category	content
key word	PXB-60xxD protocol converter, user manual, product manual
abstract	This document provides customers with instructions on the PXB-60xxD series protocol converter, including product hardware interface introduction, software configuration, etc., to facilitate customers' quick evaluation, application, and product getting started.

Revision history

edition	date	reason
V0.90	2024/02/28	Document release
V0.91	2024/03/18	Adjust the manual format and optimize the manual instructions.
V0.92	2024/08/06	Optimize some functional descriptions

Contents

1. Product Introduction

1.1 Product Introduction

The PXB-60xxD series products are various industrial fieldbus protocol converters such as Modbus to CAN/CAN FD, DeviceNet, OPC UA, etc. launched by Guangzhou Zhiyuan Electronics Co., Ltd. This series of products includes 1 CAN FD, 1 RS485 interface, 1 standard 10/100M Ethernet interface, and 1 USB Type-C interface. It is equipped with a domestically produced high-performance RISC-V processor, which is used for high-speed bidirectional conversion of Modbus and various industrial fieldbus data messages. The PXB-60xxD series protocol converter provides a configuration tool for the upper computer, which can flexibly configure relevant functions and easily achieve seamless conversion of various industrial bus protocol data to Modbus data.

Figure 1.1 Product Diagram of PXB-60xxD Series

1.2 Product Series Description

1.2.1 Naming Rules

The naming convention for PXB protocol converter series products is shown in Figure 1.2.

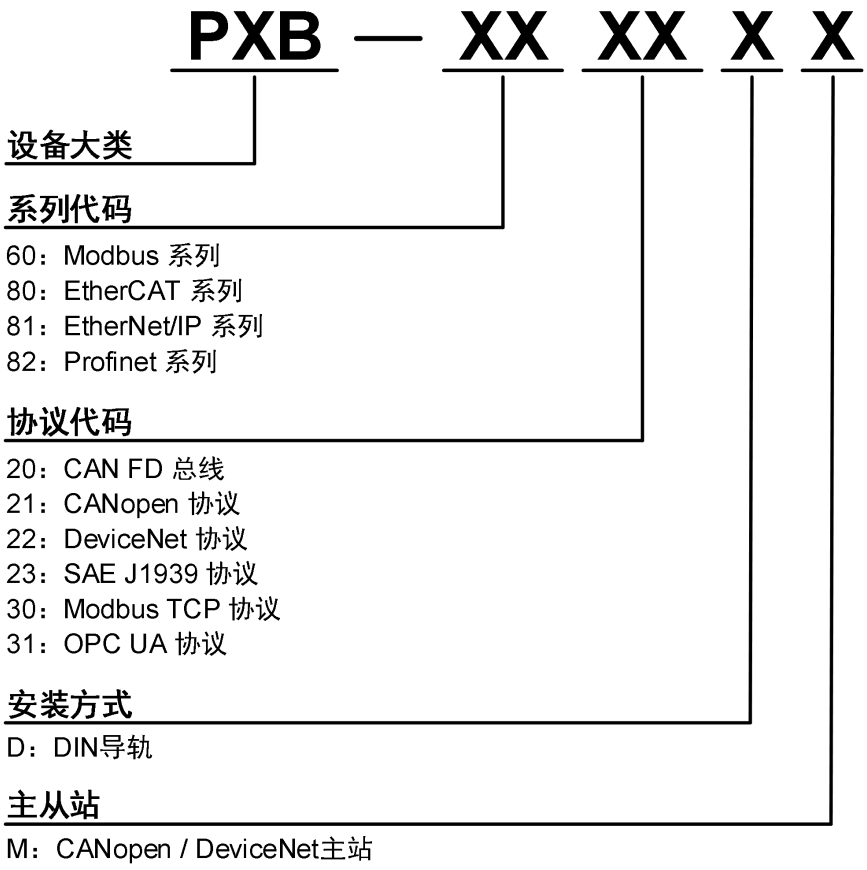

Figure 1.2 Naming Rules for Series Products

1.2.2 Ordering Information

Table 1.1 Ordering Information

Product model	Protocol conversion type	Installation method
PXB-6020D	Modbus <=> CAN/CAN FD	35mm DIN rail
PXB-6021D	Modbus <=> CANopen	35mm DIN rail
PXB-6021DM	Modbus<=>CANopen (master station)	35mm DIN rail
PXB-6022D	Modbus <=> DeviceNet	35mm DIN rail
PXB-6022DM	Modbus<=>DeviceNet (master station)	35mm DIN rail
PXB-6030D	Modbus RTU /ASCII<=> Modbus TCP	35mm DIN rail
PXB-6031D	Modbus <=> OPC UA	35mm DIN rail

1.3 Hardware Features

Table 1.2 Product Hardware Characteristics

input voltage	9 ~ 36VDC,150mA @ 12VDC
Power protection	Anti reverse connection protection, short circuit protection
Automatic restart trigger	Built in independent WDT (watchdog timer)
RS485 isolation	Digital isolation, power isolation
CAN isolation	Digital isolation, power isolation
RS485 baud rate	Up to 2Mbps
CAN FD baud rate	40k~5Mbps, supports CAN FD acceleration
Terminal resistance	Built in 120 ohms (can be configured using upper computer software)
Shell material	Metal
size	125.00mm × 76.00mm × 28.00mm (bare metal)
Installation method	Standard 35mm DIN rail
working temperature	-40 to 85 ° C (-40 to 185 ° F)
Storage temperature (including packaging)	-40 to 85 ° C (-40 to 185 ° F)
relative humidity	5 to 95% (non condensing)
EMI	EN55032，CLASS A
EMC	IEC/EN 61000-4-2 ESD: Contact: 4.0 kV; Air: 8.0 kV IEC/EN 61000-4-4 EFT: Power supply: 1.0 kV; Signal: 0.5 kV IEC/EN 61000-4-5 Surge: Power supply: 1.0 kV; Signal: 0.5 kV IEC/EN 61000-4-6 CS (150 kHz to 80 MHz): Power supply: 3 V/m; Signal: 3 V/m

1.4 Software Features

Table 1.3 Software Features

Product model	Software Features
PXB-6020D	Provide four working modes: Modbus RTU master/slave, Modbus TCP master/slave
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2M baud rate
	Serial port data bit, stop bit, and check bit can all be set
	IP address, slave ID, destination IP, and port can all be set
	Support register types: coil, input status, input register, hold register
	Support setting parameters such as CAN type, CAN FD standard, and CAN FD acceleration
	Support sending and receiving CAN, CAN FD, and CAN FD acceleration messages
	CAN message transmission supports multiple triggering methods such as cycle, status change, and single transmission
	Supports sending 128 sets of messages and receiving 128 sets of whitelist messages
	Support the conversion of data between Modbus and CAN FD buses in the form of whole frames, bytes, bits, etc
	CAN message transmission supports multiple data sources
	Support setting the waiting time for sending
	Support custom sending mode, can customize precise sending of CAN or CAN FD messages
PXB-6021D	Provide two working modes: Modbus RTU/TCP master to CANopen slave
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2M baud rate
	Serial port data bit, stop bit, and check bit can all be set
	IP address, slave ID, destination IP, and port can all be set
	Support register types: coil, input status, input register, hold register
	Support 80 sets of TPDO and 80 sets of RPDO
	Support modifying the Node ID of CANopen and the COB-ID of PDO
	Support modifying CAN baud rate, up to 1Mbps
	Support precise setting of data synchronization time for each TPDO
	Support the conversion of data between Modbus and CANopen protocols in the form of bits, bytes, words, etc
	Can update device EDS files and provide standard EDS files for supporting equipment
PXB-6021DM (CANopen Master Station)	Provide four working modes: Modbus RTU master/slave to CANopen master Modbus TCP master/slave to CANopen master
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2Mbps baud rate
	Serial port data bit, stop bit, and check bit can all be set
	IP address, slave ID, destination IP, and port can all be set
	Support register types: coil, input status, input register, hold register
	Supports 126 CANopen slave devices
	Supports 128 sets of TPDO and 128 sets of RPDO
	Support modifying the Node ID of CANopen master station

Product model	Software Features
PXB-6022D	Support SDO read and write, complete CANopen from device initialization configuration
	Support modifying CAN baud rate, up to 1Mbps
	Support precise setting of data synchronization time for each TPDO
	Support the conversion of data between Modbus and CANopen protocols in the form of bits, bytes, words, etc
	Provide two working modes: Modbus RTU/TCP master to DeviceNet slave
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2M baud rate
	Serial port data bit, stop bit, and check bit can all be set
	IP address, slave ID, destination IP, and port can all be set
	Support register types: coil, input status, input register, hold register
	This device serves as a standard DevicetNet slave and supports polling based I/O connection types
	DeviceNet supports speeds of 125K, 250K, and 500K
	Supports multi-level input and output bytes, with a maximum of 512 bytes for input and 512 bytes for output
	Support mutual conversion of data between Modbus and DeviceNet protocols in the form of bits, bytes, words, etc
	Support DeviceNet I/O scanning
	Support setting data update interval
	Provide standard EDS files for DeviceNet
PXB-6022DM (DeviceNet Master Station)	Provide four working modes: Modbus RTU master/slave to DeviceNet master
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2M baud rate
	Serial port data bit, stop bit, and check bit can all be set
	IP address, slave ID, destination IP, and port can all be set
	Support register types: coil, input status, input register, hold register
	This device serves as a standard DevicetNet master and supports up to 63 DevicetNet slave devices
	DviceNet supports speeds of 125K, 250K, and 500K
	Supports four types of I/O connections: polling, bit gating, cycle, and state change
	Each I/O connection type supports up to 64 bytes of input/output cache
	Support mutual conversion of data between Modbus and DeviceNet protocols in the form of bits, bytes, words, etc
	Support configuring the data update cycle for each DevicetNet slave station
PXB-6030D	Provide four working modes: Modbus RTU Master<=>Modbus TCP Slave Modbus ASCII Master<=>Modbus TCP Slave Modbus TCP Master<=>Modbus RTU Slave Modbus TCP Master<=>Modbus ASCII Slave
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2M baud rate
	Serial port data bit, stop bit, and check bit can all be set
	Work port, target IP, and port can all be set
	Support register types: coil, input status, input register, hold register
	Supporting protocols include: Modbus RTU Modbus ASCII、Modbus TCP、ETHERNET、ARP、 IP、UDP、TCP、DHCP、ICMP

Product model	Software Features
PXB-6031D	Modbus TCP supports up to 8 connections
	Supports up to 255 Modbus RTU/ASCII slaves
	Support setting TCP connection keep alive time and automatically disconnecting abnormal connections
	Support setting reconnection time, automatic reconnection for TCP connection disconnection
	Support device ID mapping function
	Provide two working modes: Modbus RTU master to OPC UA server Modbus TCP Master to OPC UA Server
	The serial port baud rate supports user settings of 2400-2000000bps, with a maximum support of 2M baud rate
	Serial port data bit, stop bit, and check bit can all be set
	Support register types: coil, input status, input register, hold register
	Supports up to 8 OPC UA objects and 2000 total variables
	Support setting the size end format of data and enable word swapping
	Support OPC UA object and variable editing
	Support modifying OPC UA network configuration parameters
	Support data encryption and signature
	Support user access control
	Support rich OPC UA data formats for operation
	Support setting transformation coefficients for easy data observation and calculation
	Support setting data update interval
Other characteristics	Equipped with AWPX Tools configuration software, simple and easy to use
	Support software configuration to enable CAN and RS485 terminal resistors
	Support one click factory reset and multiple guarantees
	Support one click firmware upgrade

2. Hardware Description

2.1 Product Appearance

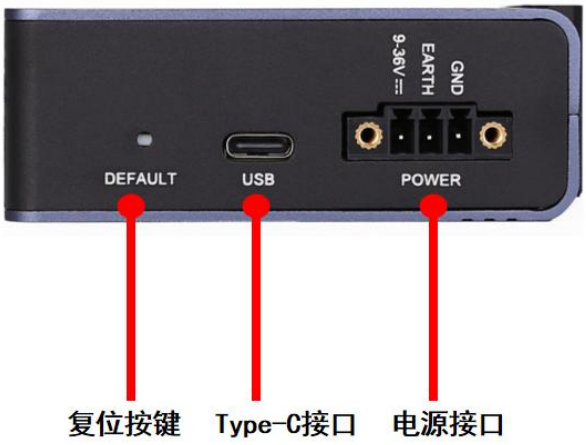

Figure 2.1 Schematic diagram of top interface definition

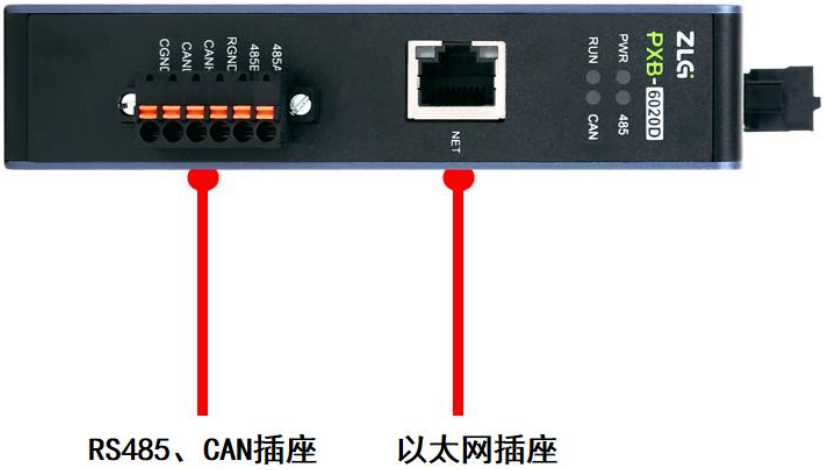

Figure 2.2 Side view of the product

Figure 2.3 Product Front View (Taking PXB-6020D as an Example)

The PXB-60xxD series products have one power interface and one implicit button for resetting to factory settings on the top. The product has one CAN/CAN FD interface and RS485 interface with electrical isolation on the side, one standard 10/100M Ethernet interface, and four LED indicator lights.

Note: For specific LED and interface definitions and usage, please refer to 2.2 Indicator Light Instructions and 2.3 Interface Definitions.

2.2 Indicator light description

The PXB-60xxD series products have four LED lights on the front, with different functional instructions in different product models. They are abbreviated according to relevant professional vocabulary to form corresponding labels, namely PWR, RUN, CAN, 485, MS, and NS. The specific instructions for the indicator lights of PXB-60xxD series products are shown in Table 2.1-2.7:

The indicator lights for PXB-6020D are described in Table 2.1.

Table 2.1 PXB-6020D Indicator Light Status Description

identification	definition	state	explain
PWR	Equipment power indicator light	The red light is always on	Equipment power supply is normal
		Not lit up	The device is not powered on or has abnormal power supply
RUN	Equipment operation indicator light	Green light flashing	The device has entered working mode and is running normally
		The red light is always on	The device is running in Modbus TCP master mode and is not connected to any Modbus TCP slave
		Not lit up	Equipment import configuration error
485	Working mode indicator light	Green light always on	The device operates in Modbus RTU mode (master or slave)
		Not lit up	The device operates in Modbus TCP mode (master or slave)
CAN	CAN communication indicator light	Green light flashing	CAN/CAN FD data transmission and reception are normal
		Red light flashing	CAN/CAN FD data transmission and reception abnormal
		Not lit up	The device is not sending or receiving CAN/CAN FD data

The indicator lights for PXB-6021D are described in Table 2.2.

Table 2.2 PXB-6021D Indicator Light Status Description

identification	definition	state	explain
PWR	Equipment power indicator light	The red light is always on	Equipment power supply is normal
		Not lit up	The device is not powered on or has abnormal power supply
RUN	Equipment operation indicator light	Green light flashing	The device has entered working mode and is running normally
		The red light is always on	The device is running in Modbus TCP master mode and is not connected to any Modbus TCP slave
		Not lit up	Equipment import configuration error
485	Working mode indicator light	Green light always on	The device operates in Modbus RTU mode
		Not lit up	The device is operating in Modbus TCP mode
CAN	CAN communication indicator light	Not lit up	The device has not established a correct connection with the CANopen master station
		Green light always on	The device has established a correct connection with the CANopen master station

The indicator lights of PXB-6021DM are described in Table 2.3.

Table 2.3 PXB-6021DM Indicator Light Status Description

identification	definition	state	explain
PWR	Equipment power indicator light	The red light is always on	Equipment power supply is normal
		Not lit up	The device is not powered on or has abnormal power supply
RUN	Equipment operation indicator light	Green light flashing	The device has entered working mode and is running normally
		The red light is always on	Equipment malfunction
		Not lit up	Equipment import configuration error
485	Modbus indicator light	Green light always on	Modbus network is normal
		The red light is always on	Modbus communication exception
CAN	CAN communication indicator light	The red light is always on	CANopen master station startup failed
		Green light flashing	Network node disconnection
		Green light always on	CANopen network is normal

The indicator lights for PXB-6022D and PXB-6022DM are described in Table 2.4.

After powering on, PXB-6022D and PXB-6022DM devices will perform LED self-test, and users can determine whether the device is running by observing the status of the LED; Firstly, MS self-test: display green for 0.25s ->display red for 0.25s ->display green; Then NS self-test: display green 0.25->display red 0.25 s ->turn off.

Table 2.4 Status Description of PXB-6022D and PXB-6022DM Indicator Lights

identification	definition	state	explain
PWR	Equipment power indicator light	The red light is always on	Equipment power supply is normal
		Not lit up	The device is not powered on or has abnormal power supply
485	Working mode indicator light	Green light always on	The device operates in Modbus RTU mode
		Not lit up	The device is operating in Modbus TCP mode
MS	Node working status indicator light	Not lit up	No power supply to the device
		Green light always on	The equipment is running normally
		Green light flashing	Due to missing, incomplete or incorrect configuration, the device needs to be debugged
		Red light flashing	Recoverable faults
		The red light is always on	Unrecoverable fault, needs to be replaced
		Red and green flashing	The device is self checking
NS	Node network	Not lit up	The device is not online The device has not completed duplicate MAC ID detection and is not powered on

		Green light flashing	The device is online but not connected and is in an established state
		Green light always on	The device is online and has one or more established connections
		Red light flashing	Communication failure: repeated MAC ID detection failed, bus BUS-OFF
		Red and green flashing	The device has detected a network access error and is in a communication failure state, and has received a point-to-point recognition offline fault request message

The indicator lights for PXB-6030D are described in Table 2.5.

Table 2.5 PXB-6030D Indicator Light Status Description

identification	definition	state	explain
PWR	Equipment power indicator light	The red light is always on	Equipment power supply is normal
		Not lit up	The device is not powered on or has abnormal power supply
RUN	Equipment operation indicator light	Green light flashing	The device has entered working mode and is running normally
		The red light is always on	The device is running in Modbus TCP master mode and is not connected to any Modbus TCP slave
		Not lit up	Equipment import configuration error
485	Protocol conversion indicator light	Green light always on	Abnormal device protocol conversion
		Green light flashing	Device protocol conversion successful

The indicator lights for PXB-6031D are described in Table 2.6.

Table 2.6 PXB-6031D Indicator Light Status Description

identification	definition	state	explain
PWR	Equipment power indicator light	The red light is always on	Equipment power supply is normal
		Not lit up	The device is not powered on or has abnormal power supply
RUN	Equipment operation indicator light	Green light flashing	The device has entered working mode and is running normally
		The red light is always on	The device is running in Modbus TCP master mode and is not connected to any Modbus TCP slave
		Not lit up	Equipment import configuration error
485	Working mode indicator light	Green light always on	The device operates in Modbus RTU mode
		Not lit up	The device is operating in Modbus TCP mode

2.3 Interface Definition

The PXB-60xxD series products have one CAN/CAN FD interface and RS485 interface with electrical isolation on the side, one standard 10/100M Ethernet interface, and four LED indicator lights, as shown in Figure 2.4.

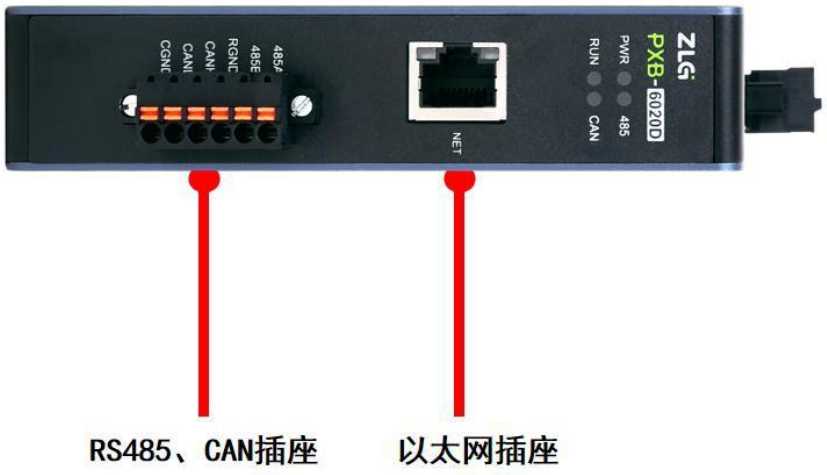

Figure2.4 Schematic diagram of side interface definition

2.3.1 Power interface

The power supply supports a wide voltage input of 9-36V, and customers can choose a power supply within the voltage range to supply power to the equipment according to the on-site environment. The physical form of the power interface connector is an OPEN3 3.08mm spacing lockable socket, and the shell silk screen label is "9V~36V". The power supply requirements for the product are shown in Table 2.7.

Table2.7 Power Interface Input Power Specification

parameter	minimum	typical	maximum	Company
working voltage	nine	12	thirty-six	V
Working current	—	100	—	mA
Product power consumption	—	1.2	—	W

2.3.2 Implicit buttons

Considering that customers may have parameter configuration errors during use, which may result in abnormal product operation, a hidden button is reserved on the top of the product to restore factory settings, and the silk screen label on the shell is "DEFAULT".

During the power on process, pressing and holding the factory reset button will automatically restore the device to its factory settings, but it will not restart; If the product is running and long pressed for 5 seconds, it will automatically reset to factory settings and restart.

After restoring the factory settings, the original configuration parameters inside the device will be cleared.

2.3.3 LED indicator light

The PXB-602xD series products have four LED lights on the front, namely PWR, RUN, CAN, and 485 lights, except for the PXB-6022D and PXB-6022DM models. The PXB-6022D and PXB-

6022DM products have four LED lights on the front, namely PWR, 485, MS, and NS lights.

The PXB-603xD series products have three LED lights on the front, namely PWR, RUN, and 485 lights.

2.3.4 CAN/CAN FD interface

When the product is the PXB-602xD series, there is one CAN/CAN FD interface on the top interface of the product, and the interface identification is shown in Table 2.8:

Table 2.8 CAN/CAN FD Interface Identification Description

identificatio	explain
CANH	Isolate CAN_Signal Line
CANL	Isolate CAN_L signal line
CGND	Isolate CAN ground

When the product is the PXB-603xD series, there is no CAN/CANFD interface on the top of the product.

2.3.5 RS485 interface

The PXB-60xxD series products have one RS485 interface on the side interface, and the interface identification is shown in Table 2.9:

Table 2.9 RS485 Interface Identification Description

identification	explain
485A	Isolate RS485 signal line
485B	Isolate RS485 signal line
RGND	Isolate RS485 ground

2.3.6 Ethernet port

The Ethernet port is labeled as NET and supports standard 10/100M Ethernet communication. This port can not only serve as a Modbus TCP communication interface, but also as a device configuration port. Users can connect to the switch or PC through this network port, and use the matching upper computer software to configure the corresponding parameters such as working mode, data baud rate, and data format of this product.

2.3.7 Terminal Resistance

The PXB-60xxD series products reserve a terminal resistor of 120 ohms for both CAN and RS485 communication ports, and do not connect to the bus by default. Users can configure it using AWPX Tools software, as shown in Figure 2.5-2.6. Users can choose to connect or disconnect the terminal resistor to the bus, which can save the trouble of external series connection or disconnection of resistors.

Figure 2.5 CAN Terminal Resistance Configuration

Figure 2.6 RS485 Terminal Resistance Configuration

3. Instructions for use

3.1 AWPX configuration software

3.1.1 Configuration software acquisition and installation

The PXB series protocol converters are configured through AWPX Tools software (hereinafter referred to as AWPX). The AWPX Tools configuration software can be downloaded by searching for "AWPX" on our official website (www.zlg. cn).https://www.zlg.cn/index.php After downloading, double-click the installation package of AWPX to start installing AWPX. The installation start interface is shown in Figure 3.1.

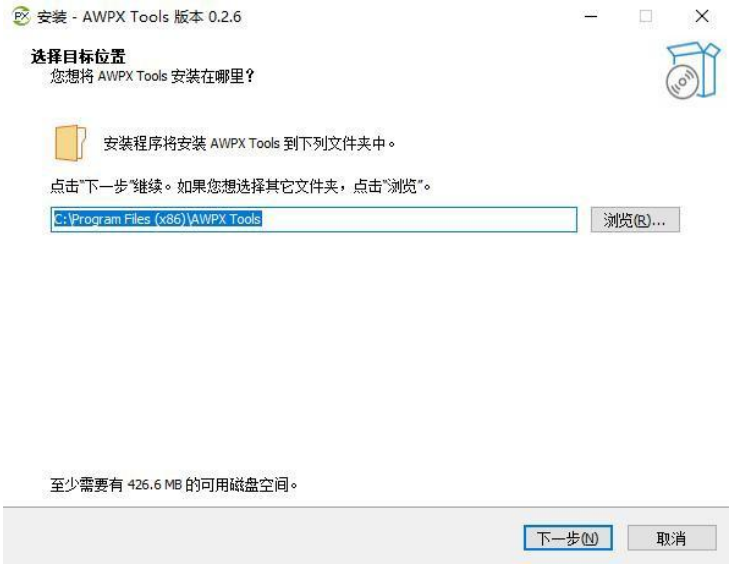

Figure 3.1 Start installing AWPX

After clicking on several 'Next' buttons, the installation interface is shown in Figure 3.2.

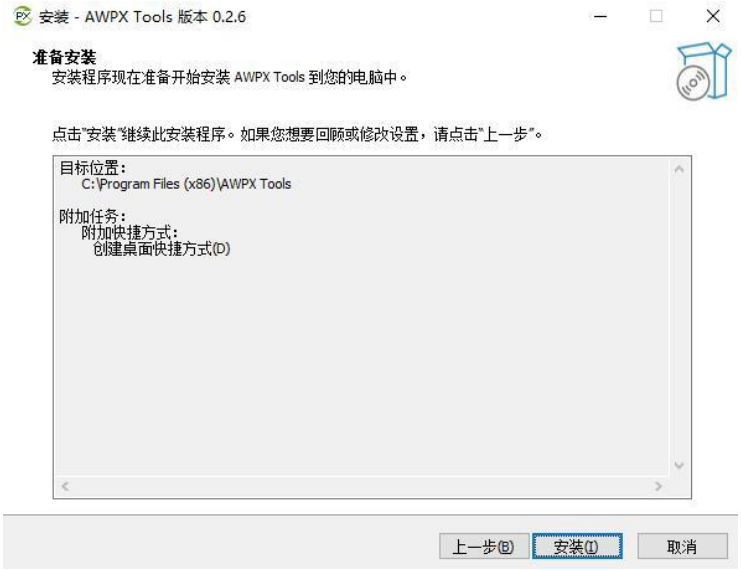

Figure 3.2 AWPX Installation Interface

Finally, click on 'Install' to officially start the installation of AWPX. Please be patient and wait for the installation to complete.

3.1.2 Introduction to Software Configuration Functions

Run the AWPX software, and the interface is shown in Figure 3.3. After powering on the product, connect the PXB-60xxD product to run AWPX.The PC host of the software is configured by connecting to the same LAN via Ethernet cable (through a switch or direct connection).

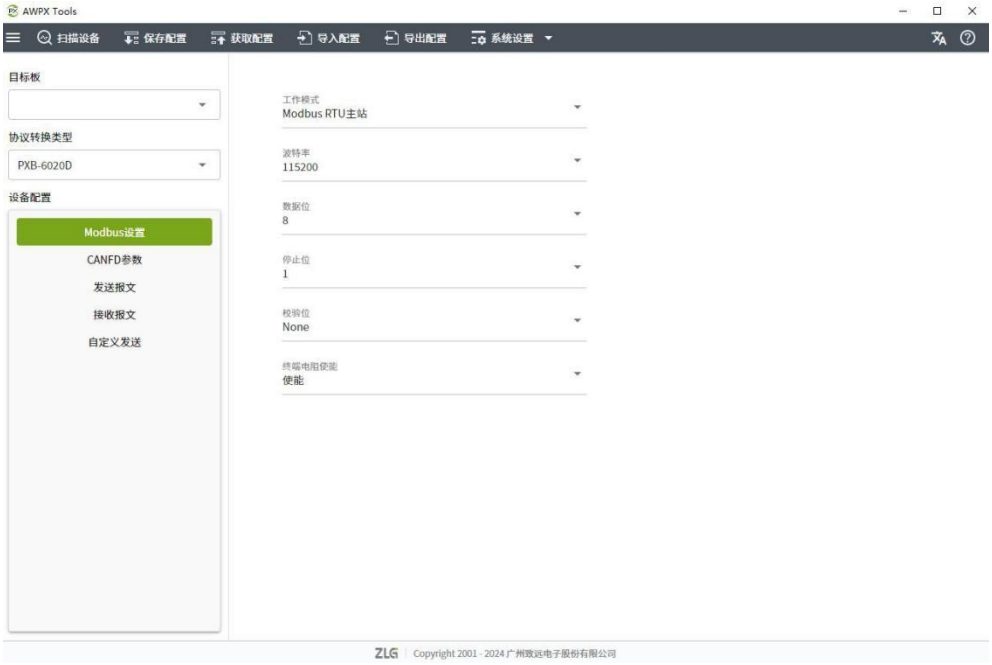

Figure 3.3 AWPX interface

On the left side of the AWPX interface is the device information of the PXB series products, including "Target Board", "Protocol Conversion Type", and "Device Configuration". The specific information of configuration options is located on the right side of the "Device Configuration" column.

The top of the interface is the menu bar button, which includes buttons such as' Scan Device ',' Save Configuration ',' Get Configuration ', etc. As shown in Figure 3.4.

1. Scanning equipment

Click the 'Scan Devices' button, and the AWPX software will search for all PXB series devices in the current local area network, and display the IP address and firmware version of the target board in the' Target Board 'drop-down box, as shown in Figure 3.5. After selecting the correct device in the "Target Board" dropdown menu, AWPX software will automatically load the configuration information of the device and display the corresponding product type in the "Protocol Conversion Type" dropdown menu.

Figure 3.5 Target board and firmware version

2. Save configuration

After modifying any parameter, click the 'Save Configuration' button to send the modified configuration to the PXB series protocol converter, making the modified configuration effective. Saving the configuration will restart the device, wait for the restart prompt at the bottom of the software to disappear.

3. Obtain configuration

After selecting the device, click the 'Get Configuration' button to obtain and display the current configuration of the PXB series protocol converter running.

4. Import configuration

After selecting the device, click the 'Import Configuration' button to import configuration files with. awp or. zip suffixes into AWPX Software. After importing the correct configuration, the imported configuration can be modified ordirectly saved to the device.

Special note: Do not import configurations across versions for use, for example: please export configurations from devices with firmware version 1.1.7.Do not import and save to devices with firmware version 1.1.8 through 'Import Configuration'.

5. Export configuration

After selecting the device, click the 'Export Configuration' button to export the current configuration parameters as a configuration file with the. awp or. zip suffix. So that the next time you use AWPX, you can quickly import and configure the locally saved configuration file by clicking the 'Import Configuration' button.

Special note: The exported configuration only supports devices with the same firmware version, such as those with firmware version 1.1.7.The configuration exported by the device only supports devices with firmware version 1.1.7.

6. System settings

Clicking on the 'System Settings' button will bring up four options:' Network Settings', 'System Information', 'Firmware Upgrade', and 'About'. The system information includes the device ID, type, and other details of the PXB-60xxD device, while the version information related to the AWPX configuration software is provided. Next, we will focus on describing two functions: network settings and firmware upgrade.

Network settings: The default IP address for PXB-60xxD series product devices is "192.168.1.136". If you need to change network parameters such as IP address, you can click the "System Settings" button at the top of the software interface, and then click the "Network Settings" button in the pop-up menu to perform network settings, as shown in Figure 3.6.

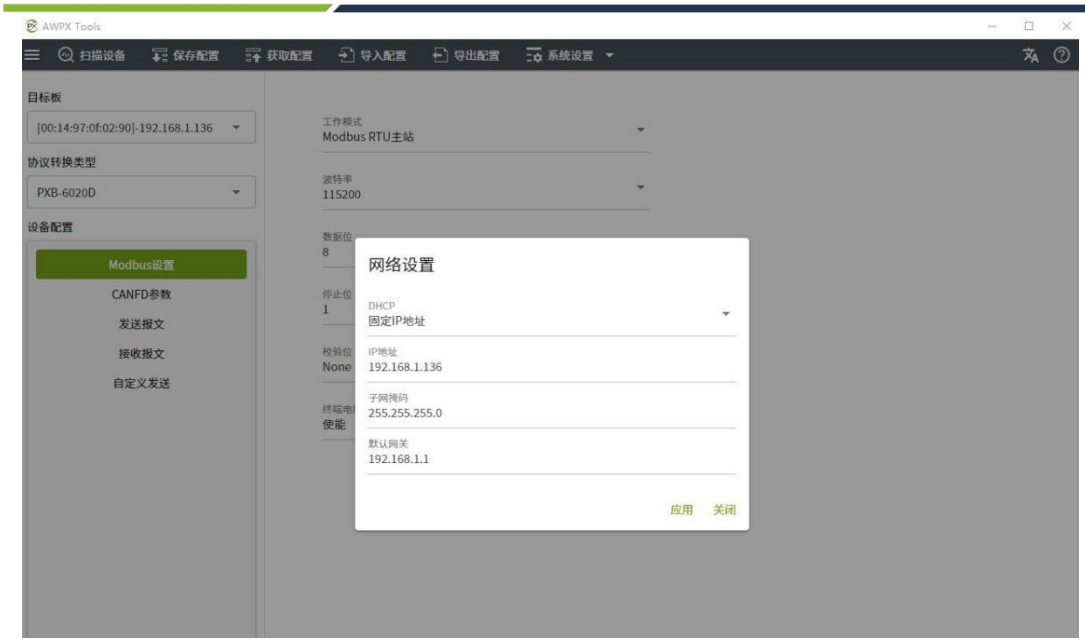

Figure 3.6 Network Setting

Firmware Upgrade: Click on "System Settings" and select "Firmware Upgrade" from the pop-up menu. AWPX will display the upgrade interface, as shown in Figure 3.7.

Figure 3.7 Firmware Upgrade Interface

Click on 'Select Firmware' in the selection box, choose the firmware (bin file) that needs to be upgraded, and click on 'Firmware Upgrade',AWPX will prompt that firmware is being downloaded to the device, and the entire upgrade process will take about 2 minutes.

After about 1 minute, AWPX will prompt that the device firmware download is successful, and the PXB-60xxD product will automatically restart. Please wait patiently for about 1 minute. During this restart process, do not disconnect the power supply of the product. After upgrading the firmware, you need to click again [Scan Device] button, rescan and select the device for configuration.

3.2 Equipment Configuration

The main steps for configuring devices using AWPX configuration software are: Step one, click on 'Scan Devices' and select the correct device. If the device is not scanned and selected, it will not be possible to Configure the PXB-60xxD series protocol converter accordingly;

Step 2, configure parameters. You can configure the parameters in the "Device Configuration" column and network parameters according to your needs; Step 3: After completing the parameter configuration, click [Save Configuration] to save the configured parameters to the device and wait for the device to restart. Just complete it.

3.2.1 PXB-6020D Parameter Configuration

1. Modbus parameter configuration

Click on "Modbus Settings" in the "Device Configuration" column of AWPX software to configure Modbus parameters, including RTU communication parameters such as working mode, baud rate, and check bit. The maximum serial port baud rate can be configured as 2M. The configuration interface is shown in Figure 3.8.

Figure 3.8 Modbus parameter settings

PXB-6020D supports four working modes, which can be selected through the "Working Mode" drop-down list box. Each working mode has corresponding Modbus parameters, and the functional descriptions and corresponding parameter descriptions of these four working modes are shown in Table 3.1.

Table 3.1 Modbus Parameter Description

Working mode	Function Description	Configuration items	Configuration item description
Modbus RTU master station	PXB-6020D works as a Modbus RTU master and supports up to 255 Modbus RTU slaves	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Terminal resistance enable	RS485 terminal resistor enable
Modbus RTU slave station	PXB-6020D works as a Modbus RTU slave, with 2400 built-in coils, input status, input registers, and hold registers, all with addresses ranging from 0 to 2399	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Local slave ID	PXB-6020D serves as the ID for the slave station
		Terminal resistance enable	RS485 terminal resistor enable
Modbus TCP Master	PXB-6020D operates as a Modbus TCP master and serves as a TCP client	Slave IP address	The IP address of the only TCP slave station
		Slave port number	The unique TCP slave port number
		Peer slave number	Unique TCP slave ID
	PXB-6020D operates as a Modbus TCP slave and serves	Local port number	PXB-6020D serves as the port number for the slave station

		Local slave number	PXB-6020D serves as the ID for the slave station

2. CAN FD parameter configuration

PXB-6020D can send CAN or CAN FD messages in any working mode. Click on 'CAN FD Parameters' in the' Device Configuration 'column of AWPX software to configure CAN FD parameters, as shown in Figure 3.9.

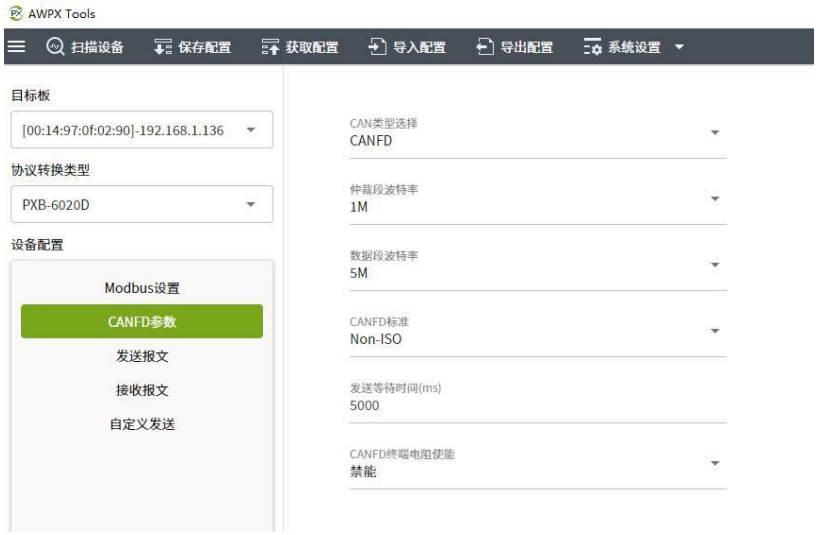

Figure 3.9 CAN FD Parameter Interface

The CAN FD parameters are described in Table 3.2.

Table 3.2 Description of CAN FD Parameters

parameter	Parameter Description
CAN type selection	The type selection can be CAN or CAN FD
Arbitration segment baud rate	Set the baud rate of the arbitration segment for CAN or CAN FD messages
Data segment baud rate	Optional when CAN type is CAN FD, set the data segment baud rate for CAN FD messages
CAN FD standard	Optional when CAN type is CAN FD, set the standard used for CAN FD messages
Sending waiting time	Used to set how long to wait before sending CAN messages after PXB-6020D power on initialization is completed
CANFD terminal resistor enable	Enable or disable the terminal resistance of CAN FD interface

3. Sending message configuration

PXB-6020D can send CAN or CAN FD messages in any working mode. Click on 'Send Message' in the 'Device Configuration' column of AWPX software to configure the parameters for sending messages. Whether the message sent is a CAN frame or a CAN FD frame depends on the 'CAN Type' setting in the 'Send Message' interface. The message sending interface is shown in Figure 3.10.

Figure 3.10 Message Sending Interface

Add up to 128 sent messages by clicking the 'Add+' button in the upper right corner of the interface.Click the [+Add Data] button to add variables for mapping data, with a maximum of 64 variables added per message. Can be accessed through the interface.The 'Delete' button on the right deletes the corresponding message and variable. The parameter description of the message sending interface is shown in Table 3.3.

Table 3.3 Explanation of Message Sending Interface Parameters

parameter	Parameter Description
Message Name	The name of this message can be used for mnemonic purposes
Frame ID	The frame ID for sending messages can be in decimal or hexadecimal format (starting with 0x)
Frame type	The frame type for sending messages can be standard frames or extended frames
Remote frame	Whether the sent frame is a remote frame, this option is invalid when the CAN type is selected as CAN FD
CAN type	Set the type of CAN message to be sent, which can be related to the "CAN Type Selection" option. When the "CAN Type Selection" is CAN, the CAN type here can only be "CAN"; When the 'CAN Type Selection' is CAN FD, 'CAN', 'CAN FD', and 'CANED Acceleration' can be selected here
Data length	The length of the transmission frame data segment is up to 8 bytes for CAN frames and 64 bytes for CAN FD frames
Trigger mode	The triggering mode for PXB-6020D to send CAN messages includes four modes: "periodic sending", "change sending", "single sending", and "frame ID triggering"
Trigger frame ID	The CAN message ID that triggers PXB-6020D to send CAN messages is valid when the trigger mode is selected as [Frame ID trigger]. Can be decimal or hexadecimal (starting with 0x)
Trigger frame type	The CAN message type that triggers PXB-6020D to send CAN messages is valid when the trigger mode is selected as [Frame ID trigger]
Cycle time	When the triggering mode is' periodic sending ', this parameter is the cycle time; When the triggering method is' Change Send ', this time is the cycle for checking Modbus data changes; When the triggering mode is [Single Send], this time is the waiting time for that single send
Variable Name	The name of this variable can be used for mnemonic purposes
Operation size	The size of the mapped data. Contains "Whole Frame Data", "BIT", "BYTE", "WORD", "WORD", and "QWORD". Among them: BYTE is 1 byte, WORD is 2 bytes, WORD is 4 bytes, QWORD is 8 bytes
Offset amount	Select which byte or bit of the CAN message data segment to start from, and sequentially convert the fixed data or Modbus Register data is mapped to CAN message data segments. When the operation size is the entire frame of data, the offset is invalid
data source	The source of the CAN message data segment includes two options: "Modbus" and "Fixed Data"
Data value	The value of the custom CAN message data segment can be decimal or hexadecimal (starting with 0x), and is valid when the data source is selected as' Fixed Data '
Modbus byte order	Modbus data storage method (big end or small end)
Register type	Supports coil, input status, input register, hold register, options related to [operation size]
From station number	When the Modbus working mode is RTU master, the user sets the corresponding RTU slave station number for the data

Register address	The sent message data is at the starting address of the device or Modbus slave's register, and the data source is selected as Valid for Modbus. Can be decimal or hexadecimal (starting with 0x)

The data segment for sending CAN/CAN FD messages on PXB-6020D can be a custom value or a Modbus register value. Custom Value: After selecting "Fixed Data" as the "Data Source" option, set the "Data Value" option. Modbus Register Value: After selecting the "Data Source" option as "Modbus", set the "Register Address".Options are sufficient. PXB-6020D will read the register data corresponding to the Modbus slave address and map the data to CAN.The data segment of the message. Finally, PXB-6020D sends CAN messages to the CAN bus.

If PXB-6020D operates in Modbus master mode, the data segment for sending CAN messages is sourced from the registers of the external Modbus slave station. If PXB-6020D is running in Modbus slave mode, the data segment for sending CAN messages comes from the Modbus register inside PXB-6020D.

The modes for triggering PXB-6020D to send CAN/CAN FD messages are:

1. Periodic sending: PXB-6020D will cyclically send CAN messages based on the cycle time.

2. Change sending: When a change in the value of the configured Modbus register is detected, PXB-6020D is triggered to send a CAN message, and the "cycle time" at this time is the Modbus detection cycle.

3. Single transmission: Send one CAN message after the device is started. This frame is sent at the time of "sending wait time+cycle time" after the device is started.

4. Frame ID trigger: When PXB-6020D receives a CAN message that matches the set 【Trigger ID】 and 【Trigger Frame Type】, it triggers PXB-6020D to send a CAN message.

Example of message sending configuration: Set the Modbus byte order to 'small end' and perform the message sending configuration as shown in Table 3.4 Set.

Table 3.4 Example of Message Sending Configuration

Frame ID	Frame type	Remote frame	CAN type	Data length	Trigger mode	Cycle time	Operation size	Offset amount	Data source	Register type	Register address
0x01	Standard frame	no	CAN	eight	Periodic sending	1000	DWORD	one	Modbus	Maintain register	0

Then PXB-6020D will start from address 0 and sequentially read the Modbus slave hold register data in the size of DWORs. Then, based on the offset of 1, starting from the first byte of the CAN message data segment, the read hold register data of the size of the WORD is sequentially mapped to the data segment of the CAN message. Finally, PXB-6020D sends the CAN message to the CAN bus.

If the corresponding Modbus slave holds register data as: 0 Address: 0x1122, 1 Address: 0x3344, then every interval 1000ms, PXB-6020D will send CAN standard frame with ID 0x01: 00 22 11 44 33 00 00 00 (hexadecimal).

4. Receive message configuration

PXB-6020D can receive CAN or CAN FD messages in any working mode. Whether the received message is a CAN frame or a CAN FD frame depends on the setting of the CAN type selection in the CAN FD parameters interface.

When receiving CAN frames, the maximum size is 8 bytes; When receiving CAN FD frames, the maximum size is 64 bytes.

Click the 'Receive Message' button in the 'Device Configuration' column of AWPX software to configure the parameters for receiving messages, as shown in Figure 3.11.

Figure 3.11 Receiving Message Interface

Add the messages to be received by clicking the 'Add+' button in the upper right corner of the interface, up to a maximum of 128 received messages can be added.The text. Click the [+Add Data] button to add variables for mapping data, with a maximum of 64 variables added per message. If you need to delete the received message or variable, you can use the [Delete] button on the right side of the interface to delete it. The parameter description of the received message interface is shown in Table 3.5.

Table 3.5 Description of Interface Parameters for Receiving Messages

parameter	Parameter Description
Message Name	The name of this message can be used for mnemonic purposes
Frame ID	The frame ID of the received message can be decimal or hexadecimal (starting with 0x)
Frame type	Is the received message a standard frame or an extended frame
Variable Name	The name of this variable can be used for mnemonic purposes
Operation size	The size of the mapped data. Contains "Whole Frame Data", "BIT", "BYTE", "WORD", "WORD", and "QWORD". Among them, BYTE is 1 byte, WORD is 2 bytes, WORD is 4 bytes, and QWORD is 8 bytes.
Offset amount	Select which byte or bit of the CAN message data segment to start from, and map the received CAN message data segment to the register of the Modbus slave station. When the operation size is the entire frame of data, the offset is invalid
Register type	Support coils and hold registers, with options related to [operation size]
From station number	When Modbus is working on the RTU master, this option is used to set the access slave ID
Modbus byte order	Modbus data storage method (big end or small end)
Register address	The received CAN message data segment is stored at the starting address of the local or Modbus slave register, which can be decimal or hexadecimal (starting with 0x)

PXB-6020D will write the received CAN/CAN FD message data segment content into the register of the Modbus slave station. If PXB-6020D is running in Modbus master mode, the received CAN message data segment content will be written to the external device.Modbus slave register. If PXB-6020D is running in Modbus slave mode, the received CAN message data segment content will be written into the Modbus register inside PXB-6020D.

Example of receiving message configuration: Set the Modbus byte order to [small end] and configure the sending message as shown in Table 3.6 Set.

Table 3.6 Example of Message Reception Configuration

Frame ID	Frame type	Operation size	Offset amount	Register type	From station number	Register address
0x02	Standard frame	DWORD	2	Maintain register	one	0x10

When PXB-6020D receives a CAN standard frame with a frame ID of 0x02 and a frame data segment of 11 22 33 44 55 66 77 88 (hexadecimal), PXB-6020D will write the contents of the CAN message data segment, starting from the second byte of the CAN frame data segment, into the hold register corresponding to the Modbus slave address based on offset 2.

Namely: Write data 0x4433 to the hold register with address 0x10 and write data 0x6655 to the hold register with address 0x11 in the Modbus slave with ID 1.

5. Customize sending configuration

Click on 'Custom Send' in the 'Device Configuration' column of AWPX software to configure the parameters for custom message sending, which can be customized to send CAN or CAN FD messages. The interface is shown in Figure 3.12.

Figure 3.12 Custom Sending Interface

Among them, [Slave ID] is the Modbus Slave ID, which can be set when the working mode is Modbus RTU Master. The Modbus Register Address option is used to configure the data source for custom message sending, which is located in the hold register of the corresponding Modbus slave station.

If PXB-6020D is running in Modbus master mode, the custom data source for sending messages is in the hold register of the external Modbus slave. If PXB-6020D is running in Modbus slave mode, the custom data source for sending messages is stored in the internal hold register of PXB-6020D.

Click the 'Custom Send' button to enable custom sending, and then fill in the data format shown in Table 3.7 in the hold register corresponding to the Modbus slave address to customize the data and format of the sent message.

Table 3.7 CAN/CAN FD Data Area Format

field	Number of registers	Sub item	describe
Transaction Number	one	/	A value greater than 0 indicates that the following data area is valid and needs to be incremented for each update sent When the value of this serial number reaches 65535, it can return to 1
CANID	two	/	0~28 is effective
Frame information	one	identification	The lower 8 bits are defined as follows: B0: Value is 1: CANFD frame, otherwise it is CAN frame B1: Value 1: CANFD acceleration is turned on, otherwise CANFD acceleration is turned off B2: Value 1: Remote frame, otherwise it is data frame B3: Value 1: Expand the frame, otherwise reserve other bits for the standard frame
		Data length	High 8 digits
CAN/CAN FD data	32 or 4	/	When working in CAN mode, the length is 4, otherwise it is 32

Example of custom sending: Enable custom sending, set 【 Slave Number 】 to 1, and 【 Modbus Register Address 】 to 30.

Then, in the Modbus slave with address 1, fill in the hold registers of addresses 30-37 in

sequence with 0x0001, 0x0000, 0x0123, 0x0800, 0x1122, 0x3344, 0x5566, and 0x7788. When the value of the hold register with address 30 is passed.Once added, PXB-6020D sends out a CAN standard frame with ID 0x0123 and data segment 22 11 44 33 66 55 88 77 (hexadecimal).

3.2.2 PXB-6021D Parameter Configuration

1. Modbus parameter configuration

Click on "Modbus Settings" in the "Device Configuration" column of AWPX software to configure Modbus parameters. The configuration interface is shown in Figure 3.13.

Figure 3.13 Modbus Settings

PXB-6021D supports two working modes, which can be selected through the "Working Mode" drop-down list box. Each working mode has corresponding Modbus parameters, and the functional descriptions and corresponding parameter descriptions of these two working modes are shown in Table 3.8.

Table 3.8 Modbus Parameter Description

Working mode	Function Description	Configuration items	Configuration item description
Modbus RTU master station	PXB-6021D works as a Modbus RTU master and supports up to 255 Modbus RTU slaves	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Terminal resistance enable	RS485 terminal resistor enable
Modbus TCP Master	PXB-6021D works as a Modbus TCP master and serves as a TCP client	Slave IP address	The IP address of the only TCP slave station
		Slave port	The unique TCP slave port number

2. CANopen parameter configuration

In any working mode, PXB-6021D operates as a CANopen slave on the CANopen side. Click on 'CANopen Parameters' in the' Device Configuration 'column of AWPX software to configure the CANopen parameters of PXB-6021D. The configuration interface is shown in Figure 3.14.

Figure 3.14 CANopen Parameters

The parameter description of CANopen is shown in Table 3.9.

Table 3.9 CANopen Parameter Description

parameter	Parameter Description
Baud rate	Set the baud rate of CAN between 10Kbps and 1Mbps
Node ID	Set PXB-6021D as the node ID for CANopen slave station, between 1 and 127
CAN terminal resistor enable	Enable or disable the terminal resistance of CAN interface

3. RPDO parameter configuration

After receiving the RPDO sent by the CANopen master, PXB-6021D writes the RPDO data field content into the Modbus slave register. Set the RPDO parameter for CANopen as shown in Figure 3.15.

Figure 3.15 RPDO parameter interface

Click the 'Add PDO+' button to add the RPDO mapping entry that needs to be operated on. Click [+Add Field] again to add which field of this RPDO needs to be operated on, and any bit or byte field of this RPDO can be operated on.

On the far right side of the entry, click the [Delete] button to delete the entry. Up to 80 RPDOs can be added, and each RPDO can add up to 16 fields.

The RPDO parameter settings for CANopen are shown in Table 3.10.

Table 3.10 RPDO parameter settings for CANopen

parameter	Parameter Description
PDO index	Set PDO mapping index between 1-80
COBID	Set the communication object number of PDO (the message frame ID of this communication object)
Operation unit	The size of the mapped data. BYTE: 1 byte, WORD: 2 bytes, DWORD: 4 bytes, QWORD: 8 bytes
Byte offset	Set the byte offset of PDO between 0-7 (choose which byte of PDO to start mapping data from)
Positional offset	The bit offset after setting the byte offset of PDO (selecting which byte and bit of PDO to start mapping data from) is valid when the operation unit is BIT
Modbus byte order	Set the Modbus big and small end mode, and this field is valid when the operating unit is BYTE, WORD, or WORD. For example: When the operating unit is BYTE, the RPDO data for CANopen is 0x10: Big end mode: Modbus register data mapped to 0x0010 Small end mode: Modbus register data mapped to 0x1000 When the operating unit is WORD, the RPDO data for CANopen is 0x10, 0x20: Big end mode: Modbus register data mapped to 0x2010 Small end mode: Modbus register data mapped to 0x1020 When the operation unit is a WORD, the RPDO data for CANopen is 0x10, 0x20, 0x30, 0x40: Big end mode: Modbus register data mapping to 0x4030, 0x2010 Small end mode: Modbus register data mapping to 0x3040, 0x1020
Modbus Slave ID	Set Modbus Slave ID
Modbus type	Modbus register type. When the operating unit is BIT, the Modbus type can select coil state, and when it is other operating units, the Modbus type can select hold register
address	Update RPDO data to Modbus slave register address, decimal or hexadecimal (starting with 0x)

4. TPDO parameter configuration

After reading the data from the corresponding address register of the Modbus slave station, PXB-6021D maps the read Modbus register data to the TPDO data field and sends it to the CANopen master station. Set the TPDO parameters for CANopen as shown in Figure 3.16.

Figure 3.16 TPDO parameter interface

Like the RPDO parameter settings, click the [Add PDO+] button to add the TPDO mapping entry that needs to be operated on. Then click the [+Add Field] button to add which field of this TPDO needs to be operated on, and any bit or byte field of this TPDO can be operated on.

On the far right side of the entry, click the [Delete] button to delete the entry. Up to 80 TPDOs can be added, and each TPDO can add up to 16 fields.

The TPDO parameter settings for CANopen are shown in Table 3.11.

Table 3.11 TPDO parameter settings for CANopen

parameter	Parameter Description
PDO index	Set PDO mapping index between 1-80
PDO synchronization period	Set the time interval in milliseconds between each update of Modbus slave register data to TPDO
COBID	Set the communication object number of PDO (the message frame ID of this communication object)
Operation unit	The size of the mapped data. BYTE: 1 byte, WORD: 2 bytes, DWORD: 4 bytes, QWORD: 8 bytes
Byte offset	Set the byte offset of PDO between 0-7 (choose which byte of PDO to start mapping data from)
Positional offset	The bit offset after setting the byte offset of PDO (selecting which byte and bit of PDO to start mapping data from) is valid when the operation unit is BIT
Modbus byte order	Set the Modbus big and small end mode, and this field is valid when the operating unit is BYTE, WORD, or WORD. For example: When the operating unit is BYTE, the Modbus register data is 0x10: Big end mode: The TPDO data of CANopen is mapped to 0x10. Small end mode: The TPDO data of CANopen is mapped to 0x00 When the operating unit is WORD, the Modbus register data is 0x1020: Big end mode: The TPDO data of CANopen is mapped to 0x20, 0x10 Small end mode: The TPDO data of CANopen is mapped to 0x10, 0x20 When the operating unit is a WORD, the Modbus register data is 0x1020, 0x3040: Big end mode: The TPDO data mapping for CANopen is 0x10, 0x30, 0x20, 0x10. Small end mode: The TPDO data mapping for CANopen is 0x10, 0x20, 0x30, 0x40
Modbus Slave ID	Set Modbus Slave ID
Modbus type	Modbus register type. When the operating unit is BIT, the Modbus type can be selected as either coil state or input state. When used for other operating units, the Modbus type can choose to hold registers or input registers
address	Update Modbus register data to the register address of TPDO. Decimal or hexadecimal (starting with 0x)

3.2.3 PXB-6021DM Parameter Configuration

1. Modbus parameter configuration

Click on "Modbus Settings" in the "Device Configuration" column of AWPX software to configure Modbus parameters. The configuration interface is shown in Figure 3.17.

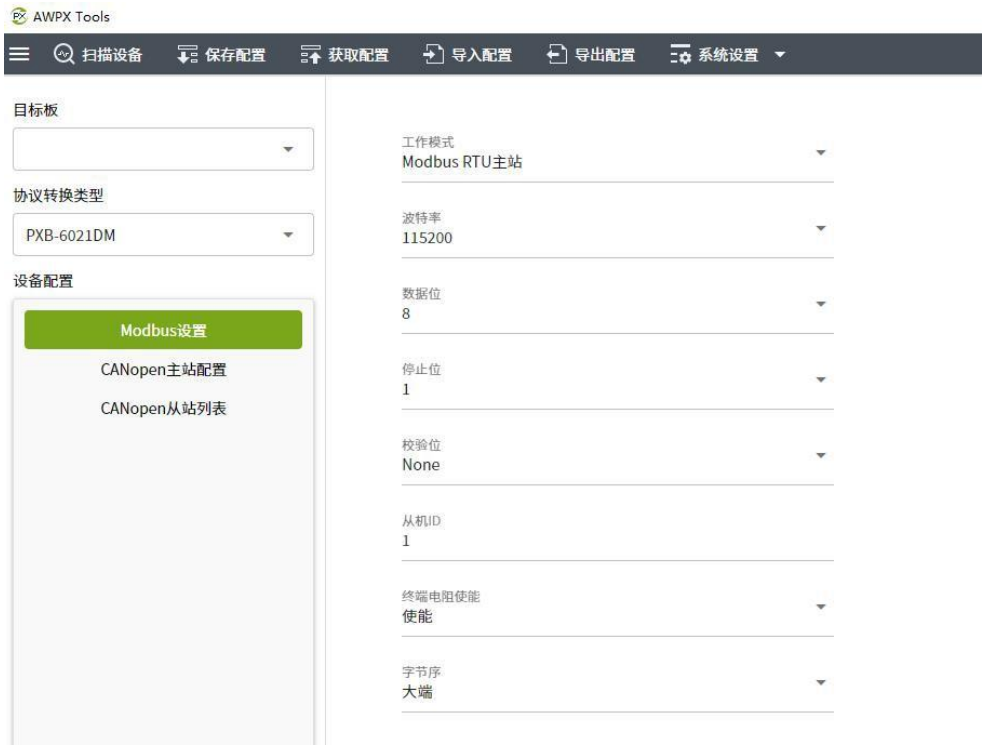

Figure 3.17 Modbus parameter settings

PXB-6021DM supports four working modes, which can be selected through the "Working Mode" drop-down list box. The functional descriptions and corresponding parameter descriptions of these four working modes are shown in Table 3.12.

Table 3.12 Modbus Parameter Description

Working mode	Function Description	Configuration items	Configuration item description
Modbus RTU master station	PXB-6021DM operates as a Modbus RTU master station and can be externally connected to a unique Modbus RTU slave station	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Slave ID	Unique external slave ID
		Terminal resistance	RS485 terminal resistor enable
		Byte order	Modbus data storage method
Modbus RTU slave station	PXB-6021DM works as a Modbus RTU slave, and the external Modbus RTU master can read and write the PXB-6021DM register	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Local slave ID	PXB-6021DM as the ID of the slave station
		Terminal resistance enable	RS485 terminal resistor enable
		Byte order	Modbus data storage method

Modbus TCP Master	PXB-6021DM operates as a Modbus TCP master and serves as a TCP client	Slave IP address	The IP address of the only TCP slave station
		Slave port number	The unique TCP slave port number
		Peer slave number	Unique TCP slave ID
		Byte order	Modbus data storage method

Continued

Working mode	Function Description	Configuration items	Configuration item description
Modbus TCP Slave	PXB-6021DM operates on Modbus TCP slave, acting as a TCP server. External Modbus TCP master readable and writable PXB-6021DM Register	Local port number	PXB-6021DM serves as the port number for the slave station
		Local slave number	PXB-6021DM as the ID of the slave station
		Byte order	Modbus data storage method

2. CANopen parameter settings

PXB-6021DM operates in CANopen master mode on both sides. Simply click on 'CANopen Master Station Configuration' in the 'Device Configuration' column of the AWPX software to configure the PXB-6021DM master station parameters. The configuration interface is shown in Figure 3.18.

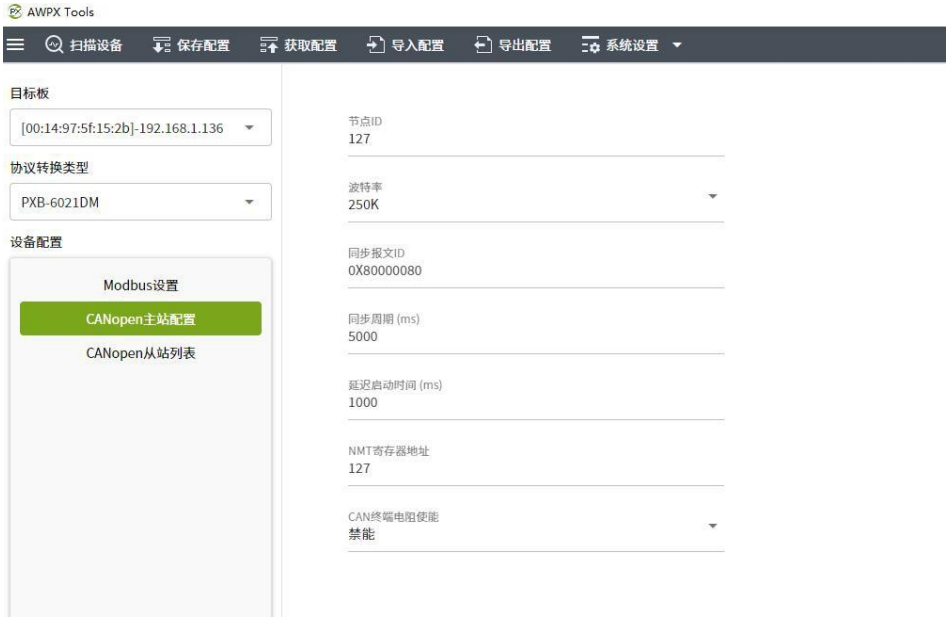

Figure 3.18 CANopen Master Station Parameters

The CANopen master station parameters are described in Table 3.13.

Table 3.13 CANopen Master Station Parameter Description

parameter	Parameter Description
Node ID	Set PXB-6021DM as the CANopen network node ID between 1 and 127
Baud rate	Set the baud rate of the CAN bus between 10Kbps and 1Mbps
Synchronize message ID	Default is 0x80000080 (hexadecimal needs to be prefixed with 0x, otherwise it is decimal)
Synchronization period	Default is 0 (0~65535) ms
Delay start time	Is the main station delayed in starting (0~2147483647) ms
NMT register address	Modbus can control the status of CANopen nodes by modifying registers, with register values up to eight digits high For CMD, the lower eight bits represent the Node Node-ID. CMD enumeration includes: 0x01=Node enters operational state; 0x02=Node enters a stopped state 0x80=Node enters pre operation state; 0x81=Reset application layer

	0x82=Reset node communication; If ID=0, it means all nodes
CAN terminal resistor enable	Enable or disable the terminal resistance of CAN interface

3. CANopen Slave List

Complete the configuration of CANopen nodes from the list of slave stations. Click on "Add Slave Station" or "Delete Slave Station" to add or delete the current slave station. The interface is shown in Figure 3.19.

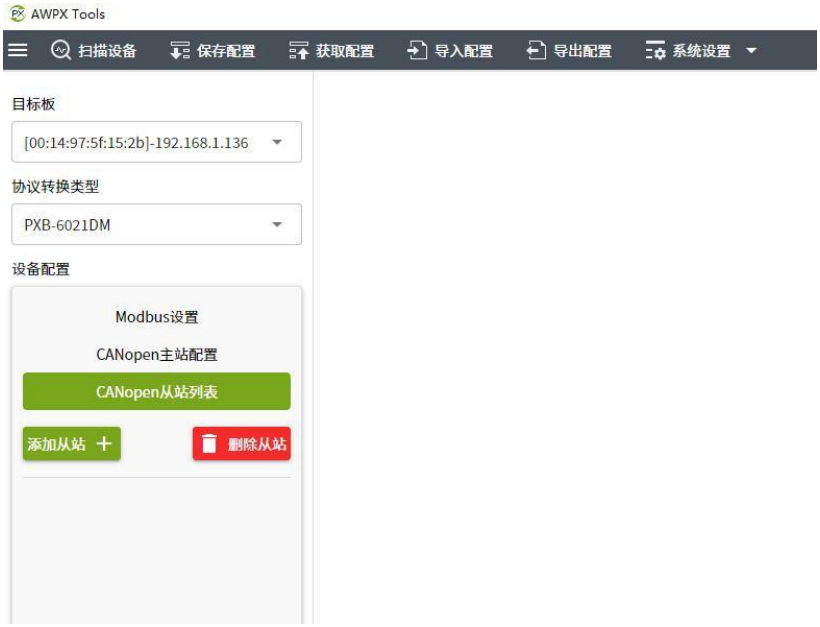

Figure 3.19 CANopen Slave List Interface

4. CANopen Slave Properties

Click on 'Add Slave' to display the Slave Configuration interface, which is used to configure the relevant attribute information of the slave. The interface is shown in Figure 3.20.As shown.

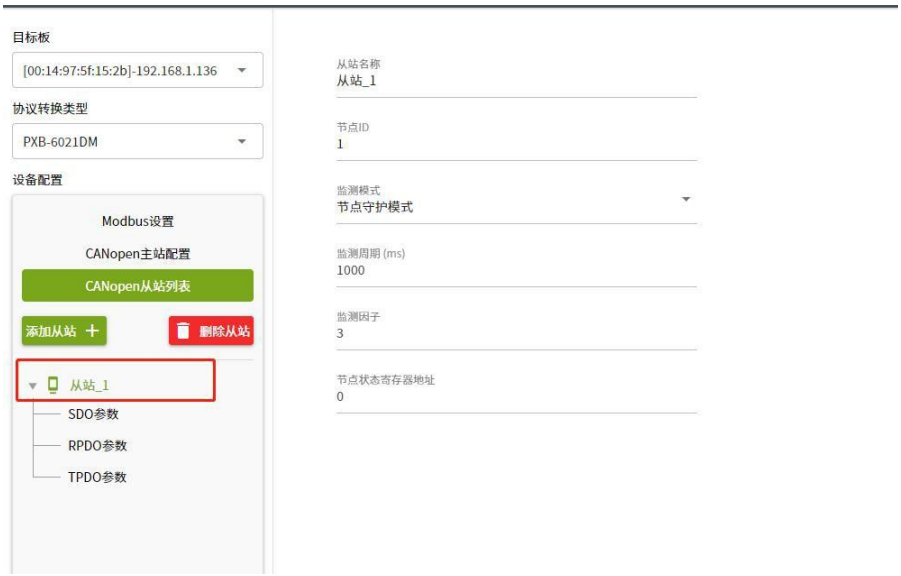

Figure 3.20 CANopen Slave Attribute Interface

The definition of its setting parameters is as follows:

Table 3.14 CANopen Slave Parameter Description

parameter	Parameter Description
Station name	From station name, user-defined
Node ID	Slave node ID (1~127)
Monitoring mode	Slave node online monitoring mode, default to heartbeat protocol
Monitoring cycle	Default 1000 (0~65535) ms
Monitoring factors	Only valid when the detection mode is node guarding
Node Status Register	Mapping CANopen network node status to Modbus registers, 0x01 indicates that the node is offline, 0x04 indicates that the node is in a stopped state, 0x05 indicates that the node is in working state (online), 0x7F indicates that the node is in a pre operation state

5. SDO parameters

SDO parameters are used to configure the initialization startup parameters of the slave station. By selecting the data source, the node dictionary content can be flexibly modified to meet the different usage scenarios of CANopen network adaptation. The interface is shown in Figure 3.21:

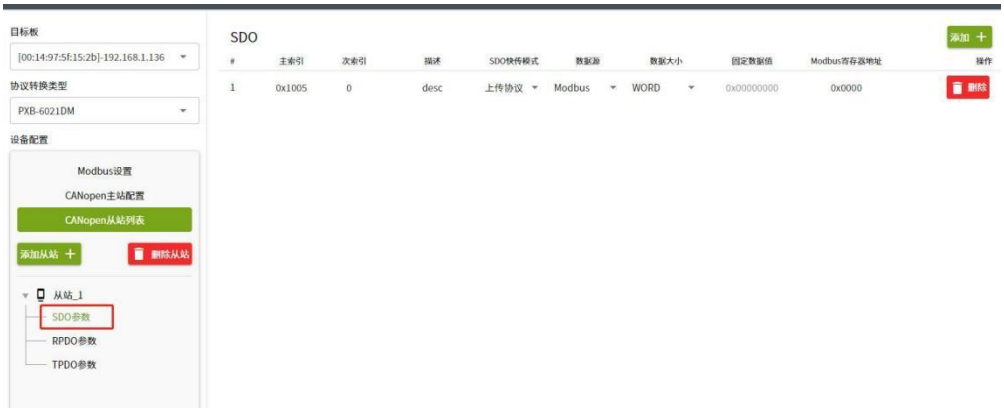

Figure 3.21 SDO Parameter Interface

The definition of its setting parameters is as follows:

Table 3.15 SDO Parameter Description

parameter	Parameter Description
Main index	Slave dictionary master index
Secondary Index	Slave dictionary index
describe	User defined description content for mnemonic purposes
SDO fast transfer mode	Upload or download
data source	Fixed value or Modbus register
data size	The maximum data size for this SDO fast transfer is four bytes
Fixed data value	Effective when the data source is a fixed value (Hexadecimal requires a prefix of 0x, otherwise it is decimal)
Modbus register address	Effective when the data source is Modbus (Hexadecimal requires a prefix of 0x, otherwise it is decimal)

6. RPDO parameters

RPDO parameters are used to configure the RPDO parameters of the slave station and the mapping rules between Modbus data and CANopen network. The interface is shown in Figure 3.22:

Figure 3.22 RPDO parameter interface

The definition of its setting parameters is as follows:

Table 3.16 RPDO Parameter Description

parameter	Parameter Description
PDO index	The RPDO communication object from the station dictionary is offset based on the 0X1400 primary index. The interface configuration value will be reduced by 1 during actual transmission. When the PDO index is configured as 1, the actual modified parameters are: 0X1400 + pdo_index - 1 = 0X1400
PDO synchronization period	Modbus to CANopen network data mapping cycle (0~65535) ms
COBID	COBID of the slave station (hexadecimal needs to be prefixed with 0x, otherwise it is decimal)
Mapping description	Used to describe the mapping function, user-defined
Operation unit	Describe the size of the data range mapped from a CAN message to Modbus. There are five options: BIT, BYTE, WORD, WORD, and QWORD. When the mapping data size exceeds one Modbus register data size, that is, when the current configuration is DWORD or QWORD, the device will automatically expand the register range to adapt to this mapping. For example, if the current configuration is a WORD and the Modbus register address is 20000, then Modbus's 20000 The data at the 20001 register address will be mapped to the corresponding CAN
Byte offset	The starting byte offset of the Modbus register mapping to the CAN frame (eight bytes) this time
Positional offset	BIT offset within one byte, only effective when the operating unit is BIT
data source	Fixed value or Modbus network
Modbus register address	Modbus register address (hexadecimal needs to be prefixed with 0x, otherwise it is decimal)
Modbus Register Types	Modbus register type (hexadecimal needs to be prefixed with 0x, otherwise it is decimal)
Fixed value	Specific fixed values (hexadecimal needs to be prefixed with 0x, otherwise decimal) are transmitted in small endian byte order and mapping rules over the CANopen network

Table 3.17 shows the actual data mapped from Modbus to CANopen network: The mapping rule is described as: byte offset is 0, bit offset is 0.

Table 3.17 RPDO Mapping Table

Operation unit	Modbus register values	CAN message data
BIT	one	Can[0] = 0x01;
BYTE	0x0001	Can[0] = 0x01;
WORD	0x0102	Can[0] = 0x02; Can[1] = 0x01;
DWORD	0x0102,0x0304	Can[0] = 0x02; Can[1] = 0x01; Can[2] = 0x04; Can[3] =0x03;
QWORD	0x0102,0x0304 0x0506,0x0708	Can[0] = 0x02; Can[1] = 0x01; Can[2] =0x04; Can[3] = 0x03; Can[4] =0x06; Can[5] = 0x05; Can[6] =0x08; Can[7] = 0x07;

Note: PXB-6021DM converts the values in Modbus registers to the small end byte order of CANopen network. When the data source is a fixed value, the same applies.

7. TPDO parameters

TPDO parameters are used to configure the TPDO parameters of the slave station and the mapping rules between Modbus data and CANopen network. The interface is shown in Figure 3.23:

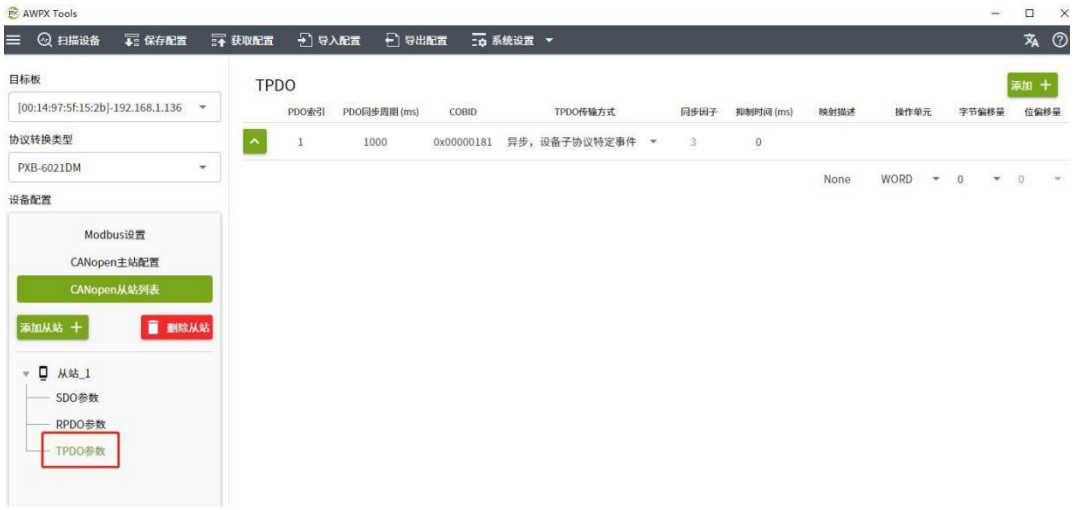

Figure 3.23 TPDO parameter interface

The definition of its setting parameters is as follows:

Table 3.18 TPDO Parameter Description

parameter	Parameter Description
PDO index	The TPDO communication object of the substation dictionary is offset based on the 0X1800 primary index. The interface configuration value will be reduced by 1 during actual transmission. When the PDO index is configured as 1, the actual modified parameters are: 0X1800 + pdo_index - 1 = 0X1800
PDO synchronization period	CANopen to Modbus network data mapping cycle
COBID	COBIND from the slave station (hexadecimal needs to be prefixed with 0x, otherwise it is decimal)
TPDO transmission method	Default is asynchronous transmission
Synchronization factor	Only effective when the transmission method is configured as synchronous cycle (1-240)
Inhibition time	Minimum interval between sending two TPDO messages (0~65535) ms
Mapping description	Used to describe the mapping function, user-defined
Operation unit	Describe the size of the data range mapped from a CAN message to Modbus. There are five types of enumeration: BIT, BYTE, WORD, WORD, and QWORD. When the mapping data size exceeds the size of a Modbus register data, that is, when the current configuration is in DWORD or QWORD, the device will automatically expand the register range to adapt to this mapping. For example, the current configuration is set to WORD, and the Modbus register address is 20000,
Byte offset	The starting offset of the Modbus register mapping to the CAN frame (8 bytes) this time
Positional offset	BIT offset within one byte, only effective when the operating unit is BIT
data source	Fixed value or Modbus network
Modbus register address	Modbus register address (hexadecimal needs to be prefixed with 0x, otherwise it is decimal)
Modbus Register Types	Modbus Register Types
Fixed value	Specific fixed values (hexadecimal needs to be prefixed with 0x, otherwise it is decimal), transmit data in Modbus network according to the configured byte order

Table 3.19 shows the actual mapping of CANopen network to Modbus: The byte offset is 0 and the bit offset is 0;

Table 3.19 TPDO Mapping Table

Operation unit	CAN message data	Modbus register values
BIT	Can[0] = 0x01;	one
BYTE	Can[0] = 0x01;	0x0001
WORD	Can[0] = 0x01; Can[1] = 0x02;	0x0201
DWORD	Can[0] = 0x01; Can[1] = 0x02; Can[2] = 0x03; Can[3] =0x04;	0x0201,0x0403

QWORD	Can[0] = 0x01; Can[1] = 0x02; Can[2] =0x03; Can[3] = 0x04; Can[4] =0x05; Can[5] = 0x06; Can[6] =0x07; Can[7] = 0x08;	0x0201,0x0304 0x0605,0x0807

Note: PXB-6021DM converts the small end byte order of CANopen network to the value of Modbus register. The specific byte order transmitted on the Modbus side depends on the byte order configured on the Modbus parameter configuration interface.

3.2.4 PXB-6022D Parameter Configuration

1. Modbus parameter configuration

Click on "Modbus Settings" in the "Device Configuration" column of AWPX software to configure Modbus parameters. The configuration interface is shown in Figure 3.24.

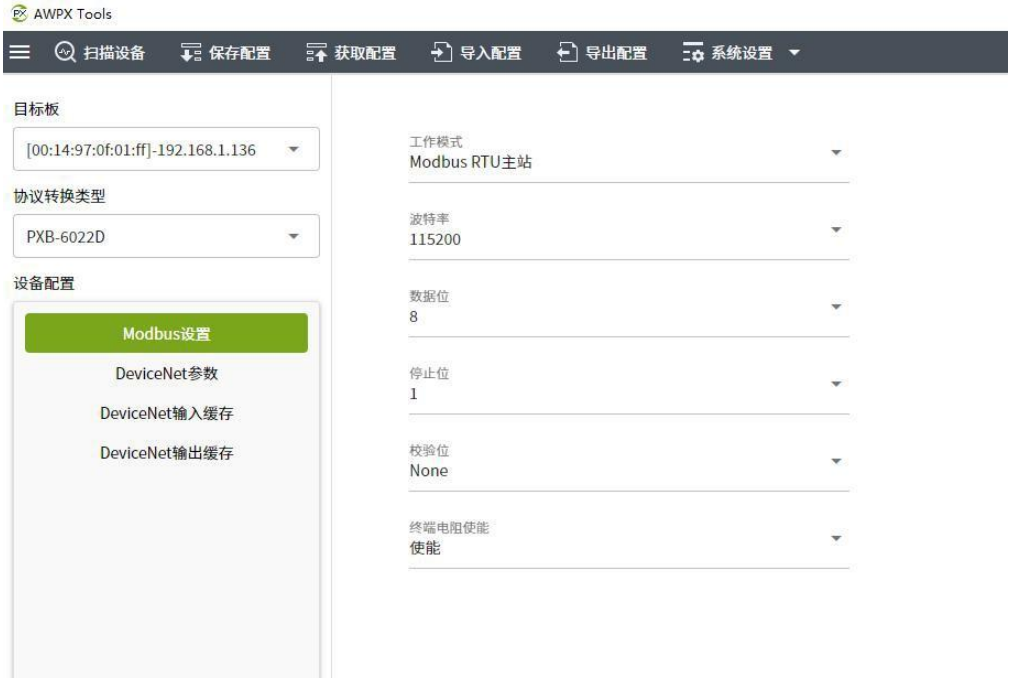

Figure 3.24 Modbus parameter settings

PXB-6022D supports two working modes, which can be selected through the "Working Mode" drop-down list box. Each working mode has corresponding Modbus parameters, and the functional descriptions and corresponding parameter descriptions of these two working modes are shown in Table 3.20.

Table 3.20 Modbus Parameter Description

Working mode	Function Description	Configuration items	Configuration item description
Modbus RTU master station	PXB-6022D works as a Modbus RTU master and supports up to 255 Modbus RTU slaves	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Terminal resistance enable	RS485 terminal resistor enable
Modbus TCP Master	PXB-6022D works as a Modbus TCP master and serves as a TCP client	Slave IP address	The IP address of the only TCP slave station
		Slave port number	The unique TCP slave port number

2. DeviceNet parameter configuration

Click on 'DeviceNet Parameters' in the' Device Configuration 'column of AWPX software to configure the DeviceNet parameters. The configuration interface is shown in Figure 3.25.

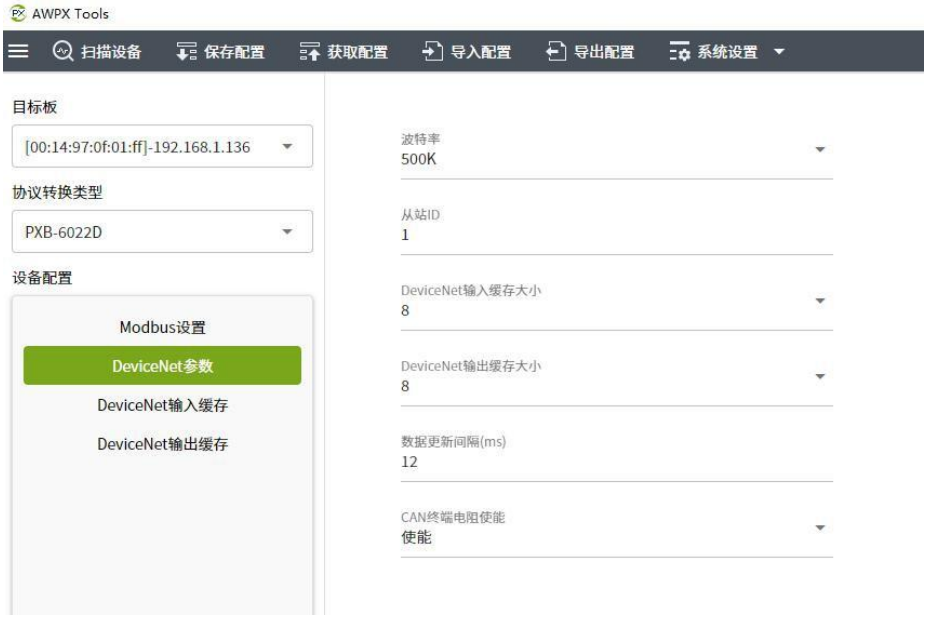

Figure 3.25 DeviceNet parameter interface

The parameter description of DeviceNet is shown in Table 3.21.

Table 3.21 Description of DeviceNet Parameters

parameter	Parameter Description
Baud rate	Set the CAN baud rate of PXB-6022D
Slave ID	Set PXB-6022D as the ID of DeviceNet slave station
DeviceNet input cache size	Set the maximum data size that PXB-6022D can receive at a time, in bytes
DeviceNet output cache size	Set the maximum data size that PXB-6022D can send at a time, in bytes
Data update interval	Synchronization interval between Modbus data and DeviceNet data. The smaller the value, the better the real-time performance. The smaller the value, the higher the performance requirements for Modbus bus, which can be filled in according to real-time needs
CAN terminal resistor enable	Enable or disable the terminal resistance of CAN interface

3. DeviceNet input cache configuration

The DeviceNet input cache stores the data transmitted from the DeviceNet master station to PXB-6022D, which then processes it.DeviceNet input cache is written to Modbus slave registers. The DeviceNet input cache configuration interface is shown in Figure 3.26.

Figure 3.26 DeviceNet Input Cache Configuration Interface

Add a mapping entry by clicking the 'Add+' button in the upper right corner of the interface, and then edit the mapping parameters. On the far right side of the entry, click the [Delete] button to delete the mapping entry. A maximum of 128 mapping entries can be added. DeviceNet
The parameter description for input cache configuration is shown in Table 3.22.

Table 3.22 Explanation of DeviceNet Input Cache Configuration Parameters

parameter	Parameter Description
Operation unit	The size of the mapped data. BYTE: 1 byte, WORD: 2 bytes, DWORD: 4 bytes, QWORD: 8 bytes
Byte offset	Set which byte of DeviceNet input cache to start mapping data from
Positional offset	The bit offset after byte offset, set which bit in the DeviceNet input cache to start mapping data from
Modbus byte order	Set the Modbus big and small end mode, and this field is valid when the operating unit is BYTE, WORD, or WORD. For example: When the operating unit is BYTE, the input data of DeviceNet is 0x10: Big end mode: Modbus register data mapped to 0x1000 Small end mode: Modbus register data mapped to 0x0010
Modbus Slave ID	Set Modbus Slave ID
Modbus type	Set Modbus register type. When the operating unit is BIT, the Modbus type can select coil state. When the operating unit is other, the Modbus type can select hold register
address	Set Modbus register address, which can be decimal or hexadecimal (starting with 0x)

4. DeviceNet output cache configuration

PXB-6022D maps the read Modbus slave register data to the DeviceNet output cache, and then sends the DeviceNet output cache to the DeviceNet master station. The DeviceNet output cache configuration interface is shown in Figure 3.27.

Figure 3.27 DeviceNet Output Cache Configuration Interface

Add a mapping entry by clicking the 'Add+' button in the upper right corner of the interface, and then edit the mapping parameters. On the far right side of the entry, click the [Delete] button to delete the mapping entry. A maximum of 128 mapping entries can be added. DeviceNet
The parameter description of the output cache configuration is shown in Table 3.23.

Table 3.23 Explanation of DeviceNet Output Cache Configuration Parameters

parameter	Parameter Description
Operation unit	The size of the mapped data. BYTE: 1 byte, WORD: 2 bytes, DWORD: 4 bytes, QWORD: 8 bytes
Byte offset	Set which byte of DeviceNet output cache to start mapping data from
Positional offset	Bit offset after byte offset, set which bit of DeviceNet output cache to start mapping data from
Modbus byte order	Set the Modbus big and small end mode, and this field is valid when the operating unit is BYTE, WORD, or WORD. For example: When the operating unit is BYTE, the Modbus register data is 0x10: Big end mode: DeviceNet's output data is mapped to 0x00 Small end mode: DeviceNet's output data is mapped to 0x10
Modbus Slave ID	Set Modbus Slave ID
Modbus type	Modbus register type. When the operating unit is BIT, the Modbus type can be selected as either coil state or input state. When used for other operating units, the Modbus type can choose to hold registers or input registers
address	Set Modbus register address, which can be decimal or hexadecimal (starting with 0x)

3.2.5 PXB-6022DM Parameter Configuration

1. Modbus parameter configuration

Click on "Modbus Settings" in the "Device Configuration" column of AWPX software to configure Modbus parameters. The configuration interface is shown in Figure 3.28.

Figure 3.28 Modbus parameter settings

PXB-6022DM supports four working modes, which can be selected through the "Working Mode" drop-down list box. The functional descriptions and corresponding parameter descriptions of these four working modes are shown in Table 3.24.

Table 3.24 Modbus Parameter Description

Working mode	Function Description	Configuration items	Configuration item description
Modbus RTU master station	PXB-6022DM operates as a Modbus RTU master station and can be externally connected to a unique Modbus RTU slave station	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Slave ID	The ID of the external unique slave station
		Byte order	Modbus data storage method
		Terminal resistance enable	RS485 terminal resistor enable
Modbus RTU slave station	PXB-6022DM works as a Modbus RTU slave, with 2400 built-in coils, input status, input registers, and hold registers, all with addresses ranging from 0 to 2399	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Local slave ID	PXB-6022DM as the ID of the slave station
		Byte order	Modbus data storage method
		Terminal resistance enable	RS485 terminal resistor enable
Modbus TCP Master	PXB-6022DM operates as a Modbus TCP master and serves as a TCP client	Slave IP address	The IP address of the only TCP slave station
		Slave port number	The unique TCP slave port number
		Peer slave number	Unique TCP slave ID
		Byte order	Modbus data storage method
Modbus TCP Slave	PXB-6022DM operates as a Modbus TCP slave and serves as a TCP server. 2400 built-in coils, input status, input registers, and hold registers, with addresses ranging from 0 to 2399	Local port number	PXB-6022DM serves as the port number for the slave station
		Local slave number	PXB-6022DM as the ID of the slave station
		Byte order	Modbus data storage method

2. DeviceNet master station parameter configuration

Click on 'DeviceNet Master Station Parameters' in the' Device Configuration 'column of the AWPX software to configure the relevant parameters of the DeviceNet master station. The configuration interface is shown in Figure 3.29.

Figure 3.29 DeviceNet Master Station Parameter Interface

The parameter description of DeviceNet master station is shown in Table 3.25.

Table 3.25 Description of DeviceNet Master Station Parameters

parameter	Parameter Description
Baud rate	CAN baud rate of PXB-6022DM
Main station MAC ID	PXB-6022DM serves as the device address for the DeviceNet master station
Internal scanning delay	The internal scanning delay is the minimum time allowed for external devices to access the network after continuous I/O scanning by the scanner. If the value is too high, it will cause a longer network scan, which will affect the execution of input and output. If the value is too small, it will slow down the scanner module's response to external devices
Expected message time	Expecting message rate, determining the timeout time for bit gating and polling messages
CAN terminal resistor enable	Enable or disable the terminal resistance of CAN interface

3. DeviceNet Slave List Configuration

Click on 'DeviceNet Slave List' in the 'Device Configuration' column of AWPX software to manage DeviceNet slaves and configure their related parameters. The configuration interface is shown in Figure 3.30.

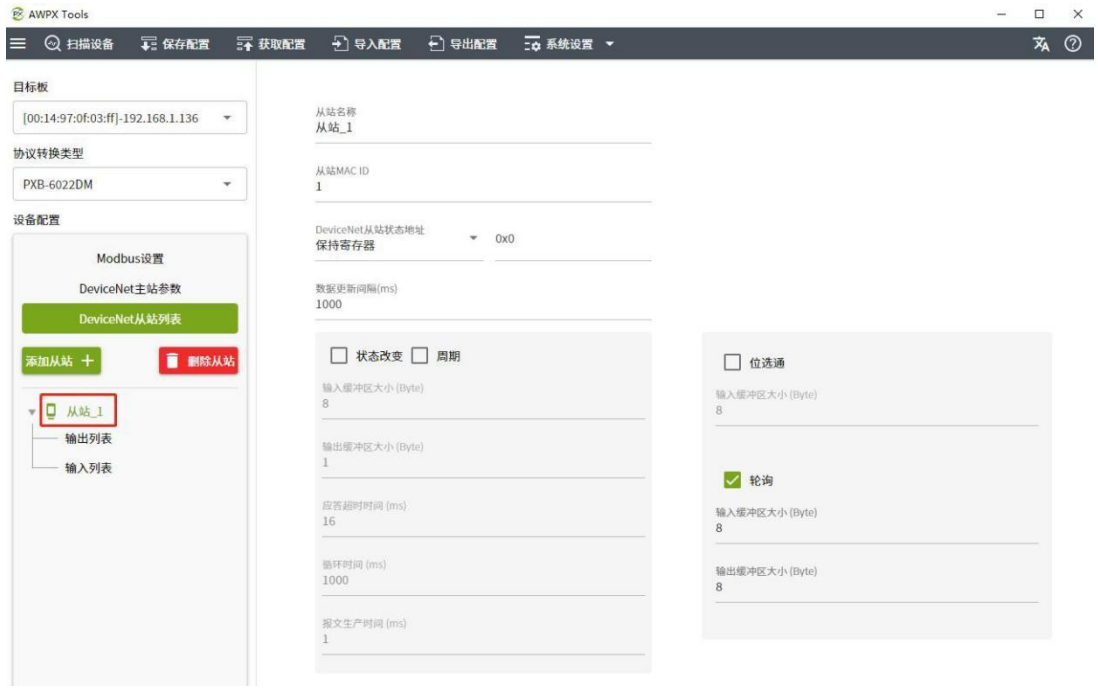

Figure 3.30 Slave Station List Interface

On the left side of this interface, DeviceNet slaves can be managed. Click the 'Add Slave+' button to create a new DeviceNet slave that needs to be connected. Click the 'Delete Slave' button to delete the corresponding DeviceNet slave.

Below the 'Add Slave' button is a list of connected DeviceNet slaves, each with an editable name. Each slave station contains an output list and an input list. Click the 【 ▼ 】 button to the left of the slave name to hide the corresponding output and input lists for that slave.

Click on the name of the corresponding DeviceNet slave to edit parameters such as the name

and MAC ID of the corresponding DeviceNet slave on the right side of the interface. The parameter description of DeviceNet slave list is shown in Table 3.26.

Table 3.26 Parameter Description of DeviceNet Slave List

parameter	Parameter Description
Station name	The name of this DeviceNet slave can be used for mnemonic purposes
Slave MAC ID	The device address of this DeviceNet slave station
DeviceNet slave status address	Note that the DeviceNet slave status value should not overlap with the register address of the mapping data when holding the register or input register address
Data update interval	Modbus data and DeviceNet synchronization interval. The smaller the value, the better the real-time performance. The smaller the value, the higher the performance requirements for Modbus bus, which can be filled in according to real-time needs

The state of DeviceNet slave during runtime is reflected by the state value of the DeviceNet slave, which can be selected to be reflected at the address corresponding to the hold register or input register.

If PXB-6022DM is running in Modbus master mode, the hold register can be selected to write the status value of DeviceNet slave to the hold register corresponding to the address of the connected Modbus slave.

If PXB-6022DM is running in Modbus slave mode, it can choose to hold registers or input registers, which will Write the status value of DeviceNet slave to the hold register or input register of the corresponding address of PXB-6022DM itself.

The status values of DeviceNet slave stations are shown in Table 3.27

Table 3.27 Description of DeviceNet Slave Status Values

Status value (decimal)	Description of Status Values
0	No errors
fifty-seven	Other undefined internal errors
sixty-one	The sub station does not exist
sixty-two	Sending data failed
sixty-three	no data
sixty-six	The main station is not online
sixty-seven	Connection does not exist
seventy-two	The device stops communicating
seventy-seven	Data length mismatch
seventy-eight	The device is in the scanning list but has not responded
eighty-four	The device has not been initialized (main station status)
eighty-six	The device enters an idle state on its own

1. I/O connection type

Below the 'Data Update Interval' parameter, you can select the I/O connection type, which includes four types: bit select, polling, state change, and cycle. Each DeviceNet slave station must select at least one I/O connection type and a maximum of three I/O connection types. State change and cycle cannot be selected simultaneously.

Bit gating: The bit gating method requires configuring the input buffer size of the DeviceNet slave station, which can be configured to be 1-64Byte; Polling: The polling method requires configuring the input and output buffer sizes of DeviceNet slave stations. The configurable sizes

are:1~64Byte；

When selecting the status change type, the interface is shown in Figure 3.31.

Figure 3.31 Mode of State Change

The parameter description of this interface is shown in Table 3.28.

Table 3.28 Description of State Change Mode Parameters

parameter	Parameter Description
Input buffer size	The maximum size of I/O data that can be input to this DeviceNet slave station at a time, ranging from 1 to 64 bytes
Output buffer size	The maximum I/O data size that this DeviceNet slave can output at a time, ranging from 1 to 64 bytes
Response timeout	The time from sending a message to receiving a response from the DeviceNet master station when the DeviceNet slave station status changes
Heartbeat message time	The interval between when DeviceNet sends heartbeat messages from the slave station. When using the state change mode, the heartbeat message can query the device status at regular intervals to prevent DeviceNet from disconnecting from the slave station
Message production time	The time for generating DeviceNet slave messages should be less than the time for heartbeat messages

When selecting the cycle type, the interface is shown in Figure 3.32.

Figure 3.32 Periodic Mode

The parameter description of this interface is shown in Table 3.29.

Table 3.29 Explanation of Periodic Mode Parameters

parameter	Parameter Description
Input buffer size	The maximum size of I/O data that can be input to this DeviceNet slave station at a time, ranging from 1 to 64 bytes
Output buffer size	The maximum I/O data size that this DeviceNet slave can output at a time, ranging from 1 to 64 bytes
Response timeout	The time from sending a message from the DeviceNet slave station to receiving a response from the DeviceNet master station
Cycle time	The time interval of DeviceNet's slave station cyclic communication can reduce unnecessary network traffic

2. Output List

On the DeviceNet Slave List interface, click on the Output List below the DeviceNet Slave name, as shown in Figure 3.33. The output list is a mapping entry list for the output data of the DeviceNet master station relative to the DeviceNet master station.

Figure 3.33 Output List

Add a mapping entry by clicking the 'Add+' button in the upper right corner of the interface, and then edit the mapping parameters. On the far right side of the entry, click the [Delete] button to delete the mapping entry. The parameter description of the DeviceNet master station output list is shown in Table 3.30.

Table 3.30 Explanation of Output List Parameters

parameter	Parameter Description
Variable Name	The name of this mapping entry can be used for mnemonic purposes
Operation unit	The size of the mapped data. BYTE: 1 byte, WORD: 2 bytes, DWORD: 4 bytes, QWORD: 8 bytes
IO type	Select the selected I/O connection type
DeviceNet byte offset	Starting from the byte of the I/O output data, map Modbus register data to the I/O output data
DeviceNet byte offset	The bit offset after byte offset is valid when the operation unit is BIT. Namely, starting from the byte and bit of the I/O output data, map Modbus register data to the I/O output data

Register type	Supports coils, input status, input registers, hold registers, and options related to the [operating unit]
Register address	Read the starting address of the Modbus slave register, which can be decimal or hexadecimal (starting with 0x)

Output list mapping instructions:

Configure an output list mapping entry as shown in Figure 3.33: Configure the operation unit as a WORD, IO type as polling, DeviceNet byte offset as 2, register type as hold register, and register address as 3.

Then PXB-6022DM will first start with address 3 and sequentially read 4 bytes of data from the Modbus slave's hold register (i.e. 2 registers). If PXB-6022DM is running in Modbus master mode, it reads the hold register corresponding to the address of the connected Modbus slave. If PXB-6022DM is running in Modbus slave mode, read the hold register corresponding to the address of PXB-6022DM itself.

Then, according to the byte offset of 2, the read hold register data is sequentially written to the positions of the 2nd, 3rd, 4th, and 5th bytes of the I/O output data. Finally, output the I/O output data to the DeviceNet slave station.

3. Input List

In the 'DeviceNet Slave List' interface, click on the 'Input List' below the DeviceNet Slave name, as shown in Figure 3.34. The input list is a mapping entry list for the input data of the DeviceNet master station relative to the DeviceNet master station.

Figure 3.34 Input List

Add a mapping entry by clicking the 'Add+' button in the upper right corner of the interface, and then edit the mapping parameters. On the far right side of the entry, click the [Delete] button to delete the mapping entry.

The parameter description of the DeviceNet master station input list is shown in Table 3.31.

Table 3.31 Input List Parameter Description

parameter	Parameter Description
Variable Name	The name of this mapping entry can be used for mnemonic purposes
Operation unit	The size of the mapped data. BYTE: 1 byte, WORD: 2 bytes, DWORD: 4 bytes, QWORD:8 bytes
IO type	Select the selected I/O connection type
DeviceNet byte offset	Starting from the byte of the I/O input data, map the I/O input data to Modbus registers
DeviceNet byte offset	The bit offset after byte offset is valid when the operation unit is BIT. Namely, starting from the byte and bit of the I/O input data, map the input I/O data to Modbus registers

Register type	Supports coils, input status, input registers, hold registers, and options related to the [operating unit]
Register address	Write the I/O data input from DeviceNet to the starting address of the Modbus slave register in sequence

Input list mapping instructions:

Configure an input list mapping entry as shown in Figure 3.34: Configure the operation unit as a WORD, IO type as polling, DeviceNet byte offset as 3, register type as hold register, and register address as 7.

Firstly, after receiving the I/O data input from the DeviceNet slave station, PXB-6022DM writes the 3rd, 4th, 5th, and 6th bytes of the I/O input data into the Modbus slave station's hold register (i.e., the hold registers for addresses 7 and 8) starting from address 7, based on byte offset 3. If PXB-6022DM is running in Modbus master mode, write to the hold register corresponding to the address of the connected Modbus slave. If PXB-6022DM is running in Modbus slave mode, write it to the hold register corresponding to the address of PXB-6022DM itself.

3.2.6 PXB-6030D Parameter Settings

1. Basic settings

Click on 'Basic Settings' in the' Device Configuration 'column of the AWPX software to select the working mode and configure the transition timeout. The configuration interface has a connection diagram corresponding to the working mode, as shown in Figure 3.35.

Figure 3.35 Basic Settings

PXB-6030D supports four working modes, which can be selected through the "Working Mode" drop-down list box. Next, we will introduce these four working modes.

1. Modbus RTU master to TCP slave

This mode PXB-6030D enables bidirectional communication between Modbus RTU master devices and Modbus TCP slave devices. Firstly, the Modbus RTU master device sends a request frame, which is received by PXB-6030D. The request frame is then converted and sent to the Modbus TCP slave device. Afterwards, the Modbus TCP slave device responds by replying with a response frame. The PXB-6030D receives the response frame, converts it, and sends it to the Modbus RTU master device. The data flow direction is: ①→②→③→④.

The wiring method is: Connect the RS485 interface of PXB-6030D to the Modbus RTU

master device, and connect the Ethernet port to Modbus TCP slave device connection, supporting up to 8 Modbus TCP slave devices. The connection diagram is shown in Figure 3.36.

Figure 3.36 Schematic diagram of Modbus RTU master to TCP slave mode connection

2. TCP master to Modbus RTU slave

This mode PXB-6030D enables bidirectional communication between Modbus TCP master devices and Modbus RTU slave devices. Firstly, the Modbus TCP master sends request frames, and the PXB-6030D receives the request frames, which are then converted and sent to the Modbus RTU slave devices. Afterwards, the Modbus RTU slave device responds with a reply frame, and the PXB-6030D receives the response frame, converts it, and sends it to the Modbus TCP master device. The data flow direction is: ①→②→③→④.

The wiring method is as follows: the Ethernet port of PXB-6030D is connected to the Modbus TCP master device, and the RS485 interface is connected to the Modbus RTU slave device, supporting up to 255 Modbus RTU slave devices. The connection diagram is shown in Figure 3.37.

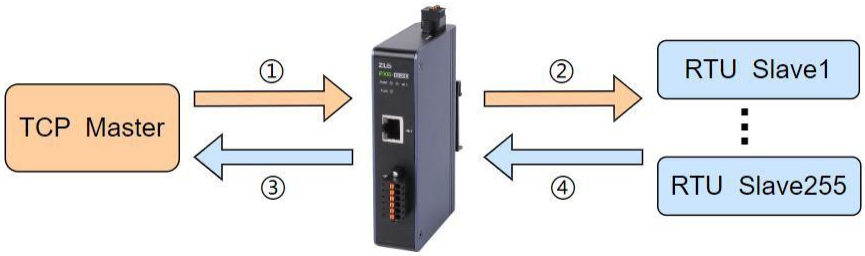

Figure 3.37 Schematic diagram of TCP master to Modbus RTU slave mode connection

3. ASCII master to TCP slave

This mode PXB-6030D enables bidirectional communication between Modbus ASCII master devices and Modbus TCP slave devices. Firstly, the Modbus ASCII master device sends a request frame, which is received by PXB-6030D. The request frame is then converted and sent to the Modbus TCP slave device. The Modbus TCP slave device responds and replies with a response frame. The PXB-6030D receives the response frame, converts it, and sends it to the Modbus ASCII master device. The data flow direction is: ①→②→③→④.

The wiring method is: Connect the RS485 interface of PXB-6030D to the Modbus ASCII master device, and connect the Ethernet port to Modbus TCP slave device connection, supporting up to 8 Modbus TCP slave devices. The connection diagram is shown in Figure 3.38.

Figure 3.38 Schematic diagram of ASCII master to TCP slave mode connection

4. TCP master to ASCII slave

This mode PXB-6030D enables bidirectional communication between Modbus TCP master devices and Modbus ASCII slave devices. Firstly, the Modbus TCP master sends request frames, and the PXB-6030D receives the request frames, which are then converted and sent to the Modbus ASCII slave devices. Afterwards, the Modbus ASCII slave device responds with a reply frame, and the PXB-6030D receives the response frame, converts it, and sends it to the Modbus TCP master device. The data flow direction is: ①→②→③→④.

The wiring method is as follows: the Ethernet port of PXB-6030D is connected to the Modbus TCP master device, and the RS485 interface is connected to the Modbus ASCII slave device, supporting up to 255 Modbus ASCII slave devices. The connection diagram is shown in Figure 3.39.

Figure 3.39 Schematic diagram of TCP master to ASCII slave mode connection

Conversion timeout: Taking PXB-6030D running Modbus RTU master to TCP slave mode as an example, when PXB-6030D receives a command from an external Modbus RTU master device, it starts timing. Within the conversion timeout, if PXB-6030D receives a correct response from the external TCP slave device, it will complete the protocol conversion. Otherwise, PXB-6030D will not respond to this Modbus RTU master command. The same applies to other modes.

PXB-6030D supports Modbus function codes as shown in Table 3.32 in any working mode.

Table 3.32 Function Codes Supported by PXB-6030D

Function code	function
01H	Read coil status
02H	Read input status
03H	Read and hold register
04H	Read input register
05H	Write a single coil
06H	Write a single hold register
0FH	Write multiple coils
10H	Write multiple hold registers

2. RS485 parameters

Click on "RS485 Parameters" in the "Device Configuration" column of AWPX software to set the PXB-6030D device RTU/ASCII communication parameters. The interface is shown in Figure 3.40.

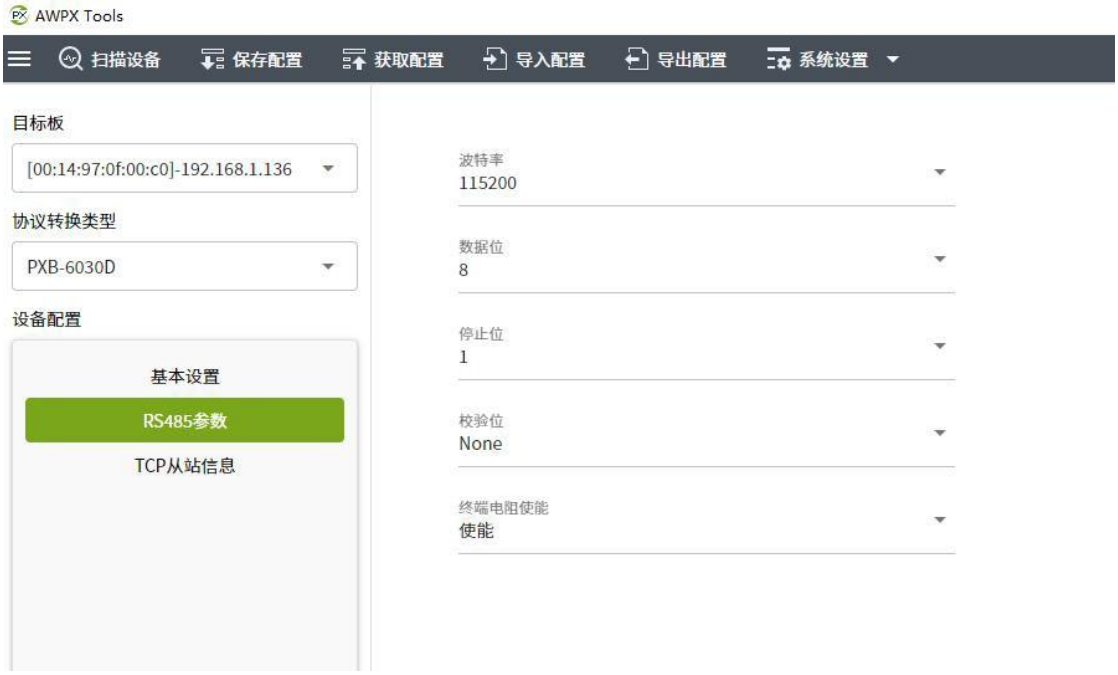

Figure 3.40 RS485 Parameter Interface

The RS485 parameter is the RTU/ASCII communication parameter of the PXB-6030D device, which needs to be consistent with the communication parameter settings of the external Modbus RTU/ASCII device. The maximum baud rate can be set to 2M. Enable RS485 terminal resistors through the 'Terminal Resistance Enable' option.

3. TCP Slave Information

Click on 'TCP Slave Information' in the 'Device Configuration' column of AWPX software to configure TCP slave information. When selecting different working modes, the interface for "TCP Slave Information" is different. When selecting the working mode of Modbus RTU master to TCP slave or ASCII master to TCP slave, the "TCP Slave Information" interface is shown in Figure 3.41.

Figure 3.41 TCP Slave Information Interface 1

When selecting the working mode of Modbus RTU master to TCP slave or ASCII master to TCP slave, this interface can set the information of the TCP slave that 8 PXB-6030D devices need to connect to. The parameter description of this interface is shown in Table 3.33.

Table 3.33 TCP Slave Information Parameter Explanation 1

parameter	Parameter Description
Enable/Disable	Enable or disable the setting for this TCP slave information
IP address	The IP address of the TCP slave station that the PXB-6030D device needs to connect to
Port number	The port number of the TCP slave station that the PXB-6030D device needs to connect to
Slave ID	When the Modbus RTU/ASCII master sends a request frame for this slave ID, it will be sent to this TCP slave
UID	When UID is set to 0, the slave ID will be set to UID and sent to the current slave device; When UID is set to>0, set this value as the frame UID and send it to the current slave device
Reconnect time	When PXB-6030D fails to establish a TCP connection or disconnects from a TCP slave, PXB-6030D reconnects with TCP The time interval for establishing a TCP connection from a slave station until PXB-6030D successfully connects to the TCP slave station
Survival time	TCP connection heartbeat detection time, used for disconnection detection. If the heartbeat response exceeds the keep alive time, PXB-6030D Disconnect the current TCP connection and reconnect based on the reconnection time

Note: UID is the unit identifier in Modbus TCP packets, which serves as the identification code for Modbus TCP slave devices and is consistent with the Modbus slave ID commonly used on Modbus serial links.

Example of using slave ID and UID: Taking PXB-6030D running Modbus RTU master to TCP slave as an example, connect two external TCP slaves. The IP address of TCP slave 1 is 192.168.1.10, and the IP address of TCP slave 2 is: 192.168.1.118. Configure TCP slave information as shown in Figure 3.42.

Figure 3.42 Example configuration of using slave ID and UID

The slave ID and UID settings for two TCP slave stations are shown in the above figure.

When the Modbus RTU master sends request frame 1:01 03 00 00 02 C4 0B and request frame 2:02 03 00 00 02 C4 38. Then PXB-6030D will change the slave ID in request frame 1 from 01 to 07, and after conversion, send it to TCP slave 1. Finally, replace the ID in the response frame from TCP slave 1 with 01 and send it to the Modbus RTU master. And without making any changes to request frame 2, after conversion, it is sent to TCP slave 2.

When selecting the working mode of TCP master to Modbus RTU slave or TCP master to ASCII slave, the interface of "TCP slave information" is shown in Figure 3.43.

Figure 3.43 TCP Slave Information Interface 2

When the PXB-6030D device operates in TCP master to Modbus RTU slave or TCP master to ASCII slave mode, the parameter specifications on this interface are shown in Table 3.34.

Table 3.34 TCP Slave Information Parameter Description 2

parameter	Parameter Description
Port number	Port number of PXB-6030D device
TCP connection keep alive time	TCP connection heartbeat detection time, used for disconnection detection. If the heartbeat response exceeds the keep alive time, PXB-6030D Disconnect the current TCP connection and reconnect based on the reconnection time
Fixed slave ID	When set to 0, the UID of the request frame sent by the TCP master station is the ID of the slave station to be accessed; When set greater than 0, change the UID of the request frame to the fixed slave ID set here, and then send it to the serial port slave device

3.2.7 PXB-6031D Parameter Configuration

1. Modbus parameter configuration

Click on "Modbus Settings" in the "Device Configuration" column of AWPX software to configure Modbus parameters. The configuration interface is shown in Figure 3.44.

Figure 3.44 Modbus parameter settings

PXB-6031D supports two working modes, which can be selected through the "Working Mode" drop-down list box. Each working mode has corresponding Modbus parameters, and the functional descriptions and corresponding parameter descriptions of these two working modes are shown in Table 3.35.

Table 3.35 Modbus Parameter Description

Working mode	Function Description	Configuration items	Configuration item description
Modbus RTU master station	PXB-6031D works as a Modbus RTU master and supports up to 255 Modbus RTU slaves	Baud rate	RTU communication parameters
		Data bits	
		Stop position	
		Checksum	
		Terminal resistance enable	RS485 terminal resistor enable
Modbus TCP Master	PXB-6031D works as a Modbus TCP master and serves as a TCP client	Slave IP address	The IP address of the only TCP slave station
		Slave port number	The unique TCP slave port number

2. OPC UA parameter configuration

Click on 'OPC UA Parameters' in the' Device Configuration 'column of the AWPX software to configure the parameters of the OPC UA server, as shown in Figure 3.45.

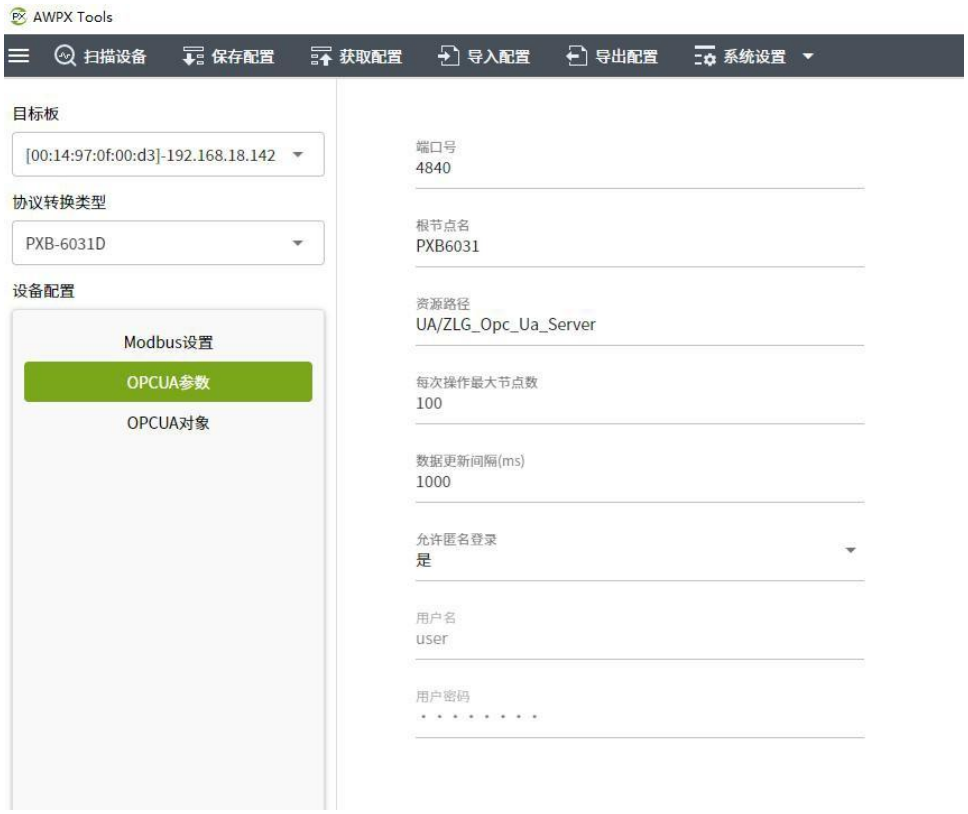

Figure 3.45 OPC UA Parameter Interface

The OPC UA parameter specifications are shown in Table 3.36.

Table 3.36 OPC UA Parameter Description

parameter	Parameter Description
Port number	Set the port number for the OPC UA server
Root node roll call	Set the root node directory name where the OPC UA object is located
Resource Path	Set the resource path for OPC UA server
Maximum number of nodes per operation	Set the maximum number of nodes that the OPC UA server can operate at a time
Data update interval	Data update interval between OPC UA and Modbus. The smaller the value, the better the real-time performance. The smaller the value, the higher the performance requirements for Modbus bus, which can be filled in according to real-time needs
Allow Anonymous Access	Used for access control, optional yes or no. When selecting 'No', the OPC UA client needs to enter the correct username and password to establish a proper connection with PXB-6031D. Otherwise, there is no need to enter a username and password
user name	Allow anonymous login. When selecting 'No', set the username of the OPC UA server here
User password	Allow anonymous login. When selecting 'No', set the user password for the OPC UA server here

PXB-6031D supports signature or encryption and signature, and supports three data encryption methods: Basic256, Basic256Sha256, and Aes128Shah256RsaOaep. OPC UA clients can choose not to encrypt or select one of these three data encryption methods.

3. OPC UA Object Configuration

Click on "OPC UA Object" in the "Device Configuration" column of AWPX software, and then click the "Add+" button to create a new OPC UA object. Up to 8 objects can be created. Click the "Delete" button on the right to delete the object. click

The name of the object, as shown in Figure 3.46 as "obj_0", can be used to configure the parameters of the OPC UA object.

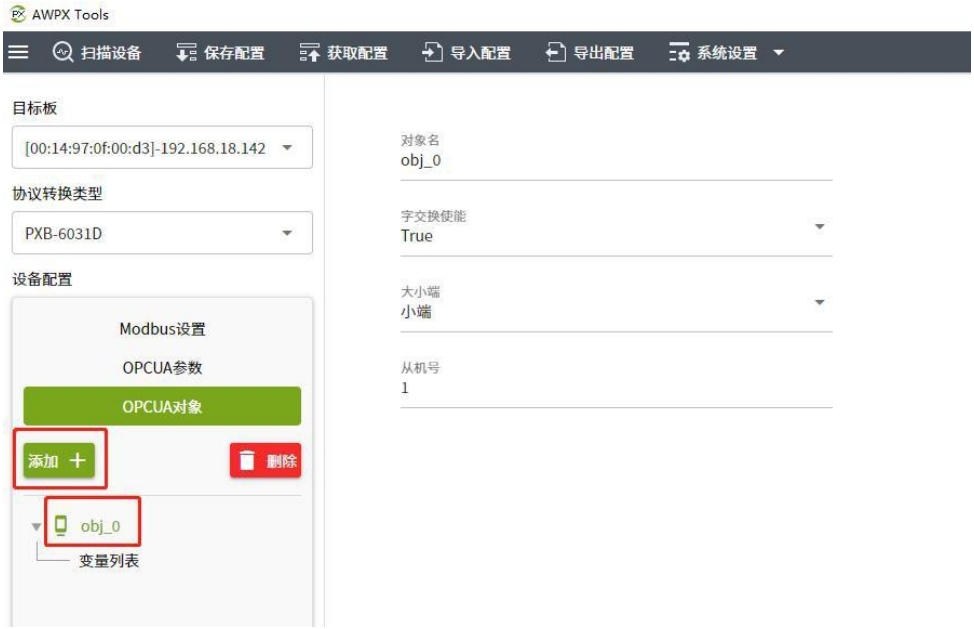

Figure 3.46 OPC UA Object Configuration Interface

The parameter specifications of OPC UA objects are shown in Table 3.37.

Table 3.37 Parameter Description of OPC UA Objects

parameter	Parameter Description
Object Name	The name of the OPC UA object, which can be used for mnemonic purposes
Word exchange enablement	When setting the data type for read and write to 32 bits, should the data in these two registers be swapped
Size end	Set the big and small end formats for reading and writing Modbus data
Slave number	Set Modbus Slave ID

Click the 'Variable List' button below the object name to create or delete variables within the object, as shown in Figure 3.47. Click the 'New Variable+' button in the upper right corner to create a new variable within the object. The total number of newly created variables in all objects can reach up to 2000. Click the 'Delete Variable' button to delete the variable.

Figure 3.47 OPC UA Variable Configuration Interface

Click the 'Export CSV' button to export the configured OPC UA objects and variables Export in table format with CSV suffix. The data format of the table exported from the OPC UA variable configured in Figure 3.47 is shown in Figure 3.48.

The CSV table file can be edited in batches according to the data format shown in Figure 3.48 for OPC UA objects and variables. After editing, click the [Import CSV] button to import into AWPX software.

A	B	C	D	E	F	G	H	I	J	K
obj_0	var_0	int16	1	write	1	3	0	1	little_endia	TRUE
obj_0	var_1	uint16	1	read_write	1	3	3	1	little_endia	TRUE

Figure 3.48 OPC UA Data Format

The parameter descriptions of the OPC UA variables in the variable list are shown in Table 3.38.

Table 3.38 OPC UA Parameter Description

parameter	Parameter Description
Variable Name	The name of OPC UA variable, which can be used for mnemonic purposes
data type	Data types for reading and writing Modbus data
Number of registers	Specify the number of Modbu slave registers for reading and writing, with data type hexstr ascstr、u16_array、i16_array Can be set with boor_array, up to a maximum of 96
Read and write properties	Set read-write Modbus slave, which can be set as read-only, write only, or read-write Modbus
Transformation coefficient	The value of Modbus data multiplied by the variable coefficient is the value of OPC UA variable data, which can be set when the data type is float
Register type	Set the type of Modbus register for reading and writing
Register address	Set the starting address of the Modbus register for reading and writing, which can be entered in decimal or hexadecimal format (starting with 0x)

Addendum:

[Data Type]: hexstr is a string of hex type, supporting characters from '0' to '9', 'a' to 'f', and 'A' to 'F'. Determine the string length by setting the number of registers, where one register corresponds to four characters. Ascstr is an ASCII code type string that supports ASCII encoded characters.

u16_array is a Uint16 array type, and the number of members in the Uint16 array is determined by setting the 'number of registers'. One register corresponds to one array member. I16uarray and boolean array are of type int16 and boolean, respectively, with the same correspondence as above. The corresponding array or string can be operated on the OPC UA client.

Read and write properties: Read and write properties are relative to OPC UA clients. If set to 'Write Modbus', only the value of OPC UA variables can be set in the OPC UA client, and then PXB-6031D writes the value to the corresponding Modbus register at the address.

[Word Swap Enable]: One register in Modbus is 2 bytes, or 16 bits. In AWPX software, there are 32 bit data types including int32, uint32, and float32. Assuming the data type is int32 and registers A and B are adjacent addresses. When the word swap setting is not enabled, the data written to register A is 0x1234, and the data written to register B is 0x5678. Enable word swapping and write the same data. The data written to register A becomes 0x5678, and the data written to register B becomes 0x1234. The same applies when reading registers.

4. Product installation

4.1 Mechanical dimensions

PXB-60xxD series product dimensions: 125mm * 76mm * 28mm.

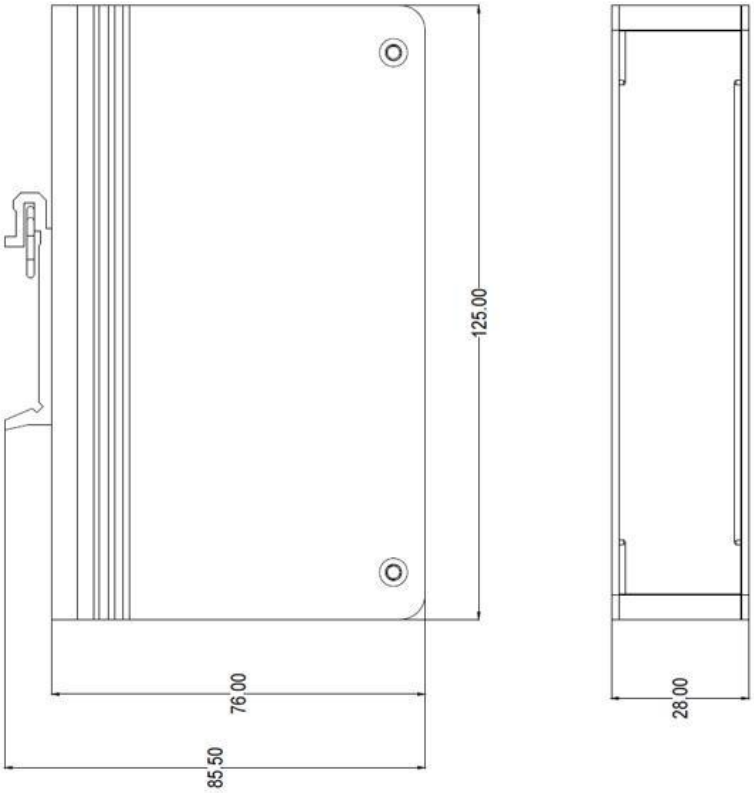

Figure 4.1 Product Dimensional Diagram

5. Product maintenance and precautions

Before powering on the product, please check if the power input voltage is within the required range, if the product wiring is reasonable, and if there are any short circuits or incorrect signal lines;

The product does not have IP protection level requirements and needs to be protected from water ingress, which may affect the normal operation of the product;

6. Appendix

6.1 Product Packing List

Serial number	name	quantity	Company	Physical picture
1	PXB-60xxD protocol converter	one	platform	
2	3P power terminal	one	only	
3	6P terminal	one	only	
4	certificate	one	Zhang	

7. Disclaimer

In accordance with the principle of providing better services to users, Guangzhou Zhiyuan Electronics Co., Ltd. (hereinafter referred to as "Zhiyuan Electronics") will present detailed and accurate product information to users as much as possible in this manual. However, due to the timeliness of the content in this manual, Zhiyuan Electronics cannot fully guarantee the timeliness and applicability of this document at any time. Zhiyuan Electronics reserves the right to update the content of this manual without prior notice. To obtain the latest version of information, please visit the official website of Zhiyuan Electronics regularly or contact Zhiyuan Electronics staff. Thank you for your tolerance and support!

Integrity and win-win, continuous learning, customer first, professional focus, only do first

Guangzhou Zhiyuan Electronics Co., Ltd

For more details, please visit

www.zlg.cn

Welcome to call the national servi

400-888-4005