网关RS485协议

|  |  |  |  |
| --- | --- | --- | --- |
| **修订时间** | **修订内容** | **修订人** | **备注** |
| **2022.10.14** | **初版** | **何东达** |  |
| **2023.1.9** | 修改设备类型2个字节，删除“修改设备地址”功能 | **何东达** |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[一、 命令格式 2](#_Toc21871)

[1.帧总长 2](#_Toc14276)

[2.CRC16 2](#_Toc3871)

[3.功能码 2](#_Toc14830)

[4.数据帧说明 2](#_Toc25356)

[5.电源监控板继电器说明 3](#_Toc30334)

[6.错误代码说明 4](#_Toc11638)

[二、 数据类型发码说明 4](#_Toc7062)

[1. 搜索在线从机 4](#_Toc9034)

[2. 读电源板采集数据和继电器状态 4](#_Toc13409)

[3. 写电源板继电器状态 5](#_Toc27961)

[三、 CRC-16 MODBUS原理说明 5](#_Toc4173)

1. **命令格式**

波特率：9600

数据位：8位

校验位：无

停止位：1位

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | 数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | 2字节 | 2字节 | |
| F5 | 5F |  |  |  |  |  |  | 0D | 0A |

## 

## **1.帧总长**

功能码+n个数据帧+数据CRC16

## **2.CRC16**

CRC-16/MODBUS校验（设备地址+帧总长+功能码+数据帧）

## **3.功能码**

|  |  |
| --- | --- |
| 功能码 | 说明 |
| 0x01 | 搜索在线从机 |
| 0x02 | 读取采集数据和继电器状态 |
| 0x03 | 写继电器状态 |
| 0x04 |  |
| 0x05 |  |
| 0x81 |  |
| 0x82 | 返回采集数据和继电器状态 |
| 0x83 | 返回继电器状态 |

**4.数据帧说明**

|  |  |  |  |
| --- | --- | --- | --- |
| 数据帧地址 | 说明 | 内容 | 备注 |
| 0x0000 | 单板继电器1~16状态 | 电源监控板1~10 | 读写 |
| 0x0001 | 单板继电器17~32状态 |  | 读写 |
| 0x0002 | 单板AC1输出电压 |  | 只读，放大100倍 |
| 0x0003 | 单板AC1输出电流 |  | 只读，放大100倍 |
| 0x0004 | 单板AC2输出电压 |  | 只读，放大100倍 |
| 0x0005 | 单板AC2输出电流 |  | 只读，放大100倍 |
| 0x0006 | 单板AC3输出电压 |  | 只读，放大100倍 |
| 0x0007 | 单板AC3输出电流 |  | 只读，放大100倍 |
| 0x0008 | 单板DC1输出电压 |  | 只读，放大100倍 |
| 0x0009 | 单板DC1输出电流 |  | 只读，放大100倍 |
| 0x000a | 单板DC2输出电压 |  | 只读，放大100倍 |
| 0x000b | 单板DC2输出电流 |  | 只读，放大100倍 |
| 0x000c | 单板DC3输出电压 |  | 只读，放大100倍 |
| 0x000d | 单板DC3输出电流 |  | 只读，放大100倍 |
| 0x000e | 单板DC4输出电压 |  | 只读，放大100倍 |
| 0x000f | 单板DC4输出电流 |  | 只读，放大100倍 |
| 0x0010 | 单板DC5输出电压 |  | 只读，放大100倍 |
| 0x0011 | 单板DC5输出电流 |  | 只读，放大100倍 |
| 0x0012 | 单板DC6输出电压 |  | 只读，放大100倍 |
| 0x0013 | 单板DC6输出电流 |  | 只读，放大100倍 |
| 0x0014 | 单板DC7输出电压 |  | 只读，放大100倍 |
| 0x0015 | 单板DC7输出电流 |  | 只读，放大100倍 |
| 0x0016 | 单板DC8输出电压 |  | 只读，放大100倍 |
| 0x0017 | 单板DC8输出电流 |  | 只读，放大100倍 |
| 0x0018 | 单板DC9输出电压 |  | 只读，放大100倍 |
| 0x0019 | 单板DC9输出电流 |  | 只读，放大100倍 |
| 0x001a | 单板DC10输出电压 |  | 只读，放大100倍 |
| 0x001b | 单板DC10输出电流 |  | 只读，放大100倍 |
| 0x001c | 单板温度1 |  | 只读，放大100倍 |
| 0x001d | 单板湿度1 |  | 只读，放大100倍 |
| 0x001e | 单板温度2 |  | 只读，放大100倍 |
| 0x001f | 单板湿度2 |  | 只读，放大100倍 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 0xff00 | 设备类型 | 0001:网关板 0002:电源监控板  0003数据采集板 00fe:所有类型 | 只读 |
| 0xff01 | 设备软件版本信息 |  | 只读 |
| 0xff02 | 设备硬件版本信息 |  | 只读 |
| 0xff03 | 设备错误代码 |  | 只读 |
| 0xff04 | 设备ID名字 | 最大32字符+“/0” |  |
| 0xff05 | 设备地址 | RS485地址 |  |
|  |  |  |  |

**5.电源监控板继电器说明**

|  |  |
| --- | --- |
| 电源监控板V1继电器说明 | |
| 0x0001 | AC1输出状态 |
| 0x0002 | DC1状态（照明） |
| 0x0004 | DC2状态 |
| 0x0008 | DC3状态 |
| 0x0010 | DC4状态 |
| 0x0020 | DC5状态 |
| 0x0040 | DC6状态 |
| 0x0080 | DC7状态 |
| 0x0100 | FAN状态 |
| 0x0200 | HEAT状态 |

**6.错误代码说明**

|  |  |
| --- | --- |
| 0x0001 | 设备类型错误 |
| 0x0002 | 继电器控制超出 |
| 0x0003 | 设备不支持功能 |
| 0x0004 |  |
| 0x0005 |  |
| 0x0006 |  |
| 0x0007 |  |
| 0x0008 |  |

1. **数据类型发码说明**
2. **搜索在线从机**

网关板发送：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | 数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | |
| F5 | 5F | 01 | 0006 | 01 | FF00 | FE |  | 0D | 0A |

从机板发送：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | 数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | |
| F5 | 5F | 01 | 0007 | 81 | FF00 | 02 |  | 0D | 0A |

1. **读电源板采集数据和继电器状态**

网关板发送：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | 数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | |
| F5 | 5F | 01 | 0006 | 02 | FF00 | 01 |  | 0D | 0A |

电源板发送：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | n数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | 2字节 | 2字节 | |
| F5 | 5F | 01 | 0007 | 82 | 0000 | 02 |  | 0D | 0A |

1. **写电源板继电器状态**

网关板发送：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | 数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | 2字节 | 2字节 | |
| F5 | 5F | 01 | 0007 | 03 | 0000 | 0001 |  | 0D | 0A |

电源板发送：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | | 设备地址 | 帧总长 | 功能码 | 数据帧 | | 数据CRC16 | 帧尾 | |
| 地址 | 数据内容 |
| 2字节 | | 1字节 | 2字节 | 1字节 | 2字节 | 2字节 | 2字节 | 2字节 | |
| F5 | 5F | 01 | 0007 | 83 | 0000 | 0001 |  | 0D | 0A |

1. **[CRC](https://so.csdn.net/so/search?q=CRC" \t "https://blog.csdn.net/qq_32761549/article/details/_blank)-16 MODBUS原理说明**

校验码的计算多项式为(X16 + X15 + X2 + 1)。具体CRC16码的计算方法是：

1．预置1个16位的寄存器为十六进制FFFF（即全为1）；称此寄存器为CRC寄存器；

2．把第一个8位二进制数据 （既通讯信息帧的第一个字节）与16位的CRC寄存器的低8位相异或，把结果放于CRC寄存器；

3．把CRC寄存器的内容右移一 位（朝低位）用0填补最高位，并检查右移后的移出位；

4．如果移出位为0：重复第3步（再次右移一位）；

如果移出位为1：CRC寄存器与多项式A001（1010 0000 0000 0001）进行异或；(Modbus)

5．重复步骤3和4，直到右移8次，这样整个8位数据全部进行了处理；

6．重复步骤2到步骤5，进行通讯信息帧下一个字节的处理；

7．将该通讯信息帧所有字节按上述步骤计算完成后，得到的16位CRC寄存器的高、低字节进行交换；

8．最后得到的CRC寄存器内容即为：CRC码。

源码：

unsigned char code TabH[] = //CRC高位字节值表

{

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

} ;

unsigned char code TabL[] = //CRC低位字节值表

{

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 int GetCRC16(unsigned char \*ptr, unsigned char len)

{

unsigned int index;

unsigned char crch = 0xFF; //高CRC字节

unsigned char crcl = 0xFF; //低CRC字节

while (len--) //计算指定长度的CRC

{

index = crch ^ \*ptr++;

crch = crcl ^ TabH[ index];

crcl = TabL[ index];

}

return ((crch<<8) | crcl);

}