Shanghai Lingkong Technology

Motor RS485 communication protocol

V2.36

Table of contents

Disclaimer

Thank you for purchasing the motor drive integrated control system of Shanghai Lingkong Technology Co., Ltd. Please read this statement carefully before use. Once used, it will be deemed as recognition and acceptance of all the contents of this statement. Please strictly abide by the product manual, control agreement and relevant laws, regulations, policies and guidelines to install and use the product. In the process of using the product, the user promises to be responsible for his own behavior and all the consequences arising therefrom. Lingkong Technology will not bear any legal responsibility for any losses caused by improper use, installation and modification by the user.

RS485 bus parameters

Bus interface:RS485

Baud rate (normal mode, single motor command):

9600bps

19200bps

38400bps

57600bps

115200bps(default)

230400bps

460800bps

1Mbps

2Mbps

4Mbps

Baud rate (broadcast mode, multi-motor commands):

1Mbps

2Mbps

4Mbps

Data bits:8

Parity: None

Stop bits:1

Single motor command

Up to 10000 RAID controllers can be mounted on the same bus.32To prevent bus conflicts, each driver needs to have different settings.ID,IDNumber1~32.

The master sends a single motor command frame to the bus, corresponding toIDThe motor executes after receiving the command and after a period of time (0.25msSend the sameIDThe command frame message and reply frame message format are as follows: frame command + frame data (optional), and the specific description is shown in the following table

	Data Description	Data length (byte)	illustrate
Frame Command	Frame Header	1	Frame header recognition,0x3E
	Order	1	CMD
	ID	1	1~32, corresponding to the motorID
	Data length	1	Describes the length of the data attached to the frame command, regardless of Depends on the command
	Frame command check byte	1	CMD_SUM, frame command all bytes check and, keep low8Position, high position abandonment
Frame data	data	0~100	Data accompanying the frame command
	Frame data check byte	0or1	DATA_SUM, all bytes of frame data are checked and, keep low8Position, high position abandonment

1.Reading the motor status1and error flag commands This command reads the current motor temperature, voltage and error status flag.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E

CMD[1]	Order	0x9A
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

The motor responds to the host after receiving the command. The frame data contains the following parameters:

1.Motor temperaturetemperature（int8_tType, Unit1℃/LSB).

2.Bus voltagevoltage（int16_tType, Unit0.01V/LSB).

3.Bus currentcurrent（int16_tType, Unit0.01A/LSB).

4.Motor statusmotorState(foruint8_tType, each bit represents a different motor status)

5.Error FlagerrorState(foruint8_tType, each bit represents a different motor error status)

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x9A
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (8byte, including calibration)		
DATA[0]	Motor temperature	DATA[0] = *(uint8_t *)(&temperature)
DATA[1]	Bus voltage low byte	DATA[1] = *(uint8_t *)(&voltage)
DATA[2]	Bus voltage high byte	DATA[2] = *((uint8_t *)(&voltage)+1)
DATA[3]	Bus current low byte	DATA[3] = *(uint8_t *)(¤t)
DATA[4]	Bus current high byte	DATA[4] = *((uint8_t *)(¤t)+1)
DATA[5]	Motor status byte	DATA[5] = motorState
DATA[6]	Error status byte	DATA[6] = errorState
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

Remark:

1. motorState = 0x00The motor is in the on state;motorState = 0x10The motor is off.

2. errorStateThe specific status table of each bit is as follows

errorStateBit	Status Description	0	1
0	Low voltage state	normal	Low voltage protection
1	High voltage state	normal	High voltage protection
2	Drive temperature status	normal	Driver over temperature
3	Motor temperature status	normal	Motor overheating
4	Motor current status	normal	Motor overcurrent
5	Motor short circuit state	normal	Motor short circuit
6	Stalled state	normal	Motor stall
7	Input signal status	normal	Input signal loss timeout

2.Clear motor error flag command

This command clears the current motor error state. The motor returns after receiving it.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x9B

CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

The motor responds to the host after receiving the command. Reply data and read the motor status1Same as Error Flag command (only command byteCMD[1] Different, here is0x9B）

Remark:

1.When the motor status does not return to normal, the error flag cannot be cleared.

3.Reading the motor status2Order

This command reads the current motor temperature, motor torque current (MF,MG)/motor output power (MS), speed, encoder position.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x9C
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

After receiving the command, the motor replies to the host. The frame data includes the following parameters.

1.Motor temperaturetemperature（int8_ttype,1℃/LSB).

2. MF,MGTorque current value of the motoriqorMSOutput power of the motorpower,int16_ttype.MGMotoriqThe resolution is (66/4096 A) / LSB;MFMotoriqThe resolution is (33/4096 A) / LSB.MSMotorpowerscope-1000~1000.

3.Motor speedspeed（int16_ttype,1dps/LSB).

4.Encoder valueencoder（uint16_ttype,14bitEncoder value range0~16383,15bitEncoder value range 0~32767,16bitEncoder value range0~65535).

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x9C
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (8byte, including calibration)		
DATA[0]	Motor temperature	DATA[0] = *(uint8_t *)(&temperature)
DATA[1]	Torque current low byte	DATA[1] = *(uint8_t *)(&iq)
	Output power low byte (MSseries)	DATA[1] = *(uint8_t *)(&power)
DATA[2]	Torque current high byte	DATA[2] = *((uint8_t *)(&iq)+1)
	Output power high byte (MSseries)	DATA[2] = *((uint8_t *)(&power)+1)
DATA[3]	Motor speed low byte	DATA[3] = *(uint8_t *)(&speed)
DATA[4]	Motor speed high byte	DATA[4] = *((uint8_t *)(&speed)+1)
DATA[5]	Encoder position low byte	DATA[5] = *(uint8_t *)(&encoder)
DATA[6]	Encoder position high byte	DATA[6] = *((uint8_t *)(&encoder)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

4.Reading the motor status3Order

becauseMSThe motor has no phase current sampling.MSNo effect on the motor.

This command reads the current motor temperature and3Phase current data

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x9D
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply (13byte)

After receiving the command, the motor replies to the host. The frame data contains the following data:

1.Motor temperaturetemperature（int8_ttype,1℃/LSB）

2.Phase current dataiA,iB,iC, the data type isint16_ttype,MGThe motor phase current resolution is (66/4096 A) / LSB;MF The motor phase current resolution is (33/4096 A) / LSB.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x9D
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (8byte, including calibration)		
DATA[5]	Motor temperature	DATA[5] = *(uint8_t *)(&temperature)
DATA[6]	APhase current low byte	DATA[6] = *(uint8_t *)(&iA)
DATA[7]	APhase current high byte	DATA[7] = *((uint8_t *)(& iA)+1)
DATA[8]	BPhase current low byte	DATA[8] = *(uint8_t *)(&iB)
DATA[9]	BPhase current high byte	DATA[9] = *((uint8_t *)(& iB)+1)
DATA[10]	CPhase current low byte	DATA[10] = *(uint8_t *)(&iC)
DATA[11]	CPhase current high byte	DATA[11] = *((uint8_t *)(& iC)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

5.Motor off command

Switch the motor from the on state (default state after power-on) to the off state,ledThe light changes from steady on to slow flashing. At this time, the motor can still respond to commands, but will not perform actions.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x80
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

Same as the host sends

6.Motor operation command

Switch the motor from off to on,ledThe light changes from slow flashing to constant on. At this time, you can send a control command to control the motor.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E

CMD[1]	Order	0x88
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

Same as the host sends

7.Motor stop command

Stop the motor, but do not clear the motor running status. Send the control command again to control the motor action.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x81
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

Same as what the host sends.

8.Brake status control and read commands

Control the opening and closing of the brake, or read the current status of the brake.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x8C
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x01
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
DATA[0]	Brake status control and read word Festival	0x00: The brake is powered off and the brake is activated 0x01: The brake is energized and the brake is released 0x10: Read the brake status
DATA_SUM	Data check byte	DATA[0]Byte Checksum

Drive Reply

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x8C
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x01
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
DATA[0]	Brake status byte	0x00: The brake is in the power-off state, and the brake is activated 0x01: The brake is powered on and the brake is released
DATA_SUM	Data check byte	DATA[0]Byte Checksum

9.Open-loop control command (this command is only available inMSMotor implementation) The host sends this command to control the open-loop voltage output to the motor.powerControlforint16_tType, value range -850~ 850, (motor current and torque vary from motor to motor).

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA0
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x02
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (3byte, including calibration)		
DATA[0]	Open loop control value low byte	DATA[0] = *(uint8_t *)(&powerControl)
DATA[1]	Open loop control value high byte	DATA[1] = *((uint8_t *)(&powerControl)+1)
DATA_SUM	Data check byte	DATA[0]~CMD[1]Byte Checksum

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA0).

10.Torque closed-loop control command (this command is only available inMF,MH,MGMotor implementation) The host sends this command to control the torque current output of the motor.iqControlforint16_tType, value range -2048~ 2048, corresponding toMFMotor actual torque current range -16.5A~16.5A,correspondMGMotor actual torque current range -33A~33A, the bus current and the actual torque of the motor vary from motor to motor.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA1
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x02
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (3byte, including calibration)		
DATA[0]	Torque current control value low byte	DATA[0] = *(uint8_t *)(& iqControl)
DATA[1]	Torque current control value high byte	DATA[1] = *((uint8_t *)(& iqControl)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[1]Byte Checksum

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA1).

11.Speed closed loop control command

The host sends this command to control the speed of the motor.speedControlforint32_tType, corresponding to the actual speed is 0.01dps/LSB.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA2
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x04
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (5byte, including calibration)		
DATA[0]	Motor speed low byte	DATA[0] = *(uint8_t *)(&speedControl)
DATA[1]	Motor speed	DATA[1] = *((uint8_t *)(&speedControl)+1)

DATA[2]	Motor speed	DATA[2] = *((uint8_t *)(&speedControl)+2)
DATA[3]	Motor speed high byte	DATA[3] = *((uint8_t *)(&speedControl)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[3]Byte Checksum

Remark:

1.This command is used tospeedControlBy the host computerMax SpeedValue restrictions.

2.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

3.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA2).

12.Multi-turn position closed-loop control command1

The host sends this command to control the position of the motor (multi-turn angle).angleControlforint64_tType, corresponding to the actual position0.01degree/LSB,Right now 36000represent360°, the direction of motor rotation is determined by the difference between the target position and the current position.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA3
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x08
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (9byte, including calibration)		
DATA[0]	Position control low byte1	DATA[0] = *(uint8_t *)(&angleControl)
DATA[1]	Position control byte2	DATA[1] = *((uint8_t *)(&angleControl)+1)
DATA[2]	Position control byte3	DATA[2] = *((uint8_t *)(&angleControl)+2)
DATA[3]	Position control byte4	DATA[3] = *((uint8_t *)(&angleControl)+3)
DATA[4]	Position control byte5	DATA[4] = *((uint8_t *)(&angleControl)+4)
DATA[5]	Position control byte6	DATA[5] = *((uint8_t *)(&angleControl)+5)
DATA[6]	Position control byte7	DATA[6] = *((uint8_t *)(&angleControl)+6)
DATA[7]	Position control high byte8	DATA[7] = *((uint8_t *)(&angleControl)+7)
DATA_SUM	Data check byte	DATA[0]~DATA[7]Byte Checksum

Remark:

1.The control value under this commandangleControlAffected by the host computerMax AngleValue restrictions.

2.The maximum speed of the motor under this command is determined by the upper computerMax SpeedValue restrictions.

3.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

4.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA3).

13.Multi-turn position closed-loop control command2

The host sends this command to control the position of the motor (multi-turn angle)

1.Control valueangleControlforint64_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°, motor

The direction of rotation is determined by the difference between the target position and the current position.

2.Control valuemaxSpeedThe maximum speed of the motor is limited touint32_tType, corresponding to actual speed0.01dps/LSB,Right now 36000represent360dps.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA4
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x0C
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (13byte, including calibration)		
DATA[0]	Position control low byte1	DATA[0] = *(uint8_t *)(&angleControl)
DATA[1]	Position control byte2	DATA[1] = *((uint8_t *)(&angleControl)+1)
DATA[2]	Position control byte3	DATA[2] = *((uint8_t *)(&angleControl)+2)
DATA[3]	Position control byte4	DATA[3] = *((uint8_t *)(&angleControl)+3)
DATA[4]	Position control byte5	DATA[4] = *((uint8_t *)(&angleControl)+4)
DATA[5]	Position control byte6	DATA[5] = *((uint8_t *)(&angleControl)+5)
DATA[6]	Position control byte7	DATA[6] = *((uint8_t *)(&angleControl)+6)
DATA[7]	Position control high byte8	DATA[7] = *((uint8_t *)(&angleControl)+7)
DATA[8]	Speed limit low byte1	DATA[8] = *(uint8_t *)(&maxSpeed)
DATA[9]	Speed Limit Bytes2	DATA[9] = *((uint8_t *)(&maxSpeed)+1)
DATA[10]	Speed Limit Bytes3	DATA[10] = *((uint8_t *)(&maxSpeed)+2)
DATA[11]	Speed limit high byte4	DATA[11] = *((uint8_t *)(&maxSpeed)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[11]Byte Checksum

Remark:

1.The control value under this commandangleControlAffected by the host computerMax AngleValue restrictions.

2.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

3.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA4)

14.Single-turn position closed-loop control command1

The host sends this command to control the position (single-turn angle) of the motor.

1.Control valuespinDirectionSet the direction of motor rotation.uint8_ttype,0x00Represents clockwise,0x01Counterclockwise

2.Control valueangleControlforuint16_tType, value range0~35999, corresponding to the actual position0.01degree/LSB, that is, the actual angle range0°~359.99°.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA5
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x04
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (5byte, including calibration)		
DATA[0]	Rotation direction byte	DATA[0] = spinDirection

DATA[1]	Position control byte1	DATA[1] = *(uint8_t *)(&angleControl)
DATA[2]	Position control byte2	DATA[2] = *((uint8_t *)(&angleControl)+1)
DATA[3]	NULL	0x00
DATA_SUM	Data check byte	DATA[0]~DATA[3]Byte Checksum

Remark:

1.The maximum speed of the motor under this command is determined by the upper computerMax SpeedValue restrictions.

2.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

3.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA5）

15.Single-turn position closed-loop control command2

The host sends this command to control the position (single-turn angle) of the motor.

1.Control valuespinDirectionSet the direction of motor rotation.uint8_ttype,0x00Represents clockwise,0x01Counterclockwise

2.Angle control valueangleControlforuint16_tType, value range0~35999, corresponding to the actual position0.01degree/LSB, that is, the actual angle range0°~359.99°.

3.Speed control valuemaxSpeedThe maximum speed of the motor is limited touint32_tType, corresponding to actual speed0.01dps/LSB, Right now36000represent360dps.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA6
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x08
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (9byte, including calibration)		
DATA[0]	Rotation direction byte	DATA[0] = spinDirection
DATA[1]	Position control byte1	DATA[1] = *(uint8_t *)(&angleControl)
DATA[2]	Position control byte2	DATA[2] = *((uint8_t *)(&angleControl)+1)
DATA[3]	NULL	DATA[3] = 0x00
DATA[4]	Speed limit low byte1	DATA[4] = *(uint8_t *)(&maxSpeed)
DATA[5]	Speed Limit Bytes2	DATA[5] = *((uint8_t *)(&maxSpeed)+1)
DATA[6]	Speed Limit Bytes3	DATA[6] = *((uint8_t *)(&maxSpeed)+2)
DATA[7]	Speed limit high byte4	DATA[7] = *((uint8_t *)(&maxSpeed)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[7]Byte Checksum

Remark:

1.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

2.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA6）

16.Incremental position closed loop control command1

The host sends this command to control the incremental position of the motor.

Control valueangleIncrementforint32_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°, the direction of motor rotation is determined by the sign of this parameter.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA7
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x04
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (5byte, including calibration)		
DATA[0]	Incremental position control low byte1	DATA[0] = *(uint8_t *)(&angleIncrement)
DATA[1]	Incremental position control byte2	DATA[1] = *((uint8_t *)(&angleIncrement)+1)
DATA[2]	Incremental position control byte3	DATA[2] = *((uint8_t *)(&angleIncrement)+2)
DATA[3]	Incremental position control high byte4	DATA[3] = *((uint8_t *)(&angleIncrement)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[3]Byte Checksum

Remark:

1.The maximum speed of the motor under this command is determined by the upper computerMax SpeedValue restrictions.

2.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

3.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA7)

17.Incremental position closed loop control command2

The host sends this command to control the incremental position of the motor.

1.Control valueangleIncrementforint32_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°, the direction of motor rotation is determined by the sign of this parameter.

2.Control valuemaxSpeedThe maximum speed of the motor is limited touint32_tType, corresponding to actual speed0.01dps/LSB,Right now 36000represent360dps.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xA8
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x08
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (9byte, including calibration)		
DATA[0]	Incremental position control low byte1	DATA[0] = *(uint8_t *)(&angleIncrement)
DATA[1]	Incremental position control byte2	DATA[1] = *((uint8_t *)(&angleIncrement)+1)
DATA[2]	Incremental position control byte3	DATA[2] = *((uint8_t *)(&angleIncrement)+2)
DATA[3]	Incremental position control high byte4	DATA[3] = *((uint8_t *)(&angleIncrement)+3)
DATA[4]	Speed Limit Bytes2	DATA[4] = *((uint8_t *)(&maxSpeed)+1)
DATA[5]	Speed Limit Bytes3	DATA[5] = *((uint8_t *)(&maxSpeed)+2)
DATA[6]	Speed limit high byte4	DATA[6] = *((uint8_t *)(&maxSpeed)+3)
DATA[7]	Speed Limit Bytes2	DATA[7] = *((uint8_t *)(&maxSpeed)+1)

DATA_SUM	Data check byte	DATA[0]~DATA[7]Byte Checksum

Remark:

1.In this control mode, the maximum acceleration of the motor is determined by the upper computer.Max AccelerationValue restrictions.

2.In this control mode,MF,MH,MGThe maximum torque current of the motor is determined by the upper computerMax Torque CurrentValue restrictions; MS The maximum power of the motor is determined by theMax PowerValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status2Order**Same (only command bytesCMD[1]Different, here is0xA8）

18.Read control parameter command

The host sends this command to read the current motor control parameters. The read parameters are represented by serial number.controlParamIDOK, see**Motor control parameters surface**

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xC0
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (3byte, including calibration)		
DATA[0]	Parameter number	DATA[0] = controlParamID
DATA[1]	Parameter Byte1	DATA[1] = 0x00
DATA[2]	Parameter Byte2	DATA[2] = 0x00
DATA[3]	Parameter Byte3	DATA[3] = 0x00
DATA[4]	Parameter Byte4	DATA[4] = 0x00
DATA[5]	Parameter Byte5	DATA[5] = 0x00
DATA[6]	Parameter Byte6	DATA[6] = 0x00
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

Drive Reply

The data returned by the driver contains the read parameter values. For specific parameters, see**Motor control parameter table**

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xC0
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (8byte, including calibration)		
DATA[0]	Parameter number	DATA[0] = controlParamID
DATA[1]	Parameter Byte1	DATA[1] = controlParamByte1
DATA[2]	Parameter Byte2	DATA[2] = controlParamByte2
DATA[3]	Parameter Byte3	DATA[3] = controlParamByte3
DATA[4]	Parameter Byte4	DATA[4] = controlParamByte4
DATA[5]	Parameter Byte5	DATA[5] = controlParamByte5
DATA[6]	Parameter Byte6	DATA[6] = controlParamByte6
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

19.Write control parameter command

The host sends this command to write control parameters toRAMIt takes effect immediately and becomes invalid after power failure. The written parameters and serial numberscontrolParamID

See**Motor control parameter table**

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0XC1
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (8byte, including calibration)		
DATA[0]	Parameter number	DATA[0] = controlParamID
DATA[1]	Parameter Byte1	DATA[1] = controlParamByte1
DATA[2]	Parameter Byte2	DATA[2] = controlParamByte2
DATA[3]	Parameter Byte3	DATA[3] = controlParamByte3
DATA[4]	Parameter Byte4	DATA[4] = controlParamByte4
DATA[5]	Parameter Byte5	DATA[5] = controlParamByte5
DATA[6]	Parameter Byte6	DATA[6] = controlParamByte6
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

Drive Reply

The data returned by the driver contains the parameter values after writing. For specific parameters, see**Motor control parameter table**

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0xC1
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x07
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (8byte, including calibration)		
DATA[0]	Parameter number	DATA[0] = controlParamID
DATA[1]	Parameter Byte1	DATA[1] = controlParamByte1
DATA[2]	Parameter Byte2	DATA[2] = controlParamByte2
DATA[3]	Parameter Byte3	DATA[3] = controlParamByte3
DATA[4]	Parameter Byte4	DATA[4] = controlParamByte4
DATA[5]	Parameter Byte5	DATA[5] = controlParamByte5
DATA[6]	Parameter Byte6	DATA[6] = controlParamByte6
DATA_SUM	Data check byte	DATA[0]~DATA[6]Byte Checksum

20.Read encoder command

The host sends this command to read the current position of the current encoder.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x90
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

After receiving the command, the motor replies to the host, and the reply data includes the following parameters.

1.Encoder positionencoder（uint16_tThe value type and value range are related to the encoder resolution), which is the value of the encoder original position minus the encoder zero offset.

2.Encoder home positionencoderRaw（uint16_ttype, and the value range is related to the encoder resolution).

3.Encoder zero offsetencoderOffset（uint16_tType, the value range is related to the encoder resolution), this point is the initial zero position after the motor is powered on.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x90
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x06
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (7byte, including calibration)		
DATA[0]	Encoder data low byte	DATA[0] =*(uint8_t *)(&encoder)
DATA[1]	Encoder data high byte	DATA[1] =*((uint8_t *)(&encoder)+1)
DATA[2]	Encoder original position low byte	DATA[2] =*(uint8_t *)(&encoderRaw)
DATA[3]	Encoder original position high byte	DATA[3] =*((uint8_t *)(&encoderRaw)+1)
DATA[4]	Encoder zero offset low byte	DATA[4] = *(uint8_t *)(&encoderOffset)
DATA[5]	Encoder zero bias high byte	DATA[5] = *((uint8_t *)(&encoderOffset)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[5]Byte Checksum

Remark:

1. 14bitResolution encoder value range0~16383;15bitResolution encoder value range0~32767;18bit Resolution encoder value range0~65535(Retain high16bit, omitting the low bit2bit).

twenty one.Set the current position as the motor zero command (writeROM) Set the encoder original value of the motor's current position as the initial zero point after the motor is powered on. Note:

1. This command will write the zero point into the driveFLASH, multiple writes will affect the life of the chip, and frequent use is not recommended

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x19
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

After receiving the command, the motor replies to the host, and the reply data includes the following parameters.

1.The encoder raw value of the current positionencoderZero

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x19
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x02
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (3byte, including calibration)		

DATA[0]	Zero point encoder original value low byte	DATA[0] =*(uint8_t *)(&encoderZero)
DATA[1]	Zero point encoder original value high byte	DATA[1] =*((uint8_t *)(&encoderZero)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[1]Byte Checksum

twenty two.Read multi-turn angle command

The host sends this command to read the multi-turn absolute angle value of the current motor.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x92
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

After receiving the command, the motor replies to the host. The frame data contains the following parameters:

1.Motor anglemotorAngle,forint64_tType data, positive value indicates clockwise cumulative angle, negative value indicates counterclockwise cumulative angle, unit0.01°/LSB.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x92
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x08
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (9byte, including calibration)		
DATA[0]	Angle low byte1	DATA[0] = *(uint8_t *)(&motorAngle)
DATA[1]	Angle Byte2	DATA[1] = *((uint8_t *)(& motorAngle)+1)
DATA[2]	Angle Byte3	DATA[2] = *((uint8_t *)(& motorAngle)+2)
DATA[3]	Angle Byte4	DATA[3] = *((uint8_t *)(& motorAngle)+3)
DATA[4]	Angle Byte5	DATA[4] = *((uint8_t *)(& motorAngle)+4)
DATA[5]	Angle Byte6	DATA[5] = *((uint8_t *)(& motorAngle)+5)
DATA[6]	Angle Byte7	DATA[6] = *((uint8_t *)(& motorAngle)+6)
DATA[7]	Angle high byte8	DATA[7] = *((uint8_t *)(& motorAngle)+6)
DATA_SUM	Data check byte	DATA[0]~DATA[7]Byte Checksum

twenty three.Clear motor revolution information command

The host sends this command to clear the current motor revolution information.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x93
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply (8byte)

Same as the host sends

twenty four.Read single-turn angle command

The host sends this command to read the multi-turn absolute angle value of the current motor.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x94
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum

Drive Reply

After receiving the command, the motor replies to the host. The frame data contains the following parameters:

1.Motor single turn anglecircleAngle,foruint32_tType data, starting from the encoder zero point, increases clockwise, and the value returns to zero when it reaches zero again.0,unit0.01°/LSB, value range0~36000-1.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x94
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x04
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (5byte, including calibration)		
DATA[0]	Single turn angle low byte1	DATA[0] = *(uint8_t *)(&circleAngle)
DATA[1]	Single turn angle byte2	DATA[1] = *((uint8_t *)(& circleAngle)+1)
DATA[2]	Single turn angle byte3	DATA[2] = *((uint8_t *)(& circleAngle)+2)
DATA[3]	Single turn angle high byte4	DATA[3] = *((uint8_t *)(& circleAngle)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[3]Byte Checksum

25.Set the current position to any angle (writeRAM) The host sends this command to set the current position of the motor as an arbitrary angle (multi-turn). The multi-turn angle valuemotorAngleforint32_tType data, data unit0.01°/LSB.

Frame Command (5byte, including calibration)		
CMD[0]	Frame Header	0x3E
CMD[1]	Order	0x95
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x04
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]Byte Checksum
Frame data (5byte, including calibration)		
DATA[0]	Multi-turn angle low byte1	DATA[0] = *(uint8_t *)(&motorAngle)
DATA[1]	Multi-turn angle bytes2	DATA[1] = *((uint8_t *)(& motorAngle)+1)
DATA[2]	Multi-turn angle bytes3	DATA[2] = *((uint8_t *)(& motorAngle)+2)
DATA[3]	Multi-turn angle high byte4	DATA[3] = *((uint8_t *)(& motorAngle)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[3]Byte Checksum

Drive Reply (8byte)

Same as the host sends

Appendix 1: Motor Control Parameters Table

Motor control parameter table	
Parameter numberParamID	**Control parameter description**
10 (0x0A)	Angle ringpid, contains three parameters anglePidKp(Angle ringkp,uint16_ttype) controlParamByte1 = *(uint8_t *)(& anglePidKp) controlParamByte2 = *((uint8_t *)(& anglePidKp)+1) anglePidKi(Angle ringki,uint16_ttype) controlParamByte3 = *(uint8_t *)(& anglePidKi) controlParamByte4 = *((uint8_t *)(& anglePidKi)+1) anglePidKd(Angle ringkd,uint16_ttype) controlParamByte5 = *(uint8_t *)(& anglePidKd) controlParamByte6 = *((uint8_t *)(& anglePidKd)+1)
11 (0x0B)	Speed ringpid, contains three parameters speedPidKp(Speed loopkp,uint16_ttype) controlParamByte1 = *(uint8_t *)(& speedPidKp) controlParamByte2 = *((uint8_t *)(& speedPidKp)+1) speedPidKi(Speed loopki,uint16_ttype) controlParamByte3 = *(uint8_t *)(& speedPidKi) controlParamByte4 = *((uint8_t *)(& speedPidKi)+1) speedPidKd(Speed loopkd,uint16_ttype) controlParamByte5 = *(uint8_t *)(& speedPidKd) controlParamByte6 = *((uint8_t *)(& speedPidKd)+1)
12 (0x0C)	Current looppid, contains three parameters currentPidKp(Current loopkp,uint16_ttype) controlParamByte1 = *(uint8_t *)(& currentPidKp) controlParamByte2 = *((uint8_t *)(& currentPidKp)+1) currentPidKi(Current loopki,uint16_ttype) controlParamByte3 = *(uint8_t *)(& currentPidKi) controlParamByte4 = *((uint8_t *)(& currentPidKi)+1) currentPidKd(Current loopkd,uint16_ttype) controlParamByte5 = *(uint8_t *)(& currentPidKd) controlParamByte6 = *((uint8_t *)(& currentPidKd)+1)
30(0x1E)	inputTorqueLimit(Maximum torque current,int16_ttype) controlParamByte3 = *(uint8_t *)(& inputTorqueLimit) controlParamByte4 = *((uint8_t *)(& inputTorqueLimit)+1)
32 (0x20)	inputSpeedLimit(Maximum speed,int32_ttype) controlParamByte3 = *(uint8_t *)(& inputSpeedLimit) controlParamByte4 = *((uint8_t *)(& inputSpeedLimit)+1) controlParamByte5 = *((uint8_t *)(& inputSpeedLimit)+2) controlParamByte6 = *((uint8_t *)(& inputSpeedLimit)+3)
34 (0x22)	inputAngleLimit(Angle limit,int32_ttype) controlParamByte3 = *(uint8_t *)(& inputAngleLimit) controlParamByte4 = *((uint8_t *)(& inputAngleLimit)+1)

	controlParamByte5 = *((uint8_t *)(& inputAngleLimit)+2) controlParamByte6 = *((uint8_t *)(& inputAngleLimit)+3)
36 (0x24)	inputCurrentRamp(Current slope,int32_ttype) controlParamByte3 = *(uint8_t *)(& inputCurrentRamp) controlParamByte4 = *((uint8_t *)(& inputCurrentRamp)+1) controlParamByte5 = *((uint8_t *)(& inputCurrentRamp)+2) controlParamByte6 = *((uint8_t *)(& inputCurrentRamp)+3)
38 (0x26)	inputSpeedRamp(Speed slope,int32_ttype) controlParamByte3 = *(uint8_t *)(& inputSpeedRamp) controlParamByte4 = *((uint8_t *)(& inputSpeedRamp)+1) controlParamByte5 = *((uint8_t *)(& inputSpeedRamp)+2) controlParamByte6 = *((uint8_t *)(& inputSpeedRamp)+3)