BL200 Modbus TCP Distributed I/O

BL200

User Manual

Version：V1.2

Date：2023-10-24

Shenzhen Beilai Technology Co.,Ltd

Website：https://www.bliiot.com

Preface

Thanks for choosing BLIIoT Distributed I/O. These operating instructions contain all the information you need for operation of BL200 series products.

Copyright

Disclaimer

This document is designed for assisting user to better understand the device. As the described device is under continuous improvement, this manual may be updated or revised from time to time without prior notice. Please follow the instructions in the manual. Any damages caused by wrong operation will be beyond warranty.

Revision History

Update Date	Version	Description	Owner
2021-10-13	V1.0	First Edition	ZLF
2022-07-01	V1.1	Add Profinet, EtherCAT protocol, add platform, logic control functions	HYQ
2023-07-27	V1.1	Change Model name	HYQ
2023-10-24	V1.2	Add BL203, BL206, BL207 description	HYQ
2023-10-24	V1.2	User manual split by model	HYQ

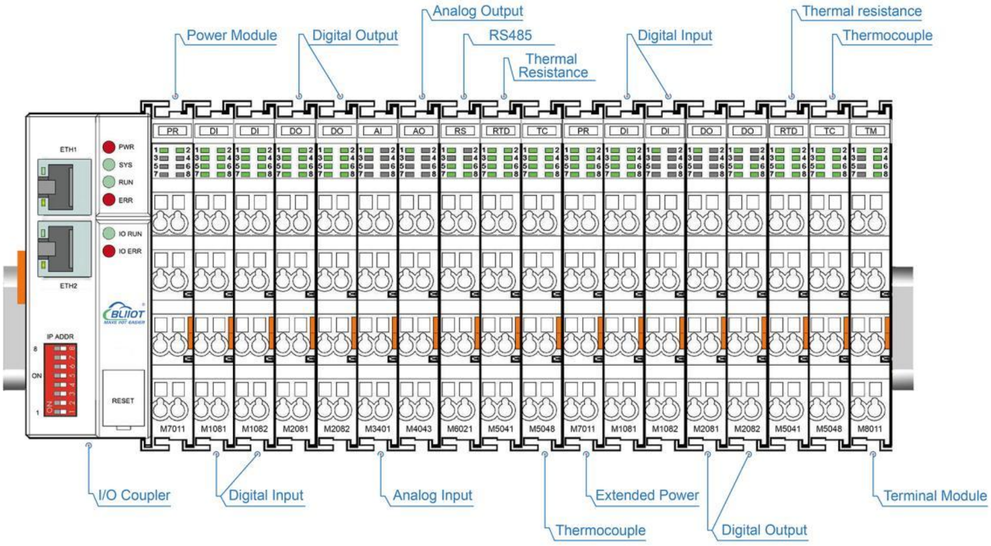
Content

1 Product Introduction

1.1 Overview

BL200 coupler is a data acquisition and control system based on a powerful 32-bit microprocessor design with Linux operating system and Modbus protocol support for quick access to on-site PLCs, SCADA, and ERP systems with built-in logic control, edge computing applications for IIoT and industrial automation applications.

The BL200 distributed I/O system consists of 3 parts: Coupler, I/O modules and terminal modules.



The communication between the node and the field devices (eg PLC) takes place via the Ethernet interface of the fieldbus coupler, and the communication between the fieldbus coupler and the I/O modules takes place via the local bus. The two Ethernet interfaces are internally integrated with a switch function, which can establish a linear topology without the need for additional switches or hubs.

The system needs to use the power module to provide 24VDC system voltage and 24VDC field voltage. Since two independent power supplies are used, the field voltage input interface and system voltage input interface of BL200 couplers are electrically isolated from each other.

When assembling fieldbus node modules, each I/O module can be arranged in any combination, and it is not required to be grouped by module type.

A terminal module must be plugged into the end of a fieldbus node to ensure correct data transmission.

1.2 Typical Application

High reliability, easy expansion, easy setting, and convenient network wiring, these capabilities let users efficiently adapt the BL200 I/O system to a variety of complex industrial solutions.

The I/O system is widely applicable to a variety of industrial solutions, such as Internet of Things, smart factories, smart cities, smart medical care, smart homes, smart transportation, data center environment monitoring, electric power, oil monitoring, automobiles, warehousing and logistics and other industries.

1.3 Features

- Each I/O system can have a maximum of I/O 32 modules.
- Support Modbus TCP protocol.
- Support programmable logic control, edge computing.
- The field side, the system side and the bus side are electrically isolated from each other.
- Support 2 X RJ45 interface, integrated switch function, can establish line topology, without the need for additional switches or hubs.
- Convenient wiring connection technology, screw-free installation.

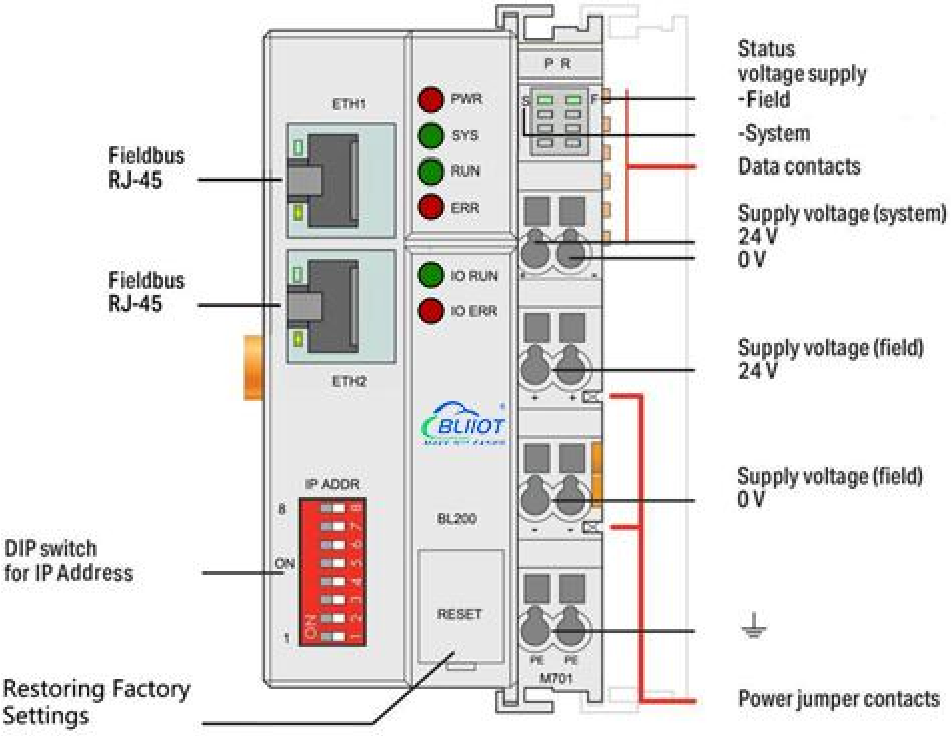
1.4 Model List

Description	Model	Channel	Type
Modbus-TCP I/O Coupler	BL200	/	Modbus TCP
Profinet I/O Coupler	BL201	/	/
EtherCAT I/O Coupler	BL202	/	/
Ethernet/IP I/O Coupler	BL203	/	/
OPC UA EdgeIO Controller	BL205	/	/
MQTT EdgeIO Controller	BL206	/	/
MQTT+OPC UA+Modbus TCP	BL206Pro	/	/
BACnet/IP I/O Coupler	BL207	/	/
BACnet/IP+MQTT+OPC UA	BL207Pro	/	/
8CH DI	M1081	8	NPN (low level trigger)
8CH DI	M1082	8	PNP (high level trigger)
16CH DI	M1161	16	NPN (low level trigger)
16CH DI	M1162	16	PNP (high level trigger)
4CH DO	M2044	4	Relay
8CH DO	M2081	8	PNP
8CH DO	M2082	8	NPN
16CH DO	M2161	16	PNP
16CH DO	M2162	16	NPN
4CH AI Single-Ended	M3041	4	0-20mA/4-20mA
4CH AI Single-Ended	M3043	4	0-5V/0-10V
4CH AI Differential	M3044	4	0-5V/0-10V
4CH AI Differential	M3046	4	±5V/±10V
4CH AO	M4041	4	0-20mA/4-20mA
4CH AO	M4043	4	0-5V/0-10V
4CH AO	M4046	4	±5V/±10V
2CH RTD	M5021	2	3Wire PT100
2CH RTD	M5022	2	3Wire PT1000
2CH RTD	M5023	2	4Wire PT100
2CH RTD	M5024	2	4Wire PT1000
4CH TC	M5048	4	TC(B/E/J/K/N/R/S/T)
2CH RS485	M6021	2	RS485
2CH RS232	M6022	2	RS232
1CH RS485, 1CH RS232	M6023	2	RS485+RS232

Power module	M7011	/	/
Terminal module	M8011	/	/

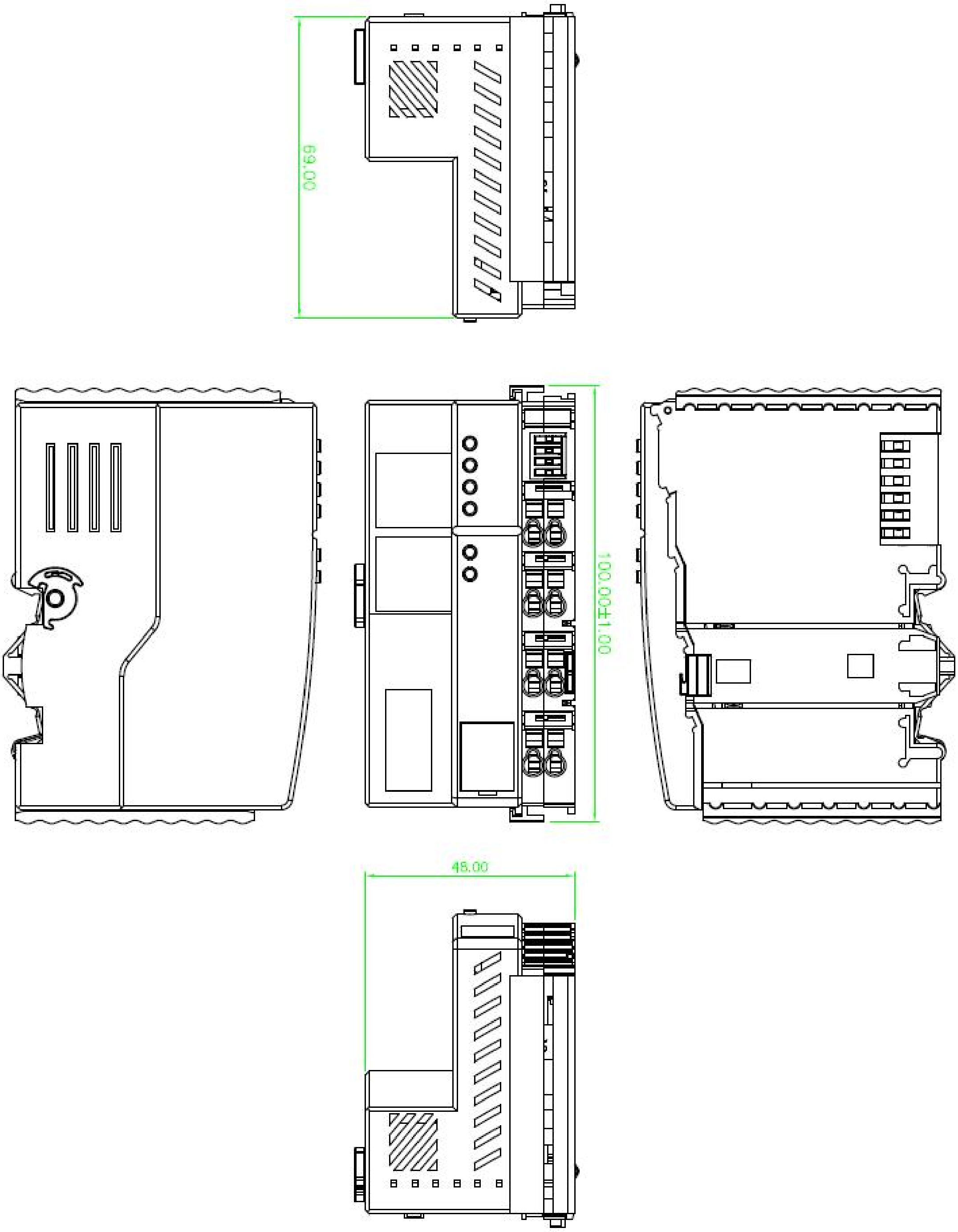
2 Hardware

2.1 I/O Coupler



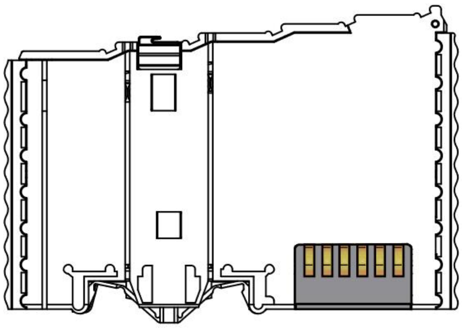
2.2 Dimension

Unit:mm



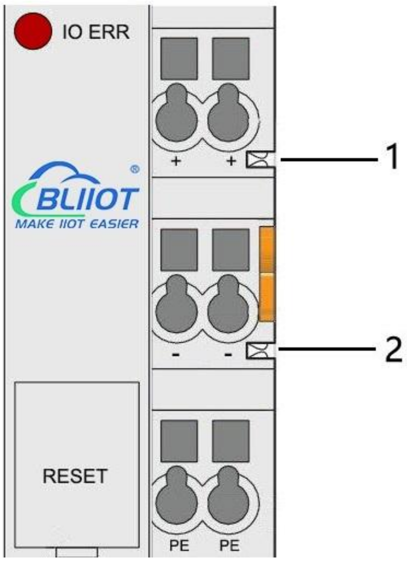
2.3 Data Contacts/Internal Bus

The communication between the fieldbus coupler/controller and the I/O modules, as well as the system power supply of the I/O modules are realized via the internal bus. The internal bus is made up of 6 data contacts, these gold-plated contacts are self-cleaning when connected.



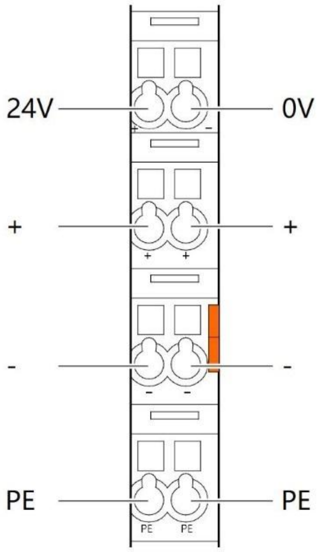
2.4 Power Jumper Contacts

The power module included with the coupler has two self-cleaning power jumper contacts for powering the field side. This power supply has a maximum current of 10A across the contacts, current exceeding the maximum will damage the contacts. When configuring the system, it must be ensured that the above-mentioned maximum current is not exceeded. If it exceeds, a power expansion module needs to be inserted.



No.	Type	Description
1	Spring contact	Supply 24V to the field side
2	Spring contact	Supply 0V to the field side

2.5 Terminal Point



Name	Description
24V	System Power 24VDC
0V	System Power 0VDC
+	Connections Field Supply 24 VDC
+	Connections Field Supply 24 VDC
-	Connections Field Supply 0 VDC
-	Connections Field Supply 0VDC
PE	Grounding
PE	Grounding

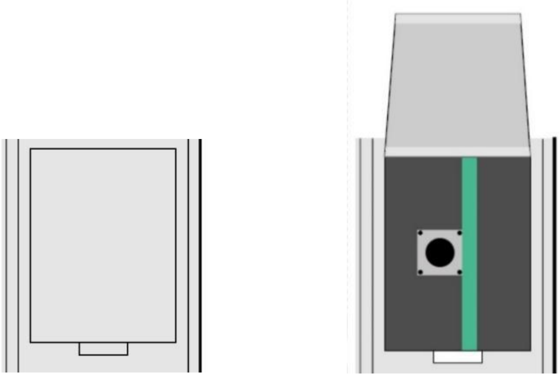
2.6 Factory Reset

This reset button is used to restore the device configuration parameters to the factory state.

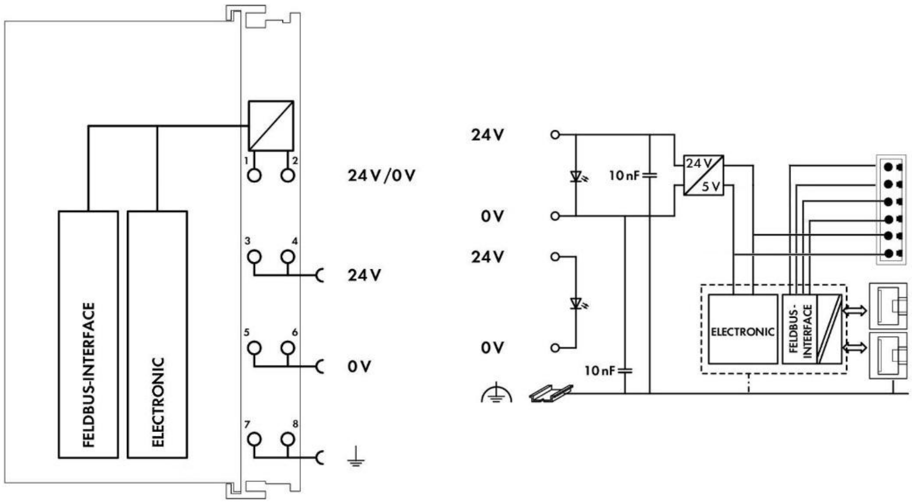
Operation steps:

1.When the device is running, open the flip cover;

2.Press and hold the button for more than 5 seconds, until all the LED lights go off, indicates reset successful, and then the device will automatically restart.



2.7 Electrical Schematic



3 Installation

3.1 Installation Sequence

All distributed couplers/controller and I/O modules from Beilai Technology must be mounted on a standard DIN 35 rail.

Starting from the coupler, the I/O modules are assembled from left to right, and the modules are installed next to each other. All I/O modules have grooves and power jumper contacts on the right side, to avoid assembly errors, I/O modules must be inserted from the right and top to avoid damage to the modules.

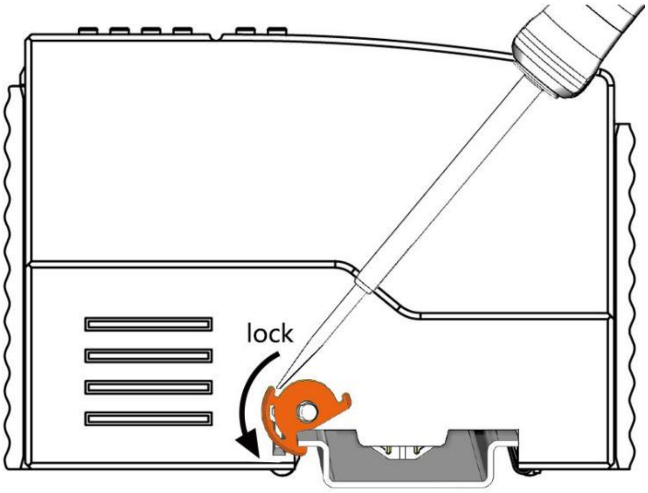
Utilizes a tongue and groove system to form a secure fit and connection. With the automatic locking function, the individual components are securely fixed on the rail after installation.

Don't forget to install the terminal module! Always plug a terminal module (eg TERM) into the end of the I/O module to ensure correct data transmission.

3.2 Install Coupler

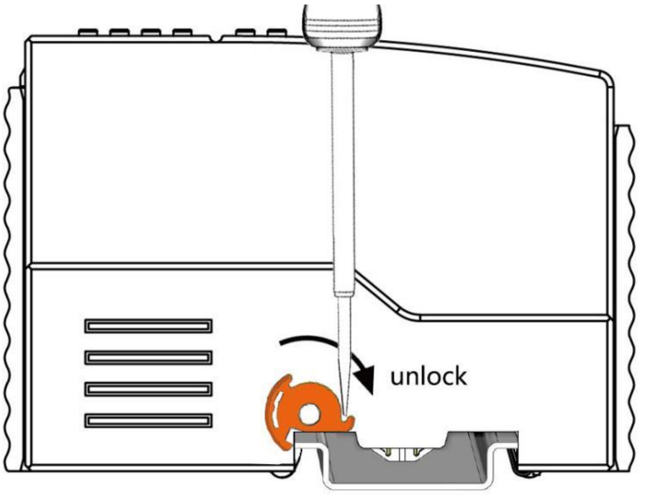
1.Snap the coupler onto the DIN rail first;

2.Use a tool such as a screwdriver to turn the locking cam until the locking cam engages the DIN rail.

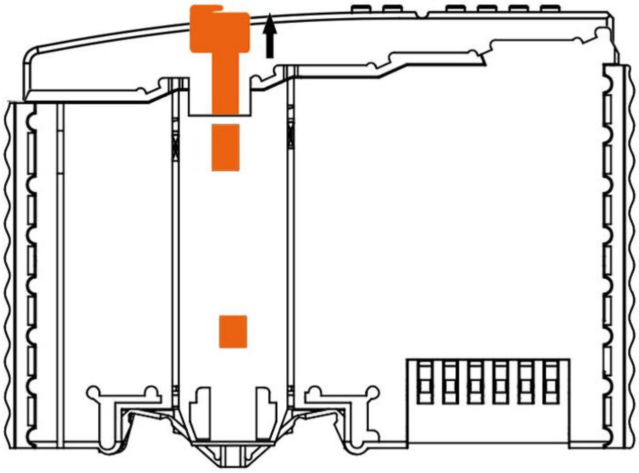


3.3 Remove Controller

1.Use a screwdriver to turn the locking disc cam until the locking cam no longer engages the rail.



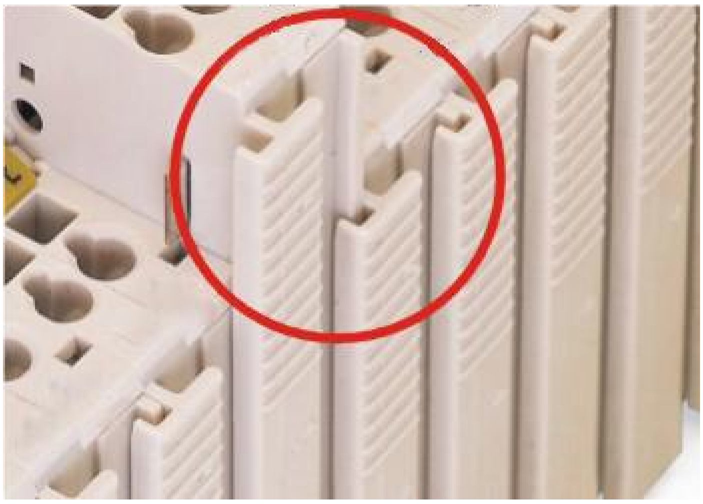
2.Pull the release tab to remove the coupler from the assembly



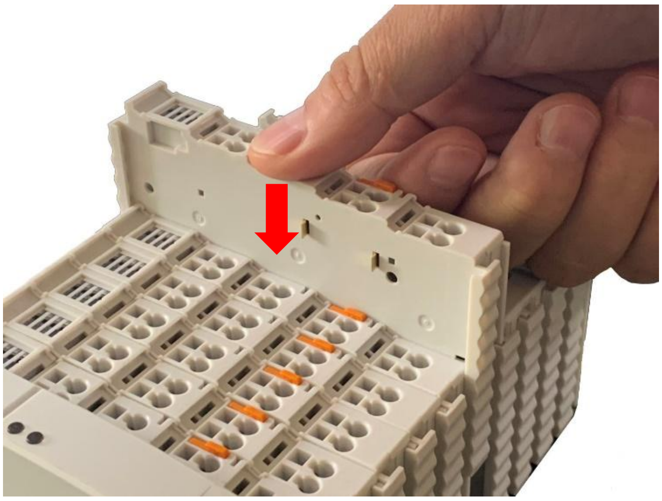
Data or power contacts are electrically disconnected from adjacent I/O modules when the coupler/controller is removed.

3.4 Insert I/O Modules

1.When inserting the module, make sure the tabs on the module line up with the grooves of the coupler or other I/O module to which it is attached.



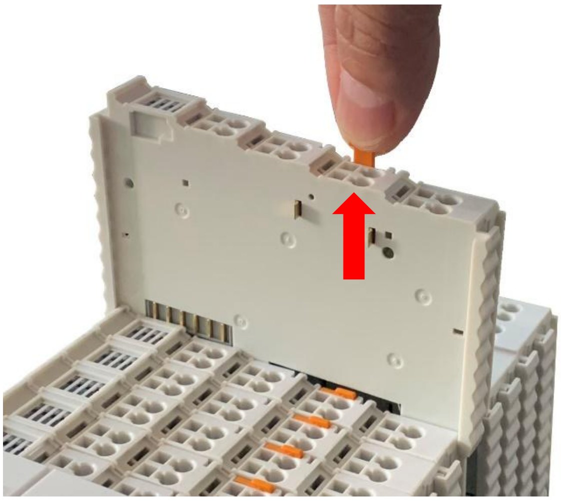
2.Press the I/O module into the assembly position until the I/O module snaps into the rail.



After the I/O module is installed, the electrical connection to the coupler (or the previous I/O module) and the following I/O module is established via the data contacts and the power jumper contacts.

3.5 Remove I/O Modules

Pull up on the latch to remove the I/O module from the assembly.



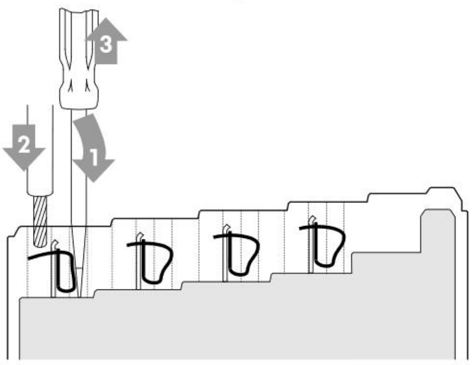
When the I/O module is removed, the electrical connection to the data or power jumper contacts is disconnection.

4 Device Connection

4.1 Wiring

CAGE CLAMP connection is suitable for solid, stranded and fine-stranded conductors. Only one wire can be connected to each CAGE CLAMP. If there is more than one wire, it must be merged into a point before being connected.

1. Open the CAGE CLAMP by inserting the tool into the opening above the junction.
2. Insert the wire into the corresponding open connection terminal.
3. Once the tool is removed, the CAGE CLAMP closes and the wire is clamped firmly by the spring.



4.2 Power Supply

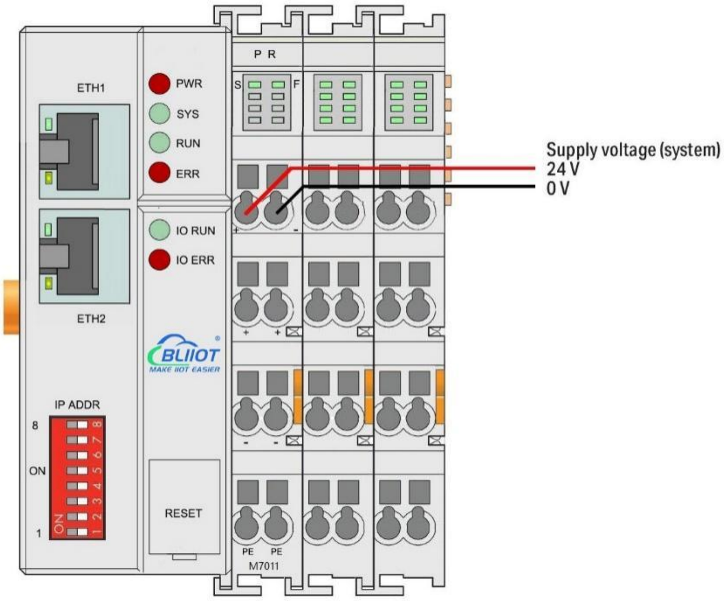
System and field voltages are supplied by power supply modules. The power supply module of the BL200 coupler supplies power for the internal electronics of the coupler and the I/O modules. If necessary (there are many I/O modules and the current is relatively high), it can also be provided through an independent power supply module. The fieldbus interface (Ethernet interface), system and field are galvanically isolated from each other.

4.2.1 System Power

BL200 series couplers require 24V DC system power, which is connected from the terminal of the power supply module. The 5V bus voltage required inside the system is converted from the 24V system voltage.

The power supply module only has proper fuse protection, please provide proper overcurrent protection externally.

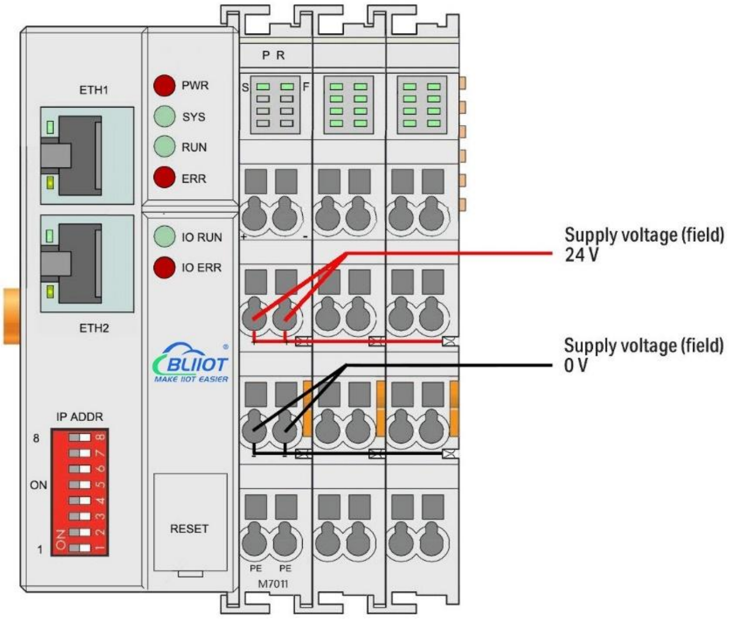
Please pay attention to matching the output power of the power supply module and the load power to avoid excessive load current.



4.2.2 On-site Power Supply

The power supply module supplies 24 VDC on the field side to power the sensors and actuators.

Field power supply only has proper fuse protection. Without overcurrent protection, electronic equipment can be damaged.



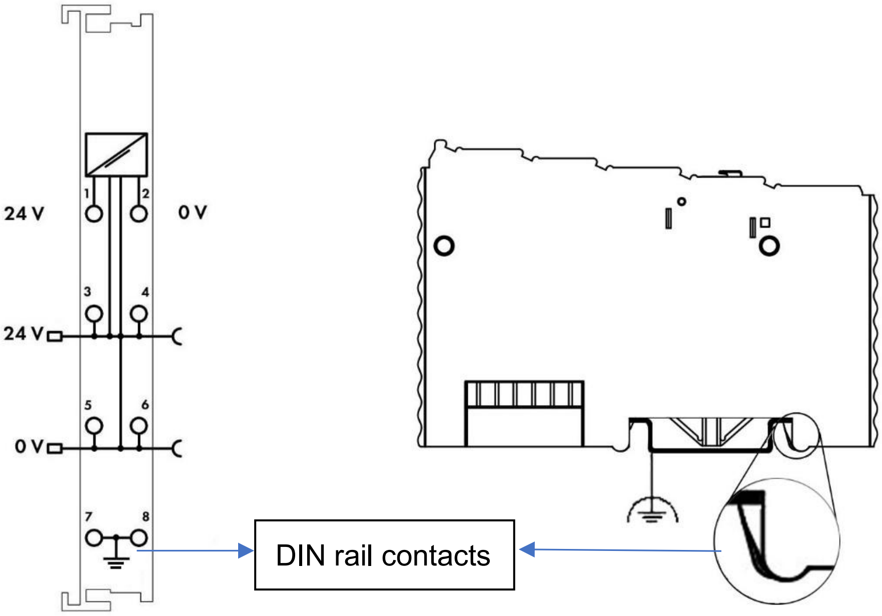
Field-side power is automatically output from the power jumper contact when the I/O module is connected. The continuous load current across the contacts of the power supply must not exceed 10 A.

The problem of excessive load power on the system side or on the field side can be solved by plugging in additional power supply modules. After plugging in an additional power supply module, a new voltage potential may appear on the field side.

In the case where electrical isolation is not required, the field power supply and the system power supply can use the same power supply.

4.2.3 Grounding

When installing the enclosure cabinet, the cabinet must be grounded, and the rail is electrically connected to the cabinet through screws to ensure that the rail is properly grounded. Grounding can increase resistance to electromagnetic interference. Some components in the I/O system have rail contacts that dissipate EMI onto the rail.



5 BL200 Modbus TCP Coupler

5.1 BL200 Coupler Overview

The Modbus TCP coupler supports standard Modbus TCP server communication, and the Ethernet supports the dual network port switch cascading function. The device supports simultaneous access by 15 clients, supports function code 01/02/03/04/05/06/15/16, and supports 32 extended I/O modules.

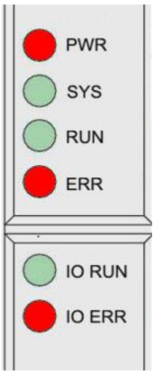
5.2 Technical Parameters

Name	Parameter	Description
System power	Input voltage(system)	24 VDC
	Input current(system)	MAX 500 mA@24VDC
	Power Efficiency	84%
	Internal bus voltage	5VDC
	Coupler current consumption	MAX 300mA@5VDC
	I/O current consumption	MAX 1700mA@5VDC
	Isolation protection	500 V system/supply
Field power	Input voltage (field)	24 VDC
	Current carrying capacity (power jumper contacts)	Max 10 ADC
Ethernet	Number	2 X RJ45
	Transmission medium	Twisted Pair STP 100 Ω Cat 5
	MAX cable length	100m
	Baud rate	10/100 Mbit/s
	Isolation protection	ESD contact: 8KV, Surge: 4KV(10/1000us)
System	Operating system	Linux
	CPU	300MHz
	RAM	64MB
	Flash	128MB
	I/O Modules	MAX 32
	Process mapping (Modbus) data points via serial port module	● Bool : 4096 ● 16 Bit : 2048 ● 32 Bit : 1024
	Protocol	Modbus TCP, HTTP, DHCP, DNS
	Maximum number of connections	15 Modbus TCP
Connection	Method	CAGE CLAMP
	Wire diameter	0.08 mm² ··· 2.5 mm², AWG 28 ··· 14
	Strip length	8 mm ··· 9 mm / 0.33 in
Environment	Working temperature	0 ··· 55 ° C
	Storage temperature	-40 ··· 70 ° C
	Relative humidity	5 ··· 95% no condensation

	Working altitude	0 ··· 2000 m
	Protection type	IP20
Dimension	Width	48mm
	Length	100mm
	Height	69mm
Material	Color	Light gray
	Housing material	Polycarbonate, Nylon 6.6
	Fire load	1.239 MJ
	Weight	180g
Mechanical	Mounting type	DIN-35 rail
Certificates	EMC	EN 55022: 2006/A1: 2007 (CE &RE) Class B
		IEC 61000-4-2 (ESD) Level 4
		IEC 61000-4-3 (RS) Level 4
		IEC 61000-4-4 (EFT) Level 4
		IEC 61000-4-5 (Surge)Level 3
		IEC 61000-4-6 (CS)Level 4
		IEC 61000-4-8 (M/S) Level 4

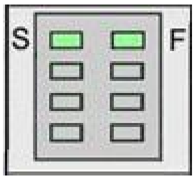
5.3 Hardware Interface

5.3.1 LED Indicators



LED	Description	Color	Status	Meaning
PWR	Power indicator	Red	ON	Power connection successful
			OFF	No power
SYS	System indicator	Green	ON	System is abnormal
			OFF	System is running normally

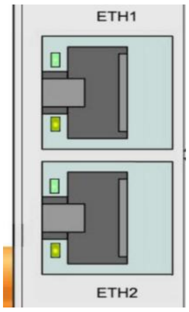
RUN	Running indicator	Green	Flashing	System is running normally
			OFF	System is abnormal
ERR	Error indicator	Red	ON	Northbound protocol connection error
			OFF	No errors
I/O RUN	I/O Running indicator	Green	Flashing	I/O module is working normally
			OFF	Module not inserted
I/O ERR	I/O Error indicator	Red	ON	I/O module communication error
			OFF	No errors



LED	Description	Color	Status	Meaning
S	System 24V power indicator	Green	ON	Power is OK
			OFF	No power
F	Field 24V power indicator	Green	ON	Power is OK
			OFF	No power

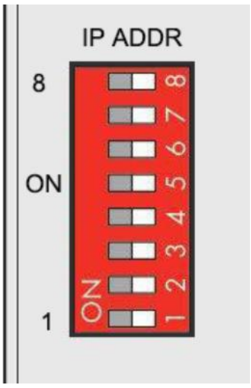
5.3.2 Ethernet Port

Connect to the Ethernet-based fieldbus through ETH2.
EHT1 is used to connect other nodes that need to be connected to the Ethernet.



5.3.3 IP Address Selection Switch

The 8-bit DIP switch is used to set the IP address. The encoding of DIP switches is done bit by bit, starting from DIP switch 1 with the least significant bit (2^0) to DIP switch 8 with the most significant bit (2^7), corresponding to decimal values: 0-255.



When the value of the DIP switch is 1111 1111 (decimal 255), the IP address is set according to the web page. The web page setting can specify the IP or set the automatic acquisition. When the web page is not set, the IP address is: 192.168.1.10
When the value of the DIP switch is 0000 0000 – 1111 1110 (decimal 0-254), determine the 3rd byte of the IP address, and the 1st, 2nd and 4th bytes are fixed bytes, namely 192.168.xxx.253

5.4 Modbus Register Mapping

The internal register map of BL200 coupler node consists of 2 parts, one part is the data map of digital input and output and analog input and output module, the address range is 1000...9999; the other part is the serial port module, the address range is 10000... 49999
The state of digital and analog I/O modules can be determined or changed through the register map (Address 1000 ... 9999).

Modbus address		Data type	Access type	Function code	Description
decimal	hex				
1000…1999	0x03 E8…0x07 CF	1 Bit	read/write	0x01/05/0F	Digital output
2000…2999	0x07 D0…0x0B B7	1 Bit	read	0x02	Digital input
3000…3999	0x0B B8…0x0F 9F	32 Bit Float	read	0x04	Analog input
4000…4999	0x0F A0…0X13 87	32 Bit Float	read/write	0x03/06/10	Analog output
5000…8999	0x13 88…0x23 27	32 Bit Unint	read/write	0x03/04/10	DI count value
9000…9999	0x23 28…0x27 0F	1 Bit	read	0x02	Module power-on status

Determine or change the state of the data mapped from the serial I/O module through address 10000 ... 49999

Modbus address		Data type	Access type	Functio n code	Description
decimal	hex				
10000…19999	0x27 10…0x4E 1F	1 Bit	read/write	0x01/05/0F	Digital output
20000…29999	0x4E 20…0x75 2F	1 Bit	read	0x02	Digital input
30000…39999	0x75 30…0x9C 3F	16 Bit	read	0x04	Analog input
40000…49999	0x9C 40…0XC3 4F	16 Bit	read/write	0x03/06/10	Analog output

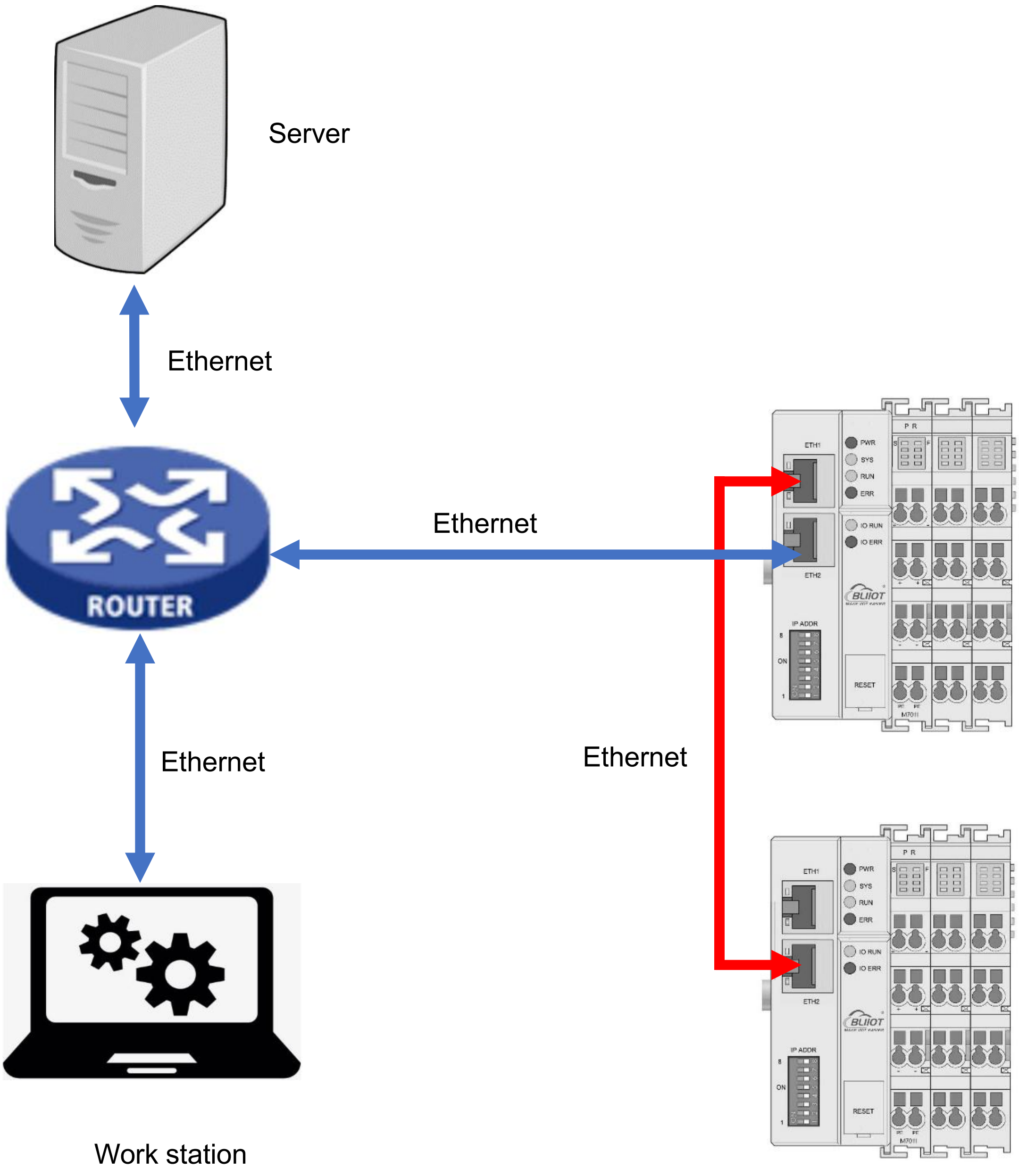
5.5 Coupler Connection

The BL200 coupler comes with 2 x RJ45 Ethernet ports, integrated switch function inside, work in store-and-forward operation mode, each port supports 10/100 Mbit transmission speed and full-duplex and half-duplex transmission mode.

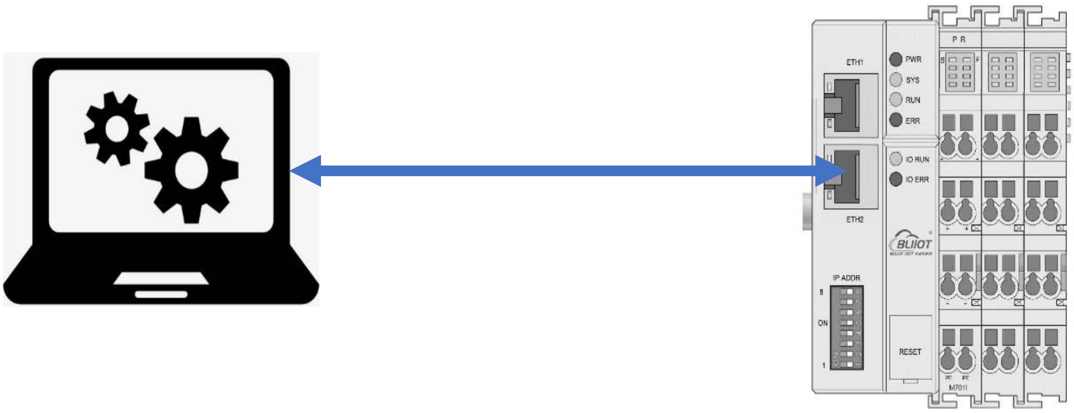
The BL200 coupler connect to the router Ethernet network via ETH2 only, while the EHT 1 is for connecting other BL200 field nodes.

The internal integrated switch supports bypass mode, which can automatically start the bypass mode when the controller system fails, and automatically maintain the link between ETH1 and EHT2.

The wiring of these Ethernet ports conforms to the 100BaseTX specification, which specifies the use of category 5 twisted pair cable as the connecting cable. Cable types S/UTP (Screened unshielded twisted pair) and STP (shielded twisted pair) can be used up to a length of 100m.



Directly connected to the computer through ETH 2.



5.6 Web Page Configuration

The BL200 coupler's built-in web server is a browser-based configuration utility. When the coupler is connected to your network, you can enter the server's IP address in a web browser to access the web console.

5.6.1 Preparation Before Configuration

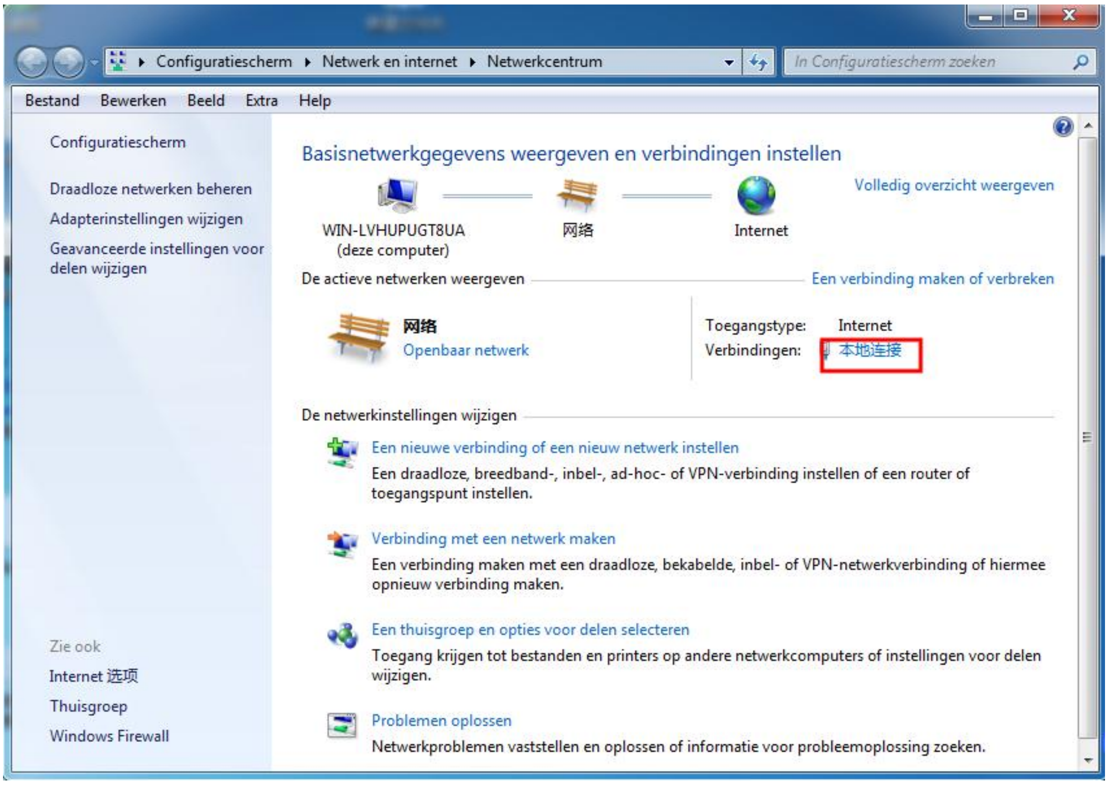
To successfully access the BL200 coupler, it must be properly installed and connected to the computer. In addition, configure them with correct IP addresses to keep them in the same network segment.

5.6.1.1 Connect Computer and Coupler

1.Mount the fieldbus node on a DIN35 rail. Follow the installation instructions in the "Installation" chapter.
2.Connect the 24 V power supply to the system power terminals.
3.The computer and the bus node can be connected in two ways, one is that the two are connected to the switch device of the local area network through the Ethernet port; the other is that the two are directly connected point-to-point. For detailed steps, follow the instructions in the "Coupler Connection" chapter.
4.Turn on the power supply and start supplying power.
The coupler is initialized after power-up, creates process image according to the I/O modules configuration of the fieldbus node.

5.6.1.2 Configure Computer IP Address

There are two ways to configure PC IP address. One is to turn on the automatic IP address option on the PC's local connection to dynamically assign DHCP in the network. The other is to configure a static IP address with the coupler node on the same network segment on the local connection of the PC.
Takes Windows 7 system as an example for configuration. Windows systems are all configured similarly.
1.Click Start > Control Panel > Network and Sharing Center, and click local connection in the window that opens.


Configuratiescherm ▸ Netwerk en internet ▸ Netwerkcentrum
In Configuratiescherm zoeken
Bestand Bewerken Beeld Extra Help
Configuratiescherm
Draadloze netwerken beheren
Adapterinstellingen wijzigen
Geavanceerde instellingen voor delen wijzigen
Zie ook
Internet 选项
Thuisgroep
Windows Firewall
WIN-LVHUPUGT8UA (deze computer)
网络
Internet
Volledig overzicht weergeven
Een verbinding maken of verbreken

Basisnetwerkgegevens weergeven en verbindingen instellen

De actieve netwerken weergeven

网络
Openbaar netwerk

Toegangstype: Internet
Verbindingen: 本地连接

De netwerkinstellingen wijzigen

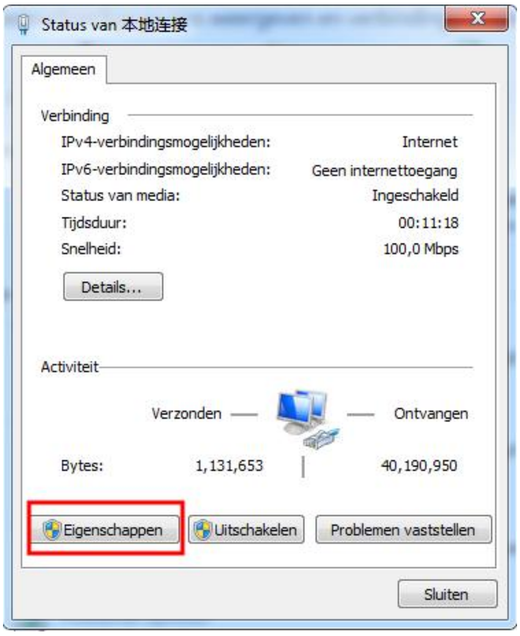
Een nieuwe verbinding of een nieuw netwerk instellen
Een draadloze, breedband-, inbel-, ad-hoc- of VPN-verbinding instellen of een router of toegangspunt instellen.

Verbinding met een netwerk maken
Een verbinding maken met een draadloze, bekabelde, inbel- of VPN-netwerkverbinding of hiermee opnieuw verbinding maken.

Een thuisgroep en opties voor delen selecteren
Toegang krijgen tot bestanden en printers op andere netwerkcomputers of instellingen voor delen wijzigen.

Problemen oplossen
Netwerkproblemen vaststellen en oplossen of informatie voor probleemoplossing zoeken.

2.In the local connection status window, click Properties.



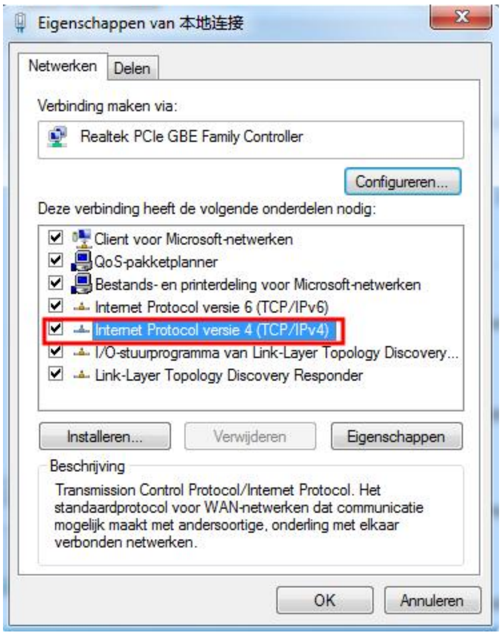
Verbinding

IPv4-verbindingsmogelijkheden:	Internet
IPv6-verbindingsmogelijkheden:	Geen internettoegang
Status van media:	Ingeschakeld
Tijdsduur:	00:11:18
Snelheid:	100,0 Mbps

Activiteit

	Verzonden	Ontvangen
Bytes:	1,131,653	40,190,950

3.Double-click "Internet Protocol Version 4 (TCP/IPv4)" on the local connection properties page.



Verbinding maken via:

Realtek PCIe GBE Family Controller

Deze verbinding heeft de volgende onderdelen nodig:

- ☑ Client voor Microsoft-netwerken
- ☑ QoS-pakketplanner
- ☑ Bestands- en printerdeling voor Microsoft-netwerken
- ☑ Internet Protocol versie 6 (TCP/IPv6)
- ☑ Internet Protocol versie 4 (TCP/IPv4)
- ☑ I/O-stuurprogramma van Link-Layer Topology Discovery...
- ☑ Link-Layer Topology Discovery Responder

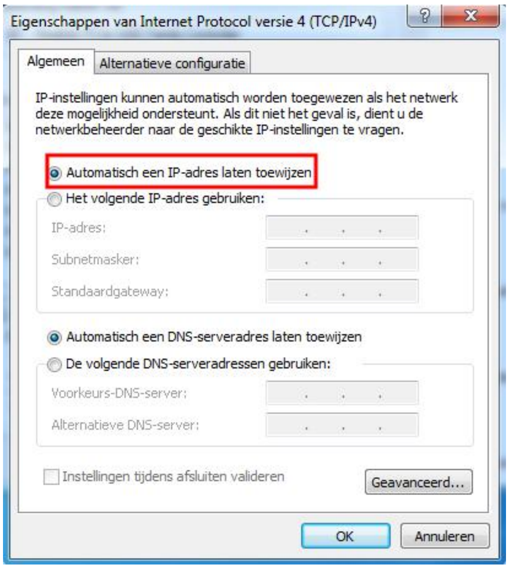
Beschrijving

Transmission Control Protocol/Internet Protocol. Het standaardprotocol voor WAN-netwerken dat communicatie mogelijk maakt met andersoortige, onderling met elkaar verbonden netwerken.

4.There are two ways to configure the IP address of the PC

- Obtain IP address automatically (system default mode)

  To obtain an IP address automatically from a DHCP server, select "Obtain an IP address automatically";


Eigenschappen van Internet Protocol versie 4 (TCP/IPv4)
Algemeen
Alternatieve configuratie
Geavanceerd...
OK
Annuleren

IP-instellingen kunnen automatisch worden toegewezen als het netwerk deze mogelijkheid ondersteunt. Als dit niet het geval is, dient u de netwerkbeheerder naar de geschikte IP-instellingen te vragen.

● Automatisch een IP-adres laten toewijzen

○ Het volgende IP-adres gebruiken:

IP-adres: . . .

Subnetmasker: . . .

Standaardgateway: . . .

● Automatisch een DNS-serveradres laten toewijzen

○ De volgende DNS-serveradressen gebruiken:

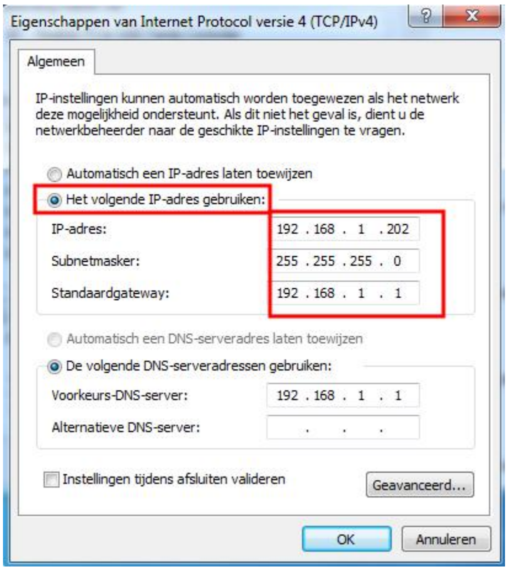
Voorkeurs-DNS-server: . . .

Alternatieve DNS-server: . . .

□ Instellingen tijdens afsluiten valideren

- Set a static IP address

  Select "Use the following IP address" and set the correct values for the IP address, subnet mask and default gateway.



Eigenschappen van Internet Protocol versie 4 (TCP/IPv4)

Algemeen

IP-instellingen kunnen automatisch worden toegewezen als het netwerk deze mogelijkheid ondersteunt. Als dit niet het geval is, dient u de netwerkbeheerder naar de geschikte IP-instellingen te vragen.

○ Automatisch een IP-adres laten toewijzen

● Het volgende IP-adres gebruiken:

IP-adres:	192 . 168 . 1 . 202
Subnetmasker:	255 . 255 . 255 . 0
Standaardgateway:	192 . 168 . 1 . 1

○ Automatisch een DNS-serveradres laten toewijzen

● De volgende DNS-serveradressen gebruiken:

Voorkeurs-DNS-server:	192 . 168 . 1 . 1
Alternatieve DNS-server:	. . .

□ Instellingen tijdens afsluiten valideren

5.6.1.3 Configure Coupler IP address

There are 2 ways to assign an IP address

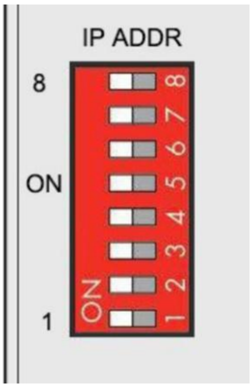
- Assignment via built-in web page (static IP or automatic IP assignment)
- Assign via DIP switch (static IP)

DIP address selector switch definition

Switch position （ON = 1）	**Value**	**Definition**
0000 0000 --- 1111 1110	0-254	Enable the DIP selector switch assignment function and determine the value of the 3rd byte. Example: 0010 0110 (22 decimal), the IP address is "192.168.22.253".
1111 1111	255	Enable the function of specifying IP on the web page, or select the function of DHCP automatic allocation. When the IP is not allocated through the web, the IP is 192.168.1.10.

5.6.1.3.1 Configuration via Web Page

The fieldbus coupler can be set to an IP address via the "Settings > Local Settings" page after entering the page, or it can be set to be assigned automatically. Select static address, if not set IP address, the IP is 192.168.1.10

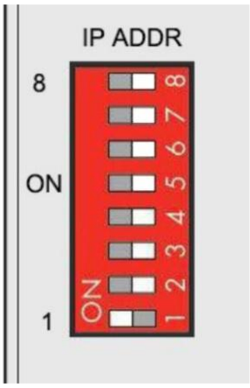


5.6.1.3.2 Assign IP via DIP Switch

Set the value of the DIP address selector switch to 0000 0000 - 1111 1110 (decimal 0 - 254), and the IP address will be assigned by the DIP switch.

The IP address consists of fixed bytes and variable bytes. The 1st, 2nd and 4th bytes are fixed bytes, the DIP selector switch determines the 3rd byte, namely: 192.168.xxx.253

The fieldbus coupler assigns an IP address via a DIP switch, and the IP address set in this way is static.



5.6.1.4 Factory Default Settings

Before logging into the web configuration page, it is necessary for you to understand the following default parameters,

Modbus TCP Server Port: 502, Modbus ID: 1

IP: Determined according to the DIP switch, if the DIP switch is 1111 1111, the default IP is 192.168.1.10

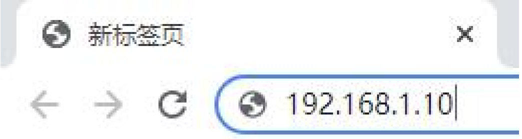
If factory default DIP switch is 0000 0000 status, then the IP is 192.168.0.253

Item	Description
Username	admin
Password	Empty

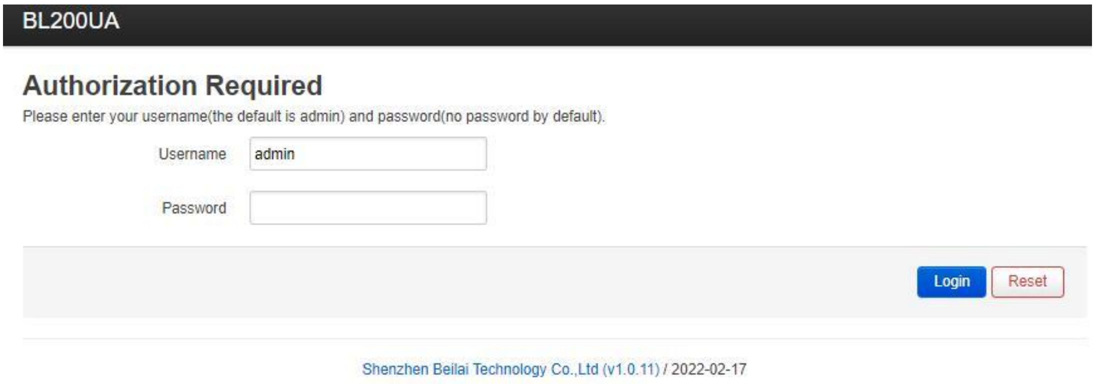
5.6.2 Login Configuration Page

1.Open a browser on your computer, such as IE, Chrome, etc.

2.Enter the IP address of the coupler node (192.168.1.10) in the address bar of the browser to enter the user login interface.



3.Enter "Username" and "Password" in the login interface, and then click Login.



BL200UA

Authorization Required

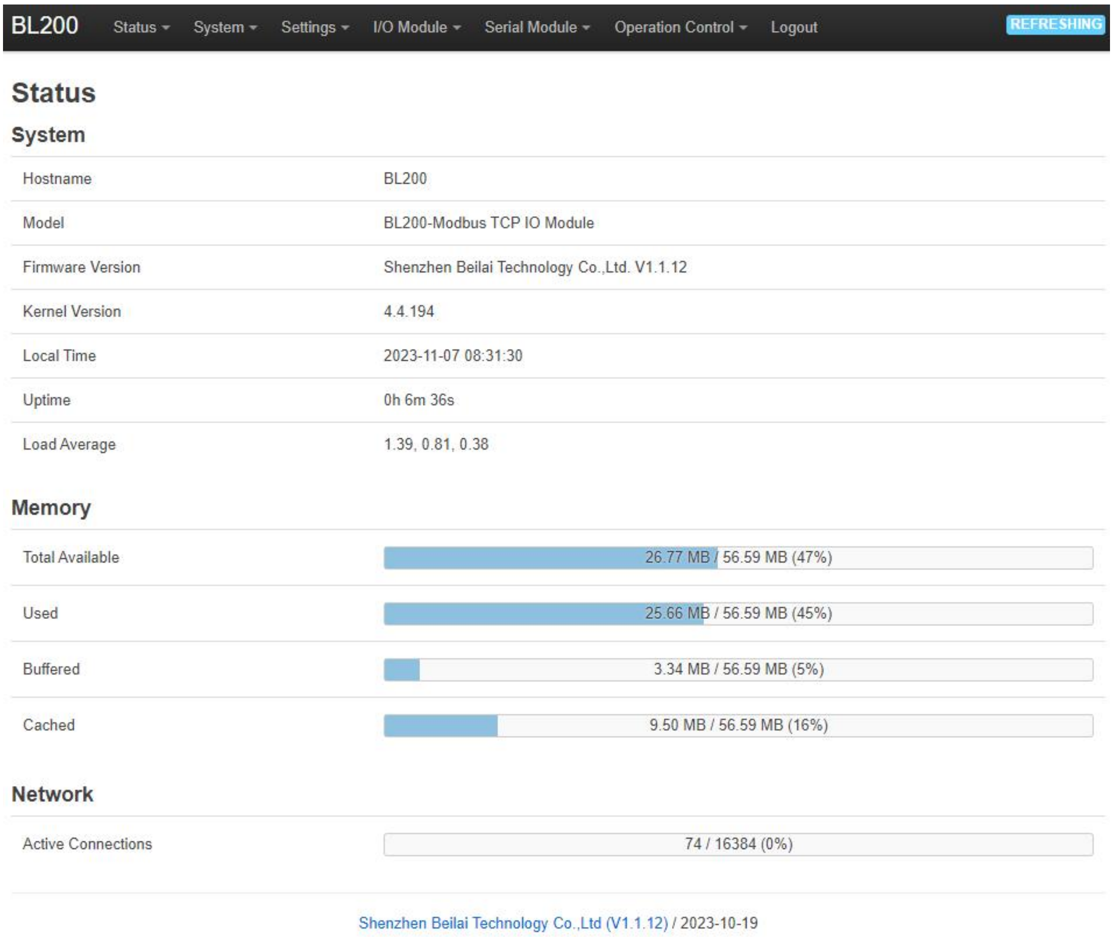
Please enter your username(the default is admin) and password(no password by default).

Username: admin

Password:

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

4.After successfully logging in to the web interface, the display is as follows



Status

System

Hostname	BL200
Model	BL200-Modbus TCP IO Module
Firmware Version	Shenzhen Beilai Technology Co.,Ltd. V1.1.12
Kernel Version	4.4.194
Local Time	2023-11-07 08:31:30
Uptime	0h 6m 36s
Load Average	1.39, 0.81, 0.38

Memory

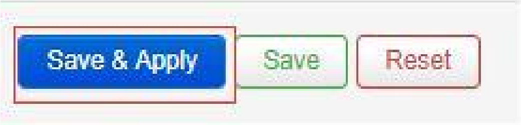
Total Available	26.77 MB / 56.59 MB (47%)
Used	25.66 MB / 56.59 MB (45%)
Buffered	3.34 MB / 56.59 MB (5%)
Cached	9.50 MB / 56.59 MB (16%)

Network

Active Connections	74 / 16384 (0%)

Shenzhen Beilai Technology Co.,Ltd (V1.1.12) / 2023-10-19

5.After configuring the parameters, you need to click the "Save and Apply" button on the page to take effect.

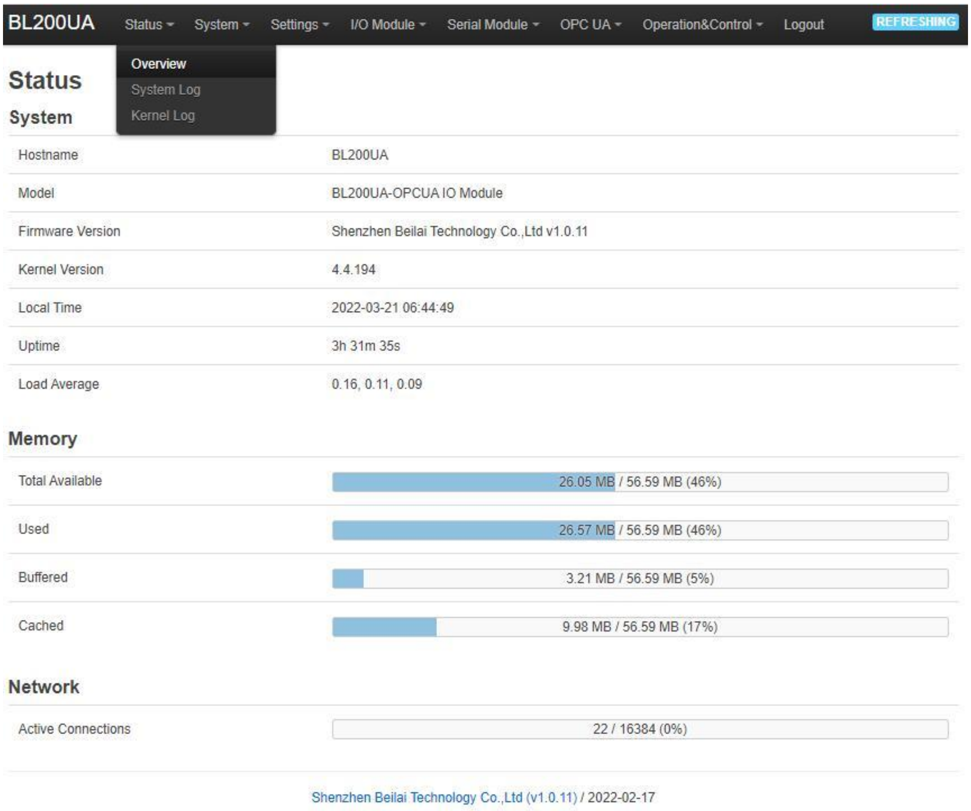


5.7 Web Configuration Page Description

5.7.1 Status

Users can check overview, system log and kernel log, as well as device parameters and device operating status.

Status > Overview



Status

System

Hostname	BL200UA
Model	BL200UA-OPCUA IO Module
Firmware Version	Shenzhen Beilai Technology Co.,Ltd v1.0.11
Kernel Version	4.4.194
Local Time	2022-03-21 06:44:49
Uptime	3h 31m 35s
Load Average	0.16, 0.11, 0.09

Memory

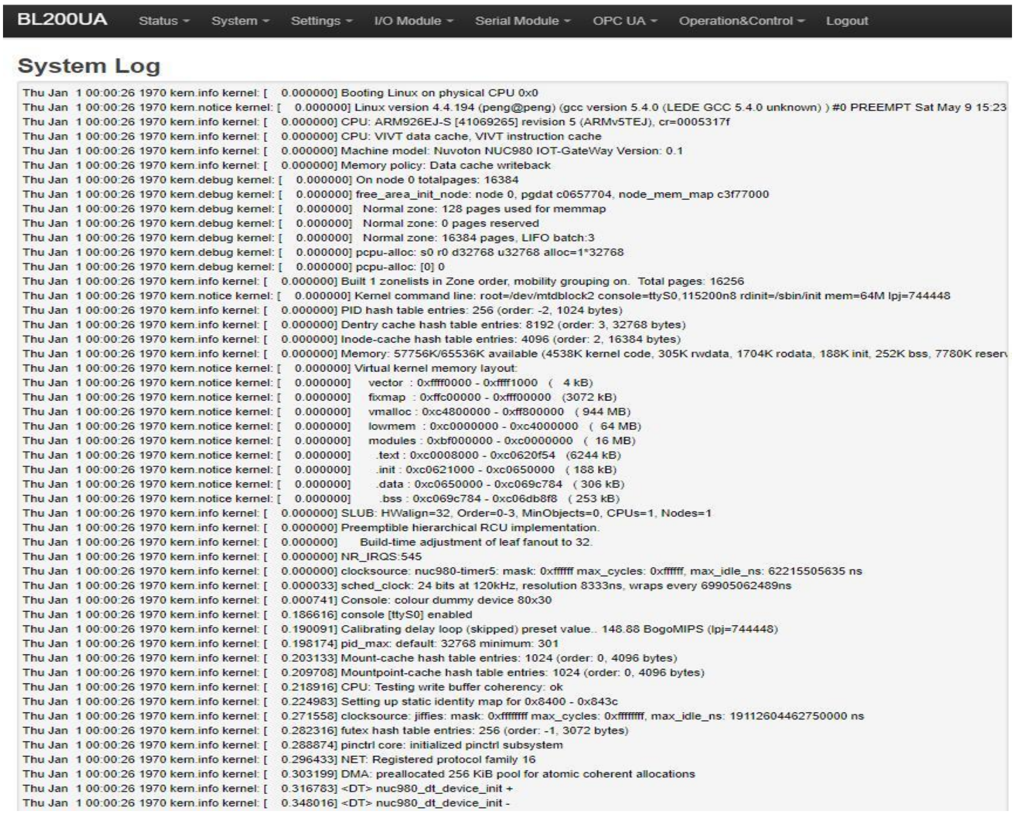
Total Available	26.05 MB / 56.59 MB (46%)
Used	26.57 MB / 56.59 MB (46%)
Buffered	3.21 MB / 56.59 MB (5%)
Cached	9.98 MB / 56.59 MB (17%)

Network

Active Connections	22 / 16384 (0%)

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

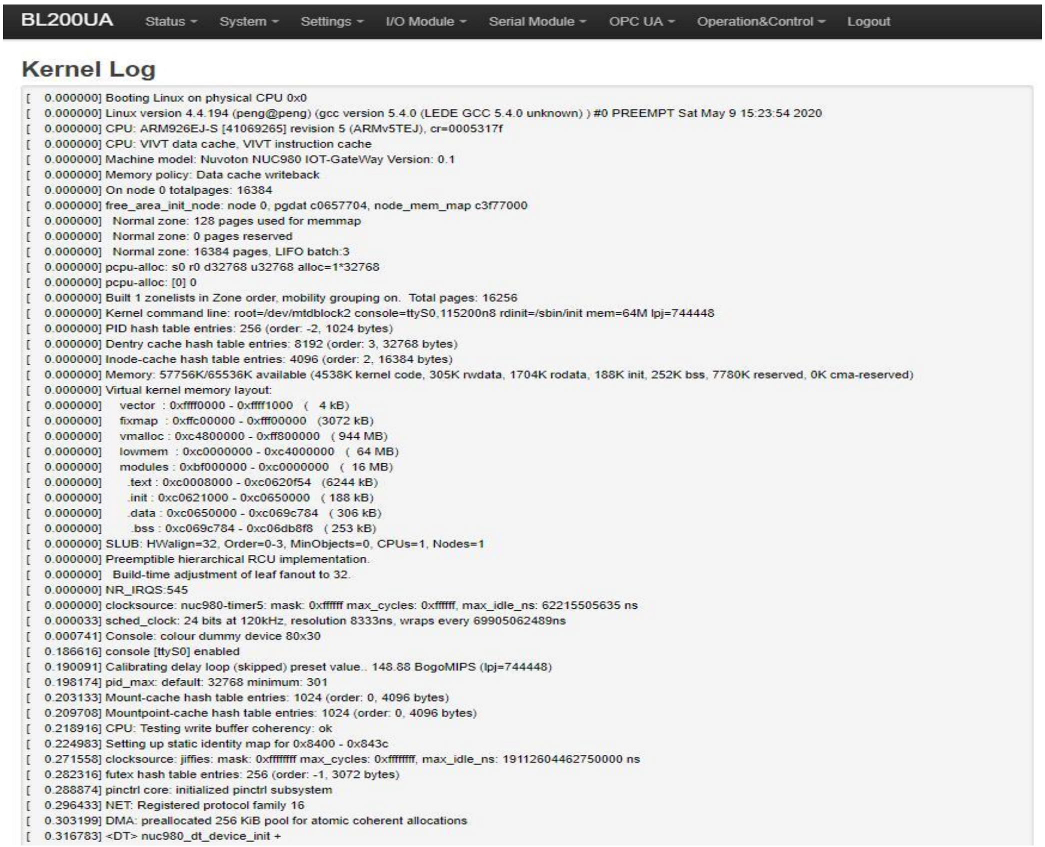
Status > System Log



System Log

```
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Booting Linux on physical CPU 0x0
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000] Linux version 4.4.194 (peng@peng) (gcc version 5.4.0 (LEDE GCC 5.4.0 unknown) ) #0 PREEMPT Sat May 9 15:23
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] CPU: ARM926EJ-S [41069265] revision 5 (ARMv5TEJ), cr=0005317f
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] CPU: VIVT data cache, VIVT instruction cache
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Machine model: Nuvoton NUC980 IOT-GateWay Version: 0.1
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Memory policy: Data cache writeback
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000] On node 0 totalpages: 16384
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000] free_area_init_node: node 0, pgdat c0657704, node_mem_map c3f77000
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000]   Normal zone: 128 pages used for memmap
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000]   Normal zone: 0 pages reserved
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000]   Normal zone: 16384 pages, LIFO batch:3
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000] pcpu-alloc: s0 r0 d32768 u32768 alloc=1*32768
Thu Jan  1 00:00:26 1970 kern.debug kernel: [    0.000000] pcpu-alloc: [0] 0
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 16256
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000] Kernel command line: root=/dev/mtdblock2 console=ttyS0,115200n8 rdinit=/sbin/init mem=64M lpj=744448
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] PID hash table entries: 256 (order: -2, 1024 bytes)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Memory: 57756K/65536K available (4538K kernel code, 305K rwdata, 1704K rodata, 188K init, 252K bss, 7780K reserv
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000] Virtual kernel memory layout:
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]     vector  : 0xffff0000 - 0xffff1000   (   4 kB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]     fixmap  : 0xffc00000 - 0xfff00000   (3072 kB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]     vmalloc : 0xc4800000 - 0xff800000   ( 944 MB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]     lowmem  : 0xc0000000 - 0xc4000000   (  64 MB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]     modules : 0xbf000000 - 0xc0000000   (  16 MB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]       .text : 0xc0008000 - 0xc0620f54   (6244 kB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]       .init : 0xc0621000 - 0xc0650000   ( 188 kB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]       .data : 0xc0650000 - 0xc069c784   ( 306 kB)
Thu Jan  1 00:00:26 1970 kern.notice kernel: [    0.000000]        .bss : 0xc069c784 - 0xc06db8f8   ( 253 kB)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] SLUB: HWalign=32, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] Preemptible hierarchical RCU implementation.
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000]      Build-time adjustment of leaf fanout to 32.
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] NR_IRQS:545
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000000] clocksource: nuc980-timer5: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 62215505635 ns
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000033] sched_clock: 24 bits at 120kHz, resolution 8333ns, wraps every 69905062489ns
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.000741] Console: colour dummy device 80x30
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.186616] console [ttyS0] enabled
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.190091] Calibrating delay loop (skipped) preset value.. 148.88 BogoMIPS (lpj=744448)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.198174] pid_max: default: 32768 minimum: 301
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.203133] Mount-cache hash table entries: 1024 (order: 0, 4096 bytes)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.209708] Mountpoint-cache hash table entries: 1024 (order: 0, 4096 bytes)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.218916] CPU: Testing write buffer coherency: ok
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.224983] Setting up static identity map for 0x8400 - 0x843c
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.271558] clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 19112604462750000 ns
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.282316] futex hash table entries: 256 (order: -1, 3072 bytes)
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.288874] pinctrl core: initialized pinctrl subsystem
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.296433] NET: Registered protocol family 16
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.303199] DMA: preallocated 256 KiB pool for atomic coherent allocations
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.316783] <DT> nuc980_dt_device_init +
Thu Jan  1 00:00:26 1970 kern.info kernel: [    0.348016] <DT> nuc980_dt_device_init -
```

Status > Kernel Log



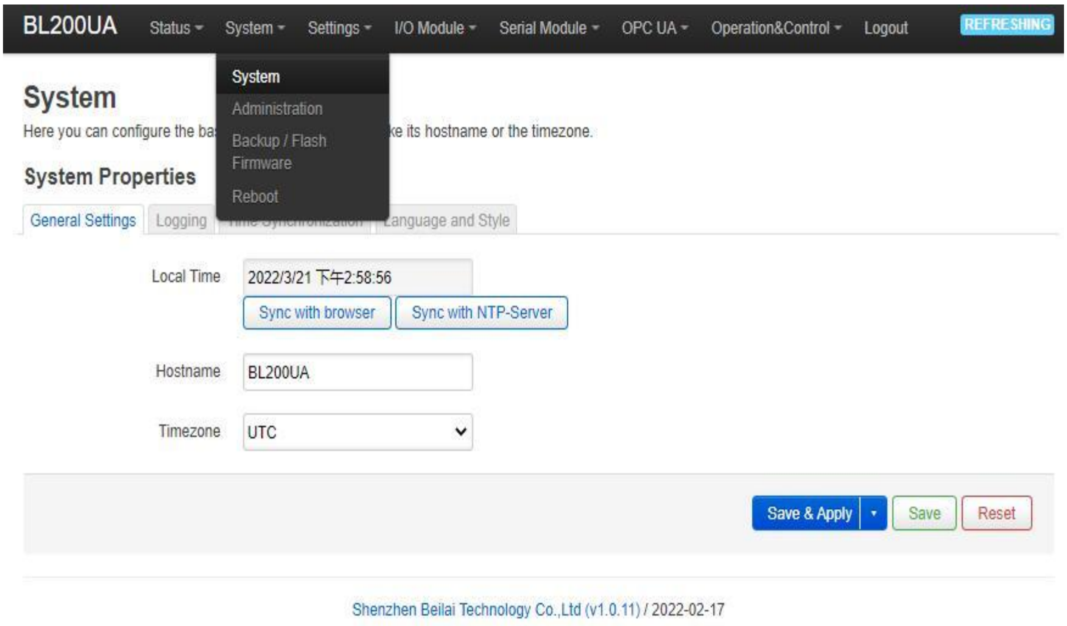
Kernel Log

```
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.4.194 (peng@peng) (gcc version 5.4.0 (LEDE GCC 5.4.0 unknown) ) #0 PREEMPT Sat May 9 15:23:54 2020
[    0.000000] CPU: ARM926EJ-S [41069265] revision 5 (ARMv5TEJ), cr=0005317f
[    0.000000] CPU: VIVT data cache, VIVT instruction cache
[    0.000000] Machine model: Nuvoton NUC980 IOT-GateWay Version: 0.1
[    0.000000] Memory policy: Data cache writeback
[    0.000000] On node 0 totalpages: 16384
[    0.000000] free_area_init_node: node 0, pgdat c0657704, node_mem_map c3f77000
[    0.000000]   Normal zone: 128 pages used for memmap
[    0.000000]   Normal zone: 0 pages reserved
[    0.000000]   Normal zone: 16384 pages, LIFO batch:3
[    0.000000] pcpu-alloc: s0 r0 d32768 u32768 alloc=1*32768
[    0.000000] pcpu-alloc: [0] 0
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 16256
[    0.000000] Kernel command line: root=/dev/mtdblock2 console=ttyS0,115200n8 rdinit=/sbin/init mem=64M lpj=744448
[    0.000000] PID hash table entries: 256 (order: -2, 1024 bytes)
[    0.000000] Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
[    0.000000] Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
[    0.000000] Memory: 57756K/65536K available (4538K kernel code, 305K rwdata, 1704K rodata, 188K init, 252K bss, 7780K reserved, 0K cma-reserved)
[    0.000000] Virtual kernel memory layout:
[    0.000000]     vector  : 0xffff0000 - 0xffff1000   (   4 kB)
[    0.000000]     fixmap  : 0xffc00000 - 0xfff00000   (3072 kB)
[    0.000000]     vmalloc : 0xc4800000 - 0xff800000   ( 944 MB)
[    0.000000]     lowmem  : 0xc0000000 - 0xc4000000   (  64 MB)
[    0.000000]     modules : 0xbf000000 - 0xc0000000   (  16 MB)
[    0.000000]       .text : 0xc0008000 - 0xc0620f54   (6244 kB)
[    0.000000]       .init : 0xc0621000 - 0xc0650000   ( 188 kB)
[    0.000000]       .data : 0xc0650000 - 0xc069c784   ( 306 kB)
[    0.000000]        .bss : 0xc069c784 - 0xc06db8f8   ( 253 kB)
[    0.000000] SLUB: HWalign=32, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
[    0.000000] Preemptible hierarchical RCU implementation.
[    0.000000]   Build-time adjustment of leaf fanout to 32.
[    0.000000] NR_IRQS:545
[    0.000000] clocksource: nuc980-timer5: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 62215505635 ns
[    0.000033] sched_clock: 24 bits at 120kHz, resolution 8333ns, wraps every 69905062489ns
[    0.000741] Console: colour dummy device 80x30
[    0.186616] console [ttyS0] enabled
[    0.190091] Calibrating delay loop (skipped) preset value.. 148.88 BogoMIPS (lpj=744448)
[    0.198174] pid_max: default: 32768 minimum: 301
[    0.203133] Mount-cache hash table entries: 1024 (order: 0, 4096 bytes)
[    0.209708] Mountpoint-cache hash table entries: 1024 (order: 0, 4096 bytes)
[    0.218916] CPU: Testing write buffer coherency: ok
[    0.224983] Setting up static identity map for 0x8400 - 0x843c
[    0.271558] clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 19112604462750000 ns
[    0.282316] futex hash table entries: 256 (order: -1, 3072 bytes)
[    0.288874] pinctrl core: initialized pinctrl subsystem
[    0.296433] NET: Registered protocol family 16
[    0.303199] DMA: preallocated 256 KiB pool for atomic coherent allocations
[    0.316783] <DT> nuc980_dt_device_init +
```

5.7.2 System

5.7.2.1 System

System Properties > General Settings



System

Here you can configure the ba ke its hostname or the timezone.

System Properties

Local Time: 2022/3/21 下午2:58:56

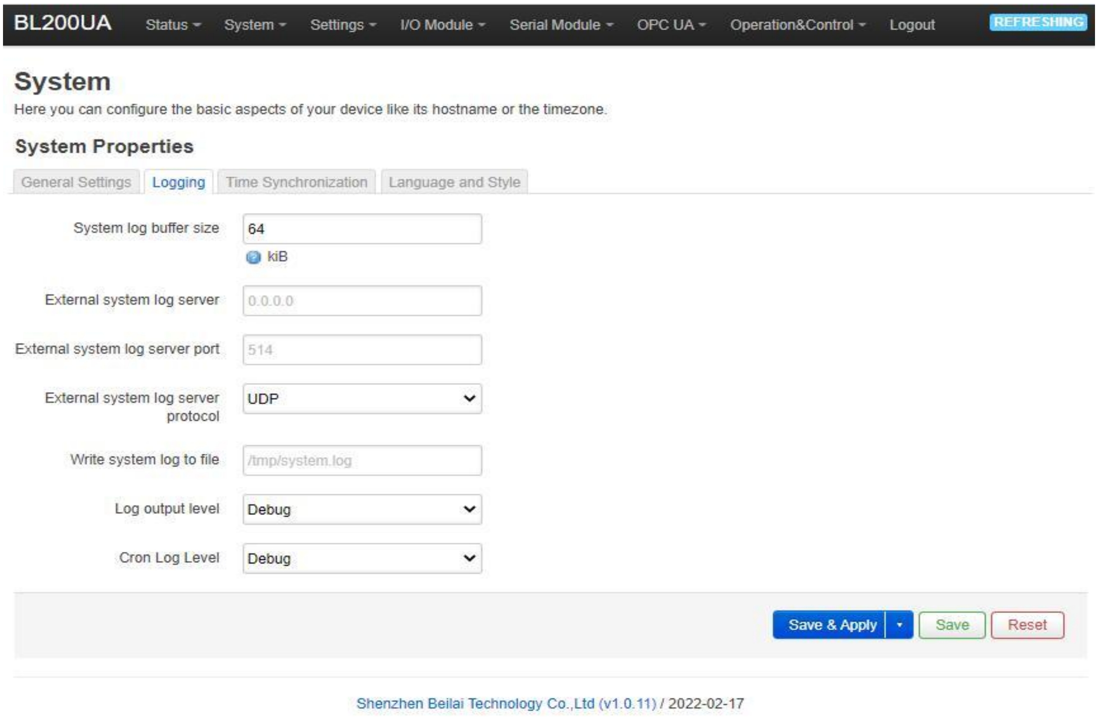
Hostname: BL200UA

Timezone: UTC

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description	Default
Local time	Displays the current time of the device. You can click the "Sync browser time" or "Sync with NTP server" button to update the device time.	--
Hostname	The device name can be customized to easily distinguish between multiple devices.	BL200
Timezone	The time zone can be selected via the drop down menu	UTC

System Properties > Logging



System

Here you can configure the basic aspects of your device like its hostname or the timezone.

System Properties

System log buffer size: 64 kiB

External system log server: 0.0.0.0

External system log server port: 514

External system log server protocol: UDP

Write system log to file: /tmp/system.log

Log output level: Debug

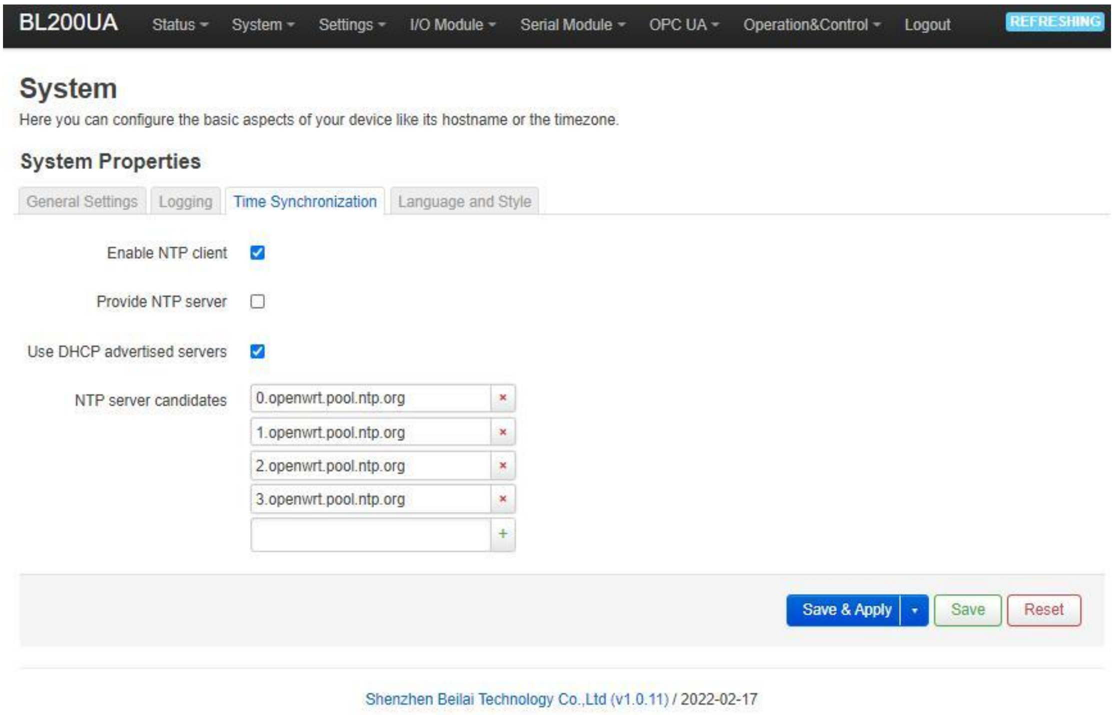
Cron Log Level: Debug

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description	Default
System log buffer size		64
External system log server		
External system log server port		
External system log server protocol		
Write system log to file		
Log output level		
Cron log level		

System Properties > Time Synchronization

An NTP server can be set to synchronize time



System

Here you can configure the basic aspects of your device like its hostname or the timezone.

System Properties

Enable NTP client ☑

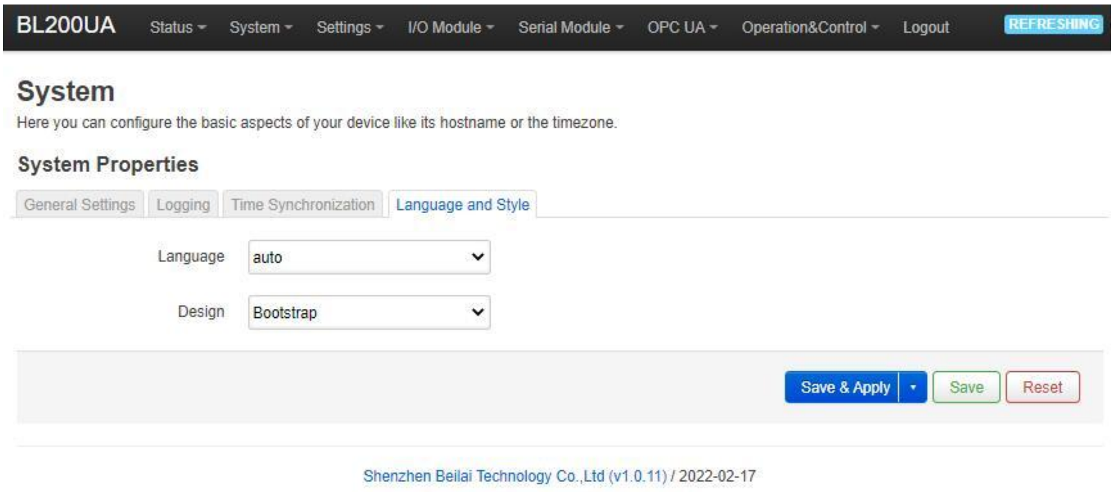
Provide NTP server □

Use DHCP advertised servers ☑

NTP server candidates
0.openwrt.pool.ntp.org
1.openwrt.pool.ntp.org
2.openwrt.pool.ntp.org
3.openwrt.pool.ntp.org

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

System Properties > Language and Style



System

Here you can configure the basic aspects of your device like its hostname or the timezone.

System Properties

Language: auto

Design: Bootstrap

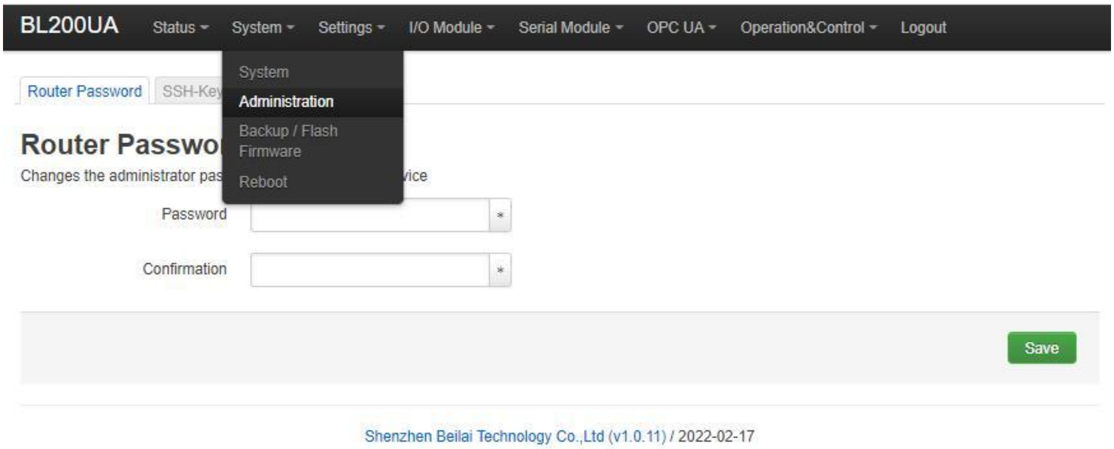
Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description	Default
Language	Available in auto, English, Chinese	auto
Design	Currently only Bootstrap is supported.	Bootstrap

5.7.2.2 Administration

Administration > Router Password

Change the administrator password for accessing the device.



Router Passwo

Changes the administrator pas…vice

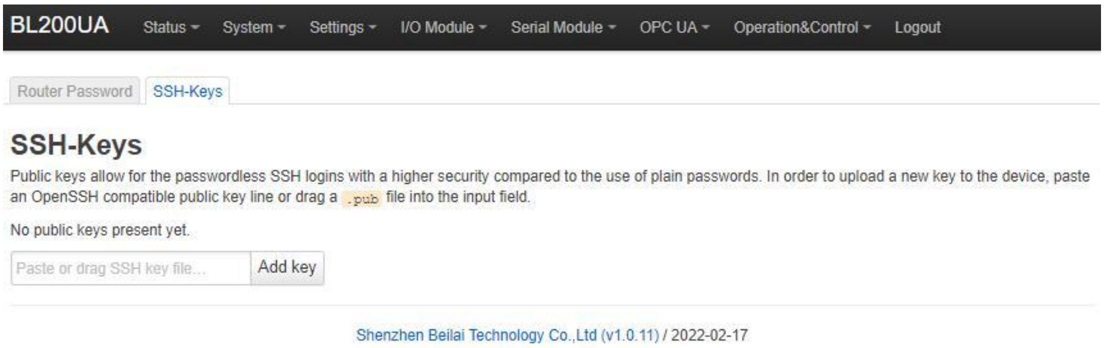
Password

Confirmation

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Administration > SSH Keys

Public keys allow for the passwordless SSH logins with a higher security compared to the use of regular passwords. In order to upload a new key to the device, paste an OpenSSH compatible public key line or drag a .pub file into the input field.



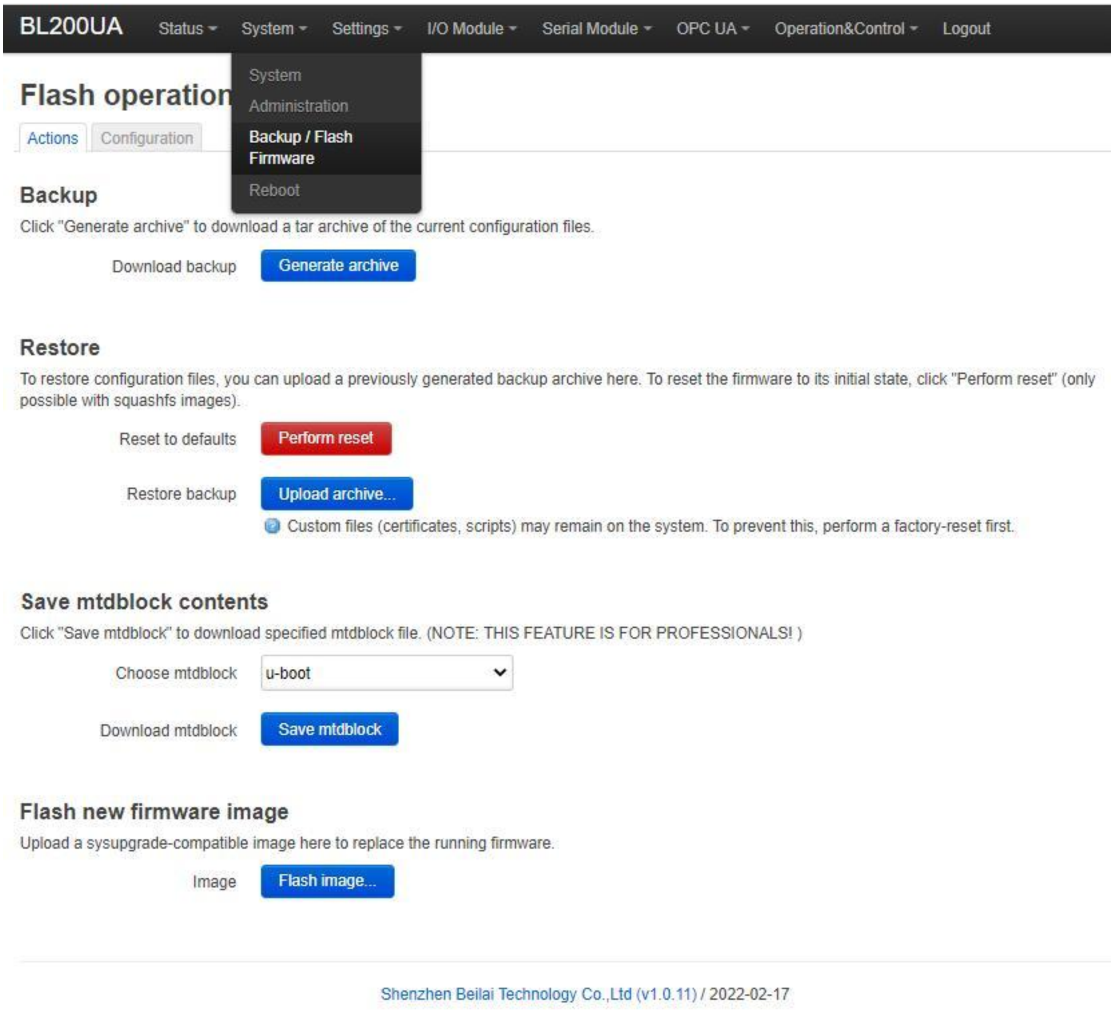
SSH-Keys

Public keys allow for the passwordless SSH logins with a higher security compared to the use of plain passwords. In order to upload a new key to the device, paste an OpenSSH compatible public key line or drag a .pub file into the input field.

No public keys present yet.

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

5.7.2.3 Backup/Flash Firmware



Flash operation

Backup

Click "Generate archive" to download a tar archive of the current configuration files.

Download backup

Restore

To restore configuration files, you can upload a previously generated backup archive here. To reset the firmware to its initial state, click "Perform reset" (only possible with squashfs images).

Reset to defaults

Restore backup

Custom files (certificates, scripts) may remain on the system. To prevent this, perform a factory-reset first.

Save mtdblock contents

Click "Save mtdblock" to download specified mtdblock file. (NOTE: THIS FEATURE IS FOR PROFESSIONALS!)

Choose mtdblock: u-boot

Download mtdblock

Flash new firmware image

Upload a sysupgrade-compatible image here to replace the running firmware.

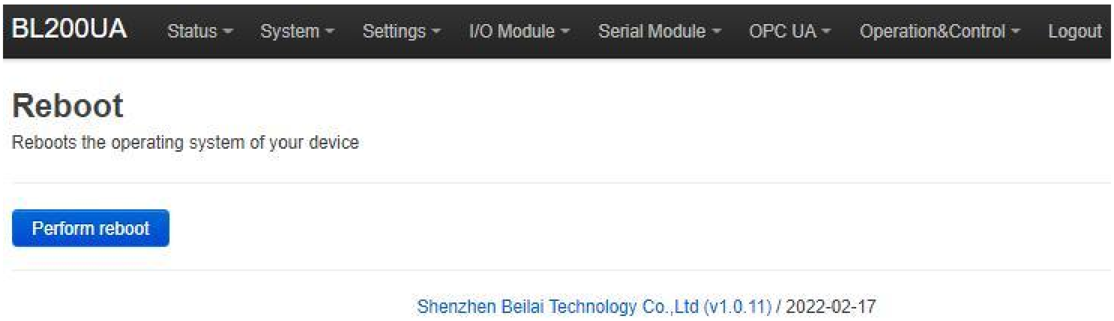
Image

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description	Default
Backup	Click "Generate archive" to download a tar archive of the current configuration files.	--
Restore	To restore configuration files, you can upload a previously generated backup archive here. To reset the firmware to its initial state, click "Perform reset" (only possible with squashfs images).	--
Save mtdblcok	Click "Save mtdblock" to download specified mtdblock file. (NOTE: THIS FEATURE IS FOR PROFESSIONALS)	--
Flash image	Upload a sysupgrade-compatible image here to replace the running firmware.	--

5.7.2.4 Reboot

Click "Perform reboot" will reboot your device

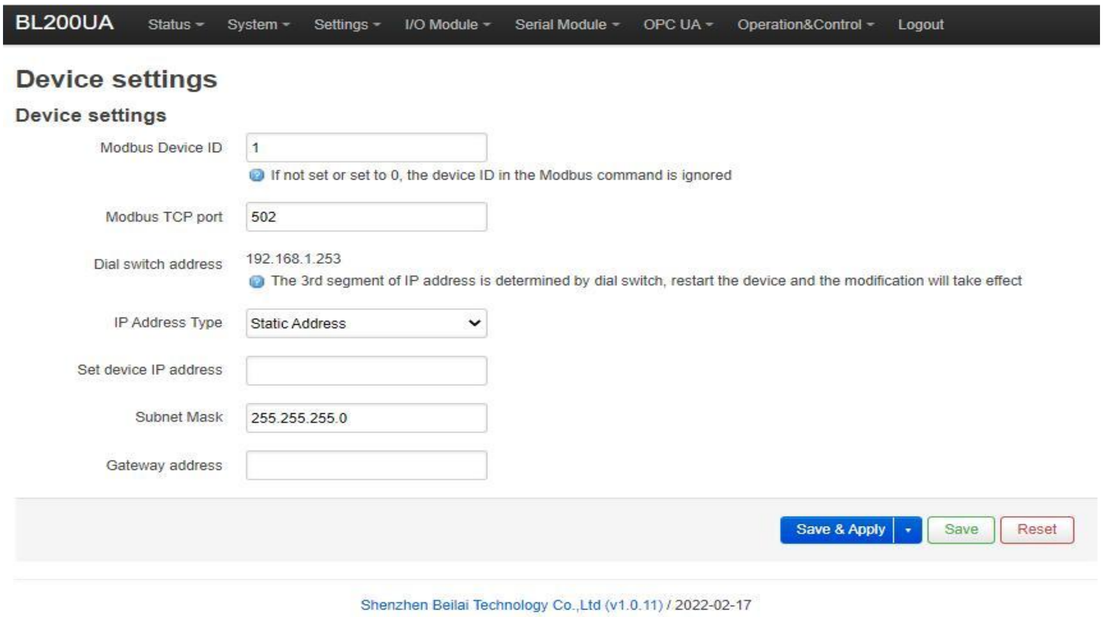


Reboot

Reboots the operating system of your device

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

5.7.3 Settings



Device settings

Device settings

Modbus Device ID: 1

If not set or set to 0, the device ID in the Modbus command is ignored

Modbus TCP port: 502

Dial switch address: 192.168.1.253

The 3rd segment of IP address is determined by dial switch, restart the device and the modification will take effect

IP Address Type: Static Address

Set device IP address:

Subnet Mask: 255.255.255.0

Gateway address:

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description	Default
Modbus Device ID	Modbus device ID range is 1~247.	1
Modbus TCP port	Modbus TCP protocol port number, which can be customized.	502
DIP switch address	Displays the IP address set by the DIP switch.	
IP address type	Select from "Static Address", "Dynamic Address(DHCP)".	
Set device IP address	The IP address of the device can be set by yourself, and it needs to be restarted to take effect after setting.	--

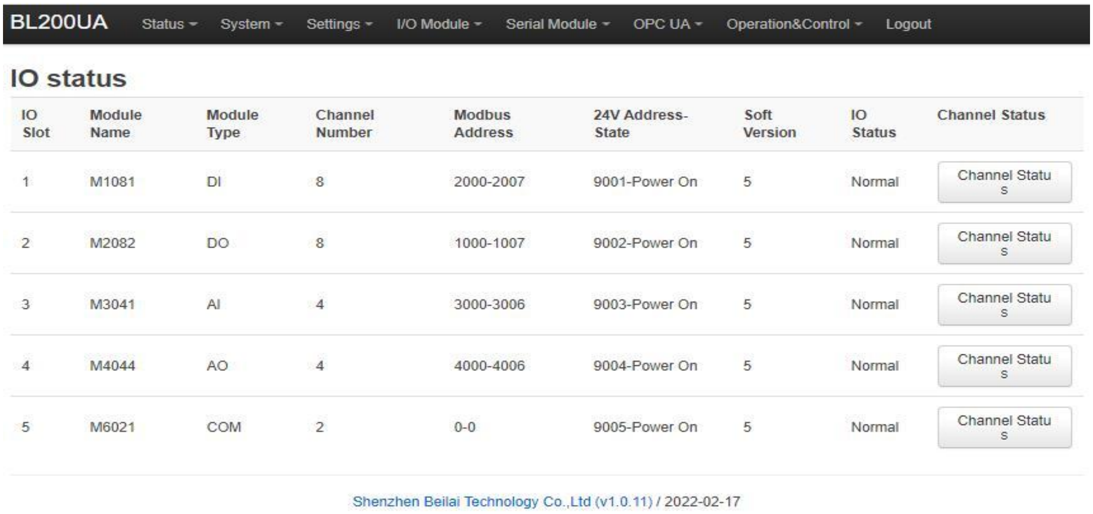
Subnet mask	Set IP subnet mask	
Gateway address	Set IP gateway address	

5.7.4 I/O Modules

After power on, the controller automatically recognizes all I/O modules connected to it and creates an internal local process image based on the module type, data width and the module's position in the node.

If I/O modules are added, changed or removed, a new process image is created and the process data addresses change. When adding an I/O module, the process data of all previous I/O modules must be considered.

The controller can connect up to 32 I/O modules, including digital input and output, analog input and output and special function modules.



IO status

IO Slot	Module Name	Module Type	Channel Number	Modbus Address	24V Address-State	Soft Version	IO Status	Channel Status
1	M1081	DI	8	2000-2007	9001-Power On	5	Normal	Channel Status
2	M2082	DO	8	1000-1007	9002-Power On	5	Normal	Channel Status
3	M3041	AI	4	3000-3006	9003-Power On	5	Normal	Channel Status
4	M4044	AO	4	4000-4006	9004-Power On	5	Normal	Channel Status
5	M6021	COM	2	0-0	9005-Power On	5	Normal	Channel Status

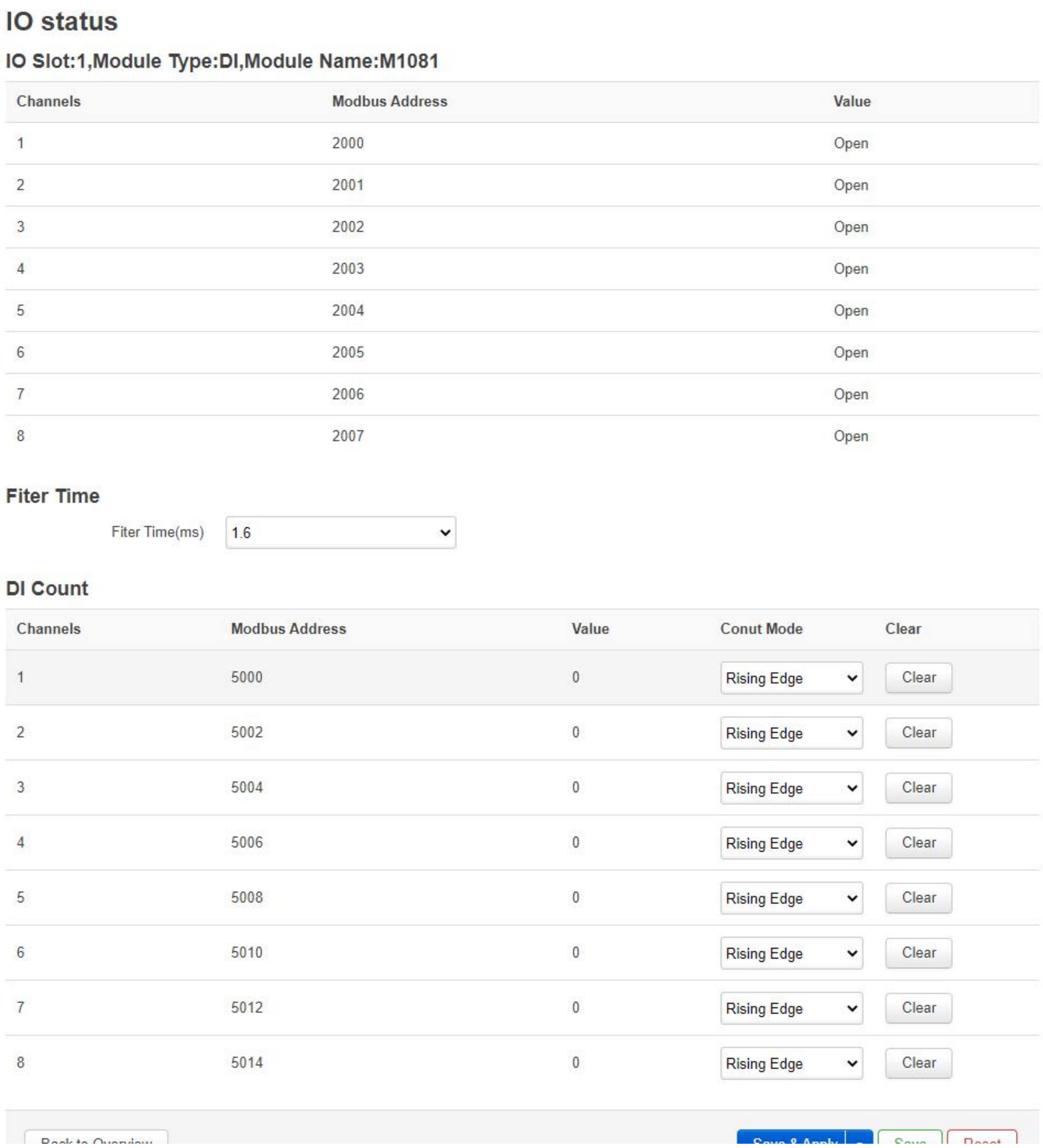
Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description
IO slot	The order of IO modules in the slot, the first module card position close to the controller is 1, and the following ones are 2 3 4...
Module name	I/O module model
Module type	I/O module function type
Channel Number	Data width of I/O module
Modbus Address	Process map address of the I/O module inside the controller
24V Address State	Power supply status on the field side of the I/O module, digital, 1 bit
Software version	I/O module internal firmware version

IO status	I/O module and controller communication status
Channel status	Click to view and set the parameters of different types of I/O modules

5.7.4.1 Digital Input Module

The digital input module can provide two types of data, one is the current input state value, Boolean type; the other is the counter value, 32-bit numerical type, which supports the clear function.



IO status

IO Slot:1,Module Type:DI,Module Name:M1081

Channels	Modbus Address	Value
1	2000	Open
2	2001	Open
3	2002	Open
4	2003	Open
5	2004	Open
6	2005	Open
7	2006	Open
8	2007	Open

Fiter Time

Fiter Time(ms): 1.6

DI Count

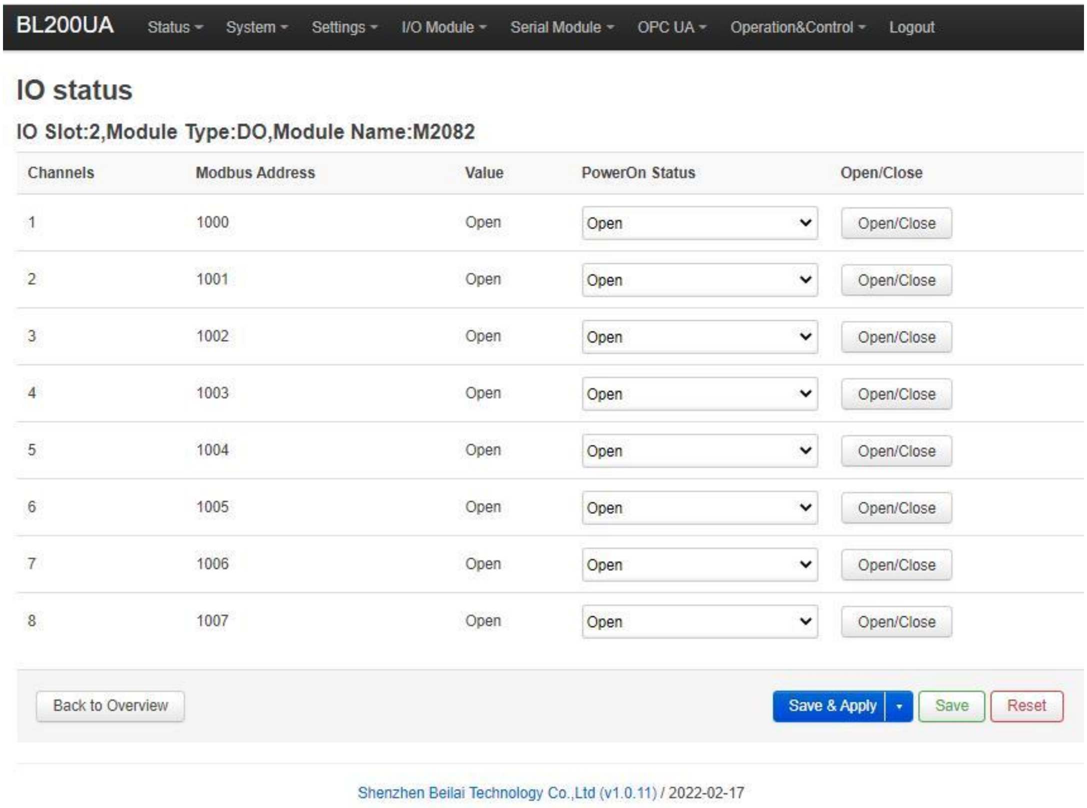
Channels	Modbus Address	Value	Conut Mode	Clear
1	5000	0	Rising Edge	Clear
2	5002	0	Rising Edge	Clear
3	5004	0	Rising Edge	Clear
4	5006	0	Rising Edge	Clear
5	5008	0	Rising Edge	Clear
6	5010	0	Rising Edge	Clear
7	5012	0	Rising Edge	Clear
8	5014	0	Rising Edge	Clear

Item	Description
Channels	Channel number of the digital input module

Modbus Address	Process map address of Boolean status data inside the coupler
Value	Display the current input state, open: logic 0, close: logic 1
Fliter Time	Selecting the time for DI filtering

Item	Description
Channels	Channel number of the digital input module
Modbus Address	Process map address of the count value inside the coupler
Value	Display the current input count value, 32-bit unsigned integer
Count Mode	Selection of "Rising Edge", "Falling Edge", "Rising Edge and Falling Edge" Trigger Counting Methods
Clear	Clear the current channel counter value

5.7.4.2 Digital Output Module



IO status

IO Slot:2,Module Type:DO,Module Name:M2082

Channels	Modbus Address	Value	PowerOn Status	Open/Close
1	1000	Open	Open	Open/Close
2	1001	Open	Open	Open/Close
3	1002	Open	Open	Open/Close
4	1003	Open	Open	Open/Close
5	1004	Open	Open	Open/Close
6	1005	Open	Open	Open/Close
7	1006	Open	Open	Open/Close
8	1007	Open	Open	Open/Close

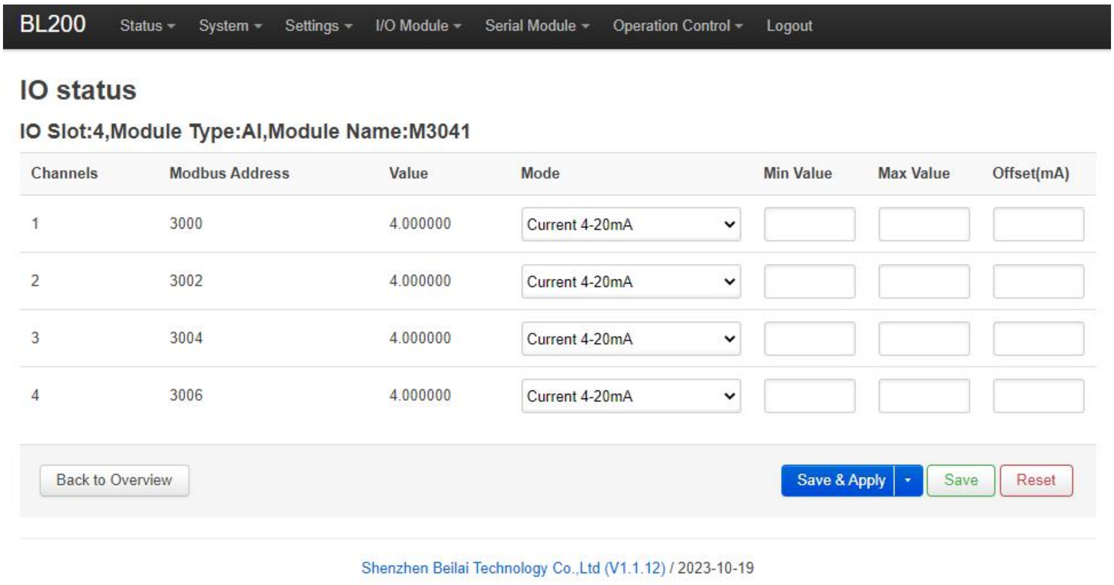
Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Item	Description
Channels	Channel number of the digital output module
Modbus Address	Process map address of the digital output boolean data inside the controller
Value	Display the current output state, open: 0, close: 1

Power-on status	Set the state of DO after power-on, select from "open", "close", "last"
Open/Close	Can control the current channel output state

5.7.4.3 Analog Input Module

The analog input (AI) type module supports setting parameters through the controller web page, so that the data conversion is automatically realized inside the module, and the actual engineering value corresponding to the sensor can be directly output.



IO status

IO Slot:4,Module Type:AI,Module Name:M3041

Channels	Modbus Address	Value	Mode	Min Value	Max Value	Offset(mA)
1	3000	4.000000	Current 4-20mA			
2	3002	4.000000	Current 4-20mA			
3	3004	4.000000	Current 4-20mA			
4	3006	4.000000	Current 4-20mA			

Shenzhen Beilai Technology Co.,Ltd (V1.1.12) / 2023-10-19

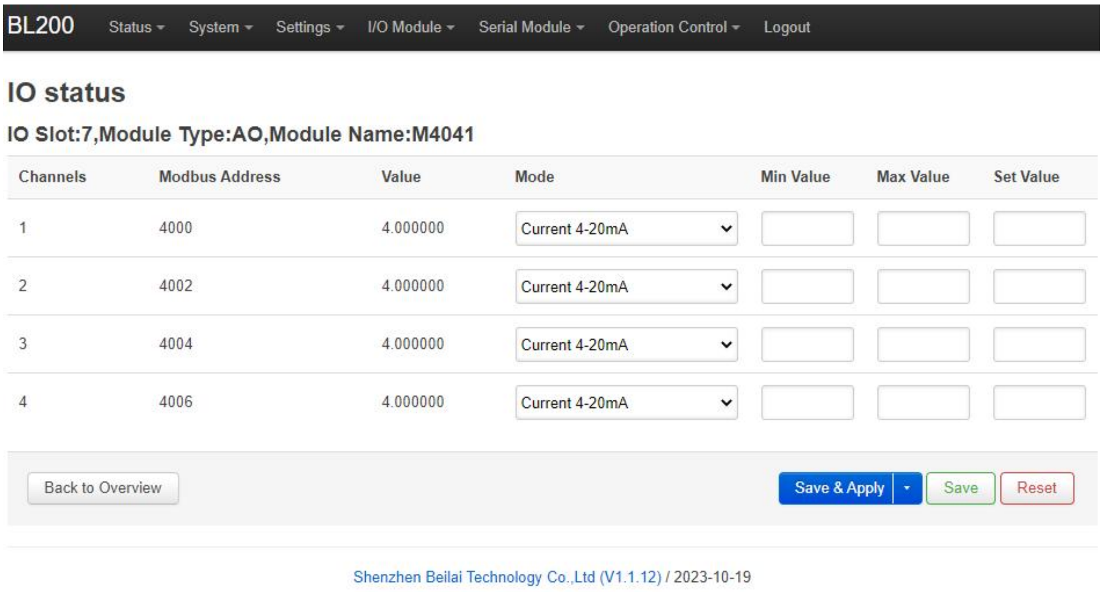
Item	Description
Channels	Channel number of the analog input module
Modbus Address	Process map address of the analog input module inside the controller
Value	Display the actual engineering value input by the current channel, 32-bit single-precision floating-point type
Mode	Different models of analog input modules have different options, please refer to the specific analog input I/O module manual for details.
Min Value	Sensor range minimum
Max Value	Sensor range maximum
Offset(mA)	The offset allows you to adjust the error between acquisition and actual.

There is a linear relationship between the electrical signal value of the analog input

module (usually a sensor) and the actual engineering value. Their formulas are as follows (take 4-20mA as an example):
Actual engineering value = (current value - 4) * ((maximum - minimum) / (20 - 4)) + minimum

Take the 4-20mA type water level sensor to measure the depth of the water tower as an example:
The known water level sensor range is 0-100m, the current data is 5.6mA, and the depth of the water tower is calculated:
Into the formula:
(5.6 - 4) * ((100 - 0) / (20 - 4)) + 0 = 10
The depth of the water tower is 10m

5.7.4.4 Analog Output Module



IO status

IO Slot:7,Module Type:AO,Module Name:M4041

Channels	Modbus Address	Value	Mode	Min Value	Max Value	Set Value
1	4000	4.000000	Current 4-20mA			
2	4002	4.000000	Current 4-20mA			
3	4004	4.000000	Current 4-20mA			
4	4006	4.000000	Current 4-20mA			

Shenzhen Beilai Technology Co.,Ltd (V1.1.12) / 2023-10-19

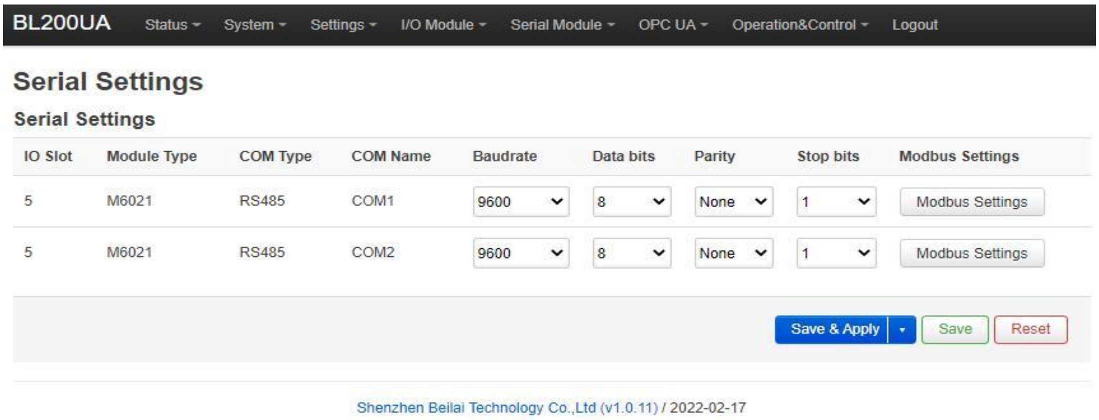
Item	Description
Channels	Channel number of the analog output module
Modbus Address	Process map address of the analog output module inside the controller
Value	Display the actual engineering value output by the current channel, 32-bit single-precision floating-point type
Mode	Different models of analog output modules have different options, please refer to the specific analog output I/O module manual for details.
Min value	Actual engineering value minimum value

Max value	Actual engineering value maximum value
Set value	You can set the actual project value required for the output

5.7.5 Serial Port Module

Various sensors, meters and other devices that support Modbus RTU(Master) protocol can be connected to the edge controller through the serial port module. It allows process mapping between external sensor data and the coupler via the local bus.

5.7.5.1 Serial Port Settings



Serial Settings

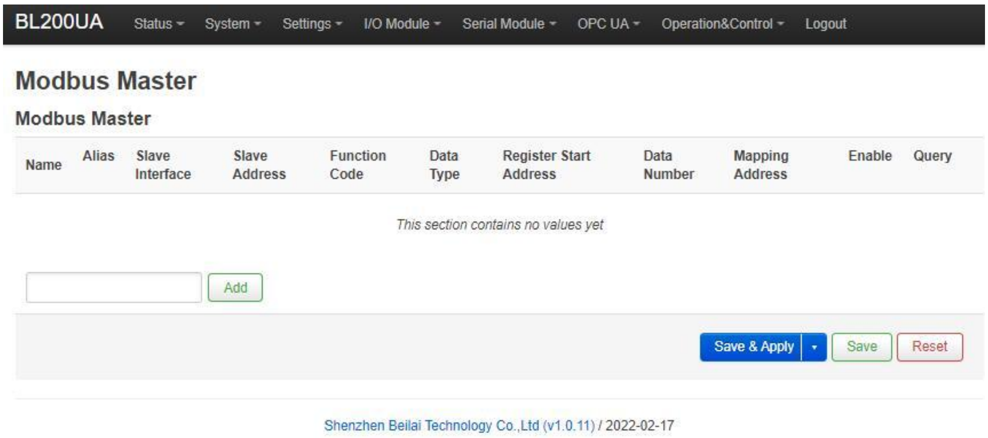
Serial Settings

IO Slot	Module Type	COM Type	COM Name	Baudrate	Data bits	Parity	Stop bits	Modbus Settings
5	M6021	RS485	COM1	9600	8	None	1	Modbus Settings
5	M6021	RS485	COM2	9600	8	None	1	Modbus Settings

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

5.7.5.2 Modbus Settings

Modbus settings are used to add Modbus RTU devices to the serial communication I/O module. A maximum of 25 Modbus commands can be created.



Modbus Master

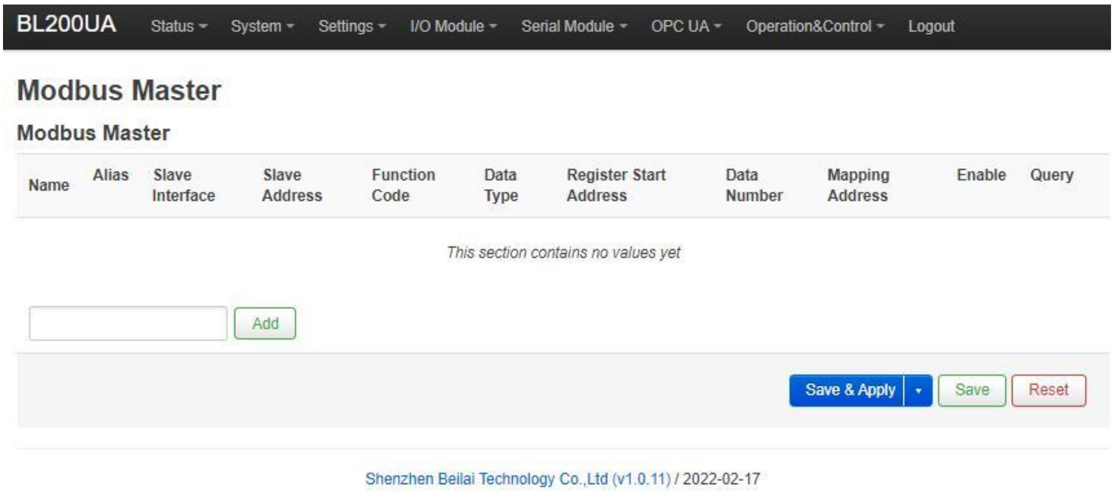
Modbus Master

Name	Alias	Slave Interface	Slave Address	Function Code	Data Type	Register Start Address	Data Number	Mapping Address	Enable	Query

This section contains no values yet

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

Enter the custom data name in the input box and click Add



Modbus Master

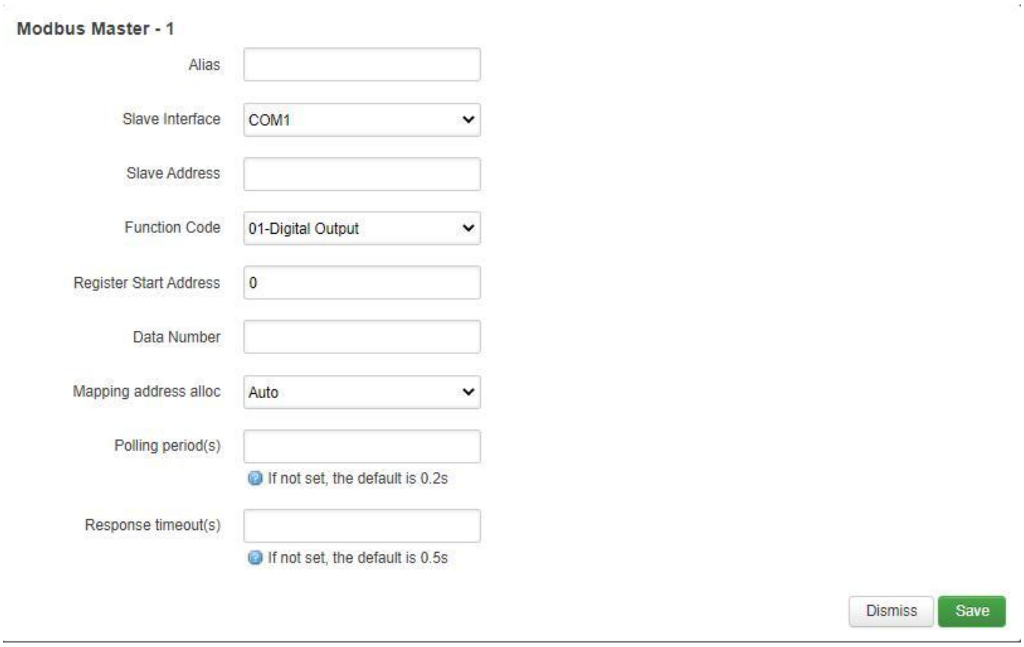
Modbus Master

Name	Alias	Slave Interface	Slave Address	Function Code	Data Type	Register Start Address	Data Number	Mapping Address	Enable	Query

This section contains no values yet

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

The configuration box pops



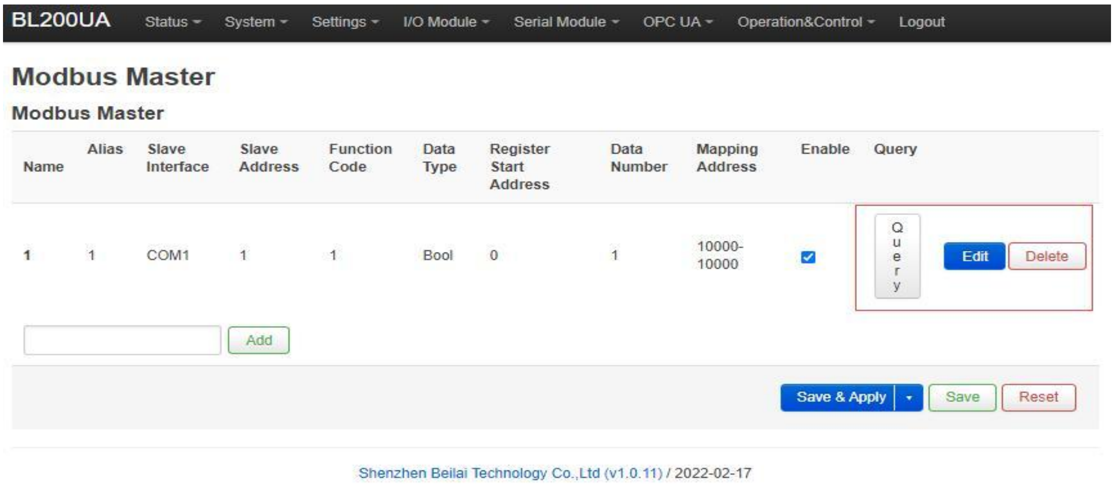
Modbus Master - 1

Alias	
Slave Interface	COM1
Slave Address	
Function Code	01-Digital Output
Register Start Address	0
Data Number	
Mapping address alloc	Auto
Polling period(s)	 If not set, the default is 0.2s
Response timeout(s)	 If not set, the default is 0.5s

Item	Description
Alias	Device nickname can be used to distinguish data
Slave Interface	Select serial channel
Slave address	Slave device address, range 1-247
Function code	Select according to the slave data type, including: "01", "02", "03", "04"
Register start address	Register start address of slave data
Data number	Number of slave data

Mapping address alloc	Support distribution method: auto According to different data types, the system automatically allocates down the starting address of the mapping, and the addresses are continuous. manual Manual allocation allows mapping addresses to be allocated across segments
Polling period (s)	The interval between two adjacent polling commands
Response timeout (s)	After sending the command to the slave, wait for the maximum time for the slave to return data. If the time exceeds this time, the slave will be considered to have no response.

You can modify, delete, and view data of slave, or you can disable collection.



Modbus Master

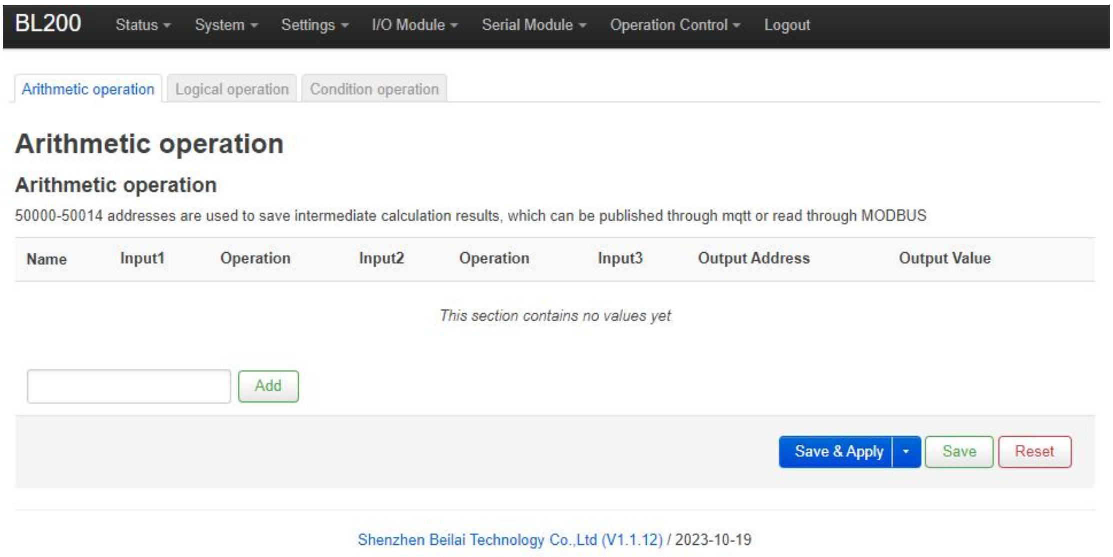
Modbus Master

Name	Alias	Slave Interface	Slave Address	Function Code	Data Type	Register Start Address	Data Number	Mapping Address	Enable	Query
1	1	COM1	1	1	Bool	0	1	10000-10000	☑	Query Edit Delete

Shenzhen Beilai Technology Co.,Ltd (v1.0.11) / 2022-02-17

5.7.6 Operation and Control

5.7.6.1 Arithmetic Operation



Arithmetic operation

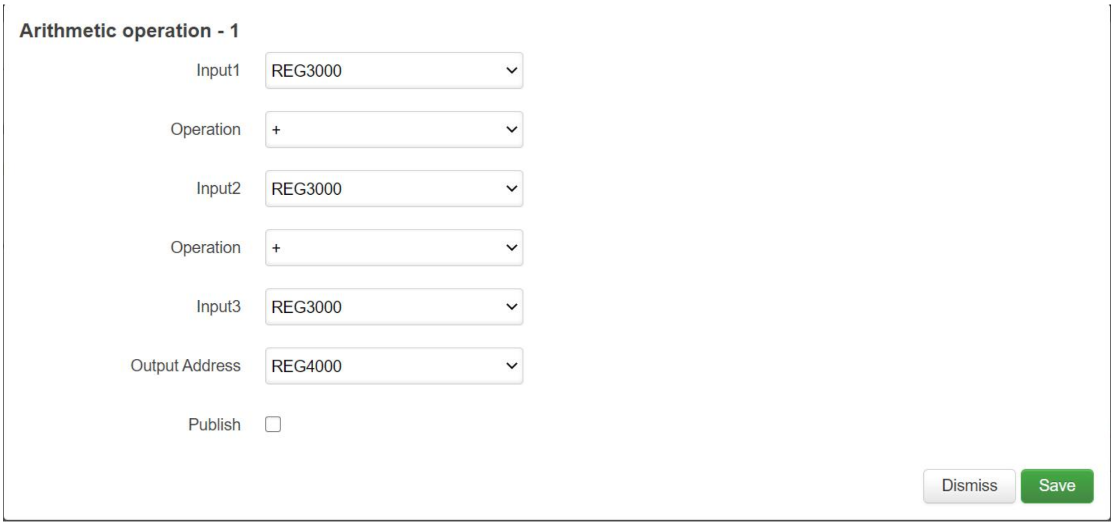
Arithmetic operation

50000-50014 addresses are used to save intermediate calculation results, which can be published through mqtt or read through MODBUS

Name	Input1	Operation	Input2	Operation	Input3	Output Address	Output Value

This section contains no values yet

Shenzhen Beilai Technology Co.,Ltd (V1.1.12) / 2023-10-19

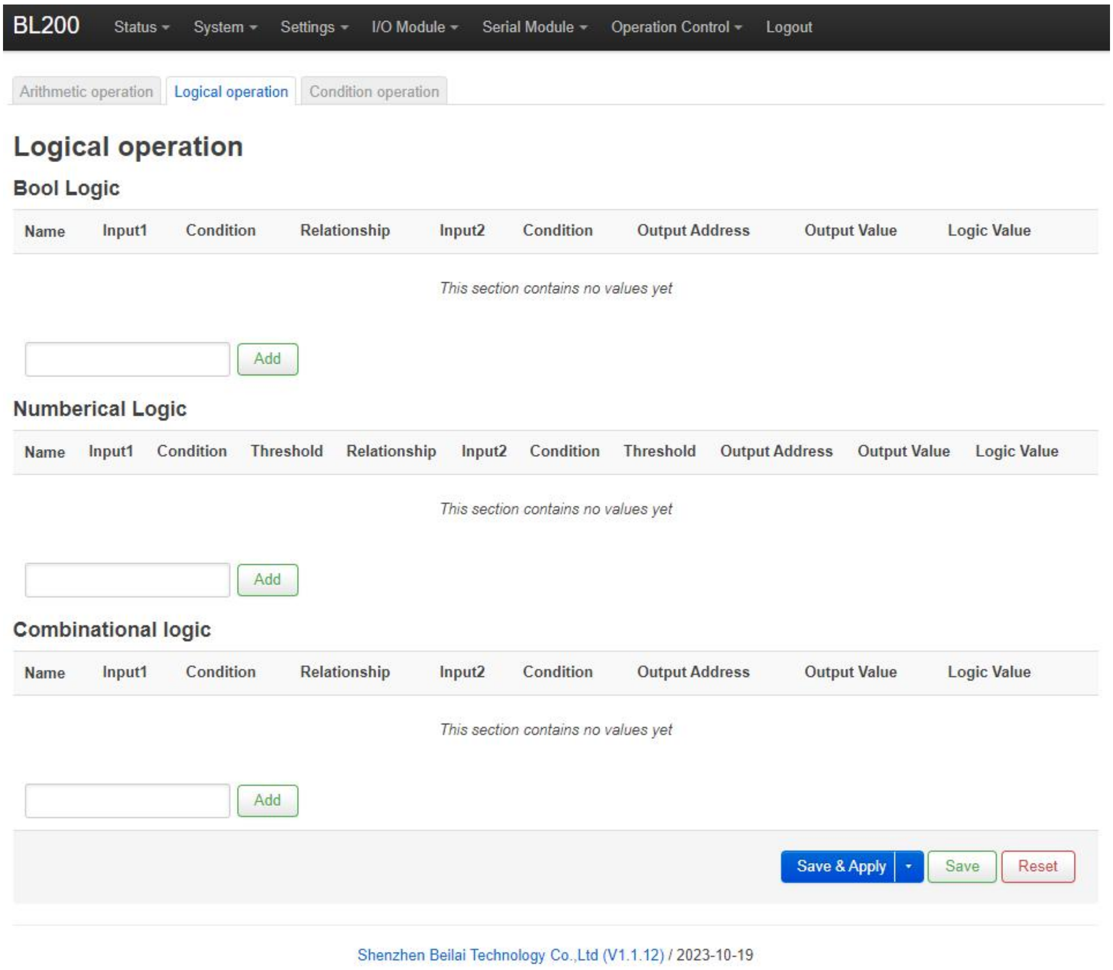


Arithmetic operation - 1

Input1	REG3000
Operation	+
Input2	REG3000
Operation	+
Input3	REG3000
Output Address	REG4000
Publish	□

It supports "addition, subtraction, multiplication, and division" operations between AI, AO, or RS485 slave numerical data, and can also perform operations with "addition, subtraction, multiplication, and division" constants, and freely match 1 or 2 conditions to combine the output results. If a 16-bit register address is used as the output result, the output with a decimal is an integer.

5.7.6.2 Logical Operation



Logical operation

Bool Logic

Name	Input1	Condition	Relationship	Input2	Condition	Output Address	Output Value	Logic Value

This section contains no values yet

Numberical Logic

Name	Input1	Condition	Threshold	Relationship	Input2	Condition	Threshold	Output Address	Output Value	Logic Value

This section contains no values yet

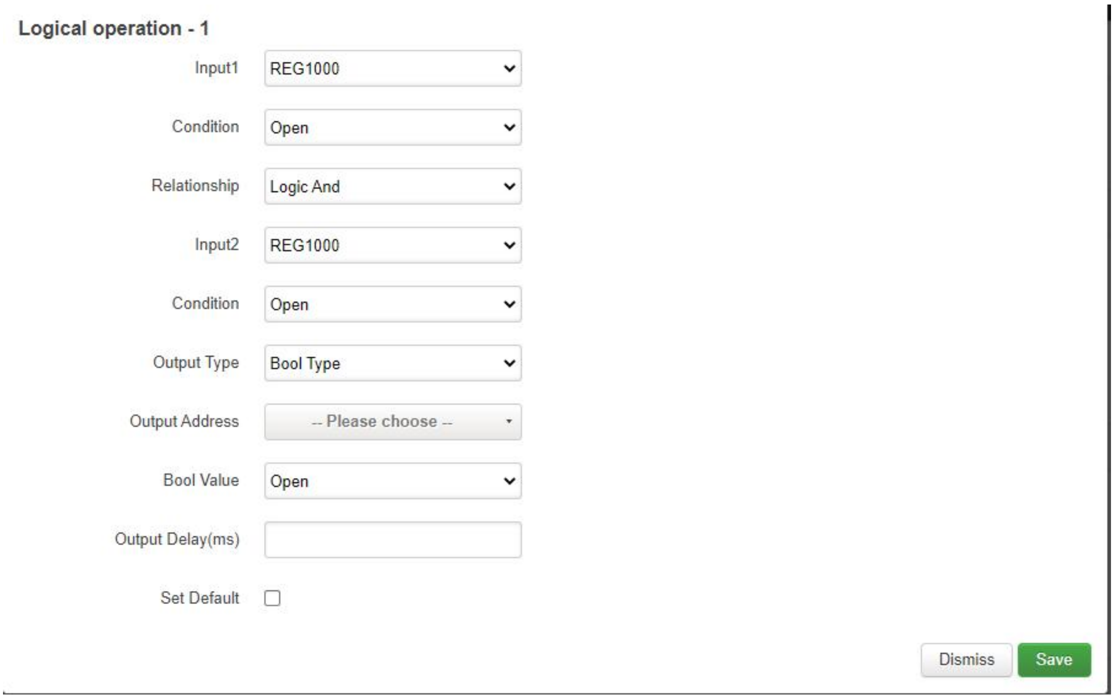
Combinational logic

Name	Input1	Condition	Relationship	Input2	Condition	Output Address	Output Value	Logic Value

This section contains no values yet

Shenzhen Beilai Technology Co.,Ltd (V1.1.12) / 2023-10-19

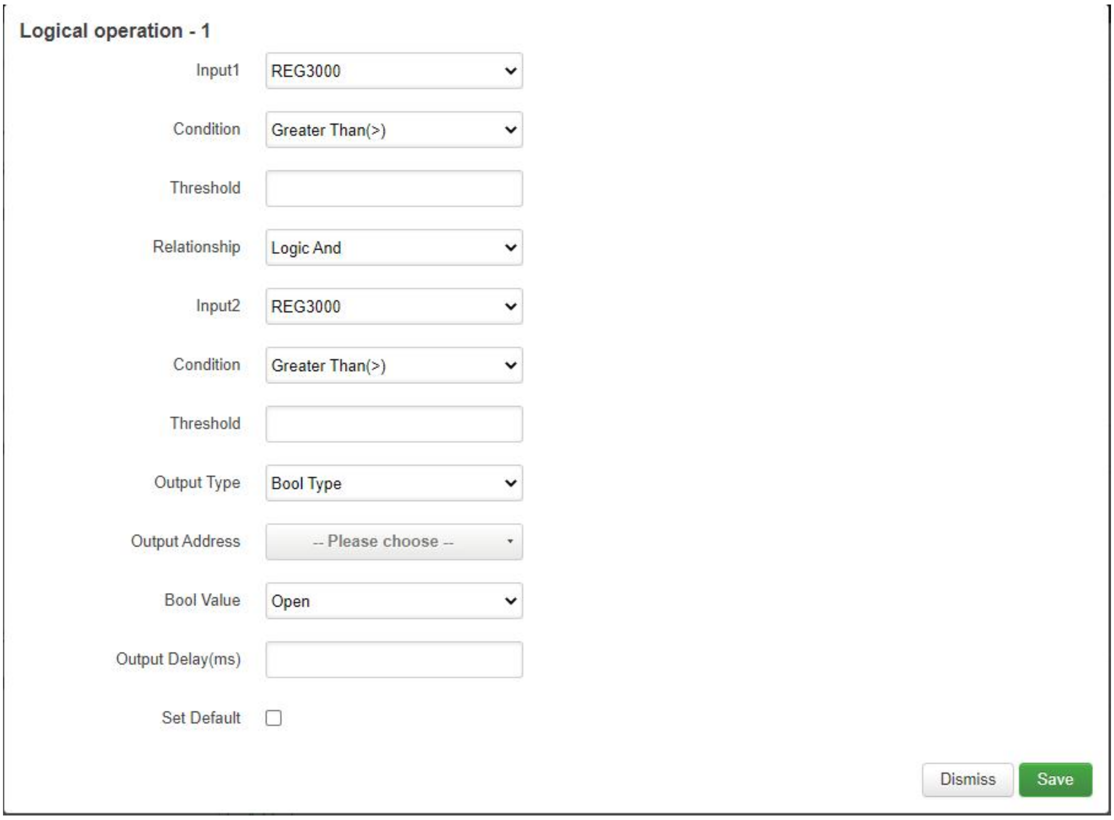
Bool logic configuration



Logical operation - 1

Input1	REG1000
Condition	Open
Relationship	Logic And
Input2	REG1000
Condition	Open
Output Type	Bool Type
Output Address	-- Please choose --
Bool Value	Open
Output Delay(ms)	
Set Default	□

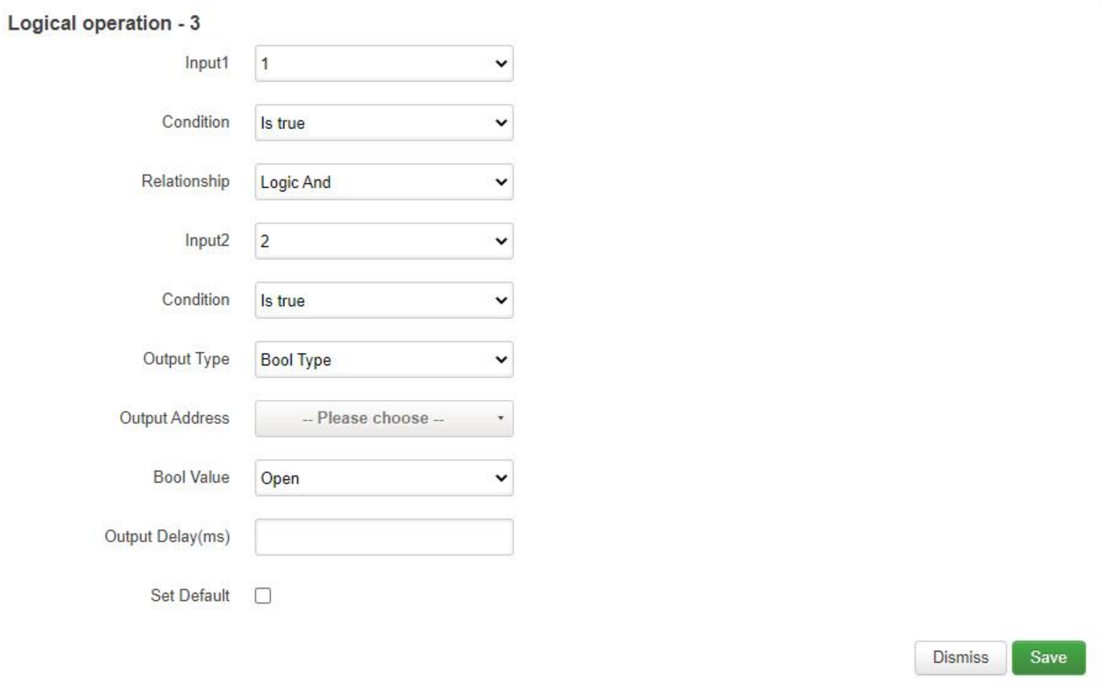
Numerical Logic Configuration



Logical operation - 1

Input1	REG3000
Condition	Greater Than(>)
Threshold	
Relationship	Logic And
Input2	REG3000
Condition	Greater Than(>)
Threshold	
Output Type	Bool Type
Output Address	-- Please choose --
Bool Value	Open
Output Delay(ms)	
Set Default	□

Combinational logic configuration



Logical operation - 3

Input1	1
Condition	Is true
Relationship	Logic And
Input2	2
Condition	Is true
Output Type	Bool Type
Output Address	-- Please choose --
Bool Value	Open
Output Delay(ms)	
Set Default	□

Users can freely set various combination linkages between I/O (digital input and output, analog input and output) or serial port modules (Modbus slave data) according to needs. Whether the built logic is triggered can be judged according to the logic value item of the web page, "0" means not triggered, and "1" means triggered. Logical value items cannot be updated automatically, and the web page must be manually refreshed.

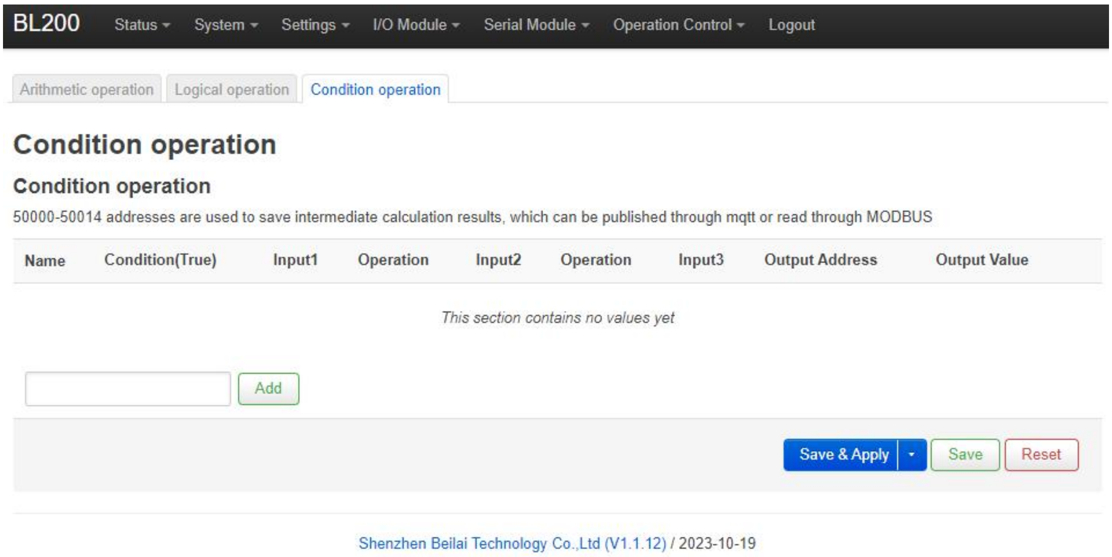
Example:

Logic 1 (And), input condition A and input condition B meet the trigger condition at the same time, output result Y.

Logic 2 (Or), any one of input condition C or input condition D satisfies the trigger condition, and the output result is Y.

Logic 3: Logic 1 + Logic 2 can be combined to form a logic 3 or more combinations.

5.7.6.3 Condition Operation



Condition operation

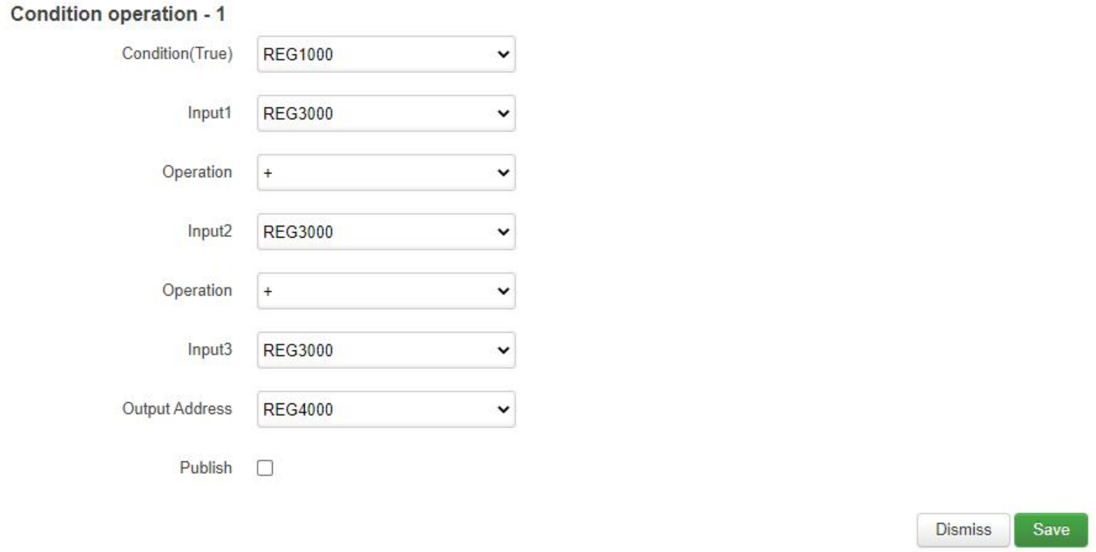
Condition operation

50000-50014 addresses are used to save intermediate calculation results, which can be published through mqtt or read through MODBUS

Name	Condition(True)	Input1	Operation	Input2	Operation	Input3	Output Address	Output Value

This section contains no values yet

Shenzhen Beilai Technology Co.,Ltd (V1.1.12) / 2023-10-19



Condition operation - 1

Condition(True)	REG1000
Input1	REG3000
Operation	+
Input2	REG3000
Operation	+
Input3	REG3000
Output Address	REG4000
Publish	□

Conditional operation is based on arithmetic operation plus condition triggering, that is, when the condition is satisfied, AI, AO or RS485 slave numerical type data or constants, these data can be free to choose 1-3 conditions for each other to "add, subtract, multiply or divide" arithmetic operation.

5.7.6.4 Example

✧ Take a simple packing system as an example

Requirements:

(1) After pressing the start button, the conveyor belt B starts to run first, and drags the empty box forward to the designated position. After reaching the designated position,

SQ2 sends a signal to stop the conveyor belt B from running.

(2) After the conveyor belt B stops, the conveyor belt A starts to run, and the products fall into the boxes one by one. The SQ1 sensor detects the products and detects that the products fall into the box. Conveyor belt A stops running, conveyor belt B starts running, and it goes on and on, until the stop button is pressed, and conveyor belts A and B stop at the same time.

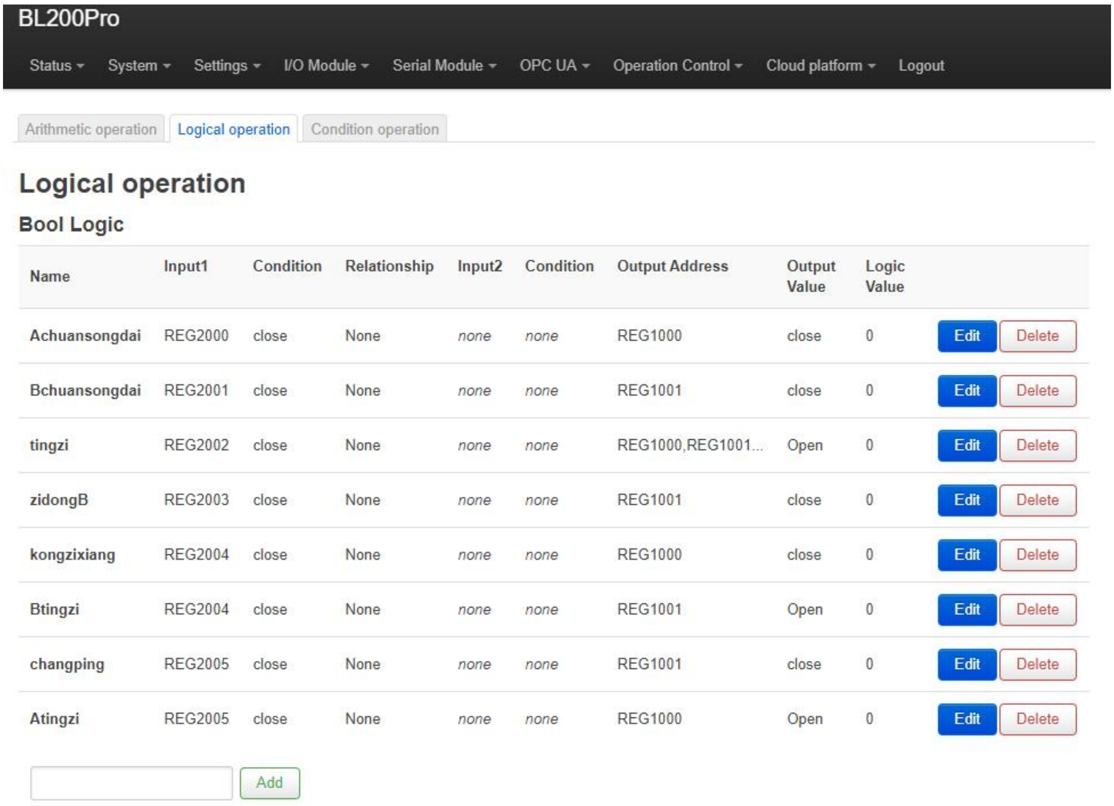
To realize such a function in S7-200SMART, the peripheral wiring needs to use DI and DQ as follows:

Input		Output	
I0.0	Automatic control button	Q0.1	Conveyor A output
I0.1	Stop button	Q0.2	Conveyor B output
I0.2	B conveyor belt moving		
I0.3	A conveyor belt moving		
I0.4	SQ2 input		
I0.5	SQ1 input		

Using BL200 calculation and control simulation to achieve such requirements, the DI and DO required for wiring are as follows:

Input		Output	
DI1	A conveyor belt moving	DO1	Conveyor A output
DI2	B conveyor belt moving	DO2	Conveyor B output
DI3	Stop button		
DI4	Automatic control button		
DI5	Detect empty box sensor, SQ2 input		
DI6	Detect product SQ1 input		

5.7.6.4.1 Bool Logic Configuration Example

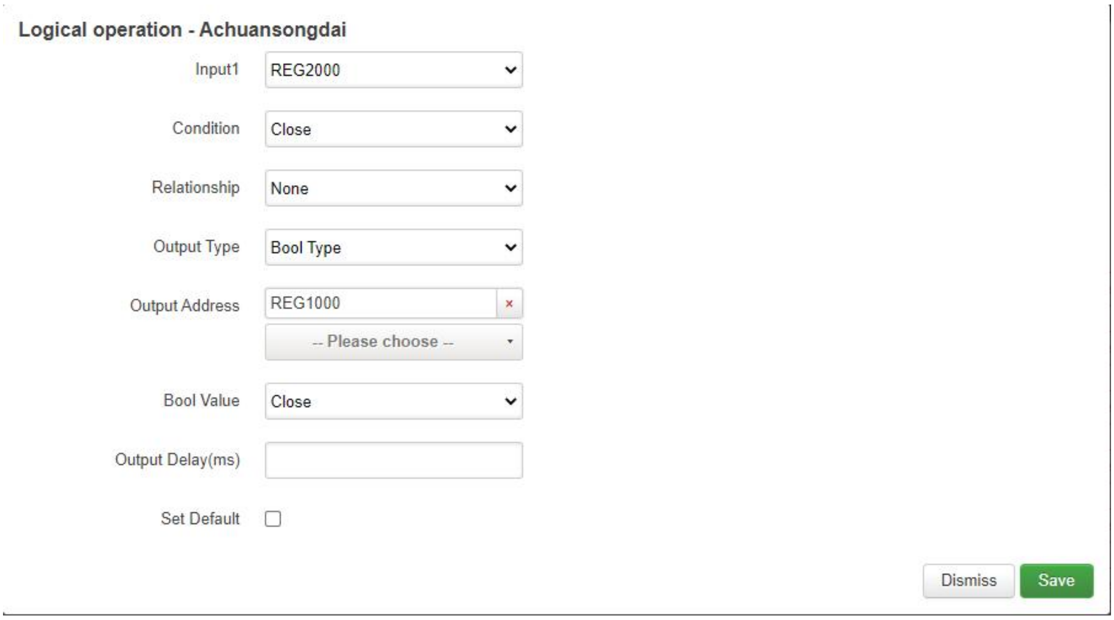


BL200Pro
Status ▾ System ▾ Settings ▾ I/O Module ▾ Serial Module ▾ OPC UA ▾ Operation Control ▾ Cloud platform ▾ Logout
Arithmetic operation | Logical operation | Condition operation

Logical operation

Bool Logic

Name	Input1	Condition	Relationship	Input2	Condition	Output Address	Output Value	Logic Value	
Achuansongdai	REG2000	close	None	*none*	*none*	REG1000	close	0	Edit Delete
Bchuansongdai	REG2001	close	None	*none*	*none*	REG1001	close	0	Edit Delete
tingzi	REG2002	close	None	*none*	*none*	REG1000,REG1001...	Open	0	Edit Delete
zidongB	REG2003	close	None	*none*	*none*	REG1001	close	0	Edit Delete
kongzixiang	REG2004	close	None	*none*	*none*	REG1000	close	0	Edit Delete
Btingzi	REG2004	close	None	*none*	*none*	REG1001	Open	0	Edit Delete
changping	REG2005	close	None	*none*	*none*	REG1001	close	0	Edit Delete
Atingzi	REG2005	close	None	*none*	*none*	REG1000	Open	0	Edit Delete



Logical operation - Achuansongdai

Input1: REG2000

Condition: Close

Relationship: None

Output Type: Bool Type

Output Address: REG1000 ×
-- Please choose --

Bool Value: Close

Output Delay(ms):

Set Default ☐

Steps:

(1) Enter Achuansongdai, click Add, and the configuration box will pop up.

(2) Enter 1: Select DI1 register REG2000.

(3) Condition: Select Close.

(4) Relationship: Select "None", because DI1 directly controls the operation of A conveyor belt, so select "None" because there are no other conditions.

(5) Output type: Select Bool type, because DO1 control is Bool.

(6) Output address: REG1000, DI1 only controls one DO1, so only select the DO1 register address, if DI controls multiple registers, you can select multiple registers. As in the third logic "tingzi", press the stop button, both conveyor belts A and B stop.

(7) Bool value: Off, DI1 controls DO1 to close, so choose to close.

(8) Output delay (milliseconds): Since it is a timely response and no delay is required, leave it blank.

(9) Set default: When the selection logic is not established, whether DO1 restores the default state, select according to the requirements.

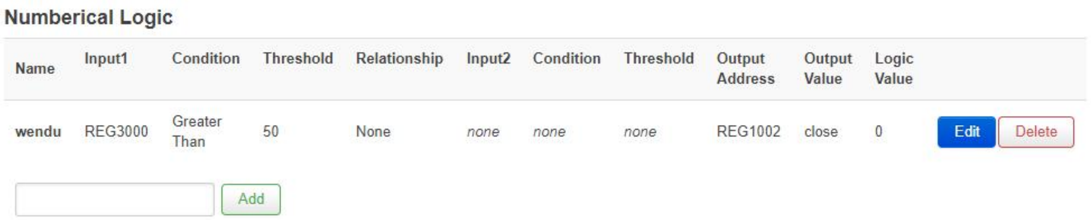
(10) Click "Save".

(11) Follow the same steps to build other logic.

(12) Click "Save and Apply" to write into the BL200 coupler.

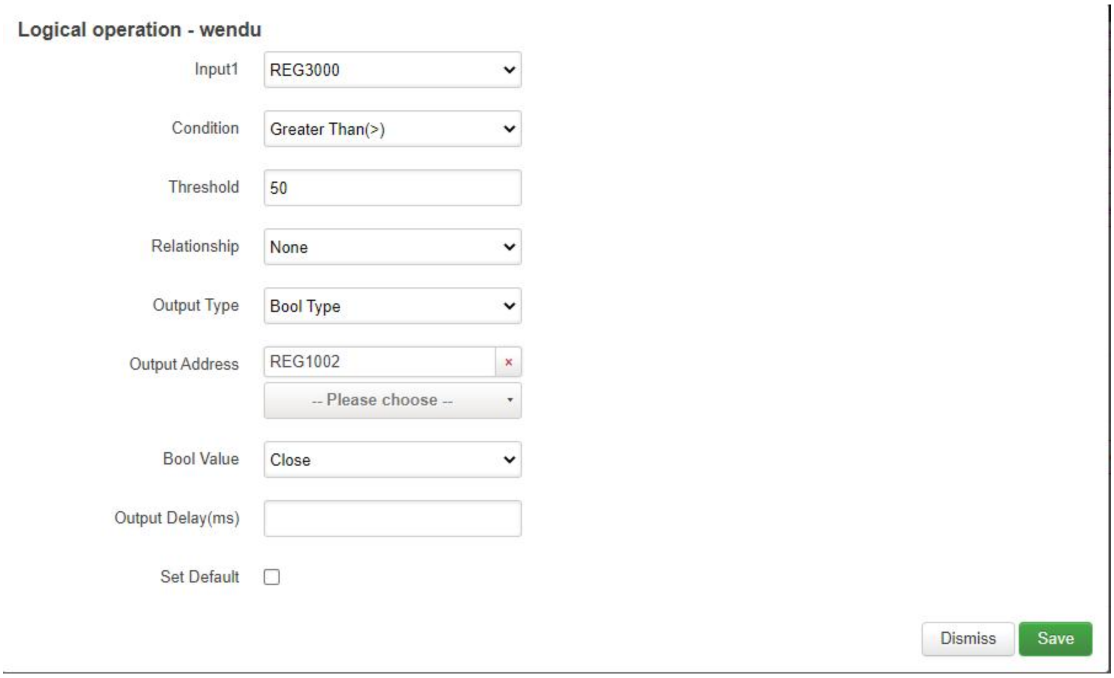
5.7.6.4.2 Numerical Logic Configuration Example

The AI1 register REG3000 is connected to the temperature sensor to monitor the temperature of the motor. When the collected temperature is greater than 50, the fan is turned on, and the fan is controlled by the DO3 register REG1002.



Numberical Logic

Name	Input1	Condition	Threshold	Relationship	Input2	Condition	Threshold	Output Address	Output Value	Logic Value	
wendu	REG3000	Greater Than	50	None	none	none	none	REG1002	close	0	Edit Delete



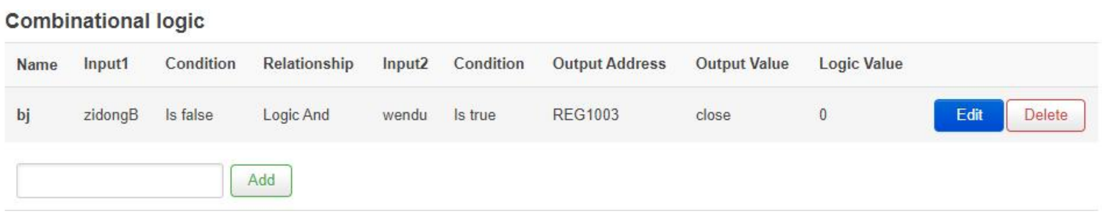
Logical operation - wendu

Input1	REG3000
Condition	Greater Than(>)
Threshold	50
Relationship	None
Output Type	Bool Type
Output Address	REG1002 × -- Please choose --
Bool Value	Close
Output Delay(ms)	
Set Default	□

Similarly, numerical logic and Bool logic have the same logic principle. Numerical logic only judges that the condition is "greater than", "less than" or "equal to" a certain value as a linkage condition.

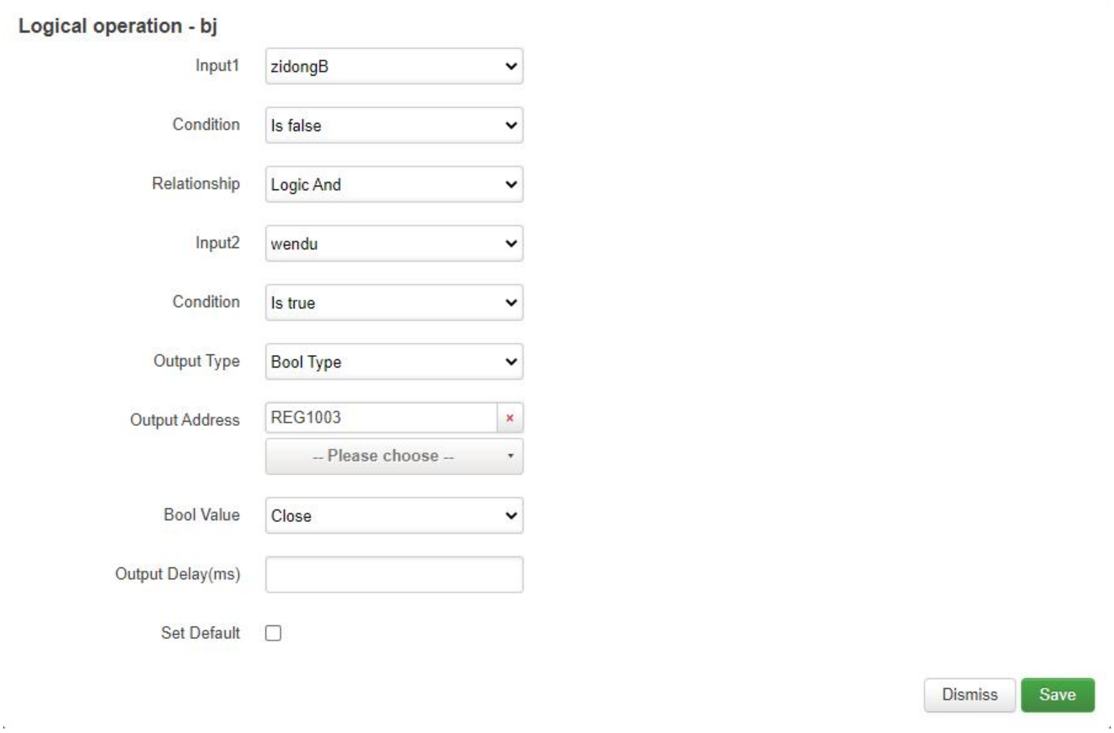
5.7.6.4.3 Combinational Logic Example

The conveyor belt is not running, the temperature of the motor exceeds 50 degrees, the fan is turned on, and the alarm DO4 register REG1003 is triggered.



Combinational logic

Name	Input1	Condition	Relationship	Input2	Condition	Output Address	Output Value	Logic Value	
bj	zidongB	Is false	Logic And	wendu	Is true	REG1003	close	0	Edit Delete



Logical operation - bj

Input1	zidongB
Condition	Is false
Relationship	Logic And
Input2	wendu
Condition	Is true
Output Type	Bool Type
Output Address	REG1003 -- Please choose --
Bool Value	Close
Output Delay(ms)	
Set Default	□

Steps:

(1) In the Combinational Logic item, input the name "bj", click Add, and the configuration box will pop up.

(2) Input 1: Select the logic name "zidongB" built in Bool logic before, you can choose Bool logic or numerical logic according to your demand.

(3) Condition: Select "Is false", according to your demand, whether the logic selected by input 1 is triggered or not as a condition.

(4) Relationship: Select "Logic And" to choose, according to your demand, the logical relationship between condition 1 and condition 2, you can also select "no" condition 2.

(5) Input 2: Select the logic name "wendu", choose Bool logic or numerical logic according to your demand.

(6) Condition: Select "Is true", according to your demand, whether the logic selected by input 2 is triggered or not as a condition.

(7) Output Type: Select "Bool Type", select Bool or numeric data according to "Output Address".

(8) Output address: Select the register address to be operated. DO4 register REG1003.

(9) Bool value: Close, DO4 closed to control the alarm

(10) Output delay (milliseconds): It is a timely response, there is no need for a delay, so do not fill in.

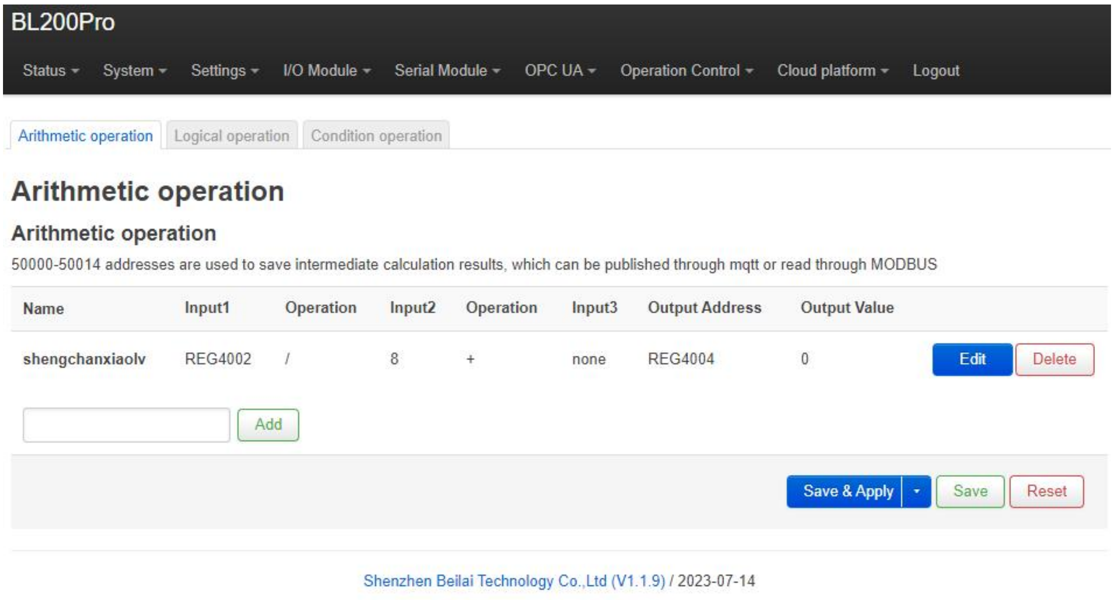
(11) Set default: Choose whether to restore the default state of DO4 when the logic is not valid, according to your demand.

(12) Click "Save".

(13) Click "Save and Apply" to write into BL200 coupler.

5.7.6.4.4 Arithmetic Operation Configurations

The sensor collects the quantity produced in a day and stores it in register REG40002, and through the arithmetic function it calculates the quantity produced in each hour of an 8-hour day and stores it in register REG40004, and the data in register REG40004 can be sent to your platform or server through MQTT, OPC UA or Modbus.



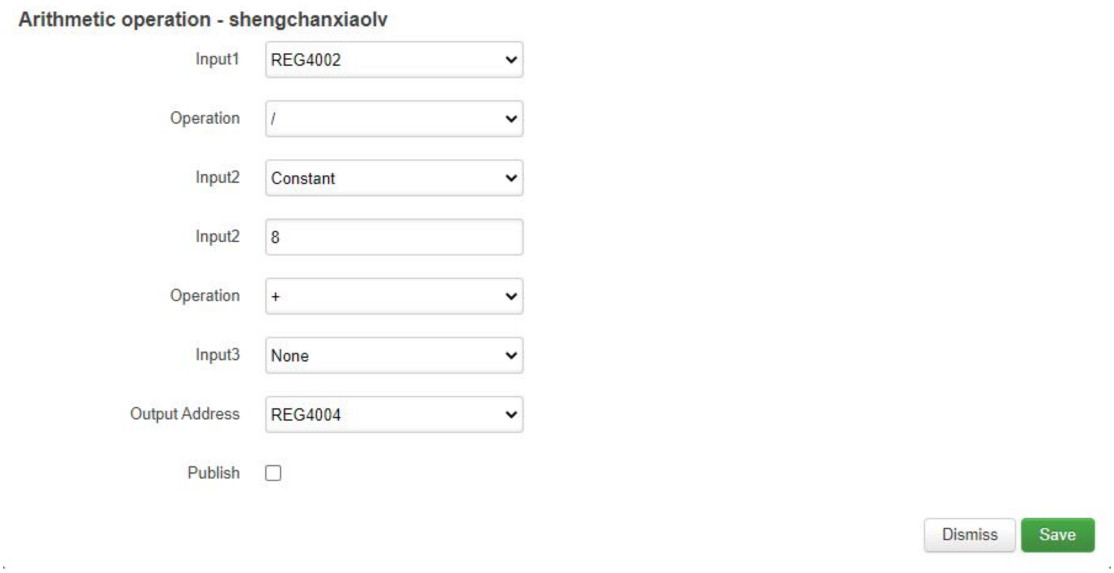
Arithmetic operation

Arithmetic operation

50000-50014 addresses are used to save intermediate calculation results, which can be published through mqtt or read through MODBUS

Name	Input1	Operation	Input2	Operation	Input3	Output Address	Output Value	
shengchanxiaolv	REG4002	/	8	+	none	REG4004	0	Edit Delete

Shenzhen Beilai Technology Co.,Ltd (V1.1.9) / 2023-07-14



Arithmetic operation - shengchanxiaolv

Input1: REG4002

Operation: /

Input2: Constant

Input2: 8

Operation: +

Input3: None

Output Address: REG4004

Publish: □

Steps

(1) Enter the name "shengchanxiaolv", click Add, and a configuration box will pop up.

(2) Input 1: Select the yield register REG40002.
(3) Operation: Select "/", you can select "add, subtract, multiply and divide" here according to your demand.
(4) Input 2: Select Constant, you can select other register address according to your demand.
(5) Input 2: Fill in the constant because constant is selected, when select a register, there is no such item.
(6) Operation: According to whether there is also a condition 3 selection, if not, then it doesn't matter.
(7) Input 3: Select "none", because there is no need for this condition option, you can also choose registers, constants, none.
(8) Output Address: Select the register address to store the result of the operation.
(9) Click "Save".
(10) Click "Save and Apply" to write into the BL200 coupler.

6 Fieldbus Communication Example

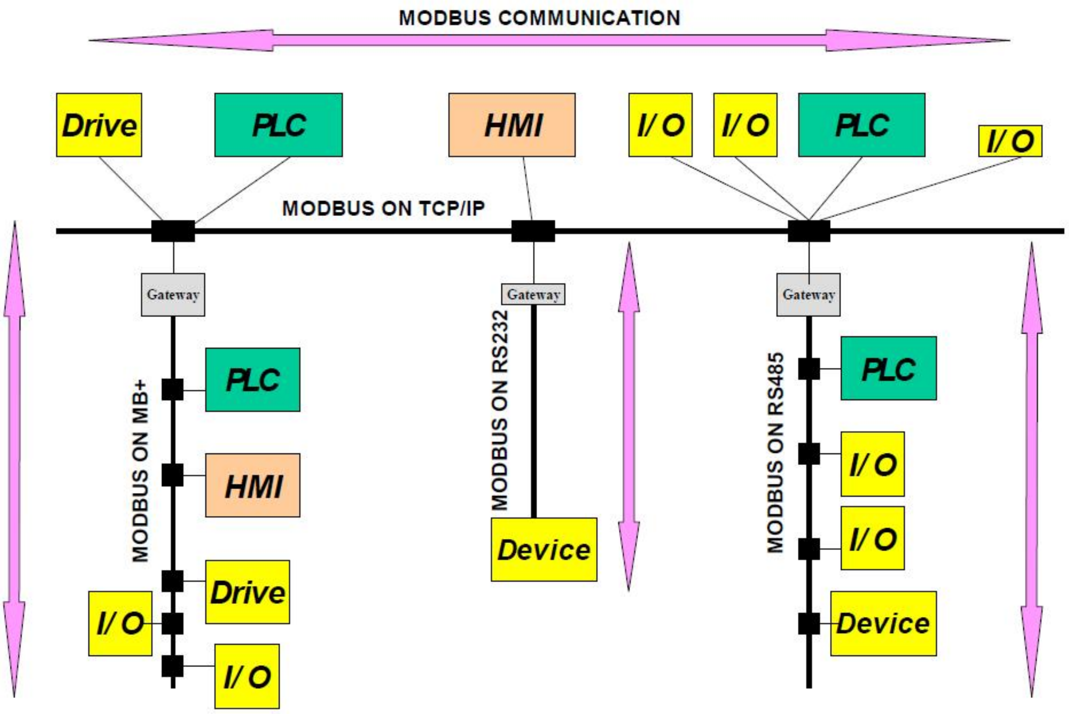
6.1 BL200 Communication Example

6.1.1 Overview

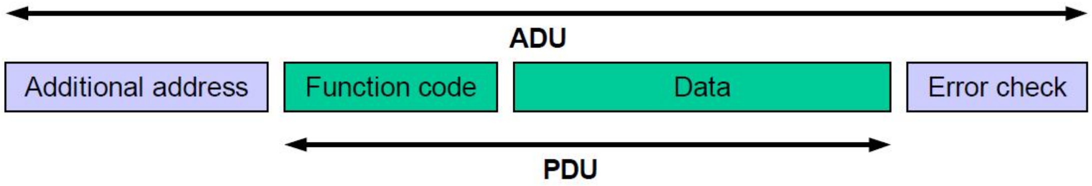
Modbus is an open, manufacturer-independent fieldbus standard protocol for a variety of applications in manufacturing and process automation.
MODBUS is an application layer messaging protocol at layer 7 of the OSI model that enables client/server communication between devices connected on different types of buses or networks.
Several commonly used networks are as follows:

- TCP/IP over Ethernet
- Asynchronous serial transmission of multiple media (wired: EIA/TIA-232-E, EIA-422, EIA/TIA-485-A; optical fiber, radio, etc.).
- MODBUS PLUS, high-speed token.

MODBUS is a request/response protocol that provides services specified by function codes.
The MODBUS protocol allows easy communication within all types of network architectures.



MODBUS protocol defines a simple protocol data unit (PDU) independent of the underlying communication layer. The mapping of the MODBUS protocol on a specific bus or network can introduce some additional fields on the Application Data Unit (ADU).



6.1.1.1 Modbus TCP

The Modbus TCP protocol is a variant of the Modbus protocol that is optimized for communication over a TCP/IP connection. The protocol is designed for data exchange at the field level (ie for I/O data exchange in the process image). On the server side, all packets are sent over a TCP connection with port number 502.

The general Modbus TCP message is as follows:

byte	0	1	2	3	4	5	6	7	8 - n
Definition	Transaction identifier		Protocol identifier(Always 00)		Field length		Slave address	Modbus function code	Data

6.1.1.2 Modbus Data Encoding

Modbus uses "big endian" representation for address and data items. This means that when transferring numbers larger than a single byte, the most significant byte is sent first.

6.1.1.3 Modbus Data Type

The modbus protocol is based on the following basic data types:

Data type	**Object type**	**Access type**	**Description**
Digital input	1 bit	read	Digital input
Coil	1 bit	read/write	Digital output
Input register	16 bit (word)	read	Analog input
Holding register	16 bit (word)	read/write	Analog output

For each basic data type, one or more function codes are defined. These function codes allow digital or analog input and output data, as well as internal variables to be set or read directly from the fieldbus node.

6.1.2 Modbus Function Code

The function codes supported by the BL200 fieldbus node are shown in the table below. To perform the required functions, please specify the respective function codes and the address of the selected input or output channel or register.

Modbus function code	Function	**Access type**	**Description**
0x02	read digital input	read	Access by 1 bit
0x01	read coil	read/write	
0x05	write a single coil	read/write	
0x0F	write multiple coils	read/write	
0x04	read input register	read	Access by 16 Bit
0x03	read multiple registers	read/write	
0x06	write a single register	read/write	
0x10	write multiple registers	read/write	

The MODBUS function is performed as follows:

1. The MODBUS TCP master (such as PC) sends a request to the BL200 fieldbus node using a specific function code;
2. The BL200 fieldbus node receives the data message, and then responds to the master with correct data according to the master's request.

If a fieldbus node receives an incorrect request, it sends an error data telegram (exception) to the master.

The meaning of the exception code contained in the exception is as follows:

Exception code	Description
0x01	illegal function
0x02	illegal data address
0x03	illegal data value
0x04	slave device failure

6.1.2.1 Function Code 0x02

This function code is used to read the continuous state of single or multiple digital inputs.

1. Request

The request specifies the starting address and the quantity to be read.

Field Name	Number of bytes	Example	Description
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x02	Read digital input, use function code 0x02
Start address	2 Byte	0x07 D0	The address is detailed in the "Modbus Register Mapping" chapter
Enter quantity	2 Byte	0x08	Read 8 digital inputs

2. Response

The data field indicates the value of the input state. A binary 1 corresponds to the on state and a 0 corresponds to the off state. The least significant bit (LSB) of the first data byte contains the first bit of the request, the others are in ascending order. If the response data is not a multiple of 8, the remaining bits of the last data byte will be padded with zeros (towards the upper bits of the byte).

Field Name	Number of bytes	Example	Description
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 04	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x02	Read digital input, use function code 0x02
Data bytes	1 Byte	0x01	Number of bytes of data
Data	1 Byte	0x89	Response data

3. Abnormal

Field Name	Number of bytes	Example	Description
…			
Function code	1 Byte	0x82	Modbus function code + 0x80
Abnormal code	1 Byte	0x01	0x01 or 0x02

4. Example

Read the value of 8 digital inputs from address 2000 to 2007.

request

0x00 01 00 00 00 06 01 02 07 D0 00 08

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	01	07 D0		00 08	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Start address		Number of coils	

response

0x00 01 00 00 00 04 01 02 01 89

Byte	1	2	3	4	5	6	7	8	9	10
Data	00 01		00 00		00 04		01	01	01	89
illust rate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Data bytes	Data

Status from 2007 to 2000 is displayed as byte value 0x89 or binary 1000 1001. Address 2007 is the most significant bit MSB of the byte, 2000 is the least significant bit LSB, the distribution from high to low is as follows:

Bit	7	6	5	4	3	2	1	0
Address	2007	2006	2005	2004	2003	2002	2001	2000
Status	1	0	0	0	1	0	0	1
illustrate	close	open	open	open	close	open	open	close

6.1.2.2 Function Code 0x01

This function code is used to read the continuous status of single or multiple coils in the remote device.

1. Request

The request specifies the starting address, which specifies the address of the first coil, and the number of coils.

Field Name	**Number of bytes**	**Example**	**illustrate**
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x01	Read coil, use function code 0x01
Start address	2 Byte	0x03 E8	The address is detailed in the "Modbus Register Mapping" chapter
Number of coils	2 Byte	0x00 08	Read 8 coil states

2. Response

The data field indicates the value of the input state. A binary 1 corresponds to the on state and a 0 corresponds to the off state. The least significant bit (LSB) of the first data byte contains the first bit of the request, the others are in ascending order. If the response data is not a multiple of 8, the remaining bits of the last data byte will be padded with zeros (towards the upper bits of the byte).

Field Name	**Number of bytes**	**Example**	**illustrate**
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 04	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x01	Read coil, use function code 0x01
Data bytes	1 Byte	0x01	Number of bytes of data
Data	1 Byte	0x89	Response data

3. Abnormal

Field Name	**Number of bytes**	**Example**	**illustrate**
…			
Function code	1 Byte	0x81	Modbus function code + 0x80
Abnormal code	1 Byte	0x01	0x01 or 0x02

4. Example

Read the status values of 8 coils from addresses 1000 to 1007.

request

0x00 01 00 00 00 06 01 01 03 E8 00 08

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	01	03 E8		00 08	
illustr	Transaction		Protocol		Message		Device	Function	Initial		Number of	

ate	identifier	identifier	length	address	code	address	coils

response

0x00 01 00 00 00 04 01 01 01 89

Byte	1	2	3	4	5	6	7	8	9	10
Data	00 01		00 00		00 04		01	01	01	89
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Data bytes	Data

Status from 1007 to 1000 is displayed as byte value 0x89 or binary 1000 1001. Address 1007 is the most significant bit MSB of the byte, 1000 is the least significant bit LSB, the distribution from high to low is as follows:

Bit	7	6	5	4	3	2	1	0
Address	1007	1006	1005	1004	1003	1002	1001	1000
Status	1	0	0	0	1	0	0	1
illustrate	close	open	open	open	close	open	open	close

6.1.2.3 Function Code 0x05

This function will write a single coil status to the slave device.

1. Request

Field Name	**Number of bytes**	**Example**	**illustrate**
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x05	To write a single coil, use function code 0x05
Register address	2 Byte	0x03 E8	The address is detailed in the "Modbus Register Mapping" chapter
Data input	2 Byte	0xFF 00	This value is: 0xFF 00 or 0x00 00. 0xFF

			00 means write 1, 0x00 00 means write 0.

2. Response

Field Name	**Number of bytes**	**Example**	**illustrate**
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x05	To write a single coil, use function code 0x05
Data bytes	2 Byte	0x03 E8	Write the register address of the coil
Data input	2 Byte	0xFF 00	This value is: 0xFF 00 or 0x00 00. 0xFF 00 means write 1, 0x00 00 means write 0.

3. Abnormal

Field Name	**Number of bytes**	**Example**	**illustrate**
…			
Function code	1 Byte	0x85	Modbus function code + 0x80
Abnormal code	1 Byte	0x81	0x01 or 0x02

4. Example

Write the state value of the coil at address 1000 as 1, that is, the closed state.

request

0x00 01 00 00 00 06 01 05 03 E8 FF 00

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	05	03 E8		FF 00	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Coil address		Write "1"	

response

0x00 01 00 00 00 06 01 05 03 E8 FF 00

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	05	03 E8		FF 00	
illust rate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Coil address		Write "1"	

6.1.2.4 Function Code 0x0F

This function code is used to set multiple consecutive coils to open or close. The on/off state of the request is specified by the content of the request data field. A logical "1" requests the corresponding output to close, and a logical "0" requests it to open. The normal response returns the function code, the starting address and the number of coils executed.

1. Request

Field Name	number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 08	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x0F	Write multiple coils, use function code 0x0F
Start address	2 Byte	0x03 E8	The address is detailed in the "Modbus Register Mapping" chapter
Number of coils	2 Byte	0x00 08	
Data bytes	1 Byte	0x01	
Data	1 Byte	0xFF	

2. Response

Field Name	number of	Example	illustrate

	bytes		
Transaction identifier	2 Byte	0x00 00	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x0F	Write multiple coils, use function code 0x0F
Start address	2 Byte	0x03 E8	
Number of coils	2 Byte	0x00 08	

3. Abnormal

Field Name	number of bytes	Example	illustrate
…			
Function code	1 Byte	0x8F	Modbus function code + 0x80
Abnormal code	1 Byte		0x01 or 0x02

4. Example

Starting from address 1000, close all 8 coils, that is, write the value of 8 coils as 0xFF.

request

0x00 01 00 00 00 08 01 0F 03 E8 00 08 01 FF

Byte	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Data	00 01		00 00		00 08		01	0F	03 E8		00 08		01	FF
illust rate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Start address		Number of coils		Data bytes	Data

response

0x00 01 00 00 00 06 01 0F 03 E8 00 08

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	0F	03 E8		00 08	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Start address		Number of coils	

6.1.2.5 Function Code 0x04

This function code is used to read consecutive input registers in multiple remote devices. The request PDU specifies the address of the starting register and the number of registers. The register data in the response message is packed into two bytes per register, and the binary content within each byte is right-aligned.

1. Request

Field Name	Number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x04	Read input register, use function code 0x04
Start address	2 Byte	0x0B B8	The address is detailed in the "Modbus Register Mapping" chapter
Number of registers	2 Byte	0x00 08	

2. Response

Field Name	Number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 00	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 13	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x04	Read input register, use function code 0x04
Data bytes	1 Byte	0x10	

Data	16 Byte	0x 3F 8E 38 86 40 0E 38 86 40 55 54 CA 40 8E 35 3F	

3. Abnormal

Field Name	Number of bytes	Example	illustrate
…			
Function code	1 Byte	0x84	Modbus function code + 0x80
Abnormal code	1 Byte	0x01	0x01 or 0x02

4. Example

Starting at address 3000, read the values of the 4 analog inputs. Since the BL200 controller node register map data type is 32Bit Float, that is, 1 analog input data = 2 registers = 4 bytes, 8 input registers need to be read.

request

0x00 01 00 00 00 06 01 04 0B B8 00 08

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	04	0B B8		00 08	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Start address		Number of registers	

response

0x00 01 00 00 00 13 01 04 10 3F 9D 70 A4 40 15 C2 8F 40 5C CC CD 40 91 EB 85

Byte	1	2	3	4	5	6	7	8	9	10…25
Data	00 01		00 00		00 13		01	04	10	xxx
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Data bytes	Data

The data part has a total of 16 bytes, which are converted into decimal as follows

Byte	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Data	3F 9D 70 A4				40 15 C2 8F				40 5C CC CD				40 91 EB 85			

Decimal	1.23	2.34	3.45	4.56
illustrate	First data	Second data	Third data	Fourth data

6.1.2.6 Function Code 0x03

This function code is used to read continuous holding registers in multiple remote devices. The request PDU specifies the address of the starting register and the number of registers. The register data in the response message is packed into two bytes per register, and the binary content within each byte is right-aligned.

1. Request

Field Name	number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x03	Read holding register, use function code 0x03
Start address	2 Byte	0x0F A0	The address is detailed in the "Modbus Register Mapping" chapter
Number of registers	2 Byte	0x00 08	Number of holding registers to read

2. Response

Field Name	Number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 00	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 13	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification

Function code	1 Byte	0x03	Read holding register, use function code 0x03
Data bytes	1 Byte	0x10	Data bytes
Data	16 Byte	0x 3F 9D 70 A4 40 15 C2 8F 40 5C CC CD 40 91 EB 85	Response data

3. Abnormal

Field Name	**Number of bytes**	**Example**	**illustrate**
…			
Function code	1 Byte	0x83	Modbus function code + 0x80
Abnormal code	1 Byte	0x01	0x01 or 0x02

4. Example

Starting at address 4000, read the values of the 4 analog outputs (belonging to the holding registers). Since the analog output I/O module register map data type is 32Bit Float, that is, 1 analog output data = 2 registers = 4 bytes, it is necessary to read 8 holding registers.

request

0x00 01 00 00 00 06 01 03 0F A0 00 08

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	03	0F A0		00 08	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Start address		Number of registers	

response

0x00 01 00 00 00 13 01 03 10 3F 9D 70 A4 40 15 C2 8F 40 5C CC CD 40 91 EB 85

Byte	1	2	3	4	5	6	7	8	9	10…25
Data	00 01		00 00		00 13		01	03	10	xxx
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Data bytes	Data

The data part has a total of 16 bytes, and the conversion to decimal is as follows:

Byte	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Data	3F 9D 70 A4				40 15 C2 8F				40 5C CC CD				40 91 EB 85			
Decimal	1.23				2.34				3.45				4.56			
illustrate	First data				Second data				Third data				Fourth data			

6.1.2.7 Function Code 0x06

This function code is used to write to holding registers in a single remote device. The request PDU specifies the address of the starting register and the number of registers. The register data in the response message is packed into two bytes per register, and the binary content within each byte is right-aligned.

This function code is only suitable for reading the serial port I/O module register mapping data, the address range: 40000 ... 49999. The data type of the analog input/output I/O module is 32Bit Float format, the complete data cannot be read, and this function cannot be used.

1. Request

Field Name	**Number of bytes**	**Example**	**illustrate**
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x06	Write a single holding register, use function code 0x06
Register address	2 Byte	0x9C 40	The address is detailed in the "Modbus Register Mapping" chapter
Data	2 Byte	0x04 D2	

2. Response

Field Name	**Number of bytes**	**Example**	**illustrate**
Transaction identifier	2 Byte	0x00 00	Identification of Modbus request/response transactions

Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 06	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x06	Write a single holding register, use function code 0x06
Register address	2 Byte	0x75 30	
Data	2 Byte	0x04 D2	

3. Abnormal

Field Name	**Number of bytes**	**Example**	**illustrate**
…			
Function code	1 Byte	0x86	Modbus function code + 0x80
Abnormal code	1 Byte	0x01	0x01 or 0x02

4. Example

Write the value of register address 40000 to 1234 (0x04 D2).

request

0x00 01 00 00 00 06 01 06 9C 40 04 D2

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	06	9C 40		04 D2	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Register address		Data	

response

0x00 01 00 00 00 06 01 06 9C 40 04 D2

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	0F	9C 40		04 D2	
illustrate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Register address		Data	

6.1.2.8 Function Code 0x10

This function code is used to write to consecutive holding registers in multiple remote devices. The request PDU specifies the address of the starting register and the number of registers. The register data in the response message is packed into two bytes per register, and the binary content within each byte is right-aligned.

1. Request

Field Name	Number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 01	Identification of Modbus request/response transactions
Protocol identifier	2 Byte	0x00 00	0x00 00: Modbus protocol
Message length	2 Byte	0x00 17	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x10	Write multiple holding registers, use function code 0x10
Start address	2 Byte	0x0F A0	The address is detailed in the "Modbus Register Mapping" chapter
Number of registers	2 Byte	0x00 08	
Data bytes	1 Byte	0x10	
Data	16 Byte	0x 3F 9D 70 A4 40 15 C2 8F 40 5C CC CD 40 91 EB 85	

2. Response

Field Name	Number of bytes	Example	illustrate
Transaction identifier	2 Byte	0x00 00	Identification of Modbus request/response transactions
Protocol	2 Byte	0x00 00	0x00 00: Modbus protocol

identifier			
Message length	2 Byte	0x00 13	The number of bytes of the following data
Device address	1 Byte	0x01	Slave address identification
Function code	1 Byte	0x10	Write multiple holding registers, use function code 0x10
Start address	2 Byte	0x0F A0	
Number of registers	2 Byte	0x00 08	

3. Abnormal

Field Name	**number of bytes**	**Example**	**illustrate**
…			
Function code	1 Byte	0x90	Modbus function code + 0x80
Abnormal code	1 Byte	0x01	0x01 or 0x02

4. Example

Starting at address 4000, write the values of the 4 analog outputs. Since the BL200 controller node register map data type is 32Bit Float, that is, 1 analog output data = 2 holding registers = 4 bytes, 8 holding registers need to be written.

request

0x00 01 00 00 00 17 01 10 0F A0 00 08 10 3F 9D 70 A4 40 15 C2 8F 40 5C CC CD 40 91 EB 85

Byte	1	2	3	4	5	6	7	8	9	10	11	12	13	14…23
Data	00 01		00 00		00 17	01		10	0F A0		00 08		10	xxx
illustrate	Transaction identifier		Protocol identifier		Message length	Device address		Function code	Start address		Number of registers		Data bytes	Data

The data part has a total of 16 bytes, and the conversion to decimal is as follows:

Byte	14															
Data	3F 9D 70 A4				40 15 C2 8F				40 5C CC CD				40 91 EB 85			
Decimal	1.23				2.34				3.45				4.56			
illustrate	First data				Second data				Third data				Fourth data			

response

0x00 01 00 00 00 06 01 10 0F A0 00 08

Byte	1	2	3	4	5	6	7	8	9	10	11	12
Data	00 01		00 00		00 06		01	10	0F A0		00 08	
illust rate	Transaction identifier		Protocol identifier		Message length		Device address	Function code	Start address		Number of registers	

7 Warranty

1) This equipment will be repaired free of charge for any material or quality problems within one year from the date of purchase.
2) This one-year warranty does not cover any product failure caused by man-made damage, improper operation, etc.

8 Technical Support

Shenzhen Beilai Technology Co., Ltd
Website: https://www.bliiot.com