```
//此处使用bin文件，hex文件包含地址信息，需要自己提取代码信息

QVector<QByteArray> groupedData;
//打开文件并分包
void MainWindow::on_OpenFile_Btn_clicked()
{
    groupedData.clear();
    QString curDir = QCoreApplication::applicationDirPath();
    *fileName  = QFileDialog::getOpenFileName(this,"选择一个txt文
件",curDir,"*.bin");
    file->setFileName(*fileName);
    if(!file->open(QIODevice::ReadOnly))
    {
        QMessageBox::information(this,"提示","文件打开失败");
        return;
    }
    qint32 FileSize = file->size();
    ui->fileName_LEdit->setText(*fileName+"  "+QString::number(FileSize)+"byte");
    *fileData = file->readAll();
    if(fileData->size() != FileSize)
    {
        QMessageBox::information(this,"提示","文件读取失败");
        return;
    }

    const int groupSize = 128;
    for (int i = 0; i < FileSize; i += groupSize) {
        // 从 fileData 里截取 groupSize 大小的字节
        QByteArray chunk = fileData->mid(i, groupSize);

        // 如果截取的字节数不足 128，则进行填充
        if (chunk.size() < groupSize) {
            chunk.append(QByteArray(groupSize - chunk.size(), static_cast<char>
(0xFF)));
        }
        // 将分组后的数据添加到 QVector
        groupedData.append(chunk);
    }
    // 输出每组数据
    QString string;
    string = QString("文件分包成功，每组128个数据，共%1组").arg(groupedData.size());
    ui->SendFile_textEdit->append(string);//显示信息到文本框
}

//开始发送文件，发送起始段
void MainWindow::on_SenFile_Btn_clicked()
{
    QString byte;
    uint16_t crcValue;
    uint8_t b[137];
    state = sendFile;
    CurrentIndex = 0;  //数据段数复位

    if(!file->isOpen())
```

```
    {
        QMessageBox::warning(this,"提示","请选择一个bin文件");
        return;
    }
    qint32 FileSize = file->size();
    uint8_t cnt =0;
    if(ui->danbo_rBtn->isChecked())  //指定地址发送，地址从ui控件获得
    {
        //指定地址发送，有返回数据，返回正确数据后CurrentIndex++，返回错误或超时，
        //CurrentIndex不变，继续发送上一次数据
        start_addr = ui->start_spinBox->text().toInt();
        over_addr = ui->end_spinBox->text().toInt();
        current_addr = start_addr;
        b[cnt++]=current_addr;//485地址
    }
    else {
        //广播时，地址为0 ，需要定时发送数据包，建议每个数据包隔100ms以上
        b[cnt++]=0;//485地址
        time5->setInterval(300);
        time5->start();

    }
    b[cnt++]=0x10;                                  //功能位
    b[cnt++]=CurrentIndex>>8;                       //寄存器起始地址
    b[cnt++]=CurrentIndex&0xFF;
    b[cnt++]=0;                                    //寄存器个数
    b[cnt++]=1;
    b[cnt++]= 2;                                   //字节数
    b[cnt++] = FileSize>>8;
    b[cnt++] = FileSize&0xFF;
    crcValue= crc->crc16_modbus(b,cnt);
    b[cnt++] = crcValue&0x00FF;
    b[cnt++] =crcValue>>8;


    for(int i=0;i<cnt;i++)
    {
        byte.append(QString("%1").arg(b[i],2,16,QLatin1Char('0')).toUpper());
        byte.append(" ");
    }

    if(port==NULL || !port->mSerialPort.isOpen())
    {
        QMessageBox::information(this,"","串口未打开或不存在");
    }
    else
    {
        ui->SendFile_textEdit->append(byte);
        port->mSerialPort.write((const char*)(b),cnt);

    }
}

//发送后续文件（发送代码数据）
void MainWindow::sendNextFile()
{
```

```
    QString byte;
    uint16_t crcValue;
    uint8_t b[150] ={0};
    state = sendFile;

    uint8_t cnt =0;
    if(ui->danbo_rBtn->isChecked())
    {
        b[cnt++]=current_addr;//485地址
        //打开定时器4
        time4->setInterval(300);
        time4->start();
    }
    else {
        b[cnt++]=0;//485地址
        CurrentIndex++;
    }
    if(CurrentIndex>groupedData.size())
    {
        on_overSend_Btn_clicked();//发送结束报文
        return;
    }

    b[cnt++]=0x10;                              //功能位
    b[cnt++]=CurrentIndex>>8;                   //寄存器起始地址
    b[cnt++]=CurrentIndex&0xFF;
    b[cnt++]=0;                                //寄存器个数
    b[cnt++]=64;
    b[cnt++]= 128;                              //字节数
    QByteArray datatosend = groupedData[CurrentIndex-1];
    std::memcpy(&b[cnt],datatosend.constData(),128);
    cnt = cnt+128;
    crcValue= crc->crc16_modbus(b,cnt);
    b[cnt++] = crcValue&0x00FF;
    b[cnt++] =crcValue>>8;


    for(int i=0;i<cnt;i++)
    {
        byte.append(QString("%1").arg(b[i],2,16,QLatin1Char('0')).toUpper());
        byte.append(" ");
    }

    if(port==NULL || !port->mSerialPort.isOpen())
    {
        QMessageBox::information(this,"","串口未打开或不存在");
    }
    else
    {
        ui->SendFile_textEdit->append(byte);
        port->mSerialPort.write((const char*)(b),cnt);
        ui->progressBar->setValue(CurrentIndex*100/groupedData.size());
    }
}

//发送结束报文
```

```
void MainWindow::on_overSend_Btn_clicked()
{
    QString byte;
    uint16_t crcValue;
    uint8_t b[137];
    state = sendFile;

    ui->progressBar->setValue(0);
    uint8_t cnt =0;
    if(ui->danbo_rBtn->isChecked())
    {
        start_addr = ui->start_spinBox->text().toInt();
        over_addr = ui->end_spinBox->text().toInt();
        current_addr = start_addr;
        b[cnt++]=current_addr;//485地址
    }
    else {
        b[cnt++]=0;//485地址
 //       CurrentIndex++;
    }
    b[cnt++]=0x10;                                  //功能位
    b[cnt++]=0xFF;                                    //寄存器起始地址
    b[cnt++]=0xFF;
    b[cnt++]=0;                                    //寄存器个数
    b[cnt++]=1;
    b[cnt++]= 2;                                   //字节数
    b[cnt++] = CurrentIndex>>8;
    b[cnt++] = CurrentIndex&0xFF;
    crcValue= crc->crc16_modbus(b,cnt);
    b[cnt++] = crcValue&0x00FF;
    b[cnt++] =crcValue>>8;


    for(int i=0;i<cnt;i++)
    {
        byte.append(QString("%1").arg(b[i],2,16,QLatin1Char('0')).toUpper());
        byte.append(" ");
    }

    if(port==NULL || !port->mSerialPort.isOpen())
    {
        QMessageBox::information(this,"","串口未打开或不存在");
    }
    else
    {
        ui->SendFile_textEdit->append(byte);
        port->mSerialPort.write((const char*)(b),cnt);
        //停止定时器44
        time4->stop();
        time5->stop();

    }
}
```