智能仓储系统的开发研究

先进计算与机器人研究所

2023 年 7 月 4 日

目录

1 第五章: 射频识别技术 (RFID-RC522)

在上一章中, 我们是直接给的物品种类以及单个物品质量的信息。实际上这是从 RC522 读卡器上的卡读到的。首先来认识 RC522 模块,

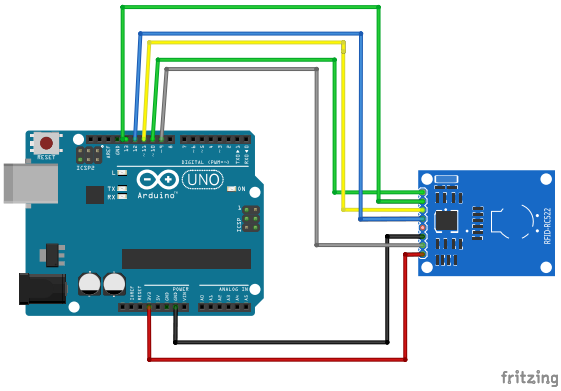

图 1: RC522 硬件连接

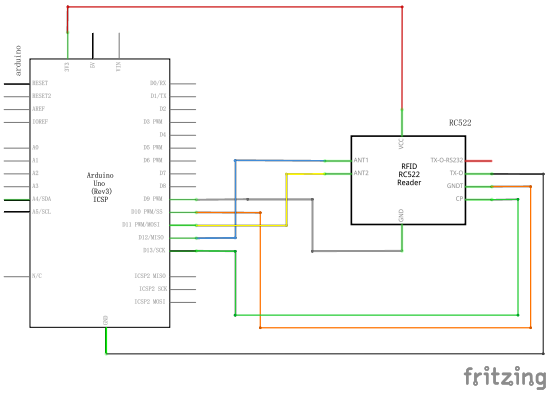

图 2: RC522 连接示意图

RC522 模块共有 8 个与外界连接的引脚, 与 arduino 的连接如图所示:

- VCC 为模块供电, 连接到 Arduino 的 3.3V 输出。;
- RST 是复位和掉电的输入。当该引脚变为低电平时, 关闭所有内部电流吸收器, 包括振荡器, 并且输入引脚与外界断开连接。在上升沿, 模块被重置;
- GND 是接地引脚, 连接到 Arduino 的 GND 引脚;
- IRQ 是一个中断引脚, 可在 RFID 标签进入附近时向微控制器发出警报;
- 当使用 SPI 接线时, MISO 上的数据从从机输出到主机;
- 当使用 SPI 接线时, MOSI 上的数据从主机输出到从机;
- SCK 是串行时钟信号, 由主机产生发送给从机;
- ss 上信号由主机发送, 以控制与哪个从机通信, 通常是低电平有效信号。

在 MFRC522 模块上只有两个引脚 RST 和 SS 可以自己定义, 这里将其定义为引脚 9 和引脚 10(除了引脚 11, 引脚 12, 引脚 13 外的任意空闲数字引脚皆可, 引脚 11,12,13 已经与 RC522 的引脚 MOSI,MISO,SPI 连接。将其定义为引脚 9 和引脚 10 是常见布局)。

1.1 RC522 类的声明

RC522 类就对应了整个 RC522 模块的功能。

```
class RC522{
private:
  int RST_PIN, SS_PIN;
  MFRC522 mfrc522;

public:
  char Type_Name[3];
  long Single_Weight;
  long Shell_Weight;

  RC522();
  void initialize(int p1, int p2);
  byte read();
  bool write();
}
```

- 私有变量: RST_PIN, SS_PIN 分别是与 RC522 的 RST,SS 连接的 arduino 引脚;
- 公共变量: Type_Name, Single_Weight, Shell_Weight 是储存从读卡器中读到的物品种类, 单个物品质量, 每一层物品质量;

默认构造函数:

```
RC522::RC522(){};
```

初始化函数用于引脚的赋值, 初始化 SPI 通信, 初始化 MFRC522 类。

```
void RC522::initialize(int p1, int p2){
  RST_PIN=p1;
  SS_PIN=p2;
  SPI.begin();
  mfrc522 = MFRC522(SS_PIN, RST_PIN);
  mfrc522.PCD_Init();
};
```

1.2 读卡模式

将下述代码烧录运行, 就做好了读卡准备。

```
byte RC522::read(){
  // init the read state
  byte read_state = 0;
  //default key
  MFRC522::MIFARE_Key key;
  for (byte i = 0; i < 6; ++i) key.keyByte[i] = 0xFF;
  //some variables we need
  byte block;
  byte len;
  MFRC522::StatusCode status;
  //-------------------------------------------
  // Reset the loop if no new card present on the sensor/reader. This saves the entire process when idle.
  if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return read_state;
  }
  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial()) {
    return read_state ;
  }
  // read one card
  read_state = 1;
  Serial.println(F("**Card Detected:**"));
  //-------------------------------------------
  // mfrc522.PICC_DumpDetailsToSerial(&(mfrc522.uid)); //dump some details about the card
  // mfrc522.PICC_DumpToSerial(&(mfrc522.uid)); //uncomment this to see all blocks in hex
  //-------------------------------------------

  byte buffer1[18];
  block = 1;
  len = 18;
  //------------------------------------------- GET TYPE NAME
  status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(mfrc522.uid)); //line 834 of MFRC522.cpp file
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("Authentication failed: "));
```

```
35      Serial.println(mfrc522.GetStatusCodeName(status));
36      return read_state;
37    }
38    status = mfrc522.MIFARE_Read(block, buffer1, &len);
39    if (status != MFRC522::STATUS_OK) {
40      Serial.print(F("Reading␣failed:␣"));
41      Serial.println(mfrc522.GetStatusCodeName(status));
42      return read_state;
43    }
44    //PRINT TYPE NAME
45    Type_Name[0] = buffer1[0];
46    Type_Name[1] = buffer1[1];
47    long singleweight = 0;
48    for (uint8_t i = 2; i < 6; ++i)
49    {
50      singleweight = singleweight*10 + (buffer1[i]-48);
51    }
52    Single_Weight = singleweight;
53    Serial.print(F("Type␣Name:␣"));
54    Serial.println(Type_Name);
55    Serial.print(F("Type␣Single␣Weight:␣"));
56    Serial.println(Single_Weight);
57    //---------------------------------------- GET WEIGHT
58
59    byte buffer2[18];
60    block = 4;
61    status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(
          mfrc522.uid)); //line 834
62    if (status != MFRC522::STATUS_OK) {
63      Serial.print(F("Authentication␣failed:␣"));
64      Serial.println(mfrc522.GetStatusCodeName(status));
65      return read_state;
66    }
67    status = mfrc522.MIFARE_Read(block, buffer2, &len);
68    if (status != MFRC522::STATUS_OK) {
69      Serial.print(F("Reading␣failed:␣"));
70      Serial.println(mfrc522.GetStatusCodeName(status));
71      return read_state;
72    }
73    //PRINT WEIGHT
74    int shellweight = 0;
75    for (uint8_t i = 1; i < 16; ++i) {
76      // Serial.write(buffer2[i] );
77      // Serial.println();
78      // Serial.println(buffer2[i]);
79      if(buffer2[i] == 32) break;
80      shellweight = shellweight*10 + (buffer2[i]-48);
81    }
82    Shell_Weight = shellweight;
83    Serial.print(F("Shell␣Weight:␣"));
84    Serial.println(Shell_Weight);
85    //----------------------------------------
86    Serial.println(F("**End␣Reading**\n"));
87    read_state = 2;
88    mfrc522.PICC_HaltA();
89    mfrc522.PCD_StopCrypto1();
90    return read_state;
91  }
```

1.3 写卡模式

将下述代码烧录运行, 就做好了写卡准备。写卡时会在串口提示输入两次字符串, 每次都是以 # 结尾。第一次是输入物品种类以及单个物品质量, 例如'AA0010#' 表示物品种类是'AA', 单个物品质量是 10g; 第二次是输入外壳质量, 例如'0#' 表示外壳质量是 0。

```
bool RC522::write(){
  //初始化读卡状态
  //未读到卡为0，读到卡为1
  bool write_state = 0;
  //创建访问密钥，用于验证并访问 MIFARE Classic RFID标签
  //这里用默认卡密钥
  MFRC522::MIFARE_Key key;
  for (byte i = 0; i < 6; i++) key.keyByte[i] = 0xFF;

  //block为卡的不同区域编号
  //len3为读到的字节数
  //status为判断对卡操作是否成功完成的状态变量
  byte block;
  byte len3;
  MFRC522::StatusCode status;

  // 如果传感器/读卡器上没有新卡，则复位循环。这可以在空闲时保存整个进程。
  if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return write_state;
  }

  // 选择一张卡片进行读取
  if ( ! mfrc522.PICC_ReadCardSerial()) {
    return write_state;
  }

  Serial.println(F("**Card␣Detected:**"));
  // 成功读取到卡片，将读卡状态设为1
  write_state = 1;

  //打印UID编号
  Serial.print(F("Card␣UID:"));
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? "␣0" : "␣");
    Serial.print(mfrc522.uid.uidByte[i], HEX);
  }
  //打印PICC类型
  Serial.print(F("␣PICC␣type:␣"));
  MFRC522::PICC_Type piccType = mfrc522.PICC_GetType(mfrc522.uid.sak);
  Serial.println(mfrc522.PICC_GetTypeName(piccType));

  byte buffer3[34];
  //等待20秒从串口输入
  Serial.setTimeout(20000L) ;
  // 提示:输入类型名称
  Serial.println(F("Type␣name,␣ending␣with␣#"));
  //从串口读入类型名称到buffer3，len为写入字节长度
  len3 = Serial.readBytesUntil('#', (char *) buffer3, 30) ;
  // 将未满字节用空格填补
  for (byte i = len3; i < 30; i++) buffer3[i] = '␣';

  block = 1;
  //选择读取区域和密钥进行身份验证，验证失败打印错误信息并提前终止
  status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(
      mfrc522.uid));
```

```
55    if (status != MFRC522::STATUS_OK) {
56      Serial.print(F("PCD_Authenticate()␣failed:␣"));
57      Serial.println(mfrc522.GetStatusCodeName(status));
58      return write_state;
59    }
60    else Serial.println(F("PCD_Authenticate()␣success:␣"));
61
62    // 对前述验证成功的区域内的信息写入字节数组，写入失败打印错误信息并提前终止
63    status = mfrc522.MIFARE_Write(block, buffer3, 16);
64    if (status != MFRC522::STATUS_OK) {
65      Serial.print(F("MIFARE_Write()␣failed:␣"));
66      Serial.println(mfrc522.GetStatusCodeName(status));
67      return write_state;
68    }
69    else Serial.println(F("MIFARE_Write()␣success:␣"));
70
71    block = 2;
72    //选择读取区域和密钥进行身份验证，验证失败打印错误信息并提前终止
73    status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(
          mfrc522.uid));
74    if (status != MFRC522::STATUS_OK) {
75      Serial.print(F("PCD_Authenticate()␣failed:␣"));
76      Serial.println(mfrc522.GetStatusCodeName(status));
77      return write_state;
78    }
79
80    // 对前述验证成功的区域内的信息写入字节数组的地址，写入失败打印错误信息并提前终止
81    status = mfrc522.MIFARE_Write(block, &buffer3[16], 16);
82    if (status != MFRC522::STATUS_OK) {
83      Serial.print(F("MIFARE_Write()␣failed:␣"));
84      Serial.println(mfrc522.GetStatusCodeName(status));
85      return write_state;
86    }
87    else Serial.println(F("MIFARE_Write()␣success:␣"));
88
89    byte buffer4[34];
90    byte len4;
91    // Ask personal data: First name
92    Serial.println(F("Type␣Weight,␣ending␣with␣#"));
93    len4 = Serial.readBytesUntil('#', (char *) buffer4, 20) ; // read first name from
          serial
94    for (byte i = len4; i < 20; i++) buffer4[i] = '␣'; // pad with spaces
95
96    block = 4;
97    //选择读取区域和密钥进行身份验证，验证失败打印错误信息并提前终止
98    status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(
          mfrc522.uid));
99    if (status != MFRC522::STATUS_OK) {
100     Serial.print(F("PCD_Authenticate()␣failed:␣"));
101     Serial.println(mfrc522.GetStatusCodeName(status));
102     return write_state;
103   }
104
105   // 对前述验证成功的区域内的信息写入字节数组，写入失败打印错误信息并提前终止
106   status = mfrc522.MIFARE_Write(block, buffer4, 16);
107   if (status != MFRC522::STATUS_OK) {
108     Serial.print(F("MIFARE_Write()␣failed:␣"));
109     Serial.println(mfrc522.GetStatusCodeName(status));
110     return write_state;
111   }
112   else Serial.println(F("MIFARE_Write()␣success:␣"));
```

```
113
114     block = 5;
115     //选择读取区域和密钥进行身份验证，验证失败打印错误信息并提前终止
116     status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(
            mfrc522.uid));
117     if (status != MFRC522::STATUS_OK) {
118       Serial.print(F("PCD_Authenticate()␣failed:␣"));
119       Serial.println(mfrc522.GetStatusCodeName(status));
120       return write_state;
121     }
122
123     // 对前述验证成功的区域内的信息写入字节数组的地址，写入失败打印错误信息并提前终止
124     status = mfrc522.MIFARE_Write(block, &buffer4[16], 16);
125     if (status != MFRC522::STATUS_OK) {
126       Serial.print(F("MIFARE_Write()␣failed:␣"));
127       Serial.println(mfrc522.GetStatusCodeName(status));
128       return write_state;
129     }
130     else Serial.println(F("MIFARE_Write()␣success:␣"));
131     delay(1000);
132
133     //--------------成功完成读取----------------
134     Serial.println(F("\n**End␣Reading**\n"));
135     //停止响应当前正在运行的命令。
136       //将 RFID 卡片置于休眠状态，以便进行下一个命令的执行。
137     mfrc522.PICC_HaltA();
138     //停止当前正在进行的加密。如果 RFID 模块正在与 RFID 卡片进行加密通信，
139     //则此代码将停止加密，并将模块置于初始状态，以便进行下一个操作。
140     mfrc522.PCD_StopCrypto1();
141     //返回写卡状态
142     return write_state;
143   }
```

1.4 RC522.ino

在上一章 master.ino 基础上, 加入了通过读卡获得 rc522 的 Single_Weight 和 Type_Name。并通过 rflag 是否为 1 判断是读还是写。目前有一点不妥的是, 当没有读到卡时, 在 switch 语句中会一直在 while 中循环直到读到卡。也就是说即使物品总质量发生变化, 也只有在重新放卡后才能发送消息到上位机, 要不然出不了 while 循环语句。

```
1   #include "master.h"
2   #include "transform.h"
3   #include "Surface.h"
4   #include "Calibrate.h"
5   #include "Oled.h"
6   #include "RC522.h"
7
8   master m1;
9   Surface YL_Surface;
10  Calibrate YL_Calibrated;
11  transform tf;
12  Oled oled;
13  RC522 rc522;
14
15  long numbefore=0, numnow=1;
16  char te[3];
17  unsigned long Sweight;
18  bool rflag=1; //1 ---> read; 0 ---> write
19
20  void setup() {
```

```
21      Serial.begin(9600);
22      m1.initialize(8); //8needschanged
23      YL_Calibrated.setpin_SCKDT(4, 5);
24      YL_Calibrated.set_range(20);
25      YL_Calibrated.kb_Initialize();
26      oled.initialize();
27      rc522.initialize(9,10);
28    }
29
30    void loop() {
31      switch(rflag){
32        case 0:
33          while(1) {
34            bool state = rc522.write();
35            if(state==1)
36              break;
37          }
38          rflag=1;
39          break;
40        case 1:
41          while(1) {
42            bool state = rc522.read();
43            if(state==1)
44              break;
45          }
46          break;
47      }
48      for (int i=0; i<sizeof(rc522.Type_Name);i++) te[i]=rc522.Type_Name[i];
49      Sweight=rc522.Single_Weight;
50
51      numbefore = numnow;
52      unsigned long CalibratedWeight = YL_Calibrated.Output_CalibratedWeight(YL_Surface.
            Get_Surface());
53      numnow = ceil(CalibratedWeight/Sweight);
54
55      bool flag= (numbefore==numnow?0:1);
56      if(flag){
57      tf.initialize(te, m1.address, numnow, CalibratedWeight);
58      tf.pack();
59      digitalWrite(3,HIGH);
60      m1.send(9, tf.Transmission_Information);
61      Serial.println(tf.Transmission_Information);
62      oled.showIIC(te, numnow);
63      digitalWrite(3,LOW);
64      }
65
66      delay(3000);
67    }
```