

**上海领控科技**  
**电机 RS485 通讯协议**  
V2.36

## 目录

上海瓴控科技 .....	1
电机 RS485 通讯协议 .....	1
免责声明 .....	4
RS485 总线参数 .....	5
单电机命令 .....	5
1. 读取电机状态 1 和错误标志命令 .....	5
2. 清除电机错误标志命令 .....	6
3. 读取电机状态 2 命令 .....	7
4. 读取电机状态 3 命令 .....	7
5. 电机关闭命令 .....	8
6. 电机运行命令 .....	8
7. 电机停止命令 .....	9
8. 抱闸器状态控制和读取命令 .....	9
9. 开环控制命令（该命令仅在 MS 电机上实现） .....	9
10. 转矩闭环控制命令（该命令仅在 MF、MH、MG 电机上实现） .....	10
11. 速度闭环控制命令 .....	10
12. 多圈位置闭环控制命令 1 .....	11
13. 多圈位置闭环控制命令 2 .....	11
14. 单圈位置闭环控制命令 1 .....	12
15. 单圈位置闭环控制命令 2 .....	13
16. 增量位置闭环控制命令 1 .....	13
17. 增量位置闭环控制命令 2 .....	14
18. 读取控制参数命令 .....	15
19. 写入控制参数命令 .....	16
20. 读取编码器命令 .....	16
21. 设置当前位置作为电机零点命令（写入 ROM） .....	17
22. 读取多圈角度命令 .....	18
23. 清除电机圈数信息命令 .....	18
24. 读取单圈角度命令 .....	19
25. 设置当前位置为任意角度（写入 RAM） .....	19
附录一：电机控制参数表 .....	20



## 免责声明

感谢您购买上海瓴控科技有限公司电机驱动一体控制系统。在使用之前，请仔细阅读本声明，一旦使用，即被视为对本声明全部内容的认可和接受。请严格遵守产品手册、控制协议和相关的法律法规、政策、准则安装和使用该产品。在使用产品过程中，用户承诺对自己的行为及因此而产生的所有后果负责。因用户不当使用、安装、改装造成的任何损失，瓴控科技将不承担法律责任。

瓴控科技是上海瓴控科技有限公司及其关联公司的商标。本文出现的产品名称、品牌等，均为其所属公司的商标或注册商标。

本产品及手册为上海瓴控科技有限公司版权所有。未经许可，不得以任何形式复制翻印。关于免责声明的最终解释权，归本公司所有。

## RS485 总线参数

总线接口：RS485

波特率（常规模式，单电机命令）：

- 9600bps
- 19200bps
- 38400bps
- 57600bps
- 115200bps（默认）
- 230400bps
- 460800bps
- 1Mbps
- 2Mbps
- 4Mbps

波特率（广播模式，多电机命令）：

- 1Mbps
- 2Mbps
- 4Mbps

数据位：8

奇偶校验：无

停止位：1

## 单电机命令

同一总线上共可以挂载多达 32 个（视总线负载情况而定）驱动，为了防止总线冲突，每个驱动需要设置不同的 ID，ID 编号为 1~32。

主控向总线发送单电机命令帧，对应 ID 的电机在收到命令后执行，并在一段时间后（0.25ms 内）向主控发送相同 ID 的回复帧。命令帧报文和回复帧报文格式如下：帧命令 + 帧数据（可选），具体描述如下表所示

	数据描述	数据长度（byte）	说明
帧命令	帧头	1	帧头识别，0x3E
	命令	1	CMD
	ID	1	1~32，对应电机的 ID
	数据长度	1	描述帧命令附带的数据长度，视不同命令而定
	帧命令校验字节	1	CMD_SUM，帧命令所有字节校验和，保留低 8 位，高位抛弃
帧数据	数据	0~100	帧命令附带的数据
	帧数据校验字节	0 或 1	DATA_SUM，帧数据所有字节校验和，保留低 8 位，高位抛弃

### 1. 读取电机状态 1 和错误标志命令

该命令读取当前电机的温度、电压和错误状态标志

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E

CMD[1]	命令	0x9A
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

### 驱动回复

电机在收到命令后回复主机，该帧数据包含了以下参数：

1. 电机温度 `temperature` (`int8_t` 类型，单位 1°C/LSB)。
2. 母线电压 `voltage` (`int16_t` 类型，单位 0.01V/LSB)。
3. 母线电流 `current` (`int16_t` 类型，单位 0.01A/LSB)。
4. 电机状态 `motorState` (为 `uint8_t` 类型，各个位代表不同的电机状态)
5. 错误标志 `errorState` (为 `uint8_t` 类型，各个位代表不同的电机错误状态)

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x9A
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (8byte, 含校验)		
DATA[0]	电机温度	DATA[0] = *(uint8_t *)&temperature
DATA[1]	母线电压低字节	DATA[1] = *(uint8_t *)&voltage
DATA[2]	母线电压高字节	DATA[2] = *((uint8_t *)&voltage)+1
DATA[3]	母线电流低字节	DATA[3] = *(uint8_t *)&current
DATA[4]	母线电流高字节	DATA[4] = *((uint8_t *)&current)+1
DATA[5]	电机状态字节	DATA[5] = motorState
DATA[6]	错误状态字节	DATA[6] = errorState
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

备注：

1. `motorState = 0x00` 电机处于开启状态； `motorState = 0x10` 电机处于关闭状态。
2. `errorState` 各个位具体状态表如下

errorState 位	状态说明	0	1
0	低电压状态	正常	低压保护
1	高电压状态	正常	高压保护
2	驱动温度状态	正常	驱动过温
3	电机温度状态	正常	电机过温
4	电机电流状态	正常	电机过流
5	电机短路状态	正常	电机短路
6	堵转状态	正常	电机堵转
7	输入信号状态	正常	输入信号丢失超时

### 2. 清除电机错误标志命令

该命令清除当前电机的错误状态，电机收到后返回

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x9B

CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

#### 驱动回复

电机在收到命令后回复主机。回复数据和读取电机状态 1 和错误标志命令相同（仅命令字节 CMD[1] 不同，这里为 0x9B）

备注：

1. 电机状态没有恢复正常时，错误标志无法清除。

### 3. 读取电机状态 2 命令

该命令读取当前电机的温度、电机转矩电流（MF、MG）/电机输出功率（MS）、转速、编码器位置。

帧命令（5byte, 含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x9C
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

#### 驱动回复

电机在收到命令后回复主机，该帧数据中包含了以下参数。

1. 电机温度 `temperature`（`int8_t` 类型，1°C/LSB）。
2. MF、MG 电机的转矩电流值 `iq` 或 MS 电机的输出功率值 `power`，`int16_t` 类型。MG 电机 `iq` 分辨率为(66/4096 A) / LSB；MF 电机 `iq` 分辨率为(33/4096 A) / LSB。MS 电机 `power` 范围-1000~1000。
3. 电机转速 `speed`（`int16_t` 类型，1dps/LSB）。
4. 编码器值 `encoder`（`uint16_t` 类型，14bit 编码器的数值范围 0~16383，15bit 编码器的数值范围 0~32767，16bit 编码器的数值范围 0~65535）。

帧命令（5byte, 含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x9C
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（8byte, 含校验）		
DATA[0]	电机温度	DATA[0] = *(uint8_t *)&temperature)
DATA[1]	转矩电流低字节	DATA[1] = *(uint8_t *)&iq)
	输出功率低字节(MS 系列)	DATA[1] = *(uint8_t *)&power)
DATA[2]	转矩电流高字节	DATA[2] = *((uint8_t *)&iq)+1)
	输出功率高字节(MS 系列)	DATA[2] = *((uint8_t *)&power)+1)
DATA[3]	电机速度低字节	DATA[3] = *(uint8_t *)&speed)
DATA[4]	电机速度高字节	DATA[4] = *((uint8_t *)&speed)+1)
DATA[5]	编码器位置低字节	DATA[5] = *(uint8_t *)&encoder)
DATA[6]	编码器位置高字节	DATA[6] = *((uint8_t *)&encoder)+1)
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

### 4. 读取电机状态 3 命令

由于 MS 电机没有相电流采样，该命令在 MS 电机上无作用。

该命令读取当前电机的温度和 3 相电流数据

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x9D
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

#### 驱动回复 (13byte)

电机在收到命令后回复主机，该帧数据包含了以下数据：

1. 电机温度 temperature (int8\_t 类型, 1°C/LSB)
2. 相电流数据 iA、iB、iC, 数据类型为 int16\_t 类型, MG 电机相电流分辨率为(66/4096 A) / LSB; MF 电机相电流分辨率为(33/4096 A) / LSB。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x9D
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (8byte, 含校验)		
DATA[5]	电机温度	DATA[5] = *(uint8_t *)&temperature
DATA[6]	A 相电流低字节	DATA[6] = *(uint8_t *)&iA
DATA[7]	A 相电流高字节	DATA[7] = *((uint8_t *)&iA)+1
DATA[8]	B 相电流低字节	DATA[8] = *(uint8_t *)&iB
DATA[9]	B 相电流高字节	DATA[9] = *((uint8_t *)&iB)+1
DATA[10]	C 相电流低字节	DATA[10] = *(uint8_t *)&iC
DATA[11]	C 相电流高字节	DATA[11] = *((uint8_t *)&iC)+1
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

#### 5. 电机关闭命令

将电机从开启状态（上电后默认状态）切换到关闭状态，LED 由常亮转为慢闪。此时电机仍然可以回复命令，但不会执行动作

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x80
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

#### 驱动回复

和主机发送相同

#### 6. 电机运行命令

将电机从关闭状态切换到开启状态，LED 由慢闪转为常亮。此时再发送控制指令即可控制电机动作。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E

CMD[1]	命令	0x88
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

#### 驱动回复

和主机发送相同

### 7. 电机停止命令

停止电机，但不清除电机运行状态。再次发送控制指令即可控制电机动作。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x81
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

#### 驱动回复

和主机发送相同。

### 8. 抱闸器状态控制和读取命令

控制抱闸器的开合，或者读取当前抱闸器的状态。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x8C
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x01
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
DATA[0]	抱闸器状态控制和读取字节	0x00: 抱闸器断电，刹车启动 0x01: 抱闸器通电，刹车释放 0x10: 读取抱闸器状态
DATA_SUM	数据校验字节	DATA[0]字节校验和

#### 驱动回复

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x8C
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x01
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
DATA[0]	抱闸器状态字节	0x00: 抱闸器处于断电状态，刹车启动 0x01: 抱闸器处于通电状态，刹车释放
DATA_SUM	数据校验字节	DATA[0]字节校验和

### 9. 开环控制命令（该命令仅在 MS 电机上实现）

主机发送该命令以控制输出到电机的开环电压，控制值 powerControl 为 int16\_t 类型，数值范围-850~850，（电机电流和扭矩因电机而异）。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA0
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x02
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (3byte, 含校验)		
DATA[0]	开环控制值低字节	DATA[0] = *(uint8_t *)&powerControl
DATA[1]	开环控制值高字节	DATA[1] = *((uint8_t *)&powerControl)+1
DATA_SUM	数据校验字节	DATA[0]~DATA[1]字节校验和

#### 驱动回复

电机在收到命令后回复主机。电机回复数据和**读取电机状态 2 命令**相同（仅命令字节 CMD[1]不同，这里为 0xA0）。

### 10. 转矩闭环控制命令（该命令仅在 MF、MH、MG 电机上实现）

主机发送该命令以控制电机的转矩电流输出，控制值 iqControl 为 int16\_t 类型，数值范围-2048~ 2048，对应 MF 电机实际转矩电流范围-16.5A~16.5A，对应 MG 电机实际转矩电流范围-33A~33A，母线电流和电机的实际扭矩因不同电机而异。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA1
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x02
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (3byte, 含校验)		
DATA[0]	转矩电流控制值低字节	DATA[0] = *(uint8_t *)&iqControl
DATA[1]	转矩电流控制值高字节	DATA[1] = *((uint8_t *)&iqControl)+1
DATA_SUM	数据校验字节	DATA[0]~DATA[1]字节校验和

#### 驱动回复

电机在收到命令后回复主机。电机回复数据和**读取电机状态 2 命令**相同（仅命令字节 CMD[1]不同，这里为 0xA1）。

### 11. 速度闭环控制命令

主机发送该命令以控制电机的速度，控制值 speedControl 为 int32\_t 类型，对应实际转速为 0.01dps/LSB。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA2
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x04
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (5byte, 含校验)		
DATA[0]	电机速度低字节	DATA[0] = *(uint8_t *)&speedControl
DATA[1]	电机速度	DATA[1] = *((uint8_t *)&speedControl)+1

DATA[2]	电机速度	DATA[2] = *((uint8_t *)&speedControl)+2)
DATA[3]	电机速度高字节	DATA[3] = *((uint8_t *)&speedControl)+3)
DATA_SUM	数据校验字节	DATA[0]~DATA[3]字节校验和

备注:

1. 该命令下电机的 speedControl 由上位机中的 Max Speed 值限制。
2. 该控制模式下，电机的最大加速度由上位机中的 Max Acceleration 值限制。
3. 该控制模式下，MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制；MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和读取电机状态 2 命令相同（仅命令字节 CMD[1]不同，这里为 0xA2）。

## 12. 多圈位置闭环控制命令 1

主机发送该命令以控制电机的位置（多圈角度），控制值 angleControl 为 int64\_t 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°，电机转动方向由目标位置和当前位置的差值决定。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA3
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x08
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（9byte，含校验）		
DATA[0]	位置控制低字节 1	DATA[0] = *((uint8_t *)&angleControl)
DATA[1]	位置控制字节 2	DATA[1] = *((uint8_t *)&angleControl)+1)
DATA[2]	位置控制字节 3	DATA[2] = *((uint8_t *)&angleControl)+2)
DATA[3]	位置控制字节 4	DATA[3] = *((uint8_t *)&angleControl)+3)
DATA[4]	位置控制字节 5	DATA[4] = *((uint8_t *)&angleControl)+4)
DATA[5]	位置控制字节 6	DATA[5] = *((uint8_t *)&angleControl)+5)
DATA[6]	位置控制字节 7	DATA[6] = *((uint8_t *)&angleControl)+6)
DATA[7]	位置控制高字节 8	DATA[7] = *((uint8_t *)&angleControl)+7)
DATA_SUM	数据校验字节	DATA[0]~DATA[7]字节校验和

备注:

1. 该命令下的控制值 angleControl 受上位机中的 Max Angle 值限制。
2. 该命令下电机的最大速度由上位机中的 Max Speed 值限制。
3. 该控制模式下，电机的最大加速度由上位机中的 Max Acceleration 值限制。
4. 该控制模式下，MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制；MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和读取电机状态 2 命令相同（仅命令字节 CMD[1]不同，这里为 0xA3）。

## 13. 多圈位置闭环控制命令 2

主机发送该命令以控制电机的位置（多圈角度）

1. 控制值 angleControl 为 int64\_t 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°，电机

转动方向由目标位置和当前位置的差值决定。

- 控制值 `maxSpeed` 限制了电机转动的最大速度，为 `uint32_t` 类型，对应实际转速 0.01dps/LSB，即 36000 代表 360dps。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA4
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x0C
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（13byte，含校验）		
DATA[0]	位置控制低字节 1	DATA[0] = *((uint8_t *)&angleControl)
DATA[1]	位置控制字节 2	DATA[1] = *((uint8_t *)&angleControl)+1
DATA[2]	位置控制字节 3	DATA[2] = *((uint8_t *)&angleControl)+2
DATA[3]	位置控制字节 4	DATA[3] = *((uint8_t *)&angleControl)+3
DATA[4]	位置控制字节 5	DATA[4] = *((uint8_t *)&angleControl)+4
DATA[5]	位置控制字节 6	DATA[5] = *((uint8_t *)&angleControl)+5
DATA[6]	位置控制字节 7	DATA[6] = *((uint8_t *)&angleControl)+6
DATA[7]	位置控制高字节 8	DATA[7] = *((uint8_t *)&angleControl)+7
DATA[8]	速度限制低字节 1	DATA[8] = *((uint8_t *)&maxSpeed)
DATA[9]	速度限制字节 2	DATA[9] = *((uint8_t *)&maxSpeed)+1
DATA[10]	速度限制字节 3	DATA[10] = *((uint8_t *)&maxSpeed)+2
DATA[11]	速度限制高字节 4	DATA[11] = *((uint8_t *)&maxSpeed)+3
DATA_SUM	数据校验字节	DATA[0]~DATA[11]字节校验和

备注：

- 该命令下的控制值 `angleControl` 受上位机中的 Max Angle 值限制。
- 该控制模式下，电机的最大加速度由上位机中的 Max Acceleration 值限制。
- 该控制模式下，MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制；MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和读取电机状态 2 命令相同（仅命令字节 CMD[1]不同，这里为 0xA4）

## 14. 单圈位置闭环控制命令 1

主机发送该命令以控制电机的位置（单圈角度）。

- 控制值 `spinDirection` 设置电机转动的方向，为 `uint8_t` 类型，0x00 代表顺时针，0x01 代表逆时针
- 控制值 `angleControl` 为 `uint16_t` 类型，数值范围 0~35999，对应实际位置为 0.01degree/LSB，即实际角度范围 0°~359.99°。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA5
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x04
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（5byte，含校验）		
DATA[0]	转动方向字节	DATA[0] = spinDirection

DATA[1]	位置控制字节 1	DATA[1] = *(uint8_t *)&angleControl
DATA[2]	位置控制字节 2	DATA[2] = *((uint8_t *)&angleControl)+1
DATA[3]	NULL	0x00
DATA_SUM	数据校验字节	DATA[0]~DATA[3]字节校验和

备注:

1. 该命令下电机的最大速度由上位机中的 Max Speed 值限制。
2. 该控制模式下，电机的最大加速度由上位机中的 Max Acceleration 值限制。
3. 该控制模式下，MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制；MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和**读取电机状态 2 命令**相同（仅命令字节 CMD[1]不同，这里为 0xA5）

## 15. 单圈位置闭环控制命令 2

主机发送该命令以控制电机的位置（单圈角度）。

1. 控制值 spinDirection 设置电机转动的方向，为 uint8\_t 类型，0x00 代表顺时针，0x01 代表逆时针
2. 角度控制值 angleControl 为 uint16\_t 类型，数值范围 0~35999，对应实际位置为 0.01degree/LSB，即实际角度范围 0°~359.99°。
3. 速度控制值 maxSpeed 限制了电机转动的最大速度，为 uint32\_t 类型，对应实际转速 0.01dps/LSB，即 36000 代表 360dps。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA6
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x08
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（9byte，含校验）		
DATA[0]	转动方向字节	DATA[0] = spinDirection
DATA[1]	位置控制字节 1	DATA[1] = *(uint8_t *)&angleControl
DATA[2]	位置控制字节 2	DATA[2] = *((uint8_t *)&angleControl)+1
DATA[3]	NULL	DATA[3] = 0x00
DATA[4]	速度限制低字节 1	DATA[4] = *(uint8_t *)&maxSpeed
DATA[5]	速度限制字节 2	DATA[5] = *((uint8_t *)&maxSpeed)+1
DATA[6]	速度限制字节 3	DATA[6] = *((uint8_t *)&maxSpeed)+2
DATA[7]	速度限制高字节 4	DATA[7] = *((uint8_t *)&maxSpeed)+3
DATA_SUM	数据校验字节	DATA[0]~DATA[7]字节校验和

备注:

1. 该控制模式下，电机的最大加速度由上位机中的 Max Acceleration 值限制。
2. 该控制模式下，MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制；MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和**读取电机状态 2 命令**相同（仅命令字节 CMD[1]不同，这里为 0xA6）

## 16. 增量位置闭环控制命令 1

主机发送该命令以控制电机的增量位置。

控制值 `angleIncrement` 为 `int32_t` 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°，电机转动方向由该参数的符号决定。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA7
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x04
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（5byte，含校验）		
DATA[0]	增量位置控制低字节 1	DATA[0] = *((uint8_t *)&angleIncrement)
DATA[1]	增量位置控制字节 2	DATA[1] = *((uint8_t *)&angleIncrement)+1
DATA[2]	增量位置控制字节 3	DATA[2] = *((uint8_t *)&angleIncrement)+2
DATA[3]	增量位置控制高字节 4	DATA[3] = *((uint8_t *)&angleIncrement)+3
DATA_SUM	数据校验字节	DATA[0]~DATA[3]字节校验和

备注：

1. 该命令下电机的最大速度由上位机中的 Max Speed 值限制。
2. 该控制模式下，电机的最大加速度由上位机中的 Max Acceleration 值限制。
3. 该控制模式下，MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制；MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和读取电机状态 2 命令相同（仅命令字节 CMD[1]不同，这里为 0xA7）

## 17. 增量位置闭环控制命令 2

主机发送该命令以控制电机的增量位置。

1. 控制值 `angleIncrement` 为 `int32_t` 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°，电机转动方向由该参数的符号决定。
2. 控制值 `maxSpeed` 限制了电机转动的最大速度，为 `uint32_t` 类型，对应实际转速 0.01dps/LSB，即 36000 代表 360dps。

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xA8
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x08
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（9byte，含校验）		
DATA[0]	增量位置控制低字节 1	DATA[0] = *((uint8_t *)&angleIncrement)
DATA[1]	增量位置控制字节 2	DATA[1] = *((uint8_t *)&angleIncrement)+1
DATA[2]	增量位置控制字节 3	DATA[2] = *((uint8_t *)&angleIncrement)+2
DATA[3]	增量位置控制高字节 4	DATA[3] = *((uint8_t *)&angleIncrement)+3
DATA[4]	速度限制字节 2	DATA[4] = *((uint8_t *)&maxSpeed)+1
DATA[5]	速度限制字节 3	DATA[5] = *((uint8_t *)&maxSpeed)+2
DATA[6]	速度限制高字节 4	DATA[6] = *((uint8_t *)&maxSpeed)+3
DATA[7]	速度限制字节 2	DATA[7] = *((uint8_t *)&maxSpeed)+1

DATA_SUM	数据校验字节	DATA[0]~DATA[7]字节校验和
----------	--------	----------------------

备注:

1. 该控制模式下, 电机的最大加速度由上位机中的 Max Acceleration 值限制。
2. 该控制模式下, MF、MH、MG 电机的最大转矩电流由上位机中的 Max Torque Current 值限制; MS 电机的最大功率由上位机中的 Max Power 值限制。

### 驱动回复

电机在收到命令后回复主机。电机回复数据和[读取电机状态 2 命令](#)相同 (仅命令字节 CMD[1]不同, 这里为 0xA8)

## 18. 读取控制参数命令

主机发送该命令读取当前电机的控制参数, 读取的参数由序号 controlParamID 确定, 见[电机控制参数表](#)

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xC0
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (3byte, 含校验)		
DATA[0]	参数序号	DATA[0] = controlParamID
DATA[1]	参数字节 1	DATA[1] = 0x00
DATA[2]	参数字节 2	DATA[2] = 0x00
DATA[3]	参数字节 3	DATA[3] = 0x00
DATA[4]	参数字节 4	DATA[4] = 0x00
DATA[5]	参数字节 5	DATA[5] = 0x00
DATA[6]	参数字节 6	DATA[6] = 0x00
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

### 驱动回复

驱动回复的数据中包含了读取的参数值, 具体参数见[电机控制参数表](#)

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xC0
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (8byte, 含校验)		
DATA[0]	参数序号	DATA[0] = controlParamID
DATA[1]	参数字节 1	DATA[1] = controlParamByte1
DATA[2]	参数字节 2	DATA[2] = controlParamByte2
DATA[3]	参数字节 3	DATA[3] = controlParamByte3
DATA[4]	参数字节 4	DATA[4] = controlParamByte4
DATA[5]	参数字节 5	DATA[5] = controlParamByte5
DATA[6]	参数字节 6	DATA[6] = controlParamByte6
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

## 19. 写入控制参数命令

主机发送该命令写入控制参数到 RAM 中，即时生效，断电后失效。写入的参数和序号 controlParamID 见[电机控制参数表](#)

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xC1
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（8byte，含校验）		
DATA[0]	参数序号	DATA[0] = controlParamID
DATA[1]	参数字节 1	DATA[1] = controlParamByte1
DATA[2]	参数字节 2	DATA[2] = controlParamByte2
DATA[3]	参数字节 3	DATA[3] = controlParamByte3
DATA[4]	参数字节 4	DATA[4] = controlParamByte4
DATA[5]	参数字节 5	DATA[5] = controlParamByte5
DATA[6]	参数字节 6	DATA[6] = controlParamByte6
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

### 驱动回复

驱动回复的数据中包含了写入后的参数值，具体的参数见[电机控制参数表](#)

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0xC1
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x07
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据（8byte，含校验）		
DATA[0]	参数序号	DATA[0] = controlParamID
DATA[1]	参数字节 1	DATA[1] = controlParamByte1
DATA[2]	参数字节 2	DATA[2] = controlParamByte2
DATA[3]	参数字节 3	DATA[3] = controlParamByte3
DATA[4]	参数字节 4	DATA[4] = controlParamByte4
DATA[5]	参数字节 5	DATA[5] = controlParamByte5
DATA[6]	参数字节 6	DATA[6] = controlParamByte6
DATA_SUM	数据校验字节	DATA[0]~DATA[6]字节校验和

## 20. 读取编码器命令

主机发送该命令以读取当前编码器的当前位置

帧命令（5byte，含校验）		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x90
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

## 驱动回复

电机在收到命令后回复主机，回复数据中包含了以下参数。

1. 编码器位置 `encoder` (`uint16_t` 类型，数值范围与编码器分辨率相关)，为编码器原始位置减去编码器零偏后的值。
2. 编码器原始位置 `encoderRaw` (`uint16_t` 类型，数值范围与编码器分辨率相关)。
3. 编码器零偏 `encoderOffset` (`uint16_t` 类型，数值范围与编码器分辨率相关)，该点为电机上电后的初始零位。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x90
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x06
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (7byte, 含校验)		
DATA[0]	编码器数据低字节	DATA[0] = *(uint8_t *)&encoder
DATA[1]	编码器数据高字节	DATA[1] = *((uint8_t *)&encoder)+1
DATA[2]	编码器原始位置低字节	DATA[2] = *(uint8_t *)&encoderRaw
DATA[3]	编码器原始位置高字节	DATA[3] = *((uint8_t *)&encoderRaw)+1
DATA[4]	编码器零偏低字节	DATA[4] = *(uint8_t *)&encoderOffset
DATA[5]	编码器零偏高字节	DATA[5] = *((uint8_t *)&encoderOffset)+1
DATA_SUM	数据校验字节	DATA[0]~DATA[5]字节校验和

备注:

1. 14bit 分辨率编码器的数值范围 0~16383; 15bit 分辨率编码器的数值范围 0~32767; 18bit 分辨率编码器的数值范围 0~65535 (保留高位 16bit, 省略低位 2bit)。

### 21. 设置当前位置作为电机零点命令 (写入 ROM)

设置电机当前位置的编码器原始值作为电机上电后的初始零点

注意:

1. 该命令会将零点写入驱动的 FLASH, 多次写入将会影响芯片寿命, 不建议频繁使用

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x19
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

## 驱动回复

电机在收到命令后回复主机，回复数据中包含了以下参数。

1. 当前位置的编码器原始值 `encoderZero`

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x19
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x02
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (3byte, 含校验)		

DATA[0]	零点编码器原始值低字节	DATA[0] =*(uint8_t *)&encoderZero)
DATA[1]	零点编码器原始值高字节	DATA[1] =*((uint8_t *)&encoderZero)+1)
DATA_SUM	数据校验字节	DATA[0]~DATA[1]字节校验和

## 22. 读取多圈角度命令

主机发送该命令以读取当前电机的多圈绝对角度值

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x92
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

### 驱动回复

电机在收到命令后回复主机，该帧数据中包含了以下参数：

1. 电机角度 `motorAngle`，为 `int64_t` 类型数据，正值表示顺时针累计角度，负值表示逆时针累计角度，单位  $0.01^\circ/\text{LSB}$ 。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x92
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x08
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (9byte, 含校验)		
DATA[0]	角度低字节 1	DATA[0] = *(uint8_t *)&motorAngle)
DATA[1]	角度字节 2	DATA[1] = *((uint8_t *)& motorAngle)+1)
DATA[2]	角度字节 3	DATA[2] = *((uint8_t *)& motorAngle)+2)
DATA[3]	角度字节 4	DATA[3] = *((uint8_t *)& motorAngle)+3)
DATA[4]	角度字节 5	DATA[4] = *((uint8_t *)& motorAngle)+4)
DATA[5]	角度字节 6	DATA[5] = *((uint8_t *)& motorAngle)+5)
DATA[6]	角度字节 7	DATA[6] = *((uint8_t *)& motorAngle)+6)
DATA[7]	角度高字节 8	DATA[7] = *((uint8_t *)& motorAngle)+6)
DATA_SUM	数据校验字节	DATA[0]~DATA[7]字节校验和

## 23. 清除电机圈数信息命令

主机发送该命令以清除当前电机的圈数信息

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x93
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

### 驱动回复 (8byte)

和主机发送相同

## 24. 读取单圈角度命令

主机发送该命令以读取当前电机的多圈绝对角度值

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x94
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x00
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和

### 驱动回复

电机在收到命令后回复主机，该帧数据中包含了以下参数：

1. 电机单圈角度 `circleAngle`，为 `uint32_t` 类型数据，以编码器零点为起始点，顺时针增加，再次到达零点时数值回 0，单位  $0.01^\circ/\text{LSB}$ ，数值范围  $0\sim 36000-1$ 。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x94
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x04
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (5byte, 含校验)		
DATA[0]	单圈角度低字节 1	DATA[0] = *((uint8_t *)&circleAngle)
DATA[1]	单圈角度字节 2	DATA[1] = *((uint8_t *)&circleAngle)+1
DATA[2]	单圈角度字节 3	DATA[2] = *((uint8_t *)&circleAngle)+2
DATA[3]	单圈角度高字节 4	DATA[3] = *((uint8_t *)&circleAngle)+3
DATA_SUM	数据校验字节	DATA[0]~DATA[3]字节校验和

## 25. 设置当前位置为任意角度 (写入 RAM)

主机发送该命令以设置电机的当前位置作为任意角度 (多圈)，多圈角度值 `motorAngle` 为 `int32_t` 类型数据，数据单位  $0.01^\circ/\text{LSB}$ 。

帧命令 (5byte, 含校验)		
CMD[0]	帧头	0x3E
CMD[1]	命令	0x95
CMD[2]	ID	0x01~0x20
CMD[3]	数据长度	0x04
CMD_SUM	帧命令校验和字节	CMD[0]~CMD[3]字节校验和
帧数据 (5byte, 含校验)		
DATA[0]	多圈角度低字节 1	DATA[0] = *((uint8_t *)&motorAngle)
DATA[1]	多圈角度字节 2	DATA[1] = *((uint8_t *)&motorAngle)+1
DATA[2]	多圈角度字节 3	DATA[2] = *((uint8_t *)&motorAngle)+2
DATA[3]	多圈角度高字节 4	DATA[3] = *((uint8_t *)&motorAngle)+3
DATA_SUM	数据校验字节	DATA[0]~DATA[3]字节校验和

### 驱动回复 (8byte)

和主机发送相同

## 附录一：电机控制参数表

电机控制参数表	
参数序号 ParamID	控制参数说明
10 (0x0A)	角度环 pid, 包含三个参数 anglePidKp (角度环 kp, uint16_t 类型) controlParamByte1 = *(uint8_t *)& anglePidKp controlParamByte2 = *((uint8_t *)& anglePidKp)+1 anglePidKi (角度环 ki, uint16_t 类型) controlParamByte3 = *(uint8_t *)& anglePidKi controlParamByte4 = *((uint8_t *)& anglePidKi)+1 anglePidKd (角度环 kd, uint16_t 类型) controlParamByte5 = *(uint8_t *)& anglePidKd controlParamByte6 = *((uint8_t *)& anglePidKd)+1
11 (0x0B)	速度环 pid, 包含三个参数 speedPidKp (速度环 kp, uint16_t 类型) controlParamByte1 = *(uint8_t *)& speedPidKp controlParamByte2 = *((uint8_t *)& speedPidKp)+1 speedPidKi (速度环 ki, uint16_t 类型) controlParamByte3 = *(uint8_t *)& speedPidKi controlParamByte4 = *((uint8_t *)& speedPidKi)+1 speedPidKd (速度环 kd, uint16_t 类型) controlParamByte5 = *(uint8_t *)& speedPidKd controlParamByte6 = *((uint8_t *)& speedPidKd)+1
12 (0x0C)	电流环 pid, 包含三个参数 currentPidKp (电流环 kp, uint16_t 类型) controlParamByte1 = *(uint8_t *)& currentPidKp controlParamByte2 = *((uint8_t *)& currentPidKp)+1 currentPidKi (电流环 ki, uint16_t 类型) controlParamByte3 = *(uint8_t *)& currentPidKi controlParamByte4 = *((uint8_t *)& currentPidKi)+1 currentPidKd (电流环 kd, uint16_t 类型) controlParamByte5 = *(uint8_t *)& currentPidKd controlParamByte6 = *((uint8_t *)& currentPidKd)+1
30(0x1E)	inputTorqueLimit (最大力矩电流, int16_t 类型) controlParamByte3 = *(uint8_t *)& inputTorqueLimit controlParamByte4 = *((uint8_t *)& inputTorqueLimit)+1
32 (0x20)	inputSpeedLimit (最大速度, int32_t 类型) controlParamByte3 = *(uint8_t *)& inputSpeedLimit controlParamByte4 = *((uint8_t *)& inputSpeedLimit)+1 controlParamByte5 = *((uint8_t *)& inputSpeedLimit)+2 controlParamByte6 = *((uint8_t *)& inputSpeedLimit)+3
34 (0x22)	inputAngleLimit(角度限制, int32_t 类型) controlParamByte3 = *(uint8_t *)& inputAngleLimit controlParamByte4 = *((uint8_t *)& inputAngleLimit)+1

	<pre>controlParamByte5 = *((uint8_t *)&amp; inputAngleLimit)+2 controlParamByte6 = *((uint8_t *)&amp; inputAngleLimit)+3</pre>
36 (0x24)	<pre>inputCurrentRamp(电流斜率, int32_t 类型) controlParamByte3 = *(uint8_t *)&amp; inputCurrentRamp controlParamByte4 = *((uint8_t *)&amp; inputCurrentRamp)+1 controlParamByte5 = *((uint8_t *)&amp; inputCurrentRamp)+2 controlParamByte6 = *((uint8_t *)&amp; inputCurrentRamp)+3</pre>
38 (0x26)	<pre>inputSpeedRamp(速度斜率, int32_t 类型) controlParamByte3 = *(uint8_t *)&amp; inputSpeedRamp controlParamByte4 = *((uint8_t *)&amp; inputSpeedRamp)+1 controlParamByte5 = *((uint8_t *)&amp; inputSpeedRamp)+2 controlParamByte6 = *((uint8_t *)&amp; inputSpeedRamp)+3</pre>