| 类型 | 外发 |
| --- | --- |
| 日期 | 2018/1/22 |
| 版本 | Ver ： 01.10 |
| 部门 | 工程部 |

# 蓝牙+GPRS 版(AC)
# 共享单车锁（硬件）
# 接口协议

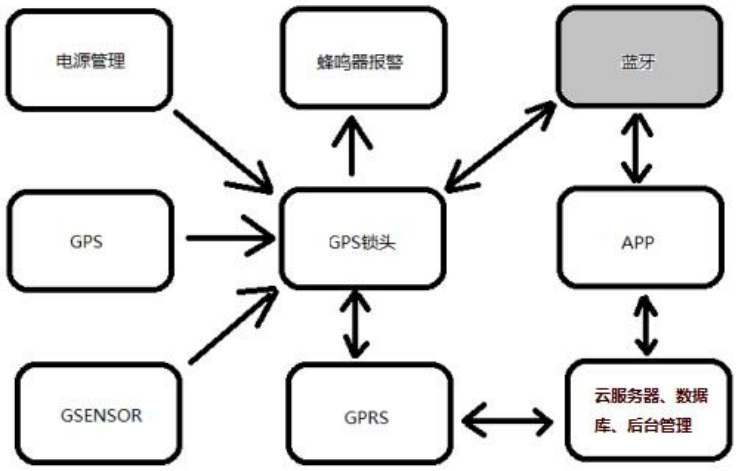
# 目录

# 系统总体框架及协议说明

## 1.1 编写目的

本文档协议用于描述本公司所设计生产的系列蓝牙锁和服务器或者 APP 之间的通信协议。

## 1.2 共享单车锁系统框图




## 1.3 通信包格式

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数,数据发送方产生，用于加密数据 |
| 2 | KEY | 通信秘钥，由车锁随机产生，APP 通过（0x11）命令获得 |
| 3 | CMD | 命令字 |
| 4 | LEN | 数据长度 |
| 5 | DATA | 数据 |
| 5+LEN | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 6+LEN | | |

## 1.4 数据加密过程

加密组成：随机数、KEY。
加密过程:
1、产生随机数 NUM
2、产生随机数变种 NUM_1 = NUM + 0x32
3、把 NUM_1 填充到数据的第 1 字节
4、用 NUM 分别异或（^）NUM 之后，CRC 之前的明文数据并把结果对应回填
5、把 CRC 之前数据做 CRC16 校验，校验值填到 CRC 位置。

见附录一

## 1.5 APP 与车锁的通信流程

1. APP 与车锁建立蓝牙连接
2. APP 向车锁发送（0x11）指令获取通信秘钥 KEY
3. 车锁返回通信秘钥 KEY, APP 需保存秘钥，用于后续通信
4. APP 与车锁进行通信

**注：秘钥 KEY 只在 APP 与车锁建立蓝牙连接时重新获取，此后通信保持不变**

## 1.6 使用的 UUID

Service UUID :0783b03e-8535-b5a0-7140-a304d2495cb7
该 service 下的 characteristic

| characteristic UUID | 操作类型 | 说明 |
|---|---|---|
| 0783b03e-8535-b5a0-7140-a304d2495cba | Write | 向硬件写指令 |
| 0783b03e-8535-b5a0-7140-a304d2495cb8 | Notify | 硬件返回的信息 |

注册通知时，需要用到的 characteristic 下 descriptor 的 UUID:
00002902-0000-1000-8000-00805f9b34fb

# 1.7 命令详述及示例

## 1.7.1 获取通信 KEY 指令(0x11)

### 1.7.1.1 APP->车锁

App 连接到蓝牙设备，首先要获取到与蓝牙设备通信的 KEY

DATA 中的设备识别 KEY，是蓝牙设备唯一识别的 ID，当连接上蓝牙设备在 5 秒内没有发送获取通信 KEY 指令，或者设备识别 KEY 发送错误，蓝牙设备会自动断开与 app 的连接。

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 0x00 |
| 3 | CMD | 0x11 |
| 4 | LEN | 0x08 |
| 5-12 | DATA | 设备识别 KEY，8 字节（默认为"yOTmK50z"） |
| 13 | CRC | CRC 之前数据经过加密后的 CRC16 校验值(0-12) |
| 14 | | |

### 1.7.1.2 车锁->APP

设备收到获取通信 KEY 指令后，在 DATA 中返回用于通信的 KEY

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x11 |
| 4 | LEN | 0x01 |
| 5 | DATA | 秘钥(KEY),用于通信的 KEY |
| 6 | CRC | CRC 之前数据经过加密后的 CRC16 校验值(0-5) |
| 7 | | |

## 1.7.2 开锁指令(0x21)

### 1.7.2.1 APP->车锁

开锁时 DATA 中传入用户 ID（4 字节）和开锁时的时间戳（4 字节）。

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x21 |
| 4 | LEN | 0x09 |
| 5-8 | DATA | APP 端用户 ID 号 |
| 9-12 | DATA | 开锁时间戳 4 字节，高位在前 |
| 13 | DATA | 开锁类型(0:围栏内开锁，1：围栏外开锁) |
| 14 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 15 | | |

### 1.7.2.2 车锁->APP

设备执行开锁动作后，会在 DATA 中返回开锁状态

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x21 |
| 4 | LEN | 0x05 |
| 5 | DATA | 返回值 0:开锁成功 1:开锁失败 |
| 6-9 | DATA | 开锁时间戳 4 字节，高位在前 |
| 10 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 11 | | |

### 1.7.2.3 APP->车锁

APP 收到锁发出的开锁指令状态指令后，回复锁

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x21 |
| 4 | LEN | 0x00 |
| 6 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 7 | | |

## 1.7.3 上锁指令(0x22)

### 1.7.3.1 车锁->APP

设备上锁操作后，在 DATA 上传上传状态和骑行时间。

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x22 |
| 4 | LEN | 0x09 |
| 5 | DATA | 0:上锁成功 1:上锁失败 |
| 6-9 | DATA | 开锁时间戳 |
| 10-13 | DATA | 骑行时间 4 字节(单位:分钟) |
| 14 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 15 | | |

### 1.7.3.2 APP->车锁

App 接受到上锁指令后回复，不回复则会产生旧数据

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x22 |
| 4 | LEN | 0x00 |
| 5 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 6 | | |

## 1.7.4 查询锁状态(0x31)

### 1.7.4.1 APP->车锁

可以获取到锁开关状态，电池电量，是否有旧数据等信息。

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x31 |
| 4 | LEN | 0x00 |
| 5 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 6 | | |

### 1.7.4.2 车锁->APP

设备接受到查询状态后，会在 DATA 中返回当前开关状态，电池电量，是否有旧数据等信息。

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x31 |
| 4 | LEN | 0x07 |
| 5 | DATA | 锁状态 0：开锁 1：关锁 |
| 6 | | 电池电量 例：37（十进制） 为 3.7V |
| 7 | | 是否有未上传数据 0：有 1：没有 |
| 8-11 | | 时间戳 4 字节 |
| 12 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 13 | | |

## 1.7.5 获取未上传数据(0x51)

注：此数据为用户关锁后未上传到 APP 或服务器的数据，包含用户 ID 及用车时间，用于结费。

### 1.7.5.1 APP->车锁

App 在获取设置状态时，如果发现有旧数据，可以获取到旧数据，上传至服务器。

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x51 |
| 4 | LEN | 0x00 |
| 5 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 6 | | |

### 1.7.5.2 车锁->APP

旧数据，即上次骑行结束后未上传到服务器的数据，包括开锁时间戳，骑行时间，用户 ID

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x51 |
| 4 | LEN | 0x012 |
| 5-8 | DATA | 用户开锁时使用的时间戳 |
| 9-12 | DATA | 用户骑行时间 单位：分钟 |
| 13-16 | DATA | 用户 ID |
| 17 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 18 | | |

## 1.7.6 清除车锁中的未上传数据(0x52)

### 1.7.6.1 APP->车锁

App 将旧数据上传至服务器后，可以清除设备中存储的旧数据

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x52 |
| 4 | LEN | 0x00 |
| 5 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 6 | | |

### 1.7.6.2 车锁->APP

设备在 DATA 中返回旧数据的清除状态

| 字节 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：0xFE |
| 1 | NUM | 随机数 |
| 2 | KEY | 通信秘钥 |
| 3 | CMD | 0x52 |
| 4 | LEN | 0x01 |
| 5 | DATA | 返回值： 0：成功 1：失败 |
| 6 | CRC | CRC 之前数据经过加密后的 CRC16 校验值 |
| 7 | | |

# 2.0 TCP 通信协议说明

## 2.1 TCP 指令列表

1 开锁指令 (L0)　　2 关锁指令 (L1)
3 定位指令 (D0)　　4 签到指令 (Q0)
5 心跳指令 (H0)　　6 获取锁信息指令 (S5)
7 响铃指令 (S8)
8 固件版本指令 (G0)
9 报警指令 (W0)
10 获取 SIM 卡 ICCID 号指令(I0)
11 获取蓝牙设备 MAC 地址指令(M0)
12 设置设备 KEY 指令(K0)
13 自动上报固件版本(U0)

## 2.1.1 开锁指令 (L0)

### 2.1.1.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：, |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：, |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：, |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：, |
| 8 | CMD | 命令代码:L0 |
| 9 | SPACER | 间隔符：, |
| 10 | VALU | 开锁(0) |
| 11 | SPACER | 间隔符：, |
| 12 | VALU | 用户 ID |
| 13 | SPACER | 间隔符：, |
| 14 | VALU | 开锁时间戳(秒的时间戳，10 位) |
| 15 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,170619195455,L0,0,20,1497689816#<LF>

### 2.1.1.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:L0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 开锁(0)/关锁(1) |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 用户 ID |
| 13 | SPACER | 间隔符：， |
| 14 | VALU | 开锁时间戳(秒的时间戳，10 位) |
| 15 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,OM,863725031194523,001497689816,L0,0,20,1497689816#<LF>

### 2.1.1.3 服务器应答

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:Re |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 服务器回应对应的指令代码 L0 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,001497689816,Re,L0#<LF>

## 2.1.2 关锁指令 (L1)

### 2.1.2.1 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:L1 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 用户 ID，4 字节，此处放字符串形式 |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 开锁时间戳 |
| 13 | SPACER | 间隔符：， |
| 14 | VALU | 骑行时间，4 字节数据，此处放字符串形式（单位:分钟） |
| 15 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,OM,863158022988725,000000000000,L1,1,1497689816,20#<LF>

### 2.1.2.2 服务器应答

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:Re |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 服务器回应对应的指令代码 L1 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,000000000000,Re,L1#<LF>

## 2.1.3 定位指令 (D0)

### 2.1.3.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:D0 |
| 9 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,000000000000,D0#<LF>

### 2.1.3.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:D0 |
| 9 | VALU | 预留位置 |
| 10 | SPACER | 间隔符：， |
| 11 | VALU | GPS 定位数据： |
| 12 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：

*CMDR,863725031194523,000000000000,

D0,0,124458.00,A,2237.75314,N,11408.62621,E,0.066,,151216,,,A#<LF>

说明：

0,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>,< 10>,<11>,<12>#

<1> UTC 时间，hhmmss(时分秒)格式

<2> 定位状态，A=有效定位，V=无效定位

<3> 纬度 ddmm.mmmm(度分)格式(前面的 0 也将被传输)

<4> 纬度半球 N(北半球)或 S(南半球)

<5> 经度 dddmm.mmmm(度分)格式(前面的 0 也将被传输)

<6> 经度半球 E(东经)或 W(西经)

<7> 地面速率(000.0~999.9 节，前面的 0 也将被传输)

<8> 地面航向(000.0~359.9 度，以真北为参考基准，前面的 0 也将被传输)

<9> UTC 日期，ddmmyy(日月年)格式

<10> 磁偏角(000.0~180.0 度，前面的 0 也将被传输)

<11> 磁偏角方向，E(东)或 W(西)

<12> 模式指示(A=自主定位，D=差分，E=估算，N=数据无效)

**注：6.0 版本后，7-卫星个数，8-HDOP，10-天线高程 ，11-单位 M**

**注：会出现如下无效定位格式**

*CMDR,863725031194523,000000000000, D0,**0,033724.00,V,,,,,,120517,,,N#**

*CMDR,863725031194523,000000000000,D0,

**0,062102.362,V,,,,0.06,277.72,120517,,,N#**

## 2.1.3.2 服务器应答

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:Re |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 服务器回应对应的指令代码 D0 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,000000000000,Re,D0#<LF>

## 2.1.4 签到指令（Q0)

### 2.1.4.1 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:Q0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 电量 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,OM,863725031194523,000000000000,Q0,410#<LF>

## 2.1.5 心跳指令（H0)

### 2.1.5.1 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:H0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 锁状态(0-开，1-关锁) |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 电量 |
| 13 | SPACER | 间隔符：， |
| 14 | VALU | GSM 信号值 |
| 15 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,OM,863158022988725,161201150000,H0,1,400,24#<LF>

## 2.1.6 获取锁信息指令 (S5)

### 2.1.6.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF, 0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 锁当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:S5 |
| 9 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,000000000000,S5#<LF>

### 2.1.6.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 锁当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:S5 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 电池电量 |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | gsm 信号值 |
| 13 | SPACER | 间隔符：， |
| 14 | VALU | 预留参数 |
| 15 | SPACER | 间隔符：， |
| 16 | VALU | 当前锁状态(0:开锁，1：关锁) |
| 17 | SPACER | 间隔符：， |
| 18 | VALU | 预留参数 |
| 19 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,XX,863158022988725,000000000000,S5,410,31,00,1,0#<LF>

## 2.1.7 寻车响铃指令（S8)

### 2.1.7.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF, 0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 锁当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:S8 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 响的秒数 |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 预留参数 |
| 13 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,OM,863725031194523,000000000000,S8,5,0#<LF>

### 2.1.7.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:S8 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 响的秒数 |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 预留参数 |
| 13 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,XX,863158022988725,000000000000,S8,05,0#<LF>

## 2.1.8 固件版本指令 (G0)

### 2.1.8.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:G0 |
| 9 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,XX,863158022988725,000000000000,G0#<LF>

### 2.1.8.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:G0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 版本号 |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 编译时间 |
| 13 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,XX,863158022988725,000000000000,G0,V1.0,May 17 2017#<LF>

## 2.1.9 报警指令 (W0)

### 2.1.9.1 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:W0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 报警状态 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,XX,863158022988725,000000000000,W0,1#<LF>

### 2.1.9.2 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:Re |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | W0 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,XX,863158022988725,000000000000,Re,W0#<LF>

# 2.1.10 获取 SIM 卡 ICCID 号指令 (I0)

## 2.1.10.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:I0 |
| 9 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,XX,863158022988725,000000000000,I0#<LF>

## 2.1.10.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:I0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | ICCID 号 |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,XX,863158022988725,000000000000,I0,800602B8001050000281#<LF>

## 2.1.11 获取 MAC 地址指令 (M0)

该指令在固件版本大于 2.7 时有效

### 2.1.11.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:M0 |
| 9 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,XX,863158022988725,000000000000,M0#<LF>

### 2.1.11.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:M0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 设备蓝牙 MAC 地址 |
| 11 | END | 命令结束符:#<LF> |

示例：*CMDR,XX,863158022988725,000000000000,M0,80:8E:00:01:00:0A#<LF>

# 2.1.12 设置设备 KEY 指令 (K0)

## 2.1.12.1 服务器发给锁

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF,0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：, |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：, |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：, |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：, |
| 8 | CMD | 命令代码:K0 |
| 9 | SPACER | 间隔符：, |
| 10 | VALU | 读写标志,1[设置],0[读取] |
| 11 | SPACER | 间隔符：, |
| 12 | VALU | 设置的 KEY[当读写标志位是 0，填充 0] |
| 13 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,XX,863158022988725,000000000000,K0,1,newkey21#<LF>

*CMDS,XX,863158022988725,000000000000,K0,0,0#<LF>

## 2.1.12.2 锁发给服务器

| 序号 | 项 | 说明 |
|---|---|---|
| 0 | STX | 数据头/帧头 固定值：*CMDR |
| 1 | SPACER | 间隔符：, |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：, |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：, |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：, |
| 8 | CMD | 命令代码:K0 |
| 9 | SPACER | 间隔符：, |
| 10 | VALU | 设备 KEY |
| 11 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDR,XX,863158022988725,000000000000,K0,newkey21#<LF>

## 2.1.13 自动上报当前版本（U0)

### 2.1.13.1 锁发给服务器

锁会每隔 1 天向服务器发送一次当前版本信息。

| 序号 | 项 | 说明 |
|---|---|---|
| | HEX | 两个字节的保留位：0xFF, 0xFF |
| 0 | STX | 数据头/帧头 固定值：*CMDS |
| 1 | SPACER | 间隔符：， |
| 2 | CODE | 设备代码 |
| 3 | SPACER | 间隔符：， |
| 4 | IMEI | 锁的 IMEI 号 |
| 5 | SPACER | 间隔符：， |
| 6 | TIME | 当前时间：年月日时分秒(保留数据)，目前是填充 000000000000 |
| 7 | SPACER | 间隔符：， |
| 8 | CMD | 命令代码:U0 |
| 9 | SPACER | 间隔符：， |
| 10 | VALU | 版本号 |
| 11 | SPACER | 间隔符：， |
| 12 | VALU | 设备类型 |
| 13 | SPACER | 间隔符：， |
| 14 | VALU | 版本编译时间 |
| 15 | END | 命令结束符:#<LF>（<LF> is “\n” in java code ） |

示例：*CMDS,XX,863158022988725,000000000000,U0,15,A1,May 4 2017#<LF>

备注：

* 本协议是根据用户产品功能需求定制；
* 本协议部分细节可能会需根据系统实际运营场景调整；

# 附录一：蓝牙加密，解密过程

1.加密：以 app 向锁获取操作 KEY，0x11 指令为例。

| item | index | hex(original) | hex(+0x32) | hex(xor 34) | calc CRC |
|---|---|---|---|---|---|
| data2 | 0 | **FE** | FE | FE | FE |
| num | 1 | **34** | 66 | 66 | 66 |
| key | 2 | **0** | 0 | 34 | 34 |
| cmd | 3 | **11** | 11 | 25 | 25 |
| len | 4 | **8** | 8 | 3C | 3C |
| data | 5 | **79** | 79 | 4D | 4D |
| data | 6 | **4F** | 4F | 7B | 7B |
| data | 7 | **54** | 54 | 60 | 60 |
| data | 8 | **6D** | 6D | 59 | 59 |
| data | 9 | **4B** | 4B | 7F | 7F |
| data | 10 | **35** | 35 | 1 | 1 |
| data | 11 | **30** | 30 | 4 | 4 |
| data | 12 | **7A** | 7A | 4E | 4E |
| crc | 13 | | | | B4 |
| crc | 14 | | | | F6 |

2.解密：以锁向 app 返回操作的 KEY 为例

| item | index | hex | step1 | step2 | step3 | step4 | step5 | step6 |
|---|---|---|---|---|---|---|---|---|
| stx | 0 | **FE** | FE | **FE** | | FE | FE | FE |
| num | 1 | **3C** | | | | 3C | 3C | A(3C-32) |
| key | 2 | **F5** | | | | F5 | F5 | FF(F5^A) |
| cmd | 3 | **1B** | | | | 1B | 1B | 11(1B^A) |
| len | 4 | **0B** | | **0B** | 1(B^(3C-32)) | 0B | 0B | 1(B^A) |
| data (key) | 5 | **F5** | | | | F5 | F5 | FF(F5^A) |
| crc | 6 | **7C** | | | | 7C | 7C | |
| crc | 7 | **C1** | | | | C1 | C1 | |
| ? | 8 | **0(IF HAVE)** | | | | | | |
| ? | 9 | **A(IF HAVE)** | | | | | | |
| ? | 10 | **B(IF HAVE)** | | | | | | |
| ? | 11 | **C(IF HAVE)** | | | | | | |
| ? | 12 | **D(IF HAVE)** | | | | | | |
| ? | 13 | **E(IF HAVE)** | | | | | | |

Step1:找到 FE 所在下标地址

Step2:找到 len 所在位置

Step3:解密 len 的值，加密值异或处理后的随机数即 1=0x0B^(0x3C-0x32)

Step4:根据指令的起始位置和长度，获取到所以指令

Step5:检测 CRC 值

Step6:解析指令，获取需要的值。

# 附录二: CRC16 计算代码

```
public class CRCUtil {
    static byte[] t_crc16_h={
        0x00, (byte) 0xC1, (byte) 0x81, 0x40, 0x01, (byte) 0xC0, (byte) 0x80, 0x41, 0x01, (byte) 0xC0,
        (byte) 0x80, 0x41, 0x00, (byte) 0xC1, (byte) 0x81, 0x40, 0x01, (byte) 0xC0, (byte) 0x80, 0x41,
        0x00, (byte) 0xC1, (byte) 0x81, 0x40, 0x00, (byte) 0xC1, (byte) 0x81, 0x40, 0x01, (byte) 0xC0,
        (byte) 0x80, 0x41, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40,
        0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01,(byte) 0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1,
        (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x01, (byte)0xC0, (byte)0x80, 0x41,
        0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1,
        (byte)0x81, 0x40, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41,
        0x00,(byte) 0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x01, (byte)0xC0,
        (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x00, (byte)0xC1, (byte)0x81, 0x40,
        0x01,(byte) 0xC0,(byte) 0x80, 0x41, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1,
        (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40,
        0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x01, (byte)0xC0,
        (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x00, (byte)0xC1, (byte)0x81, 0x40,
        0x01,(byte) 0xC0, (byte)0x80, 0x41, 0x00,(byte) 0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0,
        (byte)0x80, 0x41, 0x01,(byte) 0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40,
        0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x01, (byte)0xC0,
        (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41,
        0x00,(byte) 0xC1, (byte)0x81, 0x40, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0,
        (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41,
        0x01, (byte)0xC0, (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0,
        (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x00, (byte)0xC1, (byte)0x81, 0x40,
        0x01, (byte)0xC0, (byte)0x80, 0x41, 0x01, (byte) 0xC0, (byte) 0x80, 0x41, 0x00, (byte) 0xC1,
        (byte)0x81, 0x40, 0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41,
        0x00, (byte)0xC1, (byte)0x81, 0x40, 0x01, (byte)0xC0, (byte)0x80, 0x41, 0x01, (byte)0xC0,
```

```
        (byte)0x80, 0x41, 0x00, (byte)0xC1, (byte)0x81, 0x40
};

static byte[] t_crc16_l={
        0x00, (byte) 0xC0, (byte) 0xC1, 0x01, (byte) 0xC3, 0x03, 0x02, (byte) 0xC2, (byte) 0xC6, 0x06,
        0x07, (byte) 0xC7, 0x05, (byte) 0xC5, (byte) 0xC4, 0x04, (byte) 0xCC, 0x0C, 0x0D, (byte) 0xCD,
        0x0F, (byte) 0xCF, (byte) 0xCE, 0x0E, 0x0A, (byte) 0xCA, (byte) 0xCB, 0x0B, (byte) 0xC9, 0x09,
        0x08, (byte) 0xC8, (byte) 0xD8, 0x18, 0x19, (byte) 0xD9, 0x1B, (byte) 0xDB, (byte) 0xDA, 0x1A,
        0x1E, (byte) 0xDE, (byte) 0xDF, 0x1F, (byte) 0xDD, 0x1D, 0x1C, (byte) 0xDC, 0x14, (byte) 0xD4,
        (byte) 0xD5, 0x15, (byte) 0xD7, 0x17, 0x16, (byte) 0xD6, (byte) 0xD2, 0x12, 0x13, (byte) 0xD3,
        0x11, (byte) 0xD1, (byte) 0xD0, 0x10, (byte) 0xF0, 0x30, 0x31, (byte) 0xF1, 0x33, (byte) 0xF3,
        (byte) 0xF2, 0x32, 0x36, (byte) 0xF6, (byte) 0xF7, 0x37, (byte) 0xF5, 0x35, 0x34, (byte) 0xF4,
        0x3C, (byte) 0xFC, (byte) 0xFD, 0x3D, (byte) 0xFF, 0x3F, 0x3E, (byte) 0xFE, (byte) 0xFA, 0x3A,
        0x3B, (byte) 0xFB, 0x39, (byte) 0xF9, (byte) 0xF8, 0x38, 0x28, (byte) 0xE8, (byte) 0xE9, 0x29,
        (byte) 0xEB, 0x2B, 0x2A, (byte) 0xEA, (byte) 0xEE, 0x2E, 0x2F, (byte) 0xEF, 0x2D, (byte) 0xED,
        (byte) 0xEC, 0x2C, (byte) 0xE4, 0x24, 0x25, (byte) 0xE5, 0x27, (byte) 0xE7, (byte) 0xE6, 0x26,
        0x22, (byte) 0xE2, (byte) 0xE3, 0x23, (byte) 0xE1, 0x21, 0x20, (byte) 0xE0, (byte) 0xA0, 0x60,
        0x61, (byte) 0xA1, 0x63, (byte) 0xA3, (byte) 0xA2, 0x62, 0x66, (byte) 0xA6, (byte) 0xA7, 0x67,
        (byte) 0xA5, 0x65, 0x64, (byte) 0xA4, 0x6C, (byte) 0xAC, (byte) 0xAD, 0x6D, (byte) 0xAF, 0x6F,
        0x6E, (byte) 0xAE, (byte) 0xAA, 0x6A, 0x6B, (byte) 0xAB, 0x69, (byte) 0xA9, (byte) 0xA8, 0x68,
        0x78, (byte) 0xB8, (byte) 0xB9, 0x79, (byte) 0xBB, 0x7B, 0x7A, (byte) 0xBA, (byte) 0xBE, 0x7E,
        0x7F, (byte) 0xBF, 0x7D, (byte) 0xBD, (byte) 0xBC, 0x7C, (byte) 0xB4, 0x74, 0x75, (byte) 0xB5,
        0x77, (byte) 0xB7, (byte) 0xB6, 0x76, 0x72, (byte) 0xB2, (byte) 0xB3, 0x73, (byte) 0xB1, 0x71,
        0x70, (byte) 0xB0, 0x50, (byte) 0x90, (byte) 0x91, 0x51, (byte) 0x93, 0x53, 0x52, (byte) 0x92,
        (byte) 0x96, 0x56, 0x57, (byte) 0x97, 0x55, (byte) 0x95, (byte) 0x94, 0x54, (byte) 0x9C, 0x5C,
        0x5D, (byte) 0x9D, 0x5F, (byte) 0x9F, (byte) 0x9E, 0x5E, 0x5A, (byte) 0x9A, (byte) 0x9B, 0x5B,
        (byte) 0x99, 0x59, 0x58, (byte) 0x98, (byte) 0x88, 0x48, 0x49, (byte) 0x89, 0x4B, (byte) 0x8B,
        (byte) 0x8A, 0x4A, 0x4E, (byte) 0x8E, (byte) 0x8F, 0x4F, (byte) 0x8D, 0x4D, 0x4C, (byte) 0x8C,
        0x44, (byte) 0x84, (byte) 0x85, 0x45, (byte) 0x87, 0x47, 0x46, (byte) 0x86, (byte) 0x82, 0x42,
        0x43, (byte) 0x83, 0x41, (byte) 0x81, (byte) 0x80, 0x40
};


public static int calcCRC(byte[] data){
        return calcCRC(data,0,data.length);
}

public static int calcCRC(byte[] data,int offset,int len){
        return calcCRC(data,offset,len,0xffff);
}
public static int calcCRC(byte[] data,int offset,int len,int preval){
        int ucCRCHi =(preval & 0xff00)>>8;
```

```
            int ucCRCLo = preval& 0x00FF;
            int iIndex;
            for(int i=0;i<len;i++){
                iIndex = (ucCRCLo^data[offset+i]) &0x00FF;
                ucCRCLo = ucCRCHi^t_crc16_h[iIndex];
                ucCRCHi = t_crc16_l[iIndex];
            }
            return ((ucCRCHi&0x00FF)<<8)|(ucCRCLo&0x00FF) &0xFFFF;
        }
}
```