1.1上位机的进一步研究

主要功能函数

项目中选择串口作为主要的通信方式，上位机中选择中选用Qt5自带的QSerialPort 类作通信类，对于QSerialPort的使用，Qt中做了相应的规定



在使用QSerialPort时应该在工程文件中加入以下代码，保证编译连接库函数时系统不会报错

QT += core gui

QT += serialport

以下的代码为串口设置部分，每台电脑的串口号不是固定的，因此在代码中添加了串口的设置选项，为了避免其他选项出错，编写代码时对以下数据进行了固定，波特率、校验位、停止位和流控信息以便与STM32的代码进行同步处理，保证异步通信的准确性

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

warn\_flag = true;

curveDemo = new RealtimeDemo(this);

QGridLayout \*curveLayout = new QGridLayout(ui->widget);

curveLayout->addWidget(curveDemo);

serial = new QSerialPort(this);

readTimer = new QTimer(this);

readTimer->start(500);

serial->*open*(QIODevice::ReadWrite);

serial->startTimer(100);

foreach(const QSerialPortInfo &info,QSerialPortInfo::availablePorts())

{

QSerialPort serial;

serial.setPort(info);

if(serial.*open*(QIODevice::ReadOnly))

{

ui->comboBox->addItem(info.portName());

serial.*close*();

}

}

serial->setPortName(ui->comboBox->currentText());//portname

serial->setBaudRate(115200);//波特率

serial->setDataBits(QSerialPort::Data8);//数据位

serial->setParity(QSerialPort::NoParity);//校验位

serial->setStopBits(QSerialPort::OneStop);//停止位

serial->setFlowControl(QSerialPort::NoFlowControl);//控流信息

connect(readTimer,SIGNAL(timeout()),this,SLOT(serialReadSolots()));//定时对串口数据进行读写

connect(readTimer,SIGNAL(timeout()),this,SLOT(udpsSendDataSlots()));//定时对串口数据进行更新

}

以下为串口接收函数分析部分，串口通信采用接口协议，具体的接口协议为“F+[温度][湿度][灯光]+E”，通过协议分析可以准确的获取底部传感器采集到的数据信息，并在UI界面中进行展示

void MainWindow::serialReadSolots()

{

QByteArray readGream;

QString temp,hum;

readGream = serial->readAll();

for(int i = 0;i < readGream.length();i++) {

if(readGream[i] == 'F' && readGream[i+5] == 'E'){

//TEMP

temp = QString::number(readGream[i+1],10);

ui->temp\_label->setText(temp);

curveDemo->dataA = temp.toDouble();

//HUM

hum = QString::number(readGream[i+2],10);

ui->huncuo\_label->setText(hum);

curveDemo->dataB = hum.toDouble();

//CO

if(readGream[i+3]){

ui->ph\_label->setText(tr("正常"));

}else{

ui->ph\_label->setText(tr("警告"));

}

//LIGHT

if(readGream[i+4]){

ui->temp\_label\_2->setText(tr("正常"));

}else{

ui->temp\_label\_2->setText(tr("警告"));

}

}

}

Qt重要的特性就是信号与槽的连接，系统中所有的指令发送都是通过按钮的点击时间来出发的，然后通过串口发送相应的指令给你STM32，具体的槽代码如下：

void MainWindow::on\_pushButton\_clicked()

{

serial->setPortName(ui->comboBox->currentText());

serial->*open*(QIODevice::ReadWrite);

serial->startTimer(100);

ui->pushButton->setEnabled(false);

}

void MainWindow::on\_set\_warning\_btn\_clicked()

{

ui->temp\_lineEdit->setEnabled(false);

ui->tmd\_lineEdit->setEnabled(false);

warn\_flag = true;

}

void MainWindow::on\_change\_btn\_clicked()

{

ui->temp\_lineEdit->setEnabled(true);

ui->tmd\_lineEdit->setEnabled(true);

warn\_flag = false;

}

void MainWindow::on\_label\_3\_linkActivated(const QString &link)

{

}

1.2 keil代码初步研究

根据具体的接口协议为“F+[温度][湿度][灯光]+E”，初始化定时器，采集底层传感器数据信息

int main(void)

{

uint8\_t TEMP,RH,CO,LIGHT;

TIM1\_config();//初始化定时器

usart\_config();

CO\_config();

LIGHT\_config();

usart\_nvrcconfig();

while(1)

{

DHT11\_data(&RH,&TEMP);//初始化温度传感器

CO = GPIO\_ReadInputDataBit(CO\_PORT, CO\_PIN);

LIGHT = !(GPIO\_ReadInputDataBit(LIGHT\_PORT, LIGHT\_PIN));

usart\_send(USART1,'F');

usart\_send(USART1,TEMP);

usart\_send(USART1,RH);

usart\_send(USART1,CO);

usart\_send(USART1,LIGHT);

usart\_send(USART1,'E');

}

}