R6 system manual

System Modbus TCP Slave Description

Shenzhen Huacheng Industrial Control Co.

Catalog

1 Preface

This description introduces the Modbus TCP protocol of this system. This protocol is supported simultaneously with the [remote communication protocol] of this system.

1.1 Modification records

Category: A- Add M- Modify D- Delete

serial number	version number	category	modify the content	date	Revised by	audit
3	A2	A	1. Modify the document format	November 9, 2020	Cheng Guoxing	
2	A1	A	1. Modify the IO read and write instructions, 0 to 4 are general IO, 5 to 7 are M values, and 8 is EUIO 2. Add general output point for bitwise operation, M value function and EU output point, function code 0x5	November 4, 2020	Cheng Guoxing	
1	A0	A	1. First Edition Guidelines	November 4, 2020	Cheng Guoxing	

2 Brief description

1. Using the modbusTCP protocol, the host acts as a modbus slave;
2. The data in the example is in hexadecimal;
3. Data Format:

   (1) In bytes, the high byte is in front and the low byte is behind.
   (2) 16bit data: occupy one register, the high 8 bits are in front and the low 8 bits are in the back during transmission;
   (3) 32bit data: 2 registers are occupied, the high 16bit data is located in the low address, and the low 16bit data is located in the high address middle;

address	value
Addr0	bit31~bit16
Addr1	bit15~bit0

(4) For 64bit data: occupy 4 registers, the highest 16bit data is located at the low address, and the lowest 16bit data is located at the highest address;

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

4. Request APU example

	illustrate	size	example
MBAP header	transaction identifier Hi	1	0x15
	transaction identifier Lo	1	0x01
	protocol identifier	2	0x0000
	length	2	0x0006
	unit identifier	1	0xFF
MODBUS request	function code	1	0x03
	initial address	2	0x0005
	number of registers	2	0x0001

3 Address definition and operation requirements

3.1 Read register operation (0x03)

3.1.1 Read version number length

Request version number length:
(1) Address: 0x0000
(2) Number of registers: 1
(3) Example: `00 00 00 00 00 06 01 03 00 00 00 01`

	illustrate	size	example
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
	initial address	2	00 00
	Number of registers	2	00 01

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

	illustrate	size	Example
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
	data bytes	1	02
	data	2	00 29

3.1.2 Read the version number

It is necessary to read the data length of the version number first, and then use this length to read the version number;
since the bit width of the modbus holding register is 16, when the length of the version number is singular, the value of the last register is valid for the upper 8 bits and the lower

8 bits. Fill with 0. The starting address is fixed at 0x01, and the number of registers read is calculated by this method: (version number bytes + 1)/2

Request to read version number:

(1) Start address: 0x00 01
(2) Number of registers: (version number bytes + 1)/2
Example: `00 00 00 00 00 06 01 03 00 01 00 15`

	illustrate	size	example
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
	initial address	2	00 01
	Number of registers	2	00 15

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 of the data area into characters, you can get the version: *"AM8-QC-RXE-7.8.02-bate7_FAKE_TGF_ENCODER"*

	illustrate	size	example
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

	illustrate	size	example
	data bytes	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 according to the number of counters. The number of counters occupies one register, and each counter ID occupies 2 registers; If the number of counters is more than the number of requests, the number of requests will be intercepted to respond. If the actual number is less than the number of requests, the rest will be filled with 0xFF.

Number of request counters:

Read the current number of counters, occupying only one register, so the starting address and number of registers are fixed; Example: `00 00 00 00 00 06 01 03 00 82 00 01`

	illustrate	size	example
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
	initial address	2	00 82
	Number of registers	2	00 01

Number of response counters:

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

	illustrate	size	example
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

	illustrate	size	example
MODBUS response	function code	1	03
	data bytes	1	02
	data	2	0x00 0x02 (counter number)

Request a list of current counters:

(1) Address range: 0x0083 ~ 0x0882, the effective address is determined according to the number of existing counters (0x83 + number of registers × 2), the starting address must be the starting address of the target ID, such as 0x0083 is the start of the ID of the 0th counter Address, 0x0085 is the ID start address of the first counter;
(2) Number of registers to read: Since one counter ID occupies 2 registers, and the lower 16 bits are stored in a small address, the number of registers requested needs to be a multiple of 2, such as 2, 4, 6;
(3) 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;

	illustrate	size	example
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
	start address 2	00 83	
	Number of registers	2	00 04

(4) Example 2: `00 00 00 00 00 06 01 03 00 85 00 04`
Explanation: Request to read two consecutive counter IDs starting from the first counter;

Response current counter list:

Example: `00 00 00 00 00 0b 01 03 08 00 00 00 00 00 00 00 01`
Explanation: 2 counter IDs are read, 0 and 1;

	illustrate	size	example
MBAP header	transaction identifier Hi	1	00
	transaction identifier Lo	1	00
	protocol identifier	2	00 00
	length	2	00 0b

	illustrate	size	example
MODBUS response	unit identifier	1	01
	function code	1	03
	data bytes	1	08
	data	8	00 00 00 00 (0th counter ID) 00 00 00 01 (1st counter ID)

3.1.4 Read Counter Information

To read the counter, you need to write the ID of the counter to be read first.

Request to write the ID of the counter to be read:

(1) Start address: 0x0883
(2) Number of registers: 2,
(3) Number of bytes: 4;
(4) Value: The counter ID that exists in the counter list, the high 16bit of the counter ID is placed in the front, and the low 16bit (5) Example: `00 00 00 00 00 0B 01 10 08 83 00 02 04 00 00 00 01 Explanation: Set` the
ID of the counter to be read to 1. After the setting is successful, you can use the read function code to read the counter The data;

	illustrate	size	example
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
	initial address	2	08 83
	Number of registers	2	00 02
	data bytes	1	04
	data	4	00 00 00 01

Response write counter ID to be read:

Example: `00 00 00 00 00 06 01 10 08 83 00 02`

	illustrate	size	example
MBAP header	transaction identifier Hi	1	00

	illustrate	size	example
	transaction identifier Lo	1	00
	protocol identifier	2	00 00
	length	2	00 06
	unit identifier	1	01
MODBUS response	function code	1	10
	initial address	2	08 83
	Number of registers	2	00 02

Request to read counter status data:

(1) Address: 0x0883
(2) Number of registers: 0x06
(3) Example: `00 00 00 00 00 06 01 03 08 83 00 06`

	illustrate	size	Example
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
	initial 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 read counter with ID 1 gets its target count value 0x0a, and the current count value is 0x05;

	illustrate	size	Example
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

	illustrate	size	Example
	data bytes	1	0c
	data	12	00 00 00 01 (ID)00 00 00 0a (target value)00 00 00 05 (current value)

3.1.5 Read the current mode

Request the current operating mode:

(1) Address: 0x0889
(2) Number of registers: 1
(3) Example: `00 00 00 00 00 06 01 03 08 89 00 01`

	illustrate	size	example
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
	initial address	2	08 89
	Number of registers	2	00 01

In response to the current operating mode:

Example: `00 00 00 00 00 05 01 03 02 00 03`
Explanation: Currently in configuration mode;

	illustrate	size	example
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
	data bytes	1	02
	data	2	00 03

3.1.6 IO board operation

At present, the maximum number of IO boards supported by the system is 5;
request to read the number of IO boards:

(1) Start address: 0x088a
(2) Number of registers: 1
(3) Example: `00 00 00 00 00 06 01 03 08 8a 00 01`

	illustrate	size	example
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
	initial address	2	08 8a
	Number of registers	2	00 01

Response to read the number of IO boards:

Example: `00 00 00 00 00 05 01 03 02 00 01`
Explanation: The current configuration uses one IO board

	illustrate	size	example
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
	data bytes	1	02
	data	2	00 01

Request to read IO board input status:

(1) Start address: 0x088b (2187) ~ 0x0894 (2196), the effective address is determined according to the number of IO boards used, the input status of each IO board occupies 2 registers, which must start with the lower 16 bits of the board input; such as 0x088b (2187) is the starting address of the input signal of the 0th board, 0x088d (2189) is the starting address of the input signal of the first IO board, and so on;
(2) Number of registers: the number of registers read A multiple of 2 is required;
(3) Example: `00 00 00 00 00 06 01 03 08 8b 00 02`
Explanation: read the input state of the 0th board;

	illustrate	size	example
MBAP header	transaction identifier Hi	1	00

	illustrate	size	example
	transaction identifier Lo	1	00
	protocol identifier	2	00 00
	length	2	00 06
	unit identifier	1	01
MODBUS request	function code	1	03
	initial address	2	08 8b
	Number of registers	2	00 01

(4) Example 2: `00 00 00 00 00 06 01 03 08 8b 00 04`
Explanation: Read the input status of the 0th and 1st boards;
(5) Example 3: `00 00 00 00 00 06 01 03 08 8d 00 02`
Explanation: Read the input state of the first board;

Response to read IO board input status:

Example: `00 00 00 00 00 07 01 03 04 00 00 00 00`
Explanation: The 32-bit input status of the 0th IO board is read as 00 00 00 00;

	illustrate	size	Example
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
	data bytes	1	04
	data	4	00 00 00 00

Request to read IO board output status:

(1) Starting address: 0x8b3 (2227) ~ 0x08bc (2236), it should be adjusted according to the number of IO boards used, and the effective address is determined according to the IO boards used;
(2) Number of registers: each board The input has 32 bits and needs to occupy 2 bytes, so the number of registers needs to be a multiple of 2;
(3) Example: `00 00 00 00 00 06 01 03 08 b3 00 02`
Explanation: read the output status of the 0th board;

	illustrate	size	example
MBAP header	transaction identifier Hi	1	00
	transaction identifier Lo	1	00

	illustrate	size	example
	protocol identifier	2	00 00
	length	2	00 06
	unit identifier	1	01
MODBUS request	function code	1	03
	initial address	2	08 b3
	Number of registers	2	00 02

Response to read 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 the 0th IO board is read as 00 00 00 00;

	illustrate	size	example
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
	data bytes	1	04
	data	4	00 00 00 00

3.1.7 Axis quantity reading

Request to read the number of axes:

(1) Address: 0X08db(2267)
(2) Number of registers: 0x01
(3) Example: `00 00 00 00 00 06 01 03 08 db 00 01`

	illustrate	size	example
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
	initial address	2	08db
	Number of registers	2	00 01

Response to read IO board output status:

Example: `00 00 00 00 00 05 01 03 02 00 06`
Explanation: The number of axes currently in use is 6 axes;

	illustrate	size	example
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
	data bytes	1	02
	data	2	00 06 (number of axes)

3.1.8 Axis position

At present, the system supports up to 8 axes, the axis position data precision is 3 decimals, the unit is ° (for example, 1000 is 1°), and each position occupies 2 registers;

request: read axis position

(1) Address: 0x08dc～0x08eb (2283), the actual effective address is determined according to the number of axes used (0x08dc + number of axes × 2), the start address must be the start address of an axis, such as 0x08dc for the 0th axis Position start address, 0x08de is the position start address of the 1st axis; Number of registers: Each axis position occupies 2 registers, so the read register needs a multiple of 2, such as 2, 4;
(2) Example: `00 00 00 00 00 06 01 03 08 dc 00 02`
Explanation: read the position of axis 0;

	illustrate	size	Example
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
	initial address	2	08 dc
	Number of registers	2	00 02

(3) Example 2: `00 00 00 00 00 06 01 03 08 dc 00 04`
Explanation: read the position of axis 0 and axis 1;

(4) Example 3: `00 00 00 00 00 06 01 03 08 de 00 02`
Explanation: read the position of axis 1;

Response to read axis position:

Example: `00 00 00 00 00 07 01 03 04 00 00 00 00`
Explanation: read the position of axis 0 as 0;

	illustrate	size	example
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
	data bytes	1	04
	data	4	00 00 00 00 (axis position)

3.1.9 World Coordinate Position

At present, the system supports a total of 8 world coordinate positions, the precision of the value is 3 decimal places, the unit is mm (for example, 1000 is 1mm), and each value occupies 2 registers;
the definition of the world coordinate axis name: 0:X, 1:Y , 2:Z, 3:U, 4:V, 5:W, 6:M7, 7:M8

Request to read the world coordinate position:

(1) Address: 0x091c～0x092b (2347), the starting address must be the starting address of a world coordinate, such as 0x091c is the starting address of the world axis 0 (x) position, 0x091e is the world axis 1 (Y) The starting address of the position;
(2) Number of registers: each axis position occupies 2 registers, so the read register needs a multiple of 2, such as 2, 4, 6;
(3) Example: `00 00 00 00 00 06 01 03 09 1c 00 02`
Explanation: read the position of the world axis 0;

	illustrate	size	example
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
	initial address	2	09 1c

	illustrate	size	example
	Number of registers	2	00 02

(4) Example 2: `00 00 00 00 00 06 01 03 09 1c 00 04`
Explanation: Read the world position of axis 0 and axis 1;
(5) Example 3: `00 00 00 00 00 06 01 03 09 1e 00 02`
Explanation : read the world position of axis 1;

Response to read axis world position:

Example: `00 00 00 00 00 07 01 03 04 00 00 00 00`
Explanation: The world position of read axis 0 is 0;

	illustrate	size	Example
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
	data bytes	1	04
	data	4	00 00 00 00 (axis position)

3.1.10 Read the current alarm number

Request to read alarm number:

The alarm number occupies one register, so the number of registers is fixed at 1, and the address is 0x095c;
example: `00 00 00 00 00 06 01 03 09 5c 00 01`

	illustrate	size	example
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
	initial 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;

	illustrate	size	example
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
	data bytes	1	02
	data	2	03 25 (alarm number)

3.1.11 Cycle

The unit of cycle time is ms, and one cycle time occupies 4 registers, so the number of registers is fixed at 4;

Request current cycle time:

(1) Start address: 0X95d(2397);
(2) Number of registers: 4
(3) Example: `00 00 00 00 00 06 01 03 09 5d 00 04`

	illustrate	size	Example
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
	initial address	2	09 5d
	Number of registers	2	00 04

Response current cycle time:

Example: `00 00 00 00 00 0b 01 03 08 00 00 00 00 00 00 00 00`
Explanation: The current cycle time is 0;

	illustrate	size	example
MBAP header	transaction identifier Hi	1	00

	illustrate	size	example
	transaction identifier Lo	1	00
	protocol identifier	2	00 00
	length	2	00 0b
	unit identifier	1	01
MODBUS response	function code	1	03
	data bytes	1	08
	data	8	00 00 00 00 00 00 00 00 (current cycle time)

Request die cycle time:

(1) Start address: 0X0961(2401);
(2) Number of registers: 4
(3) Example: `00 00 00 00 00 06 01 03 09 61 00 04`

	illustrate	size	example
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
	initial address	2	09 61
	Number of registers	2	00 04

Response punch cycle time:

Example: `00 00 00 00 00 0b 01 03 08 00 00 00 00 00 00 00 00`
Explanation: The upper die cycle time is 0;

	illustrate	size	example
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

	illustrate	size	example
MODBUS response	function code	1	03
	data bytes	1	08
	data	8	00 00 00 00 00 00 00 00 (last cycle time)

3.1.12 Host address

Request to read host address:

(1) Address: 0X0965(2405)
(2) Number of registers: 1
(3) Example: `00 00 00 00 00 06 01 03 09 65 00 01

	illustrate	size	example
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
	initial 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: The host address is 0x01;

	illustrate	size	example
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
	data bytes	1	02
	data	2	00 01 (host address)

3.1.13 Read the current torque

2580 represents double the torque, and the torque of each axis occupies 1 register;

Request read torque:

(1) Address range: 0X966 (2406)～0x096d (2413)
(2) Number of registers: add the starting address and do not exceed the address range;
(3) Example: `00 00 00 00 00 06 01 03 09 66 00 01`
Explanation: read the current torque data of axis 0;

	illustrate	size	example
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
	initial 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: The current torque of axis 0 is 0;

	illustrate	size	Example
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
	data bytes	1	02
	data	2	00 00 (torque)

3.1.14 Read the current speed of the axis

The speed unit is RPM, and the speed of each axis occupies 1 register;

Request read speed:

(1) Address range: 0x0986 (2438) ~ 0x098d (2445)
(2) Number of registers: add the starting address and do not exceed the address range;

(3) Example: `00 00 00 00 00 06 01 03 09 86 00 01`
Explanation: read the current speed data of axis 0;

	illustrate	size	Example
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
	initial address	2	09 86
	Number of registers	2	00 01

Response read speed:

Example: `00 00 00 00 00 05 01 03 02 00 00`
Explanation: The current speed of axis 0 is 0;

	illustrate	size	example
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
	data bytes	1	02
	data	2	00 00 (speed)

3.1.15 Mobile state

Request to read mobile status:

(1) Address: 0X09a6(2470)
(2) Number of registers: 1
(3) Definition of moving state value: moving state; 0: Stop; 1: Move;
(4) Example: `00 00 00 00 00 06 01 03 09 a6 00 01`

	illustrate	size	Example
MBAP header	transaction identifier Hi	1	00
	transaction identifier Lo	1	00
	protocol identifier	2	00 00

	illustrate	size	Example
	length	2	00 06
	unit identifier	1	01
MODBUS request	function code	1	03
	initial address	2	09 a6
	Number of registers	2	00 01

Response to read move status:

Example: `00 00 00 00 00 05 01 03 02 00 01`
Explanation: Currently in moving state;

	illustrate	size	example
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
	data bytes	1	02
	data	2	00 01

3.2 Write single register (0x06)

3.2.1 Commands

Each command occupies a register, and writing any valid value (0～0xFFFF) to the register will execute the corresponding command; the command operation only supports function code 0x06.

Request to stop the current action:

(1) Address: 4e 20
(2) Value: Any
(3) Example: `00 00 00 00 00 06 01 06 4e 20 00 01`
Explanation: Write 1 to the register that stops the current action to stop the current action;

	illustrate	size	Example
MBAP header	transaction identifier Hi	1	00
	transaction identifier Lo	1	00
	protocol identifier	2	00 00
	length	2	00 06

	illustrate	size	Example
	unit identifier	1	01
MODBUS request	function code	1	06
	register address	2	4e 20
	register value	2	00 01

response is the same as the request frame;
other commands view the address table;

3.2.2 Modify the global speed

The global speed unit is %, the precision is 1 decimal place, that is, the effective value is 0~1000 corresponding to 0.0%~100.0%; this value can also be read by function code 0x03;

Request to modify the global speed:

(1) Address: 0x4ee8
(2) Value: 0～1000
(3) Example: `00 00 00 00 00 06 01 06 4e e8 00 64`
Explanation: Set the global speed to 0x64 (ie 10.0%);

	illustrate	size	Example
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
	register address	2	4e e8
	register value	2	00 064

The response is the same as the request;

3.3 Write multiple registers (0x10)

3.3.1 Write IO board whole board output

The current system supports up to 5 IO boards, each IO board has a 32-bit output and occupies 2 registers;

Request IO board output:

(1) Function code: 0x10
(2) Address: 0X08b3(2227)～0x08bc(2236), the starting address must be the starting

address of the output of a certain IO board, such as using 0x08b3, 0x08b5 is correct, if using 0x08b4 is wrong;
(3) Number of registers: Since each IO board occupies 2 registers, it needs to be a multiple of 2;
(4) Example: `00 00 00 00 00 0B 01 10 08 b3 00 02 04 00 00 00 01`
Explanation: set the 0th An IO board outputs the 0th port;

	illustrate	size	example
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
	initial address	2	08 b3
	Number of registers	2	00 02
	data bytes	1	04
	data	4	00 00 00 01

Response IO board output:

Example: `00 00 00 00 00 06 01 10 08 b3 00 02`

	illustrate	size	Example
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
	initial address	2	08 b3
	Number of registers	2	00 02

3.3.2 Modify the counter

The parameters include: ID, target value and current value, a total of 3, each occupies 2 registers, so the number of fixed registers for the modification counter is 6;
ID: used to specify the counter to be modified,
target value: modification is not allowed on the host side , so there is no requirement for this value;
current value: modification is allowed;

Request to modify the counter:

(1) Function code 0x10
(2) Start address: 4e 52
(3) Number of registers: 6
(4) 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 current value of the counter whose ID is 1 needs to be changed to 8;

	illustrate	size	Example
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 request	function code	1	10
	initial address	2	4e 52
	Number of registers	2	00 06
	data bytes	1	0c
	data	12	00 00 00 01 00 00 00 00 00 00 00 08

Response modifies the counter:

Example: `00 00 00 00 00 06 01 10 4e 52 00 06`

	illustrate	size	Example
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
	initial address	2	4e 52
	Number of registers	2	00 06

3.3.3 Modify the state of a single output point

It is used to modify the output state of a certain point of an IO board. The parameters to be set include the specified IO board (starting from 0), the specified output point (0~31), and the specified state (0: off 1: on); 3 Each parameter occupies one register;
IO board value range: 0～3 common IO; 4～6: M value; 7: EUIO;

output point range: 0～31
State value definition: 0: off 1: on

Request to modify the state of a single output point:

(1) Function code: 0x10
(2) Start address: 0x4e58
(3) Number of registers: 3
(4) Example: `00 00 00 00 00 0d 01 10 4e 58 00 03 06 00 00 00 01 00 01`
Explanation: set the first The first output port of the 0 board is connected;

	illustrate	size	Example
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 request	function code	1	10
	initial address	2	4e 58
	Number of registers	2	00 03
	data bytes	1	06
	data	6	00 00 00 01 00 01

In response to modifying the state of a single output point:

Example: `00 00 00 00 00 06 01 10 4e 58 00 03`

	illustrate	size	Example
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
	initial address	2	4e 58
	Number of registers	2	00 03

3.3.4 Modify address parameters

The host system parameter table shares 1000 values, each value occupies 2 registers, and the high 16 bits are located at the low address; this address can also be read using the function code 0x03;

Request to modify address parameters:

(1) Address: 0x4f4c (20300)~0x56d5 (22229)
(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 the 0th parameter to 100;

	illustrate	size	Example
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
	initial address	2	4f 4c
	Number of registers	2	00 02
	data bytes	1	04
	data	4	00 00 00 64

In response to modifying the address parameter:

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

	illustrate	size	example
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
	initial 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 the first parameter 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 the 0th parameter to 100, and the value of the 1st parameter to 200;

3.3.5 Send point data

There can be multiple sets of irregular point data in the system, which are identified by ID. The stacked ID is the data source ID, and the visual data source is fixed at 100. The data in the data source can be managed through the following process.

1. First configure the data source data information to be modified, you can send the following commands:

ask:

(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`
The modified data source ID is 100, each point contains 6 data, and only the data of axes 1, 2, and 6 are used, which is the data configuration used by general plane vision;

	illustrate	size	example
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 request	function code	1	10
	initial address	2	75 32
	Number of registers	2	00 03
	data bytes	1	06
	data	6	00 64 00 06 00 23

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

	illustrate	size	Example
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
	initial address	2	75 32
	Number of registers	2	00 03

2. Send point data, when there are few points (the data is less than 256 bytes):

(1) Address: 0x7535 (30005)~0x9C40 (40000)
(2) Number of registers: According to the number of points and format, each axis has 2 registers for 1 data
(3) Example 1: `00 00 00 00 00 0b 01 10 75 35 00 1A 34 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 00 00 00 0B 00 00 00 0C`
Explanation: Clear the original data source data and add 2 points

	illustrate	size	example
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 request	function code	1	10
	initial address	2	75 35
	Number of registers	26	00 1A
	data bytes	1	34
	data	52	00 02 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0C

3. Send point data, when there are many points (single frame data is greater than 256 bytes):

   The data can be split and sent, the first frame clears the original data source (0x7536 is set to 1), the subsequent frames are not cleared (0x7536 is set to 0), or one frame can be selected to send, this is a non-standard protocol, our system will internally The field for the number of data bytes is ignored.

4 Function code 0x03, 0x04, 0x06 and 0x10 address table definition

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Read version number length	0	0	version number length	R	1	64	
Read the version number content	1	1	version number string start	R	/	/	String, maximum length 128 bytes;
Read the version number content	...	...	...	R	/	/	
Read the version number content	64	40	version number string	R	/	/	
Read the current model number length	65	41	Current model number length	R	1	64	Host network port does not support
Read the content of the current model number	66	42	Current model number low	R	/	/	string
Read the content of the current model number	...	...	...	R	/	/	
Read the content of the current model number	129	81	Current model number high	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Number of read counter lists	130	82	number of counters	R	0	1024	
Read the contents of the counter list	131	83	ID high bits of the 0th counter	R	/	/	The number of valid addresses is determined by the number of counters; each ID occupies 32 bits;
Read the contents of the counter list	132	84	ID low bits of the 0th counter	R	/	/	
Read the contents of the counter list	133	85	ID high bits of the first counter	R	/	/	
Read the contents of the counter list	134	86	ID low order of the 1st counter	R	/	/	
Read the contents of the counter list	...	...	...	R	/	/	
Read the contents of the counter list	2178	882	ID low order of the 1023rd counter	R	/	/	
Read Counter Status ID	2179	883	Counter ID high bit to be read	W/R	/	/	Write the ID of the counter to

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
							be read first, and then read the data of this segment;
Read Counter Status ID	2180	884	Counter ID low bit to be read	W/R	/	/	
Read Counter Status	2181	885	Counter target value high	R	/	/	
Read Counter Status	2182	886	Counter target value low bits	R	/	/	
Read Counter Status	2183	887	Counter current value high	R	/	/	
Read Counter Status	2184	888	Counter current value low bit	R	/	/	
current mode	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	Temporarily use up to 4 blocks, and the fifth block is not used;
Board 0 Input Status	2187	88B	16~31 input status of board 0	R	/	/	
Board 0 Input Status	2188	88C	0~15 input status of board 0	R	/	/	
Board 1	2189	88D	16~31	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Input Status			input status of board 1				
Board 1 Input Status	2190	88E	0~15 input status of board 1	R	/	/	
Board 2 Input Status	2191	88F	16~31 input status of board 2	R	/	/	
Board 2 Input Status	2192	890	0~15 input status of board 2	R	/	/	
Board 3 Input Status	2193	891	16~31 input status of board 3	R	/	/	
Board 3 Input Status	2194	892	0~15 input status of board 3	R	/	/	
Board 4 Input Status	2195	893	16~31 input status of board 4	R	/	/	
Board 4 Input Status	2196	894	0~15 input status of board 4	R	/	/	
M board 0 input status	2197	895	16~31 output status of M board 0	R	/	/	
M board 0 input status	2198	896	0~15 output status of M board 0	R	/	/	
M board 1 input status	2199	897	16~31 output status of M board 1	R	/	/	
M board 1 input status	2200	898	0~15 output status of M	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
			board 1				
M board 2 input status	2201	899	16~31 output status of M board 2	R	/	/	
M board 2 input status	2202	89A	0~15 output status of M board 2	R	/	/	
EU board 0 input status	2203	89B	16~31 output status of EU board 0	R	/	/	
EU board 0 input status	2204	89C	0~15 output status of EU board 0	R	/	/	
reserved	...	...	reserved	R	/	/	
reserved	2226	8B2	reserved	R	/	/	
IO board 0 output status	2227	8B3	16~31 output status of IO board 0	R/W	/	/	
IO board 0 output status	2228	8B4	0~15 output status of IO board 0	R/W	/	/	
IO board 1 output status	2229	8B5	16~31 output status of IO board 1	R/W	/	/	
IO board 1 output status	2230	8B6	0~15 output status of IO board 1	R/W	/	/	
IO board 2 output status	2231	8B7	16~31 output status of IO	R/W	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
			board 2				
IO board 2 output status	2232	8B8	0~15 output status of IO board 2	R/W	/	/	
IO board 3 output status	2233	8B9	16~31 output status of IO board 3	R/W	/	/	
IO board 3 output status	2234	8BA	0~15 output status of IO board 3	R/W	/	/	
IO board 4 output status	2235	8BB	16~31 output status of IO board 4	R/W	/	/	
IO board 4 output status	2236	8BC	0~15 output status of IO board 4	R/W	/	/	
M board 0 output status	2237	8BD	16~31 output status of M board 0	R/W	/	/	
M board 0 output status	2238	8BE	0~15 output status of M board 0	R/W	/	/	
M board 1 output status	2239	8BD	16~31 output status of M board 1	R/W	/	/	
M board 1 output status	2240	8BE	0~15 output status of M board 1	R/W	/	/	
M board 2 output	2241	8BF	16~31 output	R/W	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
status			status of M board 2				
M board 2 output status	2242	8C0	0~15 output status of M board 2	R/W	/	/	
EU board 0 output status	2243	8C1	16~31 output status of EU board 0	R/W	/	/	
EU board 0 output status	2244	8C2	0~15 output status of EU board 0	R/W	/	/	
reserved	...	...	reserved	R/W	/	/	
reserved	2266	8DA	reserved	R/W	/	/	
Total number of axes	2267	8DB	Total number of axes	R	0	8	Currently supports up to 8 axes
Axis 1 position	2268	8DC	Axis 0 (J1) position high	R	/	/	The original value is double, and the precision is 3 decimal places. Enlarge it by 1000 times and convert it to an integer;
Axis 1 position	2269	8DD	Axis 0 (J1) position low	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Axis 2 position	2270	8DE	Axis 1 (J2) position high	R	/	/	
Axis 2 position	2271	8DF	Axis 1 (J2) position low	R	/	/	
Axis 3 position	2272	8E0	Axis 2 (J3) position high	R	/	/	
Axis 3 position	2273	8E1	Axis 2 (J3) position low	R	/	/	
Axis 4 position	2274	8E2	Axis 3 (J4) position high	R	/	/	
Axis 4 position	2275	8E3	Axis 3 (J4) position low	R	/	/	
Axis 5 position	2276	8E4	Axis 4 (J5) position high	R	/	/	
Axis 5 position	2277	8E5	Axis 4 (J5) position low	R	/	/	
Axis 6 position	2278	8E6	Axis 5 (J6) position high	R	/	/	
Axis 6 position	2279	8E7	Axis 5 (J6) position low	R	/	/	
Axis 7 position	2280	8E8	Axis 6 (J7) position high	R	/	/	
Axis 7 position	2281	8E9	Axis 6 (J7) position low	R	/	/	
Axis 8 position	2282	8EA	Axis 7 (J8) position	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
			high				
Axis 8 position	2283	8EB	Axis 7 (J8) position low	R	/	/	
reserved	...	...	reserved	R	/	/	
reserved	2331	91B	reserved	R	/	/	
X-axis world position	2332	91C	World axis 0 (X) world high	R	/	/	The original value is double, and the precision is 3 decimal places. Enlarge it by 1000 times and convert it to an integer;
X-axis world position	2333	91D	World coordinate axis 0 (X) position low	R	/	/	
Y-axis world position	2334	91E	World axis 1 (Y) position high	R	/	/	
Y-axis world position	2335	91F	World coordinate axis 1 (Y) position low	R	/	/	
Z axis world position	2336	920	World coordinate axis 2 (Z) position high	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Z axis world position	2337	921	World coordinate axis 2 (Z) position low	R	/	/	
U-axis world position	2338	922	World coordinate axis 3 (U) position high	R	/	/	
U-axis world position	2339	923	World coordinate axis 3 (U) position low	R	/	/	
V-axis world position	2340	924	World axis 4 (V) position high	R	/	/	
V-axis world position	2341	925	World coordinate axis 4 (V) position low	R	/	/	
W-axis world position	2342	926	World coordinate axis 5 (W) position high	R	/	/	
W-axis world position	2343	927	World coordinate axis 5 (W) position low	R	/	/	
M7 axis world position	2344	928	World coordinate axis 6 (M7) position high	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
M7 axis world position	2345	929	World coordinate axis 6 (M7) position low	R	/	/	
M8 axis world position	2346	92A	World coordinate axis 7 (M8) position high	R	/	/	
M8 axis world position	2347	92B	World coordinate axis 7 (M8) position low	R	/	/	
reserved	...	...	reserved	R	/	/	
reserved	2395	95B	reserved	R	/	/	
alarm number	2396	95C	Current alarm number	R	/	/	
current cycle	2397	95D	Current cycle 48～63bit	R	/	/	64-bit data; high-order first, low-order last;
current cycle	2398	95E	Current cycle 32～47bit	R	/	/	
current cycle	2399	95F	Current cycle 16～31bit	R	/	/	
current cycle	2400	960	Current cycle 0～15bit	R	/	/	
Upper die cycle	2401	961	Upper die cycle 48～63bit	R	/	/	
Upper die cycle	2402	962	Upper die cycle 32～	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
			47bit				
Upper die cycle	2403	963	Upper die cycle 16～31bit	R	/	/	
Upper die cycle	2404	964	Upper die cycle 0～15bit	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 5 current torque	R	/	/	
Axis 7 torque	2412	96C	Axis 6 current torque	R	/	/	
Axis 8 torque	2413	96D	Axis 7 current torque	R	/	/	
reserved	...	...	reserved	R	/	/	
reserved	2437	985	reserved	R	/	/	
Axis 1	2438	986	Axis 0	R	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
speed			current speed				
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	/	/	
reserved	...	...	reserved	R	/	/	
reserved	2469	9A5	reserved	R	/	/	
mobile state	2470	9A6	Current machine movement status	R	0	1	1 is moving, 0 is still
origin state	2471	9A7	origin state	R	0	1	After all axes have set the origin, the status is 1, otherwise it is 0
Current user	2472	9A8	current user length	R	1	64	The host does not

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
							have this item
Current user	2473	9A9	current user low	R	/	/	The host does not have this item
Current user	...	...	...	R	/	/	
Current user	2536	9E8	Current user high	R	/	/	The host does not have this item
The total number of read patterns	2537	9E9	Number of models	R	/	/	The host does not have this item
Model number list	...	...	Model number list	R	/	/	The host does not have this item
reserved	...	...	reserved	R	/	/	
Command: stop the current action immediately	20000	4E20	Stop the current action immediately	W	/	/	Command: stop the current action immediately (start from the beginning)
Command: Pause current action	20001	4E21	Pause the current action	W	/	/	Command: Pause the current action (start from the current step)
Command: enter single loop	20002	4E22	enter single loop	W	/	/	Command: enter single loop

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Command: start button	20003	4E23	startButton: start button	W	/	/	Command: start button
Command: stop key	20004	4E24	stopButton: stop button	W	/	/	Command: stop key
Command: run the next command after clearing the alarm	20005	4E25	Run the next instruction after clearing the alarm	W	/	/	Command: run the next command after clearing the alarm
Command: Clear Alarm and Continue Autorun	20006	4E26	Clear the alarm and continue automatic operation	W	/	/	Command: Clear the alarm and continue automatic operation: (in automatic operation state)
reserved	...	...	reserved	W	/	/	
reserved	20049	4E51	reserved	W	/	/	
Counter modification	20050	4E52	To be written counter ID high	W	/	/	You can first read the counter list to get the configurable counter ID
Counter modification	20051	4E53	Waiting to write counter ID low bit	W	/	/	
Counter modification	20052	4E54	Set counter target value high bit	W	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Counter modification	20053	4E55	Set counter target value low bit	W	/	/	
Counter modification	20054	4E56	Set the high bit of the current value of the counter	W	/	/	The host cannot modify the target value
Counter modification	20055	4E57	Set the low bit of the current value of the counter	W	/	/	
output point control	20056	4E58	Set output board ID	W	0	7	(0～4: IO board, 5～7: M value (manual network port version is temporarily not supported), 8: EUIO)
output point control	20057	4E59	set output point	W	0	31	(0～31)
output point control	20058	4E5A	set output state	W	0	1	(0: OFF, 1: ON)
stack modification	20059	4E5B	Stack ID to be modified	W	/	/	
stack modification	20060	4E5C	X interval low	W	/	/	data width? ? The host does not have this function

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
stack modification	20061	4E5D	X interval high	W	/	/	
stack modification	20062	4E5E	Y interval low	W	/	/	
stack modification	20063	4E5F	Y interval high	W	/	/	
stack modification	20064	4E60	Z interval low	W	/	/	
stack modification	20065	4E61	Z interval high	W	/	/	
stack modification	20066	4E62	X count low	W	/	/	
stack modification	20067	4E63	X count high	W	/	/	
stack modification	20068	4E64	Y count low	W	/	/	
stack modification	20069	4E65	Y count high	W	/	/	
stack modification	20070	4E66	Z count low	W	/	/	
stack modification	20071	4E67	Z count high	W	/	/	
reserved	...	...	reserved	W	/	/	
global velocity	20200	4EE8	global velocity	RW	0	1000	32-bit precision1
reserved	...	...	reserved	RW	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
Internal parameter table address	20300	4F4C	allpara[0] high bit	RW	/	/	1000 parameters of the host function are operated in total, each value is 32 bits
Internal parameter table address	20301	4F4D	allpara[0] low order	RW	/	/	
Internal parameter table address	20302	4F4E	allpara[1] high bit	RW	/	/	
Internal parameter table address	20303	4F4F	allpara[1] low order	RW	/	/	
Internal parameter table address	...	...	allpara[900] low order	R	/	/	
Internal parameter table address	22229	56D5	allpara[999] low order	R	/	/	
photo command	22130	5672	photo command	R	/	/	Host does not support
switch model	22131	5673	switch model	R	/	/	("d1": model number (host network port version is

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
							not supported temporarily))
ID of the data source to be viewed	30000	7530	ID of the data source to be viewed	RW	/	/	Visual data source ID100, others are stack IDs
The buffer length corresponding to the data source ID	30001	7531	The buffer length corresponding to the data source ID	R	/	/	
The ID of the data source to be modified	30002	7532	The ID of the data source to be modified	RW	/	/	Visual data source ID100, others are stack IDs
Data source point format to be modified	30003	7533	The point format of the data source to be modified, whether 6 data is a point or 4 data is a point	RW	/	/	Default is 6
bitmask of data source points to be modified	30004	7534	The bit format mask of the data source to be modified. 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	30005	7535	Length of	RW	/	/	

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
data points to be received			data points to be received				
Clear the data source data to be modified	30006	7536	Clear the data source data to be modified	RW	/	/	1: to clear
data source point data	30007	7537	Axis 1 data high	W	/	/	
data source point data	30008	7538	Axis 1 data low	W	/	/	
data source point data	...	...	...	W	/	/	
data source point data	40000	9C40	data source point data	W	/	/	

5 Function code 0x01, 0x05 address table definition

Function Description	address (decimal)	hexadecimal	Briefly	r/w	minimum	maximum	Remark
General output point Y010	0	0	0xFF00 is on, 0x0000 is off	W	/	/	
General output point Y247	159	9F	0xFF00 is on, 0x0000 is off	W	/	/	
M010	160	A0	0xFF00 is on, 0x0000 is off	W	/	/	
M147	255	FF	0xFF00 is on, 0x0000 is off	W	/	/	
EUY010	256	100	0xFF00 is on, 0x0000 is off	W	/	/	
EUY047	287	11F	0xFF00 is on, 0x0000 is off	W	/	/	