Digitized Automation for a Changing World

AX-8 Series Operation Manual

www.deltaww.com

AX-8 Series Operation Manual

Revision History

Version	Revision	Date
1st	The first version was published.	2024/03/01
2nd	1. Chapter 2: Modified the execution time of instructions for AX-8 and AX-8H1 in section 2.2.1; modified the information on the LINK indicator for Ethernet port in section 2.2.3; added SSI wiring in section 2.2.5.3. 2. Chapter 9: Added Modbus COM port number setting and range for AX-8 models with the Linux / Windows operating system.	2024/12/30

AX-8 Series Operation Manual

Table of Contents

Appendix A Troubleshooting

Appendix B BIOS Settings

Appendix C Settings and Operation of Windows/Linux

Appendix D Firmware Package Updater Operation

1

Chapter 1 Product Introduction

Table of Contents

1.1 Overview

This manual introduces the AX-8 Series CPU functions, devices, module tables, troubleshooting, and so forth.

1.1.1 Related Manuals

The related manuals for AX-8 Series programmable logic controllers are listed below.

- AX-8 Series Operation Manual

  This manual introduces CPU functions, devices, module tables, electrical specifications, appearances and dimension, basic concept of motion control, basic configurations, troubleshooting, and so forth.

- DIADesigner-AX User Manual

  This manual introduces the use of the software, and programming languages, including Ladder Diagram (LD), Sequential Function Chart (SFC), Structured Text (ST), and Function Block Diagram (FBD), as well as Program Organization Unit (POU), tasks and editing techniques for motion control programs.

- AX Series Motion Controller Manual

  This introduces single-axis and multi-axe instructions for programming the AX Series Motion Controllers.

- AX Series Standard Instructions Manual

  This introduces standard instructions for programming the AX Series Controllers.

1.1.2 Model Descriptions

Classification	Model Name	Description
AX-8 Series Logic Controller	AX-800EP0PB1T AX-800EP0PC1T	NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-800EP0PB1P AX-800EP0PC1P	PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
AX-8 Series Motion Controller	AX-816EP0CB1T AX-816EP0MB1T AX-816EP0CC1T AX-816EP0MC1T	16-axis motion controller CPU module, NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-832EP0CB1T AX-832EP0MB1T AX-832EP0CC1T AX-832EP0MC1T	32-axis motion controller CPU module, NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-864EP0CB1T AX-864EP0MB1T AX-864EP0CC1T AX-864EP0MC1T	64-axis motion controller CPU module, NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-816EP0CB1P AX-816EP0MB1P AX-816EP0CC1P AX-816EP0MC1P	16-axis motion controller CPU module, PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB;

Classification	Model Name	Description
		removable terminal blocks
	AX-832EP0CB1P AX-832EP0MB1P AX-832EP0CC1P AX-832EP0MC1P	32-axis motion controller CPU module, PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-864EP0CB1P AX-864EP0MB1P AX-864EP0CC1P AX-864EP0MC1P	64-axis motion controller CPU module, PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-816EP0CE1T AX-816EP0ME1T	16-axis motion controller CPU module, NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-832EP0CE1T AX-832EP0ME1T	32-axis motion controller CPU module, NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-864EP0CE1T AX-864EP0ME1T	64-axis motion controller CPU module, NPN output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-816EP0CE1P AX-816EP0ME1P	16-axis motion controller CPU module, PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422

Classification	Model Name	Description
		/ 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-832EP0CE1P AX-832EP0ME1P	32-axis motion controller CPU module, PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-864EP0CE1P AX-864EP0ME1P	64-axis motion controller CPU module, PNP output, inbuilt with 1x Relative Encoder, 1x SSI, 8x DI (1MHz), 8x DO (100KHz), 2x Ethernet port, 1x EtherCAT, 1x RS-422 / 485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 512MB; removable terminal blocks
	AX-8H1E30CD2T	NPN output, 8x DI (1KHz), 4x DO (1KHz), 2x Ethernet port, 2x EtherCAT, 1x RS-485, 4x USB, SD interface. Program capacity: 512MB; data capacity: 1024MB; removable terminal blocks

1.2 DIADesigner-AX Software Overview

Conformed to IEC 61131-3, DIADesigner-AX is a new programming tool for a new generation Delta PLC. With the abundant applied instructions and an adequate motion function library, DIADesigner-AX provides a friendly and multilingual programming interface for a more convenient and efficient development environment.

1.2.1 Features

DIADesigner-AX is applicable to AX series.

- Support all the programming languages that IEC 61131-3 defines, including FBD, LD, ST, and SFC, as well as POU, tasks and other programming language standard.
- Powerful and proven function library for various applications.
- Input assistance for the input and configuration.
- User-friendly programming with mouse and keyboard in IEC 61131-3 supported programming languages.
- Extensive debugging and online features for the fast optimization of the application code and to speed up testing and commissioning.
- Numerous security features for the protection of the source code and for safeguarding the operation of the controller.
- Programmable devices from different manufacturers.
- The user interface is extendible and adaptable without leaving the framework.
- Transparent internal structures of the development tool and the available components.
- Many seamlessly integrated tools for different kinds of automation tasks.

Two built-in configuration tools:

- HWCONFIG: the hardware configurations and parameter managements for the system.
- NWCONFIG: the network configurations and data exchange management for the system.

Providing various solutions for motion control including PLCopen, MC function block, G-code editor, E-CAM editor, positioning planning chart tool and many more.

- Support PLCopen POUs for single and multi-axis motions
- Support PLCopen POUs for add-on functions, including diagnostics, stop, and CAM controller
- Additional POUs for different tasks including monitoring dynamic data, following error, operating CAMs and CAM controllers
- Integrated graphical CAM editor with loads of configuration options
- Virtual and logical axes are supported.
- Integrated drivers for numerous Modbus and EtherCAT protocols
- Configuration of the drives as standard field devices.

1.3 Electrical Safety Precautions ⚠

1.3.1 Electrical Safety Precautions (English)

- In order to prevent possible severe damage caused by electric shocks, please first unplug the host power cable from the power outlet before moving the host.
- Confirm that all power cables have been unplugged before connecting or disconnecting any signal cables from the host.
- Confirm that the voltage setting of the power supply is adjusted to the standard voltage value used in this country/region. If you are unsure of the supplied voltage value of your region, please consult your local power company staff.
- If the power supply is damaged, do not attempt to fix it by yourself. Please contact Delta's professional technical service staff or the dealer.
- Restart Instructions: Pressing and holding down the reset button for 2 seconds will force restart.
- It is recommended to install this product inside a cabinet or inside an external case in order to block external collisions.
- This product is applicable to industrial automation equipment and applications. Please read this Operational Manual carefully and perform installation according to the instructions in order to prevent danger from occurring.
- If this product is not operated in accordance with the instructions described in the Manual, it will cause damage to the equipment or abnormal functions.
- The installation that the safety of any system incorporating the equipment is the responsibility of the assembler of the system.
- If the equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.
- Wipe clean with dry cloth to keep the product from humidity.
- Terminal block is mating with Plug and suitable for 16 AWG to 22 AWG. Torque value 0.5N·m Use Copper Conductors Only. The temperature rating of the input connection cable should higher than 105°C.
- To prevent electric shock, make sure to cut off the power to the module and the common input / output points before removing the module or the connected wires.
- This product uses DC power only. Do not use AC power for the product.
- Use the power adapters that comply with the power supply requirements of UL 61010-1, UL 61010-2-201, UL 62368-1, Class 2, SELV, LE, or LPS for this product.
- The power for the PLC and the common input / output points should be two independent power supplies and within the rated voltage. If you are uncertain of the voltage in your area, contact local electrical technicians

1.3.2 Précautions de sécurité électrique (Français)

- Afin déviter déventuels dommages graves causés par des chocs électriques, veuillez dabord débrancher le câble dalimentation de lhôte de la prise de courant avant de déplacer lhôte.
- Vérifiez que tous les câbles dalimentation ont été débranchés avant de connecter ou de déconnecter tout câble de signal de lhôte.
- Vérifiez que le réglage de la tension de lalimentation est ajusté à la valeur de tension standard utilisée dans ce pays / cette région. Si vous nêtes pas sûr de la valeur de tension fournie dans votre région, veuillez consulter le personnel de votre compagnie délectricité locale.
- Si le bloc dalimentation est endommagé, nessayez pas de le réparer vous-même. Veuillez contacter le service technique professionnel de Delta ou le revendeur.
- Instructions de redémarrage: Appuyez et maintenez enfoncé le bouton de réinitialisation pendant 2 secondes pour forcer le redémarrage.
- Il est recommandé dinstaller ce produit à lintérieur dune armoire ou à lintérieur dun boîtier externe afin de bloquer les collisions externes.
- Ce produit est applicable aux équipements et applications dautomatisation industrielle. Veuillez lire attentivement ce manuel dutilisation et effectuer linstallation conformément aux instructions afin déviter tout danger.
- Si ce produit nest pas utilisé conformément aux instructions décrites dans le manuel, il endommagera léquipement ou des fonctions anormales.
- linstallation selon laquelle la sécurité de tout système intégrant léquipement est à la charge de lassembleur du système.
- Si léquip ement est utilisé dune manière non spécifiée par le fabricant, la protection fournie par léquipement peut être altérée.
- Essuyez avec un chiffon sec pour protéger le produit de lhumidité.
- Le bornier se couple avec la fiche et convient pour les fils de calibre 16 à 22 AWG. La valeur de couple est de 0.5 N·m. Utilisez uniquement des conducteurs en cuivre. La température nominale du câble de connexion dentrée doit être supérieure à 105°C.
- Pour éviter les chocs électriques, assurez-vous de couper lalimentation du module et des points dentrée/sortie communs avant de retirer le module ou les fils connectés.
- Ce produit utilise uniquement une alimentation en courant continu. Ne pas utiliser une alimentation en courant alternatif pour ce produit.
- Utilisez les adaptateurs dalimentation conformes aux exigences dalimentation électrique de UL 61010-1, UL 61010-2-201, UL 62368-1, Class 2, SELV, LE, ou LPS pour ce produit.
- Lalimentation du PLC et des points dentrée/sortie communs doit être assurée par deux sources dalimentation indépendantes et dans la plage de tension nominale. Si vous nêtes pas certain de la tension dans votre région, contactez les techniciens en électricité locaux.

2

Chapter 2 Specifications and System Configurations

Table of Contents

2

2.1 General Specifications

Item	Specifications
Operating temperature	0 ~ 50°C*1
Storage temperature	-20 ~ 70°C
Operating humidity	5 ~ 95% No condensation
Storage humidity	5 ~ 95% No condensation
Work environment	No corrosive gas exists.
Installation location	In a control cabinet
Pollution degree	2
Ingress protection (IP ratings)	IP20
EMC Standard (electromagnetic compatibility)	Refer to the tables of EMI, EMS, and conducted immunity test below.
Vibration resistance	Tested with: 5Hz ≦ f ≦ 8.4Hz, constant amplitude 3.5mm; 8.4Hz ≦ f ≦ 150Hz, constant acceleration 1g Duration of oscillation: 10 sweep cycles per axis in each direction of the 3 mutually perpendicular axes International Standard IEC 61131-2 & IEC 60068-2-6 (TEST Fc)
Shock resistance	Tested with: Half-sine wave: Strength of shock 15g peak value, 11 ms duration; Shock direction: The shocks in each direction per axis, on 3 mutually perpendicular axes (total of 18 shocks) International Standard IEC 61131-2 & IEC 60068-2-27 (TEST Fc)
Safety	Conforms to IEC 61131-2, UL 61010-1
Ambient air temperature-barometric pressure-altitude	Operation: 1,080 ~ 795hPa (-1,000 ~ 2,000m) Storage: 1,080 ~ 660hPa (-1,000 ~ 3,500m)
Silicone flammability rating	UL94V-2

*1: Leave the AX-8 Series PLC in an environment within the operating temperature for at least one hour before use to ensure the AX-8 Series PLC temperature is within the operating temperature.

- **EMI**

Port	Frequency range	Level (normative)	Reference standard
Enclosure port (radiated; measured at a distance of 10 meters)	30 ~ 230MHz	40dB (µV/m) quasi-peak	IEC 61000-6-4
	230 ~ 1,000MHz	47dB (µV/m) quasi-peak	
AC power port (conducted)	0.15 ~ 0.5MHz	79dB (µV) quasi-peak	IEC 61000-6-4
		66dB (µV) average	

Port	Frequency range	Level (normative)	Reference standard
	0.5 ~ 30MHz	73dB (μV) quasi-peak	
		60dB (μV) average	

- **EMS**

Environmental phenomenon	Reference standard	Test		Test level
Electrostatic discharge	IEC 61000-4-2	Contact		±4kV
		Air		±8kV
Radiofrequency electromagnetic field - Amplitude modulation	IEC 61000-4-3	80% AM, 1kHz sinusoidal	2.0 ~ 2.7GHz	1V/m
			1.4 ~ 2.0GHz	3V/m
			80 ~ 1,000MHz	10V/m
Power frequency magnetic field	IEC 61000-4-8	60Hz		30A/m
		50Hz		30A/m

- **Conducted immunity test**

Environmental phenomenon		Fast transient burst	High energy surge	Radio frequency interference
Reference standard		**IEC 61000-4-4**	**IEC 61000-4-5**	**IEC 61000-4-6**
Interface / Port	**Specific interface / Port**	**Test level**	**Test level**	**Test level**
Data communication	Shielded cable	1kV	1kV CM	10V
Digital and analog I/O	All shielded lines (to the earth)	1kV	1kV CM	10V
Equipment power	DC power	2kV	0.5kV CM 0.5kV DM	10V
I/O power and auxiliary power output	DC I/O and DC auxiliary power	2kV	0.5kV CM 0.5kV DM	10V

2.2 CPU Module Specifications

2.2.1 Functional Specifications

- **Motion Controller CPU Module**

<table>
<tr><th colspan="4">Type</th><th>AX-8mnEP0MB1y[*1]
AX-8mnEP0CE1y[*1]
AX-8mnEP0ME1y[*1]</th><th>AX-8H1E30CD2T</th></tr>
<tr><td rowspan="2">Process time</td><td rowspan="2">Execution time</td><td colspan="2">LD instruction</td><td>0.4ns</td><td>0.2ns</td></tr>
<tr><td colspan="2">Arithmetic instructions (LREAL data type)</td><td>1.2ns</td><td>0.4ns</td></tr>
<tr><td rowspan="5">Program</td><td colspan="3">Program capacity</td><td>1GB (variable memory included)</td><td>512MB</td></tr>
<tr><td rowspan="3">Variable memory</td><td rowspan="2">Retentive</td><td>Retain</td><td>96KB
(Retain + Persist)</td><td>1.5MB</td></tr>
<tr><td>Persist</td><td>96KB
(Retain + Persist)</td><td>2.0MB</td></tr>
<tr><td colspan="2">Non-retentive</td><td>1GB (program capacity included)</td><td>1024MB</td></tr>
<tr><td>Device memory (%M)</td><td colspan="2">Size</td><td>5MB</td><td>0.5MB (retentive)</td></tr>
<tr><td rowspan="7">Motion Control</td><td rowspan="5">Number of controlled axes</td><td colspan="2">Maximum number of controlled axes</td><td>mn=16: 16 axes
mn=32: 32 axes
mn=64: 64 axes</td><td>128 axes</td></tr>
<tr><td colspan="2">EtherCAT axes[*2]</td><td>mn=16: 16 axes
mn=32: 32 axes
mn=64: 64 axes</td><td>128 axes</td></tr>
<tr><td colspan="2">Pulse Out axes</td><td colspan="2">0</td></tr>
<tr><td colspan="2">Maximum number of axes for linear interpolation axis control</td><td colspan="2">6 axes</td></tr>
<tr><td colspan="2">Maximum number of axes for circular interpolation control</td><td colspan="2">6 axes (3 slave axes)</td></tr>
<tr><td colspan="3">Maximum number of axis groups</td><td colspan="2">8 groups</td></tr>
<tr><td colspan="3">Motion control period</td><td colspan="2">The same control period as that is used for the communication cycle of data processing for EtherCAT.</td></tr>
</table>

Type				AX-8mnEP0MB1y[*1] AX-8mnEP0CE1y[*1] AX-8mnEP0ME1y[*1]	AX-8H1E30CD2T
	CAM	Number of CAM data points	Maximum points per CAM table	32,767 points	
			Maximum points for all CAM tables	655,340 points	
		Maximum number of CAM tables		160	
Ethernet port	Number of ports			2	
	Physical media types			IEEE 802.3 / 802.3u / 802.3ab 1Gbps	
	Topology			Star and line	
	Transmission speed			10 / 100 / 1,000Mbps	
	Cable			Category 5e or later, 100 meters (max.)	
	Communication protocols			ARP, IP, TCP, UDP, Modbus TCP, EtherNet/IP, OPC UA	
USB port	Number of ports			4	
	Type			USB 2.0 (0.5A)	USB 3.0 (1.0A / 2 ports shared)
Isolated RS-485 port	Number of ports			1	
	Baud rate			9,600 / 19,200 / 38,400 / 57,600 / 115,200bps	
	Serial communication format			■ Stop bit: 1, 2; ■ Parity bit: None, Odd, Even; ■ Data bit: 7, 8	
	Communication protocols			■ Master (Modbus ASCII/RTU) ■ Slave (Modbus RTU)	Modbus ASCII/RTU
Isolated RS-422 port	Number of ports			1	0
	Baud rate			9,600 / 19,200 / 38,400 / 57,600 / 76,800 / 115,200bps	N/A
	Serial communication format			■ Stop bit: 1, 2; ■ Parity bit: None, Odd, Even; ■ Data bit: 7, 8	N/A
	Communication protocols			■ Master (Modbus ASCII/RTU) ■ Slave (Modbus RTU)	N/A

Type			AX-8mnEP0MB1y[*1] AX-8mnEP0CE1y[*1] AX-8mnEP0ME1y[*1]	AX-8H1E30CD2T
EtherCAT port	Number of ports		1	2
	EtherCAT Master		Class B	
	Physical media types		IEEE 802.3 / 802.3u / 802.3ab 1Gbps	
	Transmission speed		100Mbps	
	Topology		Line, daisy chain, and branching	
	Cable		Category 5e or later, 100 meters (max.)	
	Maximum number of Slaves		256	512
	Communication cycle		250μs ~ (unit can be set to 250μs)	
TCP	Modbus TCP	Maximum number of connections	32 (Server + Client)	
	Socket	Maximum number of TCP connections		
	Modbus TCP	Maximum data length per connection	100 words	
	Socket	Maximum data length per instruction	8KB	
EtherNet/IP	CIP I/O Connection	Maximum number of connections (Scanner)	32+32	
		Maximum number of connections (Adapter)	32+32	
		Requested Packet Interval (RPI)	■ AX-8mnEP0MB1y: 10ms ~ 100ms ■ AX-8mnEP0CE1y, AX-8mnEP0ME1y: 1ms ~ 100ms	1ms□100ms
		Maximum Transmission Speed	■ AX-8mnEP0MB1y: 12,800pps ■ AX-8mnEP0CE1y, AX-8mnEP0ME1y: 32,000pps	64,000pps
		Maximum data length per connection	Default: 100 bytes Input: 509 bytes (T→O) Output: 505 bytes (O→T) (O: Originator; T: Target)	

<table>
<tr><th colspan="3">Type</th><th>AX-8mnEP0MB1y[*1]
AX-8mnEP0CE1y[*1]
AX-8mnEP0ME1y[*1]</th><th>AX-8H1E30CD2T</th></tr>
<tr><td rowspan="2"></td><td rowspan="2">CIP Explicit Message</td><td>Class 3 / UCMM</td><td colspan="2">■ Get_Attribute_Single (FB)
■ Get_Attributes_All (FB)
■ Set_Attribute_Single (FB)
■ Set_Attributes_All (FB)</td></tr>
<tr><td>CIP objects supported</td><td colspan="2">Identity, Message Router, Assembly, Connection Manager, Port, TCP/IP Interface, Ethernet Link, Vendor Specific</td></tr>
<tr><td rowspan="14">OPC UA server</td><td colspan="2">Supported profiles and models</td><td colspan="2">PLCopen and OPC Foundation:
OPC UA Information Model for IEC 61131-3</td></tr>
<tr><td colspan="2">Endpoints and connecting ports</td><td colspan="2">TCP: 4840 (Reconfigurable via configuration file)</td></tr>
<tr><td colspan="2">Maximum number of sessions (Client)</td><td colspan="2">5</td></tr>
<tr><td colspan="2">Maximum number of monitored items</td><td colspan="2">1,000</td></tr>
<tr><td colspan="2">Sampling rate of the monitored items (ms)</td><td colspan="2">100 / 300 / 500 / 1,000 / 2,500 / 5,000</td></tr>
<tr><td colspan="2">Maximum number of subscriptions</td><td colspan="2">100</td></tr>
<tr><td colspan="2">Maximum number of variables that can be published</td><td colspan="2">10,000</td></tr>
<tr><td colspan="2">Maximum number of value attributes that can be published</td><td colspan="2">10,000</td></tr>
<tr><td colspan="2">Maximum number of structure definitions
that can be published</td><td colspan="2">100</td></tr>
<tr><td colspan="2">Conditions that variable can not be published</td><td colspan="2">■ More than three-dimensional arrays
■ Array of Array
■ The OPC UA Stack will limit messages to about 300kB. This is the maximum for values, too.
■ Pointer variables, Interface variables
■ Structures containing pointers and interfaces</td></tr>
<tr><td colspan="2">Security policy/mode</td><td colspan="2">None
Sign - Basic256Sha256
SignAndEncrypt - Basic256Sha2566</td></tr>
<tr><td rowspan="2">Application authentication</td><td>Authentication</td><td colspan="2">X.509</td></tr>
<tr><td>Number of storable certificates</td><td colspan="2">■ Trusted certificates: 32
■ Issuer certificates: 32
■ Rejected certificates: 32</td></tr>
<tr><td>User authentication</td><td>Authentication</td><td colspan="2">“Anonymous” or with user name and password</td></tr>
</table>

Type			AX-8mnEP0MB1y*1 AX-8mnEP0CE1y*1 AX-8mnEP0ME1y*1	AX-8H1E30CD2T
I/O configuration	Number of I/O extension modules supported		N/A	
	I/O data capacity		IN: 131,072 bytes OUT: 131,072 bytes	
	Inbuilt I/O	Encoder	1-CH EA± / EB± / EZ± 1MHz	N/A
		SSI	1 set (shared with RS-422)	
		High-speed counter	1	
		Pulse out	0	
Memory card	SD card type		1x SD Card Slot (SD Card 3.0 Interface)	
Real-time clock	Year, Month, Date, Hour, Minute, Second, Week		Inbuilt RTC	

*1: **mn** can be 16, 32, or 64, representing the number of axes for the controller.

*2: EtherCAT axes include positioning and synchronous axes. The maximum number of axes is listed below.

Item / Model	Maximum number of positioning axes	Maximum number of synchronous axes	Maximum number of positioning and synchronous axes
AX-816EP0*1	16	16	16
AX-832EP0*2	32	32	32
AX-864EP0*3	64	64	64
AX-8H1E30CD2T	128	128	128

*1: AX-816EP0 includes AX-816EP0□□□□ series PLC.
*2: AX-832EP0 includes AX-832EP0□□□□ series PLC.
*3: AX-864EP0 includes AX-864EP0□□□□ series PLC.

2.2.2 Electrical Specifications

Item \ Model		AX-8□□EP0□□□T	AX-8□□EP0□□□P	AX-8H1E30CD2T
Supply voltage		24VDC (-15% ~ +20%)		
Power consumption (W)		28.8W (24V/1.2A)		72W (24V/3A)
Weight (g)	Net weight*1	750g		1000g
	Gross weight*2	1100g		1400g

*1: Including terminal blocks.

*2: Including terminal blocks and all packaging.

- **Electrical specifications for the inputs on the digital input/output module**

Item \ Model		AX-8□□EP0□□□T	AX-8□□EP0□□□P	AX-8H1E30CD2T
Number of inputs		8		
Connector type		Removable terminal blocks		
Input type		Digital input		
Input form		NPN (Sink)	PNP (Source)	NPN (Sink)
Input voltage/current		24VDC, 5 mA/CH (-15% ~ +20%)		
Action level	OFF→ON	> 15VDC		
	ON→OFF	< 5VDC		
Response time	OFF→ON	200ns		450μs
	ON→OFF	150ns		350μs
Maximum input frequency		200KHz		1KHz
Isolation method		Optocoupler isolation		

- **Electrical specifications for the outputs on the digital input/output module**

Item \ Model		AX-8□□EP0□□□T	AX-8□□EP0□□□P	AX-8H1E30CD2T
Number of outputs		8		4
Connector type		Removable terminal blocks		
Output type		Digital output		
Output form		NPN (Sink)	PNP (Source)	NPN (Sink)
Voltage		24VDC		
Maximum output current		100mA/CH	50mA/CH	100mA/CH
Response time	OFF→ON	Ton ≤ 1μs @ 24V/100mA	Ton ≤ 1.3μs @ 24V/50mA	Ton ≤ 60μs @ 24V/100mA
	ON→OFF	Toff ≤ 1μs @ 24V/100mA	Toff ≤ 2.6μs @ 24V/50mA	Toff ≤ 130μs @ 24V/100mA
Isolation method		Optocoupler isolation		

2

- **Electrical specifications for the inputs of the encoder**

Item \ Model		AX-8□□EP0□□□T	AX-8□□EP0□□□P
Input signal		Differential	
Endpoints		1-CH EA± / EB± / EZ±	
Response time	OFF→ON	150ns	
	ON→OFF	150ns	
Isolation method		Optocoupler isolation	

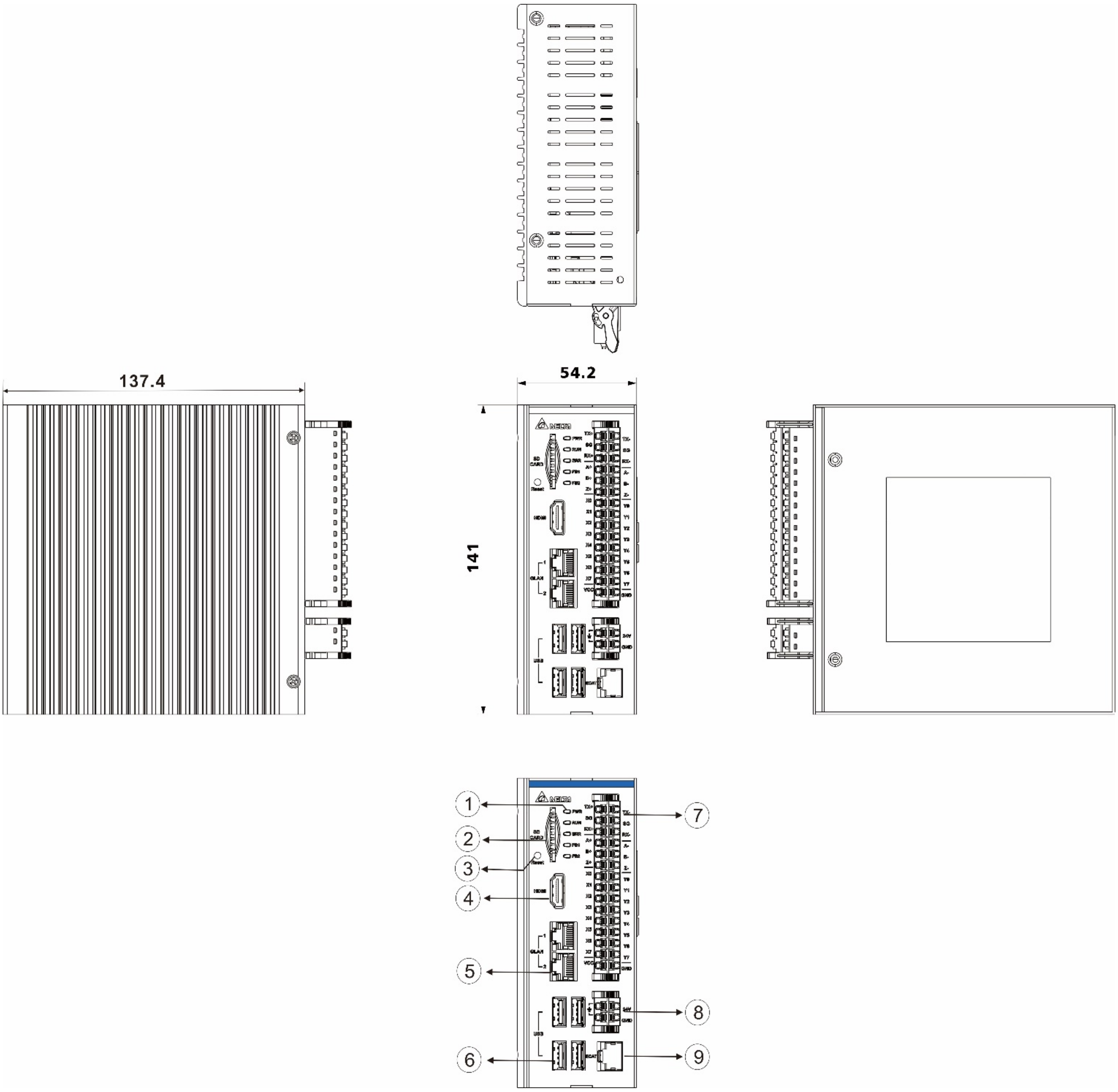
- **Electrical specifications for other parts**

Item \ Model		AX-8□□EP0□□□T	AX-8□□EP0□□□P	AX-8H1E30CD2T
Main system	CPU	Intel Celeron J1900 Quad Core 2.00GHz		Intel I3 TigerLaker UP3 Dual Core 3.00GHz
	BIOS	AMI BIOS		
	Memory	DDR3L 4GB	DDR3L 2GB	LPDDR4X 4GB
	Retentive memory	96KB MRAM		4MB
Communication interface	Network interface	2x IEEE 802.3 / 802.3u / 802.3ab 1Gbps		2x IEEE 802.3 / 802.3u / 802.3ab 1Gbps
	Fieldbus interface	1x EtherCAT		2x EtherCAT
	USB	4x USB 2.0		4x USB 3.0
	Serial port	1x Isolated RS-485 / 422		1x Isolated RS-485
Display interface		1x HDMI 1.4a		1x HDMI 2.0
Extension interface		1x SD Card Slot (SD Card 3.0 Interface)		1x SD Card Slot (SD Card 3.0 Interface)
Storage device*1	SSD	1x M.2 2242 type B&M-key SATA SSD (SATA 2.0 Interface)	N/A	1x M.2 SATA 32GB
	eMMC	N/A	1x eMMC 8GB	N/A

*1: You should write file to an SD card or USB rather than to SSD or eMMC when necessary.

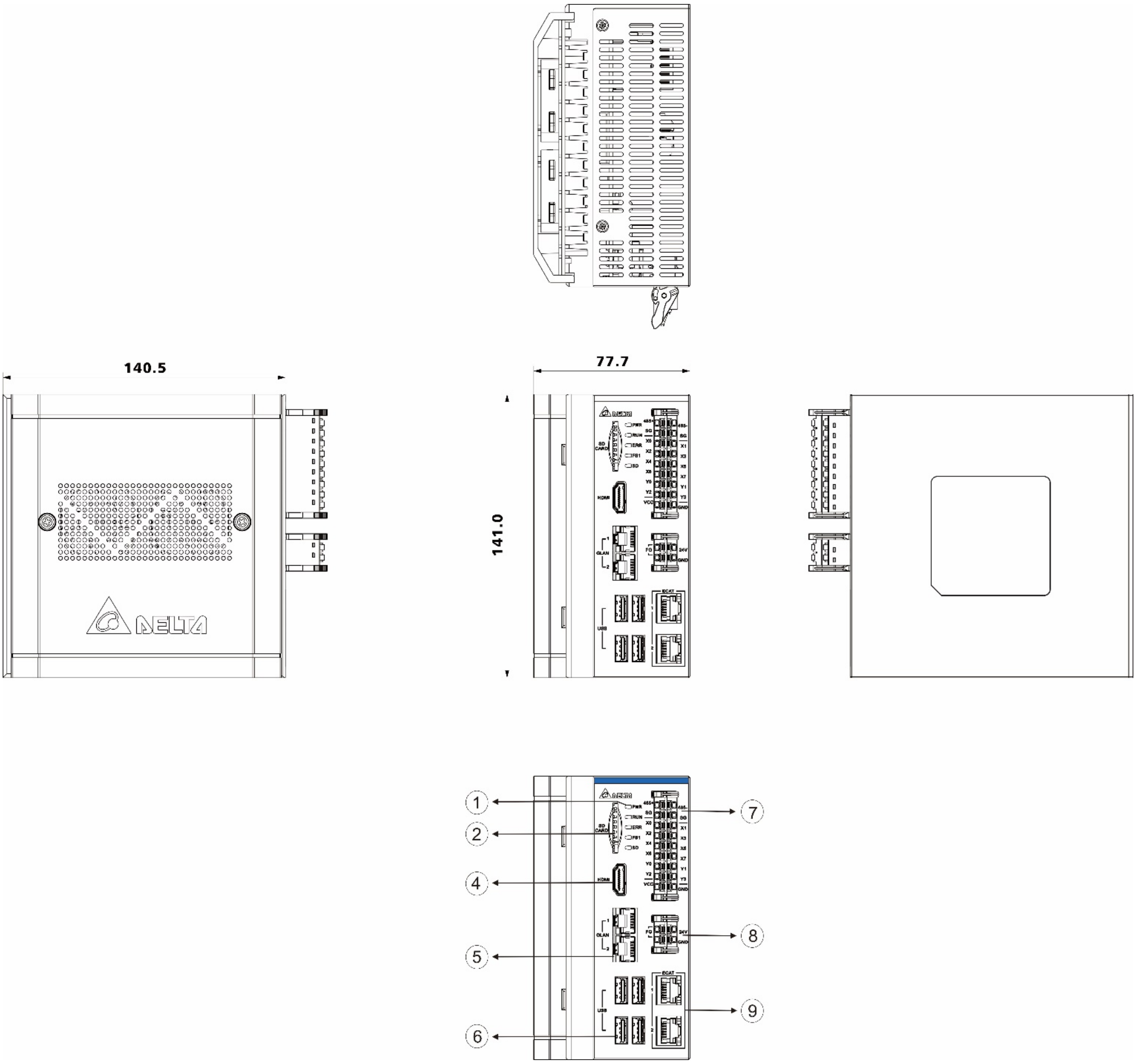
2.2.3 CPU Module Profiles

- **AX-8□□EP0**



Unit: mm

- **AX-8H1E30CD2T**



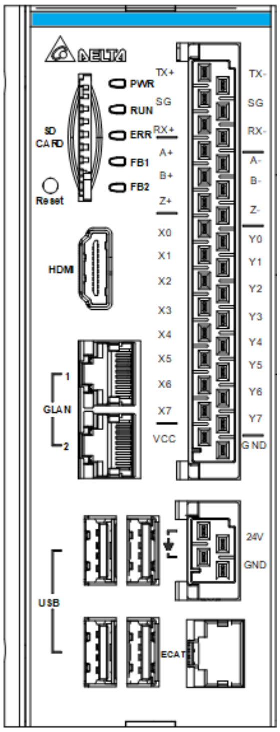
單位：mm

<table>
<tr><th rowspan="2">Number</th><th rowspan="2">Name</th><th colspan="2">Description</th></tr>
<tr><th>AX-8□□EP0</th><th>AX-8H1E30CD2T</th></tr>
<tr><td rowspan="4">1</td><td>POWER LED</td><td colspan="2">Indicates the power status of the CPU module (Orange)
■ ON: power on.
■ OFF: power off.</td></tr>
<tr><td>RUN LED</td><td colspan="2">Indicates the operation status of the CPU module (Green)
■ ON: the module is running.
■ OFF: the module is stopped.</td></tr>
<tr><td>ERROR LED</td><td colspan="2">Indicates the error status of the CPU module (Red)
■ ON: a serious error occurs in the module.
■ OFF: the module is normal.
■ Blinking: a minor error occurs in the module.</td></tr>
<tr><td>FB1 LED
FB2 LED*1
/
FB1 LED
SD LED*1</td><td>Indicates FB communication status
■ FB1 OFF: NO communication over EtherCAT
■ FB1 ON: normal communication over EtherCAT
■ FB1 Blinking: abnormal communication over EtherCAT
■ FB2 OFF: NO communication over Modbus
■ FB2 ON: communication over Modbus</td><td>Indicates FB communication status
■ FB1 OFF: NO communication over EtherCAT
■ FB1 ON: normal communication over EtherCAT
■ FB1 Blinking: abnormal communication over EtherCAT
■ SD LED OFF: SD card detached
■ SD LED ON: SD card attached</td></tr>
<tr><td>2</td><td>SD Card Slot</td><td colspan="2">Provides an interface for an SD card</td></tr>
<tr><td>3</td><td>Reset</td><td>Reset button</td><td>N/A</td></tr>
<tr><td>4</td><td>HDMI</td><td colspan="2">For display output</td></tr>
<tr><td>5</td><td>Ethernet Port</td><td colspan="2">Ethernet Switch communication port
LINK indicator:
■ ON: the network connection is established. (Orange: 1000 Mbps; Green: 100 Mbps)
■ OFF: the network connection is NOT established.
ACT indicator (Orange):
■ Blinking: data transmission (sending/receiving)
■ OFF: NO data transmission</td></tr>
<tr><td>6</td><td>USB Port</td><td>USB 2.0 port</td><td>USB 3.0 port</td></tr>
<tr><td rowspan="3">7</td><td>COM Port</td><td>Provides an interface for RS-422 and RS-485 communication</td><td>Provides an interface for RS-485 communication</td></tr>
<tr><td>Input terminals</td><td colspan="2">For input wiring</td></tr>
<tr><td>Output terminals</td><td colspan="2">For output wiring</td></tr>
<tr><td>8</td><td>Power supply</td><td colspan="2">For power supply</td></tr>
<tr><td>9</td><td>EtherCAT Port</td><td colspan="2">EtherCAT communication port
LINK indicator (Green):
■ ON: the network connection is established.
■ OFF: the network connection is NOT established.
ACT indicator (Orange):
■ Blinking: data transmission (sending/receiving)
■ OFF: NO data transmission</td></tr>
</table>

*1: FB2 LED is exclusive to AX-8□□EP0 series while SD LED is exclusive to AX-8H1E30CD2T series.

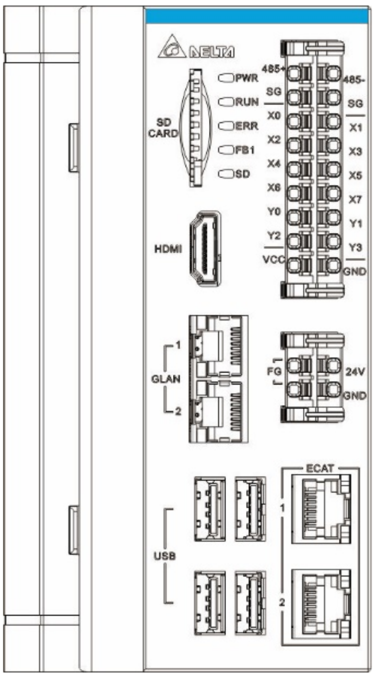
2.2.4 CPU Module Input/Output Terminals

AX-8□□EP0



GPIO	
TX+	TX-
SG	SG
RX+	RX-
A+	A-
B+	B-
Z+	Z-
X0	Y0
X1	Y1
X2	Y2
X3	Y3
X4	Y4
X5	Y5
X6	Y6
X7	Y7
VCC	GND

AX-8H1E30CD2T



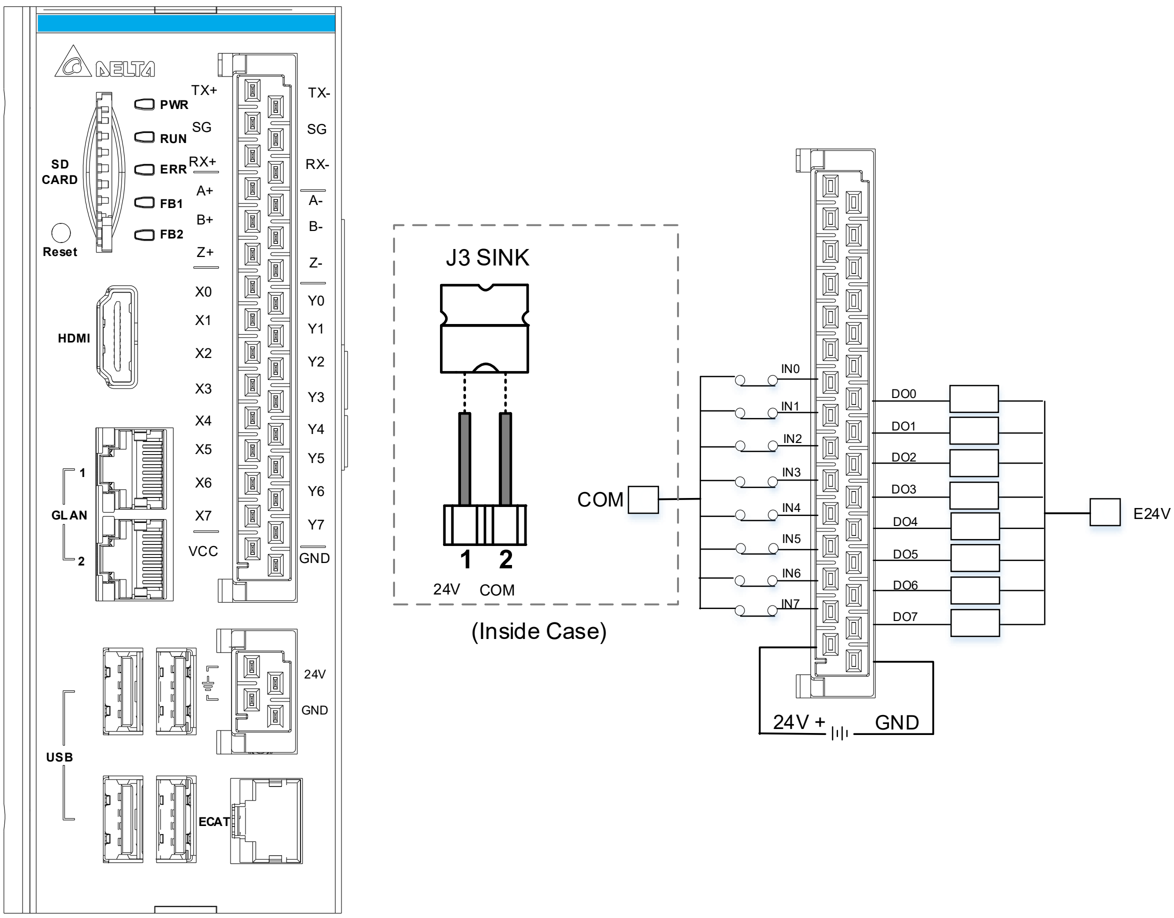
GPIO	
485+	485-
SG	SG
X0	X1
X2	X3
X4	X5
X6	X7
Y0	Y1
Y2	Y3
VCC	GND

2.2.5 Wiring Configuration

2.2.5.1 AX-8□□EP0□□□T Wiring

Input type	NPN (Sink)
Input voltage/current	24VDC, 5mA
Output type	NPN (Sink)
Voltage/current	24VDC, 100mA

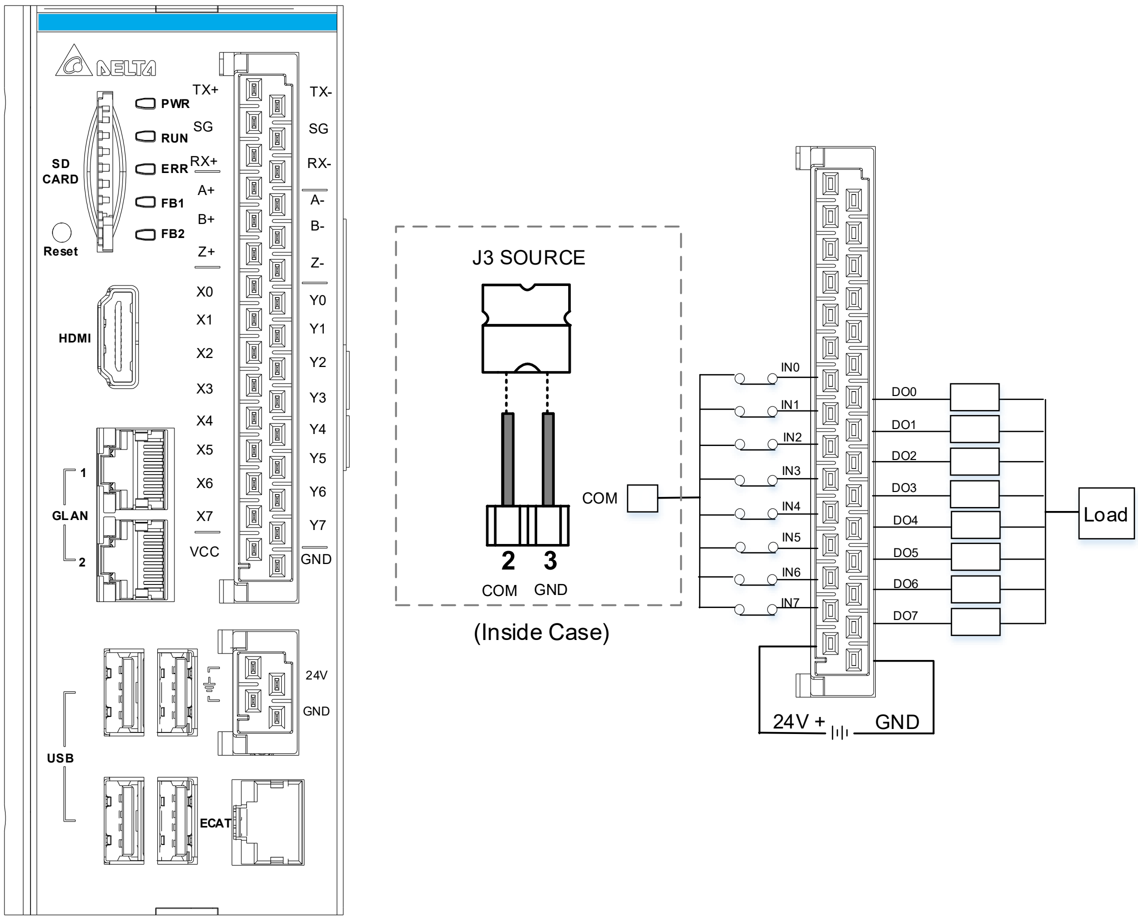
- **General Input / Output Wiring**



2.2.5.2 AX-8□□EP0□□□P Wiring

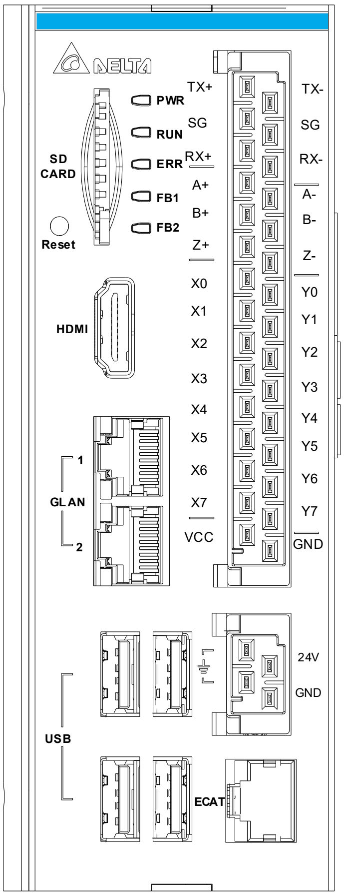
Input type	PNP (Source)
Input voltage/current	24VDC, 5mA
Output type	PNP (Source)
Voltage/current	24VDC, 50mA

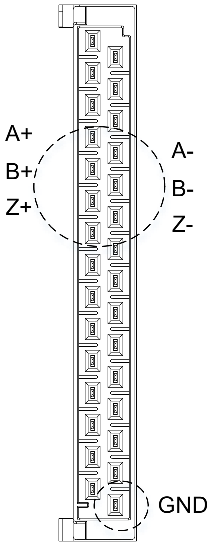
- **General Input / Output Wiring**



2.2.5.3 AX-8□□EP0□□□T, AX-8□□EP0□□□P Wiring

- **Encoder Wiring (5V supported only)**

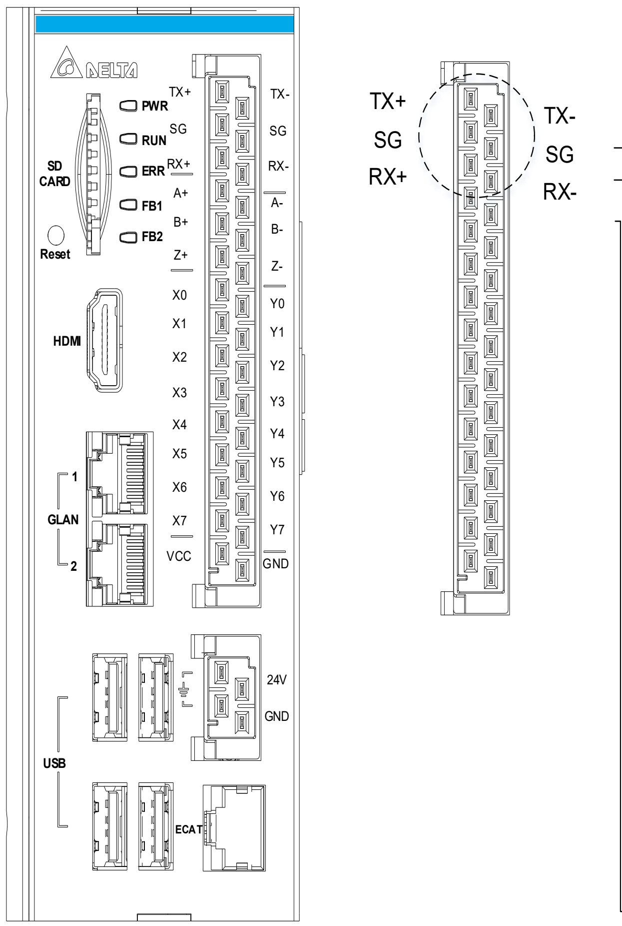




Encoder Pin	Pulse Out Pin
A+	A
A-	$\overline{A}$
B+	B
B-	$\overline{B}$
Z+	Z
Z-	$\overline{Z}$
GND	0V
	VCC

2

- **RS-422 / RS-485 / SSI Wiring**



RS422

Pin	Function	Pin
TX+	RS422 TX+	RS422 RX+
TX-	RS422 TX-	RS422 RX-
SG	RS422/485 GND	SG
RX+	RS422 RX+	RS422 TX+
RX-	RS422 RX-	RS422 TX-

RS485

Pin	Function	Pin
TX+	RS485 +	RS485 +
TX-	RS485 -	RS485 -
SG	RS422/485 GND	SG

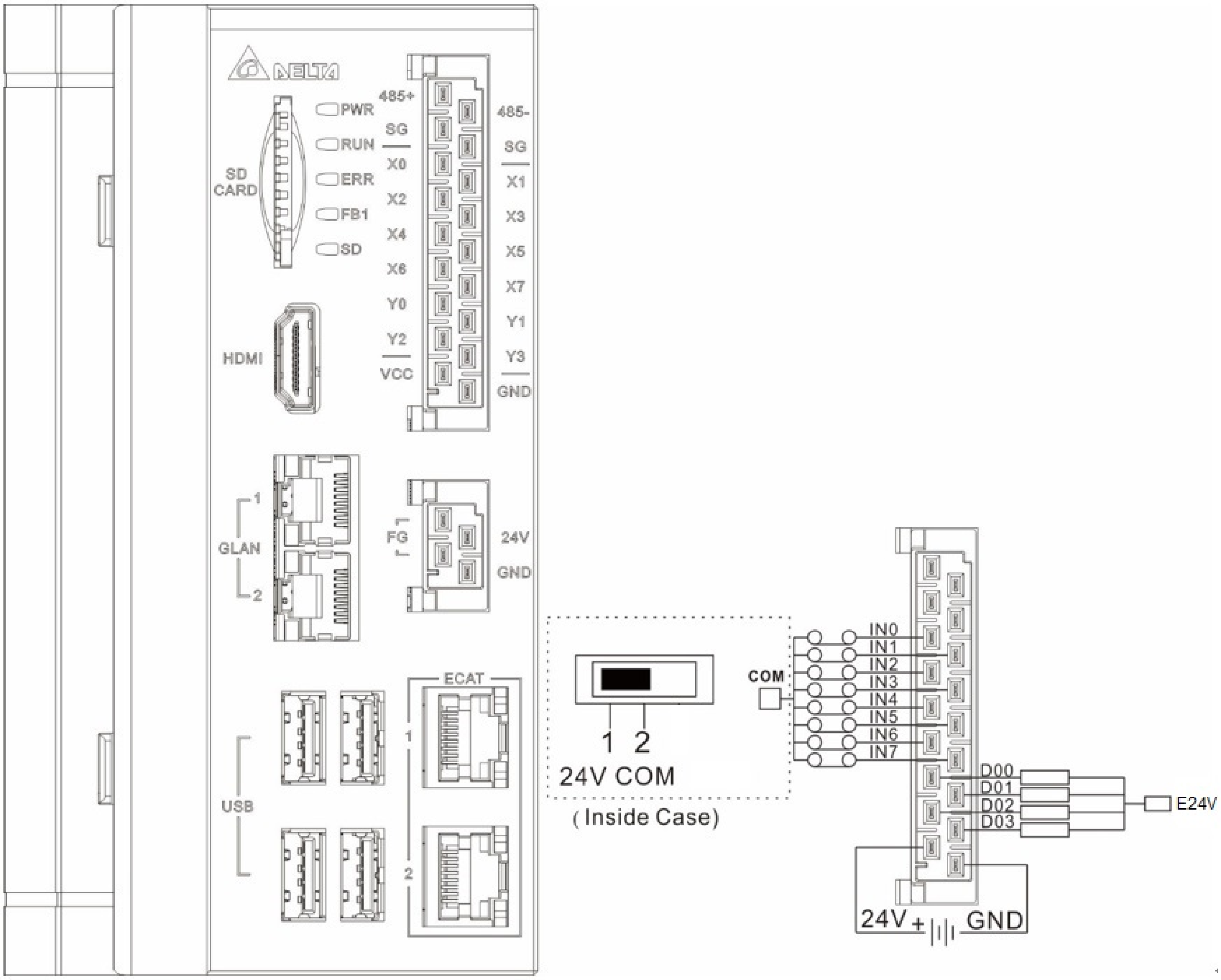
SSI

Pin	Function	Pin
TX+	CLOCK+	SSI CLOCK+
TX-	CLOCK -	SSI CLOCK-
SG	SSI GND	SG
RX+	DATA+	SSI DATA+
RX-	DATA-	SSI DATA-

2.2.5.4 AX-8H1E30CD2T Wiring

Input type	NPN (Sink)
Input voltage/current	24VDC, 5mA
Output type	NPN (Sink)
Voltage/current	24VDC, 100mA

- **General Input / Output Wiring**



MEMO

3

Chapter 3 Installing Hardware and Getting Started

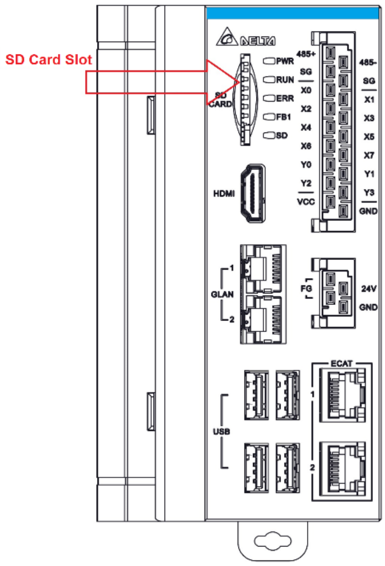
Table of Contents

3.1 Installing Hardware

3.1.1 Installing and Uninstalling an SD Card

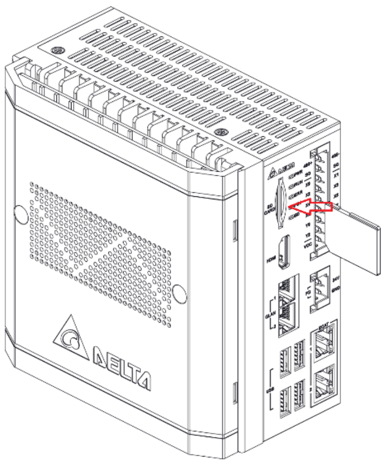
- **SD card slot of the CPU module**

The SD card slot is on the front side of the AX-8 Series PLC as indicated by the arrow below.



- **Installing an SD card**

Insert an SD card vertically into the card slot and push it to the end of the slot until it clicks. Be sure the SD card is fixed firmly in the slot; if the SD card is loose, it is not installed correctly. With a foolproof design, the SD card can only be inserted in one direction. Do not force to push the SD card into the slot in the wrong direction or you may damage the controller/card. Please refer to the figure below for instruction.

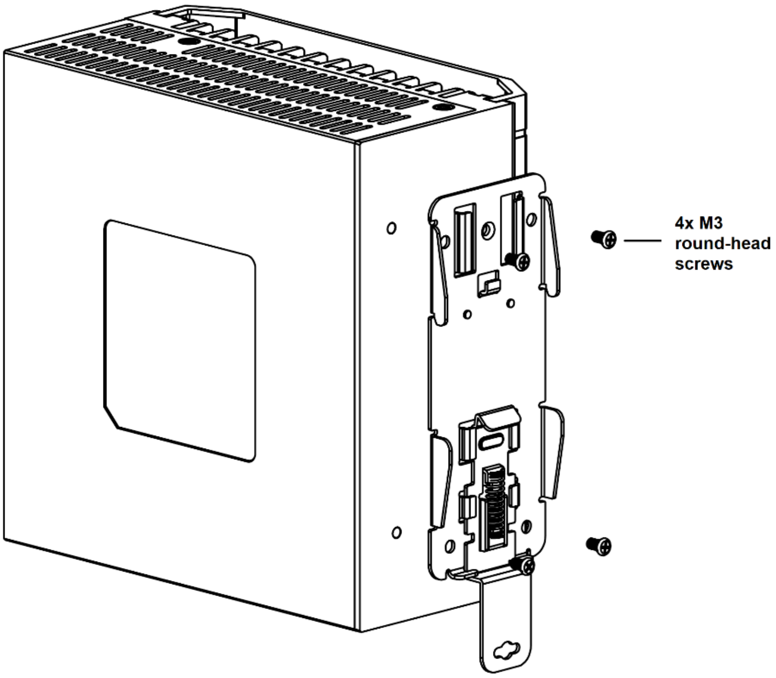


- **Uninstalling an SD card**

Push the SD card further into the slot to eject the card.

3.1.2 Installing on a Mounting Rail

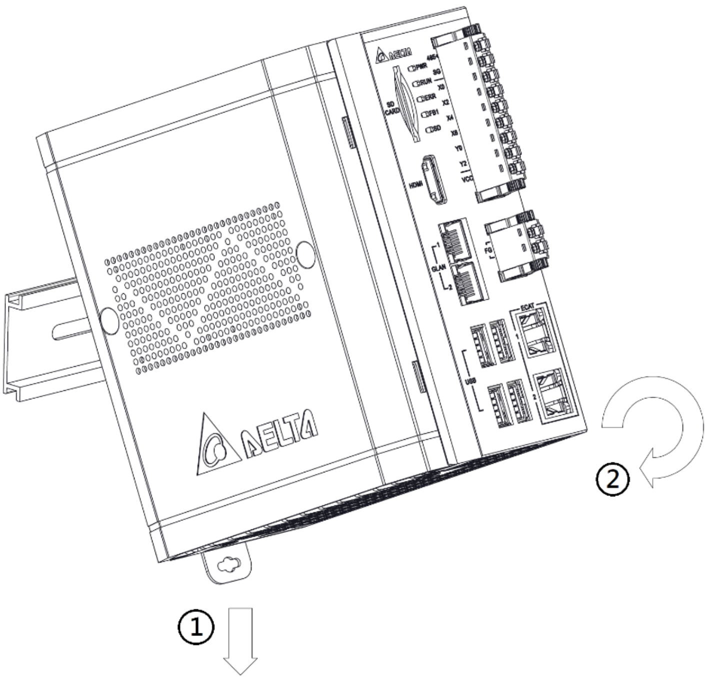
Turn to the back side of the AX-8 Series PLC as shown in the figure below. Screw the fixing clip set onto the AX-8 Series PLC with M3 round-head screws.



- **Installation**

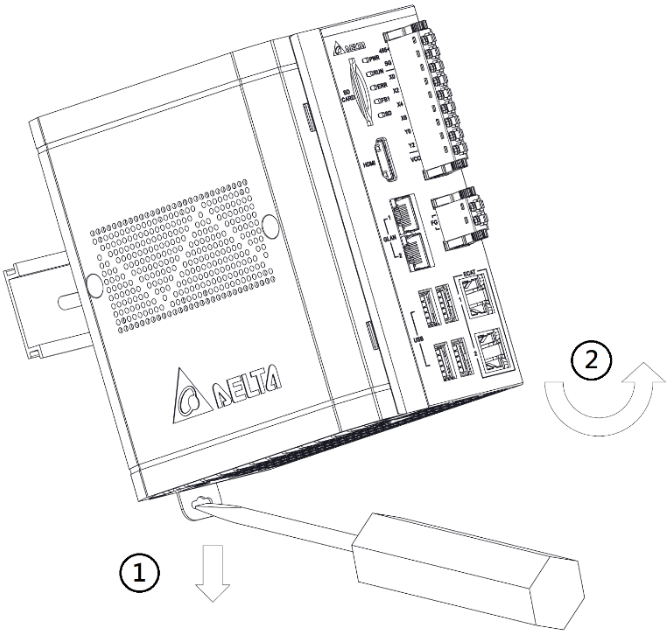
1. Pull out the fixing clip at the rear of the AX-8 Series PLC. Then edge the mounting rail with the horizontal slots which are also at the rear of the AX-8 Series PLC in direction ①.

2. Push the AX-8 Series PLC in direction ② till it clicks to lock it firmly on the mounting rail.



- **Uninstallation**

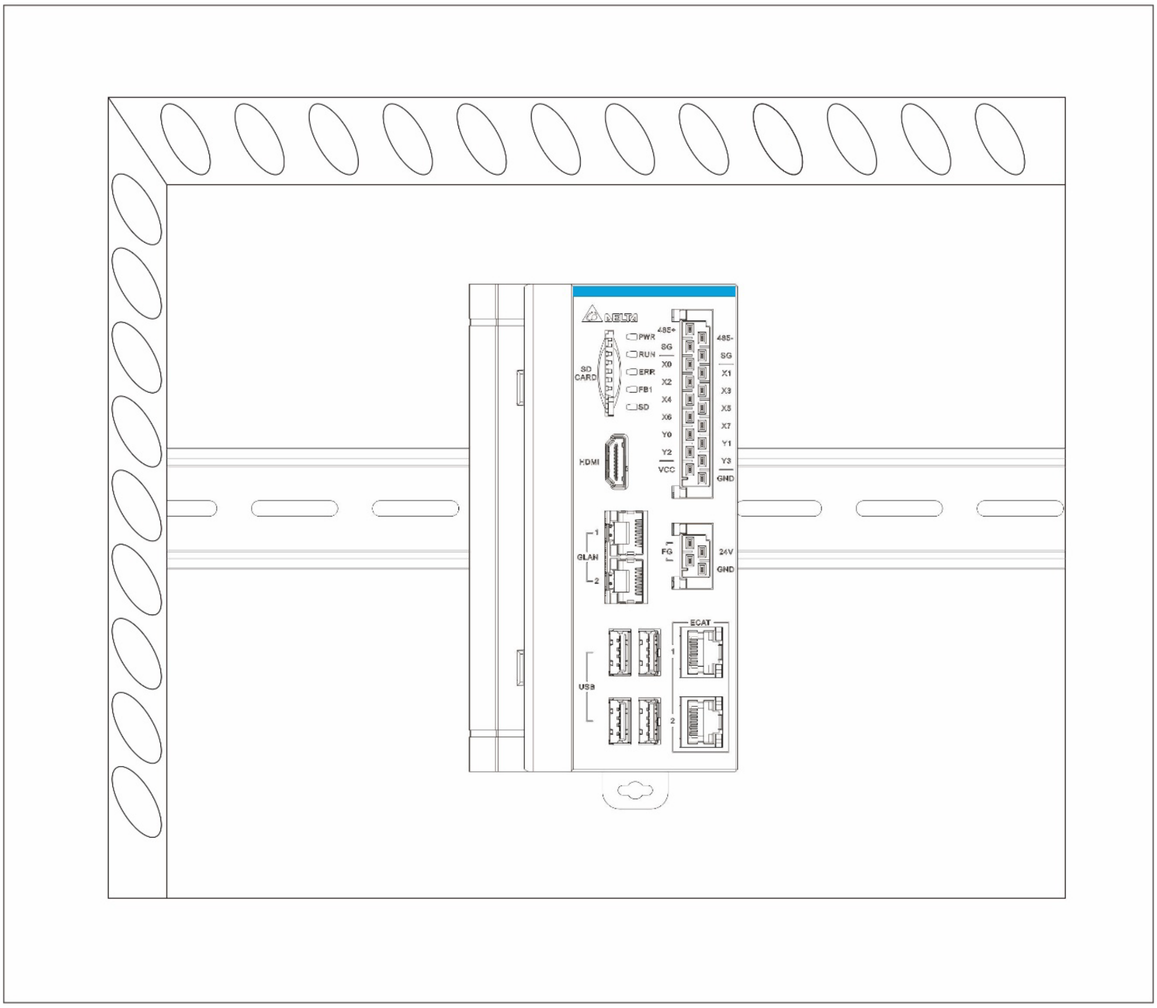
Pull down the fixing clip at the rear of the AX-8 Series PLC in direction ① with the tip of a screwdriver. Then pull the AX-8 Series PLC in direction ② to remove it from the mounting rail.



3.1.3 Installing in the Control Cabinet

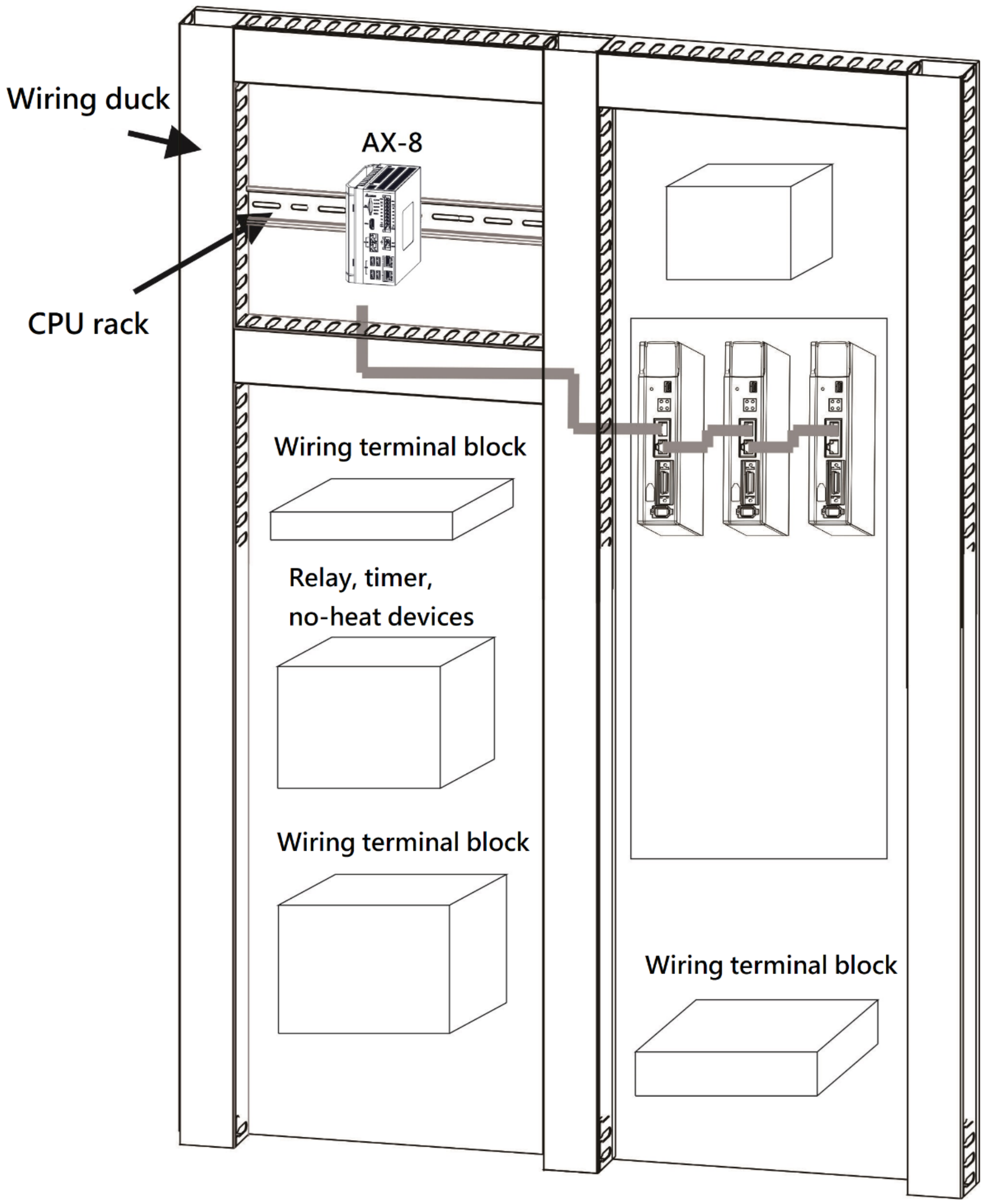
3.1.3.1 Installation on a DIN Rail

Pull out the fixing clip at the rear of the AX-8 Series PLC. Then edge the DIN rail with the horizontal slots which are also at the rear of the AX-8 Series PLC. Push the AX-8 Series PLC to fit with the DIN rail till it clicks to have the AX-8 Series PLC firmly installed on the DIN rail.



3

3.1.3.2 Installation in the Control Cabinet



3.1.3.3 Ambient Temperature Requirements for the Control Cabinet

- **Requirements for installing AX-8 Series PLC in the control cabinet**

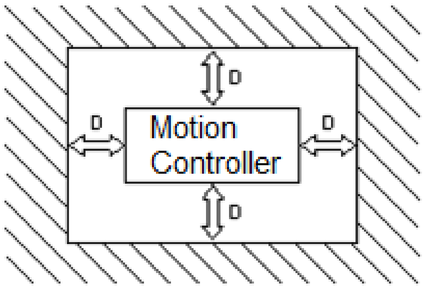
⚠ Warning
■ The ambient temperature of the control cabinet should be 0 ~ 50°C and the humidity should be 5 ~ 95%. ■ Avoid installing AX-8 Series PLC near equipment with high temperature. ■ Keep sufficient space for air ventilation. ■ Install fans or air conditioner if the ambient temperature exceeds 55°C. ■ AX-8 Series PLC is for indoor use only. ■ Install the control cabinet around 1.0 ~ 2.0m in height for easier installation and operation. ■ Keep away from the high-voltage equipment or power equipment during installation. ■ Disconnect power supply of the control cabinet before installation.

3.1.3.4 Measures to Enhance Anti-interference Ability

⚠ Warning
■ Avoid installing the AX-8 Series PLC in the control cabinet with high-voltage equipment. ■ Keep a distance of at least 200mm from the power wires during installation. ■ The control cabinet should be grounded. ■ Use the AX-8 Series PLC in accordance with the specified instructions by the manufacturer or the protection provided may be compromised.

3.1.3.5 Dimension Requirements for the Control Cabinet

- **Internal installation**

The AX-8 Series PLC should be installed in an enclosure. Keep a distance of at least 50 millimeters around the controller to ensure normal heat dissipation condition.	(D > 50mm)

3.2 Installing and Uninstalling DIADesigner-AX

Before starting the development work of AX-8 Series system, you must acquire the software **DIADesigner-AX**, which is the integrated platform for program development, hardware and network configuration.

- **System requirements**

The table below indicates the system requirements for DIADesigner-AX installation.

Item	System Requirements
Version*1	DIADesigner-AX V1.0.0.0 or later
Operating System	Windows 7 / 8.1 / 10 (32/64 bits)
CPU	Intel Celeron 540 1.8GHz (min.) Intel Core i5 M520 2.4GHz (min.)
Memory	2GB or above (4GB or above is recommended)
Hard Disk Drive	10GB or above
Monitor	Resolution: 1920 x 1080 pixels is recommended
Keyboard/Mouse	General keyboard and mouse or Windows-compatible devices
PC interface	Ethernet, USB, Serial port (depending on product interface)
Software	Microsoft .Net Framework 4.6.2

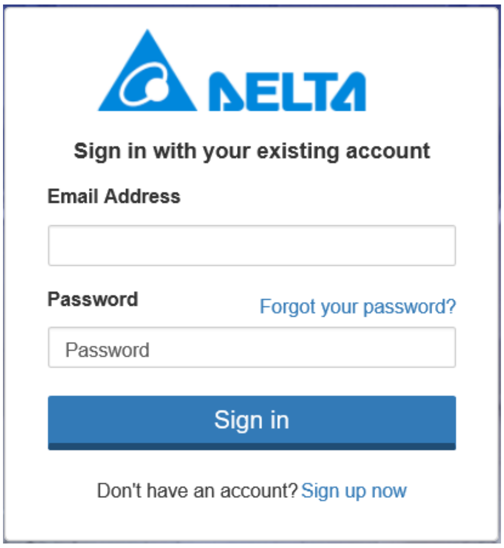
*1: The latest version available is V1.4 as of this writing. Please refer to the official website - DIAStudio for later releases.

3.2.1 Installing DIADesigner-AX

Before installation, ensure the computer used for installing DIADesigner-AX meets the minimum system requirements listed in section 3.2 and has been equipped with DIAInstaller.

DIAInstaller is used to easily download, install, and update all DIAStudio software including DIADesigner-AX. You can download **DIAStudio** for **DIAInstaller** from https://diastudio.deltaww.com/home/downloads.

You have to sign in or sign up before accessing the download page.



After login, click **DIAStudio** download button to download **DIAInstaller** as the image shown below.

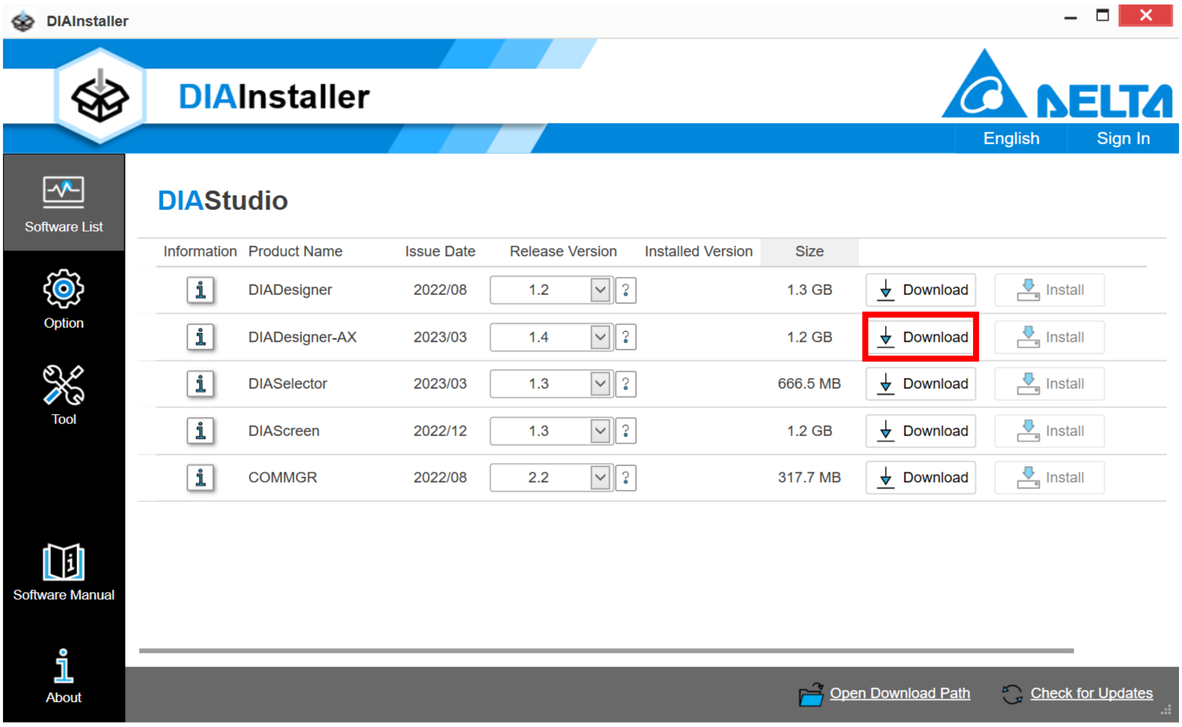
Software

Software Name	Description	OS	Issue Date	File
COMMGR V2.2.1	COMMGR V2.2.1	Windows 10 64 bits	2023/02/20	
DIAStudio V1.2.2	DIAStudio Software download and Installation Tool	Windows 10 64 bits	2023/06/16	

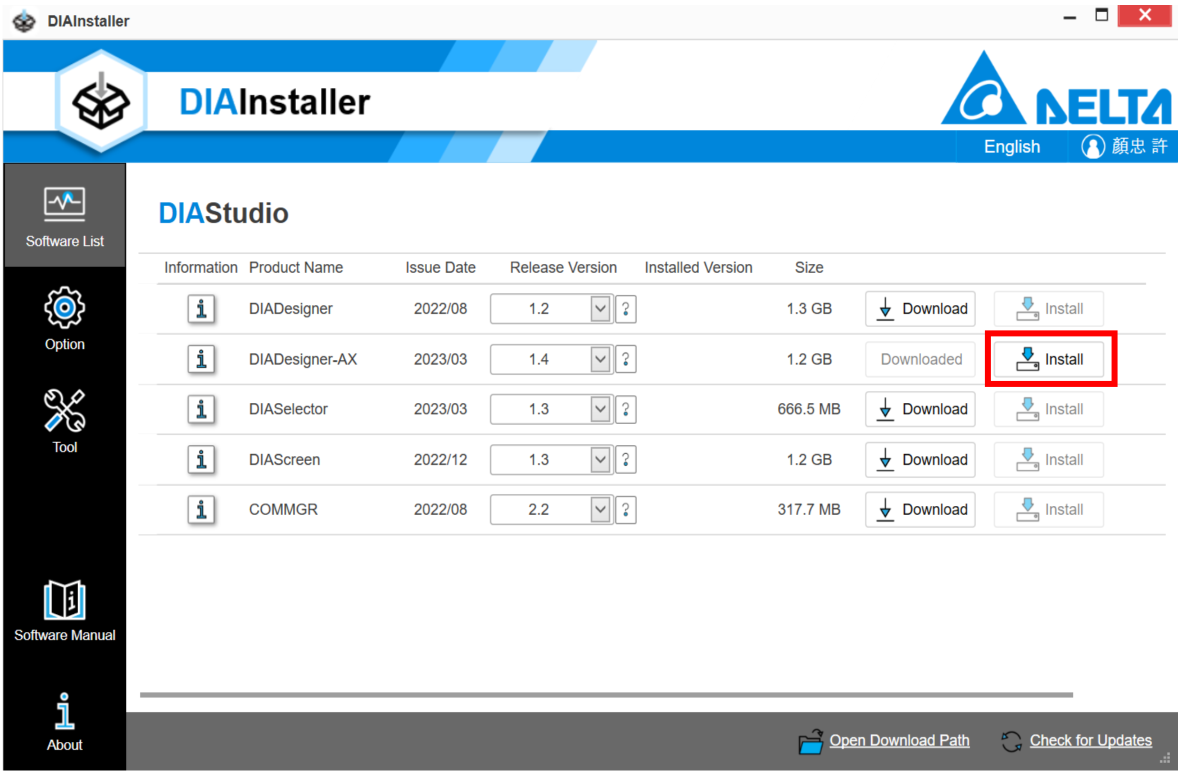
Follow the steps below to install DIADesigner-AX.

1. Double-click **DIAInstaller** icon to check the latest version of DIADesigner-AX.

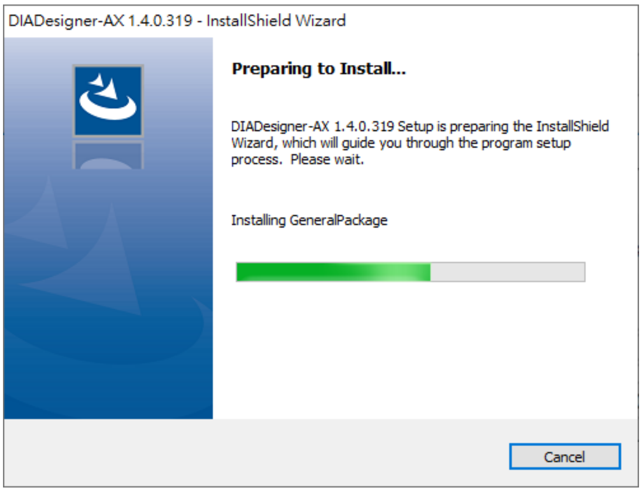
2. Click **Download**.

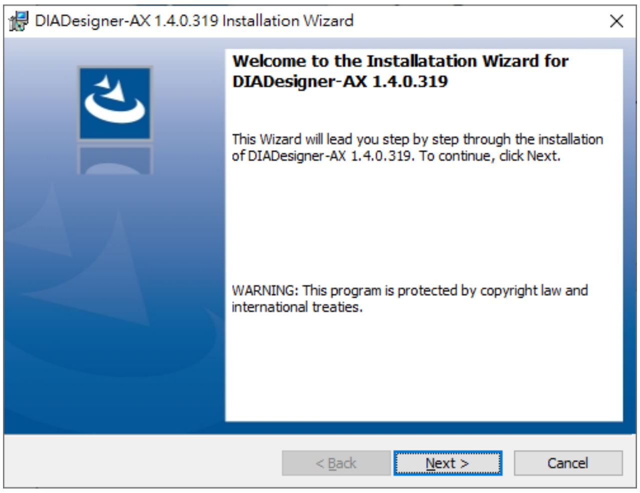


3. After DIADesigner-AX is downloaded and grayed out, click **Install**.

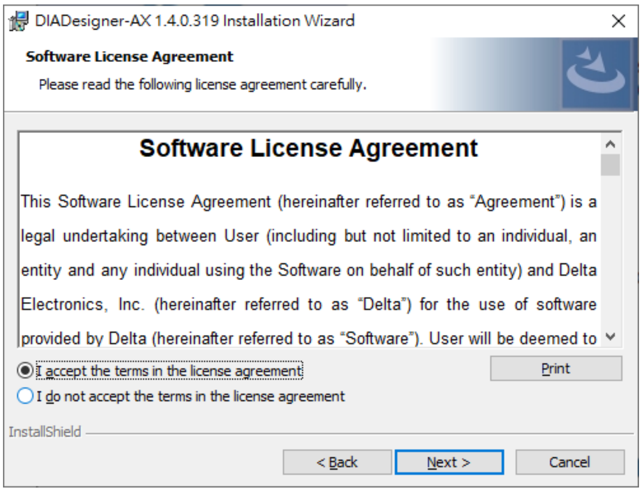


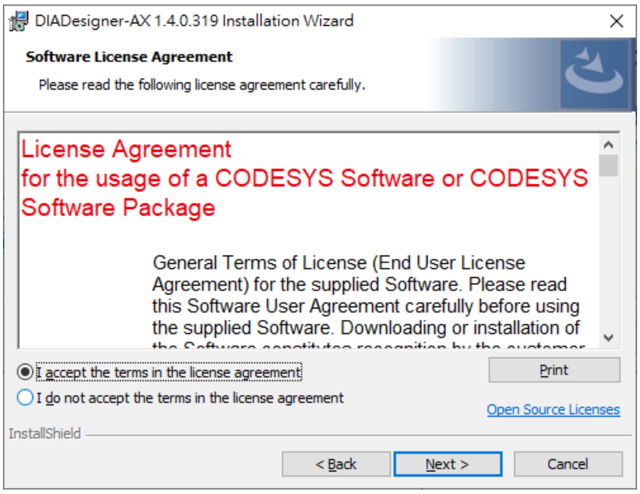
4. An **InstallShield Wizard** shows up and starts installing. Click **Next** after completion.



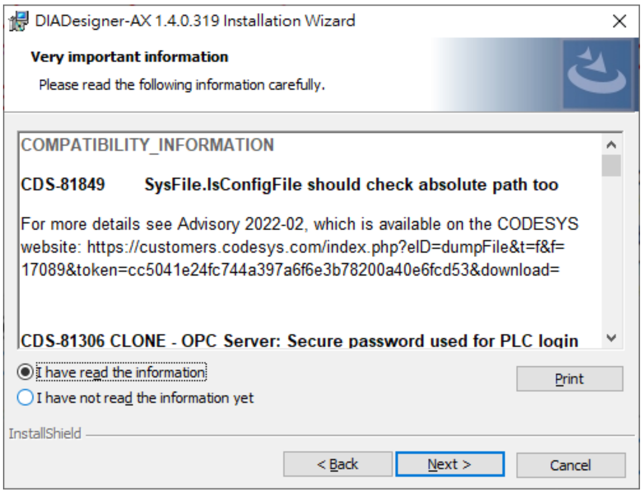


5. The window of **License Agreement** shows up. Select **I accept the terms in the license agreement** and click **Next**.

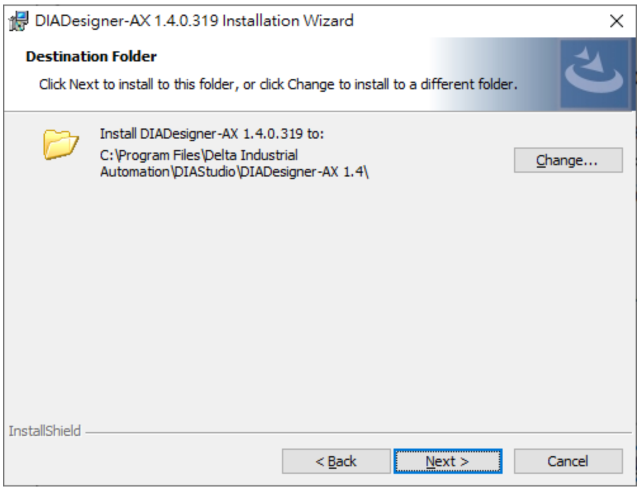


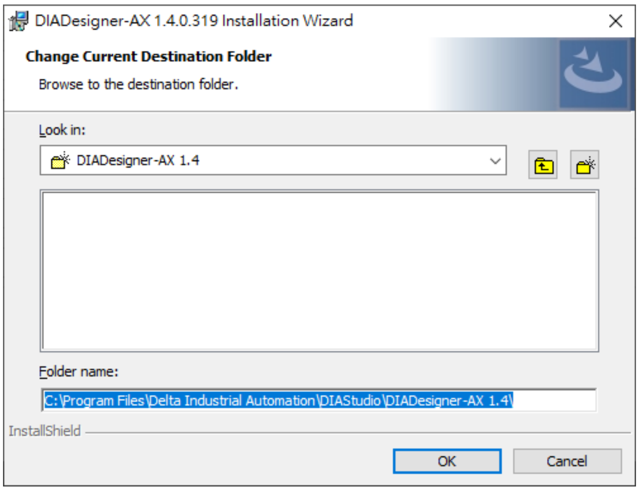


6. The window of **Very important information** shows up. Select **I have read the information** and click **Next**.

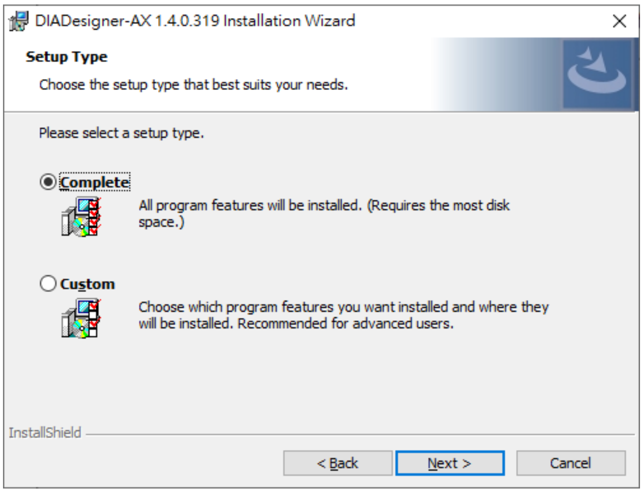


7. Click **Change…** to change the download path. Or leave the default path unchanged. Click **Next**.

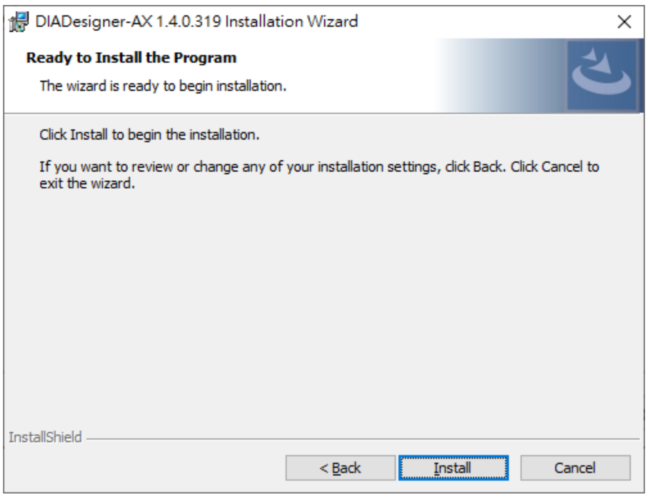




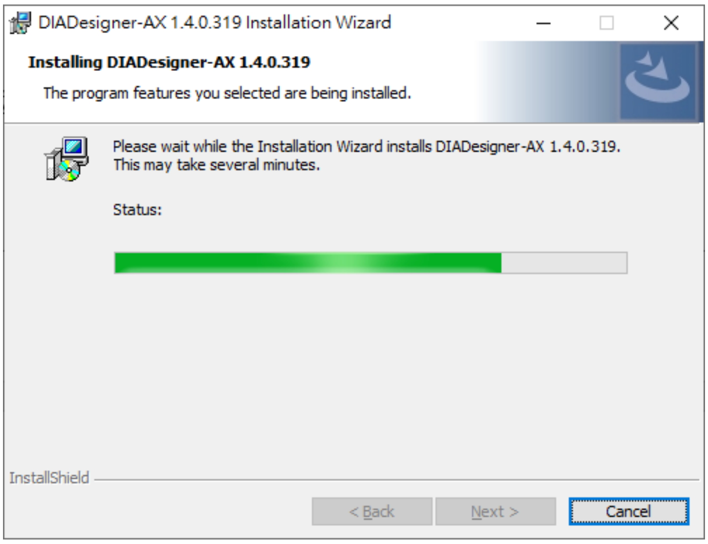
8. The window of **Setup Type** shows up as the image shown below. Select the one you need and then click **Next**.



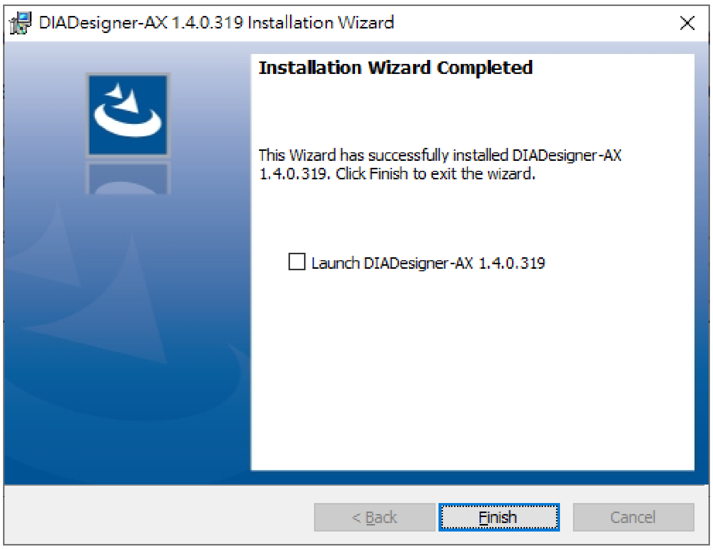
9. The window of **Ready to Install the Program** appears as below. Click **Install** to continue.



10. It may take some time to install.



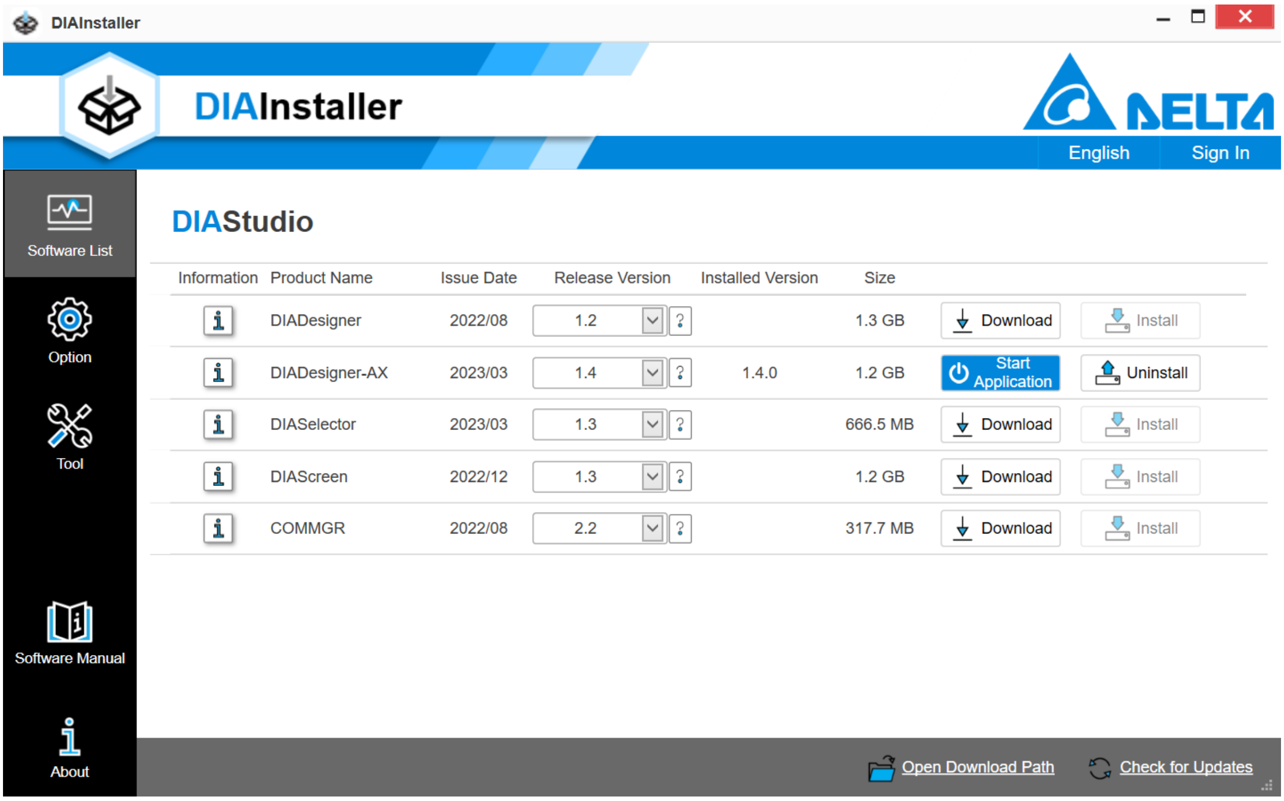
11. After installation, the window of **InstallShield Wizard Completed** appears. Click **Finish** to complete the installation.



3.2.2 Uninstalling DIADesigner-AX

Follow the steps below to uninstall DIADesigner-AX.

1. Double-click **DIAInstaller** icon to open and then click **Uninstall**.



2. The system will remove DIADesigner-AX from your computer in the background automatically.

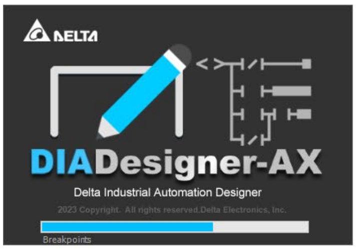
3.3 Getting Started and Setting up Communication

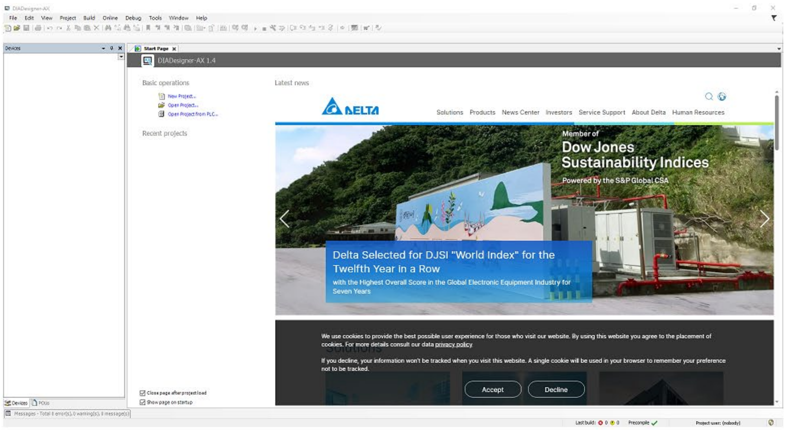
3.3.1 Getting Started

After DIADesigner-AX is successfully installed, click **Start** . You can find it under the folder of **Delta Industrial Automation** or find its shortcut on the desktop. Double-click either one to start the software. Besides, you can open more than one DIADesigner-AX software to achieve multitasking.

OR

After the display of the welcome page, you can see the homepage as below. Refer to **Chapter 4** for more details on operation.



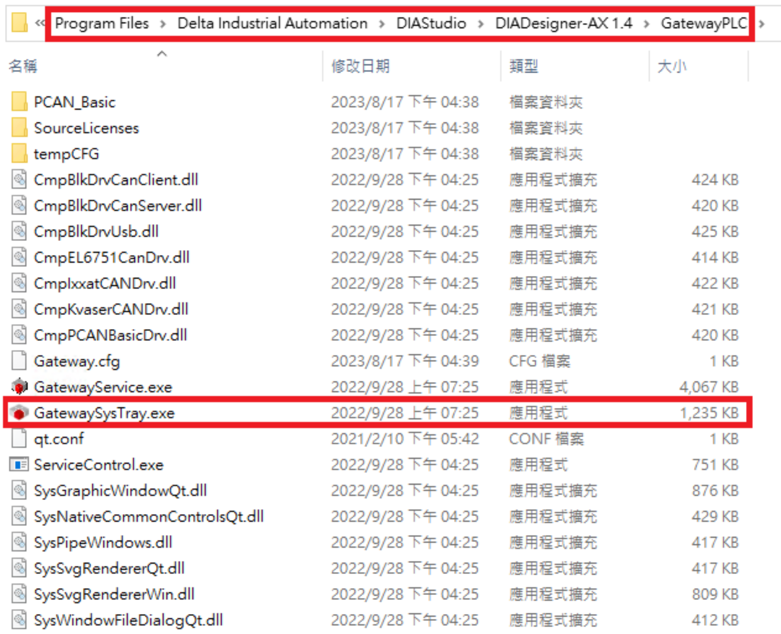


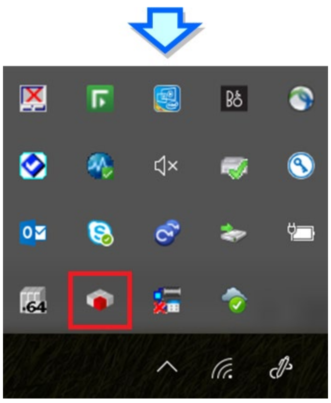
3.3.2 Setting up Communication

After DIADesigner-AX is successfully installed, the system creates an execution file **CODESYS Gateway V3** under the folder of Delta Industrial Automation and **GatewaySysTray.exe** in the folder of Program Files. Double-click either one to start the Gateway.

Hereafter, the Gateway is started automatically whenever you turn on your computer. Its icon  shall appear on the taskbar. If not, go to the execution file **CODESYS Gateway V3** under the folder of Delta Industrial Automation or **GatewaySysTray.exe** in the Program Files folder to start the Gateway manually.

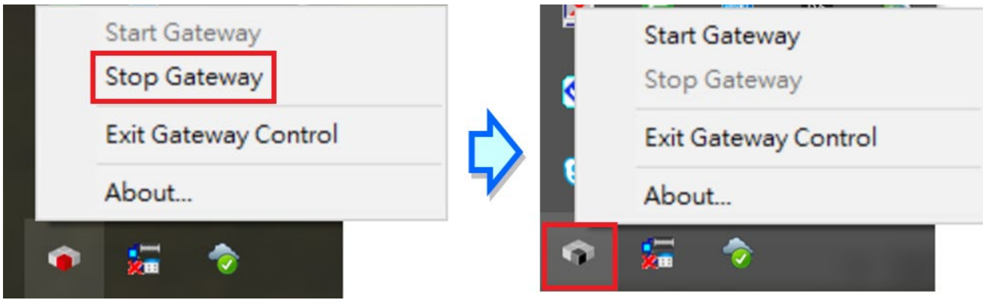
OR





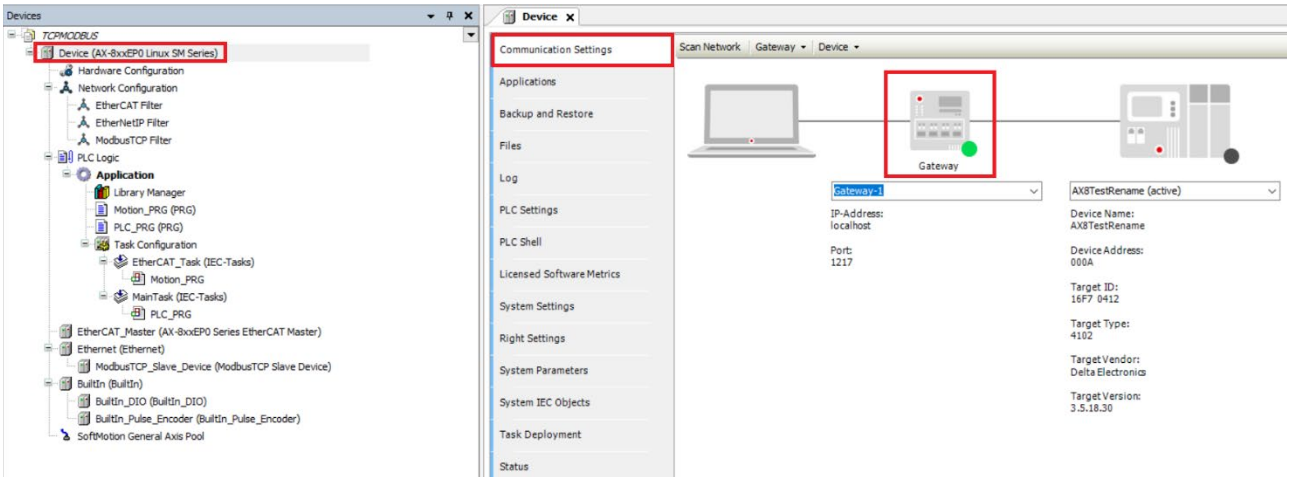
You can click the Gateway icon  on the taskbar to check its status. The icon is in red when the Gateway is started.

Click **Stop Gateway** if you need to stop gateway working. The icon turns grey when the gateway is stopped.

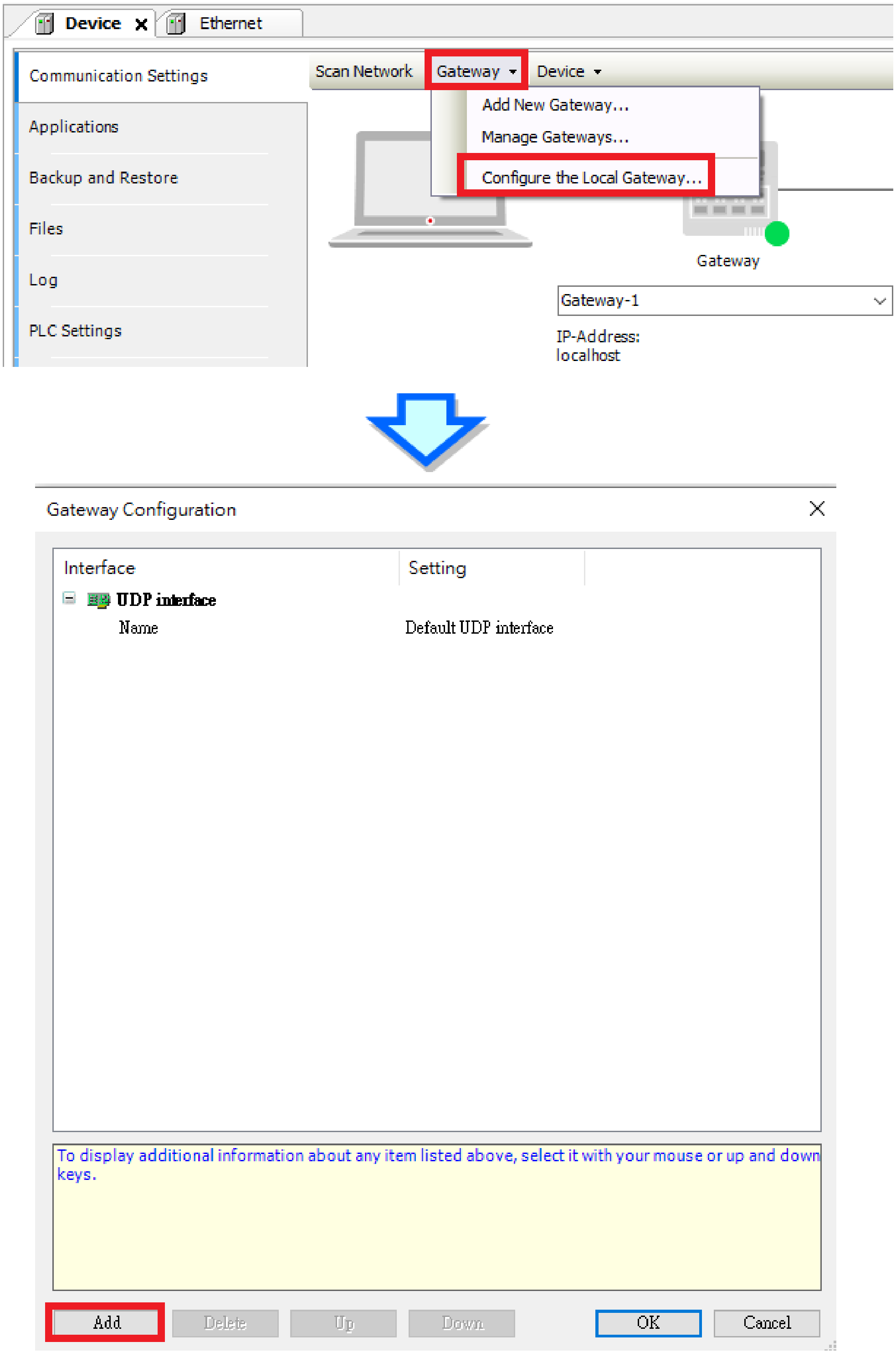


If you need to discontinue the execution of GatewaySysTray completely, select **Exit Gateway Control** and the icon  will disappear from the taskbar.

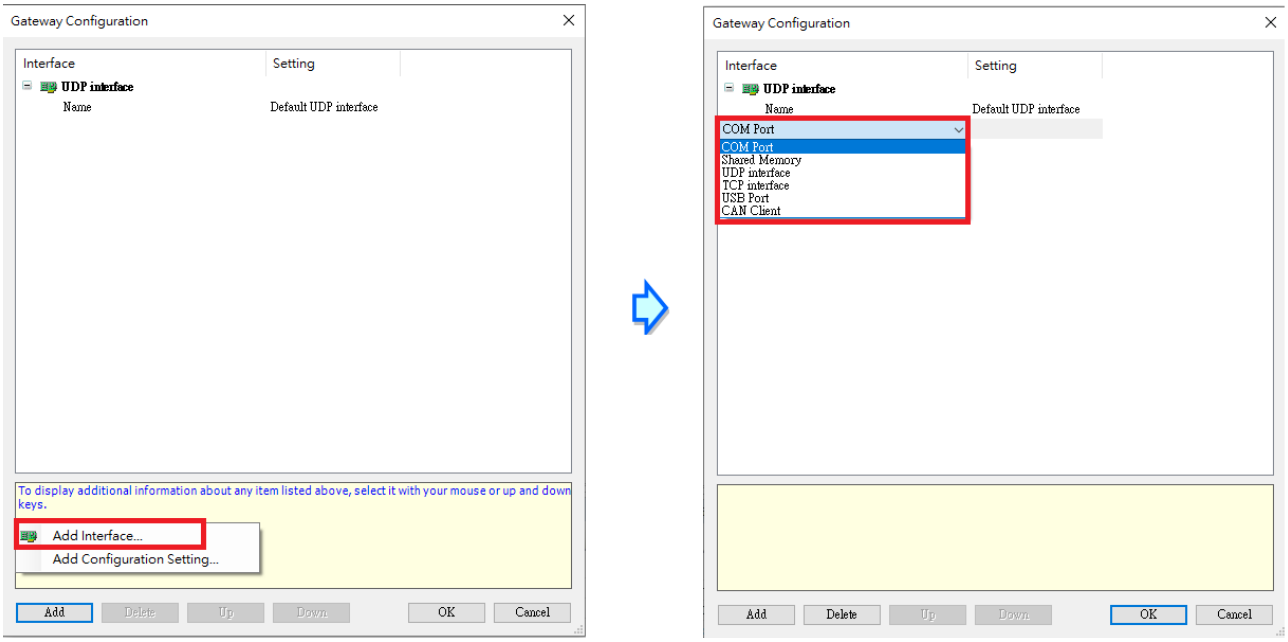
Get into the software **DIADesigner-AX** and open or create a project to see the project-setting page. Double-click **Device** (Product Name) in the left-hand column to open the device-setting page. The Gateway status is under the **Communication Settings** tab. If the Gateway is started, its light is green. If the Gateway is stopped, its light is red.



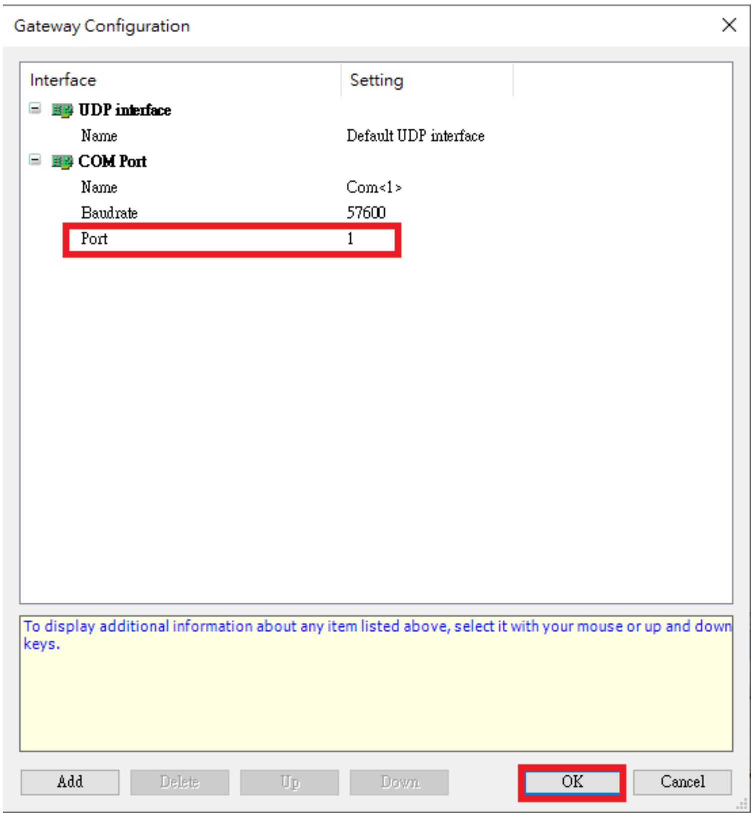
To configure the Local Gateway, click **Gateway** and select **Configure the Local Gateway…** to open the setting window.



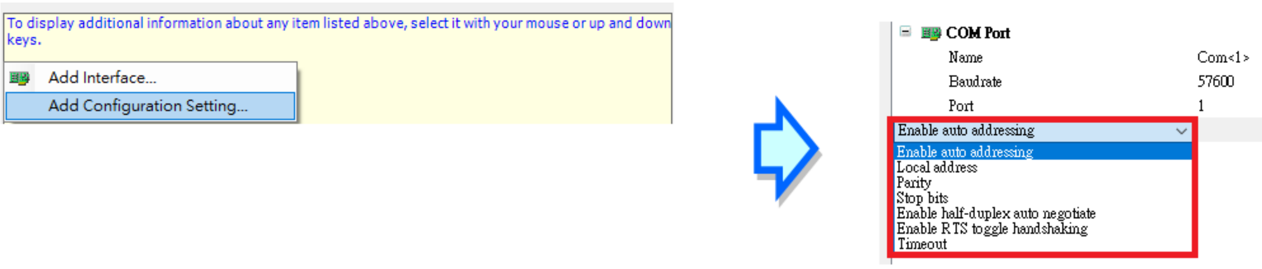
You can create a different port such as COM Port for connection. Click **Add**, select **Add Interface…** and then use the drop-down list above to select **COM Port** to add.



After COM Port is added, you can set up its name, baud rate and the corresponding port. Click **OK** once the setting is done. You should Stop/Start GateSysTray again to ensure the following actions, such as Scan Network, work properly. Please refer to the previous steps to run GateSysTray again.

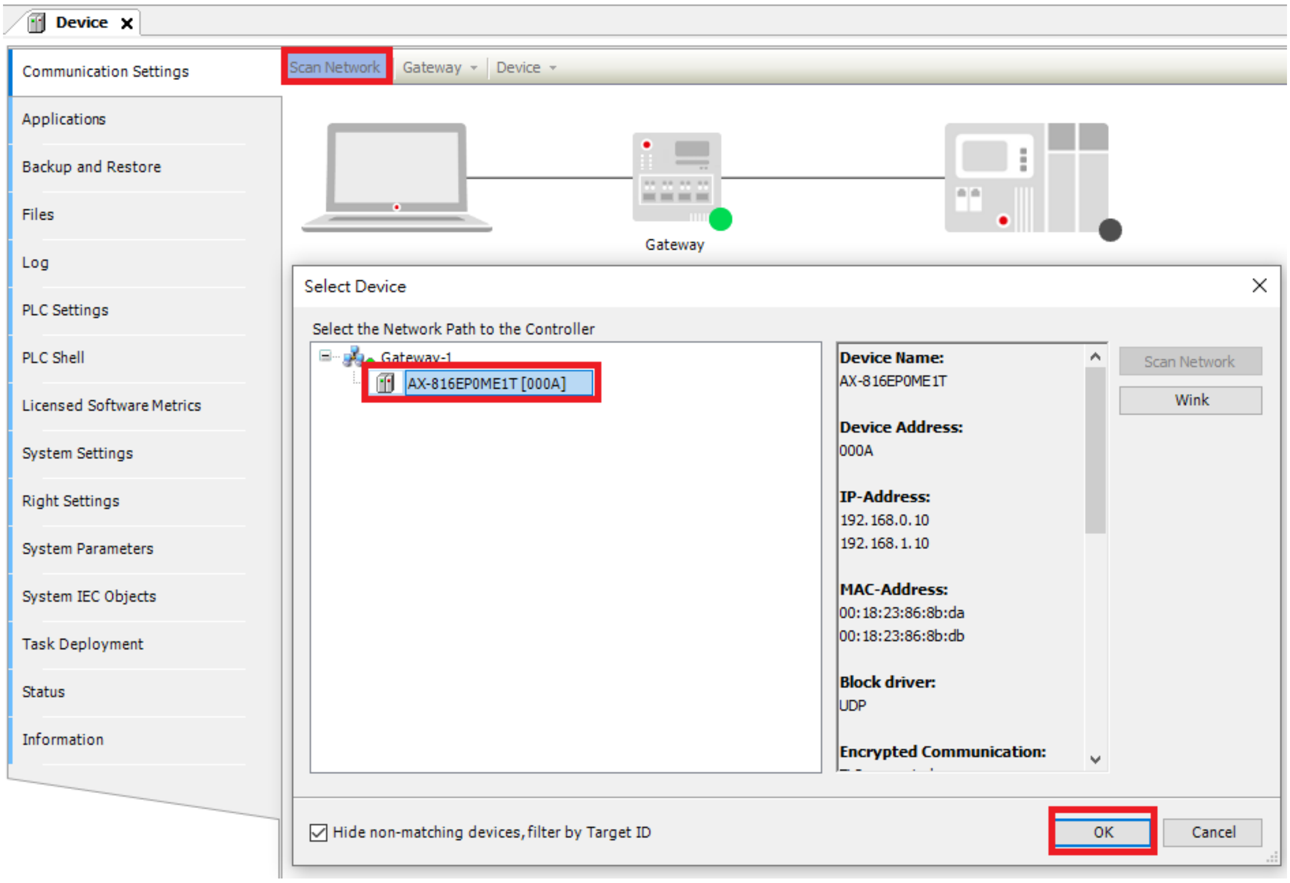


You can also add configuration settings under COM Port. Right-click the **COM Port** icon COM Port , click **Add** and select **Add Configuration Setting…** to add the setting items. After that, you can further define the setting values. Click **OK** once the setting is done.

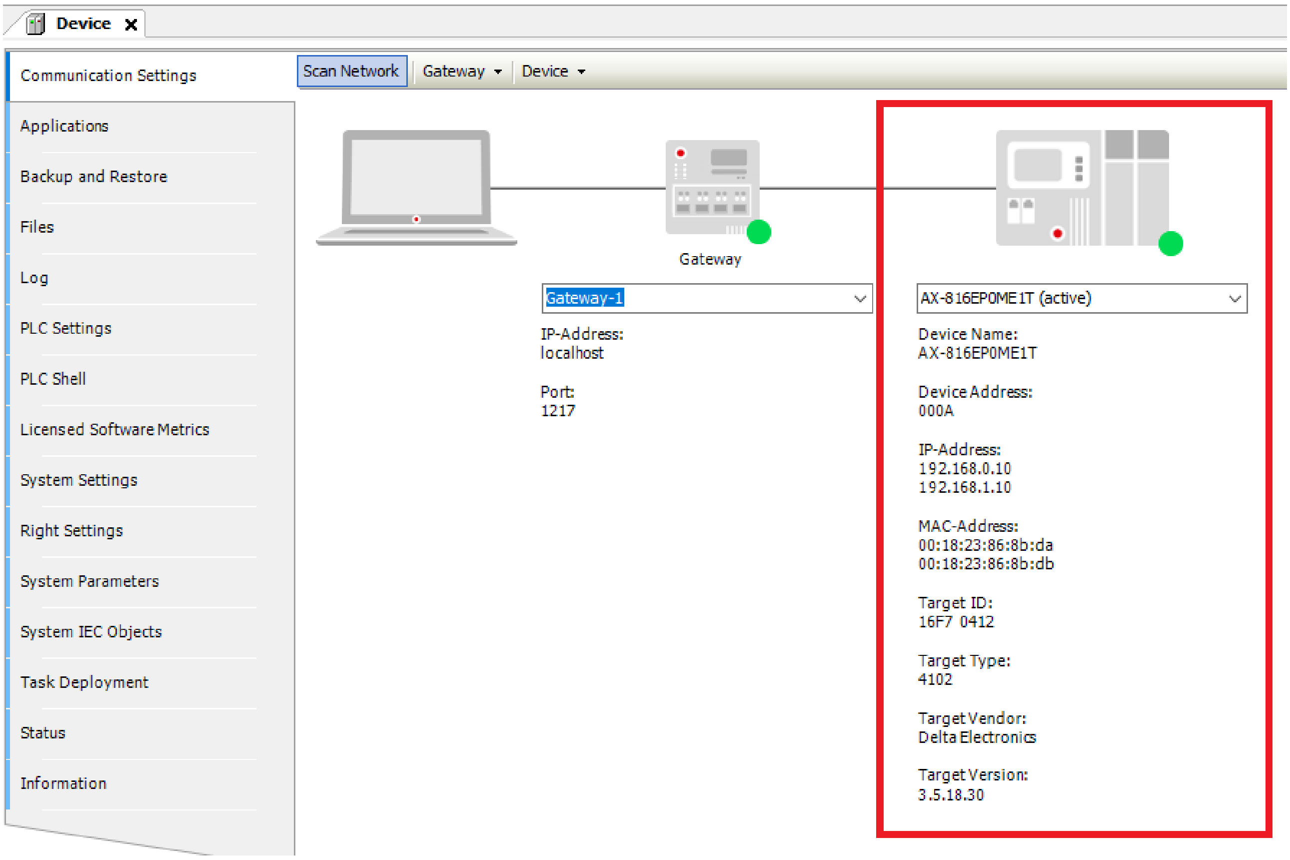


3

After the configurations of Local Gateway are nailed down, go to the **Scan Network** tab to bring out network scanned results on the **Select Device** setting page. Select **AX-816EP0ME1T** and then click **OK.**



If the connection is successfully established, the status light is green with the device information shown below.



MEMO

3

4

Chapter 4 Basic Operation

Table of Contents

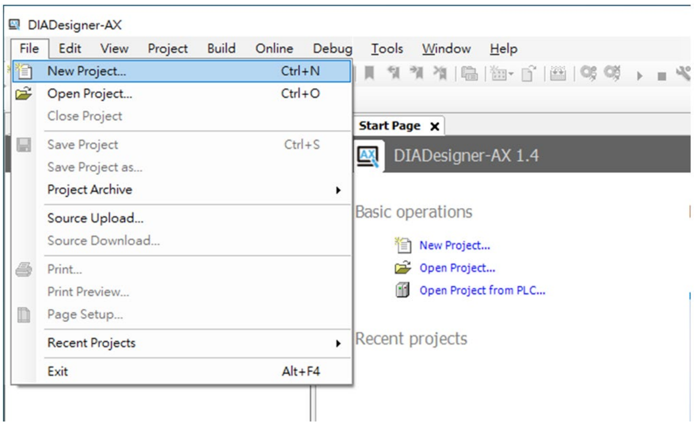
4.1 Introduction to DIADesigner-AX

DIADesigner-AX is an open platform for PLC development system and the core for industrial automation. The adaptable DIADesigner-AX provides an easy way to create professional engineering of IEC 61131-3 automation projects. Based on the IEC 61131-3 data structure and the high-level language programming, DIADesigner-AX is strong in functionality, easy to develop, reliable, extendable and open for development. Integrated with components such as visualization and safety solution, DIADesigner-AX offers a variety of user-friendly engineering functions for your professional applications in controller development system sectors including PLC and motion control.

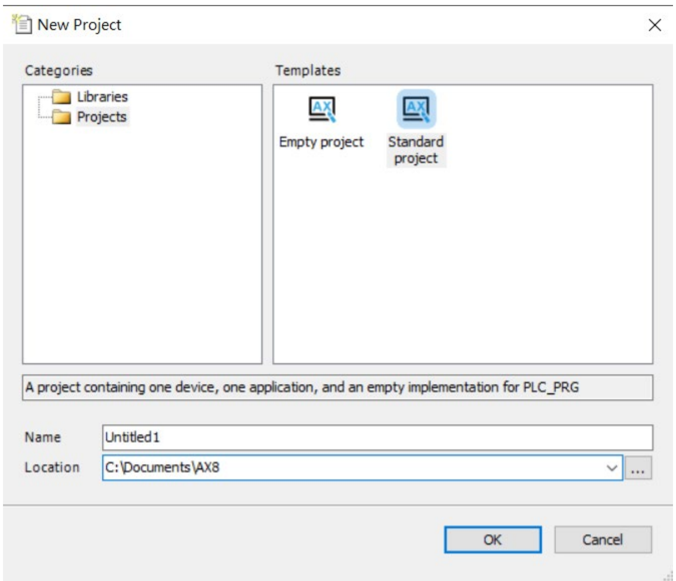
In DIADesigner-AX, you can customize the user interface by arranging the window layout and the appearance of menus, toolbars and commands according to your requirements.

4.1.1 Creating a New Project

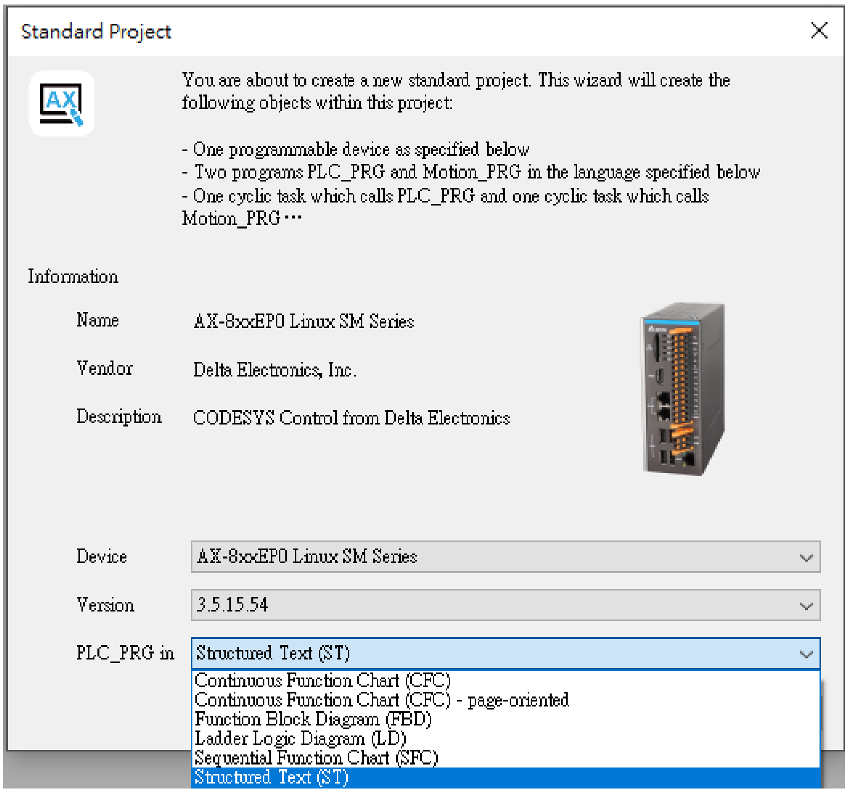
1. Double-click the DIADesigner-AX icon to open the software. To create a new project, click **New Project** on the Start Page or click **File** on the toolbar and select **New Project** (Ctrl+N).



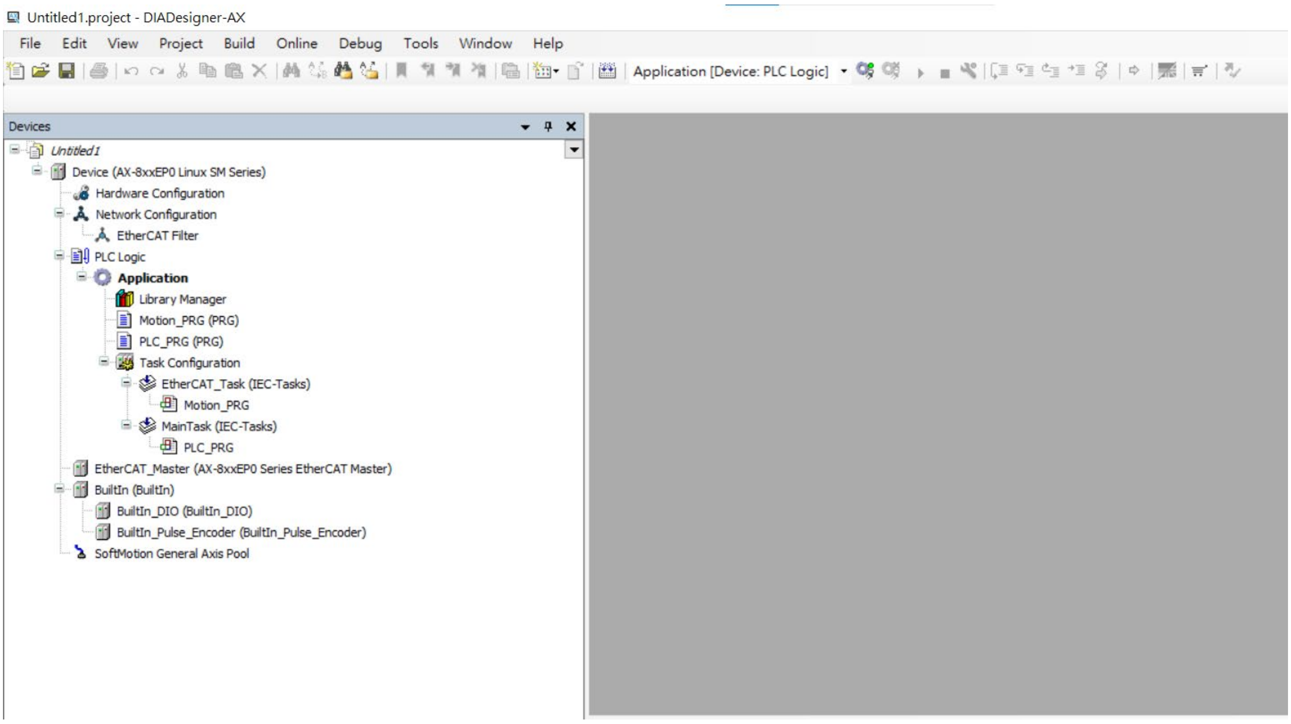
2. A window with two sections, Categories and Templates, will pop up. Click **Projects** in the Categories section and choose **Standard project** in the Templates section. After that, create a name and specify a location for the project and then click **OK**.



3. The Standard Project dialog then appears. Select the device and the programming language from the drop-down lists. Click **OK** after selection, the system will generate a cyclic task with a default PLC_PRG.

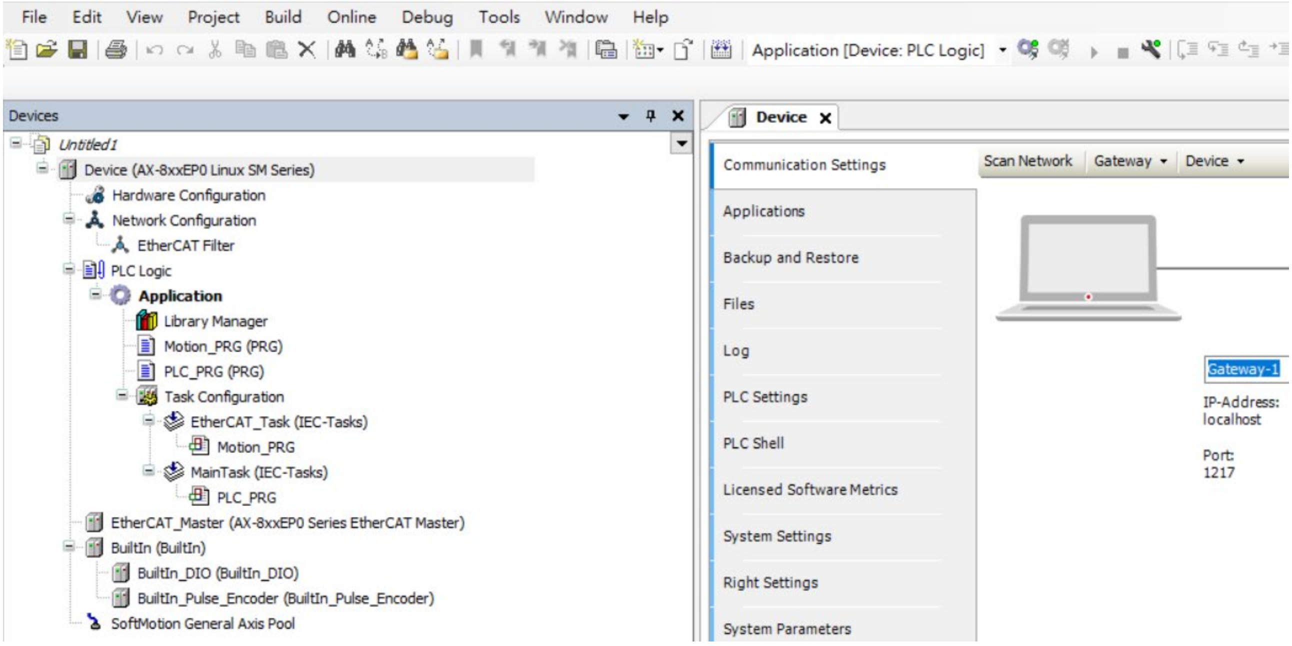


4. After a new project is created, a project management area will be added on the left-hand side. All the options are listed in nodes. If nothing appears in the project management area, click **View** > **Devices (Alt+0)** on the toolbar to open it.



4.2 Settings on the Device Page

In this section, all setting items on the Device page will be introduced respectively.



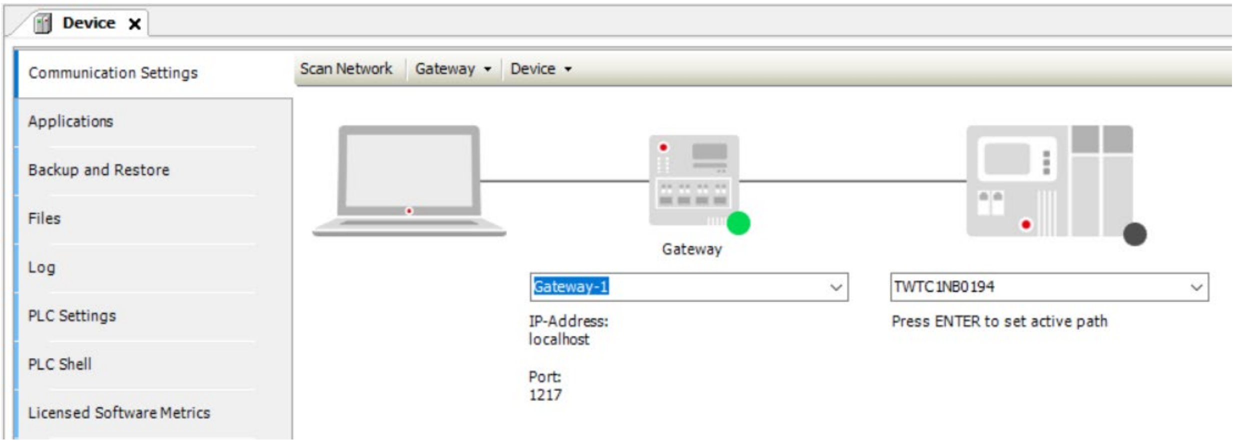
4

4

4.2.1 CPU Parameter Settings

4.2.1.1 Communication Settings

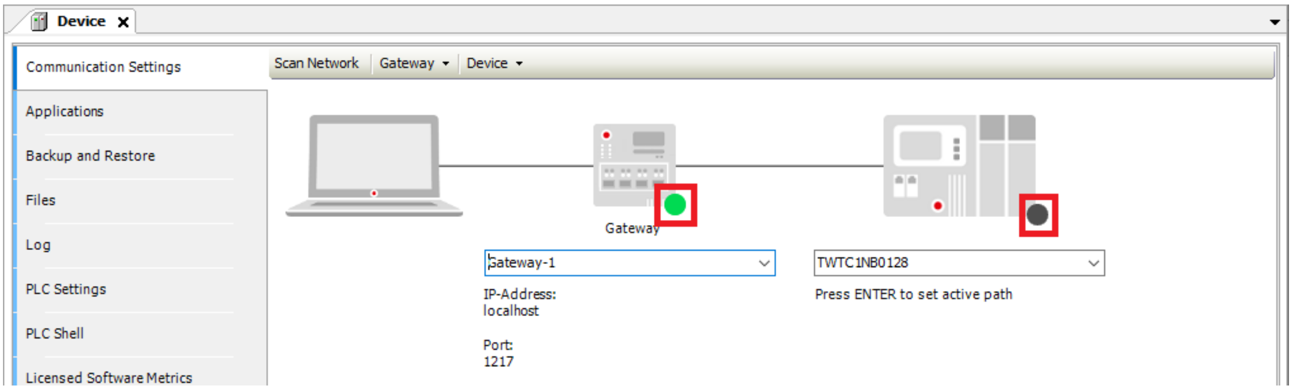
On the **Communication Settings** page, you can define method for communication between DIADesigner-AX and the controller. Use the drop-down list of the Gateway tab to add new gateways, manage existing gateways or configure local gateways. You can simply specify an IP address (e.g. 192.168.1.5) or its device name (e.g. AX-816EP0ME1T) in the field under the image of controller. After that, DIADesigner-AX will automatically scan for the controller within the network of the Gateway.



- **Connection Status**

The dots in the bottom right corner of the images of Gateway and the controller indicate the connection status.

- Red: unable to establish connection.
- Green: connection is established.
- Black: unknown connection status.



Tab	Description
Scan network	Click **Scan Network** to open the **Select Device** window. All configured gateways with the associated controllers are listed here. You can select a target device from the list.

Gateway	The following setting items are included in this menu: ■ **Add New Gateway**: add and define a new gateway channel. ■ **Manage Gateways**: an overview of all gateways is provided. You can add / delete entries or change the order here. ■ **Configure the Local Gateway**: click to open the **Gateway Configuration** page. You can configure the block drivers for the local gateway here.
Device	The following setting items are included in this menu: ■ **Options** - **Add Current Device to Favorites**: add the selected device to the list of favorite devices. - **Manage Favorite Devices**: click to open a list of all preferred devices. You can add / delete entries or change their order here. The device on the top is set as default. - ☑ **Filter Network Scans by Target ID**: only the device whose target ID is the same as the current device configured in the project is displayed. - ☑ **Confirmed Online Mode**: DIADesigner-AX requires you to confirm when calling the following online commands for safety purposes: Force values, Write values, Multiple loading, Remove force list, Single cycle, Start, and Stop. - **Store Communication Settings in Project**: ☑: DIADesigner-AX stores the communication settings in the project, which can be reused on the same computer later. If you use this project on another computer, the valid path must be reset. ☐: DIADesigner-AX stores the communication settings in the locally installed options and user can reuse this setting on the same computer. ■ **Rename Active Device** Click to open the **Change Device Name** window. ■ **Send Echo Service** DIADesigner-AX sends five echo services like the networking utility, Ping, to the controller to test network connections. These services are sent without data packet in the beginning and then sent with packet. The scope of data packets depends on the communication buffer of the controller. Information about each and the average response time for the transmission is shown on a new dialog box. ■ ☑ **Encrypted Communication** Communication to the controller is encrypted. A certificate is required to log in to the controller. If the certificate is not available, an error message will show up, prompting whether the certificate should be displayed / installed or not. Encrypted Communication is disabled when the **Enforce Encrypted Communication** option in the **Security Level** list is selected (**View > Security Screen**).

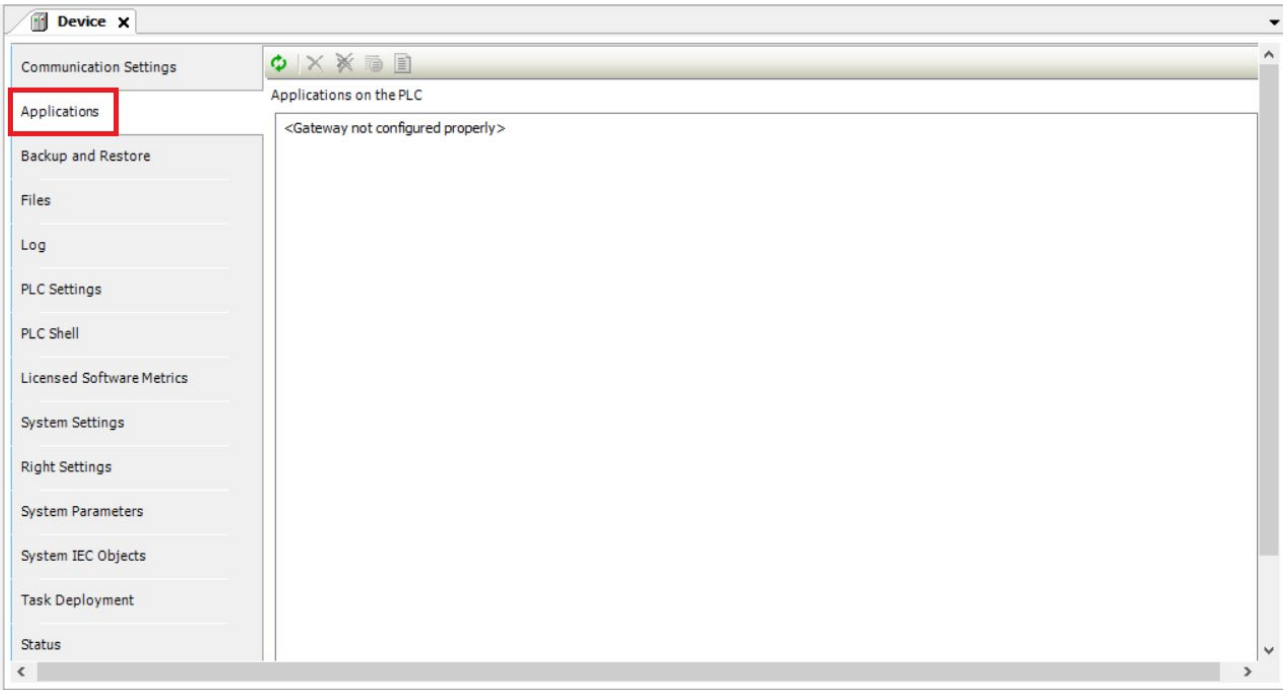
■ **Change Runtime Security Policy**

Click to open the **Change Runtime Security Policy** window to set encrypted communication for the device. The configuration of the controller will also be changed if the communication policy is modified.

Communication	
Current policy	The currently selected policy for communication encryption.
New policy	- No encryption: encrypted communication is NOT supported. - Optional encryption: both encrypted and unencrypted communications are supported. - Enforced encryption: only encrypted communication is supported.
Device User Management	
Current policy	The currently selected policy for user management.
New policy	- Optional user management: users can decide whether to activate encrypted communication or not. - Enforced user management: encrypted communication must be activated.

4.2.1.2 Applications

You can check and manage the applications on the controller on this page.

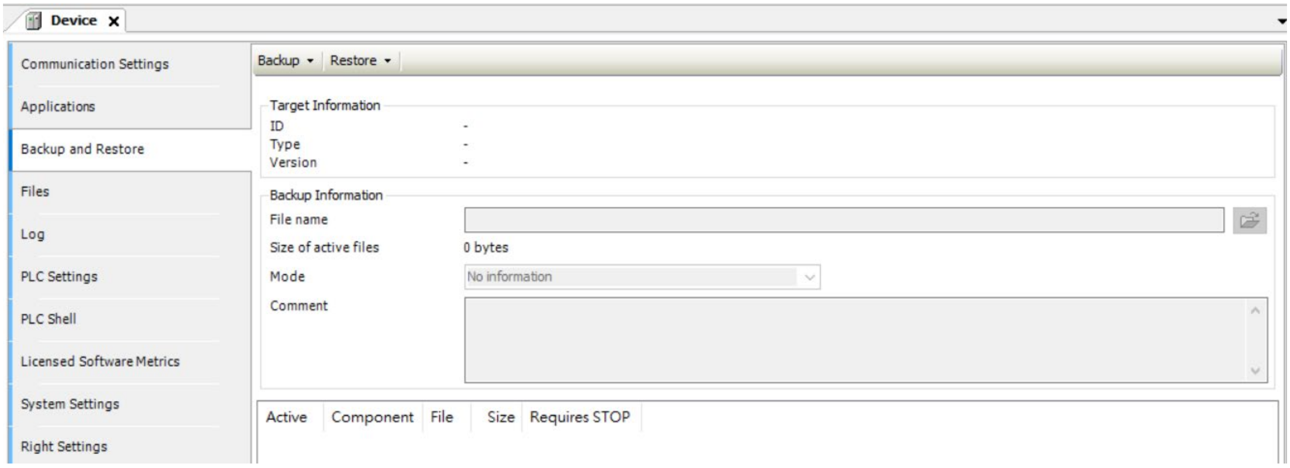


Item	Description
Refresh List	To have the controller scanned for applications to update the list accordingly.
Remove	To delete the application selected in the list.
Remove All	To delete all listed applications on the controller.
Details	To check the attributes defined for the application on the Information tab.
Content*1	To check additional information about the differences between the latest generated code and the application code that exists on the controller, which are displayed in a comparison view.

*1: It is required to click **Application** under **Applications on the PLC** to activate **Content** button (**Application** > **Properties** > **Application generation options**).

4.2.1.3 Backup and Restore

You can backup and restore the application-specific files on the controller by saving and reading zip archive.



Tab	Description
Backup	The items on the drop-down list are as follows: ■ **Read Backup Information from Device…**: searching and listing all application-specific files in the **PlcLogic** directory. ■ **Create Backup File and Save to Disc…**: compressing the chosen files into a backup zip file. (The file extension is .tbf, i.e. Target Backup File). ■ **Save Backup File to Device**: saving the backup file to the controller.
Restore	The items on the drop-down list are as follows: ■ **Load Backup File from Disc…**: selecting one backup file from the list found on the disk to view its content. ■ **Load Backup File from Device…**: selecting one backup file from the list found on the controller to view its content. ■ **Restore Backup to Device…**: restoring the backup file to the controller.

- **Target Information**

ID	Device ID
Type	Device type
Version	Device version

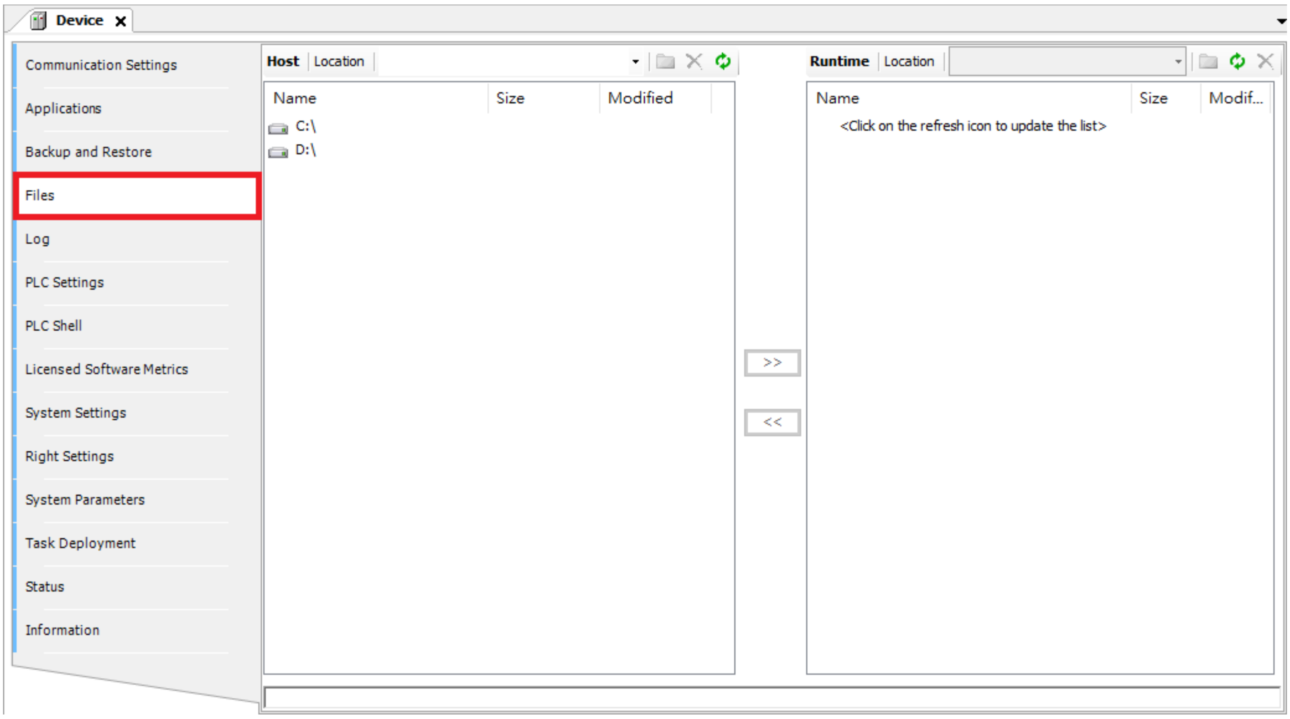
- **Backup Information**

File name	Storage path for the backup file
Size of active files	Total size of the backup files set as active in the table
Mode	The scope for backup

File name	Storage path for the backup file
Comment	Optional entry for comment of the backup file; saved in the meta.info file and can be read when the files are restored

4.2.1.4 Files

This section is for file transfer between the computer and the controller via DIADesigner-AX.



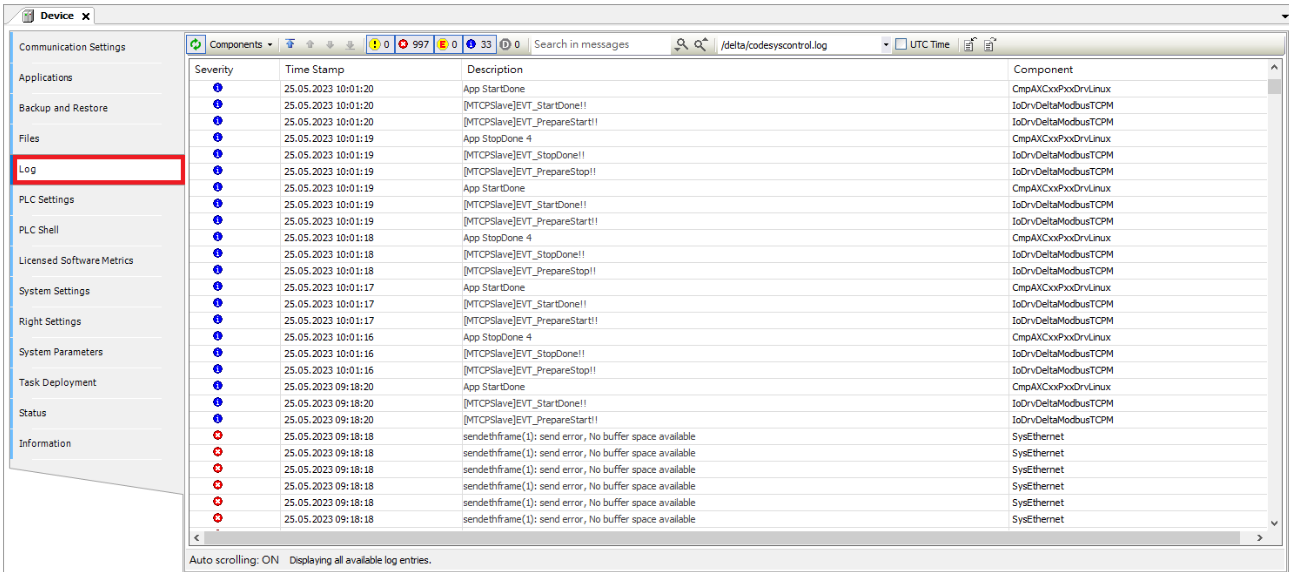
Location	Current file path in the computer system The files with its name, size, and date modified are listed in the subdirectories below.
	To create a new file folder
	To delete the selected files or folders
	To update the list of files and folders in the set path (Location)
>>	To write files to the controller
<<	To restore files from the controller

4.2.1.5 Log

In Log, it lists the events that occurred in the target system, including:

- Events which have occurred from system startup to shutdown (components loaded including version)
- Record of application operation including application download, online change, etc.
- Custom entries
- Log entries from I/O driver

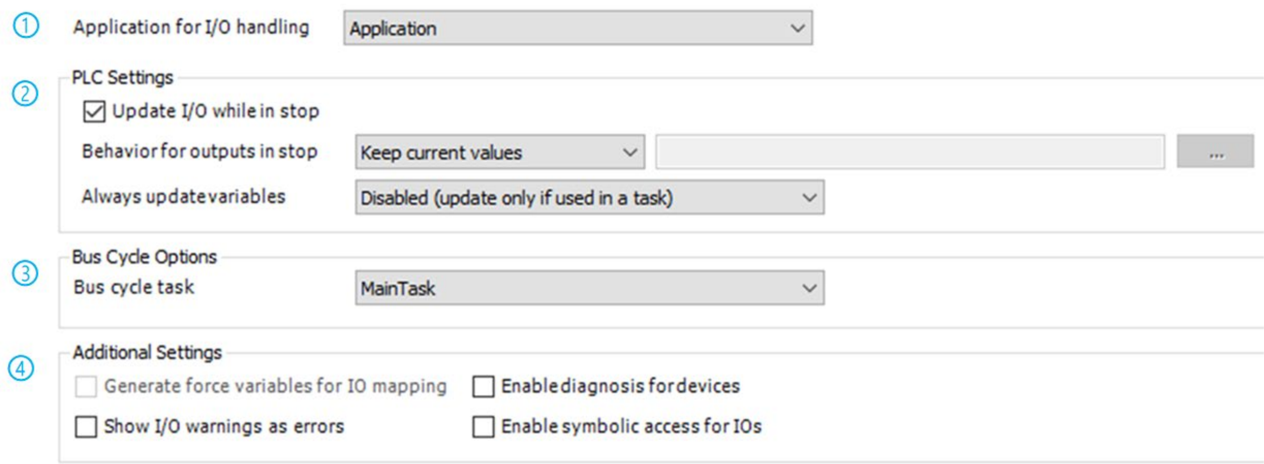
- **Log entries from data sources**



Item	Description
UTC time	☐ : Standard setting; the time stamp is converted into the local time on the computer as indicated by the time zone of the operating system. ☑ : The time stamp of the runtime system is displayed.
Severity	Five categories based on the severity of the event: ■ : Information ■ : Warning ■ : Error ■ : Exception ■ : Debugging You can show or hide each category by clicking the corresponding buttons in the bar. The number of log entries of the category concerned is shown on each button.
Time stamp	Date and time (e.g. 03.05.2023 20:10:31)
Description	Description of the event
Components	The library names used at runtime, e.g. CmpApp
Drop-down list with component names	To display the events related to the selected components
	To refresh the log list
	To export the list contents to an XML file
	To import an XML file to the list (Log)

4.2.1.6 PLC Settings

The basic settings for PLC configuration can be completed here. For instance, settings for I/O handling and bus cycle task.



① **Application for I/O handling**

Item	Description
Application for I/O handling	Application for managing I/O operation

② **PLC Settings**

Item	Description
Update I/O while in stop	☐ : When PLC is in stop, DIADesigner-AX does NOT refresh the values of I/O channels. ☑ : When PLC is in stop, DIADesigner-AX still refreshes the values of I/O channels. The outputs are set to the predefined default values if the watchdog detects a malfunction.
Behavior for outputs in stop	The way to deal with the output channels when the controller is in stop: ■ **Keep current values**: the current values are retained. ■ **Set all outputs to default**: the default values resulting from the I/O mapping are assigned. ■ **Execute program**: you can control the output values via a program contained in the project, which DIADesigner-AX executes in the stopped state. Enter the program name in the field on the right side.
Always update variables	Global setting, which defines whether or not DIADesigner-AX updates the I/O variables in the bus cycle task. This setting is effective for I/O variables of the slaves and modules only when their update settings are defined as **Disabled**.

Item	Description
	■ **Disabled (update only if used in a task)**: DIADesigner-AX updates the I/O variables only if they are used in a task. ■ **Enabled 1 (use bus cycle task if not used in any task)**: DIADesigner-AX updates the I/O variables in the bus cycle task if they are not applied in any other tasks. ■ **Enabled 2 (always in bus cycle task)**: DIADesigner-AX updates all I/O variables in each cycle of the bus cycle task, regardless of whether they are used or whether they are mapped to an input or output channel.

③ **Bus Cycle Options**

Item	Description
Bus cycle task[*1]	Task that manages bus cycle. By default, it enters the task cycle by the device-defined options.

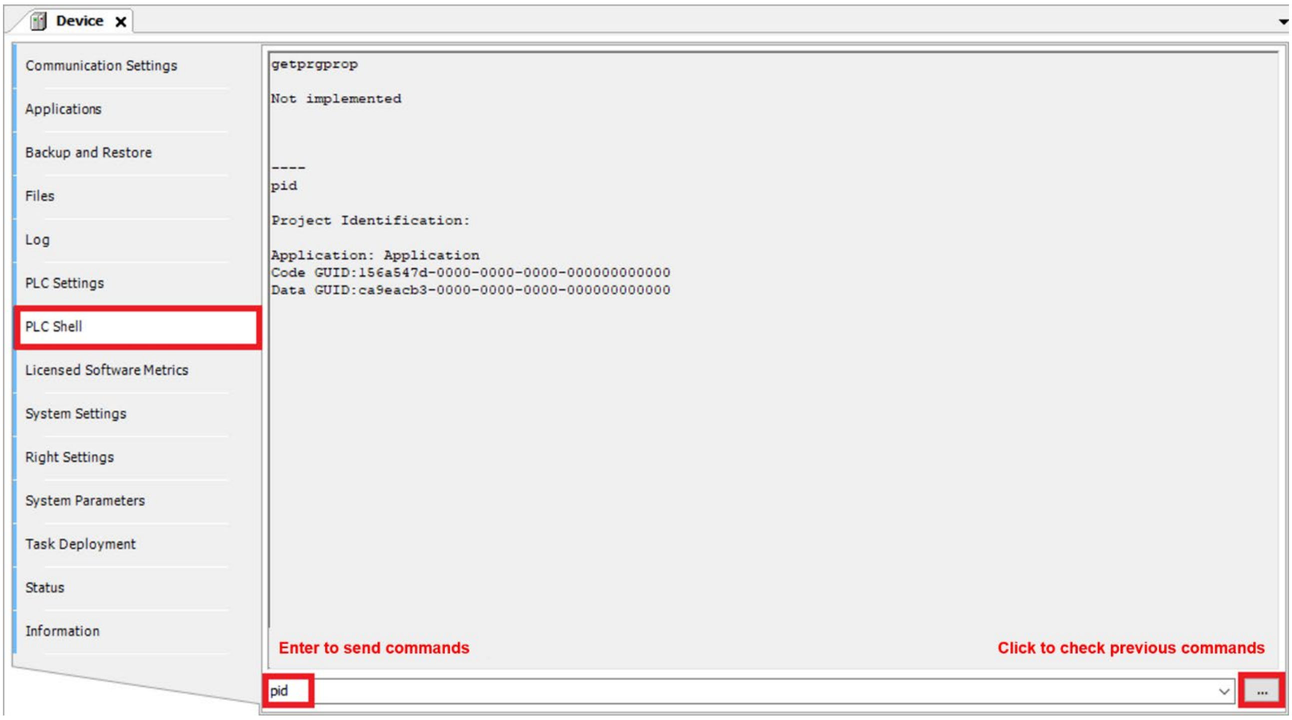
*1: Before you select **<unspecified>** for the bus cycle task, be aware that the default values given in the device description would take effect. You should therefore check this description. You can define the task with the shortest cycle time as default, but you can equally choose the task with the longest cycle time as default.

④ **Additional Settings**

Item	Description
Generate force variables for IO mapping	The device does NOT support this function.
Enable diagnosis for devices	☑ : DIADesigner-AX automatically integrates the library CAA Device Diagnosis into the project and creates a hidden function block for each device. If there is already a function block for the device, then either an extended function block is used (e.g. with EtherCAT) or a further function block instance is added. It also contains the general implementation of device diagnosis.
Show I/O warnings as errors	Warnings concerning the I/O configuration are displayed as errors.
Enable symbolic access for IOs	Symbolic access is activated for I/O.

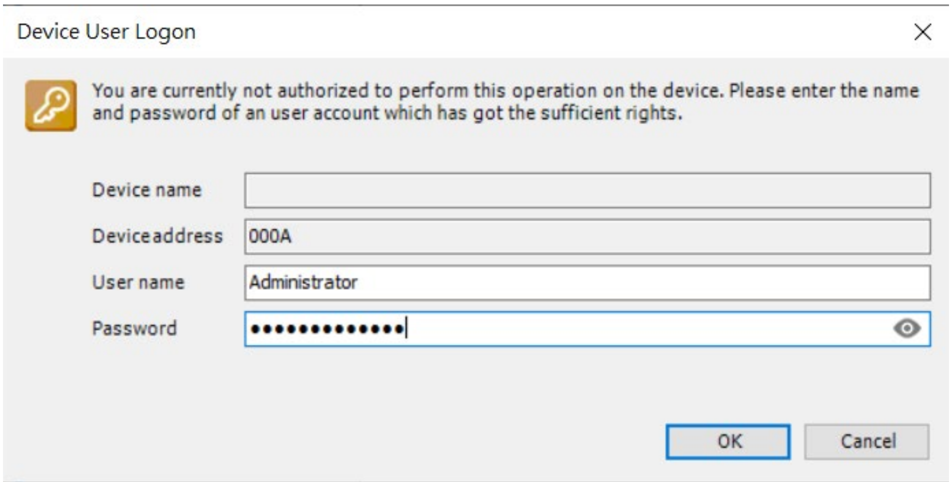
4.2.1.7 PLC Shell

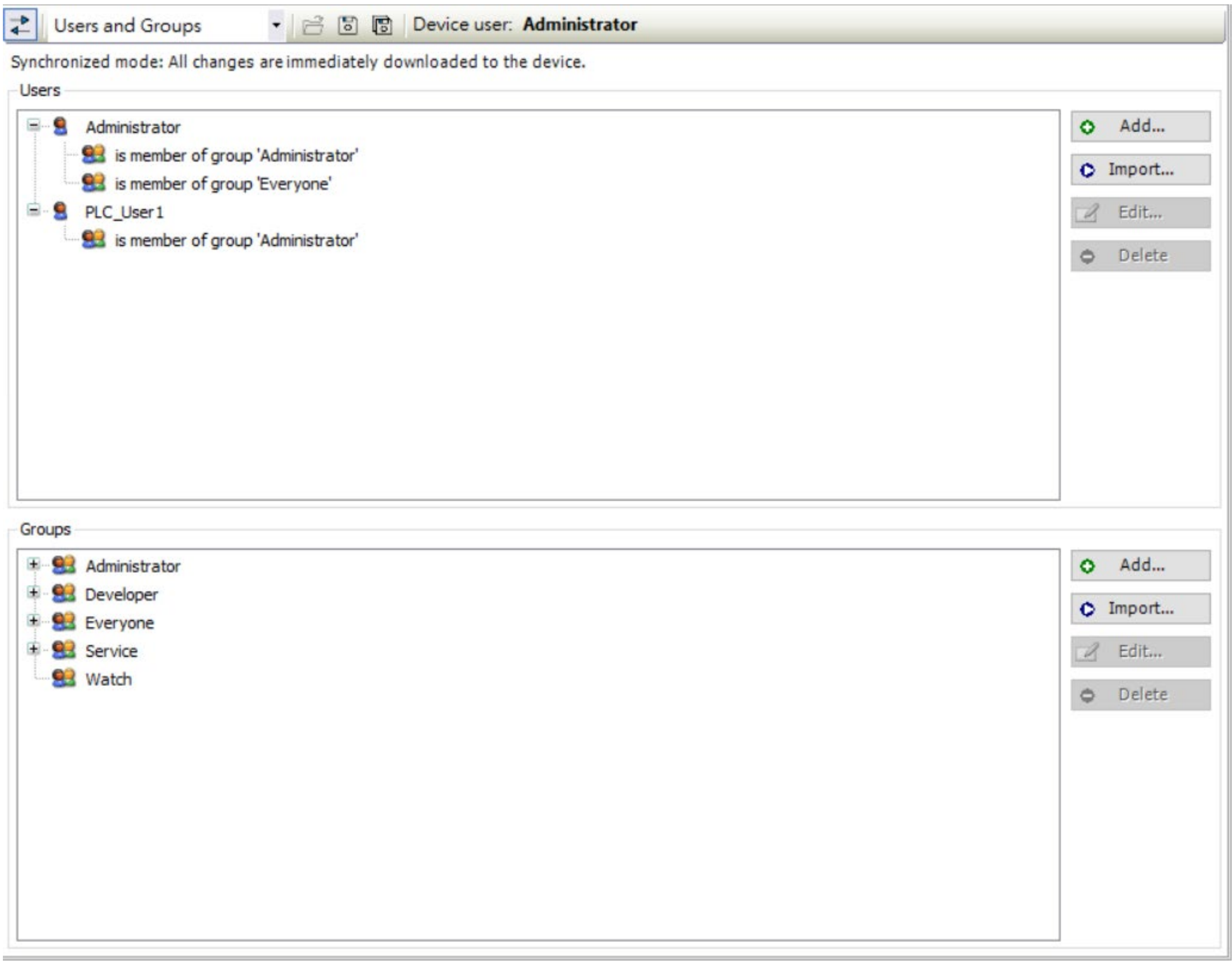
Users can use this text-based control monitor for accessing specific information from the controller. Specify device-dependent commands for this and receive responses from the controller in a result window.



4.2.1.8 User and Groups

User accounts and groups can be edited here to manage the access to objects and files at runtime. Use the default account (Administrator) and password (Administrator) at first logon and change your password afterwards.





- **Toolbar of the tab**

Item	Description
Synchronization	To enable and disable the synchronization function between the editor and the user management on the device. ■ If the button is not pressed, then the editor is blank or it displays a configuration that is loaded from the hard disk. ■ If the button is pressed, then DIADesigner-AX synchronizes the display in the editor continuously with the current user management on the connected device. ■ If the synchronization is activated while the editor contains a user configuration that is not synchronized with the device yet, then a prompt message shall appear with the following options: - **Upload from the device and overwrite the editor content**: the configuration on the device is loaded into the editor, overwriting the current contents. - **Download the editor content to the device and overwrite the user management there**: the configuration in the editor is transferred to the device and applied.
Import from disk	To import a user management configuration from the file.
Export to disk	To save the current configuration of user management as an XML file.
Export all to disk	To save all user management configurations to the file.
Device user	User name of the user currently logged in.

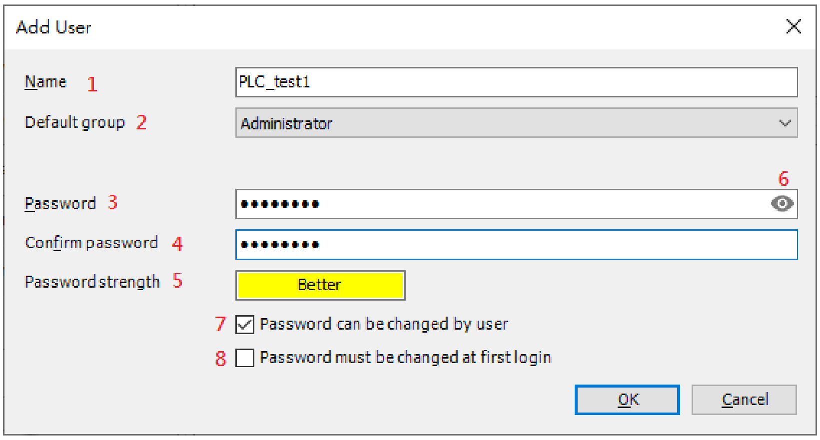
- **Users**

Item	Description
Add…	To create a new user account*1
Import…	To select the desired entries and import users into the device user management*2
Edit…	To change the settings of the selected user account
Delete	To delete the selected user account

- **Groups**

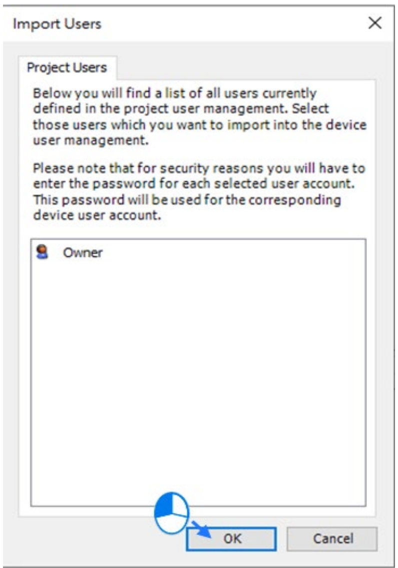
Item	Description
Add…	To create a new user group*3
Import…	To select the desired entries to import groups into the device user management*4
Edit…	To change the settings of the selected group
Delete	To delete the selected group

*1: The **Add User** setting page: click **OK** after finishing the setting.

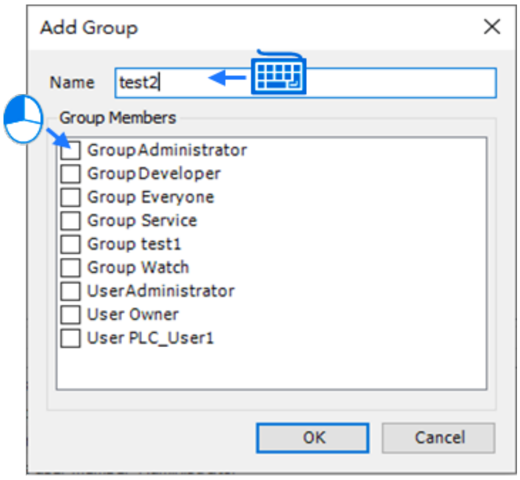


	Item	Description
1	**Name**	Define the user name.
2	**Default group**	Select the default group.
3	**Password**	Set the password.
4	**Confirm password**	Enter the password again to confirm the password.
5	**Password strength**	Security levels; from **Very weak** to **Very good.**
6	**Hide password**	: The password is shown with asterisks * when it is typed in. : Click to reveal the password.
7	**Password can be changed by user**	: Password can be changed by the user.
8	**Password must be changed at first login**	: Password must be changed at first login.

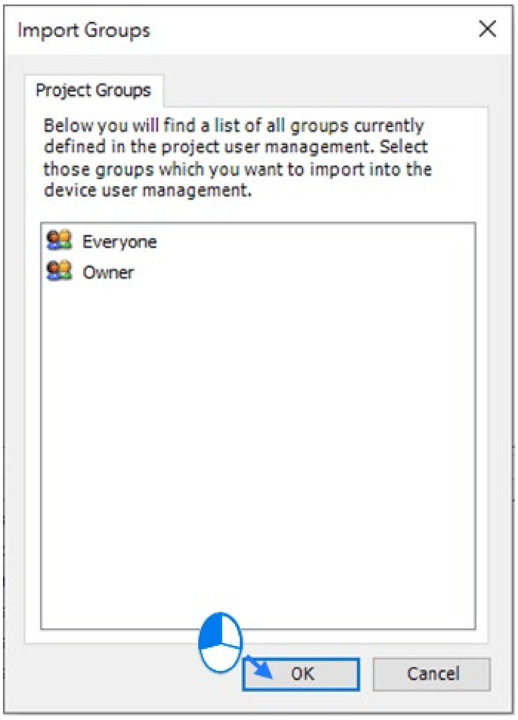
*2: The **Import Users** setting page: click **OK** after selecting the user from the list.



*3: The **Add Group** setting page: enter the new group name and select the group members that are to be added for this new group. Then click **OK**.



*4: The **Import Groups** setting page: click **OK** to import after selecting the group from the list.



4.2.1.9 Access Rights

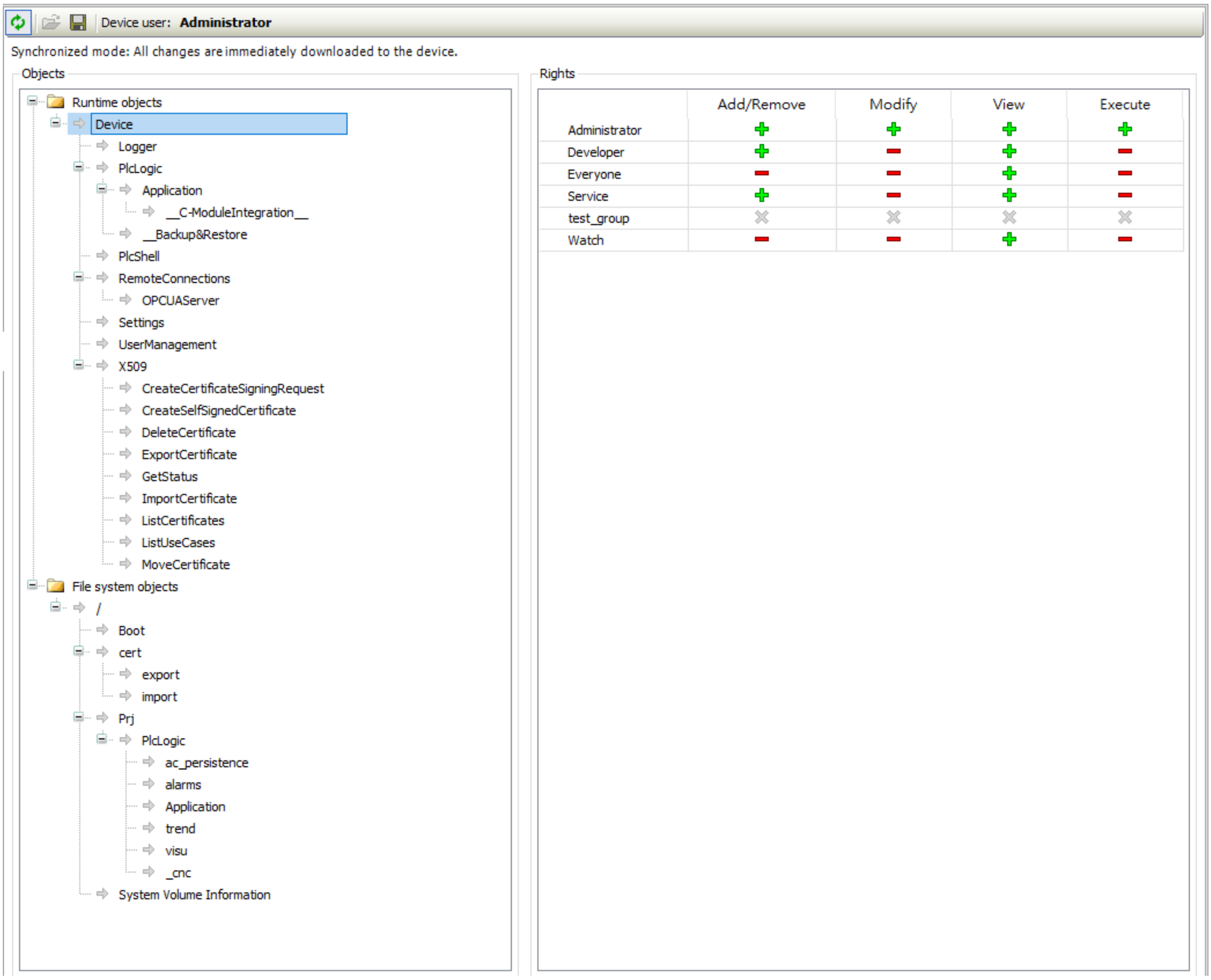
Here you can define the access rights to the AX-8 Series PLC. As in the previous section **Users and Groups**, the users must be members of at least one user group; and, only user groups can be granted certain access rights.

Requirements for displaying the Access Rights page:

- In the DIADesigner-AX options, the Device editor category must be selected to display the Access Rights page. Note that this DIADesigner-AX option can be overwritten by the device description.

The access rights are granted to user groups instead of users:

- The primary requirement is that user management component should be available on the controller.
- Users and user groups are configured in **Users and Groups**.



- **Toolbar of the tab**

Item	Description
Synchronization	To enable and disable the synchronization between the editor and the user management on the device. ■ If the button is not pressed, then the editor is blank or it displays a configuration that is loaded from the hard disk. ■ If the button is pressed, then DIADesigner-AX synchronizes the display in the editor continuously with the current user management on the connected device.
Import from disk	N/A
Export to disk	To save the current user management configuration as an XML file.
Device user	User name of the user currently logged in.

- **Objects**

Description
In the tree structure, the objects to which actions can be executed at runtime are listed and sorted in the respective groups. In the **Rights** view, you can configure the access options of the user group for the selected object.
Object source (root node): ■ **Runtime objects > /**: in these objects, the rights can be granted to folders of the current execution directory. ■ **File system objects > Device**: in these objects, those with online access are managed and therefore the access rights should be controlled. Refer to Overview of the objects for the descriptions of the objects.
Object groups and objects (indented): E.g. under **Device**, there are child nodes such as **Logger, PlcLogic, Settings, UserManagement**.

- **Rights**

Description
In general, the access rights of the subobjects are inherited from the root object (also Device or /). It implies that any changes of the access rights of a user group to a parent object also apply to its child objects. The table applies to the object that is currently selected in the tree structure. It shows the rights currently configured for each user group on this object.

Description

Applicable actions on the object:

- Add/Remove
- Modify
- View
- Execute

When an object is selected, the table on the right side shows the access rights of the available user groups for the selected object.

The table provides a quick overview of:

- Which access rights are evaluated by an object
- Which user group is assigned with which kind of rights for the object

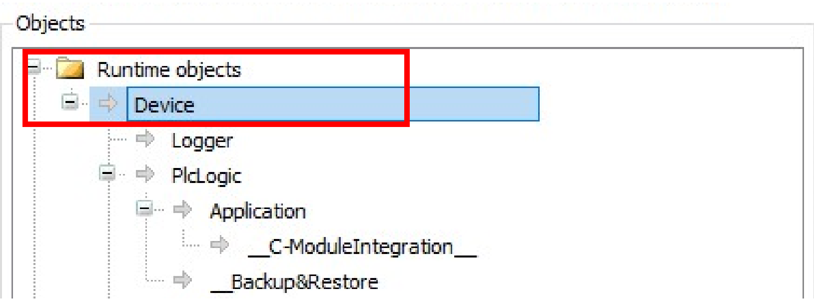
Meanings of the symbols:

- : access right granted explicitly
- : access right denied explicitly
- : access right granted through inheritance
- : access right denied through inheritance
- : access right not granted or denied explicitly or inherited from the parent object. Inaccessible.
- No symbol: multiple objects with different access rights are selected.

Change the access rights by clicking the symbols for once.

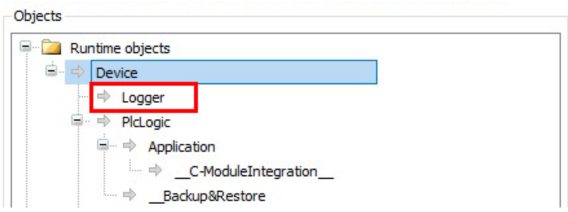
- **Overview of the objects**

  - Runtime objects > Device

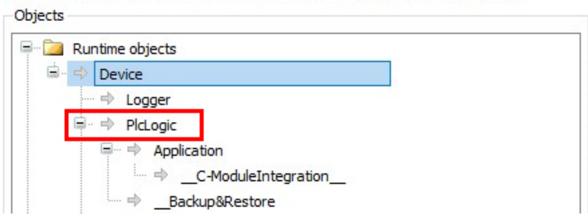


- Logger:

  Logger in the Access Rights tab is read-only as it is used online. Therefore, the only right that can be granted is the View action.

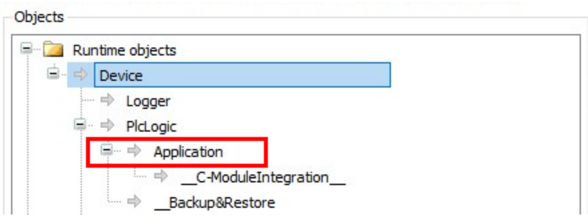


- PlcLogic:



  All IEC applications are inserted here automatically as subobjects during download. This allows specific control of online access for application. Access rights can be assigned centrally over all applications in the PlcLogic. The Administrator and Developer user groups have full access to the IEC applications while the Service and Watch user groups only have read access (e.g. read-only value monitoring).

- Application:



  The following table lists the actions affected when specific access rights are granted for an IEC application.

  x : access not granted

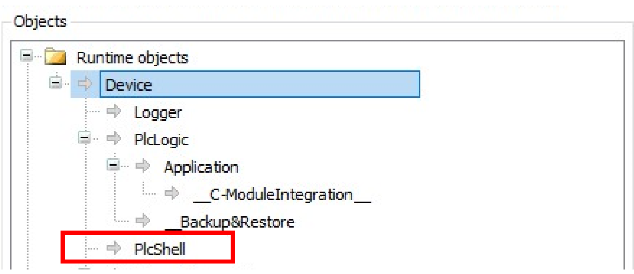
  ● : access granted

	Operation	**Access rights**			
		Add/Remove	**Execute**	**Modify**	**View**
Application	Login	●	●	●	x
	Create	x	●	●	●
	Create child object	x	●	●	●
	Delete	x	●	●	●
	Download / online change	x	●	●	●

	Create boot application	x	●	●	●
	Read variable	●	●	●	x
	Write variable	●	●	x	x
	Force variable	●	●	x	x
	Set and delete breakpoint	●	x	x	●
	Set next statement	●	x	x	●
	Read call stack	●	●	●	x
	Single cycle	●	x	●	●
	Switch on flow control	●	x	x	●
	Start / Stop	●	x	●	●
	Reset	●	x	●	●
	Restore retain variables	●	x	●	●
	Save retain variables	●	●	●	x

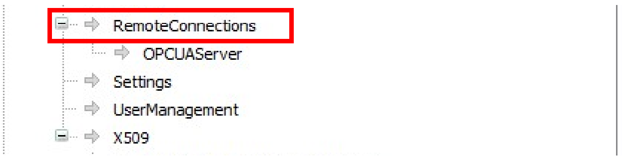
- PlcShell:

  Only Modify is granted here. This means that only when the Modify right has been granted to a user group can the PLC shell commands be used.



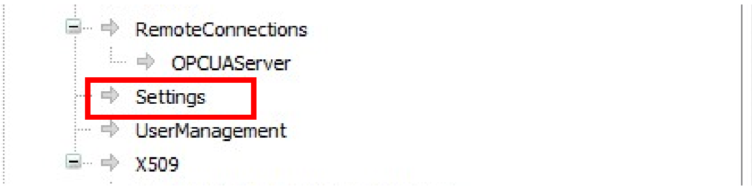
- RemoteConnections:

  Additional external connections to the AX-8 Series PLC can be configured under this node. In general, access to OPC UA server can be configured here.



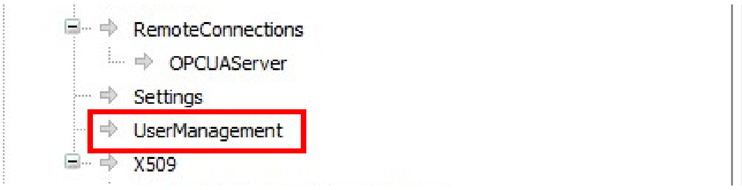
- Settings:

  This is an online access to the configuration settings of the AX-8 Series PLC. By default, access to Modify is granted only to the administrator.

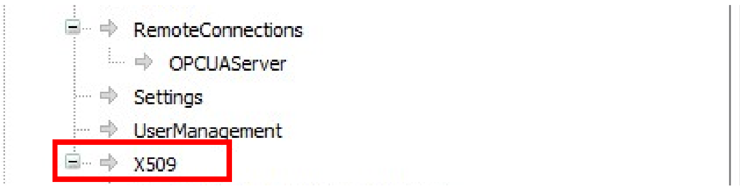


- UserManagement:

  This is an online access to the user management of AX-8 Series PLC. By default, access to read / write is granted only to the administrator.



- X509:



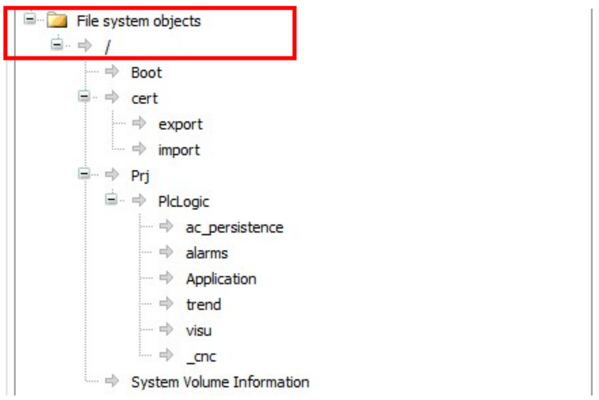
  This manages the online access of the X.509 certificate. There are two types of accesses:

  1. Read (View)
  2. Write (Modify)

  Each operation is assigned one of these two access rights and is inserted as the child object under X509. Therefore, access of each operation can be further fine-tuned.

■ File system objects > /

All folders in the execution path of the AX-8 Series PLC are inserted under thefile system object /. This allows you to grant specific rights to each folder of the file system.

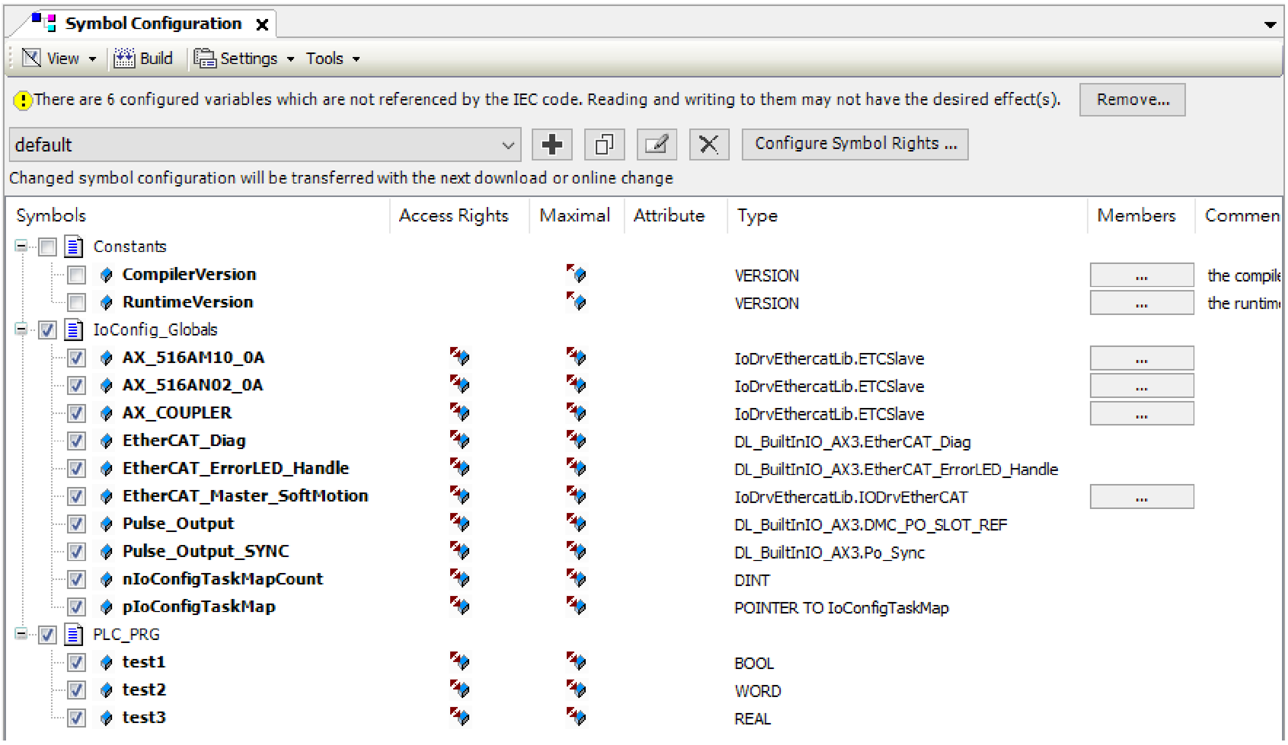


4.2.1.10 Symbol Rights

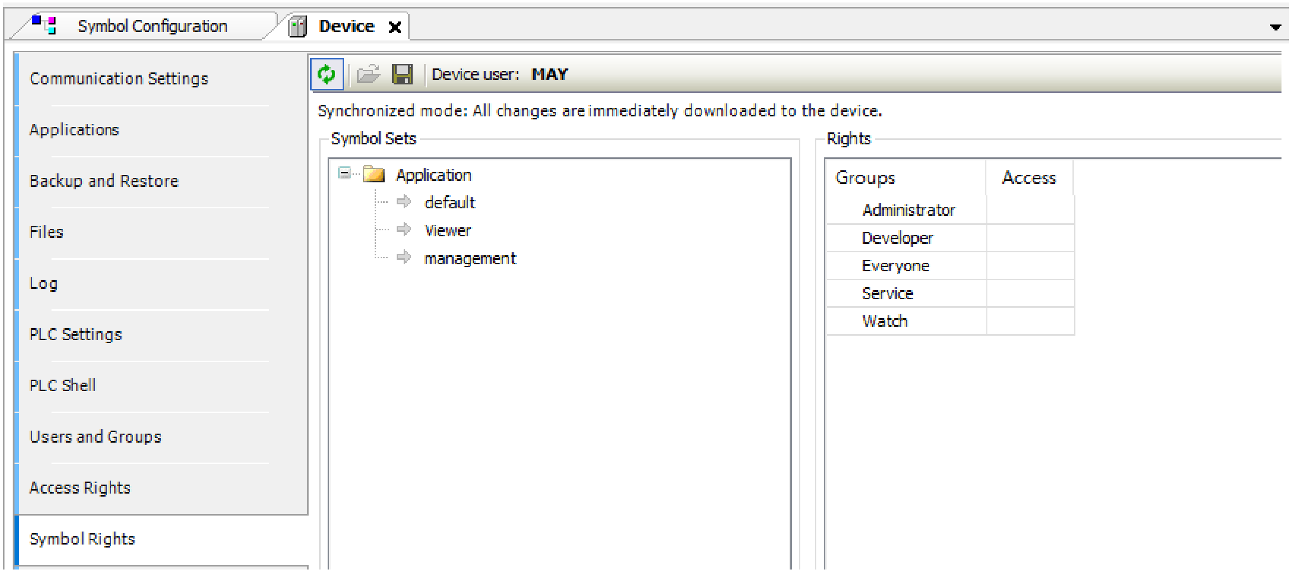
Here you can define the access rights of different user groups to individual symbol sets available on the AX-8 Series PLC.

Requirements:

1. User management must be set up on the controller.
2. Download a project to AX-8 Series PLC and make sure Symbol Sets are defined in DIADesigner-AX Object.
3. User groups have the data access right for logging on to the AX-8 Series PLC.



In the **Symbol Sets** view, all symbol sets are listed under Application whose definition and applications are downloaded to the AX-8 Series PLC altogether.

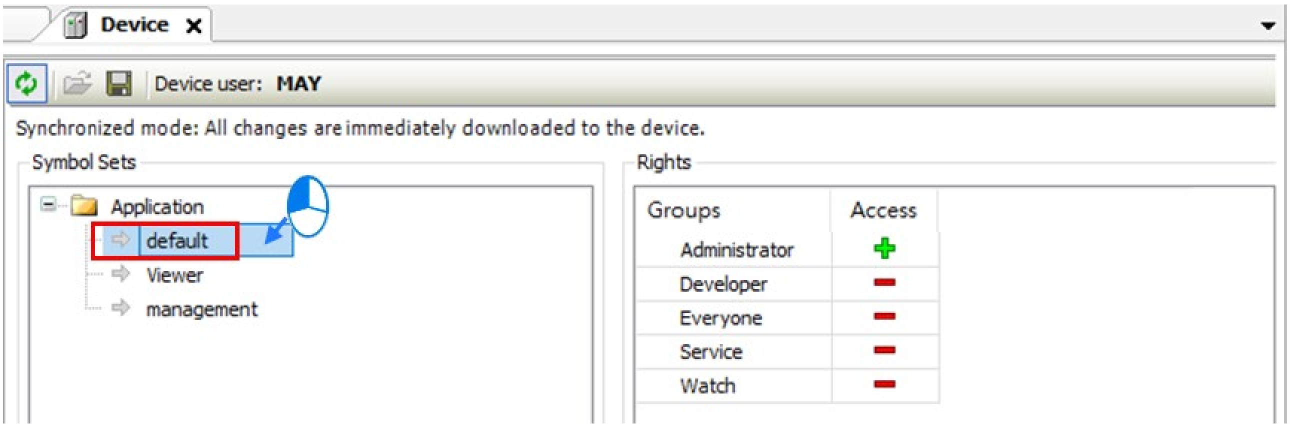


In the **Rights** view, it lists the user groups defined in the user management of the controller. Select a symbol set to check the corresponding access rights.

: access granted

: inaccessible not granted

Switch between and by double-clicking the symbol.



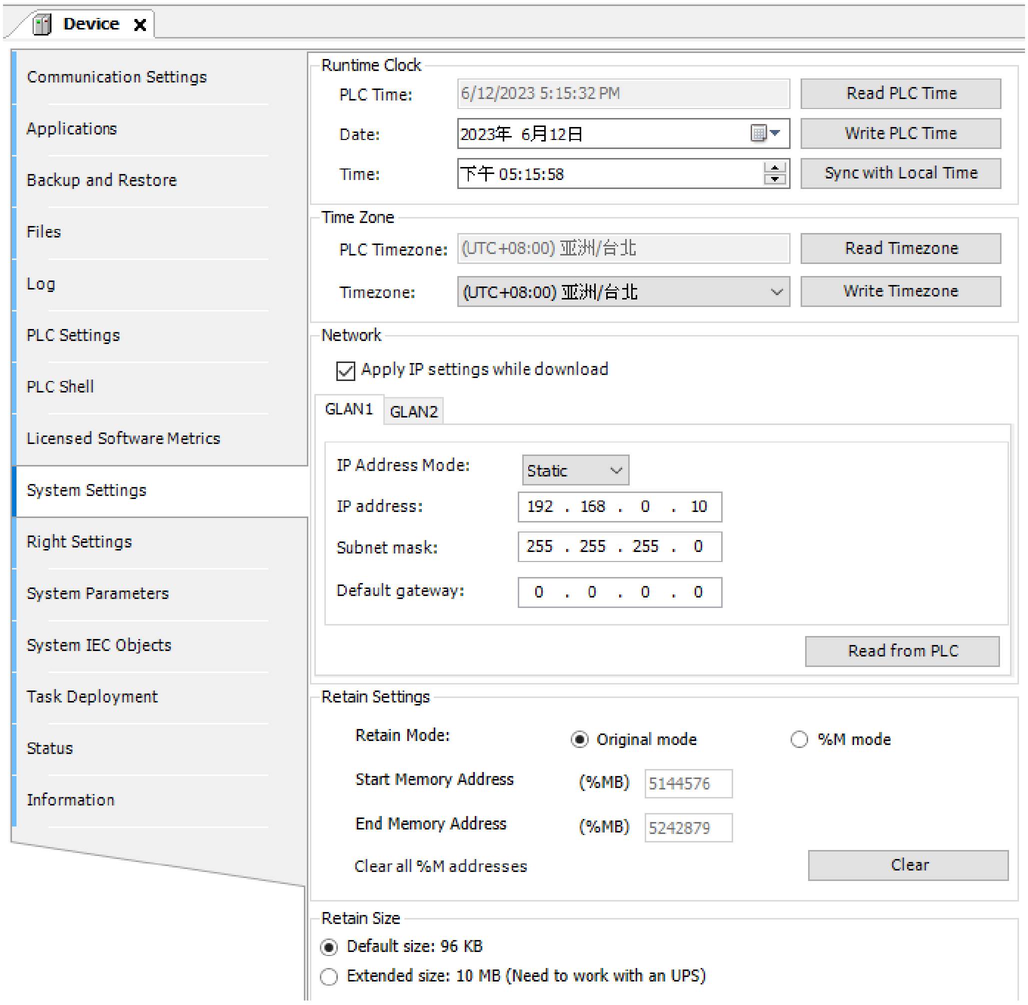
Click to save the current configuration to an XML file. The file type is Device symbol management files (*.dsm). Click to read the file of the same type from the computer.

4.2.1.11 System Settings

Here you can set up the system settings[*1] for the AX-8 Series PLC. Before setting up, make sure that DIADesigner-AX is successfully connected to the controller. Refer to **section 4.2.1.1** for establishing the connection.

*1: The tab is called **Runtime Clock Configuration** in DIADesigner-AX V1.0.0 while is modified as **System Settings** in V1.1.0 as new function (IP Address setup) is added.

Go to **Device** > **System Settings**.



Item	Description
Runtime Clock	■ PLC Time: current date and time on the controller. Click **Read PLC Time** to update. ■ Date: click **Write PLC Time** to write the date on DIADesigner-AX (PC) into the controller. ■ Time: click **Sync with Local Time** to write the time on DIADesigner-AX (PC) into the controller.
Time Zone	■ PLC Timezone: click **Read Timezone** to update the current time zone of the controller. ■ Timezone: click **Write Timezone** to write the time zone on DIADesigner-AX (PC) into the controller.
Network[*1]	■ ☑ Apply IP settings while download: the IP of the controller is updated as the configured IP address while download. ■ GLAN 1 & GLAN 2: select to switch to the page you are looking for.

Item	Description
	■ Enable Gateway and DNS Setting: the settings for Default gateway, Preferred DNS server and Alternate DNS server are available. ■ IP Address Mode: static IP or DHCP. ■ IP Setting: IP address / Subnet mask / Default gateway can be set by users. ■ DNS Setting: users can select **Obtain DNS server address automatically** or define your own DNS server addresses. ■ Read from PLC: read and update the IP address from the active controller.
Project ID[*2]	If Project ID status is active, the setting must be completed before downloading projects. Click **Set Project ID** to activate the feature.
Retain Settings	■ **Original mode**: keep original retain setting of AX-8 Series PLC. ■ **%M mode**: set up the %M address by giving the values of Start Memory Address and End Memory Address. ■ **Clear**: clear all the %M addresses assigned to the variables.

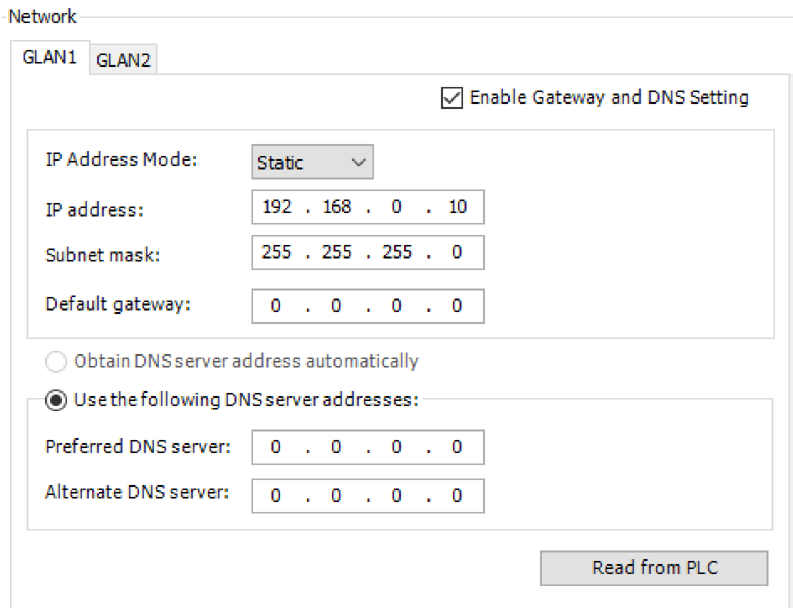
*1:

Network parameter settings are available only when the DDF on AX-8 Series PLC is V1.0.1.0 or above.

AX-8 Series PLC support two Ethernet ports and the settings are illustrated as follows:

Take GLAN 1 for instance, when **Enable Gateway and DNS Setting** is checked, Gateway and DNS are configured based on the value set on this page. Meanwhile, the configuration is not available on the other tab, which is GLAN 2.

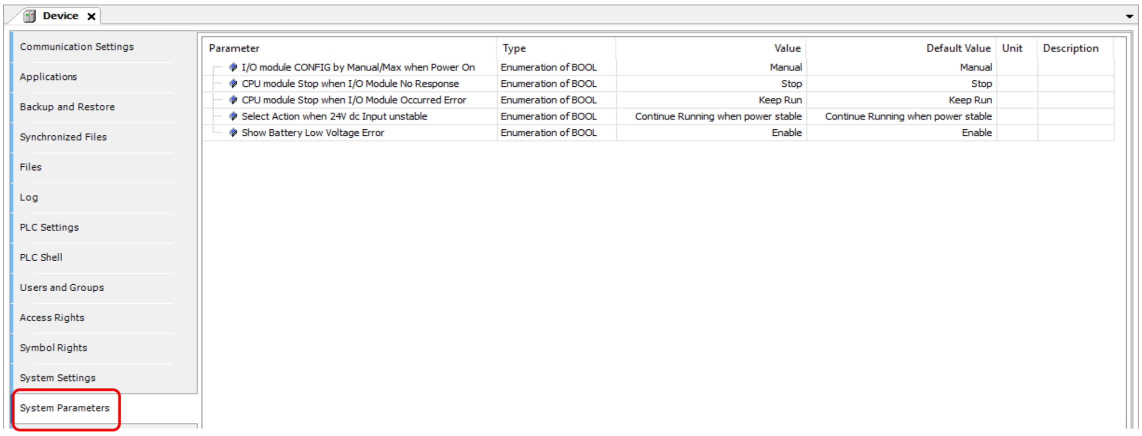
Conversely, if **Enable Gateway and DNS Setting** is left unchecked, the fields below are greyed out and configuration is unavailable.



*2: Refer to **Chapter 8.2 of DIADesigner-AX Software Manual** for Project ID operation.

4.2.1.12 System Parameters

You can set up the parameters for the controller in the **System Parameters** tab but must note that editing during online monitoring is not supported.



1. I/O module CONFIG by Manual/Max when Power On: to set the configuration of I/O modules.
   - Manual (default): the actual module placement should be consistent with the configuration set in HWCONFIG for the PLC to run normally.
   - Max: set the maximum number for the module placement. An alarm pops up if the actual I/O module placement is larger than the maximum number defined.

2. CPU module Stop when I/O Module No Response: to set whether the CPU and other modules which function normally will continue to run or not when there is any extension module out of control or disconnected with no response.
   - Stop (default): the CPU module stops and reports error. Power-on reset is required for the CPU module to again function normally.
   - Keep RUN: the CPU module and other normal modules keep running.

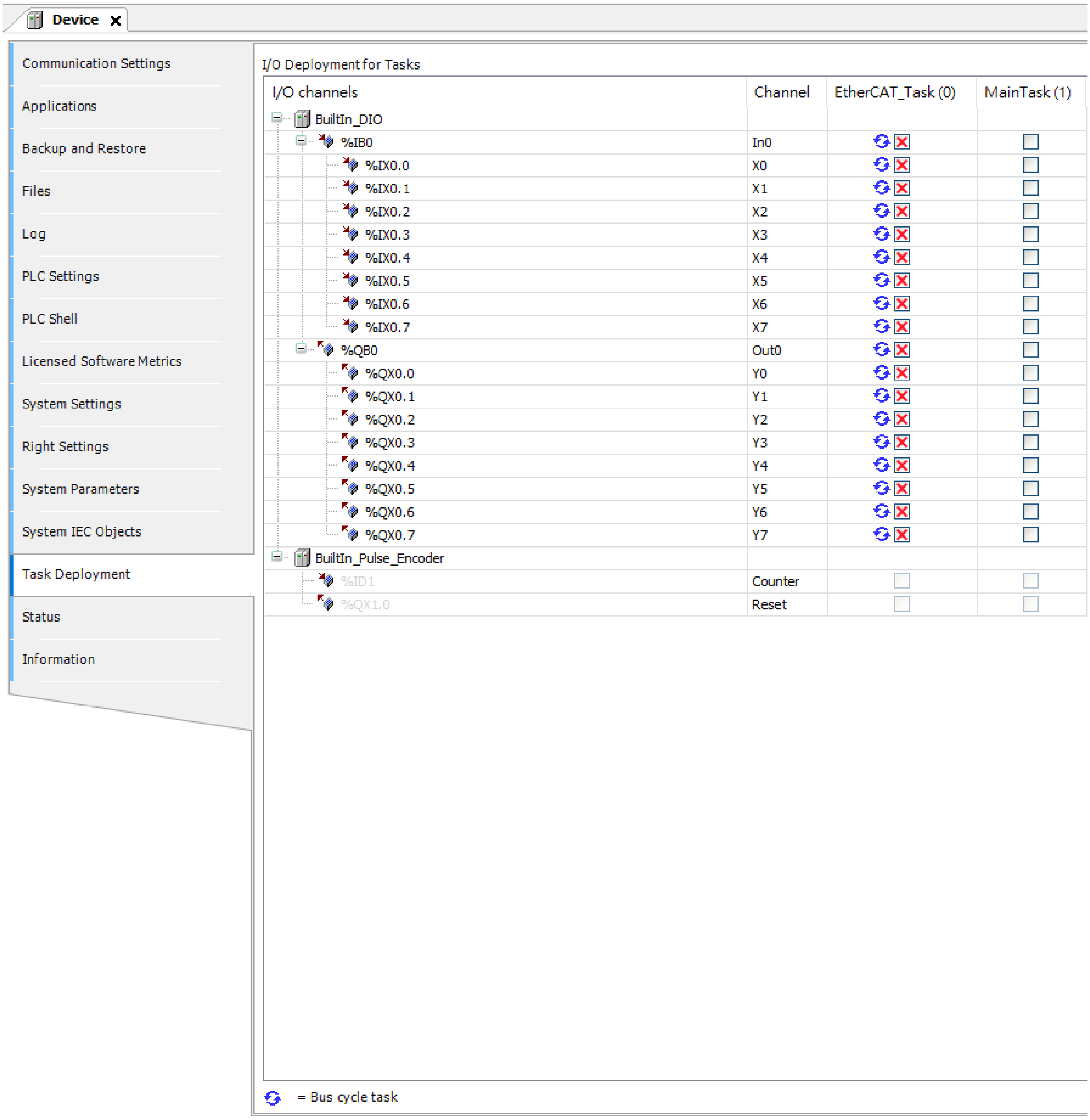
3. CPU module Stop when I/O Module Occurred Error: to set the solution when minor errors occur in extension modules.
   - Stop: the CPU module stops and reports an error. The CPU module is restarted after the error is cleared.
   - Keep RUN (default): the CPU module keeps running with the warning messages recorded.

4. Select Action when 24Vdc Input unstable: to set the solution to power failure for 24Vdc input.
   - Continue Running when power stable (default): the CPU stops automatically till the power supply is stable to run again.
   - Into Error Status: the CPU stops with ERROR LED blinking; the CPU stays in STOP even after the power supply becomes stable again.

5. Show Battery Low Voltage Error: to set whether to show an alarm or not when the lithium battery is of low voltage or not installed.
   - Disable: there is no alarm when the lithium battery is of low voltage or not installed.
   - Enable (default): an alarm is displayed when the lithium battery is of low voltage or not installed

4.2.1.13 Task Deployment

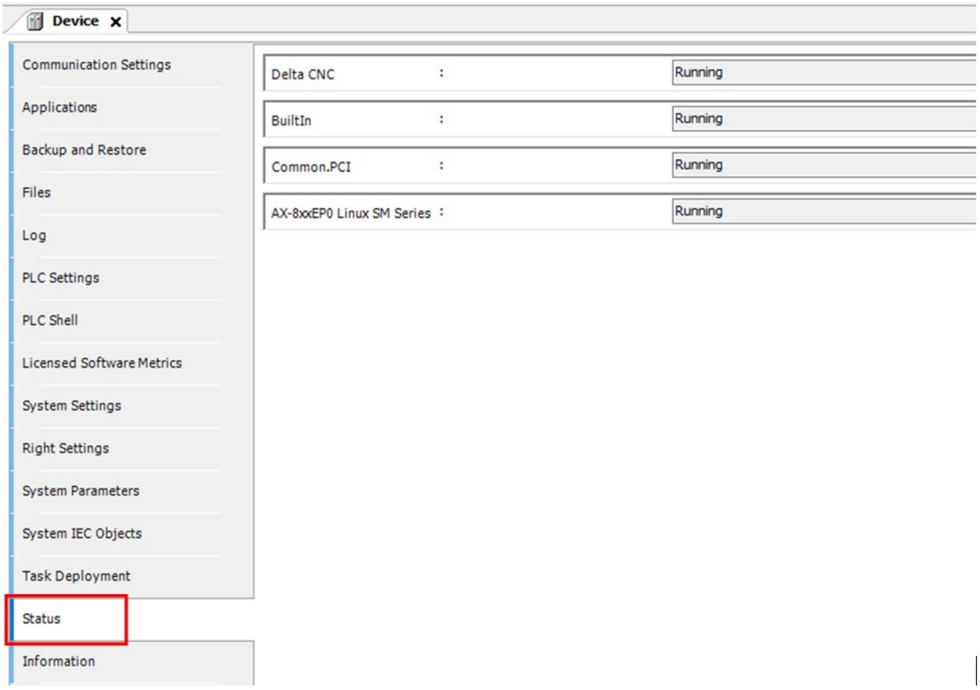
In the following table it displays the inputs / outputs and their assignments to the defined tasks as well as bus cycle task. The information can be used for troubleshooting when the execution is below expectations. The contents are refreshed only after the project is compiled and downloaded to the CPU.



	The task is defined as Bus cycle task in the PLC settings.
	Inputs or outputs that are written or read by the task.

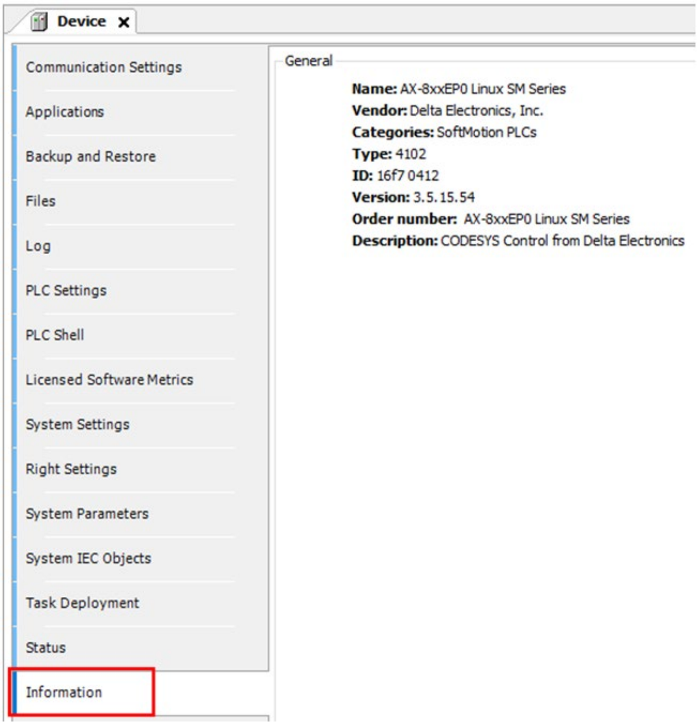
4.2.1.14 Status

Information about the status of the controller can be found here. For example, **Running**, **Stopped**, and other specific diagnostic messages from the respective device. It also includes information about card used and internal bus system.



4.2.1.15 Information

Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and so on.



4.3 Data Type and Variables

4.3.1 Data Type

Data Type	Minimum Value	Maximum Value	Data Width
BOOL	FALSE	TRUE	1-bit
BYTE	0	255	8-bit
WORD	0	65535	16-bit
DWORD	0	4294967295	32-bit
LWORD	0	2^{64}-1	64-bit
SINT	-128	127	8-bit
USINT	0	255	8-bit
INT	-32768	32767	16-bit
UINT	0	65535	16-bit
DINT	-2147483648	2147483647	32-bit
UDINT	0	4294967295	32-bit
LINT	-2^{63}	2^{63}-1	64-bit
ULINT	0	2^{64}-1	64-bit
REAL	-3.402823E+38	3.402823E+38	32-bit
LREAL	-1.7976931348623157E+308	1.7976931348623157E+308	64-bit
TIME	T#0ms	T#49d17h2m47s295ms	32-bit
LTIME	LTIME#0ns	LTIME#213503d23h34n33s 709ms551us615ns	64-bit
TIME_OF_DAY(TOD)	TOD#00：00：00.000	TOD#23：59：59.999	32-bit
DATE	D#1970-1-1 (01/01/70)	DATE#2106-2-7 (February 07，2106)	32-bit
DATE_AND_TIME	DT#1979-1-1-00：00：00 (01/01/1970 00：00：00)	DT#2106-2-7-6：28：15 (February 07，2106 6：28：15)	32-bit
STRING	ASCII format (8-bit): up to 255 characters		
WSTRING	Unicode format (16-bit): no limit on the length		

4.3.2 Variables

Rules for variable identifiers:

- No spaces or special characters.
- Not case-sensitive (e.g. Var0 and VAR0 are deemed as the same variable).
- No consecutive underscore characters (e.g. b_Var0 is acceptable but b__Var0 is not permitted).

Rules for multiple use of identifier:

- Local variable cannot be declared more than once.
- Local variables have priority over global variables. If they share the same name, the local variable is given precedence within the POU than the global variable.
- Variables with the same name can be declared in different global variable list (e.g. globe_list1.bvar and globe_list2.bvar can co-exist in two different global variable lists).

Comments:

- Single comment: the symbol **//** indicates a single comment. For example: // Variable Define.
- Multiple comments: the symbol **(* XX : XX *)** indicates multiple comments from XX to XX. For example, (*Variable Define : Variable Define*).

4.3.2.1 Declaration of Variables

In DIADesigner-AX projects, variables are declared using the following methods.

Syntax: <Variable Name> : <Data Type> := <Initialization> ;

Example:

```
VAR
    bVar     :   BOOL ;
    byVar    :   BYTE := 1 ;
    wVar     :   WORD := 16#0001 ;
    todVar   :   TOD  := TOD#02:30:15.100;
END_VAR
```

Array

Syntax: <Variable Name>：ARRAY[0..N] OF <Data Type>

Example:

```
VAR
    byVar_Array :   ARRAY[0..10] OF BYTE ;
    wVar_Array  :   ARRAY[0..30] OF WORD ;
    rVar_Array  :   ARRAY[0..50] OF REAL ;
END_VAR
```

4.3.2.2 Address Assignment

In AX-8 Series PLC, memory areas are divided into three ranges: I (input memory range), Q (output memory range) and M (flag memory range). You can use specific character strings to express memory position and size. There are two modes for non-volatile memory on AX-8 Series PLC, which can be configured in CPU parameter settings. When set to Mdev mode, the value of %M is retained; if not, it is the value of Retain & Persistent being retained.

Syntax: %<Memory Area Prefix><Size Prefix><Memory Position>

Memory Area	Description	Range
I	Input Memory Range	128 KB
Q	Output Memory Range	128 KB
M	Flag Memory Range	5 MB
--	Non-volatile Memory Size	96 KB
Size Prefix	**Data Type**	**Data Width**
X	--	1-bit
B	Byte	8-bit
W	Word	16-bit
D	DWord	32-bit
L	LWord	64-bit

- **Memory Area**

The numbering used for addressing the memory position depends on the target system. Before specifying the address value in the memory area, you need to know the corresponding memory-mapped position to prevent overlap of memory ranges. See the table below for reference.

<table>
<tr><th colspan="8">Memory Area</th></tr>
<tr><td>X0.63~X0.56</td><td>X0.55~X0.48</td><td>X0.47~X0.40</td><td>X0.39~X0.32</td><td>X0.31~X0.17</td><td>X0.23~X0.16</td><td>X0.15~X0.8</td><td>X0.7~X0.0</td></tr>
<tr><td>X7.7~X7.0</td><td>X6.7~X6.0</td><td>X5.7~X5.0</td><td>X4.7~X4.0</td><td>X3.7~X3.0</td><td>X2.7~X2.0</td><td>X1.7~X1.0</td><td>X0.7~X0.0</td></tr>
<tr><td>B7</td><td>B6</td><td>B5</td><td>B4</td><td>B3</td><td>B2</td><td>B1</td><td>B0</td></tr>
<tr><td colspan="2">W3</td><td colspan="2">W2</td><td colspan="2">W1</td><td colspan="2">W0</td></tr>
<tr><td colspan="4">D1</td><td colspan="4">D0</td></tr>
<tr><td colspan="8">L0</td></tr>
</table>

- **Example**

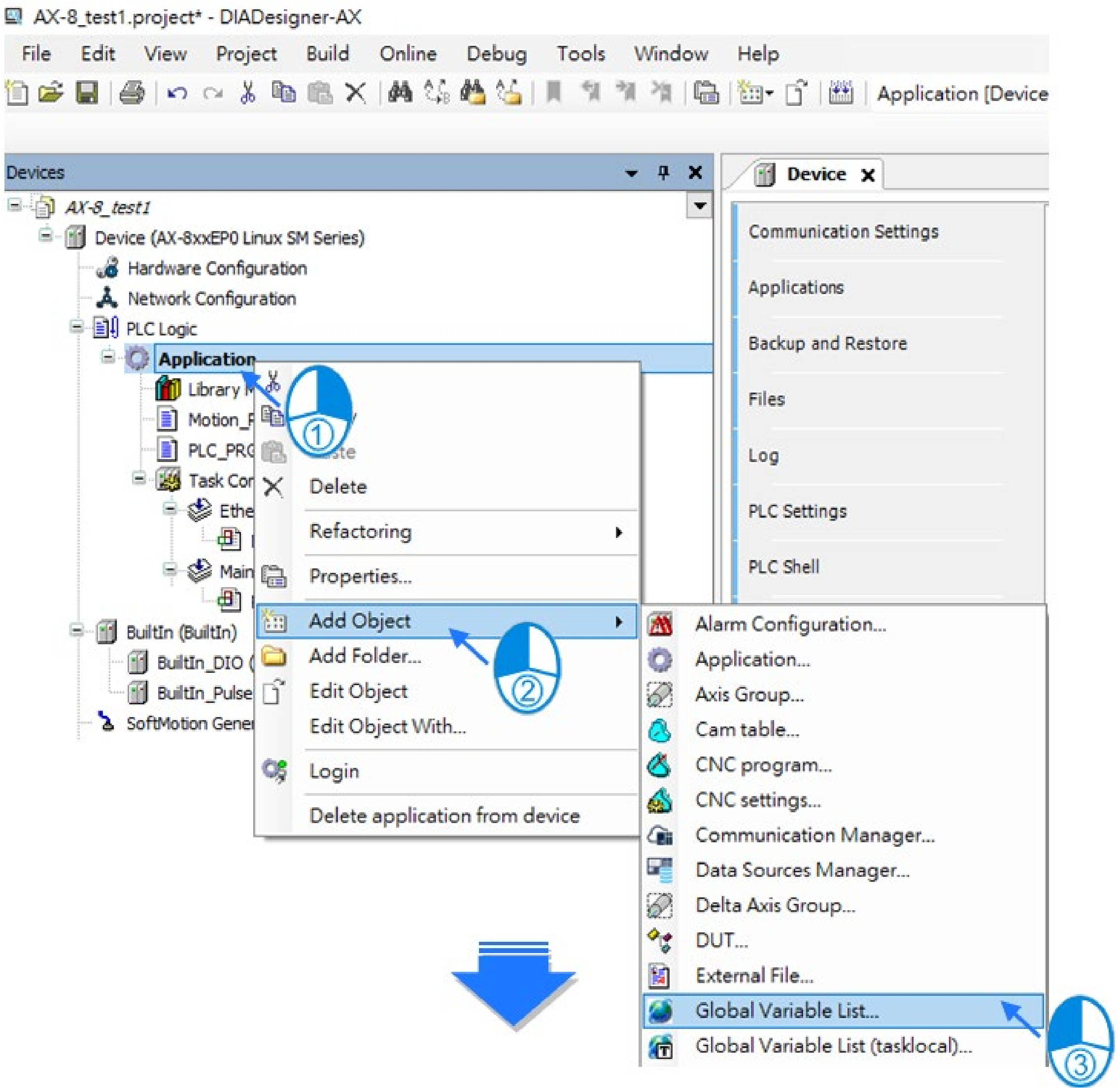
Address	Description
%QX7.5	Single bit address of the output bit 7.5
%IW215	Word address of the input word 215
%QB7	Byte address of the output byte 7
%MD48	Address of a double word at memory position 48 in flag memory
VAR wVar0 AT %IW0：WORD； END_VAR	Variable declaration with address information of an input word
VAR bVar0 AT IX7.5：BOOL； END_VAR	Boolean variable declaration with address information of an input bit X7.5.

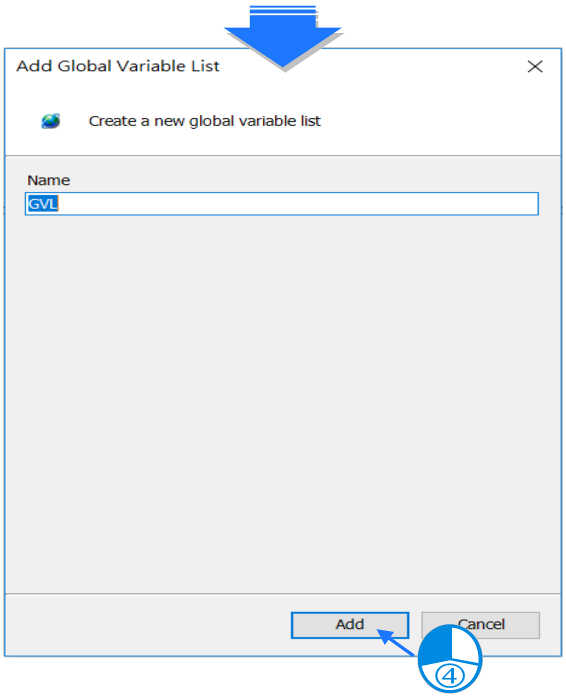
4.3.2.3 Variables

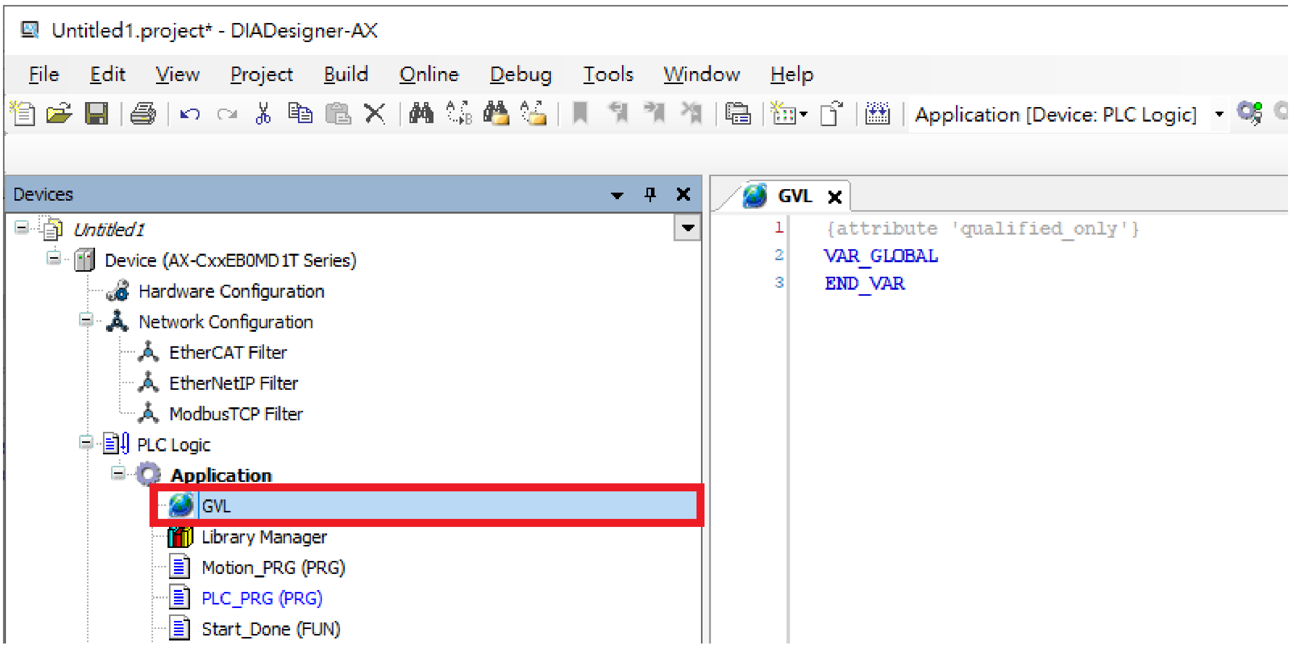
- **Global Variables**

If a variable is declared in a POU, it is a local variable that can only be used within the POU. However, when a variable is declared in the global variable list, it is a global variable and can be used in any POU.

Add Global Variable List:







- **Constant Variables**

You can declare a variable as a constant variable. Constant variables can be accessed as read-only without assigning an initialization value.

Declaration of Constant Variables:

```
VAR CONSTANT
    pi   :   REAL := 3.14159 ;
END_VAR
```

- **Retain Variables**

You can declare a variable as retentive or use persistent variable directly. Refer to the table below for comparisons among variables, retain variables and persistent variables.

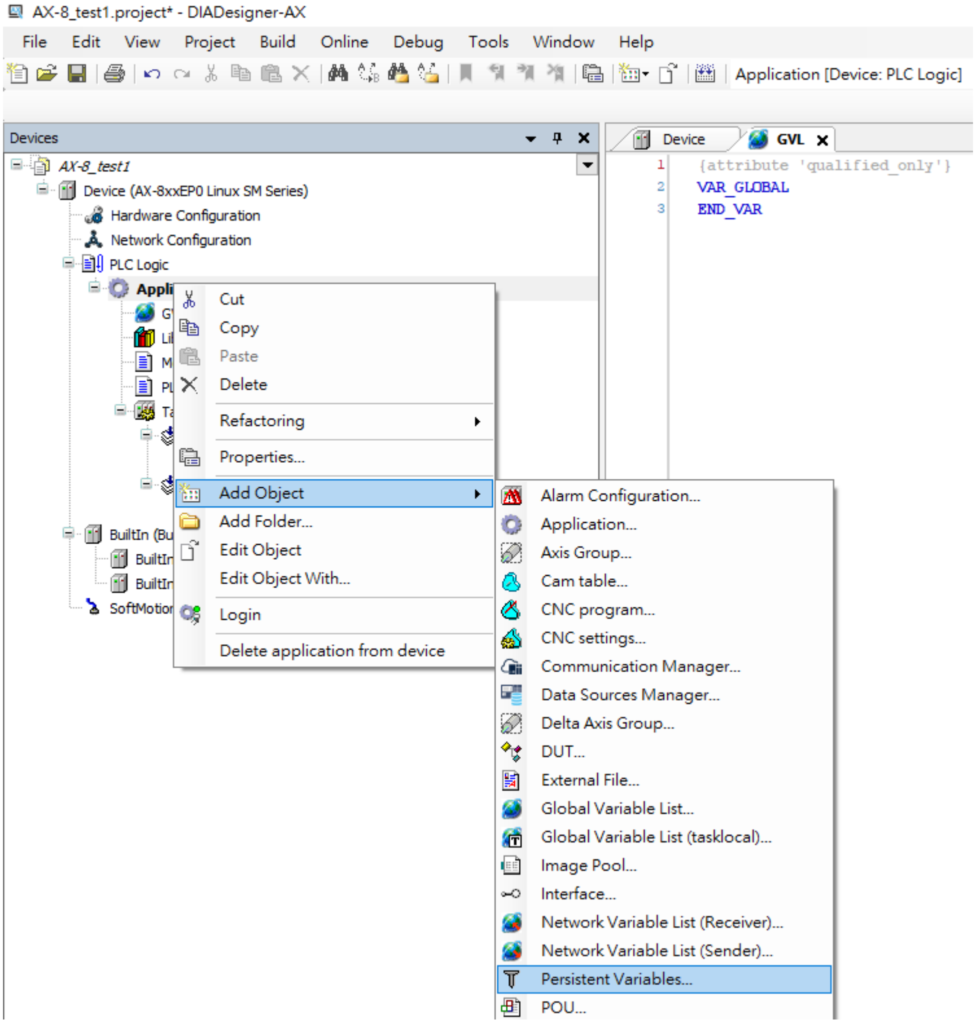
	Initialize				
	Reboot PLC	**Reset warm**	**Reset cold**	**Download**	**Reset origin**
Variable	O	O	O	O	O
Retain Variable	X	X	O	O	O
Persistent Variable	X	X	X	X	O

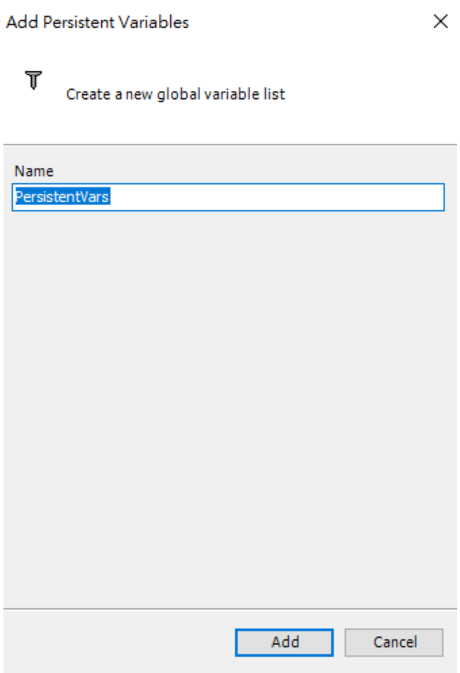
Declaration of Retain Variables:

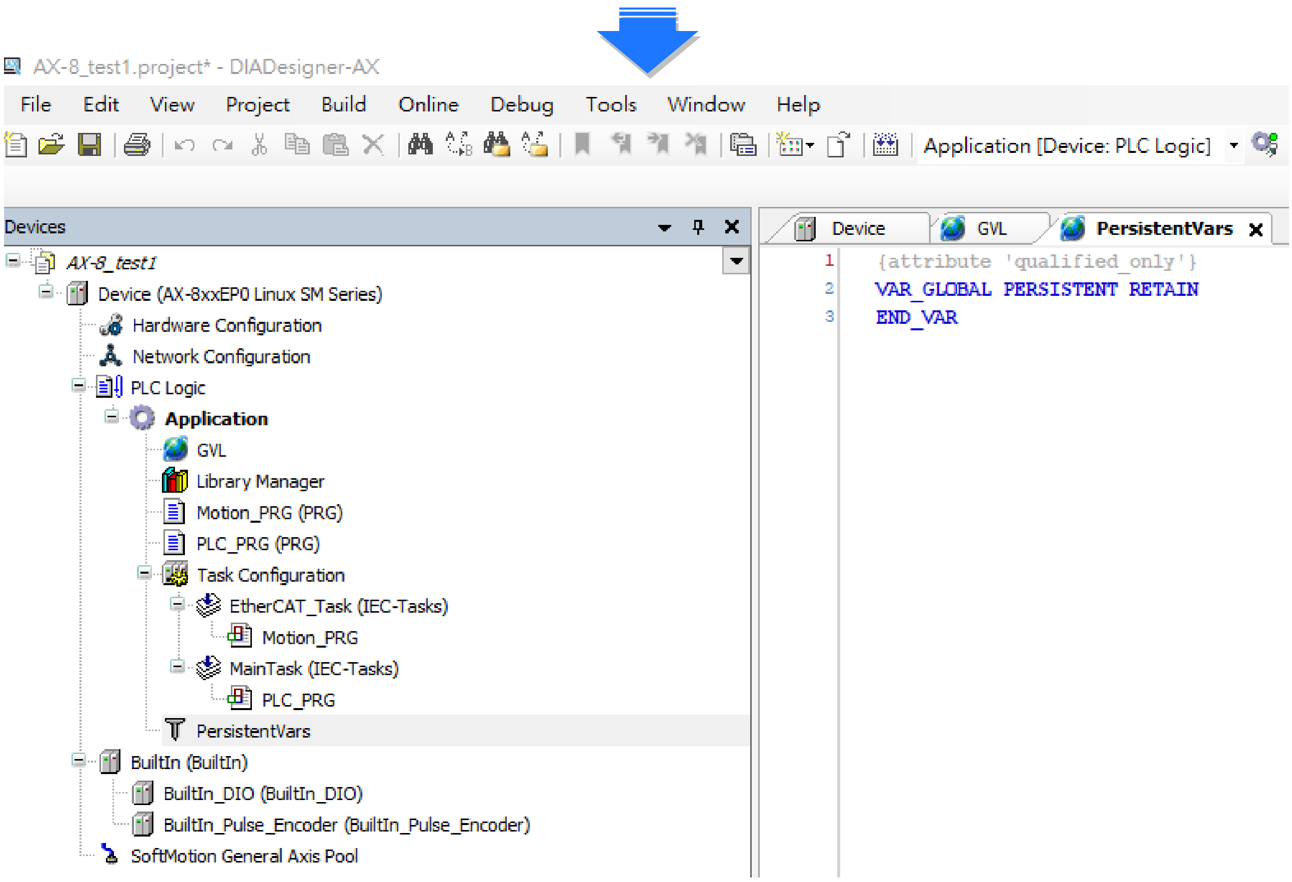
```
VAR RETAIN
    bVar : BOOL ;
    byVar : BYTE ;
    wVar : WORD ;
END_VAR
```

To create persistent variable, you can declare **RETAIN PERSISTENT**, **PERSISTENT RETAIN** and **PERSISTENT** in Persistent Variable Object for the same outcome.

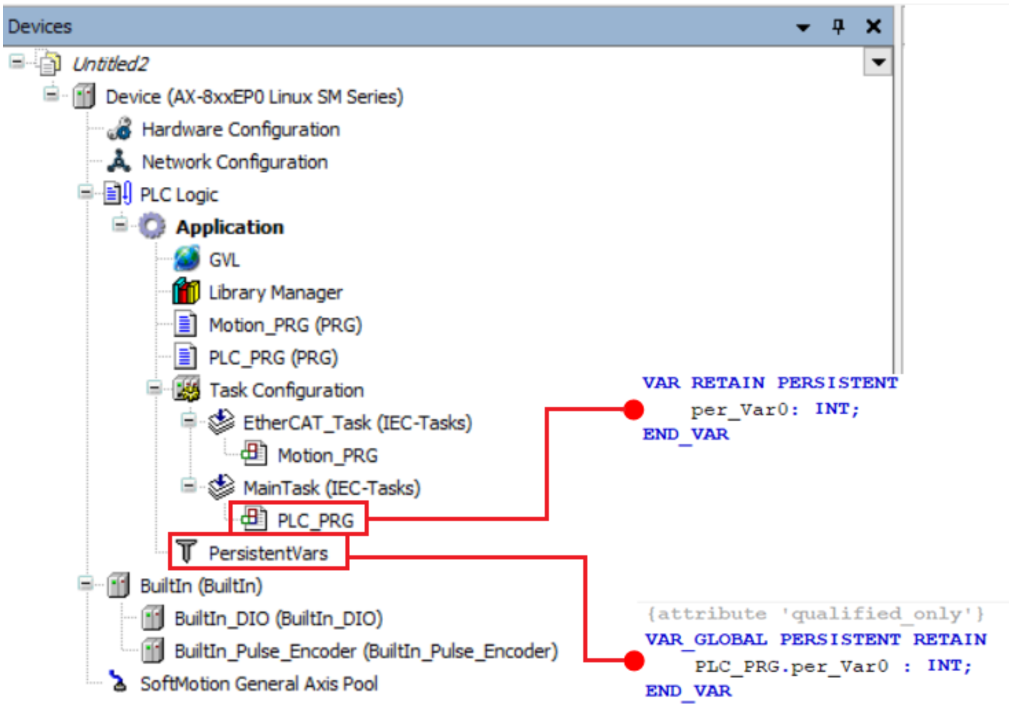
Add Persistent Variable List:







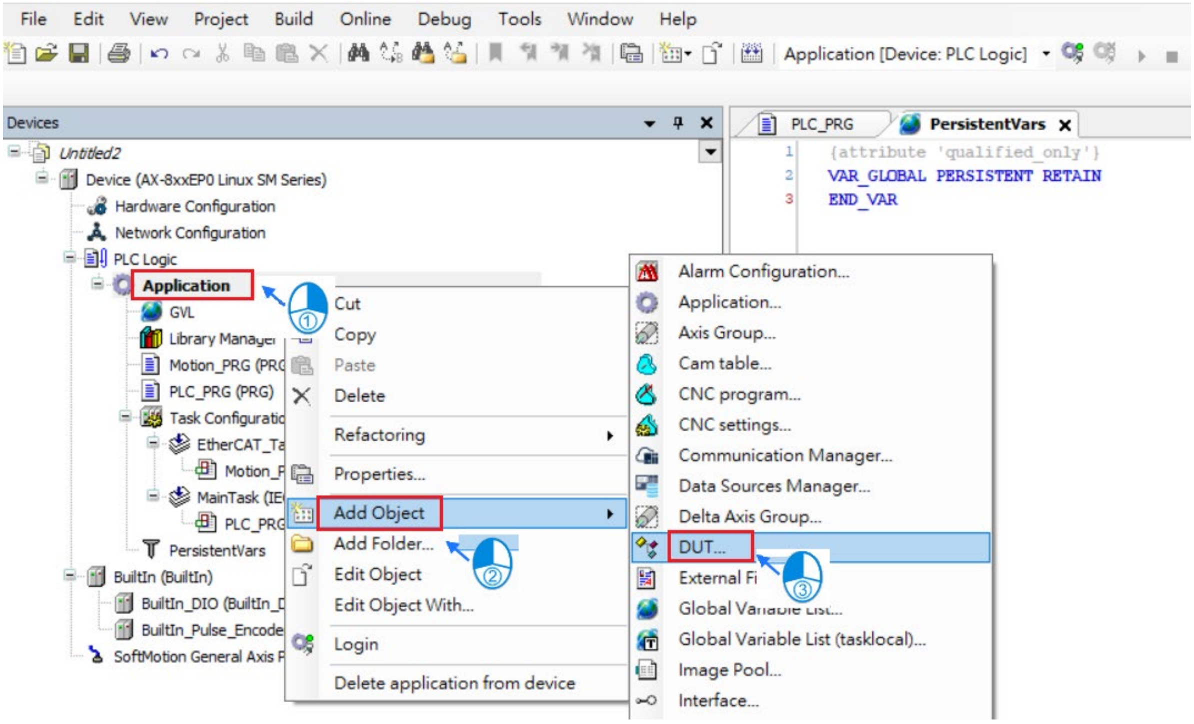
If it is required to declare a local variable as persistent, you have to add the variable instance path to the persistent variable list.



4.3.2.4 User-defined Data Type

You can create your own data type, DUT (Data Type Unit) or UDT (User-defined Data Type), by clicking **Add Object** and selecting **DUT**. There are four data types that can be created: Structure, Enumeration, Alias and Union.

Add DUT:



- **Structure**

A structure is a compound data type used for grouping simple data types or other compound data types.

Syntax:

TYPE <Structure Name>:

STRUCT

<Variable Declaration 1>

...

<Variable Declaration n>

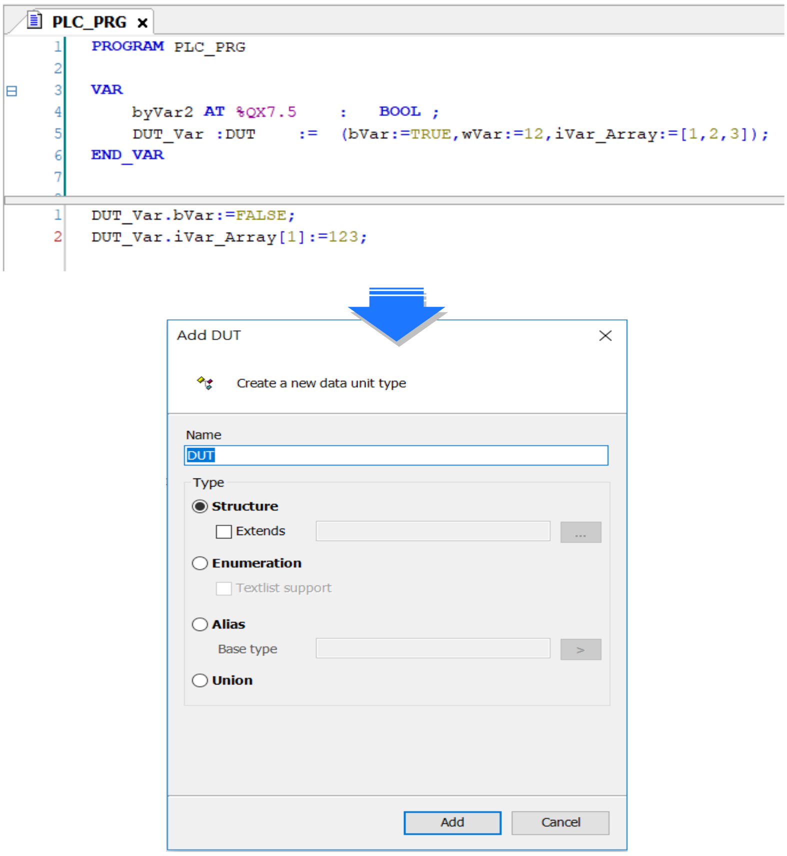
END_STRUCT

END_TYPE

Example:

```
TYPE DUT :
STRUCT
    bVar       :   BOOL     ;
    wVar       :   WORD     ;
    iVar_Array   :   ARRAY[0..2]OF INT   ;
END_STRUCT
END_TYPE
```

Application:



- **Enumeration**

An enumeration is used to map a set of names to numeric values. Enumerated data types make code more self-documenting and program listing more readable.

Syntax:

TYPE <Enumeration Name>:

(

<First Component Declaration>:= Component Declaration,

… ,

< Last Component Declaration >:= Component Declaration

) <Basic Data Type> := Default Variable Initialization;

END_TYPE

Example:

```
TYPE Enumeration_0 :
(
    GREEN := 0,
    YELLOW:=3,
    RED:=8
) INT:=YELLOW;
END_TYPE
```

- **Alias**

Alias is a scalar data type for a variable that can save a single value and self-define the data type.

Example:

TYPE <Alias Name> : STRING(20); END_TYPE

- **Union**

Union is a data structure that contains different data types. All components share the same amount of memory.

Syntax:

TYPE <Union Name>:

UNION

<Variable Declaration 1>

...

<Variable Declaration n>

END_UNION

END_TYPE

Example:

```
TYPE DUT_Union :
UNION
    unVar0:WORD;
    unVar1:DWORD;
END_UNION
END_TYPE
```

4.3.2.5 Timing for the Variable being Cleared to Zero

For different types of variables, the timings to clear the variables to zero are various. Refer to the table below for the timings on different occasions*1.

● = value retained

○ = value cleared to zero

Action	VAR	VAR Retain	VAR Retain Persistent
Online Change	●	●	●
Reboot PLC	○	●	●
Reset Warm	○	●	●
Reset Cold	○	○	●
Download	○	○	●
Reset Origin	○	○	○

*1: The default value takes effect when value retention feature is unavailable.

4.3.2.6 Timing for the Default Value to be Valid

● = invalid

○ = valid

Action	VAR	VAR Retain	VAR Retain Persistent
Online Change	●	●	●
Reboot PLC	○	●	●
Reset Warm	○	●	●
Reset Cold	○	○	●
Download	○	○	●

4.4 Task

4.4.1 Task Configuration

Task is used for controlling and executing the program blocks (POUs) in the PLC. You can define a task with its name, priority and the exact condition that initiates the commencement of the task. This condition is defined either by time (cyclic-interval, freewheeling) or by the occurrence of an internal or external event to initiate the task.

A task calls one or multiple program blocks (POUs). The sequence of task execution depends on the condition and priority defined. It is also possible to configure a watchdog for each task.

Rules for the order in which the defined tasks are processed:

- If the task condition is satisfied, then the system starts processing the task.
- If several tasks satisfy the condition for processing at the same time, then tasks with the highest priority*1 are processed first.
- If several tasks with the same priority level satisfy the condition for processing at the same time, then the system processes the task which has been waiting for the longest time first.
- The program calls are processed in the order they appear in the configuration dialog of the task. If it happens that a POU name in the project is identical with one in the library used, the POU in the project takes precedence to be executed.

*1: Set the priority level from 0 to 31. A set number closer to 0 has a higher priority.

4.4.1.1 Task Type

There are five task types:

- **Cyclic Task**
  The system processes the task in cycles. The execution cycle of the task is defined in the Interval input field.

- **Event Task**
  The system starts processing the Event Task as soon as the global variable defined in the Event input field contains a rising edge. The variable should be a Boolean variable.

- **External Task**
  The system processes the task once as soon as the defined built-in input contains a rising edge.

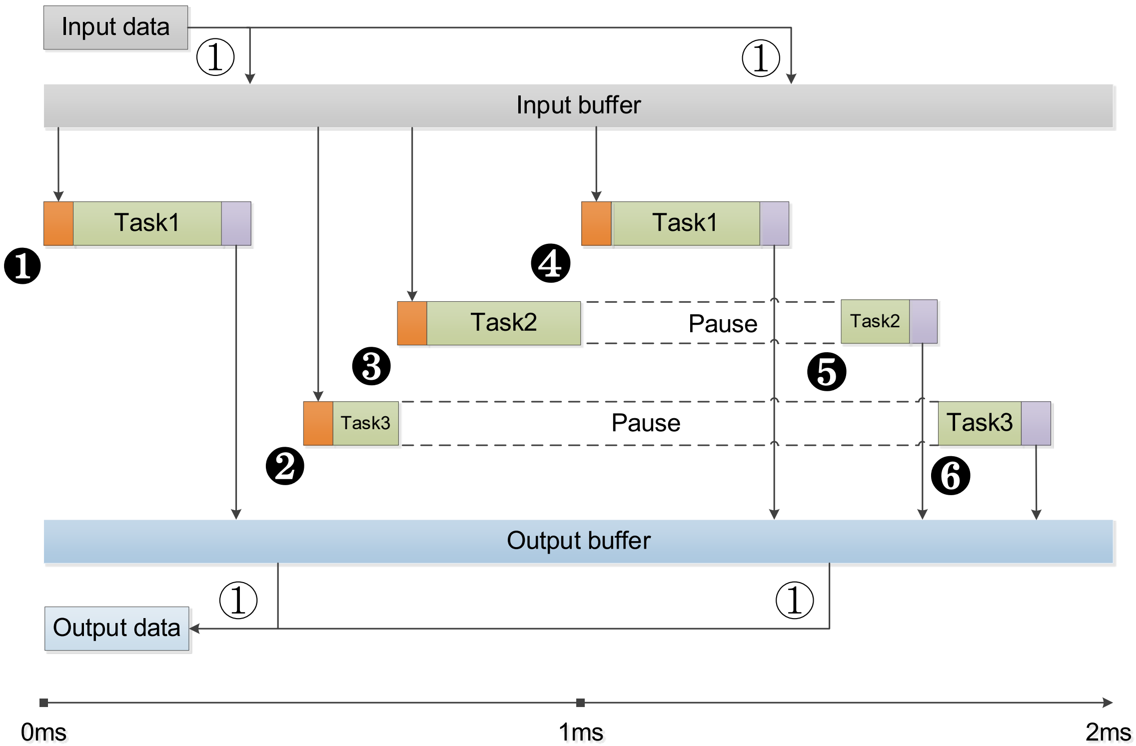
- **Freewheeling Task**
  When the controller starts running, the system begins to process the Freewheeling Task automatically in a continuous loop till the controller stops.

- **Status Task**
  The system starts Status Task processing as soon as the variable defined in the Event input field yields the Boolean value TRUE.

4.4.1.2 Bus Cycle Task

The system processes the task when its condition is satisfied. The priority level is set from 0 to 31 and the number closer to 0 has higher priority. The system processes the task in the order of Task Group in Task Configuration.

Sequence of Bus Cycle Task



Task 1: Priority = 1, Bus cycle Task, Cyclic Task

Task 2: Priority = 3, Event Task

Task 3: Priority = 5, Freewheeling Task

❶ The condition for starting Task 1 is met; Task 1 starts.

❷ Task 1 is completed and the data from I/O buffer is exchanged with the I/O channel (physical hardware). Task 3 starts.

❸ The condition for starting Task 2 is met and Task 2 has higher priority than Task 3 does. Thus Task 2 starts whereas Task 3 halts.

❹ The condition for starting Task 1 is met and Task 1 has higher priority than Task 2 does. Thus Task 3 starts whereas Task 4 halts.

❺ Task 1 is completed and the I/O data from buffer is exchanged with the I/O channel (physical hardware). Task 2 starts again.

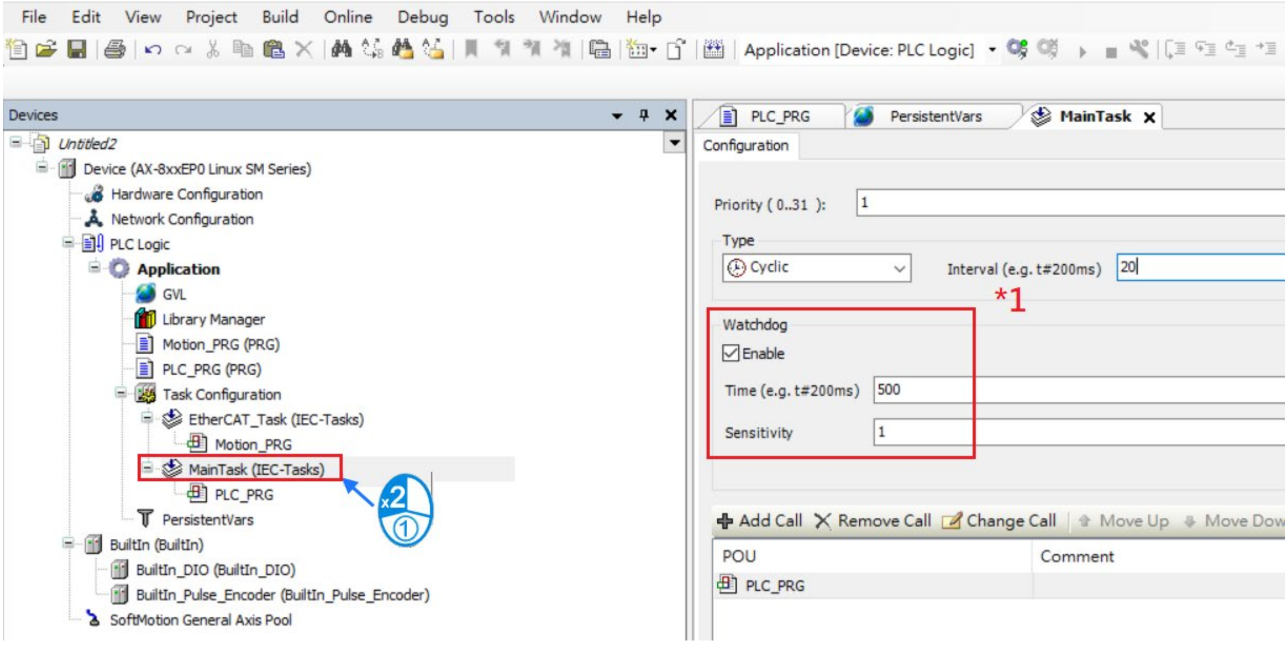
❻ Task 2 is completed and the Task 3 starts again.

① Bus Cycle: The messages are normally sent on the bus in this task. Other tasks copy only the I/O data from an internal buffer that is exchanged only with the physical hardware in the bus cycle task.

4.4.1.3 Watchdog

If the time for task execution exceeds the time set for watchdog, then the task is halted with an error status.

To enable watchdog:



*1: several consecutive timeouts:

sensitivity = 0, watchdog timeout = time x 1

sensitivity = n, watchdog timeout = time x n

4.4.1.4 Motion Instructions for Each Type of Task

The table below introduces the motion instructions for different task types. **V** means the motion instruction is available for that task type.

- **Synchronous axes**[*1]

Classification	Instruction	Task Type		
		Cyclic	Freewheeling	Bus Cycle EtherCAT
Motion Control Function Blocks	MC_Home			V
	MC_Stop			V
	MC_Halt			V
	MC_MoveAbsolute			V
	MC_MoveRelative			V
	MC_MoveAdditive			V
	MC_MoveSuperImposed			V
	MC_CamIn			V
	MC_CamOut			V
	MC_MoveVelocity			V
	MC_PositionProfile			V
	MC_VelocityProfile			V
	MC_AccelerationProfile			V
	MC_Jog			V
	MC_GearIn			V
	MC_GearOut			V
	MC_GearInPos			V
	MC_Phasing			V
	DMC_TorqueControl			V
	DMC_VelocityControl			V
	DMC_MoveLinearAbsolute			V
	DMC_MoveLinearRelative			V
	DMC_MoveCircularAbsolute			V
	DMC_MoveCircularRelative			V
	DMC_GroupStop			V
	DMC_GroupHalt			V
	DMC_Home_P			V
	DMC_GroupInterrupt			V

	DMC_GroupContinue			V
	DMC_ImmediateStop_P			V
Instructions for Management	MC_Power			V
	MC_SetPosition			V
	MC_ReadParameter			V
	MC_WriteParameter			V
	MC_ReadBoolParameter			V
	MC_WriteBoolParameter			V
	MC_ReadActualPosition			V
	MC_ReadActualVelocity			V
	MC_ReadActualTorque			V
	MC_Reset			V
	MC_ReadStatus			V
	MC_ReadAxisError			V
	MC_CamTableSelect			V
	MC_TouchProbe			V
	MC_AbortTrigger			V
	MC_DigitalCamSwitch			V
	DMC_GroupEnable			V
	DMC_GroupDisable			V
	DMC_GroupReadStatus			V
	DMC_GroupReadError			V
	DMC_GroupReset			V
	DMC_CamReadTappetStatus			V
	DMC_CamReadTappetValue			V
	DMC_CamWriteTappetValue			V
	DMC_CamAddTappet			V
	DMC_CamDeleteTappet			V
	DMC_CamReadPoint			V
	DMC_CamWritePoint			V
	DMC_ChangeMechanismGearRation			V
	DMC_ReadMotionState			V
	DMC_GroupReadParameter			V
	DMC_GroupWriteParameter			V

*1: It is suggested creating a motion function block within a Bus Cycle EtherCAT to avoid inconsistency in movement.

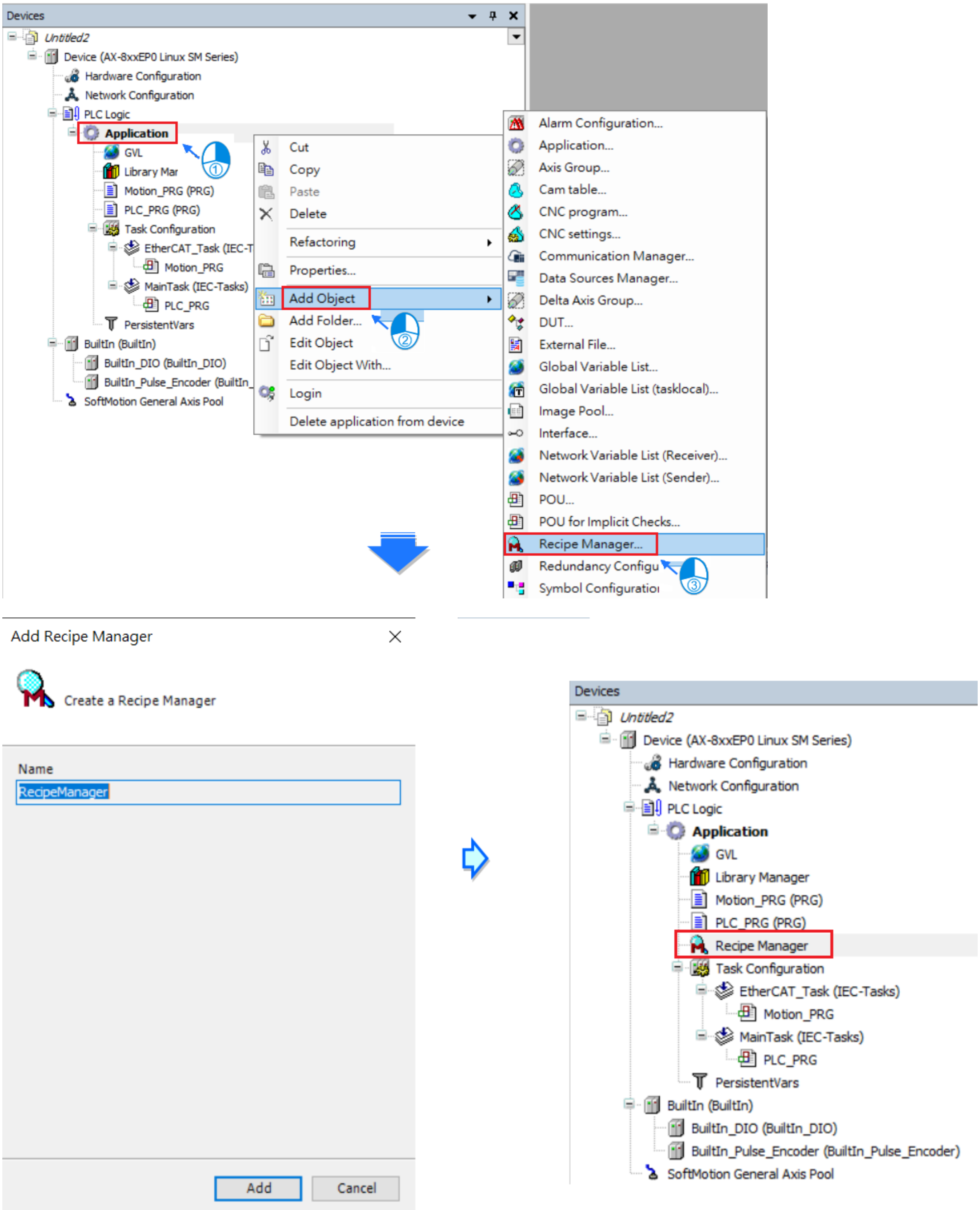
- **Positioning axes**

Classification	Instruction	Task Type		
		Cyclic	Freewheeling	Bus Cycle EtherCAT
Motion Control Function Blocks	MC_Halt_DML			V
	MC_Home_DML			V
	MC_MoveAbsolute_DML			V
	MC_MoveRelative_DML			V
	MC_MoveVelocity_DML			V
	MC_Stop_DML			V
Instructions for Management	MC_Power_DML			V
	MC_ReadBoolParameter_DML			V
	MC_ReadParameter_DML			V
	MC_ReadStatus_DML			V
	MC_Reset_DML			V
	MC_WriteBoolParameter_DML			V
	MC_WriteBoolParameter_DML			V
	MC_ChangeAxisConfig_DML			V
	MC_ReinitDrive_DML			V
	MC_SetOpmode_DML			V
	MC_StartupDrive_DML			V

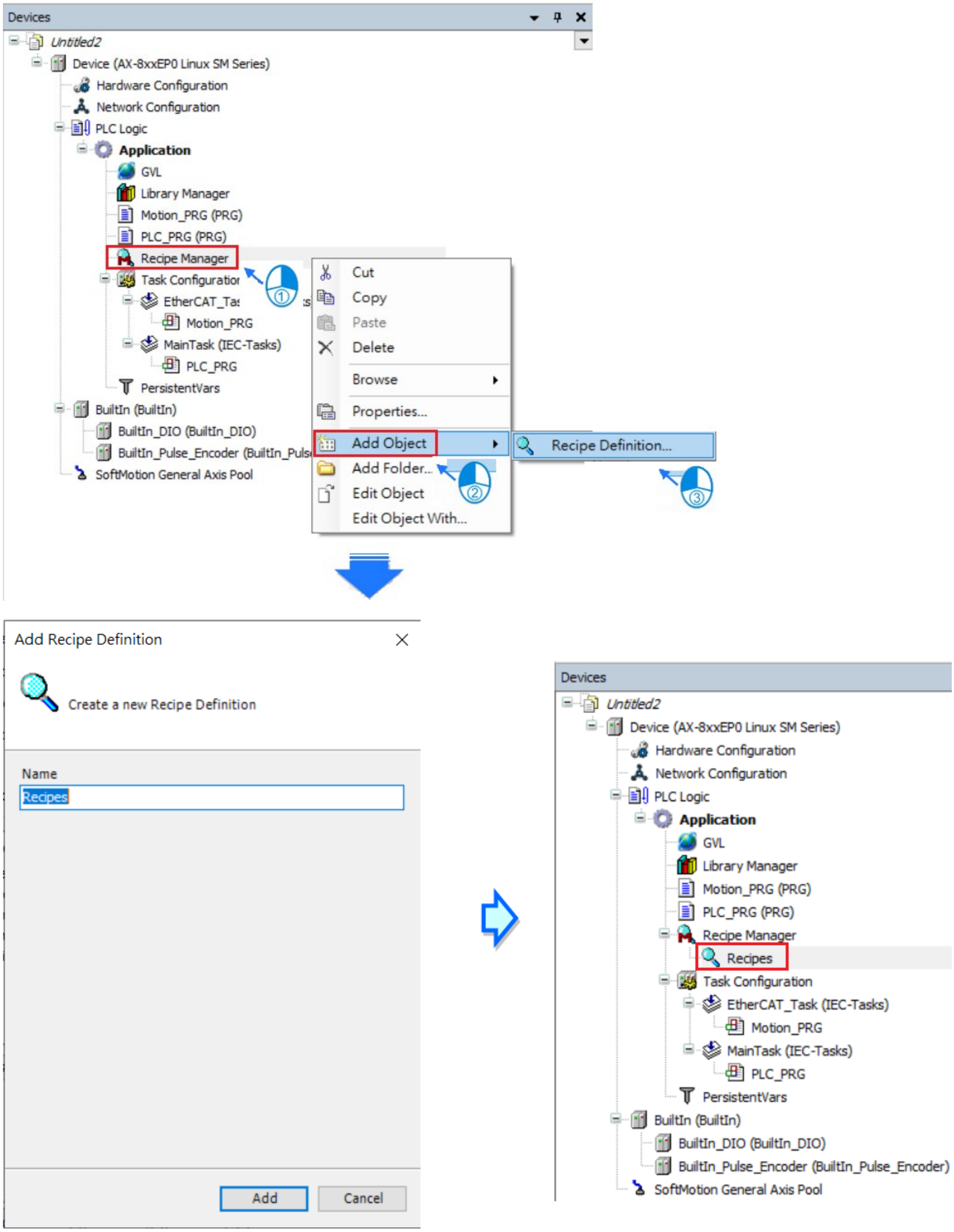
4.5 Recipe Manager

With Recipe Manager, you can import recipe files and export specific parameters by using **RecipeManCommands** from the function block **Recipe_Management.library**.

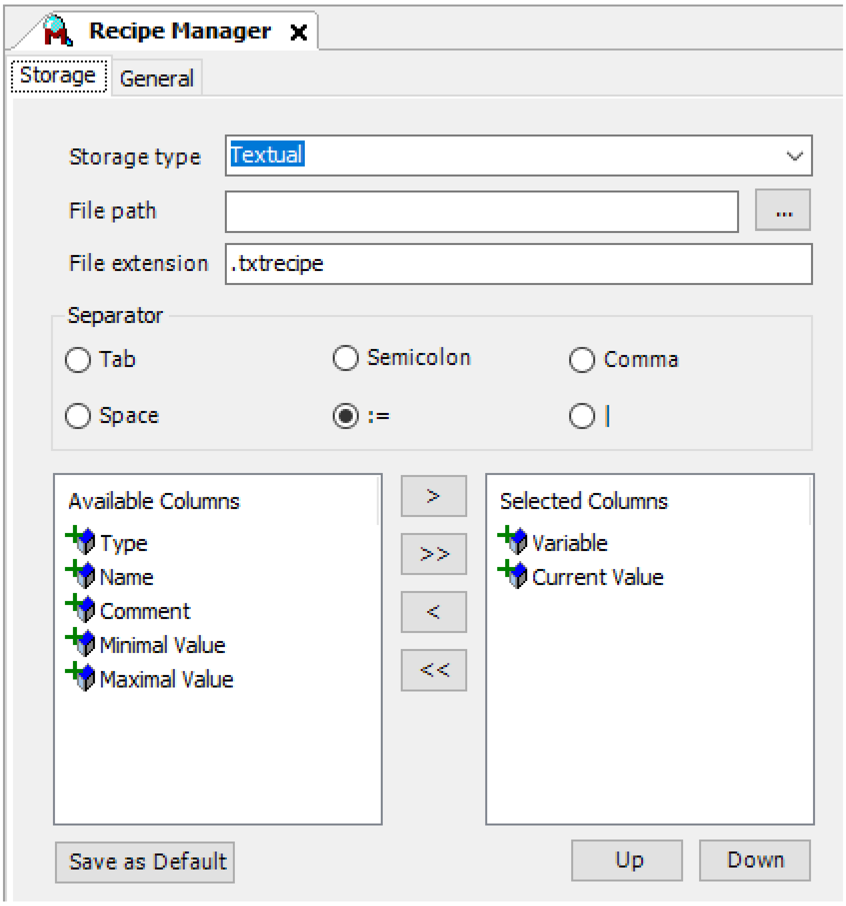
- **Add Recipe Manager:**



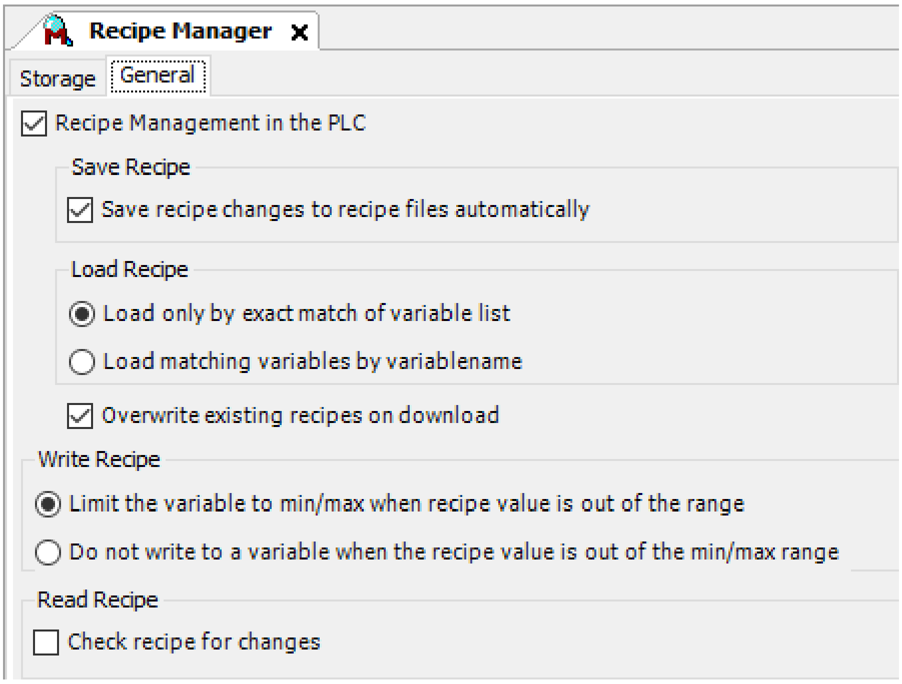
- **Add Recipe Definition:**



4.5.1 Recipe Manager

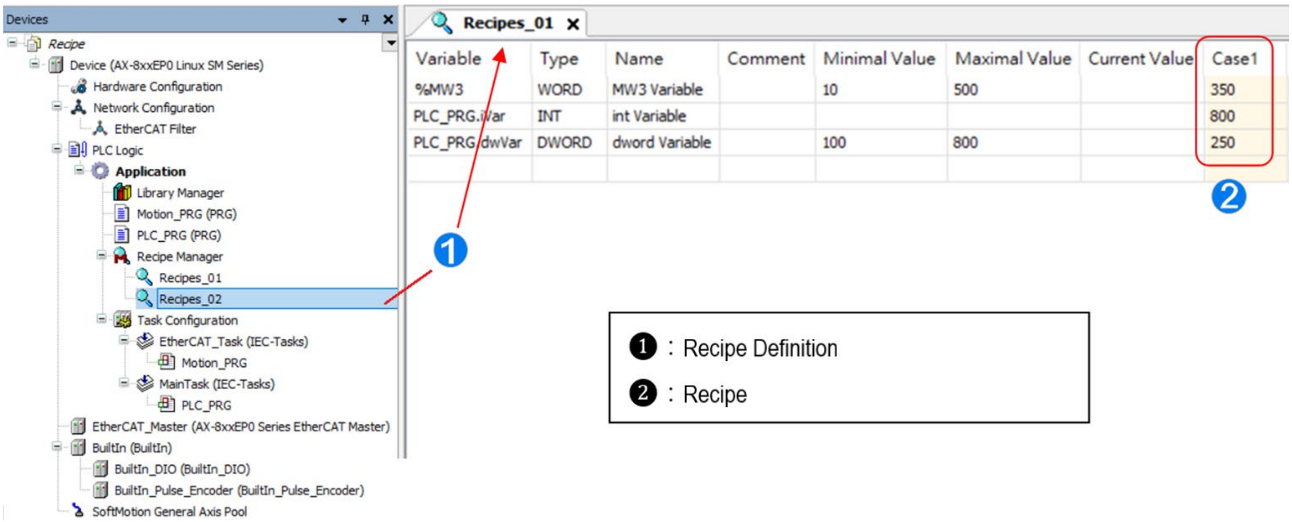


Item	Description
Storage type	The file format to save recipe files. You can choose between Textual and Binary.
File path	The path to save recipe files. Example: If choosing to save files in AllRecipes, the path would be PlcLogic/AllRecipes.
File extension	The extension of the file .<file extension> The naming format of extension files <recipe>.<recipe definition>.<file extension>.
Separator	The separators between each value in recipe files.
Available Columns **Selected Columns**	Define content and order of the recipe files.
Save as Default	Apply the setting to all the recipe managers in the project.
Up	Move the item selected in the Selected Columns upward.
Down	Move the item selected in the Selected Columns downward.



Item	Description
Recipe Management in the PLC	☑ : Recipe Manager is activated.
Save Recipe	
Save recipe changes to recipe files automatically	☑ : Recipe files are updated automatically while downloading projects. In case that Recipe changes, it is auto-saved to the recipe file.
Load Recipe	
Load only by exact match of variable list	☑ : The variables in the file must be in the same order as in the variable list while loading the recipe. Otherwise, the recipe cannot be loaded. (Additional entries at the end are ignored.)
Load matching variables by variable name	☑ : Only variables with matching variable names from the recipe file are loaded, even though the order of variables or the contents in Name column do not match to the setting in the variable list.
Write Recipe	
Limit the variable to min/max when recipe value is out of the range	In case that the recipe value is out of the min/max range, the maximum or minimum value would be written to the corresponding variables in the controller.
Do not write to a variable when the recipe value is out of the min/max range	Prevent a value from being written to the controller if the recipe contains a value that is out of the value range.

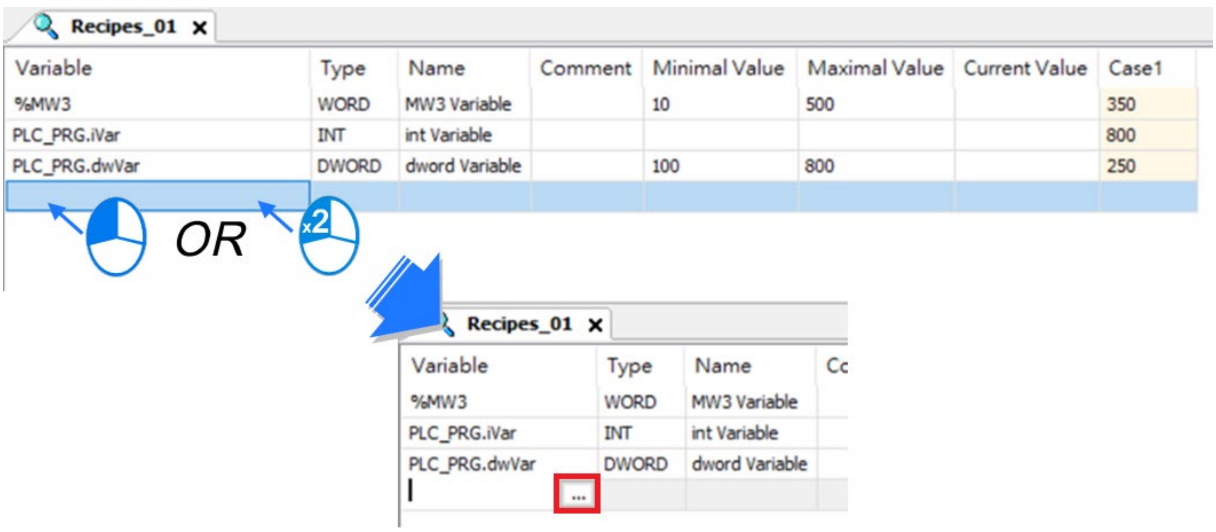
4.5.2 Recipe Definition



Parameter	Description
Variable	In the table, you can specify any variable including those self-defined in a POU.
Type	This automatically displays the relevant data type of the specified variable.
Name	You can define names of variables for inspection and comparison by Load Recipe.
Comment	Additional information.
Minimal Value **Maximal Value**	You can specify the maximum and minimum values to be written to this variable. When the recipe value is out of range, the controller would determine whether to write the value to the variable according to the recipe manager.
Current Value	The current value would be displayed in online mode.

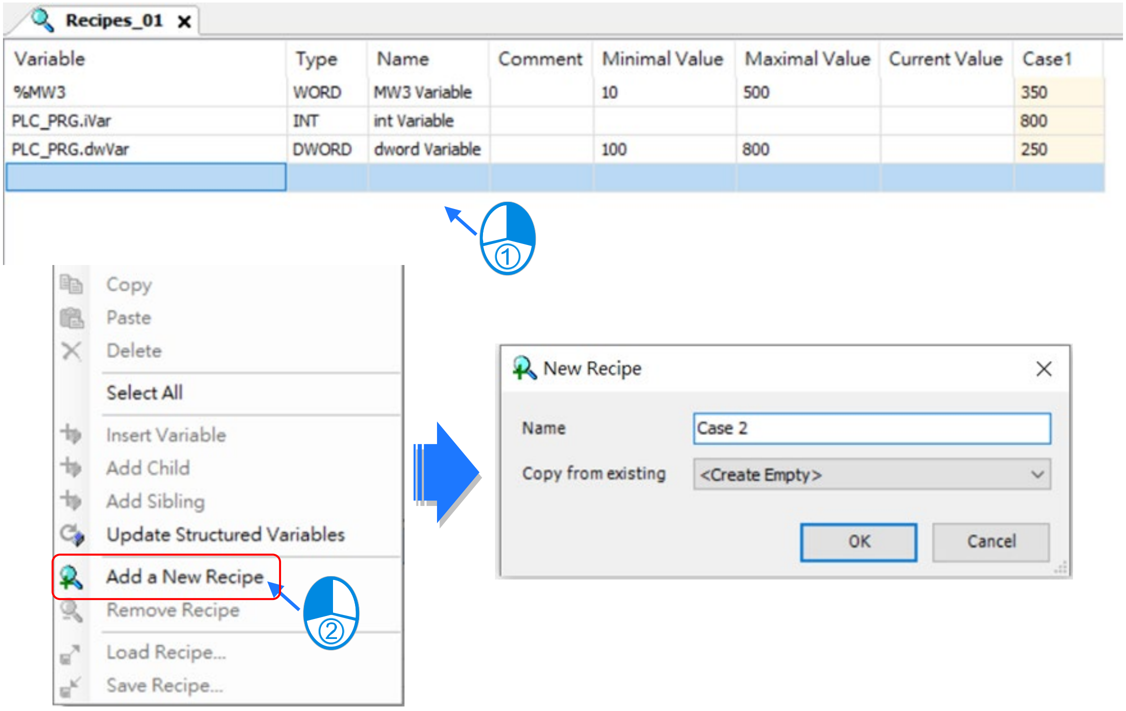
- **Add a New Variable:**

Enter the variable name in the blank or double-click the blank to open **Input Assistant** and choose the target variable.

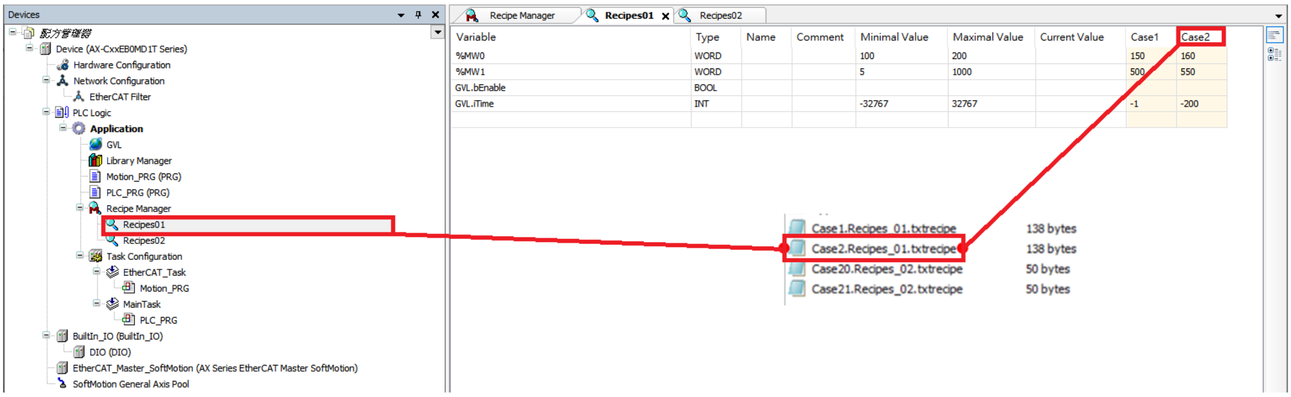


- **Add a New Recipe:**

Right-click the page to select **Add a New Recipe**.



- **The recipe file created in the controller**



4.5.3 RecipeManCommands

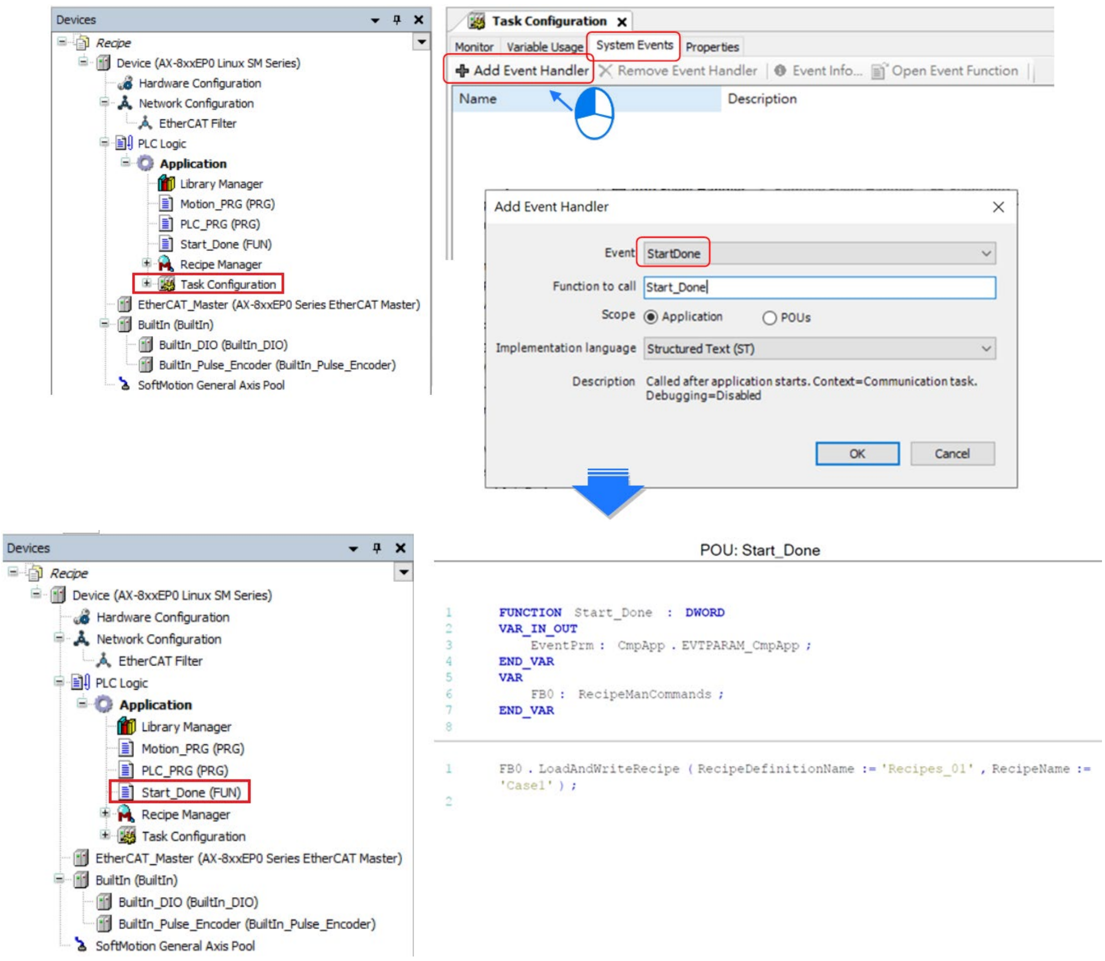
The function block **RecipeManCommands** from **Recipe_Management.library** provides several avenues for users to load recipe files or export recipe files from the controller.

RecipeManCommands	Description
LoadAndWriteRecipe	Load the default recipe file and write the recipe to variables in the controller.
LoadFromAndWriteRecipe	Load the specified recipe file and write the recipe to variables in the controller.
ReadAndSaveAs	Save the variables of the controller to the target file.
ReadAndSaveRecipe	Read the current PLC variables into the default recipe file.
ReadAndSaveRecipeAs	Read the current PLC variables into the default recipe file and save the recipe to the specified recipe file.

4

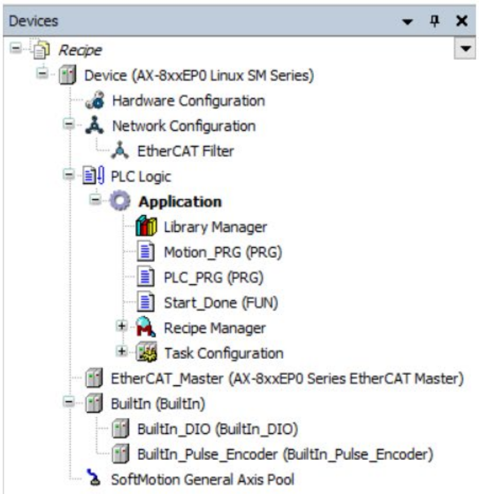
- **Example 1**

In this example, we add **startDone** event through **Add Event Handler**. And with **LoadAndWriteRecipe**, the recipe **Case1** from the recipe definition **Recipes_01** is automatically loaded to the corresponding variables in the controller as soon as the controller's status turns from **STOP** to **RUN**.



- **Example 2**

In this example, we use **ReadAndSaveRecipe** and **ReadAndSaveAs** respectively to read the current PLC values into the default recipe and the specified recipe file.



POU: PLC_PRG

```
PROGRAM PLC_PRG
VAR
    iVar : INT ;
    dwVar , dw_Retur : DWORD ;
    udi_Return : UDINT ;
    bVar0 , bVar1 : BOOL ;
    FB1 : RecipeManCommands ;
END_VAR
```

```
IF bVar0 THEN
    %MW3 := 50 ;
    iVar := 60 ;
    dwVar := 70 ;
    udi_Return := FB1 . ReadAndSaveRecipe ( RecipeDefinitionName :=
'Recipes_01' , RecipeName := 'Case1' ) ;
    bVar0 := FALSE ;
END_IF

IF bVar1 THEN
    %MW3 := %MW3 + 10 ;
    iVar := iVar + 20 ;
    dwVar := dwVar + 30 ;
    udi_Return := FB1 . ReadAndSaveAS ( RecipeDefinitionName := 'Recipes_01' ,
FileName := 'POU_Variable.txtrecipe' ) ;
    bVar1 := FALSE ;
END_IF
```

- **Below are the files and contents generated by the controller when bVar0 and bVar1 are set as ON.**

Name	Size	Modified
..		
visu		
trend		
alarms		
ac_persistence		
_cnc		
Application		
Case1.Recipes_01.txtrecipe	89 bytes	2000/1/1 上午 08:34
Case2.Recipes_01.txtrecipe	92 bytes	2000/1/1 上午 08:33
Case20.Recipes_02.txtrecipe	50 bytes	2000/1/1 上午 08:33
Case21.Recipes_02.txtrecipe	50 bytes	2000/1/1 上午 08:33
POU_Variable.txtrecipe	90 bytes	2000/1/1 上午 08:34

Case1.Recipes_01.txtrecipe - Notepad

File Edit Format View Help

```
%MW3:=50:=WORD:=:=:=:=
PLC_PRG.iVar:=60:=INT:=:=:=:=
PLC_PRG.dwVar:=70:=DWORD:=:=:=:=
```

POU_Variable.txtrecipe - Notepad

File Edit Format View Help

```
%MW3:=60:=WORD:=:=:=:=
PLC_PRG.iVar:=80:=INT:=:=:=:=
PLC_PRG.dwVar:=100:=DWORD:=:=:=:=
```

MEMO

5

Chapter 5 Hardware Configuration

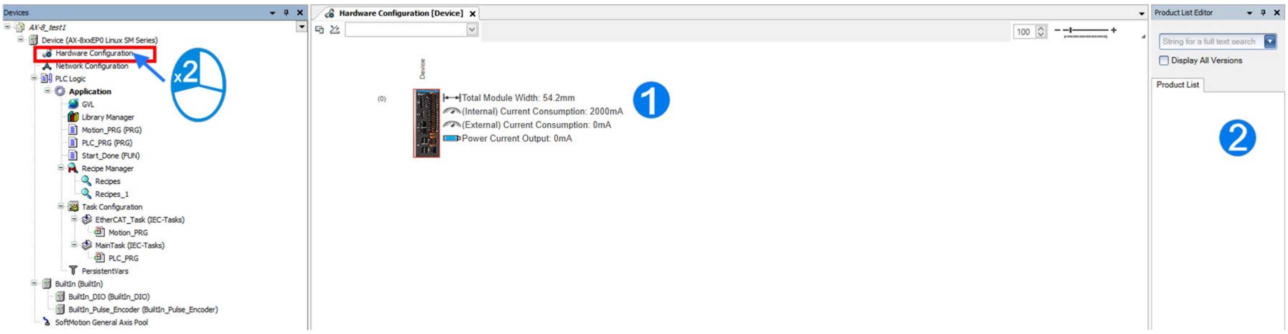
Table of Contents

5.1 Hardware Configuration

Hardware Configuration is the tool in DIADesigner-AX for hardware development. It provides functions including parameter settings for CPU and modules. This chapter provides introduction to the abovementioned functions.

5.1.1 Environment of Hardware Configuration

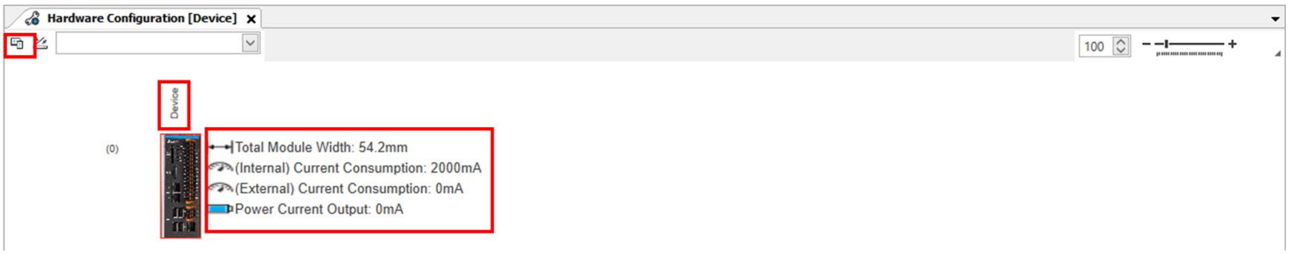
Double-click Hardware Configuration on the Device pane to open the **Hardware Configuration [Device]** window as the image shown below.



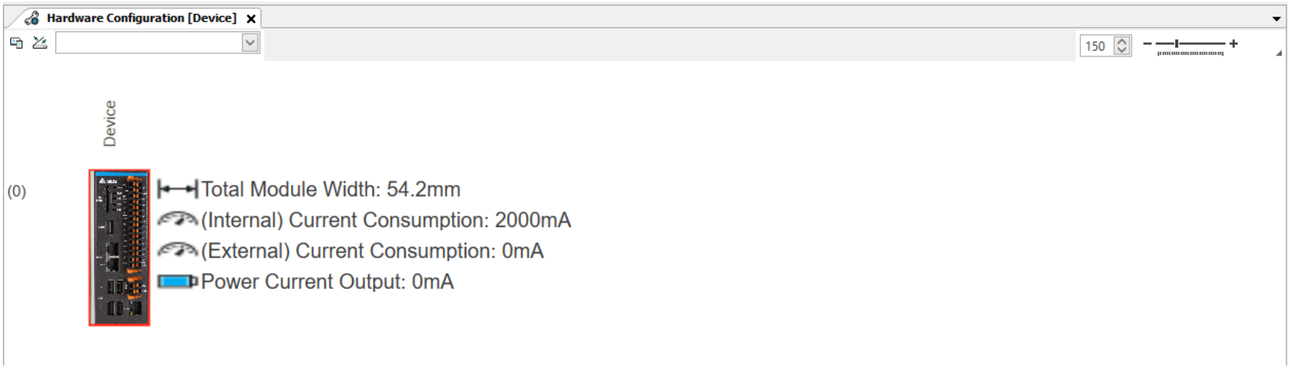
❶ Hardware Configuration [Device]: this is the main working area for system configuration and settings.

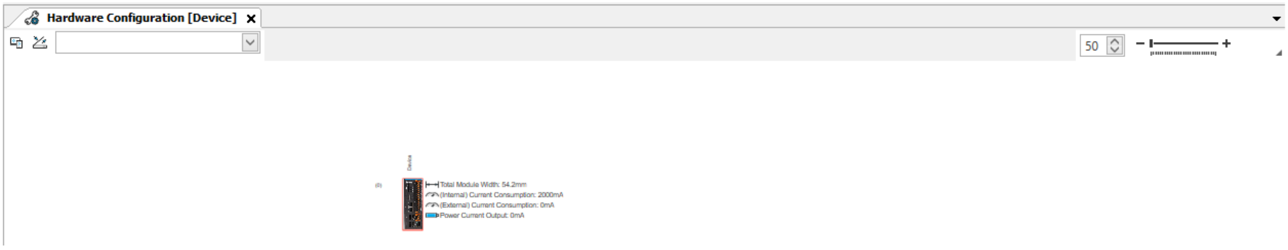
❷ Product List Editor: here lists all the supported modules for the selected CPU.

Click in the top left corner to see the current configuration including information such as the added module names, total width of the connected modules, current consumption and so on.



Use  in the top right corner to rearrange the device image for better viewing experience and easier operation. Below is the display when the controller is in connection.





MEMO

5

6

Chapter 6 Network Configuration

Table of Contents

6.1 Network Configuration

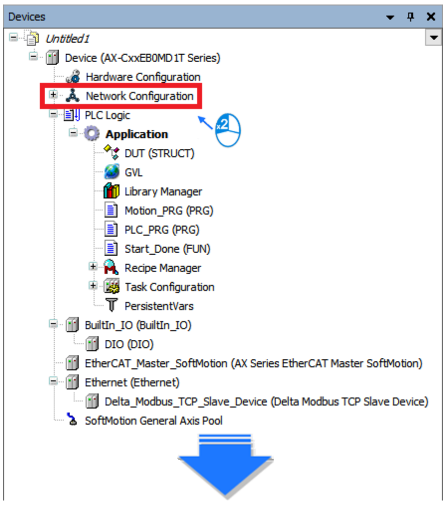
DIADesigner-AX provides a platform for user to configure the network in the project. Details about the settings are covered in this chapter.

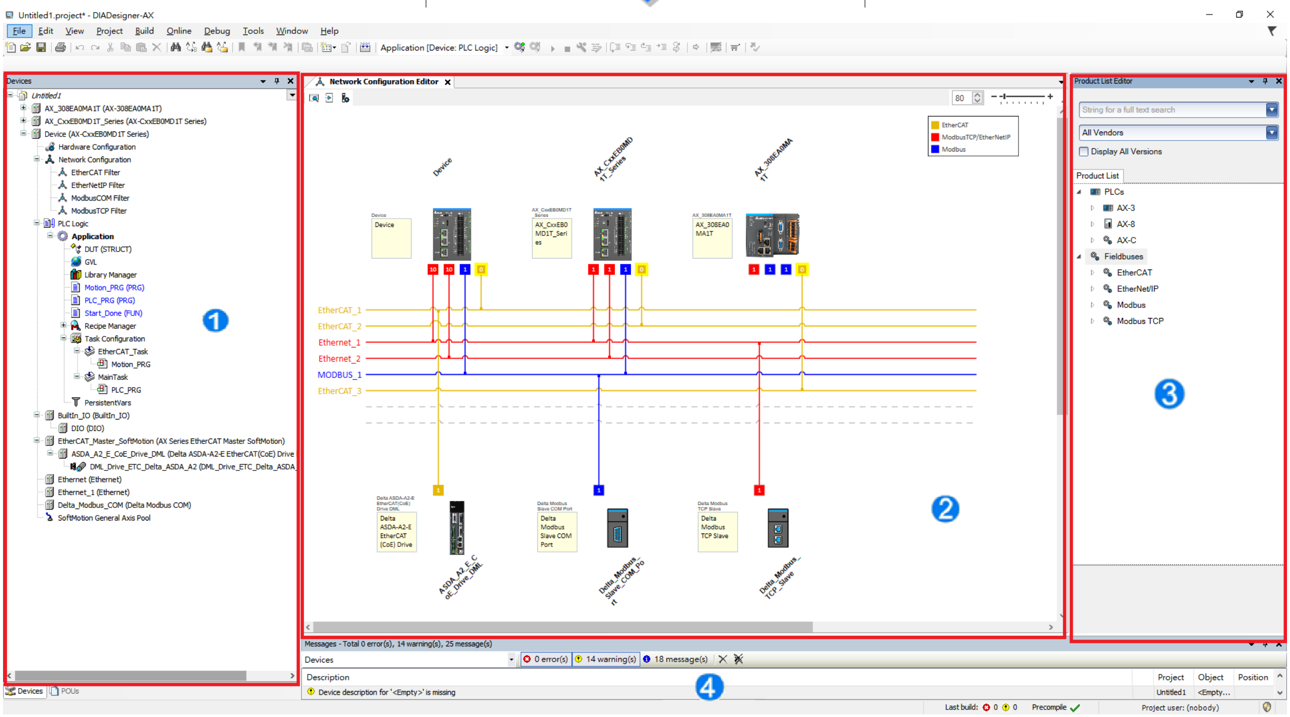
6.1.1 Introduction

Network Configuration is used for:

1. Creating networks such as EtherCAT, Modbus, Ethernet and CANOpen in a project, and setting up paths for data transmission.
2. Setting up EtherCAT master.
3. Setting up Modbus COM port.
4. Setting up Ethernet IP.

Double click Network Configuration in the Device tree to open the setting page.

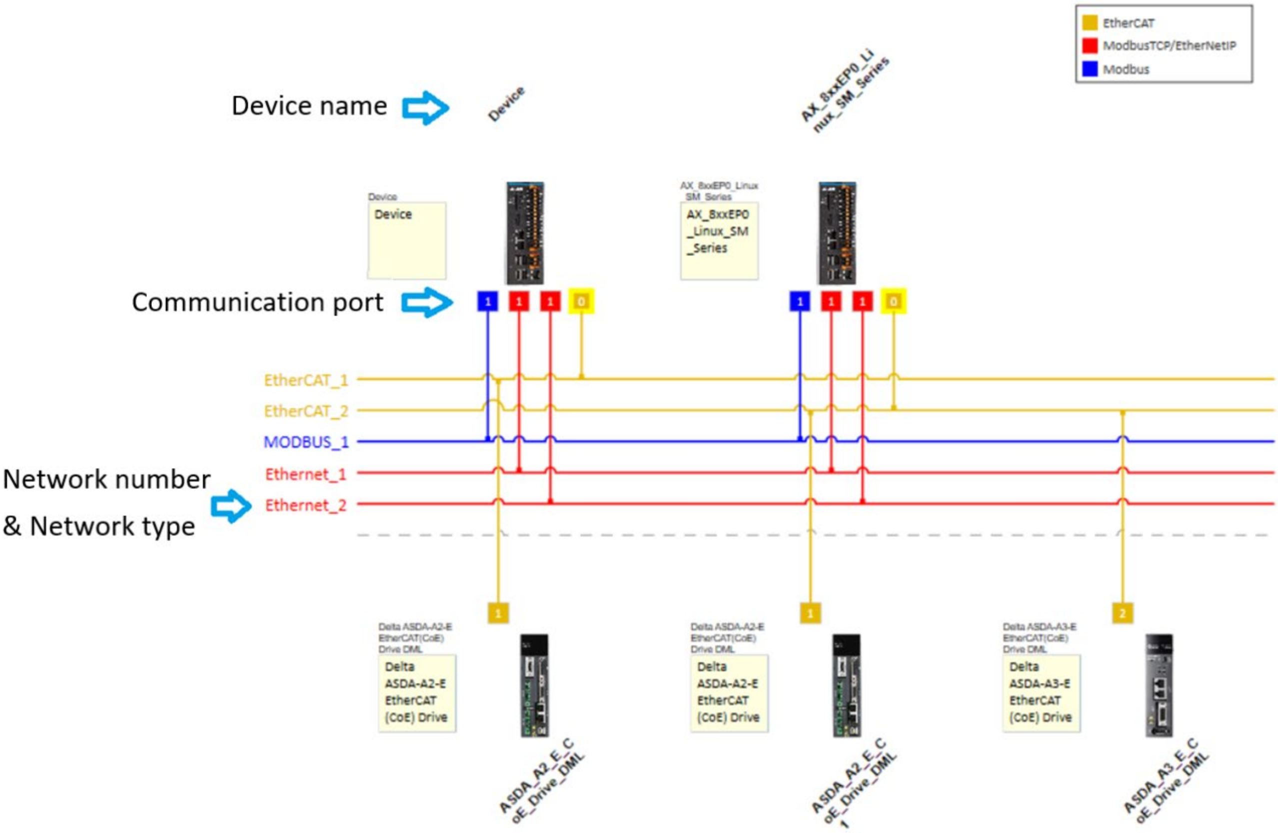




❶ **Device:** all the configured devices are shown in a tree view.
❷ **Working area:** the main working area for overall network configuration.
❸ **Device list:** all the available devices are shown in a tree view.
❹ **Message display area:** operational messages are shown here.

6.1.2 Basic Knowledge

In this section, basic knowledge for network configuration is introduced.



- **Device and Network**

Device is the fundamental element in a network. It can be a PLC, a servo, a driver or any device that you defined. Network is a collection of devices which are interconnected. In a network, every communication port must be assigned with a single network type such as Modbus, Ethernet or EtherCAT. Here, the communication ports of a device serve as the physical interface to connect to a network; therefore, the device can connect to different networks at the same time if it is equipped with more than two ports.

- **Device Name**

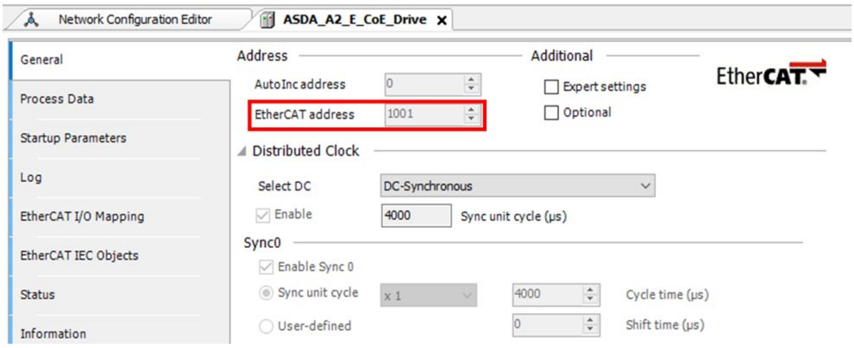
Device name serves as the identity of the device that allows you to identify it in the Device tree. Nevertheless, it bears little significance on operation.

- **Network and Communication Port**

■ EtherCAT

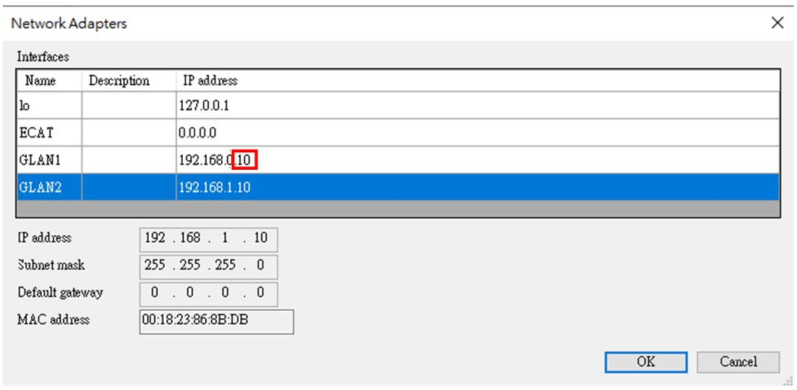
The yellow line  represents the EtherCAT communication. Double-click the Master Station node to open the EtherCAT setting page of the Master. The number of Master Station is 0 and must not be changed.

Double-click the connection node of the Slave  to open the EtherCAT setting page of the Slave. The last digit is used as an indicator of this connection on the Network Configuration Editor page. For instance, the node number is 1 in the EtherCAT address 1001.



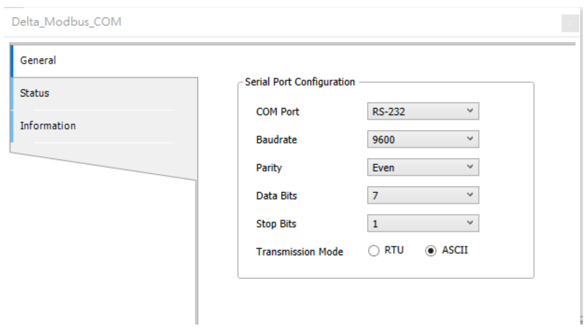
■ Modbus TCP / EtherNet/IP

The red line represents Ethernet port, including Modbus TCP and EtherNet/IP communication. Double-click the line to open its setting page and edit IP addresses. The last digit appeared in the last section of the IP address is used as an indicator of this connection.



■ Modbus

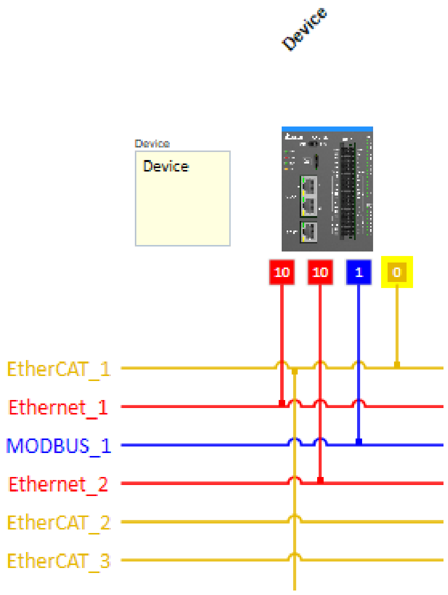
The blue line represents the Modbus communication (RS-485). Double-click the line to open the setting page for Modbus communication port.



6.1.3 Creating a Network Topology

6.1.3.1 Creating Station Nodes

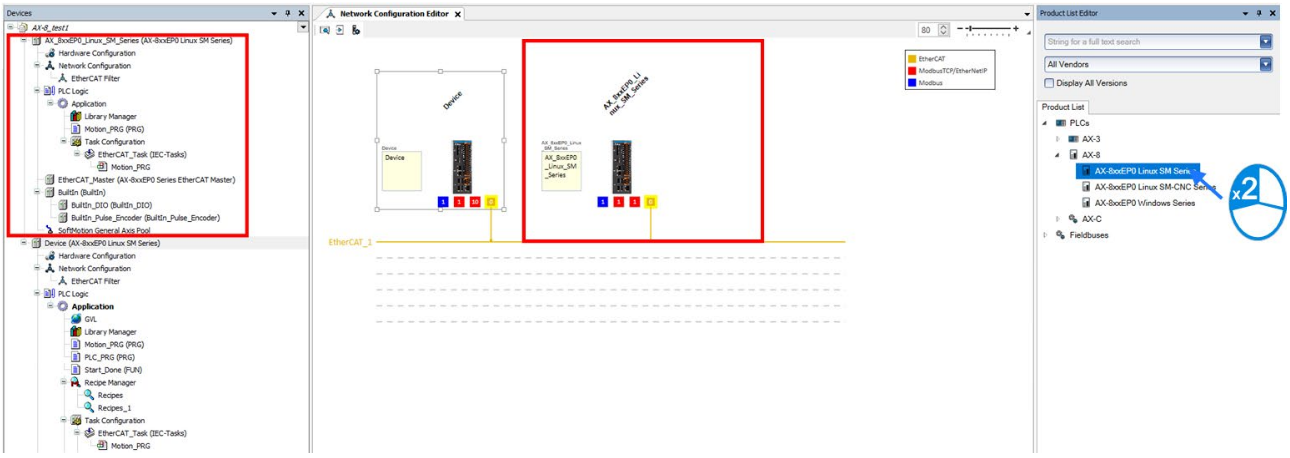
For the first time the network configuration page is accessed, the system automatically creates a graphical representation based on the devices built in the project



You can add additional devices like PLCs, servo motors, and drivers in the network topology in the following ways:

- **Method 1**

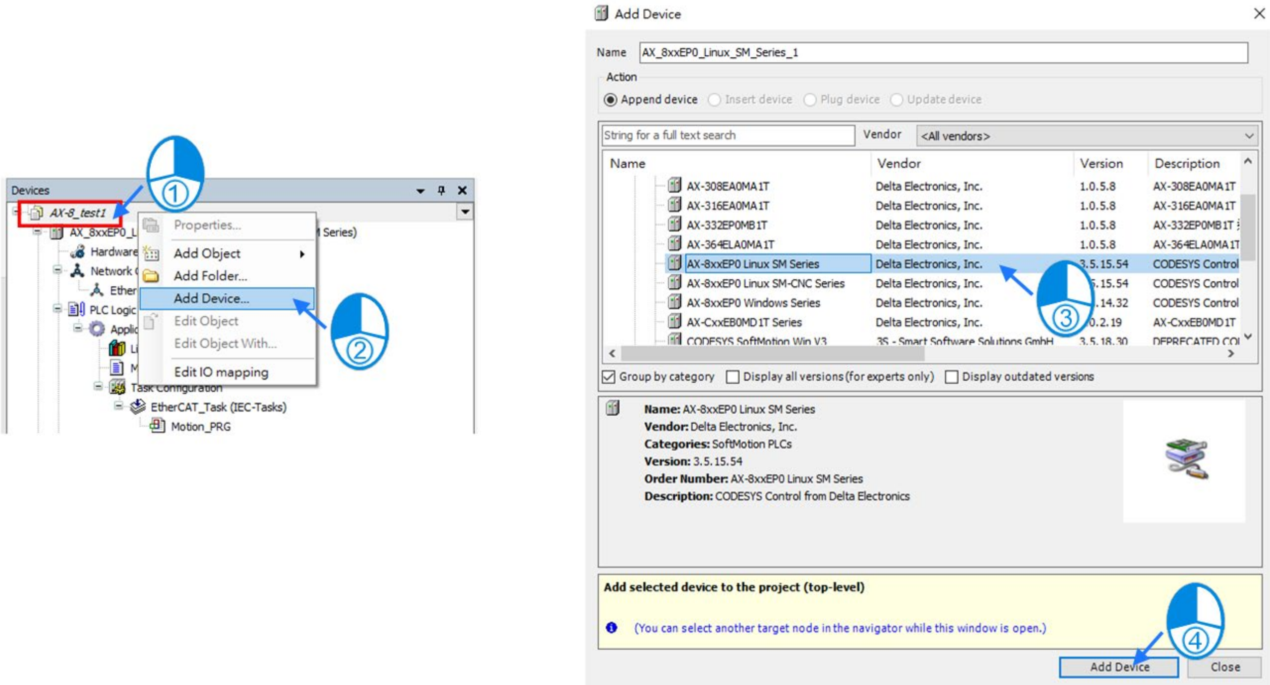
Double-click the device to add from the **Product List Editor** pane on your right-hand side. The device added will be updated in the graphical representation as well as in the Device tree immediately.



6

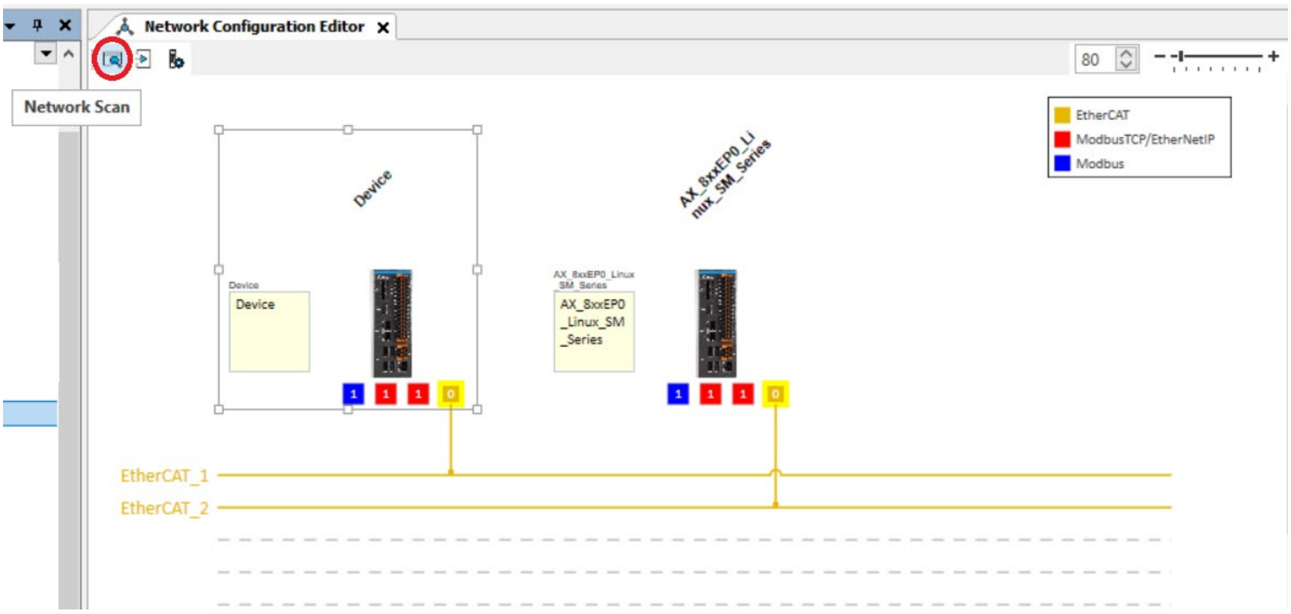
- **Method 2**

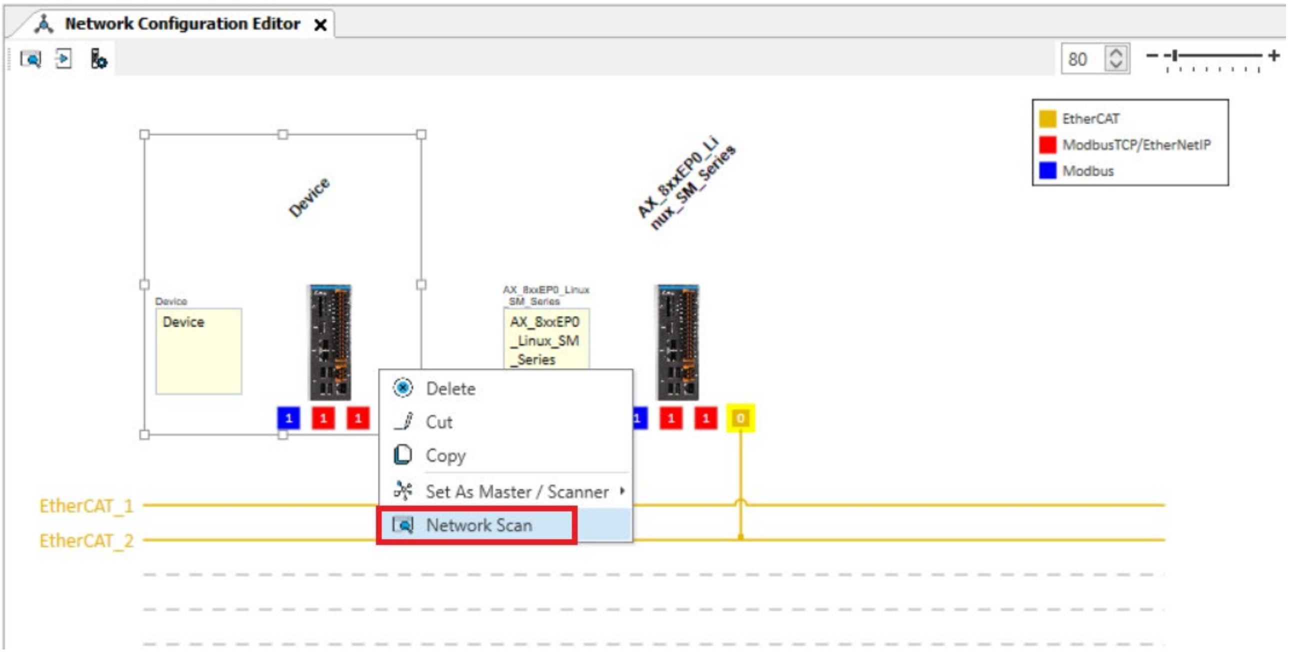
Right-click the project name in the Device tree to bring out the context menu. Double-click **Add Device** on the context menu to open a setting page for adding devices. Double-click the target device or click **Add Device** after selecting the target device.



- **Method 3**

Click the icon  or right-click the device and select **Network Scan** on the context menu to scan and then add the devices and network that have been configured in the project.



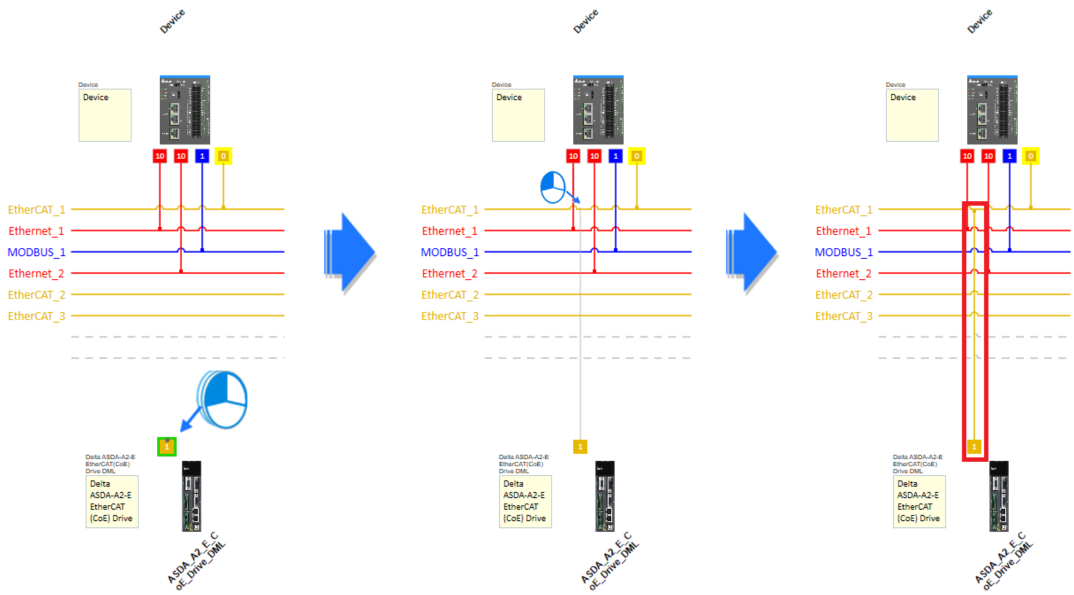


6.1.3.2 Creating Connections

After the station nodes are created, the next step is to establish connections. The network types supported have been listed in **section 6.1.2**. Follow the methods below to establish connections.

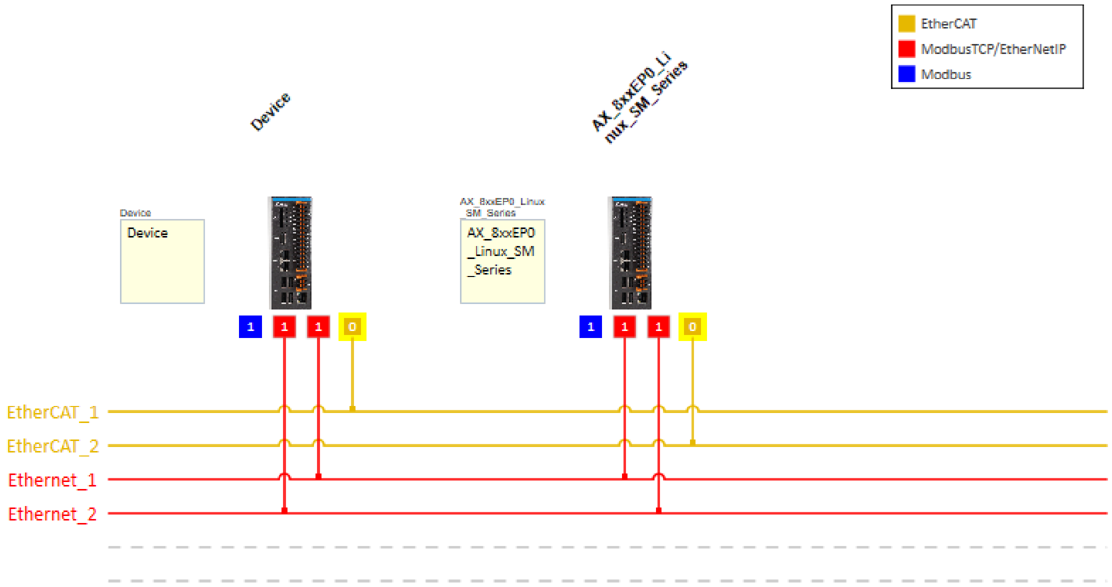
- **Scenario 1**

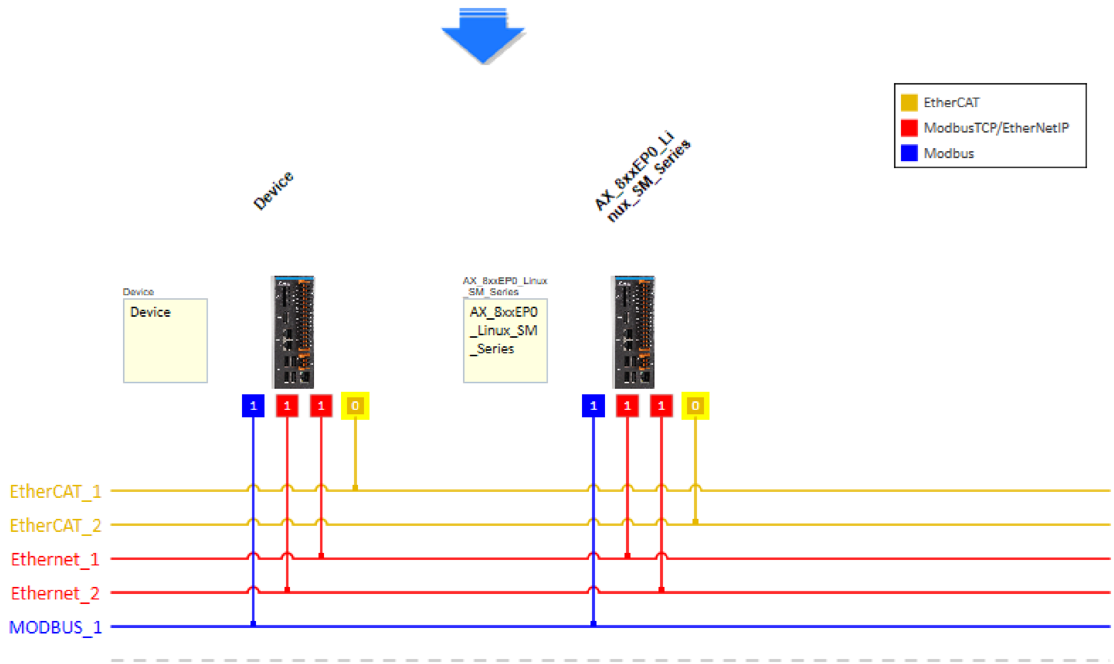
Drag and drop the communication port to the corresponding network type shown in line to create a connection between the devices.



- **Scenario 2**

Hold the communication port and drag it to the unused dotted line to create a network connection that is the same as the selected network communication type. Concurrently, a new unused dotted line in grey is added below.





7

Chapter 7 Basic Settings & Operation of Motion Control

Table of Contents

7

7.1 Overview of Motion Control Instructions

7.1.1 Motion Control Instructions

This section introduces the elements of a motion control program, including devices, symbols, and motion control instructions.

Motion control instructions are defined as function blocks (FBs) and are used in programs to perform various motion control purposes. The MC motion control instructions introduced in this manual are developed based on the specifications of PLCopen [*1] motion control function block.

This section provides an overview of the two motion control instructions, PLCopen and Delta-defined function blocks. PLCopen defines the interface of programs and function blocks, thus realizing the standardized motion control programming environment defined in IEC61131-3. Using PLCopen-based instructions together with Delta-defined instructions can reduce training and technical support costs.

Before using the instructions, make sure you fully understand the functionality of the device, symbols, and instructions. It is recommended to refer to the appendices of the AXn motion control instruction list and related error codes.

*1: PLCopen is an international standard organization that promotes industrial control standards based on IEC61131-3, which is widely adopted for PLC programming. For more information about PLCopen, please check the official website http://www.plcopen.org/.

7.1.2 Application Notes on Motion Control Instructions

This section describes the important operating specifications and limitations when applying motion commands. The detailed information of each command will be described in **Section 7.6.3** Motion Control Programming.

- **Programming languages supported by motion control instructions**

DIADesigner-AX supports all programming languages to create, edit or maintain programs. Supported languages include Ladder Diagram (LD), Sequential Function Chart (SFC), Continuous Function Chart (CFC), Structured Language (ST), and Function Block Diagram (FBD).

For detailed information about the programming language, refer to the **DIADesigner-AX Software Manual**.

7

7.1.3 Categories of Motion Control Instructions

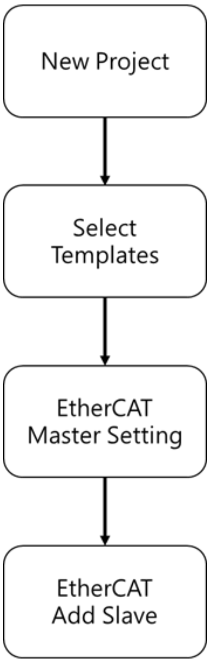
This section introduces the categories of motion control instructions. The corresponding instructions can be found in the libraries SM3_Basic, DL_MotionControl, and DL_MotionControlLight. For more details, refer to the **AX Series Motion Controller Manual**.

Category	Type	Function Group	Description
Single-axis motion control instructions	Motion	Positioning	"MC_": Motion control instruction based on PLCopen "DMC_": Delta custom motion control command "SMC_": Motion command "MC_XXX_DML": Delta's custom motion control instructions, used for positioning axes.
		Velocity control	
		Torque control	
		Synchronized control	
	Administrative	Administrative function	
Multi-axis motion control instructions	Motion	Axis group movement	Multi-axis motion
	Administrative	Administrative functions on multiple axes	Multi-axis configuration, monitoring and reset.

7.2 Creating Motion Control Project

7.2.1 Flowchart

The flowchart below illustrates the process of motion control project creation and axis configuration.

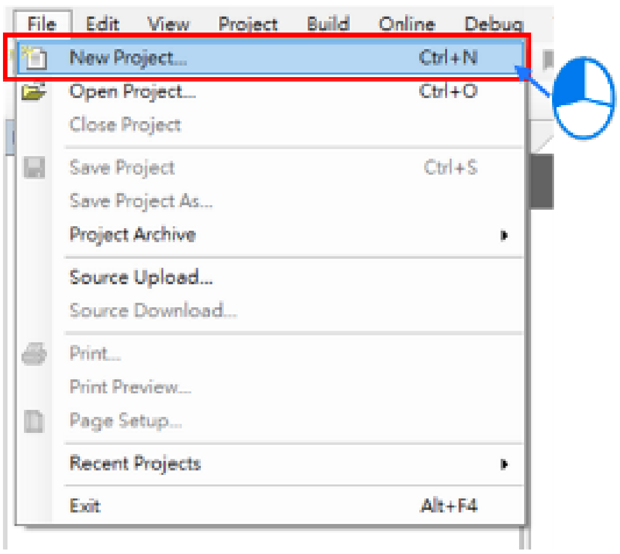


7.2.2 Process for Creating a Project

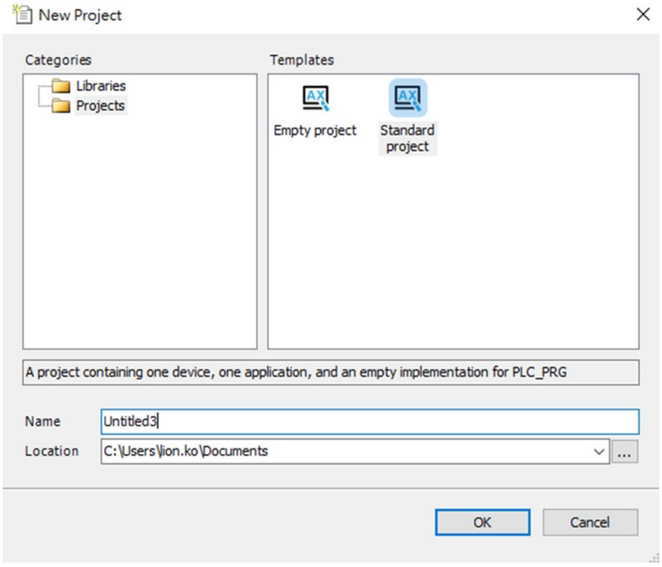
- **Establish a new project**

On the desktop shortcut, double-click  to launch DIADesigner-AX.

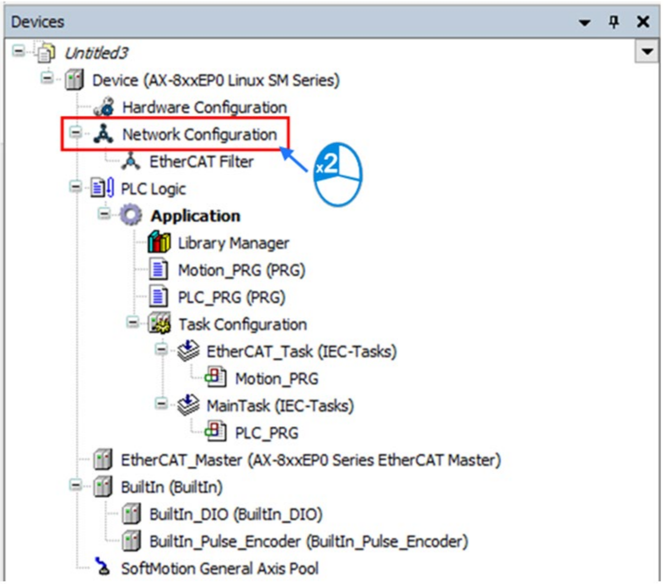
1. Click **File** > **New Project** in the toolbar or click  **New Project...** on the **Start Page**.



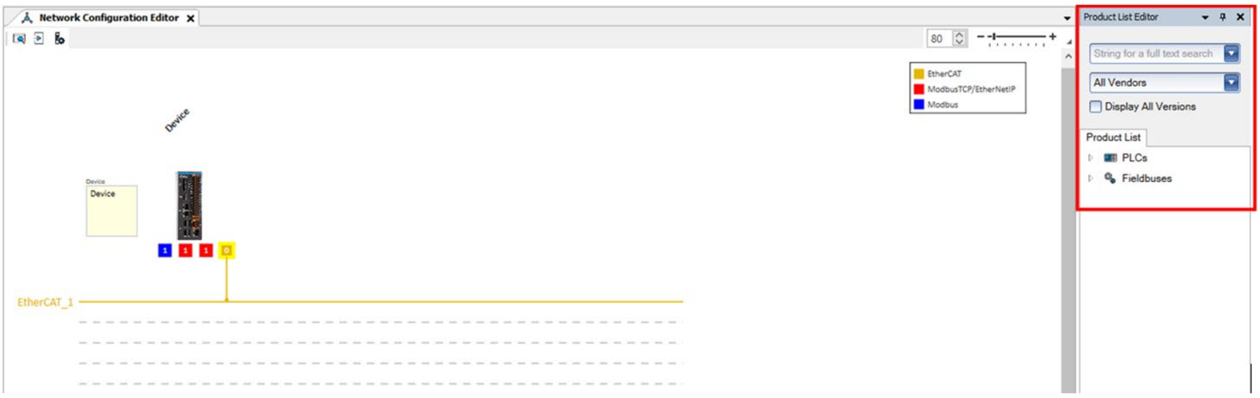
2. Enter **Name** and **Location** in the **New Project** window, and select the desired template to create a project. In this section, Model AX-8 is taken as an example; therefore, we select **Standard project** and then click **OK**.



3. Double-click on **Network Configuration** to continue with EtherCAT settings.

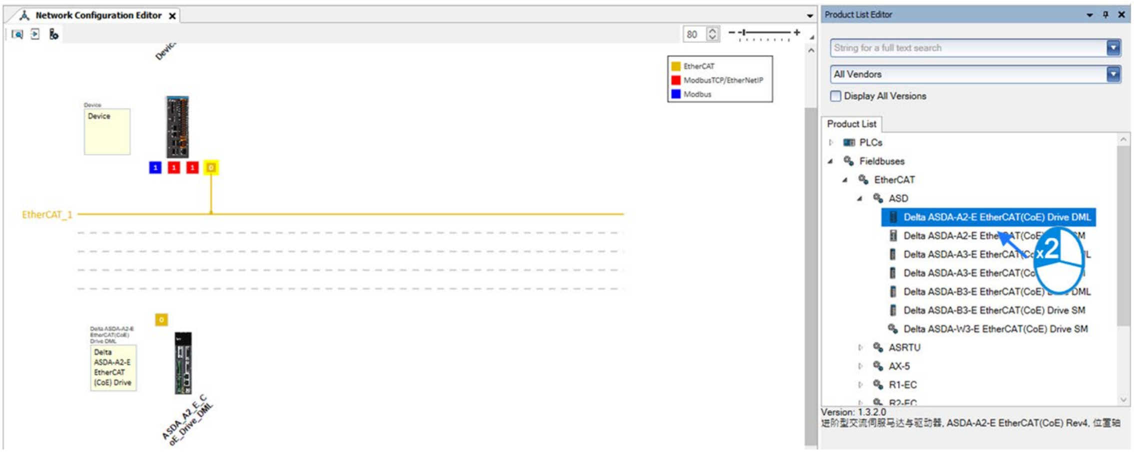


4. After double-clicking, the **Network Configuration Editor** window pops. The slave device can be found and added from the **Product List Editor** on the right.

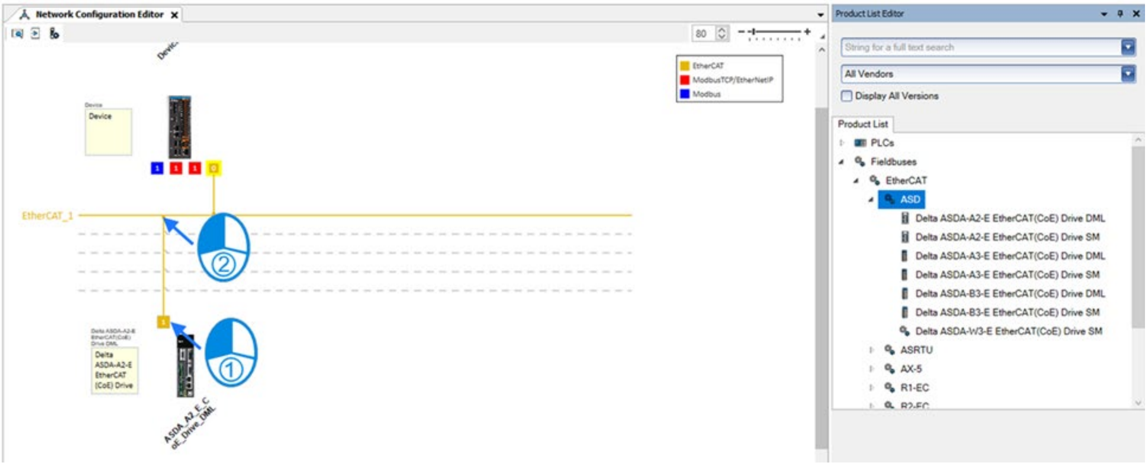


7

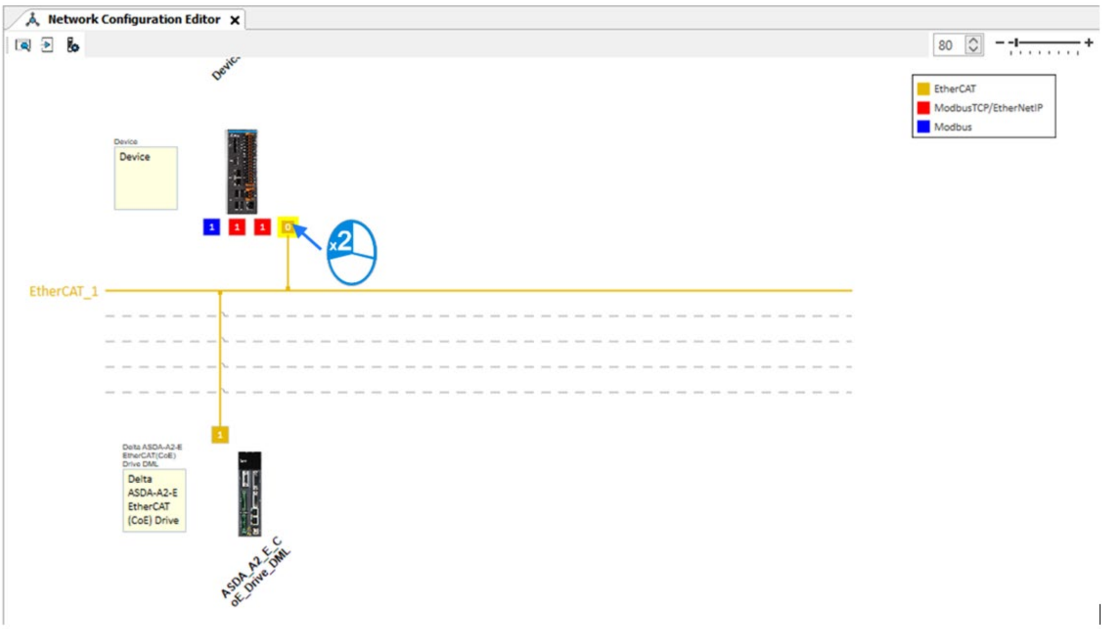
5. Select **Fieldbuses** > **EtherCAT** > **ASD** > **Delta ASDA-A2-E EtherCAT (CoE) Drive DML**, double-click and it will be automatically added to the **Network Configuration Editor**.



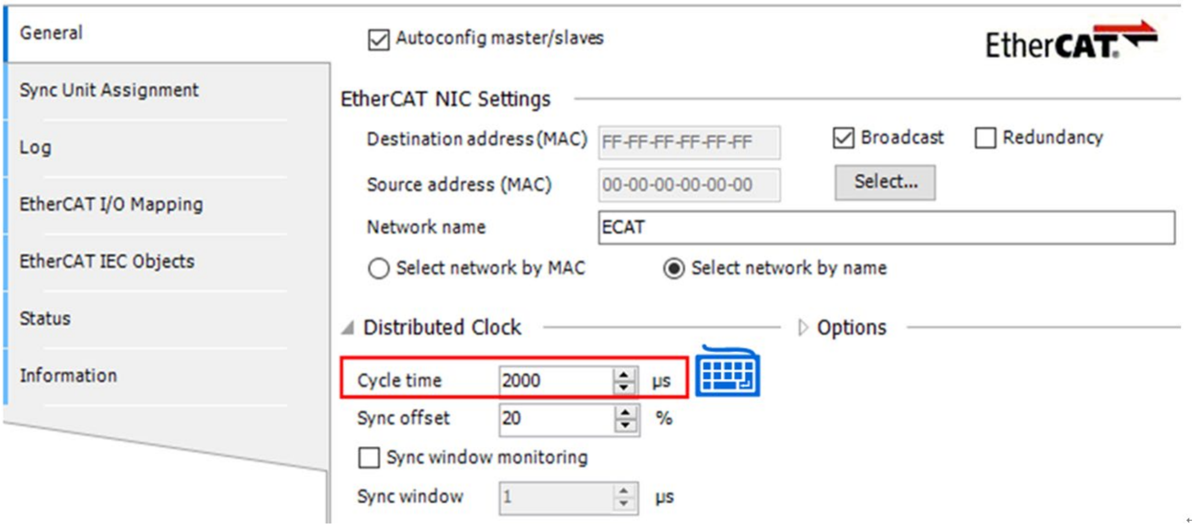
6. Click on the yellow box above the slave station and drag it to the EtherCAT main line to complete the connection configuration between the master station and the slave station.



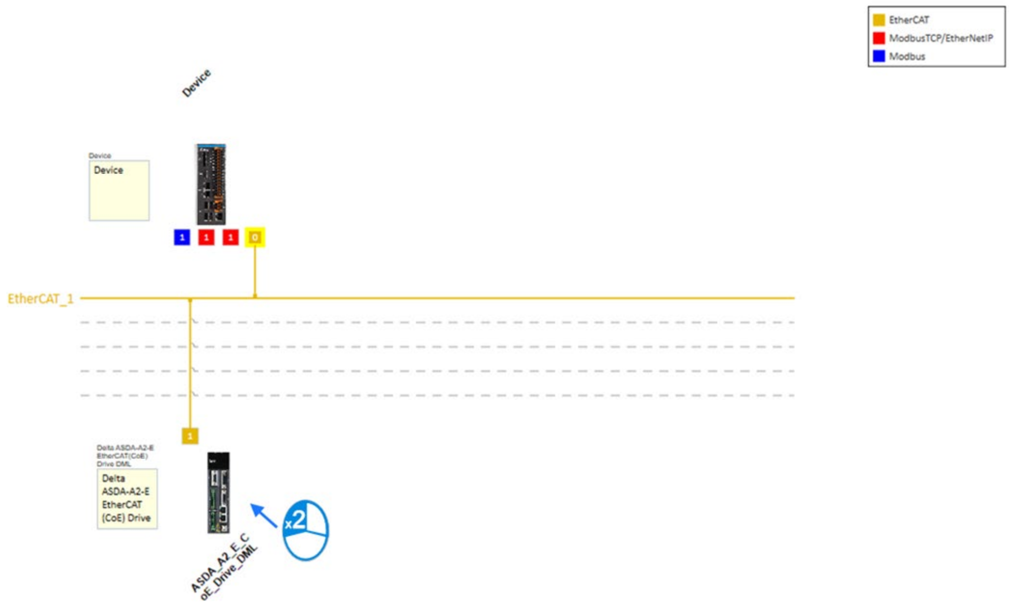
7. Double-click the small yellow box of the master station to set the parameters of the EtherCAT master station. In the master station setting, the EtherCAT synchronization time can be set.

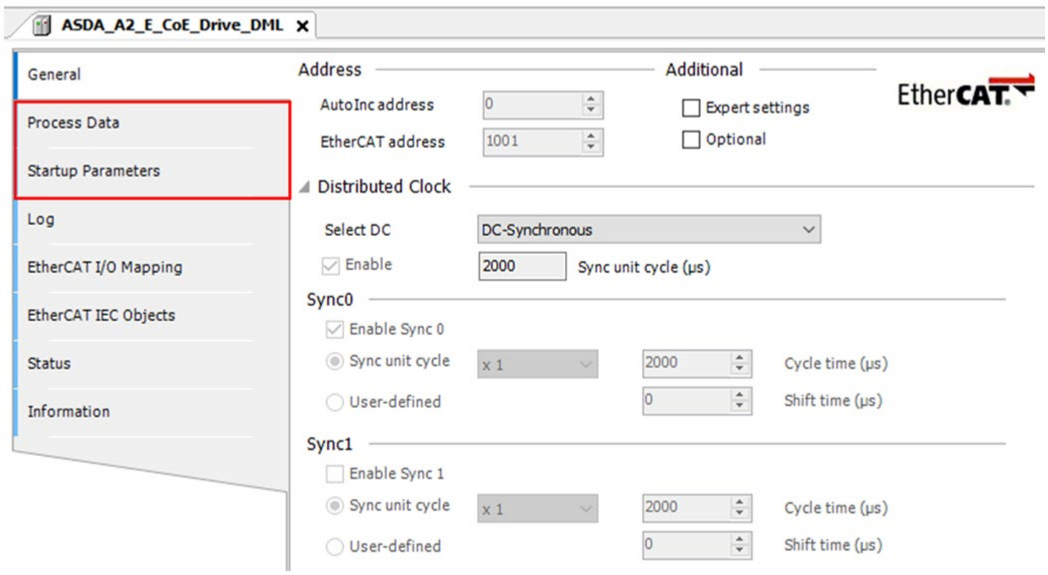


7

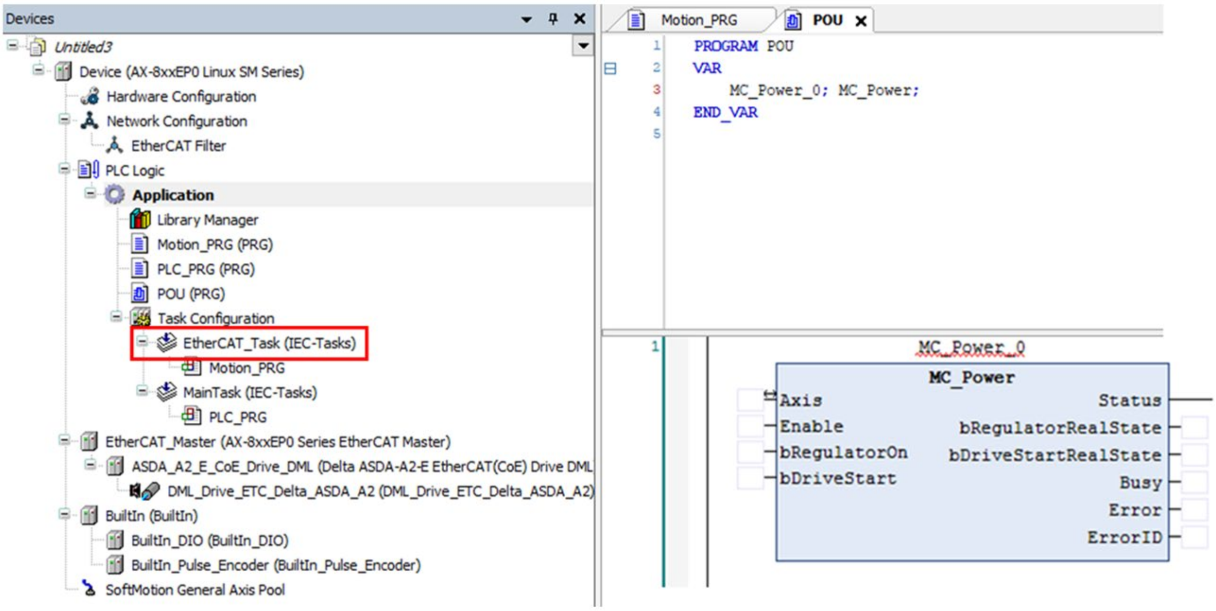


8. Double-click slave device to enter the slave EtherCAT settings. After double-clicking, the slave station related setting page appears, such as station number setting, **Process Data** and **Startup Parameters** setting, etc.





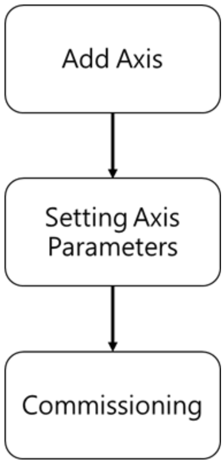
9. You can then start programming. It should be noted that for the motion command part, use the motion function block to write the program in the POU under **EtherCAT_Task** to prevent the motion function block from running normally.



7.3 Commissioning

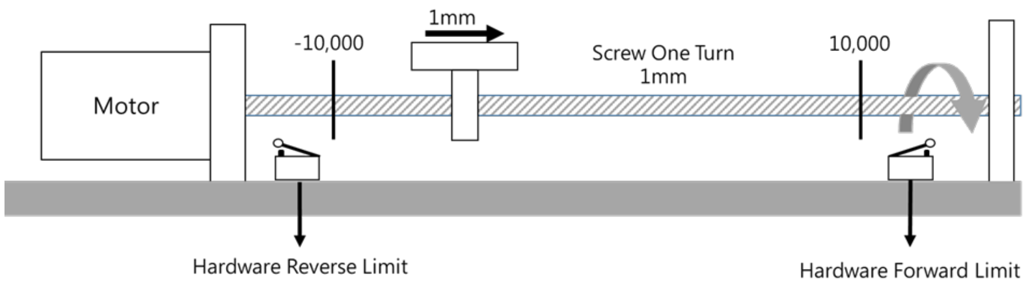
7.3.1 Procedure for Commissioning

The steps to establish a commissioning process are as follows:

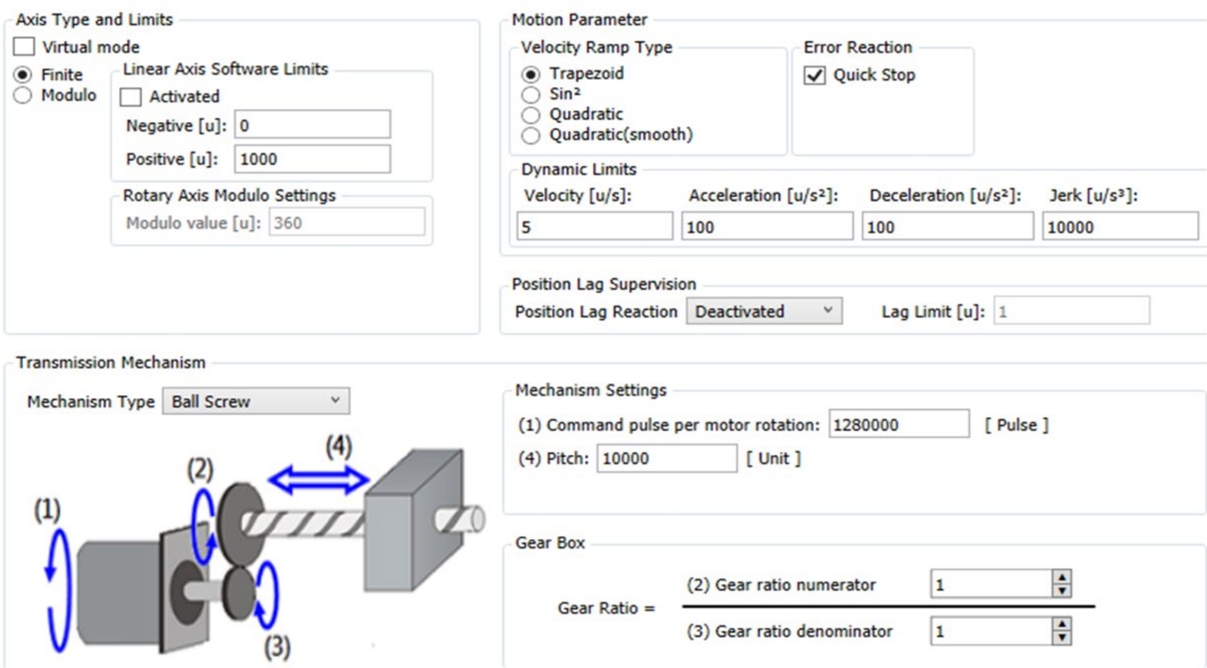


7.3.2 Example of Axis Parameter Settings

Before using software to perform commissioning, axis parameters must be set first. The setting method is illustrated below:



- **Axis configuration screen**



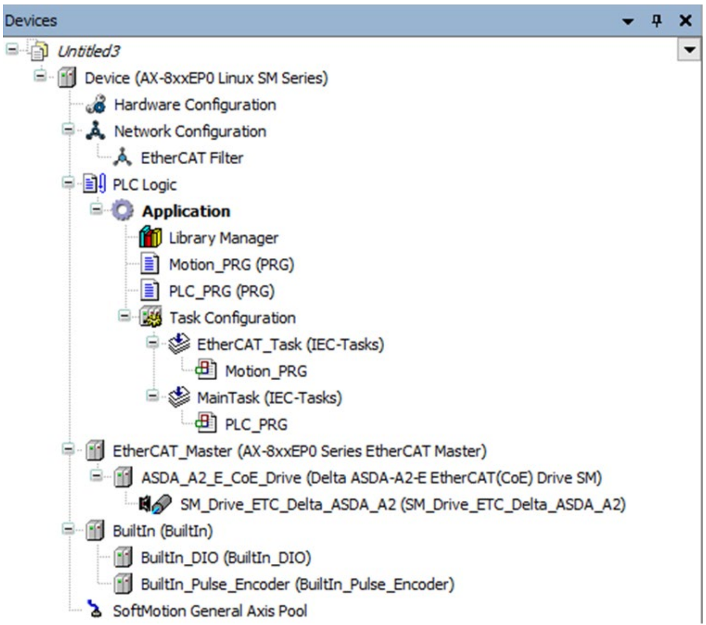
- **Parameter settings**

Parameter Name	Setup
Axis type ①	Linear axis
Command pulse per motor rotation ③	10,000
Pitch ③	1*1
Gear ratio denominator	128*2
Gear ratio numerator	1*2
Software limit_Positive ②	10,000
Software limit_Negative ②	-10,000

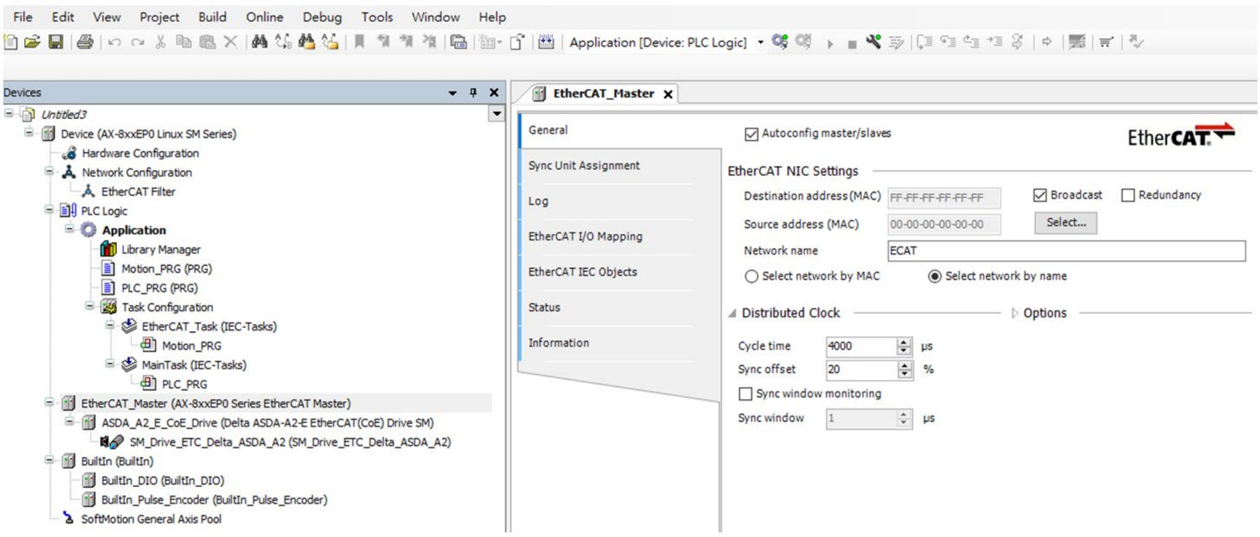
*1: The unit is 1mm. For instance, if you need to move by 1μm, the input parameter would be 0.001.
*2: It is a must to set P1-44 and P1-45 of the servo drive.

7.3.3 Performing Axes Commissioning

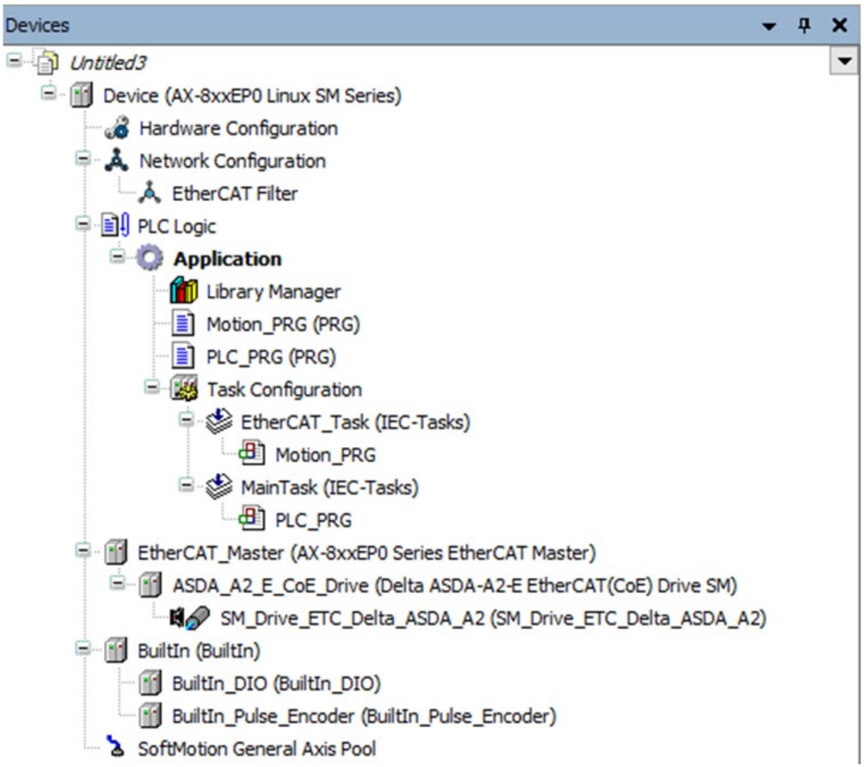
1. Double-click to select **EtherCAT_Master_SoftMotion**.



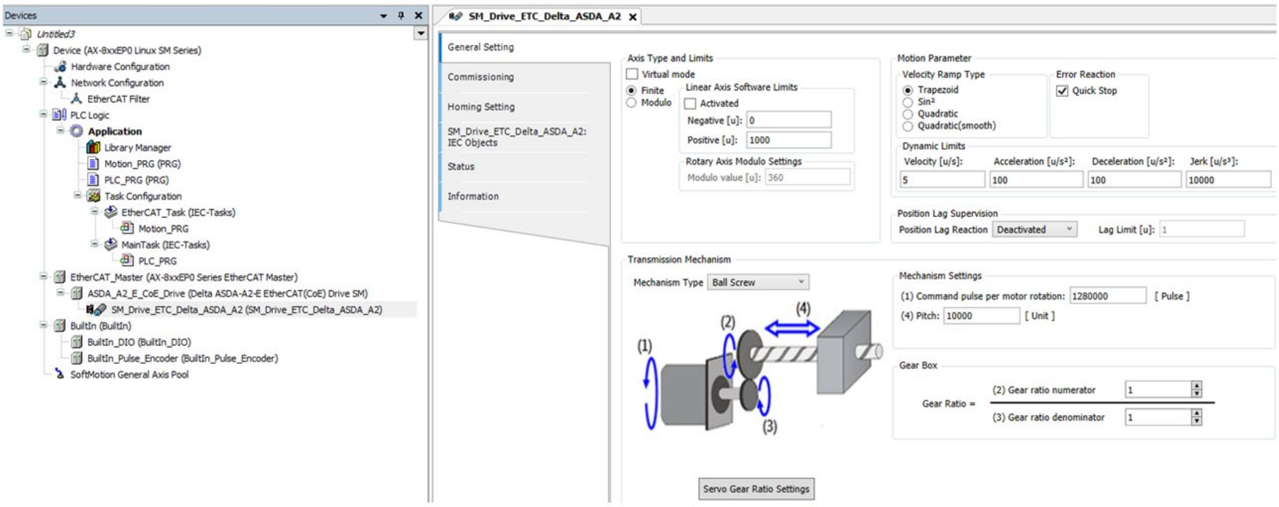
2. Double-click to select **Online Config Mode**.



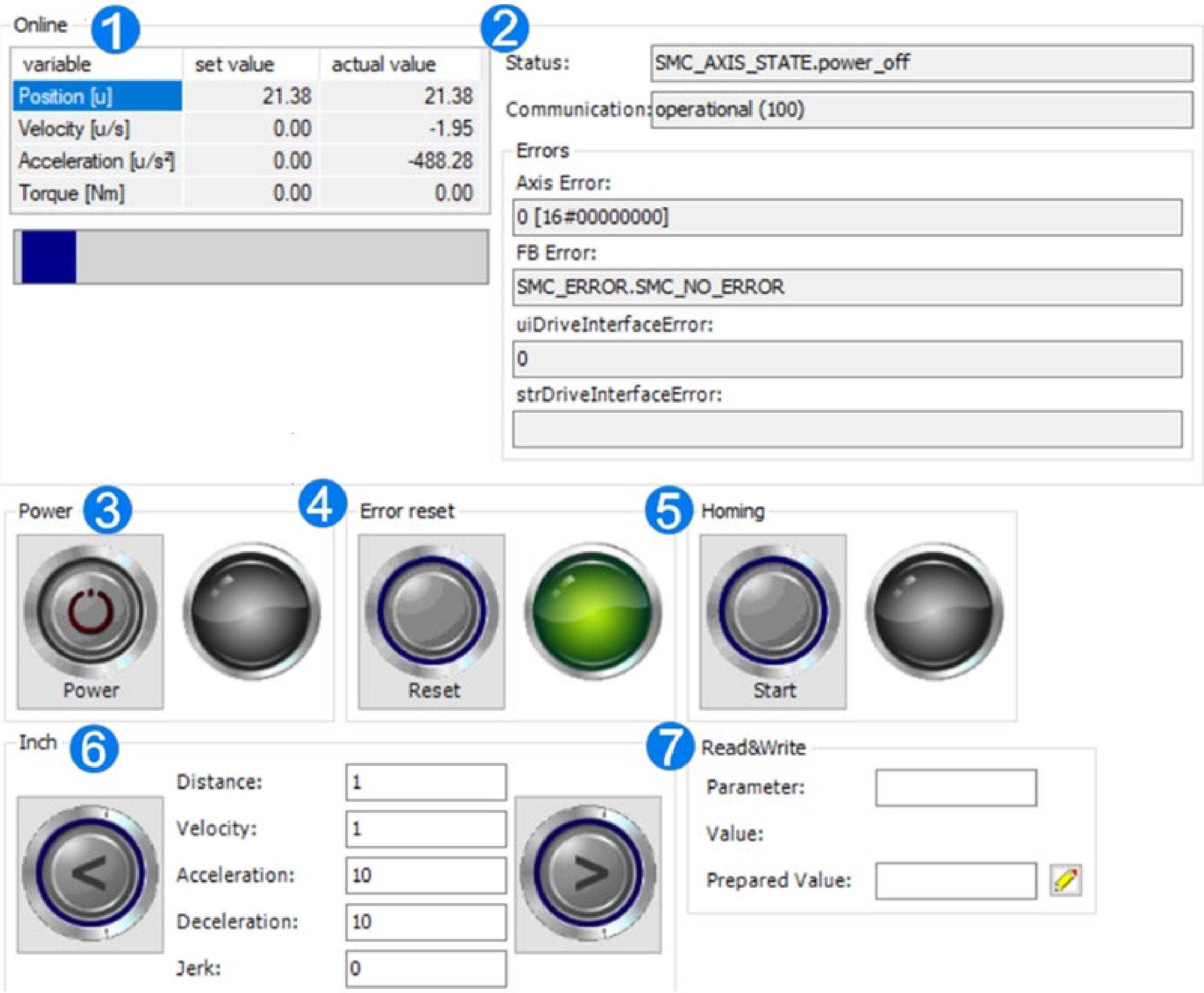
3. After entering the online commissioning, double-click to select **SM_Drive_ETC_Delta_ASDA_A2**.



4. After entering the axis parameter setting page, open the **Commissioning** tab.



- **Display on the Commissioning page**



- ① Information on axis commands

Name	Function
Position [u]	Set position and actual position
Velocity [u/s]	Set value and actual value
Acceleration [u/s^2]	Set value and actual value
Torque [Nm]	Set value and actual value

- ② Axis status and communication status

Name	Function
Status	Axis status
Communication	Communication status

- ③ Axis power: Set power on / off
- ④ Error reset: Clear servo axis error messages
- ⑤ Homing: Set the axis back to the original position

■ ⑥ Inch

Name	Function
Distance	Moving distance
Velocity	Moving velocity
Acceleration	Acceleration rate
Deceleration	Deceleration rate
Jerk	Command value of jerk

■ ⑦ Read & Write:

Read and write the upper axis parameters. To read or modify Object Dictionary, please follow the steps:

- Read and write the parameter 0x6098 in object dictionary.

  $16\#\boxed{1}\boxed{6098}\boxed{00}$

  1 = fixed number

  6098 = the prameter to be read and written

  00 = sub for the parameter

1. Convert 0x1609800 to decimal number as 23,107,584.
2. Turn 23,107,584 into -23,107,584.
3. Enter -23,107,584 in **Parameter** field to read the parameter “0x6098”.

7.4 Motion Control Device

7.4.1 Axis Setting

Motion control equipment is mainly used to set axis parameters. In most applications, user can set axis parameters in DIADesigner-AX. DIADesigner-AX provides user with a convenient editing environment. In this editing environment, the axis parameter setting of motion control is defined as its single data structure. This data structure contains multiple related parameters.

7.4.2 Axis Introduction

In a motion control system, the object used to perform motion control is called an axis, which contains physical servo drives, encoders and virtual servo drives. The axis types are listed in the table below.

Types	Function
Positioning axis*1	Perform basic positioning control such as absolute positioning, relative positioning etc through EtherCAT communication.
Velocity axis*1	Perform velocity control and torque control through EtherCAT communication (as seen in CIA 402 Velocity Mode).
Synchronous axis*2	Perform servo control through EtherCAT communication, including basic positioning control as well as synchronous motion control such as electronic cams.
Pulse type servo axis	Perform physical servo control through pulses.
Virtual axis	Perform motion control commands without using physical servo control.
Encoder axis	Use physical encoder (SSI or incremental encoder) as feedback signals.
Virtual encoder axis	Used only in a program without encoders.

*1:

Positioning axis and velocity axis need to use DL_MotionControlLight library.

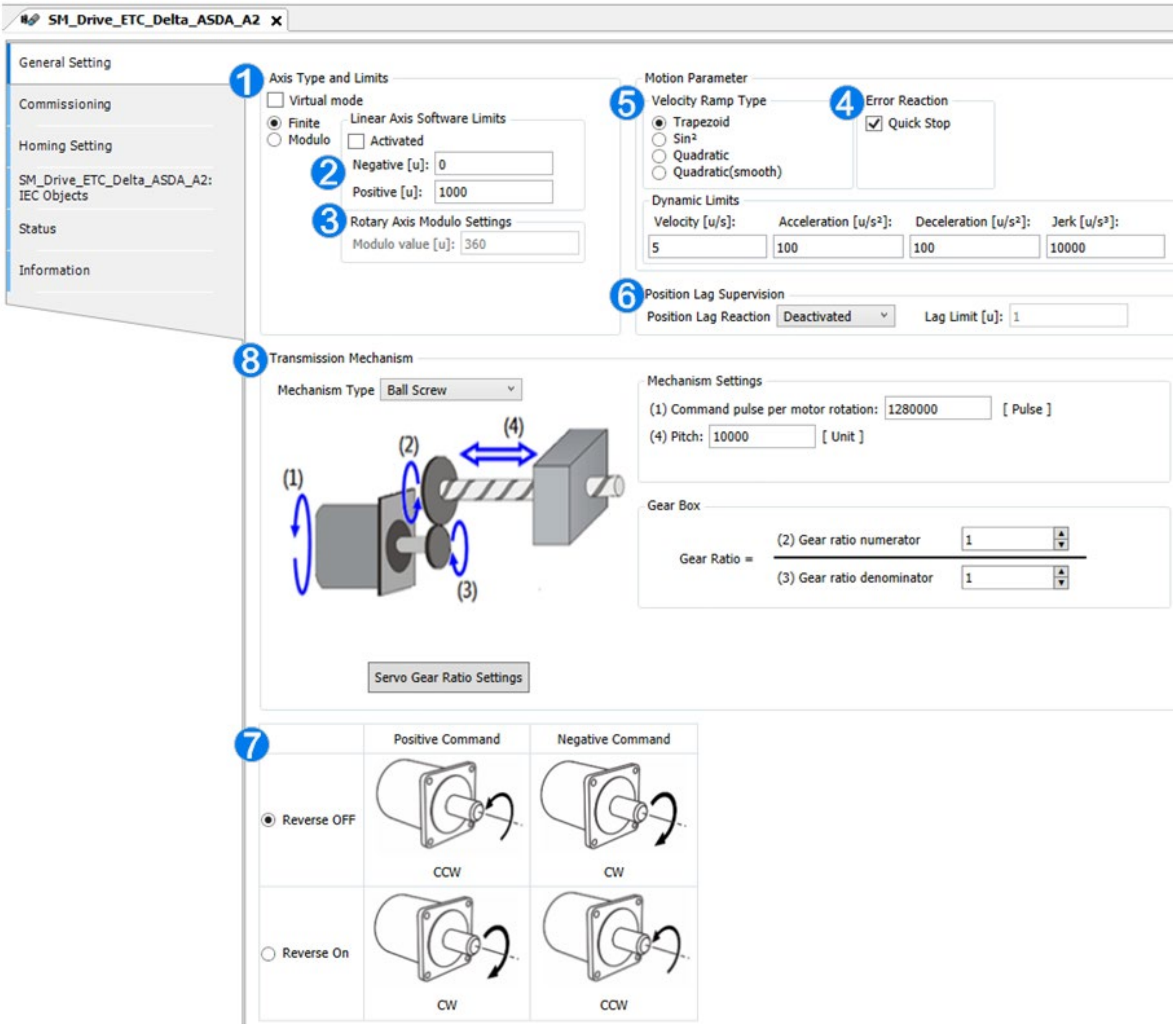
When the number of EtherCAT axes used by AX-8 exceeds 64 and the Soft Motion version is below V4.7.0.0, the MAX_MAILBOX_CHANNELS and MAX_SDO_Channels parameters in the Library (IODrvEtherCat → ETC_Parameter) must be changed to 128.

*2: Synchronous axis need to be used with DL_MotionControl and SM3_Basic library.

7.4.2.1 Introduction to Axis Parameters

A servo axis is established with the corresponding axis parameters attendant. The following table lists its related instructions:

- **Synchronous Axis**



- ① Axis Type and Limits

Name	Function
Virtual	Activate virtual axis.
Finite / Modulo	Set to be linear axis or rotary axis.

- ② Linear Axis Software Limits

Name	Function
Activated	Activate software limit (only linear axis is supported).
Negative [u]	Reverse software limit.
Positive [u]	Forward software limit.

■ ③ Rotary Axis Modulo Setting

Name	Function
Modulo value [u]	Set the area of rotation for a turn (only for rotary axis).

■ ④ Error Reaction

Name	Function
Quick Stop	Axis emergency stop
Deceleration [u/s^2]	Axis deceleration to stop (only valid when Quick Stop is not activated)

■ ⑤ Velocity Ramp Type

Name	Function
Trapezoid / Sin^2 / Quadratic / Quadratic (Smooth)	Set axis motion trajectory

■ ⑥ Position Lag Supervision

Name	Function
Positon Lag Reaction	Set the reaction for position lag.
Lag Limit [u]	Set the value of lag limit.

■ ⑦ Positive / Negative Command

Name	Function
Reverse OFF / Reverse ON	Enable or disable reverse function for positive / negative command.

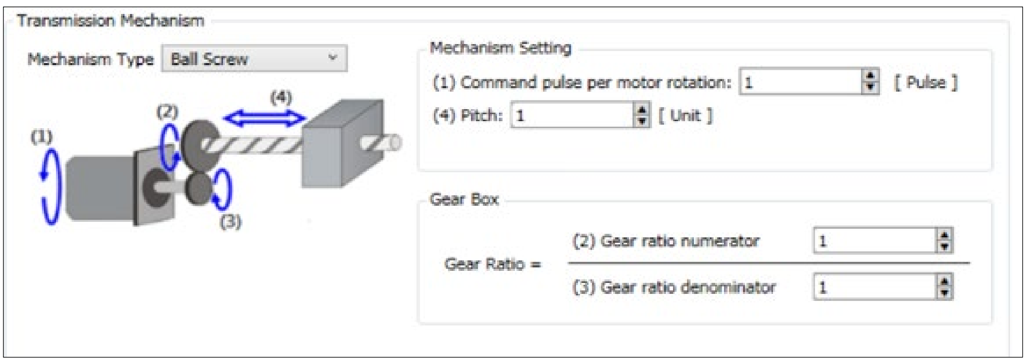
■ ⑧ Transmission Mechanism

- Servo Gear Ratio Setting

Name	Function
Unit Numerator	Numerator factor of the electronic gear unit
Unit Denominator	Denominator factor of the electronic gear unit

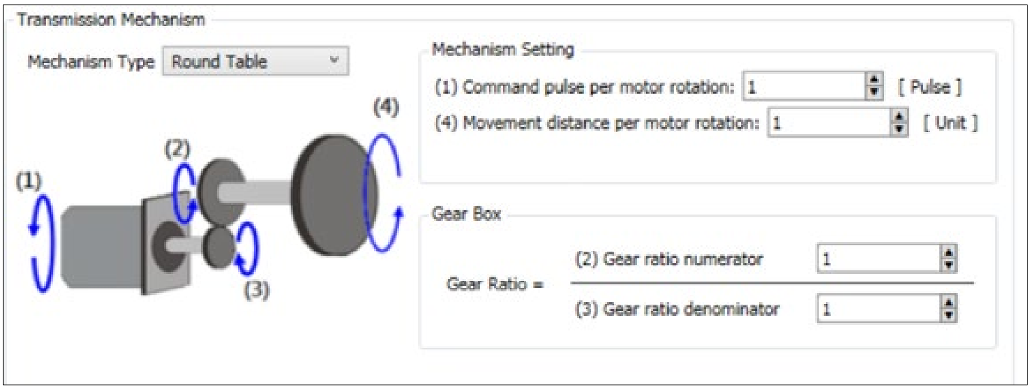
The introductions of different mechanisms are given below:

- Ball Screw



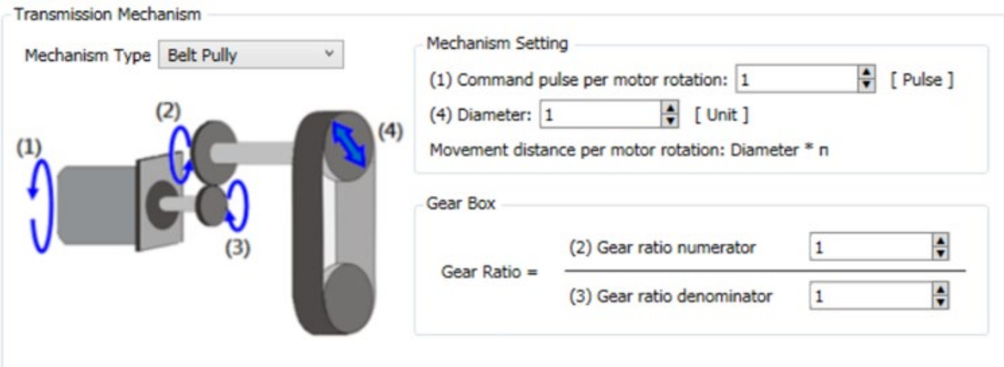
Name	Function
(1) Command pulse per motor rotation	Command pulse value per motor rotation
(4) Pitch	Distance between screw threads
(2) Gear ratio numerator	Numerator of gear ratio
(3) Gear ratio denominator	Denominator of gear ratio

- Round Table



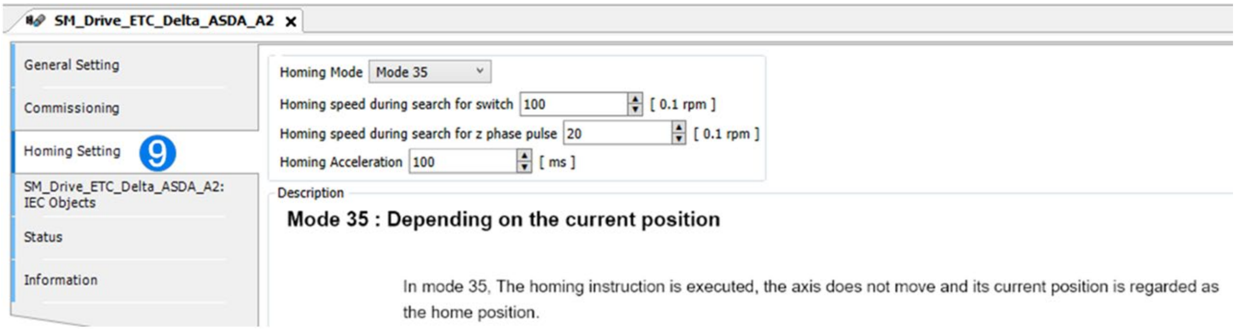
Name	Function
(1) Command pulse per motor rotation	Command pulse value per motor rotation
(4) Movement distance per motor rotation	Distance of the movement per motor rotation
(2) Gear ratio numerator	Numerator of gear ratio
(3) Gear ratio denominator	Denominator of gear ratio

- Belt Pulley



Name	Function
(4) Diameter (moving distance motor rotation: diameter *n)	Diameter Distance of the movement per motor rotation: diameter*n
(1) Command pulse per motor rotation	Command pulse value per motor rotation
(2) Gear ratio numerator	Numerator of gear ratio
(3) Gear ratio denominator	Denominator of gear ratio

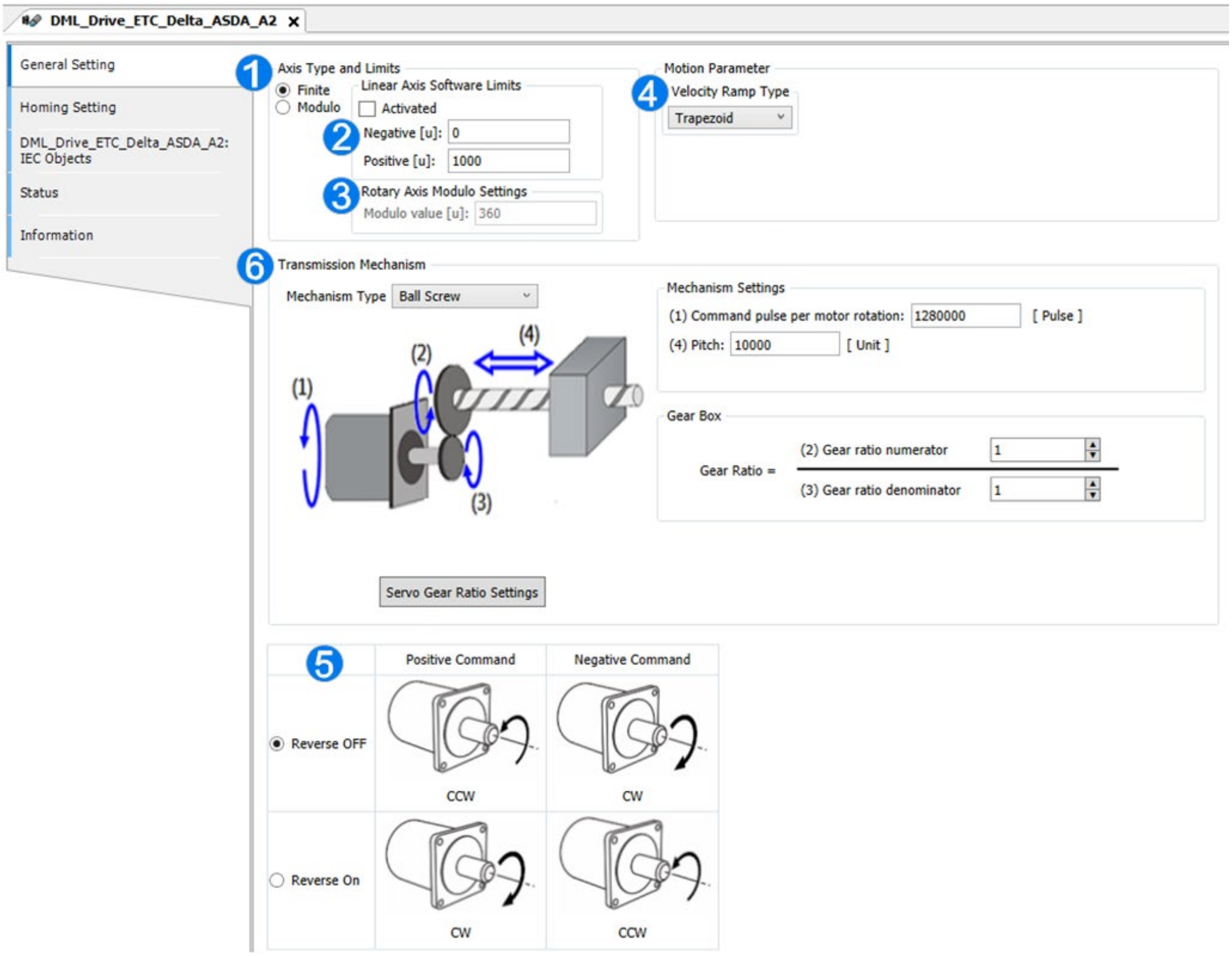
■ ⑨ Homing Setting



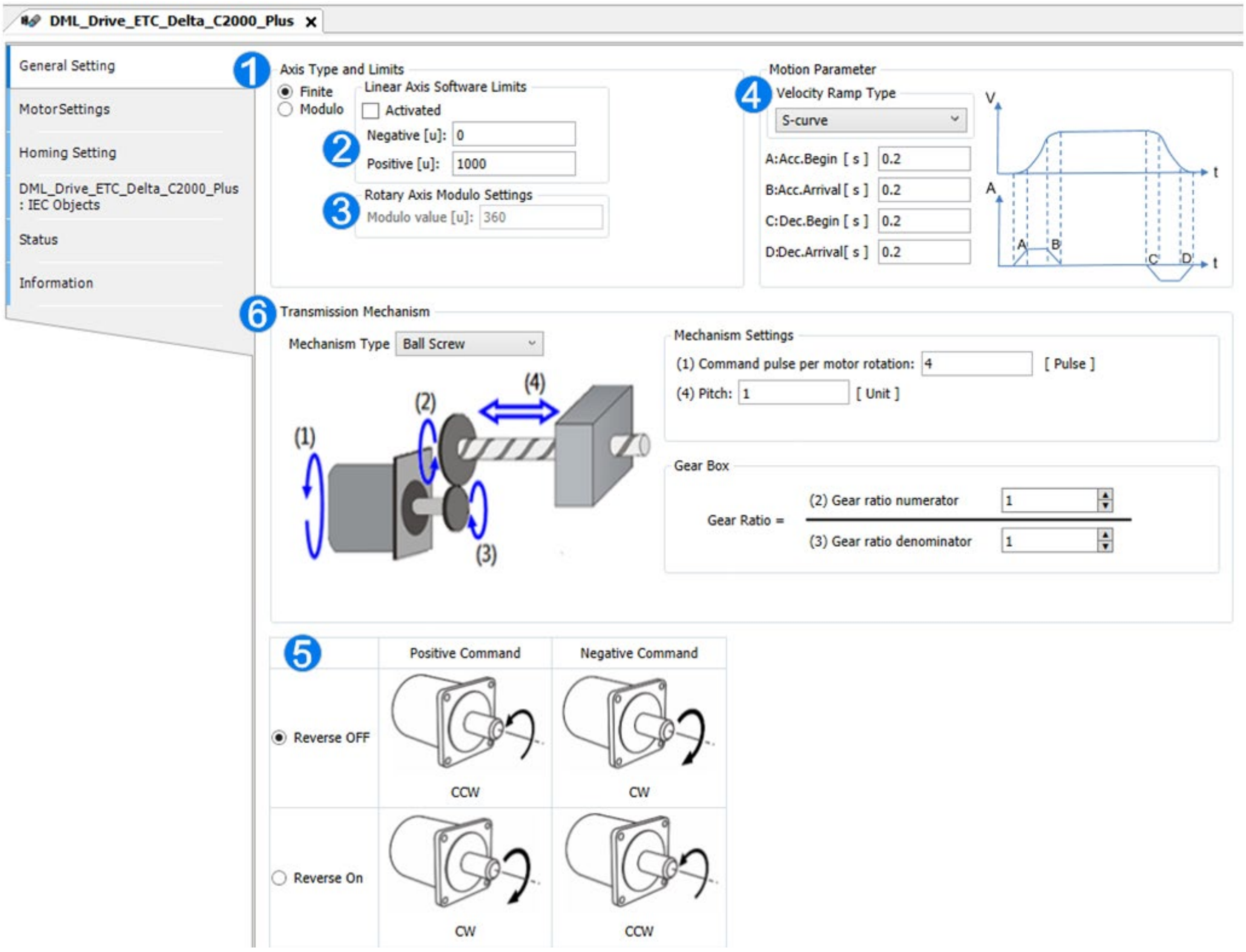
Name	Function
Homing Mode	Configure homing mode setting.
Homing speed during search for switch	Set the homing speed during search for switch.
Homing speed during search for z phase pulse	Set the homing speed during search for Z phase pulse.
Homing Acceleration	Set the homing acceleration rate.

● **Positioning axis**

❖ Positioning axis - Delta servo motors



❖ Positioning axis - Delta drives



■ ① Axis Type and Limits

Name	Function
Finite / Modulo	Set to be linear axis or rotary axis.

■ ② Linear Axis Software Limits

Name	Function
Activated	Activate software limit (only linear axis is supported).
Negative [u]	Reverse software limit.
Positive [u]	Forward software limit.

■ ③ Rotary Axis Modulo Setting

Name	Function
Modulo value [u]	Set the area of rotation for a turn (only for rotary axis).

- ④ Velocity Ramp Type

  ❖ Delta Servo Motors

Name	Function
Trapezoid / Sin^2	Set the motion trajectory of axes

  ❖ Delta drives

Name	Function
Trapezoid / S-Curve	Set the motion trajectory of axes
A：Acc.Begin	S acceleration start time setting 1 (s)
B：Acc.Arrival	S acceleration arrival time setting 2 (s)
C：Dec.Begin	S deceleration start time setting 1 (s)
D：Dec.Arrival	S deceleration arrival time setting 2 (s)

- ⑤ Positive / Negative Command

Name	Function
Reverse OFF／ON	Enable or disable reverse function for positive / negative command.

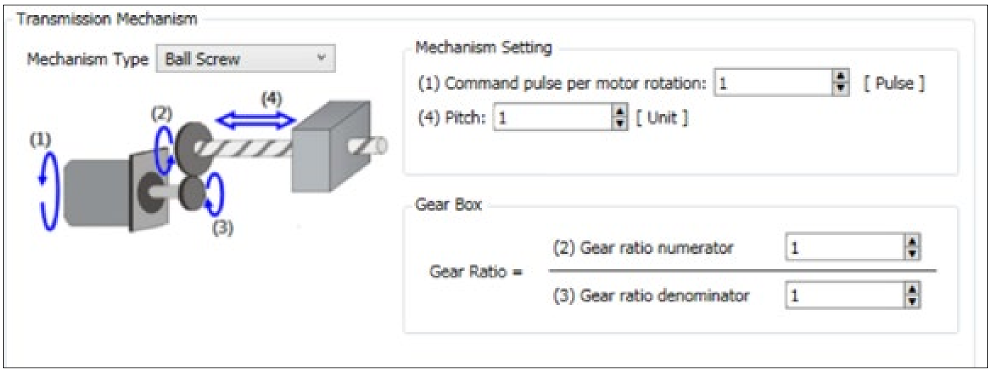
- ⑥ Transmission Mechanism

  - Servo Gear Ratio Setting

Name	Function
Unit Numerator	Numerator factor of the electronic gear unit
Unit Denominator	Denominator factor of the electronic gear unit

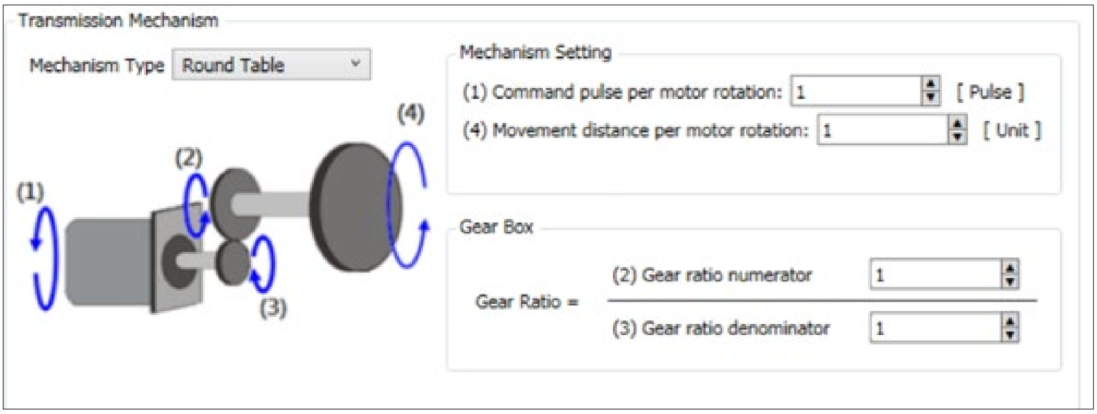
The following columns are introduced for different mechanisms:

  - Ball Screw



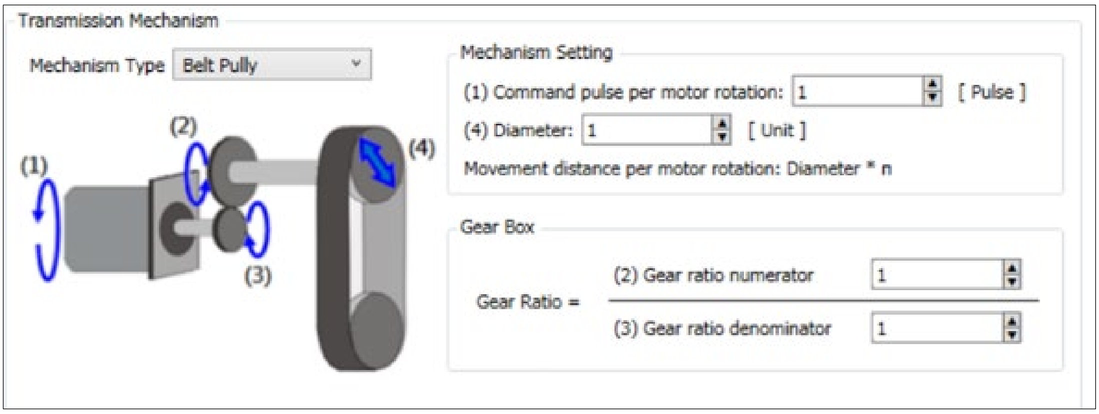
Name	Function
(1) Command pulse per motor rotation	Command pulse value per motor rotation
(4) Pitch	Distance between screw threads
(2) Gear ratio numerator	Numerator of gear ratio
(3) Gear ratio denominator	Denominator of gear ratio

- Round Table



Name	Function
(1) Command pulse per motor rotation	Command pulse value per motor rotation
(4) Movement distance per motor rotation	Distance of the movement per motor rotation
(2) Gear ratio numerator	Numerator of gear ratio
(3) Gear ratio denominator	Denominator of gear ratio

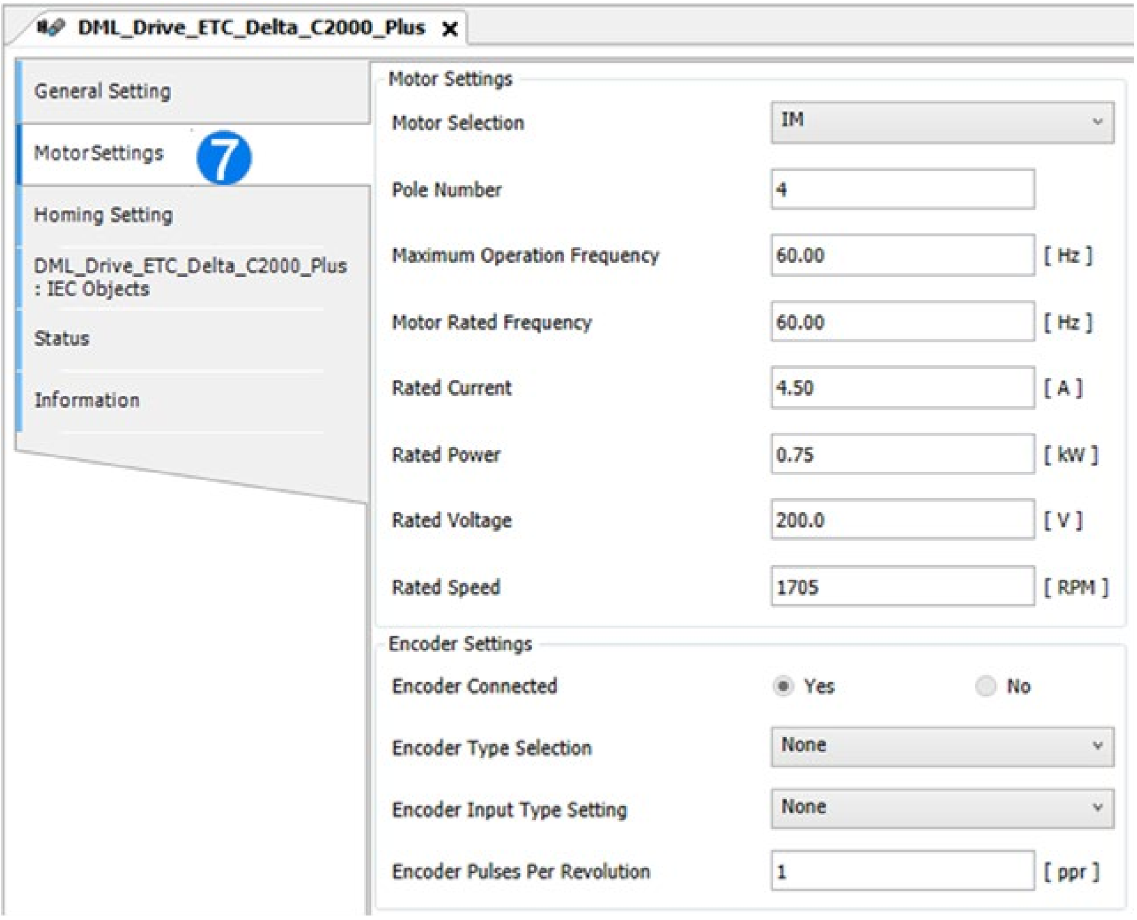
- Belt Pulley



Name	Function
(1) Command pulse per motor rotation	Command pulse value per motor rotation
(4) Diameter (moving distance motor rotation: diameter *n)	Diameter Distance of the movement per motor rotation: diameter*n
(2) Gear ratio numerator	Numerator of gear ratio
(3) Gear ratio denominator	Denominator of gear ratio

7

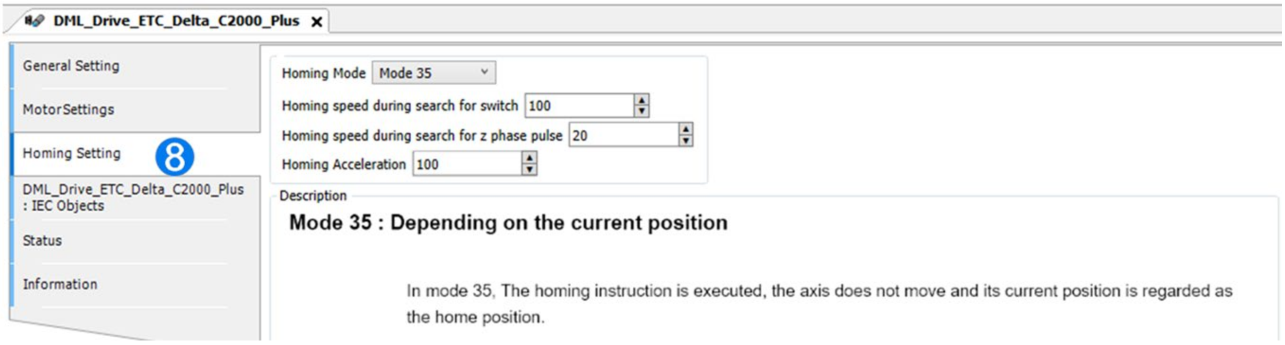
■ ⑦ Motor Settings—settings for Delta drives



Name	Function
Motor Selection	Set the motor type
Pole Number	Set the number of motor poles
Maximum Operation Frequency	Set the maximum operation frequency of the motor
Motor Rated Frequency	Set the motor rated frequency
Rated Current	Set the motor rated current current
Rated Power	Set the motor rated power
Rated Voltage	Set the motor rated voltage
Rated Speed	Set the motor rated speed
Encoder Connected	Set whether to start the encoder
Encoder Type Selection	Set the type of encoder
Encoder Input Type Setting	Set the the input type of encoder
Encoder Pulses Per Revolution	Set the number of encoder pulses per revolution

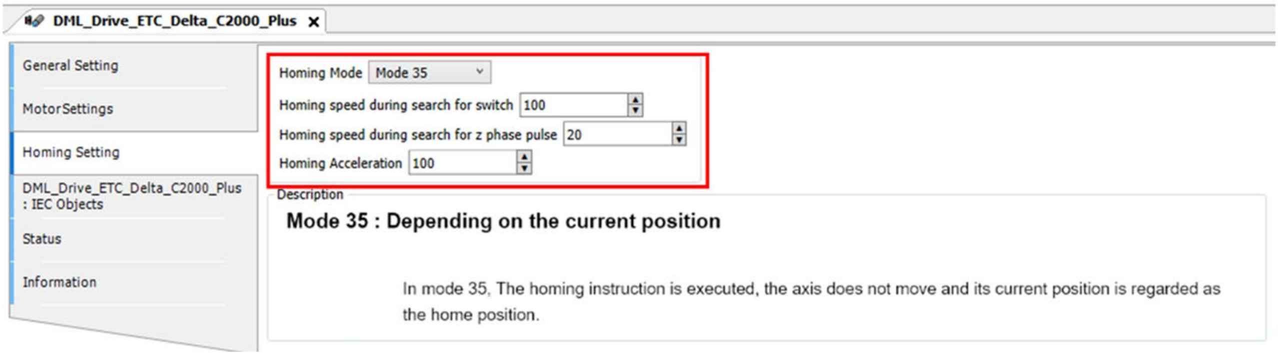
■ ⑧ Homing Setting

❖ Delta servo motors



❖ Delta drives

(Supported models: C2000Plus, CH2000, MH300, MS300; DDF V1.0.1.0 (inclusive) and above support this interface)

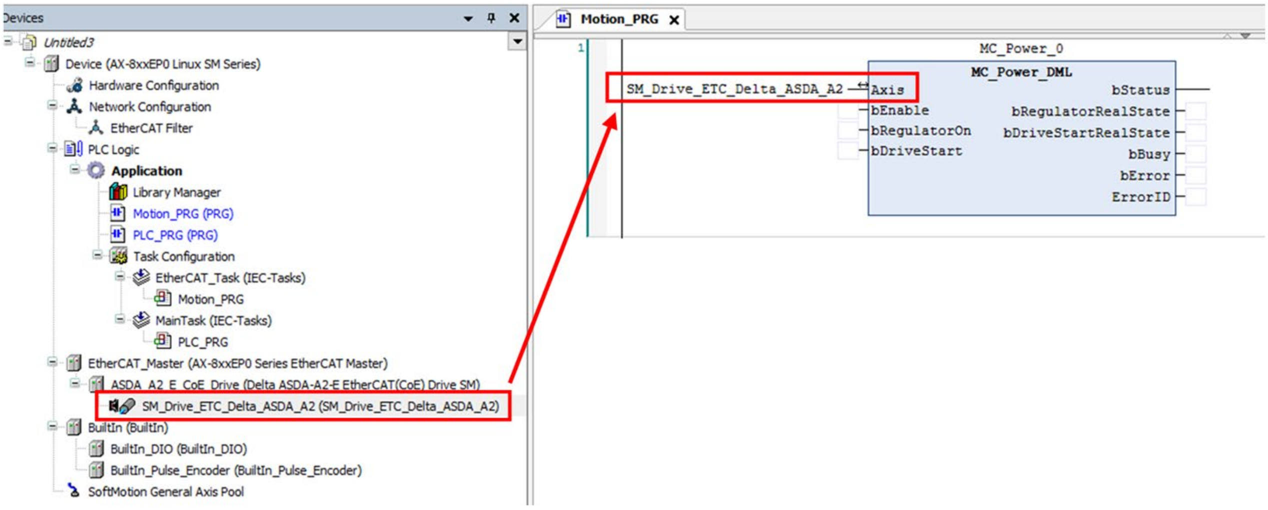


Name	Function
Homing Mode	Configure homing mode setting.
Homing speed during search for switch	Set the homing speed during search for switch.
Homing speed during search for z phase pulse	Set the homing speed during search for Z phase pulse.
Homing Acceleration	Set the homing acceleration rate.

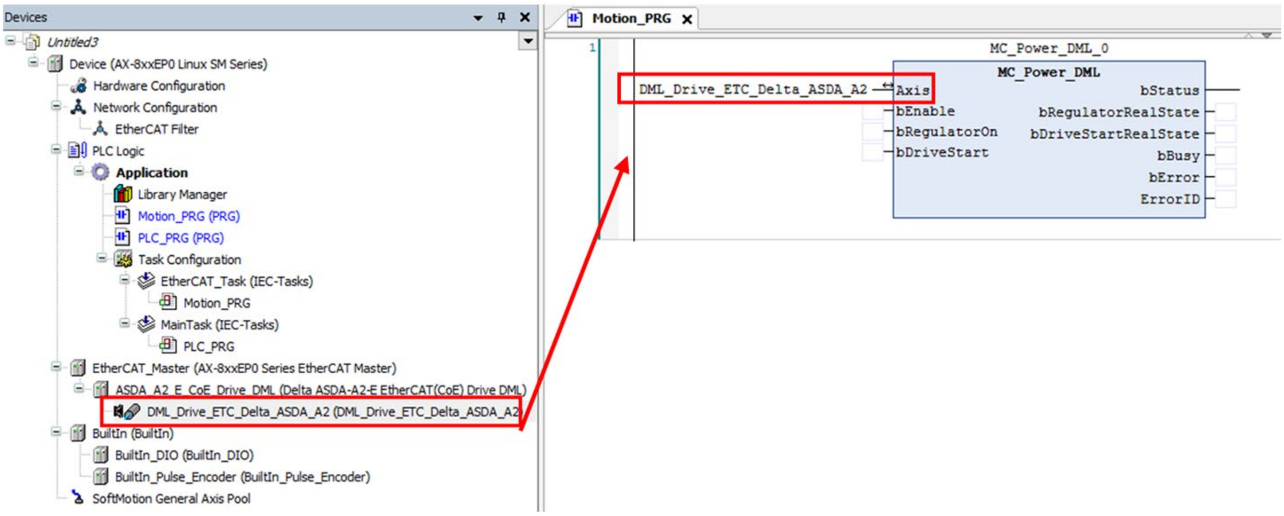
7.4.2.2 Axis Application in the Program

When a servo axis is added to the project, the system will automatically generate its servo axis name (the user can also modify its name) and input the name to the function block.

- **Synchronous axis**

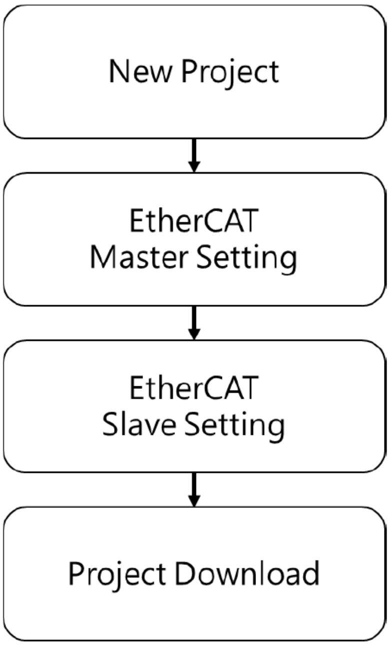


- **Positioning axis**

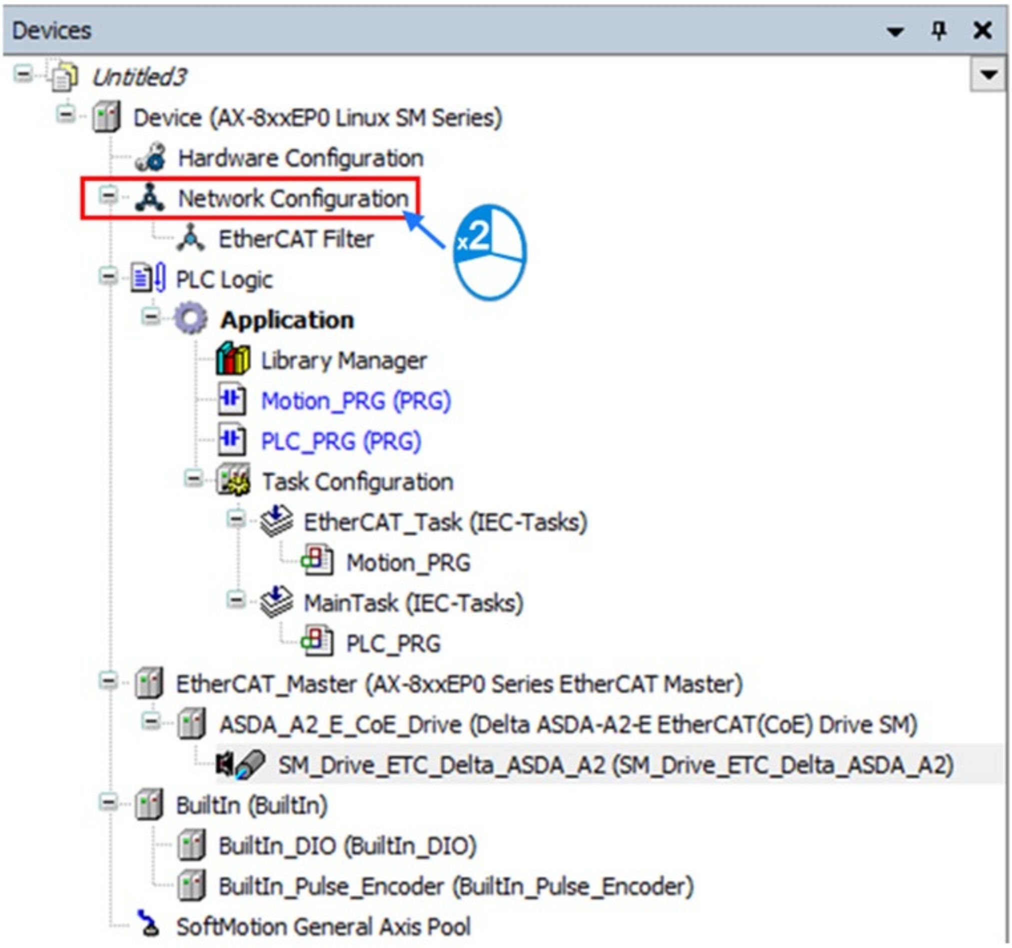


7.4.3 Procedure for Single-axis Configuration

The axis setting procedure is shown in the figure below. For details of PLC project creation, refer to **section 7.2** of this manual.

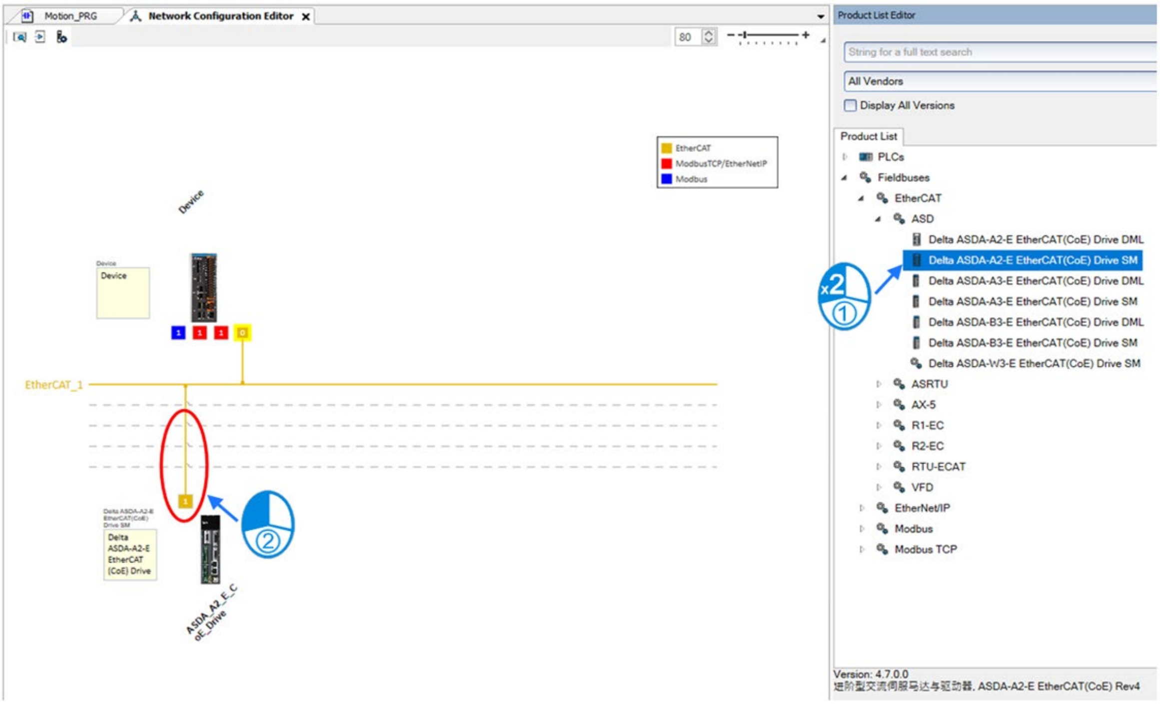


1. Open the project, click on **Network Configuration**.



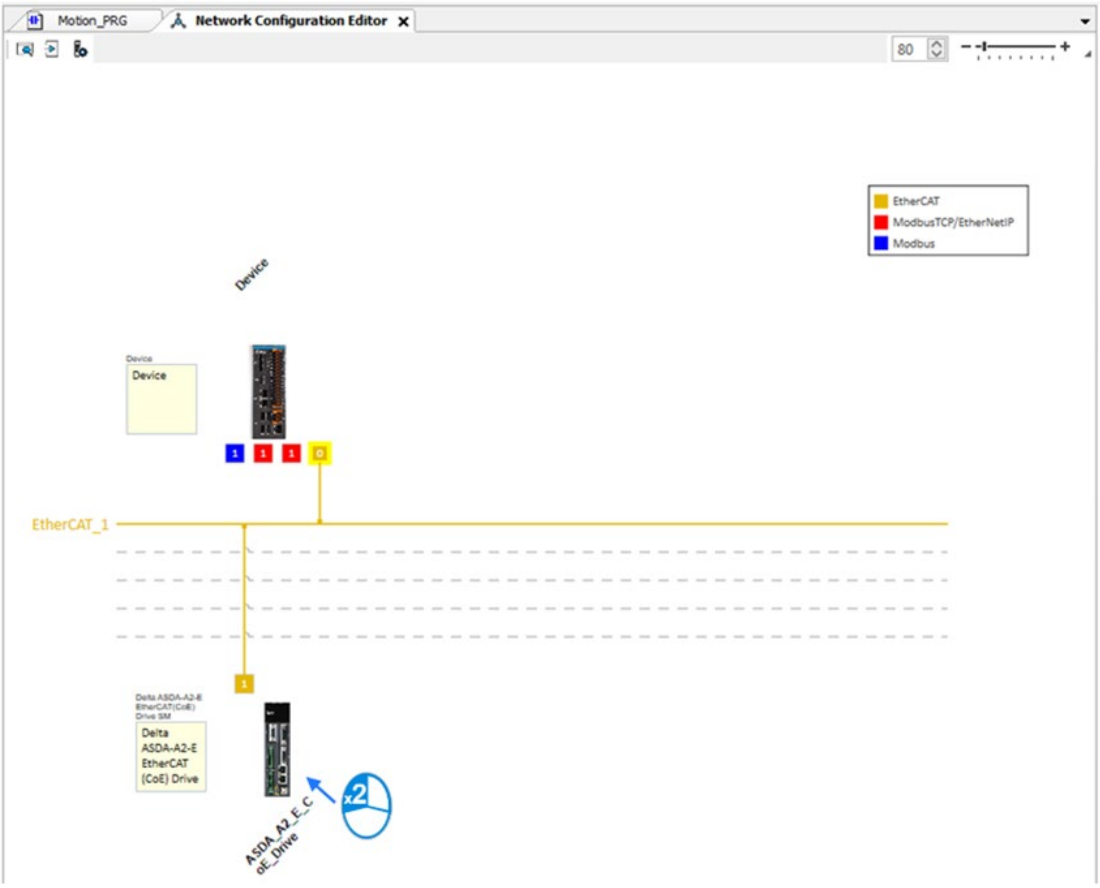
7

2. After entering the **Network Configuration** page, click **Delta ASDA-A2-E EtherCAT (CoE) Drive Rev4_SM**[*1] and then wire to the yellow line above.



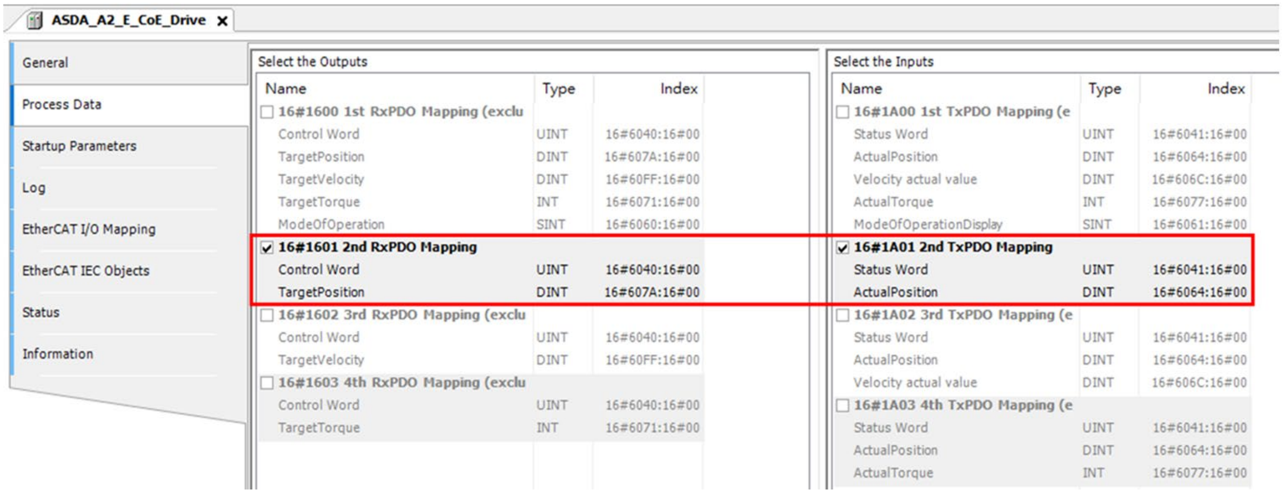
*1: Delta ASDA-A2-E EtherCAT (CoE) Drive Rev4_SM is a synchronous axis. If you need to add a new positioning axis, select Delta ASDA-A2-E EtherCAT (CoE) Drive Rev4_DML. The subsequent steps are the same.

3. After the axis is created, double-click the slave device to be set.



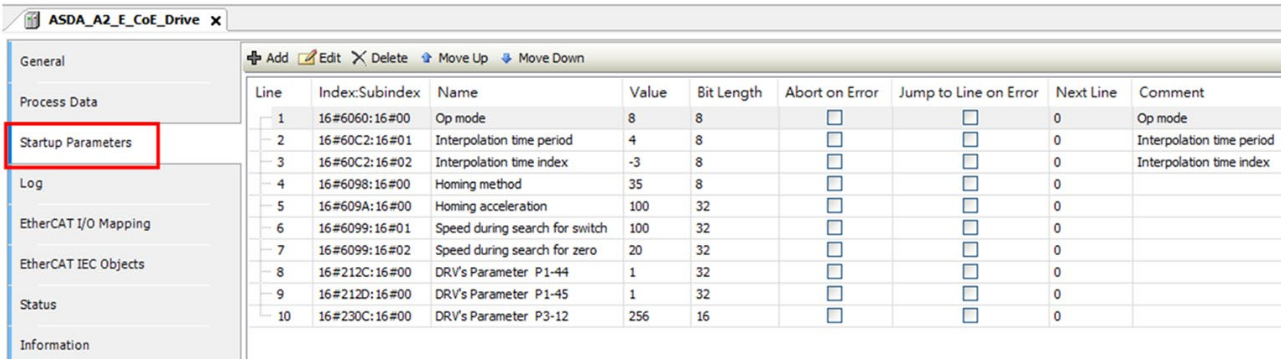
7

4. Switch to the page **Process Data** and set up its PDO mapping group. The default setting for Delta ASDA-A2 is the 2nd group, which can operate normally with most function blocks. If additional groups or PDO parameters are required, please refer to the content regarding function blocks in the **AX Series Motion Controller Manual**.

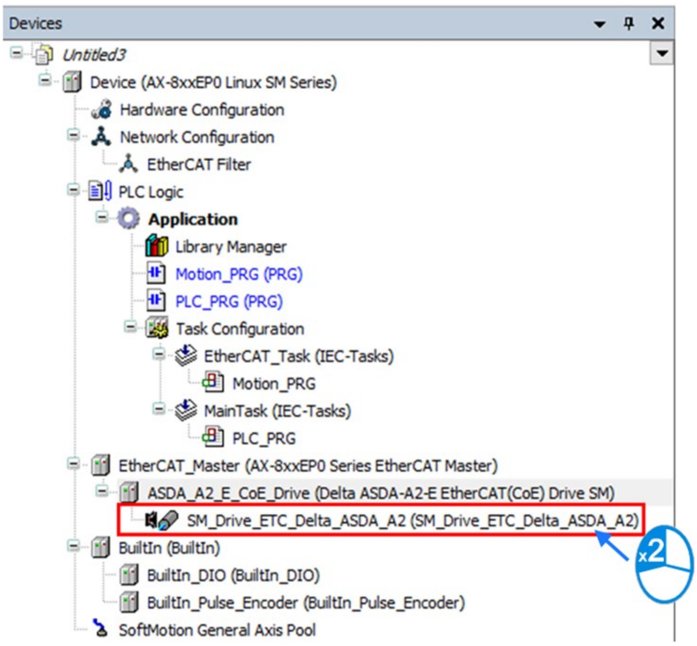


5. Initialize EtherCAT communication

   Where it is necessary to write fixed values for the required Object Dictionary after each initialization, go to the **Startup Parameters** page to set up.

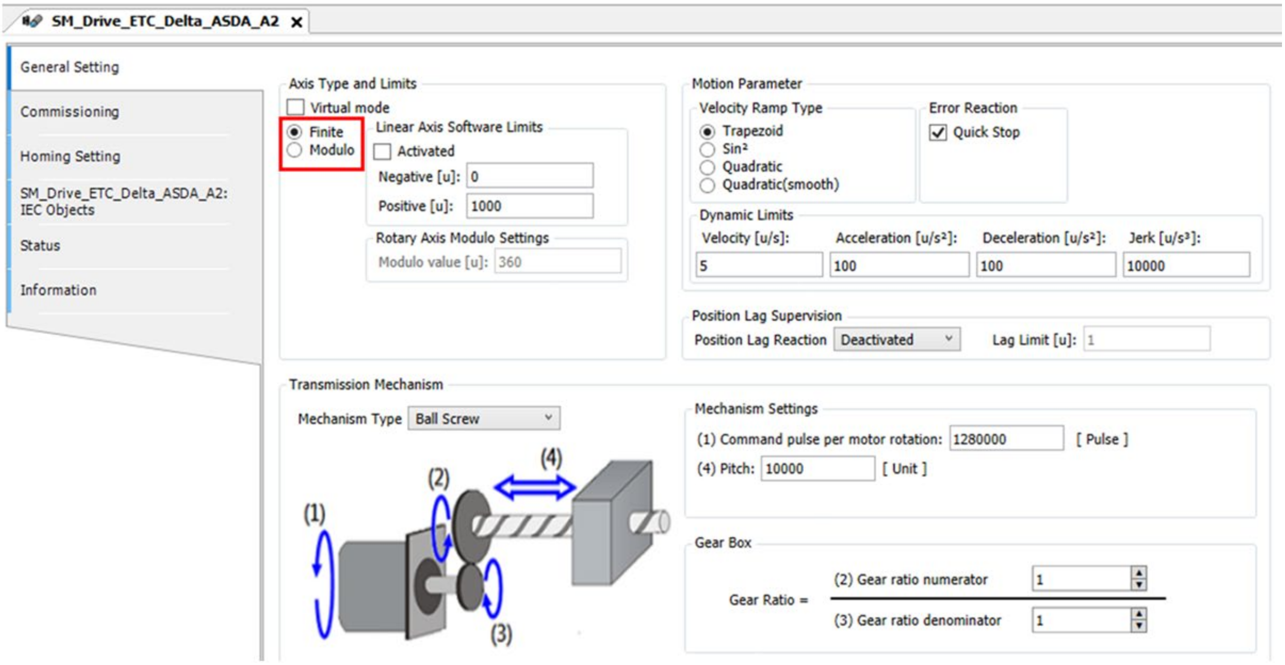


6. After completing related settings of axis communication, double-click **SM_Drive_ETC_Delta_ASDA_A2**.

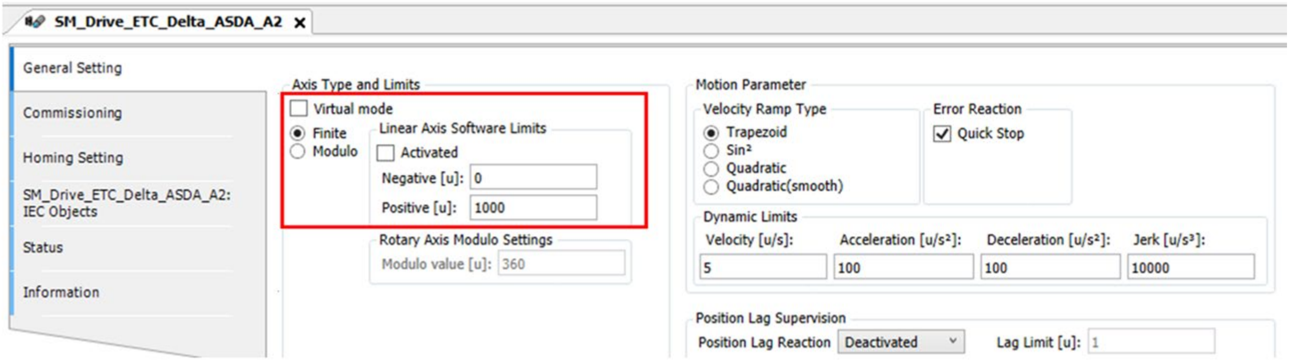


7. Axis setting page

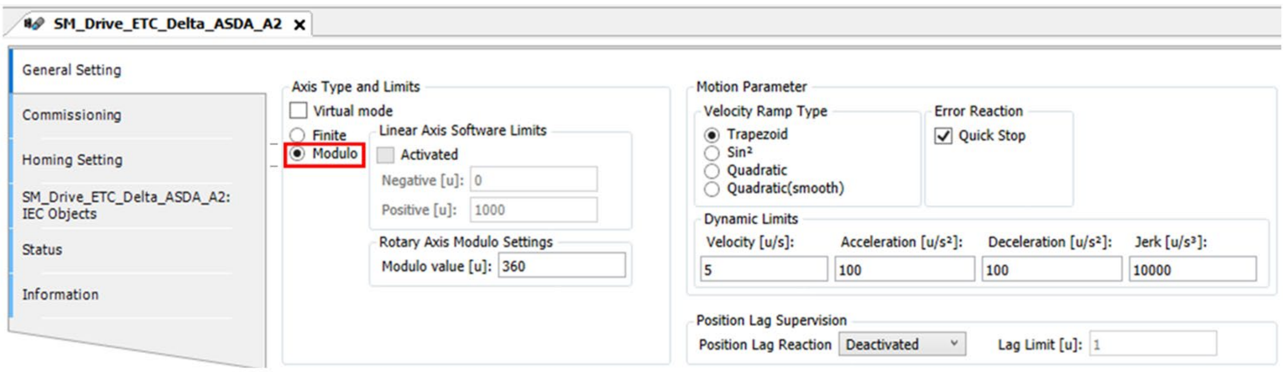
Axis types: **Modulo** stands for rotary axis; **Finite** stands for linear axis.



8. Set the **Linear Axis Software Limits**. When **Activated** is checked, the software limits will be activated. **Negative** refers to negative limit whereas **Positive** refers to positive limit.

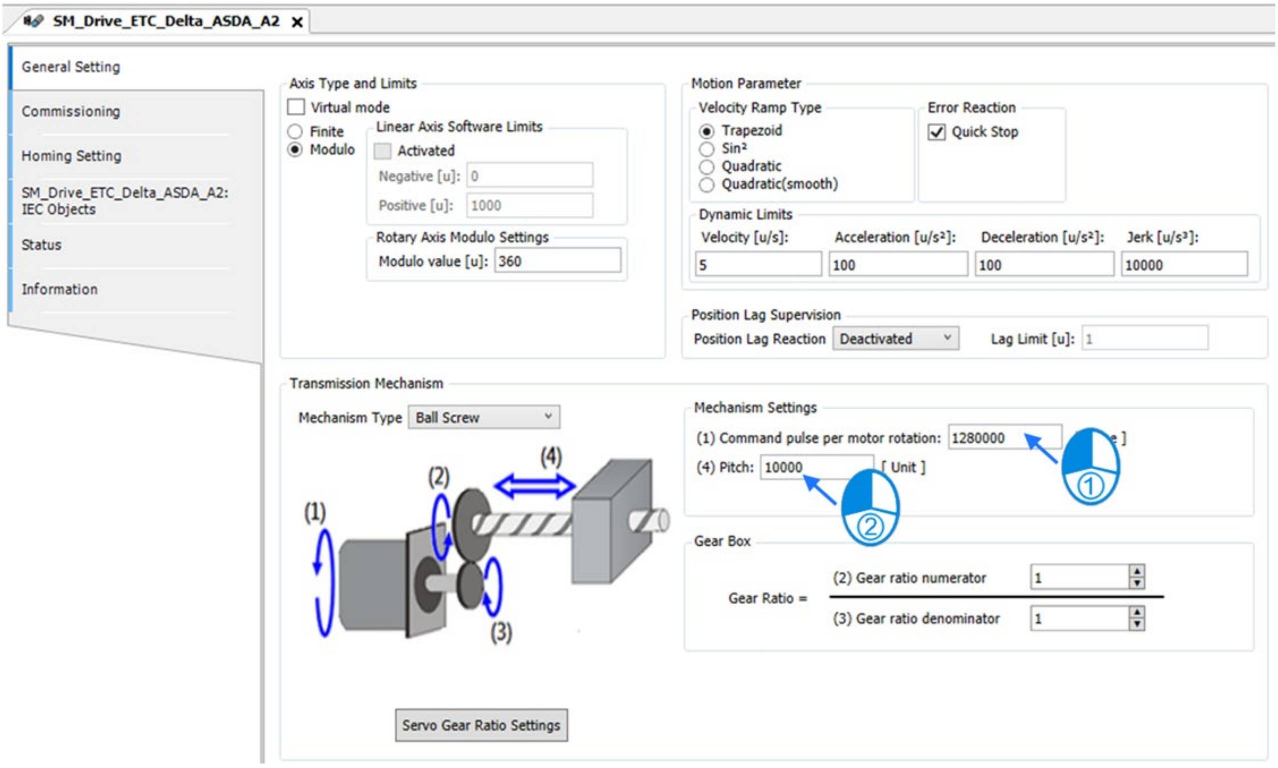


9. After the rotary axis settings are done, the rotation range must be defined. You should set the **Modulo value** in **Rotary Axis Modulo Settings**.

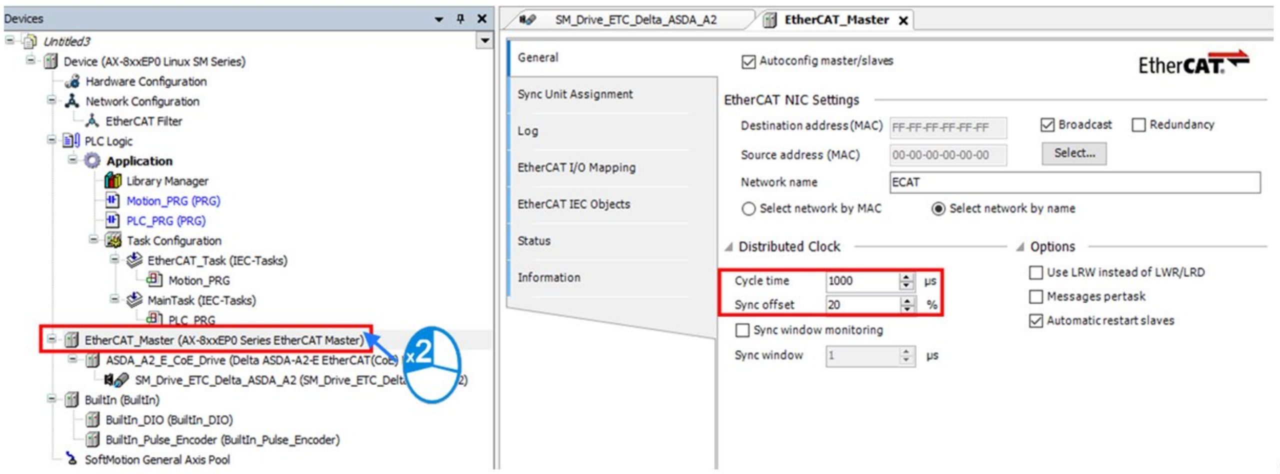


10. Scaling / Mapping page

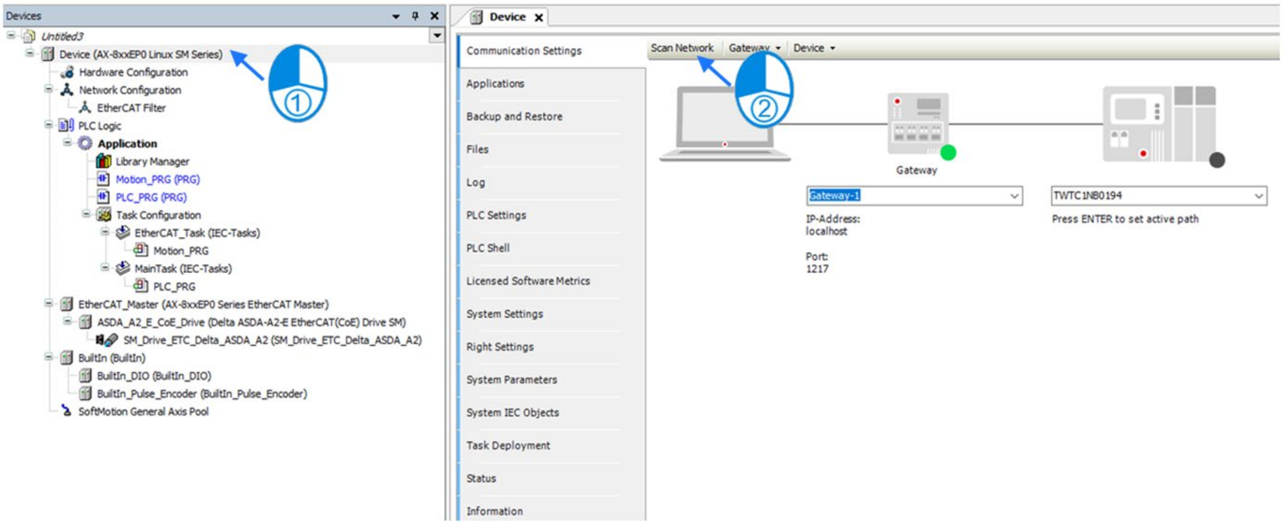
    Set the pulse value for **Command pulse per motor rotation**.
    Set the moving distance per motor revolution, **Pitch**.



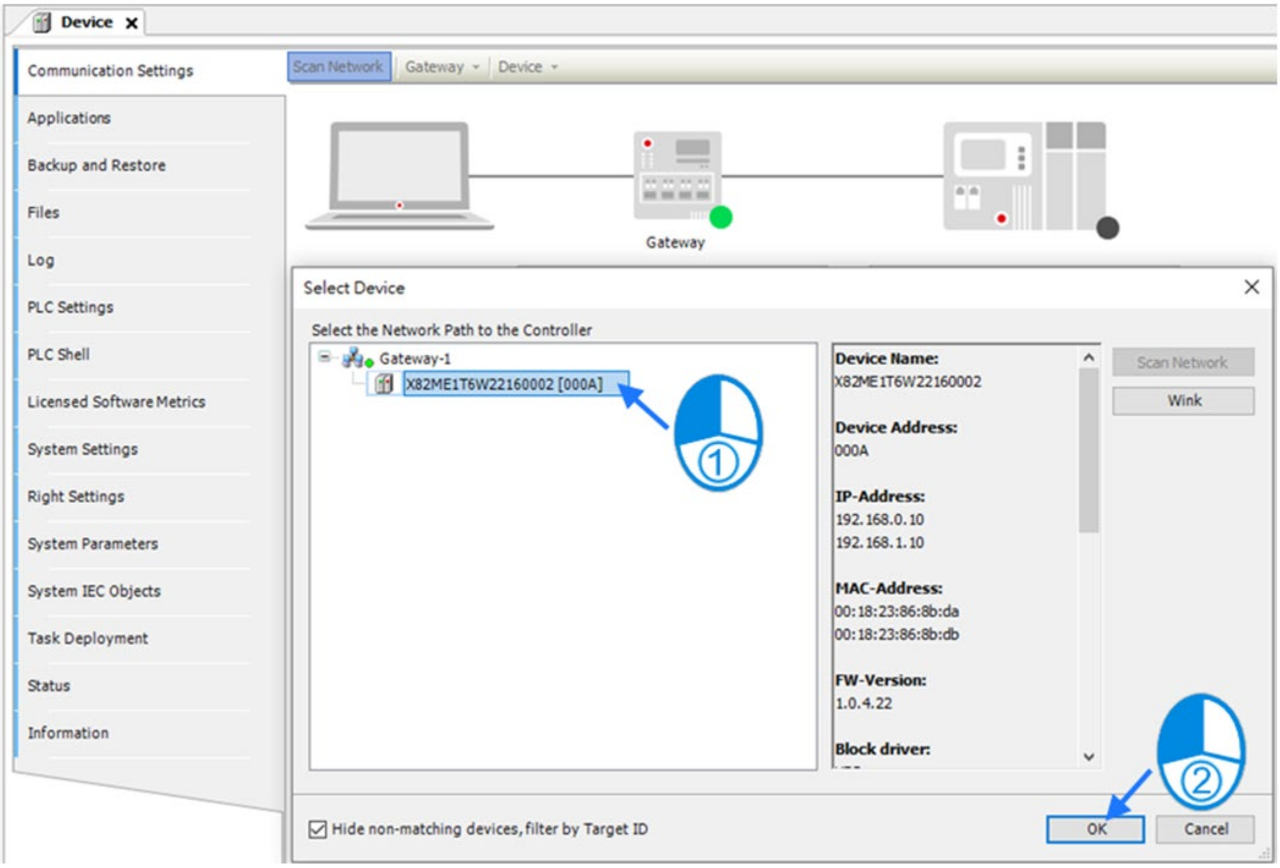
11. To set the EtherCAT communication cycle, click **EtherCAT_Master_SoftMotion** and set the **Cycle time** to **4000**, **Sync offset** to **20**.



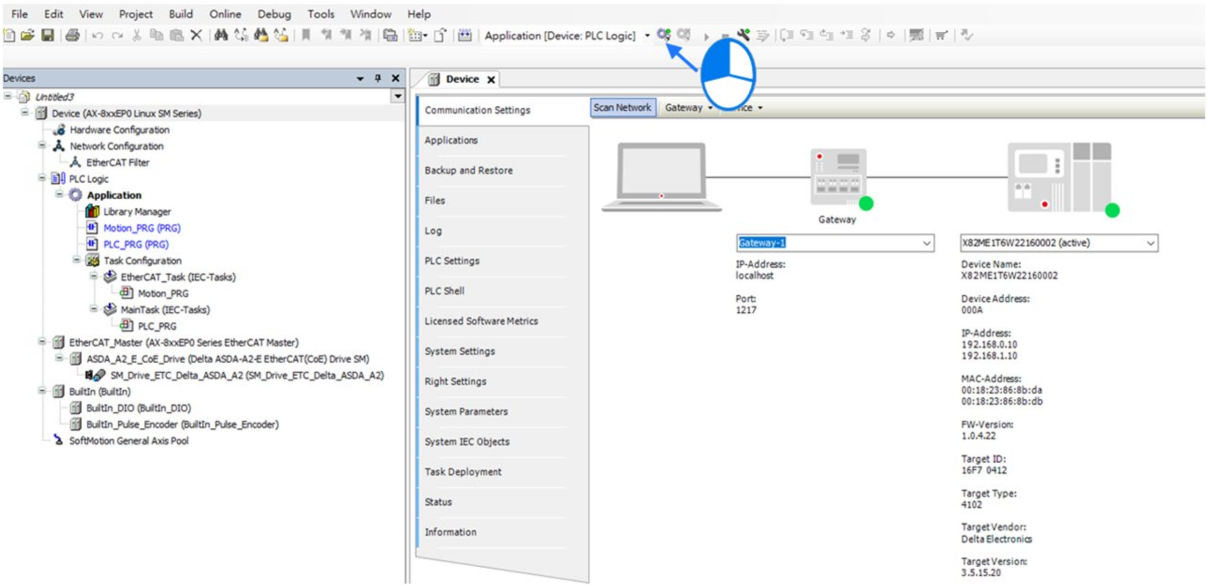
12. Scan the network for PLC.



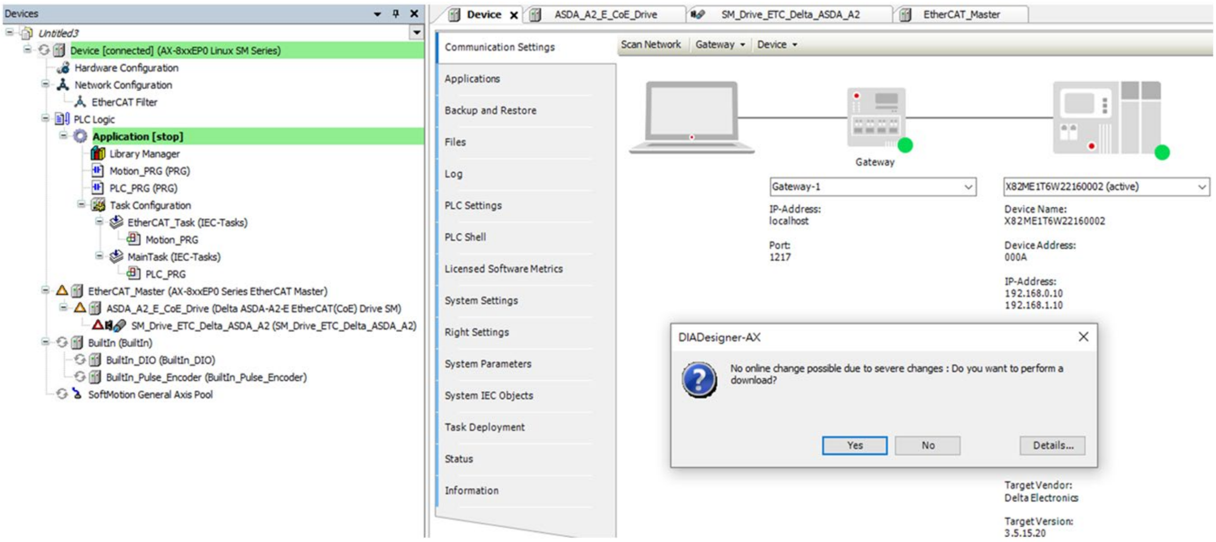
13. After adding the scanned PLC, click **OK**.

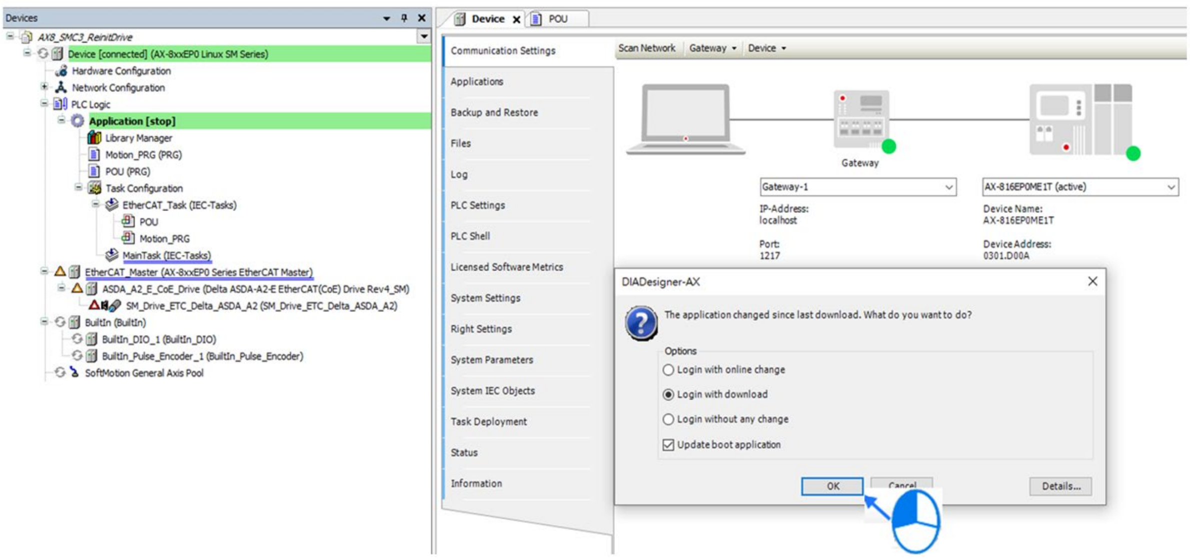


14. A green light appears after the controller is successfully connected. Click **Login**.



15. Here, a window showing whether to download the program will pop up. Click **YES** to continue.





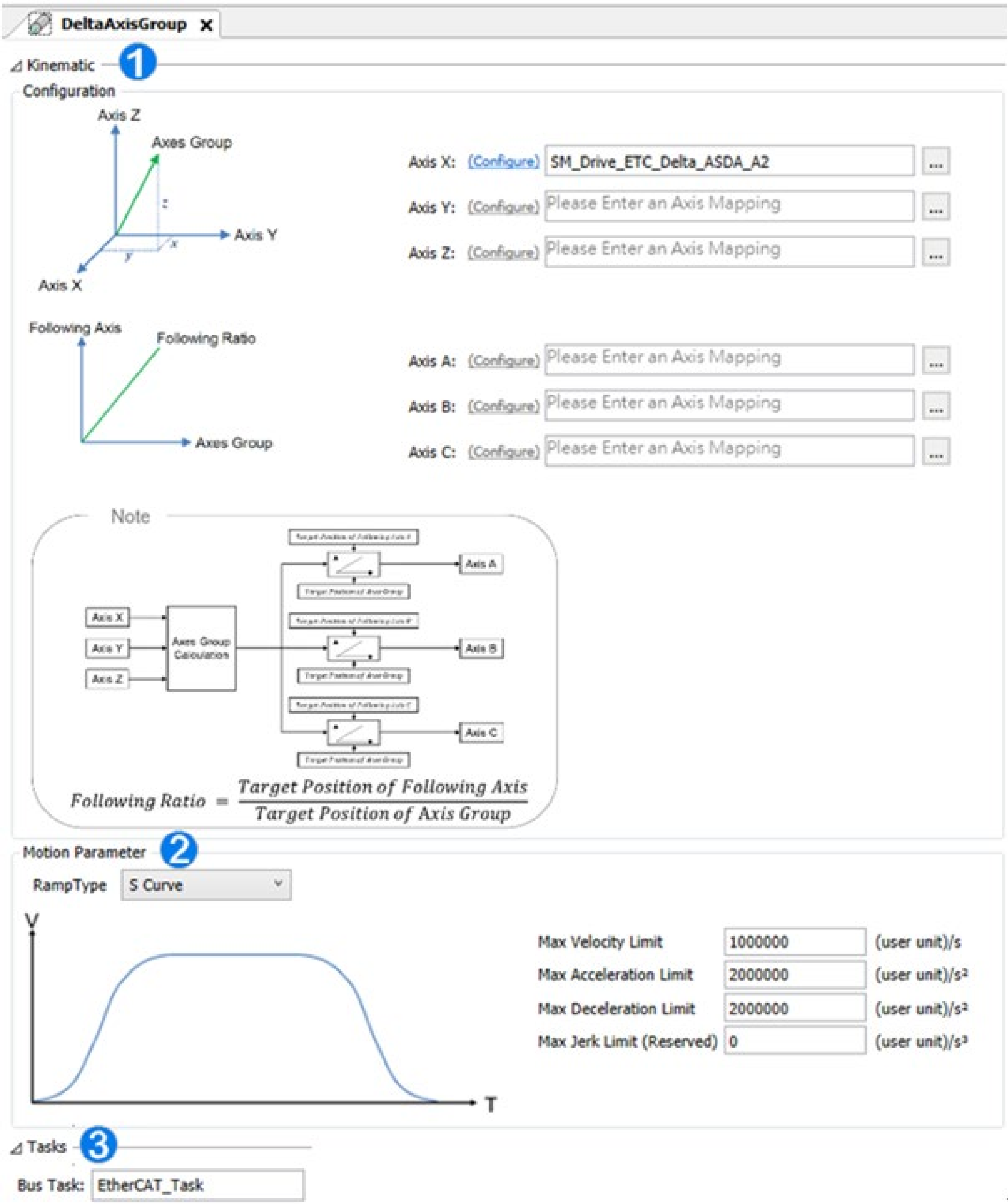
7.4.4 Axis Group Settings

Axis group movement should be applied via DIADesigner-AX software when multiple axes are executed simultaneously with linear and helical interpolation functions required.

Number of control axes	**Maximum number of axes controlled by linear interpolation**	6 axes
	Number of control axes controlled for helical interpolation	6 axes (3 slave axes)

7.4.4.1 Introduction to Axis Group Parameters

The parameters used for axis group motion are as follows:



- ① **Kinematic**

Name	Function
Axis X*1	X axis in axis group
Axis Y*1	Y axis in axis group
Axis Z*1	Z axis in axis group
Axis A*1	A axis in axis group
Axis B*1	B axis in axis group
Axis C*1	C axis in axis group

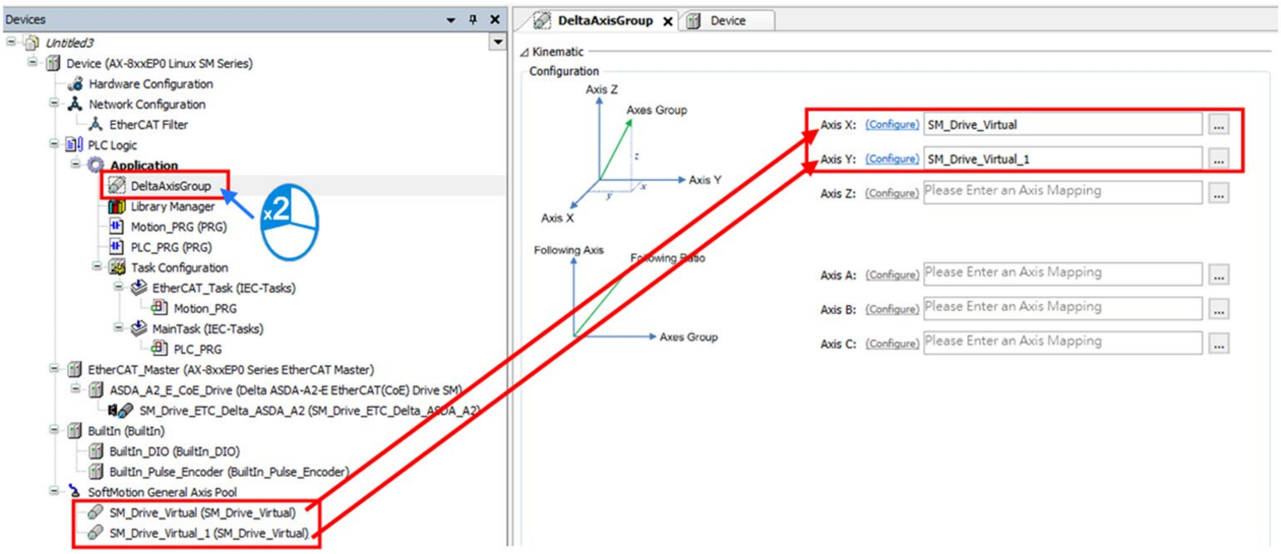
- ② **Motion Parameters**

Name	Function
Ramp Type*2	Velocity ramp type
Max Velocity Limit*3	Maximum velocity of axis group
Max Acceleration Limit*3	Maximum acceleration of axis group
Max Deceleration Limit*3	Maximum deceleration of axis group
Max Jerk Limit (Reserved)*3	Maximum jerk rate of axis group (this function is reserved)

- ③ **Tasks**

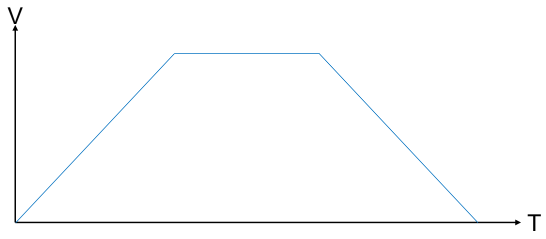
Name	Function
Bus Task	Set update task for axis group

*1: Axis X ~ Axis C: Input the individual axis name.

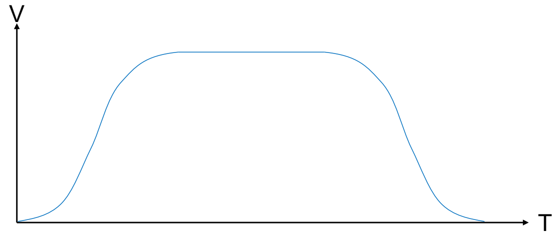


*2: There are two ramp types: Trapezoid and S Curve, which are displayed as follows.

- Trapezoid



- S Curve

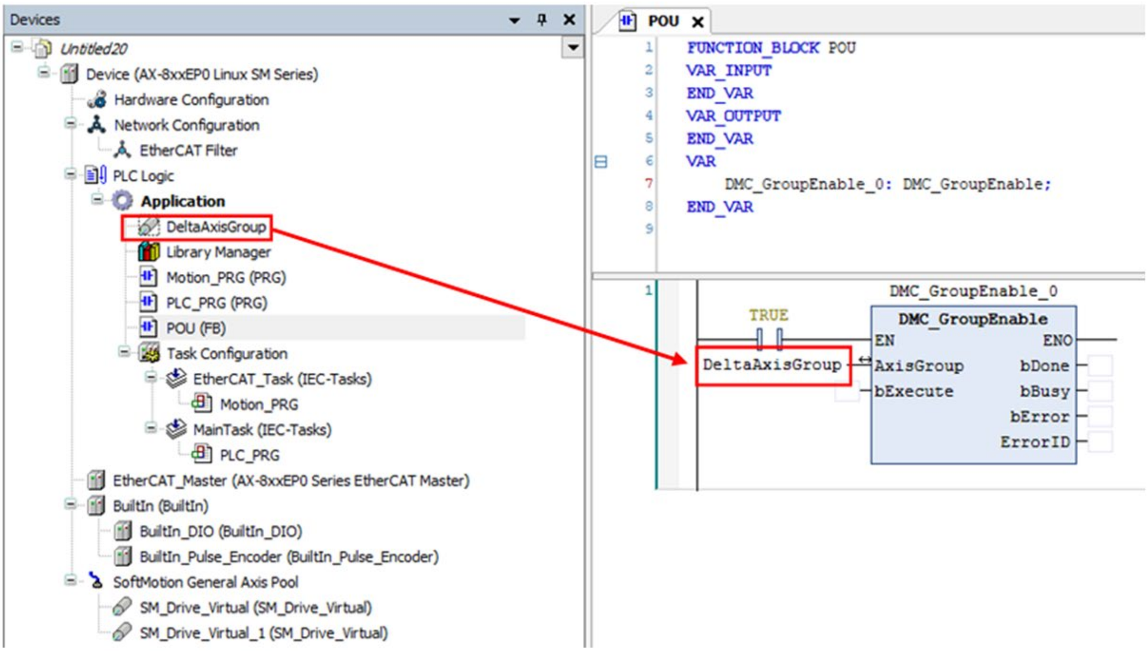


*3:

- Max Velocity Limit: When the movement speed of the axis group exceeds the set parameters, an error will be reported.
- Max Acceleration Limit: When the axis group movement acceleration exceeds the set parameter, an error will be reported.
- Max Deceleration Limit: When the axis group movement deceleration exceeds the set parameter, an error will be reported.

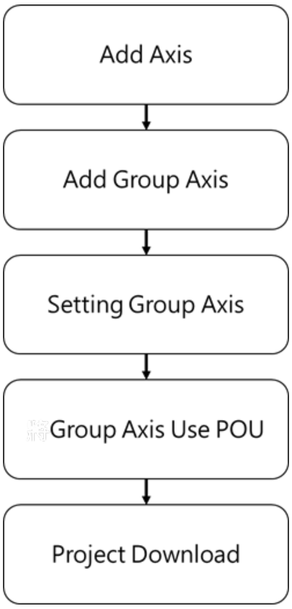
7.4.4.2 Using Axis Groups in the Program

Before using the axis group function block in the program, you must add an axis group node in the project tree and set the name of the single axis to be used by the axis group. After the setting is complete, please point the node name of the axis group to the AxisGroup pins of each functional block.



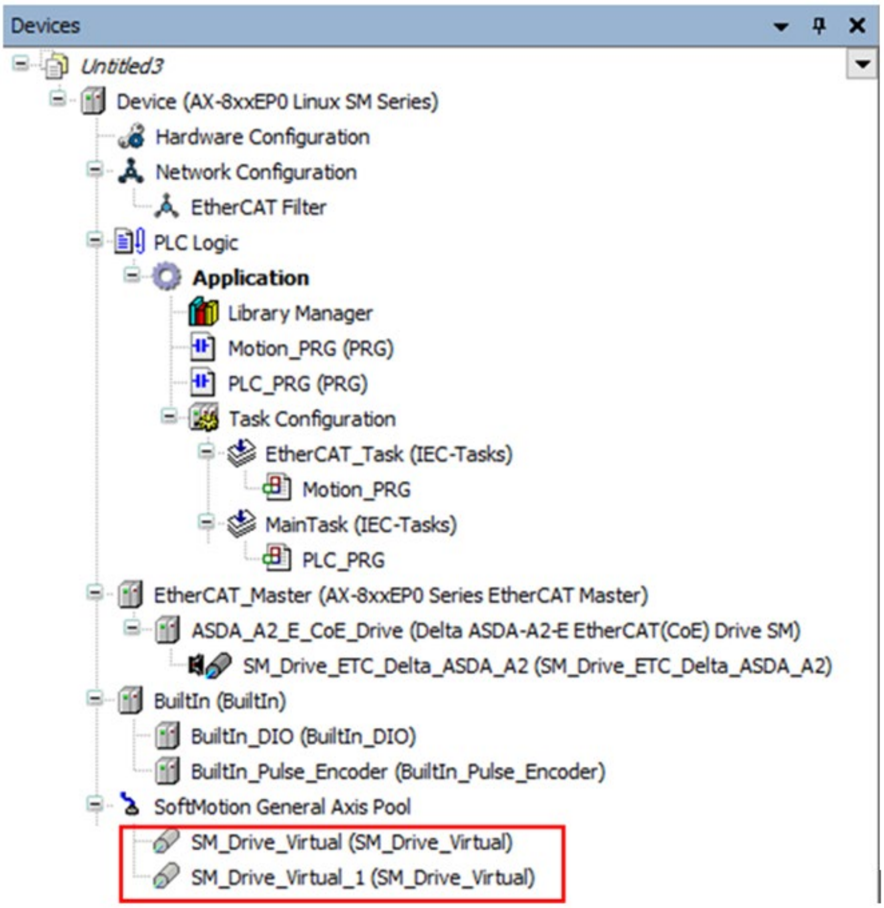
7.4.5 Process for Axis Group Configuration

To use the axis group movement, you must set the axis group name and the corresponding individual axis through DIADesigner-AX. The flowchart of axis group creation is as follows.

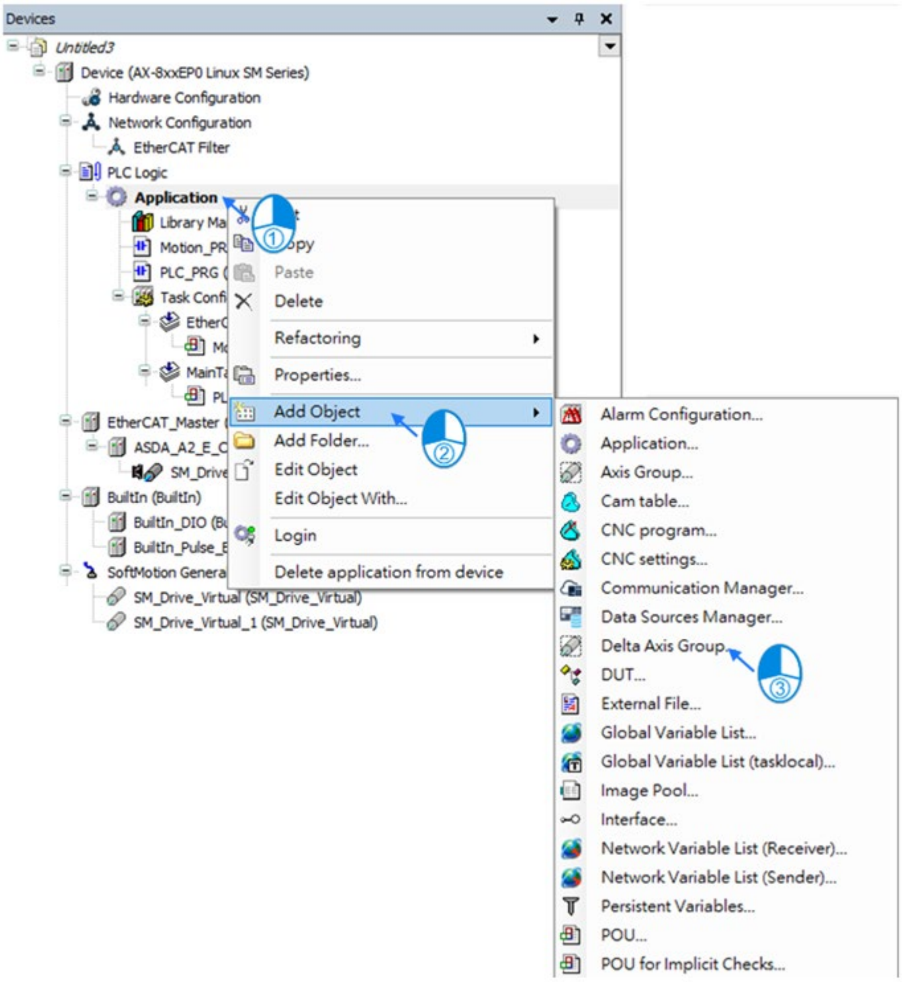


- **Steps to create an axis group**

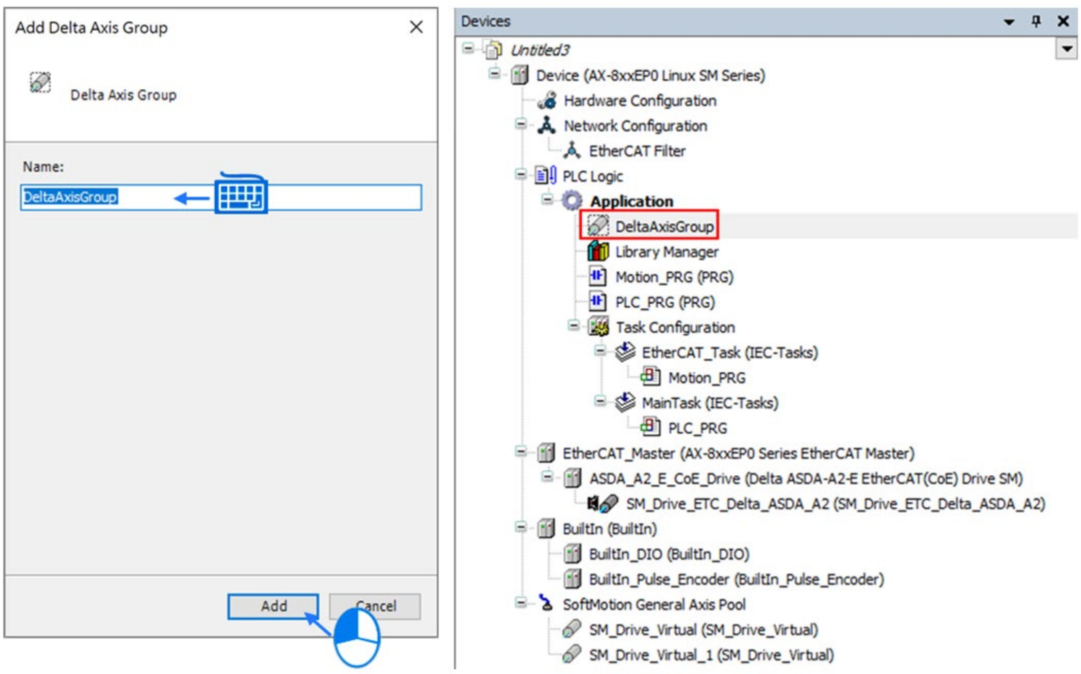
1. Add a single axis. The following process takes two virtual axes creation as an example.



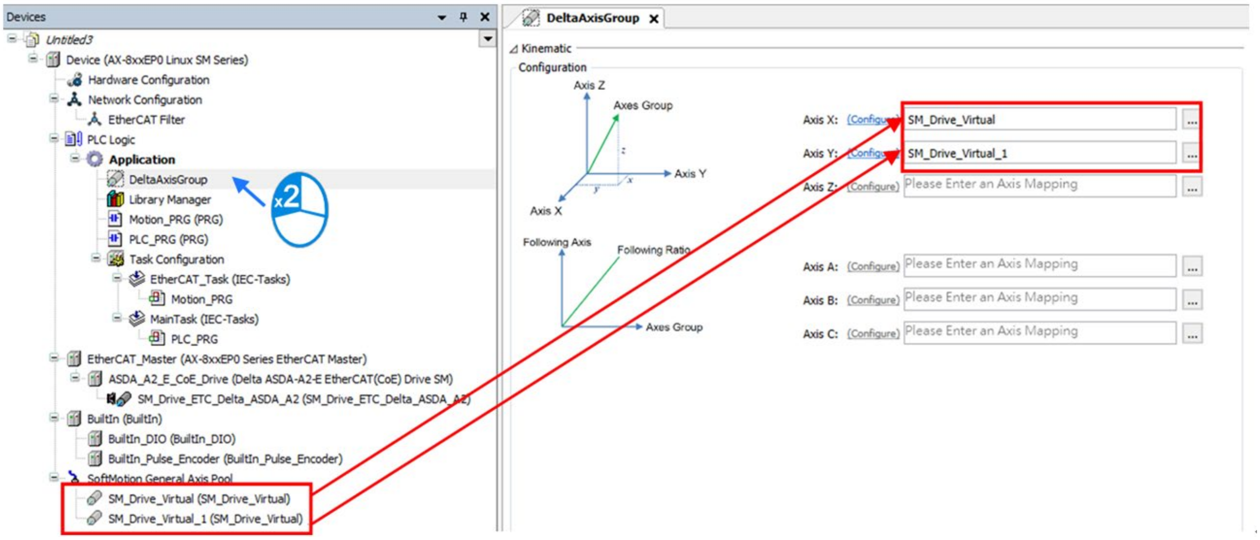
2. After the creation is complete, right-click **Application** and select **Add Object** > **Delta Axis Group**.



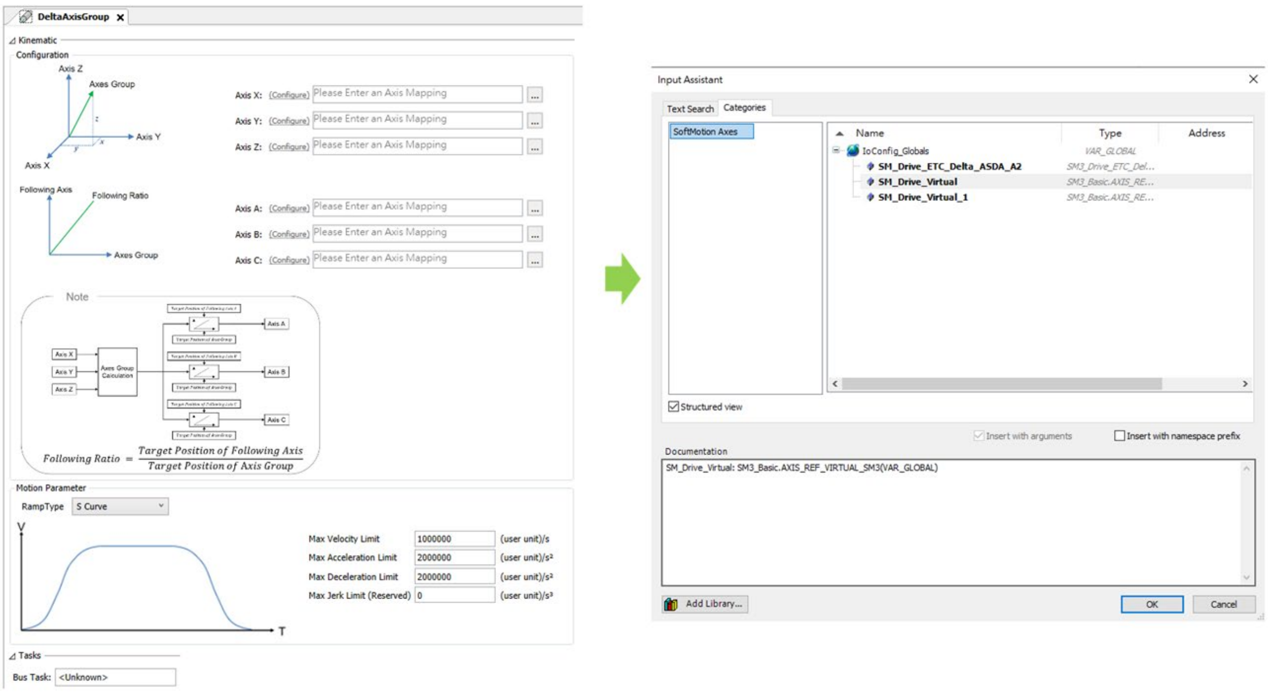
3. Set the name for axis group in the **Add Delta Axis Group** window and click **Add**. **DeltaAxisGroup** will then appear in the project tree on the left.



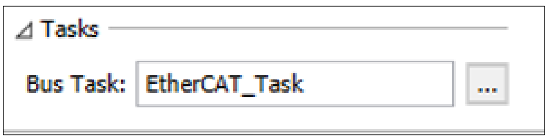
4. Click on **DeltaAxisGroup**, input the names of the two virtual axes into **Axis X** and **Axis Y**.



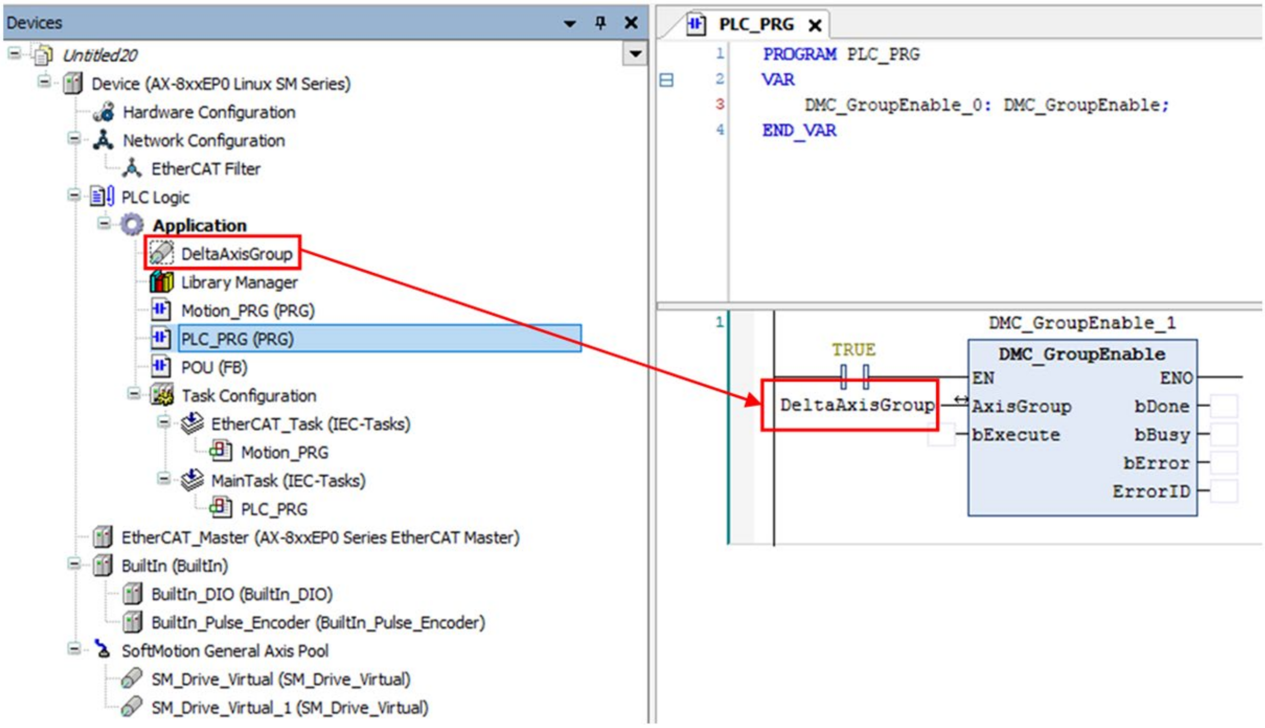
5. Then click **Bus Task** to enter the **Input Assistant** page. Select **EtherCAT_Task** and click **OK**.



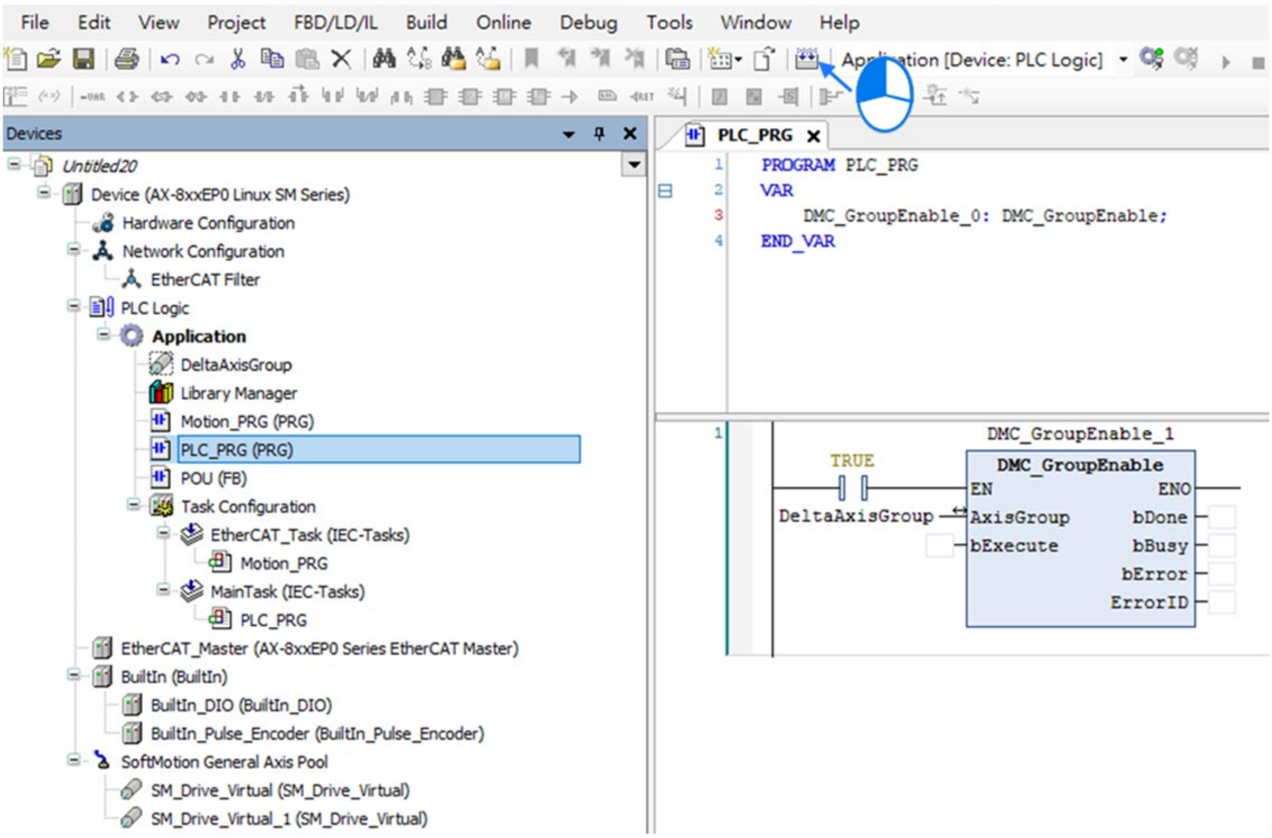
6. After setting the task, **EtherCAT_Task** appears in the field of **Tasks**.



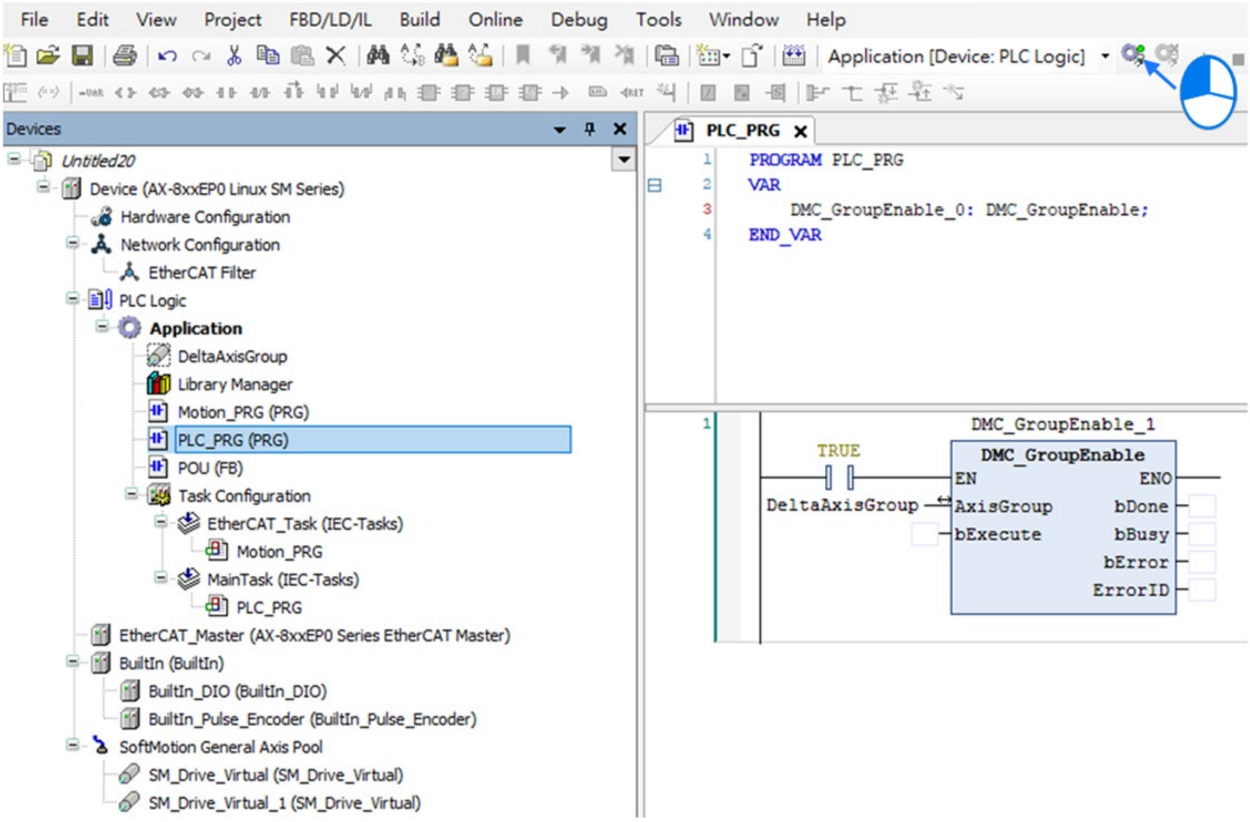
7. Add **DMC_GroupEnable** function block under **PLC_PRG**, and fill in the axis group name in the **AxisGroup** pin.



8. After the program is written, click **Compile** icon to check the validity.



9. After compilation, click on **Online Monitoring** button to download the program.



7.5 Motion Axis Variables

7.5.1 Variables for Single Axis

After axes are created in the project tree of the software DIADesigner-AX, the corresponding axis parameters, which are only for reading, will be generated automatically. Axes are divided into two types: synchronous axis (Axis_REF_SM3) and positioning axis (Axis_REF_DML), which are set out in the following tables:

- **Synchronous axis (Axis_REF_SM3)**

No.	Name	Data Type	Default Value	Description
1000	nAxisState	SMC_AXIS_STATE(INT)	Standstill (3)	Show current axis state according to MC_ReadStatus
1012	bCommunication	BOOL	FALSE	TRUE when EtherCAT axis communication is normal; FALSE if disconnected
1014	uiDriveInterfaceError	UINT	0	If the Driver Interface detects an error, it will display the corresponding error code
1021	wDriveId	WORD	Driver	The number in driver nodes on the Fieldbus
1025	fTaskCycle	LREAL	Driver	EtherCAT task cycle time
1035	fbeFBError	ARRAY [0..g_SMC_NUMBER_FB_ERRORS] OF SMC_FBERROR	0	Axis-related error table
1040	bVirtual	BOOL	FALSE	TRUE if it is a virtual axis; FALSE if it is a real axis.
1051	iRatioTechUnitsNum	DINT	1	Change gear ratio in axis setting (denominator)
1052	dwRatioTechUnitsDenom	DWORD	1	Change gear ratio in axis setting (numerator)
1060	iMovementType	INT	1	0 = Modulo 1 = Finite
1061	fPositionPeriod	LREAL	1000	Maximum moving distance of rotary axis
1062	eRampType	SMC_RAMPTYPE	Trapez	Velocity ramp type: ■ Trapezoid ■ sin^2 ■ Quadtatic ■ Quadtatic (smooth)
1100/1	fSetPosition	LREAL	0	Command position (user-defined unit)
1101	fActPosition	LREAL	0	Feedback position (user-defined unit)

7

No.	Name	Data Type	Default Value	Description
1110,11	fSetVelocity	LREAL	0	Command velocity (user-defined unit/s)
1111,10	fActVelocity	LREAL	0	Feedback velocity (user-defined unit/s)
1115	bConstantVelocity	BOOL	FALSE	TRUE when the velocity is constant
1120	fSetAcceleration	LREAL	0	Acceleration command (user-defined unit/s^2)
1125	bAccelerating	BOOL	FALSE	TRUE when axis is accelerating
1135	bDecelerating	BOOL	FALSE	TRUE when axis is decelerating
1140	fSetJerk	LREAL	0	Commanded jerk value
1160	fSetTorque	LREAL	0	Commanded torque (Nm)
1161	fActTorque	LREAL	0	Actual torque (Nm)
1200,2	fSWLimitPositive	LREAL	0	Set the range of software positive limit
1201,3	fSWLimitNegative	LREAL	0	Set the range of software negative limit
1204	bSWEndSwitchActive	BOOL	FALSE	TRUE when software limit switch activated; the state machine changes to Errorstop
1205	bSWLimitEnable	BOOL	FALSE	Software limit switch: TRUE (Enable) / FALSE (Disable)
-	strDriveInterfaceError	STRING	‘’	Axis error

- **Positioning axis (Axis_REF_DML)**

No.	Name	Data type	Default value	Description
1000	nAxisState	SML_AXIS_STATE	SML_AS_PowerOff (0)	Show current axis state according to MC_ReadStatus
1012	bCommunication	BOOL	FALSE	TRUE when EtherCAT axis communication is normal; FALSE if disconnected
1014	uiDriveInterfaceError	UINT	0	If the Driver Interface detects an error, it will display the corresponding error code
1051	iRatioTechUnitsNum	DINT	1	Change gear ratio in axis setting (denominator)
1052	dwRatioTechUnitsDenom	DWORD	1	Change gear ratio in axis setting (numerator)
1060	iMovementType	SML_Movemen	SML_MT_MO	Axis types:

No.	Name	Data type	Default value	Description
		tType	DULO	SML_MT_MODULO = rotary axis SML_MT_FINITE = linear axis
1062	eRampType*1	SMC_RAMP TYPE	Trapez	Velocity ramp type: ■ Trapezoid ■ sin^2
1101	fActPosition	LREAL	0	Feedback position (user-defined unit)
-	strDriveInterfac e Error	STRING	‘’	Axis error

*1: Only Trapezoid and sin^2 are supported.

7

7.5.2 Variables for Axis Group

After axis groups are created in the project tree of the software DIADesigner-AX, the system will automatically create the corresponding axis variables, which are set out in the following table.

Name	Data type	Set value (default)	Function
GroupState	DMC_ GROUP_ STATE	GroupDisabled / GroupStandby / GroupMoving / GroupHoming / GroupStopping / GroupErrorstop (GroupDisabled)	Commands for axis groups status
bError	BOOL	TRUE / FALSE (FALSE)	TRUE when an error occurs in the axis group
dwErrorId	DMC_ ERROR	DMC_ERROR (DMC_GM_NO_ ERROR)	Detailed error description
lrVelocity	LREAL	0 ~ 1.798E+308 (0)	Current velocity of axis group
lrAcceleration	LREAL	Positive, negative, or zero (0)	Current acceleration of axis group
lrJerk	LREAL	Positive, negative, or zero (0)	Current jerk of axis group
bAccelerating	BOOL	TRUE / FALSE (FALSE)	TRUE when acceleration
bDecelerating	BOOL	TRUE / FALSE (FALSE)	TRUE when deceleration
bConstantVelocity	BOOL	TRUE / FALSE (FALSE)	TRUE when constant velocity (includes velocity = 0)
bInPosition	BOOL	TRUE / FALSE (FALSE)	TRUE when positioning is completed
bContinueDataWrit en	BOOL	TRUE / FALSE (FALSE)	TRUE if relevant data is available for DMC_GroupContinue after axis group motion is suspended
ContinuePos	ARRAY [0..5] OF LREAL	[0,0,0,0,0,0]	Record the current position of the axis group when the execution of DMC_GroupInterrupt is done
AxisX_Name*1	String		Display the axis name used by the current axis group in Axis_X
AxisY_Name*1	String		Display the axis name used by the current axis

Name	Data type	Set value (default)	Function
			group in Axis_Y
AxisZ_Name[*1]	String		Display the axis name used by the current axis group in Axis_Z
AxisA_Name[*1]	String		Display the axis name used by the current axis group in Axis_A
AxisB_Name[*1]	String		Display the axis name used by the current axis group in Axis_B
AxisC_Name[*1]	String		Display the axis name used by the current axis group in Axis_C
RampType	DMC_GROUP_RAMP_TYPE	Trapezoid / S Curve (S Curve)	Ramp type of the current axis group
lrMaxVelocityLimit	LREAL	positive number or zero (1000000)	Maximum velocity of axis group
lrMaxAccelerationLimit	LREAL	positive number or zero (2000000)	Maximum acceleration of axis group
lrMaxDecelerationLimit	LREAL	positive number or zero (2000000)	Maximum deceleration of axis group
lrMaxJerkLimit(Reserved)	LREAL	positive number or zero (0)	Maximum jerk of axis group (reserved)
bVelocityWarning	BOOL	TRUE / FALSE (FALSE)	TRUE when the axis group velocity exceeds the value set in lrVelocityWarningPercentage
bAccelerationWarning	BOOL	TRUE / FALSE (FALSE)	TRUE when the axis group acceleration exceeds the value set in lrAccelerationWarningPercentage
bDecelerationWarning	BOOL	TRUE / FALSE (FALSE)	TRUE when the axis group deceleration exceeds the value set in lrDecelerationWarningPercentage
bJerkWarning (Reserved)	BOOL	TRUE / FALSE (FALSE)	TRUE when the axis group jerk exceeds the value set in lrJerkWarningPercentage
StopMethod	Enum of BYTE	Immediate Stop / MaxGroupDecStop / MaxAxisDecStop (Immediate Stop)	Set the stopping method for the axis group when errors occur or when the movement is meant to stop
lrVelocityWarningPercentage	LREAL	0 ~ 1 (0)	When setting the maximum ratio of the axis group speed limit, the speed warning is displayed, and setting 0 disables the speed warning.
lrAccelerationWarningPercentage	LREAL	0 ~ 1 (0)	When setting the maximum ratio of the axis group acceleration limit, the acceleration warning is displayed, and setting 0 is to disable the acceleration warning.
lrDecelerationWarni	LREAL	0 ~ 1	When setting the maximum proportion of the axis

7

Name	Data type	Set value (default)	Function
ngPercentage		(0)	group deceleration limit, a deceleration warning is displayed. Set 0 to disable the deceleration warning.
lrJerkWarning Percentage (Reserved)	LREAL	0 ~ 1 (0)	When setting the maximum ratio of the axis group jerk limit, the jerk warning is displayed, and setting 0 disables the jerk warning.
Radius Correction	LREAL	0 ~ 100 (0,1)	The is to set the permissible error of the radius when helical interpolation is selected. Error % = the distance between [the center point] and [the vertical bisector of the starting and ending positions]/radius.

*1: When the axis type is a rotation axis, the set movement distance cannot exceed the modulo setting value. If it exceeds the limit, an **Axis limit violated** error will be reported.

7.6 Motion Control Programming

7.6.1 Motion Control Program

Please refer to the following descriptions before starting programming in DIADesigner-AX.

7.6.1.1 DIADesigner-AX Programming Structure and Types

In the standard structure, all programs, including subroutines, are written in the same source code for a PLC. With the size of program getting larger, maintenance and debugging also becomes a considerable burden. Under the framework of IEC 61131-3, the program is divided into several units according to the functions or characteristics, which makes the development and maintenance of the program easier. Because the Program Organization Unit (POU) is modular, different POUs can be developed by different designers, which can bring significant benefits to the allocation and execution of project manpower. In DIADesigner-AX, POUs are of three types: program (PROG), function block (FB) and function (FC).

1. Program (Program, PROG)

   A program POU can determine its role based on its designated task (TASK) type. If it is assigned to periodic work, the program POU will play the role of the main program; if it is assigned to interrupt type work, the program POU will play the role of the interrupt subroutine. In addition, the program POU can also call function blocks (FB).

2. Function Block (Function Block, FB)

   Static (memory) variable symbols can be declared inside the function block POU. Therefore, the value of the static variable symbol can be maintained after the operation. Since the value stored in the function block is calculated together with the input value during execution, even if the input value is the same, the resulting output may be different. In addition, the function block can call other function blocks internally.

3. Function (Function, FC)

   Function is used to pass back the operation results. In POU, only minimal permissions are provided for Function. Compared with FB, it cannot access memory space and can only return one single value. Function is used for calculation and then returns the results. In use, Function can only call Function but not FB because Function does not have memory space.

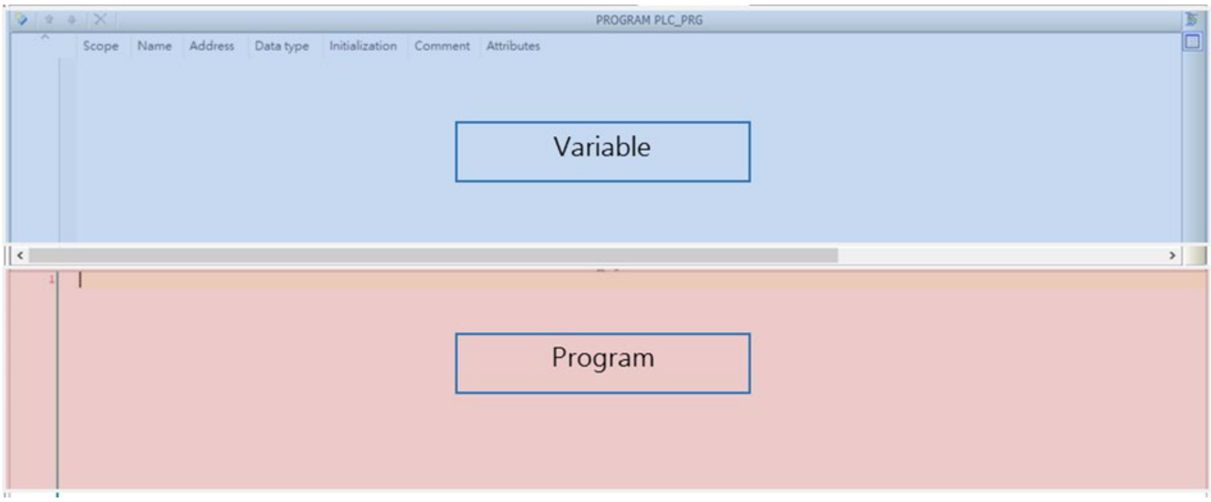
After creating a POU, each program POU must specify a task (TASK). Work (TASK) is used to determine the sequence of program POU execution or startup. The characteristic of the IEC61131-3 program architecture is that the program is divided into several independently developed POUs. When POUs are compiled, all POUs are rearranged and combined into an executable code that can be scanned step by step. The rearrangement order of POUs is based on the work (TASK) to which POUs are assigned. Work (TASK) is of the following types:

- Cyclic
- Event
- External
- Freewheeling
- Status

Refer to **Section 4.4.1** of this manual for details.

7.6.1.2 POU in DIADesigner-AX

All POUs created by users will be listed in the project management area. Program POU and function block POU will be managed separately. In addition, depending on the language in which the program POU and function block POU are compiled, the POU icon displayed will be different, and POU-related information will also be displayed behind each POU name. Double-click POU in the project management area to open the editing window of the POU. The POU editing window consists of two parts: the upper part of the window is the area symbol table of the POU, and the lower part of the window is the main body of the program. In addition, depending on the programming language, the editing environment in the lower part of the window will also be different.

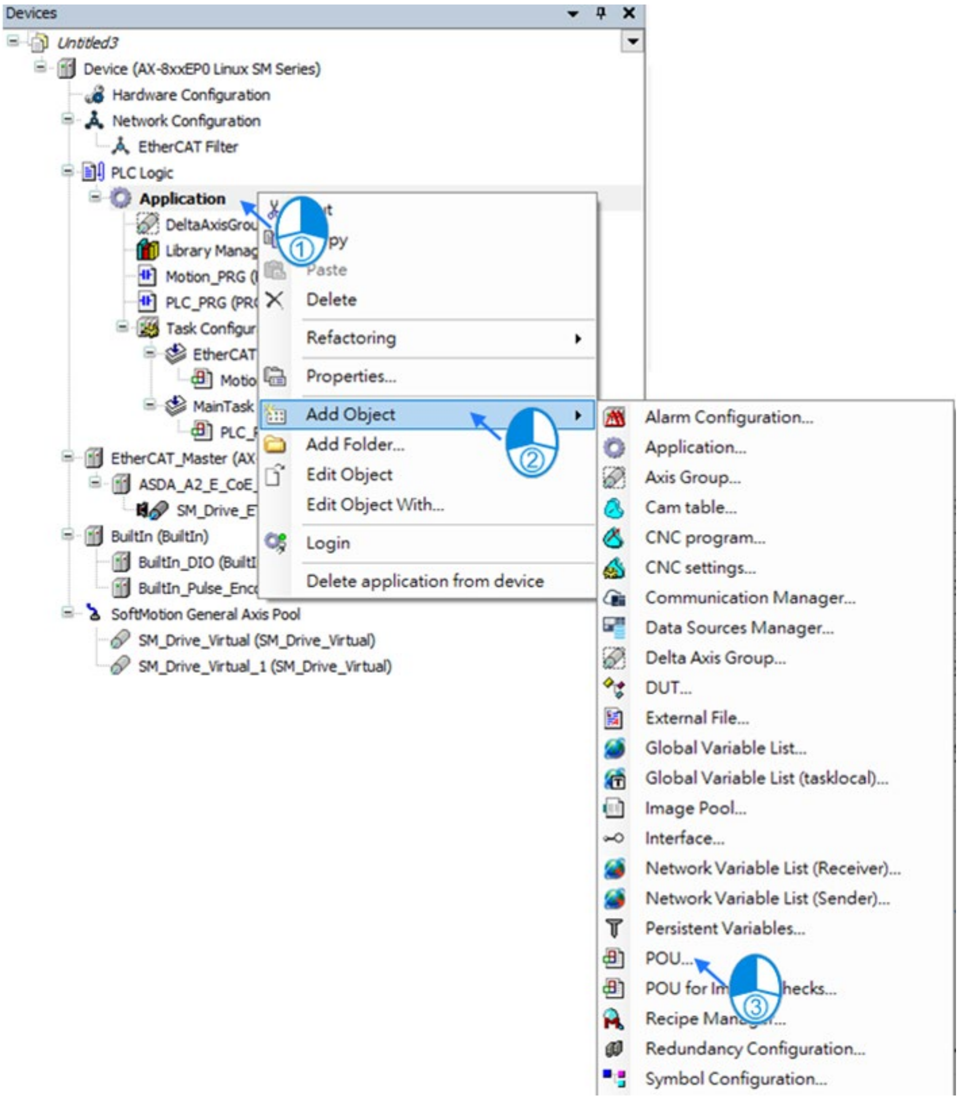


You can add the variables you need to use in the regional symbol table. In addition, in the system, when the axis / axis group is established, the corresponding variables will be automatically generated. The content of the variables can be referred to **Section 7.5**.

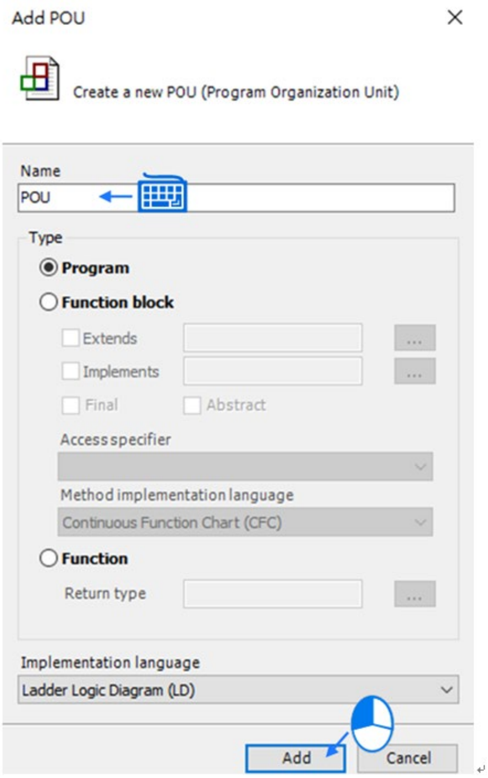
7

7.6.1.3 Adding POU in DIADesigner-AX

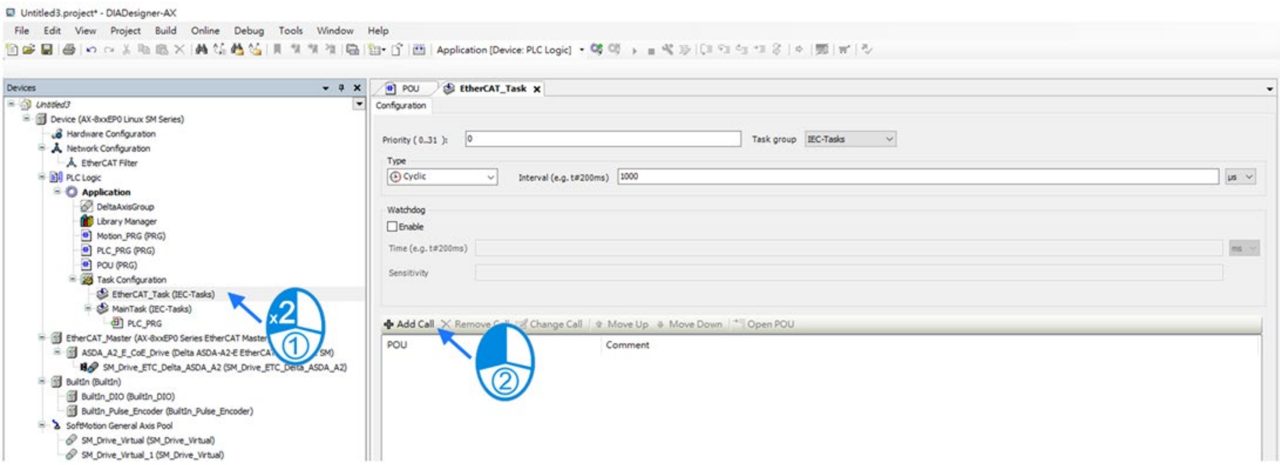
1. Open an existing project in DIADesigner-AX. Right-click **Application**, select **Add Object** and then **POU**.



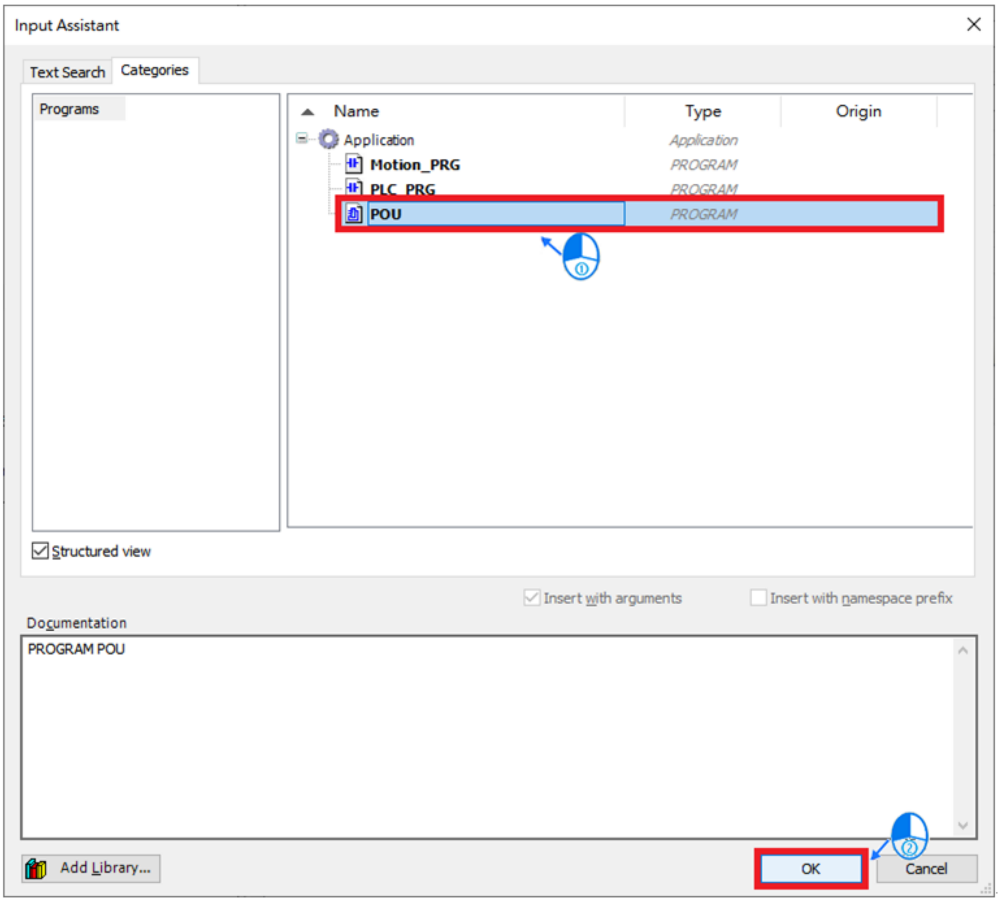
2. Enter the program name in the **Name** field and select the language in the **Implementation language** field. Then click **Add**.



3. At this point, the POU appears on the left. Double-click **EtherCAT_Task** and select **Add Call**.



4. Select the **POU** which has been created and click **OK**.



5. Select **POU** under **EtherCAT_Task** to start programming.

7.6.1.4 PDO Mapping

Set the communication of the PDO (Process Data Objects) mapping between the DIADesigner-AX software and the AX motion control CPU, before using motion control instructions.

PDO Mapping Settings:

RxPDO (1600 hex)	Control Word (6040 hex), TargetPosition (607A hex)
TxPDO (1A00 hex)	Status Word (6041 hex), ActualPosition (6064 hex)

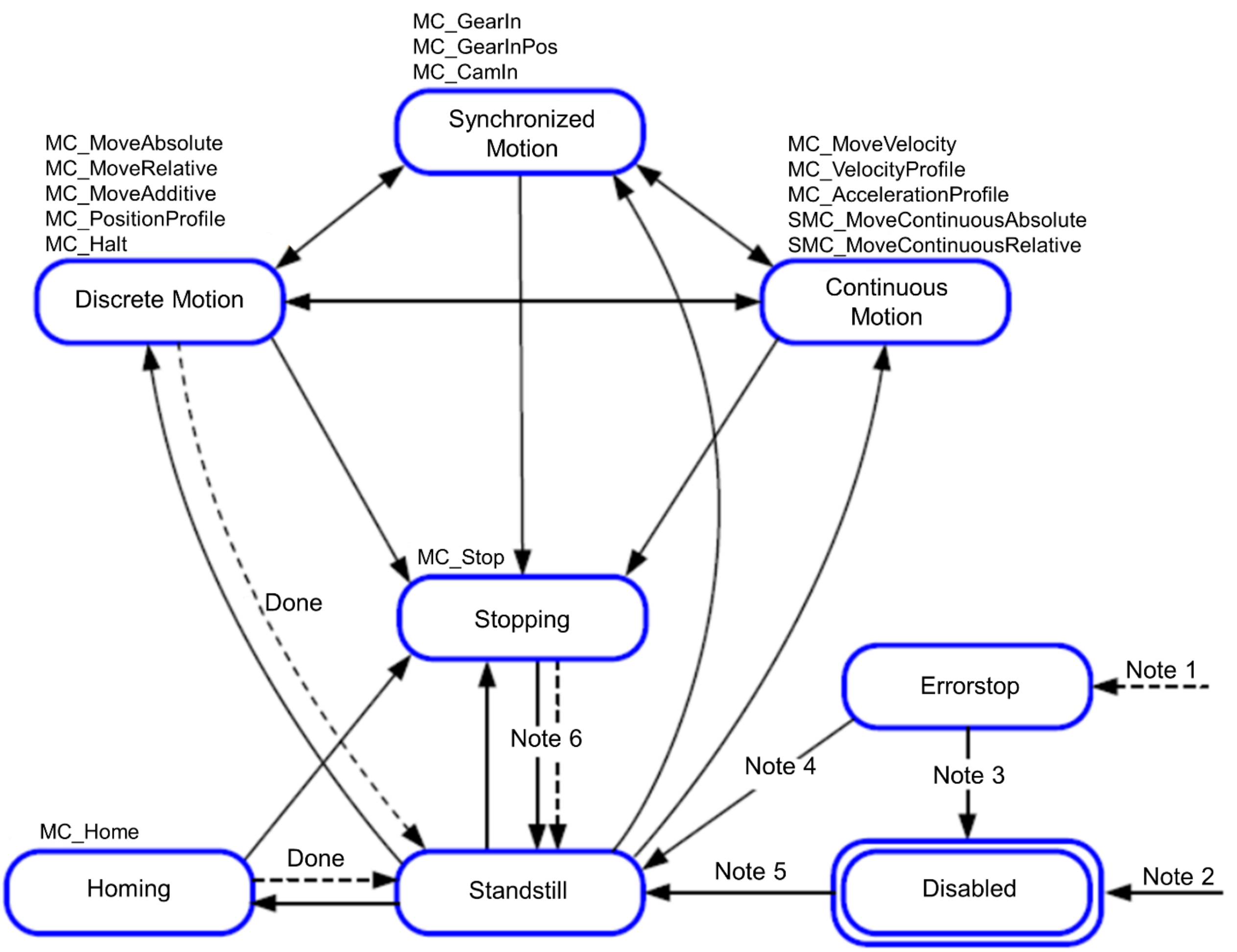
The table above pre-sets the PDO mapping parameters that will be used for ASDA-A2-E. For the PDO parameters required by the corresponding motion function blocks, refer to the **AX Series Motion Controller Manual.**

7.6.2 Axis State Transition

This section introduces transition of single axis state and multi-axis state in axis groups when different motion function blocks are used. The transition rules fulfill PLCopen motion control standard.

7.6.2.1 Axis State

- **Synchronous Axis**



Note 1: Whenever an error occurs, regardless of the state

Note 2: When MC_Power Enable = FALSE and no axis error occurs

Note 3: Execute MC_Reset and MC_Power = FALSE

Note 4: Execute MC_Reset, MC_Power = TRUE and MC_Power Enable = TRUE

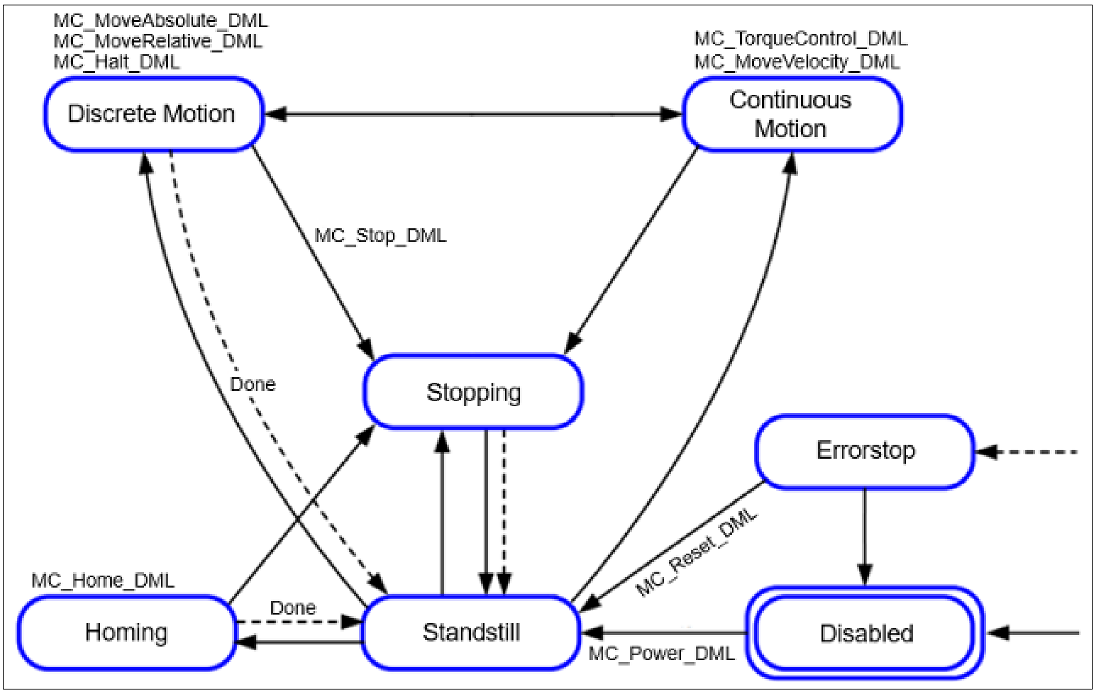
Note 5: MC_Power Enable = TRUE and MC_Power =TRUE

Note 6: When MC_Stop Done = TRUE and MC_Stop Execute = FALSE

State	Description
Disabled	The axis is in Servo Off state.
Standstill	The axis is in Servo On state.
Discrete Motion	The state is Discrete Motion while executing single-axis motion instructions.

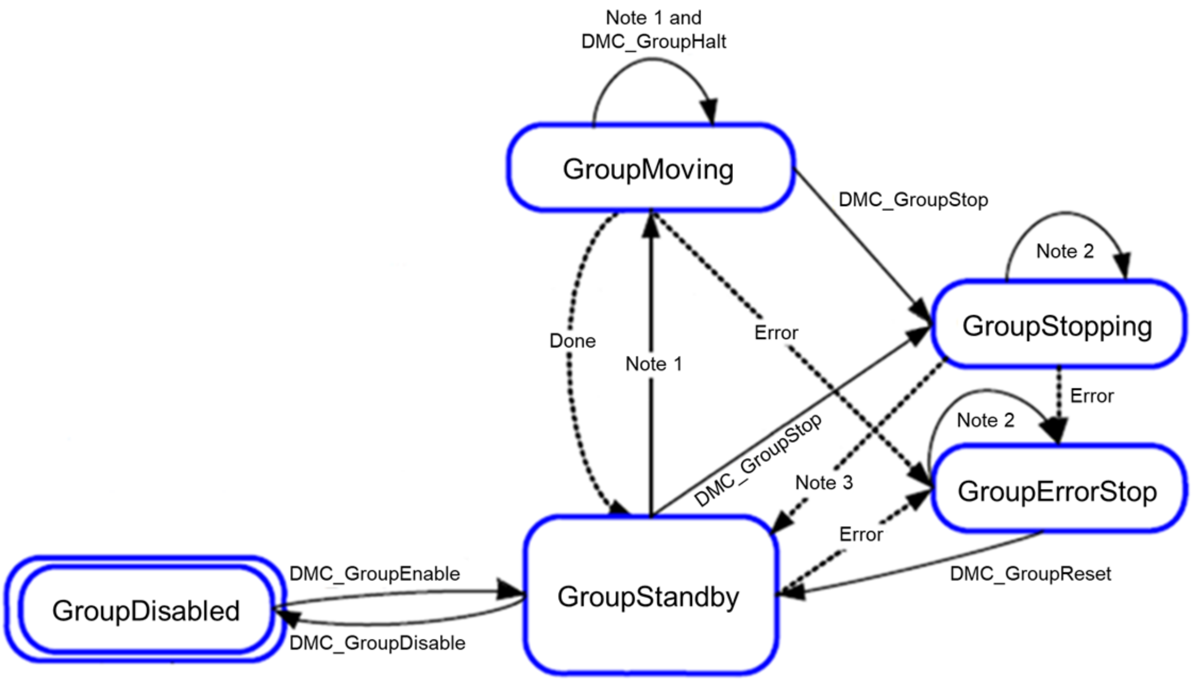
Continuous Motion	The state is Continuous Motion while executing single-axis continuous motion instructions.
Synchronized Motion	The state is Synchronized Motion while executing synchronized motion instructions, including synchronized waiting instruction.
Stopping	After the execution of MC_Stop instruction stops; Active is TRUE. In this state, no single-axis motion instructions can be executed.
Errorstop	Whenever an error occurs in single axis, it enters the state of Errorstop, in which no single-axis motion instructions can be executed.
Homing	The state is Homing while executing MC_Home or MC_HomeWithParameter instructions for single axis.

- **Positioning Axis**



State	Definition
Disabled	The axis is in Servo Off state
Standstill	The axis is in Servo On state
Discrete Motion	The state is Discrete Motion while executing single-axis motion instructions.
Continuous Motion	The state is Continuous Motion while executing single-axis continuous motion instructions.
Stopping	After the execution of MC_Stop_DML instruction stops, Active is TRUE. In this state, no single-axis motion instructions can be executed.
Errorstop	Whenever an error occurs in single axis, it enters the state of Errorstop, in which no single-axis motion instructions can be executed.
Homing	The state is Homing while executing MC_Home or MC_HomeWithParameter instructions for single axis.

7.6.2.2 Axis Group State



Note 1: Applicable to all axis group movement function blocks, non-managed.

Note 2: When the axis group state is GroupErrorStop or GroupStopping, all axis group motion function blocks can be executed, but the axis group will not have any action, except for DMC_GroupResets and GroupErrorStop and any error occurs, the axis group state will change to GroupStandby or GroupErrorStop.

Note 3: When DMC_GroupStop is Done or MC_GroupStop is not Execute.

Note 4: GroupDisabled can only be transitioned in GroupStandby state. If DMC_GroupDisable is executed when the axis group status is not GroupStandby, an error will be reported.

State	Definition
GroupDisabled	Execute the MC_GroupDisable instruction to switch the axis state to GroupDisabled.
GroupStandby	No axis group motion command is executed, and the state of the axis group is GroupStandby.
GroupMoving	The axis group positioning command is being executed, and the axis group status will be GroupMoving.
GroupStopping	When the Active of MC_GroupStop is True, the state of the axis group will be GroupStopping. The axis group is in this state and cannot execute any axis group motion commands.
GroupErrorStop	Whenever an error occurs in the axis group, it will enter GroupErrorStop.

- **Interaction between single-axis state and axis group state**
  - If one axis in the group is in Errorstop and the group is not in GroupDisabled, the entire group is in GroupErrorStop.
  - When GroupMoving/GroupStopping/GroupHoming disconnects the power of an axis, the Group is in GroupErrorStop state.
  - If all axes are in StandStill state, the group can be in GroupStandby, GroupDisabled or GroupErrorStop state.
  - If the single-axis motion interrupts the group motion, the other single axes in the group will stop immediately and

enter the stop state, and the group state will enter GroupStandby.

- If the group is in the GroupStandby state, the single-axis state does not need to be in the StandStill state.
- Group movement commands (including MC_GroupStop), all single axes in the group are in the SynchronizedMotion state.
- When an error occurs during axis group movement, all axes in the axis group will stop immediately until the group enters GroupErrorStop. Each non-error uniaxial will enter the Standstill state.
- If the state of the axis group is GroupErrorStop, the state of the single axis will not be affected.

7.6.3 Execution and Status of Motion Control Instructions

AX series controllers are divided into the following two categories based on the motion function blocks:

Type	Description
MC_	PLCoopen motion control function blocks
DMC_	Delta-defined function blocks*1

The general pins in each motion control function block instruction include input, output and in/out. This section elaborates on the definition of these pins and their behaviors.

*1: Delta-defined function block instructions (DMC) include Delta-defined motion control function blocks and other administrative / non-administrative function blocks applicable to AX series motion control CPUs. For details, please refer to the **AX Series Motion Controller Manual**.

7.6.3.1 Basic Rules for Instruction Execution

- **Definition of input and output pins**

The general input and output pins of motion control function blocks are listed below. Usually, a function block contains at least one or a part of the following pins. For example, based on the properties of motion control function blocks, a function block basically contains either of the input pins, Execute and Enable.

Input			
Name	**Description**	**Data Type**	**Set Value (default)**
En	Receive logic status from instruction front end	BOOL	True / False (False)
Enable	Enable Motion Control Function Blocks	BOOL	True / False (False)
Execute	Execute motion control function block	BOOL	True / False (False)
Output			
Name	**Description**	**Data Type**	**Set Value (default)**
Eno	Transmits the logic state of the En input to the next chained command	BOOL	True / False (False)
Done	Indicates that the execution of the function block has been completed	BOOL	True / False (False)
Valid	Indicates that the value displayed by the output pin is a valid value	BOOL	True / False (False)
Busy	Indicates that the motion function block has been queued for execution	BOOL	True / False (False)
Active	Indicates that the axis is being controlled by the function block	BOOL	True / False (False)
CommandAborted	Indicates that the execution of the motion function block was interrupted	BOOL	True / False (False)
Error	Indicates the status of an error in a function block	BOOL	True / False (False)

A motion control function block usually has either Execute or Enable input pin, which is used to execute or enable the motion control function block. In addition, a motion control function block also contains Busy and Done output pins for display of its status. When the execution of a motion control function block can be interrupted by another function block, the output pin, CommandAborted, is included in this function block. Furthermore, when an error occurs during the execution of a motion control function block, the output pin Error will turn into True.

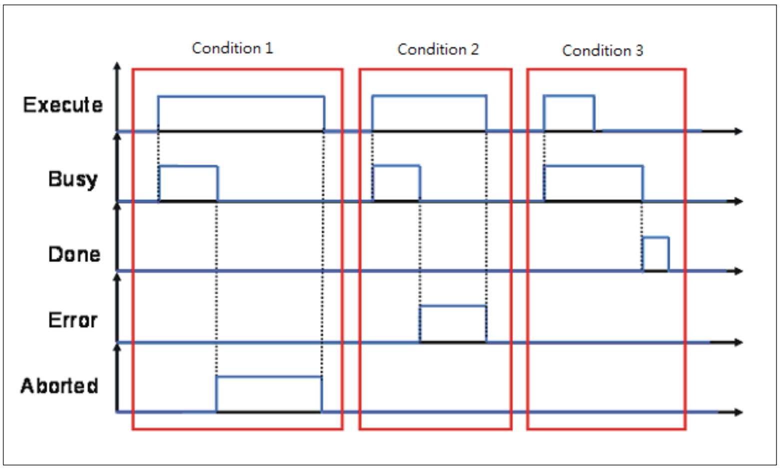
The motion control function block not only has Execute/Enable input, but may also contain input pins for input value/status, whose characteristics are described below.

- Use input value
  - For a function block with an Execute input, the value of each input pin is taken when the Execute input signal transfers from False to True. However, when Execute is retriggered, the input value will not be updated.
  - For function blocks with Enable input, the value of each input pin is taken when the Enable input signal t transfers from False to True. Compared with Execute input, function blocks of Enable input usually have more input values that need to be continuously updated (see the description of each function block for details).
- The input value is out of range
  - After the motion control function block is activated, the input of value that exceeds permitted range will be restricted by the system, or a function block error will occur during execution, resulting in errors in the motion axes. Users should avoid entering incorrect input values in the programs.
- Mutually exclusive of output pins
  - For a function block with Execute input, Busy output, Done output, CommandAborted output or Error output, only one of the states will be set True at one time. When Execute input is set to be True, one of the rest must be set True.
  - For a function block with Enable input, Valid output and Error output are mutually exclusive, which means only one of them can be set True.
- Valid time for output data/status value
  - When a function block contains Execute input and the input signal changes from True to False, the current Done output, Error output, CommandAborted output of current True and output pin data will be reset or cleared. However, when the function block is busy, even if the Execute input signal changes from True to False, the execution of the function block will not stop. The expected output states such as Done output, Error output, CommandAborted output will still generate to True and retain for one cycle.
  - For a function block with Enable input, when the Enable input signal changes from True to False, the Valid output, Busy output and Error output will be reset.
- Characteristics of Done output
  - The Done output of the motion control function block will be set to True after the motion control function block has successfully executed.
- Characteristics of Busy output
  - A function block with an Execute input uses the Busy output to indicate that the execution has not yet completed and that a new output status (value) will be generated later. When the Execute input signal transfers from False to True, the Busy output is set to True. When the Done output, CommandAborted output or Error output is set to True, the Busy output is reset.
  - A function block with an Enable input uses a Busy output to indicate that the execution is not yet completed and that a new output state (value) will be generated later. When the Enable input signal transfers from False

to True, the Busy output is set to True and if Busy outpout is True, the changes in input state (value) can still be expected.

- Characteristics of CommandAborted output
  - When the execution of a motion control function block is interrupted, the CommandAborted output is set to True.
- Relation between Enable input and Valid output
  - Function blocks with Enable input use Valid output to indicate whether the current output data/status is valid. The Valid output will be set to True only if the Enable input is set to True and the output data/status is valid. If an error occurs in the function block, the output data/status will be invalid and the Valid output will be set to False. The Valid output will be set to True only after the error in the motion control function block is eliminated and the output data/status becomes valid.

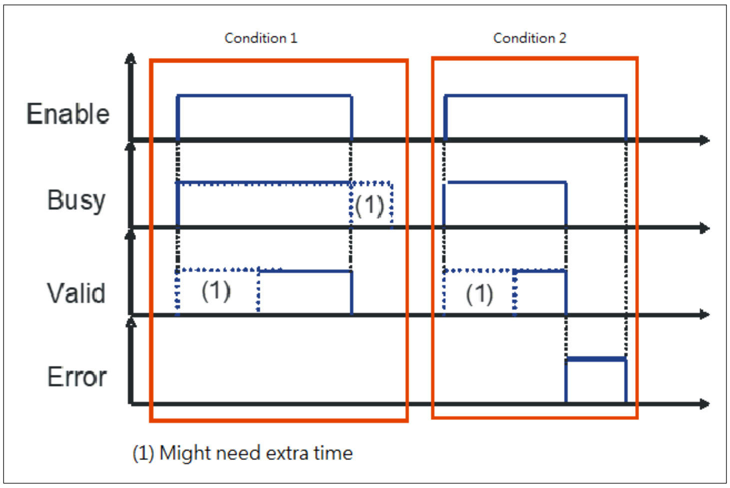
7.6.3.2 Timing Diagram for Inputs / Outputs



Condition 1: The execution of the motion control function block is interrupted.

Condition 2: An error occurs in the motion control function block.

Condition 3: The motion control function block is executed normally.



Condition 1: The motion control function block is executed normally.

Condition 2: An error occurred while controlling the function block in motion.

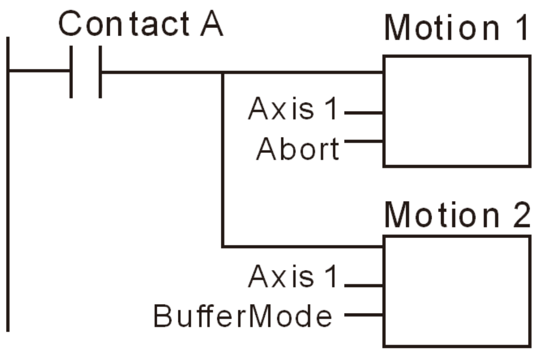
7.6.3.3 Repeated Execution of a Single Motion Control Instruction

During execution of single-axis motion function blocks (Busy state), variables of input pins can be modified to perform rising edge trigger on function block pins again. At this time, the state of the output pin of the function block remains unchanged (Busy state), but the action performed by the system is equivalent to interrupting the previous rising edge trigger instruction with Aborting in buffer mode. For similar behavior patterns, refer to **Section 7.6.3.5**.

7.6.3.4 Multi-execution of Motion Control Instructions

This section explains the situation when multiple motion control instructions for one axis or axis group are executed within the same scan period.

- In the following, when contact A is ON, motion 1 and motion 2 corresponding to axis 1 start in the same task period.
- According to the ladder diagram logic, instructions are executed in a top-down order. As a result, motion 1 will be started first and followed by motion 2 after completion.
- This situation is regarded as multi-execution of motion control instructions. As the motion combination is determined by input variable, BufferMode, it is the BufferMode setting in Motion 2 that decides the combination mode between Motion 1 and 2.



7.6.3.5 Synchronous Execution of Motion Control Instructions

Single axis with SoftMotion V4.10.0.0[*1] (inclusive) and above, and SM3_Basic V4.10.0.0 (inclusive) and above supports Buffered Mode feature; DL_MotionControl V1.2.0.0 (inclusive) and above supports SoftMotion V4.10.0 .0.

*1: In SoftMotion V4.10.0.0, modification of parameters in Axis_REF is prohibited; if any, the error message **SMC_MOVING_WITHOUT_ACTIVE_MOVEMENT** will appear.

- **Buffer mode**

For the same axis, you can activate another motion control instruction when the axis is in motion. There are six types of buffer mode you can choose from to proceed with multi-execution of the instructions. The BufferMode is selected based on the BufferMode parameter setting of the later motion instruction.

The meanings of the terms related to BufferMode are as follows:

- Current instruction: The motion control instruction at runtime right before the mutil-execution instruction.
- Buffered instruction: The motion control instruction waiting to be executed.
- Transit velocity: The velocity applied in transition from current instruction to buffered instruction.
- Target velocity: Velocity parameter of the instruction.
- Target position: Position or Distance parameters in instructions related to displacement.

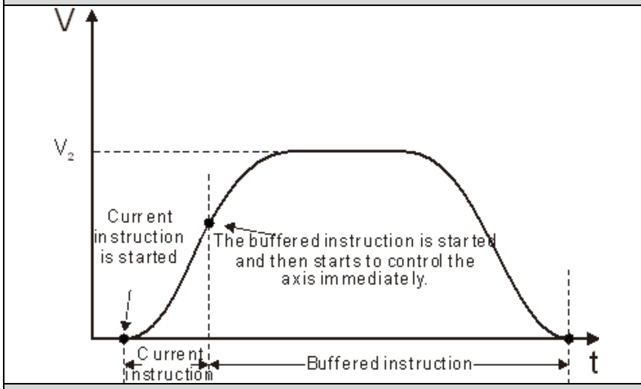
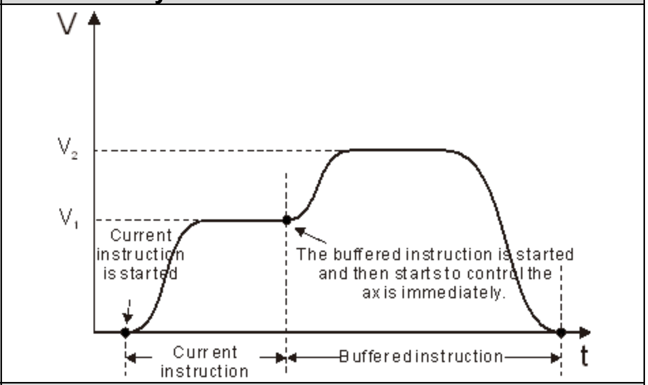
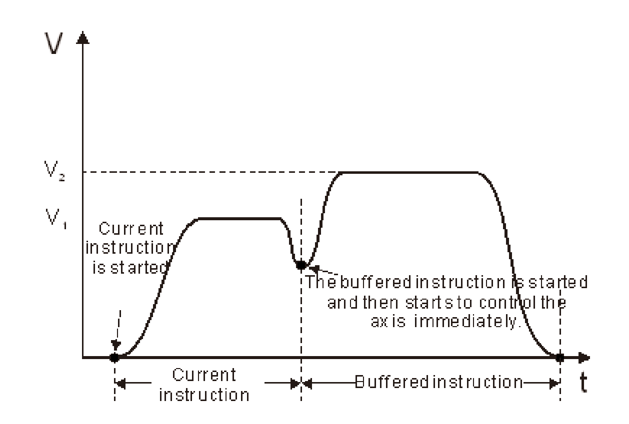
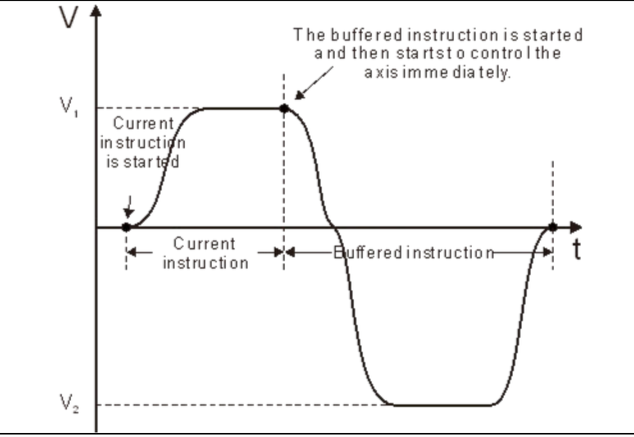
BufferMode	Operation
0: mcAborting (Aborting)	The current instruction is immediately aborted to execute the buffered instruction.
1: mcBuffered (Buffered)	The buffered instruction is not executed until the current instruction is completed as expected.
2: mcBlendingLow (Low velocity)	The buffered instruction is executed after the target position of the current instruction is achieved. The transit velocity is set to be the target velocity of either the current or the buffered instruction, whichever is lower.
3: mcBlendingPrevious (Previous velocity)	The buffered instruction is executed after the target position of the current instruction is achieved. The transit velocity is the target velocity of the current instruction.
4: mcBlendingNext (Next velocity)	The buffered instruction is executed after the target position of the current instruction is achieved. The transit velocity is the target velocity of the buffered instruction.
5: mcBlendingHigh (High velocity)	The buffered instruction is executed after the target position of the current instruction is achieved. The transit velocity is set to be the target velocity of either the current or the buffered instruction, whichever is higher.

- **Example: a brief introduction with two relative displacement instructions**

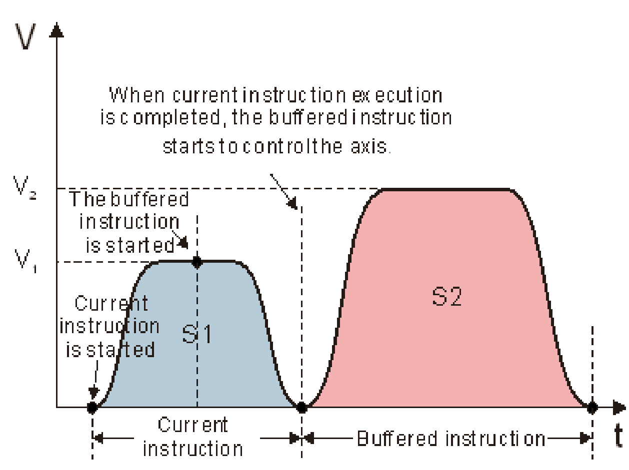
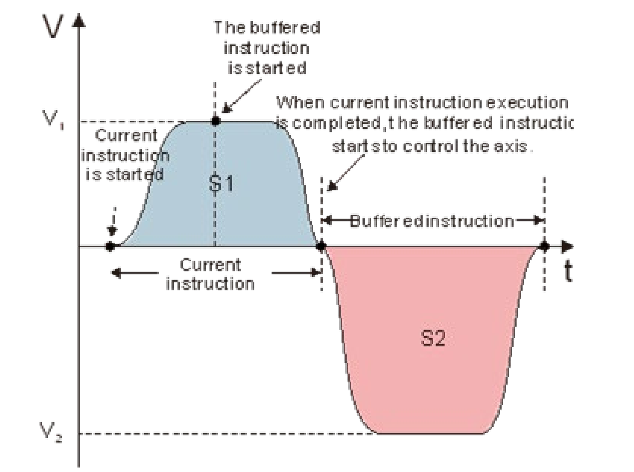
Here, the maximum velocity of the first relative displacement instruction is V1 and the displacement is S1 while the maximum velocity of the second relative displacement instruction is V2 and the displacement is S2.

By changing the BufferMode of the second displacement instruction, it results in different transiting processes as shown in the figures below:

■ BufferMode=mcAborting

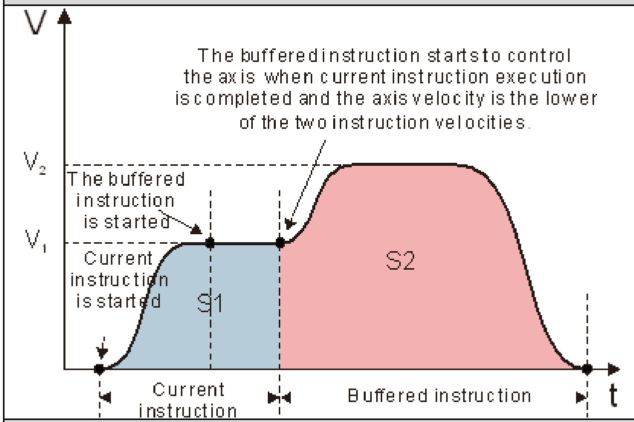
1. The current instruction being aborted during acceleration	2. The current command being aborted at constant velocity
	
3. The current instruction being aborted during deceleration	**4. The current instruction and the buffered instruction are in opposite directions**
	

■ BufferMode=mcBuffered

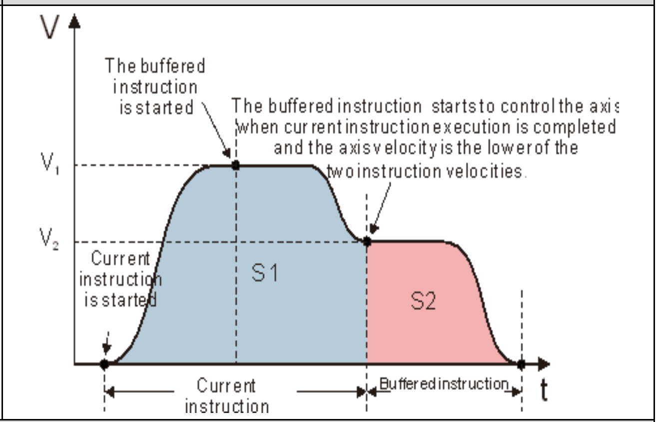
1. The current instruction and the buffered instruction are in the same direction	2. The current instruction and the buffered instruction are in opposite directions
	

■ BufferMode=mcBlendingLow

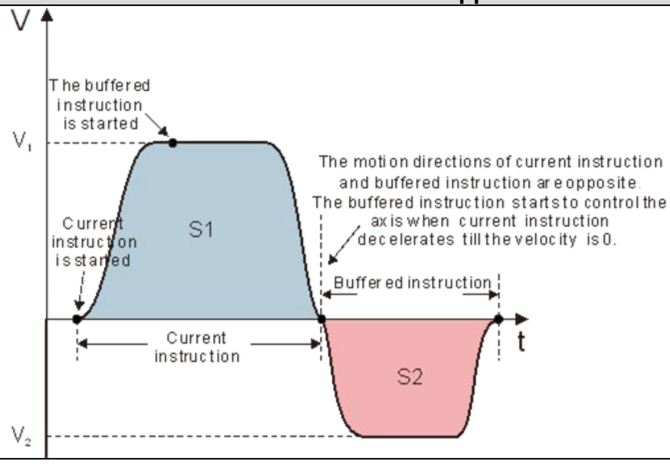
1. The velocity of the current instruction is lower than that of the buffered instruction



2. The velocity of the current instruction is higher than the buffered instruction

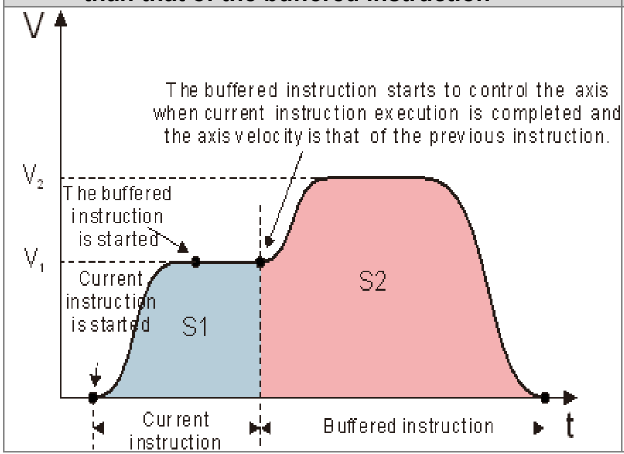


3. The current instruction and the buffered instruction are in opposite directions

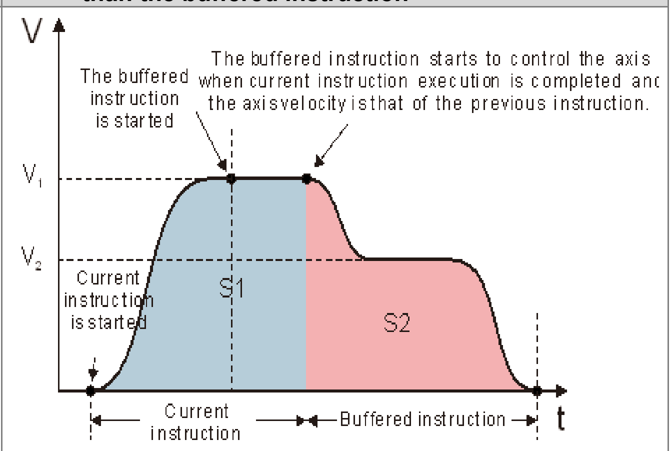


■ BufferMode=mcBlendingPrevious

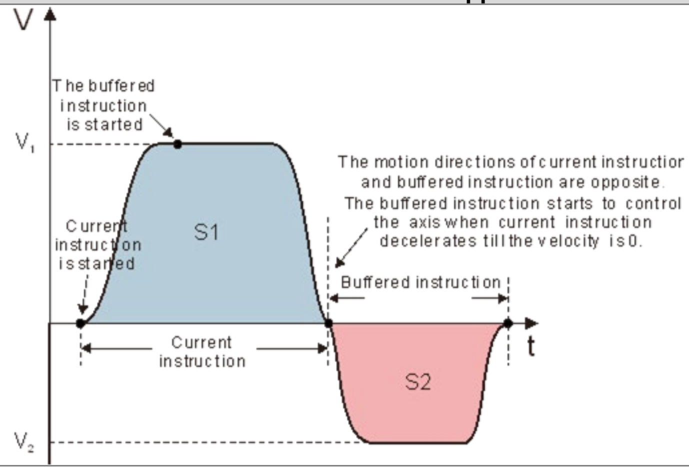
1. The velocity of the current instruction is lower than that of the buffered instruction



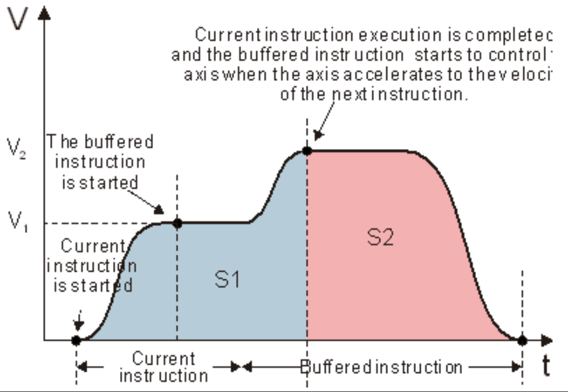
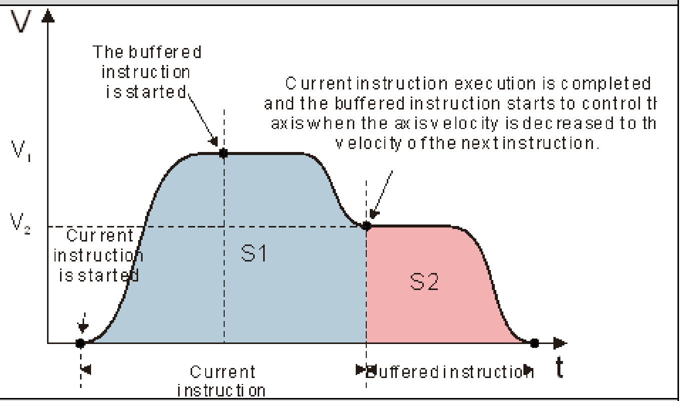
2. The velocity of the current instruction is higher than the buffered instruction



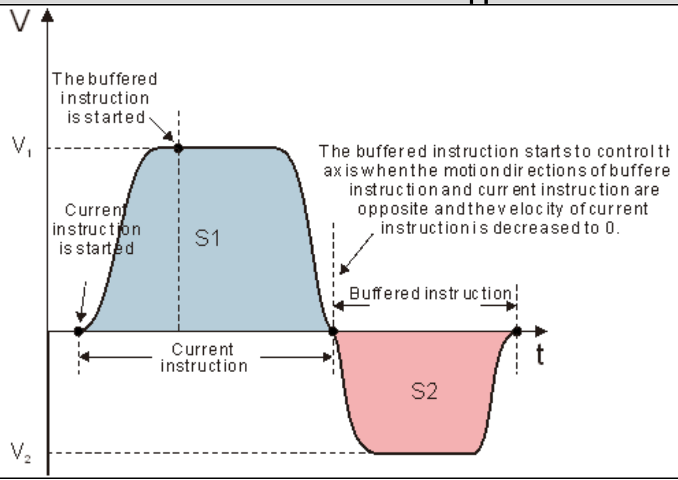
3. The current instruction and the buffered instruction are in opposite directions



- BufferMode=mcBlendingNext

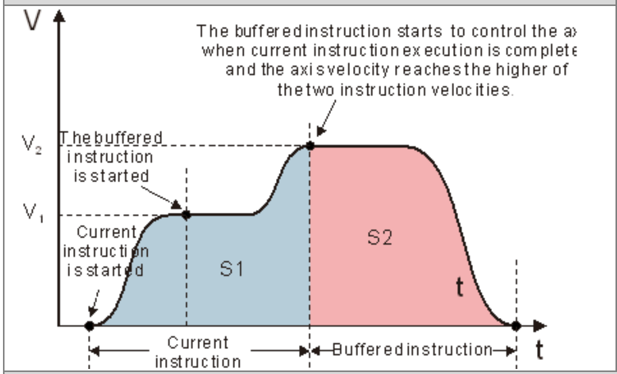
1. The velocity of the current instruction is lower than that of the buffered instruction	2. The velocity of the current instruction is higher than the buffered instruction
	

3. The current instruction and the buffered instruction are in opposite directions

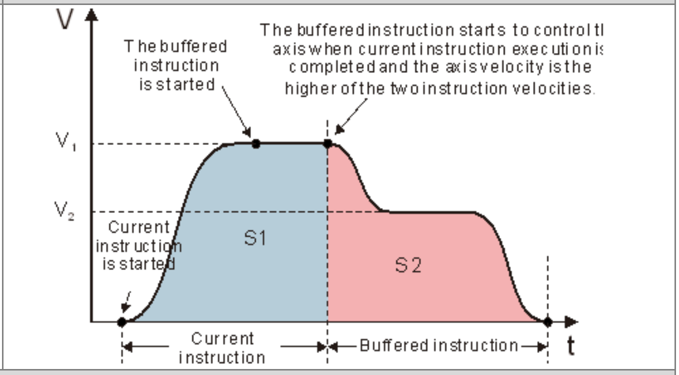


■ BufferMode=mcBlendingHigh

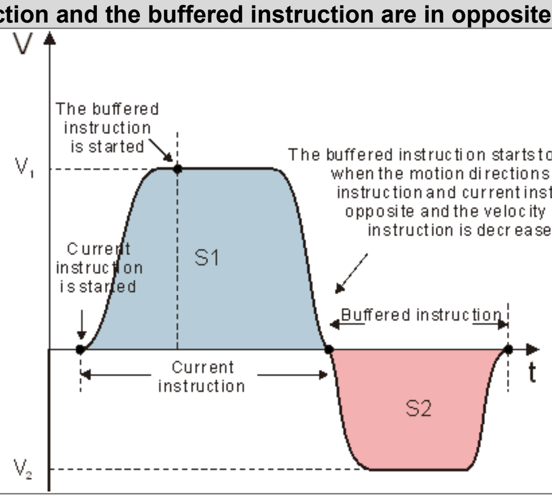
1. The velocity of the current instruction is lower than that of the buffered instruction



2. The velocity of the current instruction is higher than the buffered instruction

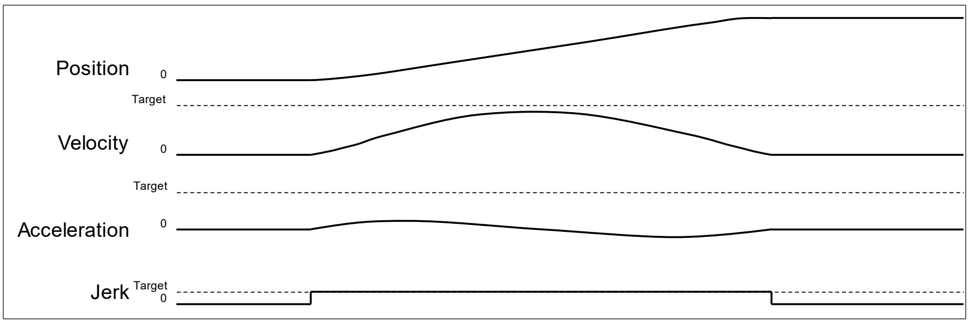


3. The current instruction and the buffered instruction are in opposite directions



■ BufferMode curve is planned mainly based on input parameters of function block by satisfying the sequence: Jerk → Acceleration / Deceleration → Position → Velocity.

For example, when Position < Velocity, Velocity = Acceleration / Deceleration, and Jerk < Acceleration / Deceleration, the curve planning is as follows:



Due to insufficient path, the planned Acceleration, Deceleration, and Velocity will not reach Target as Jerk does.

7.6.4 Position

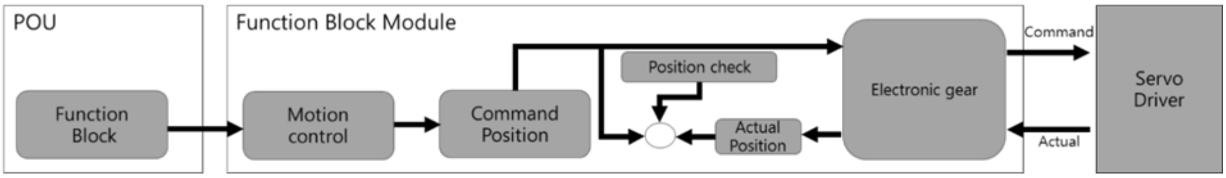
This section introduces the positioning process of motion control programming.

7.6.4.1 Types of Positions

The MC function blocks are composed of the following two types of positions:

1. Command position: The command position given by the motion function block.
2. Actual position: the feedback position from servo drives.

The figures below illustrate the relation between the command position and the actual position:



The following items are the same for the command position and the actual position.

Type	Definition
Command position	The position output by the motion controller to servo drive
Actual position	The position feedback from servo drive or encoder

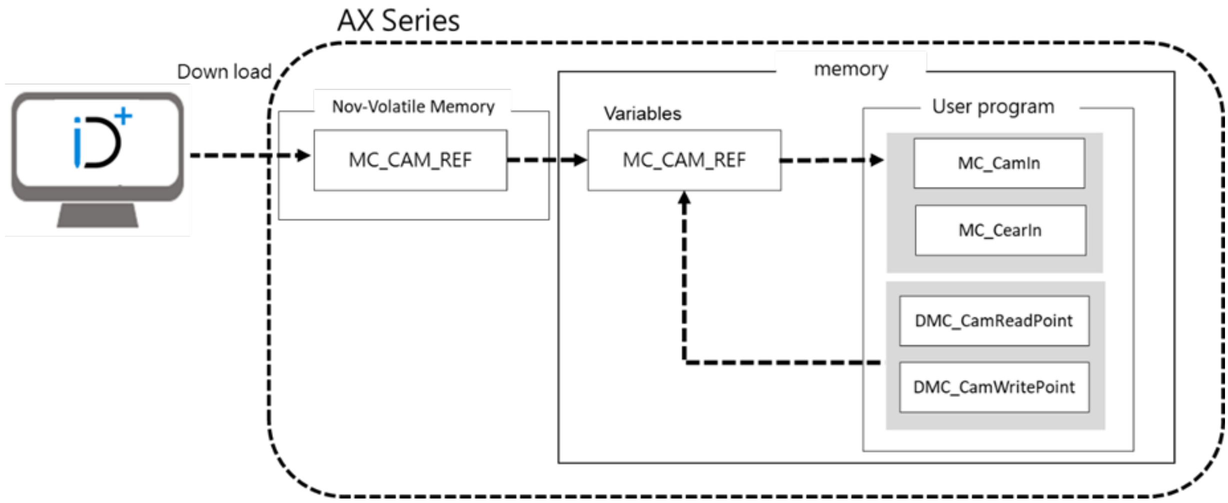
Note: For axes configured as virtual, the command position is equal to the actual position.

7.6.5 Cam Tables and Framework

This section introduces electronic cam (E-CAM) operation as well as applications, and cam table generation as well as setting through DIADesigner-AX software. For detailed instructions, refer to the **AX Series Motion Controller Manual**.

7.6.5.1 E-CAM Framework

Adopt cam Editor of DIADesigner-AX for cam curve planning and download to PLC through communication protocols so that MC function blocks can be used to control cam.



7.6.5.2 E-CAM Establishment

The data that defines the relation between the master / slave (cam shaft) is called E-CAM data.

To use cam Editor of DIADesigner-AX, it is crucial to know the relation between master and slave positions via the following two methods.

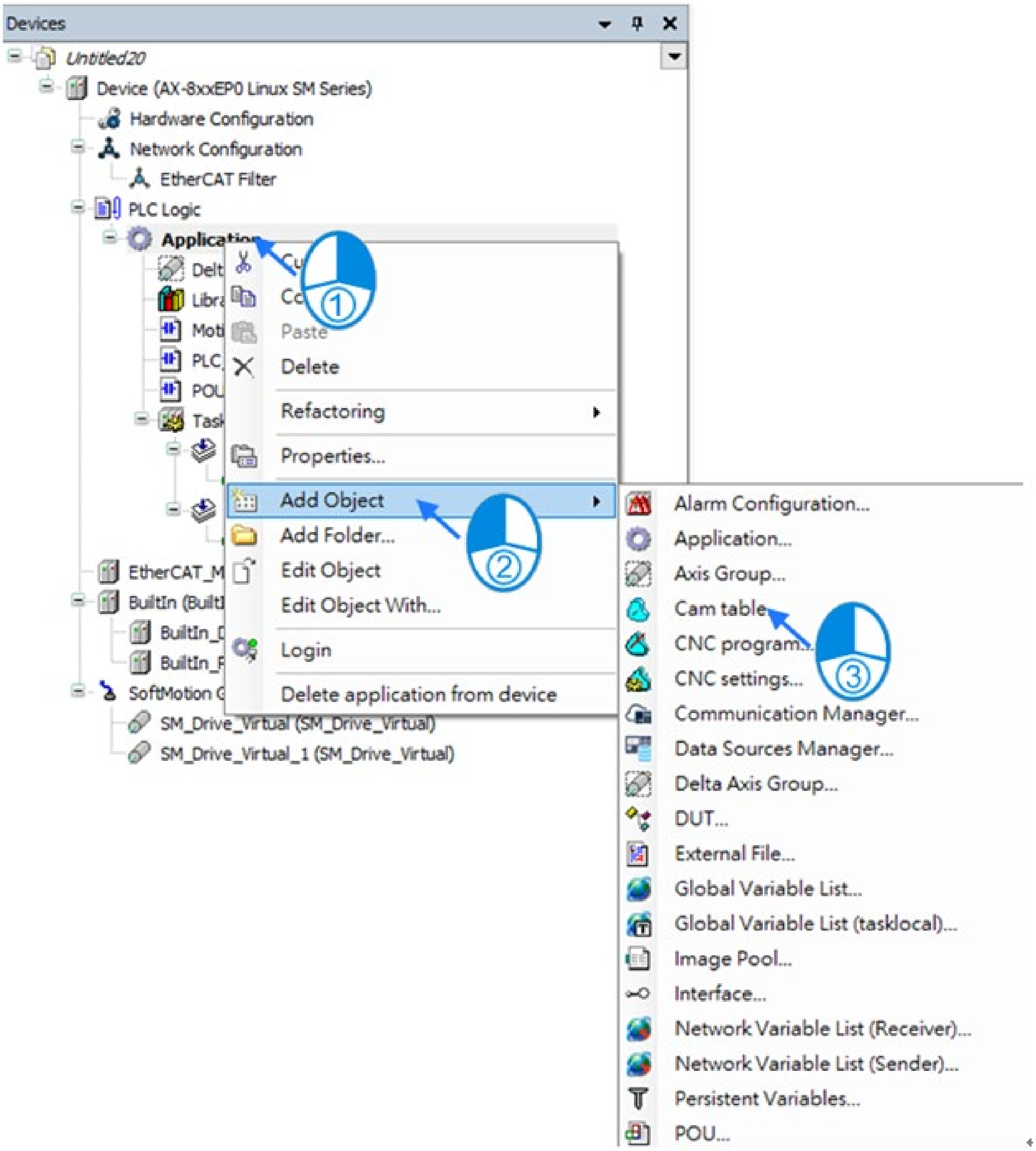
Method 1: Obtain the connection between positions of master and slave axes based on E-CAM data setting.

Method 2: Measure the correspondence between positions of master and slave axes from actual work.

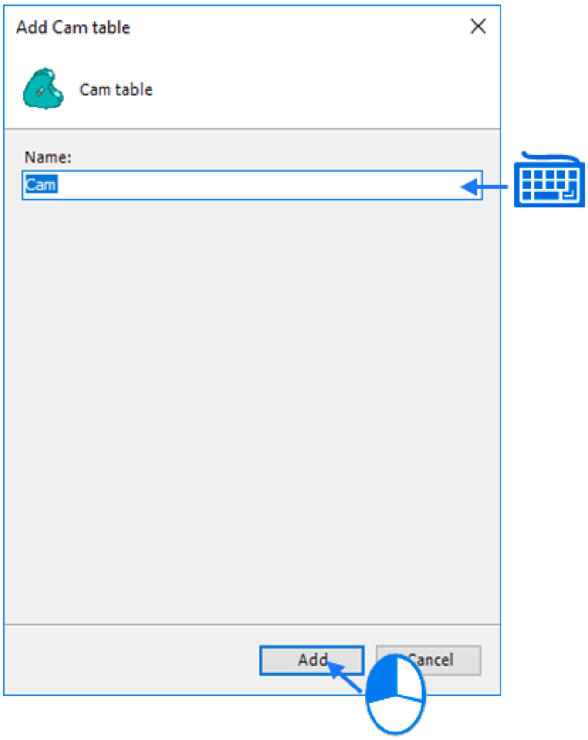
After the relationship between cam master and slave is confirmed, the slave axis position can be obtained based on the the master axis position.

- **Creating DIADesigner-AX cam tables**

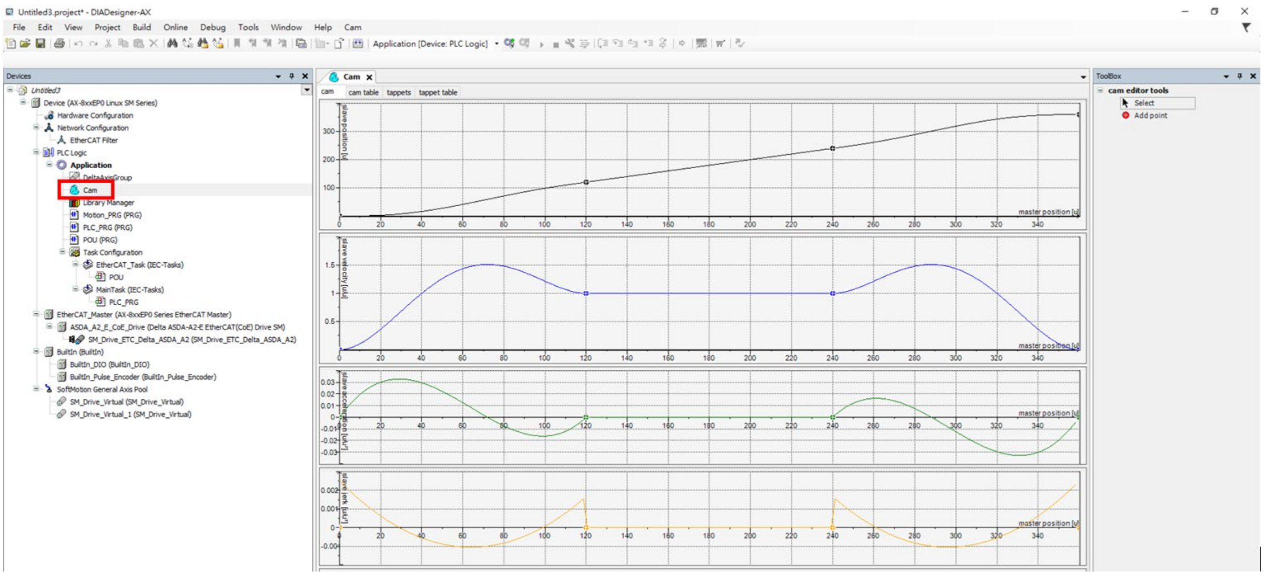
1. Right-click **Application** on the left, select **Add Object**, and then select **Cam table**.



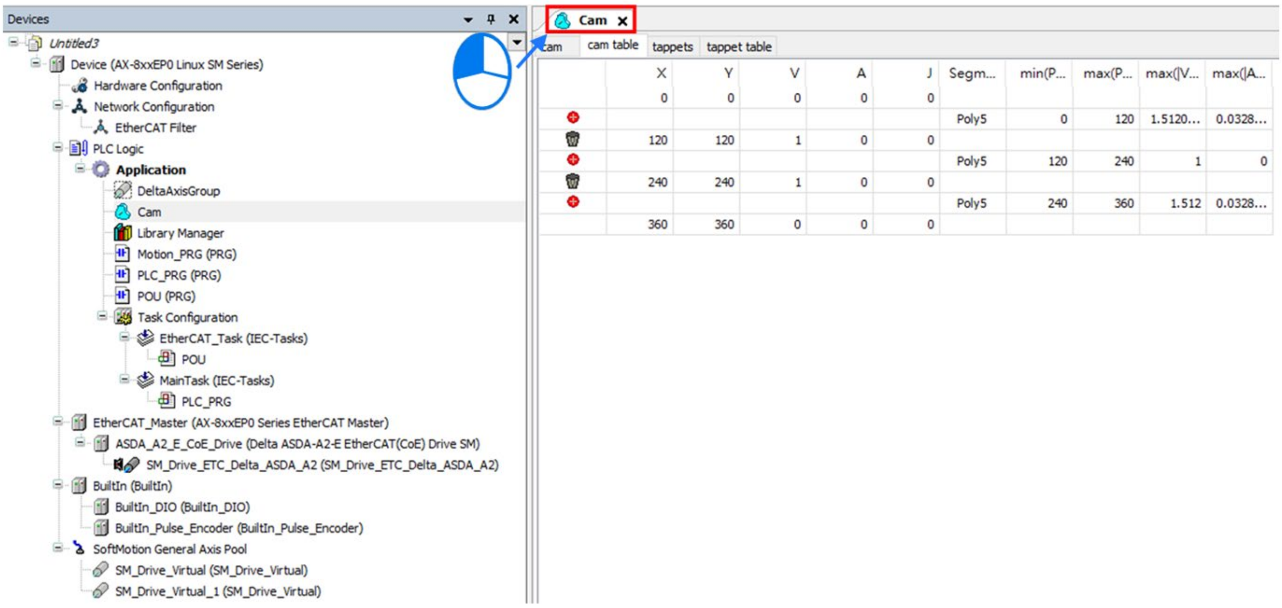
2. In the **Add Cam table** window, **Name** refers to the name of the cam table, which can be modified by yourself.



3. After clicking **Add**, you can see the **Cam** icon on the left.

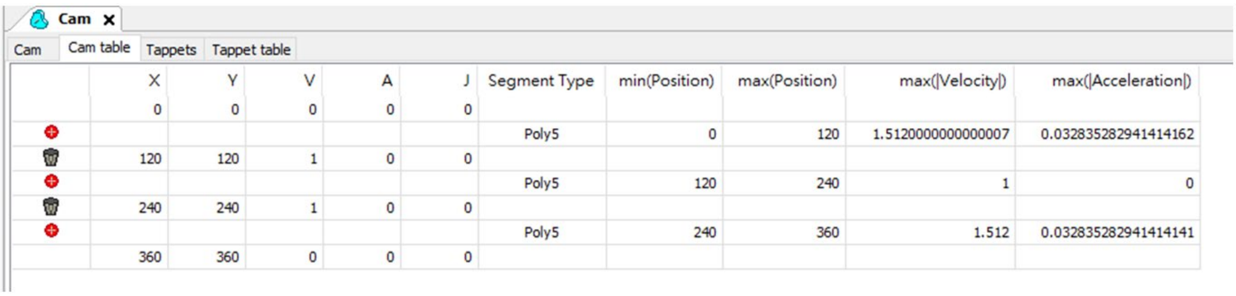


4. Click **Cam table** on the **Cam** page.

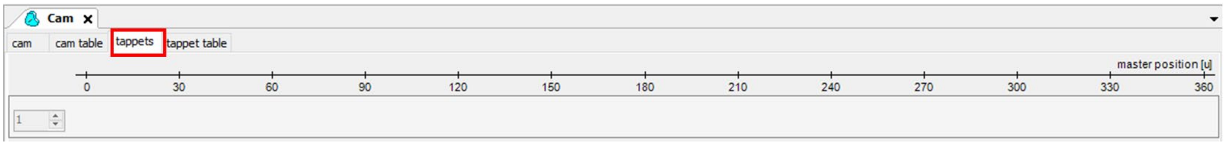


5. On the **Cam Table** screen, you can add or delete cam data.

- Click to add new cam data
- Click to delete cam data
- X: Spindle Point
- Y: Slave Axis Point
- V: Slave Axis Speed
- A: Slave Axis Acceleration
- J: Slave Axis Jerk (Jerk)
- Segment Type: Curve Type

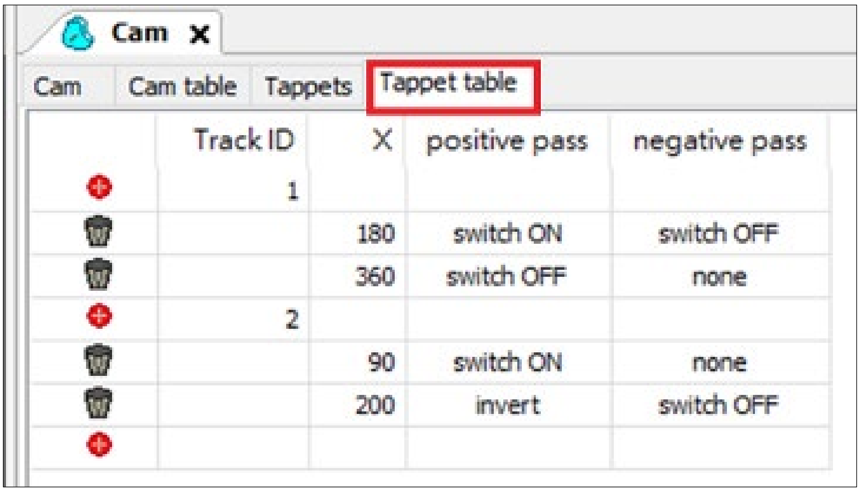


6. Multiple tappets can be configured on the **Tappets** page, and several tappet points can be set for each tappet ID. After setting the **Tappet table**, a diagram which illustrates the relation between tappets and master axes would be shown on **Tappets** page. While dragging the points on **Tappets** page, the setting parameters on the **Tappet table** page change simultaneously.



7. You can configure tappets on Tappet table screen and use SMC_GetTappetValue to read the status of the tappet, which can also be modified according to the settings in Tappet table and the direction when cam master passes the tappets.

- Click to add a Track ID.
- Click to delete Track ID.
- Track ID: track ID
- X: Tappet point position
- Positive pass: When forward rotation encounters the tappet point action, the action settings are as follows:
  - None: no action
  - Switch to ON: TRUE
  - Switch to OFF: FALSE
  - Invert: reverse
- Negative pass: When the reversal encounters the tappet point action, the action setting is as follows:
  - None: no action
  - Switch to ON: TRUE
  - Switch to OFF: FALSE
  - Invert: reverse



	Track ID	X	positive pass	negative pass
	1			
		180	switch ON	switch OFF
		360	switch OFF	none
	2			
		90	switch ON	none
		200	invert	switch OFF

7.7 Motion Control Functions

7.7.1 System Structure

The single axis motion instructions of MC function blocks can be used in three control modes - velocity control, position control, and torque control - to plan the motion trajectory of the axis according to the parameters set by the user.

The AX series controllers support the CoE (CANopen over EtherCAT) application protocol. According to the CiA402 standard, the motion modes supported include Cyclic Synchronous Position Mode, Cyclic Synchronous Velocity Mode, and Cyclic Synchronous Torque Mode, which will be further explained in the following sections.

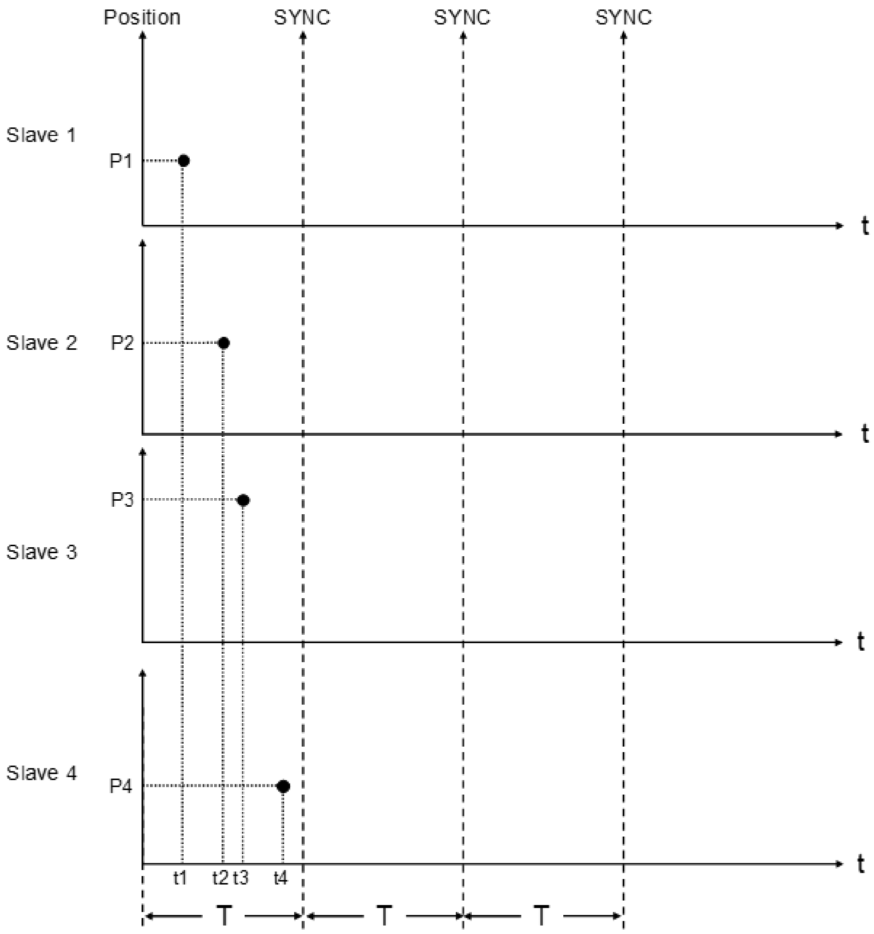
7.7.2 Single-axis Control

7.7.2.1 Cyclic Synchronous Position Mode

The data synchronization between AX series controllers and servo drives is accomplished via sync signal transmission from controllers to servo drives. These incoming data will not take effect until Distributed Clock (DC) in each servo drives are synchronized.

As shown in the figure below, four servo drives are connected and sent with control data at different timing (t1, t2, t3, t4) within one synchronous cycle (T). However, the control data become valid only when all servo drives are synchronized with the SYNC event of the DC system.

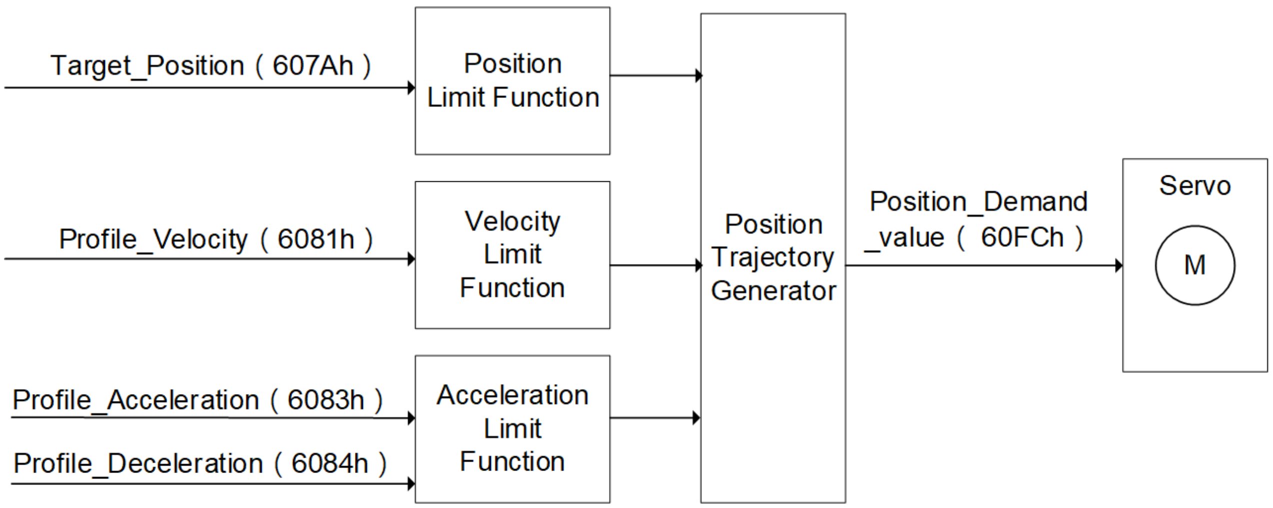
*1: Cyclic synchronous position mode is used for synchronous axes.



7.7.2.2 Profile Position Mode

After the servo drive receives the position command sent by the master device, it drives the motor to reach the target position. In profile position mode*1, the master device only informs the drive of configuration about target position, velocity command, acceleration and deceleration at the beginning; the mtion plannings, from activation of command to achievement of target position, are executed by the trajectory generator inside servo drive.

*1: The profile position mode is used for positioning axes.

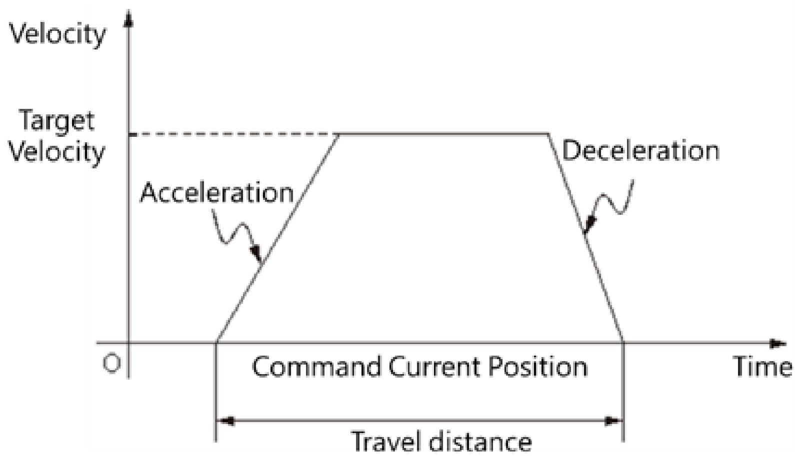


7.7.2.3 Positioning

- **Absolute positioning**

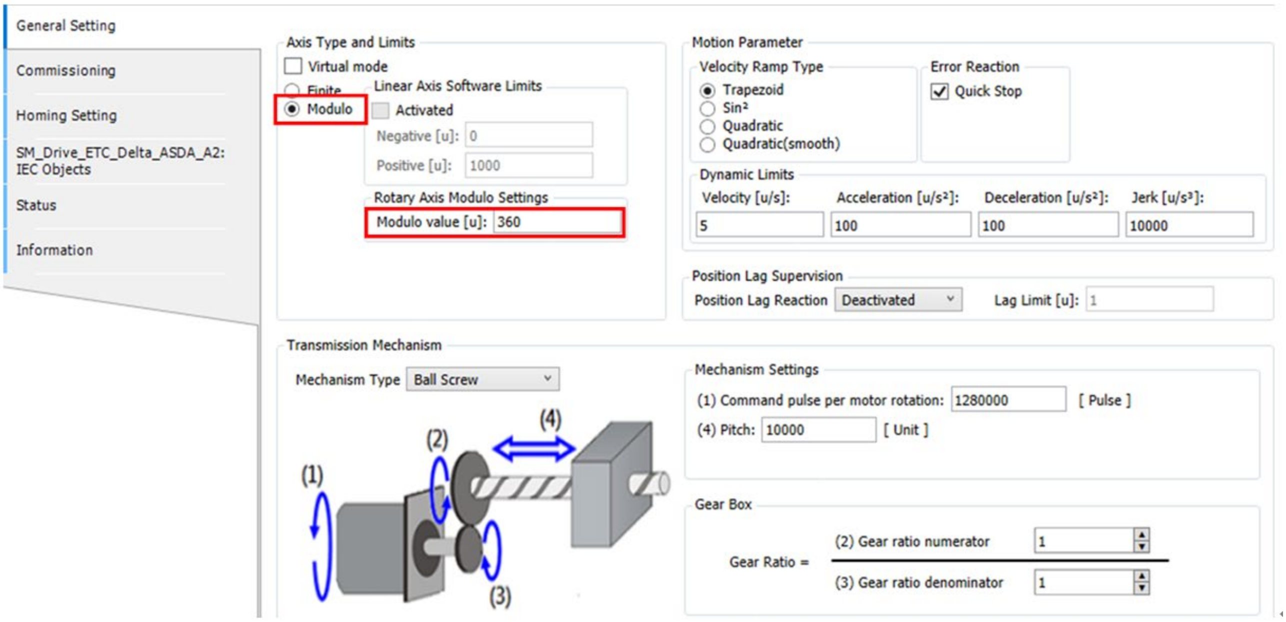
The curve for motion planning is based on the input value set by the user and the current position. It enables the axis to move to the absolute coordinate of the target position with the origin as the reference point. In addition, the range of the absolute position for modulo axis is limited to the range of its rotation period. For more information, refer to the MC_MoveAbsolute function block.

The motion trajectory of absolute positioning is shown in the figure below.



■ Rotary axes setting

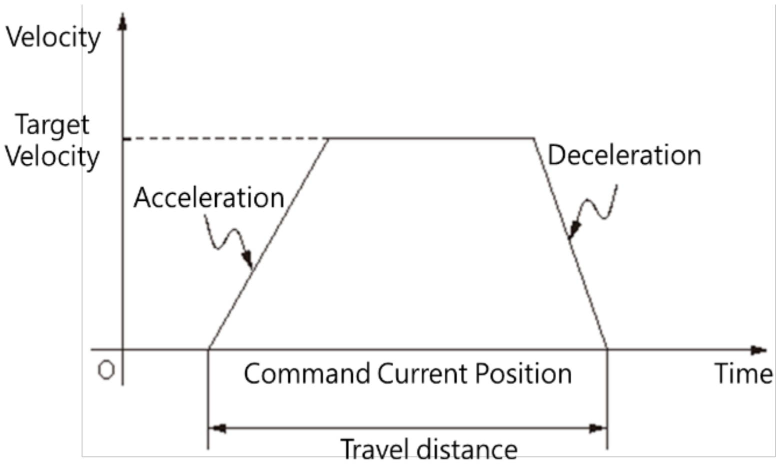
- After selecting **Rotary Axis** for axis type, set the range of rotation for the rotary axis in **Modulo value**.



- **Relative Positioning**

The curve for motion planning is based on the input value set by the user and the current position. It enables the axis to move to the relative coordinate of the target position in relation to the current actual position. For more information, refer to the MC_MoveRelative function block.

The motion trajectory of relative positioning is shown in the figure below.



7.7.2.4 Stop Method

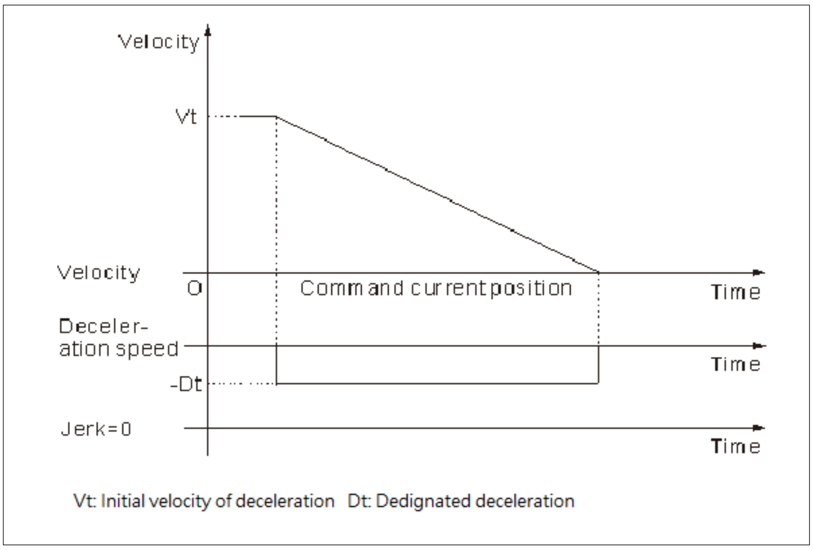
The stopping state can be achieved by giving motion control instructions, or enabling limit input which results in error stop.

- **Using motion control instructions to stop**

  To stop single-axis movement, use MC_Stop or MC_Halt.

  - MC_Stop
    - MC_Stop stops the axis in motion according to the specified method and changes the axis state to **Stopping**.
    - This instruction aborts any instructions in progress. When the axis state is **Stopping**, no instructions can be executed.
    - The state **Stopping** continues until velocity reaches 0 or Execute becomes False. When velocity reaches 0, Done turns into True.
    - When Done becomes True and Execute becomes False, the axis will enter **Standstill** state.

    The motion trajectory of MC_Stop is shown in the figure below. The velocity is determined according to the specified deceleration (Dt).



  - MC_Halt
    - MC_Halt temporarily stops the axis in motion and switches the axis state to **DiscreteMotion** until the axis velocity reaches 0. After the axis stops, the state turns into **Standstill**.
    - During deceleration, other motion instructions can be executed to abort MC_Halt operation.

- **Limit input stop**

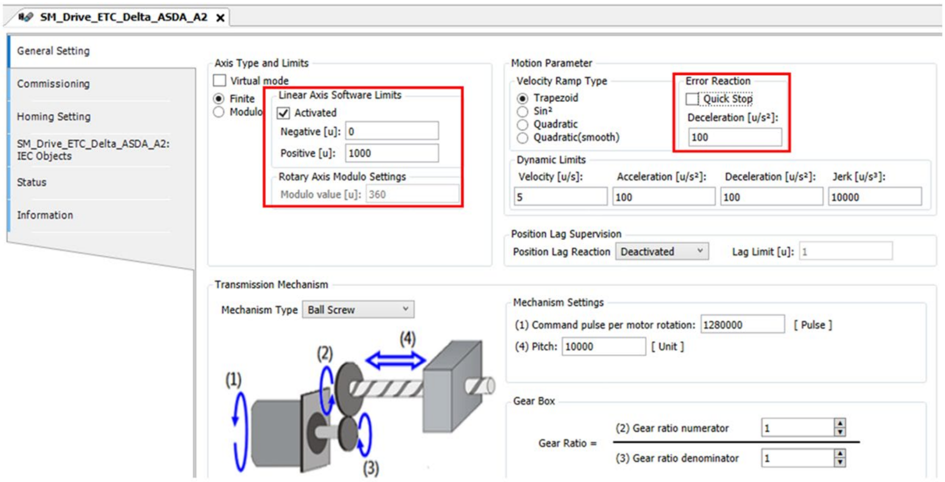
  - Software limit

    Sotware limit can be activated / deactivated and configure its parameter settings on axis parameter setting page.

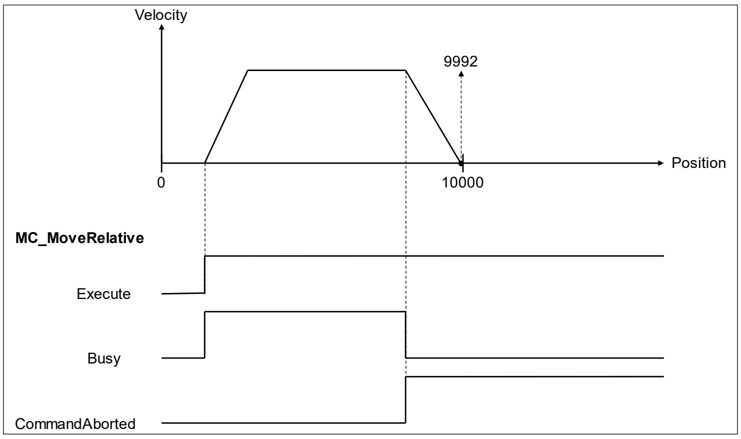
    When the axis is approaching software limit, it will start decelerating based on the axis parameters and stop within the software limit.

    Example:

    - Set the positive limit to 10,000 and the negative limit to 0 with software limit **Activated**. Then set Deceleration to 1,000 (the item appears after Quick Stop is unchecked).



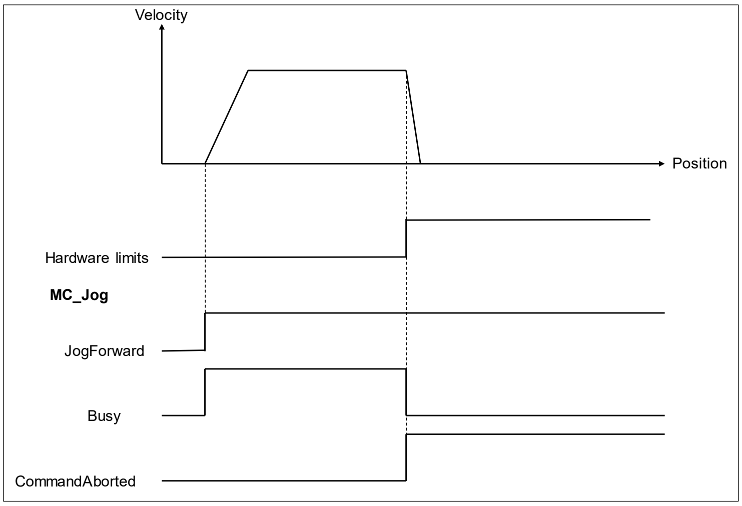
- Use function block MC_MoveRelative and activate it after the position reaches 11,000. When the axis moves to around 8,000, Busy of the function block will shift from TRUE to FALSE while CommandAborted will shift from FALSE to TRUE. The axis then starts decelerating till stop within the software limit.



■ Hardware limit

As hardware signals are carried in the EtherCAT servo wires, the stop methods for hardware limit differ from manufacturers to manufacturers. In the following, Delta ASDA-A2-E is taken as an example for further explanation.

- Use MC_Jog to perform forward rotation. Once the ahardware limit is reached during axis rotation, ASDA-A2-E servo drive will be stopped and error messages will be reported through communication.

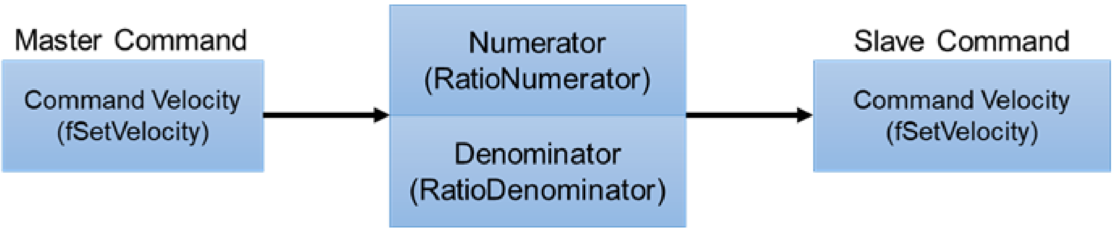


7

After reaching software / hardware limit, you can use MC_Reset to clear errors. The system will automatically synchronize the command position based on the values of the feedback position and keep the axis moving in the direction away from the limit so as to resume normal operation.

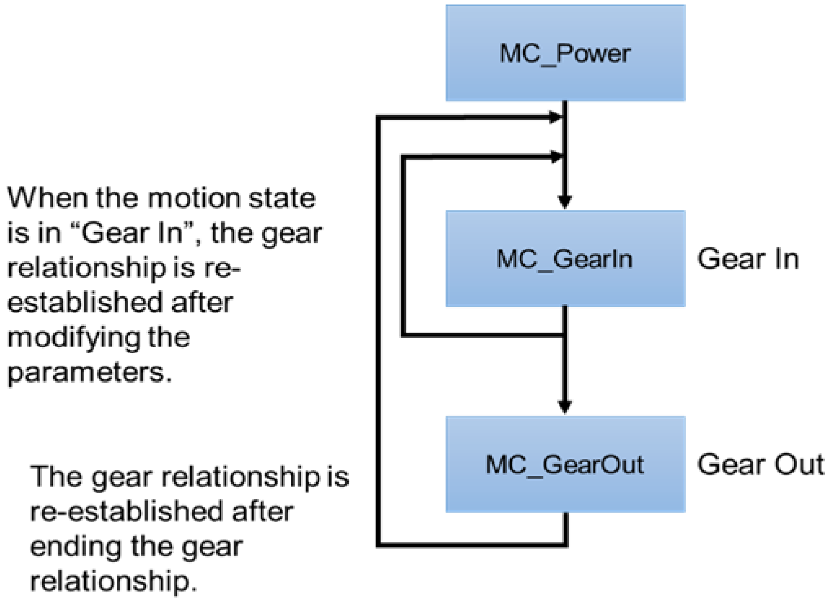
7.7.2.5 MC_GearIn

Use MC_GearIn instruction to control gear movements and MC_GearOut instruction to cancel synchronization between master and slave axes. Please refer to the **AX Series Motion Controller Manual** for details about function blocks.



You can specify the master and slave axes, gear ratio numerator and denominator, acceleration and deceleration as well as jerk in MC_GearIn.

- **Execution order of instructions for electronic gears**



When executing MC_GearIn, the slave axis enters the state of synchronized motion; however, for MC_GearOut execution, the slave axis gets out of the synchronized state, the slave axis will continue to move at the speed at the moment of separation, and the slave axis state machine enters Continuous Motion.

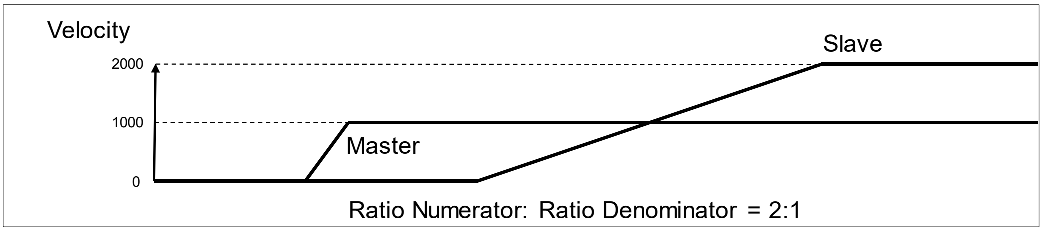
During synchronized motion, if executing MC_Stop on the slave axis, MC_GearIn will be aborted. The master axis maintains in Continuous Motion state while the slave axis enters Stopping state and will not return to Standstill until the MC_Stop is Done.

When the slave axis is in the state of Synchronized motion, its velocity alters according to the velocity of the master axis through adjustment of gear ratio.

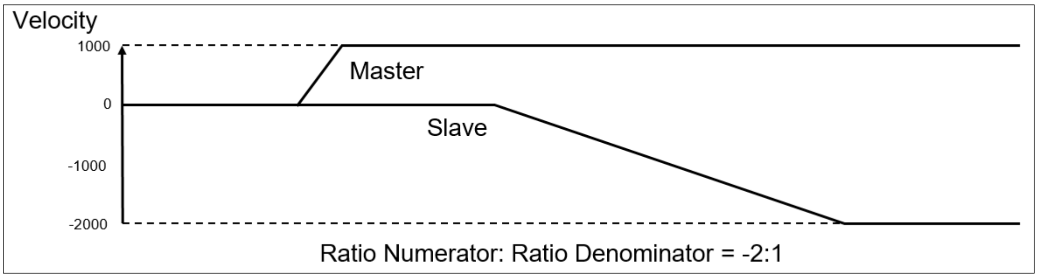
When the master axis and the slave axis are in synchronization state, use MC_SetPosition to avoid accidents caused due to rapid rotational speed of motors.

- **Set the gear ratio between master and slave axes through RatioNumerator and RatioDenominator in MC_GearIn.**

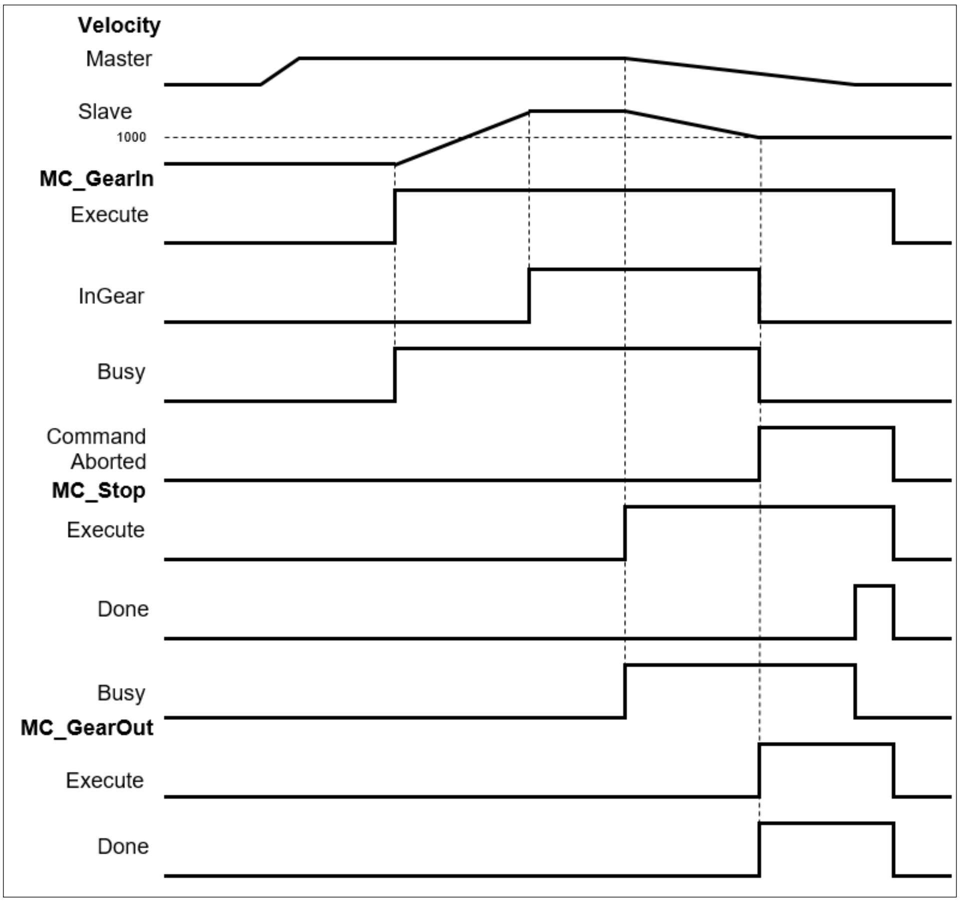
When the gear ratio is positive, the master and slave axes move in the same direction.



Conversely, when the gear ratio is negative, the master and slave axes move in opposite directions.



- **Synchronization of master and slave axes is achieved when the slave axis velocity reaches the value set in the instruction.**



When MC_GearIn is enabled, the slave axis starts to engage with the master axis and its velocity is twice the master's velocity (RatioNumerator: RatioDenominator = 2:1).

When InGear is True, the master and slave axes are synchronized and the slave axis state enters Synchronized Motion.

When MC_Stop is enabled, the master axis start decelerating and the slave axis also decelerates synchronously according to the gear ratio.

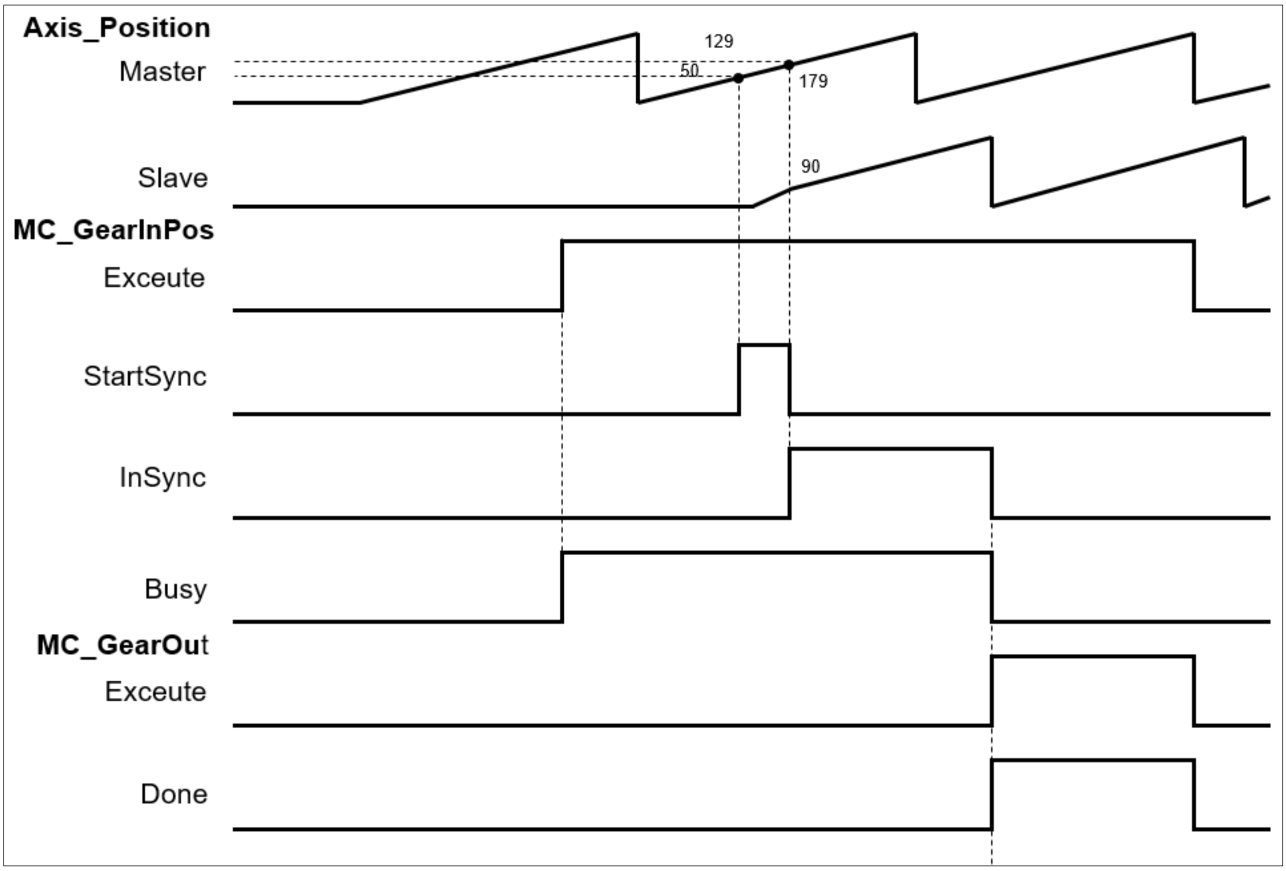
When MC_Stop is in operation and MC_GearOut is enabled, the master and slave axes are out of synchronization. The slave axis keeps moving at the velocity when synchronization is aborted and then enters the state of Continuous Motion.

7.7.2.6 MC_GearInPos

MC_GearInPos is adopted to assign the starting position for synchronization of master and slave axes.

- **MC_GearInPos sequence**

In MC_GearInPos, it is allowed to assign master axis and slave axis, gear ratio numerator and gear ratio denominator, synchronous starting position, master axis distance when starting synchronization and whether to enable reversal. The function block will automatically complete planning motion curve of the slave axis according to the settings, so as to engage the master and slave axes at the specified position.



The position where the master axis starts to synchronize is MasterSyncPosition (180) - MasterStartDistance (50). When the master axis reaches this position, StartSync ⇒ True starts to synchronize.

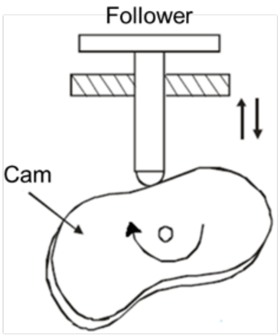
The slave axis generates a motion curve on its own based on other parameters. When the master axis reaches MasterSyncPosition (180), the slave axis also reaches SlaveSyncPosition (90). At this time, StartSync is False and InSync is True.

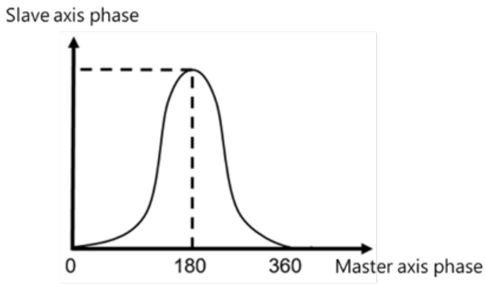
When MasterStartDistance ≦ 0, the function block is completed upon execution. In the meantime, the position of the slave axis jumps directly to the specified synchronous position.

If the slave reversal is not allowed, the user must set AvoidReversal to True.

7.7.2.7 MC_CamIn

The slave axis synchronizes with the master axis based on cam table. You can use MC_CamTableSelect to pre-assign the cam table to follow and designate master and slave axes. In addition, the instruction MC_CamIn can be utilized for cam engagement whereas MC_CamOut is used to remove gear engagement.





After engagement, synchronization between master and slave axes is achieved and the state of slave axis is Synchronized Motion. The following describes the information regarding E-CAM establishment:

- **Initial settings**
  - Create E-CAM data

  The E-CAM curve data can be drawn by the following two methods:

  Method 1: The relation of master and slave positions is determined according to standard functions.

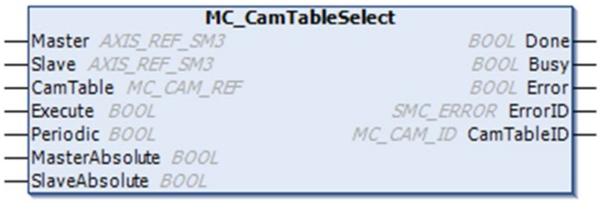
  Method 2: The corresponding relation between the master and slave axes is measured in actual work.

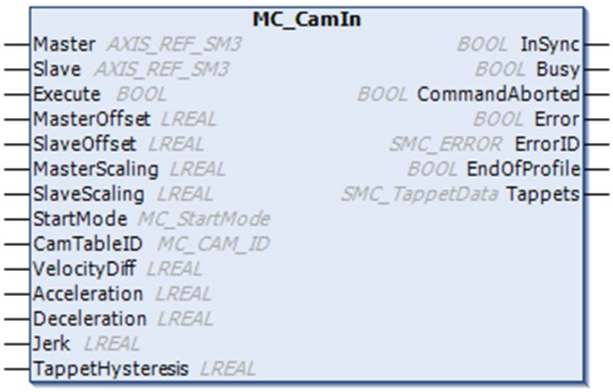
- **Setting and operation of E-CAM master and slave axes**

By using MC_CamIn and MC_CamTableSelect, you can set up E-CAM master and slave axes and basic operation.

  - Master and slave axes source setting

  In function blocks, MC_CamTableSelect and MC_CamIn, the master source is determined by the input pin Master while the slave source is determined by the pin Slave*1.



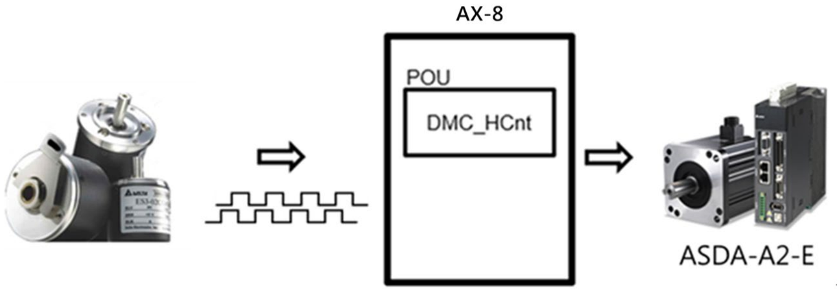


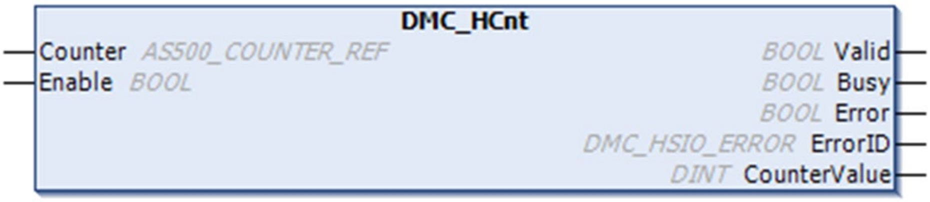
*1: For information on pin function, refer to the **AX Series Motion Controller Manual**.

- An external counter as the master axis

The sources of E-CAM master axis are actual and virtual axes as well as counters. Using an external counter as the master source, it is required to activate external pulse counting function via DFB_HCnt function block.

System structure and DMC_HCnt:





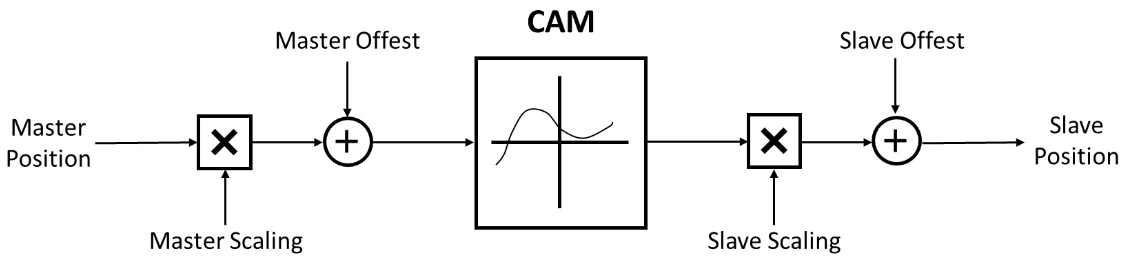
- The relation between master and slave positions

The cam relation pre-planned via the software refers to the relative positions of master and slave axes instead of their actual positions. If the pre-planned cam relation is regarded as cam function, its input is the cam master phase, and the output is the cam slave phase.

x: cam master phase; y: cam slave phase; y = CAM(x)

The cam phase is derived from the axis positions in which conversion may take place. The conversion in between is related to the parameters MasterAbsolute, SlaveAbsolute, MasterOffset, SlaveOffset, MasterScaling, and SlaveScaling.

Under the instruction MC_CamIn, the slave axis follows the master axis to perform synchronized motion. The relation between the master and slave positions is built on the pre-planned cam relation (cam curve or cam table). The process to derive position of the slave axis from the master position is depicted below:



Calculation:

Position_Slave = SlaveScaling×CAM (MasterScaling×MasterPosition + MasterOffset) + SlaveOffset

When master is in absolute mode, the current master position is the arithmetic result of rotary axis; on the contrary, when the master is in relative mode, its position is the starting point relative to cam (usually = 0).

- Relation between StartMode and MasterAbsolute / SlaveAbsolute of CamTableSelect

Absolute mode (StartMode = 0): At the start of E-CAM synchronization, the cam calculation is independent from the current slave position. If the current slave position differs from the calculated starting position, it will result in Jump.

Relative mode (StartMode = 1): Cam changes in accordance with the current slave axis position. The slave position is derived by adding the current slave position. Also, Jump occurs when position at which the slave axis engages is different from the result of calculated starting position plus the current position.

Ramp mode (StartMode = 2, 3, 4): Add a curve for motion compensation based on VelocityDiff, Acceleration, Deceleration and Jerk to prevent Jemp during cam engagement.

MC_CamTableSelect.MasterAbsolute	Master mode
absolute	Absolute mode
relative	Relative mode

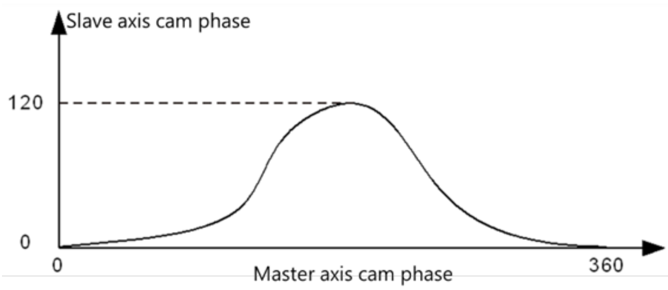
MC_CamIn.StartMode	MC_CamTableSelect.SlaveAbsolute	Slave mode
absolute	True	Absolute mode
absolute	False	Relative mode
relative	True	Relative mode
relative	False	Relative mode
ramp_in	True	Ramp in Absolute mode
ramp_in	False	Ramp in Relative mode
ramp_in_pos	True	Forward Ramp in Absolute Mode
ramp_in_pos	False	Forward Ramp in Relative mode
ramp_in_neg	True	Reverse Ramp in Absolute mode
ramp_in_neg	False	Reverse Ramp in Relative mode

■ Offset and scaling (MasterOffset/MasterScaling/SlaveOffset/Slavescaling)

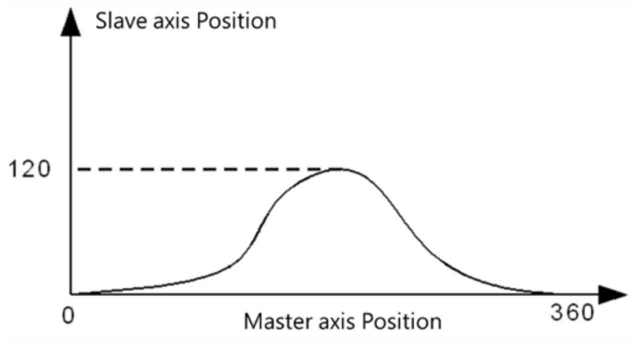
Although the cam relation between master and slave axis is pre-planned, during cam execution, you can still perform position offset or position scaling based on the relation via the parameters Offset or Scaling. For instance, only one kind of cam relation is required despite the fact that there are different sizes for the same product after being processed because you are able to change the parameters Offset and Scaling for adaption. The user can specify the offset and scaling factors for master and slave axes respectively.

7

The values of offset and scaling of the master and slave axes jointly determine the actual cam relation. The effect is described in the following example. Below is the preset cam relation:

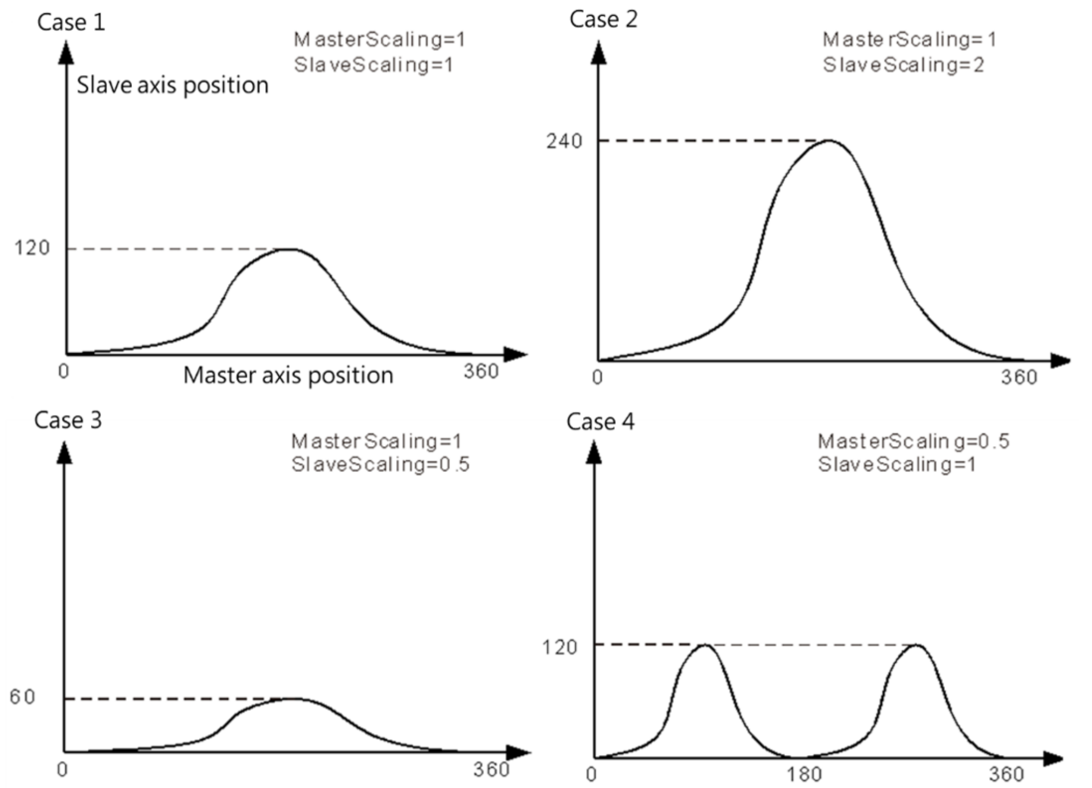


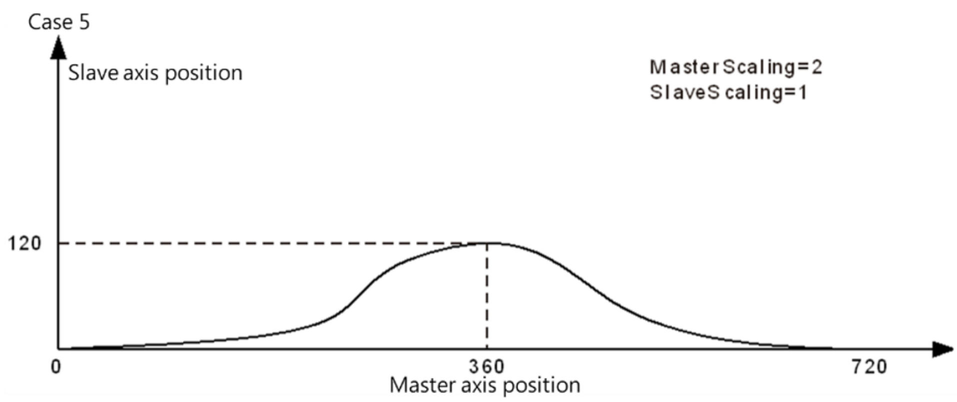
When master and slave axes are in absolute mode and engaging with each other, the axis positions are both 0. Without offset and scaling (default), the cooresponding relation of actual master and slave positions during cam execution is shown in the figure below.



When the offset or scaling is not in default value, its influence on the corresponding relation between the actual position of master and slave axes during cam execution is drawn below:

- The influence of scaling ratio on relation between master and slave positions during cam execution when offset for both axes is 0:





Case 1: When the scaling ratio for master and slave is 1 and the offset is 0, the actual relation is consistent with the pre-planned one.

Case 2: When the scaling ratio for master is 1 and for slave is 2 with the offset as 0, the slave position in relation to the master position becomes twice the amount of the pre-planned one.

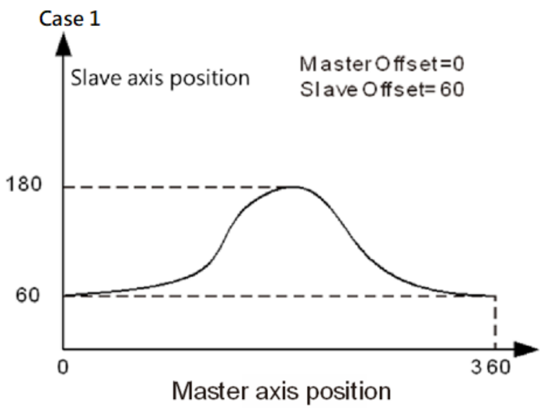
Case 3: When the master scaling ratio is 1, the slave scaling ratio is 0.5 and the offset is 0, the slave position in relation to the master position becomes half of the pre-planned value.

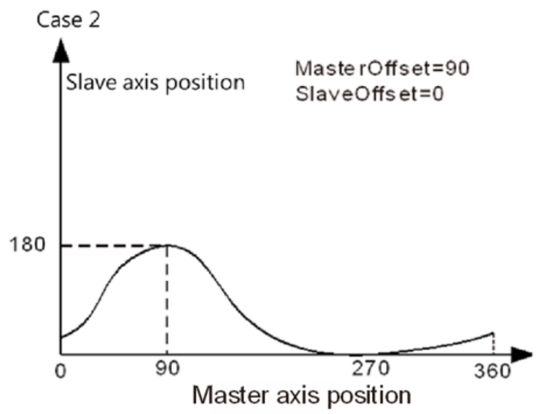
Case 4: When the master scaling ratio is 2, the slave scaling ratio is 1 and the offset is 0, the master position corresponding to the slave's becomes twice as planned. From the perspective of cam phase, the master cam phase is twice the amount of the pre-planned one, meaning that the master cam cycle changes from 360 to 180 while the slave cam phase remains unchanged.

Case 5: When the scaling ratio for master is 0.5 and for slave is 1 with the offset as 0, the master position corresponding to the slave position becomes half of the pre-planned one. From the perspective of cam phase, the master cam is half of the pre-planned one, meaning that the master cam cycle changes from 360 to 720 while the slave cam phase remains unchanged.

- The influence of offset on relation between master and slave positions during cam execution when scaling ratio for both axes is 1:

  Master offset affects the horizontal movement of cam curves whereas slave offset affects the vertical movement of cam curves.



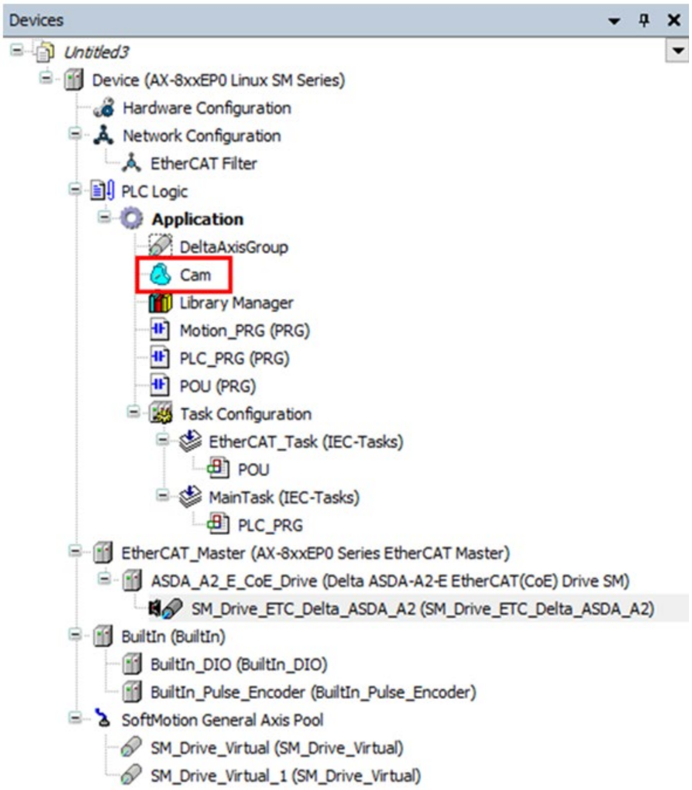


Case 1: When the scaling ratio is 1, the master offset is 0, and the slave offset is 60, the pre-planned value of the slave position corresponding to the master position is added 60. For instance, in the plan the master position is 180 and the slave position that cooresponds is 180. But in the actual situation, the corresponding slave position is 240 (240=180+60).

Case 2: When the scaling ratio is 1, the master offset is 90, and the slave offset is 0, the master position corresponding to the slave position offsets by 90 (adding offset value) based on the pre-planned value. For instance, in the plan the master position is 180 and the slave position that cooresponds is 180; in the actual situation, the master position 90 cooresponds to the slave position 180, which is the cooresponding slave position of the pre-planned master position 180 (180=90+90).

- **Cam Table**

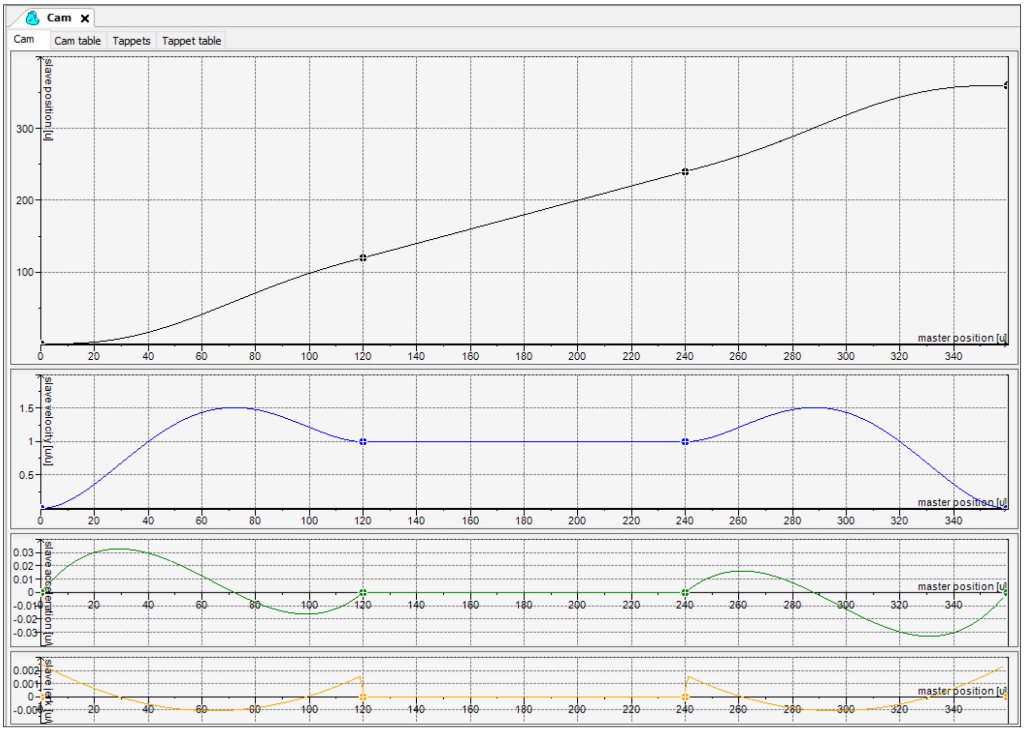
By selecting **Cam** in DIADesigner-AX project tree, you can edit cam curve which determines the operation of slave cam.



- Features of cam curve table
  - The cam curve provides information on the movement of the corresponding slave axis at any time point including moving range, velocity, acceleration and jerk. The curve type between each key point can be set, for example, as straight line or quintic curve.
  - By default, the master coordinate starts from 0 and end at 360, and can be modified according to the actual path.

■ Cam curve editing

- Graph editing in DIADesigner-AX:



Cam table is edited graphically. The abscissa indicates the master position and its length decides the range of cam operation. There are four curves above, which respectively represent position, velocity, acceleration and jerk (jump). In terms of cam design, curves that indicate position and velocity are used for motion range adjustment while curve representing acceleration are modified to achieve smoother movement.

- Cam table editing in DIADesigner-AX:

In addition to editing on graphics, the software allows revisions directly to cam table, including any increase or decrease on key points and positions.

Cam ×

Cam | Cam table | Tappets | Tappet table

	X	Y	V	A	J	Segm...	min(P...	max(P...	max(|V...	max(|A...
	0	0	0	0	0					
						Poly5	0	120	1.5120...	0.0328...
	120	120	1	0	0					
						Poly5	120	240	1	0
	240	240	1	0	0					
						Poly5	240	360	1.512	0.0328...
	360	360	0	0	0					

- Programming editing:

Users can also adopt function blocks to modify key points on cam table. But modifications through programming will not appear on software display accordingly.

Modification through the function block DMC_CamWritePoint is as explained below:

⓪ Specified cam table

① Execute function blocks

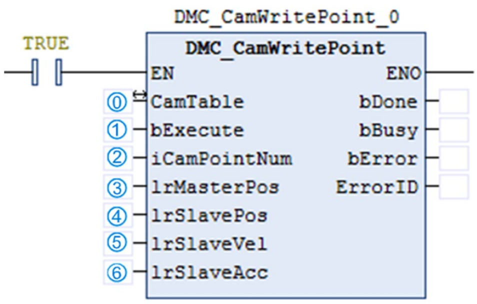
② Select the cam point number to be read

③ Position of cam master axis

④ Position of cam slave axis

⑤ Velocity of cam slave axis

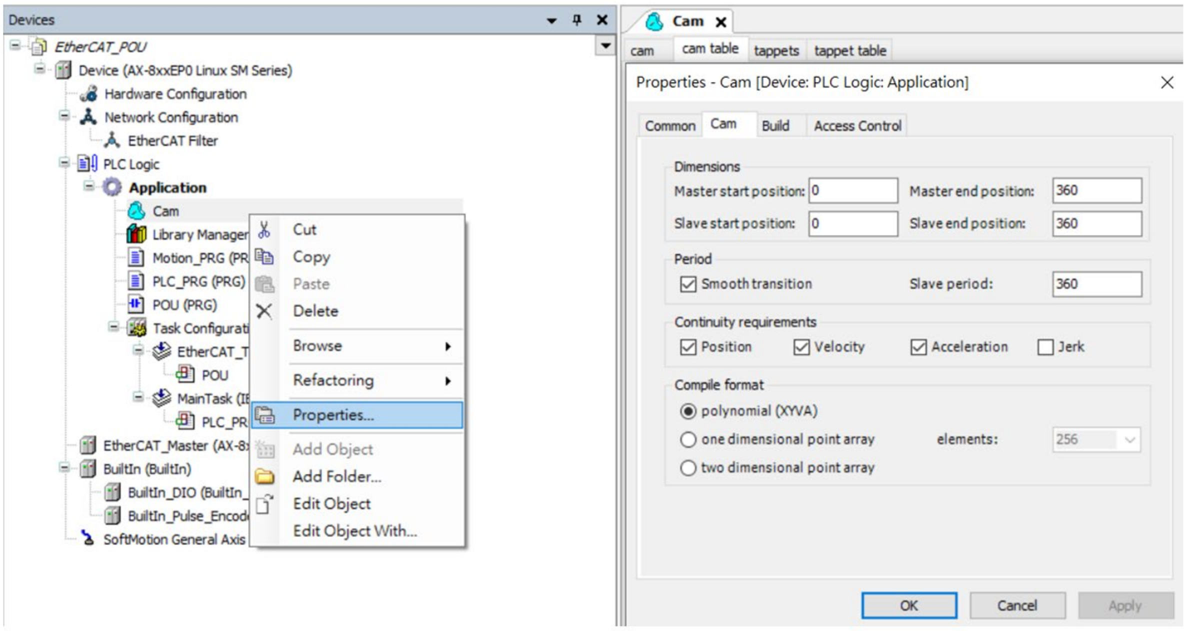
⑥ Acceleration of cam slave axis



For details of function blocks, refer to the **AX Series Motion Controller Manual**.

■ Cam table properties

You can adjust the properties of cam table such as the starting and ending positions of master and slave axes, periodic parameters setup, required curve continuity and editing format on the **Properties** window.



■ E-Cam setup

1. Cam table configuration: set up the ranges for master and slave axes, create starting point, ending point and other key points, and adjust curve type.

2. Using MC_CamTableSelect to connect the configured cam table to the actual one and obtain cam ID for further use in the following steps.

3. After obtaining cam ID, use MC_CamIn to engage the specified master axis with the slave axis.

4. Use MC_CamOut to terminate the master-slave relation. In addition, for synchronous movement, adopting MC_Stop and MC_Halt on the slave axis can stop synchronization between the axes.

■ Cam table switching

The reference cam table can be changed at runtime by using MC_CAM_REF to replace the reference cam table of MC_CamTableSelect.

- Variable Declaration

```
P : MC_CAM_REF;         //CamTable reference
CamTableID : INT;       //CamTable Switch
```

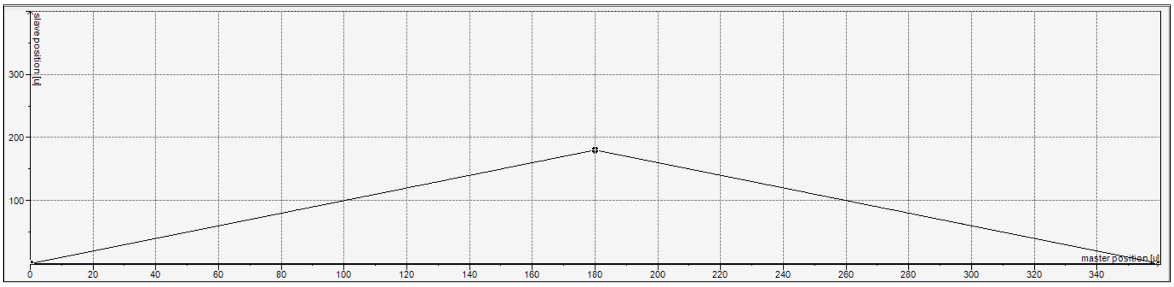
- Switch cam table

```
CASE CamTableID OF
    0: P:=Cam;
    1: P:=Cam_1;
END_CASE
```

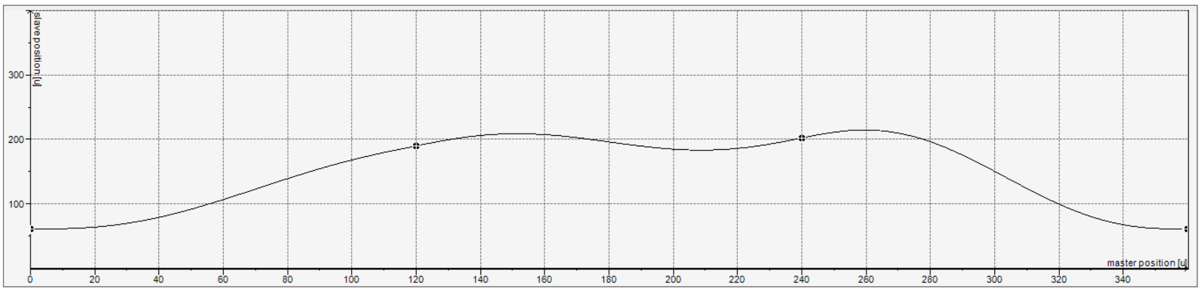
The example switches CamTableID to change MC_CAM_REF so as to switch between cam tables.

The two cam tables are as follows:

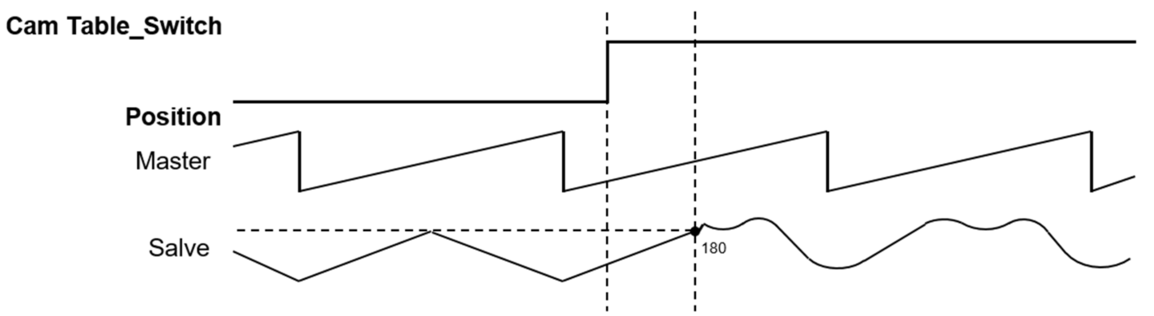
The first cam table (Cam)



The second cam table (Cam_1)



Timing diagram for cam table switching



Switching the cam tables, the slave axis still follows the trajectory planned in the first table and turns to follow the trajectory of the second table until the master position reaches the next key point of the first table.

7.7.3 Velocity Control

Velocity control is classified into three servo control modes: Cyclic Synchronous Position (CSP), Cyclic Synchronous Velocity (CSV) and Profiel Velocity Mode.

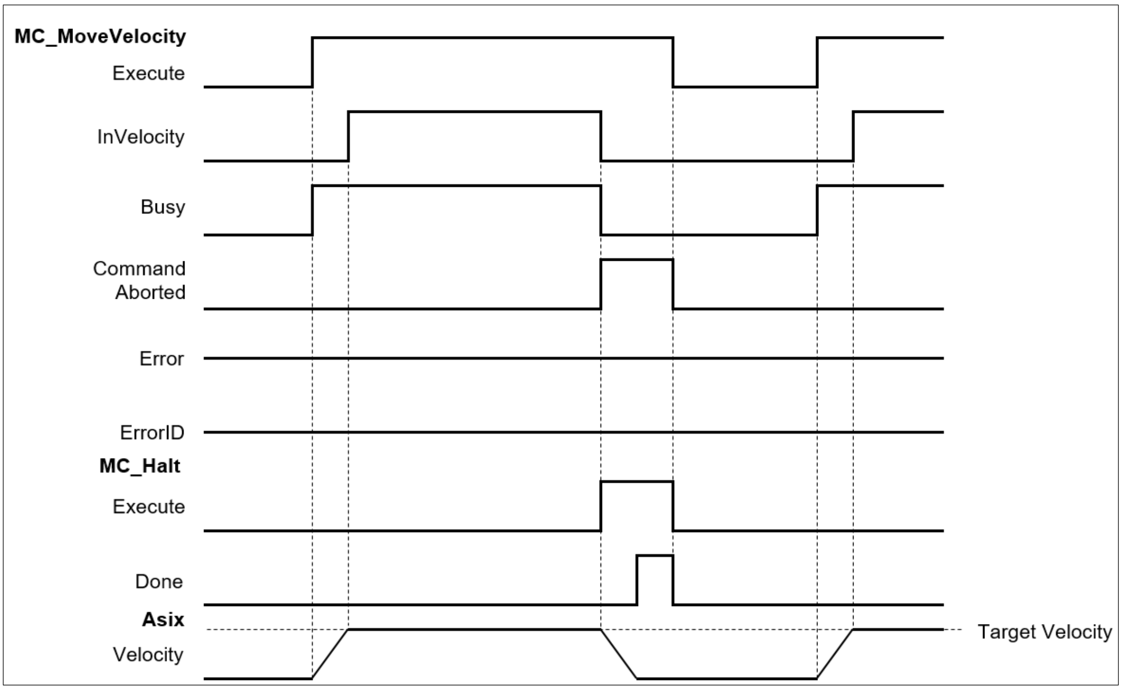
7.7.3.1 CSP Mode

As introduced in **Section 7.7.2.1**, the CSP mode stands for cyclic synchronous position mode. In this mode, the controller calculates the command position that it should reach in each cycle according to the assigned velocity (as well as acceleration, deceleration, and jerk), and then send the command position in every cycle to the servo for positioning.

If the current servo position falls behind the command position due to external interference, vibration occurs in order to compensate for the errors.

The application of function block MC_MoveVelocity means executing velocity control in CSP mode, in which the axis will enter the continuous_motion state. You can specify its acceleration, deceleration, and jerk before reaching the specified velocity or when buffering. MC_Stop, MC_Halt, or other instructions are used to stop the control mode when needed.

In the following diagram, MC_MoveVelocity is used for velocity control and MC_Halt is applied to stop it:

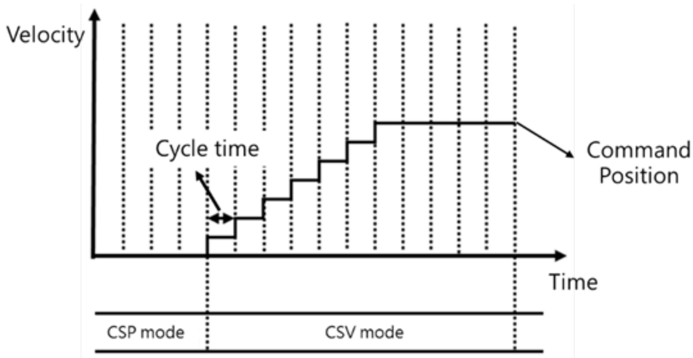


The velocity can be specified as 0. Although being stationary, the system is still in the state of continuous_motion.

In the AX series, MC_MoveVelocity can be used for single-axis velocity control in CSP mode. For details, please refer to the **AX Series Motion Controller Manual**.

7.7.3.2 CSV Mode

CSV mode stands for cyclic synchronous velocity mode. The velocity control is achieved by the controller calculating the velocity for each cycle according to the assigned velocity (as well as acceleration, deceleration, and jerk), and then send the command in each cycle to the servo for execution.

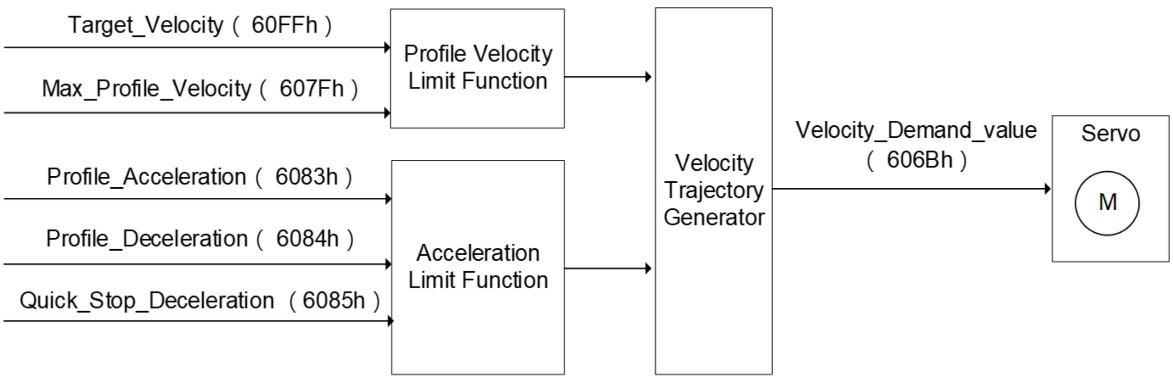


In this mode, there is no changes in velocity for compensation resulting from external interference like that in CSP mode since it is the command velocity given to the servo in each cycle.

In AX series, MC_VelocityControl can be used for single-axis velocity control in CSV mode. For details of function blocks, refer to **AX Series Motion Controller Manual.**

7.7.3.3 Profile Velocity Mode

In profile velocity mode[*1], the motion trajectory is planned by the velocity trajectory generator based on the conditions such as velocity command, acceleration and deceleration specified by the host device.



*1: This mode is used for positioning axes.

7.7.4 Torque Control

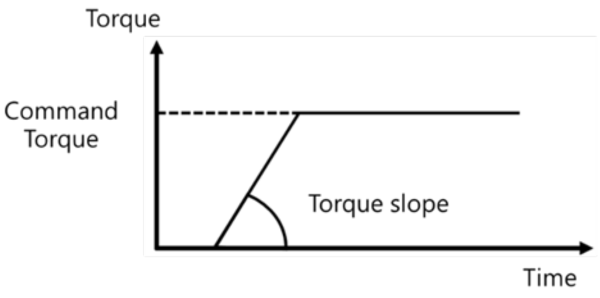
Torque control is classified into two servo control modes: Cyclic Synchronous Torque (CST) and Profile Torque (PT).

- **Profile Torque[*1] (PT)**

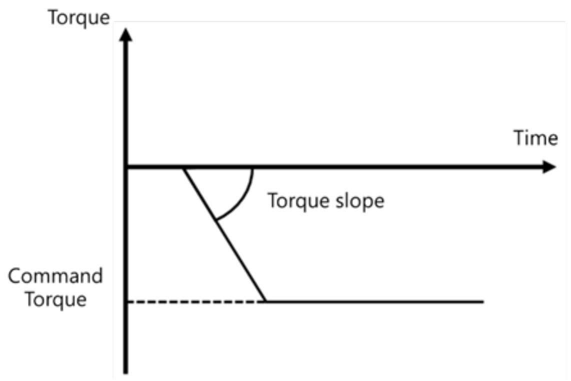
Use DMC_TorqueControl to assign a single axis to continuously generate the specified torque output.

Precautions:

- When using DMC_TorqueControl, the control mode is switched to cyclic synchronous torque mode.
- When using DMC_TorqueControl, the control mode is switched to torque mode and function blocks regarding displacement or velocity are unavailable. Instead of using MC_Stop to stop the motor, set DMC_TorqueControl.Enable to False and you can stop it immediately.
- Torque cannot be set to 0. If so, DMC_TorqueControl will be reported as error.
- The velocity of DMC_TorqueControl can be utilized to set the maximum velocity for servo motors with the intention to avoid high-speed rotation as motor load declines in PT mode.
- Adopt TorqueRamp to set the torque change rate for achieving the target torque value.
- When Torque > 0, the motor runs in the forward direction.



- When Torque < 0, the motor runs in the reverse direction.



*1:

ASDA-A3-E Series V1.1165 or later supports Profile Torque Mode.

ASDA-B3-E Series V1.0665 or later supports Profile Torque Mode.

7.7.5 Common Functions of Single-axis Control

The common functions of single-axis control are shown below.

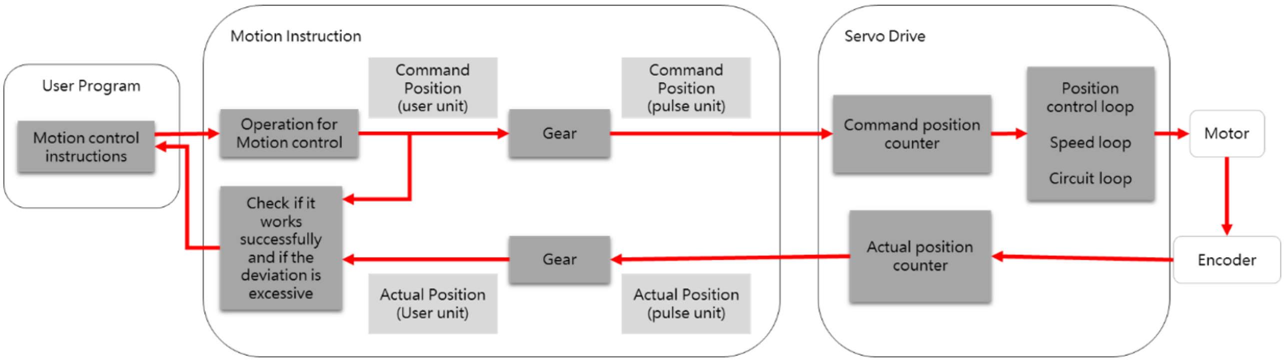
7.7.5.1 Position Command

Positions in MC function modules can be subdivided into the following two types.

Type	Definition
Command position	The function block output used to control the position of the axis
Actual position	Feedback position from the servo drive*1

*1: If it is a virtual axis, since there is no position feedback from the servo drive, the command position directly substitutes for the actual position

The connection is drawn in the figure below:



Comparisons between the two types:

Item	Command Position	Actual Position
Counting mode	Linear axis / rotary axis	The same setting as in command position.
Command unit	Length unit (m, mm, inch...) / angle unit (degree) /…	The same setting as in command position.
Software limit	Set the limit of the position range for motion instructions	The same setting as in command position.
Positioning	Subject to be positioned to anywhere within a limited range	The same setting as in command position but possibly with position lag*1

*1: Affected by the mechanism and servo settings, there may be subtle differences between command position and actual position, the so-called position lag. Position lag may increase slightly as the motion velocity increases. To control this gap, you can monitor position lag by adjusting axis settings and take actions in response to increasing position lag. There is no position lag when the actual position of a virtual axis equals its command position.

The relevant parameters are as follows:

- **Position unit**

The position unit refers to the command unit.

7

- **Setting of position lag**

Setting	Value	Definition
Position lag supervision	deactivated	Position lag is not checked.
	disable drive	The axis is in Servo Off when position lag exceeds the set limit.
	do quick stop	The axis is in quick stop when position lag exceeds the set limit.
	stay enabled	The axis remains Servo On when position lag exceeds the set limit.
Lag limit [u]	LREAL	The limit of permissible error

Except for the value deactivated, the error SMC_ERROR.SMC_DI_POSITIONLAGERROR will be reported when the limit of error is exceeded.

- **Software limits**

Setting	Value	Definition
Software limits Activated	Checked / Unchecked	Whether to activate soft limits
Negative [u]	LREAL	Soft limits in negative direction
Positive [u]	LREAL	Soft limits in positive direction

- **Description of position in MC Function Modules**

There are two different descriptions for the input variables regarding position in MC function modules.

Item	Definition
Position	Target position (absolute position)
Distance	Moving distance (relative position)

- **Position monitoring**

Observation of position changes can be done by monitoring the following two axis variables (AXIS_REF_SM3 type).

Variable Name	Position Type	Data Type
.FSetposition	Command position	LREAL
.fActPosition	Actual position	LREAL

7.7.5.2 Velocity Command

There are two types of velocity in MC function modules.

Type	Definition
Command velocity	The outputs of MC function modules is used to control the axis velocity
Actual velocity	Velocity derived from the actual position provided by the servo drive at each time point*1

*1: For virtual axis, since there is no servo drive, the command position will replace the actual position.

The relevant parameters are explained as follows:

- **Velocity unit**

The velocity unit: command unit/s.

- **Velocity ramp type**

Setting	Value	Definition
Velocity ramp type	Trapezoid	The velocity curve is trapezoidal (the acceleration of each line segment is constant)
	Sin^2	Velocity curve as sin2 function (fixed acceleration curve)
	Quadratic	The acceleration curve is trapezoidal (acceleration is limited by jerk) speed curve
	Quadratic (smooth)	Same as Quadratic but the jerk (jump) curve is a continuous speed curve

- **Description of velocity in MC function modules**

The input variables related to speed in the motion function block are as follows:

Item	Definition
Velocity	Target velocity*1

*1: The target velocity might not be achieved due to inadequate trajectory length, poor acceleration and jerk, etc.

- **Velocity monitoring**

To observe the speed change, you can monitor the two axis variables (AXIS_REF_SM3 type):

Variable Name	Type	Data Type
.fSetVelocity	Command velocity	LREAL
.fActVelocity	Actual velocity	LREAL

7.7.5.3 Acceleration and Deceleration Command

Acceleration in MC function modules can be subdivided into the following two types.

Position Type	Definition
Command acceleration	Acceleration for axis control from the outputs of MC function modules
Actual acceleration	Acceleration derived rom actual the velocity

The relevant parameters are explained as follows:

- **Acceleration unit**

The acceleration unit: command units/s^2.

- **Acceleration ramp type**

Refer to **Section 7.7.5.2** of this manual.

- **Description of acceleration in MC function modules**

The input variables related to acceleration / deceleration in the MC function modules are as follows:

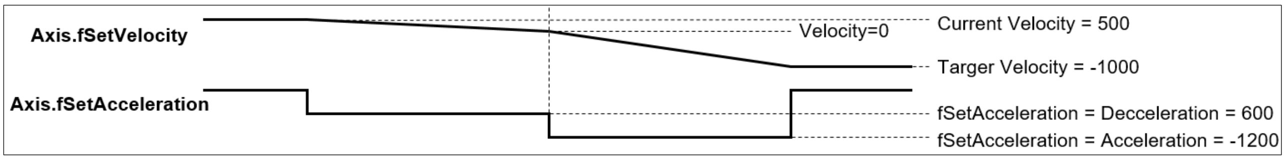
Item	Definition
Acceleration	Target acceleration*1
Deceleration	Target deceleration*1

*1:

The target acceleration and deceleration might not be achieved due to inadequate trajectory length and jerk, etc.

The application standard of acceleration and deceleration is that when the absolute value of the current velocity needs to be reduced, Deceleration will be applied; on the contrary, when the absolute value of the current velocity needs to be increased, Acceleration will be applied.

For example, the following diagram depicts the velocity and acceleration curves under the situation when the current velocity is 500 and the motion command is in reverse direction (Velocity = 1000, Acceleration = 1200, Deceleration = 600):



- **Acceleration monitoring**

If observation of changes in acceleration is needed, you can monitor the following two axis variables (AXIS_REF_SM3 type):

Variable	Type	Data Type
.fSetAcceleration	Command acceleration	LREAL
.fActAcceleration	Actual acceleration	LREAL

7.7.5.4 Jerk Command

Jerk assigns the rate of changes in acceleration / deceleration. When the jerk is specified, the velocity ramp is in S-curve (the ramp is shown as increasing or decreasing without jerk), which reduces mechanical shock. There are two types of jerk:

Type	Definition
Command jerk	The outputs of MC function modules used for jerk control
Actual jerk	Jerk derived from the actual acceleration

The relevant parameters are described as follows:

- **Jerk unit**

The jerk unit: command unit/s^3.

- **Jerk ramp type**

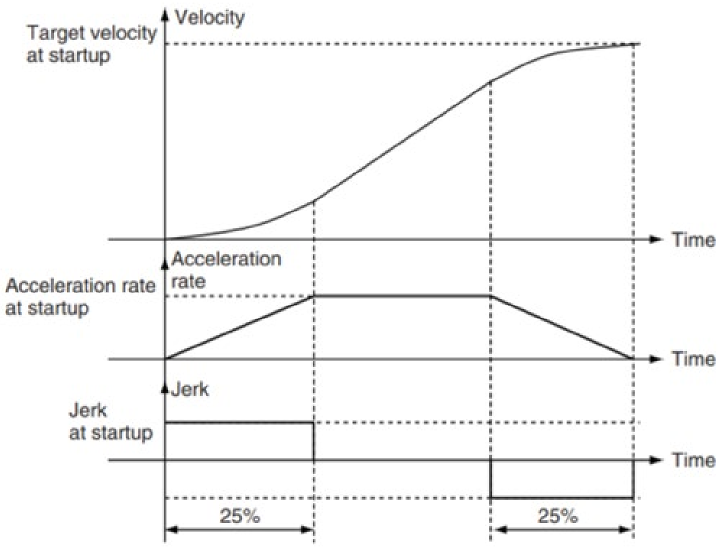
Refer to **section 7.7.5.2** of this manual.

- **Description of jerk in MC function modules**

The input variable related to jerk in MC function modules:

Project	Function
Jerk	Target jerk*1

*1: When the velocity ramp type is Trapezoid or Sin2, the setting values of jerk are not applied in the movement; contrarily, when the velocity ramp type is Quadratic or Quadratic smooth, the setting values of jerk will affect the velocity ramp.



- **Jerk monitoring**

To observe changes in jerk, you can monitor the following two axis variables (AXIS_REF_SM3 type):

Variable Name	location type	data type
.fSetJerk	Command jerk	LREAL
.fActJerk	Actual jerk	LREAL

7.7.5.5 Axis Direction

The following cases require specified direction for operation:

- When the axis is moving at a constant velocity and the input value of velocity is an absolute value
- When the axis is set as a rotary one and is able to reach the target position in both forward and reverse directions

The input variables related to direction in MC function modules are as follows:

Item	Setting	Definition
Direction	negative	Move in negative direction
	shortest	Move along the shortest path (only valid for rotary axis)
	positive	Move in positive direction
	current	Move along the current motion direction (only valid for rotary axis)
	fastest	Move along the path which takes the least time (only valid for rotary axis)*1

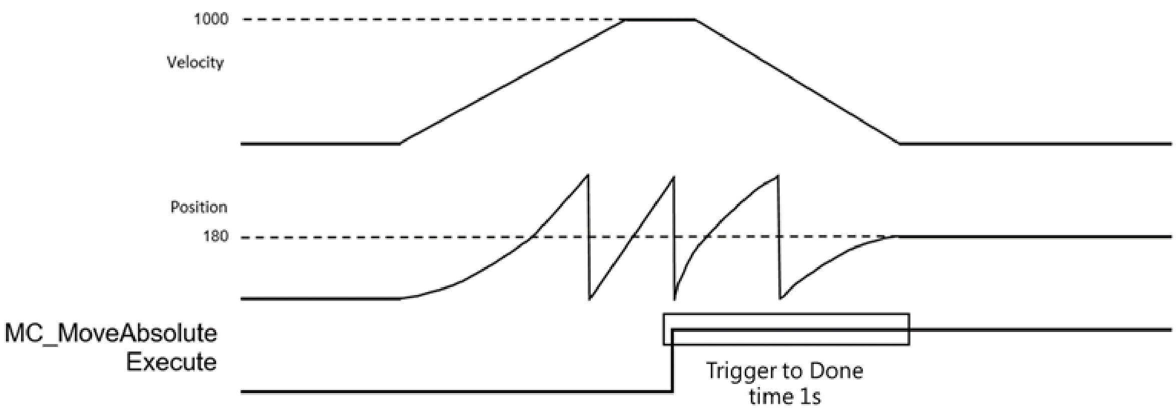
*1: shortest (moving distance) and fastest (moving time) are similar in concept, but not completely the same. Please refer to the following example:

- Settings:

  Set the axis as rotary axis with a range of 360.

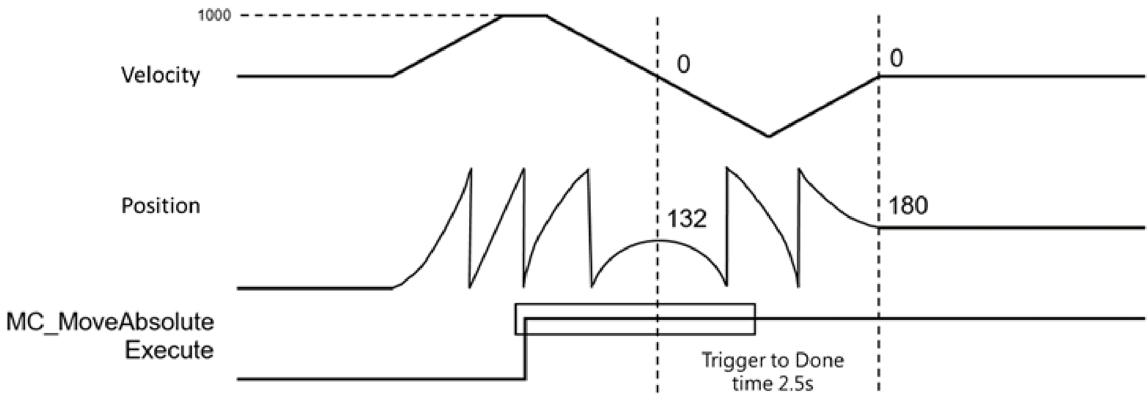
  Set the velocity ramp type of the axis to Trapezoid.

- Steps:
  - Use MC_MoveVelocity to move at constant velocity (velocity = 1000).
  - When the motor reaches 350 with the velocity coming to 1000, MC_MoveAbsolute is executing twice respectively in different directions under the same circumstance.
    - Execute MC_MoveAbsolute (Position=180, Velocity=Acceleration=Deceleration=1000, Direction=fastest).



When MC_MoveAbsolute.Execute is triggered, the system determines that the least time-consuming direction to reach position 180 is to decelerate in positive direction to 0. The process takes about 1 second.

- Execute MC_MoveAbsolute (Position = 180, Velocity = Acceleration = Deceleration = 1000, Direction = shortest).



When MC_MoveAbsolute.Execute is triggered, the system determines that the shortest path to reach position 180 is to go in opposite direction (350 - 180 = 170). However, it turns out to take more time because of reversal of the velocity during the process. The process takes about 2.5 seconds.

7

7.7.6 Axis Group Control

An axis group must consist of more than one axis to achieve axis group movement, which is configured via DIADesigner-AX. Up to six axes are supported for linear axes while three axes are supported by rotary axes with extra three axes as following axes.

7.7.6.1 Linear Interpolation

TransitionMode: Changing the interpolation trajectory while in motion may result in machine vibrations and noises. To alleviate the situation, it is suggested to apply the input variable TransitionMode.

- **Available TransitionMode**

Mode	Describe
None	No effect (default)
Overlap	Continue by combining the deceleration of the previous motion with the acceleration of the current motion

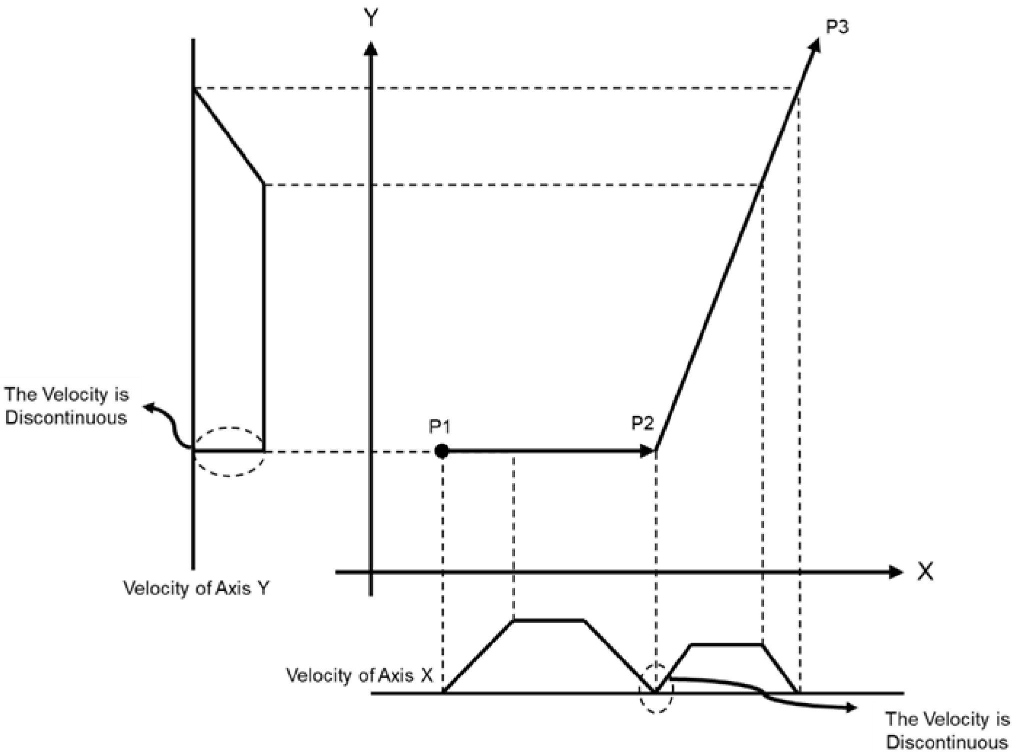
- **Buffer modes supported by TransitionMode**

Mode	Aborting	Buffered	Blending Low	Blending Previous	Blending Next	Blending High
None	A	A	N	N	N	N
Overlap	A	A	D	D	D	D

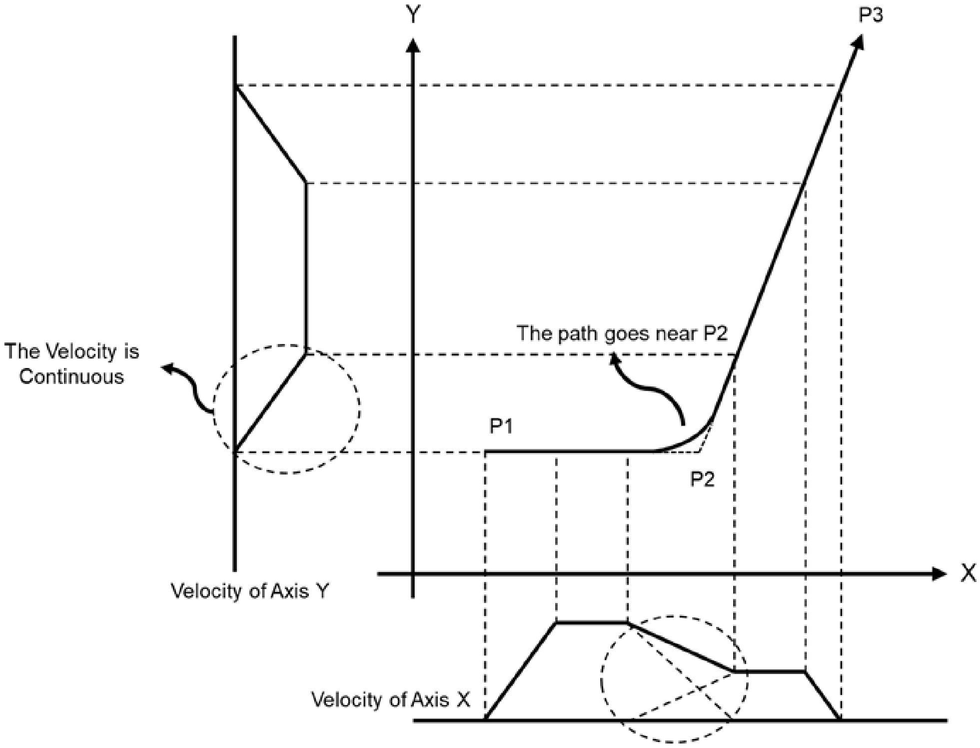
A = Supported
N = Not supported
D = Continue with Blending mode

- **TransitionMode**

In the situation below, set the mode to None or Overlap and then select Buffered.

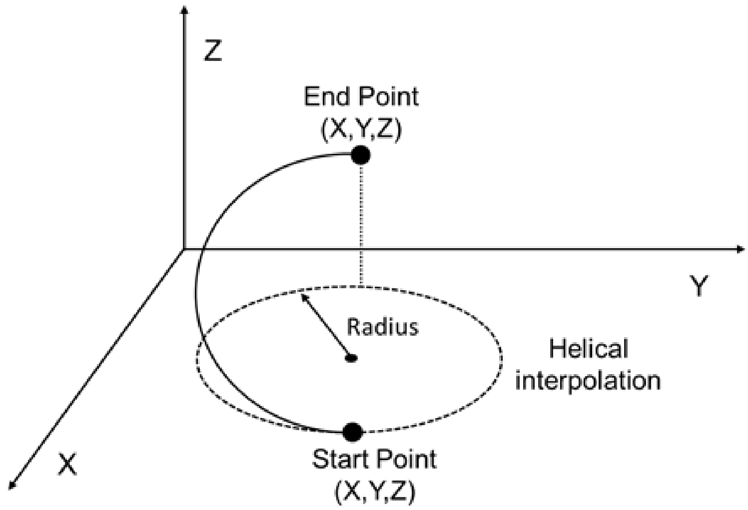


In the situation below, set the mode to Overlap and then select Blending. It is suggested refering to the deceleration and acceleration given to the motion function blocks of each axis group for planning.



7.7.6.2 Helical Interpolation

In helical interpolation, it supports three axes and performs X, Y, Z axes movement in a 3D system with three additional following axes.

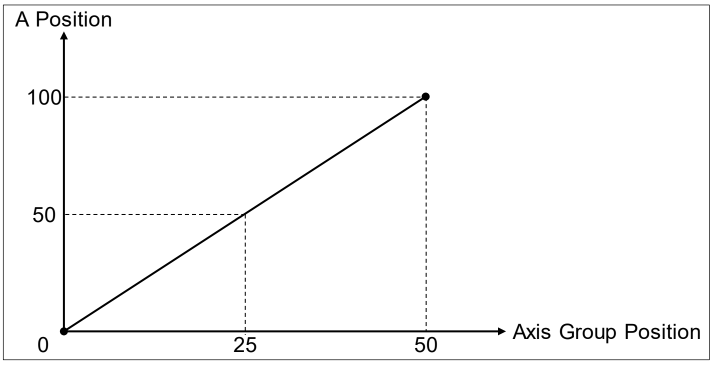


- **Concept of following axes in axis group**

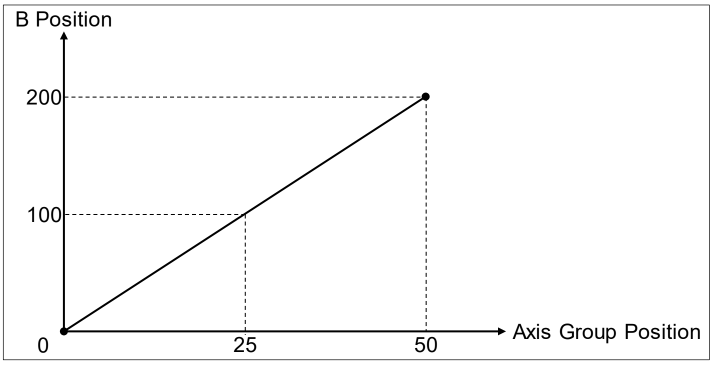
The following axes (A, B, C) move synchronously in a proportional manner as the axes (X, Y, Z) are moving.

The movement of the axis group starts from 0, in which the axes (X, Y, Z) move to (30, 40, 0) with a distance of 50 and the following axes move to (100, 200, 300). The following figures show the synchronous movements of each following axis:

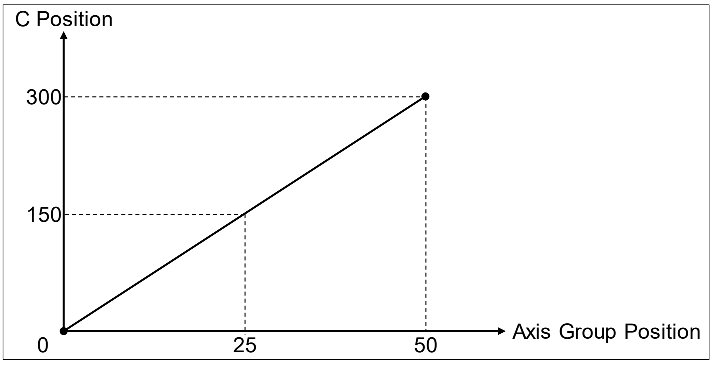
■ The following axis A



■ The following axis B



■ The following axis C



When the axes (X, Y, Z) stay still, the input velocity given to the axis group is instead applied to the following axis which keeps the longest distanc; and, the other following axes will follow synchronously in proportion to the distance.

7.7.6.3 Stop Command for Axis Group

There are two ways to stop an axis group:

- **Programming stop**

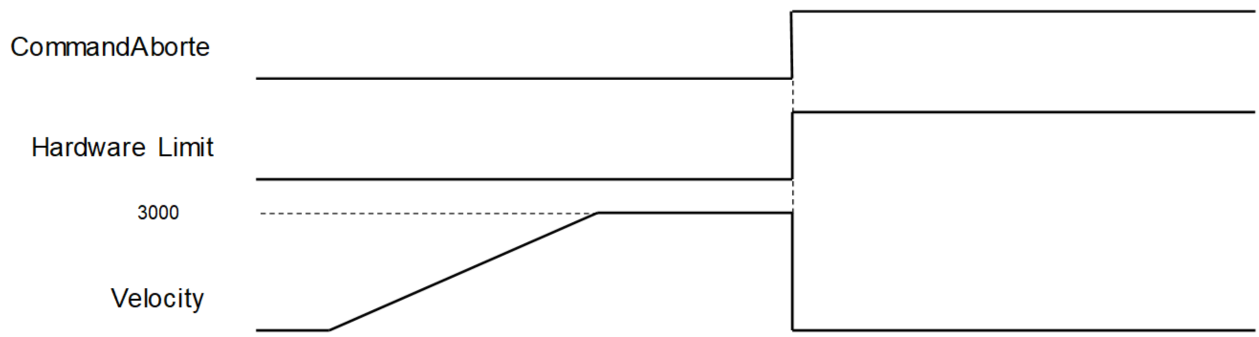
Adopt DMC_GroupStop in programming to decelerate and stop the axis group at its current position. At this time, the axis group state switches to GroupStopping in which no motion instruction can be executed.

The velocity for deceleration stop should be set to the lrDeceleration pin.

- **Error stop**

The axis group stops running once it encounters errors during movement.

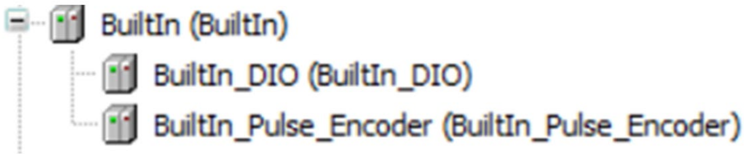
For example, if Hardware Limit is reached while the axis group is in motion, the velocity drops to zero as a result of the output of CommandAborted.



7.7.7 High-speed I/O

This chapter provides information regarding I/Os including configurations as well as parameter settings.

7.7.7.1 I/O Configuration



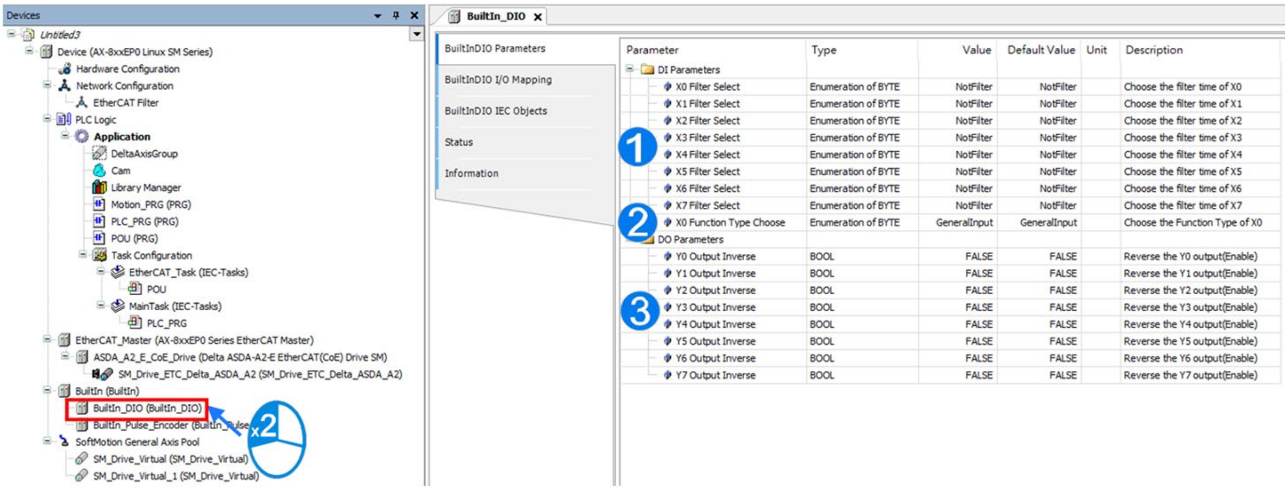
The following configurations are available for high-speed I/Os on the CPU module:

- DIO: Filtering, polarity and other functions are set here. Details are in **Section 7.7.7.2**.
- SSI Encoder: SSI encoder, clock frequency and SSI data size are set here. Details are in **Section 7.7.7.3.**
- Pulse Encoder: Variable declaration of high-speed counter/ timer and pulse-type encoder and encoder Z inverse are set here. Details are in **Section 7.7.7.4**.
- Capture / Compare: Variable declarations of high-speed capture or high-speed comparator are set here. Details are in **Section 7.7.7.5**.

7.7.7.2 DIO Settings

This section introduces functions including filtering and polarity of I/Os on device with DIO.

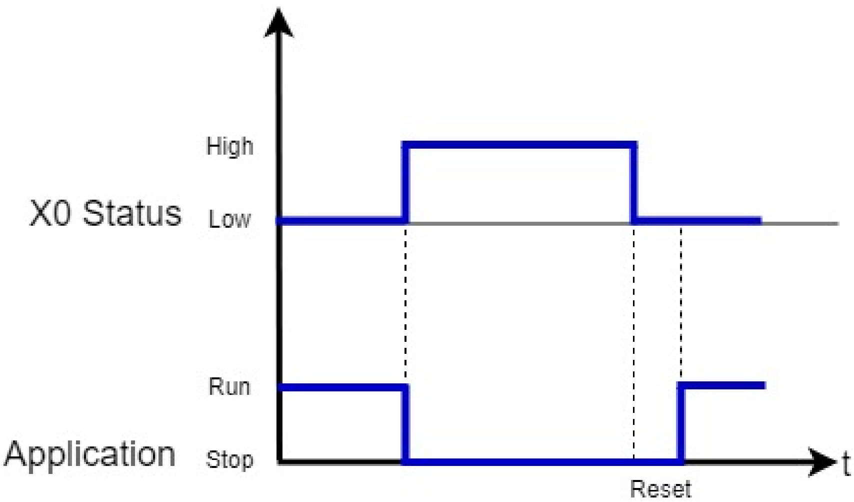
Double-click **DIO** to enter the setting page.



- **Configuration**

Function	Description
① Filter	Set the filter time: default is NotFilter. Drop-down list: Not Filter, 0.1ms, 0.5ms, 1ms
② X0 Function Type	Set X0 Function Type, default is GeneralInput. Drop-down list: GeneralInput、Stop/Start PLC Control
③ Polarity	Set the output Inverse, default is not reverse. True: Reverse False: Not Reverse

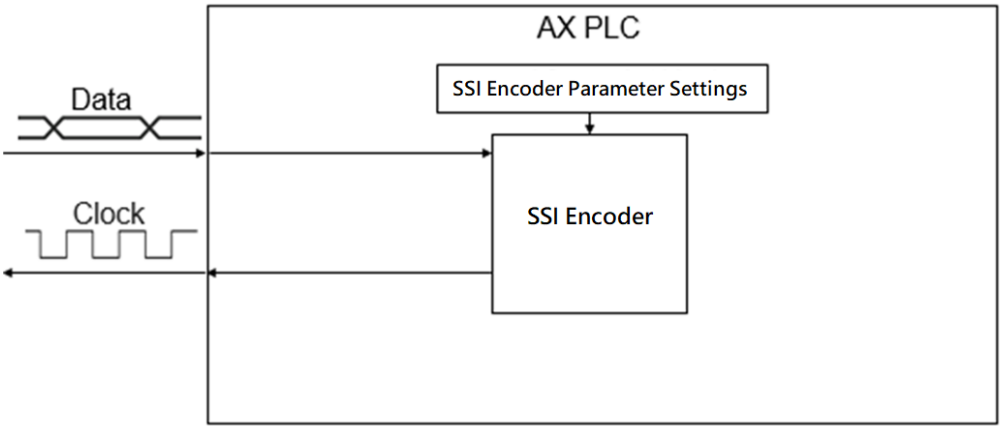
■ Stop / Start PLC Control



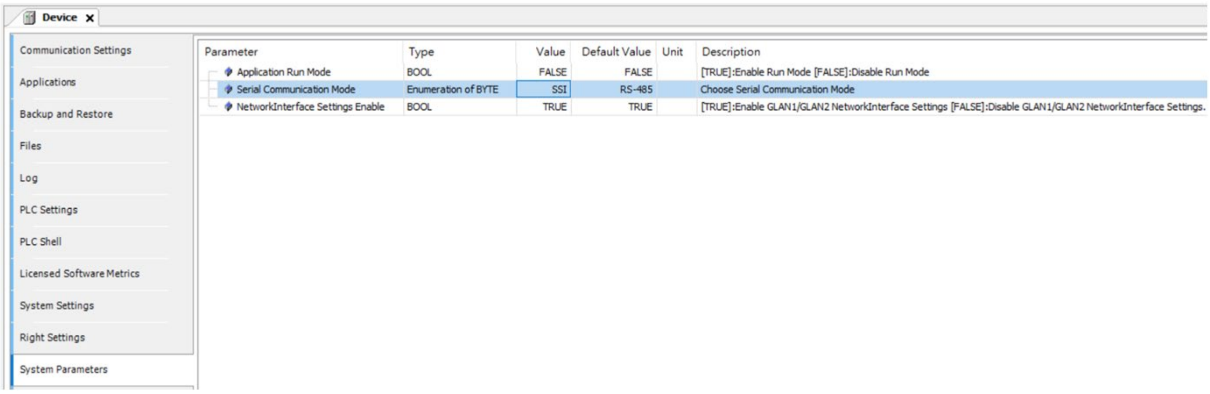
7.7.7.3 SSI Encoder Settings

AX Series PLC supports one set of SSI Encoder through Rx and Tx connection with the controller, providing 5V power input to support the encoder. User needs to set the Serial Communication Mode as SSI and activate DFB_SSI_Encoder. After setting up the preferred parameters, dataset can be received through the hardware channel.

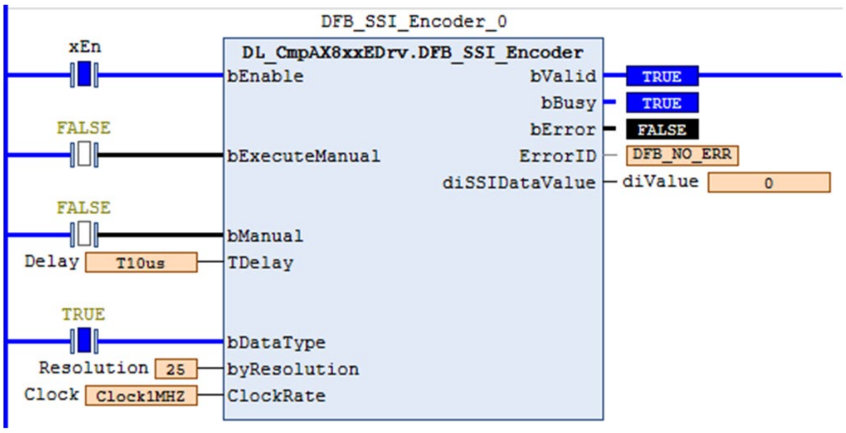
- **SSI Encoder Structure**



- **Serial Communication Mode**
  - Go to **Device** > **System Parameter** and select **SSI** in the **Serial Communication Mode** drop-down list.
  - Parameter settings in the offline mode
  - Effective after downloading



- **DFB_SSI_Encoder**



7.7.7.4 Pulse Encoder Settings

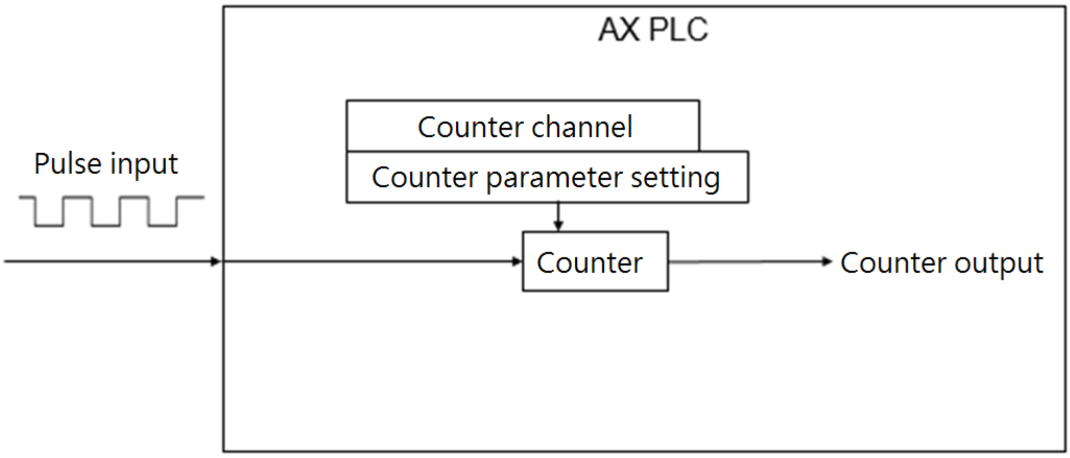
AX-8 Series PLC CPU supports:

1. 1 differential input: A, B and Z encoder connecting to the PLC can be utilized to count / time the pulse value or the frequency of the encoder.
2. 8 sets of open collector inputs: connecting through the blocks installed on the I/O board; up to 8 sets of high-speed counters / timers can be utilized to count or time the pulse value or the frequency of the encoder.

You need to enable pulse-type encoder function and set up the required parameters before you can receive data via the configured hardware channel.

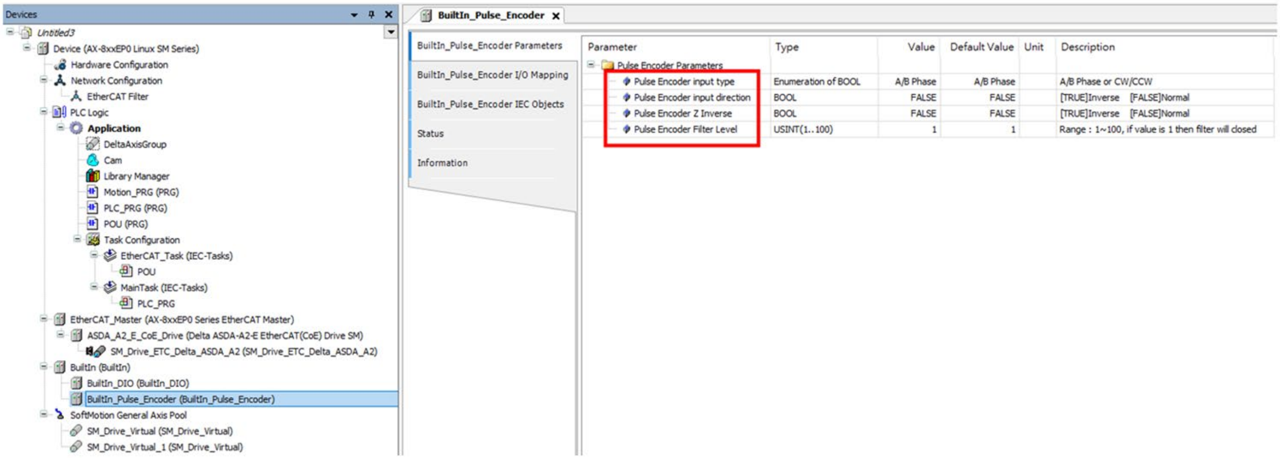
The following section introduces the function modules of the pulse-type encoder. The maximum group of high-speed counters and timers supported by AX-8 is up to 9.

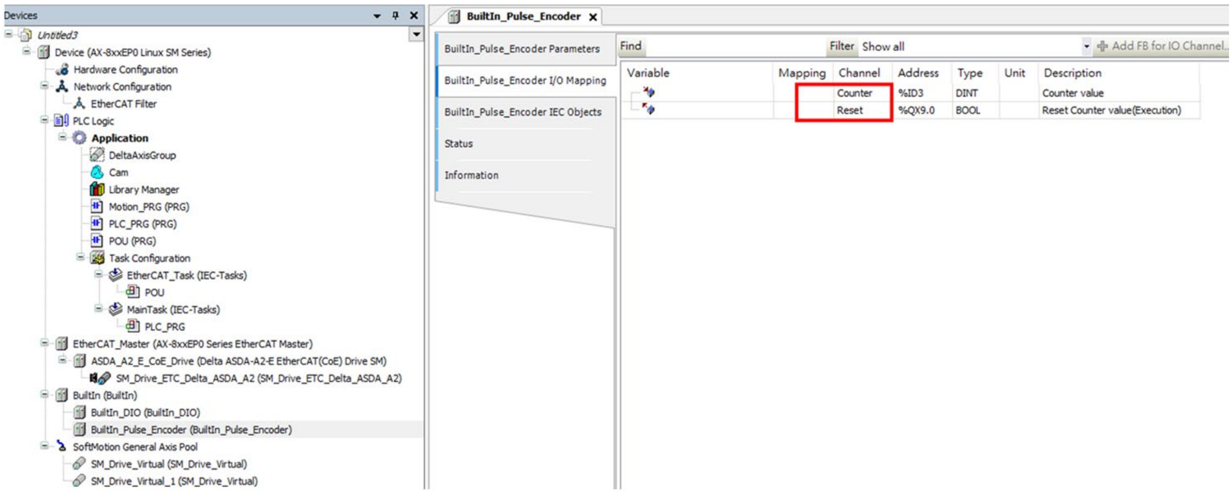
- **High-speed counter (Cnt)**



- **Differential input high-speed counting**

  ■ Double-click **BuiltIn_Pulse_Encoder** to enter I/O Mapping setting page.

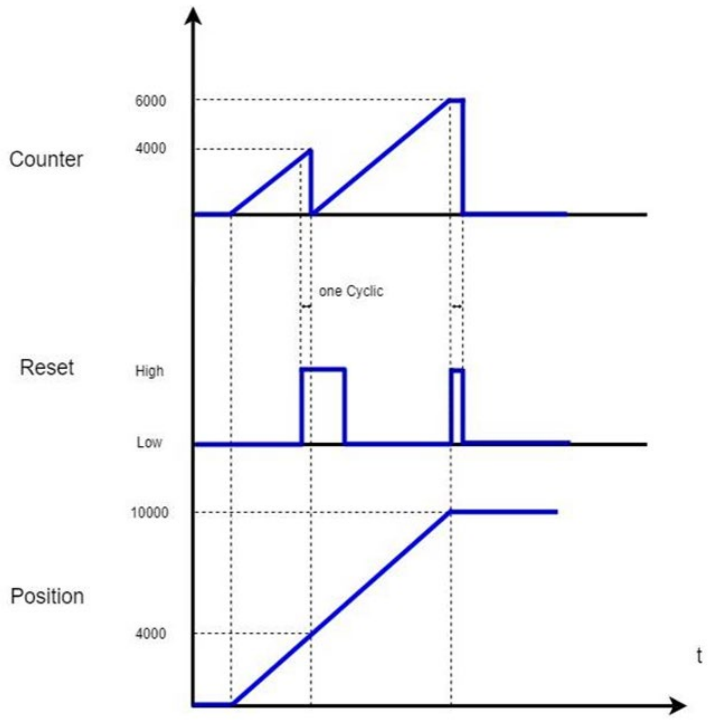




■ Configuration.

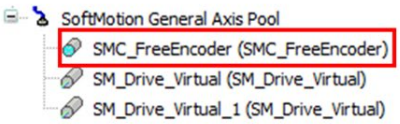
Function	Description
Encoder Type	Settings of the differential input encoder type, the default is A/B Phase. Drop-down list: A/B Phase, CW/CCW
Encoder direction	Settings of the differential input encoder direction. True: Reverse False: Not Reverse
Encoder Z Inverse	Settings of the encoder Z inverse polarity; the default is not reverse. True: Reverse False: Not Reverse
Encoder Filter Level	Settings of differential input filter level, setting range: 1~100. 1 stands for no filter and 100 represents moving average of the pulse input (including the 100th) within 100 periods.
Counter	Settings of PLC program variables and Channel Counter for Mapping. The differential input pulse count can be read through variables in the program.
Reset	Settings of PLC program variables and Channel Reset for Mapping. The differential input pulse count can be cleared through the upper source trigger variables in the program. True → False: Valid

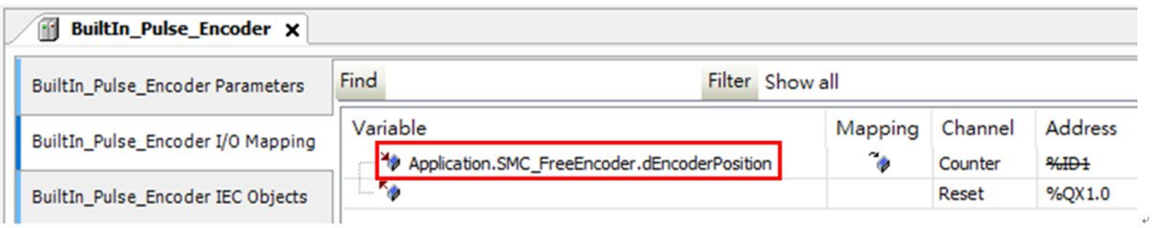
■ Reset Diagram

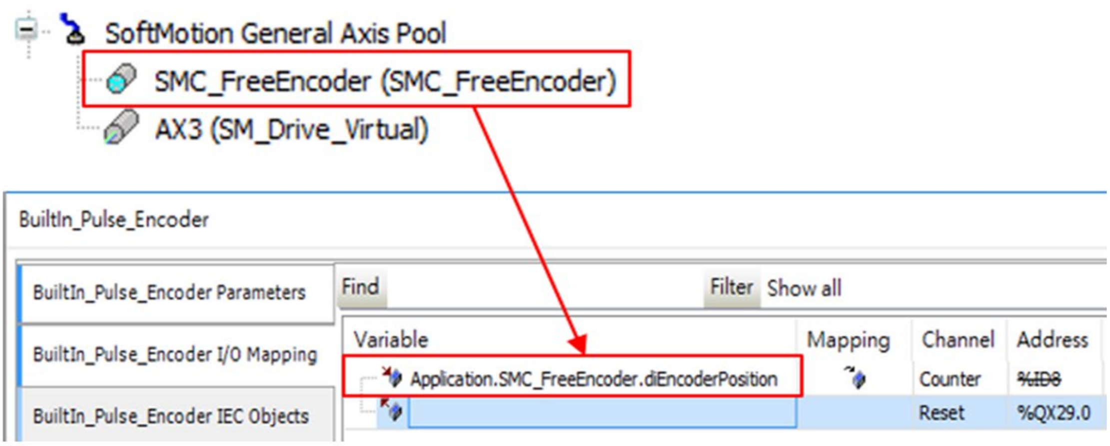


- **Differential input high-speed counting used in the program**

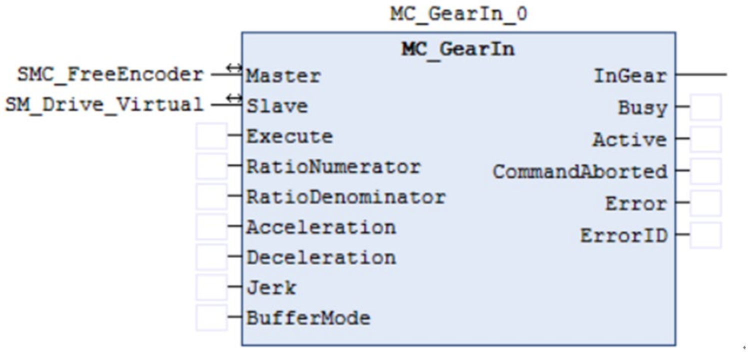
■ Add SMC_FreeEncoder in the project and use SMC_FreeEncoder.dEncoderPosition and Channel Counter for Mapping.







■ Use the **MC_GearIn** function block in POU. The source of the master axis is SMC_FreeEncoder.



- **Open collector input high-speed counting**

Setting and activation through DFB_Counter. Maximum of 8 sets of high-speed counters / timers can be utilized to count or time the pulse value or the frequency of the encoder.

■ DFB_Counter

■ Counter_Mode

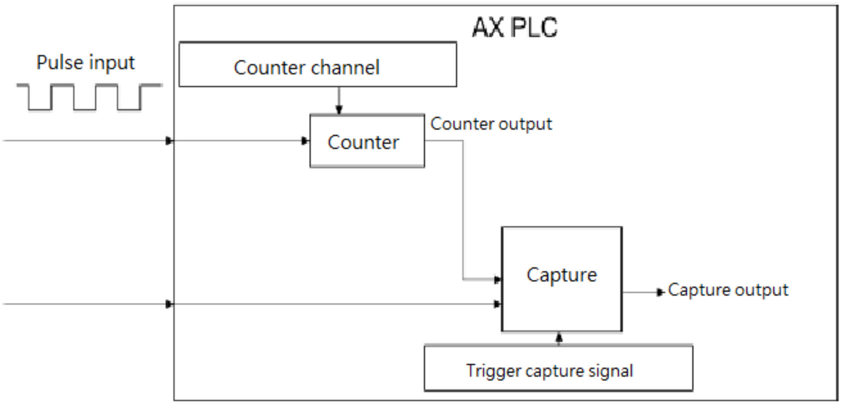
For example: Up to 8 sets of high-speed counting in Mode3. When Counter Number 0 is enabled, user can count up or down through the X0 input of I/O panel.

Number	0	1	2	3	4	5	6	7	
Mode 3	U/D1	U/D2	U/D3	U/D4	U/D5	U/D6	U/D7	U/D8	8 sets
Mode 4	U/D1	R1	U/D3	R3	U/D5	R5	U/D7	R7	4 sets
Mode 5	U/D1	DI 1	R1	S1	U/D5	DI 5	R5	S5	2 sets
Mode 6	U1	D1	R1	S1	U5	D5	R5	S5	2 sets

7.7.7.5 Capture / Compare Settings

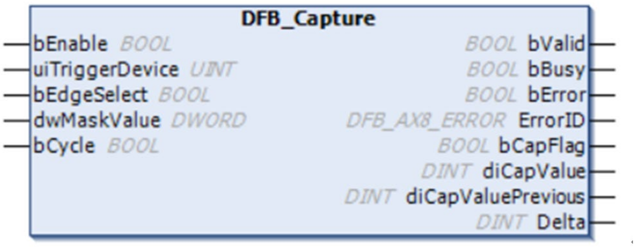
This section is about the function modules of capture and compaer. A maximum of 4 groups of high-speed captures and one set of high-speed compare are supported by AX series motion controllers. The high-speed compare allows a maximum of 256 compare units.

- **Capture**

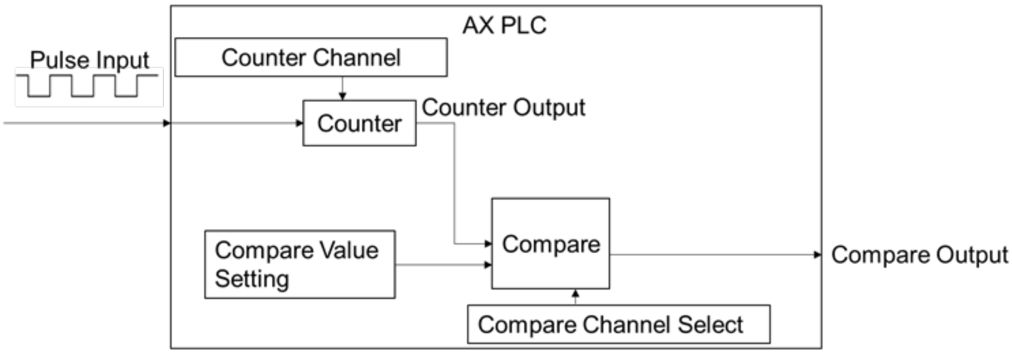


- **DFB_Capture**

Here you can choose from X0~X3 to set up the TrggerDevice. For example, select X0 as trigger source and activate Capture function. As the X0 is triggered, the function block will capture the A and B impulse input value.



- **Compare**



- **DFB_Compare**

After setting up the comparioson position table and the output pulse length then activate the comparison function, the output Y0 will be enabled according to the pulse length. A maximum of 256 positions can be written into the table.

7.7.8 Other Features

7.7.8.1 Changing Current Position

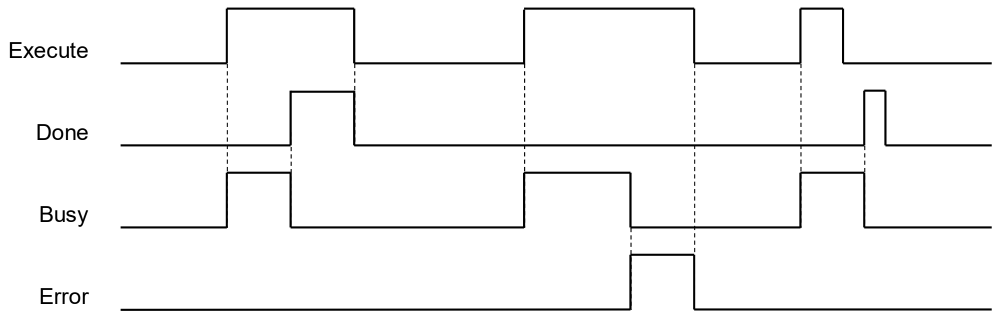
- **MC_SetPosition**

Execution on servo axis turns the current position into the specified target position; on the other hand, execution on encoder axis changes the current feedback position to the specified target position.

Change of the commanded current position also influence the current feedback position. The position lag between two positions remains unchanged.

This function block executes in the way of shifting the coordinate system without any displacement in servo or motor.

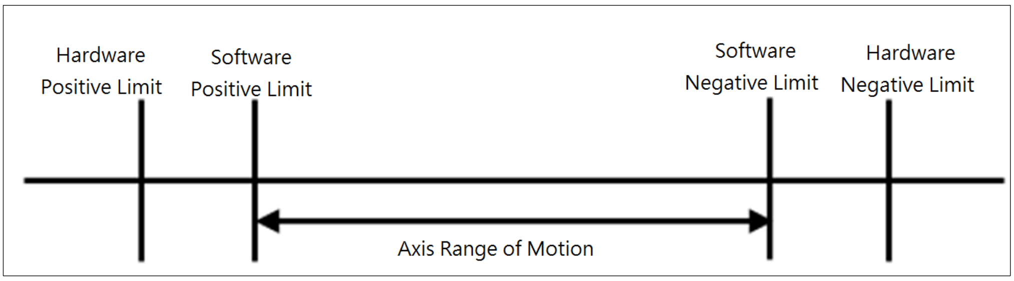
- **Timing Diagram**



7.7.8.2 Software Limit

In addition to hardware limit, software limit serves as another way to limit the range for axis movement.

Values of range for forward and reverse limit must be set in advance of activating software limit*1. To avoid damage to the device resulting from inappropriate operation, software limit is not activated as default.



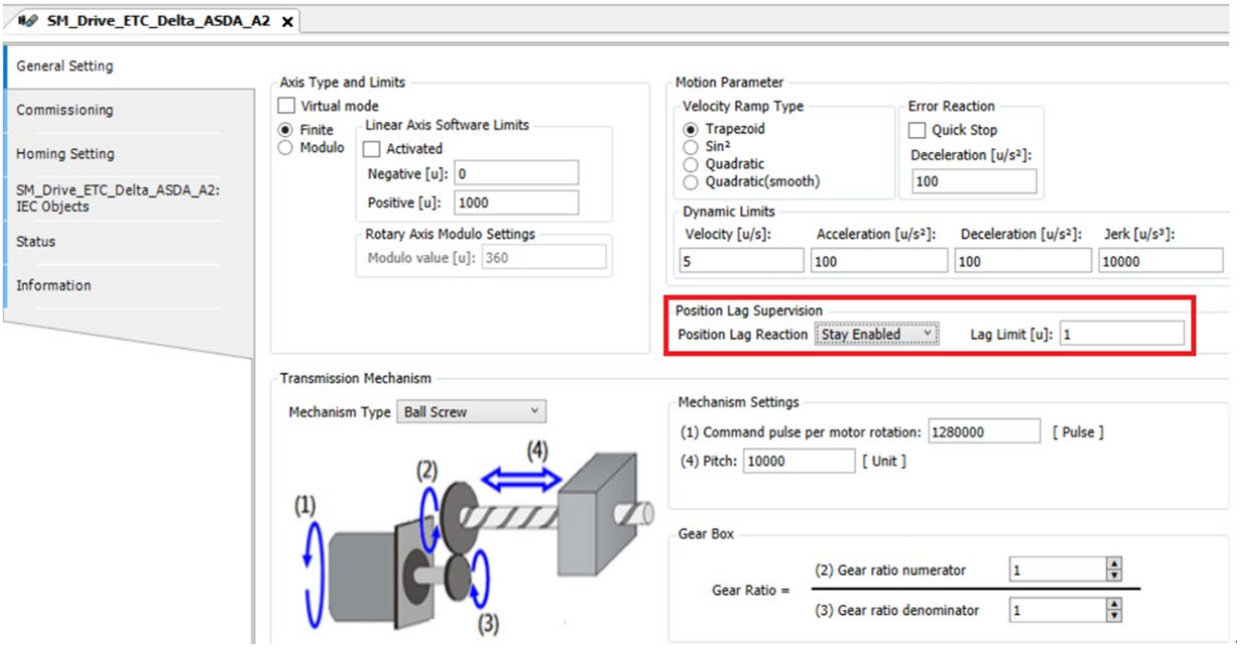
*1：For examples on Stop Method, refer to **Section 7.7.2.4**.

7.7.8.3 Position Lag Settings

The command position and the actual position reported back are 0 when the axis moves, and a runtime error is ouput when the difference between the command position and the reported actual position is too large.

By default, the position lag function is not enabled.

Name	Function
Deactivated	Not enabled.
Disable drive	When the position lag exceeds the set limit, the axis shifts to Servo Off.
Do quickstop	When the position lag exceeds the set limit, the axis shifts to quick stop.
Stay enabled	When the position lag exceeds the set limit, the axis remains Servo On.



7.7.8.4 Cam Switch

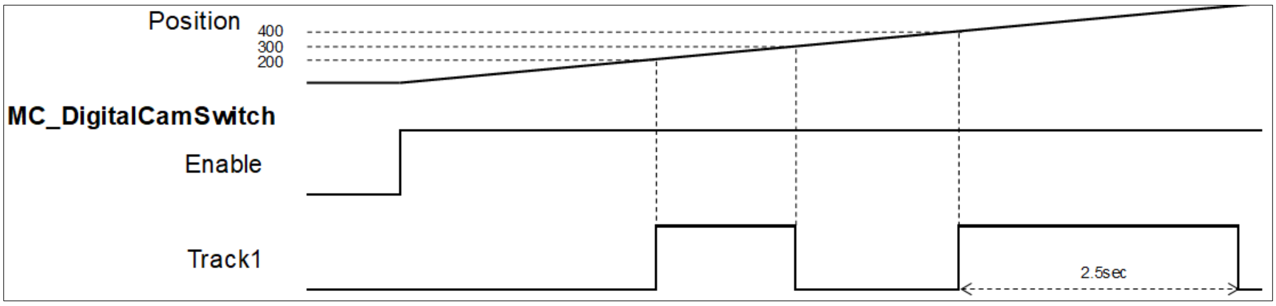
- **MC_DigitalCamSwitch**

Designate the tappet position. It is True when the axis moves to the specified position and turns False when passing by. The following is an example in which two switches are used in the same track with MC_DigitalCamSwitch instruction.

- Parameter setting

Parameter	Type	Switch1	Switch2
TrackNumber	INT	1	1
FirstOnPosition [u]	REAL	200	400
LastOnPosition [u]	REAL	300	-
AxisDirection	INT	0=Both	0=Both
CamSwitchMode	INT	0=Position	1=TIME
Duration	TIME	-	2500ms

- Trigger and timing



- Switch1 on Track1 turns on when the position reaches 200 and turns off when reaching 300.
- Switch1 turns on again when the position reaches 400 but turns off after passing 2500ms.

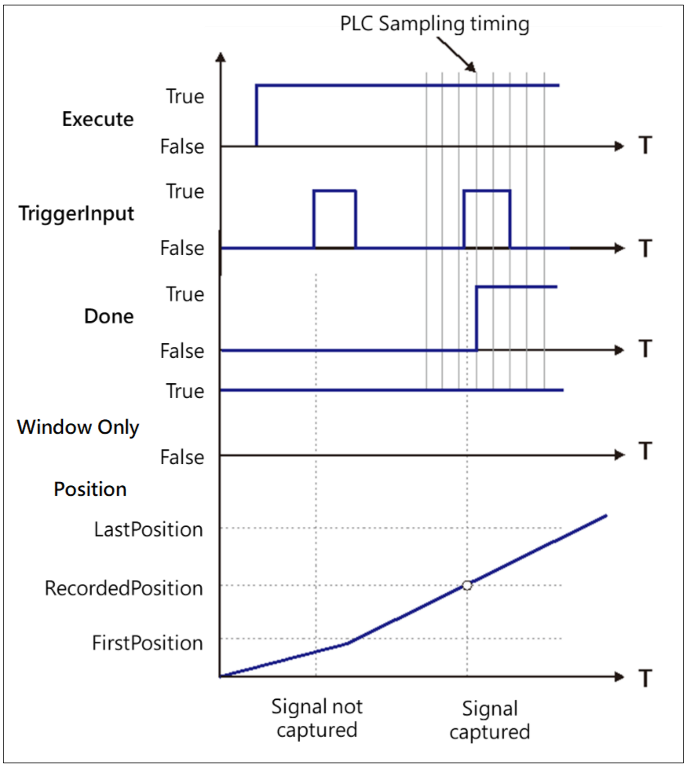
7.7.8.5 Position Capture

In function block MC_TouchProbe, a senser is used as trigger signal. The current position of the servo is captured and recorded when a trigger event occurs.

For each axis it is allowed to set two trigger signals.

Plus, MC_AbortTrigger is used to abort the capture function.

- **Function description**
  - When the input pin Execute of the function block MC_TouchProbe is True, only the effective position value of the first trigger signal will be captured and the subsequent signals are ignored.
  - One function block instance only relates to one MC_TouchProbe instruction. If there are multiple MC_TouchProbe instances on the same axis, another TouchProbeID will be added to MC_TRIGGER_REF to distinguish different TouchProbe actions. TouchProbeID is also associated with MC_AbortTrigger.
  - The operation of window mask function in MC_TouchProbe:



- For the first trigger signal input, the signal is not captured because the axis position has not reached the Window Mask section.
- For the second input, the signal is captured as the axis position has reached the Window Mask section. And the signal Done is output as True in the following cycle right after the capture is completed.

7.8 Programming Example

A programming example is elaborated in the following sections.

7.8.1 Device Framework

The following devices are applied in the example.

Device	Model Name
CPU	AX-8
Power	DVP-PS02
Servo drive	Delta ASDA-A2-E
Servo motor	Delta ECMA-C

7.8.1.1 Utilization

Please refer to the following manuals for information regarding device configuration and wiring.

Device	Reference
CPU and power	Chapter 2
Server drive	Delta servo drive manuals
Wiring for EtherCAT slave	Delta ASDA A2-E EtherCAT Interface Servo Drive User Manual

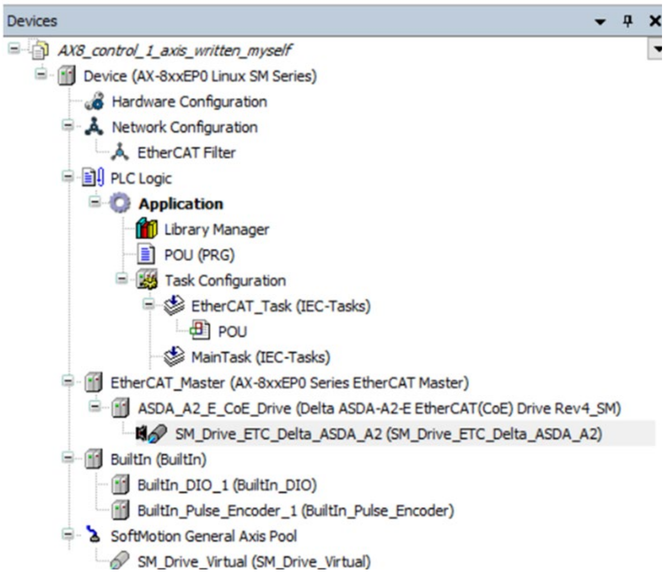
7.8.1.2 Configuration

The following configuration is applied in the example.

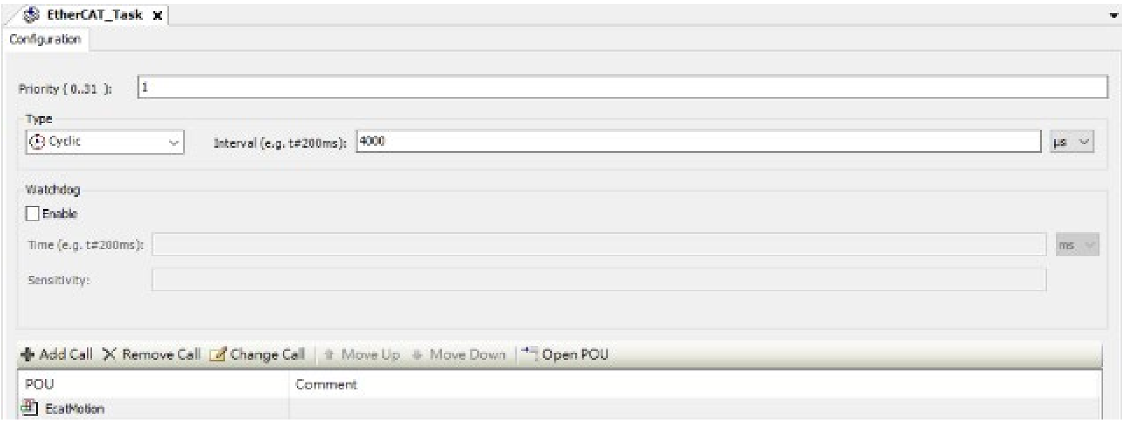
Device	Configuration Setting
Controller	The default settings in Chapter 2 in this manual are applied.
Motion control settings	Chapter 7
Servo Parameters	The default settings of ASDA-A2-E slave (gear ratio = 10000:10000) are applied.

7.8.2 Examples

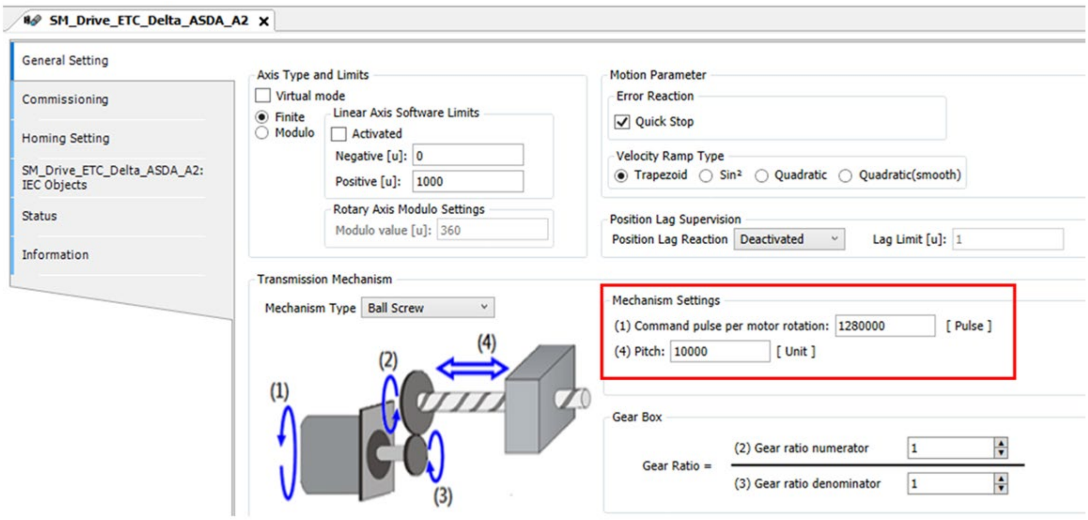
The following example takes the same POU in the EtherCAT task for demonstration. All variables used are declared and utilized in this POU task (the POU naming in LD and ST languages is different for demonstration purpose).



The time for EtherCAT synchronization is set to 4ms.



The electronic gear ratio is set to 10000:10000.



7.8.2.1 Servo On

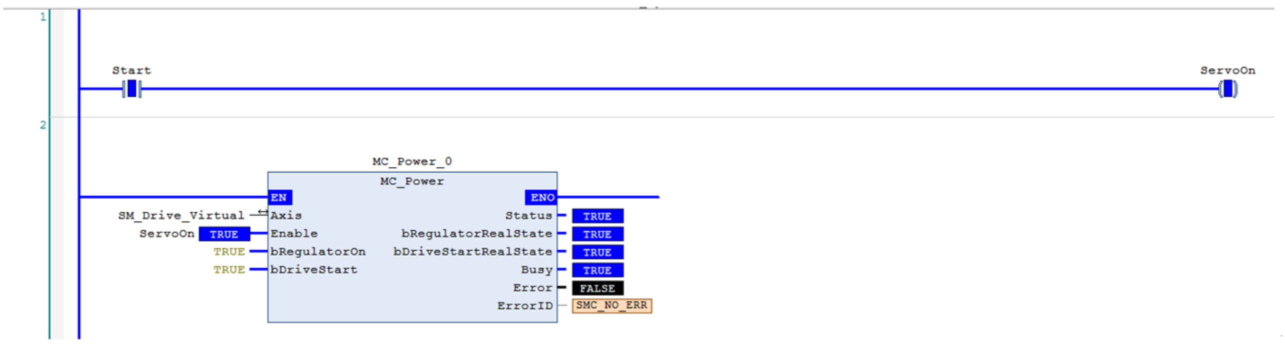
In this example, when EtherCAT data communication is established, the MC_Power (Servo On) instruction is executed to enable the servo drive. Ladder diagram (LD) and structured text (ST) languages are provided in the following example.

- **Main variables used in programming**

Name	Data Type	Initial Value	Annotation
SM_Drive_Virtual	AXIS_REF_SM3	-	Virtual Axis Variables
Start	BOOL	FALSE	When turning on the servo, switch to True to start Servo On

- **LD language**

When Start is True, confirm whether the EtherCAT network communication works. If the communication is normal, enable MC_Power via Servo On output. The activation is succeded when Status is True.



- **ST language**

When Start is True, confirm whether the EtherCAT network communication works. If the communication is normal, enable MC_Power via Servo On output. The activation is succeded when Status is True.

At the same time, you can also observe the output status of the variables through monitoring window with no need to name the output variables.

```
IF Start THEN
        ServoOn :=TRUE;
        ELSE
        ServoOn :=FALSE;
END_IF

//MC_Power
MC_Power_0(
        Axis:= SM_Drive_Virtual,
        Enable:= ServoOn,
        bRegulatorOn:= TRUE,
        bDriveStart:= TRUE,
        Status=> ,
        bRegulatorRealState=> ,
        bDriveStartRealState=> ,
        Busy=> ,
        Error=> ,
        ErrorID=> );
```

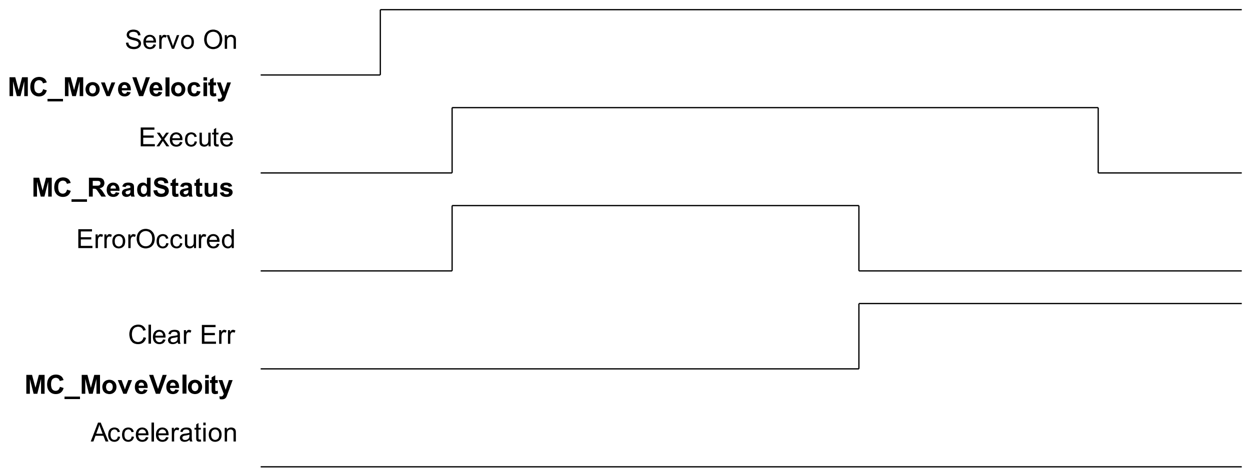
7.8.2.2 Resetting and Monitoring Single-axis Errors

In this example, the error information of variable state can be viewed through the watch table. Take MC_MoveVelocity input as example. When the acceleration is set to 0, Execute is True and Error is True with the ErrorID status shown as Row Data 301, the error message SMC_MV_INVALID_ACCDEC_VALUES can be checked in the watch table. After troubleshooting is done, MC_MoveVelocity functions well by shifting Execute from False to True. As for MC_Reset, it is specially for servo error clearance. Ladder diagram (LD) and structured text (ST) languages are provided in the following example.

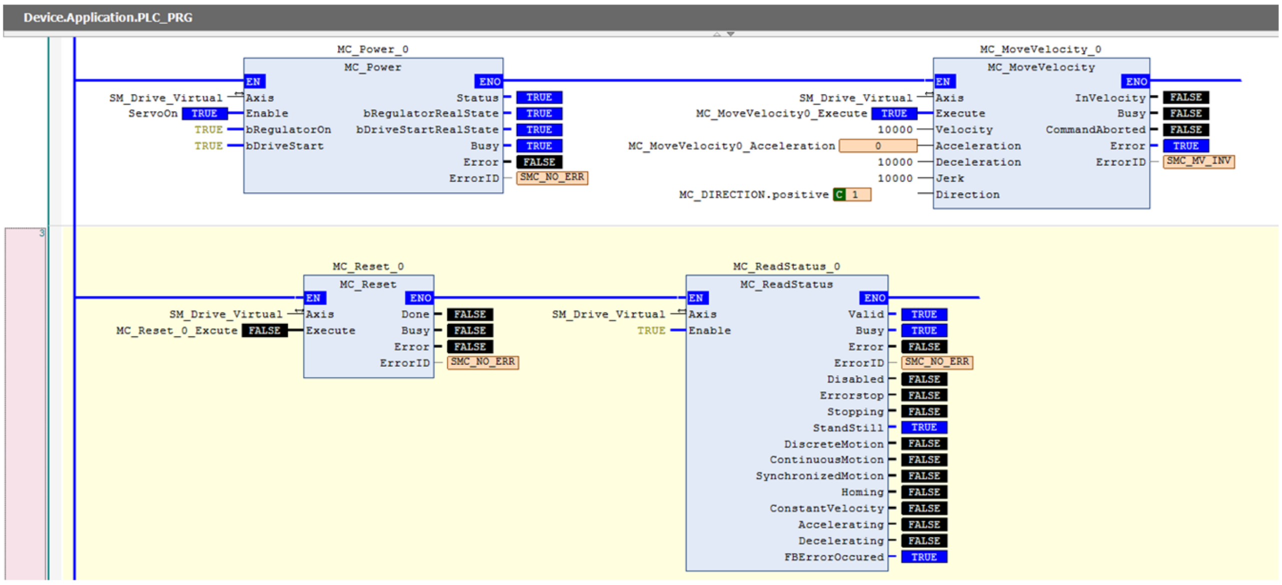
- **Main variables used in programming**

Name	Data Type	Initial Value	Annotation
SM_Drive_Virtual	AXIS_REF_SM3	-	Virtual axis variables
ServoOn	BOOL	FALSE	To enable MC_Power
MC_MoveVelocity0_Execute	BOOL	FALSE	Velocity command for Execute
MC_MoveVelocity0_Acceleration	LREAL	0	Velocity command for Acceleration, used to set acceleration
MC_DIRECTION.positive	MC_Direction	-	Specify the moving direction as forward
FBErrorOccured	MC_ReadStatus	FALSE	True when an error occurs
ClearErr	BOOL	FALSE	When FBErrorOccured is True, SMC_ClearFBError can be triggered to clear errors

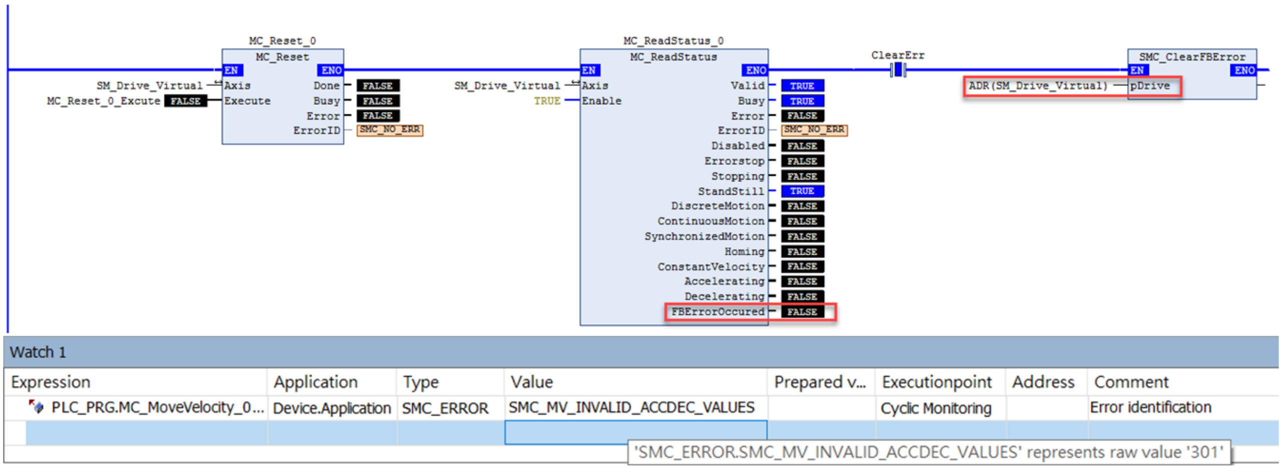
- **Timing diagram**



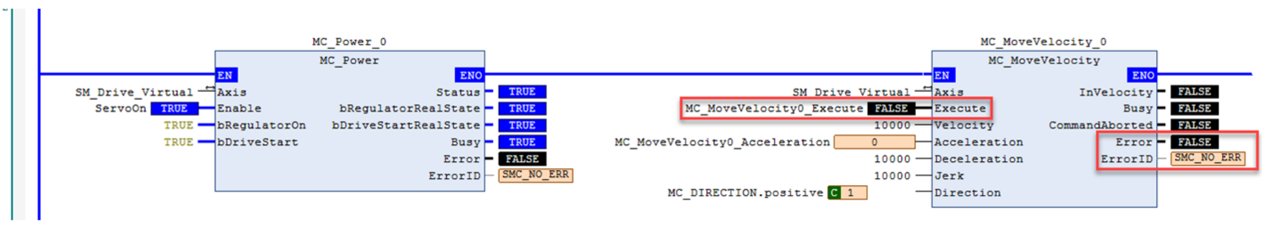
- **LD language**



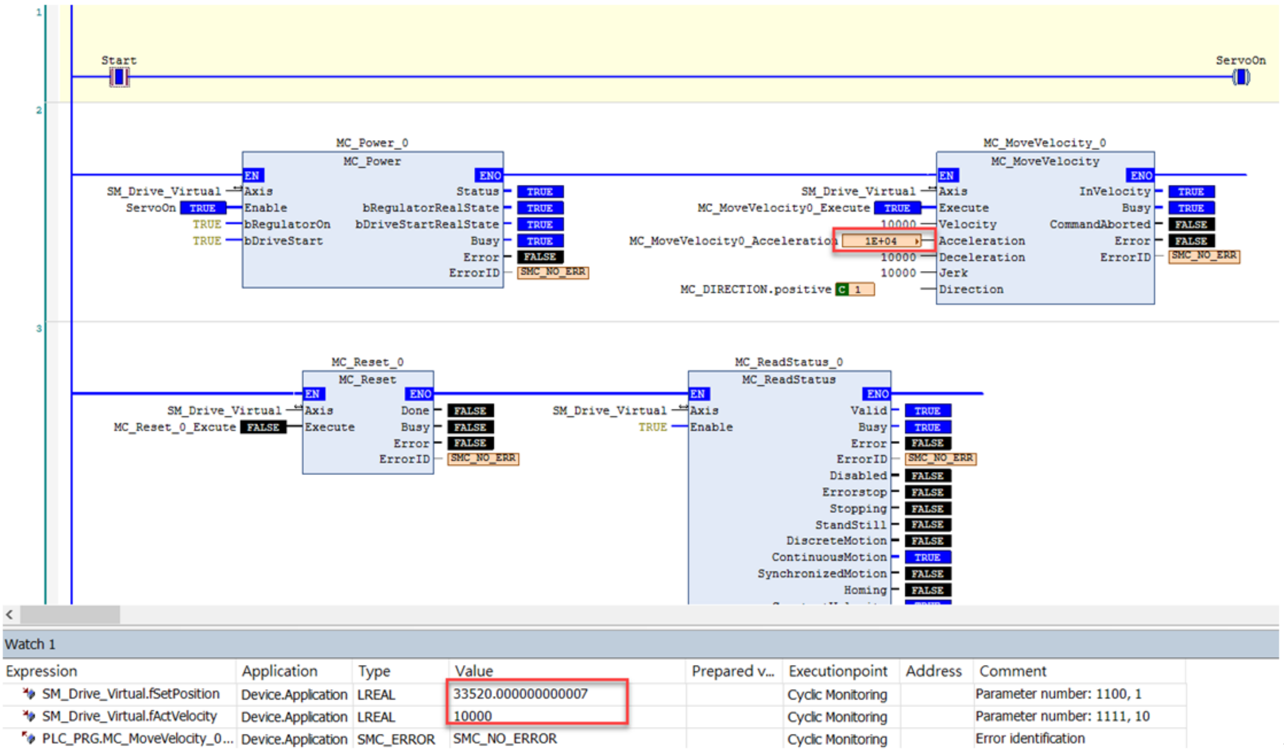
When an error occurs, the output FBErrorOccured of MC_ReadStatus can be cleared via SMC_ClearFBError. As the input to SMC_ClearFBError is transmitted through pointers, it is necessary to add ADR (input variable) and use bool to clear the error flag in function block.



Disable Execute input in MC_MoveVelocity to update the status of Error output.



Set the MC_MoveVelocity acceleration to 10000 and then restart Execute to True. The output of MC_MoveVelocity is Busy and the fSetVelocity and fSetPosition on the watch table are operating normally.



- **ST language**

MC_MoveVelocity_0(
    Axis:= SM_Drive_Virtual,
    Execute:= MC_MoveVelocity0_Execute,
    Velocity:= 10000,
    Acceleration:= MC_MoveVelocity0_Acceleration,
    Deceleration:= 10000,
    Jerk:= 10000,
    Direction:= MC_DIRECTION.positive,
    InVelocity=> ,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=>);

MC_ReadStatus_0(
    Axis:= SM_Drive_Virtual,
    Enable:= TRUE);

Set the MC_MoveVelocity acceleration to 10000 and then restart Execute to True. The output of MC_MoveVelocity is Busy and the fSetVelocity and fSetPosition on the watch table are operating normally.

MC_MoveVelocity_0(
    Axis:= SM_Drive_Virtual,
    Execute:= MC_MoveVelocity0_Execute,

```
    Velocity:= 10000,
    Acceleration:= MC_MoveVelocity0_Acceleration := 10000,
    Deceleration:= 10000,
    Jerk:= 10000,
    Direction:= MC_DIRECTION.positive,
    InVelocity=> ,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );

MC_ReadStatus_0(
    Axis:= SM_Drive_Virtual,
    Enable:= TRUE    );
```

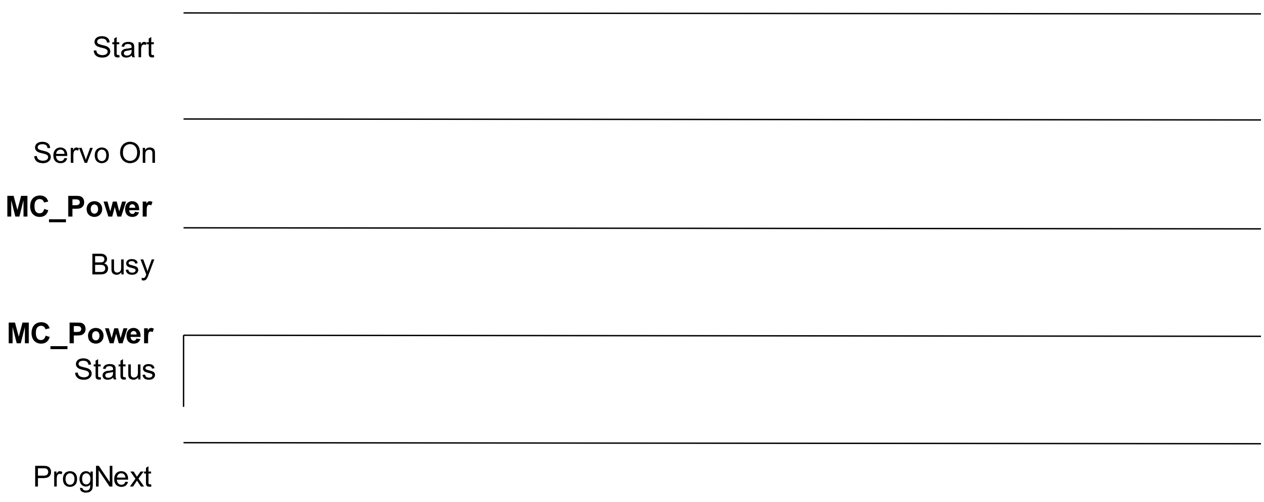
7.8.2.3 Monitoring Instruction Errors

In this example, if an error occurs while executing the MC_Power (Servo On) instruction, no further processing will be performed. However, the ProgNext will indicate whether to proceed with the execution. Ladder diagram (LD) and structured text (ST) languages are provided in the following example.

- **Main variables used in programming**

Name	Data Type	Initial Value	Annotation
SM_Drive_Virtual	AXIS_REF_SM3	-	Virtual axis variables
ServoOn	BOOL	FALSE	To enable MC_Power
ProgNext	BOOL	FALSE	ProgNext indicates whether further execution is possible
MC_Power_0.Status	BOOL	FALSE	Axis is ready to move when the state is True
MC_Power_0.Busy	BOOL	FALSE	Execution of function block has not yet completed when the state is True

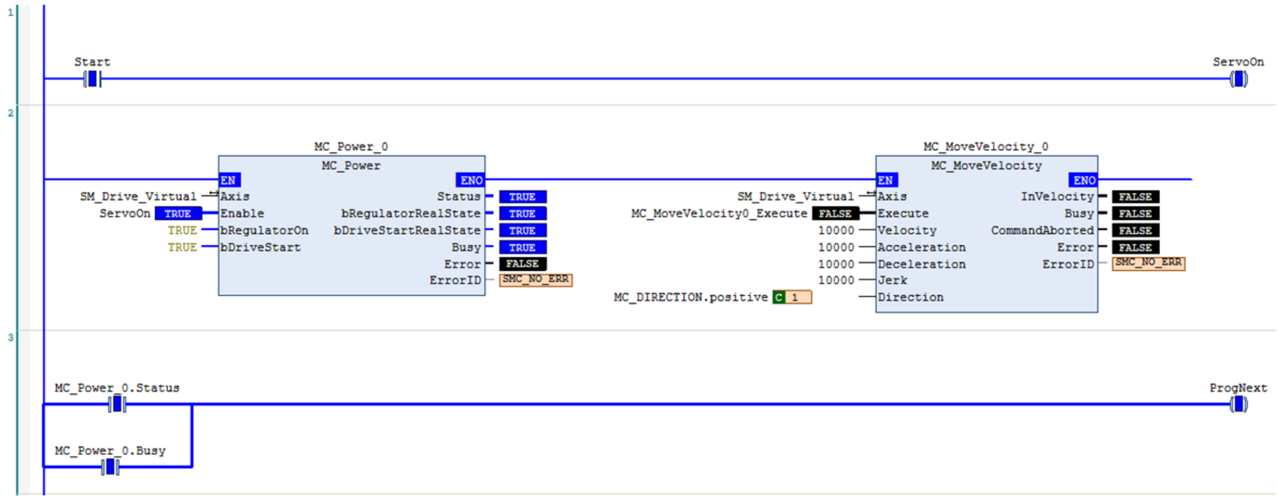
- **Timing diagram**



7

- **LD language**

Check if there is any error in MC_Power before further processing.



- **ST language**

```
IF Start THEN
        ServoOn :=TRUE;
        ELSE
        ServoOn :=FALSE;
END_IF

IF (MC_Power_0.Status=TRUE) OR (MC_Power_0.Busy=TRUE) THEN
        ProgNext :=TRUE;
        ELSE
        ProgNext :=FALSE;
END_IF

//MC_Power
MC_Power_0(
        Axis:= SM_Drive_Virtual,
        Enable:= ServoOn,
        bRegulatorOn:= TRUE,
        bDriveStart:= TRUE,
        Status=> ,
        bRegulatorRealState=> ,
        bDriveStartRealState=> ,
        Busy=> ,
        Error=> ,
        ErrorID=> );
```

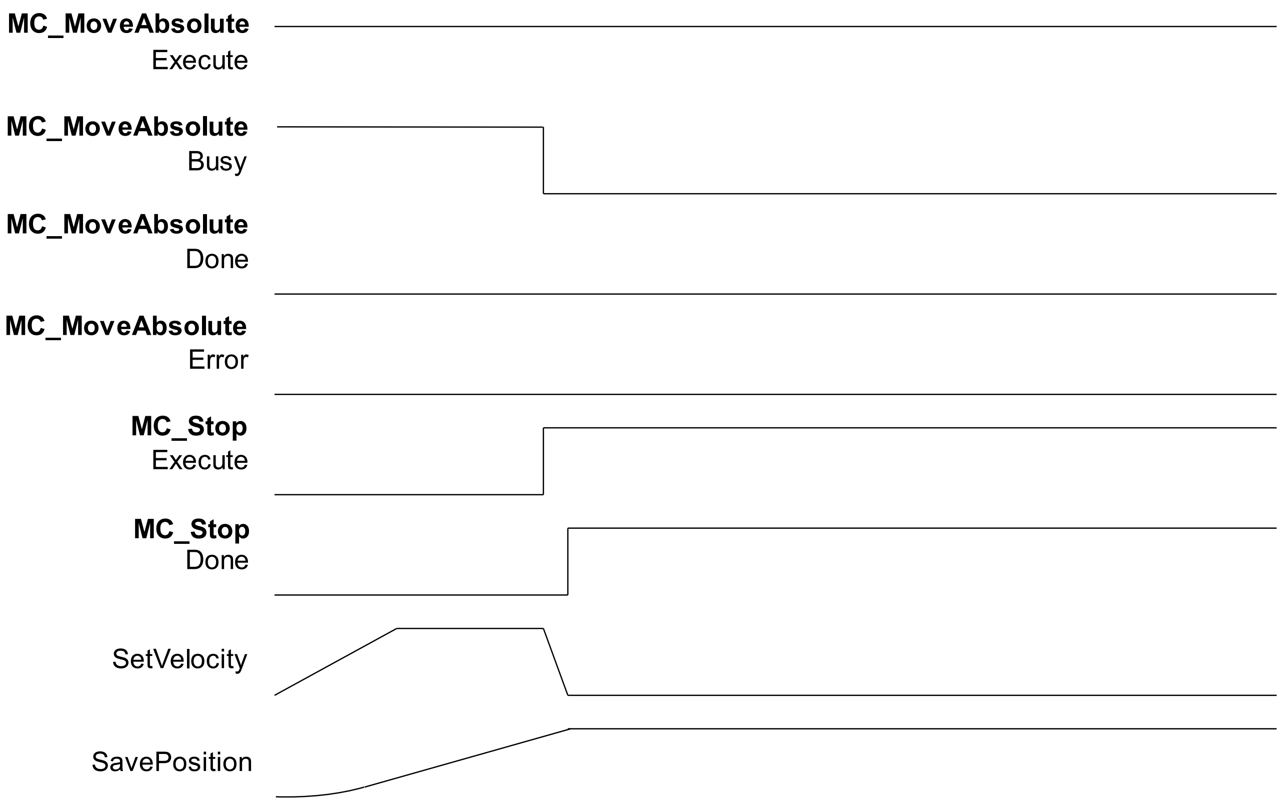
7.8.2.4 Quick Stop for Single Axis

In this example, if minor error occurs during the execution of MC_MoveAbsolute (absolute positioning), use MC_Stop to stop single-axis movement. Ladder diagram (LD) and structured text (ST) languages are provided in the following example.

- **Main variables used in programming**

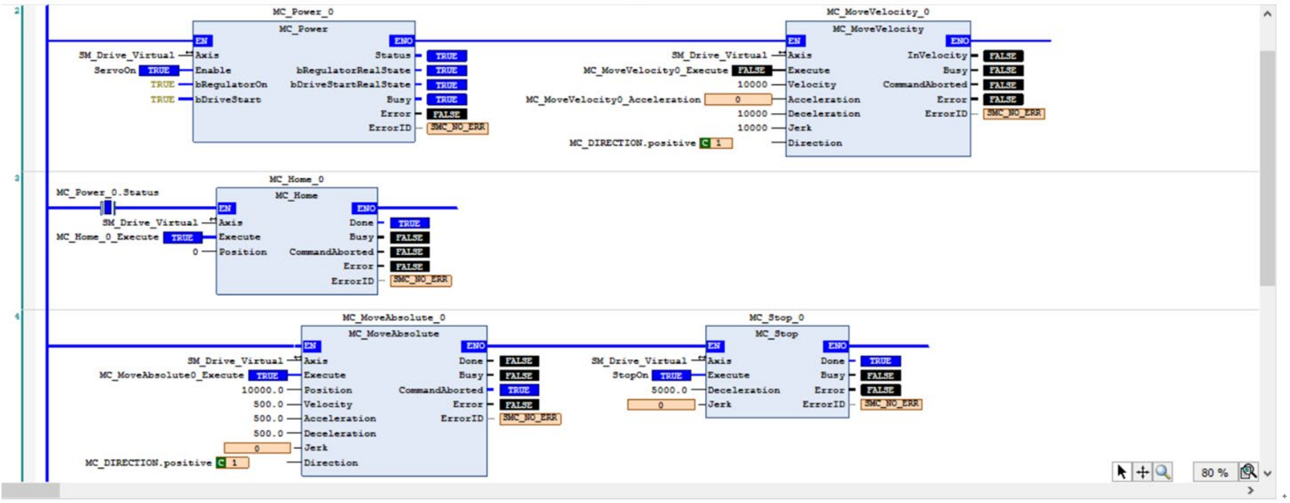
Name	Data Type	Initial Value	Annotation
SM_Drive_Virtual	AXIS_REF_SM3	-	Virtual axis variables
ServoOn	BOOL	FALSE	To enable MC_Power
MC_MoveAbsolute0_Execute	BOOL	FALSE	Execute input of MC_MoveAbsolute
MC_DIRECTION.positive	MC_Direction	-	Specify the moving direction - forward (valid for rotary axis)
StopOn	BOOL	FALSE	Activate MC_Stop when the state is True
MC_Stop_0.Done	BOOL	FALSE	MC_Stop is done when the state is True

- **Timing diagram**



- **LD language**

Execute homing when the status of MC_Power output is normal. Once homing is completed, execute MC_MoveAbsolute (absolute positioning). If an emergency stop is in need, you can execute MC_Stop where absolute positioning will be aborted with the state of CommandAborted output being True. At this time, the axis stops according to the assigned deceleration, and the Done output of MC_Stop shifts to True after the stop instruction is completed.



- **ST language**

The process is the same as that of LD. After MC_Home is finished, the state would be in StandStill.

```
//MC_Power
MC_Power_0(
        Axis:= SM_Drive_Virtual,
        Enable:= ServoOn,
        bRegulatorOn:= TRUE,
        bDriveStart:= TRUE,
        Status=> ,
        bRegulatorRealState=> ,
        bDriveStartRealState=> ,
        Busy=> ,
        Error=> ,
        ErrorID=> );

//MC_Home
IF MC_Power_0.Status THEN
        MC_Home_0(
        Axis:= SM_Drive_Virtual,
        Execute:= MC_Home_0_Execute,
        Position:= 0,
        Done=> ,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=>);
END_IF
```

Execute MC_Stop if an emergency stop is required during execution of MC_MoveAbsolute. As a result, absolute positioning is aborted and the state would be in Stopping.

```
//MC_MoveAbsolute & MC_Stop
MC_MoveAbsolute_0(
    Axis:= SM_Drive_Virtual,
    Execute:= MC_MoveAbsolute0_Execute,
    Position:= 10000.0,
    Velocity:= 500.0,
    Acceleration:= 500.0,
    Deceleration:= 500.0,
    Jerk:= ,
    Direction:= MC_DIRECTION.positive,
    Done=> ,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=>);
    MC_Stop_0(
    Axis:= SM_Drive_Virtual,
    Execute:= StopOn,
    Deceleration:= 5000.0,
    Jerk:= ,
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=>);
```

7.8.2.5 Homing

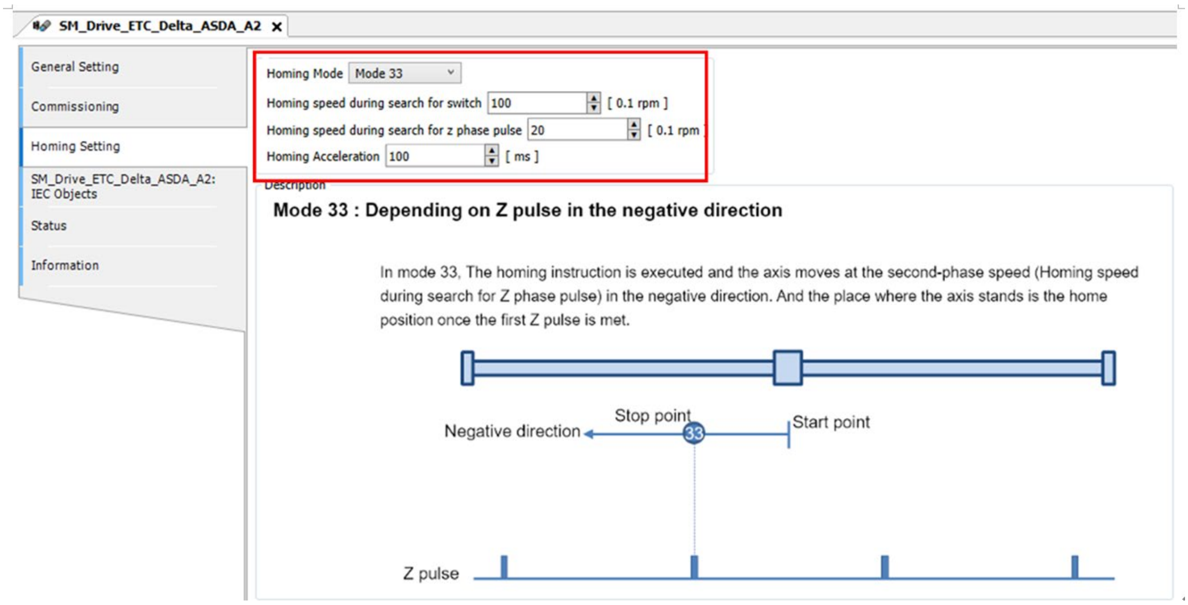
Delta currently supports homing modes 0-35, and the OD used is 6098 (Homing method) / 6099sub1 (Speed during search for switch) / 6099sub2 (Speed during search for zero). For details, refer to the user manual **Delta AC Servo Drive ASDA-A2-E Series**.

In this section, the application of homing instruction is introduced with the example below.

Add A2-E servo to the EtherCAT slave, then specify the parameters of the abovementioned OD:

Homing method = 33 (perform homing once meeting Z pulse)
Speed during search for switch =1000[unit 1.0rpm] (search for limit switch at the speed of 100rpm)
Speed during search for z phase pulse=20[unit 0.1rpm] (search for zero at the speed of 2rpm)

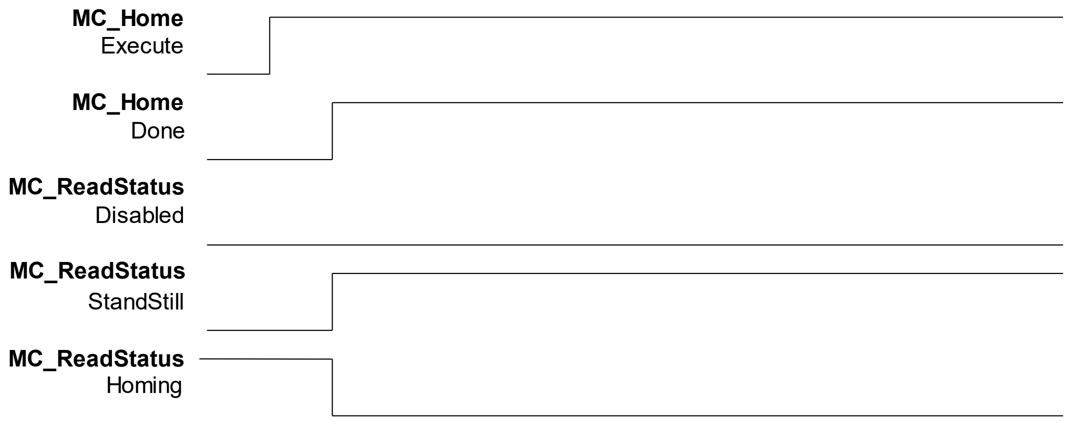
After setup, execute MC_Home in either LD or ST language with the cooresponding OD.



- **Main variables used in programming**

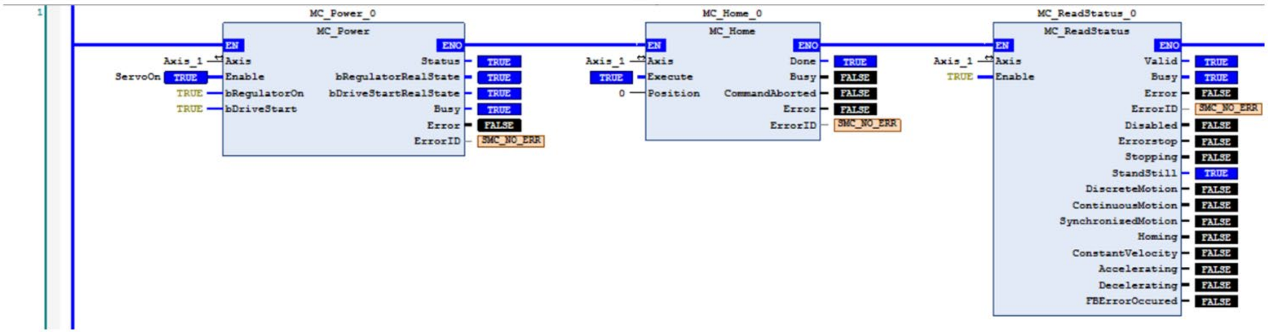
Name	Data Type	Initial Value	Annotation
Axis_1	AXIS_REF_SM3	-	Real axis variables
ServoOn	BOOL	FALSE	To enable MC_Power

- **Timing diagram**



- **LD language**

Confirm that the ouptout status of MC_Power is normal and the state is in Standstill. Then execute MC_Home with the state shifting to homing and back to Standstill after compeletion.



- **ST language**

The process is the same as that of LD. After MC_Home is completed, the state returns to StandStill. The output status can be checked by assigning variables or through the watch table.

```
MC_Home_0(
        Axis:= Axis_1,
        Execute:= ,
        Position:= 0,
        Done=> ,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

MC_ReadStatus_0(
        Axis:= Axis_1,
        Enable:= TRUE,
        Valid=> ,
        Busy=> ,
        Error=> ,
        ErrorID=> ,
        Disabled=> ,
        Errorstop=> ,
        Stopping=> ,
        StandStill=> ,
        DiscreteMotion=> ,
        ContinuousMotion=> ,
        SynchronizedMotion=> ,
        Homing=> ,
        ConstantVelocity=> ,
        Accelerating=> ,
        Decelerating=> ,
        FBErrorOccured=>);
```

7

7.8.2.6 Absolute Positioning

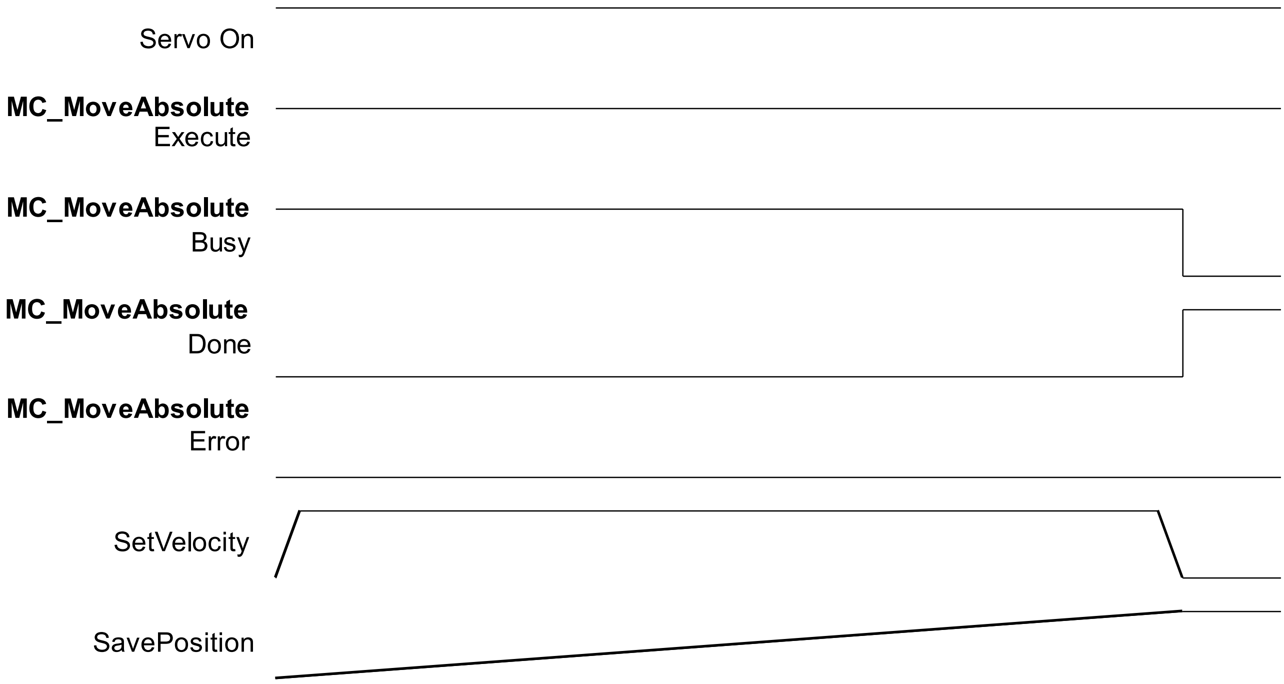
The example below introduces the way to perform displacement at one speed via MC_MoveAbsolute.

Ladder diagram (LD) and structured text (ST) languages are provided in the following example.

- **Main variables used in programming**

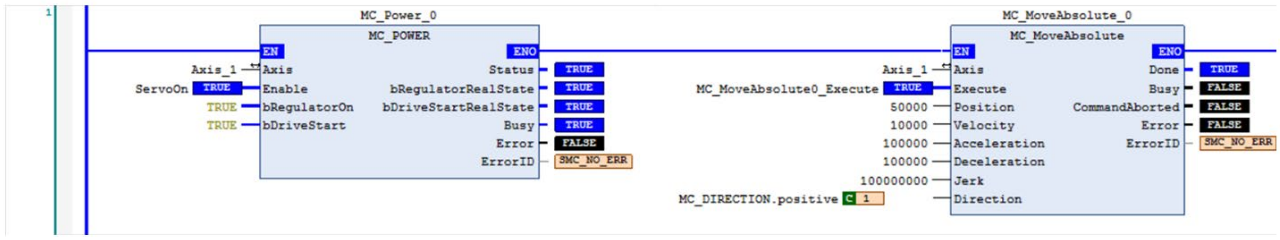
Name	Data Type	Initial Value	Annotation
Axis_1	AXIS_REF_SM3	-	Real axis variables
ServoOn	BOOL	FALSE	To enable MC_Power
MC_MoveAbsolute0_Execute	BOOL	FALSE	Execute input of MC_MoveAbsolute
MC_DIRECTION.positive	MC_Direction	-	Specify the moving direction - forward (valid for rotary axis)

- **Timing diagram**



- **LD language**

Confirm that the ouptout status of MC_Power is normal. Then execute MC_MoveAbsolute to move from the starting position 0 to the assigned position 50000.



- **ST language**

```
MC_Home_0(
        Axis:= Axis_1,
        Execute:= ,
        Position:= 0,
        Done=> ,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

MC_MoveAbsolute_0(
        Axis:= Axis_1,
        Execute:= MC_MoveAbsolute0_Execute,
        Position:= 50000,
        Velocity:= 10000,
        Acceleration:= 100000,
        Deceleration:= 100000,
        Jerk:= 100000,
        Direction:= SM3_Basic.MC_DIRECTION.positive,
        Done=> ,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

7.8.2.7 Switching Cam Tables during Cam Operation

The example shows how to switch cam table when the cam is in operation.

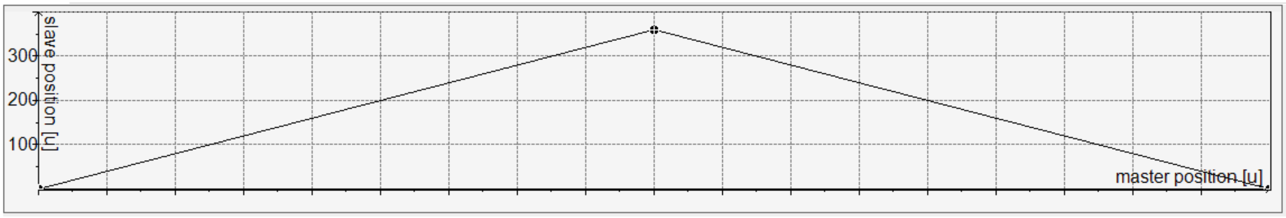
Add master and slave axes and create two MC_CamIn instructions to switch between two cam tables. Different output parameters are configured for the two cams. Use CamTable1 when the instruction position of the master axis is below 3000 and switch to CamTable2 when the position exceeds 3000.

Two programming languages, ladder diagram (LD) and structured text (ST), are provided in the following example.

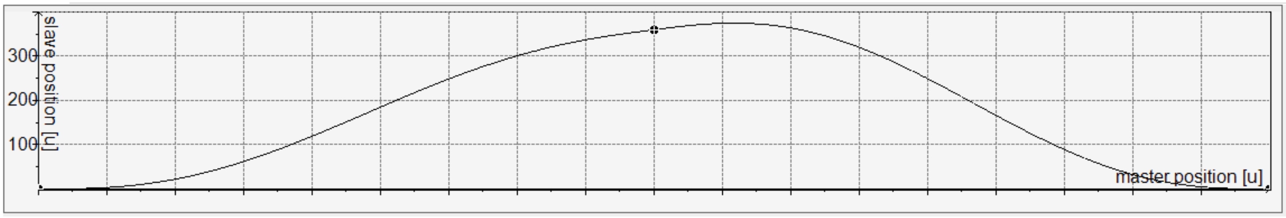
- **Main variables used in programming**

Name	Variable Type	Initial Value	Annotation
Axis_Master	AXIS_REF_VIRTUAL_SM3	-	Master-related axis variables
Axis_Slave	AXIS_REF_VIRTUAL_SM3	-	Slave-related axis variables
CamTable1	MC_CAM_REF	-	Relevant variables in cam table 1
CamTable2	MC_CAM_REF	-	Relevant variables in cam table 2
StartFlag	BOOL	FALSE	If this variable is TRUE and communication with the axes is normal, Servo On will be activated with the subsequent actions being performed
MC_Power0_Status	BOOL	FALSE	Variables of MC_Power output Status for master; TRUE when Servo On
MC_Power1_Status	BOOL	FALSE	Variables of MC_Power output Status for slave; TRUE when Servo On
MC_Home0_Done	BOOL	FALSE	Variables of MC_Home output Done for master; TRUE when homing is completed
MC_Home1_Done	BOOL	FALSE	Variables of MC_Home output Done for slave; TRUE when homing is completed
MC_MoveAbs_Busy	BOOL	FALSE	Variables of MC_MoveAbsolute output Busy for master; TRUE when in execution
CamTableSelect	MC_CAM_REF	-	Variables of the coorsponding cam table which is specified
CamTable1_En	BOOL	FALSE	TRUE when CamTable1 is chosen
CamTable2_En	BOOL	FALSE	TRUE when CamTable2 is chosen
CamTableID	MC_CAM_ID	-	The internal data structure of the selected cam table, which is from MC_CamTableSelect and uased as MC_CamIn input
MC_CamIn1_InSync	BOOL	FALSE	Variables of MC_CamIn output InSync in cam table 1; TRUE when master and slave cams are synchronized
MC_CamIn2_InSync	BOOL	FALSE	Variables of MC_CamIn output InSync in cam table 2; TRUE when master and slave cams are synchronized

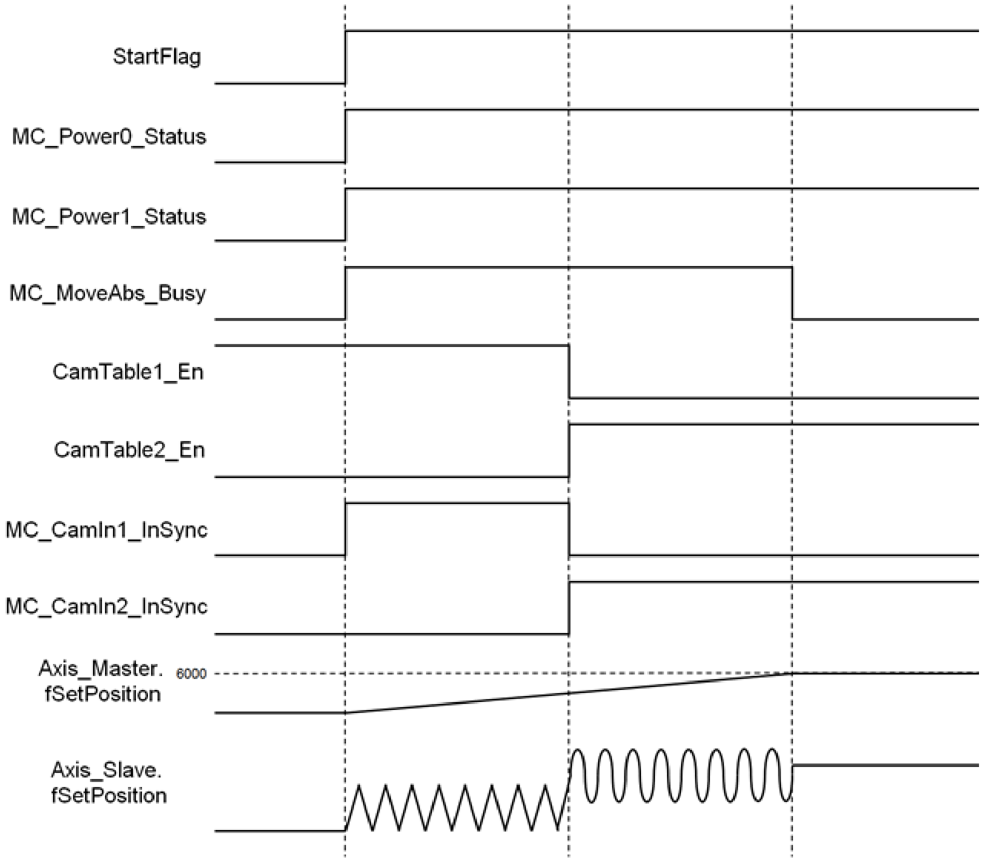
- **CamTable1:**



- **CamTable2:**



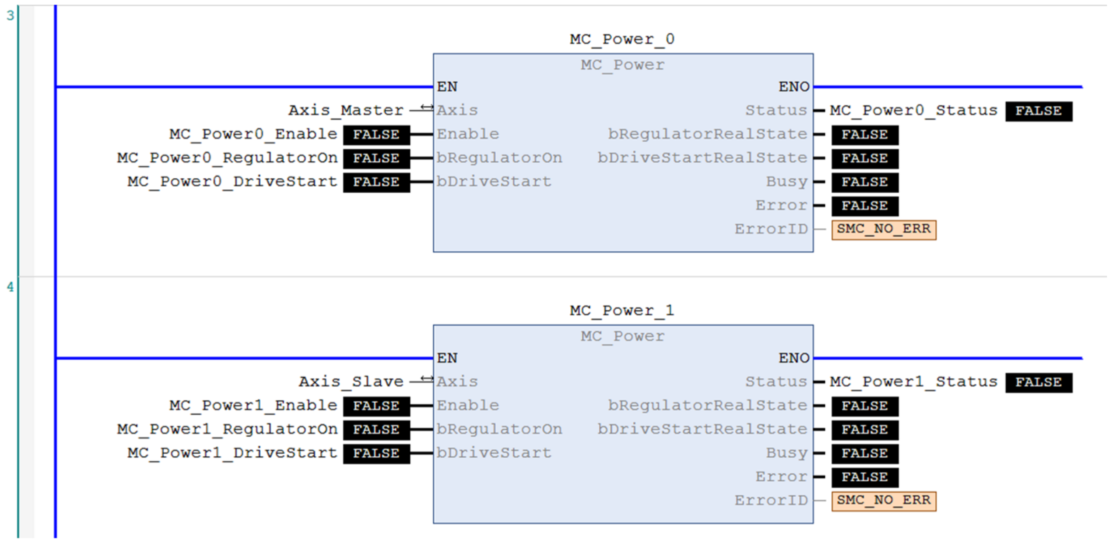
- **Timing Diagram**



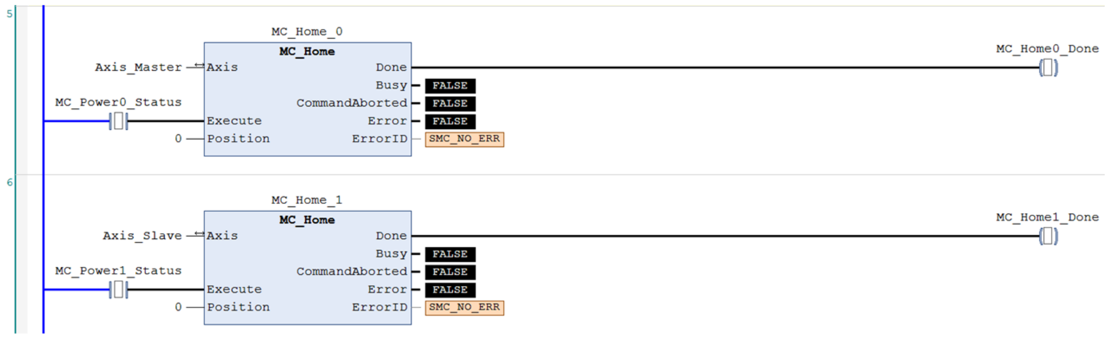
- **LD language**

Set StartFlag to True, then it will check whether the communication of the master axis and slave axes is normal or not.

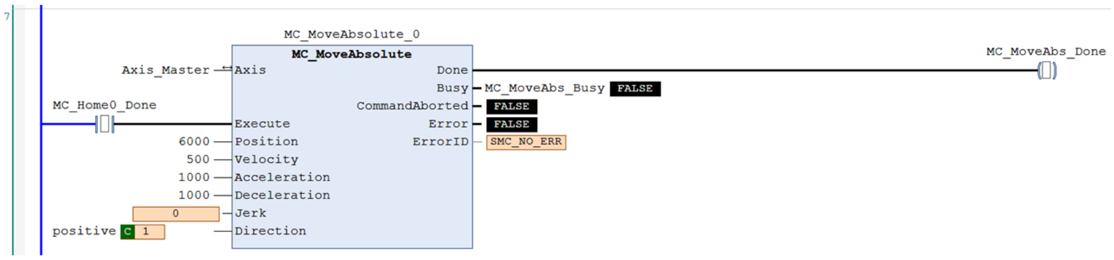
If normal, the master and slave axes are then set to the state of Servo On.



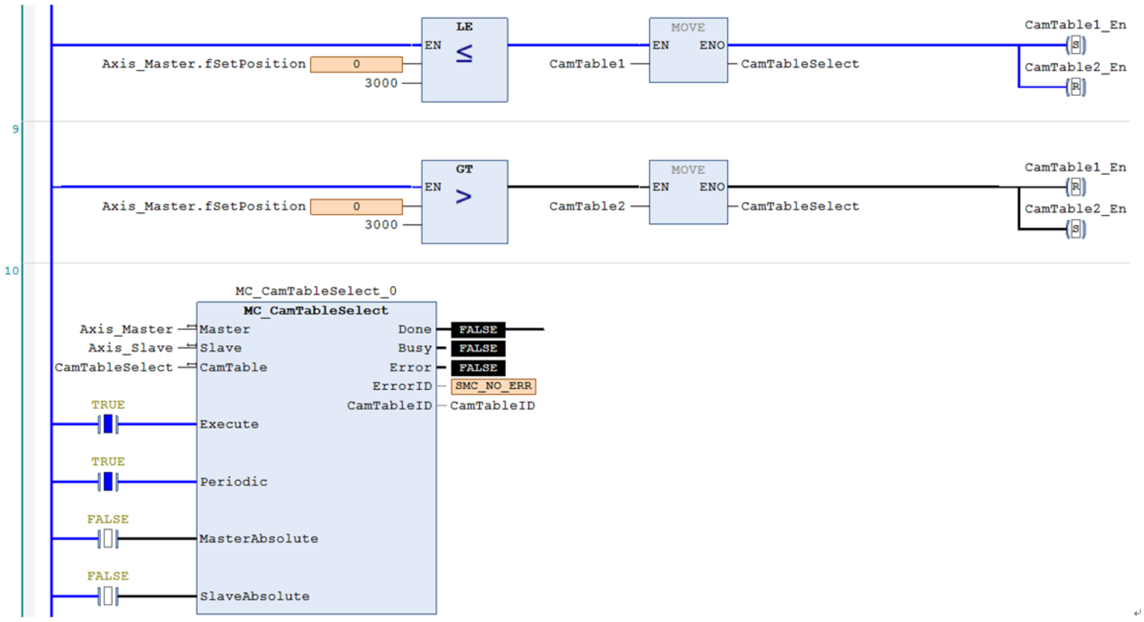
If the master and slave axes are in the Servo On state with unsure starting point, perform homing once.



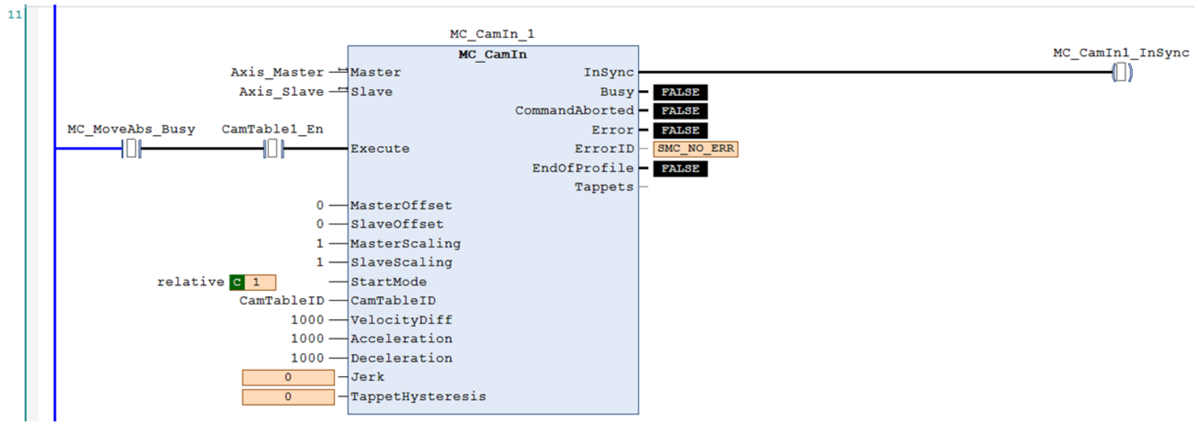
After homing of the master axis is completed, execute MC_MoveAbsolute.



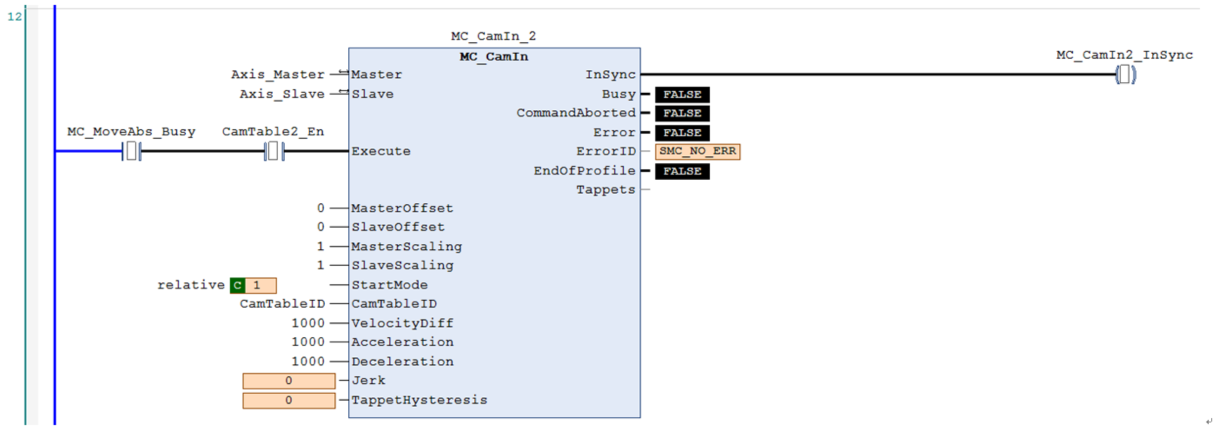
Use CamTable1 when the instruction position of the master axis is under 3000 (CamTable1_En=True; CamTable2_En=False); instead, choose CamTable2 when it is greater than 3000 (CamTable1_En=False, CamTable2_En=True). Set the respective cam table according to the instruction MC_CamTableSelect.



When the master axis performs absolute positioning and CamTable1_En=True, go with CamTable1.



When the master axis performs absolute positioning and CamTable2_En=True, go with CamTable2.



- **ST language**

Set StartFlag to True, then it will check whether the communication of the master axis and slave axes is normal or not.

```
IF StartFlag = TRUE THEN
    IF Axis_Master.bCommunication = TRUE THEN
        MC_Power0_Enable := TRUE;
        MC_Power0_RegulatorOn := TRUE;
        MC_Power0_DriveStart := TRUE;
    END_IF

    IF Axis_Slave.bCommunication = TRUE THEN
        MC_Power1_Enable := TRUE;
        MC_Power1_RegulatorOn := TRUE;
        MC_Power1_DriveStart := TRUE;
    END_IF
END_IF
```

If normal, the master and slave axes are then set to the state of Servo On.

```
MC_Power_0(
    Axis:= Axis_Master,
    Enable:= MC_Power0_Enable,
    bRegulatorOn:= MC_Power0_RegulatorOn,
    bDriveStart:= MC_Power0_DriveStart,
    Status=> MC_Power0_Status,
    bRegulatorRealState=> ,
    bDriveStartRealState=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );

MC_Power_1(
    Axis:= Axis_Slave,
    Enable:= MC_Power1_Enable,
    bRegulatorOn:= MC_Power1_RegulatorOn,
    bDriveStart:= MC_Power1_DriveStart,
    Status=> MC_Power1_Status,
    bRegulatorRealState=> ,
    bDriveStartRealState=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

If the master and slave axes are in the Servo On state with unsure starting point, perform homing once.

```
IF MC_Power0_Status = TRUE THEN
    MC_Home0_Execute := TRUE;
END_IF
IF MC_Power1_Status = TRUE THEN
    MC_Home1_Execute := TRUE;
END_IF
```

```
MC_Home_0(
        Axis:= Axis_Master,
        Execute:= MC_Home0_Execute,
        Position:= 0,
        Done=> MC_Home0_Done,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=>);

MC_Home_1(
        Axis:= Axis_Slave,
        Execute:= MC_Home1_Execute,
        Position:= 0,
        Done=> MC_Home1_Done,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

After homing of the master axis is completed, execute MC_MoveAbsolute.

```
MC_MoveAbsolute(
        Axis:= Axis_Master,
        Execute:= MC_Home1_Done,
        Position:= 6000,
        Velocity:= 500,
        Acceleration:= 1000,
        Deceleration:= 1000,
        Jerk:= ,
        Direction:= positive,
        Done=> MC_MoveAbs_Done,
        Busy=> MC_MoveAbs_Busy,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

Use CamTable1 when the instruction position of the master axis is under 3000 (CamTable1_En=True; CamTable2_En=False); instead, choose CamTable2 when it is greater than 3000 (CamTable1_En=False, CamTable2_En=True). Set the respective cam table according to the instruction MC_CamTableSelect.

```
IF Axis_Master.fSetPosition > 3000 THEN
        CamTableSelect := CamTable2;
        CamTable1_En := FALSE;
        CamTable2_En := TRUE;
ELSE
        CamTableSelect := CamTable1;
        CamTable1_En := TRUE;
        CamTable2_En := FALSE;
END_IF
```

```
IF (CamTable1_En = TRUE) OR (CamTable2_En = TRUE) THEN
    CamTable_En := TRUE;
END_IF

MC_CamTableSelect(
    Master:= Axis_Master,
    Slave:= Axis_Slave,
    CamTable:= CamTableSelect,
    Execute:= CamTable_En,
    Periodic:= TRUE,
    MasterAbsolute:= FALSE,
    SlaveAbsolute:= FALSE,
    Done=> MC_CamTableSelect_Done,
    Busy=> ,
    Error=> ,
    ErrorID=> ,
    CamTableID=> CamTableID);
```

When the master axis performs absolute positioning and CamTable1_En=True, go with CamTable1.

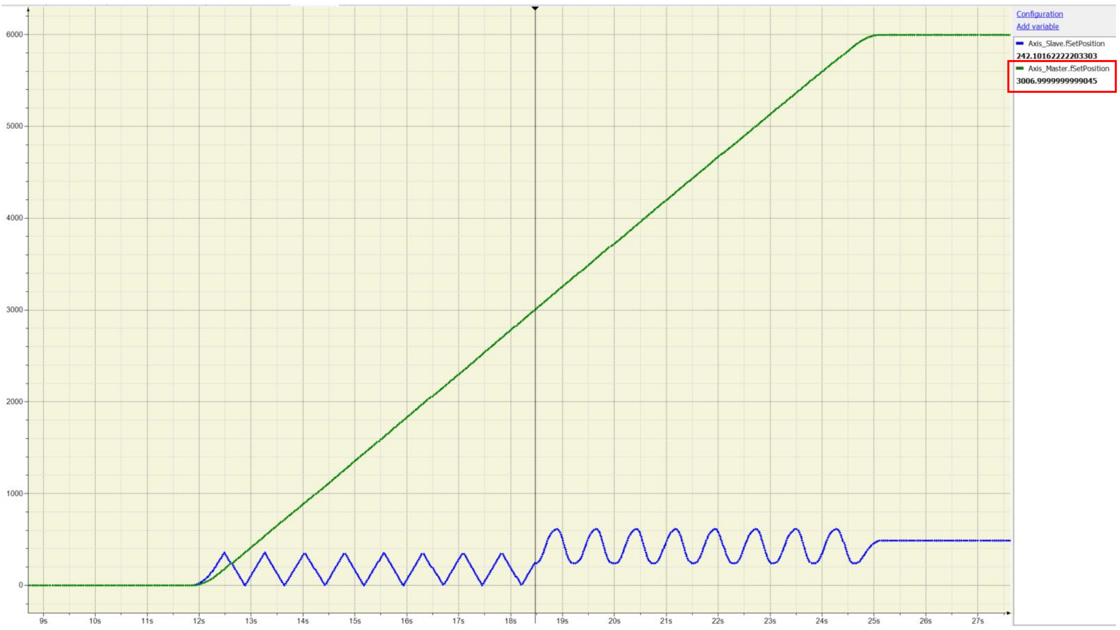
```
IF (MC_MoveAbs_Busy = TRUE) AND (CamTable1_En = TRUE) THEN
    MC_CamIn_1(
    Master:= Axis_Master,
    Slave:= Axis_Slave,
    Execute:= TRUE,
    MasterOffset:= 0,
    SlaveOffset:= 0,
    MasterScaling:= 1,
    SlaveScaling:= 1,
    StartMode:= relative,
    CamTableID:= CamTableID,
    VelocityDiff:= 1000,
    Acceleration:= 1000,
    Deceleration:= 1000,
    Jerk:= ,
    TappetHysteresis:= ,
    InSync=> MC_CamIn1_Insync,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> ,
    EndOfProfile=> ,
    Tappets=> );
END_IF
```

When the master axis performs absolute positioning and CamTable2_En=True, go with CamTable2.

```
IF (MC_MoveAbs_Busy = TRUE) AND (CamTable2_En = TRUE) THEN
    MC_CamIn_2(
    Master:= Axis_Master,
```

```
        Slave:= Axis_Slave,
        Execute:= TRUE,
        MasterOffset:= 0,
        SlaveOffset:= 0,
        MasterScaling:= 1,
        SlaveScaling:= 1,
        StartMode:= relative,
        CamTableID:= CamTableID,
        VelocityDiff:= 1000,
        Acceleration:= 1000,
        Deceleration:= 1000,
        Jerk:= ,
        TappetHysteresis:= ,
        InSync=> MC_CamIn2_Insync,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> ,
        EndOfProfile=> ,
        Tappets=> );
END_IF
```

Based on the settings above, switch between two different cam tables when the master position is over 3000.



7.8.2.8 Executing Master Phase Offset during Cam Operation

After the slave axis is aborted during execution of MC_phasing, it synchronizes with the master axis which is under control. When PhasingActive is True, execute MC_Phasing to compensate the phase offset of the master axis. The slave axis will move synchronously after offset is done.

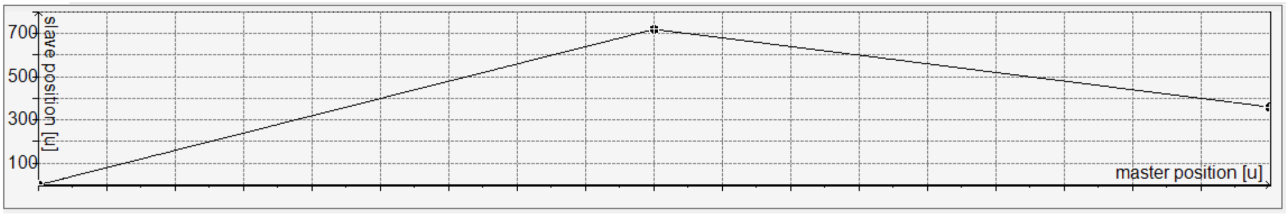
Two programming languages, ladder diagram (LD) and structured text (ST), are provided in the following example.

- **Main Variables Used in Programming**

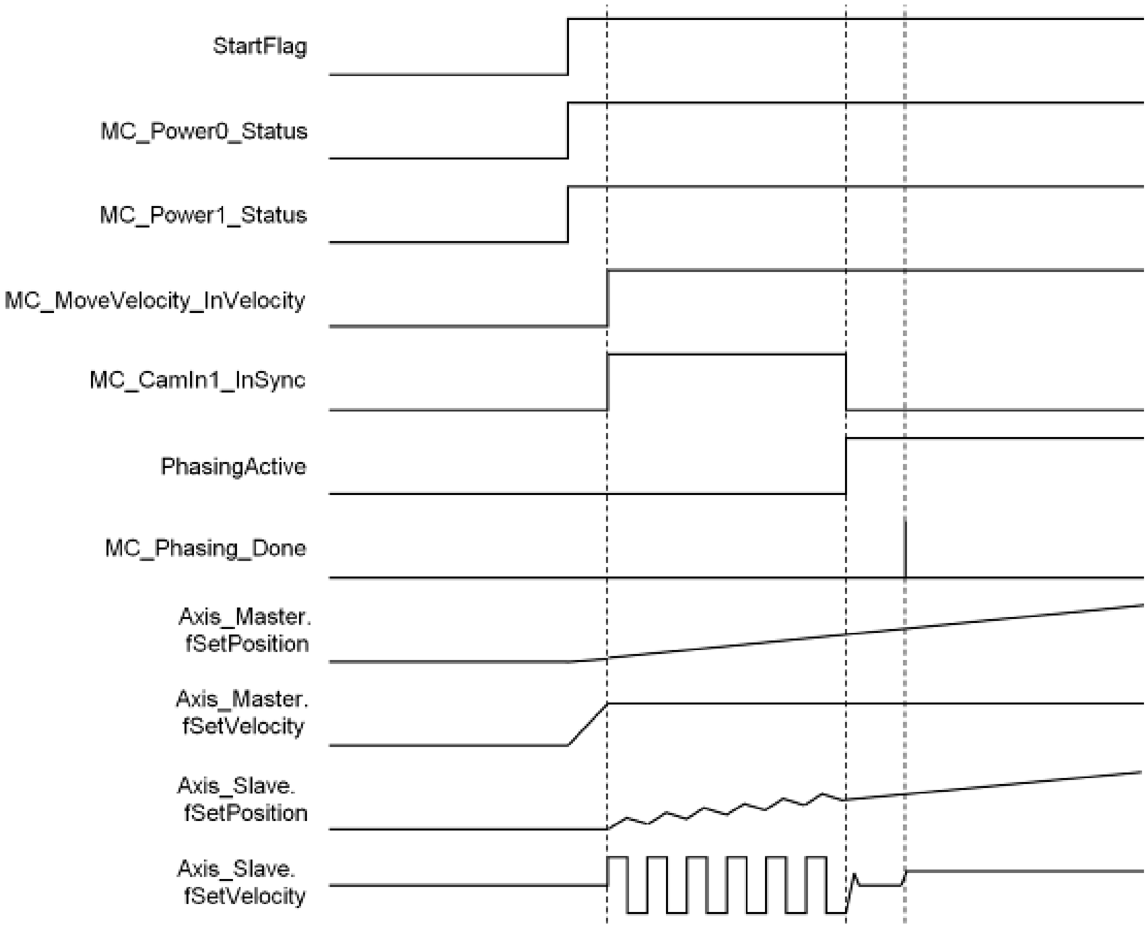
Name	Variable Type	Initial Value	Annotation
Axis_Master	AXIS_REF_ VIRTUAL_SM3	-	Master-related axis variables
Axis_Slave	AXIS_REF_ VIRTUAL_SM3	-	Slave-related axis variables
CamTable	MC_CAM_REF	-	Relevant variables of cam table
StartFlag	BOOL	FALSE	If this variable is TRUE and axis communication is normal, Servo On will be activated with the subsequent actions being performed
MC_Power0_Status	BOOL	FALSE	Variables of MC_Power output Status for master; TRUE when Servo On
MC_Power1_Status	BOOL	FALSE	Variables of MC_Power output Status for slave; TRUE when Servo On
MC_Home0_Done	BOOL	FALSE	Variables of MC_Home output Done for master; TRUE when homing is completed
MC_Home1_Done	BOOL	FALSE	Variables of MC_Home output Done for slave; TRUE when homing is completed
MC_MoveVelocity_ Velocity	LREAL	500	The target velocity for the master axis to move at a constant velocity
MC_MoveVelocity_ InVelocity	BOOL	FALSE	The variables of MC_MoveVelocity output InVelocity; TRUE when the specified velocity reaches the target velocity
CamTableID	MC_CAM_ID	-	The internal data structure of the selected cam table, which is from MC_CamTableSelect and uased as MC_CamIn input
MC_CamIn1_InSync	BOOL	FALSE	Variables of MC_CamIn output InSync in cam table 1; TRUE when master and slave cams are synchronized
PhasingActive	BOOL	FALSE	If this variable is TRUE and the cam is in InSync, MC_Phasing begins execution
MC_Phasing_PhaseShift	LREAL	500	Specify the relative phase offset for the master axis and the slave axis
MC_Phasing_Velocity	LREAL	300	Specify the relative velocity of the master axis and slave axis when performing phase offset

Name	Variable Type	Initial Value	Annotation
MC_Phasing_Done	BOOL	FALSE	The variable of MC_Phasing output Done; TRUE when phase offset is completed

- CamTable



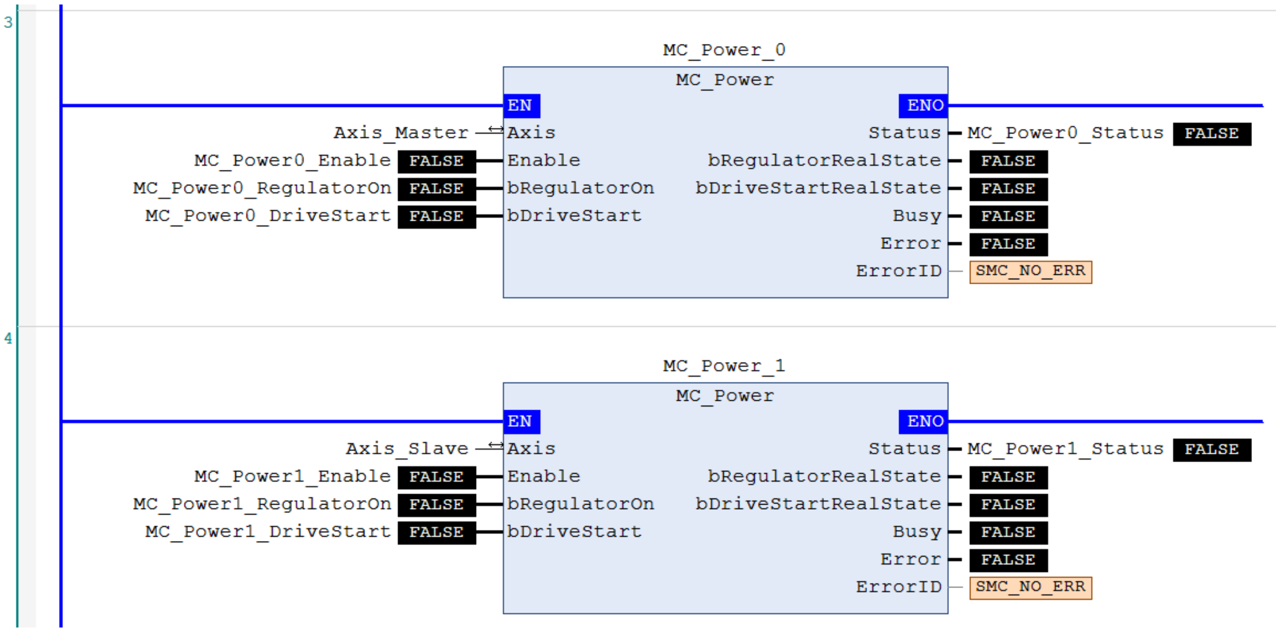
- **Timing diagram**



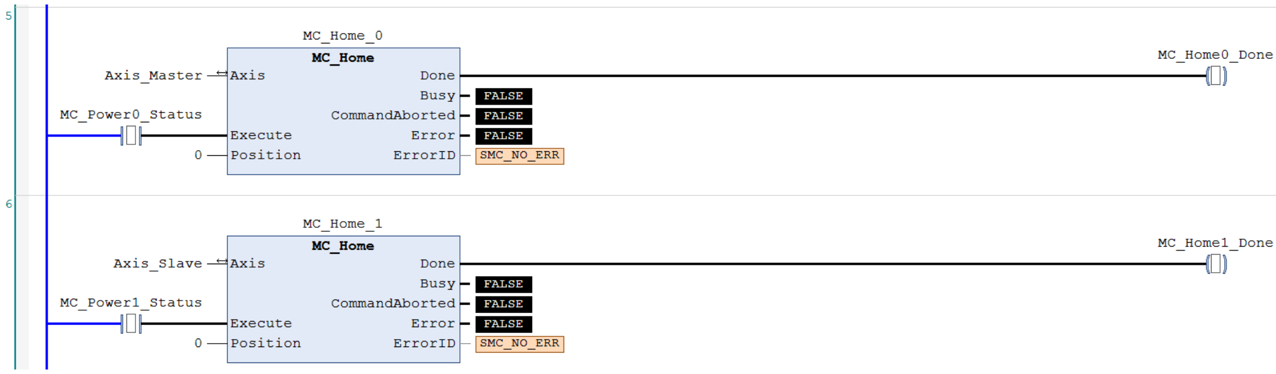
- **LD language**

Set StartFlag to True, then it will check whether the communication of the master axis and the slave axis is normal or not.

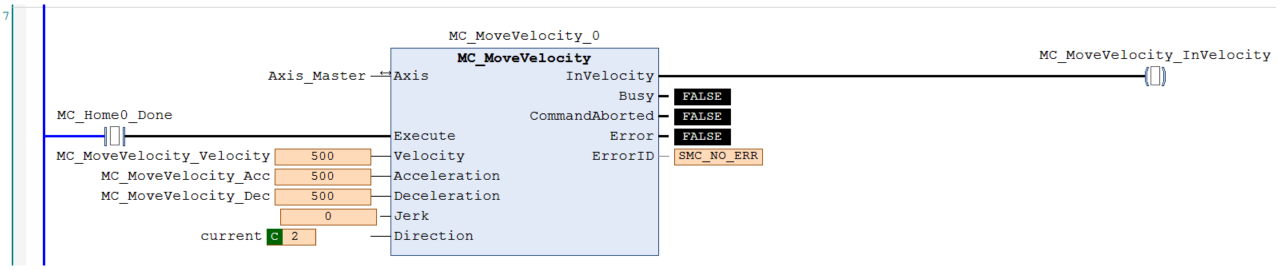
If normal, the master and slave axes are then set to the state of Servo On.



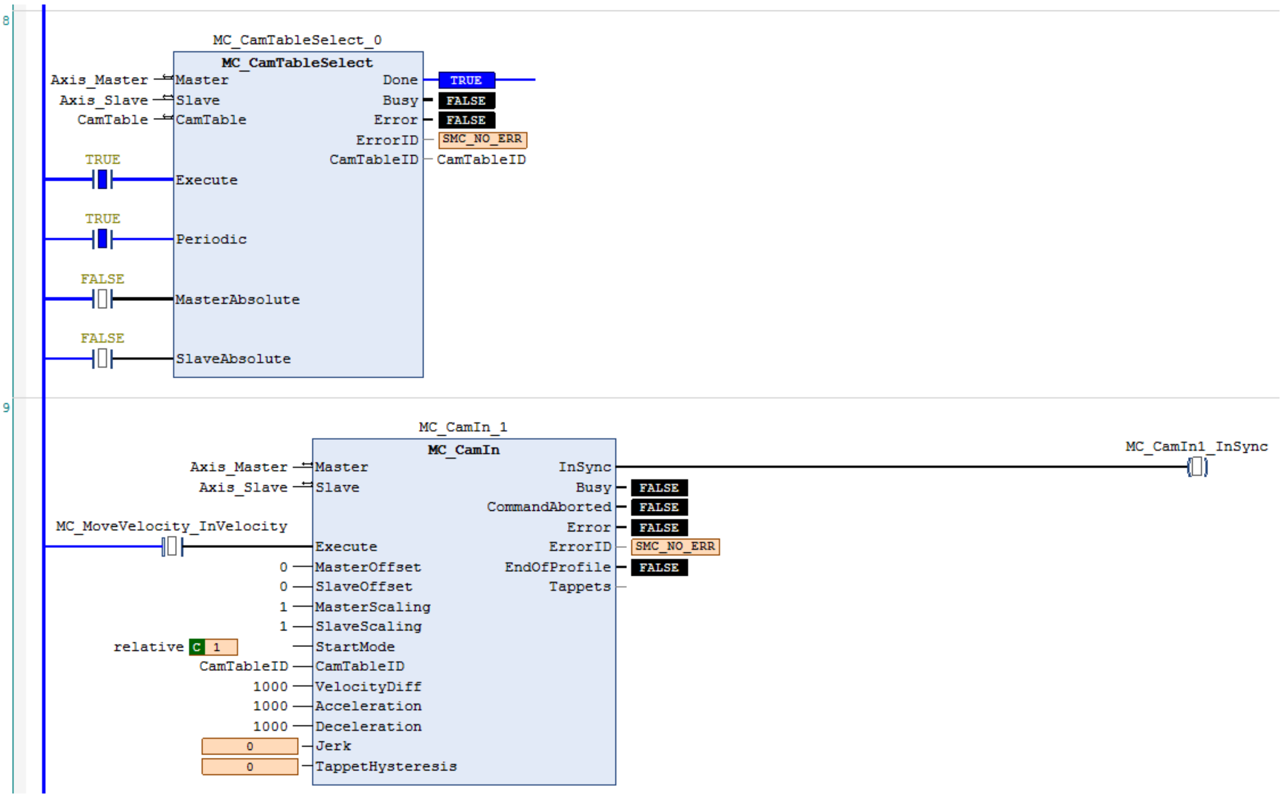
If the master and slave axes are in the Servo On state with unsure starting point, perform homing once.



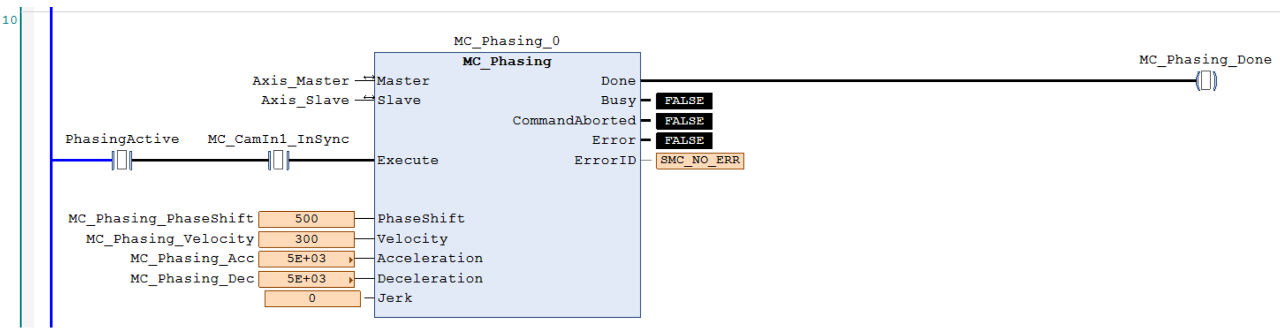
After homing of the master axis is completed, execute MC_MoveVelocity.



After the master axis attains the target velocity, set the respective cam table based on MC_CamTableSelect settings to execute cam motion.



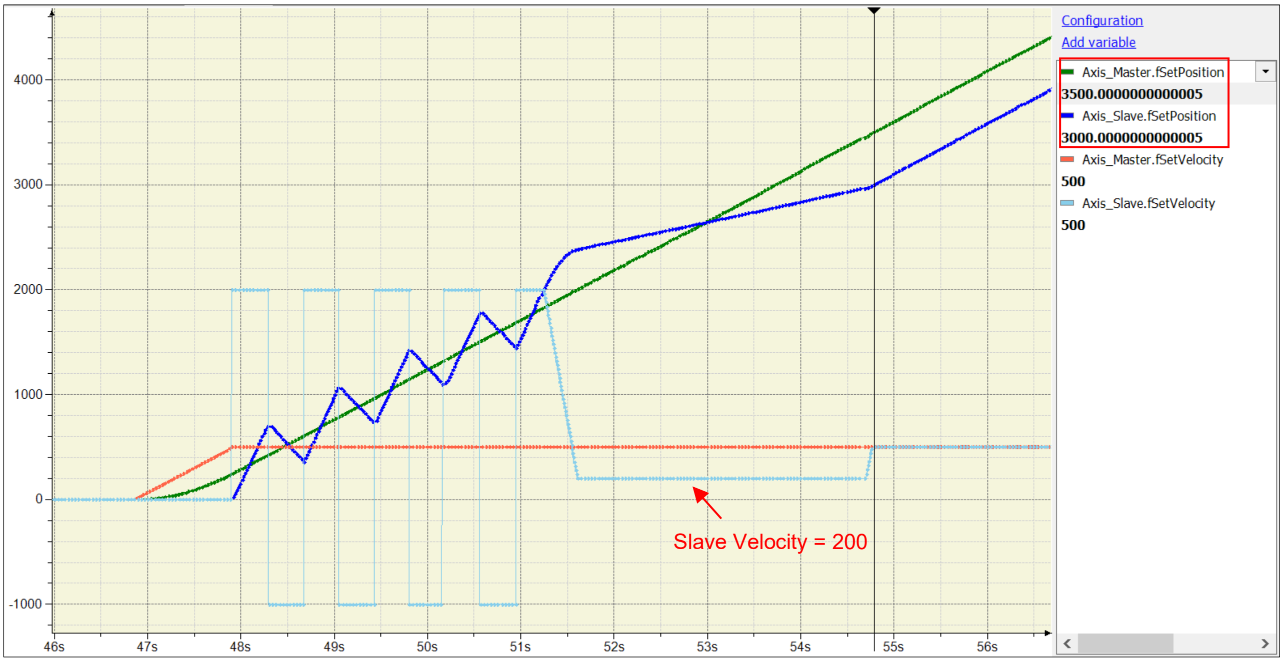
If PhasingActive is True and the cam is synchronizing, it starts phase offset between the master axis and the slave axis. Where the original master-slave axis relationship in cam is broken, the slave axis starts phase synchronization with the master axis according to the setting of MC_Phasing.



7

When performing phase offset of the master axis based on the settings above, the slave axis will synchronize with the phase after offset is done and the phase difference is fixed.

Take the cursor timing 3500-3000 in the fugure as example, the PhaseShift is 500 and the velocity of the slave axis is 200 (velocity of the master axis 500 minus velocity 300) when offset is performed.



- **ST language**

Set StartFlag to True, then it will check whether the communication of the master axis and the slave axis is normal or not.

```
IF StartFlag = TRUE THEN
    IF Axis_Master.bCommunication = TRUE THEN
        MC_Power0_Enable := TRUE;
        MC_Power0_RegulatorOn := TRUE;
        MC_Power0_DriveStart := TRUE;
    END_IF
    IF Axis_Slave.bCommunication = TRUE THEN
        MC_Power1_Enable := TRUE;
        MC_Power1_RegulatorOn := TRUE;
        MC_Power1_DriveStart := TRUE;
    END_IF
END_IF
```

If normal, the master and slave axes are then set to the state of Servo On.

```
MC_Power_0(
    Axis:= Axis_Master,
    Enable:= MC_Power0_Enable,
    bRegulatorOn:= MC_Power0_RegulatorOn,
    bDriveStart:= MC_Power0_DriveStart,
    Status=> MC_Power0_Status,
    bRegulatorRealState=> ,
    bDriveStartRealState=> ,
    Busy=> ,
```

```
        Error=> ,
        ErrorID=> );

MC_Power_1(
        Axis:= Axis_Slave,
        Enable:= MC_Power1_Enable,
        bRegulatorOn:= MC_Power1_RegulatorOn,
        bDriveStart:= MC_Power1_DriveStart,
        Status=> MC_Power1_Status,
        bRegulatorRealState=> ,
        bDriveStartRealState=> ,
        Busy=> ,
        Error=> ,
        ErrorID=> );
```

If the master and slave axes are in the Servo On state with unsure starting point, perform homing once.

```
IF MC_Power0_Status = TRUE THEN
        MC_Home0_Execute := TRUE;
END_IF

IF MC_Power1_Status = TRUE THEN
        MC_Home1_Execute := TRUE;
END_IF

MC_Home_0(
        Axis:= Axis_Master,
        Execute:= MC_Home0_Execute,
        Position:= 0,
        Done=> MC_Home0_Done,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

MC_Home_1(
        Axis:= Axis_Slave,
        Execute:= MC_Home1_Execute,
        Position:= 0,
        Done=> MC_Home1_Done,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

After homing of the master axis is completed, execute MC_MoveVelocity.

```
MC_MoveVelocity(
        Axis:= Axis_Master,
        Execute:= MC_Home0_Done,
        Velocity:= MC_MoveVelocity_Velocity,
```

7

```
        Acceleration:= MC_MoveVelocity_Acc,
        Deceleration:= MC_MoveVelocity_Dec,
        Jerk:= ,
        Direction:= current,
        InVelocity=> MC_MoveVelocity_InVelocity,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

After the master axis attains the target velocity, set the respective cam table based on MC_CamTableSelect settings to execute cam motion.

```
MC_CamTableSelect(
        Master:= Axis_Master,
        Slave:= Axis_Slave,
        CamTable:= CamTable,
        Execute:= TRUE,
        Periodic:= TRUE,
        MasterAbsolute:= FALSE,
        SlaveAbsolute:= FALSE,
        Done=> MC_CamTableSelect_Done,
        Busy=> ,
        Error=> ,
        ErrorID=> ,
        CamTableID=> CamTableID);

IF MC_MoveVelocity_InVelocity = TRUE THEN
        MC_CamIn_1(
        Master:= Axis_Master,
        Slave:= Axis_Slave,
        Execute:= TRUE,
        MasterOffset:= 0,
        SlaveOffset:= 0,
        MasterScaling:= 1,
        SlaveScaling:= 1,
        StartMode:= relative,
        CamTableID:= CamTableID,
        VelocityDiff:= 1000,
        Acceleration:= 1000,
        Deceleration:= 1000,
        Jerk:= ,
        TappetHysteresis:= ,
        InSync=> MC_CamIn1_Insync,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> ,
        EndOfProfile=> ,
        Tappets=> );
```

```
END_IF
```

If PhasingActive is True and the cam is synchronizing, it starts phase offset between the master axis and the slave axis. Where the original master-slave axis relationship in cam is broken, the slave axis starts phase synchronization with the master axis according to the setting of MC_Phasing.

```
IF (PhasingActive = TRUE) AND (MC_CamIn1_Insync = TRUE) THEN
    MC_Phasing_Execute := TRUE;
END_IF

MC_Phasing(
    Master:= Axis_Master,
    Slave:= Axis_Slave,
    Execute:= MC_Phasing_Execute,
    PhaseShift:= MC_Phasing_PhaseShift,
    Velocity:= MC_Phasing_Velocity,
    Acceleration:= MC_Phasing_Acc,
    Deceleration:= MC_Phasing_Dec,
    Jerk:= ,
    Done=> MC_Phasing_Done,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );
```

7.8.2.9 Changing Current Position in Movement

To change the current position of the axis in the coordinate system to the specified target position in movement, change the current feedback position to the specified target position. In this example, the interactive influence of the function blocks MC_MoveRelative and MC_SetPosition is explained.

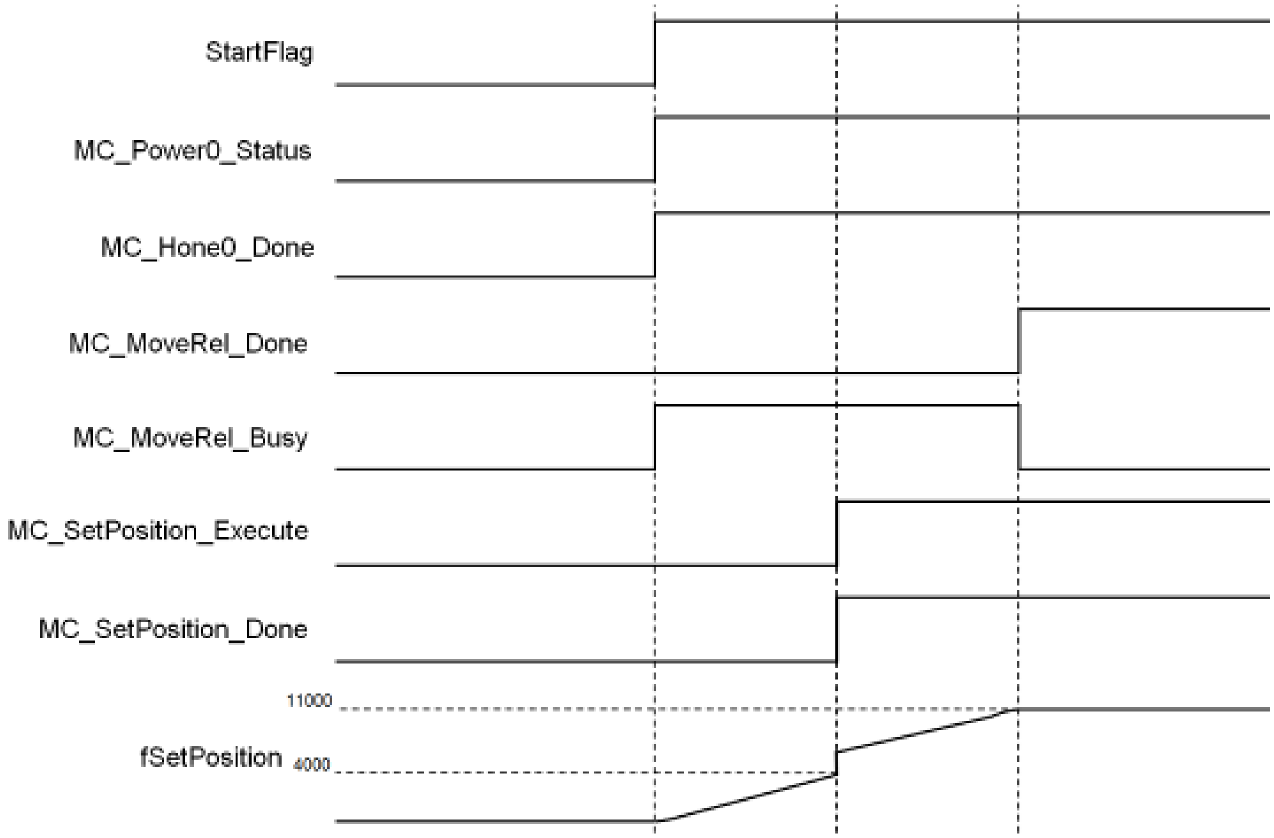
Two programming languages, ladder diagram (LD) and structured text (ST), are provided in the following example.

- **Main variables used in programming**

Name	Variable Type	Initial Value	Annotation
Axis_Virtual	AXIS_REF_VIRTUAL_SM3	-	Relevant variables of axis
StartFlag	BOOL	FALSE	If this variable is TRUE and axis communication is normal, Servo On will be activated with the subsequent actions being performed
MC_Power0_Status	BOOL	FALSE	Variables of MC_Power output Status; TRUE when Servo On
MC_Home0_Done	BOOL	FALSE	Variables of MC_Home output Done; TRUE when homing is completed
MC_MoveRel_Distance	LREAL	8000	Target relative position of MC_MoveRelative
MC_MoveRel_Done	BOOL	FALSE	Variables of MC_MoveRelative output Done; TRUE when relative positioning is complete
MC_MoveRel_Busy	BOOL	FALSE	Variable of MC_MoveRelative output Busy; TRUE when the instruction is triggered and executed
MC_SetPosition_Execute	BOOL	FALSE	If TRUE, MC_SetPosition is executed
MC_SetPosition_Position	LREAL	3000	The absolute position or relative distance changed by MC_SetPosition
MC_SetPosition_Mode	BOOL	TRUE	MC_SetPosition is to set the axis position to be the absolute or relative position
MC_SetPosition_Done	BOOL	FALSE	Variables of MC_SetPosition output Done; TRUE when position is modified

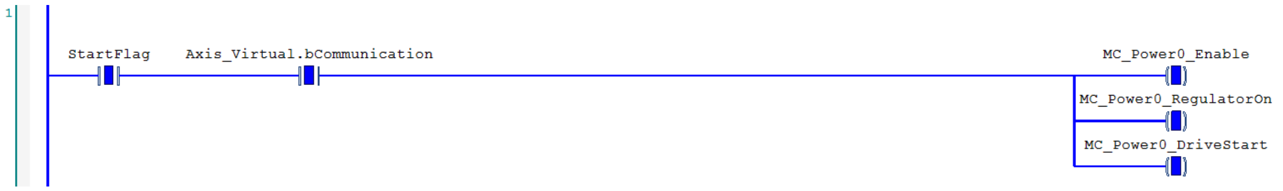
7

- **Timing diagram**

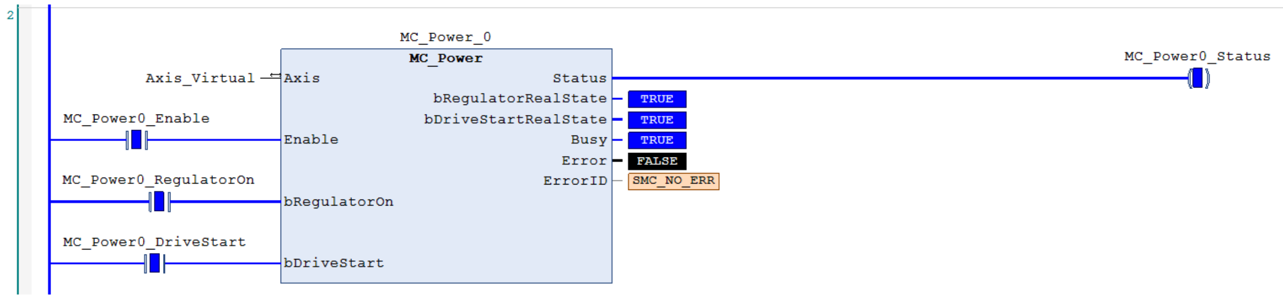


- **LD language**

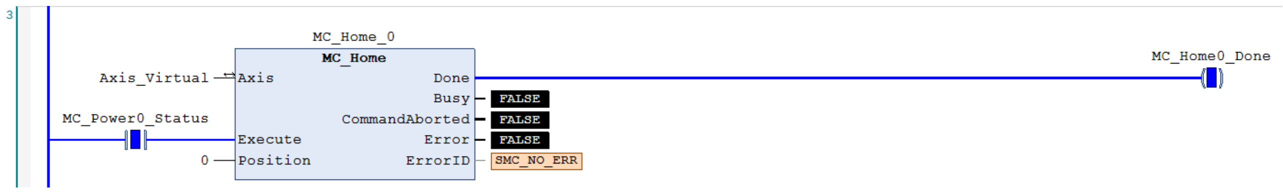
Set StartFlag to True, then it will check whether the axis communication is normal or not.



If normal, set the axis to the state of Servo On.

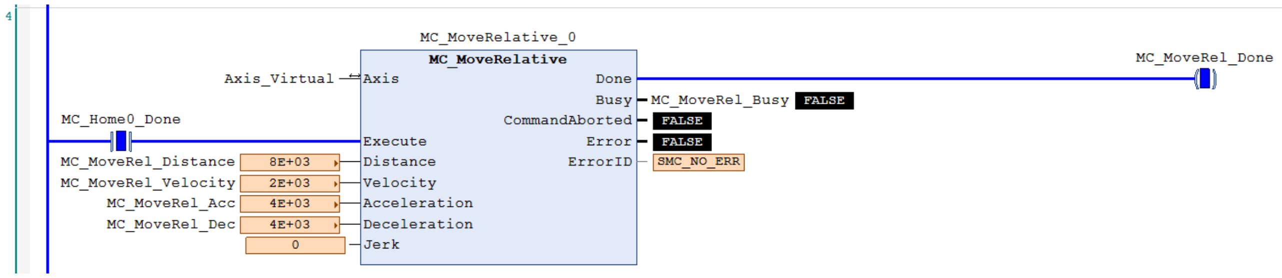


If it is in the state of Servo On with unsure starting point, perform homing once.

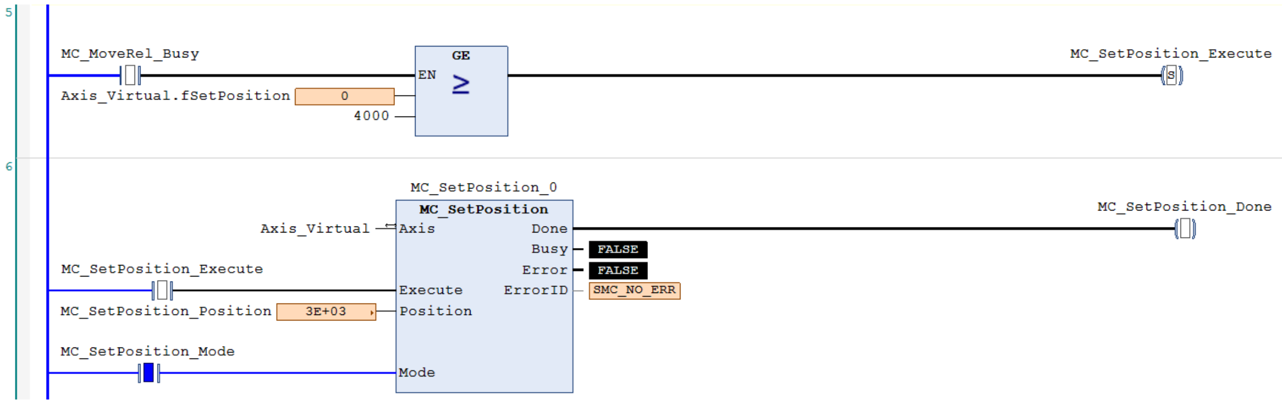


After homing is completed, execute MC_MoveRelative.

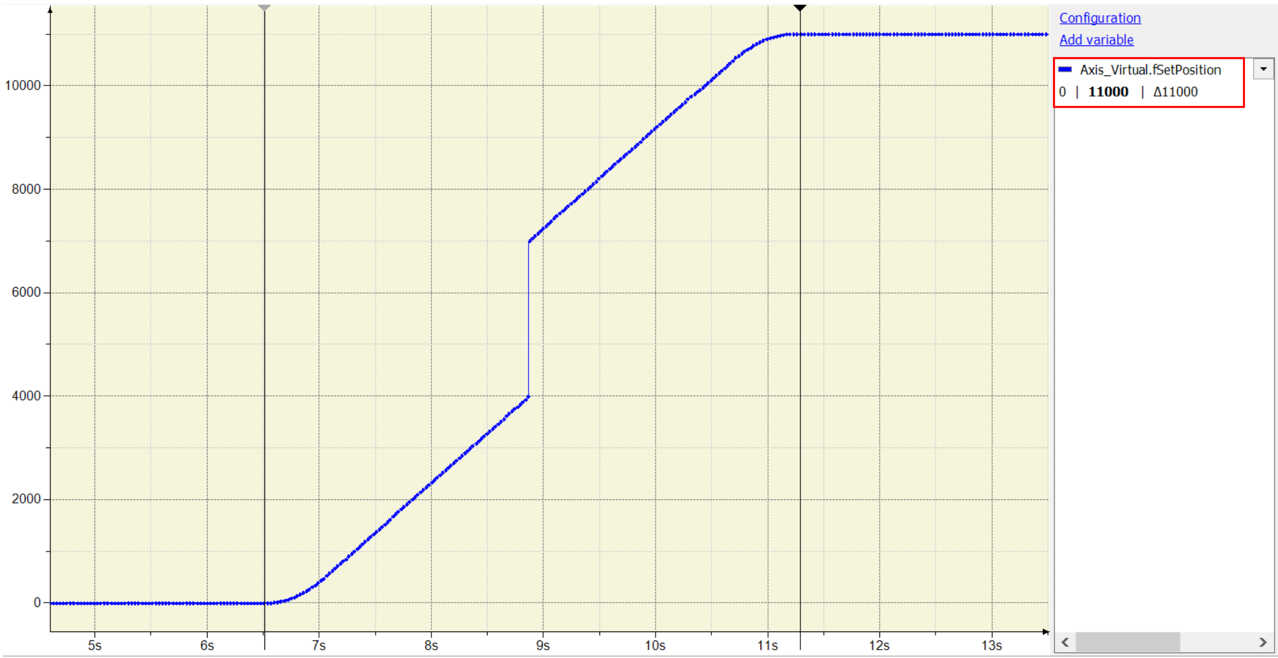
The target position for relative displacement = 8000.



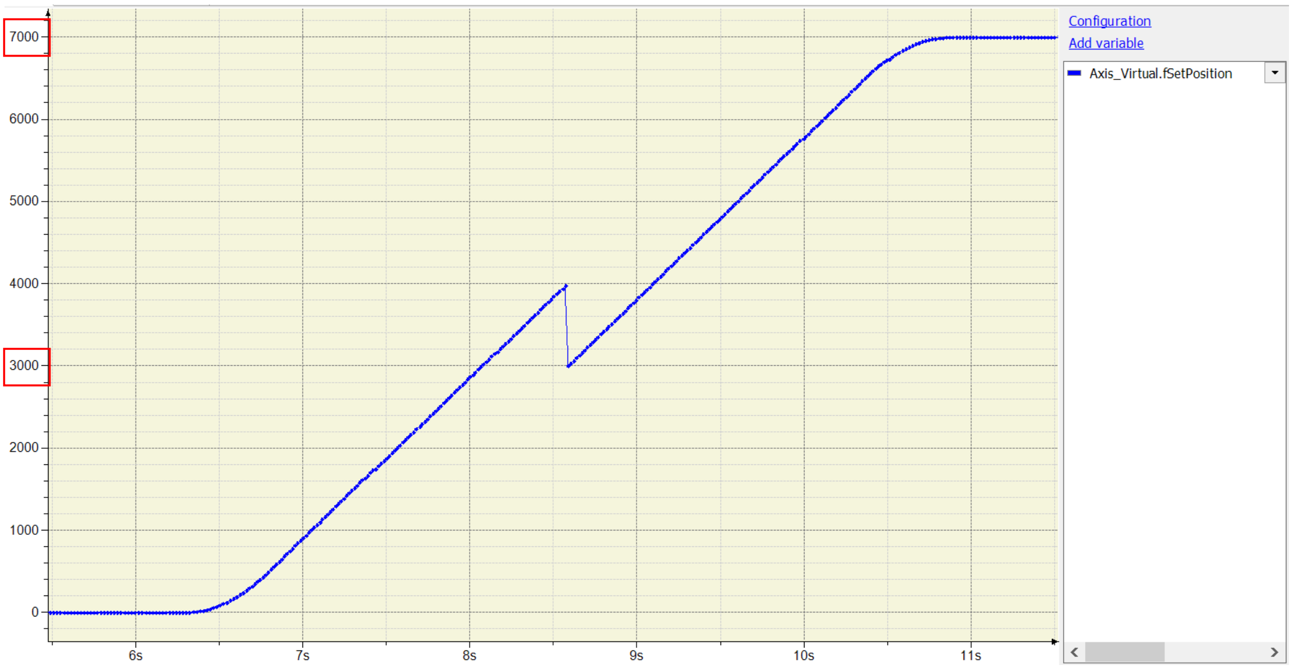
When the current position of the axis passes 4000, execute MC_SetPosition (Mode = Relative, Distance = 3000) and change the current position to be the specified target position.



According to the above settings, the current position of the axis in the coordinate system is changed to the relative target position in movement control, and the final position reaches 11000 as changed (11000 = 4000 + 3000 + (8000 - 4000)). The execution has no influence on the displacement of the physical mechanism in MC_MoveRelative operation, and the moving distance is the same as the set originally, which is 8000 (8000 = (4000 - 0) + (11000 - 7000)).



The difference between the picture below and the picture above is that the Mode of MC_SetPoition is changed to Absolute (Position = 3000), and the current position of the axis in the coordinate system is changed to the absolute target position, and the final position reaches 7000 after the change (7000 = 3000 + (8000 – 4000)), the execution of the change command position will not affect the displacement of the physical movement mechanism in the MC_MoveRelative operation, and the moving distance is the same as the original setting of 8000 (8000 = (4000 - 0) + (7000 - 3000)).



- **ST language**

Set StartFlag to True, then it will check whether the axis communication is normal or not.

```
IF StartFlag = TRUE THEN
    IF Axis_Virtual.bCommunication = TRUE THEN
        MC_Power0_Enable := TRUE;
        MC_Power0_RegulatorOn := TRUE;
        MC_Power0_DriveStart := TRUE;
    END_IF
END_IF
```

If normal, set the axis to the state of Servo On.

```
MC_Power_0(
    Axis:= Axis_Virtual,
    Enable:= MC_Power0_Enable,
    bRegulatorOn:= MC_Power0_RegulatorOn,
    bDriveStart:= MC_Power0_DriveStart,
    Status=> MC_Power0_Status,
    bRegulatorRealState=> ,
    bDriveStartRealState=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

If it is in the state of Servo On with unsure starting point, perform homing once.

7

```
IF MC_Power0_Status = TRUE THEN
    MC_Home0_Execute := TRUE;
END_IF

MC_Home_0(
    Axis:= Axis_Virtual,
    Execute:= MC_Home0_Execute,
    Position:= 0,
    Done=> MC_Home0_Done,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );
```

After homing is completed, execute MC_MoveRelative. The target position for relative displacement = 8000.

```
MC_MoveRelative(
    Axis:= Axis_Virtual,
    Execute:= MC_Home0_Done,
    Distance:= MC_MoveRel_Distance,
    Velocity:= MC_MoveRel_Velocity,
    Acceleration:= MC_MoveRel_Acc,
    Deceleration:= MC_MoveRel_Dec,
    Jerk:= ,
    Done=> MC_MoveRel_Done,
    Busy=> MC_MoveRel_Busy,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );
```

When the current position of the axis passes 4000, execute MC_SetPosition (Mode = Relative, Distance = 3000) and change the current position to be the specified target position.


```
IF (MC_MoveRel_Busy = TRUE) AND (Axis_Virtual.fSetPosition >= 4000) THEN
    MC_SetPosition_Execute := TRUE;
END_IF

MC_SetPosition(
    Axis:= Axis_Virtual,
    Execute:= MC_SetPosition_Execute,
    Position:= MC_SetPosition_Position,
    Mode:= MC_SetPosition_Mode,
    Done=> MC_SetPosition_Done,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

7.8.2.10 Performing Superimposition during Gear Engagement

Perform MC_MoveSuperImposed on the designated slave axis while the master and slave axes engage with each other. The final target position of the slave axis will be the relative displacement to the master axis according to the gear ratio plus the specific distance superimposed during movement.

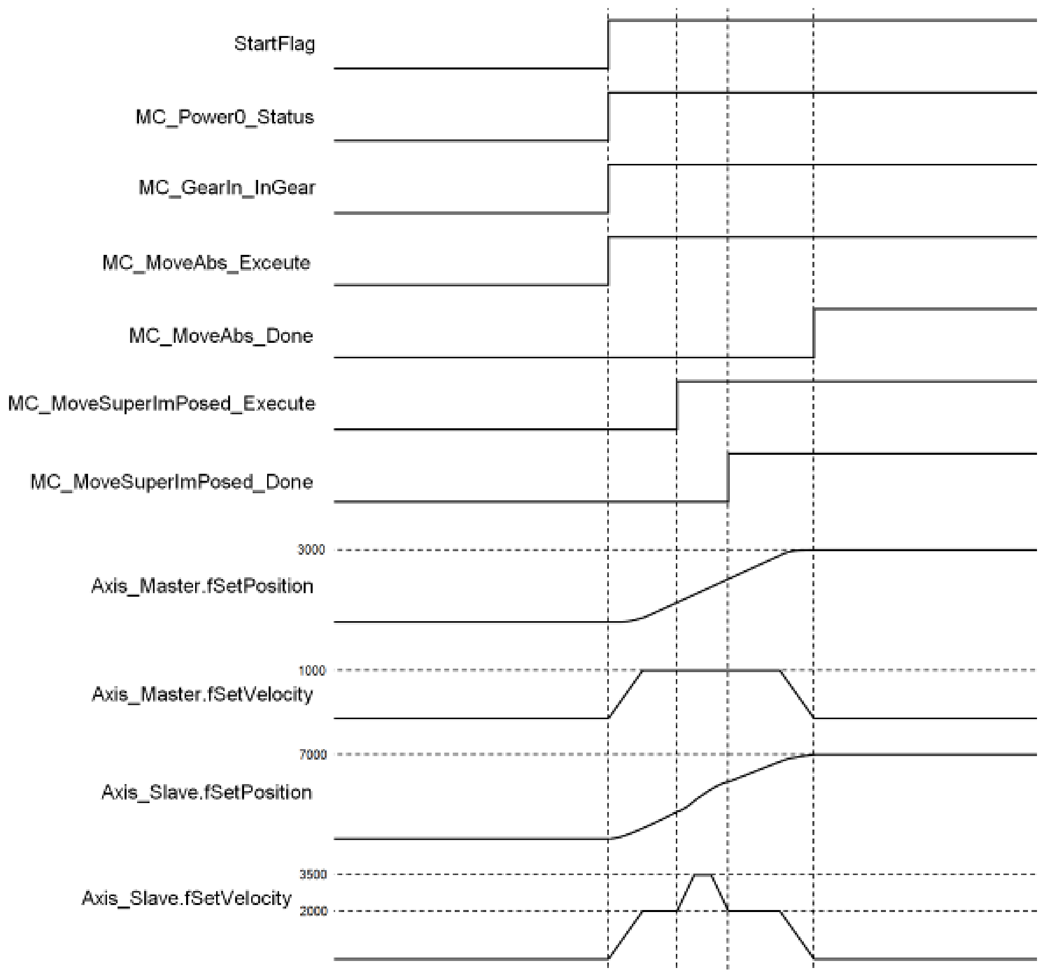
Two programming languages, ladder diagram (LD) and structured text (ST), are provided in the following example.

- **Main variables used in programming**

Name	Variable Type	Initial Value	Annotation
Axis_Master	AXIS_REF_ VIRTUAL_SM3	-	Master-related axis variables
Axis_Slave	AXIS_REF_ VIRTUAL_SM3	-	Slave-related axis variables
StartFlag	BOOL	FALSE	If this variable is TRUE and axis communication is normal, Servo On will be activated with the subsequent actions being performed
MC_Power0_Status	BOOL	FALSE	Variables of MC_Power output Status for master; TRUE when Servo On
MC_Power1_Status	BOOL	FALSE	Variables of MC_Power output Status for slave; TRUE when Servo On
MC_Home0_Done	BOOL	FALSE	Variables of MC_Home output Done for master; TRUE when homing is completed
MC_Home1_Done	BOOL	FALSE	Variables of MC_Home output Done for slave; TRUE when homing is completed
MC_GearIn_InGear	BOOL	FALSE	Variables of MC_GearIn output InGear; TRUE when the engagement is completed
MC_GearIn_RatioNumerator	DINT	2	Numerator of the gear ratio between the master and slave axes
MC_GearIn_RatioDenominator	UDINT	1	Denominator of the gear ratio between the master and slave axes
MC_MoveAbs_Execute	BOOL	FALSE	MC_MoveAbsolute is executed when this variable is TRUE,
MC_MoveAbs_Position	LREAL	3000	Specify the absolute target position of the master axis
MC_MoveAbs_Velocity	LREAL	1000	Specify the target velocity of the master axis

Name	Variable Type	Initial Value	Annotation
MC_MoveAbs_Done	BOOL	FALSE	Variables of MC_MoveAbsolute output Done for master; TRUE when absolute positioning is completed.
MC_MoveAbs_Busy	BOOL	FALSE	Variables of MC_MoveAbsolute output Busy for master; TRUE when the instruction is executed
MC_MoveSuperImposed_Execute	BOOL	FALSE	MC_MoveSuperImposed is executed when this variable is TRUE
MC_MoveSuperImposed_Done	BOOL	FALSE	Variables of MC_MoveSuperImposed output Done for slave; TRUE when superimposition is completed.
MC_MoveSuperImposed_Distance	LREAL	1000	Specify the superimposed displacement of the slave axis
MC_MoveSuperImposed_VelocityDiff	LREAL	1500	Specify the relative velocity of the slave axis during superimposition compared to the master axis

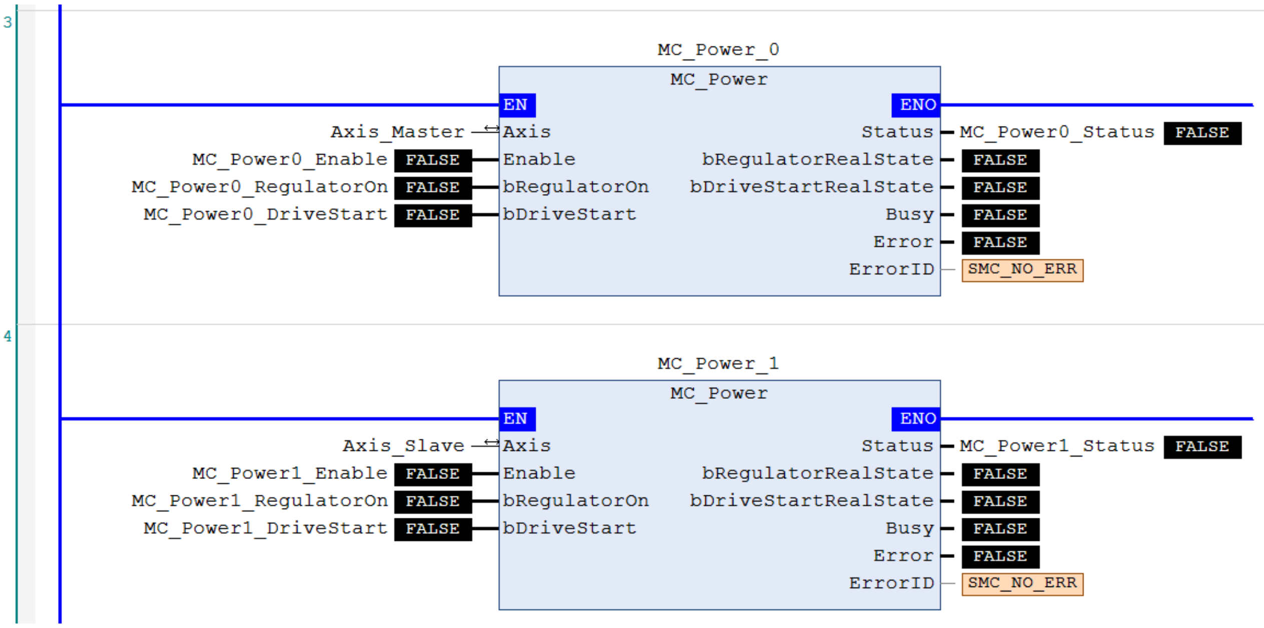
- **Timing diagram**



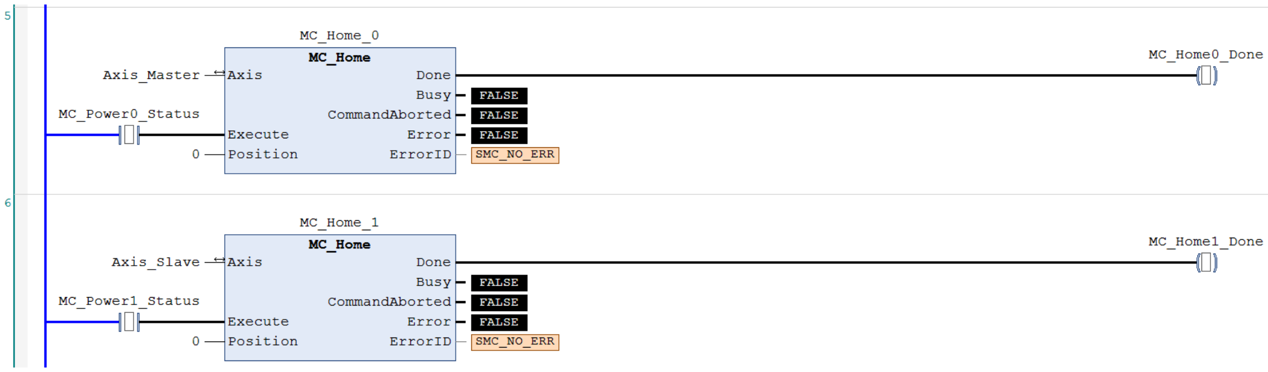
- **LD language**

Set StartFlag to True, then it will check whether the communication of the master axis and the slave axis is normal or not.

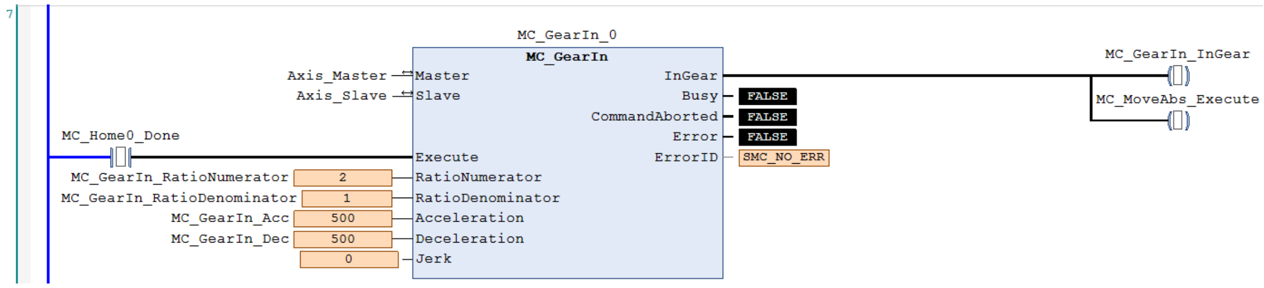
If normal, the master and slave axes are then set to the state of Servo On.



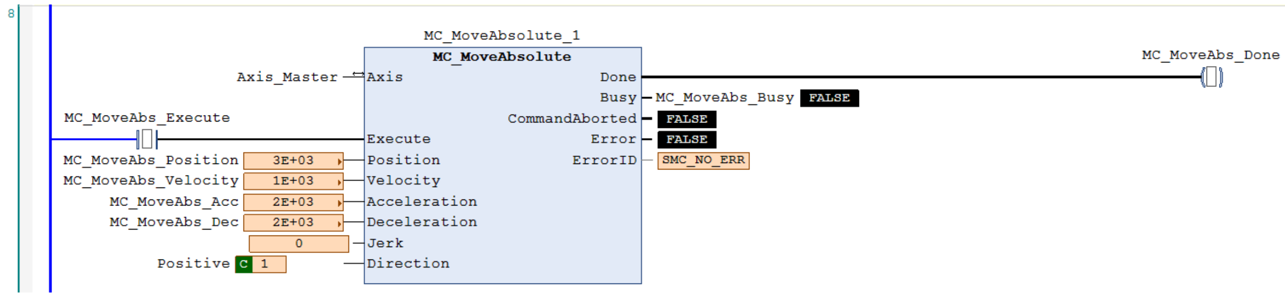
If the axes are in the state of Servo On with unsure starting point, perform homing once.



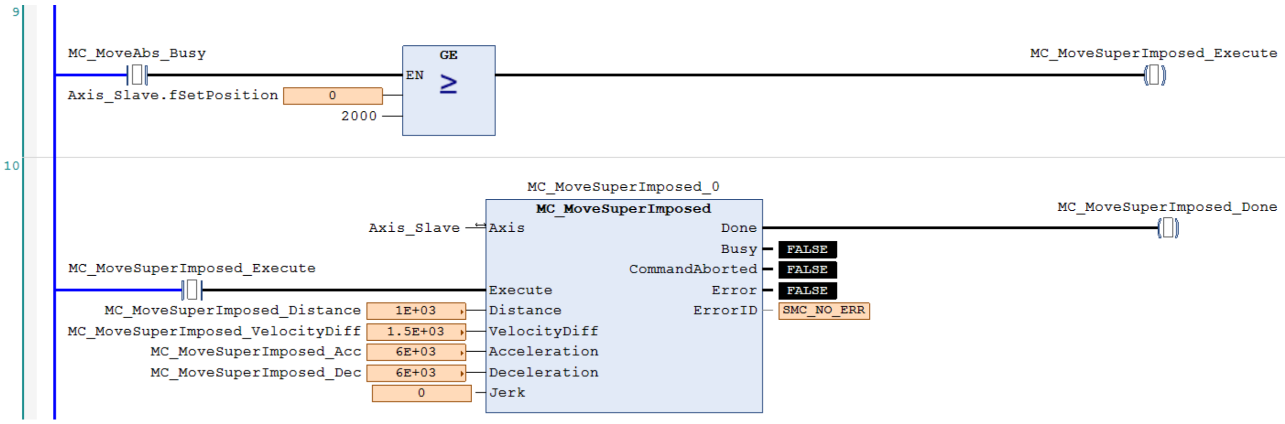
After homing of the master axis is completed, execute MC_GearIn to establish gear synchronization between the axes.



Right after the engagement InGear is completed, execute MC_MoveAbsolute on the master axis.



As for the slave axis, MC_MoveSuperImposed is executed when it moves in the manner of engagement to the preset triggering position 2000, in which the slave axis will move at a distance of the preset target position derived from the original gear ration plus the superimposed displacement.



Based on the settings above, the slave axis moves at a velocity set when superimposition is implemented on it and to the final target position which is the relative displacement to the master axis according to the gear ratio plus the specific distance superimposed during movement.

In the figure below, the moving distance of the master axis is 3000. As for the slave axis, its original target position is 6000 as derived from the gear ratio of 1:2; however, the target position increases to 7000 because an extra distance of 1000 is superimposed on the slave axis. During engagement, the master axis moving at the velocity 1000 and the slave axis at 2000. Yet, the velocity of the slave axis becomes 3500 (original velocity 2000 plus VelocityDiff 1500) during superimposition.

- **ST language**

Set StartFlag to True, then it will check whether the communication of the master axis and the slave axis is normal or not.

```
IF StartFlag = TRUE THEN
    IF Axis_Master.bCommunication = TRUE THEN
        MC_Power0_Enable := TRUE;
        MC_Power0_RegulatorOn := TRUE;
        MC_Power0_DriveStart := TRUE;
    END_IF
    IF Axis_Slave.bCommunication = TRUE THEN
        MC_Power1_Enable := TRUE;
        MC_Power1_RegulatorOn := TRUE;
        MC_Power1_DriveStart := TRUE;
    END_IF
END_IF
```

If normal, the master and slave axes are then set to the state of Servo On.

```
MC_Power_0(
    Axis:= Axis_Master,
    Enable:= MC_Power0_Enable,
    bRegulatorOn:= MC_Power0_RegulatorOn,
    bDriveStart:= MC_Power0_DriveStart,
    Status=> MC_Power0_Status,
    bRegulatorRealState=> ,
    bDriveStartRealState=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );

MC_Power_1(
```

7

```
    Axis:= Axis_Slave,
    Enable:= MC_Power1_Enable,
    bRegulatorOn:= MC_Power1_RegulatorOn,
    bDriveStart:= MC_Power1_DriveStart,
    Status=> MC_Power1_Status,
    bRegulatorRealState=> ,
    bDriveStartRealState=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

If the axes are in the state of Servo On with unsure starting point, perform homing once.

```
IF MC_Power0_Status = TRUE THEN
    MC_Home0_Execute := TRUE;
END_IF

IF MC_Power1_Status = TRUE THEN
    MC_Home1_Execute := TRUE;
END_IF

MC_Home_0(
    Axis:= Axis_Master,
    Execute:= MC_Home0_Execute,
    Position:= 0,
    Done=> MC_Home0_Done,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );

MC_Home_1(
    Axis:= Axis_Slave,
    Execute:= MC_Home1_Execute,
    Position:= 0,
    Done=> MC_Home1_Done,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );
```

After homing of the master axis is completed, execute MC_GearIn to establish gear synchronization between the axes.

```
MC_GearIn(
    Master:= Axis_Master,
    Slave:= Axis_Slave,
    Execute:= MC_Home0_Done,
    RatioNumerator:= MC_GearIn_RatioNumerator,
    RatioDenominator:= MC_GearIn_RatioDenominator,
    Acceleration:= MC_GearIn_Acc,
    Deceleration:= MC_GearIn_Dec,
```

```
        Jerk:= ,
        InGear=> MC_GearIn_InGear,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );

IF MC_GearIn_InGear = TRUE THEN
        MC_MoveAbs_Execute := TRUE;
END_IF
```

Right after the engagement InGear is completed, execute MC_MoveAbsolute on the master axis.

```
MC_MoveAbsolute(
        Axis:= Axis_Master,
        Execute:= MC_MoveAbs_Execute,
        Position:= MC_MoveAbs_Position,
        Velocity:= MC_MoveAbs_Velocity,
        Acceleration:= MC_MoveAbs_Acc,
        Deceleration:= MC_MoveAbs_Dec,
        Jerk:= ,
        Direction:= Positive,
        Done=> MC_MoveAbs_Done,
        Busy=> MC_MoveAbs_Busy,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

As for the slave axis, MC_MoveSuperImposed is executed when it moves in the manner of engagement to the preset triggering position 2000, in which the slave axis will move at a distance of superimposing the specified displacement on the preset target position of the original gear ratio.

```
IF MC_MoveAbs_Busy = TRUE THEN
        IF Axis_Slave.fSetPosition >= 2000 THEN
                MC_MoveSuperImposed_Execute := TRUE;
        END_IF
END_IF

MC_MoveSuperImposed(
        Axis:= Axis_Slave,
        Execute:= MC_MoveSuperImposed_Execute,
        Distance:= MC_MoveSuperImposed_Distance,
        VelocityDiff:= MC_MoveSuperImposed_VelocityDiff,
        Acceleration:= MC_MoveSuperImposed_Acc,
        Deceleration:= MC_MoveSuperImposed_Dec,
        Jerk:= ,
        Done=> MC_MoveSuperImposed_Done,
        Busy=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> );
```

MEMO

7

8

Chapter 8 OPC UA Server

Table of Contents

8.1 OPC UA Server

The standard installation of DIADesigner-AX includes an OPC UA server. You can use it to access the variable interface of the controller via a client. The OPC UA server communicates with connected OPC UA clients over a separate TCP connection. Therefore, the connections should be examined again separately for security purpose. Currently the OPC UA server can be safeguarded by using encrypted communication to protect the client and OPC UA user management. Details about the settings can be found in the following sections.

Below are features supported by the OPC UA server:

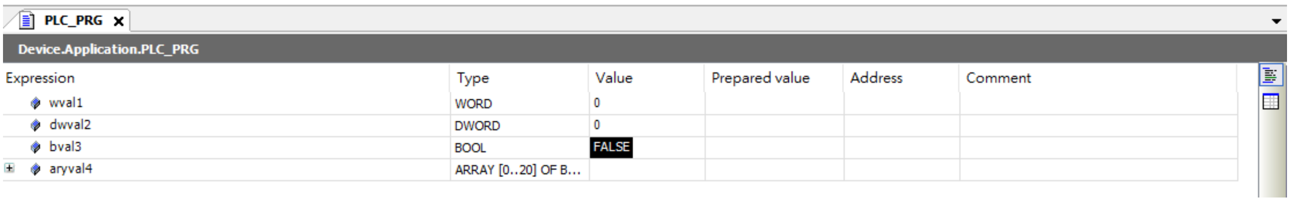
- Browse through data types and variables
- Standard read / write services
- Notification of value changes (focus on subscription and monitored items)
- Encrypted communication according to OPC UA standard (profile: Basic256SHA256)

8.1.1 Creating a Project for OPC UA Access

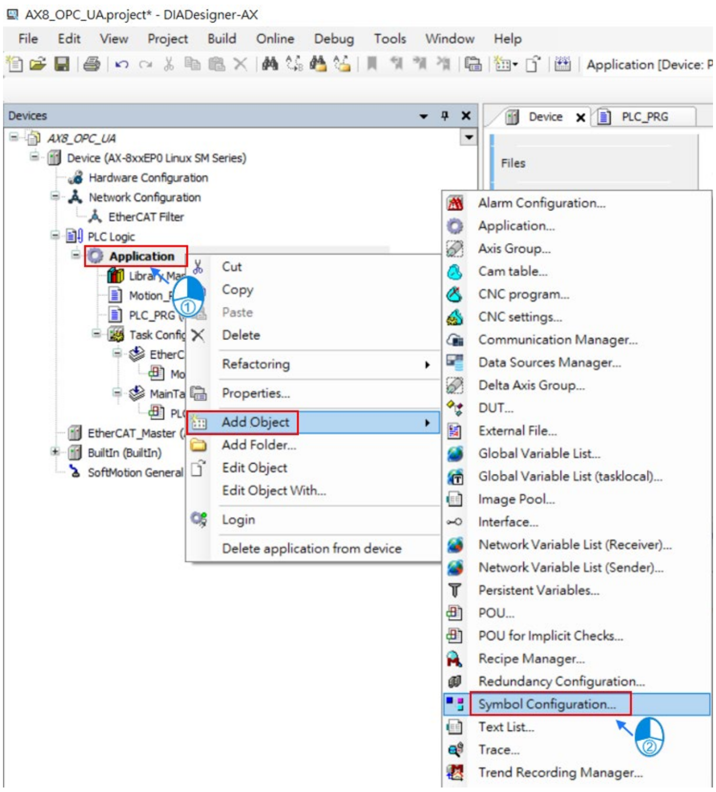
Before using OPC UA server, the first thing is creating a project for OPC UA access. Follow the steps below:

1. Create a new DIADesigner-AX project.

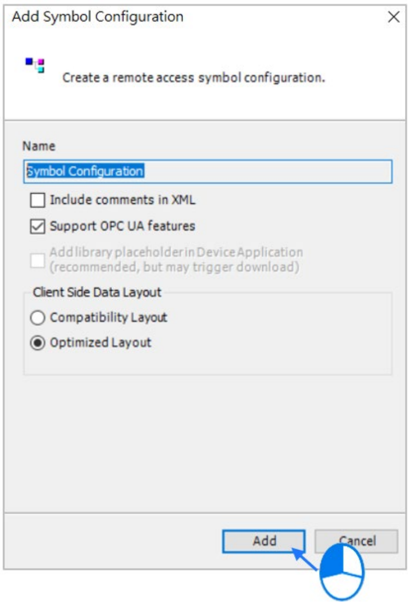
2. Declare some variables of different types in **PLC_PRG** program.



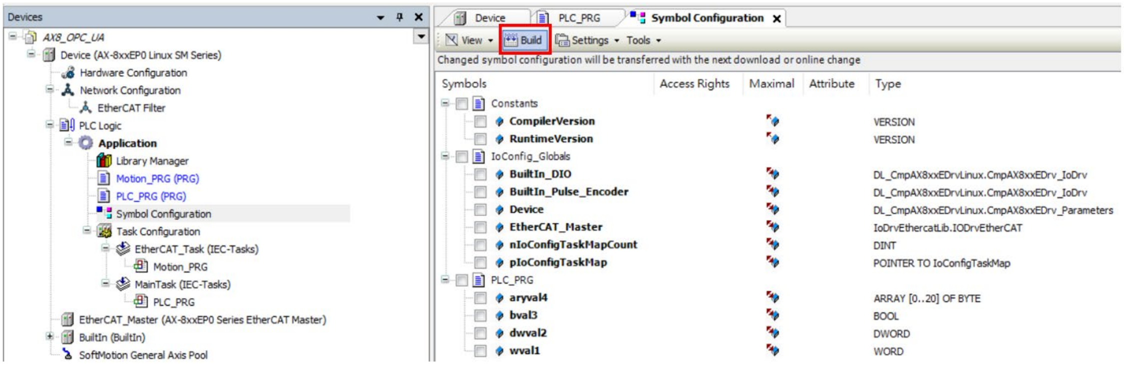
3. Go to **Application** > **Add Object** > **Symbol Configuration** to add a Symbol Configuation object.



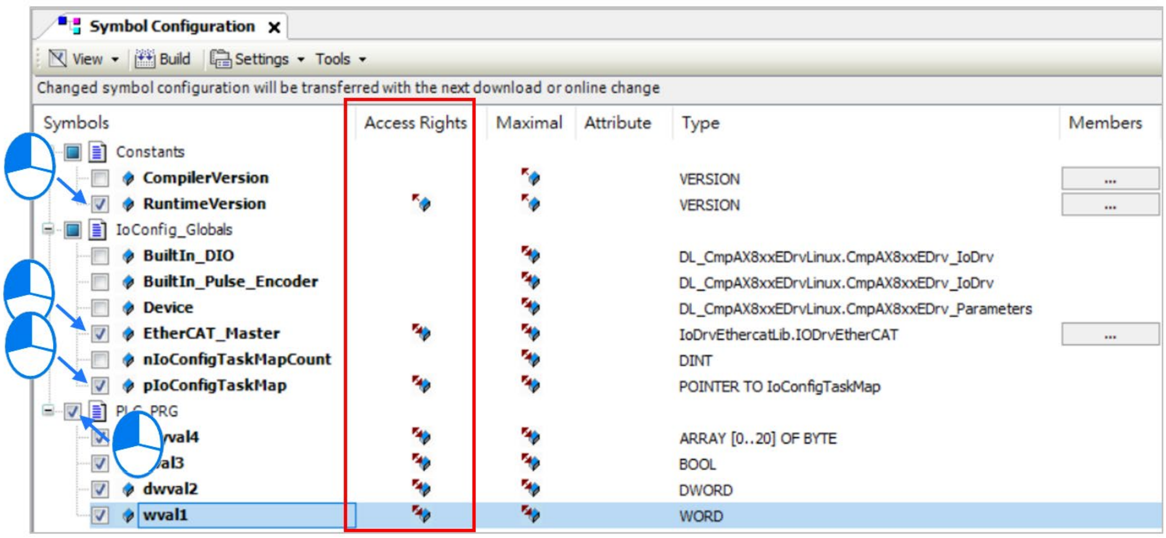
4. Select **Support OPC UA features** and then click **Add** on the setting page of **Add Symbol Configuration**.



5. The tab **Symbol Configuration** appears in the setting window with a warning message. Click **Build** then the projects and variables are displayed in a tree structure. The warning message disappears after the click.



6. Select the variables whereof configuration and monitoring by OPC UA clients are allowed. Specify the access rights of each variable and then click **Build** again to complete the setting.



7. Lastly, download the project to the AX-8 Series PLC.

8

8.2 Connection Settings

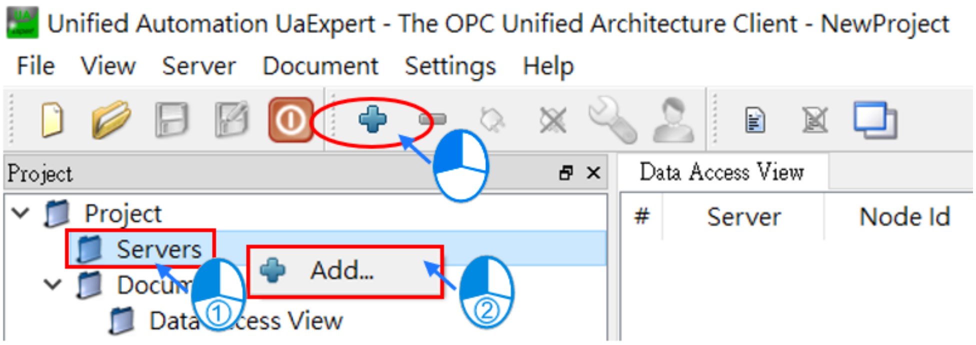
8.2.1 Setting up a Connection with UaExpert

Connect the client to the OPC UA server using unencrypted connection. The following illustrates how to set up the connection between the OPC UA server and the client, **UaExpert**. The steps can also be applied to other OPC UA clients.

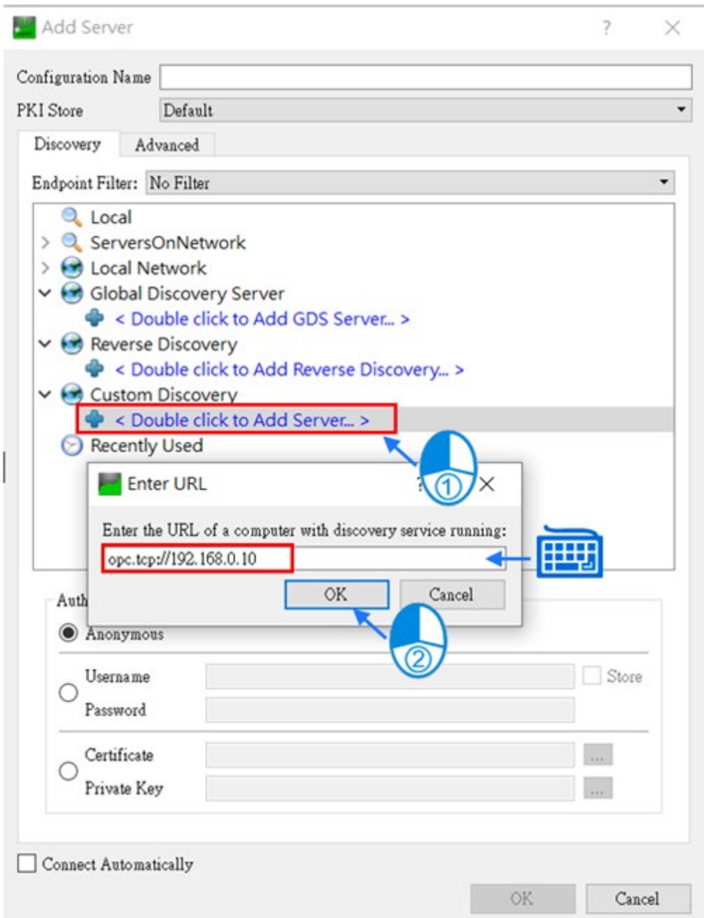
The OPC UA client, UaExpert, is freely accessible here:
https://www.unified-automation.com/downloads/opc-ua-clients.html.

1. Double-click to start **UaExpert**.

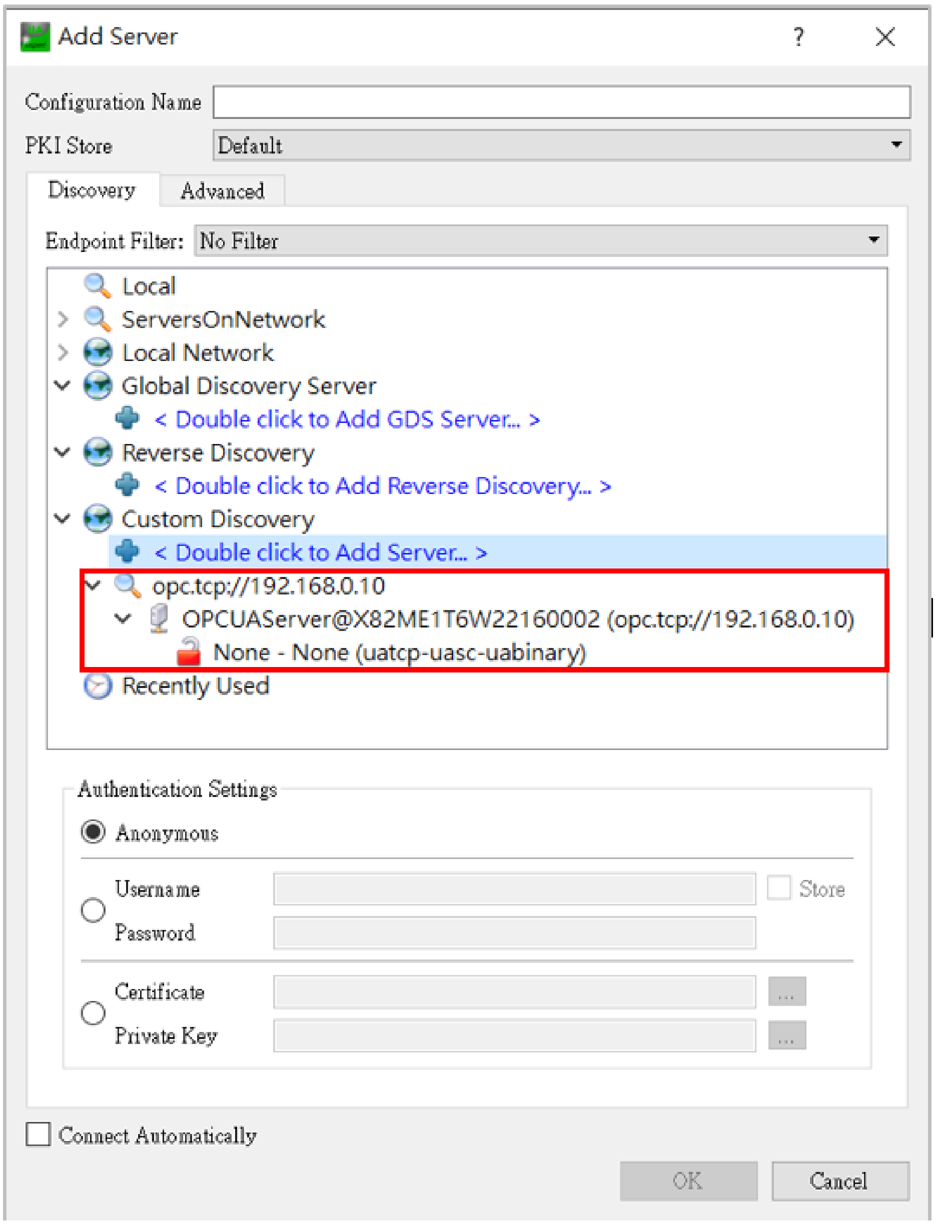
2. Right-click **Servers** and then click **Add** to open the **Add Server** window.



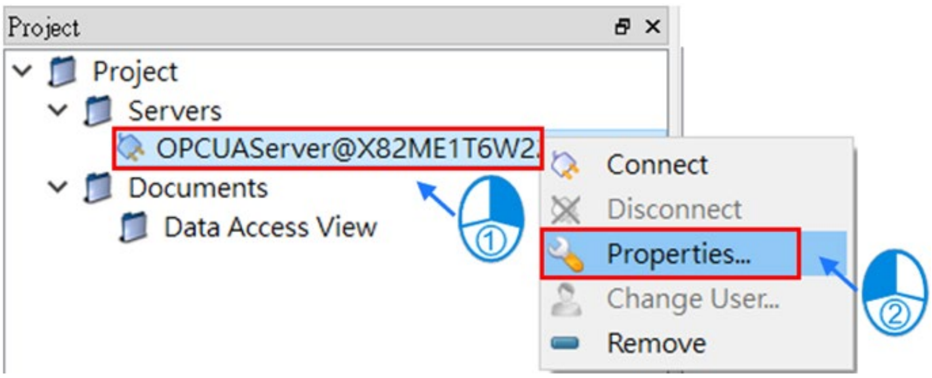
3. Go to **Custom Discovery** > **<Double click to Add Server…>** > **Enter URL**: enter **opc.tcp://192.168.0.10** and click **OK**.



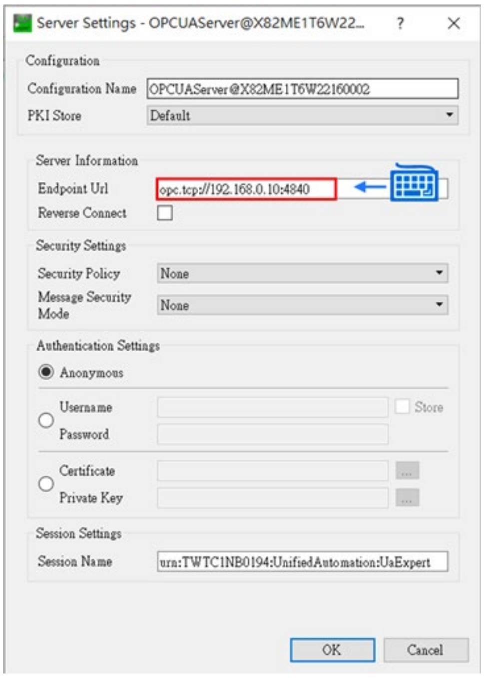
4. The window below pops up right after. You can find **AX-816E** under **opc.tcp://192. 168.0.10.** Click **OK** to close the window. As mentioned, we are using the unencrypted connection so only the node **None-None** is detected.



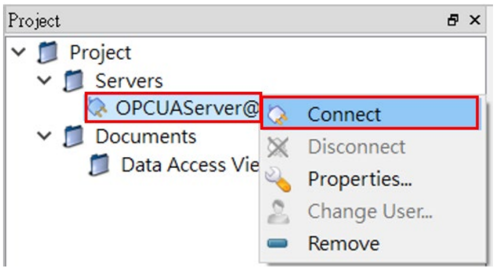
5. To edit the server properties, go back to the starting page. Expand the option under **Project**. Then right-click **OPCUAServer@X82ME1Txxxx** and select **Properties…** to open the **Server Settings** window.



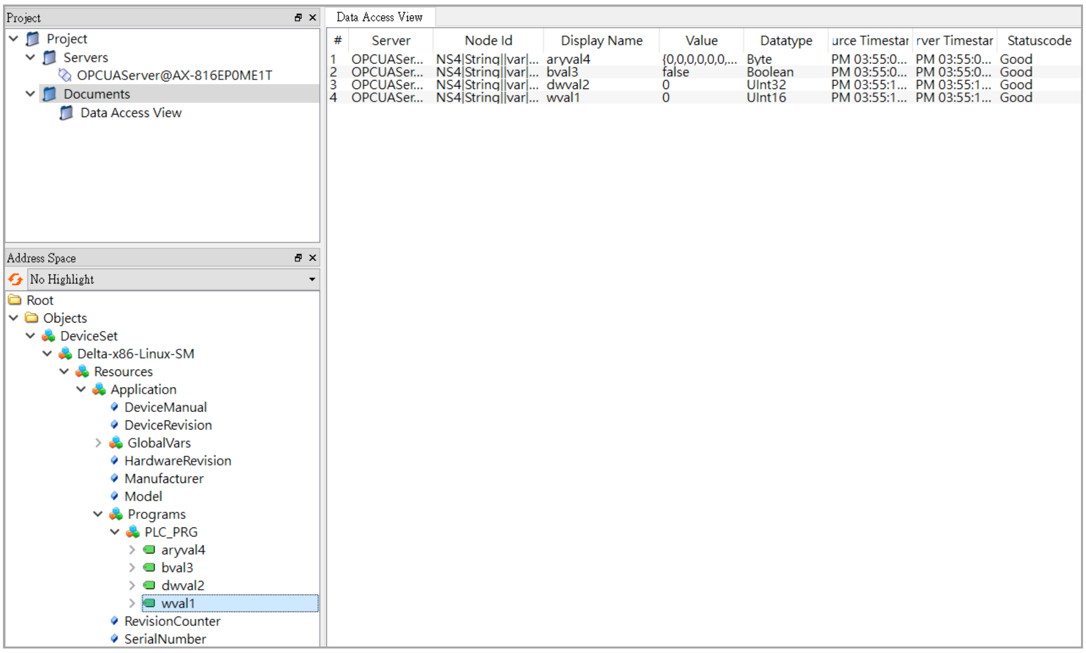
6. Rewrite **Endpoint Url** as **opc.tcp://192.168.0.10:4840**. Click **OK** to close the window.



7. Right-click **OPCUAServer@X82ME1Txxxx** and select **Connect** on the context menu.



8. After the connection is built, you are able to change the variables via OPC UA client. Drag the variable you would like to modify from **Address Space** to **Data Access View**. Then double-click the variables in the column **Value** to edit.



8

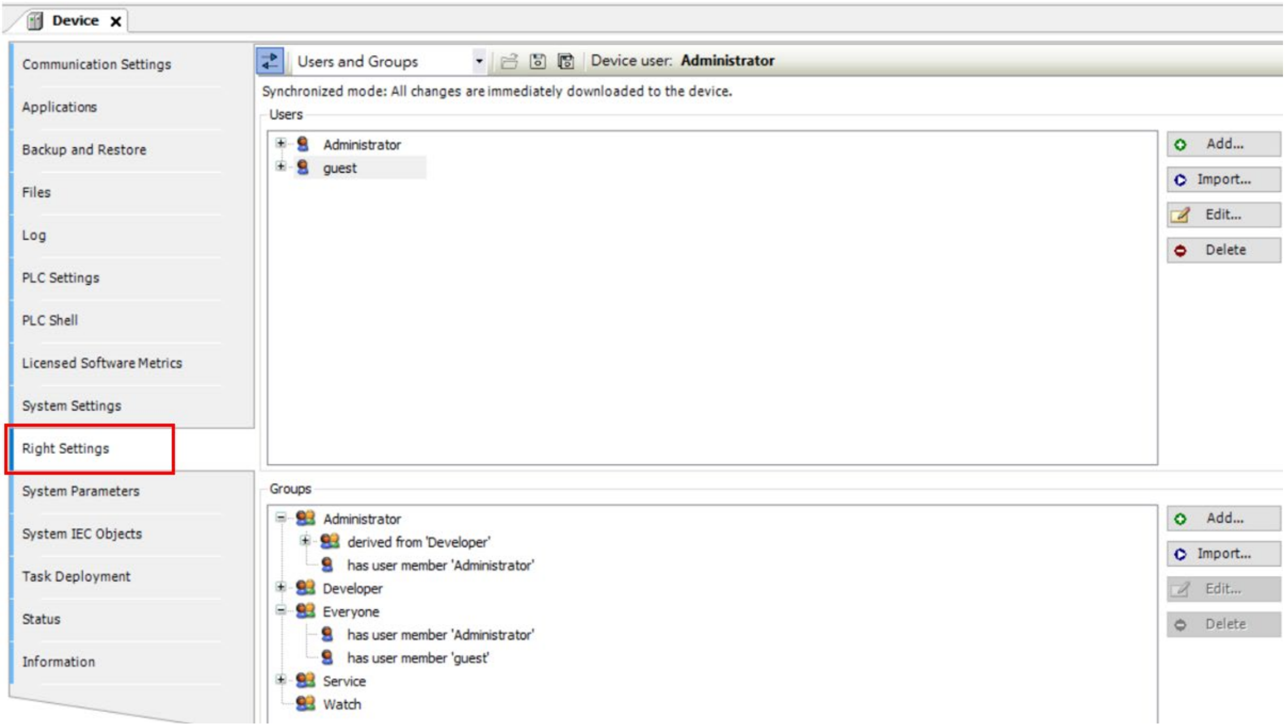
8.3 Setting up an Encrypted Connection

To have a secure and encrypted connection, it is recommended to follow the instructions below to create certificates for OPC UA server and OPC UA client.

8.3.1 Setting up User Account and Password

Account and password setup for OPC UA server is the same as the way for controller. Refer to **section 4.2.1.8** of this manual for more information.

Below is an example of setting up a new account as guest. The default account is Administrator. And here you can see two accounts in the image.

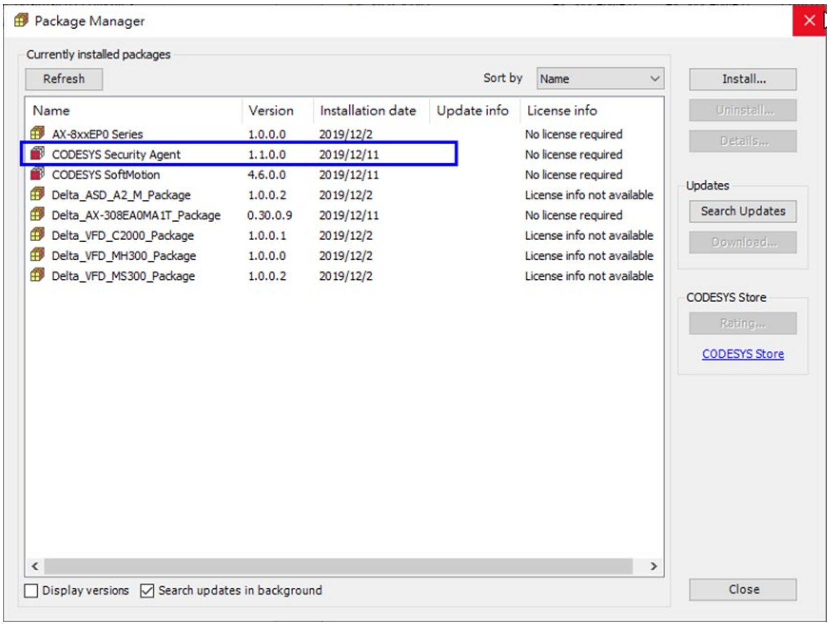


8.3.2 CODESYS Security Agent

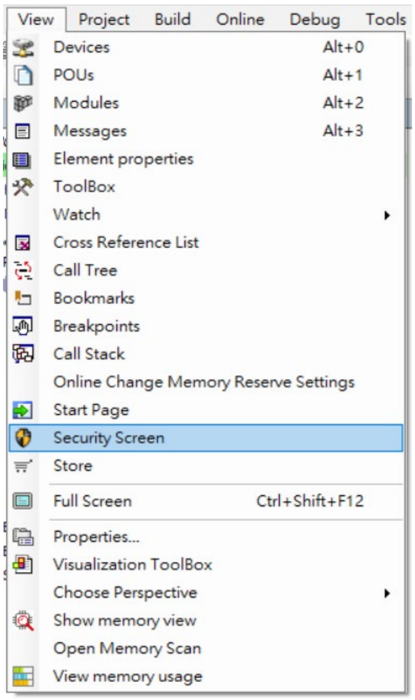
In order to encrypt data and exchange it with the client safely, when the connection is established for the first time, a certificate which has been classified as trusted by the client is required for the OPC UA server. And, CODESYS Security Agent is indispensable for creating a certificate for DIADesigner-AX

1. Visit CODESYS Store to download the software for free: https://store.codesys.com/matrikon-flex-opc-ua-editor.html?___SID=U

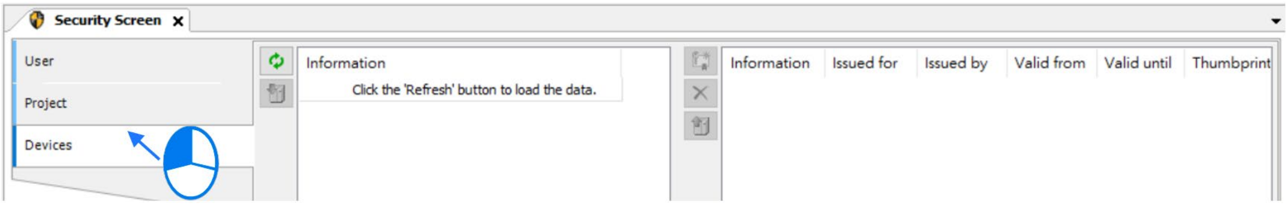
2. Install the add-on  **CODESYS Security Agent**. After installation, you should restart DIADesigner-AX. The window after installation is as shown below:



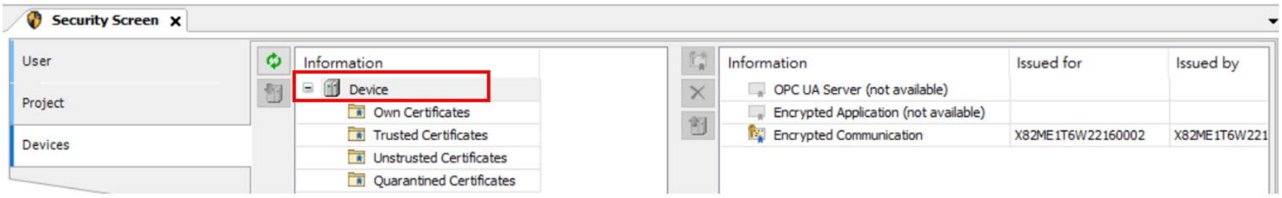
3. Open DIADesigner-AX to create a project. Click **View** on the toolbar and then select **Security Screen** to open the setting page.



4. Select the **Devices** tab.

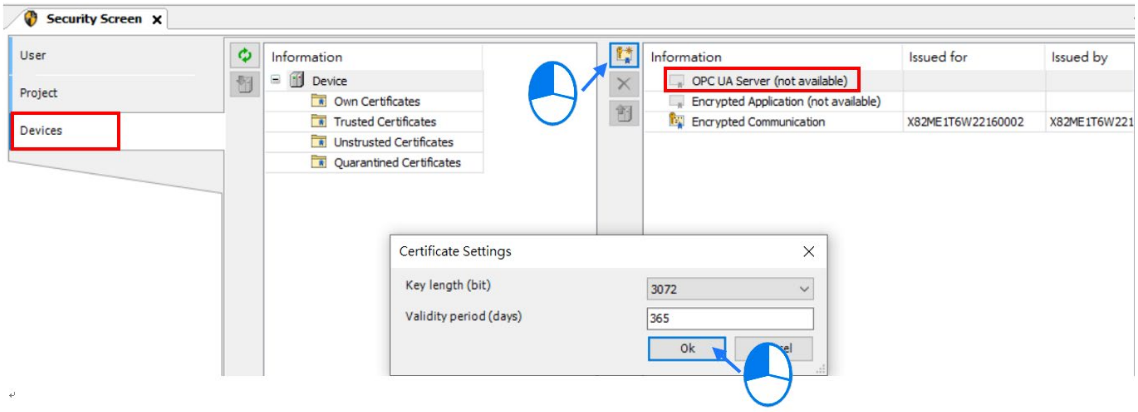


5. Click  to refresh and all services that require a certificate are listed on the right-hand side of the window.

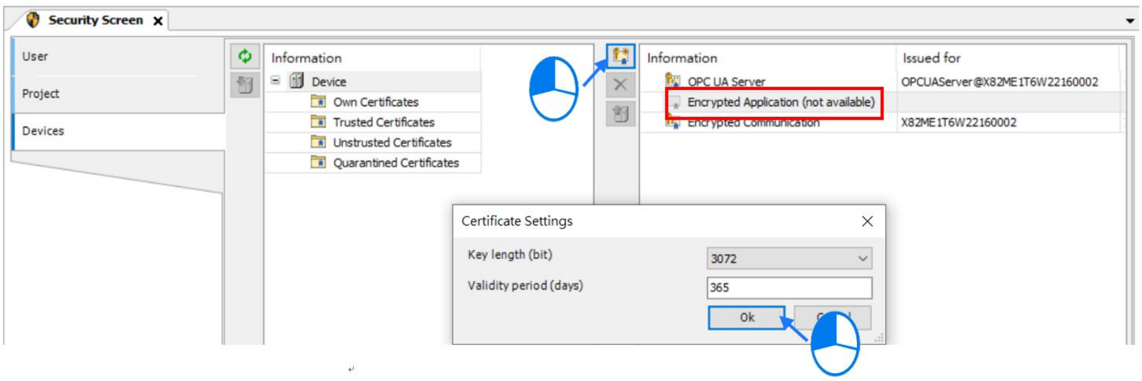


6. Click  to open the **Certificate Settings** window for creating a new certificate for the device.

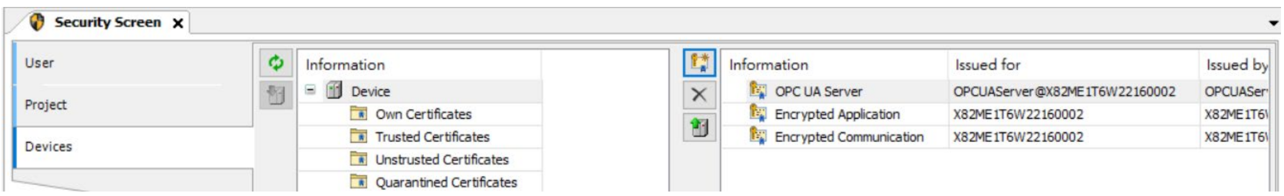
   a. Click the service **OPU UA Server (not available)**.



   b. Select **Encrypted Application**.



7. Then you have created two certificates **OPC UA Server** and **Encrypted Application** on the controller*1.



*1: You need to power off the controller and restart again to enable the certificates.

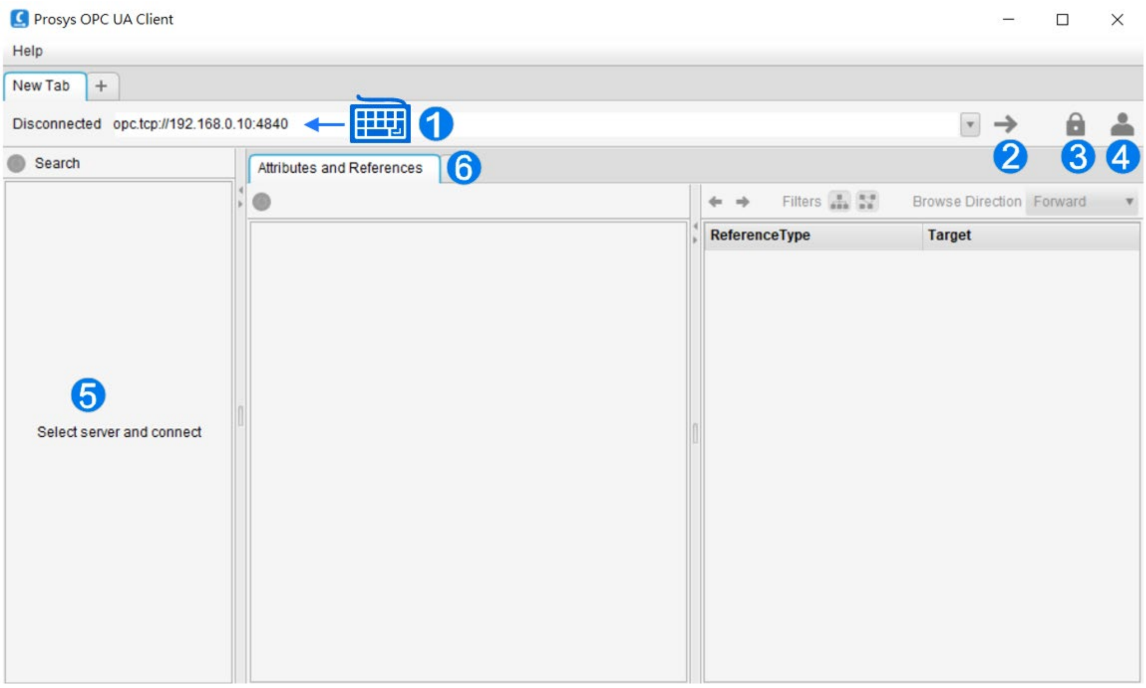
8.3.3 Setting up an Encrypted Connection with Prosys OPC UA Client

In this section, we use **Prosys OPC UA Client V3.2.0** as an example for connection to the OPC UA server. The following illustrates how to set up the connection and the steps can also be applied to other OPC UA clients.

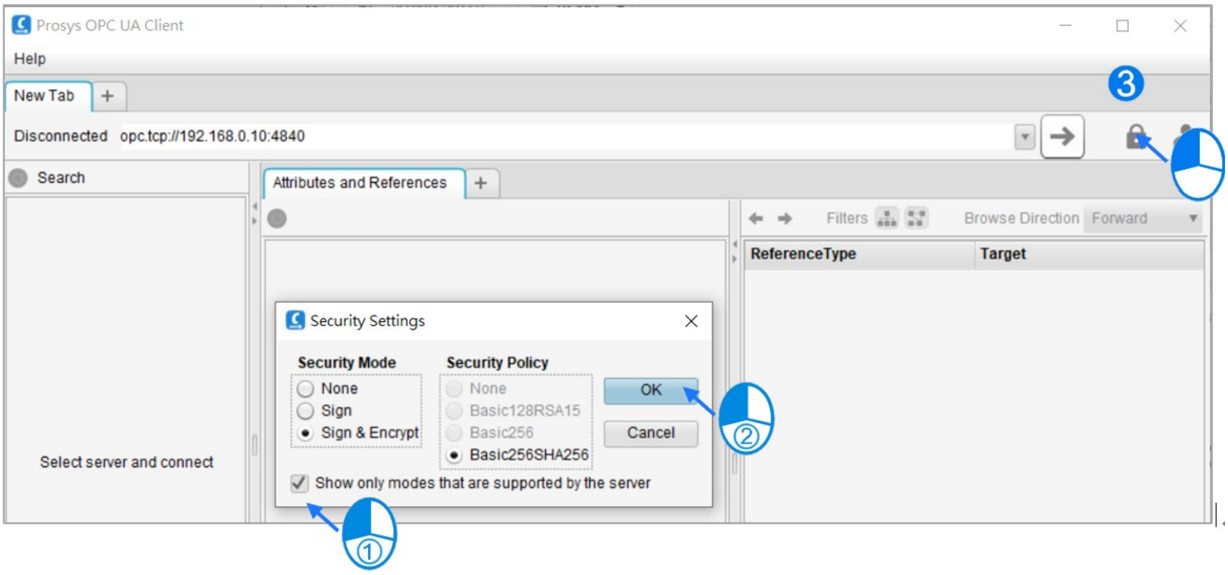
The OPC UA client, **Prosys OPC UA Client**, is freely accessible here: https://downloads.prosysopc.com/opc-ua-client-downloads.php.

1. Double-click to start **Prosys OPC UA Client**.

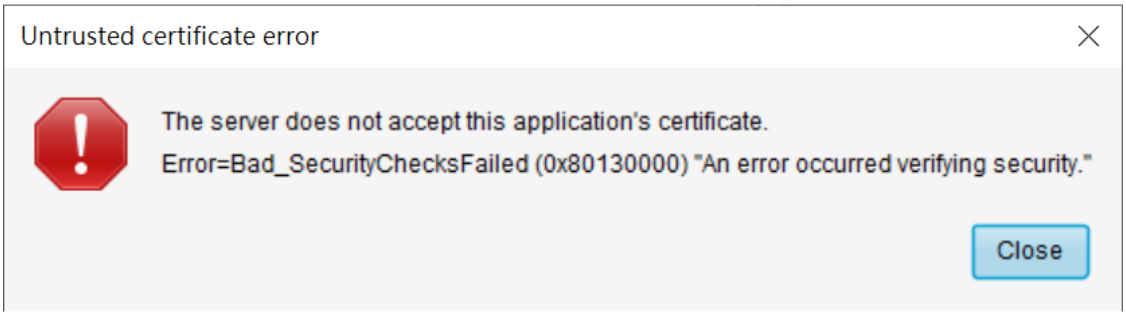
2. Type in the OPC UA Server IP address **opc.tcp://192.168.0.10:4840** in the field of **Disconnected** as shown in ①.



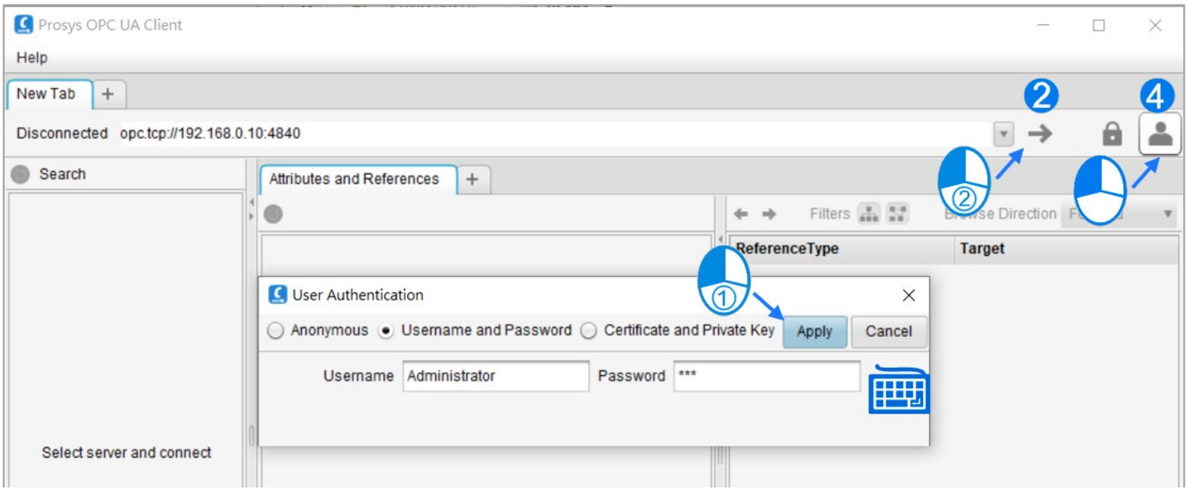
3. Click ③ to open the **Security Settings** window (only the connection type **Basic256SHA256** is supported). Check **Show only modes that are supported by the server** and select both **Sign & Encrypt** and **Basic256SHA256** in the window. Then click **OK**.



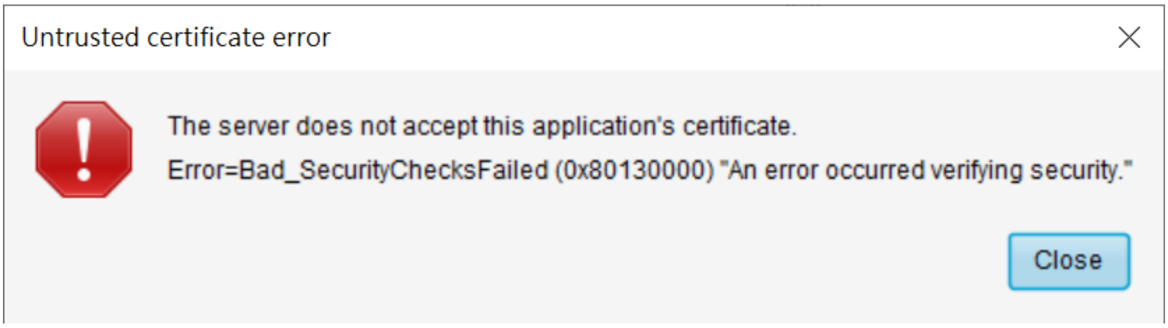
After click **OK**, you will see a warning message **Untrusted certificate error**, stating that **the server does not accept this application's certificate**.



4. Click ④ to set up **Username and Password** in the **User Authentication** setting window. Click **Apply** after the setting is done and then click ② to connect to the controller.

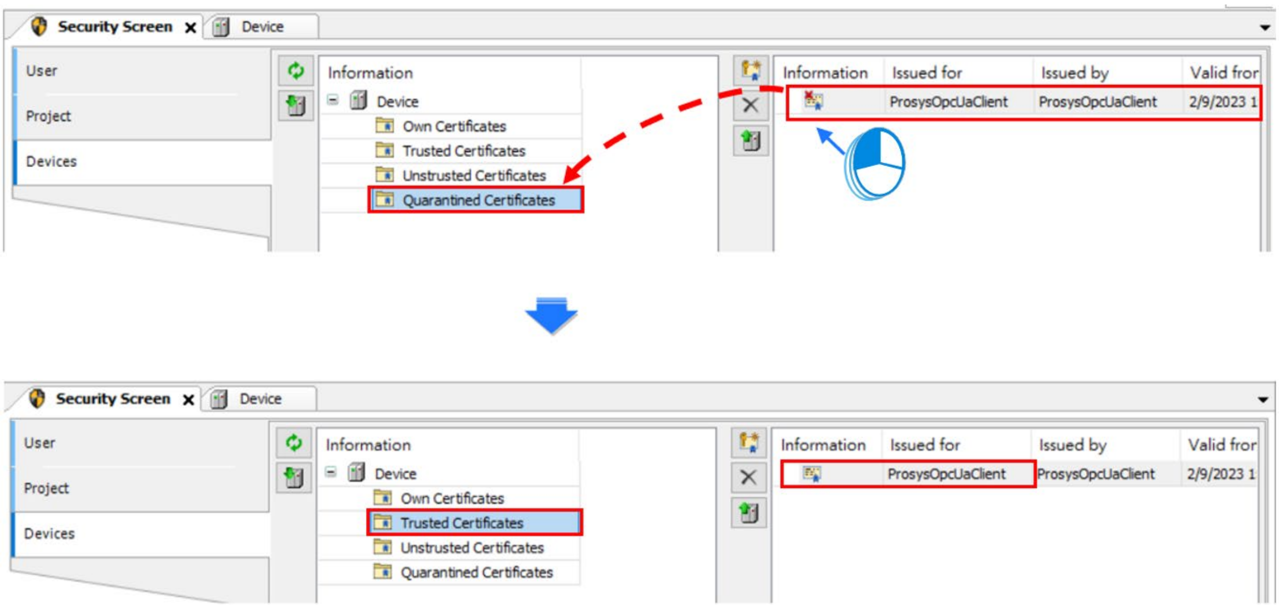


You will see a warning message **Untrusted certificate error** stating that the server does not accept this application's certificate. It is because ProsysOpcUaClient is not a trusted certificate for the AX-8 Series PLC.



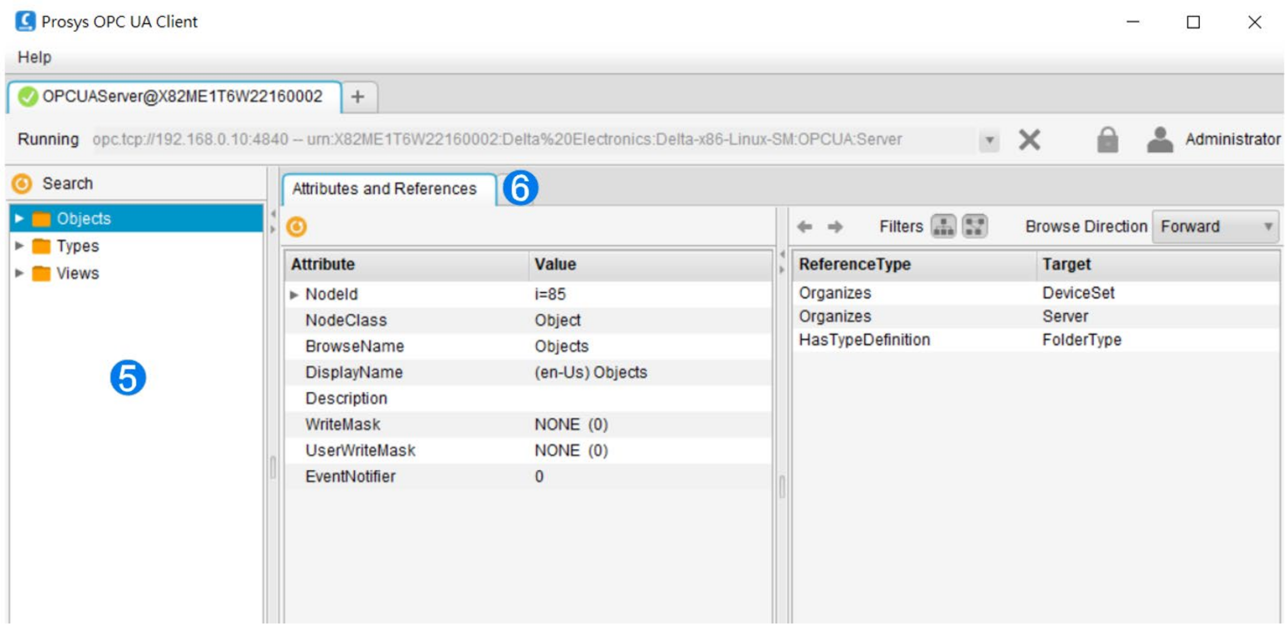
8

5. To deal with the error described in point 3 and 4, go back to DIADesigner-AX. Click **View** on the toolbar and go to **Security Screen** > **Quarantined Certificates**. Find **ProsysOpcUaClient** and drag it to the folder **Trusted Certificates**. Note that you must make sure the AX-8 Series PLC has been connected to the software beforehand.



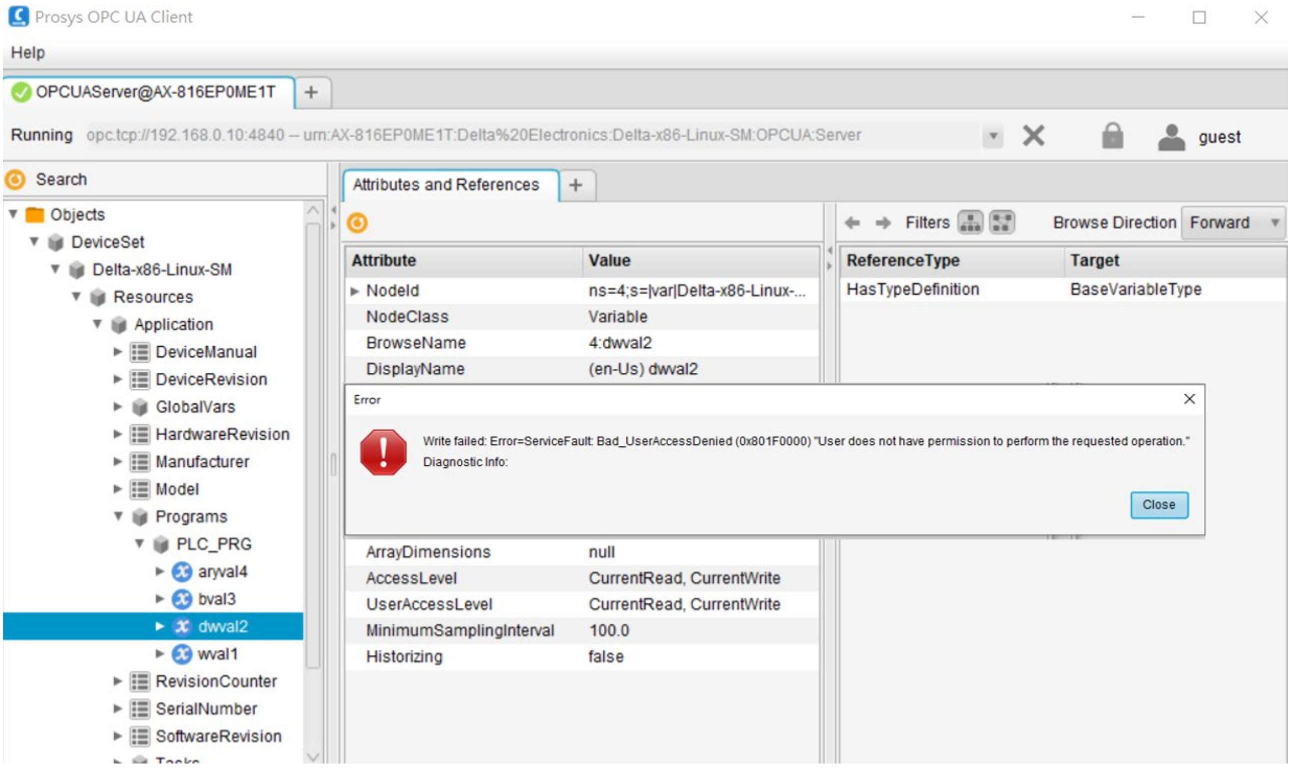
6. Back to ProsysOpcUaClient and you can find the connection to OPC UA Server showned in ⑤. Then you can edit the settings in the controller through ProsysOpcUaClient in ⑥.

   a. User: Administrator



8

b. User: Guest (not authorized to write)



8

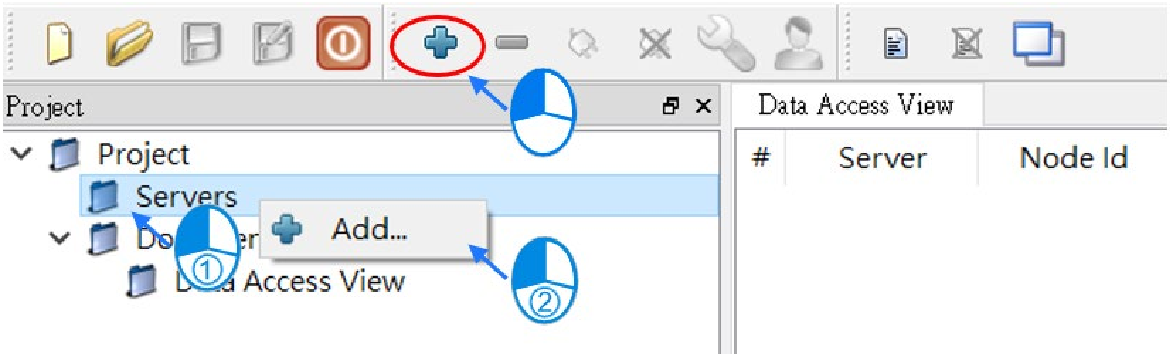
8.3.4 Setting up an Encrypted Connection with UaExpert

In this section, we use **UaExpert V1.5** as an example to connect to the OPC UA server. The following illustrates how to set up the connection and the steps can also be applied to other OPC UA clients.

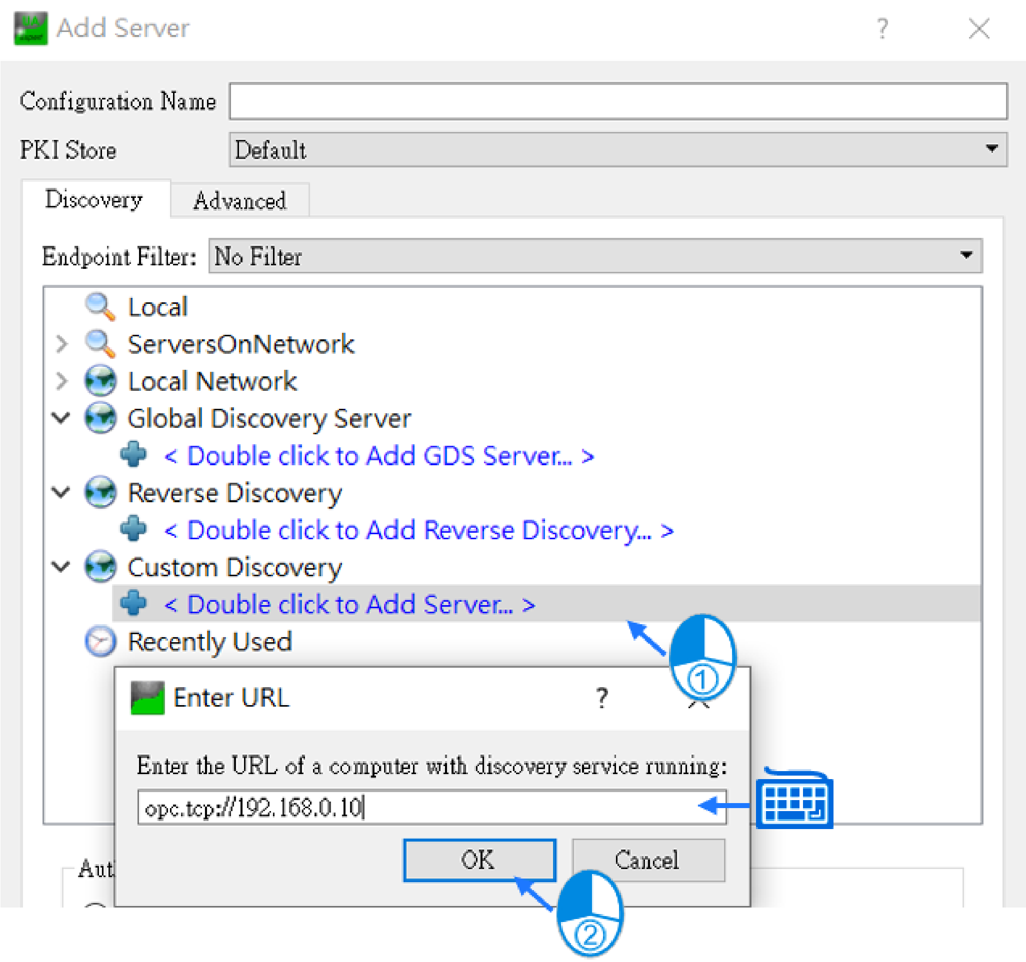
The OPC UA client, UaExpert, is freely accessible here:
https://www.unified-automation.com/downloads/opc-ua-clients.html.

1. Double-click to start UaExpert.

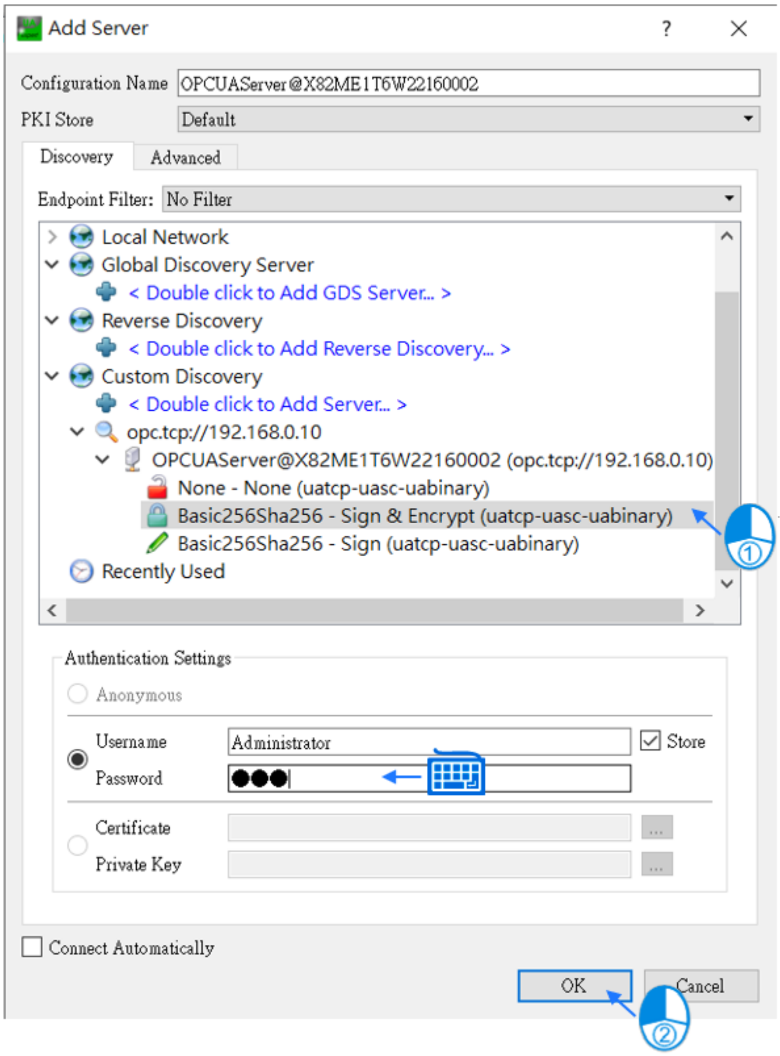
2. Right-click **Servers** and then click **Add** to open **Add Server** window.



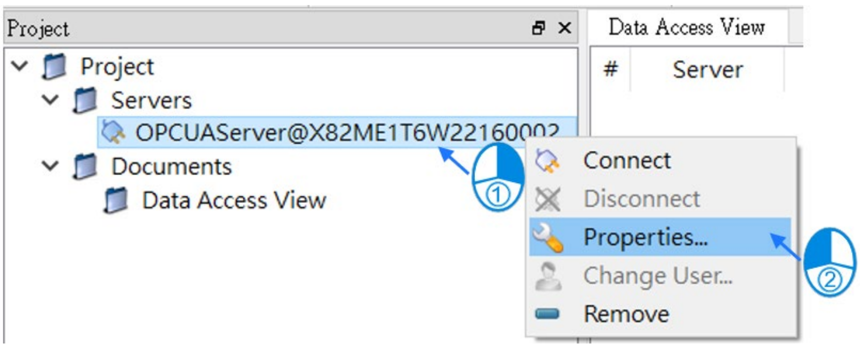
3. Go to **Custom Discovery** > **<Double click to Add Server...>** > **Enter URL**: enter **opc.tcp://192.168.0.10** and click **OK**.



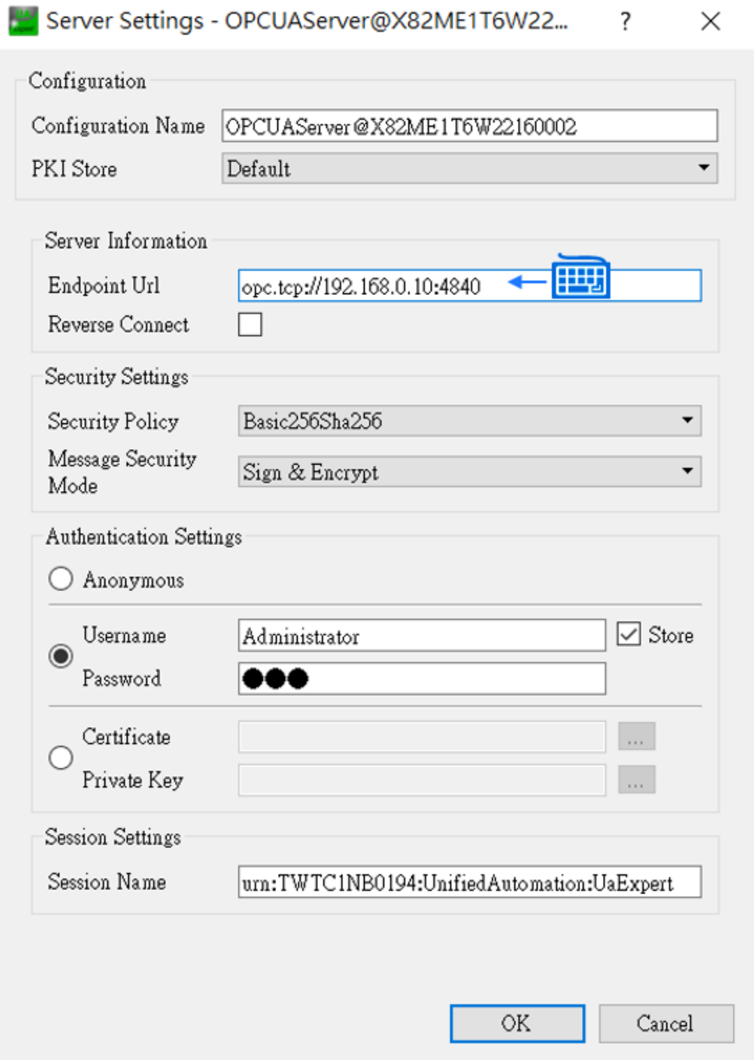
4. The window below pops up right after. You can find **AX-816E** under **opc.tcp://192.168.0.10**. Select **Basic256Sha256** and set up **Username** and **Password** in the **Authentication Settings** field. Click **OK** to create an encrypted connection.



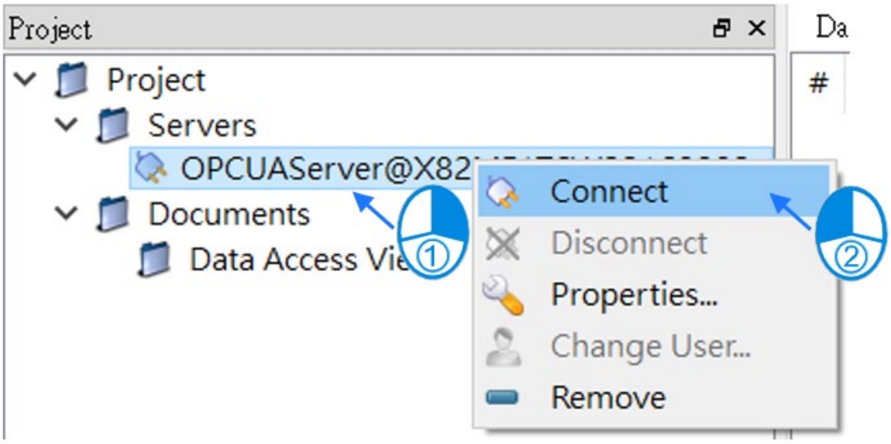
5. To edit the server properties, go back to the starting page. Expand the option under **Project**. Then right-click **OPCUAServer@X8xxxxxxx** and select **Properties…** to open the **Server Settings** window.



6. Rewrite the **Endpoint Url** as **opc.tcp://192.168.0.10:4840**. Click **OK** to close the window.

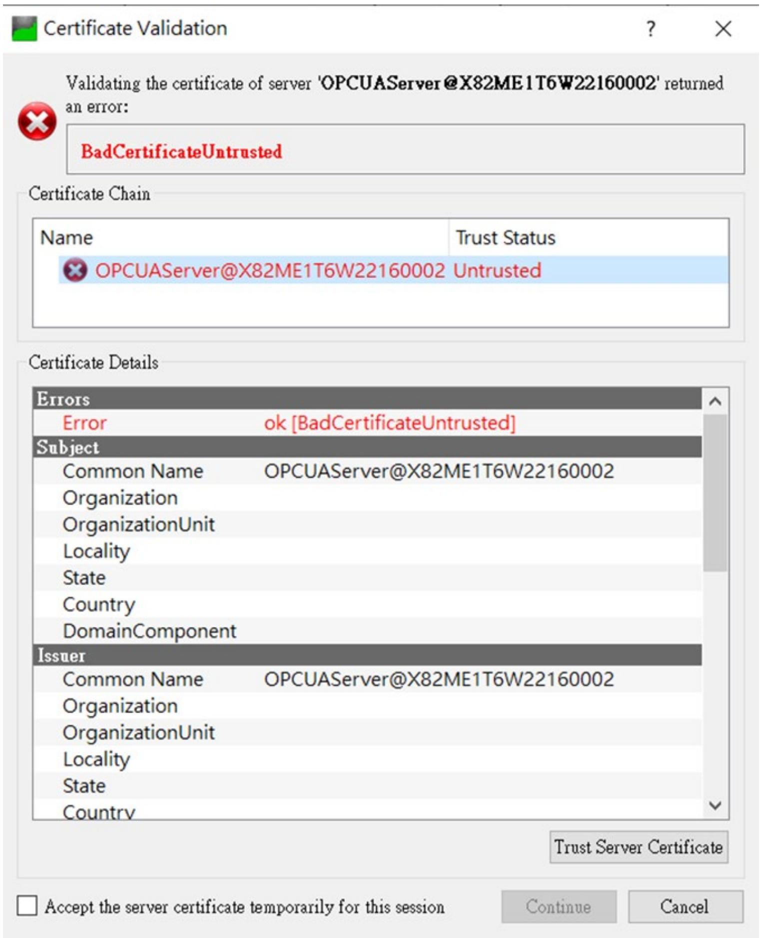


7. Right-click **OPCUAServer@X8xxxxxxx** and select **Connect** on the context menu.

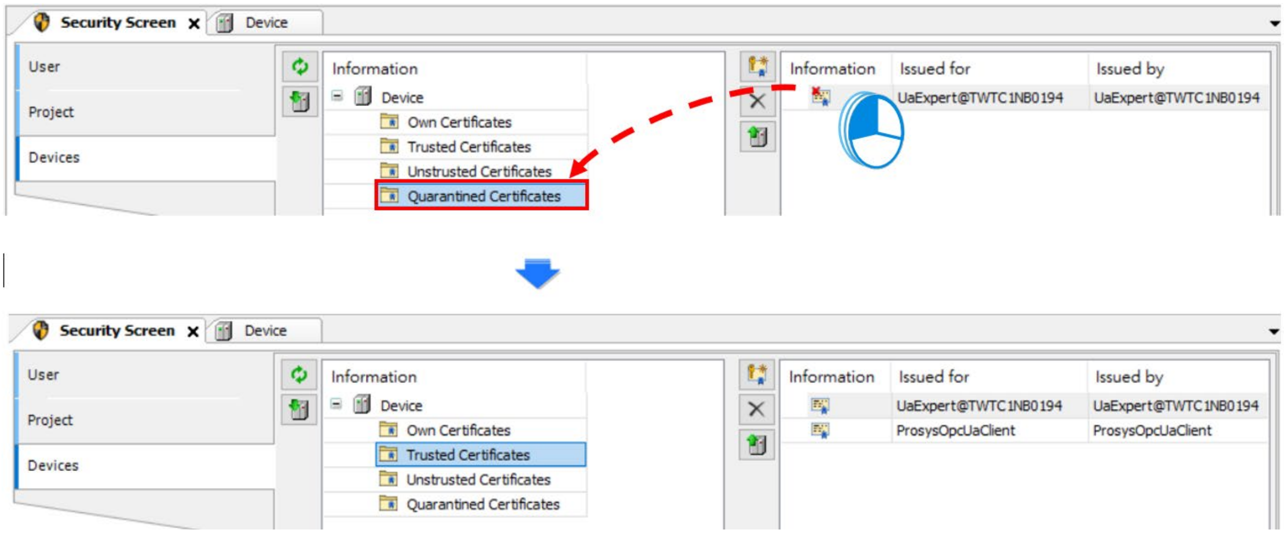


8

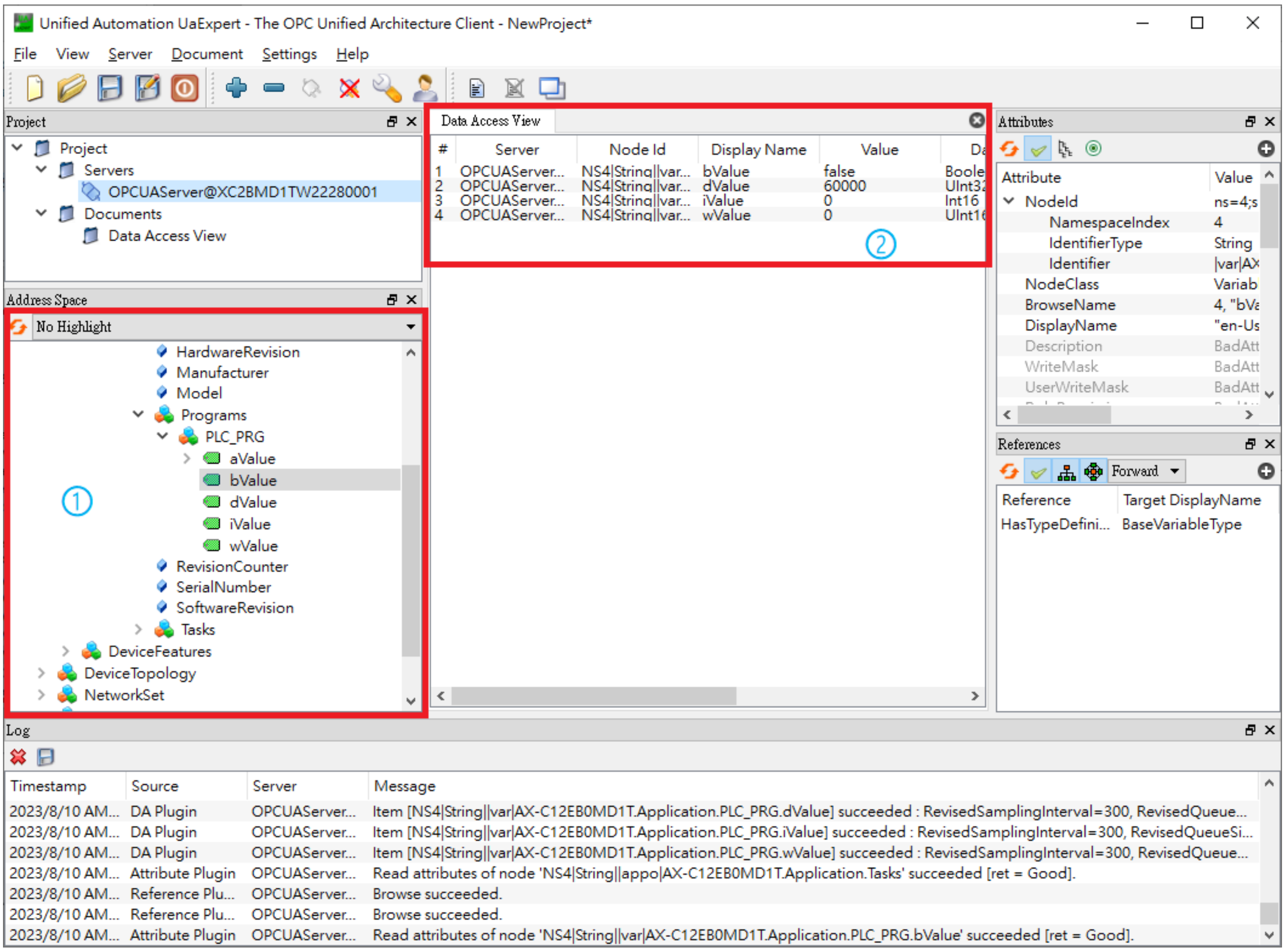
8. You will see a warning message as shown below. Click **Trust Server Certificate**.



9. Go back to DIADesigner-AX. Click **View** on the toolbar and open **Security Screen** setting page. Select the **Devices** tab to find **UaExpert** under **Quarantined Certificates**. Drag **UaExpert** to the folder **Trusted Certificates**.



10. Repeat step 7 to connect to the OPC UA server as an administrator. After the connection is established, you will see the tree node on the left side and the settings in the controller can be edited through UaExpert.



9

Chapter 9 Communication

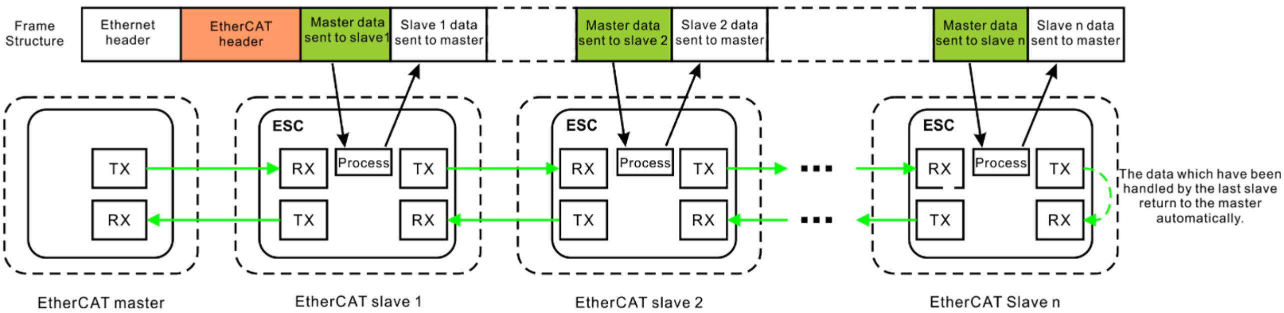
Table of Contents

9.1 Introduction to EtherCAT Communication

9.1.1 Features of EtherCAT Fieldbus

The EtherCAT bus is the Ethernet-based fieldbus. The communication rate of the EtherCAT network is 100Mbps and the distance between two adjacent nodes is within 50 meters. The EtherCAT network is noticeably very different from the general Ethernet network. One EtherCAT network has just one EtherCAT master and EtherCAT slaves contain ESC chips (EtherCAT Slave Controller) specially used for processing EtherCAT communication data and inserting the data which slaves need to transmit to the master into the EtherCAT frame. The last EtherCAT slave in the network will return the data which have been handled to the master in chronological order. See the illustration of data transmission shown below.

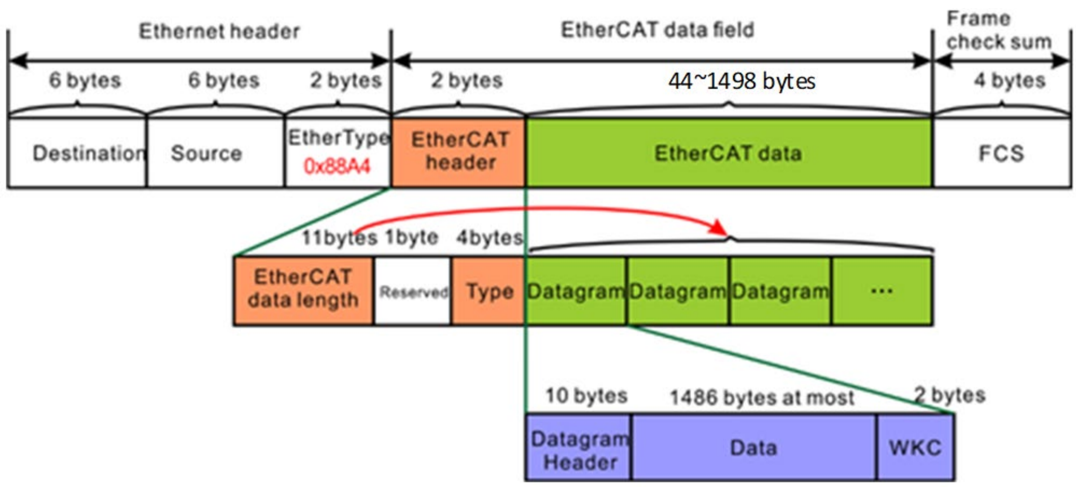
Thanks to the ESC chips in slaves, the master can make a communication with all slaves in an EtherCAT data frame and thus the communication efficiency is enhanced.



- **EtherCAT Communication between the Controller and Slaves**

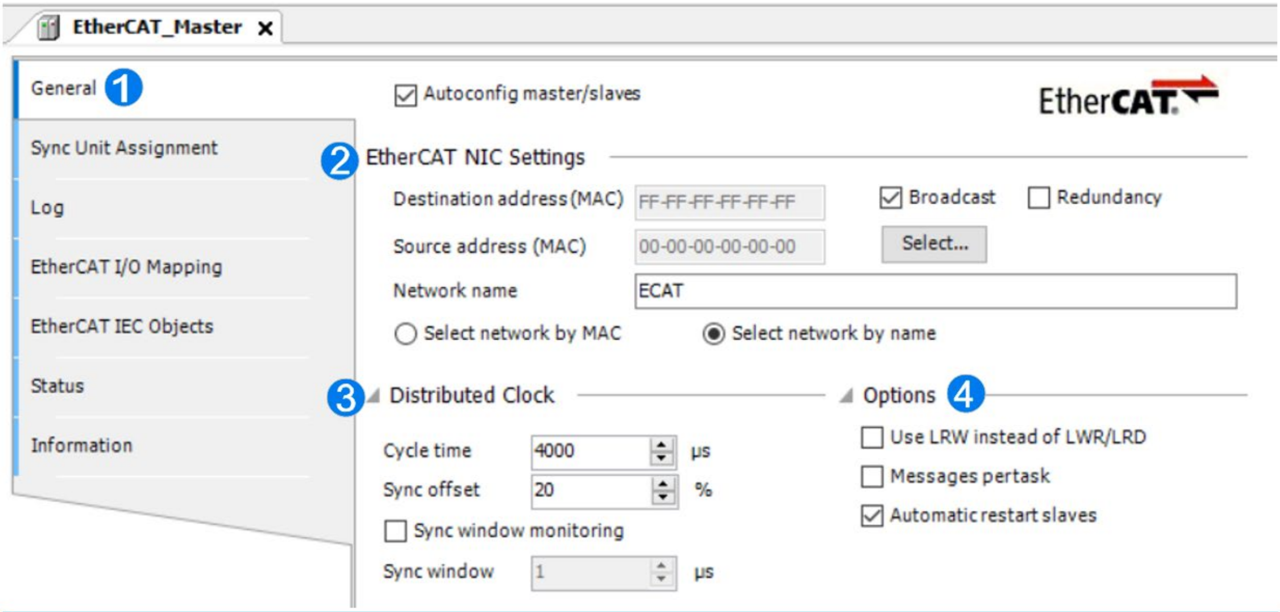
Since the EtherCAT bus is the EtherNet-based fieldbus, the EtherCAT data frame still adopts the UDP/IP Ethernet data frame structure.

EtherCAT data field includes 2 bytes of EtherCAT data header and 44~1498 bytes of EtherCAT data. EtherCAT Data field consists of one or more EtherCAT datagrams. EtherCAT Data can be defined and analyzed in a protocol as long as the master and slaves comply with the protocol. Currently, the mostly used two protocols are COE (CANopen Over EtherCAT) and SOE (Sercos Over EtherCAT). EtherCAT data frame structure is as displayed below.



9.1.2 Setting up EtherCAT Master

This section introduces functions in the tab of EtherCAT_Master.



- **General**

① Autoconfig Master/Slaves: Enable this option to have basic configurations done. Suggested to use this option.

② EtherCAT NIC Setting:

- Destination address (MAC): MAC address of the device in the EtherCAT network that is to receive the telegrams.
- Source address (MAC): MAC address of the controller (Browse Scan Slave Station and choose ECAT)
- Network name: Name or MAC of the network, depending on which of the following options is activated.
- Select network by MAC*1: The network is specified by the MAC ID (The default name is ECAT*1).
- Select network by name: Network is identified by the network name and the project is device-independent.

*1: The default name is ECAT.

③ Distributed Clock

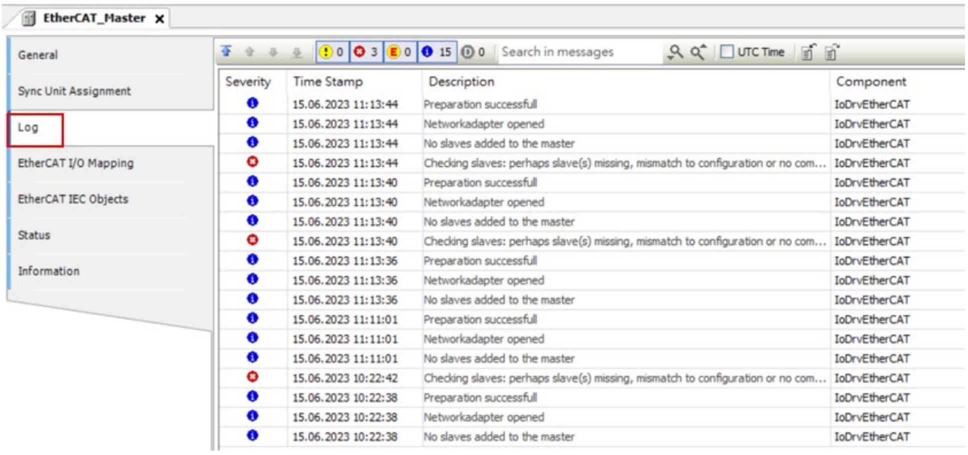
- Cycle time: Master sends out corresponding data to the Slaves in a cycle time specified here.
- Sync offset: Parameter for setting the delay time between the Distributed Clock time base of the EtherCAT slave and the cycle start of the PLC. With the default value of 20%, the PLC cycle starts 20% of the bus cycle time after the sync interrupt of the slave. For the controller program, 80% of the cycle is always available. Here the Sync offset determines only when the EtherCAT data of the master is exchanged to and from the slaves relative to the time base of the EtherCAT slave.
- Sync window monitoring: Enabled to monitor the synchronization of the slaves.
- Sync window: Time for Sync window monitoring.

④ Options

- Use LRW instead of LWR/LRD: Use combined read/write commands/PDO (LRW) instead of separating read (LRD) and write commands (LWR).
- Automatic restart salves: In the case of a communication breakdown, the master immediately attempts to restart the slaves.

● **Log**

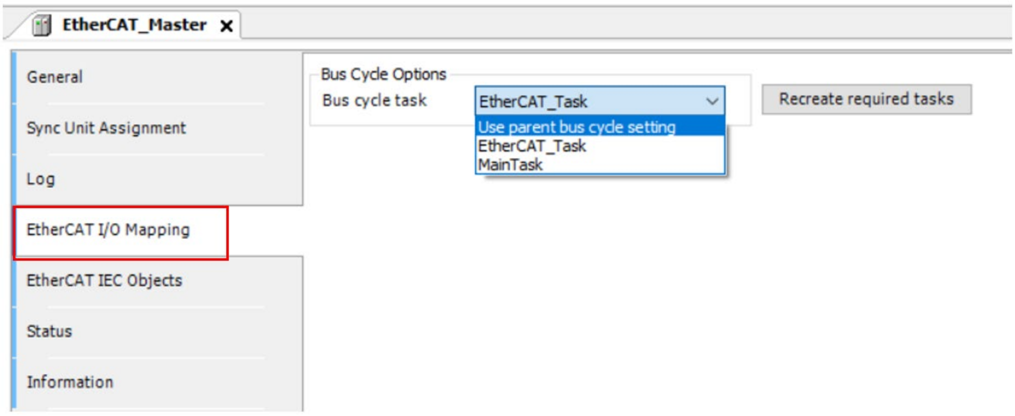
Here you can view the PLC log. It lists the events that was recorded on the target system. Refer to **section 4.2.1.5 Log** for more information.



● **EtherCAT I/O Mapping**

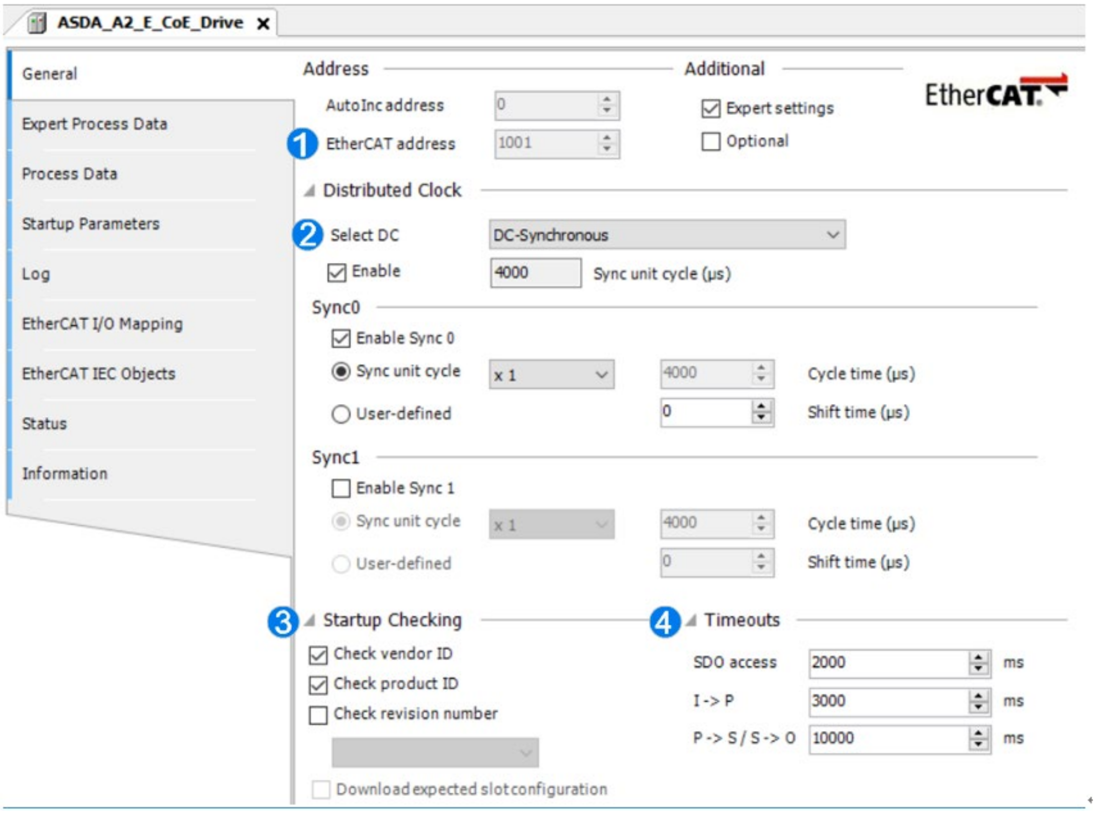
Here you can select the bus cycle task for EtherCAT communication. The bus cycle task selected will be synchronized with the specified EtherCAT_Master cycle time.

- Bus cycle task: Select a bus cycle task to synchronize with the EtherCAT communication time. When the option "Use parent bus cycle setting is selected", the system uses the shortest cycle time as the EtherCAT cycle time.



9.1.3 Setting up EtherCAT Slave

This section introduces functions in the tab of Slave. You can either add slaves from the Product list or scan the network to add the slaves in.



- **General**
  - Address

    ① EtherCAT address: Final address of the slaves, assigned by the master during bootup. The address is independent of the position of the slave in the network.

  - Distributed Clocks

    ② Select DC: Cycle time for the data exchange.

  - ③ Startup Checking

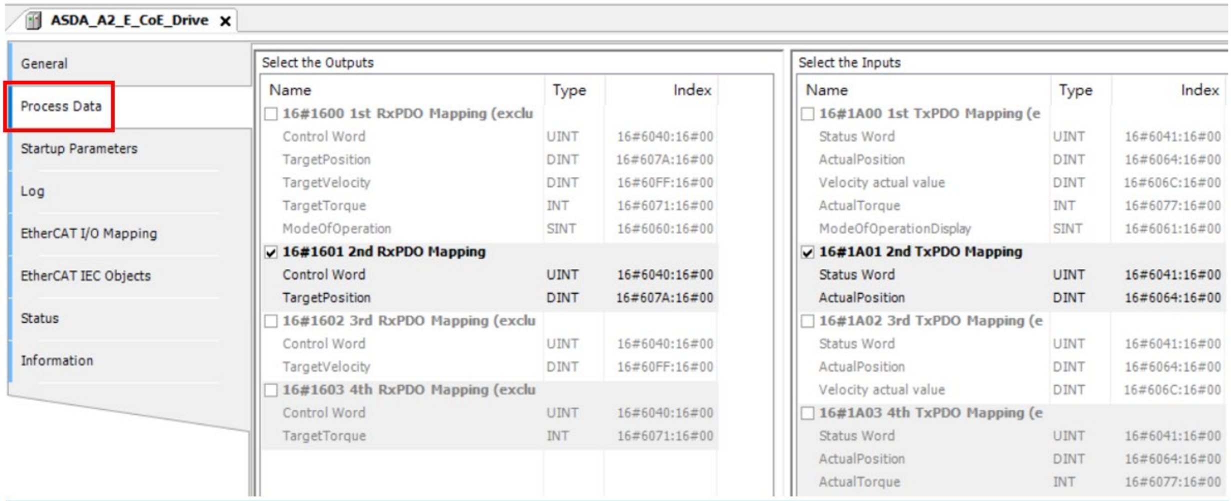
Function	Description
Check vendor ID	Once the system starts, it checks if the vendor ID and product ID are the same as the configured. If not, the system stops without any further operation.
Check product ID	
Check revision number	Once the system starts, it checks if the revision number is the same as the drop-down list showed.

■ ④ Timeouts

Function	Description
SDO access	Once the system starts, the SDO also starts transmitting. Unit: ms
I → P	Switching form Init mode to Pre-operational mode. Unit: ms
P → S / S → O	Switching from Pre-operational mode to Safe Operational mode. Or switching from Safe-Op mode to Operational mode. Unit: ms

- **Process Data**

The data mapping of the EtherCAT network is a cyclic data exchange between the master and slave through the CoE-based PDO mapping. The data that a slave sends to the master are packed in TxPDO and the data that the slave reads from the master are packed in RxPDO. The inputs and outputs on the pages of Select the Outputs and Select the Inputs contain the lists of PDOs which are available for data exchange and can be edited. For ESI file of a device, the PDOs and PDO contents for option have been defined and some PDO contents can be edited by users themselves as defined in ESI.

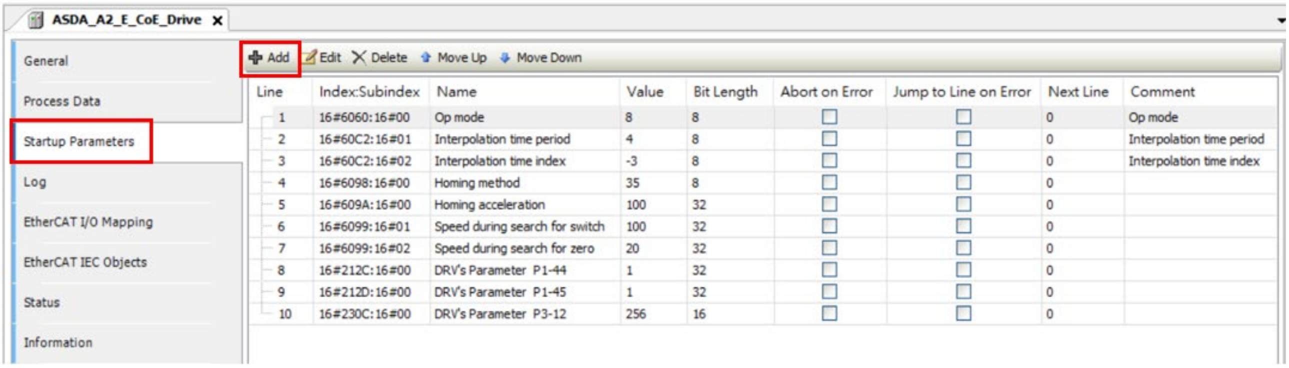


If outputs of the device are activated here (for writing), these outputs can be assigned to project variables in the EtherCAT I/O Mapping window. And if inputs of the device are activated here (for reading), these inputs can be assigned to project variables in the EtherCAT I/O Mapping window. It takes more PLC system resources, if you use more PDOs.

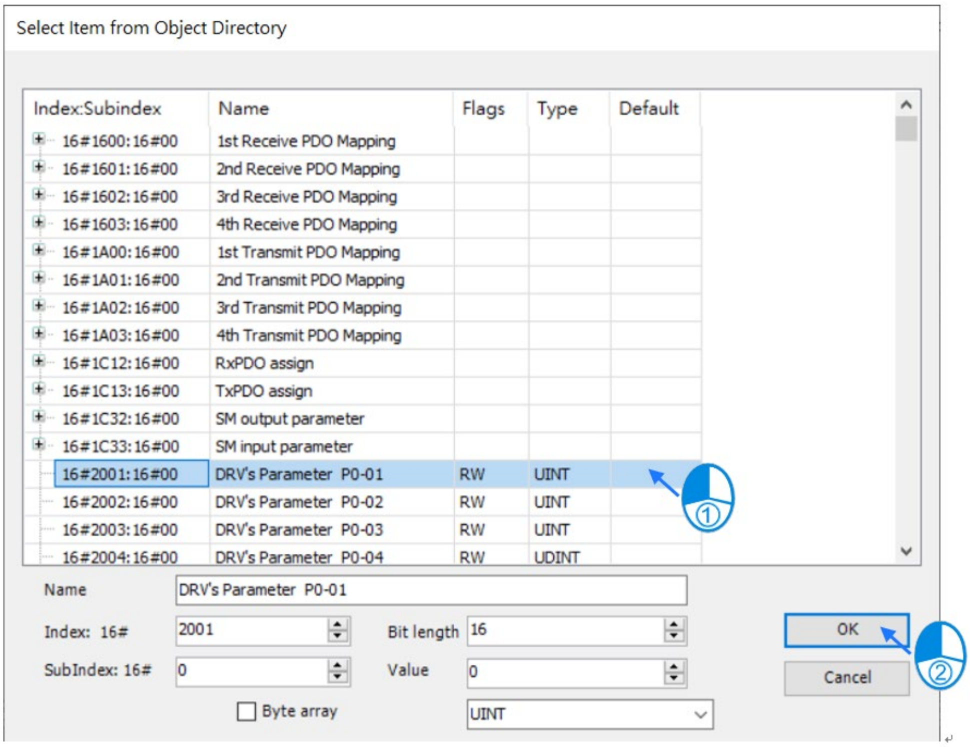
- **Startup Parameters**

The table shows the commands which have been defined by default in ESI file when the master will read and write values to the slave in the specific status of EtherCAT network operation. Users can add or reduce or modify commands in the table.

Function Button	Description
Add	By specifying new index/subindex entries, a new object can be added to the SDO that is not yet described in the EDS file. This is useful if only an incomplete object directory or none at all is present.
Edit	In this window you can change the parameters of the SDO before the SDO is added to the configuration.
Move Up	Moves the selected line upwards by one line.
Move Down	Moves the selected line downwards by one line.



Click **Add button** to open the **Select Item from Object Directory** window. And select the parameter that you'd like to add and then click **OK** to add the item in.



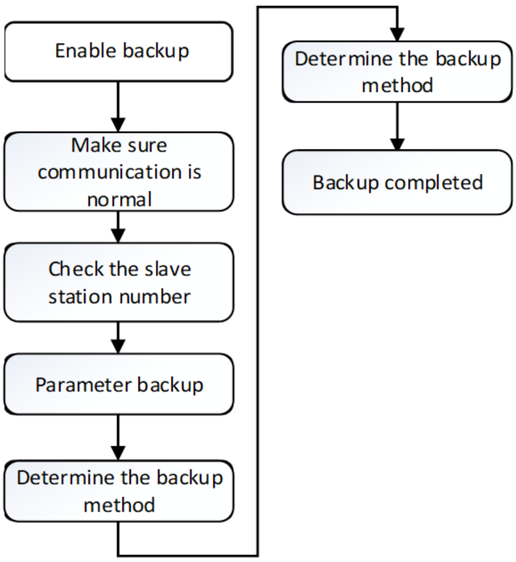
9

9.1.4 Backup Parameters for EtherCAT Slaves

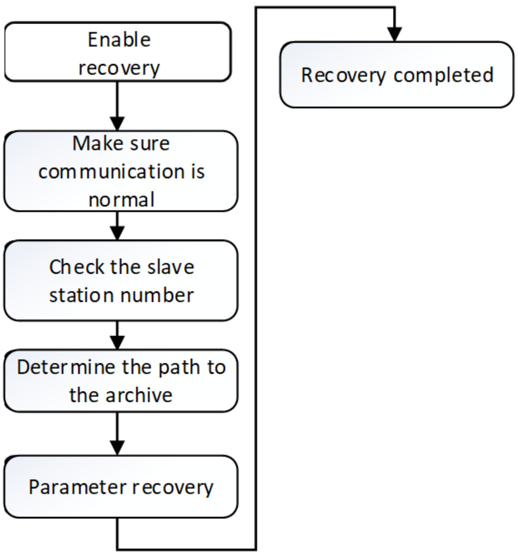
While using EtherCAT communication, we provide customized parameter storage feature for ASDA series servo drives with the backup feature to backup and restore parameters of all slave stations.

9.1.4.1 Data Backup Procedure

- **Backup procedure**



- **Recover procedure**



If there's any existing axis error while executing parameter backup or recovery, the corresponding slave station would be skipped in the backup/ recovery procedure. After the backup/ recovery of all the rest of slave stations are completed, all the related messages of axis errors would be displayed.

9.1.4.2 Introduction to Backup and Restore

- **Supported version for backup and recovery**
  - AX-8 series firmware version: V1.0.x.x and above
  - DIADesigner-AX version: V1.5 and above
  - Only models ASDA-A3-E and ASDA-B3-E are supported for parameter backup and recovery.

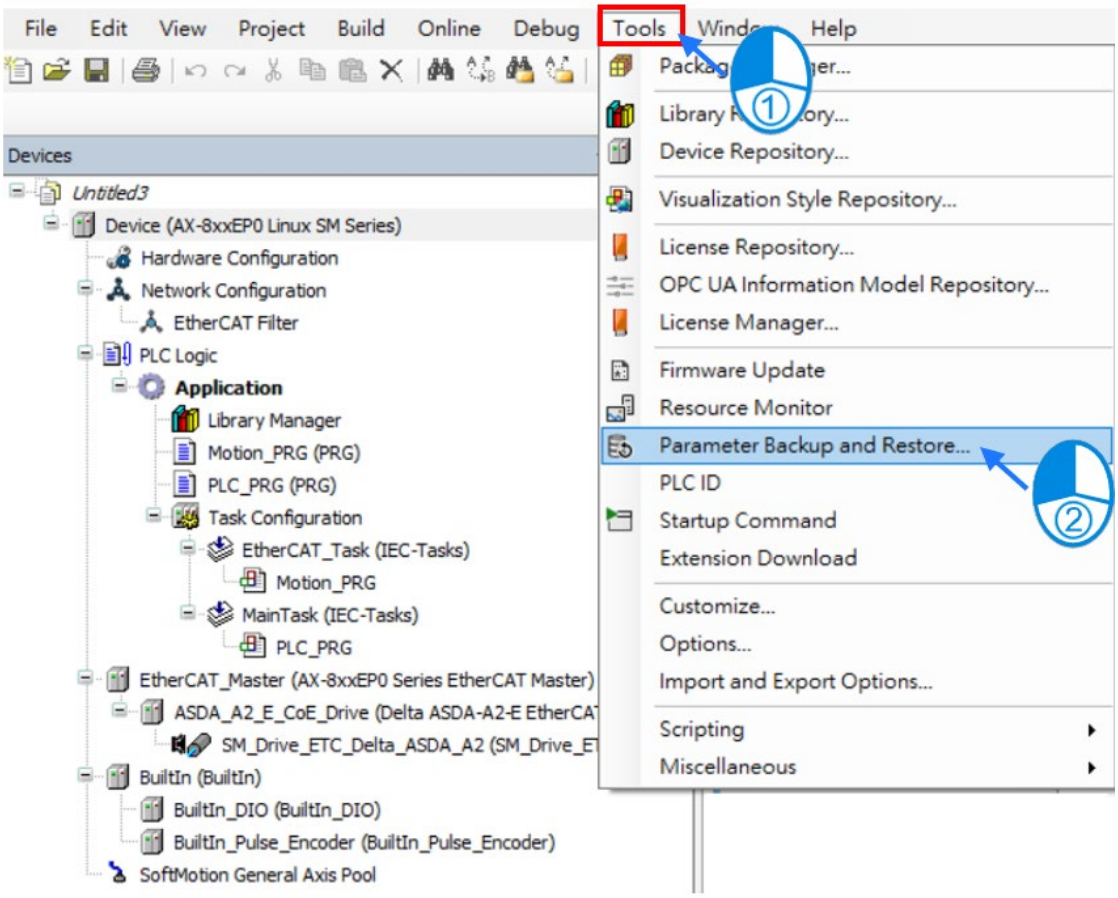
    - ASDA-A3-E firmware version: V11165 sub 92 and above

    - ASDA-B3-E firmware version: V10665 sub 75 and above

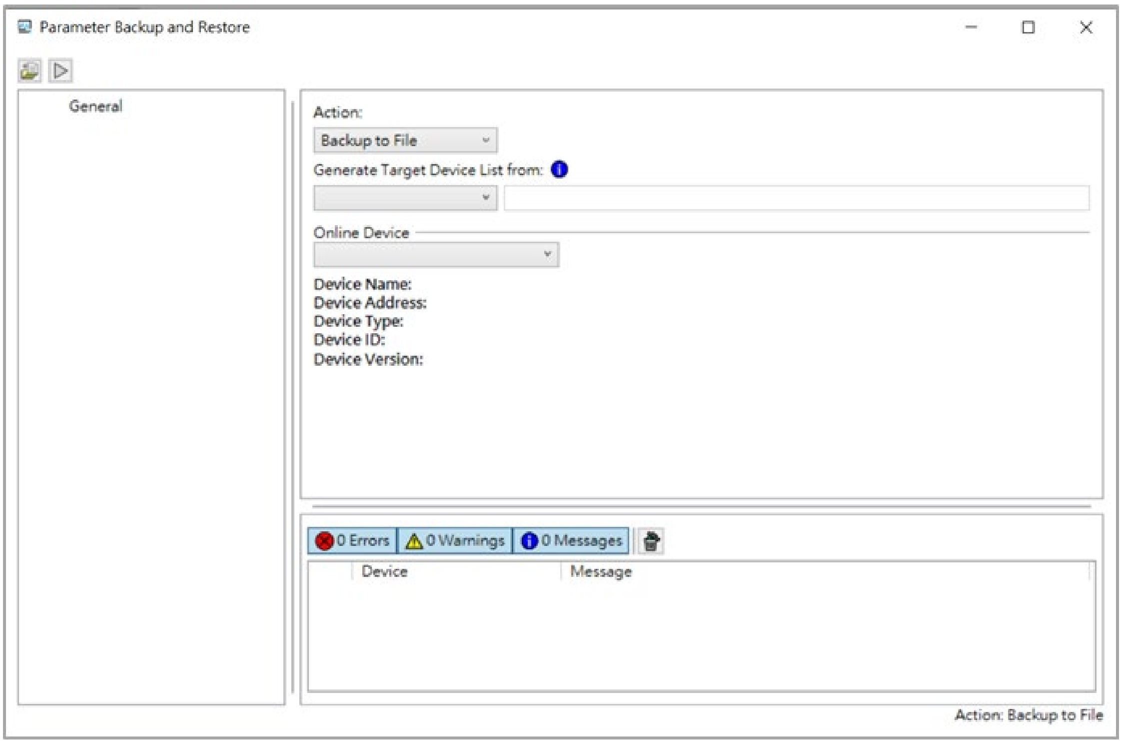
- **Data that is backed up**

Servo parameters P0~P4 (Not including P0.001 and P4.000), P5.0003 + P5.0008~P5.0009 + P5.0020~P5.0030 + P6.0000~P6.0001.

- **Enter the parameter backup page**



- **Enter the parameter restore page**



Name	Function
Online Device	Select the target device to connect.
Generate Target Device List from	Select EtherCAT project tree ■ Archive File → EtherCAT topology file ■ Online Topology → Online EtherCAT topology
Action	Select the target action ■ Backup to File → Backup parameters to files. ■ Backup to SD Card → Backup parameters to external SD cards. ■ Restore from File → Restore parameters from files. ■ Restore from SD Card → Restore parameters from SD cards.
	Save the current EtherCAT topology (Archive File)
	Execute the backup / restore feature

- **External SD card backup path**

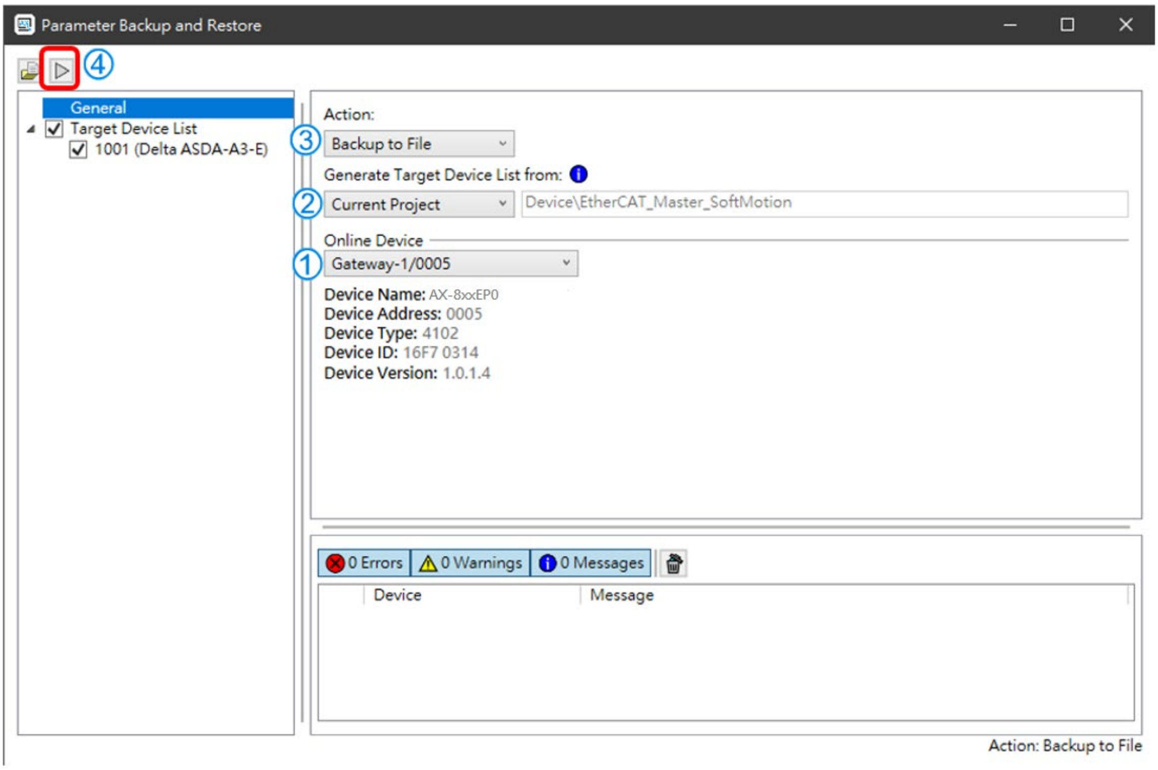
External SD card path: /PLC CARD/AX_/SysDup/ECAT/BackupRestore/ (The "_" in the path represents model types. For example, model AX-8 would be defined as AX8 here.)

9

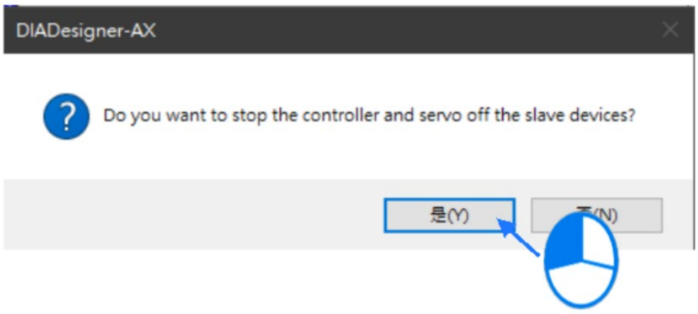
9.1.4.2.1 Operation for Backup Function

- **Parameter backup**

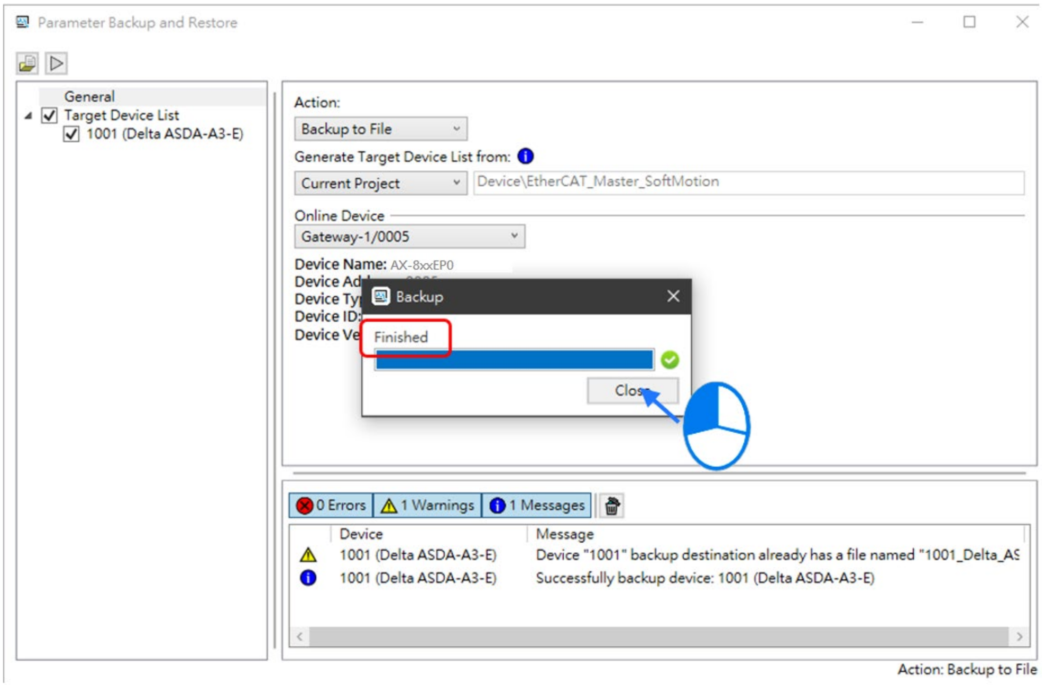
  ① Select Device
  ② Configure EtherCAT topology for the current project
  ③ Set Action to “Backup to File”
  ④ Execute backup



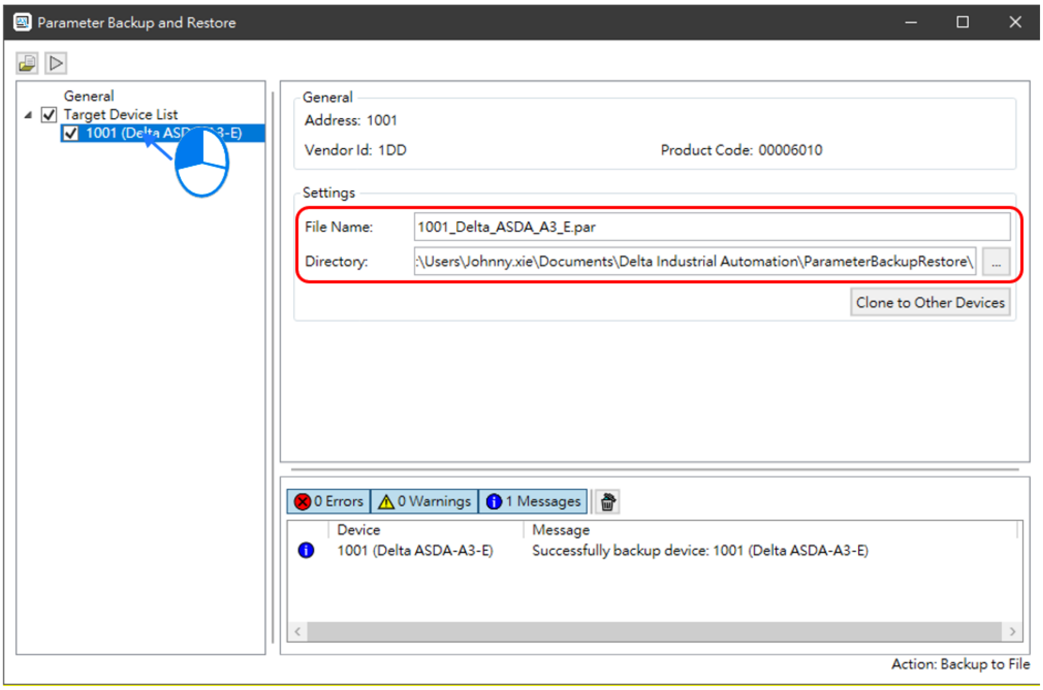
- **Change the PLC state to Stop.**



- **Click Close after the parameter backup is completed.**



- **Backup directory**

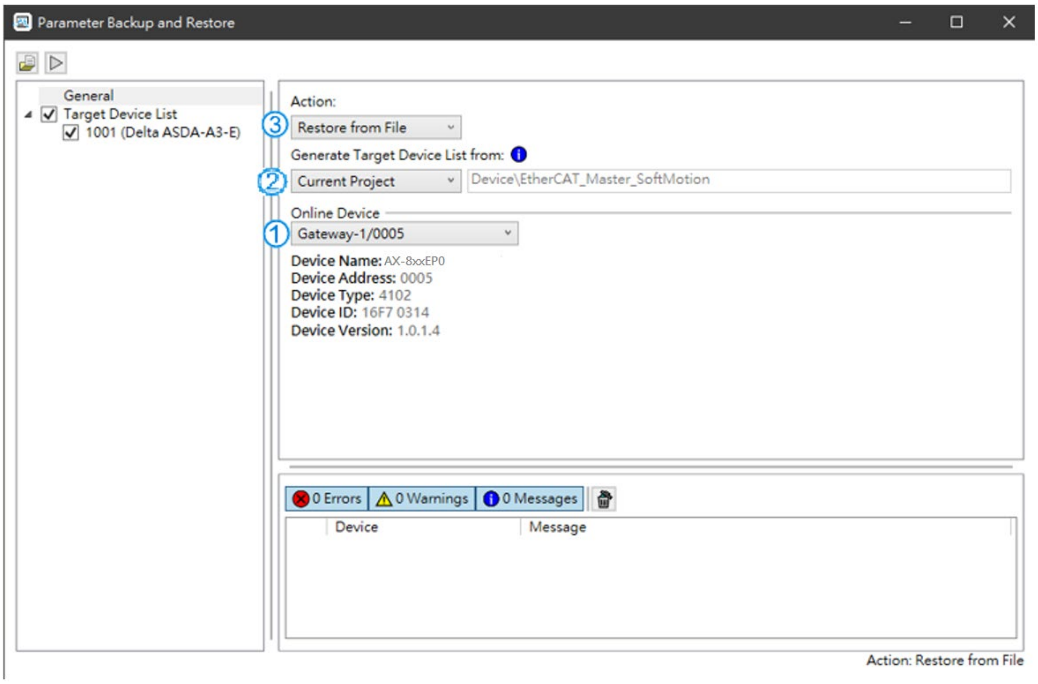


Name	Function
File Name	Set the name for parameter backup file.
Directory	Set the backup directory.
Clone to Other Devices	Change all the backup directory of other devices.

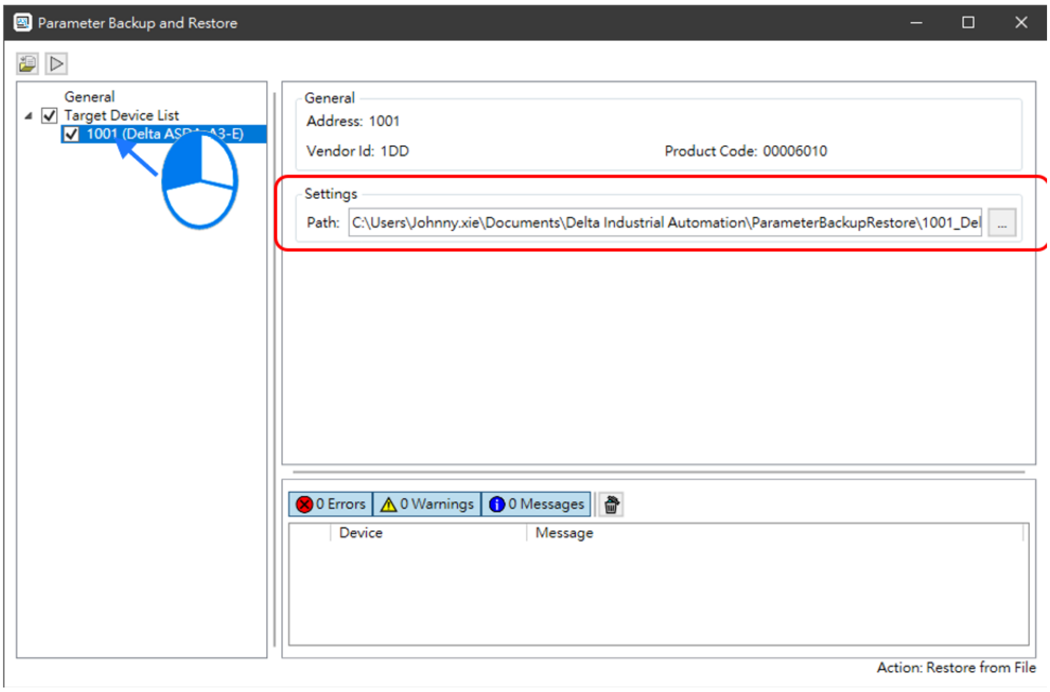
9.1.4.2.2 Operation for Restore Function

- **Parameter restoration**

  ① Select Device
  ② Configure EtherCAT topology for the current project
  ③ Set Action to **Restore from File**

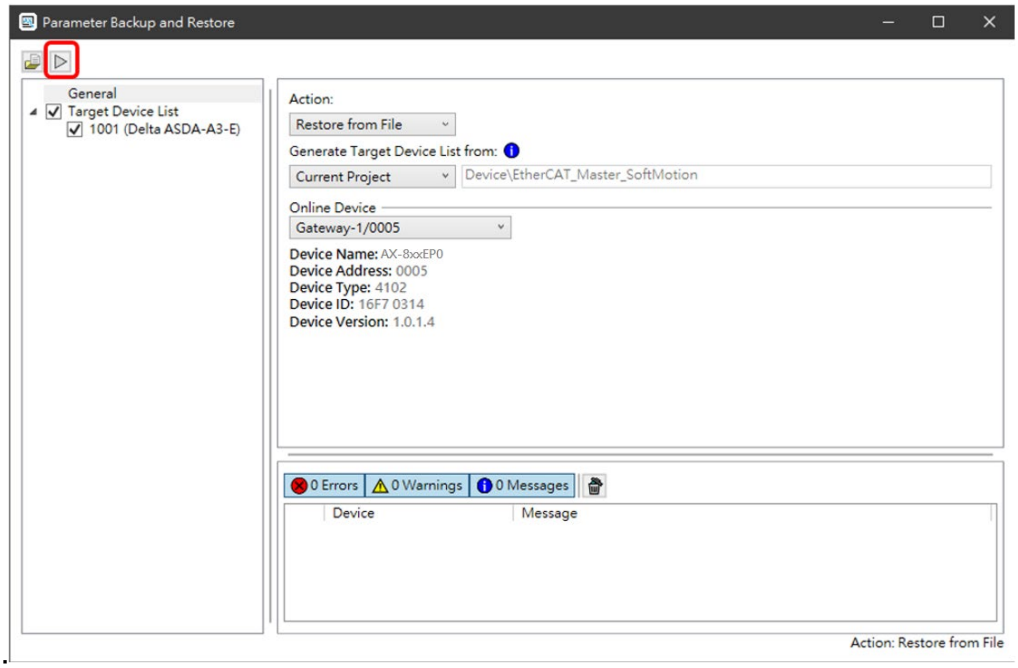


- **Click on the target device and set the path to the file to restore.**

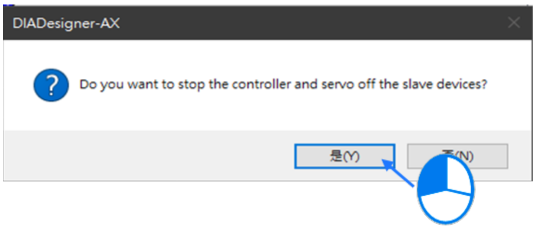


9

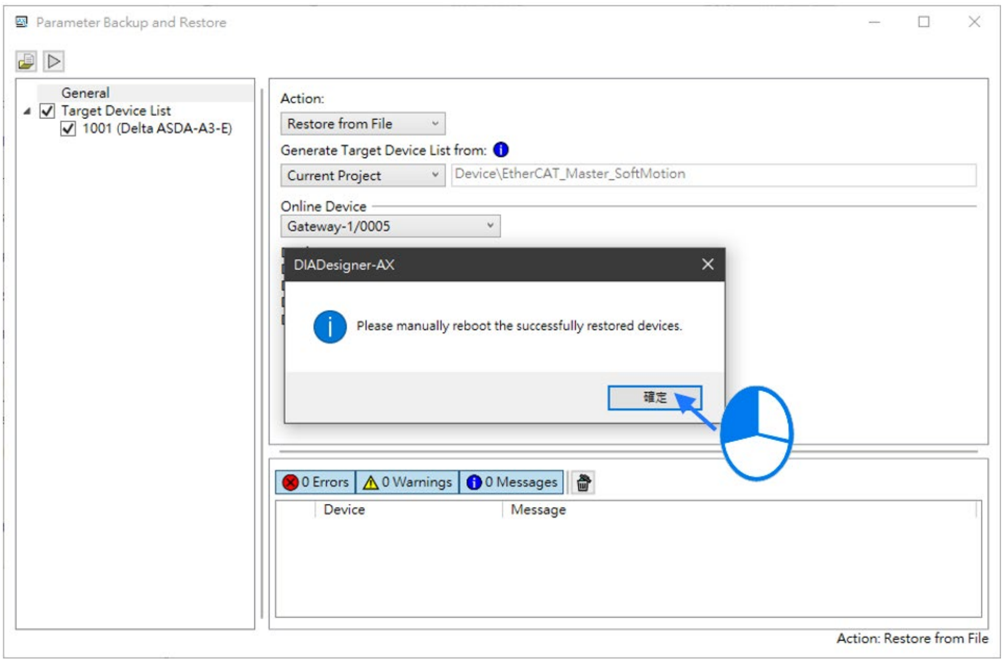
- **Activate the restore function**



- **Change the PLC state to Stop.**



- **Click Yes after the restoration is completed, then reboot the device.**

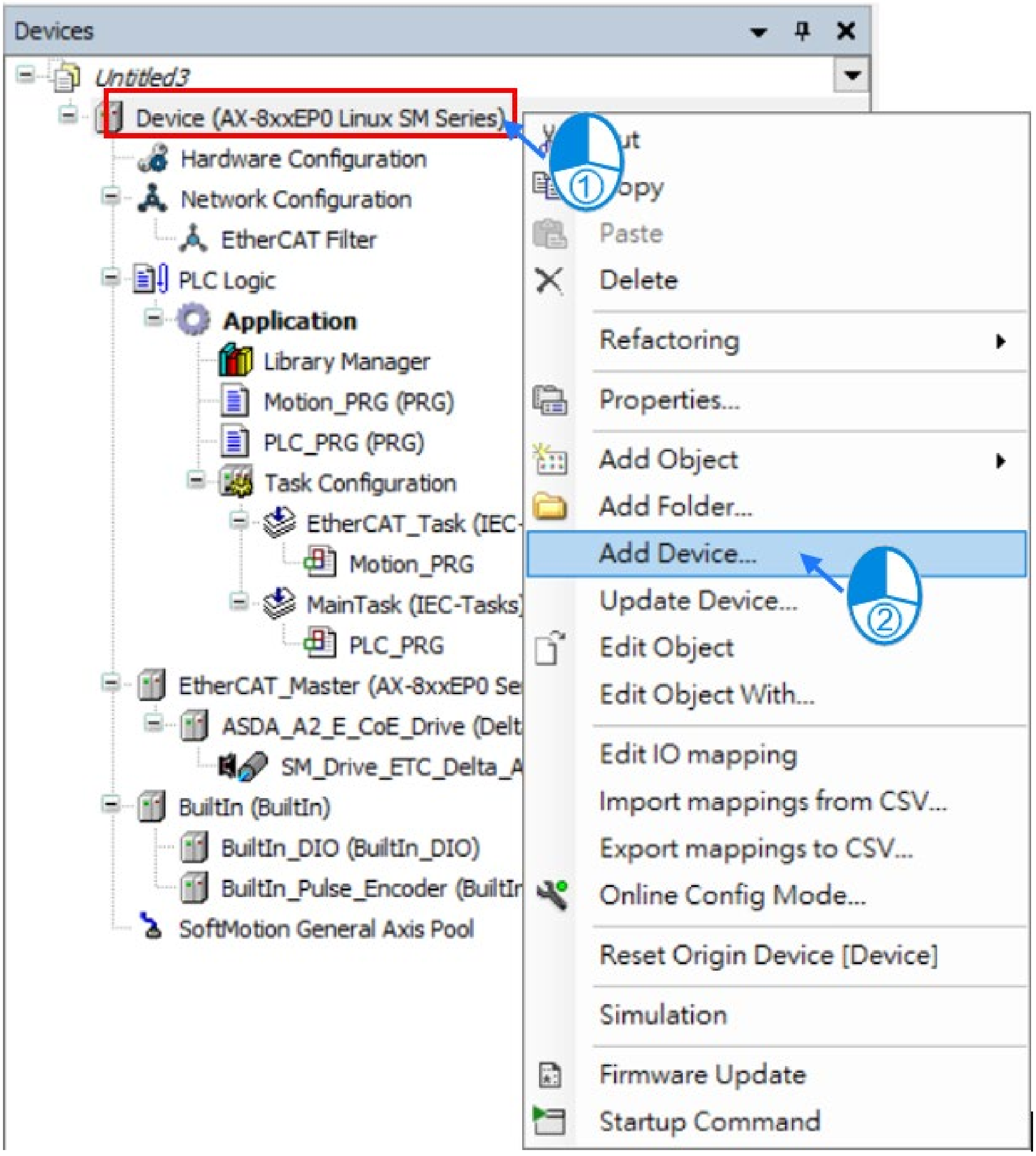


9.2 Introduction to Modbus Serial Communication

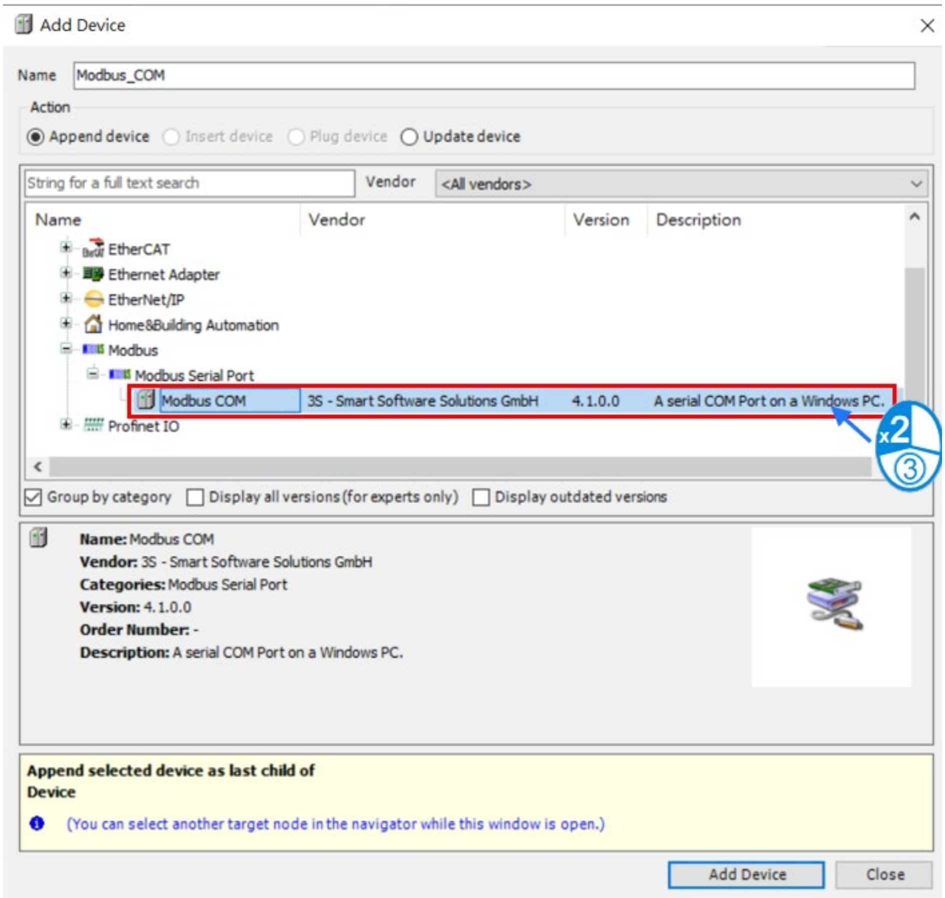
9.2.1 Modbus Serial Port

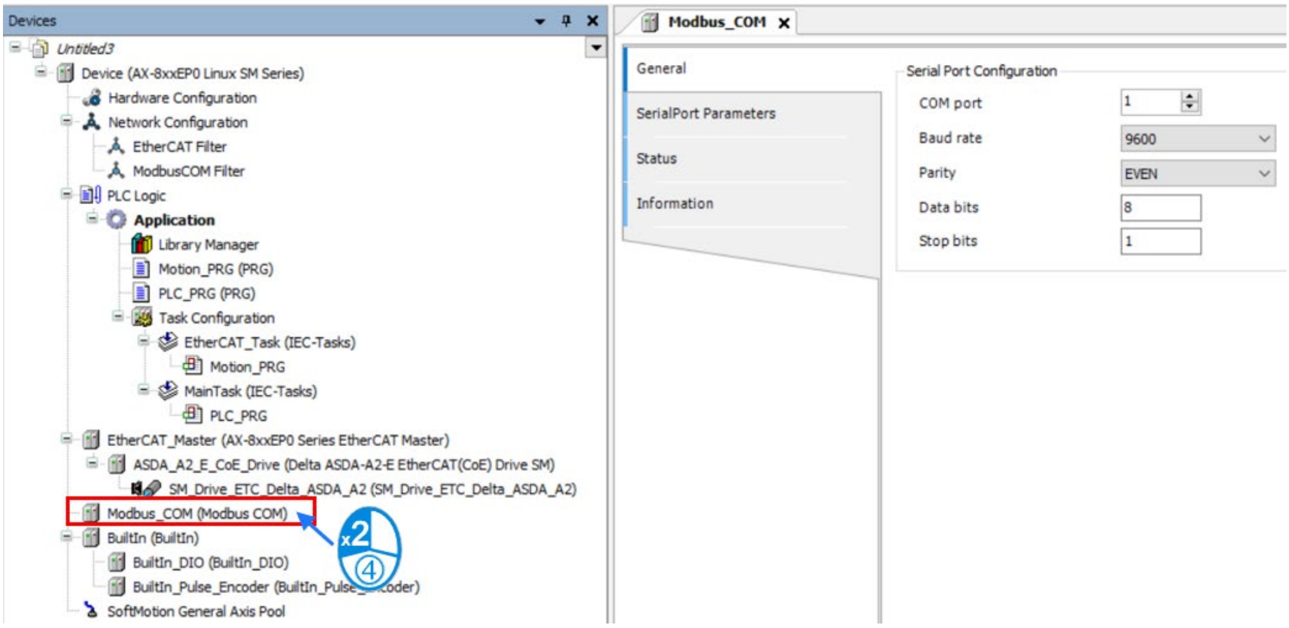
Related parameters of serial communication can be modified by adding new Modbus Serial Port. AX-8 Series supports the following Modbus network types, including one RS-422 and one RS-485, and each Modbus Serial Port allows one master. Follow the below section to set up the basic settings for communication via the serial port for the Modbus serial port.

9.2.1.1 Adding Delta Modbus COM

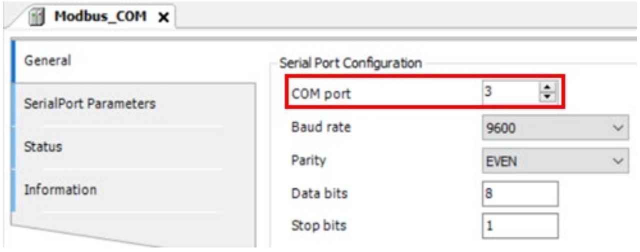


Find **Modbus COM** (Modbus > Modbus Serial Port > Modbus COM) and then double-click it or click **Add Device** to add this port in.

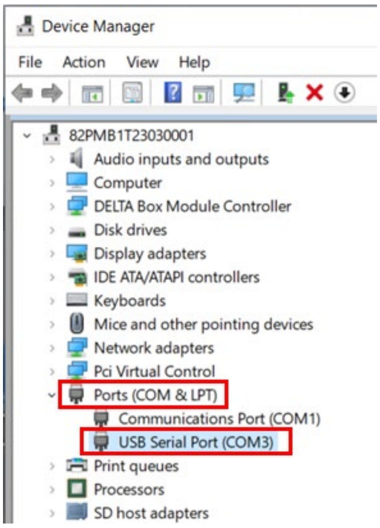




- **Modbus Serial COM Port Number**



For AX-8 with the Windows operating system, the setting value for **COM port** must be the same as the port number in Device Manager of the Windows system.



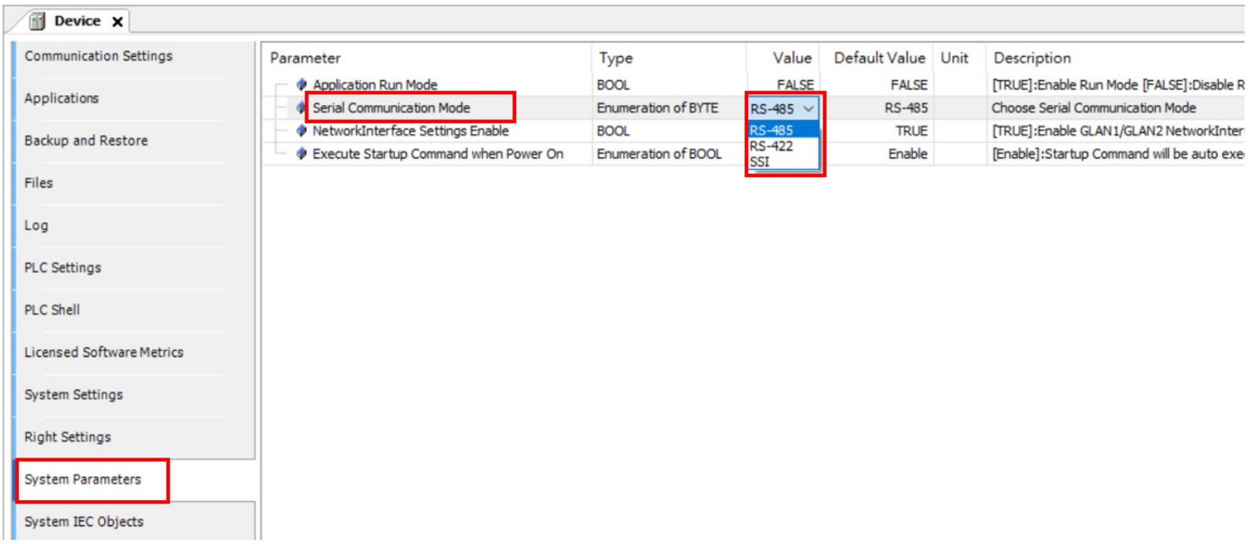
For AX-8 with the Linux operating system, the setting for **COM port** is fixed to 3 when the RS-485 or RS-422 port is used. If a USB-to-COM port device is used, the setting for **COM port** varies with different extension devices. If it is identified as a ttyACM device, the port number is in the range 8 to 11, in the order of device inserting. If it is identified as a ttyUSB device, the port number is in the range 4 to 7, in the order of device inserting.

9

9.2.1.2 Setting up Delta Modbus COM

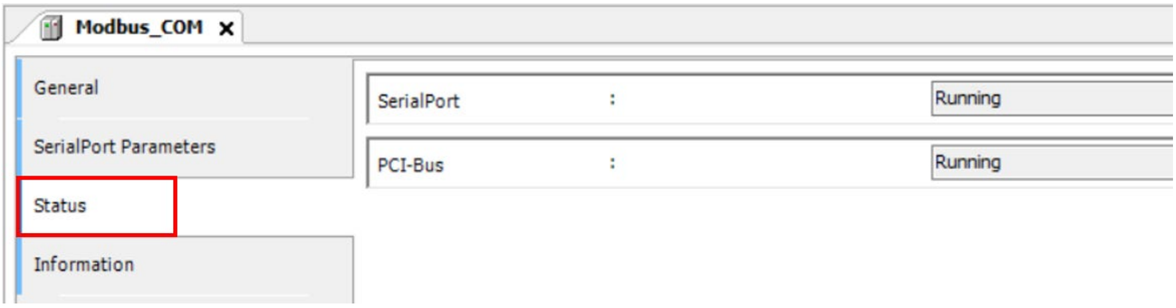
- **System Parameters**

Here you can configure Serial Port Parameters. Settings include Serial Communication Mode (RS-422/RS-485).



- **Status**

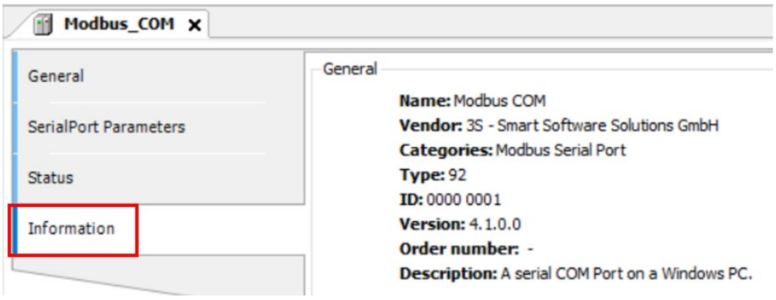
Here you can find the device status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device, also information about the card used and the internal bus system.



Item	Description
SerialPort	The status of Modbus Serial Communication

- **Information**

Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.

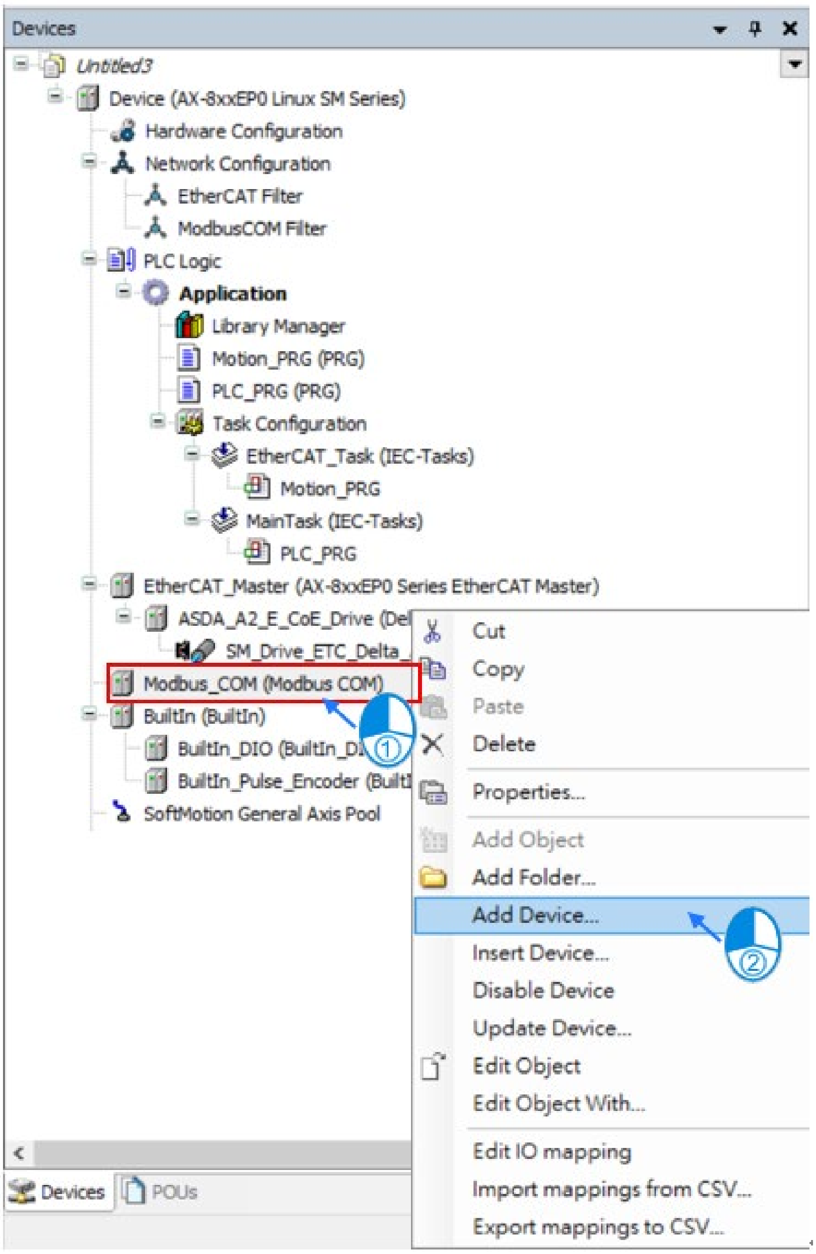


9.2.2 Modbus Serial Master

In addition to providing the standard Modbus communication protocol, the AX-8 Series PLC further executes the Delta controller internal device conversion (X, M, D devices, etc.), eliminating the need for you to check the conversion table. When AX-8 is set to act as a Modbus Serial Master, you need to first create Modbus Master COM port and then add Modbus Slave COM port to continue further settings. Follow the below section to set up the Modbus Serial Master.

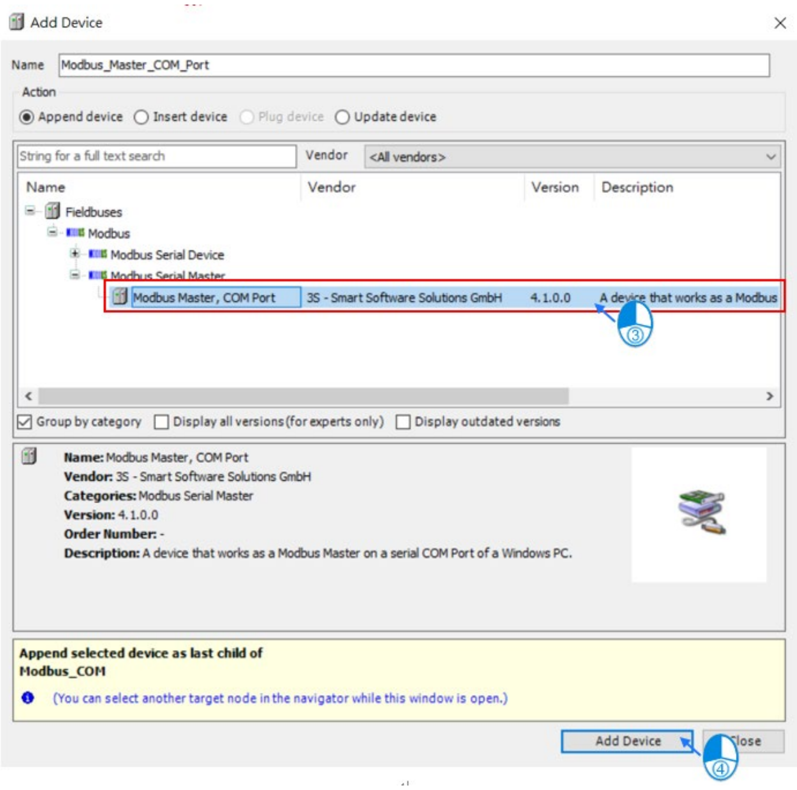
9.2.2.1 Adding Delta Modbus Master/Slave COM

1. Right-click the created **Modbus_COM (Modbus COM)** in the tree view to open a context menu. And click **Add Device**…to open the Add Device setting window.

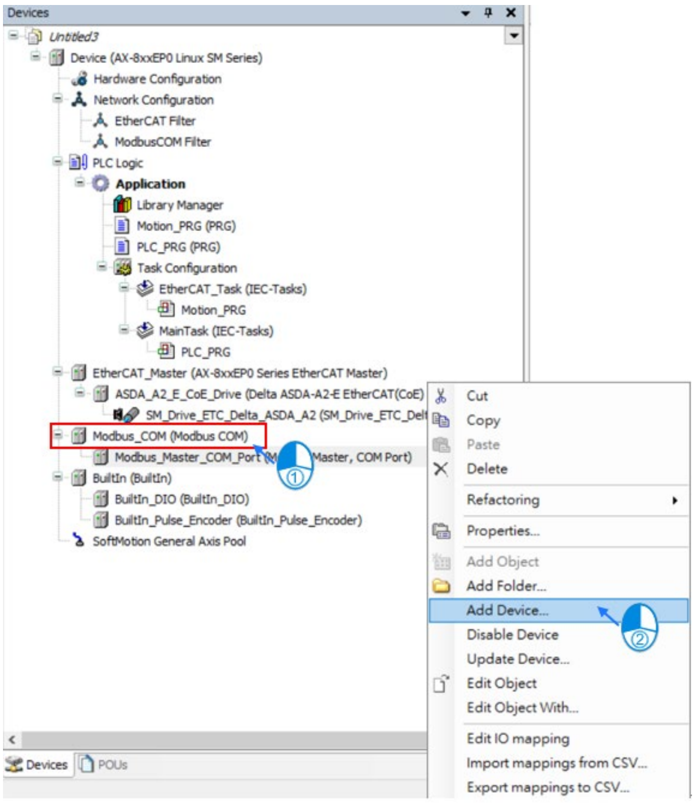


9

2. Find and double-click **Modbus Master COM Port** (Fieldbuses > Modbus > Modbus Serial Master > Modbus Master COM Port) or click **Add Device** to add this port in.

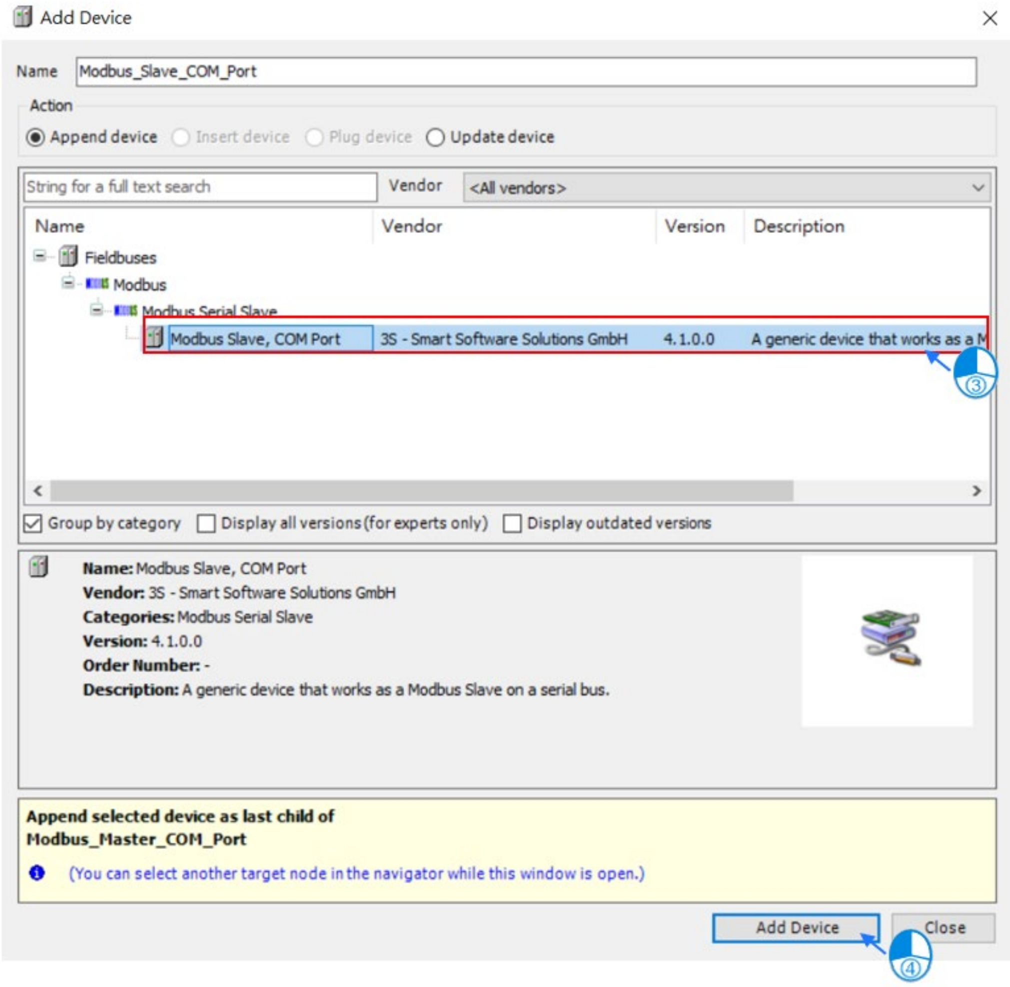


3. Find the added port **Modbus_Master_COM_Port (Modbus Master COM Port)** in the tree view and double-click it to open the setting window to set up.



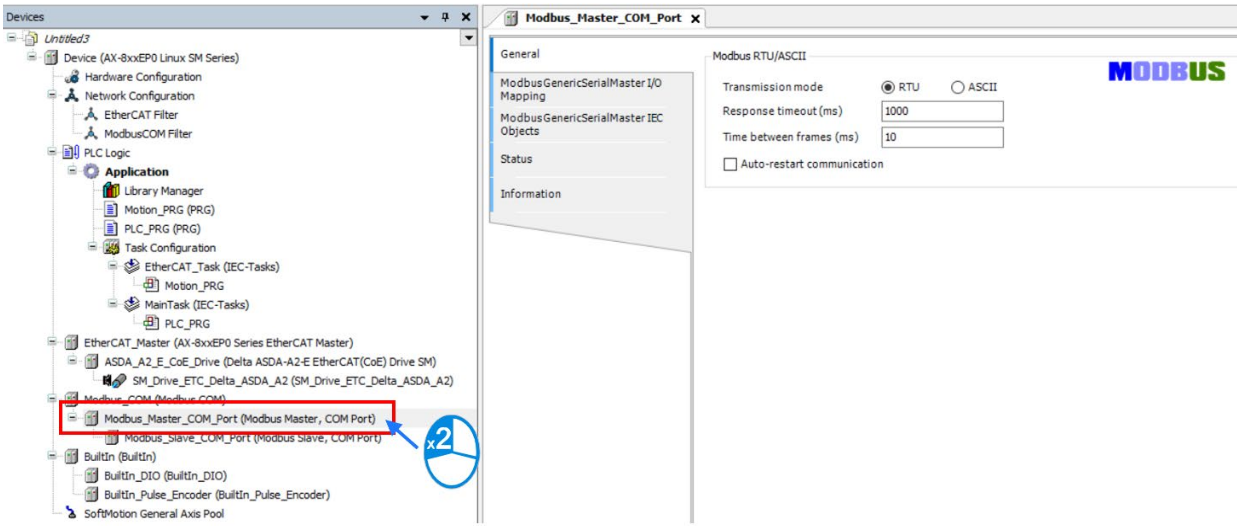
9

4. Find and double-click **Modbus Slave COM port** (Fieldbuses > Modbus > Modbus Serial Slave > Modbus Slave COM Port) or click **Add Device** to add this port in.



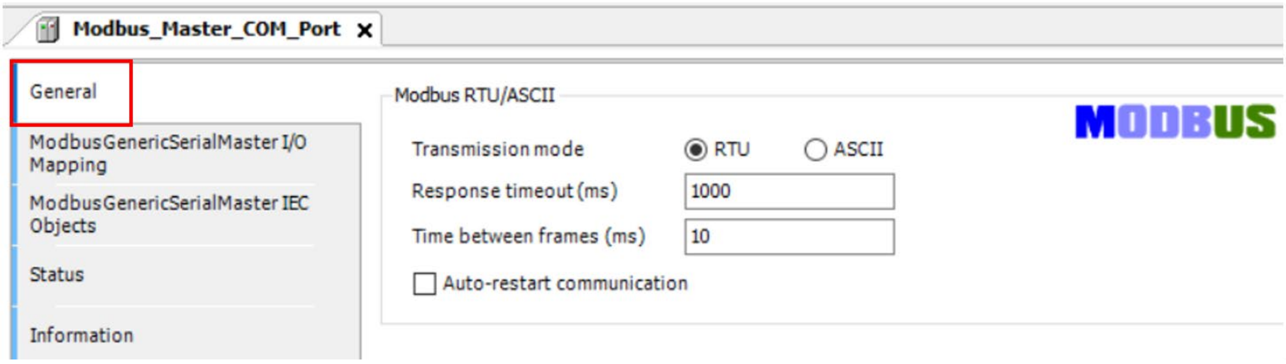
9.2.2.2 Setting up Modbus Serial Master

Double-click the created **Modbus Master COM Port** to open the setting menu.



- **General**

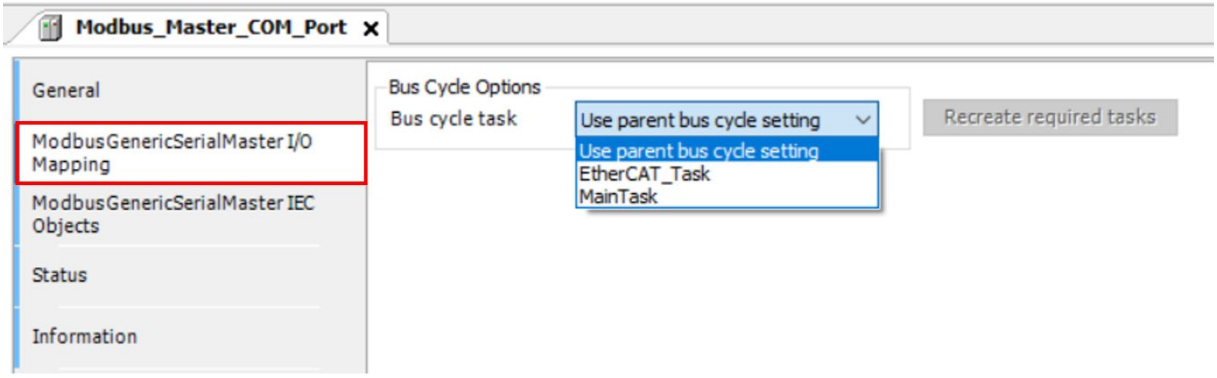
Here you can configure the basic settings for Modbus Master COM Port.



Item	Description
Auto-restart communication	Enable this option to have this port to reconnect automatically if an error occurs or connection timeout occurs.

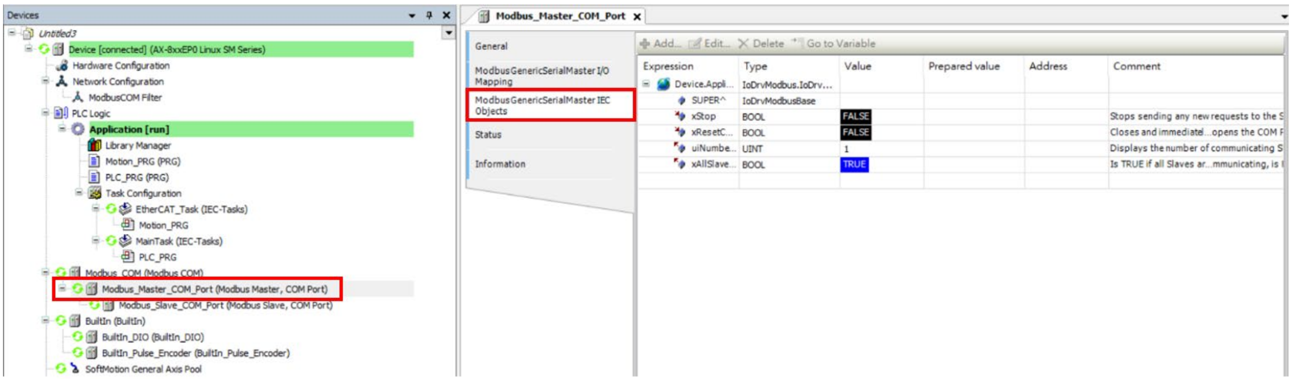
- **Modbus GenericSerial Master I/O Mapping**

Here you can select the bus cycle task of Modbus Serial Master to synchronize with the Modbus communication time. Refer to **section 4.2.1.6 PLC Settings** for more information.



- **Modbus Serial Master IEC Objects**

Here you can check the status of Modbus Serial Master under this tab.

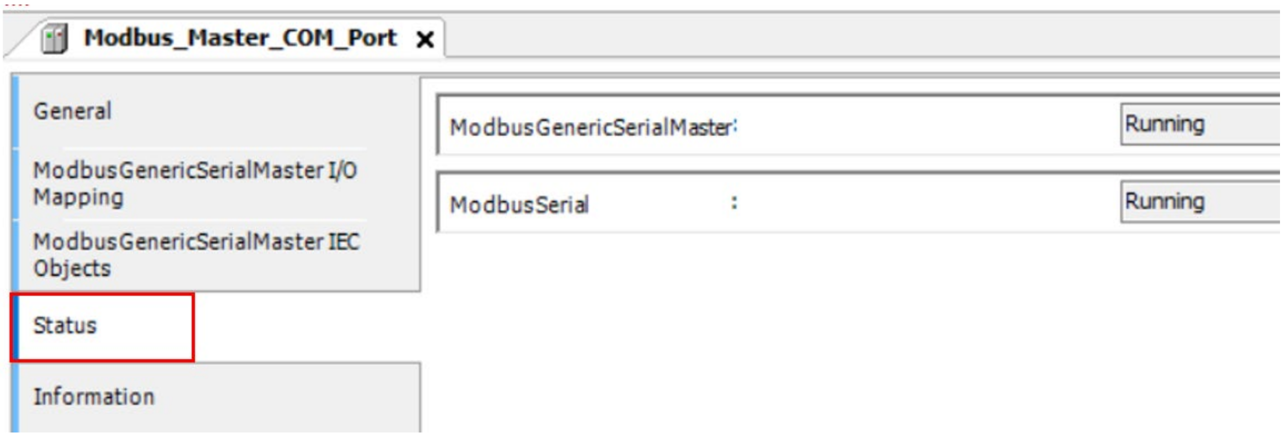


Expression	Description
bStop	Stop sending the Slave any new request
bResetComPort	Reset the COM port
uiNumberOfCommunicatingSlaves	The number of the Slaves that are in communication
bAllSlavesOk	The communication status of the Slave

9

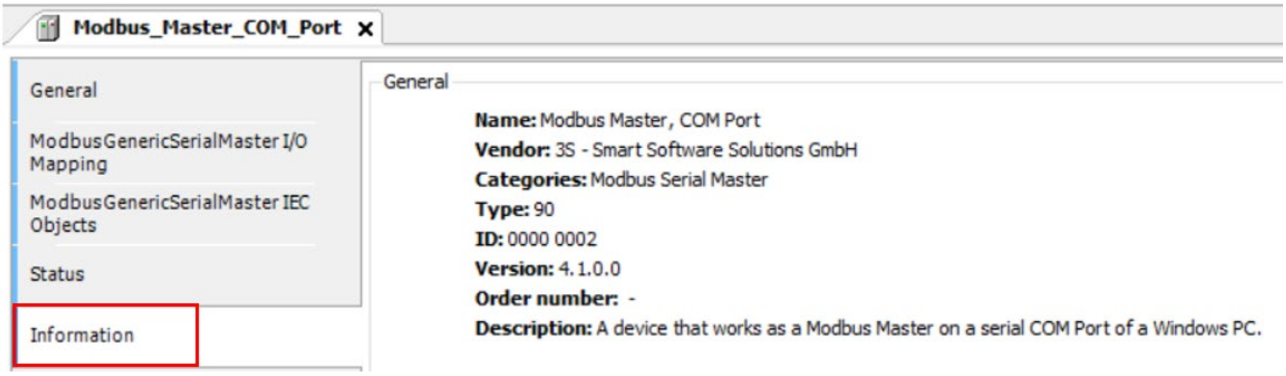
- **Status**

Here you can find the device status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device, also information about the card used and the internal bus system.



- **Information**

Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.

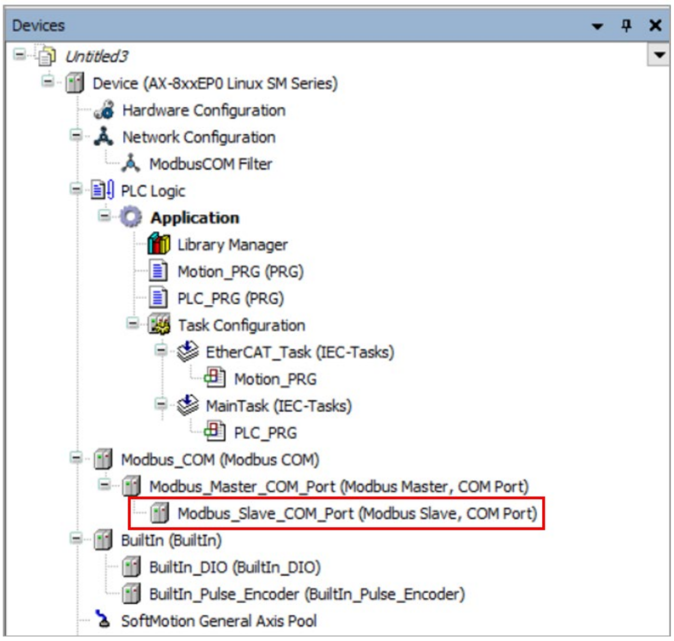


Note:

1. Each Modbus Serial Port allows one master. A maximum of 32 slaves can be attached to a master.
2. Since RS-422 has no multipoint capability, only point-to-point connection is possible. And only the FIRST slave can communicate with the master. Since RS-485 has multipoint capability, RS-485 does NOT have such limitations.

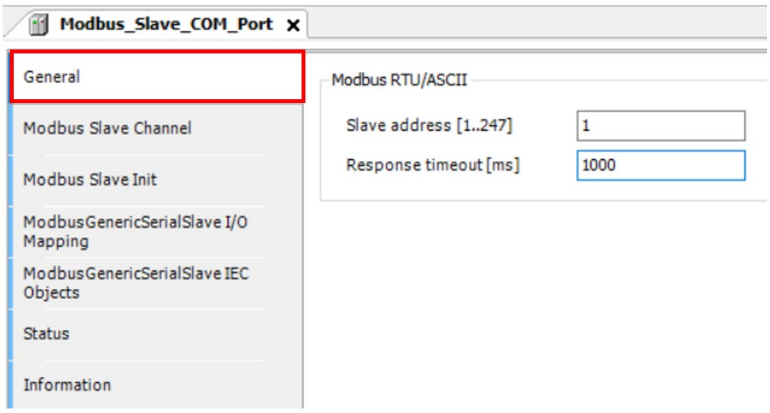
9.2.2.3 Setting up Modbus Serial Slave

In the tree view, find the added port **Modbus_Slave_COM_Port (Modbus Slave COM Port)**. Double-click it to open the setting window to set up.



- **General**

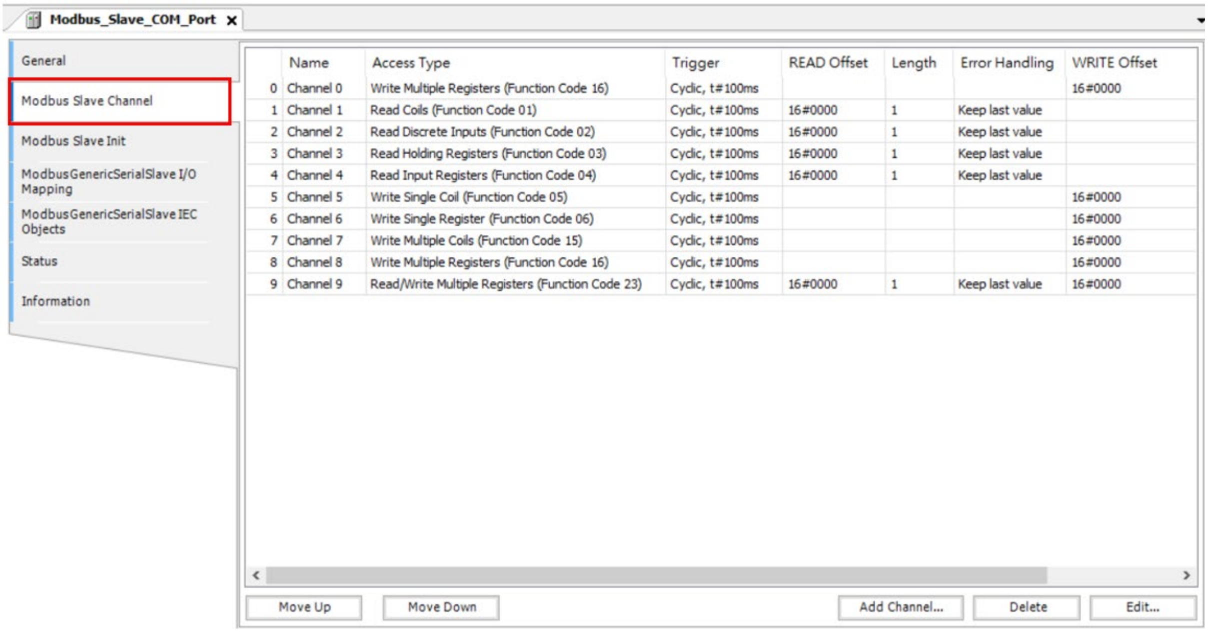
Here you can configure the basic settings for Modbus Slave COM Port, such as Slave Address, Response Timeout and Device Type.



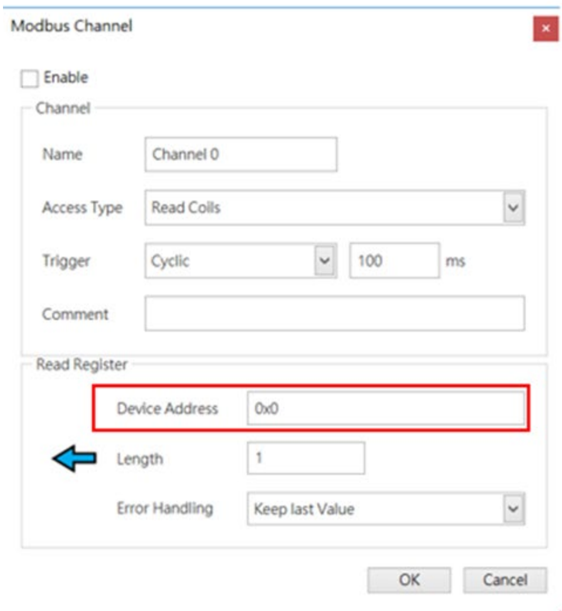
Item	Description
Slave Address	Address of a serial Modbus device
Response Timeout	Time interval for the master to wait for the response from the slave. This is especially configured for this slave node and overwrites the general response timeout setting of the respective master.

- **Modbus Slave Channel**

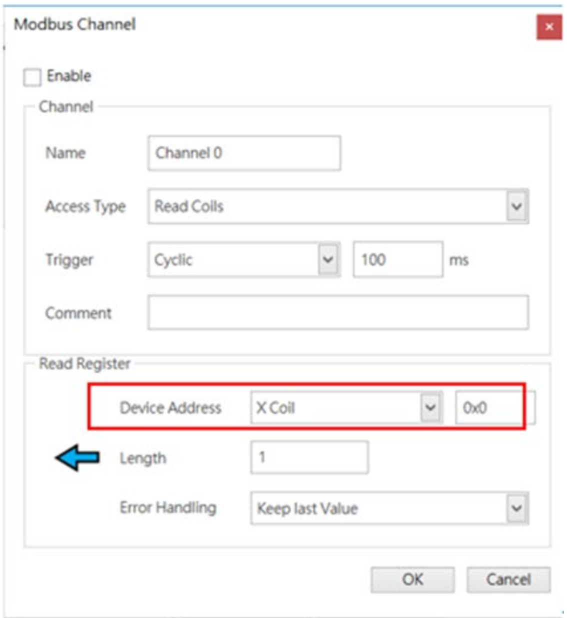
Here you can define slave channels. Each channel represents a single Modbus request. You can create up to 10 channels for each slave. AX-8 Series PLC will send out Modbus request packets in chronological order. All channels share the same Modbus connection.



Device Type：Standard Modbus Device



Device Type：AH Series



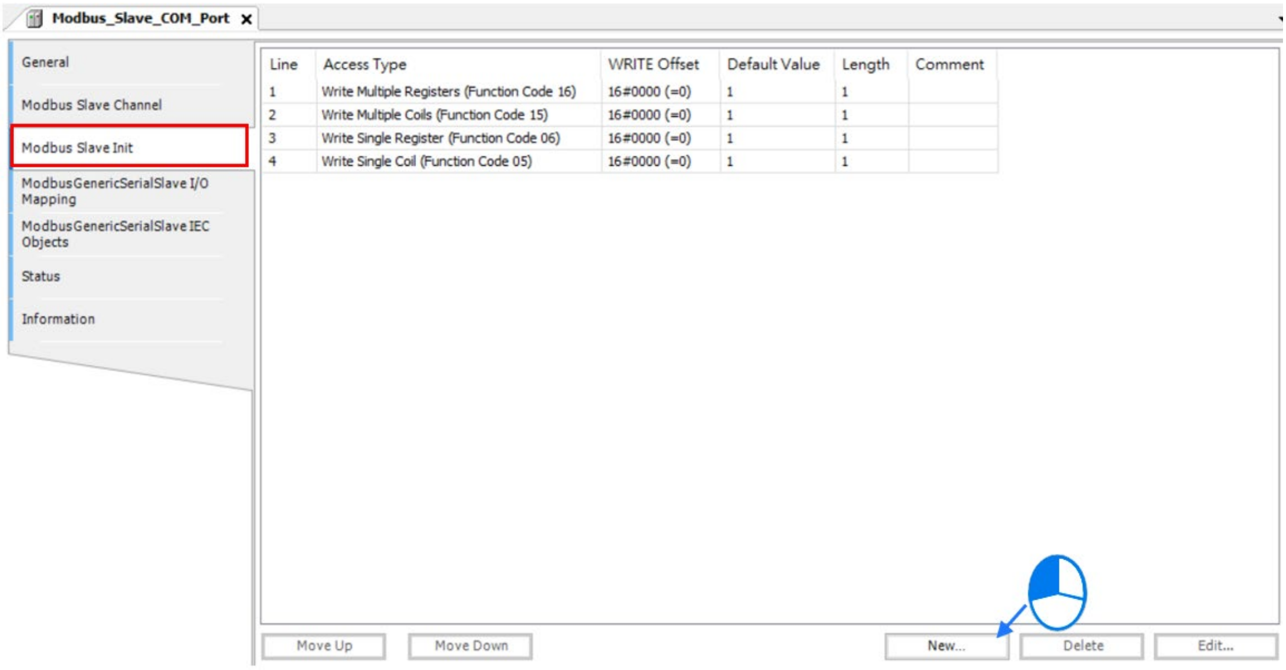
9

<table>
<tr><th>Item</th><th colspan="2">Description</th></tr>
<tr><th>Device Type</th><th>Standard Modbus Device</th><th>Delta Series Device</th></tr>
<tr><td>Enable</td><td colspan="2">Activates this channel</td></tr>
<tr><td>Name</td><td colspan="2">Defines this channel name</td></tr>
<tr><td>Access Type</td><td>Modbus function code
● Read Coils (0x01)
● Read discrete inputs (0x02)
● Read holding Registers (0x03)
● Read input Registers (0x04)
● Read single Coil (0x05)
● Write single Register (0x06)
● Write multiple Coils (0x0F)
● Write multiple Registers (0x10)
● Read/Write multiple Registers (0x17)</td><td>Read/Write Registers
● Read Coils
● Read Registers
● Write Coils
● Write Registers

Note: PLC uses the corresponding Modbus function code according to the read/write register of the device type.</td></tr>
<tr><td>Trigger</td><td>● Cyclic: The request occurs periodically.
● Rising edge: The request occurs as a reaction to a rising edge of the Boolean trigger variables. The trigger variable is defined in the tab I/O Mapping.
● Application: The Modbus request is triggered by DFB_ModbusComChannel</td><td>● Cyclic: The request occurs periodically.
● Rising edge: The request occurs as a reaction to a rising edge of the Boolean trigger variables. The trigger variable is defined in the tab I/O Mapping.
● Application: The Modbus request is triggered by DFB_ModbusComChannel</td></tr>
<tr><td>Comment</td><td colspan="2">Description of the channel</td></tr>
<tr><td>Device Address</td><td>Modbus protocol address</td><td>Delta register address (will be converted into Modbus protocl in the background)</td></tr>
<tr><td>Length</td><td>Number of the register to be read/written to. (up to 100 coils and 100 registers)</td><td>Number of the register to be read/written to. (up to 256 coils and 100 registers)</td></tr>
<tr><td>Error Handling</td><td colspan="2">What to do with the data in case of a communication error:
● Set To ZERO
● Keep last value</td></tr>
</table>

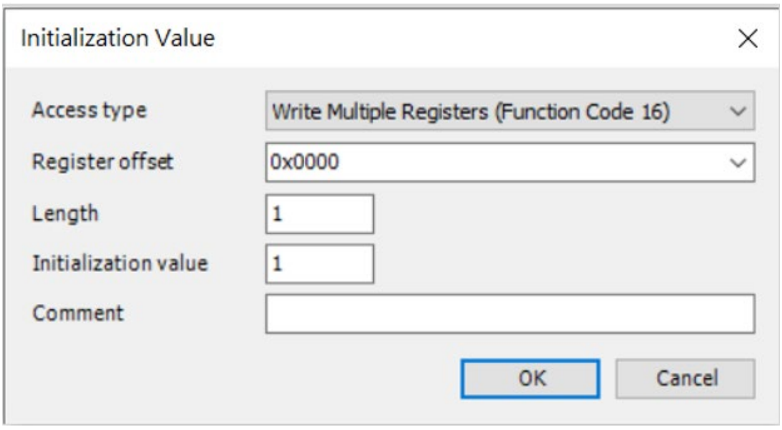
- **Modbus Slave Init**

After the Modbus connection between AX-8 Series PLC and the slaves is established, you can use **Add Channel** button to edit the Initialization Value of the Coil/Registers.



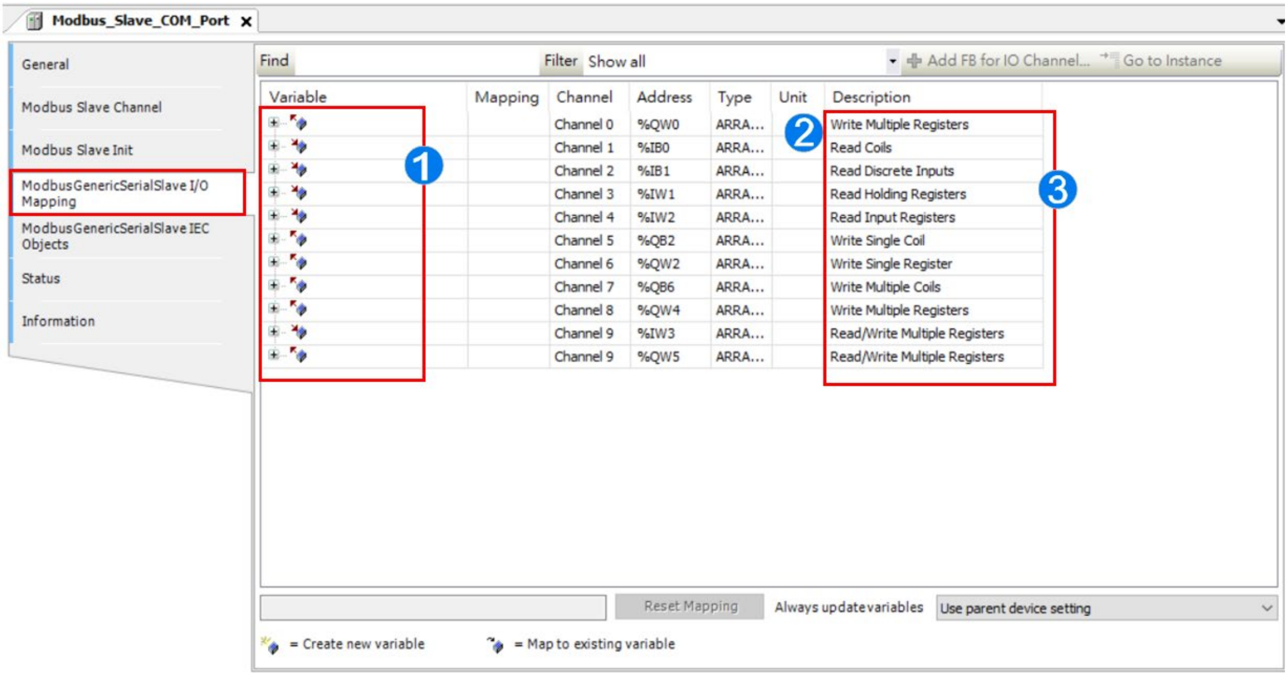
Click **New**, you can edit the Access Type, Device Address, Length, Initialization Value and Comment.

Click **OK** to confirm the settings.



- **Modbus Generic Serial Slave I/O Mapping**

After you have added channels under the tab of **Modbus Slave COM Port**, you can find the variables and the set access types under this tab.



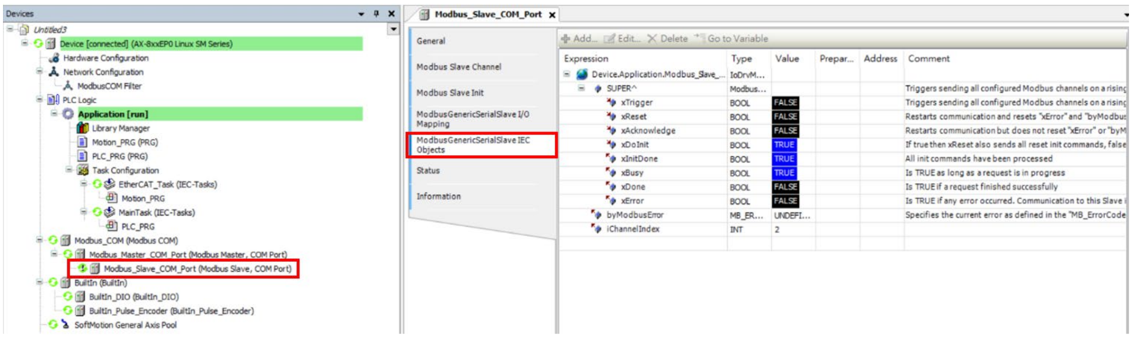
① The descriptions here reflect what you have set for the Access Type in Modbus Slave Channel tab.

② The triggered Boolean variable for this channel. When the Trigger type is set to Rising edge in Modbus Slave Channel

③ The controller Coils/Registers that are read/written by this channel.

- **Modbus Generic Serial Slave IEC Objects**

Here you can check the status of Modbus Serial Slave under this tab.

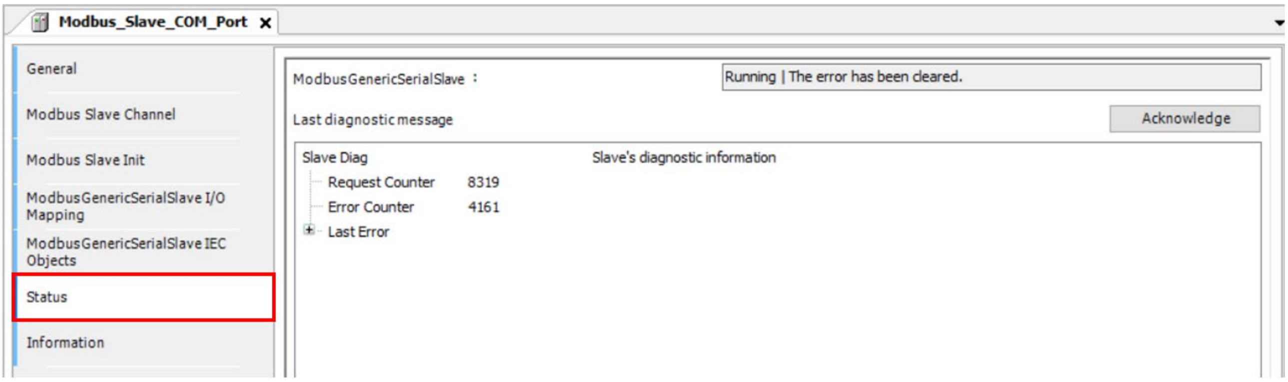


Expression	Description
bTrigger	Trigger all Modbus channels at one time.
bReset	Re-establish the connection and reset bError and ModbusRrror when the connection status shows error. And this function is only available when the option “Auto-Reconnect” is NOT enabled.

Expression	Description
bAcknowledge	Re-establish the connection and continue to execute the data transmission from the Modbus channels that showed error previously. And this function is only available when the option “Auto-Reconnect” is NOT enabled.
bDoInit	Initialized the Slave.
bInitDone	The initialization of the Slave is completed.
bBusy	This channel is in data transmission.
bDone	The data transmission via this channel is completed.
bError	Error occurs when this channel is in data transmission.
ModbusError	Records of the Modbus error.
iChannelIndex	The number of the channel that is in execution.

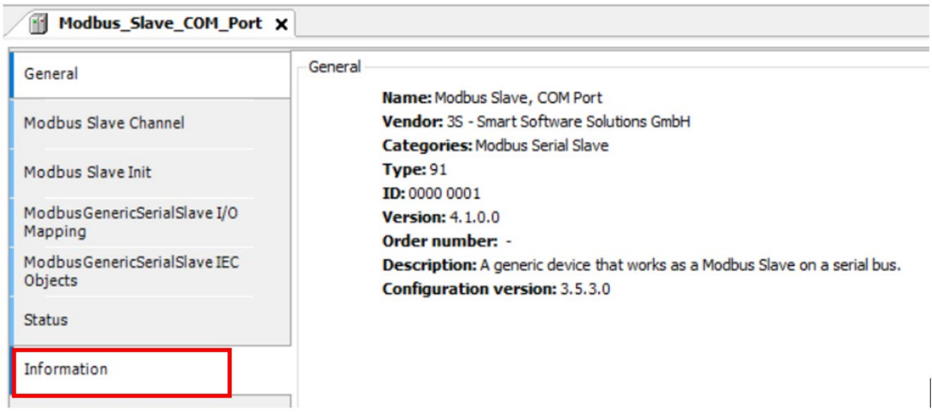
- **Status**

Here you can find the Modbus Slave COM Port status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device, also information about the card used and the internal bus system.



- **Information**

Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.

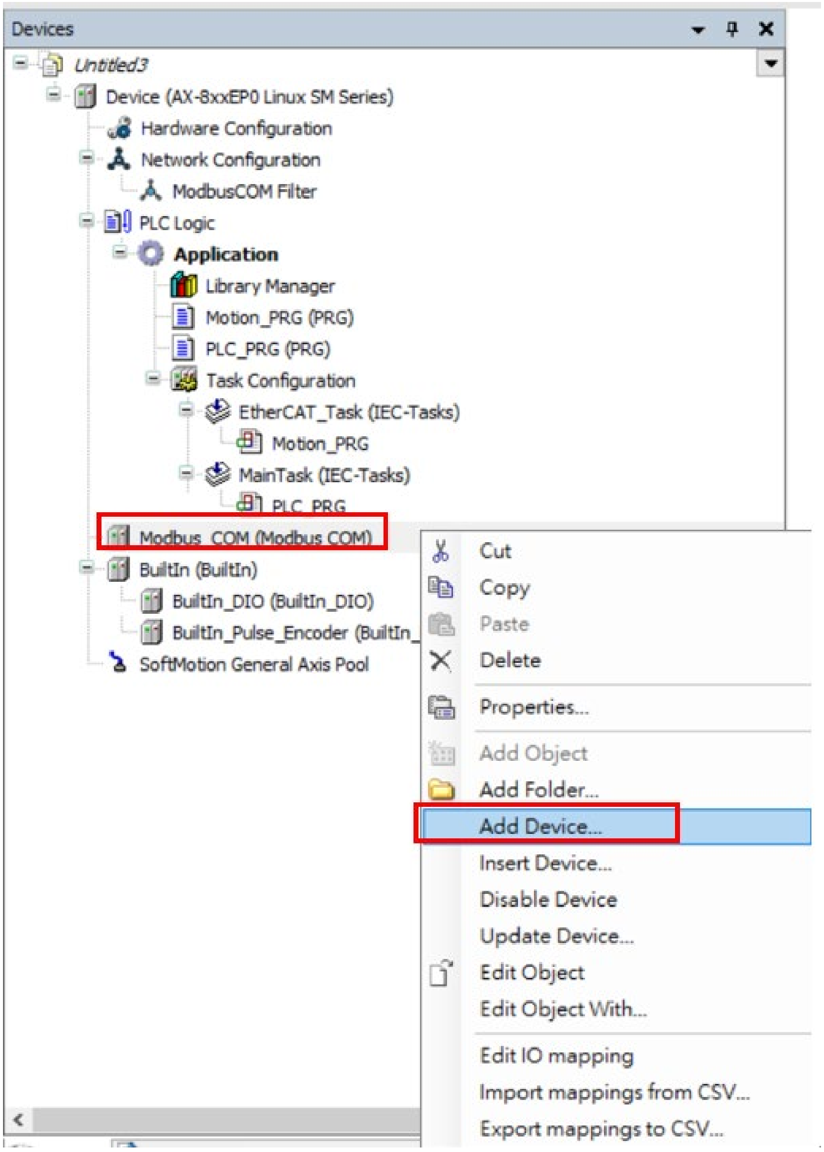


9.2.3 Modbus Serial Slave

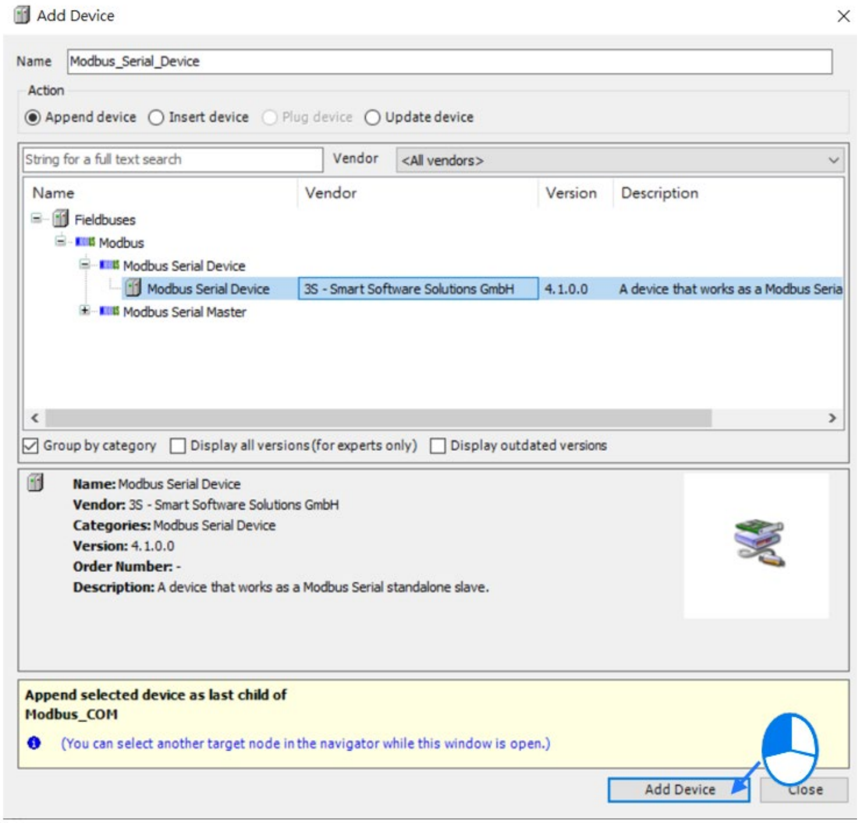
When AX-8 Series PLC is set to act as a Modbus Serial Slave, you need to add Modbus Serial Device in and set up the allowable areas for Coils/Register. If Modbus Serial Master uses Delta device communication protocol, there is no access restrictions. Follow the below section to set up the Modbus Serial Slave.

9.2.3.1 Adding a Modbus Serial Device

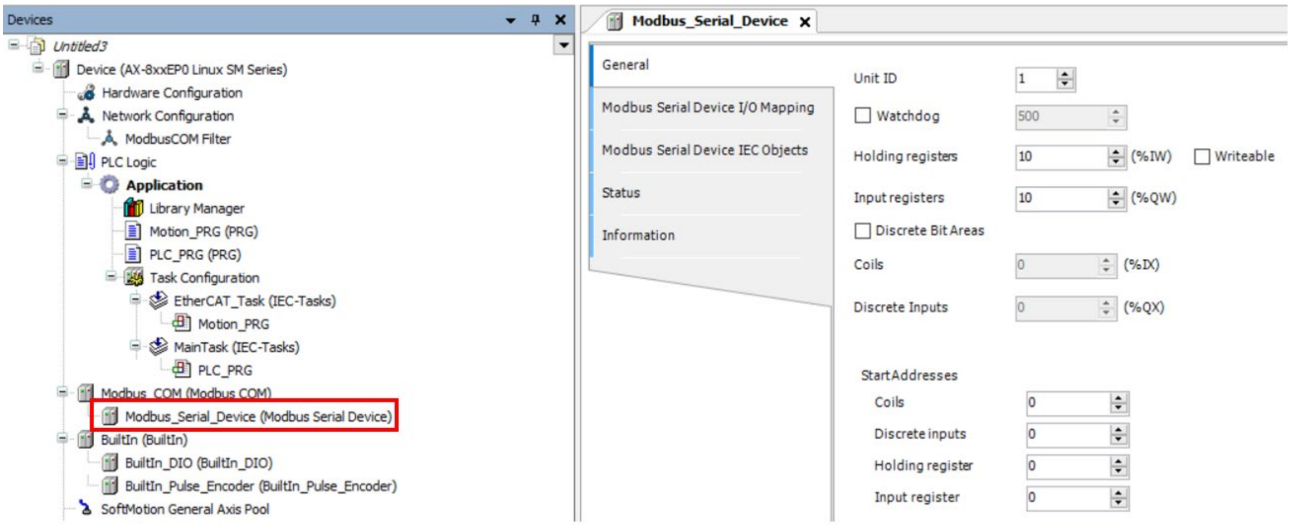
1. Right-click the created **Modbus_COM (Modbus COM)** in the tree view to open a context menu. And click **Add Device**…to open the Add Device setting window.



2. Find and double-click **Modbus Serial Device** (Fieldbuses > Modbus > Modbus Serial Device > Modbus Serial Device) or click **Add Device** to add this port in.



3. Find the added port **Modbus_Serial_Device (Modbus Serial Device)** in the tree view and double-click it to open the setting window to set up.

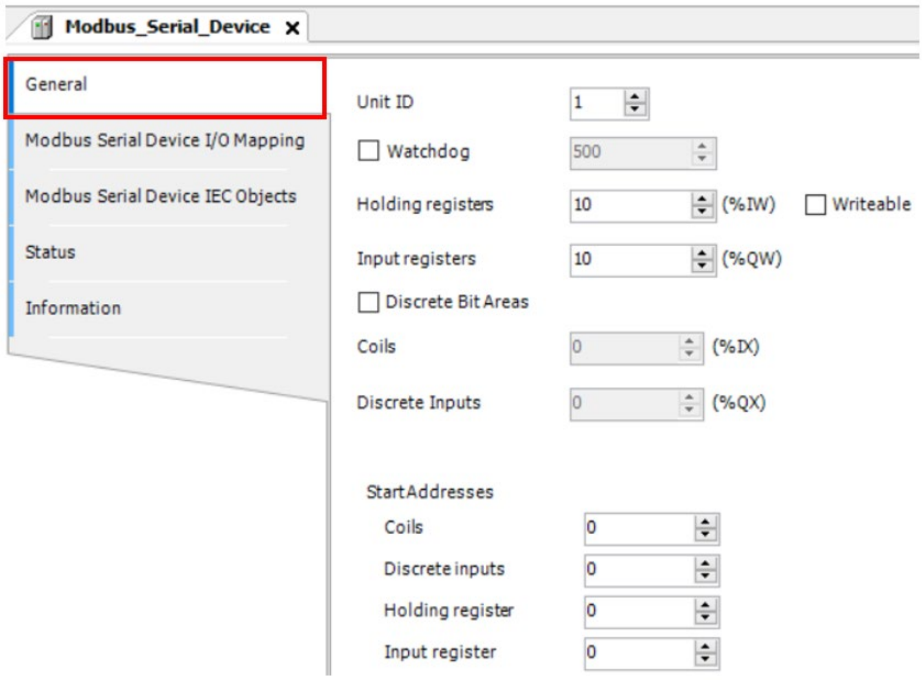


9

9.2.3.2 Setting up the Modbus Serial Device

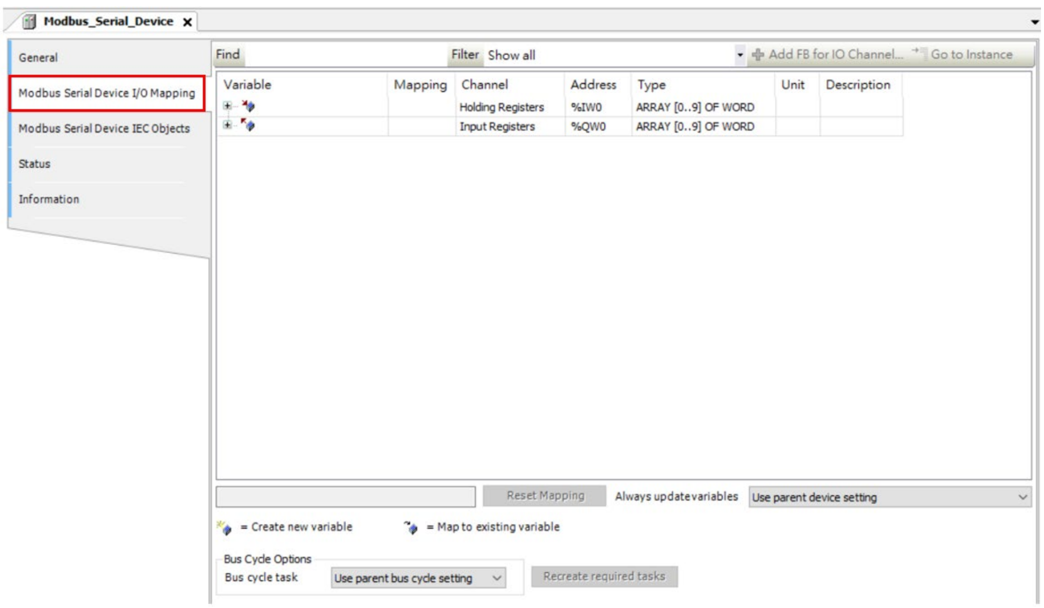
- **General**

Here you can configure the basic settings for Modbus Serial Device. Set up the allowable areas for Coils/Registers. If Modbus Serial Master uses Delta device communication protocol, there is no access restrictions.



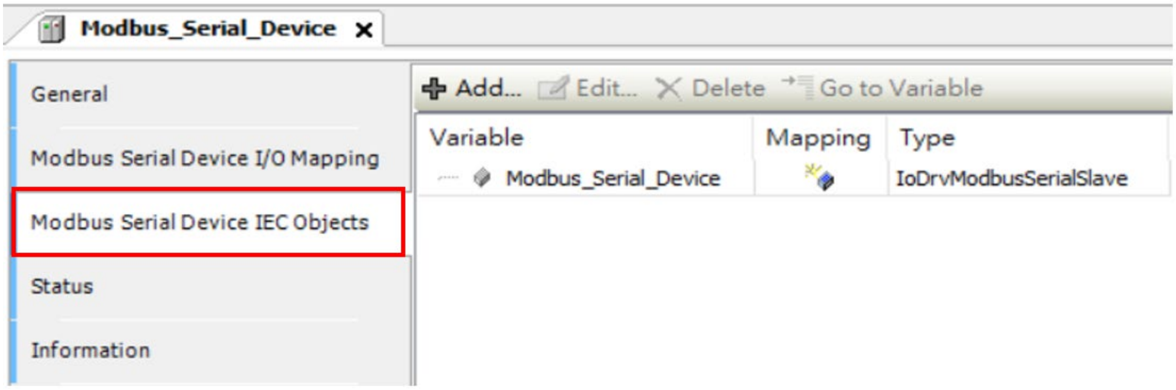
- **Modbus Serial Device I/O Mapping**

Here you can select the bus cycle task of Modbus TCP Slave Device to synchronize with the Modbus communication time. Refer to **section 4.2.1.6 PLC Settings** for more information.



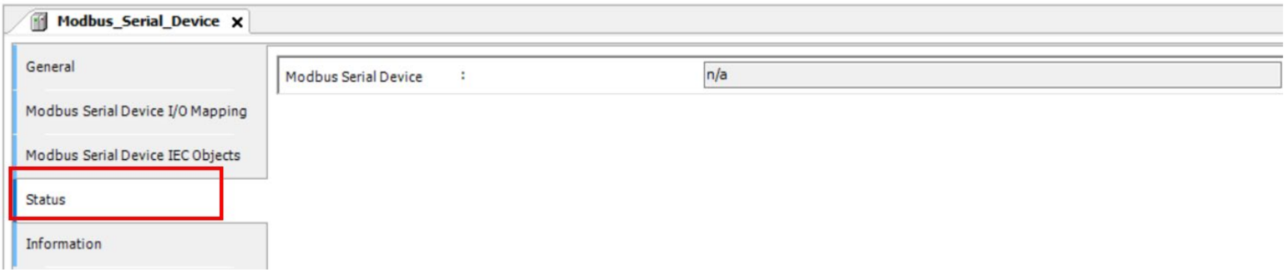
- **Modbus Serial Device IEC Objects**

Here you can check the status of Modbus Serial Device under this tab.



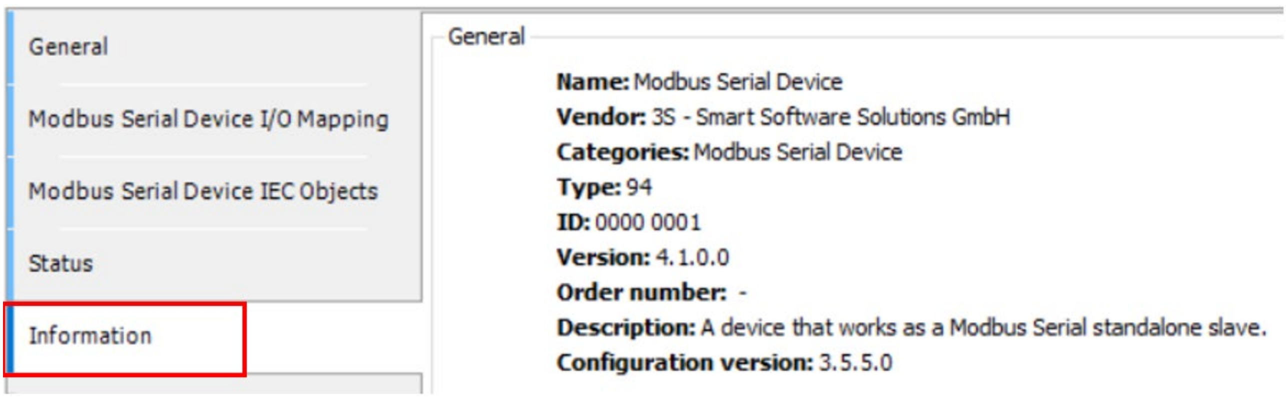
- **Status**

Here you can find the Modbus Serial Slave Device status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device, also information about the card used and the internal bus system.



- **Information**

Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.

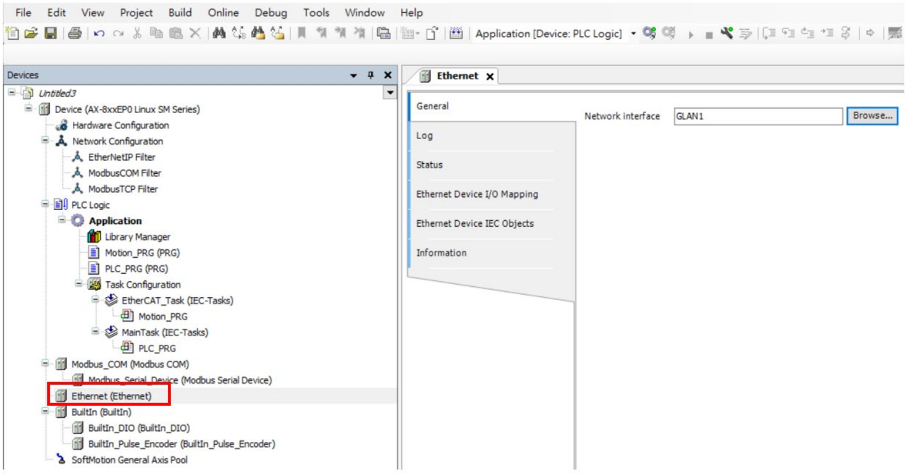


9.3 Introduction to Ethernet Communication

Network-related parameters can be modified by adding new Ethernet device. All functions related to network must be adjusted under this tab (e.g. Modbus TCP and EtherNet/IP). Follow the below section to set up the basic settings for communication via the Ethernet Adapter.

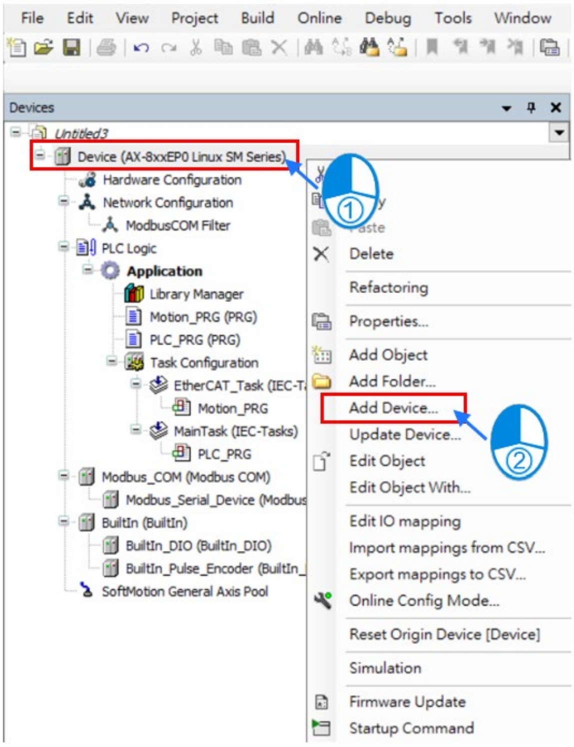
9.3.1 Ethernet

- **Ethernet tree diagram**



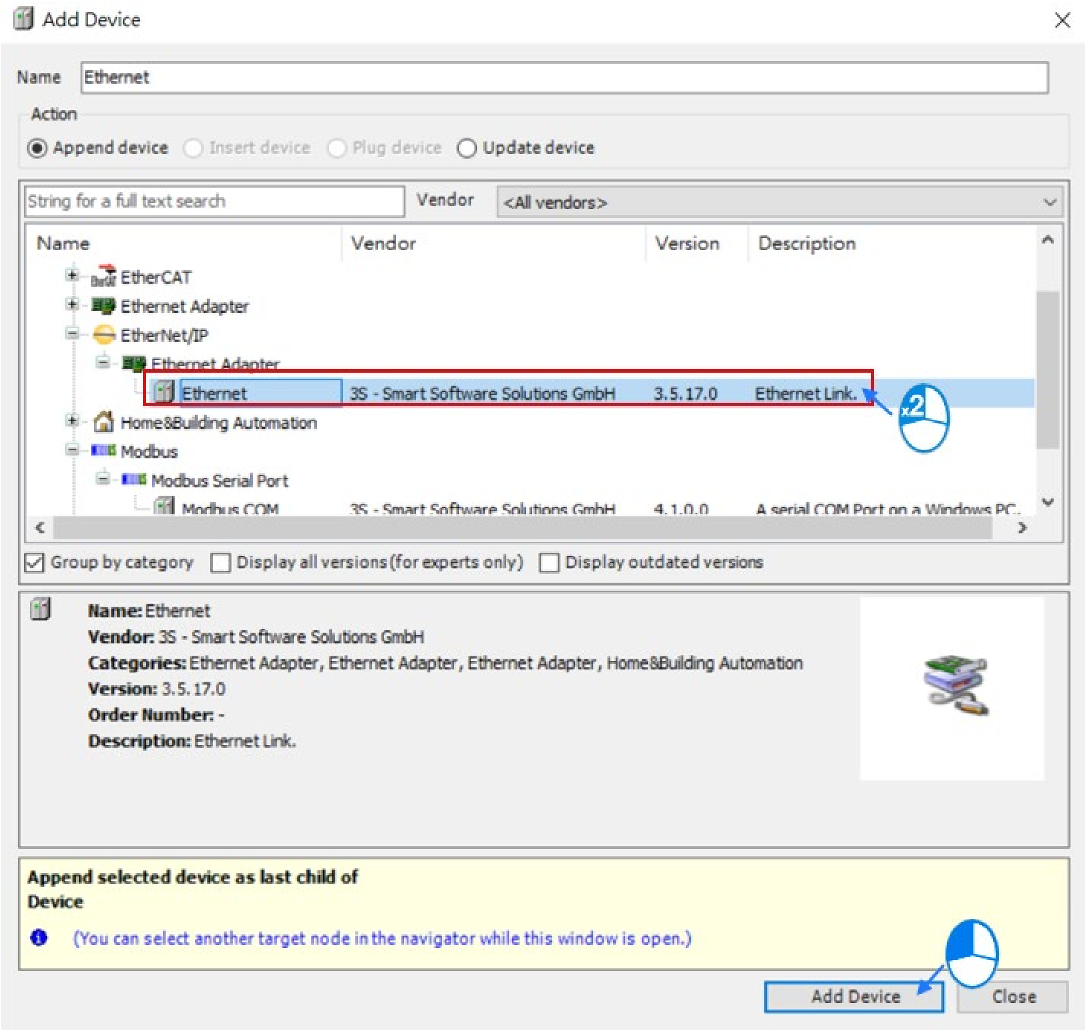
9.3.1.1 Adding an Ethernet Adapter Device

1. Right-click the PLC in the tree view to see a context menu. Click **Add Device**…to open the setting window.



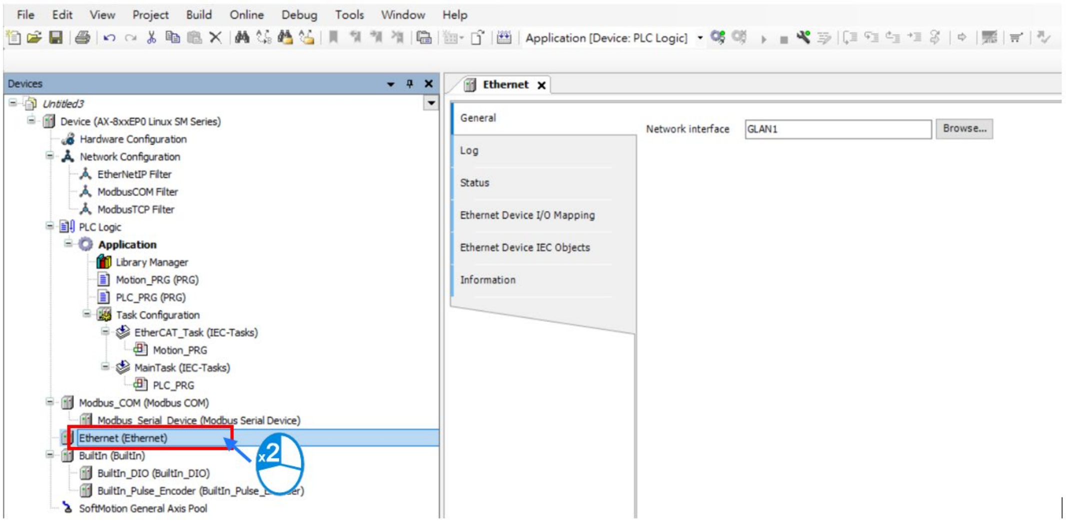
9

2. Find and double-click **Ethernet** (Fieldbuses > Ethernet Adapter > Ethernet) or click **Add Device** to add it in.

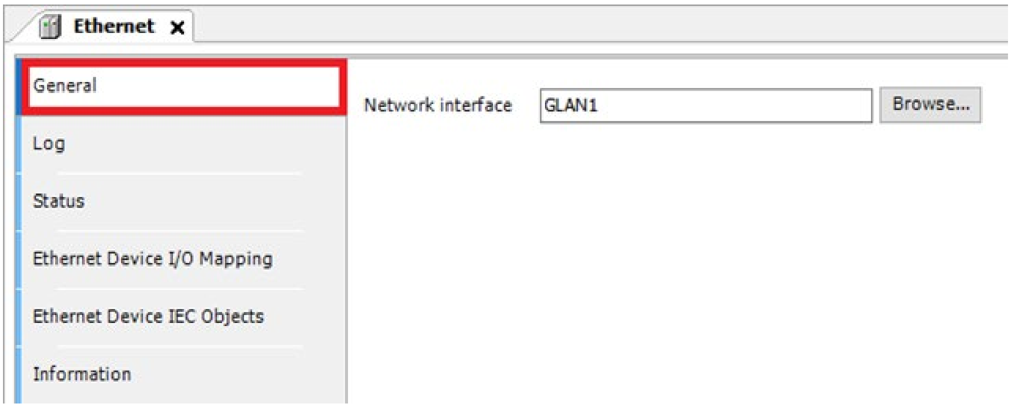


9.3.1.2 Setting up the Ethernet

Find the added **Ethernet (Ethernet)** in the tree view and double-click it to open the setting window for setup.



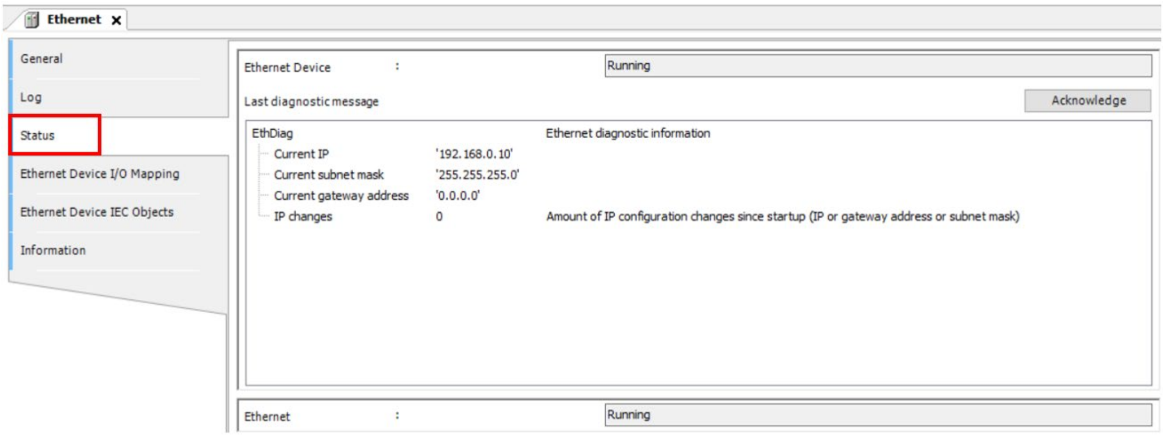
- **General**



Item	Description
Network Interface	Current communication interface

- **Status**

Here you can find the device status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device.

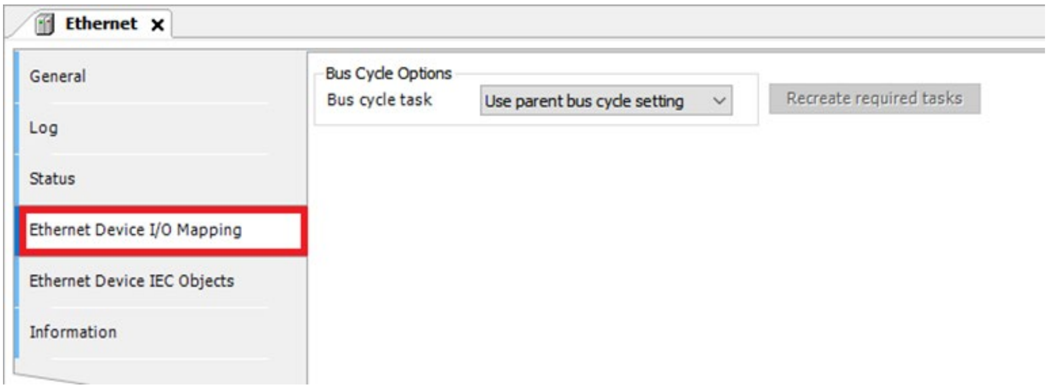


Item	Description
Ethernet Device	The status of Ethernet Communication
Last Diagnostic Message	Network diagnosis

9

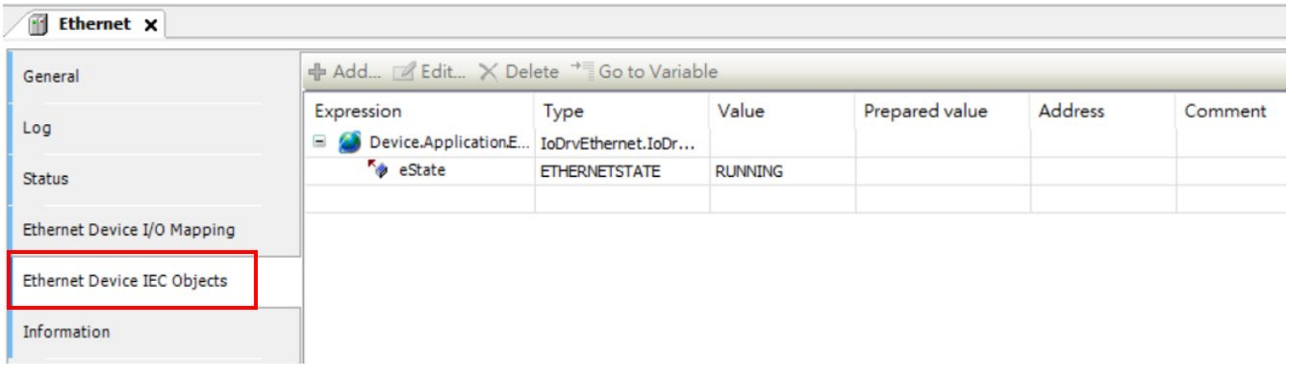
- **Ethernet Device I/O Mapping**

Here you can select the bus cycle task of Ethernet Device to synchronize with the communication time. Refer to **section 4.2.1.6 PLC Settings** for more information.



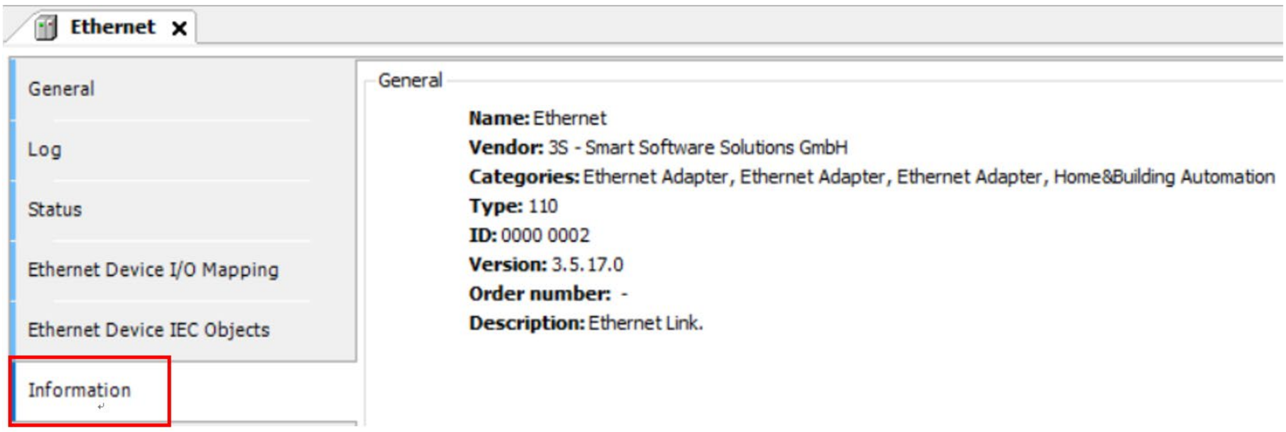
- **Ethernet Device IEC Objects**

Here you can find the objects defined by Ethernet Device. “Objects” are listed that allow for access to the device from the IEC application. In online mode, you can use the table of IEC objects as a monitoring view.



- **Information**

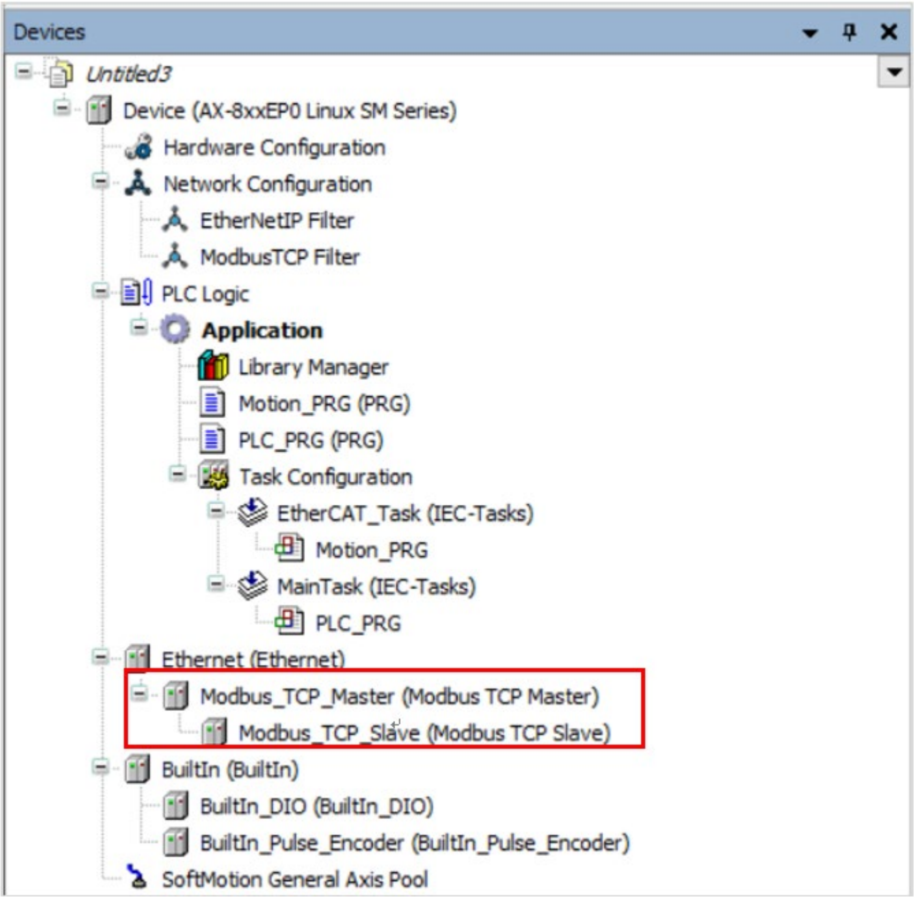
Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.



9.3.2 Modbus TCP Master

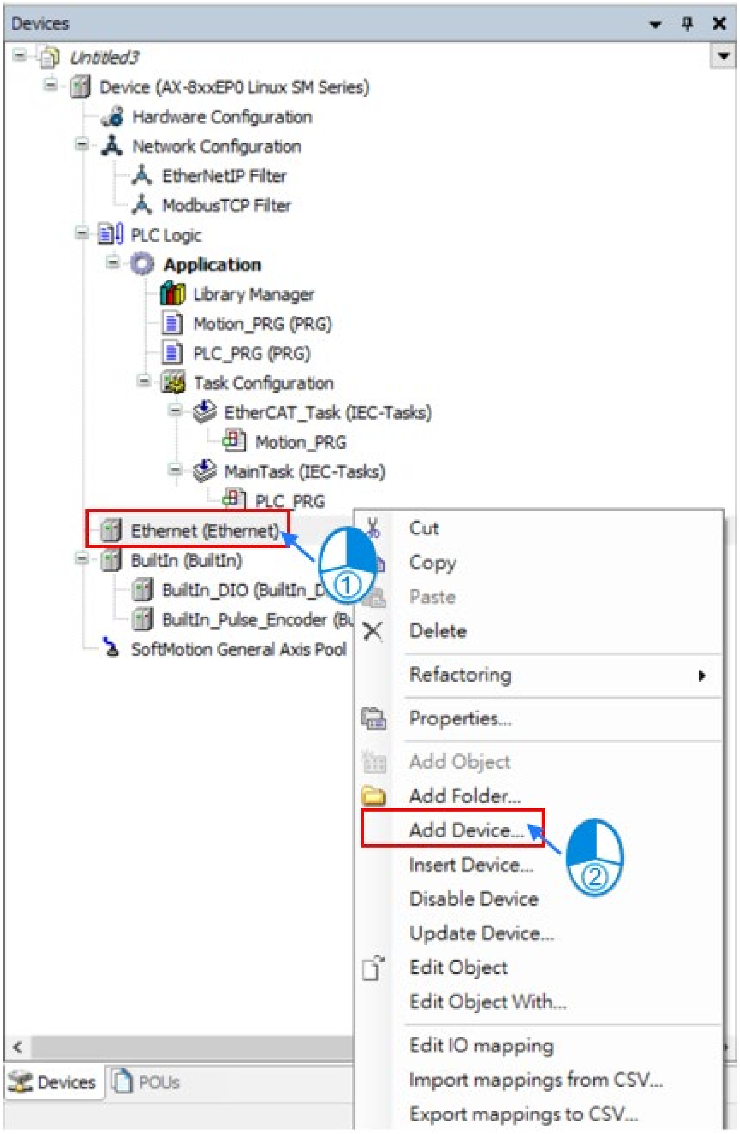
In addition to providing the standard Modbus communication protocol, the AX-8 Series PLC further executes the Delta controller internal device conversion (X, M, D devices, etc.), eliminating the need for you to check the conversion table. When AX-8 Series PLC is set to act as a Modbus TCP Master, you need to first create Modbus TCP Master and then add Modbus TCP Slave to continue further settings. Follow the below section to set up the Modbus TCP Master.

- **Modbus TCP Master tree diagram**

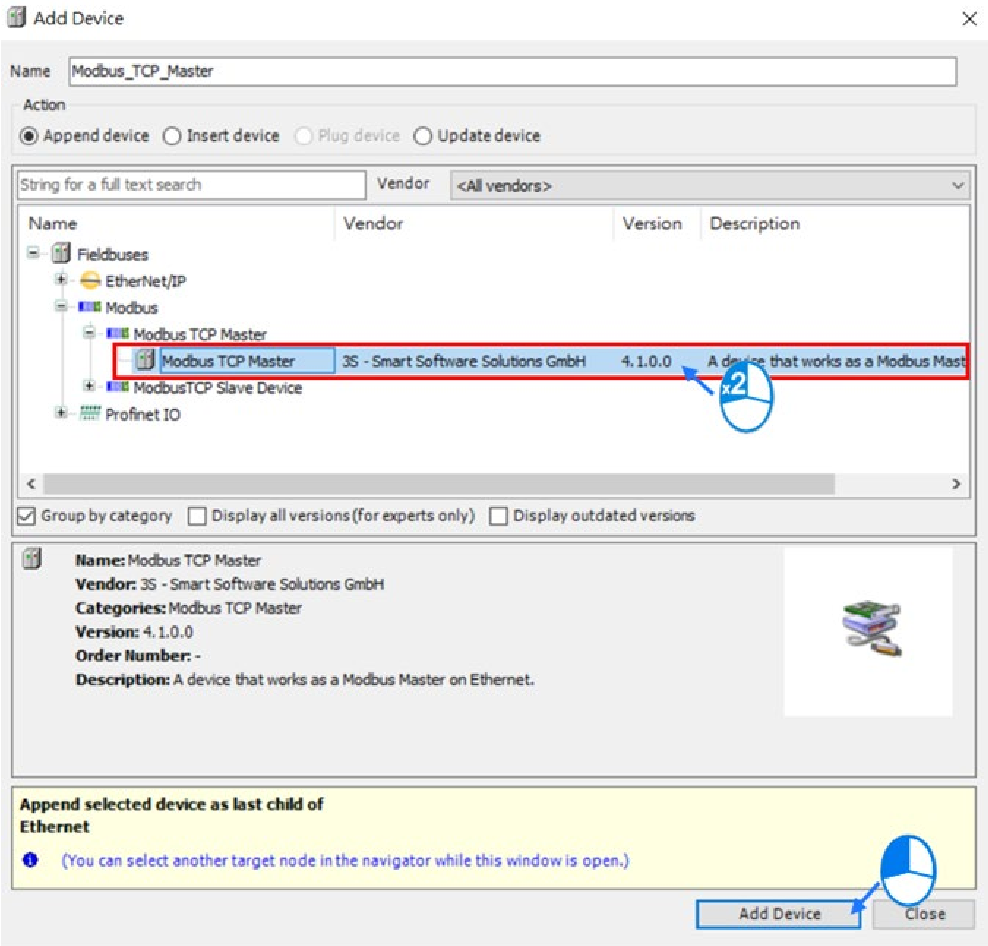


9.3.2.1 Adding a Modbus TCP Master/Slave

1. Right-click the **Ethernet (Ethernet)** node in the tree view to open a context menu. And click **Add Device**…to open the Add Device setting window.

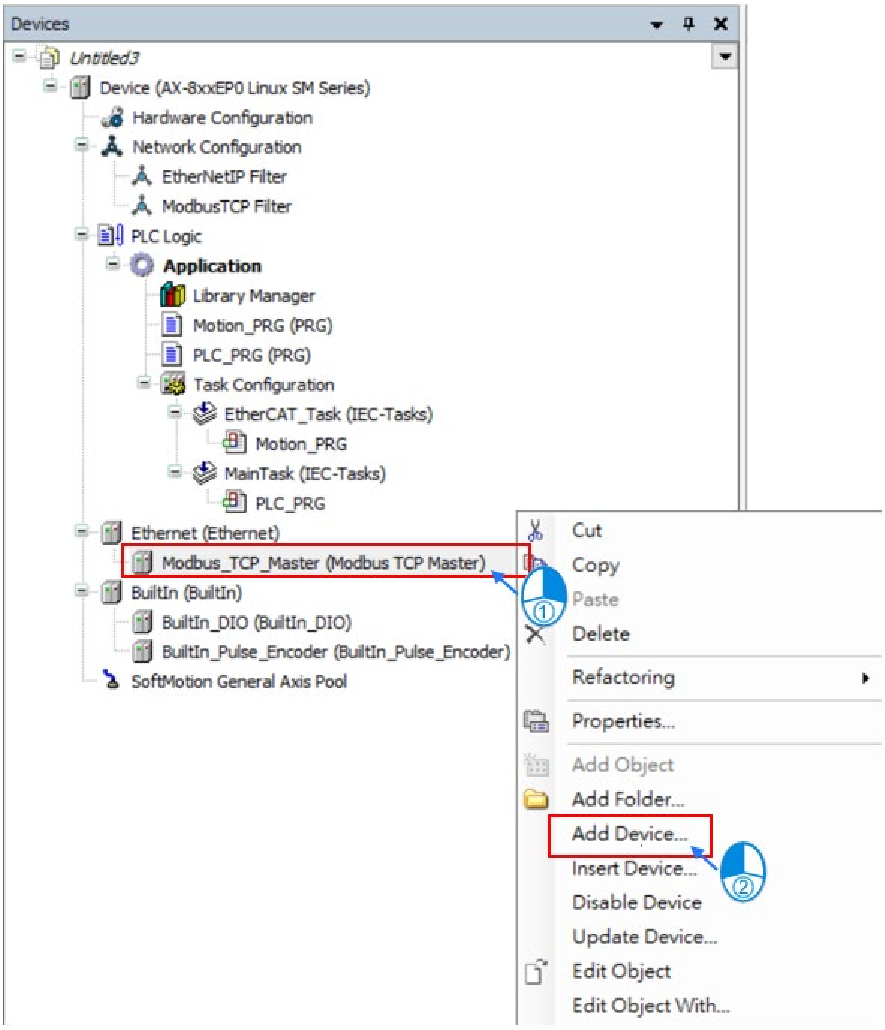


2. Find and double-click **Modbus TCP Master** or click **Add Device** to add this port in. After that you can find **Modbus_TCP_Master** under the Ethernet node in the tree view.

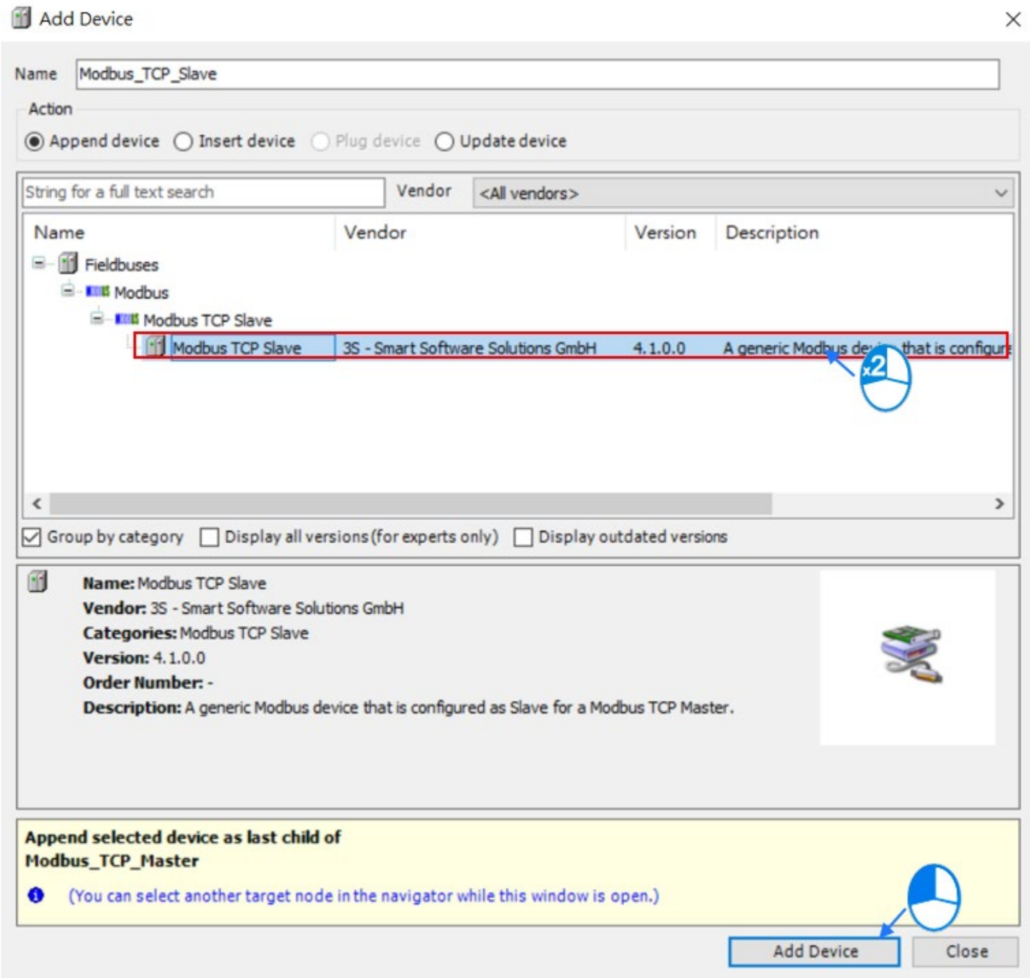


9

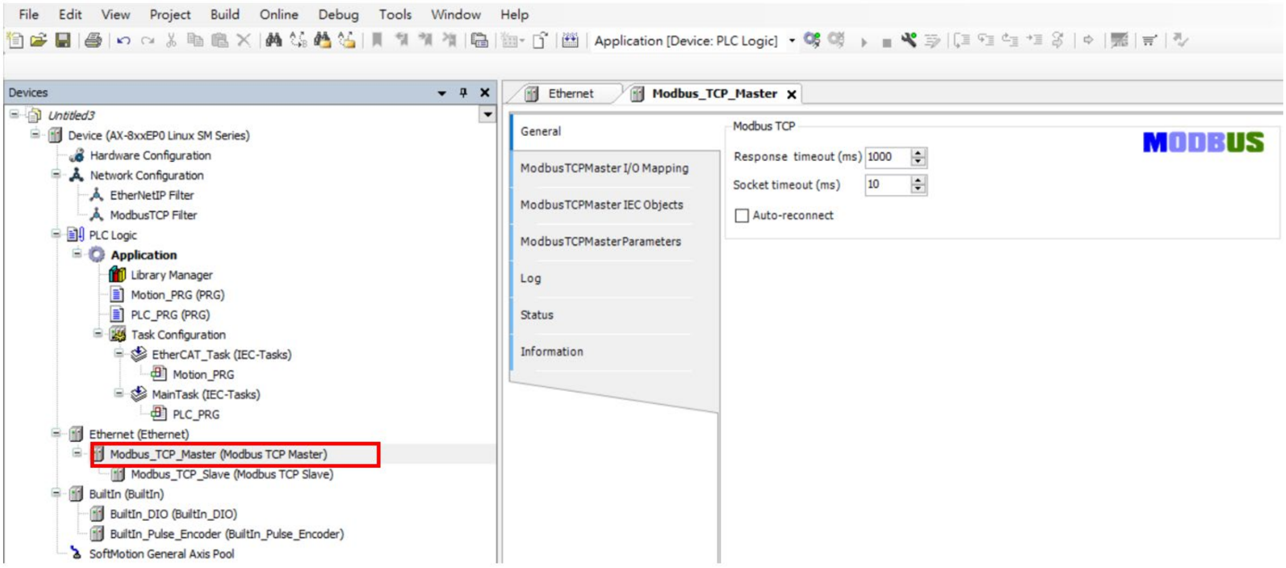
3. Right-click **Modbus_TCP_Master** under the **Ethernet** node in the tree view to open a context menu. And click **Add Device**…to open the Add Device setting window.



4. Find and double-click **Modbus TCP Slave** (Fieldbuses > Modbus > Modbus TCP Slave > Modbus TCP Slave) or click **Add Device** to add it in.



9.3.2.2 Setting up the Modbus TCP Master

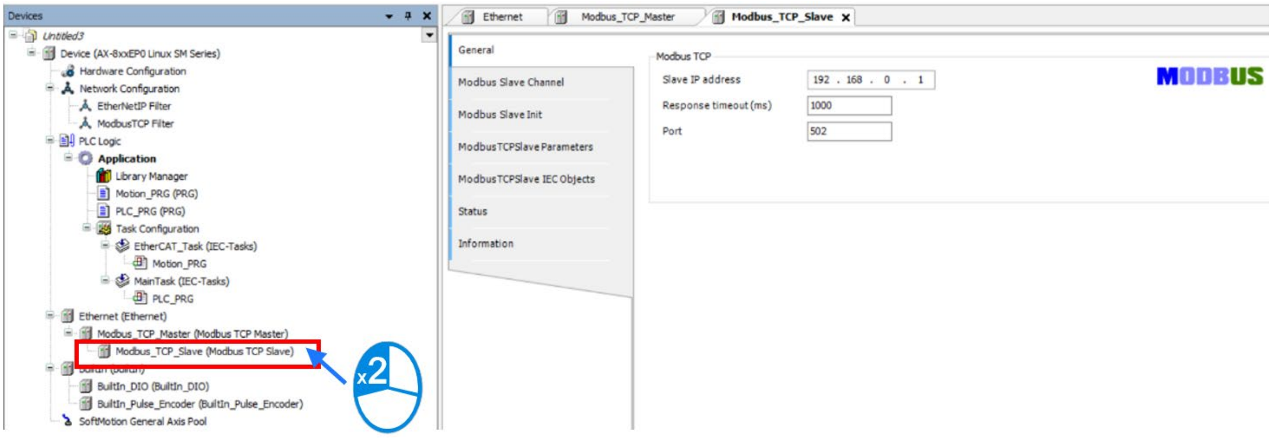


Expression	Description
Modbus TCP Master I/O Mapping	Users can select the bus cycle task of Modbus TCP Master to synchronize with the Modbus communication time. Refer to **4.2.1.6 section PLC Settings** for more information.
Modbus TCP Master IEC Objects	You can check the status of Modbus TCP Master under this tab. ● bStop: TRUE > Stop sending Modbus TCP packets. ● bSlaveError: TRUE > connection/communication with the Slave is abnormal ● uiConnectedSlaves: the number of the connected Slaves e.g.(ST programming language): Modbus TCP Master.bStop:= TRUE;
Status	Here you can find the device status information as well as information about the card used and the internal bus system.
Information	Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.

9

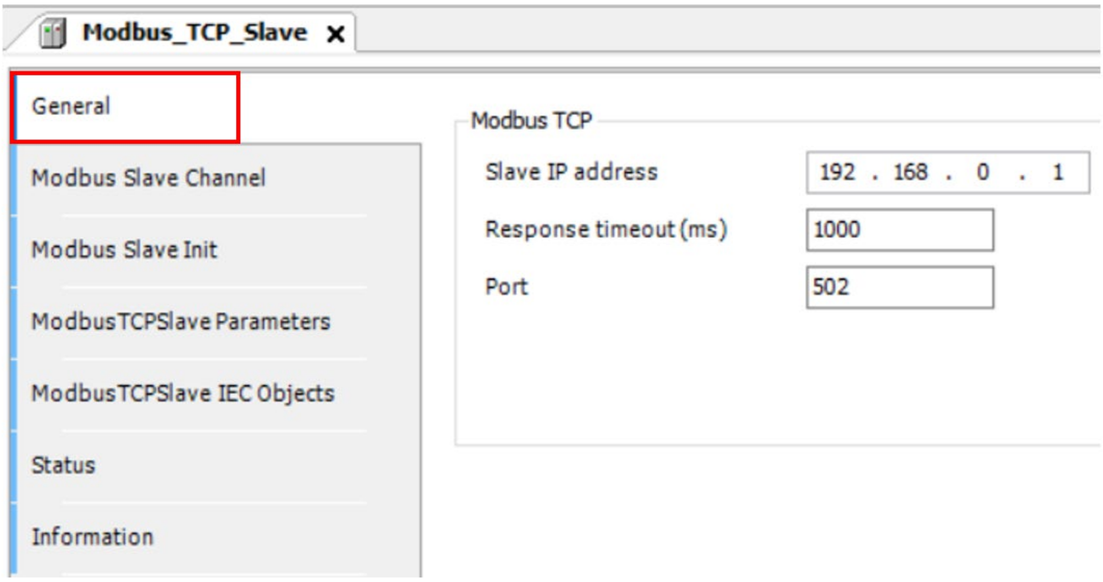
9.3.2.3 Setting up the Modbus TCP Slave

In the tree view, find the **Modbus_TCP_Slave (Modbus TCP Slave)** and double-click it to open the setting window to set up.



- **General**

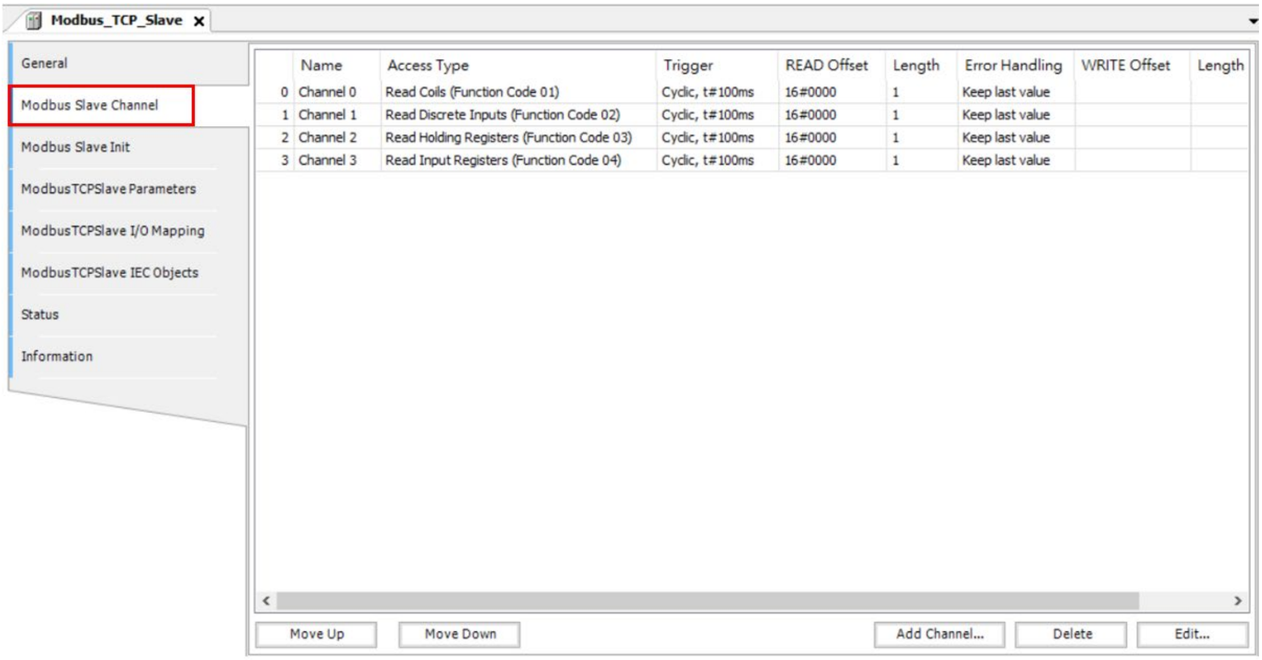
Here you can configure the basic settings for Modbus TCP Slave, such as Slave Address, Response Timeout and Device Type.



Item	Description
Slave IP Address	Address of the slave IP
Response Timeout	Time interval for the master to wait for the response from the slave. This is especially configured for this slave node and overwrites the general response timeout setting of the respective master.
Port	Slave communication port

- **Modbus Slave Channel**

Here you can define slave channels. Each channel represents a single Modbus request. You can create up to 100 channels for each slave. AX-8 Series PLC will send out Modbus request packets in chronological order. All channels share the same Modbus TCP connection.

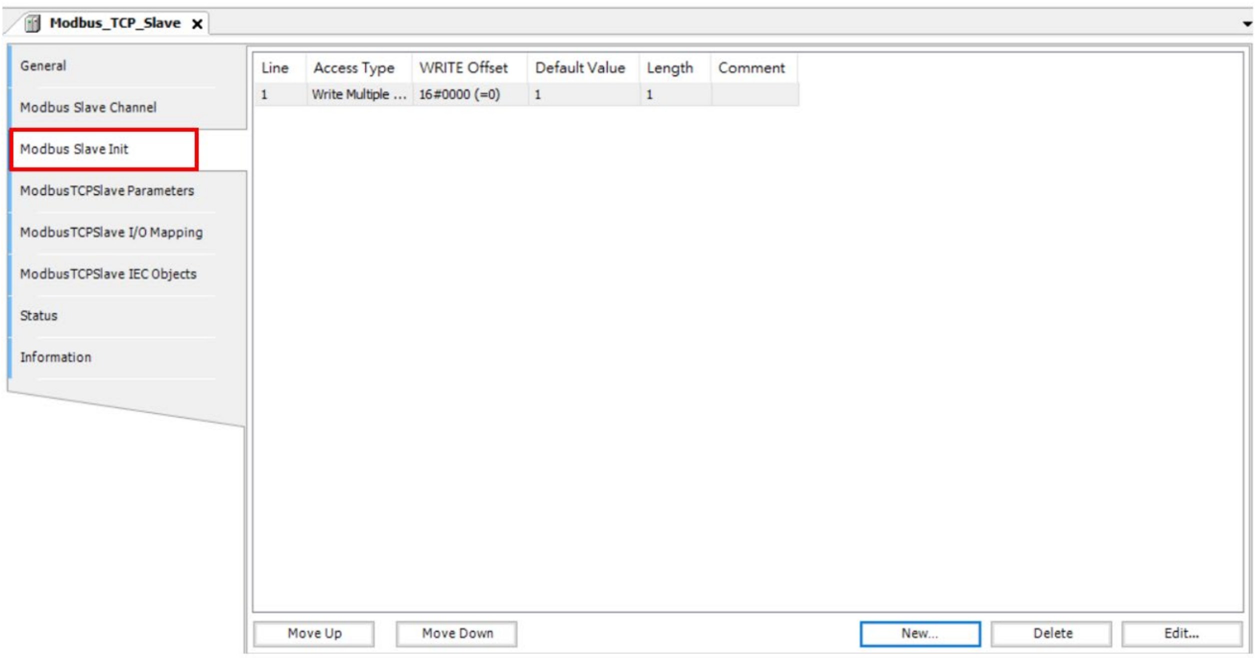


Device Type	Standard Modbus Device	Series Device
Enable	Activates this channel	
Name	Defines this channel name	
Access Type	Modbus function code ● Read Coils (0x01) ● Read discrete inputs (0x02) ● Read holding Registers (0x03) ● Read input Registers (0x04) ● Read single Coil (0x05) ● Write single Register (0x06) ● Write multiple Coils (0x0F) ● Write multiple Registers (0x10) ● Read/Write multiple Registers (0x17)	Read/Write Registers ● Read Coils ● Read Registers ● Write Coils ● Write Registers Note: PLC uses the corresponding Modbus function code according to the read/write register of the device type.
Trigger	● **Cyclic:** The request occurs periodically. ● **Rising edge:** The request occurs as a reaction to a rising edge of the Boolean trigger variables. The trigger variable is defined in the tab I/O Mapping. ● **Application:** The Modbus request is	● **Cyclic:** The request occurs periodically. ● **Rising edge:** The request occurs as a reaction to a rising edge of the Boolean trigger variables. The trigger variable is defined in the tab I/O Mapping. ● **Application:** The Modbus request is triggered

	triggered by **DFB_ModbusTCPChannel**	by **DFB_ModbusTCPChannel**
Comment	Description of the channel	
Device Address	Modbus protocol address	Delta register address (will be converted into Modbus protocl in the background)
Length	Number of the register to be read/written to.	Number of the register to be read/written to. (up to 256 coils and 100 registers)
Error Handling	What to do with the data in case of a communication error: ● Set To ZERO ● Keep last value	

- **Modbus Slave Init**

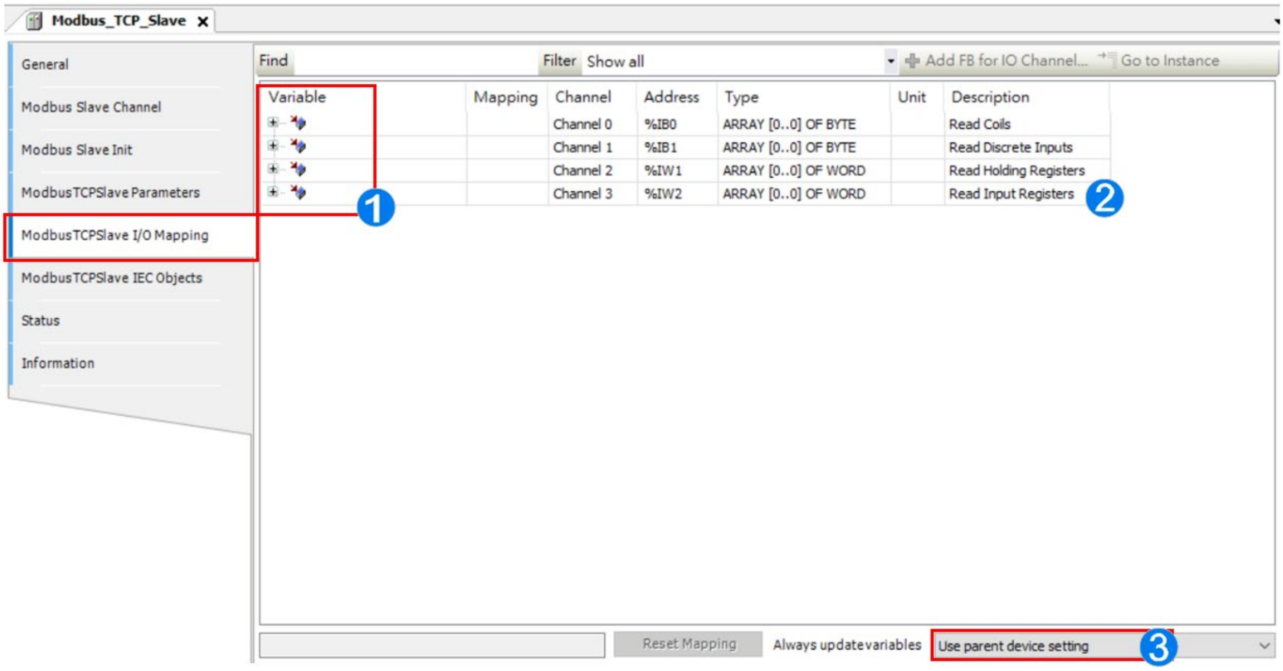
After the Modbus connection between AX-8 Series PLC and the slaves is established, you can use **Add Channel** button to edit the Initialization Value of the Coil/Registers.



9

- **Modbus TCP Slave I/O Mapping**

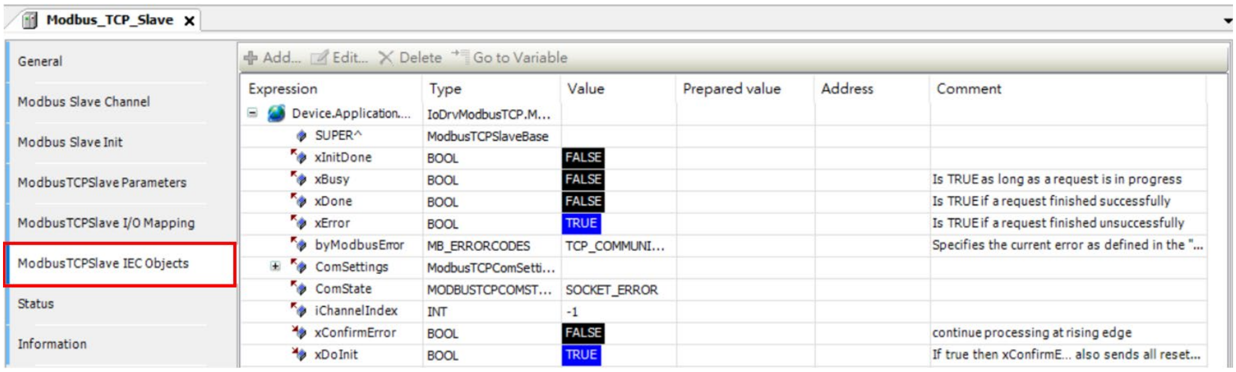
After you have added channels under the tab of Modbus Slave Channel, you can find the variables and the set access types under this tab.



① The descriptions here reflect what you have set for the **Access Type** in Modbus Slave Channel tab.
② The controller registers that are read/written by this channel.
③ Timing for the data refreshing; refer to **section 4.2.1.6 PLC Settings** for more information.

- **Modbus TCP Slave IEC Object**

You can check the status of Modbus TCP Slave under this tab.



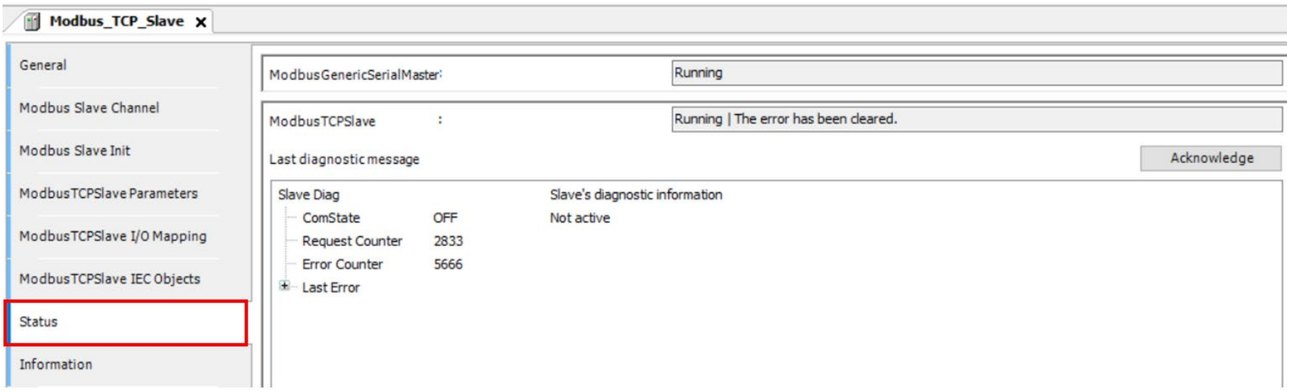
9

Expression	Description
bConfirmError	If the option “Auto-Reconnect” is NOT enabled, during the data transmission, any channel that showed error stops. After the bConfirmError shows “TRUE”, the channel that showed error previously continues to execute.
bDoInit	Initialized the Slave.

Expression	Description
bInitDone	The initialization of the Slave is completed.
bBusy	This channel is in data transmission.
bDone	The data transmission via this channel is completed.
bError	Error occurs when this channel is in data transmission.
ModbusError	Records of the Modbus error.
iChannelIndex	The number of the channel that is in execution.

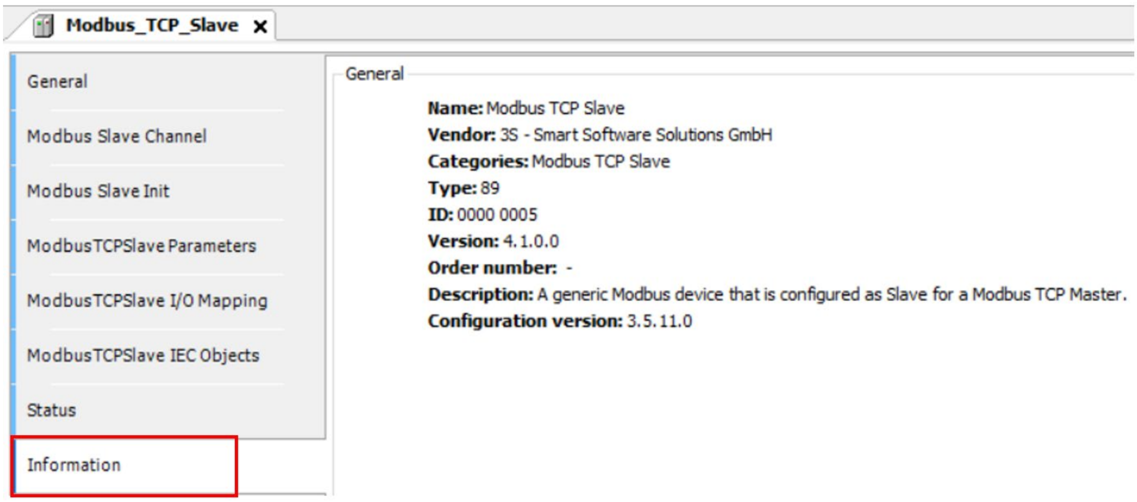
- **Status**

Here you can find the Modbus TCP Slave status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device, also information about the card used and the internal bus system.



- **Information**

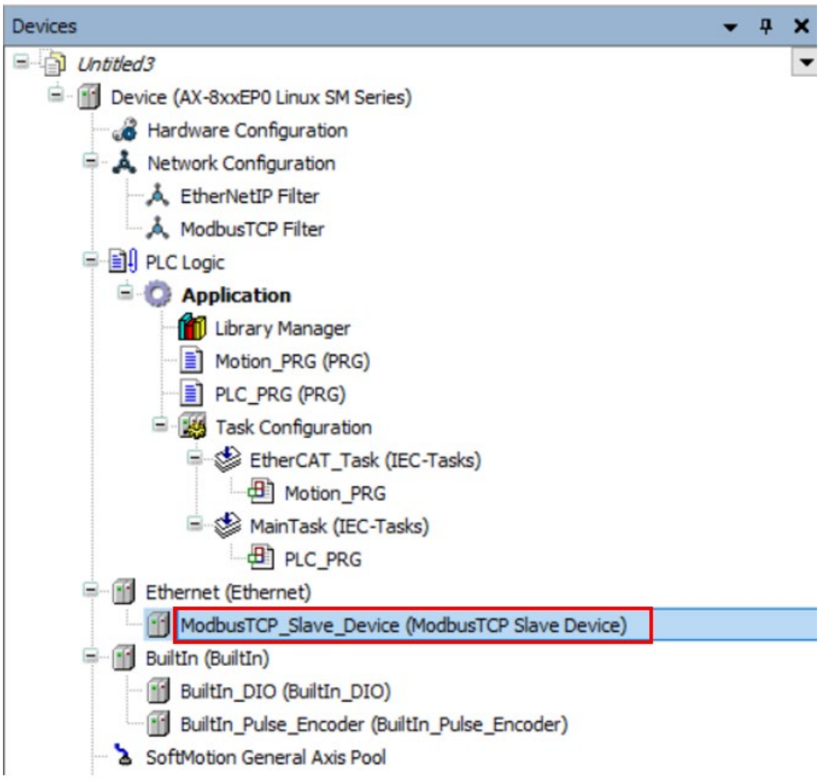
Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.



9.3.3 Modbus TCP Slave

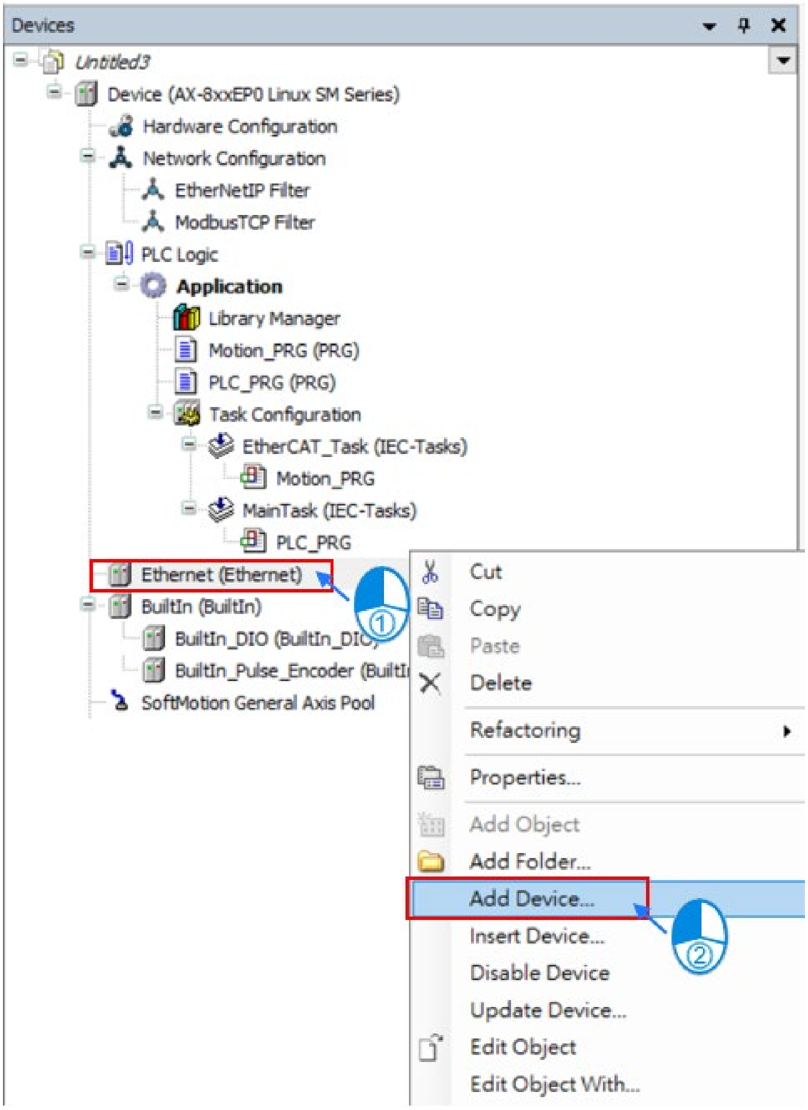
If AX-8 Series PLC is set to act as a Modbus TCP Slave, you need to add Modbus TCP Slave Device in and set up the allowable areas for Coils/Registers. If Modbus TCP Master uses Delta device communication protocol, there is no access restrictions. Follow the below section to set up the Modbus TCP Slave.

- **Modbus TCP Slave tree diagram**

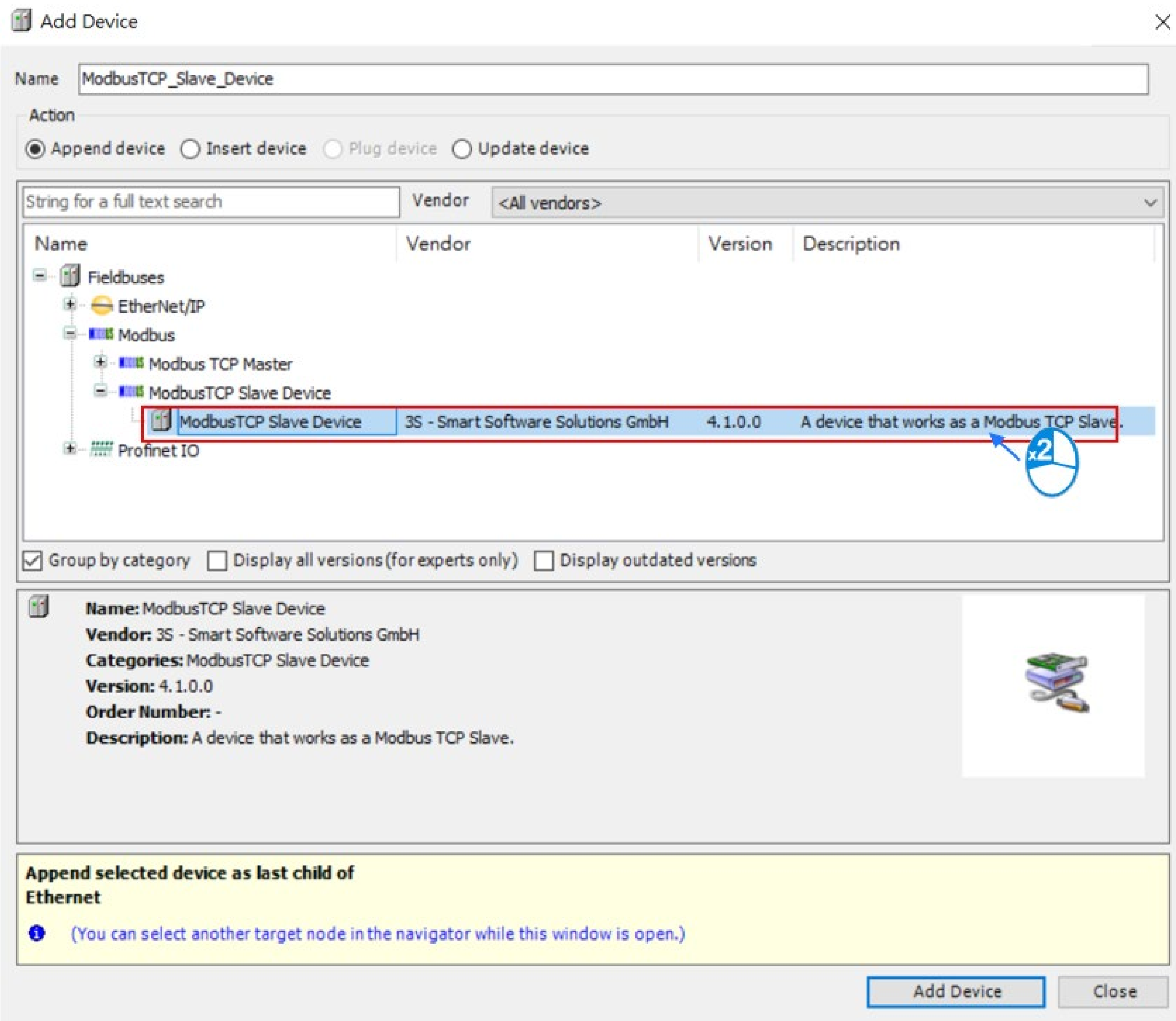


9.3.3.1 Adding a Modbus TCP Slave Device

1. Right-click the **Ethernet (Ethernet)** node in the tree view to open a context menu. And click **Add Device**…to open the Add Device setting window.



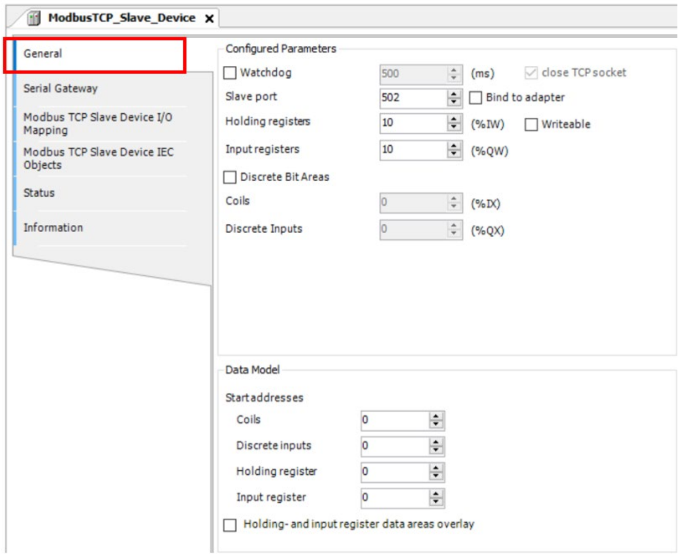
2. Find and double-click **Modbus TCP Slave Device** or click **Add Device** to add this port in.



9.3.3.2 Setting up the Modbus TCP Slave Device

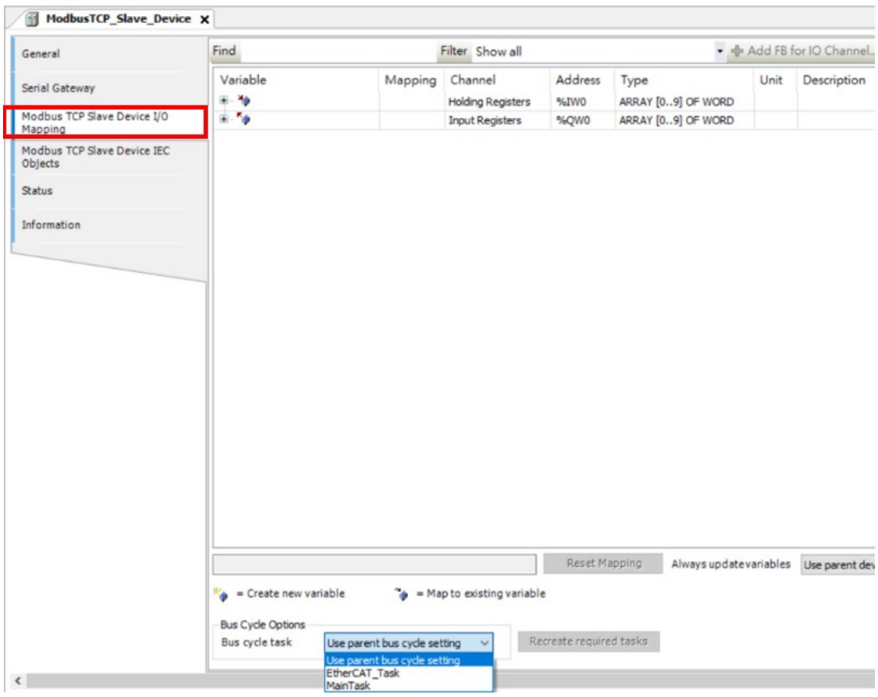
- **General**

Here you can configure the basic settings for Modbus TCP Slave Device. Set up the allowable areas for Coils/Registers. If Modbus TCP Slave uses Delta device communication protocol, there is no access restrictions.



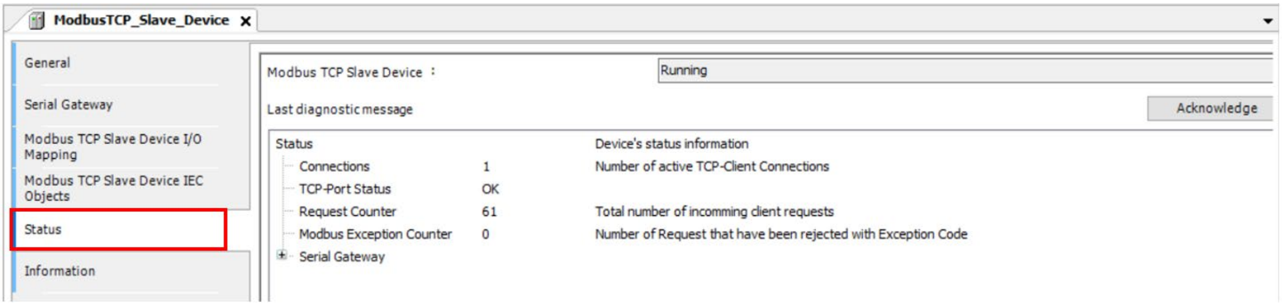
- **Modbus TCP Slave Device I/O Mapping**

Here you can select the bus cycle task of Modbus TCP Slave Device to synchronize with the Modbus communication time. Refer to **section 4.2.1.6 PLC Settings** for more information.



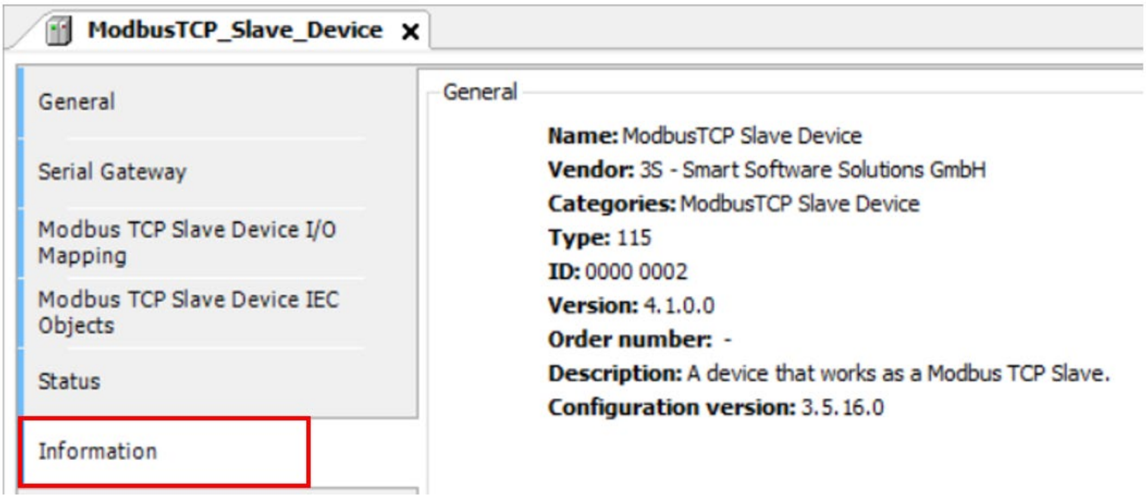
- **Status**

Here you can find the Modbus TCP Slave Device status information, for example **Running** or **Stopped**, and specific diagnostic messages from the respective device, also information about the card used and the internal bus system.



- **Information**

Here you can find general information that originates from the device description file: name, vendor, categories, version, order number, description, and other relevant information.



9.4 EtherNet/IP

Network-related parameters can be modified by adding new Ethernet device. All functions related to network must be adjusted under this tab (e.g. Modbus TCP and EtherNet/IP). Follow the below section to set up the basic settings for communication via the Ethernet Adapter.

9.4.1 Introduction to EtherNet/IP

9.4.1.1 EtherNet/IP Overview

Ethernet Industrial Protocol (EtherNet/IP) is an open industrial networking standard, managed by ODVA (Open DeviceNet Vendors Association).

EtherNet/IP works on a TCP/UDP/IP based Ethernet network and uses most widely deployed collections of Ethernet standards to provide a broad range of applications in different industries that require high-speed and stability including Factory Automation (FA), Building Automation (BA), Process Automation (PA) and many more.

Delta covers a full range of controller and drive products supported by EtherNet/IP, including Programmable Logic Controllers (PLC), inverters, Human Machine Interfaces (HMI) and so on. Refer to **section 9.4.5** for a full product list supported by EtherNet/IP. In addition, users can also use the EDS file to connect to the EtherNet/IP devices of other brands.

9.4.1.2 Definition

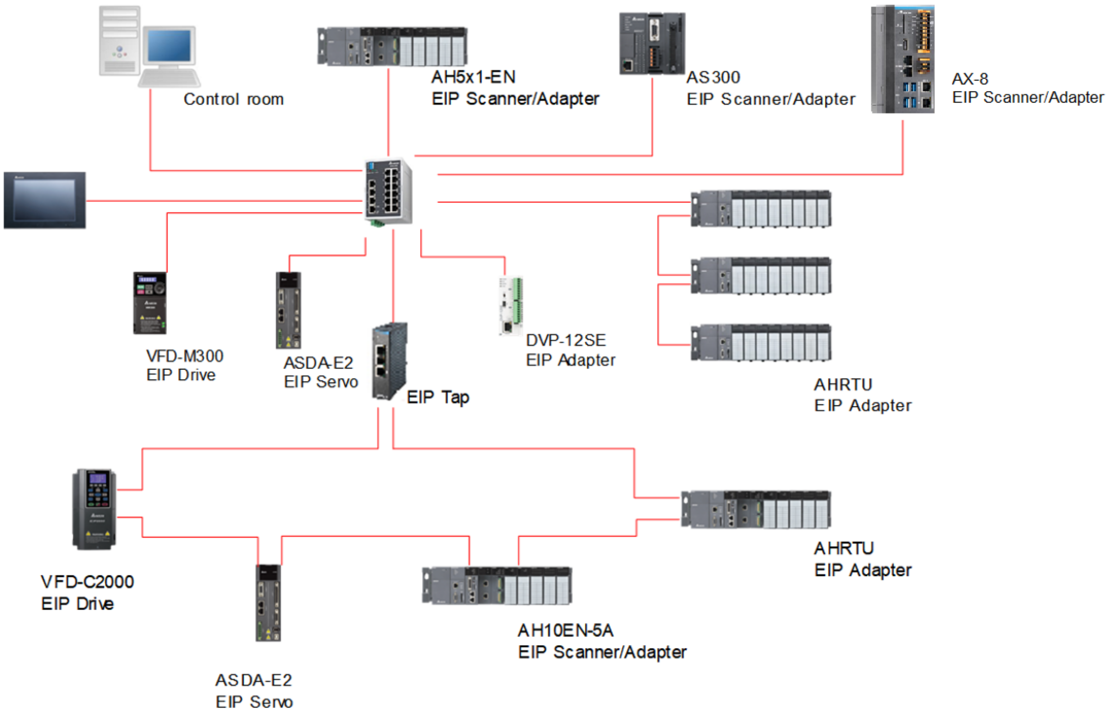
Term	Definition
ODVA	Open DeviceNet Vendor Association for EtherNet/IP
EIP	EtherNet/IP, an industrial Ethernet network, provides interoperability for system providers. IP stands for Industrial Protocol. The term “EIP” (EtherNet/IP) will be used throughout this manual.
I/O Connection	Via the I/O connection to connect to EtherNet/IP and to exchange data cyclically
Explicit Message	Connect to EtherNet/IP and to exchange data non-cyclically. Data will be exchanged piece by piece via instructions.
RPI	Requested Packet Interval, via the I/O connection to connect to EtherNet/IP to exchange data at regular time intervals
ACD	Address Conflict Detection to detect IP address duplications.
P/C TAG	Produced / Consumed TAG. A produced TAG sends its data to consumed TAGs (consumers) without using logic.TAGs are the methods used for assigning and referencing memory locations for Rockwell PLCs, the same as the registers for Delta PLCs.
EDS	Electronic Data Sheets; EDS files are simple text files used by EtherNet/IP network configuration tools to help you identify EtherNet/IP products and easily commission them on a network.
Data Mapping	Exchange data between devices.

EIP Scanner	The master station is called Scanner in EtherNet/IP.
EIP Adapter	The slave station is called Adapter in EtherNet/IP.
MODBUS TCP	MODBUS TCP is a MODBUS communication protocol, widely used on Ethernet.

9.4.1.3 Features of Ethernet

9.4.1.3.1 Delta EIP Structure

This typical Delta EIP architecture includes EIP Scanner and Adapter; data mapping can be achieved between devices via an I/O connection and explicit message.



9.4.1.3.2 Features of EIP

- **Flexibility**
  - Flexible topology: EIP devices may include an Ethernet single port as well as Ethernet dual port, and provide applicable networks such as linear topology, ring topology and ring topology for faster expansion and easier management.
  - Network compatible: IT specialists are not required for Internet connection setup, while the Wi-Fi connection is provided.

9

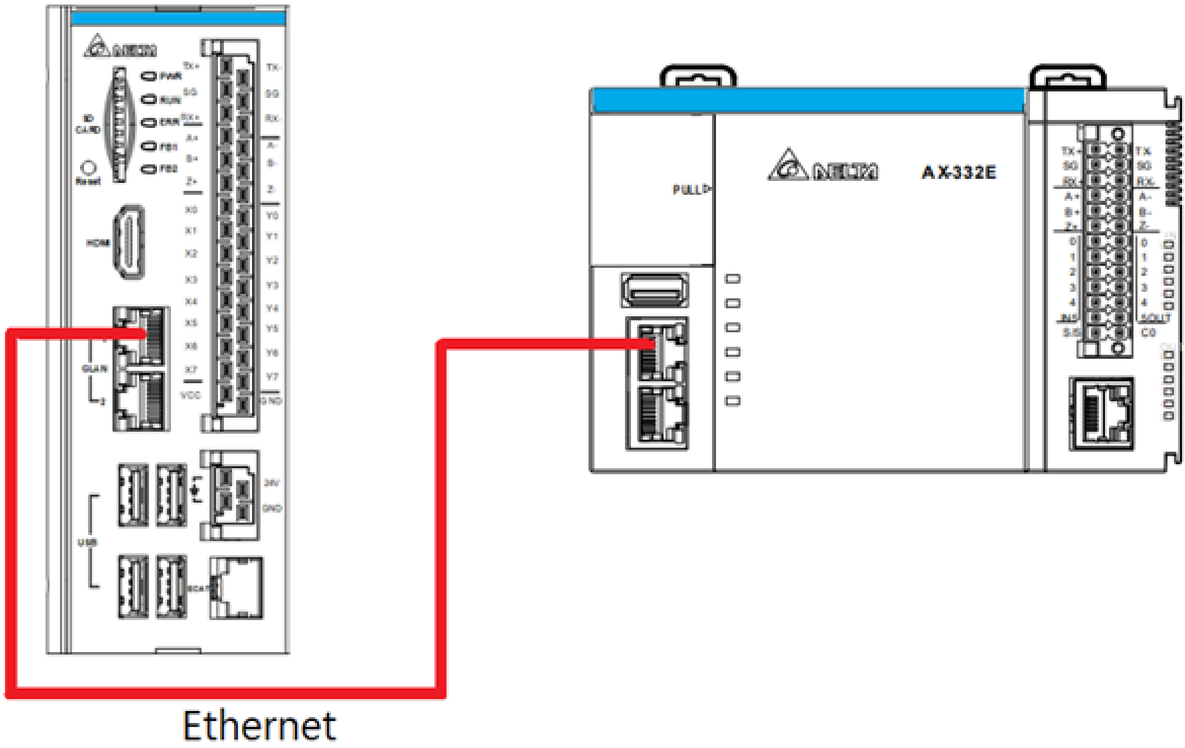
- **Simplicity**
  - Via a connector: Delta provides a full range of product line, including human machine interfaces (HMI), programmable logic controllers (PLC) and inverter drives, for application in an industrial operation. Simply via a RJ-45 connector, a network can be built up, saving costs on cables and other connecting tools.
  - Single network: In replace with the 3-tier industrial architecture, single network architecture provides 100Mbps high-speed cyclical and non-cyclical data mapping function, ensuring a complete network diagnosis and effectively shortening debugging time.

9.4.2 EtherNet/IP Scanner Function

9.4.2.1 Setting up AX-8 Series PLC

9.4.2.1.1 Hardware Configuration

This application example is to connect AX-8 Series PLC to AX-332 via Ethernet.



9.4.2.1.2 Read-Write Setting for Implicit Messages

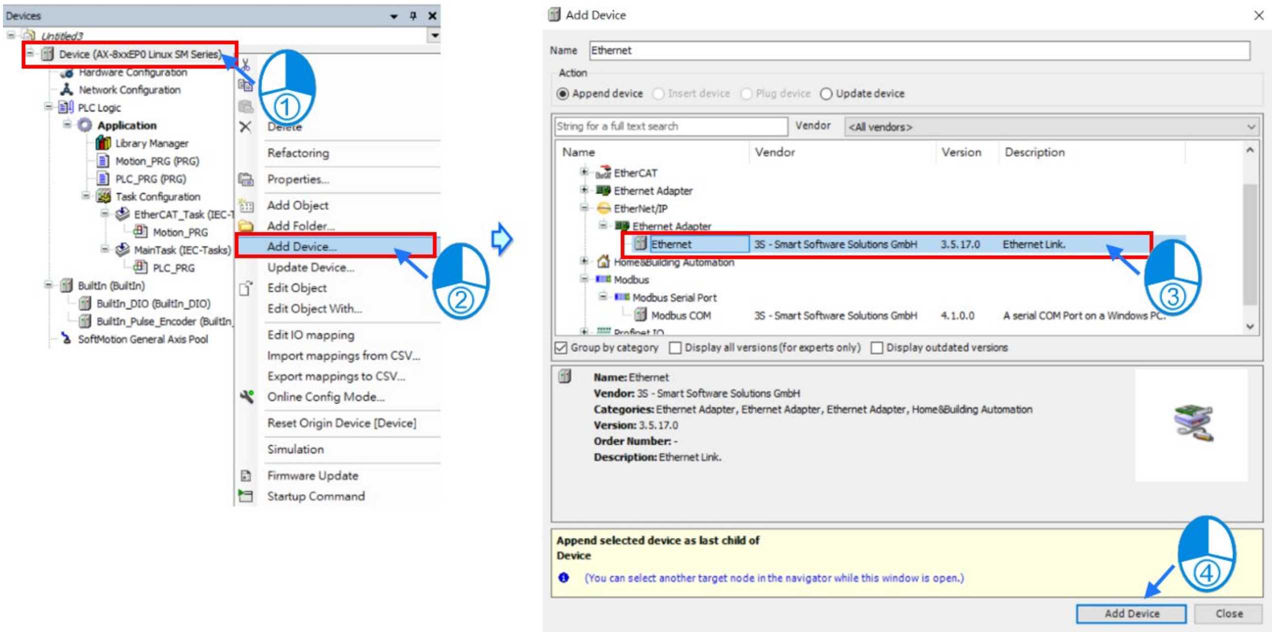
Map the read/write address to the register in option card via the master station (Scanner) to exchange data cyclically and one-time read/write data via the register for implicit messages in EtherNet/IP.

- **Example for creating EIP**

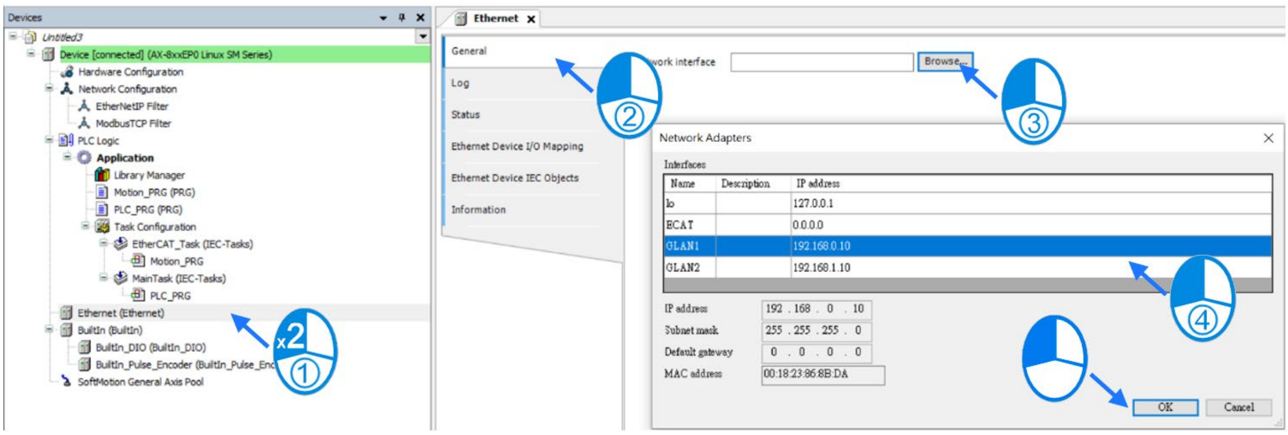
The IP address of the devices applied in this example are shown as follows:

Devices	**AX-8**	192.168.1.10 (default)
	AX-332	192.168.1.11

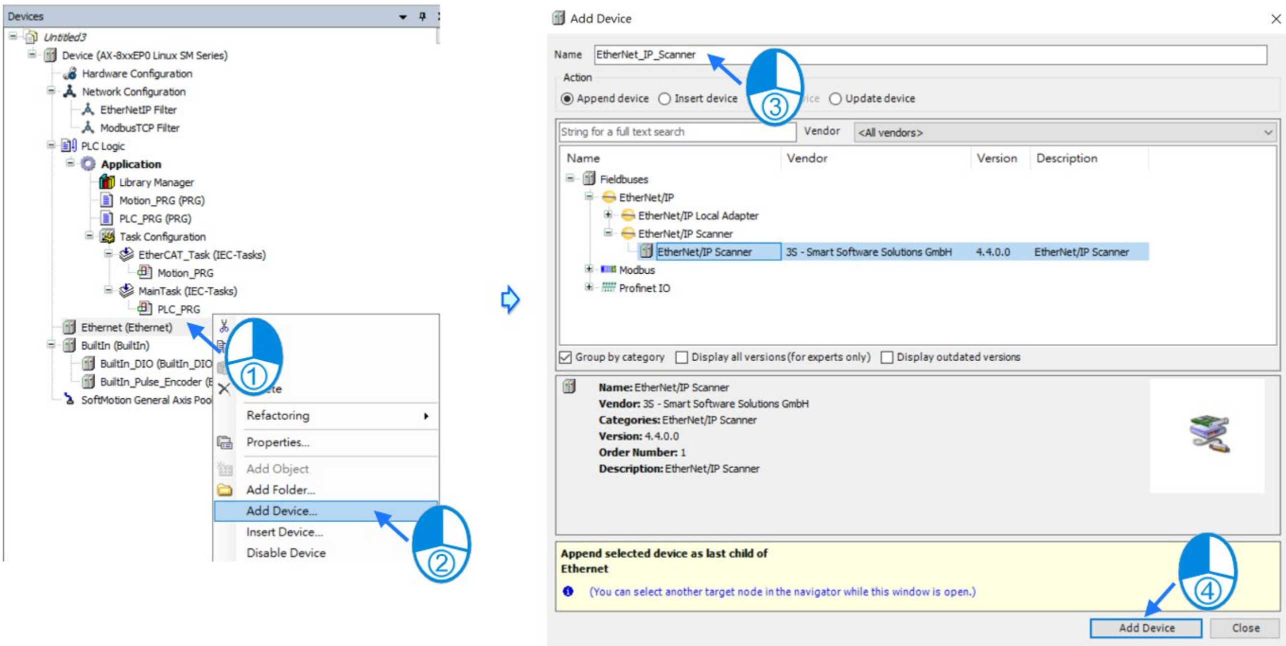
1. Right-click the PLC in the tree view to see a context menu. Click **Add Device**…to open the setting window and create Ethernet Device.



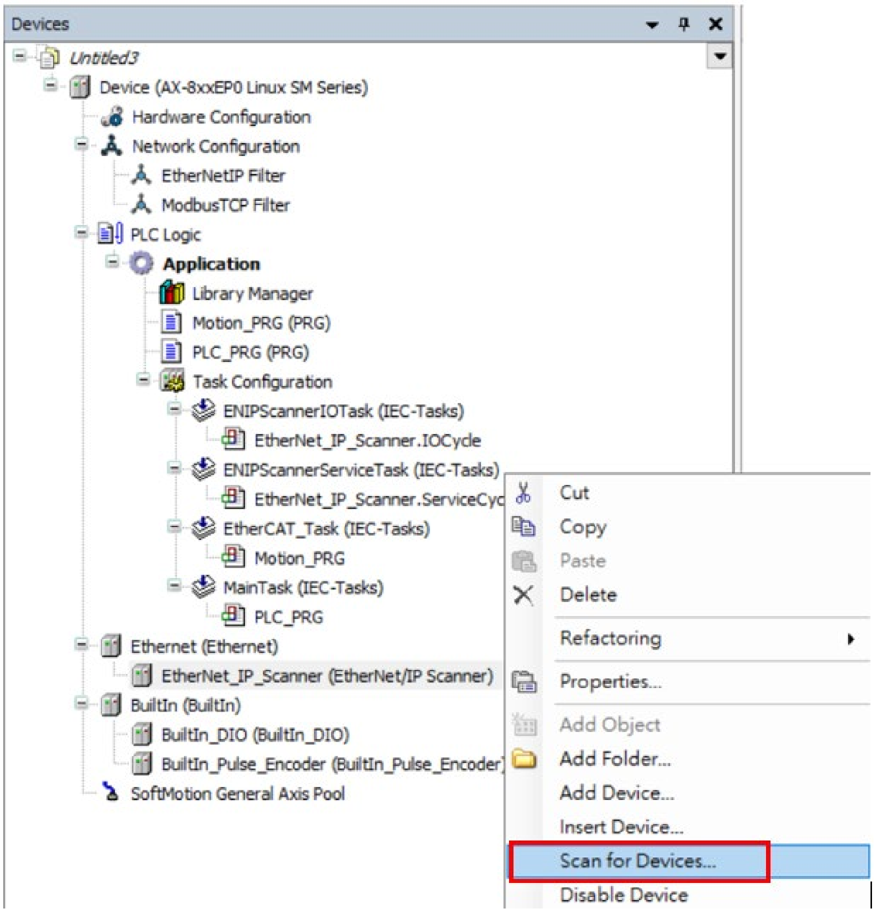
2. Create Interface. Go to **Ethernet (Ethernet)** > **General**.



3. Right-click the **Ethernet (Ethernet)** node in the tree view to see a context menu. Click **Add Device**…to open the setting window and create **EtherNet/IP Scanner**.



4. Adapters can be created via **Scan for Devices**.



9.4.2.1.3 CIP Object Read-Write for Explicit Messages

Please refer to **Appendix A <EtherNet/IP Service and Object>** in VFD EtherNet/IP Application Manual to check the objects supported by the option card and make sure to understand read-write methods for explicit messages before using this function. The master is allowed to configure the setting values of drives directly with the relevant Object Class address. The object class code is 0x300 for drives and the address is formatted as the following shown.

EIP communication data format

Object class		Instance		Attribute
0x300	+	Pr. Group	+	Pr. Number

- **Read-write Example**

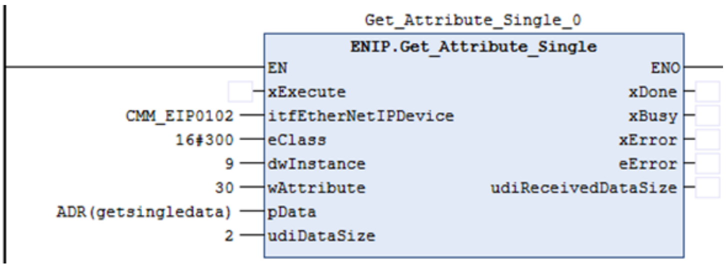
To read and write parameter 09-30 (decoding with Ethernet/IP):

- Declare function blocks and variables.

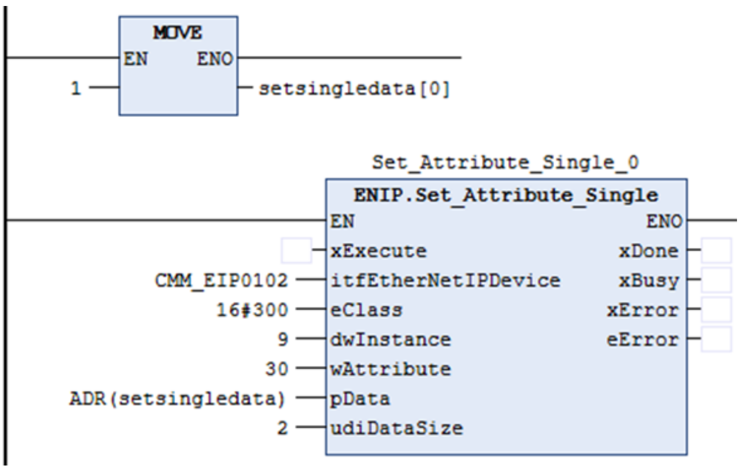
```
PROGRAM PLC_PRG
VAR

    Get_Attribute_Single_0: ENIP.Get_Attribute_Single;
    Set_Attribute_Single_0: ENIP.Set_Attribute_Single;
    getsingledata: ARRAY[0..999] OF BYTE;
    setsingledata: ARRAY[0..999] OF BYTE;
END_VAR
```

- Read parameter 9-30 via the function block as shown below.



- Write 1 to parameter 9-30 via the function block as shown below.



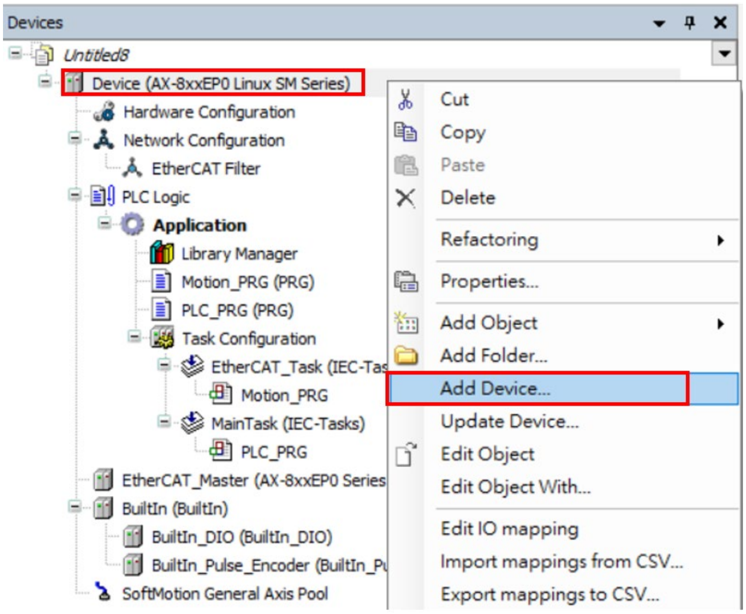
9.4.3 Adapter Function

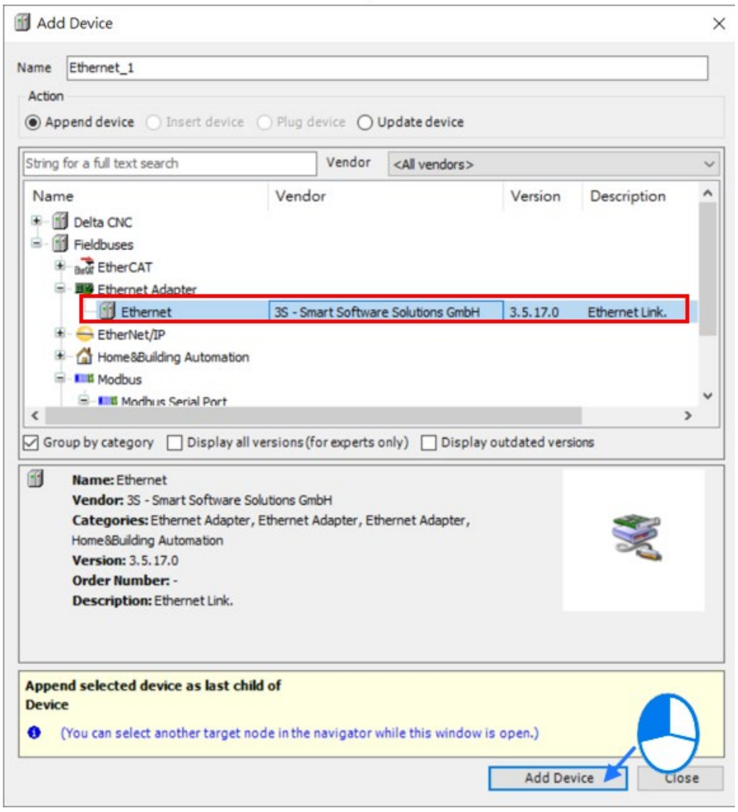
This section will explain how to add an adapter and set up its data structure. Refer to the following steps for configuration.

9.4.3.1 Create an EDS File

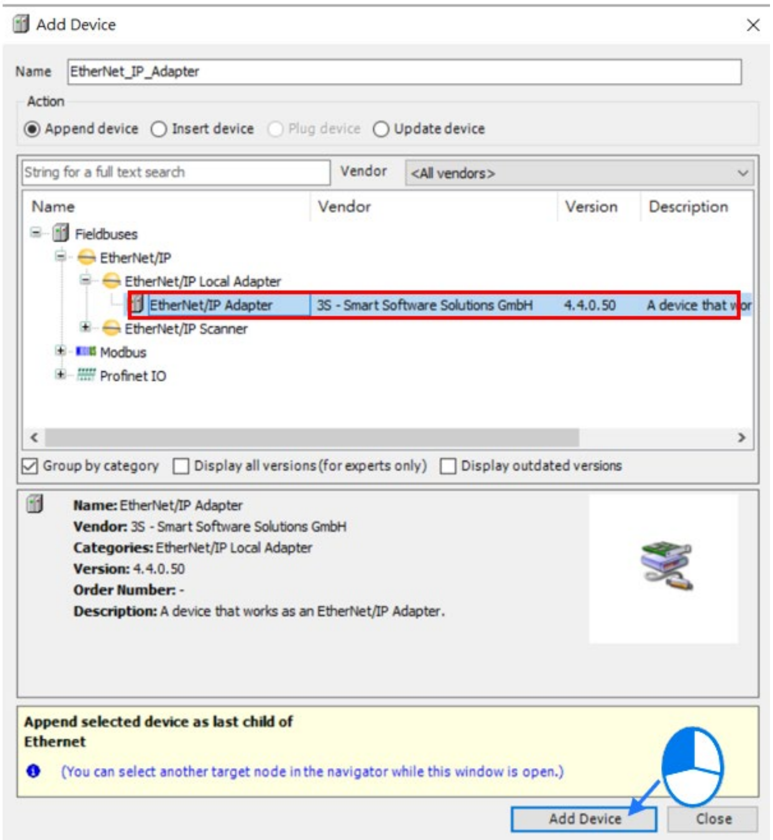
This section will introduce how AX-8 Series PLC creates an EDS File.

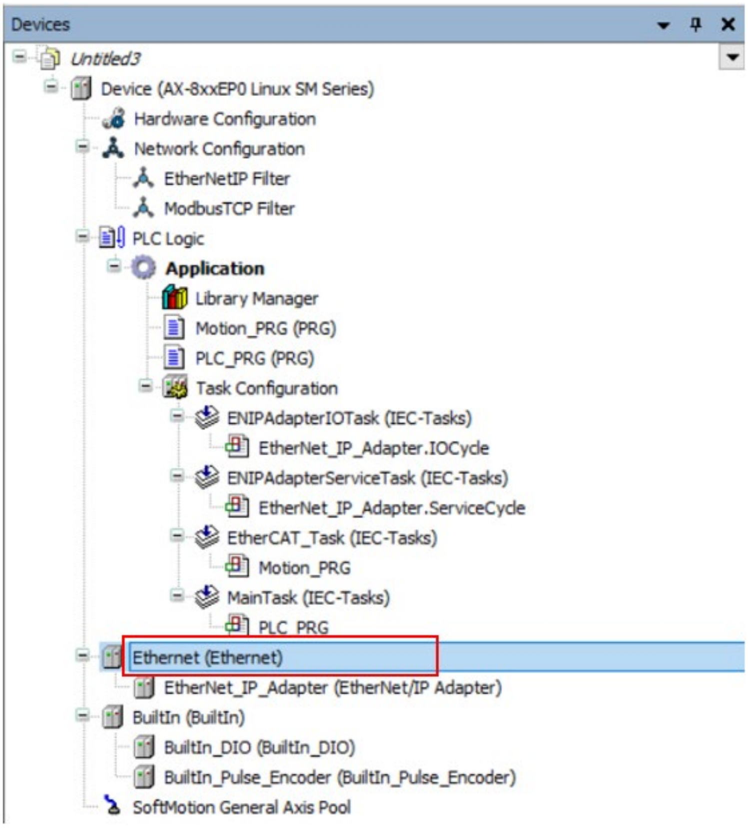
- **Add an Ethernet Device**

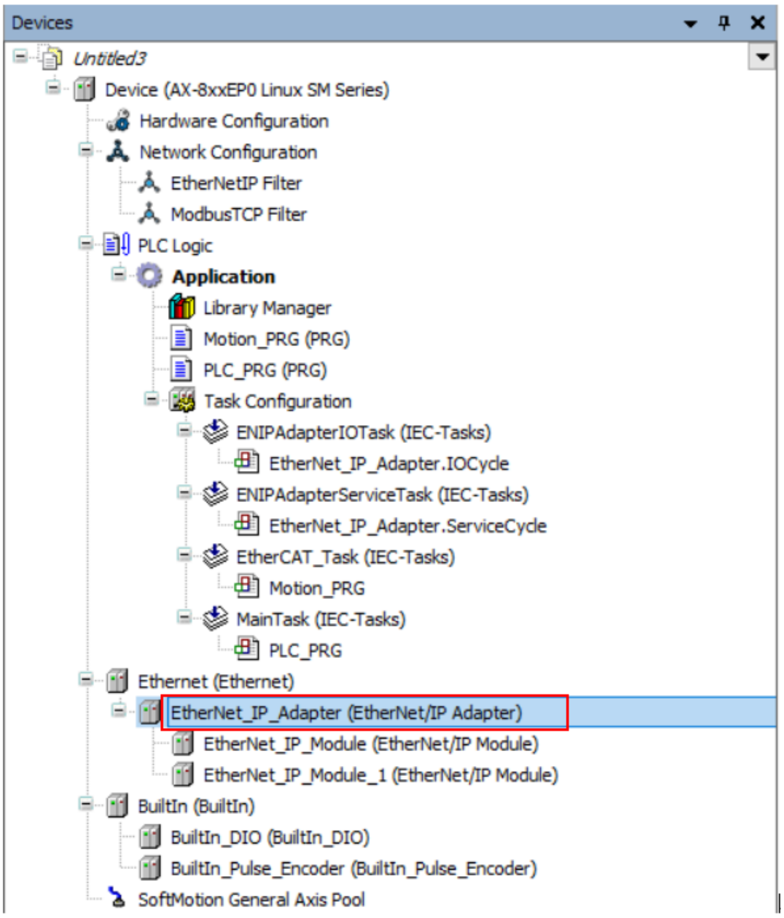




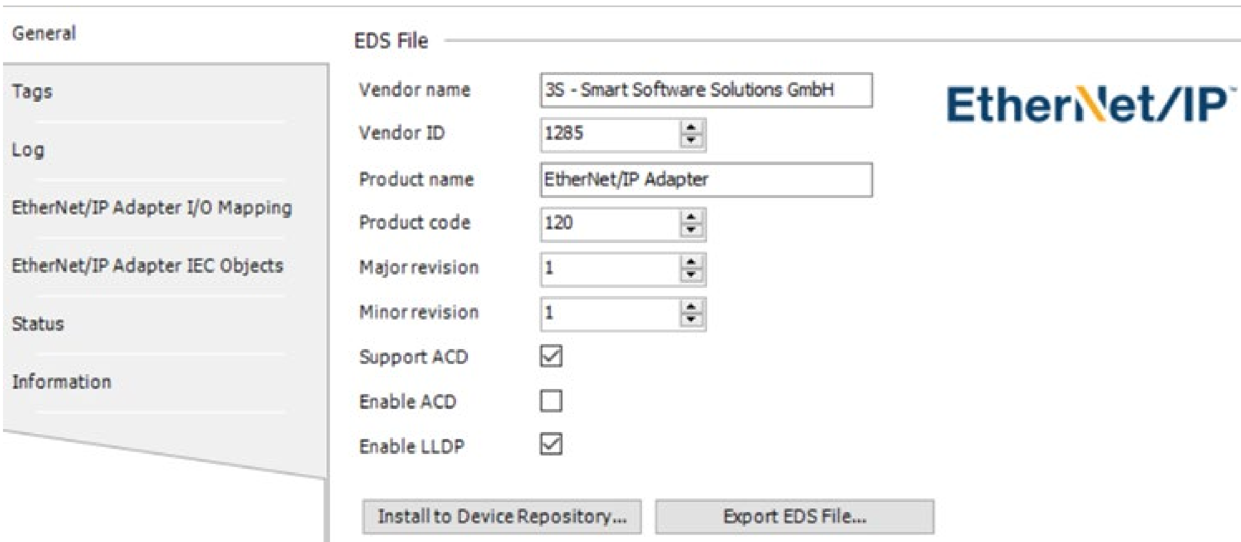
- **Add EtherNet_IP_Adapter Device**







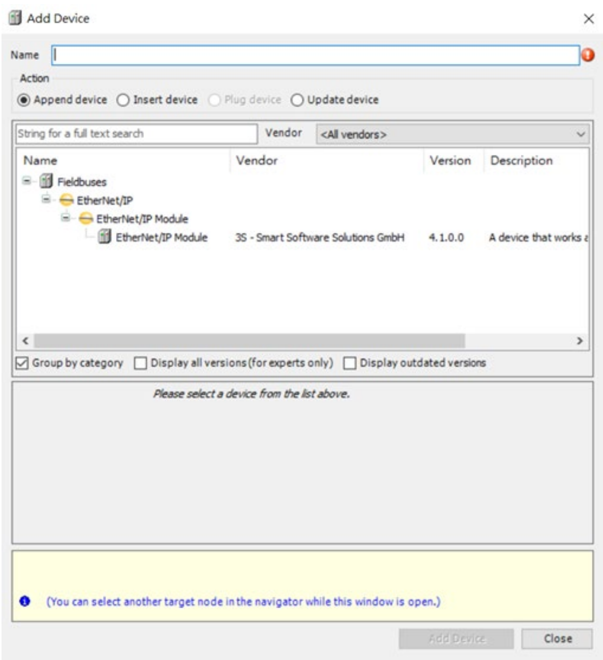
- General- EDS File Settings

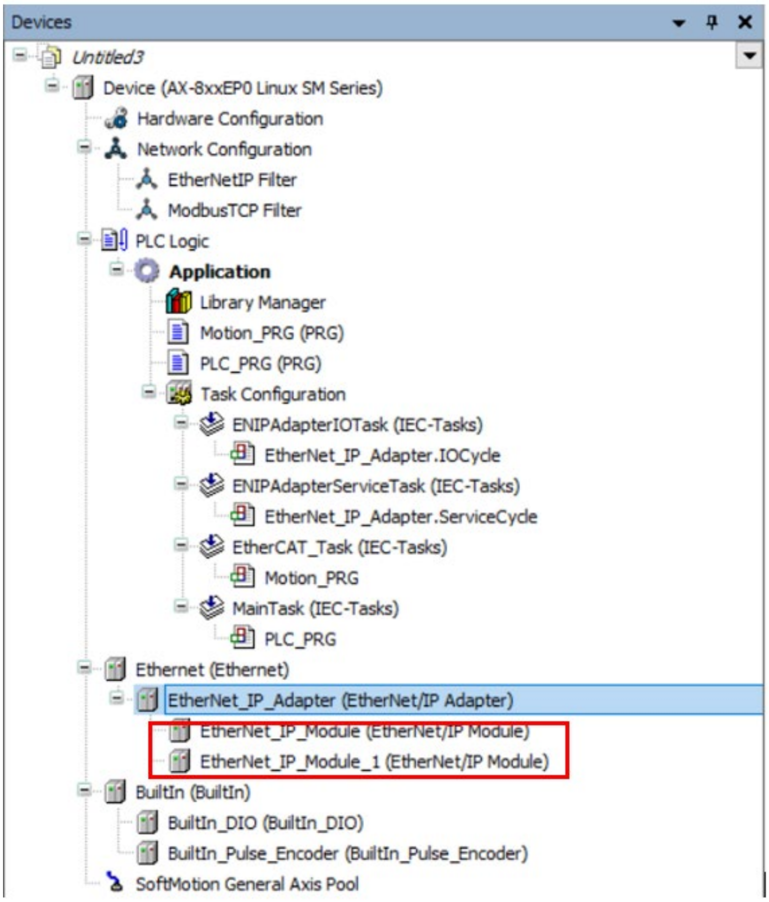


9

Item	Description	Default
Vender name	The name of the supplier	3S - Smart Software Solutions GmbH
Vendor ID	Supplier ID	1285
Product name	The name of the product	EtherNet/IP Adapter
Product code	Product code	120
Major revision	Major revision	1
Minor revision	Minor revision	1
Install to Device Repository	In case that a device with the same device identification has already been installed, you would be asked whether the device should be overwritten. If the device is taken as the remote adapter inserted directly below the EtherNet/IP scanner, you would be asked to update the device automatically.	
Export EDS File	The EDS file is created and stored on the local computer. In this way, the EDS file can be used in an external configuration file.	

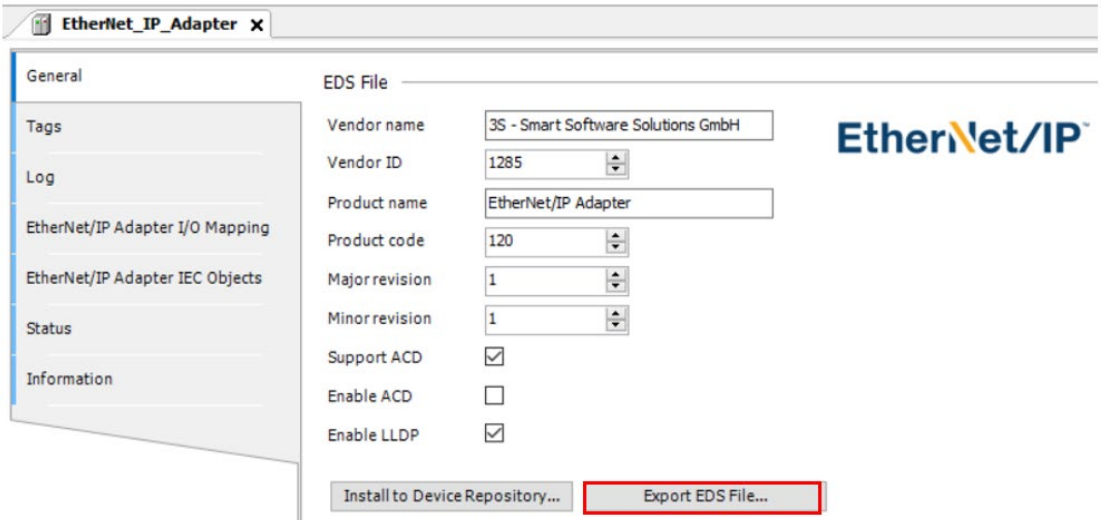
- **Add EtherNet/IP Module to create data file**





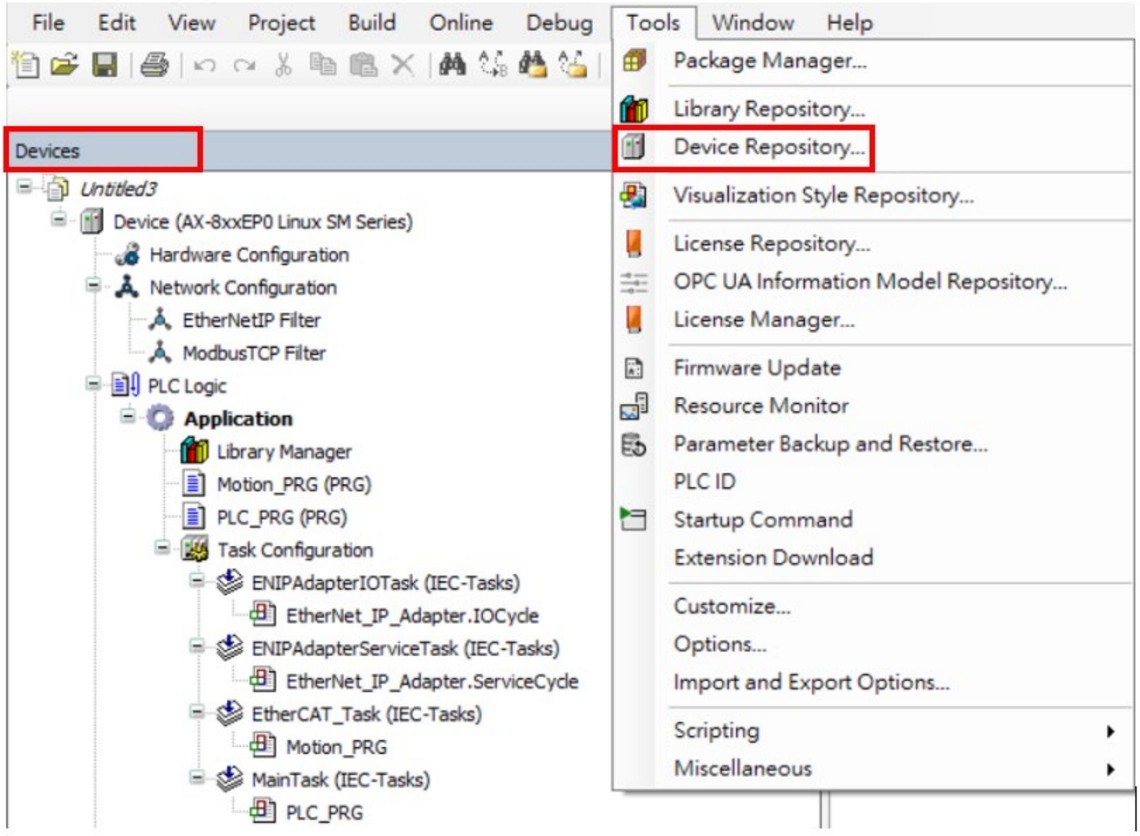
- **Export EDS File**

After the configuration is completed, export the EDS file and store the EDS file – EtherNet_IP Adapter.eds in the PC.

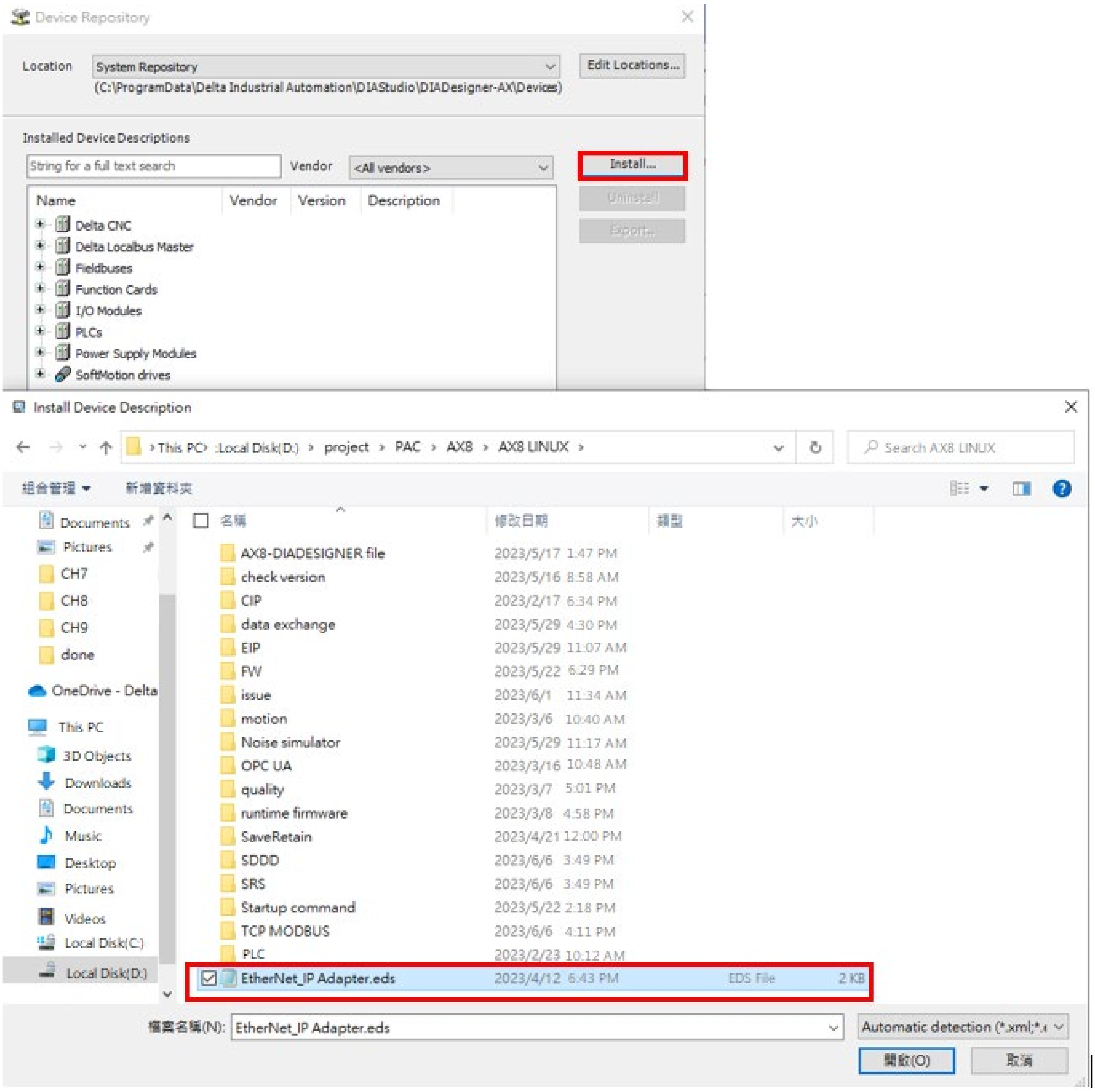


9.4.3.2 Import an EDS File

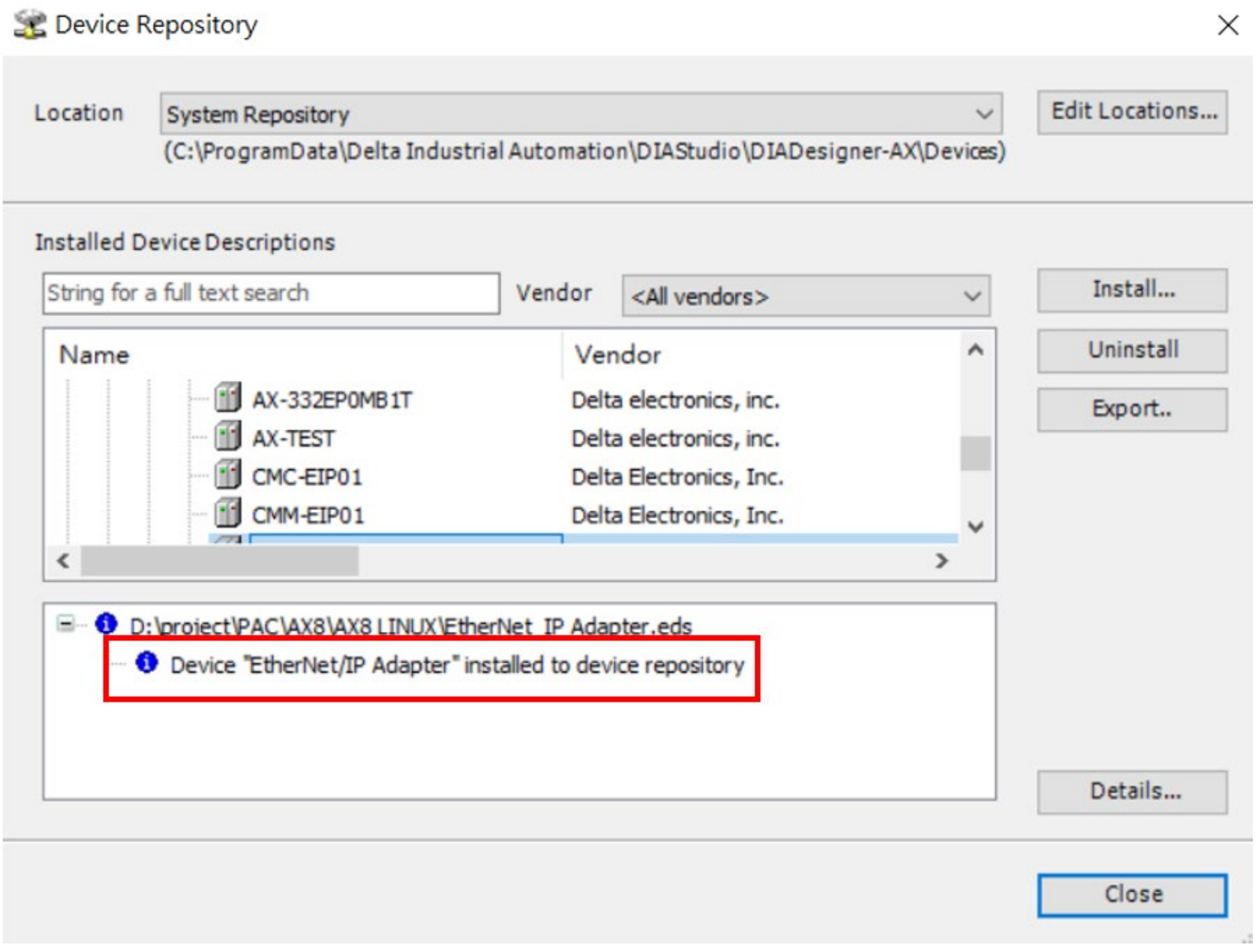
1. Choose **Tools** > **Device Repository**.



2. Choose **Install** and select the target EDS file to download: EtherNet_IP Adapter.eds.



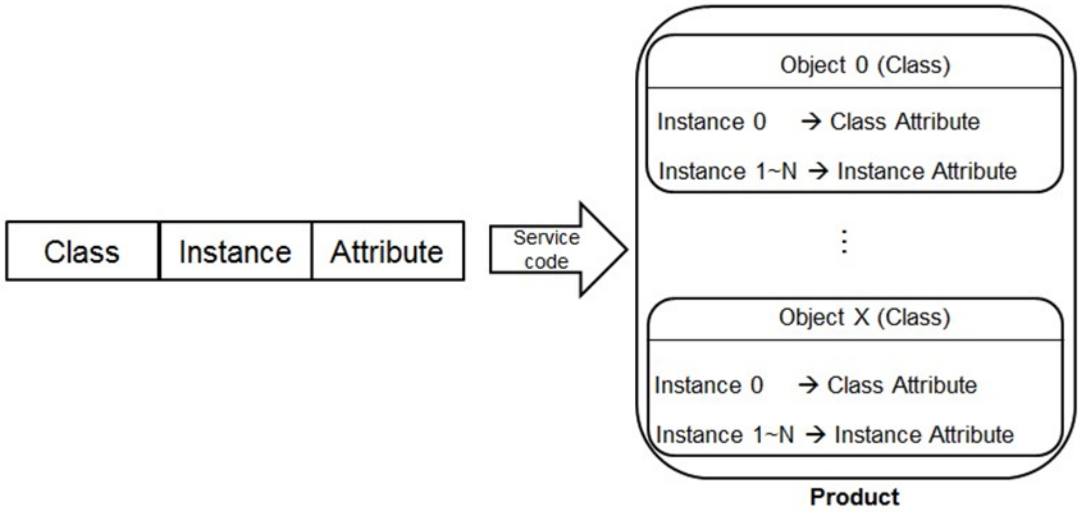
3. Import successfully when you see **Device "EtherNet/IP Adapter" installed to device repository**.



9.4.4 CIP Object

9.4.4.1 Object list

In EtherNet/IP, object is referred to as a set of parameters that is structured accordingly by Class, Instance and Attribute. For example, Instance 0 contains basic information of every object, e.g. version and length. While Instance 1~N creates connection or status of required parameters for each product. Users can obtain product parameters from the supported service code via objects (see diagram below).



Read or write objects by using EtherNetIP Services.library or explicit message tool. The supported EtherNet/IP objects are listed below. Refer to the **section 9.4.4.2** for the data type definition. Refer to the **section 9.4.4.3 ~ 9.4.4.6** for object contents.

Object Name	Function	Class ID
Identity Object	Provides information including manufacturer, device types and versions.	1 (H'01)
Assembly Object	Defines parameter of I/O connection data exchange	4 (H'04)
TCP/IP Interface Object	Displays methods of IP configuration and interface	245 (H'F5)
Ethernet Link Object	Shows the connection status of each Ethernet port on the device.	246 (H'F6)

9.4.4.2 Data Type

This section will provide an overview of the supported data types by objects.

<table>
<tr><th>Data Type</th><th>Description</th></tr>
<tr><td>BOOL</td><td>False (H’00) or True (H’01)</td></tr>
<tr><td>SIGNED
INTEGER</td><td>SINT (1 byte) , INT (2 bytes) , DINT (4 bytes) , LINT (8 bytes)
<table>
<tr><th>Num ber</th><th>1st</th><th>2nd</th><th>3rd</th><th>4th</th><th>5th</th><th>6th</th><th>7th</th><th>8th</th></tr>
<tr><td>SINT</td><td>0LSB</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td></tr>
<tr><td>INT</td><td>0LSB</td><td>1LSB</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td></tr>
<tr><td>DINT</td><td>0LSB</td><td>1LSB</td><td>2LSB</td><td>3LSB</td><td>--</td><td>--</td><td>--</td><td>--</td></tr>
<tr><td>LINT</td><td>0LSB</td><td>1LSB</td><td>2LSB</td><td>3LSB</td><td>4LSB</td><td>5LSB</td><td>6LSB</td><td>7LSB</td></tr>
</table>
E.g. DINT value = H’12345678
<table>
<tr><th>Number</th><th>1st</th><th>2nd</th><th>3rd</th><th>4th</th></tr>
<tr><td>DINT</td><td>78</td><td>56</td><td>34</td><td>12</td></tr>
</table>
</td></tr>
<tr><td>UNSIGNED
INTEGER</td><td>USINT (1 byte) , UINT (2 bytes) , UDINT (4 bytes) , ULINT (8 bytes)
E.g. UDINT value = H’AABBCCDD
<table>
<tr><th>Number</th><th>1st</th><th>2nd</th><th>3rd</th><th>4th</th></tr>
<tr><td>UDINT</td><td>DD</td><td>CC</td><td>BB</td><td>AA</td></tr>
</table>
</td></tr>
<tr><td>STRING</td><td>ASCII, 1 or 2 bytes
STRING : 2 bytes character count + 1 byte character
<table>
<tr><th></th><th colspan="2">Contents (Charcount)</th><th colspan="4">Contents (String contents)</th></tr>
<tr><td>STRING</td><td>04</td><td>00</td><td>4D</td><td>69</td><td>6C</td><td>6C</td></tr>
</table>
STRING2 : 2 bytes character count + 2 byte character
<table>
<tr><th></th><th colspan="2">Contents (Charcount)</th><th colspan="8">Contents (String contents)</th></tr>
<tr><td>STRING2</td><td>04</td><td>00</td><td>4D</td><td>00</td><td>69</td><td>00</td><td>6C</td><td>00</td><td>6C</td><td>00</td></tr>
</table>
SHORT_STRING : 1 byte character count + 1 byte character
<table>
<tr><th></th><th>Contents (Charcount)</th><th colspan="4">Contents (String contents)</th></tr>
<tr><td>STRING</td><td>04</td><td>4D</td><td>69</td><td>6C</td><td>6C</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Data Type</th><th>Description</th></tr>
<tr><td>Fixed LENGTH
BIT STRING</td><td>BYTE（1 byte），WORD（2 bytes），DWORD（4 bytes），LWORD（8 bytes）
<table>
<tr><th></th><th>1st</th><th>2nd</th><th>3rd</th><th>4th</th><th>5th</th><th>6th</th><th>7th</th><th>8th</th></tr>
<tr><td>Byte</td><td>7…0</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td></tr>
<tr><td>WORD</td><td>7…0</td><td>15…8</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td><td>--</td></tr>
<tr><td>DWORD</td><td>7…0</td><td>15…8</td><td>23…16</td><td>31…24</td><td>--</td><td>--</td><td>--</td><td>--</td></tr>
<tr><td>LWORD</td><td>7…0</td><td>15…8</td><td>23…16</td><td>31…24</td><td>39…32</td><td>47…40</td><td>55…48</td><td>63…56</td></tr>
</table>
</td></tr>
<tr><td>STRINGI</td><td>A single string consists multiple language representation.
<table>
<tr><th>Name</th><th>Data Type</th><th>Meaning</th></tr>
<tr><td>Number</td><td>USINT</td><td>The number of internationalized character strings</td></tr>
<tr><td>Strings</td><td>Array of：
Struct of：</td><td>Array of individual internationalized character strings</td></tr>
<tr><td>LanguageChar1</td><td>USINT</td><td>The first ASCII character of the ISO 639-2/T language</td></tr>
<tr><td>LanguageChar2</td><td>USINT</td><td>The second ASCII character of the ISO 639-2/T language</td></tr>
<tr><td>LanguageChar3</td><td>USINT</td><td>The third ASCII character of the ISO 639-2/T language</td></tr>
<tr><td>CharStringStruct</td><td>USINT</td><td>The structure of the character string，limited to the Elementary Data type value 0xD0（STRING），0xD5（STRING2），0xD9（STRINGN）and 0xDA（SHORT_STRING）</td></tr>
<tr><td>CharSet</td><td>UINT</td><td>The character set which the character string is based on which comes from IANA MIB Printer Code（RFC 1759）.</td></tr>
<tr><td>InternationalString</td><td>Defined in CharStringStruct</td><td>An array of 8-bit octet elements which is the actual international character string</td></tr>
</table>
ISO 639-2/T language：
<table>
<tr><th>Language</th><th>First Character</th><th>Second Character</th><th>Third Character</th></tr>
<tr><td>English</td><td>e</td><td>n</td><td>G</td></tr>
<tr><td>French</td><td>f</td><td>r</td><td>e</td></tr>
<tr><td>Spanish</td><td>s</td><td>p</td><td>a</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Data Type</th><th>Description</th></tr>
<tr><td></td><td><table><tr><td>Italian</td><td>i</td><td>t</td><td>a</td></tr></table></td></tr>
<tr><td>STRUCT</td><td>STRUCT of：Any Data Type composes the structure.
E.g. STRUCT of { BOOL、UINT、DINT } = { TRUE、H’1234、H’56789ABC }<table><tr><th></th><th>1st</th><th>2nd</th><th>3rd</th><th>4th</th><th>5th</th><th>6th</th><th>7th</th></tr><tr><td>Byte</td><td>01</td><td>34</td><td>12</td><td>BC</td><td>9A</td><td>78</td><td>56</td></tr></table></td></tr>
<tr><td>ARRAY</td><td>Array of：Any Data Type composes the array.
E.g. ARRAY of UINTs = { 1、2、3 }<table><tr><th>Number</th><th>1st</th><th>2nd</th><th>3rd</th><th>4th</th><th>5th</th><th>6th</th></tr><tr><td>Array</td><td>01</td><td>00</td><td>02</td><td>00</td><td>03</td><td>00</td></tr></table></td></tr>
<tr><td>EPATH</td><td>It’s a path that consists of multiple segments and references the class, instance and attribute of another object.
E.g. Identity Object, Instance attribute 5 = “20 01 24 01 30 05 “</td></tr>
</table>

9.4.4.3 Identity Object (Class ID: 01 Hex)

Identity information is stored in the Identity Object and consists of the Vendor ID, Device Type, Product Code and Major Revision for your device.

- **Service Code**

Service code	Service Name	Attribute		Description
		Class Attribute	Instance Attribute	
H'01	Get_ Attributes_ All	X	V	Read all attributes.
H'05	Reset	X	V	Reset.
H'0E	Get_ Attribute_ Single	X	V	Read a single attribute.

- **Class**
  - Class ID：H'01
- **Instance**
  - H'01：Instance Attribute
  - When Instance =1, Instance Attribute are listed below:

Instance Attribute	Name	Access Rule	Data Type	Values	Description
H'01	Vendor ID	Get	UINT	H'31F	Delta Electronics, Inc.
H'02	Device Type	Get	UINT	H'0C	Data Type: Communication Adapter
H'03	Product Code	Get	UINT	H'4002	Product code
H'04	Revision	Get	STRUCT	--	Revision of this device: Major, Minor.
	Major Revision		USINT	H'01	Major Revision Range: H'01~H'7F
	Minor Revision		USINT	H'01	Minor Revision Range: H'01~H'FF
H'05	Status	Get	WORD	H'64	Status*1
H'06	Serial Number	Get	UDINT	H'2374F75C	The last 8 characters of the MAC address 23: 74: f7: 5C
H'07	Product Name	Get	SHORT_STRING	The maximum number of a product name is 32 words. (Data length +Product Name)	

Instance Attribute	Name	Access Rule	Data Type	Values	Description
				(H'0D) AX-816EP0ME1T	

*1：Status Description (H'05)

Bit (s)	Name	Description
0	Owned	Display if the device has an owner connection. 0: No 1: Yes
1	Reserved	0：Always OFF
2	Configured	Display if the device is configured or not. 0: No 1: Yes
3	Reserved	0：Always OFF
4-7	Extended Device Status	0：Self-Testing 1：Firmware Update 2：At least one faulted I/O connection 3：No I/O connections established 4：Non-Volatile Configuration bad 5：Major Fault 6：At least one I/O connection in run mode 7：At least one I/O connection established，all in idle mode 8-15：Reserved
8	Minor Recoverable Fault	0: No minor recoverable fault detected 1: Minor recoverable fault detected
9	Minor Unrecoverable Fault	0: No minor unrecoverable fault detected 1: Minor unrecoverable fault detected
10	Major Recoverable Fault	0: No major recoverable fault detected 1: Major recoverable fault detected
11	Major Unrecoverable Fault	0: No major unrecoverable fault detected 1: Major unrecoverable fault detected

9.4.4.4 Assembly Object (Class ID: 04 Hex)

Assembly Objects are used to aggregate data for the input data and output data associated with I/O connections.

- **Service Code**

Service Code	Service Name	Attribute		Description
		Class Attribute	Instance Attribute	
H’0E	Get_ Attribute_ Single	X	V	Read a single attribute.

- **Class**
  - Class ID：H’04
- **Instance**
  - H’64：Output assembly
  - H’65：Input assembly
  - H’66：Dummy (needed for compatibility)
  - When Instance = 64~66, Instance Attribute are listed below:

Instance Attribute	Name	Access Rule	Data Type	Values	Description
H’03	Data	Get	ARRAY of BYTE	H’2	IO Connection Data

- **Example of reading and writing objects**
  - To read Output assembly data, write the data as shown below

    Service code：H’0E

    Class ID：H’04

    Instance ID：H’64

    Attribute ID：H’03

  - To read Input assembly data, write the data as shown below

    Service code：H’0E

    Class ID：H’04

    Instance ID：H’65

    Attribute ID：H’03

9.4.4.5 TCP/IP Interface Object (Class ID: F5 Hex)

- **Service Code**

Service Code	Service Name	Attribute		Description
		Class Attribute	Instance Attribute	
H'0E	Get_Attribute_ Single	V	V	Read a single attribute
H'10	Set_Attribute_ Single	X	V	Set values of a single attribute

- **Class**
  - Class ID = H'F5
- **Instance**
  - H'00：Class Attribute
  - H'01：Instance Attribute
  - When Instance = 0, Class Attribute are listed below:

Class Attribute	Name	Access Rule	Data Type	Values	Description
H'01	Revision	Get	UINT	H'4	Object revision

  - When Instance = 1, Instance Attribute are listed below:

Instance Attribute	Name	Access Rule	Data Type	Values	Description
H'01	Status	Get	DWORD	H'2	IP Status[*1]
H'02	Configuration Capability	Get	DWORD	H'20	Configuration capability[*2]
H'03	Configuration Control	Get/Set	DWORD	H'0	Configuration control[*3]
H'04	Physical Link Object：	Get	STRUCT of	--	Path to physical link object
	Path Size		UINT	H'0	Size of Path
	Path		EPATH	--	Logical segments identifying the physical link object
H'05	Interface	Get/Set	STRUCT	--	TCP/IP network

Instance Attribute	Name	Access Rule	Data Type	Values	Description
	Configuration：		of		interface configuration.
	IP Address		UDINT	192.168.1.5	The device's IP address
	Network Mask		UDINT	255.255.255.0	The device's network mask
	Gateway Address		UDINT	0	Default gateway address
	Name Server		UDINT	0	Primary name server
	Name Server 2		UDINT	0	Secondary name server
	Domain Name		STRING	00 00	Default domain name
H'06	Host Name	Get	STRING	AX-8xxxxxxxx	Device name
H'13	Encapsulation Inactivity Timeout	Get/Set	UINT	120	EIP equipment connection time; unit: seconds; range of values: 0~3600

When the master is communicating, the instance attribute H'03 and H'05 cannot be written.

- **Example of reading and writing objects**

  - To read Instance Attribute H'03, write the data as shown below

    Service code：H'0E
    Class ID：H'F5
    Instance ID：H'01
    Attribute ID：H'03

  - To read Instance Attribute H'05, write the data as shown below

    Service code：H'10
    Class ID：H'F5
    Instance ID：H'01
    Attribute ID：H'05
    Data Byte[0~3]：IP Address=192.168.1.5
    Byte[4~7]：Network Mask=255.255.255.0
    Byte[8~11]：Gateway Mask=0.0.0.0
    Byte[12~15]：Name Server =0
    Byte[16~19]：Name Server2 =0

*1：Interface status

Status	Description
0	Interface Configuration attribute has not been configured.
1	The Interface Configuration attribute contains valid configuration obtained from BOOTP, DHCP or non-volatile memory.
2	The Interface Configuration attribute contains valid configuration obtained from hardware.

*2：Interface capability flags

Bit	Description
0	BOOTP Client
1	DNS Client
2	DHCP Client
3	DHCP-DNS Update
4	Configuration Settable
5	Hardware Configurable
6	Interface Configuration Change Requires Reset

*3：Interface Configuration Control

Status	Description
0	The device shall use the interface configuration values previously stored (for example, in non-volatile memory or via hardware witches).
1	The device shall obtain its interface configuration values via BOOTP.
2	The device shall obtain its interface configuration values via DHCP upon start-up.

9.4.4.6 Ethernet Link Object (Class ID：F6 Hex)

- **Service Code**

Service Code	Service Name	Attribute		Description
		Class Attribute	Instance Attribute	
H'0E	Get_Attribute_Single	V	V	Read a single attribute

- **Class**
  - Class ID：H'F6
- **Instance**
  - H'00：Class Attribute
  - H'01：Instance Attribute
  - When Instance = 0, Class Attribute are listed below:

Class Attribute	Name	Access Rule	Data Type	Values	Description
H'01	Revision	Get	UINT	H'04	Object revision

- When Instance =1, Instance Attribute are listed below:

Instance Attribute	Name	Access Rule	Data Type	Values	Description
H'01	Interface Speed	Get	DWORD	0	Interface speed（indeterminate）
H'02	Interface Flags	Get	DWORD	H'0D	Ethernet port status*1
H'03	Physical Address	Get	ARRAY of 6 USINTs	By Product	MAC address
H'0B	Interface Capability	Get	STRUCT of :	--	Capabilities of Ethernet interface*2
	Capability Bits		DWORD	H'02000000	The definition of Ethernet interface capability
	Speed/Duplex Options		STRUCT of :	--	The definition of speed and duplex options of Ethernet interface.
	Speed/Duplex		USINT	H'00	The count of speed/ duplex

Instance Attribute	Name	Access Rule	Data Type	Values	Description
	Array Count				options.
	Speed/Duplex Array		ARRAY of STRUCT of：	--	Speed and duplex settings
	Interface Speed		UINT	H’00	Ethernet interface speed. For example, 10 bps and 100 bps would be H’0A and H’64 accordingly.
	Interface Duplex Mode		USINT	H’00	Duplex mode capability of Ethernet interface. For example, half and full duplex would be H’00 and H’01 accordingly.

*1：Interface Flag Table

Bit(s)	Name	Description
0	Link Status	0 indicates an inactive link 1 indicates an active link
1	Half/Full Duplex	0 indicates half duplex 1 indicates full duplex
2-4	Negotiation Status	0：Auto-negotiation in progress 1：Auto-negotiation and speed detection failed 2：Auto negotiation failed but detected speed 3：Successfully negotiated speed and duplex 4：Auto-negotiation not attempted. Forced speed and duplex.
5	Manual Setting Requires Reset	shall be set zero
6	Local Hardware Fault	0 indicates the interface detects no local hardware fault 1 indicates a local hardware fault is detected
7-31	Reserved	0

*2：Interface Capability Bits

Bit (s)	Name	Description
0	Manual Setting Requires Reset	Indicates whether or not the device requires a reset when instance attribute #6 (Interface Control attribute) changes. 0 indicates the device does not require a reset 1 indicates the device requires a rest
1	Auto-negotiate	0 indicates the interface does not support auto-negotiation 1 indicates the interface supports auto-negotiation
2	Auto-MDIX	0 indicates the interface does not support auto MDIX operation 1 indicates the interface supports auto MDIX operation
3	Manual Speed/Duplex	0 indicates the interface does not support to set speed/duplex. (Instance attribute #6, Interface Control attribute) 1 indicates the interface supports to set speed/duplex
4-31	Reserved	shall be set 0

9.4.5 Delta EIP Product List

9.4.5.1 Delta EIP Product List (Adapters Supported)

Positioning	Product	Version
Mid-range PLC	AHCPU501-EN、AHCPU511-EN、AHCPU521-EN、AHCPU531-EN	V2.00
	AHCPU560-EN2	V1.00
	AH10EN-5A	V2.00
	AHRTU-ETHN-5A	V1.00
	AH10EMC-5A	V1.00
	AS300 Series	V1.00
	AS200 Series	V1.00
	AS300 Series（AS-FEN02 communication card）	V1.06（V1.00）
	AS00SCM-A（AS-FEN02 communication card）	V2.02（V1.00）
	AX-3 Series	V1.01
	AX-8 Series	V1.01
Small PLC	DVPES2-E Series	V3.60
	DVP26SE	V1.00
	DVP-ES3 Series	V1.00
Inverter	CMM-EIP01/02 communication card（VFD-M300 Series）	V1.00
	CMC-EIP01 communication card（VFD-C2000 Series）	V1.06
	CMM-EIP03 communication card（VFD-M300 Series）	V1.00
	CMC-EIP02 communication card（VFD-C2000 Series）	V1.00

9.4.5.2 Delta EIP Product List (Scanners Supported)

Positioning	Product	Version
Mid-range PLC	AHCPU501-EN、AHCPU511-EN、AHCPU521-EN、AHCPU531-EN	V2.00
	AHCPU560-EN2	V1.00
	AH10EN-5A	V2.00
	AS300 Series、AS200 Series	V1.00
	AX-3 Series	V1.01
	AX-8 Series	V1.01
Small PLC	DVP-ES3 Series	V1.00

9.5 Network Security

We suggest you to use closed network or use local network with a firewall to secure and prevent the Ethernet network as well as our products from any unwanted attacks.

MEMO

A

Appendix A Troubleshooting

Table of Contents

A.1 Troubleshooting

A.1.1 Basic Troubleshooting Steps

This section introduces the types of errors which might occur during operation and their causes as well as resolutions. Complete the checklist below before looking further into the errors.

1. Check the following items:
   - The controller is operating under the specified conditions (considering the environmental, electronic, vibration safety and so on).
   - Power supply is connected properly to the controller.
   - The modules, terminals and cables are installed correctly.
   - All LED indicators work properly.
   - Set all the switches correctly.
2. Check the operational functions:
   - Check and eliminate errors from external devices.
   - Check system operation and logs through System Log in DIADesigner-AX.
3. Identify possible causes based on examination in point 1 and 2.
   - AX-8 Series
   - CPU
   - Parameter or program settings

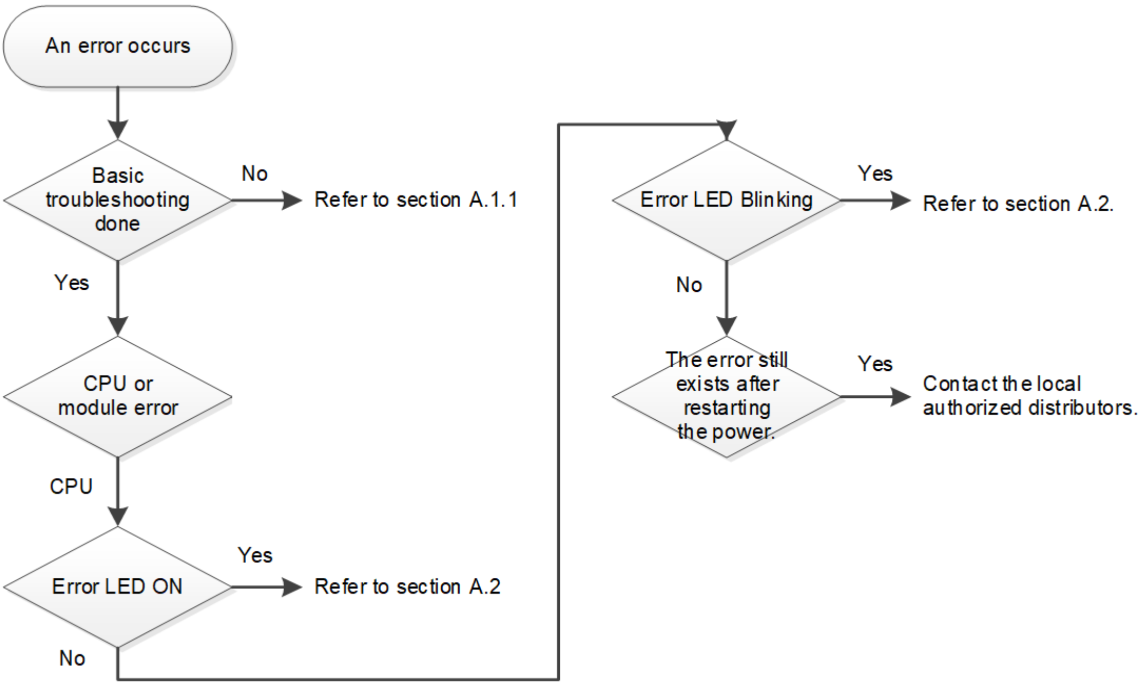
A.1.2 Clear Error States

This section introduces the methods to clear the error states. The system will keep going into error mode if the source errors are not resolved.

1. Turn off the CPU and restart again.
2. Perform **Reset Warm** to clear the error logs through DIADesigner-AX.
3. Perform **Reset Origin** to reset CPU to default and redownload the program through DIADesigner-AX.

A

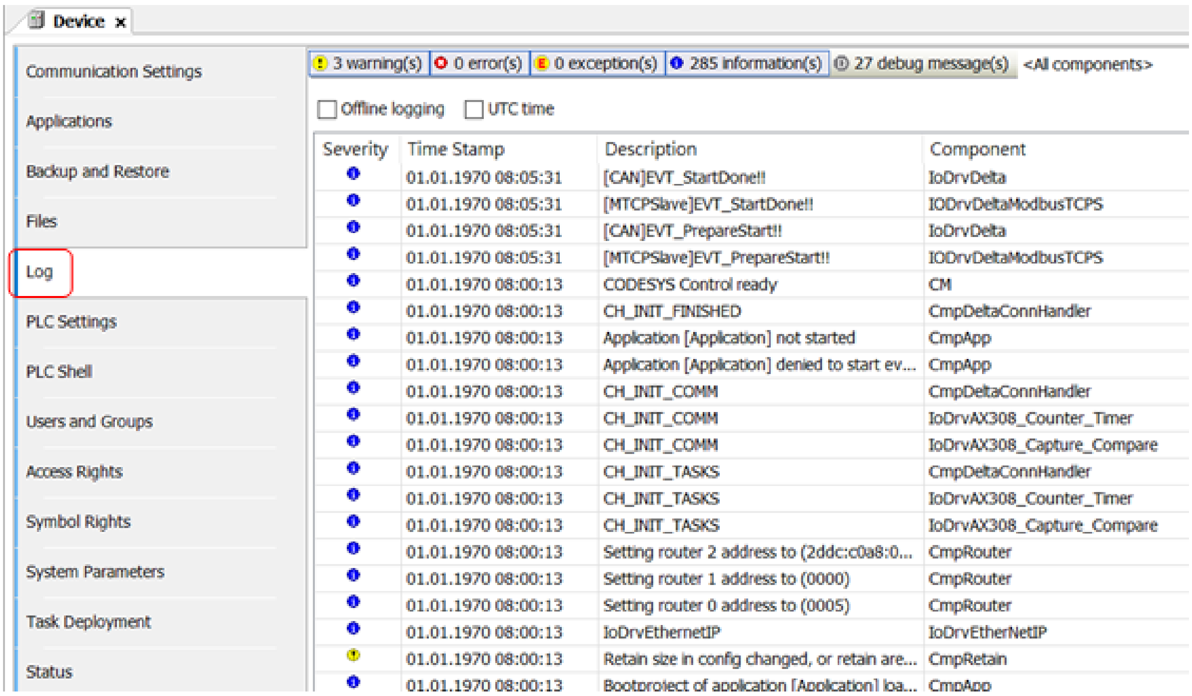
A.1.3 Troubleshooting SOP



A.1.4 Viewing Log

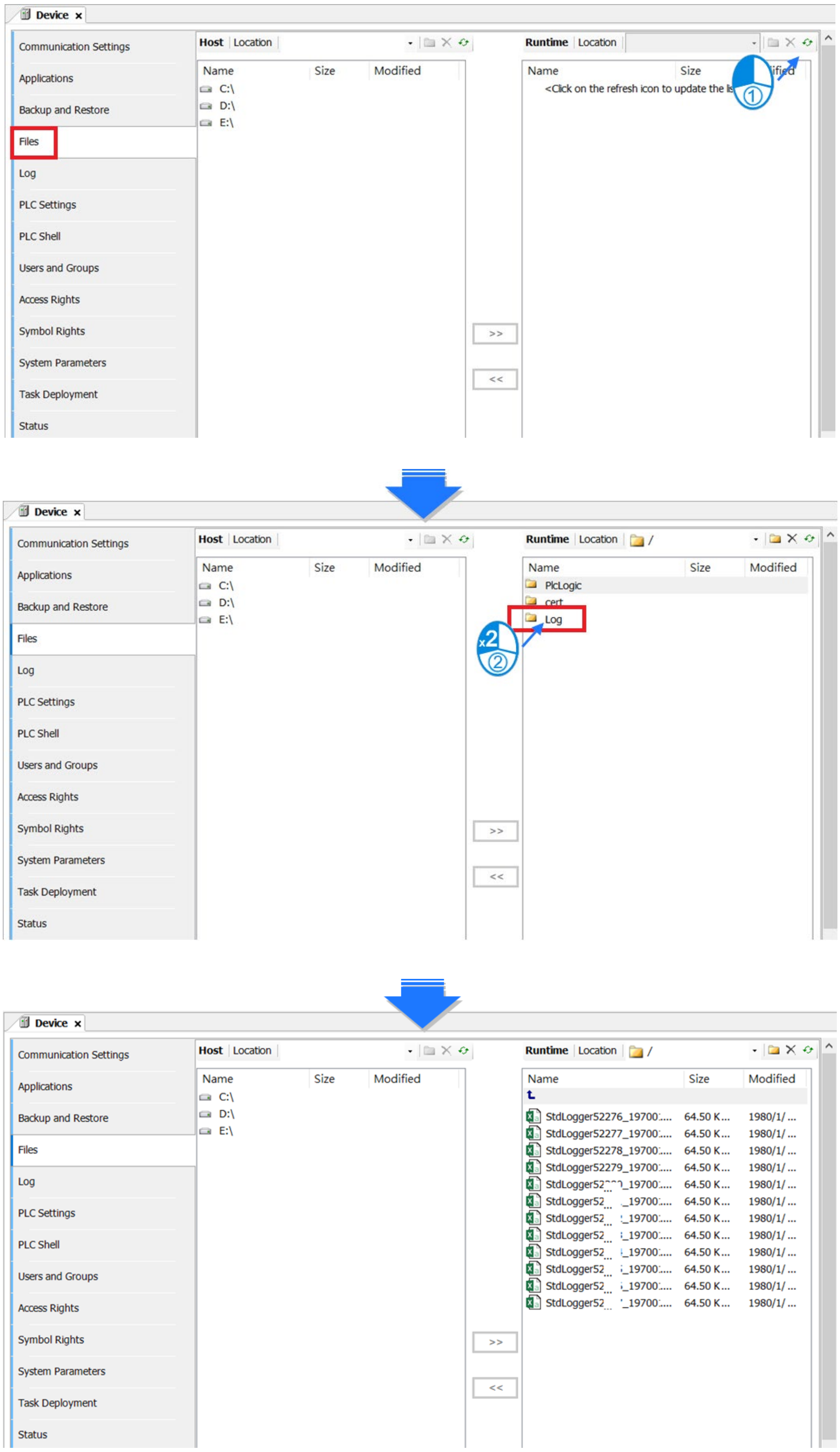
When an error occurs, the corresponding error codes are generated and stored in the controller. On the **Log** tab of the **Device** pane, you are able to check all the events recorded during system startup and shutdown, application download and loading of boot application, custom entries, log entries from I/O drivers and from data source. Refer to **section 4.2.1.5** for more information on Log.

1. Log Tab (**Device** > **LOG**)



2. Files

Log files (.csv) are generated after shutdown of the controller or when the log exceeds 64KB. The log files can be found on the **Files** tab of the **Device** pane.



A.2 Troubleshooting of CPU Modules

Check the LED indicators and the error codes from the CPU modules, and refer to the following table for troubleshooting.

A.2.1 ERROR LED Indicators Blinking Every 0.5 Seconds

- **CPU error**

Error Code (16#)	Description	Solution
0x2000	CPU memory access is denied	If the problem persists, contact the local authorized distributors.
0x2001	CPU external memory access is denied	If the problem persists, contact the local authorized distributors.
0x2002	Either OS file or firmware-related file is corrupted or system backup is activated	If the problem persists, contact the local authorized distributors or update the firmware.
0x2005	GPIO failed to initialize	If the problem persists, contact the local authorized distributors or update the firmware.
0x2006	Semaphore failed to establish	If the problem persists, contact the local authorized distributors or update the firmware.
0x2007	Task failed to establish	If the problem persists, contact the local authorized distributors or update the firmware.
0x2008	Unable to access data file	If the problem persists, contact the local authorized distributors or update the firmware.
0x2009	PowerIC failed to initialize	If the problem persists, contact the local authorized distributors or update the firmware.
0x200A	FPGA done condition mismatched	If the problem persists, contact the local authorized distributors or update the firmware.
0x200B	Ethernet IP failed to initialize	Redownload the project and select **Apply IP settings while download.**
0x200C	PLC ID access denied	If the problem persists, contact the local authorized distributors or update the firmware.
0x200D	Error detected in the check sum of %M device	1. Check whether the 24V power supply is sufficient or not. 2. Amplify 24V power supply if the problem persists.
0x200E	The MRAM position set by the cfg file and FW do NOT match.	If the problem persists, contact the local authorized distributors

0x200F	Codesyscontrol file is missing or corrupted.	If the problem persists, contact the local authorized distributors
0x2100	The number of MODBUS TCP connections exceed the range.	Please refer to **section 2.2 CPU Module Specifications** and adjust the connection to the maximum.
0x3000	Hardware access error	If the problem persists, contact the local authorized distributors.
0x3002	SSD lifespan reached limit.	If the problem persists, contact the local authorized distributors

A.2.2 ERROR LED Indicators Blinking Rapidly Every 2 Seconds

Error Code (16#)	Description	Solution
0x2004	The external 24V power supply is in low voltage.	Check whether the power supply connected to the module is stable or not.

A.2.3 ERROR LED Indicators Blinking Rapidly Every 1 Second

Error Code (16#)	Description	Solution
0x3001	Factory test info error	If the problem persists, contact the local authorized distributors.

A

A.3 Troubleshooting of Function Blocks

A.3.1 DL_CmpAX8xxEDrv

The following errors are specified as warnings. No error indicators will appear and the CPU can still run.

Error code (16#)	Item	Description	Solution
0	DFB_NO_ERROR	No error	N/A
0x5	DFB_CAP_SOURCE_OUT_ RANGE	Capture trigger source is out of range.	Source range is between **BuiltIn_DIO** X0~X3
0x6	DFB_CAP_SOURCE_IS_ACTIVE	Capture trigger source is already active.	Please check if any other function block has used the same trigger source.
0x7	DFB_PWM_INPUT_OUT_RANGE	DFB_PWM Input is out of range or values are invalid.	Please check if the parameters of DFB_PWM are correct
0x8	DFB_PWM_DO_OTH_FUN_ ACT	DO is already in use for other function.	Please check if any other function block has used the same DO for other function or PWM is already active in other function block.
0x9	DFB_UDC_COUNTER_NUM_MISSMATCH_MODE	Counter number (0~7) mismatched with counter mode.	Please check if the counter number has supported this counter mode.
0xA	DFB_UDC_MODE_OUT_RANGE	Counter mode is out of range.	Please check if the parameters of DFB_Counter are correct.
0xB	DFB_UDC_COUNTER_NUM_IS_ACT	Channel of counter number has been active or counter mode is already set by other function block.	Disable other DFB_Counter function blocks or set up the same settings.
0xC	DFB_TCMP_FIFO_FULL	Compare FIFO position table is full.	Please reset FIFO table.
0xD	DFB_TCMP_OS_TIME_OUT_RANGE	**rOneShotTime** or **byTimeExten** is out of range.	**rOneShotTime** range of 5000 ~ 0.0003(ms) **byTimeExten** range of 1~15
0xE	DFB_TCMP_DO_OTH_FUN_ACT	DO is already in use for other function.	Please check if any other function block has used the same DO for other functions or PWM is already active in other function block.

Error code (16#)	Item	Description	Solution
0xF	DFB_TCMP_FIFO_WRITE_ACT	FIFO table is writing.	Please check if any other function block has used other functions.
0x10	DFB_TCMP_ACT	TCMP function is active.	Please check if any other function block has used other functions.
0x11	DFB_TCMP_TABLE_SIZE_OUT_RANGE	TCMP table size is out of range.	Range: 1~255
0x12	DFB_DVR_GET_VER_ERROR	Get device version error	Not supported in FPGA version
0x14	DFB_SRT_TYPE_ERROR	Retain type error	Please check if the parameter Retain Type of DFB_SetRetainType is correct.
0x15	DFB_SRD_PASSWORD_ERROR	Password value error	Please check if the parameter sPassword of DFB_SetRestorePwd is correct.
0x16	DFB_SIPC_LAN_PORT_INVAILD	Lanport number error	Please check if the parameter LanPort of DFB_SetIPConfig is correct.
0x17	DFB_SIPC_MASK_INVAILD	Mask address value error	Please check if the parameter SubnetMask of DFB_SetIPConfig is correct.
0x18	DFB_SIPC_GATEWAY_INVALID	Gateway address value error	Please check if the parameter Gateway of DFB_SetIPConfig is correct.
0x19065	DFB_SSI_INPUT_OUT_RANGE	DFB_SSI_Encoder Input is out of range or values are invalid.	Please check if the parameter of DFB_SSI_Encoder is correct.
0x19066	DFB_SSI_CONF_ERROR	Serial communication mode config error	Change device parameter **Serial communication mode** settings to SSI.
0x19067	DFB_SSI_NOT_SUPPORTED	Device does not support this function.	Check if the FPGA version is v0x25 or later.
0x19068	DFB_SSI_IS_ACTIVE	SSI Encoder is already active	SSI Encoder is already active.

A.3.2 Motion Control Related Instructions

The errors which occur in DL_MotionControl or DL_MotionControlLight are specified as warnings; however, no error indicators will appear and the CPU can still run. Refer to **AX Series Motion Controller Manual** for the troubleshooting of DL_MotionControl.

B

Appendix B BIOS Settings

Table of Contents

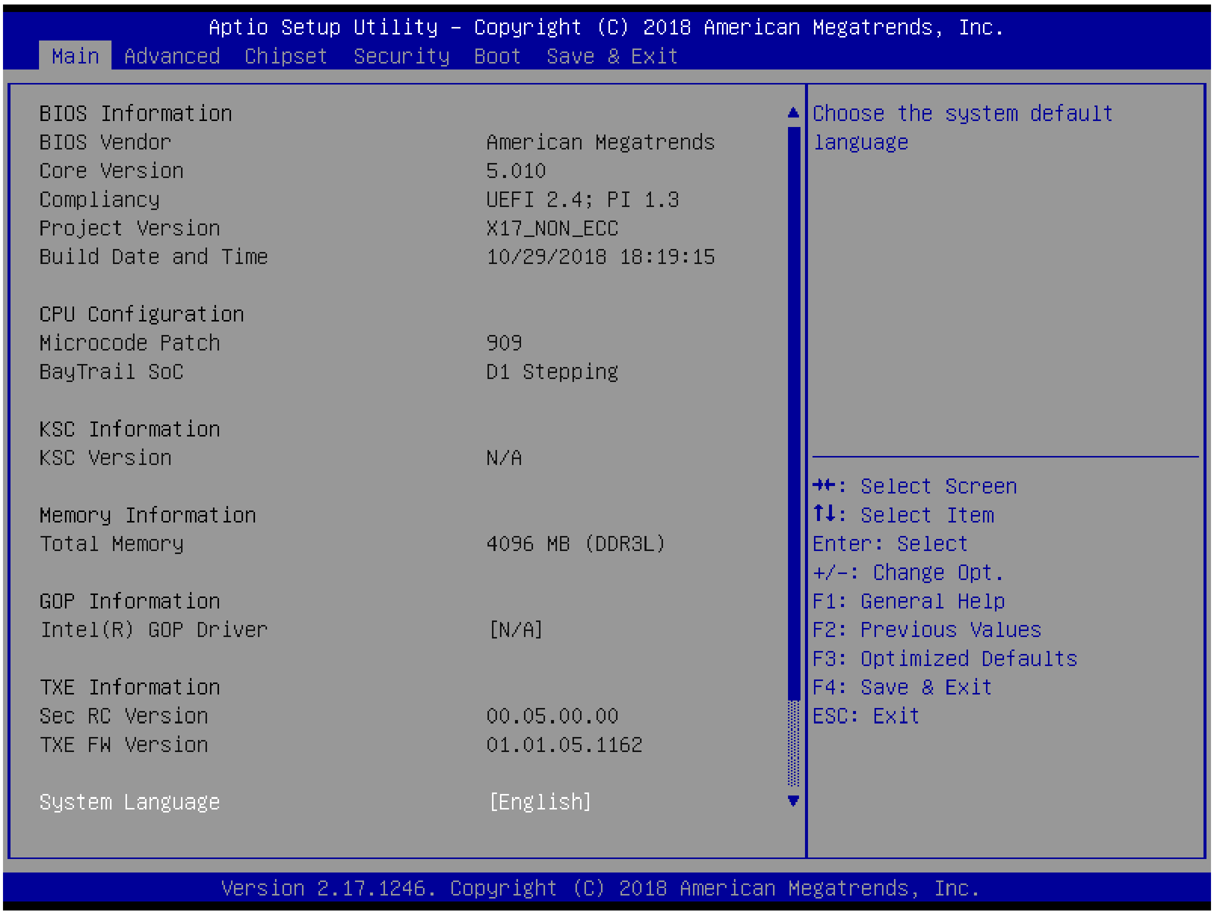
B.1 Basic Settings and Operation of BIOS

When the computer boots up, following with the message **Press <Del> or <F2> to enter setup** on the screen, click either the **Del** key or the **F2** key to enter BIOS settings.

1. BIOS keyboard shortcuts

Key	Function	Key	Function
↑↓←→	Move among items	F1	Keys operation support
Enter	Enter or select the current item	F2	Restore device to the previous settings
+ -	Adjust values	F3	Restore device to factory settings
Esc	Escape	F4	Save current settings

2. Menu bar

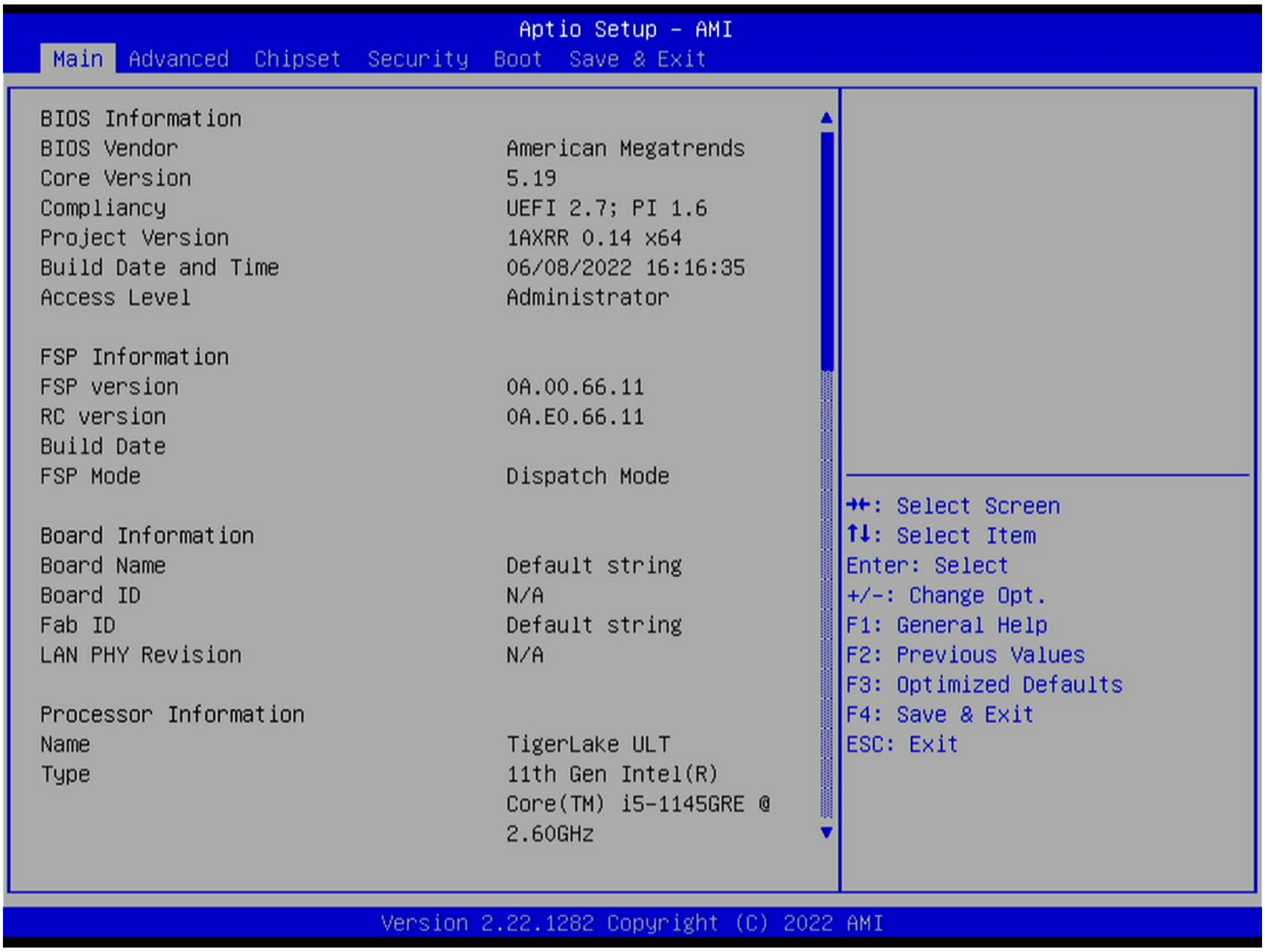


Menu	Function	Menu	Function
Main	Basic settings	Security	Security settings
Advanced	Settings of advanced functions	Boot	Startup settings
Chipset	Chipset settings	Save & Exit	Save changes and exit

(Use the ← 、→ key to browse each menu.)

B.1.1 Main

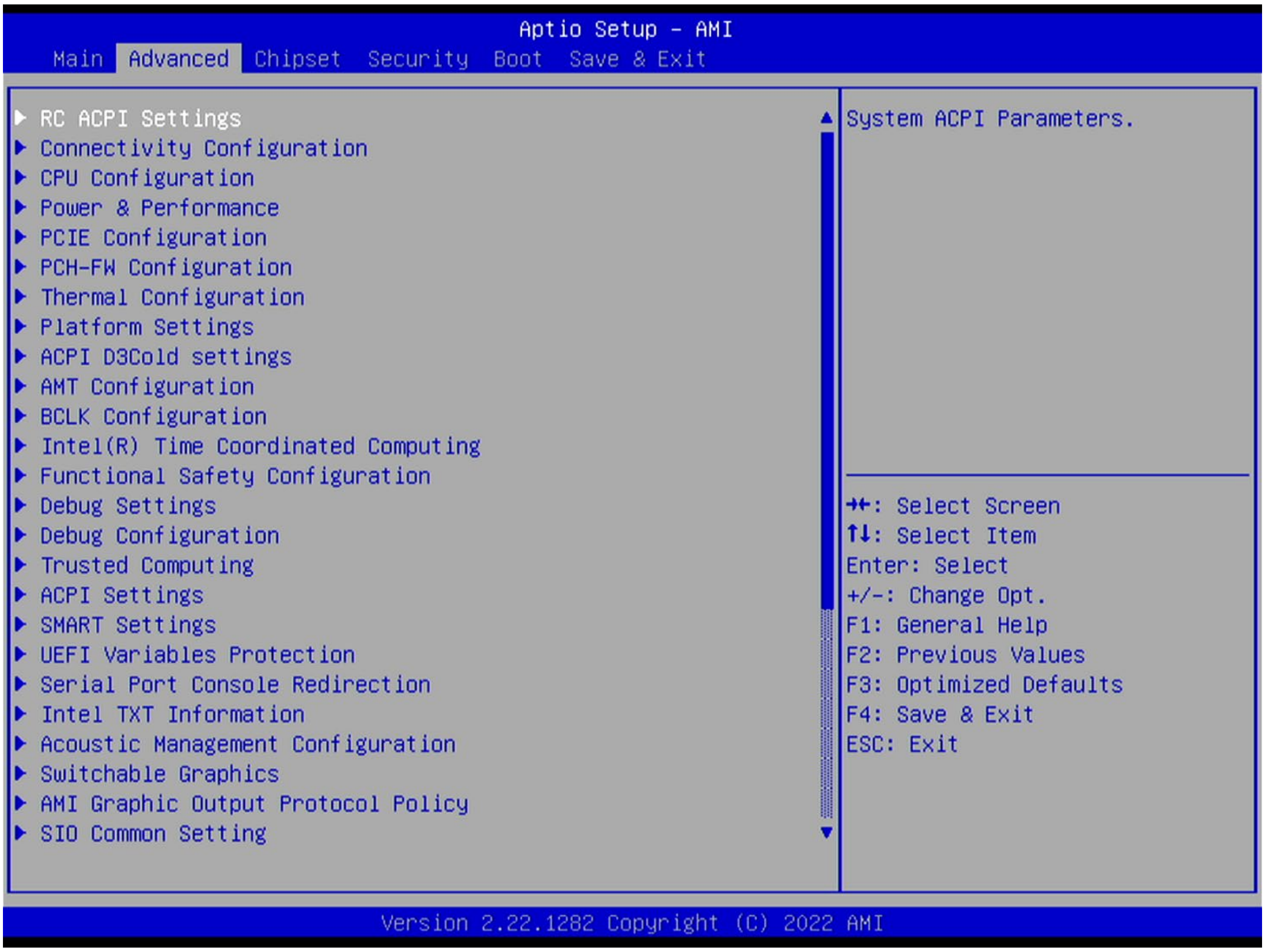
Here you can find information such as Total Memory, System Language, etc. in the Main menu:



Item	Default	Description
System Language	English	N/A
System Date	N/A	Settings of system date
System Time	N/A	Settings of system time

B.1.2 Advanced

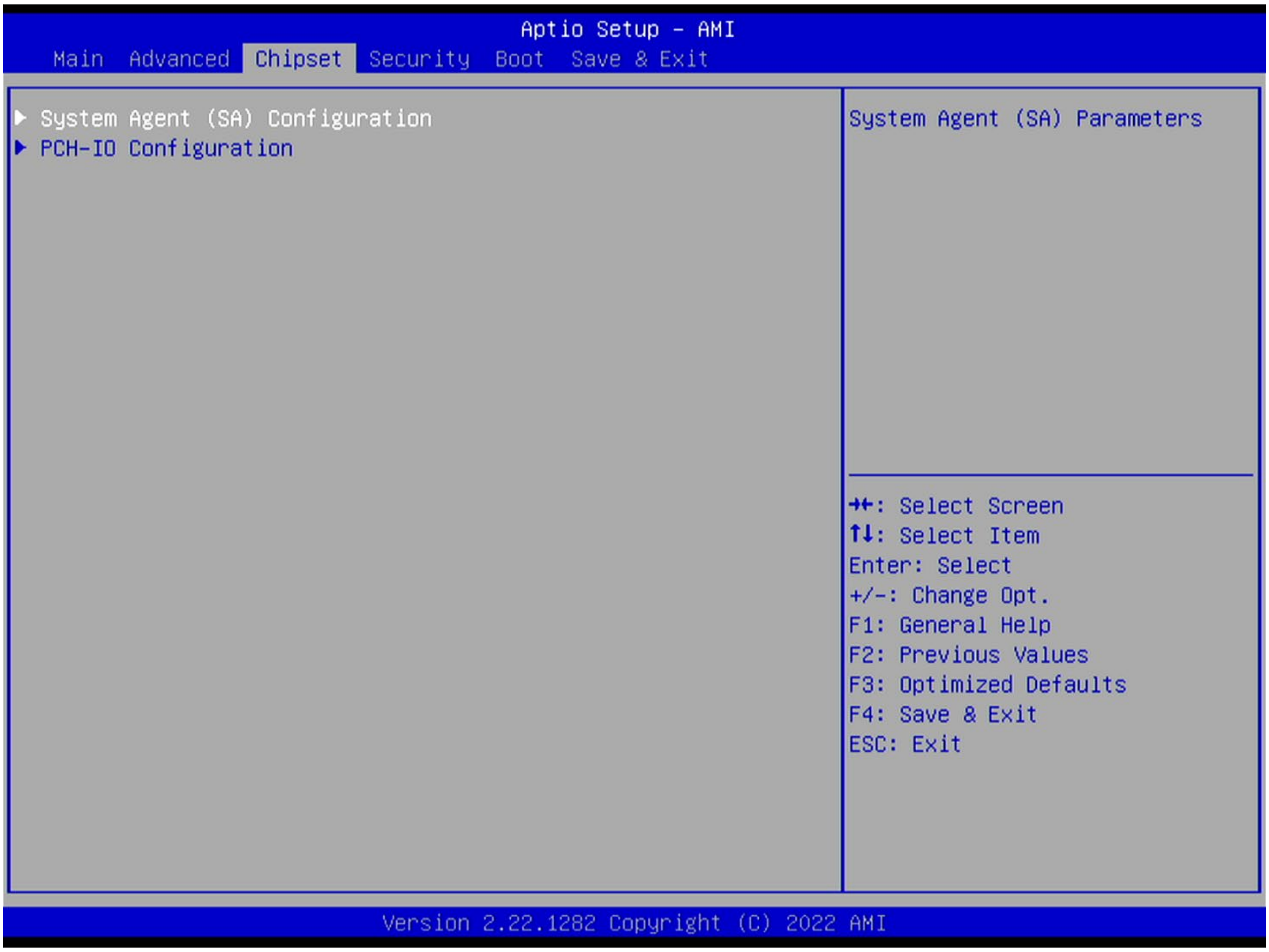
Here you can find information such as CPU Configuration, Power & Performance, etc. in the Advanced menu.



Item	Default	Description
CPU Configuration	N/A	Settings of CPU parameters
Power & Performance	N/A	Settings of CPU and GPU power performance
Thermal Configuration	N/A	Settings of Thermal parameters
Intel® Time Coordinated Computing	N/A	Settings of Intel TCC
ACPI Setting	N/A	Settings of ACPI
USB Configuration	N/A	Settings of USB parameter

B.1.3 Chipset

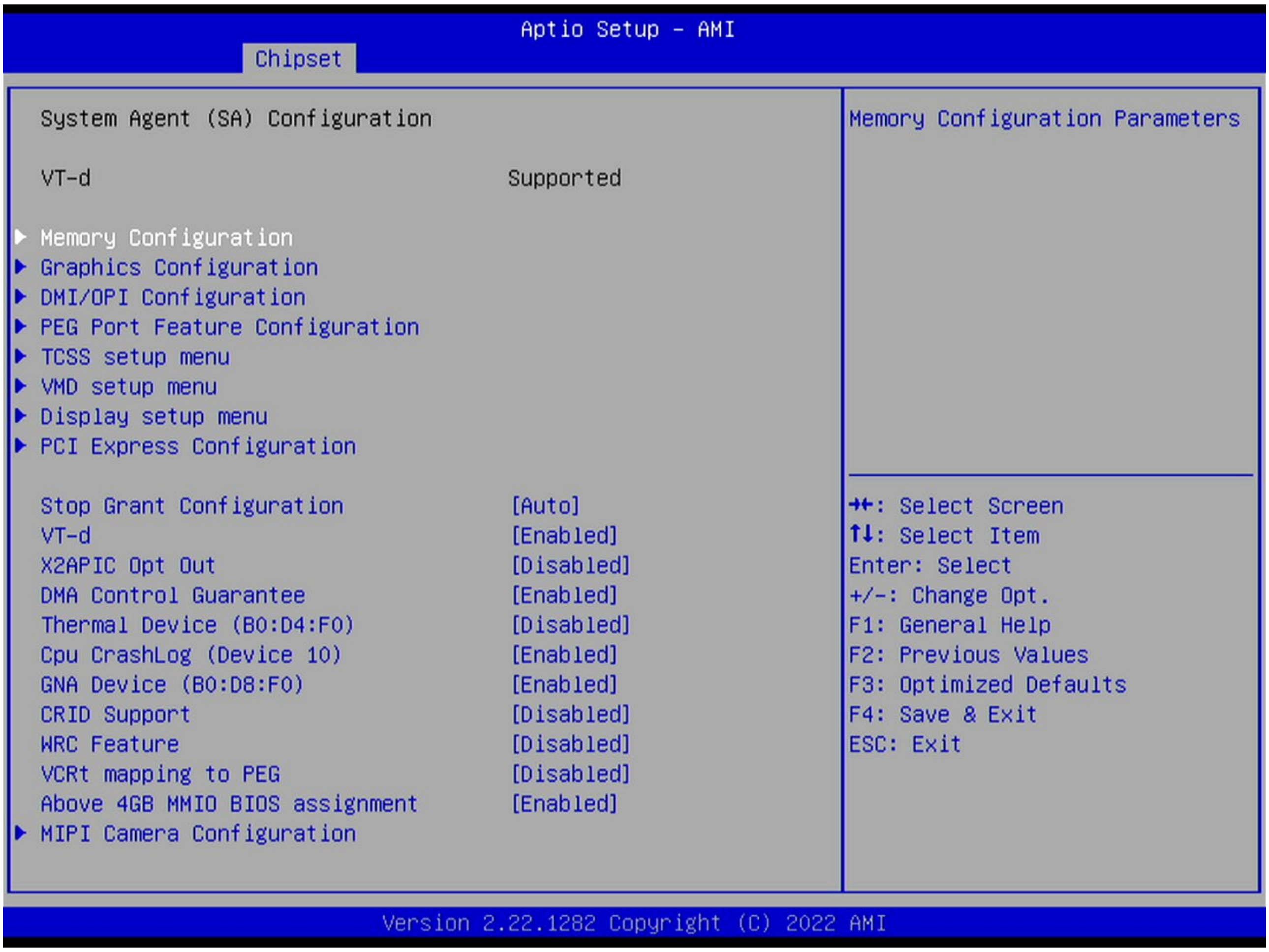
Here you can find information such as System Agent Configuration, PCH-IO Configuration, etc. in the Chipset menu:



Item	Default	Description
System Agent Configuration	N/A	Settings of system agent
PCH-IO Configuration	N/A	Settings of PCH-IO

- **System Agent Configuration**

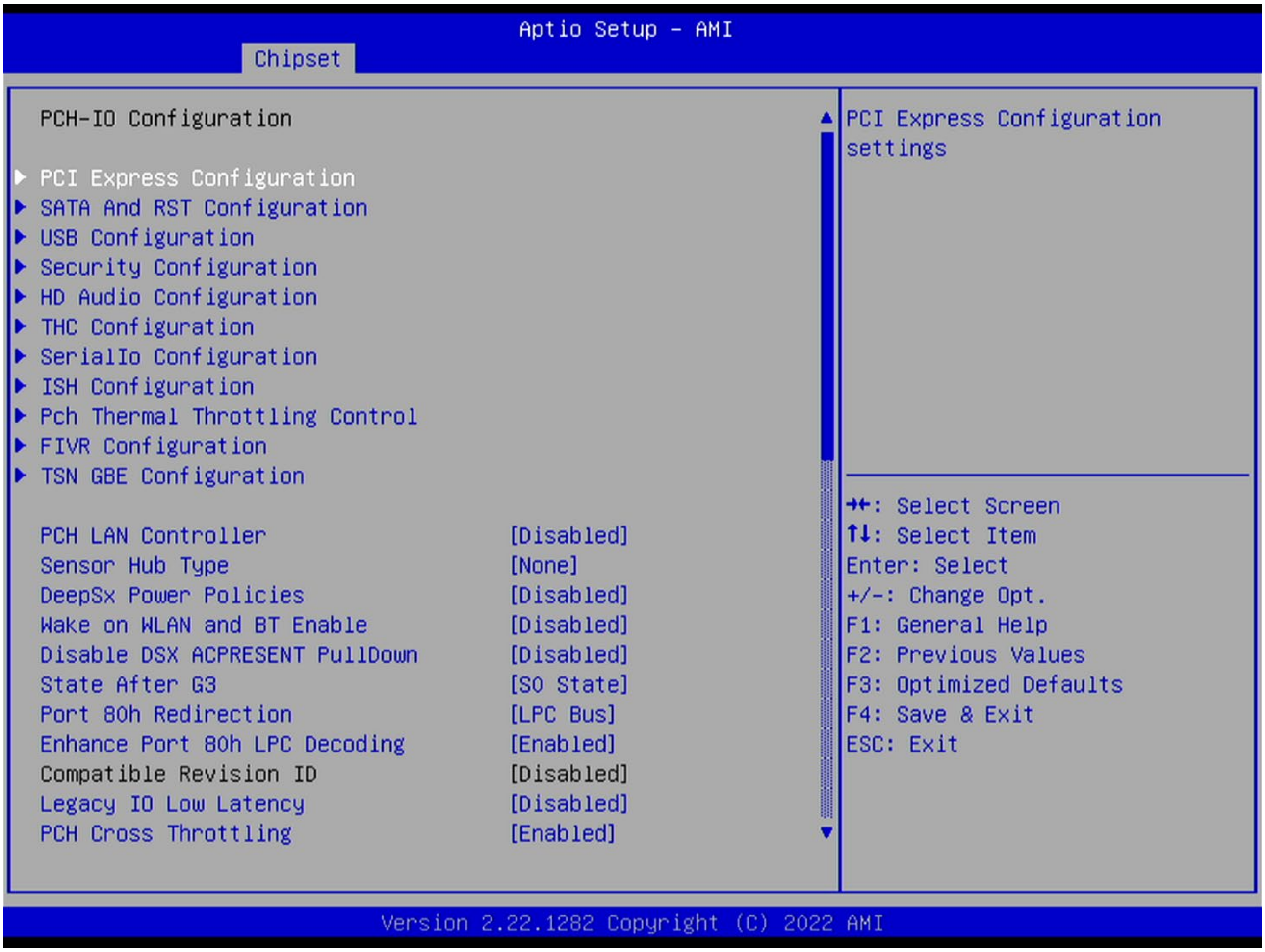
Here you can find information such as Memory Configuration and Graphic Configuration in the **System Agent Configuration** view:



Item	Default	Description
Memory Configuration	N/A	Settings of memory parameters
Graphics Configuration	N/A	Settings of graphics output
Display setup Menu	N/A	Settings of display interface

- **PCH-IO Configuration**

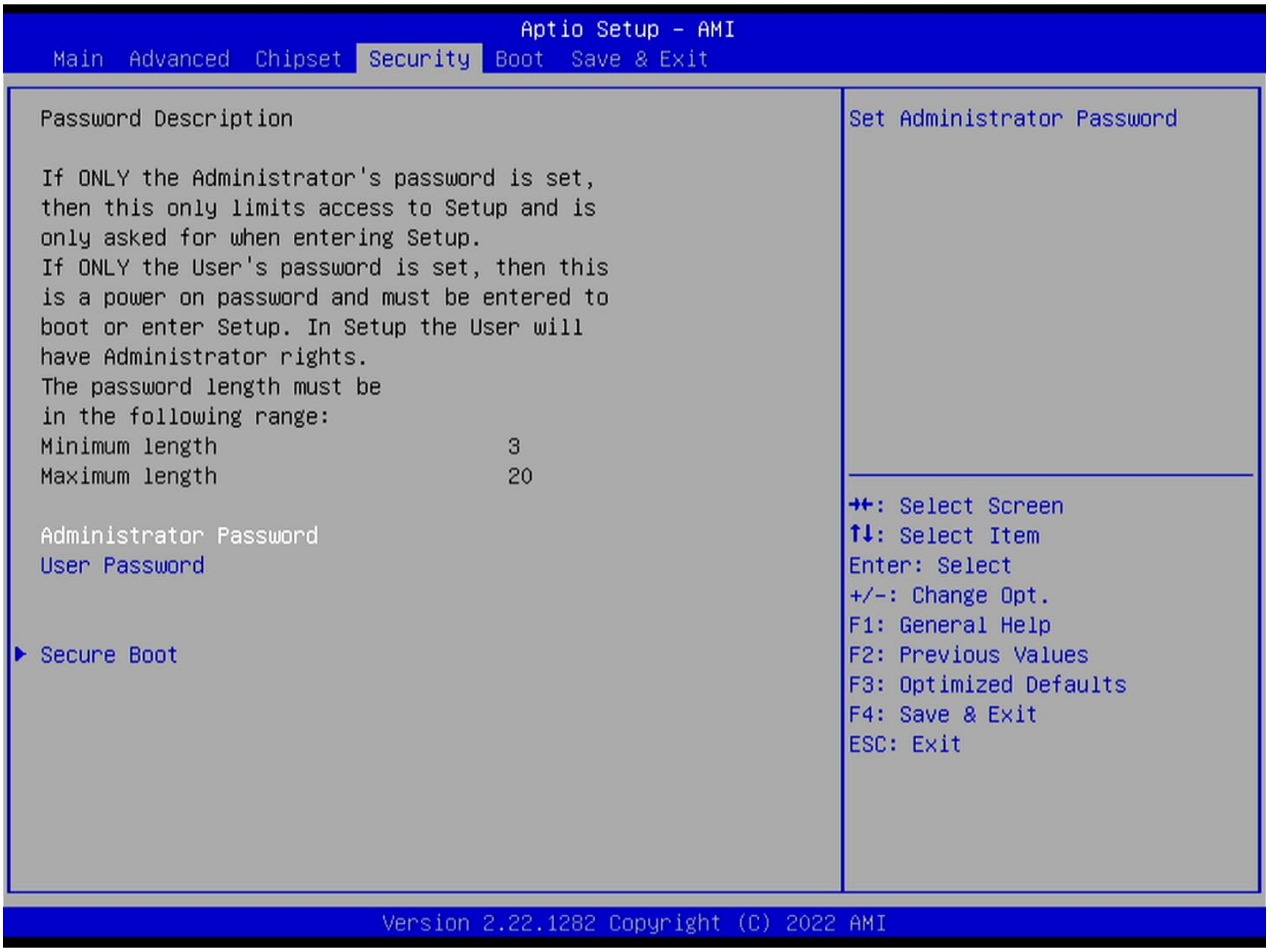
Here you can find information such as PCIE, SATA and USB Configuration in the PCH-IO Configuration view:



Item	Default	Description
PCI Express Configuration	N/A	PCIE settings
SATA And RST Configuration	N/A	SATA settings
USB Configuration	N/A	USB settings
Serial IO Configuration	N/A	Serial I/O settings
Status After G3	S0 Status	Settings of auto startup in power-on state

B.1.4 Security

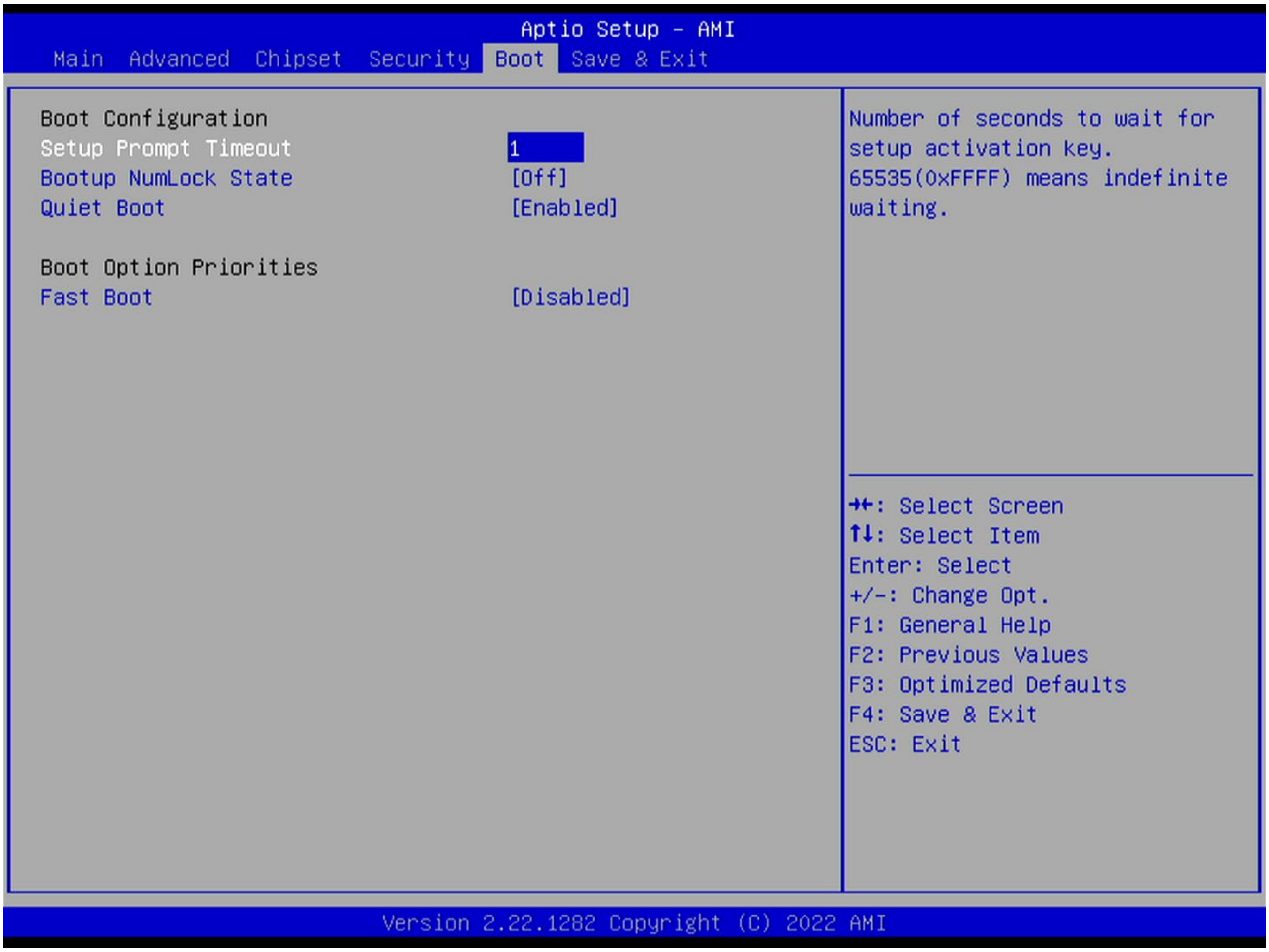
Here you can find information such as Administrator Password and User Password in the Security menu:



Item	Default	Description
Administrator Password	N/A	Settings of the administrative password
User Password	N/A	Settings of the user password
Secure Boot	N/A	Secure boot menu

B.1.5 Boot

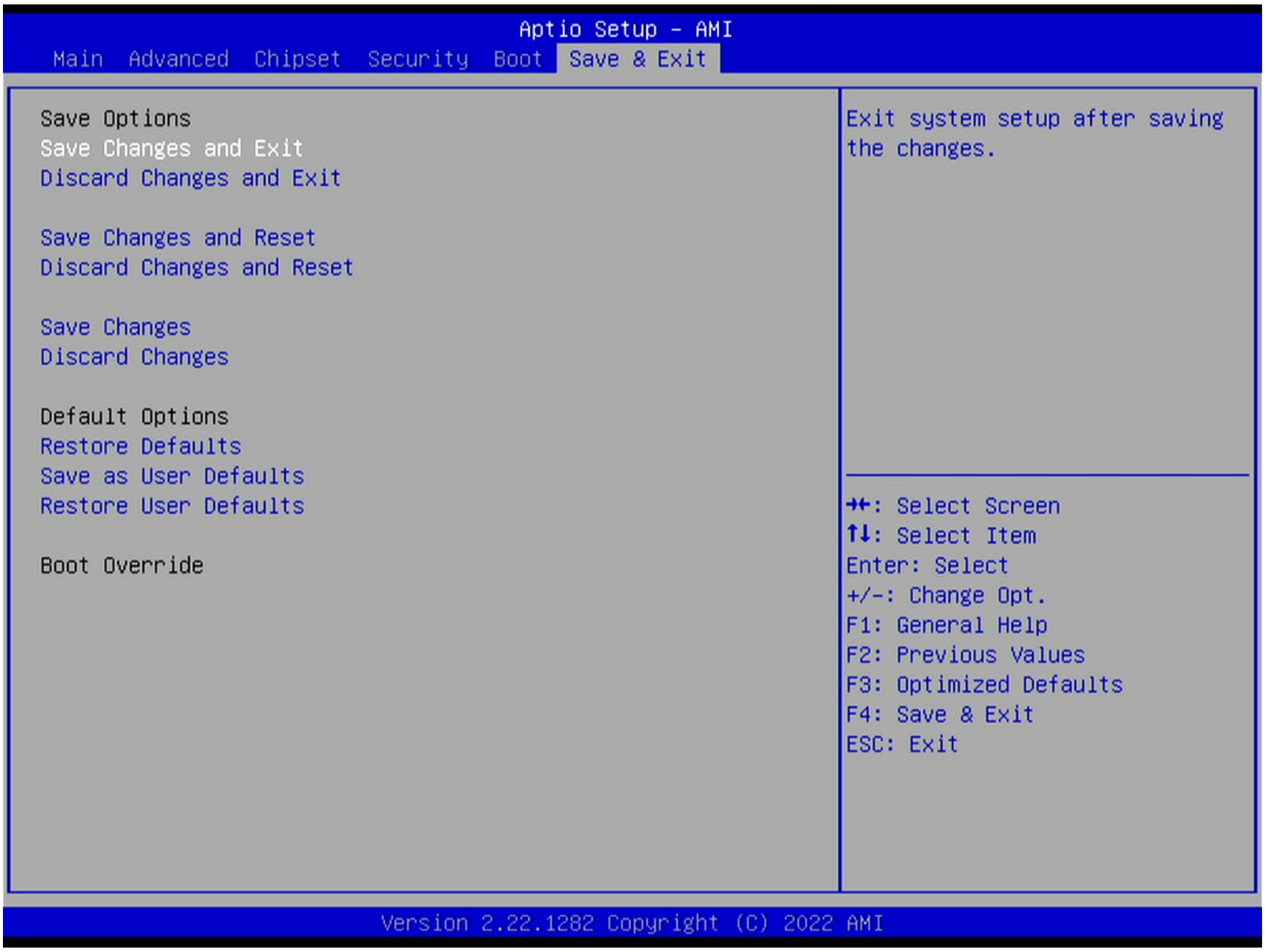
Here you can find information such as Setup Prompt Timeout, Bootup NumLock State, etc. in the Boot menu:



Item	Default	Description
Setup Prompt Timeout	1	N/A
Bootup NumLock State	off	N/A
Quiet Boot	Enabled	N/A
Boot Option Priorities	N/A	Setting the boot order for devices
Fast Boot	Disabled	N/A

B.1.6 Save & Exit

Here you can find information such as Save Changes and Exit, Discard Changes and Exit, etc. in the Save & Exit menu:



Item	Default	Description
Save Changes and Exit	N/A	Save the current changes and exit
Discard Changes and Exit	N/A	Restore device to the previous settings and exit
Save Changes and Reset	N/A	Save the current changes then reset the PC
Discard Changes and Reset	N/A	Restore device to the previous settings then reset the PC
Save Changes	N/A	Save the current changes
Discard Changes	N/A	Restore device to the previous settings
Restore Defaults	N/A	Restore device to factory settings
Save as User Defaults	N/A	Save the current user settings as defaults
Restore User Defaults	N/A	Restore the user defaults
Boot Override	N/A	Force reboot the device

C

Appendix C Settings and Operation of Windows/Linux

Table of Contents

C.1 Basic Settings of UWF

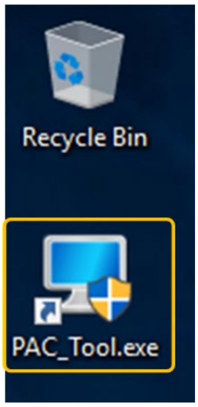
In factory settings, the Unified Write Filter (UWF) is turned on to protect Windows system from file corruptions in abnormal power outages or crashes.

If you need to turn off UWF for software installation or parameter adjustment reasons, be sure to turn it on again right after the installation or settings are completed.

When UWF is switched off, the LED indicator on AX-8 Series PLC, FB1 and FB2, will flash consecutively to alert the user that UWF is currently off.

C.1.1 UWF Execution via PAC_Tool

The main function of PAC_Tool is to protect the C drive in Windows system. When UWF is turned on, the data written to the C drive will be stored in the memory and it will be removed once the device is rebooted. To perform this function, click **PAC_Tool.exe** on the computer desktop.



C.1.2 Read Current Status

Open PAC_Tool and the **Current Status** will be shown as below:

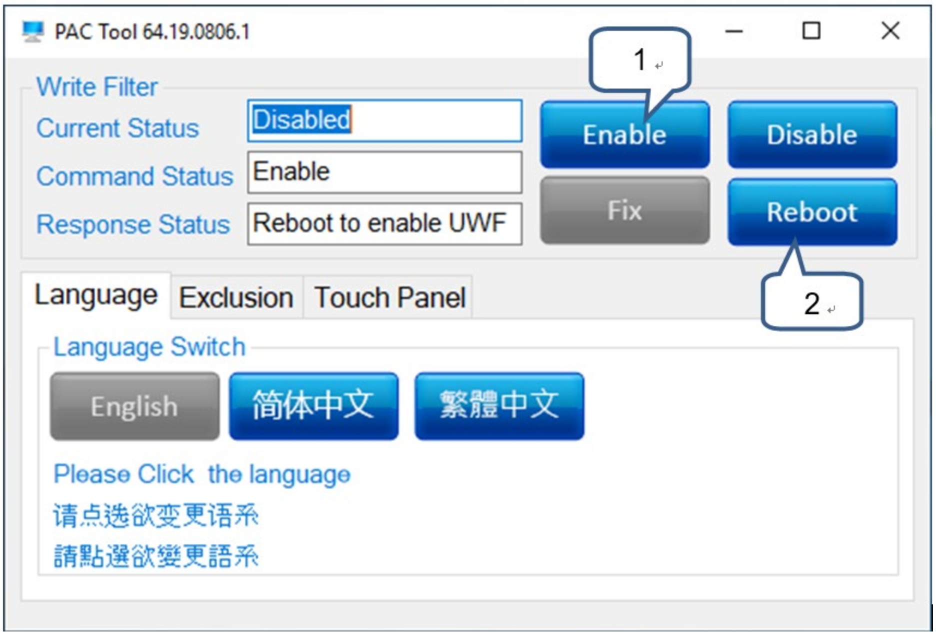
- **Disabled:** UWF is currently turned off and changes will be saved after power outages.

- **Enabled:** UWF is currently turned on and changes will not be saved after power outages.

C.1.3 Enable UWF

Refer to the following steps to activate UWF.

1. Click **Enable.**
2. Click **Reboot** to restart.



C.1.4 Disable UWF

Refer to the following steps to deactivate UWF.

1. Click **Enable**.
2. Click **Reboot** to restart.

C.1.5 Fix UWF

Refer to the following steps to fix UWF.

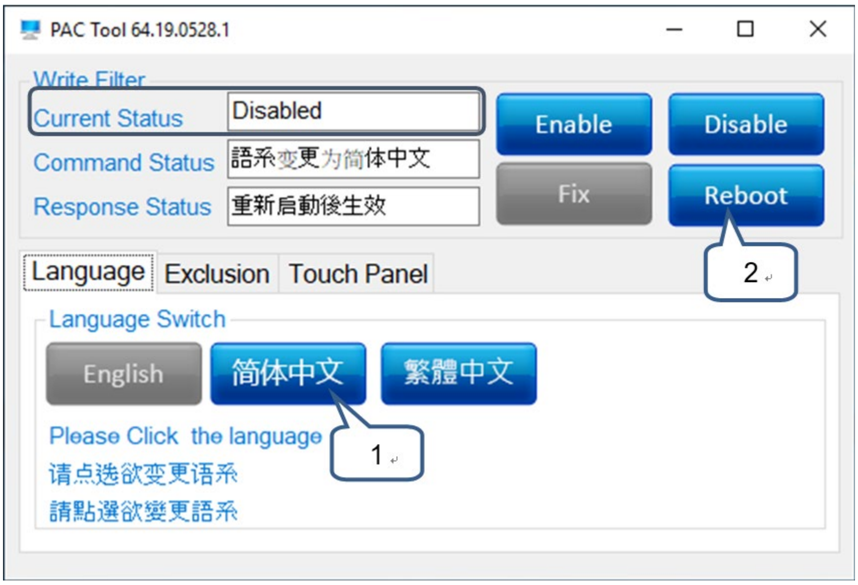
1. Click **Fix**.
2. Click **Reboot** to restart.

C.2 Display Language Settings

Refer to the following steps to switch display languages.

- If the Current Status is **Disabled**:

  1. Select the preferred language.
  2. Click **Reboot** to restart.



- If the Current Status is **Enabled**, you should turn off UWF in advanced.

  1. Click **Disable**.
  2. Click **Reboot** to restart.
  3. Select the preferred language.
  4. Click **Reboot** to restart.

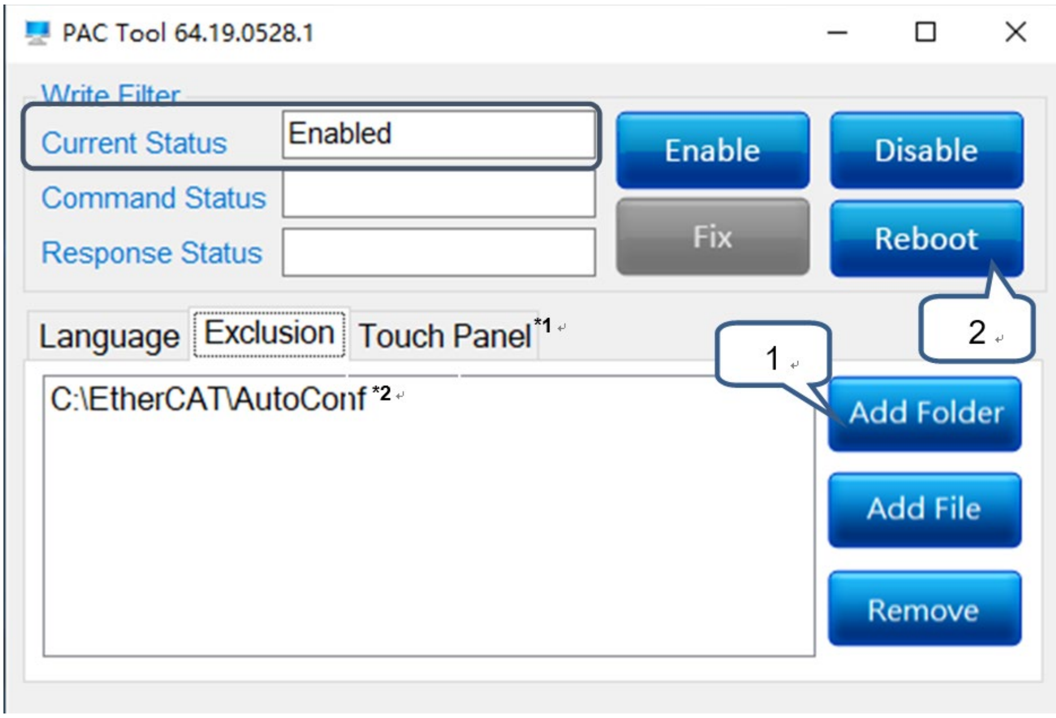
C

C.3 UWF Exclusion Settings

Refer to the following steps to exclude folders or files from write protection while the UWF is enabled.

- If the Current Status is **Enabled.**

  1. Click **Add Folder** to select the folders or files to be excluded.
  2. Click **Reboot** to restart.



*1: The Touch Panel tab is not available here since it is exclusively for Panel devices.

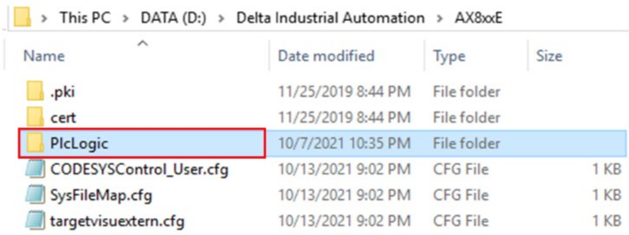
*2: In some models the storage path is set to be in the D drive; hence, files and data could be lost due to unexpected shutdown of the device.

C.4 Replace Original Project with USB Backup Project

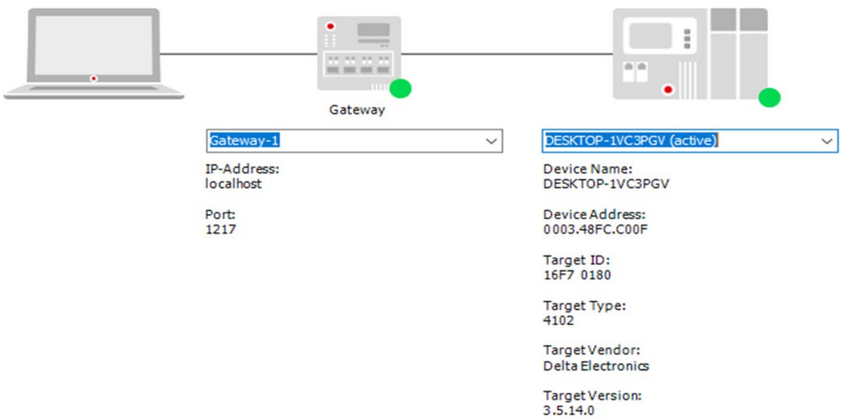
This section introduces how to create an USB backup project and the way to replace the ongoing project in AX-8 Series PLC with the USB backup project.

C.4.1 Retrieve a Project from PLC

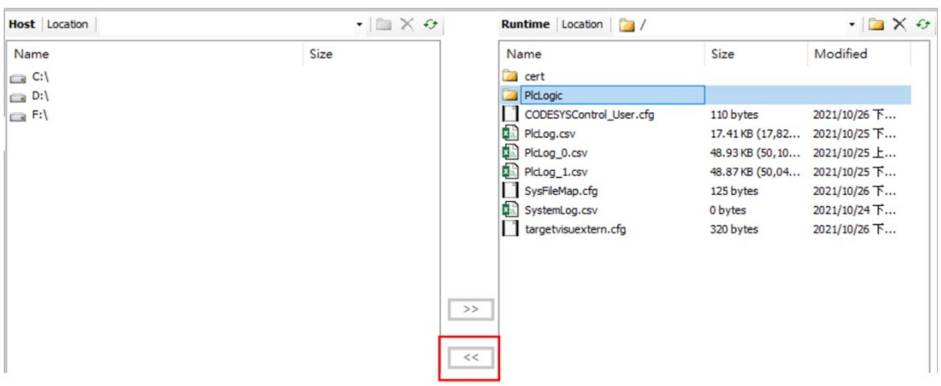
1. Directly go to Windows desktop or DIADesigner-AX software to retrieve the project.
2. Enter Windows desktop via AX-8 Series PLC HDMI connection or remote desktop.
3. PLC project is stored in the PlcLogic folder (path: D:\Delta Industrial Automation\AX8xxE \PlcLogic).



4. The PlcLogic folder of AX-8 Series PLC also can be accessed through DIADesigner-AX.
5. Open new project to check if the connection with the PLC has been established (path: **Device** > **Communication Setting** > **Scan Network**).

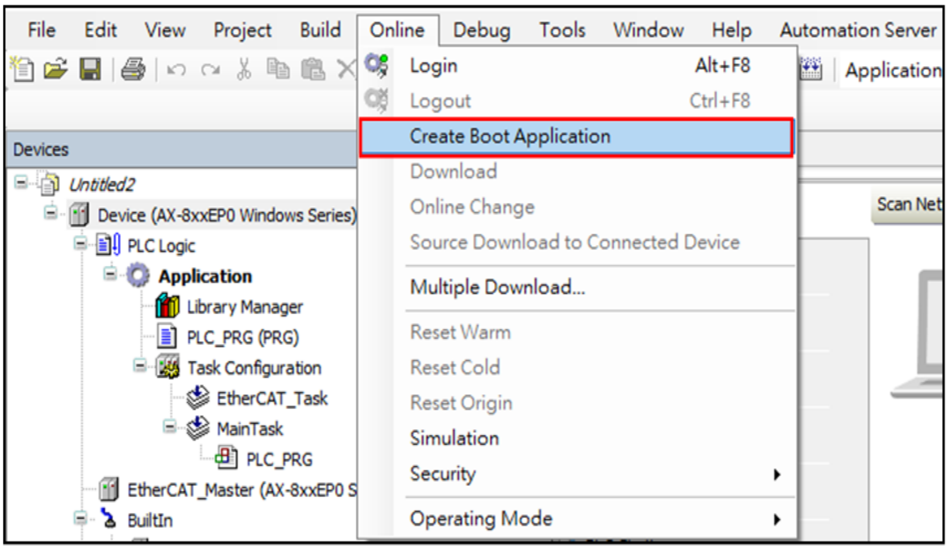


6. Check the AX-8 Series PLC folder system through the path: **Device** > **Files**. Click the button marked below to retrieve the PlcLogic folder.

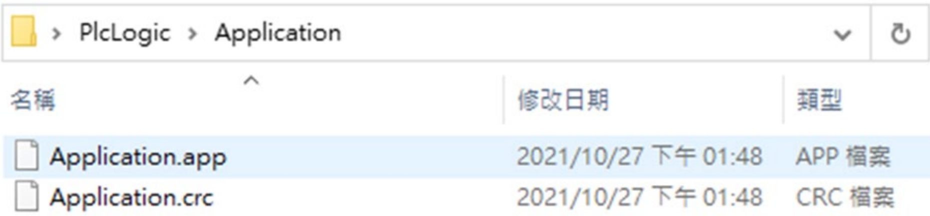


C.4.2 Create a Project from DIADesigner-AX

1. **Online** > **Create Boot Application**.



2. Create folder structure **PlcLogic\Application** and drag **Boot Application** into the **Application** folder.



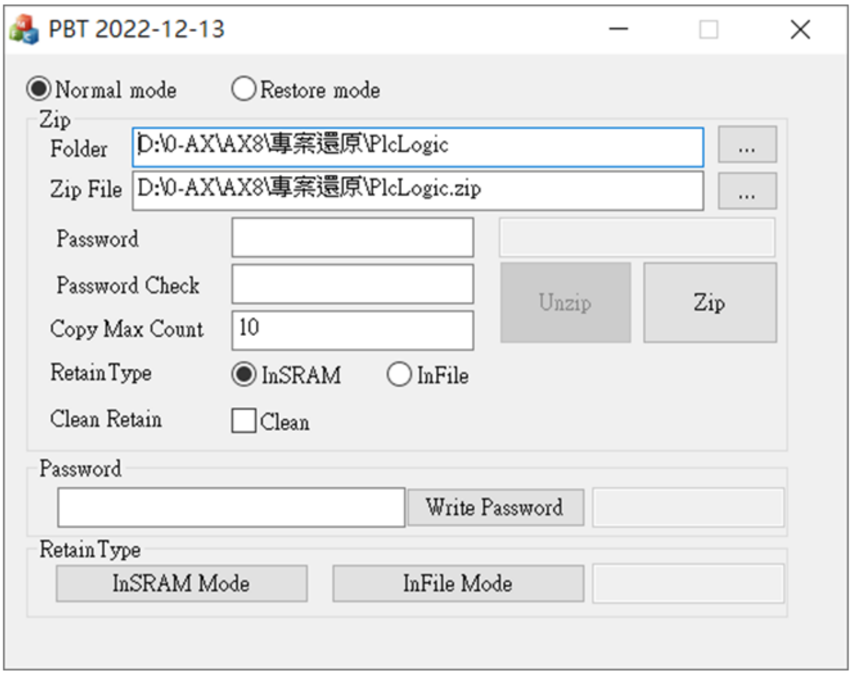
C.4.3 Create an USB Backup Project

Software: PBT

Operating system: Windows 7 (Service Pack 1 or higher) / 8 / 10 (32 / 64 Bit)

Download path: https://downloadcenter.deltaww.com/downloadCenterCounter.aspx?DID=41632&DocPath=1&hl=zh-TW

Use the zip tool, PBT.exe, to compress PlcLogic folder mentioned earlier into PlcLogic.zip.

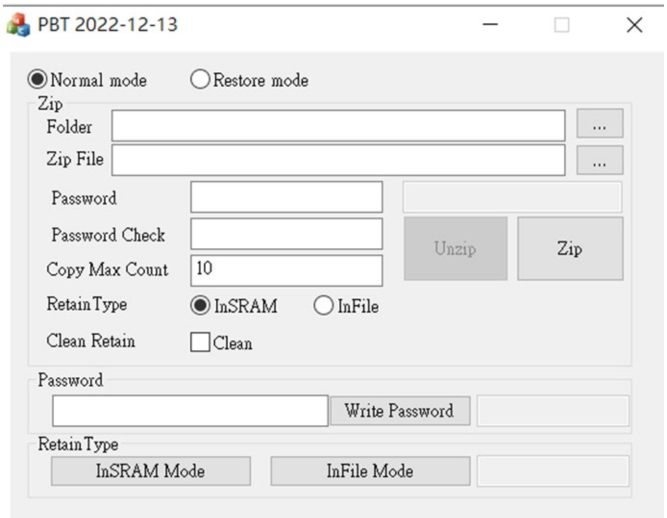


1. Folder: folder to restore
2. Zip File: location to generate the zip file
3. Password: password of the zip file (optional)
4. Password Check: password confirmation (only for Windows system)
5. Copy Max Count: maximum times of the project to be restored. When exceeding the limit, the restored file will be removed.
6. Clean Retain: Whether current MRAM needs to be removed before the project restore.

To write the password, you can also call DFB_SetRestorePwd in the CODESYS project to store the matching password in the PLC. Password length is limited to 20 characters.

C.4.4 Project Mode

1. Normal mode: restore PLC project



Folder: folder to restore

Zip File: location to generate the zip file

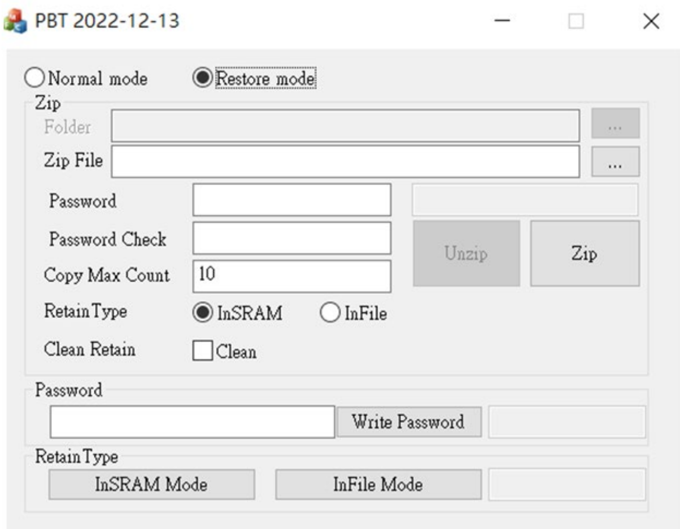
Password: password of the zip file (optional)

Password Check: password confirmation (for Windows system only)

Copy Max Count: maximum time of the project to be restored. When exceeding the limit, the restored file will be removed.

Clean Retain: whether the current MRAM needs to be removed before project restoration.

2. Restore mode: remove PLC project

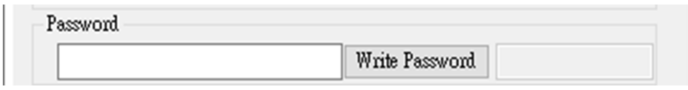


Zip File: location to generate the zip file

Password: password of the zip file (optional)

Password Check: password confirmation

3. **AX-8 project password settings**



Write password: write the password into the AX-8 Series PLC MRAM.

Users can write the password directly on the AX-8 Series PLC (for Windows only). There is also a corresponding function block (DFB_SetRetainType& DFB_SetRestorePwd) can be written into through PLC.

4. **Retain Type**

InSRAM: save in MRAM and the size is 96KB.

InFile: save in M.2 and the size is 10MB.

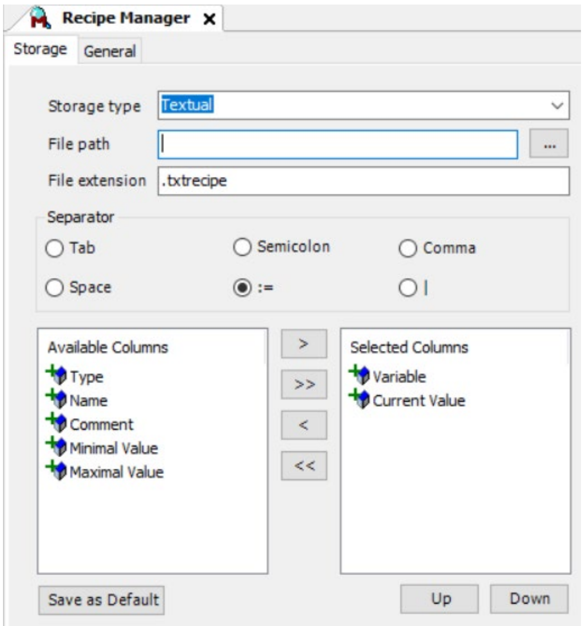
C.4.5 Project Replacement

1. Make sure to follow steps above to create an USB backup project.
2. In the power-off mode, insert the USB flash drive into the PLC and then turn the power on.
3. If the zip file is encrypted, the password for decompression should be written onto the PLC (refer to **section C.4.3**).
4. After the AX-8 Series PLC is powered on, replacement will be executed automatically with intermittent buzzing sounds to remind users.
5. Remove the USB flash drive after hearing buzzing sounds.
6. After the USB flash drive is removed, the PLC will reboot automatically and then replacement is completed.
7. Repeat step 2 to 6 if the project replacement fails.

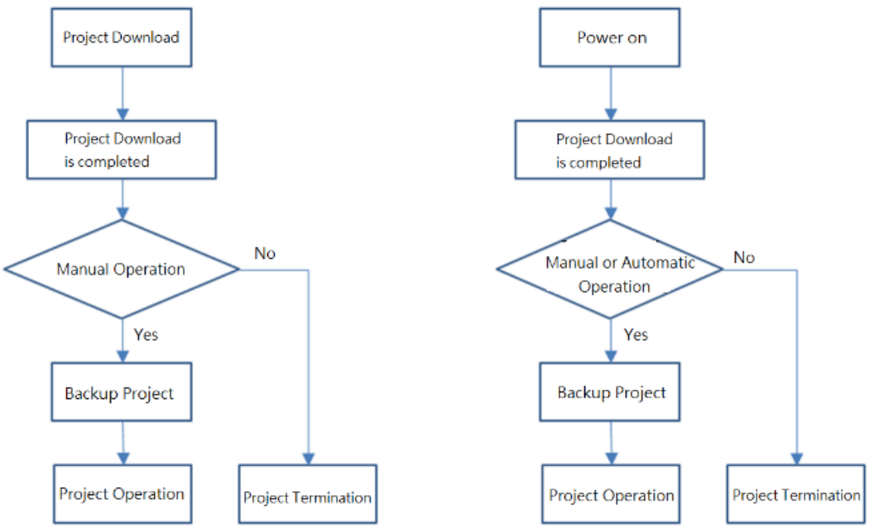
C.5 Windows System Operation

Refer to the following steps for operation of PLC with Windows operating system.

1. When users are creating or writing files, the file path should be set in external devices such as SD cards or USB flash drives. Storing in the device itself does not guarantee file integrity and accuracy.
2. As for recipe management, complete file path and exact file extension of the external storage devices (SD card, USB flash drive) are required for saving the recipe file. Storing in the device itself does not guarantee file integrity and accuracy.



3. If the project downloaded from DIADesigner-AX software has been executed before, the project is backed up automatically to the backup area to avoid unexpected file damage or loss. However, users should avoid power outages to ensure that the download procedure is completely correct.



4. Avoid creating or writing files manually when accessing the PLC with remote desktop.
5. Please follow the procedures stated in **section C.1.3** to turn on UWF as the PLC is in operation.

C.6 IP Address Settings

C.6.1 Modify through Project Download

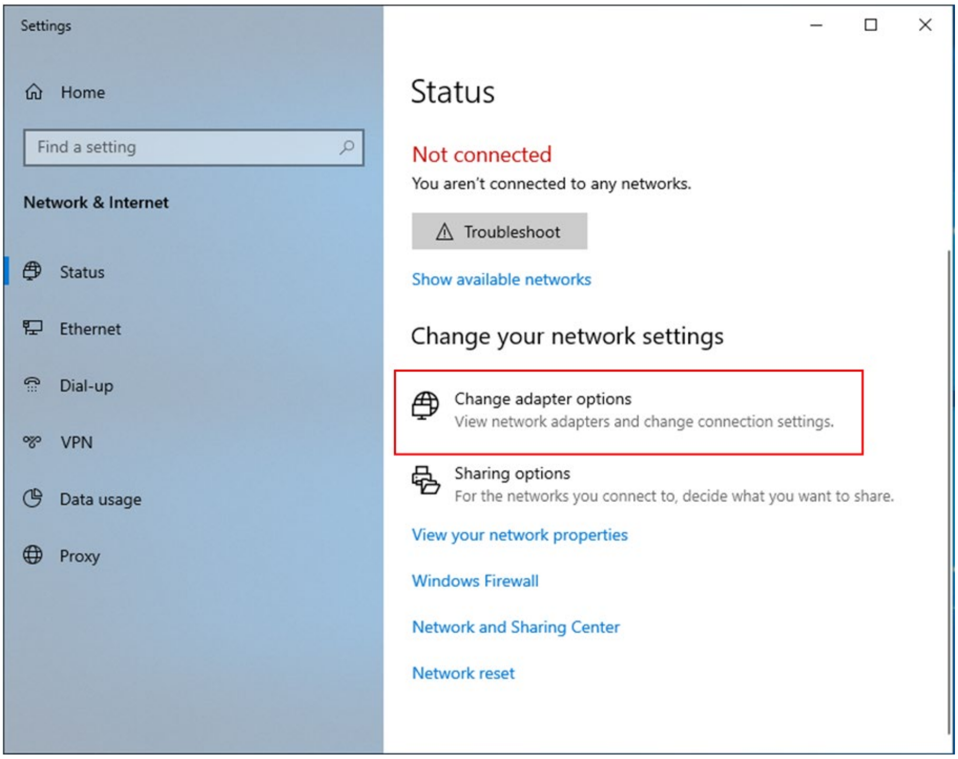
The IP address of AX-8 Series PLC can be modified by downloading the project. To Enter the IP settings page, go to **Device** > **Parameters**. IP-related settings can be adjusted in the **Value** column of the respective network interface. After the completion, download the project to the AX-8 Series PLC and the settings take effect immediately.

Parameter	Type	Value	Default Value	Unit	Description
Application Run Mode	BOOL	false	false		[True]:Enable Run Mode [False]:Disable Run Mode
Serial Communication Mode	Enumeration of BYTE	RS-485	RS-485		Choose Serial Communication Mode
GLAN1 NetworkInterface					
GLAN1 NetworkInterfaceMode	Enumeration of BOOL	static-IP	static-IP		Configure NetworkInterface Mode of the Controller on the EtherNet GLAN
GLAN1 IPAddress	ARRAY[0..3] OF BYTE	[192, 168, 0, 10]	[192, 168, 0, 10]		Configure IP Address of the Controller on the EtherNet GLAN1 (IP).
GLAN1 SubnetMask	ARRAY[0..3] OF BYTE	[255, 255, 255, 0]	[255, 255, 255, 0]		Configure IP Address of the Controller on the EtherNet GLAN1 (Mask).
GLAN2 NetworkInterface					
GLAN2 NetworkInterfaceMode	Enumeration of BOOL	static-IP	static-IP		Configure NetworkInterface Mode of the Controller on the EtherNet GLAN
GLAN2 IPAddress	ARRAY[0..3] OF BYTE	[192, 168, 1, 10]	[192, 168, 1, 10]		Configure IP Address of the Controller on the EtherNet GLAN2 (IP).
GLAN1 SubnetMask	ARRAY[0..3] OF BYTE	[255, 255, 255, 0]	[255, 255, 255, 0]		Configure IP Address of the Controller on the EtherNet GLAN2 (Mask).

C.6.2 Modify through Windows Settings

To enter Windows desktop, you can either go through remote desktop connection*1 or through HDMI cable connected to the PLC.

After Windows desktop is accessed, go to **Settings** > **Network & Internet** to change the IP address*2.



*1: The default IP address of GLAN1 is 192.168.0.10 while the default IP address of GLAN2 is 192.168.1.10.

*2: When UWF is on and the AX-8 Series PLC is rebooted, the IP address modified through Windows system will be replaced by the previous settings.

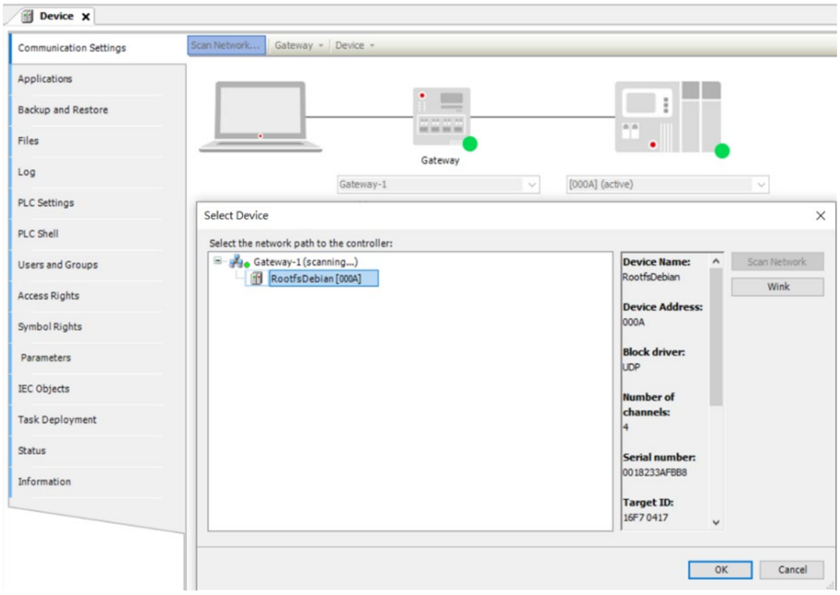
C.7 Reset IP Address

If the IP address[*1] is unable to be confirmed due to project lost or other factors, users can restore the system to factory settings and connect to the PLC with the default IP address or replace the project in the PLC with a backup one.

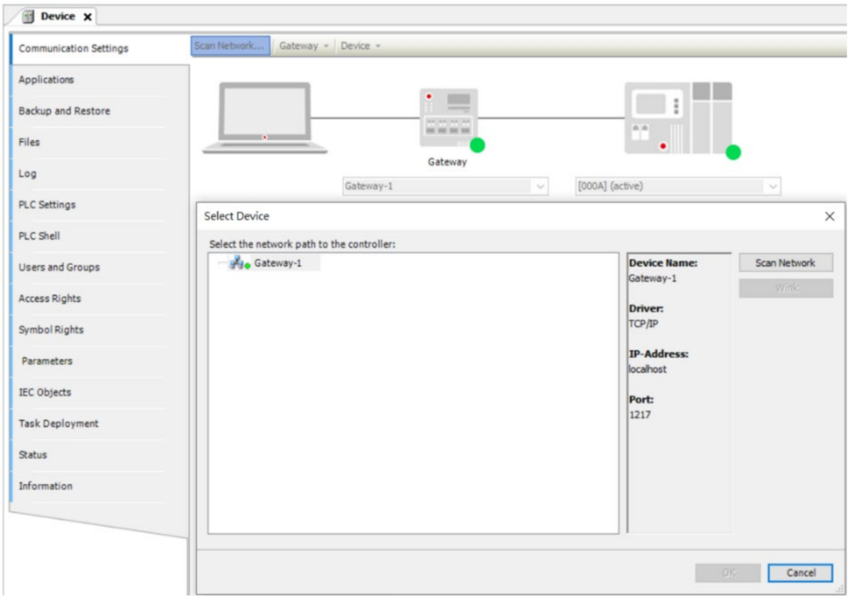
*1: The default IP addresses are 192.168.0.10 for GLAN 1 and 192.168.1.10 for GLAN 2.

C.7.1 Connection Denied to PLC

When the PC with DIADesigner-AX software and the AX-8 Series PLC are in the same network domain, users can detect the PLC through **Device** > **Communication Settings** > **ScanNetwork**; download the new project to change the IP address of the PLC.



On the contrary, the AX-8 Series PLC will not be detected by the abovementioned scan function if it is not located in the same domain where the PC exists.



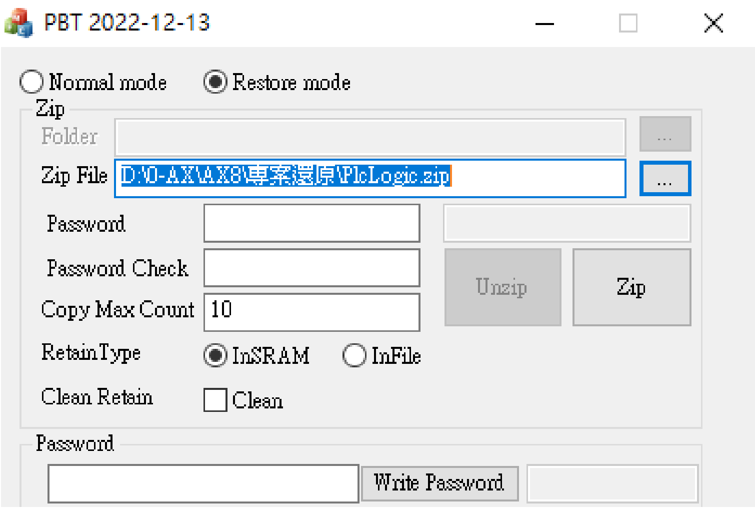
When it is unable to confirm the IP address, refer to **section C.7.2. Restore PLC**; the IP address will be restored to default after restoration. Or refer to **C.8 Replace Original Project with Backup Project;** the IP address will be the backup project IP address after replacement.

C.7.2 Restore PLC

Refer to the following steps to restore the device:

1. Use the zip tool PBT.exe, select Restore mode to generate a zip file PlcLogic.zip for restoration, and store this file in the root directory of the USB flash drive.
2. Insert the USB flash drive into the AX-8 Series PLC and turn the power on (it is acceptable if the PLC is already on).
3. Few seconds after, the PLC will be restored to the default settings and the original project will be removed, along with the PLC buzzing intermittently and each LED indicator keeping flashing.
4. Remove the USB flash drive and the PLC will reboot automatically. And after rebooting, you can use the default IP address for connection.

When the zip file is encrypted, execute DFC_SetRestorePwd on the PLC in advance to store the matching password in the PLC. Password length is limited to 20 characters.

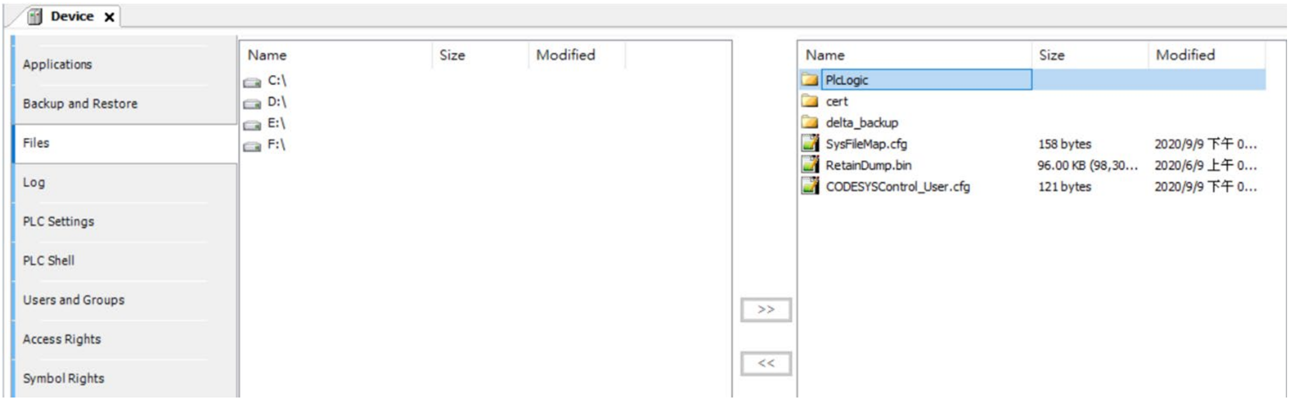


C.8 Replace Original Project with Backup Project

Users can save the project in the USB flash drive and retrieve it from the PLC. This section is about project backup and project restoration to the PLC.

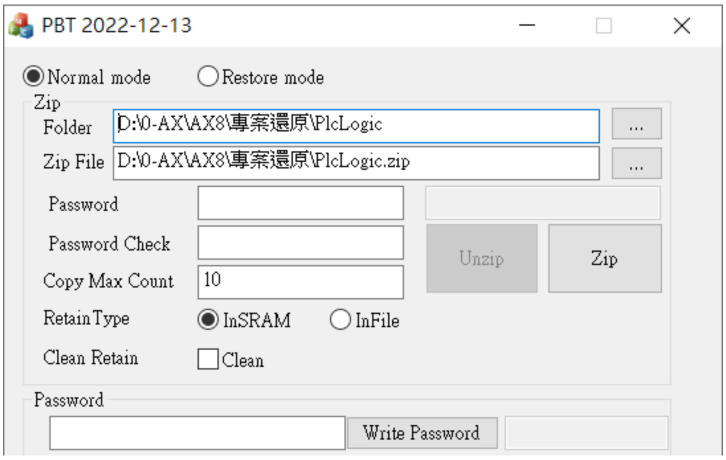
C.8.1 Retrieve a Project

Projects are stored in the PlcLogic folder which can be found on the page **Device** > **Files** in DIADesigner-AX software.



C.8.2 Create a Backup Project

Use the zip tool PBT.exe to compress the PlcLogic.zip folder generated in **section C.8.1** into PlcLogic.zip. Save this compressed folder to the directory and the USB flash drive with which you can switch among devices.



C.8.3 Project Replacement

Refer to the following steps to replace the original project:

1. Insert the USB flash drive into the AX-8 Series PLC and turn the power on (it is acceptable if the PLC is already on).
2. After few seconds, the original project will be replaced by the backup project with the buzzer sounding intermittently and the LED indicators keeping flashing.
3. Remove the USB flash drive then the PLC will reboot automatically.
4. After restarting the CPU, the IP address of the backup project can be connected.

When the zip file is encrypted, execute DFC_SetRestorePwd on the PLC in advance to enter the matching password. Password length is limited to 20 characters.

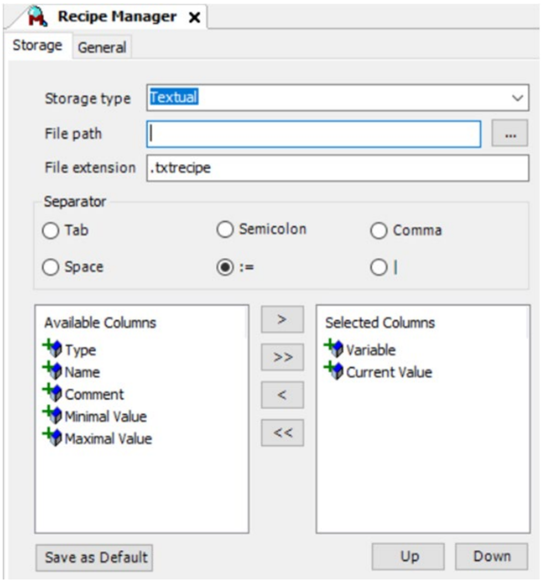
Failure to process the backup project in the USB will be accompanied with buzzing sounds and flashing Error LED indicators. Remove the USB flash drive and the PLC will reboot automatically. After rebooting, the PLC will return to the state before replacement.

C

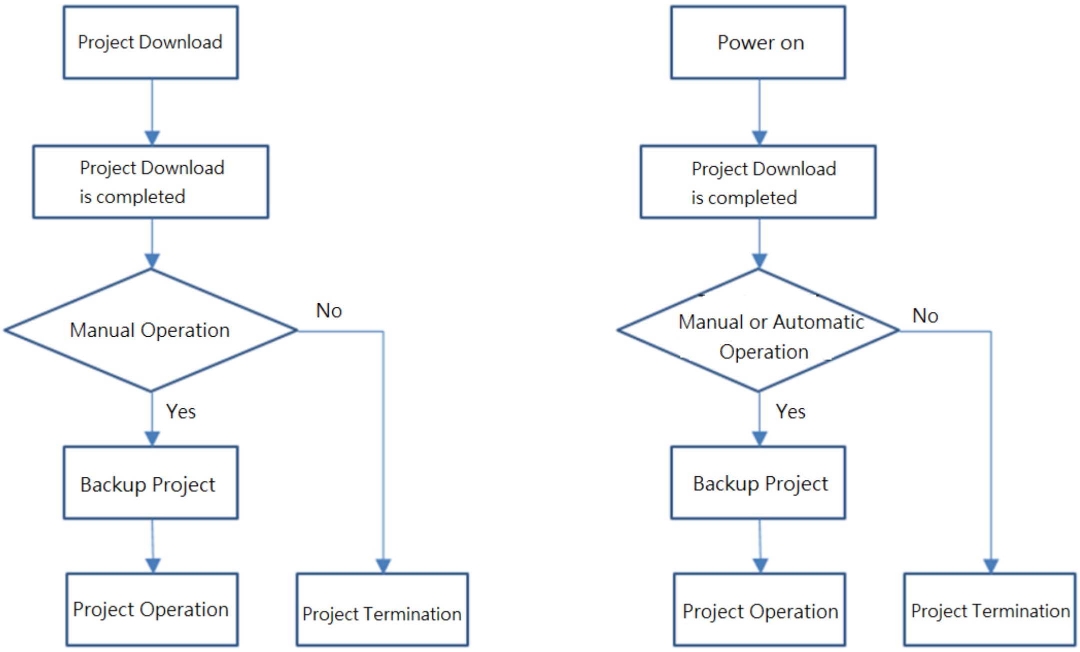
C.9 Linux System Operation

Refer to the following steps for operation of PLC with Linux operating system.

1. When users are generating or writing the file, the target path should be set in the external devices (SD Card, USB flash drive). Storing in the PLC does not guarantee file integrity and accuracy.
2. On the Recipe Manger page, users need to enter complete file path and exact file extension of the external devices (SD Card, USB flash drive) in the fields for storage. Storing in the PLC does not guarantee file integrity and accuracy.



3. If the project downloaded from DIADesigner-AX software has been executed before, the PLC will automatically back up the project to avoid unexpected file damage or loss. However, users should avoid power outages to ensure that the download procedure is completed correct.



D

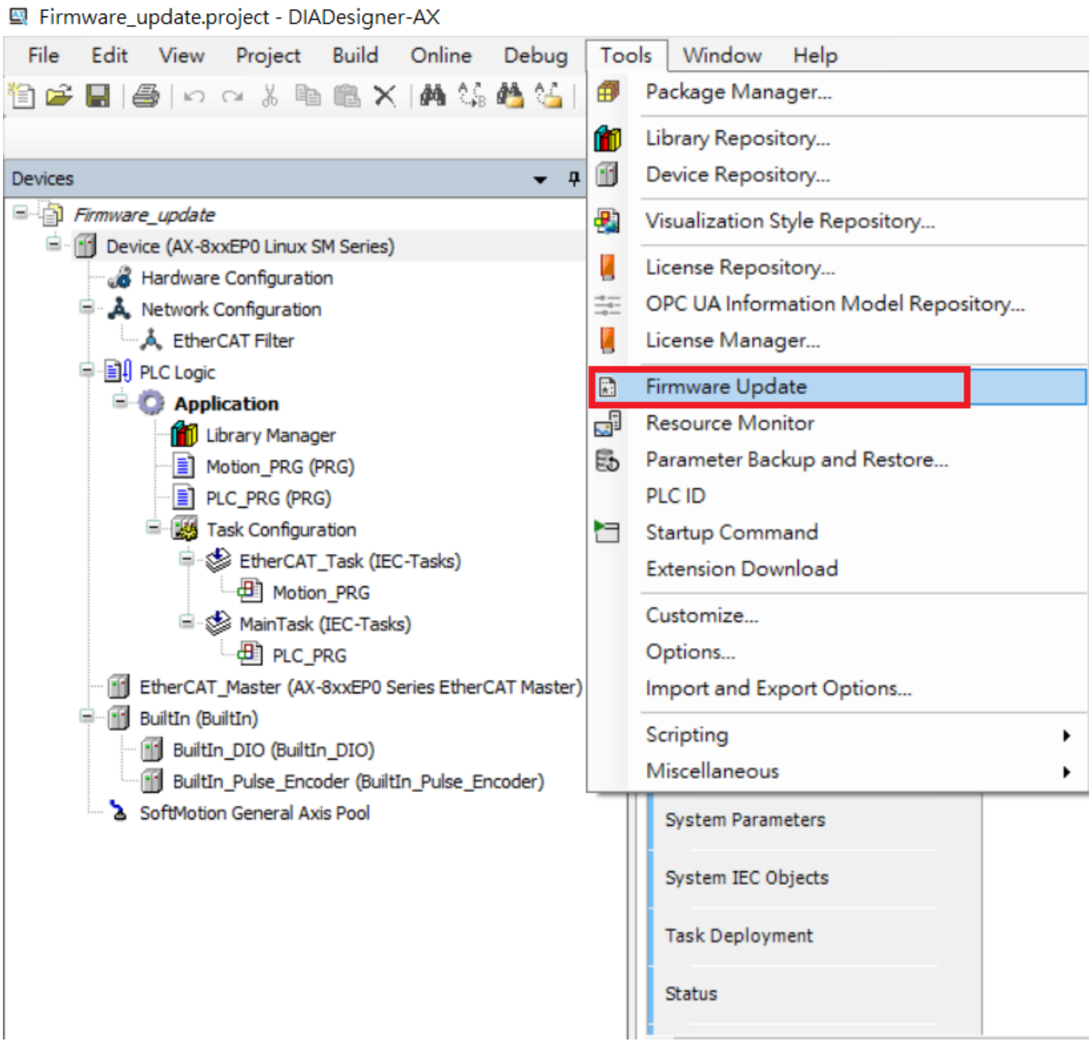
Appendix D Firmware Package Updater Operation

Table of Contents

D.1 Firmware Package Updater Operation

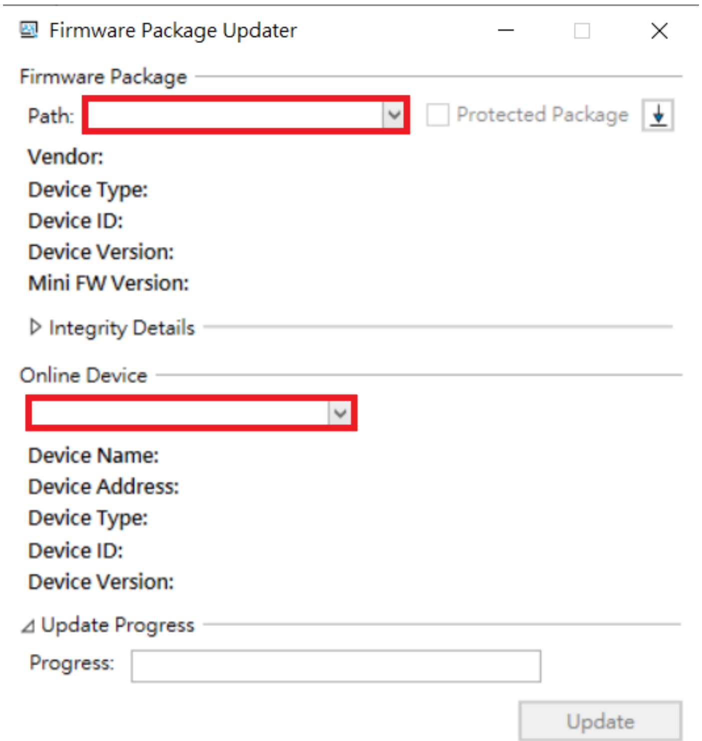
Open DIADesigner-AX 1.5. Click **Tools** on the toolbar and select **Firmware Update** as shown below.

- **Drop-down menu**



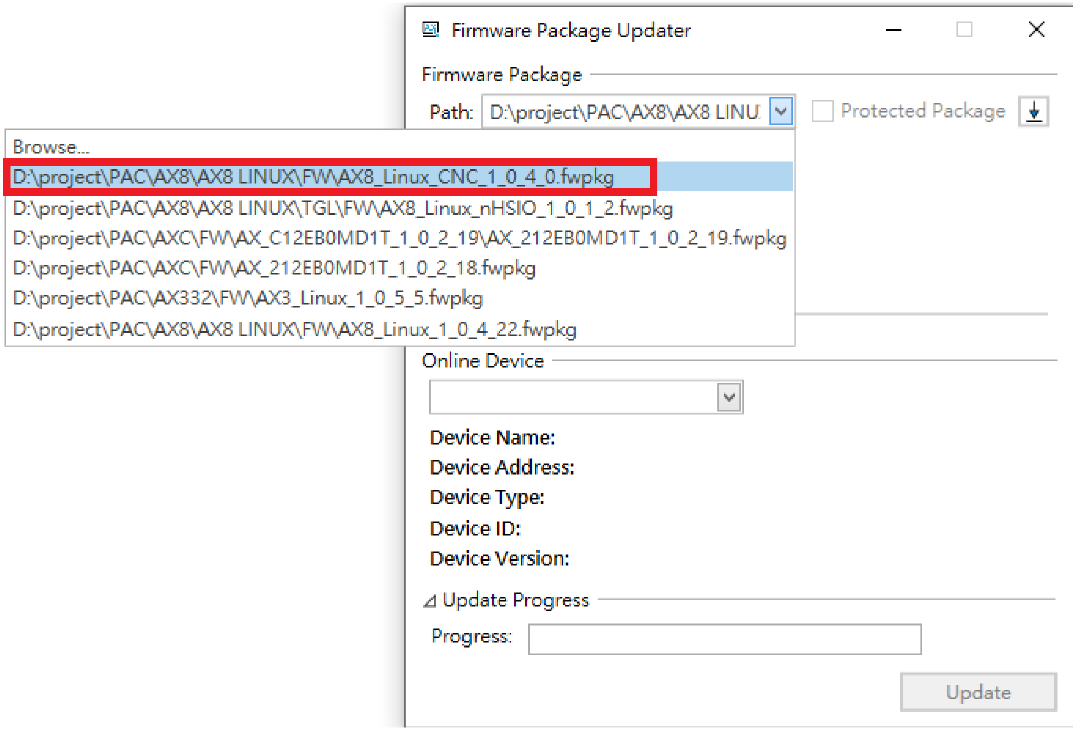
D.1.1 Firmware Package Updater

The setting window, **Firmware Package Updater**, allows you to set up **Path** (firmware file path) and **Online Device** (device update).



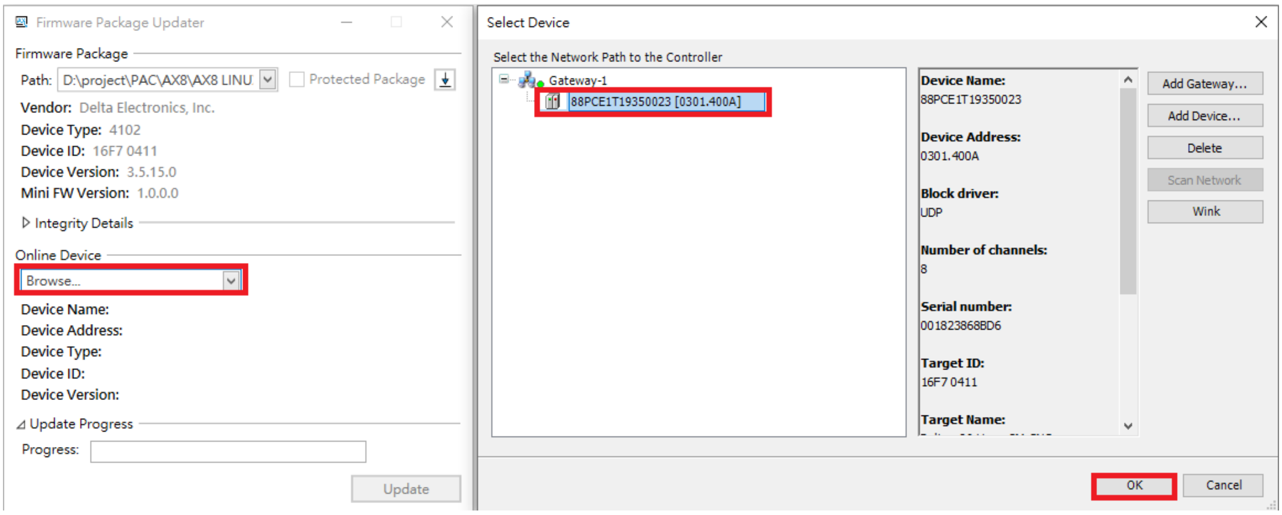
D.1.2 Firmware Path

You can set up the path for storage of firmware files under **Firmware Path**.



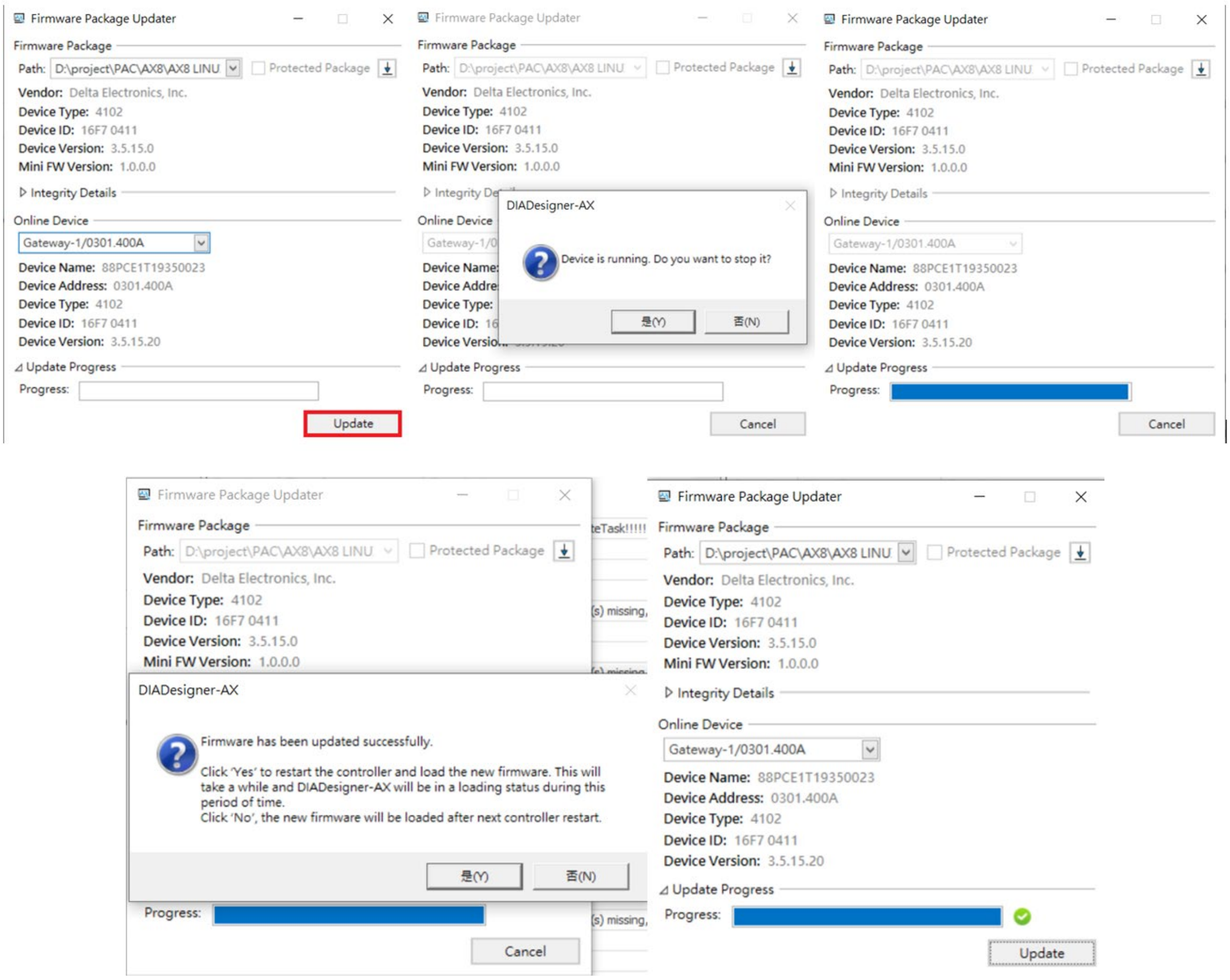
D.1.3 Online Device

Select the device for update under **Online Device.**



D.1.4 Update

Click **Update** to start firmware update.



Smarter. Greener. Together.

Industrial Automation Headquarters

Delta Electronics, Inc.
Taoyuan Technology Center
No.18, Xinglong Rd., Taoyuan District,
Taoyuan City 330477, Taiwan
TEL: +886-3-362-6301 / FAX: +886-3-371-6301

Asia

Delta Electronics (Shanghai) Co., Ltd.
No.182 Minyu Rd., Pudong Shanghai, P.R.C.
Post code : 201209
TEL: +86-21-6872-3988 / FAX: +86-21-6872-3996
Customer Service: 400-820-9595

Delta Electronics (Japan), Inc.
Industrial Automation Sales Department
2-1-14 Shibadaimon, Minato-ku
Tokyo, Japan 105-0012
TEL: +81-3-5733-1155 / FAX: +81-3-5733-1255

Delta Electronics (Korea), Inc.
1511, 219, Gasan Digital 1-Ro., Geumcheon-gu,
Seoul, 08501 South Korea
TEL: +82-2-515-5305 / FAX: +82-2-515-5302

Delta Energy Systems (Singapore) Pte Ltd.
4 Kaki Bukit Avenue 1, #05-04, Singapore 417939
TEL: +65-6747-5155 / FAX: +65-6744-9228

Delta Electronics (India) Pvt. Ltd.
Plot No.43, Sector 35, HSIIDC Gurgaon,
PIN 122001, Haryana, India
TEL: +91-124-4874900 / FAX: +91-124-4874945

Delta Electronics (Thailand) PCL.
909 Soi 9, Moo 4, Bangpoo Industrial Estate (E.P.Z),
Pattana 1 Rd., T.Phraksa, A.Muang,
Samutprakarn 10280, Thailand
TEL: +66-2709-2800 / FAX: +66-2709-2827

Delta Electronics (Australia) Pty Ltd.
Unit 2, Building A, 18-24 Ricketts Road,
Mount Waverley, Victoria 3149 Australia
Mail: IA.au@deltaww.com
TEL: +61-1300-335-823 / +61-3-9543-3720

Americas

Delta Electronics (Americas) Ltd.
5101 Davis Drive, Research Triangle Park, NC 27709, U.S.A.
TEL: +1-919-767-3813

Delta Electronics Brazil Ltd.
Estrada Velha Rio-São Paulo, 5300 Eugênio de
Melo - São José dos Campos CEP: 12247-004 - SP - Brazil
TEL: +55-12-3932-2300 / FAX: +55-12-3932-237

Delta Electronics International Mexico S.A. de C.V.
Gustavo Baz No. 309 Edificio E PB 103
Colonia La Loma, CP 54060
Tlalnepantla, Estado de México
TEL: +52-55-3603-9200

EMEA

Delta Electronics (Netherlands) B.V.
Sales: Sales.IA.EMEA@deltaww.com
Marketing: Marketing.IA.EMEA@deltaww.com
Technical Support: iatechnicalsupport@deltaww.com
Customer Support: Customer-Support@deltaww.com
Service: Service.IA.emea@deltaww.com
TEL: +31(0)40 800 3900

Delta Electronics (Netherlands) B.V.
Automotive Campus 260, 5708 JZ Helmond, The Netherlands
Mail: Sales.IA.Benelux@deltaww.com
TEL: +31(0)40 800 3900

Delta Electronics (Netherlands) B.V.
Coesterweg 45, D-59494 Soest, Germany
Mail: Sales.IA.DACH@deltaww.com
TEL: +49 2921 987 238

Delta Electronics (France) S.A.
ZI du bois Challand 2, 15 rue des Pyrénées,
Lisses, 91090 Evry Cedex, France
Mail: Sales.IA.FR@deltaww.com
TEL: +33(0)1 69 77 82 60

Delta Electronics Solutions (Spain) S.L.U
Ctra. De Villaverde a Vallecas, 265 1º Dcha Ed.
Hormigueras – P.I. de Vallecas 28031 Madrid
TEL: +34(0)91 223 74 20

Carrer Llacuna 166, 08018 Barcelona, Spain
Mail: Sales.IA.Iberia@deltaww.com

Delta Electronics (Italy) S.r.l.
Via Meda 2–22060 Novedrate(CO)
Piazza Grazioli 18 00186 Roma Italy
Mail: Sales.IA.Italy@deltaww.com
TEL: +39 039 8900365

Delta Greentech Elektronik San. Ltd. Sti. (Turkey)
Şerifali Mah. Hendem Cad. Kule Sok. No:16-A
34775 Ümraniye – İstanbul
Mail: Sales.IA.Turkey@deltaww.com
TEL: + 90 216 499 9910

Eltek Dubai (Eltek MEA DMCC)
OFFICE 2504, 25th Floor, Saba Tower 1,
Jumeirah Lakes Towers, Dubai, UAE
Mail: Sales.IA.MEA@deltaww.com
TEL: +971(0)4 2690148

AX-8-0389420-02
2024/12/30