Kinco步科

MD series integrated Servo Motor

en.kinco.cn

Shenzhen kinco Electric Co., Ltd.

PrefaceProduct Acceptance

Thank you for using Kinco Servo product!

The accessories of Kinco every series & different types drivers are different. We advice youaccept products before use.

Item for Acceptance	Remark
Whether the model of a delivered MD series integrated servo motor is consistent with the specified model	Check the nameplate of a servo motor and that of a servodriver
Whether the accessories included in the packing listare complete	Check the packing list
Whether any breakage occurs	Check the external appearance completely for any lossesthat are caused by transportation

MD series integrated servo motor Parts list

port		Docking terminal		
		Name	Specification model	Quantity
X1 (CANor RS485)		Plug	ZER-04V-S	2
		Pin	SZE-002T-P0.3	8
X2 RS232		2.0mm 4P Plug	CJT A2008H-04P	1
		Single row of metal pins	CJT A2008-TP	4
X3 IO		Terminal (Head)	CJT A2008H-2x6P	1
		Crimp pin	CJT A2008-TP-A	12
X5	MD60Power input	Terminal (Head)	DINKLE 0226-0704	1
	MD80Power input	Terminal (Head)	DINKLE 0227-0704	1

If there is any problem with any of the above, please contact our company or your supplier to solve it.

Catalog

Chapter 1 System configuration and types

1.1 Product specification

Servo integrated models		MD60-020-D□■K-★ A-000	MD60-040-D□■K-★ A-000	MD80-075-D□■K-★ A-000
power		24VDC～70VDC	24VDC～70VDC	24VDC～70VDC
Rated powerPn(W)		200	400	750
Rated speednN(rpm)		3000	3000	3000
Rated torqueTs(Nm)		0.64	1.27	2.39
Maximum torque Tm(Nm)		1.92	3.81	7.17
Rotor moment of inertiaJm(Kg·cm²)		0.214	0.405	1.087
Brake chopper		Via wiring an external braking resistor（mainly in quick start and stop application）		
Brake chopper threshold		DC73V ± 2V（Default value, Adjustable via software）		
Over-voltage alarming threshould		DC83V ± 2V		
Under-voltage alarming threshould		DC18V±2V		
Cooling method		Natural air cooling		
weight		1.2kg	1.6kg	2.9kg
General function	Input specification	COMI terminal for 4 digital inputs		
	Output specification	COMO terminal for 2 digital outputs		
	Pulse direction control	Pulse+direction 、CCW+CW、 phase A+ phase B（5V~24V）		
	Brake	Built-in brake power supply 24V maximum current 0.5A		Built-in brake power supply 24V maximum current 1 A
	RS232	Default baudrate setting is 38400，the max. baudrate is 115.2KHz, use Kinco software to communicate with PC, or via free protocol to communicate with controller.		
RS485		The max. baudrate is 115.2KHz, use Modbus RTU protocol to communicate with controller.		

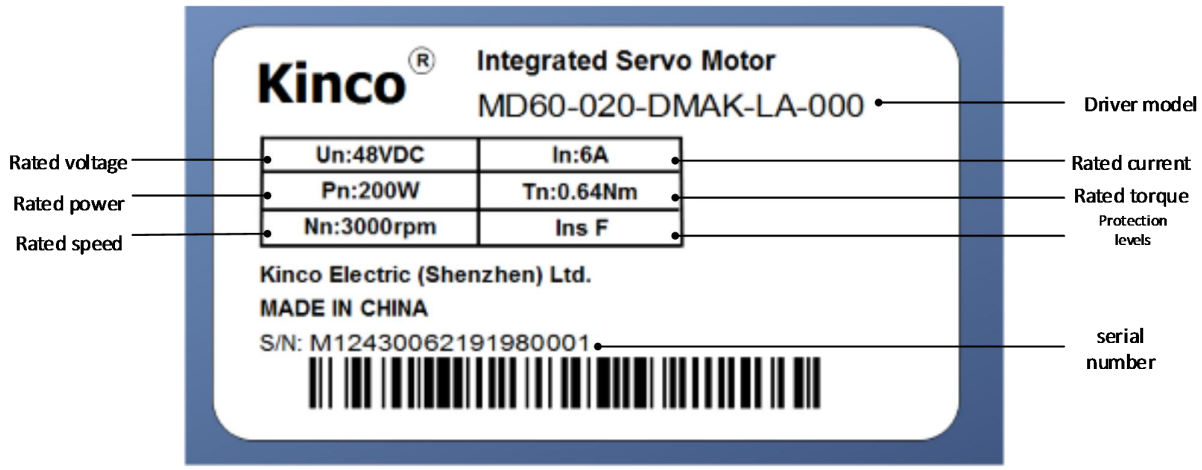
CAN BUS	Support maximum 1MHz baudrate. Communicate with controller via CANopen protocol		
Mechanical dimensions	100*95*60mm(without brake)	130*95*60mm(without brake)	140*115*80mm(without brake)
note1：□=A：Incremental Differential 5V Encoder =M：16 bit single-turn magnetoelectric encoder note2：■=A：without brake =B：brake note3：★=L：Communication portRS232、RS485 =C：Communication portRS232、CANopen =E：Communication portRS232、EtherCAT			

1.2 Product description

1.1.1 MD naming rule

Product model naming rules											
name MD：integrated servo motor	MD	60-	040	-	D	M	A	K	-	CA	-	000
Flange size 60：60*60(mm) 80：80*80(mm)	MD	60-	040	-	D	M	A	K	-	CA	-	000
power 020：200W 040：400W 075：750W	MD	60-	040	-	D	M	A	K	-	CA	-	000
Main supply voltage B：DC24V C：DC36V D：DC48V L：AC220V H：AC380V	MD	60-	040	-	D	M	A	K	-	CA	-	000
Encoder type M：16 bit Magnetoelectric encoder A：2500PPR incremental encoder	MD	60-	040	-	D	M	A	K	-	CA	-	000
brake A：without brake B：brake	MD	60-	040	-	D	M	A	K	-	CA	-	000
Out of axis style A：Axis without keyway K：With key	MD	60-	040	-	D	M	A	K	-	CA	-	000
Control mode LA：RS232，RS485 CA：RS232，CANopen EA：RS232，EtherCAT	MD	60-	040	-	D	M	A	K	-	CA	-	000
Version code 000：Standard version	MD	60-	040	-	D	M	A	K	-	CA	-	000

1.1.2 Nameplate description



Chapter 2 System installment requirements and precautions

2.1 Installation of integrated servo motor

- Please ensure this menu can be provided for design engineer, operators and staffs (or machine) who is responsible to adjust and use this product
- Please ensure to follow requirements of this file all the time. And consider other accessory and module's file

  Please consider destination's law, and:

  —regulations and standards

  —test organization and insurance company's regulation

  —national specifications

2.1.1 ransportation and saving conditions

- Please ensure product do not overburn during the process of transportation and saving, including:
  —Mechanical load
  —non-allowed temperature
  —Water
  —Corrosive gas

- Please use original package to save and transport. Original package provide efficient protection so as to avoid influence of general issues

2.1.2 echnology requirements

Must follow:

- Specified connection and environment condition in product technology data and all of other connecting accessory's technology requirements. As long as product specification requirements are conformed, users are allowed to operate according to related safety regulations.
- Please follow instructions and alerts in this product

2.1.3 Operator’s requirements

- This product must be operated by electrical engineers who are familiar with instructions below：

—Electrical control system's installation and operation

—Regulations of operating safety project system

—Regulations of accident protection and occupation safety

—Product using menu

2.1.4 Environment requirements

Environment	Requirement
Working temperature	0 - 40℃ (no ice)
Working humidity	5 - 95%RH (no condensation)
Storage temperature	-10 - 70℃ (no ice)
Storage humidity	5 - 95%RH (no condensation)
Assembly requirement	Indoors without sunlight, corrosive gas, non-flammable gas, no dust.
Altitude	Less than 2000 m, power derating between 1000m and 2000m
Vibration	Less than 5.9m/s2, 10～60Hz (not to be used at the resonance point)
Protection level	IP20

2.1.5 Precautions

Item	Description
Stain proofing	Please wipe anti-rust agent on the motor's shaft and then make some anti-rust treatments.
Installation method	Improper installation method will cause damage of motor's encoder. Please note the following during the installation process: ● When operators installation pulleys on the servo motor shaft with key, it is necessary to use screw hole. In order to install pulleys, operators need to insert double-headed nail into screw holes and use washers on the surface of coupled end. Then use nuts to fix into pulleys gradually. ● For servo motor shaft with keys, Operator need to use screw hole on the shaft to install. For motors shaft with no key, operators need to use friction coupling or other analogous methods. ● When operators need to disassemble pulleys, operators need to use pulley remover so as to make shaft avoid strong impact of load. ● In order to make it more safe, it is necessary to install protection cover or some analogous equipment in rotation area. For example, pulleys installed on the shaft.
Centering	●When it is connected with machine, please use coupling and make shaft center of servo motor and machine stay in a line. When operators install servo motors, please achieve requirements of centering accuracy. If centering is not accurate, there will be shock and sometimes it will make bearings and encoders.
Installation direction	● Servo motors can be installed in vertical or horizontal direction.
Oil & water solution	When it is used in the occasion with drops, please use after make sure protection level of servo. When oil will drop into shaft penetrating part (beside shaft penetrating part, please choose servo motors with oil seal. The using condition of servo motors with oil seal: ● Make sure the oil level is lower than month of oil seal. ● Please use when oil seal make sure that oil splash degree is good. ● When servo motors are installed in vertical upward direction, please avoid oil accumulating in the month of oil seal.

2.1.6 Installing oil seal

The bearing of the motor has double flour dustproof effect. Assembling oil seal will increase t

he loss of the motor and lead to the decrease of motor efficiency. If it is not necessary to install oi l seal, it is not recommended to install oil seal. Before assembling the oil seal, please ensure that t he mounting hole groove and the oil seal are free of debris, oil stain, dust, etc. During assembly, p lease fill the oil seal The Lips with high-temperature grease (Greatwall HR12 and grease with temp erature resistance of 150 degrees are recommended) so as to enhance the lubrication and temper ature resistance performance and increase the sealing waterproof effect of the oil seal. When payi ng attention to water and oil prevention, the oil seal is installed outward from the side of the self-tightening spring (i.e. the side with groove). Please refer to the following steps to correctly install the oil seal.

1. Apply high-temperature lubricating grease evenly on the sealing ring of the oil seal lip.
2.Turn the side of the oil seal with groove outward to ensure that the oil seal is perpendicular to the machine shaft, and push the oil seal into the cavity with uniform force application.
3.After successfully installing, check whether the oil seal is inclined. The oil seal needs to be at tached to the motor bearing cover. The lip of the oil seal needs to be completely closed to en sure the tightness of the oil seal.

note

- Please operate and install the servo system strictly in accordance with the requirements of this manual. it can help you to set up and operate the driver correctly and make the driver perform optimally.

2.2 Installation dimension drawing

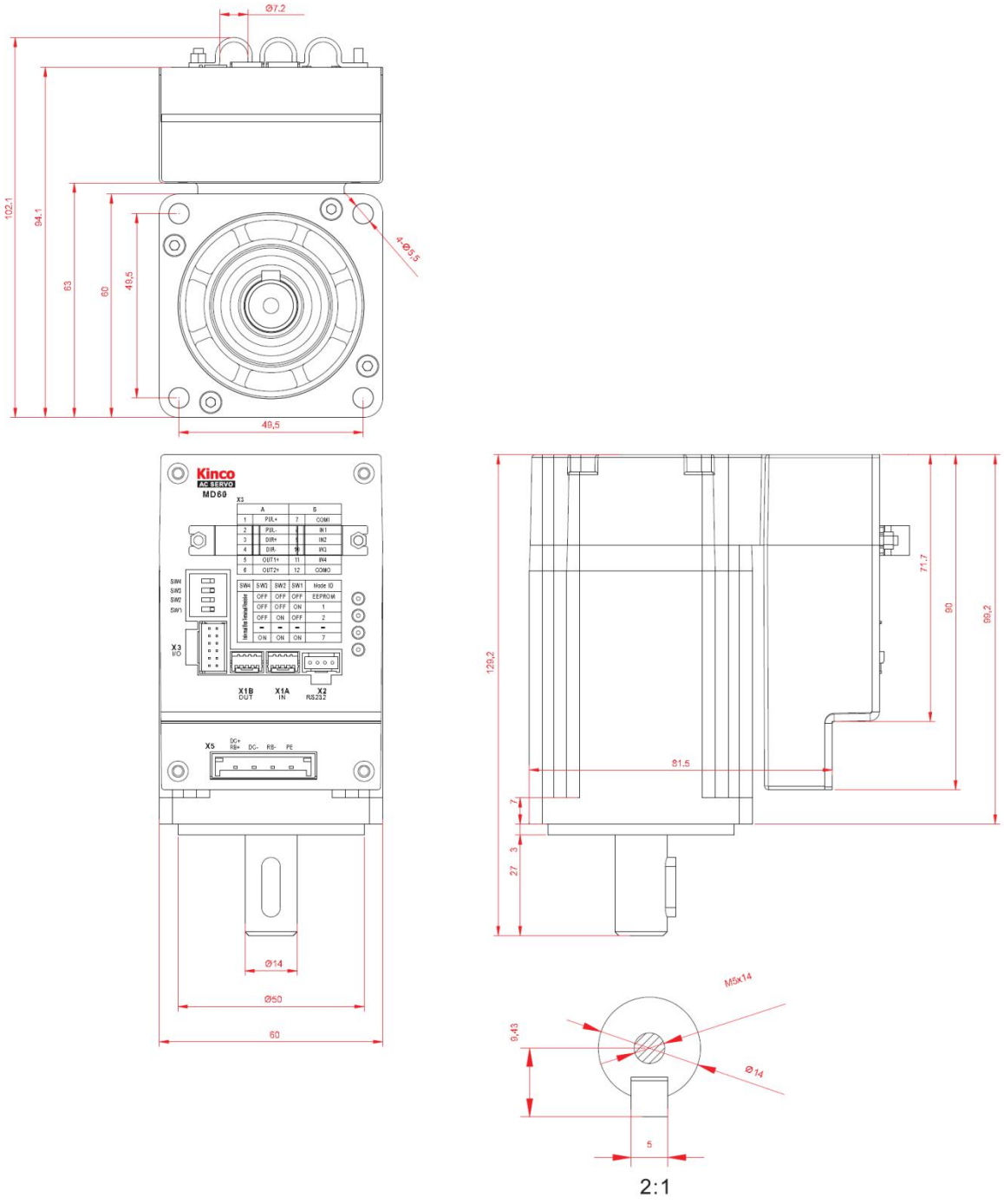


Figure 2-1 installation dimension diagram of MD60 (200 W)

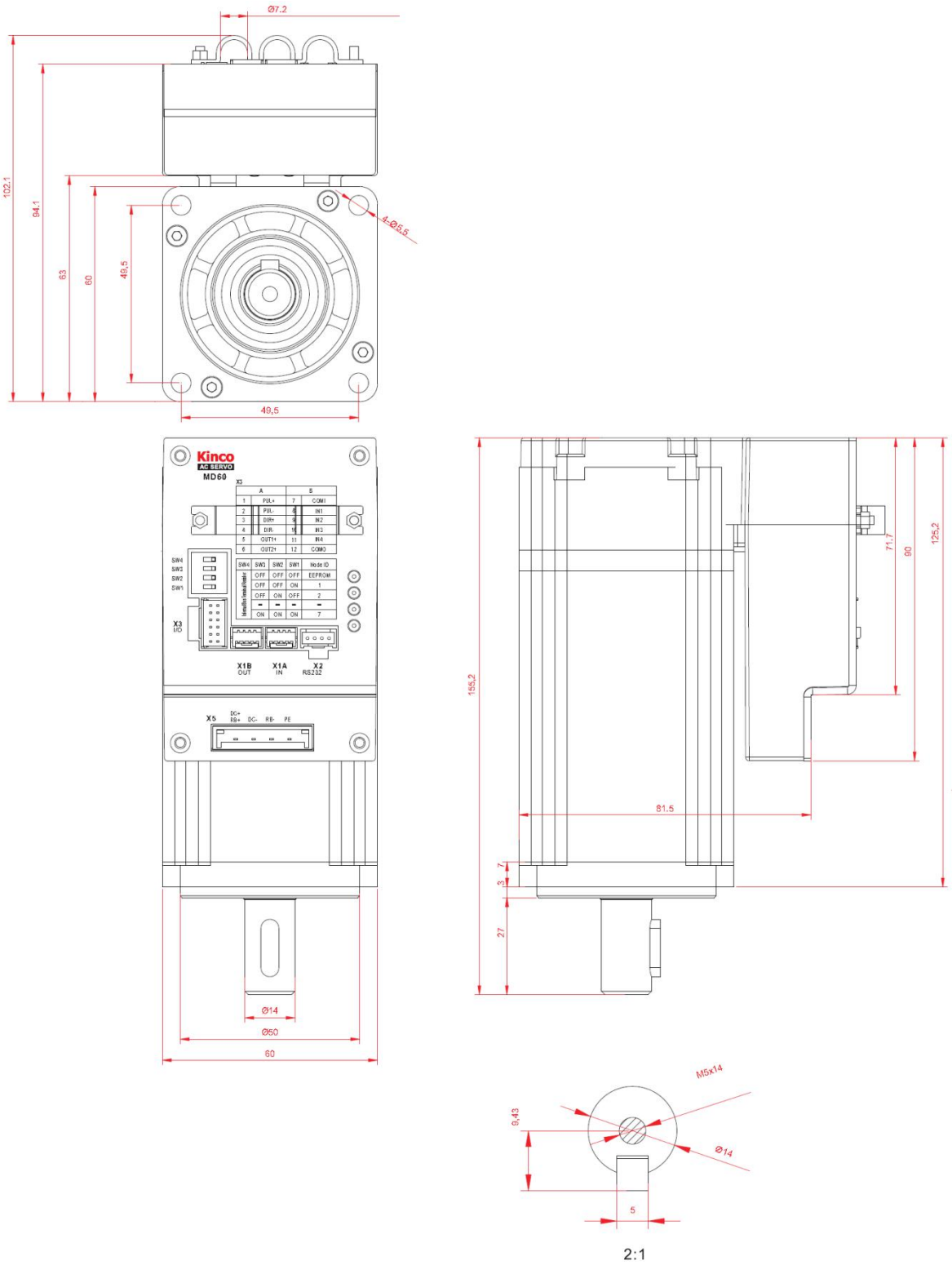


Figure 2-2 Installation Dimension Drawing of MD60 (400W)

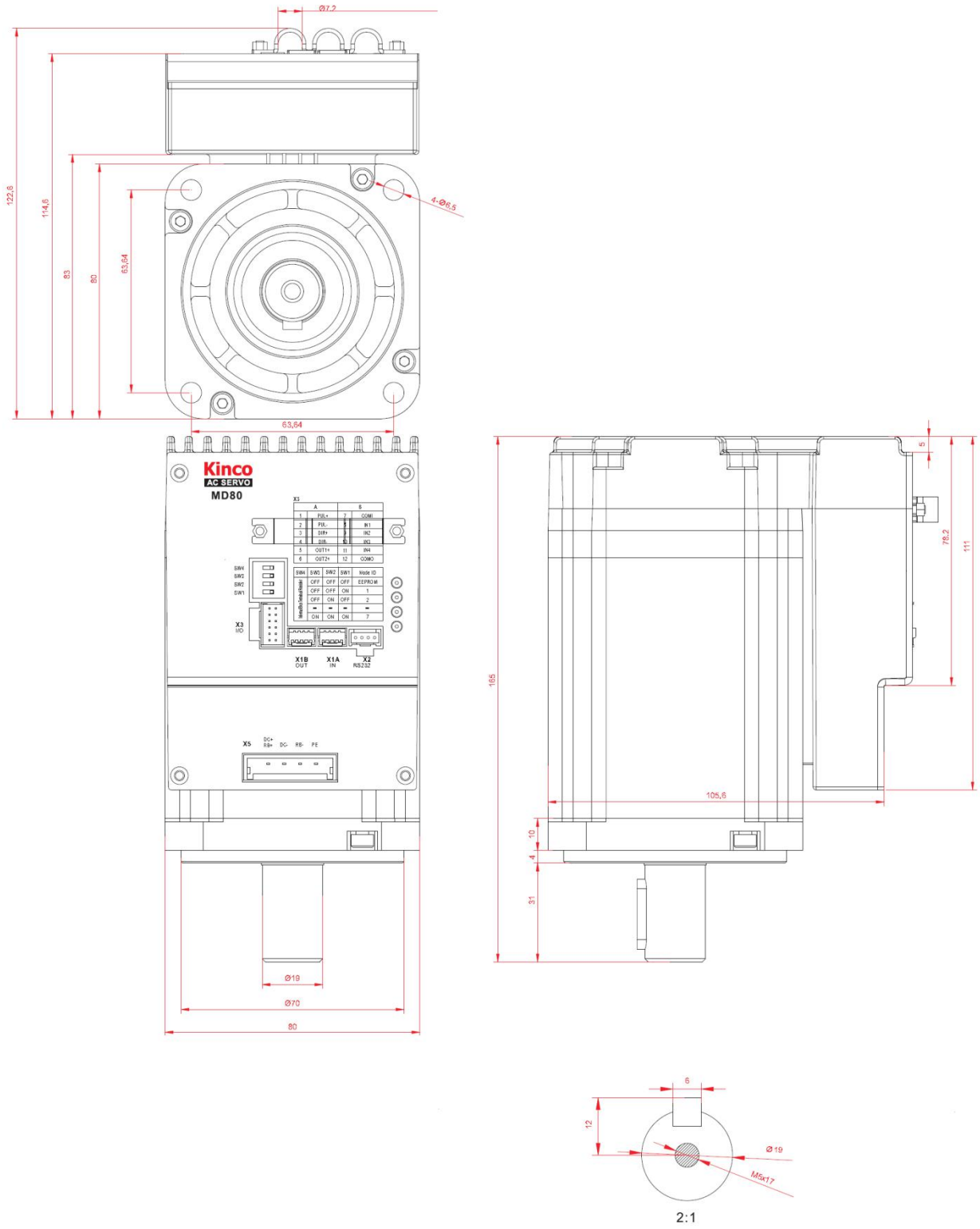


Figure 2-3 installation dimension Diagram of MD80 (750 W)

2.3 Servo motor torque-speed curve

2.3.1 200W servo motor torque-speed curve

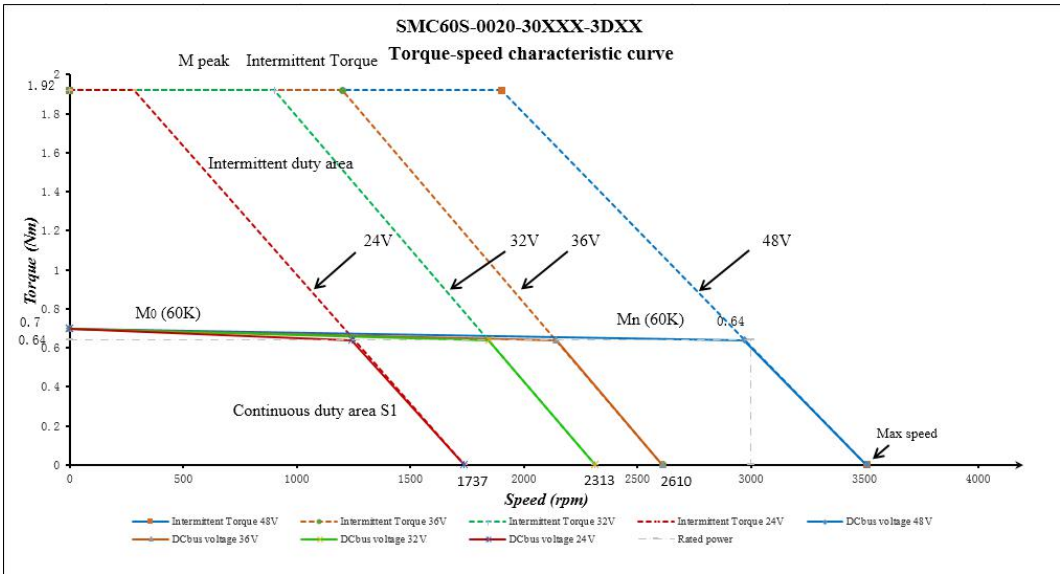


Figure 2-4 200W Motor Curve

2.3.2 400W servo motor torque-speed curve

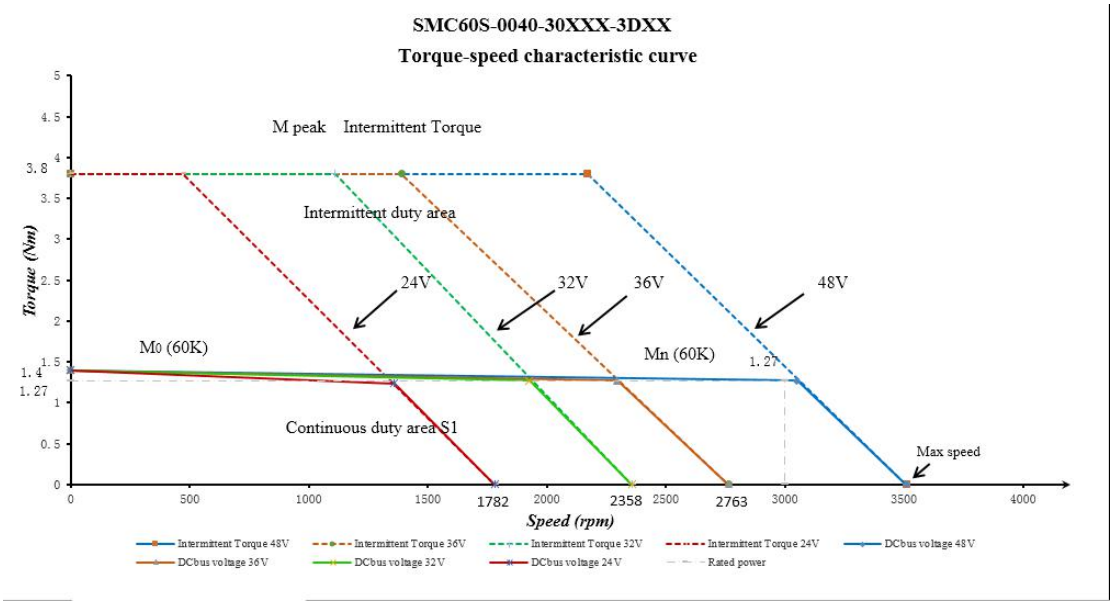


Figure 2-5 400W Motor Curve

2.3.3 750W servo motor torque curve

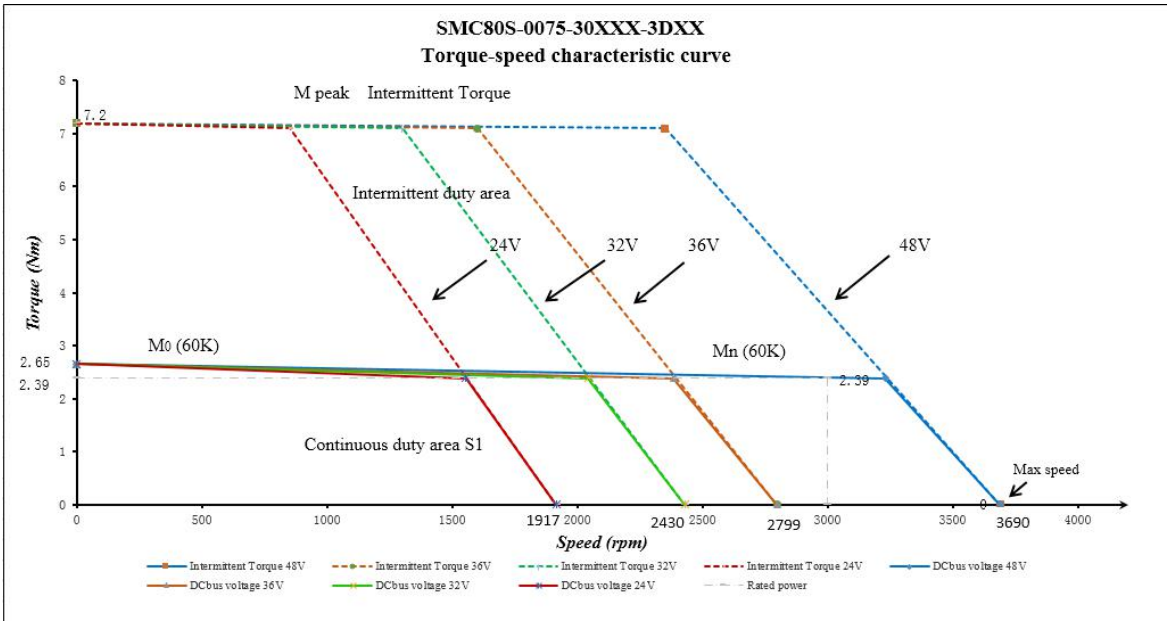


Figure 2-6 750W Motor Curve

Chapter 3 Interface and wiring

3.1 integrated servo motor components name

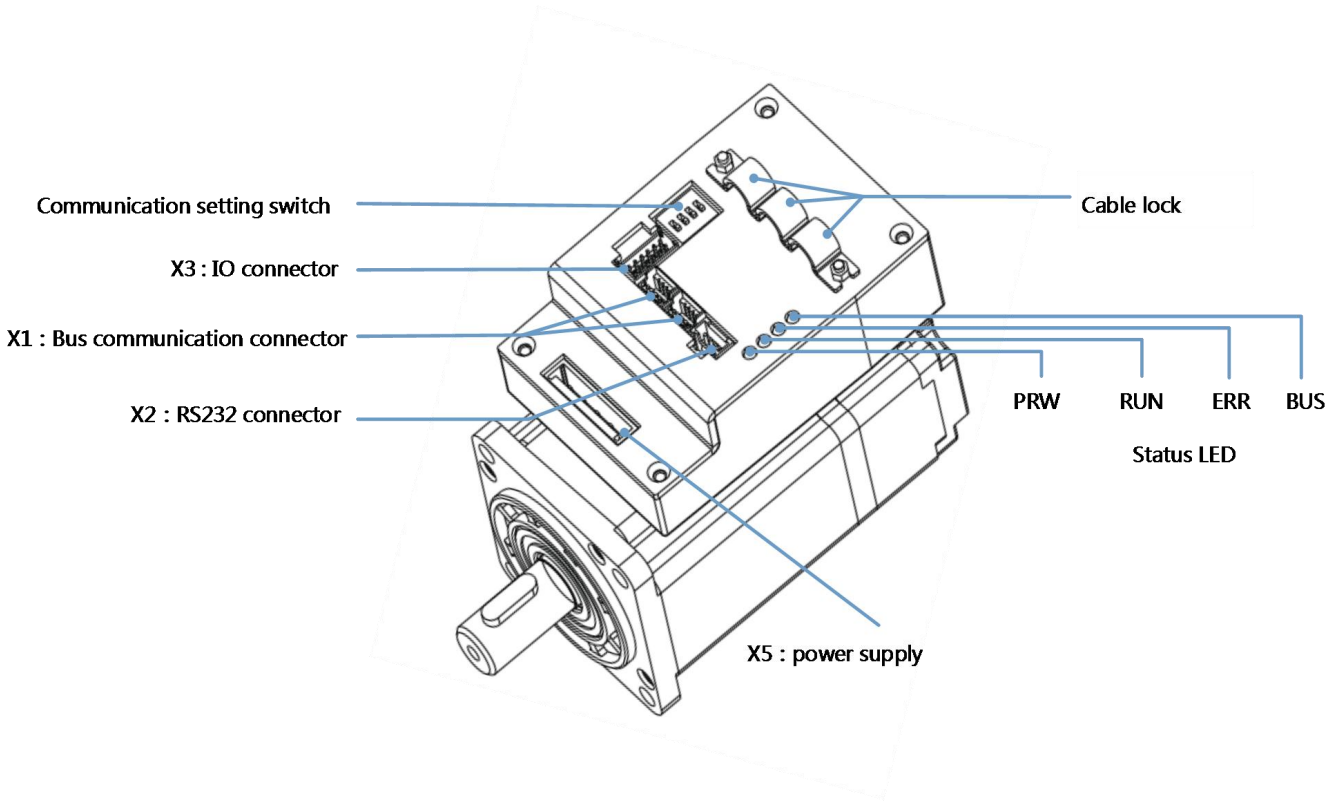


Figure 3−1 MD Series Interface Definition

3.2 External wiring

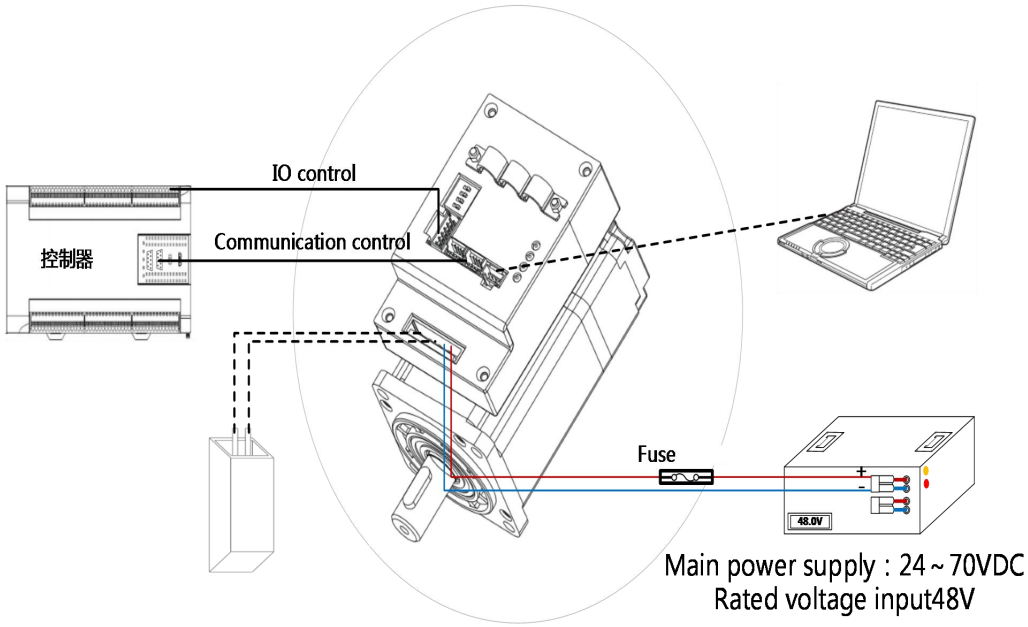


Figure 3−2MD seriesexternal wiring diagram

Note:

- The interface definition for different power products in MD series is the same, with MD60(200W) as an example in figure. 3-1 and figure 3-2
- Fuses should be selected according to table 3-1. please refer to the appendix for brake resistance specification recommendations.

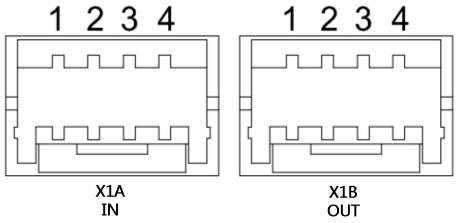
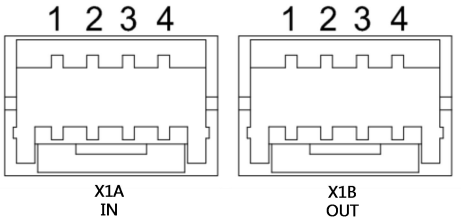
Table 3-1 Fuse specification recommendation

Integrated servo motor models	power(W)	Fuse reference specification
MD60-020-DMAK-□A-000 MD60-020-DMBK-□A-000	200	20A/58VDC
MD60-040-DMAK-□A-000 MD60-040-DMBK-□A-000	400	20A/58VDC
MD80-075-DMAK-□A-000 MD80-075-DMBK-□A-000	750	40A/58VDC

3.3 Interface description

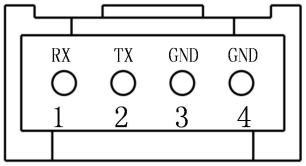
3.3.1 Bus communication interface（X1）

Table 3-2 X1Interface definition

<table>
<tr><td>Bus Type</td><td colspan="2">CANopen</td><td colspan="2">RS485</td></tr>
<tr><td>Applicable Products</td><td colspan="2">MD60-020-DMAK-CA-000
MD60-040-DMAK-CA-000
MD80-075-DMAK-CA-000
MD60-020-DMBK-CA-000
MD60-040-DMBK-CA-000
MD80-075-DMBK-CA-000</td><td colspan="2">MD60-020-DMAK-LA-000
MD60-040-DMAK-LA-000
MD80-075-DMAK-LA-000
MD60-020-DMBK-LA-000
MD60-040-DMBK-LA-000
MD80-075-DMBK-LA-000</td></tr>
<tr><td rowspan="6">Pin Definition</td><td colspan="2"> 
</td><td colspan="2"> 
</td></tr>
<tr><td>Pin No.</td><td>Pin name</td><td>Pin No.</td><td>Pin name</td></tr>
<tr><td>1</td><td>GND</td><td>1</td><td>GND</td></tr>
<tr><td>2</td><td>GND</td><td>2</td><td>GND</td></tr>
<tr><td>3</td><td>CAN_L</td><td>3</td><td>485+</td></tr>
<tr><td>4</td><td>CAN_H</td><td>4</td><td>485-</td></tr>
</table>

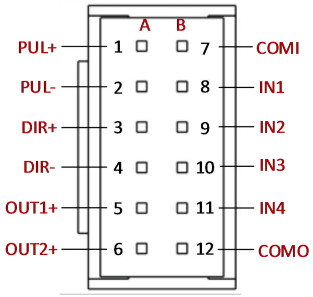
3.3.2 RS232 port（X2）

Table 3-3 X2Interface definition

<table>
<tr><td rowspan="5"> 
</td><td>Pin No.</td><td>Pin name</td><td>Pin function</td></tr>
<tr><td>1</td><td>RX</td><td>The driver receives data</td></tr>
<tr><td>2</td><td>TX</td><td>The driver sends data</td></tr>
<tr><td>3</td><td>GND</td><td>Signal ground</td></tr>
<tr><td>4</td><td>GND</td><td>Signal ground</td></tr>
</table>

3.3.3 External input&output（X3）

Table 3-4 X3Interface definition



Pin No.	Pin name	Pin function
1	PUL+	Pulse input function Input voltage ：3.3V～24V Maximum frequency ：500KHz
2	PUL-	
3	DIR+	
4	DIR-	
5	OUT1+	Digital signal output Maximum output current ：100mA
6	OUT2+	
7	COMI	Input common terminal
8	IN1	Digital signal input high level ：12.5VDC~30VDC Low level ：0VDC~5VDC Input frequency ：<1KHz
9	IN2	
10	IN3	
11	IN4	
12	COMO	Output common

note

- Please refer to appendix 4 for wire gauges and wire making methods of X1, X2 and X3 terminals.
- The above figure shows the definition of the interface on the driver side, not the communication cable. Please be careful to avoid the wrong welding line.

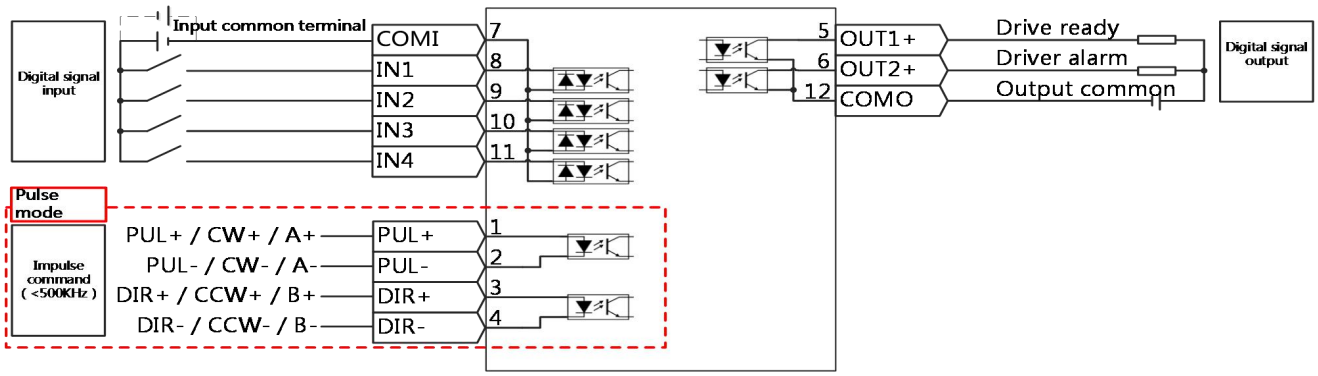
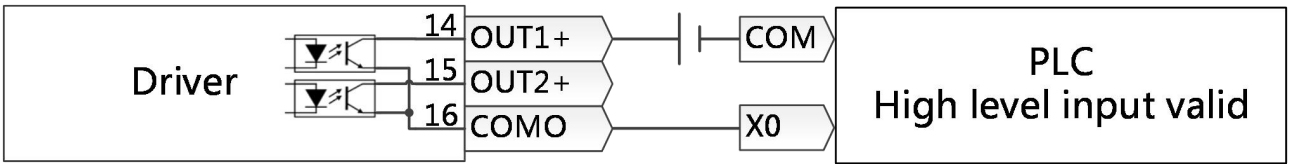


Figure3−3MD seriescontrol wiring diagram

note

- Figure 3-5 shows the wiring with default IO function. More IO functions can be defined by the Kinco servo software. For more details on IO functions, please refer to the relevant sections.
- For digital output, Figure 3-2 only shows NPN connection, and Figure 3-3 shows PNP connection



PFigure3−4 PNP input wiring

3.3.4 power interface（X5）

表 3-5 X5 接口定义

Pin name	Pin function
DC-	DC power input（24-70V）
DC+	
RB-	External brake resistor
RB+	
PE	ground

Table 3-6 Specification for power cable

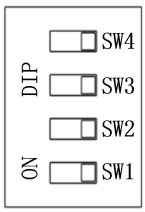
model	X5 Power interface wiring specifications
MD60 （200W）	Crimp terminal wiring specification range：0.21~1.31mm² （24~16AWG） Recommended conductor cross-sectional area：0.75~1.31mm² （18~16AWG） Recommended stripping cable length：8~9mm
MD60 （400W）	Crimp terminal wiring specification range：0.21~1.31mm² （24~16AWG） Recommended conductor cross-sectional area：1.3~1.5mm² （16~15AWG） Recommended stripping cable length：8~9mm
MD80 （750W）	Crimp terminal wiring specification range：0.21~3.3mm² （24~12AWG） Recommended conductor cross-sectional area：2.5~3.3mm² （13~12AWG） Recommended stripping cable length：12~13mm

note

- Please use the power cable with shielding layer, twist and fold the wire core into bundles and insert it into the crimping terminal. After the cable is connected, pull the cable to confirm that the wire is firmly connected with the terminal, and there is no flying wire or touching wire on the adjacent cable.
- The bending radius of the cable shall be more than 10 times of the outer diameter of the cable itself, and frequent bending of the cable shall be avoided.

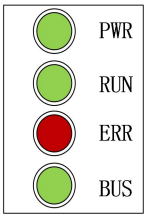
3.3.5 Dip switch and Indicators

Table 3-7 dip switch

	Pin name	Pin function
	SW1	The equipment station number is determined by the BCD code composed of SW1-SW3. Restart of the driver takes effect after dip switch changes. When SW1-SW3 are all OFF, the driver reads the equipment station number in EEPROM.
	SW2	
	SW3	
	SW4	When SW4 is ON, turn on the terminal resistance

note
Integrated servo system factory default SW1 is ON, other dial codes are OFF

Table 3-8 Indicator light

	Pin name	Pin function
	PWR	The driver has been POWERed on, and the power lamp is always on.
	RUN	When the drive is ready, it is always on and associated with out3.
	ERR	When the driver reports an error, it is in a normally bright state and is associated with out4.
	BUS	When there is message transmission on CANopen bus, it will flash, and the flashing frequency is related to the message transmission speed.

note

- In the software, out3 defines drive readiness by default and out4 defines drive failure by default. When the RUN and ERR lights do not illuminate, please check whether the default definition has been modified.

Chapter 4 KincoServo software introduction

This chapter will introduce how to use KincoServo software adjust and configure servo driver.

Figure0–1 Software main window

4.1 Fast start

4.1.1 Language configuration

Language can be switched between English and Chinese via menu item **Tools**->**Language**.

4.1.2 Opening and saving project files

Create a new project file via menu item **File**->**New**, or by clicking the button.

Open an existing project via menu item **File**->**Open**, or by clicking the button and selecting a .kpjt file.

Save a project via menu item **File**->**Save**, or by clicking the button and saving as a .kpjt file.

Note

Only the windows (object list, scope etc.) are saved-parameters in the controller can’ t be saved in this way.

4.1.3 Start communication

Click menu item **Communication**->**Communication settings**. The following window appears:

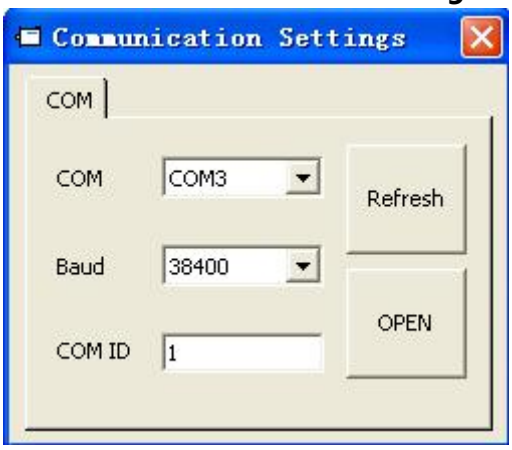


Figure0–2 Communication setting

Select the right COM port (if it’ s not shown click the “Refresh” button), baud rate and COM ID (Node ID), and then click the "OPEN” button.

Once communication has been established with the controller, communication can be opened or closed by clicking the button.

4.1.4 Node ID and baud rate

If more than one controller is being used in an application, you may need different node ID for different controllers in order to distinguish among them.

The controller’ s Node ID can be changed via menu item **Controller**->**Controller Property**.

Internal address	Type	Name	Value	Unit
100B0008	Usigned8	Node ID		DEC
2FE00010	Usigned16	RS232 baud rate		Baud

Note

Node ID and baud rate setting are not activated until after saving and rebooting.

4.1.5 Object (add , delete , help)

Open any window with an object list, move the mouse pointer to the object item and right click. The following selection window appears:

5	606000	int8	Operation_Mode	
6	604000	uint16	Controlword	
7	607A00	int32	Target_Position	
8	608100	uint32	Profile_Speed	
9	608300	uint32	Profile_Acc	
10	608400	uint32	Profile_Dec	

Add
Delete
Help

Click **Add** and double click the required object from the **Object Dictionary**. The selected object is then added to the list.

Click **Delete**. The selected object is removed from the list.

Click **Help** to read a description of the selected object in the **Object Dictionary**.

4.2 Initialize, save and reboot

Click **Controller**->**Init Save Reboot**. The following window appears:

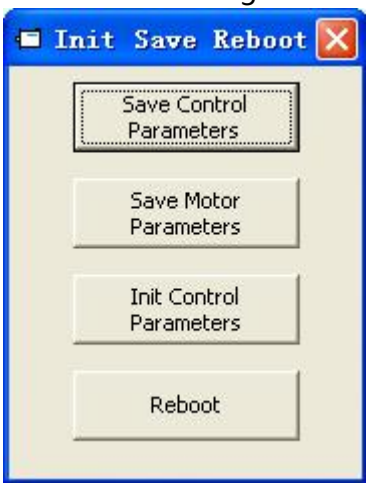


Figure0−3 Initialize, save, reboot

Click the corresponding item to finish the necessary operation.

Note

After completing the **Init Control Parameters**, the Save Control Parameters and Reboot buttons must be clicked to load the default control parameters to the controller.

4.3 Firmware update

A new motor controller is always delivered with the latest firmware version. If the firmware needs to be updated for any reason, load the new firmware via menu item **Controller**->**Load Firmware**.

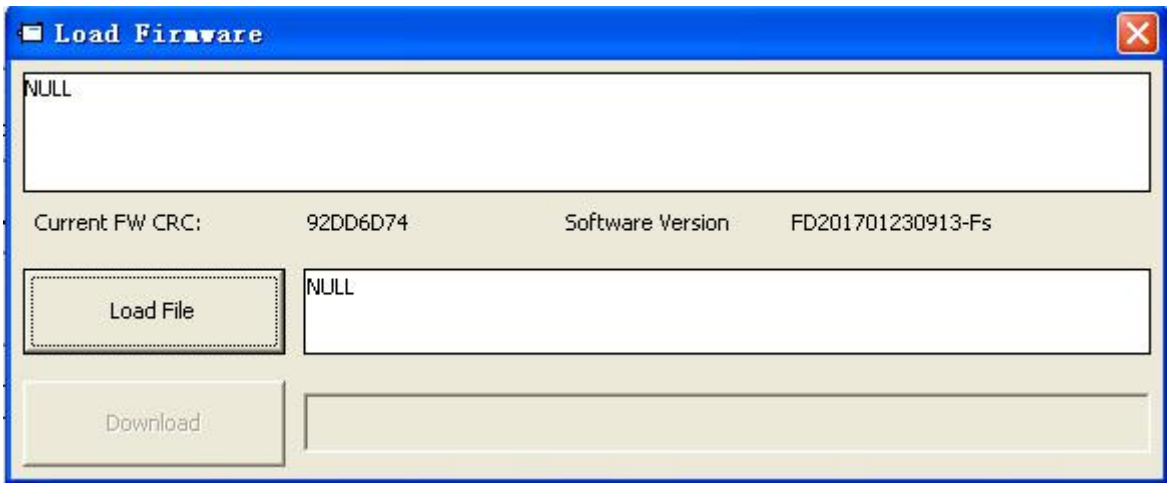


Figure0–4 Load Firmware

Click **Load File** to select the firmware file (.servo) and then click **Download** to start loading firmware to the controller.

➔ **Note**

Do not switch off the power or disconnect the RS232 cable during firmware loading. If the download process is interrupted, first reset controller power. Then select the firmware file and click the Download button, and finally start RS232 communication.

4.4 Read/write controller configuration

This function can be used to read / write multiple parameters simultaneously for large production lots, in order to avoid setting the controller parameters one by one.

4.4.1 Read setting from controller

Click **Tools**->**R/W Controller Configuration**->**Read Settings** from Controller or click the button. The following window appears.

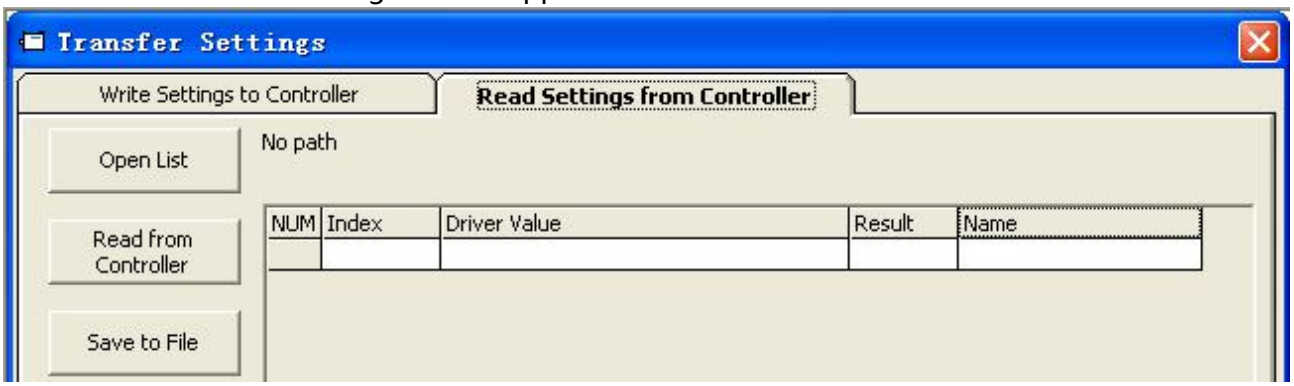


Figure0–5 Read driver configuration

Click **Open List** to select a parameter list file (.cdo). The parameter appears in the window. Click **Read Settings from Controller** to get the **Drive Value** and **Result**, and then click **Save to File** to save the settings as a .cdi file.

Note

The .cdo file defines which objects will be read out, but if the object doesn't exist in the controller, the result will be "False" (displayed in red).

4.4.2 Write settings to controller

Click **Tools**->**R/W Controller Configuration**->**Write Settings to Controller** or click the button.

The following window appears:

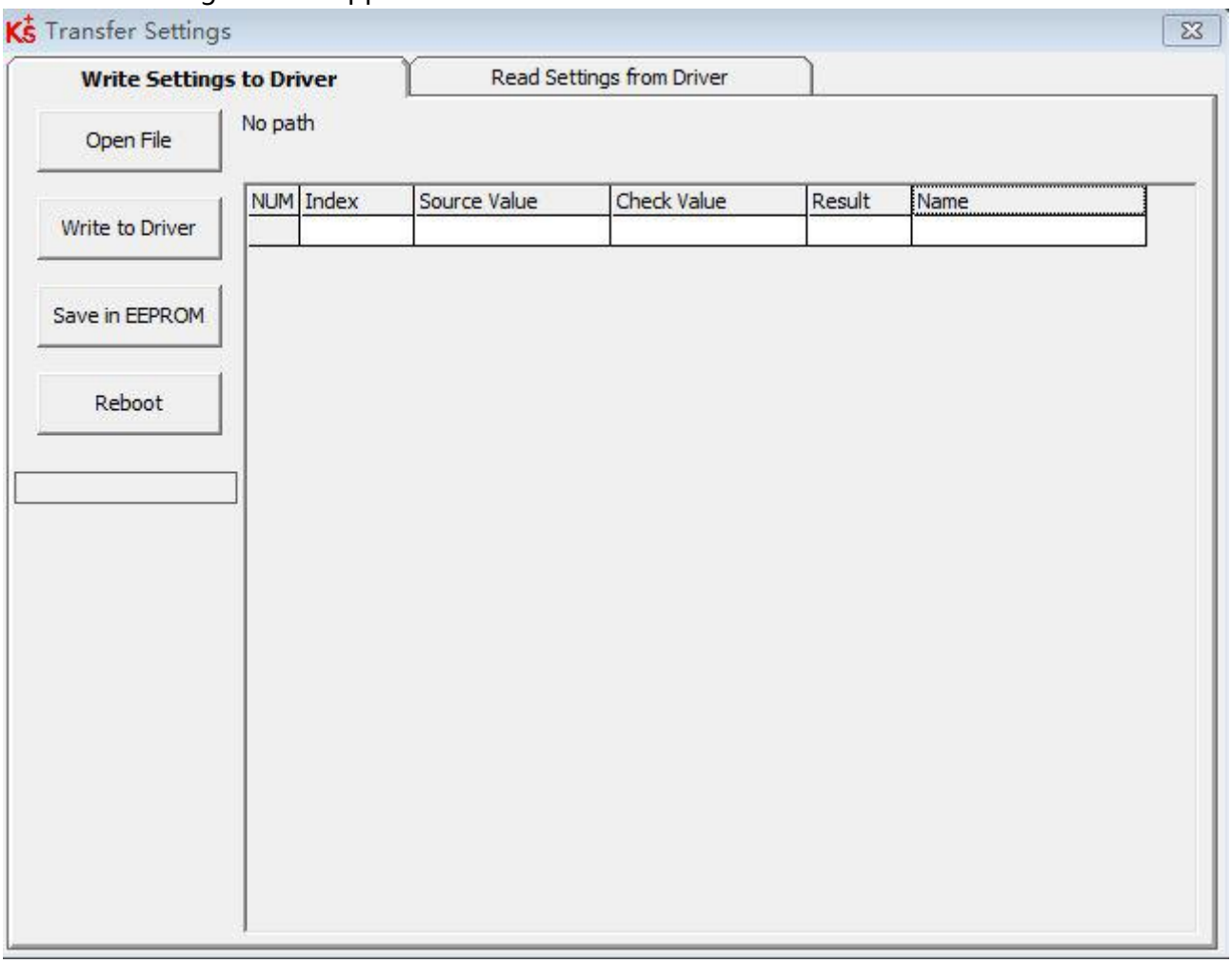


Figure0–6 Write driver configuration

Click **Open File** to select a parameter settings file (.cdi). The parameter settings appear in the window.

Click **Write to Controller** to get the **Check Value** and **Result**. The "False" **Result** means the value has not been written successfully, probably because the object doesn't exist in the controller.

Click **Save in EEPROM** and **Reboot** to activate all parameters.

Note

Before write setting to driver, please cancel driver enable. If driver is enabled, some object cannot be written.

4.5 Digital IO functions

Click menu item **Controller**->**Digital IO Functions** or click the I-O button. The following window appears. Function and polarity are shown as defaults here.

Note

FD1X3 support 4 road digital inputs (Din1, Din2, Din3, Din4) and 2 road digital outputs (Dout1, Dout2).

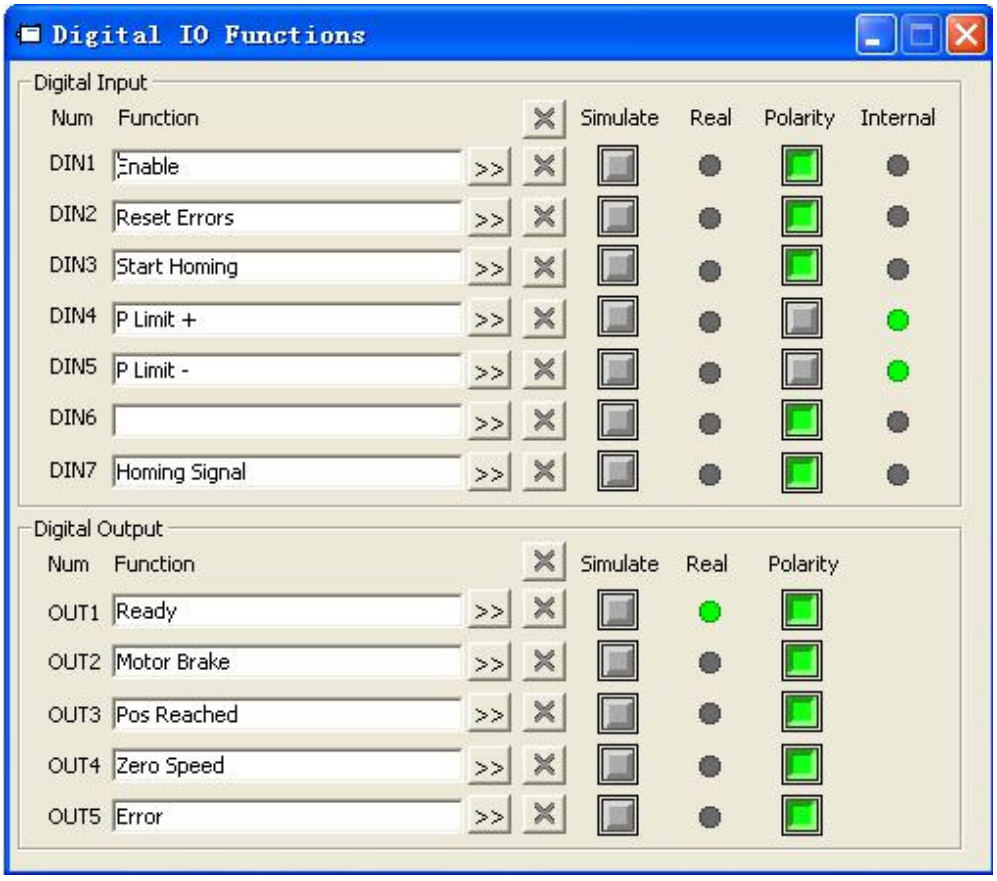


Figure 0−7 Digital input output

4.5.1 Digital input

Figure0–8 Digital input

Function: Click >> to select DIN function setting, click ✕ to delete the DIN function setting.

Simulate: Simulates the digital input active hardware signal.

Real: Shows the real digital input hardware status.

Polarity : means Internal is set to 1 by "active" signal. means Internal is set to 1 by "inactive" signal.

Internal : This is the result of Simulate, Real and Polarity via the logic formula:

Internal=(Real OR Simulate) XOR (NOT Polarity)

means "active", logic status of the selected function is 1 ; means "inactive", logic status of the selected function is 0.

DIN function	Description
Enable	Controller enabling 1: Enable controller (Controlword=Din_Controlword(2020.0F) , default value=0x2F) 0: Disable controller (Controlword = 0x06)
Reset errors	Sets the Controlword to reset errors, active edge: 0 -> 1
Operation mode sel	Operation_Mode selection 1: Operation_Mode=EL.Din_Mode1 (2020.0E), default value = -3 0: Operation_Mode=EL.Din_Mode0 (2020.0D), default value = -4
Kvi off	Velocity control loop integrating gain off
Limit+	Positive / negative position limit switch input for "normally closed" limit switches 0: position limit is active, the related direction is blocked
Limit-	
Home signal	Home switch signal, for homing
Invert Direction	Inverts command direction in the velocity and torque mode
Din Vel Index 0	Din_Speed Index in the DIN speed mode
Din Vel Index 1	
Din Vel Index 2	
Quick stop	Sets the controlword to start quick stop. After quick stop, the controlword needs to be set to 0x06 before 0x0F for enabling (if the enable function is configured in Din, just re-enable it)

Start homing	Starts homing. Only makes sense if the controller is enabled. The controller returns to the previous operation mode after homing.
Activate command	Activates the position command. Controls bit 4 of the Controlword, e.g. Controlword=0x2F->0x3F
Pre enable	For safety reasons, Pre_Enable can serve as a signal for indicating whether or not the entire system is ready. 1: controller can be enabled 0: controller can not be enabled

Note

Relative/Absolute position control select (2020.0F) default setting is 0x2F. For Control word definition, please refer to Chapter 6.1.

4.5.2 Digital output

Figure0–9 Digital output

Function: Click >> to select the OUT function setting. Click to delete the OUT function setting

Simulate: Simulates the digital output function logic status 1.

Real: Shows the real digital input hardware status. This is the result of Simulate, Polarity and Logic State, means that digital input is ON, means that digital input is OFF.

Polarity: Inverts the logic status of the digital output function.

1 means **Real** physical digital output is set to ON by digital output function logic status 1

0 means **Real** physical digital output is set to ON by digital output function logic status 0

Real: This is the result of Simulate, Polarity and real input.

activate, logic state of corresponding function is 1.

deactivate, logic state of corresponding function is 0.

OUT function	Description
Ready	Controller is ready to be enabled
Error	Controller error
Pos Reached	Under position mode, position difference between Pos_Actual and Pos_Target<Target_Pos_Window(6067.00),duration>=Position_Window_time(6068.00)

Zero Speed	|Speed_1ms(60F9.1A)|<=Zero_Speed_Window(2010.18) and duration >=Zero_Speed_Time(60F9.14)
Motor brake	Signal for controlling the motor brake. By this signal an external relay can be controlled, by which the motor brake is controlled. (see chapter 3.2.4).
Speed Reached	|Speed_Error(60F9.1C)|<Target_Speed_Window(60F9.0A)
Enc Index	Encoder position is inside a range around the index position. This range is defined by Index_Window(2030.00).
Speed Limit	In torque mode actual speed reached Max_Speed(607F.00)
Driver Enabled	Controller enabled
Position Limit	Position limit function is active
Home Found	Home found

4.6 Scope

During operation, if performance does not meet the requirement or any other unexpected behaviour occurs, it’ s highly advisable to use the scope function to do the analysis.

Click **Controller**-->**Scope** or click to open the scope window

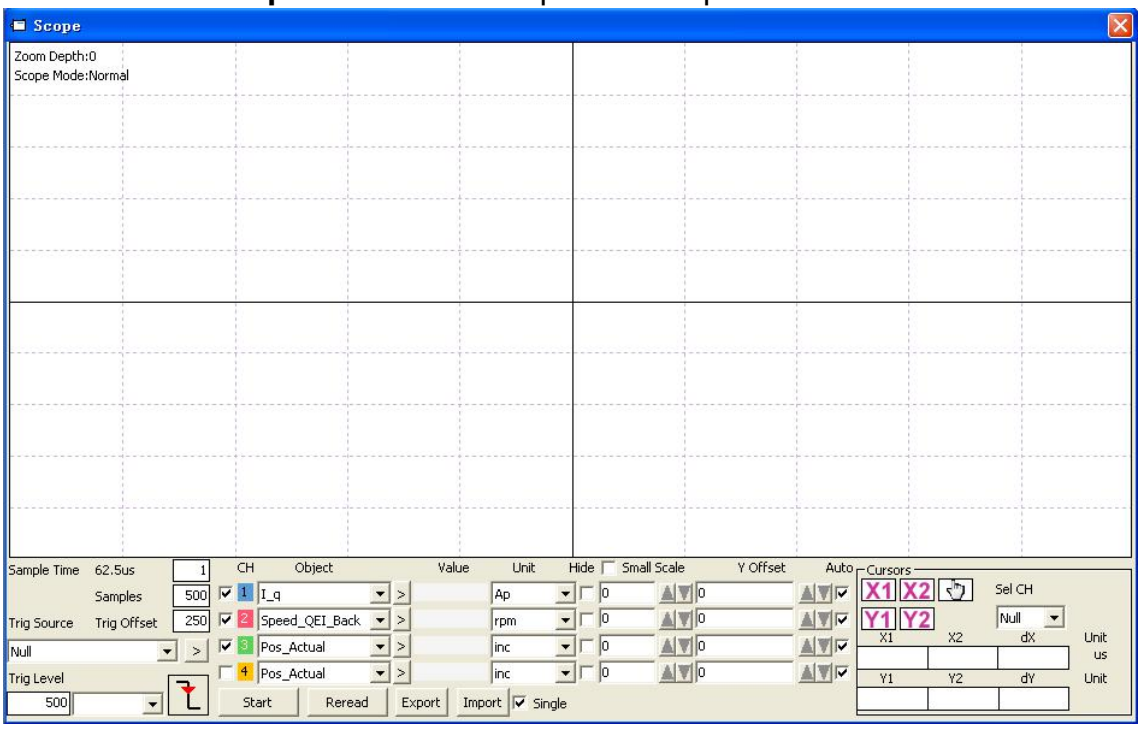


Figure 4-10 scope display

Sample time: The period of data collection, set to 1, means one data is collected every 62.5us.

Samples : Indicates how many data were collected in this sampling. Setting to 500 indicates that 500 data were collected.

Trig offset: Number of samples before the trigger event occurs.

Trigger source and trigger level: The trigger condition is set in fig. 4-10 to start data acquisition when the effective target current q rises to 100DEC, DEC is the internal unit and can be switched to current unit.

Trigger edge:The display  is a rising edge trigger and can be changed to a falling edge trigger after clicking.

Object: Maximum 64-bit length data can be taken in one sample, e.g.: 2 Int32 objects bit or 4 Int16 objects.

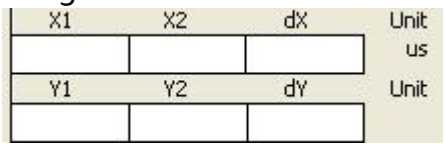
Single:  means sample for one trigger event only.  means sample continuously.

Zoom in / zoom out the oscillogram: Hold down the right mouse button, drag the mouse to the lower right to enlarge the oscillogram, and drag the mouse to the upper left to reduce the oscillogram.

Cursors: Up to 4 scope cursors can be selected by clicking the respective button:  . The scope cursors appear in the oscillogram. Select a channel in the **Sel CH** list box.

Move cursor:

Move the mouse pointer to the scope cursor. Press left mouse button and drag the scope cursor to move it. A sample value and the differences of X1, X2 and Y1, Y2 appear in the following fields:



Export: Export sampled data to a. scope file

Import: Import the. scope file and display the oscillogram.

Rereading data: Read out the recently collected data from the driver and display the oscillogram.

Automatic: If the automatic option box is checked, the oscillograph will automatically select the appropriate scale and axis offset for display. if the automatic option box is not checked, the oscillograph will display 

according to the scale and offset in the following areas. The scale and offset values can be increased or decreased by clicking buttons  and  . If the small scale option box is

checked, the scale increase/decrease corresponding to each button will be 10% of the original value.

Oscilloscope mode: At the top left of the oscilloscope, it will be shown that the oscilloscope mode is normal or imported.

___Normal: All buttons of oscilloscope are available

___Import: oscillogram is imported from. scope file. in this mode, the start and reread data buttons are disabled, and you can exit the import mode according to the software prompt.

4.7 Error display and error history

Error: Click **Controller**->**Error Display** or click the  button (which turns red  if an error occurs). The **Error Display** window appears. It shows the last errors.

Error History: Click menu item **Controller**->**Error History**. The error history list window appears. It shows the last 8 errors' Error codes and respective the related DCBUS voltage, speed, current, controller temperature, Operation_Mode, and controller working time at the moment when the error occurred.

Error_state information:

Bit	Error name	Error code	Description
0	Extended Error		Refer to object "Error_State 2" (2602.00)
1	Encoder not connected	0x7331	No communication encoder connected
2	Encoder internal	0x7320	Internal encoder error
3	Encoder CRC	0x7330	Communication with encoder disturbed
4	Controller Temperature	0x4210	Heatsink temperature too high
5	Overvoltage	0x3210	DC bus overvoltage
6	Undervoltage	0x3220	DC bus undervoltage
7	Overcurrent	0x2320	Power stage or motor short circuit
8	Chop Resistor	0x7110	Overload, brake chopper resistor
9	Following Error	0x8611	Max. following error exceeded
10	Low Logic Voltage	0x5112	Logic supply voltage too low
11	Motor or controller IIt	0x2350	Motor or power stage IIt error
12	Overfrequency	0x8A80	Pulse input frequency too high
13	Motor Temperature	0x4310	Motor temperature sensor alarm
14	Encoder information	0x7331	No encoder connected or no encoder communication reply
15	EEPROM data	0x6310	EEPROM checksum fault

Error_state2 information:

Bit	Error name	Error code	Description
0	Current sensor	0x5210	Current sensor signal offset or ripple too large
1	Watchdog	0x6010	Software watchdog exception
2	Wrong interrupt	0x6011	Invalid interrupt exception
3	MCU ID	0x7400	Wrong MCU type detected
4	Motor configuration	0x6320	No motor data in EEPROM / motor never configured
5	Reserved		
6	Reserved		
7	Reserved		
8	External enable	0x5443	DIN "pre_enable" function is configured, but the DIN is inactive when the controller is enabled / going to be enabled
9	Positive limit	0x5442	Positive position limit (after homing) – position limit only causes error when Limit_Function (2010.19) is set to 0.
10	Negative limit	0x5441	Negative position limit (after homing) position limit only causes error when Limit_Function(2010.19) is set to 0.
11	SPI internal	0x6012	Internal firmware error in SPI handling
12	Reserved		
13	Closed loop direction	0x8A81	Different direction between motor and position encoder in closed loop operation by a second encoder.
14	Reserved		
15	Master counting	0x7306	Master encoder counting error

note

- Click on the menu bar in the software interface “Help” -> “error code”, Error code description can be opened.

Chapter 5 Operation mode

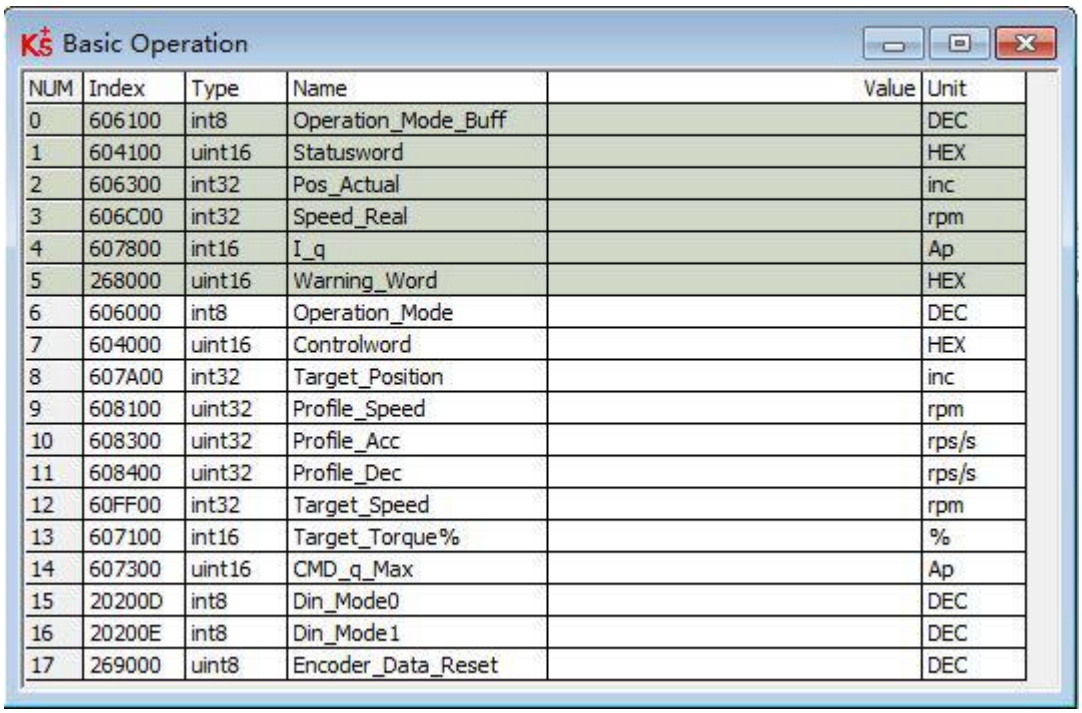
5.1 Velocity mode (-3, 3)

There are 2 kinds of velocity mode: -3 and 3. The velocity command can be specified via Target_Speed or analog input (analog speed mode), or via digital input (DIN speed mode).

Table0−1 Velocity mode

Internal address	Type	Name	Description	value
60600020	Integer8	Operation mode	-3: The velocity command is specified directly by Target_Speed. Only the velocity control loop is active. 3: The velocity command is specified by Target_Speed with profile acceleration and profile deceleration. Velocity- and position control loops are active	-3 and 3
60400010	Unsigned16	Control word	0x0F: Enable the controller ；0x06: Disable the controller	0x0F
60FF0020	Integer32	Target-speed	Target velocity, cannot over motor rated speed	User defined
60810020	Unsigned32	Profile_Acc	Active in mode 1 and 3	Default as 100rps/s
60830020	Unsigned.32	Profile_Dcc	Active in mode 1 and 3	Default as 100rps/s

In software "**Basic operation**" window, we can find these parameters and set, on the 6th, 7th, 10th, 11th, 12th, respectively.



NUM	Index	Type	Name	Value	Unit
0	606100	int8	Operation_Mode_Buff		DEC
1	604100	uint16	Statusword		HEX
2	606300	int32	Pos_Actual		inc
3	606C00	int32	Speed_Real		rpm
4	607800	int16	I_q		Ap
5	268000	uint16	Warning_Word		HEX
6	606000	int8	Operation_Mode		DEC
7	604000	uint16	Controlword		HEX
8	607A00	int32	Target_Position		inc
9	608100	uint32	Profile_Speed		rpm
10	608300	uint32	Profile_Acc		rps/s
11	608400	uint32	Profile_Dec		rps/s
12	60FF00	int32	Target_Speed		rpm
13	607100	int16	Target_Torque%		%
14	607300	uint16	CMD_q_Max		Ap
15	20200D	int8	Din_Mode0		DEC
16	20200E	int8	Din_Mode1		DEC
17	269000	uint8	Encoder_Data_Reset		DEC

Figure 5-1 “Basic operation” window

5.1.1 DIN speed mode introduction

First, when using Din speed mode, at least one of Din Vel index 0, Din Vel index 1 and DIN Vel index 2 must be defined in the I/O configuration as the switching signal of the speed segment.

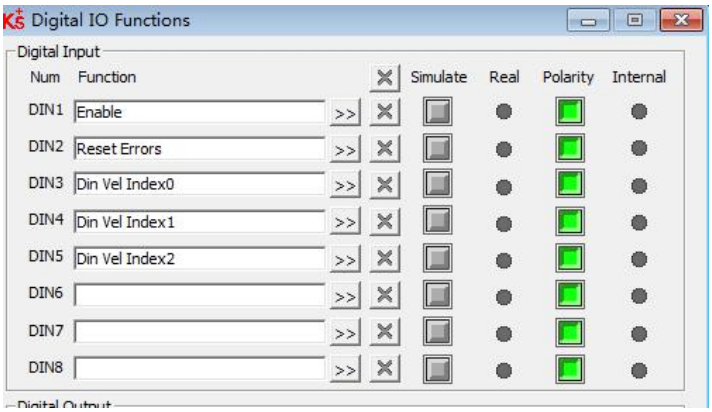
The DIN speed object window in the PC software can be accessed via menu item **Controller->Control Modes->DIN Speed Mode**.

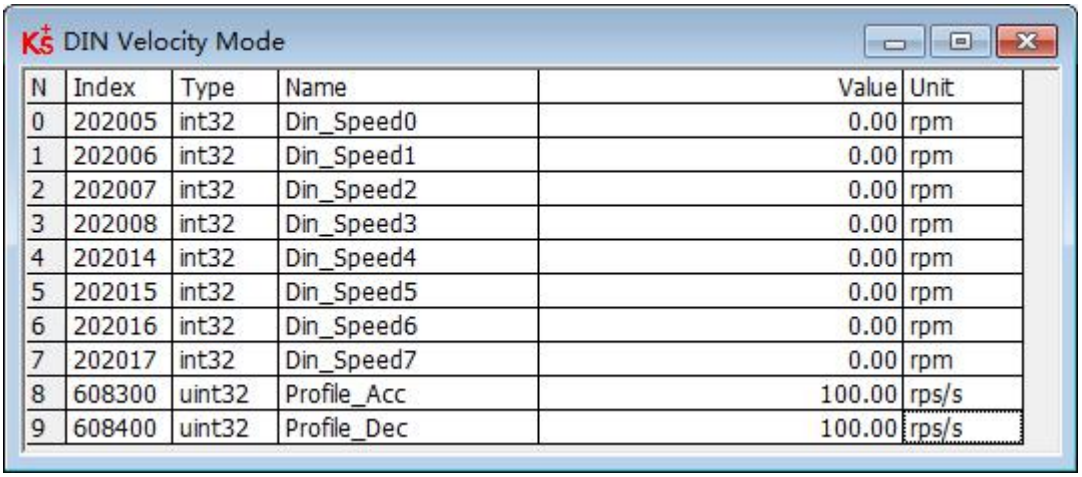
Table5−3DIN speed mode

Internal address	Type	Name	Description	设置值
20200520	Integer32	Din speed0	The speed command of the driver is specified by DIN speed [x], where x is the BCD code from: ： Bit 0: Din Vel index**0;** Bit 1:Din Vel index**1** Bit 2:Din Vel index**2;**	用户定义
20200620	Integer32	Din speed1		
20200720	Integer32	Din speed2		
20200820	Integer32	Din speed3		
20201420	Integer32	Din speed4		
20201520	Integer32	Din speed5		
20201620	Integer32	Din speed6		
20201720	Integer32	Din speed7		

For example:

I/o configuration windowz:





N	Index	Type	Name	Value	Unit
0	202005	int32	Din_Speed0	0.00	rpm
1	202006	int32	Din_Speed1	0.00	rpm
2	202007	int32	Din_Speed2	0.00	rpm
3	202008	int32	Din_Speed3	0.00	rpm
4	202014	int32	Din_Speed4	0.00	rpm
5	202015	int32	Din_Speed5	0.00	rpm
6	202016	int32	Din_Speed6	0.00	rpm
7	202017	int32	Din_Speed7	0.00	rpm
8	608300	uint32	Profile_Acc	100.00	rps/s
9	608400	uint32	Profile_Dec	100.00	rps/s

Figure 5-4IO DIN Speed Mode" window

Table 5–4DIN Speed ModeRelated settings

Internal address	Type	Name	Value	unit
20200E08	Integer8	Din_Mode1	-3	DEC
20200732	Integer32	Din_Speed[2]	500	rpm

5.1.2 DIN Speed mode

The Din_Speed object window in PC software can be accessed from menu item **Controller->Control Modes->DIN Speed Mode**.

To make the DIN Speed Mode available, at least one of the following has to be configured to DIN: **Din Vel Index0**, **Din Vel Index1**, **Din Vel Index2**.

Table0−3 DIN speed mode

Internal address	Type	Name	Description	Value
20200520	Integer32	Din speed[0]	The velocity command is specified via Din_Speed[x]. x is the BCD code of Bit 0: Din Vel Index0 Bit 1: Din Vel Index1 Bit 2: Din Vel Index2 A bit which is not configured means 0.	User defined
20200620	Integer32	Din speed[1]		
20200720	Integer32	Din speed[2]		
20200820	Integer32	Din speed[3]		
20201420	Integer32	Din speed[4]		
20201520	Integer32	Din speed[5]		
20201620	Integer32	Din speed[6]		
20201720	Integer32	Din speed[7]		

Example：

I/O configuration：

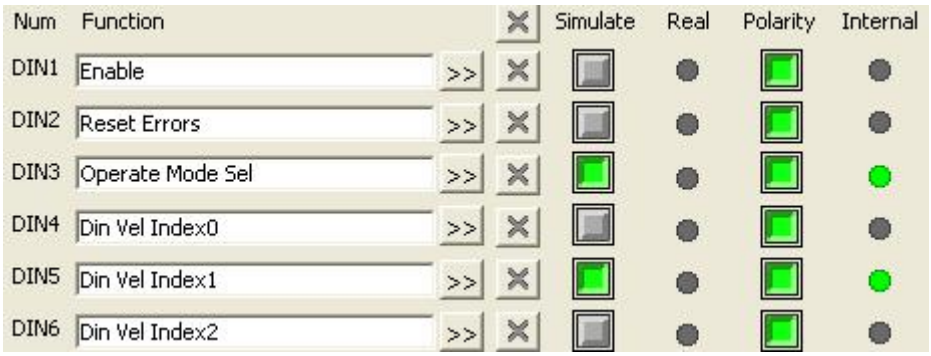
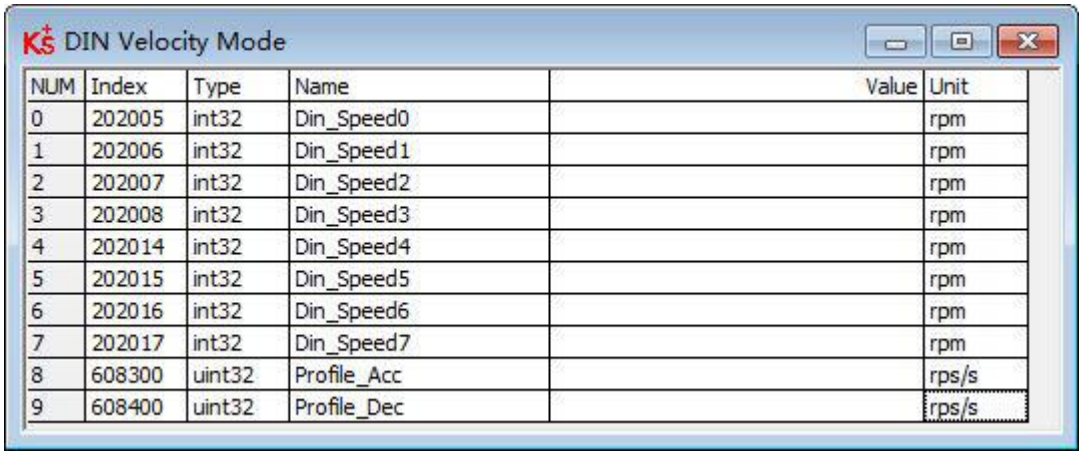


Figure 0−3 IO configuration



NUM	Index	Type	Name	Value	Unit
0	202005	int32	Din_Speed0		rpm
1	202006	int32	Din_Speed1		rpm
2	202007	int32	Din_Speed2		rpm
3	202008	int32	Din_Speed3		rpm
4	202014	int32	Din_Speed4		rpm
5	202015	int32	Din_Speed5		rpm
6	202016	int32	Din_Speed6		rpm
7	202017	int32	Din_Speed7		rpm
8	608300	uint32	Profile_Acc		rps/s
9	608400	uint32	Profile_Dec		rps/s

Figure 0–4 IO "DIN speed mode" window

Table0–4 DIN Speed mode

Internal address	Type	Name	Value	Unit
20200E08	Integer8	Din mode 1	-3	
20200732	Integer32	Din speed [2]	500	rpm

Din Vel Index0=0; **Din Vel Index1**=1; **Din Vel Index2**=0. As soon as DIN1 is active, the controller runs the motor in the velocity mode(Operation_Mode=-3) at 500rpm speed if there aren't any unexpected errors or limits.

5.2 Torque mode (4)

In the torque mode, the CD3 motor controller causes the motor to rotate with a specified torque value.

Table 0–5 Torque mode

Internal address	Type	Name	Description	Value
60600008	Integer8	Operation_mode		4
60710010	Integer16	Target_Torque %	Target torque, percentage of rated torque	User define
60400010	Unsigned16	Controlword	Enable driver	0x0F

5.2 Position mode (1)

In the position mode, the driver causes the motor to rotate to an absolute or relative position. The position / velocity command is specified via Target_Position / Profile_Speed or via position table (Position Table Mode)

Table 5−7 Position mode

Internal address	Type	Name	Description	Value
60600008	Integer8	Operation_ Mode	Way of control motor	1
607A0020	Integer32	Target_Pos ition	Target absolute / relative position	User defined
60810020	Unsigned32	Profile_Sp eed	Profile speed for positioning	User defined
60400010	Unsigned16	Controlwor d	Switch from 0x2F to 0x3F：Absolute position; Switch from 0x4F to 0x5F：Relative position	0x2F->0x3F or 0x4F->0x5F

5.3 Pulse mode (-4)

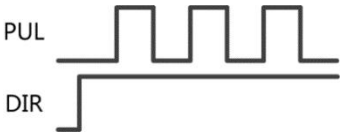
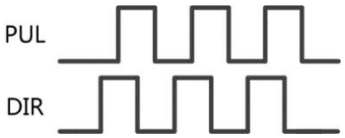
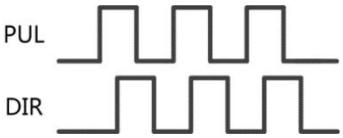
In the pulse mode, the target velocity command is specified via the pulse input with gear ratio.

Table5−8 Pulse mode

Internal address	Type	Name	Description	Value
60600008	Integer8	Operation_Mode	Operation mode	-4
25080110	Integer16	Gear_Factor[0]	Gear_ratio=Gear_Factor/Gear_Divider	User define
25080210	Unsigned16	Gear_Divider[0]		
60400010	Unsigned16	Controlword	Enable driver	0x2F:
25080308	Unsigned 8	PD_CW	Pulse train mode 0: CW / CCW	0, 1, 2

			1: Pulse / direction 2: A / B (incremental encoder)	
25080610	Unsigned16	PD_Filter	Pulse filter (ms)	User define
25080810	Unsigned16	Frequency_Check	Frequency limit (inc/ms), if pulse count (in 1 ms) is greater than Frequency_Check, over frequency error occurs.	

Table0–9 PD_CW schematic

Pulse mode	Forward	Reverse
P/D		
CW/CCW		
A/B		

Note

Forward means positive position counting’ s defaulted to the CCW direction. You can set Invert_Dir(607E.00) to 1 in order to invert the direction of motor shaft rotation.

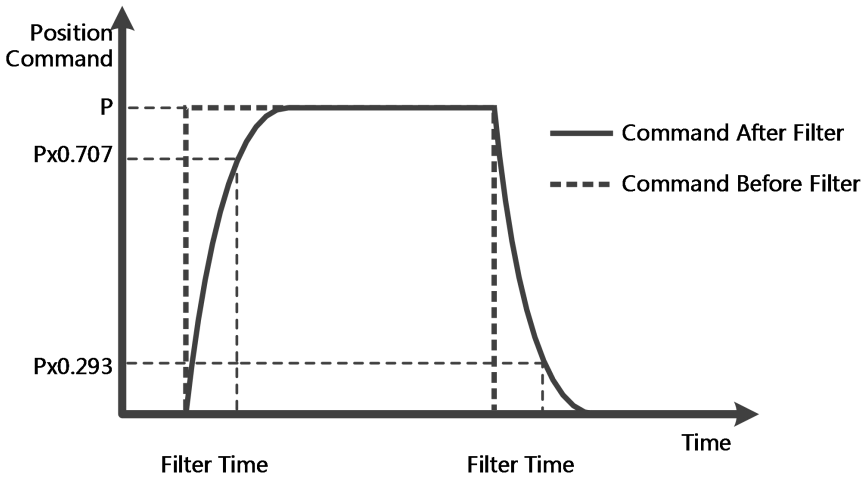


Figure 5-6 Pulse filter principle

5.4 Homing mode (6)

For some applications, the system needs to start from the same position every time after power on. In the homing mode, the user can specify the system’ s home position and a zero (starting) position.

Click menu item **Controller**->**Control Modes**->**Homing definition**, and the following window appears:

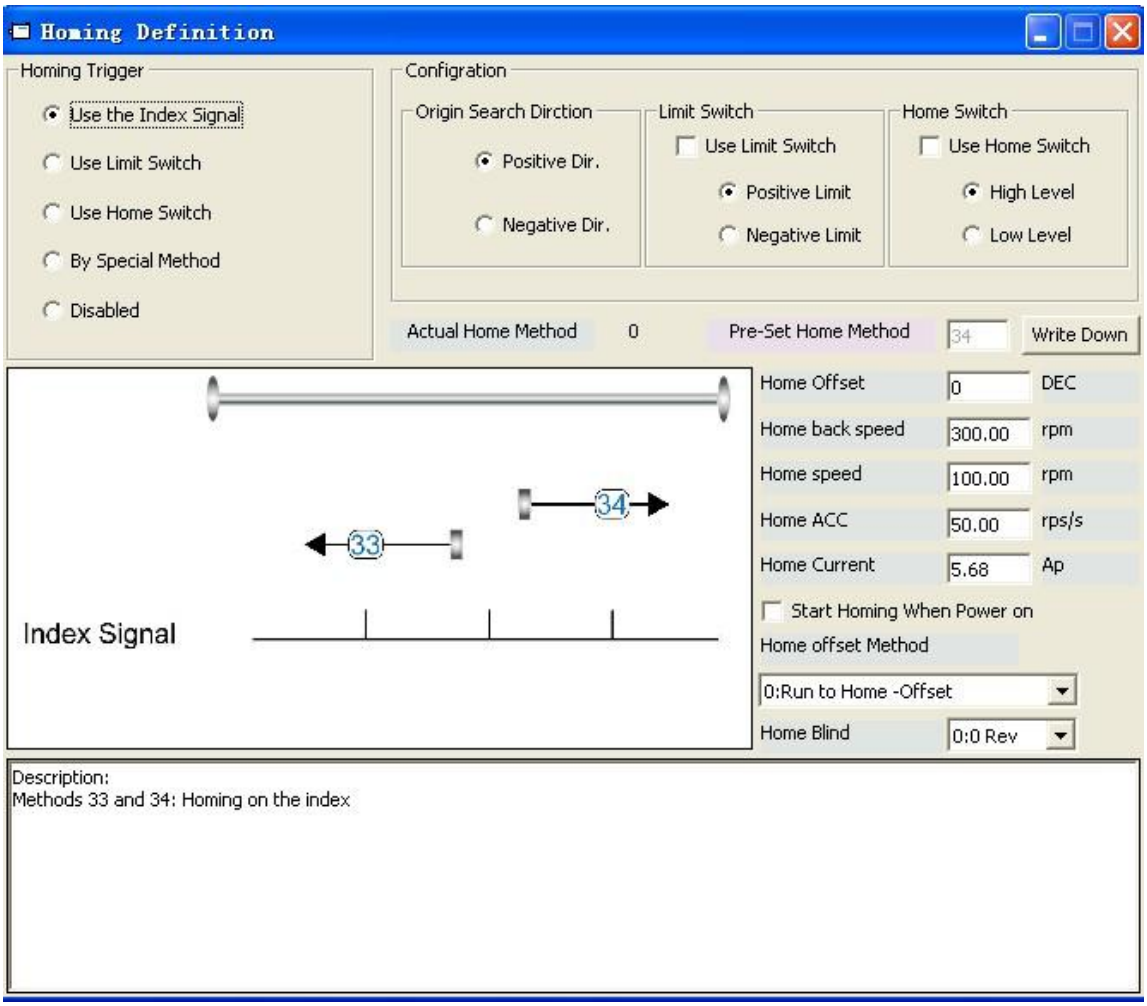


Figure 0–7 Homing settings

Select a home trigger under **Homing Trigger**. The related items appear in the **configuration** area. Select a suitable item according to mechanical design and wiring. The Appropriate homing_method then appears in the **Pre-Set Home Method** box. If **Disabled** is selected under homing trigger, you enter a number directly to the **Pre-Set Home Method** field. Click Write Down to set it to the controller.

The corresponding diagram of the Pre-Set Home method appears in the middle area.

All homing mode objects are listed in following table:

Table0−10 Homing mode

Internal address	Name	Type	Value	Description
607C0020	Home_Offset	Integer32	User define	Zero position offset to the home position
60980008	Homing_Method	Integer 8	User define	Way of homing method
60990220	Homing_Speed_Zero	Unsigned20	User define	Velocity for finding home position and zero position
60990308	Homing_Power_On	Unsigned 8	0，1	1: Start homing after power on or reboot and first controller enable
609A0020	Homing_Acceleration	Unsigned32	User define	Profile deceleration and acceleration during homing
60990120	Homing_Speed_Switch	Unsigned32	User define	Velocity for searching position limit switch / home switch signal
60990410	Homing_Current	Integer8	User define	Max. current during homing
60990508	Home_Offset_Mode	Unsigned 8	0，1	0: Go to the homing offset point. The actual position will be 0. 1: Go to the home trigger point. The actual position will be -homing offset.
60990608	Home_N_Blind	Unsigned 8	0，1	Home blind window 0: 0rev 1: 0.25rev 2: 0.5rev
60600008	Operation_Mode	Integer8	6	Operation mode
60400010	Controlword	Unsigned16	0x0F->0x1F	Enable driver

Note

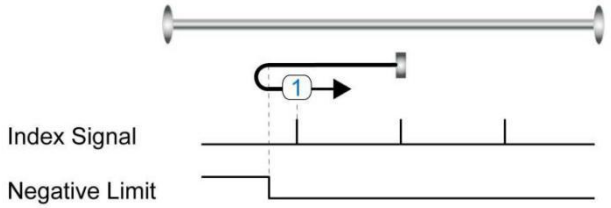
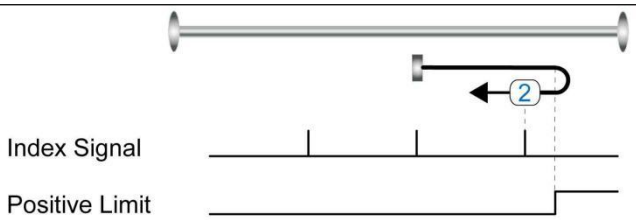
Homing_Power_On=1 causes the motor to start rotating as soon as the controller is enabled after power on or reboot. Consider all safety issues before using.

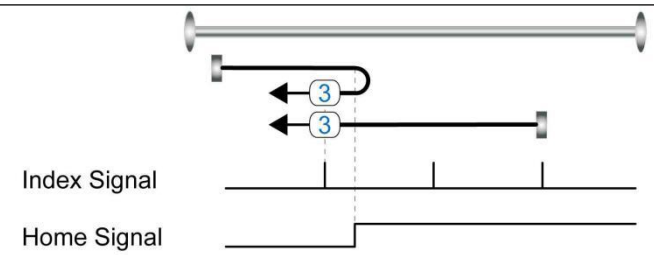
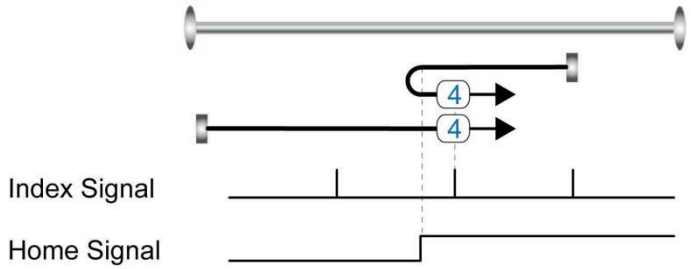
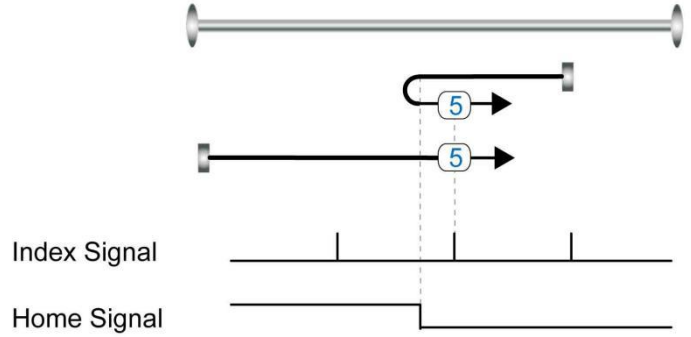
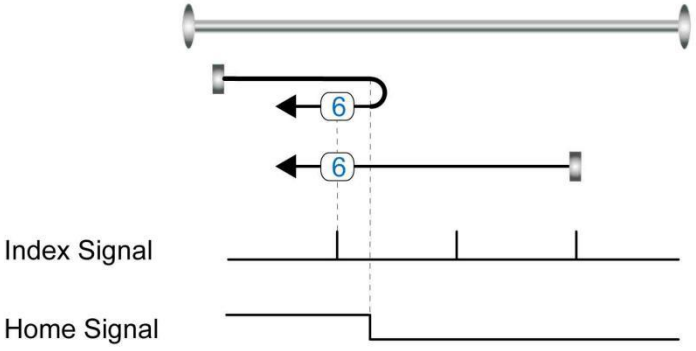
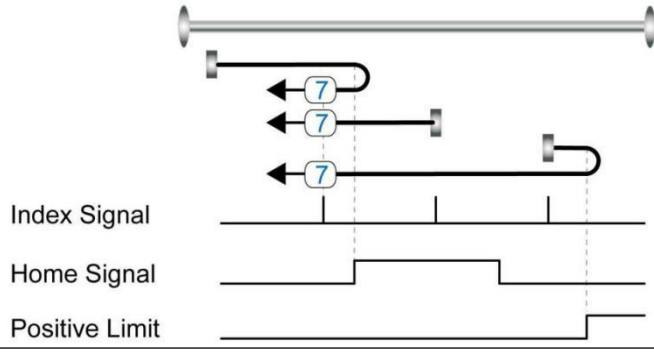
Home_N_Blind:

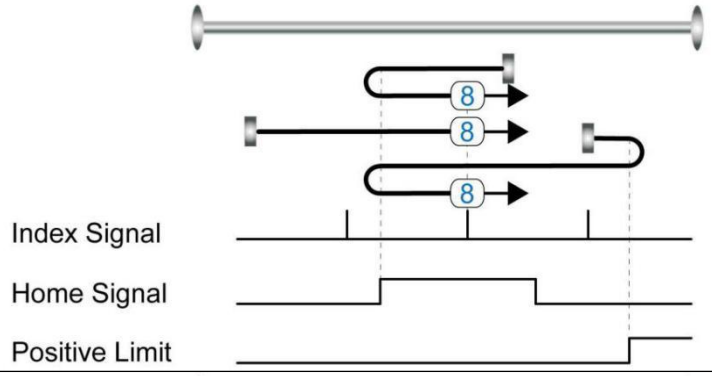
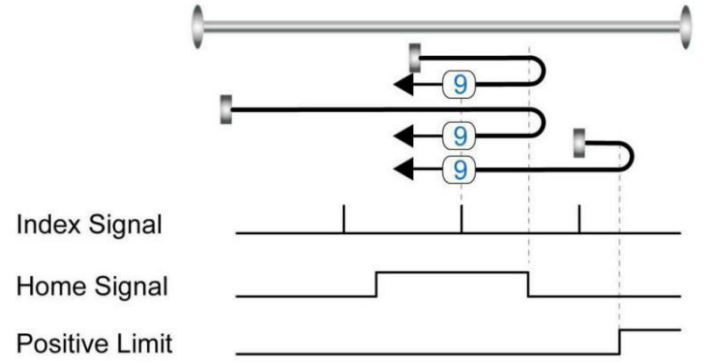
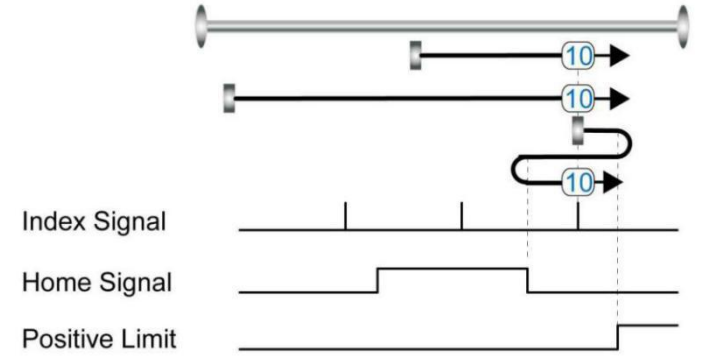
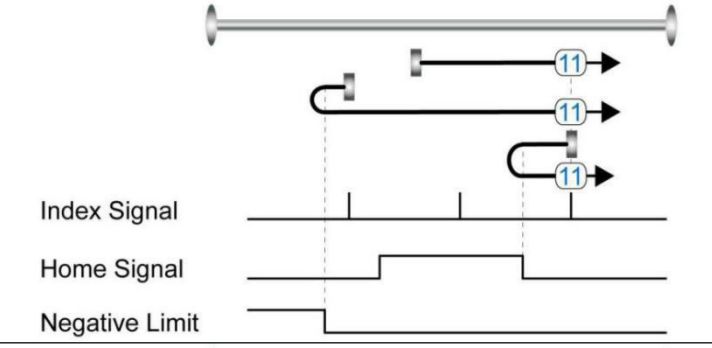
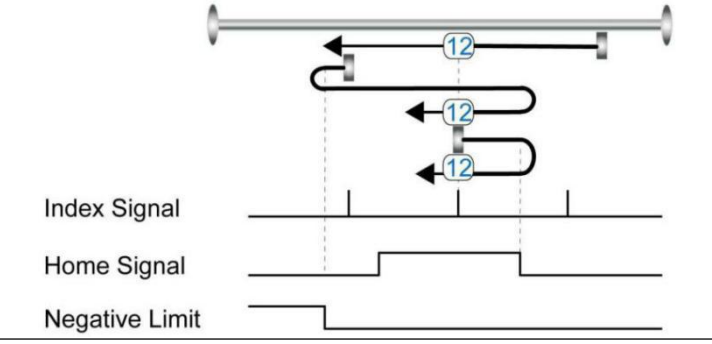
If the homing_method needs home signal (position limit / home switch) and index signal, Home_N_Blind function can avoid the homing result being different with the same mechanics, when the Index signal is very close to the home signal. By setting to 1 before homing, the controller detects a suitable blind window for homing automatically. It can be used to assure that homing results are always the same.

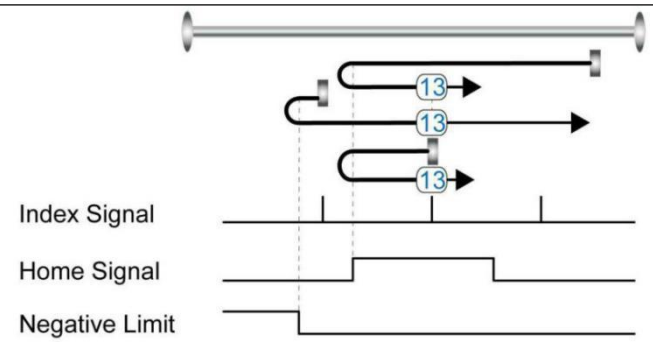
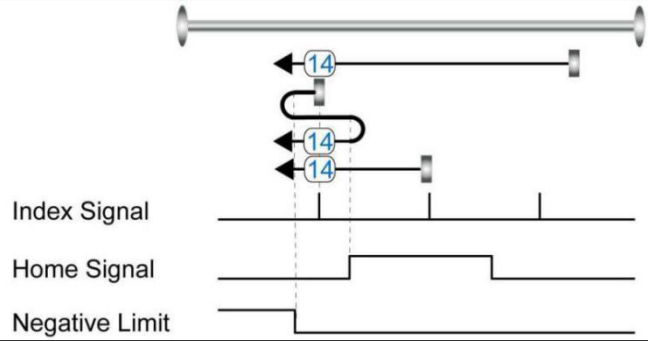
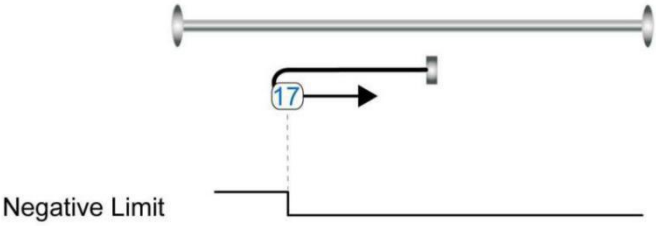
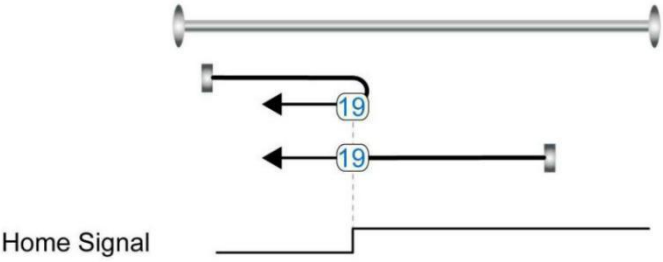
During homing, the index signal inside this blind window is ignored after the home signal is found. Home_N_Blind (0:0rev;1:0.25rev;2:0.5rev) is defaulted to 0. If it's set to 1, it' s changed to 0 or 2 after homing depending on the index signal position relative to the homing signal.This parameter needs to be saved. If the mechanical assembly is changed or the motor has been replaced, just set it to 1 again for initial homing.

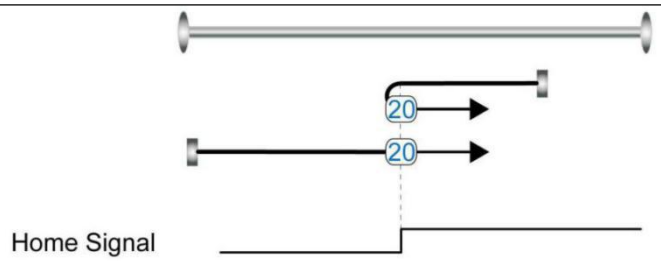
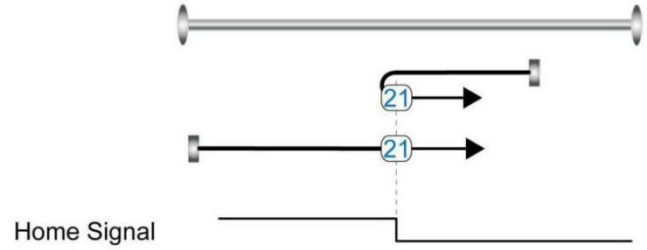
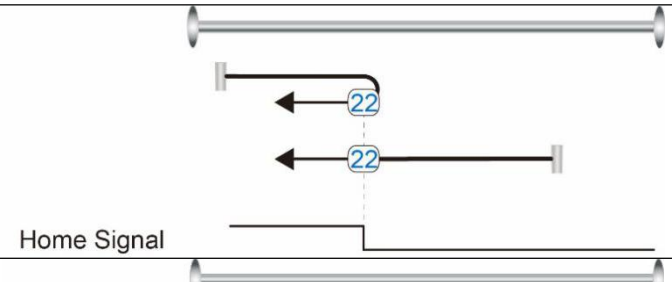
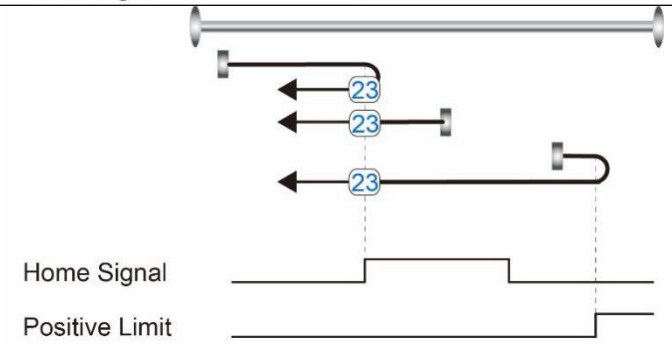
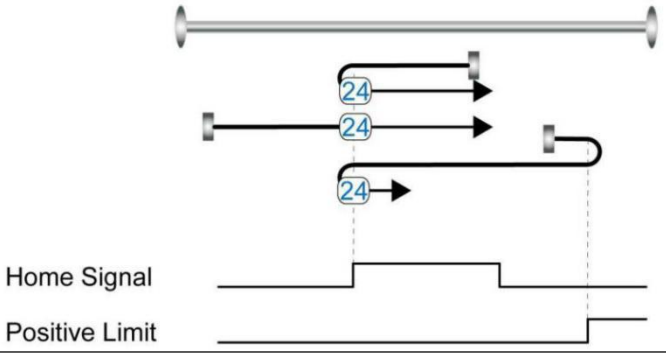
Table0−11 Introduction to homing mode

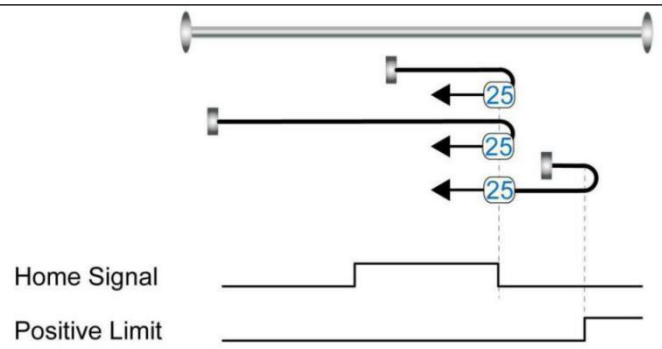
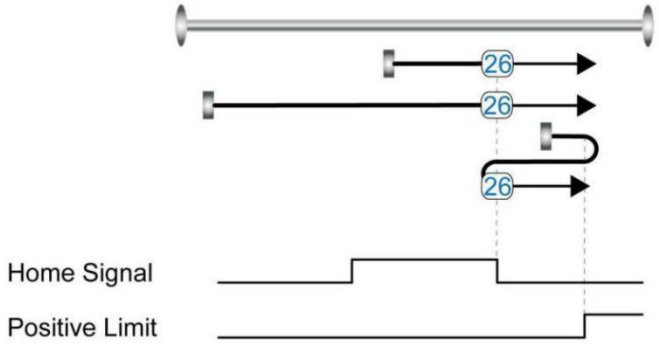
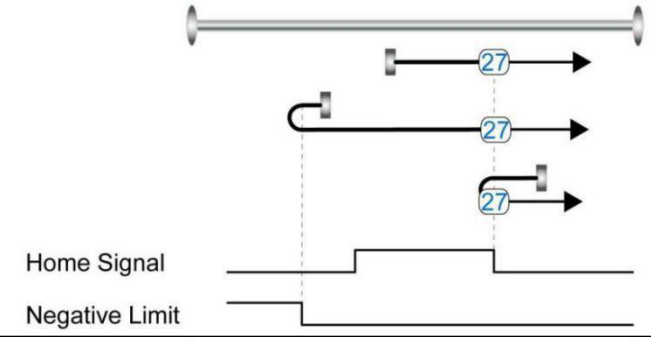
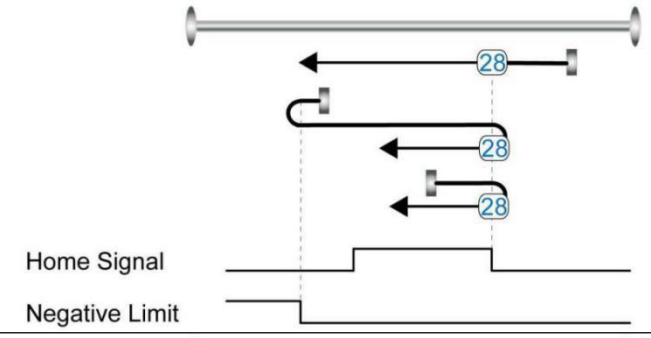
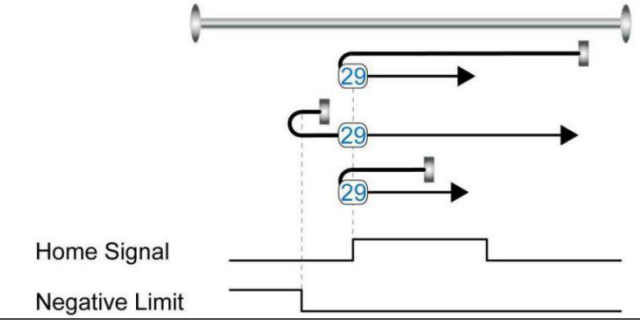
Homing_method	Description	Schematic
1	Homing with negative position limit switch and index pulse	
2	Homing with positive position limit switch and index pulse	

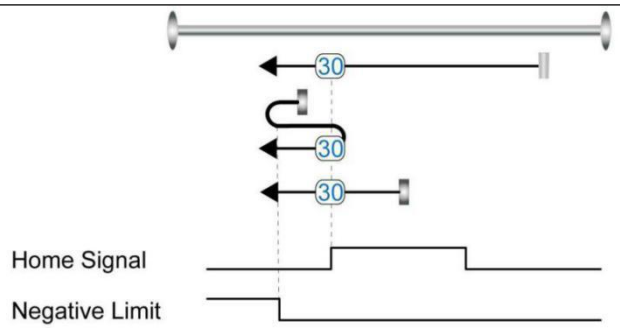
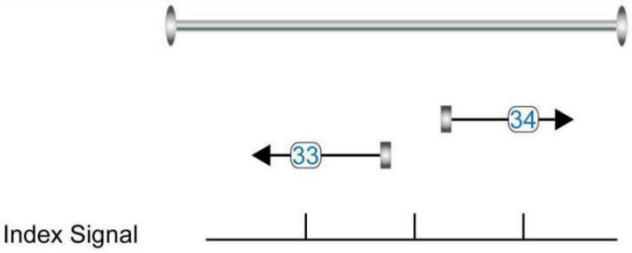
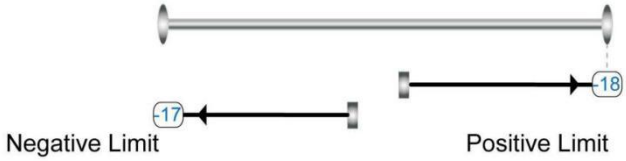
3	Homing with home switch and index pulse	
4	Homing with home switch and index pulse	
5	Homing with home switch and index pulse	
6	Homing with home switch and index pulse	
7	Homing with positive position limit switch, home switch and index pulse	

8	Homing with positive position limit switch, home switch and index pulse	
9	Homing with positive position limit switch, home switch and index pulse	
10	Homing with positive position limit switch, home switch and index pulse	
11	Homing with negative position limit switch, home switch and index pulse	
12	Homing with negative position limit switch, home switch and index pulse	

13	Homing with negative position limit switch, home switch and index pulse	
14	Homing with negative position limit switch, home switch and index pulse	
17	Homing with negative position limit switch	
18	Homing with positive position limit switch	
19	Homing with home switch	

20	Homing with home switch	
21	Homing with home switch	
22	Homing with home switch	
23	Homing with positive position limit switch and home switch	
24	Homing with positive position limit switch and home switch	

25	Homing with positive position limit switch and home switch	
26	Homing with positive position limit switch and home switch	
27	Homing with negative position limit switch and home switch	
28	Homing with negative position limit switch and home switch	
29	Homing with negative position limit switch and home switch	

30	Homing with negative position limit switch and home switch	
33, 34	Homing with index pulse	
35	Homing to actual position	
-17, -18	Homing via mechanical limit	

Chapter 6 Tuning of the servo system control

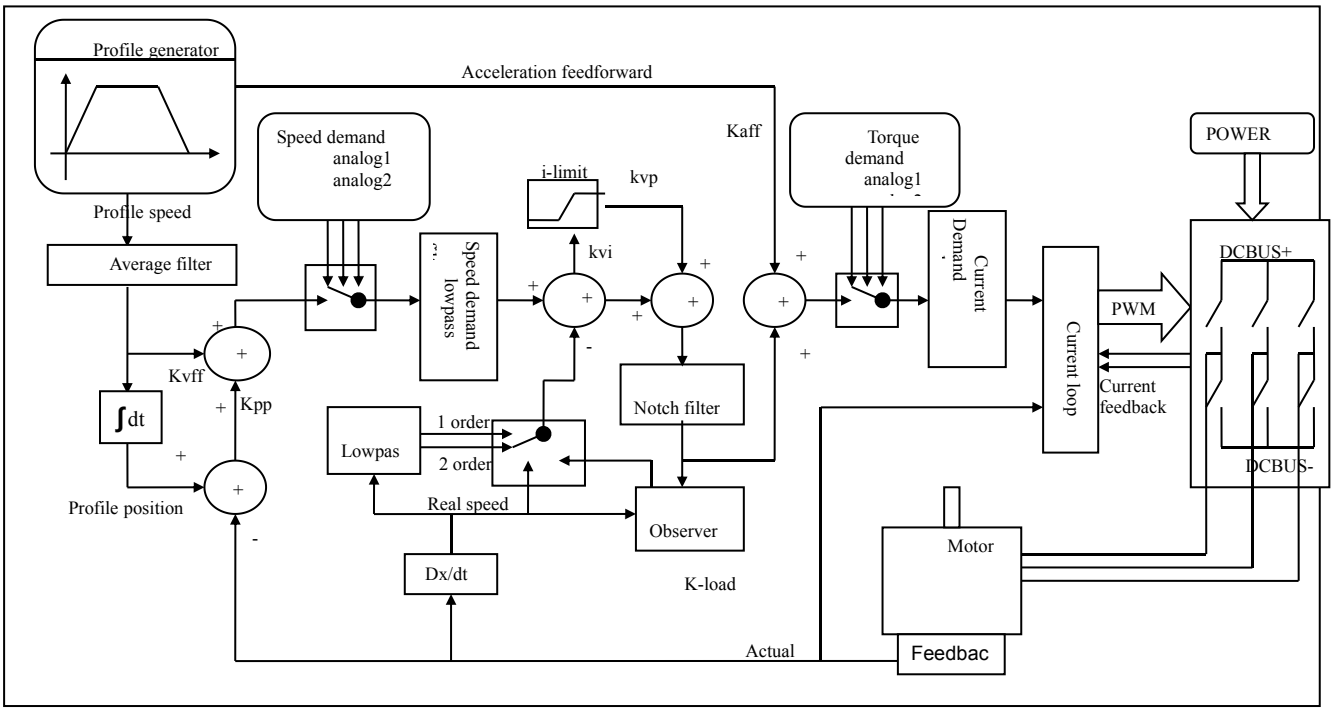


Figure 6–1Servo system control block diagram

Figure 6-1 shows the servo system control block diagram. It can be seen from the figure that the servo system generally includes three control loops: current loop, velocity loop and position loop.

The adjustment process of a servo system is used to set loop gain and filters to match the mechanical characteristics, and finally to prevent the entire system from oscillating, to permit it to follow commands quickly and to eliminate abnormal noise.

- kaff：Position loop acceleration feedforward
- kvff：Position loop velocity feedforward
- kvp：Velocity loop proportional gain
- kvi：Velocity loop integration gain
- kpp：Position loop proportional gain

6.1 Tuning of velocity loop

Table0−1 List of velocity loop parameters

Internal address	Name	Description	Default	Range
60F90110	Kvp[0]	Proportional velocity loop gain Can be displayed in Hz in the PC tool can if the inertia ratio is right.	/	1~32767
60F90210	Kvi[0]	Integral velocity loop gain	/	0-1023
60F90710	Kvi/32	Integral velocity loop gain of in a smaller unit of measure	/	0-32767
60F90508	Speed_Fb_N	Used to set Velocity feedback filter bandwidth Filter bandwidth=100+Speed_Fb_N*20	7	0~45
60F90608	Speed_Mode	Used to set the velocity feedback mode 0: 2nd order FB LPF 1: Directly feedback the original velocity 2: Velocity feedback after velocity observer 4: Velocity feedback after 1st order LPF 10:Velocity feedback after 2nd order LPF and the velocity command is filtered by a 1st order LPF. Both filters have the same bandwidth. 11: The velocity command is filtered by a 1st order LPF 12: Velocity feedback after velocity observer, the velocity command is filtered by a 1st	1	/

		order LPF 14: Velocity feedback after 1st order LPF and the velocity command is filtered by a 1st order LPF. Both filters have the same bandwidth		
60F91508	Output_Filter_N	A 1st order lowpass filter in the forward path of the velocity loop	1	1-127
60F90820	Kvi_Sum_Limit	Integral output limit of the velocity loop	/	0-2^15

Step of Velocity loop tuning is shown below:

Velocity feedback filter adjustment

The velocity feedback filter can reduce noise that comes from the feedback path, e.g. reduce encoder resolution noise. The velocity feedback filter can be configured as 1st and 2nd order via the Speed_Mode for different applications. The 1st order filter reduces noise to a lesser extent, but its also results in less phase shifting so that velocity loop gain can be set higher. The 2nd order filter reduces noise to a greater extent, but its also results in more phase shifting so that velocity loop gain can be limited.

Normally, if the machine is stiff and light, we can use the 1st feedback filter or disable the feedback filter. If the machine is soft and heavy, we can use the 2nd order filter.

If there’ s too much motor noise when velocity loop gain is adjusted, velocity loop feedback filter parameter Speed_Fb_N can be reduced accordingly. However, velocity loop feedback filter bandwidth F must be more than twice as large as the velocity loop bandwidth. Otherwise, it may cause oscillation. Velocity loop feedback filter bandwidth F=Speed_Fb_N*20+100 [Hz].

Output filter adjustment

The output filter is a 1st order torque filter. It can reduce the velocity control loop to output high frequency torque, which may stimulate overall system resonance.

The user can try to adjust Output_Filter_N from small to large in order to reduce noise.

The filter bandwidth can be calculated using the following formula.

$$\frac{1}{2}\frac{\ln\left(1-\frac{1}{Output\_Filter\_N}\right)}{Ts\,\pi},\ Ts=62.5\,us$$

Velocity loop bandwidth calculation

Use the following formula to calculate velocity loop bandwidth:

$$kvp=\frac{1.853358080\,10^{5}\,J\pi^{2}\,Fbw}{\mathrm{I}_{Max}\,kt\,encoder}$$

kt motor torque constant, unit: Nm/Arms*100

J inertia, unit: kg*m^2*10^6

Fbw Velocity loop bandwidth, unit: Hz

Imax max motor current I_max(6510.03) as DEC value

encoder resolution of the encoder

Integral gain adjustment

Integral gain is used to eliminate static error. It can boost velocity loop low frequency gain, and increased integral gain can reduce low frequency disturbance response.

Normally, if the machine has considerable friction, integral gain (kvi) should be set to a higher value.

If the entire system needs to respond quickly, integral should be set to a small value or even 0, and the gain switch should be used.

Adjust Kvi_sum_limit

Normally the default value is fine. This parameter should be added if the application system has a big extend force, or should be reduced if the output current is easily saturation and the saturation output current will cause some low frequency oscillation.

6.2 Tuning of position loop

Table0−2 List of position loop parameters

Internal address	Name	Description	Default	Range
60FB0110	Kpp[0]	Proportional position loop gain. Used to set the position loop response. unit: 0.01Hz	10	0 ~ 32767
60FB0210	K_Velocity_FF	0 means no feedforward, 1000 means 100% feedforward.	100	0 ~ 100
60FB0310	K_Acc_FF	The unit only is right if the inertia ratio is correctly set. If the inertia ratio is unknown, set K_Acc_FF(60FB.03) instead.	/	0-32767
60FB0510	Pos_Filter_N	The time constant of the position demand LPFunit: ms	1	1~255

60650020	Max_Following_Error_16	Maximum allowable error, Max_Following_Error (6065.00) = 100 * Max_Following_Error_16	10000	/

Step of Position loop tuning is shown below:

Position loop proportional gain adjustment

Increasing position loop proportional gain can improve position loop bandwidth, thus reducing positioning time and following error, but setting it too high will cause noise or even oscillation. It must be set according to load conditions. Kpp = 103 * Pc_Loop_BW, Pc_Loop_BW is position loop bandwidth. Position loop bandwidth cannot exceed velocity loop bandwidth. Recommended velocity loop bandwidth: Pc_Loop_BW<Vc_Loop_BW / 4, Vc_Loop_BW.

Position loop velocity feedforward adjustment

Increasing the position loop velocity feedforward can reduce position following error, but can result in increased overshooting. If the position command signal is not smooth, reducing position loop velocity feedforward can reduce motor oscillation.

The velocity feedforward function can be treated as the upper controller (e.g. PLC) have a chance to directly control the velocity in a position operation mode. In fact this function will expend part of the velocity loop response ability, so if the setting can’ t match the position loop proportional gain and the velocity loop bandwidth, the overshot will happen.

Besides, the velocity which feedforward to the velocity loop may be not smooth, and with some noise signal inside, so big velocity feedforward value will also amplified the noise.

Position loop acceleration feedforward

It is not recommended that the user adjust this parameter. If very high position loop gain is required, acceleration feedforward K_Acc_FF can be adjusted appropriately to improve performance.

The acceleration feedforward function can be treat as the upper controller (e.g. PLC) have a chance to directly control the torque in a position operation mode. in fact this function will expend part of the current loop response ability, so if the setting can ’ t match the position loop proportional gain and the velocity loop bandwidth, the overshot will happen.

Besides, the acceleration which feedforward to the current loop can be not smooth, and with some noise signal inside, so big acceleration feedforward value will also amplified the noise.

Acceleration feedforward can be calculated with the following formula:

ACC_%=6746518/ K_Acc_FF/EASY_KLOAD*100

ACC_%: the percentage which will be used for acceleration feedforward.

K_Acc_FF(60FB.03): the final internal factor for calculating feedforward.

EASY_KLOAD(3040.07): the load factor which is calculated from auto-tuning or the right inertia ratio input.

➔ **Note**

The smaller the K_Acc_FF, the stronger the acceleration feedforward.

Smoothing filter

The smoothing filter is a moving average filter. It filters the velocity command coming from the velocity generator and makes the velocity and position commands more smooth. As a consequence, the velocity command will be delayed in the controller. So for some applications likeCNC, it' s better not to use this filter and to accomplish smoothing with the CNC controller.

The smoothing filter can reduce machine impact by smoothing the command. The Pos_Filter_N parameter define the time constant of this filter in ms. Normally, if the machine system oscillates when it starts and stops, a larger Pos_Filter_N is suggested.

Notch filter

The notch filter can suppress resonance by reducing gain around the resonant frequency.

Antiresonant frequency=Notch_N*10+100

Setting Notch_On to 1 turns on the notch filter. If the resonant frequency is unknown, the user can set the maximum value of the d2.14 current command small, so that the amplitude of system oscillation lies within an acceptable range, and then try to adjust Notch_N and observe whether the resonance disappears.

Resonant frequency can be measured roughly according to the Iq curve when resonance occurs on the software oscilloscope.

Table0–3 Notch filter list

Internal address	Name	Description	Default	Range
60F90308	Notch_N	Used to set the frequency of the internal notch filter to eliminate mechanical resonance generated when the motor drives the machine. The formula is F=Notch_N*10+100. For example, if mechanical resonance frequency F=500 Hz, the parameter	45	0~90

		setting should be 40.		
60F90408	Notch_On	Used to turn on or turn off the notch filter. 0：Turn on the notch filter 1：Turn off the notch filter	0	0~1

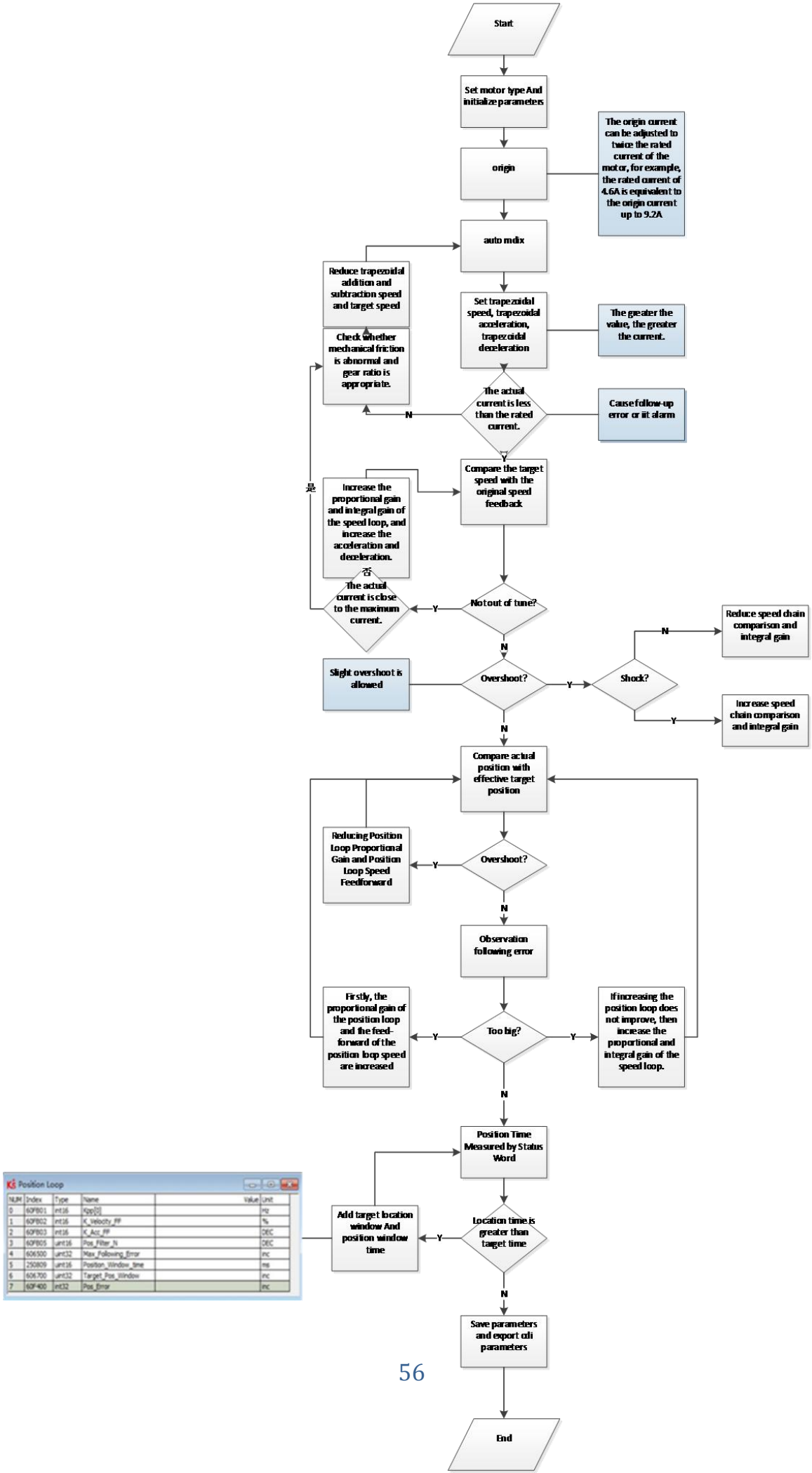
6.3 Factors which influence tuning results

The control command is created by the upper controller (e.g. PLC):

The control command should be smooth as much as possible, and must be correct. For example, the control command should not create the acceleration commands (inside the position commands) that the motor cannot provide. Also, the control command should follow the bandwidth limit of the control loop.

The machine design:

In the actual application, performance is normally limited by the machine. Gaps in the gears, soft connection in the belts, friction in the rail, resonance in the system – all of these can influence final control performance. Control performance affects the machine’s final performance, as well as precision, responsiveness and stability. However, final machine performance is not only determined by control performance.


Start
Set motor type And initialize parameters
origin
The origin current can be adjusted to twice the rated current of the motor, for example, the rated current of 4.6A is equivalent to the origin current up to 9.2A
auto mdix
Reduce trapezoidal addition and subtraction speed and target speed
Set trapezoidal speed, trapezoidal acceleration, trapezoidal deceleration
The greater the value, the greater the current.
Check whether mechanical friction is abnormal and gear ratio is appropriate.
The actual current is less than the rated current.
Cause follow-up error or int alarm
N
Y
Compare the target speed with the original speed feedback
Increase the proportional gain and integral gain of the speed loop, and increase the acceleration and deceleration.
The actual current is close to the maximum current.
Not out of tune?
Y
N
Reduce speed chain comparison and integral gain
Slight overshoot is allowed
Overshoot?
Y
Shock?
N
Y
Increase speed chain comparison and integral gain
N
Compare actual position with effective target position
Reducing Position Loop Proportional Gain and Position Loop Speed Feedforward
Overshoot?
Y
N
Observation following error
Firstly, the proportional gain of the position loop and the feed-forward of the position loop speed are increased
Too big?
Y
Y
If increasing the position loop does not improve, then increase the proportional and integral gain of the speed loop.
N
Position Time Measured by Status Word
Add target location window And position window time
Location time is greater than target time
Y
N
Save parameters and export oli parameters
End
Position Loop
NUM Index Type Name Value Unit
0 60FB01 int16 Kpp[0] Hz
1 60FB02 int16 K_Velocity_FF %
2 60FB03 int16 K_Acc_FF DEC
3 60FB05 unit16 Pos_Filter_N DEC
4 606500 unit32 Max_Following_Error inc
5 250809 unit16 Position_Window_time ms
6 606700 unit32 Target_Pos_Window inc
7 60F400 int32 Pos_Error inc

Chapter 7 Alarms and troubleshooting

When driver generate an alarm, red light, ERR, will shine.

If you need more detailed information about errors and error history, please connect the controller to the PC via RS232.

Table0−1 Alarm codes of Error_State 1

Alarm	Code	Name	Reason	Troubleshooting
000.1		Extended Error	Errors occurs in Error_State2	Press the SET key to enter Error_State2 (d1.16), read the error bit, check the error meaning in table 7-2.
000.2	7380	Encoder ABZ signal incorrect (suitable for incremental encoder motor)	Encoder ABZ wiring is wrong or disconnected	Check encoder cable is correctly connected Check if corresponding pins of encoder cable is on (refer servo product menu)
		Encoder communication incorrect (suitable for magnetoelectric encoder motor)	The encoder wiring is incorrect or disconnected.	
000.4	7381	Encoder UVW signal incorrect (suitable for incremental encoder motor)	Encoder UVW wiring is wrong or disconnected	Check encoder cable is correctly connected Check if corresponding pins of encoder cable is on (refer servo product menu) Change motor
		Encoder internal (suitable for magnetoelectric encoder motor)	Encoder internal is incorrect or encoder is broken	

000. 8	7306	Encoder count wrong（suitable for incremental encoder motor） Encoder CRC (suitable for magnetoelectric encoder motor)	Encoder is interfered	1.Check encoder cable is correctly connected (different from motor PE cable) 2.Make sure the equipment is well grounded 3.Use isolated power supply to provide power
001. 0	4210	Controller temperature	The temperature of controller ' s power module has reached the alarm value	Add fan ，improve the cooling environment of the controller. Add driver installment distance Vertically install driver
002. 0	3210	Overvoltage	Supply power voltage exceeds the allowable input voltage range	Check if supply power is higher than standard output voltage Check to see if supply power voltage is unstable
			In case of emergency stop, there is no external braking resistor or braking.	Connect suitable braking resistor Open software "Driver"-> ″ Panel menu″ -> ″（F005）controller setting″ Correctly set "brake resistor value" an "brake resistor power"
			Brake resistor is not configured	Change Connect suitable braking resistor Open software "Driver"-> ″ Panel menu″ -> ″（F005）controller setting″ Correctly set "brake resistor value" an "brake resistor power"
004. 0	3220	Undervoltage	The power voltage input is lower than the low voltage protection alarm value.	Check if power supply output power can meet with the requirement Change power supply of bigger power
008. 0	2320	Short circuit of driver output	Short circuit of driver UVW and PE output	Check if motor power cable connection is correct Driver is broken, change driver

010. 0	7110	Driver brake resistor is abnormal	Not configure correct brake resistor parameters	Open software "Driver"-> ″ Panel menu″ -> ″(F005) controller setting″ Correctly set "brake resistor value" an "brake resistor power"
020. 0	8611	Following error	Stiffness of control loop is too small	Open software "Driver""control loop""velocity loop""position loop" Increase "kpp[0]""kvp[0]"
			Motor UVW phase sequence is incorrect	Exchanging wire of U and V
			The controller and motor together can′t match the requirement of the application	Change motor and driver with bigger power
			Max_Following_Error is too small	Open software "Driver""control loop""velocity loop""position loop" Increase "max_following_error" (Ensure control loop parameters is fine, user can change this parameter)
040. 0	5122	Low logic voltage	Logic voltage is less than 18V，power supply voltage is pulled down	Check if power supply output power can meet with requirements Change power supply with bigger power
080. 0	2350	Motor or controller IIt	The brake is not released when the motor shaft is rotating (only for brake motor)	Check if brake cable wiring is correct Check brake power can meet with the requirements (output voltage is DC24V, input current is 1A, output power is bigger than 24W)
			Machine equipment stuck or excessive friction	Cancel motor enable, or power off driver Please drag load to make it move back and forth in motor's running route. Ensure that there is no machine equipment stuck or excessive friction Add lubricate
			Motor UVW phase	Exchange motor wiring of phase U and

			sequence is incorrect	phase V
100.0	8A80	Over input frequency	External input pulse frequency is too high	Reduce external pulse input frequency When ensure safely use motor, increase "Frequency_Check" （Open "Driver"->"Control modes"->"Pulse mode" -> "Frequency_Check" ），max 600
200.0	4310	Motor temperature	The motor temperature exceeds the specified value	Reduce ambient temperature of the motor and improve cooling conditions Reduce acceleration and deceleration Reduce load
400.0	7122	Motor excitation （suitable for incremental encoder）	Motor UVW phase sequence is wrong	Exchange motor wiring of phase U and phase V
			Encoder is not connected	Check encoder cable
		Encoder information （suitable for magnetoelectric encoder）	Communication is incorrect when the encoder is initialized	Check encoder wiring, restart driver
			The encoder type is wrong, e.g. an unknown encoder is connected	
			The data stored in the encoder is wrong	
			The controller can't support the current encoder type	
800.0	6310	EEPROM data	Data is damaged when the power is turned on and data is read from the EEPROM	Open software "Driver" -> "Init Save Reboot" Click "Init Control Parameters" -> "Save Control Parameters" -> "Save Motor Parameters" -> "Reboot" Import cdi file by software

Table 0–2Alarm codes of Error_State2 (extended)

Alarm	Code	Name	Reason	Trouble shooting
000.1	0x5210	Current sensor	Current sensor signal offset or ripple too big	Circuit of current sensor is damaged, please contact the supplier
000.2	0x6010	Watchdog	Software watchdog exception	Please contact the supplier and try to update the firmware
000.4	0x6011	Wrong interrupt	Invalid interrupt exception	Please contact the supplier and try to update the firmware
000.8	0x7400	MCU ID	Wrong MCU type detected	Please contact the supplier
001.0	0x6320	Motor configuration	Motor type is not auto-recognized, no motor data in EEPROM / motor never configured	Install a correct motor type to the controller and reboot
010.0	0x5443	External enable	DIN function ″pre_enable″ is configured, but the input is inactive when the controller is enabled or should become enabled	Solve according to the reason
020.0	0x5442	Positive limit	Positive position limit (after homing), position limit only causes error when Limit_Function (2010.19) is set to 0	Exclude the condition which causes the limit signal
040.0	0x5441	Negative limit	Positive position limit (after homing), position limit only causes error when Limit_Function (2010.19) is set to 0	Exclude the condition which causes the limit signal
080.	0x60	SPI internal	Internal firmware error	Please contact the supplier

0	12		in SPI handling	
200.0	0x8A81	Close loop direction	Different direction between motor and position encoder	Change the encoder counting direction
800.0	0x7306	Master counting	Master encoder counting error	Ensure that the ground connection and the encoder shield work well.

Chapter 8 List of motor controller parameters

Note

CANopen address is same as 232 communication address :

- Use Index (16 bits address), Subindex (8 bit subaddress) to show register addressing ,
- 0x08 means that data length of register store is 1 Byte, 0x10 means that data length of register store is 2 Byte, 0x20 means that data length of register store is 4 Byte ,
- R: Read , W: Write , S:Save , M: Map ,
- A complete CANopen address format is : 60400010(controlword), Modbus address is 4 bits Hexadecimal number
- A complete Modbus address format is : 3100(controlword)

8.1 Mode and Control（0x6040）

Name	CANopen	Modbus	RWS	Data type	Description
Controlword	60400010	3100	RWM	Unsigned16	0x06：Power off motor 0x0F：Power on motor 0x0B：Quick stop，load stop - voltage off 0x2F → 3F：Start absolute positioning 0x4F→5F：Start relative positioning 0x103F：Start absolute positioning while target position change 0x0F-1F：Home positioning 0X80：Clear internal shooting

Status word	60410010	3200	RM	Unsigned16	Status word display driver state bit0：Ready_on bit1：Switch_on bit2：Operation enable bit3：Fault bit4：Voltage_enable bit5：Quick stop bit6：Switch_disabled bit8：Maunufacture0 bit9：Remote bit10：Target_reached bit11：Intlim_active(negative/positive position limit actived) bit12：Setpoint_Ack bit13：Following_Error bit14：Communication_Found bit15：Reference_found
Operation mode	60600008	3500	RWM	Integer8	Operation mode: 1：Position control 3：Speed control 4：Torque control -3：Speed control (Quick fast mode) -4：Pulse train control 6：Home mode 7：Differential complementarity based on CANopen
Absolute/Relative positioning control select	20200F	0CF0	RWS	Unsigned16	When "Driver enable" function is configured to Din and Din is 1, "Controlword" (6040.00) will be set； 0x2F：Absolute positioning control 0x4F：Relative positioning control

8.2 Data measuring

Name	CANopen	Modbus	RWS	Data type	Description
Pos_Actual	60630020	3700	RM	Integer32	
Real current	60780010	3E00	RM	Integer16	
Status of input port	60FD0020	6D00	RM	Unsigned32	bit0: negative limit switch bit1: positive limit switch bit2: home switch bit3: interlock
Real speed	606C0020	3B00	RM	Integer32	rpm

➔ **Note**

0x606C0020，conversion between engineering unit and internal unit of common objects DEC=[(RPM*512*Encoder_Resolution)/1875]

8.3 Target object（0x607A）

Name	CANopen	modbus	RWS	Data type	Description
Invert direction	607E0008	4700	RWS	Unsigned8	Invert motion 0：CCW is positive direction 1：CW is positive direction
Target position	607A0020	4000	RWM	Integer32	Target position in position mode 1. If controlword is set to start motion and transfer to effective command position inc
Profile speed	60810020	4A00	RWM	Unsigned32	Speed of trapezoidal curve in work mode 1
Target speed	60FF0020	6F00	RWM	Integer32	Target speed in mode 3 and -3
Max speed	60800010	4900	RW	Unsigned16	Default 5000rpm
Profile acc	60830020	4B00	RWSM	Unsigned32	Default：610.352rps/s
Profile dec	60840020	4C00	RWSM	Unsigned32	Default：610.352rps/s

Target torque	60710010	3C00	RW	Integer16	Torque command in torque mode , target_torque/rated_torque*100 (unit:%)
Group current loop	60F60810	5880	RWM	Integer16	Current command in torque mode
Maximal current command	60730010	3D00	RWSM	Unsigned16	Maximal current command, unit Arms

➔ **Note**

Speed address：0x60810020，0x60800020，0x60FF0020

Conversion between engineering unit and internal unit of common objects DEC=[(rpm*512*Encoder_Resolution)/1875]

Acc & Dec address：60830020，60840020，

Conversion between engineering unit and internal unit of common objects DEC=[(rps/s*65536*Encoder_Resolution)/4000000]

Current address：60710010，60730010

Conversion between engineering unit and internal unit of common objects 1Arms=(2048/Ipeak/1.414)DEC Ipeak is driver max current

8.4 Din speed/position（0x2020）

Name	CANopen	modbus	RWS	Data type	Description
Din_Pos0	20200120	0C10	RWS	Integer32	
Din_Pos1	20200220	0C20	RWS	Integer32	
Din_Pos2	20200320	0C30	RWS	Integer32	
Din_Pos3	20200420	0C40	RWS	Integer32	
Din_Pos4	20201020	0D00	RWS	Integer32	
Din_Pos5	20201120	0D10	RWS	Integer32	
Din_Pos6	20201220	0D20	RWS	Integer32	
Din_Pos7	20201320	0D30	RWS	Integer32	
Din_Speed0	20200520	0C50	RWS	Integer32	
Din_Speed1	20200620	0C60	RWS	Integer32	

Din_Speed2	20200720	0C70	RWS	Integer32	
Din_Speed3	20200820	0C80	RWS	Integer32	
Din_Speed4	20201420	0D40	RWS	Integer32	
Din_Speed5	20201520	0D50	RWS	Integer32	
Din_Speed6	20201620	0D60	RWS	Integer32	
Din_Speed7	20201720	0D70	RWS	Integer32	

8.5 Performance objects（0x6065）

Name	Subindex	modbus	RWS	Data type	Description
Max_Following_Error	60650020	3800	RWSM	Unsigned32	Following error alarm Default 10000inc
Target_Pos_Window	60670020	3900	RWS	Unsigned32	"Target_Pos_Reached" error range，default 10inc
Position_Window_time	25080916	1990	RW	Unsigned16	Target（Position、velocity）to time window，determine position to signal with 60670020
Target_Speed_Window	60F90A20	63A0	RWS	Integer32	The error window when actual speed reach to target speed or profile speed, determine speed to signal with 25080916
Zero_Speed_Window	20101810	0980	RWS	Unsigned16	Error window when actual speed is 0
Zero_Speed_Time	60F91410	6440	RWS	Unsigned16	When zero output speed window 0x201018 reach to specified range, it output zero speed signal after keep some time. Time is determined by zero speed output time.
Soft_Positive_Limit	607D0120	4410	RWS	Integer32	Software limit positive setting inc

Soft_Negative_Limit	607D0220	4420	RWS	Integer20	Software limit positive setting inc
Limit function	20101908	0990	RWS	Unsigned8	Define creating error or not after position limit 0: will be error if position limit happen after home found 1: do nothing

8.6 Home control（0x6098）

Name	CANopen	Modbus	RWS	Data type	Description
Homing method	60980008	4D00	RWSM	Integer8	Search homing Details in chapter “Homing control mode”
Homing speed switch	60990120	5010	RWSM	Unsigned32	Velocity for searching position_limit switch/home_switch signal
Homing speed zero	60990220	5020	RWSM		Velocity for searching home signal and Zero position
Homing acceleration	609A0020	5200	RWS	Unsigned32	Acceleration when search home rps/s
Homing offset	607C0020	4100	RWSM	Integer32	Offset after homing inc
Homing offset mode	60990508	5050	RWS	Unsigned8	Homing offset control mode 0：run to the homing offset point. The actual position will be 0 1：run to the home event happen point. The actual position will be "-homing offset"

8.7 Velocity loop（0x60F9）

Name	CANopen	Modbus	RWS	Data type	Description
Kvp	60F90110	6310	RW	Unsigned16	Bigger Kvp, bigger gain, but it may cause motor noise
Kvi	60F90210	6320	RW	Unsigned16	Bigger Kvi, bigger gain, but it may cause motor noise
Kvi/32	60F90710	6370	RWSL	Unsigned16	It is 1/31 of the normal kvi
Speed_Fb_N	60F90508	6350	RW	Unsigned8	Bandwidth of speed feedback filter BW=Speed_Fb_N*20+100[Hz]

8.8 Position loop（0x60FB）

Name	CANopen	modbus	RWS	Data type	Description
Kpp	60FB0110	6810	RWS	Unsigned16	Kpp of position loop
K_Velocity_FF	60FB0210	6820	RWS	Unsigned16	Velocity feedforward of position loop
K_Acc_FF	60FB0310	6830	RWS	Unsigned16	Acceleration feedforward of position loop
Pos_Filter_N	60FB0510	6850	RWS	Unsigned16	Modify in non-enable

8.9 Input & Output（0x2010）

Name	CANopen	modbus	RWS	Data type	Description
Din1_Function	20100310	0830	RWS	Unsigned16	Defined according to functions below
Din2_Function	20100410	0840	RWS	Unsigned16	
Din3_Function	20100510	0850	RWS	Unsigned16	
Din4_Function	20100610	0860	RWS	Unsigned16	
Dout1_Function	20100F10	08F0	RWS	Unsigned16	
Dout2_Function	20101010	0900	RWS	Unsigned16	
Din_Real	20100A10	08A0	RM	Unsigned16	bit0：Din1 bit1：Din2 bit2：Din3 bit3：Din4
Dout_Real	20101410	0940	RM	Unsigned16	bit0：Dout1 bit1：Dout2
Din_Polarity	20100110	0810	RWS	Unsigned16	0：Normal close；1：Normal open bit0：Din1 bit1：Din2 bit2：Din3 bit3：Din4 bit4：Din5 bit5：Din6 bit6：Din7 bit7：Din8 Default 0xFF

Dout_Polarity	20100D10	08D0	RWSM	Unsigned16	Define the polarity of Din signal
Din_Simulate	20100210	0820	RW	Unsigned16	bit0：Din1 bit1：Din2 bit2：Din3 bit3：Din4 bit4：Din5 bit5：Din6 bit6：Din7 bit7：Din8
Dout_Simulate	20100E10	08E0	RWM	Unsigned16	bit0：Dout1 bit1：Dout2 bit2：Dout3 bit3：Dout4 bit4：Dout5

Note

Din function（HEX）	Dout function（HEX）
0001：Enable	0001：Ready
0002：Reset Errors	0002：Error
0004：Operate mode sel	0004：Pos reached
0008：Kvi Off	0008：Zero speed
0010：P_Limit +	0010：Motor brake
0020：P_Limit -	0020：Speed reached
0040：Home signal	0040：Enc index
0080：Invert Direction	0080：Speed limit (Torque mode)
0100：Din Velocity Index0	0100：Driver enabled
0200：Din Velocity Index1	0200: position limiting
0400：Din Position Index0	0400: reference found
0800：Din Position Index1	0800: max current reached
1000：Quick stop	1000: multi DOUT 0
2000: Start homing	2000: multi DOUT 1
4000: Active command	4000: multi DOUT 2
8001: Din Velocity Index2	8001: STO active
8002: Din Position Index2	
8004: Multifunction 0（Configure Multi gear ratio）	
8008: Multifunction 1	
8010: Multifunction 2	
8020: Gain switch0	
8040: Gain switch1	
8080: MaxCur Switch	
8100: Motor Error	
8200: Pre Enable（IO must have enable signal. Otherwise, there will be an alarm. It is used in occasions which users run machine when ensure motor is safe.）	
8400: fast capture 1	
8800: fast capture 2	

8.10 Pulse input（0x2508）

Name	CANopen	modbus	RWS	Data type	Description
Gear_Factor0	25080110	0x1910	RWSM	Integer16	Gear factor 0
Gear_Divider0	25080210	0x1920	RWSM	Unsigned16	Gear divider 0
PD_CW	25080310	0x1930	RWSB	Integer16	0：CW/CCW 1：Pulse/Direction 2：A/B Mode 10：422 double pulse mode 11 :422 pulse-direction mode 12：422 incremental encoder mode
Gear Master	25080410	0x1940	RWM	Integer16	Master_encoder pulse input counting without gear ratio
Gear Slave	25080510	0x1950	RW	Integer16	Master_encoder pulse input counting with gear ratio
PD_Filter	25080610	0x1960	RWS	Unsigned16	Master_encoder pulse input filter
Master_Speed	25080C10	0x19C0	RM	Integer16	Master input pulse speed without gear ratio (pulse/mS)
Slave_Speed	25080D10	0x19D0	RW	Integer16	Master input pulse speed with gear ratio (pulse/mS)

8.11 Save（0x2FF0）

Name	Subindex	modbus	RWS	Data type	Description
Store_Data	2FF00108	2910	RW	Unsigned8	1：Store all parameters of control loop 10：Initiative all parameters of control loop Note :Store control loop parameters, not include motor parameters
Store_Motor_Data	2FF00308	2930	RW	Unsigned8	1 :Store the parameter relevant to motor

8.12 Error code（0x2601）

Name	CANopen	modbus	RWS	Data type	Description
Error_State	26010010	1F00	RM	Unsigned16	Error states bit0：Extended error bit 1：Encoder ABZ/not connected bit 2：Encoder UVW/Encoder internal bit 3：Encoder Counting/Encoder CRC bit 4：Driver temperature bit 5：Over voltage bit 6：Under voltage bit 7：Over current bit 8：Chop resistor bit 9：Position following bit 10：Low logic voltage bit 11：Motor or driver IIt bit 12：Over frequency bit 13：Motor temperature bit 14：Motor communication bit 15：EEPROM data

8.13 Stop

Name	CANopen	modbus	RWS	Data type	Description
Quick_Stop_Mode	605A0010	3400	RWS	Integer16	Limit switch, quick stop switch, or controlword is 0x000B 0： stop without control 1：stop by using ramp, then switch off 2 :stop by using quick stop deceleration, then switch off 5： stop with profile deceleration, stay in quick stop active 6：stop with quick stop deceleration, stay in quick stop active
Shutdown_Stop_Mode	605B0010	3410	RWS	Integer16	Shutdown stop mode（Cancel enable） 0：Stop without 1： Stop by using ramp, then switch off 2：Stop by using quick stop deceleration, then switch off
Disable_Stop_Mode	605C0010	3420	RWS	Integer16	0：Stop without 1： Stop by using ramp, then switch off 2：Stop by using quick stop deceleration, then switch off
Halt_Mode	605D0010	3430	RWS	Integer16	Controlword bit8 is 1 Make motor stop and enable 1：Stop by current ramp 2：Stop by quick stop deceleration
Fault_Stop_Mode	605E0010	3440	RWS	Integer16	Fault 0：Stop without control 1：Stop by using ramp, then switch off 2：Stop by using quick stop deceleration, then switch off
Profile_Dec	60840020	4C00	RWSM	Unsigned32	Deceleration in work mode 1 and 3
Quick_Stop_Dec	60850020	3300	RWS	Unsigned32	Deceleration for quick stop

Chapter 9 RS232

Driver can be used to configuration and adjustment by RS232 port (X3) . Port definition and communication protocol is described below :

9.1 RS232 wiring definition

If PLC or other controllers use RS485 port, it needs a RS485-to-RS232 module.

Console configuration line is a adapter cable between driver and driver. One side connects with RS232 of PC(DB9). The other side connects with RS232 of driver （RJ45port) .

Profile display below :

Figure 9-1 Serial port to RJ45 line

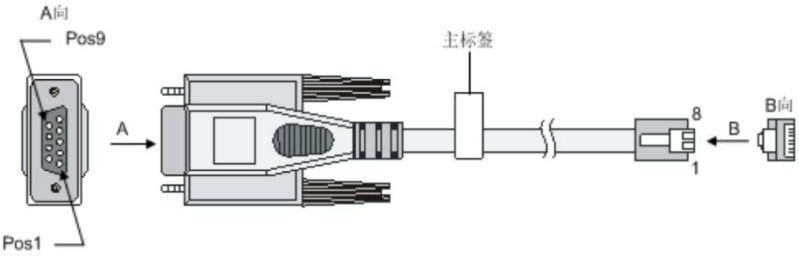


Figure 9-2 USB to serial port

9.1.1 Pin definition

Figure 0−3 DSub 9 of PC and X3 of driver

9.1.2 Multi-point connection

The communication protocol provides network operation with a host computer operating as a master and several CD3 controllers working as communication slaves （RS232_Loop_Enable(d5.15) must be set to1, save and reboot the controller after setting). In that case the RS232 cabling must have a loop structure as follows:

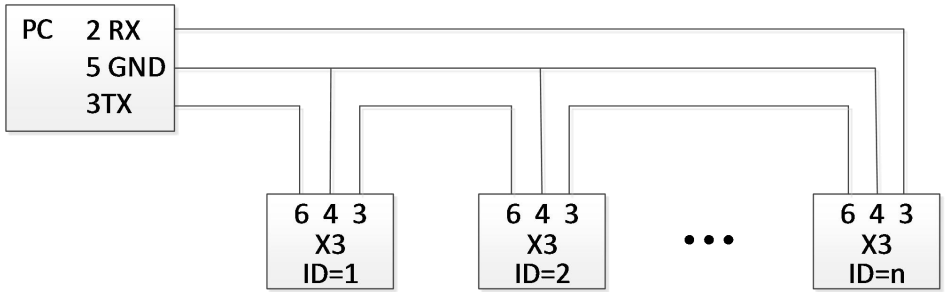


Figure 0–4 Cascaded wiring

9.2 Transport protocol

RS232 communication of motor controller strictly follows master / slave protocol. The host computer send data to controller. The controller checks the data regarding a checksum and the correct ID number, processes the data and returns an answer. Default communication settings for motor controller are as follows:

Baud rate：38400bps

Data bits：8

Stop bits：1

No parity check

The baud rate can be changed in RS232 BaudRate(d5.02). After changing the value it’ s necessary to save the setting and reboot the system.

The controller’ s ID can be changed in Node ID(d5.02).

The transport protocol uses a telegram with a fixed length of 10 bytes

Byte 0	Byte 1 ...Byte 8	Byte 9
ID	Data	CHKS

CHKS = -SUM(byte 0 …. byte 8)

9.2.1 Point-to-point protocol

One host communicates with one controller, RS232_Loop_Enable(d5.15)=0)

The host sends:

Byte 0	Byte 1 ...Byte 8	Byte 9
ID	Host data	CHKS

The salve sends/The host receives:

Byte 0	Byte 1 ...Byte 8	Byte 9
ID	Slave data	CHKS

If the slave finds it’ s own ID in the host telegram, it checks the CHKS value. If the checksum does not match the slave would not generate an answer and the host telegram would be discarded.

9.2.2 Multi-point protocol

One host communicates with several controllers, RS232_Loop_Enable(0x65100B08)=1

The host sends:

Byte 0	Byte 1 ...Byte 8	Byte 9
ID	Host data	CHKS

The slave sends / The host receives(RS232_Loop_Enable(0x65100B08)=1):

Byte 0	Byte 1 ...Byte 8	Byte 9
ID	Host data	CHKS

Byte 0	Byte 1 ...Byte 8	Byte 9
ID	Host data	CHKS

If the host sends a telegram with an unused ID data will pass the RS232 loop but no slave answer will return.The slave which finds it’ s own ID in the host telegram checks the CHKS value. If the checksum does not match the slave would not generate an answer and the host telegram would be discarded by that slave.

9.3 Data protocol

The data content of the transport protocol is the data protocol. It contains 8 bytes. The definition of the CD3 motor controller’s RS232 data protocol is compatible with the CANopen SDO

protocol, as well as the internal data organization complies to the CANopen standard. All parameters, values and functions are accessible via a 24-bit address, built of a 16-bit index and 8-bit sub-index.

9.3.1 Download (from host to slave)

Download means that the host sends a command to write values to the objects in the slave, the slave generates an error message if when the value is downloaded to a non-existent object.

The host sends:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
CMD	INDEX		SUB INEDX	DATA			

➔ **Note**

CMD: Specifies the direction of data transfer and the size of data.

23 (hex) Sends 4-byte data (bytes 4...7 contain 32 bits)

2b (hex) Sends 2-byte data (bytes 4 and 5 contain 16 bits)

2f (hex) Sends 1-byte data (bytes 4 contains 8 bits)

INDEX: Index in the object dictionary where data should be sent

SUB INDEX: Sub-index in object dictionary where data should be sent

DATA: 8, 16 or 32 bit value

The slave answer:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
CMD	INDEX		SUB INDEX	RESERVED			

➔ **Note**

RES: Displays slave response:

60(hex) Data successfully sent

80(hex) Error, bytes 4...7 contain error cause

INDEX: 16-bit value, copy of index in host telegram

SUBINDEX: 8-bit value, copy of sub index in host telegram

RESERVED: Not used

9.3.2 Upload (from slave to host)

Upload means the master sends a command to read the object value from the slave. The slave generates an error if a non-existent object is requested.

The master sends:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
CMD	INDEX		SUB INDEX	RESERVED			

➔ **Note**

CMD: Specifies the direction of data transfer

40(hex) always

INDEX: 16-bit value, index in the object dictionary where requested data reside.

SUBINDEX: 8-bit value, index, sub index in the object dictionary where requested data reside.

RESERVED: Bytes 4…7 not used

The slave answers:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
CMD	INDEX		SUB INDEX	DATA			

➔ **Note**

RES: Displays slave response:

43(hex) bytes 4...7 contain 32-bit data

4B(hex) bytes 4 and 5 contain 16-bit data

4F(hex) byte 4 contains 8-bit data

80(hex) error, bytes 4 … 7 contain error cause

INDEX: 16-bit value, copy of index in host telegram

SUBINDEX: 8-bit value, copy of subindex in host telegram

DATA: Data or error cause, depending on RES

9.4 RS232 telegram example

RS232 telegram example：

ID	R/W	Index	Sub index	Data	Checksum	Meaning
01	2B	40 60	00	2F 00 00 00	05	Set Controlword = 0x2F, enable the controller
01	2F	60 60	00	06 00 00 00	0A	Set Operation_Mode = 0x06
01	23	7A 60	00	50 C3 00 00	EF	Set Target_position = 50000
01	40	41 60	00	00 00 00 00	1E	Read the Statusword

Under each mode, it sends message, All of them use 1 as station number.

Home mode（Controlword F to 1F)				
CANOpen	Name	Value	Message（ID=1）	Meaning
60400010	Controlword	F	01 2B 40 60 00 0F 00 00 00 25	Homing_Speed_Switch and Homing_Speed_Zero fault unit is DEC，DEC=[(RPM*512*Encoder resolution)/1875]
60600008	Operation_mode	6	01 2F 60 60 00 06 00 00 00 0A	
60980008	Homing_Method	33	01 2F 98 60 00 21 00 00 00 B7	
60990120	Homing_Speed_Switch	200RPM	01 23 99 60 01 55 55 08 00 30	
60990220	Homing_Speed_Zero	150RPM	01 23 99 60 02 00 40 06 00 9B	
60400010	Controlword	1F	01 2B 40 60 00 1F 00 00 00 15	
01 40 41 60 00 00 00 00 00 1ERead state word，C037means home found				

Position mode（Controlword Absolute positioning 2F to 3F Relative positioning 4F to 5F，103F Start absolute positioning)				
CANOpen	Name	Value	Message（ID=1）	Meaning
60400010	Controlword	F	01 2B 40 60 00 0F 00 00 00 25	DEC=[(RPM*512*Encoder resolution)/1875]
60600008	Operation_mode	1	01 2F 60 60 00 01 00 00 00 0F	
607A0020	Target_Position	50000inc	01 23 7A 60 00 50 C3 00 00 EF	
60810020	Profile_Speed	200RPM	01 23 81 60 00 55 55 08 00 49	
60830020	Profile_Acc	610.352rps/s	Default	DEC=[(RPS/S*65536*Encoder
60840020	Profile_Dec	610.352rps/s	Default	

60400010	Controlword	2F	01 2B 40 60 00 2F 00 00 00 05	resolution)/1000/4000]
		3F(Absolute positioning)	01 2B 40 60 00 3F 00 00 00 F5	
		4F	01 2B 40 60 00 4F 00 00 00 E5	
		5F(Relative positioning)	01 2B 40 60 00 5F 00 00 00 D5	
01 40 41 60 00 00 00 00 00 1E		Read state word，D437 means position to		

Speed mode				
CANOpen	Name	Value	Message（ID=1）	Meaning
60600008	Operation_Mode	3	01 2F 60 60 00 03 00 00 00 0D	Default unit of Target speed DEC，DEC=[(RPM*512*Encoder Resolution)/1875] Default unit of Profile acc/dec DEC，DEC=[(RPS/S*65536*Encoder Resolution)/1000/4000]
60FF0020	Target_Speed	150RPM	01 23 FF 60 00 00 40 06 00 37	
60400010	Controlword	F	01 2B 40 60 00 0F 00 00 00 25	
60830020	Profile_Acc	610.352 rps/s	Default	
60840020	Profile_Dex	610.352 rps/s	Default	

Note

Under communication mode, data are transmitted in HEX.

Chapter 10 RS485 communication

10.1 RS485 wiring

FD1X3 RS485 port support RS485、RS422，this function can be used to modify modify servo internal parameters and monitor servo state, etc. Wiring is shown in figure 10-1..

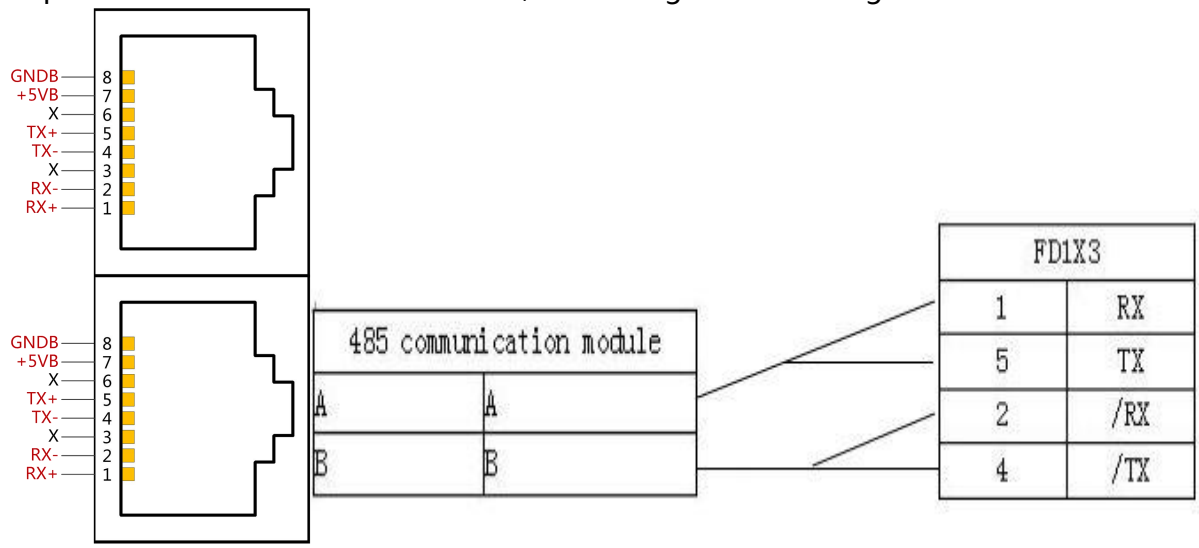


Figure0−1 RS485 Wiring

10.2 RS485 communication parameters

CANOpen	Name	Meaning	Default
100B0010	ID_Com	Station No. of Drivers	1
2FE20010	RS485 baud rate	Set baud rate of RS485 port 1080————9600 540————19200 270————38400 90————115200 Note: Save and reboot	540
65100C08	RS485 communication protocol selection	0：Modbus protocol 1：RS232 protocol Note: Set to 0, save and reboot	1

65100E10	RS485_Mode	data=8，stop=1，no check bit	default

10.3 MODBUS RTU

The RS485 interface of FD2S Servo driver supports Modbus RTU protocol. It's internal address is discontinuous 16-bit data register (R/W in software is 4X)

Modbus RTU protocol format：

Station No.	Function code	Data	CRC
1 byte	1 byte	N bytes	2 bytes

10.4 Function code of Modbus

- 0x03：read data registers

Request format:

Station No.	Function code	Modbus address		Address length (word)		CRC
		High byte	Low byte	High byte	Low byte	
1 byte	03	1 byte	1 byte	1 byte	1 byte	2 bytes

Normal response format：

Station No.	Function code	Return data length	Register data		.....	CRC
			High byte	Low byte		
1 byte	03	1 byte	1 byte	1 byte	.....	2 bytes

➔ **Note**

If there is error such as non-exist address,then it will return function code 0x81.

- Function code 0x06：write single data register

Request format:

Station No.	Function code	Modbus address		Writing value		CRC
		High byte	Low byte	High byte	Low byte	
1 byte	06	1 byte	1 byte	1 byte	1 byte	2 bytes

Response format:If writing successful,then return the same message.

➔ **Note**

If there is error such as address over range,non-exist address and the address is read only,then it will return function code 0x86.

- Function code 0x10：Write multiple registers

Request format：

Station No.	Function code	Modbus address	Data length（word）		Data length of data-in	Low byte of data		High byte of data		CRC
			High byte	Low byte		High byte	Low byte	High byte	Low byte	

1 byte	10	2 bytes	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	2 bytes

Normal response registers：

Station No.	Function code	Modbus address	Data length（word）		CRC
			High byte	Low byte	
1 byte	10	2 bytes	1 byte	1 byte	2 bytes

Note

If there is error such as address over range,non-exist address and the address is read only,then it will return function code 0x90.

For example：send message 01 10 6F 00 00 02 04 55 55 00 08 1A 47

Meaning：01——ID No.；

10——function code，write multiple WORD；

6F 00 — — WORD Modbus address for writing data. This is the address corresponding to parameter "Target Velocity"（60FF0020), data length 2；

00 02——write 2 WORD；

04——data length 4 bytes（2个WORD)；

55 55 00 08——write data 00085555 (HEX)，546133 (OCT)，convert to 200RPM；

1A 47——CRC check.

10.5 Modbus message example

Under different modes, message is sent when station No. is 1:

Table0−1485 message

Internal address	Name	Meaning	Message（ID=1）
3500	Operation_mode	Operate mode is 3	010635 0000 03C6 07
6F00	Targte_Speed	Speed 150RPM	01106F 0000 020455 5500 081A 47
3100	Controlword	Enable F	010631 0000 0FC7 32
3200	Status word	Read register status	01 03 32 00 00 02 CA B3

Home（Controlword F to 1F)				
Internal address	Name	Value	Message（ID=1）	Meaning
60400010	Controlword	F	010631 0000 0FC7 32	
60600008	Operation_Mode	6	010635 0000 0606 04	
60980008	Home_Method	33	01064D 0000 215E BE	
60990120	Home_Speed_Switch	200RPM	011050 1000 020455 5500 080E BA	
60990220	Home_Speed_Zero	150RPM	011050 2000 020440 0000 0698 76	
60400010	Controlword	1F	010631 0000 1FC6 FE	
010332 0000 02CA B3 read status word，C037 means home found				
Position（Controlword Absolute positioning 2F to 3F Relative positioning 4F to 5F，103F Start absolute positioning)				
Internal address	Name	Value	Message（ID=1）	Meaning
60400010	Controlword	F	010631 0000 0FC7 32	
60600008	Operation_Mode	1	010635 0000 0147 C6	
607A0020	Target_Position	50000inc	011040 0000 0204C3 5000 00FE 39	
60810020	Profile_Speed	200RPM	01104A 0000 020455 5500 08BC D6	

60830020	Profile_Acc	610.352rps/s	Default	
60840020	Profile_Dec	610.352rps/s	Default	
60400010	Controlword	2F	010631 0000 2FC6 EA	
		3F(Absolute position)	010631 0000 3FC7 26	
		4F	010631 0000 4FC6 C2	
		5F(Relative position)	010631 0000 5FC7 0E	
010332 0000 02CA B3read state word，D437 means position to				

Speed				
Internal address	Name	Value	Message（ID=1）	Meaning
60600008	Operation_Mode	3	010635 0000 03C6 07	
60FF0020	Target_Speed	150RPM	01106F 0000 020455 5500 081A 47	
60400010	Controlword	F	010631 0000 0FC7 32	
60830020	Profile_Acc	610.352rps/s	Default	
60840020	Profile_Dec	610.352rps/s	Default	

➔ **Note**

Under communication mode, data are transmitted in HEX.

➔ **Note**

In message above, if station No. is 00, message belongs to broadcast message. AllModbus slaves will accept this message.

Chapter 11 CANopen

11.1 CANopen communication protocol

CANopen is one of the most famous and successful open fieldbus standards.It has been widelyrecognized and applied a lot in Europe and USA. In 1992,CiA (CANinAutomation) was set up in Germany,and began to develop application layer protocol CANopen for CAN in automation. Since then, members of CiA developed a series of CANopen products,and applied in a large number of applications in the field of machinery manufacturing such as railway, vehicles, ships, pharmaceutical, food processing etc..Nowadays CANopen protocol has been the most important industrial fieldbus standard EN-50325-4 in Europe

The FD1X3 series servo supports standard CAN (slave device), strictly follow CANopen2.0A / B protocol, any host computer which support this protocol can communicate with it. FD1X3 Servo uses of a strictly defined object list, we call it the object dictionary, this object dictionary design is based on the CANopen international standards, all objects have a clear definition of the function. Objects said here similar to the memory address, we often say that some objects, such as speed and position,can be modified by an external controller, some object were modified only by the drive itself, such as status and error messages.

Table 0−1 Object dictionary example list

Complete CANOpen address			Attribute	Meaning
Index	Subindex	Bits(data length)		
0x6040	00	0x10	RW	Control word
0x6060	00	0x08	RW	Operation mode
0x607A	00	0x20	W	Target position
0x6041	00	0x10	MW	Status word

The attributes of objects are as follows:

1. RW(read&write)：The object can be both read and written;
2. RO(only read)：The object can be read only;
3. WO（only write）：The object can be written only;
4. M（map）：The object can be mapping,similar to indirect addressing;
5. S（save）：The object can be stored in Flash-ROM without lost after power failure;

11.2 Hardware Introduction

CAN communication protocol describes a way of transmitting information between devices, The definition of CAN layer is the same as the open systems interconnection model OSI, each layer communicates with the same layer in another device, the actual communication takes place adjacent layers in each device,but the devices only interconnect by the physical media of thephysical layer in the model.CAN standard defines data link layer and physical layer in the mode. The physical layer of CAN bus is not strictly required, it can use a variety of physical media such as twisted pair Fibre. The most commonly used is twisted pair signal, sent by differential voltage transmission (commonly used bus transceiver). The two signal lines are called CAN_H and CAN_L. The static voltage is approximately 2.5V, then the state is expressed as a logical 1, also called hidden bit. It represents a logic 0 when CAN_H is higher than the CAN_L, we called it apparent bit,then the voltage is that CAN_H = 3.5V and CAN_L= 1.5V,apparent bit is in high priority.

The standard CAN interface is shown in figure11-1.

Figure 0−1 FD1X3 CAN interface

➔ **Note**

This is pin definition of driver, not plugs.

Table0−2 Pin and function

Pin	Symbol	Meaning
1	CAN_H	CAN_H bus（high dominant）
2	CAN_L	CAN_L bus wire (low dominant)
3	CAN_GND	CAN ground
4	NC	Reserve
5	CAN_SHLD	Optional CAN shield
6	GND	Optional ground
7	NC	Reserve
8	NC	Reserve
9	CAN_V+	（NC）not connect

➔ **Note**

1、All CAN_L and CAN_H of slaves connect directly by using series connection, not star connection;

2、There must be connected a 120 ohm resistance in start terminal(master) and end terminal(slave)；

3、All FD3 Servo driver don’ t need external 24VDC supply for CAN interface；

4、 Please use the shield wires for communication cable,and make good grounding(Pin.3 is advised to grounding when communication is in long distance and high baudrate）；

5、The max. distance at different baudrate are shown in following table.

Table0–3The max. distance at different baudrate are shown in following table (Theory)

Communication speed（bit/s）	Communication distance（M）
1M	25
500K	100
250K	250
125K	500
50K	600

11.3 Software introduction

11.3.1 EDS introduction

EDS(Electronic Data Sheet)file is an identification documents or similar code of slave device,to identify what kind of slave device is(Like 401,402 and 403,or which device type of 402).This file includes all information of slaves,such as manufacturer,sequence No.,software version,supportable baudrate,mappable OD and attributes of each OD and so on,similar to the GSD file for Profibus.Therefore,we need to import the EDS file of slave into the software of master before we configure the hardware.

11.3.2 SDO introduction

SDO is mainly used in the transmit the low priority object between the devices, typically used to configure and mange the device,such as modifying PID parameters in current loop,velocity loop and position loop,and PDO configuration parameters and so on.This data transmission mode is the same as Modbus,that is it needs response from slave when master sends data to slave.This communication mode is suitable for parameters setting,but not for data transmission frequently.

SDO includes upload and download.The host can use special SDO instructions to read and write the OD of servo.In CANopen protocol, SDO (Service Data Object) can be used to modify object dictionary. SDO structure and guidelines are shown below:

SDO basic structure is：Client→Server/Server→Client

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x600+Node_ID	8	Send command word	Object index		Object subindex				

Receive SDO message when read parameters

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x580+Node_ID	8	Receive command	Object index		Object subindex	Max 4 bytes data			

➔ **Note**

When SDO message are sent, commands are 0x40;

When received data is 1 byte, received command is 0x4F;

When received data is 2 bytes, received command is 0x4B;

When received data is 4 bytes, received command is 0x43;

If received data have errors, received command is 0x80.

Send SDO message (Modify parameters)

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x600+Node_ID	8	Send command	Object index		Object subindex	Max 4 bytes data			

Receive SDO message (Modify parameters)

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x580+Node_ID	8	Receive command	Object index		Object subindex	Max 4 bytes data			

➔ **Note**

If sent data ready is 1 byte, command is 0x2F;

If sent data ready is 2 bytes, command is 0x2B;

If sent data ready is 4 bytes, command is 0x23;

If SDO message is sent successfully, receive command is 0x60;

If SDO message is not sent successfully, receive command is 0x80.

11.3.3 PDO introduction

PDO can transport 8 bytes of data at one time,and no other protocol preset(Mean the content

of the data are preset),it is mainly used to transmit data in high frequency.PDO uses brand new mode for data exchange,it needs to define the data receiving and sending area before the transmission between two devices,then the data will transmit to the receiving area of devices directly when exchanging data.It greatly increase the efficiency and utilization of the bus communication.

11.3.3.1 PDO COB-ID introduction

COB-ID is a unique way of CANopen communication protocol,it is the short name of Communication Object Identifier. These COB-ID defines the respective transmission levels for PDO, These transport level, the controller and servo will be able to be configured the same transmission level and the transmission content in the respective software.Then both sides know the contents of data to be transferred, there is no need to wait for the reply to check whether the data transmission is successful or not when transferring data.

The default ID allocation table is based on the CAN-ID(11 bits) defined in CANopen 2.0A (The COB-ID of CANopen 2.0B protocol is 27 bits) ,include function code(4 bits) and Node-ID(7 bits) as shown in diagram 11-2.

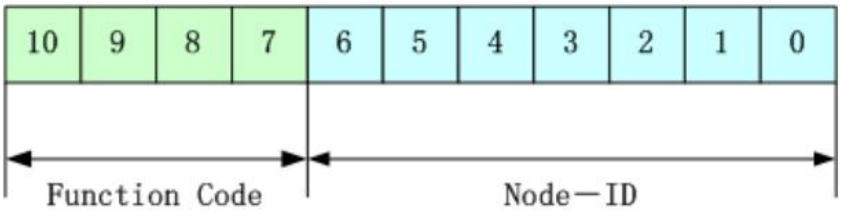


Figure0–2 Default ID allocation table

Note

Node-ID —— Servo station No. , Node-ID range is 1～127 ;

Function Code — — The function code for data transmission define the transmission level of PDO,SDO and management message.The smaller the function code,the higher the priority.

Table0–4The allocation table for CAN identifiers in master/slave connection set predefined by CANopen

Object	COB-ID
NMT Module Control	000H
SYNC	080H
TIME SSTAMP	100H
Object	COB-ID

Emergency	081H-0FFH
PDO1（Send）	181H-1FFH
PDO1（Receive）	201H-27FH
PDO2（Send）	281H-2FFH
PDO2（Receive）	301H-37FH
PDO3（Send）	381H-3FFH
PDO3（Receive）	401H-47FH
PDO4（Send）	481H-4FFH
PDO4（Receive）	501H-57FH
SDO（Send/Server）	581H-5FFH
SDO（Receive/Client）	601H-67FH
NMT Error Control	701H-77FH

➔ **Note**

1、The smaller the COB-ID,the higher the priority;

2、The function codes of COB-ID in every level are fixed;

3、COB-ID of 00H, 80H, 100H, 701H-77FH, 081H-0FFH are system management format;

11.3.3.2 COB-ID

Send PDO of servo means servo sends out data,and these data are received by PLC.The function codesof send PDO (COB-ID) are as follow:

1、0x180+Station No. of Servo

2、0x280+Station No. of Servo

3、0x380+Station No. of Servo

4、0x480+Station No. of Servo

Receive PDO of servo means servo receive data,and these data are sent by PLC.The function codes of receive PDO(COB-ID) are as follows:

1、0x200+Station No. of Servo

2、0x300+Station No. of Servo

3、0x400+Station No. of Servo

4、0x500+Station No. of Servo

➔ **Note**

FD2S Servo is designed according to the standard of CANopen 2.0A protocol,and it also supports CANopen 2.0B protocol.Therefore,if 8 PDOs are not enough,users can define new PDO,for example,set 0x43FH as the

communication PDO of Station No.1,but it needs the controllers and servo define PDO by the same rule.

1.3.3.3PDO transmission types

PDO support two transmission mode：

SYNC——Transmission is triggered by the synchronization message（Transmission type:0-240）

In this transmission mode, controller must have the ability to send synchronous messages（The message is sent periodically at a maximum frequency of 1KHz）,and servo will send after receiving the synchronous message.

Acyclic:Pre-triggered by remote frame,or by specific event of objects specified by the equipment sub-protocol.In this mode,servo will send out data as soon as receiving the data of synchronous message PDO.

Cyclic:Triggered after sending 1 to 240 SYNC messages.In this mode,servo will send out data in PDO after receiving n SYNC messages.

ASYNC (Transmission type：254/255)

Slave sends out message automatically as soon as the data change,and it can define an interval time between two messages which can avoid the one in high priority always sending message.(The smaller number of PDO,the higher its priority)

For FD1X3 series servo drivers, they supports 256 transmission methods. Users only need to use supporting method of transmission methods (controller) to choose driver's transmission methods.

➔ **Note**

Each PDO can define an inhibit time,that is the minimum interval time between two continuous PDO transmission.It is used to avoid the PDO in higher priority always occupying the communication.The inhibit time is 16bit unsigned integer,its unit is 100us.

11.3.3.3 Protection mode (Supervision)

Supervision type is to choose which way master uses to check slave during operation,and check whether slave is error or not and handle the error!

1、Heartbeat message

Heartbeat message:Slave send message to master cyclically during supervision time.If master hasn’ t received the message from slave after heartbeat time,then master will consider slave as

error!

Message format

(0x700+NodeID)+Status

Status：

0：Start 4:Stop 5:Run 127:Pre-operational

2、Node guarding

Slave send message to master cyclically during supervision time.If master hasn’ t received the message from slave after supervision time,then master will consider slave as error !

The format of master request message——（0x700+NodeID）（No data in this message）

Format of slave response message——（0x700+NodeID）+Status

The bit7 of the data is triggered bit.This bit will alternately set to 0 or 1 in the response message.It will be set to 0 at the first request of node guarding.The bit0 ~ bit6 indicate the status of node.

Status: 0:Initialization 1:No connection 2.Connection 3:Operational 4:Stop 5:Run 127:Pre-operational

Normally standard CAN slave only one protection mode,but FD2S Servo can support both.

11.3.3.4 Boot-up process

During the process of internet initialization, CANopen support extending boot-up and support min boot-up process. The boot-up process is shown in following figure11-3:

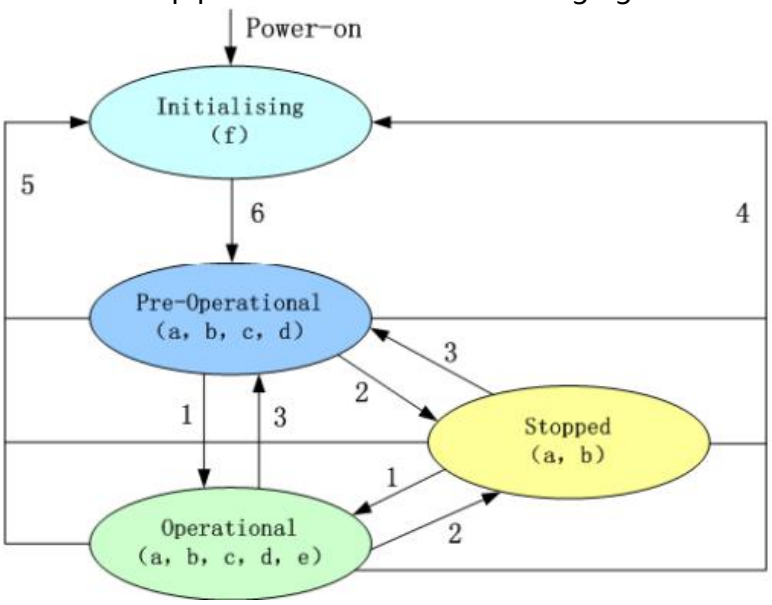


Figure0−3 Boot-up process

Table0−5 CANopen Internet process

Code	Meaning
a	NMT

b	Node Guard
c	SDO
d	Emergency
e	PDO
F	Boot-up

Management message format

COB-ID	DLC	Byte0	Byte1
0x000	02	CS	Station No.

When Node-ID is 0, all NMT slave devices are addressed. CS is command, value table is shown in table 11-6.

Table 0–6 CS Value table

Command	NMT service
0x01	Open node, start PDO transmission
0x02	Close node, end PDO transmission
0x80	Come to pre-operation status
0x81	Reset node
0x82	Reset communication

NMT management message can be used to change the modes.Only NMT-Master node can send NMT Module Control message, and all slave must support NMT Module Control service,meanwhile NMT Module Control message needn’ t response.

The format of NMT message is as follows：NMT-Master→NMT Slave(s)

➔ **Note：**

Only in operation status 0x5, PDO can be transmitted. If users want to open node which operation status is 6, then controllers can send messages below:

COB-ID	DLC	Byte0	Byte1
0x000	02	01	06

11.4 CANopen communication example

11.4.1 Connect to KincoServo+

When users need to configure CANopen communication parameters, they must use KincoServo+, which needs to be installed.

Download website：http://www.kinco.cn/download/software/servo

After installing software according to prompt, open software in figure 11-4:

Figure0–4

Click "communication" ->"communication setting". Set COM, COM ID. Baud rate is 38400, COM ID default is 1. If users don't know COM ID, they can use broadcast address 127. After configuration, click "OEPN".

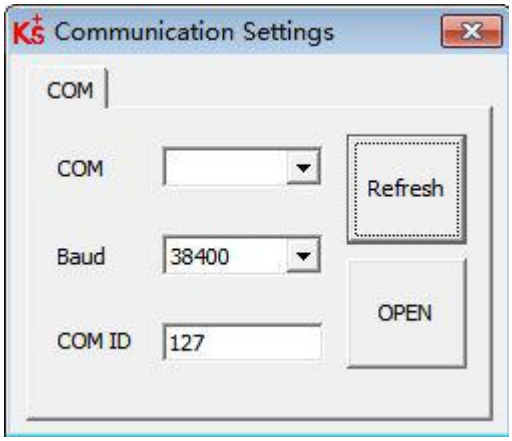


Figure0–5

Note

As driver follow the logic - IO control prior to communication control, so users need to cancel IO function when apply CANopen. If driver is enable or operation mode switch, communication cannot control servo control word,

operation mode, etc.

Note

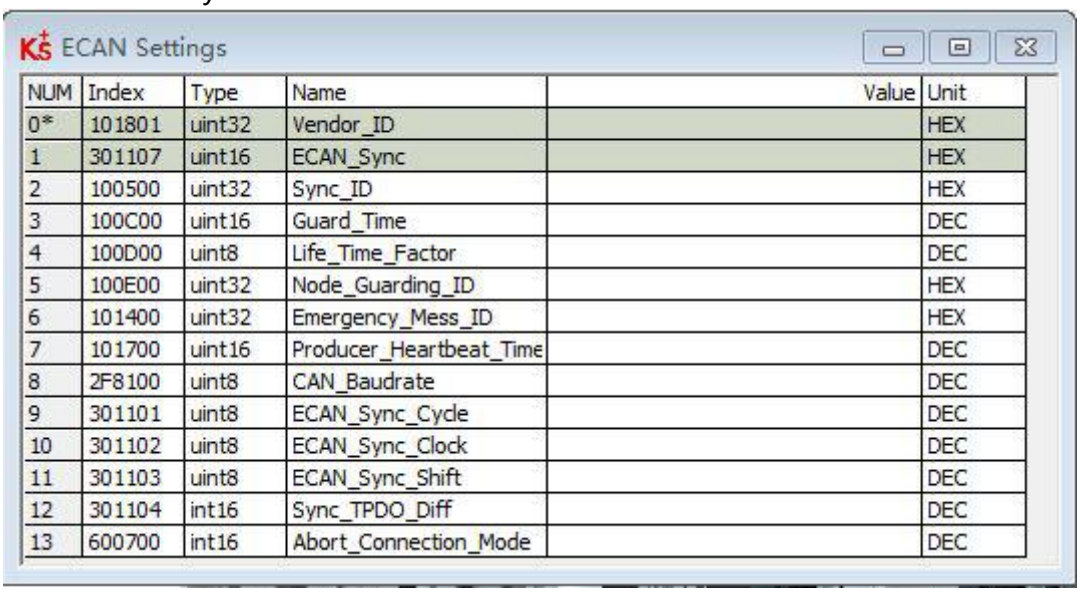
Use DIP switch to configure driver ID. Configuration method is shown in outside printing of driver.

11.4.2 Configure CANopen parameters

Note

For CANopen parameters, please check **driver → ECAN configuration→Others**

The maters which have function of network management can initialize slave parameters by sending SDO. Commonly, SYNC ID, node protection time, node protection time factor, node protection station No., emergency message station No., heartbeat message production time don’ t need to be set by users.



NUM	Index	Type	Name	Value	Unit
0*	101801	uint32	Vendor_ID		HEX
1	301107	uint16	ECAN_Sync		HEX
2	100500	uint32	Sync_ID		HEX
3	100C00	uint16	Guard_Time		DEC
4	100D00	uint8	Life_Time_Factor		DEC
5	100E00	uint32	Node_Guarding_ID		HEX
6	101400	uint32	Emergency_Mess_ID		HEX
7	101700	uint16	Producer_Heartbeat_Time		DEC
8	2F8100	uint8	CAN_Baudrate		DEC
9	301101	uint8	ECAN_Sync_Cycle		DEC
10	301102	uint8	ECAN_Sync_Clock		DEC
11	301103	uint8	ECAN_Sync_Shift		DEC
12	301104	int16	Sync_TPDO_Diff		DEC
13	600700	int16	Abort_Connection_Mode		DEC

Figure0−6 KincoServoCANopen configuration

Table0−7 CANopen parameters

CANopen address	Name	Description	Value
10050020	Sync_ID	Active when communication type is sync mode(1-240). No setting in async mode.	80
100C0010	Guard_Time	Nodes protect masters, it can monitor current state of every node. Masters (node protection time as period) send remote frame to	1000

CANopen address	Name	Description	Value
100D0008	Life_Time_Factor	check slave node state (Default COBID is 0x700+ID, message without content).Nodes need to response in a time. Otherwise, master nodes regard slave nodes off-line. Driver will have alarms.	3
100E0020	Node_Guard_ID	700+driver ID	
10140020	Emergency_Mess_ID	80+driver ID	
10170010	Producer_Heartbeat_Time	Slave nodes cyclically send master nodes. If master nodes don't receive message, master nodes will regard slave nodes off-line within a time.	
2F810008	CAN_Baudrate	CAN baudrate setting 100：1M 50：500k 25：250k 12：125k 5： 50k 1： 10k	50
30110108	ECAN_Sync_Cycle	In differential complement mode, ECAN_Sync_Cycle is configured according to master sync message period. Async mode don't need to be configured. 0:1ms 1:2ms 2:4ms 4:8ms	2
30110208	ECAN_Sync_Clock	In differential complement mode, ECAN_Sync_Clock (1) to open sync clock. In non-differential complement mode, ECAN_Sync_Clock (0) to close sync clock.	0
30110410	ECAN_TPDO_Diff	Under Sync mode, it monitor communication state. If value keep all the time, communication state is good. If value changes continuously, it means there is interface or sync period is wrong.	
60070010	Abort_Connection_Mode	CAN communication abort time，determine action logic when driver still do not receive node protection message over node protection time*node protection factor	0

CANopen address	Name	Description	Value
		0：no process 1：error	

Configure PDO by initialize PLC

For CANopen master which can load EDS, it is unnecessary to configure PDO in servo. And user can configure PDO in master directly. After power-on, PLC will initialize and send SDO message to configure servo' s PDO. When configuration is finished, master will send message to open slave. Then PDO communication can start. Most of PLCs CAN use this method. For example, Schneider PLC, Siemens S7-1200+CM CANOPEN module, Kinco F1, etc.

➔ EDS download address：
http://download.kinco.cn/D_Software/Servo/EDS.zip

Configure PDO by KincoServo

For part of PLCs, PDO parameters will to configured in servo driver, example is shown below:

Table0−8 Common control object

Name	CANopen	Address	RWS	Meaning
Controlword	0x60400010	2 bytes	RW	Control word
Operation_Mode	0x60600008	1 byte	RW	Operation mode
Target_Position	0x607A0020	4 bytes	W	Target position
Target_Speed	0x60FF0020	4 bytes	W	Target speed in speed mode
Profile_Speed	0x60810020	4 bytes	W	Speed in position mode
Statusword	0x60410010	2 bytes	R	Status word of drive
Pos_Actual	0x60630020	4 bytes	R	Actual position of motor

TPDO in servo is: (Servo send to PLC)

TPDO1: Actual position + status word

RPDO in servo is: (PLC send to servo)

RPDO1: Target position+Operation mode+Control word;

RPDO2: Target speed+profile speed;

➔ Overall object length in each PDO is not more than 8 bytes.

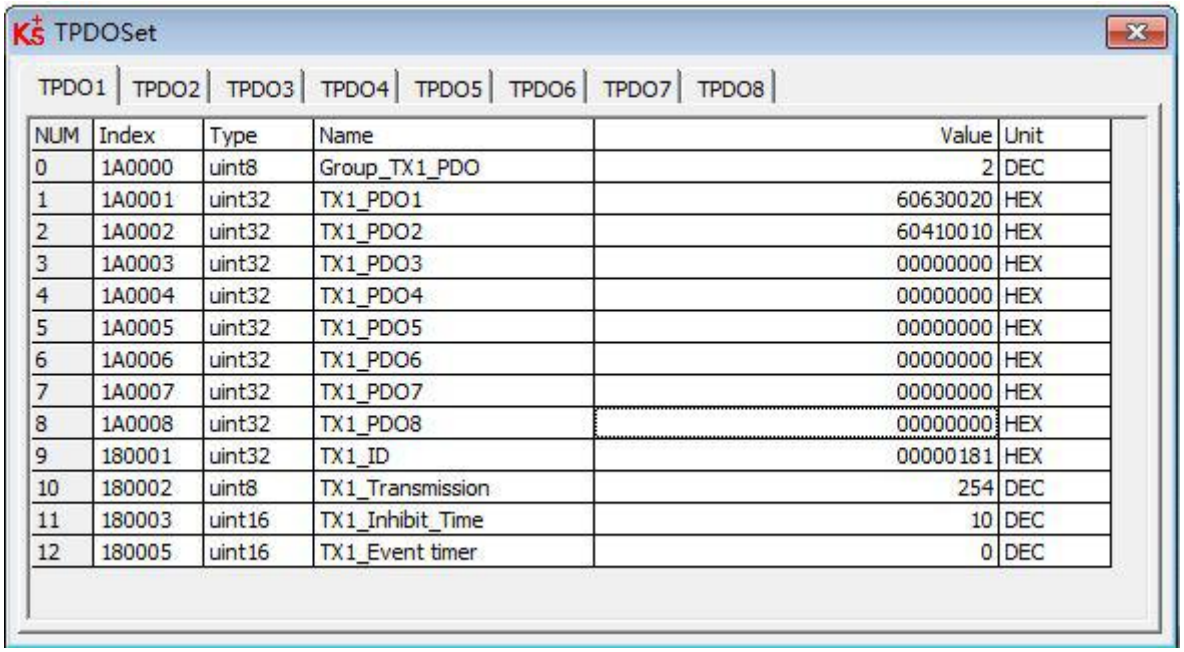
Before using PDO to transmit data, it is necessary to send message open node of management message. Take No. 2 station for example

COBID	DLC	Message
000	02	01 02

11.4.3 PDO transmission mode configuration

Async transmission mode

In Async transmission mode, mapping data in PDO will transmit once they change.



NUM	Index	Type	Name	Value	Unit
0	1A0000	uint8	Group_TX1_PDO	2	DEC
1	1A0001	uint32	TX1_PDO1	60630020	HEX
2	1A0002	uint32	TX1_PDO2	60410010	HEX
3	1A0003	uint32	TX1_PDO3	00000000	HEX
4	1A0004	uint32	TX1_PDO4	00000000	HEX
5	1A0005	uint32	TX1_PDO5	00000000	HEX
6	1A0006	uint32	TX1_PDO6	00000000	HEX
7	1A0007	uint32	TX1_PDO7	00000000	HEX
8	1A0008	uint32	TX1_PDO8	00000000	HEX
9	180001	uint32	TX1_ID	00000181	HEX
10	180002	uint8	TX1_Transmission	254	DEC
11	180003	uint16	TX1_Inhibit_Time	10	DEC
12	180005	uint16	TX1_Event timer	0	DEC

Figure0−7TPDO configuration in Async mode

Table0−9 TPDO configuration in Async transmission mode

Name	Meaning
TPDO1 mapping group	Paired object No. in this PDO. In TPDO1, actual position and status word are configured.
Map 1-8	Configure servo CANopen control object
TPDO1 station No.	180+driver ID (TPDO2 station No.:280+driver ID)
TPDO1 transmission type	254 or 255, Async transmission mode
TPDO1 prohibited time	Unit is ms, in case that frequency of message sent by servo is too high so as to block network, it is indeed configured in multi-shaft async transmission mode

Note

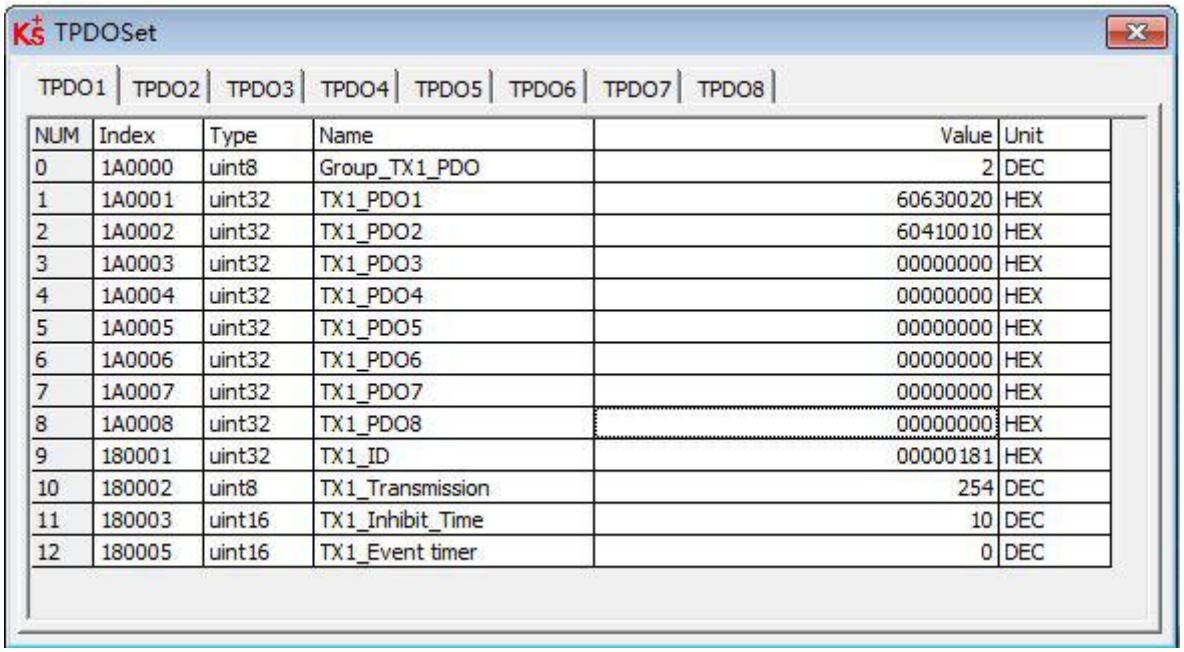
In TPDO1, overall length of paired object actual position and status word is

4+2=6

Default of RPDO transmission mode is 254. Users do not need to set. After receiving data, it is immediate effect.

Event time timing reported function

In sync transmission mode, except for (transmission once change), if driver upload data to controller periodically , event time can be configured.



NUM	Index	Type	Name	Value	Unit
0	1A0000	uint8	Group_TX1_PDO	2	DEC
1	1A0001	uint32	TX1_PDO1	60630020	HEX
2	1A0002	uint32	TX1_PDO2	60410010	HEX
3	1A0003	uint32	TX1_PDO3	00000000	HEX
4	1A0004	uint32	TX1_PDO4	00000000	HEX
5	1A0005	uint32	TX1_PDO5	00000000	HEX
6	1A0006	uint32	TX1_PDO6	00000000	HEX
7	1A0007	uint32	TX1_PDO7	00000000	HEX
8	1A0008	uint32	TX1_PDO8	00000000	HEX
9	180001	uint32	TX1_ID	00000181	HEX
10	180002	uint8	TX1_Transmission	254	DEC
11	180003	uint16	TX1_Inhibit_Time	10	DEC
12	180005	uint16	TX1_Event timer	0	DEC

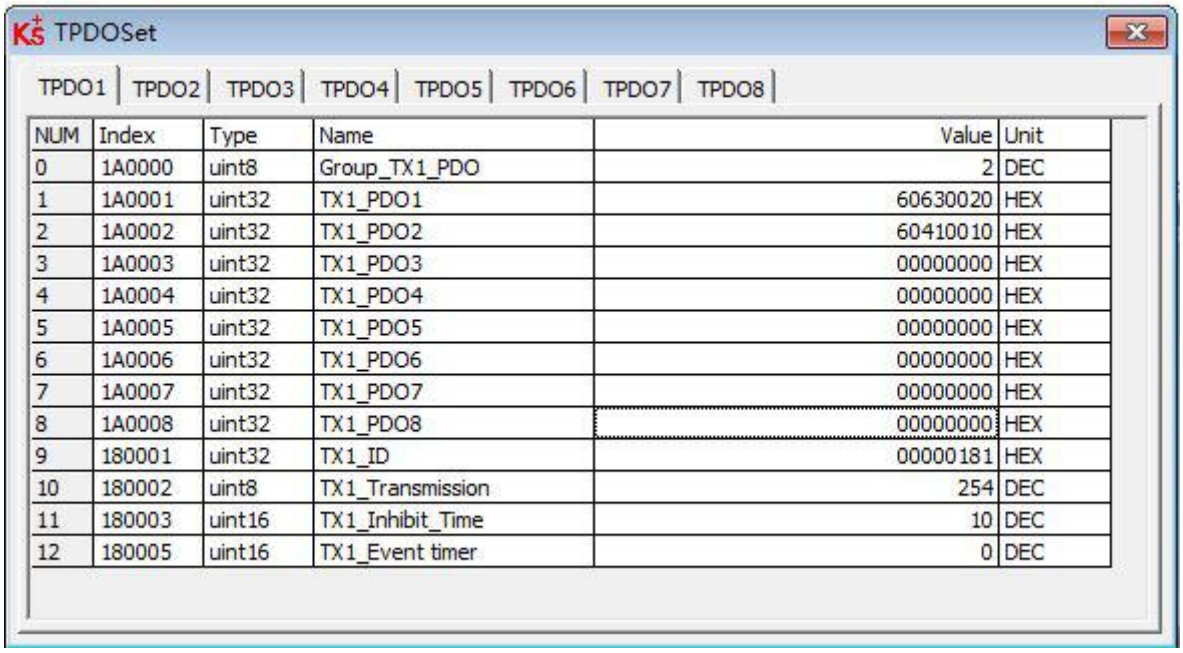
Figure0−8 Event time timing reported in async mode

Table0−10 Event time timing reported in async mode

Name	Meaning
TPDO1 mapping group	Paired object No. in this PDO. In TPDO1, actual position and status word are configured.
Map 1-8	Configure servo CANopen control object
TPDO1 station No.	180+driver ID (TPDO2 station No.:280+driver ID)
TPDO1 transmission type	254 or 255，Async transmission mode
TPDO1 prohibited time	254 or 255, Async transmission mode
TPDO1 event time	Period (Driver send PDO to controller), unit is ms
Note: In TPDO1, overall length of paired object actual position and status word is 4+2=6	

Sync transmission mode

When CANopen communication configuration is sync transmission mode, mapping data in TPDO will be upload after driver receive sync message.



NUM	Index	Type	Name	Value	Unit
0	1A0000	uint8	Group_TX1_PDO	2	DEC
1	1A0001	uint32	TX1_PDO1	60630020	HEX
2	1A0002	uint32	TX1_PDO2	60410010	HEX
3	1A0003	uint32	TX1_PDO3	00000000	HEX
4	1A0004	uint32	TX1_PDO4	00000000	HEX
5	1A0005	uint32	TX1_PDO5	00000000	HEX
6	1A0006	uint32	TX1_PDO6	00000000	HEX
7	1A0007	uint32	TX1_PDO7	00000000	HEX
8	1A0008	uint32	TX1_PDO8	00000000	HEX
9	180001	uint32	TX1_ID	00000181	HEX
10	180002	uint8	TX1_Transmission	254	DEC
11	180003	uint16	TX1_Inhibit_Time	10	DEC
12	180005	uint16	TX1_Event timer	0	DEC

Figure0–9 TPDO configuration in sync mode

Table 0–11TPDO configuration in sync mode

Name	Meaning
TPDO1 mapping group	2, No. Of paired objects in this PDO, actual position and status word configured in TPDO1
Map 1-8	Configure servo CANopen control object
TPDO1 station No.	80+driver ID (TPDO2 station No.:280+driver ID)
TPDO1 transmission type	Sync transmission mode, driver send TPDO after receiving sync message
TPDO1 prohibited time	0
Note：In TPDO1, overall length of paired object actual position and status word is 4+2=6	

➔ **Note**

In TPDO1, overall length of paired object actual position and status word is 4+2=6

Default of RPDO transmission mode is 254. Users do not need to set. After receiving data, it is immediate effect.

➔ **Note:**

Default sync message is

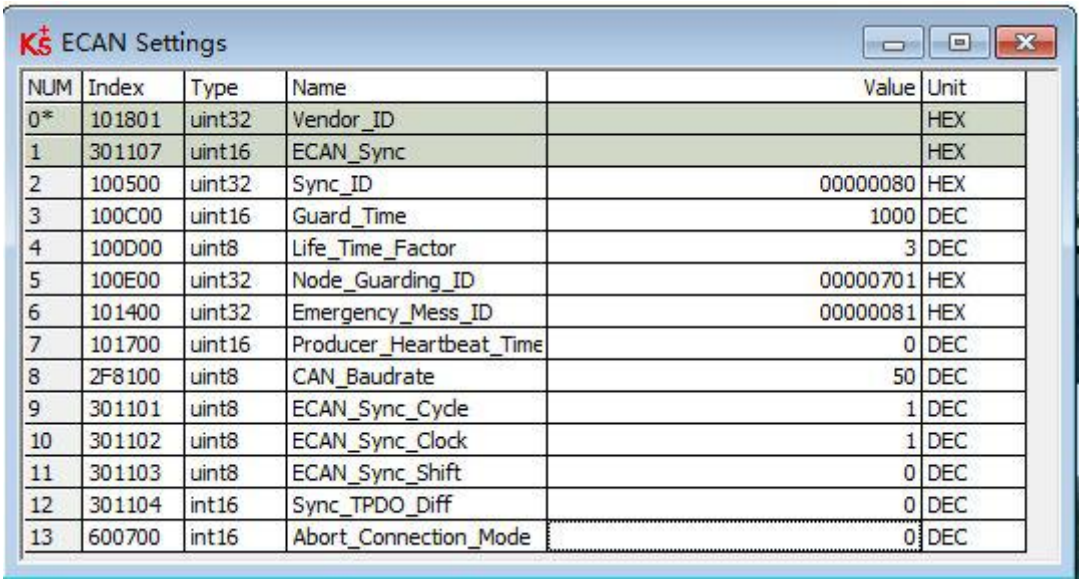
COB-ID	DLC
0x80	0

Differential complement mode based on CANopen

In differential complement mode based on CANopen, operation mode (0x60600008) is 7.

Diagram0−12 Differential complement operation mode is 7

Name	CANopen address	Length	RWS	Meaning
Operation_Mode	60600008	1 byte	RW	Operation mode



NUM	Index	Type	Name	Value	Unit
0*	101801	uint32	Vendor_ID		HEX
1	301107	uint16	ECAN_Sync		HEX
2	100500	uint32	Sync_ID	00000080	HEX
3	100C00	uint16	Guard_Time	1000	DEC
4	100D00	uint8	Life_Time_Factor	3	DEC
5	100E00	uint32	Node_Guarding_ID	00000701	HEX
6	101400	uint32	Emergency_Mess_ID	00000081	HEX
7	101700	uint16	Producer_Heartbeat_Time	0	DEC
8	2F8100	uint8	CAN_Baudrate	50	DEC
9	301101	uint8	ECAN_Sync_Cycle	1	DEC
10	301102	uint8	ECAN_Sync_Clock	1	DEC
11	301103	uint8	ECAN_Sync_Shift	0	DEC
12	301104	int16	Sync_TPDO_Diff	0	DEC
13	600700	int16	Abort_Connection_Mode	0	DEC

Figure 0−10 Parameters in differential complement mode

Table0−13 Parameters in differential complement mode

CANopen address		Name	Meaning	Default

30110108		ECAN_Sync_Cycle	In differential complement mode, ECAN_Sync_Cycle is configured according to master sync message period. Async mode don't need to be configured. 0: 1ms 1: 2ms 2: 4ms 4: 8ms	2
30110208		ECAN_Sync_Clock	In differential complement mode, ECAN_Sync_Clock (1) to open sync clock. In non-differential complement mode, ECAN_Sync_Clock (0) to close sync clock.	0

CAN communication interruption alarm function

For CAN communication interruption alarm function, parameters below need to be configured

Table0−14 Communication interruption alarm function configuration

CANopen address	Name	Meaning	Default
100C0010	Guard_Time	Nodes protect masters, it can monitor current state of every node. Masters (node protection time as period) send remote frame to check slave node state (Default COBID is 0x700+ID, message without content).Nodes need to response in a time. Otherwise, master nodes regard slave nodes off-line. Driver will have alarms.	1000
100D0008	Life_Time_Factor		3
100E0020	Node_Guarding_ID	700+driver ID	
10140020	Emergency_Mess_ID	80+driver station No.	
60070010	Abort_Connection_Mode	CAN communication abort time，determine action logic when driver still do not receive node protection message over node protection time*node protection factor 0：no process 1：Error	0

11.4.4 CANopen send message example

Node protection message and heartbeat message

CANopen node will transmit heartbeat message at a fixed frequency, which is used to inform controller that communication connection is normal. Format of message is simple. COB-ID is 0x700+Node_ID

No	Time	CAN channel	Transmission direction	ID	Frame type	Frame format	Data length	Data
0	16:50:14:031	0	Receive	0706	Data frame	Standard frame	1	7F
1	16:50:15:093	0	Receive	0706	Data frame	Standard frame	1	7F
2	16:50:16:171	0	Receive	0706	Data frame	Standard frame	1	7F
3	16:50:17:234	0	Receive	0706	Data frame	Standard frame	1	7F
4	16:50:18:296	0	Receive	0706	Data frame	Standard frame	1	7F
5	16:50:19:375	0	Receive	0706	Data frame	Standard frame	1	7F
6	16:50:20:437	0	Receive	0706	Data frame	Standard frame	1	7F
7	16:50:21:500	0	Receive	0706	Data frame	Standard frame	1	7F
8	16:50:22:578	0	Receive	0706	Data frame	Standard frame	1	7F
9	16:50:23:640	0	Receive	0706	Data frame	Standard frame	1	7F
10	16:50:24:718	0	Receive	0706	Data frame	Standard frame	1	7F
11	16:50:25:781	0	Receive	0706	Data frame	Standard frame	1	7F
12	16:50:26:859	0	Receive	0706	Data frame	Standard frame	1	7F
13	16:50:27:921	0	Receive	0706	Data frame	Standard frame	1	7F
14	16:50:29:000	0	Receive	0706	Data frame	Standard frame	1	7F
15	16:50:30:062	0	Receive	0706	Data frame	Standard frame	1	7F

Figure0−11 Node message and heartbeat message

In diagram above, ID is 706, which is the heartbeat of 06 node and state is 0x7F. It is Pre-Operational state (After node initialization finishes, it comes to Pre-Operational state). By checking time, time interval between each heartbeat message is about 1s.

NMT management message

NMT is management message, used to achieve some management operations such as node open, enter-in Operational state, etc. Format of NMT message is simple and ID is 000. Data consists of one byte command and one byte node number (0 is broadcast)

No	Time	CAN channel	Transmission direction	ID	Frame type	Frame format	Data length	Data
0	16:51:35:296	0	Receive	0706	Data frame	Standard frame	1	7F
1	16:51:36:375	0	Receive	0706	Data frame	Standard frame	1	7F
2	16:51:37:796	0	Send	0000	Data frame	Standard frame	2	01 06
3	16:51:38:437	0	Receive	0706	Data frame	Standard frame	1	7F
4	16:51:39:500	0	Receive	0706	Data frame	Standard frame	1	05
5	16:51:40:562	0	Receive	0706	Data frame	Standard frame	1	05
6	16:51:41:625	0	Receive	0706	Data frame	Standard frame	1	05
7	16:51:42:687	0	Receive	0706	Data frame	Standard frame	1	05

Figure0−12 Open node

The second message can make 06 node enter in Operational state. After running, node state in node heartbeat message becomes Operational state. Under this state, PDO start to transmit.

No	Time	CAN channel	Transmission direction	ID	Frame type	Frame format	Data length	Data
0	16:51:47:843	0	Receive	0706	Data frame	Standard frame	1	05
1	16:51:48:906	0	Receive	0706	Data frame	Standard frame	1	05
2	16:51:49:968	0	Receive	0000	Data frame	Standard frame	1	05
3	16:51:50:031	0	Receive	0706	Data frame	Standard frame	1	05
4	16:51:51:578	0	Send	0000	Data frame	Standard frame	2	02 06
5	16:51:52:109	0	Receive	0706	Data frame	Standard frame	1	05
6	16:51:53:156	0	Receive	0706	Data frame	Standard frame	1	04
7	16:51:54:218	0	Receive	0706	Data frame	Standard frame	1	04
8	16:52:55:281	0	Receive	0706	Data frame	Standard frame	1	04

Figure0−13 Close node

Send stop remote node command and go to Stopped state. Obviously, heartbeat still exist and only node doesn’ t work.

No	Time	CAN channel	Transmission direction	ID	Frame type	Frame format	Data length	Data
0	16:53:58:890	0	Receive	0706	Data frame	Standard frame	1	04
1	16:53:59:953	0	Receive	0706	Data frame	Standard frame	1	04
2	16:54:00:375	0	Send	0000	Data frame	Standard frame	2	81 06
3	16:54:01:015	0	Receive	0706	Data frame	Standard frame	1	04
4	16:54:02:093	0	Receive	0706	Data frame	Standard frame	1	00
5	16:54:03:156	0	Receive	0706	Data frame	Standard frame	1	7F
6	16:54:04:218	0	Receive	0706	Data frame	Standard frame	1	7F

Figure0−14 Reset node

This is reset node command, which is used to make node reset. After reset, it enters in Initializing node (for 0x00 in heartbeat message). After initialization is finished, it enters in Pre-Operational state (for 0x7F in heartbeat message).

Send and receive SDO

SDO is mainly used to visit node’ s OD. Node in CANopen at least need to support SDO_Server. OD is data organization of CANopen node, which includes each parameter and data of CANopen node such as sending frequency of heartbeat frequency, system start times, node communication parameters and etc. So SDO is used to set each running parameter of CANopen node.

No	Time	CAN channel	Transmission direction	ID	Frame type	Frame format	Data length	Data
0	16: 55: 24: 421	0	Receive	0706	Data frame	Standard frame	1	05
1	16: 55: 25: 281	0	Send	0606	Data frame	Standard frame	8	40 17 10 00 00 00 00 00
2	16: 55: 25: 500	0	Receive	0706	Data frame	Standard frame	1	05
3	16: 55: 25: 500	0	Receive	0706	Data frame	Standard frame	8	4B 17 10 00 E8 03 00 00
4	16: 55: 26: 562	0	Receive	0706	Data frame	Standard frame	1	05

Figure0−15 Send SDO message and read data

In diagram 11-17, No.1 message 0606:40 17 10 00 00 00 00 00 is a SDO_Read message, which can tell node to read OD’ s index, subindex and also data length. Then node will send corresponding data (8 bytes message behind). The first byte is command. The second and third byte are OD’ s main address. The fourth byte is OD’ s subaddress. The last four bytes are data. In diagram above, master send data command to read OD (in 1017:00). This position stores heartbeat frequency and result of read is 0x03EB (1000ms).

No	Time	CAN channel	Transmission direction	ID	Frame type	Frame format	Data length	Data
0	17:09:35:828	0	Receive	0706	Data length	Standard frame	1	05
1	17:09:36:921	0	Receive	0706	Data length	Standard frame	1	05
2	17:09:38:015	0	Receive	0706	Data length	Standard frame	1	05
3	17:09:39:109	0	Receive	0706	Data length	Standard frame	1	05
4	17:09:40:187	0	Receive	0706	Data length	Standard frame	1	05
5	17:09:41:281	0	Receive	0706	Data length	Standard frame	1	05
6	17:09:42:375	0	Receive	0706	Data length	Standard frame	1	05
7	17:09:43:453	0	Receive	0706	Data length	Standard frame	1	05
8	17:09:44:546	0	Receive	0706	Data length	Standard frame	1	05
9	17:09:45:437	0	Send	0606	Data length	Standard frame	8	2B 17 10 00 FF 01 00 00
10	17:09:45:640	0	Receive	0706	Data length	Standard frame	1	05
11	17:09:45:640	0	Receive	0566	Data length	Standard frame	8	60 17 10 00 00 00 00 00
12	17:09:46:187	0	Receive	0706	Data length	Standard frame	1	05
13	17:09:46:734	0	Receive	0706	Data length	Standard frame	1	05
14	17:09:47:265	0	Receive	0706	Data length	Standard frame	1	05
15	17:09:47:812	0	Receive	0706	Data length	Standard frame	1	05
16	17:09:48:359	0	Receive	0706	Data length	Standard frame	1	05
17	17:09:48:906	0	Receive	0706	Data length	Standard frame	1	05
18	17:09:49:453	0	Receive	0706	Data length	Standard frame	1	05
19	17:09:50:000	0	Receive	0706	Data length	Standard frame	1	05

Figure0−16 Send SDO message and modify data

No.9 message is SDO_Write, which writes data in OD (1017:00). It is also to modify heartbeat frequency. After receiving response, heartbeat frequency changes.

Send and receive data message in different mode (Station No. is 1)

Homing（Controlword F to 1F)				
CANopen	Name	Value	Send and reply message（ID=1）	Meaning
60400010	Controlword	F	6012B40 60000F 00 5816040 60000F 00	
60600008	Operation_Mode	6	6012F60 600006 00 5816060 600006 00	
60980008	Homing_Method	33	6012F98 600021 00 5816098 600021 00	
60990120	Homing_Speed_Switch	200RPM	6012399 600155 55 08 00 5816099 600155 55 08 00	
60990220	Homing_Speed_Zero	150RPM	6012399 600200 40 06 00 5816099 600200 40 06 00	
60400010	Controlword	1F	6012B40 60001F 00 5816040 60001F 00	
6014041 600000 00 00 00read status word，C037means home found				

Position（Controlword Absolute positioning 2F to 3F Relative positioning 4F to 5F，103F Start absolute positioning)				
CANopen	Name	Value	Message（ID=1）	Meaning
60400010	Controlword	F	6012B40 60000F 00 5816040 60000F 00	
60600008	Operation_Mode	1	6012F60 600001 00 5816060 600001 00	
607A0020	Target_Position	50000inc	601237A 600050 C3 00 00 581607A 600050 C3 00 00	
60810020	Profile_Speed	200RPM	6012381 600055 55 08 00 5816081 600055 55 08 00	
60830020	Profile_Acc	610.352rps/s	Default	
60840020	Profile_Dcc	610.352rps/s	Default	
60400010	Controlword	2F	6012B40 60002F 00 5816040 60002F 00	
		3F(absolute positioning)	6012B40 60003F 00 5816040 60003F 00	
		4F	6012B40 60004F 00 5816040 60004F 00	
		5F(relative positioning)	6012B40 60005F 00 5816040 60005F 00	
6014041 600000 00 00 00read status word，D437means position found				

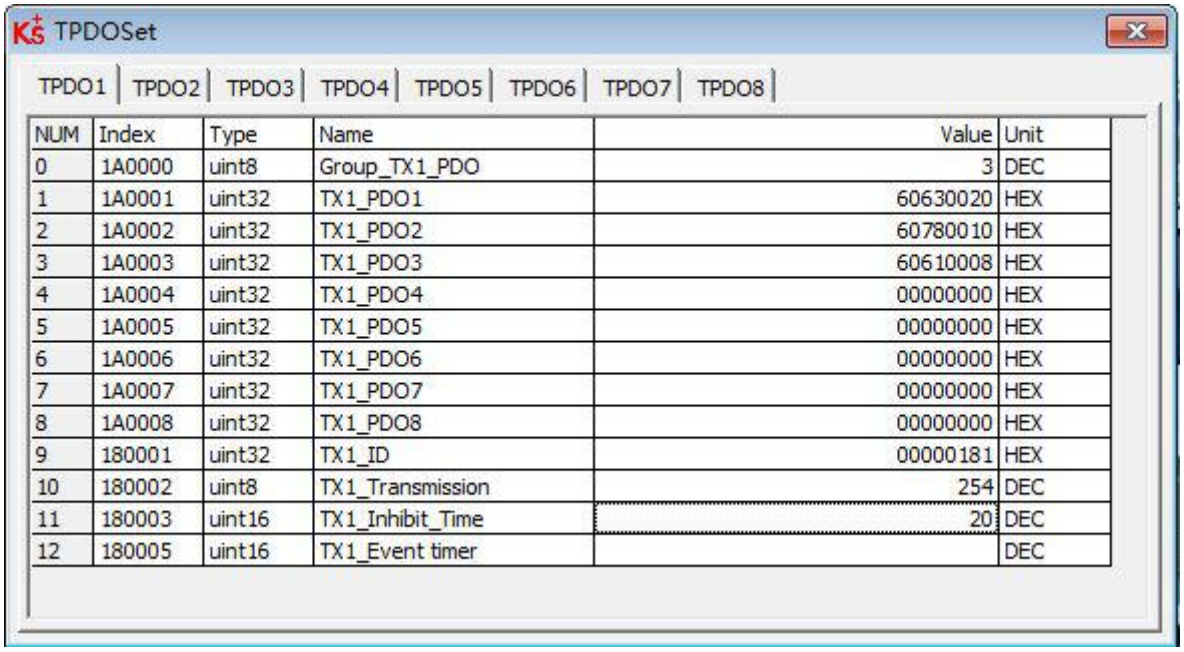
Speed				
CANopen	Name	Value	Message（ID=1）	Meaning
60600008	Operation_Mode	3	6012F60 600003 00 5816060 600003 00	
60FF0020	Target_Speed	150RPM	60123FF 600000 40 06 00 58160FF 600000 40 06 00	

60400010	Controlword	F	6012B40 60000F 00 5816040 60000F 00	
60830020	Profile_Acc	Default 610.352rps/s	Default	
60840020	Profile_Dcc	Default 610.352rps/s	Default	

Note：PDO Under communication mode, data are transmitted, sent and received in HEX.

SDO Send and received SDO

PDO is used to send (TPDO) and receive (RPDO) data. There are different triggering methods such as sync transmission and async transmission. PDO’ s data content is defined in OD by mapping. One node can have multiple PDO channels. PDO’ s communication parameters can be modified by which SDO visits OD.



NUM	Index	Type	Name	Value	Unit
0	1A0000	uint8	Group_TX1_PDO	3	DEC
1	1A0001	uint32	TX1_PDO1	60630020	HEX
2	1A0002	uint32	TX1_PDO2	60780010	HEX
3	1A0003	uint32	TX1_PDO3	60610008	HEX
4	1A0004	uint32	TX1_PDO4	00000000	HEX
5	1A0005	uint32	TX1_PDO5	00000000	HEX
6	1A0006	uint32	TX1_PDO6	00000000	HEX
7	1A0007	uint32	TX1_PDO7	00000000	HEX
8	1A0008	uint32	TX1_PDO8	00000000	HEX
9	180001	uint32	TX1_ID	00000181	HEX
10	180002	uint8	TX1_Transmission	254	DEC
11	180003	uint16	TX1_Inhibit_Time	20	DEC
12	180005	uint16	TX1_Event timer		DEC

Figure0−17 Content in TPDO1 mapping group

In TPDO1, it map 3 objects, actual position, actual current and effective work mode in turn. Then from cutting PDO message, actual position is 0xFaE84270.

```
835)      6560.6  Rx        0186  7  70 42 E8 FA 00 00 00
836)      6560.7  Rx        0181  7  70 42 94 5B 00 00 00
837)      6560.8  Rx        0182  7  70 42 23 C8 00 00 00
838)      6560.9  Rx        0183  7  70 42 25 11 00 00 00
839)      6561.0  Rx        0184  7  70 42 D5 E7 00 00 00
840)      6561.1  Rx        0185  7  70 42 3A 41 00 00 00
841)      6561.2  Rx        0287  8  20 9C FF FF 00 00 00 00
842)      6561.4  Rx        0187  7  70 C2 20 9C FF FF 00
843)      6562.4  Rx        0201  6  00 00 94 5B 00 00
844)      6562.5  Rx        0202  6  00 00 23 C8 00 00
845)      6562.6  Rx        0203  6  00 00 25 11 00 00
846)      6562.7  Rx        0204  6  00 00 D5 E7 00 00
847)      6562.8  Rx        0205  6  00 00 3A 41 00 00
848)      6562.9  Rx        0206  6  00 00 E8 FA 00 00
849)      6563.0  Rx        0207  6  00 00 20 9C FF FF
```

Figure0−18 Cutting PDO message content

Appendix I Common Formulas

Motor in card walking, suitable for motor + gearbox + wheel

Formula：T*n=μ*m*g*d/2	
Wheel diameter d	m
Reduction speed of gearbox n	1：n
Torque of motor T	Nm, kgm^2/s^2
Load ability of car m	kg
Coefficient of friction μ	No unit
Gravitational acceleration g	m/s^2

Under the pulse mode, relationship between No. of pulse and mechanical displacement

Formula：N*A/B=s*n*r/P	
Gear ratio numerator A	No unit
Gear ratio denominator B	No unit
Screw pitch P	mm
Motor single-turn pulse r	No unit
Gear ratio 1：n	No unit
Mechanical displacement s	mm
Pulse N	No unit

Relationship between rotation speed and line velocity

Formula：n = v÷r÷π	
Rotation speed n	rpm
Line velocity v	mm/s
Radius r	mm

Name	Engineering unit	Internal unit	Conversion
Velocity	rpm	DEC	DEC=[(RPM*512*encoder resolution)/1875]

Acceleration		DEC	DEC=[(RPS/S*65536*encoder resolution)/4000000]
Current	A	DEC	1Arms=（2048/(driver peak current)Ipeak/1.414）dec

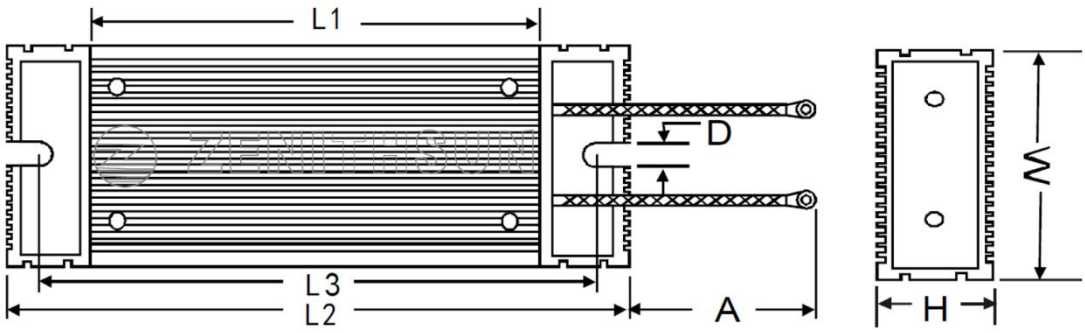
For example, the speed engineering unit is rpm, the internal unit is dec, and the relationship between the two is that 1RPM is approximately equal to 2730dec (encoder resolution 10000)! Assuming that the required speed is 10rpm, the writing speed for communication control is 27300dec, and the hexadecimal number is 6AA4. similarly, when the motor encoder resolution is 65536, the relationship between the two is 1RPM approximately equal to 17896DEC. The current engineering unit is Arms and the internal unit is dec. Assuming that the driver used is MD60 (the peak driver current Ipeak is 50A), then 1Arms is approximately equal to 29dec. If the target current limit needs to be set to 10Arms, the write current needs to be 290dec when using communication control.

Table 1 integrated servo motorpeak current Identification

model	MD60-020	MD60-040	MD80-075
Peak current（Ipeak）	50Ap	50Ap	80Ap

Appendix 2 Use of brake resistor

The energy generated by the servo motor in the braking state will be fed back to the DC bus of the driver. When the voltage value of the DC bus exceeds the protection range, the driver will report the bus voltage is too high and the excess energy needs to be consumed by external braking resistor. The resistance value of the optional braking resistor shall not be lower than the recommended resistance value. Connect the brake resistor through the Rb+and RB- at the power end, and correctly set the brake resistor resistance and brake resistor power.



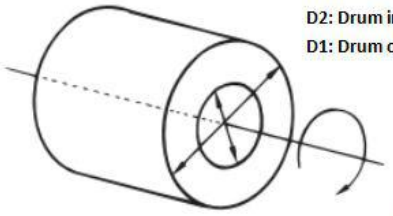
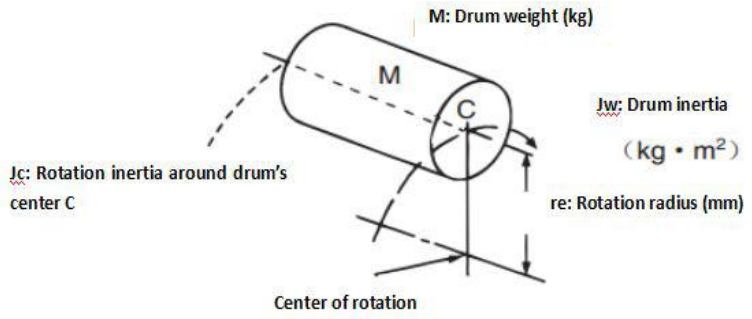
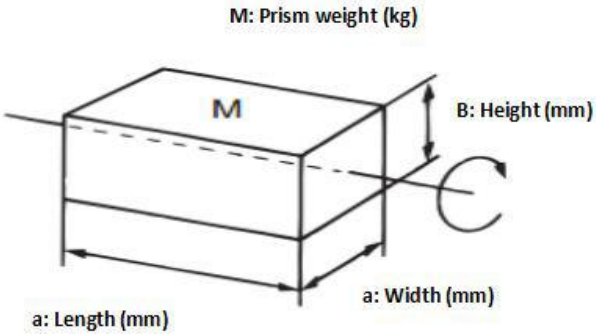
Power	Size						
	W±1	H±1	L1±2	L2±2	L3±2	D±0.5	A±10
100W	40	20	110	140	125	5.2	300

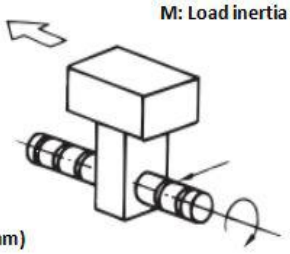
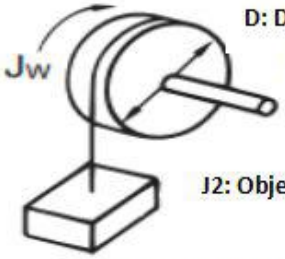
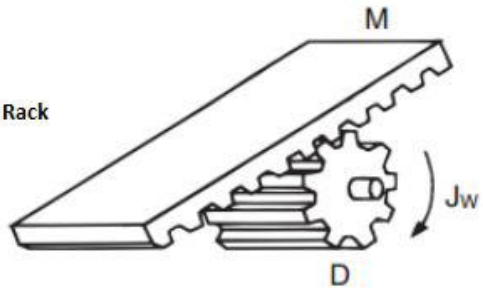
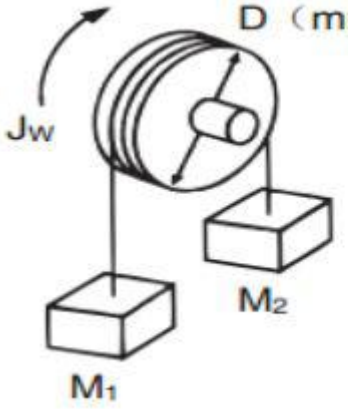
Driver type	Brake resistor type	Brake resistor resistance[Ω]	Brake resistor power[W]	Brake resistor voltage resistant [VDC] （Minimum）
MD60(200W)	T-27R-100	27	100	500
MD60(400W)	T-10R-100	10	100	500
MD80(750W)	T-5R-100	5	100	500

Brake resistance parameter setting

Address	Data type	Modbus address	RWS	Unit	Meaning
65100810	Unsigned16	0x6810	RW	V	Chopping voltage point, default 70V
60F70110	Unsigned16	0x6010	RW	Ω	Brake resistance
60F70210	Unsigned16	0x6020	RW	W	Brake resistance power

Appendix 3 General load inertia calculation

Drum inertia		$J_W = \frac{M(D_1^2+D_2^2)}{8} \times 10^{-6}(kg \cdot m^2)$
Eccentric circular plate inertia /Drum inertia (Rotation center is shifted)		$J_W=J_C+M \cdot re^2 \times 10^{-6}(kg \cdot m^2)$
Rotating prism inertia		$J_W = \frac{M(a^2+b^2)}{12} \times 10^{-6}(kg \cdot m^2)$

Rectilinear motion object inertia	 M: Load inertia (mm) JB: Slip-ball screw inertia (kg・m²) P: Slip-ball screw pitch (mm) Jw: Inertia (kg・m²)	$J_W=M\left(\frac{P}{2\pi}\right)^2\times10^{-6}+J_B(kg\cdot m^2)$
Object inertia when it is lifted by pulley	 D: Diameter (mm) Jw M1: Drum weight (kg) J1: Drum inertia (kg·m²) J2: Object inertia (kg·m²) M2: Object weight (kg) Jw: Inertia (kg·m²)	$J_W=J_1+J_2$ $=\left(\frac{M_1\cdot D^2}{8}+\frac{M_2\cdot D^2}{4}\right)\times10^{-6}(kg\cdot m^2)$
Object inertia when it is transmitted by rack or gear	 M Rack Jw D Jw: Inertia (kg·m²) M: Weight (kg) D: Gear diameter (mm)	$J_W=\frac{M\cdot D^2}{4}\times10^{-6}\ (kg\cdot m^2)$
Inertia with counterweight	 D (mm) Jw M2 M1 Jw: Inertia (kg·m²) M1: Weight (kg) M2: Weight (kg)	$J_W=\frac{D^2(M_1+M_2)}{4}\times10^{-6}\ (kg\cdot m^2)$

Inertia when object is transmitted by conveyor belt	M3: Object weight (kg) M4: Conveyor belt weight (kg) D1: Roller 1 diameter (mm) Jw: Weight (kg·m²) Jw: Inertia (kg·m²) J1: Roller 1 inertia (kg·m²) M1: Roller 1 weight (kg) J2: Inertia produced by roller 2 (kg·m²) J3: Inertia produced by object (kg·m²) J4: Inertia produced by conveyor belt (kg·m²) D2: Roller 2 diameter (mm) M2: Roller 2 weight (kg)	$J_W=J_1+J_2+J_3+J_4$ $=\left(\frac{M_1\cdot D_1^2}{8}+\frac{M_2\cdot D_2^2}{8}\cdot\frac{D_1^2}{D_2^2}+\frac{M_3\cdot D_1^2}{4}+\frac{M_4\cdot D_1^2}{4}\right)\times 10^{-6}$ (kg·m²)
Inertia when workpiece is nipped by roller	Jw: System inertia (kg·m²) J1: Roller 1 inertia (kg·m²) J2: Roller 2 inertia (kg·m²) D1: Roller 1 diameter (mm) D2: Roller 2 diameter (mm) M: Workpiece equivalent weight (kg) J1 Roller 1 D1 Jw M D2 Roller 2 J2	$J_W=J_1+\left(\frac{D_1}{D_2}\right)^2 J_2+\frac{M\cdot D_1^2}{4}\times 10^{-6}$ (kg·m²)
Load inertia when convert to motor shaft	Load Gear Z2: No. of gear near to load J2: Gear inertia near to load (kg·m²) Jw: Load inertia (kg·m²) Motor Z1: No. Of gear near to motor J1: Gear inertia near to load (kg·m²) JL: Load inertia when convert to motor shaft (kg·m²) Variable transmission ratio G=Z1/Z2	$J_L=J_1+G^2(J_2+J_W)(kg\cdot m^2)$

Appendix 4 Control Terminal Wiring Instructions

The MD series distributes the plug terminals and pins of X1 and X2 communication ports and X3 external output ports with the products. wires in the specification range of 30~22AWG shall be used together, and DuPont terminal crimping pliers shall be used to make cables.

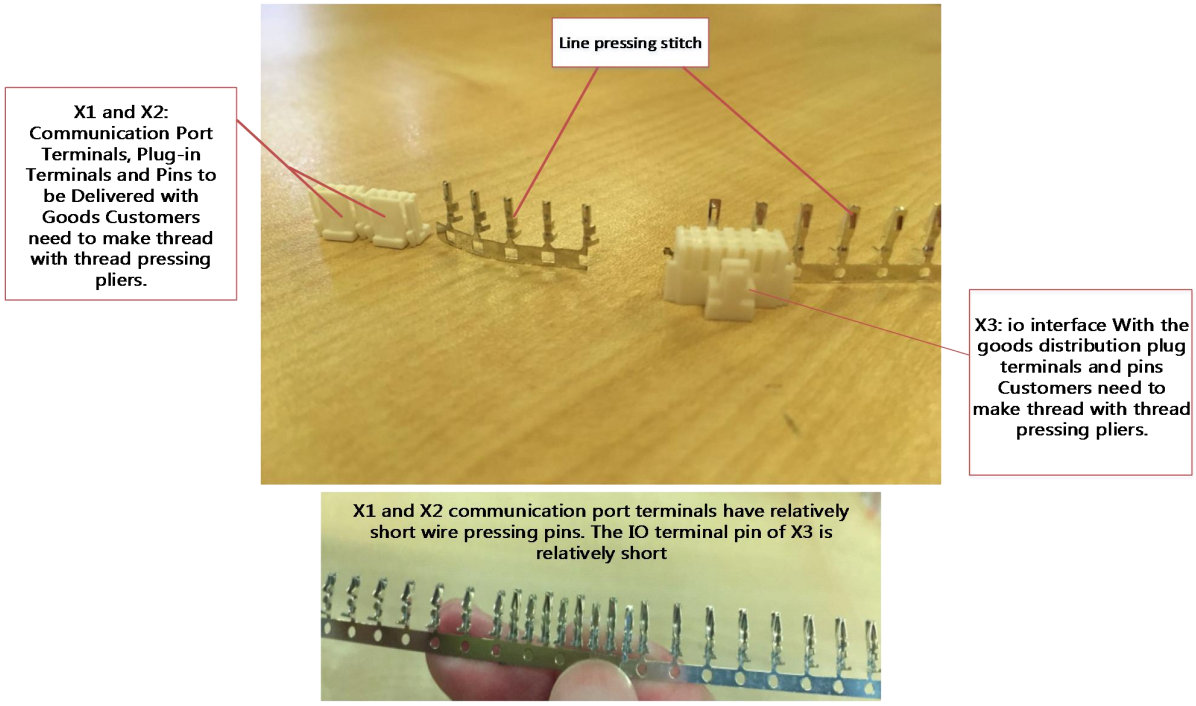
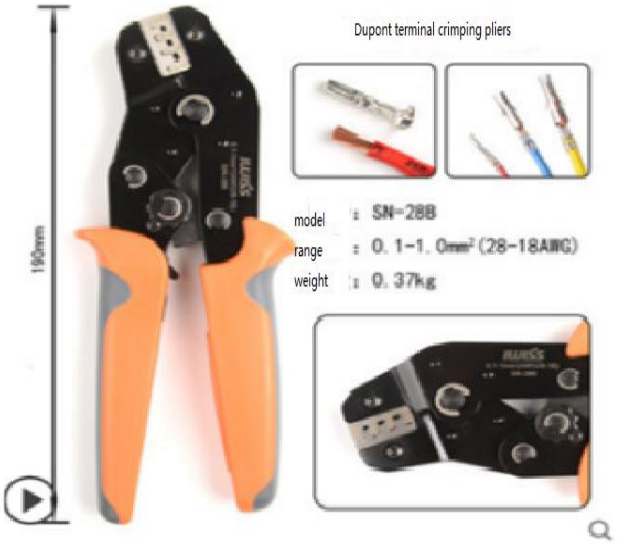


Figure. 1 description of crimping terminals and crimping pins



Instructions for use and purchase link of crimping tool (for reference only) :

https://detail.tmall.com/item.htm?id=43561904131&spm=a1z09.2.0.0.6f0e2e8dckA6sS&_u=d2393kj15fc&skuId=3503574049108

Figure. 2 illustration of wire press

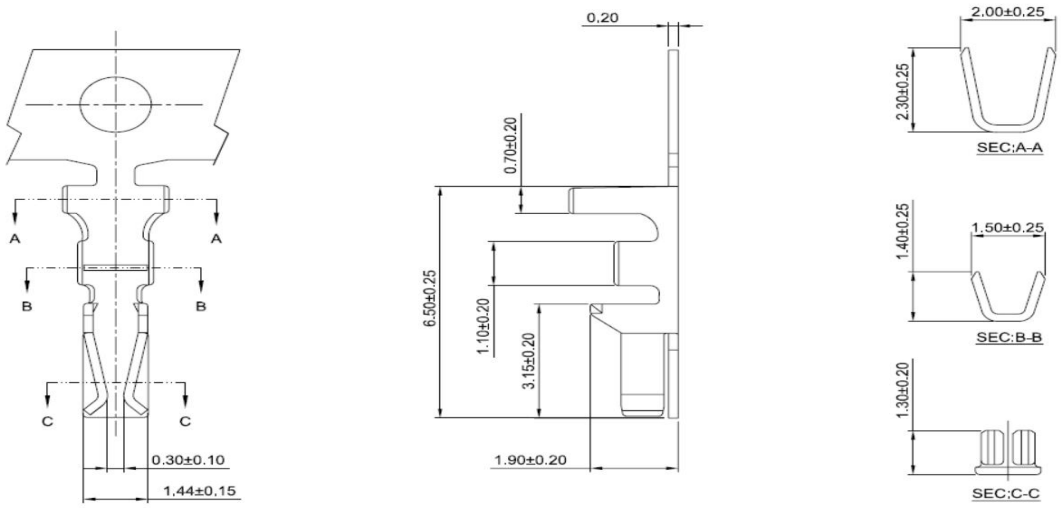


Figure 3 Specifications of Pins for X1 and X2 Communication Ports

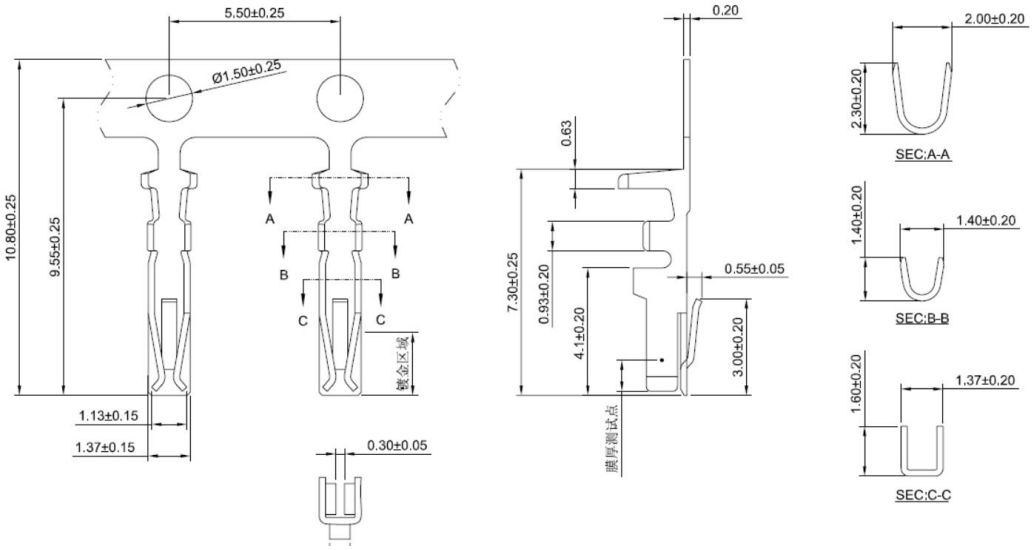


Figure 4 X3 IO Port Press Pin Specification

MD series integrated servo motor Parts list

port		Docking terminal		
		Name	Specification model	Quantity
X1 (CANor RS485)		Plug	ZER-04V-S	2
		Pin	SZE-002T-P0.3	8
X2 RS232		2.0mm 4P Plug	CJT A2008H-04P	1
		Single row of metal pins	CJT A2008-TP	4
X3 IO		Terminal (Head)	CJT A2008H-2x6P	1
		Crimp pin	CJT A2008-TP-A	12
X5	MD60Power input	Terminal (Head)	DINKLE 0226-0704	1
	MD80Power input	Terminal (Head)	DINKLE 0227-0704	1