

串行总线智能舵机

SMS1.0 协议手册

修订历史

| 日期 | 版本 | 更新内容 | 制定人 |
|------------|-------|------|----------|
| 2017.03.01 | V1.00 | 初次制定 | Alex lee |

1.0 通信协议概要

SMS1.0 协议主要适用于 FEETECH SMS 系列舵机。SMS 系列舵机采用 ARM 32 位单片机为主控制核心，位置感应采用 360 度 12 位精度的磁铁感应角度方案，通讯电平采用具有较强抗干扰能力的 RS485 方式。

控制器和舵机之间采用问答方式通信，控制器发出指令包，舵机返回应答包。

一个总线控制网络中允许有多个舵机，所以每个舵机都分配有一个网络内唯一的 ID 号。控制器发出的控制指令中包含 ID 信息，只有匹配上 ID 号的舵机才能完整接收这条指令，并返回应答信息。

通信方式为串行异步方式，一帧数据分为 1 位起始位，8 位数据位和 1 位停止位，无奇偶校验位，共 10 位。

1.1 指令包

指令包格式：

| 字头 | ID 号 | 数据长度 | 指令 | 参数 | 校验和 |
|-----------|------|--------|-------------|----------------------------|-----------|
| 0XFF 0XFF | ID | Length | Instruction | Parameter1 ... Parameter N | Check Sum |

字头：连续收到两个 0XFF，表示有数据包到达。

ID 号：每个舵机都有一个 ID 号。ID 号范围 0~253, 转换为十六进制 0X00~0XFD。

广播 ID：ID 号 254 为广播 ID, 若控制器发出的 ID 号为 254 (0XFE)，所有的舵机均接收指令，除 PING 指令外其它指令均不返回应答信息 (多个舵机连接在总线上不能使用广播 PING 指令)。

数据长度：等于待发送的参数长度 N 加上 2，即 “N+2”。

指令：数据包操作功能码，详见 1.3 指令类型。

参数：除指令另外需要补充的控制信息，参数最大支持双字节参数表示一个内存值，字节顺序参考舵机使用手册内存控制表 (不同型号舵机字节顺序不一样)。

校验和：校验和 Check Sum，计算方法如下

$$\text{Check Sum} = \sim (\text{ID} + \text{Length} + \text{Instruction} + \text{Parameter1} + \dots \text{Parameter N})$$
 若括号内的计算和超出 255，则取最低的一个字节，“~”表示取反。

1.2 应答包

应答包是舵机对控制器的应答，应答包格式如下：

| 字头 | ID号 | 数据长度 | 当前状态 | 参数 | 校验和 |
|-----------|-----|--------|-------|---------------------------|-----------|
| 0XFF 0XFF | ID | Length | ERROR | Parameter1 ...Parameter N | Check Sum |

返回的应答包包含舵机的当前状态 ERROR，若舵机当前工作状态不正常，会通过这个字节反映出来（各状态所代表意义详见手册内存控制表），若 ERROR 为 0，则舵机无报错信息。

若指令是读指令（READ DATA），则 Parameter1 ...Parameter N 是读取的信息。

1.3 指令类型

SMS1.0 协议可用指令如下：

| 指令 | 功能 | 值 | 参数长度 |
|---------------------|--|------|-------|
| PING（查询） | 查询工作状态 | 0x01 | 0 |
| READ DATA（读） | 查询控制表里的字符 | 0x02 | 2 |
| WRITE DATA（写） | 往控制表里写入字符 | 0x03 | 大于等于1 |
| REGWRITE DATA(异步写) | 类似于WRITE DATA，但是控制字符写入后并不马上动作，直到ACTION指令到达 | 0x04 | 不小于2 |
| ACTION（执行异步写） | 触发REG WRITE写入的动作 | 0x05 | 0 |
| SYCNWRITE DATA（同步写） | 用于同时控制多个舵机 | 0x83 | 不小于2 |
| RESET（复位） | 把控制表复位为出厂值 | 0x06 | 0 |

1.3.1 查询状态指令 PING

功能 读取舵机的工作状态
长度 0X02
指令 0X01
参数 无

PING 指令使用广播地址，舵机同样返回应答信息。

例 1 读取 ID 号为 1 的舵机的工作状态

指令帧：FF FF 01 02 01 FB`（按十六进制发送）

| 字头 | ID | 有效数据长度 | 指令 | 校验和 |
|-----------|------|--------|------|------|
| 0XFF 0XFF | 0X01 | 0X02 | 0X01 | 0XFB |

返回的数据帧：FF FF 01 02 00 FC（十六进制显示）

| 字头 | ID | 有效数据长度 | 工作状态 | 校验和 |
|-----------|------|--------|------|------|
| 0XFF 0XFF | 0X01 | 0X02 | 0X00 | 0XFC |

1.3.2 读指令 READ DATA

功能 从舵机内存控制表里读出数据
长度 0X04
指令 0X02
参数1 数据读出段的首地址
参数2 读取数据的长度

例 2 读取 ID 为 1 的舵机的当前位置(低位字节在前, 高位字节在后)。
在控制表里从地址 0X38 处读取二个字节。

指令帧: FF FF 01 04 02 38 02 BE (按十六进制发送)

| 字头 | ID | 有效数据长度 | 指令 | 参数 | 校验和 |
|-----------|------|--------|------|-----------|------|
| 0XFF 0XFF | 0X01 | 0X04 | 0X02 | 0X38 0X02 | 0XBE |

返回的数据帧: FF FF 01 04 00 18 05 DD (十六进制显示)

| 字头 | ID | 有效数据长度 | 工作状态 | 参数 | 校验和 |
|-----------|------|--------|------|-----------|------|
| 0XFF 0XFF | 0X01 | 0X04 | 0X00 | 0X18 0X05 | 0XDD |

读出两个字节数据分别是: 低位字节 L 0X18 高位字节 H 0X05

两字节合成 16 位数据 0X0518, 用十进制表示当前的位置为 1304。

1.3.3 写指令 WRITE DATA

功能 写数据到舵机内存控制表
长度 N+3 (N 为参数长度)
指令 0X03
参数 1 数据写入段的首地址
参数 2 写入的第一个数据
参数 3 第二个数据
参数 N+1 第 N 个数据

例 3 把一个任意编号的 ID 设置为 1。

在控制表里保存 ID 号的地址为 5, 所以在地址 5 处写入 1 即可。发送指令包的 ID 使用广播 ID (0xFE)。

指令帧: FF FF FE 04 03 05 01 F4 (按十六进制发送)

| 字头 | ID | 有效数据长度 | 指令 | 参数 | 校验和 |
|-----------|------|--------|------|-----------|------|
| 0XFF 0XFF | 0XFE | 0X04 | 0X03 | 0X05 0X01 | 0XF4 |

因为采用广播 ID 发送指令, 所以不会有数据返回。

另外内存表 EPROM 具有保护锁开关, 修改 ID 前需要把它关掉, 不然示例 ID 号掉电将不保存。详细操作请查阅具体舵机型号的内存表或操作手册。

例 4 控制 ID1 舵机以 1000 每秒的速度转动到 2048 的位置。

在控制表里目标位置的首地址为 0X2A, 所以在地址 0X2A 处开始写入连续六个字节数据, 分别是位置数据 0X0800(2048); 时间数据 0X0000(0); 速度数据 0X03E8(1000)。发送指令包的 ID 使用非广播 ID (0xFE), 所以指令接收完成后舵机将返回状态包。

指令帧: FF FF 01 09 03 2A 00 08 00 00 E8 03 D5 (按十六进制发送)

| 字头 | ID | 有效数据长度 | 指令 | 参数 | 校验和 |
|-----------|------|--------|------|---|------|
| 0XFF 0XFF | 0X01 | 0X09 | 0X03 | 0X2A 0X00 0X08 0X00 0X00 0XE8 0X03 | 0XD5 |

返回的数据帧: FF FF 01 02 00 FC (十六进制显示)

| 字头 | ID | 有效数据长度 | 工作状态 | 校验和 |
|-----------|------|--------|------|------|
| 0XFF 0XFF | 0X01 | 0X02 | 0X00 | 0XFC |

返回工作状态为 0, 表示舵机正确无错接收到指令并已开始执行。

1.3.4 异步写指令 REG WRITE

REG WRITE 指令类似于 WRITE DATA, 只是执行的时间不同。当收到 REG WRITE 指令帧时, 把收到的数据储存在缓冲区备用, 并把 Registered Instruction 寄存器置 1。当收到 ACTION 指令后, 储存的指令最终被执行。

长度 N+3 (N 为要写入数据的个数)

指令 0X04

参数 1 数据要写入区的首地址

参数 2 要写入的第一个数据

参数 3 要写入的第二个数据

参数 N+1 要写入的第 N 个数据

例 5 控制 ID1 到 ID10 舵机以 1000 每秒的速度转动到 2048 的位置, 以下指令包中只有 ID 在总线上并接收到指令并返回, 其他 ID 号不在总线返回。

ID 1 异步写指令包: FF FF 01 09 04 2A 00 08 00 00 E8 03 D4

ID 1 返回包: FF FF 01 02 00 FC

ID 2 异步写指令包: FF FF 02 09 04 2A 00 08 00 00 E8 03 D3

ID 3 异步写指令包: FF FF 03 09 04 2A 00 08 00 00 E8 03 D2

ID 4 异步写指令包: FF FF 04 09 04 2A 00 08 00 00 E8 03 D1

ID 5 异步写指令包: FF FF 05 09 04 2A 00 08 00 00 E8 03 D0

ID 6 异步写指令包: FF FF 06 09 04 2A 00 08 00 00 E8 03 CF

ID 7 异步写指令包: FF FF 07 09 04 2A 00 08 00 00 E8 03 CE

ID 8 异步写指令包: FF FF 08 09 04 2A 00 08 00 00 E8 03 CD

ID 9 异步写指令包: FF FF 09 09 04 2A 00 08 00 00 E8 03 CC

ID10 异步写指令包: FF FF 0A 09 04 2A 00 08 00 00 E8 03 CB

1.3.5 执行异步写指令 ACTION

| | |
|----|-----------------|
| 功能 | 触发 REG WRITE 指令 |
| 长度 | 0X02 |
| 指令 | 0X05 |
| 参数 | 无 |

ACTION 指令在同时控制多个舵机时非常有用。

在控制多个舵机时，使用 ACTION 指令可以使第一个和最后一个舵机同时执行各自的动作，中间无延时。

对多个舵机发送 ACTION 指令时，要用到广播 ID (0xFE)，因此，发送此指令不会有数据帧返回。

例 6 控制 ID1 到 ID10 舵机以 1000 每秒的速度转动到 2048 的位置的异步写指令发完后，需要执行异步写指令时，需要发送以下指令包 (FF FF FE 02 05 FA)。所有在总线上的舵机接收到此指令后将运行前面接收到的异步写指令。

1.4.5 同步写指令 SYNC WRITE

| | |
|---------|--|
| 功能 | 用于同时控制多个舵机。 |
| ID | 0xFE |
| 长度 | $(L + 1) * N + 4$ (L: 发给每个舵机的数据长度, N: 舵机的个数) |
| 指令 | 0X83 |
| 参数 1 | 写入数据的首地址 |
| 参数 2 | 写入的数据的长度 (L) |
| 参数 3 | 第一个舵机的 ID 号 |
| 参数 4 | 写入第一个舵机的第一个数据 |
| 参数 5 | 写入第一个舵机的第二个数据 |
| ... | |
| 参数 L+3 | 写入第一个舵机的第 L 个数据 |
| 参数 L+4 | 第二个舵机的 ID 号 |
| 参数 L+5 | 写入第二个舵机的第一个数据 |
| 参数 L+6 | 写入第二个舵机的第二个数据 |
| ... | |
| 参数 2L+4 | 写入第二个舵机的第 L 个数据 |
| ... | |

不同于 REG WRITE+ACTION 指令的是实时性比它更高，一条 SYNC WRITE 指令可一次修改多个舵机的控制表内容，而 REG WRITE+ACTION 指令是分步做到的。尽管如此，使用 SYNC WRITE 指令时，写入的数据长度和保存数据的首地址必须相同。

例 7 对 ID1-ID4 共 4 个舵机首地址 0X2A 写入位置 0X0800 时间 0X0000 和速度 0X03E8 (低位字节在前，高位字节在后)。

指令帧：FF FF FE 20 83 2A 06 01 00 08 00 00 E8 03 02 00 08 00 00 E8 03 03 00 08
00 00 E8 03 04 00 08 00 00 E8 03 58 (按十六进制发送)

| 字头 | ID | 有效数据长度 | 指令 | 参数 | 校验和 |
|-----------|------|--------|------|---|------|
| 0XFF 0XFF | 0XFE | 0X20 | 0X83 | 0X2A 0X06 0X01 0X00 0X08 0X00 0X00 0XE8 0X03 0X02 0X00 0X08 0X00 0X00 0XE8 0X03 0X03 0X00 0X08 0X00 0X00 0XE8 0X03 0X04 0X00 0X08 0X00 0X00 0XE8 0X03 | 0X58 |

因为采用广播 ID 发送指令，所以不会有数据返回。

1.4.6 复位指令 RESET

功能 把内存控制表里特定的数据进行复位（特定舵机型号采用）
长度 0X02
指令 0X06
参数 无

例 复位舵机，ID 号为 0。

指令帧：FF FF 01 02 06 F6 (按十六进制发送)

| 字头 | ID | 有效数据长度 | 指令 | 校验和 |
|-----------|------|--------|------|------|
| 0XFF 0XFF | 0X00 | 0X02 | 0X06 | 0XF7 |

返回的数据帧：FF FF 01 02 00 FC (十六进制显示)

| 字头 | ID | 有效数据长度 | 工作状态 | 校验和 |
|-----------|------|--------|------|------|
| 0XFF 0XFF | 0X01 | 0X02 | 0X00 | 0XFC |