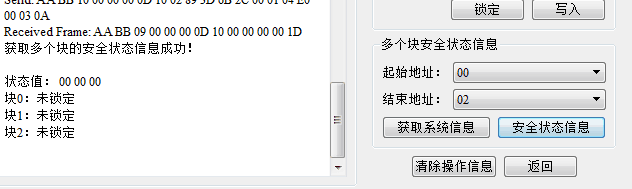


图3.12

如图所示，测试多个块的安全状态信息，选择起始地址为00，结束地址为02。就可以获取到0,1,2三个数据块的安全状态信息，即块是否被加锁。



图3.13

如图所示，同样的特征位操作中，可以通过读取系统信息来获得卡片的ID、内存容量等信息。

3.2.6熟悉和了解高频 HF1356M 15693 开发实例，掌握高频读写器 API 函数， 并通过编程实现 ISO15693 协议下标签的读写功能。

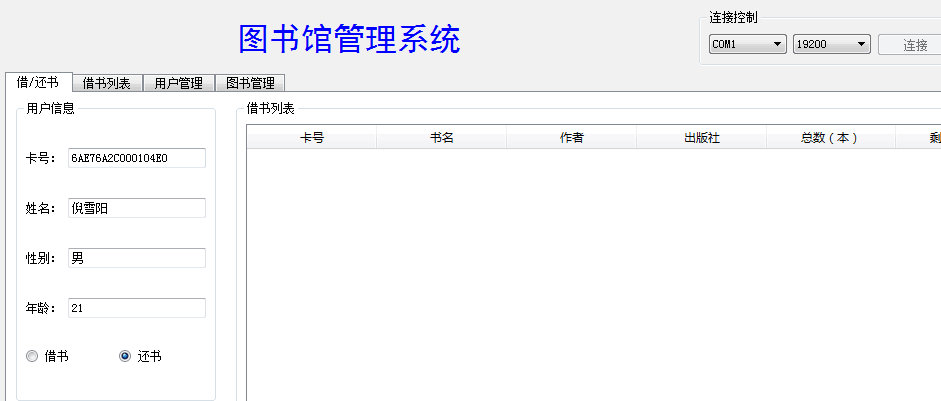


图3.14

如图所示，HF1356M 15693高频标签的应用场景之一就是图书馆管理系统。可以使用卡片来模拟用户卡和图书卡。

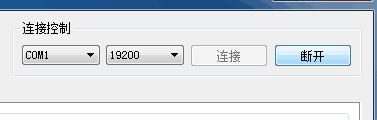


图3.15

如图所示，在右上角的连接控制中可以进行连接电脑软件和读卡器硬件，在这里使用了端口COM1，和波特率为19200的交互通信。

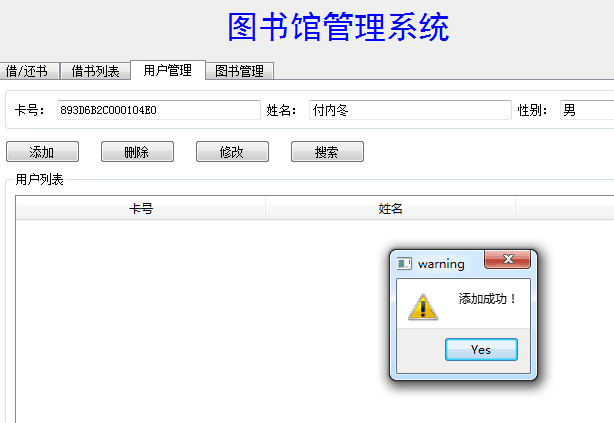


图3.16

如图所示，用户管理界面中，刷一张空白卡后，会自动读取出卡号，输入姓名、性别、年龄等信息后即可以直接添加用户信息，可以看到软件弹出提醒说明添加成功。

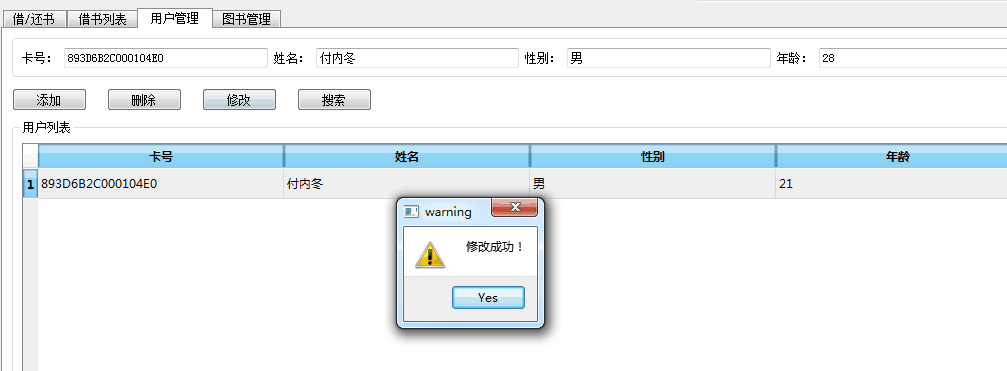


图3.17

如图所示，重新输入数据后，可以点击修改按钮进行修改，程序也弹出了提示修改成功。



图3.18

如图所示，在图书管理选项卡中刷一张新卡，能够自动读取卡号，填写完善信息后，就可以完成添加。在下方的图书列表可以看到刚刚添加进去的图书。



图3.19

如图所示，在借还书界面刷用户卡后，动作选为借书后，刷任意书籍卡就可以完成结束操作。在右侧的借书列表中可以看到新增的借书记录。



图3.20

如图所示，在完成了借书行动后，在借书列表中可以查询到借书的记录，从而方便图书馆管理员进行管理。

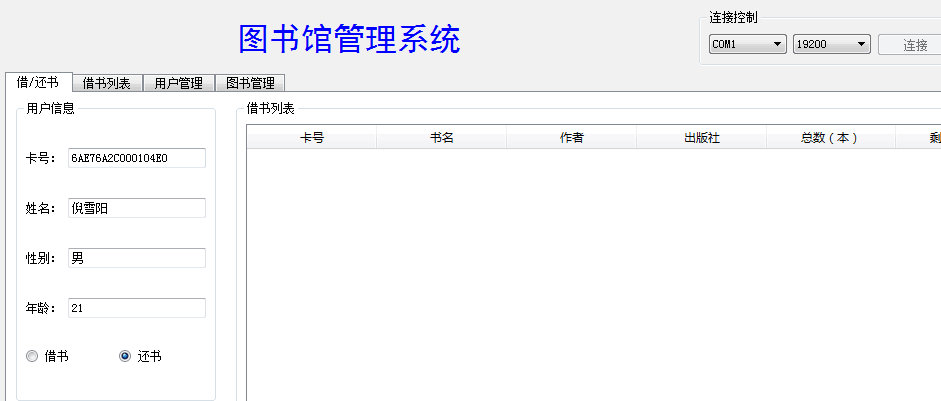


图3.21

如图所示，在借还书界面中，选择还书进行刷卡后，可以看到借书列表中对应的表项被删除了。

C:\Users\28529\OneDrive\RFID实验\第三次实验\23。代码.png

图3.22

如图所示，在程序的控制台输出中可以看到输出了对应的删除借书信息的SQL语句。

3.2.7 思考题

1、什么是 AFI？AFI 如何编码？在通过编程对 AFI 进行读写、锁定时，其

对应 ISO15693 的协议命令代码、上位机对读写器的命令代码和数据包分别是怎 么样的？

答：AFI 是应用标识，表示读写器锁定的应用类型，仅选择符合应用类型的标签。 由于为了防止卡片被锁定后无法进行解锁，我们在实验中并没有直接进行锁定卡片。但经过查询文档后得知，对与AFI写入的命令代码为0x27，对AFI锁定的命令代码为0x28。发送的请求数据包打包了帧开始标志SOF，标识Flags，命令代码Command Code，参数Parameters，数据Data，CRC，帧结束EOF。

2、什么是DSFID？在通过编程对DSFID进行读写、锁定时，其对应ISO15693 的协议命令代码、上位机对读写器的命令代码和数据包分别是怎么样的？

答：DSFID 是数据存储格式标识识，DSFID指明了标签的数据存储格式。由于为了防止卡片被锁定后无法进行解锁，我们在实验中并没有直接进行锁定卡片。但经过查询文档后得知，对与DSFID写入的命令代码为0x29，对DSFID锁定的命令代码为0x2A。发送的请求数据包打包了帧开始标志SOF，标识Flags，命令代码Command Code，参数Parameters，数据Data，CRC，帧结束EOF。

3、ISO15693 协议的电子标签 ID 有何特点？

答：工作频率范围为13.56MHz±7KHz，工作磁场中工作场最小值0.15A/m,最大值5A/m，支持多种调制方式，支持防冲突和传输协议。

**3.3 实验体会与总结**

1、学会了通过试验箱对 ISO15693 协议下标签指定内存地址的数据进行读写

操作；

2、理解和掌握应用族标识符（AFI）、数据存储格式标识符（DSFID）以及锁的基本概念和含义；

3、掌握高频读写器 API 函数的调用方法，并能够通过编程实现对 ISO15693 协议下标签数据的读写控制。

**3.4 核心源码说明**

Booksmanage.cpp文件中主要负责前端和图书数据库的交互逻辑，由于和数据库交互的逻辑大同小异。举一个添加书籍的函数为例子。

void BooksManage::add\_books()//添加按钮槽函数

{

int residue;//图书的剩余数量

/\*文本框为空时显示错误提示\*/

QString LabelName[] = {"卡号：", "书名：", "作者：", "出版社：", "总数（本）"};

for(int i = 0; i < Edit\_Count\_BOOKS-1; i++)

{

if(Edit[i]->text().isEmpty())

{

QMessageBox::warning(NULL, "warning", LabelName[i]+"不能为空！", QMessageBox::Yes, QMessageBox::Yes);

return;

}

}

if (sql->SelectUser(Edit[ID\_Books]->text()).next())

{

QMessageBox::warning(NULL, "warning", "卡号已经注册为用户！", QMessageBox::Yes, QMessageBox::Yes);

return;

}

if (Edit[Residue\_Books]->text().toInt() > Edit[Count\_Books]->text().toInt())

{

QMessageBox::warning(NULL, "warning", "剩余数量不可以超出总数！", QMessageBox::Yes, QMessageBox::Yes);

return;

}

/\*不填写剩余数量默认为总数量\*/

if (Edit[Residue\_Books]->text().isEmpty())

{

residue = Edit[Count\_Books]->text().toInt();

}

else

{

residue = Edit[Residue\_Books]->text().toInt();

}

//向数据库中添加书籍

bool ret = sql->InsertBooks(Edit[ID\_Books]->text(),Edit[Name\_Books]->text(),Edit[Author\_Books]->text(),Edit[PublishingHouse\_Books]->text(),Edit[Count\_Books]->text().toInt(),residue);

if(!ret)

{

QMessageBox::warning(NULL, "warning", "添加失败，卡号已存在！", QMessageBox::Yes, QMessageBox::Yes);

return;

}

QMessageBox::warning(NULL, "warning", "添加成功！", QMessageBox::Yes, QMessageBox::Yes);

ClearEdit(); //清空文本框

ShowTable(sql->SelectBooks());//更新表格

}

如上代码所示，先检查添加书籍的每一条信息填写的是否正确，是否有空行，防止写入的信息出现了错误。然后在数据库中查询这张卡号相关的信息，如果这卡号已经被注册过，则提醒一个warning来提示已经注册。检查填写的新信息是否符合逻辑，如书本的剩余数量是否高于总数量，如果是则使用一个warning来进行提醒。没有填写剩余数量，则自动设置为所有数量均为剩余。确定完信息后，通过数据库写入，来保存新的书籍信息。

**4 实验四 超高频读写器实验**

**4.1 实验目的**

通过本次实验了解超高频读写器的基本原理，学会如何使用超高频读写器，掌握超高频读写器和标签参数的含义和设置方法。

进一步加深对 Gen2 协议下标签的存储结构以及 Gen2 协议的理解。通过读写 器试验箱，掌握对 Gen2 协议下标签读写操作，并熟悉超高频读写器 API 函数的调用。

**4.2 实验内容及结果**

4.2.1超高频读写器的基本认知，完成超高频读写器频率和功率读取和设置实

验；

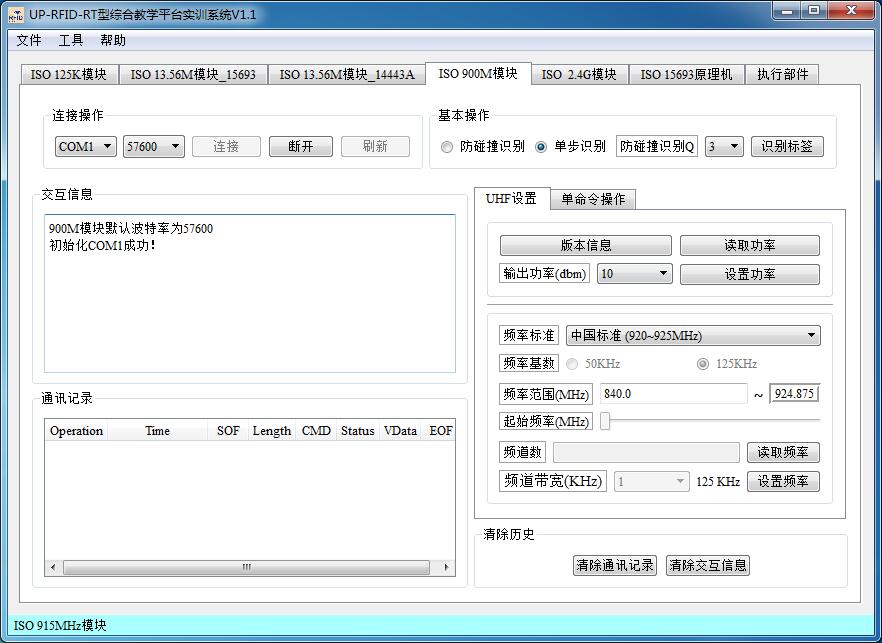


图4.1

如图所示，运行RFID Demo软件进行测试。先通过串口来连接计算机软件和读卡器硬件。可以点击左上角的连接操作面板中的连接按钮，看到交互信息面板中输出了初始化成功的提醒。

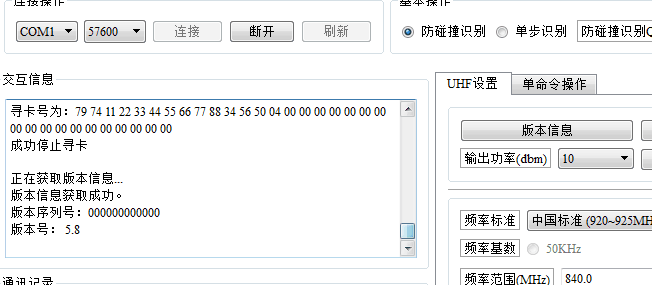


图4.2

如图所示，在界面中点击UHF设置中的版本信息按钮后，交互信息面板中输出了相关的版本信息。

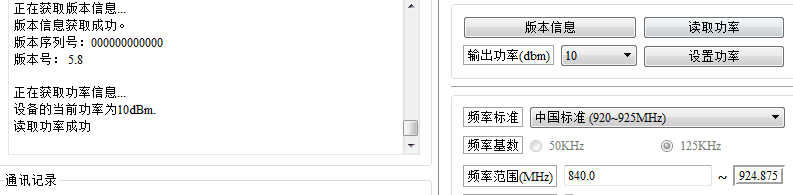


图4.3

如图所示，点击了读取功率按钮后，看到交互信息面板中输出了设备当前的功率值。

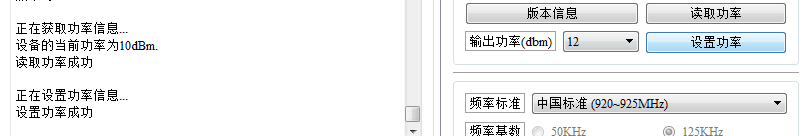


图4.4

如图所示，设置一个输出功率之后，可以点击设置功率来为其设置一个不同的功率值。

4.2.2完成 Gen2 协议下单标签和多标签识别实验；

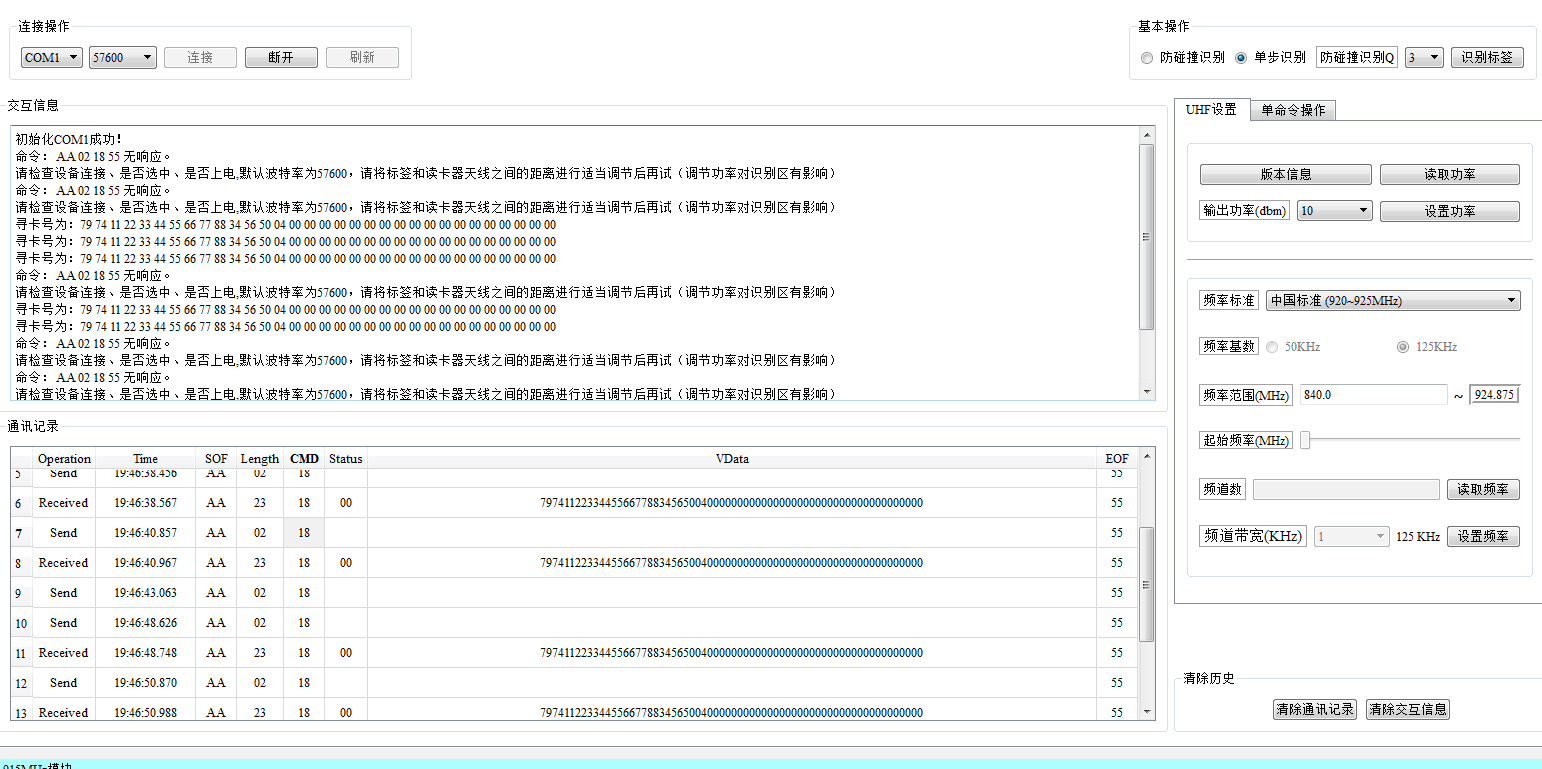


图4.5

如图所示，点击单步识别来识别卡片，点击了多次，可以看到在天线和卡片有不同的距离的情况下，连接的情况有一些不稳定的现象。

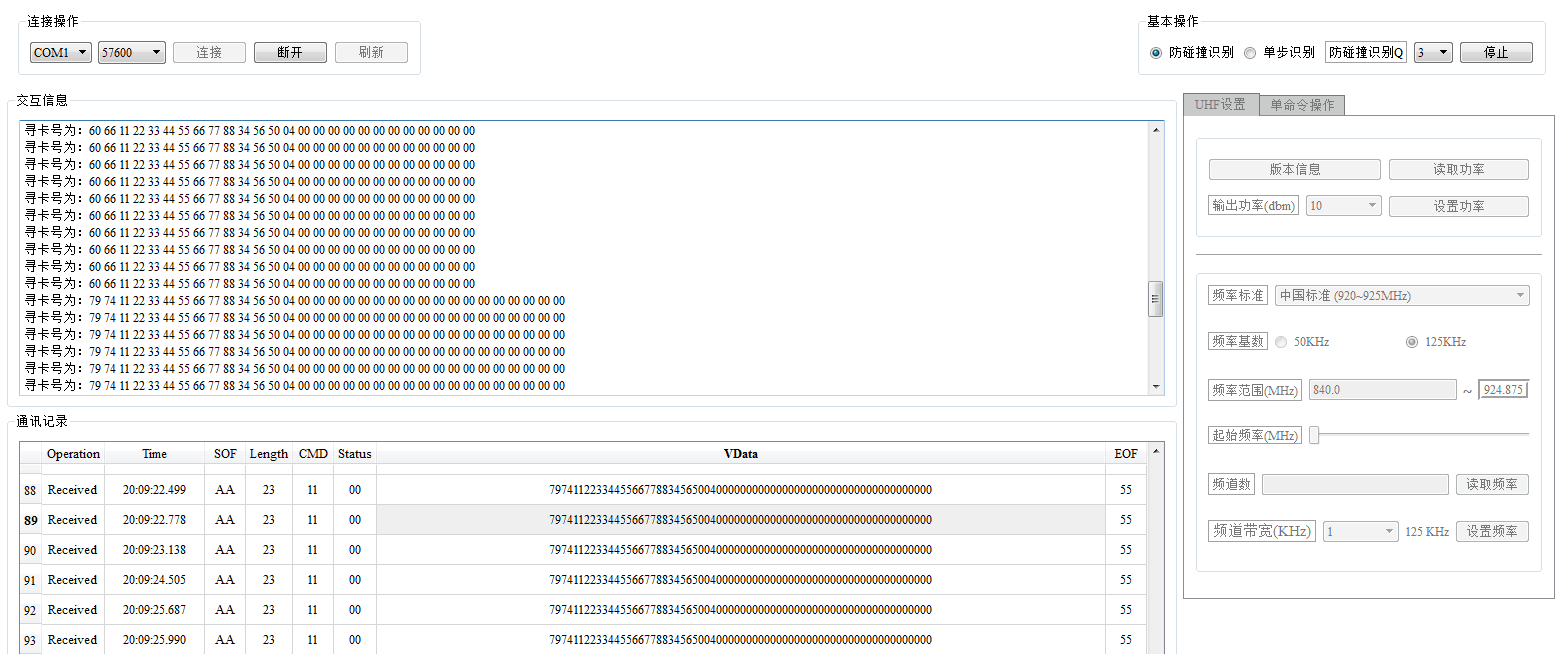


图4.6

如图所示，在多张卡片的情况下点击自动防碰撞识别，可以看到能够顺利读取出多张卡片的卡号，说明算法自动规避了碰撞情况，完成了数据通信。

4.2.3执行 Gen2 协议下单命令操作实验，并分别对 EPC 标签各个存储区进行 读写擦除操作试验；

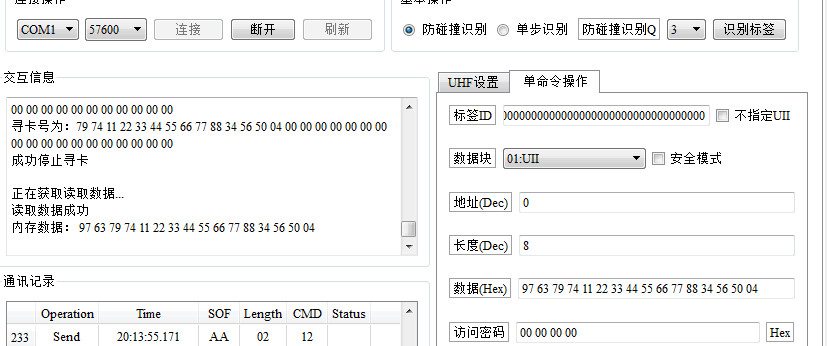


图4.7

如图所示，在单命令操作中，选择读取数据操作，可以看到在交互信息的界面中输出了读取得到的内存数据。

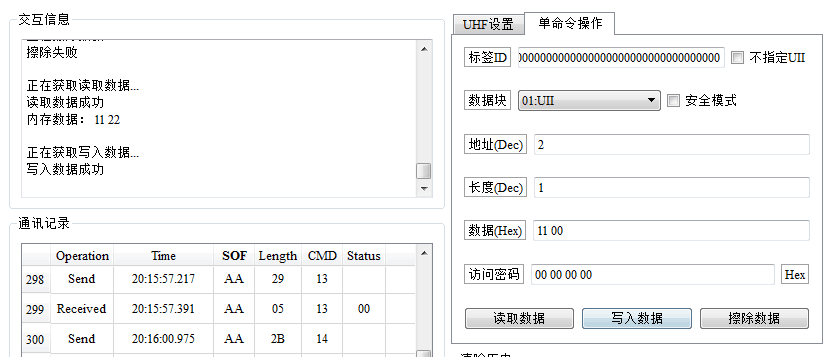


图4.8

如图所示，选择了写入数据操作后，指定一个地址位置、长度、待写入的数据，就可以对这个位置写入一个定长长度的数据，可以看到在交互信息面板中输出了写入数据成功的提示。

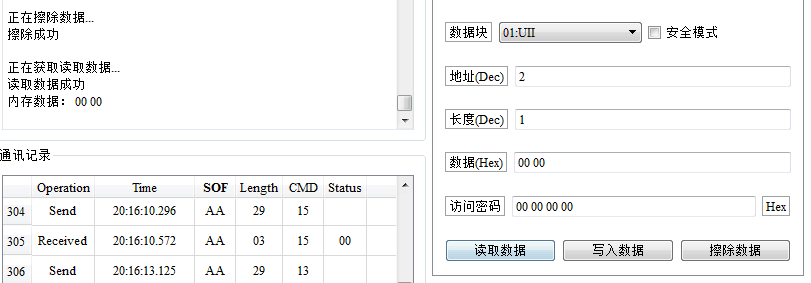


图4.9

如图所示，选择了擦除数据命令后，交互信息面板输出了查处成功的提醒。再次尝试读取数据之后，可以看到读取得到的数据为00 00，说明数据被成功擦除了。

4.2.4熟悉和了解超高频 UHF-900M 开发实例，掌握超高频读写器 API 函数， 并通过编程实现 Gen2 协议下标签的读写功能。

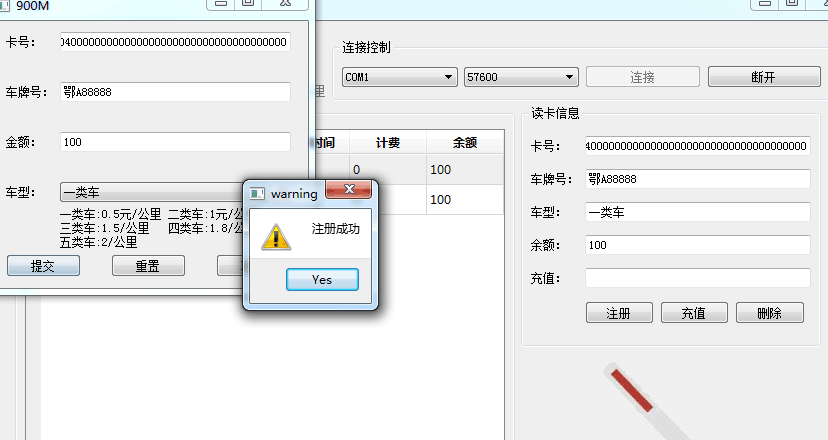


图4.10

如图所示，以一个汽车收费站作为高频RFID读写实验的例子。在右上角的连接控制中指定串口端口号和波特率，就可以完成连接。刷任意一张空白卡之后，自动读取出一个卡号，就可以填入车牌号、初始化金额，来完成卡片的注册。

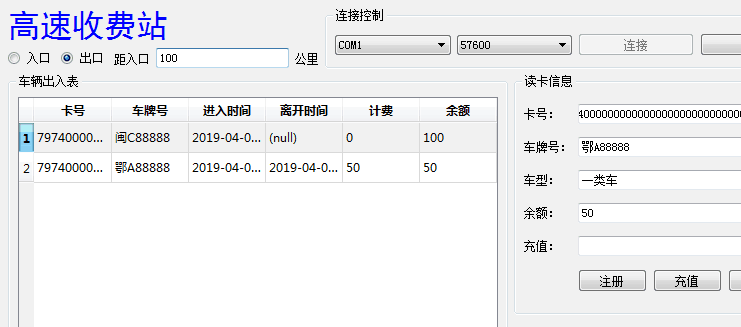


图4.11

如图所示，在注册好卡之后，再刷卡片就可以记录对应车俩的进出信息，同时记录下时间、车牌、余额等信息。指定一个出口的收费金额字后，就可以自动进行扣款。在车辆出入表中可以看到车辆初始化100金额后，出除了收费站被收费50，只剩下50，符合预期的运行结果。

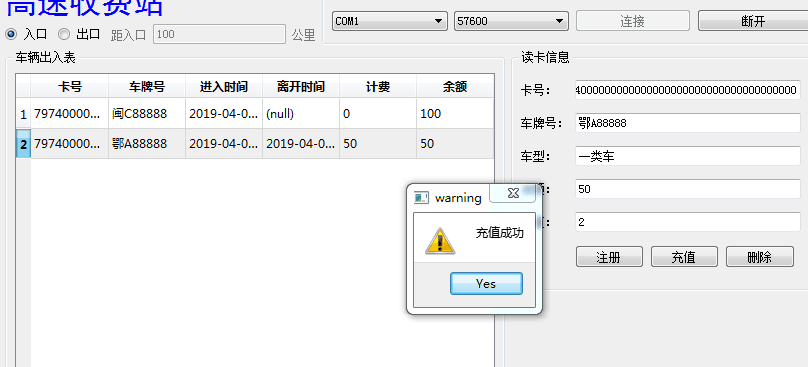


图4.12

如图所示，刷一张已经注册的卡之后，输入一个指定的金额再点击充值，就可以看到金额被成功充值进入，程序弹框提示充值成功。

4.2.5 思考题

1、试从定性和定量两个方面分析读写器功率与标签读写距离的关系；

答：从定性的角度来说，随着读写器功率的增大，标签读写距离更长。从定量的角度来说，读写器的功率单位为dBm，所以功率的改变对读取的距离的影响为指数级别。

2、EPC 标签存储器分为哪几个区？各区有何功能？

答：分为Tag memory（标签内存）分为Reserved（保留），EPC（电子产品代码），TID（标签识别号）和User（用户）四个独立的存储区块（Bank）。Reserved区：存储Kill Password（灭活口令）和Access Password（访问口令）。EPC区：存储EPC号码等。TID区：存储标签识别号码，每个TID号码应该是唯一的。User区：存储用户定义的数据。

3、EPC 标签可以通过哪些措施来保证各个存储区的信息安全？

答：EPC标签通过有限状态机的思想来构建了七种状态，只有在指定的几个状态才能进行读写锁定等操作，同时进入操作的状态时，需要先经过Access Password的验证。

4、EPC 和 TID 分别表示什么含义？二者结构有何特点？

答：EPC指的是Electronic Product Code电子产品代码，为每一个单品进行跟踪和溯源，TID指的是Tag Identifier标签识别号，是标签之间身份区别的标志。EPC码可以被复制，但TID是唯一的，具有一个Lock状态位，来防止出厂后被改写，才能被永久锁定。

**4.3 实验体会与总结**

在本次实验中，无线通信的模块由于支持了较远情况下的高频通信，在硬件的配置上，经常出现错误，不能正常读取出卡片的信息，在这个地方进行了比较长时间的调整过程，才使得能够正常读取900M卡片。

1、学会通过试验箱对 Gen2 协议下标签指定存储区的数据读写；

2、理解和掌握 Gen2 协议下标签存储器结构的特点及含义；

3、通过多次设置读写器功率，不断移动电子标签与读写器之间的距离，分析理解读写器功率和频率对电子标签读写的影响；

4、理解和掌握访问密码的用途和使用方法；

5、掌握超高频读写器 API 函数的调用方法，并能够通过编程实现对 Gen2 协议下标签数据的读写控制。

**4.4 开发实例源码**

核心的源码部分为widget.cpp文件中的get\_info函数，每次感应到卡片的时候，都会进行回调调用。

void Widget::Get\_Info(QString cardID)//刷卡响应

{

float consumption,balance;//消费、余额

QString id, plate\_number, time;//卡号、车牌号、当前时间

/\*清空显示\*/

for(int i = 0; i < EDIT\_COUNT; i++)

{

Edit[i]->clear();

}

mutex.lock();//上锁

char where[256];

sprintf(where, "cardID = '%s'", cardID.toUtf8().data());

QSqlQuery query = sqlite->select("user\_900M",where);//获取查询user表中卡的信息的结果

if(!query.next())

{

mutex.unlock();

if(CardID!=cardID)

{

QMessageBox::warning(NULL, "warning", "卡号未注册！", QMessageBox::Yes, QMessageBox::Yes);

CardID = cardID;

}

Edit[0]->setText(cardID);

return ;

}

/\*取 卡号、车牌号、余额、当前时间\*/

id = query.value(0).toString();

plate\_number = query.value(1).toString();

time = QDateTime::currentDateTime().toString("yyyy-MM-dd hh:mm:ss ddd"); //获取时间

balance = query.value(3).toFloat();

/\*当收费站为入口时\*/

if(In->isChecked())

{

if( balance <= 0 )

QMessageBox::warning(NULL, "warning", "余额不足！", QMessageBox::Yes, QMessageBox::Yes);

else

In\_station(id, plate\_number, time, balance);//调用进站处理函数

}

/\*当收费站为出口时\*/

if(Out->isChecked())

{

if(Distance->text().isEmpty())//没有输入距离时

{

QMessageBox::warning(NULL, "warning", "请输入距入口的公里数!", QMessageBox::Yes, QMessageBox::Yes);

}

else

{

/\*根据车的类型计算收费\*/

int type = 0;//车的类型

QString types[] = {"一类车","二类车","三类车","四类车","五类车"};

for(int i = 0; i < 5; i++)

{

if(query.value(2).toString() == types[i])

{

type = i;

break;

}

}

float distance = Distance->text().toFloat();//

if( Distance->text().isEmpty() || distance < 1)

distance = 1;

consumption = distance \* Charge\_standard[type];//消费

Out\_station(id, plate\_number, time, consumption, balance);//调用出站处理函数

}

}

mutex.unlock();//解锁

/\*uhf区域显示相关信息\*/

for(int i = 0; i < EDIT\_COUNT-1; i++)

{

Edit[i]->setText(query.value(i).toString());

}

//更新表

Show\_Table();

}

如上所示，每次刷卡响应的时候，先清空当前的显示界面，然后进行信号量的上锁，读取得到卡号等信息后，对SQLite数据库进行查询操作，对user\_900M表进行查询。若查询没有结果，则弹出一个warning无该卡号的警告。反之则通过卡号来获得车牌号、余额、当前时间等信息。判断当前的收费站是否为出口，若是，则进行判断case来进行扣费。反之，判断余额是否大于0，若大于0则能通行，若不大于则弹出一个warning提示余额不足。对信号量进行解锁，再进行展示。通过对信号量的加锁解锁操作，来保证不会同时访问一个区域，来保证数据库数据的安全性。

**5 实验五 RFID综合应用实验**

**5.1 需求分析**

**5.2 系统详细设计**

**5.3 系统实现与系统测试**

**5.4 总结**

**5.5 系统源代码**

五号字体，单倍行距