# 格式说明

**协议修改点（8.16）：**

1. **包头包尾相同且固定，设置为0xC0；**
2. **发送/接收时数据时，需要进行转义，详见** [**1.3数据转义规则**](#_数据转义规则)**；**
3. **增加时间戳字节；**
4. **数据长度从2字节改为1字节；**
5. **校验位的计算改为从包头的下一个字节开始，校验位的前一个字节结束；**

## 协议变更背景

之前协议痛点：

1. 命令码不多（127个），后期将会超过；
2. 命令重复，例如在EC和SM中都将出现获取电池信息；
3. 数据包中无数据长度定义，无法灵活拓展；例如获取风扇转速，风扇个数被限制死；
4. 无校验，数据不可靠；
5. 数据需拆分为7位7位发送，不直观；

当前协议优点：

1. 解决以上痛点；
2. 命令标志中的应答位，不用再额外定义应答命令，并统一应答规范；
3. 命令标志中的跳过校验位，使在手动/调试输命令时，允许不计算校验位；
4. 增加发送者ID，使接收者可以判断发送来源并选择性接收，且后续数据的操作空间更大；
5. 增加接收者ID，使接收者可以判断该包数据是不是发给它的，可选择性接收。

## 数据包格式



数据包的具体含义如下：

* **包头、包尾：1 Byte，固定0xC0。**
* **发送者ID：1 Byte。**

|  |  |
| --- | --- |
| 类型 | ID |
| W1 | 0xA0 |
| Drivertool | 0xA1 |
| Update | 0xA2 |
|  |  |
| EC固件 | 0xB0 |
| SM固件 | 0xB1 |
| KB固件 | 0xB2 |
| PHV固件 | 0xB3 |
|  |  |
| FPGA ? | 0xC0 |

* **接收者ID：1 Byte。**

同发送者ID，但若接收者ID为0x00，代表广播所有接收者。

* **命令ID：2 Byte**。

|  |  |
| --- | --- |
| 命令ID | 含义 |
| 0x0000 | 握手命令 |
| … | … |
| 0x0001 | 获取版本号 |
| 0x0002 | 获取svn号 |
| … | … |
| 0xF001 | xx测试命令 |

* **命令标志：1 Byte。**



* **数据长度：1 Byte。**
* **数据：长度根据 数据长度 字节定。**
* **校验：2 Byte。**

## 数据转义规则

**发送：**

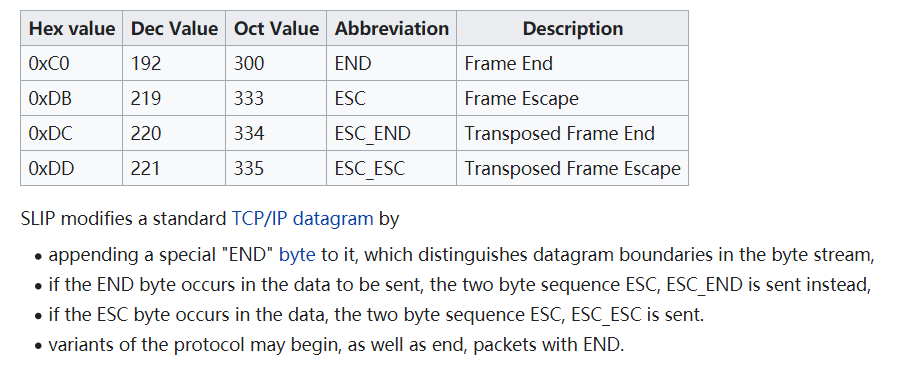
数据长度、校验都是以转义前的原数据进行计算。当数据中出现0xC0，则将该字节替换为0xDB和0xDC；当数据中出现0xDB，则将该字节替换为0xDB和0xDD。

接收时：

**接收：**

数据长度、校验都是以转义回原数据进行计算。当数据中出现0xDB、0xDC两字节，则替换为0xC0字节；当数据中出现0xDB、0xDD两字节，则替换为0xDB。

参考： <https://en.wikipedia.org/wiki/Serial_Line_Internet_Protocol>



## 串口通信

1. 波特率： 115200 bps；
2. 1位起始位；
3. 8位数据位；
4. 无奇偶校验位；
5. 1位停止位。

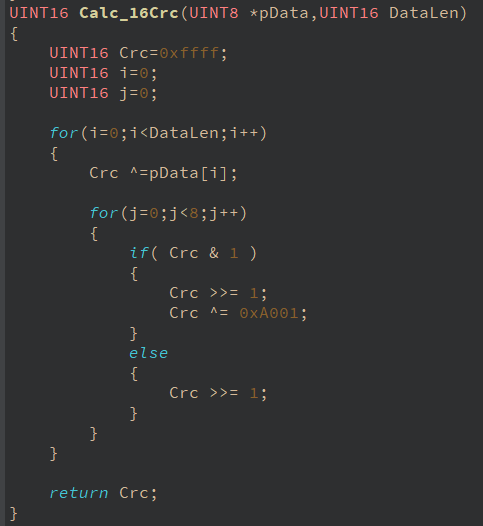
## 数据约定

采取小端模式：

1. 多字节数据，低位先发送，高位后发送；
2. 数组型数据，低字节数据先发送，高字节数据后发送。

## 数据校验

1. 校验方式：CRC 16位校验和；
2. 校验字节数：2个字节；
3. 约定方式：从包头后第一个字节到校验位的前一个字节(包括该字节)，进行CRC校验，发送CRC校验的时候遵循小端模式，即先发低字节，再发高字节；
4. CRC校验实现：



## 应答机制

为了防止丢包，通讯必须有应答机制。

* 部分模块对主机的部分命令要有应答，应答包有固定的命令ID，且应答包的时间戳为所应答包的时间戳、数据为需要应答的命令ID号；
* 主机没有收到应答之前要有重发机制，若主机连续重发三次还未收到模块的应答，则主机提示出错信息；
* 若应答包的应答位为1，代表主机需将命令重发。

## 应答包约定

* **帧格式：**与1.6节所述的常规格式保持一致。
* **命令ID：**固定为0xFFFF**。**
* **数据长度：**2Byte**。**
* **数据：**应答数据为所应答的命令ID。

## 通用性

* 命令ID中不再区分接收者，而是专门设立一个接收者ID。