10

Chapter 10 CANopen Function and Operation

Table of Contents

10.1 Introduction to CANopen

1. Because of its simple wiring, immediate and stable communication, strong debugging ability, and low cost, the CANopen network is widely used in fields such as industrial automation, the automotive industry, the medical equipment industry, and the building trades.
2. The AS300 Series PLC is compatible with CANopen when using the AS-FCOPM function card for the CAN port, and can work in master mode or slave mode. The CAN port conforms to the basic communication protocol CANopen DS301
3. This chapter explains the functions of CANopen. In master mode, CANopen can support up to 64 slaves.
4. CANopen Builder is the CANopen network configuration software for the AS Series. You set the CANopen station address and the communication rate with this software. ISPSoft is the programming software for the AS Series.
5. This chapter mainly focuses on the CANopen functions. Refer to Section 10.3 for more information.

10.1.1 CANopen Function Descriptions

The CAN port has the following functions when acting as a master.

1. It supports the standard CANopen protocol DS301 V4.02.
2. It supports the NMT (network management object) service.
   - It supports NMT state control.

     Use NMT state control to control the state of a slave in the CANopen network.
   - It supports NMT error control.

     Use NMT error control detect the disconnection of a slave. The NMT error control is classified into two types; Heartbeat and Node Guarding. The AS Series PLC does not support Node Guarding.
3. It supports the PDO (process data object) service.
   - Use PDO messaging to transmit immediate input and output data.
   - It supports up to 256 RxPDO and 1894 bytes.
   - It supports up to 256 TxPDO and 1894 bytes.
   - It supports synchronous and asynchronous modes for the PDO transmission type.
4. It supports the SDO (Service Data Object) service.
   - Use SDO to read, write, or configure the slave parameters.
   - It supports standard SDO transmission mode.
   - It supports automatic SDO functions. You can write up to 20 pieces of data to a slave.
   - It supports using the SDO service in a PLC ladder diagram to read the data from a slave or write the data to a slave.
5. It supports the reading emergencies from a slave service.
   - Use this service to read an error or an alarm from a slave.
   - You can store up to 5 emergencies in a slave.

- You can read emergencies from a slave through a PLC ladder diagram.

6. It supports the SYNC object (synchronous object) service.
7. Several devices can operate synchronously through the synchronous object service.
8. The supported CANopen communication rates are: 20K, 50K, 125K, 250K, 500K, and 1Mbps.

The supported mapping data types are:

Storage	Data type
8-bit	SINT USINT BYTE
16-bit	INT UINT WORD
32-bit	DINT UDINT REAL DWORD
64-bit	LINT ULINT LREAL LWORD

The CAN port has the following functions when acting as a slave.

- It supports the standard CANopen protocol DS301 V4.02.
- It supports the NMT (network management object) service.
  - It supports the NMT state control.

    The state of the AS series in the CANopen network is controlled by a master.
  - It supports the NMT error control.

    The AS Series supports Heartbeat but not Node Guarding.
- It supports the PDO (process data object) service.
  - The PDO message transmits the immediate input data and output data.
  - It supports up to 8 TxPDO and 8 RxPDO.
  - The PDO transmission type: synchronous mode and asynchronous mode
- It supports the emergency service.

  If an error or an alarm occurs in the AS series, the master is notified through the emergency service.

10.1.2 The Input/Output Mapping Areas

The following table lists the CANopen DS301 specifications for the AS series PLC.

Type	Item	Description
Master	Maximum slave nodes	Up to 64 nodes
	Maximum transfer size of a PDO (Read + Write)	Up to 2000 Bytes (including some system configurations)
Slave	Maximum transfer size of a PDO (Read + Write)	Up to 8 PDOs; each PDO with up to 8 bytes can be transferred at a time

The output mapping areas are D25000-D25999, and the input mapping areas are D24000-D24999, as the following table shows.

Device in the PLC	Mapping area	Mapping length
D25000–D25031	SDO request information, NMT service information, and Emergency request information	64 bytes
D24000–D24031	SDO reply information, and Emergency reply information	64 bytes
D25032–D25978	RxPDO mapping area	1894 bytes
D24032–D24978	TxPDO mapping area	1894 bytes

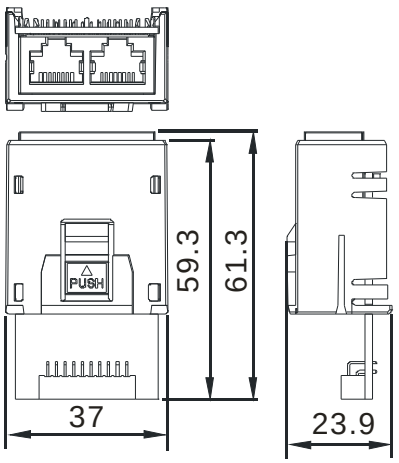
If an AS Series functions as a slave, the output mapping areas are D25032–25063, and the input mapping areas are D24032–24063 as the following table shows.

Device in the PLC	Mapping area	Mapping length
D6032–D6063	RxPDO mapping area	64 bytes
D6282–D6313	TxPDO mapping area	64 bytes

10.2 Installation and Network Topology

This section introduces the physical dimensions of AS-FCOPM function card, the HWCONFIG settings, the CAN interface, the CANopen network framework, and the maximum communication distance.

10.2.1 The Dimensions of AS-FCOPM



Unit: mm

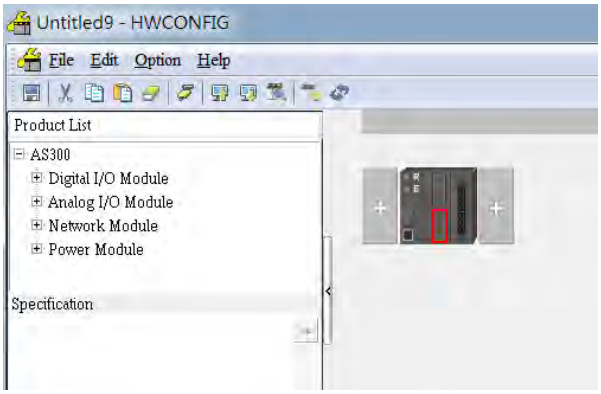
10.2.2 AS-FCOPM on AS300 PLC

Note: the AS-FCOPM card can only be installed in the Card 2 slot in the AS series

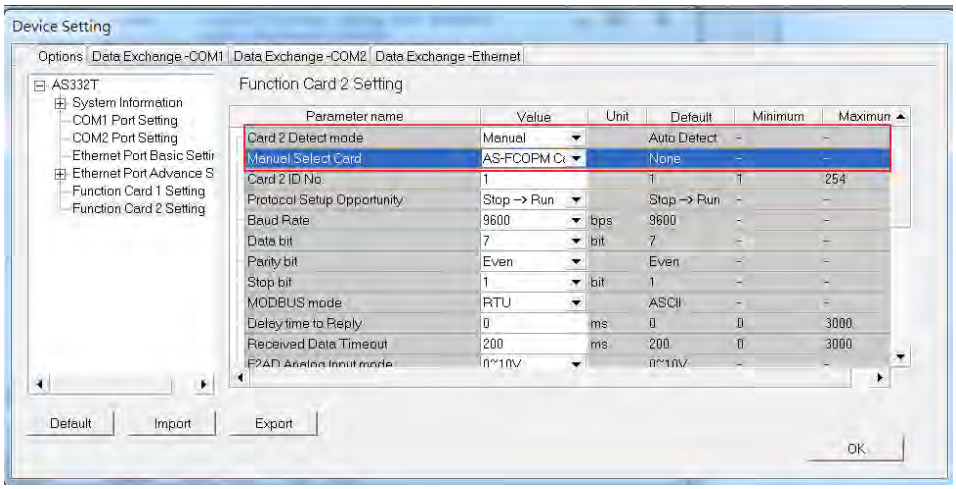
10.2.3 Configure the AS-FCOPM with HWCONFIG

Before using the AS-FCOPM card in an AS series PLC, use HWCONFIG in ISPSoft to configure the AS-FCOPM card.

1. You can only install the AS-FCOPM in the card 2 slot as marked in red below. Double click the function card 2 slot to go to the Device Setting dialog box.



2. Select **Manual** for **Card 2 Detect mode** and select **AS-FCOPM Card** for **Manual Select Card**.

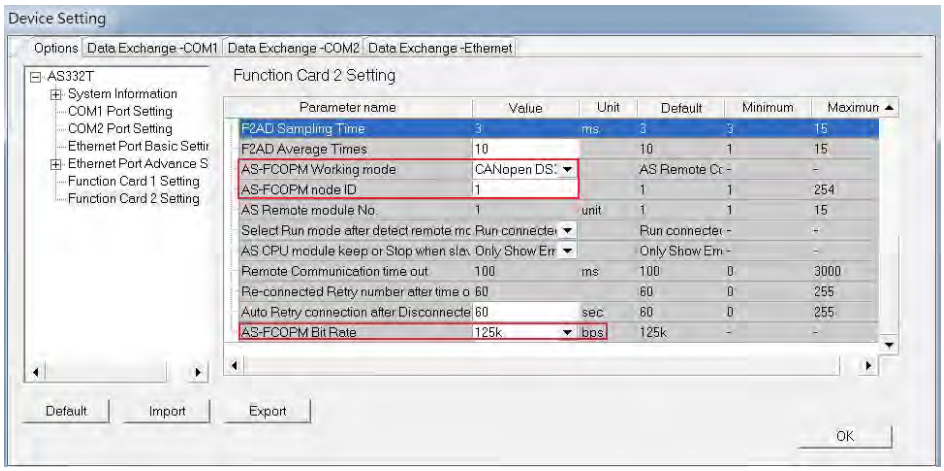


3. Select the working mode, node ID and the bit rate for AS-FCOPM.

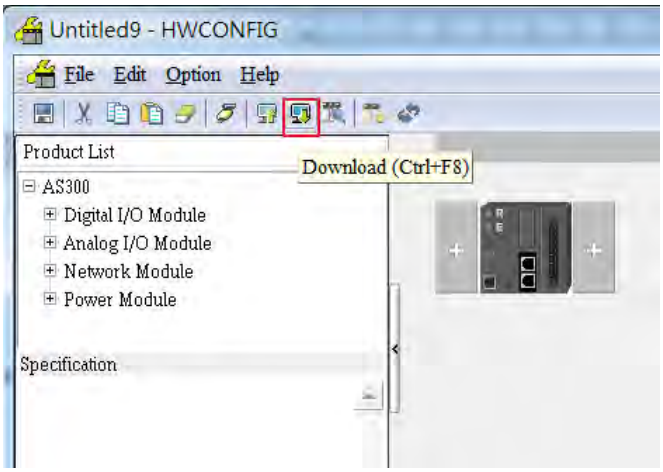
   * Working mode: CANopen DS301

   * Node ID: 1

   * Bit rate: 125k bps (the default, or you can select your own bit rate)



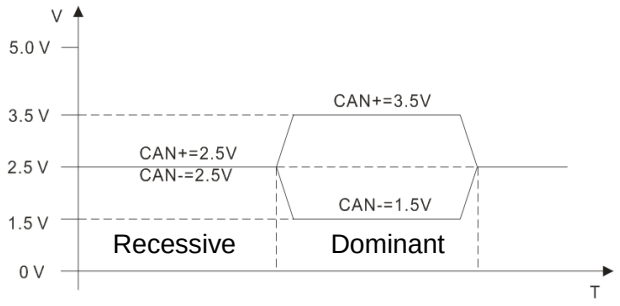
4. When finished, click the **Download** button on the toolbar to download the settings to the PLC.



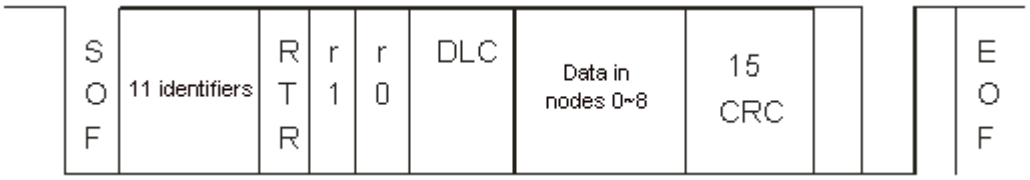
10.2.4 The CAN Interface and Network Topology

10.2.4.1 Definitions of the CAN Signal and Data Types

The CAN signal is a differential signal. The voltage of the signal is the voltage difference between CAN+ and CAN-. The CAN+ and CAN- voltages take SG as a reference point. The CAN network can be in one of two states. One state is a dominant level, and is indicated by the logical “0”. The other state is a recessive level, and is indicated by the logical “1”. The CAN signal level shows below.

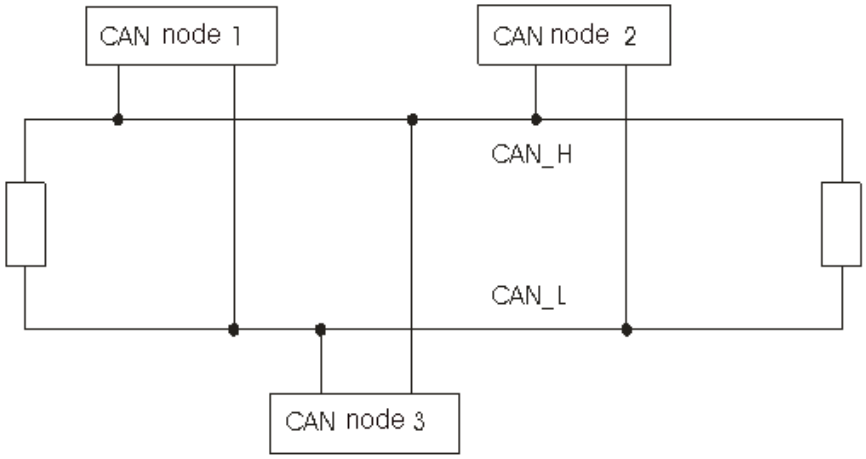


The following picture shows the data frame format. The CAN nodes transmit the CAN messages to the network from left to right.

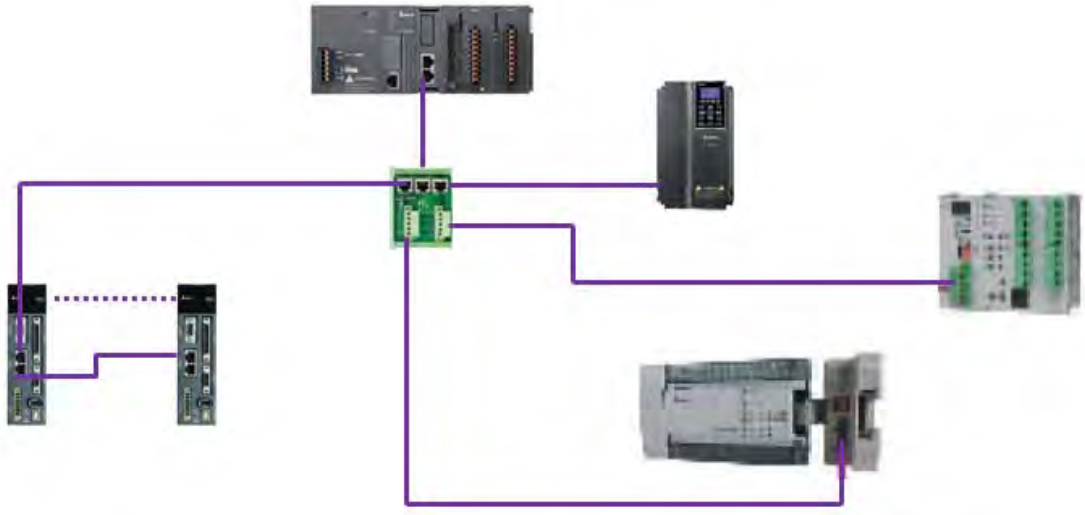


10.2.4.2 The CAN Network Endpoint and the Topology Structure

In order to make the CAN communication more stable, the two endpoints of the CAN network are connected to 120 ohm terminal resistors. The topology structure of the CAN network appears below.



10.2.4.3 The Topology Structure of the CANopen Network



1) Use standard Delta cables when wiring the CANopen network. These cables are the thick cable UC-DN01Z-01A, the thin cable UC-DN01Z-02A, and the thin cable UC-DN01Z-02A. Separate the communication cables from any power cables to avoid interference.

2) Connect the CAN+ (white) and CAN- (blue), which are at the endpoints of the network, to 120 ohm resistors. The AS-FCOPM card is equipped with a 120 ohm resistor switch; you can enable the resistor with the switch. Purchase the standard Delta terminal resistor for use with the other devices and the RJ45 connecter.

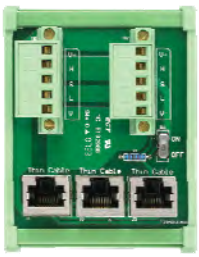
3) Note the limitation on the length of the CANopen network. The transmission distance of the CANopen network depends on the transmission rate of the CANopen network. The following table shows the relation between the transmission rate and the maximum communication distance.

Transmission rate (bit/seconds)	20K	50K	125K	250K	500K	1M
Maximum communication distance (meters)	2500	1000	500	250	100	25

4) The list below shows the Delta network products for the CANopen network.

Product	Model	Function
	AS332T-A AS332P-A AS324MT-A	The AS300 series PLC can function as the CANopen master or slave when you install an AS-FCOPM function card. The AS-FCOPM is equipped with a 120 ohm resistor controlled by a switch.
	DVP32ES200RC DVP32ES200TC	The DVP-ES2-C series PLC has a built-in CAN interface. It can function as the CANopen master or slave.
	DVPCOPM-SL	DVPCOPM-SL is a module connected to the left side of an S series PLC. It can function as the CANopen master or slave. The PLCs that you can connect to the DVPCOPM-SL are the DVP-28SV, DVP-28SV2, DVP-SX2, DVP-SA2, and DVP-EH2-L.
	IFD9503	The IFD9503 gateway converts CANopen to Modbus, and connects a device (with an RS-232 or RS-485 interface) that conforms to the standard Modbus protocol to a CANopen network. You can connect up to 15 devices.
	DVPCP02-H2	The CANopen slave module is connected to the right side of an EH2 series PLC. It can connect the EH2 series PLC to a CANopen network.

Product	Model	Function
	IFD6503	This analyzes CANopen network data, and has ports both ends for a CAN interface and a USB interface. Use it to monitor CAN network data, or allow CAN nodes to transmit the data. The product is used with the Netview Builder software.
	ASD-A2-xxxx-M servo driver	This servo driver has a built-in CANopen interface. It controls positioning, speed, and torque.
	VFD-C2000/CP2000/C200 series AC motor drives	This AC motor drive has a built-in CANopen function, and controls positioning, speed, and torque. For the C2000/CP2000 series AC motor drives, you must purchase a CMC-COP01 to provide the CAN interface. Only the C200 series AC motor drive has the built-in CANopen interface.
	VFD-EC series AC motor drive	The EC series AC motor drive has a built-in CANopen interface. It controls speed and torque.
	TAP-CN01	This CANopen network topology distribution box has a 120 ohm resistor enabled with a switch.
	TAP-CN02	This CANopen network topology distribution box has a 120 ohm resistor enabled with a switch.

Product	Model	Function
	TAP-CN03	This CANopen network topology distribution box has a 120 ohm resistor enabled with switch.
	UC-CMC003-01A UC-CMC005-01A UC-CMC010-01A UC-CMC015-01A UC-CMC020-01A UC-CMC030-01A UC-CMC050-01A UC-CMC100-01A UC-CMC200-01A	These CANopen sub cables have RJ45 connectors at both ends. UC-CMC003-01A: 0.3 meters UC-CMC005-01A: 0.5 meters UC-CMC010-01A: 1 meter UC-CMC015-01A: 1.5 meters UC-CMC020-01A: 2 meters UC-CMC030-01A: 3 meters UC-CMC050-01A: 5 meters UC-CMC100-01A: 10 meters UC-CMC200-01A: 20 meters
	UC-DN01Z-01A UC-DN01Z-02A	CANopen network cable UC-DN01Z-01A: CANopen main cable UC-DN01Z-02A: CANopen sub cable
	TAP-TR01	This 120 ohm resistor has an RJ45 connector.

10.3 The CANopen Protocol

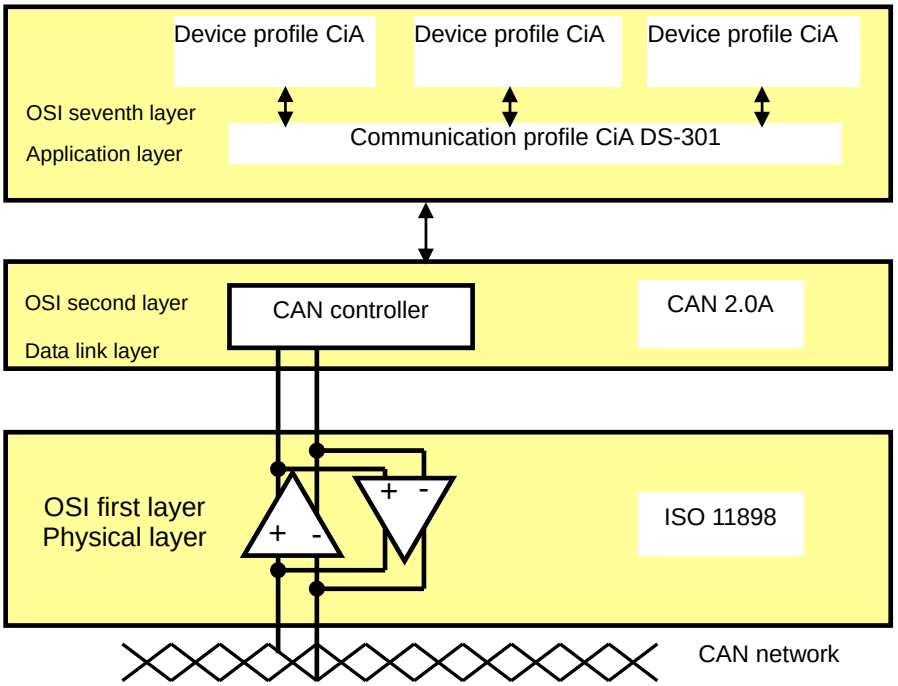
10.3.1 Introduction to the CANopen Protocol

The CAN (controller area network) fieldbus only defines the physical layer and the data link layer of a network. See the ISO11898 standard for information. The CAN fieldbus does not define the application layer. In practice, the hardware contains the physical layer and the data link layer. The CAN fieldbus itself is not complete, and needs a superior protocol to define the use of 11/29-bit identifier and 8-byte data.

The CANopen protocol is the superior protocol based on the CAN fieldbus. It is one of the protocols defined and maintained by CiA (CAN-in-Automation) and was developed on the basis of the CAL (CAN application layer) protocol, using a subset of the CAL communication and service protocols.

The CANopen protocol contains the application layer and the communication profile (CiA DS301). It also contains a framework for programmable devices (CiA 302), recommendations for cables and connectors (CiA 303-1), and SI units and prefix representations (CiA 303-2).

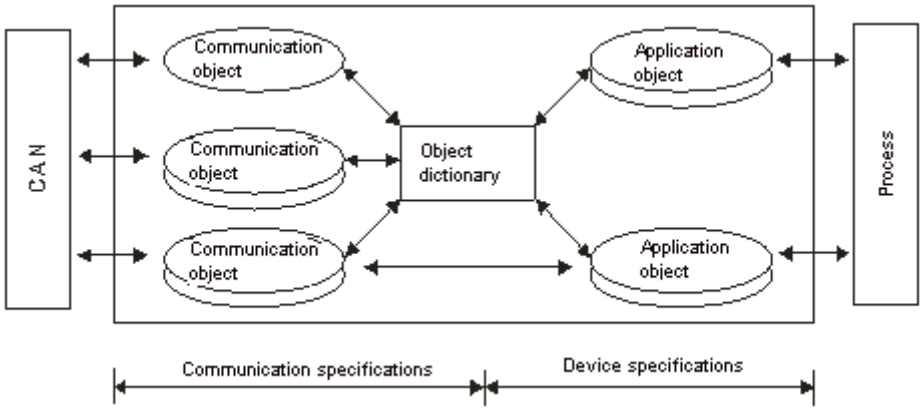
In the OSI model, the relation between the CAN standard and the CANopen protocol is described in the following diagram.



1. **The object dictionary**

CANopen uses an object-based method to define a standard device. Every device is represented by a set of objects and can be visited by the network. The diagram below illustrates the CANopen device model. The object dictionary is the interface between the communication program and the superior application program.

The core concept of CANopen is the device object dictionary (OD). It is an orderly set of objects. Every object has a 16-bit index for addressing and also defines an 8-bit subindex. Every node in the CANopen network has an object dictionary that includes the parameters that describe the device and the network behavior. The object dictionary of a node is also described in the electronic data sheet (EDS) for the device.



10.3.2 The CANopen Communication Object

The CANopen communication protocol contains the following communication objects.

1. PDO (process data object)

- The PDO provides the direct channel for the device application object, and transmits the real-time data. It has high priority. Every byte in the PDO CAN message data list transmits data, and the message usage rate is high.
- There are two kinds of uses for PDOs; data transmission and data reception. They are distinguished by Transmit-PDOs (TxPDOs) and Receive-PDOs (RxPDOs). Devices supporting TxPDOs are called PDO producers, and devices that receive PDOs are called PDO consumers.
- The PDO is described by the “producer/consumer mode”. The data transmits from one producer to one or many consumers. The data that can be transmitted is limited to between 1-byte and 8-byte data. After the producer transmits the data, the consumer does not need to reply to the data. Every node in the network detects the transmitted data and decides whether to process the received data .
- Every PDO is described by two objects in the object dictionary: the PDO communication parameters and the PDO mapping parameters

  PDO communication parameters: the COB-ID used by PDO, the transmission type, the prohibition time, and the counter cycle

  PDO mapping parameters: the object list in an object dictionary. These objects are mapped into the PDO, including the data length (in bits). To explain the contents of the PDO, the producer and the consumer both have to understand the mapping.

  The PDO transmission modes: synchronous and asynchronous

  Synchronous mode: synchronous periodic and synchronous non-periodic

  Asynchronous: The producer transmits the PDO when the data changes, or after a trigger.

  The following table lists supported transmission modes.

Type	PDO transmission				
	Periodic	Non-periodic	Synchronous	Asynchronous	RTR
0		X	X		
1 – 240	X		X		
254				X	
255				X	

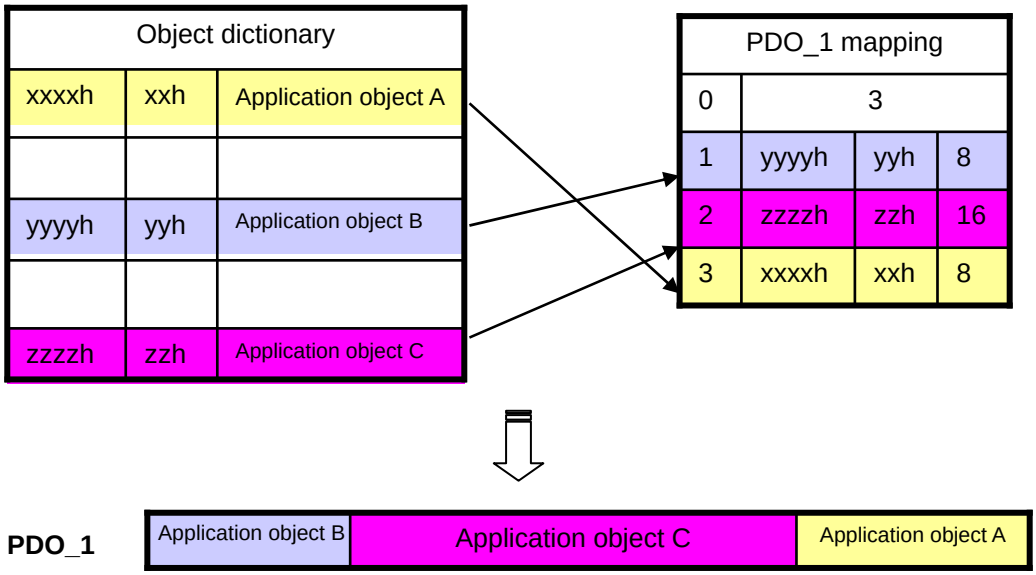
Mode 0: The PDO information is transmitted only when the PDO data changes and the synchronous signal is received.

Modes 1–240: One piece of PDO information is transmitted every 1–240 synchronous signals.

Mode 254: The trigger is defined the manufacturer. The definition in the PLC is the same as mode 255.

Mode 255: The PDO is transmitted when the data changes, or it is transmitted after a trigger.

All the data in the PDO has to be mapped from the object dictionary. The following diagram shows an example of PDO mapping.



The following table shows the data format for RxPDO and TxPDO.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Object identifier	Data							

2. SDO (service data object)

- The SDO builds the client/server relation between two CANopen devices. The client device can read the data from the object dictionary on the server device and write the data into the object dictionary on the server device. The SDO visit mode is “client/server” mode. The mode which is visited is the SDO server. Every CANopen device has at least one service data object that provides the visit channel for the object dictionary of the device. SDO can read all the objects in the object dictionary, and write all objects into the object dictionary.
- The SDO message contains the index and subindex information used to position the objects in the object dictionary, and the composite data structure can easily be passed by the SDO visit. After the SDO client sends the reading/writing request, the SDO server replies. The client and the server can stop SDO transmission. The requested message and the reply message are separated by different COB-IDs.
- The SDO can transmit the data in any length. If the data length is more than 4 bytes, the data must be transmitted by segment. The last segment of the data contains an end flag.
- The following table shows the structures of the SDO requested message and reply message.

  The format of the requested message:

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
600（hex）	Requested code	Object index		Object subindex	Requested data			
+Node-ID		LSB	MSB		bit7-0	bit15-8	bit23-16	bit31-24

The definition of the requested code in the requested message:

Request code (hex)	Description
23	Writing 4-byte data
2B	Writing 2-byte data
2F	Writing 1-byte data
40	Reading data
80	Stopping the current SDO function

The format of the reply message:

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
580（hex）+Node-ID	Reply code	Object index		Object subindex	Reply data			
		LSB	MSB		bit7-0	bit15-8	bit23-16	bit31-24

The definition of the reply code in the reply message:

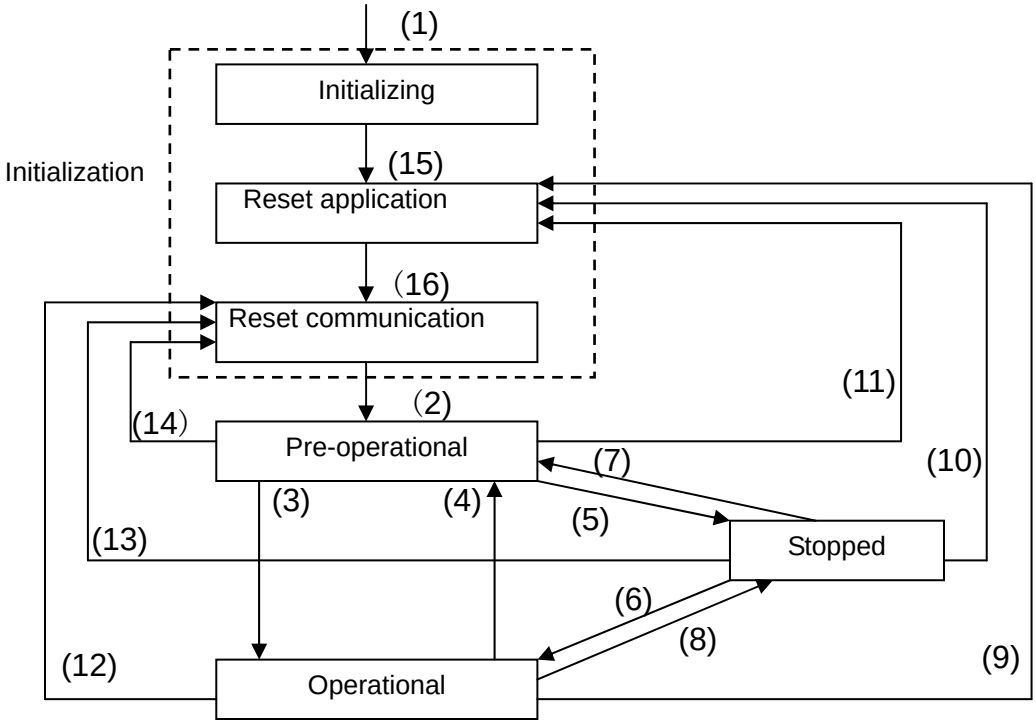
Reply code (hex)	Description
43	Reading 4-byte data
4B	Reading 2-byte data
4F	Reading 1-byte data
60	Writing 1/2/4-byte data
80	Stopping the current SDO function

3. NMT (network management object)

The CANopen network management conforms to the master/slave mode. Only one NMT master exists in the CANopen network, and all other nodes are considered to be slaves. NMT includes three services: module control, error control, and boot-up services.

- **Module control services**

  The master node in the CANopen network controls the slave by sending commands. The slave receives and executes the command, and does not need to reply. All CANopen nodes have internal NMT states. The slave node has four states: initialization, pre-operational, operational, and stop states. The following diagram illustrates the device states.



(1) After power is supplied, the device automatically enters the initialization state.

(2) After the initialization is complete, the device automatically enters the pre-operational state.

(3)(6) The remote node starts.

(4)(7) The device enters the pre-operational state.

(5)(8) The remote node stops.

(9)(10)(11) The application layer resets.

(12)(13)(14) The communication resets.

(15) After the initializing is complete, the device automatically enters the reset application state.

(16) After the reset application state is complete, the device automatically enters the reset communication state.

The following table shows the relation between the communication object and the state. You can only execute the communication object service in the correct state. For example, you can only execute SDO in the operational state and pre-operational states.

	Initialization	Pre-operational	Operational	Stopped
PDO			X	
SDO		X	X	
SYNC		X	X	
Time Stamp		X	X	
EMCY		X	X	
Boot-up	X			
NMT		X	X	X

The control message format for the node state:

COB-ID	Byte 0	Byte 1
0	Command specifier (CS)	Slave address (0: Broadcast)

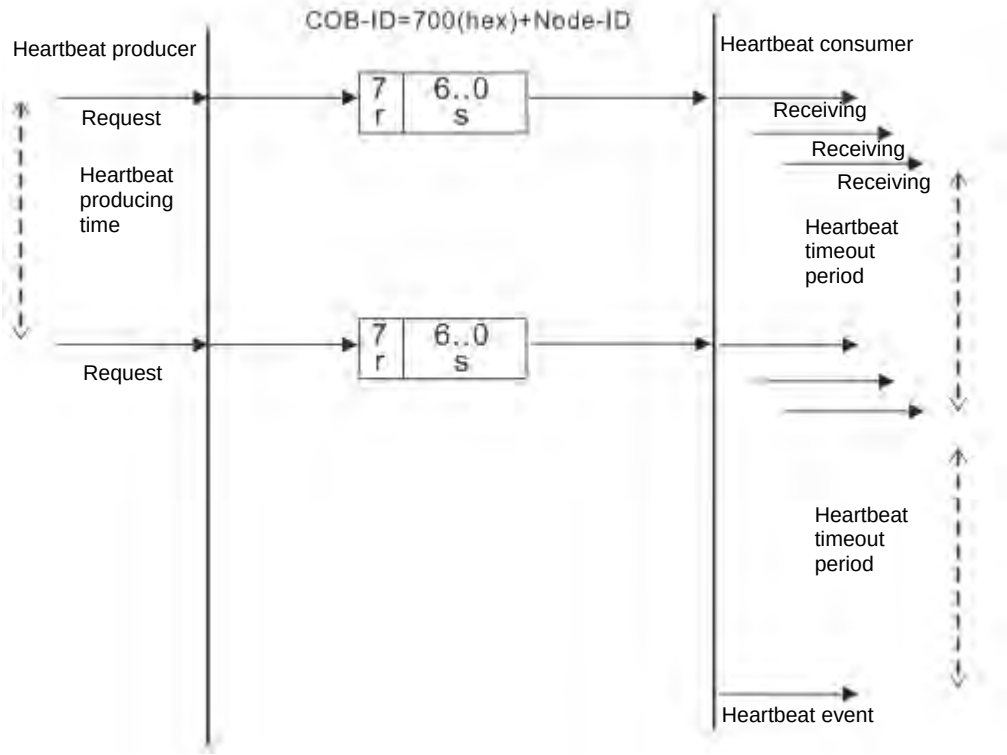
The command specifiers:

Command specifier (hex)	Function
01	Start the remote node
02	Stop the remote node
80	Enter the pre-operational state
81	Reset the application layer
82	Reset the communication

- **Error control services**

The error control service detects the disconnection of a network node. The error control services are classified into two types: Heartbeat and Node Guarding. The AS Series PLC only supports Heartbeat. For example, the master can detect the disconnection of the slave only after the slave enables the Heartbeat service.

The following illustrates the Heartbeat principle. The Heartbeat producer transmits the Heartbeat message according to the set Heartbeat producing time. One or many Heartbeat consumers detect the message transmitted by the Heartbeat producer. If the consumer does not receive the message transmitted by the producer within the timeout period, there is a problem in the CANopen communication or the producer is disconnected.



- **Boot-up services**

  After the slave completes the initialization and enters the pre-operational state, it transmits the Boot-up message.

1. Other predefined CANopen communication objects (SYNC and EMCY)

- **SYNC Object (Synchronous object)**

  The synchronous object is the message that the master node periodically broadcasts on the CANopen network. This object recognizes the network clock signal. Every device decides whether to use the event use synchronous communication with other network devices depending on its configuration. For example, when controlling a driving device, the devices do not act immediately after they receive the command sent by the master. They do act when they receive the synchronous message. This makes multiple devices act synchronously.

  The format of the SYNC message:

COB-ID
80 (hex)

- **Emergency object**

  The emergency object is used by a CANopen device to indicate an internal error. When an emergency error occurs in the device, the device sends the emergency message (including the emergency error code), and the device enters an error state. After the error is eliminated, the device sends another emergency message with emergency error code 0, and the device enters the normal state.

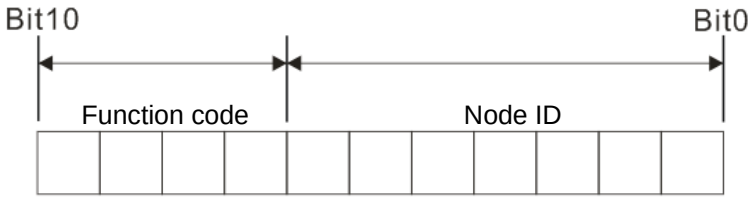
  The format of the emergency message:

<table>
<tr><th>COB-ID</th><th>Byte 0</th><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th><th>Byte 5</th><th>Byte 6</th><th>Byte 7</th></tr>
<tr><td rowspan="2">80 (hex)
+Node-ID</td><td colspan="2">Emergency error code</td><td rowspan="2">Error register</td><td colspan="5" rowspan="2">Factory-defined error code</td></tr>
<tr><td>LSB</td><td>MSB</td></tr>
</table>

Note: The value in the error register is mapped to index 1001 (hex) in the object dictionary. If the value is 0, no error occurred. If the value is 1, a normal error occurred. If the value is H'80, an internal error occurred in the device.

10.3.3 The Predefined Connection Set

In order to decrease the configuration workload of the network, CANopen defines a default identifier. In the predefined connection set, the structure of the 11-bit identifier is as follows.



The following tables list the objects that are supported and the COB-IDs that are assigned to the objects.

1. The broadcast object in the predefined connection setting

Object	Function code	COB-ID	Index of the communication parameter
NMT	0000	0	-
SYNC	0001	128 (80h)	1005h, 1006h, 1007h
Time stamp	0010	256 (100h)	1012h, 1013h

2. The corresponding object in the predefined connection set

Object	Function code	COB-ID	Index of the communication parameter
Emergency	0001	129 (81h)–255 (FFh)	1014h, 1015h
PDO1 (TX)	0011	385 (181h)–511 (1FFh)	1800h
PDO1 (RX)	0100	513 (201h)–639 (27Fh)	1400h
PDO2 (TX)	0101	641 (281h)–767 (2FFh)	1801h
PDO2 (RX)	0110	769 (301h)–895 (37Fh)	1401h
PDO3 (TX)	0111	879 (381h)–1023 (3FFh)	1802h
PDO3 (RX)	1000	1025 (401h)–1151 (47Fh)	1402h
PDO4 (TX)	1001	1153 (481h)–1279 (4FFh)	1803h
PDO4 (RX)	1010	1281 (501h)–1407 (57Fh)	1403h
SDO (TX)	1011	1409 (581h)–1535 (5FFh)	1200h
SDO (RX)	1100	1537 (601h)–1663 (67Fh)	1200h
NMT Error Control	1110	1793 (701h)–1919 (77Fh)	1016h, 1017h

10.4 Sending SDO, NMT and Reading Emergency Message through the Ladder Diagram

You can edit the request message mapping area to affect the transmission of SDO, NMT and Emergency messages. The following table shows the corresponding relations between the request message mapping area, response message mapping area, and PLC device.

PLC device	Mapping area	Mapping length
D25000-D25031	SDO request message, NMT service message and Emergency request message	64 bytes
D24000-D24031	SDO response message and Emergency response message	64 bytes

The CANopen master can only send one SDO, NMT, or Emergency request message to the same device at a time. Clear the request message mapping area to zero when sending SDO, NMT, or Emergency request message through the WPL program.

10.4.1 Data Structure of SDO Request Message

Sending SDO through the ladder diagram reads or writes the slave parameter.

1. The data format of the SDO request message:

PLC device	Request message		
		High byte	Low byte
D25000	Message Header	ReqID	Command (Fixed to 01)
D25001		Reserved	Size
D25002		Type	Node ID
D25003	Message Data	High byte of main index	Low byte of main index
D25004		Reserved	Sub-index
D25005		Datum 1	Datum 0
D25006		Datum 3	Datum 2
D25007 - D25031		Reserved	

- ReqID: the request ID. Whenever an SDO request message is sent out, the message is given a ReqID for CANopen master to identify. When reading/writing another SDO message, the original ID number must be changed. In other words, reading/writing SDO is triggered by changing of the value of “ReqID”. The ReqID range is between 00–FF (Hex).
- Size: the length of the message data. The counting starts from D6253 with a byte as the unit. When reading, it is fixed to four and when writing, it is four plus the byte number of data types of index and subindex and the maximum value is eight. But when writing, if the data type of the index and subindex is word, the data length is six. The data length is file if the data type is byte.
- Node ID: the node address for the target equipment on a CANopen network.
- Type: 01 indicates the read access; 02 indicates the write access.

The following table shows the data format of the SDO response message.

PLC device	Response message		
		High byte	Low byte
D24000	Message Header	ResID	Status code
D24001		Reserved	Size
D24002		Type	Node ID
D24003	Message Data	High byte of main index	Low byte of main index
D24004		Reserved	Sub-index
D24005		Datum 1	Datum 0
D24006		Datum 3	Datum 2
D24007~D24031		Reserved	

- Status code:

The following table lists the status code values in the response message.

Status code	Explanation
0	No data transmission request
1	SDO message transmission succeeds.
2	SDO message is being transmitted.
3	Error: SDO transmission time-out

Status code	Explanation
4	Error: Illegal command code
5	Error: the length of the transmitted data is illegal.
6	Error: the length of the response data is illegal.
7	Error: Equipment to be sent messages is busy.
8	Error: Illegal type
9	Error: Incorrect node address
0A	Error message (See the error code for SDO response message)
0B-FF	Reserved

- ResID: the same as the request ID in the request message.
- Size: the length of the message data, maximum of 20 bytes. The unit is bytes. When writing, the maximum is four; the data length is decided by the data type of index and subindex when reading.
- Node ID: the node address of the target equipment on CANopen network.
- Type: in the SDO response message, 43 (Hex) refers to reading four bytes of data; 4B (Hex) refers to reading two bytes of data; 4F (Hex) refers to reading one byte of data; 60 (Hex) refers to writing 1/2/4 byte(s) of data; 80 (Hex) refers to stopping SDO command.

Example 1: write 010203E8 (hex) to (Index_subindex) 212D_0 in slave of No. 3 through SDO; the data type of (Index_subindex) 212D_0 is double words (32 bits).

- Request data:

PLC device	Request message		
		High byte(Hex)	Low byte(Hex)
D25000	Message Header	ReqID=01	Command =01
D25001		Reserved =0	Size =8
D25002		Type =02	Node ID =03
D25003	Message data	Main index high byte =21	Main index low byte =2D
D25004		Reserved =0	Subindex =0
D25005		Datum 1=03	Datum 0=E8
D25006		Datum 3=01	Datum 2=02

- Response data:

PLC device	Response message		
		High byte(Hex)	Low byte(Hex)
D24000	Message Header	ResID =01	Command =01
D24001		Reserved =0	Size =4
D24002		Type =60	Node ID =03
D24003	Message data	Main index high byte =21	Main index low byte =2D
D24004		Reserved =0	Subindex =0
D24005		Datum 1=00	Datum 0=00
D24006		Datum 3=00	Datum 2=00

Example 2: read the value of (Index_subindex) 212D_0 in slave of No. 3 through SDO; the data type of (Index_subindex) 212D_0 is double words (32 bits).

- Request data:

PLC device	Request message		
		High byte(Hex)	Low byte(Hex)
D25000	Message Header	ReqID =01	Command =01
D25001		Reserved =0	Size =4
D25002		Type =01	Node ID =03
D25003	Message data	Main index high byte =21	Main index low byte =2D
D25004		Reserved =0	Subindex =0
D25005		Datum 1=0	Datum 0=0
D25006		Datum 3=0	Datum 2=0

10.4.2 Data Structure of NMT Message

Use the NMT service to manage the CANopen network such as start, operation, reset of nodes, etc.

The following table shows the data format of the NMT request message.

PLC device	Request message		
		High byte	Low byte
D25000	Message Header	ReqID	Command (Fixed to 01)
D25001		Reserved	Size (Fixed to 04)
D25002		Type (Fixed to 03)	Node ID
D25003	Message data	Reserved	NMT service code
D25004		Reserved	Node ID

- Command: Fixed to 01.
- ReqID: the request ID. Whenever an NMT request message is sent, the message is given a ReqID for the CANopen master to identify. Before another NMT request message is sent out, the original ID number must be changed. In other words, change the value of ReqID. The ReqID range is between 00–FF (Hex) to trigger sending the NMA request message.
- Node ID: the node address for the target equipment on the CANopen network (0: Broadcast).
- NMT service code:

NMT service code (Hex)	Function
01	Start remote node
02	Stop remote node
80	Enter the pre-operational state
81	Reset application
82	Reset communication

The following table shows the data format of the NMT Response message.

PLC device	Response message		
		High byte	Low byte
D24000	Message header	ResID	Status code
D24001		Reserved	Reserved
D24002		Reserved	Node ID

- When status code is 1, the NMT operation has succeeded. When status code is not equal to1, the NMT operation has failed and you should verify that the data in NMT request message is correct.
- Node ID: the node address for the target equipment on the CANopen network.

Example 1: Stop slave of No. 3 through NMT

- Request data:

PLC device	Request message		
		High byte(Hex)	Low byte(Hex)
D25000	Message header	ReqID =01	Command =01
D25001		Reserved =0	Size =04
D25002		Type =03	Node ID =03
D25003	Message data	Reserved	NMT service code =02
D25004		Reserved	Node ID =03

- Response data:

PLC device	Response message		
		High byte(Hex)	Low byte(Hex)
D24000	Message header	ResID=01	Status code =01
D24001		Reserved =0	Reserved =0
D24002		Reserved =0	Node ID =03

10.4.3 Data Structure of EMERGENCY Request Message

The Emergency request message communicates the slave error and alarm information.

The following table shows the data format of the Emergency request message.

PLC device	Request message		
		High byte	Low byte
D25000	Message header	ReqID	Command (Fixed to 1)
D25001		Reserved	Size (Fixed to 0)
D25002		Type (Fixed to 04)	Node ID
D25003~D25031	Message data	Reserved	

- Command: Fixed to 01.
- ReqID: the request ID. Whenever an Emergency message is sent, the message is given a ReqID for the CANopen master to identify. Before another Emergency request message is sent out, the original ID number must be changed. In other words, change the value of ReqID. The ReqID range is between 00–FF (Hex) to trigger the sending the Emergency request message.

- Node ID: the node address of the target equipment on CANopen network.

The following table shows data format of the Emergency response message.

PLC device	Response message		
		High byte(Hex)	Low byte(Hex)
D24000	Message header	ResID	Status code
D24001		Reserved	Size Fixed to 2A
D24002		Type (Fixed to 04)	Node ID
D24003	Message data	Total number of data	Number of data stored
D24004		Datum 1	Datum 0
D24005		Datum 3	Datum 2
D24006		Datum 5	Datum 4
D24007		Datum 7	Datum 6
D24008-D24011		Emergency2	
D24012-D24015		Emergency3	
D24016-D24019		Emergency4	
D24020-D24023		Emergency5	
D24024-D24031		Reserved	

- Command: Fixed to 01(Hex).
- When status code is 1, reading the Emergency message has succeeded. When status code is not equal to1, reading the Emergency message has failed and you should verify that the data in the Emergency message is correct.
- Node ID: the node address for the target equipment on the CANopen network.
- Total number of data: the total number of Emergency messages CANopen master receives from the slave.
- Number of data stored: the latest number of Emergency messages CANopen master receives from the slave (5 messages at most).
- An Emergency 1 consists of the data in D6004-D6007 and every Emergency message consists of 8 bytes of data.

  The following table shows the data format of Emergency messages on the CAN bus. Datum 0–datum 7 in Emergency response message correspond to byte 0–byte 7 respectively.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
80（hex）+Node-ID	Emergency error code		Error storage register	Vendor custom error code				

Example 1: read the Emergency message from the slave No.2, and the Emergency messages the slave sends out successively.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
82（hex）	43	54	20	14	0	0	0	0

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
82（hex）	42	54	20	15	0	0	0	0

- Request data:

PLC device	Request message		
		High byte	Low byte
D25000	Message header	ReqID=01	Command =01
D25001		Reserved	Size =0
D25002		Type =04	Node ID =03

- Emergency response data

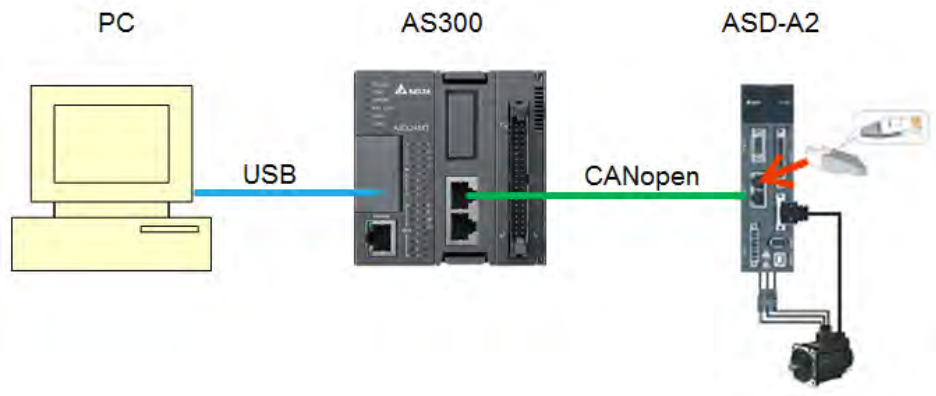
PLC device	Response message		
		High byte	Low byte
D24000	Message header	ResID=01	Status code =01
D24001		Reserved =0	Size =2A (Hex)
D24002		Type =04	Node ID =03
D24003	Message data	Total number of data =1	Number of data stored =1
D24004		Datum 1=54	Datum 0=42
D24005		Datum 3=20	Datum 2=14
D24006		Datum 5=0	Datum 4=0
D24007		Datum 7=0	Datum 6=0

10.4.4 Example of Sending SDO through the Ladder Diagram

1. Control Requirement:

Read the value of P0-09 from the servo through SDO.

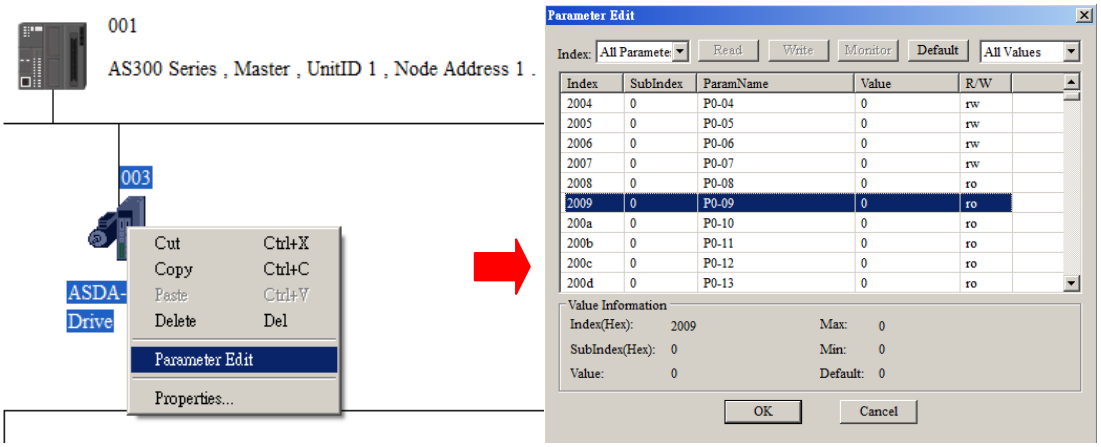
2. Hardware Connection:



3. The Corresponding Relation between Slave Parameter and Index/Subindex

The index_subindex corresponding to P0-09 in servo is 2009_0. In the CANopen Builder network configuration software, right click the servo icon, and then click **Parameter Edit**. In the **Parameter Edit** dialog box, you can see the index_subindex corresponding to the servo parameter.

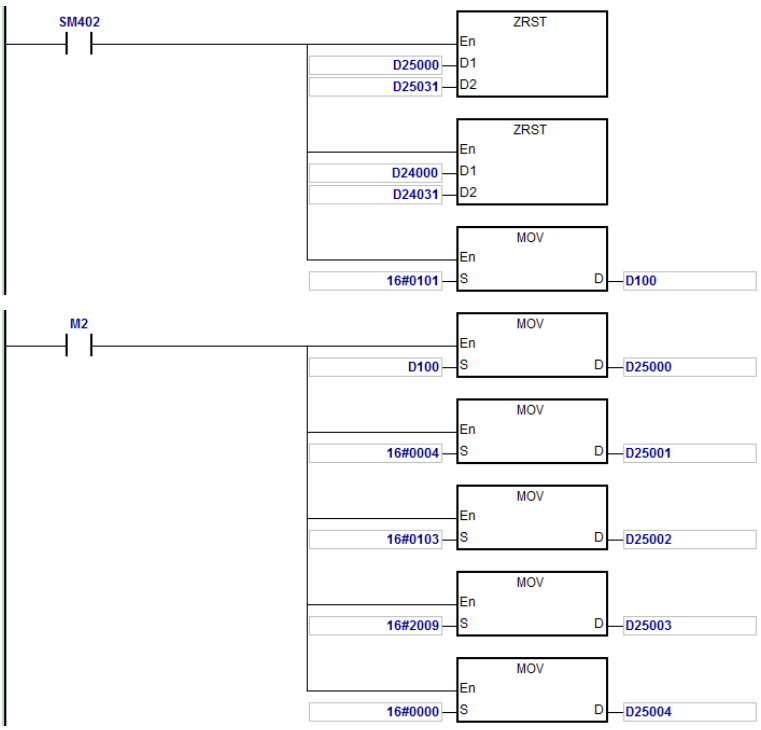
For more details on how to operate the network configuration interface, refer to Section 11.1.1 in the CANopen Builder software help file



4. The Structure of the Request Message Devices:

PLC device		Content (Hex)	Explanation	
			High byte(Hex)	Low byte(Hex)
SDO request message mapping area	D25000	0101	ReqID = 01	Command = 01
	D25001	0004	Reserved	Size = 04
	D25002	0102	Type = 01	Node ID = 03
	D25003	2009	Index high byte = 20	Index low byte = 09
	D25004	0000	Reserved	Subindex = 00

5. Editing the Ladder Diagram in ISPSoft



When M2=ON, after reading succeeds, the program stores the data from the target device in D24000–D24005. The value of D24005: 100（hex）is the value read from P0-09.

6. The Structure of the Response Message Devices:

PLC device		Content (Hex)	Explanation	
			High byte(Hex)	Low byte(Hex)
SDO response message mapping area	D24000	0101	ResID = 01	Status code = 01
	D24001	0006	Reserved	Size = 08
	D24002	4303	Type = 43	Node ID = 03
	D24003	2009	Main index high byte = 20	Index low byte = 09
	D24004	0004	Reserved	Subindex = 00
	D24005	0100	Datum 1= 01	Datum 0= 00

10.5 Troubleshooting

10.5.1 CANopen Network Node State Display

1. In the AS300 Series PLC, while you enable the CANopen function, it uses SR825–893 as the special registers as shown in the following table.

Special register	Function
SR825	Displays the state of AS300 series PLC.
SR830–SR893	Displays the state of 64 nodes in the network
SR826	Flag for the state of the slave 1–16
SR827	Flag for the state of the slave 17–32
SR828	Flag for the state of the slave 33–48
SR289	Flag for the state of the slave 49–64
SR821	Version of CANopen DS301
SR822	Displays the CANopen baud rate (unit: 1kpps)

2. As a master, the AS300 series PLC supports a maximum of 64 slaves ranging from node 1 to node 64. You can use SR826–829 to monitor the state of the nodes in the network. The 16 bits in SR826 correspond to 16 slaves and their corresponding relations are shown in the following table.

Bit	b7	b6	b5	b4	b3	b2	b1	b0
Node	Node 8	Node 7	Node 6	Node 5	Node 4	Node 3	Node 2	Node 1
Bit	**b15**	**b14**	**b13**	**b12**	**b11**	**b10**	**b9**	**b8**
Node	Node16	Node15	Node14	Node13	Node12	Node11	Node10	Node 9

When the node in the master node list is normal, the corresponding bit is OFF; when the node in the master node list is abnormal (for example, initializing fails or the slave is offline for some reason), the corresponding bit is ON.

3. The error code of every node is displayed through the corresponding special register (SR830–893) and the relations between special register and corresponding node (1–16) is shown in the following table.

Special register	SR830	SR831	SR832	SR833	SR834	SR835	SR836	SR837
Node	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8

Special register	SR830	SR831	SR832	SR833	SR834	SR835	SR836	SR837
Special register	SR838	SR839	SR840	SR841	SR842	SR843	SR844	SR845
Node	Node 9	Node10	Node11	Node12	Node13	Node14	Node15	Node16

4. Codes displayed in SR830–893 when the AS300 Series PLC is the master:

Code	Indication	How to correct
E0	AS300 Series PLC master module receives the emergency message sent from slave.	Read the relevant message with the PLC program
E1	PDO data length returned from the slave is not consistent with the length set in the node list.	Set the PDO data length of the slave and re-download.
E2	PDO of slave is not received.	Check and ensure the setting is correct.
E3	Downloading auto SDO fails.	Check and ensure auto SDO is correct.
E4	Configuration of PDO parameter fails.	Ensure that the PDO parameter setting is legal.
E5	Error in key parameter setting.	Ensure that the connected slave device is consistent with the configured slave in the software.
E6	The slave does not exist in the network	Ensure that the power supply of slave is normal and slave is correctly connected to the network.
E7	Slave error control is timed-out.	
E8	The node IDs of master and slave repeat.	Set the node ID of the master and slave again and ensure their node IDs are unique.

5. Codes displayed in SR825 when the AS300 Series PLC is the master:

Code	Indication	How to correct
F1	Slave has not been added to node list of CANopen Builder software	Add slave into the node list and then re-download the configuration data.
F2	The data are being downloaded to AS300 Series PLC	Wait to finish downloading the configuration data.
F3	AS300 Series PLC is in error status	Download parameter configuration again.
F4	Bus-OFF is detected.	Check that the CANopen bus cables are properly connected and ensure that all the node devices run at the same baud rate before you reboot.
F5	AS300 Series PLC setting error such as incorrect node address	The node address in the AS300 Series PLC should be between: 1–127.
F8	Internal error; the error is detected in the internal memory	If the same error occurs after cycling the power, replace it with a new AS300 PLC.
FB	The sending buffer in the AS300 Series PLC is full.	Check that the CANopen bus cable is properly connected and then reboot.
FC	The receiving buffer in the AS300 Series PLC is full.	Check that the CANopen bus cable is properly connected and then reboot.

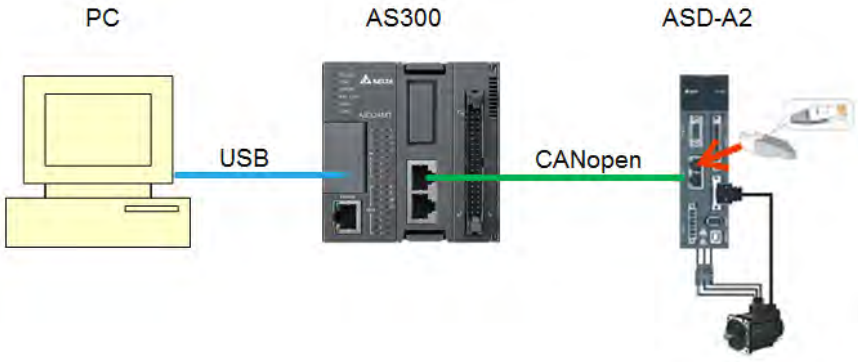
6. Codes displayed in SR825 when the AS300 Series PLC is the slave:

Code	Indication	How to correct
A0	AS300 series PLC is being initialized.	--
A1	AS300 series PLC is pre-operational.	Check that the CANopen bus cable is properly connected
A3	The data are being downloaded to AS300 series PLC	Wait to finish downloading the configured data.
B0	Heartbeat message time-out	Check that the CANopen bus cable is properly connected.
B1	PDO data length returned from the slave is not consistent with the length set in the node list.	Reset the PDO data length in the slave and download the new setting to the AS300 Series PLC.
F4	BUS-OFF state is detected.	Check that CANopen bus cables are properly connected and ensure that all the node devices run at the same baud rate before you reboot.
FB	The sending buffer in the AS300 Series PLC is full.	Check that the CANopen bus cable is properly connected and then reboot.
FC	The receiving buffer in the AS300 Series PLC is full.	Check that the CANopen bus cable is properly connected and then reboot.

10.6 Application Example

The AS300 Series PLC can control Delta A2 servo rotation, and monitor the actual rotation speed of the motor in real time. It does this by mapping the relevant servo drive parameters to the corresponding PDO, and reads or writes the relevant servo drive parameters through the CAN bus.

1. Connectting the Hardware



Note:

- Use a standard communication cable such as UC-DN01Z-01A / UC-DN01Z-02A / UC-CMC010-01A. and connect the terminal resistors (Delta standard terminal resistor TAP-TR01) to both ends of the network when you construct the network.
- M of ASD-A2-xxxx-M refers to the model code and currently only the M-model servo supports CANopen communication.

2. Setting Servo Parameters:

- Set servo parameters as shown in the following table.

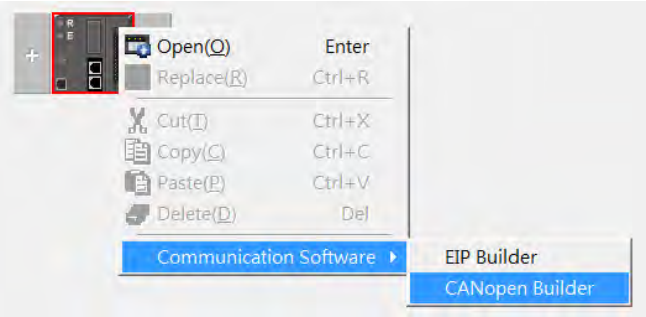
Parameter	Setting	Explanation
3-00	03	Node ID of the A2 servo is 2
3-01	400	CAN communication rate is 1Mbps.
1-01	04	Speed mode
0-17	07	Drive displays the motor rotation speed (r/min)
2-10	101	Set DI1 as the signal for Servo On
2-12	114	Set DI3 as the signal _SPD0 for speed selection
2-13	115	Set DI4 as the signal _SPD1 for speed selection

3. Setting the CANopen Baud Rate and Node ID of AS300 Series PLC

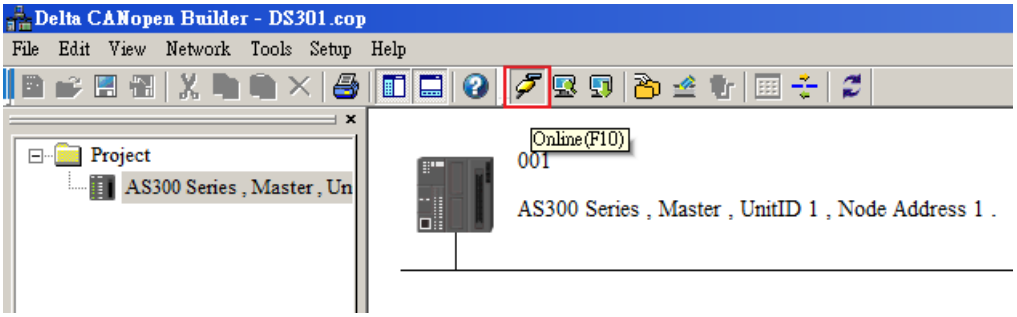
The AS300 Series PLC uses the default setting values: Node ID: 1 and baud rate: 1Mbps.

You set the CANopen Node ID and baud rate for the AS300 series PLC in the CANopen Builder software, as shown in the following steps.

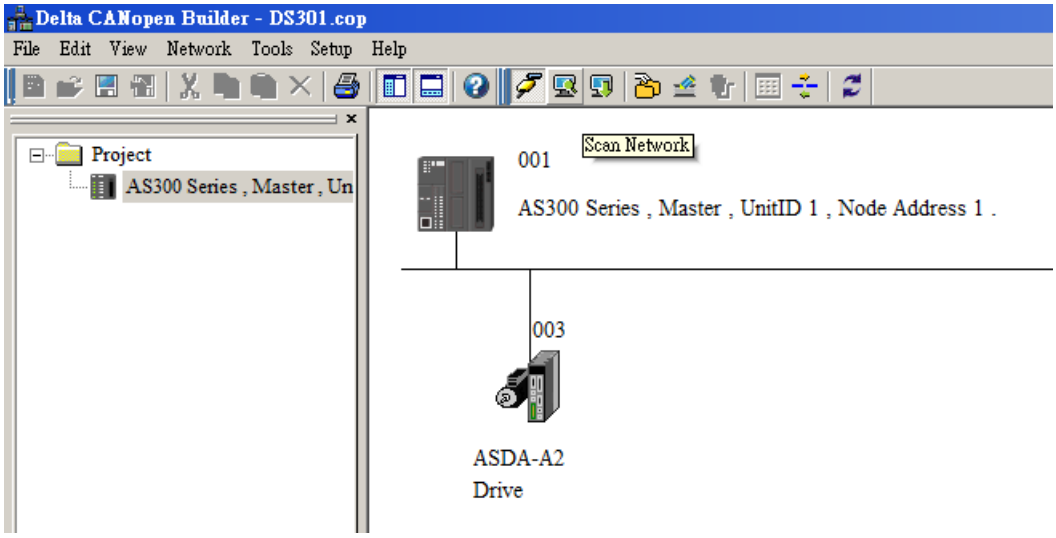
A. Right click the AS300 Series PLC icon in HWCONFIG, then click **Communication Software**, and then click **CANopen Builder**.



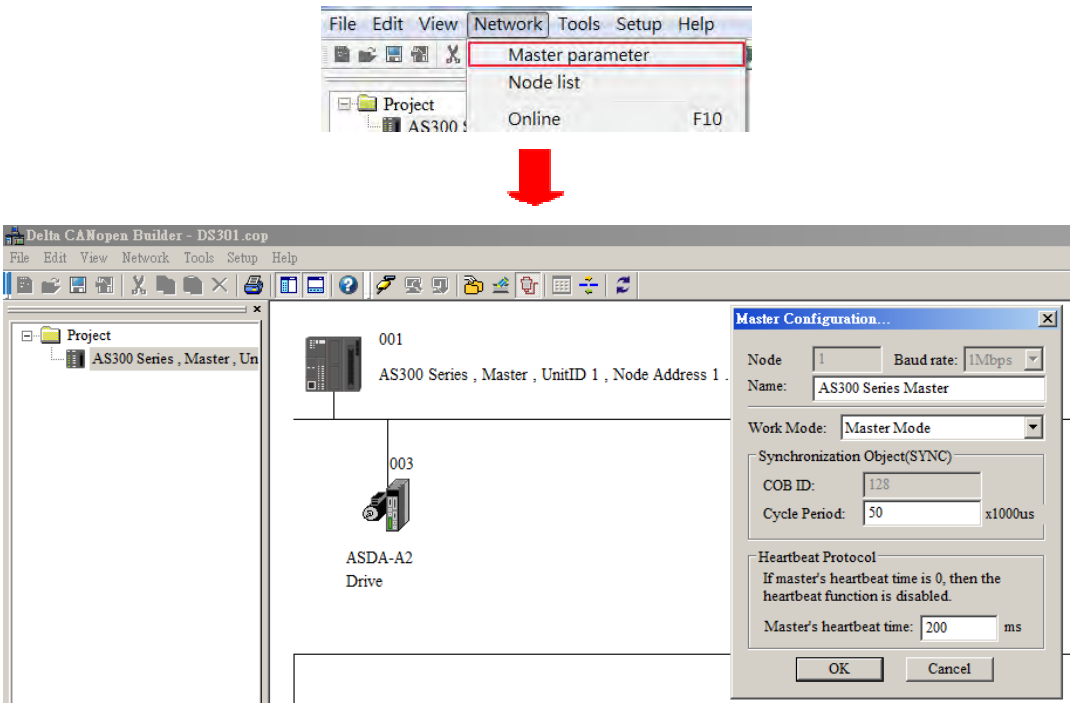
B. Click the Online  button on the Toolbar to enter on-line mode.



C. Click the Scan Network button on the Toolbar to scan the network.



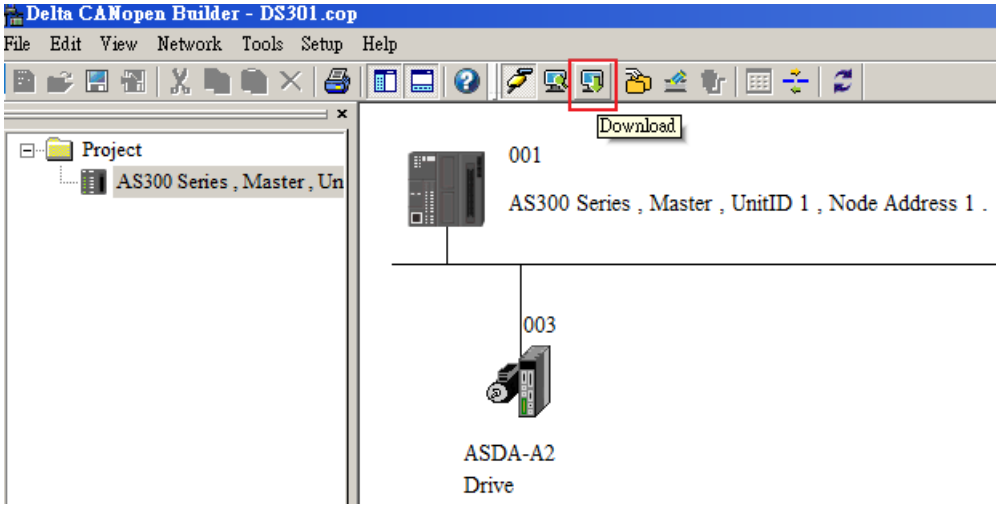
D. On the **Network** menu, click **Master parameter** to display the Master Configuration dialog box.



Item	Explanation	Default
Node ID	Node ID of AS300 series PLC on the CANopen network	1
Baud rate	CANopen communication rate	1M bit/second
Work mode	CANopen master/slave mode	Master
Cycle period	Cycle time for sending one SYNC message	50ms
Master heartbeat time	Interval time for sending the master heartbeat message	200ms

Configure the CANopen communication stations and rates in HWCONFIG.

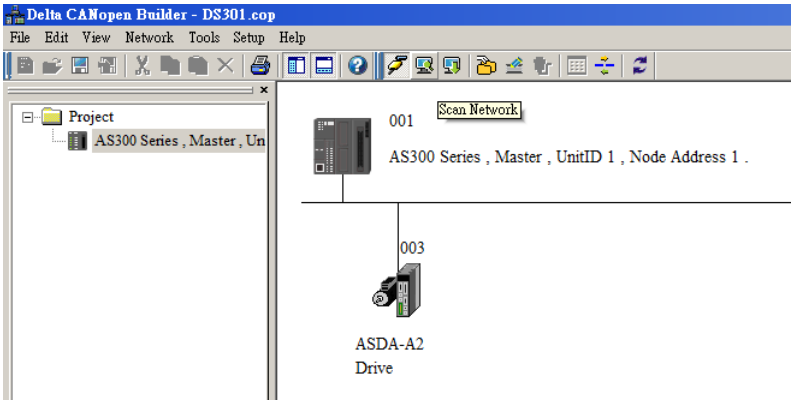
E. After you complete the previous steps, click the Download button on the Toolbar to download the parameters to the PLC.



Note: you must reboot the AS300 Series PLC to enable the downloaded parameters.

4. Scanning the Network:

On the **Network** menu, click **Online** or click the button to scan for the master and slaves on the CANopen network. The master and slave found by the scan appear in CANopen Builder. For more information, refer to Section 11.1.1 in the CANopen Builder software help file.



5. Configurating Nodes:

Double click the slave icon in CANopen Builder to display the **Node configuration** dialog box.

- **Error Control Protocol**

  Sets the error control protocol for the master to monitor if the slave is offline.

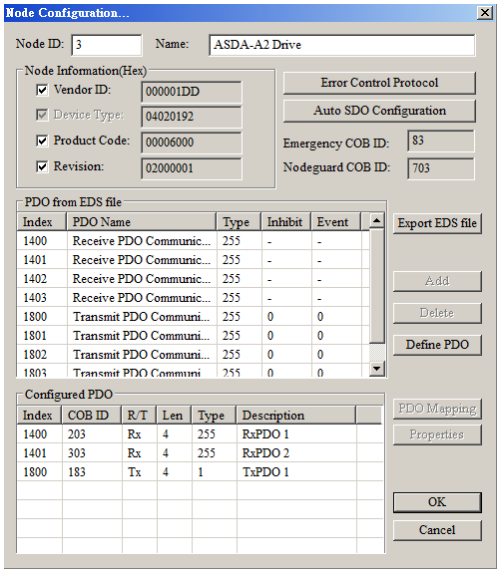
- **Auto SDO Configuration**

  Perform one write action to the slave parameter with SDO. The write action is finished when the slave enters the operational state from the pre-operational state. You can configure up to 20 SDOs by clicking Auto SDO configuration.

- **PDO Mapping** and **Properties**

  Sets the mapping parameter and transmission type for the PDO.

  For more details on the function of these buttons, refer to Section 11.1.1 in the CANopen Builder software help file.

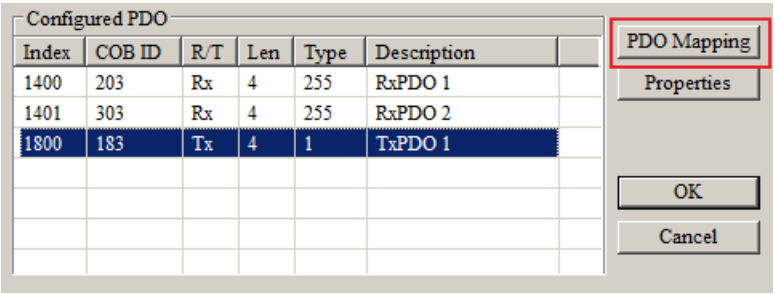


- PDO Mapping:

  RxPDO1: mapping parameter P1-09; transmission type 255.

  RxPDO2: mapping parameter P3-06, P4-07; transmission type 255.

  TxPDO1: mapping parameter P0-09; transmission type 1.



- PDO transmission type :

  PDOs can be classified into RxPDO or TxPDO. RxPDO data are sent from master to slave and TxPDO data are sent from slave to master.

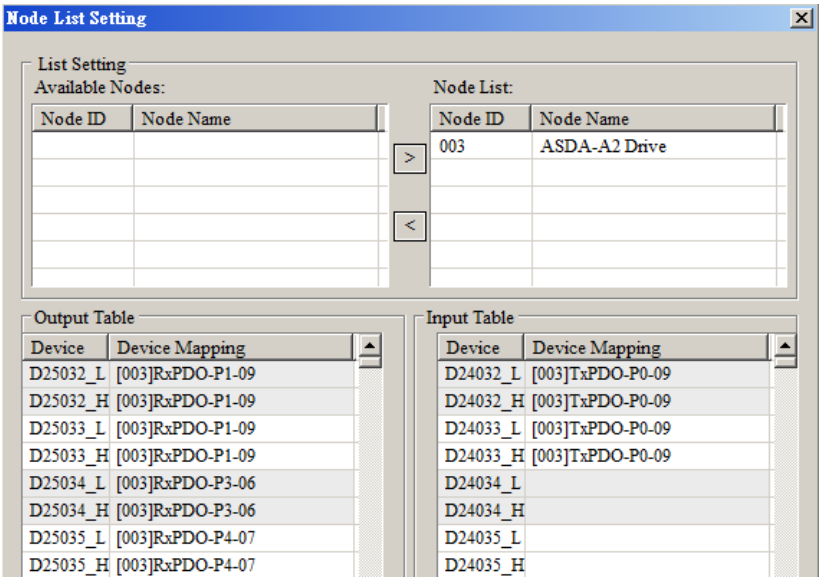
  The PDO transmission types can be synchronous or asynchronous. In synchronous transmission, the master sends out the SYNC message in a fixed cycle. You set the length of the cycle in the Master Properties dialog box (default is 50ms). In asynchronous transmission, the message is sent out when the PDO mapping parameter changes.
  The following table describes the PDO Transmission types.

Transmission Type		Description	Remark
0	RxPDO	When any change in the mapped data happens, RxPDO data is sent out immediately. The data the slave receives is valid only when receiving the next SYNCH message. RxPDO data is not sent out if there is no change in the data.	SYNCH non-cycle
	TxPDO	When any change in the mapped data happens and the slave receives the SYNC message, the data are sent out immediately. The TxPDO data are valid immediately after master receives them. TxPDO data is not sent out if there is no change in the data.	
N （N:1–240）	RxPDO	After N messages are sent out, and regardless of whether the mapped data changes, the data that the slave receives is valid only when receiving the next SYNCH message.	SYNCH cycle
	TxPDO	After N messages are sent out and regardless of whether the mapped data is changed, the data that the master receives is valid at once.	
254	RxPDO	The mapped data is sent out immediately when it changes and is valid when the slaves receives it. RxPDO data is not sent out if there is no change in the data.	ASYNCH
	TxPDO	The slave sends out the data once every one Event timer time. After that, the TxPDO data is not allowed to be sent out within an inhibit timer time. When the Event timer and Inhibit timer are both equal to 0, the slave sends TxPDO data to the master immediately when the data changes, and the data that master receives is immediately valid.	
255	Same as Type254		

Note**:**

- Synchronous transmission type can fulfill multi-axis motion at the same time.
- If you monitor a real-time changing parameter such as the actual rotation speed of the motor, set the TxPDO to the synchronous transmission type; otherwise the frequent change in the slave data can block the CANopen network.

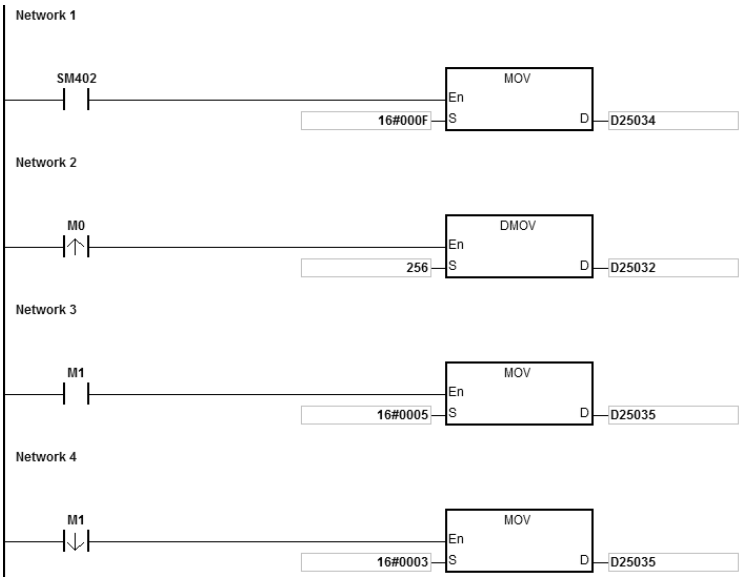
After you finish setting the above parameters, double click the master device to display the **Node List Setting** dialog box. Select ASDA-A2 Drive, and click **>** to move the A2 drive to the right-side list, and then download the configured data.



The mapping relation between master and slave:

AS300 Series PLC master register	Data transmission on CANopen bus	A2 device
D25032	⇨	Low word of P1-09 of servo
D25033		High word of P1-09 of servo
D25034		P3-06 of servo
D25037		P4-07 of servo
D24032	⇦	Low word of P0-09 of servo
D24033		High word of P0-09 of servo

6. Program control: D25032 is given the value 256 in ISPSoft; that is, the speed command is set as 256r/min in the following diagram.



7. Program explanation

When the AS300 Series PLC runs for the first time, set the parameter P3-06 for servo drive to F.

- When M0 switches from OFF to ON, the instruction writes 256 to D25032 and then writes the value to the servo parameter P1-09 through RxPDO1.
- When M1 switches from OFF to ON, change P4-07 to 5. DI1 and DI3 are ON. DI1 means the SERVO is ON and DI3 calls the speed specified by parameter P1-09 for servo rotation.
- When M1 switches from ON to OFF, the speed command becomes 0 and the motor stops running.

10.7 Object Dictionary

The following table lists the communication objects in the object dictionary.

Index	Subindex	Object name	Data type	Attribute	Default value
H'1000	H'00	Device type	Unsigned 32 bits	R	0x00000000
H'1001	H'00	Error register	Unsigned 8 bits	R	0
H'1005	H'00	COB-ID SYNC	Unsigned 32 bits	RW	0x00000080
H'1008	H'00	Manufacturer device name	Vis-String	R	AS300 Series PLC
H'1014	H'00	COB-ID EMCY	Unsigned 32 bits	R	0x80 + Node-ID
H'1016	--	Consumer heartbeat time			
	H'00	Number of valid subindex	Unsigned 8 bits	R	1
	H'01	Consumer heartbeat time	Unsigned 32 bits	RW	0
H'1017	H'00	Producer heartbeat time	Unsigned 16 bits	RW	0
H'1018	--	Identity Object			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	Vendor-ID	Unsigned 32 bits	R	0x000001DD
	H'02	Product code	Unsigned 32 bits	R	0x00000055
	H'03	Revision number	Unsigned 32 bits	R	0x00010002
H'1400	--	RxPDO1 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO1	Unsigned 32 bits	RW	0x00000200+ Node-ID
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1401	--	RxPDO2 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3

Index	Subindex	Object name	Data type	Attribute	Default value
	H'01	COB-ID of RxPDO2	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1402	--	RxPDO3 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO3	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
H'1402	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1403	--	RxPDO4 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO4	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1404	--	RxPDO5 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO5	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1405	--	RxPDO6 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO6	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1406	--	RxPDO7 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO7	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1407	--	RxPDO8 communication parameter			

Index	Subindex	Object name	Data type	Attribute	Default value
	H'00	Number of valid subindex	Unsigned 8 bits	R	3
	H'01	COB-ID of RxPDO8	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	0
H'1600	--	RxPDO1 mapping parameter			
H'1600	H'00	Number of valid subindex	Unsigned 8 bits	RW	4
	H'01	The first mapped object	Unsigned 32 bits	RW	0x20000110
	H'01	The second mapped object	Unsigned 32 bits	RW	0x20000210
	H'02	The third mapped object	Unsigned 32 bits	RW	0x20000310
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0x20000410
H'1601	--	RxPDO2 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1602	--	RxPDO3 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1603	--	RxPDO4 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1604	--	RxPDO5 mapping parameter			

Index	Subindex	Object name	Data type	Attribute	Default value
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
H'1604	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1605	--	RxPDO6 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1606	--	RxPDO7 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1607	--	RxPDO8 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'01	The second mapped object	Unsigned 32 bits	RW	0
	H'02	The third mapped object	Unsigned 32 bits	RW	0
	H'03	The fourth mapped object	Unsigned 32 bits	RW	0
H'1800	--	TxPDO1 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO1	Unsigned 32 bits	RW	0x00000180+ Node-ID
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
H'1800	H'05	Timer	Unsigned 16 bits	RW	100
H'1801	--	TxPDO2 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO2	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100

Index	Subindex	Object name	Data type	Attribute	Default value
H'1802	--	TxPDO3 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO3	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1803	--	TxPDO4 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO4	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1804	--	TxPDO5 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO5	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1805	--	TxPDO6 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO6	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
H'1805	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1806	--	TxPDO7 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO7	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100
H'1807	--	TxPDO8 communication parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	R	5
	H'01	COB-ID of TxPDO8	Unsigned 32 bits	RW	0x80000000
	H'02	Transmission mode	Unsigned 8 bits	RW	0xFF
	H'03	Inhibit time	Unsigned 16 bits	RW	50
	H'05	Timer	Unsigned 16 bits	RW	100

Index	Subindex	Object name	Data type	Attribute	Default value
H'1A00	--	TxPDO1 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	4
	H'01	The first mapped object	Unsigned 32 bits	RW	0x20010110
	H'02	The second mapped object	Unsigned 32 bits	RW	0x20010210
	H'03	The third mapped object	Unsigned 32 bits	RW	0x20010310
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0x20010410
H'1A01	--	TxPDO2 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A02	--	TxPDO3 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
H'1A02	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A03	--	TxPDO4 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
H'1A04	--	TxPDO5 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A05	--	TxPDO6 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A06	--	TxPDO7 mapping parameter			
	H'00	Number of valid subindex	Unsigned 8 bits	RW	0

Index	Subindex	Object name	Data type	Attribute	Default value
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0
H'1A07	--	TxPDO8 mapping parameter			
H'1A07	H'00	Number of valid subindex	Unsigned 8 bits	RW	0
	H'01	The first mapped object	Unsigned 32 bits	RW	0
	H'02	The second mapped object	Unsigned 32 bits	RW	0
	H'03	The third mapped object	Unsigned 32 bits	RW	0
	H'04	The fourth mapped object	Unsigned 32 bits	RW	0