

IOT Relay Programing Manual

V2.0.2

1 Protocol:Dingtian string.....	3
1.1 Query status command/KeepAlive.....	3
1.2 Control command.....	4
1.2.1 ON/OFF/Toggle example.....	4
1.2.2 Delay example.....	4
1.2.3 Jogging(Pulse) example.....	5
1.2.4 Flash example.....	5
1.2.5 Toggle example.....	5
2 Protocol:Dingtian binary.....	6
2.1 default setting.....	6
2.2 command.....	6
2.2.1 read relay status.....	7
2.2.2 write relay.....	8
2.2.3 write relay with delay.....	9
2.2.4 write relay with jogging.....	10
2.2.5 relay keep alive.....	11
2.2.6 write relay with flash.....	12
2.2.7 write relay with toggle.....	13
2.2.8 find relay baord IP(UDP multicast).....	14
3 Protocol:HTTP GET CGI.....	15
3.1 load relay status.....	15
3.2 set relay.....	18
3.3 set relay(multiple).....	21
3.3.1 HTTP Method: GET.....	21
3.3.2 HTTP Method: POST.....	23
3.4 command queue.....	25
3.4.1 HTTP Method: GET.....	25
3.5 load input status.....	27
3.6 load one input status.....	30
3.7 load one relay status.....	30
3.8 Session check.....	31
3.8.1 web config to Enable HTTP Session check.....	31
3.8.2 HTTP CGI test tool.....	32
3.8.3 full example(success):.....	33
3.8.4 full example(fail).....	34
4 Protocol:Modbus-RTU/TCP/ASCII.....	35
4.1 Registers.....	36
4.2 Modbus-RTU + Modbus-RTU Over TCP/UDP.....	38
4.2.1 0x03:Read holding register.....	38
4.2.2 0x06:Write Single Register.....	39

4.2.3 0x10: Write Multiple Register.....	39
4.3 Modbus-TCP/UDP.....	44
4.3.1 0x03:Read holding register.....	44
4.3.2 0x06:Write Single Register.....	45
4.3.3 0x10: Write Multiple Register.....	46
4.4 Modbus-ASCII + Modbus-ASCII Over TCP/UDP.....	50
4.4.1 0x03:Read holding register.....	50
4.4.2 0x06:Write Single Register.....	51
4.4.3 0x10: Write Multiple Register.....	52
5 Protocol:MQTT.....	56
5.4 MQTT Topic.....	57
6 Protocol:CoAP.....	59
6.1 Compile libcoap.....	59
6.2 Get relay status.....	59
6.3 Control relay(simple).....	60
6.4 Control relay.....	61

1 Protocol:Dingtian string

Suport TCP client, TCP server, UDP, CAN/RS485

1.1 Query status command/KeepAlive

command code	00(2 character)	<div>Return character (0:OFF, 1:ON)</div> <div>Format</div> <div>[relay1][relay2]...[relay32]:[input1][input2]...[input32]:[channel count]</div> <div>example(2 channel):</div> <div>00:11:2</div> <div>example(4 channel):</div> <div>0000:1111:4</div> <div>example(8 channel):</div> <div>00000000:11111111:8</div> <div>example(16 channel):</div> <div>0000000000000000:1111111111111111:16</div> <div>example(24 channel):</div> <div>000000000000000000000000:111111111111111111111111:24</div> <div>example(32 channel):</div> <div>00000000000000000000000000000000:11111111111111111111111111111111:32</div>
--------------	-----------------	--

Remarks

1 The command code is a text string and does not need to be followed by a return.

1.2 Control command

X	R	C	P
1(ON) 2(OFF) T(toggle)	1~32 relay X for all relay	null(ON/OFF) :(Delay) *(Jogging) F(flash)	C parameter

1.2.1 ON/OFF/Toggle example

11 # relay 1 ON
21 # relay 1 OFF

13 # relay 3 ON
23 # relay 3 OFF

132 # relay 32 ON
232 # relay 32 OFF

1X # relay all ON
2X # relay all OFF

T1 # toggle R1
T2 # toggle R2

1.2.2 Delay example

Delay ON/OFF some time and then OFF/ON
Delay second range 1-65535

11:30 # relay 1 ON,delay 30second OFF
21:30 # relay 1 OFF,delay 30second ON

13:30 # relay 3 ON,delay 30second OFF
23:30 # relay 3 OFF,delay 30second ON

132:30 # relay 32 ON,delay 30second OFF
232:30 # relay 32 OFF,delay 30second ON

1X:30 # relay all ON,delay 30second OFF
2X:30 # relay all OFF,delay 30second ON

1.2.3 Jogging(Pulse) example

Jogging ON/OFF little time(ms) and then OFF/ON

when no time give after “*”, Jogging time is 500ms(can change from web page),

11* # relay 1 ON,Jogging 500ms OFF
13* # relay 3 ON,Jogging 500ms OFF
132* # relay 32 ON,Jogging 500ms OFF
1X* # all relay ON,Jogging 500ms OFF
11*10 # relay 1 ON,Jogging 10*100ms=1000ms OFF

1.2.4 Flash example

Jogging ON/OFF little time(ms) and then OFF/ON

Jogging time is 500ms(can change from web page),

11F5 # relay 1 ON,Flash(ON<->OFF) interval with 5*100=500ms
21F0 # relay 1 OFF Flash

1.2.5 Toggle example

T1 #relay 1 toggle

2 Protocol:Dingtian binary

Support Different network segment communication(Multicast only support UDP)

Multicast addr: 224.0.2.11

Support password

2.1 default setting

IP	192.168.1.100
Netmask	255.255.255.0
Gateway	192.168.1.1
UDP Port	60000
Multicast addr	224.0.2.11

2.2 command

data bytes >=2byte store format is LSB

example:0x1234,store format is 0x34,0x12

format

field	bytes	comment
command	1	0xFF: set relay 0x07: multicast set relay
result(xor 0xAA)	1	pc->device: 0 xor 0xAA device->pc: result xor 0xAA result=0 success result=other fail
session	1	0~255 device reply the same
relay command	1	0: read relay status 1:write relay 2:write relay with delay 3:write relay with jogging 4:relay keep alive 8:write relay with flash 9:write relay with toggle
password	2	0~9999 0:no password Password incurrent device no reply
command data	x	

2.2.1 read relay status

pc send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	0: read relay status
password	2	0~9999 0:no password

device reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	0: read relay status
Relay status	1(2/4/8CH) 2(16CH) 3(24CH) 4(32CH)	Bit0~31 map to relay1~32 Bit=1 relay on Bit=0 relay off
Input status	1(2/4/8CH) 2(16CH) 3(24CH) 4(32CH)	Bit0~31 map to input1~32 Bit=1 input High Bit=0 input Low

Example:

pc send:

FF AA 00 00 34 12 # password 0x1234

device reply:

FF AA 00 00 01 FF # relay 1 on, all input High

2.2.2 write relay

pc send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	1:write relay
password	2	0~9999 0:no password
relay mask	1(2/4/8CH) 2(16CH) 3(24CH) 4(32CH)	Bit0~31 map to relay relay1~32 Bit=1,relay need update
relay set	1(2/4/8CH) 2(16CH) 3(24CH) 4(32CH)	Bit0~31 map to relay relay1~32 Bit=1,relay on Bit=0,relay off

device reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	1:write relay

Example:

pc send:

FF AA 00 01 34 12 05 01 # relay 1 on, rely 3 off

device reply:

FF AA 00 01

2.2.3 write relay with delay

pc send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	2:write relay with delay
password	2	0~9999 0:no password
relay index and relay on/off	1	Bit0=1 relay on Bit0=0 relay off Bit1~bit7=relay index
Relay delay second	2	1~65535 unit(second)

device reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	2:write relay with delay

Example:

pc send:

FF AA 00 02 34 12 03 05 00 # relay 1 on, delay 5 second off

device reply:

FF AA 00 02

2.2.4 write relay with jogging

pc send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	3:write relay with jogging
password	2	0~9999 0:no password
relay index and relay on/off	1	Bit0=1 relay on Bit0=0 relay off Bit1~bit7=relay index
Relay jogging 100ms	2	1~65535 unit(100ms) 1=100ms 5=500ms

device reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	3:write relay with jogging

Example:

pc send:

FF AA 00 03 34 12 05 05 00 # relay 2 on, jogging

device reply:

FF AA 00 03

2.2.5 relay keep alive

device send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 pc not change
relay command	1	4: relay keep alive
device MAC	6	device MAC address
Relay status	1(2/4/8CH) 2(16CH) 3(24CH) 4(32CH)	Bit0~31 map to relay1~32 Bit=1 relay on Bit=0 relay off
Input status	1(2/4/8CH) 2(16CH) 3(24CH) 4(32CH)	Bit0~31 map to input1~32 Bit=1 input High Bit=0 input Low

pc reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 pc not change
relay command	1	4: relay keep alive

Example:

device send:

FF AA 00 04 BC 34 88 12 34 56 00 FF # MAC BC:34:88:12:34:56 00:relay1-8 OFF, FF:input1-8 HIGH

pc reply:

FF AA 00 00

2.2.6 write relay with flash

pc send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	8:write relay with flash
password	2	0~9999 0:no password
relay index and relay on/off	1	Bit0=1 relay on Bit0=0 relay off Bit1~bit7=relay index
Relay flash 100ms	2	1~65535 unit(100ms) 1=100ms 5=500ms

device reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	8:write relay with flash

Example:

pc send:

FF AA 00 08 34 12 05 05 00 # relay 2 on, flash with 500ms

device reply:

FF AA 00 08

2.2.7 write relay with toggle

pc send

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	9:write relay with toggle
password	2	0~9999 0:no password
relay index	1	Bit0~bit6=relay index

device reply

field	bytes	comment
command	1	0xFF
result(xor 0xAA)	1	0 xor 0xAA=0xAA
session	1	0~255 device not change
relay command	1	9:write relay with toggle

Example:

pc send:

FF AA 00 09 34 12 02 # relay 3 toggle

device reply:

FF AA 00 09

2.2.8 find relay board IP(UDP multicast)

pc send to multicast IP 224.0.2.11:60000

field	bytes	comment
command	1	0x05
result(xor 0xAA)	1	0 xor 0xAA=0xAA

device reply

field	bytes	comment
command	1	0x05
result(xor 0xAA)	1	0 xor 0xAA=0xAA
find_data	32	response C struct(LSB):

response C struct(LSB):

```
typedef struct _find_data_{  
    u32_dt sn;  
    u32_dt sw_ver;  
    u32_dt hw_ver;  
    u32_dt model;  
    u32_dt ip;  
    u32_dt netmask;  
    u32_dt gateway;  
    u32_dt dns;  
} find_data; /* 32 bytes */
```

Example:

pc send:

05 AA # find relay board IP with UDP multicast

device reply:

head sn sw_ver hw_ver model ip netmask gateway dns

05 AA 14920000 0103D400 06030000 2E000000 6401A8C0 00FFFFFF 0101A8C0 0101A8C0

SN:37396

IP:192.168.1.100

netmask:255.255.255.0

gateway:192.168.1.1

dns:192.168.1.1

3 Protocol:HTTP GET CGI

Relay board as HTTP server, accept HTTP GET CGI request.

Support CGI relay on/off

Support CGI relay jogging

Support CGI relay delay

Support CGI password verification

Support CGI Session check(details: 3.8 Session check)

3.1 load relay status

HTTP Method: **GET**

HTTP GET request

parameter	filed	data	comment
1	CGI API	relay_cgi_load.cgi	cgi changeable suffix relay_cgi_load.cgi, relay_cgi_load.php, relay_cgi_load.cs is work ok

HTTP GET response

parameter	filed	data	comment
1	result	0	0: ok other fail
2	relay count	2/4/8/16/24/32	
3	relay 1 status	0/1	0:off 1:on
4	relay 2 status	0/1	0:off 1:on
5	relay 3 status	0/1	0:off 1:on
6	relay 4 status	0/1	0:off 1:on
7	relay 5 status	0/1	0:off 1:on
8	relay 6 status	0/1	0:off 1:on
9	relay 7 status	0/1	0:off 1:on
10	relay 8 status	0/1	0:off 1:on
11	relay 9 status	0/1	0:off

			1:on
12	relay 10 status	0/1	0:off 1:on
13	relay 11 status	0/1	0:off 1:on
14	relay 12 status	0/1	0:off 1:on
15	relay 13 status	0/1	0:off 1:on
16	relay 14 status	0/1	0:off 1:on
17	relay 15 status	0/1	0:off 1:on
18	relay 16 status	0/1	0:off 1:on
19	relay 17 status	0/1	0:off 1:on
20	relay 18 status	0/1	0:off 1:on
21	relay 19 status	0/1	0:off 1:on
22	relay 20 status	0/1	0:off 1:on
23	relay 21 status	0/1	0:off 1:on
24	relay 22 status	0/1	0:off 1:on
25	relay 23 status	0/1	0:off 1:on
26	relay 24 status	0/1	0:off 1:on
27	relay 25 status	0/1	0:off 1:on
28	relay 26 status	0/1	0:off 1:on
29	relay 27 status	0/1	0:off 1:on
30	relay 28 status	0/1	0:off 1:on
31	relay 29 status	0/1	0:off 1:on
32	relay 30 status	0/1	0:off 1:on

33	relay 31 status	0/1	0:off 1:on
34	relay 32 status	0/1	0:off 1:on

example(4 channel relay):

HTTP GET request

http://192.168.1.100/relay_cgi_load.cgi # request relay board HTTP CGI API

HTTP GET response

[&0&4&1&0&1&0&](#) # ok,4 relay,relay 1 on,relay 2 off,relay 3 on, relay 4 off

3.2 set relay

HTTP Method: **GET**

HTTP GET request

parameter	filed	data	comment
1	CGI API	relay_cgi.cgi	cgi suffix variable relay_cgi.cgi, relay_cgi.php, relay_cgi.cs is work ok
2	type	0/1/2/3/4	0:relay on/off 1:relay jogging 2:relay delay 3:relay flash 4:relay toggle
3	relay	0~31	
4	on	0/1	0:off 1:on
5	time	0 1~255 1~65535	0:on/ff 0:time 1:jogging 1~255:time(1=100ms) 2:delay 1~65535:time(second) 3:flash 1~255:time(1=100ms)
6	pwd	0~9999	0~9999 Password incurrent device no response

HTTP GET response

parameter	filed	data	comment
1	result	0	0: ok other fail
2	type	0/1/2/3/4	0:relay on/off 1:relay jogging 2:relay delay
3	relay	0~31	0:relay 1 1:relay 2 ...

			31:relay 32
4	on	0/1	0:off 1:on
5	time	0 1~255 1~65535	0:type 0:time 1:type 1~255:time(1=100ms) 2:type 1~65535:time(second)

example 1(relay on):

HTTP GET request(request relay board HTTP CGI API, set relay 0 on ,time 0,password 0)

<http://192.168.1.100/relay.cgi?type=0&relay=0&on=1&time=0&pwd=0&>

HTTP GET response

[&0&0&0&1&0&](#) # ok, type 0 on/off,relay 0 on,time 0

example 2(relay off):

HTTP GET request(request relay board HTTP CGI API, set relay 0 off ,time 0,password 0)

<http://192.168.1.100/relay.cgi?type=0&relay=0&on=0&time=0&pwd=0&>

HTTP GET response

[&0&0&0&0&0&](#) # ok, type 0 on/off,relay 0 off,time 0

example 3(relay 1 jogging on):

HTTP GET request(request relay board HTTP CGI API, set relay 1 jogging on ,time 500ms,password 4660)

<http://192.168.1.100/relay.cgi?type=1&relay=1&on=1&time=5&pwd=4660&>

HTTP GET response

[&0&1&1&1&5&](#) # ok, type 1 jogging,relay 1 on,time 5(500ms)

example 4(relay 1 jogging off):

HTTP GET request(request relay board HTTP CGI API, set relay 1 jogging off,time 500ms,password 4660)

<http://192.168.1.100/relay.cgi?type=1&relay=1&on=0&time=5&pwd=4660&>

HTTP GET response

[&0&1&1&0&5&](#) # ok, type 1 jogging,relay 1 off,time 5(500ms)

example 5(relay 1 on delay 10 second off):

HTTP GET request(request relay board HTTP CGI API, set relay 1 on delay 10 second off ,time 5 second,password 4660)

<http://192.168.1.100/relay.cgi?type=2&relay=1&on=1&time=10&pwd=4660&>

HTTP GET response

&0&2&1&1&10& # ok, type 2 delay,relay 1 on,time 10 second

example 6(relay 1 off delay 10 second on):

HTTP GET request(request relay board HTTP CGI API, set relay 1 off delay 10 second on ,time 5 second,password 4660)

http://192.168.1.100/relay_cgi.cgi?type=2&relay=1&on=0&time=10&pwd=4660&

HTTP GET response

&0&2&1&0&10& # ok, type 2 delay,relay 1 off,time 10 second

3.3 set relay(multiple)

3.3.1 HTTP Method: GET

HTTP GET request

parameter	filed	data	comment
1	HTTP GET CGI API	relay_bat.cgi	cgi suffix variable relay_bat.cgi, relay_bat.php, relay_bat.cs is work ok
2	count	1-32	
3	pass	0-65535	max 5number password 0: no password 65535: mas password
4	type	0/1/2/3/4	0:relay on/off 1:relay jogging 2:relay delay 3:relay flash 4:relay toggle
5	idx	0~31	0: relay1 1: relay2 ... 31: relay32
6	status	0/1	0:off 1:on
7	time	0 1~255 1~65535	0:on/ff 0:time 1:jogging 1~255:time(1=100ms) 2:delay 1~65535:time(second) 3:flash 1~255:time(1=100ms)

HTTP GET response(json format)

parameter	filed	data	comment
1	status	0	0: ok other fail

example 1(relay1 on, relay2 delay ON 5second then OFF):

HTTP GET request

[http://192.168.1.100/relay_bat.cgi?
count=2&pass=123&type=0,2&idx=0,1&status=1,1&time=0,5&](http://192.168.1.100/relay_bat.cgi?count=2&pass=123&type=0,2&idx=0,1&status=1,1&time=0,5&)

HTTP GET response:

```
{"status":0}
```

3.3.2 HTTP Method: POST

HTTP POST request(json format)

parameter	filed	data	comment
1	HTTP POST CGI API	relay_bat.cgi	cgi suffix variable relay_bat.cgi, relay_bat.php, relay_bat.cs is work ok
2	count	1-32	
3	pass	0-65535	max 5number password 0: no password 65535: mas password
4	type	0/1/2/3/4	0:relay on/off 1:relay jogging 2:relay delay 3:relay flash 4:relay toggle
5	idx	0~31	0: relay1 1: relay2 ... 31: relay32
6	status	0/1	0:off 1:on
7	time	0 1~255 1~65535	0:on/ff 0:time 1:jogging 1~255:time(1=100ms) 2:delay 1~65535:time(second) 3:flash 1~255:time(1=100ms)

HTTP response(json format)

parameter	filed	data	comment
1	status	0	0: ok other fail

example 1(relay1 on, relay2 delay ON 5second then OFF):

HTTP POST request

http://192.168.1.100/relay_bat.cgi

POST data(json format):

```
{"count":2,"pass":123,"type":[0, 2],"idx":[0, 1],"status":[1, 1],"time":[0, 5]}
```

HTTP response:

```
{"status":0}
```


3.4 command queue

3.4.1 HTTP Method: GET

HTTP GET request

parameter	filed	data		comment
1	HTTP GET CGI API	relay_queue.cgi		cgi suffix variable relay_queue.cgi, relay_queue.php, relay_queue.cs is work ok
2	count	2-64		Max channel * 2 example: 4channel max is 4*2=8
3	pass	0-65535		max 5number password 0: no password 65535: mas password
4...	cmd*count	P0	0: RELAY_ONOFF command(3 parameters)	
		P1	Relay index 0-31 map to relay1-32	
		P2	Relay status 0:OFF 1:ON	
		P3	Delay n*100ms goto next command. example n=5 means delay 500ms goto next command	
		P0	1: RELAY_JOGGING command(4 parameters)	
		P1	Relay index 0-31 map to relay1-32	
		P2	Relay status 0:OFF 1:ON	
		P3	100ms count, example 5=500ms	
		P4	Delay n*100ms goto next command. example n=5 means delay 500ms goto next command	
		P0	2: RELAY_DELAY command(4 parameters)	
		P1	Relay index 0-31 map to relay1-32	
		P2	Relay status 0:OFF 1:ON	
		P3	second count, example 5=5second	
		P4	Delay n*100ms goto next command. example n=5 means delay 500ms goto next command	
		P0	3: RELAY_FLASH command(4 parameters)	
		P1	Relay index 0-31 map to relay1-32	
		P2	Relay status 0:OFF 1:ON	
		P3	100ms count, example 5=500ms	
		P3	Delay n*100ms goto next command. example n=5 means delay 500ms goto next command	
		P0	4: RELAY_TOGGLE command(2 parameters)	

		P1	Relay index 0-31 map to relay1-32
		P2	Delay n*100ms goto next command. example n=5 means delay 500ms goto next command
			5: reserved command(not used)
		P0	6: DELAY command(1 parameters)
		P1	second count, example 5=5second
		cmd split with "&"	

HTTP GET response(json format)

parameter	filed	data	comment
1	status	0	0: ok other fail

Example 1:

0 RELAY_ONOFF relay1 ON, **delay 1000ms**(10*100ms) goto next command
2 RELAY_DELAY relay2 ON 5second then OFF, goto next command **immediately**
6 DELAY 10second
4 RELAY_TOGGLE relay1, goto next command **immediately**

HTTP GET request

[http://192.168.1.100/relay_queue?
count=4&pass=123&cmd=0,0,1,10&cmd=2,1,1,5,0&cmd=6,10&cmd=4,0,0&](http://192.168.1.100/relay_queue?count=4&pass=123&cmd=0,0,1,10&cmd=2,1,1,5,0&cmd=6,10&cmd=4,0,0&)

HTTP GET response:

{"status":0}

3.5 load input status

HTTP Method: **GET**

HTTP GET request

parameter	filed	data	comment
1	CGI API	input.cgi	cgi changeable suffix input.cgi, input.php, input.cs is work ok

HTTP GET response

parameter	filed	data	comment
1	result	0	0: ok other fail
2	Input start postion	0	default: 0
3	input count	2/4/8/16/24/32	
4	Input 1 status	0/1	0:low 1:high
5	Input 2 status	0/1	0:low 1:high
6	Input 3 status	0/1	0:low 1:high
7	Input 4 status	0/1	0:low 1:high
8	Input 5 status	0/1	0:low 1:high
9	Input 6 status	0/1	0:low 1:high
10	Input 7 status	0/1	0:low 1:high
11	Input 8 status	0/1	0:low 1:high
12	Input status	0/1	0:low 1:high
13	Input 10 status	0/1	0:low 1:high
14	Input 11 status	0/1	0:low 1:high
15	Input 12 status	0/1	0:low 1:high

16	Input 13 status	0/1	0:low 1:high
17	Input 14 status	0/1	0:low 1:high
18	Input 15 status	0/1	0:low 1:high
19	Input 16 status	0/1	0:low 1:high
20	Input 17 status	0/1	0:low 1:high
21	Input 18 status	0/1	0:low 1:high
22	Input 19 status	0/1	0:low 1:high
23	Input 20 status	0/1	0:low 1:high
24	Input 21 status	0/1	0:low 1:high
25	Input 22 status	0/1	0:low 1:high
26	Input 23 status	0/1	0:low 1:high
27	Input 24 status	0/1	0:low 1:high
28	Input 25 status	0/1	0:low 1:high
29	Input 26 status	0/1	0:low 1:high
30	Input 27 status	0/1	0:low 1:high
31	Input 28 status	0/1	0:low 1:high
32	Input 29 status	0/1	0:low 1:high
33	Input 30 status	0/1	0:low 1:high
34	Input 31 status	0/1	0:low 1:high
35	Input 32 status	0/1	0:low 1:high

example(4 channel relay):

HTTP GET request

<http://192.168.1.100/input.cgi>

request relay board HTTP CGI API

HTTP GET response

&0&0&4&1&0&1&0&

low

ok, reserve, 4 input, input 1 high, Input 2 low, Input 3 high, Input 4

3.6 load one input status

HTTP Method: **GET**

HTTP GET request

parameter	filed	data	comment
1	CGI API	i.x(x is 1 to 32)	http://192.168.1.100/i.1 http://192.168.1.100/i.2 ... http://192.168.1.100/i.32

HTTP GET response

parameter	filed	data	comment
1	Input (x is 1 to 32) status	0/1	0:low 1:high

example:

HTTP GET request

<http://192.168.1.100/i.1> # request input 1 status

HTTP GET response

0 # 0:low, 1:High

3.7 load one relay status

HTTP Method: **GET**

HTTP GET request

parameter	filed	data	comment
1	CGI API	r.x(x is 1 to 32)	http://192.168.1.100/r.1 http://192.168.1.100/r.2 ... http://192.168.1.100/r.32

HTTP GET response

parameter	filed	data	comment
1	relay x(x is 1 to 32) status	0/1	0:low 1:high

example:

HTTP GET request

<http://192.168.1.100/r.1> # request relay 1 status

HTTP GET response

0 # 0:low, 1:High

3.8 Session check

The HTTP CGI session check is implemented by adding a “Cookie” header

“Cookie” header example:

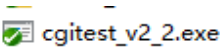
Cookie: session=12345678

3.8.1 web config to Enable HTTP Session check

The screenshot shows a web browser window with the address bar displaying '192.168.1.100/menu_page.html'. The page title is 'Dingtian IOT Relay'. The main content area is titled 'Setting' and contains a table of configuration parameters. A sidebar menu on the left lists various settings like 'Setting', 'Relay Connect', 'Relay CGI Test', etc. The 'HTTP Session' parameter is highlighted with a red annotation 'to YES'.

Hardware Version	V3.6A
Software Version	V3.1.276A
Build Date	2021-12-29 12:21:20
Model	Dingtian DT-R008
Serial Number	7943
Date Time	1970/1/1 08:49:36 Sync Time
NTP Server	pool.ntp.org
Hostname	Dingtian-Relay7943
Hostname+Suffix	Dingtian-Relay + SN ▼
HTTP or HTTPS	HTTP ▼
HTTP Server Port	80 ▼
HTTPS Server Port	443 ▼
HTTP Session	Yes ▼ to "YES"
HTTP Magic Session ID	12345678 ▼
ETH DHCP	No ▼
ETH IP	192.168.1.100
ETH Netmask	255.255.255.0
ETH Gateway	192.168.1.1
ETH DNS	192.168.1.1
ETH MAC	ba:34:88:00:1e:58
STA Enable	No ▼
STA Auth	WPA2 PSK ▼
STA DHCP	No ▼
STA IP	192.168.1.97
STA Netmask	255.255.255.0
STA Gateway	192.168.1.1
STA DNS	192.168.1.1
STA MAC	bc:34:88:00:1e:58
STA WiFi SSID	
STA WiFi Password	
AP Enable	Yes ▼
AP Hide	No ▼

3.8.2 HTTP CGI test tool



CGI test

Relay Board IP

192.168.1.100

port

80

Cookie Session ID

12345678

CGI param

relay.cgi?type=0&relay=0&on=0&time=0&pwd=0&

Call CGI

CGI req

GET /relay.cgi?type=0&relay=0&on=0&time=0&pwd=0& HTTP/1.1
Host: 192.168.1.9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: keep-alive
Cookie: session=12345678

clear log

CGI res

HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 11

%0%0%0%0%0%

3.8.3 full example(success):

HTTP GET request:

GET /relay_cgi.cgi?type=0&relay=0&on=1&time=0&pwd=0& HTTP/1.1

Host: 192.168.1.9

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Encoding: gzip, deflate

Connection: keep-alive

Cookie: session=12345678

HTTP GET response:

HTTP/1.1 200 OK

Content-Type: text/html

Content-Length: 11

&0&0&0&1&0&

3.8.4 full example(fail)

HTTP GET request:

GET /relay_cgi.cgi?type=0&relay=0&on=0&time=0&pwd=0& HTTP/1.1

Host: 192.168.1.9

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Encoding: gzip, deflate

Connection: keep-alive

Cookie: session=1234567

HTTP GET response:

HTTP/1.1 200 OK

Content-Type: text/html

Content-Length: 7

&302&/&

4 Protocol:Modbus-RTU/TCP/ASCII

Support Modbus:

Modbus-RTU

Modbus-TCP/UDP

Modbus-ASCII

Modbus-RTU Over TCP/UDP

Modbus-ASCII Over TCP/UDP

Support Modbus Function:

0x03read holding register

0x06Write Single register

0x10Write Multiple register(CAN bus not support)

Notice:

Modbus-RTU Over UDP/TCP,Modbus-ASCII Over UDP/TCP use RS485 addr

Dingtian IOT Relay

Menu

Setting

Relay Connect

Relay Control

Relay Task

Input Status

Input Link Relay

Input Link URL

Input Relay Alias

IP WatchDog

Transparent Transmission

Reset User

To Factory

Upgrade

Reboot

Logout

Relay

Channel	Protocol	Addr	Baud	Databits	Stopbits	Parity
RS485	Modbus-RTU	1	115200bps	8bit	1bit	None
CAN	Dingtian String	1	125Kbps	Standard Frame		
UDP1	Dingtian Binary	Remote Address	Remote Port	Local Port		
UDP2	Dingtian String	Remote Address	Remote Port	Local Port		
TCP Server	Modbus-TCP			Local Port		
TCP Client	Modbus-RTU Over TCP	Remote Address	Remote Port			
MQTT	MQTT	Broker Address	Broker Port	Broker Username	Broker Password	

Head slash("/")

TL S

QoS 1(At least once)

Retain

KeepAlive 120 S

LWT KeepAlive 120 S

MFR dingtian

Area

HA Discover

HA Clear

Other

Relay Password

Keep Alive Second

Power Failure Recovery Relay

Save

Relay Test

R1:On

R2:Off

4.1 Registers

Register	Name	0x03/0x06/0x10	Value
0x0000	Relay Count	0x03	2/4/8/16/32
0x0001	Relay Status	0x03	bit0~7 map to relay1~8
0x0002	Write Relay	0x06	bit0~7 new status of relay1~8(bit=1 ON,bit=0 OFF) bit8~15 map to relay1~8 need update(bit=1 Update)
0x0003	Advance Write Type	0x10	Bit0~5: 1:Write ON/OFF 2:Write with delay 3:Write with Jogging bit6~15:(only for Type:Write ON/OFF(1)) relay group:0~3 r1~8:G0 r9~16:G1 r17~24:G2 r25~32:G3
0x0004	Advance Write Password	0x10	Password 0~65535 when password in current do nothing
0x0005	Advance Write Relay	0x10	Type:Write ON/OFF(1) bit0~7 new status of relay1~8(bit=1 ON,bit=0 OFF) bit8~15 map to relay1~8 need update(bit=1 Update) Type:Write with delay(2) bit0: bit=1 ON,bit=0 OFF bit1~7:relay index 0~31 Type:Write with Jogging(3) bit0: bit=1 ON,bit=0 OFF bit1~7:relay index 0~31
0x0006	Advance Write Time	0x10	Type:Write ON/OFF(1) 0 Type:Write with delay(2) Number of Second need delay Type:Write with Jogging(3) Number of 100ms need jogging(1=100ms)
0x0007	Expand Write Status Group	0x10	relay1~16:G0 relay16~32:G1
0x0008	Expand Write Relay Mask	0x10	bit0~15 map to relay G0:R1~16 / G1:R17~32 need update(bit=1 Update)
0x0009	Expand Write Relay	0x10	bit0~15 map to relay G0:R1~16 / G1:R17~32
0x000A	Expand Input Status 1~16	0x03	input1~16
0x000B	Expand Input Status 17~32	0x03	input17~32
0x000C	Expand Input Status 33~48	0x03	input33~48
0x000D	Expand Input Status 49~64	0x03	input49~64
0x000E	Expand Relay Status 1~16	0x03	relay1~16

0x000F	Expand Relay Status 17~32	0x03	relay17~32
0x0010	Expand Relay Status 33~48	0x03	relay33~48
0x0011	Expand Relay Status 49~64	0x03	relay49~64
0x0012	Expand Write Relay Mask 1~16	0x10	relay1~16 mask bits; bit=1 need change,bit=0 no change
0x0013	Expand Write Relay mask 17~32	0x10	relay17~32 mask bits; bit=1 need change,bit=0 no change
0x0014	Expand Write Relay bits 1~16	0x10	relay1~16 relay bits; bit=1(ON),bit=0(OFF)
0x0015	Expand Write Relay bits 17~32	0x10	relay17~32 relay bits, bit=1(ON),bit=0(OFF)
0x0016 to 0x0035	Input status one input one register max 32 inputs	0x03	0x0016 map to input1 0x0017 map to input2 ... 0x0035 map to input32 value 0: LOW 1: HIGH
0x0036 to 0x0055	relay status one relay one register max 32 relays	0x03/0x06/0x10	0x0036 map to relay1 0x0037 map to relay2 ... 0x0055 map to relay32 value 0: OFF 1: ON

Notice:

1、0x0003~6/0x0007~9/0x0012~15 is block , must written at the same time.

4.2 Modbus-RTU + Modbus-RTU Over TCP/UDP

4.2.1 0x03:Read holding register

Read all Relay Status

Send:

01 03 0000 0002 C40B

Recv:

01 03 04 0004 0000 BBF2

Read all Input Status(2/4/8 channel)

Send:

01 03 000A 0001 A408

Recv:

01 03 02 FFFF B9F4

Read all Input Status(support 16/24/32 channel)

Send:

01 03 000A 0002 E409

Recv:

01 03 04 FFFF 00FF BA57

Read all Relay Status(2/4/8 channel)

Send:

01 03 000E 0001 E5C9

Recv:

01 03 02 0000 B844

Read all Relay Status(support 16/24/32 channel)

Send:

01 03 000E 0002 A5C8

Recv:

01 03 04 0000 0080 FB93

Read all input and Relay Status(support 16/24/32 channel)

Send:

01 03 0016 0040 A5FE

Recv:

01 03 04 0000 0080 FB93

////////////////one register one relay/input example////////////////////////////////////

0x16~0x35 map to input1~32

0x36~0x55 map to relay1~32

01 03 0016 0040 A5FE # read input1~32 and relay1~32
01 03 0016 0020 A5D6 # read input1~32
01 03 0036 0020 A41C # read relay1~32

4.2.2 0x06:Write Single Register

4 Relay All ON

Send:

01 06 0002 0f0f 6DFE

Recv:

01 06 0002 0f0f 6DFE

4 Relay All OFF

Send:

01 06 0002 0f00 2DFA

Recv:

01 06 0002 0f00 2DFA

Relay 1,4 ON; Relay 2,3 stay the same

Send:

01 06 0002 0909 EE5C

Recv:

01 06 0002 0909 EE5C

4.2.2.1 one register one relay write

0x36~0x55 map to relay1~32

01 06 0036 0001 A804 # relay1 on

01 06 0037 0001 F9C4 # relay2 on

...

01 06 003d 0001 D9C6 # relay8 on

...

01 06 0055 0001 581A # relay32 on

4.2.3 0x10: Write Multiple Register

1、ON/OFF

4 Relay All ON

Send:

01 10 0003 0004 08 0001 0000 0f0f 0000 91A9

Recv:

01 10 0003 0004 31 CA

4 Relay All OFF

Send:

01 10 0003 0004 08 0001 0000 0f00 0000 A1AA

Recv:

01 10 0003 0004 31 CA

Relay 2,3 ON; Relay 1,4 stay the same

Send:

01 10 0003 0004 08 0001 0000 0606 0000 4237

Recv:

01 10 0003 0004 31 CA

2、Delay

Relay 1 OFF Delay 5 Second ON

Send:

01 10 0003 0004 08 0002 0000 0000 0005 51BD

Recv:

01 10 0003 0004 31 CA

Relay 1 ON Delay 5 Second OFF

Send:

01 10 0003 0004 08 0002 0000 0001 0005 007D

Recv:

01 10 0003 0004 31 CA

Relay 2 ON Delay 5 Second OFF

Send:

01 10 0003 0004 08 0002 0000 0003 0005 A1BD

Recv:

01 10 0003 0004 31 CA

Relay 3 ON Delay 5 Second OFF

Send:

01 10 0003 0004 08 0002 0000 0005 0005 41BC

Recv:

01 10 0003 0004 31 CA

Relay 4 ON Delay 5 Second OFF

Send:

01 10 0003 0004 08 0002 0000 0007 0005 E07C

Recv:

01 10 0003 0004 31 CA

3、Jogging

Relay 4 ON Joging 500ms OFF, Password 0x1234

Send:

01 10 0003 0004 08 0003 1234 0007 0005 420A

Recv:

01 10 0003 0004 31 CA

Relay 1 OFF Joging 500ms ON

Send:

01 10 0003 0004 08 0003 0000 0000 0005 417D

Recv:

01 10 0003 0004 31 CA

Relay 1 ON Joging 500ms OFF

Send:

01 10 0003 0004 08 0003 0000 0001 0005 10BD

Recv:

01 10 0003 0004 31 CA

Relay 2 ON Joging 500ms OFF

Send:

01 10 0003 0004 08 0003 0000 0003 0005 B17D

Recv:

01 10 0003 0004 31 CA

Relay 3 ON Joging 500ms OFF

Send:

01 10 0003 0004 08 0003 0000 0005 0005 517C

Recv:

01 10 0003 0004 31 CA

Relay 4 ON Joging 500ms OFF

Send:

01 10 0003 0004 08 0003 0000 0007 0005 F0BC

Recv:

01 10 0003 0004 31 CA

4 16/24/32 channel write relay

8 channel all relay ON

Send:

01 10 0012 0004 08 00FF 0000 00FF 0000 B17D

Recv:

01 10 0012 0004 61CF

8 channel all relay OFF

Send:

01 10 0012 0004 08 00FF 0000 0000 0000 814D

Recv:

01 10 0012 0004 61CF

16 channel all relay ON

Send:

01 10 0012 0004 08 FFFF 0000 FFFF 0000 CE6D

Recv:

01 10 0012 0004 61CF

16 channel all relay OFF

Send:

01 10 0012 0004 08 FFFF 0000 0000 0000 CE49

Recv:

01 10 0012 0004 61CF

24 channel all relay ON

Send:

01 10 0012 0004 08 FFFF 00FF FFFF 00FF 9A39

Recv:

01 10 0012 0004 61CF

24 channel all relay OFF

Send:

01 10 0012 0004 08 FFFF 00FF 0000 0000 DA5D

Recv:

01 10 0012 0004 61CF

32 channel all relay ON

Send:

01 10 0012 0004 08 FFFF FFFF FFFF FFFF CFC6

Recv:

01 10 0012 0004 61CF

32 channel all relay OFF

Send:

01 10 0012 0004 08 FFFF FFFF 0000 0000 CE52

Recv:

01 10 0012 0004 61CF

5 one register one relay/input write multiple

```
01 10 0036 0008 10 0001 0000 0001 0000 0001 0000 0001 0000 50A3 # relay1,3,5,7 on,relay
2,4,6,8 off
01 10 0036 0008 10 0001 0001 0001 0001 0001 0001 0001 0001 A3B2 # relay1~8 on
01 10 0036 0008 10 0000 0000 0000 0000 0000 0000 0000 0000 D45F # relay1~8 off
01 10 003d 0001 02 0001 637D # relay8 on
01 10 003d 0001 02 0000 A2BD # relay8 off
01 10 0055 0001 02 0001 6B95 # relay32 on
```

4.3 Modbus-TCP/UDP

4.3.1 0x03:Read holding register

Read all Relay Status

Send:

0000 0000 0006 FF 03 0000 0002

Recv:

0000 0000 0007 FF 03 04 0004 000F

Read all Relay Status(Extend support 16/24/32 channel relay board)

Send:

0000 0000 0006 FF 03 0000 0002

Recv:

0000 0000 0007 FF 03 04 0004 000F

Read all Input Status(2/4/8 channel)

Send:

0000 0000 0006 FF 03 000A 0001

Recv:

0000 0000 0005 FF 03 02 FFFF

Read all Input Status(support 16/24/32 channel)

Send:

0000 0000 0006 FF 03 000A 0002

Recv:

0000 0000 0007 FF 03 04 FFFF 00FF

Read all Relay Status(2/4/8 channel)

Send:

0000 0000 0006 FF 03 000E 0001

Recv:

0000 0000 0005 FF 03 02 0000

Read all Relay Status(support 16/24/32 channel)

Send:

0000 0000 0006 FF 03 000E 0002

Recv:

0000 0000 0007 FF 03 04 0000 0000

////////////////one register one relay/input////////////////

0x16~0x35 map to input1~32

0x36~0x55 map to relay1~32

0001 0000 0008 FF 03 0016 0040 # read input1~32 and relay1~32
0001 0000 0008 FF 03 0016 0020 # read input1~32
0001 0000 0008 FF 03 0036 0020 # read relay1~32

4.3.2 0x06:Write Single Register

4 Relay All ON

Send:

0000 0000 0006 FF 06 0002 0f0f

Recv:

0000 0000 0006 FF 06 0002 0f0f

4 Relay All OFF

Send:

0000 0000 0006 FF 06 0002 0f00

Recv:

01 06 0002 0f00 2DFA

Relay 1,4 ON; Relay 2,3 stay the same

Send:

0000 0000 0006 FF 06 0002 0909

Recv:

0000 0000 0006 FF 06 0002 0909

////////////////one register one relay////////////////////////

0x36~0x55 map to relay1~32

0001 0000 0006 FF 06 0036 0001 # relay1 on

0001 0000 0006 FF 06 0037 0001 # relay2 on

...

0001 0000 0006 FF 06 003d 0001 # relay8 on

...

0001 0000 0006 FF 06 0055 0001 # relay32 on

0001 0000 0006 FF 06 0036 0000 # relay1 off

0001 0000 0006 FF 06 0037 0000 # relay2 off

...

0001 0000 0006 FF 06 003d 0000 # relay8 off

...

0001 0000 0006 FF 06 0055 0000 # relay32 off

4.3.3 0x10: Write Multiple Register

1 ON/OFF

4 Relay All ON

Send:

0001 0000 000F FF 10 0003 0004 08 0001 0000 0f0f 0000

Recv:

0001 0000 0006 FF 10 0003 0004

4 Relay All OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0001 0000 0f00 0000

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 2,3 ON; Relay 1,4 stay the same

Send:

0001 0000 000F FF 10 0003 0004 08 0001 0000 0606 0000

Recv:

0001 0000 0006 FF 10 0003 0004

2 Delay

Relay 1 OFF Delay 5 Second ON

Send:

0001 0000 000F FF 10 0003 0004 08 0002 0000 0000 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 1 ON Delay 5 Second OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0002 0000 0001 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 2 ON Delay 5 Second OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0002 0000 0003 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 3 ON Delay 5 Second OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0002 0000 0005 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 4 ON Delay 5 Second OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0002 0000 0007 0005

Recv:

0001 0000 0006 FF 10 0003 0004

3 Jogging

Relay 4 ON Joging 500ms OFF, Password 0x1234

Send:

0001 0000 000F FF 10 0003 0004 08 0003 1234 0007 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 1 OFF Joging 500ms ON

Send:

0001 0000 000F FF 10 0003 0004 08 0003 0000 0000 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 1 ON Joging 500ms OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0003 0000 0001 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 2 ON Joging 500ms OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0003 0000 0003 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 3 ON Joging 500ms OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0003 0000 0005 0005

Recv:

0001 0000 0006 FF 10 0003 0004

Relay 4 ON Joging 500ms OFF

Send:

0001 0000 000F FF 10 0003 0004 08 0003 0000 0007 0005

Recv:

0001 0000 0006 FF 10 0003 0004

4 16/24/32 channel write relay

8 channel all relay ON

Send:

0001 0000 000F FF 10 0012 0004 08 00FF 0000 00FF 0000

Recv:

0001 0000 0006 FF 10 0012 0004

8 channel all relay OFF

Send:

0001 0000 000F FF 10 0012 0004 08 00FF 0000 0000 0000

Recv:

0001 0000 0006 FF 10 0012 0004

16 channel all relay ON

Send:

0001 0000 000F FF 10 0012 0004 08 FFFF 0000 FFFF 0000

Recv:

0001 0000 0006 FF 10 0012 0004

16 channel all relay OFF

Send:

0001 0000 000F FF 10 0012 0004 08 FFFF 0000 0000 0000

Recv:

0001 0000 0006 FF 10 0012 0004

24 channel all relay ON

Send:

0001 0000 000F FF 10 0012 0004 08 FFFF 00FF FFFF 00FF

Recv:

0001 0000 0006 FF 10 0012 0004

24 channel all relay OFF

Send:

0001 0000 000F FF 10 0012 0004 08 FFFF 00FF 0000 0000

Recv:

0001 0000 0006 FF 10 0012 0004

32 channel all relay ON

Send:

0001 0000 000F FF 10 0012 0004 08 FFFF FFFF FFFF FFFF

Recv:

0001 0000 0006 FF 10 0012 0004

32 channel all relay OFF

Send:

0001 0000 000F FF 10 0012 0004 08 FFFF FFFF 0000 0000

Recv:

0001 0000 0006 FF 10 0012 0004

5 one register one relay

0x36~0x55 map to relay1~32

0001 0000 0017 FF 10 0036 0008 10 0001 0000 0001 0000 0001 0000 0001 0000 # relay1,3,5,7
on,relay 2,4,6,8 off

0001 0000 0017 FF 10 0036 0008 10 0001 0001 0001 0001 0001 0001 0001 0001 # relay1~8 on

0001 0000 0017 FF 10 0036 0008 10 0000 0000 0000 0000 0000 0000 0000 0000 # relay1~8 off

0001 0000 0009 FF 10 0036 0001 02 0001 # relay1 on

0001 0000 0009 FF 10 0036 0001 02 0000 # relay1 off

0001 0000 0009 FF 10 003d 0001 02 0001 # relay8 on

0001 0000 0009 FF 10 003d 0001 02 0000 # relay8 off

0001 0000 0009 FF 10 0055 0001 02 0001 # relay32 on

4.4 Modbus-ASCII + Modbus-ASCII Over TCP/UDP

4.4.1 0x03:Read holding register

Read all Relay Status

Send:

ASCII : 01 03 0000 0002 BA \r\n

HEX 3A 3031 3033 30303030 30303032 4241 0D0A

Recv:

ASCII : 01 03 04 0004 0000 54 \r\n

HEX 3A 3031 3033 3034 30303034 30303030 3534 0D0A

Read all Input Status(2/4/8 channel)

Send:

ASCII : 01 03 000A 0001 AA \r\n

HEX 3A 3031 3033 30303041 30303031 4141 0D0A

Recv:

ASCII : 01 03 02 FFFF C2 \r\n

HEX 3A 3031 3033 3032 46464646 4332 0D0A

Read all Input Status(support 16/24/32 channel)

Send:

ASCII : 01 03 000A 0002 A9 \r\n

HEX 3A 3031 3033 30303041 30303032 4139 0D0A

Recv:

ASCII : 01 03 04 FFFF 00FF D4 \r\n

HEX 3A 3031 3033 3034 46464646 30304646 4434 0D0A

Read all Relay Status(2/4/8 channel)

Send:

ASCII : 01 03 000E 0001 A6 \r\n

HEX 3A 3031 3033 30303045 30303031 4136 0D0A

Recv:

ASCII : 01 03 02 0000 1A \r\n

HEX 3A 3031 3033 3032 30303030 3141 0D0A

Read all Relay Status(support 16/24/32 channel)

Send:

ASCII : 01 03 000E 0002 A5 \r\n

HEX 3A 3031 3033 30303045 30303032 4135 0D0A

Recv:

ASCII : 01 03 04 0000 0080 FB93 \r\n

HEX 3A 3031 3033 3034 30303030 30303830 3530 0D0A

4.4.2 0x06:Write Single Register

4 Relay All ON

Send:

ASCII : 01 06 0002 0F0F 8B \r\n

HEX 3A 3031 3036 30303032 30463046 3842 0D0A

Recv:

ASCII : 01 06 0002 0F0F 8B \r\n

HEX 3A 3031 3036 30303032 30463046 3842 0D0A

4 Relay All OFF

Send:

ASCII : 01 06 0002 0F00 A1 \r\n

HEX 3A 3031 3036 30303032 30463030 4131 0D0A

Recv:

ASCII : 01 06 0002 0F00 A1 \r\n

HEX 3A 3031 3036 30303032 30463030 4131 0D0A

4.4.3 0x10: Write Multiple Register

1 ON/OFF

4 Relay All ON

Send:

ASCII :01 10 0003 0004 08 0001 0000 0F0F 0000 22 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303031 30303030 30463046 30303030
3232 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

4 Relay All OFF

Send:

ASCII :01 10 0003 0004 08 0001 0000 0F00 0000 38 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303031 30303030 30463030 30303030
3338 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

Relay 2,3 ON; Relay 1,4 stay the same

Send:

ASCII :01 10 0003 0004 08 0001 0000 0606 0000 42 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303031 30303030 30363036 30303030
3432 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

2 Delay

Relay 1 ON Delay 5 Second OFF

Send:

ASCII :01 10 0003 0004 08 0002 0000 0001 0005 47 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303032 30303030 30303031 30303035
3437 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

Relay 4 ON Delay 5 Second OFF

Send:

ASCII :01 10 0003 0004 08 0002 0000 0007 0005 41 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303032 30303030 30303037 30303035
3431 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

3 Jogging

Relay 4 ON Joging 500ms OFF, Password 0x1234

Send:

ASCII :01 10 0003 0004 08 0003 1234 0007 0005 36 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303033 31323334 30303037 30303035
3336 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

Relay 1 ON Joging 500ms OFF

Send:

ASCII :01 10 0003 0004 08 0003 0000 0001 0005 46 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303033 30303030 30303031 30303035
3436 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

Relay 4 ON Joging 500ms OFF

Send:

ASCII :01 10 0003 0004 08 0003 0000 0007 0005 40 \r\n

HEX 3A 3031 3130 30303033 30303034 3038 30303033 30303030 30303037 30303035
3430 0D0A

Recv:

ASCII :01 10 0003 0004 B7 \r\n

HEX 3A 3031 3130 30303033 30303034 4237 0D0A

4 16/24/32 channel write relay

8 channel all relay ON

Send:

ASCII :01 10 0012 0004 08 00FF 0000 00FF 0000 F7 \r\n

HEX 3A 3031 3130 30303132 30303034 3038 30304646 30303030 30304646 30303030
4637 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

8 channel all relay OFF

Send:

ASCII :01 10 0012 0004 08 00FF 0000 0000 0000 23 \r\n

HEX 3A 3031 3130 30303132 30303034 3038 30304646 30303030 30303030 30303030
3233 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

16 channel all relay ON

Send:

ASCII :01 10 0012 0004 08 FFFF 0000 FFFF 0000 9F \r\n

HEX 3A 3031 3130 30303132 30303034 3038 46464646 30303030 46464646 30303030
3946 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

16 channel all relay OFF

Send:

ASCII :01 10 0012 0004 08 FFFF 0000 0000 0000 F7 \r\n

HEX 3A 3031 3130 30303132 30303034 3038 46464646 30303030 30303030 30303030
4637 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

24 channel all relay ON

Send:

ASCII :01 10 0012 0004 08 FFFF 00FF FFFF 00FF \r\n

HEX 3A 3031 3130 30303132 30303034 3038 46464646 30304646 46464646 30304646
3437 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

24 channel all relay OFF

Send:

ASCII :01 10 0012 0004 08 FFFF 00FF 0000 0000 CB \r\n

HEX 3A 3031 3130 30303132 30303034 3038 46464646 30304646 30303030 30303030
4342 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

32 channel all relay ON

Send:

ASCII :01 10 0012 0004 08 FFFF FFFF FFFF FFFF CFC6 \r\n

HEX 3A 3031 3130 30303132 30303034 3038 46464646 46464646 46464646 46464646
4546 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

32 channel all relay OFF

Send:

ASCII :01 10 0012 0004 08 FFFF FFFF 0000 0000 CE52 \r\n

HEX 3A 3031 3130 30303132 30303034 3038 46464646 46464646 30303030 30303030
3946 0D0A

Recv:

ASCII :01 10 0012 0004 B7 \r\n

HEX 3A 3031 3130 30303132 30303034 4237 0D0A

5 Protocol:MQTT

MQTT version 3.1.1

Relay board as MQTT client,communcation with broker..

Support relay ON/OFF

Support relay JOGGING(PULSE)

Support relay DELAY

Support password verification

Support LWT

Dingtian IOT Relay

Setting

Menu

Setting

Relay Connect

Relay Control

Relay Task

Input Status

Input Link Relay

Input Link URL

Input Relay Alias

IP WatchDog

Transparent Transmission

Reset User

To Factory

Upgrade

Reboot

Logout

Hardware Version

V3.6J

Software Version

V3.1.3392A

Build Date

2024/01/26 0:21:17

Model

Dingtian DT-R002

Serial Number

33630

Date Time

1970/1/1 09:39:40

Extend Module

Internal RS485

Auto Reboot Every Day

No

0

H

0

M

0

S

NTP Server

pool.ntp.org

Hostname

Dingtian-Relay33630

Hostname+Suffix

Dingtian-Relay

SN

HTTP or HTTPS

HTTP

HTTP Server Port

80

HTTPS Server Port

443

HTTP Session

No

HTTP Magic Session ID

12345678

Ethernet/WiFi Priority

Ethernet

Must reboot

ETH DHCP

No

ETH IP

192.168.1.100

ETH Netmask

255.255.255.0

ETH Gateway

192.168.1.1

ETH DNS

192.168.1.1

0.0.0.0 use DHCP DNS

ETH MAC

ba:34:88:00:82:af

WiFi Enable

No

WiFi Auth

WPA2 PSK

WiFi DHCP

No

WiFi IP

192.168.1.97

WiFi Netmask

255.255.255.0

WiFi Gateway

192.168.1.1

WiFi DNS

192.168.1.1

0.0.0.0 use DHCP DNS

WiFi MAC

bc:34:88:00:82:af

WiFi SSID

WiFi Password

AP Enable

Yes

AP Hide

No

AP IP

192.168.7.1

AP Netmask

255.255.255.0

AP Gateway

192.168.7.1

AP DNS

192.168.7.1

AP MAC

be:34:88:00:82:af

AP SSID

dtrelay33630

AP Password

Save

Relay board Ethernet MQTT Client Id

dingtian-relay+SN

example:

below relay board "Serial Number" is 33630

so MQTT client id is:dingtian-relay33630

5.4 MQTT Topic

topic	type	value		
/dingtian/relaySN/in/control example: /dingtian/relay33630/in/control	subscribe	parameter	filed	data
		type	command type	ON/OFF DELAY JOGGING(PULSE) FLASH TOGGLE
		idx	relay index	1~32
		status	relay status	ON,OFF
		time	time for type	ON/OFF:0 DELAY:1~65535second JOGGING:1~255*100ms FLASH:1~255*100ms TOGGLE:0
		pass	password	0~9999
		example: { "type": "ON/OFF", "idx": "1", "status": "ON", "time": "0", "pass": "0" } { "type": "DELAY", "idx": "2", "status": "ON", "time": "5", "pass": "0" } { "type": "JOGGING", "idx": "3", "status": "ON", "time": "5", "pass": "0" } { "type": "ON/OFF", "idx": "4", "status": "OFF", "time": "0", "pass": "0" } { "type": " FLASH", "idx": "6", "status": "ON", "time": "5", "pass": "0" } { "type": " TOGGLE ", "idx": "7", "status": "ON", "time": "0", "pass": "0" }		
/dingtian/relaySN/in/rX example: /dingtian/relay33630/in/r1 /dingtian/relay33630/in/r2	subscribe	X:1~32 value: ON,OFF,1,0,TRUE,FALSE		
/dingtian/relaySN/out/rX example: /dingtian/relay33630/out/r1 /dingtian/relay33630/out/r2	publish	X:1~32 value: ON,OFF		
/dingtian/relaySN/out/iX	publish	X:1~32 value: ON,OFF		

example: /dingtian/relay33630/out/i1 /dingtian/relay33630/out/i2				
example: /dingtian/relay33630/out/relay1 /dingtian/relay33630/out/relay2	publish	parameter	filed	data
		idx	relay index	1~32
		status	relay status	ON,OFF
		example: {"idx": "1", "status": "OFF"}		
example: /dingtian/relay33630/out/input1	publish	parameter	filed	data
		idx	relay index	1~32
		status	relay status	HIGH,LOW
		example: {"idx": "1", "status": "HIGH"} {"idx": "1", "status": "LOW"}		
example: /dingtian/relay33630/out/ip	publish	example: 192.168.1.100		
example: /dingtian/relay33630/out/sn	publish	example: 33630		
example: /dingtian/relay33630/out/mac	publish	example: bc:34:88:00:00:00		
example: /dingtian/relay33630/out/input_cnt	publish	2,4,8,16,32		
example: /dingtian/relay33630/out/relay_cnt	publish	2,4,8,16,32		
example /dingtian/relay33630/out/ lwt_availability	publish	online,offline		

6 Protocol:CoAP

Relay board as CoAP server, accept CoAP Client request.

Support relay on/off

Support relay jogging

Support relay delay

Support password verification

you need linux system to compile libcoap-v4.3.1

6.1 Compile libcoap

```
git clone --recurse-submodules https://github.com/obgm/libcoap
./autogen.sh
./configure --disable-manpages --enable-examples --enable-tests
make
```

6.2 Get relay status

Relay Status(1:ON, 0:OFF)

```
./coap-client -m get coap://192.168.1.100/dingtian/r1
./coap-client -m get coap://192.168.1.100/dingtian/r2
./coap-client -m get coap://192.168.1.100/dingtian/r3
./coap-client -m get coap://192.168.1.100/dingtian/r4
./coap-client -m get coap://192.168.1.100/dingtian/r5
./coap-client -m get coap://192.168.1.100/dingtian/r6
./coap-client -m get coap://192.168.1.100/dingtian/r7
./coap-client -m get coap://192.168.1.100/dingtian/r8
...
./coap-client -m get coap://192.168.1.100/dingtian/r32
```

Input Status(1:High, 0:Low)

```
./coap-client -m get coap://192.168.1.100/dingtian/i1
./coap-client -m get coap://192.168.1.100/dingtian/i2
./coap-client -m get coap://192.168.1.100/dingtian/i3
./coap-client -m get coap://192.168.1.100/dingtian/i4
./coap-client -m get coap://192.168.1.100/dingtian/i5
./coap-client -m get coap://192.168.1.100/dingtian/i6
./coap-client -m get coap://192.168.1.100/dingtian/i7
./coap-client -m get coap://192.168.1.100/dingtian/i8
...
./coap-client -m get coap://192.168.1.100/dingtian/i32
```

6.3 Control relay(simple)

```
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r1 # relay1 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r1 # relay1 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r2 # relay2 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r2 # relay2 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r3 # relay3 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r3 # relay3 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r4 # relay4 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r4 # relay4 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r5 # relay5 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r5 # relay5 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r6 # relay6 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r6 # relay6 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r7 # relay7 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r7 # relay7 OFF
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r8 # relay8 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r8 # relay8 OFF
...
./coap-client -e "1" -m put coap://192.168.1.100/dingtian/r32 # relay32 ON
./coap-client -e "0" -m put coap://192.168.1.100/dingtian/r32 # relay32 OFF
```

6.4 Control relay

format:

status:type:time:password

parameter	filed	data	comment
status	relay status	0,1	
type	ON/OFF DELAY JOGGING		
time	time for type	ON/OFF:0 DELAY:1~65535second JOGGING:1~255*100m s	
password	password	0~9999	

example:

1:ON/OFF:0:4660

status:1

type:ON/OFF

time:0

password:4660

ON/OFF example:

```
./coap-client -e "1:ON/OFF:0:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 ON
```

```
./coap-client -e "1:ON/OFF:0:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 ON
```

```
./coap-client -e "0:ON/OFF:0:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 OFF
```

```
./coap-client -e "0:ON/OFF:0:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 OFF
```

...

```
./coap-client -e "0:ON/OFF:0:4660" -m put coap://192.168.1.100/dingtian/r32 # relay32 OFF
```

DELAY example:

```
./coap-client -e "1:DELAY:5:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY ON 500ms
```

```
./coap-client -e "1:DELAY:0:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY stop
```

```
./coap-client -e "1:DELAY:5:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY ON 500ms
```

```
./coap-client -e "1:DELAY:0:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY stop
```

```
./coap-client -e "0:DELAY:5:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY OFF 500ms
```

```
./coap-client -e "0:DELAY:0:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY stop
```

```
./coap-client -e "0:DELAY:5:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY OFF 500ms
```

```
./coap-client -e "0:DELAY:0:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY stop
```

...

```
./coap-client -e "0:DELAY:5:4660" -m put coap://192.168.1.100/dingtian/r32 # relay32 DELAY OFF 500ms
```

JOGGING example:

```
./coap-client -e "1:JOGGING:5:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY ON 500ms
```

```
./coap-client -e "1:JOGGING:0:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY stop
./coap-client -e "1:JOGGING:5:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY ON 500ms
./coap-client -e "1:JOGGING:0:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY stop
./coap-client -e "0:JOGGING:5:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY OFF 500ms
./coap-client -e "0:JOGGING:0:4660" -m put coap://192.168.1.100/dingtian/r1 # relay1 DELAY stop
./coap-client -e "0:JOGGING:5:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY OFF 500ms
./coap-client -e "0:JOGGING:0:4660" -m put coap://192.168.1.100/dingtian/r2 # relay2 DELAY stop
...
./coap-client -e "0:JOGGING:5:4660" -m put coap://192.168.1.100/dingtian/r32 # relay32 DELAY OFF 500ms
```

TOGGLE example:

```
./coap-client -e "1:TOGGLE:0:4660" -m put coap://192.168.1.100/dingtian/r1 #relay1 toggle
./coap-client -e "1:TOGGLE:0:4660" -m put coap://192.168.1.100/dingtian/r2 #relay2 toggle
...
./coap-client -e "0:TOGGLE:0:4660" -m put coap://192.168.1.100/dingtian/r32 #relay32 toggle
```