

单相电量表

Modbus_RTU 通讯规约
(Ver C10.01)

2012-01-05

目录

1、字节格式:	3
2、 通讯数据格式	3
3、帧格式	3
4、通讯波特率	5
5、仪表地址	5
6、通讯功能码:	5
7、 通讯数据 CRC 校验	5
8、仪表数据寄存器地址	5
附录 1	9

本规约采用Modbus 规约RTU模式，可以方便地与多种组态软件相连接，其通讯驱动与Modicon Modbus_RTU格式完全兼容。

1、字节格式：



每字节含8位二进制码，传输时加上一个起始位(0)，一个停止位(1)，共10位。其传输序列如下图所示，D0是字节的最低有效位，D7是字的最高有效位。先传低位，后传高位。

2、通讯数据格式

通讯时数据以字(WORD—2字节)的形式回送，回送的每个字中，高字节在前，低字节在后，如果2个字连续回送(如：浮点或长整形)，则高字在前，低字在后。

数据类型	寄存器数	字节数	说明
字节数据	1	1	
整形数据	1	2	一次送回，高字节在前，低字节在后
长整形数	2	4	分两个字回送，高字在前，低字在后
浮点数据			

3、帧格式

3.1读取仪表保持寄存器内容（功能码 03H 或 04H）

3.1.1上位机发送的帧格式：

顺序	代码	示例	说明
1	仪表地址	1	仪表的通讯地址（1-255之间）
2	03H 或 04H	03H	功能码
3	起始寄存器地址高字节	01H	寄存器起始地址
4	起始寄存器地址低字节	00H	
5	寄存器个数高字节	00H	寄存器个数
6	寄存器个数低字节	02H	
7	CRC16 校验低字节	C5H	CRC 校验数据
8	CRC16 校验高字节	F7H	

3.1.2 仪表回送的帧格式（数据正常）

顺序	代 码	说 明
1	仪表地址	仪表的通讯地址（1-255之间）
2	03H 或 04H	功能码
3	回送数据域字节数(M)	
4	第一个寄存器数据	
.....	
	第N个寄存器数据	
M+4	CRC 校验低字节	
M+5	CRC 校验高字节	

3.1.3 如果起始寄存器地址或寄存器个数错误，仪表回送：

顺序	代 码	示 例	说 明
1	仪表地址	1	仪表的通讯地址（1-255之间）
2	83H 或 84H	83H	功能码——针对03H, 04H
3	02H	02H	错误代码
4	CRC 校验低字节	F1H	
5	CRC 校验高字节	C0H	

3.2 设置仪表寄存器内容（功能码 06H 或10H 或16H）

3.2.1.1 功能码06H写单路，将一个字（2 字节）数据写入仪表寄存器中，上位机发送的帧格式：

顺序	代 码	示 例	说 明
1	仪表地址	1	仪表的通讯地址（1-255 之间）
2	06H	06H	功能码
3	寄存器地址高字 节	09H	寄存器地址0905H
4	寄存器地址低字 节	05H	
5	写入数据高字节	00H	写入数据43H
6	写入数据低字节	43H	
7	CRC 校验低字节	A6H	CRC 校验数据A6DBH
8	CRC 校验高字节	DBH	

3.2.1.2 仪表回送：如果写入正确，则仪表回送相同的数据。

3.2.2.1 功能码 16H 或10H 写多路寄存器，上位机发送的帧格式：

顺序	代 码	示 例	说 明
1	仪表地址	1	仪表的通讯地址（1-255之间）
2	16H 或10H	10H	功能码
3	寄存器起始地址高字节	09H	寄存器地址0923H

4	寄存器起始地址低字节	03H	
5	寄存器个数高字节	00H	00H
6	寄存器个数低字节	02H	字节数据、整形数据：01H 浮点数据、长整形数：02H
7	字节数 (M)	4	字节数据、整形数据：02H 浮点数、长整形数：04H
8	数据高字节	00H	设置的数据为两个整形数10、50
	数据次高字节	0AH	
	数据次低字节	00H	
	数据低字节	32H	
M+8	CRC校验低字节	3DH	CRC校验数据
M+9	CRC校验高字节	78H	

3.2.2 仪表回送: (写入成功)

顺序	代码	示例	说明
1	仪表地址	1	仪表的通讯地址 (1-255之间)
2	16H 或10H	10H	功能码
3	起始地址高字节	09H	寄存器起始地址0923H
4	起始地址低字节	03H	
5	寄存器个数高字节	00H	寄存器个数2
6	寄存器个数低字节	02H	
7	CRC校验低字节	54H	CRC校验数据
8	CRC校验高字节	B2H	

3.2.3 仪表回送: (寄存器地址或数据错误)

顺序	代码	说明
1	仪表地址	仪表的通讯地址 (1-255之间)
2	96H 或90H 或86H	功能码——针对16H, 10H, 06H
3	03H	错误代码
4	CRC校验低字节	
5	CRC校验高字节	

注：以上介绍中CRC校验为16位，高字节在前，低字节在后。

4、通讯波特率：通讯波特率可以在9600、19200、38400之间选择。出厂时，仪表已设置某一波特率。

5、仪表地址：仪表地址可以在1-253之间选择。仪表出厂时，已设置某一地址。

6、通讯功能码：03H或04H(召测数据) 06H (10H 或16H) (数据设置)

7、通讯数据CRC 校验：

7.1 校验多项式： $X^{16}+X^{12}+X^5+1$

7.2 CRC 检验码的计算例程见附录。

7.3 CRC 检验从第1 字节开始至CRC 校验高字节前面的字节数据结束。

8、仪表数据寄存器地址

表1 寄存器地址表

寄存器地址	数据名称	量程系数	单位	数据格式	备注
-------	------	------	----	------	----

常规数据 功能码03H、04H读取；一次可最多读取123个连续字节；（特别注意：长整形数据和浮点型数据占两个寄存器，必须一次读出，若读取一半将返回错误信息，数据读取组侦时务必注意数据格式，比如，报警参数数据从A00H开始排列，A00H处是报警功能控制字（整形数据），A01H是电压报警上限参数（长整形数据），如果要连续读取这两个参数，寄存器个数应设置为3个，若只读取报警功能控制字，寄存器个数应设置为1个；若读电压报警上限参数，寄存器个数应设置为2个）					
电压					
0100H	电压	1000	伏特	长整形	
电流					
0102H	电流	100	安培	长整形	
功率					
0104H	有功功率	10	瓦	浮点形	
0106H	无功功率	10	瓦	浮点形	
0108H	视在功率	10	瓦	浮点形	
功率因数					
010AH	功率因数	1000		长整形	
频率					
010CH	频率	1000	赫兹	长整形	
能量					
0600H	总有功电能（代数和）	10	兆瓦时	长整形	
0602H	总无功电能（代数和）	10	兆乏时	长整形	
0604H	视在电能	10	兆伏安时	长整形	
版本					
0800H	仪表型号	---	---	ASSIC码	见表2
0805H	仪表版本号	---	---	ASSIC码	
080AH	通讯规约版本	---	---	ASSIC码	
系统配置数据：功能码03H读取；功能码10H写入，一次可最多写入120个连续字节即60个连续字；					
0900H	当前日期和时间	---	---	整形	见表3
0903H	电压倍率	1		整形	0-1000
0904H	电流倍率	1		整形	0-1000
0905H	通讯地址			整形	1-253
0906H	通讯速率			整形	0： 9600 1： 19200 2： 38400
报警通道1参数设置					
0A00H	电压报警上限	100	伏特	长整形	
0A02H	电压报警下限	100	伏特	长整形	
0A04H	电流报警上限	1000	安	长整形	
0A06H	电流报警下限	1000	安	长整形	
0A08H	有功功率报警上限	10	瓦特	长整形	
0A0AH	有功功率报警下限	10	瓦特	长整形	
0A0CH	无功功率报警上限	10	乏	长整形	
0A0EH	无功功率报警下限	10	乏	长整形	
0A10H	功率因素报警上限	1000		长整形	
0A12H	功率因素报警下限	1000		长整形	
0A14H	工频周波报警上限	1000	赫兹	长整形	
0A16H	工频周波报警下限	1000	赫兹	长整形	
报警通道2参数设置					
0A20H	电压报警上限	100	伏特	长整形	
0A22H	电压报警下限	100	伏特	长整形	
0A24H	电流报警上限	1000	安	长整形	
0A26H	电流报警下限	1000	安	长整形	
0A28H	有功功率报警上限	10	瓦特	长整形	
0A2AH	有功功率报警下限	10	瓦特	长整形	

0A2CH	无功功率报警上限	10	乏	长整形	
0A2EH	无功功率报警下限	10	乏	长整形	
0A30H	功率因素报警上限	1000		长整形	
0A32H	功率因素报警下限	1000		长整形	
0A34H	工频周波报警上限	1000	赫兹	长整形	
0A36H	工频周波报警下限	1000	赫兹	长整形	
0A38H	报警上限回差	100		长整形	
0A3AH	报警下限回差	100		长整形	
报警通道1控制参数					
0A50H	通道1报警功能选择			整形	见表4
报警通道2控制参数					
0A70H	通道2报警功能选择			整形	见表4
变送设置数据					
0B00H	变送输出选择	1		整形	0-7(见表5)
0B01H	变送下限电流对应值	1		整形	0、4、8
0B02H	变送上限			长整形	大于下限值
0B04H	变送下限			长整形	小于上限值
命令寄存器					
0C00H				写入不同命令,有不同作用,具体见命令列表	
命令列表					
2000H					清除电能

表2 版本格式

寄存器地址	高字节	低字节
寄存器800H-81D	0	ASCII

表3 日期和时间的格式

寄存器地址	高字节	低字节
寄存器0	年(BCD码)	月(BCD码)
寄存器1	日(BCD码)	时(BCD码)
寄存器2	分(BCD码)	秒(BCD码)

表4 报警功能控制字表

功能号	功能	备注
0	关闭报警	
1	电压上限报警	
2	电压下限报警	
3	电流上限报警	
4	电流下限报警	
5	有功功率上限报警	
6	有功功率下限报警	
7	无功功率上限报警	
8	无功功率下限报警	
9	功率因素上限报警	
10	功率因素下限报警	
11	频率上限报警	
12	频率下限报警	

表5 变送输出序号对应表

序号	输出参数	备注
0	无	变送输出关闭
1	电压	
2	电流	
3	有功功率	
4	无功功率	
5	视在功率	
6	功率因素	
7	频率	

表6 功能配置表

序号	仪表类型	备注
0	有功功率表	默认
1	无功功率表	
2	视在功率表	
3	功率因素表	

附录1 CRC 校验码的计算——计算法

```
unsigned short CRC16(puchMsg, usDataLen)
unsigned char *puchMsg ; /* 要进行CRC校验的消息 */
unsigned short usDataLen ; /* 消息中字节数 */
{
    unsigned char uchCRChi = 0xFF ; /* 高CRC字节初始化 */
    unsigned char uchCRCLo = 0xFF ; /* 低CRC 字节初始化 */
    unsigned uIndex ; /* CRC循环中的索引 */
    while (usDataLen--) /* 传输消息缓冲区 */
    {
        uIndex = uchCRChi ^ *puchMsgg++ ; /* 计算CRC */
        uchCRChi = uchCRCLo ^ auchCRChi[uIndex] ;
        uchCRCLo = auchCRCLo[uIndex] ;
    }
    return (uchCRChi << 8 | uchCRCLo) ;
}

/* CRC 高位字节值表 */
static unsigned char auchCRChi[] = {
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40
} ;

/* CRC低位字节值表*/
static char auchCRCLo[] = {
    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06,
    0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD,
    0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
    0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A,
    0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4,
    0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
    0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3,
    0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
    0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
    0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29,
    0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED,
    0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
```

0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60,
0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67,
0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E,
0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71,
0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B,
0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B,
0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,
0x43, 0x83, 0x41, 0x81, 0x80, 0x40

};