Communication Protocol

General MODBUS-RTU communication protocol for counting, timing and frequency measurement products

Ⅰ. MODBUS-RTU Communication format

1. Basic Rules

1.1 Only one host is allowed in the same network.

1.2 All RS485 communication loops should follow the master/slave method for communication.

1.3 No communication can be initiated by the slave.

1.4. On the RS485 bus, all communications are transmitted in "information frames". "Information frame" is a character string composed of several "data frames". It is a standard asynchronous serial data composed of an information header and transmitted encoded data.

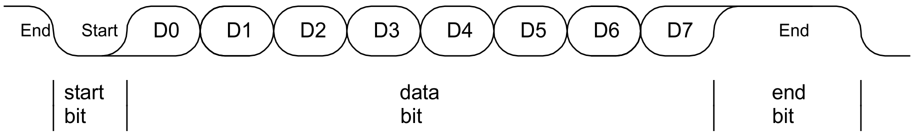
1.5. If the master and slave receive information frames containing unknown commands, they will not respond.

2. Transmission method

Communication is based on bytes (data frames) and is transmitted asynchronously.

3. "Data frame" format

Each "data frame" contains a start bit, 8 data bits, parity or no parity bit, and a stop bit, a total of 10 bits of data.



4."Information frame" format

Address code	Function code	Data area	CRC check code
1 byte	1 byte	N byte	2 byte(Low byte first and high byte last)

When the communication command is sent from the master to the slave, the slave that matches the table address sent by the master receives the command. If the CRC check is correct and the command format is correct, the slave performs the corresponding operation and then returns the execution result to the master.

4.1 address code (1 byte)

Included in the address field of the "Information Frame", the address range is 1-247. The master strobes the slave by putting the slave table address into the command's address field. When the slave returns data, it puts its own table address into the address field of the returned information, so that the master knows which slave has responded (the table address of each device in the same bus must be unique).

4.2 function code (1 byte)

Contained in the function code field of the "information frame". When sent from the master to the slave, the function code will tell the slave that those operations need to be performed. When the slave responds, the function code is used to indicate a normal response or an error occurs (abnormal response).

For a normal response, the slave only returns the received function code. For abnormal response, the slave will return the highest position of the received function code.

Function code definition

Function code	Definition	Operation
0x03	Read registers	Read data from single or multiple registers
0x10	Write multiple registers	Write n 32-bit binary data to n consecutive registers

4.3. Data Area

Included in the data field of the message, the data length varies depending on the function code.

4.4 CRC check code

The redundant cyclic code (CRC) contains 2 bytes, that is, 16-bit binary. The CRC code is calculated by the sending end and placed at the end of the sent information. The device at the receiving end recalculates the CRC code of the received information and compares the calculated CRC code with the received one. If the two do not match, it indicates an error.

The calculation method of the CRC code is to first preset all 16-bit registers. Then gradually process every 8-bit data information. When calculating the CRC code, only 8 data bits, start bit and stop bit are used. If there is a parity bit, it also includes the parity bit and does not participate in the CRC code calculation.

When calculating the CRC code, the 8-bit data and the data of the register are XORed, and the result is shifted to the lower one bit, and the highest bit is filled with 0. Check the lowest bit again. If the lowest bit is 1, XOR the contents of the register with the preset number. If the lowest bit is 0, no XOR operation is performed.

This process has been repeated 8 times. After the 8th shift, the next 8 bits are XORed with the contents of the current register again. This process is repeated 8 times as above. When all the data information is processed, the content of the last register is the CRC code value.

CRC-16 code calculation steps

4.4.1. Set the 16-bit register to hexadecimal FFFF (that is, all is 1). Call this register the CRC register.

4.4.2. XOR an 8-bit data with the lower bits of the 16-bit CRC register, and put the result in the CRC register.

4.4.3. Move the contents of the register one bit to the right (toward the low bit), fill the highest bit with 0, and check the lowest bit (shift out bit).

4.4.4. If the lowest bit is 0: repeat step 3 (shift again).

If the least significant bit is 1: the CRC register is XORed with the polynomial A001 (1010 0000 0000 0001).

4.4.5. Repeat steps 3 and 4 until shifting to the right 8 times, so that the entire 8-bit data has been processed.

4.4.6. Repeat Step 2 to Step 5 for the next 8-bit processing.

4.4.7. The resulting CRC register is the CRC code, with the low byte first and the high byte second.

Ⅱ. Command format of master and message format returned from slave

In order to support some hosts without 64-bit data type (such as some configuration software, PLC), the data in the address segment of 0x1000–0x105B has been enlarged by 2^{32} times. The purpose is to ensure the accuracy of the data and make the integer part and decimal part of the data can be processed separately.

2.1. read multiple registers

Example 1: Read count (timing) value (complete data, 64-bit data format)

1. If the current count value of the meter = 123.456789, the host sends a command to read the 4 registers starting at 0x1000, and the meter returns 0x7B74F01FB8
2. Divide 0x7B74F01FB8, which is 530242871224 decimal, 2^{32} = the current count value of the slave is 123.456789

Command format	Host sends commands		Communication data order		
			=1234	=4321	=2143
Address field	Table address		0x01		
Fucntion field	Function code		0x03		
Data field	Start register address	High byte	0x10		
		Low byte	0x00		
	Read registers qty	High byte	0x00		
		Low byte	0x04		
Error check field	CRC check code	Low byte	0x40		
		High byte	0xC9		

Message format	Slave return message		Communication data order		
			=1234	=2143	=4321
Address field	Table address		0x01		
Fucntion field	Function code		0x03		
Data field	Number of data bytes		0x08		
	Count (timer) value register	High high byte	0x00	0x00	0x1F
			0x00	0x7B	0xB8
		High byte	0x00	0x00	0x74
			0x7B	0x00	0xF0
		Low byte	0x74	0x1F	0x00
			0xF0	0xB8	0x7B
		Low low byte	0x1F	0x74	0x00
			0xB8	0xF0	0x00
Error check field	CRC check code	Low byte	0x62	0xFE	0xD6
		High byte	0x5C	0x65	0x28

Example 2: Read count (timing) value (read-only integer part, 32-bit data format)

1. Assuming the current count value of the slave = 19088743.568, read the integer part of the count value, and the slave returns data = 0x01234567.
2. When reading the integer part of a parameter separately, the returned data 0x01234567=19088743 is the current actual value of the slave (no need to divide by 2)^{32}

Command format	Host sends commands		Communication data order		
			=1234	=2143	=4321
Address field	Table address		0x01		
Fucntion field	Function code		0x03		
Data field	Start register address	High byte	0x10	0x10	
		Low byte	0x00	0x02	
	Number of read registers	High byte	0x00	0x00	
		Low byte	0x02	0x02	
Error check field	CRC check code	Low byte	0xC0	0x61	
		High byte	0xCB	0x0B	

Message format	Slave return message		Communication data order		
			=1234	=2143	=4321
Address field	Table address		0x01		
Fucntion field	Function code		0x03		
Data field	Number of data bytes		0x04		
	The integer part of the count (timer) value	High byte	0x01	0x45	0x45
			0x23	0x67	0x67
		Low byte	0x45	0x01	0x01
			0x67	0x23	0x23
Error check field	CRC check code	Low byte	0x79	0x1E	0x1E
		High byte	0x7F	0xA9	0xA9

2.2. Write multiple registers

Example 3: Write 12345.678 to the slave PS2 set value register

1. If the host supports the 64-bit data format, you can directly multiply 12345.678 by 2^{32}= 53024283256946, and then send it in hexadecimal format (53024283256946 = 0x00003039AD916872. A total of 8 bytes, fill in 0 in the upper bits when not enough)
2. If the host only supports the 32-bit data format, the integer part and decimal part of 12345.678 need to be processed separately.

2.1. The integer part does not need to be processed, directly put 12345 in hexadecimal format into the upper 4 bytes of the data to be sent (if there are not enough 4 bytes, fill in 0 in the high bit. 12345 = 0x00003039).

2.2. The fractional part of 0.678 needs to be multiplied by 2^{32}= 2911987826, and put in the lower 4 bytes of the data to be sent in hexadecimal format (if there are not enough 4 bytes, fill in 0 in the upper bits. 2911987826 = 0x AD916872).

2.3. Then send the processed 8 bytes of data in the order from high byte to low byte (1234) (0x00003039AD916872), or from low byte to high wbyte (4321) (0x6872AD9130390000)

Command format	Host sends commands		Communication data order		
			=1234	=2143	=4321
Address field	Table address		0x01		
Fucntion field	Function code		0x10		
Data field	Start register address	High byte	0x10		
		Low byte	0x30		
	Number of write registers	High byte	0x00		
		Low byte	0x04		
	Write data bytes		0x08		
	Ready to write PS2 set value registration data (64-bit data, high byte first and low byte last)	High high byte	0x00	0x30	0x68
			0x00	0x39	0x72
		High byte	0x30	0x00	0xAD
			0x39	0x00	0x91
		Low byte	0xAD	0x68	0x30
			0x91	0x72	0x39
		Low low byte	0x68	0xAD	0x00
			0x72	0x91	0x00
Error check field	CRC check code	Low byte	0x8F	0x63	0xA6
		High byte	0xFB	0xFA	0x4E

Message format	Slave return message		Communication data order		
			=1234	=2143	=4321
Address field	Table address		0x01		
Fucntion field	Function code		0x10		
Data field	Start register address	High byte	0x10		
		Low byte	0x30		
	Number of write registers	High byte	0x00		
		Low byte	0x04		
Error check field	CRC check code	Low byte	0xC5		
		High byte	0x05		

Ⅲ. Communication error handling

When the meter detects other errors than the CRC check code error, it will return an error message to the host. The slave will set the highest position of the received function code to 1, and then return it as an error message together with the table address and error code.

3.1 Slave return error code format

Address code	Function code (highest byte 1)	Error code	CRC check code low byte	CRC check code high byte
1 byte	1 byte	1 byte	1 byte	1 byte

3.2 Error codeIllegalIllegal function code function codeIllegal function code

0x01	Illegal function code	The meter does not support the received function code
0x02	Illegal register address	The received register address exceeds the address range of the meter's register
0x03	Illegal number of registers	The received register numbers exceeds the number of the meter's register
0x04	Illegal data value	The received data value exceeds the data range of the corresponding address

Ⅳ. Data and mapped address

4.1 The data of each parameter in the address segment of 0x1000–0x105B has been enlarged by a factor of 2^{32}. It needs to be multiplied by 2^{32} before writing and divided by 2^{32} when reading.

4.2 Each parameter in the address segment 0x1000–0x105B occupies 4 register addresses (4 words, 8 bytes), and the internal data is divided into 1234 (default, high byte first low byte last) , 4321 and 2143 (low byte first high byte last) are arranged in three order.

4.3. This agreement is a general communication protocol. Please refer to the corresponding product operation manual for whether the instrument has the functions in the agreement and the value range of the register.

No.	Data add	Parameter name	Data length	Data type	Attributes	Remarks
1	0x1000 0x1001 0x1002 0x1003	Counting (timer) value	4	Signed 64-bit integer	R/W	1. When writing, can only write 0, otherwise it returns an error 2. Timing mode, unit is second Example: Register value = 0xCE3D70A3D Actual time = 0xCE3D70A3D/2^{32} = 12.89 seconds
2	0x1004 0x1005 0x1006 0x1007	Batch or total value	4	Signed 64-bit integer	R/W	1. When writing, can only write 0, otherwise it returns an error
3	0x1008 0x1009 0x100a 0x100b	Frequency, speed, linear speed value	4	Signed 64-bit integer	R	
4	reserved					
5	0x1010 0x1011 0x1012 0x1013	Initial count value	4	Signed 64-bit integer	R/W	
6	0x1014 0x1015 0x1016 0x1017	Counting factor value	4	Signed 64-bit integer	R/W	
7	0x1018 0x1019 0x101a 0x101b	Linear speed or batch factor value	4	Signed 64-bit integer	R/W	
8	reserved					
9	0x1020 0x1021 0x1022 0x1023	PS1 count setting value	4	Signed 64-bit integer	R/W	
10	0x1024 0x1025 0x1026 0x1027	PS1 output delay time	4	Signed 64-bit integer	R/W	Unit: second
11	0x1028 0x1029 0x102a 0x102b	PS1 hysteresis	4	Signed 64-bit integer	R/W	
12	reserved					
13	0x1030 0x1031 0x1032 0x1033	PS2 count (timer) setting value	4	Signed 64-bit integer	R/W	1. In the timer mode, the unit is second, and its set value range is determined by the time setting parameter. For example: meter timing range = 99H59M59S99, then change the register's writable range = 0.01~35999999S
14	0x1034 0x1035 0x1036 0x1037	PS2 count (timer) output delay time	4	Signed 64-bit integer	R/W	Unit: second
15	0x1038 0x1039 0x103a 0x103b	PS2 hysteresis	4	Signed 64-bit integer	R/W	
16	reserved					
17	0x1040 0x1041 0x1042 0x1043	LSV setting value	4	Signed 64-bit integer	R/W	
18	0x1044 0x1045 0x1046 0x1047	LSV output delay time	4	Signed 64-bit integer	R/W	Unit: second
19	0x1048 0x1049 0x104a 0x104b	LSV hysteresis	4	Signed 64-bit integer	R/W	
20	reserved					
21	0x1050 0x1051 0x1052 0x1053	BAS setting value	4	Signed 64-bit integer	R/W	
22	0x1054 0x1055 0x1056 0x1057	BAS output delay time	4	Signed 64-bit integer	R/W	Unit: second
23	0x1058 0x1059 0x105a 0x105b	BAS hysteresis	4	Signed 64-bit integer	R/W	
	reserved					

4.4 Each parameter in the 0x1100--0x1164 address segment occupies 1 register address (1 word, 2 bytes), the data in the register is high byte first low byte last.

No.	Data Add	Parameter Name	Data Length	Data Type	Attributes	Remarks
24	0x1100	Communication address	1	Unsigned 16-bit integer	R/W	1~247
25	0x1101	Reserved				
26	0x1102	Reserved				
27	0x1103	Communication baud rate	1	Unsigned 16-bit integer	R/W	4800=4800bit/s 、9600=9600bit/s 、19200=192 00bit/s
28	0x1104	Communication verification method	1	Unsigned 16-bit integer	R/W	0 = no check, 1 = odd check, 2 = even check
29	0x1105	Communication data byte (register) sequence selection	1	Unsigned 16-bit integer	R/W	Example: When sending or receiving data 0x1020304050607080, The corresponding order of different settings as follows: =1234, the order of receiving and sending = 10 20 30 40 50 60 70 80; = 2143, the order of receiving and sending = 30 40 10 20 70 80 50 60; =4321, receiving and sending sequence = 70 80 50 60 30 40 10 20
30	0x1106	Batch/total accumulation method selection	1	Unsigned 16-bit integer	R/W	0 = accumulate by batch, 1 = accumulate by total
31	0x1107	Function selection	1	Unsigned 16-bit integer	R/W	0=count, 1=time, 2=frequency, 3=speed, 4=line speed
32	0x1108	Ascending or descending method selection	1	Unsigned 16-bit integer	R/W	0 = ascending , 1 = descending
33	0x1109	NPN, PNP selection	1	Unsigned 16-bit integer	R/W	0 = NPN, 1 = PNP
34	0x110a	Input type selection	1	Unsigned 16-bit integer	R/W	0=U, 1=D, 2=UD -A, 3=UD -B, 4=UD - C, 5=UD - D
35	0x110b	Input frequency selection	1	Unsigned 16-bit integer	R/W	1=1Hz ,30=30Hz ,1000=1KHz ,5000=5KHz , 10000=10KHz , 20000= 20KHz
36	0x110c	External signal width selection	1	Unsigned 16-bit integer	R/W	Actual pulse width, unit: ms
37	0x110d	Reserved				
38	0x110e	Reserved				
39	0x110f	Timing range selection	1	Unsigned 16-bit integer	R/W	0 = 999999s99 256 = 99h59m59s99 512 = 9999h59m59s Note: Non-time relay or timing mode, invalid write
40	0x1110	Delay range selection	1	Unsigned 16-bit integer	R/W	
41	0x1111	Reserved				
42	0x1112	Show decimal point selection	1	Unsigned 16-bit integer	R/W	0=no decimal point or floating decimal point, 1=1 decimal point, 2=2 decimal point, ...
43	0x1113	Display refresh time selection	1	Unsigned 16-bit integer	R/W	Unit (10ms): 0=auto refresh, 50=0.5 seconds, 100=1 second
44	0x1114	Reserved				
45	0x1115	Reserved				
46	0x1116	Count output mode selection	1	Unsigned 16-bit integer	R/W	0 = F, 1 = N, 2 = C, 3 = R 4 = K, 5 = P, 6 = Q, 7 = A 8 = S, 9 = T, 10 = D, 11 = M
47	0x1117	Timer output mode selection	1	Unsigned 16-bit integer	R/W	0 = OND, 1 = OND.1, 2 = OND.2, 3 = FLK 4 = FLK.1, 5 = FLK.2, 6 = INT, 7 = INT.1 8 = OFD
48	0x1118	SV1 output mode selection (reserved)	1	Unsigned 16-bit integer	R/W	
49	0x1119	SV2 output mode selection (reserved)	1	Unsigned 16-bit integer	R/W	
50	0x111a	SV3 output mode selection (reserved)	1	Unsigned 16-bit integer	R/W	
51	0x111b	LSV output mode selection (reserved)	1	Unsigned 16-bit integer	R/W	
52	0x111c	BSV output mode selection (reserved)	1	Unsigned 16-bit integer	R/W	
53	0x111d	Power failure memory function	1	Unsigned 16-bit integer	R/W	0 = OFF, 1=ON
54	0x111e	Start function	1	Unsigned 16-bit integer	R/W	0 = OFF, 1=ON
55	0x111f	Reserved				
56	0x1120	Reserved				
57	0x1121	Reserved				
58	0x1122	Password setting				
59	0x1160	OUT1 output status	1	Unsigned 16-bit integer	R	0 = no action, 1 = action
60	0x1161	OUT2 output status	1	Unsigned 16-bit integer	R	0 = no action, 1 = action
61	0x1162	OUT3 output status	1	Unsigned 16-bit integer	R	0 = no action, 1 = action
62	0x1163	LSO output status	1	Unsigned 16-bit integer	R	0 = no action, 1 = action
63	0x1164	BAO output status	1	Unsigned 16-bit integer	R	0 = no action, 1 = action