Content

Chapter 1 EtherCAT protocol

1.1 EtherCAT overview

EtherCAT (Ethernet for Control Automation Technology) is a real-time industrial Ethernet technology developed by BECKHOFF and offers the following advantages:

(1) High speed: Transmission rates up to 2 x 100 Mbit/s (full duplex mode) using Fast Ethernet technology.

(2) Flexible: Supports various topologies such as linear and ring; inter-station distance can reach 100m, and the number of nodes can theoretically reach 65536.

(3) Compatible: Conforms to Ethernet specifications, allowing the use of standard Ethernet devices such as switches.

(4) Synchronization: Uses ASIC to achieve hard real-time and achieve clock synchronization accuracy of less than 1μs.

(5) High efficiency: Data refresh cycle of less than 100μs can be achieved, which is suitable for closed-loop control of the servo.

EtherCAT has a master-slave communication architecture, where the master can use a general-purpose Ethernet controller and the slaves require a special communication chip called EtherCAT Slave Controller (ESC). The PLC or controller is the master and the servo drives are the slaves.

EtherCAT uses the physical layer of standard Ethernet, modifies the data link layer to transfer data using a specialized protocol, and defines the application layer. Figure 1-1 shows a model of a Kinco Servo EtherCAT slave supporting clock synchronization and CANopen protocol, as detailed in the subsequent chapters.

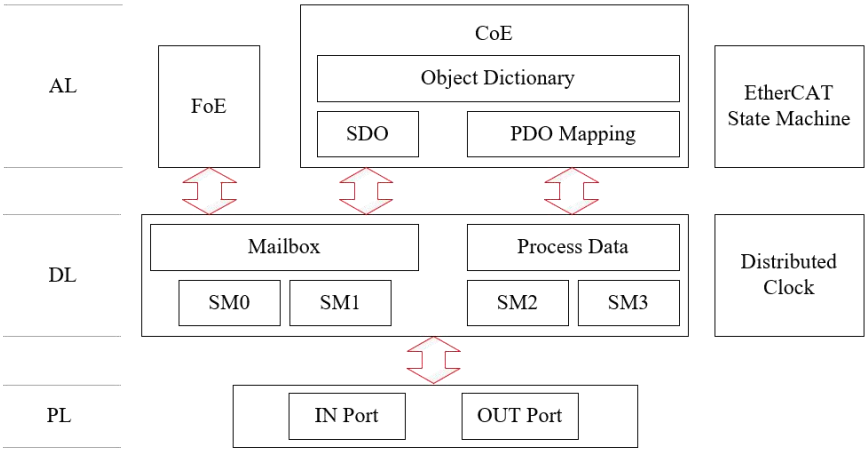


Figure 1-1 Kinco servo EtherCAT slave model

Note

- EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

1.2 EtherCAT specifications

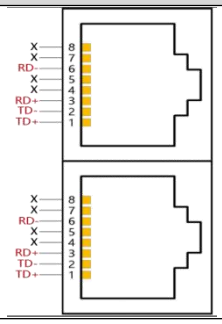
Table 1-1 describes the specification of Kinco servo EtherCAT communication

Table 1-1 EtherCAT communication specification

Item		Specification
Physical Layer	Transmission standard	IEEE 802.3
	Transmission interface	2 x RJ45 (IN、OUT)
	Transmission medium	100 Base-TX standard Ethernet cable
	Transmission distance	100 m
	Transmission rate	2 x 100 Mbit/s (full duplex)
	Topological structure	Linear, circular
	Number of nodes	No more than 100
	EtherCAT frame length	44 bytes ~ 1498 bytes
Data Link Layer	Communication standard	IEC 61158 type 12, IEC 61784 type 12
	DC (Distributed Clock)	64 bit
	Synchronous jitter	Less than 1μs
	Refresh time	100 axes about 100 μs
	Number of FMMU	8
	Number of Sync Manager	8
	Size of DPRAM	8KB
	Communication mode	Free Run
		DC mode, synchronized to SYNC0
Application Layer	Application standard	IEC 61800-7-201 (DS 402), IEC 61800-7-301
	Application Layer function	CoE (CANopen over EtherCAT)
		FoE (File Access over EtherCAT)
	CoE communication	Variable PDO mapping
		SDO request, SDO response
	CoE extended operating mode	Cyclic synchronous position mode (CSP)
		Cyclic synchronous velocity mode (CSV)
		Cyclic synchronous torque mode (CST)

See Table 1-2 for Kinco servo EtherCAT interface.

Table 1-2 EtherCAT interface

RJ45 pin definition	Pin number	Name	Description
	1	TD+	Data transmitting +
	2	TD-	Data transmitting -
	3	RD+	Data receiving +
	6	RD-	Data receiving -

1.3 EtherCAT system

The EtherCAT system consists of a master and a number of slaves. The master initiates the communication, the slave processes the received message and extracts or inserts relevant data into the message, and then transmits the message to the next slave. The last slave returns the fully processed message and passes it in reverse order to the first slave and finally to the master, as shown in Figure 1-2.

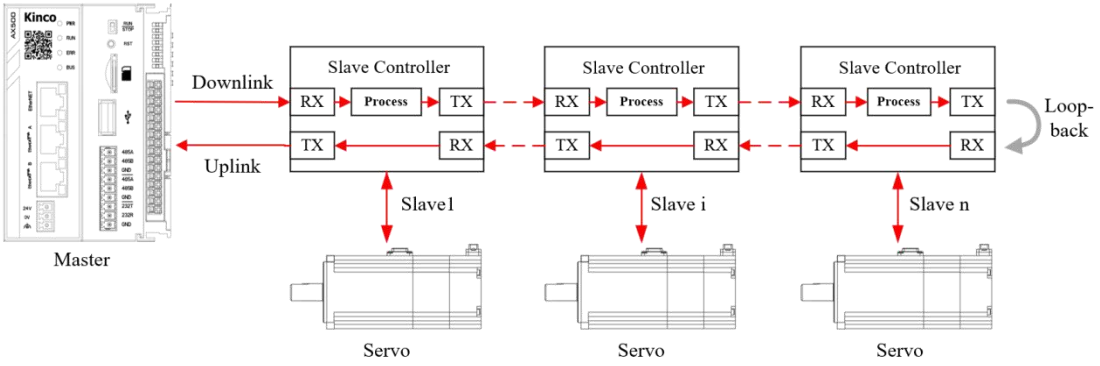


Figure 1-2 EtherCAT system operation

To make a network connection, connect the EtherCAT port of master to the IN port of slave, and the OUT port of slave to the IN port of the next slave, as shown in Figure 1-3.

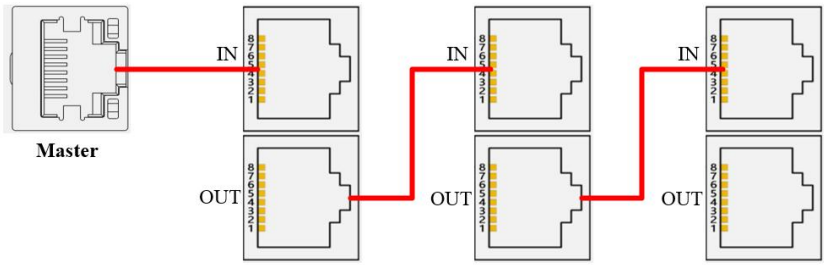


Figure 1-3 EtherCAT system topology

1.4 Distributed Clock

Distributed Clock enables all EtherCAT devices to use the same system time, and the servo slaves generate synchronized interrupt signals based on the synchronized system time to execute tasks. This is key to achieving strictly synchronized and coordinated motion of multiple servos for precise contour trajectory control.

Each DC slave has a local clock. EtherCAT specifies that the clock of the first DC slave connected to the master is the reference clock, and the clocks of the other slaves and the master are slave clocks, which need to be synchronized to the reference clock, as shown in Figure 1-4. The distributed clock automatically calculates the transmission delay, initial offset from the reference clock to all slave clocks and dynamic clock drift and compensates them to ensure that the local system time of all slaves remains consistent.

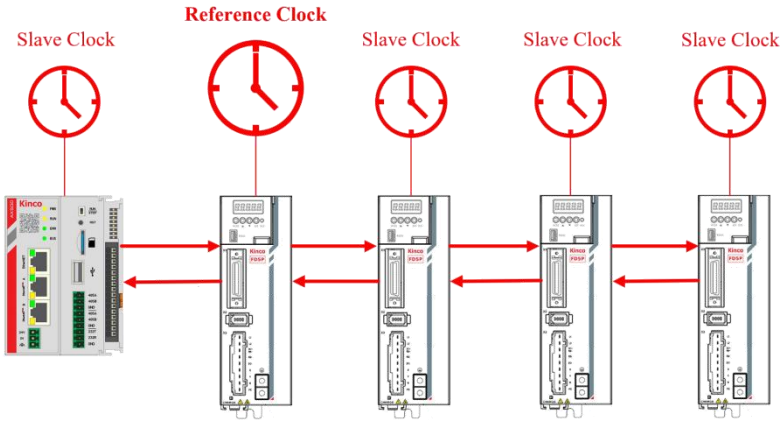
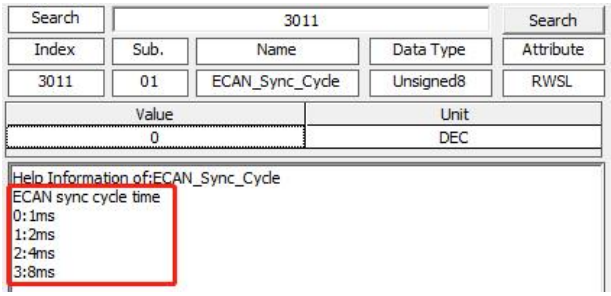
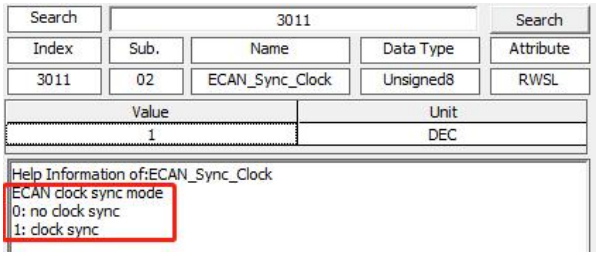


Figure 1-4 Distributed Clock

Kinco servo can select DC synchronization cycle of 1ms, 2ms, 4ms and 8ms, which is set by ECAN_Sync_Cycle [301101]:



When using DC mode, the servo needs to turn on synchronization mode, which is set via the ECAN_Sync_Clock [301102]:



In DC mode, the servo local cycle is triggered by the synchronization signal SYNC0. The distributed clock and synchronization signal configuration related registers are shown in Table 1-3. The distributed clock is initialized, configured, started running and maintained by the master.

Table 1-3 ESC Distributed Clock related registers

Address	bit	Description
0x0910:0x0917	0~63	Local system time
0x0920:0x0927	0~63	Offset of the local clock and reference clock
0x0928:0x092B	0~31	Transmission delay between reference and slave clocks
0x0981	0	0: Invalid; 1: Activate cycle run
	1	0: Invalid; 1: Enable SYNC0 synchronization signal
0x0990:0x0997	0~63	Cycle run start time
0x09A0:0x09A3	0~31	SYNC0 cycle time

1.5 EtherCATState Machine

EtherCAT State Machine (ESM) is responsible for coordinating the communication between master and slaves.

The Kinco servo supports all 5 ESM states:

(1) Init (I)
(2) Pre-Operational (P)
(3) Safe-Operational (S)
(4) Operational (O)
(5) Boot-Strap (B)

Table 1-4 describes the meanings of ESM status

Table 1-4 ESM state

State	Meaning
Init	• No communication between master and slave at the application layer • Master initializes ESC configuration related registers • Master configures mailbox channel parameters
Pre-Operational	• Mailbox communication is possible • No process data communication
Safe-Operational	• Mailbox communication is possible • Slave transmits input process data (TxPDO) • The output process data (RxPDO) sent by master is invalid
Operational	• Mailbox communication is possible • Input process data and output process data are valid
Boot-Strap	• Mailbox communication available, but only for downloading firmware via FoE protocols

The ESM is shown in Figure 1-5, and the slave transitions states according to the following rules:

(1) When changing from Init state to Op state, the sequence "Init → Pre-Op → Safe-Op → Op" must be followed.
(2) Cross-over transitions are allowed when returning from Op state.
(3) Bootstrap state only allows interchanges with Init state.

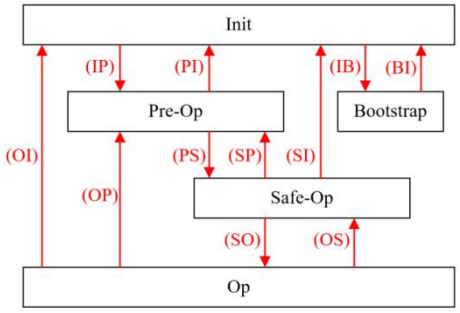


Figure 1-5 EtherCAT State Machine

The meaning of each transition in Figure 1-5 is shown in Table 1-5.

Table 1-5 ESM state transitions

Transition	Meaning
IP	• Master configures mailbox channel parameters and starts mailbox communication • Master configures Distributed Clock related registers • Master configures slave address registers
PI	• Stop mailbox communication
PS	• Master initializes process data mapping using mailbox • Master configures SM channel used for process data communication • Master configures FMMU • Start updating input process data (TxPDO)
SP	• Stop updating input process data (TxPDO)
SO	• Start updating output process data (RxPDO)
OS	• Stop updating output process data (RxPDO)
OP	• Stop process data communication
SI	• Stop process data communication • Stop mailbox communication
OI	• Stop process data communication • Stop mailbox communication
IB	• Start boot mode
BI	• Restart the slave.

Master writes the target state to the slave AL (Application Layer) control register to change the slave state; reads the slave AL status register to get the actual state of slave, the related ESC registers are shown in Table 1-6.

Table 1-6 ESC application layer related registers

Address	Length (B)	Description
0x0120:0x0121	2	Application layer control
0x0130:0x0131	2	Application layer status
0x0134:0x0135	2	Application layer status code

Chapter 2 CoE Protocol

Kinco's EtherCAT servo supports the DS 402 guild regulation(CoE) in the CANopen protocol, as shown in Figure 2-1. The CoE uses email to access the object dictionary for device configuration and aperiodic parameter reading and writing ; Use process data communication to periodically transmit instruction data and status data. The relevant terms are explained as follows——

(1) Object dictionary: Use object to describe the full functionality of the CANopen device , it is a list of all parameters of the device.

(2) SDO: Service Data Object is used for aperiodic mailbox communication.

(3) PDO: Process Data Objects is used for periodic process data communication.

(4) SM: Sync Manager is used to ensure the consistency and security of data exchange between master and slave.

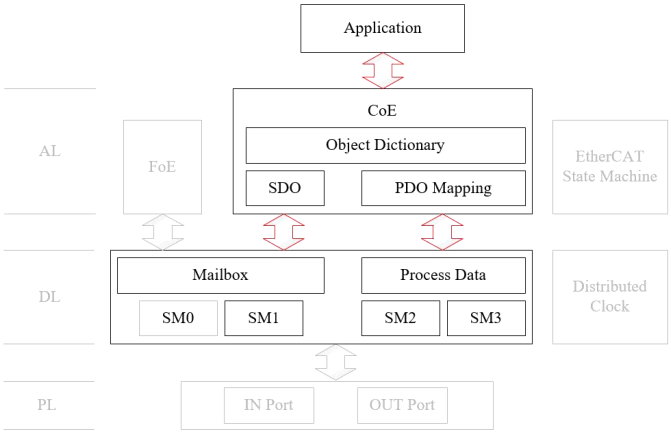


Figure 2-1 Coe based Kinco servo slave model

2.1 CoE object dictionary

CoE complies with the CANopen protocol and has the same definition in its object dictionary. Related communication objects 0x1C00 to 0x1C4F for EtherCAT communication are extended for setting SM channels and PDO allocation. See Table 2-1.

Table 2-1 CoE object dictionary definition

Index	Definition
0x0000:0x0FFF	Data type description
0x1000:0x1018	Device type and identifier
0x1600:0x17FF	RxPDO mapping Subindex 0: Number of object Subindex 1: The first output data object of the mapping ⋮ Subindex n: The last output data object of the mapping

0x1A00:0x1BFF	TxPDO mapping Subindex 0: Number of object Subindex 1: The first input data object of the mapping ⋮ Subindex n: The last input data object of the mapping
0x1C00	SM Channel type 0：Mailbox output, aperiodic data communication 1：Mailbox input, aperiodic data communication 2：Process data output, periodic data communication 3：Process data input, periodic data communication
0x1C10:0x1C2F	PDO assignment of SM channel during process data communication Subindex 0: Numbe of assigned PDO Subindex 1~n: PDO mapping object index
0x1C30:0x1C4F	SM channel parameter
0x2000:0x5FFF	Data object are defined by kinco
0x6000:0x9FFF	Data object are defined by DS 402 guild regulation

2.2 Process data communication

Real-time is the ability to complete a task in a certain amount of time. As a real-time industrial Ethernet, EtherCAT periodically transmits real-time data, known as process data communication. Process data is divided into output process data (RxPDO) and input process data (TxPDO). RxPDO contains the command data updated by the master station, and TxPDO contains the status data updated by the slave station, as shown in Figure 2-2.

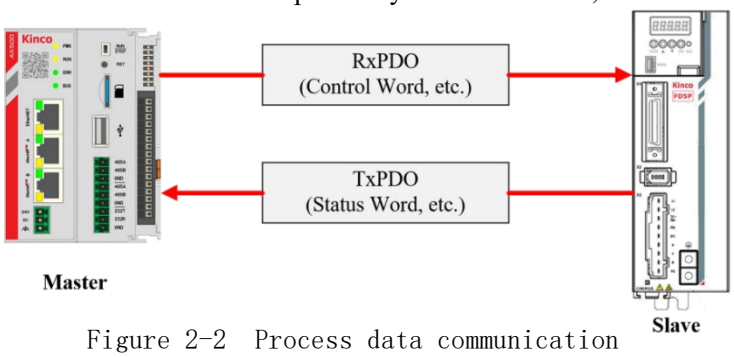


Figure 2-2 Process data communication

For the user, the actual data objects that need to be used are PDO entries, such as control word, target location, status word, actual location, etc. As shown in Figure 2-3, the data communication process is as follows:

(1) Select the desired PDO entry.

(2) Perform PDO mapping

(3) Perform PDO allocation, that is, configure the synchronization manager.

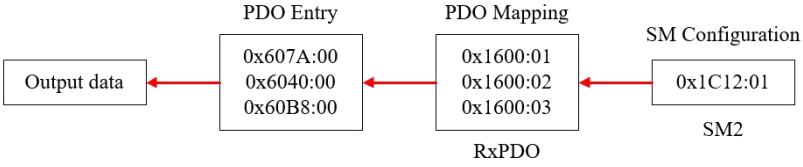


Figure 2-3 Configuring process data communication (using output data as an example)

For example, if you compare PDO entries to desired goods, then the data objects 0x1600 and 0x1A00 are trucks. Mapping PDO entries to 0x1600 and 0x1A00 is the equivalent of loading goods onto a truck. After the goods are loaded, roads are needed. The CoE builds one one-way street (SM2 channel and SM3 channel) from the primary station to the secondary station and from the secondary station to the primary station. Using data objects 0x1C12 and 0x1C13, PDO allocation is equivalent to putting a truck on the corresponding road.

Kinco servo supports variable PDO mapping, as shown in Table 2-2, including 3 sets of RxPDO and 2 sets of TxPDO, which can be configured through CoE communication and the PDO content can be changed.

Table 2-2 PDO mapping

Variable PDO mapping	Index	Default PDO entry
RxPDO	0x1600	Target_Position [607A00] Controlword [604000] Touch_Probe_Function [60B800]
	0x1601	Target_Position [607A00] Controlword [604000] Operation_Mode [606000] Profile_Acc [608300] Profile_Dec [608400] Touch_Probe_Function [60B800]
	0x1602	Profile_Speed [608100] Target_Speed [60FF00]
TxPDO	0x1A00	Pos_Actual [606400] Status word [604100] Error_Code [603F00] Operation_Mode_Buff [606100] Touch_Probe_Status [60B900] Touch_Probe_Rising1 [60BA00] Digital_Input [60FD00]
	0x1A01	Pos_Actual [606400] Status word [604100] Operation_Mode_Buff [606100] Speed_Real [606C00] Dout2_Function [201010] Error_State [260100] Error_State2 [260200]

CoE uses data object 0x1C10:0x1C2F to define the PDO mapping object list of the SM channel. The output process data uses the SM2 channel and is allocated by PDO defined by data object 0x1C12. The input process data uses the SM3 channel and is assigned by the PDO defined by the data object 0x1C13, as shown in Table 2-3.

Table 2-3 PDO assignment

PDO allocation (Sync Manager configuration)	Index	Subindex	PDO mapping index
SM2	0x1C12	0x01	0x1600 or 0x1601 or 0x1602
SM3	0x1C13	0x01	0x1A00 or 0x1A01

After configuring process data communication, in KincoServo PC software, click Driver → ECAN Configuration→RPDO/TPDO, and you can see the configured PDO:

RPDOSet

RPDO1 | RPDO2 | RPDO3 | RPDO4 | RPDO5 | RPDO6 | RPDO7 | RPDO8

N	Index	Type	Name	Value	Unit
0	160000	uint8	Group_RX1_PDO	3	DEC
1	160001	uint32	RX1_PDO1	607A0020	HEX
2	160002	uint32	RX1_PDO2	60400010	HEX
3	160003	uint32	RX1_PDO3	60B80010	HEX
4	160004	uint32	RX1_PDO4	00000000	HEX
5	160005	uint32	RX1_PDO5	00000000	HEX
6	160006	uint32	RX1_PDO6	00000000	HEX
7	160007	uint32	RX1_PDO7	00000000	HEX
8	160008	uint32	RX1_PDO8	00000000	HEX
9	140001	uint32	RX1_ID	?????	HEX
10	140002	uint8	RX1_Transmission	?????	DEC
11	140003	uint16	RX1_Inhibit_Time	?????	DEC

TPDOSet

TPDO1 | TPDO2 | TPDO3 | TPDO4 | TPDO5 | TPDO6 | TPDO7 | TPDO8

N	Index	Type	Name	Value	Unit
0	1A0000	uint8	Group_TX1_PDO	7	DEC
1	1A0001	uint32	TX1_PDO1	60640020	HEX
2	1A0002	uint32	TX1_PDO2	60410010	HEX
3	1A0003	uint32	TX1_PDO3	603F0010	HEX
4	1A0004	uint32	TX1_PDO4	60610008	HEX
5	1A0005	uint32	TX1_PDO5	60B90010	HEX
6	1A0006	uint32	TX1_PDO6	60BA0020	HEX
7	1A0007	uint32	TX1_PDO7	60FD0020	HEX
8	1A0008	uint32	TX1_PDO8	00000000	HEX
9	180001	uint32	TX1_ID	?????	HEX
10	180002	uint8	TX1_Transmission	?????	DEC
11	180003	uint16	TX1_Inhibit_Time	?????	DEC
12	180005	uint16	TX1_Event timer	?????	DEC

2.3 Mailbox data communication

Mailbox communication uses SM0 and SM1 channels to access entries in the object dictionary through SDO for communication configuration, device configuration, and aperiodic parameter reading and writing.

SDO transfers are divided into download and upload. Download transfers are often used by the master to set the parameters of the slave , and upload transfers are often used by the master to read the performance parameters of the slave . Take SDO download as an example, the master sends the SDO download request to the slave SM0 channel, the slave reads the mailbox data, executes the corresponding processing, and writes the response data to the SM1 channel; The master then reads the SM1 channel to obtain the data and judge the execution result of the slave.

Kinco servo supports SDO request service and SDO response service.

2.4 DS402 guild regulation

Figure 2-4 shows the CoE servo driver structure based on DS402 guild regulation. The following briefly describes the device control state machine and periodic synchronization working mode of DS402 guild regulation.

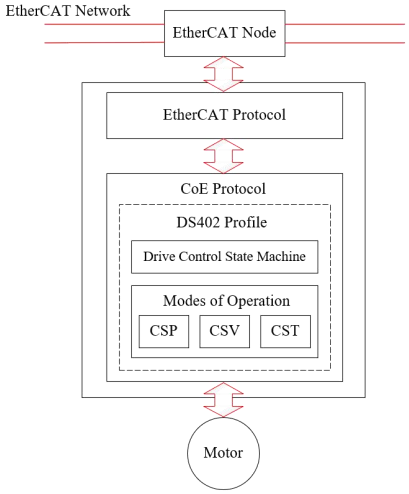


Figure 2-4 CoE servo drive structure

2.4.1 Device control state machine

The DS 402 guild regulation defines the device control state machine of the servo drive, as shown in Figure 2-5. The state of the servo must be changed according to certain rules. The master controls the operating state of the servo by modifying the control word [604000], and obtains the current state of the servo by reading the status word [604100]. The specific meanings of the control word and the status word are described in the Object parameter List section of the Servo user manual.

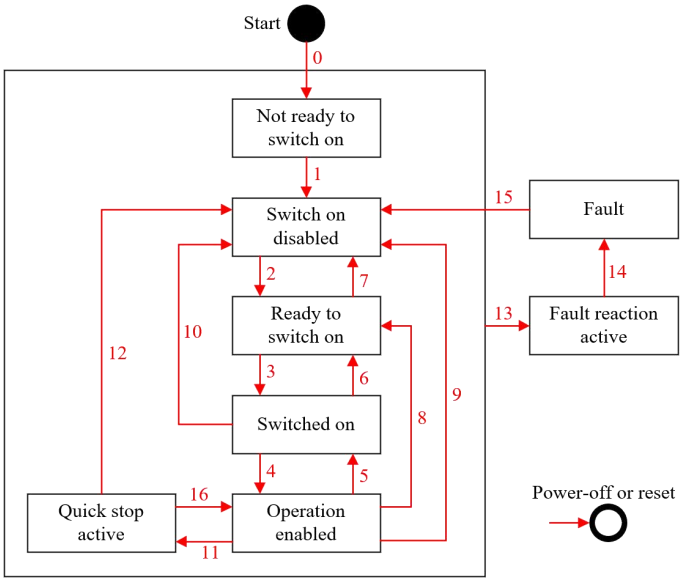


Figure 2-5 Drive control state machine

Table 2-4 and Table 2-5 describe the meaning of each state and transition.

Table 2-4 Device control state machine status description

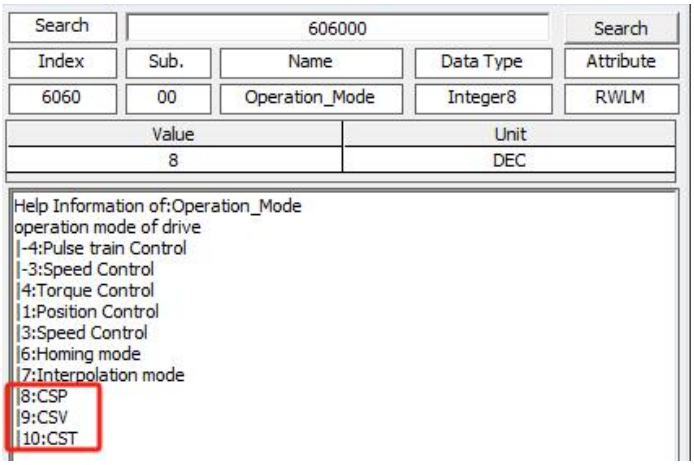
Status	Description	The bit of the status word [604100]					
		Bit6	Bit5	Bit3	Bit2	Bit1	Bit0
Not ready to switch on	The controller is powered on, but initialization is not complete	0	×	0	0	0	0
Switch on disabled	Server initialization is complete	1	×	0	0	0	0
Not ready to switch on	The control power is on, the power switch is off, and the motor has no torque	0	1	0	0	0	1
Switched on	The power is on, the motor has no torque	0	1	0	0	1	1
Operation enabled	The servo driver controls the motor according to the configuration, and the motor has torque	0	1	0	1	1	1
Quick stop	Servo stops as set	0	0	0	1	1	1
Fault reaction	Servo error occurred, stop the machine according to the error setting, the motor has torque	0	×	1	1	1	1
Fault	Error status, the motor has no torque	0	×	1	0	0	0

Table 2-5 Device control state machine transition description

Transition	Description	The bit of the control word [604000]				
		Bit7	Bit3	Bit2	Bit1	Bit0
0	Automatic transition after power-on or reset					
1	Automatic transfer					
2	Get the power cut command from the master	0	×	1	1	0
3	Get the power on command from the master	0	×	1	1	1
4	Get the enable servo run command from the master	0	1	1	1	1
5	Get a stop servo run command from the master	0	0	1	1	1
6	Get the power cut command from the master	0	×	1	1	0
7	Get an quickly stop or power cut command from the master	0	×	×	0	×
		0	×	0	1	×
8	Get the power cut command from the master	0	×	1	1	0
9		0	×	×	0	×
10	Get an quickly stop or power cut command from the master	0	×	×	0	×
		0	×	0	1	×
11	Get the quickly stop command from the master	0	×	0	1	×
12	After the quickly stop function is executed, the power off command is obtained from the master	0	×	×	0	×
13	Fault					
14	Automatic transfer					
15	Get the error reset command from the master	rising edge	×	×	×	×
16	Not recommended	0	1	1	1	1

2.4.2 Operation mode

The periodic synchronous operation mode is an extension of the CoE to the DS 402, including CSP (8 mode), CSV (9 mode), and CST (10 mode). The master sets the servo operating mode by writing the operating mode [606000], and the servo uses the effective operating mode [606100] to represent the actual operating mode.



The periodic synchronous operation mode uses Bit12 of the status word [604100], as shown in Table 2-6.

Table 2-6 Definition of status word bit 12 in periodically synchronized running mode

Bit	Value	Description
12	0	• CSP：Ignore target position • CSV：Ignore target speed • CST：Ignore target torque
	1	• CSP：Target position valid • CSV：Target speed valid • CST：Target torque valid

Figure 2-6 shows the cyclic synchronous position mode (CSP) structure. The master sends periodically synchronized position command values to the servo, and the servo performs position control, speed control, and torque control. The servo can provide the actual position value, the actual speed value and the actual torque value to the master.

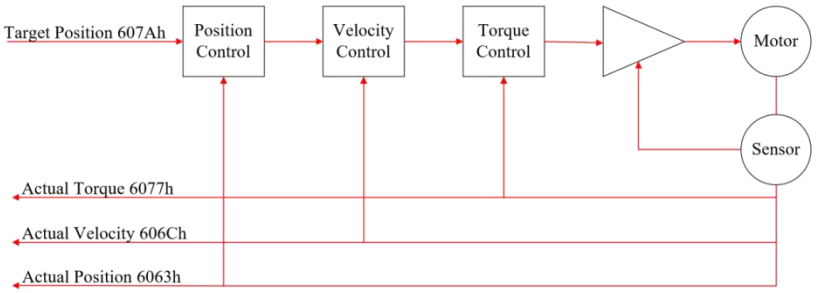


Figure 2-6 Cyclic synchronous position mode

Figure 2-7 shows the cyclic synchronous velocity mode(CSV). The master periodically sends target speed instructions to the servo, the servo performs speed control and torque control, and the position loop can be realized by the master . The servo can provide the actual position value, the actual speed value and the actual torque value to the master .

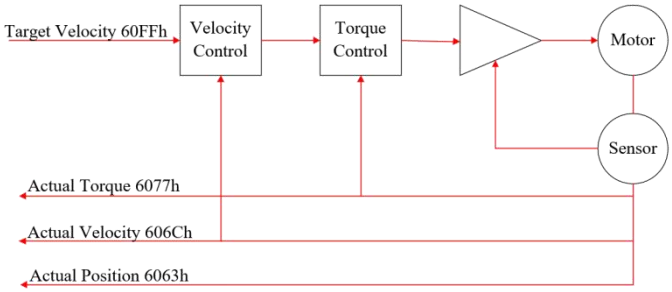


Figure 2-7 Cyclic synchronous velocity mode

Figure 2-8 shows the structure of the Cyclic synchronous torque mode (CST). The main periodically sends target torque instructions to the servo, which performs torque control. The servo can provide the actual position value, the actual speed value and the actual torque value to the master . Cyclic synchronous torque mode can also be extended for torque limits and speed limits to limit dynamic values.

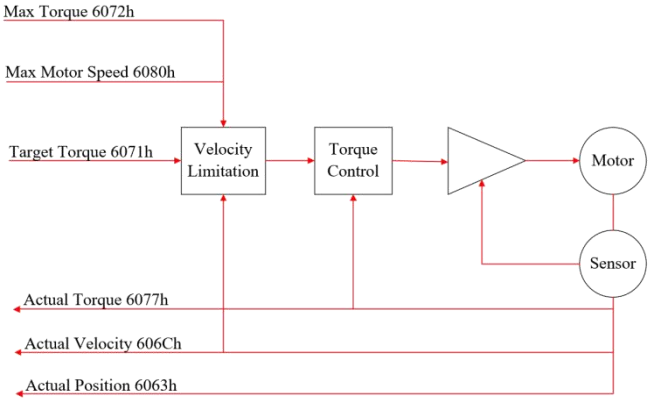


Figure 2-8 Cyclic synchronous torque mode

Chapter 3 Application cases

3.1 Kinco AX500 controller application

When AX500 controller communicates with a single drive, directly use the network cable to connect the EtherCAT port of the controller and the IN port of the drive; when the controller connects with multiple drives, connect the OUT port of the previous drive to the IN port of the next drive, as shown in Figure 3-1.

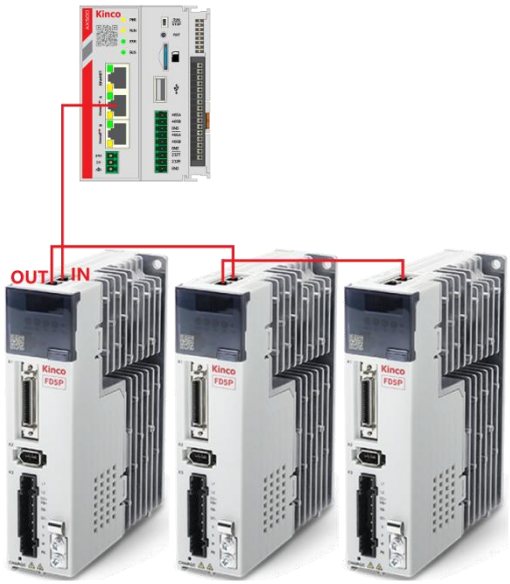
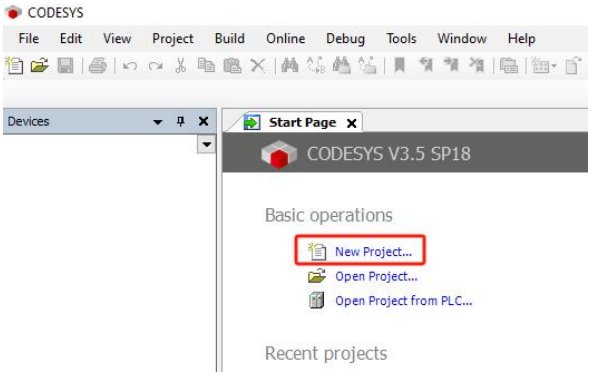


Figure 3-1 Kinco AX500 controller connecting drives

1. Create a new CODESYS project

(1) Start CODESYS → "New Project".

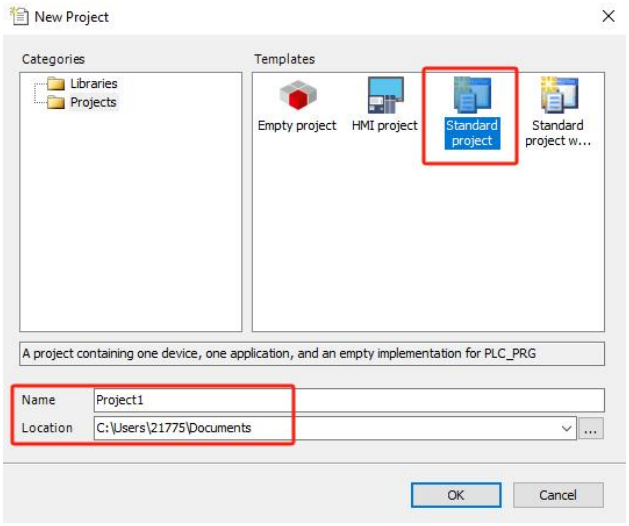


(2) The following options are available in "New Project":

① Select "Standard project".

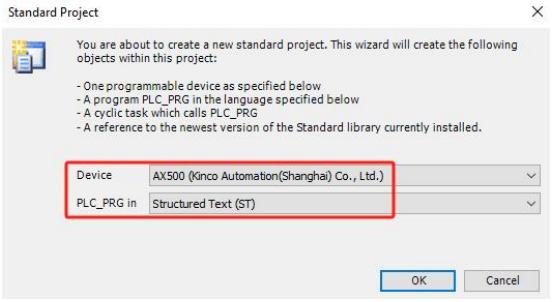
② Fill in the project name, such as "Project1".

③ Select the location to save the project.



④ Select the controller - AX500.

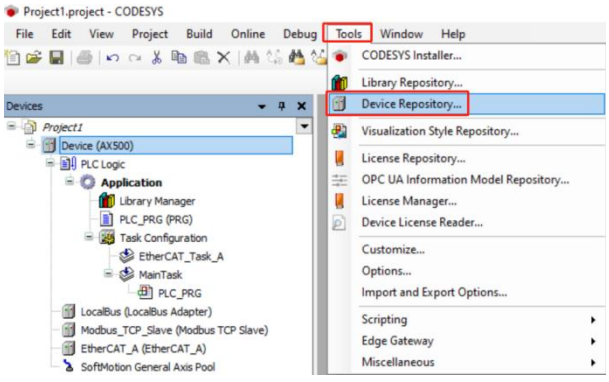
⑤ Select a programming language, such as Structured Text (ST).



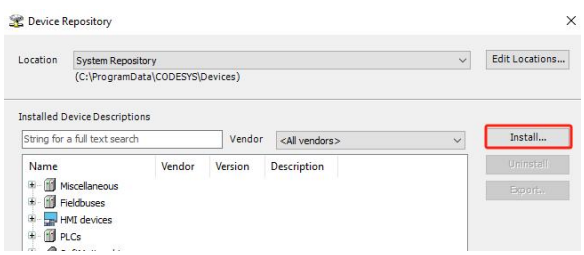
⑥ After adding the controller, the master "EtherCAT_A" of type EtherCAT Master SoftMotion is automatically added.

2. Install the servo XML file

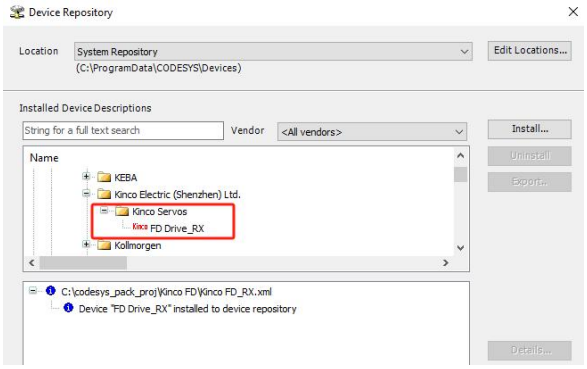
(1) Click "Tools" → "Device Repository... " in the menu bar.



(2) Click "Install... ".

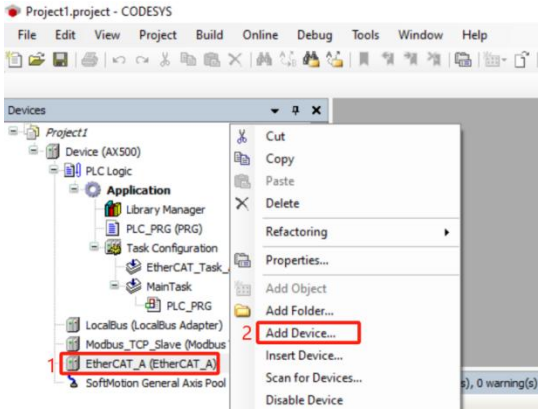


(3) Find the XML file for the drive and install it. The installation is completed as shown below.

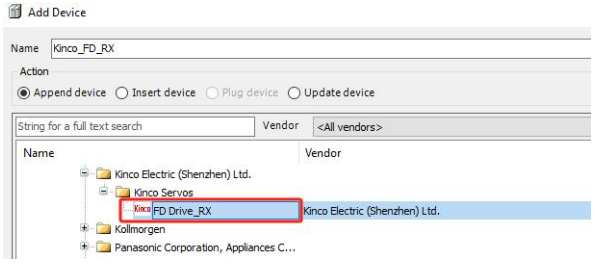


3. Add a servo slave

(1) The EtherCAT master has been added automatically, right click on "EtherCAT_A (EtherCAT_A) " → "Add Device... ".

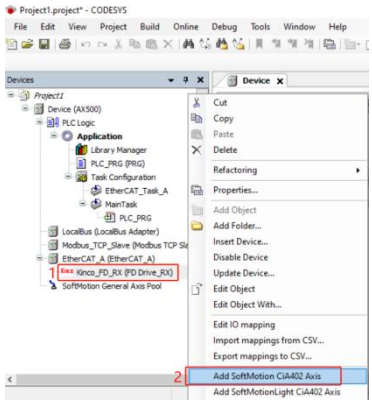


(2) To add a slave, find the servo drive and double-click to add it.



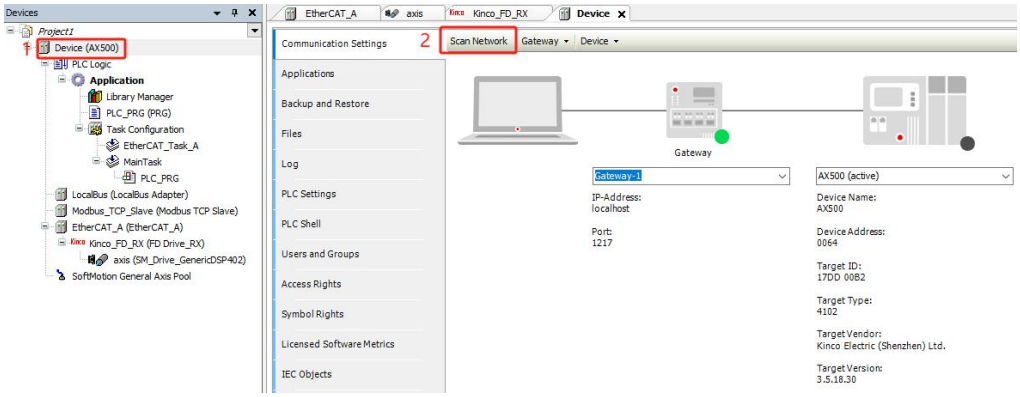
Note: Multiple slaves can be added here, the number added depends on the actual number of drives in the network.

(3) To add a CiA402 axis, right-click on the slave → "Add SoftMotion CiA402 Axis".



4. Connect the controller

(1) Double-click "Device (AX500) " → "Communication Settings" → "Scan Network".

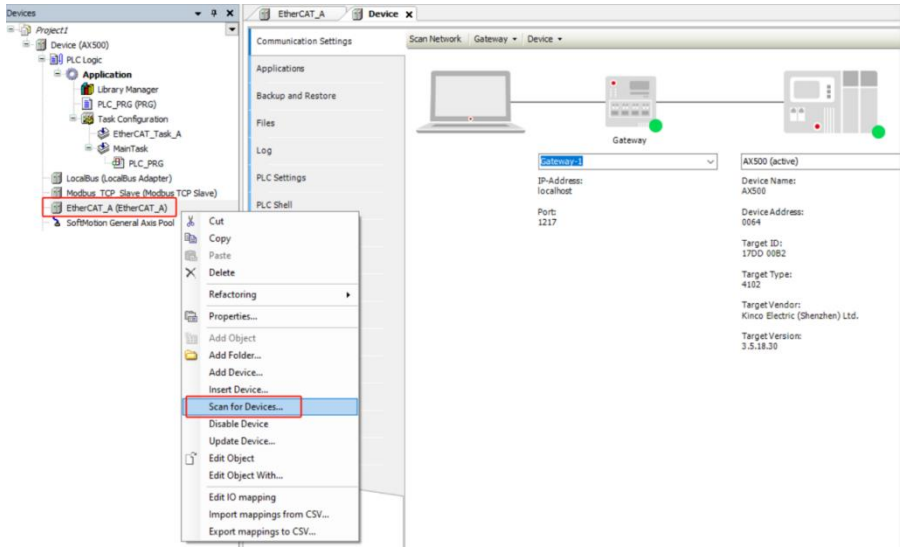


(2) Click on "Scan Network", select "AX500", and then OK.

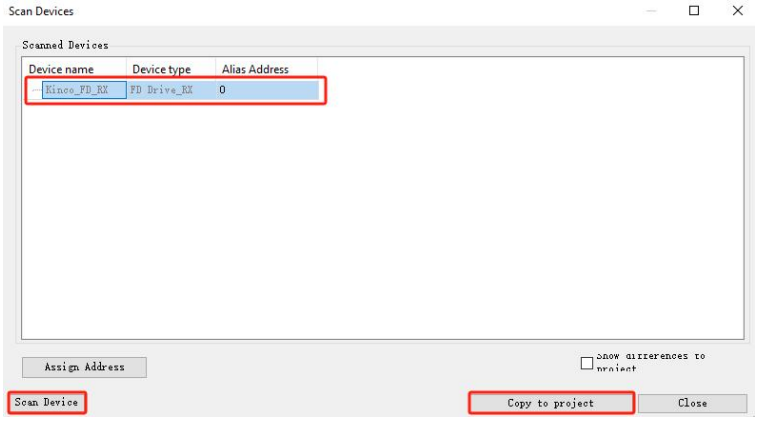
Note: Make sure that the IP of the controller is in the same network segment as the IP of the computer's network card.

(3) Slaves can also be added by online scanning once the controller is connected:

① Right-click "EtherCAT_A (EtherCAT_A) " → "Scan for Devices... ".



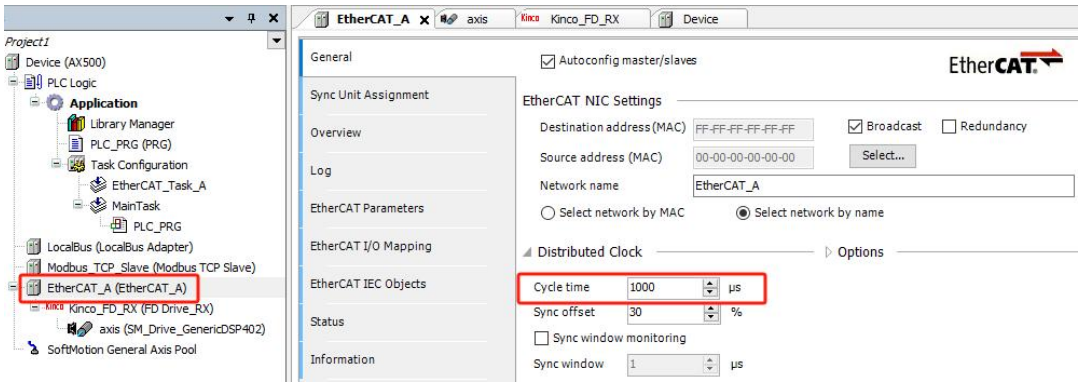
② Click "Scan Device", select the scanned device, click "Copy to project", the slave will be added to the device tree on the left.



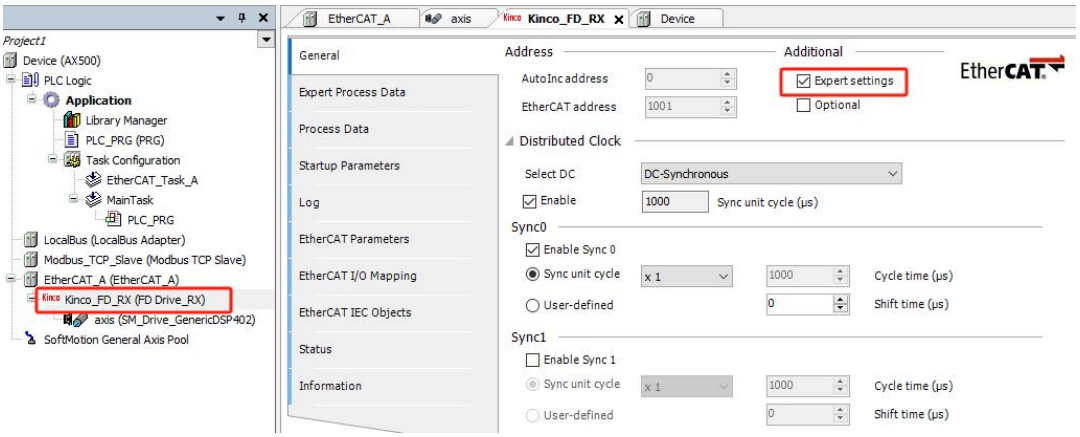
Note: This method of adding a slave is an alternative to the method in step 3.

5. Set the EtherCAT communication parameters

(1) Double click "EtherCAT_A(EtherCAT_A) " → "General" → "Distributed Clock", you can set the synchronization cycle, the following figure is 1ms.

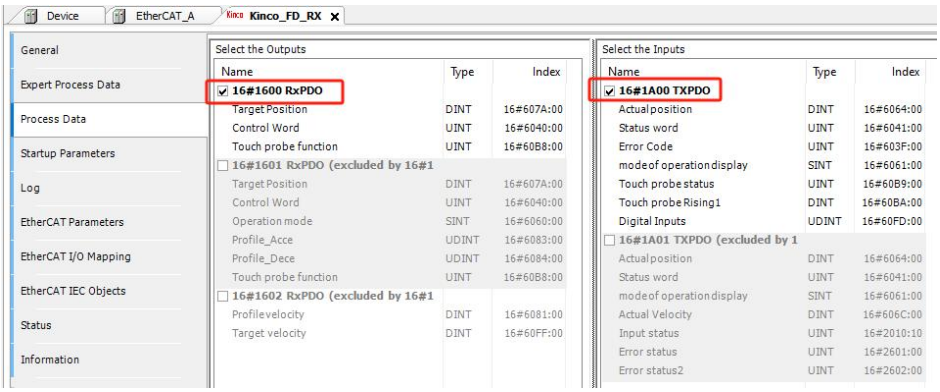


(2) Double-click "Kinco_FD_RX (FD Drive_RX) " → "General" to enable "Expert settings".

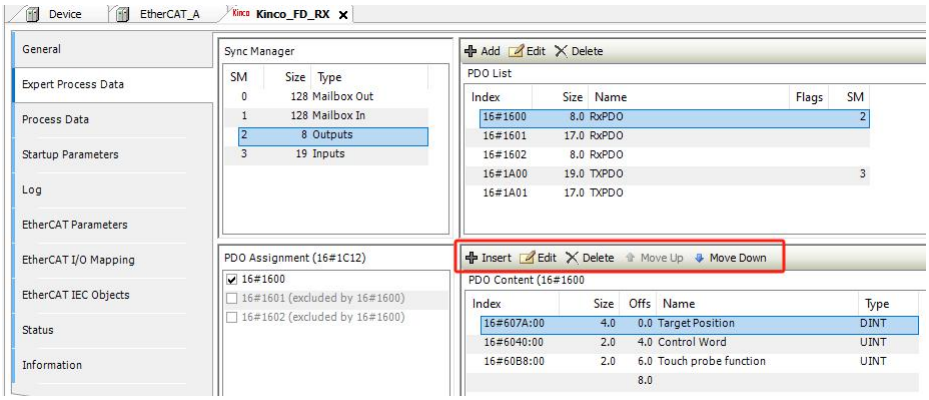


(3) Configure PDO

① To add RxPDO and TxPDO, click "Process Data" to select RxPDO and TxPDO.

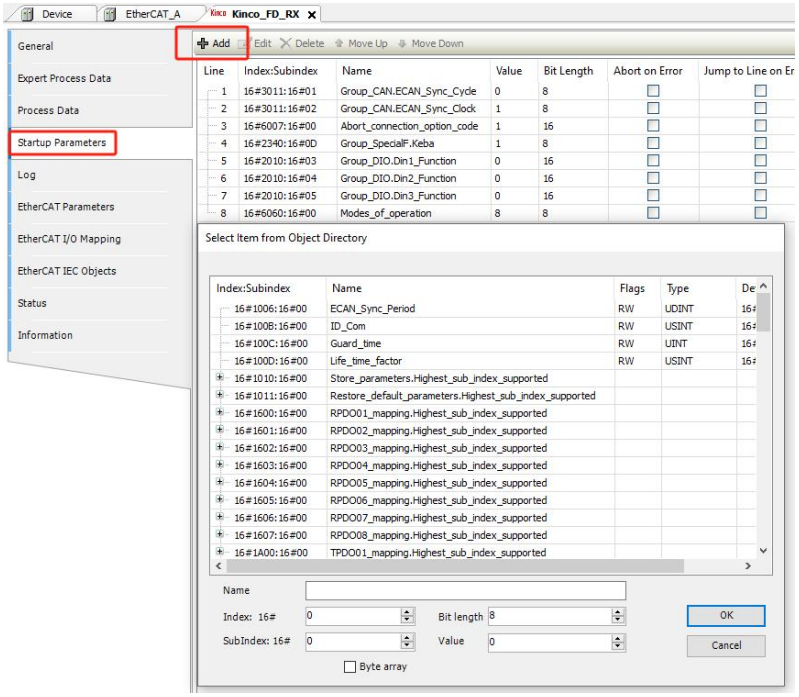


② If the default PDO does not meet your needs, you can add or delete it in the "Expert Process Data" screen.

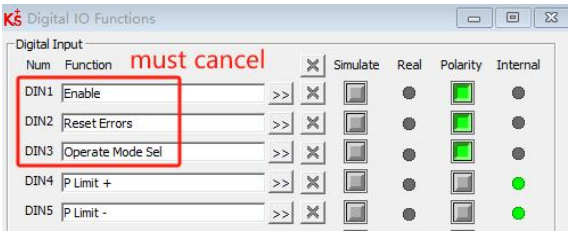


(4) Set startup parameters

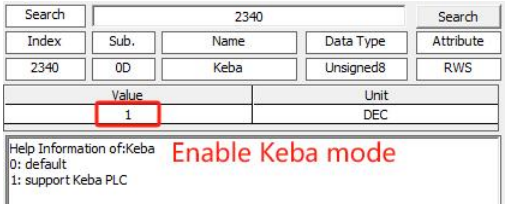
① Drive parameters can be set automatically by SDO at power-on. Click "Add" in the "Startup Parameters" interface to select the parameter to be set at power-on and set the parameter value.



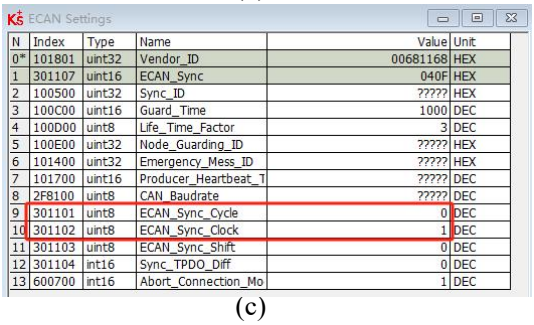
② Drive parameters need to be set for DIN, Keba mode (For details, see Table 3-1), synchronization cycle and synchronization mode, see the following figures. The synchronization cycle[301101] should be the same as the controller setting, 0 means 1ms, and the synchronization mode [301102] should be enabled. These parameters can be set in the "Startup Parameters" screen above.



(a)



(b)



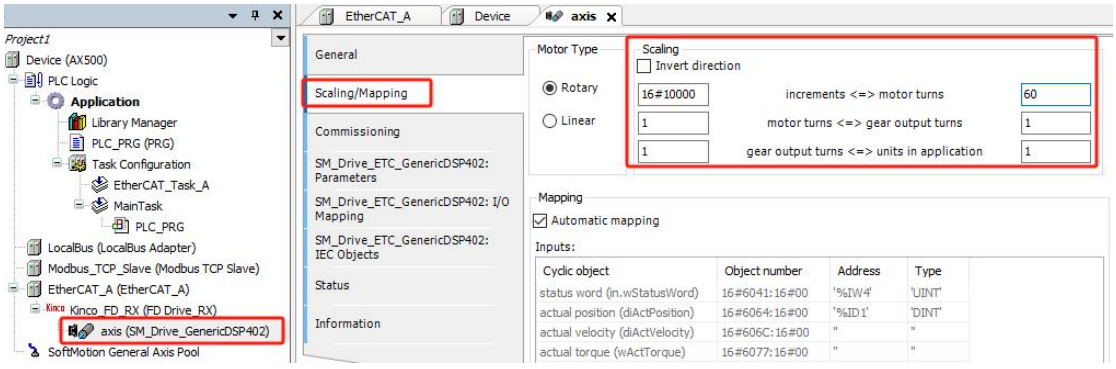
(c)

Table 3-1 Keba parameters Function description

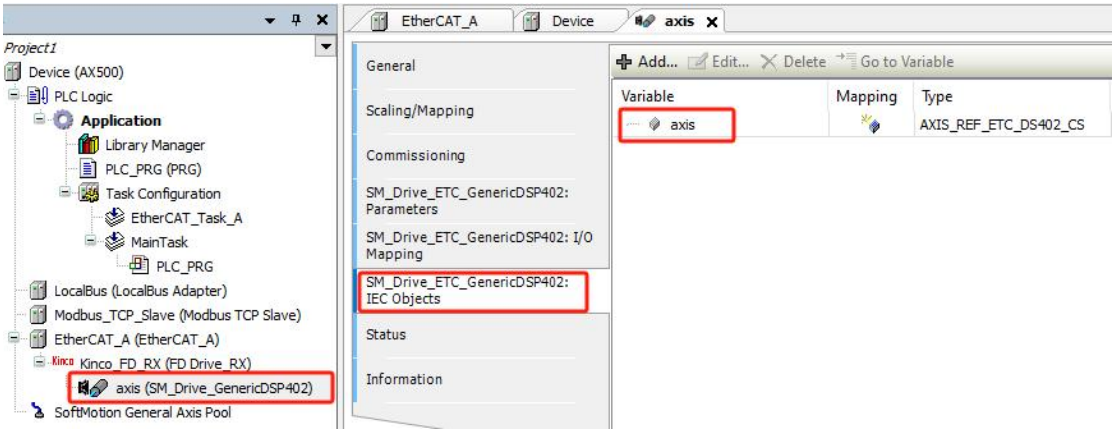
Keba	Description
0	When the servo is not enabled, the Operation_Mode_Buff [606100] remains 0, and the Operation_Mode [606000] is immediately updated to the Operation_Mode_Buff [606100] after the servo is enabled.
1	Whether the servo is enabled or not, the Operation_Mode [606000] is immediately updated to the Operation_Mode_Buff [606100]

6. Set the CiA402 axis parameter

(1) Set the scaling of the axis. Double-click the CiA402 axis and set it in "Scaling/Mapping".

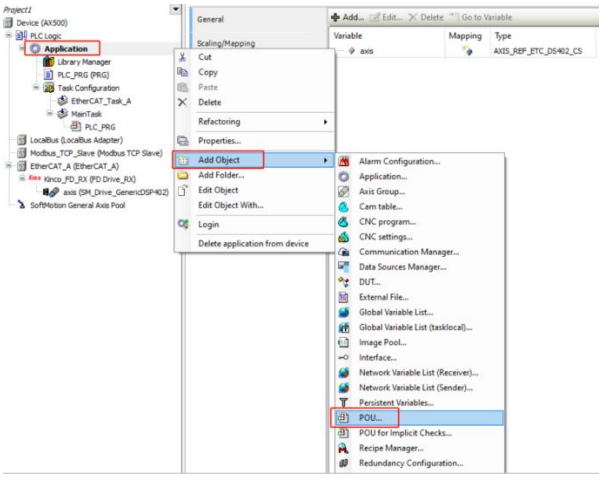


(2) Set the axis name. Set in "SM_Drive_ETC_GenericDSP402:IEC Objects", the name of the object is the axis name in the MC instruction.

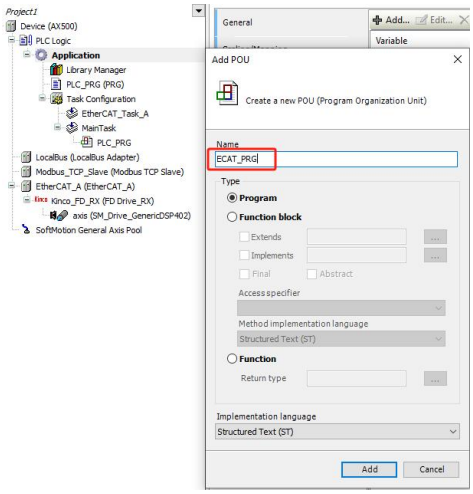


7. Write a PLC program

(1) Create a new POU program as shown in the following figures, right-click "Application" → "Add Object" → "POU... ", and enter the name of the POU.

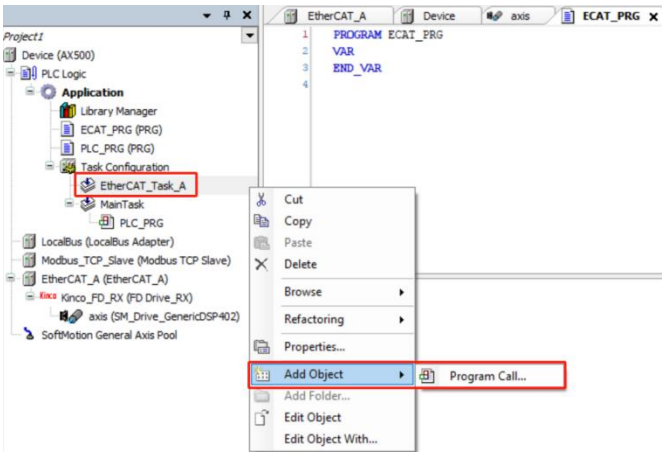


(a)

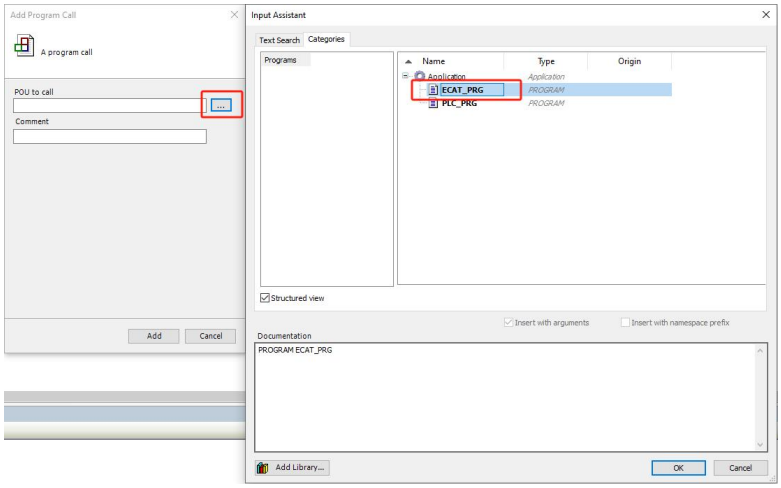


(b)

(2) The program "ECAT_PRG" is called in the task "EtherCAT_Task_A" as shown in the following figures.

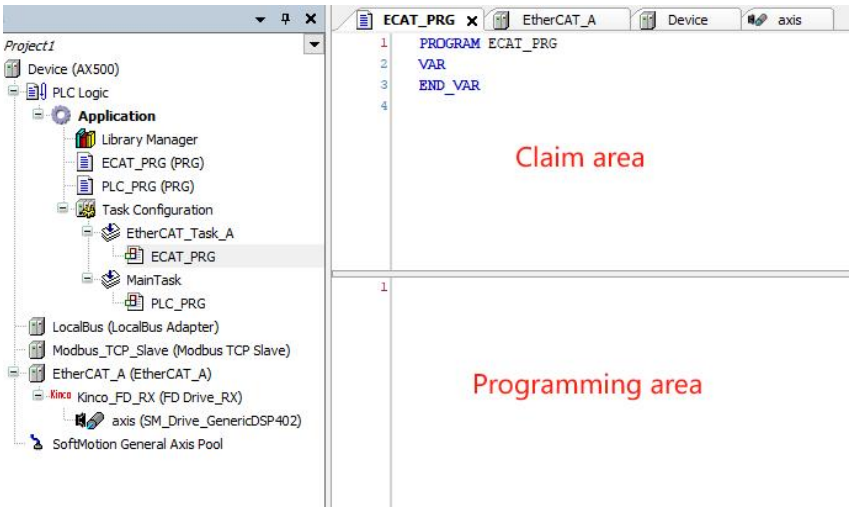


(a)

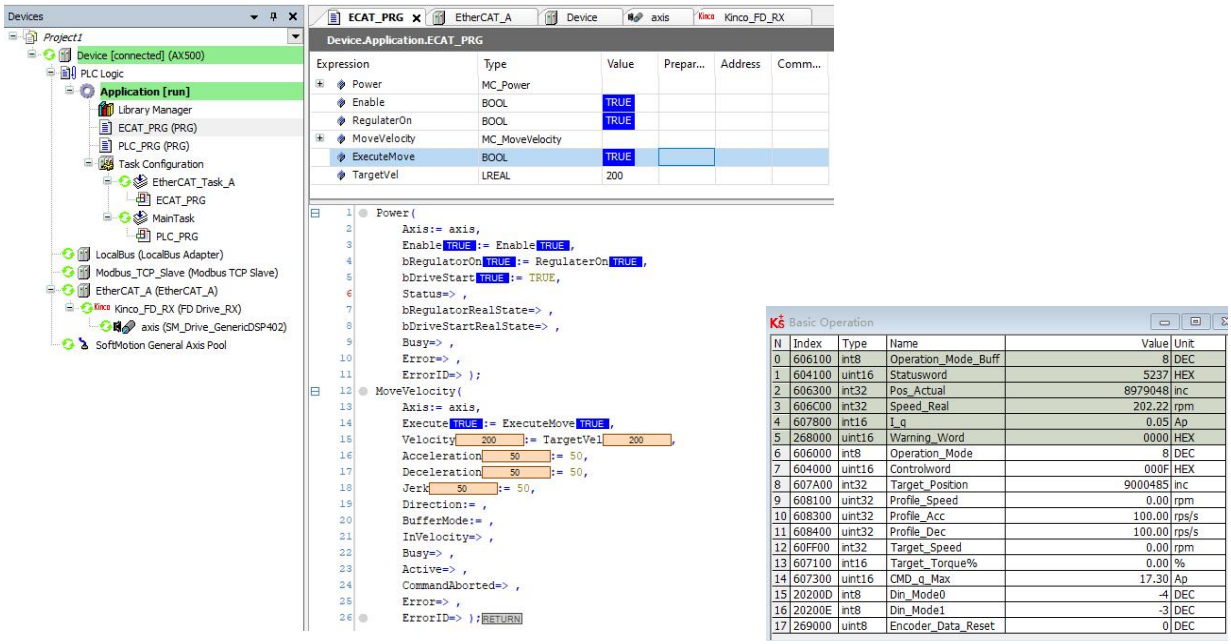


(b)

(3) Open "ECAT_PRG", the upper part is the declaration area, the lower part is the programming area.



(4) Write a program to control the drive using motion control instructions.



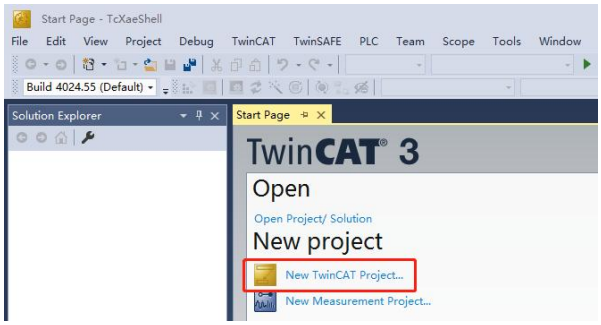
3.2 BECKHOFF TwinCAT 3 application

3.2.1 Direct servo control with TwinCAT

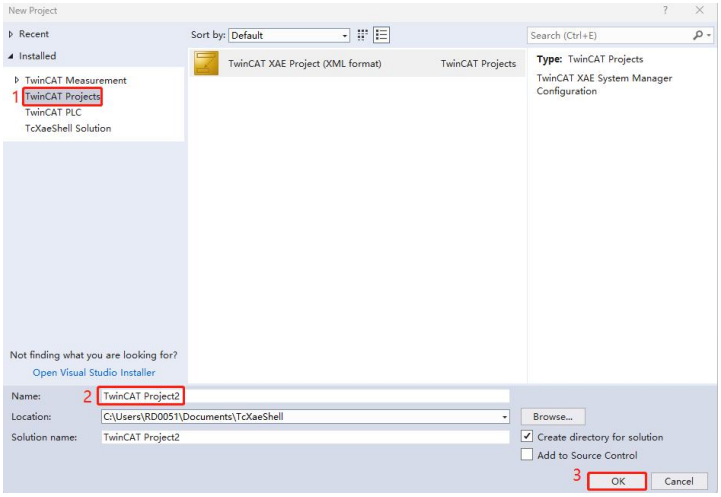
1. Create a new TwinCAT project

(1) Before starting TwinCAT, you need to copy the XML file of the drive to the installation directory of TwinCAT, the default path is C:\TwinCAT\3.1\Config\Io\EtherCAT.

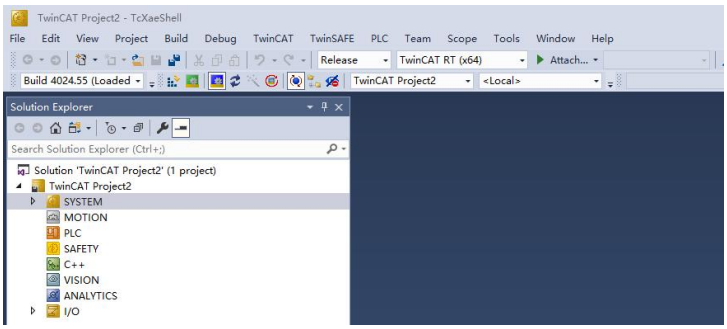
(2) Start TwinCAT and create a new TwinCAT project as shown in the following figures.



(a)



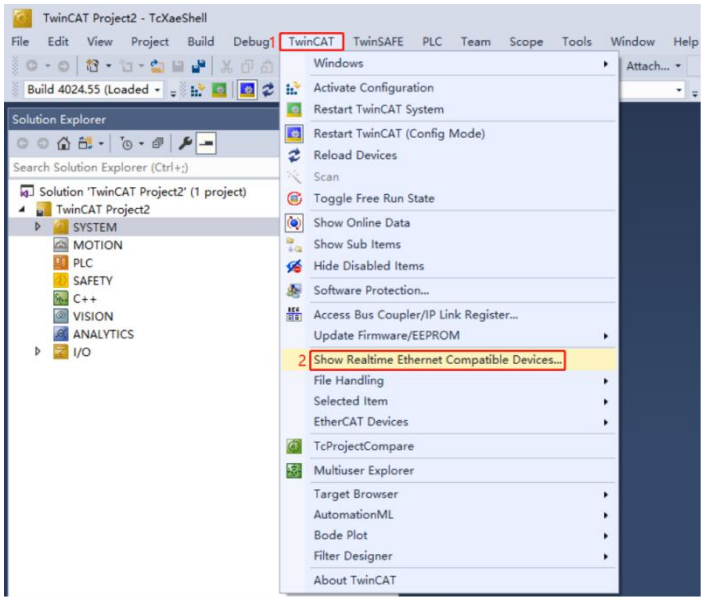
(b)



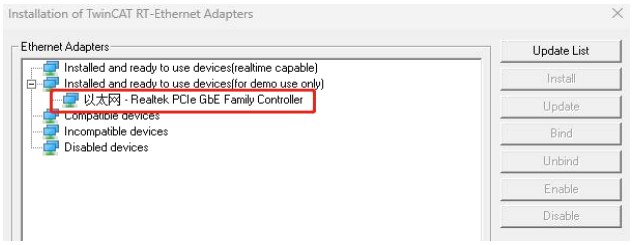
(c)

2. Install Ethernet real time driver

(1) TwinCAT can do NC control of servo directly without PLC, you need to install Ethernet real time driver before doing NC control.



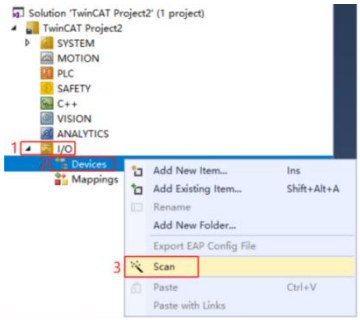
(2) As shown in the figure above, open the menu "TwinCAT" → "Show Realtime Ethernet Compatible Devices..." The following window is displayed. Select a local NIC (Network Interface Card) in the "Incompatible devices" column and click "Install". After successful installation, the NIC is displayed in "Installed and ready to use devices(for demo use only) " column:



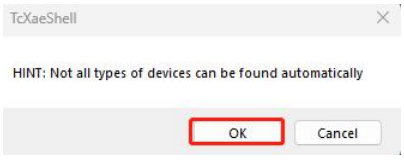
(3) After the Ethernet driver is installed successfully, you can scan for slaves. According to the official instructions of BECKHOFF, the computer NIC should choose Intel 100 Mbit/s NIC, other NICs are not guaranteed to successfully install the driver.

3. Scan for the slave

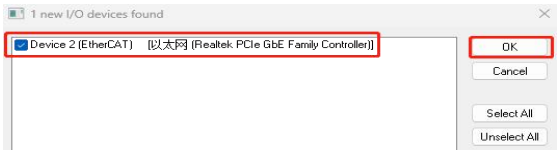
(1) Right click on "I/O" → "Devices" and click "Scan" to scan the slave.



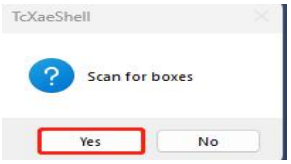
(2) As shown in the following figures, after scanning the slave, continue to click "Scan for boxes", and finally click "Append linked axis to NC - Configuration"



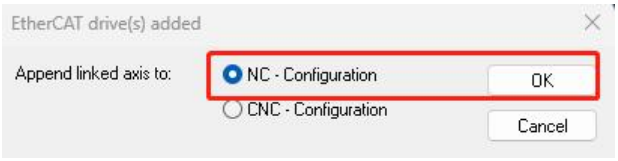
(a)



(b)

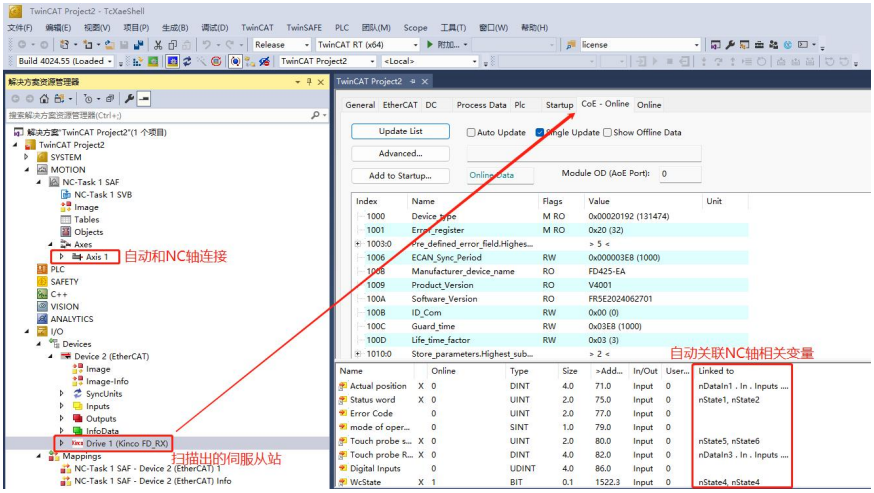


c)



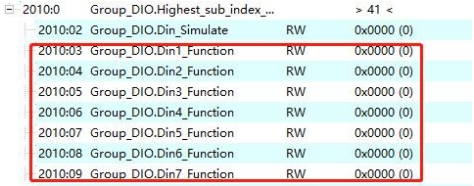
(d)

(3) After scanning the slave, you can see the slave information of "Drive1 (KincoFD_RX)" under "Devices", and it is automatically connected to the NC axis, TwinCAT automatically turns to the config mode.  If this blue gear does not appear, it will not be able to read and write SDO normally. The scanning result is shown below.



4. Set drive parameters via COE-Online

(1) In the COE-Online interface you can set the DIN of the drive. The default functions of DIN1-DIN7 that must be canceled are Enable (value 1), Reset Errors (value 2) and Operate Mode Select (value 4). Others such as Position Limit (value 0x10 and 0x20) can be set as needed. For details, see the IO definition chapter of the drive manual.



(2) Configure the appropriate motor for the drive, and fill in the object [641001] with the hexadecimal motor code. For details of the relevant motor codes, please refer to the motor chapter of the drive manual.

(3) Set the synchronization cycle [301101] of the drive, and enable the synchronization clock mode ([301102] value is 1), as below, the default has enabled the synchronization mode, and the synchronization cycle is 2ms ([301101] value is 1).

3011:0	Group_CAN.ECAN		> 8 <
3011:01	Group_CAN.ECAN_Sync_Cycle	RW	0x01 (1)
3011:02	Group_CAN.ECAN_Sync_Clock	RW	0x01 (1)
3011:03	Group_CAN.ECAN_Sync_Shift	RW	0x00 (0)
3011:04	Group_CAN.Sync_TPDO_Diff	RW	0

(4) As above, the drive parameters have been set. If it is the first time to set, you need to store the control parameters ([2FF001] is set to 1) and motor parameters ([2FF003] is set to 1), and it will take effect after restarting the drive with power off.

2FF0:0	Group_Store	RO	> 3 <
2FF0:01	Group_Panel.Store_Data	RW	0x01 (1)
2FF0:02	Group_Panel.Store_Calibrate_D...	RW	0x00 (0)
2FF0:03	Group_Panel.Store_Motor_Data	RW	0x01 (1)

(5) After setting the drive parameters, let the drive work in 8 mode ([606000] is set to 8), NC control is possible.

6060	Modes_of_operation	RW P	8

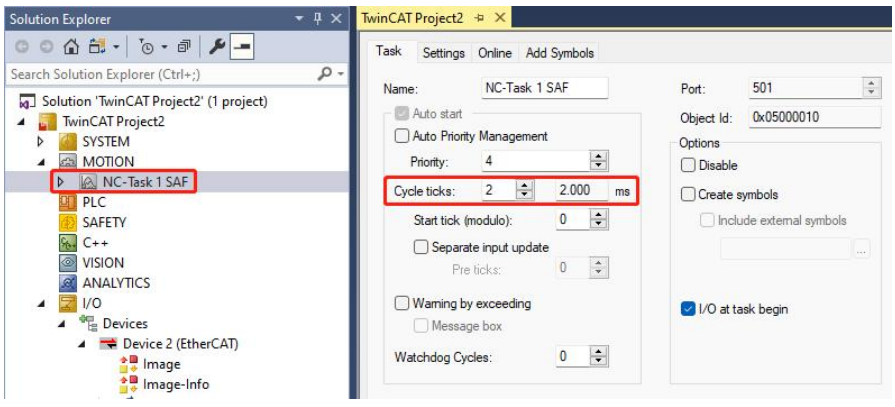
(6) If you need to adjust the drive PI and other performance parameters, you can set the speed loop (0x60F9 group) and position loop (0x60FB group), etc., for details, see the performance adjustment chapter of the drive manual. Note that the object units here are drive internal units, need to be converted, see the unit conversion chapter of the drive manual.

60F9:0	Group_Speed_Loop.Highest_sub_inde...	> 0 <
60FB:0	Group_Position_Loop.Highest_sub_ind...	> 1000 <

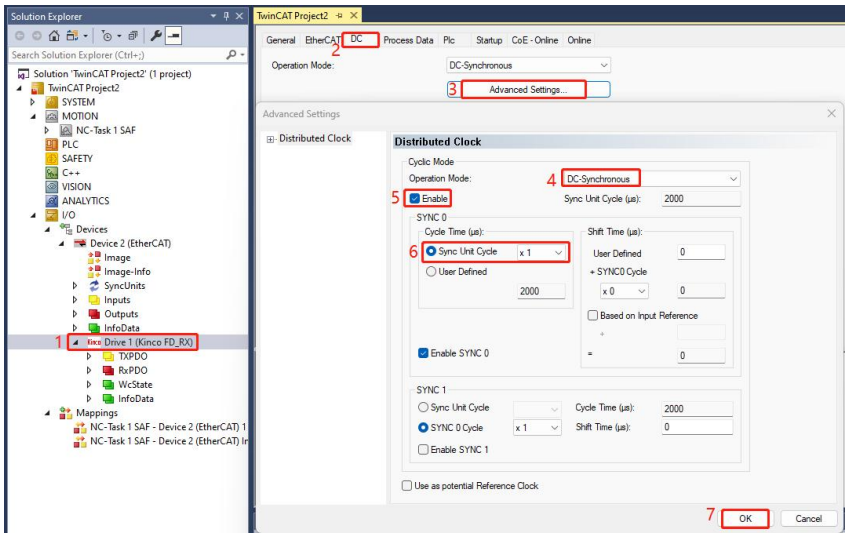
(7) If you can't find the object in COE-Online or want to monitor the drive more conveniently, please use the KincoServo software.

5. TwinCAT project setting

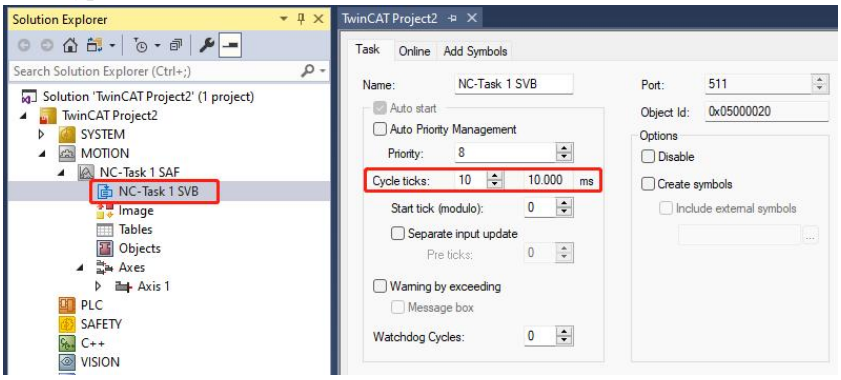
(1) Set the control cycle of the NC task, i.e. the cycle of the NC-Task SAF (the cycle in which the NC axis exchanges data with the drive), here set to 2 ms. In this task, TwinCAT NC completes the calculation of the set values for position, speed and acceleration.



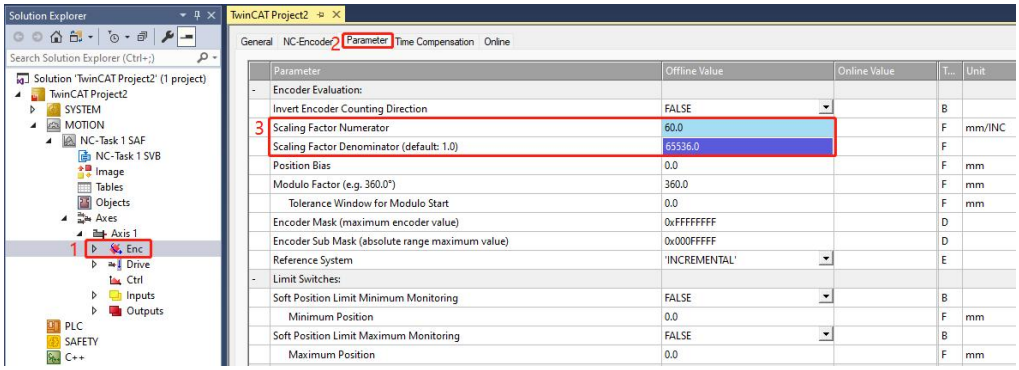
(2) Enable the distributed clock, note that the "Cycle Time" here should be the same as the ECAN synchronization cycle [301101] of the drive.



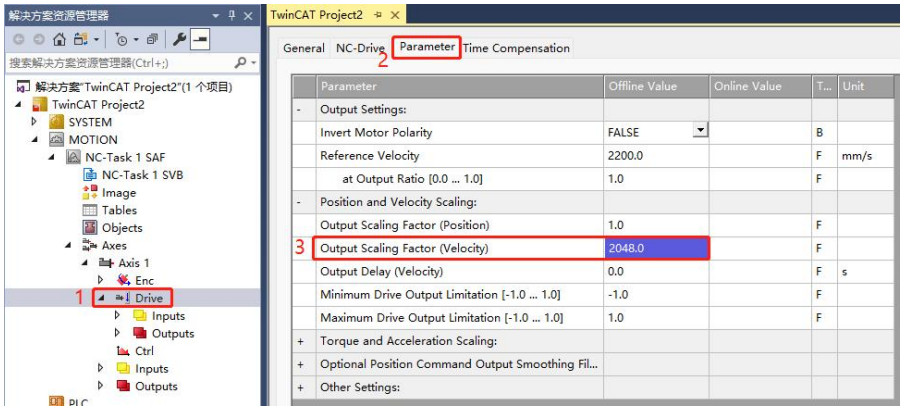
(3) The other NC cycle, the NC-Task SVB cycle, is the cycle in which the NC axis exchanges data with the PLC, with a typical value of 10ms.



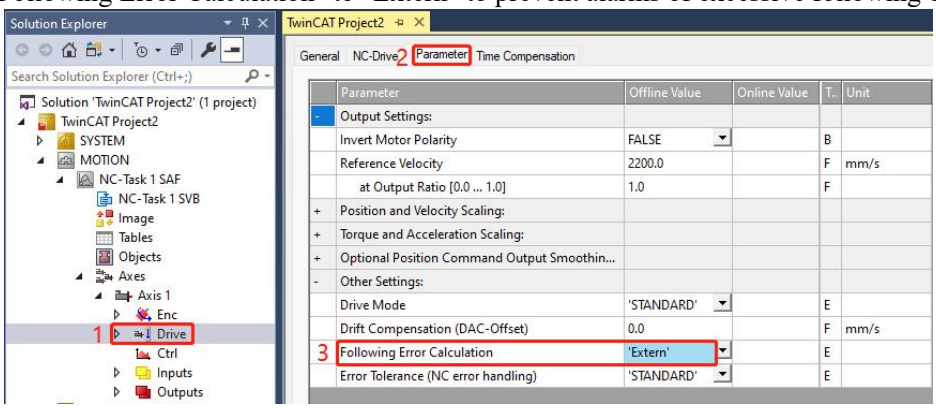
(4) In "Axis1" → "Enc", you can set the "Scaling Factor", i.e. the distance for each encoder pulse.



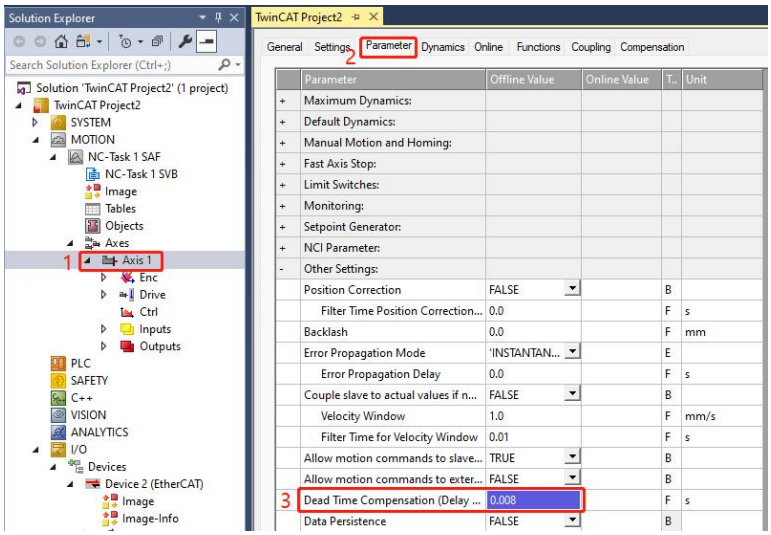
(5) In "Axis1" → "Drive", you can set the scaling factor of the set value.



(6) Set "Following Error Calculation" to "Extern" to prevent alarms of excessive following error.

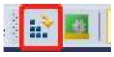


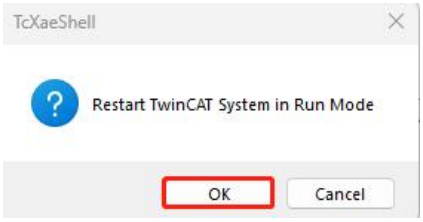
(7) Or increase the dead time compensation. Set the dead time compensation to an integer multiple of the synchronization cycle. 4 times the synchronization cycle is recommended. As shown in the following figure, if the synchronization cycle is 2ms, the dead time compensation value is set to 0.008s.



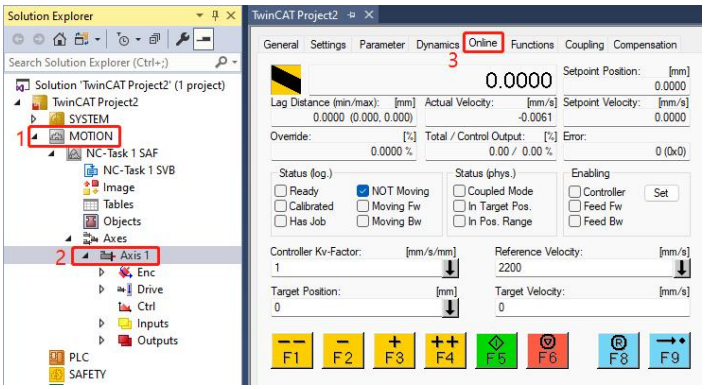
The above completes the basic settings of the drive in NC mode.

6. NC control

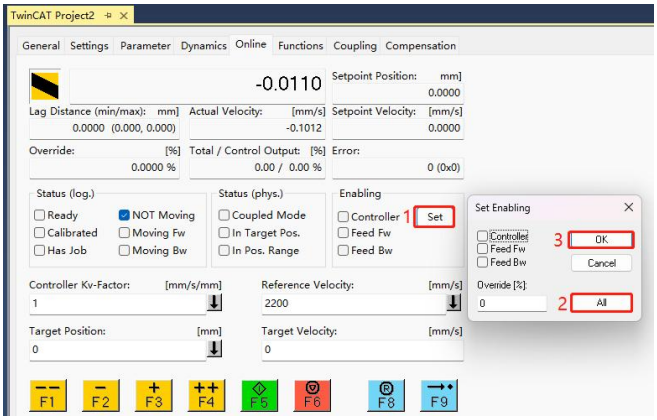
(1) After the above settings are completed, NC control can be carried out. First, Activate  Configuration, click OK to turn to run mode, as shown in the following figure.



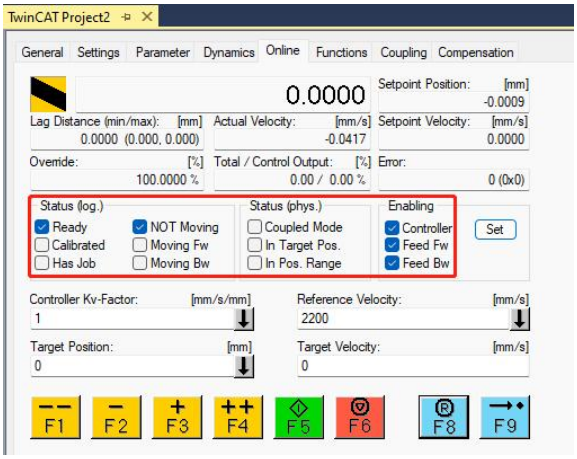
(2) Click on the "MOTION" → "Axis 1" → "Online" tab to commission the drive.



(3) Click "Set" in the current page, and then click "All" to enable the drive. The "Set Enabling" window allows you to set the axis enabling, forward and reverse rotation, and speed ratio.



(4) After enabling, the "Ready" status will be ticked, press F1 to F4 to perform the jog operation, the jog speed is set in "Manual Velocity" in "MOTION" → "Axis 1" → "Parameter" tab, the default speed is 100 mm/s and 600 mm/s, which corresponds to slow and fast jog respectively.



The above is an example of direct servo control with TwinCAT, more TwinCAT NC control methods see the official manual of BECKHOFF.

3.2.2 Servo control with PLC

When BECKHOFF PLC communicates with a single drive, use the network cable to connect the EtherCAT port of the PLC and the IN port of the drive; when the PLC connects with multiple drives, connect the OUT port of the previous drive to the IN port of the next drive as shown in the figure below (take BECKHOFF CX5020 as an example).

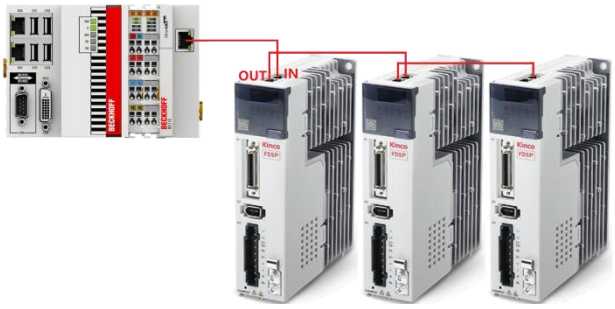
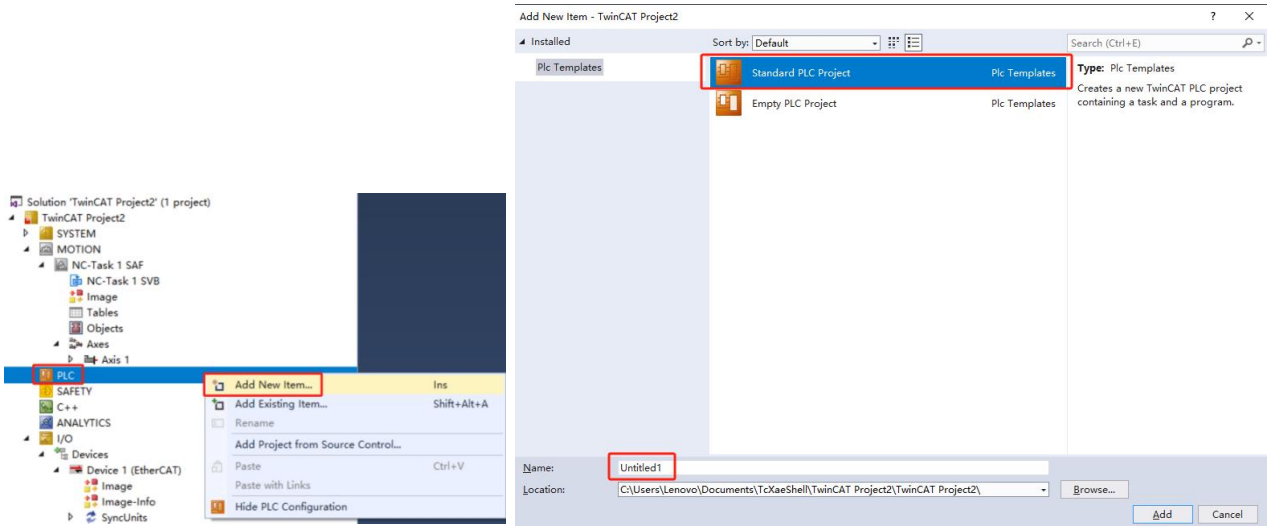
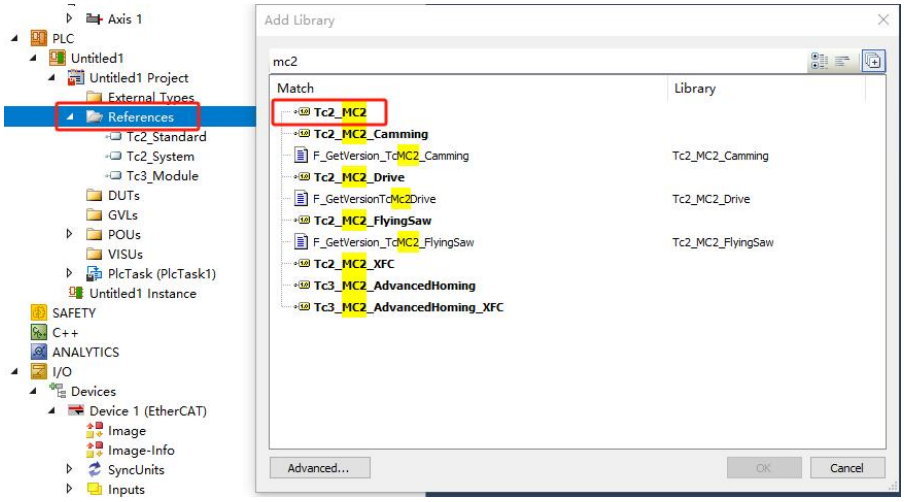


Figure 3-2 BECKHOFF CX5020 connecting drives

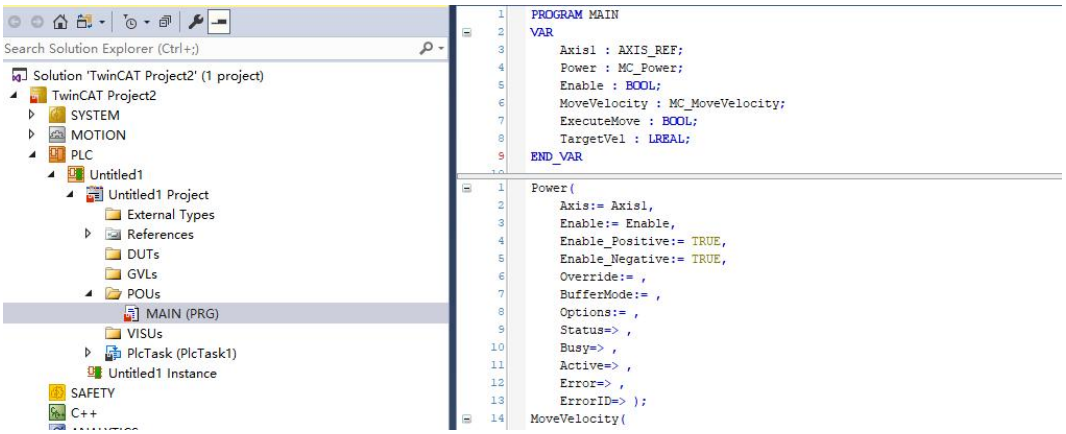
1. Create a new PLC project



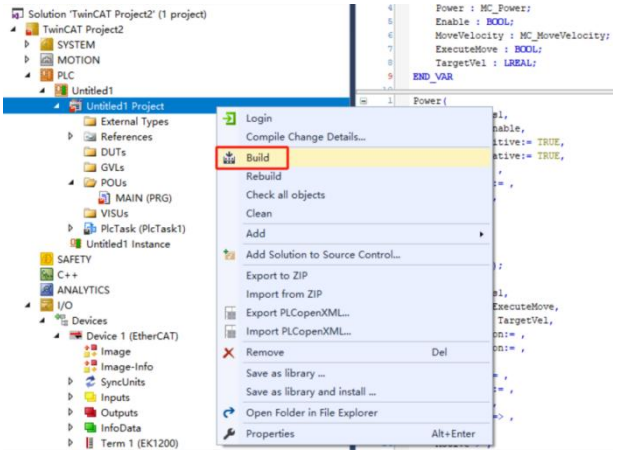
2. Add the motion control library "Tc2_MC2"



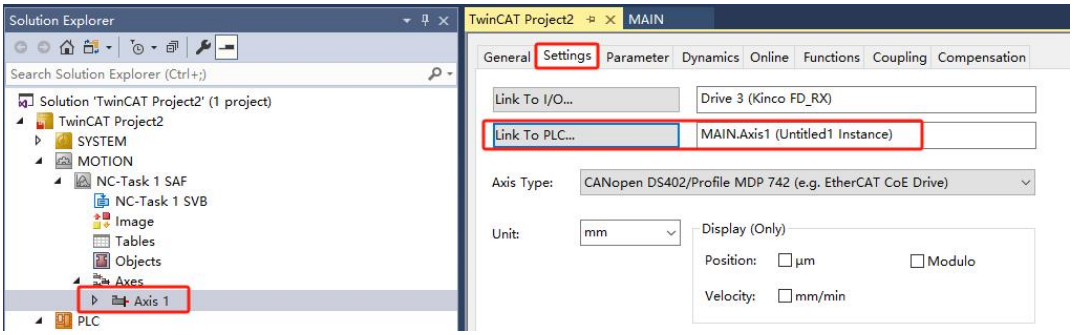
3. Call the function blocks and write a program



4. Compile the program

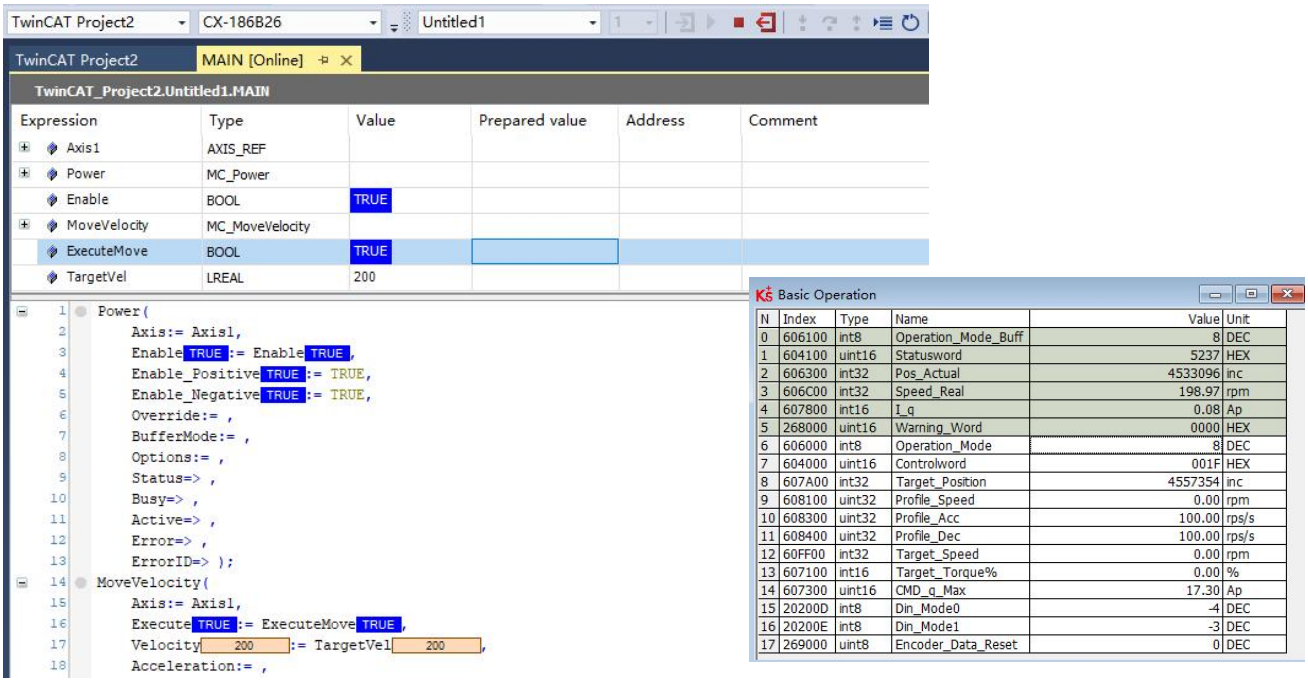


5. Associate the NC axis with the PLC axis



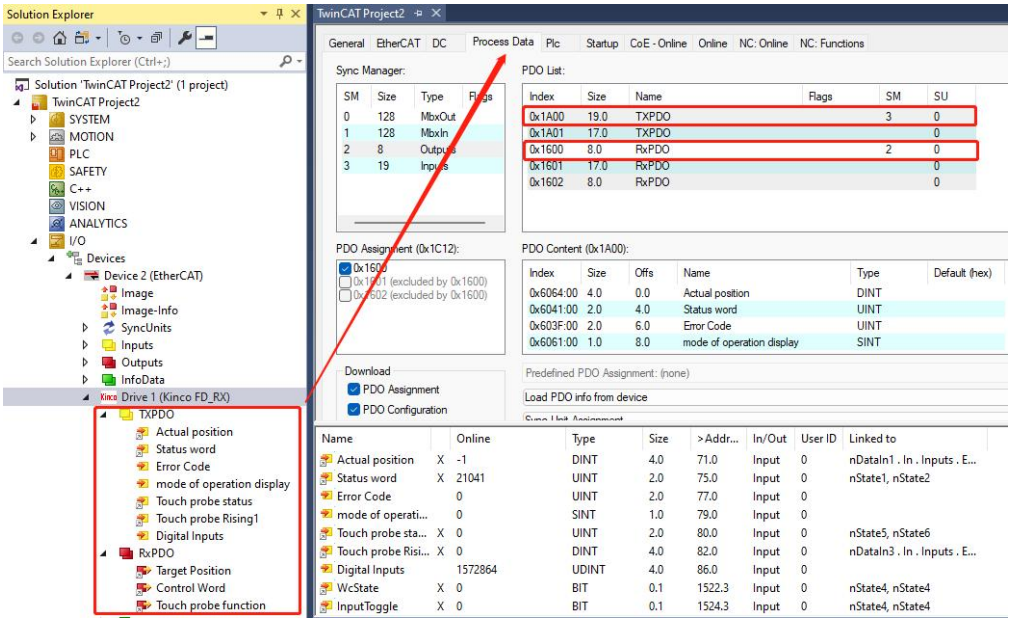
6. Log in to the PLC after activating the configuration

7. Run the program to control the servo motion

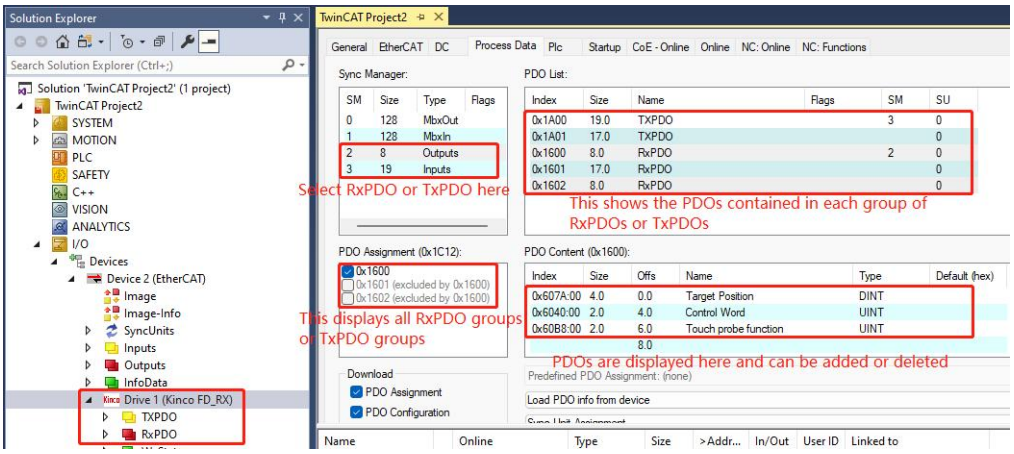


3.2.3 PDO configuration

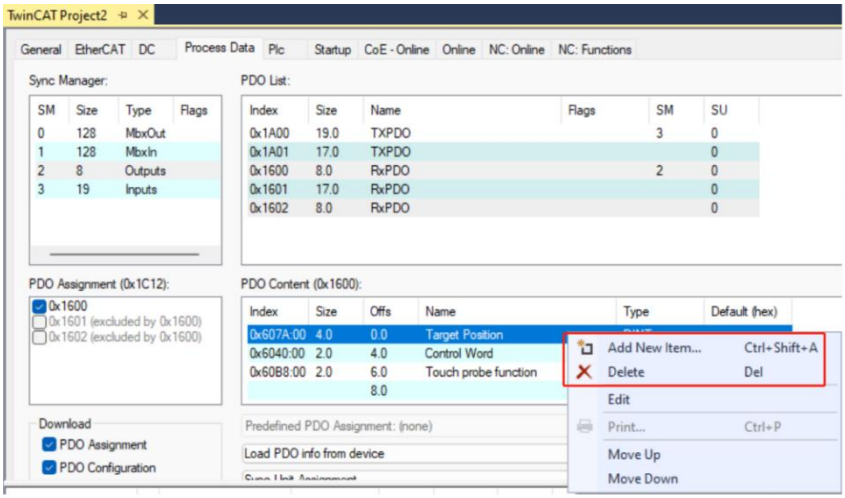
1. The following figure shows the default RxPDO group (0x1600) and TxPDO group (0x1A00) of the drive. The default parameters are sufficient for NC control.



2. If the default PDO group does not meet the requirements, then refer to the following figure to select another PDO group.

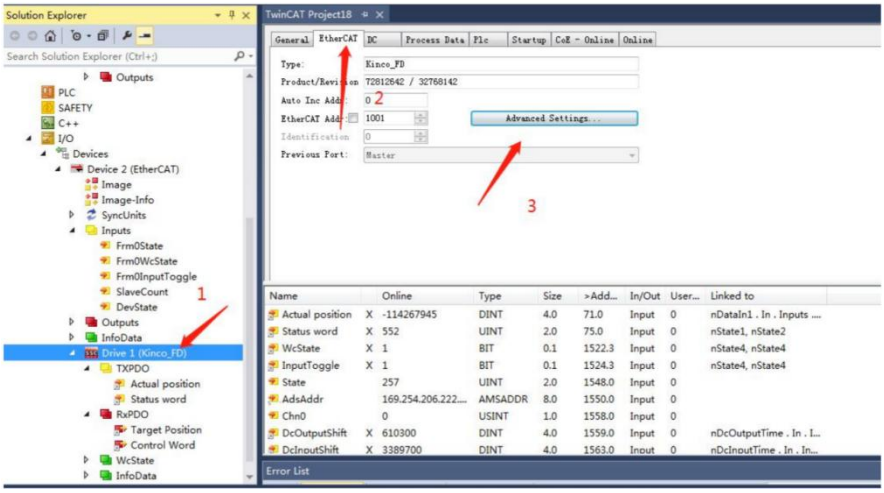


3. If you do not have the desired object in the PDO group, you can add or delete it. Right-click in the "PDO Content" window to remove the default PDO and add the required PDO.

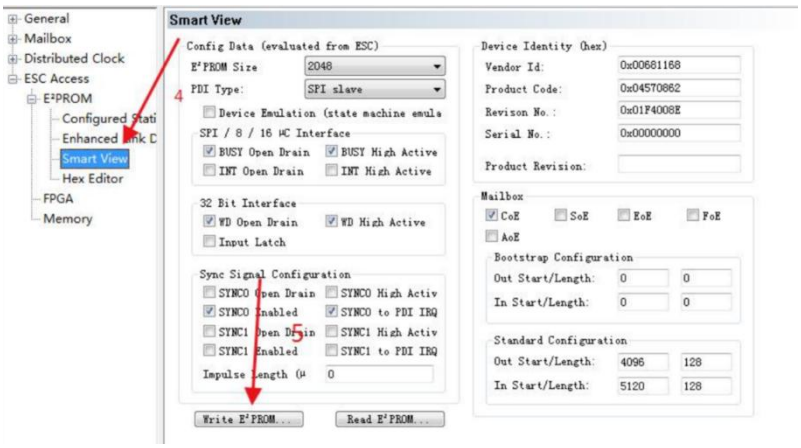


3.2.4 Write slave E2PROM

1. If TwinCAT has already scanned the slave, but the slave Drive1 (Kinco FD) has a question mark next to it or the PDO cannot be read or written due to a mismatch in the XML file, then you need to write the correct XML into the drive's E^{2}PROM, as shown in the following figures.



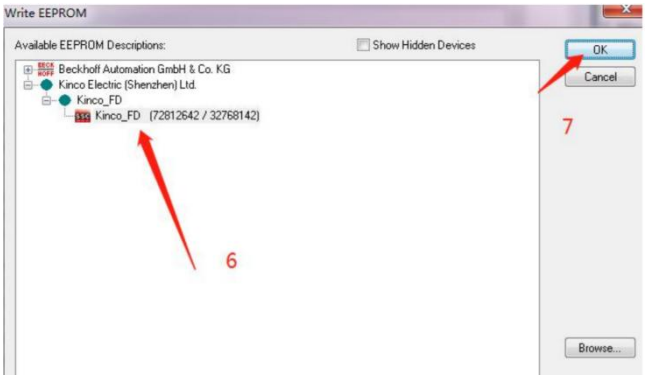
(a)



(b)

Note: Before writing E2PROM, you need to copy the drive XML file to the TwinCAT installation directory, default path C:\TwinCAT\3.1\Config\Io\EtherCAT.

2. After clicking "Write E²PROM" the following window will pop up, in the window will appear in the list of XML has been added, select the correct XML, click OK to download.



3. At the same time, there is a progress  bar in the lower right corner of the TwinCAT window, please observe the status of the progress bar. After the operation is finished, please power off and restart the PLC and drive, and re-scan the slave to make the newly written XML take effect.

3.3 OMRON NJ series controller application

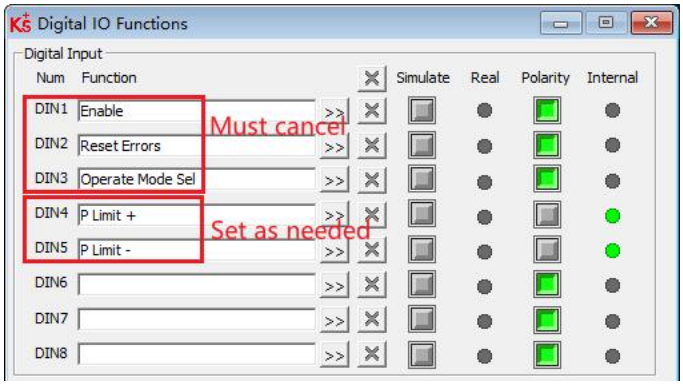
When Omron NJ series PLC controller communicates with a single drive, directly use the network cable to connect the EtherCAT port of the controller and the IN port of the drive; when the controller connects with multiple drives, connect the OUT port of the previous drive to the IN port of the next drive, as shown in Figure 3-3.

Figure 3-3 Omron NJ controller connecting drives

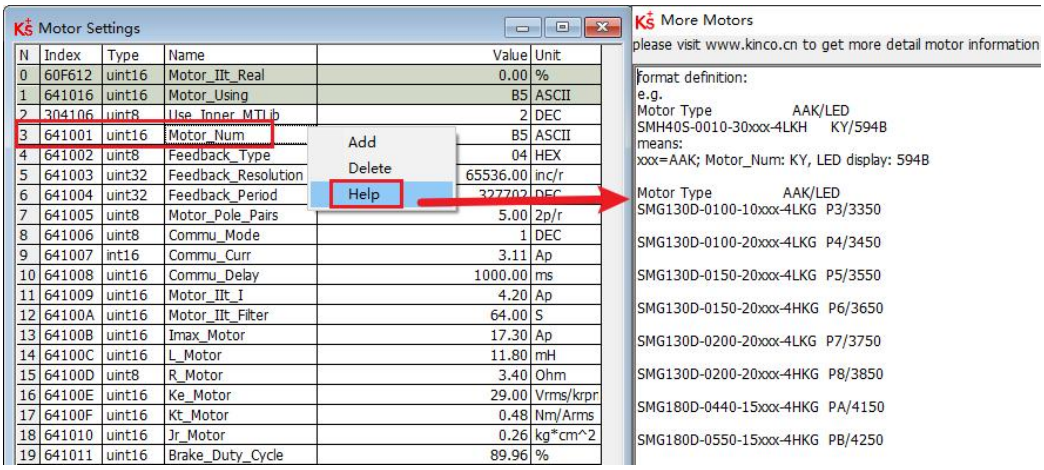
3.3.1 Drive parameters setting

Use KincoServo software to set the drive parameters, refer to the drive user manual for details, and only the parameters that must be set and the function description of commonly used parameters are listed below.

1. Cancel the default IO port function definition of the drive. Click Toolbar → Driver → Digital IO Functions, as shown below. Enable, Reset Errors and Operate Mode Set must be canceled, others such as Position Limit can be set as needed. See IO Definition section of drive user manual for details.



2. Configure the corresponding motor, click Toolbar → Motor → Motor Settings, as shown below. Just fill in the hex code of the corresponding motor in the object of the Motor_Num in the diagram. The motor code can be obtained by querying the motor section of the drive user manual or by right-clicking the help.



3. Set the sync cycle and enable the sync mode. Click Toolbar → Driver →ECAN Configuration → Others, as shown below. The drive has enabled the sync clock with a sync cycle of 2 ms (value of 1), 1 ms (value of 0) and 2 ms (value of 1) are recommended, 4 ms and 8 ms lose the meaning of using EtherCAT. Please note that the drive only supports the four sync cycles of 1, 2, 4, and 8 ms, but the PLC generally supports more, and only one of the four supported by the drive can be selected at the PLC. The shift of the sync point can adjust the jitter when the sync signal of the PLC and the command signal are not synchronized. Generally, it is recommended to set it to 1, indicating that the drive receives the command signal after the sync signal is shifted by 62.5 us.

9	301101	uint8	ECAN_Sync_Cycle	0	DEC
10	301102	uint8	ECAN_Sync_Clock	1	DEC
11	301103	uint8	ECAN_Sync_Shift	0	DEC

4. After setting the drive parameters, save the control parameters and motor parameters. Click Toolbar → Driver → Init Save Reboot.

5. If you need to adjust the PI and other parameters, click Toolbar → Driver → Control Loops to set the velocity loop and position loop parameters. See the performance adjustment section of the drive user manual for details.

3.3.2 Controller parameters setting

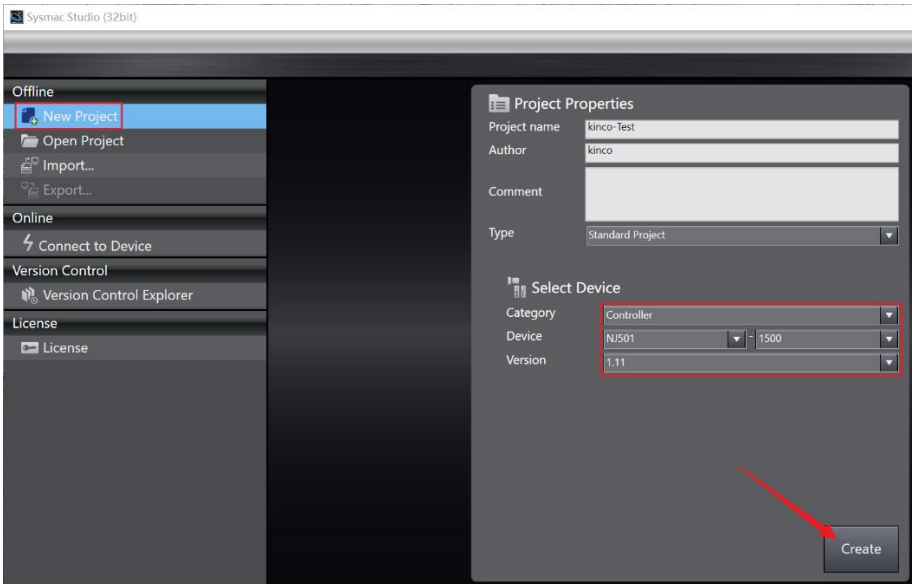
1. The drive XML file can be downloaded from the official website of Kinco:
   https://www.kinco.cn/download/sfqdqdj36.html

2. Copy the drive XML file to the NJ series programming software Sysmac Studio installation directory, such as:
   C:\Program Files\OMRON\Sysmac Studio\IODeviceProfiles\EsiFiles\SystemEsiFiles
   When the right toolbar can not find the drive slave, you can add the XML file to:

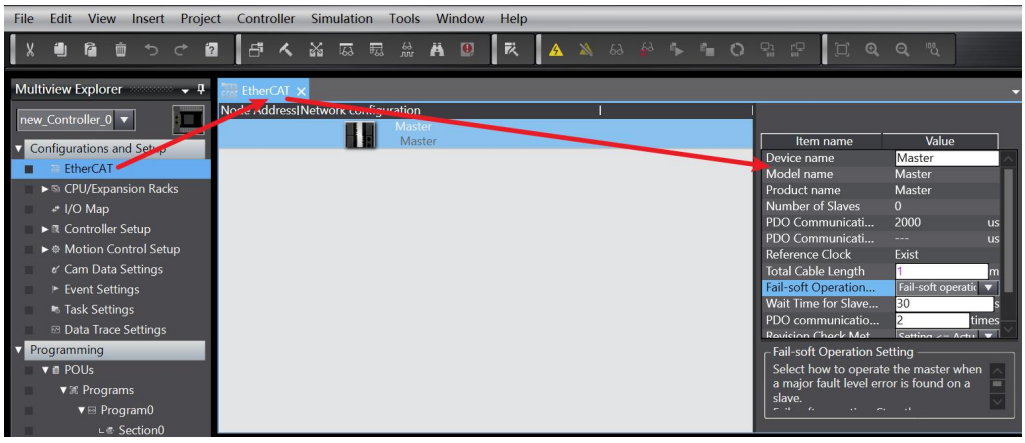
C:\Program Files\OMRON\Sysmac Studio\IODeviceProfiles\EsiFiles\UserEsiFiles

If you need to add or change XML files, please exit Sysmac Studio first. The XML files will not take effect until Sysmac Studio is restarted.

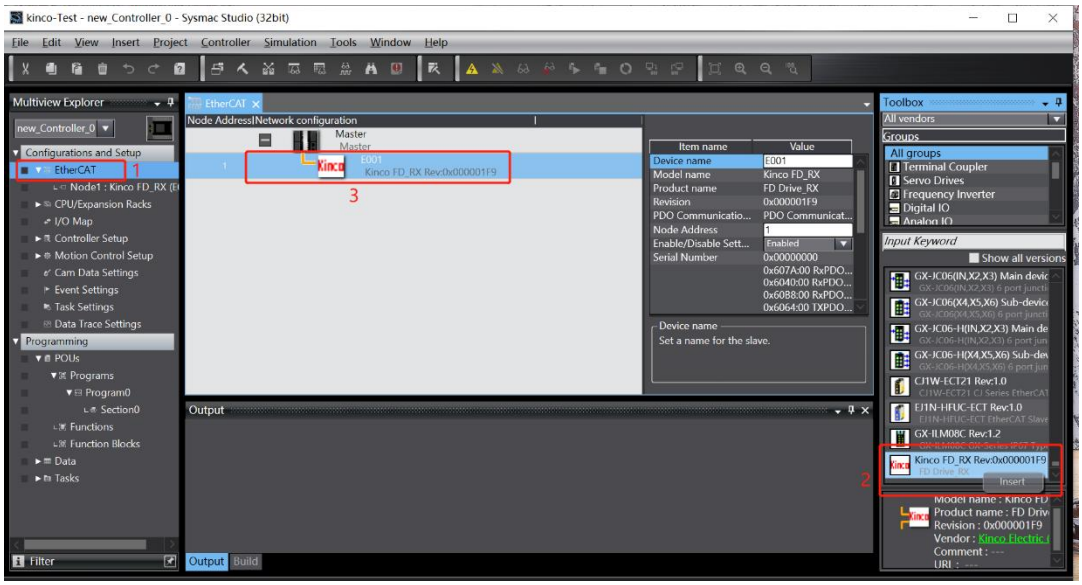
3. Connect the EtherNet port of the computer and the controller with a network cable, open Sysmac Studio, click New Project, and select the device and version information of the connected controller (found in the Controller product TAB).



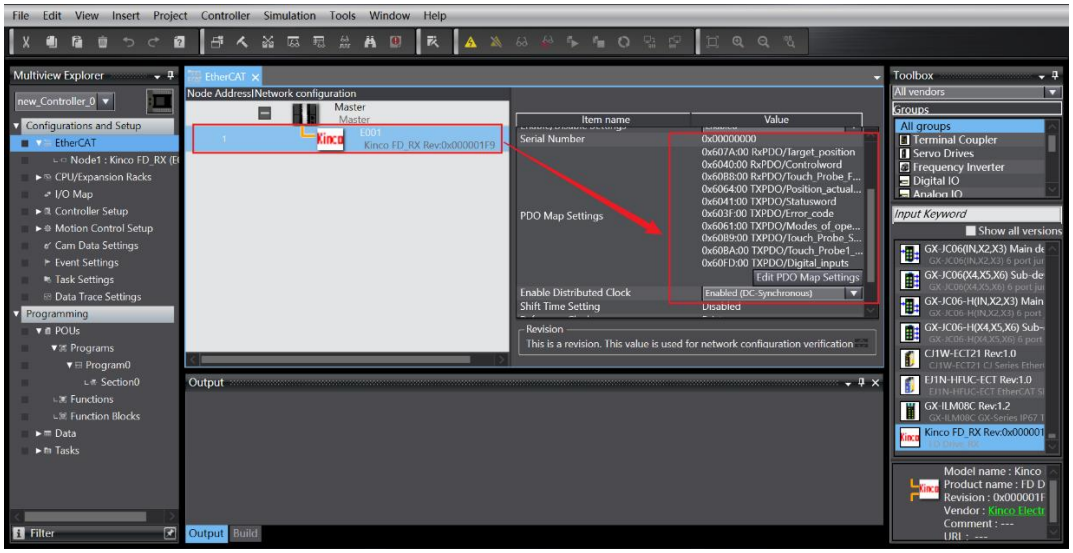
4. After creating the project, double-click "EtherCAT" in "Configurations and Setup" on the left, and then bring up the master view. Then set the parameters of the right master station (generally, the parameters can be set by default).



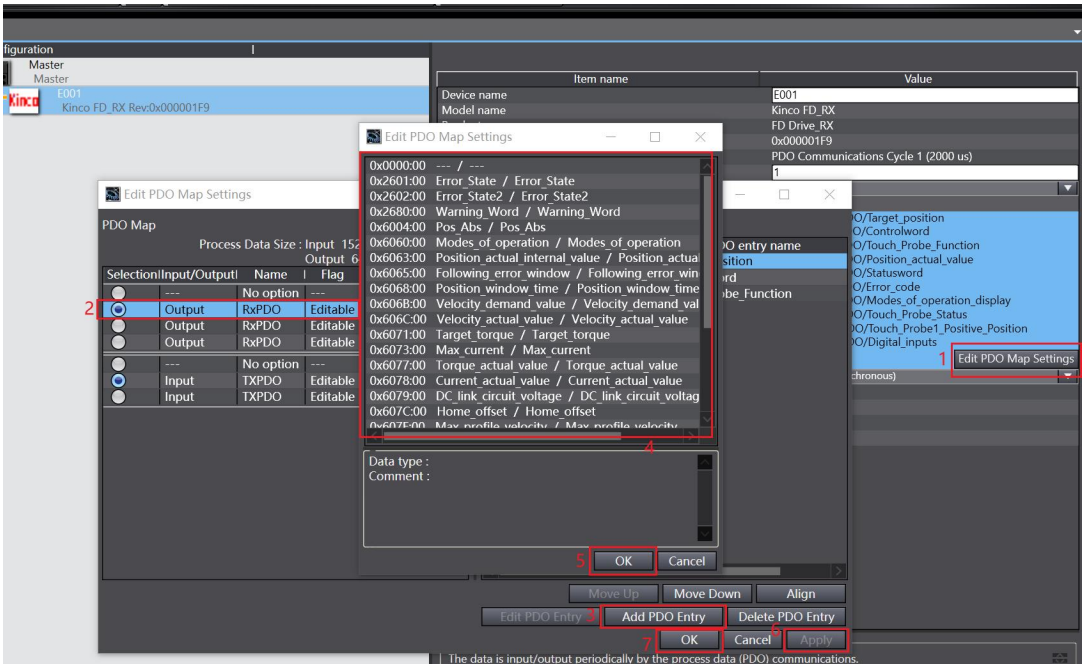
5. Find the kinco slave at the bottom of the right toolbox, right click Insert, add the connected slave (insert a few of the actual connections), after adding the slave, the main view area and the left configuration area will display the slave and assign the node address, as shown in the following figure (the test takes the single axis as an example).



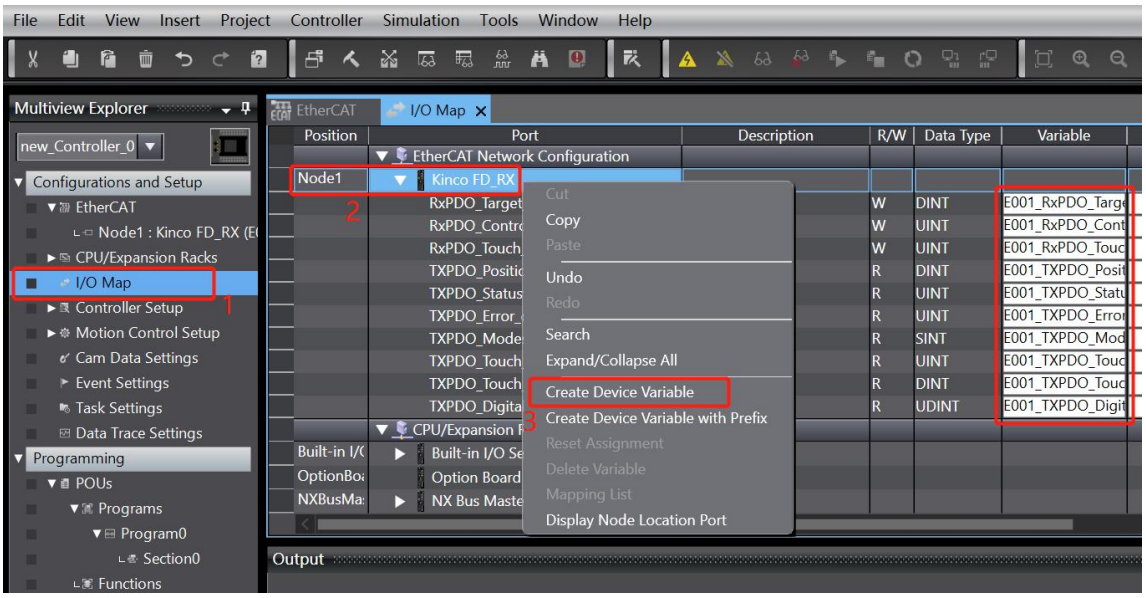
6. Click the slave to set the slave parameters. Generally, the default parameters in the box as shown below can meet the basic motion control requirements. If the parameters of multiple slave stations are set, you can right-click to copy → paste after setting one slave parameter.



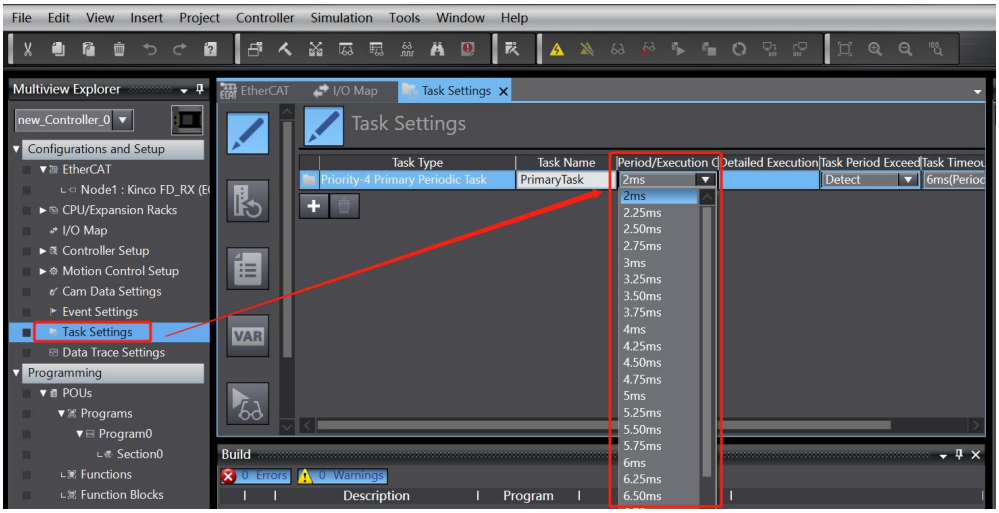
If there are other control requirements (for example, if there is no motion control, most of the PDOs need to be configured only as an ordinary EtherCAT slave), you need to add more PDOs. Click Edit PDO Map Settings, select output RPDO channel or input TPDO channel, click Add PDO Entry, and select the PDO to be added in the PDO menu bar that pops up. Then click Apply and OK.



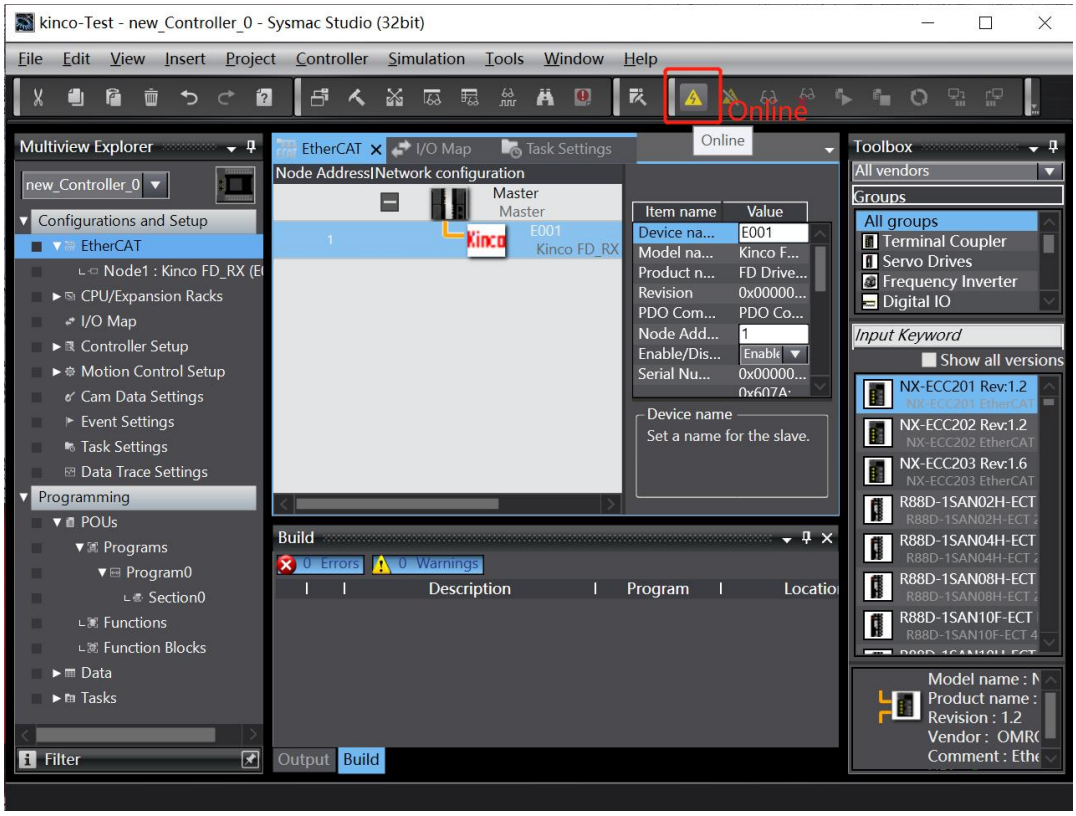
7. Associate the drive PDO to the controller local IO. In the left menu bar Configuration and Setup, double-click the I/O Map, then select the slave (note that only one slave can be set at a time, all slaves need to be set), right-click and select Create Device Variable, and the system will allocate the local IO variables.



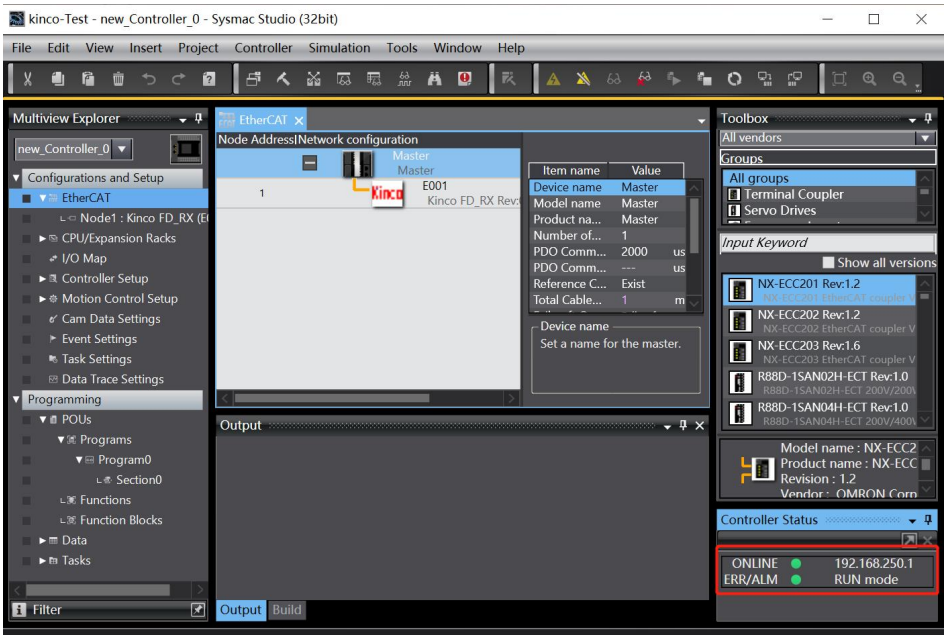
8. Sets the period in the task settings. In the Configurations and Setup area, double-click Task Settings to set the period of the main fixed-cycle task. Note that this task cannot be deleted and the period must correspond to the sync cycle set by the drive. Otherwise, it will cause the loss of sync messages and cause the motor operation oscillation!



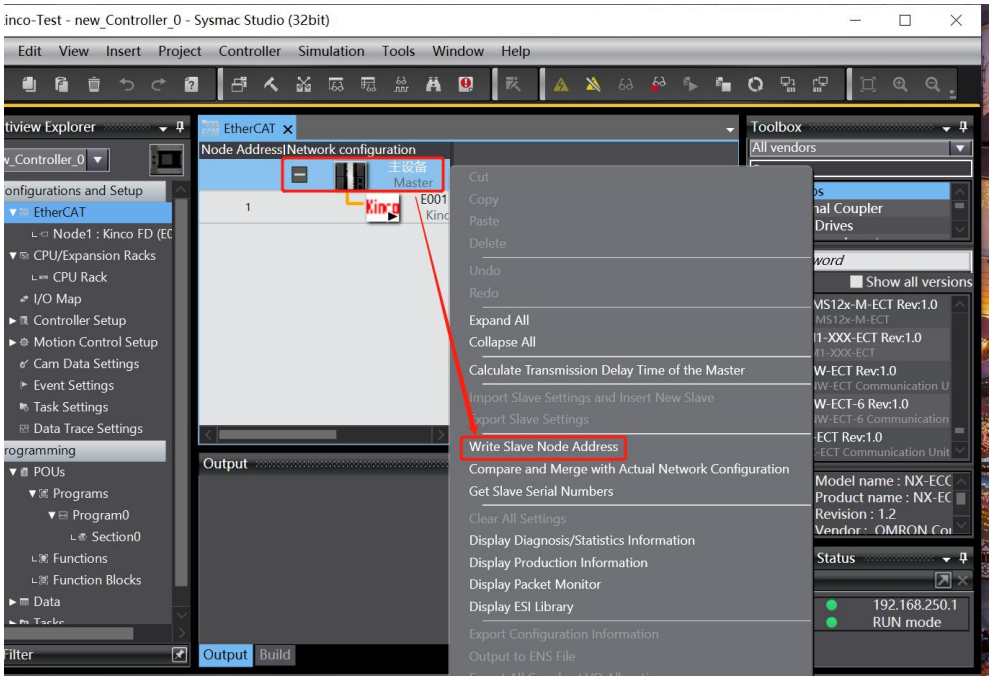
9. Select Project → Check All Programs for errors.
10. The basic communication parameters are set, and the program can be downloaded and debugged:
    (1) Select Controller → Communication Setup, select the programming port used (such as network port).
    (2) Select Controller → Online. After online, the controller status will be displayed in the bottom right corner.



(3) Select Controller → Synchronize. After synchronization, the comparison between the controller and the current local engineering program and parameters will be displayed. Select transfer to the controller, select "Yes" in the prompt box to enter the programming mode, and click "Yes" after the transfer is completed to enter the running mode.

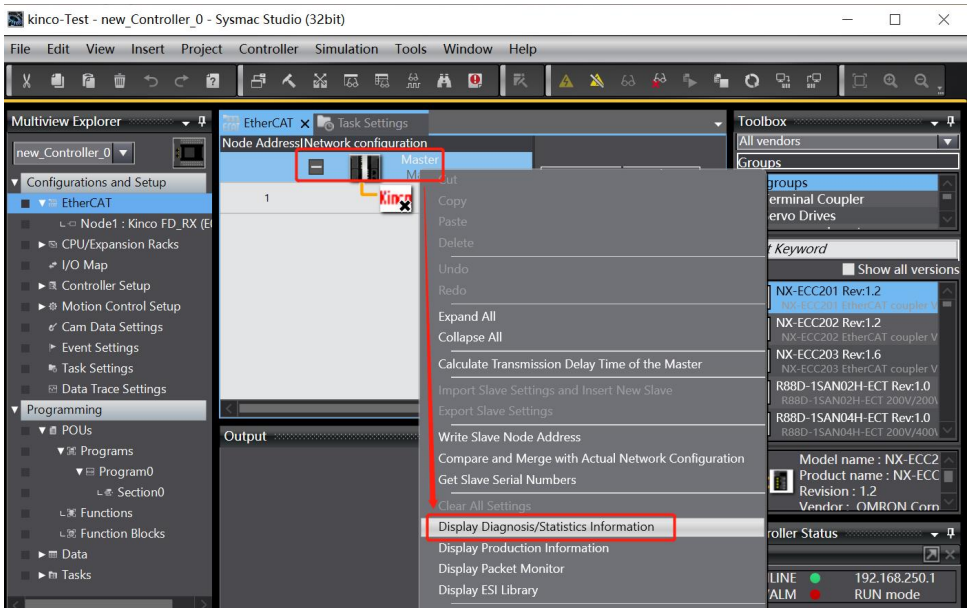


11. If more than one drive is used, you can go back to the EtherCAT settings page, select the master, right-click to reallocate the slave node address, and assign the slave node address in the settings window that appears. Generally, the slave address is assigned in order from 1 to 32, etc. Set it and click Write Slave Node Address.

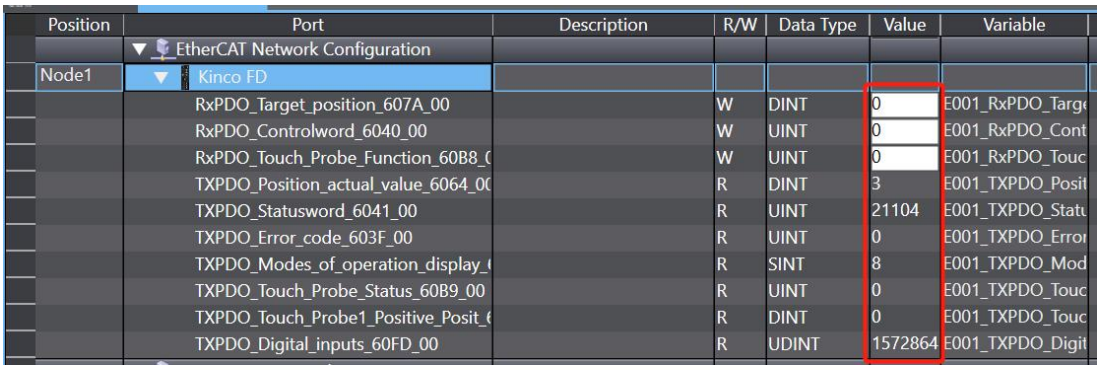


12. Power off and reboot the controller and drive, power on and online again. The monitor shows the slave station has been connected normally.

If the connection is wrong, the box below shows a cross instead of a triangular arrow. Click the Master, right click to display diagnosis/statistics information, can monitor the network status. If there is an error, you can select the Toolbar → Tools → Troubleshooting to see the specific error information.



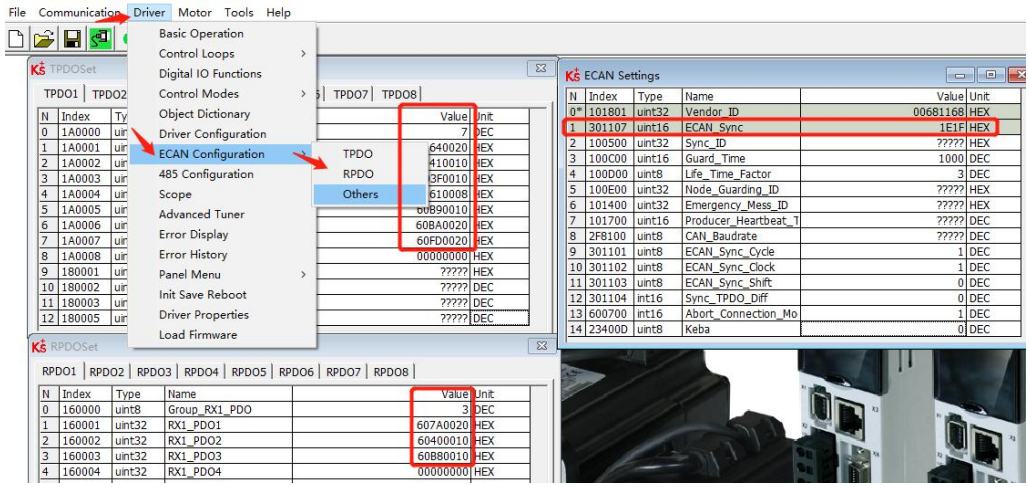
13. The corresponding slave data can be viewed and read in the IO map, indicating successful communication.



Can also view the three indicators of the EtherCAT port in the lower right corner of the controller body. The meaning of the status of each indicator is as follows:

Indicator	Color	Status		Meaning
NET RUN	Green		Lit.	EtherCAT communications are in progress. • Inputs and outputs for I/O data are in operation.
			Flashing	EtherCAT communications are established. This indicator shows either of the following conditions. • Only message communications are in operation. • Only message communications and I/O data inputs are in operation.
			Not lit.	EtherCAT communications are stopped. • The power supply is OFF or the CPU Unit was reset. • A MAC address error, communications Controller error, or other error occurred.
NET ERR	Red		Lit.	A hardware error or unrecoverable error occurred, such as for exception processing.
			Flashing	A recoverable error occurred.
			Not lit.	There are no errors.
LINK/ACT	Yellow		Lit.	A link is established.
			Flashing	Data communications are in progress after establishing link. Flashes every time data is sent or received.
			Not lit.	The link was not established.

At this time, in the KincoServo software, click Driver → ECAN Configuration, you can see the PDO information of successful configuration, as shown below. You can also see whether there is sync data in the "Others" (the ECAN_Sync is not 0, which means there is sync data and has been synchronized).



According to the above information, it means that the controller has successfully communicated with the EtherCAT of the drive.

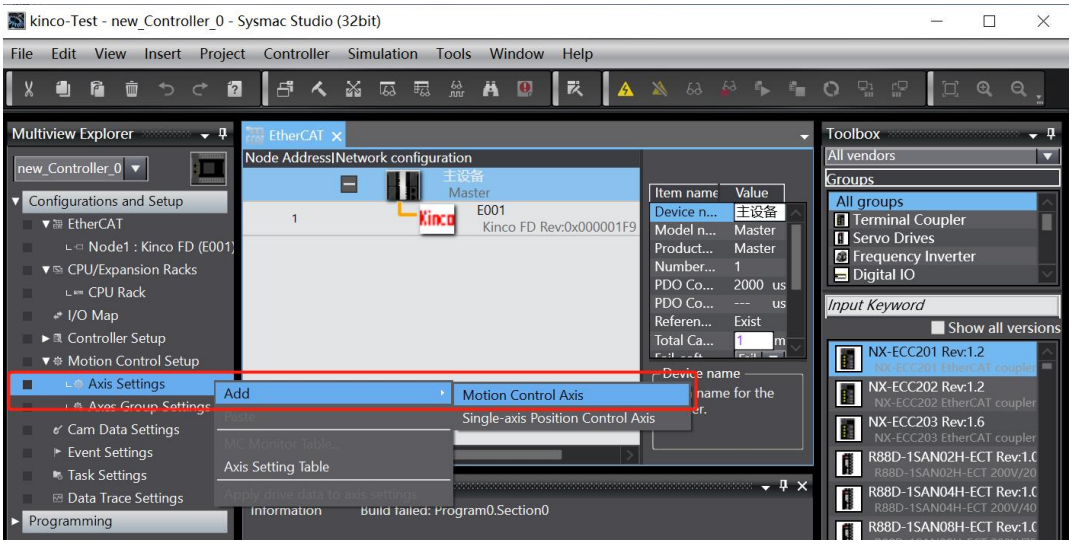
3.3.3 Programming control axis motion

1. Point-to-point control

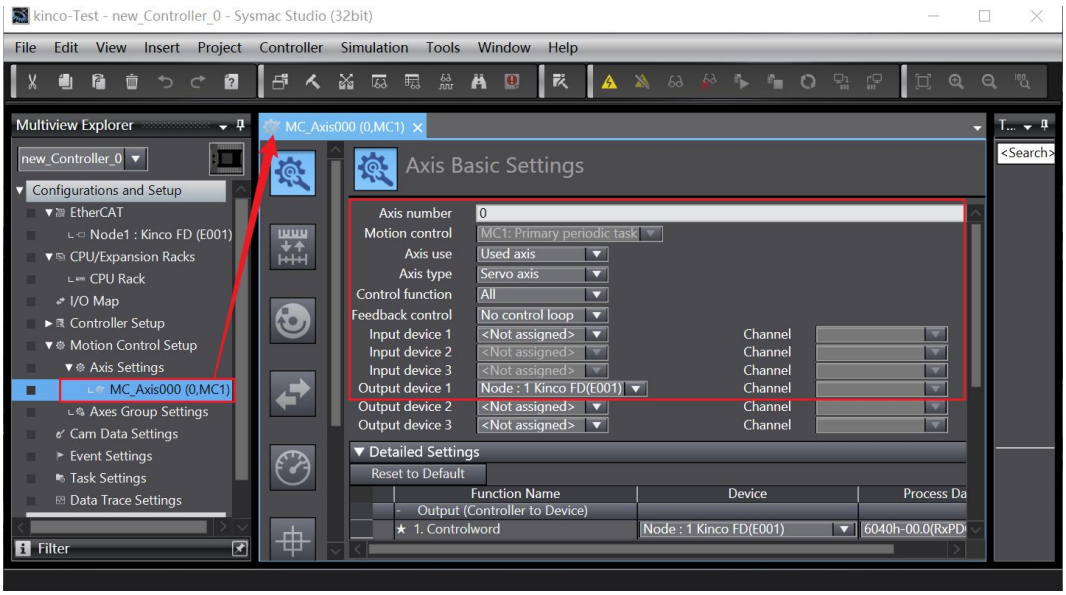
After the communication is established, the variables mapped to the controller by IO in the program can be assigned to each object of the drive, to control the drive motion. For example, a value can be assigned to the target velocity, the control word is assigned to 0xF, and the operation mode is set to 3 to run the velocity mode. See the drive user manual and controller programming information for details.

2. Motion control axis configuration

(1) Add axis. In Configurations and Setup, double-click Motion Control Setup, select Axis Settings, click Add → Motion Control Axis, the project needs to control a few axes to add a few axes (must actually connect these axes, otherwise an error will be reported).



(2) Associate PDO to axis control variables. Double-click the axis you added to configure the axis. With the motion control axis, the point-to-point normal control can no longer be used, and the IO mapping will be automatically removed when the axis is configured.

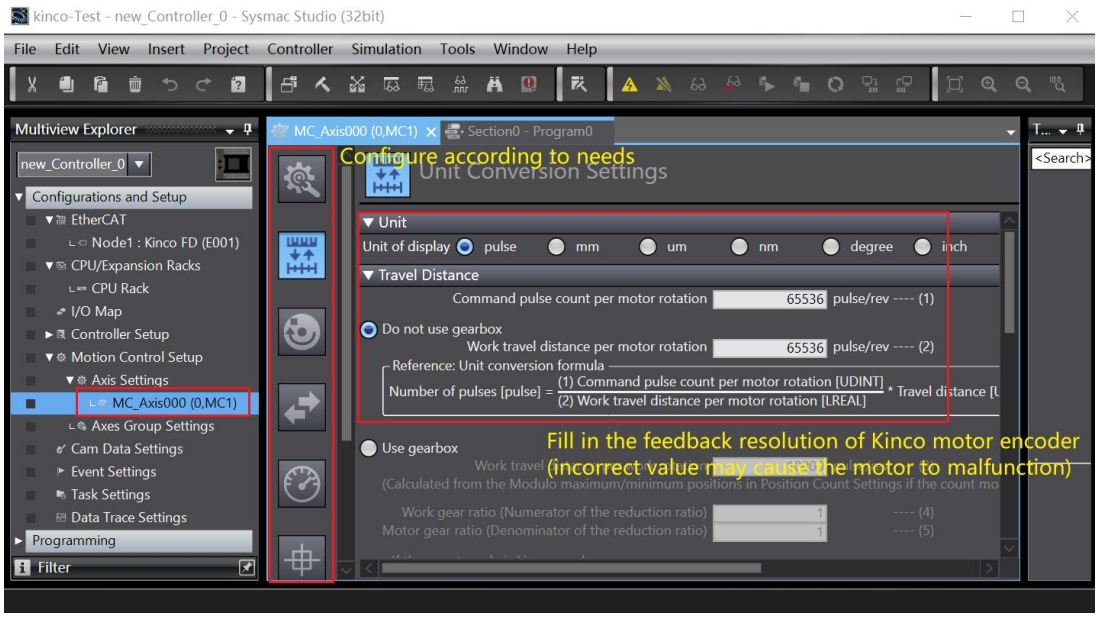


To use the instructions of the MC function module for the axis, the following objects must be set (if the corresponding PDO map is not selected, please refer to "Edit PDO Settings" on page 50 to add it).

Function Name	Device	Process Data
- Output (Controller to Device)		
★ 1. Controlword	Node : 1 Kinco FD(E001)	6040h-00.0(RxPDO_Co
★ 3. Target position	Node : 1 Kinco FD(E001)	607Ah-00.0(RxPDO_Tai
5. Target velocity	<Not assigned>	<Not assigned>
7. Target torque	<Not assigned>	<Not assigned>
9. Max profile Velocity	<Not assigned>	<Not assigned>
11. Modes of operation	Node : 1 Kinco FD(E001)	6060h-00.0(RxPDO_Mc
15. Positive torque limit value	<Not assigned>	<Not assigned>
16. Negative torque limit value	<Not assigned>	<Not assigned>
21. Touch probe function	<Not assigned>	<Not assigned>
44. Software Switch of Encoder's Input	<Not assigned>	<Not assigned>
- Input (Device to Controller)		
★ 22. Statusword	Node : 1 Kinco FD(E001)	6041h-00.0(TXPDO_Sta
★ 23. Position actual value	Node : 1 Kinco FD(E001)	6064h-00.0(TXPDO_Po
24. Velocity actual value	<Not assigned>	<Not assigned>
25. Torque actual value	<Not assigned>	<Not assigned>
27. Modes of operation display	Node : 1 Kinco FD(E001)	6061h-00.0(TXPDO_Mc
40. Touch probe status	<Not assigned>	<Not assigned>
41. Touch probe pos1 pos value	<Not assigned>	<Not assigned>
42. Touch probe pos2 pos value	<Not assigned>	<Not assigned>

For configuration of other input and output PDOs, refer to the controller manual. If PDO is not necessary, it is not recommended to configure. Too much configuration will occupy the EtherCAT bus resources and affect the transmission efficiency.

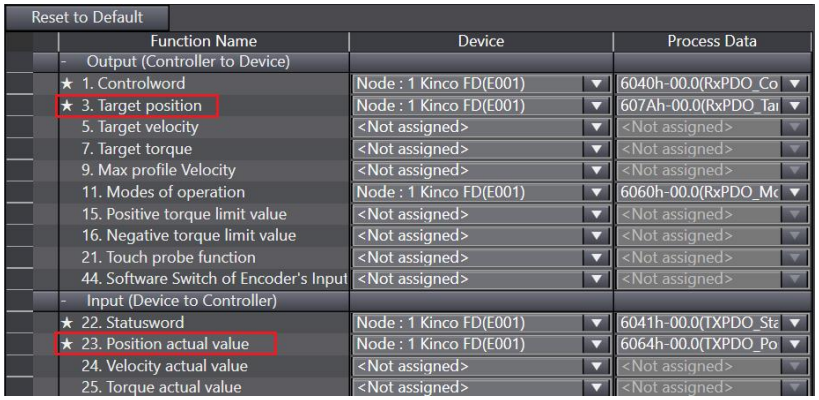
(3) Set axis parameters such as encoder resolution, maximum velocity, home definition, position limit, etc. Please refer to the controller manual for details.



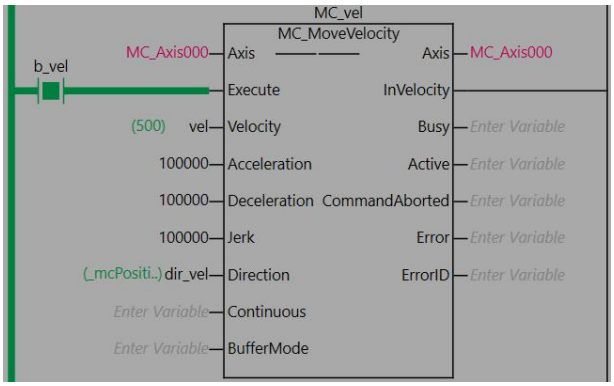
NJ series motion control instructions conform to PLCopen standard. Kinco drive is the standard EtherCAT slave. The following is a brief description of CSP mode, CSV mode, and the MC_Home homing instruction. See the controller manual for more MC instructions.

3. Cyclic Synchronous Position mode (CSP) instruction

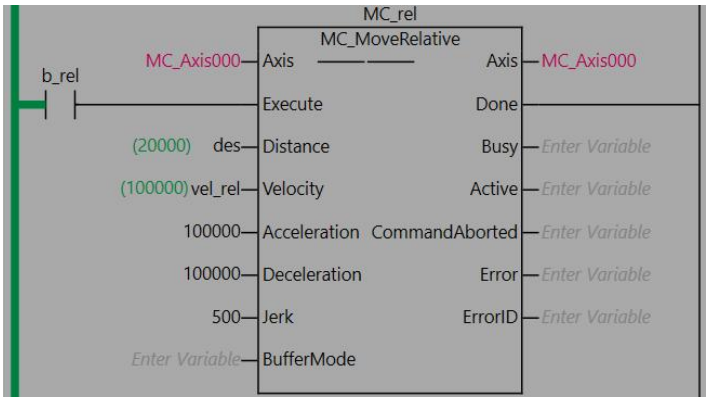
This mode can meet the requirements of most working conditions. For specific application instructions, please refer to the controller manual. The following PDOs need to be configured for this mode.



MC_MoveVelocity instruction:



MC_MoveRelative instruction:

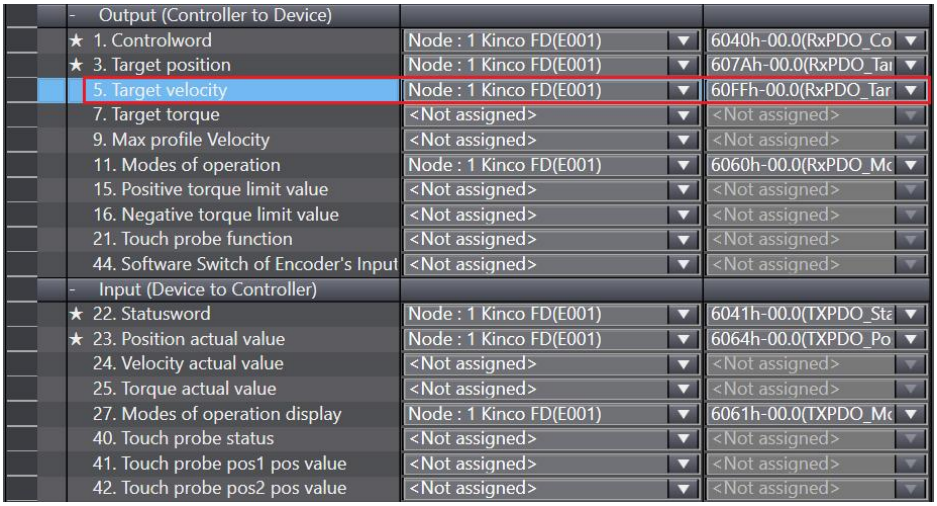


Note

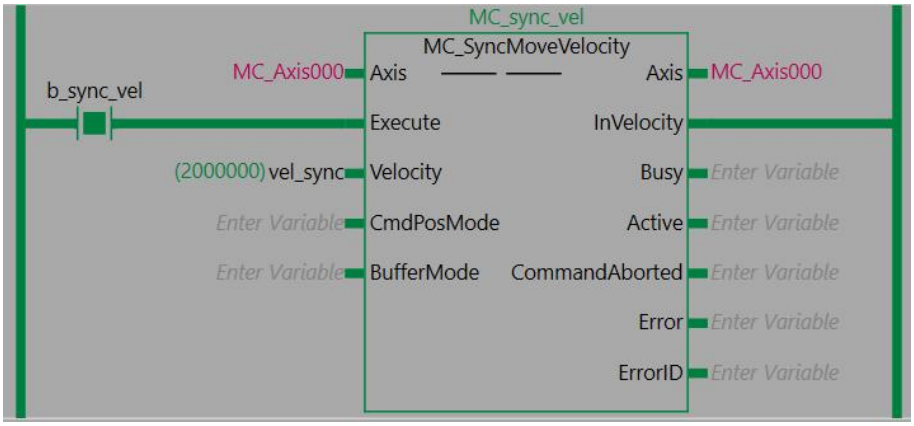
- Calculate the actual velocity (RPM) of the motor according to the feedback resolution of the motor encoder. Because the instruction is CSP mode instruction, the actual velocity of the motor is about 100000/65536 * 60 = 92 RPM. (65536 is the feedback resolution of the kinco motor encoder)

4. Cyclic Synchronous Velocity mode (CSV) instruction

To use this instruction, the target velocity (0x60FF) must be mapped in the controller's axis settings. Here are the PDOs that need to be configured for this mode.



The MC_SyncMoveVelocity instruction block is called in the program to run.

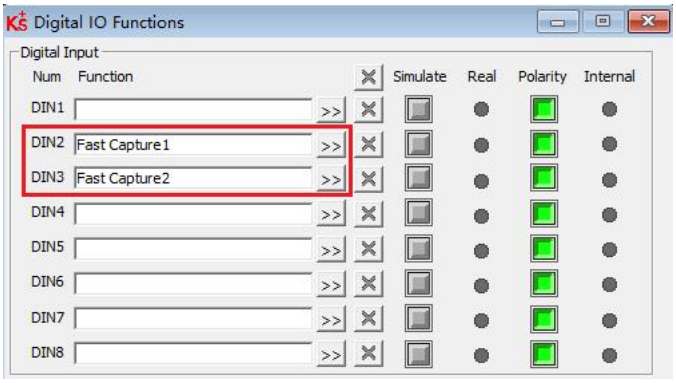


Note

- Calculate the actual velocity (RPM) of the motor according to the feedback resolution of the motor encoder. Since the instruction is in CSV mode, the actual velocity of the motor is about 2000000/65536/16.384 * 60 = 111 RPM. (65536 is the feedback resolution of the kinco motor encoder, 16.384 is the basic unit coefficient of the drive)

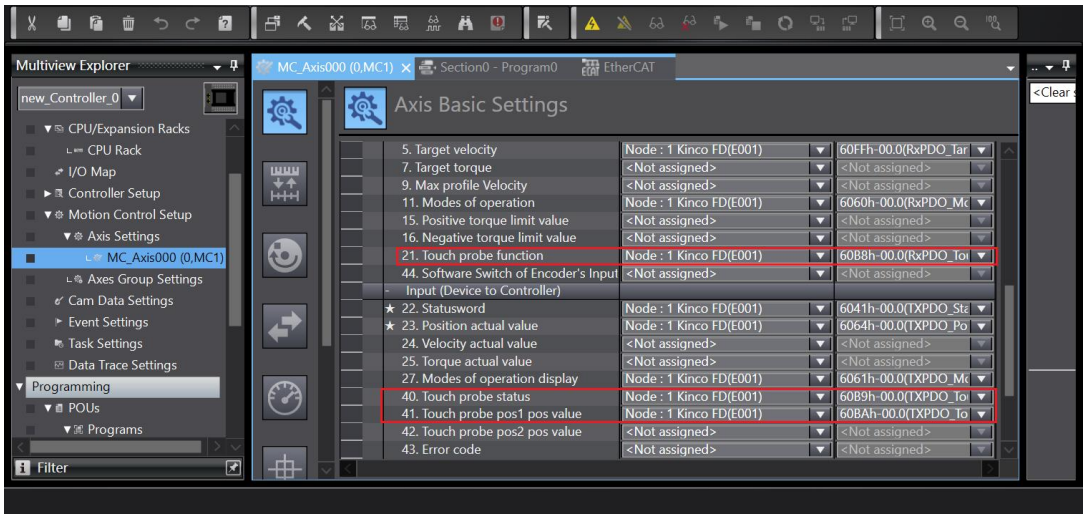
5. Use the probe for homing

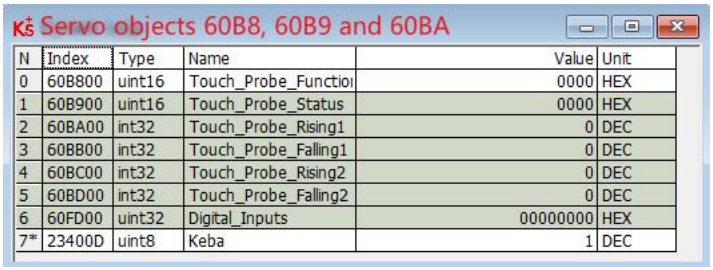
(1) The drive DIN port defines the fast capture function, as shown in the following figure, DIN2 must be defined as Fast Capture 1, DIN3 as Fast Capture 2, so that the input state 0x60FD00 bit17 and bit18 exactly correspond to Fast Capture 1 and Fast Capture 2.



When using the Z phase signal as the trigger signal, because the Z signal is small, the drive starts the Z signal capture function after the fast capture 1 rising edge is triggered. That is, the drive will capture the position where the first Z signal is encountered after the fast capture 1 signal is set, so attention should be paid to reset the fast capture 1 signal when capturing continuously. The reset of the fast capture signal in DIN does not necessarily require the digital input port connection level for signal trigger, but can also be triggered by DIN simulation (0x201002).

(2) Add output and input mapping objects to the axis basic settings. When using the Omron MC_Home home reset instruction, three objects 0x60B8, 0x60B9 and 0x60BA must be mapped. If these settings are not made, the controller will alarm the set process data object insufficient error 0x3461.





N	Index	Type	Name	Value	Unit
0	60B800	uint16	Touch_Probe_Functio	0000	HEX
1	60B900	uint16	Touch_Probe_Status	0000	HEX
2	60BA00	int32	Touch_Probe_Rising1	0	DEC
3	60BB00	int32	Touch_Probe_Falling1	0	DEC
4	60BC00	int32	Touch_Probe_Rising2	0	DEC
5	60BD00	int32	Touch_Probe_Falling2	0	DEC
6	60FD00	uint32	Digital_Inputs	00000000	HEX
7*	23400D	uint8	Keba	1	DEC

Set the capture function (0x60B800) according to the application requirements, observe the capture state (0x60B900) and capture location. The capture function and the capture state are described as follows:

Capture function 0x60FD00 description:

Bit	Value	Description
0	0	Negative position limit signal input is valid
	1	Negative position limit signal input is invalid
1	0	Positive position limit signal input is valid
	1	Positive position limit signal input is invalid
2	0	Home signal input is valid
	1	Home signal input is invalid
16	0	Fast capture 1 signal input is valid
	1	Fast capture 1 signal input is invalid
17	0	Fast capture 2 signal input is valid
	1	Fast capture 2 signal input is invalid

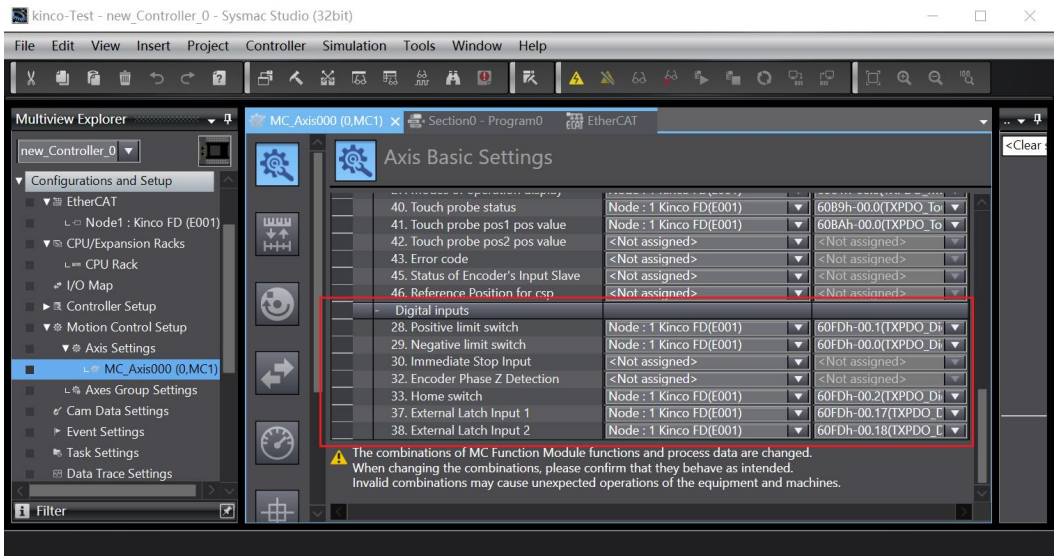
Capture function 0x60B800 description:

Bit	Value	Description
0	0	Fast capture 1 is invalid
	1	Fast capture 1 is valid
1	0	Single capture, only a single capture of the position. If you need to trigger fast capture 1 again to capture the position, you need to reset → set bit 0.
	1	Continuous capture, the position is continuously captured, and the position is captured every time the fast capture signal 1 is triggered, without the reset → set bit0.
2	0	Fast capture 1 triggered by an external signal
	1	Fast capture 1 triggered by Z signal
3	-	None
4	0	Fast capture 1 rising edge is invalid
	1	Fast capture 1 rising edge is valid
5	0	Fast capture 1 falling edge is invalid
	1	Fast capture 1 falling edge is valid
6,7	-	None
8	0	Fast capture 2 is invalid
	1	Fast capture 2 is valid
9	0	Single capture, only a single capture of the position. If you need to trigger fast capture 2 again to capture the position, you need to reset → set bit 0.
	1	Continuous capture, the position is continuously captured, and the position is captured every time the fast capture signal 2 is triggered, without the reset → set bit0.
10	0	Fast capture 2 triggered by an external signal
	1	Fast capture 2 triggered by Z signal
11	-	None
12	0	Fast capture 2 rising edge is invalid
	1	Fast capture 2 rising edge is valid
13	0	Fast capture 2 falling edge is invalid
	1	Fast capture 2 falling edge is valid
14,15	-	None

Capture function 0x60B900 description:

Bit	Value	Description
0	0	Fast capture 1 is invalid
	1	Fast capture 1 is valid
1	0	Fast capture 1 or Z signal rising edge capture is not completed
	1	Fast capture 1 or Z signal rising edge capture is completed
2	0	Fast capture 1 or Z signal falling edge capture is not completed
	1	Fast capture 1 or Z signal falling edge capture is completed
3,4,5,6,7	-	None
8	0	Fast capture 2 is invalid
	1	Fast capture 2 is valid
9	0	Fast capture 2 or Z signal rising edge capture is not completed
	1	Fast capture 2 or Z signal rising edge capture is completed
10	0	Fast capture 2 or Z signal falling edge capture is not completed
	1	Fast capture 2 or Z signal falling edge capture is completed
11,12,13,14,15	-	None

(3) In addition to the above three objects, we also need to add a mapping of the digital input to transmit the relevant signal through 0x60FD. See the figure below to set the digital input.



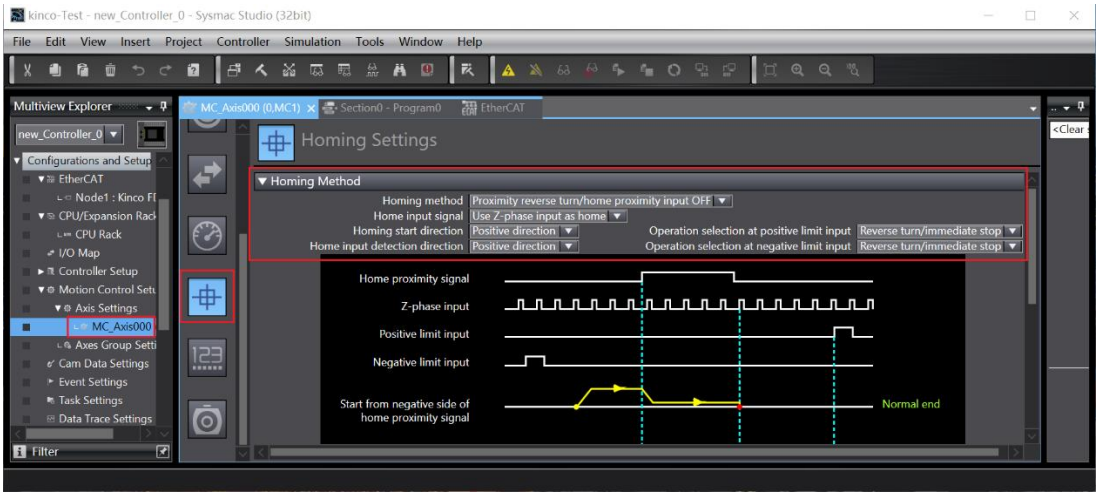
In the above figure, the **Home switch** is the home proximity input signal of the Omron controller, which corresponds to the home signal of the drive.

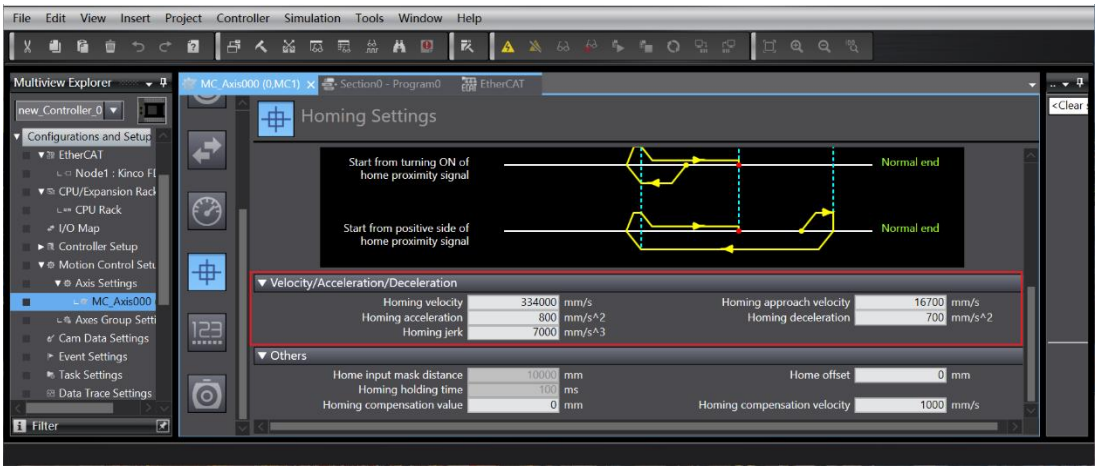
External Latch Input 1 is the home proximity input signal, which corresponds to the fast capture 1 of the servo.

Positive limit switch is a positive position limit signal, corresponding to the positive limit of the drive.

Negative limit switch is the negative position limit signal, corresponding to the negative limit of the drive.

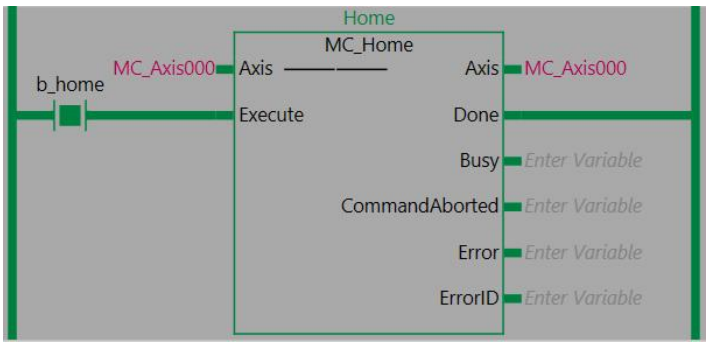
(4) There are 10 homing methods for selection according to the actual application. Through the homing setting interface, set the homing method. Please refer to the controller manual for detailed parameter information.



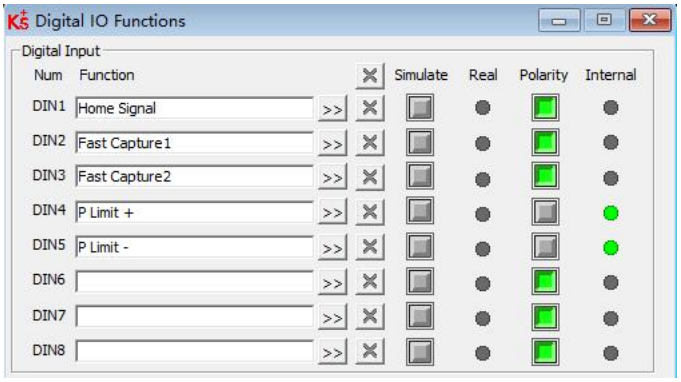


Please refer to the controller manual for the specific setting of the velocity/acceleration/deceleration of the drive to return the home. Since the homing instruction is CSP mode instruction, the return velocity of the motor is about 334000/65536 * 60 = 306 RPM

(5) After enabling the drive via MC_Power, the execution is triggered by giving a rising edge signal to the MC_Home instruction block. The following figure is the status of the instruction to complete homing.



(6) The homing method in this example is to use the home proximity input OFF, that is, the motor runs at the home return velocity after triggering the home instruction, switches to the home return velocity after touching the rising edge of the home signal, and triggers the fast capture 1 after meeting the falling edge of the home signal to return the home. The overall definition of DIN input of the drive is as follows:



(7) Through the Controller → MC Monitor Table in the Sysmac Studio software toolbar, the running status, current operation mode, actual position, and actual velocity of the drive can be monitored. The final homing status of the axis can be monitored in the MC monitor table. If the homing cannot be successfully achieved, please refer to step (3) to check whether the digital input setting is correct.