

上海慕纬度科技
电机 CAN 总线通讯协议
V2.35

目录

免责声明	4
一、 CAN 总线参数	5
二、 单电机命令	5
1. 电机关闭命令	5
2. 电机运行命令	5
3. 电机停止命令	6
4. 开环控制命令（该命令仅在 MS 电机上实现）	6
5. 转矩闭环控制命令（该命令仅在 MF、MH、MG 电机上实现）	7
6. 速度闭环控制命令（1 帧）	8
7. 位置闭环控制命令 1（1 帧）	8
8. 位置闭环控制命令 2（1 帧）	9
9. 位置闭环控制命令 3（1 帧）	9
10. 位置闭环控制命令 4（1 帧）	10
11. 位置闭环控制命令 5（1 帧）	10
12. 位置闭环控制命令 6（1 帧）	11
13. 读取 PID 参数命令（1 帧）	11
14. 写入 PID 参数到 RAM 命令（1 帧）	12
15. 写入 PID 参数到 ROM 命令（1 帧）	12
16. 读取加速度命令（1 帧）	13
17. 写入加速度到 RAM 命令（1 帧）	13
18. 读取编码器数据命令（1 帧）	13
19. 写入编码器值到 ROM 作为电机零点命令（1 帧）	14
20. 写入当前位置到 ROM 作为电机零点命令（1 帧）	14
21. 读取多圈角度命令（1 帧）	15
22. 读取单圈角度命令（1 帧）	16
23. 清除电机角度命令（1 帧）暂未实现	16
24. 读取电机状态 1 和错误标志命令（1 帧）	17
25. 清除电机错误标志命令（1 帧）	17
26. 读取电机状态 2 命令（1 帧）	18

27. 读取电机状态 3 命令 (1 帧)	19
三、 多电机命令	19
1. 多电机转矩闭环控制命令 (1 帧)	19

免责声明

感谢您购买本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，本公司将不承担法律责任。

一、CAN 总线参数

总线接口：CAN

波特率（常规模式，单电机命令）：

- 1Mbps（默认）
- 500kbps
- 250kbps
- 125kbps
- 100kbps

波特率（广播模式，多电机命令）：

- 1Mbps
- 500kbps

二、单电机命令

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

主控向总线发送单电机命令，对应 ID 的电机在收到命令后执行，并在一段时间后（0.25ms 内）向主控发送回复。命令报文和回复报文格式如下：

- 标识符：0x140 + ID(1~32)
- 帧格式：数据帧
- 帧类型：标准帧
- DLC：8 字节

1. 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，清除电机转动圈数及之前接收的控制指令，LED 由常亮转为慢闪。此时电机仍然可以回复控制命令，但不会执行动作。

数据域	说明	数据
DATA[0]	命令字节	0x80
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复

和主机发送相同。

2. 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。

数据域	说明	数据
DATA[0]	命令字节	0x88

DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复

和主机发送相同。

3. 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。

数据域	说明	数据
DATA[0]	命令字节	0x81
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

和主机发送相同。

4. 开环控制命令（该命令仅在 MS 电机上实现，其他电机无效）

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）。

数据域	说明	数据
DATA[0]	命令字节	0xA0
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	开环控制值低字节	DATA[4] = *((uint8_t *)&powerControl)
DATA[5]	开环控制值高字节	DATA[5] = *((uint8_t *)&powerControl)+1
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

备注：

1. 该命令中的控制值 powerControl 不受上位机中的 Max Power 值限制。

驱动回复（1 帧）

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℃/LSB。
2. 电机输出功率值 power，int16_t 类型，范围-850~850。
3. 电机转速 speed，int16_t 类型，1dps/LSB。

4. 编码器位置值 encoder, uint16_t 类型, , 15bit 编码器的数值范围 0~32767; 18bit 编码器的数值范围 0~65535 (保留高 16bit, 省略低 2bit)。

数据域	说明	数据
DATA[0]	命令字节	0xA0
DATA[1]	电机温度	DATA[1] = *(uint8_t *)&temperature
DATA[2]	转矩电流低字节	DATA[2] = *(uint8_t *)&power
DATA[3]	转矩电流高字节	DATA[3] = *((uint8_t *)&power)+1
DATA[4]	电机速度低字节	DATA[4] = *(uint8_t *)&speed
DATA[5]	电机速度高字节	DATA[5] = *((uint8_t *)&speed)+1
DATA[6]	编码器位置低字节	DATA[6] = *(uint8_t *)&encoder
DATA[7]	编码器位置高字节	DATA[7] = *((uint8_t *)&encoder)+1

5. 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。

数据域	说明	数据
DATA[0]	命令字节	0xA1
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	转矩电流控制值低字节	DATA[4] = *(uint8_t *)&iqControl
DATA[5]	转矩电流控制值高字节	DATA[5] = *((uint8_t *)&iqControl)+1
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

备注:

1. 该命令中的控制值 iqControl 不受上位机中的 Max Torque Current 值限制。

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℃/LSB。
2. 电机的转矩电流值 iq, int16_t 类型, 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。
3. 电机转速 speed, int16_t 类型, 1dps/LSB。
4. 编码器位置值 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; 18bit 编码器的数值范围 0~65535 (保留高位 16bit, 省略低位 2bit)。

数据域	说明	数据
DATA[0]	命令字节	0xA1
DATA[1]	电机温度	DATA[1] = *(uint8_t *)&temperature
DATA[2]	转矩电流低字节	DATA[2] = *(uint8_t *)&iq
DATA[3]	转矩电流高字节	DATA[3] = *((uint8_t *)&iq)+1
DATA[4]	电机速度低字节	DATA[4] = *(uint8_t *)&speed
DATA[5]	电机速度高字节	DATA[5] = *((uint8_t *)&speed)+1

DATA[6]	编码器位置低字节	DATA[6] = *(uint8_t *)&encoder
DATA[7]	编码器位置高字节	DATA[7] = *((uint8_t *)&encoder)+1

6. 速度闭环控制命令（1 帧）

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。

数据域	说明	数据
DATA[0]	命令字节	0xA2
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	速度控制低字节	DATA[4] = *(uint8_t *)&speedControl
DATA[5]	速度控制	DATA[5] = *((uint8_t *)&speedControl)+1
DATA[6]	速度控制	DATA[6] = *((uint8_t *)&speedControl)+2
DATA[7]	速度控制高字节	DATA[7] = *((uint8_t *)&speedControl)+3

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（1 帧）

电机在收到命令后回复主机。MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA2）；MF、MH、MG 电机回复数据和转矩闭环控制命令相同（仅命令字节不同，这里为 0xA2）。

7. 多圈位置闭环控制命令 1（1 帧）

主机发送该命令以控制电机的位置（多圈角度），控制值 angleControl 为 int32_t 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°，电机转动方向由目标位置和当前位置的差值决定。

数据域	说明	数据
DATA[0]	命令字节	0xA3
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	位置控制低字节	DATA[4] = *(uint8_t *)&angleControl
DATA[5]	位置控制	DATA[5] = *((uint8_t *)&angleControl)+1
DATA[6]	位置控制	DATA[6] = *((uint8_t *)&angleControl)+2
DATA[7]	位置控制高字节	DATA[7] = *((uint8_t *)&angleControl)+3

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（1 帧）

电机在收到命令后回复主机，MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA3）；MF、MH、MG 电机回复数据和力矩闭环控制命令相同（仅命令字节不同，这里为 0xA3）。

8. 多圈位置闭环控制命令 2（1 帧）

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 `int32_t` 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°，电机转动方向由目标位置和当前位置的差值决定。
2. 控制值 `maxSpeed` 限制了电机转动的最大速度，为 `uint16_t` 类型，对应实际转速 1dps/LSB，即 360 代表 360dps。

数据域	说明	数据
DATA[0]	命令字节	0xA4
DATA[1]	NULL	0x00
DATA[2]	速度限制低字节	DATA[2] = *((uint8_t *)&maxSpeed)
DATA[3]	速度限制高字节	DATA[3] = *((uint8_t *)&maxSpeed)+1
DATA[4]	位置控制低字节	DATA[4] = *((uint8_t *)&angleControl)
DATA[5]	位置控制	DATA[5] = *((uint8_t *)&angleControl)+1
DATA[6]	位置控制	DATA[6] = *((uint8_t *)&angleControl)+2
DATA[7]	位置控制高字节	DATA[7] = *((uint8_t *)&angleControl)+3

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（1 帧）

电机在收到命令后回复主机。MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA4）；MF、MH、MG 电机回复数据和力矩闭环控制命令相同（仅命令字节不同，这里为 0xA4）。

9. 单圈位置闭环控制命令 1（1 帧）

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. 控制值 `angleControl` 为 `uint32_t` 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°。

数据域	说明	数据
DATA[0]	命令字节	0xA5
DATA[1]	转动方向字节	DATA[1] = spinDirection
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	位置控制字节 1 (bit0 : bit7)	DATA[4] = *((uint8_t *)&angleControl)
DATA[5]	位置控制字节 2 (bit8 : bit15)	DATA[5] = *((uint8_t *)&angleControl)+1
DATA[6]	位置控制字节 3 (bit16 : bit23)	DATA[6] = *((uint8_t *)&angleControl)+2
DATA[7]	位置控制字节 4 (bit24: bit31)	DATA[7] = *((uint8_t *)&angleControl)+3

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（1 帧）

电机在收到命令后回复主机。MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA5）；MF、MH、MG 电机回复数据和力矩闭环控制命令相同（仅命令字节不同，这里为 0xA5）。

10. 单圈位置闭环控制命令 2（1 帧）

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
- angleControl 为 uint32_t 类型，对应实际位置为 0.01degree/LSB，即 36000 代表 360°。
- 速度控制值 maxSpeed 限制了电机转动的最大速度，为 uint16_t 类型，对应实际转速 1dps/LSB，即 360 代表 360dps。

数据域	说明	数据
DATA[0]	命令字节	0xA6
DATA[1]	转动方向字节	DATA[1] = spinDirection
DATA[2]	速度限制字节 1 (bit0 : bit7)	DATA[2] = *((uint8_t *)&maxSpeed)
DATA[3]	速度限制字节 2 (bit8 : bit15)	DATA[3] = *((uint8_t *)&maxSpeed)+1)
DATA[4]	位置控制字节 1 (bit0 : bit7)	DATA[4] = *((uint8_t *)&angleControl)
DATA[5]	位置控制字节 2 (bit8 : bit15)	DATA[5] = *((uint8_t *)&angleControl)+1)
DATA[6]	位置控制字节 3 (bit16 : bit23)	DATA[6] = *((uint8_t *)&angleControl)+2)
DATA[7]	位置控制字节 4 (bit24: bit31)	DATA[7] = *((uint8_t *)&angleControl)+3)

备注：

- 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（1 帧）

电机在收到命令后回复主机。MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA6）；MF、MH、MG 电机回复数据和力矩闭环控制命令相同（仅命令字节不同，这里为 0xA6）。

11. 增量位置闭环控制命令 1（1 帧）

主机发送该命令以控制电机的位置增量。

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

数据域	说明	数据
DATA[0]	命令字节	0xA7
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	位置控制低字节	DATA[4] = *((uint8_t *)& angleIncrement)
DATA[5]	位置控制	DATA[5] = *((uint8_t *)& angleIncrement)+1)
DATA[6]	位置控制	DATA[6] = *((uint8_t *)& angleIncrement)+2)

DATA[7]	位置控制高字节	DATA[7] = *((uint8_t *)&angleIncrement)+3)
---------	---------	--

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（1 帧）

电机在收到命令后回复主机。MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA7）；MF、MH、MG 电机回复数据和力矩闭环控制命令相同（仅命令字节不同，这里为 0xA7）。

12. 增量位置闭环控制命令 2（1 帧）

主机发送该命令以控制电机的位置增量。

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 1dps/LSB，即 360 代表 360dps。

数据域	说明	数据
DATA[0]	命令字节	0xA8
DATA[1]	NULL	0x00
DATA[2]	速度限制低字节	DATA[2] = *(uint8_t *)&maxSpeed)
DATA[3]	速度限制高字节	DATA[3] = *((uint8_t *)&maxSpeed)+1)
DATA[4]	位置控制低字节	DATA[4] = *(uint8_t *)&angleIncrement)
DATA[5]	位置控制	DATA[5] = *((uint8_t *)&angleIncrement)+1)
DATA[6]	位置控制	DATA[6] = *((uint8_t *)&angleIncrement)+2)
DATA[7]	位置控制高字节	DATA[7] = *((uint8_t *)&angleIncrement)+3)

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（1 帧）

电机在收到命令后回复主机。MS 电机回复数据和开环控制命令相同（仅命令字节不同，这里为 0xA8）；MF、MH、MG 电机回复数据和力矩闭环控制命令相同（仅命令字节不同，这里为 0xA8）。

13. 读取 PID 参数命令（1 帧）

主机发送该命令读取当前电机的的 PID 参数

数据域	说明	数据
DATA[0]	命令字节	0x30
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00

DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

驱动回复数据中包含了各个控制环路的 PI 参数。

数据域	说明	数据
DATA[0]	命令字节	0x30
DATA[1]	NULL	0x00
DATA[2]	位置环 P 参数	DATA[2] = anglePidKp
DATA[3]	位置环 I 参数	DATA[3] = anglePidKi
DATA[4]	速度环 P 参数	DATA[4] = speedPidKp
DATA[5]	速度环 I 参数	DATA[5] = speedPidKi
DATA[6]	转矩环 P 参数	DATA[6] = iqPidKp
DATA[7]	转矩环 I 参数	DATA[7] = iqPidKi

14. 写入 PID 参数到 RAM 命令（1 帧）

主机发送该命令写入 PID 参数到 RAM 中，断电后写入参数失效

DATA[0]	命令字节	0x31
DATA[1]	NULL	0x00
数据域	说明	数据
DATA[2]	位置环 P 参数	DATA[2] = anglePidKp
DATA[3]	位置环 I 参数	DATA[3] = anglePidKi
DATA[4]	速度环 P 参数	DATA[4] = speedPidKp
DATA[5]	速度环 I 参数	DATA[5] = speedPidKi
DATA[6]	转矩环 P 参数	DATA[6] = iqPidKp
DATA[7]	转矩环 I 参数	DATA[7] = iqPidKi

驱动回复（1 帧）

电机在收到命令后回复主机，回复命令和接收命令一致

15. 写入 PID 参数到 ROM 命令（1 帧）

主机发送该命令写入 PID 参数到 ROM 中，断电仍然有效

数据域	说明	数据
DATA[0]	命令字节	0x32
DATA[1]	NULL	0x00
DATA[2]	位置环 P 参数	DATA[2] = anglePidKp
DATA[3]	位置环 I 参数	DATA[3] = anglePidKi
DATA[4]	速度环 P 参数	DATA[4] = speedPidKp
DATA[5]	速度环 I 参数	DATA[5] = speedPidKi
DATA[6]	转矩环 P 参数	DATA[6] = iqPidKp
DATA[7]	转矩环 I 参数	DATA[7] = iqPidKi

驱动回复（1 帧）

电机在收到命令后回复主机，回复命令和接收命令一致

16. 读取加速度命令（1 帧）

主机发送该命令读取当前电机的加速度参数

数据域	说明	数据
DATA[0]	命令字节	0x33
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

驱动回复数据中包含了加速度参数。加速度数据 Accel 为 int32_t 类型，单位 1dps/s

数据域	说明	数据
DATA[0]	命令字节	0x33
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	加速度低字节 1	DATA[4] = *((uint8_t *)&Accel)
DATA[5]	加速度字节 2	DATA[5] = *((uint8_t *)&Accel)+1
DATA[6]	加速度字节 3	DATA[6] = *((uint8_t *)&Accel)+2
DATA[7]	加速度字节 4	DATA[7] = *((uint8_t *)&Accel)+3

17. 写入加速度到 RAM 命令（1 帧）

主机发送该命令写入加速度到 RAM 中，断电后写入参数失效。加速度数据 Accel 为 int32_t 类型，单位 1dps/s

数据域	说明	数据
DATA[0]	命令字节	0x34
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	加速度低字节 1	DATA[4] = *((uint8_t *)&Accel)
DATA[5]	加速度字节 2	DATA[5] = *((uint8_t *)&Accel)+1
DATA[6]	加速度字节 3	DATA[6] = *((uint8_t *)&Accel)+2
DATA[7]	加速度字节 4	DATA[7] = *((uint8_t *)&Accel)+3

驱动回复（1 帧）

电机在收到命令后回复主机，回复命令和接收命令一致

18. 读取编码器数据命令（1 帧）

主机发送该命令以读取编码器的当前位置

数据域	说明	数据
DATA[0]	命令字节	0x90

DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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. 编码器位置 `encoder`（`uint16_t` 类型，14bit 编码器的数值范围 0~16383），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，14bit 编码器的数值范围 0~16383）。
3. 编码器零偏 `encoderOffset`（`uint16_t` 类型，14bit 编码器的数值范围 0~16383），该点作为电机角度的 0 点。

数据域	说明	数据
DATA[0]	命令字节	0x90
DATA[1]	NULL	0x00
DATA[2]	编码器位置低字节	DATA[2] = *((uint8_t *)&encoder)
DATA[3]	编码器位置高字节	DATA[3] = *((uint8_t *)&encoder)+1
DATA[4]	编码器原始位置低字节	DATA[4] = *((uint8_t *)&encoderRaw)
DATA[5]	编码器原始位置高字节	DATA[5] = *((uint8_t *)&encoderRaw)+1
DATA[6]	编码器零偏低字节	DATA[6] = *((uint8_t *)&encoderOffset)
DATA[7]	编码器零偏高字节	DATA[7] = *((uint8_t *)&encoderOffset)+1

19. 写入编码器值到 ROM 作为电机零点命令（1 帧）

主机发送该命令以设置编码器的零偏，其中，需要写入的编码器值 `encoderOffset` 为 `uint16_t` 类型，14bit 编码器的数值范围 0~16383。

数据域	说明	数据
DATA[0]	命令字节	0x91
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	编码器零偏低字节	DATA[6] = *((uint8_t *)&encoderOffset)
DATA[7]	编码器零偏高字节	DATA[7] = *((uint8_t *)&encoderOffset)+1

驱动回复（1 帧）

电机在收到命令后回复主机，该帧数据和主机发送的命令相同。

20. 写入当前位置到 ROM 作为电机零点命令（1 帧）

将电机当前编码器位置作为初始位置写入到 ROM

注意：

1. 该命令需要重新上电后才能生效

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

数据域	说明	数据
DATA[0]	命令字节	0x19
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

电机在收到命令后回复主机，数据中 encoderOffset 为设置的 0 偏值

数据域	说明	数据
DATA[0]	命令字节	0x19
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	编码器零偏低字节	DATA[6] = *(uint8_t *)&encoderOffset
DATA[7]	编码器零偏高字节	DATA[7] = *((uint8_t *)&encoderOffset)+1

21. 读取多圈角度命令（1 帧）

主机发送该命令以读取当前电机的多圈绝对角度值

数据域	说明	数据
DATA[0]	命令字节	0x92
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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° /LSB。

数据域	说明	数据
DATA[0]	命令字节	0x92
DATA[1]	角度低字节 1	DATA[1] = *(uint8_t *)&motorAngle
DATA[2]	角度字节 2	DATA[2] = *((uint8_t *)& motorAngle)+1
DATA[3]	角度字节 3	DATA[3] = *((uint8_t *)& motorAngle)+2

DATA[4]	角度字节 4	DATA[4] = *((uint8_t *)& motorAngle)+3)
DATA[5]	角度字节 5	DATA[5] = *((uint8_t *)& motorAngle)+4)
DATA[6]	角度字节 6	DATA[6] = *((uint8_t *)& motorAngle)+5)
DATA[7]	角度字节 7	DATA[7] = *((uint8_t *)& motorAngle)+6)

22. 读取单圈角度命令（1 帧）

主机发送该命令以读取当前电机的单圈角度

数据域	说明	数据
DATA[0]	命令字节	0x94
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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° /LSB，数值范围 0~36000*减速比-1。

数据域	说明	数据
DATA[0]	命令字节	0x94
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	单圈角度低字节 1	DATA[4] = *((uint8_t *)& circleAngle)
DATA[5]	单圈角度字节 2	DATA[5] = *((uint8_t *)& circleAngle)+1)
DATA[6]	单圈角度字节 3	DATA[6] = *((uint8_t *)& circleAngle)+2)
DATA[7]	单圈角度高字节 4	DATA[7] = *((uint8_t *)& circleAngle)+3)

23. 清除电机角度命令（1 帧）暂未实现

该命令清除电机的多圈和单圈角度数据，并将当前位置设为电机的零点，断电后失效

注意：该命令会同时清除所有位置环的控制命令数据

数据域	说明	数据
DATA[0]	命令字节	0x95
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

电机在收到命令后回复主机，帧数据和主机发送相同

24. 读取电机状态 1 和错误标志命令（1 帧）

该命令读取当前电机的温度、电压和错误状态标志

数据域	说明	数据
DATA[0]	命令字节	0x9A
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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 $1^{\circ}\text{C}/\text{LSB}$ ）。
2. 电压 `voltage`（`uint16_t` 类型，单位 $0.1\text{V}/\text{LSB}$ ）。
3. 错误标志 `errorState`（为 `uint8_t` 类型，各个位代表不同的电机状态）

数据域	说明	数据
DATA[0]	命令字节	0x9A
DATA[1]	电机温度	$\text{DATA}[1] = *(\text{uint8_t } *)(&\text{temperature})$
DATA[2]	NULL	0x00
DATA[3]	电压低字节	$\text{DATA}[3] = *(\text{uint8_t } *)(&\text{voltage})$
DATA[4]	电压高字节	$\text{DATA}[4] = *(\text{uint8_t } *)(&\text{voltage}) + 1$
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	错误状态字节	$\text{DATA}[7] = \text{errorState}$

备注：

1. `errorState` 各个位具体状态表如下

<code>errorState</code> 位	状态说明	0	1
0	电压状态	电压正常	低压保护
1	无效		
2	无效		
3	温度状态	温度正常	过温保护
4	无效		
5	无效		
6	无效		
7	无效		

25. 清除电机错误标志命令（1 帧）

该命令清除当前电机的错误状态，电机收到后返回

数据域	说明	数据
DATA[0]	命令字节	0x9B

DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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 $1^{\circ}\text{C}/\text{LSB}$ ）。
2. 电压 `voltage`（`uint16_t` 类型，单位 $0.1\text{V}/\text{LSB}$ ）。
3. 错误标志 `errorState`（为 `uint8_t` 类型，各个位代表不同的电机状态）。

数据域	说明	数据
DATA[0]	命令字节	0x9A
DATA[1]	电机温度	DATA[1] = <code>*(uint8_t *)&temperature</code>
DATA[2]	NULL	0x00
DATA[3]	电压低字节	DATA[3] = <code>*(uint8_t *)&voltage</code>
DATA[4]	电压高字节	DATA[4] = <code>*((uint8_t *)&voltage)+1</code>
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	错误状态字节	DATA[7]= <code>errorState</code>

备注：

1. 电机状态没有恢复正常时，错误标志无法清除。
2. `errorState` 各个位具体状态参考[读取电机状态 1 和错误标志命令](#)。

26. 读取电机状态 2 命令（1 帧）

该命令读取当前电机的温度、电压、转速、编码器位置。

数据域	说明	数据
DATA[0]	命令字节	0x9C
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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， $1^{\circ}\text{C}/\text{LSB}$ ）。
2. 电机的转矩电流值 `iq`（`int16_t` 类型，范围 -2048~2048，对应实际转矩电流范围 -33A~33A）。
3. 电机转速 `speed`（`int16_t` 类型， $1\text{dps}/\text{LSB}$ ）。
4. 编码器位置值 `encoder`（`uint16_t` 类型，14bit 编码器的数值范围 0~16383）。

数据域	说明	数据
DATA[0]	命令字节	0x9C
DATA[1]	电机温度	DATA[1] = *(uint8_t *)&temperature)
DATA[2]	转矩电流低字节	DATA[2] = *(uint8_t *)&iq)
DATA[3]	转矩电流高字节	DATA[3] = *((uint8_t *)&iq)+1)
DATA[4]	电机速度低字节	DATA[4] = *(uint8_t *)&speed)
DATA[5]	电机速度高字节	DATA[5] = *((uint8_t *)&speed)+1)
DATA[6]	编码器位置低字节	DATA[6] = *(uint8_t *)&encoder)
DATA[7]	编码器位置高字节	DATA[7] = *((uint8_t *)&encoder)+1)

27. 读取电机状态 3 命令（1 帧）

该命令读取当前电机的温度和相电流数据

数据域	说明	数据
DATA[0]	命令字节	0x9D
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

驱动回复（1 帧）

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℃/LSB）
2. A 相电流数据，数据类型为 int16_t 类型，对应实际相电流为 1A/64LSB。
3. B 相电流数据，数据类型为 int16_t 类型，对应实际相电流为 1A/64LSB。
4. C 相电流数据，数据类型为 int16_t 类型，对应实际相电流为 1A/64LSB。

数据域	说明	数据
DATA[0]	命令字节	0x9D
DATA[1]	电机温度	DATA[1] = *(uint8_t *)&temperature)
DATA[2]	A 相电流低字节	DATA[2] = *(uint8_t *)&iA)
DATA[3]	A 相电流高字节	DATA[3] = *((uint8_t *)&iA)+1)
DATA[4]	B 相电流低字节	DATA[4] = *(uint8_t *)&iB)
DATA[5]	B 相电流高字节	DATA[5] = *((uint8_t *)&iB)+1)
DATA[6]	C 相电流低字节	DATA[6] = *(uint8_t *)&iC)
DATA[7]	C 相电流高字节	DATA[7] = *((uint8_t *)&iC)+1)

三、多电机命令

多电机命令需要在设定软件中打开，多电机命令和单电机命令无法同时使用

1. 多电机转矩闭环控制命令（1 帧）

用于同时向多个电机发送命令的报文格式如下：

标识符：0x280
帧格式：DATA
帧类型：标准帧
DLC：8 字节

主机发送该命令以同时控制最多 4 个电机的转矩电流输出，控制值 iqControl 为 int16_t 类型，数值范围-2000~ 2000，对应实际转矩电流范围-32A~32A（母线电流和电机的实际扭矩因不同电机而异）。

电机 ID 应当设置为#1~#4，并且不能重复，与帧数据中的 4 个转矩电流对应

数据域	说明	数据
DATA[0]	转矩电流 1 控制值低字节	DATA[0] = *(uint8_t *)&iqControl_1
DATA[1]	转矩电流 1 控制值高字节	DATA[1] = *((uint8_t *)&iqControl_1)+1
DATA[2]	转矩电流 2 控制值低字节	DATA[2] = *(uint8_t *)&iqControl_2
DATA[3]	转矩电流 2 控制值高字节	DATA[3] = *((uint8_t *)&iqControl_2)+1
DATA[4]	转矩电流 3 控制值低字节	DATA[4] = *(uint8_t *)&iqControl_3
DATA[5]	转矩电流 3 控制值高字节	DATA[5] = *((uint8_t *)&iqControl_3)+1
DATA[6]	转矩电流 4 控制值低字节	DATA[6] = *(uint8_t *)&iqControl_4
DATA[7]	转矩电流 4 控制值高字节	DATA[7] = *((uint8_t *)&iqControl_4)+1

驱动回复（1 帧）

各个电机回复命令的报文格式如下：

标识符：0x140 + ID(1~4)
帧格式：DATA
帧类型：标准帧
DLC：8 字节

各个电机根据 ID 从小到大依次回复，各个电机的回复数据与单电机转矩闭环控制命令回复数据相同

技术支持



淘宝购买

