RS485 protocol

Table of contents

RS485 bus communication protocol

1. RS485 bus parameters

Bus interface: RS485

Baud rate: 9600, 19200, 38400, 57600, 115200 (default), 230400, 460800, 1Mbps, 2Mbps

Data bits: 8

Parity: None

Stop bit: 1

2. Command frame description and single motor command list

A total of up to 32 drivers (depending on the bus load) can be mounted on the same bus. In order to prevent bus conflicts, each driver needs to be set differently

The same ID, please refer to the basic settings in the previous section for details.

The master sends a control command to the driver. After receiving the command, the driver with the corresponding ID analyzes the data and selects the control method (angle closed) according to the command type

loop, speed closed loop, torque closed loop), and send a reply to the main control after a period of time (within 0.5ms).

Each control command is composed of 2 parts: frame header + data. The specific description is as follows:

Type Data Description Frame		Data length description	
Command Header Byte		1	Frame header identification, 0x3E
	Command Byte	1	CMD
	ID byte	1	1~32, corresponding to the motor
	data length byte 1 frame		ID description data length, depending on different commands.
	header check byte 1		Header checksum
frame data data data		0~60	The data attached to the command depends on the different commands.
	check byte 0 or 1		Data checkSum

The RS485 control commands currently supported by the M series motor driver are as follows:

name	command data
Read PID parameters	0x30
command to write PID Parameters to RAM	0x31
Command to write PID parameters to ROM	0x32
Command to read	0x33
acceleration command to write	0x34
acceleration to RAM	0x90
Command to read encoder command to write encoder value to	0x91
ROM As a motor zero point command, write the current position	0x19
to ROM serves as the motor	0x92
zero point command. Read	0x94
the multi-turn angle command. Read the single-turn angle	0x95
command. Clear the motor angle command (set the	0x9A
initial position of the motor). Read	0x9B
the motor status 1 and error flag command. Clear the motor error flag command. Read the motor status 2 command.	0x9C

Read motor status 3 Command motor	0x9D
off Command motor	0x80
stop Command motor	0x81
run command Open-	0x88
loop control command	0xA0
Torque closed-loop control	0xA1
command Speed closed-loop	0xA2
control command Multi-turn position closed-	0xA3
loop control command 1 Multi-turn position	0xA4
closed-loop control command 2 Single-turn	0xA5
position closed-loop control command 1	0xA6
Circle position closed-loop control command	0xA7
2 Incremental position closed-loop control	0xA8
command 1 Incremental position closed-loop control command 2 Read drive and motor model command	0x12

3. Single motor command description

(1) Read PID parameter command (5byte)

The host sends this command to read the parameters of the current motor's PID.

Data field description	data
DATA[0] header byte	0x3E
DATA[1] command byte	0x30
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x00
DATA[4] Frame header check byte	DATA[0]~DATA[3] byte checksum

driver reply (12byte)

The driver reply data contains the PI parameters of each control loop.

Data field description	data
DATA[0] header byte	0x3E
DATA[1] command byte	0x30
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x06
DATA[4] frame header check byte	DATA[0]~DATA[3] byte checksum
DATA[5] Position loop P parameter	DATA[5] = anglePidKp
DATA[6] Position loop I parameter	DATA[6] = anglePidKi
DATA[7] Speed loop P parameter	DATA[7] = speedPidKp
DATA[8] Speed loop I parameter	DATA[8] = speedPidKi
DATA[9] Torque loop P parameter	DATA[9] = iqPidKp
DATA[10] Torque loop I parameter	DATA[10] = iqPidKi
DATA[11] Frame header check byte	DATA[5]~DATA[10] byte checksum

(2) Write PID parameters to RAM command (12byte)

The host sends this command to write the PID parameters into RAM. The written parameters will become invalid after the power is turned off.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x31
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x06
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Position loop P parameter		DATA[5] = anglePidKp
DATA[6] Position loop I parameter		DATA[6] = anglePidKi
DATA[7] Speed loop P parameter		DATA[7] = speedPidKp
DATA[8] Speed loop I parameter		DATA[8] = speedPidKi
DATA[9] Torque loop P parameter		DATA[9] = iqPidKp
DATA[10] Torque loop I parameter		DATA[10] = iqPidKi
DATA[11] Frame header check byte		DATA[5]~DATA[10] byte checksum

driver reply (12byte)

The driver's reply data is consistent with the received command parameters.

(3) Write PID parameters to ROM command (12byte)

The host sends this command to write the PID parameters into RAM. It is still valid when the power is off.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x32
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x06
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Position loop P parameter		DATA[5] = anglePidKp
DATA[6] Position loop I parameter		DATA[6] = anglePidKi
DATA[7] Speed loop P parameter		DATA[7] = speedPidKp
DATA[8] Speed loop I parameter		DATA[8] = speedPidKi
DATA[9] Torque loop P parameter		DATA[9] = iqPidKp
DATA[10] Torque loop I parameter		DATA[10] = iqPidKi
DATA[11] Frame header check byte		DATA[5]~DATA[10] byte checksum

driver reply (12byte)

The driver's reply data is consistent with the received command parameters.

(4) Read acceleration command (5byte)

The host sends this command to read the current acceleration parameters of the motor.

Data field description		data
DATA[0] header byte		0x3E
DATA[1] command byte		0x33
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] Frame header check byte		DATA[0]~DATA[3] byte checksum

driver reply (10byte)

The driver reply data contains acceleration parameters. The acceleration data Accel is of type int32_t, and the unit is 1dps/s.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x33
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x04
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Acceleration low byte 1		DATA[5] = *(uint8_t *)(&Accel)
DATA[6] Acceleration byte 2		DATA[6] = *((uint8_t *)(&Accel)+1)
DATA[7] Acceleration byte 3		DATA[7] = *((uint8_t *)(&Accel)+2)
DATA[8] acceleration byte 4		DATA[8] = *((uint8_t *)(&Accel)+3)
DATA[9] Data check byte		DATA[5]~DATA[8] byte checksum

(5) Write acceleration to RAM command (10byte)

The host sends this command to write acceleration parameters into RAM. The written parameters will become invalid after power off. Acceleration data Accel is uint32_t class

Type, unit 1dps/s

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x34
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x04
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Acceleration low byte 1		DATA[5] = *(uint8_t *)(&Accel)
DATA[6] Acceleration byte 2		DATA[6] = *((uint8_t *)(&Accel)+1)
DATA[7] Acceleration byte 3		DATA[7] = *((uint8_t *)(&Accel)+2)
DATA[8] Acceleration byte 4		DATA[8] = *((uint8_t *)(&Accel)+3)
DATA[10] Data check byte driver		DATA[5]~DATA[8] byte checksum

reply (10byte)

The driver's reply data is consistent with the received command parameters.

(6) Read encoder command (5byte)

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

Data field description DATA[0]		data
header byte		0x3E
DATA[1] command byte		0x90
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] Frame header check byte		DATA[0]~DATA[3] byte checksum

driver reply (12byte)

The motor replies to the host after receiving the command, and the reply data contains the following parameters.

1. Encoder position encoder (uint16_t type, 14bit encoder value range 0~16383), which is the original position of the encoder minus

   The value after removing the encoder zero offset.

2. Encoder original position encoderRaw (uint16_t type, 14bit encoder value range 0~16383).
3. Encoder zero offset encoderOffset (uint16_t type, 14bit encoder value range 0~16383), this point is used as the motor angle

   0 o'clock in degrees.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x90
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x06
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Encoder data low byte=*(uint8_t *)(&encoder)		
DATA[6] Encoder data high byte=*((uint8_t *)(&encoder)+1)		
DATA[7] Encoder original position low byte =*(uint8_t *)(&encoderRaw)		
DATA[8] Encoder original position high byte =*((uint8_t *)(&encoderRaw)+1)		
DATA[9] encoder zero-biased low byte		= *(uint8_t *)(&encoderOffset) =
DATA[10] Encoder zero-biased high byte		*((uint8_t *)(&encoderOffset)+1)
DATA[11] Data check byte		DATA[5]~DATA[10] byte checksum

(7) Write the encoder value as the motor zero point command (8byte)

The host sends this command to set the zero offset of the encoder. The encoder value encoderOffset that needs to be written is of type uint16_t.

The numerical range of 14bit encoder is 0~16383

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x91
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x02
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] encoder zero low byte		= *(uint8_t *)(&encoderOffset) =
DATA[6] Encoder zero-biased high byte		*((uint8_t *)(&encoderOffset)+1)
DATA[7] Frame header check byte		DATA[5]~DATA[6] byte checksum

driver reply (8byte)

The data returned by the driver is the same as the command sent by the host.

(8) Write the current position to ROM as the motor zero point command (5byte)

Write the current encoder position of the motor to the ROM as the initial

position. Note:

1. **This command needs to be powered on again to take effect.**
2. **This command will write zero points into the driver's ROM. Multiple writes will affect the life of the chip. Frequent use is not recommended.**

Data field data	illustrate	
DATA[0] header byte		0x3E
DATA[1] command byte		0x19
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum

For example, the host sends the zero point setting command to driver 1# as follows (HEX)

3E 19 01 00 58

Driver reply (26byte)

This order is not open yet.

(9) Read multi-turn angle command (5byte)

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

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x92
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] Frame header check byte		DATA[0]~DATA[3] byte checksum

driver reply (14byte)

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

1. Motor angle motorAngle is int64_t type data. Positive value indicates clockwise accumulated angle, and negative value indicates counterclockwise accumulated angle.

Degree, unit 0.01°/LSB.

Data field description		data
DATA[0] header byte		0x3E
DATA[1] command byte		0x92
DATA[2]	ID byte	0x01~0x20
DATA[3] Data length byte 0x08 DATA[4] Frame		
header check byte checksum from DATA[0] to DATA[3]		
DATA[5] Angle low byte 1 DATA[6]		DATA[5] = *(uint8_t *)(&motorAngle)
Angle byte 2 DATA[7] Angle		DATA[6] = *((uint8_t *)(& motorAngle)+1)
byte 3 DATA[8] Angle byte 4		DATA[7] = *((uint8_t *)(& motorAngle)+2)
DATA[9] Angle byte 5 DATA[10]		DATA[8] = *((uint8_t *)(& motorAngle)+3)
Angle byte 6 DATA [11] Angle		DATA[9] = *((uint8_t *)(& motorAngle)+4)
byte 7 DATA[12] Angle high		DATA[10] = *((uint8_t *)(& motorAngle)+5)
byte 8 DATA[13] Data check		DATA[11] = *((uint8_t *)(& motorAngle)+6)
byte checksum from DATA[5] to		DATA[12] = *((uint8_t *)(& motorAngle)+6)
DATA[12]		

(10) Read single-turn angle command (5byte)

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

Data field description		data
DATA[0] header byte		0x3E
DATA[1] command byte		0x94
DATA[2]	ID byte	0x01~0x20
DATA[3] data length byte 0x00		
DATA[4] Checksum of frame header check bytes DATA[0] to DATA[3]		

Driver reply (10 byte)

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

1. The motor single-turn angle circleAngle is uint32_t type data. It takes the encoder zero point as the starting point and increases clockwise until it reaches

When reaching zero point, the value returns to 0, the unit is 0.01°/LSB, the value range is 0~36000*reduction ratio-1.

Data field description		data
DATA[0] header byte		0x3E
DATA[1] command byte		0x94
DATA[2]	ID byte	0x01~0x20

DATA[3] data length byte 0x04	
DATA[4] Checksum of frame header check bytes DATA[0] to DATA[3]	
DATA[5] Single circle angle low byte 1 DATA[5] = *(uint8_t *)(&circleAngle)	
DATA[6] Single turn angle byte 2	DATA[6] = *((uint8_t *)(& circleAngle)+1)
DATA[7] Single turn angle byte 3	DATA[7] = *((uint8_t *)(& circleAngle)+2)
DATA[8] Single circle angle high byte 4 DATA[8] = *((uint8_t *)(& circleAngle)+3)	
DATA[9] Checksum of data check bytes DATA[5] to DATA[8]	

(11) Read motor status 1 and error flag command (5byte)

This command reads the current temperature, voltage and error status flags of the motor

Data field description		data
DATA[0] header byte		0x3E
DATA[1] command byte		0x9A
DATA[2]	ID byte	0x01~0x20
DATA[3] data length byte 0x00		
DATA[4] Checksum of frame header check bytes DATA[0] to DATA[3]		

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, unit 1ÿ/LSB).
2. Voltage (uint16_t type, unit 0.1V/LSB).
3. Error flag errorState (of type uint8_t, each bit represents a different motor state)

data field		data
Description DATA[0] header byte		0x3E
DATA[1] command byte		0x9A
DATA[2]	ID byte	0x01~0x20
DATA[3] Data length byte 0x07 DATA[4] Frame		
header check byte checksum from DATA[0] to DATA[3]		
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)
DATA[6]	NULL	0x00
DATA[7] voltage low byte		DATA[7] = *(uint8_t *)(&voltage)
DATA[8] voltage high byte		DATA[8] = *((uint8_t *)(& voltage)+1)
DATA[9]	NULL	0x00
DATA[10]	NULL	0x00
DATA[11] Error status byte DATA[11]=errorState DATA[12] Data		
check byte DATA[5] to DATA[11] checksum		

Remark:

1. The specific status table of each errorState bit is as follows:

errorState bit state description		0	1
0	voltage	Voltage is normal	Low voltage protection
1	status		
2			
3	invalid invalid temperature status		Over temperature protection
4			
5	temperature normal invalid invalid		

6	invalid		
7	invalid		

(12) Clear motor error flag command (5byte)

This command clears the current error status of the motor, and the motor returns after receiving

Data field description		data
DATA[0] header byte		0x3E
DATA[1] command byte		0x9B
DATA[2]	ID byte	0x01~0x20
DATA[3] data length byte 0x00		
DATA[4] Checksum of frame header check bytes DATA[0] to DATA[3]		

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, unit 1ÿ/LSB).
2. Voltage (uint16_t type, unit 0.1V/LSB).
3. Error flag errorState (of type uint8_t, each bit represents a different motor state).

Data field description DATA[0]		data
header byte		0x3E
DATA[1] command byte		0x9B
DATA[2]	ID byte	0x01~0x20
DATA[3] Data length byte 0x07 DATA[4] Frame		
header check byte checksum from DATA[0] to DATA[3]		
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)
DATA[6]	NULL	0x00
DATA[7] voltage low byte		DATA[7] = *(uint8_t *)(&voltage)
DATA[8] voltage high byte		DATA[8] = *((uint8_t *)(& voltage)+1)
DATA[9]	NULL	0x00
DATA[10]	NULL	0x00
DATA[11] Error status byte DATA[11]=errorState DATA[12] Data		
check byte DATA[5] to DATA[11] checksum		

Remark:

1. When the motor status does not return to normal, the error flag cannot be cleared.
2. For the specific status of each errorState bit, please refer to the read motor status 1 and error flag command.

(13) Read motor status 2 command (5byte)

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

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x9C
DATA[2] ID byte		0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] Frame header check		Checksum from DATA[0] to DATA[3]

byte driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).

2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right

The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.

3. Motor speed speed (int16_t type, 1dps/LSB).

4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0x9C	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(14) Read motor status 3 command (5byte)

This command is only implemented on MF and MG

This command reads the current temperature and phase current

data of the motor. Data field description		data
DATA[0] Header byte		0x3E
DATA[1] command byte		0x9D
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] Frame header check byte		Checksum from DATA[0] to DATA[3]

driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB)

2. Phase A current data, the data type is int16_t type, corresponding to the actual phase current is 1A/64LSB.

3. Phase B current data, the data type is int16_t type, corresponding to the actual phase current is 1A/64LSB.

4. Phase C current data, the data type is int16_t type, corresponding to the actual phase current is 1A/64LSB.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x9D
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x07
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)

DATA[6] A phase current low byte		DATA[6] = *(uint8_t *)(&iA)
DATA[7] A phase current high byte		DATA[7] = *((uint8_t *)(& iA)+1)
DATA[8]	B phase current low byte	DATA[8] = *(uint8_t *)(&iB)
DATA[9]	B phase current high byte	DATA[9] = *((uint8_t *)(& iB)+1)
DATA[10] C phase current low byte		DATA[10] = *(uint8_t *)(&iC)
DATA[11] C phase current high byte		DATA[11] = *((uint8_t *)(& iC)+1)
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]

(15) Motor shutdown command (5byte)

Turn off the motor and clear the motor running status and previously received control instructions.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x80
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum

For example, the host sends the motor shutdown command to driver 1# as follows (HEX)

3E 80 01 00 BF

Driver reply (5byte)

Same as sent by host

(16) Motor stop command (5byte)

Stop the motor, but do not clear the motor running status and previously received control commands

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x81
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum

For example, the host sends the motor stop command to driver 1# as follows (HEX)

3E 81 01 00 C0

Driver reply (5byte)

Same as sent by host

(17) Motor running command (5byte)

Resume motor operation from motor stop command (restore control mode before stop)

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x88
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum

For example, the host sends the motor stop command to driver 1# as follows (HEX)

3E 88 01 00 C7

Driver reply (5byte)

Same as sent by host

(18) Open loop control command (8byte)

This command is only implemented on MS

The host sends this command to control the open-loop output power of the motor. The control value powerControl is of type int16_t, with a value range of -1000~1000, (motor bus current and torque vary with different motors).

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0xA0
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x02
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Output power control value low byte DATA[5] = *(uint8_t *)(&powerControl)		
DATA[6] Output power control value high byte DATA[6] = *((uint8_t *)(&powerControl)+1)		
DATA[7] Data check byte		DATA[5]~DATA[6] byte checksum

Remark:

1. The control value powerControl in this command is not limited by the Max Power value in the host computer (LK-Motor Tool).

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).

2. The motor output power value power is of int16_t type, ranging from -1000~1000.

3. Motor speed speed (int16_t type, 1dps/LSB).

4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data	
DATA[0] header byte		0x3E
DATA[1] command byte		0xA0
DATA[2] ID byte		0x01~0x20
DATA[3] data length bytes		0x07
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)
DATA[6] Output power low byte DATA[6] = *(uint8_t *)(&power)		
DATA[7] Output power high byte DATA[7] = *((uint8_t *)(&power)+1)		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)		
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)		
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)		
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)		
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]

(19) Torque closed-loop control command (8byte)

This command is only implemented on MF and MG

The host sends this command to control the torque current output of the motor. The control value iqControl is of type int16_t, with a value range of -2000~2000. Corresponding to the actual torque current range -32A~32A (the bus current and the actual torque of the motor vary with different motors).

data field	illustrate	data
DATA[0] header byte		0x3E

DATA[1] command byte	0xA1
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x02
DATA[4] frame header check byte	DATA[0]~DATA[3] byte checksum
DATA[5] Torque current control value low byte DATA[5] = *(uint8_t *)(&iqControl)	
DATA[6] Torque current control value high byte DATA[6] = *((uint8_t *)(&iqControl)+1)	
DATA[7] Data check byte	DATA[5]~DATA[6] byte checksum

remarks:

1. The control value iqControl in this command is not limited by the Max Torque Current value in the host computer (LK-Motor Tool).

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).

2. The torque current value iq of the motor. iq is of int16_t type, ranging from -2048~2048, corresponding to the actual torque current range -33A~33A. 3. Motor speed speed (int16_t type, 1dps/LSB).

4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data
DATA[0] header byte	0x3E
DATA[1] command byte	0xA1
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x07
DATA[4] frame header check byte	Checksum from DATA[0] to DATA[3]
DATA[5] Motor temperature	DATA[5] = *(uint8_t *)(&temperature)
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)	
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)	
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)	
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)	
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)	
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)	
DATA[12] Data check byte	Checksum from DATA[5] to DATA[11]

(20) Speed closed-loop control command (10byte)

The host sends this command to control the speed of the motor. The control value speedControl is of type int32_t, and the corresponding actual speed is 0.01dps/LSB

Data field description DATA[0]	data
header byte	0x3E
DATA[1] command byte	0xA2
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x04
DATA[4] frame header check byte	DATA[0]~DATA[3] byte checksum
DATA[5] motor speed low byte	DATA[5] = *(uint8_t *)(&speedControl)
DATA[6] Motor speed	DATA[6] = *((uint8_t *)(&speedControl)+1)
DATA[7] Motor speed	DATA[7] = *((uint8_t *)(&speedControl)+2)
DATA[8] Motor speed high byte	DATA[8] = *((uint8_t *)(&speedControl)+3)
DATA[9] Data check byte	DATA[5]~DATA[8] byte checksum

remarks:

1. The speedControl of the motor under this command is limited by the Max Speed value in the host computer (LK-Motor Tool).

2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).

3. In this control mode, the maximum torque current of MF and MF motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max Power value in the host computer.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).
2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.
3. Motor speed speed (int16_t type, 1dps/LSB).
4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA2	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(21) Multi-turn position closed-loop control command 1 (14byte)

The host sends this command to control the position of the motor (multi-turn angle). The control value angleControl is of type int64_t, corresponding to the actual b

Set to 0.01degree/LSB, that is, 36000 represents 360°, and the motor rotation direction is determined by the difference between the target position and the current position.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0xA3
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x08
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Position control low byte 1		DATA[5] = *(uint8_t *)(&angleControl)
DATA[6] Position control byte 2		DATA[6] = *((uint8_t *)(&angleControl)+1)
DATA[7] Position control byte 3		DATA[7] = *((uint8_t *)(&angleControl)+2)
DATA[8] Position control byte 4		DATA[8] = *((uint8_t *)(&angleControl)+3)
DATA[9] Position control byte 5		DATA[9] = *((uint8_t *)(&angleControl)+4)
DATA[10] Position control byte 6		DATA[10] = *((uint8_t *)(&angleControl)+5)
DATA[11] Position control byte 7		DATA[11] = *((uint8_t *)(&angleControl)+6)

DATA[12] Position control high byte		DATA[12] = *((uint8_t *)(&angleControl)+7)
8 DATA[13] Data check byte		DATA[5]~DATA[13] byte checksum

Remarks:

1. The control value angleControl under this command is limited by the Max Angle value in the host computer (LK-Motor Tool).
2. The maximum speed of the motor under this command is limited by the Max Speed value in the host computer (LK-Motor Tool).
3. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).
4. In this control mode, the maximum torque current of the MF and MG motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max Power value in the host computer.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).
2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.
3. Motor speed speed (int16_t type, 1dps/LSB).
4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA3	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(22) Multi-turn position closed-loop control command 2 (18byte)

The host sends this command to control the position of the motor (multi-turn angle). The control value angleControl is of type int64_t, corresponding to the actual b Set to 0.01degree/LSB, that is, 36000 represents 360°, and the motor rotation direction is determined by the difference between the target position and the current position. control value maxSpeed limits the maximum speed of motor rotation, which is uint32_t type, corresponding to the actual speed of 0.01dps/LSB, that is, 36000 represents 360dps.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0xA4
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x0C
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] Position control low byte 1		DATA[5] = *(uint8_t *)(&angleControl)
DATA[6] Position control byte 2		DATA[6] = *((uint8_t *)(&angleControl)+1)

DATA[7] Position control byte 3	DATA[7] = *((uint8_t *)(&angleControl)+2)
DATA[8] Position control byte 4	DATA[8] = *((uint8_t *)(&angleControl)+3)
DATA[9] Position control byte 5	DATA[9] = *((uint8_t *)(&angleControl)+4)
DATA[10] Position control byte 6	DATA[10] = *((uint8_t *)(&angleControl)+5)
DATA[11] Position control byte 7	DATA[11] = *((uint8_t *)(&angleControl)+6)
DATA[12] Position Control high byte 8	DATA[12] = *((uint8_t *)(&angleControl)+7)
DATA[13] Speed limit low byte 1	DATA[13] = *(uint8_t *)(&maxSpeed)
DATA[14] Speed limit byte 2	DATA[14] = *((uint8_t *)(&maxSpeed)+1)
DATA[15] Speed limit byte 3	DATA[15] = *((uint8_t *)(&maxSpeed)+2)
DATA[16] Speed limit high byte 4	DATA[16] = *((uint8_t *)(&maxSpeed)+3)
DATA[17] Data verification Byte	DATA[5]~DATA[16] byte checksum

notes:

1. The control value angleControl under this command is limited by the Max Angle value in the host computer (LK-Motor Tool).
2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).
3. In this control mode, the maximum torque current of the MF and MG motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max value in the host computer (LK-Motor Tool). Power value limit.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).
2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right
   The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.
3. Motor speed speed (int16_t type, 1dps/LSB).
4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA4	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(23) Single-turn position closed-loop control command 1 (10byte)

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

1. The control value spinDirection sets the direction of motor rotation, which is uint8_t type. 0x00 represents clockwise and 0x01 represents counterclockwise.
2. The control value angleControl is of uint16_t type, with a value range of 0~35999, and the corresponding actual position is 0.01degree/LSB, that is, the actual position
   The actual angle range is 0°~359.99°.

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0xA5
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x04
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] rotation direction byte		DATA[5] = spinDirection
DATA[6] Position control low byte 1		DATA[6] = *(uint8_t *)(&angleControl)
DATA[7] Position control high byte 2		DATA[7] = *((uint8_t *)(&angleControl)+1)
DATA[8] NULL		0x00
DATA[9] Data check byte remarks:		DATA[5]~DATA[8] byte checksum

1. The maximum speed of the motor under this command is limited by the Max Speed value in the host computer (LK-Motor Tool).
2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).
3. In this control mode, the maximum torque current of the MF and MG motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max value in the host computer (LK-Motor Tool). Power value limit.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).
2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right
   The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.
3. Motor speed speed (int16_t type, 1dps/LSB).
4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA5	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(24) Single-turn position closed-loop control command 2 (14byte)

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

1. The control value spinDirection sets the direction of motor rotation, which is uint8_t type. 0x00 represents clockwise and 0x01 represents counterclockwise.
2. The control value angleControl is of uint16_t type, with a value range of 0~35999, and the corresponding actual position is 0.01degree/LSB, that is, the actual position
   The actual angle range is 0°~359.99°.

3. The control value maxSpeed limits the maximum speed of the motor rotation, which is of uint32_t type and corresponds to the actual speed of 0.01dps/LSB, that is

36000 represents 360dps.

Data field description DATA[0]		data
header byte		0x3E
DATA[1] command byte		0xA6
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x08
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5] rotation direction byte		DATA[5] = spinDirection
DATA[6] Position control low byte 1		DATA[6] = *(uint8_t *)(&angleControl)
DATA[7] Position control high byte 2		DATA[7] = *((uint8_t *)(&angleControl)+1)
DATA[8] NULL		0x00
DATA[9] Speed limit low byte 1		DATA[9] = *(uint8_t *)(&maxSpeed)
DATA[10] Speed limit byte 2		DATA[10] = *((uint8_t *)(&maxSpeed)+1)
DATA[11] Speed limit Byte 3		DATA[11] = *((uint8_t *)(&maxSpeed)+2)
DATA[12] Speed limit high byte 4		DATA[12] = *((uint8_t *)(&maxSpeed)+3)
DATA[13] Data check byte		DATA[5]~DATA[12] byte checksum

Remarks:

1. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).
2. In this control mode, the maximum torque current of the MF and MG motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max Power value in the host computer.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).
2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right

The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.

3. Motor speed speed (int16_t type, 1dps/LSB).
4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA6	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(25) Incremental position closed-loop control command 1 (10byte)

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

1. The control value angleIncrement is of int32_t type, and the corresponding actual position is 0.01degree/LSB, that is, 36000 represents 360°.

The direction of motor rotation is determined by the sign of this parameter.

Data field description DATA[0]	data
header byte	0x3E
DATA[1] command byte	0xA7
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x03
DATA[4] frame header check byte	DATA[0]~DATA[3] byte checksum
DATA[5] Incremental position control low	DATA[5] = *(uint8_t *)(&angleIncrement)
byte 1 DATA[6] Incremental position	DATA[6] = *((uint8_t *)(&angleIncrement)+1)
control byte 2 DATA[7] Incremental	DATA[7] = *((uint8_t *)(&angleIncrement)+2)
position control byte 3 DATA[8] Incremental	DATA[8] = *((uint8_t *)(&angleIncrement)+3)
position control high byte 4	DATA[5]~DATA[8] byte checksum

DATA[9] Data check byte remarks:

1. The maximum speed of the motor under this command is limited by the Max Speed value in the host computer (LK-Motor Tool).
2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).
3. In this control mode, the maximum torque current of the MF and MG motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max value in the host computer (LK-Motor Tool). Power value limit.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).
2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.
3. Motor speed speed (int16_t type, 1dps/LSB).
4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA7	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x07	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte		DATA[6] = *(uint8_t *)(&iq)	MF MG
	Output power low byte	DATA[6] = *(uint8_t *)(&power)	MS
DATA[7] Torque current high byte		DATA[7] = *((uint8_t *)(&iq)+1)	MF MG
	Output power high byte	DATA[7] = *((uint8_t *)(&power)+1)	MS
DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)			
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)			
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)			
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)			
DATA[12] Data check byte		Checksum from DATA[5] to DATA[11]	

(26) Incremental position closed-loop control command 2 (14byte)

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

1. The control value angleIncrement is of int32_t type, and the corresponding actual position is 0.01degree/LSB, that is, 36000 represents 360°.

The direction of motor rotation is determined by the sign of this parameter.

2. The control value maxSpeed limits the maximum speed of the motor rotation, which is uint32_t type, corresponding to the actual speed 0.01dps/LSB, that is 36000 represents 360dps.

Data field description DATA[0]	data
header byte	0x3E
DATA[1] command byte	0xA8
DATA[2] ID byte	0x01~0x20
DATA[3] data length bytes	0x08
DATA[4] frame header check byte	DATA[0]~DATA[3] byte checksum
DATA[5] Incremental position control low	DATA[5] = *(uint8_t *)(&angleIncrement)
byte 1 DATA[6] Incremental position	DATA[6] = *((uint8_t *)(&angleIncrement)+1)
control byte 2 DATA[7] Incremental	DATA[7] = *((uint8_t *)(&angleIncrement)+2)
position control byte 3 DATA[8] Incremental	DATA[8] = *((uint8_t *)(&angleIncrement)+3)
position control high byte 4 DATA[9]	DATA[9] = *(uint8_t *)(&maxSpeed)
Speed limit low byte 1 DATA[10]	DATA[10] = *((uint8_t *)(&maxSpeed)+1)
Speed limit byte 2 DATA[11] Speed	DATA[11] = *((uint8_t *)(&maxSpeed)+2)
limit byte 3 DATA[12] Speed limit high	DATA[12] = *((uint8_t *)(&maxSpeed)+3)
byte 4 DATA[13] Data check byte	DATA[5]~DATA[12] byte checksum

remarks:

1. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the host computer (LK-Motor Tool).

2. In this control mode, the maximum torque current of the MF and MG motors is limited by the Max Torque Current value in the host computer (LK-Motor Tool); the maximum power of the MS motor is limited by the Max value in the host computer (LK-Motor Tool). Power value limit.

Driver reply (13byte)

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

1. Motor temperature temperature (int8_t type, 1ÿ/LSB).

2. The torque current value iq of MF and MG or the output power value power of MS. iq is int16_t type, range -2048~2048, right The actual torque current range should be -33A~33A; power is int16_t type, range -1000~1000.

3. Motor speed speed (int16_t type, 1dps/LSB).

4. Encoder position value encoder (uint16_t type, 14bit encoder value range 0~16383).

data field	Description data		
DATA[0] header byte		0x3E	
DATA[1] command byte		0xA8	
DATA[2] ID byte		0x01~0x20	
DATA[3] data length bytes		0x08	
DATA[4] frame header check byte		Checksum from DATA[0] to DATA[3]	
DATA[5] Motor temperature		DATA[5] = *(uint8_t *)(&temperature)	
DATA[6] Torque current low byte DATA[6] = *(uint8_t *)(&iq)			MF MG
	Output power low byte DATA[6] = *(uint8_t *)(&power)		MS
DATA[7] Torque current high byte DATA[7] = *((uint8_t *)(&iq)+1)			MF MG
	Output power high byte DATA[7] = *((uint8_t *)(&power)+1) MS		

DATA[8] motor speed low byte DATA[8] = *(uint8_t *)(&speed)	
DATA[9] Motor speed high byte DATA[9] = *((uint8_t *)(&speed)+1)	
DATA[10] Encoder position low byte DATA[10] = *(uint8_t *)(&encoder)	
DATA[11] Encoder position high byte DATA[11] = *((uint8_t *)(&encoder)+1)	
DATA[12] Data check byte	Checksum from DATA[5] to DATA[11]

(27) Read drive and motor model command (5byte)

This command is used to read the driver model, motor model, hardware version number and firmware version number.

data field	Description data	
DATA[0] header byte		0x3E
DATA[1] command byte		0x12
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x00
DATA[4] Frame header check byte. For		DATA[0]~DATA[3] byte checksum

example, the host sends this command to driver 1# as follows (HEX)

3E 12 01 00 51

Driver reply (48byte)

data field	illustrate	data
DATA[0] header byte		0x3E
DATA[1] command byte		0x12
DATA[2]	ID byte	0x01~0x20
DATA[3] data length bytes		0x2A
DATA[4] frame header check byte		DATA[0]~DATA[3] byte checksum
DATA[5~46] drive device information		productInfo structure
The productInfo structure of		Checksum of DATA[5]~DATA[46] bytes

DATA[47] data check byte driver device information is as follows:

```
struct productInfo {

    uint8_t driverName[20];          // driver name
    uint8_t motorName[20];           //Motor name
    uint8_t hardwareVersion;         // Driver hardware version
    uint8_t firmwareVersion;         // Firmware version
};
```

Among them, the driver hardware version displayed in the host computer = hardwareVersion/10.0f, and the firmware version = firmwareVersion/10.0f