BORUNTE Process Instruction

Borunte Modbus TCP Communication Protocol Manual

BRT_EN_V1.1_20234.10.9

- www.borunte.com -

Z Tao = X Method + Y Tactics + 1 Tool

||

Practical Philosophy + Yin's Economics +
Regulations Governing Enterprises = Common Prosperity

Practical Philosophy

II

Self-discipline + Practical + Simplicity + All beings are equal

II

BORUNTE goals:

BORUNTE's goal: Annual net profit ⩾ 1 yuan
Step 1: Realize an annual operating income of over
RMB 1 trillion and obtain over 10,000 licensed invention patents.
Step 2: Realize an annual operating income of over
RMB 10 trillion and obtain over 100,000 licensed invention patents.
Step 3: Realize an annual operating income of over
RMB 100 trillion and obtain over 1,000,000 licensed invention patents.

+

Common Prosperity Methodology

BORUNTE: BORUNTE adopts "BOM pricing method", this is the people's longing for a better life, and also the goal that BORUNTE strives for.

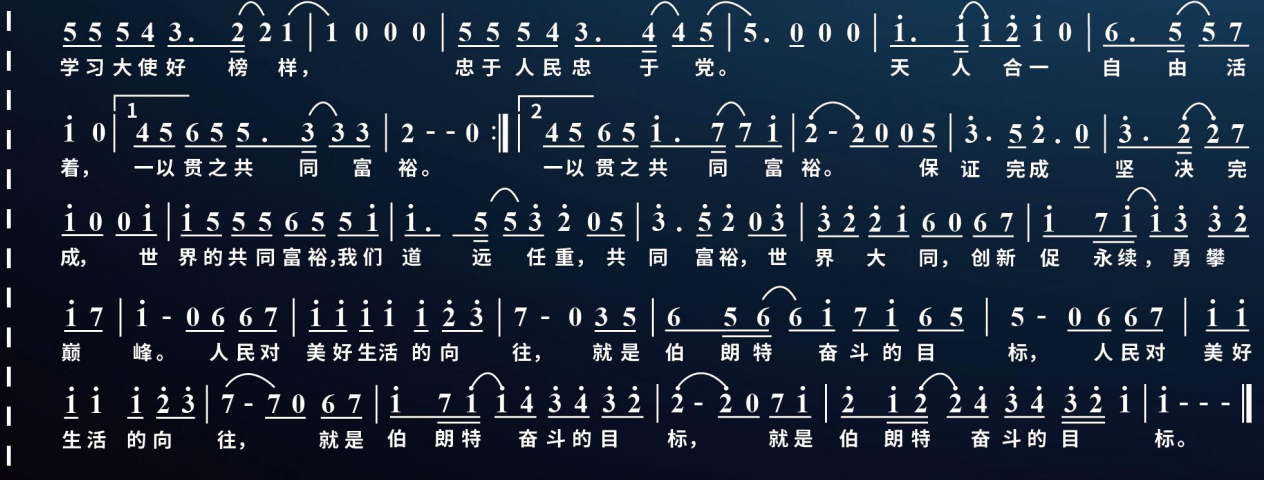
Integrator: Place an order for 1000 sets BORUNTE products of a single model, then you can become the integrator of BORUNTE, you should complete the payment and delivery within 120 months. BORUNTE provide 50% rebate for integrator. And the rebate can be cashed if you place an order again and the order quantity must be greater than twice the number of rebates.

Supplier: The payment method is cash transfer for the current month, which is the reconciliation period from the 16th of the previous month to the 15th of this month. The cash transfer will be completed before the 25th of this month and under the premise of ensuring quality and the cost has priority, the supplier will become the primary supplier.

+

学习大使好榜样

1=C $\frac{4}{4}$



"Yin's Economics"

The material world of common prosperity is chief daily necessities, while the spiritual world is additional items of life. In addition, BORUNTE 's "Practical Philosophy" is "harmony between heaven and humanity, and common prosperity", while in Ambassador Yin's "Yin's Economics", the philosophy is "negative gross profit margin and positive cash flow", which is "the unity of Taoism and invincibility". In other words, BORUNTE can achieve "eternal life".

If a person doesn't have it, they want to have it, but once they have it, they want more and worry about losing it. This is why you can have money, not more than me. But the unchanging pursuit of humanity for 5000 years is: you don't envy me for being richer than you, and I don't envy you for being richer than me. That is to say, in the market, earning is also what you should earn, and losing is also what you should lose. And the common prosperity can only be achieved by yourself. This is the market where all things grow, all flowers bloom, a thousand people and a thousand faces and stand firm for a thousand years. That is to say, Ambassador Yin is just a "small potato" in the market.

Integrator is people, and people can be integrator, supplier is people and people can be supplier. That is, what the people need, what the BORUNTE produce. That is to say, the people will never refuse the “negative gross profit margin positive cash flow” BORUNTE products. But the people will definitely worry about the “negative gross margin profit negative cash flow” BORUNTE products. That is the end of “negative gross profit margin positive cash flow” is “positive gross profit margin and positive cash flow” , which means that BORUNTE has achieved common prosperity under the innovative drive of “positive gross profit margin and positive cash flow”.

The embryonic form of the BORUNTE's "black hole economy" with "negative gross profit margin and positive cash flow" has emerged, but at its end lies the "white hole economy" with "positive gross profit margin and positive cash flow". This is the formation of "Yin's economics" in "The Market Theory" and "Black and White Hole Economy", but its underlying logic is actually BORUNTE's "Practical Philosophy". That is to say, the theoretical innovation of "practical philosophy+Yin's economics=common prosperity" has been completed.

Regulations Governing Enterprises

On April 1, 2024, BORUNTE officially began the era of "Regulations Governing Enterprises"

One unity pervading all things=BORUNTE target: annual net profit>1 yuan=Integrator buying and selling BORUNTE product=Supplier quality and cost=Common Prosperity

1 tool=Regulations governing enterprise=Integrator buying and selling BORUNTE product=Supplier quality and cost

Safety Precautions

Before using this process package, please ensure that you thoroughly read and understand this specification and any accompanying materials. This specification categorizes safety precautions into "Danger," "Caution," "Mandatory," and "Prohibition."

Caution: Mishandling may pose risks, possibly resulting in moderate injuries, minor accidents, or equipment malfunction.

Prohibition: Actions that are strictly forbidden.

Mandatory: Actions that must be strictly adhered to.

It should be noted that even items listed under "Caution" can have serious consequences depending on the situation. Hence, it is crucial to strictly adhere to all instructions.

Danger: Mishandling may lead to deaths or severe accidents.

Danger

☆Press the emergency stop button on the demonstrator, confirm that main power supply of servo motor has been cut off, the motor is powered off and under braking system before operating the robot. Once the servo power is turned off, the teaching pendant will trigger an emergency stop alarm, the control cabinet's red alarm light will flash, and the buzzer will sound. In an emergency, if the robot cannot be stopped promptly, it may result in personal injury or equipment damage.

Emergency Stop Button

☆When servo power supply needs to be switched on after releasing emergency stop, remove the fault that triggers emergency stop before connecting the servo power supply. Actions due to mishandling of the robot may result in serious injuries.

☆Please follow the rules below when making demonstration within the activity range of robot: Observe the robot from the front side

Follow the operation steps strictly

Consider contingency plans if the robot unexpectedly moves towards your location and ensure there is a designated escape route just in case.

Actions due to mishandling of the robot may result in serious injuries.

☆Make sure there's no person within the motion range of robot and operator is in a safe position before starting the following operation: Switch on power supply of robot control cabinet.

When programming the robot using the teaching pendant. During test runs.

In automatic mode.

Accidentally entering the robot's operational range or making contact with the robot may result in serious injuries. In case of abnormalities, immediately press the emergency stop.


Caution

☆Robot operations must be confirmed.
The operator has received safety training
The operator has adequate understanding of the robot's movement characteristics.
The operator has sufficient knowledge of the robot's hazards
Do not operate after drinking

☆Before using the robot teaching device, check the following points and address any anomalies promptly or take other necessary measures.
Robot movement is normal
The origin is calibrated correctly
External auxiliary devices associated with the robot are functioning properly.

☆Put the demonstrator back and fix it after use.
If the teaching device is carelessly left on the robot, fixture, or ground, it may collide with the robot or fixture when the robot moves, potentially causing injury or equipment damage.
Prevent accidental dropping of the teaching device as it might cause unintended robot movement, leading to injury or equipment damage.


Mandatory

Safety operating procedures
All robot system operators should undergo the training on system in order to learn the safety protection measures and robot functions.
Check if the robot and peripherals are normal before running the robot.
Switch off power supply or press the emergency stop button before entering the operation area, even if the robot is not running.
Assign a specific supervisor when robot is programming in working area, to make sure the robot can stop quickly in case of emergency.
Do not wear gloves when the demonstrator inches robot. Low speed inching is preferred. Stop the robot effectively in case of a fault
Master the position of emergency stop buttons on the robot and peripheral control equipment, in order to press them correctly in case of emergency
7. Never assume that when the robot is stopped, its program has finished. At that time, the robot may be waiting for an input signal to continue its movement

catalogue

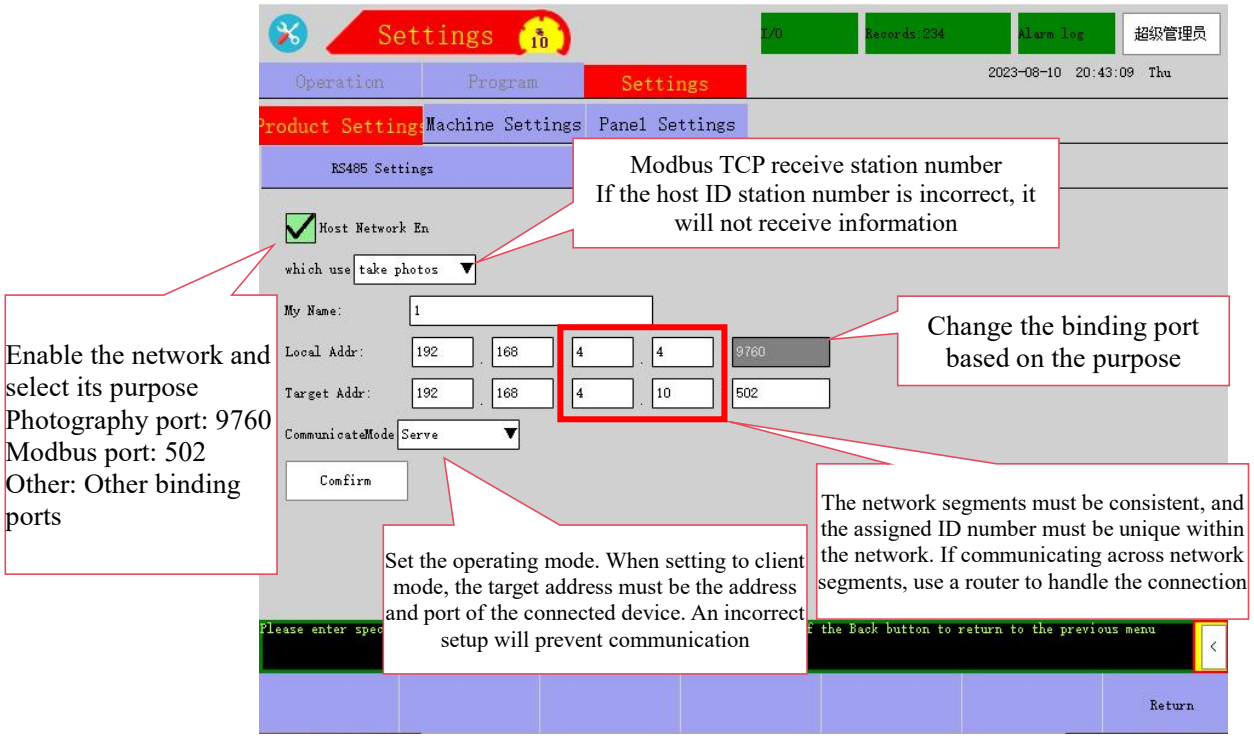
Borunte Modbus TCP Communication Protocol Manual

1. Communication settings

1.1 Ethernet port connection

1.2 Communication settings

Stop Mode -> Product Settings -> Communication Configuration -> Host Network Settings

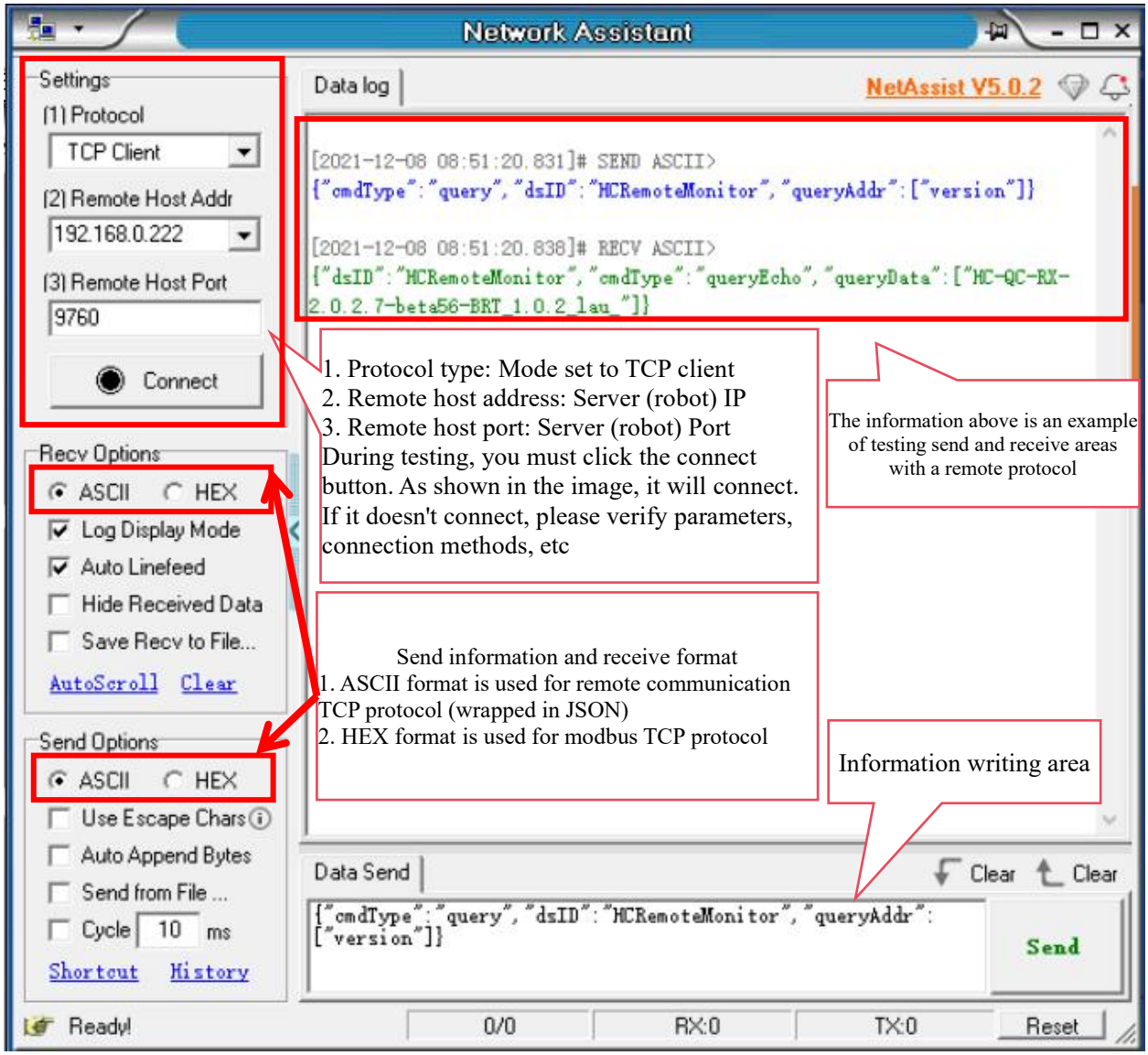


After setting the parameters, click confirm for the changes to take effect

Note: If there are no special requirements, the general communication mode is set to server!!!

1.3 Communication testing

Please use network debugging tools or similar software to verify if the settings are correct. Below is an example when the robot is set to server:



Note: Set it up in the manner described above, ensuring network protocol settings correspond with robot settings. Once connected, communicate per protocol. If the robot's remote TCP protocol messages or modbus TCP data send and receive correctly, communication can be confirmed as normal

If there's no response, please confirm:

1. The sent information format is correct
2. The content of the information is correct
3. Whether the sent command is compatible with the version, etc

Borunte Robot System ModbusTCP Slave Guide

Version: 1.0

2 Overview

Using the modbusTCP protocol, the host acts as a Modbus slave

2.1 Data in the examples are all in hexadecimal format

2.1.1 Data format:

(1) In bytes, with the high byte first and the low byte last

(2) 16-bit data: Occupies one register, with the high 8 bits first and the low 8 bits last during transmission

(3) 32-bit data: Occupies 2 registers, with the high 16-bit data in the low address and the low 16-bit data in the high address

Address	Value
Addr0	bit31~bit16
Addr1	bit15~bit0

(4) For 64-bit data: Occupies 4 registers, the highest 16-bit data is in the lowest address, and the lowest 16-bit data is in the highest address;

Address	Value

Addr0	bit63~bit48
Addr1	bit47~bit32
Addr2	bit31~bit16
Addr3	bit15~bit0

2.1.2 Request APU example

Description		Size	Instance
MBAP header	Transaction identifier HL	1	0x15
	Transaction identifier Lo	1	0x01
	Protocol identifier	2	0x0000
	Length	2	0x0006
	Unit identifier	1	0xFF
MODBUS request	Function code	1	0x03
	Starting address	2	0x0005
	Number of registers	2	0x0001

3 Address definition and operational requirements

3.1 Read register operation (0x03)

3.1.1 Read version number length

Request version number length:
Address: 0x0000
Number of registers: 1
Example: 00 00 00 00 00 06 01 03 00 00 00 01

Description		Size	Instance
MBAP header	Transaction identifier Hi	1	00
	Transaction identifier Lo	1	00
	Protocol identifier	2	00 00
	Length	2	00 06
	Unit identifier	1	01
MODBUS request	Function code	1	03
	Starting address	2	00 00
	Number of registers	2	0x0001

Response version number length:
Example: 00 00 00 00 00 05 01 03 02 00 29
Explanation: The byte length of the version number is 0x29

Description		Size	Instance
MBAP header	Transaction identifier Hi	1	00
	Transaction identifier Lo	1	00
	Protocol identifier	2	00 00
	Length	2	00 05
	Unit identifier	1	01
MODBUS request	Function code	1	03
	Number of bytes of data	1	02
	Data	2	00 29

3.1.2 Read version number

First read the version number data length, then use this length to read the version number;
Because the modbus holding register has a bit width of 16, when the version number length is odd, the high 8 bits of the last register value are valid, and the low 8 bits are filled with 0. The starting address is fixed at 0x01. The number of registers read is calculated by this method: (Version byte number +1) / 2
Request to read version number:
Starting address: 0x00 01
Number of registers: (Version byte number +1) / 2
Example: 00 00 00 00 00 06 01 03 00 01 00 15

Description		Size	Instance
MBAP header	Transaction identifier Hi	1	00
	Transaction identifier Lo	1	00
	Protocol identifier	2	00 00
	Length	2	00 06
	Unit identifier	1	01
MODBUS request	Function code	1	03
	Starting address	2	00 01
	Number of registers	2	0x0015

Response version number:

Example:
0x00 0x00 0x00 0x00 0x00 0x2c 0x01 0x03 0x2a 0x41 0x4d 0x38 0x2d 0x51 0x43 0x2d 0x52 0x58 0x45 0x2d 0x37 0x2e 0x38 0x2e 0x30 0x32 0x2d 0x62 0x61 0x74 0x65 0x37 0x5f 0x46 0x41 0x4b 0x45 0x5f 0x54 0x47 0x46 0x5f 0x45 0x4e 0x43 0x4f 0x44 0x45 0x52 0x00
Explanation: After converting each byte in the data area into characters, you can get the version:
“AM8-QC-RXE-7.8.02-bate7_FAKE_TGF_ENCODER”

Description		Size	Instance
MBAP header	Transaction identifier Hi	1	00
	Transaction identifier Lo	1	00
	Protocol identifier	2	00 00
	Length	2	00 2c
	Unit identifier	1	01
MODBUS response	Function code	1	03
	Number of bytes of data	1	2a
	Data	2a	0x41 0x4d 0x38 0x2d 0x51 0x43 0x2d 0x52 0x58 0x45 0x2d 0x37 0x2e 0x38 0x2e 0x30 0x32 0x2d 0x62 0x61 0x74 0x65 0x37 0x5f 0x46 0x41 0x4b 0x45 0x5f 0x54 0x47 0x46 0x5f 0x45 0x4e 0x43 0x4f 0x44 0x45 0x52 0x00

3.1.3 Read counter list

First, read the number of counters, and then request the corresponding counter ID based on the number of counters. Each counter ID occupies 2 registers; since the number of counters might be out of sync with the read quantity, if the actual number of counters exceeds the requested quantity, truncate the requested quantity for the response. If the actual number is less than the requested quantity, fill the rest with 0xFF.
Request counter number:
Read the current number of available counters, occupying only one register, so the starting address and the number of registers are fixed;
Example:
00 00 00 00 00 06 01 03 00 82 00 01

Description		Size	Instance
MBAP header	Transaction identifier Hi	1	00
	Transaction identifier Lo	1	00
	Protocol identifier	2	00 00
	Length	2	00 06
	Unit identifier	1	01
MODBUS request	Function code	1	03
	Starting address	2	00 82
	Number of registers	2	0001

Response counter number:
Example: 00 00 00 00 00 0x05 0x01 0x03 0x02 0x00 0x02
Explanation: The number of counters read is 2;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	0x00 0x02 (Counter number)

Request current counter list:
Address range: 0x0083 to 0x0882, where the valid address is determined by the existing number of counters (0x83 + Number of Registers ×2). The starting address must be the starting address of the target ID, such as 0x0083 is the starting address of the 0th counter ID, and 0x0085 is the starting address of the 1st counter's ID;
Number of read registers: Since 1 counter ID occupies 2 registers, the lower 16 bits are stored in the smaller address, so the requested number of registers needs to be a multiple of 2, such as 2, 4, 6;
Example 1: 00 00 00 00 00 06 01 03 00 83 00 04
Explanation: Request to read 2 consecutive counter IDs starting from the 0th counter.

Description		Size	Instance
MBAP header	Transaction identifier Hi	1	00

	Transaction identifier Lo	1	00
	Protocol identifier	2	00 00
	Length	2	00 06
	Unit identifier	1	01
MODBUS request	Function code	1	03
	Starting address	2	00 83
	Number of registers	2	0004

(1)Example 2: 00 00 00 00 00 06 01 03 00 85 00 04
(2)Explanation: Request to read 2 consecutive counter IDs starting from the 1st counter.
(3)Response for the current list of counters:
(4)Example: 00 00 00 00 00 0b 01 03 08 00 00 00 00 00 00 00 01
(5)Explanation: 2 counter IDs were read, namely 0 and 1;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0b
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	08
Data	8	00 00 00 00 (0th counter ID) 00 00 00 01 (1st counter ID)

3.1.4 Retrieve counter information

To read the counter, you must first write the counter ID you wish to read. Request to write the counter ID to be read:
Starting address: 0x0883
Number of registers: 2,
Byte count: 4;
Value: Counter IDs already on the list, with the high 16 bits first, followed by the low 16 bits;
Example: 00 00 00 00 00 0B 01 10 08 83 00 02 04 00 00 00 01
Explanation: Set the counter ID to be read to 1. After setting successfully, you can use the read function code to retrieve the data of this counter.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0b
Unit identifier	1	01
MODBUS request function code	1	10
Starting address	2	08 83
Number of registers	2	00 02
Number of bytes of data	1	04
Data	4	00 00 00 01

Response to writing the counter ID to be read:
Example: 00 00 00 00 00 06 01 10 08 83 00 02

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	08 83
Number of registers	2	00 02

Request counter status data:
Address: 0x0883
Number of registers: 0x06
Example: 00 00 00 00 00 06 01 03 08 83 00 06

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00

Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	08 83
Number of registers	2	00 06

Response counter status data:
Example: 00 00 00 00 00 0f 01 03 0c 00 00 00 01 00 00 00 0a 00 00 00 05
Explanation: The retrieved counter with ID 1 has a target count value of 0x0a and a current count value of 0x05.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0f
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	0c
Data	12	00 00 00 01 (ID) 00 00 00 0a (target value) 00 00 00 05 (current value)

3.1.5 Retrieve current mode

Request the current operation mode:
Address: 0x0889
Number of registers: 1
Example: 00 00 00 00 00 06 01 03 08 89 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	08 89
Number of registers	2	00 01

Response to the current operation mode:
Example: 00 00 00 00 00 05 01 03 02 00 03
Explanation: Currently in configuration mode.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	00 03

3.1.6 IO board operation

The system currently supports up to 5 IO boards. Request to read the number of IO boards:
Starting address: 0x088a
Number of registers: 1
Example: 00 00 00 00 00 06 01 03 08 8a 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	08 8a
Number of registers	2	00 01

Response to the number of IO boards read:

Example: 00 00 00 00 00 05 01 03 02 00 01
Explanation: Currently, one IO board is in use.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	2	02
Data	2	00 01

1. Request to read IO board input status:

(1)Starting address ranges from 0x088b (2187) to 0x0894 (2196). The valid address is determined by the number of IO boards in use.
(2)Each IO board's input status occupies 2 registers, starting from the board's lower 16 inputs; for instance, 0x088b (2187) is the starting address for the input signals of the 0th board,
(3)0x088d (2189) is for the 1st board's input signals, and so on.
(4)Number of registers: The number of registers to be read must be in multiples of 2.
(5)Example: 00 00 00 00 00 06 01 03 08 8b 00 02
(6)Explanation: Read the input status of block 0.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	08 8b
Number of registers	2	00 01

(7)Example 2: 00 00 00 00 00 06 01 03 08 8b 00 04
(8)Explanation: Read the input status of blocks 0 and 1.
(9)Example 3: 00 00 00 00 00 06 01 03 08 8d 00 02
(10)Explanation: Read the input status of block 1;
(11)Response to reading IO board input status:
(12)Example: 00 00 00 00 00 07 01 03 04 00 00 00 00
(13)Explanation: The 32-bit input status of IO board block 0 is read as 00 00 00 00.
(14)Data arrangement corresponds to the IO board. For received data: 04 03 02 01, 04 corresponds to X40-X47, 03 corresponds to X30-X37, 02 corresponds to X20-X27, and 01 corresponds to X10-X17.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 07
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	2	04
Data	4	00 00 00 01

2. Request to read IO board output status:

(1)Starting address: 0x8b3 (2227) to 0x08bc (2236). Adjust according to the number of IO boards in use. The valid address is determined by the IO board in use.
(2)Register count: Each board input has 32 bits, occupying 2 bytes, so the register count needs to be a multiple of 2;
(3)Data arrangement corresponds to the IO board. For received data: 04 03 02 01, 04 corresponds to X40-X47, 03 corresponds to X30-X37, 02 corresponds to X20-X27, and 01 corresponds to X10-X17.
(4)Example: 00 00 00 00 00 06 01 03 08 b3 00 02
(5)Explanation: Read the output status of block 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00

Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	08 b3
Number of registers	2	00 02

Response to reading IO board output status:
Example: 00 00 00 00 00 07 01 03 04 00 00 00 00
Explanation: The 32-bit output status of IO board block 0 is read as 00 00 00 00;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 07
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	04
Data	4	00 00 00 00

3.1.7 Axis count reading

Request to read the axis count:
Address: 0X08db(2267)
Register count: 0x01
Example: 00 00 00 00 00 06 01 03 08 db 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	08 db
Number of registers	2	00 01

Response to reading IO board output status:
Example: 00 00 00 00 00 05 01 03 02 00 06
Explanation: The current number of axes in use is read as 6 axes;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	4	00 06(Axis count)

3.1.8 Axis position

The current system supports up to 8 axes. The position data precision is 3 decimal places, in units of degrees (e.g., 1000 is equivalent to 1°). Each position occupies
2 registers; Request: Read axis position
Address: 0x08dc to 0x08eb (2283). The actual valid address is determined by the number of axes in use (0x08dc + axis count ×2). The starting address must be the starting address of an axis, e.g., 0x08dc is the starting position of axis 0, and 0x08de is the starting position of axis 1. Register count: Each axis position occupies 2 registers, so the read registers need to be multiples of 2, such as 2, 4.
Example: 00 00 00 00 00 06 01 03 08 dc 00 02
Explanation: Read the position of axis 0.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03

Starting address	2	08 dc
Number of registers	2	00 02

(1)Example 2: 00 00 00 00 00 06 01 03 08 dc 00 04
(2)Explanation: Read the position of axis 0 and axis 1.
(3)Example 3: 00 00 00 00 00 06 01 03 08 de 00 02
(4)Explanation: Read the position of axis 1; Response to reading axis position:
(5)Example: 00 00 00 00 00 07 01 03 04 00 00 00 00
(6)Explanation: The position of axis 0 is read as 0.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 07
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	04
Data	4	00 00 00 00(Axis position)

3.1.9 World coordinate position

The current system supports up to 8 world coordinate positions. The value precision is 3 decimal places, in units of mm (e.g., 1000 is equivalent to 1mm). Each value occupies 2 registers;
World coordinate axis names defined: 0:X, 1:Y, 2:Z, 3:U, 4:V, 5:W, 6:M7, 7:M8. Request to read world coordinate position.
Address: 0x091c~0x092b (2347). The starting address must be the starting address of a specific world axis, such as 0x091c for the starting address of world axis 0 (X), and 0x091e for the starting address of world axis 1 (Y).
Number of registers: Each axis position occupies 2 registers, so the number of registers read should be a multiple of 2, such as 2, 4, 6;
Example: 00 00 00 00 00 06 01 03 09 1c 00 02
Explanation: Read the position of world axis 0.

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 1c
Number of registers	2	00 02

(1)Example 2: 00 00 00 00 00 06 01 03 09 1c 00 04
(2)Explanation: Read the world position of axis 0 and axis 1.
(3)Example 3: 00 00 00 00 00 06 01 03 09 1e 00 02
(4)Explanation: Read the world position of axis 1. Response to read world axis position:
(5)Example: 00 00 00 00 00 07 01 03 04 00 00 00 00
(6)Explanation: The world position of axis 0 is 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 07
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	04
Data	4	00 00 00 00(Axis position)

3.1.10 Retrieve current alarm number

Request to read alarm number:
Each alarm number occupies one register, so the number of registers is fixed at 1, with an address of 0x095c;
Example: 00 00 00 00 00 06 01 03 09 5c 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06

Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 5c
Number of registers	2	00 01

Response to read alarm number:
Example: 00 00 00 00 00 05 01 03 02 03 25
Explanation: The current alarm number is 0x0325;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	03 25 (Alarm)

3.1.11 Period

The unit of period time is ms, one period time occupies 4 registers, so the number of registers is fixed at 4. Request current period time:
Starting address: 0X95d(2397);
Number of registers: 4
Example: 00 00 00 00 00 06 01 03 09 5d 00 04

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 5d
Number of registers	2	00 04

Response to current period time:
Example: 00 00 00 00 00 0b 01 03 08 00 00 00 00 00 00 00 00
Explanation: The current period time is 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0b
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	08
Data	8	00 00 00 00 00 00 00 00 (Current period time)

Request for the last period time:
Starting address: 0X0961(2401);
Number of registers: 4
Example: 00 00 00 00 00 06 01 03 09 61 00 04

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 61
Number of registers	2	00 04

Response to last period time:
Example: 00 00 00 00 00 0b 01 03 08 00 00 00 00 00 00 00 00
Explanation: The last period time is 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00

Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0b
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	08
Data	8	00 00 00 00 00 00 00 00 (Last period time)

3.1.12 Host address

Request to read host address:
Address: 0X0965(2405)
Number of registers: 1
Example: 00 00 00 00 00 06 01 03 09 65 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 65
Number of registers	2	00 01

Response to read host address:
Example: 00 00 00 00 00 05 01 03 02 00 01 Explanation: Host address is 0x01;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	00 01 (Host address)

3.1.13 Read current torque

2580 represents one-time torque, each axis's torque occupies 1 register. Request to read torque:
Address range: 0X966 (2406)~0x096d (2413)
The number of registers: The sum with the starting address should not exceed the address range;
Example: 00 00 00 00 00 06 01 03 09 66 00 01 Explanation: Read the current torque data of axis 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 66
Number of registers	2	00 01

Response to read torque:
Example: 00 00 00 00 00 05 01 03 02 00 00
Explanation: Axis 0 current torque is 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	00 00 (Torque)

3.1.14 Read current speed of the axis

Speed unit RPM, each axis speed occupies 1 register; Request to read speed:
Address range: 0x0986 (2438) to 0x098d (2445)

The number of registers: The sum with the starting address should not exceed the address range;
Example: 00 00 00 00 00 06 01 03 09 86 00 01
Explanation: Read the current speed data of axis 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 86
Number of registers	2	00 01

Respond to speed reading:
Example: 00 00 00 00 00 05 01 03 02 00 00
Explanation: Axis 0 current speed is 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	00 00 (Speed)

3.1.15 Movement status

Request to read movement status:
Address: 0X09a6 (2470)
Number of registers: 1
Movement status value definition: Movement status; 0: Stop; 1: Move;
Example: 00 00 00 00 00 06 01 03 09 a6 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	03
Starting address	2	09 a6
Number of registers	2	00 01

Respond to movement status reading:
Example: 00 00 00 00 00 05 01 03 02 00 01
Explanation: Currently in movement status;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS response function code	1	03
Number of bytes of data	1	02
Data	2	00 01

3.2 Write to a single register (0x06)

3.2.1 Command

Each command occupies one register, writing any valid value (0-0xFFFF) will execute the corresponding command;
Command operation only supports function code 0x06.
Request to stop the current action:
Address: 4e 20
Value: Any
Example: 00 00 00 00 00 06 01 06 4e 20 00 01
Explanation: Write 1 to the register to stop the current action;

Description	Size	Instance

MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	06
Starting address	2	4e 20
Number of registers	2	00 01

The response frame is the same as the request frame; see the address table for other commands;

3.2.2 Modify global speed

Global speed unit is %, precision to 1 decimal place, i.e., valid values are 0-1000 corresponding to 0.0% to 100.0%; This value can also be read with function code 0x03;

.

Request to modify global speed:
Address: 0x4ee8
Value: 0-1000
Example: 00 00 00 00 00 06 01 06 4e e8 00 64
Explanation: Set global speed to 0x64 (i.e. 10.0%);

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	06
Starting address	2	4e e8
Number of registers	2	00 64

Response is the same as request;

3.3 Write to multiple registers (0x10)

3.3.1 Write to entire IO board output

The current system supports up to 5 IO boards, each IO board output has 32 bits, occupying 2 registers; Request IO board output:
Function code: 0x10
Address: 0X08b3 (2227) to 0x08bc (2236), the starting address must be the starting address of some IO board output, for example, 0x08b3, 0x08b5 are correct, while 0x08b4 is wrong;
Register count: Since each IO board occupies 2 registers, it needs to be a multiple of 2;
Register count: Since each IO board occupies 2 registers, it needs to be a multiple of 2;
Example: 00 00 00 00 00 0B 01 10 08 b3 00 02 04 00 00 00 01
Explanation: Set the output of IO board 0 at port 0;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0b
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	08 b3
Number of registers	2	00 02
Number of bytes of data	1	04
Data	4	00 00 00 01

Respond to IO board output:
Example: 00 00 00 00 00 06 01 10 08 b3 00 02

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	10
Starting address	2	08 b3
Number of registers	2	00 02

3.3.2 Modify single output point status

Used to modify the output status of a point on a particular IO board, the parameters to set include specifying the IO board (starting from 0), specifying the output point (0-31), and specifying the status (0: off, 1: on); Each of the 3 parameters occupies one register;
IO board value range: 0-3 regular IO; 4-6: M value; 7: EUIO;
Output Range: 0-31
Status value definition: 0: Off, 1: On
Request to modify individual output point status:
Function code: 0x10
Starting address: 0x4e58
Number of registers: 3
Example: 00 00 00 00 00 0d 01 10 4e 58 00 03 06 00 00 00 01 00 01
Explanation: Set the output port 1 of block 0 to 'On';

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0d
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	4e 58
Number of registers	2	00 03
Number of bytes of data	1	06
Data	6	00 00 00 01 00 01

Respond to modification of individual output point status:
Example: 00 00 00 00 00 06 01 10 4e 58 00 03

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	10
Starting address	2	4e 58
Number of registers	2	00 03

3.3.3 Modify counter

Parameters include: ID, target value, and current value. There are 3 in total, each occupying 2 registers, so the counter modification has a fixed register count of 6;
ID: Used to specify the counter to be modified.
Target value: Cannot be modified on the host side, so there are no requirements for this value;
Current value: Can be modified; Request to modify counter:
Function code: 0x10
Starting address: 4e 52
Number of registers: 6
Example: 00 00 00 00 00 13 01 10 4e 52 00 06 0c 00 00 00 01 00 00 00 00 00 00 00 08
Explanation: The counter with ID 1 needs to be modified to a current value of 8;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 13
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	4e 52
Number of registers	2	00 06
Number of bytes of data	1	0c
Data	12	00 00 00 01 00 00 00 00 00 00 00 08

Response to counter modification:
Example: 00 00 00 00 00 06 01 10 4e 52 00 06

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00

Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	10
Starting address	2	4e 52
Number of registers	2	00 06

3.3.4 Modify address parameters

The host system parameter table uses a total of 1000 values, each value occupies 2 registers, with the high 16bit located at the lower address; This block of address can also be read using function code 0x03; Request to modify address parameters:

(1)Address: 0x4f4c (memory address 20300) to 0x56d5 (memory address 22229), open for use addresses: 558C (memory address 21900)~

(2)Number of registers: Must be a multiple of 2;

(3)Example 1: 00 00 00 00 00 0b 01 10 4f 4c 00 02 04 00 00 00 64

Explanation: Set the value of parameter 0 to 100;

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0b
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	4f 4c
Number of registers	2	00 02
Number of bytes of data	1	04
Data	4	00 00 00 64

Respond to address parameter modification:

Example 1: 00 00 00 00 00 06 01 10 4f 4c 00 02

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	10
Starting address	2	4f 4c
Number of registers	2	00 02

Example 2: 00 00 00 00 00 0b 01 10 4f 4e 00 02 04 00 00 00 64

Explanation: Set the value of parameter 1 to 100;

Example 3: 00 00 00 00 00 0f 01 10 4f 4c 00 04 08 00 00 00 64 00 00 00 C8

Explanation: Set the value of parameter 0 to 100, and the value of parameter 1 to 200;

3.3.5 Transmitting position data

Within the system, there can be multiple sets of irregular position data, identified by ID. The stacked ID is the data source ID, with the visual data source fixed at 100. You can manage the data in the data source using the following process.

1. First, configure the data information of the data source to be modified, and send the following instruction:

Request:

(1)Address: 0x7532 (30002)~0x7534 (30004)

(2)Number of registers: 3

(3)Example 1: 00 00 00 00 00 0b 01 10 75 32 00 03 06 00 64 00 06 00 23

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 0d
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	75 32
Number of registers	2	00 03
Number of bytes of data	1	06
Data	6	00 64 00 06 00 23

Explanation: Set the data source ID to be modified to 100, each point contains 6 pieces of data, using only data from axes 1,2,6. This is the typical data configuration used for planar vision;

Response: 00 00 00 00 00 06 01 10 4f 4c 00 02

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00

Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	10
Starting address	2	75 32
Number of registers	2	00 03

2. Transmit position data, in cases with fewer positions (data is less than 256 bytes):

(1)Address: 0x7535 (30005)~0x9C40 (40000)

(2)Register count: Calculated based on position number and format, 2 registers per axis 1 data

(3)Example 1: 00 00 00 00 00 3b 01 10 75 35 00 1A 34 00 02 00 01 00 00 00 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00 05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00 09 00 00 00 0A 00 00 00 0B 00 00 00 0C

Explanation: Clear the original data source data and add 2 points

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 3b
Unit identifier	1	01
MODBUS response function code	1	10
Starting address	2	75 32
Number of registers	26	00 1a
Number of bytes of data	1	34
Data	52	00 02 00 01 00 00 00 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00 05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00 09 00 00 00 0A 00 00 00 0B 00 00 00 0C

3. Transmit position data, in cases with many positions (single frame data exceeding 256 bytes):

The data can be split for transmission. The first frame clears the original data source (0x7536 set to 1), subsequent frames do not clear (0x7536 set to 0). You can also opt to send in a single frame; this would be a non-standard protocol, and our system internally will ignore the data byte count field

3.4 Read/Write bit (0x01, 0x05)

Example: **0x05**

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS response function code	1	05
Starting address	2	00 00
Number of registers	2	FF 03 ON or 00 00 OFF
Register write 0xFF 00 ON, 0x00 00 OFF		

Response is the same as the transmission:

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS response function code	1	05
Starting address	2	00 00
Number of registers	2	FF 03 00 00
Register write 0xFF 00 ON, 0x00 00 OFF		

Example: **0x01 00 00 00 00 00 06 01 01 00 a0 00 0A**

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS request function code	1	01
Starting address	2	00 A0
Number of registers	2	00 0A Start at the 10th count

Response: 00 00 00 00 00 05 01 01 02 FD 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00

Protocol identifier	2	00 00
Length	2	00 05
Unit identifier	1	01
MODBUS request function code	1	01
Starting address	2	02
Number of registers	2	FD 01 Data expands based on 8/16/32 bit high/low (low bit as the starting bit) corresponding to reading IO status Normally read 1 bit, which is one byte, 0 OFF or 1 ON

Example:

M010 output ON 00 00 00 00 00 06 01 05 00 A0 FF 00 Response is the same
M010 status read 00 00 00 00 00 06 01 01 00 A0 00 01 Read 1 bit, confirm tail 0 OFF 1 ON
Y010 output ON 00 00 00 00 00 06 01 05 00 00 FF 00 Response is the same
Y010 status read 00 00 00 00 00 06 01 01 00 00 00 01 Read 1 bit, confirm tail 0 OFF 1 ON

4 Address table definition function code: 0x03 0x06 0x10

Feature description	Decimal address	Hexadecimal	Description	R/W	Maximum	Minimum	Remarks
Read version number length	0	0	Version number length	R	1	64	
Read version number content	1	1	Start of version number string	R	/	/	String, maximum length of 128 content bytes
Read version number content		0		R	/	/	
Read version number content	64	40	Start of version number string	R	/	/	
Read current model number length	65	41	Current model length	R	1	64	Main network port not supported
Read current model number content	66	42	Current model number lower byte	R	/	/	String
Read current model number content	……			R	/	/	
Read current model number content	129	81	Current model number upper byte	R	/	/	
Read counter list count	130	82	Counter count	R	0	1024	
Read counter list content	131	83	High byte of 0th counter's ID	R	/	/	The number of valid addresses is determined by the counter count; each ID occupies 32 bits;
Read counter list content	132	84	Low byte of 0th counter's ID	R	/	/	
Read counter list content	133	85	High byte of 1st counter's ID	R	/	/	
Read counter list content	134	86	Low byte of 1st counter's ID	R	/	/	
Read counter list content	……			R	/	/	
Read counter list content	2178	882	Low byte of 1023rd counter's ID	W/R	/	/	
Read counter status ID	2179	883	High byte of counter to be read ID	W/R	/	/	Write the counter ID to be read first, then read that segment of data.
Read counter status	2181	885	Counter target value high byte	R	/	/	
Read counter status	2182	886	Counter current value low byte	R	/	/	
Read counter status	2183	887	Counter current value high byte	R	/	/	
Read counter status	2184	888	Counter current value low byte	R	/	/	
Read counter status	2185	889	Current mode	R	/	/	Status, 1 is manual, 2 is automatic.
Total number of IO boards	2186	88A	Total number of IO boards	R	0	5	At most 4 are currently used, the 5th is not used.
Board 0 input status	2187	88B	Input status of board 0's inputs 16–31	R	/	/	
Board 0 input status	2188	88C	Input status of board 0, pins 0–15	R	/	/	
Board 1 input status	2189	88D	Input status of board 1, pins 16–31	R	/	/	
Board 1 input status	2190	88E	Input status of board 1, pins 0–15	R	/	/	
Board 2 input status	2191	88F	Input status of board 2, pins 16–31	R	/	/	
Board 2 input status	2192	890	Input status of board 2, pins 0–15	R	/	/	
Board 3 input status	2193	891	Input status of board 3, pins 16–31	R	/	/	
Board 3 input status	2194	892	Input status of board 3, pins 0–15	R	/	/	
Board 4 input status	2195	893	Input status of board 4, pins 16–31	R	/	/	
Board 4 input status	2196	894	Input status of board 4, pins 0–15	R	/	/	

ReservedReserved	……		ReservedReserved	R	/	/	
ReservedReserved	2226	8B2	ReservedReserved	R	/	/	
Board 0 output status	2227	8B3	Output status of board 0, pins 16–31	R/W	/	/	
Board 0 output status	2228	8B4	Output status of board 0, pins 0–15	R/W	/	/	
Board 1 output status	2229	8B5	Output status of board 1, pins 16–31	R/W	/	/	
Board 1 output status	2230	8B6	Output status of board 1, pins 0–15	R/W	/	/	
Board 2 output status	2231	8B7	Output status of board 2, pins 16–31	R/W	/	/	
Board 2 output status	2232	8B8	Output status of board 2, pins 0–15	R/W	/	/	
Output status of board 3	2233	8B9	Output status of board 3, pins 16–31	R/W	/	/	
Output status of board 3	2234	8BA	Output status of board 3, pins 0–15	R/W	/	/	
Output status of board 4	2235	8BB	Output status of board 4, pins 16–31	R/W	/	/	
Output status of board 4	2236	8BC	Output status of board 4, pins 0–15	R/W	/	/	
ReservedReserved	……		ReservedReserved	R/W	/	/	
ReservedReserved	2266	8DA	ReservedReserved	R/W	/	/	
Total number of axes	2267	8DB	Total number of axes	R	0	8	Currently supports up to 8 axes
Position of axis 1	2268	8DC	High byte of axis 0 (J1)	R	/	/	The original value is a double, with a precision of 3 decimal places; it is magnified 1000 times and converted to an integer
Position of axis 1	2269	8DD	Low byte of axis 0 (J1)	R	/	/	
Position of axis 2	2270	8DE	High byte of axis 1 (J2)	R	/	/	
Position of axis 2	2271	8DF	Low byte of axis 1 (J2)	R	/	/	
Position of axis 3	2272	8E0	High byte of axis 2 (J3)	R	/	/	
Position of axis 3	2273	8E1	Low byte of axis 2 (J3)	R	/	/	
Position of axis 4	2274	8E2	High byte of axis 3 (J4)	R	/	/	
Position of axis 4	2275	8E3	Low byte of axis 3 (J4)	R	/	/	
Position of axis 5	2276	8E4	High byte of axis 4 (J5)	R	/	/	
Position of axis 5	2277	8E5	Low byte of axis 4 (J5)	R	/	/	
Position of axis 6	2278	8E6	High byte of axis 5 (J6)	R	/	/	
Position of axis 6	2279	8E7	Low byte of axis 5 (J6)	R	/	/	
Position of axis 7	2280	8E8	High byte of axis 6 (J7)	R	/	/	
Position of axis 7	2281	8E9	Low byte of axis 6 (J7)	R	/	/	
Position of axis 8	2282	8EA	High byte of axis 7 (J8)	R	/	/	
Position of axis 8	2283	8EB	Low byte of axis 7 (J8)	R	/	/	
ReservedReserved	……		ReservedReserved	R	/	/	
ReservedReserved	2331	91B	ReservedReserved	R	/	/	

World position of X-axis	2332	91C	High byte of world coordinate axis 0 (X)	R	/	/	The original value is a double, with a precision of 3 decimal places; it is magnified 1000 times and converted to an integer
World position of X-axis	2333	91D	Low byte of world coordinate axis 0 (X)	R	/	/	
World position of Y-axis	2334	91E	High byte of world coordinate axis 1 (Y)	R	/	/	
World position of Y-axis	2335	91F	Low byte of world coordinate axis 1 (Y)	R	/	/	
World position of Z-axis	2336	920	High byte of world coordinate axis 2 (Z)	R	/	/	
World position of Z-axis	2337	921	Low byte of world coordinate axis 2 (Z)	R	/	/	
World position of U-axis	2338	922	High byte of world coordinate axis 3 (U)	R	/	/	
World position of U-axis	2339	923	Low byte of world coordinate axis 3 (U)	R	/	/	
World position of V-axis	2340	924	High byte of world coordinate axis 4 (V)	R	/	/	
World position of V-axis	2341	925	Low byte of world coordinate axis 4 (V)	R	/	/	
World position of W-axis	2342	926	High byte of world coordinate axis 5 (W)	R	/	/	
World position of W-axis	2343	927	Low byte of world coordinate axis 5 (W)	R	/	/	
World position of M7-axis	2344	928	High byte of world coordinate axis 6 (M7)	R	/	/	
World position of M7-axis	2345	929	Low byte of world coordinate axis 6 (M7)	R	/	/	
World position of M8-axis	2346	92A	High byte of world coordinate axis 7 (M8)	R	/	/	
World position of M8-axis	2347	92B	Low byte of world coordinate axis 7 (M8)	R	/	/	
ReservedReserved	……		ReservedReserved	R	/	/	
ReservedReserved	2395	95B	ReservedReserved	R	/	/	
Alarm code	2396	95C	Current alarm code	R	/	/	
Current cycle	2397	95D	Current cycle 48-63 bits	R	/	/	64-bit data; high byte first, low byte last.
Current cycle	2398	95E	Current cycle 32-47 bits	R	/	/	
Current cycle	2399	95F	Current cycle 16-31 bits	R	/	/	
Current cycle	2400	960	Current cycle 0-15 bits R	R	/	/	
Previous cycle	2401	961	Previous cycle 48-63 bits	R	/	/	
Previous cycle	2402	962	Previous cycle 32-47 bits	R	/	/	
Previous cycle	2403	963	Previous cycle 16-31 bits	R	/	/	
Previous cycle	2404	964	Previous cycle 0-15 bits R	R	/	/	
Machine name	2405	965	Machine name (Host ID)	R	/	/	
Axis 1 torque	2406	966	Axis 0 current torque	R	/	/	
Axis 2 torque	2407	967	Axis 1 current torque	R	/	/	
Axis 3 torque	2408	968	Axis 2 current torque	R	/	/	
Axis 4 torque	2409	969	Axis 3 current torque	R	/	/	
Axis 5 torque	2410	96A	Axis 4 current torque	R	/	/	
Axis 6 torque	2411	96B	Axis 7 torque	R	/	/	

Axis 7 torque	2412	96C	Axis 6 current torque	R	/	/	
Axis 8 torque	2413	96D	Axis 7 current torque	R	/	/	
ReservedReserved	·····		ReservedReserved	R	/	/	
ReservedReserved	2437	985	ReservedReserved	R	/	/	
Axis 1 speed	2438	986	Axis 0 current speed	R	/	/	
Axis 2 speed	2439	987	Axis 1 current speed	R	/	/	
Axis 3 speed	2440	988	Axis 2 current speed	R	/	/	
Axis 4 speed	2441	989	Axis 3 current speed	R	/	/	
Axis 5 speed	2442	98A	Axis 4 current speed	R	/	/	
Axis 6 speed	2443	98B	Axis 5 current speed	R	/	/	
Axis 7 speed	2444	98C	Axis 6 current speed	R	/	/	
Axis 8 speed	2445	98D	Axis 7 current speed	R	/	/	
ReservedReserved	·····		ReservedReserved	R	/	/	
ReservedReserved	2469	9A5	ReservedReserved	R	/	/	
Movement status	2470	9A6	Current machine movement status	R	0	1	1 for moving, 0 for stationary
Home status	2471	9A7	Home status	R	0	1	Once all axes are set to home, this status is 1, otherwise 0.
Current user	2472	9A8	Current user length	R	1	64	The host does not have this item
Current user	2473	9A9	Current user low byte	R	/	/	The host does not have this item
Current user	·····		·····	R	/	/	
Current user	2536	9E8	Current user high byte	R	/	/	The host does not have this item
Read total mode number	2537	9E9	Mode number quantity	R	/	/	The host does not have this item
Mode number list	·····		Mode number list	R	/	/	The host does not have this item
ReservedReserved	·····		ReservedReserved	R	/	/	
Maximum speed of shaft 1	2472	9A8	Maximum speed of shaft J1	R	1	64	
Maximum speed of shaft 2	2473	9A9	Maximum speed of shaft J2	R	/	/	
Maximum speed of shaft 3	2474	9AA	Maximum speed of shaft J3	R			
Maximum speed of shaft 4	2475	9AB	Maximum speed of shaft J4	R			
Maximum speed of shaft 5	2476	9AC	Maximum speed of shaft J5	R			
Maximum speed of shaft 6	2477	9AD	Maximum speed of shaft J6	R			
Maximum speed of shaft 7	2478	9AE	Maximum speed of shaft J7	R			
Maximum speed of shaft 8	2479	9AF	Maximum speed of shaft J8	R			
Shaft 1 bus voltage	2504	9C8	Shaft J1 bus voltage	R	/	/	
Shaft 2 bus voltage	2505	9C9	Shaft J2 bus voltage	R	/	/	

Shaft 3 bus voltage	2506	9CA	Shaft J3 bus voltage	R	/	/	
Shaft 4 bus voltage	2507	9CB	Shaft J4 bus voltage	R	/	/	
Shaft 5 bus voltage	2508	9CC	Shaft J5 bus voltage	R	/	/	
Shaft 6 bus voltage	2509	9CD	Shaft J6 bus voltage	R	/	/	
Shaft 7 bus voltage	2510	9CE	Shaft J7 bus voltage	R	/	/	
Shaft 8 bus voltage	2511	9CF	Shaft J8 bus voltage	R	/	/	
Load rate of shaft 1	2536	9E8	Load rate of shaft J1	R	/	/	
Load rate of shaft 2	2537	9E9	Load rate of shaft J2	R	/	/	
Load rate of shaft 3	2538	9EA	Load rate of shaft J3	R	/	/	
Load rate of shaft 4	2539	9EB	Load rate of shaft J4	R	/	/	
Load rate of shaft 5	2540	9EC	Load rate of shaft J5	R	/	/	
Load rate of shaft 6	2541	9ED	Load rate of shaft J6	R	/	/	
Load rate of shaft 7	2542	9EE	Load rate of shaft J7	R	/	/	
Load rate of shaft 8	2543	9EF	Load rate of shaft J8	R	/	/	
Axis 1 temperature	2568	A08	Axis J1 temperature				
Axis 2 temperature	2569	A09	Axis J2 temperature				
Axis 3 temperature	2570	A0A	Axis J3 temperature				
Axis 4 temperature	2571	A0B	Axis J4 temperature				
Axis 5 temperature	2572	A0C	Axis J5 temperature				
Axis 6 temperature	2573	A0D	Axis J6 temperature				
Axis 7 temperature	2574	A0E	Axis J7 temperature				
Axis 8 temperature	2575	A0F	Axis J8 temperature				
Axis 1 position	2600	A28	J1 position high position	R	/	/	This address is a floating-point number with a precision of 6 decimal places.
	2601	A29	J1 position low	R	/	/	
Axis 2 position	2602	A2A	J2 position high position	R	/	/	
	2603	A2B	J2 position low	R	/	/	
Axis 3 position	2604	A2C	J3 position high position	R	/	/	
	2605	A2D	J3 position low	R	/	/	
Axis 4 position	2606	A2E	J4 position high position	R	/	/	
	2607	A2F	J4 position low	R	/	/	
Axis 5 position	2608	A30	J5 position high position	R	/	/	
	2609	A31	J5 position low	R	/	/	
Axis 6 position	2610	A32	J6 position high position	R	/	/	

	2611	A33	J6 position low	R	/	/	
Axis 7 position	2612	A34	J7 position high position	R	/	/	
	2613	A35	J7 position low	R	/	/	
Axis 8 position	2614	A36	J8 position high position	R	/	/	
	2615	A37	J8 position low	R	/	/	
X-axis world position	2664	A68	X-axis world position high position	R	/	/	This address is a floating-point number with a precision of 6 decimal places.
	2665	A69	X-axis world position low	R	/	/	
Y-axis world position	2666	A6A	Y-axis world position high position	R	/	/	
	2667	A6B	Y-axis world position low	R	/	/	
Z-axis world position	2668	A6C	Z-axis world position high position	R	/	/	
	2669	A6D	Z-axis world position low	R	/	/	
U-axis world position	2670	A6E	U-axis world position high position	R	/	/	
	2671	A6F	U-axis world position low	R	/	/	
V-axis world position	2672	A70	V-axis world position high position	R	/	/	
	2673	A71	V-axis world position low	R	/	/	
W-axis world position	2674	A72	W-axis world position high position	R	/	/	
	2675	A73	W-axis world position low	R	/	/	
M7-axis world position	2676	A74	M6-axis world position high position	R	/	/	
	2677	A75	M6-axis world position low	R	/	/	
M8-axis world position	2678	A76	M7-axis world position high position	R	/	/	
	2679	A77	M7-axis world position low	R	/	/	
J1 axis servo version number	2728	AA8	J1 axis servo version number(99)	R			This address is only available for version HCRoboHost-HC-QC-RX-7.8.07-F5-Beta 55 and above.
J2 axis servo version number	2729	AA9	J2 axis servo version number(99)	R			
J3 axis servo version number	2730	AAA	J3 axis servo version number(99)	R			
J4 axis servo version number	2731	AAB	J4 axis servo version number(99)	R			
J5 axis servo version number	2732	AAC	J5 axis servo version number(99)	R			
J6 axis servo version number	2733	AAD	J6 axis servo version number(99)	R			
J7 axis servo version number	2734	AAE	J7 axis servo version number(99)	R			
J8 axis servo version number	2735	AAF	J8 axis servo version number(99)	R			
J1 Axis Servo Subversion Number	2760	AC8	J1 Axis Servo Subversion Number	R			This address is only available for version HCRoboHost-HC-QC-RX-7.8.07-F5-Beta 55 and above.
J2 Axis Servo Subversion Number	2761	AC9	J2 Axis Servo Subversion Number	R			
J3 Axis Servo Subversion Number	2762	ACA	J3 Axis Servo Subversion Number	R			
J4 Axis Servo Subversion Number	2763	ACB	J4 Axis Servo Subversion Number	R			

J5 Axis Servo Subversion Number	2764	ACC	J5 Axis Servo Subversion Number	R			
J6 Axis Servo Subversion Number	2765	ACD	J6 Axis Servo Subversion Number	R			
J7 Axis Servo Subversion Number	2766	ACE	J7 Axis Servo Subversion Number	R			
J8 Axis Servo Subversion Number	2767	ACF	J8 Axis Servo Subversion Number	R			
Current value of counter 0	3526	DC6	high	R/W			Current value of counter 0–100
	3527	DC7	low	R/W			
Current value of counter N		0	high	R/W			
		0	low	R/W			
Current value of counter 100	3726	E8E	high	R/W			
	3727	E8F	low	R/W			
Current value of timer 0	3728	E90	high	R/W			Current value of timer 0–100
	3729	E91	low	R/W			
Current value of timer N		0	high	R/W			
		0	low	R/W			
Current value of timer 100	3928	F58	high	R/W			
	3929	F59	low	R/W			
home address	4500	1194	Allpara[800] remains high after power failure.	RW	/	/	This address is used for power-off saving. This address is only available in version HCRoboHost-HC-QC-RX-7.8.07-F5-Beta 55 and above.
	4501	1195	Allpara[800] remains low after power failure.	RW	/	/	
	4502	1196	Allpara[801] remains high after power failure.	RW	/	/	
	4503	1197	Allpara[801] remains low after power failure.	RW	/	/	
				RW	/	/	
	4696	1258	Allpara[898] remains high after power failure.	R	/	/	
	4697	1259	Allpara[898] remains low after power failure.	R	/	/	
		0					
Command: Cease the current action immediately	20000	4E20	Immediately cease the current action	W	/	/	Immediately halt the current action (restart from the beginning)
Command: Pause the current action	20001	4E21	Pause the current action	W	/	/	Command: Pause the current action (start from the current step)
Command: Enter single loop mode	20002	4E22	Enter single loop mode	W	/	/	Command: Enter single loop mode
Command: Activate button	20003	4E23	startButton: Activate button	W	/	/	Command: Activate button
Command: Stop button press	20004	4E24	stopButton: Stop button	W	/	/	Command: Stop button
Command: Clear the alarm and execute the next instruction	20005	4E25	Clear the alarm and execute the next instruction	W	/	/	Command: Clear the alarm and execute the next instruction

Command: Clear the alarm and resume automatic operation	20006	4E26	Clear the alarm and continue automatic operation	W	/	/	Command: Clear the alarm and resume automatic operation (under automatic operation mode)
ReservedReserved	……		ReservedReserved	W	/	/	
ReservedReserved	20049	4E51	ReservedReserved	W	/	/	
Counter modification	20050	4E52	To be written counter ID high byte	W	/	/	First read the counter list to get the configurable counter ID
Counter modification	20051	4E53	To be written counter ID low byte	W	/	/	
Counter modification	20052	4E54	Set counter target value high byte	W	/	/	
Counter modification	20053	4E55	Set counter target value low byte	W	/	/	
Counter modification	20054	4E56	Set current counter value high byte	W	/	/	Host cannot modify target value
Counter modification	20055	4E57	Set current counter value low byte	W	/	/	
Output point control	20056	4E58	Set output board ID	W	0	7	(0–3: IO board, 4–6: M value (manual control network version not supported temporarily), 7: EUIO)
Output point control	20057	4E59	Set output point	W	0	31	（0～31）
Output point control	20058	4E5A	Set output status	W	0	1	（0：OFF，1：ON）
Stack modification	20059	4E5B	Stack ID to be modified	W	/	/	
Stack modification	20060	4E5C	X interval low byte	W	/	/	Data width?? Host lacks this feature
Stack modification	20061	4E5D	X interval high byte	W	/	/	
Stack modification	20062	4E5E	Y interval low byte	W	/	/	
Stack modification	20063	4E5F	Y interval high byte	W	/	/	
Stack modification	20064	4E60	Z interval low byte	W	/	/	
Stack modification	20065	4E61	Z interval high byte	W	/	/	
Stack modification	20066	4E62	X count low byte	W	/	/	
Stack modification	20067	4E63	X count high byte	W	/	/	
Stack modification	20068	4E64	Y count low byte	W	/	/	
Stack modification	20069	4E65	Y count high byte	W	/	/	
Stack modification	20070	4E66	Z count low byte	W	/	/	
Stack modification	20071	4E67	Z count high byte	W	/	/	
ReservedReserved	……		ReservedReserved	W	/	/	
Global speed	20200	4EE8	Global speed	RW	0	1000	32-bit precision 1
ReservedReserved	……		ReservedReserved	RW	/	/	
Internal parameter table address	20300	4F4C	allpara[0] high byte	RW	/	/	The host function operates 1000 parameters in total, each value is 32 bits, occupying 4 bytes (2 storages) Start address: 4F4C, 4F4D corresponds to storage 0, the rest follow in sequence Address 558C (21900) corresponds to storage 800. Open user address 800–900 is available
Internal parameter table address	20301	4F4D	allpara[0] low byte	RW	/	/	
Internal parameter table address	20302	4F4E	allpara[1] high byte	RW	/	/	
Internal parameter table address	20303	4F4F	allpara[1] low byte	RW	/	/	
Internal parameter table address	……		allpara[900] low byte	R	/	/	
Internal parameter table address	22229	56D5	allpara[999] low byte	R	/	/	

Take a photo command	22130	5672	Take a photo command	R	/	/	The host does not support
Switch mode number	22131	5673	Switch mode number	R	/	/	("d1": Mode number (host Ethernet version temporarily not supported))
		0					
Machine number length	23000	59D8	Machine number length				
Machine number length	23001	59D9					The robot number needs to be set before it can be read.
	23002	59DA					
	23257	5AD9					
		0					
Machine specification length	23300	5B04	Machine specification length				
Machine specification length	23301	5B05					Robot specifications, which need to be set before reading.
	23302	5B06					
	23557	5C05					
Manual version length	23900	5D5C					
Manual control version	23901	5D5D					Manual controller robot system program version
	23902	5D5E					
	24004	5DC4	103				
Data source ID to be viewed	30000	7530	Data source ID to be viewed	R	/	/	Visual data source ID100, others are stack IDs
Buffer length corresponding to data source ID	30001	7531	Buffer length corresponding to the data source ID	R	/	/	
Visual data source ID to be modified	30002	7532	Visual data source ID to be modified	RW	/	/	Visual data source ID100 to be modified, its corresponding stack ID
Data source position format to be modified	30003	7533	Data source position format to be modified, whether 6 data make one point or 4 data make one point	RW	/	/	Default is 6
Data source position mask to be modified	30004	7534	Data source position format mask. Bits 0–5 correspond to Axes 1–6	RW	/	/	If a point has 6 axis positions, but only 1, 2, 6 are valid, then the mask can be set to 0x23
Length of the data point to be received	30005	7535	Length of the data point to be received	RW	/	/	
Clear pending data source data	30006	7536	Clear pending data source data	RW	/	/	1: Clear
Data source position data	30007	7537	Axis 1 data high byte	W	/	/	
Data source position data	30008	7538	Axis 1 data low byte	W	/	/	
Data source position data	……		...	W	/	/	
Data source position data	40000	9C40	Data source position data	W	/	/	

5 function code 0x01 0x05 address table definition

Feature description	Address in decimal	Hexadecimal	Description	Read(r)/Write(w)	Smallest	Largest	Remarks
Y010 —> Y047	0 —> 31	0 —> 1F	Value written to address 05 0xFF00 is ON 0x0000 is OFF Start address for 01 read	r/w	/	/	Y01O corresponds to address 0, Y011 corresponds to address 1, and so on, note the use of function code Use function code 0x01 for reading, 0x05 for writing Write function code 05, tail 0xFF00 ON, 0x0000 OFF Read function code 01, reply tail: 0x00 OFF, 0x01 ON
Y110 —> Y147	32 —> 63	20 —> 3F					
Y210 —> Y247	64 —> 95	40 —> 5F					
Y310 —> Y347	96 —> 127	60 —> 8F					
Y410 —> Y447	128 —> 159	80 —> 9F					
M010 —> M047	160 —> 191	A0 —> BF					
M110 —> M147	192 —> 223	C0 —> DF					
M210 —> M247	224 —> 255	E0 —> FF					
EUY010 —> EUY147	256 —> 287	100 —> 11F					

Use function code 0x01 for reading, similar to function code 03, where 01 is bit reading unit, and 03 is word reading unit

00 00 00 00 00 06 01 01 00 00 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS response function code	1	01
Starting address	2	00 00
Data	2	00 01 (Reading count, when there are multiple bits, expand to corresponding IO status)

Use function code 0x05 for writing, similar to function code 06, where 05 is bit single write, and 06 is word single write

00 00 00 00 00 06 01 01 00 00 00 01

Description	Size	Instance
MBAP header transaction Identifier Hi	1	00
Transaction identifier Lo	1	00
Protocol identifier	2	00 00
Length	2	00 06
Unit identifier	1	01
MODBUS response function code	1	05
Starting address	2	00 02
Number of registers	2	FF 00 (FF 00 ON, 00 00 OFF)

Outputs Y, M, EUY are in octal and correspond to increasing addresses

伯朗特机器人股份有限公司

地址：广东省东莞市大朗镇沙步村沙富路 83 号

电话：400-870-8989

邮箱：rd316@borunte.com

因改良等原因，规格及外观有所变更时，不另行通知，敬请谅解。因排样需要，某些部件有所增减，请按实际订单为准。

BORUNTE ROBOT CO., LTD.

Address：NO. 83, Shafu Road, Shabu Village, Dalang Town, Dongguan City, Guangdong Province, China

Tel：0086-400-870-8989

E-mail : rd316@borunte.com

All stated here is subject to change without advance notice. Some parts have been increased or deleted because of the shooting needed, please order as the actual standard.

抖音号　　公众号　　小程序

- www.borunte.com -

- 2024.09.11 出版 -