DS9L通讯协议

公司官网：www.toky.com.cn

▣通信协议

一、MODBUS串行通信协议基本规则

1、仪表使用Modbus RTU通信协议，进行RS485半双工通信，读功能号0x03，写功能号0x10，采用16位CRC校验，仪表对校验错误不返回。
数据帧格式：

起始位	数据位	停止位	校验位
1	8	1	无

（1）所有RS485回路通信应遵照主/从方式。在这种方式下，信息和数据在单个主站和最多32个从站（监控设备）之间传递；
（2）主站将初始化和控制所有在RS485通信回路上传递的信息；
（3）无论如何都不能从一个从站开始通信；
（4）所有RS485环路上的通信都以“打包”方式发生。一个数据包就是一个简单的字符串（每个字符串8位），一个包中最多可含128个字节。组成这个包的字节构成标准异步串行数据，并按8位数据位，1位停止位，无校验位的方式传递。
（5）主站发送称为请求，从站发送称为响应；
（6）任何情况从站只能响应主站一个请求。

2、每个MODBUS数据包都由以下几个部分组成：
（1）从站地址；（2） 要执行的功能码；（3）寄存器地址（变量地址）；（4）数据 ；（5）CRC校验；
（1）从站地址：地址长度为1个字节，有效的从站地址范围为1-247，从站如果接收到一帧地址信息与自身地址相符合的数据包时，就执行数据包中包含的命令。
（2） MODBUS数据包中功能码长度为一个字节用以通知从站应当执行何种操作从站响应数据包中应当包含主站所请求操作的相同功能码字节。有关功能码参照下表：

功能码	含义	功能
0x03	读取寄存器	读取一个或多个当前寄存器值
0x06	写单寄存器	将指定数值写入内部一个寄存器内
0x10	写多寄存器	将指定数值写入内部多个寄存器内（厂家默认为写单寄存器）

（3）寄存器地址变量：从机执行有效命令时数据区域存储的位置。不同变量占用不同寄存器个数，有些地址变量占用两个寄存器，4字节数据，有些变量占用一个寄存器，2字节数据，请根据实际情况使用。
（4）数据区：数据区包含了终端执行特定功能所需要的数据或者终端响应查询时采集到的数据。这些数据的内容可能是数值、参考地址或者设置值；例如：功能码告诉终端读取一个寄存器，数据区则需要指明从哪个寄存器开始及读取多少个数据，内嵌的地址和数据依照类型和从机之间的不同内容而有所不同；寄存器数值发送顺序为：高位字节在前，低位字节在后。
（5）CRC校验： MODBUS-RTU模式采用16位CRC校验。发送设备应当对包裹中的每一个数据都进行CRC16计算，最后结果存放入检验域中。接收设备也应当对包裹中的每一个数据（除校验域以外）进行CRC16计算，将结果域校验域进行比较；只有相同的包裹才可以被接受，具体的CRC校验算法参照附录。

二、 网络时间考虑

在RS485网络上传送包裹需要遵循以下有关时间的规定：
（1） 波特率设置为9600时，主站两次请求之间的延时推荐为300ms或以上，使用更小延时时可能会产生丢包；
（2）推荐波特率是9600时的，使用更小波特率时请适当放大延时时间，例如使用2400波特率时，两次请求应设为500ms以上。

三、通信异常处理：

如果主站发送了一个非法的数据包或者是主站请求一个无效的数据寄存器时，异常的数据响应就会产生。这个异常数据响应由从站地址、功能码、故障码和校验域组成。当功能码域的高比特位置为1 时，说明此时的数据帧为异常响应。
下表说明异常功能码的含义：
根据MODBUS通讯要求，异常响应功能码=请求功能码+0x80；异常应答时，将功能号的最高位置1。例如：主机请求功能号为0x04，则从机返回的功能号对应为0x84。

错误码类型	名称	内容说明
0x01	功能码错误	仪表接收到不支持的功能号
0x02	变量地址错误	主机指定的数据位置超出仪表的范围或接收到非法的寄存器操作
0x03	变量数据值错误	主机发送的数据值超出仪表对应的数据范围或数据结构不完整。

四、通讯帧格式说明

1、读多寄存器
例：主机读取UA（A相电压），设现测量到A相电压为220.0V。
UA的地址编码是0x4000,因为UA是定点数(4字节)，占用2个数据寄存器，220.0V对应的十六进制数据是：0x0000898（2200）。
主机请求

从站地址	读功能号	寄存器地址（变量）		寄存器数量		CRC校验码	
1	2	3	4	5	6	7	8
表地址	功能号	起始地址 高位	起始地址 地位	高位	低位	CRC码 的低位	CRC码 的高位
0x01	0x03	0x40	0x00	0x00	0x02	0xD1	0xCB

从机正常应答(高字在前)

从站地址	读功能号	字节数（2倍寄存器数目）	寄存器数据		寄存器数据		CRC校验码	
1	2	3	4	5	6	7	8	9
表地址	功能号	数据字节长度	数据1高位	数据1低位	数据2高位	数据2低位	CRC码的低位	CRC码的高位
0x01	0x03	0x04	0x00	0x00	0x08	0x98	0xFC	0x59

从机正常应答(低字在前)

从站地址	读功能号	字节数（2倍寄存器数目）	寄存器数据		寄存器数据		CRC校验码	
1	2	3	4	5	6	7	8	9
表地址	功能号	数据字节长度	数据2高位	数据2低位	数据1高位	数据1低位	CRC码的低位	CRC码的高位
0x01	0x03	0x04	0x08	0x98	0x00	0x00	0x79	0xBC

功能号异常应答：(例如主机请求功能号为0x04)。

从机异常应答(读多寄存器)				
1	2	3	8	9
表地址	功能号	错误码	CRC码的低位	CRC码的高位
0x01	0x84	0x01	0x82	0xC0

例：当前测量电流值为：Ia=100 A,Ib=200 A,Ic=300 A,分别一次读取三个电流的值。主机发送读01地址仪表，读从400C（A相电流）寄存器开始的电流值数据。100.000对应的十六进制数为000186A0;200.000对应的十六进制数为00030D40;300.000对应的十六进制数为000493E0;数据采用32 位无符号数据表示，带有3位小数点。例如，数据值为12345，则实际数值为12.345。

主机发送

表地址	功能号	地址		寄存器数量		CRC校验码	
01	03	40	0C	00	06	10	0B

仪表返回

表地址	功能号	读字节数量	数据1				数据2				数据3				CRC校验码	
01	03	0C	00	01	86	A0	00	03	0D	40	00	04	93	E0	8F	1D

2、写单路寄存器

例：主机写定点数第1路报警方式AD1。

假设AD1的地址编码是0x4900，因为AD1是定点数，占用1个数据寄存器，十进制11对应为0X000B。

主机请求(写单寄存器)

从站地址	写功能号	寄存器地址（变量）		寄存器数量		字节数（2倍寄存器数目）	寄存器数据		CRC校验码	
1	2	3	4	5	6	7	8	9	10	11
表地址	功能号	起始地址高位	起始地址低位	高位	低位	数据字节长度	数据1高位	数据1低位	数据2高位	数据2低位
0x01	0x06	0x49	0x00	0x00	0x01	0x02	0x00	0x0B	0xBE	0x75

从机正常应答(写单寄存器)

从站地址	写功能号	寄存器地址（变量）		寄存器数量		CRC校验码	
1	2	3	4	5	6	7	8
表地址	功能号	起始地址高8位	起始地址低8位	高位	低位	CRC码的低位	CRC码的高位
0x01	0x06	0x49	0x00	0x00	0x01	0x5E	0x56

3、写多路寄存器

例：主机写定点数第1路报警方式AD1。

假设AD1的地址编码是0x4800，因为AD1是定点数，占用1个数据寄存器，十进制11对应为0X000B。

主机请求(写多寄存器)										
1	2	3	4	5	6	7	8	9	10	11
表地址	功能号	起始地址高位	起始地址低位	数据字长高位	数据字长低位	数据字节长度	数据1高位	数据1低位	CRC码的低位	CRC码的高位
0x01	0x10	0x49	0x00	0x00	0x01	0x02	0x00	0x0B	0x3F	0x53

从机正常应答(写多寄存器)							
1	2	3	4	5	6	7	8
表地址	功能号	起始地址高8位	起始地址低8位	数据字长高位	数据字长低位	CRC码的低位	CRC码的高位
0x01	0x10	0x49	0x00	0x00	0x01	0x17	0x95

数据位置错误应答：(例如主机请求写地址索引为0x0050)。

从机异常应答(写多寄存器)				
1	2	3	4	5
表地址	功能号	错误码	CRC码的低位	CRC码的高位
0x01	0x90	0x02	0xCD	0xC1

相关参数地址映像表　　注：地址号相当变量数组的索引

序号	地址映射	变量名称	寄存器数量	数据类型	读写属性	数据转换说明
1	0x4000	相电压A	2	long	R	0.1V 注⑥
2	0x4002	相电压B	2	long	R	
3	0x4004	相电压C	2	long	R	
4	0x4006	线电压AB	2	long	R	
5	0x4008	线电压BC	2	long	R	
6	0x400a	线电压CA	2	long	R	
7	0x400c	相电流A	2	long	R	0.001A 注⑥
8	0x400e	相电流B	2	long	R	
9	0x4010	相电流C	2	long	R	
10	0x4012	有功功率A	2	long	R	0.1W 注⑥
11	0x4014	有功功率B	2	long	R	
12	0x4016	有功功率C	2	long	R	
13	0x4018	总有功功率	2	long	R	
14	0x401a	无功功率A	2	long	R	0.1VAR
15	0x401c	无功功率B	2	long	R	
16	0x401e	无功功率C	2	long	R	
17	0x4020	总无功功率	2	long	R	0.1VA 注⑥
18	0x4022	视功功率A	2	long	R	
19	0x4024	视在功率B	2	long	R	
20	0x4026	视在功率C	2	long	R	
21	0x4028	总视在功率	2	long	R	
22	0x402a	功率因数A	2	long	R	0.001 注⑥
23	0x402c	功率因数B	2	long	R	
24	0x402e	功率因数C	2	long	R	
25	0x4030	总功率因数	2	long	R	
26	0x4032	频率	2	long	R	0.01Hz 注⑥
27	0x4034	总有功电度	2	long	R	0.001kWh
28	0x4036	总无功电度	2	long	R	
29	0x4038	正有功电度	2	long	R	
30	0x403a	负有功电度	2	long	R	0.001kvarh 注⑥
31	0x403c	正无功电度	2	long	R	
32	0x403e	负无功电度	2	long	R	
33	0x4800	电压变比PT1	2	long	R/W	0.001 注⑥
34	0x4802	电压变比PT2	2	long	R/W	
35	0x4804	电流变比CT1	2	long	R/W	
36	0x4806	电流变比CT2	2	long	R/W	

37	0x4808	第1路报警值	2	long	R/W	0.001 注⑥
38	0x480a	第1路回差值	2	long	R/W	
39	0x480c	第2路报警值	2	long	R/W	
40	0x480e	第2路回差值	2	long	R/W	
保留扩展						
41	0x4900	第1路报警方式值（见附表1）	1	int	R/W	无小数点
42	0x4901	第1路报警单位 注③	1	int	R/W	
43	0x4902	第1路报警延时	1	int	R/W	
44	0x4903	第1路切除延时	1	int	R/W	
45	0x4904	第2路报警方式值（见附表1）	1	int	R/W	
46	0x4905	第2路报警单位 注③	1	int	R/W	
47	0x4906	第2路动作延时	1	int	R/W	
48	0x4907	第2路切除延时	1	int	R/W	
保留扩展						
49	0x4a00	接线方式 注①	1	int	R	无小数点
50	0x4a01	通信地址	1	int	R	
51	0x4a02	波特率 注②	1	int	R	
52	0x4a03	数据格式	1	int	R	
53	0x4a07	开关量输出(报警状态有效） 注④	1	int	R	
54	0x4a08	开关量输入 注⑤	1	int	R	
55	0x4a09	遥控输入	1	int	R/W	
56	0x4a0a	背光时间	1	int	R/W	

注①:接线方式

通信数值	0	1
菜单显示	3-4	3-3

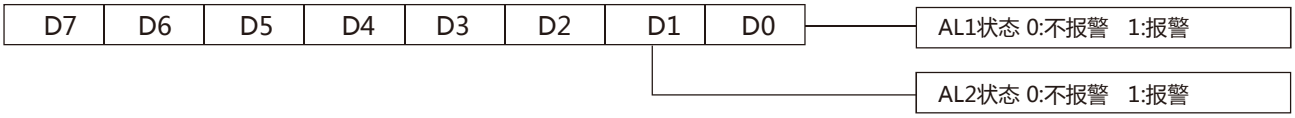
注②:波特率

通信数值	0	1	2	3
菜单显示	1K2	2K4	4K8	9K6

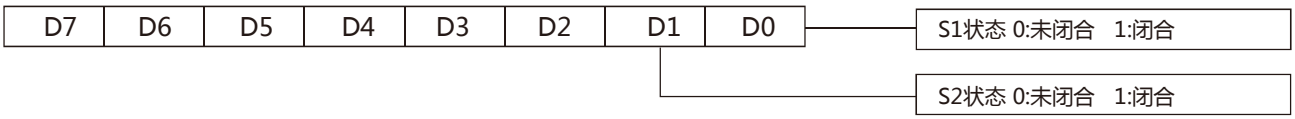
注③:报警/变送值单位

通信数值	0	1	2
菜单显示	1	K	M

注④:报警状态指示



注⑤:开关量输入状态指示



注⑥:读出或写入的实际值

读出或写入的实际值=通信读出值x单位
例如：要读出A相相电压，如果读出的值为0x00000898，
对应的十进制数位：2200，因为电压的单位为0.1V，则实际的A相电压值为2200x0.1V=220.0V

生成一个CRC的流程为：（可参考后面的程序例子）

1、预置一个16位寄存器为0FFFFH(全1)，称之为CRC寄存器。

2、把数据帧中的第一个字节的8位与CRC寄存器中的低字节进行异或运算，结果存回CRC寄存器。

3、将CRC寄存器向右移一位，最高位填以0，最低位移出并检测。

4、如果最低位为0，重复第三步(下一次移位);如果最低位为1，将CRC寄存器与一个预设的固定值(0A001H)进行异或运算。

5、重复第三步和第四步直到8次移位。这样处理完了一个完整的八位。

6、重复第2步到第5步来处理下一个八位，直到所有的字节处理结束。

7、最终CRC寄存器的值就是CRC的值。此外还有一种利用预设的表格计算CRC的方法，它的主要特点是计算速度快，但是表格需要较大的存储空间，该方法此处不再赘述，请参阅相关资料。

16位CRC校验码获取程序

```
unsigned int Get_CRC (uchar*pBuf,uchar num)
{
      unsigned i,j;
    unsigned int wCrc=0xFFFF;
    for(i=0;i < num;i++)
      {
    wCrc^=(unsigned int)(pBuf[i]);
    for(j=0;j < 8;j++)
        {
    if(wCrc &1){wCrc > > =1;   wCrc^=0xA001;}
    else wCrc > > =1;
        }
      }
      return wCrc;
    }
```