上海道路综合设备箱

通信协议 第 2 部分：

综合设备箱监控终端与电子锁通信

目　　录

前　　言

本部分是根据《上海市道路综合设备箱技术要求》中对综合设备箱电子锁的功能要求编制的通信协议。

本部分规定了道路综合设备箱中监控终端与电子锁之间进行数据传输的帧格式、数据编码及传输规则。

本通信协议适用于上海架空线入地合杆整治工程，上海市其他道路合杆工程可参照执行。

上海市道路综合设备箱
通信协议 第 2 部分：
综合设备箱监控终端与电子锁通信

1 范围

本协议规定了道路综合设备箱监控管理单元中的监控终端（以下简称“终端”）与综合设备箱的电子门锁之间进行数据传输的帧格式、数据编码及传输规则。

本协议适用于点对点及一点对多点的通信方式，适用于监控系统对终端执行主从问答方式以及终端主动上传方式的通信。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 19582.1-2008 基于Modbus协议的工业自动化网络规范 第1部分：Modbus应用协议

3 术语和定义

3.1 综合设备箱监控终端

综合设备箱监控终端是安装在综合设备箱内，负责综合设备箱及各用户舱的用电信息、综合设备箱内环境信息的采集、数据管理、数据双向传输以及执行平台下发的控制命令的设备，以下简称终端。

3.2 综合设备箱电子锁

综合设备箱电子锁是安装在综合设备箱箱门上的电子锁，支持管理平台通过综合设备箱监控终端远程开锁，支持APP程序蓝牙开锁，支持电子钥匙本地开锁。

3.3 电子锁设备号（PA）

电子锁设备号（PA）是由电子锁生产厂商在出厂前烧录在电子锁内的设备的唯一编号。电子锁设备号可由管理单位分配给电子锁生产厂商，也可以由生产厂商编制，但必须保证其唯一性。综合设备箱供货时必须提供各箱门配备的电子锁的设备号。

3.4 电子锁标识号（ID）

电子锁标识号（ID）是电子锁在综合杆工程管理信息平台中的唯一编号。电子锁标识号由管理单位根据管理需求制定的规则分配给各个电子锁，可在综合设备箱接入平台后设置到各个电子锁。

4 帧结构

4.1 字节格式

帧的基本单元为8位字节。链路层传输顺序为低位在前，高位在后；低字节在前，高字节在后。

4.2 帧格式

4.2.1 帧格式定义

参考Modbus协议，每帧数据都包含地址域、功能码域、数据域和校验域。

电子锁 ID	地址域（4 字节）
功能码	功能域（1 字节）
数据	数据域
CRC 校验	校验域（2 字节）

图1 帧格式

4.2.2 功能域

表 2 功能码定义

功 能 码（十六进制）		应用功能定义
码	子码	
0x02		查询锁舌状态
0x03		查询锁信息
0x05		开锁控制
0x06	0x01	设置锁参数/设置电子锁标识号 ID
	0x02	设置锁参数/ 设置电子锁自动闭锁延时时间和报警时长
	0x03	设置锁参数/ 恢复出厂设置
0x42	0x01	升级准备
	0x02	升级开始
	0x03	升级
	0x04	升级完成

4.2.3 地址域

地址域由电子锁标识号组成，格式见 0：

表 1 地址域格式

地 址 域	数据格式	字 节 数
电子锁标识号	BCD	4

4.2.4 校验域

帧校验码（CRC）是地址域、功能域和数据域的 CRC 校验值。

4.3 传输方式

终端与电子锁之间采用RS485通信方式，通信速率9600bps。

字节传输按异步方式进行，它包含8个数据位、1个起始位“0”和1个停止位“1”，定义见：

0	D0	D1	D2	D3	D4	D5	D6	D7	1
起始位	8 个数据位								停止位

图2

5 功能码说明

5.1 查询锁舌状态（0x02）

查询锁舌的状态。

表 3 查询锁舌状态信息请求

功 能 码	1 字节（十六进制）	0x03

表 4 查询锁舌状态信息响应

功 能 码	1 字节（十六进制）	0x03
锁舌状态	1 字节（十六进制）	0x00：锁舌打开 0x01：锁舌关闭； 0x02：未知

5.2 查询锁信息（0x03）

根据电子锁设备号PA查询锁号ID。锁接收该命令时先判断电子锁设备号PA，与本电子锁设备号相同则上报相应信息。

表 3 查询锁信息请求

功 能 码	1 字节（十六进制）	0x03
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999

表 4 查询锁信息响应

功 能 码	1 字节（十六进制）	0x03
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
电子锁标识号（ID）	4 字节（BCD 码）	00000000-99999999
电子锁厂家名称	8 字节（ASCII 字符串）	“某某厂家”
电子锁硬件版本信息	4 字节（BCD 码）	XX.XX.XX.XX
电子锁软件版本信息	4 字节（BCD 码）	XX.XX.XX.XX

5.3 开锁控制（0x05）

控制电子锁开锁。

表 5 开锁请求

功 能 码	1 字节（十六进制）	0x05

表 6　开锁响应

功 能 码	1 字节（十六进制）	0x05
执行结果	1 字节（十六进制）	0x01：执行成功，0x02：执行失败

5.4　设置锁参数（0x06）

5.4.1　设置电子锁标识号

根据电子锁设备号PA设置电子锁标识号ID。

表 7　设置电子锁标识号请求

功 能 码	1 字节（十六进制）	0x06
子功能码	1 字节（十六进制）	0x01
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
电子锁标识号（ID）	4 字节（BCD 码）	00000000-99999999

表 8　设置电子锁标识号响应

功 能 码	1 字节（十六进制）	0x06
子功能码	1 字节（十六进制）	0x01
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
电子锁标识号（ID）	4 字节（BCD 码）	00000000-99999999
执行结果	1 字节（十六进制）	0x01：设置成功 0x02：设置失败

注：电子锁接收本报文时，不判断报文帧头的电子锁标识号（ID）的符合性。

5.4.2　设置电子锁自动闭锁延时时间和报警时长

设置自动闭锁延时时间和声光报警持续时间。自动闭锁延时时间：下达开锁命令后在闭锁延时时间内无人开锁则自动闭锁；声光报警持续时间：超过时间则停止声光报警。

表 9　设置电子锁自动闭锁延时时间和报警时长请求

功 能 码	1 字节（十六进制）	0x06
子功能码	1 字节（十六进制）	0x02
自动闭锁延时时间	2 字节（BCD 码）	005~600（秒）
声光报警持续时间	2 字节（BCD 码）	005~600（秒）

表 10　设置电子锁自动闭锁延时时间和报警时长响应

功 能 码	1 字节（十六进制）	0x06
子功能码	1 字节（十六进制）	0x02
执行结果	1 字节（十六进制）	0x01：设置成功 0x02：设置失败

5.4.3　恢复出厂设置

初始化电子锁参数，恢复出厂默认参数：电子锁标识号ID恢复为出厂时的设备号PA，即ID=PA。闭锁延时参数恢复默认参数20秒；报警时长参数恢复默认参数10秒。

表 9　恢复出厂设置请求

功 能 码	1 字节（十六进制）	0x06

子功能码	1 字节（十六进制）	0x03
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999

表 10　恢复出厂设置响应

功 能 码	1 字节（十六进制）	0x06
子功能码	1 字节（十六进制）	0x03
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
执行结果	1 字节（十六进制）	0x01：执行成功 0x02：执行失败

5.5　固件升级（0x42）

5.5.1　升级准备

根据控制器下发升级准备指令。

表 11　固件升级通知

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x01
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999

表 12　固件升级通知响应

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x01
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
执行结果	1 字节（十六进制）	0x01：可执行 0x02：不可执行

5.5.2　升级开始

根据控制器下发升级开始数据包，默认一帧大小为1024Byte，尾帧数据根据实际长短发送。

表 13　升级开始通知

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x02
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
总帧数	2 字节（十六进制）	1-65535
文件大小（字节数）	4 字节（十六进制）	0-4294967295

表 14　升级开始通知响应

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x02
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999

执行结果	1 字节（十六进制）	0x01：可执行 0x02：不可执行

5.5.3 升级

根据控制器下发固件数据包，默认一帧大小为1024Byte，尾帧数据根据实际长短发送。数据传输交互采用一问一答方式，电子锁数据接收后校验成功回复响应数据包，当回复异常时应重新下发未通过验证的数据包。每隔300ms同一数据包至多重发3次，如3次仍未通过则停止升级，升级失败。

表 15 升级包

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x03
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
帧序号	2 字节（十六进制）	1-65535
数据长度 Len（字节数）	2 字节（十六进制）	0-65535
数据内容	Len 字节（十六进制）	

表 16 升级包回复

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x03
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
帧序号	2 字节（十六进制）	1-65535，
执行结果	1 字节（十六进制）	0x01：成功 0x02：异常
异常说明	1 字节（十六进制）	0x01：数据校验错误 0x02：本地存储空间不足 0x03：其他

5.5.4 升级完成

数据包传输完毕后下发此指令，当电子锁回复完成响应后即自动进入固件升级模式。

表 17 升级完成通知

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x04
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999

表 18 升级完成通知响应

功 能 码	1 字节（十六进制）	0x42
子功能码	1 字节（十六进制）	0x04
电子锁设备号（PA）	4 字节（BCD 码）	00000000-99999999
执行结果	1 字节（十六进制）	0x01：执行成功，0x02：执行失败

AA

附 录 A

（规范性附录）

CRC 校验算法

循环冗余校验 CCRC）域占用两个字节，包含了一个16位的二进制值 。CRC值由传送设备计算出来，然后附加到数据帧上，接收设备在接收数据时重新计算CRC值，然后与接收到的CRC域中的值进行比较，如果这两个值不相等，就发生了错误。

CRC运算时，首先将一个16位的寄存器预置为全1，然后连续把数据帧中的每个字节中的8位与该寄存器的当前值进行运算，仅仅每个字节的8个数据位参与生成 CRC ，起始位和终止位以及可能使用的奇偶位都不影响CRC。

在生成CRC时，每个字节的8位与寄存器中的内容进行异或，然后将结果向低位移位，高位则用“0”补充，最低位（LSB）移出并检测，如果是1，该寄存器就与一个预设的固定值（OAOOlH）进行一次异或运算，如果最低位为0，不作任何处理。

上述处理重复进行，直到执行完了8次移位操作，当最后一位〈第8位〉移完以后，下一个8位字节与寄存器的当前值进行异或运算，同样进行上述的另一个8次移位异或操作，当数据帧中的所有字节都作了处理，生成的最终值就是CRC值。

生成一个CRC的流程为：

1）预置一个16位寄存器为0FFFFH（全1），称之为CRC寄存器。

2）把数据帧中的第一个字节的8位与CRC寄存器中的低字节进行异或运算，结果存回CRC寄存器。

3）将CRC寄存器向右移一位，最高位填以0，最低位移出并检测 。

4）如果最低位为0：重复第三步〈下一次移位） 。

如果最低位为1：将CRC寄存器与一个预设的固定值（OAOOlH ）进行异或运算 。

1）重复第三步和第四步直到8次移位。这样处理完了一个完整的八位 。

2）重复第2步到第5步来处理下一个八位，直到所有的字节处理结束 。

3）最终CRC寄存器得值就是CRC的值。

c＋＋代码样例如下 ：

```
WORD GetCRC16(BYTE *data,int len)
{
    WORD ax,lsb ;
    int 1,j ;

    ax=0xFFFF;
    for (i= 0 ;i<len ;i+ + )
    {
        ax ^= data[i];
        for (j= 0 ;j<8 ;j + + )
        {
            lsb= ax&0x0001 ;
            ax ^= ax >> l ;
            if Clsb ! = 0)
                ax^= 0xA001 ;
        }
    }
    return ax ;
}
```