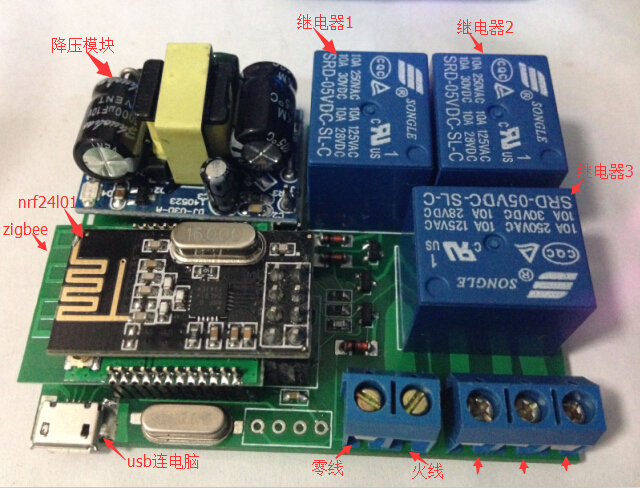
上篇[WIFI作品DIY教程07-《wifi家居网关》家居控制中心zigbee/nrf24l01等](http://www.znck007.com/forum.php?mod=viewthread&tid=24216)完成了家居网关。

这篇我们就教大家如何DIY智能开关，可以用来做电灯开关，也可以做插座开关，并这次是3路控制满足一般家庭使用了。

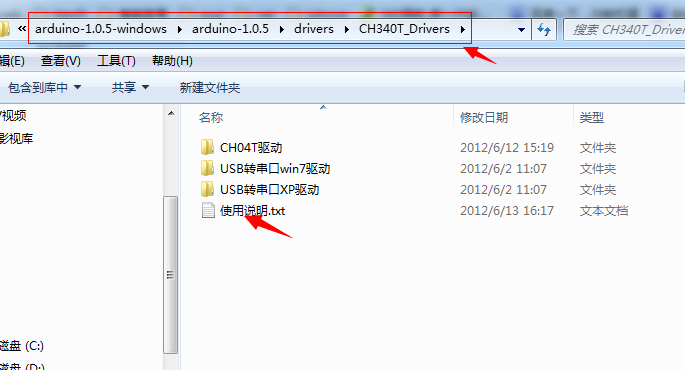


我们需要的配件：多路开关板、nrf24l01（或zigbee等）

**多路开关板的原理**：网关nrf24l01或zigbee发送{ck0030001}到多路开关板，收到字符后判断0001则是0路开，0000则是0路关，0011则是1路开，0010则是1路关，0021则是2路开，0020则是2路关。多路开关板使用arduino控制继电器间接控制220v电路。

第一部分：下载编译源代码

1、安装CH340驱动



在drivers目录里找到CH340驱动，然后按说明安装（之前安装了就不用再安装了）。

2、源代码

/\*

\*NRF24l01针脚连接线

\* MISO -> 12

\* MOSI -> 11

\* SCK -> 13

\* Configurable:

\* CE -> 8

\* CSN -> 7

\*/

#include <SPI.h>

#include <Mirf.h>

#include <nRF24L01.h>

#include <MirfHardwareSpiDriver.h>

#include <EEPROM.h>

int sid=3;//模块类型

int nid=0;//模块编号

//无线串口通信处理(zigbee/bluetooth等)

unsigned long serial1nowlast;

char serial1buff[129]={0};

char serial1Data;

int serial1i=0;

//NRF24l01

unsigned long nrf24l01nowlast;

char nrf24l01buff[33]={0};

char nrf24l01Data;

int nrf24l01i=0;

//声名变量

int pinIn0=3;

int pinIn1=4;

int pinIn2=5;

int val0;

int val1;

int val2;

int pinOut0=A0;

int pinOut1=A1;

int pinOut2=A2;

int state0=0;

int state1=0;

int state2=0;

void setup()

{

Serial.begin(115200);

char client[10]={0};//client

sprintf(client,"clie%d",sid);

//初始化Mirf，用于NRF24l01收发

Mirf\_Init(0,client,sid);

pinMode(pinIn0,INPUT);

pinMode(pinIn1,INPUT);

pinMode(pinIn2,INPUT);

pinMode(pinOut0,OUTPUT);

pinMode(pinOut1,OUTPUT);

pinMode(pinOut2,OUTPUT);

pinMode(pinIn0,INPUT\_PULLUP); //将管脚设置为输入并且内部上拉模式

pinMode(pinIn1,INPUT\_PULLUP); //将管脚设置为输入并且内部上拉模式

pinMode(pinIn2,INPUT\_PULLUP); //将管脚设置为输入并且内部上拉模式

digitalWrite(pinIn0, HIGH);

digitalWrite(pinIn1, HIGH);

digitalWrite(pinIn2, HIGH);

state0 = EEPROM.read(0);

state1 = EEPROM.read(1);

state2 = EEPROM.read(2);

Serial.println("zwifi\_kaiguan");

Serial.print("state0=");

Serial.print(state0);

Serial.print("/state1=");

Serial.print(state1);

Serial.print("/state2=");

Serial.println(state2);

if(state0==255) state0=0;

if(state1==255) state1=0;

if(state2==255) state2=0;

}

void loop()

{

//检测无线串口数据处理 (zigbee/bluetooth等)

{

unsigned long serial1now = millis();//获取现在的时间

if(serial1now - serial1nowlast >= 5000)//如果数据间隔超过5秒而清空字符（为了防止数据错乱）

{

serial1nowlast = millis();

memset(serial1buff, 0, 129);

serial1i=0;

}

while( Serial.available() )//如果无线串口有数据

{

if(serial1i==0)

{

Serial.println("serial->");//打印出来方便调试

}

serial1Data=(char)Serial.read();//读取串口数据

//Serial.print(serial1Data);////这里不打印，否则检测到{ckxxxx}就认为是命令

serial1buff[serial1i]=serial1Data;////保存到数组

serial1i++;////数组长度+1

if(serial1Data=='}' || serial1i>=129)//如果发现}而说明命令结束（并少于129个字符，太长会出错）

{

serial1nowlast = millis(); //更新当前时间，不然5秒就超时了

char body[129]={0};

get\_znck\_body(serial1buff,body);//获取只是{ckxxxxxx}的字符,因为这是我们的命令格式

//serial.println(body);

//如果命令格式真确则发送到无线串口

if(strstr(body,"{ck") && strstr(body,"}") )

{

//Serial.println(body);

if(strlen(body)>10)

{

set\_onoff(body);

}

}

serial1i=0;//字符长度为0

Serial.println("-------------------");

delay(100);

}

}

}

unsigned long nrf24l01now = millis();//获取现在的时间

if(nrf24l01now - nrf24l01nowlast >= 5000)//如果数据间隔超过5秒而清空字符（为了防止数据错乱）

{

nrf24l01nowlast = millis();

memset(nrf24l01buff, 0, 33);

nrf24l01i=0;

}

byte data[Mirf.payload];

if(Mirf.dataReady()){//!Mirf.isSending() &&

Mirf.getData(data);

Mirf.rxFifoEmpty(); //清理24L01援存

//Serial.println((char)\*data);

for (int i = 0; i < Mirf.payload; i++) //把收到的信息拼起来

{

if(nrf24l01i==0)

{

Serial.println("nrf24l01->");//打印出来方便调试

}

nrf24l01Data=(char)data[i];[/i]

if( nrf24l01Data=='{') nrf24l01i=0;

nrf24l01buff[nrf24l01i]=nrf24l01Data;//保存到数组

nrf24l01i++;////数组长度+1

if(nrf24l01Data=='}' || nrf24l01i>=33)//如果发现}而说明命令结束（并少于33个字符，太长会出错）

{

nrf24l01nowlast = millis(); //更新当前时间，不然5秒就超时了

char body[33]={0};

get\_znck\_body(nrf24l01buff,body);//获取只是{ckxxxxxx}的字符,因为这是我们的命令格式

//Serial.println(body);

//如果命令格式真确则发送到无线串口

if(strstr(body,"{ck") && strstr(body,"}") )

{

//Serial.println(body);

if(strlen(body)>10)

{

set\_onoff(body);

}

}

memset(nrf24l01buff, 0, 33);

nrf24l01i=0;//字符长度为0

Serial.println("-------------------");

delay(100);

}

}

}

val0=digitalRead(pinIn0);//读取数字

if(val0==LOW)//检测按键是否按下

{

Serial.print("val0 state0=");

Serial.println(state0);

if(state0==0)

{

digitalWrite(pinOut0,HIGH);

state0=1;

EEPROM.write(0, state0);

delay(2000);

}

else

{

digitalWrite(pinOut0,LOW);

state0=0;

EEPROM.write(0, state0);

delay(2000);

}

}

val1=digitalRead(pinIn1);//读取数字

if(val1==LOW)//检测按键是否按下

{

Serial.print("val1 state1=");

Serial.println(state1);

if(state1==0)

{

digitalWrite(pinOut1,HIGH);

state1=1;

EEPROM.write(1, state1);

delay(2000);

}

else

{

digitalWrite(pinOut1,LOW);

state1=0;

EEPROM.write(1, state1);

delay(2000);

}

}

val2=digitalRead(pinIn2);//读取数字

if(val2==LOW)//检测按键是否按下

{

Serial.print("val2 state2=");

Serial.println(state2);

if(state2==0)

{

digitalWrite(pinOut2,HIGH);

state2=1;

EEPROM.write(2, state2);

delay(2000);

}

else

{

digitalWrite(pinOut2,LOW);

state2=0;

EEPROM.write(2, state2);

delay(2000);

}

}

//ON OFF SET

if(state0==1)

{

digitalWrite(pinOut0,HIGH);

}

else

{

digitalWrite(pinOut0,LOW);

}

//ON OFF SET

if(state1==1)

{

digitalWrite(pinOut1,HIGH);

}

else

{

digitalWrite(pinOut1,LOW);

}

//ON OFF SET

if(state2==1)

{

digitalWrite(pinOut2,HIGH);

}

else

{

digitalWrite(pinOut2,LOW);

}

}

//初始化Mirf 0初始化1为接收2为发送

void Mirf\_Init(int txrx,char \*server,int channel){

//初始化Mirf，用于NRF24l01收发

if(txrx==0) {

Mirf.spi = &MirfHardwareSpi;

Mirf.init();

Mirf.setRADDR((byte \*)server);//设置接收地址

}

if(txrx==1) {

Mirf.setRADDR((byte \*)server);//设置接收地址

}

if(txrx==2) {

Mirf.setTADDR((byte \*)server);//设置发送地址

}

Mirf.payload = sizeof(char);//收发字节

Mirf.channel = channel;

Mirf.config();

}

//NRF24l01发送函数

void Mirf\_Send(int channel,char \*server,char \*str){

Mirf\_Init(2,server,channel);

int bufi=0;

for(bufi=0;bufi<strlen(str);bufi++){//循环发送

char words=str[bufi];//发送的字符

Mirf.send((byte \*)&words);//发送命令

while(Mirf.isSending()){//等待发送完闭

}

delay(50);//延时，否则可能出现发送丢失现象

//Serial.print(words);

}

//Serial.println("");

}

//获取只是{ckxxxxxx}的字符,因为这是我们的命令格式

void get\_znck\_body(char \*p,char \*s){

char rechar[33]={0};

int bufi=0;

bool isend=false;

int charnum=0;

for(bufi=0;bufi<strlen(p);bufi++){

//Serial.print(p[bufi]);

if(p[bufi]=='{'){

isend=true;

}

if(p[bufi]=='}' && isend==true){

isend=false;

rechar[charnum]=p[bufi];

break;

}

if(isend){

if(charnum<33)

{

rechar[charnum]=p[bufi];//Serial.print(rechar[charnum]);

charnum++;

}

}

}

//Serial.println("");

//memcpy(s,rechar,33);

sprintf(s,"%s",rechar);

}

//

void set\_onoff(char \*body){

int s=get\_sid(body);

int n=get\_nid(body);

int d=get\_data(body);

if( s==sid && n==0 )

{

if( d==1 )

{

state0=1;

EEPROM.write(0, state0);

}

if( d==0 )

{

state0=0;

EEPROM.write(0, state0);

}

}

if( s==sid && n==1 )

{

if( d==1 )

{

state1=1;

EEPROM.write(1, state1);

}

if( d==0 )

{

state1=0;

EEPROM.write(1, state1);

}

}

if( s==sid && n==2 )

{

if( d==1 )

{

state2=1;

EEPROM.write(2, state2);

}

if( d==0 )

{

state2=0;

EEPROM.write(2, state2);

}

}

if( s==sid ){

char server[10]={0};//server

sprintf(server,"serv%d",1);

//Serial.println(server);

char updateData[33]={0};

char front[10]={0};

//memcpy(front,body,9);

sprintf(front," {ck%03d%03d",s,n);

sprintf(updateData,"%supdate}",front);

Serial.println(updateData);

Mirf\_Send(1,server,updateData);

char client[10]={0};//client

sprintf(client,"clie%d",sid);

Mirf\_Init(1,client,sid);

}

}

int get\_sid(char \*buff){

if( strstr(buff,"{ck") && strstr(buff,"}") && strlen(buff)>10)

{

char charSid[4]={0};

memcpy(charSid,buff+3,3);

Serial.println(charSid);

int intSid=atoi(charSid);

Serial.println(intSid);

return intSid;

}

else

{

return 0;

}

}

int get\_nid(char \*buff){

if( strstr(buff,"{ck") && strstr(buff,"}") && strlen(buff)>10)

{

char charNid[4]={0};

memcpy(charNid,buff+6,3);

Serial.println(charNid);

int intNid=atoi(charNid);

Serial.println(intNid);

return intNid;

}

else

{

return 0;

}

}

int get\_data(char \*buff){

if( strstr(buff,"{ck") && strstr(buff,"}") && strlen(buff)>10)

{

char charData[4]={0};

memcpy(charData,buff+9,1);

Serial.println(charData);

int intData=atoi(charData);

Serial.println(intData);

return intData;

}

else

{

return 0;

}

}

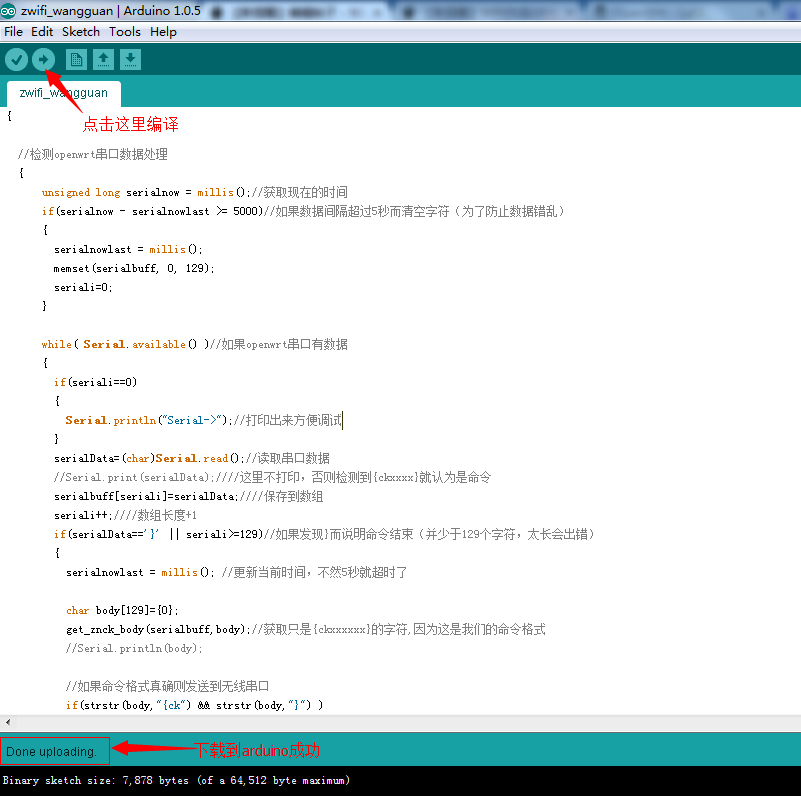
值得注意的是，我们采用了EEPROM的3个字节做为记录上次的状态，就算断电再供电，还会是原来的开或闭状态！

刷写arduino程序选哪个板？

请选择板是Arduino Uno，并记得选对对应的串口。

刷写ardruino源代码时，提示库不存在？

请联系qq610854837下载zwifi资料，软件下载目录里的arduino-1.0.5，里面包含了教程使用的所有库。



最后下载程序到arduino多路开关板。

第二部分：测试多路开关板通信

打开：<http://192.168.1.1/znck007/mysqltest.php?mode=select>



如图填写数据，然天点击添加！数据就会插入到数据库，网关会自动获取当前状态等于0，然后发送到003这个模块！



点击查看数据库记录，会看到你添加的记录。



当刷新发现status=1的话，那么说明网关和多路开关模块通信就正常了。

如果status不等于1，然后num等于6（num是超时计数），那么通信不正常，请用串口检查哪里出现问题！

同时也可以用数据库软件Navicat for MySQL直接操作！

