MKS SERVO 42/57D Closed-Loop Stepper Motor

RS485 User Manual V1.0.9

MKS SERVO42/57D_RS485 Version Notes			
manual	content	Firmware	date
V1.0.0	First release version	V1.0.0	Mar-2023
V1.0.1	1.Added SR_OPEN, SR_CLOSE control mode. 2.It can set any working current. 3.Redefined speed and acceleration for serial Mode. 4. Add the "92" command , It can set the current position to 0 point. 5. Add the "8D" command, It can set the group address.	V1.0.1	Apr-2023
V1.0.2	1.Add long data package. 2.Support for Modbus-RTU communication protocol. 3.Slave does not answer if broadcast address or group address or long data package is used. 4. OUT_1 port output stall indication.	V1.0.2	May-2023
V1.0.3	1. Add the "9A" command, It can set the parameter of 0_Mode. 2. Add the "8F" command, It can locked the key. 3. Add the "34" command, It can read the IO Ports status. 4. the number of slave addresses can be set by menu is change to 16. 5. add left and right endstop limit function.	V1.0.3	Jul-2023
V1.0.4	1. Added menu or command (9BH) to set holding current percentage function. 2. Added absolute motion by pulses(FEH). 3. Modify the 8CH command and add the option of active. 4. Add emergency stop command(F7H). 5. Add limit port remap command (9EH).	V1.0.4	Sep-2023
V1.0.5	1. Support no limit switch for “go home” function. 2. Added menu options: “Hm_Mode” and “Hm_Ma”. 3. Add the "94" command. 4. Fix the bug of command “F4H” and “F5H”. 5. The command”F5H”supports real-time data update. 6. Add the restart motor command(41H).	V1.0.5	May-2024
V1.0.6	1. Add the "46H,47H,48H" command to read/write all parameters. 2. Add the "35H" command to read RAW data of encoder. 3. Add the “9DH”command , En triggers single-turn zero return and position error protection function. 4. It can release the protection status by En. 5. Add the “36H”command to write the IO port.	V1.0.6	Sep-2024

MKS SERVO42D/57D_RS485 Version Notes			
manual	content	version	date
V1.0.7	1. 3BH instruction, add the function of reading the return to zero state value.	V1.0.7	Mar-2025
	2. 83H instruction, add the function of changing the current value during operation without saving.		
	3. F6H instruction, add the function of setting the running time in speed mode.		
V1.0.8	1. Added IAP firmware upgrade function, see 5.6.	V1.0.8	Jul-2025
	2. After restoring the default parameters, there is no need to recalibrate.		
	3. Added the function to read configuration parameters, see 5.1.10.		
	4. Added multi-motor synchronous control operation function, see Part 12.		
	5. Added a location arrival threshold setting function, see 5.2.17.		
	6. 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 5.2.16.		
	4. Added coordinate zeroing function, see 7.2, 7.7.		

Catalog

Part 1. Product Overview

1.1 Product Introduction

The MKS SERVO42D/57 D_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 function, origin switch zeroing function, and mechanical limit switch zeroing function;
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. onboard RS-485 interface, 256 slave addresses, and supports broadcast and packet addresses.
16. Supports MODBUS-RTU communication protocol;
17. Maximum operating current 5.2A, MOSFET continuous operating current 46A;
18. Onboard OLED display and buttons allow for easy parameter modification with automatic saving and immediate effect.
19. It has built-in stall protection function;
20. Features encoder self-calibration function;

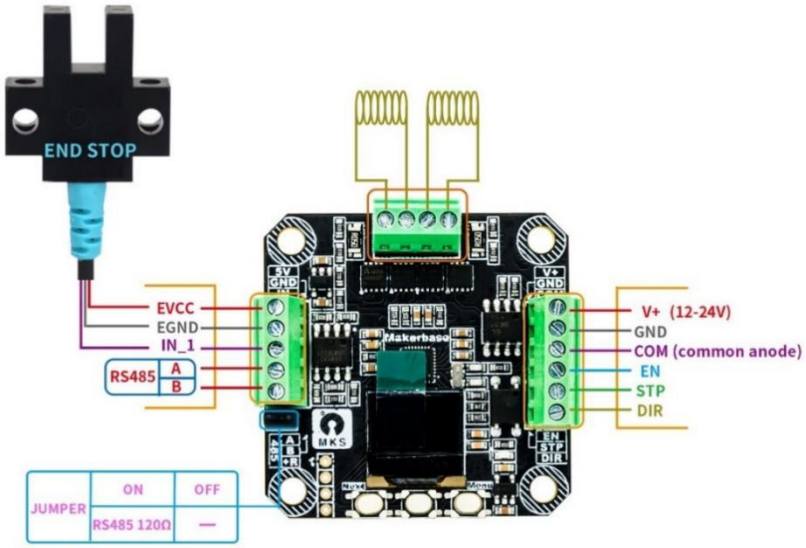
21. One-click quick factory reset;
22. High-speed performance is stable, operation is smooth and vibration-free, and emergency stop is possible;
23. The integrated aluminum alloy casing provides effective heat dissipation and ensures more stable continuous high-current operation of the motor.
24. Provides an open-source host computer and STM32/Arduino usage examples;

1.3 Product Parameters

Product Parameters		
Motherboard Model	MKS SERVO 42 D V1.0	MKS SERVO57D V1.2
MCU main controller	N32L403 (Cortex-M4)	N32L406 (Cortex-M4)
MOSFET	AP4008QD (40V , 20A)	AP30H80Q (30V, 70A)
Magnetic encoder	MT6816 (14-bit)	
RS485 transceiver	SP485EEN	
Operating voltage	12V-24V	
Operating current	0-3000mA	0-5200mA
Closed-loop feedback frequency	Torque ring 20KHz	
	Speed loop 10kHz	
	Position ring 10KHz	
Maximum speed	3000RPM+	
Subdivision support	1-256 can be further subdivided	
Silent/Vibration	Ultra-quiet, ultra-low vibration	
Motor temperature	FOC control mode, the motor does not heat up.	
Pulse signal input	3.3V-24V (Common Anode)	
Pulse signal frequency	Up to 160kHz	
RS485 interface speed	9 600/19200/…/115200/25600	
RS485 interface address	1 broadcast address, 255 slave addresses	

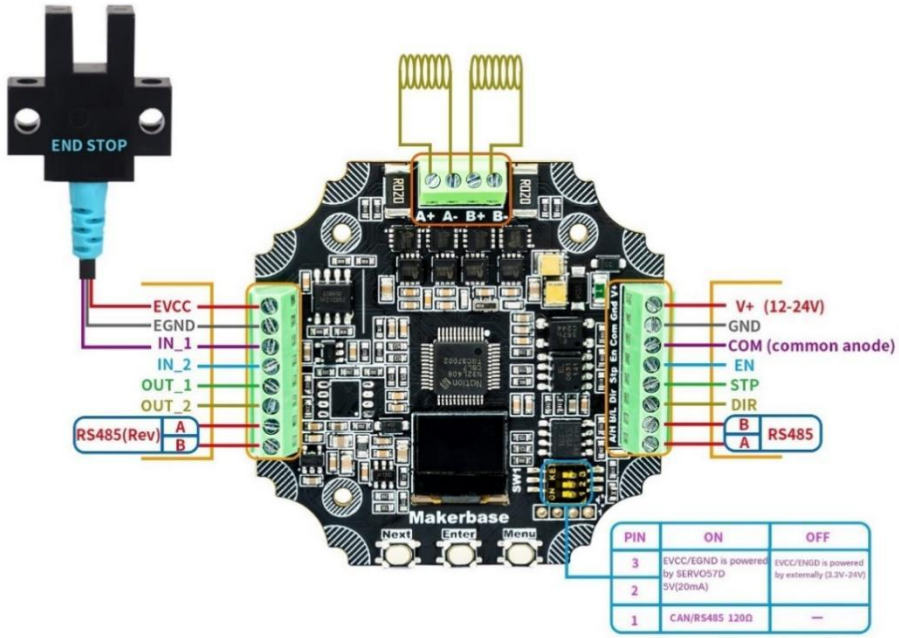
1.4 Interface Description

① S ERVO42D RS485 Interface Description

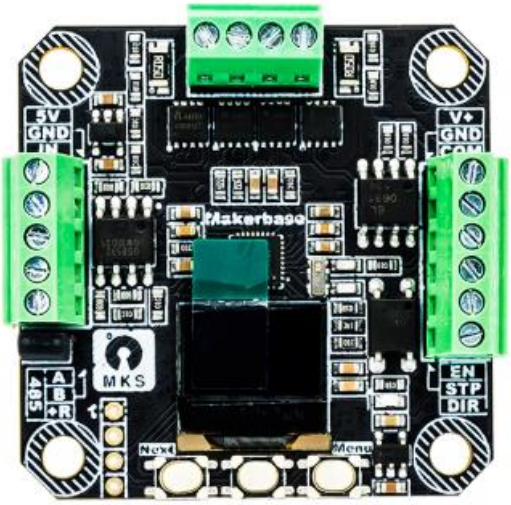


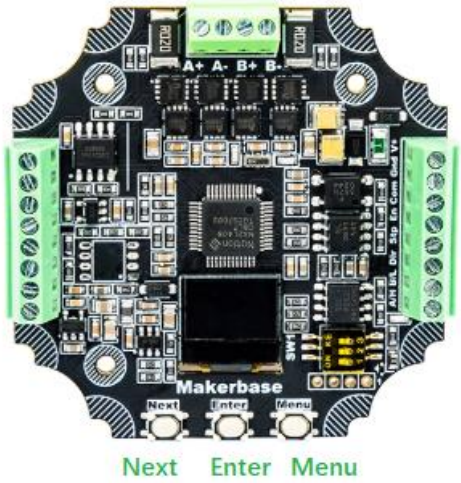
Note: EVCC/ENGD is powered by a 5.0V/20mA power supply from the SERVO42D driver board .

② SERVO57D_RS485 Interface Description



1.5 Key Operation





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 " 90H "** ;

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 "91H"** ;

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. **Return to zero with endstop**: 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. **Return to zero with mechanical limit** 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**: 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 addresses for multi-motor control** is suitable for controlling all motors on a bus to execute the same command.

2. **Using grouped addresses 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 t 5.2.15.

3. **Multi-motor control using multi-instruction data frames** is suitable for controlling motors on a bus to perform different actions, with a maximum of 5 motors controlled. See 12.3.

4. **The synchronous control function is used for multi-motor control** , suitable for synchronously running different motors on a bus to execute different operation commands, with no limit on the number of motors controlled. See 12.4.

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 "FA 01 80 00 CRC"
   CRC = (0xFA + 0x01 + 0x80 + 0x00) & 0xFF = 0x1 7B & 0xFF = 0x 7B
   In other words, after adding each byte, only the last two digits are taken.
3. Simply check the "Automatically send check bit" option, see 13.1.

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 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) .

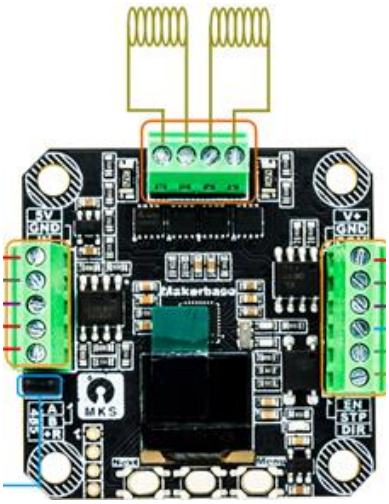
Part 2. Wiring method

2.1 Motor wiring method

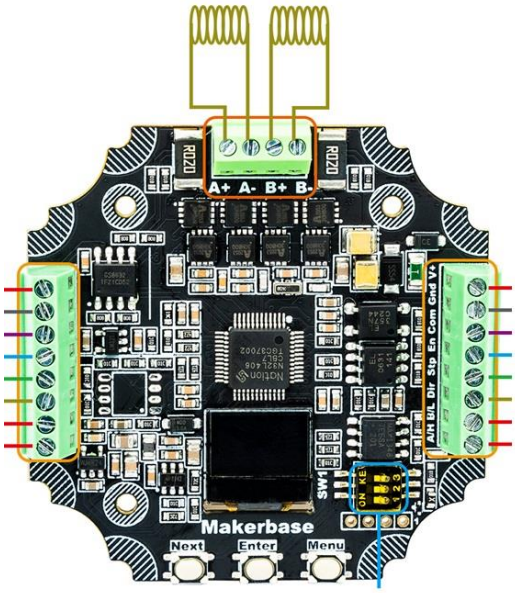
Note: The internal resistance of the motor should be less than 10 ohms .

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

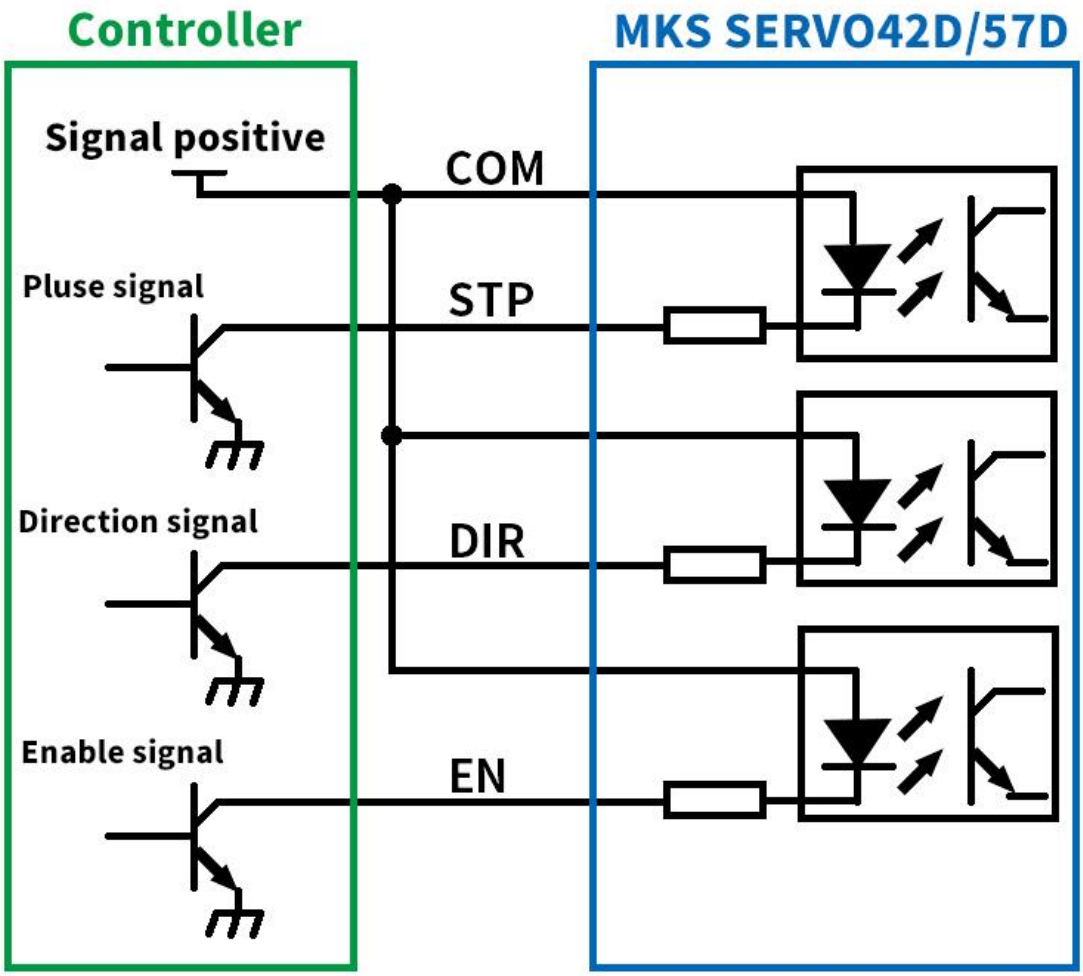
① **SERVO42D RS485 Wiring Method**



② **SERVO57D_RS485 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.

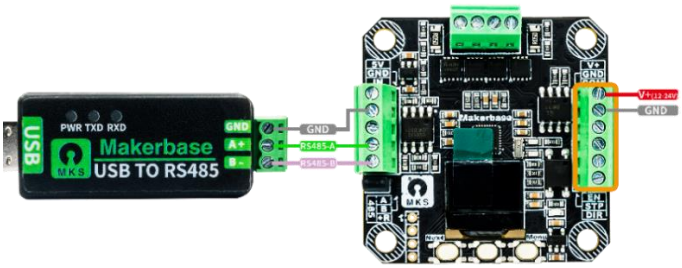
...

2.3 RS485 Wiring Method

Note: To reduce bus interference, the host computer and the motor should share a common ground, and RS485 A and B signals should be transmitted using shielded twisted-pair cable.

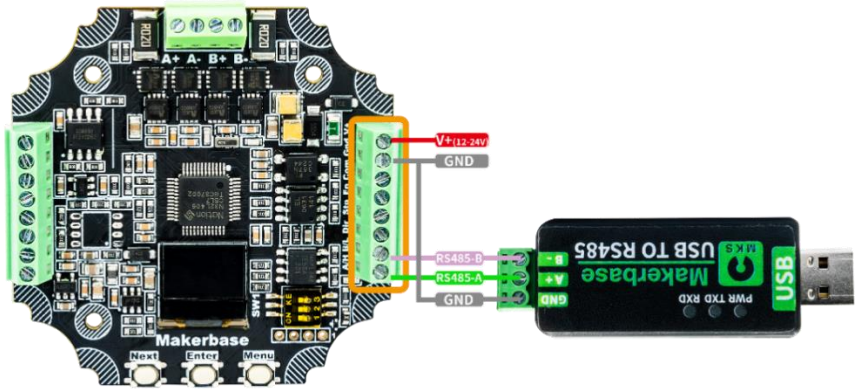
1. RS485 standalone communication wiring

① MKS SERVO42D RS485 standalone wiring



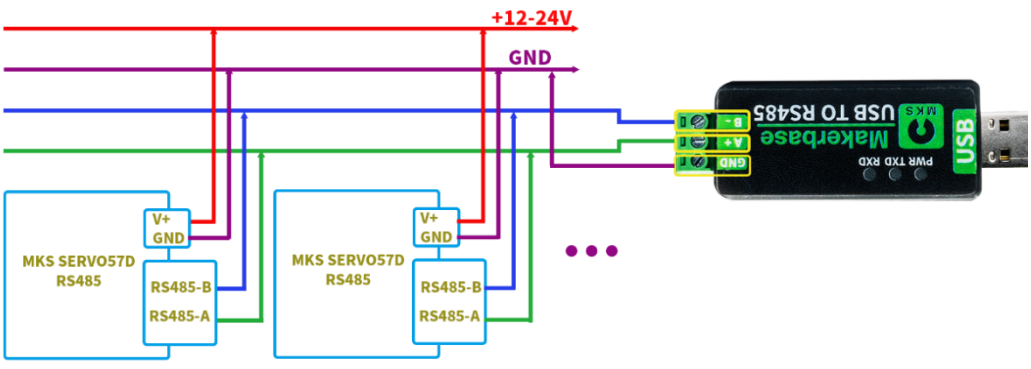
Note: A 120Ω terminating resistor is not required for stand-alone communication (i.e., do not connect a shorting cap).

② MKS SERVO57D RS485 standalone wiring



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

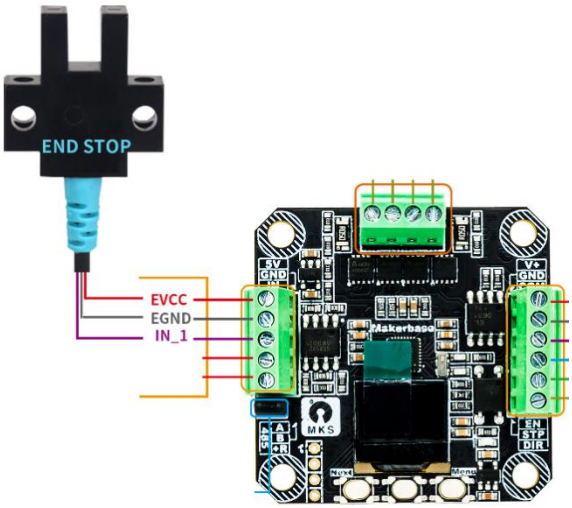
2. RS485 multi-machine communication wiring



2.4 External switch wiring method

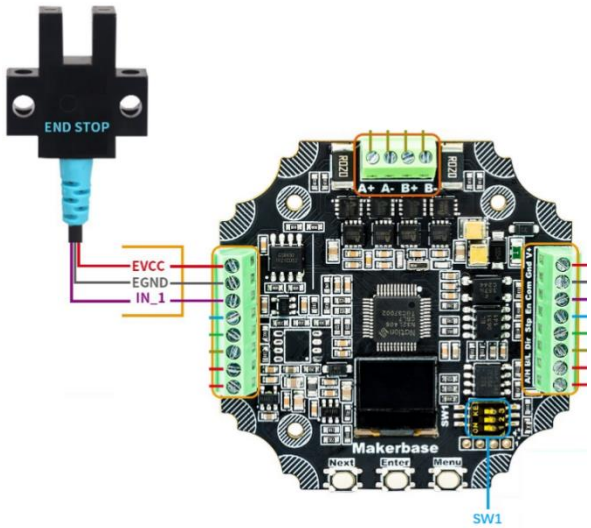
Note: NPN type limit switches are used by default.

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	EVCC/EGND are powered by SERVO57D at 5V (20mA).	EVCC/E G ND is powered by an external source (3.3V-24V).
2		
1	RS485 120Ω 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 **UartBaud** : Set the baud rate of serial.
9600, 19200, 25000, 38400 (default), 57600 , 115200 , 256000 .

3.12 **UartAddr** : Set the the slave address of seria.
01
…
15
16
(Default: 01)
Note: The addresses greater than 16 need to be set by serial command. After it is set, it will be added to this option.

3.13 **UART RSP** :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 **Mb_RTU** : Choose whether to use MODBUS-RTU communication protocol.
Disable: Disable the MODBUS-RTU protocol (default).
Enable: Enable the MODBUS-RTU protocol.

3.15 Single-cycle return-to-zero parameters

3.15.1 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.

3.15.2 Set O : Set the zero point for go back when power on.
(O_Mode must not be Disable)

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

0: Slowest gear

...

4: The fastest gear

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

CW: Clockwise (default)

CCW: Counterclockwise

3.16 return to zero with endstop parameter

3.16.1Hm_Trig : Set the effective level of the end stop.

Low : Low level (default)

High : High level

3.16.2Hm_Dir : Set the direction of go home.

CW: Clockwise (default)

CCW: Counterclockwise

3.16.3Hm_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.

3.16.4Hm_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.

3.16.5 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.

3.16.6 **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.

3.17 **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")

3.18 **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.

3.19 **About** : Show version parameters.

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

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

Part 4. Serial data format description

downlink frames (host → SERV042/57D board)							
Frame header	Slave address	function code	Instruction data				CRC check code
FA	addr	code					CRC
Uplink frame (host← SERV042D/57D board)							
Frame header	Slave address	function code	Returned data				CRC check code
FB	addr	code					CRC

1. Instruction data and return data are stored in big endian format
2. Downlink package Head is “FA”, uplink package Head is “FB”.
3. The slave address(addr) range is 00~255.（default is 01）.
   00 is the broadcast address;
   01~16 can be set in the UartAddr option of the display menu;
   greater than 16 need to be set by serial commands.
4. The function code (code) executes the corresponding command.
   for example, 0x80 executes the calibration command.
5. Command data or return data, see "Serial Command Description" for details
6. The Check code is CHECKSUM 8bit
   For example: command “FA 01 80 00 CRC”
   CRC = (0xFA + 0x01 + 0x80 + 0x00) & 0xFF = 0x17B & 0xFF = 0x7B
7. When the host computer sends a command, the timing between the Bytes of a single command (FA ... CRC) must be continuous, and there must not be more than one Byte delay, otherwise multiple idle line will be detected and the motor may fail to receive the command..

Note: Slave does not answer if broadcast address is used.

Part 5. General Instructions

Note 1: When sending commands using broadcast addresses or packet addresses, 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

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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	30H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)					
Byte 1	Byte 2	Byte 3	Bytes 4-7	Bytes 8-9	Byte 10
header	address	code	carry	value	Checksum
FB	addr	30H	turns(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.

Note: The encoder value is updated regardless of whether the motor is enabled or not.

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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	31H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-9	Byte 10
header	address	code	value	Checksum
FB	addr	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.

5.1.3 Read the real-time speed of the motor

The read command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	32H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6
header	address	code	motor speed	Checksum
FB	addr	32H	rpm(int16_t)	CRC

rpm: Real-time motor speed, unit RPM, data type int16_t

Note: The counterclockwise rotation speed is greater than 0, and the clockwise rotation speed is less than 0.

5.1.4 Read the number of received pulses

The read command is as follows:

Downlink frame (PC → SERV0xxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	33H	CRC

Returned data:

Uplink frame (PC ← SERV0xxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-7	Byte 8
header	address	code	pulse count	Checksum
FB	addr	33H	pulses	CRC

pulses: The number of pulses received, data type int32_t

5.1.5 Read the raw accumulated 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 frame (PC → SERV0xxD)			
Byte 1	Byte 2	Byte 3	Byte 4
Frame header	Slave address	function code	Checksum
FA	addr	35H	CRC

Returned data:

Uplink frame (PC ← SERV0xxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-9	Byte 10
header	address	code	Encoder value	Checksum
FB	addr	35H	value	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.

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 frame (PC → SERV0xxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	39H	CRC

Returned data:

Uplink frame (PC ← SERV0xxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-7	Byte 8
header	address	code	Error value	Checksum
FB	addr	39H	errors	CRC

errors: Real-time position error value, data type int32_t

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 frame (PC → SERV0xxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	3AH	CRC

Returned data:

Uplink frame (PC ← SERV0xxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable state	Checksum
FB	addr	3AH	status	CRC

status=01 Motor is enabled

status=00 Motor not enabled

5.1.8 Read the motor return to zero status

The read command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	3BH	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
header	address	code	Single-cycle to zeroing	returns to zero	Checksum
FB	addr	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 frame (PC → SERVOxxD)					
Byte 1	Byte 2	Byte 3	Byte 4	Bytes 5-6	Byte 7
header	address	command	code	Timer	Checksum
FA	addr	01H	code	times	CRC

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

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

Returned data:

Uplink frame (PC ← SERVOxxD)					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
header	address	command	code	state	Checksum
FB	addr	01H	code	status	CRC

status=00 Set fail.

status=01 Set success.

The parameter data format returned periodically is as follows:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4-n	Byte n+1
header	address	code	Parameter value	Checksum
FB	addr	code	param	CRC

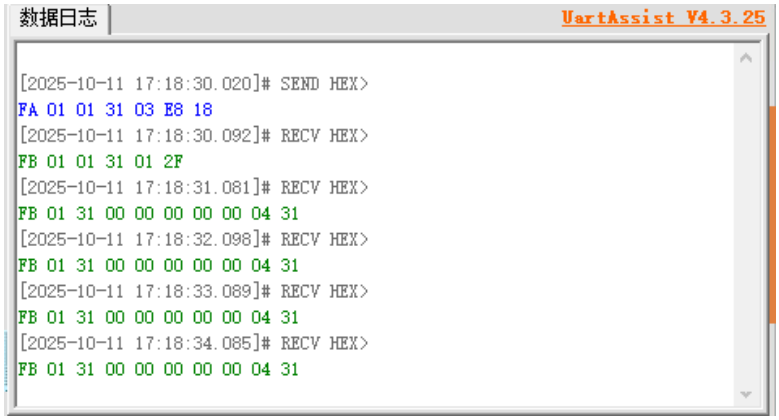
param: The corresponding read-only parameter

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

Send FA 01 01 31 03 E8 18

Return to FB 01 01 31 01 2F

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 configuration parameters is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	command	code	Checksum
FA	addr	00H	code	CRC

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

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

The returned parameter data format is as follows:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4-n	Byte n+1
header	address	code	Parameter value	Checksum
FB	addr	code	param	CRC

param: The corresponding configuration parameter

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

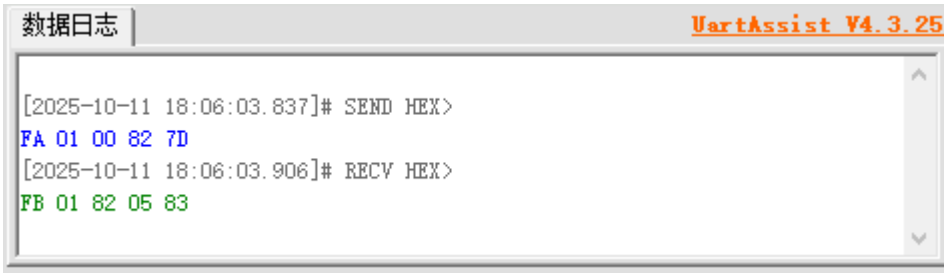
If this parameter does not support reading, the returned data will be as follows:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6
header	address	code	Parameter value	Checksum
FB	addr	code	0xFFFF	CRC

Taking reading the "working mode" as an example:

Send FA 01 00 82 7D

Return to FB 01 82 05 83, as shown in the image below.



5.1.11 Read version information

Read hardware and firmware information.

The read command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
Frame header	Slave address	function code	Checksum
FA	addr	40H	CRC

The returned data format is as follows:

Uplink frame (PC ← SERVOxxD)						
Byte 1	Byte 2	Byte 3	Byte 4		Bytes 5-7	Byte 8
header	address	code	B7-b4 Calibration status	b3-b0 Hardware version	Firmware version	check
FB	addr	40	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 for easy identification.

1. Write user ID number

The command to write the ID is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-7	Byte 8
header	address	code	ID	Checksum
FA	addr	42H	ID	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable state	Checksum
FB	addr	42H	status	CRC

status=00 Write failed

status=01 Write successful

2. Read user ID

The command to read the ID is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	42H	CRC

Return ID data:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-7	Byte 8
header	address	code	ID	Checksum
FB	addr	42H	ID	CRC

5.2 Write-only parameter command

5.2.1 Calibrate encoder

（Same as the "Cal" option on screen）

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Data	Checksum
FA	addr	80H	00H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	state	Checksum
FB	addr	80H	status (uint8_t)	CRC

status = 0 Calibration in progress …

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.

Note2: After the calibration is completed, the driver board will automatically reset and restart.

5.2.2 Set the work mode

（Same as the "Mode" option on screen）

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Work mode	Checksum
FA	addr	82H	mode	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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	82H	status (uint8_t)	CRC

status = 0 Set fail.

status = 1 Set success.

5.2.3 Set the working current

（Same as the "Ma" option on screen）

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6
header	address	code	current (mA)	Checksum
FA	addr	83H	current	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 working current of the motor without saving it during operation, the command is as follows:

Downlink frame (PC → SERVOxxD)					
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6	Byte 7
header	address	code	current (mA)	data	Checksum
FA	addr	83H	current	00H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	83H	status (uint8_t)	CRC

status = 0 Set failed.

status = 1 Set success with saved.

status = 2 Set success without saved.

5.2.4 Set holding current percentage

（Same as the "HoldMa" option on screen）

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	percentage	Checksum
FA	addr	9BH	ratio	CRC

ratio = 00 10%

ratio = 01 20%

ratio = 02 30%

…

ratio = 08 90%

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	9BH	status (uint8_t)	CRC

status = 0 Set fail.

status = 1 Set success.

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

5.2.5 Set subdivisions

（Same as the "MStep" option on screen）

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Value	Checksum
FA	addr	84H	MS	CRC

Note1:the new micstep will show in the screen of MStep option.

Note2:powers of 2 to subdivide the number of steps to reduce errors, such as 2, 4, 16, 32, 64, etc.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	effective level	Checksum
FA	addr	85H	Level	CRC

Level = 00 active low (L)

Level = 01 active high (H)

Level = 02 active always (Hold)

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	85H	status (uint8_t)	CRC

status = 0 Set fail.

status = 1 Set success.

Note 1: 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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	direction	Checksum
FA	addr	86H	dir	CRC

dir = 00 CW

dir = 01 CCW

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	86H	status (uint8_t)	CRC

status = 1 Set success.

status = 0 Set fail.

Note: Only valid for pulse interface, the direction of serial interface is determined by the instruction.

5.2.8 Set automatic screen-off function

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

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Frame header	Slave address	function code	Enable	Checksum
FA	addr	87H	enable	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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable	Checksum
FA	addr	89H	enable	CRC

enable = 01 enabled interpolation function.

enable = 00 disabled interpolation function.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	89H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.10 Set the serial port baud rate

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

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	baud rate	Checksum
FA	addr	8AH	baudrate	CRC

baud = 01 9600.

baud = 02 19200.

baud = 03 25000.

baud = 04 38400.

baud = 05 57600.

baud = 06 115200.

baud = 07 256000.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	8AH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.11 Set slave address

（Same as the "UautAddr" option on screen）

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Frame header	Slave address	function code	Slave address	Checksum
FA	addr	8BH	address	CRC

Note1:the new address will show in the screen of UartAddr option.

Note2: 0 is the broadcast address

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	8BH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.12 Configure slave respond and active

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)					
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
header	address	code	data1	data2	Checksum
FA	addr	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 FA 01 FD 02 80 02 00 00 FA 00 76

a. **In no-response mode (XX=0, YY=0 or 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 0 1 or fails 0 0
c. **In the default mode (XX=1, YY=1)**
   Slave immediately returns to position control start 0 1 or fails 0 0
   After the motor finishes running or stops when it hits the limit switch, return to 0 2 or 0 3.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 function code "F 1 H".

5.2.13 Configure MODBUS -RTU communication protocol

（Same as the "MB_RTU" option on screen）

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	RTU Protocol	Checksum
FA	addr	8EH	enable	CRC

enable = 01 enabled Modbus-RTU

enable = 00 disabled Modbus-RTU (default)

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	8EH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

Note: In MODBUS-RTU communication protocol mode, if you need to switch back to using the MKS communication protocol, you can send the following command to disable the MODBUS-RTU communication protocol.

01 06 00 8E 00 00 E9 E1

5.2.14 Set key lock function

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	control word	Checksum
FA	addr	8FH	enable	CRC

enable = 01 lock the button

enable = 00 unlock the button

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	8FH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.15 Set group address

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	address	Checksum
FA	addr	8DH	groupAddr	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	8DH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

For example, there are 6 motors with the settings address:

	Broadcast address	Slave address	Group address
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 FA 01 FD 01 2C 64 00 00 0C 80 15, motor 1 will rotate a turn
send FA 00 FD 01 2C 64 00 00 0C 80 14, motor1-6 will rotate a turn
send FA 50 FD 01 2C 64 00 00 0C 80 64, motor1-3 will rotate a turn
send FA 51 FD 01 2C 64 00 00 0C 80 65, motor4-6 will rotate a turn

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

5.2.16 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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-7	Byte 8
header	address	code	Protection time	Checksum
FA	addr	98H	times	CRC

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

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

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	98H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.2.17 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 frame (PC → SERVOxxD)					
Byte 1	Byte 2	Byte 3	Byte 4	Bytes 5-6	Byte 7
header	address	code	Enable	threshold	Checksum
FA	addr	95H	enable	values(uint16_t)	CRC

enable: 00 disables the threshold determination function.
01 enables the threshold determination function (default)

values: Location threshold (default value 200)

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

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	95H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

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 PID parameters for vFOC mode

The command format is as follows:

Downlink frame (PC → SERVOxxD)							
Byte 1	Byte 2	Byte 3	Bytes 4-5	Bytes 6-7	Bytes 8-9	Bytes 10-11	Byte 12
header	address	code	KP	KI	KD	KV	Checksum
FA	addr	96H	Kp	Ki	Kd	Kv	CRC

Kp: Range 0-1024 (Default: 0xDC)
Ki: Range 0-1024 (Default: 0x64)
Kd: Range 0-1024 (Default: 0x10E)
Kv: Range 0-1024 (Default: 0x140)

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	96H	status (uint8_t)	CRC

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

5.3.2 Set CLOSE mode PID parameters

The command format is as follows:

Downlink frame (PC → SERVOxxD)							
Byte 1	Byte 2	Byte 3	Bytes 4-5	Bytes 6-7	Bytes 8-9	Bytes 10-11	Byte 12
header	address	code	KP	KI	KD	KV	Checksum
FA	addr	97H	Kp	Ki	Kd	Kv	CRC

Kp: Range 0-1024 (Default: 0xC8)
Ki: Range 0-1024 (Default: 0x50)
Kd: Range 0-1024 (Default: 0xFA)
Kv: Range 0-1024 (Default: 0x12C)

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	97H	status (uint8_t)	CRC

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

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

（Same as the "**Protect**" option on screen）:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Stall protection	Checksum
FA	addr	88H	enable	CRC

enable = 00 disables Stall protection (default).

enable = 01 enables Stall protection

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	88H	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

5.4.2 Set position out-of-tolerance protection parameters

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Byte 4	Bytes 5-6	Bytes 7-8	Byte 9
header	address	code	Protection	time	Error count	Checksum
FA	addr	9DH	Enble	Tim	Errors	CRC

Enble 0 : Disable position deviation protection (default)
1 : Enable position deviation protection

Tim: uint16_t sets the error statistics time length.
Note: 1 (Tim) unit is approximately equal to 15 ms.

Errors : uint16_t sets the protection error count

Note: When Errrors = 28000 , the motor is misaligned by 360 degrees.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	9DH	status (uint8_t)	CRC

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

For example:
send FA 01 9D 0 1 00 14 36 B0 9 3
return FB 01 9D 01 9A
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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	3EH	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable state	Checksum
FB	addr	3EH	status	CRC

status=00 Motor 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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	3DH	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable state	Checksum
FB	addr	3DH	status	CRC

status=00 Release failed

status=01 Release successful

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

（Same as the "Restore" option on screen）

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	3FH	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Restored status	Checksum
FB	addr	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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	41H	CRC

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

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	41H	status (uint8_t)	CRC

status = 0 Reset failed

status = 1 Reset successful

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

5.6 IAP Firmware Online Upgrade Instructions

IAP firmware upgrades do not require disassembling or recalibrating the motor, the user configuration parameters will be retained after firmware upgrade.

There are two IAP upgrade modes:

IAP Mode 1:

Press and hold the "Menu" button, then power on the device to enter IAP mode 1.

IAP Mode 2:

Send a control command to enter IAP mode 2.

Instructions for IAP upgrades can be found in "MKS SERVO42&57D IAP Upgrade Instructions.pdf".

The IAP upgrade operation video can be found in "MKS SERVO42&57D IAP Upgrade Operation Video.mp4".

IAP Mode 2 are as follows:

Downlink frame (PC → SERV0xxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	data	Checksum
FA	addr	50H	cmd	CRC

cmd = 01 Enter boot mode

cmd = 02 Enter silent state

cmd = 03 Exit silent state

Control word cmd description:

When there is only one motor on the bus, send the command cmd = 01 directly to enter boot mode;

When there are multiple motors on the bus, the following steps can be taken to avoid data interference:

a. First, broadcast the command cmd = 02 (FA 00 50 02 CRC) to put all motors into a silent state;
b. Then send the command cmd = 01 to the motor to be upgraded to enter boot mode and upgrade the firmware;
c. After the upgrade is complete, broadcast the command cmd = 03 (FA 00 50 03 CRC) to make all motors exit the silent state.

Note: In silent mode, the motor does not respond to commands other than 50.

Note: After receiving the cmd = 01 command, the motor will automatically restart and enter IAP mode 2, waiting to receive the upgrade file.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	50H	status (uint8_t)	CRC

status = 0 Reset failed

status = 1 Reset successful

5.7 Read/Write all parameters commands

5.7.1 Command to write all configuration parameters

The write command is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-37	Byte 38
header	address	code	Parameter	Checksum
FA	addr	46H	params	CRC

the parmas parameter are defined in the table below.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	46H	status (uint8_t)	CRC

status = 0 write failed

status = 1 write successful

The default write parameters are as follows:

FA 01 46 02 0C 80 04 10 00 00 00 00 01 04 01 00 01 01 00 00 00 00
00 3C 00 00 00 20 00 00 03 20 00 00 00 02 00 6C

(The last byte “6C” is the checksum)

5.7.2 Command to read all configuration parameters

The read command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	47H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-37	Byte 38
header	address	code	Parameter value	Checksum
FB	addr	47H	params	CRC

the parmas parameter are defined in the table below.

Command to set all motor parameters (46)			
byte	format	instruction	Default (hexl)
1	FA		
2	01		
3	46		
4	Work mode	82	2
5	Operating current	83	C / 6 / 3 / 2
6			80 / 40 / 20 / 58
7	Maintaining current	9B	4
8	Job breakdown	84	10
9	En effective level	85	0
10	Motor direction	86	0
11	Automatic screen off	87	0
12	Stall protection	88	0
13	Subdivision interpolation	89	1
14	baud rate	8A	4
15	Slave address	8B	1
16	Group address	8D	0
17	Response method	8C	1
18			1
19	MODBUS	8E	0
20	Button lock	8F	0
21	Limit trigger level	90	0
22	Limit direction		0
23	Limit speed		0
24			3C
25	Limit enable		0
26	Return distance	94	0
27			0
28			20
29			0
30	Zero mode		0
31	Zero-return current		3 / 1 / 0 / 0
32			20 / 90 / C8 / C8
33	Limit remapping	9E	0
34	Single zero mode	9A	0
35	setting 0 point		0
36	return to zero speed		2
37	direction		1
38	Check value		

Return all motor parameters (47)	
byte	format
1	FB
2	01
3	47
4	Work mode
5	Operating current
6	
7	Maintaining current
8	Job breakdown
9	En effective level
10	Motor direction
11	Automatic screen off
12	Stall protection
13	Subdivision interpolation
14	baud rate
15	Slave address
16	Group address
17	Response method
18	
19	MODBUS
20	Button lock
21	Limit trigger level
22	Limit direction
23	Limit speed
24	
25	Limit enable
26	Return distance
27	
28	
29	
30	Zero mode
31	Zero-return current
32	
33	Limit remapping
34	Single zero mode
35	Reserved (FF)
36	return to zero speed
37	direction
38	Check value

Note: xx/xx/xx/xx data correspond to 57D/42D/35D/28D parameters.

5.7.3 Command to read all status parameters

The read command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	48H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Bytes 4-31	Byte 32
header	address	code	Parameter value	Checksum
FB	addr	48H	params	CRC

the parmas parameter are defined in the table below.

byte	Return data format	Corresponding instruction
1	FB	
2	1	
3	48	
4	Motor operating status	F1
5	Encoder value	31
6		
7		
8		
9		
10		
11	Real-time rotation speed	32
12		
13	pulse count	33
14		
15		
16		
17	IO status	34
18	raw encoder value	35
19		
20		
21		
22		
23		
24	Angular error	39
25		
26		
27		
28	Enable state	3A
29	Single lap return to zero	3B
30	stalled state	3E
31	Return to zero state	3B
32	Check value	

Part 6. IO Port Operation Instructions

6.1 Read I/O port status

The read command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	Slave address	code	Checksum
FA	addr	34H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Port status	Checksum
FB	addr	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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Port mode	Checksum
FA	addr	9FH	type	CRC

type = 0 IN_1 As the input port (default)

type = 1 IN_1 as the output port

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	9FH	status (uint8_t)	CRC

status = 0 Set failed

status = 1 Set successful

6.3 Write data to IO port

The command to write a port is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Port data	Checksum
FA	addr	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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	36H	status (uint8_t)	CRC

status =1 write success.

status =0 write fail.

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:

Byte 1	Byte 2	Byte 3	Byte 4	Bytes 5-8	Byte 9
header	address	code	Start level	period	Checksum
FA	addr	99F	divLevel	divPeriod	CRC

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

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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 “origin homing”

There are two ways to "return to zero":

" return to zero by endstop" and " return to zero by mechanical limit ".

1. Return to to zero by endstop.

Connect the endstop to the corresponding port:

endstop access port	
Default port	Remapped port
IN1	EN

For details, please refer to: 1.7 IO Port DescriptionDescription

The process of return to zero is as follows:

a. The motor searches for the endstop based on the set "direction" and "speed " ;
b. Stop immediately upon encountering the rising edge of the endstop 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. Returns to zero by mechanical limit.

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 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 homing”

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

The 28/35/42D motor has only a left limit port. In serial control mode, limit port remapping can be enabled to add a right limit port.

For the 57D motor, limit port remapping can also be enabled if required to facilitate wiring.

Left limit -> En port

Right limit -> Dir port

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

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable	Checksum
FA	addr	9EH	enable	CRC

enable = 00 Disable port mapping (default)

enable = 01 Enable port mapping

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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.

（Same as the “HmTrig、HmDir、HmSpeed、EndLimit” option on screen）

Downlink package（PC → SERVO42D/57D）							
Byte1	Byte2	Byte3	Byte 4	Byte 5	Byte 6-7	Byte 8	Byte 9
Head	Slave addr	Function	level	dir	speed	enable	Check
FA	01	90H	HmTrig	HmDir	HmSpeed	EndLimit	CRC

HmTrig the effective level of the end stop

0: Low 1: High

HmDir the direction of go home

0: CW 1: CCW

HmSpeed the speed of go home

0~3000 (RPM)

EndLimit

0: disable endstop-limit

1: enable endstop-limit

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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Bytes 4-7	Byte 8	Bytes 9-10	11 bytes
header	address	code	offset	Zero mode	Zero-return current	Checksum
FA	addr	94H	retValue	hm_mode	hm_ma	CRC

hm_mode 0: Return to zero by endstop
1: Return to zero by mechanical limit
2: Single lap to zero
hm_ma Mechanical limit return-to-zero current
retValue Origin offset
For example:
retValue = 0x4000 (Returns one full rotation, 360 degrees)
retValue = 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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	94H	status (uint8_t)	CRC

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

7.3.4 Set single-cycle zero-return parameters

（Same as the “0_Mode、Set 0、0_Speed、0_Dir” option on screen）

Downlink frame (PC → SERVOxxD)							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
header	address	code	method	Set 0	speed	direction	Checksum
FA	addr	9AH	mode	enable	speed	dir	CRC

mode:

0: Disable do not go back to zero

1: DirMode go back to zero with direction

2: NearMode go back to zero with minimum angle

enable:

0: clean zero

1: set zero

2: not modify the zero

speed:

0 ~ 4 (0:slowest 4:fastest)

dir:

0: CW

1: CCW

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	9AH	status (uint8_t)	CRC

status =1 Set success.

status =0 Set fail.

7.3.5 Set zero-point command

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

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	92H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	Slave address	code	method	Checksum
FA	addr	91H	goZeroMode	CRC

goZeroMode 00: Execute the " origin homing " function.

01: Execute the " coordinate homing " function

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

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

The following two instructions are equivalent

FA 01 91 8C

FA 01 91 00 8C

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	91H	status (uint8_t)	CRC

status =0 go home fail.

status =1 go home start.

status =2 go home sucess.

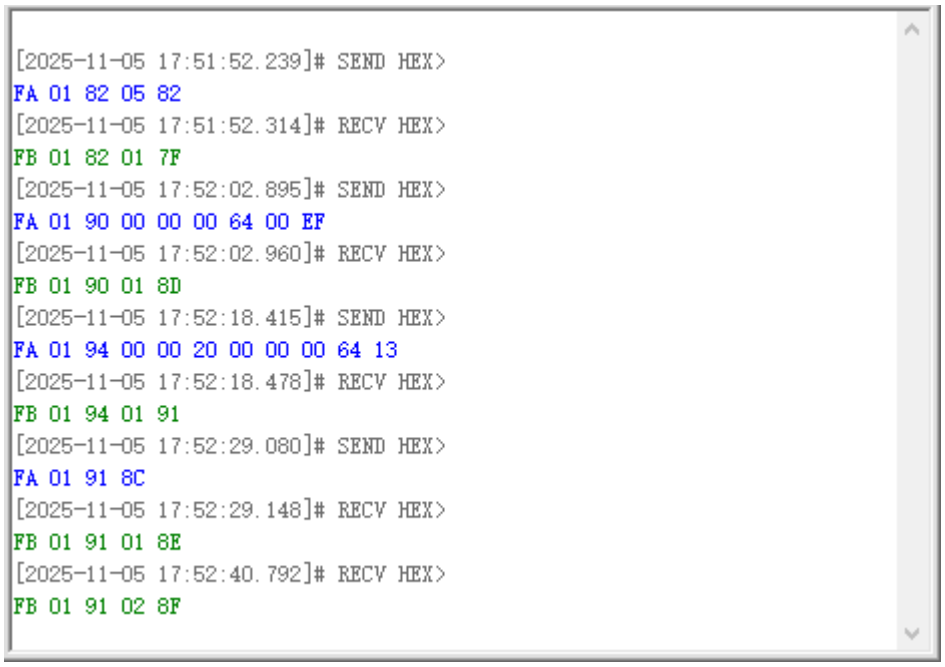
7.4 “return to zero by endstop”configuration example

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
   FA 01 82 05 82 (Bus FOC Mode)
2. Set trigger level , homing direction , and homing speed.
   FA 01 90 00 00 00 64 00 EF
3. Set to zero mode
   FA 01 94 00 00 20 00 00 00 64 13
4. Perform "return to zero"
   FA 01 91 8C

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



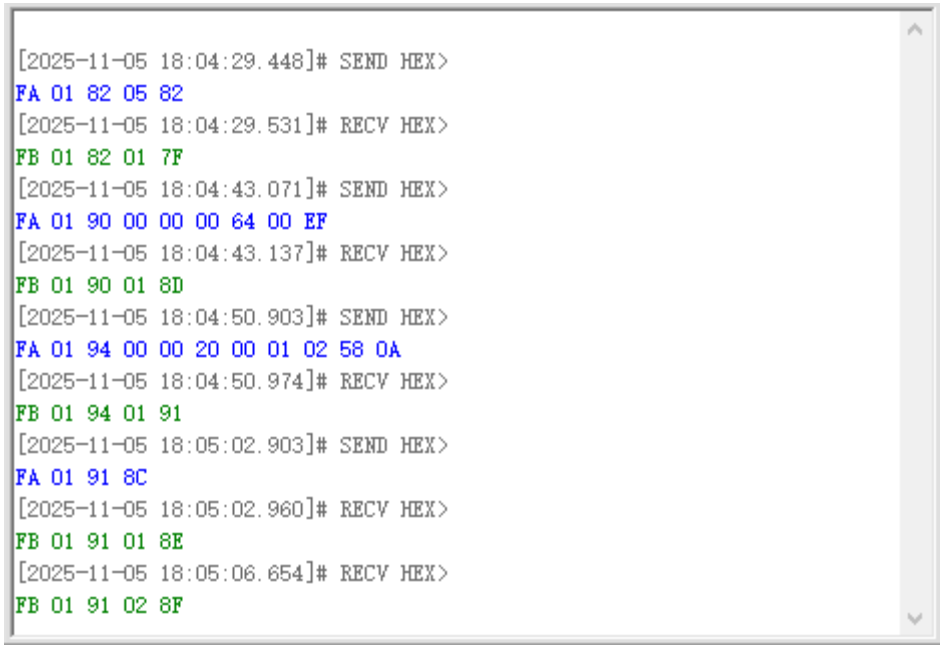
7.5 “return to zero by mechanical limit”configuration example

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
   FA 01 82 05 82 (Bus FOC Mode)
2. Set the zero -return direction and zero-return speed.
   FA 01 90 00 00 00 64 00 EF
3. Set origin offset , zero-return mode , and zero-return current.
   FA 01 94 00 00 20 00 01 02 58 0A
4. Perform "return to zero"
   FA 01 91 8C

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.



7.6 Single-lap zero-return configuration example

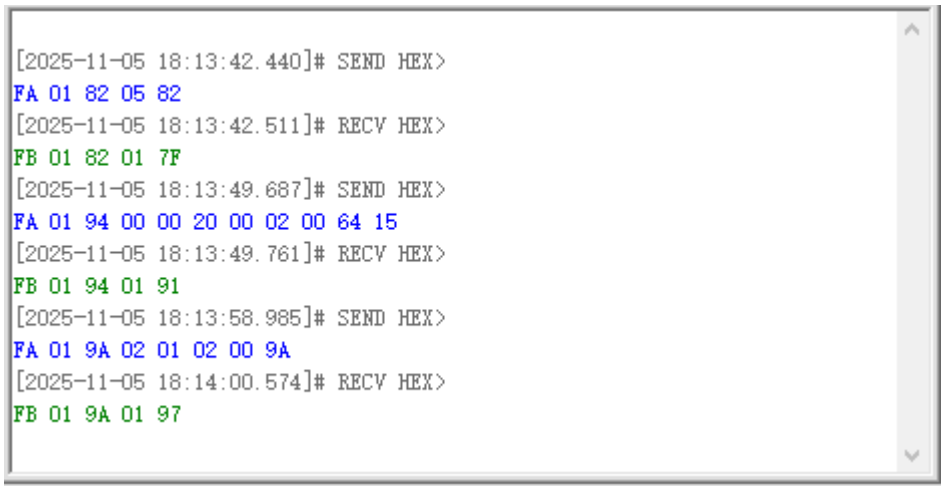
First, move the motor shaft to the appropriate "zero point" position.

The configuration steps are as follows:

1. Set working mode
   FA 01 82 05 82
2. Set to zero mode
   FA 01 94 00 00 20 00 02 00 64 15
3. Set the single-lap zero-return mode : " Zero Point " , " Speed " , " Direction ".
   FA 01 9 A 0 2 0 1 0 2 00 9A (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.



7.7 Example of coordinate homing

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.

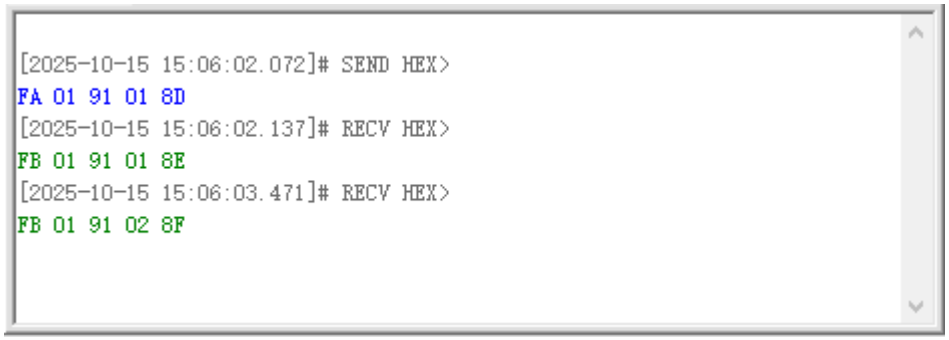
For the single-loop zero-return method, simply set the "zero point" coordinates.

1. Execute "Coordinate homing"

FA 01 91 01 8D

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, 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.

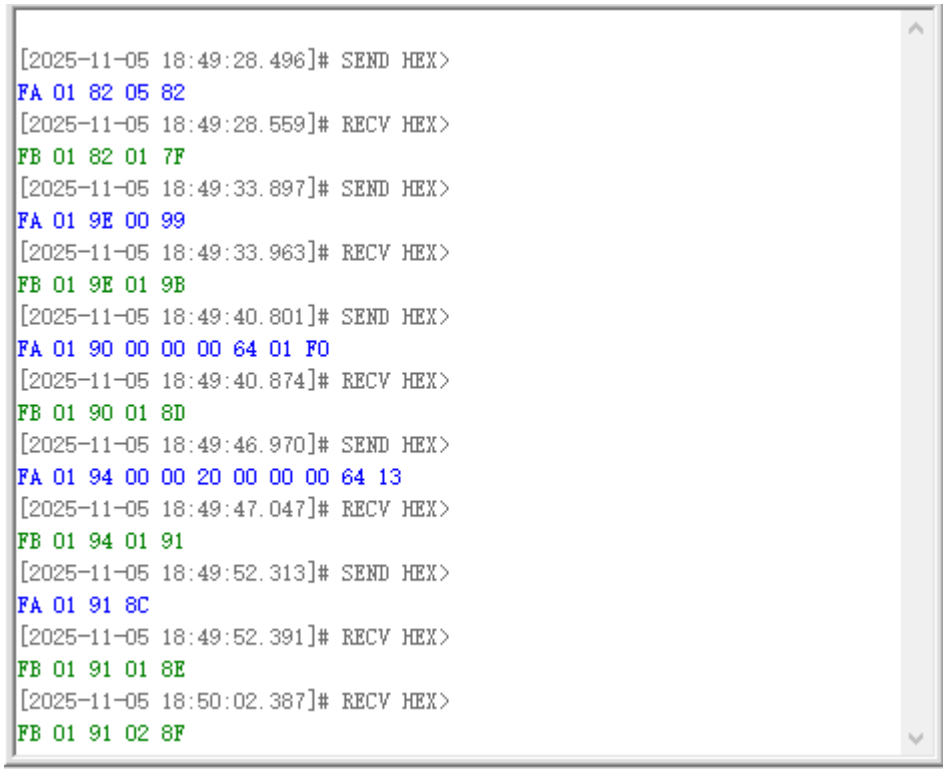
Whether the limit function is enabled is set by 90H, see 7.3.2.

8.2 Limit switch configuration example

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
   FA 01 82 05 82
2. Disable port mapping
   FA 01 9E 00 99 (No mapping required)
5. Set trigger level , homing direction , homing speed , and enable limit switch.
   FA 01 90 00 00 00 64 01 F0
6. Set to zero mode
   FA 01 94 00 00 20 00 00 00 64 13
7. Perform "return to zero"
   FA 01 91 8C
   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.

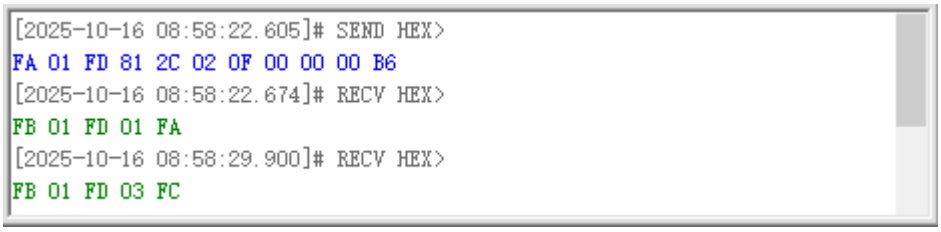


8.3 Left limit switch operation test

After configuring the parameters according to 8.2

FA 01 FD 81 2C 02 0F 00 00 00 B6

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

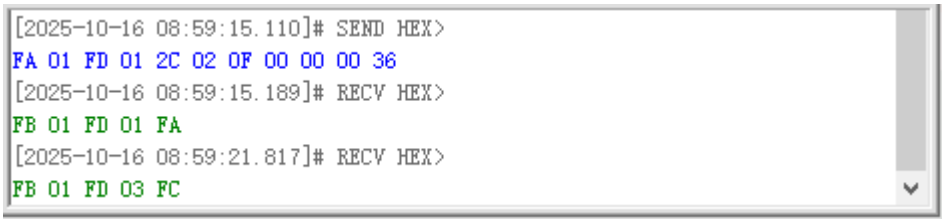


8.4 Right limit switch operation test

After configuring the parameters according to 8.2

FA 01 FD 01 2C 02 0F 00 00 00 36

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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	F1H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Motor status	Checksum
FB	addr	F1H	status	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 the detailed format of the 95H command 5.2.17.

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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable	Checksum
FA	addr	F3H	enable	CRC

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

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	F3H	status (uint8_t)	CRC

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

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

9.2.3 Emergency stop command for motor

The emergency stop command is as follows:

Downlink frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	addr	F7H	CRC

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	state	Checksum
FB	addr	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, the motor can be controlled to run continuously at the set acceleration and speed; it also supports the function of automatically stopping the motor after running for a period of time.

10.1 Run the motor in speed mode

The execution command format is as follows:

Downlink frame (PC → SERVOxxD)								
Byte 1	Byte 2	Byte 3	Byte 4			Byte 5	Byte 6	Byte 7
header	address	code	dir	rev	speed		acc	CRC
FA	addr	F6H	b7	b6-b4	b 3 -b0	b7-b0	acc	CRC
			dir	- -	speed			

Byte 4: The highest bit indicates the direction, the lower 4 bits and Byte 5 together indicate the speed

Byte 5: The lower 4 bits of Byte 5 and Byte 4 together indicate speed

The parameter description is as follows:

addr: slave address, the value range is 0-255

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 "FA 01 F6 02 80 02 75",

the motor rotates forward at acc=2, speed=640RPM

Send "FA 01 F6 82 80 02 75",

the motor reverses at acc=2, speed=640RPM

If the motor needs to stop automatically after running for a period of time, the running instructions are as follows

Byte1	Byte2	Byte3	Byte4			Byte5	Byte6	Byte7-10	Byte11
Head	addr	Func	dir	rev	speed		acc	runTime	CRC
FA	addr	F6	b7	b6-b4	b3-b0	b7-b0	acc	runTime	CRC
			dir	--	speed				

runTime (00H ~ 0x0FFFFFFF) uints: 10ms

Uplink package (PC ← SERVO42D/57D)				
Head	Slave addr	Function	Data	CRC
FB	01	F6	status(uint8_t)	CRC

status = 0 run fail.

status = 1 run success.

status = 2 run complete.

status = 5 Data has received for Synchronous mode.

10.2 Stop the motor in speed mode

Downlink package (PC → SERVO42D/57D)								
Byte1	Byte2	Byte3	Byte4			Byte5	Byte6	Byte7
Head	Slave addr	Function	dir	Rev	speed		acc	CRC
FA	addr	F6	b7	b6-b4	b3-b0	b7-b0	acc	CRC
			0	0	0			

The stop command can stop the motor slowly, or stop the motor immediately.

When setting acc ≠ 0, the motor decelerates and stops slowly

When setting acc = 0, the motor stops immediately

① Deceleration and stop the motor slowly (acc ≠ 0)

for example:

Send FA 01 F6 00 00 02 F3

Stop the motor with deceleration acc=2

② Immediate stop command (acc = 0)

for example:

Send FA 01 F6 00 00 00 F1

Stop the motor immediately

Note1: If the motor rotating more than 1000RPM, it is not a good idea to stop the motor immediately!

Uplink package (PC ← SERVO42D/57D)				
Head	Slave addr	Function	Data	CRC
FB	01	F6	status(uint8_t)	CRC

status = 0 stop the motor fail.

status = 1 start to stop the motor.

status = 2 stop the motor success.

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

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 frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Enable	Checksum
FA	addr	FFH	enable	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.

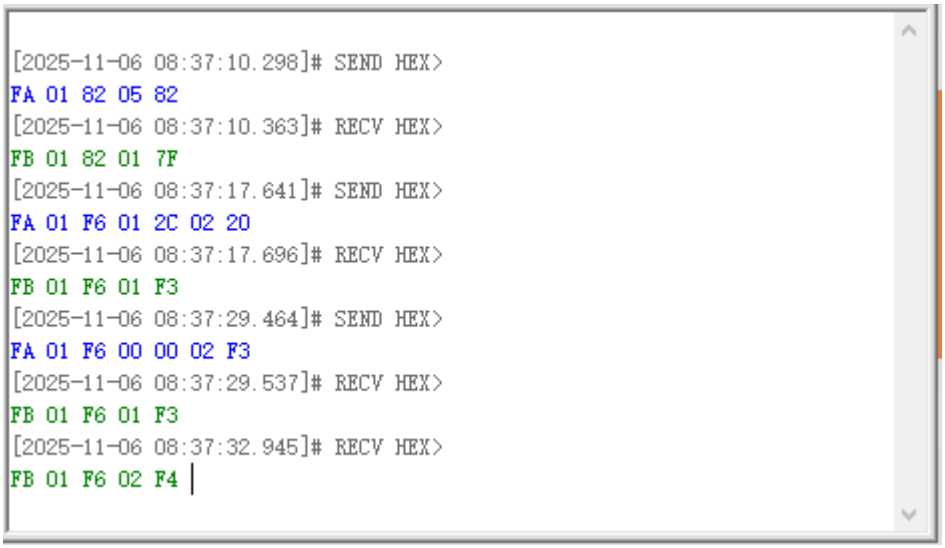
Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	FFH	status (uint8_t)	CRC

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

10.4 Speed mode running example

1. Set working mode
   FA 01 82 05 82
2. Set the running direction, speed, and acceleration parameters.
   FA 01 F6 01 2C 02 20
   At this point, the motor can be observed to run in the set direction, speed, and acceleration.
   To stop the operation, execute the stop command.
3. Stop command
   FA 01 F6 00 00 02 F3
   At this point, the motor can be observed to stop at the set acceleration.

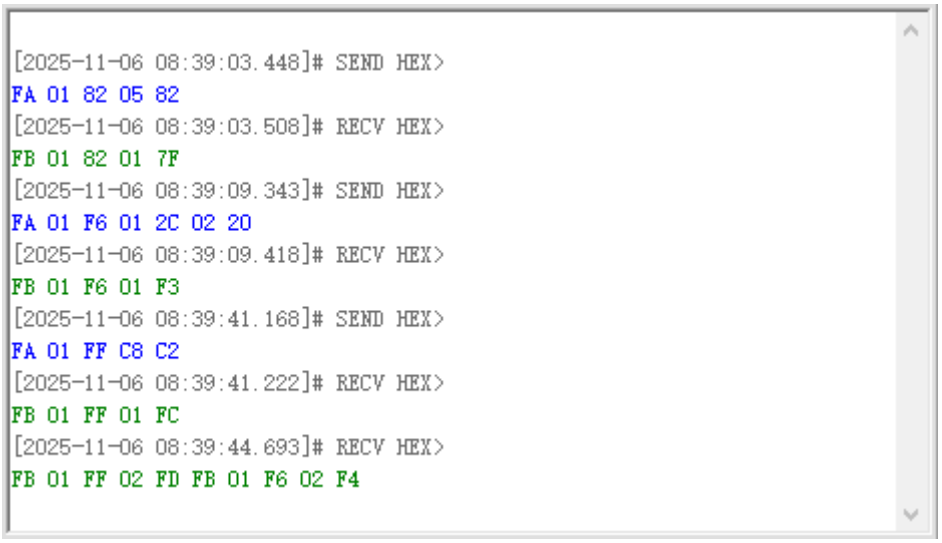


10.5 automatic operation upon power-on Example

1. Set working mode
   FA 01 82 05 82
2. Set the running direction, speed, and acceleration parameters.
   FA 01 F6 01 2C 02 20
   At this point, the motor can be observed to run in the set direction, speed, and acceleration.
3. Enable automatic operation upon power-on
   FA 01 FF C8 C2
   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:
FA 01 FF CA C4



Part 11. Position control mode description

11.1 Position mode1: relative motion by pulses

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 Run the motor in position mode1

Downlink package (PC → SERVO42D/57D)									
Byte1	Byte2	Byte3	Byte4			Byte5	Byte6	Byte7-10	Byte11
Head	Slave addr	Function	dir	Rev	speed		acc	pulses	CRC
FA	addr	FD	b7	b6-b4	b3-b0	b7-b0	acc	relPulses	CRC
			dir	--	speed				

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

Byte 5: The lower 4 bits of Byte 5 and Byte 4 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 - 0xFFFFFFFF

for example:

Send FA 01 FD 02 80 02 00 00 FA 00 76,

the motor rotates 20 times in the forward direction with acc=2, speed=640RPM (16 subdivisions);

Send FA 01 FD 82 80 02 00 00 FA 00 F6,

the motor rotates 20 times in the reverse direction with acc=2, speed=640RPM (16 subdivisions);

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	FDH	status (uint8_t)	CRC

The operation failed at position status = 0.

status = 1, start running

status = 2 position completed

status = 3 Stopped upon touching limit switch

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 frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6	Bytes 7-10	11 bytes
header	address	code	data	acceleration	data	Checksum
FA	addr	FDH	00	acc	00	CRC

The stop command can stop the motor slowly, or stop the motor immediately.

When setting acc ≠ 0, the motor decelerates and stops slowly

When setting acc = 0, the motor stops immediately

① Deceleration and stop the motor slowly (acc ≠ 0)

for example:

Send FA 01 FD 00 00 02 00 00 00 00 FA

Stop the motor with deceleration acc=2

② Immediate stop command (acc = 0)

for example:

Send FA 01 FD 00 00 00 00 00 00 00 F8

Stop the motor immediately

Note1: If the motor rotating more than 1000RPM, it is not a good idea to stop the motor immediately!

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	FDH	status (uint8_t)	CRC

status = 0 stop the motor fail.

status = 1 stop the motor starting….

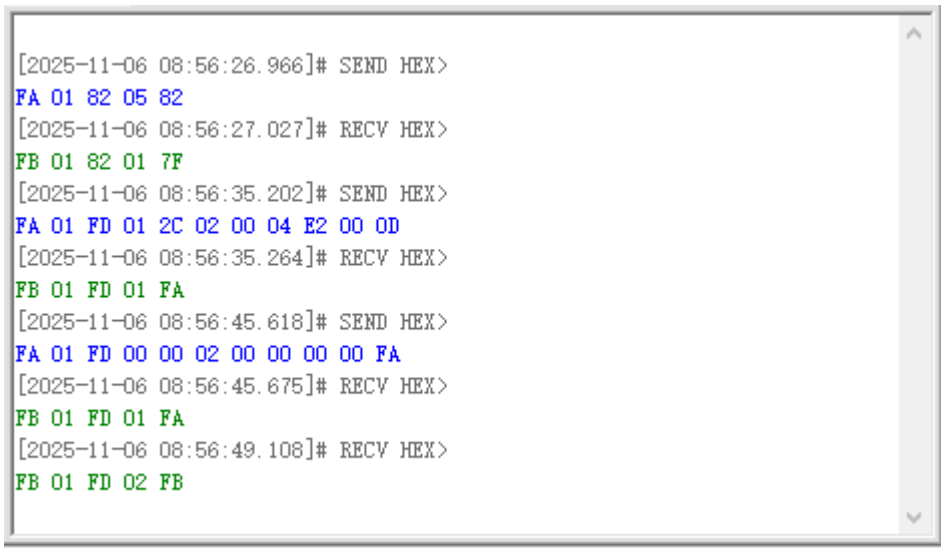
status = 2 stop the motor complete.

status = 3 stop by endstop.

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

11.1.3 Pulse Relative Operation Example

1. Set working mode
   FA 01 82 05 82
2. Set the running direction, speed , acceleration , and relative pulse count.
   FA 01 FD 01 2C 02 00 04 E2 00 0D
   (At speed = 300 RPM , acceleration = 2, 100 revolutions clockwise (16 subdivisions))
3. slow down and stop the motor during operation , the command is as follows:
   FA 01 FD 00 00 02 00 00 00 00 FA
4. If an emergency stop is required during operation, the command is as follows:
   FA 01 F7 F2



11.2 Position mode2: absolute motion by pulses

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 0x91 or 0x92) to mark the "zero point".

11.2.1 Run the motor in position mode2

The execution command format is as follows:

Downlink frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Byte 4 - 5	Byte 6	Bytes 7-10	11 bytes
header	address	code	speed	acc	Absolute pulse	Checksum
FA	addr	FEH	speed	acc	absPulses	CRC

The parameter description is as follows:

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

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

absPulses: the absolute pulses, int32_t

For example:

If the current axis is any value

Send FA 01 FE **02 58 02** 00 00 0C 80 95

The motor runs at a speed of **600** (RPM), acceleration **2**, and absolute forward rotation of 3200 pulses (1 circle at 16 subdivisions).

After move the pulses is 0x0C80.

If the current axis is any value

Send FA 01 FE **02 58 02** FF FF F3 80 13

The motor runs at a speed of **600** (RPM), acceleration **2**, and absolute reverse rotation to 3200 pulses (1 circle at 16 subdivisions).After move the pulses is -0x0C80.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	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 frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6	Bytes 7-10	bytes11
header	address	code	data	acc	data	crc
FA	addr	FEH	00	acc	00	CRC

The stop command can stop the motor slowly, or stop the motor immediately.

When setting acc ≠ 0, the motor decelerates and stops slowly

When setting acc = 0, the motor stops immediately

① Deceleration and stop the motor slowly (acc ≠ 0)

for example:

Send FA 01 FE 00 00 04 00 00 00 00 FD

Stop the motor with deceleration acc=4

② Immediate stop command (acc = 0)

for example:

Send FA 01 FE 00 00 00 00 00 00 00 F9

Stop the motor immediately

Note1: If the motor rotating more than 1000RPM, it is not a goog idea to stop the motor immediately!

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	FEH	status (uint8_t)	CRC

status = 0 stop fail.

status = 1 stop starting….

status = 2 stop complete.

status = 3 end limit stoped.

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

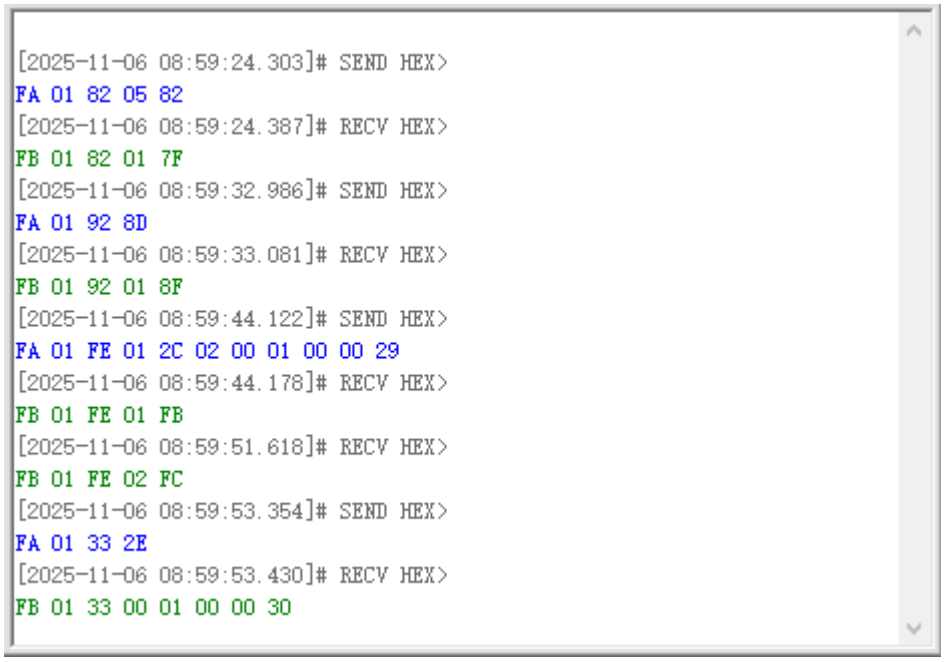
11.2.3 Pulse Absolute Operation Example

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
   FA 01 82 05 82
2. Set zero point
   FA 01 92 8D
3. Set the running speed , acceleration , and absolute pulse count.
   FA 01 FE 01 2C 02 00 01 00 00 29
   (Run at speed = 300 RPM , acc = 2, to pulse position 0x10000)
4. After the process is complete, you can view the number of pulses.
   FA 01 33 2E
5. to slow down and stop the motor during operation , the command is as follows:
   FA 01 FE 00 00 02 00 00 00 00 FB
6. If an emergency stop is required during operation, the command is as follows:
   FA 01 F7 F2



11.3 Position mode3: relative motion by axis

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 Run the motor in position mode3

The execution command format is as follows:

Downlink frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Byte 4 - 5	Byte 6	Bytes 7-10	bytes11
header	address	code	speed	acc	relative axis	crc
FA	addr	F4	speed	acc	relAxis	CRC

he parameter description is as follows:

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

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

relAxis: the relative axis, int32_t

For example:

If the current axis is 0x8000.(read by code "31")

Send FA 01 F4 **02 58 02 00 00 40 00** 8B

The motor will relative move **0x4000** (speed = 600(RPM),acc =2)

After move the axis is 0xC000.(0x8000+0x4000=0xC000)

If the current axis is 0x8000.(read by code "31")

Send FA 01 F4 **02 58 02 FF FF C0 00** 03

The motor will relative move **-0x4000** (speed = 600(RPM),acc =2)

After move the axis is 0x4000.(0x8000-0x4000=0x4000)

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	F4H	status (uint8_t)	CRC

The operation failed at position status = 0.

status = 1, start running

status = 2 position completed

status = 3 Stopped upon touching limit switch

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 frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6	Bytes 7-10	bytes11
header	address	code	data	acc	data	crc
FA	addr	F4H	00	acc	00	CRC

The stop command can stop the motor slowly, or stop the motor immediately.

When setting acc ≠ 0, the motor decelerates and stops slowly

When setting acc = 0, the motor stops immediately

① Deceleration and stop the motor slowly (acc ≠ 0)

for example:

Send FA 01 F4 00 00 04 00 00 00 00 F3

Stop the motor with deceleration acc=4

② Immediate stop command (acc = 0)

for example:

Send FA 01 F4 00 00 00 00 00 00 00 EF

Stop the motor immediately

Note1: If the motor rotating more than 1000RPM, it is not a good idea to stop the motor immediately!

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	F4H	status (uint8_t)	CRC

status = 0 stop fail.

status = 1 stop starting….

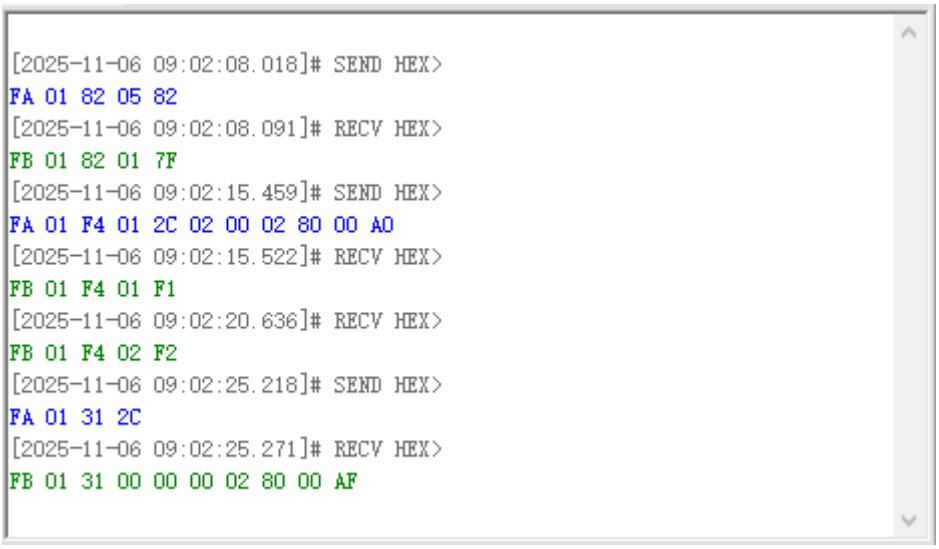
status = 2 stop complete.

status = 3 end limit stoped.

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

11.3.3 Axis relative operation example

1. Set working mode
   FA 01 82 05 82
2. Set the running speed , acceleration , and relative coordinates.
   FA 01 F4 01 2C 02 00 02 80 00 A0
   (with speed = 300 (RPM) , a cc = 2, relative to coordinates 0x28000)
3. After the process is complete, you can view the coordinates.
   FA 01 31 2C
4. to slow down and stop the motor during operation , the command is as follows:
   FA 01 F4 00 00 02 00 00 00 00 F1
5. If an emergency stop is required during operation, the command is as follows:
   FA 01 F7 F2



11.4 Position mode4: absolute motion by axis

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 0x91 or 0x92) to mark the "zero point".

11.4.1 Run the motor in position mode4

The execution command format is as follows:

Downlink frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Byte 4 - 5	Byte 6	Bytes 7-10	Bytes11
header	address	code	speed	acc	absolute axis	crc
FA	addr	F5	s peed	acc	absAxis	CRC

The parameter description is as follows:
speed: the speed, the value range is 0-3000(RPM)
acc: the acceleration, the value range is 0-255
AbsAxis: the absolute axis, int32_t
For example:
If the current axis is any value
Send FA 01 F5 **02 58 02 00 00 40 00** 8C
The motor will move to **0x4000** (speed = **600**(RPM),acc =**2**)
After move the axis is 0x4000.
If the current axis is any value
Send FA 01 F5 **02 58 02 FF FF C0 00** 0A
The motor will move to **-0x4000** (speed = **600**(RPM),acc =**2**)
After move the axis is -0x4000.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	F5H	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 frame (PC → SERVOxxD)						
Byte 1	Byte 2	Byte 3	Bytes 4-5	Byte 6	Bytes 7-10	11 bytes
Frame header	Slave address	function code	control word	acceleration	control word	Checksum
FA	addr	F5H	00	acc	00	CRC

The stop command can control the motor to slow down and stop gradually, or it can control the motor to stop immediately.
When acc is set to ≠ 0 , the motor slows down and stops gradually.
When acc is set to 0 , the motor stops immediately.

① Slow down and stop command (acc ≠ 0)
for example:
Send FA 01 F5 0 0 00 02 00 00 00 00 F 2
Let the motor decelerate at an acceleration acc = 2. Stop rotating

② Immediate stop command (acc = 0)
for example:
Send FA 01 F5 0 0 00 00 00 00 00 00 F0
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 frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Frame header	Slave address	function code	Setting status	Checksum
FB	addr	F5H	status (uint8_t)	CRC

Status = 0 - Stop failed
status = 1 (Stop/ Start)
status = 2 position stopped completing
status = 3 Stopped upon touching limit switch

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

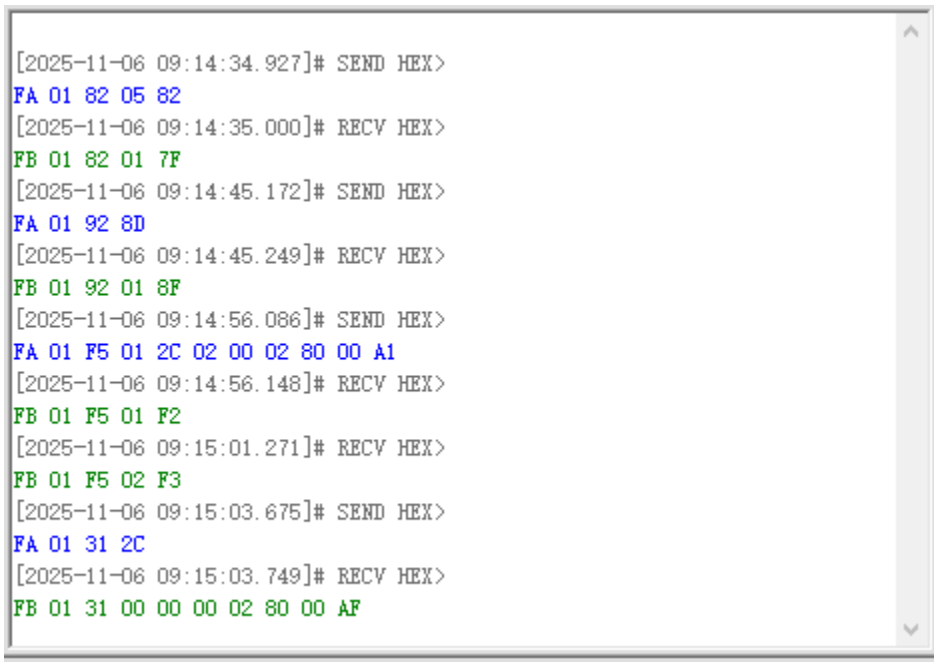
11.4.3 Axis absolute operation example

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
   FA 01 82 05 82
2. Set zero point
   FA 01 92 8D
3. Set the running speed , acceleration , and absolute coordinates.
   FA 01 F5 01 2C 02 00 02 80 00 A1
   (Run at speed = 300 RPM , acc = 2, to coordinate position 0x28000)
4. After the process is complete, you can view the coordinates.
   FA 01 31 2C
5. to slow down and stop the machine during operation , the command is as follows:
   FA 01 F5 00 00 02 00 00 00 00 F2
6. If an emergency stop is required during operation, the command is as follows:
   FA 01 F7 F2



11.4.4 Real-time updates examples

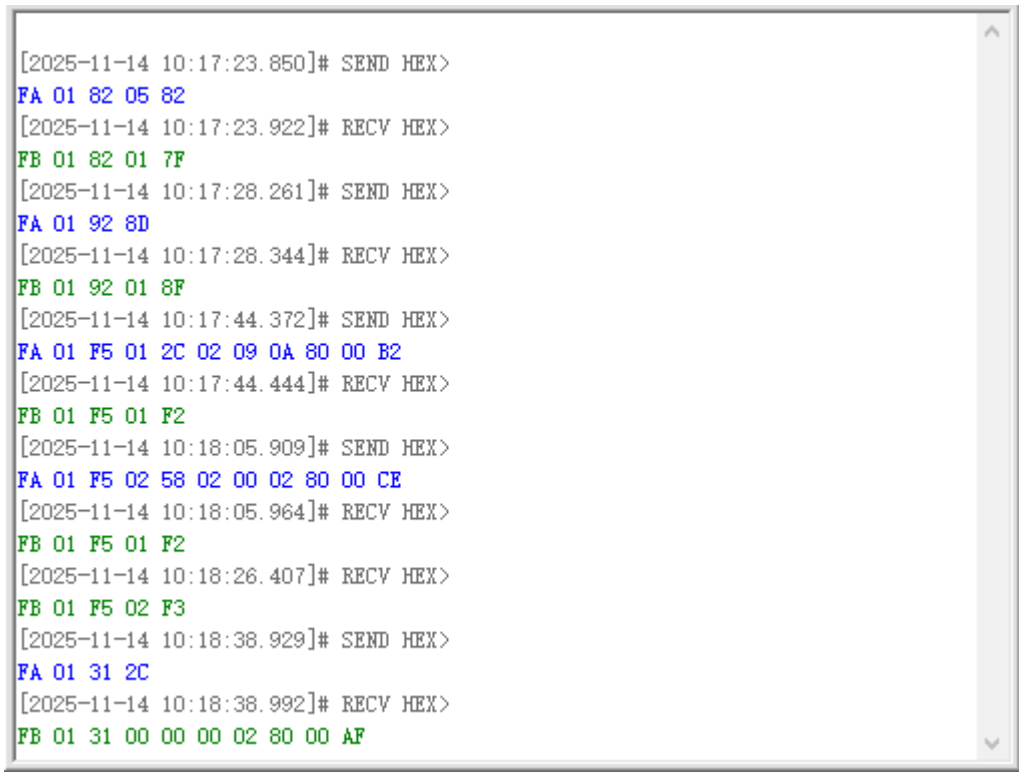
This mode supports real-time updates of speed and coordinates, meaning that new commands can be issued to change speed and coordinates while the previous command is running.

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
   FA 01 82 05 82
2. Set zero point
   FA 01 92 8D
3. Set the running speed , acceleration , and absolute coordinates.
   FA 01 F5 01 2C 02 09 0A 80 00 B2
   (Run at speed = 30 0 RPM , a cc = 2 , to coordinate position 0x90A8000)
   Wait for the motor to run for about 20 seconds.
4. Real-time update speed , absolute coordinates
   FA 01 F5 02 58 02 00 02 80 00 CE
   (Updated to 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.
   FA 01 31 2C
6. to slow down and stop the machine during operation , the command is as follows:
   FA 01 F5 00 00 02 00 00 00 00 F2
7. If an emergency stop is required during operation, the command is as follows:
   FA 01 F7 F2


[2025-11-14 10:17:23.850]# SEND HEX>
FA 01 82 05 82
[2025-11-14 10:17:23.922]# RECV HEX>
FB 01 82 01 7F
[2025-11-14 10:17:28.261]# SEND HEX>
FA 01 92 8D
[2025-11-14 10:17:28.344]# RECV HEX>
FB 01 92 01 8F
[2025-11-14 10:17:44.372]# SEND HEX>
FA 01 F5 01 2C 02 09 0A 80 00 B2
[2025-11-14 10:17:44.444]# RECV HEX>
FB 01 F5 01 F2
[2025-11-14 10:18:05.909]# SEND HEX>
FA 01 F5 02 58 02 00 02 80 00 CE
[2025-11-14 10:18:05.964]# RECV HEX>
FB 01 F5 01 F2
[2025-11-14 10:18:26.407]# RECV HEX>
FB 01 F5 02 F3
[2025-11-14 10:18:38.929]# SEND HEX>
FA 01 31 2C
[2025-11-14 10:18:38.992]# RECV HEX>
FB 01 31 00 00 00 02 80 00 AF

Part 12. Multi- motor synchronous motion description

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

Method 1:

Sending a command using broadcast address 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 address, 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:

Using multi-instruction data frames, a single instruction contains different control commands for multiple motors, suitable for different motors on the bus to execute different motion commands. This method can control up to 5 motors simultaneously.

Method 4:

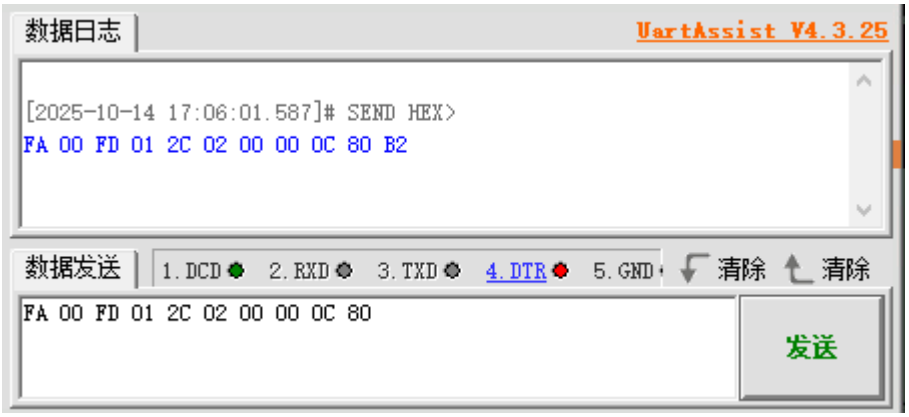
The multi-motor synchronization flag function is used to enable different motors on the bus to execute different motion commands. This method allows for the control of an unlimited number of motors.

12.1 synchronous motion mode 1 - Broadcast mode

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

FA 00 FD 01 2C 02 00 00 0C 80 B2

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



12.2 synchronous motion mode 2 - grouping method

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

FA 50 FD 01 2C 02 00 00 0C 80 02

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

See the section 5.2.15 for group address settings .

12.3 synchronous motion mode 3_Multi-command data frame

Multi-command data frames, meaning a single frame can contain up to five commands. The slave determines which command to execute based on the address. This means a single frame can control up to five motors.

If there are more than five motors, multi-motor synchronous control can be used; see Section 6.9.

If there are fewer than five motors, all remaining command Bytes are set to 0.

Long data packets format:

Head	0xFC				1Byte
	Byte 1	Byte 2	…	Byte 10	
command 1	slaveAddr1	code	…		10 Byte
command 2	slaveAddr 2	code	…		10 Byte
command 3	slaveAddr 3	code	…		10 Byte
command 4	slaveAddr 4	code	…		10 Byte
command 5	slaveAddr 5	code	…		10 Byte
checksum	CRC				1 Byte

Note:

1. The length of the long data packet is 52 Bytes in total.
2. The length of each command X is 10 Bytes, when it is less than 10 Bytes, add 0 to supplement.
3. Command X is the corresponding ordinary command, remove the frame header (FA) and checksum.
4. If the slave addresses of command X and command Y (X<Y) are the same, only command X is executed.
5. Slave does not answer.

For example, sending the following long data packet can control 5 motors to perform different actions (16 subdivisions)

```
FC
01 F6 00 32 0A 00 00 00 00 00
02 FD 01 2C 02 00 04 E2 00 00
03 FE 02 58 02 00 04 E2 00 00
04 F4 02 58 64 00 0C 80 00 00
05 F5 04 B0 C8 00 0C 80 00 00
CA
```

```
FC 01 F6 00 32 0A 00 00 00 00 00 02 FD 01 2C 02 00 04 E2 00 00 03 FE 02
58 02 00 04 E2 00 00 04 F4 02 58 64 00 0C 80 00 00 05 F5 04 B0 C8 00 0C
80 00 00 CA
```

Motor 1 rotates continuously forward in speed mode (speed=0x32, acc=0x0A).

Motor 2 rotates forward 100 times in position mode 1 (speed=0x12C, acc=0x02).

Motor 3 rotates forward 100 times in position mode 2 (speed=0x258, acc=0x02).

Motor 4 moves to coordinate 0xC8000 in position mode 3 (speed=0x258, acc=0x64).

Motor 5 moves to coordinate 0xC8000 in position mode 4 (speed=0x4B0, acc=0xC8).

12.4synchronous motion mode 4 - 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.

12.4.1 Command to enable/disable synchronization function

The configuration command is as follows:

Downlink frame (PC → SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	status	CRC
FA	addr	4A	enable	CRC

enable = 00 disables the synchronization function (default).
enable = 01 Enables the synchronization function.

Returned data:

Uplink frame (PC ← SERVOxxD)				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
header	address	code	Setting status	Checksum
FB	addr	4AH	status (uint8_t)	CRC

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

12.4.2 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 frame (PC → SERVOxxD)			
Byte 1	Byte 2	Byte 3	Byte 4
header	address	code	Checksum
FA	00	4B	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.

12.4.3 Example of multi-motor synchronous operation

Taking a bus with 5 motors as an example, different motion commands are executed synchronously.

1. Setting the operating mode: RS485 bus closed-loop FOC mode

   Motor 1: FA 01 82 05 CRC

   …

   Motor 5: FA 01 82 05 CRC

2. Set 4AH parameters: Enable multi-motor synchronous control function.

   Motor 1: FA 01 4A 01 CRC

   …

   Motor 5: FA 05 4A 01 CRC

3. Set the motion mode and parameters for each motor separately.

(1) Motor 1: Speed mode, forward rotation, speed 640 RPM, acceleration 2

FA 01 F6 0 2 80 02 75

(2) Motor 2: Position control mode 1, reverse, speed 100 RPM, acceleration 2 , 320000 pulses (100 revolutions in 16 microsteps).

FA 0 2 FD 8 0 64 02 00 04 E2 00 CRC

(3) Motor 3: Position control mode 2, reverse, speed 100 RPM, acceleration 2 , 320000 pulses (100 revolutions in 16 microsteps).

FA 03 FE 8 0 64 02 00 04 E2 00 CRC

(4) Motor 4: Position control mode 3, forward rotation, speed 300 RPM, acceleration 200 , running to coordinate 0x280000 (160 revolutions in 16 microsteps).

FA 04 F4 0 1 2C C8 02 80 00 00 CRC

(5) Motor 5: Position control mode 4, forward rotation, speed 300 RPM, acceleration 200, running to coordinate 0x280000 (160 revolutions in 16 microsteps).

FA 05 F5 0 1 2C C8 02 80 00 00 CRC

4. Send synchronization command: All motors with the above-configured parameters will begin synchronous operation.

   FA 00 4B CRC (1MS interval)

   FA 00 4B CRC (1MS interval)

   …

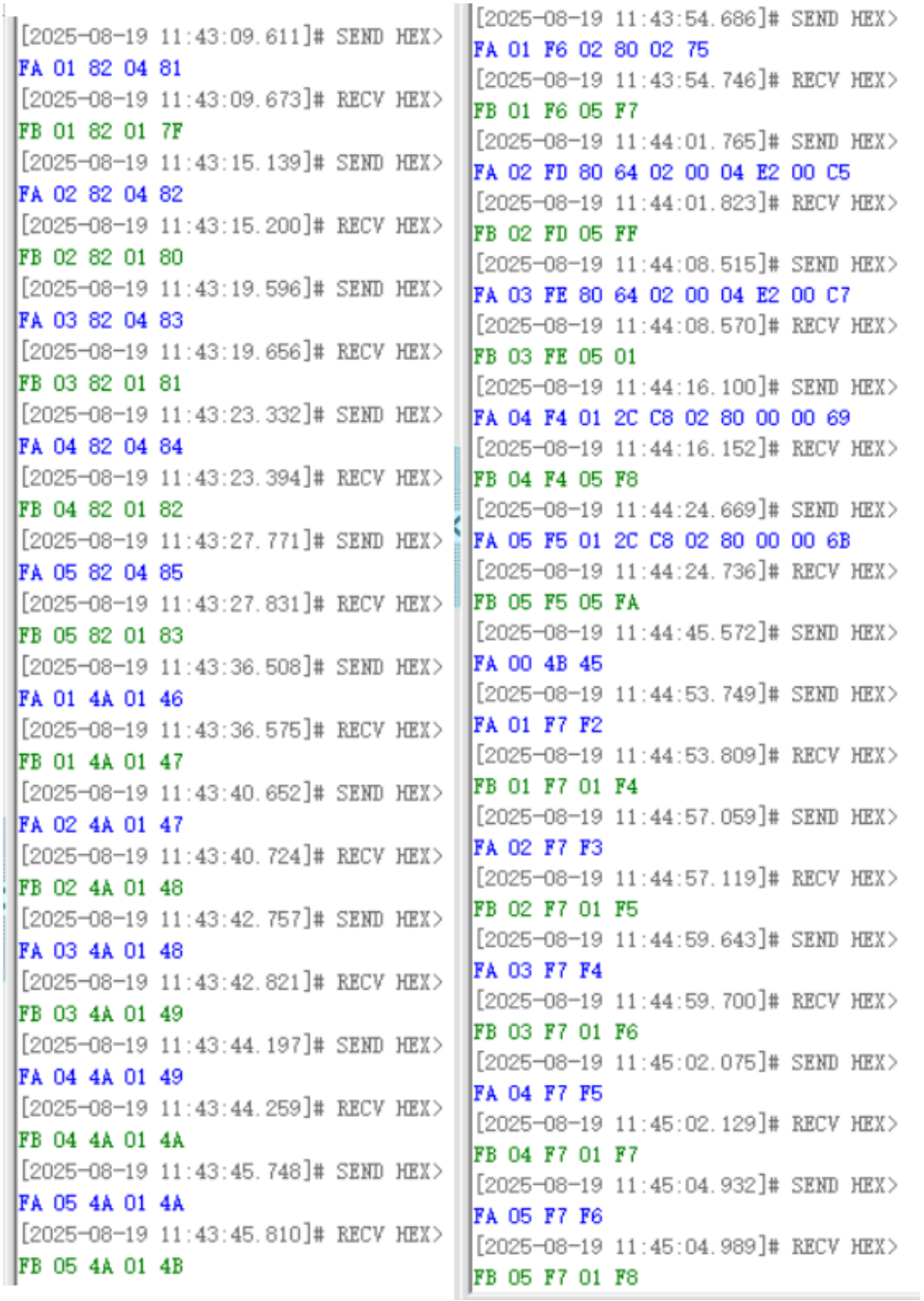
5. Motor stop: During operation, a command can be issued to stop the motor.

   FA 01 F7 CRC

   …

   FA 05 F7 CRC

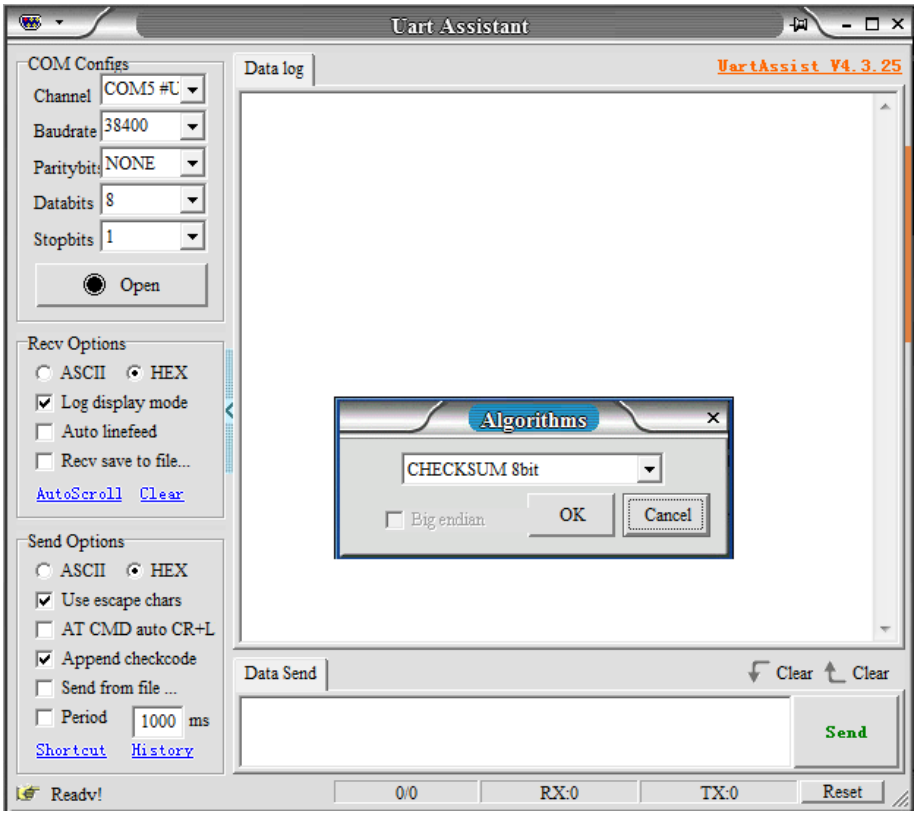
See the image below for detailed configuration instructions:



Part 13. Serial Port Assistant Setup Instructions

13.1 Serial Port Assistant Configuration Example

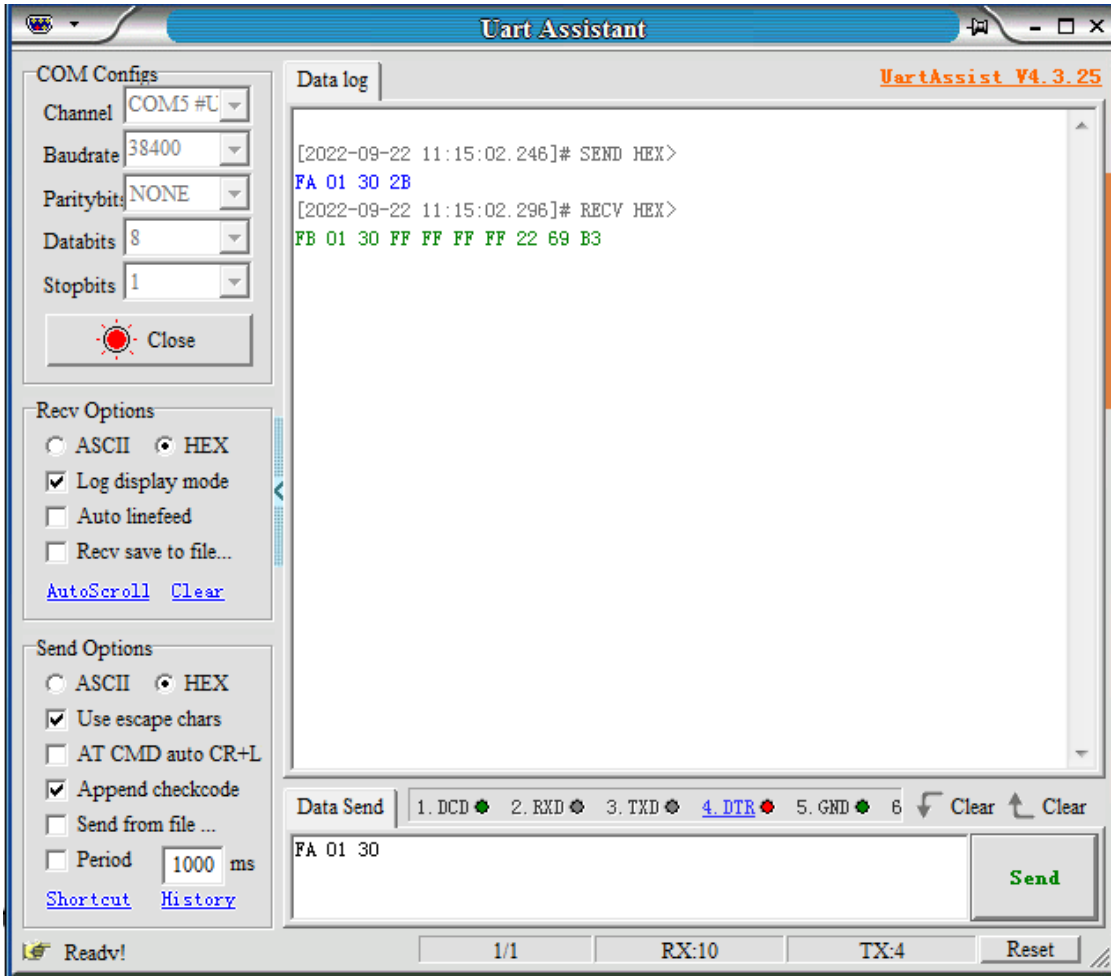
Select the serial port number: (COMxx)
Select baud rate: 38400
Select check digit : NONE
Select data bits : 8
Select stop bit: 1
Receive settings, select: Hex
Send settings, select: Hex
Automatically send checksum bit , select: CHECKSUM 8 bit
As shown in the image below:



13.2 Example of reading encoder values

Send FA 01 30 2B

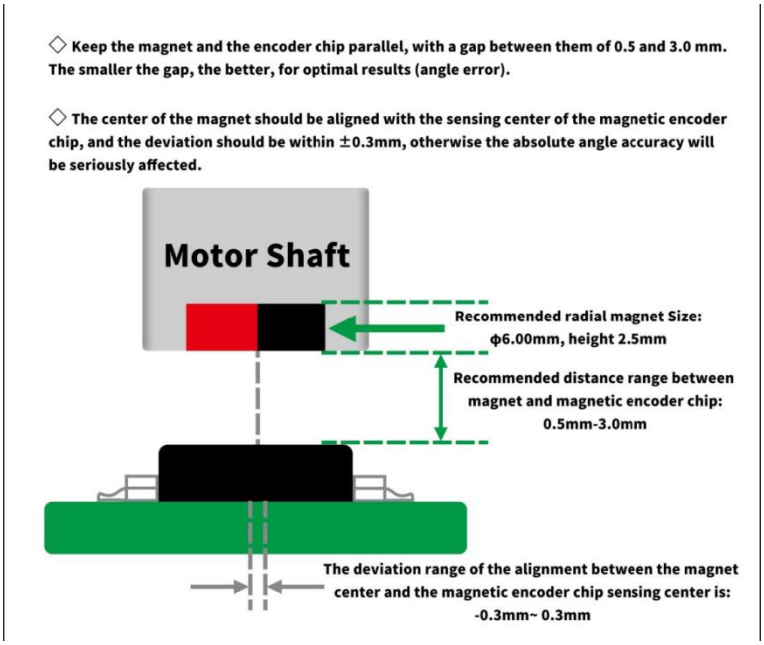
Return to FB 01 30 FF FF FF FF 22 69 B3



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 40 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