4-channel DI switch detection counter, MQTT protocol, online webpage display, WiFi module WJ162

W external antenna N internal antenna X suction cup antenna

Figure 1 Appearance of WJ162 module

Product features:

- 4-channel switch input, supporting NPN and PNP inputs
- Each DI channel can be used as a counter or frequency measurement
- Supports MQTT communication protocol and Modbus TCP
- The number of pulses per revolution can be set for speed measurement
- Built in web page function, data can be queried through web pages
- Wide power supply range: 8~32VDC
- High reliability, easy programming, and easy application
- Standard DIN35 rail installation, convenient for centralized wiring
- Users can set module IP addresses and other parameters on the webpage
- Low cost, small size, modular design
- Dimensions: 79 x 69.5x 25mm

Typical applications:

- Flow meter pulse counting or flow measurement
- Counting the number of punch presses
- Production output counting
- Ethernet industrial automation control system
- Industrial site signal isolation and long-distance transmission
- Hall sensor speed measurement
- Level detection and counting of photoelectric sensors
- Motor speed measurement
- IoT switch signal acquisition

Product Overview:

The WJ162 product is an IoT and industrial Ethernet acquisition module that enables transparent data exchange between sensors and networks. The switch data of the sensor can be forwarded to the network.

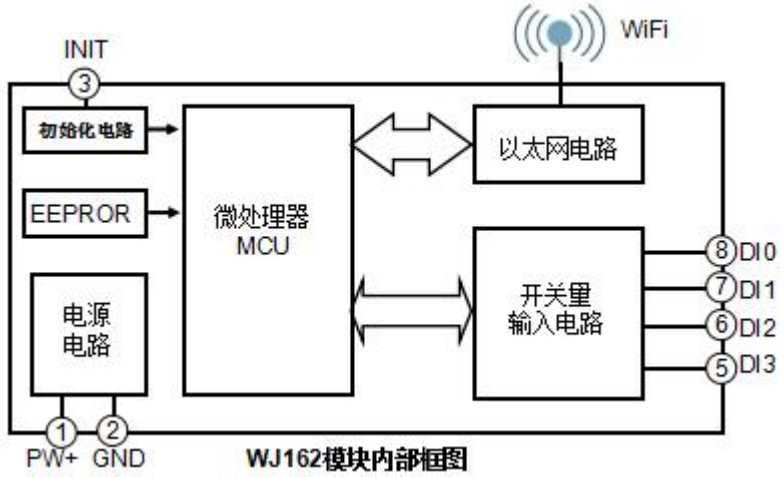


Figure 2 Internal Block Diagram of WJ162 Module

The WJ162 series products include power conditioning, switch quantity acquisition, and WiFi network interface communication. The communication method adopts MQTT protocol. TCP is a transport layer based protocol that is widely used and a reliable connection oriented protocol. Users can directly set module IP addresses, subnet masks, etc. on the webpage. Can be used for monitoring and controlling the operation of sensor devices.

The WJ162 series products are intelligent monitoring and control systems based on microcontrollers, where user set module IP addresses, subnet masks, and other configuration information are stored in non-volatile memory EEPROM.

The WJ162 series products are designed and manufactured according to industrial standards, with strong anti-interference ability and high reliability. The working temperature range is -45 ℃ to+85 ℃.

Function Introduction:

The WJ162 remote I/O module can be used to measure four switch signals. Can be used as a 4-channel counter or 4-channel frequency and speed measurement.

1、 Switch signal input

4-channel switch signal input, can be connected to dry contact NPN and wet contact PNP. Please refer to the wiring diagram for details.

2、 Communication Protocol

Communication interface: WiFi network interface. It can connect to WiFi in the local area network and then connect to Ethernet.

Communication protocol: Supports MQTT protocol and can connect to various MQTT servers such as Alibaba Cloud, Tencent Cloud, Huawei Cloud, China Mobile IoT OneNET, private cloud, etc. MODBUS TCP protocol can also be used to achieve industrial Ethernet data exchange.

Network cache: 2K bytes (for both sending and receiving)

Communication response time: less than 10mS.

3、 anti-interference

There is a transient suppression diode inside the module, which can effectively suppress various surge pulses and protect the module.

Product model:

WJ162 – WiFi - ___

Form of antenna

W external antenna

N built-in antenna

X suction cup antenna

Communication interface

WiFi: Output as WiFi network interface

WJ162 General Parameters:

(Typical @+25 ℃, Vs is 24VDC)

Input type: switch input, 4-channel (DI0~DI3).

Low level: Input<1V

High level: Input 3.5~30V

Frequency range 0-20KHz

Counting range 0-0xFFFFFFFF

Input resistance: 30K Ω

Communication: MQTT communication protocol or MODBUS TCP communication protocol

Web page: Support web access module, view current data, support web page setting module parameters.

Interface: WiFi network interface.

Working power supply:+8~32VDC wide power supply range, with internal anti reverse and overvoltage protection circuits

Power consumption: less than 1W

Working temperature: -45~+80 ℃

Working humidity: 10~90% (no condensation)

Storage temperature: -45~+80 ℃

Storage humidity: 10~95% (no condensation)

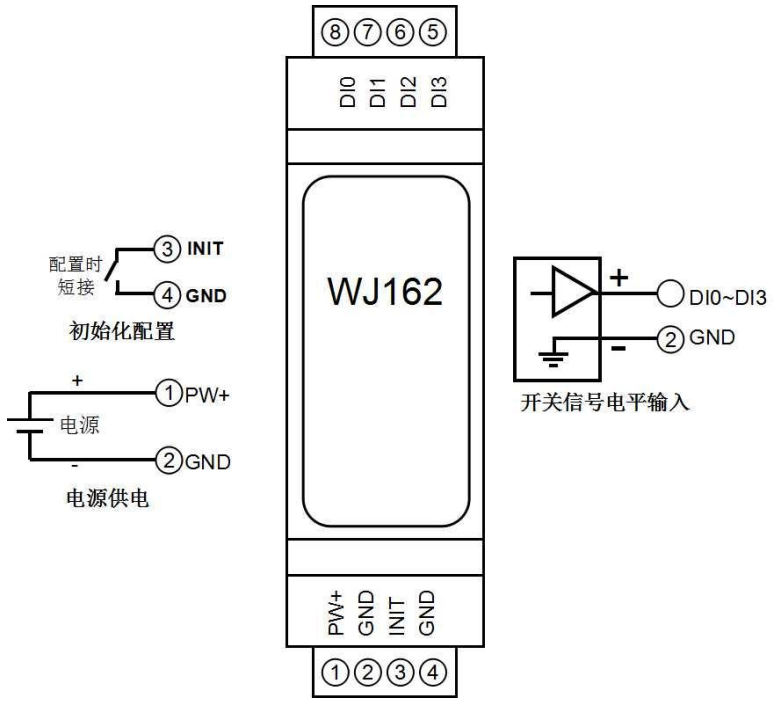
Isolation voltage resistance: non isolated

Dimensions: 79 mm x 69.5mm x 25mm

Pin definition and wiring:

Pin	name	Description	Pin	name	Description
one	PW+	Positive end of power supply	**five**	DI3	Channel 3 switch signal input terminal
two	GND	Negative terminal of power supply, signal common ground	**six**	DI2	Channel 2 switch signal input terminal
three	INIT	Set communication parameters	**seven**	DI1	Channel 1 switch signal input terminal
four	GND	Signal public area	**eight**	DI0	Channel 0 switch signal input terminal

Note: **The** pins **with the** same name are internally connected



Firstly, configure the WJ162 module through your mobile phone

1. Put the module into AP mode

(1) Short circuit the 3rd pin (Initiat) and 4th pin (GND) of the module.

(2) Open the wireless LAN on your phone or Settings → WLAN ", find the WiFi name starting with" wifi 8 "and connect to it. The WiFi password is 12345678

2. Enter the module webpage.

After connecting to the WiFi of the module, wait a few seconds and it will automatically redirect to the built-in webpage of the module, as shown in the left figure. If the phone cannot automatically redirect, you can also open the mobile browser and enter the website 192.168.4.1 to log in.

Click on the configuration module parameter link to enter the configuration interface

3. Configure module DI parameters

Please modify the following parameters according to actual needs:

（1） DI input method: Choose NPN or PNP input based on the actual sensor connected. After selecting NPN input, internally connect the pull-up voltage to the positive power supply, with a pull-up resistance of 10K ohms; Select PNP input and turn off the pull-up voltage internally.

（2） Whether the DI level state is reversed: If the read state is opposite to the actual state, you can set the DI level state to be reversed and output.

（3） DI counting edge: Different edge trigger counts can be set, and the default rising edge count can be used normally. If set to count both rising and falling edges, the count value will be twice the actual number of pulses.

（4） DI0~DI3 pulses per revolution: The number of pulses per revolution for DI. If you need to measure the speed, please set it according to the actual parameters. The module will automatically convert the rotational speed per minute.

（5） DI0~DI3 filtering time: The value range is 0 to 65535.
If it is 0, it means no filtering; The other values represent the filtering time, in mS (milliseconds). If the DI input point is a mechanical switch or mechanical relay, it is recommended to set the filtering time to 20mS.

（6） DI0~DI3 pulse rate: Set the actual value corresponding to each pulse, default to 1, and convert the actual engineering value to the actual pulse based on this value. For example, if each pulse is 0.005mm and can be set to 0.005, then the actual engineering value is 0.005 * number of pulses.

4. Configure module WiFi parameters

Please modify the following parameters according to actual needs:

（7） WiFi account: Connect to the WiFi coverage in this area.

（8） WiFi password: Fill in the WiFi password, if already connected, do not re-enter.

（9） Local IP settings: If only MQTT protocol is used, it can be set to automatically obtain IP. If you want to access data through Modbus TCP or web pages, it is recommended to manually set it to a fixed IP address to facilitate communication between the IP address and the module.

（10） IP address: Set the IP address of the module, which must be in the current WiFi network segment and not the same as the IP address of other devices in the local area network. For example, if the IP of the WiFi router is 192.168.0.1, the IP of the module can be set to 192.168.0.7

（11） Default gateway: The gateway of the module, fill in the IP address of the current WiFi router. For example, if the IP address of a WiFi router is 192.168.0.1, simply fill in this IP address

（12） Subnet Mask: The subnet mask of the module. If there is no cross network segment, fill in the default value of 255.255.255.0

（13） Module Name: User defined name for a module to distinguish between different modules.

（14） MQTT settings: If MQTT communication is used, the MQTT function needs to be turned on.

（15） MQTT server address: Fill in the URL of the MQTT server,

For example: brokere.emqx.io
If the local server IP is 192.168.0.100, you can write 192.168.0.100

（16） Please fill in the MQTT client ID, username, password, port, publish topic, subscribe topic, and other parameters according to the requirements of the MQTT server. The QoS of MQTT is 0 and cannot be modified.

（17） MQTT publishing interval: The time interval in milliseconds during which the module automatically publishes data to the MQTT server. Set to 0 to cancel the scheduled publishing function.

（18） Automatic MQTT publishing for DI status changes: default is' No '. This function is only suitable for situations where the pulse changes very slowly. If any channel has a pulse change, it will publish data to the MQTT server once. It is not recommended to set it to "Yes" for situations with rapid pulse changes.
Otherwise, there will be a large amount of data sent.

5. Save parameters

After completing the parameter settings, click the save and restart button, and the module will save the parameters and automatically restart.
Users can disconnect the short circuit between pin 3 (Initiat) and pin 4 (GND) of the module. The module will enter normal working mode (STA mode) and automatically connect to the current WiFi.

After successful saving, the following interface will appear:

设置已经保存，模块将自动重启，点击这里返回主页.

6. View data online on the webpage

Click on the online data viewing link on the module's homepage to enter the data viewing interface. As shown in the left figure.
If the IP address of the module is 192.168.0.5, users can also obtain JSON format data by accessing the link 192.168.0.5/readData.

The DI state represents the input level state, which can also be the flipped state.

The pulse counter is the cumulative number of measured pulses.

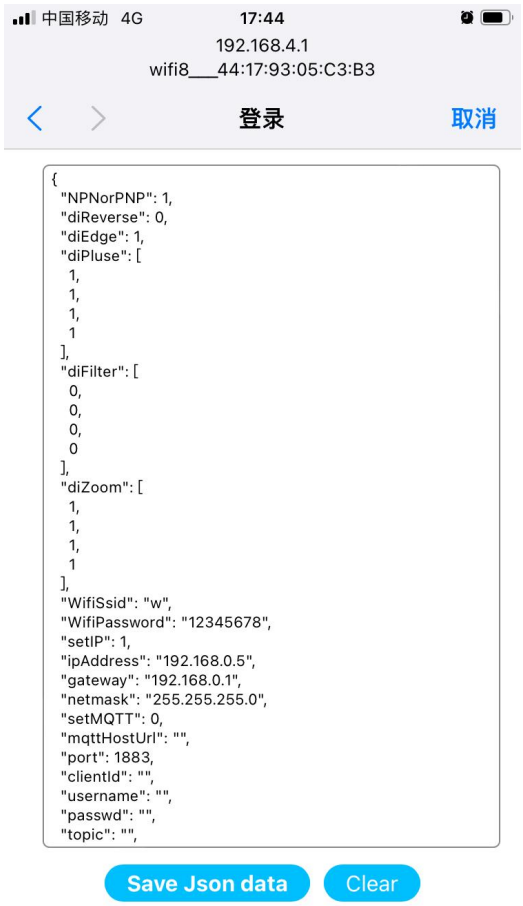
The pulse frequency is the number of pulses per second.

The pulse time interval is the time interval between the two most recent pulses.
The unit is (seconds)

The actual engineering value is obtained by multiplying the value of the pulse counter by the pulse multiplier set on the webpage. Used for automatically converting actual flow, length, production, and other data.

The rotational speed is obtained by converting the frequency and the number of pulses per revolution. Used for automatically converting actual revolutions per minute.

The reset count value can be written as 0 to the table of the corresponding channel, and then click on Settings to reset the count value. Other values can also be set to modify the count value.



7. Batch setting parameters

Click on the Json Batch Configuration link on the module's homepage to enter the Batch Settings interface. As shown in the left figure.
The data must be in standard JSON format, and all parameters can be set or only some parameters can be set.
If there are many products to be set up, batch setting can save time.
After completing the filling, click the button Save Json data.

Example 1: Only changing the WiFi account password can send:

```
{
  "WifiSsid": "w",
  "WifiPassword": "12345678",
  "setIP": 1,
  "ipAddress": "192.168.0.5",
  "gateway": "192.168.0.1",
  "netmask": "255.255.255.0",
}
```

Example 2: Only modifying MQTT parameters can send:

```
{
  "setMQTT": 1,
  "mqttHostUrl": "broker.emqx.io",
  "port": 1883,
```

	"clientId": "mqtt_test_001", "username": "", "passwd": "", "topic": "mqtt_topic_001", "pubTime": 2000, "puboncchange": 0 }

8. The module webpage can also be opened on the local area network

If the module is already connected to the local WiFi, you can enter the module IP in the computer or mobile browser, such as 192.168.0.5, to open the module webpage (provided that the computer IP or mobile IP is in the same network segment as the module, and the login operation should be based on the current module IP address), and then enter the internal webpage of the module. You can also configure modules or read module data, and the operation method is the same as the table above.

MQTT protocol

After setting the parameters of MQTT, the module will automatically connect to the MQTT server and publish the collected data to the set MQTT publishing topic in the following format:

{
"devName":"98CDAC3FA407",
"time":135536,
"diState":[1,1,1,0],
"counter":[0,0,0,0],
"frequency":[0,0,0,0],
"cycle":[0,0,0,0],
"actualData":[0,0,0,0],
"speed":[0,0,0,0]
}

Format Description:

The module name 'devName' can be modified on the webpage as needed

The internal time of the 'time' module, measured in mS.

The 'diState' represents the input level state, which can also be a flipped state.

The "counter" pulse counter is the cumulative number of measured pulses.

The pulse frequency is the number of pulses per second.

The "cycle" pulse time interval refers to the time interval between the two most recent pulses. The unit is (seconds)

The actual engineering value of 'practicalData' is obtained by multiplying the value of the pulse counter by the pulse multiplier set on the webpage. Used for automatically converting actual flow, length, production, and other data.

The speed is obtained by converting the frequency and the number of pulses per revolution. Used for automatically converting actual revolutions per minute.

2. Publish the following data to the MQTT subscription topic of the module, which can be reset or modified to the pulse counter.

Set pulse counters DI0~DI3, which can be 0 or other numerical values:

```
{
"setDI0Count":"0",
"setDI1Count":"0",
"setDI2Count":"0",
"setDI3Count":"0"
}
```

Or:

```
{
"setDI0Count":"1000",
"setDI1Count":"2000",
"setDI2Count":"3000",
"setDI3Count":"4000"
}
```

Only set a single product:

```
{ "setDI0Count":"0" }
```

Simultaneously set the same value for all channels:

```
{"setAllDICount":"0"}
```

Modbus TCP protocol

(1) Modbus TCP data frames:

Transmission over TCP/IP Ethernet, supporting Ethernet II and 802.3 frame formats. As shown in Figure 3, the Modbus TCP data frame consists of three parts: packet header, function code, and data.

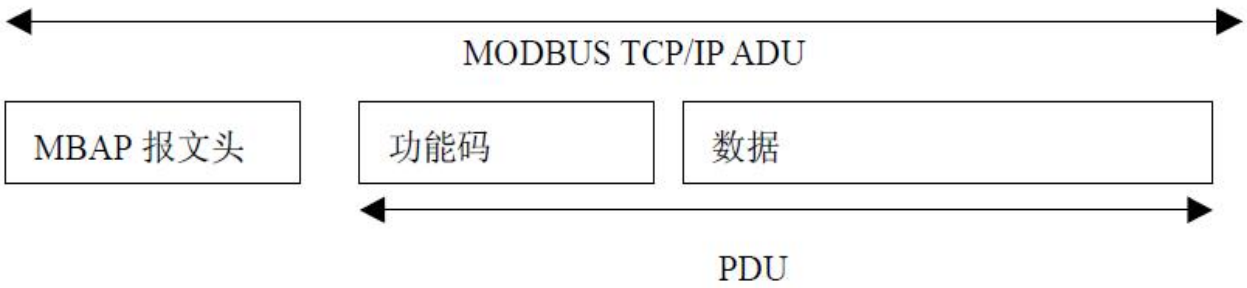


Figure 6: Request/Response of MODBUS on TCP/IP

(2) MBAP message header description:

The MBAP header (MBAP, Modbus Application Protocol, Modbus Application Protocol) is divided into 4 fields, totaling 7 bytes, as shown in Table 1.

Table 1: MBAP Message Header

Domain	Length (B)	Description
Transmission	2 bytes	Indicate the transmission of a MODBUS query/response

identification		
Protocol Logo	2 bytes	0=MODBUS protocol
Length	2 bytes	Subsequent byte count
Unit identifier	1 byte	Identification code of remote slave station connected on serial link or other bus

(3) Modbus function code:

Modbus function codes are divided into three types, namely:

(1) Public Function Code: Defined function codes that ensure their uniqueness and are recognized by Modbus.org;

(2) There are two sets of user-defined function codes, namely 65-72 and 100-110, which do not require approval but do not guarantee the uniqueness of code usage. If it becomes public code, it needs to be approved by RFC;

(3) The reserved functional code, which is used by certain companies on certain traditional devices, cannot be used for public purposes.

Among the commonly used public function codes, WJ162 supports some function codes, as shown below:

Function code		**name**	**explain**
01	Read Coil Status	Read coil status	1 represents high level, 0 represents low level.
03	Read Holding Register	Read and hold register	1 represents high level, 0 represents low level.
05	Write Single Coil	Write a single coil	1 indicates that the transistor is conducting, and 0 indicates that the transistor is disconnected.
06	Write Single Register	Write a single register	1 indicates that the transistor is conducting, and 0 indicates that the transistor is disconnected.
fifteen	Write Multiple Coils	Write multiple coils	
sixteen	Write Multiple Registers	Write multiple registers	

(4) Description of supported function codes

01 (0x01) Reading coil

In a remote device, use this function code to read the continuous status of the coil from 1 to 2000. The request PDU specifies the starting address, which is the designated first coil address and coil number. Address the coil from scratch. Therefore, addressing coils 1-16 are 0-15.

Divide the coils in the response message into individual coils based on each bit in the data field. The indication status is 1=ON and 0=OFF. The first data serves as the LSB (least significant bit) of the byte, and the subsequent coil data is arranged in ascending order to form an 8-bit byte. If the returned output quantity is not a multiple of eight, the remaining bits in the last data byte will be filled with zeros (up to the high-order end of the byte). The byte count field indicates the complete number of bytes in the data

Example of function code 01, read 8-channel DI data, register addresses 00033~00040:

request			**response**		
Field Name		**hexadecimal**	**Field Name**		**hexadecimal**
	Transmission identification	01		Transmission identification	01
		00			00

MBAP message header	n		MBAP message header		
	Protocol Logo	00		Protocol Logo	00
		00			00
	length	00		length	00
		06			04
	Unit identifier	01		Unit identifier	01
Function code		01	Function code		01
Starting address Hi		00	Byte count		01
Starting address Lo		twenty	Output status DI7-DI0		00
Output quantity Hi		00			
Output quantity Lo		08			

03 (0x03) Read hold register

In a remote device, use this function code to read the contents of consecutive blocks in the hold register. The request PDU specifies the starting register address and the number of registers. Address registers from scratch. Therefore, addressing registers 1-16 are 0-15. In the response message, each register has two bytes, with the first byte being the data high bit and the second byte being the data low bit.

Example of function code 03, read 8-channel DI data, register address 40033:

request			**response**		
Field Name		**hexadecimal**	**Field Name**		**hexadecimal**
MBAP message header	Transmission identification	01	MBAP message header	Transmission identification	01
		00			00
	Protocol Logo	00		Protocol Logo	00
		00			00
	length	00		length	00
		06			05
	Unit identifier	01		Unit identifier	01
Function code		03	Function code		03
Starting address Hi		00	Byte count		02
Starting address Lo		twenty	Register value Hi (0x00)		00
Register number Hi		00	Register value Lo (DI7-DI0)		00
Register number Lo		01			

05 (0x05) Write a single coil

On a remote device, use this function code to write a single output as ON or OFF. The request PDU specifies the mandatory coil address. Address the coil from scratch. Therefore, addressing coil address 1 is 0. The constant of the coil range indicates the requested ON/OFF state. Hexadecimal value 0xFF00 requests the coil to be ON. Hexadecimal value 0x0000 requests the coil to be OFF. All other values are illegal and have no effect on the coil. The correct response is the same as a request.

For example, for function code 05, set channel DO0 to ON, which is 1, and register address 00001:

request			response		
Field Name		hexadecimal	Field Name		hexadecimal
MBAP message header	Transmission identification	01	MBAP message header	Transmission identification	01
		00			00
	Protocol Logo	00		Protocol Logo	00
		00			00
	length	00		length	00
		06			06
	Unit identifier	01		Unit identifier	01
Function code		05	Function code		05
Output Address Hi		00	Output Address Hi		00
Output address Lo		00	Output address Lo		00
Output value Hi		FF	Output value Hi		FF
Output value Lo		00	Output value Lo		00

06 (0x06) Write a single register

In a remote device, use this function code to write a single hold register. The request PDU specifies the address written to the register. Address registers from scratch. Therefore, address register address 1 is 0.

The correct response is the same as a request.

For example, for function code 06, set all channels DO0~DO7 to 1, hexadecimal to 0xFF, and register address 40001:

request			response		
Field Name		hexadecimal	Field Name		hexadecimal
MBAP message header	Transmission identification	01	MBAP message header	Transmission identification	01
		00			00
	Protocol Logo	00		Protocol Logo	00
		00			00
	length	00		length	00
		06			06
	Unit identifier	01		Unit identifier	01
Function code		06	Function code		06
Register Address Hi		00	Register Address Hi		00
Register Address Lo		00	Register Address Lo		00
Register value Hi		00	Register value Hi		00
Register value Lo		FF	Register value Lo		FF

15 (0x0F) Write multiple coils

On a remote device, use this function code to write multiple outputs as ON or OFF. The request PDU specifies the mandatory coil address. Address the coil from scratch. Therefore, addressing coil address 1 is 0. The constant of the coil range indicates the requested ON/OFF state. The data is converted from hexadecimal to binary and arranged in bits, with a bit value of 1 requesting the coil to be ON and a bit value of 0 requesting the coil to be OFF.

For example, for function code 15, set channel DO0 and DO1 to ON, which is 00000011, and register address 00001:

request			response		
Field Name		hexadecimal	Field Name		hexadecimal
MBAP message header	Transmission identification	01	MBAP message header	Transmission identification	01
		00			00
	Protocol Logo	00		Protocol Logo	00
		00			00
	length	00		length	00
		06			06
	Unit identifier	01		Unit identifier	01
Function code		0F	Function code		0F
Start address Hi		00	Start address Hi		00
Starting address Lo		00	Starting address Lo		00
Number of coils Hi		00	Number of coils Hi		00
Number of coils Lo		02	Number of coils Lo		02
Byte count		01			
Output value		02			

16 (0x10) Write multiple registers

In a remote device, use this function code to write multiple hold registers. The request PDU specifies the address written to the register. Address registers from scratch. Therefore, address register address 1 is 0. Example of function code 16, set the PWM values for channels DO0 and DO1 to 5 and 6, register address 40001:

request			response		
Field Name		hexadecimal	Field Name		hexadecimal
MBAP message header	Transmission identification	01	MBAP message header	Transmission identification	01
		00			00
	Protocol Logo	00		Protocol Logo	00
		00			00
	length	00		length	00
		06			06
	Unit	01		Unit identifier	01

identifier			
Function code	ten	Function code	ten
Start register address Hi	00	Start register address Hi	00
Start register address Lo	00	Start register address Lo	00
Number of registers Hi	00	Number of registers Hi	00
Number of registers Lo	02	Number of registers Lo	02
Byte count	04		
Register value Hi	00		
Register value Lo	05		
Register value Hi	00		
Register value Lo	06		

(5) Description of register addresses for WJ162 (note: addresses are all decimal numbers)

Modbus TCP serves as the server-side, with the address being the IP address of the module and the port being 502

Support Function Code 01

Address 0X (PLC)	**Address (PC, DCS)**	**Data content**	**attribute**	**Data Explanation**
00001	0	DI0 input status	read-only	Level status of DI channels 0-3 0 represents a low-level input, 1 represents a high-level input You can set it as inverted output on the webpage as needed
00002	one	DI1 input status	read-only	
00003	two	DI2 input status	read-only	
00004	three	DI3 input status	read-only	

Support function codes 03, 06, 16

Address 4X (PLC)	**Address (PC, DCS)**	**Data content**	**attribute**	**Data Explanation**
forty thousand and one	0	DI0 input status	read-only	Level status of DI channels 0-3 0 represents a low-level input, 1 represents a high-level input You can set it as inverted output on the webpage as needed
forty thousand and two	one	DI1 input status	read-only	
forty thousand and three	two	DI2 input status	read-only	
forty thousand and four	three	DI3 input status	read-only	
40005~40006	4~5	DI0 pulse counter	Read/ Write	Long integers (0x0000000~0xFFFFFFFF), DI channel 0-3 pulse counting, unsigned, storage order is CDAB, DI0 low 16 bits 40005, high 16 bits 40006, the same applies to other channels. The counter can be reset by directly
40007~40008	6~7	DI1 pulse counter	Read/ Write	
40009~40010	8~9	DI2 pulse counter	Read/ Write	

40011~40012	10~11	DI3 pulse counter	Read/ Write	writing 0 to the corresponding register, or other values can be written as needed.
40013~40014	12~13	DI0 pulse frequency	read-only	The pulse frequency of DI channels 0-3, data is a 32-bit floating-point number, and the storage order is CDAB.
40015~40016	14~15	DI1 pulse frequency	read-only	
40017~40018	16~17	DI2 pulse frequency	read-only	
40019~40020	18~19	DI3 pulse frequency	read-only	
40021~40022	20~21	DI0 pulse time interval	read-only	The pulse time interval of DI channels 0-3 is the time interval between the last two pulses, with data being 32-bit floating-point numbers and stored in CDAB order. The unit is seconds (s).
40023~40024	22~23	DI1 pulse time interval	read-only	
40025~40026	24~25	DI2 pulse time interval	read-only	
40027~40028	26~27	DI3 pulse time interval	read-only	
40029~40030	28~29	DI0 actual engineering value	read-only	The actual engineering values of DI channels 0-3 are represented as 32-bit floating-point numbers and stored in CDAB order. The value is the pulse count multiplied by the pulse multiplier set on the webpage. Used for automatic calculation of flow or length, etc.
40031~40032	30~31	DI1 actual engineering value	read-only	
40033~40034	32~33	Actual engineering value of DI2	read-only	
40035~40036	34~35	Actual engineering value of DI3	read-only	
40037~40038	36~37	DI0 speed	read-only	Long integers (0x0000000~0xFFFFFFFF), The storage order is CDAB, with rotational speed per minute. The speed is calculated based on the number of pulses per revolution set in the configuration webpage.
40039~40040	38~39	DI1 speed	read-only	
40041~40042	40~41	DI2 speed	read-only	
40043~40044	42~43	DI3 speed	read-only	
forty thousand two hundred and eleven	two hundred and ten	Module Name	read-only	High bit: 0x01 Low bit: 0x62

Common problems with WJ162

1、 How to determine the status of a module based on lighting

The **light** is on **twice** for **1 second:** the module is waiting for the configured AP mode, and you can use your phone to connect to the module's WiFi settings starting from WiFi 8.

The **light** is on **once** every **1** second**:** the module is currently connected to WiFi. If it cannot be connected for a long time, please reset the WiFi parameters of the module.

The **light** is on **once** every **5** seconds**:** the module has been connected to WiFi and is working normally.

2. Cross network segment issues

If the IP of the device and the communicating PC are not in the same network segment and are directly connected via Ethernet or under the same sub router, then the two cannot communicate at all.

give an example:

Device IP: 192.168.0.7

Subnet mask: 255.255.255.0

PC's IP: 192.168.1.100

Subnet mask: 255.255.255.0

Due to the device's IP being 192.168.0.7, it is unable to log in to the device's webpage or ping it on the PC.

If you want the two to communicate, you need to set the subnet mask of the device and PC, as well as the subnet mask on the router, to 255.255.0.0, so that you can log in to the module webpage.

3. The device can ping, but the webpage cannot be opened

There may be several reasons for this:

1) The device has set a static IP address that conflicts with the IP addresses of existing devices in the network

2) The HTTP server port has been modified (default should be 80)

3) Other reasons

Solution: Reset the device to an unused IP address; Restore factory settings or enter the correct port when opening the browser.

4. Every once in a while, there is a disconnection and reconnection

Every once in a while, there will be a phenomenon of disconnection and reconnection

Reason: There is an issue of IP address conflict between the serial server and other devices

5. Communication is **abnormal, network connection cannot** be established**, or search cannot** be **found**

The firewall of the current computer needs to be turned off (in the Windows firewall settings)

Three local ports must not conflict, meaning they must be set to different values. Default values are 23, 26, and 29

Having illegal MAC addresses, such as full FF MAC addresses, may result in inability to connect to the target IP address or duplicate MAC addresses.

Illegal IP addresses, such as network segments that are not in the same network segment as the router, may not be able to access the external network.

6. Hardware problem search

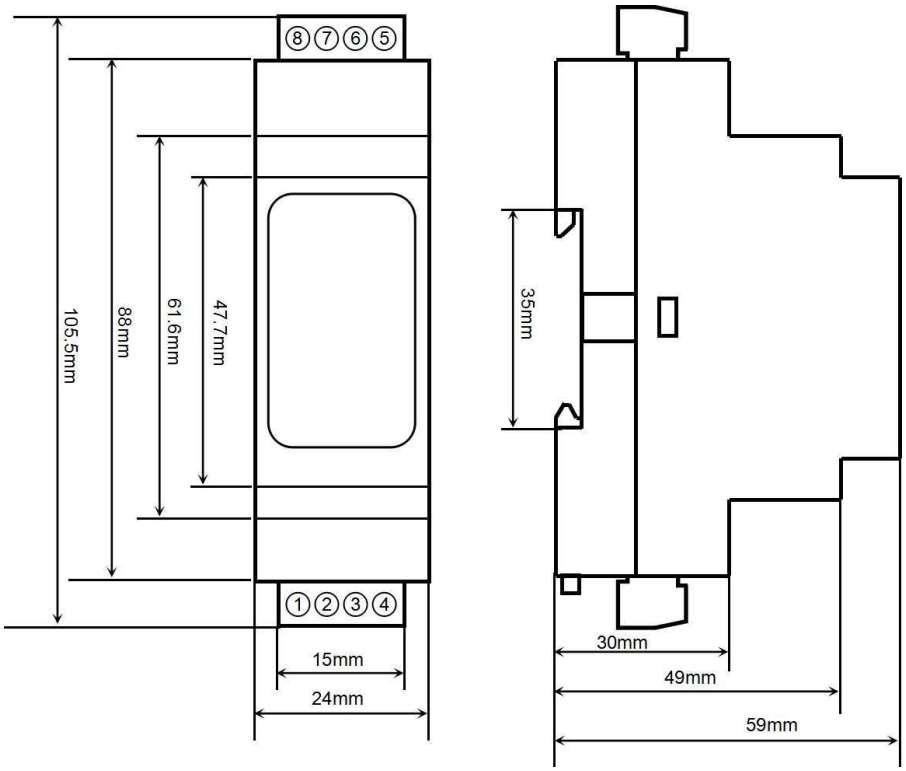
Poor power supply from the power adapter or poor contact of the plug

If the power light and network port light are not on, it means there is no power supply or the hardware is broken

7. MODBUS TCP connection cannot be established

Using Modbus TCP, the port number can only be 502 and cannot be any other numerical value.

Dimensions: (Unit: mm)



guarantee:

Within two years from the date of sale, if the user complies with the storage, transportation, and usage requirements and the product quality is lower than the technical specifications, it can be returned to the factory for free repair. If damage is caused due to violation of operating regulations and requirements, device fees and maintenance fees shall be paid.

Copyright:

Trademark:

The other trademarks and copyrights mentioned in this manual belong to their respective owners.

Version number: V1.0
Date: July 2022