

M350-Modbus 手册

V1.1

目录

- 第一章 简介 2
- 第二章 系统参数设置 2
- 第三章 接线示意图 2
- 第四章 Modbus 宏函数 3
 - 4.1 读函数 MGETDATA[] 3
 - 4.1.1 (01H)读线圈 4
 - 4.1.2 (02H)读离散量输入 6
 - 4.1.3 (03H)读保持寄存器 8
 - 4.1.4 (04H)读输入寄存器 10
 - 4.2 写函数 MSETDATA[] 12
 - 4.2.1 (0FH)写多个线圈 13
 - 4.2.2 (10H)写多个寄存器 14
 - 4.3 字节转换为数据类型 MBYTE2DATA [] 15
 - 4.3.1 示例 15
 - 4.4 数据类型转换为字节 MDATA2BYTE [] 16
 - 4.4.1 示例 16

第一章 简介

M350 支持 modbus RTU 通信协议；

- 串口通信接口 RS-232 或者 RS-485；
- 波特率: B2400、B4800、B9600、B19200、B115200；
- 主从模式：默认主机模式
- 支持的功能码: 01H、02H、03H、04H、0FH、10H；

第二章 系统参数设置

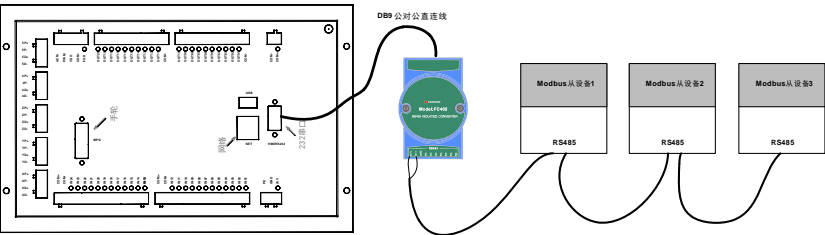
279 号参数：Modbus RTU 使能

267 号参数：串口 2 波特率 B2400、B4800、B9600、B19200、B115200；

默认数据格式： 8 位数据位，1 位停止位，无校验位；

参数设置完后 请重启系统！

第三章 接线示意图



系统与多个从设备接线示意图

第四章 Modbus 宏函数

4.1 读函数 MGETDATA[]

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

X1: 50-499,读取的每个字节存储到以#50 到#499 连续地址空间(注:此时一个宏地址对应一个字节)。

X2: 从站(SLAVE)端站号。

X3: 读取数据的起始地址。

X4: 读取数据字节长度(注:modbus 一个寄存器占用 2 个字节)。

X5: 读数据方式。

读操作功能码分配表(RTU 模式)

功能码	X5
01H	1
02H	2
03H	3
04H	4

X6: 50-499, 异常代码返回值会保存在 X6 的宏地址(#50-#499)中。

异常响应代码列表:

X6	描述
0x00	正常
0x01	无效的或不支持的功能码
0x02	无效的或不支持的地址
0x03	无效的或不支持的数据
0x04	动作执行失败
0x05	动作执行中（可能需较长时间）
0x06	设备正忙，暂时不能执行动作
0x08	文件数据校验出错
0x0A	无效的网关路径
0x0B	目标设备无响应
0xE0	传输出错或非法的 modbus 数据帧
0xFF	超时

Oxe1	未定义的动作
------	--------

4.1.1 (01H)读线圈

请求读离散量输出 20-38 的实例：

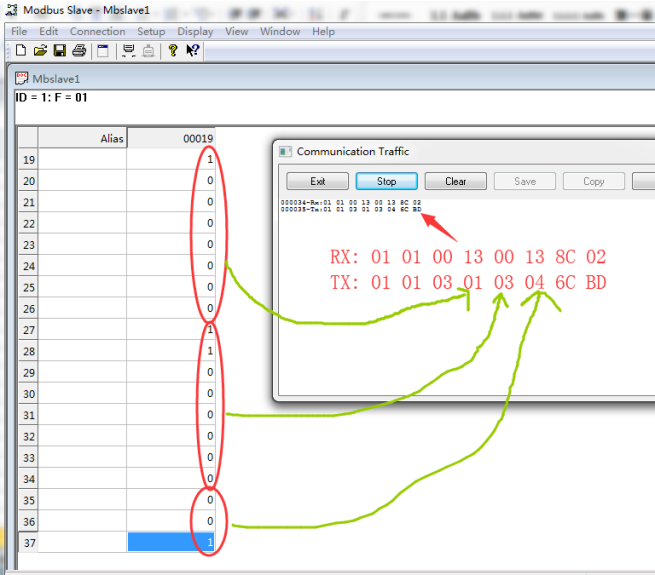
MGETDATA[200,1,19,1,300]

数据帧如下：

RX: 01 01 00 13 00 13 8C 02

TX: 01 01 03 01 03 04 6C BD

M350 与电脑虚拟从机通信：



请求	
域名	16 进制
ID	01
功能	01
起始地址 Hi	00
起始地址 Lo	13
输出数量 Hi	00
输出数量 Li	13
CRC Hi	8c
CRC Lo	02

响应	
域名	16 进制
ID	01
功能	01
字节数	03
输出状态 27-20 [#200]	01
输出状态 35-28 [#201]	03
输出状态 38-36 [#202]	04
CRC Hi	6C
CRC Lo	BD

4.1.2 (02H)读离散量输入

请求读离散量输入 197-216 的实例：

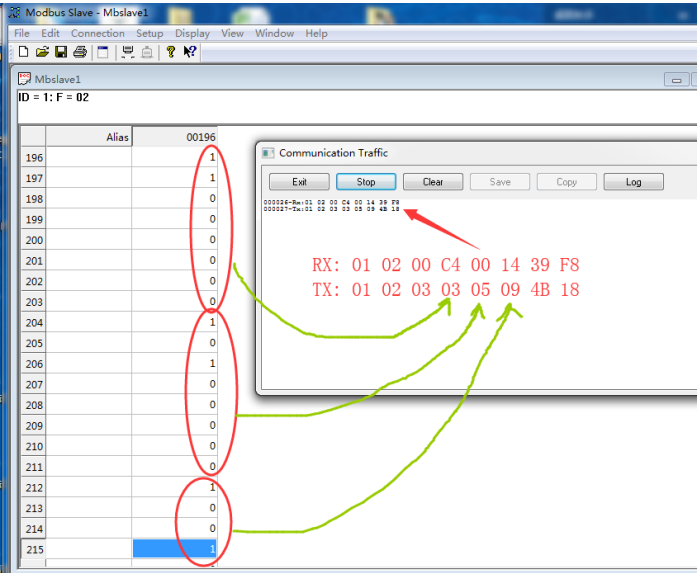
```
MGETDATA[200,1,196,20,2,300]
```

数据帧如下：

```
RX: 01 02 00 C4 00 14 39 F8
```

```
TX: 01 02 03 03 05 09 4B 18
```

M350 与电脑虚拟从机通信：



请求		
域名	16 进制	
ID	01	
功能	02	
起始地址 Hi	00	
起始地址 Lo	C4	
输出数量 Hi	00	
输出数量 Li	14	
CRC Hi	39	
CRC Lo	F8	

响应	
域名	16 进制
ID	01
功能	02
字节数	03
输出状态 204-197 [#200]	03
输出状态 212-205 [#201]	05
输出状态 216-213 [#202]	09
CRC Hi	4B
CRC Lo	18

4.1.3 (03H)读保持寄存器

读取寄存器 108-110 的实例：

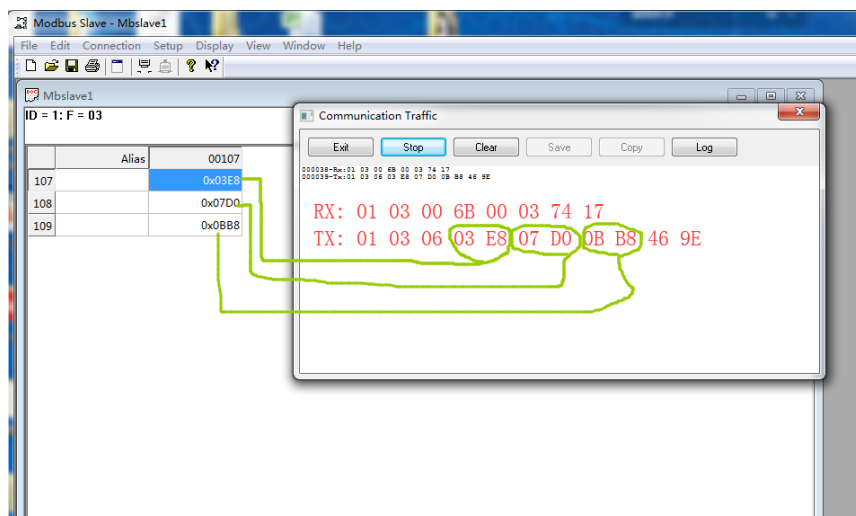
MGETDATA[200,1,107,6,3,300]

数据帧如下：

RX: 01 03 00 6B 00 03 74 17

TX: 01 03 06 03 E8 07 D0 0B B8 46 9E

M350 与电脑虚拟从机通信：



请求	
域名	16 进制
ID	01
功能	03
起始地址 Hi	00
起始地址 Lo	6B
寄存器数量 Hi	00
寄存器数量 Li	03
CRC Hi	74
CRC Lo	17

响应	
域名	16 进制
ID	01
功能	03
字节数	06
寄存器值 Hi(108) [#200]	03
寄存器值 Lo(108) [#201]	E8
寄存器值 Hi(109) [#203]	07
寄存器值 Lo(109) [#202]	D0
寄存器值 Hi(109) [#205]	0B
寄存器值 Lo(109) [#204]	B8
CRC Hi	46
CRC Lo	9E

4.1.4 (04H)读输入寄存器

请求读输入寄存器 9 的实例：

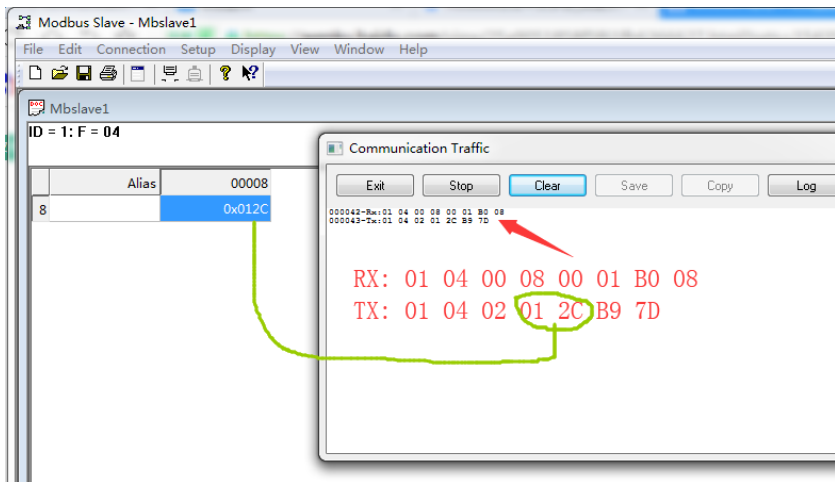
MGETDATA[200,1,8,2,4,300]

数据帧如下：

RX: 01 04 00 08 00 01 B0 08

TX: 01 04 02 01 2C B9 7D

M350 与电脑虚拟从机通信：



请求	
域名	16 进制
ID	01
功能	04
起始地址 Hi	00
起始地址 Lo	08
寄存器数量 Hi	00
寄存器数量 Li	01
CRC Hi	B0
CRC Lo	08

响应	
域名	16 进制
ID	01
功能	04
字节数	02
输入寄存器值 Hi(9) [#201]	01
输入寄存器值 Lo(9) [#200]	2C
CRC Hi	B9
CRC Lo	7D

4.2 写函数 MSETDATA[]

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

X1: 50-499, #50 到#499 连续地址空间为要写入的数据内容(注:此时一个宏地址对应一个字节)。

X2: 从站(SLAVE)端站号。

X3: 写数据的起始地址。

X4: 写数据的字节长度(注:modbus 一个寄存器占用 2 个字节)。

X5: 写数据方式

写操作功能码分配表(RTU 模式)

功能码	X5
0FH	15
10H	16

X6: 50-499, 异常代码返回值会保存在 X6 的宏地址(#50-#499)中。

异常响应代码列表:

X6	描述
0x00	正常
0x01	无效的或不支持的功能码
0x02	无效的或不支持的地址
0x03	无效的或不支持的数据
0x04	动作执行失败
0x05	动作执行中（可能需较长时间）
0x06	设备正忙，暂时不能执行动作
0x08	文件数据校验出错
0x0A	无效的网关路径
0x0B	目标设备无响应
0xE0	传输出错或非法的 modbus 数据帧
0xFF	超时
0xe1	未定义的动作

4.2.1 (0FH)写多个线圈

从线圈 7 开始写入 12 个线圈的实例:

#200 = 3

#201= 4

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

数据帧如下:

RX: 01 0F 00 06 00 0C 02 03 04 E4 E5

域名	16 进制
ID	01
功能	0F
起始地址 Hi	00
起始地址 Lo	06
输出数量 Hi	00
输出数量 Li	0C
字节数	02
输出值 Hi	03
输出值 Lo	04
CRC Hi	E4
CRC Lo	E5

4.2.2 (10H)写多个寄存器

将十六进制 08 07 和 0A 09 写入以 4 开始的两个寄存器的实例：

#200 = 7

#201= 8

#200 = 9

#201= 10

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

数据帧如下：

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

域名	16 进制
ID	01
功能	10
起始地址 Hi	00
起始地址 Lo	05
寄存器数量 Hi	00
寄存器数量 Li	02
字节数	04
寄存器值 Hi	08
寄存器值 Lo	07
寄存器值 Hi	0A
寄存器值 Hi	09
CRC Hi	46
CRC Lo	97

4.3 字节转换为数据类型 MBYTE2DATA []

MBYTE2DATA [X1,X2,X3]

X1: 转换后的数据保存宏地址, 50-499: #50-#499;

X2: 需要转换的字节起始宏地址,50-499: #50-#499;

X3: 目标数据类型

数据类型对照表:

X3	数据类型
0	32 位浮点型
1	16 位有符号
2	16 位无符号
3	32 位有符号
4	32 位无符号

4.3.1 示例

字节转换为浮点型为例

//0xF5C3 ,0x4148

#200 = 195 //0xC3

#201 = 245 //0xF5

#202 = 72 //0x41

#203 = 65 //0x48

MBYTE2DATA[50,200,0]

M30

运行结果:

#50=12.56;

提示:

MGETDATA[] 与 MBYTE2DATA[] 可以一起使用。

4.4 数据类型转换为字节 MDATA2BYTE []

MDATA2BYTE [X1,X2,X3]

X1: 转换后字节保存的起始宏地址, 50-499: #50-#499;

X2: 转换前的数据宏地址,50-499: #50-#499;

X3: 目标数据类型

数据类型对照表:

X3	数据类型
0	32 位浮点型
1	16 位有符号
2	16 位无符号
3	32 位有符号
4	32 位无符号

4.4.1 示例

浮点型数据转换为字节实例如下:

#50 = 12.56 //转换前的浮点型数据

MDATA2BYTE[200,50,0]

M30

运行结果:

#200 = 0xC3

#201 = 0xF5

#202 = 0x41

#203 = 0x48

提示:

MSETDATA [] 与 MDATA2BYTE [] 可以一起使用。