锂电协议GT版通信说明

版本：锂电协议GT版2024年7月 第1.0版

协议类型及波特率

- 协议类型：RS485
- 通讯波特率：9600

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：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
字符型	1	1	一次送回两个字符，不足两个用0补充
整型	1	2	一次送回，高字节在前，低字节在后
长整型	2	4	分两个字回送，高字在前，低字在后

帧格式

- ## 寄存器内容查询（功能码03H）

查询设备发送的帧格式

字节顺序	代码	示例	说明
0	设备地址	01H	BMS主机地址固定为1，逆变器地址范围(1~247)
1	03H	03H	功能码
2	起始寄存器地址高字节	00H	寄存器地址高8位
3	起始寄存器地址低字节	10H	寄存器地址低8位
4	寄存器个数高字节	00H	寄存器个数高8位
5	寄存器个数低字节	02H	寄存器个数低8位
6	CRC16校验高字节	C0H	CRC16校验高8位
7	CRC16校验低字节	CBH	CRC16校验低8位

BMS或逆变器解析成功回送帧格式

字节顺序	代码	说明
0	设备地址	BMS主机地址固定为1，逆变器地址范围(1~247)
1	03H	功能码
2	回送数据字节数(N)	N = 请求的寄存器个数*2
3	第1个寄存器数据高字节	
4	第1 个寄存器数据低字节	
……	……	
……	……	
	第N 个寄存器数据高字节	
	第N 个寄存器数据低字节	
N+3	CRC16校验高字节	
N+4	CRC16校验低字节	

BMS或逆变器解析错误回送帧格式

字节顺序	代码	说明
0	设备地址	BMS主机地址固定为1，逆变器地址范围(1~247)
1	03H	功能码
2	回送数据字节数(N)	N = 请求的寄存器个数*2
3	第1个0	总共回送N个0
4	第2个0	
.....	.....	
.....	.....	
	第N-1 个0	
	第N 个0	
N+3	CRC16校验高字节	
N+4	CRC16校验低字节	

寄存器读示例：

查询设备读取BMS电池电压与电流，BMS回复50.0v，10.0A

查询设备：01 03 00 16 00 02 25 CF

BMS：01 03 04 13 88 03 E8 7E 23

BMS寄存器地址

- R：表示可读即支持03 H 命令。W：表示可写即支持10 H 命令。
- Int：整型；Long：长整型；UInt：无符整型；ULong：无符长整型；ASC：ASCII码
- Max：取最大值；Min：取最小值
- BMS主机地址固定为1，负责汇总其他BMS数据，逆变器只与主机通讯
- 下表中的地址均以十进制表示

数据名称	单位	数据格式	起始地址	寄存器数	读写	备注
电池状态		UInt	19	1	R	电池状态位详见电池状态表1
预留			20	1	R	
池电组SOC平均值	1%	Int	21	1	R	
池电组电压平均值	10mV	Int	22	1	R	
电池组总电流	10mA	Int	23	1	R	正数为充电，负数为放电
预留			24	1	R	
池电组最大充电电流	10mA	Int	25	1	R	
池电组剩余容量	10mAh	Int	26	1	R	
池电组总容量	10mAh	Int	27	1	R	
预留			28	1	R	
预留			29	1	R	
预留			30	1	R	
预留			31	1	R	
预留			32	1	R	
池电组最大充电电压	10mV	Int	33	1	R	
预留			34	1	R	
池电组最大放电电流	10mA	Int	35	1	R	

电池状态表1

例：0000 0000 0000 0000（从右往左数，最右边一个0是bit0，最左边是bit15)

状态位	说明	备注
bit 0~4	保留	
bit 5	放电禁止	0：禁放 1：正常
bit 6	充电禁止	0：禁充 1：正常
bit 7~11	保留	
bit 12	强制充电	0：正常 1：强充
bit 13~15	保留	

逆变器寄存器地址

- R：表示可读即支持03 H 命令。W：表示可写即支持10 H 命令。
- Int：整型；Long：长整型；UInt：无符整型；ULong：无符长整型；ASC：ASCII码
- Max：取最大值；Min：取最小值
- 下表中的地址均以十进制表示

数据名称	单位	数据格式	起始地址	寄存器数	读写	备注
电池状态		Int	760	1	R	电池状态位详见电池状态表2
最大充电电压	0.1V	Int	761	1	R	
最低放电电压	0.1V	Int	762	1	R	
最大充电电流	0.1A	Int	763	1	R	
最大放电电流	0.1A	Int	764	1	R	
在线的BMS个数		int	765	1	R	与逆变器建立连接的BMS个数
预留			766	1	R	
预留			768	1	R	
预留			769	1	R	
电池电压	0.1V	Int	770	1	R	
总电池容量	0.1Ah	Int	771	1	R	
电池剩余容量	0.1Ah	Int	772	1	R	
电池百分比	1%	Int	773	1	R	
电池充电电流	0.1A	Int	774	1	R	
电池放电电流	0.1A	Int	775	1	R	

电池状态表2

状态位	说明	备注
bit 0~11	保留	
bit 12	放电禁止	0：正常 1：禁放
bit 13	充电禁止	0：正常 1：禁充
bit 14	强制充电	0：正常 1：强充
bit 15	与BMS建立连接	1：连接 0：异常

CRC校验算法

参数模型:CRC-16/MODBUS X16+X15+X2+1

C语言代码

```
const 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, 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,
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, 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
} ;

const 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
} ;

unsigned short sModbusCrc16(INT8U *chMsg, INT16U dataLen)
{
    unsigned char ubCRCHi = 0xFF;
    unsigned char ubCRCLo = 0xFF;
    unsigned char duwIndex;
    while (dataLen --)
    {
        duwIndex = 0xff&(ubCRCHi ^ *chMsg++);
        ubCRCHi = 0xff&(ubCRCLo ^ auchCRCHi[duwIndex]);
        ubCRCLo = auchCRCLo[duwIndex];
    }
    return (ubCRCHi << 8 | ubCRCLo);
}
```