SHANGHAI LINGKONG TECHNOLOGY CO.,LTD

RS485 PROTOCOL V2.35

Catalog

Disclaimer

Thank you for purchasing motor and driver integrated system from Shanghai LingKong Technology Co.,Ltd. Please read this statement carefully before using our product. Please read this statement carefully before using.It's considered to be the recognition and acceptance of the entire statement once using.Please ensure all the manual,relevant laws,regulations and policies are strictly observed when you operate the product.The user take responsibility for his own behavior during the process.We will not be liable for any loss caused by improper use,improper installation and modification by users.

1.RS485 bus parameter

Bus interface:RS485

Baud rate(Regular mode, single motor command):

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

Baud rate(Broadcast mode, multi-motor command):

1Mbps
2Mbps
4Mbps

Data bit:8

Odd-even check:None

Stop bit:1

2.Single motor command

Up to 32 (depending on the bus load) can be mounted on the same bus, in order to prevent bus conflicts, each driver needs to set a different ID, ID number 1~32.

The master sends a single-motor command frame to the bus, and the corresponding ID motor executes after receiving the command, and sends a reply frame with the same ID to the master after a period of time (within 0.25ms). The command frame and reply frame message format are as follows:

frame command + frame data (optional)

Type	Data description	Data length(byte)	Instructions
frame command	Head Byte	1	Frame head identification,0x3E
	Command Byte	1	CMD
	ID	1	1~32, corresponding motor ID
	Data length byte	1	Describe the length of the data attached to the frame command. Depends on different commands.
	Frame command check byte	1	CMD_SUM,The frame commands all byte checksum, keeping the lower 8 bits and discarding the high bits
frame data	Data	0~100	Data attached to the frame command
	Frame data check byte	0 or 1	DATA_SUM, frame data all byte checksum, keeping the lower 8 bits and discarding the high bits

1. Read motor state 1 and error command

The command read motor current temperature,voltage and error state.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x9A
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond:

Motor reply to the host after receiving the commands. The frame data include the following parameters:

1. Motor temperature(int8_t , 1℃/LSB).
2. Motor voltage(uint16_t , 0.01V/LSB).
3. ErrorState(uint8_t , The bits represent different motor states)

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Motor temperature	DATA[0] = *(uint8_t *)(&temperature)
DATA[1]	Voltage low byte	DATA[1] = *(uint8_t *)(&voltage)
DATA[2]	Voltage high byte	DATA[2] = *((uint8_t *)(& voltage)+1)
DATA[3]	NULL	DATA[3] = 0x00
DATA[4]	NULL	DATA[4] = 0x00
DATA[5]	Motor state byte	DATA[5] = 0x00 Motor is ON DATA[5] = 0x10 Motor is OFF
DATA[6]	Error state byte	DATA[6]=errorState
DATA_SUM	Data check byte	DATA[0]~DATA[6]byte checksum

Remark:

1. errorState:the specific state of each bit are as follows:

errorState	State description	0	1
0	Low voltage	Normal	Protect under voltage
1	High voltage	Normal	Protect over voltage
2	Driver temperature	Normal	Protect driver temperature
3	Motor temperature	Normal	Protect motor temperature
4	Motor current state	Normal	Protect over current
5	Motor short-circuit	Normal	Protect short-circuit
6	Stall	Normal	Protect stall

7	Input	Normal	Protect lost input time

2. Clear error command

The command clear motor current error state,motor respond after receiving the command.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x9B
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state1 and error state(only CMD[1] is different,it's 0x9B)

Remark:

1.Error can't be cleared unless the motor state is back to normal.

3. Read motor state 2 command

The command read current motor temperature, motor torque current(MF,MG)/ motor output power(MS), speed, encoder position.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x9C
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Motor reply to the host after receiving the commands. The frame data include the following parameters:

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

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Motor temperature	DATA[0] = *(uint8_t *)(&temperature)
DATA[1]	Torque current low byte	DATA[1] = *(uint8_t *)(&iq)
	Output power low byte	DATA[1] = *(uint8_t *)(&power)
DATA[2]	Torque current high byte	DATA[2] = *((uint8_t *)(&iq)+1)
	Output power high byte	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. Read motor state3 command

The command can only apply to MF and MG series.

The command read motor current temperature and phase current.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x9D
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond(13byte)

Motor reply to the host after receiving the commands. The frame data include the following parameters:

1. Motor temperature(int8_t,1℃/LSB)
2. A phase current data , int16_t, corresponding actual phase current is 1A/64LSB.
3. B phase current data , int16_t, corresponding actual phase current is 1A/64LSB.
4. C phase current data , int16_t, corresponding actual phase current is 1A/64LSB.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
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_SUM	Data check byte	DATA[0]~DATA[6]byte checksum

5. Motor off command

Switch the motor from the on state (the default state after power-on) to the off state, and the LED changes from always on to slow flashing. The motor can still reply to commands, but will not perform actions.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x80
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum bytes	CMD[0]~CMD[3]byte checksum

Driver respond

Same as the host sending command.

6. Motor on command

Swift the motor from off state to on state, LED changes from slow flashing to always on.We can send commands to control the motor now.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x88
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Same as the host sending command.

7. Motor stop command

Stop motor but don't clear the motor state.Send commands again can control the motor.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x81
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Same as the host sending command.

8. Control and read holding brake state command

Control holding brake or read current holding brake state.

Frame command（5byte,include check）		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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]	Holding brake control and read byte	0x00:Holding brake off, brake on. 0x01:Holding brake on, brake release. 0x10:Read holding brake state
DATA_SUM	Data check byte	DATA[0]Byte checksum

Driver respond

Frame command（5byte,include check）		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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]	Holding brake state byte	0x00:Holding brake is off, brake on. 0x01:Holding brake is on, brake release.
DATA_SUM	Data check byte	DATA[0]Byte checksum

9. Open loop control command(The command can only be applied to MS series motor)

Host send commands to control the open loop voltage. PowerControl value is int16_t, range is -850~850,(Motor current and torque is different depends on motor).

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0xA0
CMD[2]	ID	0x01~0x20
CMD[3]	Byte length	0x02
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum
Frame data(3byte,include check)		
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

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA0)

10. Torque closed loop control command(the command can only be applied to MF,MH,MG series)

Host send commands to control the torque current output, iqControl value is int16_t, range is -2048~2048, corresponding MF motor actual torque current range is -16.5A~16.5A, corresponding MG motor actual torque current range is -33A~33A. The bus current and the actual torque of the motor vary from motor to motor.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
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 checksum byte	DATA[0]~DATA[1]byte checksum

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA1)

11. Speed closed loop control command

Host send commands to control motor speed. SpeedControl value is int32_t, corresponding actual speed is 0.01dps/LSB.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
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. In this command, speed control is limited by Max speed value in LK motor tool.
2. In this control mode, max acceleration is limited by Max Acceleration in LK motor tool.
3. In this control mode,max torque current of MF/MH/MG motor is limited by Max Torque Current in LK motor tool. Max power of MS motor is limited by Max Power in LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA2)

12. Multi loop angle control command 1

Host send this command to control the position of the motor(Multi turn angle). anglecontrol is int64_t, corresponding actual position is 0.01degree/LSB, i.e 36000 corresponding to 360°, motor spin direction is determined by the difference between the target position and the current position.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Angle control low byte1	DATA[0] = *(uint8_t *)(&angleControl)
DATA[1]	Angle control byte2	DATA[1] = *((uint8_t *)(&angleControl)+1)
DATA[2]	Angle control byte3	DATA[2] = *((uint8_t *)(&angleControl)+2)
DATA[3]	Angle control byte4	DATA[3] = *((uint8_t *)(&angleControl)+3)
DATA[4]	Angle control byte5	DATA[4] = *((uint8_t *)(&angleControl)+4)
DATA[5]	Angle control byte6	DATA[5] = *((uint8_t *)(&angleControl)+5)
DATA[6]	Angle control byte7	DATA[6] = *((uint8_t *)(&angleControl)+6)
DATA[7]	Angle control high byte8	DATA[7] = *((uint8_t *)(&angleControl)+7)
DATA_SUM	Data check byte	DATA[0]~DATA[7] byte checksum

Remark:

1. In this command, angle control is limited by Max Angle value in LK motor tool.
2. In this command, max speed is limited by Max Speed in LK motor tool.
3. In this control mode, max acceleration is limited by Max Acceleration in LK motor tool.
4. In this control mode,max torque current of MF/MH/MG motor is limited by Max Torque Current in LK motor tool. Max power of MS motor is limited by Max Power in LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA3)

13. Multi loop angle control command 2

Host send this command to control the position of the motor(Multi turn angle).

1. angleControl is int64_t,corresponding actual position is 0.01degree/LSB, i.e 36000 corresponding to 360°, motor spin direction is determined by the difference between the target position and the current position.
2. maxSpeed limit the max speed, it is uint32_t, corresponding actual speed is 0.01dps/LSB, i.e 36000 corresponding to 360dps.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Angle control low byte1	DATA[0] = *(uint8_t *)(&angleControl)
DATA[1]	Angle control byte2	DATA[1] = *((uint8_t *)(&angleControl)+1)
DATA[2]	Angle control byte3	DATA[2] = *((uint8_t *)(&angleControl)+2)
DATA[3]	Angle control byte4	DATA[3] = *((uint8_t *)(&angleControl)+3)
DATA[4]	Angle control byte5	DATA[4] = *((uint8_t *)(&angleControl)+4)
DATA[5]	Angle control byte6	DATA[5] = *((uint8_t *)(&angleControl)+5)
DATA[6]	Angle control byte7	DATA[6] = *((uint8_t *)(&angleControl)+6)
DATA[7]	Angle control high byte8	DATA[7] = *((uint8_t *)(&angleControl)+7)
DATA[8]	Speed limit low byte1	DATA[8] = *(uint8_t *)(&maxSpeed)
DATA[9]	Speed limit byte2	DATA[9] = *((uint8_t *)(&maxSpeed)+1)
DATA[10]	Speed limit byte3	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. In this command, angleControl value is limited by Max Angle of LK motor tool.
2. In this control mode, motor max acceleration is limited by Max Acceleration of LK motor tool.
3. In this control mode, max torque current of MF/MH/MG motor is limited by Max torque current of LK motor tool. Max power of MS series is limited by Max Power of LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA4)

14. Single loop angle control command 1

Host send this command to control the position of the motor(single turn angle).

1. spinDirection for motor spin direction setting, is uint8_t, 0x00 means clockwise,0x01 means Counterclockwise.

2. angleControl is uint16_t, range is 0~35999,corresponding actual position is 0.01degree/LSB, i.e actual angel is 0°~359.99°.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Spin direction byte	DATA[0] = spinDirection
DATA[1]	Angle control byte1	DATA[1] = *(uint8_t *)(&angleControl)
DATA[2]	Angle 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. In this command, max speed of motor is limited by the Max speed of LK motor tool.
2. In this control mode, max acceleration of motor is limited by Max Acceleration of LK motor tool.
3. In this control mode,max torque current of MF/MH/MG motor is limited by Max torque current of LK motor tool. Max power of MS series is limited by Max Power of LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA5)

15. Single loop angle control command 2

Host send this command to control the position of the motor(single turn angle).

1.spinDirection for motor spin direction setting, is uint8_t, 0x00 means clockwise,0x01 means Counterclockwise.

2.angleControl is uint16_t, range is 0~35999,corresponding actual position is 0.01degree/LSB, i.e actual angel is 0°~359.99°.

3.maxSpeed limit the max speed, it is uint32_t, corresponding actual speed is 0.01dps/LSB, i.e 36000 corresponding to 360dps.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Spin direction byte	DATA[0] = spinDirection

DATA[1]	Angle control byte1	DATA[1] = *(uint8_t *)(&angleControl)
DATA[2]	Angle 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 byte2	DATA[5] = *((uint8_t *)(&maxSpeed)+1)
DATA[6]	Speed limit byte3	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, max acceleration of motor is limited by Max Acceleration of LK motor tool.

2.In this control mode,max torque current of MF/MH/MG motor is limited by Max torque current of LK motor tool. Max power of MS series is limited by Max Power of LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA6)

16. Increment angle control command 1

Host send commands to control the increment angle of the motor.

angleIncrement is int32_t, corresponding actual position is 0.01degree/LSB, i.e 36000 corresponding to 360° ,motor spin direction is determined by the symbol of parameter.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Incremental angle control low byte1	DATA[0] = *(uint8_t *)(&angleIncrement)
DATA[1]	Incremental angle control byte2	DATA[1] = *((uint8_t *)(&angleIncrement)+1)
DATA[2]	Incremental angle control byte3	DATA[2] = *((uint8_t *)(&angleIncrement)+2)
DATA[3]	Incremental angle control high byte4	DATA[3] = *((uint8_t *)(&angleIncrement)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[3]byte checksum

Remark:

1. In this command, max speed of motor is limited by the Max speed of LK motor tool.

2. In this control mode, max acceleration of motor is limited by Max Acceleration of LK motor tool.

3.In this control mode,max torque current of MF/MH/MG motor is limited by Max torque current of LK motor tool. Max power of MS series is limited by Max Power of LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA7)

17. Increment angle control command 2

Host send commands to control the increment angle of the motor.

1. angleIncrement is int32_t, corresponding actual position is 0.01degree/LSB, i.e 36000 corresponding to 360°,motor spin direction is determined by the symbol of parameter.

2. maxSpeed limit the max speed, it is uint32_t, corresponding actual speed is 0.01dps/LSB, i.e 36000 corresponding to 360dps.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0xA8
CMD[2]	ID	0x01~0x20
CMD[3]	Byte length	0x08
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum
Frame data(9byte,include check)		
DATA[0]	Incremental angle control low byte1	DATA[0] = *(uint8_t *)(&angleIncrement)
DATA[1]	Incremental angle control byte2	DATA[1] = *((uint8_t *)(&angleIncrement)+1)
DATA[2]	Incremental angle control byte3	DATA[2] = *((uint8_t *)(&angleIncrement)+2)
DATA[3]	Incremental angle control high byte4	DATA[3] = *((uint8_t *)(&angleIncrement)+3)
DATA[4]	Speed limit low byte1	DATA[4] = *((uint8_t *)(&maxSpeed)+1)
DATA[5]	Speed limit byte2	DATA[5] = *((uint8_t *)(&maxSpeed)+2)
DATA[6]	Speed limit byte3	DATA[6] = *((uint8_t *)(&maxSpeed)+3)
DATA[7]	Speed limit high byte4	DATA[7] = *((uint8_t *)(&maxSpeed)+1)
DATA_SUM	Data checksum	DATA[0]~DATA[7]byte check sum

Remark:

1. In this control mode, max acceleration of the motor is limited by Max Acceleration of LK motor tool.
2. In this control mode,max torque current of MF/MH/MG motor is limited by Max torque current of LK motor tool. Max power of MS series is limited by Max Power of LK motor tool.

Driver respond

Motor respond after receiving the command. The respond data is same as the command of read state2 (only CMD[1] is different,it's 0xA8)

18. Read encoder command

Host send commands to read current position of encoder.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x90
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Motor reply to the host after receiving the commands. The frame data include the following parameters:

1. encoder(uint16_t, range is relevant to encoder resolution), is encoder raw value minus encoder offset value.
2. encoderRaw(uint16_t,range is relevant to encoder resolution).
3. encoderOffset(uint16_t,range is relevant to encoder resolution). This point is the initial zero position of the motor after power on.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
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 raw low byte	DATA[2] =*(uint8_t *)(&encoderRaw)
DATA[3]	Encoder raw high byte	DATA[3] =*((uint8_t *)(&encoderRaw)+1)
DATA[4]	Encoder offset low byte	DATA[4] = *(uint8_t *)(&encoderOffset)
DATA[5]	Encoder offset high byte	DATA[5] = *((uint8_t *)(&encoderOffset)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[5]byte checksum

Remark:

1. 14bit encoder range is 0~16383; 15bit encoder range is 0~32767, 18bit encoder range is 0~65535(Keep the high 16bit and omit the low 2bit)

19. Write the current position to ROM as the motor zero command

Set the original encoder value of the current position of the motor as the initial zero point after the motor power on

Remark:

1. This command will write the zero point to the driver's FLASH, multiple writes will affect the chip life, and frequent use is not recommended.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x19
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Motor reply to the host after receiving the commands. The frame data include the following parameters:

1. Encoder raw value for current position: encoderZero

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Encodee zero low byte	DATA[0] =*(uint8_t *)(&encoderZero)
DATA[1]	Encoder zero high byte	DATA[1] =*((uint8_t *)(&encoderZero)+1)
DATA_SUM	Data check byte	DATA[0]~DATA[1]byte checksum

20. Read multi-loop angle command

Host send commands to read current motor multi loop absolute value

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x92
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Motor reply to the host after receiving the commands. The frame data include the following parameters:

1. motorAngle is int64_t, positive values represent clockwise cumulative angles, and negative values represent counterclockwise cumulative angles, unit 0.01° /LSB.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
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

21. Clear multi-loop angle command

Host send commands to clear current motor multi-loop angle info.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x93
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond(8byte)

same as host sending commands

22. Read single-loop angle command

Host send commands to read current motor single-loop angle absolute value.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x94
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Motor reply to the host after receiving the commands. The frame data include the following parameters:

1. circleAngle is motor single loop, is uint32_t,Starting at the zero point of the encoder, it increases clockwise, and the value returns to 0 when the zero point is reached again,unit is 0.01°/LSB,value range is 0~36000-1.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	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,include check)		
DATA[0]	Single loop angle low byte1	DATA[0] = *(uint8_t *)(&circleAngle)
DATA[1]	Single loop angle byte2	DATA[1] = *((uint8_t *)(& circleAngle)+1)
DATA[2]	Single loop angle byte3	DATA[2] = *((uint8_t *)(& circleAngle)+2)
DATA[3]	Single loop angle byte4	DATA[3] = *((uint8_t *)(& circleAngle)+3)
DATA_SUM	Data check byte	DATA[0]~DATA[3]byte checksum

23. Write the current position to RAM as the motor zero command

Host send commands to set motor current position as the zero position, clear the loop info at the same time.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x95
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond(8byte)

Same as the host sending commands.

24. Read PID parameters command

Host send commands to read current PID parameters. Parameter type is determined by ParamID.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x40
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,include check)		
DATA[0]	Parameter ID	DATA[0] = ParamID
DATA[1]	NULL	DATA[1] = 0x00
DATA_SUM	Data check byte	DATA[0]~DATA[1]byte checksum

Driver respond

Driver respond parameters include the one need to be read.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x40
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,include check)		
DATA[0]	Parameter ID	DATA[0] = ParamID
DATA[1]	Parameter byte1	DATA[1]
DATA[2]	Parameter byte2	DATA[2]
DATA[3]	Parameter byte3	DATA[3]
DATA[4]	Parameter byte4	DATA[4]
DATA[5]	Parameter byte5	DATA[5]
DATA[6]	Parameter byte6	DATA[6]
DATA_SUM	Data check byte	DATA[0]~DATA[6]byte checksum

Remark:

The correspondence between ParamID and specific motor setting parameters are as follows.

Motor PID parameters sheet	
ParamID	**Motor PID description**
150 (0x96)	Angle PID, include below 3 parameters anglePidKp(uint16_t) DATA[1]= *(uint8_t *)(& anglePidKp) DATA[2] = *((uint8_t *)(& anglePidKp)+1) anglePidKi(uint16_t) DATA[3]= *(uint8_t *)(& anglePidKi) DATA[4] = *((uint8_t *)(& anglePidKi)+1) anglePidKd(uint16_t) DATA[5]= *(uint8_t *)(& anglePidKd) DATA[6] = *((uint8_t *)(& anglePidKd)+1)
151 (0x97)	Speed PID, include below 3 parameters speedPidKp(uint16_t) DATA[1]= *(uint8_t *)(& speedPidKp) DATA[2] = *((uint8_t *)(& speedPidKp)+1) speedPidKi(uint16_t) DATA[3]= *(uint8_t *)(& speedPidKi) DATA[4] = *((uint8_t *)(& speedPidKi)+1) speedPidKd(uint16_t) DATA[5]= *(uint8_t *)(& speedPidKd) DATA[6] = *((uint8_t *)(& speedPidKd)+1)
152 (0x98)	Current PID,include below 3 parameters

	currentPidKp(uint16_t) DATA[1]= *(uint8_t *)(& currentPidKp) DATA[2] = *((uint8_t *)(& currentPidKp)+1) currentPidKi(uint16_t) DATA[3]= *(uint8_t *)(& currentPidKi) DATA[4] = *((uint8_t *)(& currentPidKi)+1) currentPidKd(uint16_t) DATA[5]= *(uint8_t *)(& currentPidKd) DATA[6] = *((uint8_t *)(& currentPidKd)+1)
153 (0x99)	maxTorqueCurrent(int16_t) DATA[1]= *(uint8_t *)(& maxTorqueCurrent) DATA[2] = *((uint8_t *)(& maxTorqueCurrent)+1)
154 (0x9A)	maxSpeed(int32_t) DATA[1]= *(uint8_t *)(& maxSpeed) DATA[2] = *((uint8_t *)(& maxSpeed)+1) DATA[3] = *((uint8_t *)(& maxSpeed)+2) DATA[4] = *((uint8_t *)(& maxSpeed)+3)
155 (0x9B)	4 low byte of maxAngle DATA[1]= *(uint8_t *)(& maxAngle) DATA[2] = *((uint8_t *)(& maxAngle)+1) DATA[3] = *((uint8_t *)(& maxAngle)+2) DATA[4] = *((uint8_t *)(& maxAngle)+3)
156 (0x9C)	4 high byte of maxAngle DATA[1] = *((uint8_t *)(& maxAngle)+4) DATA[2] = *((uint8_t *)(& maxAngle)+5) DATA[3] = *((uint8_t *)(& maxAngle)+6) DATA[4] = *((uint8_t *)(& maxAngle)+7)
157 (0x9D)	currentRamp(int16_t) DATA[1]= *(uint8_t *)(& currentRamp) DATA[2] = *((uint8_t *)(& currentRamp)+1)
158 (0x9E)	speedRamp(int32_t) DATA[1]= *(uint8_t *)(& speedRamp) DATA[2] = *((uint8_t *)(& speedRamp)+1) DATA[3] = *((uint8_t *)(& speedRamp)+2) DATA[4] = *((uint8_t *)(& speedRamp)+3)

25. Write PID parameter to RAM command

Host send this command to write PID parameters to RAM, parameter are invalid when power off. Take Motor PID parameters sheet for reference.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x42
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,include check)		
DATA[0]	Parameter ID	DATA[0] = ParamID
DATA[1]	Parameter byte1	DATA[1]
DATA[2]	Parameter byte2	DATA[2]
DATA[3]	Parameter byte3	DATA[3]
DATA[4]	Parameter byte4	DATA[4]
DATA[5]	Parameter byte5	DATA[5]
DATA[6]	Parameter byte6	DATA[6]
DATA_SUM	Data check byte	DATA[0]~DATA[6]byte checksum

Driver respond

Driver respond parameters are as follows:

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x42
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,include check)		
DATA[0]	Parameter ID	DATA[0] = ParamID
DATA[1]	Write successfully indication	Success:DATA[0] = 0;Failure:DATA[0] = 1
DATA_SUM	Data check byte	DATA[0]~DATA[1]byte checksum

26. Write PID parameter to ROM command

Host send this command to write PID parameters to ROM, parameters are valid when power off. Take Motor PID parameters sheet for reference.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x44
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,include check)		
DATA[0]	Parameter ID	DATA[0] = ParamID
DATA[1]	Parameter byte1	DATA[1]
DATA[2]	Parameter byte2	DATA[2]

DATA[3]	Parameter byte3	DATA[3]
DATA[4]	Parameter byte4	DATA[4]
DATA[5]	Parameter byte5	DATA[5]
DATA[6]	Parameter byte6	DATA[6]
DATA_SUM	Data check byte	DATA[0]~DATA[6]byte checksum

Driver respond

Driver respond parameters are as follows:

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x44
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,include check)		
DATA[0]	Parameter ID	DATA[0] = ParamID
DATA[1]	Write successfully indication	Success:DATA[0] = 0;Failure:DATA[0] = 1
DATA_SUM	Data check byte	DATA[0]~DATA[1]byte checksum

27. Read driver and motor type command

This command is to read driver type,motor type, hardware version and firmware version.

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x12
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	0x00
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum

Driver respond

Frame command(5byte,include check)		
CMD[0]	Frame head	0x3E
CMD[1]	Command	0x12
CMD[2]	ID	0x01~0x20
CMD[3]	Data length	58(0x3A)
CMD_SUM	Frame command checksum byte	CMD[0]~CMD[3]byte checksum
Frame data(59byte,include check)		
DATA[0~57]	Driver info data	productInfo structure
DATA_SUM	Data check byte	DATA[0]~DATA[57]byte checksum

productInfo structure of driver are as follows:

```
struct productInfo
{
```

```
    uint8_t driverName[20];          // Driver name
    uint8_t motorName[20];           // Motor name
    uint8_t motorId[12];             // Motor id
    uint16_t hardwareVersion;        // Driver hardware version
    uint16_t motorVersion;           // Motor version
    uint16_t firmwareVersion;        // Firmware version
};
```

Among them, driver hardware version in LK motor tool=hardwareVersion/10.0f,motor version=hardwareVersion/10.0f,firmware version= firmwareVersion/10.0f