Shanghai yikong

technology

Motor CAN bus communication protocol

V2.36

Shanghai yikong technology 1

disclaimer

Thank you for purchasing the motor drive integrated control system of Shanghai Xiankong Technology Co., Ltd. Please read this statement carefully before using it. Once used, it will be regarded as the 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 this product. In the process of using the product, the user promises to be responsible for his own behavior and all the consequences arising therefrom. For any loss caused by improper use, installation and modification by users, Xunkong Technology will not bear legal responsibility.

Xiankong Technology is the trademark of Shanghai Xiankong Technology Co., Ltd. and its affiliated companies. The product names, brands, etc. appearing in this article are trademarks or registered trademarks of the companies to which they belong.

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 command):

1Mbps

500kbps

Single motor command

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

The master sends a single motor command to the bus, and the motor with the corresponding ID executes it after receiving the command, and sends a reply to the master after a period of time (within 0.25ms). The format of command message and reply message is 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: 8 bytes

1. **Read motor status 1 and error flag command**

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

Data domain	expl ain	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 recovery

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

1. Motor temperature(int8_t type, unit 1℃/LSB).
2. Bus voltage(int16_t type, unit 0.01V/LSB).
3. Bus current(int16_t int 16 _ t, unit 0.01A/LSB).
4. MotorState motor state (uint8_t type, each bit represents a different motor state).
5. Error flag errorState (uint8_t type, each bit represents a different motor error state).

Data domain	expl ain	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 *)(&current)
DATA[5]	Bus current high byte	DATA[5] = *((uint8_t *)(&current)+1)
DATA[6]	Motor status byte	DATA[6] = motorState
DATA[7]	Error status byte	DATA[7] = errorState

Remarks:

1. MotorState = 0x00 Motor is on; MotorState = 0x10 The motor is off.
2. The specific state table of each bit of errorState is as follows

ErrorState bit	State description	0	one
0	Low voltage state	normal	Low voltage protection
one	High voltage state	normal	High voltage protection
2	Driving temperature state	normal	Driving overtemperature
three	Motor temperature state	normal	Motor overtemperature
four	Motor current state	normal	Motor overcurrent
five	Motor short-circuit state	normal	Motor short circuit
six	Locked rotor state	normal	Motor stall
seven	Input signal state	normal	Input signal loss timeout

2. **Clear motor error flag command**

This command clears the error state of the current motor, and the motor returns after receiving it.

Data domain	**expl ain**	**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 recovery

The motor replies to the host after receiving the command. The reply data is the same as the command to read motor status 1 and error flag (only command byte DATA[0]

Different, here is 0x9B)

Note:

1. When the motor state does not return to normal, the error sign cannot be cleared.

3. **Read motor status 2 command**

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

Data domain	**expl ain**	**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 recovery

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

1. Motor temperature(int8_t type, 1℃/LSB).
2. Torque current iq of MF, MG motor or output power of MS motor, int16_t type. The iq resolution of MG motor is (66/4096 A)/LSB；; The iq resolution of MF motor is (33/4096 A)/LSB. MS motor power range -1000~1000.
3. Motor speed speed(int16_t type, 1dps/LSB).
4. Encoder value encoder(uint16_t type, the numerical range of 14bit encoder is 0~16383, and the numerical range of 15bit encoder.
   0~32767, the numerical range of 16-bit encoder is 0~65535).

Data domain	expl ain	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) DATA[2] = *(uint8_t *)(&power)
	Output power low byte (MS series)	
DATA[3]	Torque current high byte	DATA[3] = *((uint8_t *)(&iq)+1)
	Output power high byte (MS series)	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. **Read motor status 3 command**

Since the MS motor has no phase current sampling, this command has no effect on the MS motor. This command reads the temperature and 3-phase current data of the current motor.

Data domain	expl ain	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 recovery

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

1. Motor temperature(int8_t type, 1℃/LSB)
2. Phase current data iA, iB, iC, the data type is int16_t, and the phase current resolution of MG motor is (66/4096 A)/LSB；; MF
   The motor phase current resolution is (33/4096 A)/LSB.

Data domain	expl ain	data
DATA[0]	command byte	0x9D
DATA[1]	Motor temperature	DATA[1] = *(uint8_t *)(&temperature)
DATA[2]	A phase current low byte	DATA[2] = *(uint8_t *)(&iA)
DATA[3]	A phase current high byte	DATA[3] = *((uint8_t *)(& iA)+1)

DATA[4]	B phase current low byte	DATA[4] = *(uint8_t *)(&iB)
DATA[5]	B phase current high byte	DATA[5] = *((uint8_t *)(& iB)+1)
DATA[6]	C phase current low byte	DATA[6] = *(uint8_t *)(&iC)
DATA[7]	C phase current high byte	DATA[7] = *((uint8_t *)(& iC)+1)

5. **Motor shutdown command**

Switch the motor from the on state (default state after power-on) to the off state, and clear the number of turns of the motor and the previously received control instructions.

The LED changes from normally on to slowly flashing. At this time, the motor can still reply to the control command, but it will not perform the action.

Data domain	explain	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 recovery

Same as the host sending.

6. **Motor running command**

Switch the motor from off state to on state, and the LED will turn from slow flashing to constant lighting. Then send a control command to control the motor action.

Data domain	explain	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 recovery

Same as the host sending.

7. **Motor stop command**

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

Data domain	explain	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 (1 frame)

Same as the host sending.

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

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

Data domain	explain	data
DATA[0]	command byte	0x8C

DATA[1]	Brake state control and byte reading	0x00: The brake is turned off and the brake is started. 0x01: The brake is energized and the brake is released. 0x10: Read the state of the brake.
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 (1 frame)

Data domain	expl ain	data
DATA[0]	command byte	0x8C
DATA[1]	Brake status byte	0x00: The brake device is in power-off state, and the brake is started. 0x01: The brake is energized 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 implemented on MS motor, and other motors are invalid)**

The host sends this command to control the open-loop voltage output to the motor. The control value powerControl is of type int16_t, and the numerical range is -850~ 850 (the motor current and torque vary with the motor).

Data domain	expl ain	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

Remarks:

1. The control value powerControl in this command is not limited by the Max Power value in the upper computer. Drive reply (1 frame)

The motor replies to the host after receiving the command. The motor reply data is the same as the command to read motor state 2 (only the command byte DATA[0] is different, here it is 0xA0).

10. **Torque closed-loop control command (this command is only implemented on MF, MH and MG motors)**

The host computer sends this command to control the torque and current output of the motor. The control value iqControl is int16_t type, and the numerical range is -2048~ 2048, corresponding to the actual torque and current range of MF motor -16.5A~16.5A and MG motor-33a. The bus current and actual torque of the motor vary with different motors.

Data domain	expl ain	data
DATA[0]	command byte	0xA1
DATA[1]	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

Remarks:

1. The control value iqControl in this command is not limited by the Max Torque Current value in the upper computer. Drive recovery

The motor replies to the host after receiving the command. The motor recovery data is the same as the command to read motor status 2 (only the command byte DATA[0] is different, here it is 0xA1).

11. **Speed closed-loop control command**

The host sends this command to control the speed of the motor with torque limitation. The control value speedControl is of type int32_t, and the corresponding actual speed is 0.01 dps/LSB; The control value iqControl is of int16_t type, and its numerical range is -2048~ 2048, corresponding to the actual torque and current range of MF motor -16.5A~16.5A, and corresponding to the actual torque and current range of MG motor-33a. The bus current and the actual torque of motor vary with different motors.

Data domain	expl ain	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)

Remarks:

1. The speedControl of the motor under this command is limited by the Max Speed value in the upper computer.
2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer. Drive recovery

The motor replies to the host after receiving the command. The motor recovery data is the same as the command to read motor status 2 (only the command byte DATA[0] is different, here it is 0xA2).

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

The host sends this command to control the position of the motor (multi-turn angle). The control value angleControl is of type int32_t, corresponding to the actual position of 0.01degree/LSB, that is, 36000 stands for 360, and the rotation direction of the motor is determined by the difference between the target position and the current position.

Data domain	expl ain	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)

Remarks:

1. The control value angleControl under this command is limited by the Max Angle value in the upper computer.
2. The maximum speed of the motor under this command is limited by the Max Speed value in the upper computer.

3. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer.
4. In this control mode, the maximum torque current of MF, MH and MG motors is limited by the Max Torque Current value in the upper computer.

   The maximum power of MS motor is limited by the Max Power value in the upper computer.

Drive recovery

The motor replies to the host after receiving the command. The motor reply data is the same as the command to read motor state 2 (only the command byte DATA[0] is different, here it is 0xA3).

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

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

1. The control value angleControl is of type int32_t, corresponding to the actual position of 0.01degree/LSB, that is, 36000 stands for 360, and the rotation direction of the motor is determined by the difference between the target position and the current position.
2. The control value maxSpeed limits the maximum speed of motor rotation, and it is of type uint16_t, corresponding to the actual speed of 1dps/LSB, that is, 360.

   Represents 360dps.

Data domain	expl ain	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)

Remarks:

1. The control value angleControl under this command is limited by the Max Angle value in the upper computer.
2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer.
3. In this control mode, the maximum torque current of MF, MH and MG motors is limited by the Max Torque Current value in the upper computer.

   The maximum power of MS motor is limited by the Max

Power value in the upper computer. Drive reply (1 frame)

The motor replies to the host after receiving the command. The motor recovery data is the same as the command to read motor status 2 (only the command byte DATA[0] is different, here it is 0xA4).

14. **Single lap position closed loop control command 1**

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

1. The control value spinDirection sets the direction of motor rotation, which is of type uint8_t, 0x00 stands for clockwise and 0x01 stands for counterclockwise.
2. The control value angleControl is of type uint32_t, and the corresponding actual position is 0.01degree/LSB, that is, 36000 represents 360.

Data domain	expl ain	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 byte 1 (bit0: bit7)	DATA[4] = *(uint8_t *)(&angleControl)
DATA[5]	Position control byte 2 (bit8: bit15)	DATA[5] = *((uint8_t *)(&angleControl)+1)

DATA[6]	Position control byte 3 (bit16: bit23)	DATA[6] = *((uint8_t *)(&angleControl)+2)
DATA[7]	Position control byte 4 (bit24: bit31)	DATA[7] = *((uint8_t *)(&angleControl)+3)

Remarks:

1. The maximum speed of the motor under this command is limited by the Max Speed value in the upper computer.
2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer.

3. In this control mode, the maximum torque current of MF, MH and MG motors is limited by the Max Torque Current value in the upper computer.

   The maximum power of MS motor is limited by the Max Power value in the upper computer.

Drive recovery

The motor replies to the host after receiving the command. The motor recovery data is the same as the command to read motor status 2 (only the command byte DATA[0] is different, here it is 0xA5).

15. **Single lap position closed loop control command 2**

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

1. The control value spinDirection sets the direction of motor rotation, which is of type uint8_t, 0x00 stands for clockwise and 0x01 stands for counterclockwise.
2. The angleControl is of uint32_t type, and the corresponding actual position is 0.01degree/LSB, that is, 36000 stands for 360.
3. The speed control value maxSpeed limits the maximum speed of motor rotation, and it is of type uint16_t, corresponding to the actual speed of 1dps/LSB, that is, 360 represents 360dps.

Data domain	expl ain	data
DATA[0]	command byte	0xA6
DATA[1]	Rotation direction byte	DATA[1] = spinDirection
DATA[2]	Speed limit byte 1 (bit0: bit7)	DATA[2] = *(uint8_t *)(&maxSpeed)
DATA[3]	Speed limit byte 2 (bit8: bit15)	DATA[3] = *((uint8_t *)(&maxSpeed)+1)
DATA[4]	Position control byte 1 (bit0: bit7)	DATA[4] = *(uint8_t *)(&angleControl)
DATA[5]	Position control byte 2 (bit8: bit15)	DATA[5] = *((uint8_t *)(&angleControl)+1)
DATA[6]	Position control byte 3 (bit16: bit23)	DATA[6] = *((uint8_t *)(&angleControl)+2)
DATA[7]	Position control byte 4 (bit24: bit31)	DATA[7] = *((uint8_t *)(&angleControl)+3)

Remarks:

1. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer.
2. In this control mode, the maximum torque current of MF, MH and MG motors is limited by the Max Torque Current value in the upper computer.

   The maximum power of MS motor is limited by the Max

Power value in the upper computer. Drive reply (1 frame)

The motor replies to the host after receiving the command. The motor recovery data is the same as the command to read motor status 2 (only the command byte DATA[0] is different, here it is 0xA6).

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

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

The control value angleIncrement is of type int32_t, corresponding to the actual position of 0.01degree/LSB, that is, 36000 stands for 360, and the rotation direction of the motor is determined by the sign of this parameter.

Data domain	expl ain	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)

Remarks:

1. The maximum speed of the motor under this command is limited by the Max Speed value in the upper

computer.

2. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer.
3. In this control mode, the maximum torque current of MF, MH and MG motors is limited by the Max Torque Current value in the upper computer.

   The maximum power of MS motor is limited by the Max Power value in the upper computer.

Drive recovery

The motor replies to the host after receiving the command. The motor reply data is the same as the command to read motor state 2 (only the command byte DATA[0] is different, here it is 0xA7).

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

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

1. The control value angleIncrement is of type int32_t, corresponding to the actual position of 0.01degree/LSB, that is, 36000 stands for 360, and the rotation direction of the motor is determined by the sign of this parameter.
2. The control value maxSpeed limits the maximum speed of motor rotation, and it is of type uint32_t, corresponding to the actual speed of 1dps/LSB, that is, 360.
   Represents 360dps.

Data domain	expl ain	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)

Remarks:

1. In this control mode, the maximum acceleration of the motor is limited by the Max Acceleration value in the upper computer.
2. In this control mode, the maximum torque current of MF, MH and MG motors is limited by the Max Torque Current value in the upper computer.
   The maximum power of MS motor is limited by the Max Power value in the upper computer.

Drive recovery

The motor replies to the host after receiving the command. The motor reply data is the same as the command to read motor state 2 (only the command byte DATA[0] is different, here it is 0xA8).

18. **Read control parameter command**

The host sends this command to read the current control parameters from RAM, and the read control parameters are determined by the serial number. See.控制参数表

Data domain	expl ain	data
DATA[0]	command byte	0xC0
DATA[1]	Control parameter serial 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 recovery

The data that drives the reply contains the read parameter values. See for details.控制参数表

Data domain	expl ain	data
DATA[0]	command byte	0xC0

DATA[1]	Control parameter serial number	DATA[1] = controlParamID
DATA[2]	Control parameter byte 1	DATA[2] = controlParamByte1
DATA[3]	Control parameter byte 2	DATA[3] = controlParamByte2
DATA[4]	Control parameter byte 3	DATA[4] = controlParamByte3
DATA[5]	Control parameter byte 4	DATA[5] = controlParamByte4

DATA[6]	Control parameter byte 5	DATA[6] = controlParamByte5
DATA[7]	Control parameter byte 6	DATA[7] = controlParamByte6

19. **Write control parameter command**

The host sends the command to write the control parameters into RAM, which takes effect immediately and becomes invalid after power failure. See for the written control parameters and serial numbers.控制参数表

Data domain	**explain**	**data**
DATA[0]	command byte	0xC1
DATA[1]	Control parameter serial number	DATA[1] = controlParamID
DATA[2]	Control parameter byte 1	DATA[2] = controlParamByte1
DATA[3]	Control parameter byte 2	DATA[3] = controlParamByte2
DATA[4]	Control parameter byte 3	DATA[4] = controlParamByte3
DATA[5]	Control parameter byte 4	DATA[5] = controlParamByte4
DATA[6]	Control parameter byte 5	DATA[6] = controlParamByte5
DATA[7]	Control parameter byte 6	DATA[7] = controlParamByte6

Drive recovery

The written parameter values are included in the data that drives the reply. See for details.控制参数表

Data domain	**expl ain**	**data**
DATA[0]	command byte	0xC1
DATA[1]	Control parameter serial number	DATA[1] = controlParamID
DATA[2]	Control parameter byte 1	DATA[2] = controlParamByte1
DATA[3]	Control parameter byte 2	DATA[3] = controlParamByte2
DATA[4]	Control parameter byte 3	DATA[4] = controlParamByte3
DATA[5]	Control parameter byte 4	DATA[5] = controlParamByte4
DATA[6]	Control parameter byte 5	DATA[6] = controlParamByte5
DATA[7]	Control parameter byte 6	DATA[7] = controlParamByte6

Control				
controls parameter Control Parameter	scope of data Data Range	Data	Value	
Single parameter command (One Parameter Command)				
Position loop PID Position Loop PID Data type: uint16 Data Type: uint16	Data range: 0~2000 Data Range: 0~2000	DATA[1]	0x0A	
		DATA[2]	Position Loop Kp	Bit 7:0
		DATA[3]	Position Loop Kp	Bit 15:8
		DATA[4]	Position Loop Ki	Bit 7:0
		DATA[5]	Position Loop Ki	Bit 15:8
		DATA[6]	Position Loop Kd	Bit 7:0
		DATA[7]	Position Loop Kd	Bit 15:8
Speed loop PID Speed Loop PID Data type: uint16 Data Type: uint16	Data range: 0~2000 Data Range: 0~2000	DATA[1]	0x0B	
		DATA[2]	Speed Loop Kp	Bit 7:0
		DATA[3]	Speed Loop Kp	Bit 15:8
		DATA[4]	Speed Loop Ki	Bit 7:0
		DATA[5]	Speed Loop Ki	Bit 15:8
		DATA[6]	Speed Loop Kd	Bit 7:0

		DATA[7]	Speed Loop Kd	Bit 15:8
Current loop PID Current Loop PID Data type: uint16 Data Type: uint16	Data range: 0~2000 Data Range: 0~2000	DATA[1]	0x0C	
		DATA[2]	Current Loop Kp	Bit 7:0
		DATA[3]	Current Loop Kp	Bit 15:8
		DATA[4]	Current Loop Ki	Bit 7:0
		DATA[5]	Current Loop Ki	Bit 15:8
		DATA[6]	Current Loop Kd	Bit 7:0
		DATA[7]	Current Loop Kd	Bit 15:8
Torque current limit Input Torque Limit Data type: int16 Data Type: int16	Data range: 0~850 (MS motor); 0~2000 (MF, MHF, MG Motor) Data Range： 0~850 (MS series); 0~2000 (MF,	DATA[1]	0x1E	
		DATA[2]	0x00	
		DATA[3]	0x00	
		DATA[4]	Input Torque Limit	Bit 7:0
		DATA[5]	Input Torque Limit	Bit 15:8
		DATA[6]	0x00	
		DATA[7]	0x00	
Speed limit Input Speed Limit Data type: int32 Data Type: int32	Data range: 0~600000 Data Range: 0~600000	DATA[1]	0x20	
		DATA[2]	0x00	
		DATA[3]	0x00	
		DATA[4]	Input Speed Limit	Bit 7:0
		DATA[5]	Input Speed Limit	Bit 15:8
		DATA[6]	Input Speed Limit	Bit 23:16
		DATA[7]	Input Speed Limit	Bit 31:24
Upper limit of angle Input Angle Upper Limit Data type: int32 Data Type: int32 Note: the upper limit value of the Lower limit value of degree	Data range: (-231) ~ (231–1) Data Range: (-231)~(231 – 1)	DATA[1]	0x22	
		DATA[2]	0x00	
		DATA[3]	0x00	
		DATA[4]	Input Angle Upper Limit	Bit 7:0
		DATA[5]	Input Angle Upper Limit	Bit 15:8
		DATA[6]	Input Angle Upper Limit	Bit 23:16
		DATA[7]	Input Angle Upper Limit	Bit 31:24
Angle lower limit Input Angle Lower Limit Data type: int32 Data Type: int32 Note: the lower limit value of the Upper limit value of degree	Data range: (-231) ~ (231–1) Data Range: (-231)~(231 – 1)	DATA[1]	0x23	
		DATA[2]	0x00	
		DATA[3]	0x00	
		DATA[4]	Input Angle Lower Limit	Bit 7:0
		DATA[5]	Input Angle Lower Limit	Bit 15:8
		DATA[6]	Input Angle Lower Limit	Bit 23:16
		DATA[7]	Input Angle Lower Limit	Bit 31:24
Velocity slope Speed Ramp Data type: int32 Data Type: int32	Data range: 0~600000 Data Range: 0~600000	DATA[1]	0x26	
		DATA[2]	0x00	
		DATA[3]	0x00	
		DATA[4]	Speed Ramp	Bit 7:0
		DATA[5]	Speed Ramp	Bit 15:8
		DATA[6]	Speed Ramp	Bit 23:16

		DATA[7]	Speed Ramp	Bit 31:24

20. **Read motor encoder data command**

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

Data domain	expl ain	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 recovery

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

1. Encoder position encoder(uint16_t type, the numerical range of 14bit encoder is 0~16383), which is the original position of encoder minus encoder zero bias.
2. Original encoder position encoderRaw(uint16_t type, the numerical range of 14bit encoder is 0~16383).
3. Encoder zero offset (uint16 _ t type, the numerical range of 14bit encoder is 0~16383), which is taken as the zero point of motor angle.

Data domain	expl ain	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 home position high byte	DATA[5] = *((uint8_t *)(&encoderRaw)+1)
DATA[6]	Encoder zero low byte	DATA[6] = *(uint8_t *)(&encoderOffset)
DATA[7]	Encoder zero high byte	DATA[7] = *((uint8_t *)(&encoderOffset)+1)

21. **Calibrate encoder command**

This command is used to calibrate the encoder, which only needs to be calibrated once.

The calibration value is stored in ROM and is permanently valid. note:

1. This command will write calibration related parameters into ROM. Repeated writing will affect the life of the chip, so it is not recommended to use it frequently.
2. During calibration, the motor should be in no-load or light-load state.

Data domain	expl ain	data
DATA[0]	command byte	0x18
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 recovery

The motor performs calibration after receiving the command. During the calibration, the motor rotates repeatedly for several seconds, and after the calibration is completed, it returns to the host computer. The reply frame data contains the following parameters.

1. The alignment value is uint32_t data.
2. The calibration ratio value is uint16_t data, which should be around 1000. The closer to 1000, the better the calibration effect.
3. AlignState state byte, which bit4 indicates the phase sequence of calibration identification (0: forward; 1: inverted phase), bit0 indicates whether the calibration is successful (0: failed; 1: Success)

Data domain	expl ain	data
DATA[0]	command byte	0x18
DATA[1]	Calibration value byte 1	DATA[1] = *(uint 8_t *)(&AlignValue)
DATA[2]	Calibration value byte 2	DATA[2] = *((uint 8_t *)(&AlignValue) + 1)
DATA[3]	Calibration value byte 3	DATA[3] = *((uint 8_t *)(&AlignValue) + 2)
DATA[4]	Calibration value byte 4	DATA[4] = *((uint 8_t *)(&AlignValue) + 3)
DATA[5]	Calibration scale value low byte	DATA[5] = *(uint 8_t *)(&AlignRatio)
DATA[6]	Calibration scale value high byte	DATA[6] = *((uint 8_t *)(&AlignRatio) + 1)
DATA[7]	Calibration status byte	DATA[7] = AlignState

22. **Set the current position as the zero point command of the motor (write it into ROM and save it permanently).**

Set the original encoder value of the current position of the motor as
the initial zero point after the motor is powered on Note:

3. The command needs to be powered on again before it can take effect.
4. This command will write the zero point into the ROM of the driver. Repeated writing will affect the life of the chip, so it is not recommended to use it frequently.

Data domain	expl ain	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 recovery

The motor replies to the host computer after receiving the command, and the encoderOffset in the data is the set bias value of 0.

Data domain	expl ain	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 low byte	DATA[6] = *(uint8_t *)(&encoderOffset)
DATA[7]	Encoder zero high byte	DATA[7] = *((uint8_t *)(&encoderOffset)+1)

23. **Read multi-turn angle command**

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

Data domain	expl ain	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 recovery

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

4. MotorAngle is int64_t data, positive value indicates clockwise cumulative angle, negative value indicates counterclockwise cumulative angle, and the unit is 0.01°/LSB.

Data domain	explain	data
DATA[0]	command byte	0x92
DATA[1]	Angle low byte 1	DATA[1] = *(uint8_t *)(&motorAngle)
DATA[2]	Angle byte 2	DATA[2] = *((uint8_t *)(& motorAngle)+1)
DATA[3]	Angle byte 3	DATA[3] = *((uint8_t *)(& motorAngle)+2)
DATA[4]	Angle byte 4	DATA[4] = *((uint8_t *)(& motorAngle)+3)
DATA[5]	Angle byte 5	DATA[5] = *((uint8_t *)(& motorAngle)+4)
DATA[6]	Angle byte 6	DATA[6] = *((uint8_t *)(& motorAngle)+5)
DATA[7]	Angle byte 7	DATA[7] = *((uint8_t *)(& motorAngle)+6)

24. Read single lap angle command

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

Data domain	explain	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 recovery

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

1. The circleAngle of the motor is uint32_t data, which starts from the encoder zero point and increases clockwise. When it reaches the zero point again, the value returns to 0, with the unit of 0.01°/LSB, and the value range is 0~36000* reduction ratio -1.

Data domain	explain	data
DATA[0]	command byte	0x94
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	Single circle angle low byte 1	DATA[4] = *(uint8_t *)(& circleAngle)
DATA[5]	Single circle angle byte 2	DATA[5] = *((uint8_t *)(& circleAngle)+1)
DATA[6]	Single circle angle byte 3	DATA[6] = *((uint8_t *)(& circleAngle)+2)

DATA[7]	Single turn angle high byte 4	DATA[7] = *((uint8_t *)(& circleAngle)+3)

25. **Set the current position to zero (write to RAM)**

The host sends this command to set the current position of the motor as zero point and write it into RAM. After the command is sent, the motor will switch to the motor stop state. The zero point is invalid after being re-powered.

Data domain	explain	data
DATA[0]	command byte	0x95
DATA[1]	NULL	0x00
DATA[2]	NULL	0x00
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive recovery

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

26. **Read command of setting parameters**

The host sends this command to read the set parameters. See for the read set parameters.设定参数表

Data domain	explain	data
DATA[0]	command byte	0x40
DATA[1]	Parameter byte 1	DATA[1]
DATA[2]	Parameter byte 2	DATA[2]
DATA[3]	NULL	0x00
DATA[4]	NULL	0x00
DATA[5]	NULL	0x00
DATA[6]	NULL	0x00
DATA[7]	NULL	0x00

Drive recovery

The data of the drive reply includes the read set parameter values, see for details.设定参数表

Data domain	explain	data
DATA[0]	command byte	0x40
DATA[1]	Parameter byte 1	DATA[1]
DATA[2]	Parameter byte 2	DATA[2]
DATA[3]	Parameter byte 3	DATA[3]
DATA[4]	Parameter byte 4	DATA[4]
DATA[5]	Parameter byte 5	DATA[5]
DATA[6]	Parameter byte 6	DATA[6]
DATA[7]	Parameter byte 7	DATA[7]

27. **Write setting parameter command**

The host sends this command to write the setting parameters. See for the setting parameters written.设定参数表. note:

a. After writing the setting parameters, you need to send a command to save the setting parameters before writing the data into ROM.
b. You can write multiple setting parameters before sending the command to save the setting parameters.

Data domain	explain	data
DATA[0]	command byte	0x42
DATA[1]	Parameter byte 1	DATA[1]

DATA[2]	Parameter byte 2	DATA[2]
DATA[3]	Parameter byte 3	DATA[3]

DATA[4]	Parameter byte 4	DATA[4]
DATA[5]	Parameter byte 5	DATA[5]
DATA[6]	Parameter byte 6	DATA[6]
DATA[7]	Parameter byte 7	DATA[7]

Drive recovery

The written parameter values are included in the data that drives the reply. See for details.设定参数表

Data domain	**expl ain**	**data**
DATA[0]	command byte	0x42
DATA[1]	Parameter byte 1	DATA[1]
DATA[2]	Parameter byte 2	DATA[2]
DATA[3]	Parameter byte 3	DATA[3]
DATA[4]	Parameter byte 4	DATA[4]
DATA[5]	Parameter byte 5	DATA[5]
DATA[6]	Parameter byte 6	DATA[6]
DATA[7]	Parameter byte 7	DATA[7]

Set parameter table				
setup parameter Setting Parameter	scope of data Data Range	Data	Value	
Single parameter command (One Parameter Command)				
Motor ID Driver ID Data type: uint8 Data Type: uint8	Data range: 0~32 Data Range: 0~32.	DATA[1]	0x05	
		DATA[2]	0x0A	
		DATA[3]	0x00	
		DATA[4]	Driver ID	Bit 7:0
		DATA[5]	0x00	
		DATA[6]	0x00	
		DATA[7]	0x00	
Bus type Bus Type Data type: uint8 Data Type: uint8	Data range: 0~2 Data Range: 0~2 0: None. 1: RS485 2: CAN	DATA[1]	0x05	
		DATA[2]	0x0B	
		DATA[3]	0x00	
		DATA[4]	Bus Type	Bit 7:0
		DATA[5]	0x00	
		DATA[6]	0x00	
		DATA[7]	0x00	
RS485 baud rate RS485 Baudrate Data type: uint8 Data Type: uint8	Data range: 0~10 Data Range: 0~10. 0: 9600bps 1: 19200bps 2: 38400bps 3: 57600bps 4: 115200bps 5: 230400bps 6: 460800bps 7: 921600bps 8: 1000000bps 9: 2000000bps 10: 4000000bps	DATA[1]	0x05	
		DATA[2]	0x0C	
		DATA[3]	0x00	
		DATA[4]	RS485 Baudrate	Bit 7:0
		DATA[5]	0x00	
		DATA[6]	0x00	
		DATA[7]	0x00	
CAN baud rate	Data range: 0~4	DATA[1]	0x05	

CAN Baudrate	Data Range: 0~4.	DATA[2]	0x0D	
Data type: uint8 Data Type: uint8	0: 100Kbps 1: 125Kbps 2: 250Kbps 3: 500Kbps 4: 1Mbps	DATA[3]	0x00	
		DATA[4]	CAN Baudrate	Bit 7:0
		DATA[5]	0x00	
		DATA[6]	0x00	
		DATA[7]	0x00	
Maximum power Max Power Data type: int16 Data Type: int16	Data range: 0~850 (MS motor); 0~2000 (MF, MHF, MG motor) Data Range：0~850 (MS series); 0~2000 (MF, MHF, MG series)	DATA[1]	0x05	
		DATA[2]	0xB0	
		DATA[3]	0x00	
		DATA[4]	Max Power	Bit 7:0
		DATA[5]	Max Power	Bit 15:8
		DATA[6]	0x00	
		DATA[7]	0x00	
top speed Max Speed Data type: int32 Data Type: int32	Data range: 0~600000 Data Range: 0~600000	DATA[1]	0x05	
		DATA[2]	0xB2	
		DATA[3]	0x00	
		DATA[4]	Max Speed	Bit 7:0
		DATA[5]	Max Speed	Bit 15:8
		DATA[6]	Max Speed	Bit 23:16
		DATA[7]	Max Speed	Bit 31:24
Maximum angle Max Angle Data type: int32 Data Type: int32	Data range: 0 ~ (231–1) Data Range: 0~(231 – 1)	DATA[1]	0x05	
		DATA[2]	0xB4	
		DATA[3]	0x00	
		DATA[4]	Max Angle	Bit 7:0
		DATA[5]	Max Angle	Bit 15:8
		DATA[6]	Max Angle	Bit 23:16
		DATA[7]	Max Angle	Bit 31:24
Current slope Current Ramp Data type: int16 Data Type: int16	Data range: 0~30000 Data Range: 0~30000	DATA[1]	0x05	
		DATA[2]	0xBA	
		DATA[3]	0x00	
		DATA[4]	Current Ramp	Bit 7:0
		DATA[5]	Current Ramp	Bit 15:8
		DATA[6]	0x00	
		DATA[7]	0x00	
Velocity slope Speed Ramp Data type: int32 Data Type: int32	Data range: 0~600000 Data Range: 0~600000	DATA[1]	0x05	
		DATA[2]	0xBC	
		DATA[3]	0x00	
		DATA[4]	Speed Ramp	Bit 7:0
		DATA[5]	Speed Ramp	Bit 15:8
		DATA[6]	Speed Ramp	Bit 23:16
		DATA[7]	Speed Ramp	Bit 31:24

Multiple Parameter Command.

Position loop PID Position Loop PID Data type: uint16 Data Type: uint16	Data range: 0~2000 Data Range: 0~2000.	DATA[1]	0xA0	
		DATA[2]	Position Loop Kp	Bit 7:0
		DATA[3]	Position Loop Kp	Bit 15:8
		DATA[4]	Position Loop Ki	Bit 7:0
		DATA[5]	Position Loop Ki	Bit 15:8
		DATA[6]	Position Loop Kd	Bit 7:0
		DATA[7]	Position Loop Kd	Bit 15:8
Speed loop PID Speed Loop PID Data type: uint16 Data Type: uint16	Data range: 0~2000 Data Range: 0~2000.	DATA[1]	0xA4	
		DATA[2]	Speed Loop Kp	Bit 7:0
		DATA[3]	Speed Loop Kp	Bit 15:8
		DATA[4]	Speed Loop Ki	Bit 7:0
		DATA[5]	Speed Loop Ki	Bit 15:8
		DATA[6]	Speed Loop Kd	Bit 7:0
		DATA[7]	Speed Loop Kd	Bit 15:8
Current loop PID Current Loop PID Data type: uint16 Data Type: uint16	Data range: 0~2000 Data Range: 0~2000.	DATA[1]	0xA8	
		DATA[2]	Current Loop Kp	Bit 7:0
		DATA[3]	Current Loop Kp	Bit 15:8
		DATA[4]	Current Loop Ki	Bit 7:0
		DATA[5]	Current Loop Ki	Bit 15:8
		DATA[6]	Current Loop Kd	Bit 7:0
		DATA[7]	Current Loop Kd	Bit 15:8

28. **Save the setting parameter command.**

The host sends this command to save the set parameters in ROM. After saving, it needs to be powered on again or send a restart command.

Data domain	**explain**	**data**
DATA[0]	command byte	0x44
DATA[1]	fixed value	DATA[1] = 0x05
DATA[2]	fixed value	DATA[2] = 0xFA
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 recovery

The data that drives the reply contains the saved status.

Data domain	**explain**	**data**
DATA[0]	command byte	0x44
DATA[1]	fixed value	DATA[1] = 0x05
DATA[2]	Save mark	DATA[2] = 0x01 parameter saved successfully. Failed to save DATA[2] = 0x00 parameter.
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