技术应用文档

# Caven\_info\_frame

消息帧结构以及功能列表

消息帧结构以及功能列表

[otl协议技术应用文档](https://kdocs.cn/l/cjMY23q4aET2)

## 简介

功能：介绍该技术的功能和作用

作用：介绍该技术的功能和作用

目标群体：介绍该技术的目标群体或用户

使用场景：举例适合使用该技术的场景

本文档的所有示例，addr为0x01。

## 消息帧格式

使用前先看看这个。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Versions | Type | Addr | CMD | CMD\_SUB | Size | Data | Result | 帧尾（CRC校验） |
| U16 | U8 | U8 | U8 | U8 | U8 | U16 | 由[Size]决定 | U8 | U16 |
| FA 8A | 01 | 00 | 01 | 00 | 00 | 00 00 |  | 00 | 12 C1 |

示例(16进制)

[FA 8A] 01 00 01 00 00 [00 00]————————00 [12 C1]

9(byte) + 0(byte) data + 1(byte)Result + 2(byte)crc

FA 8A 01 00 01 00 00 00 00 00 12 C1

### 帧头

**FA 8A**

遇到这个数据说明来了一条消息。

### 版本

目前是 0x01

这个版本是这条**消息帧**的版本，理论上**向下兼容**。

### Type

设备类型（00）。

### Addr

这个是设备地址，例如主机对1号设备发送指令，那这个位置就是0x01。

当然他也有广播的作用（addr = 0xFF），主机才允许向设备广播。广播时，从机不允许回复。

如果主机发送的地址（addr = 0x00），说明这是特殊通道（这个通道之在一对一方式下使用），设备需要把它当作自己的addr来响应，响应时addr回复也应该为0x00。

### CMD

命令主体类型。如果CMD = 0x00，那就是测试位（无意义，但是合法）。

### CMD\_SUB

命令主体的细分指令。如果CMD\_SUB = 0x00，那就是测试位（无意义，但是合法）。

### Size

这个的数值，决定了后面Data的长度。

### Data

消息帧的有效内容，长度由[Size]决定。如果Size=0，那么Data区不存在。

### Result

设备收到消息帧的合法（00），消息帧协议本版不支持（01），CMD不支持（02），CMD\_SUB不支持（03），本帧消息不完整（04），其他待补充。

主机这一位发送的内容无效。

### 帧尾

把前面除去帧头部分的内容做CRC16校验，得出的值就是尾帧。

[CRC在线计算参考](https://www.lddgo.net/encrypt/crc) 选择CRC-16-MODBUS

1. //x16+x15+x2+1
2. int ModBusCRC16(unsigned char \*data, int len)
3. {
4. int i, j, temp, CRC16;
5. CRC16 = 0xFFFF; //CRC寄存器初始值
6. for (i = 0; i < len; i++)
7. {
8. CRC16 ^= data[i];
9. for (j = 0; j < 8; j++)
10. {
11. temp = (int)(CRC16 & 0x0001);
12. CRC16 >>= 1;
13. if (temp == 1)
14. {
15. CRC16 ^= 0xA001; //异或多项式
16. }
17. }
18. }
19. return CRC16;
20. }

### 参考

结构体参考

1. typedef stuart
2. {
3. unsigned short Head; //2
4. unsigned char Versions; //1
5. unsigned char Type; //1
6. unsigned char Addr; //1
7. unsigned char Cmd; //1
8. unsigned char Cmd\_sub; //1
9. unsigned short Size; //2
10. unsigned char \*p\_Data; //
11. unsigned char Result; //1
12. unsigned short End\_crc; //2
13. }Caven\_info\_frame\_Type;

结构体如果都是变量，结构体以[\_Type]结尾。

结构体如果都是函数指针，结构体以[\_Way]结尾。

结构体如果存在变量+函数指针（混合结构体），结构体命名以[\_Mix]结尾。

在混合结构体中，变量不做规定(指针除外)，指针函数以[\_pFun]结尾。

数据命名参考

1. int temp;    // 常规变量
2. const int c\_temp;
3. static int s\_temp;
4. extern int g\_temp;
5. int \*p\_temp = null;
6. int (\*temp\_pFun)(void) = NULL; // 强调其为指针函数
7. typedef enum
8. {
9. m\_GPO = 1, // 强调其为枚举内部数据
10. m\_MOSA,
11. m\_MOSB,
12. }User\_GPIO\_mType;    // 强调枚举类型

## **通用命令CMD = 0x01**

也就是符合此协议的所有设备都会响应的CMD（理论上boot模式也会有）。

如果是通用命令那么CMD = 0x01。

### CMD\_SUB功能列表

|  |  |  |  |
| --- | --- | --- | --- |
| CMD\_SUB | 含义 | 备注 | 属性读（R）/写（W） |
| 00 | 测试位 | 无意义 | 无 |
| 01 | 设备版本号 | 4byte（0.0.1.0） | R |
| 02 | 设备编号 | 20byte(ASCII) | R\W |
| 03 | 串口配置 | 查询或设置 | R\W |
| 04 | 地址设置 | 0x01-0xFE | R\W |
|  |  |  |  |

无意义数据，直接返回收到的内容。

### 设备版本号

查询设备版本号，查询一直响应，设置只允许设置一次。

#### 主机指令：

CMD:01

CMD\_SUB:01

查询：Data[0]:01

#### 示例：

例如设备版本号为：0.0.1.0

CMD:01

CMD\_SUB:01

Data: 00 00 01 00

主机发送：

FA 8A 01 00 01 01 01 00 01 01 00 E1 7B

7+2+[1]+1+2=13

设备回复：

FA 8A 01 00 01 01 01 00 05 01 00 00 01 00 00 D2 5C

7+2+[5]+1+2=17

### 设备编号

设备编号指令，这个是程序内固定的，不可通过本命令修改。

#### 主机指令：

CMD:01

CMD\_SUB:02

查询：Data[0]:01

设置：Data[0]:02

#### 示例：

##### 查询：

例如设备编号为：xxxxxxxxxx0123456789

回复

Data: 01 78 78 78 78 78 78 78 78 78 78 30 31 32 33 34 35 36 37 38 39

ASCII的'x'，16进制是0x78。

主机发送：

FA 8A 01 00 01 01 02 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 01 02 00 15 01 78 78 78 78 78 78 78 78 78 78 30 31 32 33 34 35 36 37 38 39 00 xx xx

##### 设置：

例如要写入设备编号为：aaaaaaaaaa0123456789

回复设置成功

Data: 02 00

回复设置失败

Data: 02 01

如果已经设置过一次，则需要报无权限

Data: 02 02

主机发送：

FA 8A 01 00 01 01 02 00 15 02 61 61 61 61 61 61 61 61 61 61 30 31 32 33 34 35 36 37 38 39 00 xx xx

设备回复：

FA 8A 01 00 01 01 02 00 02 02 00 00 xx xx

### 串口配置

设备查询配置串口波特率和设置。

#### 主机指令：

CMD:01

CMD\_SUB:03

查询：Data[0]:01

设置：Data[0]:02

#### 示例：

波特率枚举

0：无串口

1：9600

2：19200

3：38400

4：115200

5：230400

6：460800

##### 查询：

例如设备波特率为：115200

回复

Data: 01 04

主机发送：

FA 8A 01 00 01 01 03 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 01 03 00 02 01 04 00 xx xx

##### 设置：

例如要设置设备波特率为：9600

回复设置成功

Data: 02 00

如果不支持串口波特率

Data: 02 01

如果不支持串口通信

Data: 02 02

主机发送：

FA 8A 01 00 01 01 03 00 02 02 01 00 xx xx

设备回复：

FA 8A 01 00 01 01 03 00 02 02 00 00 xx xx

### 地址设置

设备配置地址查询和设置。

#### 主机指令：

CMD:01

CMD\_SUB:04

查询：Data[0]:01

设置：Data[0]:02

#### 示例：

##### 查询：

例如设备地址为：0x06。发送此命令应该在一对一情况下，向地址00发送此指令。

回复

Data: 01 06

主机发送：

FA 8A 01 00 00 01 04 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 00 01 04 00 02 01 06 00 xx xx

##### 设置：

例如要设置设备地址为：0x06。

回复设置成功

Data: 02 00

如果不支持地址

Data: 02 01

主机发送：

FA 8A 01 00 01 01 04 00 02 02 06 00 xx xx

设备回复：

FA 8A 01 00 01 01 04 00 02 02 00 00 xx xx

随后设备Addr变为0x06。

## BOOT指令 CMD = 0x02

部分大容量设备支持bootloarder升级。

如果是升级命令那么CMD = 0x02。

### CMD\_SUB功能列表

|  |  |  |
| --- | --- | --- |
| CMD\_SUB | 含义 | 备注 |
| 00 | 测试位 | 无意义 |
| 01 | 升级开始 | boot / app |
| 02 | 升级数据 | boot |
| 03 | 升级结束 | boot |
| 04 | 查询程序状态 | boot / app |
| 05 | 重启 | boot / app |
| 06 | debug | debug输出等级 |

### 升级开始

boot区：抹除app\_flag地址值，让app\_flag = 0x01，保存掉电参数区 。

app区：抹除app\_flag地址值，让app\_flag = 0x01，保存掉电参数区，进入boot区。

app\_flag地址为flash掉电保存区Sys\_BOOTSet结构体的参数。

#### 指令：

CMD:02

CMD\_SUB:01

Data[0]:01

#### 示例：

设备返回：如果**设备状态**支持升级 Data[1] = 00，不支持升级Data[1] = 01。

CMD:02

CMD\_SUB:01

设备状态支持升级：Data: 01 00

设备状态不支持升级：Data: 01 01

主机发送：

FA 8A 01 00 01 02 01 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 02 01 00 02 01 00 00 xx xx

### 升级数据

仅在BOOT区生效。

app区：回应Result = 3。

#### 指令：

CMD:02

CMD\_SUB:02

Data[0-1]：数据编号。

Data[2-n]：数据内容。

n = size - 1

#### 示例：

设备返回：如果本编号成功：00，失败：01（主机应该重发此编号数据）。

CMD:02

CMD\_SUB:02

本编号成功：Data: 编号 00

本编号成功：Data: 编号 01

主机发送：

FA 8A 01 00 01 02 02 00 0a 00 01 00 00 00 00 00 00 00 00 00 xx xx

模仿发送数据编号1，并且数据内容为0。

设备回复：

FA 8A 01 00 01 02 02 00 03 00 01 00 00 xx xx

### 升级结束

boot区：写入app\_flag地址值，让app\_flag = 0xE8，保存掉电参数区 。

app区：回应Result = 3。

app\_flag地址为flash掉电保存区Sys\_BOOTSet结构体的参数。

#### 指令：

CMD:02

CMD\_SUB:03

Data[0]:01

#### 示例：

设备返回：如果成功Data[1] = 00，不成功Data[1] = 01。

CMD:02

CMD\_SUB:03

成功：Data: 01 00

不成功：Data: 01 01

主机发送：

FA 8A 01 00 01 02 03 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 02 03 00 02 01 00 00 xx xx

### 查询程序状态

查询程序目前处于boot还是app区。

#### 指令：

CMD:02

CMD\_SUB:04

Data[0]:01

#### 示例：

设备返回：如果设备处于boot区Data[1] = 00，处于app区Data[1] = 01。

CMD:02

CMD\_SUB:04

设备在boot区：Data: 01 00

设备在app区：Data: 01 01

主机发送：

FA 8A 01 00 01 02 04 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 02 04 00 02 01 00 00 xx xx

### 重启

重启系统

#### 指令：

CMD:02

CMD\_SUB:05

Data[0]:01

#### 示例：

设备返回：允许并执行Data[1] = 00，不允许且不执行Data[1] = 01。

CMD:02

CMD\_SUB:05

执行：Data: 01 00

不执行：Data: 01 01

主机发送：

FA 8A 01 00 01 02 05 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 02 05 00 02 01 00 00 xx xx

## 业务指令 CMD = 0x03

设备功能实现的指令。

### CMD\_SUB功能列表

|  |  |  |
| --- | --- | --- |
| CMD\_SUB | 含义 | 备注 |
| 00 | 测试位 | 无意义 |
| 01 | 功能码1 | app |
|  |  |  |

### 功能码1

#### 指令：

CMD:03

CMD\_SUB:01

Data[0]:01

#### 示例：

设备返回： Data[1] = 00 。

CMD:03

CMD\_SUB:01

设备：Data: 00 00

主机发送：

FA 8A 01 00 01 03 01 00 01 01 00 xx xx

设备回复：

FA 8A 01 00 01 03 01 00 02 01 00 00 xx xx

## **建议与解答**

用什么软件来发送和调试？

答：VOFA+

主要通信？

答：串口、USB之类...

一主多从的模式？

答：主要是将设备变成一问一答的形式，物理通道是全双工的。有些数据量大要上传的，协议会让其打包一起询问上传。

## 更新和升级

### 更新日志

* 开始建立——2023.10.9
* 数据命名参考

## **参考与文献**

1. 参照于modbus协议。
2. 结构体、函数、变量命名规则。
3. 各种稀奇古怪。

## **Maker**

1. Cavendish

[GitHub](https://github.com/SwiperWitty)

...

1. Who are you

底线