Shanghai Lingkong Technology

Motor CAN bus 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.

Lingkong Technology is a trademark of Shanghai Lingkong Technology Co., Ltd. and its affiliated companies. The product names and brands appearing in this article are trademarks or registered trademarks of their respective companies.

CAN bus parameters

Bus interface:CAN

Baud rate (normal mode, single motor command):

1Mbps(default)

500kbps

250kbps

125kbps

100kbps

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

1Mbps

500kbps

Single motor command

Up to 10000 RAID controllers can be mounted on the same bus.32(depending on the bus load) drivers. To prevent bus conflicts, each driver needs to be set differently.ID.

The master sends a single motor command to the bus, corresponding toIDThe motor executes after receiving the command and after a period of time (0.25msThe command message and reply message formats are as follows:

Command message identifier:0x140 + ID (1~32)

Reply message identifier:0x180 + ID (1~32) Frame

format: data frame

Frame type: Standard frame

DLC:8byte

1.**Reading the motor status**1**and error flag commands** This command reads the current motor

temperature, voltage and error status flag.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9A
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

After receiving the command, the motor replies to the host. 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)

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9A
DATA[1]	Motor temperature	DATA[1] = *(uint8_t *)(&temperature)
DATA[2]	Bus voltage low byte	DATA[2] = *(uint8_t *)(&voltage)

DATA[3]	Bus voltage high byte	DATA[3] *((uint8_t *)(&voltage)+1)
DATA[4]	Bus current low byte	DATA[4] = *(uint8_t *)(¤t)
DATA[5]	Bus current high byte	DATA[5] = *((uint8_t *)(¤t)+1)
DATA[6]	Motor status byte	DATA[6] = motorState
DATA[7]	Error status byte	DATA[7] = errorState

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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9B
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

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 byteDATA[0] 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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9C
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

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).

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9C
DATA[1]	Motor temperature	DATA[1] = *(uint8_t *)(&temperature)
DATA[2]	Torque current low byte	DATA[2] = *(uint8_t *)(&iq)
	Output power low byte (MSseries)	DATA[2] = *(uint8_t *)(&power)
DATA[3]	Torque current high byte	DATA[3] = *((uint8_t *)(&iq)+1)
	Output power high byte (MSseries)	DATA[3] = *((uint8_t *)(&power)+1)
DATA[4]	Motor speed low byte	DATA[4] = *(uint8_t *)(&speed)
DATA[5]	Motor speed high byte	DATA[5] = *((uint8_t *)(&speed)+1)
DATA[6]	Encoder position low byte	DATA[6] = *(uint8_t *)(&encoder)
DATA[7]	Encoder position high byte	DATA[7] = *((uint8_t *)(&encoder)+1)

4.**Reading the motor status3Order**

becauseMSThe motor has no phase current sampling.MSThis command reads the

current motor temperature and3Phase current data

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9D
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x9D
DATA[1]	Motor temperature	DATA[1] = *(uint8_t *)(&temperature)
DATA[2]	APhase current low byte	DATA[2] = *(uint8_t *)(&iA)
DATA[3]	APhase current high byte	DATA[3] = *((uint8_t *)(& iA)+1)
DATA[4]	BPhase current low byte	DATA[4] = *(uint8_t *)(&iB)
DATA[5]	BPhase current high byte	DATA[5] = *((uint8_t *)(& iB)+1)
DATA[6]	CPhase current low byte	DATA[6] = *(uint8_t *)(&iC)
DATA[7]	CPhase current high byte	DATA[7] = *((uint8_t *)(& iC)+1)

5.**Motor off command**

Switch the motor from the on state (default state after power-on) to the off state, clear the number of motor revolutions and previously received control instructions, ledThe light changes from steady on to slow flashing. At this time, the motor can still respond to control commands, but will not perform actions.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x80
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

Same as what 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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x88
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

Same as what 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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x81
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply (1frame)

Same as what the host sends.

8.**Brake control and status reading commands**

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0x8C

DATA[1]	Brake status control and read bytes	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[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply (1frame)

Data Domain	illustrate	data
DATA[0]	Command Byte	0x8C
DATA[1]	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[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

9.**Open-loop control command (this command is only available in**MS**motor, invalid for other motors)** 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).

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA0
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	Open loop control value low byte	DATA[4] = *(uint8_t *)(&powerControl)
DATA[5]	Open loop control value high byte	DATA[5] = *((uint8_t *)(&powerControl)+1)
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Remark:

1.The control value in this commandpowerControlNot affected by the host computerMax PowerValue restrictions.

Drive Reply (1frame)

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status**2**Order**Same (only command bytesDATA[0]Different, here is0xA0).

10.**Torque closed-loop control command (this command is only available in**MF,MH,MG**Motor 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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA1
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00

DATA[3]	NULL	0x00
DATA[4]	Torque current control value low byte	DATA[4] = *(uint8_t *)(&iqControl)
DATA[5]	Torque current control value high byte	DATA[5] = *((uint8_t *)(&iqControl)+1)
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Remark:

1.The control value in this commandiqControlNot affected by the host computerMax Torque CurrentValue restrictions.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status**2**Order**Same (only command bytesDATA[0]Different, here is0xA1).

11.**Speed closed loop control command**

The host sends this command to control the speed of the motor with torque limit. Control valuespeedControlforint32_tType, corresponding to the actual speed is0.01dps/LSB; Control valueiqControlforint16_tType, value range -2048~ 2048,correspondMFMotor 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.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA2
DATA[1]	NULL	0x00
DATA[2]	Torque current control value low byte	DATA[2] = *(uint8_t *)(&iqControl)
DATA[3]	Torque current control value high byte	DATA[3] = *((uint8_t *)(&iqControl)+1)
DATA[4]	Speed control low byte	DATA[4] = *(uint8_t *)(&speedControl)
DATA[5]	Speed control	DATA[5] = *((uint8_t *)(&speedControl)+1)
DATA[6]	Speed control	DATA[6] = *((uint8_t *)(&speedControl)+2)
DATA[7]	Speed control high byte	DATA[7] = *((uint8_t *)(&speedControl)+3)

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.

Drive Reply

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status**2**Order**Same (only command bytesDATA[0]Different, here is0xA2).

12.**Multi-turn position closed-loop control command**1

The host sends this command to control the position of the motor (multi-turn angle). Control valueangleControlforint32_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°, the direction of motor rotation is determined by the difference between the target position and the current position.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA3
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	Position control low byte	DATA[4] = *(uint8_t *)(&angleControl)
DATA[5]	Position Control	DATA[5] = *((uint8_t *)(&angleControl)+1)
DATA[6]	Position Control	DATA[6] = *((uint8_t *)(&angleControl)+2)
DATA[7]	Position control high byte	DATA[7] = *((uint8_t *)(&angleControl)+3)

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 status**2**Order**Same (only command bytesDATA[0]Different, here is0xA3).

13.**Multi-turn position closed-loop control command**2

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

1.Control valueangleControlforint32_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°, the direction of motor rotation is determined by the difference between the target position and the current position.

2.Control valuemaxSpeedThe maximum speed of the motor is limited touint16_tType, corresponding to actual speed1dps/LSB,Right now360 represent 360dps.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA4
DATA[1]	NULL	0x00
DATA[2]	Speed limit low byte	DATA[2] = *(uint8_t *)(&maxSpeed)
DATA[3]	Speed limit high byte	DATA[3] = *((uint8_t *)(&maxSpeed)+1)
DATA[4]	Position control low byte	DATA[4] = *(uint8_t *)(&angleControl)
DATA[5]	Position Control	DATA[5] = *((uint8_t *)(&angleControl)+1)
DATA[6]	Position Control	DATA[6] = *((uint8_t *)(&angleControl)+2)
DATA[7]	Position control high byte	DATA[7] = *((uint8_t *)(&angleControl)+3)

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 (**1**frame)**

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status**2**Order**Same (only command bytesDATA[0]Different, here is0xA4).

14.**Single-turn position closed-loop control command**1

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 valueangleControlforuint32_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA5
DATA[1]	Rotation direction byte	DATA[1] = spinDirection
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	Position control byte1 (bit0 : bit7)	DATA[4] = *(uint8_t *)(&angleControl)
DATA[5]	Position control byte2 (bit8 : bit15)	DATA[5] = *((uint8_t *)(&angleControl)+1)
DATA[6]	Position control byte3 (bit16 : bit23)	DATA[6] = *((uint8_t *)(&angleControl)+2)
DATA[7]	Position control byte4 (bit24: bit31)	DATA[7] = *((uint8_t *)(&angleControl)+3)

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 status**2**Order**Same (only command bytesDATA[0]Different, here is0xA5).

15.**Single-turn position closed-loop control command**2

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. angleControlforuint32_tType, corresponding to the actual position0.01degree/LSB,Right now36000represent360°.

3.Speed control valuemaxSpeedThe maximum speed of the motor is limited touint16_tType, corresponding to actual speed1dps/LSB, Right now360represent360dps.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA6
DATA[1]	Rotation direction byte	DATA[1] = spinDirection
DATA[2]	Speed Limit Bytes1 (bit0 : bit7)	DATA[2] = *(uint8_t *)(&maxSpeed)
DATA[3]	Speed Limit Bytes2 (bit8 : bit15)	DATA[3] = *((uint8_t *)(&maxSpeed)+1)
DATA[4]	Position control byte1 (bit0 : bit7)	DATA[4] = *(uint8_t *)(&angleControl)
DATA[5]	Position control byte2 (bit8 : bit15)	DATA[5] = *((uint8_t *)(&angleControl)+1)
DATA[6]	Position control byte3 (bit16 : bit23)	DATA[6] = *((uint8_t *)(&angleControl)+2)
DATA[7]	Position control byte4 (bit24: bit31)	DATA[7] = *((uint8_t *)(&angleControl)+3)

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 (**1**frame)**

The motor responds to the host after receiving the command. The motor responds with data and**Reading the motor status**2**Order**Same (only command bytesDATA[0]Different, here is0xA6).

16.**Incremental position closed loop control command**1

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

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA7
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	Position control low byte	DATA[4] = *(uint8_t *)(& angleIncrement)
DATA[5]	Position Control	DATA[5] = *((uint8_t *)(& angleIncrement)+1)
DATA[6]	Position Control	DATA[6] = *((uint8_t *)(& angleIncrement)+2)
DATA[7]	Position control high byte	DATA[7] = *((uint8_t *)(& angleIncrement)+3)

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 status**2**Order**Same (only command bytesDATA[0]Different, here is0xA7).

17.**Incremental position closed loop control command**2

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

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

2.Control valuemaxSpeedThe maximum speed of the motor is limited touint32_tType, corresponding to actual speed1dps/LSB,Right now360 represent 360dps.

Data Domain	illustrate	data
DATA[0]	Command Byte	0xA8
DATA[1]	NULL	0x00
DATA[2]	Speed limit low byte	DATA[2] = *(uint8_t *)(&maxSpeed)
DATA[3]	Speed limit high byte	DATA[3] = *((uint8_t *)(&maxSpeed)+1)
DATA[4]	Position control low byte	DATA[4] = *(uint8_t *)(& angleIncrement)
DATA[5]	Position Control	DATA[5] = *((uint8_t *)(& angleIncrement)+1)
DATA[6]	Position Control	DATA[6] = *((uint8_t *)(& angleIncrement)+2)
DATA[7]	Position control high byte	DATA[7] = *((uint8_t *)(& angleIncrement)+3)

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 status**2**Order**Same (only command bytesDATA[0]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**

Data Domain	illustrate	data
DATA[0]	Command Byte	0xC0
DATA[1]	Control parameter number	DATA[1] = controlParamID
DATA[2]	NULL	DATA[2] = 0x00
DATA[3]	NULL	DATA[3] = 0x00
DATA[4]	NULL	DATA[4] = 0x00
DATA[5]	NULL	DATA[5] = 0x00
DATA[6]	NULL	DATA[6] = 0x00
DATA[7]	NULL	DATA[7] = 0x00

Drive Reply

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0xC0
DATA[1]	Control parameter number	DATA[1] = controlParamID
DATA[2]	Control parameter byte1	DATA[2] = controlParamByte1
DATA[3]	Control parameter byte2	DATA[3] = controlParamByte2
DATA[4]	Control parameter byte3	DATA[4] = controlParamByte3

DATA[5]	Control parameter byte4	DATA[5] = controlParamByte4
DATA[6]	Control parameter byte5	DATA[6] = controlParamByte5
DATA[7]	Control parameter byte6	DATA[7] = controlParamByte6

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**

Data Domain	illustrate	data
DATA[0]	Command Byte	0xC1
DATA[1]	Control parameter number	DATA[1] = controlParamID
DATA[2]	Control parameter byte1	DATA[2] = controlParamByte1
DATA[3]	Control parameter byte2	DATA[3] = controlParamByte2
DATA[4]	Control parameter byte3	DATA[4] = controlParamByte3
DATA[5]	Control parameter byte4	DATA[5] = controlParamByte4
DATA[6]	Control parameter byte5	DATA[6] = controlParamByte5
DATA[7]	Control parameter byte6	DATA[7] = controlParamByte6

Drive Reply

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0xC1
DATA[1]	Control parameter number	DATA[1] = controlParamID
DATA[2]	Control parameter byte1	DATA[2] = controlParamByte1
DATA[3]	Control parameter byte2	DATA[3] = controlParamByte2
DATA[4]	Control parameter byte3	DATA[4] = controlParamByte3
DATA[5]	Control parameter byte4	DATA[5] = controlParamByte4
DATA[6]	Control parameter byte5	DATA[6] = controlParamByte5
DATA[7]	Control parameter byte6	DATA[7] = controlParamByte6

Note: For the control parameters and their serial numbers, see**Read control parameter command**Remark

20.**Read motor encoder data command**

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0x90
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

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

1.Encoder positionencoder（uint16_ttype,14bitEncoder value range0~16383), which is the value obtained by subtracting the encoder zero offset from the encoder original position.

2.Encoder home positionencoderRaw（uint16_ttype,14bitEncoder value range0~16383).

3.Encoder zero offsetencoderOffset（uint16_ttype,14bitEncoder value range0~16383), which is used as the motor angle0 point.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x90
DATA[1]	NULL	0x00
DATA[2]	Encoder position low byte	DATA[2] = *(uint8_t *)(&encoder)
DATA[3]	Encoder position high byte	DATA[3] = *((uint8_t *)(&encoder)+1)
DATA[4]	Encoder original position low byte	DATA[4] = *(uint8_t *)(&encoderRaw)
DATA[5]	Encoder original position high byte	DATA[5] = *((uint8_t *)(&encoderRaw)+1)
DATA[6]	Encoder zero offset low byte	DATA[6] = *(uint8_t *)(&encoderOffset)
DATA[7]	Encoder zero bias high byte	DATA[7] = *((uint8_t *)(&encoderOffset)+1)

twenty one.**Set the current position to**ROM**As the motor zero command** 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 take effect only after power is turned on again
2. This command will write the zero point into the driveROM, multiple writes will affect the life of the chip, and frequent use is not recommended

Data Domain	illustrate	data
DATA[0]	Command Byte	0x19
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

After receiving the command, the motor replies to the host.encoderOffsetFor setting0Bias

Data Domain	illustrate	data
DATA[0]	Command Byte	0x19
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	Encoder zero offset low byte	DATA[6] = *(uint8_t *)(&encoderOffset)
DATA[7]	Encoder zero bias high byte	DATA[7] = *((uint8_t *)(&encoderOffset)+1)

twenty two.**Read multi-turn angle command**

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0x92
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00

DATA[7]	NULL	0x00

Drive Reply

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

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0x92
DATA[1]	Angle low byte1	DATA[1] = *(uint8_t *)(&motorAngle)
DATA[2]	Angle Byte2	DATA[2] = *((uint8_t *)(& motorAngle)+1)
DATA[3]	Angle Byte3	DATA[3] = *((uint8_t *)(& motorAngle)+2)
DATA[4]	Angle Byte4	DATA[4] = *((uint8_t *)(& motorAngle)+3)
DATA[5]	Angle Byte5	DATA[5] = *((uint8_t *)(& motorAngle)+4)
DATA[6]	Angle Byte6	DATA[6] = *((uint8_t *)(& motorAngle)+5)
DATA[7]	Angle Byte7	DATA[7] = *((uint8_t *)(& motorAngle)+6)

twenty three.**Read single-turn angle command**

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0x94
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive Reply

After receiving the command, the motor replies to the host. The frame data includes 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*Reduction ratio -1.

Data Domain	illustrate	data
DATA[0]	Command Byte	0x94
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	Single turn angle low byte1	DATA[4] = *(uint8_t *)(& circleAngle)
DATA[5]	Single turn angle byte2	DATA[5] = *((uint8_t *)(& circleAngle)+1)
DATA[6]	Single turn angle byte3	DATA[6] = *((uint8_t *)(& circleAngle)+2)
DATA[7]	Single turn angle high byte4	DATA[7] = *((uint8_t *)(& circleAngle)+3)

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

Data Domain	illustrate	data
DATA[0]	Command Byte	0x95
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00

DATA[3]	NULL	0x00
DATA[4]	Multi-turn angle low byte1	DATA[4] = *(uint8_t *)(&motorAngle)
DATA[5]	Multi-turn angle bytes2	DATA[5] = *((uint8_t *)(& motorAngle)+1)
DATA[6]	Multi-turn angle bytes3	DATA[6] = *((uint8_t *)(& motorAngle)+2)
DATA[7]	Multi-turn angle high byte4	DATA[7] = *((uint8_t *)(& motorAngle)+3)

Drive Reply

The motor replies to the host after receiving the command, and the frame data is the same as that sent by the host.

Appendix 1: Motor Control Parameters Table

Motor control parameter table	
Parameter number ParamID	**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)