1. **连接控制层（半双工）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1B | 2B | 0-2B | nB | 1B | 1B |
| F | C | A | 数据 | CRC8 | F |

1. 分隔字符 F= 0xE7
2. 转义字符 ESC = 0xEB

说明:<数据区>不允许出现F和ESC,这些字符必须按如下方式转义:

* 0xE7 => 0xEB, 0x07
* 0xEB => 0xEB, 0x0B

1. 控制字符C

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | | **9** | | **8** | | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **I** 帧 | 0 | 加密key | | | | | 0 | | AV | | 总ID | | | 当前ID | | | N(S) | | |
| **S** 帧 | 1 | 0 | 0 | 0 | 0 | 0 | | AV | | 0 | | 0 | 0 | 状态控制S | | | N(R) | | |

* AV 地址有效位。1表示带2B地址，0表示无地址模式
* N(S) 当前发送的帧号，每次加一，可用于确认上次的帧
* N（R） 下次接收的帧号，可用于确认接收
* 状态控制S

000 RR 准备接收

001 RNR 接收未准备，但是确认之前的所有帧

010 REJ 否认从N(R)开始的帧，但之前的已接收

011 SREJ 仅否认N(R)帧

100 RA 地址请求request address

* I帧多帧发送完一帧后可以由S帧来确认，最后一帧可以由I帧确认
* I帧单帧发送完后可以由S帧确认，如果没收到S帧收到I帧也表示发送已被确认
* 总ID，当前ID：用于分帧传输，总ID，当前ID相同则说明一条完整数据传输结束，如果单帧则两者都为0。

1. 地址字符A

根据控制字符的相应地址有效位，可以不存在，或者是2B地址。

1. **指令层**

|  |  |  |  |
| --- | --- | --- | --- |
| 命令控制 | Toggle/参数长度 | 参数 | Crc16 |
| 2B | 2B | X | 2B |

* 命令控制

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 指令帧 | 0 | P/F | ERR/MUT |  |  | 指令类CLA | | | | | | 命令CMD | | | | |

* P/F 查询/结束(Poll/Final)比特，作为主机命令帧发送时的查询比特，以 P（1） 位出现；作为从机响应帧发送时的结束比特，以 F（0） 位出现。
* ERR/MUT

主机发送时表示是否需要响应，0表示需要响应，1表示静默。

从机返回时表示指令执行是否出错，出错的话指令参数为错误代码

* Toggle/参数长度 指令层参数的长度，最大4095。

# 公共指令CLA=0

1. 读取设备id

发送：

Len=0

返回：Len=8

Data0-data7：设备id号

说明：这个命令直接返回，透传设备不透传

1. inventory设备id

发送：

Len=9

Data0-7：设备id

Data8：设备地址mask位数量x

返回：Len=8

Data0-data7：设备id号

指令错误则无返回值

说明：只有id号码和发送的id号前x位相同的设备才会回应

1. 静默

发送：

Len=0

返回：一般不需要返回

说明：静默后所有广播指令不响应

1. 激活

发送：Len=0

返回：一般不需要返回

1. 重启

发送：Len=1

Data0:0直接重启，1重启进升级模式

返回：len=0

1. 写IO口

发送：Len=x

Data0：起始io口索引

Data1：io口数量

Data2-datax：io口数据，一个口占用一个bit

返回：len=0

1. 读取IO口

发送：Len=2

Data0：起始io口索引

Data1：io口数量

返回：len=x

Data0-datax：io口数据

1. 写存储器

发送：Len=x

Data0-data3：起始地址

Data4-data7：数据数量

Data8-datax：数据

返回：len=0

1. 读存储器

发送：Len=x

Data0-data3：起始地址

Data4-data7：数据数量

返回：len=x

Data8-datax：数据

1. 毁灭

发送：Len=8

Data0-data3：设备地址

Data4-data7：0x33D5F980

# 键盘鼠标指令CLA=1

1. 鼠标控制数据（INS=0）
2. 键盘控制数据（INS=1）

# 电机控制指令CLA=2

# 传感器控制指令CLA=3

1. 查询传感器数量

发送：

Len=0

返回：Len=2

Data0-data1：传感器数量

1. 查询传感器属性

发送：

Len=4

Data0-data1：传感器起始索引

Data2-data3：查询数量

返回：Len=8\*x

Data0-data1：传感器1类型

Data2-data3：传感器1版本

Data4-Data7：传感器1 ID

传感器2

传感器3

。。。

1. 读取传感器当前值

发送：

Len=4

Data0-data1：传感器起始索引

Data2-data3：查询数量

返回：Len=4

Data0-data3：传感器1数据

Data4-data7：传感器2数据

。。。

。。。

1. 传感器数据透传

发送：

Len=4

Data0-data1：传感器起始索引

Data2-data3：保留

Data4-dataX：透传数据

返回：Len=x

Data0-dataX：传感器返回值

1. 设置高报警值

发送：

Len=x

Data0-data1：传感器起始索引

Data2-data3：传感器数量

Data4-datax7：传感器1预警上限

Data8-datax11：传感器1报警上限

Data12-datax15：传感器2预警上限

Data16-datax19：传感器2报警上限

。。。

返回：Len=0

1. 读取高报警值

发送：

Len=4

Data0-data1：传感器起始索引

Data2-data3：传感器数量

返回：Len=x

Data0-datax3：传感器1预警上限

Data4-datax7：传感器1报警上限

Data8-datax11：传感器2预警上限

Data12-datax15：传感器2报警上限

。。。

1. 设置报警值下限

发送：

Len=x

Data0-data1：传感器起始索引

Data2-data3：传感器数量

Data4-datax7：传感器1预警上限

Data8-datax11：传感器1报警上限

Data12-datax15：传感器2预警上限

Data16-datax19：传感器2报警上限

。。。

返回：Len=0

1. 读取报警值下限

发送：

Len=4

Data0-data1：传感器起始索引

Data2-data3：传感器数量

返回：Len=x

Data0-datax3：传感器1预警下限

Data4-datax7：传感器1报警下限

Data12-datax15：传感器2预警下限

Data16-datax19：传感器2报警下限

1. 读取当前报警状态

发送：

Len=4

Data0-data1：传感器起始索引

Data2-data3：传感器数量

返回：Len=x

Data0-datax：传感器报警状态（一个传感器一个字节）

1. 消除当前报警状态

发送：

Len=4+x

Data0-data1：传感器起始索引

Data2-data3：传感器数量

Data4-dataX:报警状态（一个传感器一个字节，置位表示消除）

返回：Len=0