MKS SERVO 42/57D Closed-Loop Stepper Motor

CAN User Manual V1.0.9

Note: This manual corresponds to firmware version V 1.0.9 .

MKS SERVO42D/57D_CAN Version Notes			
manual	content	Firmware	date
V1.0.0	First release version	V1.0.0	Mar-2023
V1.0.1	1. Two control modes have been added: serial open-loop and serial closed-loop.	V1.0.1	Apr-2023
	2. The operating current can be set to any value.		
	3. Speed and acceleration have been redefined, and the curve acceleration and deceleration functions have been improved.		
	4. Added a command to set the current position to 0 o'clock.		
	5. Added group CAN ID management.		
V1.0.2	1. OUT_1 port outputs a stall indicator.	V1.0.2	May-2023
	2. When using a broadcast CAN ID or a packet CAN ID, the slave device does not respond.		
V1.0.3	1. Added serial command to set single-turn zero return parameter (9 AH).	V1.0.3	Jul-2023
	2. Added a serial command lock button (8FH) .		
	3. Add IO port status reading function (34H).		
	4. The menu allows setting up to 16 slave CAN IDes .		
	5. Add left and right limit functions.		
V1.0.4	1. Added a menu or command (9BH) function to set the percentage of shutdown holding current.	V1.0.4	Sep-2023
	2. Add absolute motion based on pulse count (FEH) .		
	3. Modify the 8C instruction to add the option of not actively initiating data.		
	4. Add emergency stop command (F7H) .		
	5. Add limit port remapping instruction (9EH) .		
V1.0.5	1. Supports zero-return function for infinite position switch .	V1.0.5	May-2024
	2. Added " Hm_Mode " and " Hm_Ma " to the menu.		
	3. Add (9 4H) instruction.		
	4. The F4 and F5 commands have been optimized to eliminate step count errors.		
	5. The F5 command supports real-time updates of position and speed.		
	6. Add a reset and restart motor command (41H).		

MKS SERVO42D/57D_CAN Version Notes			
manual	content	Firmware	date
V1.0.6	1. Added a command to read the encoder's raw value (35H).	V1.0.6	Sep-2024
	2. Added En signal-triggered single-turn zeroing and position over-tolerance protection functions (9DH).		
	3. Added the function to read system parameters, see 5.1.10.		
	4. Adding the En signal can disable the stall protection function.		
	5. Add a write I/O port instruction (36H).		
V1.0.7	1. The 3BH instruction adds the ability to read the zero-state value.	V1.0.7	Mar-2025
	2. The 83H instruction adds the function of changing the current value during operation without saving it.		
	3. The F6H command adds the function of setting the runtime.		
V1.0.8	1. After restoring the default parameters, there is no need to recalibrate.	V1.0.8	Jul-2025
	2. Added the function to read system parameters, see 5.1.10.		
	3. Added multi-motor synchronous control operation function, see Part 12.		
	4. Added a location arrival threshold setting function, see 5.2.16.		
	5. Added the function to set PID parameters, see 5.3.		
V1.0.9	1. **The instruction manual has been redesigned, categorized by function, and examples have been added.**	V1.0.9	Nov-2025
	2. **Added** the function of automatically returning read-only parameters at regular intervals, see 5.1.9.		
	3. **Increase** the heart rate protection time, see 5.2.15.		
	4. Added coordinate zeroing function, see 7.2, 7.7.		
	5. Added a no-verification mode for easier testing. See 5.2.17.		

Catalog

Part 1. Product Overview

1.1 Product Introduction

The MKS SERVO42D/57D_RS485 closed-loop stepper motor is a product independently developed by the Maker Base to meet market demands. It features a pulse interface and an RS485 interface, a built-in high-efficiency FOC vector algorithm , and a high-precision encoder . Through position feedback , it effectively prevents step loss. It is suitable for applications such as small robotic arms , 3D printers , engraving machines, writing machines , automation products , and e-sports competitions .

1.2 Features

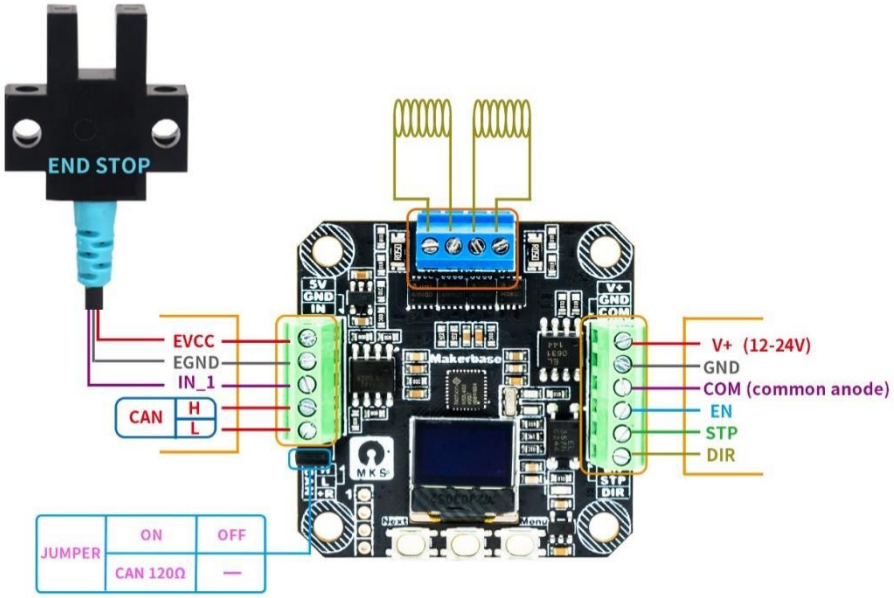
1. Supports 6 operating modes: pulse interface (open loop, closed loop, FOC mode) and serial interface (open loop, closed loop, FOC mode);
2. High-performance FOC vector control algorithm, torque, speed, and position three-loop control;
3. It supports curved acceleration and deceleration, resulting in smoother motor start-up and stopping;
4. Supports single-turn zeroing, origin switch zeroing, and mechanical limit zeroing functions;
5. Supports left and right limit functions;
6. Supports direct setting of the midnight time;
7. Supports both relative and absolute position control modes;
8. Supports any number of subdivision steps from 1 to 256;
9. Built-in 256-step subdivision interpolation algorithm, ultra-quiet motor operation, ultra-low vibration;
10. Maximum input pulse frequency: 160kHz; maximum speed: 3000RPM+.
11. Motor angle information is updated in real time (motor enabled or disabled);
12. Onboard industrial- grade high-precision 16384-line magnetic encoder;
13. 42D board has four high-power MOSFETs, 40V/20A.
14. 57D board has eight high-power MOSFETs, 30V/70A.
15. Supports CAN interface, 2048 slave CAN IDes , and supports broadcast CAN IDes and packet CAN IDes ;
16. Maximum operating current 5.2A, MOSFET continuous operating current 46A;
17. Onboard OLED display and buttons allow for easy parameter modification with automatic saving and immediate effect.
18. It has built-in stall protection function;
19. Features encoder self-calibration function;
20. One-click quick factory reset;
21. High-speed performance is stable, operation is smooth and vibration-free, and emergency stop is possible;
22. The integrated aluminum alloy casing provides effective heat dissipation and ensures more stable continuous high-current operation of the motor.
23. Provides an open-source host computer and STM32/Arduino usage examples;

1.3 Product Parameters

Product Parameters		
Motherboard model	MKS SERVO42D V1.0	MKS SERVO57D V1.2
MCU	N32L403(Cortex-M4)	N32L406(Cortex-M4)
MOSFET	AP4008QD (40V,20A)	AP30H80Q(30V,70A)
Magnetic encoder	MT6816 (14位)	
CAN transceiver	TJA1051T	
Operating voltage	12V-24V	
Working current	0-3000mA	0-5200mA
Closed-loop feedback frequency	Torque loop 20 kHz	
	Speed loop 10 kHz	
	Position loop 10 kHz	
Maximum speed	3000 RPM+	
Segmentation support	1-256 arbitrary subdivisions	
Silent/vibration	Ultra-quiet, ultra-low vibration	
Motor temperature	FOC control mode, no motor heat generation	
Pulse signal input	3.3V-24V (common anode)	
Pulse signal frequency	Up to 160 kHz	
CAN interface rate	125K/250K/500K/1M	
CAN interface CAN ID	1 broadcast CAN ID, 2047 slave CAN ID	

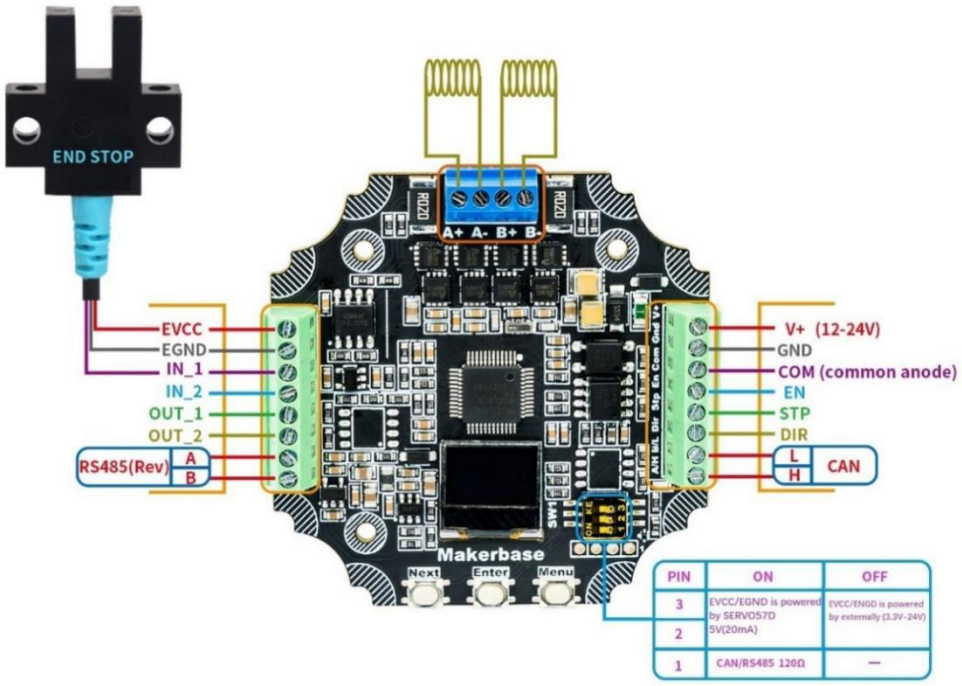
1.4 Interface Description

① SERVOC42D CAN Interface Description

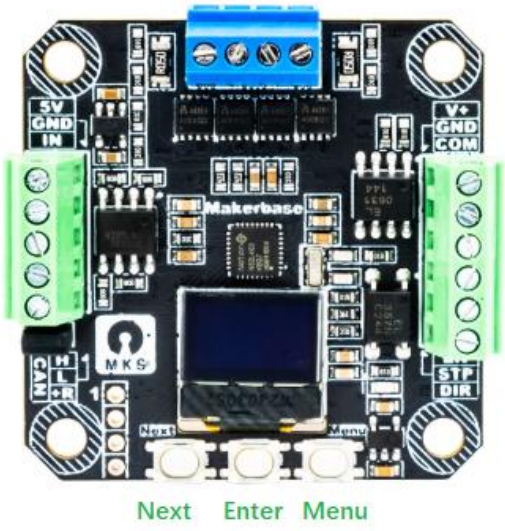


Note: EVCC/ENGD is powered by a 5.0V/20mA power supply from the SERVO42D driver board .

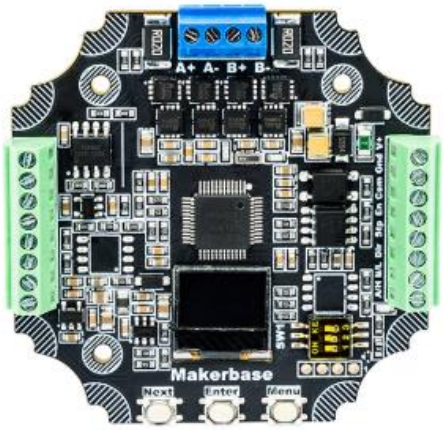
② SERVO57D CAN Interface Description



1.5 Key Operation



Next Enter Menu



Next Enter Menu

Key	Function
Next	move down
Enter	Confirm
Menu	Enter/exit parameter setting menu

1. How to View parameter

Press the “Menu” key to Enter the Menu

press the “Next” key to move to the sub-option

press the “Enter”key, then it show the value.

2. How to setting Parameter：

Press the “Menu” key to Enter the Menu

press the “Next” key to move to sub-option

press the “Enter”key, it show the value.

1.6 Screen parameter description

1. 0.0° - the angle of the motor shaft.(unit degree).
   （Note : It calculated based on the read encoder value, dynamically displayed）
2. 0.00err - the err of the motor shaft angle.
3. 0clk - the pulses have been received.

1.7 IO Port Description

Default port	Function	57D	28/35/42D
IN_1	home or left-limit	√	√
IN_2	right-limit	√	X
OUT_1	stall indication: 0-protected; 1-unprotected	√	X
OUT_2	Pulse frequency division output	√	X

Note: After the limit remapping function is enabled by the 9E instruction, the IN_1 and IN_2 input signals are invalid.

The port is defined as follows after remapping:

Remapped port	Function	57D	28/3 5 / 42D
En	home or left-limit	√	√
Dir	right-limit	√	√
Com	High level required	√	√

1.8 Limit function description

Limiting refers to technical measures that constrain the range of motion of mechanical equipment through physical or procedural means to prevent it from exceeding the preset boundary.

1. Limit function needs to be enabled :

Menu - > EndLimit Or **the serial command " 90 "** ;

2. After using the limit function for the first time or changing the limit parameters, you need to perform a limit reset once ;

Menu -> GoHome Or **the serial instruction "91"** ;

3. After the left limit switch level is triggered, the motor will no longer move to the left;

4. After the right limit switch level is triggered, the motor will no longer move to the right;

5. Limit remapping function can be enabled **(bus mode only)** .

Left limit switch - > En port

Right limit switch - > Dir port

The COM port must be connected to the corresponding high level.

1.9 Home Function Description

The stepper motor aligns its current position with the preset mechanical origin through a zero-homing operation , providing a unified reference point for subsequent positioning and path planning. If it fails to hom, the system may experience incorrect motion trajectories due to initial position deviations, affecting the accuracy and stability of the equipment. MKS motors offer four zero-return modes, allowing users to select the appropriate control method based on their specific needs.

1. **Directly set the "zero point"** to zero, which is suitable for control applications with low precision requirements. See 7.3.5.
2. **Origin switch homing** : By setting limit switches along the motor's movement path, when the motor reaches the limit switch position , a homing operation is triggered , thus calibrating the motor position to its initial state. Suitable for high-precision positioning requirements, structurally stable equipment, and safety-sensitive applications. See 7.1 , 7.4.
3. **Mechanical limit return to zero** is a zero-point return method that does not require a physical limit switch. It determines the zero point position by detecting the current change when the motor is stalled. Specifically, when the motor runs at a fixed torque until it encounters an obstacle and stops, then runs in reverse for a certain distance before stopping, this stopping point is set as the zero point. It is suitable for cost-sensitive, space-constrained, or collision-avoidance applications. See 7.1, 7.5.
4. **Automatic zero-return upon single-turn power-** on: After power-on, the system automatically returns to zero (the zero point within a single turn) based on pre-set parameters. This is suitable for applications requiring the retention of a single power-on zero point within a single turn. See 7.1, 7.6.

1.10 Multi-motor synchronization function description

Multi-motor control functionality reduces the number of commands required when controlling multiple motors, thus improving work efficiency. MKS motors offer four multi-motor control modes, allowing users to select the appropriate mode based on their specific needs.

1. **Using broadcast CAN IDes for multi-motor control** is suitable for controlling all motors on a bus to execute the same command.

2. **Using grouped CAN IDes for multi-motor control** is suitable for controlling motor A to perform action ' a' and motor B to perform action 'b' on a bus. See 5.2.14.

3. **Synchronous control is used for multi-motor control** , suitable for synchronously running different motors on a bus to execute different operating commands, with no limit on the number of motors controlled. See 12.3.

1.11 Unit Conversion Instructions

The numerical base in this article is hexadecimal by default.

1.11.1 CRC conversion

1. Instruction data and return data are stored in big-endian format.
2. CRC checksum is CHECKSUM 8bit

   For example: the instruction " 01 83 0C 80 CRC "

   CRC = (0x 01 + 0x 83 + 0x 0C + 0x 8 0) & 0xFF = 0x1 10 & 0xFF = 0x 10

   In other words, after adding each byte, only the last two digits are taken.

1.11.2 Conversion of rotations, pulses, and coordinate values

1. This product uses a 16384-line encoder, so the decimal value of one revolution (360°) is 16384, and the corresponding hexadecimal value is 0x4000; the decimal pulse count under 16 subdivisions is 3200, and the corresponding hexadecimal value is 0xC80.
2. When you need to control the number of rotations of the motor, you can select the position control mode—relative/absolute motion based on the number of pulses.

   For example, if you need to control the motor to rotate 10 times, you can select the position control mode - relative/absolute movement by the number of pulses. At this time, the number of

pulses in the command is 0x7D00 (32000 pulses in 16 microsteps). See 11.1, 11.2.

3. When you need to control the motor's rotation to the corresponding position coordinates, you can select the position control mode—movement relative to/absolutely based on coordinate values.
   For example, if you need to control the motor to move to coordinate 0x4000, you can select the position control mode - move relative to/absolutely according to the coordinate value. In this case, the coordinate value in the command is 0x4000. see 11.3, 11.4..
   If the coordinates of a point are unknown, they can be read using the serial port 31H command ,see 5.1.2.

1.11.3 Explanation of absolute and relative positions

1. Absolute position refers to controlling the motor to move to a specific target position within a preset fixed origin (zero point) coordinate system. Each motion command calculates the displacement based on the origin, regardless of the current position.
2. Relative position refers to controlling the motor to move a specified distance (number of steps or pulses) with the current position as a temporary reference point. The target position changes dynamically with the current position.

A vivid example: Suppose you are reading a 500-page book , turning 100 pages at the absolute level and 100 pages at the relative level.

Absolute position 100 pages → Directly flip to page 100 (target is fixed, regardless of the current page). If you are already on page 100, giving the command "absolute position 100 pages" again will not trigger page turning.

Relative position 100 pages → Scroll backwards 100 pages from the current page. If you are currently on page 10, scroll to page 110 ; if you give the command " relative position 100 pages" again on page 110 , scroll to page 210 (the target changes with the current position) .

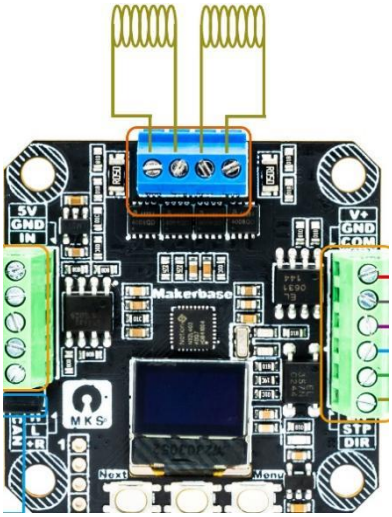
Part 2. Wiring method

2.1 Motor wiring method

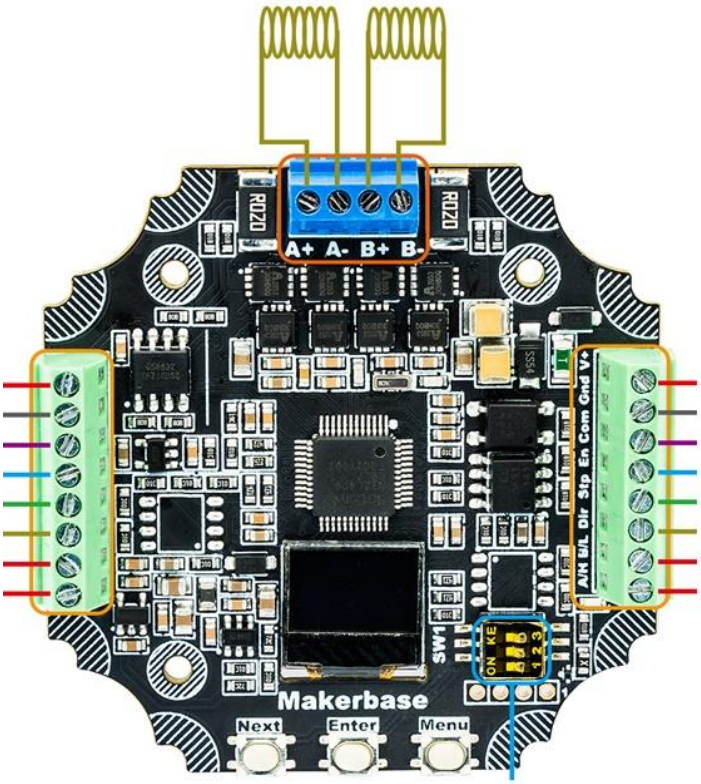
Note: The internal resistance of the motor should be less than 10 ohms .

Connect A+ and A- to one phase of the motor, and connect B+ and B- to the other phase of the motor.

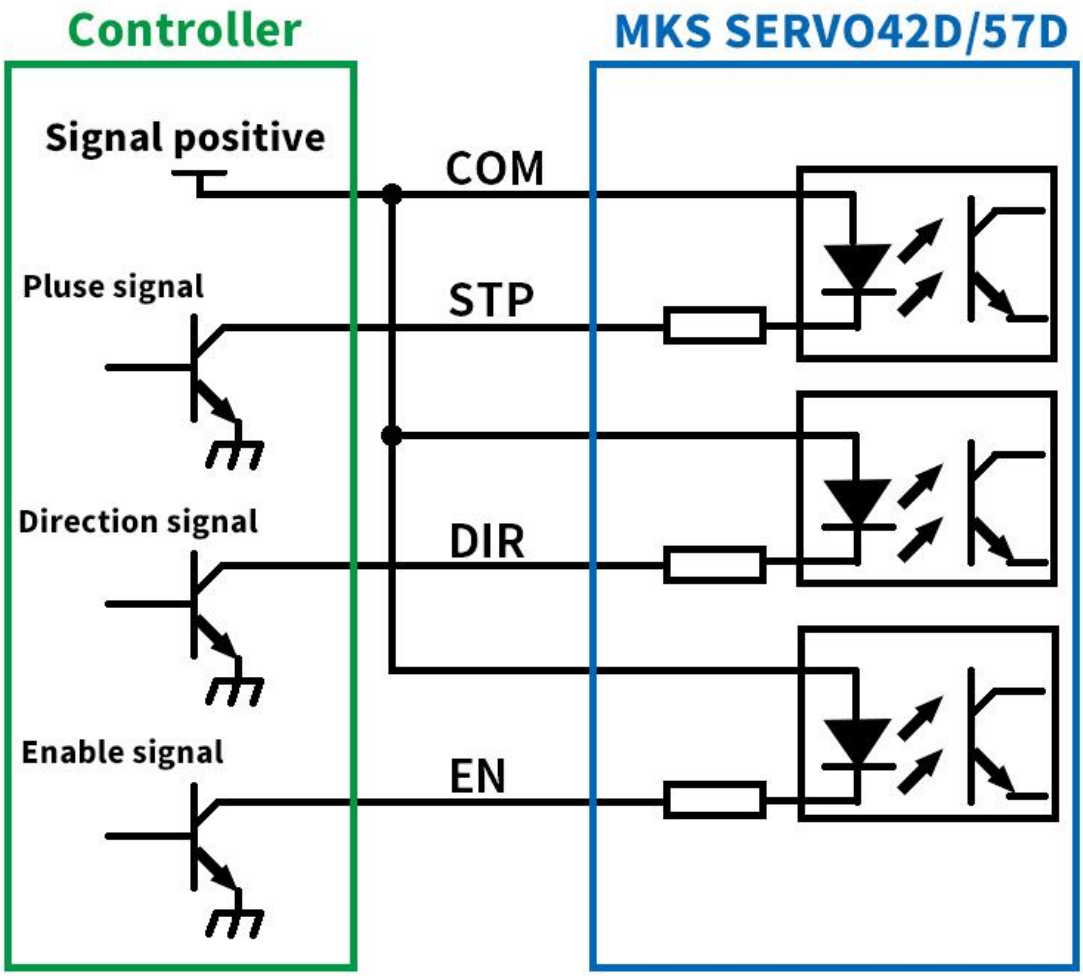
① **SERVO42D CAN Wiring Method**



② **SERVO57D CAN Wiring Method**



2.2 Pulse control wiring method



Note: If the high level of the (STP/DIR/EN) signal is 3.3V, COM must be connected to 3.3V.

If the high level of the (STP/DIR/EN) signal is 5.0V, then COM must be connected to 5.0V.

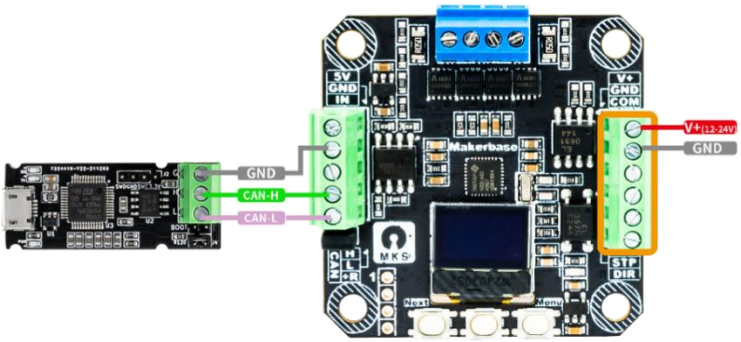
...

2.3 CAN wiring method

Note: To reduce bus interference, the host computer and the motor should share a common ground, and the CAN signal should be transmitted using shielded twisted-pair cable.

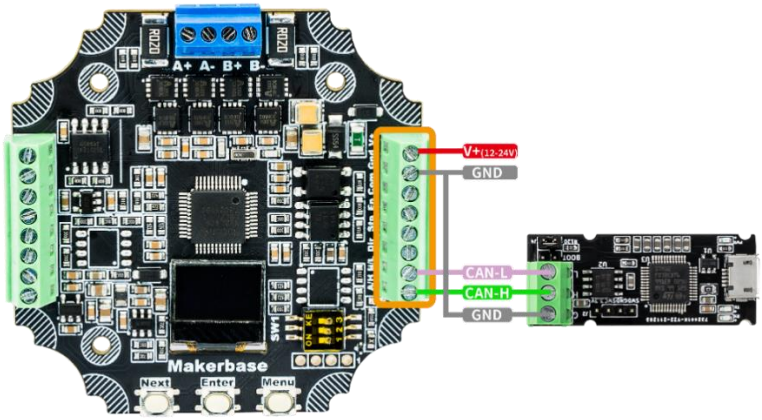
1. CAN standalone communication wiring

① MKS SERVO42D CAN standalone wiring



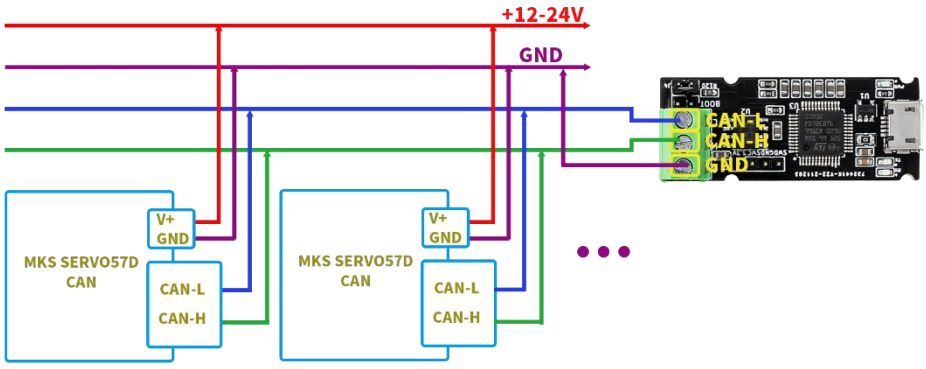
Note: A 120Ω terminating resistor is not required for stand-alone communication (i.e., do not connect a shorting cap).

② MKS SERVO57D CAN standalone wiring



Note: A 120Ω terminating resistor is not required for standalone communication.

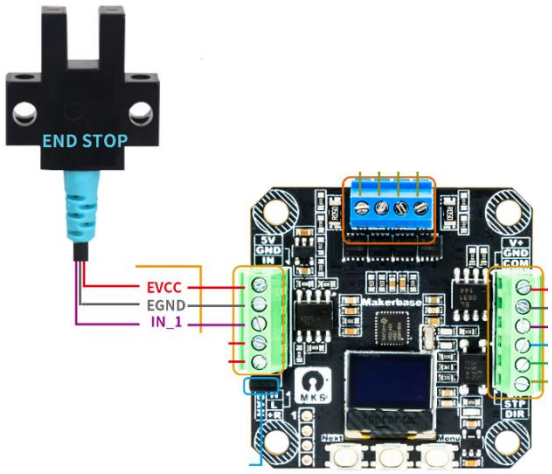
2. CAN multi-machine communication wiring



2.4 External switch wiring method

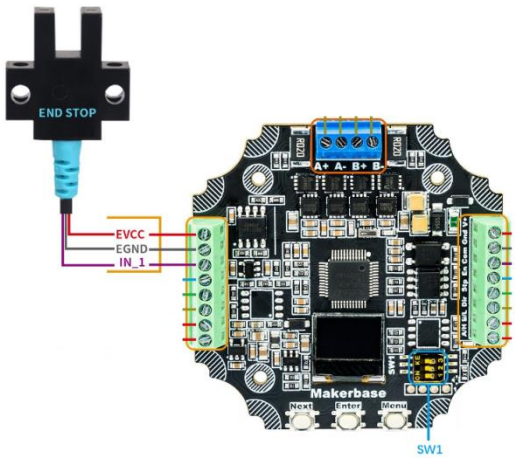
Note: NPN type limit switches are used by default.

① **MKS SERVO42D limit switch wiring**



Note: EVCC/ENGD is powered by the SERVO42D driver board at 5.0V/20mA.

② **MKS SERVO57D limit switch wiring**



SW1		
PIN	ON	OFF
3 2	EVCC/EGND are powered by SERVO57D at 5V (20mA).	EVCC/EGND is powered by an external source (3.3V-24V).
1	CAN 120 Ω terminating resistor	NULL

The mechanical switch only needs to be connected to two signal lines: “EGND” and “IN_1”. Pin 2 of the DIP switch must be in the ON state.

The DIP switches are in the OFF state by default. If the limit switches require internal power supply, the operation should be performed according to the table above.

Part 3. Menu Description

3.1 CAL : Calibrate the motor.
The encoder is calibrated in closed-loop mode,
but this is ineffective in open-loop mode.

3.2 Mode : Work mode selection.

CR_OPEN : pulse interface Open mode, the motor run without encoder.

CR_CLOSE : pulse interface Close mode, the motor run with encoder.

CR_vFOC : pulse interface FOC mode, the motor run with encoder.

SR_OPEN : serial interface Open mode, the motor run without encoder.

SR_CLOSE : serial interface Close mode, the motor run with encoder.

SR_vFOC : serial interface FOC mode, the motor run with encoder.

(Default: CR_vFOC)

	Mode		MAX RPM	Work Current
OPEN	pulse interface	CR_OPEN	400RPM	Fix, the work current is Ma
	serial interface	SR_OPEN		
CLOSE	pulse interface	CR_CLOSE	1500RPM	Fix, the work current is Ma
	serial interface	SR_CLSOE		
vFOC	pulse interface	CR_vFOC	3000RPM	self-adaption, the Max current is Ma
	serial interface	SR_vFOC		

3.3 Ma : Set the working current.

42D current options: 0, 200,400 ... 3000 (mA). (Default 1600mA)

57D current options: 0, 400,800 ... 5200 (mA). (Default 3200mA)

28D current options: 0, 200,400 ... 3000 (mA). (Default 600mA)

35D current options: 0, 200,400 ... 3000 (mA). (Default 800mA)

Other Current such as 123mA need to be set by serial command .It will be added to the last options.

3.4 Hold Ma : Set holding current percentage.

10%, 20%, ..., 90%

(Default: 50%, the holding current at half the working current)

Note: Only for OPEN mode and CLOSE mode, vFOC mode is invalid.

3.5 MStep : Set subdivisions.

Supports subdivision from 1 to 256. (Default: 16)

subdivisions 1,2,4,8,16,32,64,128, and 256 can be set by Menu.

Other subdivisions such as 67 subdivisions need to be set by serial command . It will be added to the last options.It is recommended to use subdivision steps that are powers of 2 to reduce errors, such as 2, 4, 16, 32, 64, etc.

3.6 En : Set the effective level of EN pin.
L : Low level is effective. (Default)
H : High level is valid.
Hold : the driver board is always enabled.

3.7 Dir : Set the positive direction of motor rotation.
CW : Clockwise rotation is positive (Default)
CCW : Counterclockwise rotation is positive
Note: only for pulse interface,the direction of serial interface is set by command.

3.8 AutoSDD : Set auto turn off the OLED screen.
Disable : disable auto turn off the OLED (Default)
Enable : enable auto turn off the OLED
If set to Enable, the screen will automatically turn off after about 15 seconds, and any button can wake up the screen again.

3.9 Protect : Set the motor shaft locked-rotor protection function.
Disable: disable protection (Default)
Enable: enable protection
After this option is enabled, the protection will be triggered when it is detected to be locked-rotor, and the motor will be release.
Note: After the stall protection is activated, there are three ways to release it:
1. Press the Enter button to release the stall protection;
2. Use the serial port command (3D) to release the stall protection;
3. Release the motor, the stall protection can be released.

3.10 MPlyer : Set internal 256 subdivision.
Disable : disable internal 256 subdivision
Enable : enable internal 256 subdivision (Default)
Note: After this option is Enabled, it automatically enable internal 256 subdivision, it can reduce the vibration and noise when the motor at low speed.

3.11 CanRate : Set the bit rate of CAN interface.
125K, 250K, 500K(default),1M

3.12 **CanID** : Set the the slave CAN ID of CAN interface.

01

…

15

16

(Default: 01)

Note: The CAN IDes greater than 16 need to be set by serial command. After it is set, it will be added to this option.

3.13 **CanRSP** :Set the slave respond mode in speed/positon mode.

Disable: disable respond

Enable: enable respond (Default)

Note: If disable respond, It can query the running status of the motor by command “F1”.

3.14 Single-cycle return-to-zero parameters

3.14.1 O_Mode : The motor will go back to zero when power on.

Disable : do not go back to zero. (Default)

DirMode : go back to zero with direction of CW or CCW (the direction is set in O_Dir menu).

NearMode : go back to zero with minimum angle.

3.14.2 Set O : Set the zero point for go back when power on.

(O_Mode must not be Disable)

3.14.3 O_Speed : Set the speed of go back to zero point.

0: Slowest gear

...

4: The fastest gear

3.14.4 O_Dir : Set the direction of go back to zero point.

CW: Clockwise (default)

CCW: Counterclockwise

3.15 return to zero with endstop parameter

3.15.1**Hm_Trig** : Set the effective level of the end stop.

Low : Low level (default)

High : High level

3.15.2**Hm_Dir** : Set the direction of go home.

CW: Clockwise (default)

CCW: Counterclockwise

3.15.3**Hm_Speed** : Set the speed (RPM) of go home.

30 60 90 120 150 180

Other speed such as 600(RPM) need to be set by serial command . It will be added to the last options.

3.15.4**Hm_Mode** : Set the method of go home.

Limited : used endstop for go home(default)

noLimit : mechanical limit for go home

When "noLimit" for go home, the motor will runs with a fixed torque (Hm_Ma setting) until it stops when it encounters an obstacle, and then runs in reverse for a certain distance (94H command setting) and then stops. The stopping point is the zero point.

3.15.5 Hm_Ma : Set the current of "noLimit" go home.

42D current options: 0,200,400...3000 (mA). (Default 400mA)

57D current options: 0,400,800...5200 (mA). (Default 800mA)

28D current options: 0,200,400...3000 (mA). (Default 200mA)

35D current options: 0,200,400...3000 (mA). (Default 200mA)

Note: The "Hm_Ma" is only valid during "noLimit" go home operation.

It should be set to a smaller current as much as possible to avoid damaging the motor.

3.15.6 **GoHome** : Go home.

Note1: It need an "end stop". The motor will keep running until it hits the limit switch.

Note2: If the limit switch is already closed, the motor will rotate in the opposite direction to homeDir until the limit switch is opened, and then go home.

3.16 **EndLimit** : Set the endstop-limit function.

Disable: Turn off endstop function. (default)

Enable: Turn on endstop function.

Note 1: After using the endstop function for the first time or changing the parameters, you need to "go home" once.

(Menu -> GoHome) (or command "91H")

3.17 **Restore** : Reload the default parameters.

After successful recovery, the LED lights flash and the driver board automatically restarts without recalibrating the motor.

Note: Press the "Next" key first, then power on, it can quickly restore the default parameters.

3.18 **About** : Show version parameters.

You can view information such as firmware version number and date.

3.19 **Exit** : Exit the parameter setting menu.

Part 4. CAN data format description

The message uses a standard frame, and the maximum length of the data field is Byte8.

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2...Byte(n-1)	Byte n (n≤8)
ID		DLC(n)	code	data	CRC

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2...Byte(n-1)	Byte n (n≤8)
ID		DLC(n)	code	data	CRC

1. The CAN ID range is 00~2047. (default is 01).
   00 is the broadcast CAN ID;
   01~10 can be set in the CanID option of the display menu;
   greater than 10 need to be set by serial commands.
2. The code (code) executes the corresponding command.
   for example, 0x80 executes the calibration command.
3. The Check code is CHECKSUM 8bit
   For example, the command to read encoder values:

Downlink message (host computer → driver board)				
ID	...	DLC	Byte1	byte2
01		2	30	CRC

CRC = (0x01 + 0x30) & 0xFF = 0x31 & 0xFF = 0x31

Note: When sending commands using a broadcast CAN ID, the slave device will not respond.

Part 5. General Instructions

Note 1: When sending commands using broadcast CAN IDes or packet CAN IDes, the slave device will not respond.

Note 2: Using the 8CH command, you can set whether the slave device should respond, or select the response method, see 5.2.12.

5.1 Read-only parameter instructions

5.1.1 Reading the value of a multi-turn encoder

Note: Encoder single-turn value range is 0~0x4000

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	30H	CRC

Returned data:

Uplink message (host computer ← driver board)							
CAN ID	...	DLC	Byte1	Byte2-Byte5	Byte6	Byte7	Byte8
01		8	code	carry	value		Checksum
			30H	carry (int32_t)	value(uint16_t)		CRC

carry: the carry vaule of the encoder.

value: the current vaule of the encoder.(range 0~0x3FFF)

When value is greater than 0x3FFF, carry +=1.

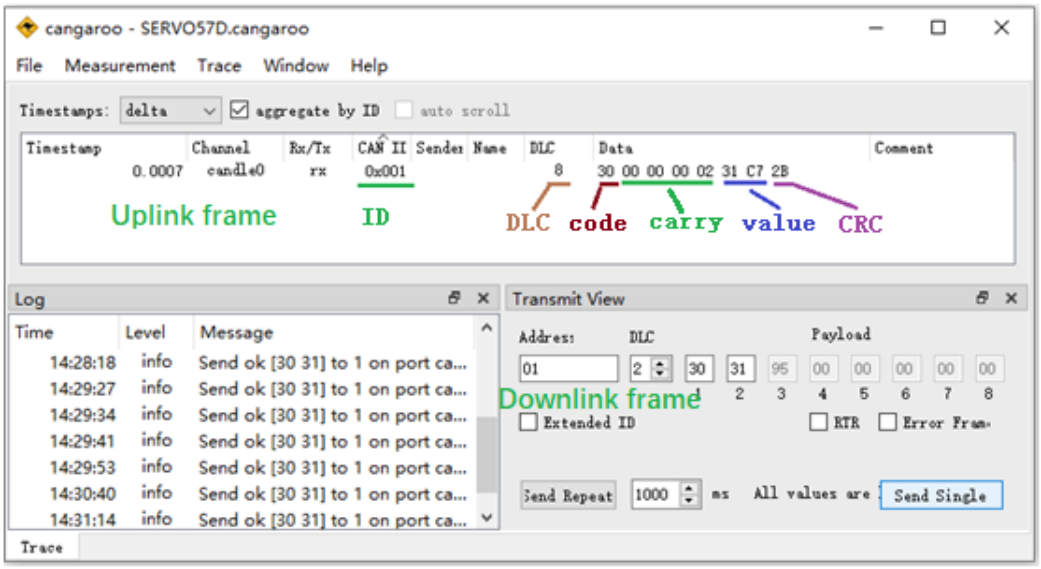
When Value is less than 0, carry -=1.

For example:

If the current carry|value is 0x3FF0, After one turn CCW,the carry|value (+0x4000) is 0x13FF0.

If the current carry |value is 0x3FF0, After one turn CW,the carry|value (-0x4000)is 0xFFFFFFFF3FF0.

The Cangaroo example is as follows:



5.1.2 Read the cumulative multi-turn encoder value

It can read the motor rotation range recorded by the encoder after power-on.

Note: Encoder single-turn value range is 0~0x4000

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	31H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2-Byte 7	Byte8
01		8	Code	value	Checksum
			31H	value (int48_t)	CRC

After one turn clockwise, the value += 0x4000;

After one turn CCW, the value -= 0x4000;

For example:

If the current value is 0x3FF0, After one turn CCW, the value(+0x4000) is 0x7FF0.

If the current value is 0x3FF0, After one turn CW, the value(-0x4000) is 0xFFFFFFFFFFF0.

Note: When moving relative to or absolutely according to coordinate values, use the encoder value as the coordinate.

5.1.3 Read the real-time speed of the motor

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	32H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2...Byte3	Byte4
01		4	Code	speed	Checksum
			32H	rpm (int16_t)	CRC

Note : if it run CCW, the speed > 0 (RPM)
if it run CW, the speed < 0 (RPM)

5.1.4 Read the number of received pulses

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	33H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2-Byte5	Byte6
01		6	Code	pulse count	Checksum
			33H	value(int32_t)	CRC

5.1.5 Read the raw accumulated multi-turn encoder value

read the RAW encoder value.

Note: Encoder single-turn value range is 0~0x4000

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	35H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2-Byte 7	Byte8
01		8	Code	Accumulated multi-turn encoder value	Checksum
			35H	value (int48_t)	CRC

After one turn CW, the value += 0x4000;

After one turn CCW, the value -= 0x4000;

5.1.6 Read position angle error

Angle error is the difference between the position angle controlled by the command and the real-time angle position of the motor. The unit is 0~ 51200 , which means 0~360°. For example, when the error is 1°, the value is 51200 / 360° = 1 42.22 , and so on.

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	39H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2-Byte5	Byte6
01		4	Code	Angular error	Checksum
			39H	value(int32_t)	CRC

5.1.7 Read motor enable status

When using bus control, this command can be used to obtain the enable status of the driver board.

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	3AH	CRC(3B)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Enable state	Checksum
			3AH	enable (uint8_t)	CRC

enable = 1 The motor is enabled.

enable = 0 Motor is not enabled.

5.1.8 Read the motor return to zero status

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	3BH	CRC

Returned data:

Uplink message (host computer ← driver board)						
CAN ID	...	DLC	Byte1	Byte2	Byte3	Byte4
01		4	Code	data1	data2	Checksum
			3BH	status1	status2	CRC

statusx =0 going to zero.

statusx =1 go back to zero success.

statusx =2 go back to zero fail.

Note: status1 indicates single-turn zero return state;

Status2 indicates non-single-turn zero return state;

5.1.9 Automatically return read-only parameters at regular intervals

This command can set parameters so that the motor can automatically return a certain read-only parameter at regular intervals, without the host computer needing to frequently issue corresponding query commands.

The command format is as follows:

Downlink message (host computer → driver board)						
CAN ID	...	DLC	Byte1	Byte2	Bytes 3-4	Byte5
01	...	5	Cmd	Code	Timer	Checksum
			01H	code	times	CRC

code: The code corresponding to the read-only parameter.

times: The timeout period, in milliseconds (when times = 0, no data is returned).

Returned data:

Uplink message (host computer ← driver board)						
CAN ID	...	DLC	Byte1	Byte2	Byte3	Byte4
01	...	4	Cmd	Code	state	Checksum
			01H	code	status	CRC

status=00 Set fail.

status=01 Set success.

The parameter data format returned periodically is as follows:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2-n	Byte n+1
01	...	n	Code	Parameter value	Checksum
			code	param	CRC

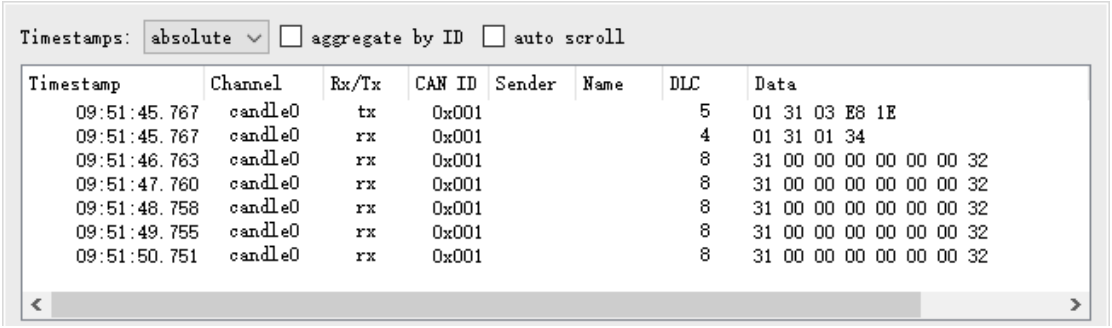
param: The corresponding read-only parameter

Taking the reading of the "accumulator multi-turn encoder value" every 1000ms as an example:

Send 01 01 31 03 E8 1E

Return 01 01 31 01 34

After successful setup, the motor will return the "cumulative multi-turn encoder value" every 1000 milliseconds, as shown in the figure below.



5.1.10 Read configuration parameter

After writing the configuration parameters, you can use this command to read and view them.

The command format for reading write-only configuration parameters is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Cmd	Code	Checksum
			00H	code	CRC

code: The code corresponding to the parameter

For example, to read the "working mode", the corresponding code is 82H.

The returned parameter data format is as follows:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2 - n	Byte n+1
01	...	n+1	Code	Parameter value	Checksum
			code	param	CRC

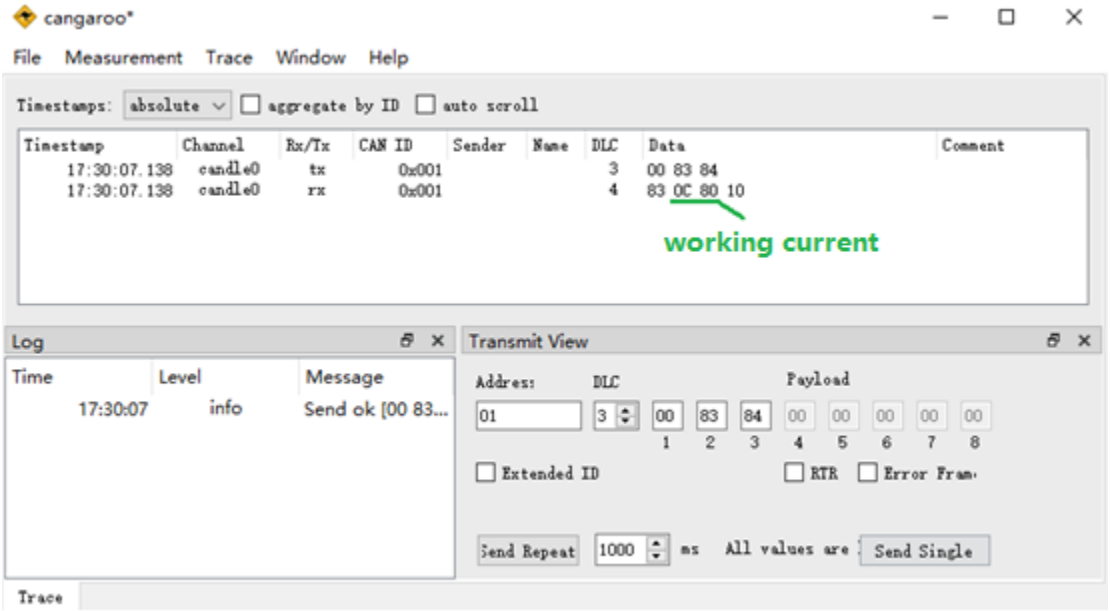
param: The corresponding system parameter

Note: The data format of param should be consistent with the data format when setting this parameter.

If this parameter does not support reading, the returned data will be as follows:

Uplink message (host computer ← driver board)						
CAN ID	...	DLC	Byte1	Byte2	Byte3	Byte4
01	...	4	Code			Checksum
			code	FFH	FFH	CRC

The following image shows an example of reading "operating current":



5.1.11 Read version information

Read hardware and firmware information.

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	40H	CRC

The returned data format is as follows:

Uplink message (host computer ← driver board)							
CAN ID		DLC	Byte1	Byte2 (b7-b4)	Byte2 (b3-b0)	Bytes 3-4	Byte5
01	...	5	Code	Calibration status	Hardware version	Firmware version	Checksum
			40H	cal	hardVer	firmVer [3]	CRC

cal = 1 The motor has been calibrated.

cal = 0 Motor not calibrated

The hardware version correspondence is as follows:

Board type	hardVer
S42D_485	1
S42D_CAN	2
S57D_485	3
S57D_CAN	4
S28D_RS485	5
S28D_CAN	6
S35D_RS485	7
S35D_CAN	8

5.1.12 Read / Write User ID

User-defined IDs allow users to assign their own ID numbers to motors, making it easier to distinguish between them.

1. Write user ID number

The command to write the ID is as follows:

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Bytes 2-5	Byte6
01		6	Code	User ID	Checksum
			42H	ID	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Write status	Checksum
			42H	status (uint8_t)	CRC

status=00 Write failed

status=01 Write successed

2. Read user ID

The command to read the ID is as follows:

Downlink message (host computer → driver board)				
CAN ID	…	DLC	Byte1	Byte2
01		2	Code	Checksum
			42H	CRC

Return ID data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Bytes 2-5	Byte6
01		6	Code	User ID	Checksum
			42H	ID	CRC

5.2 Write-only parameter command

5.2.1 Calibrate encoder

（Same as the "Cal" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	80H	00	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	status	Checksum
			80H	status (uint8_t)	CRC

status = 0 Calibrating…

status = 1 Calibration successful

status = 2 Calibration failed

Note1: The motor must be unloaded. It is recommended to calibrate it before installing it into the machine.

5.2.2 Set working mode

（Same as the "Mode" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Work mode	Checksum
			82H	mode (uint8_t)	CRC

The corresponding working modes are shown in the table below.

List of working modes		
mode	Work mode	Remark
00H	CR_OPEN (Pulse interface open-loop mode)	
01H	CR_CLOSE (Pulse Interface Closed-Loop Mode)	
02H	CR_vFOC (Pulse Interface FOC Mode)	(default)
03H	SR_OPEN (Bus interface open-loop mode)	
04H	SR_CLOSE (Bus interface closed-loop mode)	
05H	SR_vFOC (Bus Interface FOC Mode)	

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			8 2 H	status (uint8_t)	CRC

status = 0 Set fail.

status = 1 Set success.

5.2.3 Set working current

（Same as the "Ma" option on screen）

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2 - 3	Byte4
01		4	Code	Current value	Checksum
			83H	value (uint16_t)	CRC

Note1:the new current will show in the screen of Ma option.

SERVO42D/28D/35D: Maximum Current =3000mA

SERVO57D: Maximum Current =5200mA

To set the operating current of the motor without saving it during operation, the command is as follows:

Downlink message (host computer → driver board)						
CAN ID	...	DLC	Byte1	Byte2 - 3	Byte4	Byte5
01		5	Code	Current	Cmd	Checksum
			83H	value (uint16_t)	00H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			83H	status (uint8_t)	CRC

status = 0 Set failed.

status = 1 Set success with saved.

status = 2 Set success without saved.

Note: It is not recommended that users directly set the maximum current as the operating current to avoid damage to the driver board due to fluctuations.

5.2.4 Set holding current percentage

（Same as the "HoldMa" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	percentage	Checksum
			9BH	ratio (uint8_t)	CRC

ratio = 00 10%
ratio = 01 20%
ratio = 02 30%
…
ratio = 08 90%

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			9BH	status (uint8_t)	CRC

status = 0 Set fail.
status = 1 Set success.

5.2.5 Set subdivisions

（Same as the "MStep" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Set subdivisions	Checksum
			84H	micstep (uint8_t)	CRC

powers of 2 to subdivide the number of steps to reduce errors, such as 2, 4, 16, 32, 64, etc.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			84H	status (uint8_t)	CRC

status = 0 Set fail.
status = 1 Set success.

5.2.6 Set the active level of the En pin.

（Same as the "**En**" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Set the level	Checksum
			85H	Level (uint8_t)	CRC

Level = 00 active low (L)
Level = 01 active high (H)
Level = 02 active always (Hold)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			8 5 H	status (uint8_t)	CRC

status = 0 Set fail.
status = 1 Set success.
Note: If you are unsure which level to set, you can directly set it to Hold.

5.2.7 Set the motor rotation direction

（Same as the "**Dir**" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Set direction	Checksum
			86H	dir (uint8_t)	CRC

dir = 00 CW
dir = 01 CCW

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			86H	status (uint8_t)	CRC

status = 1 Set success.
status = 0 Set fail.
Note: This is only valid for pulse interfaces; the direction of the serial interface is determined by the command.

5.2.8 Set automatic screen-off function

（Same as the "AutoSDD" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			87H	enable (uint8_t)	CRC

enable = 01 enabled

enable = 00 disabled

If set to Enable, the screen will automatically turn off after about 15 seconds, and any button can wake up the screen again.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			87H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.9 Set subdivision interpolation function

（Same as the "Mplyer" option on screen）

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Set protection	Checksum
			89H	enable (uint8_t)	CRC

enable = 00 disables subdivision interpolation.

enable = 01 Enables subdivision interpolation (default value)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			89H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.10 Set CAN interface bit rate

（Same as the "CanRate" option on screen）

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Set protection	Checksum
			8AH	bitRate (uint8_t)	CRC

bitRate = 00 125K
bitRate = 01 250K
bitRate = 02 500K (default)
bitRate = 03 1M

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			8AH	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

5.2.11 Set slave CAN ID

（Same as the "CanID" option on screen）

Downlink message (host computer → driver board)						
CAN ID	...	DLC	Byte1	Byte2	Byte3	Byte4
01		4	Code	Slave CAN ID		Checksum
			8BH	address (uint16_t)		CRC

Note1:the new address will show in the screen of CanID option.
Note2: 00 is the broadcast address

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			8BH	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

5.2.12 Configure slave response method

The configuration command is as follows:

Downlink message (host computer → driver board)						
CAN ID	...	DLC	Byte1	Byte2	Byte3	Byte4
01	...	4	Code	data 1	data 2	Checksum
			8CH	XX	YY	CRC

XX = 01 enabled respond (default)

XX = 00 disabled respond

YY = 01 enabled active (default)

YY = 00 disabled active

Example of different response methods, taking position control mode 1 as an example:

The host sends 01 FD 02 80 02 00 00 FA 00 7C

a. **In no-response mode (XX=0, YY=0 or YY=1)**
   The slave device does not return any information.

b. **In the mode where data is not actively initiated (XX=1, YY=0)**
   Slave immediately returns to position control start 01 or fails 00

c. **In the default mode (XX=1, YY=1)**
   Slave immediately returns to position control start 01 or fails 00
   After the motor finishes running or stops when it hits the limit switch, return to 02 or 03.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	Setting status	Checksum
			8CH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Note: After setting the slave device to no response, the motor operating status can be queried using the code "F 1 H".

5.2.13 Set Key lock function

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Set protection	Checksum
			8FH	enable (uint8_t)	CRC

enable = 00 Key lock

enable = 01 Key lock

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			8FH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.14 Set group CAN ID

The configuration command is as follows:

Downlink message (host computer → driver board)						
CAN ID	...	DLC	Byte1	Byte2	Byte3	Byte4
01		4	Code	data		Checksum
			8DH	ID(01~0x7FF)		CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			8DH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Application instructions for using packet CAN IDes:

Assuming there are 6 motors, the CAN ID settings are as follows:

	Broadcast CAN ID	Slave CAN ID	Group CAN ID
Motor 1	0	1	0x 50
Motor 2	0	2	0x 50
Motor 3	0	3	0x 50
Motor 4	0	4	0x 51
Motor 5	0	5	0x 51
Motor 6	0	6	0x 51

send 01 FD 01 2C 64 00 0C 80 1B, motor 1 will rotate a turn

send 00 FD 01 2C 64 00 0C 80 1A, motor1-6 will rotate a turn

send 50 FD 01 2C 64 00 0C 80 6A, motor1-3 will rotate a turn

send 51 FD 01 2C 64 00 0C 80 6B, motor4-6 will rotate a turn

Note: When sending commands using a packet CAN ID, the slave device will not respond.

5.2.15 Set heartbeat protection time

Heartbeat protection refers to the system that, if the motor does not receive any instructions from the host computer within the set protection time, it will control the motor to stop urgently, preventing abnormal accidents caused by communication interruption.

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Bytes 2-5	Byte6
01		6	Code	Protection time	Checksum
			98H	times	CRC

times: Protection time, in milliseconds. (Default: 0)

Note: When times=0, the heartbeat protection function is turned off.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			98H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.16 Set the location to reach the threshold.

In bus position control mode, the "position operation completed" message will only be returned when the actual position of the motor is less than the threshold.

The configuration command is as follows:

Downlink message (host computer → driver board)						
CAN ID	…	DLC	Byte1	Byte2	Bytes 3-4	Byte5
01		5	Code	Enable	threshold	Checksum
			95H	enable	values(uint16_t)	CRC

enable: 00 disables the threshold detection function.

01 Enable the threshold detection function (default)

values: Location threshold (default value 200)

Note: The maximum value is 65535, which is 360 degrees.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			95H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.17 Set no-verification mode

No-check mode means that after the motor receives a message, it will execute the message command without checking whether the CRC checksum is correct.

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	control word	Checksum
			0FH	01H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	Setting status	Checksum
			0FH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Note: This instruction is for testing purposes only and cannot be saved. The motor will enter a verification mode every time it is powered on.

5.3 PID Parameter Setting Instructions

Note: MKS motors have their PID parameters pre-adjusted at the factory. Unless otherwise required, users are advised against adjusting the PID parameters themselves. If such adjustment is necessary, proceed with extreme caution to avoid damaging the motor!

5.3.1 Set the Kp and Ki parameters in vFOC mode .

The command format is as follows:

Downlink message (host computer → driver board)							
ID	...	DLC	Byte1	Byte2	Bytes 3-4	Bytes 5-6	Byte7
01	...	7	Code	CMD	KP	KI	Checksum
			96H	00H	Kp	Ki	CRC

Kp: Range 0-1024 (Default: 0xDC)

Ki: Range 0-1024 (Default: 0x64)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	Setting status	Checksum
			96H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Note: The corresponding read parameter command is 01 00 96 00 CRC.

5.3.2 Set the Kd and Kv parameters in vFOC mode .

The command format is as follows:

Downlink message (host computer → driver board)							
ID	...	DLC	Byte1	Byte2	Bytes 3-4	Bytes 5-6	Byte7
01	...	7	Code	CMD	KD	KV	Checksum
			96H	01H	Kd	Kv	CRC

Kd : Range 0-1024 (Default: 0x10E)

Kv : Range 0-1024 (Default: 0x140)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	Setting status	Checksum
			96H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Note: The corresponding read parameter command is 01 00 96 01 CRC.

5.3.3 the Kp and Ki parameters for CLOSE mode.

The command format is as follows:

Downlink message (host computer → driver board)							
ID	…	DLC	Byte1	Byte2	Bytes 3-4	Bytes 5-6	Byte7
01		7	Code	CMD	KP	KI	Checksum
			97H	00	Kp	Ki	CRC

Kp: Range 0-1024 (Default: 0xC8)
Ki: Range 0-1024 (Default: 0x50)

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			97H	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

Note: The corresponding read parameter command is 01 00 97 00 CRC.

5.3.4 the Kd and Kv parameters for CLOSE mode.

The command format is as follows:

Downlink message (host computer → driver board)							
ID	…	DLC	Byte1	Byte2	Bytes 3-4	Bytes 5-6	Byte7
01		7	Code	CMD	KD	KV	Checksum
			97H	01	Kd	Kv	CRC

Kd: Range 0-1024 (Default: 0xFA)
Kv: Range 0-1024 (Default: 0x12C)

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			97H	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

Note: The corresponding read parameter command is 01 00 97 01 CRC.

5.4 Stall protection instructions

If the motor fails to reach the designated position within the response time due to overload or any other reason, the stall protection will be triggered, the motor will unlock and the shaft will be released to avoid damage.

There are two protection modes:

1. Current overload protection mode
   If motor overcurrent is detected, the stall protection will be activated.
2. Position out-of-tolerance protection mode
   If the motor position error exceeds y within time period x , the protection mechanism will be activated. (x and y are configurable)

These two protection modes can be turned on or off independently. They monitor the motor independently, and the motor protection will be activated when either protection condition is triggered.

Note: When the current overload protection is triggered, the screen will display "Wrong…"

When the position error protection is triggered, the screen displays "Wrong2…".

5.4.1 Set overcurrent protection

The corresponding instructions for the "Protect" option on the screen are as follows:

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Set protection	Checksum
			88H	enable (uint8_t)	CRC

enable = 00 disables current overload protection (default).

enable = 01 Enables current overload protection

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			88H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Note: After the stall protection is activated, there are three ways to deactivate it:

1. Press the Enter key to release the stall protection;
2. The stall protection can be released via CAN command (3D);
3. Loosening the motor shaft can release the stall protection .

5.4.2 Set position out-of-tolerance protection parameters

The configuration command is as follows:

Downlink message (host computer → driver board)							
ID		DLC	Byte1	Byte2	Bytes 3-4	Bytes 5-6	Byte7
01	...	7	Code	Protection	time	Error count	Checksum
			9DH	Enble	Tim (uint16_t)	Errors (uint16_t)	CRC

Enble 0 : Disable position deviation protection (default)
1 : Enable position deviation protection
Tim: sets the error statistics time length.
Note: 1 (Tim) unit is approximately equal to 15 ms.
Errors : sets the boot protection error count
Note: When Errrors = 28000 , the motor is misaligned by 360 degrees.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	...	3	Code	state	Checksum
			9DH	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

For example:
send 01 9D 01 00 14 36 B0 99
return 01 9D 01 9F
Enable location protection.
Error statistics time (0x0014) 20 * 15 ms = 300 ms
Error count (0x36B0) 14000 (180 degrees)
If the motor detects a misalignment of more than 180 degrees within 300ms, the protection will be activated.

After the stall protection is activated , simply loosen the motor shaft to release the stall.
In bus control mode, the stall state can also be released by sending a command (3DH).

5.4.3 Read motor stall status

When a motor stalls, a stall flag will be set. This command can be used to determine whether a stall has occurred. If the stall protection option is enabled, the drive board will automatically shut down the driver upon a stall.

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	3EH	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	stalled state	Checksum
			3EH	status (uint8_t)	CRC

status=00 motor is not stalled

status=01 motor is stalled.

Note: When the motor stalls, the stall state can be released by writing the command (3DH).

5.4.4 Release the motor from stall state

When the motor stalls, sending this command can release the current stall state.

If a stall occurs again after the stall is cleared , the stall protection will still be triggered.

The release command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	3DH	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Released status	Checksum
			3DH	status (uint8_t)	CRC

status = 0 release failed.

status = 1 release sucessed.

Note: After the stall protection is activated , the stall can be released by loosening the motor shaft .

5.5 Recovery parameters and reset instructions

5.5.1 Restore factory settings

The command to restore factory settings is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	3FH	CRC

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Restored status	Checksum
			3FH	status (uint8_t)	CRC

status = 0 Recovery failed

status = 1 Recovery successful

Note 1: After restoring the default parameters, the driver board will automatically restart, and there is no need to recalibrate the motor.

Note 2: Press and hold the "Next" button before powering on. Wait for the LED light to illuminate to restore the default parameters.

5.5.2 Reset and restart the motor

The command to restore factory settings is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	41H	CRC

Note : This command only resets and restarts the motor; it does not modify configuration parameters.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Restored status	Checksum
			41H	status (uint8_t)	CRC

status = 0 Reset failed

status = 1 Reset successful

Part 6. IO Port Operation Instructions

6.1 Read I/O port status

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	34H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Port status	Checksum
			34H	status (uint8_t)	CRC

Port status							
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Undefined				OUT_2	OUT_1	IN_2	IN_1

Note: After the 9EH instruction sets the limit remapping function to be effective, bit0 corresponds to the En state and bit1 corresponds to the Dir state .

6.2 Configure port input/output mode

This only applies to setting the IN_1 port of 28D/35D/42D, and the command is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Port mode	Checksum
			9FH	type	CRC

type = 0 IN_1 As the input port (default)
type = 1 IN_1 as the output port

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			9FH	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

6.3 Write data to IO port

The command to write a port is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Port data	Checksum
			36H	data	CRC

The data is defined as follows:

data							
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
OUT2_mask		OUT1_mask		OUT_2	OUT_1/IN_1	0	0

OUT2_mask 0: Do not write to OUT_2 IO port(default)
1: Write OUT_2 value to OUT_2 IO port
2: OUT_2 IO port value remains unchanged

OUT1_mask 0: Do not write to OUT_1 IO port (default)
1: Write OUT_1 value to OUT_1 IO port
2: OUT_1 IO port value remains unchanged

OUT_2 OUT_2 port write value (0/1)

OUT_1 OUT_1 port write value (0/1)

Note: When using the 9FH instruction to configure IN_1 of 28D/35D/42D as an output port, the value of bit2 can be directly written to the IN_1 port without being restricted by OUT1_mask.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Write status	Checksum
			36H	status (uint8_t)	CRC

status = 0 Write failed

status = 1 Write successful

6.4 IO port pulse frequency division output

Map the OUT_2 port to a pulse divider output port (only for 57D). The configuration command is as follows:

Downlink message (host computer → driver board)						
CAN ID		DLC	Byte1	Byte2	Bytes 3-6	Byte7
01	…	7	Code	Start level	division period	Checksum
			99H	divLevel	divPeriod	CRC

divLevel 0: Low start level (default)
1: High start level

divPeriod Frequency division period (default 0)

When divPeriod < 100, no frequency divider output

When divPeriod >= 100, the PEND port toggles once every divPeriod pulse cycle. For example, setting 16 subdivisions, divPeriod = 3200, then the PEND port flips once for every revolution of the motor.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Setting status	Checksum
			99H	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

Note: To disable this feature, simply set divPeriod = 0.

Part 7. Motor return to zero instructions

motor homing methods : "origin homing" and "coordinate homing".

7.1 Explanation of the method of returning to zero at the origin

There are two ways to "return to zero": "return to zero by origin switch" and "return to zero by mechanical limit switch".

1. Return the origin switch to zero.

Connect the origin switch to the corresponding port:

Origin switch access port	
Default port	Remapped port
IN1	EN

For details, please refer to: 1.7 IO Port DescriptionDescription

The process of returning to zero is as follows:

a. The motor searches for the switch based on the set "direction" and "speed " ;
b. Stop immediately upon encountering the rising edge of the switch signal;
c. Mark the current position as "coordinate zero point", and the origin has successfully returned to zero.

Note: If the motor starts in the closed position of the switch, it will first run in reverse to disengage from the switch.

2. Mechanical limit switch returns to zero.

The zero-return torque when the mechanical limit is returned to zero needs to be set in advance via command. The set torque should be sufficient to drive the load, but should not be too large to avoid damaging the equipment.

The process of returning to zero is as follows:

a. searches for the mechanical limit position using the set "speed " , "direction", and "torque " ;
b. When encountering a mechanical limit switch, the rotor will stop and then run to the preset "origin offset" position to stop.
c. Mark the current position as "coordinate zero point", and the origin has successfully returned to zero.

3. Single lap to zero

The "zero point" of the coordinates within a single circle needs to be set in advance via instructions.

The direction of the single-turn zeroing needs to be set in advance via command: "forward", "reverse", or "nearest".

The process of returning to zero is as follows:

a. The motor returns to the "coordinate zero point" position within a preset single revolution in the set zero-return direction and speed;
b. Upon arrival, the current position is reset to zero, and the single-lap coordinates have successfully returned to zero.

7.2 Explanation of coordinate zeroing method

To return to zero directly, you need to first execute the "Return to Zero" function to determine the "zero point".

The process of returning to zero is as follows:

No search process is required; it directly runs to the "coordinate zero point" position and returns to zero successfully.

7.3 Set parameters related to zero return.

7.3.1 Configure port mapping

(Bus control mode only)

28/35/42D motor, since it only has a left limit port, in bus control mode, the limit port remapping can be enabled to add a right limit port.

57D motors, the limit port can be remapped if wiring is convenient.

After enabling port forwarding:

Left limit switch - > En port

Right limit switch - > Dir port

The COM port must be connected to the corresponding high level.

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Enable	Checksum
			9EH	enable (uint8_t)	CRC

enable = 00 Disable port mapping (default)

enable = 01 Enable port mapping

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			9EH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

7.3.2 Set parameters such as zero- return direction and speed.

The corresponding screen options are **HmTrig**, **HmDir**, **HmSpeed**, **EndLimit**, **HmMode**.

The configuration command is as follows:

Downlink message (host computer → driver board)									
ID		DLC	Byte1	Byte2	Byte3	Bytes 4-5	Byte6	Byte7	Byte8
01	…	8	Code	effective level	Return to zero direction	return speed	Limit enable	Zero mode	Checksum
			90H	homeTrig	homeDir	homeSpeed	EndLimit	hm_mode	CRC

homeTrig Effective level when home switch is closed
0: Low level when closed
1: High level when closed
homeDir Return to Zero Direction
0: Clockwise
1: Counterclockwise
HomeSpeed zero-return speed 0~3000 (RPM).
EndLimit limit enable
0: Disable limit function
1: Enable limit function
hm_mode 0: Return to zero using the origin switch
1: Mechanical limit switch for zeroing
2: Single lap to zero

Note 1: After using the limit function for the first time or changing the limit parameters, you need to perform a limit reset once.
(Menu -> GoHome or serial command "91H")

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Setting status	Checksum
			90H	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

7.3.3 Set the zero- return torque and origin offset parameters.

The configuration command is as follows:

Downlink message (host computer → driver board)						
ID	...	DLC	Byte1	Byte2-5	Bytes 6-7	Byte8
01	...	8	Code	Return offset	return current	Checksum
			94H	Hm_offset	Hm_ma	CRC

hm_ma: Mechanical limit return-to-zero current
(invalid for switch return-to-zero and single-turn return-to-zero).

Hm_offset: Origin offset
(invalid for switch return to zero and single-turn return to zero)

For example:

Hm_offset = 0x4000 (Returns one full rotation, 360 degrees)

Hm_offset = 0x2000 (Returns half a circle, 180 degrees) (Default)

Note 1: When setting the mechanical limit to return to zero current, the specific value will not be displayed on the screen.

If the value is set to 200mA <= hm_ma < 400mA, the screen will display 200mA, and so on.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	Setting status	Checksum
			94H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

7.3.4 Set single-cycle zero-return parameters

the corresponding screen options"0_Mode, Set 0, 0_Speed , 0_Dir", the settings are as follows:

Downlink message (host computer → driver board)								
ID		DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6
01	...	5	Code	Zero mode	Set 0 points	speed	direction	Checksum
			9AH	mode	enable	speed	dir	CRC

mode: Return to zero method

0 : Disable

1: DirMode

2 : NearMode

enable: Set to 0 points

0 : Clear 0 points

1 : Set 0 point

2: Keep 0 points

speed: Zeroing speed

0 ~ 4 (The higher the value, the faster the speed)

dir : Return to zero direction

0 : CW

1 : CCW

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	...	3	Code	Setting status	Checksum
			9AH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

7.3.5 Setting Zero Point Instructions

This command directly sets the current position to "zero".

The configuration command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	92H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	...	3	Code	Calibration status	Checksum
			92H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

7.3.6 Execute the zero-return instruction

The execution instructions are as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Execution method	Checksum
			91H	goZeroMode	CRC

goZeroMode 00: Execute the "Return to Zero" function

01: Execute the "Coordinate Zeroing" function

Note 1: You can execute "Return to Zero" first to determine the "Zero Point Coordinates" before executing "Return to Zero".

Note 2: If the command does not include the "execution mode" parameter, the "origin return to zero" function will be executed by default.

The following two instructions are equivalent

01 91 92

01 91 00 92

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Return to zero state	Checksum
			91H	status (uint8_t)	CRC

status = 0 failed to return to zero

status = 1 starting return to zero

status = 2 return to zero complete

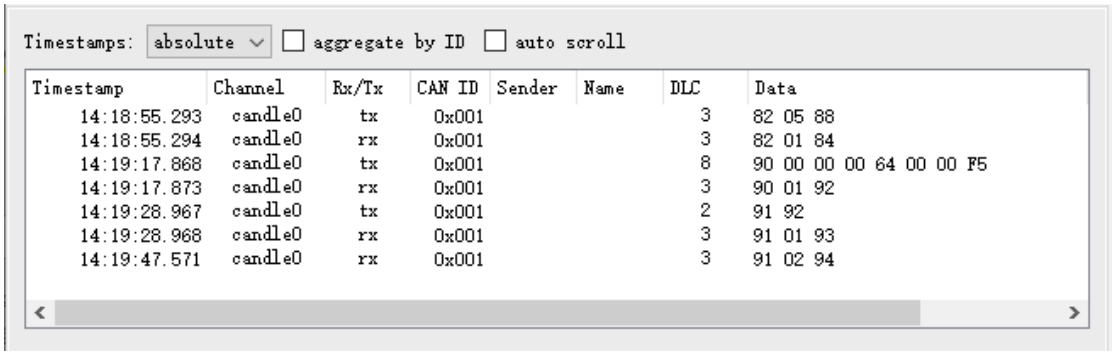
7.4 Example of Origin switch zero-return configuration

Connect the origin switch to the corresponding port.
When the DIP switches PIN3 and PIN2 of the 57D motor are switched to the ON position, 5V power is supplied to the external switch.

The configuration steps are as follows:

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set trigger level , homing direction , homing speed , homing mode.
   01 90 00 00 00 64 00 00 F5
3. Perform "return to zero"
   01 91 92

It can be observed that: the motor (100RPM) rotates in the forward direction → the switch is triggered, and the return to zero is completed.



Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data
14:18:55.293	candle0	tx	0x001			3	82 05 88
14:18:55.294	candle0	rx	0x001			3	82 01 84
14:19:17.868	candle0	tx	0x001			8	90 00 00 00 64 00 00 F5
14:19:17.873	candle0	rx	0x001			3	90 01 92
14:19:28.967	candle0	tx	0x001			2	91 92
14:19:28.968	candle0	rx	0x001			3	91 01 93
14:19:47.571	candle0	rx	0x001			3	91 02 94

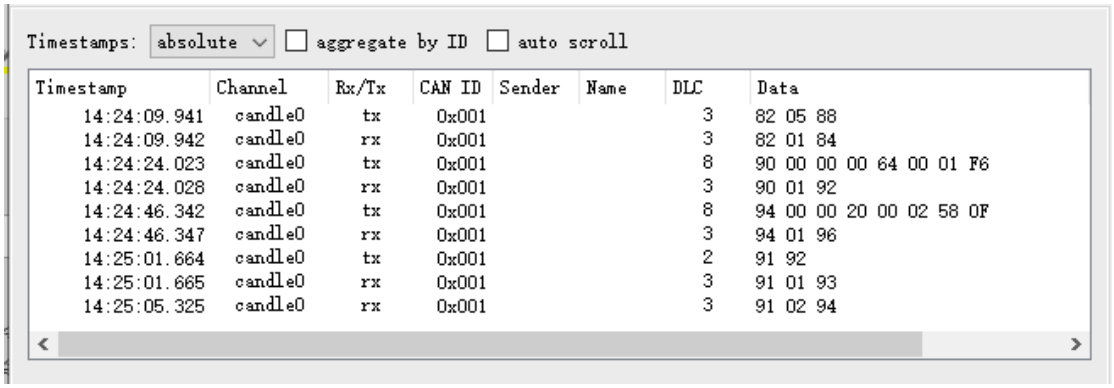
7.5 Example of mechanical limit switch homing configuration

A suitable zero- return current needs to be set for mechanical limit switches. The zero-return current should not be too large, just enough to drive the load, to avoid damaging the mechanical device.

The configuration steps are as follows:

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set the zero-return direction , speed , mode.
   01 90 00 00 00 64 00 01 F6
3. Set the origin offset and the zero-return current.
   01 94 00 00 20 00 02 58 0F
4. Perform "return to zero"
   01 91 92

It can be observed that: the motor (100RPM) rotates forward → after touching the mechanical limit switch → the motor stops → the motor runs in reverse to the position offset from the origin, and the return to zero is completed.



Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data
14:24:09.941	candle0	tx	0x001			3	82 05 88
14:24:09.942	candle0	rx	0x001			3	82 01 84
14:24:24.023	candle0	tx	0x001			8	90 00 00 00 64 00 01 F6
14:24:24.028	candle0	rx	0x001			3	90 01 92
14:24:46.342	candle0	tx	0x001			8	94 00 00 20 00 02 58 0F
14:24:46.347	candle0	rx	0x001			3	94 01 96
14:25:01.664	candle0	tx	0x001			2	91 92
14:25:01.665	candle0	rx	0x001			3	91 01 93
14:25:05.325	candle0	rx	0x001			3	91 02 94

7.6 Example of Single-lap zero-return configuration

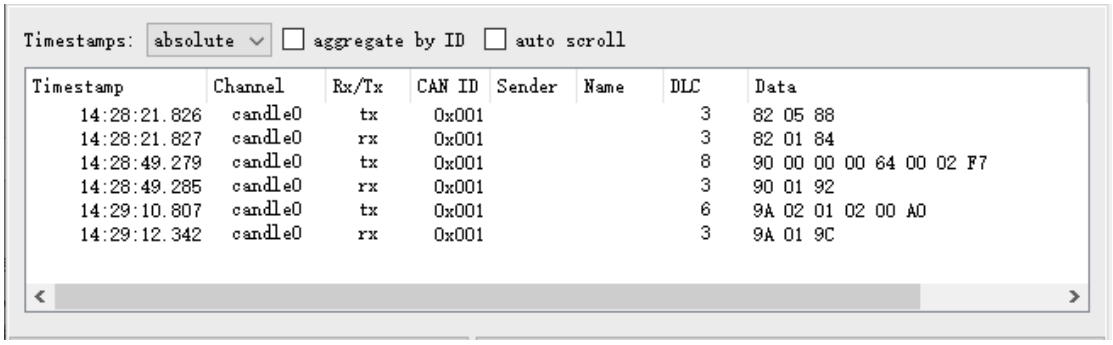
First, move the motor shaft to the appropriate "zero point" position.

The configuration steps are as follows:

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set to zero mode
   01 90 00 00 00 64 00 02 F7
3. Set the single-lap zero-return mode, Zero Point, Speed, Direction .
   01 9A 02 01 02 00 A0 (Nearest possible zero)

After power is cut off, move the motor shaft away from the "zero point" position.

Upon powering back on, it can be observed that the motor moves to the nearest "zero" position and completes the return to zero.



Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data
14:28:21.826	candle0	tx	0x001			3	82 05 88
14:28:21.827	candle0	rx	0x001			3	82 01 84
14:28:49.279	candle0	tx	0x001			8	90 00 00 00 64 00 02 F7
14:28:49.285	candle0	rx	0x001			3	90 01 92
14:29:10.807	candle0	tx	0x001			6	9A 02 01 02 00 A0
14:29:12.342	candle0	rx	0x001			3	9A 01 9C

7.7 Example of coordinate zeroing

Before performing "coordinate zeroing", the coordinates of the "zero point" must be determined:

For switch-based or mechanical limit switch-based zeroing, "origin zeroing" must be performed first.

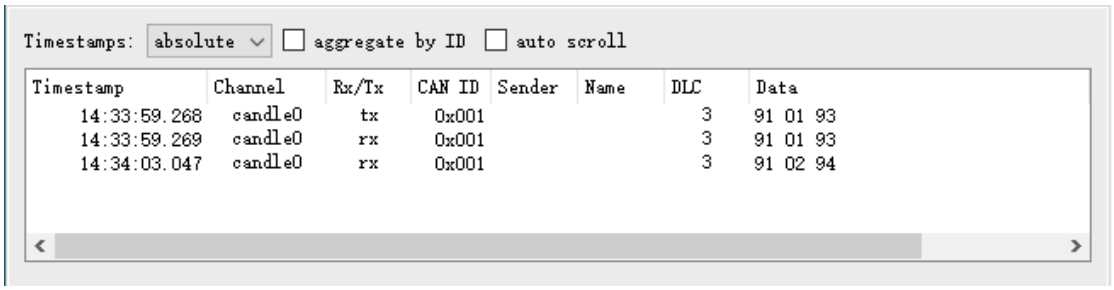
For the single-lap zero-return method, simply set the "zero point" coordinates.

1. Execute "Coordinate Zeroing"

01 91 01 93

Once the "zero point" coordinates are determined, regardless of the motor's position, executing "coordinate return to zero" will cause the motor to return to the zero point position at the zero-point speed .

Note: The motor must be enabled before executing "coordinate return to zero".



Part 8. Left and right limit switch instructions

For limit switch wiring instructions, please refer to section 1.7 IO Port DescriptionDescription

Limit enable description:

1. When limit switch enable is enabled, in bus control mode
   Left limit switch triggered, motor stops moving to the left.
   Right limit switch triggered, motor stops moving to the right.
2. After using the limit function for the first time or changing the limit parameters, you must perform a "return to zero" operation once .
3. The limit function is invalid in pulse control mode.

8.1 Set limit switch parameters

Limit switch parameters include:

Whether the limit port is mapped	is set by 9EH, see 7.3.1.
Whether the limit function is enabled	is set by 90H, see 7.3.2.

8.2 Example of Limit switch configuration

Taking 57D as an example, connect the limit switch to the corresponding port.

When the DIP switches PIN3 and PIN2 of the 57D motor are switched to the ON position, 5V power is supplied to the external switch.

1. Set working mode

   01 82 05 88 (Bus FOC mode)

2. Disable port mapping

   01 9E 00 9F (No mapping required)

3. Set trigger level , zero-return direction , zero-return speed , enable limit switch , and zero-return mode.

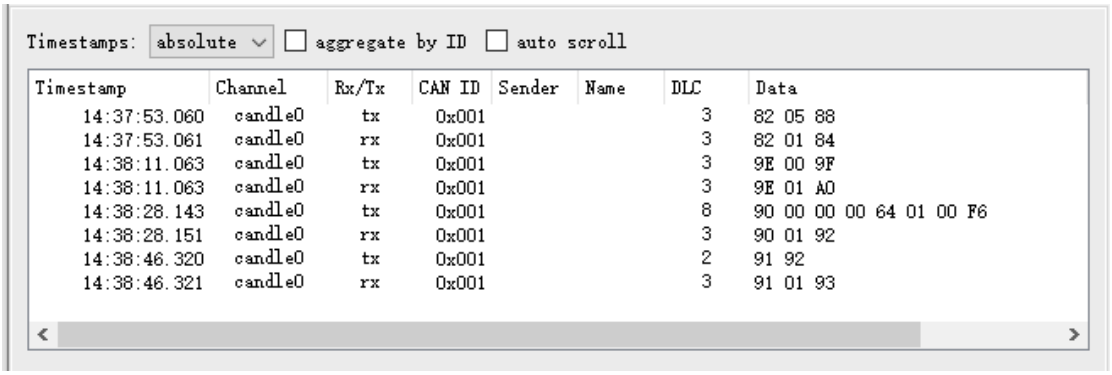
   01 90 00 00 00 64 01 00 F6

4. Perform "return to zero"

   01 91 92

   the origin switch has returned to zero , the limit switch function can be used normally.

If the above parameters remain unchanged, there is no need to reconfigure the parameters or execute "origin return to zero" on the next power-on; the limit switch function can be used directly.



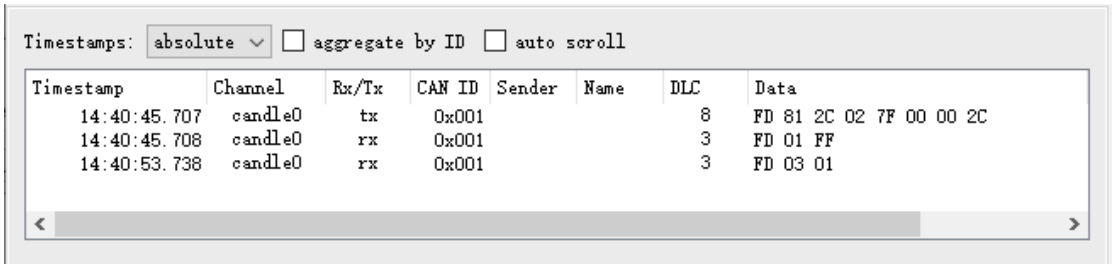
Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data
14:37:53.060	candle0	tx	0x001			3	82 05 88
14:37:53.061	candle0	rx	0x001			3	82 01 84
14:38:11.063	candle0	tx	0x001			3	9E 00 9F
14:38:11.063	candle0	rx	0x001			3	9E 01 A0
14:38:28.143	candle0	tx	0x001			8	90 00 00 00 64 01 00 F6
14:38:28.151	candle0	rx	0x001			3	90 01 92
14:38:46.320	candle0	tx	0x001			2	91 92
14:38:46.321	candle0	rx	0x001			3	91 01 93

8.3 Example of Left limit switch operation

After configuring the parameters according to 8.2

01 FD 81 2C 02 7F 00 00 2C

When the motor is running, triggering the left limit switch will stop the motor.

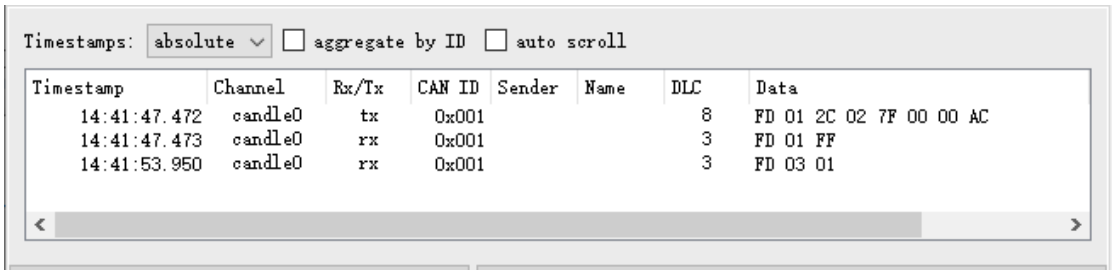


8.4 Example of Right limit switch operation

After configuring the parameters according to 8.2

01 FD 01 2C 02 7F 00 00 AC

When the motor is running, triggering the right limit switch will stop the motor.



Part 9. Bus control mode parameter description

Note: The instructions in this section are only valid in bus control mode (SR_OPEN/SR_CLOSE/ SR_vFOC) .

9.1 Description the parameters of speed and acceleration

1.speed

The speed parameter ranges from 0 to 3000. The larger the value, the faster the motor rotates.

When speed = 0, the motor stops rotating.

The maximum speed of the control mode is as follows:

	Control mode		Max speed
Open mode	Pulse interface	CR_OPEN	400(RPM)
	Serial interface	SR_OPEN	
Close mode	Pulse interface	CR_CLOSE	1500(RPM)
	Serial interface	SR_CLSOE	
FOC mode	Pulse interface	CR_vFOC	3000(RPM)
	Serial interface	SR_vFOC	

If the set speed is greater than the maximum speed of the control mode, the motor runs at the maximum speed of the control mode.

Note: The speed value is calibrated based on 16/32/64 subdivisions, and the speeds of other subdivisions need to be calculated based on 16 subdivisions.

For example, setting speed=1200

At 8 subdivisions, the speed is 2400 (RPM)

At 16/32/64 subdivisions, the speed is 1200 (RPM)

At 128 subdivisions, the speed is 150 (RPM)

2. acceleration

The value of the acceleration(acc) ranges from 0 to 255. The larger the value, the faster the motor accelerates/decelerates.

If acc=0, the motor runs without acceleration or deceleration, and runs directly at the set speed.

① accelerates

Suppose at time t1， the current speed is V_{t1} (V_{t1} < speed)

at time t2， the current speed is V_{t2}

t2 – t1 = （256-acc）* 50 (uS)

The relationship between the current speed V_{ti}, acc, and speed is as follows:

$V_{t2} = V_{t1} + 1$ (V_{t2} <= speed)

For example: acc = 236,speed = 3000

T(ms)	speed （RPM）
0	0
1	1
2	2
3	3
…	…

T(ms)	speed (RPM)
…	…
…	…
2998	2998
2999	2999
3000	3000

② decelerates

Suppose at time t1， the current speed is V_{t1} (V_{t1} > speed)

at time t2， the current speed is V_{t2}

t2 – t1 = （256-acc）* 50 (uS)

The relationship between the current speed V_{ti}, acc, and speed is as follows:

$V_{t2} = V_{t1} - 1$ (V_{t2} >= speed)

9.2 Bus control general instructions

9.2.1 Read motor operating status

The read command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	F1H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Running status	Checksum
			F1H	status (uint8_t)	CRC

status: Motor operating status, corresponding codes are shown in the table below.

Motor operating status	status	Remark
Query failed	0	
stop	1	
speed up	2	
speed down	3	
full speed	4	
homing	5	
Calibration	6	

Note 1: This instruction is only valid in " SR_OPEN/SR_CLOSE/ SR_vFOC " mode.
Note 2: This command can only query motor calibration operation in "CR_OPEN/CR_CLOSE/ CR_vFOC " mode.
Note 3: If this command is sent to a motor without a magnet or a motor far from the drive board, an incorrect return value may occur. In this case, first send the 95H command (setting position reaches threshold) to disable the encoder's judgment, then use open-loop mode to check the motor's operating status. See 5.2.16.

9.2.2 Set motor enable state

In bus control mode, the motor's enable state is no longer controlled by the level of the En pin, but is controlled by the command.

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Work mode	Checksum
			F3H	enable (uint8_t)	CRC

enable = 00 Motor is disabled (loose shaft).
enable = 01 Motor enable (shaft lock)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			F3H	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

Note: This instruction is only valid in bus control mode.

9.2.3 Motor emergency stop command

The emergency stop command is as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
01		2	F7H	CRC

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Stopped state	Checksum
			F7H	status (uint8_t)	CRC

status = 0 Emergency stop failed
status = 1 Emergency stop successful

Note: Emergency stop commands are not recommended when the motor speed exceeds 1000 RPM !

Part 10. Speed Control Mode Description

In speed control mode,

It can control the motor to run continuously at a set acceleration and speed.

It can control the motor to run at a set acceleration and speed for a set period of time.

It allows the motor to run at a set speed and acceleration as soon as it is turned on.

Note: The instructions in this section are only valid in bus control mode (SR_OPEN/SR_CLOSE/ SR_vFOC) .

10.1 Speed control mode operation instructions

The execution command format is as follows:

Downlink message (host computer → driver board)									
ID		DLC	Byte1	Byte2			Byte3	Byte4	Byte5
01	...	5	Code	direction	reserve	speed		acceleration	Checksum
			F6H	b7	b6-b4	b3-b0	b7-b0	acc	CRC
				dir	——	speed			

Byte 2: The highest bit indicates the direction, the lower 4 bits and Byte 3 together indicate the speed

Byte 3: The lower 4 bits of Byte3 and Byte2 together indicate speed

The parameter description is as follows:

dir: the value range is 0/1 (CCW/CW)

speed: the speed, the value range is 0-3000

acc: the acceleration, the value range is 0-255

for example:

Send 01 F6 01 40 02 3A , the motor rotates forward at an acc= 2 , speed= 320 (RPM).

Send 01 F6 81 40 02 BA , the motor reverses at acc= 2 , speed= 320 (RPM).

If you want the motor to automatically stop after running for a period of time, the operating command is as follows:

Downlink message (host computer → driver board)									
ID		DLC	Byte1	Byte2			Byte3	Byte4	Bytes 5-7	Byte8
01	...	8	Code	dir	reserve	speed		acc	runtime	crc
			F6H	b7	b6-b4	b3-b0	b7-b0	acc	runTime	CRC
				dir	——	speed				

runTime unit: 10ms , range (0 ~ 0xFFFFFF)

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	...	3	Code	Running status	Checksum
			F6H	status (uint8_t)	CRC

status = 0 Running failed
status = 1 Start running
status = 2 Running completed
status = 5 Data has received for Synchronous mode.

10.2 Speed control mode stop command

The stop command is as follows:

Downlink message (host computer → driver board)								
CAN ID	...	DLC	Byte1	Byte2			Byte3	Byte4	Byte5
01	...	5	Code	direction	reserve	speed		acceleration	Checksum
			F6H	b7	b6-b4	b3-b0	b7-b0	acc	CRC

The stop command can control the motor to slow down and stop gradually, or it can control the motor to stop immediately.
When setting acc ≠ 0, the motor decelerates and stops slowly
When setting acc = 0, the motor stops immediately
① Deceleration and slow stop command (acc ≠ 0)
for example:
Send 01 F6 0 0 00 02 F9
stop rotating at a deceleration of acc=2.
② Immediate stop command (acc = 0)
for example:
Send 01 F6 0 0 00 00 F7
Stop the motor immediately
Note: It is not recommended to use the immediate stop command when the motor speed exceeds 1000 RPM !

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01	...	3	Code	Stopped state	Checksum
			F6H	status (uint8_t)	CRC

status = 0 Stop failure
status = 1 Stop starting
status = 2 Stopped completing

Note: You can also use the emergency stop command F7H to stop the operation. See 9.2.3.

10.3 Set automatic start command upon power-on

After setting automatic operation, the motor will automatically run in the set direction, speed, and acceleration every time the machine is turned on.

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Enable	Checksum
			FFH	enable (uint8_t)	CRC

enable = C8 Enable automatic operation upon power-on
enable = CA Turn off automatic operation upon power-on.

To enable automatic operation upon power-on, you must first send a speed control mode operation command to make the motor run in the desired direction/speed/acceleration. Then, use this command to enable automatic operation upon power-on. After power-on, the motor will run according to the saved parameters.

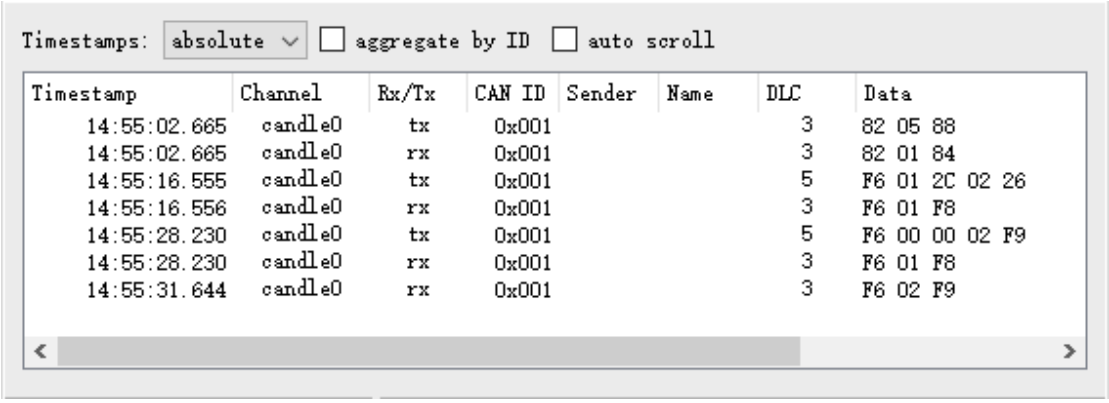
Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	state	Checksum
			FFH	status (uint8_t)	CRC

status = 0 Setting failed
status = 1 Start setting
status = 2 Setup complete

10.4 Example of Speed mode running

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set the running direction , speed , acceleration parameters.
   01 F6 01 2C 02 26
   At this point, the motor can be observed to run in the set direction, speed, and acceleration.
   To slow down and stop operation, execute the stop command.
3. Stop command
   01 F6 00 00 02 F9
   At this point, the motor can be observed to stop at the set acceleration.



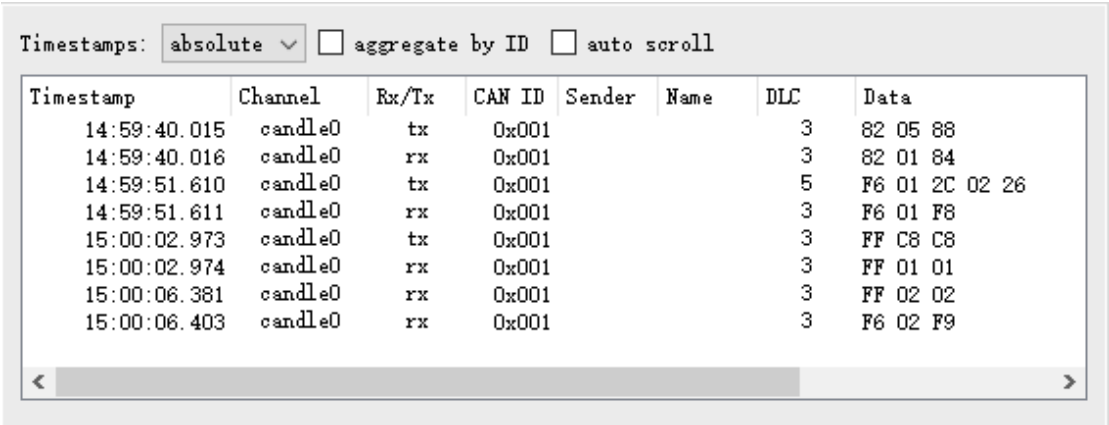
Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data
14:55:02.665	candle0	tx	0x001			3	82 05 88
14:55:02.665	candle0	rx	0x001			3	82 01 84
14:55:16.555	candle0	tx	0x001			5	F6 01 2C 02 26
14:55:16.556	candle0	rx	0x001			3	F6 01 F8
14:55:28.230	candle0	tx	0x001			5	F6 00 00 02 F9
14:55:28.230	candle0	rx	0x001			3	F6 01 F8
14:55:31.644	candle0	rx	0x001			3	F6 02 F9

10.5 Example of automatic operation upon power-on

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set the running direction , speed , and acceleration parameters.
   01 F6 01 2C 02 26
   At this point, the motor can be observed to run in the set direction, speed, and acceleration.
3. Enable automatic operation upon power-on
   01 FF C8 C8
   At this point, the motor will slow down and stop.

After powering on again, the motor can be observed to run in the set direction, speed, and acceleration.

To cancel the automatic start-up function, the command is as follows:
01 FF CA CA



Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data
14:59:40.015	candle0	tx	0x001			3	82 05 88
14:59:40.016	candle0	rx	0x001			3	82 01 84
14:59:51.610	candle0	tx	0x001			5	F6 01 2C 02 26
14:59:51.611	candle0	rx	0x001			3	F6 01 F8
15:00:02.973	candle0	tx	0x001			3	FF C8 C8
15:00:02.974	candle0	rx	0x001			3	FF 01 01
15:00:06.381	candle0	rx	0x001			3	FF 02 02
15:00:06.403	candle0	rx	0x001			3	F6 02 F9

Part 11. Position control mode description

In position control mode, the command " 8CH " can be used to set whether to return to the running state.

If you are using a motor without a magnet or a motor far from the drive board, sending this command may result in an incorrect return value. You should first send the 95H command (setting position reaches threshold) to disable the encoder's judgment, then use open-loop mode to check the motor's operating status. See 5.2.16..

Note: The instructions in this section are only valid in bus control mode (SR_OPEN/SR_CLOSE/ SR_vFOC) .

11.1 Relative motion according to pulse number

In this mode, the motor can be controlled to run to a specified position relative to the set acceleration and speed, based on the number of pulses. It is suitable for controlling the number of revolutions required for the motor to run.

11.1.1 Pulse relative operation command

The execution command format is as follows:

<table>
<tr><td colspan="10">Downlink message (host computer → driver board)</td></tr>
<tr><td>ID</td><td rowspan="3">...</td><td>DLC</td><td>Byte1</td><td colspan="3">Byte2</td><td>Byte3</td><td>Byte4</td><td>Bytes5-7</td><td>Byte8</td></tr>
<tr><td rowspan="3">01</td><td rowspan="3">8</td><td>Code</td><td>DIR</td><td colspan="2">reserve</td><td colspan="2">speed</td><td>acceleration</td><td>Location</td><td>Checksum</td></tr>
<tr><td rowspan="2">FDH</td><td>b7</td><td>b6-b4</td><td>b3-b0</td><td>b7-b0</td><td rowspan="2">acc</td><td rowspan="2">relpulses</td><td rowspan="2">CRC</td></tr>
<tr><td>dir</td><td colspan="2">——</td><td colspan="2">speed</td></tr>
</table>

Byte2: The highest bit indicates the direction, the lower 4 bits and together indicate the speed

Byte3: The lower 4 bits of Byte3 and Byte2 together indicate speed

The parameter description is as follows:

dir: the value range is 0/1 (CCW/CW)

speed: the speed, the value range is 0-3000 (RPM)

acc: the acceleration, the value range is 0-255

relpulses: the motor run steps, the range is 0 - 0xFFFFFF

for example:

send 01 FD 01 40 02 00 FA 00 3B ,

the motor rotates 20 revolutions in the forward direction (16 microsteps) with acc= 2 and speed= 0x140 ;

send 01 FD 81 40 02 00 FA 00 BB ,

the motor rotates in the opposite direction for 20 revolutions (16 microsteps) with acc= 2 and speed= 0x140 ;

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Running status	Checksum
			FDH	status (uint8_t)	CRC

The operation failed at position status = 0.
status = 0 run fail.
status = 1 run starting….
status = 2 run complete.
status = 3 end limit stoped.
status = 5 Data has received for Synchronous mode.

Note 1: The 8CH instruction can be set to whether to return data.
Note 2: If the position set by the 95H instruction reaches the threshold, it may not return "Position completed".

11.1.2 Stop command

The stop command is as follows:

Downlink message (host computer → driver board)									
ID	...	DLC	Byte1	Byte2			Byte3	Byte4	Bytes5-7	Byte8
01	...	8	Code	dir	reserve	speed		ACC	Location	Checksum
			FDH	b7	b6-b4	b3-b0	b7-b0	acc	0	CRC
				0	0	0				

The stop command can control the motor to slow down and stop gradually, or it can control the motor to stop immediately.
When setting acc ≠ 0, the motor decelerates and stops slowly
When setting acc = 0, the motor stops immediately

① Slow down and stop command (acc ≠ 0)
for example:
Send 01 FD 00 00 04 00 00 00 02
Let the motor decelerate at an acceleration of acc = 4. Stop rotating

② Immediate stop command (acc = 0)
for example:
Send 01 FD 00 00 00 00 00 00 FE
Stop the motor immediately

Note: It is not recommended to use the immediate stop command when the motor speed exceeds 1000 RPM !

Returned data:

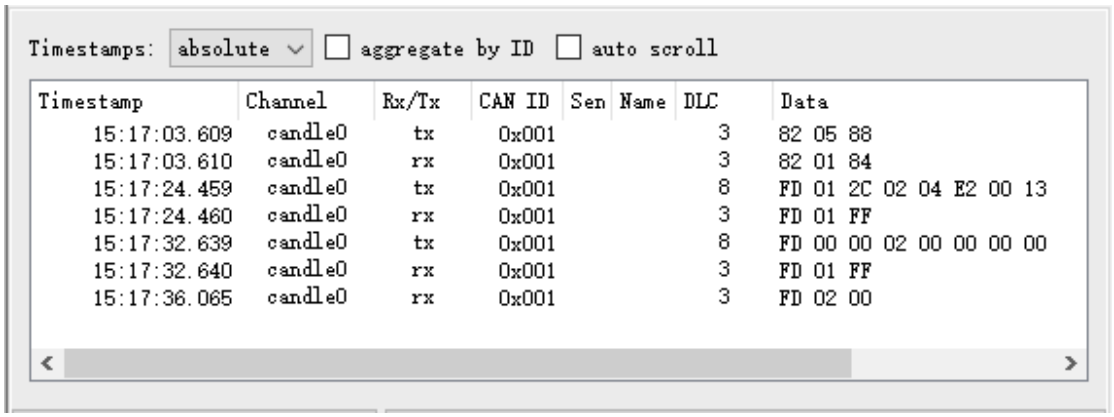
Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Stopped state	Checksum
			FD H	status (uint8_t)	CRC

status = 0 stop the motor fail.
status = 1 stop the motor starting….
status = 2 stop the motor complete.
status = 3 stop by endstop.

Note: You can also use the emergency stop command F7H to stop the operation. See 9.2.3.

11.1.3 Example of Pulse Relative Operation

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set the running direction, speed , acceleration , and relative pulse count.
   01 FD 01 2C 02 04 E2 00 13
   (At speed = 300 (RPM) , acceleration = 2 , 100 revolutions clockwise (16 subdivisions))
3. If you need to slow down and stop the machine during operation, the command is as follows:
   01 FD 00 00 02 00 00 00 00
4. If an emergency stop is required during operation, the command is as follows:
   01 F7 F8



Timestamp	Channel	Rx/Tx	CAN ID	Sen	Name	DLC	Data
15:17:03.609	candle0	tx	0x001			3	82 05 88
15:17:03.610	candle0	rx	0x001			3	82 01 84
15:17:24.459	candle0	tx	0x001			8	FD 01 2C 02 04 E2 00 13
15:17:24.460	candle0	rx	0x001			3	FD 01 FF
15:17:32.639	candle0	tx	0x001			8	FD 00 00 02 00 00 00 00
15:17:32.640	candle0	rx	0x001			3	FD 01 FF
15:17:36.065	candle0	rx	0x001			3	FD 02 00

11.2 Absolute motion based on pulse count

In this mode, the motor can be controlled to run to a specified position at a set acceleration and speed, based on the absolute number of pulses. It is suitable for controlling the number of revolutions required for the motor to run.

Note: Before using absolute motion, a zeroing process must be performed once (instruction 0x9 1 or 0x9 2) to mark the "zero point".

11.2.1 Pulse Absolute Operation Command

The execution command format is as follows:

Downlink message (host computer → driver board)								
ID		DLC	Byte1	Byte2	Byte3	Byte4	Bytes 5-7	Byte8
01	...	8	Code	speed		acceleration	Absolute Pulse	Checksum
			FEH	speed		acc	absPulses	CRC

The parameters are explained below:

speed range from 0 to 3000 (RPM)

acc Accelerator range of 0 - 255.

absPulses the absolute pulse count, with a value range of int24_t (-8388607 , +8388607) .

For example, the current absolute pulse count can be any value.

Send 01 FE 02 58 02 00 0C 80 E7

The motor runs at a speed of 600 (RPM) and an acceleration of 2 , reaching absolute 0x0C80 .

That is, after the motor finishes running, the current absolute pulse count becomes 0x0C80 .

For example, the current absolute pulse count can be any value.

Send 01 FE 02 58 02 FF F3 80 CD

The motor runs at a speed of 600 (RPM) and an acceleration of 2 , with an absolute value of -0x0C80 .

That is, after the motor finishes running, the current absolute pulse count becomes -0x0C80 .

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Stopped state	Checksum
			FEH	status (uint8_t)	CRC

status = 0 run fail.
status = 1 run starting….
status = 2 run complete.
status = 3 end limit stoped.
status = 5 Data has received for Synchronous mode.

Note 1: The 8CH instruction can be set to whether to return data.
Note 2: If the position set by the 95H instruction reaches the threshold, it may not return "Position completed".

11.2.2 Stop command

The stop command is as follows:

Downlink message (host computer → driver board)								
ID	…	DLC	Byte1	Byte2	Byte3	Byte4	Bytes 5-7	Byte8
01		8	Code	speed		acceleration	Absolute Pulse	Checksum
			FEH	0		acc	0	CRC

The stop command can control the motor to slow down and stop gradually, or it can control the motor to stop immediately.
When setting acc ≠ 0, the motor decelerates and stops slowly
When setting acc = 0, the motor stops immediately

① Slow down and stop command (acc ≠ 0)
for example:
Send 01 FE 00 00 **04** 00 00 00 03
Let the motor decelerate at an acceleration of acc = 4. Stop rotating

② Immediate stop command (acc = 0)
for example:
Send 01 FE 00 00 **00** 00 00 00 FF
Stop the motor immediately

Note: It is not recommended to use the immediate stop command when the motor speed exceeds 1000 RPM !

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	…	DLC	Byte1	Byte2	Byte3
01		3	Code	Stopped state	Checksum
			FEH	status (uint8_t)	CRC

status = 0 stop the motor fail.
status = 1 stop the motor starting….
status = 2 stop the motor complete.
status = 3 stop by endstop.

Note: You can also use the emergency stop command F7H to stop the operation. See 9.2.3.

11.2.3 Example of Pulse Absolute Operation

When operating in absolute pulse mode, the "zero point" pulse position needs to be set first. There are two ways to set it:

1. "Return to Zero" setting;
2. Setting the "92H command mode";

If not set, the default power-on location is "midnight".

1. Set working mode

   01 82 05 88 (Bus FOC mode)

2. Set zero point

   01 92 93

3. Set the running speed , acceleration , and absolute pulse count.

   01 FE 01 2C 02 01 00 00 2F

   (Run at speed = 300 RPM , acc = 2 , until pulse position 0x10000)

4. After the process is complete, you can view the number of pulses.

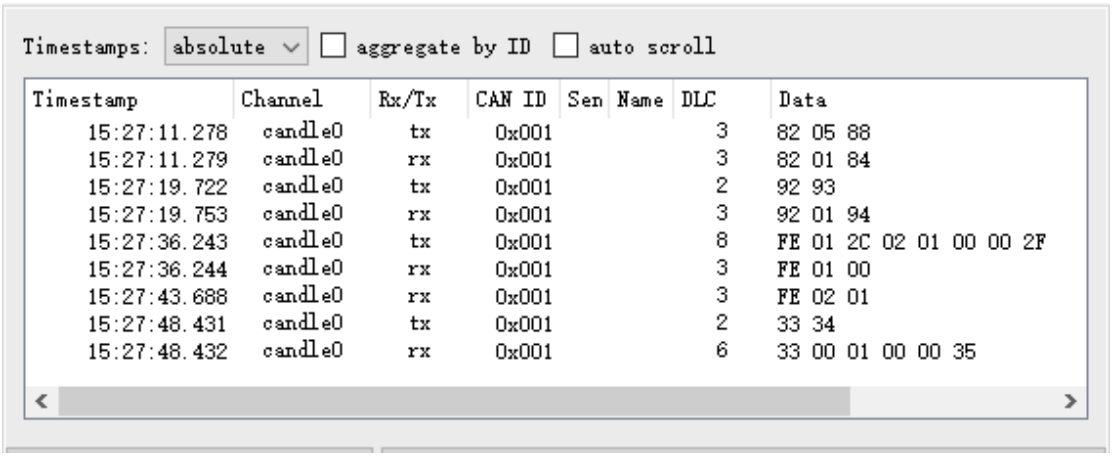
   01 33 34

5. If you need to slow down and stop the machine during operation, the command is as follows:

   01 FE 00 00 02 00 00 00 01

6. If an emergency stop is required during operation, the command is as follows:

   01 F7 F8



Timestamp	Channel	Rx/Tx	CAN ID	Sen	Name	DLC	Data
15:27:11.278	candle0	tx	0x001			3	82 05 88
15:27:11.279	candle0	rx	0x001			3	82 01 84
15:27:19.722	candle0	tx	0x001			2	92 93
15:27:19.753	candle0	rx	0x001			3	92 01 94
15:27:36.243	candle0	tx	0x001			8	FE 01 2C 02 01 00 00 2F
15:27:36.244	candle0	rx	0x001			3	FE 01 00
15:27:43.688	candle0	rx	0x001			3	FE 02 01
15:27:48.431	candle0	tx	0x001			2	33 34
15:27:48.432	candle0	rx	0x001			6	33 00 01 00 00 35

11.3 Relative motion according to coordinate values

In this mode, the motor can be controlled to move to a specified position relative to the coordinate values at a set acceleration and speed. It is suitable for controlling the motor to move to a specified coordinate position.
Note: The coordinate values are the cumulative multi-turn encoder values (16384/ turn), read using the command "3 1 H".

11.3.1 Coordinate relative running instructions

The execution command format is as follows:

Downlink message (host computer → driver board)								
ID		DLC	Byte1	Byte2	Byte3	Byte4	Bytes 5-7	Byte8
01	…	8	Code	speed		acceleration	relative coordinates	Checksum
			F4H	speed		acc	relAxis	CRC

The parameters are explained below:
speed range 0 to 3000 (RPM)
acc accelerator range 0 - 255.
relAxis the relative coordinate value, with a range of int24_t (-8388607 , +8388607) .

For example, the current coordinate is 0x8000
Send 01 F4 02 58 02 00 40 00 91
The motor operates at a speed of 600 RPM, an acceleration of 2, and a relative speed of 0x4000 .
That is, after the motor finishes running, the current coordinate becomes 0xC000 .

For example, the current coordinate is 0x8000
Send 01 F4 02 58 02 FF FF C0 00 09
The motor operates at a speed of 600 RPM, an acceleration of 2, and a relative speed of -0x4000 .
That is, after the motor finishes running, the current coordinate becomes 0x4000 .

Note: The "31H" command can read coordinate values.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Running status	Checksum
			F4H	status (uint8_t)	CRC

The operation failed at position status = 0.

status = 0 run fail.

status = 1 run starting….

status = 2 run complete.

status = 3 end limit stoped.

status = 5 Data has received for Synchronous mode.

Note 1: The 8CH instruction can be set to whether to return data.

Note 2: If the position set by the 95H instruction reaches the threshold, it may not return "Position completed".

11.3.2 Stop command

The stop command is as follows:

Downlink message (host computer → driver board)								
ID		DLC	Byte1	Byte2	Byte3	Byte4	Bytes 5-7	Byte8
01	…	8	Code	speed		acceleration	relative coordinates	Checksum
			F4H	0		acc	0	CRC

The stop command can control the motor to slow down and stop gradually, or it can control the motor to stop immediately.
When setting acc ≠ 0, the motor decelerates and stops slowly
When setting acc = 0, the motor stops immediately

① Slow down and stop command (acc ≠ 0)
for example:
Send 01 F4 00 00 **04** 00 00 00 F9
Let the motor decelerate at an acceleration of acc = 4. Stop rotating

② Immediate stop command (acc = 0)
for example:
Send 01 F4 00 00 **00** 00 00 00 F5
Stop the motor immediately

Note: It is not recommended to use the immediate stop command when the motor speed exceeds 1000 RPM !

Returned data:

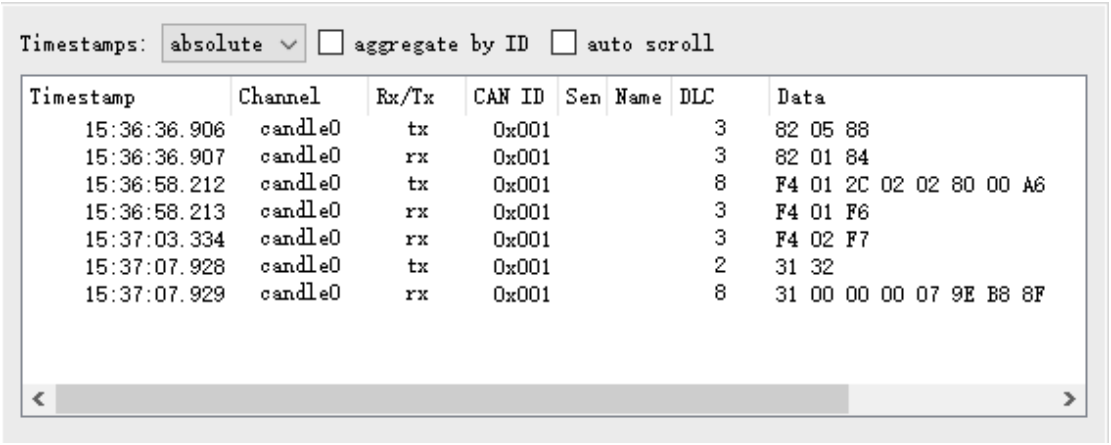
Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Stopped state	Checksum
			F4H	status (uint8_t)	CRC

status = 0 stop the motor fail.
status = 1 stop the motor starting….
status = 2 stop the motor complete.
status = 3 stop by endstop.

Note: You can also use the emergency stop command F7H to stop the operation. See 9.2.3.

11.3.3 Example of coordinate relative operation

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set the running speed , acceleration , and relative coordinates.
   01 F4 01 2C 02 02 80 00 A6
   (speed = 300 (RPM) , acc = 2 , relative coordinates 0x28000)
3. After the process is complete, you can view the coordinates.
   01 31 32
4. If you need to slow down and stop the machine during operation, the command is as follows:
   01 F4 00 00 02 00 00 00 F6
5. If an emergency stop is required during operation, the command is as follows:
   01 F7 F8



Timestamp	Channel	Rx/Tx	CAN ID	Sen	Name	DLC	Data
15:36:36.906	candle0	tx	0x001			3	82 05 88
15:36:36.907	candle0	rx	0x001			3	82 01 84
15:36:58.212	candle0	tx	0x001			8	F4 01 2C 02 02 80 00 A6
15:36:58.213	candle0	rx	0x001			3	F4 01 F6
15:37:03.334	candle0	rx	0x001			3	F4 02 F7
15:37:07.928	candle0	tx	0x001			2	31 32
15:37:07.929	candle0	rx	0x001			8	31 00 00 00 07 9E B8 8F

11.4 Absolute motion based on coordinate values

In this mode, the motor can be controlled to move to a specified position with a set acceleration and speed, based on the coordinate values. It is suitable for controlling the motor to move to a specified coordinate position.

Note 1: The coordinate values are the cumulative multi-turn encoder values (16384/ turn), read using the command "3 1 ".

Note 2 : Supports real-time updates of speed and coordinates, meaning that a new command can be issued to change the speed and coordinates while the previous command is running.

Note 3: Before using absolute motion, a zeroing process must be performed once (instruction 0x9 1 or 0x9 2) to mark the "zero point".

11.4.1 absolute coordinate execution command

The execution command format is as follows:

Downlink message (host computer → driver board)								
ID		DLC	Byte1	Byte2	Byte3	Byte4	Bytes 5-7	Byte8
01	...	8	Code	speed		acceleration	relative coordinates	Checksum
			F5H	speed		acc	absAxis	CRC

The parameters are explained below:

speed range 0 to 3000 (RPM)

acc accelerator range 0 - 255.

absAxis the absolute coordinate value, with a range of int24_t (-8388607 , +8388607) .

For example, the current coordinates are any value

Send 01 F5 02 58 02 00 40 00 92

The motor runs at a speed of 600 (RPM) with an acceleration of 2 , and absolutely reaches 0x4000 .

That is, after the motor finishes running, the current coordinate becomes 0x4000 .

For example, the current coordinates are any value

Send 01 F5 02 58 02 FF C0 00 11

The motor operates at a speed of 600 RPM, with an acceleration of 2 , and an absolute speed of -0x4000 .

That is, after the motor finishes running, the current coordinate becomes -0x4000 .

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Running status	Checksum
			F5 H	status (uint8_t)	CRC

The operation failed at position status = 0.

status = 0 run fail.

status = 1 run starting….

status = 2 run complete.

status = 3 end limit stoped.

status = 5 Data has received for Synchronous mode.

Note 1: The 8CH instruction can be set to whether to return data.

Note 2: If the position set by the 95H instruction reaches the threshold, it may not return "Position completed".

11.4.2 Stop command

The stop command is as follows:

Downlink message (host computer → driver board)								
ID		DLC	Byte1	Byte2	Byte3	Byte4	Bytes 5-7	Byte8
01	…	8	Code	speed		acceleration	relative coordinates	Checksum
			F5H	0		acc	0	CRC

The stop command can control the motor to slow down and stop gradually, or it can control the motor to stop immediately.
When setting acc ≠ 0, the motor decelerates and stops slowly
When setting acc = 0, the motor stops immediately

① Slow down and stop command (acc ≠ 0)
for example:
Send 01 F5 00 00 04 00 00 00 FA
Let the motor decelerate at an acceleration of acc = 4. Stop rotating

② Immediate stop command (acc = 0)
for example:
Send 01 F5 00 00 00 00 00 00 F6
Stop the motor immediately

Note: It is not recommended to use the immediate stop command when the motor speed exceeds 1000 RPM !

Returned data:

Uplink message (host computer ← driver board)					
CAN ID		DLC	Byte1	Byte2	Byte3
01	…	3	Code	Stopped state	Checksum
			F5 H	status (uint8_t)	CRC

status = 0 stop the motor fail.
status = 1 stop the motor starting….
status = 2 stop the motor complete.
status = 3 stop by endstop.

Note: You can also use the emergency stop command F7H to stop the operation. See 9.2.3.

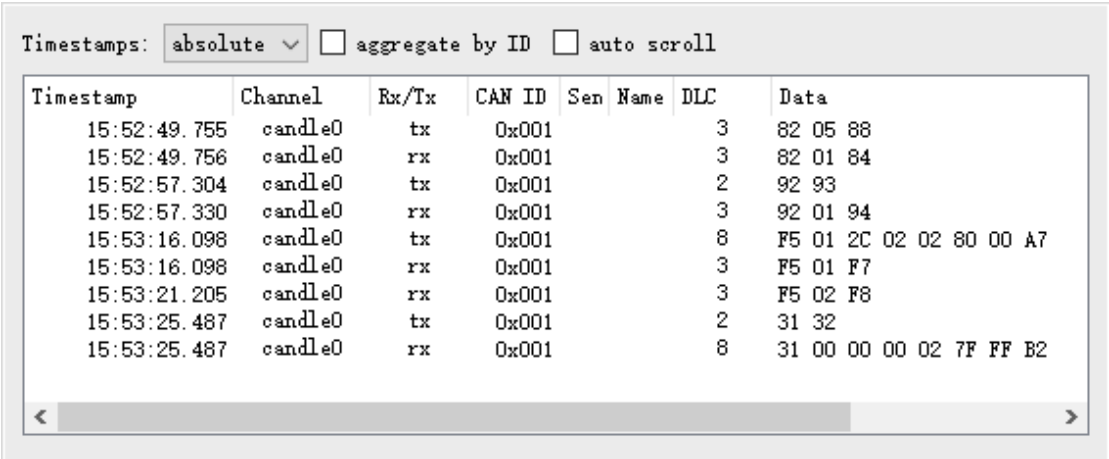
11.4.3 Example of running with absolute coordinates

You need to first set the "zero point" coordinates. There are two ways to do this:

1. "Return to Zero" setting;
2. Setting the "92H command mode";

If not set, the default power-on position is the "zero point" coordinate.

1. Set working mode
   01 82 05 88 (Bus FOC mode)
2. Set zero point
   01 92 93
3. Set the running speed , acceleration , and absolute coordinates.
   01 F5 01 2C 02 02 80 00 A7
   (Run at speed = 30 0 RPM , a cc = 2 , to the absolute coordinate position 0x28000)
4. After the process is complete, you can view the coordinates.
   01 31 32
5. If you need to slow down and stop the machine during operation, the command is as follows:
   01 F5 00 00 02 00 00 00 00 F8
6. If an emergency stop is required during operation, the command is as follows:
   01 F7 F8



Timestamp	Channel	Rx/Tx	CAN ID	Sen	Name	DLC	Data
15:52:49.755	candle0	tx	0x001			3	82 05 88
15:52:49.756	candle0	rx	0x001			3	82 01 84
15:52:57.304	candle0	tx	0x001			2	92 93
15:52:57.330	candle0	rx	0x001			3	92 01 94
15:53:16.098	candle0	tx	0x001			8	F5 01 2C 02 02 80 00 A7
15:53:16.098	candle0	rx	0x001			3	F5 01 F7
15:53:21.205	candle0	rx	0x001			3	F5 02 F8
15:53:25.487	candle0	tx	0x001			2	31 32
15:53:25.487	candle0	rx	0x001			8	31 00 00 00 02 7F FF B2

11.4.4 Example of Real-time updates of running

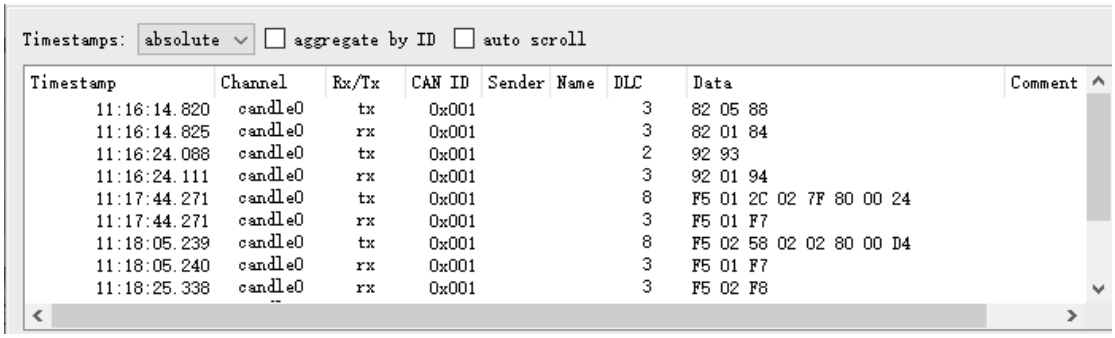
This mode supports real-time updates of speed and coordinates, meaning that new commands can be issued to change speed and coordinates while the previous command is running.

You need to first set the "zero point" coordinates. There are two ways to do this:

1. "Return to Zero" setting;
2. Setting the "92H command mode";

If not set, the default power-on position is the "zero point" coordinate.

1. **Set working mode**
   01 82 05 88 (Bus FOC mode)
2. **Set zero point**
   01 92 93
3. **Set the running speed , acceleration , and absolute coordinates.**
   01 F5 01 2C 02 7F 80 00 24
   (Run at speed = 30 0 RPM , a cc = 2 , to the absolute coordinate position 0x7F8000)
   Wait for the motor to run for about 20 seconds.
4. **Real-time update speed , absolute coordinates**
   01 F5 02 58 02 02 80 00 D4
   (Updated to absolute coordinates 0x28000 with speed = 60 0 RPM and a cc = 2)
   The motor update speed and coordinate operation can be observed.
5. **After the process is complete, you can view the coordinates.**
   01 31 32
6. **If you need to slow down and stop the machine during operation, the command is as follows:**
   01 F5 00 00 02 00 00 00 00 F8
7. **If an emergency stop is required during operation, the command is as follows:**
   01 F7 F8



Timestamp	Channel	Rx/Tx	CAN ID	Sender	Name	DLC	Data	Comment
11:16:14.820	candle0	tx	0x001			3	82 05 88	
11:16:14.825	candle0	rx	0x001			3	82 01 84	
11:16:24.088	candle0	tx	0x001			2	92 93	
11:16:24.111	candle0	rx	0x001			3	92 01 94	
11:17:44.271	candle0	tx	0x001			8	F5 01 2C 02 7F 80 00 24	
11:17:44.271	candle0	rx	0x001			3	F5 01 F7	
11:18:05.239	candle0	tx	0x001			8	F5 02 58 02 02 80 00 D4	
11:18:05.240	candle0	rx	0x001			3	F5 01 F7	
11:18:25.338	candle0	rx	0x001			3	F5 02 F8	

Part 12. Multi- motor synchronous motion description

Synchronous motion refers to starting multiple motors connected to a single bus to move simultaneously. There are three ways to achieve this:

Method 1:

Sending a command using broadcast CAN ID 0 will cause all motors on the bus to execute the command, which is suitable for all motors on the bus to execute the same command.

Method 2:

When a command is sent using a group CAN ID, all motors in the same group on the bus will execute it. This is suitable for group execution of commands by motors on the bus.

Method 3:

The multi-motor synchronization flag function is used to enable different motors on the bus to execute different motion commands. This method allows for the control of an unlimited number of motors.

12.1 Multi-motor synchronous motion 1 - Broadcast

To send motor movement commands using broadcast CAN ID 0, such as instructing all motors on the bus to move relative to each other for 3200 pulses, the following command can be sent.

00 FD 01 2C 02 00 0C 80 B8

All motors on the bus will move relative to each other for 3200 pulse positions.

12.2 Multi-motor synchronous motion 2 - grouping method

To send motor movement commands using group CAN IDes, such as instructing all motors with group CAN ID 0x50 on the bus to run relative to each other for 3200 pulses, the following command can be sent.

50 FD 01 2C 02 00 0C 80 08

All motors with group CAN ID 0x50 on the bus will run for 3200 pulse positions.

See the section 5.2.14 for group CAN ID settings .

12.3 Multi-motor synchronous motion 3 - Synchronization mark

After using the synchronization flag function, the motor will not execute the motion command immediately after receiving it. It will wait for the synchronization execution command to be received before it can start executing.
This method allows control of all motors on the bus.

12.3.1 Command to enable/disable synchronization function

The configuration command is as follows:

Downlink message (host computer → driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Setting status	Checksum
			4AH	enable	CRC

enable = 00 disables the multi-motor synchronization flag function (default).
enable = 01 Enables the multi-motor synchronization flag function.

Returned data:

Uplink message (host computer ← driver board)					
CAN ID	...	DLC	Byte1	Byte2	Byte3
01		3	Code	Stopped state	Checksum
			4AH	status (uint8_t)	CRC

status =1 Set success.
status =0 Set fail.

12.3.2 Synchronous execution instructions

The motor only begins to execute the previously received motion commands after receiving the synchronization execution command. The synchronous execution instructions are as follows:

Downlink message (host computer → driver board)				
CAN ID	...	DLC	Byte1	Byte2
00		2	Code	Checksum
			4BH	CRC

Returned data: None.

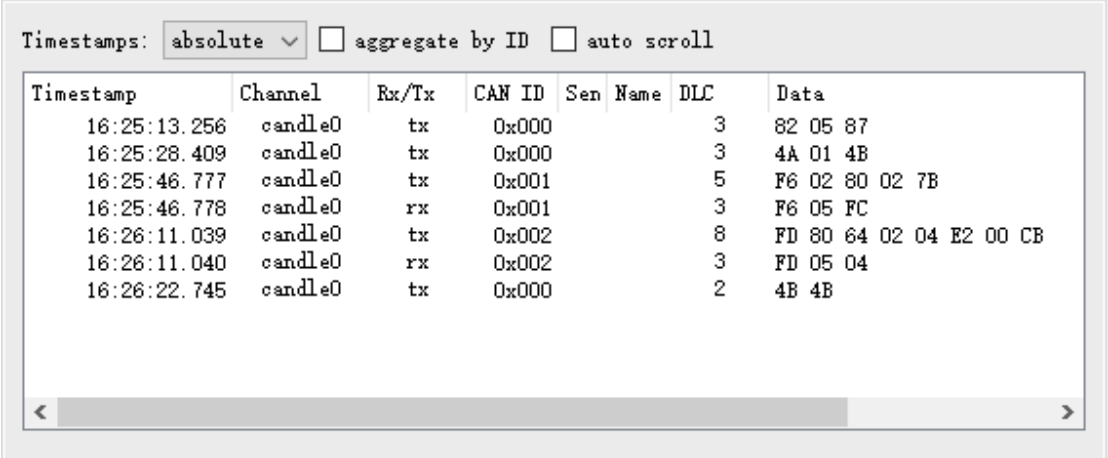
If there are many motors on the bus, some motors may not receive the synchronization execution command due to interference or other reasons. In this case, consider sending the synchronization execution command repeatedly at intervals of about 1ms.

12.3.3 Example of multi-motor synchronous operation

Taking two motors on the bus as an example, they execute different motion commands synchronously.

1. Broadcast settings working mode
   00 82 05 87 (Bus FOC mode)
2. Broadcast enables multi-motor synchronous control function
   00 4A 01 4B
3. Motor 1 setting: Speed mode, forward rotation, speed 640 RPM, acceleration 2
   01 F6 02 80 02 7B
4. Motor 2 settings: relative pulse mode, reverse, speed 100 RPM, acceleration 2 , 320000 pulses.
   02 FD 80 64 02 04 E2 00 CB
5. Send synchronous execution command
   00 4B 4B
   … (Whether it is necessary to send synchronization commands multiple times at 1ms intervals depends on the situation)

It can be observed that all motors begin to run synchronously according to their configuration parameters.



Part 13. CANGAROO usage examples

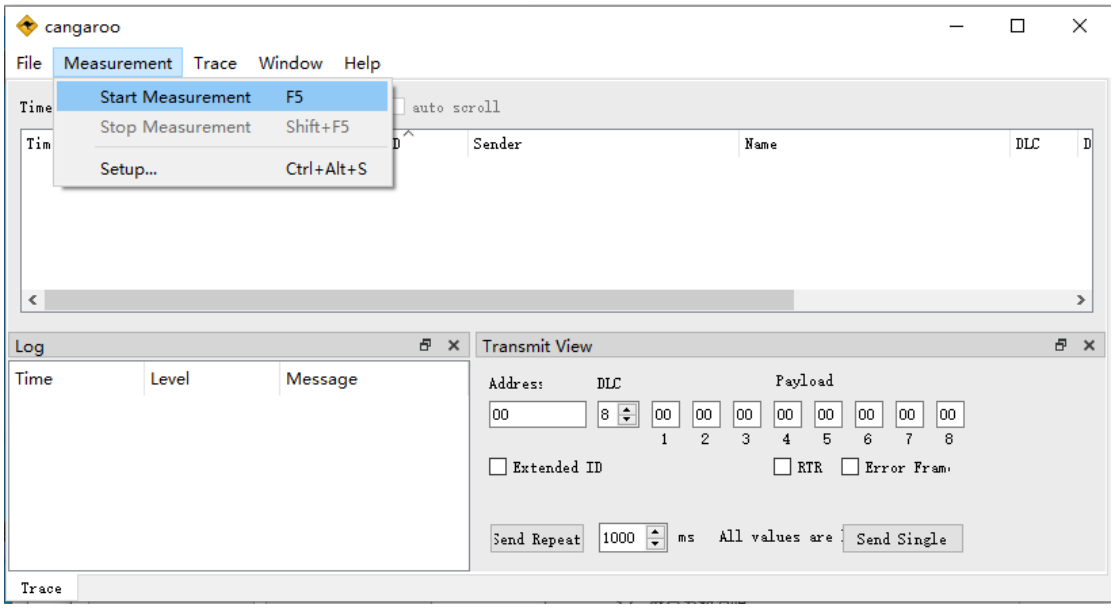
The following example uses the " cangaroo " host computer software and the " MKS CANable " USB to CAN module.

13.1 Motor parameter configuration

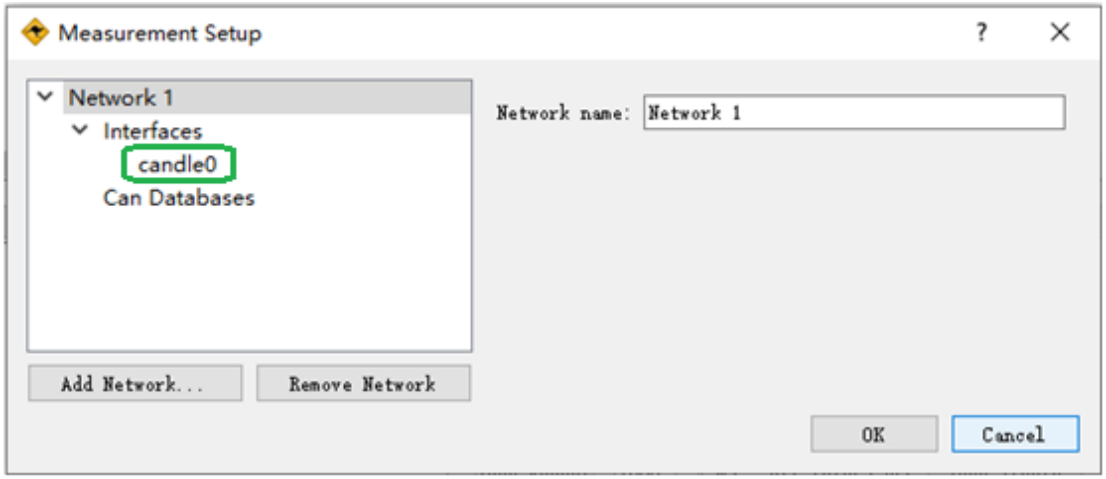
1. Select control mode: Mode -> 05 CAN Bus Closed-Loop FOC Mode
2. Set the baud rate: CanRate -> 500K (default)
3. Set the slave CAN ID: CanID -> 01 (default value)

13.2 cangaroo parameter configuration

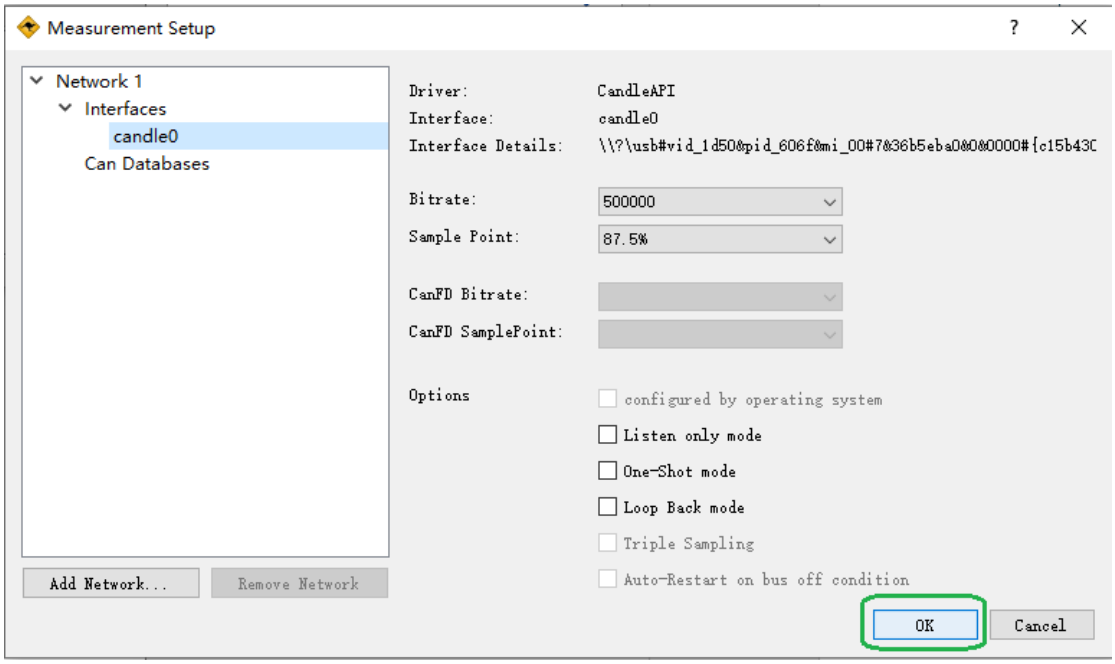
1. Double-click " cangaroo.exe" to run the host computer software;
2. In the Cangaroo window , select the menu " Measurement " - > " Start Measurement ", as shown in the image below.



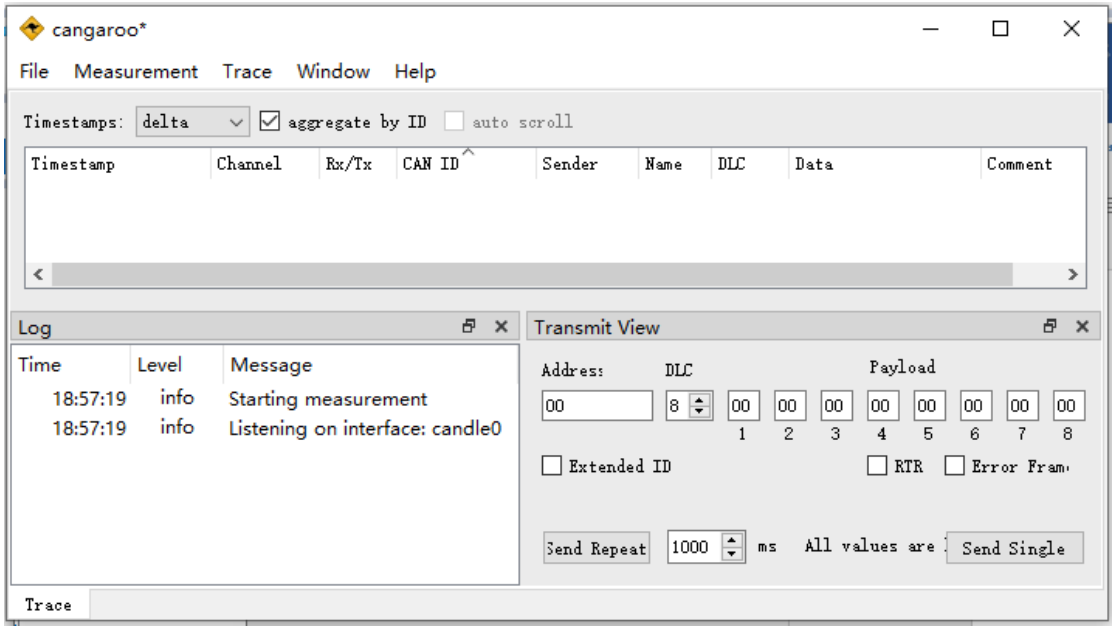
3. In the pop-up Measurement Setup window, click " candle0 ", as shown in the image below.



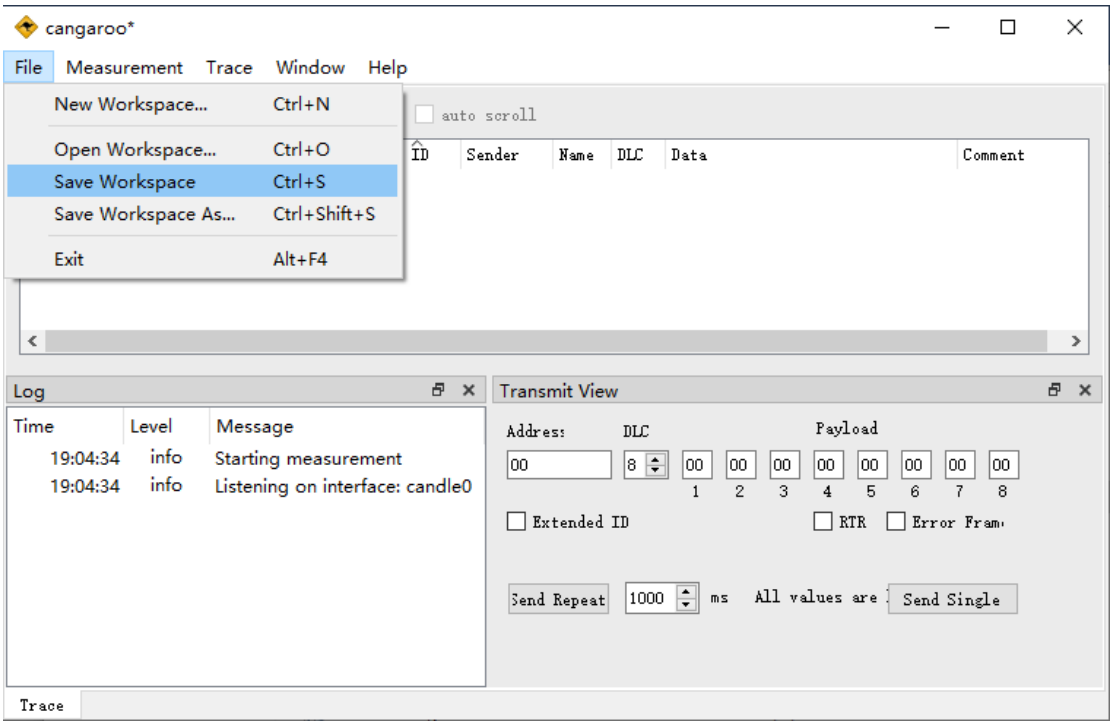
4. Use the default parameters without making any changes, and click "OK", as shown in the image below.



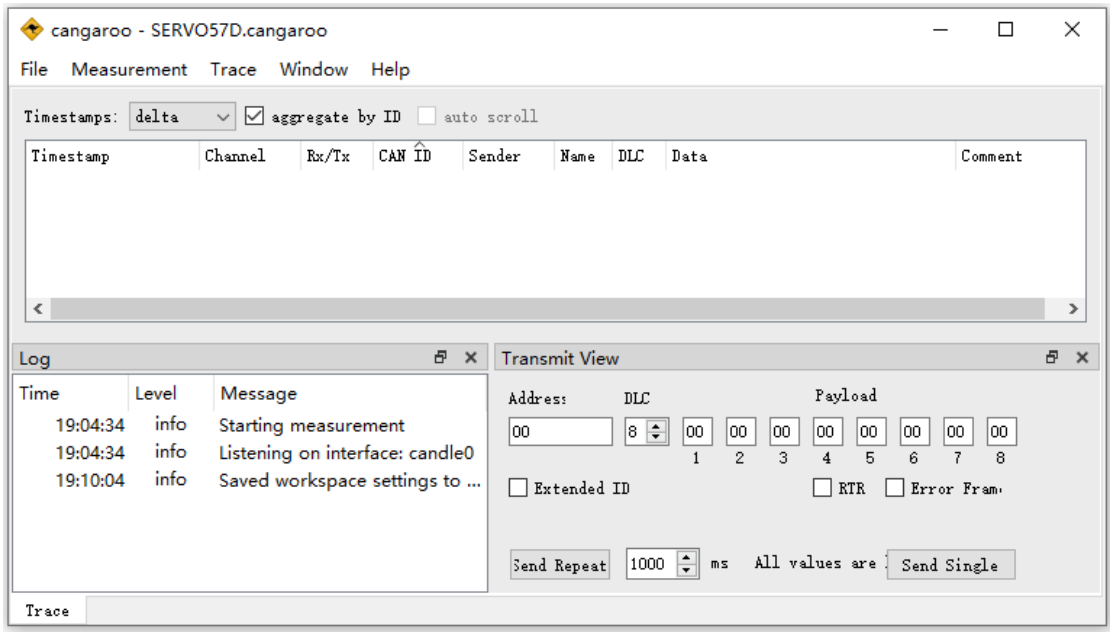
5. Configuration complete, as shown in the image below.



6. Select "File" -> "Save Workspace" from the menu, choose the save path and name, and save the configuration.

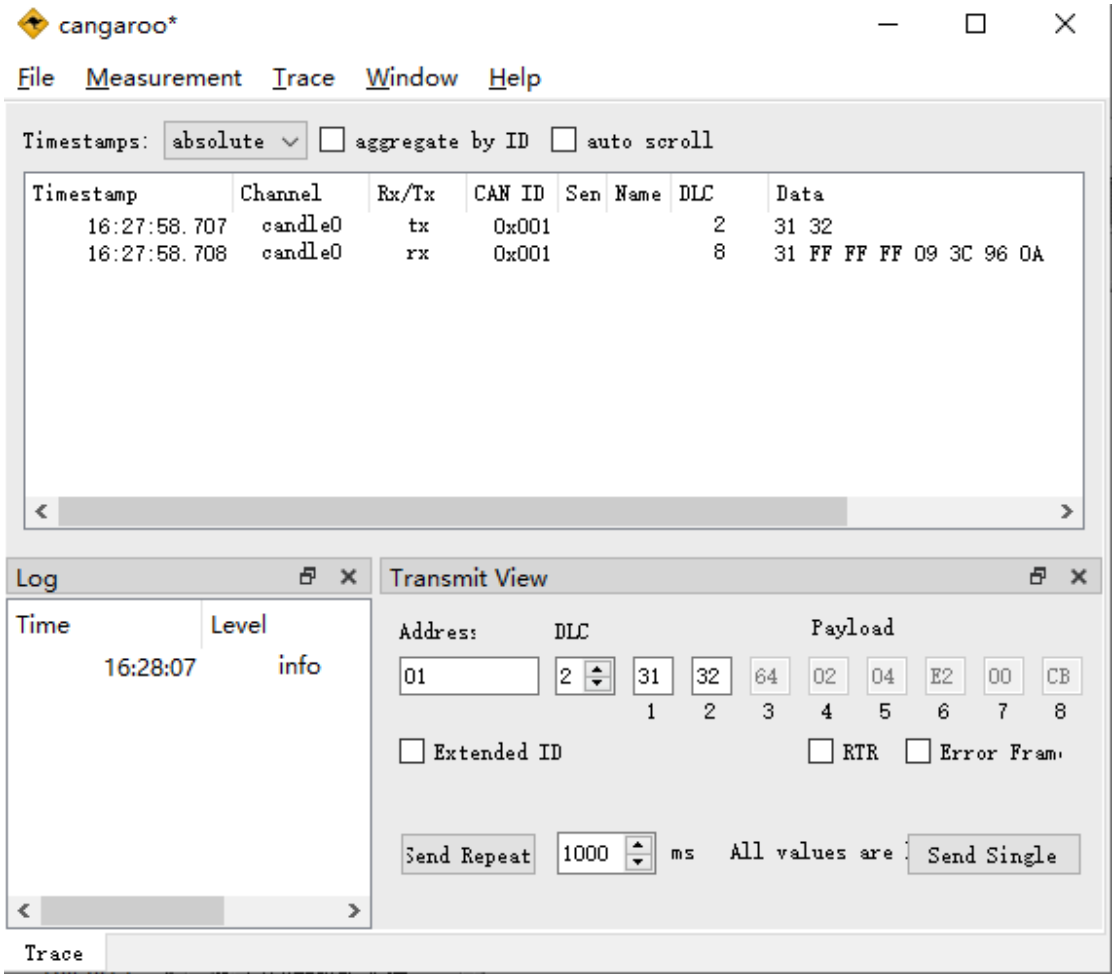


7. After saving, it will look like the image below.



13.3 Example of reading encoder values

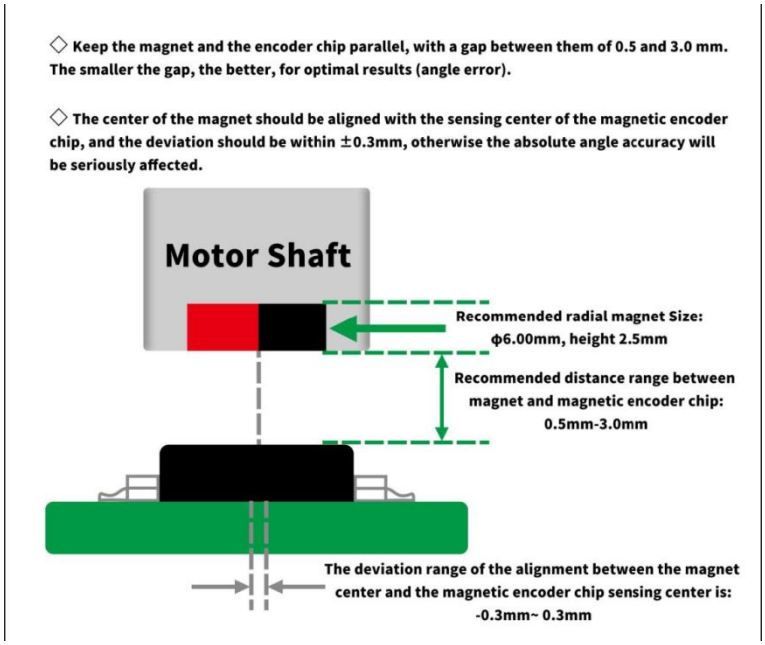
Send 01 31 32 to read encoder value.
Return 01 31 FF FF FF 09 3C 96 0A
As shown in the figure below



Part 14. Frequently Asked Questions and Precautions

14.1 Precautions

1. Power input voltage 12V-24V;
2. Do not plug or unplug the power cord or signal cable while the circuit is powered on, as this may damage the driver board.
3. Press and hold the "Next" button, then power on the device to quickly restore factory default settings.
4. Do not apply a load when calibrating the motor;
5. the driver board is first installed on the motor, or after the motor wiring sequence is changed , the motor needs to be recalibrated.
6. If the system displays "Phase Line Error!" before powering on for calibration:
    a) Check the motor wiring sequence;
    b) Check the power supply voltage and output power (24V/1A, 12V/2A);
    c) If the MKS APT module is connected to the motherboard power supply, try connecting the MKS APT module to ports X, Y, Z, E, etc., and then power on for calibration.
    d) Do not use the MKS APT module for power supply before calibration; connect the power supply directly to V+ and Gnd .
7. If the LED light stays on after powering on, or if the screen displays an error message, please refer to the "Frequently Asked Questions" for troubleshooting.
8. We recommend that users purchase our matching motors directly to avoid the risk of incompatibility. If you choose to use a custom motor in closed-loop mode instead of our motors, the following conditions must be met:
    (1) The motor step distance is 1.8 degrees.
    (2) The motor's internal resistance is less than 10 ohms.
    (3) A radial magnet can be installed on the back of the motor.
    (4) When installing magnets, please note the following diagram:



14.2 Frequently Asked Questions

No	Question	Solution
1	Not Cal	Calibrate the motor.
2	Reverse Lookup Error!	Calibrate Fail, Check magnet and motor shaft.
3	Magnet Loss!	Not install the magent.
4	Magnet Strong!	the magnet too near.
5	Magnet Weak!	the magnet too far.
6	Encoder Error!	Check magnet and motor shaft.
7	Offset Current Error!	Reference voltage error.
8	Phase Line Error!	The motor line sequence is wrong or the power supply is not enough.
9	Wrong Protect!	Locked-rotor protection.
10	Coming Back to Origin..	Going back to zero.
11	Reboot Again	The motor need to be restart.
12	Press Next Key To Fixed	Press Next Key， until it reboot.
13	Low Voltage Error!	The supply voltage is too low.

14.3 Host computer and firmware version compatibility instructions

host computer	Firmware version
1.0.3/1.0.3.1	1.0.3
1.0.3.1	1.0.4
1.0.3.1	1.0.5
Version 1.0.6.1/1.0.6.2	1.0.6
Version 1.0.6.1/1.0.6.2	1.0.7
Version 1.0.6.1/1.0.6.2	1.0.8
1.0.9	1.0.9

Note: The firmware version can be viewed on the screen when the driver board is powered on. If there is no screen, you can send a serial port 40H command to read it.

Part 15. Schematic

Please download 《MKS SERVO42D/57D V1.0 Schematic.pdf》 in
https://github.com/makerbase-motor/MKS-SERVO42D
https://github.com/makerbase-motor/MKS-SERVO57D

Part 16. contact us

https://makerbase.aliexpress.com/
https://www.youtube.com/channel/UC2i5I1tcOXRJ2ZJiRxwpCUQ
https://github.com/makerbase-motor