M350-Modbus Manual

V1.1

1. Introduction

The M350 supports the Modbus RTU communication protocol；

- Serial communication interface: RS-232 or RS-485;
- Baud rates: B2400, B4800, B9600, B19200, B115200;
- Master-slave mode: Default master mode;
- Supported function codes: 01H, 02H, 03H, 04H, 0FH, 10H.

2. System Parameter Settings

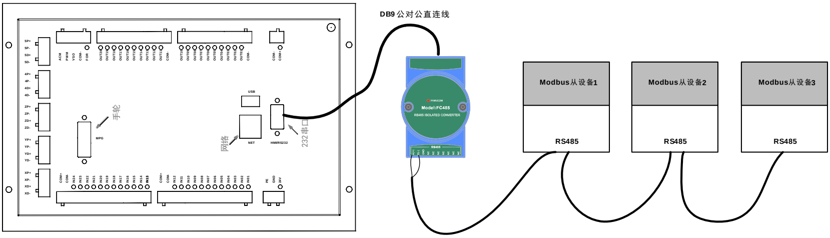
Parameter 279: Modbus RTU Enable

Parameter 267: Serial Port 2 Baud Rate B2400, B4800, B9600, B19200, B115200;

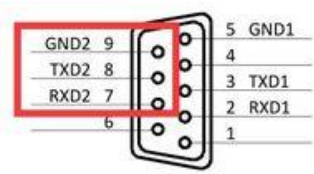
Default Data Format: 8 data bits, 1 stop bit, no parity bit;

Please restart the system after setting the parameters!

3. Wiring Diagram



Wiring Diagram for System and Multiple Slave Devices



Modbus uses the serial ports corresponding to pins 7, 8, and 9.

4. Modbus Macro Functions

4.1 Read Functions:MGETDATA[]

MGETDATA[X1,X2,X3,X4,X5,X6]

X1: 50-499, each byte read is stored in consecutive address space from #50 to #499 (Note: At this time, one macro address corresponds to one byte).

X2: Slave (SLAVE) station number.

X3: Starting address of the data to be read.

X4: Length of data in bytes to be read (Note: In Modbus, one register occupies 2 bytes).

X5: Data read mode.

Read Operation Function Code Allocation Table (RTU Mode)：

Function Code	X5
01H	1
02H	2
03H	3
04H	4

X6: 50-499, the exception code returned will be stored in the macro address X6 (#50-#499).

List of Exception Response Codes:

X6	Description
0x00	Normal
0x01	Invalid or unsupported function code
0x02	Invalid or unsupported address
0x03	Invalid or unsupported data
0x04	Action execution failed
0x05	Action execution in progress (may take a long time)
0x06	Device is busy and cannot perform the action at the moment
0x08	File data checksum error
0x0A	Invalid gateway route
0x0B	No response from the target device
0xE0	Transmission error or illegal Modbus data frame

0xFF	Timeout
0xe1	Undefined action

4.1.1 (01H) Read Coil

Example of Request to Read Discrete Output from 20-38:

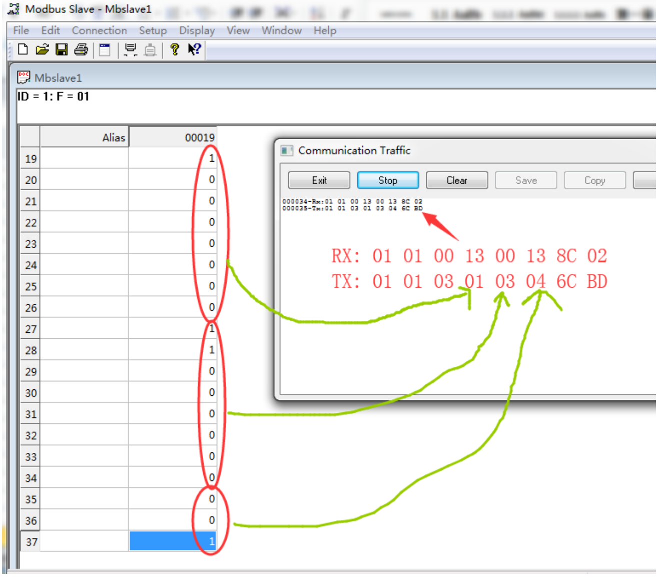
MGETDATA[200,1,19,19,1,300]

The data frame is as follows:

RX： 01 01 00 13 00 13 8C 02

TX： 01 01 03 01 03 04 6C BD

M350 Communication with Computer Virtual Slave:



ID = 1: F = 01

	Alias	00019
19		1
20		0
21		0
22		0
23		0
24		0
25		0
26		0
27		1
28		1
29		0
30		0
31		0
32		0
33		0
34		0
35		0
36		0
37		1

```
000034-Rx:01 01 00 13 00 13 8C 02
000035-Tx:01 01 03 01 03 04 6C BD
```

RX: 01 01 00 13 00 13 8C 02

TX: 01 01 03 01 03 04 6C BD

Request	
Domain Name	HEX
ID	01
Function	01
Starting Address Hi	00
Starting Address Lo	13
Output Quantity Hi	00
Output Quantity Lo	13
CRC Hi	8c
CRC Lo	02

Response	
Domain Name	HEX
ID	01
Function	01
Byte Count	03
Output Status 27-20 [#200]	01
Output Status 35-28 [#201]	03
Output Status 38-36 [#202]	04
CRC Hi	6C
CRC Lo	BD

4.1.2 (02H) Read Discrete Input

Example of Request to Read Discrete Input from 197-216:

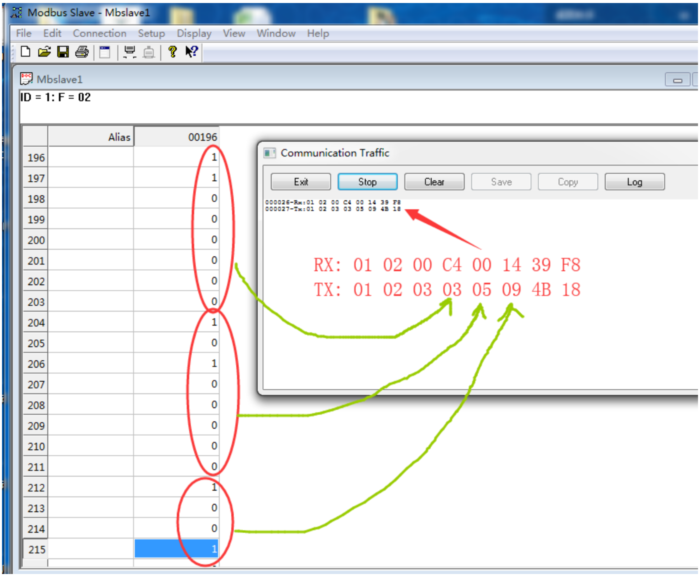
MGETDATA[200,1,196,20,2,300]

The data frame is as follows:

RX： 01 02 00 C4 00 14 39 F8

TX： 01 02 03 03 05 09 4B 18

M350 Communication with Computer Virtual Slave:



ID = 1: F = 02

	Alias	00196
196		1
197		1
198		0
199		0
200		0
201		0
202		0
203		0
204		1
205		0
206		1
207		0
208		0
209		0
210		0
211		0
212		1
213		0
214		0
215		1

```
000026-Rx:01 02 00 C4 00 14 39 F8
000027-Tx:01 02 03 03 05 09 4B 18
```

RX: 01 02 00 C4 00 14 39 F8
TX: 01 02 03 03 05 09 4B 18

Request	
Domain Name	HEX
ID	01
Function	02
Starting Address Hi	00
Starting Address Lo	C4
Quantity Hi	00
Quantity Lo	14
CRC Hi	39
CRC Lo	F8

Response	
Domain Name	HEX
ID	01
Function	02
Byte Count	03
Output Status 204-197 [#200]	03
Output Status 212-205 [#201]	05
Output Status 216-213 [#202]	09
CRC Hi	4B
CRC Lo	18

4.1.3 (03H) Read Holding Register

Example of Reading Registers 108-110:

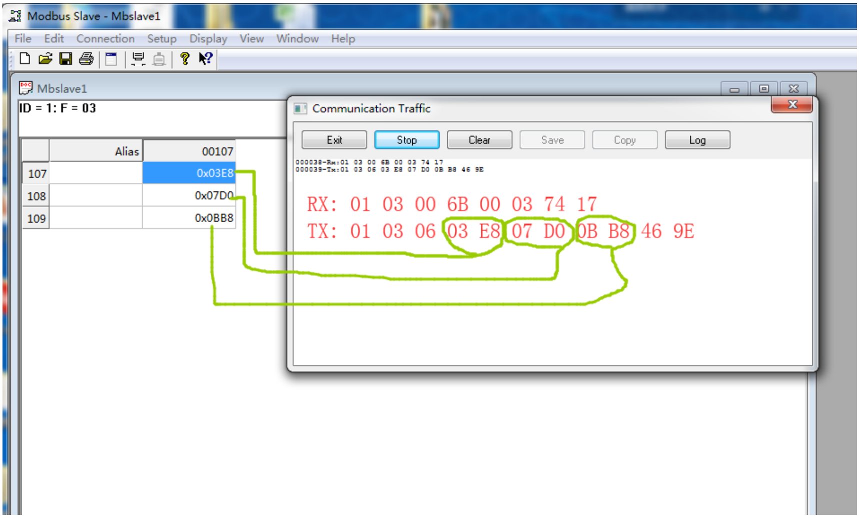
MGETDATA[200,1,107,6,3,300]

The data frame is as follows:

RX： 01 03 00 6B 00 03 74 17

TX： 01 03 06 03 E8 07 D0 0B B8 46 9E

M350 Communication with Computer Virtual Slave:



ID = 1: F = 03

	Alias	00107
107		0x03E8
108		0x07D0
109		0x0BB8

```
000038-Rx:01 03 00 6B 00 03 74 17
000039-Tx:01 03 06 03 E8 07 D0 0B B8 46 9E
```

```
RX: 01 03 00 6B 00 03 74 17
TX: 01 03 06 03 E8 07 D0 0B B8 46 9E
```

Request	
Domain Name	HEX
ID	01
Function	03
Starting Address Hi	00
Starting Address Lo	6B
Register Quantity Hi	00
Register Quantity Lo	03
CRC Hi	74
CRC Lo	17

Response	
Domain Name	HEX
ID	01
Function	03
Byte Count	06
Register Value Hi (108) [#200]	03
Register Value Lo (108) [#201]	E8
Register Value Hi (109) [#203]	07
Register Value Lo (109) [#202]	D0
Register Value Hi (110) [#205]	0B
Register Value Lo (110) [#204]	B8
CRC Hi	46
CRC Lo	9E

4.1.4 (04H) Read Input Register

Example of Request to Read Input Register 9:

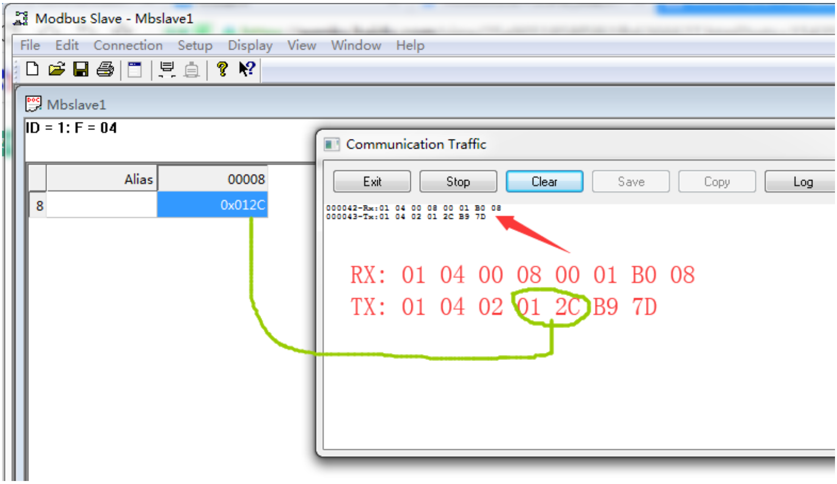
MGETDATA[200,1,8,2,4,300]

The data frame is as follows:

RX： 01 04 00 08 00 01 B0 08

TX： 01 04 02 01 2C B9 7D

M350 Communication with Computer Virtual Slave:



ID = 1: F = 04

	Alias	00008
8		0x012C

```
000042-Rx:01 04 00 08 00 01 B0 08
000043-Tx:01 04 02 01 2C B9 7D
```

RX: 01 04 00 08 00 01 B0 08
TX: 01 04 02 01 2C B9 7D

Request	
Domain Name	HEX
ID	01
Function	04
Starting Address Hi	00
Starting Address Lo	08
Register Quantity Hi	00
Register Quantity Lo	01
CRC Hi	B0
CRC Lo	08

Response	
Domain	HEX
ID	01
Function	04
Byte Count	02
Input Register Value Hi(9) [#201]	01
Input Register Value Lo(9) [#200]	2C
CRC Hi	B9
CRC Lo	7D

4.2 Write Functions:MSETDATA[]

MSETDATA[X1,X2,X3,X4,X5,X6]

X1: 50-499, The continuous address space from #50 to #499 is the data content to be written (Note: At this time, one macro address corresponds to one byte).

X2: Slave (SLAVE) station number.

X3: Starting address of the data to be written.

X4: Byte length of the data to be written (Note: In Modbus, one register occupies 2 bytes).

X5: Data writing method.

Write Operation Function Code Assignment Table (RTU Mode):

Function Code	**X5**
0FH	15
10H	16

X6: 50-499, Exception code return values will be stored in the macro address of X6 (#50-#499).

Exception Response Code List:

X6	**Description**
0x00	Normal
0x01	Invalid or unsupported function code
0x02	Invalid or unsupported address
0x03	Invalid or unsupported data
0x04	Action execution failed
0x05	Action in progress (may require a long time)
0x06	Device is busy, unable to execute action temporarily
0x08	File data verification error
0x0A	Invalid gateway path
0x0B	Target device is unresponsive
0xE0	Transmission error or illegal Modbus data frame
0xFF	Timeout
0xe1	Undefined action

4.2.1 (0FH) Write Multiple Coils

Example of Writing 12 Coils Starting from Coil 7:

#200 = 3

#201= 4

MSETDATA[200,1,6,12,15,300]

The data frame is as follows:

RX：01 0F 00 06 00 0C 02 03 04 E4 E5

Domain	**HEX**
ID	01
Function	0F
Starting Address Hi	00
Starting Address Lo	06
Quantity of Outputs Hi	00
Quantity of Outputs Lo	0C
Byte Count	02
Output Value Hi	03
Output Value Lo	04
CRC Hi	E4
CRC Lo	E5

4.2.2 (10H) Write Multiple Registers

Example of Writing Hexadecimal 08 07 and 0A 09 into Two Registers Starting from Address 4:

#200 = 7

#201= 8

#200 = 9

#201= 10

MSETDATA[200,1,5,4,16,300]

The data frame is as follows:

RX：01 10 00 05 00 02 04 08 07 0A 09 46 97

Domain	HEX
ID	01
Function	10
Starting Address Hi	00
Starting Address Lo	05
Number of Registers Hi	00
Number of Registers Lo	02
Byte Count	04
Register Value Hi	08
Register Value Lo	07
Register Value Hi	0A
Register Value Hi	09
CRC Hi	46
CRC Lo	97

4.3 Convert Bytes to Data Types : MBYTE2DATA []

MBYTE2DATA [X1,X2,X3]

X1: Macro address where the converted data will be saved, 50-499: #50-#499;

X2: Starting macro address of the bytes to be converted, 50-499: #50-#499;

X3: Target data type

Data Type Mapping Table:

X3	Data Type
0	32-bit Float
1	16-bit Signed
2	16-bit Unsigned
3	32-bit Signed
4	32-bit Unsigned

4.3.1 Example

Example of Converting Bytes to Float

```
//0xF5C3 ,0x4148
#200 = 195          //0xC3
#201 = 245          //0xF5
#202 = 72           //0x41
#203 = 65           //0x48
MBYTE2DATA[50,200,0]
M30
```

Result:

#50=12.56;

Note:

MGETDATA[] and **MBYTE2DATA[]** can be used together.

4.4 Convert Data Types to Bytes : MDATA2BYTE []

MDATA2BYTE [X1,X2,X3]

X1: Starting macro address where the converted bytes will be saved, 50-499: #50-#499;

X2: Macro address of the data before conversion, 50-499: #50-#499;

X3: Target data type

Data Type Mapping Table:

X3	Data Type
0	32-bit Float
1	16-bit Signed
2	16-bit Unsigned
3	32-bit Signed
4	32-bit Unsigned

4.4.1 Example

Example of Converting Float Data to Bytes:

#50 = 12.56 // Float data before conversion

MDATA2BYTE[200,50,0]

M30

Result:

#200 = 0xC3

#201 = 0xF5

#202 = 0x41

#203 = 0x48

Note:

MSETDATA[] and MDATA2BYTE[] can be used together.