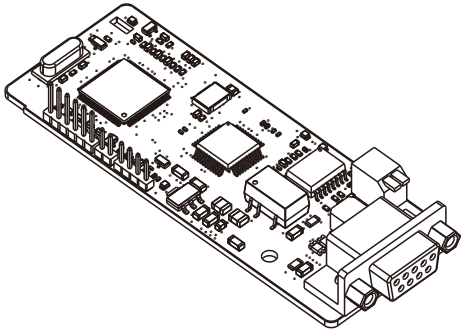
# Operation Manual

## Communication Card



SHENZHEN INVT ELECTRIC CO., LTD.

## Safety precautions

The expansion card can be installed and operated only by people who have taken part in professional training on electrical operation and safety knowledge, obtained the certification, and been familiar with all steps and requirements for installing, performing commissioning on, operating, and maintaining the device, and are capable of preventing all kinds of emergencies.

Before installing, removing, or operating the communication card, read the safety precautions described in this manual and the variable-frequency drive (VFD) operation manual carefully to ensure safe operation.

For any physical injuries or damage to the device caused due to your neglect of the safety precautions described in this manual and the VFD operation manual, our company shall not be held liable.

- You need to open the housing of the VFD when installing or removing the communication card. Therefore, you must disconnect all power supplies of the VFD and ensure that the voltage inside the VFD is safe. For details, see the description in the VFD operation manual. Severe physical injuries or even death may be caused if you do not follow the instructions.
- Store the communication card in a place that is dustproof and damp-proof without electric shocks or mechanical pressure.
- The communication card is electrostatic sensitive. Take measurements to prevent electrostatic discharge when performing operations involving it.
- Tighten the screws up when installing the communication card. Ensure that it is firmly fixed and properly grounded.

## Terminology and abbreviations

| | |
|---|---|
| CAN | Controller Area Network |
| COB | Communication object, a transmitted unit on a CAN network. Communication objects (COBs) carry data and can be transmitted through the whole network. A COB is part of a CAN message frame. |
| EDS | Electronic datasheet, an ASCII file for node configuration, required when a CANopen network is configured. An EDS file contains general information about nodes and their dictionary objects (parameters). |
| NMT | Network management, one of the CAN application-layer service elements in the CAN reference model. It is used for the initialization, configuration, and fault handling of a CAN network. |
| Object dictionary | Stores information about all COBs identified by a device. |
| PDO | Process data object, a type of COBs, used to transmit process data, such as control command, set values, state values, and actual values. |
| PDOn Tx | PDO command transmitted by a slave to the master, where n refers to 1, 2, 3, 4. |
| PDOn Rx | PDO command transmitted by the master and received by a slave, where n refers to 1, 2, 3, 4. |
| SDO | Service data object, a type of COB, used to transmit non-time key data, such as parameter values. |
| RO | Indicates read-only access. |
| RW | Indicates the read and write access. |
| SYNC | Indicates synchronous transmission. |
| Node-ID | Node ID, that is, address of a communication card. |
| 0x | Indicates that a number with this prefix is a hexadecimal value, for example, 0x10 indicates the decimal value 16. |

# Contents

# Chapter 1 Product confirmation

Check the following after receiving a communication expansion card product:

- Whether the communication card is damaged.
- Whether the received communication card is the one you purchase according to the bar code label on the PCB.
- Whether all the following items are contained in the product package:
- One communication card, one tie wrap, one tie, one M3 screw, and one manual.
- If the communication card is damaged, a wrong model is delivered, or some items are missing, contact the supplier in a timely manner.
- Obtain the EDS file of the communication card from INVT. The file is named *communication card model.eds*.
- Confirm the environmental requirements for application.

Table 1-1 Environmental requirements

| Item | Requirement |
|---|---|
| Operation temperature | -10–+50°C |
| Storage temperature | -20–+60°C |
| Relative humidity | 5%–95% |
| Other weather conditions | No condensation, ice, rain, snow, or hail;<br>solar radiation < 700 W/$m^2$ |
| Air pressure | 70–106 kPa |
| Vibration and impact | 5.8m/$s^2$ (0.6g) at the sine vibration of 9 Hz to 200 Hz |

# Chapter 2 PROFIBUS communication card

## 2.1 Overview

PROFIBUS communication cards are optional accessories for VFDs. They can be used to connect VFDs to PROFIBUS networks. On a PROFIBUS network, VFDs are slave devices. The following functions can be performed by using a PROFIBUS communication card:

- Transmit control commands (such as start, stop, and fault reset) to a VFD.
- Transmit speed or torque reference signals to a VFD.
- Obtain state values and actual values from a VFD.
- Modify parameter values of a VFD.

## 2.2 Features

1. PROFIBUS is an international open fieldbus standard that can implement data exchange between various automation components. It is widely applicable to automation in various industries, such as the manufacturing, process, building, transportation, and power industries. It provides effective solutions for implementing integrated automation and intelligentization of field devices.
2. PROFIBUS consists of three mutually compatible components, namely PROFIBUS-Decentralised Peripherals (DP), PROFIBUS-Process Automation (PA), and PROFIBUS-Fieldbus Message Specification (FMS). It adopts the master-slave mode and is generally used for periodic data exchange between VFD devices. PRNV PROFIBUS-DP adapter modules support only the PROFIBUS-DP protocol.
3. The transmission media of a PROFIBUS field bus are twisted pairs (complying with the RS-485 standard), paired cables, or optical cables. The maximum length of a fieldbus cable must be within the range of 100 m to 1200 m, and the specific length depends on the selected transmission rate (see the chapter of "Technical Data" in the VFD manual). A maximum of 31 nodes can be connected to one PROFIBUS network segment when no repeater is used. If repeaters are used, a maximum of 127 nodes (including the repeaters and master stations) can be connected.
4. In PROFIBUS communication, tokens are transmitted between master stations or by master stations to slave stations. Single-master or multi-master systems are supported. The node to respond to the command of a master is selected by the master station, generally a programmable logic controller (PLC). For cyclic master-slave user data transmission and non-cyclic master-master data transmission, a master can also transmit commands to multiple nodes in broadcast mode. When the broadcast mode is adopted, the nodes do not need to transmit feedback signals to the master. On PROFIBUS networks, nodes cannot communicate with each other.

5. The PROFIBUS protocol is described in details in the EN50170 standard. For more information about PROFIBUS, refer to the EN50170 standard.

## 2.3 Electrical connection

1. Node selection

   The node address of a device is unique on a PROFIBUS bus. The node address is set through the function parameter P15.01, and the value ranges from 0 to 127.

2. Fieldbus terminator

   Each fieldbus segment is configured with two bus terminators, one on each end, to prevent operation errors. Bus terminators can protect the fieldbus signal against electrical reflections. The dual in-line package (DIP) switch on the printed circuit board (PCB) of a communication card is used to connect to the fieldbus terminator. If the communication card is the last or first module on the network, the bus terminator must be set to ON. When a PROFIBUS D-sub connector with a built-in terminator is used, you must disconnect the communication card from the terminator.

## 2.4 Bus network connection

1. Bus communication interfaces

   The most common PROFIBUS transmission mode is the shielded twisted-pair copper cable transmission, in which shielded twisted-pair copper cables (complying with the RS-485 standard) are used.

   The basic characteristics of this transmission technology are described as follows:

   - Network topology: Linear bus with one active fieldbus terminal resistor on each end
   - Media: Shielded or unshielded twisted-pair cables, depending on the EMC environmental conditions
   - Number of stations: 32 on each network segment (without repeater); a maximum of 127 (with repeaters)
   - Plug connection: 9-pin D-type plug. The following figure shows the pins of the connector.

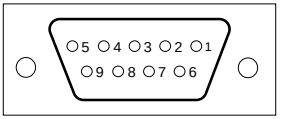



Figure 2-1 Plug of the connector

Table 2-1 Connector pins

| Connector pin | | Description |
|---|---|---|
| 1 | - | Unused |
| 2 | - | Unused |
| 3 | B-Line | Data+ (twisted-pair wire 1) |
| 4 | RTS | Transmitting requests |
| 5 | GND_BUS | Isolation ground |
| 6 | +5V BUS | Isolated 5 V DC power supply |
| 7 | - | Unused |
| 8 | A-Line | Data- (twisted-pair wire 2) |
| 9 | - | Unused |
| Housing | SHLD | PROFIBUS cable shielding wire |

The +5V and GND_BUS pins are used for bus terminators. Optical transceivers (RS-485) and some other devices may need to obtain external power supplies through these pins.

For some devices, the transmission direction is determined by using the RTS pin. In regular application, only the A-Line, B-Line, and SHLD pins are used.

It is recommended that you use the standard DB9 connectors manufactured by Siemens. If the communication baud rate is required to be higher than 187.5 kbps, strictly follow the wiring standards stipulated by Siemens.

2. Repeaters

A maximum of 32 stations (including the master station) can be connected to each fieldbus segment. If the number of stations to be connected to a fieldbus segment exceeds 32, you need to use repeaters to connect the fieldbus segments. Generally, the number of repeaters connected in series cannot exceed 3.

**Note:** No station address is provided for repeaters, but they are calculated as stations.

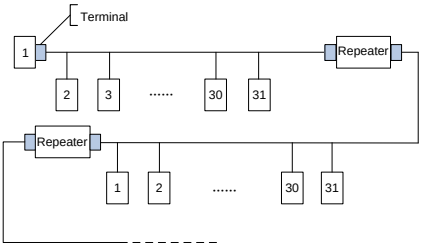



Figure 2-2 Repeaters

3. Transmission rates and maximum transmission distances

The maximum length of a cable depends on the transmission rate. Table 2-2 describes the transmission rates and corresponding transmission distances.

Table 2-2 Transmission rates and corresponding transmission distances

| Transmission rate (kbps) | A-type wire (m) | B-type wire (m) |
|---|---|---|
| 9.6 | 1200 | 1200 |
| 19.2 | 1200 | 1200 |
| 93.75 | 1200 | 1200 |
| 187.5 | 1000 | 600 |
| 500 | 400 | 200 |
| 1500 | 200 | ----- |

Table 2-3 Transmission wire parameters

| Parameter | A-type wire | B-type wire |
|---|---|---|
| Impedance (Ω) | 135–165 | 100–130 |
| Capacitance of a unit length (pF/m) | < 30 | < 60 |
| Circuit resistance (Ω/km) | 110 | -------- |
| Wire core diameter (mm) | 0.64 | > 0.53 |
| Sectional area of wire core ($mm^2$) | > 0.34 | > 0.22 |

Besides the shielded twisted-pair copper cables, you can also use optical fibers for transmission in a PROFIBUS system. When a PROFIBUS system is applied in an environment with strong electromagnetic interference, you can use optical fiber conductors to increase the high-speed transmission distance. Two types of optical fiber conductors can be used. One is low-cost plastic fiber conductors that can be used when the transmission distance is shorter than 50 m; and the other is glass fiber conductors that can be used when the transmission distance is shorter than 1 km.

4. PROFIBUS bus connection diagram

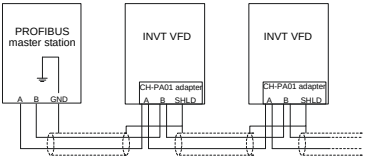



Figure 2-3 PROFIBUS bus connection

Figure 2-3 shows the terminal wiring. The cables are standard PROFIBUS cables, each

consisting of a twisted pair and shielding layer. The shielding layers of PROFIBUS cables are directly grounded on all nodes. You can select a proper grounding mode based on the actual situation on site.

**Note:**

1. When connecting the stations, ensure that the data cables are not twisted together. For systems to be used in environments with strong electromagnetic radiation, you need to use cables with shielding layers. The shielding layers can improve electromagnetic compatibility (EMC).

2. If shielding braid or shielding foil is used, connect the two ends of it to the protective ground and cover an area as large as possible to ensure high conductivity. In addition, data cables need to be separated from high-voltage cables.

3. When the data transmission rate is higher than 500 kbit/s, do not use short stub. Use the plugs available in the market. Data input and output cables can be directly connected to those plugs, and the plug of the communication card can be connected or disconnected at any time without interrupting data communication of other stations.

## 2.5 System configuration

1. System configuration

   After the communication card is properly installed, you need to configure the master station and VFD to enable the communication between the master station and communication card.

   One device description file named GSD file is required for each PROFIBUS slave station on the PROFIBUS bus. The GSD file is used to describe the characteristics of the PROFIBUS-DP device. The software we provide for users includes information about the GSD file of the VFD. You can obtain the type definition files (GSD files) of various masters from us.

Table 2-4 Communication card configuration parameters

| Parameter No. | Parameter name | Setting options | | Default setting |
|---|---|---|---|---|
| 0 | Module type | Read-only | | PROFIBUS-DP |
| 1 | Node address | 0–99 | | 2 |
| 2 | Baud rate setting | kbit/s | 0: 9.6 | 6 |
| | | | 1: 19.2 | |
| | | | 2: 45.45 | |
| | | | 3: 93.75 | |

| Parameter No. | Parameter name | Setting options | | Default setting |
|---|---|---|---|---|
| | | Mbit/s | 4: 187.5 | |
| | | | 5: 500 | |
| | | | 6: 1.5 | |
| | | | 7: 3 | |
| | | | 8: 6 | |
| | | | 9: 9 | |
| | | | 10: 12 | |
| 3 | PZD3 | 0–65535 | | 0 |
| 4 | PZD4 | 0–65535 | | 0 |
| … | … | 0–65535 | | 0 |
| 10 | PZD12 | 0–65535 | | 0 |

2. Module type

   This parameter displays the model of the communication card detected by the VFD. You cannot modify the value of this parameter. If the parameter is not defined, communication between the communication card and VFD cannot be established.

3. Node address

   On the PROFIBUS network, each device corresponds to one unique node address. The node address is set through P15.01.

4. GSD file

   One device description file named GSD file is required for each PROFIBUS slave station on the PROFIBUS bus. The GSD file is used to describe the characteristics of the PROFIBUS-DP device. The GSD file includes all parameters defined for the device, including the supported bard rate, supported information length, input/output data amount, and definitions of diagnosis data.

   You can obtain the type definition files (GSD files) of various masters from INVT's official website and copy the GSD files to the corresponding subdirectories on the configuration tool software. For details about the operation and how to configure the PROFIBUS system, see the instructions for the related system configuration software.

## 2.6 PROFIBUS-DP communication

1. PROFIBUS-DP

   PROFIBUS-DP is a distributed input/output (I/O) system. It enables a master to use a large number of peripheral modules and on-site devices. Data transmission is periodic: The master reads information input from a slave and then transmits a feedback signal to

the slave.

2. SAP

The PROFIBUS-DP system uses the services at the data link layer (Layer 2) through service access points (SAPs). Functions of each SAP are clearly defined. For more information about SAPs, see the related PROFIBUS master user manuals, that is, PROFIdrive—PROFIBUS models or EN50170 standards (PROFIBUS protocol) for variable-speed drives.

3. PROFIBUS-DP information frame data structure

The PROFIBUS-DP system allows fast data exchange between the master and VFD devices. For VFD devices, data is always read and written in the master/slave mode. VFDs always function as slave stations, and one address is clearly defined for each slave station. PROFIBUS transmits 16-bit packets periodically. Figure 2-4 shows the structure of the packet.

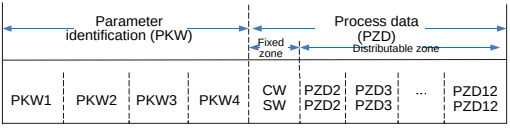



Figure 2-4 PROFIBUS-DP information frame data structure

Parameter zone:

PKW1—Parameter identification

PKW2—Array index number

PKW3—Parameter value 1

PKW4—Parameter value 2

Process data:

CW—Control word (transmitted from the master to a slave. For description, see Table 2-5)

SW—State word (transmitted from a slave to the master. For description, see Table 2-7.)

PZD—Process data (defined by users)

(When the process data is output by the master to a slave, it is a reference value; and when the process data is input by a slave to the master, it is an actual value.)

PZD zone (process data zone): The PZD zone in a communication packet is designed for controlling and monitoring a VFD. The master and slave stations always process the received PZD with the highest priority. The processing of PZD takes priority over that of PKW, and the

master and slave stations always transmit the latest valid data on the interfaces.

CWs and SWs

Using CWs is the basic method of the fieldbus system to control VFDs. A CW is transmitted by the fieldbus master station to a VFD device. In this case, the EC-TX-103 communication card functions as a gateway. The VFD device responds to the bit code information of the CW and feeds state information back to the master through an SW.

Reference value: A VFD device may receive control information in multiple channels, including analog and digital input terminals, VFD control panel, and communication modules (such as RS485 and EC-TX-103 communication cards). To enable the control over VFD devices through PROFIBUS, you need to set the communication module as the controller of the VFD device.

Actual value: An actual value is a 16-bit word that includes information about VFD device operation. The monitoring function is defined through VFD parameters. The conversion scale of an integer transmitted as an actual value from the VFD device to the master depends on the set function. For more description, see the related VFD operation manual.

**Note:** A VFD device always checks the bytes of a CW and reference value.

Task packet (master station -> VFD)

CW: The first word in a PZD task packet is a VFD CW. Table 2-5 describes Goodrive350 series VFD CWs.

Table 2-5 Goodrive350 series VFD CWs

| Bit | Name | Value | State to be entered/description |
|---|---|---|---|
| 0–7 | Communication-based control command | 1 | Forward running |
| | | 2 | Reverse running |
| | | 3 | Forward jogging |
| | | 4 | Reverse jogging |
| | | 5 | Decelerating to stop |
| | | 6 | Coasting to stop |
| | | 7 | Fault reset |
| | | 8 | Jogging stopped |
| | | 9 | Emergency stop |
| 8 | Enabling writing | 1 | Enabling writing (mainly through PKW1 to PKW4) |
| 9–10 | Motor group setting | 00 | Motor 1 |
| | | 01 | Motor 2 |
| 11 | Control mode switching | 1 | Enabling the switching between torque control and speed control |

| Bit | Name | Value | State to be entered/description |
|---|---|---|---|
| | | 0 | No switching |
| 12 | Resetting power consumption to zero | 1 | Enabling the function for resetting power consumption to zero |
| | | 0 | Disabling the function for resetting power consumption to zero |
| 13 | Pre-excitation | 1 | Enabling pre-excitation |
| | | 0 | Disabling pre-excitation |
| 14 | DC braking | 1 | Enabling DC braking |
| | | 0 | Disabling DC braking |
| 15 | Heartbeat reference | 1 | Enabling heartbeat |
| | | 0 | Disabling heartbeat |

Reference value (REF): The second to twelfth words in a PZD task packet are the main settings. The main frequency settings are provided by the main setting signal source. Table 2-6 describes the settings of Goodrive350 series VFD.

Table 2-6 Settings of Goodrive350 series VFD

| Function code | Word | Value range | Default value |
|---|---|---|---|
| P15.02 | Received PZD2 | 0–31<br>0: Invalid<br>1: Set frequency (0–Fmax, unit: 0.01 Hz)<br>2: PID reference (0–1000, in which 1000 corresponds to 100.0%)<br>3: PID feedback (0–1000, in which 1000 corresponds to 100.0%)<br>4: Torque setting (-3000–+3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>5: Setting of the upper limit of forward running frequency (0–Fmax, unit: 0.01 Hz)<br>6: Setting of the upper limit of reverse running frequency (0–Fmax, unit: 0.01 Hz)<br>7: Upper limit of the electromotive torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>8: Upper limit of the brake torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor) | 0 |
| P15.03 | Received PZD3 | | 0 |
| P15.04 | Received PZD4 | | 0 |
| P15.05 | Received PZD5 | | 0 |
| P15.06 | Received PZD6 | | 0 |
| P15.07 | Received PZD7 | | 0 |
| P15.08 | Received PZD8 | | 0 |
| P15.09 | Received PZD9 | | 0 |
| P15.10 | Received | | 0 |

| Function code | Word | Value range | Default value |
|---|---|---|---|
| | PZD10 | 9: Virtual input terminal command, 0x000–0x3FF (corresponding to S8, S7, S6, S5, HDIB, HDIA, S4, S3, S2, and S1 in sequence) | |
| P15.11 | Received PZD11 | 10: Virtual output terminal command, 0x00–0x0F (corresponding to RO2, RO1, HDO, and Y1 in sequence) | 0 |
| P15.12 | Received PZD12 | 11: Voltage setting (for V/F separation) (0–1000, in which 1000 corresponds to 100.0% of the rated voltage of the motor)<br>12: AO output setting 1 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>13: AO output setting 2 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>14: MSB of position reference (signed number)<br>15: LSB of position reference (unsigned number)<br>16: MSB of position feedback (signed number)<br>17: LSB of position feedback (unsigned number)<br>18: Position feedback setting flag (position feedback can be set only after this flag is set to 1 and then to 0)<br>19: Function parameter mapping (PZD2–PZD12 correspond to P14.49–P14.59)<br>20–31: Reserved | 0 |

Response packet (VFD -> master station)

SW: The first word in a PZD response packet is a VFD SW. Table 2-7 describes the VFD SWs.

Table 2-7 Goodrive350 series VFD SWs

| Bit | Name | Value | State to be entered/description |
|---|---|---|---|
| 0–7 | Running state | 1 | In forward running |
| | | 2 | In reverse running |
| | | 3 | Stopped |
| | | 4 | Faulty |
| | | 5 | POFF |
| | | 6 | In pre-excitation |
| 8 | Bus voltage established | 1 | Ready to run |
| | | 0 | Not ready to run |
| 9–10 | Motor group feedback | 0 | Motor 1 |
| | | 1 | Motor 2 |
| 11 | Motor type feedback | 1 | Synchronous motor |

| Bit | Name | Value | State to be entered/description |
|---|---|---|---|
| | | 0 | Asynchronous motor |
| 12 | Overload pre-alarm feedback | 1 | Overload pre-alarm generated |
| | | 0 | No overload pre-alarm generated |
| 13 | Run/Stop mode | 0 | Keypad-based control |
| | | 1 | Terminal-based control |
| 14 | | 2 | Communication-based control |
| | | 3 | Reserved |
| 15 | Heartbeat feedback | 1 | Heartbeat feedback |
| | | 0 | No heartbeat feedback |

Actual value (ACT): The second to twelfth words in a PZD task packet are the main actual values. The main actual frequency values are provided by the main actual value signal source.

Table 2-8 Actual state values of Goodrive350 series VFD

| Function code | Word | Value range | Default value |
|---|---|---|---|
| P15.13 | Transmitted PZD2 | 0–31<br>0: Invalid<br>1: Running frequency (×100, Hz)<br>2: Set frequency (×100, Hz)<br>3: Bus voltage (×10, V)<br>4: Output voltage (×1, V)<br>5: Output current (×10, A)<br>6: Actual output torque (×10, %)<br>7: Actual output power (×10, %)<br>8: Rotating speed of the running (×1, RPM)<br>9: Linear speed of the running (×1, m/s)<br>10: Ramp frequency reference<br>11: Fault code<br>12: AI1 value (×100, V)<br>13: AI2 value (×100, V)<br>14: AI3 value (×100, V)<br>15: HDIA frequency (×100, kHz)<br>16: Terminal input state<br>17: Terminal output state<br>18: PID reference (×100, %)<br>19: PID feedback (×100, %)<br>20: Rated torque of the motor<br>21: MSB of position reference (signed number)<br>22: LSB of position reference (unsigned | 0 |
| P15.14 | Transmitted PZD3 | | 0 |
| P15.15 | Transmitted PZD4 | | 0 |
| P15.16 | Transmitted PZD5 | | 0 |
| P15.17 | Transmitted PZD6 | | 0 |
| P15.18 | Transmitted PZD7 | | 0 |
| P15.19 | Transmitted PZD8 | | 0 |
| P15.20 | Transmitted PZD9 | | 0 |
| P15.21 | Transmitted PZD10 | | 0 |
| P15.22 | Transmitted PZD11 | | 0 |
| P15.23 | Transmitted PZD12 | | 0 |

| Function code | Word | Value range | Default value |
|---|---|---|---|
| | | number)<br>23: MSB of position feedback (signed number)<br>24: LSB of position feedback (unsigned number)<br>25: State word 2<br>26: HDIB frequency value (×100, kHz)<br>27: High-order bit of PG card pulse feedback<br>28: Low-order bit of PG card pulse feedback<br>29: High-order bit of PG card pulse reference<br>30: Low-order bit of PG card pulse reference<br>31: Function parameter mapping (PZD2–PZD12 correspond to P14.60–P14.70) | |

PKW zone (parameter identification flag PKW1—numerical zone): The PKW zone describes the processing mode of the parameter identification interface. A PKW interface is not a physical interface but a mechanism that defines the transmission mode (such reading and writing a parameter value) of a parameter between two communication ends.

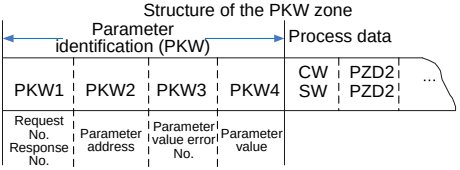



Figure 2-5 Parameter identification zone

In the periodic PROFIBUS-DP communication, the PKW zone consists of four 16-bit words. Table 2-9 describes each word in the PKW zone.

Table 2-9 Each word in the PKW zone

| First word PKW 1 (16 bits) | | |
|---|---|---|
| Bits 15–00 | Task or response identification flag | 0–7 |
| Second word PKW2 (16 bits) | | |
| Bits 15–00 | Basic parameter address | 0–247 |
| Third word PKW3 (16 bits) | | |
| Bits 15–00 | Value (most significant word) of a parameter or error code of the returned value | 00 |
| Fourth word PKW4 (16 bits) | | |
| Bits 15–00 | Value (least significant word) of a parameter | 0–65535 |

**Note:** If the master station requests the value of a parameter, the values in PKW3 and PKW4 of the packet that the master station transmits to the VFD are no longer valid.

Task request and response: When transmitting data to a slave, the master uses a request number, and the slave uses a response number to accept or reject the request.

Table 2-10 describes the request and response functions.

Table 2-10 Task identification flag PKW1

| **Request No. (from the master to a slave)** | | **Response signal** | |
|---|---|---|---|
| **Request No.** | **Function** | **Acceptance** | **Rejection** |
| 0 | No task | 0 | – |
| 1 | Requesting the value of a parameter | 1, 2 | 3 |
| 2 | Modifying a parameter value (one word) [modifying the value only on RAM] | 1 | 3 or 4 |
| 3 | Modifying a parameter value (two words) [modifying the value only on RAM] | 2 | 3 or 4 |
| 4 | Modifying a parameter value (one word) [modifying the value on both RAM and EEPROM] | 1 | 3 or 4 |
| 5 | Modifying a parameter value (two words) [modifying the value only on both RAM and EEPROM] | 2 | 3 or 4 |

The requests #2, #3, and #5 are not supported currently.

Table 2-11 Response identification flag PKW1

| Response No. (from a slave to the master) | |
|---|---|
| **Response No.** | **Function** |
| 0 | No response |
| 1 | Transmitting the value of a parameter (one word) |
| 2 | Transmitting the value of a parameter (two words) |
| 3 | The task cannot be executed and one of the following error number is returned:<br>1: Invalid command<br>2: Invalid data address<br>3: Invalid data value<br>4: Operation failure<br>5: Password error<br>6: Data frame error<br>7: Parameter read only<br>8: Parameter cannot be modified during VFD running<br>9: Password protection<br>10: Function code mapping failed |
| 4 | Reserved |

PKW examples

Example 1: Reading the value of a parameter

You can set PKW1 to 1 and PKW2 to 0A to read a frequency set through keypad (the address of the frequency set through keypad is 10), and the value is returned in PKW4. The following data is in hexadecimal format.

Request (master station -> VFD)

| | PKW1 | | PKW2 | | PKW3 | | PKW4 | | CW | | PZD2 | | PZD3 | | ... | PZD12 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Request | 00 | 01 | 00 | 0A | 00 | 00 | 00 | 00 | xx | xx | xx | xx | xx | xx | ... | xx | xx |

0010: Parameter address

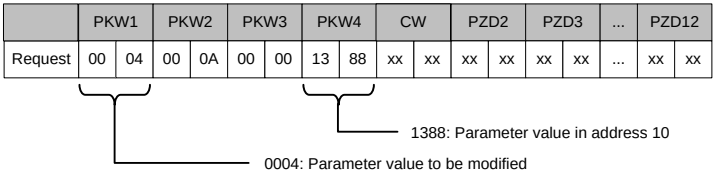
0001: Request for reading parameter values

Response (VFD -> master station)

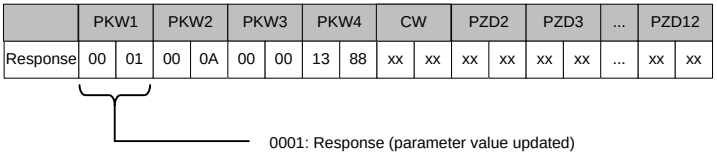

Example 2: Modifying the value of a parameter (on both RAM and EEPROM)

You can set PKW1 to 4 and PKW2 to 10 to modify a frequency set through keypad (the address of the frequency set through keypad is 10), and the value to be modified (50.00) is in PKW4.

Request (master station -> VFD)




Response (VFD-> master station)




PZD examples: The transmission of the PZD zone is implemented through VFD function code settings. For the function codes, see the related INVT VFD operation manual.

Example 1: Reading the process data of a VFD

In this example, PZD3 is set to "8: Rotating speed of the running" through the VFD parameter P15.14. This operation sets the parameter forcibly. The setting remains until the parameter is set to another option.

Response (VFD -> master station)

| | PKW1 | | PKW2 | | PKW3 | | PKW4 | | CW | | PZD2 | | PZD3 | | … | PZD12 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Response | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | 00 | 0A | … | xx | xx |

Example 2: Writing process data to a VFD device

In this example, PZD3 is set to "2: PID reference" through the VFD parameter P15.03. The parameter specified in each request frame is updated with the information contained in PZD3 until another parameter is specified.

Request (master station -> VFD)

| | PKW1 | | PKW2 | | PKW3 | | PKW4 | | CW | | PZD2 | | PZD3 | | … | PZD12 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Response | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | 00 | 00 | … | xx | xx |

Subsequently, the information contained in PZD3 is used as tractive force reference in each request frame until another parameter is specified.

## 2.7 Example of PROFIBUS-DP communication networking

1. Preparation before networking

Hardware: One PC, three PPROFIBUS communication cards, three GD350 VFDs, and one Siemens PLC S7-300.

Software: Win10 system and Siemens TIAPORTAL V13.

2. GD350 VFD parameter configuration

Set P00.01 (Channel of running commands) to 2 (Communication).

Set P00.02 (Communication channel of running commands) to 1 (PROFIBUS communication).

Set P00.06 (Frequency A command setting mode) to 9 (PROFIBUS communication).

Module address (P15.01) of three VFDs are set to 3, 4, and 5 respectively.

Set P15.02 (Received PZD2) to 1 (Set frequency).

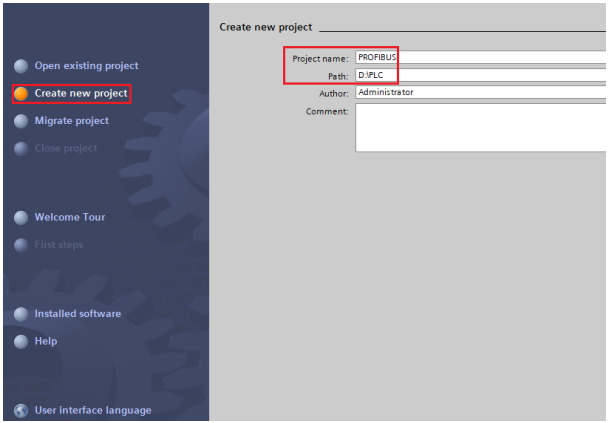
Set P15.13 (Transmitted PZD2) to 1 (Running frequency).

Set P15.14 (Transmitted PZD3) to 3 (Bus voltage).

3. PLC configuration

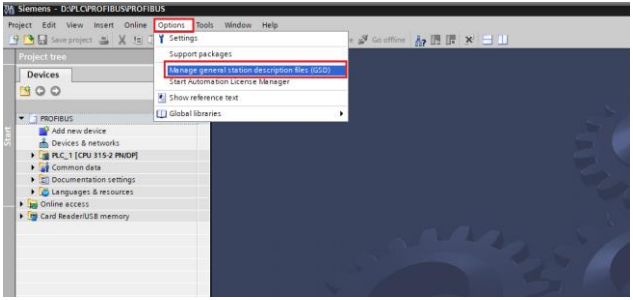
(1) Create a project.

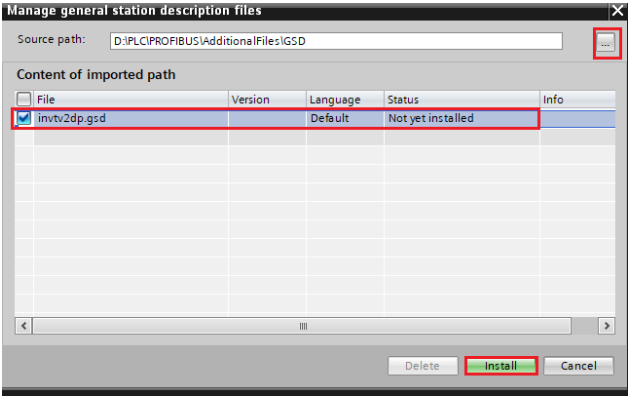
Click **Create new project**, fill in **Project name**, and select the path where the project is stored, as shown in the following figure.



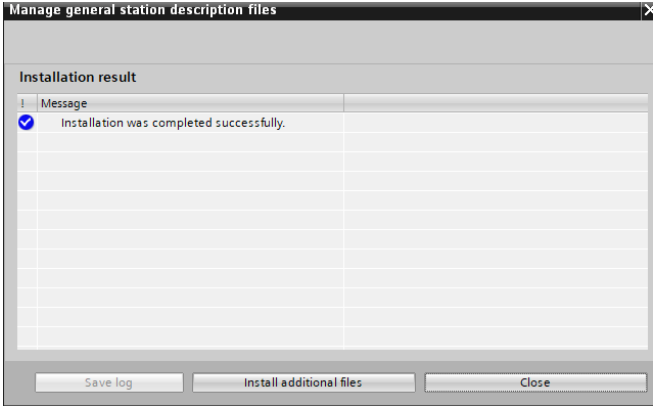

(2) Add GSD files.

In the following project view, choose **Options** on the toolbar, and choose **Manage general station description files (GSD)** from the drop-down list. Enter the directory where the INVT GSD file is located in the source path, select the GSD file, and click the **Install** button to start the installation.



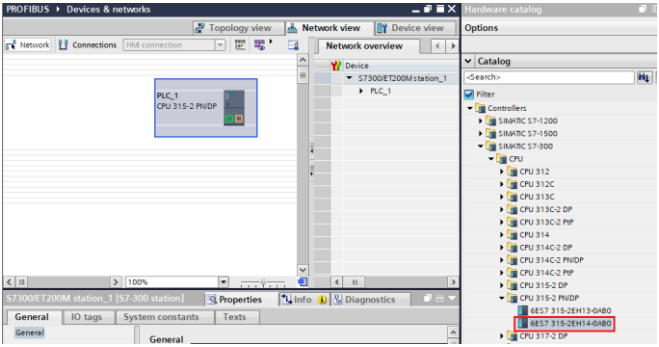



After the installation was complete successfully, a prompt pops up, indicating that the GSD file has been installed successfully.

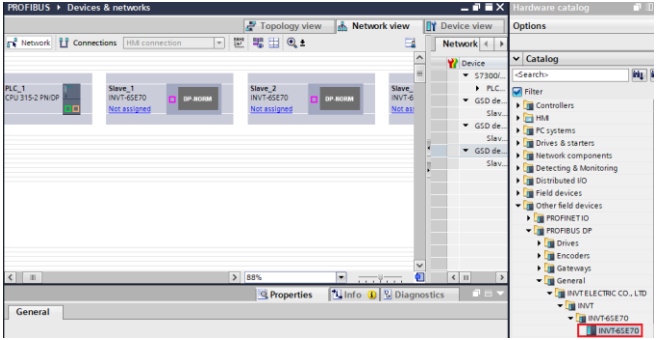



(3) Add the project device and PROFIBUS network.

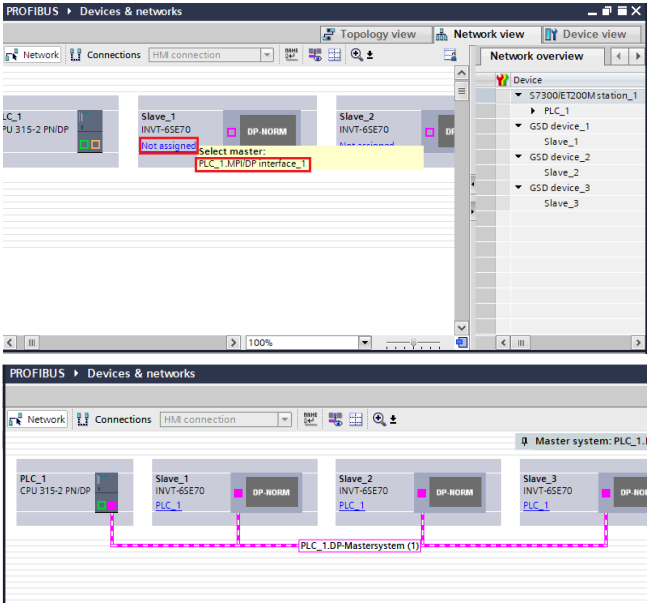
In the **Hardware catalog** on the right sidebar, choose **Controllers** > **SIMATIC S7-300** > **CPU** > **CPU 315-2 PN/DP** > **6ES7 315-2EH14-0A0B**, and double click the **6ES7 315-2EH14-0A0B** icon or drag it to the project.




In the **Hardware catalog pane**, choose **Other field devices** > **PROFIBUS DP** > **General** > **INVT ELECTRIC CO.,LTD** > **INVT** > **INVT-6SE70**, and double click the **INVT-6SE70** icon or drag it to the view of **Devices & networks**. The communication card is displayed as **Not assigned**. The example shows three slave bus networking, thus you need to add two slaves.




As shown in the following figure, click the **Not assigned** option of **INVT-6SE70** and select **PLC_1.MPI/DP interface_1**, and CPU and INVT-6SE70 in the network view are connected to the same PROFIBUS network. Click the **Not assigned** option of the remaining two PROFIBUS slaves and select **PLC_1.MPI/DP interface_1**, then PROFIBUS master and three slaves are connected to the same PROFIBUS network.

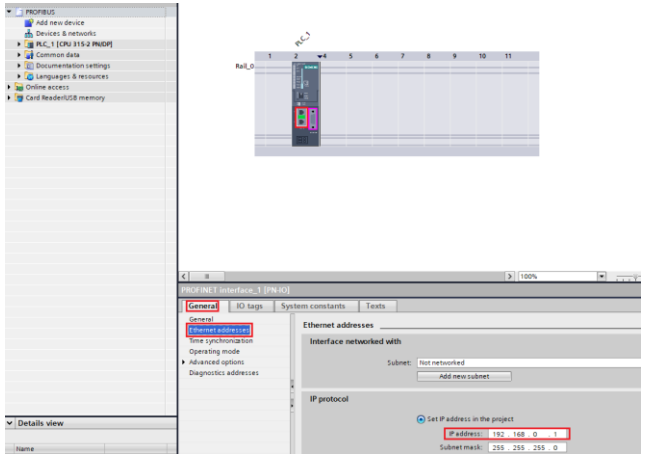


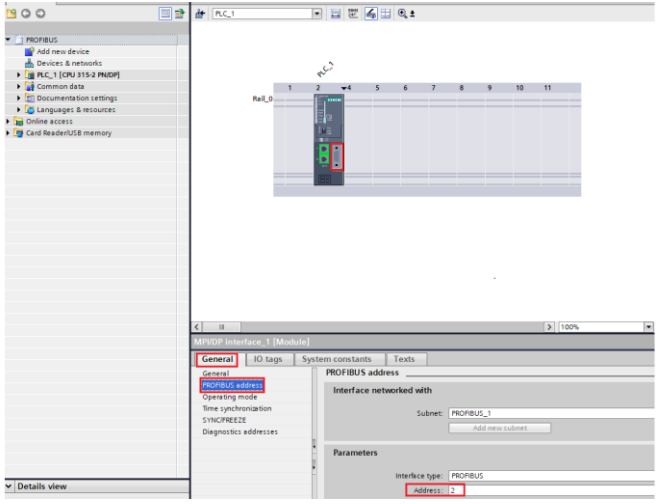

(4) Perform PROFIBUS master and slave setting.

PROFIBUS master setting

Click **PROFINET interface_1** network interface position in the PLC icon to enter the PROFINET interface_1 property editing interface of the PLC, as shown in the following figure. Click the **Ethernet addresses** option in the **General** list to set the IP address of the PLC and the IP address of the PC to be in the same subnet.

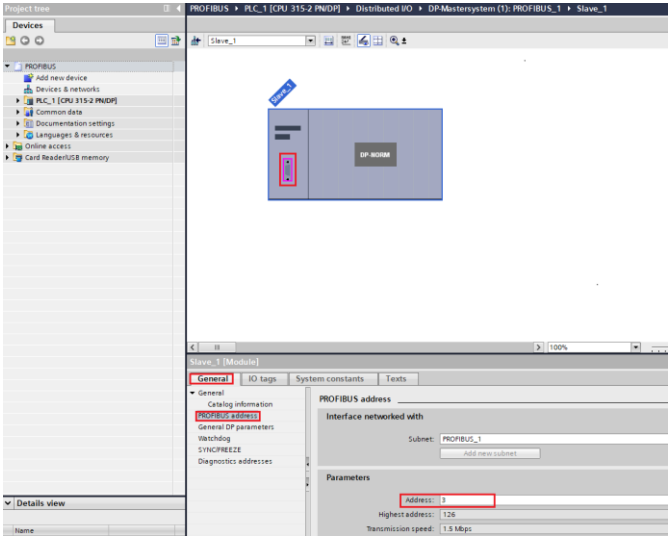
Click **MPI/DP interface_1** in the PLC icon to enter the MPI/DP interface_1 property editing interface of PLC. Click the **PROFIBUS address** option in the **General** list to set PROFIBUS address of PLC to 2.


PROFIBUS
Add new device
Devices & networks
PLC_1 [CPU 315-2 PN/DP]
Common data
Documentation settings
Languages & resources
Online access
Card Reader/USB memory
Details view
Name
PLC_1
Rail_0
PROFINET interface_1 [PN-IO]
General
IO tags
System constants
Texts
General
Ethernet addresses
Time synchronization
Operating mode
Advanced options
Diagnostics addresses
Ethernet addresses
Interface networked with
Subnet:
Not networked
Add new subnet
IP protocol
Set IP address in the project
IP address:
192 . 168 . 0 . 1
Subnet mask:
255 . 255 . 255 . 0


PROFIBUS
Add new device
Devices & networks
PLC_1 [CPU 315-2 PN/DP]
Common data
Documentation settings
Languages & resources
Online access
Card Reader/USB memory
Details view
PLC_1
Rail_0
MPI/DP interface_1 [Module]
General
IO tags
System constants
Texts
General
PROFIBUS address
Operating mode
Time synchronization
SYNC/FREEZE
Diagnostics addresses
PROFIBUS address
Interface networked with
Subnet:
PROFIBUS_1
Add new subnet
Parameters
Interface type:
PROFIBUS
Address:
2

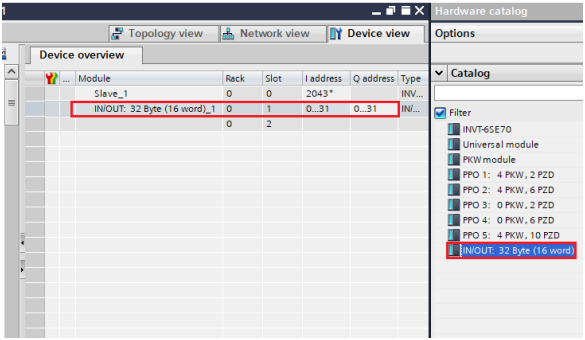
PROFIBUS slave setting

Double click the network interface position in the **INVT-6SE70** slave icon to enter the PROFIBUS interface editing interface. Click the **PROFIBUS address** option in the **General** list, set the slave address to 3, and set addresses of the remaining two slaves to 4 and 5 respectively.



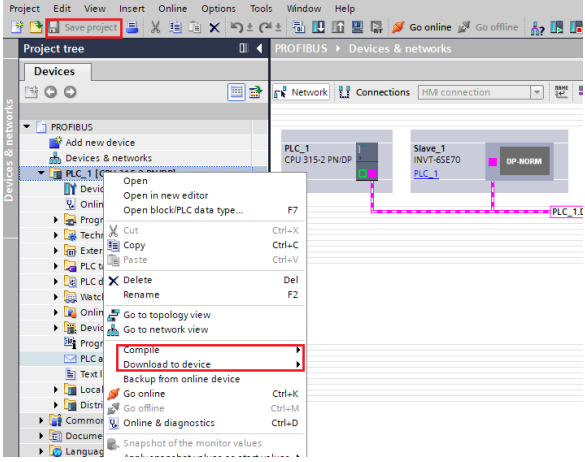

PROFIBUS slave module setting

Double click the **INVT-6SE70** slave icon in the **Devices & networks** view to enter the INVT device view interface. Double click the **IN/OUT:32Byte(16word)** module or drag it to the blank space in **Device view**. After **IN/OUT:32Byte(16word)** module is added to the project, you need to set I address and Q address of **IN/OUT:32Byte(16word)** to **0…31**. The remaining two slaves repeat this operation, and the addresses are increased by degrees.

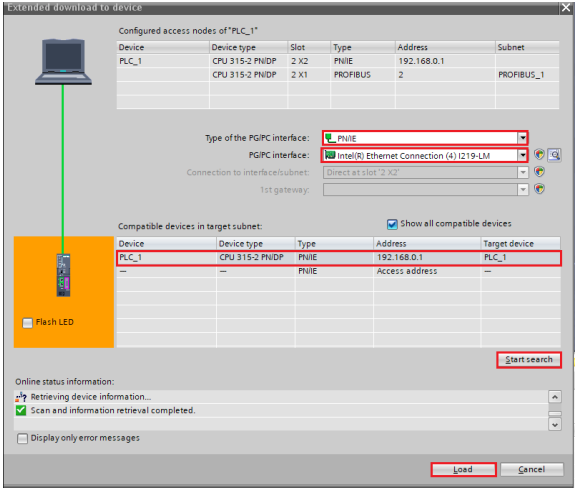



4. Save, compile, and download the project.

After PLC configuration is completed, you need to download the project configuration information to PLC S7-300, as shown in the following figure. Click **Save project** to save the entire project, and right click **PLC_1[CPU 315-2 PN/DP]** and choose **Compile** > **Hardware and software (change only)** to compile the entire project. Click the **Download to device** icon to download the project configuration to the PLC controller.

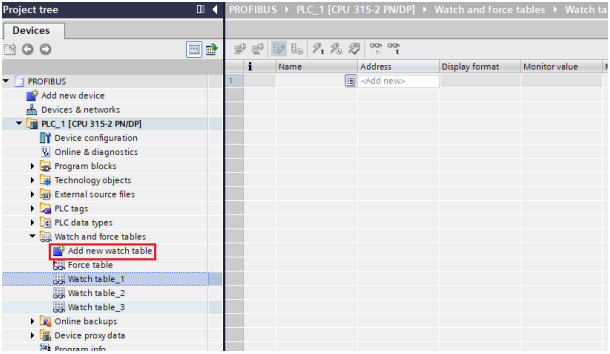


Select **PN/IE** from the drop-down list of **Type of the PG/PC interface**, and Click the **Start search** button in the lower right corner to start scanning and detecting PLC devices in the subnet, as shown in the following figure. After searching is completed, the PLCs that are connected to the PC will be displayed in the list of **Compatible devices in target subnet**. Click the **Download** button to download the configuration information and PLC program to the selected PLC.

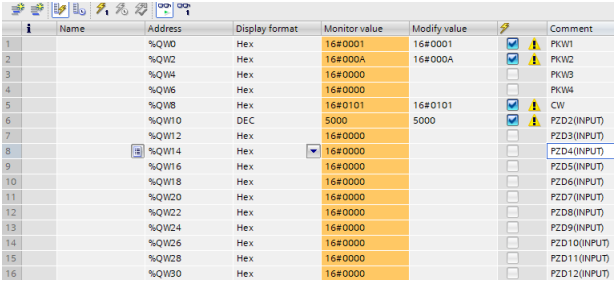



5. View VFD parameters.

Double click **Add new watch table** to create three watch tables for monitoring three VFD parameters respectively, as shown in the following figure.




Create target watch variables—PZD, PKW, control word and status word variables of the VFD in the newly created **Watch table_1**, and click **Watch all** and **Modify selected values at one time immediately**, as shown in the following figure. Operations in **Watch table_2** and **Watch table_3** are similar to those in **Watch table_1**.



| | Name | Address | Display format | Monitor value | Modify value | | Comment |
|---|---|---|---|---|---|---|---|
| 1 | | %QW0 | Hex | 16#0001 | 16#0001 | ☑ | PKW1 |
| 2 | | %QW2 | Hex | 16#000A | 16#000A | ☑ | PKW2 |
| 3 | | %QW4 | Hex | 16#0000 | | ☐ | PKW3 |
| 4 | | %QW6 | Hex | 16#0000 | | ☐ | PKW4 |
| 5 | | %QW8 | Hex | 16#0101 | 16#0101 | ☑ | CW |
| 6 | | %QW10 | DEC | 5000 | 5000 | ☑ | PZD2(INPUT) |
| 7 | | %QW12 | Hex | 16#0000 | | ☐ | PZD3(INPUT) |
| 8 | | %QW14 | Hex | 16#0000 | | ☐ | PZD4(INPUT) |
| 9 | | %QW16 | Hex | 16#0000 | | ☐ | PZD5(INPUT) |
| 10 | | %QW18 | Hex | 16#0000 | | ☐ | PZD6(INPUT) |
| 11 | | %QW20 | Hex | 16#0000 | | ☐ | PZD7(INPUT) |
| 12 | | %QW22 | Hex | 16#0000 | | ☐ | PZD8(INPUT) |
| 13 | | %QW24 | Hex | 16#0000 | | ☐ | PZD9(INPUT) |
| 14 | | %QW26 | Hex | 16#0000 | | ☐ | PZD10(INPUT) |
| 15 | | %QW28 | Hex | 16#0000 | | ☐ | PZD11(INPUT) |
| 16 | | %QW30 | Hex | 16#0000 | | ☐ | PZD12(INPUT) |

The above figure shows the examples of setting PKW read parameters and PZD.

- Set PKW1 to 0001 and PKW2 to 000A, indicating the request to read the value of P00.10 (set frequency through keypad).
- Set CW to 0101, indicating that PKW reading and writing function is enabled, and the VFD is controlled to run forward.
- Received PZD2 is set to 5000, indicating that the running frequency of the VFD is set to 50.00Hz.

| | | | | | | |
|---|---|---|---|---|---|---|
| 17 | %IW0 | Hex | 16#0001 | | | PKW1 |
| 18 | %IW2 | Hex | 16#000A | | | PKW2 |
| 19 | %IW4 | Hex | 16#0000 | | | PKW3 |
| 20 | %IW6 | DEC | 5000 | | | PKW4 |
| 21 | %IW8 | Hex | 16#4101 | | | SW |
| 22 | %IW10 | DEC | 5000 | | | PZD2(OUTPUT) |
| 23 | %IW12 | DEC | 5656 | | | PZD3(OUTPUT) |
| 24 | %IW14 | Hex | 16#0000 | | | PZD4(OUTPUT) |
| 25 | %IW16 | Hex | 16#0000 | | | PZD5(OUTPUT) |
| 26 | %IW18 | Hex | 16#0000 | | | PZD6(OUTPUT) |
| 27 | %IW20 | Hex | 16#0000 | | | PZD7(OUTPUT) |
| 28 | %IW22 | Hex | 16#0000 | | | PZD8(OUTPUT) |
| 29 | %IW24 | Hex | 16#0000 | | | PZD9(OUTPUT) |
| 30 | %IW26 | Hex | 16#0000 | | | PZD10(OUTPUT) |
| 31 | %IW28 | Hex | 16#0000 | | | PZD11(OUTPUT) |
| 32 | %IW30 | Hex | 16#0000 | | | PZD12(OUTPUT) |

The above figure shows the response results after setting PKW read parameters and PZD.

- PKW1: 0001, PKW2: 000A, PKW4: 5000, indicating that the value read from P00.10 is 5000, and 5000 indicates that the frequency set by keypad is 50.00Hz.
- SW: 4101, indicating that the running mode is set to communication control, the bus voltage is established, the VFD is ready to run, and the VFD is in forward running.
- Transmitted PZD2: 5000, indicating that the running frequency of the VFD is 50.00Hz.
- Transmitted PZD3: 5656, indicating that the bus voltage of the VFD is 565.6V.

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | %QW0 | Hex | 16#0004 | 16#0004 | | PKW1 |
| 2 | %QW2 | Hex | 16#000A | 16#000A | | PKW2 |
| 3 | %QW4 | Hex | 16#0000 | | | PKW3 |
| 4 | %QW6 | DEC | 4000 | 4000 | | PKW4 |
| 5 | %QW8 | Hex | 16#0105 | 16#0105 | | CW |
| 6 | %QW10 | DEC | 5000 | 5000 | | PZD2(INPUT) |
| 7 | %QW12 | Hex | 16#0000 | | | PZD3(INPUT) |
| 8 | %QW14 | Hex | 16#0000 | | | PZD4(INPUT) |
| 9 | %QW16 | Hex | 16#0000 | | | PZD5(INPUT) |
| 10 | %QW18 | Hex | 16#0000 | | | PZD6(INPUT) |
| 11 | %QW20 | Hex | 16#0000 | | | PZD7(INPUT) |
| 12 | %QW22 | Hex | 16#0000 | | | PZD8(INPUT) |
| 13 | %QW24 | Hex | 16#0000 | | | PZD9(INPUT) |
| 14 | %QW26 | Hex | 16#0000 | | | PZD10(INPUT) |
| 15 | %QW28 | Hex | 16#0000 | | | PZD11(INPUT) |
| 16 | %QW30 | Hex | 16#0000 | | | PZD12(INPUT) |

The above figure shows the examples of setting PKW write parameters and PZD.

- Set PKW1 to 0004, PKW2 to 000A and PKW4 to 4000, indicating that the value of P00.10 (set frequency through keypad) is changed to 40.00Hz.
- Set CW to 0105, indicating that PKW reading and writing function is enabled, and the VFD is controlled to decelerate to stop.
- Received PZD2 is set to 5000, indicating that the running frequency of the VFD is set to 50.00Hz.

| | | | | | | |
|---|---|---|---|---|---|---|
| 17 | | %IW0 | Hex | 16#0001 | | PKW1 |
| 18 | | %IW2 | Hex | 16#000A | | PKW2 |
| 19 | | %IW4 | Hex | 16#0000 | | PKW3 |
| 20 | | %IW6 | DEC | 4000 | | PKW4 |
| 21 | | %IW8 | Hex | 16#4103 | | SW |
| 22 | | %IW10 | DEC | 0 | | PZD2(OUTPUT) |
| 23 | | %IW12 | DEC | 5683 | | PZD3(OUTPUT) |
| 24 | | %IW14 | Hex | 16#0000 | | PZD4(OUTPUT) |
| 25 | | %IW16 | Hex | 16#0000 | | PZD5(OUTPUT) |
| 26 | | %IW18 | Hex | 16#0000 | | PZD6(OUTPUT) |
| 27 | | %IW20 | Hex | 16#0000 | | PZD7(OUTPUT) |
| 28 | | %IW22 | Hex | 16#0000 | | PZD8(OUTPUT) |
| 29 | | %IW24 | Hex | 16#0000 | | PZD9(OUTPUT) |
| 30 | | %IW26 | Hex | 16#0000 | | PZD10(OUTPUT) |
| 31 | | %IW28 | Hex | 16#0000 | | PZD11(OUTPUT) |
| 32 | | %IW30 | Hex | 16#0000 | | PZD12(OUTPUT) |

The above figure shows the response results after setting PKW write parameters and PZD.

- PKW1: 0001, PKW2: 000A, PKW4: 4000, indicating that the value read from P00.10 is 4000, and 4000 indicates that the frequency set by keypad is 40.00Hz.
- SW: 4103, indicating that the running mode is set to communication control, the bus voltage is established, the VFD is ready to run, and the VFD is in stopping.
- Transmitted PZD2: 0, indicating that the running frequency of the VFD is 0.00Hz.
- Transmitted PZD3: 5683, indicating that the bus voltage of the VFD is 568.3V.

# Chapter 3 CANopen communication card

## 3.1 Overview

1. Thanks for choosing INVT CANopen communication cards. This manual describes the function specifications, installation, basic operation and settings, and information about the network protocol. To ensure that you install and operate the product properly, read this manual and the communication protocol section in the VFD operation manual carefully before you use the product.
2. This manual only describes how to operate the CANopen communication card and the related commands but does not provide details about the CANopen protocol. For more information about the CANopen protocol, read the related specialized articles or books.
3. This communication card is defined as a CANopen slave station communication card and is used on a VFD that supports CANopen communication.
4. The CANopen communication of this communication card supports access to VFDs through process data objects (PDOs) and service data objects (SDOs). PDOs and SDOs are used to read the object dictionary defined by the manufacturer.

Table 3-1 Protocol selection relationship for switch SW2

| Switch SW2 | | |
|---|---|---|
| **1** | **2** | **Protocol** |
| OFF | OFF | CANopen |
| ON | OFF | CAN master/slave |

**Note: For EC-TX505C, before power-on, set the switch according to the protocol selection relationship to correspond to the actually used protocol.**

## 3.2 Features

1. Supported functions

- Supports the CAN2.0A protocol.
- Supports CANopen DS301.

2. Supported CANopen services

- PDO: Supports four pairs of PDO services (PDO1 TX to PDO4 TX, and PDO1 RX to PDO4 RX), where the PDO1 pair is used to read and write parameters of a VFD, and the PDO2 to PDO4 pairs are used to control and obtain the actual parameter values of the VFD in real time.
- SDO: SDO information adopts the "client/server" mode and is used to configure slave nodes and provide access to the object dictionary of each node.

- Supports the emergency service.
- Supports node protection (NMT Node Guarding).
- Supports heartbeat packets (Heartbeat Producer).
- Supports network management (NMT).
  - Supports NMT module control.
  - Supports NMT broadcast addresses.
  - Supports NMT error control.
  - Supports boot-up.
- Supports SYNC (1–240).
- Supports asynchronous transmission of 254 and 255.
- Supports disabled time.
- Supports event timers.
- Supports manufacturer-defined object dictionary. You can use SDOs to control and obtain the actual parameter values of a VFD in real time.

3. Non-supported CANopen services

- Saves object dictionary parameters at power outage
- Time stamp service

4. Supported CANopen addresses and baud rates

Table 3-1 Supported addresses and baud rates

| Item | Supported specification |
|---|---|
| Address | 1–127 (decimal) |
| Baud rate | 1000 kbps |
| | 800 kbps |
| | 500 kbps |
| | 250 kbps |
| | 125 kbps |
| | 100 kbps |
| | 50 kbps |
| | 20 kbps |

**Note:** To enable the CANopen functions (except the CANopen communication timeout fault time and baud rate), you need only to select the related PROFIBUS channels. If modification is made on the VFD operation manual, the operation is subject to the CANopen channel,

without prior notice in this manual.

## 3.3 Electrical wiring

Use shielding wires as the bus cable, if possible. It is recommended that you connect the shielding wire to the PE terminal of the communication card. When there are only two devices for CAN master-slave communication, both devices shall be connected to the terminal resistor. When there are more than two devices, the starting device and terminal device shall be connected to the terminal resistor. The terminal resistor of the communication card can be connected through its terminal resistor switch. Figure 3-1 shows the electrical wiring.

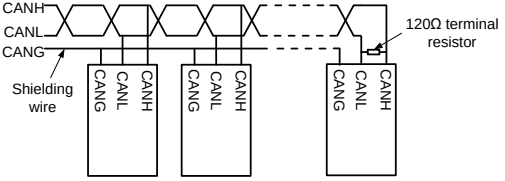



Figure 3-1 Electrical wiring diagram

## 3.4 Terminal wiring

### 3.4.1 Terminal layout

Table 3-2 CANopen card terminal layout

| | | | | | |
|---|---|---|---|---|---|
| | | PGND | PE | CANH | CANL |

### 3.4.2 Terminal functions

Table 3-3 CANopen card terminal functions

| Signal | Port | Terminal function |
|---|---|---|
| PGND | / | Isolation ground |
| PE | / | CAN bus shield |
| CANH | / | CAN bus high-level signal |
| CANL | / | CAN bus low-level signal |
| Terminal resistor | OFF-bounced up | No terminal resistor is connected between CAN_H and CAN_L. |
| | ON-pressed | A 120Ω terminal resistor is connected between CAN_H and CAN_L. |
| Switch applicable to cards | 00 | CANopen |
| | 10 | CAN master/slave |
| | 01 | Reserved |

| Signal | Port | Terminal function |
|---|---|---|
| | 11 | Reserved |

Table 3-4 Indicator functions

| Indicator | Definition | Function |
|---|---|---|
| LED1 | Status indicator | On: The expansion card is establishing a connection with the control board.<br>Blinking periodically: The expansion card is properly connected to the control board (the period is 1s, on for 0.5s, and off for the other 0.5s).<br>Off: The expansion card is disconnected from the control board. |
| LED2 | Run indicator | On: The communication card is running.<br>Off: A fault occurs. Check whether the reset pin of the communication card and the power supply are properly connected. The communication card is in the stopped state.<br>Blinks: The communication card is in the pre-operation state.<br>Blinks once: The communication card is in the stopped state. |
| LED3 | Alarm indicator | On: The CAN controller bus is off or the VFD has a fault.<br>Off: The communication card is in the working state. |
| LED4 | Power indicator | It is on since the control board feeds power to the communication card. |

## 3.5 Communication

### 3.5.1 Packet format

CAN2.0A packets are used to transmit data between the master station and bus nodes through data frames.

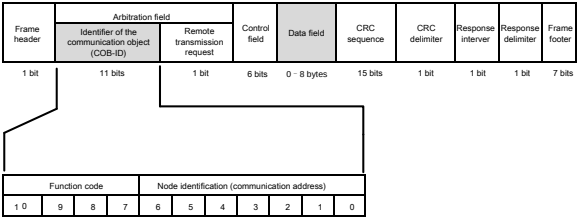



Figure 3-2 Packet structure

| Communication object | Function code (binary) | COB-ID (hexadecimal) |
|---|---|---|
| NMT | 0 | 0x00 |
| SYNC | 1 | 0x80 |
| EMERGENCY | 1 | 0x81–0xFF |
| PDO1 Tx | 11 | 0x181–0x1FF |
| PDO1 Rx | 100 | 0x201–0x27F |
| PDO2 Tx | 101 | 0x281–0x2FF |
| PDO2 Rx | 110 | 0x301–0x37F |
| PDO3 Tx | 111 | 0x381–0x3FF |
| PDO3 Rx | 1000 | 0x401–0x47F |
| PDO4 Tx | 1001 | 0x481–0x4FF |
| PDO4 Rx | 1010 | 0x501–0x57F |
| SDO Tx | 1011 | 0x581–0x5FF |
| SDO Rx | 1100 | 0x601–0x67F |
| Node protection | 1110 | 0x701–0x77F |

COB-IDs vary according to communication address, but for one command, the COB-IDs are within a certain range.

**Note:** The commands described in this manual are all data frames if it is not specified that they are remote frames.

### 3.5.2 CANopen state transition

The start sequence defined in the CANopen communication protocol is supported. Figure 3-3 shows the NMT state transition diagram.

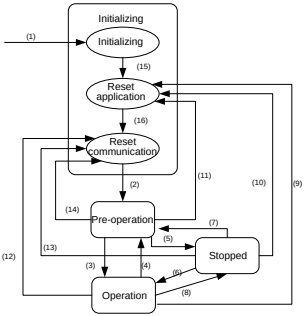



Figure 3-3 NMT state diagram

Table 3-2 NMT state transition

| State transition | Required triggering event |
|---|---|
| (1) | Automatic initialization after power-on |
| (2) | Automatic change after initialization |
| (3), (6) | Command of the NMT master station for starting a remote node |
| (4), (7) | Command of the NMT master station for entering the pre-operation state |
| (5), (8) | Command of the NMT master station for entering the stopped state |
| (9), (10), (11) | Command of the NMT master station for resetting a remote node |
| (12), (13), (14) | Command of the NMT master station for resetting a remote node communication parameter |

Different services are supported in different states, as described in Table 3-3.

Table 3-3 Services supported in various NMT states

| Service | Pre-operation state | Operation state | Stopped state |
|---|---|---|---|
| PDOs | No | Yes | No |
| SDOs | Yes | Yes | No |
| SYNC packets | Yes | Yes | No |
| Emergency packets | Yes | Yes | No |
| Network management | Yes | Yes | No |
| Error control | Yes | Yes | Yes |

**3.5.3 Management service command (NMT)**

This function is used by the master station to control the NMT states of slave station nodes.

- Command

Master station -> slave station

| COB-ID | Byte0 | Byte1 |
|---|---|---|
| 0x000 | Command specifier (CS) | Node-ID (Node ID) |

- Description

In this command, the COB-ID is 0×00. If Node-ID is set to 0, the command is broadcast to all CANopen slave stations, and each slave station must execute the NMT command. Table 3-4 describes the function of each CS.

Table 3-4 Function of each CS

| NMT CS | NMT service (control action) |
|---|---|
| 0x01 | Starts a slave station device. |
| 0x02 | Stops a slave station device. |
| 0x80 | Enables a slave station to enter the pre-operation state. |
| 0x81 | Resets a slave station. |
| 0x82 | Resets communication of a node. |

- Example

For example, the command to enable EC-TX505, whose node ID is 3, to enter the pre-operation state is described as follow.

| COB-ID | Byte0 | Byte1 |
|---|---|---|
| 0x000 | 0x80 | 0x03 |

For another example, the command to start all EC-TX505 nodes on the CANopen network is described as follows.

| COB-ID | Byte0 | Byte1 |
|---|---|---|
| 0x000 | 0x01 | 0x00 |

### 3.5.4 Node protection (NMT Node Guarding)

By using the node protection service, the NMT master node can detect the current state of each node.

- **Command**

Request: Master station (remote frame) –> slave station

| COB-ID | No data |
|---|---|
| 0x700 + Node-ID | |

Response: Slave station -> master station

| COB-ID | Byte0 (state value) |
|---|---|
| 0x700 + Node-ID | Bit 7: Triggering bit; Bits 0 to 6: State |

- Description

The most significant bit (MSB) bit 7 of Byte0 (state value) in the response command is the triggering bit, that is, the value of bit 7 is alternated between 0 and 1 each time when the slave station transmits a response frame to distinguish frames. Bits 0 to 6 indicate the state of the slave station. Table 3-5 describes the state values and their corresponding state.

Table 3-5 State values and their corresponding states

| State value (Byte0: Bits 0–6) | State |
|---|---|
| 0x00 | Initializing |
| 0x04 | Stopped |
| 0x05 | Operation |
| 0x7F | Pre-operational |

- Example

For example, the command for the master station to detect the state of slave station 3.

Master station (remote frame) -> slave station

| COB-ID | No data |
|---|---|
| 0x703 | / |

After receiving the node protection command transmitted by the master station, the slave station transmits the following command response to the master station.

| COB-ID | Byte0 (state value) |
|---|---|
| 0x703 | 0x85 |

In the command, bit 7 of Byte0 is 1, and the state value is 0×05, indicating that slave station 3 is in the operation state. If receiving another node protection command, the slave station transmits a command frame in which the state value is 0×05 to the master station, and the value of bit 7 is alternated to 0.

**3.5.5 Heartbeat packet (Heartbeat Producer)**

In some cases, the master station requires that a slave station automatically transmits a frame of heartbeat packets at an interval, so that it can learn the state of the slave station in real time. The interval parameter (data length: 16 bits; unit: ms) is defined in the object dictionary 0x1017. If the interval is set to 0, the slave station does not transmit heartbeat packets. For this CANopen communication card, the interval is set to 0 by default.

- Command

Slave station -> master station

| COB-ID | Byte0 |
|---|---|
| 0x700 + Node-ID | State value |

- Description

The heartbeat packets are in the same format with the node protection response frames. The difference between them is that no triggering bit alternation is performed for heartbeat packets (the triggering bit is always 0). Table 3-5 describes the state values.

- Example

For example, if slave station 3 is in the operation state and the interval parameter in 0x1017 is set to 100, slave station 3 transmits a frame of heartbeat packets every 100 ms.

| COB-ID | Byte0 |
|---|---|
| 0x703 | 0x05 |

SDOs can be used to disable heartbeat packets, transmitting 2B 17 10 00 00 00 00 00 (setting the interval to 0).

**Note:** On the communication card, node protection and heartbeat packets cannot be used simultaneously.

### 3.5.6 Start packet (NMT Boot-up)

After being initialized (booted up), the communication card transmits a start packet.

- Command

Slave station -> master station

| COB-ID | Byte0 |
|---|---|
| 0x700 +Node-ID | 0x00 |

- Example

For example, after being initialized, the communication card whose node ID is 3 transmits the following start packet.

| COB-ID | Byte0 |
|---|---|
| 0x703 | 0x00 |

### 3.5.7 Synchronous packet object (SYNC)

Generally, SYNC signals are transmitted by the CANopen master station cyclically. A SYNC signal does not contain any data and is used mainly to request PDO Tx of a slave station node of the synchronous transmission type. 0x1005 in the object dictionary defines COB-IDs of the objects that receive synchronous packets, and they are set to 0x80 in the CANopen pre-defined connection set. For PDO Tx, the transmission types of 1 to 240 indicate synchronous transmission.

- Command

Master station -> slave station

| COB-ID | No data |
|---|---|
| 0x80 | / |

### 3.5.8 Emergency packet object (EMCY)

This packet is transmitted when an internal error occurs on the communication card or VFD, or an error is deleted.

- Command

Slave station -> master station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x80 + Node-ID | Emergency error code | | Error register | VFD error code | | | | |
| | LSB | MSB | | bit7-0 | bit15-8 | bit23-16 | bit31-24 | bit39-32 |

- Description

An emergency error code is two bytes. Byte0 is the least significant byte (LSB), and Byte1 is the most significant byte (MSB). A VFD error code is five bytes. Byte3 is the LSB, and Byte7 is the MSB.

An emergency error code indicates the type of the current error, as described in Table 3-6. The error register stores the type of the current error. You can determine the error type indicated by the current emergency packet according to the value stored in the register. Table 3-7 describes the indication of the bits of the error register. For information about the VFD error codes, see the VFD operation manual. The function code P07.27 in Appendix B describes the error codes of Goodrive350 VFD.

Table 3-6 Emergency error codes

| Emergency error code (hex) | Code function description |
|---|---|
| 00xx | Error reset or no error |
| 10xx | Generic error |
| 20xx | Current |
| 21xx | Current error on the, device input side |
| 22xx | Current error inside the device |
| 23xx | Current error on the device output side |
| 30xx | Voltage error |
| 31xx | Mains voltage |
| 32xx | Voltage inside the device |
| 33xx | Output voltage |
| 40xx | Temperature |
| 41xx | Ambient temperature |
| 42xx | Device temperature |
| 50xx | Device hardware |
| 60xx | Device software |
| 61xx | Internal software |
| 62xx | User software |
| 63xx | Data set |

| Emergency error code (hex) | Code function description |
|---|---|
| 70xx | Additional modules |
| 80xx | Monitoring |
| 81xx | Communication error |
| 8110 | CAN overrun |
| 8120 | Error passive |
| 8130 | Life guard Error or heartbeat error |
| 8140 | Recovered from Bus-Off |
| 82xx | Protocol error |
| 8210 | PDO not processed due to length error |
| 8220 | Length exceeded |
| 90xx | External error |
| F0xx | Additional functions |
| FFxx | Device specific |

Table 3-7 Error register bits

| Error register bit | Error type |
|---|---|
| 0 | Generic error or no error |
| 1 | Current error |
| 2 | Voltage error |
| 3 | Temperature error |
| 4 | Communication error |
| 5 | Device description error |
| 6 | Reserved (=0) |
| 7 | Manufacturer-defined error |

- Example

For example, if the "inverter unit phase U protection (OUT1)" fault occurs on the Goodrive350 VFD whose node ID is 3, and the fault type is 1 (that is, the VFD error code is 1), the communication card transmits the following emergency packet.

| COB-ID | Emergency error code | | Error register | VFD error code | | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x83 | 0x00 | 0x30 | 0x04 | 0x01 | 0x00 | 0x00 | 0x00 | 0x00 |

As you can see in the command, the emergency error code is 0x3000, indicating a voltage error. The error register is 0x04, that is, the second bit is "1", indicating a voltage error. The device error code is 0x0000000001. See the Goodrive350 VFD operation manual, and you can find that the error code 1 indicates the "inverter unit phase U protection (OUT1)" fault.

After the fault is reset, the communication card transmits the following emergency packet to notify the master station that the slave station is no longer faulty.

| COB-ID | Emergency error code | | Error register | VFD error code | | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x83 | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 |

### 3.5.9 Service data object (SDO)

SDOs are mainly used to transmit non-time key data. By using SDOs, the master station can read data from and write data to the object dictionary of a device.

- Command

Request: master station -> slave station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x600+NodeID | Request code | Object index | | Subindex | Response data | | | |
| | | LSB | MSB | | bit7-0 | bit15-8 | bit23-16 | bit31-24 |

Response: Slave station -> master station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x580+NodeID | Respons e code | Object index | | Subindex | Response data | | | |
| | | LSB | MSB | | bit7-0 | bit15-8 | bit23-16 | bit31-24 |

- Description

An object index is two bytes. Byte1 is the LSB, and byte2 is the MSB. For information about the indexes and subindexes, see the object dictionary in the appendix. Request codes include request codes for reading and those for writing.

Request codes for writing vary according to the character length of items in the object dictionary, and the request code for reading are 0×40. See Table 3-8.

Response codes indicating successful reading vary according to the character length of items in the object dictionary, and the response code indicating successful writing are 0×60. The response codes indicating reading failure and writing failure are both 0x80. See Table 3-9.

Table 3-8 SDO request codes and requested data

| Request code type | Request code | Command description | Requested data | | | |
|---|---|---|---|---|---|---|
| | | | Byte4 | Byte5 | Byte6 | Byte7 |
| Write | 0x23 | Writes 4-byte data | bit7-0 | bit15-8 | bit23-16 | bit31-24 |
| | 0x2B | Writes 2-byte data | bit7-0 | bit15-8 | - | - |
| | 0x2F | Writes 1-byte data | bit7-0 | - | - | - |
| Read | 0x40 | Reads data | - | - | - | - |

Table 3-9 SDO response codes and response data

| Response code type | Response code | Command description | Response data | | | |
|---|---|---|---|---|---|---|
| | | | Byte4 | Byte5 | Byte6 | Byte7 |
| Read | 0x43 | Reads 4-byte data | bit7-0 | bit15-8 | bit23-16 | bit31-24 |
| | 0x4B | Reads 2-byte data | bit7-0 | bit15-8 | - | - |
| | 0x4F | Reads 1-byte data | bit7-0 | - | - | - |
| Write | 0x60 | Writing succeeds | - | - | - | - |
| Read/write | 0x80 | Reading/writing fails | Interruption error code | | | |
| | | | bit7-0 | bit15-8 | bit23-16 | bit31-24 |

**Note:** The symbol "-" in Table 3-8 and Table 3-9 indicates that the byte is reserved and provides no function.

Table 3-10 describes the interruption error codes.

Table 3-10 Interruption error codes

| Interruption code | Code function description |
|---|---|
| 0503 0000 | Triggering bit not alternated |
| 0504 0000 | SDO protocol times out |
| 0504 0001 | Invalid or unknown client/server |
| 0504 0002 | Invalid block size |
| 0504 0003 | Invalid sequence number |
| 0504 0004 | CRC error |
| 0504 0005 | Memory overflow |
| 0601 0000 | No access to the object |
| 0601 0001 | Attempts to read a write-only object |
| 0601 0002 | Attempts to write information to a read-only object |
| 0602 0000 | Object cannot be found in the object dictionary |
| 0604 0041 | Object cannot be mapped to PDO |
| 0604 0042 | Number and length of the object to be mapped exceeds the PDO length |
| 0604 0043 | Common parameter incompatibility |
| 0604 0047 | Common internal incompatibility of the device |
| 0606 0000 | Object access failure caused by hardware error |
| 0607 0010 | Data type not matched; service parameter length not matched |
| 0609 0011 | Subindex cannot be found in the object dictionary |
| 0609 0030 | Parameter value range exceeded |
| 0609 0031 | Written parameter value too large |

| Interruption code | Code function description |
|---|---|
| 0609 0032 | Written parameter value too small |
| 0609 0036 | Max. value less than Min. value |
| 0800 0000 | Common error |
| 0800 0020 | Data failed to be transmitted or stored in the application |
| 0800 0021 | Data failed to be transmitted or stored in the application due to device control |
| 0800 0022 | Data failed to be transmitted or stored in the application due to the current state of the device |
| 0800 0023 | Error occurs dynamically on the object dictionary or object dictionary cannot be found |

- Example

For example, slave station 3 reads data from and writes data to the object whose index is 0x1801 and subindex is 03. (The object whose index is 0x1801 and subindex is 03 indicates the disabled time of PDO2 Tx. For more information, see Chapter 4.)

Write operation example: To modify the disabled time of PDO2 Tx to 1000 ms, the master station transmits the following write operation command.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x2B | 0x01 | 0x18 | 0x03 | 0xe8 | 0x03 | 0x00 | 0x00 |

After receiving the command transmitted by the master station, the slave station transmits the following command response if the modification is successful.

| COB-ID | Response code | Object index | | Subindex | Response data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x583 | 0x60 | 0x01 | 0x18 | 0x03 | 0x00 | 0x00 | 0x00 | 0x00 |

Read operation example: To read the disabled time of PDO2 Tx, the master station transmits the following read operation command.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x40 | 0x01 | 0x18 | 0x03 | 0x00 | 0x00 | 0x00 | 0x00 |

After receiving the command transmitted by the master station, the slave station transmits the following command response if the current disabled time of PDO2 Tx is 1000 ms.

| COB-ID | Response code | Object index | | Subindex | Response data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x583 | 0x43 | 0x01 | 0x18 | 0x03 | 0xe8 | 0x03 | 0x00 | 0x00 |

Read/write error example: The master station transmits the following read operation command to read an object (whose index is 0x6000 and subindex is 0x00) that cannot be found.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x40 | 0x00 | 0x60 | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 |

The object cannot be found, and therefore the slave station transmits the following read/write error command response.

| COB-ID | Response code | Object index | | Subindex | Response data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x583 | 0x80 | 0x00 | 0x60 | 0x00 | 0x00 | 0x00 | 0x02 | 0x06 |

The error code in the response is 0x06020000, indicating that "Object cannot be found in the object dictionary".

## 3.6 Process data object (PDO)

The communication card provides four PDO Tx commands (whose indexes are 0x1800 to 0x1803) and four PDO Rx commands (whose indexes are 0x1400 to 0x1403). PDO Rx is a PDO command transmitted by the master station to a slave station, that is, it is a master station command. PDO Tx is a PDO command transmitted by a slave station to the master station.

The CW, SW, setting, and return value of each PDO of the communication card are all defined with a "manufacturer-defined object dictionary". In this way, the process data of a VFD can be monitored not only through PDOs but also through SDOs. For more information, see the next chapter. Each PDO command is labeled with "manufacturer-defined object dictionary" in the format of 0xXXXX.HH, where XXXX indicates an index, HH indicates a subindex, and both of them are hexadecimal.

### 3.6.1 Triggering mode of PDO Tx

Each PDO Tx is defined with a transmission type, disabled time, and event timer. The corresponding subindex of the transmission type is 0x02, that of the disabled time is 0x03, and that of the event timer is 0x05. Therefore, the object dictionary index corresponding to PDO2 Tx is 0x1801, and the subindex is 0x02. The same principle applies to other PDO Tx commands. For more information, see Appendix A.

Synchronous triggering: When the transmission type is set to 1 to 240, PDO Tx is synchronous transmission. For example, if you set the transmission type of PDO2 Tx to *n* (1≤*n*≤240), a slave station transmits one PDO2 Tx command every time after it receives *n* synchronous packet objects. The same principle applies to other PDO Tx commands.

Asynchronous triggering (254): When the value of the event timer is not zero, a slave station transmits PDO Tx commands periodically. For example, if the event timer of PDO2 Tx is set to 200, the slave station transmits a PDO2 Tx command at the interval of 200 ms. When the value of the event timer is zero, the slave station transmits a PDO Tx command once the corresponding PDO Tx data changes, and the transmission interval is subject to the disabled time. A PDO Tx packet can be transmitted only once in the disabled time, which effectively reduces the load of the bus. When the disabled time is set to a period shorter than 50 ms, 50 ms is used as the disabled time.

Asynchronous triggering (255): When the value of the event timer is not zero, a slave station transmits PDO Tx commands periodically. For example, if the event timer of PDO2 Tx is set to 200, the slave station transmits a PDO2 Tx command at the interval of 200 ms. When the value of the event timer is zero, the slave station transmits a PDO Tx command once a corresponding PDO Rx command is received. For example, after receiving a PDO2 Rx command, the slave station transmits a PDO2 Tx command.

Table 3-11 Triggering modes supported by the communication card

| Triggering mode | Transmission type (decimal) | Event triggering | PDO1 TX | PDO2 TX | PDO3 TX | PDO4 TX |
|---|---|---|---|---|---|---|
| Synchronous | 1–240 | / | Non-supported | Supported | Supported | Supported |
| Asynchronous | 254 | Event timer | Non-supported | Supported | Supported | Supported |
| | | Disabled time | Non-supported | Supported | Supported | Supported |
| | 255 | Event timer=0 | Supported | Supported | Supported | Supported |
| | | Event timer=0 | Non-supported | Supported | Supported | Supported |

Table 3-12 Default PDO Tx settings of the communication card

| | PDO1 TX | PDO2 TX | PDO3 TX | PDO4 TX |
|---|---|---|---|---|
| **Transmission type** | 255 | 254 | 254 | 254 |
| **Event timer (ms)** | 0 | 0 | 0 | 0 |
| **Disabled time (ms)** | 500 | 500 | 500 | 500 |

For how to set the triggering type of PDO Tx, see the description of SDO commands.

### 3.6.2 PDO1

PDO1 is used to read and write parameters of the VFD. The function of PDO1 is similar to that of an SDO. SDOs are used to read and write objects of an object dictionary, and PDO1 is used

to read and write parameters of the VFD.

**Note:** PDO1 Tx support only the transmission type of asynchronous transmission 255. Do not set it to other transmission types, and do not try to set the event timer to periodically transmits PDO1 Tx to the master station.

#### 3.6.2.1 PDO1 Rx

- Command

Request: Master station –> slave station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 |
|---|---|---|---|---|---|---|
| 0x200+NODE-ID | Request code | | Parameter address | | Requested data | |
| | 0x2100.00 | | 0x2100.01 | | 0x2100.02 | |

- Description

A request code is two bytes. Byte0 is the LSB, and Byte1 is the MSB. The manufacturer defines the index 0x2100 and subindex 0x00 for the request codes. Table 3-13 describes the functions of the request codes.

Table 3-13 Request codes

| Request code | Function |
|---|---|
| 0 | No task |
| 1 | Reading the value of a parameter |
| 2 | Modifying a parameter value [modifying the value only on RAM] |
| 4 | Modifying a parameter value [modifying the value only on both RAM and EEPROM] (reserved) |

A parameter address is two bytes. Byte2 is the LSB, and Byte3 is the MSB. It indicates the address of the parameter to be read or modified.

Goodrive350 series VFD function code address representation rules: The MSB is the hexadecimal form of the number before the dot mark, and LSB is that of the number behind the dot mark. Take P10.01 as an example, the number before the dot mark is 10, that is, the MSB of the parameter address is 0×0A; and the number behind the dot mark is 01, that is, the LSB is 0×01. Therefore, the function code address is 0×0A01.

Table 3-14 Goodrive350 series VFD parameter addresses

| Function code | Name | Detailed parameter description | Default value | Modify |
|---|---|---|---|---|
| P10.00 | Simple PLC mode | 0: Stops after running once<br>1: Keeps running in the final value after running once<br>2: Cyclic running | 0 | ○ |
| P10.01 | Simple PLC memory selection | 0: Not saving data at power outage<br>1: Saving data at power outage | 0 | ○ |

VFD parameter address representation rules: You can see the function code in the function parameter list in the VFD operation manual. The hexadecimal form of the value corresponding to the function code is the parameter address. For example, the value corresponding to the function code P13.14 is 1314, and therefore the parameter address of the function code is 0×522 (that is, 1314 in the decimal form).

A piece of requested data is two bytes. Byte4 is the LSB, and Byte5 is the MSB. It indicates the data to be modified. When the command is transmitted for reading data, the requested data is not used.

**Note:** The data domain of PDO1 Rx must be six bytes. Otherwise, the communication card reports an emergency packet.

#### 3.6.2.2 PDO1 Tx

- Command

Response: Slave station -> master station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x180+NODEID | Response code | | Error code | | Response data | | 0x00 | 0x00 |
| | 0x2000.00 | | 0x2000.01 | | 0x2000.02 | | - | - |

- Description

Byte6 and Byte7 are reserved and both are 0x00.

A response code is two bytes. Byte0 is the LSB, and Byte1 is the MSB. Table 3-15 describes the functions of the response codes.

Table 3-15 Response codes

| Response code | Function |
|---|---|
| 0 | No response |
| 1 | Reading or writing succeeds |
| 3 | A reading or writing error occurs. Table 3-16 describes the error codes. |

A piece of response data is four bytes. Byte4 is the LSB, and Byte7 is the MSB. When a write command is responded, the response data is the data to be modified; and when a read command is responded, the response data is the data to be read.

An error code is two bytes. Byte2 is the LSB, and Byte3 is the MSB. Error codes are valid only when the response code is 3. An error code indicates the reason why it fails to respond to PDO1 Rx. Table 3-16 describes the definitions of the error codes.

Table 3-16 Error codes

| Code | Name | Definition |
|---|---|---|
| 00H | No error | / |
| 01H | Invalid command | The operation corresponding to the request code is not allowed to be executed. The possible causes are as follows:<br>• The function code is applicable only on new devices and is not implemented on this device.<br>• The slave station is in the faulty state when processing this request. |
| 02H | Invalid data address | For a slave device, the data address in the request of the master station is not allowed. In particular, the combination of the register address and the number of the to-be-transmitted bytes is invalid. |
| 03H | Invalid data value | The received data domain contains a value that is not allowed. The value indicates the error of the remaining structure in the combined request.<br>**Note:** It does not mean that the data item submitted for storage in the register includes a value unexpected by the program. |
| 04H | Operation failure | The parameter is set to an invalid value in the write operation. For example, a function input terminal cannot be set repeatedly. |
| 05H | Password error | The password entered in the password verification address is different from that set by the user. |
| 06H | Data frame error | The length of the data frame transmitted by the upper computer is incorrect, or in the RTU format, the value of the CRC check bit is inconsistent with the CRC value calculated by the lower computer. |
| 07H | Parameter read-only | The parameter to be modified in the write operation of the master station is a read-only parameter. |

| Code | Name | Definition |
|---|---|---|
| 08H | Parameter cannot be modified in running | The parameter to be modified in the write operation of the master station cannot be modified during the running of the VFD. |
| 09H | Password protection | A user password is set, and the master station does not provide the password to unlock the system when performing a read or write operation. The error of system locked is reported. |

- Example of PDO1

The VFD is a Goodrive350 series VFD, and the slave station address is 3. Assume that you want to set the function code P15.13 of the VFD to 1.

Command analysis: The parameter address of P15.13 is 0×0F0D. According to the protocol, the request code of PDO1 Rx is 0×02, the parameter address is 0x0F0D, and the requested data is 0x01, and therefore PDO1 Rx transmitted by the master station is as follows.

| COB-ID | Request code | | Parameter address | | Requested data | |
|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 |
| 0x203 | 0x02 | 0x00 | 0x0D | 0x0F | 0x01 | 0x00 |

If the VFD parameter is successfully modified, the following PDO1 Tx command is returned.

| COB-ID | Response code | | Error code | | Response data | | - | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x183 | 0x01 | 0x00 | 0x00 | 0x00 | 0x01 | 0x00 | 0x00 | 0x00 |

### 3.6.3 PDO2 Rx

PDO2 Rx is used to modify CWs and real-time process data (setting 1, setting 2, and setting 3) of a VFD. A CW is used to control the start and stop of a VFD, and settings are used to control the real-time running values of the VFD, such as set frequency.

- Command

Master station -> slave station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x300+NODEID | CW | | Setting 1 | | Setting 2 | | Setting 3 | |
| | 0x2101.00 | | 0x2100.03 | | 0x2100.04 | | 0x2100.05 | |

- Description

A CW is two bytes. Byte0 is the LSB, and Byte1 is the MSB. Table 3-17 describes Goodrive350 series VFD CWs.

Table 3-17 Goodrive350 series VFD CWs

| Bit | Name | Value | Description |
|---|---|---|---|
| 0–7 | Communication-based control command | 1 | Forward running |
| | | 2 | Reverse running |
| | | 3 | Forward jogging |
| | | 4 | Reverse jogging |
| | | 5 | Stop |
| | | 6 | Coast to stop |
| | | 7 | Fault reset |
| | | 8 | Stop jogging |
| | | 9 | Emergency stop |
| 8 | Enable write | 1 | Enable writing (mainly through PKW1 to PKW4) |
| 9–10 | Motor group setting | 00 | Motor 1 |
| | | 01 | Motor 2 |
| 11 | Control mode switching | 1 | Enable torque/speed control switching |
| | | 0 | Disable switching |
| 12 | Reset power consumption to zero | 1 | Enable |
| | | 0 | Disable |
| 13 | Pre-excitation | 1 | Enable |
| | | 0 | Disable |
| 14 | DC braking | 1 | Enable |
| | | 0 | Disable |
| 15 | Heartbeat reference | 1 | Enable |
| | | 0 | Disable |

The function of each setting can be set through the corresponding function code of the VFD. The setting method is the same as that for "received PZD" in PROFIBUS communication. For details, see the VFD operation manual. Setting 1, setting 2, and setting 3 correspond to received PZD2, received PZD3, and received PZD4, respectively. To set the function of setting 1 to "Set frequency", you need only to set "Received PZD2" to "1: Set frequency". The same principle applies to other settings. When multiple settings are enabled, the failure to set one setting (for example, the set value exceeds the setting range) does not affect the setting of other settings.

- Example

Assume that the VFD is a Goodrive350 series VFD, the slave station address is 3, you control the running of the VFD through CANopen communication, and you want to set the running

frequency to 50 Hz through CANopen communication.

Command analysis: You need to set the VFD start mode and frequency reference mode to CANopen communication (P00.01=2, P00.02=1, P00.06=9) first. In this example, use Setting 2 to set the running frequency (P15.03=1, that is, set Received PZD3 to "1: Set frequency").

When a CW is 0×01, it indicates that the VFD is to be run. To set the frequency to 50 Hz, you need to set Setting 2 to 5000, that is, 0x1388.

The PDO2 Rx command transmitted by the master station is as follows.

| COB-ID | CW | | Setting 1 | | Setting 2 | | Setting 3 | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x303 | 0x01 | 0x00 | 0x00 | 0x00 | 0x88 | 0x13 | 0x00 | 0x00 |

**3.6.4 PDO2 Tx**

PDO2 Tx is a command transmitted by a VFD to the master station. It contains a SW and real-time process data (Returned value 1, returned value 2, and returned value 3). A SW is used to notify of the state of the VFD, and returned values are used to transmit the real-time running values of VFD, such as running frequency.

The default transmission type of PDO2 Tx is 254, and therefore PDO2 Tx is transmitted once data corresponding to a SW or returned value changes.

- Command

Slave station -> master station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x280+NODEID | SW | | Returned value 1 | | Returned value 2 | | Returned value 3 | |
| | 0x2001.00 | | 0x2000.03 | | 0x2000.04 | | 0x2000.05 | |

- Description

A SW is two bytes. Byte0 is the LSB, and Byte1 is the MSB. Table 3-18 describes the definitions of the Goodrive350 series VFD SWs. For VFD of other series, see the corresponding VFD operation manual.

Table 3-18 Goodrive350 series VFD SWs

| Bit | Name | Value | Description |
|---|---|---|---|
| 0–7 | Running state | 1 | In forward running |
| | | 2 | In reverse running |
| | | 3 | Stopped |
| | | 4 | Faulty |
| | | 5 | POFF |
| 8 | Bus voltage established | 1 | Ready to run |

| Bit | Name | Value | Description |
|---|---|---|---|
| | | 0 | Not ready to run |
| 9–10 | Motor group feedback | 0 | Motor 1 |
| | | 1 | Motor 2 |
| 11 | Motor type feedback | 1 | Synchronous motor |
| | | 0 | Asynchronous motor |
| 12 | Overload pre-alarm feedback | 1 | Overload pre-alarm generated |
| | | 0 | No overload pre-alarm generated |
| 13–14 | Run/stop mode | 0 | Keypad-based control |
| | | 1 | Terminal-based control |
| | | 2 | Communication-based control |
| | | 3 | Reserved |
| 15 | Heartbeat feedback | 1 | Heartbeat feedback |
| | | 0 | No heartbeat feedback |

The function of each returned value can be set through the corresponding function code of the VFD. The setting method is the same as that for "Transmitted PZD" in PROFIBUS communication. For details, see the VFD operation manual. Returned value 1, returned value 2, and returned value 3 correspond to transmitted PZD2, transmitted PZD3, and transmitted PZD4, respectively. To set the function of returned value 1 to "Running frequency", you need only to set "Transmitted PZD2" to "1: Running frequency". The same principle applies to other returned values. Multiple returned values can be enabled simultaneously.

- Example

Assume that the VFD is a Goodrive350 series VFD, the slave station address is 3, the VFD is running, and the running frequency is 50.00 Hz. Returned value 1 is set to "Running frequency", returned value 2 is set to "Output voltage", and returned value 3 is set to no function.

Command analysis: You need to set returned value 1 to the running frequency of the VFD (P15.13=1), returned value 2 to the output voltage of the VFD (P15.14=4), and returned value 3 to invalid (P15.15=0) first.

The VFD is running and the bus voltage has been established, and therefore the SW is 0x0101. The running frequency is 50.00 Hz, and therefore returned value 1 is 5000, that is, 0x1388. If the output voltage is 380 V, returned value 2 is 0x017C.

The PDO2 Tx command transmitted by the VFD is as follows.

| COB-ID | SW | | Returned value 1 | | Returned value 2 | | Returned value 3 | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x283 | 0x01 | 0x01 | 0x88 | 0x13 | 0x7C | 0x01 | 0x00 | 0x00 |

### 3.6.5 PDO3 Rx and PDO4 Rx

PDO3 Rx and PDO4 Rx are used to modify the real-time process data of a VFD, such as set frequency.

- PDO3 Rx command

Master station -> slave station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x400+NODEID | Setting 4 | | Setting 5 | | Setting 6 | | Setting 7 | |
| | 0x2100.06 | | 0x2100.07 | | 0x2100.08 | | 0x2100.09 | |

- PDO4 Rx command

Master station -> slave station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x500+NODEID | Setting 8 | | Setting 9 | | Setting 10 | | Setting 11 | |
| | 0x2100.0a | | 0x2100.0b | | 0x2100.0c | | 0x2100.0d | |

- Description

The application methods for PDO3 Rx and PDO4 Rx are the same as that for PDO2 Rx. For the relationship between the settings and PZD in PROFIBUS communication, see Table 3-19.

### 3.6.6 PDO3 Tx and PDO4 Tx

PDO3 Tx and PDO4 Tx are used by the VFD to transmit real-time process data to the master station, such as running frequency.

The default transmission type of PDO3 Tx and PDO4 Tx is 254, and therefore PDO3 Tx or PDO4 Tx is transmitted once data corresponding to a returned value in the same command changes.

- PDO3 Tx command

Slave station -> master station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x380+NODEID | Returned value 4 | | Returned value 5 | | Returned value 6 | | Returned value 7 | |
| | 0x2000.06 | | 0x2000.07 | | 0x2000.08 | | 0x2000.09 | |

- PDO4 Tx command

Slave station -> master station

| COB-ID | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
|---|---|---|---|---|---|---|---|---|
| 0x480+NODEID | Returned value 8 | | Returned value 9 | | Returned value 10 | | Returned value 11 | |
| | 0x2000.0a | | 0x2000.0b | | 0x2000.0c | | 0x2000.0d | |

- Description

The application methods for PDO3 Tx and PDO4 Tx are the same as that for PDO2 Tx. For the relationship between the returned values and PZD in PROFIBUS communication, see

Table 3-20.

## 3.7 Monitoring process data through SDO commands

The communication can use SDOs as well as PDOs to monitor the process data of a VFD. You can select a monitoring mode as required. You can monitor the VFD by using SDOs to read the manufacturer-defined object dictionary.

For the definition and application of the CWs, SWs, settings, and returned values in the manufacturer-defined object dictionary, see the PDO description section. For application of SDOs, see the SDO description section. Do not try to use SDOs to read and write VFD parameters.

Table 3-19 and

Table 3-20 describe the manufacturer-defined object dictionary.

Table 3-19 Objects with the control function in the manufacturer-defined object dictionary

| Index (hexadecimal) | Subindex (hexadecimal) | Function | Access permission | Data length | Corresponding to |
|---|---|---|---|---|---|
| 2100 | 0 | Request code (do not use it) | RW | 2 bytes | / |
| | 1 | Parameter address (do not use it) | RW | 2 bytes | / |
| | 2 | Requested data (do not use it) | RW | 2 bytes | / |
| | 3 | Setting 1 | RW | 2 bytes | Received PZD2 |
| | 4 | Setting 2 | RW | 2 bytes | Received PZD3 |
| | 5 | Setting 3 | RW | 2 bytes | Received PZD4 |
| | 6 | Setting 4 | RW | 2 bytes | Received PZD5 |
| | 7 | Setting 5 | RW | 2 bytes | Received PZD6 |
| | 8 | Setting 6 | RW | 2 bytes | Received PZD7 |
| | 9 | Setting 7 | RW | 2 bytes | Received PZD8 |
| | A | Setting 8 | RW | 2 bytes | Received PZD9 |

| Index (hexadecimal) | Subindex (hexadecimal) | Function | Access permission | Data length | Corresponding to |
|---|---|---|---|---|---|
| | B | Setting 9 | RW | 2 bytes | Received PZD10 |
| | C | Setting 10 | RW | 2 bytes | Received PZD11 |
| | D | Setting 11 | RW | 2 bytes | Received PZD12 |
| | E | Reserved | RW | 2 bytes | / |
| | F | Reserved | RW | 2 bytes | / |
| 2101 | 0 | CW | RW | 2 bytes | / |

Table 3-20 Objects with the monitoring function in the manufacturer-defined object dictionary

| Index (hexadecimal) | Subindex (hexadecimal) | Function | Access permission | Data length | Corresponding to |
|---|---|---|---|---|---|
| 2000 | 0 | Response code (do not use it) | RO | 2 bytes | / |
| | 1 | Error code (do not use it) | RO | 2 bytes | / |
| | 2 | Response data (do not use it) | RO | 2 bytes | / |
| | 3 | Returned value 1 | RO | 2 bytes | Transmitted PZD2 |
| | 4 | Returned value 2 | RO | 2 bytes | Transmitted PZD3 |
| | 5 | Returned value 3 | RO | 2 bytes | Transmitted PZD4 |
| | 6 | Returned value 4 | RO | 2 bytes | Transmitted PZD5 |
| | 7 | Returned value 5 | RO | 2 bytes | Transmitted PZD6 |
| | 8 | Returned value 6 | RO | 2 bytes | Transmitted PZD7 |
| | 9 | Returned value 7 | RO | 2 bytes | Transmitted PZD8 |
| | A | Returned value 8 | RO | 2 bytes | Transmitted PZD9 |
| | B | Returned value 9 | RO | 2 bytes | Transmitted PZD10 |

| Index (hexadecimal) | Subindex (hexadecimal) | Function | Access permission | Data length | Corresponding to |
|---|---|---|---|---|---|
| | C | Returned value 10 | RO | 2 bytes | Transmitted PZD11 |
| | D | Returned value 11 | RO | 2 bytes | Transmitted PZD12 |
| | E | Reserved | RO | 2 bytes | / |
| | F | Reserved | RO | 2 bytes | / |
| 2001 | 0 | SW | RO | 2 bytes | / |

- Examples

Example 1: To instruct the VFD whose address is 3 to run forwardly, the master station transmits the following SDO command.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x2B | 0x01 | 0x21 | 0x00 | 0x01 | 0x00 | 0x00 | 0x00 |

Example 2: Assume that the address of the VFD slave station is 3, and the function of setting 1 is defined as "Set frequency". To set the frequency to 50.00 Hz (that is, setting 1=0x1388), the master station transmits the following SDO command.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x2B | 0x00 | 0x21 | 0x03 | 0x88 | 0x13 | 0x00 | 0x00 |

Example 3: To read the running state of the VFD whose address is 3, the master station transmits the following SDO command.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x40 | 0x01 | 0x20 | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 |

If the VFD is running forward, the following SDO command is returned to the master station.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x583 | 0x4B | 0x01 | 0x20 | 0x00 | 0x01 | 0x01 | 0x00 | 0x00 |

Example 4: Assume that the address of the VFD slave station is 3, and the function of setting 1 is defined as "Set frequency". To set the frequency to 50.00 Hz (that is, setting 1=0x1388), the master station transmits the following SDO command.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x603 | 0x40 | 0x00 | 0x20 | 0x03 | 0x00 | 0x00 | 0x00 | 0x00 |

If the running frequency of the VFD is 50.00 Hz, the following SDO command is returned to the master station.

| COB-ID | Request code | Object index | | Subindex | Requested data | | | |
|---|---|---|---|---|---|---|---|---|
| | Byte0 | Byte1 | Byte2 | Byte3 | Byte4 | Byte5 | Byte6 | Byte7 |
| 0x583 | 0x4B | 0x00 | 0x20 | 0x03 | 0x88 | 0x13 | 0x00 | 0x00 |

## 3.8 Baud rate and communication address setting

### 3.8.1 Baud rate setting

After setting the CANopen baud rate and communication address, you need to restart the VFD to enable the settings to take effect.

The CANopen baud rate is set through the corresponding VFD function parameter. For description of function code addresses, see the VFD operation manual. Table 3-21 describes the values of the function parameter and their corresponding baud rates.

Table 3-21 Baud rate setting

| Function parameter value | Baud rate (bit/s) |
|---|---|
| 0 | 1000 k |
| 1 | 800 k |
| 2 | 500 k |
| 3 | 250 k |
| 4 | 125 k |
| 5 | 100 k |
| 6 | 50 k |
| 7 | 20 k |

### 3.8.2 Communication address setting

The CANopen communication address is set through the function parameter P15.01.

### 3.8.3 Function codes related to transmitted and received PZD

Table 3-22 Received PZD

| Function code | Word | Value range | Default value |
|---|---|---|---|
| P15.02 | Received PZD2 | 0–31<br>0: Invalid<br>1: Set frequency (0–Fmax, unit: 0.01 Hz)<br>2: PID reference (0–1000, in which 1000 corresponds to 100.0%)<br>3: PID feedback (0–1000, in which 1000 corresponds to 100.0%)<br>4: Torque setting (-3000–+3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>5: Setting of the upper limit of forward running frequency (0–Fmax, unit: 0.01 Hz)<br>6: Setting of the upper limit of reverse running frequency (0–Fmax, unit: 0.01 Hz)<br>7: Upper limit of the electromotive torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>8: Upper limit of the brake torque (0–2000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>9: Virtual input terminal command, 0x000–0x3FF (corresponding to S8, S7, S6, S5, HDIB, HDIA, S4, S3, S2, and S1 in sequence)<br>10: Virtual output terminal command, 0x00–0x0F (corresponding to RO2, RO1, HDO, and Y1 in sequence)<br>11: Voltage setting (for V/F separation) (0–1000, in which 1000 corresponds to 100.0% of the rated voltage of the motor)<br>12: AO output setting 1 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>13: AO output setting 2 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>14: MSB of position reference (signed number)<br>15: LSB of position reference (unsigned number)<br>16: MSB of position feedback (signed number)<br>17: LSB of position feedback (unsigned number)<br>18: Position feedback setting flag (position feedback can | 0 |
| P15.03 | Received PZD3 | | 0 |
| P15.04 | Received PZD4 | | 0 |
| P15.05 | Received PZD5 | | 0 |
| P15.06 | Received PZD6 | | 0 |
| P15.07 | Received PZD7 | | 0 |
| P15.08 | Received PZD8 | | 0 |
| P15.09 | Received PZD9 | | 0 |
| P15.10 | Received PZD10 | | 0 |
| P15.11 | Received PZD11 | | 0 |
| P15.12 | Received PZD12 | | 0 |

| Function code | Word | Value range | Default value |
|---|---|---|---|
| | | be set only after this flag is set to 1 and then to 0)<br>19: Function parameter mapping (PZD2–PZD12 correspond to P14.49–P14.59)<br>20–31: Reserved | |

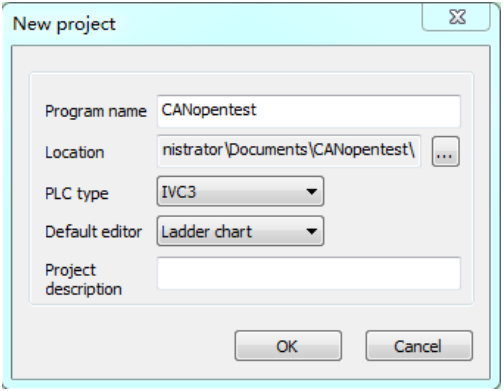
Table 3-23 Transmitted PZD

| Function code | Word | Value range | Default value |
|---|---|---|---|
| P15.13 | Transmitted PZD2 | 0–31<br>0: Invalid<br>1: Running frequency (×100, Hz)<br>2: Set frequency (×100, Hz)<br>3: Bus voltage (×10, V)<br>4: Output voltage (×1, V)<br>5: Output current (×10, A)<br>6: Actual output torque (×10, %)<br>7: Actual output power (×10, %)<br>8: Rotating speed of the running (×1, RPM)<br>9: Linear speed of the running (×1, m/s)<br>10: Ramp frequency reference<br>11: Fault code<br>12: AI1 value (×100, V)<br>13: AI2 value (×100, V)<br>14: AI3 value (×100, V)<br>15: HDIA frequency (×100, kHz)<br>16: Terminal input state<br>17: Terminal output state<br>18: PID reference (×100, %)<br>19: PID feedback (×100, %)<br>20: Rated torque of the motor<br>21: MSB of position reference (signed number)<br>22: LSB of position reference (unsigned number)<br>23: MSB of position feedback (signed number)<br>24: LSB of position feedback (unsigned | 0 |
| P15.14 | Transmitted PZD3 | | 0 |
| P15.15 | Transmitted PZD4 | | 0 |
| P15.16 | Transmitted PZD5 | | 0 |
| P15.17 | Transmitted PZD6 | | 0 |
| P15.18 | Transmitted PZD7 | | 0 |
| P15.19 | Transmitted PZD8 | | 0 |
| P15.20 | Transmitted PZD9 | | 0 |
| P15.21 | Transmitted PZD10 | | 0 |
| P15.22 | Transmitted PZD11 | | 0 |
| P15.23 | Transmitted PZD12 | | 0 |

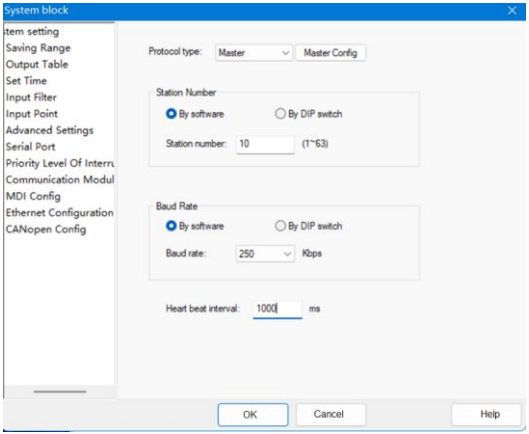
| Function code | Word | Value range | Default value |
|---|---|---|---|
| | | number)<br>25: State word<br>26: HDIB frequency value (×100, kHz)<br>27: High-order bit of PG card pulse feedback<br>28: Low-order bit of PG card pulse feedback<br>29: High-order bit of PG card pulse reference<br>30: Low-order bit of PG card pulse reference<br>31: Function parameter mapping (PZD2–PZD12 correspond to P14.60–P14.70) | |

## 3.9 Example of communication between CANopen and IVC3

Step 1 Create a project. Open INVT small PLC programming software Auto Station, choose **File** > **New project** and then fill in a program name, location, PLC type, and other required information. The interface is shown as follows.

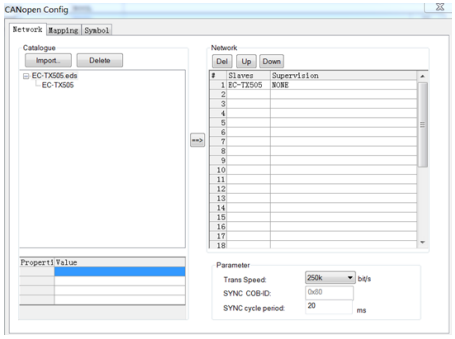



Step 2 Complete the CANopen configuration. Choose **Project manager** > **System block** > **CANopen configuration** to enter PLC master station setting. The interface is shown as follows.



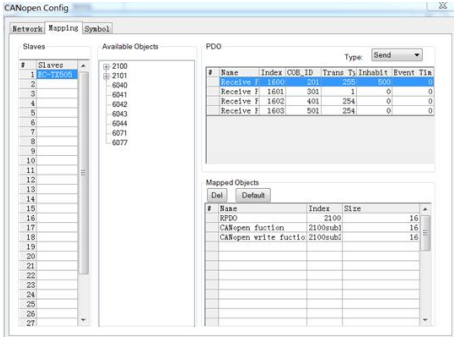

PLC station number and baud rate can be set through the software and dial-up. In this example, PLC station number and baud rate are set to 10 and 250K respectively through the software (by default).

Step 3 Import the EDS file for the slave node. Choose **Master configuration** > **Import** to import the EDS file **EC-TX505.eds** of GD350 series high performance vector VFD, select slave station number, set transmission speed and interval time of synchronous messages, and other information. The interface is shown as follows.

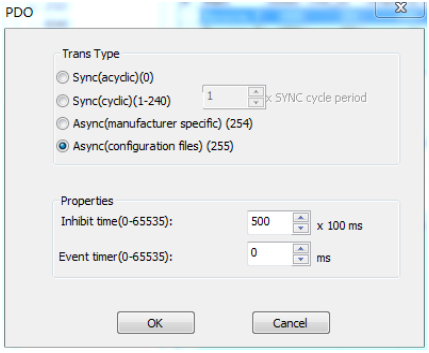


In this example, the slave number is 1, baud rate is 250K, and synchronization cycle period is 20ms.

Step 4 Configure the slave PDO data. Each slave station has four received PDOs and four transmitted PDOs. Since each PDO has multiple transmission modes, you can configure the response data and transmission modes according to the actual communication situation. Take transmitted PDO 1, transmitted PDO 2, received PDO 1 and received PDO 2 for example. The interface is shown as follows.



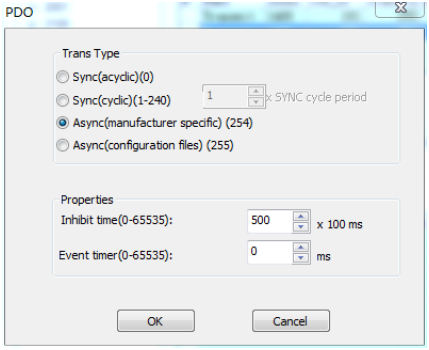

Double click one PDO to set the transmission mode, such as **sync** (mode 1–240) and **asyn** (mode 254 and 255). The interface is shown as follows.



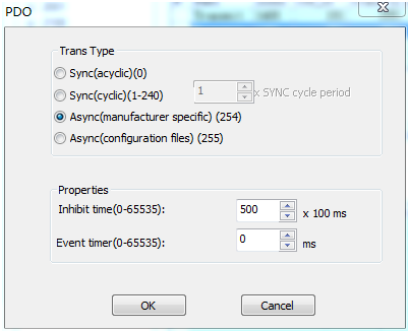

Refer to the description of CANopen communication of GD350 VFD, transmitted PDO

1 only supports 255 transmission mode, and does not support event timer mode. Therefore, received PDO 1 mode is configured as asynchronous 255 mode, and event timer is configured as 0ms, inhibition time is configured as 50ms, that is, transmitted PDO 1 message is sent at most once within 50ms.

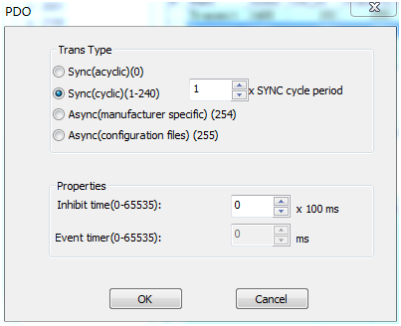
Transmitted PDO 2 supports all transmission modes. Generally, it is configured as 254 mode with an appropriate inhibition time. Transmitted PDO 2 message is sent upon data change, but it can be only sent once in each inhibition time so as to use bus resources reasonably. In the following interface, transmission mode is configured as 254, and inhibition time is configured as 50ms.



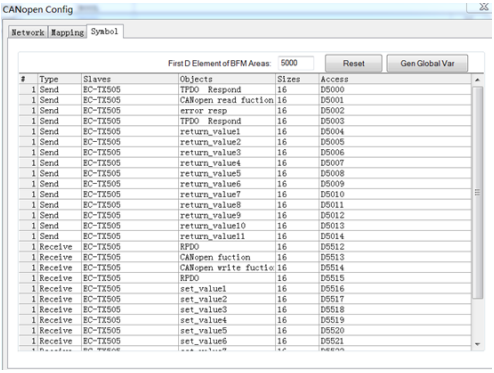

Received PDOs support all transmission modes. Generally, received PDO 1 is configured as 254 mode with an appropriate inhibition time while received PDO 2–4 are configured as synchronous mode. Different synchronous modes are configured according to real-time requirements of communication data. In the following interface, transmission mode of received PDO 1 is configured as 254, and inhibition time is configured as 50ms.




In the following interface, transmission mode of received PDO 2 is configured as 1, that is, the master transmits received PDO 2 once per sync cycle period. The sync cycle period is configured as 20ms, that is, the master transmits received PDO 2 once every 20ms.




Step 5 Perform symbol mapping. Map the configuration data to the internal storage area of the PLC. Choose **Symbol** > **Reset** > **Generate global variables**. The interface is shown as follows.




Step 6 Set VFD function parameters. The parameters are set as follows.

| Function code | Setting | Description |
|---|---|---|
| P00.01 | 2 | Start/stop through communication |
| P00.02 | 1 | CANopen communication mode |
| P00.06 | 9 | Set frequency through CANopen communication |
| P15.01 | 1 | Communication node number |
| P15.02 | 1 | Set frequency |
| P15.13 | 1 | Running frequency |
| P15.14 | 3 | Bus voltage |
| P15.15 | 4 | Output voltage |
| P15.16 | 5 | Output current |
| P15.27 | 3 | Communication baud rate 250kbps |

The configuration is completed. The data variables corresponding to transmitted PDO 1 are D5000–D5002, data variables corresponding to transmitted PDO 2 are D5003–D5006, data variables corresponding to received PDO 1 are D5512–D5514, and data variables corresponding to received PDO 2 are D5515–D5518.

Of which, D5000 is a request code for reading and writing, D5001 is parameter address,

D5002 is request data. Set D5000=1, indicating a request for reading the value of a parameter. Set D5001=11, namely, P00.11 (Acceleration time) address, indicating that received PDO 1 is to read the acceleration time of slave 1.

Return data D5512=1, indicating that the parameter is read successfully. D5514=400, indicating P00.11 is set to 40.0.

Set D5515=1, indicating that the VFD starts in the forward direction. Set D5516=264, indicating that communication frequency of the VFD is set to 2.64Hz.

Transmitted PDO 2 returns the running state and data regularly, in which the state word is D5003=16#4101 (heartbeat feedback, ready to run, VFD is in forward running), D5004=264 (running frequency 2.64Hz), D5005=5793 (bus voltage 579.3V), D5006=18 (output Voltage 18V), and D5007=0 (output current 0.0A). The interface is shown as follows.

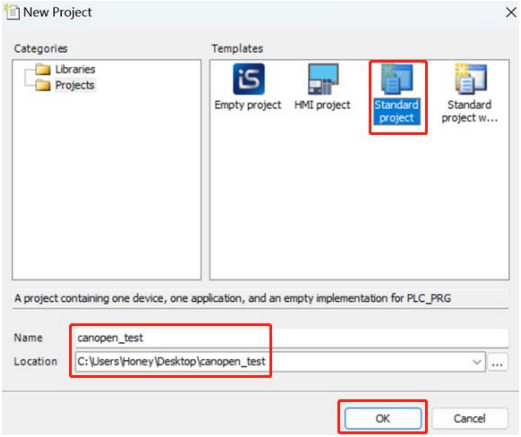
| | Element Name | data type | display forma | current value | new value |
|---|---|---|---|---|---|
| 1 | D5512 | WORD | Decimal | | 1 |
| 2 | D5513 | WORD | Decimal | | 11 |
| 3 | D5514 | WORD | Decimal | | 5 |
| 4 | D5515 | WORD | Decimal | | 1 |
| 5 | D5516 | WORD | Decimal | | 264 |
| 6 | D5000 | WORD | Decimal | | |
| 7 | D5001 | WORD | Decimal | | |
| 8 | D5002 | WORD | Decimal | | |
| 9 | D5003 | WORD | Hexadecimal | | |
| 10 | D5004 | WORD | Decimal | | |
| 11 | D5005 | WORD | Decimal | | |
| 12 | D5006 | WORD | Decimal | | |
| 13 | D5007 | WORD | Decimal | | |

## 3.10 Example of communication between CANopen and AX70

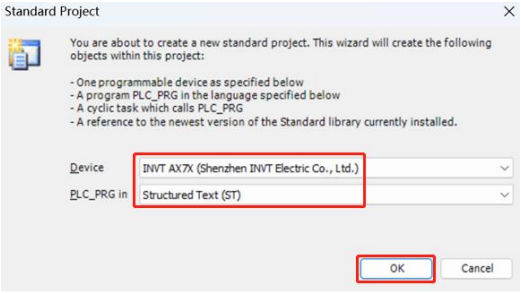
1. Set parameters of the VFD.

| Function code | Setting value | Description |
|---|---|---|
| P00.01 | 2 | Start/stop through communication |
| P00.02 | 1 | CANopen communication mode |
| P00.06 | 9 | Set frequency through CANopen communication |
| P15.01 | 1 | Communication node number |
| P15.02 | 1 | Set frequency |
| P15.13 | 1 | Running frequency |
| P15.14 | 3 | Bus voltage |
| P15.15 | 4 | Output voltage |
| P15.16 | 5 | Output current |
| P15.27 | 3 | Communication baud rate 250kbps |

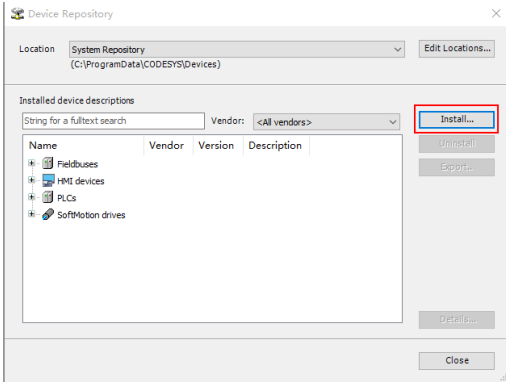
2. Open CODESYS V3.5 SP15 Patch 1, click **New project**, select **Templates**, and fill in **Name** and **Location**.



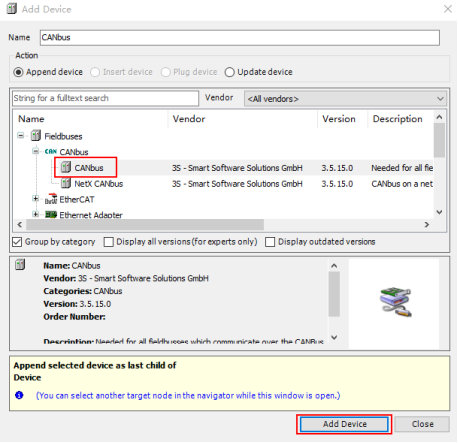

3. Select the device and programming language.



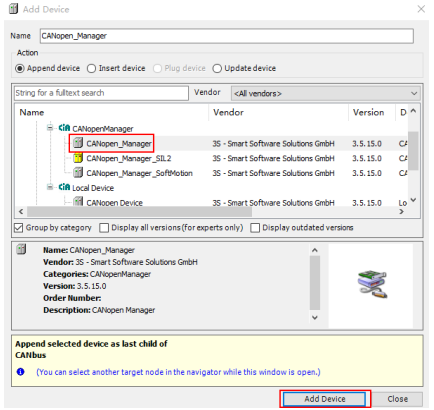

4. Click **Tools** in the menu bar and select **System Repository** as shown in the following figure. Click **Install** to import the EDS file.



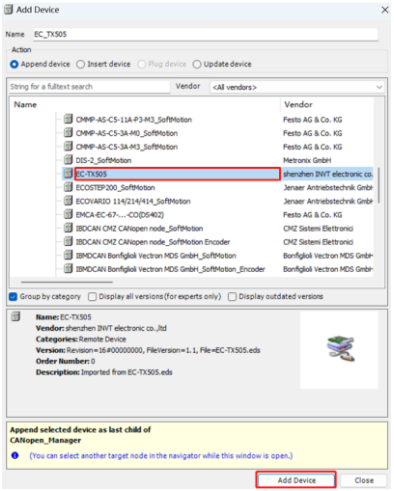

5. Right click **Device (INVT AX7X)** in the **Devices** pane, and choose **Add Device...** > **CANbus** > **Add Device**.



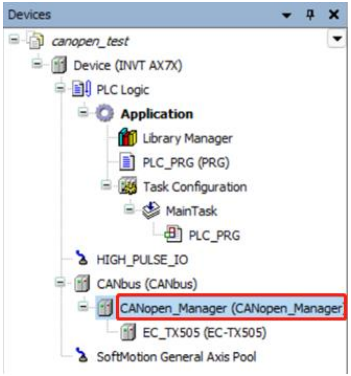

6. Right click **CANbus** in the **Devices** pane, and choose **Add Device...** > **CANopen_Manager** > **Add Device**.



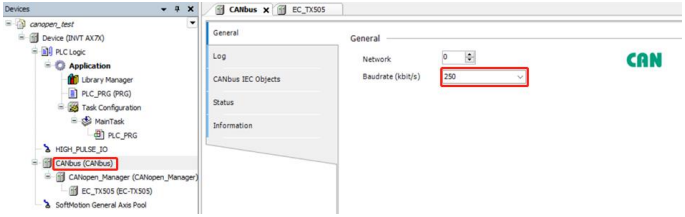

7. Right click **CANopen_Manager** in the **Devices** pane, and choose **Add Device...** > **EC-TX505** > **Add Device**.



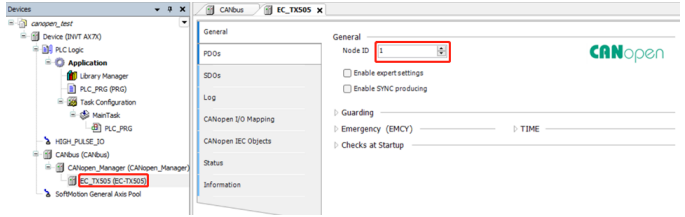
After devices are added completely, the interface is shown as follows.



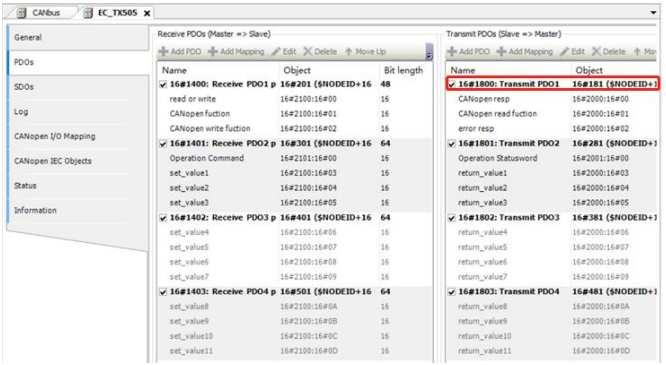

8. Double click **CANbus** in the **Devices** pane to set the baud rate of the network.

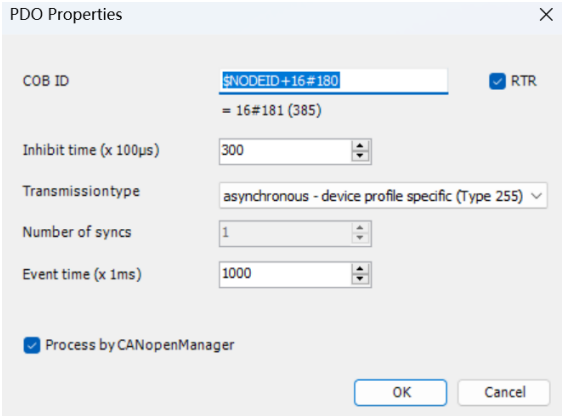



9. Double click **EC_TX505** in the **Devices** pane to set the node ID of the slave station.

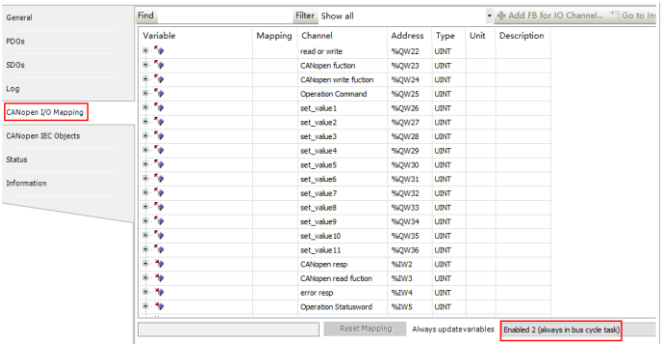



10. Double click **PDOs** in the above figure, and double click PDO mapping to set the PDO as shown in the following figure.

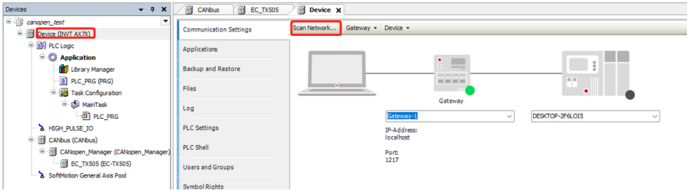


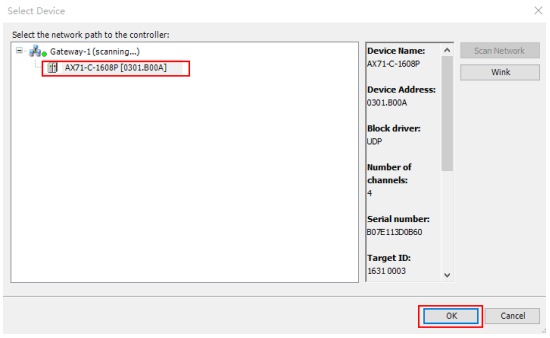




11. Double click **CANopen I/O Mapping** and select **Enabled 2 (always in bus cycle task)**.



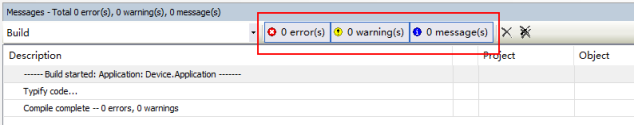

12. Double click **Device (INVT AX7X)** in the **Devices** pane, choose **Scan Network...**, and choose the PLC.



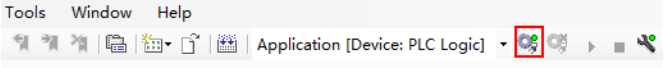



13. Click the **Compile** icon in the toolbar.

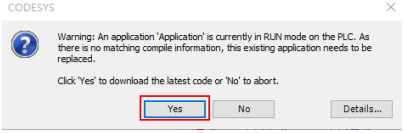

Make sure that there is no error.




14. Click the **Login** icon.

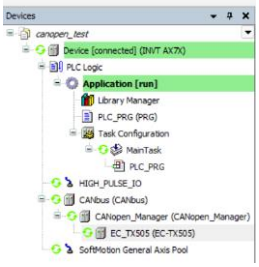



Click **Yes**.

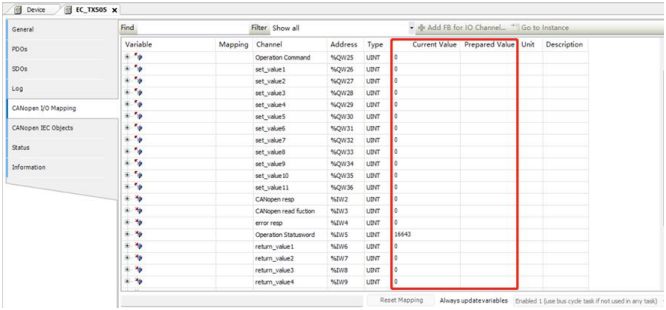



15. Click the **Run** icon.


The normal operation of the device is shown as follows.



16. Open **CANopen I/O Mapping** in the EC_TX505 page, modify the parameters of the VFD, and view the status of the VFD.



# Chapter 4 BACnet MSTP communication card

## 4.1 Overview

As a BACnet slave, it can realize operations such as the reading and writing of VFD process data and function codes, reading of VFD status words, and writing of VFD control words. The application layer supports setting 32 analog value objects and supports host controllers such as Yet Another BACnet Explorer (YABE) and PLC controller.

## 4.2 Features

In market demand, BACnet MS/TP is a data communication protocol used primarily in the building automation and HVAC (heating, ventilation and air conditioning) industries. The protocol supports devices such as fans, pumps and ventilation units to establish communication with PLCs. This facilitates a high level of automation in buildings.

The physical transmission medium for the bus is twisted pair (RS485 compliant), two-wire cable or fiber optic cable. The baud rate ranges from 9.6 kbit/s to 115.2 kbit/s. The maximum length of the bus cable is in the range of 100 m to 1200 m, depending on the selected transmission rate. A maximum of 31 nodes can be connected to the same network segment when repeaters are not used, while the number of nodes connected to the network (including repeaters and master nodes) can be increased to 127 when repeaters are used.

Table 4-1 BACnet MSTP card technical specifications

| Category | | Specifications |
|---|---|---|
| Main functions | Feature | ● Supports the BACnet protocol and BACnet MSTP devices.<br>● Provides one BACnet MSTP port and supports half-duplex operations of 115.2kbps.<br>● Supports the line, star, and daisy chain connection network topologies, with the number of slave nodes up to 31.<br>● Supports timeout detection. |
| | Service | ● Single-property reading service<br>● Multi-property reading service<br>● Single-property writing service<br>● Multi-property writing service<br>● I-Am service<br>● I-Have service<br>● Device communication control service<br>● Device re-initialization service |
| | Function | ● Supports the setting of 32 analog objects.<br>● Supports the reading and writing of VFD process data and function codes, reading of VFD status words, and writing of VFD control words. |

| Category | | Specifications |
|---|---|---|
| | | ● Supports host controllers such as YABE and PLC. |
| Others | Input power | Power voltage of 24V, current of 200mA, and maximum power of 4.8W |
| | Installation method | GD350 option, inserted to the slot |
| | Running environment temperature | -10 – +50 °C |
| | Storage temperature | -20 – +60 °C |
| | Relative humidity | 5%–95% (No condensation) |
| | Distance | Up to 100 m (cable of shielded twisted pair or with magnetic ring) from the controller |

## 4.3 Electrical wiring

1. Node selection

A node address is the unique address of a device on the bus, and the address number of a node is set by the function code P15.01.

2. Bus terminator

Each segment has a bus terminator at the head and at the tail to ensure error-free operation. The switches on the PCB are used to switch on the bus terminators, which prevent signal reflection at the bus cable end. If the communication card is the last or first module in the network, the bus terminator must be set to ON. When a D-sub connector with a built-in terminator is used, the communication card terminator must be disconnected.

Table 4-2 RS485 hardware specifications

| Item | Description |
|---|---|
| Number of ports | 1 |
| Isolation | 1.5kV electrical isolation |
| Standard | TIA/EIA-485 |
| Communication speed and duplex | 0.96k–115.2k baud (protocol depended), half-duplex |
| Connector type | Swappable, 3-bits, screw-type wiring board |
| Cable type | Cable of shielded twisted pair or with magnetic ring |
| Cable length | 100 m |
| Topology | Line, star, and daisy chain connection |
| Unit load | 1/8 |

## 4.4 Terminal wiring

### 4.4.1 DS-P-1004 control terminal layout

Table 4-3 BACnet MSTP card terminal layout

| | | | | | |
|---|---|---|---|---|---|
| 485+ | 485- | PGND | PE | | |

### 4.4.2 DS-P-1004 control terminal functions

Table 4-4 BACnet MSTP card terminal functions

| Signal | Port | Terminal function description |
|---|---|---|
| 485+ | RS-485 | Positive differential signal |
| 485- | | Negative differential signal |
| PGND | / | Signal ground |
| PE | / | Earth |

Table 4-5 Indicator functions

| Indicator | Definition | Function | | |
|---|---|---|---|---|
| LED1 | Status indicator | On: The expansion card is establishing a connection with the control board.<br>Blinking periodically: The expansion card is properly connected to the control board (the period is 1s, on for 0.5s, and off for the other 0.5s).<br>Off: The expansion card is disconnected from the control board. | | |
| LED2 | Run indicator (Green) | Off | Not powered or in fault state | Not powered or in fault state |
| | | Blinking periodically (on-off at a 0.5s interval) | Online. Waiting to receive BACnet data frames | Waiting to receive data |
| | | On | BACnet communication state | Received BACnet data frames |
| LED3 | Fault indicator (Red) | Off | No fault | No fault |
| | | Blinking periodically (on-off at a 0.5s interval for twice and 2s off) | Faulty | Duplicate address. The VFD keypad reports E-bAC. |
| | | Blinking | Faulty | No BACnet data frames |

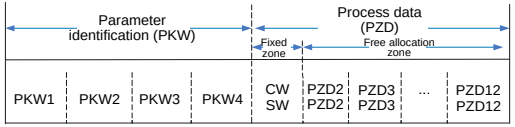
| Indicator | Definition | Function | | |
|---|---|---|---|---|
| | | periodically (on-off at a 0.5s interval for three times and 2s off) | | received in specified time. (Timeout detection must be enabled, that is, the timeout time cannot be 0). The VFD keypad reports E-bAC. |
| LED4 | Power indicator | It is on once the control board feeds power to the communication card. | | |
| LED5 | 485TX indicator (Green) | Off | No transmission | The device does not transmit data on the RS485 network. |
| | | Blinking | Transmitting | The device is transmitting data on the RS485 network. |
| | | On | Transmitting | The device is transmitting data fast on the RS485 network. |
| LED6 | 485RX indicator (Green) | Off | No receiving | The device does not receive data on the RS485 network. |
| | | Blinking | Receiving | The device is receiving data on the RS485 network. |
| | | On | Receiving | The device is receiving data fast on the RS485 network. |

## 4.5 Connecting the BACnet MSTP card to the PLC

### 4.5.1 Bus communication networking

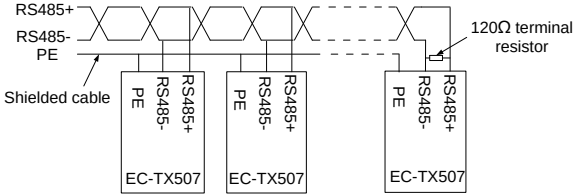
#### 4.5.1.1 Communication packet structure

The communication data frame structure (PKW+PZD) of BACnet MSTP is similar to that of PROFINET. See the following figure.

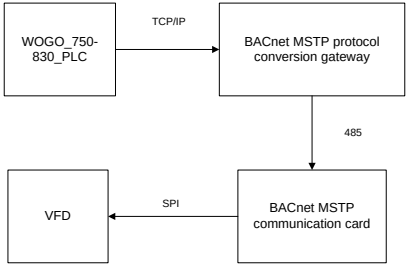



#### 4.5.1.2 Networking topology

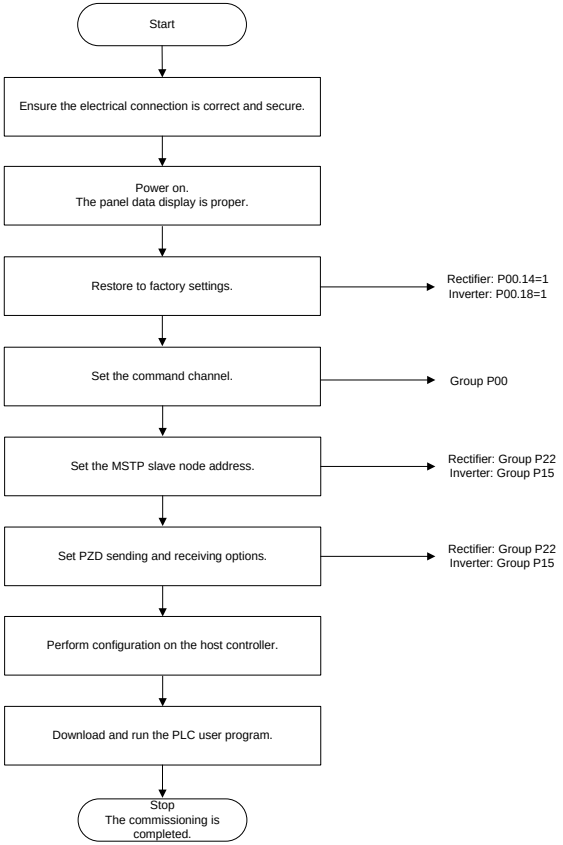
The BACnet MSTP communication card adopts standard 4-pin terminal interface, which can be used in line network topology, star network topology and daisy chain topology, and its electrical wiring diagram is shown in the following. BACnet MSTP cards form a communication network with the PLC by inserting a BACnet MSTP communication card into each unit.




Due to the constraints, the connection method currently used is to add the BACnet MSTP protocol conversion gateway between the PLC and the communication card. See the following figure.

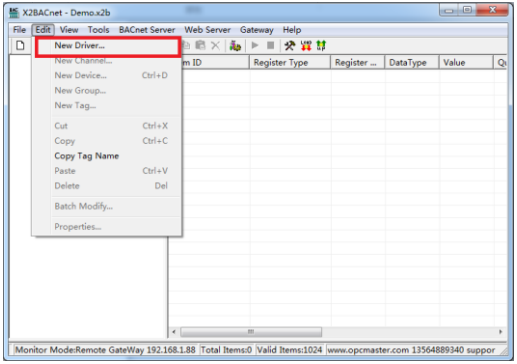



The BACnet MSTP networking commissioning flowchart is as follows.

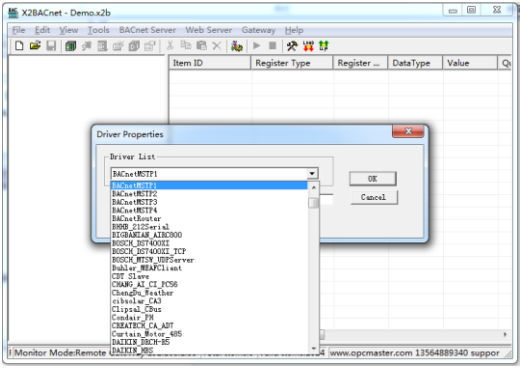



### 4.5.2 Configuring the BACnet MSTP protocol conversion gateway on the Sunfull PLC

1. Open X2BACnet, select the **New Driver** option in the menu bar of the host controller configuration interface, as shown below.

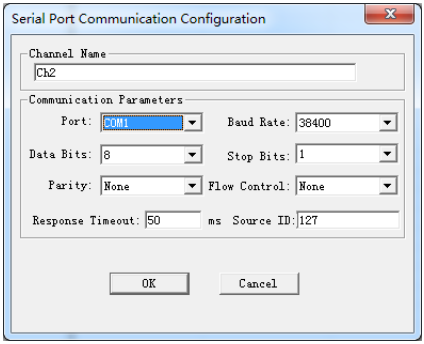



In the displayed window, select **BACnetMSTP1**. (For every COM port that is enabled to collect MSTP, an MSTP drive needs to be created. If two buses are used and there are COM1 and COM2, you can create BACnetMSTP1 and BACnetMSTP2 to correspond to COM1 and COM2. Create BACnetMSTP1, BACnetMSTP2, BACnetMSTP3 and BACnetMSTP4 in total to correspond to 4 COM ports, as shown in the following figure.

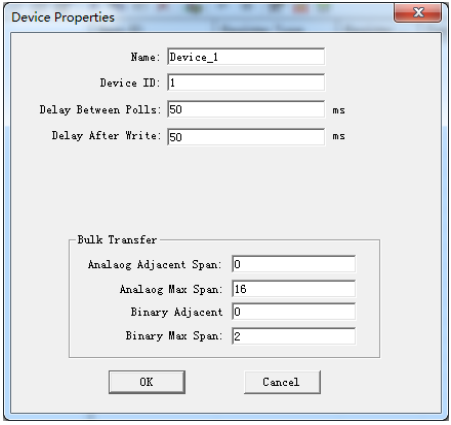



2. After drives are added, choose to create channels. As serial communication, the default BACnet MSTP channel serial port is COM1, and the other default communication parameters are as follows: 38400bps baud rate, 8 data bits, 1 stop bit,

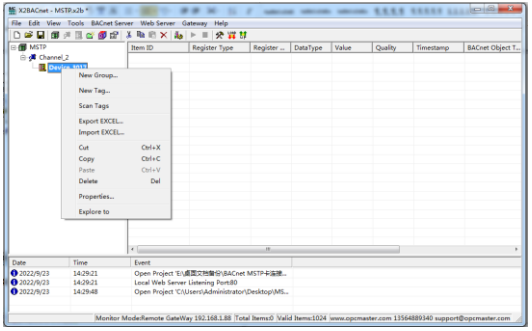
no parity bit, 50ms timeout time (the timeout time does not need to be modified in most cases), and master node MAC address of 127, which is not recommended to modify. See the following figure.



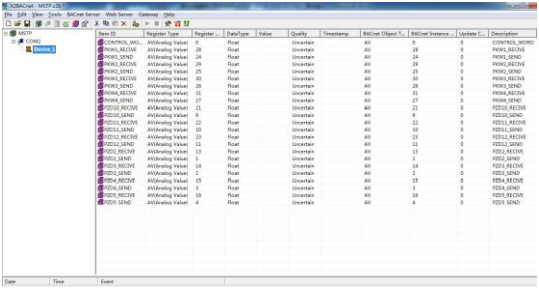

After the channel is created, choose **New Device**, set the device properties in the pop-up dialog box, including the device ID. You can query which devices are online on the bus through BACnetScan automatic scanning. The request frame interval (**Delay Between Polls**) is 50 milliseconds. See following figure.



3. Choose **Scan Tags** to scan the device for all tags. See the following figure.

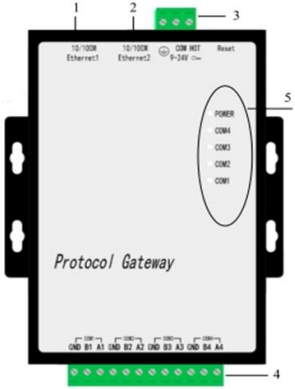



X2BACnet connects to the BACnet MSTP card through RS485. Add all scanned records, and save the project. See the following figure.

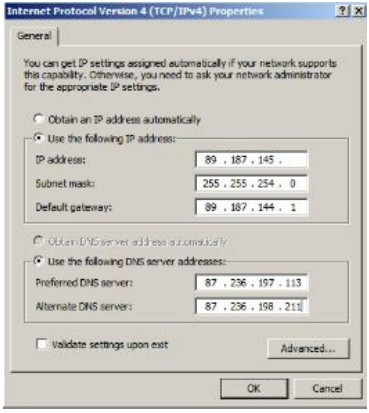


4. The PC port connects to port 2 of the protocol gateway 192.168.1.88. COM2 connects to the RS485 port of the BACnet MSTP card. Set the PC port properties. See the following figure.

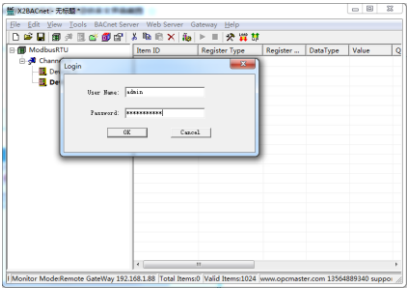
Open the X2BACnet host controller and import the saved project.



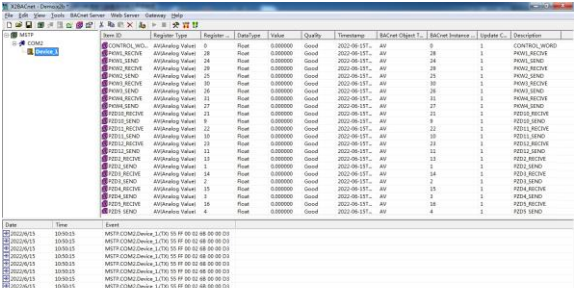

1. Port 1
2. Port 2
3. AC/DC power 24V
4. Fully electrical isolated 4-channel RS485 port
5. LED communication indicators




Click  to upload the project to the gateway. Enter the user name and password in the pop-up dialog box. The user name is fixed to **admin**, and the password is fixed to **admin123456**. Then you can set the system parameters related to the gateway. See the following figure.




You can choose the monitoring mode under **Monitor Mode** in the toolbar, or double click **Monitor Mode** in the bottom status bar to switch the mode. To use gateway monitoring, you need to switch the monitoring mode to gateway mode. Configure the project in X2BACnet and upload it to the hardware gateway to realize the protocol conversion function through the hardware gateway. Click ▶ to start monitoring on the X2BACnet software to monitor the communication status of the hardware gateway simultaneously. See the following figure.



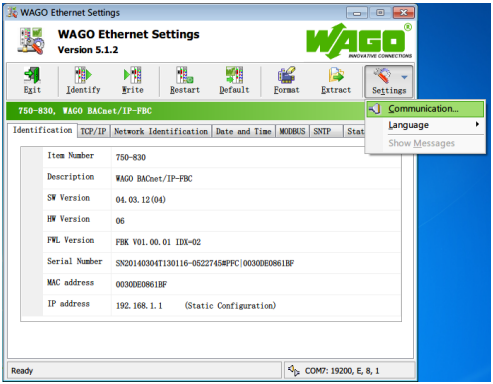
The configuration is completed.

### 4.5.3 Configuring WOGO 750-830 PLC

Connect the PLC to your PC with a network cable. Set your PC IP address (such as 192.168.1.250) on your network. Connect the PLC to the PC USB port with a burn-in cable for setting the PLC IP address (such as 192.168.1.1)

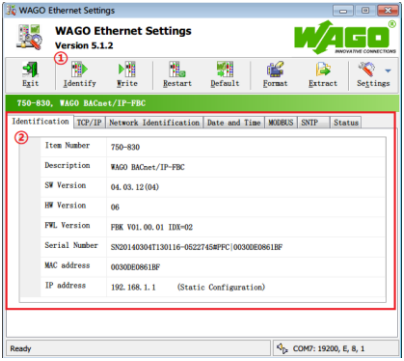
#### 4.5.3.1 Configuring the PLC (750-830) IP address through the WAGO Ethernet Settings

1. Open the WAGO Ethernet Settings software, click on **Settings** in the upper right corner, and then choose **Communication**, as shown in the following figure.

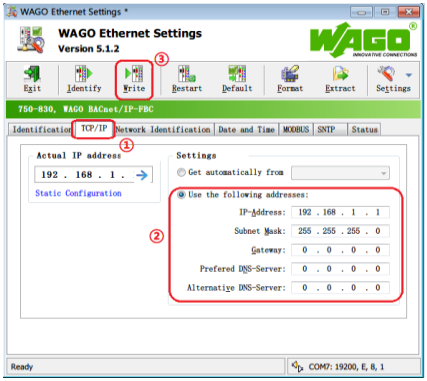



2. Select **Serial Ports** from the **Connection** drop-down list box, select the corresponding port number from the **Port** drop-down list box, and select the corresponding serial port number (which can be viewed in by choosing **Device Manager** > **Ports** on your PC), as shown in the following figure.

3. Click **Identify**, and wait for a while. If the PC and PLC are connected properly, the area in the red box as shown in the following figure will appear, showing information such as the PLC hardware and software versions, MAC address, and IP address. If the content in the red box area indicates that the connection has failed, check whether the connection between the PC and PLC is normal, as shown in the following figure.




4. Click **TCP/IP** and set the PLC IP address in red box ②, and ensure the PC IP address and the PLC IP address are in the same network segment. Then click **Write** to write the set IP address to the PLC. See the following figure.

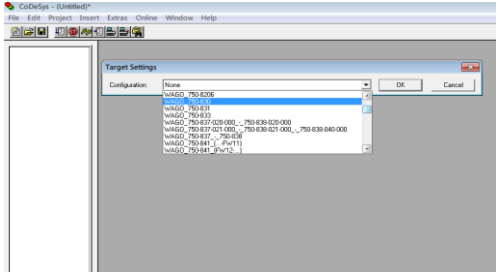


5. After the writing is successful, open CoDeSys V2.3.

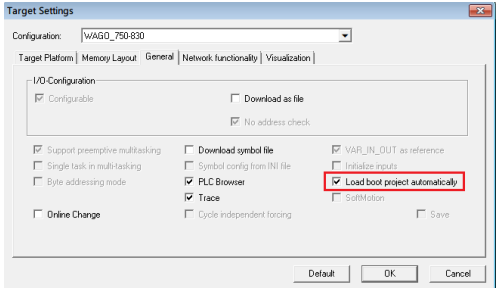
A. Use CoDeSys V2.3 to configure the PLC.

CoDeSys V2.3 has the same functions with WAGO-IO-PRO V2.3.

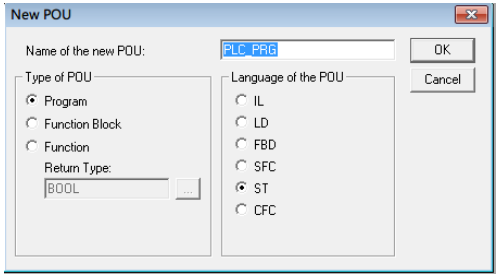
a) Open the CoDeSys software, click **File-New** to create a new document. The following window is displayed. Select the PLC model **WAGO_750-830** from the drop-down list box, and click **OK**. See the following figure.



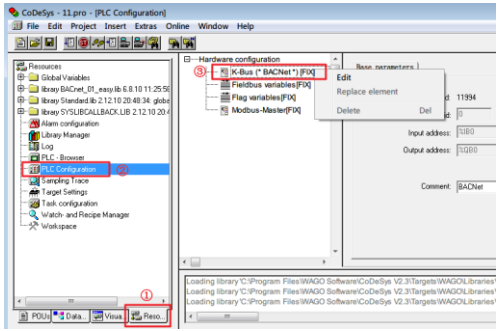

b) Click the **General** tab, choose **Load boot project automatically**, and click **OK**. See the following figure.



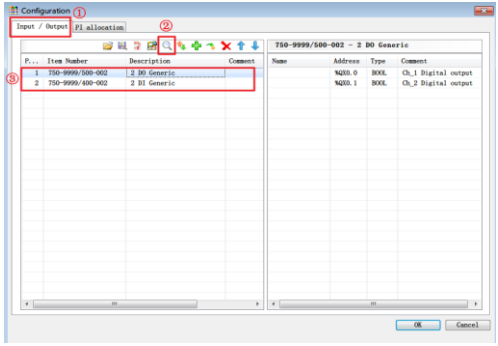

c) Create a POU program. Since the PLC must contain a PLC_PRG program, and keep the default settings and click **OK**. See the following figure.

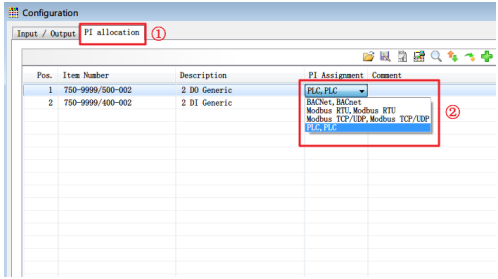



d) Configure the PLC hardware. Select **PLC Configuration** from the **Resource** drop-down list box. At the right, choose **Hardware configuration**, right-click **K-Bus**, and choose **Edit**. See the following figure.




e) Click the **Input/Output** tab in the **Configuration** dialog box. Click the icon in the toolbar to scan the devices on the PLC hardware bus. When the scan is completed, the devices in red box ③ will appear. Click on any device. The corresponding device address, data type and function description will appear in the right window. Select **PLC, PLC** from **PI Assignment** in the **PI allocation** tab, indicating the PLC controlled IO devices. If other types are selected, the IO devices are communication controlled. See the following figure.



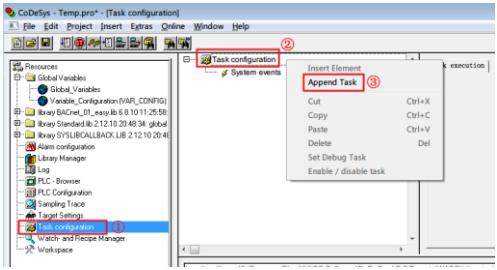




If no device is scanned, check the following:

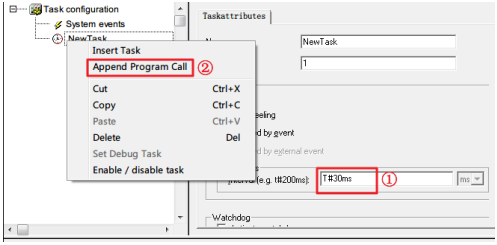
- Whether the PLC is properly powered.
- Whether the PLC and PC are connected properly and are in the same network segment.

f) Configure tasks.

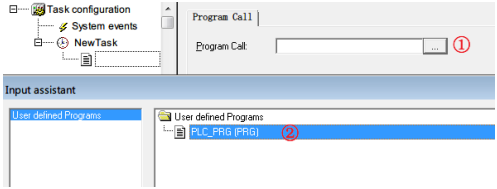
Create a task.




Set the circular interval to 30ms and append a program. See the following figure.




Select and add the program PLC_PRG. See the following figure.

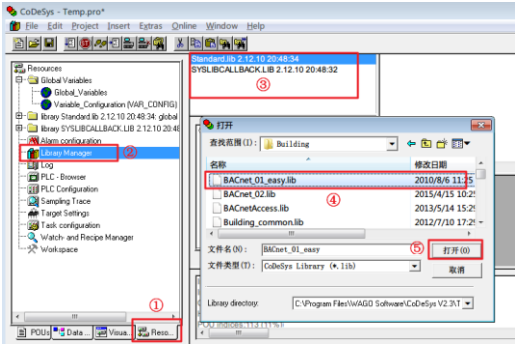



g) Import the BACnet library file.

Copy all files from **00_Libraries_BA** to the **Building** folder in the CoDeSys installation directory, for example, ***C:\Program Files\WAGO Software\CoDeSys V2.3\Targets\WAGO\Libraries\Building***.

After the copy is complete, choose **Library Manager** in the **Resource** tab in the CoDeSys main interface. Right-click on the blank area marked as ③, and choose Additional Library, choose **Building** > **BACnet_01_easy.lib**, and click **Open**. See

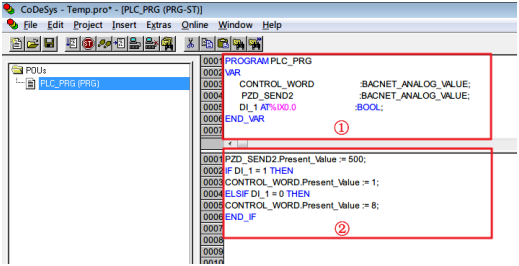
the following figure.




B. Create PLC programs.

This section describes how to write a simple application program, using the ST structured language for programming. The program achieves the following functions: If the VFD frequency is set to 5Hz, when the I0.0 terminal input of the digital input module 750–400 is high, the VFD runs forward; when the terminal input is low, the VFD decelerates to stop. (The VFD frequency is given by BACnet communication, the control mode is communication, and VFD received PZD2 is the set frequency.)

Variable declaration is in red box ①, and the main program is in red box ②. The format of data variable declaration is in the format of *Variable name*: *Data type of variable*.




Description of the content in red box ①:

**CONTROL_WORD** is VFD control word; **PZD_SEND2** is the given PZD of VFD, **DI_1** is the channel 0 of digital input module, and **%IX0.0** is the address of channel 0 of digital output module.

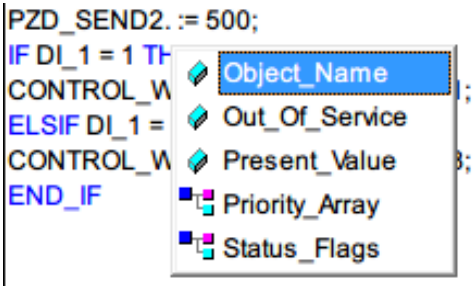
**BACNET_ANALOG_VALUE** is a data type of BACnet communication, which is defined in the library file **BACnet_01_easy.lib**.

```
PROGRAM PLC_PRG
VAR
  CONTROL_WORD                    :BACNET_ANALOG_VALUE;
   PZD_SEND2                      :BACNET_ANALOG_VALUE;
  DI_1 AT%IX0.0                                :BOOL;
END_VAR
```

Description of the content in red box ②:

**PZD_SEND2**: Several properties of this object can follow at the right of the decimal point.

**Present_Value** indicates the current value.




The syntax ":=" is the assignment symbol.

```
PZD_SEND2.Present_Value := 500;
IF DI_1 = 1 THEN
CONTROL_WORD.Present_Value := 1;
ELSIF DI_1 = 0 THEN
CONTROL_WORD.Present_Value := 8;
```

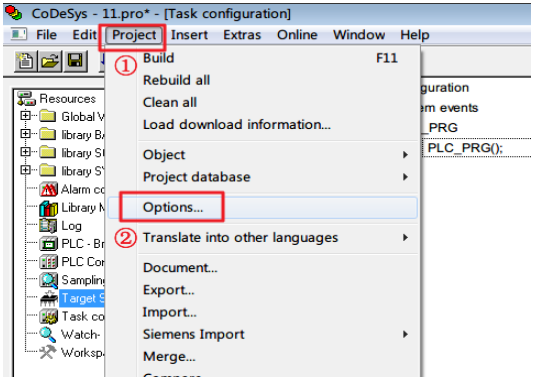
*END_IF*

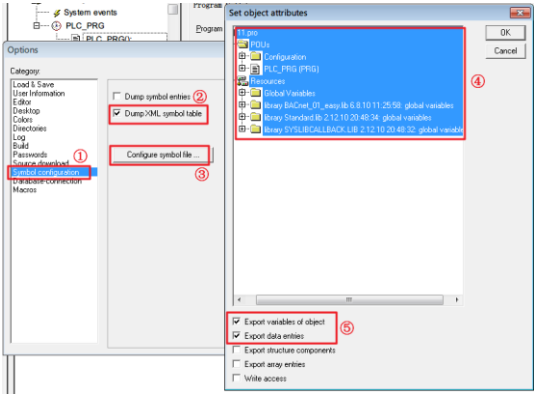
After the program is completed, press F11 to compile the project and proceed to the next step if there are no errors.

**Note:** After the program is completed, the PLC does not control the VFD directly, and you need to map the defined variables to the VFD variable table through **WAGO BACnet configurator**. Next, you need to generate the **SYM_XML variable** table and import it into the **WAGO BACnet configurator**.

C. Configure the symbol file.

In the main interface, choose **Project** > **Options** from the menu bar. In the pop-up **Options** dialog box, select **Dump XML Symbol table**, and click **Configura symbol file**. In the pop-up **Set object attributes** dialog box, select all, as shown in red box ④, and select the two items shown in red box ⑤. See the following figures.






After the setup, when the POU program is completed and the PLC is in login state, the file ***xxx*.sys_xml** (*xxx* is the project name) will be generated automatically and saved in the created project folder.

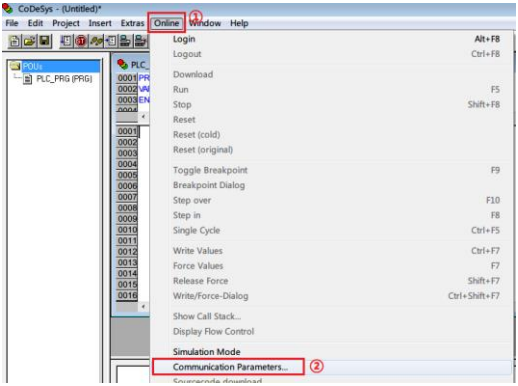
The **SYS_XML** file contains all variables in the project and can be imported directly into the **WAGO BACnet configurator** to configure the properties of each variable or object. This file is essential for the subsequent use of the PLC.

**Note:**

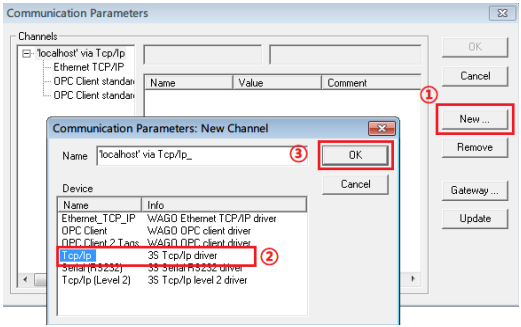
- If the system prompts that there is no relevant file locally during the generation process, you need to check whether the project built in CoDySys has been saved.
- If there are no variables declared in the POU program in the generated variable table, you need to check whether you have followed the steps in e) strictly.
- The XML file is generated only during the first PLC login. If the POU program has not changed, the XML file will not be regenerated subsequently, even if you log in again.

D. Download the program to the PLC to generate the **SYM_XML** file.

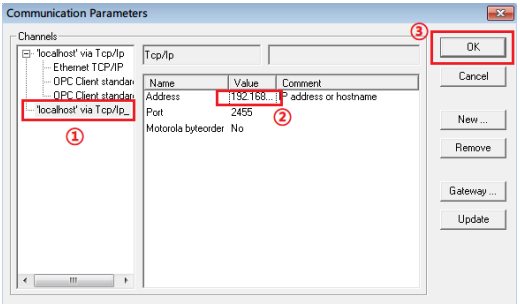
a) Choose **Online** > **Communication Parameters**. See the following figure.



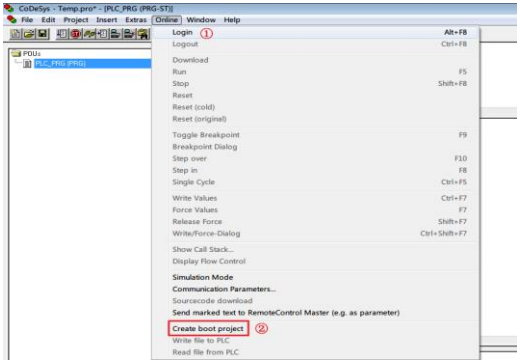

b) In the **Communication Parameters** window, click the **NEW** button. Select **Tcp/Ip (3S Tcp/Ip driver)** in the new pop-up window, and then click **OK**. See the following figure.




c) Select the newly created channel in the **Communication Parameters** window, and then fill in the PLC IP address in the **Value** column next to **Address**, and click **OK**. See the following figure.




d) Choose **Online** > **Login** on the menu bar. The software will compile the project and generate the **SYM_XML** file, which is located in the root directory of the project. After that, choose **Create boot project** to import the program file into the PLC, so that the PLC can run independently even if the PLC is disconnected from the PC. See the following figure.

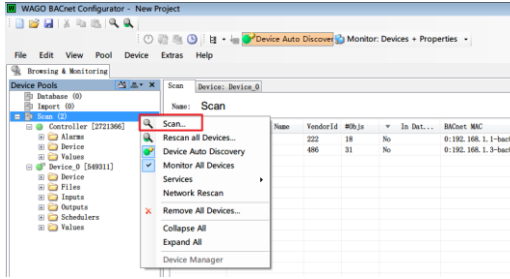



E. Configure the PLC through the WAGO BACnet configurator.

a) Set the IP addresses of the PLC, PC, VFD and other slave stations, which are required to be in the same network segment, and connect each node through the network cable.

b) Open the WAGO BACnet configurator and it will automatically scan the devices on the bus. Alternatively, scan manually by right-clicking on **Device Pools-Scan**

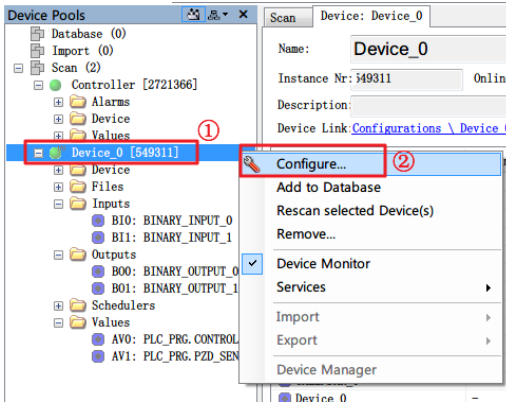
and choosing **Scan**. See the following figure.



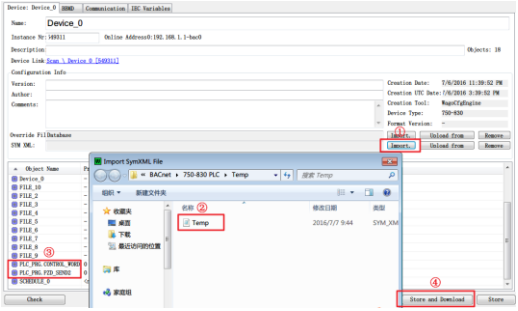

In the preceding figure, **Controller [2721366]** is the VFD node in the bus and **Device_0 [549311]** is the PLC node. The numeric part in [] is the decimal number converted from this node MAC address.

c) Import the **SYM_XML** file.

Right click on **Device_0** (PLC device), and choose **Configure**. See the following figure.




Import the previously generated **SYM_XML** file. After successful import, the objects (CONTROL_WORD, PZD_SEND2) declared in the previous project will appear in area ③. See the following figure.




**Note:**

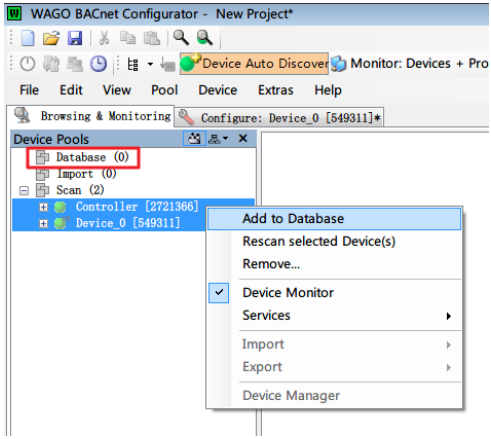
- After the import is completed, check the I/O indicator of the PLC. If the import is correct, the indicator is steady on in green. If it is blinking in red periodically, you need to follow the instructions to perform troubleshooting.
- You can check the PLC status through the **Status** window of WAGO Ethernet Settings.
- For details about PLC panel indicators, see WAGO 750-830 description document.

d) Map PLC variables to VFD variables.

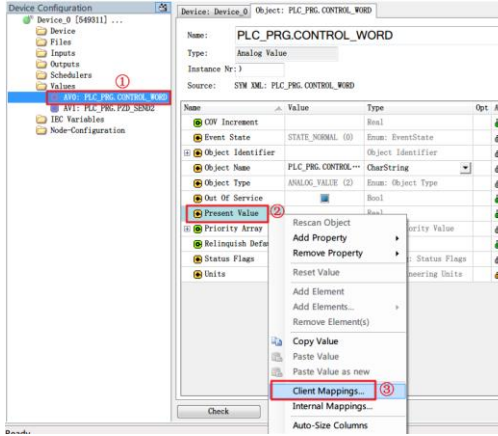
Since the WAGO 750-830 PLC can only be used as a slave, not as a master, when used in a BACnet network. In actual use, it is necessary to map the relevant variables of the VFD or other devices to the PLC so that the VFD can be controlled by only operating the mapped address in the PLC through the PLC program.

The following example associates the **CONTROL_WORD** control word in the PLC program with the **CONTROL_WORD** in the VFD to control VFD startup, stop, and reset. The associated operations of other variables are similar.

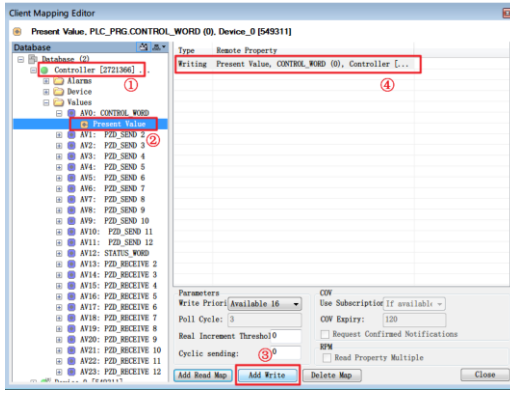
a) Select all devices that are scanned through the Scan menu. Right-click **Add to Database**. Choose **Device Pools** > **Database** to view the added devices. See the following figure.



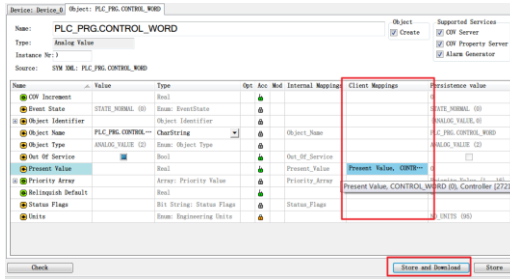

b) Access the **Configuration** interface of Device_0 (PLC device). Chose **Values** > **AV0: PLC_PRG.CONRTOL_WORD**. In the right area, choose **Present Value** > **Client Mappings**. See the following figure.



c) In the **Client Mapping Editor** window, select the VFD variable to be mapped with the PLC variable **AV0: PLC_PRG.CONRTOL_WORD**. Select **Controller – AV0: CONTROL_WORD-Present Value**, and click **Add Write**. See the following figure.



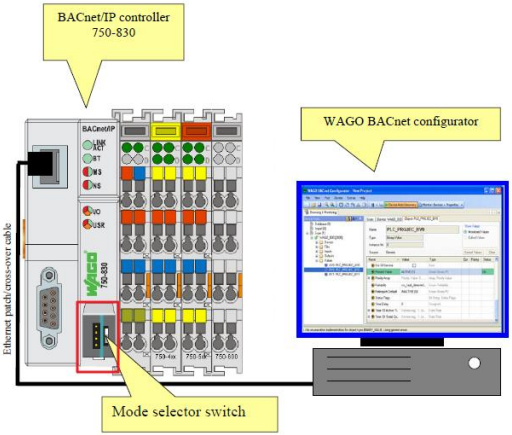

d) You can view the added variable in the **Client Mappings** column in the property window of the PLC variable **AV0: PLC_PRG.CONRTOL_WORD**. Map other variables with the same method. After mapping all variables, click **Store and Download** to download the property to the PLC. See the following figure.




F. Run the program.

Disconnect the PLC from the PC. Slide the PLC run switch to the top. The mode selection switch is in the red rectangular in the following figure. When the switch is

slid to the top, it is in the running state. See the following figure.

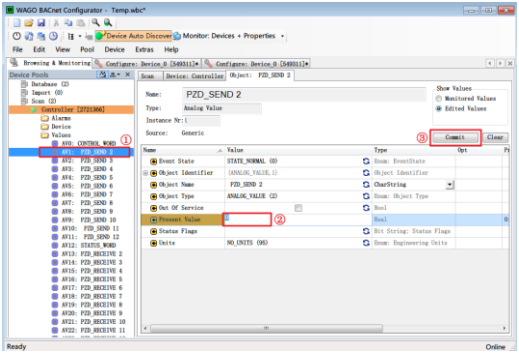



### 4.5.4 Controlling the VFD through WAGO BACnet configurator

If you only need to read and write VFD parameters and achieve basic control VFD functions, you can control the VFD directly through the WAGO BACnet configurator, without writing a PLC program.

1. Set the PZD related function codes of the VFD, and set the VFD control mode to communicate.

2. In the scanned devices, select the device (VFD) that needs to be controlled, and take **Controller [2721366]** as an example. Select the PZD functions or control words to be operated, taking **PZD_SEND2** as an example. Enter the required value in the **Present Value** field. Then click **Commit**.




Set other parameters in the same way.

# Chapter 5 CAN master/slave expansion card

## 5.1 CAN master/slave expansion card operation

### 5.1.1 Electrical connection

Use shielding wires as the bus cable, if possible. It is recommended that you connect the shielding wire to the PE terminal of the communication card. When there are only two devices for CAN master-slave communication, both devices shall be connected to the terminal resistor. When there are more than two devices, the starting device and terminal device shall be connected to the terminal resistor. The terminal resistor of the communication card can be connected through its terminal resistor switch. Figure 5-1 shows the electrical wiring.

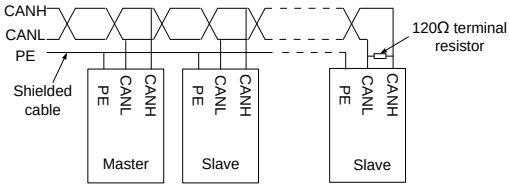



Figure 5-1 Electrical wiring diagram

### 5.1.2 Electrical wiring

| Connection operation | Description |
|---|---|
| 1. Determine the master count and slave count, and prepare the shielded twisted pairs. | Pay attention to the cable specifications. |
| 2. Connect the cable according to Figure 5-1, and install the terminal resistor at the head and tail properly. | Pay attention to the terminal resistors. |
| 3. Check the card hardware version according to the order finished-product card model. | Pay attention to the board silk model number. |
| 4. Power on in a unified way.<br>Check card type related function codes P19.00/P19.01/P19.02, card version related function codes P19.03/P19.04/P19.05, and control board software version function code P07.13 for consistency. | Check whether the latest version is used. |
| 5. Use the following function code setting examples for reference. | Pay attention to the actual configuration. |
| 6. Power on in a unified way, and check whether the running is normal. | If both the salve and slave work properly, the networking is |

| Connection operation | Description |
|---|---|
| | successful. |
| 7. Set torque and frequency related function codes according to the actual application. | Pay attention to the actual application. |

**5.1.3 Function code setting**

| Master setting example | | |
|---|---|---|
| **Function code viewing and setting** | **Value** | **Description** |
| P19.00/P19.01/19.02 | 15 | CAN master/slave communication card type is successfully identified according to the card slot. |
| P19.03/P19.04/19.05 | *.** | CAN master/slave communication card is successfully identified according to the card type and version number. |
| **P28.00** | **1** | **The local device is a master.** |
| **P28.06** | **N** | **Slave count** |
| P15.28 | / | The communication address is invalid when it is a master. |
| P15.29 | 2 | The communication baud rate is 125kbps. |
| P15.30 | 5.0 | The CAN communication timeout time is 5s. |

| Example of slave 1 | | |
|---|---|---|
| **Function code viewing and setting** | **Value** | **Description** |
| P19.00/P19.01/19.02 | 15 | CAN master/slave communication card type is successfully identified according to the card slot. |
| P19.03/P19.04/19.05 | *.** | CAN master/slave communication card is successfully identified according to the card type and version number. |
| **P28.00** | **2** | **The local device is a slave.** |
| P28.06 | / | The count is invalid when it is a slave. |
| **P15.28** | **1** | **The slave CAN communication** |

| Example of slave 1 | | |
|---|---|---|
| **Function code viewing and setting** | **Value** | **Description** |
| | | **address starts from 1.** |
| P15.29 | 2 | The communication baud rate is 125kbps. |
| P15.30 | 5.0 | The CAN communication timeout time is 5s. |

| Example of slave N | | |
|---|---|---|
| **Function code viewing and setting** | **Value** | **Description** |
| P19.00/P19.01/19.02 | 15 | CAN master/slave communication card type is successfully identified according to the card slot. |
| P19.03/P19.04/19.05 | *.** | CAN master/slave communication card is successfully identified according to the card type and version number. |
| **P28.00** | **2** | **The local device is a slave.** |
| P28.06 | / | The count is invalid when it is a slave. |
| **P15.28** | **N** | **The slave CAN communication address ends at N.** |
| P15.29 | 2 | The communication baud rate is 125kbps. |
| P15.30 | 5.0 | The CAN communication timeout time is 5s. |

**5.1.4 Fault handling**

| Fault | Description |
|---|---|
| Neither the master nor the slave has a fault. | The master/slave networking is successful. |
| The salve reports fault code 58: CAN communication fault (ESCAN) | The cause is that the slave timeout time is reached. Check the slave electrical wiring and function code settings. Check whether the fault is removed after reset. |
| The master reports fault code 69: CAN slave fault in master/slave synchronization (S-Err). | The cause is that a slave encounters a fault. Check the slave electrical wiring and function code settings. Check whether the fault is |

| Fault | Description |
|---|---|
| | removed after reset. |
| The salve reports fault code 58: CAN communication fault (ESCAN). | The cause is that the master timeout time is reached. Check the master electrical wiring and function code settings. Check whether the fault is removed after reset. |

## 5.2 Related function parameters

| Function code | Name | Parameter description | Default | Modify |
|---|---|---|---|---|
| P15.28 | CAN communication address | 0–127 | 1 | ◎ |
| P15.29 | CAN communication baud rate | 0–5<br>0: 50kbps<br>1: 100kbps<br>2: 125kbps<br>3: 250kbps<br>4: 500kbps<br>5: 1Mbps | 2 | ◎ |
| P15.30 | CAN communication timeout period | 0.0–60.0(s)<br>**Note:** It is invalid when the value is 0.0. | 5.0 | ○ |
| P19.00 | Type of expansion card at slot 1 | 0–65535<br>0: No card<br>1: PLC card<br>2: I/O card<br>3: Incremental PG card<br>4: Incremental PG card with UVW<br>5: Ethernet<br>6: DP<br>7: Bluetooth card<br>8: Resolver PG card<br>9: CANopen Communication card<br>10: WIFI card<br>11: PROFINET communication card<br>12: Sine-cosine PG card without CD signals<br>13: Sine-cosine PG card with CD signals | 0 | ● |
| P19.01 | Type of expansion card at slot 2 | | 0 | ● |
| P19.02 | Type of expansion card at slot 3 | | 0 | ● |

| Function code | Name | Parameter description | Default | Modify |
|---|---|---|---|---|
| | | 14: Absolute encoder PG card (reserved)<br>15: CAN master/slave communication card<br>16: Modbus TCP communication card<br>17: EtherCAT communication card (reserved)<br>18: BACnet communication card (reserved)<br>19: DeviceNET communication card (reserved)<br>20: PT100/PT1000 temperature detection card<br>21: EtherNet IP card<br>22: MECHATROLINK card (reserved)<br>23–65535: Reserved | | |
| P28.00 | Master/slave mode | 0–2<br>0: Master/slave control is invalid.<br>1: The local device is a master.<br>2: The local device is a slave. | 0 | ◎ |
| P28.01 | Master/slave communication data selection | 0–1<br>0: CAN<br>1: Reserved | 0 | ◎ |
| P28.02 | Master/slave control mode | 0x000–0x112<br>Ones place: Master/slave running mode selection<br>0: Master/slave mode 0<br>1: Master/slave mode 1<br>2: Master/slave mode 2<br>Tens place: Slave start command source<br>0: Master<br>1: Determined by P00.01<br>Hundreds place: Whether to enable master/slave to send/receive data | 0x001 | ◎ |

| Function code | Name | Parameter description | Default | Modify |
|---|---|---|---|---|
| | | 0: Enable<br>1: Disable | | |
| P28.03 | Slave speed gain | 0.0–500.0% | 100.0% | ○ |
| P28.04 | Slave torque gain | 0.0–500.0% | 100.0% | ○ |
| P28.05 | Speed/torque mode switching frequency point in master/slave mode 2 | 0.00–10.00Hz | 5.00Hz | ○ |
| P28.06 | Slave count | 1–15 | 1 | ◎ |
| P28.07 | Hidden | 0–0 | 0 | ● |
| P28.08 | Hidden | 0–0 | 0 | ● |
| P28.09 | CAN slave torque offset | -100.0–100.0% | 0.0% | ○ |

# Appendix A CANopen object dictionary

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| 1000 | 0 | Device type | RO | Unsigned32 | 0x0000 0000 |
| 1001 | 0 | Error register | RO | Unsigned8 | / |
| 1003 | **Error code register** | | | | |
| | 0 | Number of subindexes | RW | / | / |
| | 1 | Error code | RO | Unsigned32 | / |
| 1005 | 0 | COB-ID SYNC | RW | Unsigned32 | / |
| 1006 | 0 | Communication cycle period | RW | Unsigned32 | / |
| 1007 | 0 | Length of synchronous window | RW | Unsigned32 | / |
| 1008 | 0 | Manufacturer-defined device name | CONST | String | INVT CANopen |
| 1009 | 0 | Manufacturer-defined hardware version | CONST | String | V1.00 |
| 100A | 0 | Manufacturer-defined software version | CONST | String | V1.00 |
| 100C | 0 | Protection time | RW | Unsigned16 | 0 |
| 100D | 0 | Life cycle factor | RW | Unsigned16 | 0 |
| 1016 | **Consumer heartbeat time** | | | | |
| | 0 | Number of subindexes | RO | Unsigned8 | / |
| | 1 | Consumer heartbeat time | RW | Unsigned32 | / |
| 1017 | 0 | Producer heartbeat time | RW | Unsigned16 | 0 |
| 1018 | **Identifier objects** | | | | |
| | 0 | Number of subindexes | RO | Unsigned8 | 4 |
| | 1 | Supplier ID | RO | Unsigned32 | 0x0000 0000 |
| | 2 | Product code | RO | Unsigned32 | 0x0000 0000 |

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| | 3 | Revision No. | RO | Unsigned32 | 0x0000 0000 |
| | 4 | Sequence No. | RO | Unsigned32 | 0x0000 0000 |
| 1200 | **Servo SDO** | | | | |
| | 0 | Number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID Client -> server (Rx) | RO | Unsigned32 | 600H+Node ID |
| | 2 | COB-ID Server -> client (Tx) | RO | Unsigned32 | 580H+Node ID |
| 1280 | **SDO** | | | | |
| | 0 | Number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID Client -> server (Rx) | RO | Unsigned32 | / |
| | 2 | COB-ID Server -> client (Tx) | RO | Unsigned32 | / |
| | 3 | Node ID of server SDO | RO | Unsigned8 | / |
| 1400 | **PDO1 Rx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | / |
| | 3 | / | / | Unsigned16 | / |
| | 4 | / | / | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | / |
| 1401 | **PDO2 Rx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | / |
| | 3 | / | / | Unsigned16 | / |
| | 4 | / | / | Unsigned8 | / |

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| | 5 | Event timer | RW | Unsigned16 | / |
| 1402 | **PDO3 Rx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | / |
| | 3 | / | / | Unsigned16 | / |
| | 4 | / | / | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | / |
| 1403 | **PDO4 Rx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | / |
| | 3 | / | / | Unsigned16 | / |
| | 4 | / | / | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | / |
| 1600 | **PDO1 Rx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 3 |
| | 1 | First mapped object | RW | Unsigned32 | 0x21000010 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x21000110 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x21000210 |
| 1601 | **PDO2 Rx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 4 |

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| | 1 | First mapped object | RW | Unsigned32 | 0x21010010 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x21000310 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x21000410 |
| | 4 | Fourth mapped object | RW | Unsigned32 | 0x21000510 |
| 1602 | **PDO3 Rx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 4 |
| | 1 | First mapped object | RW | Unsigned32 | 0x21000610 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x21000710 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x21000810 |
| | 4 | Fourth mapped object | RW | Unsigned32 | 0x21000910 |
| 1603 | **PDO4 Rx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 4 |
| | 1 | First mapped object | RW | Unsigned32 | 0x21000a10 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x21000b10 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x21000c10 |
| | 4 | Fourth mapped object | RW | Unsigned32 | 0x21000d10 |
| 1800 | **PDO1 Tx communication parameters** | | | | |
| | 0 | Supported Max. number of | RO | Unsigned8 | / |

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| | | subindexes | | | |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | 255 |
| | 3 | Disabled time | RW | Unsigned16 | 500 |
| | 4 | Reserved | RW | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | 0 |
| 1801 | **PDO2 Tx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | 254 |
| | 3 | Disabled time | RW | Unsigned16 | 500 |
| | 4 | Reserved | RW | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | 0 |
| 1802 | **PDO3 Tx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | 254 |
| | 3 | Disabled time | RW | Unsigned16 | 500 |
| | 4 | Reserved | RW | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | 0 |
| 1803 | **PDO4 Tx communication parameters** | | | | |
| | 0 | Supported Max. number of subindexes | RO | Unsigned8 | / |
| | 1 | COB-ID used by PDO | RW | Unsigned32 | / |
| | 2 | Transmission type | RW | Unsigned8 | 254 |
| | 3 | Disabled time | RW | Unsigned16 | 500 |
| | 4 | Reserved | RW | Unsigned8 | / |
| | 5 | Event timer | RW | Unsigned16 | 0 |

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| 1A00 | **PDO1 Tx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 3 |
| | 1 | First mapped object | RW | Unsigned32 | 0x20000010 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x20000110 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x20000210 |
| 1A01 | **PDO2 Tx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 4 |
| | 1 | First mapped object | RW | Unsigned32 | 0x20010010 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x20000310 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x20000410 |
| | 4 | Fourth mapped object | RW | Unsigned32 | 0x20000510 |
| 1A02 | **PDO3 Tx mapping parameters** | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 4 |
| | 1 | First mapped object | RW | Unsigned32 | 0x20000610 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x20000710 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x20000810 |
| | 4 | Fourth mapped object | RW | Unsigned32 | 0x20000910 |

| Index (hexadecimal) | Subindex | Description | Access permission | Data type | Default value |
|---|---|---|---|---|---|
| 1A03 | PDO4 Tx mapping parameters | | | | |
| | 0 | Number of application program objects mapped in PDO | RW | Unsigned8 | 4 |
| | 1 | First mapped object | RW | Unsigned32 | 0x20000a10 |
| | 2 | Second mapped object | RW | Unsigned32 | 0x20000b10 |
| | 3 | Third mapped object | RW | Unsigned32 | 0x20000c10 |
| | 4 | Fourth mapped object | RW | Unsigned32 | 0x20000d10 |

# Appendix B Related function codes

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P00.01 | Channel of running commands | 0: Keypad<br>1: Terminal<br>2: Communication | 0–2 | 0 |
| P00.02 | Communication mode of running commands | 0: Modbus/Modbus TCP communication<br>1: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>2: Ethernet communication<br>3: EtherCAT/PROFINET/EtherNet IP communication<br>4: Programmable card<br>5: Wireless communication card<br>6: USB communication (reserved)<br>**Note:** The values 1–5 correspond to extended functions that are available only with respective cards. | 0–6 | 0 |
| P00.06 | A frequency command selection | 0: Set via keypad<br>1: Set via AI1<br>2: Set via AI2<br>3: Set via AI3<br>4: Set via high speed pulse HDIA<br>5: Set via simple PLC program<br>6: Set via multi-step speed running<br>7: Set via PID control<br>8: Set via Modbus/Modbus TCP communication<br>9: Set via PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>10: Set via Ethernet communication<br>11: Set via high speed pulse HDIB<br>12: Set via pulse train AB<br>13: Set via EtherCAT/PROFINET communication | 0–15 | 0 |
| P00.07 | B frequency command selection | | 0–15 | 15 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 14: Set via programmable card<br>15: Reserved | | |
| P00.11 | ACC time 1 | 0.0–3600.0s | 0.0–3600.0s | Model depended |
| P03.11 | Torque setting mode selection | 0–1: Keypad (P03.12)<br>2: AI1<br>3: AI2<br>4: AI3<br>5: Pulse frequency HDIA<br>6: Multi-step torque<br>7: Modbus/Modbus TCP communication<br>8: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>9: Ethernet communication<br>10: Pulse frequency HDIB<br>11: EtherCAT/PROFINET/EtherNet IP<br>12: Programmable card<br>**Note:** For these settings, 100% corresponds to the motor rated current. | 0–12 | 0 |
| P03.14 | Setting source of FWD rotation frequency upper limit in torque control | 0: Keypad (P03.16)<br>1: AI1<br>2: AI2<br>3: AI3<br>4: Pulse frequency HDIA<br>5: Multi-step setting<br>6: Modbus/Modbus TCP communication<br>7: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>8: Ethernet communication<br>9: Pulse frequency HDIB<br>10: EtherCAT/PROFINET/EtherNet IP communication<br>11: Programmable card | 0–12 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 12: Reserved<br>**Note:** For these settings, 100% corresponds to the max. frequency. | | |
| P03.15 | Setting source of REV rotation frequency upper limit in torque control | 0: Keypad (P03.17)<br>1: AI1<br>2: AI2<br>3: AI3<br>4: Pulse frequency HDIA<br>5: Multi-step setting<br>6: Modbus/Modbus TCP communication<br>7: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>8: Ethernet communication<br>9: Pulse frequency HDIB<br>10: EtherCAT/PROFINET/EtherNet IP communication<br>11: Programmable card<br>12: Reserved<br>**Note:** For these settings, 100% corresponds to the max. frequency. | 0–12 | 0 |
| P03.18 | Setting source of electromotive torque upper limit | 0: Keypad (P03.20)<br>1: AI1<br>2: AI2<br>3: AI3<br>4: Pulse frequency HDIA<br>5: Modbus/Modbus TCP communication<br>6: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>7: Ethernet communication<br>8: Pulse frequency HDIB<br>9: EtherCAT/PROFINET/EtherNet IP communication<br>10: Programmable card | 0–11 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 11: Reserved<br>**Note:** For these settings, 100% corresponds to the motor rated current. | | |
| P03.19 | Setting source of braking torque upper limit | 0: Keypad (P03.21)<br>1: AI1<br>2: AI2<br>3: AI3<br>4: Pulse frequency HDIA<br>5: Modbus/Modbus TCP communication<br>6: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>7: Ethernet communication<br>8: Pulse frequency HDIB<br>9: EtherCAT/PROFINET/EtherNet IP communication<br>10: Programmable card<br>11: Reserved<br>**Note:** For these settings, 100% corresponds to the motor rated current. | 0–11 | 0 |
| P04.27 | Voltage setting channel | 0: Keypad; output voltage is determined by P04.28<br>1: AI1<br>2: AI2<br>3: AI3<br>4: HDIA<br>5: Multi-step (the set value is determined by P10 group)<br>6: PID<br>7: Modbus/Modbus TCP communication<br>8: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>9: Ethernet communication<br>10: HDIB | 0–13 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 11: EtherCAT/PROFINET/EtherNet IP<br>12: Programmable card<br>13: Reserved | | |
| P06.01 | Y1 output | 0: Invalid<br>1: In running<br>2: In forward running<br>3: In reverse running<br>4: In jogging<br>5: VFD fault<br>6: Frequency level detection FDT1<br>7: Frequency level detection FDT2<br>8: Frequency reached<br>9: Running in zero speed<br>10: Reach upper limit frequency<br>11: Reach lower limit frequency<br>12: Ready to run<br>13: In pre-exciting<br>14: Overload pre-alarm<br>15: Underload pre-alarm<br>16: Simple PLC stage completed<br>17: Simple PLC cycle completed<br>18: Reach set counting value<br>19: Reach designated counting value<br>20: External fault is valid<br>21: Reserved<br>22: Reach running time<br>23: Virtual terminal output of Modbus communication<br>24: Virtual terminal output of POROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>25: Virtual terminal output of Ethernet communication<br>26: DC bus voltage established<br>27: z pulse output<br>28: During pulse superposition | 0–63 | 0 |
| P06.02 | HDO output | | 0–63 | 0 |
| P06.03 | Relay output RO1 | | 0–63 | 1 |
| P06.04 | Relay output RO2 | | 0–63 | 5 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 29: STO act<br>30: Positioning completed<br>31: Spindle zeroing completed<br>32: Spindle scale-division completed<br>33: In speed limit<br>34: EtherCAT/PROFINET/EtherNet IP<br>35: Reserved<br>36: Speed/position control switchover completed<br>37: Any frequency reached<br>38–40: Reserved<br>41: Y1 from the programmable card<br>42: Y2 from the programmable card<br>43: HDO from the programmable card<br>44: RO1 from the programmable card<br>45: RO2 from the programmable card<br>46: RO3 from the programmable card<br>47: RO4 from the programmable card<br>48: EC PT100 detected OH pre-alarm<br>49: EC PT1000 detected OH pre-alarm<br>50: AI/AO detected OH pre-alarm<br>51: Stopped or running at zero speed<br>52: Disconnection detected in tension control<br>53: Roll diameter setting reached<br>54: Max. roll diameter reached<br>55: Min. roll diameter reached<br>56: Fire control mode enabled<br>57–63: Reserved | | |
| P06.14 | Analog output AO1 | 0: Running frequency (0–Max. output frequency) | 0–63 | 0 |
| P06.16 | HDO high-speed pulse output | 1: Set frequency (0–Max. output frequency)<br>2: Ramp reference frequency (0–Max. output frequency)<br>3: Rotational speed (100% corresponds | 0–63 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | to the speed at max. output frequency.)<br>4: Output current (100% corresponds to twice the VFD rated current.)<br>5: Output current (100% corresponds to twice the motor rated current.)<br>6: Output voltage (100% corresponds to 1.5 times the VFD rated voltage.)<br>7: Output power (100% corresponds to twice the motor rated power.)<br>8: Set torque (100% corresponds to twice the motor rated current.)<br>9: Output torque (Absolute value; 100% corresponds to twice the motor rated torque.)<br>10: AI1 input (0–10V/0–20mA)<br>11: AI2 input (0–10V)<br>12: AI3 input (0–10V/0–20mA)<br>13: HDIA input (0.00–50.00kHz)<br>14: Value 1 set through Modbus (0–1000)<br>15: Value 2 set through Modbus (0–1000)<br>16: Value 1 set through PROFIBUS/CANopen/DeviceNet/BACnet MSTP (0–1000)<br>17: Value 2 set through PROFIBUS/CANopen/DeviceNet/BACnet MSTP (0–1000)<br>18: Value 1 set through Ethernet 1 (0–1000)<br>19: Value 2 set through Ethernet 2 (0–1000)<br>20: HDIB input (0.00–50.00kHz)<br>21: Value 1 set through EtherCAT/PROFINET/EtherNet IP (0–1000) | | |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 22: Torque current (bipolar; 100% corresponds to triple the motor rated current.)<br>23: Exciting current (bipolar; 100% corresponds to triple the motor rated current.)<br>24: Set frequency (bipolar; 0–Max. output frequency)<br>25: Ramp reference frequency (bipolar; 0–Max. output frequency)<br>26: Rotational speed (bipolar; 0–Speed corresponding to max. output frequency)<br>27: Value 2 set through EtherCAT/PROFINET/EtherNet IP communication (0–1000)<br>28: AO1 from the programmable card (0–1000)<br>29: AO2 from the programmable card (0–1000)<br>30: Rotational speed (100% corresponds to twice the motor rated synchronous speed)<br>31: Output torque (Actual value, 100% corresponds to twice the motor rated torque)<br>32: AI/AO temperature detection output<br>33–63: Reserved<br>**Note:**<br>When the output comes from the programmable card (28–29), if the card is a Codesys programmable card, P27.00 must be set to 1.<br>When AO1 is of the current output type, 100% corresponds to 20mA; when AO1 is of the voltage output type, 100% corresponds to 10V; 100% of HDO | | |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | corresponds to the output of P06.30. | | |
| P07.27 | Type of current fault | 0: No fault<br>1: Inverter unit U phase protection (OUt1)<br>2: Inverter unit V phase protection (OUt2)<br>3: Inverter unit W phase protection (OUt3)<br>4: Overcurrent during acceleration (OC1)<br>5: Overcurrent during deceleration (OC2)<br>6: Overcurrent during constant speed (OC3)<br>7: Overvoltage during acceleration (OV1)<br>8: Overvoltage during deceleration (OV2)<br>9: Overvoltage during constant speed (OV3)<br>10: Bus undervoltage fault (UV)<br>11: Motor overload (OL1)<br>12: VFD overload (OL2)<br>13: Phase loss on input side (SPI)<br>14: Phase loss on output side (SPO)<br>15: Rectifier module overheat (OH1)<br>16: Inverter module overheat (OH2)<br>17: External fault (EF)<br>18: Modbus/Modbus TCP communication fault (CE)<br>19: Current detection fault (ItE)<br>20: Motor autotuning fault (tE)<br>21: EEPROM operation fault (EEP)<br>22: PID feedback offline fault (PIDE)<br>23: Brake unit fault (bCE)<br>24: Running time reached (END)<br>25: Electronic overload (OL3)<br>26: Keypad communication error (PCE) | 0–79 | 0 |
| P07.28 | Type of last fault | | 0–79 | 0 |
| P07.29 | Type of 2nd-last fault | | 0–79 | 0 |
| P07.30 | Type of 3rd-last fault | | 0–79 | 0 |
| P07.31 | Type of 4th-last fault | | 0–79 | 0 |
| P07.32 | Type of 5th-last fault | | 0–79 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 27: Parameter upload error (UPE)<br>28: Parameter download error (DNE)<br>29: PROFIBUS communication fault (E-DP)<br>30: Ethernet communication fault (E-NET)<br>31: CANopen communication fault (E-CAN)<br>32: To-ground short-circuit fault 1 (ETH1)<br>33: To-ground short-circuit fault 2 (ETH2)<br>34: Speed deviation fault (dEu)<br>35: Mal-adjustment fault (STo)<br>36: Underload fault (LL)<br>37: Encoder offline fault (ENC1o)<br>38: Encoder reversal fault (ENC1d)<br>39: Encoder Z pulse offline fault (ENC1Z)<br>40: Safe torque off (STO)<br>41: Channel H1 safety circuit exception (STL1)<br>42: Channel H2 safety circuit exception (STL2)<br>43: Channel H1 and H2 exception (STL3)<br>44: Safety code FLASH CRC fault (CrCE)<br>45: Programmable card customized fault 1 (P-E1)<br>46: Programmable card customized fault 2 (P-E2)<br>47: Programmable card customized fault 3 (P-E3)<br>48: Programmable card customized fault 4 (P-E4) | | |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 49: Programmable card customized fault 5 (P-E5)<br>50: Programmable card customized fault 6 (P-E6)<br>51: Programmable card customized fault 7 (P-E7)<br>52: Programmable card customized fault 8 (P-E8)<br>53: Programmable card customized fault 9 (P-E9)<br>54: Programmable card customized fault 10 (P-E10)<br>55: Repetitive expansion card type fault (E-Err)<br>56: Encoder UVW loss fault (ENCUV)<br>57: PROFIBUS communication fault (E-PN)<br>58: CANopen communication fault (SECAN)<br>59: Motor over-temperature fault (OT)<br>60: Card slot 1 card identification failure (F1-Er)<br>61: Card slot 2 card identification failure (F2-Er)<br>62: Card slot 3 card identification failure (F3-Er)<br>63: Card slot 1 card communication timeout fault (C1-Er)<br>64: Card slot 2 card communication timeout fault (C2-Er)<br>65: Card slot 3 card communication timeout fault (C3-Er)<br>66: EtherCAT communication fault (E-CAT)<br>67: BACnet communication fault (E-BAC)<br>68: DeviceNet communication fault (E-DEV) | | |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 69: Master-slave synchronous CAN slave fault (S-Err)<br>70: EC PT100 detected overheating (OtE1)<br>71: EC PT1000 detected overheating (OtE2)<br>72: EtherNet/IP communication timeout (E-EIP)<br>73: No upgrade bootload (E-PAO)<br>74: AI1 disconnected (E-AI1)<br>75: AI2 disconnected (E-AI2)<br>76: AI3 disconnected (E-AI3)<br>77: AI/AO detected OH (OH3)<br>78: Brake feedback fault (E-brF)<br>79: Stalling in V/F control (E-StK)<br>80: Losing steps in V/F control (E-LSt) | | |
| P08.31 | Motor 1 and motor 2 switching channel | LED ones place: Switching channel<br>0: Terminal<br>1: Modbus/Modbus TCP communication<br>2: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>3: Ethernet communication<br>4: EtherCAT/PROFINET/EtherNet IP communication<br>LED tens place: Switching in running<br>0: Disabled<br>1: Enabled | 0x00–0x14 | 0x00 |
| P09.00 | PID reference source | 0: Set by P09.01<br>1: AI1<br>2: AI2<br>3: AI3<br>4: High-speed pulse HDIA<br>5: Multi-step<br>6: Modbus communication<br>7: | 0–12 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>8: Ethernet communication<br>9: High-speed pulse HDIB<br>10: EtherCAT/PROFINET communication<br>11: Programmable card<br>12: Reserved | | |
| P09.02 | PID feedback source | 0: AI1<br>1: AI2<br>2: AI3<br>3: High-speed pulse HDIA<br>4: Modbus communication<br>5: PROFIBUS/CANopen/DeviceNet/BACnet MSTP communication<br>6: Ethernet communication<br>7: High-speed pulse HDIB<br>8: EtherCAT/PROFINET communication<br>9: Programmable expansion card<br>10: Reserved | 0–10 | 0 |
| P15.01 | Module address | 0–127 | 0–127 | 2 |
| P15.02 | Received PZD2 | 0–31<br>0: Invalid<br>1: Set frequency (0–Fmax, unit: 0.01 Hz)<br>2: PID reference (-1000–1000, in which 1000 corresponds to 100.0%)<br>3: PID feedback (-1000–1000, in which 1000 corresponds to 100.0%)<br>4: Torque setting (-3000–+3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>5: Setting of the upper limit of forward running frequency (0–Fmax, unit: 0.01 Hz) | 0–31 | 0 |
| P15.03 | Received PZD3 | | 0–31 | 0 |
| P15.04 | Received PZD4 | | 0–31 | 0 |
| P15.05 | Received PZD5 | | 0–31 | 0 |
| P15.06 | Received PZD6 | | 0–31 | 0 |
| P15.07 | Received PZD7 | | 0–31 | 0 |
| P15.08 | Received PZD8 | | 0–31 | 0 |
| P15.09 | Received PZD9 | | 0–31 | 0 |
| P15.10 | Received PZD10 | | 0–31 | 0 |
| P15.11 | Received PZD11 | | 0–31 | 0 |
| P15.12 | Received PZD12 | | 0–31 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 6: Setting of the upper limit of reverse running frequency (0–Fmax, unit: 0.01 Hz)<br>7: Upper limit of the electromotive torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>8: Upper limit of the brake torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>9: Virtual input terminal command, 0x000–0x3FF<br>(corresponding to S8, S7, S6, S5, HDIB, HDIA, S4, S3, S2, and S1 in sequence)<br>10: Virtual output terminal command, 0x00–0x0F<br>(corresponding to RO2, RO1, HDO, and Y1 in sequence)<br>11: Voltage setting (for V/F separation) (0–1000, in which 1000 corresponds to 100.0% of the rated voltage of the motor)<br>12: AO1 output setting 1 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>13: AO2 output setting 2 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>14: MSB of position reference (signed number)<br>15: LSB of position reference (unsigned number)<br>16: MSB of position feedback (signed number)<br>17: LSB of position feedback (unsigned number)<br>18: Position feedback setting flag (position feedback can be set only after | | |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | this flag is set to 1 and then to 0)<br>19: Function code mapping (PZD2–PZD12 correspond to P14.49–P14.59 respectively.)<br>20–31: Reserved | | |
| P15.13 | Transmitted PZD2 | 0–47<br>0: Invalid<br>1: Running frequency (×100, Hz)<br>2: Set frequency (×100, Hz)<br>3: Bus voltage (×10, V)<br>4: Output voltage (×1, V)<br>5: Output current (×10, A)<br>6: Actual output torque (×10, %)<br>7: Actual output power (×10, %)<br>8: Rotating speed of the running (×1, RPM)<br>9: Linear speed of the running (×1, m/s)<br>10: Ramp frequency reference<br>11: Fault code<br>12: AI1 value (×100, V)<br>13: AI2 value (×100, V)<br>14: AI3 value (×100, V)<br>15: HDIA frequency (×1000, kHz)<br>16: Terminal input state<br>17: Terminal output state<br>18: PID reference (×10, %)<br>19: PID feedback (×10, %)<br>20: Rated torque of the motor<br>21: MSB of position reference (signed number)<br>22: LSB of position reference (unsigned number)<br>23: MSB of position feedback (signed number)<br>24: LSB of position feedback (unsigned number)<br>25: Status word<br>26: HDIB frequency value (×1000, kHz) | 0–47 | 0 |
| P15.14 | Transmitted PZD3 | | 0–47 | 0 |
| P15.15 | Transmitted PZD4 | | 0–47 | 0 |
| P15.16 | Transmitted PZD5 | | 0–47 | 0 |
| P15.17 | Transmitted PZD6 | | 0–47 | 0 |
| P15.18 | Transmitted PZD7 | | 0–47 | 0 |
| P15.19 | Transmitted PZD8 | | 0–47 | 0 |
| P15.20 | Transmitted PZD9 | | 0–47 | 0 |
| P15.21 | Transmitted PZD10 | | 0–47 | 0 |
| P15.22 | Transmitted PZD11 | | 0–47 | 0 |
| P15.23 | Transmitted PZD12 | | 0–31 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | 27: MSB of PG card pulse feedback count<br>28: LSB of PG card pulse feedback count<br>29: MSB of PG card pulse reference count<br>30: LSB of PG card pulse reference count<br>31: Function code mapping (PZD2–PZD12 correspond to P14.60–P14.70 respectively.)<br>32: Status word 3<br>33–47: Reserved | | |
| P15.25 | DP communication timeout time | 0.0 (invalid)–60.0s | 0.0–60.0 | 5.0s |
| P15.26 | CANopen communication timeout time | 0.0 (invalid)–60.0s | 0.0–60.0 | 5.0s |
| P15.27 | CANopen communication baud rate | 0: 1Mbps<br>1: 800K bps<br>2: 500K bps<br>3: 250K bps<br>4: 125K bps<br>5: 100K bps<br>6: 50K bps<br>7: 20K bps | 0–7 | 3 |
| P15.28 | CAN communication address | 0–127 | 0–127 | 1 |
| P15.29 | CAN baud rate setting | 0: 50K bps<br>1: 100K bps<br>2: 125K bps<br>3: 250K bps<br>4: 500K bps<br>5: 1M bps | 0–5 | 2 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P15.30 | CAN communication timeout time | 0.0 (invalid)–60.0s | 0.0–60.0 | 5.0s |
| P15.31 | DeviceNet communication timeout time | 0.0 (invalid)–60.0s | 0.0–60.0 | 5.0s |
| P15.41 | BACnet communication mode selection | 0: P15.46 is valid.<br>1: P15.42 is valid. | 0–1 | 0 |
| P15.42 | BACnet_MSTP baud rate | 0–5<br>0: 9600bps<br>1: 19200bps<br>2: 38400bps<br>3: 57600bps<br>4: 76800bps<br>5: 115200bps | 0–5 | 0 |
| P15.43 | Communication control word expression format | 0: Decimal format<br>1: Binary format | 0–1 | 0 |
| P15.44 | Communication control word/status word display selection | 0: Display currently identified card (only one)<br>1: DP card<br>2: CANopen card<br>3: PROFINET card<br>4: Ethernet IP card<br>5: Modbus TCP card<br>6: EtherCAT card | 0–6 | 0 |
| P15.45–P15.69 | Reserved | | | |
| P16.01 | Reserved | | | |
| P16.02 | Ethernet monitoring card IP address 1 | 0–255 | 0–255 | 192 |
| P16.03 | Ethernet monitoring card IP address 2 | 0–255 | 0–255 | 168 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P16.04 | Ethernet monitoring card IP address 3 | 0–255 | 0–255 | 0 |
| P16.05 | Ethernet monitoring card IP address 4 | 0–255 | 0–255 | 1 |
| P16.06 | Ethernet monitoring card subnet mask 1 | 0–255 | 0–255 | 255 |
| P16.07 | Ethernet monitoring card subnet mask 2 | 0–255 | 0–255 | 255 |
| P16.08 | Ethernet monitoring card subnet mask 3 | 0–255 | 0–255 | 255 |
| P16.09 | Ethernet monitoring card subnet mask 4 | 0–255 | 0–255 | 0 |
| P16.10 | Ethernet monitoring card subnet gateway 1 | 0–255 | 0–255 | 192 |
| P16.11 | Ethernet monitoring card subnet gateway 2 | 0–255 | 0–255 | 168 |
| P16.12 | G Ethernet monitoring card subnet gateway 3 | 0–255 | 0–255 | 1 |
| P16.13 | Ethernet monitoring card subnet gateway 4 | 0–255 | 0–255 | 1 |
| P16.14 | Ethernet card monitoring variable address 1 | 0x0000–0xFFFF | 0000–FFFF | 0x0000 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P16.15 | Ethernet card monitoring variable address 2 | 0x0000–0xFFFF | 0000–FFFF | 0x0000 |
| P16.16 | Ethernet card monitoring variable address 3 | 0x0000–0xFFFF | 0000–FFFF | 0x0000 |
| P16.17 | Ethernet card monitoring variable address 4 | 0x0000–0xFFFF | 0000–FFFF | 0x0000 |
| P16.18–P16.23 | Reserved | | | |
| P16.24 | Expansion card identification time of slot 1 | 0.0–600.0s<br>When this parameter is set to 0.0, disconnection fault detection is not performed. | 0.0–600.0 | 0.0s |
| P16.25 | Expansion card identification time of slot 2 | 0.0–600.0s<br>When this parameter is set to 0.0, disconnection fault detection is not performed. | 0.0–600.0 | 0.0s |
| P16.26 | Expansion card identification time of slot 3 | 0.0–600.0s<br>When this parameter is set to 0.0, disconnection fault detection is not performed. | 0.0–600.0 | 0.0s |
| P16.27 | Expansion card communication timeout time of slot 1 | 0.0–600.0s<br>When this parameter is set to 0.0, disconnection fault detection is not performed. | 0.0–600.0 | 0.0s |
| P16.28 | Expansion card communication timeout time of slot 2 | 0.0–600.0s<br>When this parameter is set to 0.0, disconnection fault detection is not performed. | 0.0–600.0 | 0.0s |
| P16.29 | Expansion card communication timeout time of slot 3 | 0.0–600.0s<br>When this parameter is set to 0.0, disconnection fault detection is not performed. | 0.0–600.0 | 0.0s |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P16.30 | Reserved | | | |
| P16.31 | PROFINET communication timeout time | 0.0 (invalid)–60.0s | 0.0–60.0 | 5.0s |
| P16.32 | Received PZD2 | 0: Invalid<br>1: Set frequency (0–Fmax, unit: 0.01 Hz)<br>2: PID reference (-1000–1000, in which 1000 corresponds to 100.0%)<br>3: PID feedback (-1000–1000, in which 1000 corresponds to 100.0%)<br>4: Torque setting (-3000–+3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>5: Setting of the upper limit of forward running frequency (0–Fmax, unit: 0.01 Hz)<br>6: Setting of the upper limit of reverse running frequency (0–Fmax, unit: 0.01 Hz)<br>7: Upper limit of the electromotive torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>8: Upper limit of the brake torque (0–3000, in which 1000 corresponds to 100.0% of the rated current of the motor)<br>9: Virtual input terminal command, 0x000–0x3FF (bit9–bit0 correspond to S8/S7/S6/S5/HDIB/HDIA/S4/S3/S2/S1 in sequence)<br>10: Virtual output terminal command, 0x00–0x0F (bit3–bit0 correspond to RO2/RO1/HDO/Y1 in sequence)<br>11: Voltage setting (for V/F separation) (0–1000, in which 1000 corresponds to 100.0% of the rated voltage of the | 0–31 | 0 |
| P16.33 | Received PZD3 | | 0–31 | 0 |
| P16.34 | Received PZD4 | | 0–31 | 0 |
| P16.35 | Received PZD5 | | 0–31 | 0 |
| P16.36 | Received PZD6 | | 0–31 | 0 |
| P16.37 | Received PZD7 | | 0–31 | 0 |
| P16.38 | Received PZD8 | | 0–31 | 0 |
| P16.39 | Received PZD9 | | 0–31 | 0 |
| P16.40 | Received PZD10 | | 0–31 | 0 |
| P16.41 | Received PZD11 | | 0–31 | 0 |
| P16.42 | Received PZD12 | | 0–31 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | motor)<br>12: AO1 output setting 1 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>13: AO2 output setting 2 (-1000–+1000, in which 1000 corresponds to 100.0%)<br>14: MSB of position reference (signed number)<br>15: LSB of position reference (unsigned number)<br>16: MSB of position feedback (signed number)<br>17: LSB of position feedback (unsigned number)<br>18: Position feedback setting flag (position feedback can be set only after this flag is set to 1 and then to 0)<br>19: Function code mapping (PZD2–PZD12 correspond to P14.49–P14.59 respectively.)<br>20–31: Reserved | | |
| P16.43 | Transmitted PZD2 | 0: Invalid<br>1: Running frequency (×100, Hz)<br>2: Set frequency (×100, Hz)<br>3: Bus voltage (×10, V)<br>4: Output voltage (×1, V)<br>5: Output current (×10, A)<br>6: Actual output torque (×10, %)<br>7: Actual output power (×10, %)<br>8: Rotating speed of the running (×1, RPM)<br>9: Linear speed of the running (×1, m/s)<br>10: Ramp frequency reference<br>11: Fault code<br>12: AI1 value (×100, V)<br>13: AI2 value (×100, V)<br>14: AI3 value (×100, V)<br>15: HDIA frequency (×1000, kHz)<br>16: Terminal input state | 0–47 | 0 |
| P16.44 | Transmitted PZD3 | | 0–47 | 0 |
| P16.45 | Transmitted PZD4 | | 0–47 | 0 |
| P16.46 | Transmitted PZD5 | | 0–47 | 0 |
| P16.47 | Transmitted PZD6 | | 0–47 | 0 |
| P16.48 | Transmitted PZD7 | | 0–47 | 0 |
| P16.49 | Transmitted PZD8 | | 0–47 | 0 |
| P16.50 | Transmitted PZD9 | | 0–47 | 0 |
| P16.51 | Transmitted PZD10 | | 0–47 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P16.52 | Transmitted PZD11 | 17: Terminal output state<br>18: PID reference (×10, %)<br>19: PID feedback (×10, %)<br>20: Rated torque of the motor<br>21: MSB of position reference (signed number)<br>22: LSB of position reference (unsigned number)<br>23: MSB of position feedback (signed number)<br>24: LSB of position feedback (unsigned number)<br>25: Status word<br>26: HDIB frequency value (×1000, kHz)<br>27: MSB of PG card pulse feedback count<br>28: LSB of PG card pulse feedback count<br>29: MSB of PG card pulse reference count<br>30: LSB of PG card pulse reference count<br>31: Function code mapping (PZD2–PZD12 correspond to P14.60–P14.70 respectively.)<br>32: Status word 3<br>33–47: Reserved | 0–47 | 0 |
| P16.53 | Transmitted PZD12 | | 0–47 | 0 |
| P16.54 | EtherNet IP communication timeout time | 0.0–60.0s | 0.0–60.0s | 5.0s |
| P16.55 | EtherNet IP communication rate setting | 0: Self-adaption<br>1: 100M full duplex<br>2: 100M half duplex<br>3: 10M full duplex<br>4: 10M half duplex | 0–4 | 0 |
| P16.56 | Bluetooth pairing code | 0–65535 | 0–65535 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P16.57 | Bluetooth host type | 0: No host connection<br>1: Mobile APP<br>2: Bluetooth box<br>3–8: Reserved | 0–8 | 0 |
| P16.58 | Industrial Ethernet communication card IP address 1 | 0–255 | 0–255 | 192 |
| P16.59 | Industrial Ethernet communication card IP address 2 | 0–255 | 0–255 | 168 |
| P16.60 | Industrial Ethernet communication card IP address 3 | 0–255 | 0–255 | 0 |
| P16.61 | Industrial Ethernet communication card IP address 4 | 0–255 | 0–255 | 20 |
| P16.62 | Industrial Ethernet communication card subnet mask 1 | 0–255 | 0–255 | 255 |
| P16.63 | Industrial Ethernet communication card subnet mask 2 | 0–255 | 0–255 | 255 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| P16.64 | Industrial Ethernet communication card subnet mask 3 | 0–255 | 0–255 | 255 |
| P16.65 | Industrial Ethernet communication card subnet mask 4 | 0–255 | 0–255 | 0 |
| P16.66 | Industrial Ethernet communication card gateway 1 | 0–255 | 0–255 | 192 |
| P16.67 | Industrial Ethernet communication card gateway 2 | 0–255 | 0–255 | 168 |
| P16.68 | Industrial Ethernet communication card gateway 3 | 0–255 | 0–255 | 0 |
| P16.69 | Industrial Ethernet communication card gateway 4 | 0–255 | 0–255 | 1 |
| P19.00 | Type of card at slot 1 | 0: No card<br>1: Programmable card<br>2: I/O card<br>3: Incremental PG card<br>4: Incremental PG card with UVW<br>5: Ethernet communication card<br>6: DP communication card<br>7: Bluetooth card<br>8: Resolver PG card<br>9: CANopen communication card<br>10: WIFI card<br>11: PROFINET communication card<br>12: Sine-cosine PG card without CD | 0–50 | 0 |
| P19.01 | Type of card at slot 2 | | 0–50 | 0 |
| P19.02 | Type of card at slot 3 | | 0–50 | 0 |

| Function code | Name | Parameter description | Setting range | Default value |
|---|---|---|---|---|
| | | signals<br>13: Sine-cosine PG card with CD signals<br>14: Absolute encoder PG card (reserved)<br>15: CAN master/slave communication card<br>16: Modbus TCP communication card<br>17: EtherCAT communication card (reserved)<br>18: BACnet communication card (reserved)<br>19: DeviceNet communication card (reserved)<br>20: PT100/PT1000 temperature detection card<br>21: EtherNet IP communication card<br>22: MECHATROLINK communication card (reserved)<br>23: Bluetooth card 2<br>24–31: Reserved<br>32: SSI-PG card<br>33–50: Reserved | | |

E-mail: overseas@invt.com.cn Website: www.invt.com

The products are owned by **Shenzhen INVT Electric Co.,Ltd.**
Two companies are commissioned to manufacture: (For product code, refer to the 2nd/3rd place of S/N on the name plate.)

**Shenzhen INVT Electric Co.,Ltd.** (origin code: 01)
Address: INVT Guangming Technology Building, Songbai Road, Matian, Guangming District, Shenzhen, China

**INVT Power Electronics (Suzhou) Co.,Ltd.** (origin code: 06)
Address: No. 1 Kunlun Mountain Road, Science & Technology Town, Gaoxin District, Suzhou, Jiangsu, China

Industrial Automation: HMI, PLC, VFD, Servo System, Elevator Intelligent Control System, Rail Transit Traction System

Energy & Power: UPS, DCIM, Solar Inverter, SVG, New Energy Vehicle Powertrain System, New Energy Vehicle Charging System, New Energy Vehicle Motor

66001- 00600



202311 (V1.6)