博瑞程控电源系列Modbus协议通信协议

**端口参数配置**

波特率：9600

版本：VER 04

数据位：8

停止位：1

奇偶校验：无

寄存器定义：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **寄存器地址** | **名称** | **03功能码** | **06功能码** | **10功能码** | **说明** |
| 0x0000 | Remote Mode | Y | Y | N | 远程控制状态 0:本地模式 1:远程模式 |
| 0x0001 | V\_SET | Y | N | Y | 电压设定寄存器 float型 |
| 0x0003 | A\_SET | Y | N | Y | 电流设定寄存器 float型 |
| 0x001B | OUTPUT | Y | Y | N | 输出状态设定寄存器 0:关闭输出 1:打开输出 |
| 0x001D | V\_OUT | Y | N | N | 输出实际电压寄存器 float型 |
| 0x001F | A\_OUT | Y | N | N | 输出实际电流寄存器 float型 |
| 0x0021 | CV/CC | Y | N | N | 输出工作状态寄存器 0:CV状态 1:CC状态 |

注意：一个寄存器是16位(2个字节)的数据单元，所有float型参数都占用两个寄存器地址

功能码表及间隔时间如下：

|  |  |  |
| --- | --- | --- |
| 功能码 | 对应功能 | 两次操作间隔时间 |
| 0x03 | 连读一个或多个寄存器命令 | N \* 5ms |
| 0x06 | 单写一个寄存器命令 | 10ms |
| 0x10 | 连写一个或多个寄存器命令 | N \* 5ms |

注：N为寄存器个数

通信协议格式如下：

功能码0x03(读保持寄存器)

PC发送:8 Byte

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 功能码 | 起始地址高字节 | 起始地址低字节 | 寄存器个数高字节 | 寄存器个数低字节 | CRC16校验码低字节 | CRC16校验码高字节 |
| 0x01 | 0x03 |  |  |  |  |  |  |

电源返回: 5+N\*2 Byte

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 功能码 | 数据长度字节 | 返回数据  高字节 | 返回数据  低字节 | 返回数据N+1  高字节 | 返回数据N+1  低字节 | CRC16校验码低字节 | CRC16校验码高字节 |
| 0x01 | 0x03 |  |  |  |  |  |  |  |

举例一：读单个寄存器(输出电压值0x001D)

PC发送：01 03 00 1D 00 02 54 0D (开始地址为0x001D，长度为2个寄存器(float型数据)，CRC16校验结果为0x0D54)

电源返回：01 03 04 40 A0 00 00 EF D1 (0x04位数据长度，0x40A00000为5.000V，0xD1EF为CRC16校验结果)

举例二：读多个寄存器（输出电压电流值0x001D~0x001F)

PC发送：01 03 00 1D 00 04 D4 0F (开始地址为0x001D，长度为4个寄存器，CRC16校验结果为0x0FD4)

电源返回：01 03 08 40 A0 00 00 40 00 00 00 24 2D (0x08位数据长度，0x40A00000表示5.00V，0x40000000表示2.0000A，0x2D24为CRC16校验结果)

功能码0x06(写保持寄存器)

PC发送: 8 Byte

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 功能码 | 写入地址高字节 | 写入地址低字节 | 写入数据高字节 | 写入数据低字节 | CRC16校验码低字节 | CRC16校验码高字节 |
| 0x01 | 0x06 |  |  |  |  |  |  |

电源返回： 8Byte

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 功能码 | 写入地址高字节 | 写入地址低字节 | 写入数据高字节 | 写入数据低字节 | CRC16校验码低字节 | CRC16校验码高字节 |
| 0x01 | 0x06 |  |  |  |  |  |  |

注意：所有float型参数寄存器都不支持06码,06码只支持单个寄存器操作

举例一：远程控制状态(0x0000)

PC 发送：01 06 00 00 00 01 48 0A (写入地址为0x0000，写入数据为0x0001，CRC16校验结果为0x480A)

电源返回：01 06 00 00 00 01 48 0A (写入地址为0x0000，写入数据为0x0001，CRC16校验结果为0x480A)

举例二：打开电源输出(0x001B)

PC 发送：01 06 00 1B 00 01 38 0D (写入地址为0x001B，写入数据为0x0001，CRC16校验结果为0x380D)

电源返回：01 06 00 1B 00 01 38 0D (写入地址为0x001B，写入数据为0x0001，CRC16校验结果为0x380D)

功能码 0x10(写多个保持寄存器)

PC发送: 9+N\*2 Byte

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 功能码 | 起始地址高字节 | 起始地址低字节 | 寄存器个数高字节 | 寄存器个数低字节 | 数据长度  字节 | 写入数据  高字节 | 写入数据  低字节 | 写入数据N+1  高字节 | 写入数据N+1  低字节 | CRC16校验码低字节 | CRC16校验码高字节 |
| 0x01 | 0x10 |  |  |  |  |  |  |  |  |  |  |  |

电源返回: 8 Byte

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 功能码 | 起始地址高字节 | 起始地址低字节 | 数据长度  高字节 | 数据长度  低字节 | CRC16校验码低字节 | CRC16校验码高字节 |
| 0x01 | 0x10 |  |  |  |  |  |  |

举例一：同时设定电压电流值(0x0001~0x0003)

PC 发送：01 10 00 01 00 04 08 40 A0 00 00 40 00 00 00 FA 43 (开始地址为0x0001, 长度为4个寄存器(8字节), 0x40A00000为5.00V/40 00 00 00为1.0000A,

CRC16校验结果为0x43FA)

电源返回：01 10 00 01 00 04 90 0A (开始地址为0x0001, 4个寄存器,CRC16校验结果为0x0A90)

CRCR16计算代码：

const unsigned char CRCHTalbe[] =

{

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 unsigned char CRCLTalbe[] =

{

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 crc16(unsigned char \*DData,unsigned char len)

{

unsigned char CRCHi = 0XFF;

unsigned char CRCLo = 0XFF;

unsigned int wlndex;

unsigned int CRC\_DData;

while(len--)

{

wlndex = CRCLo ^ \*DData++;

CRCLo = CRCHi ^ CRCHTalbe[wlndex];

CRCHi = CRCLTalbe[wlndex];

}

CRC\_DData = CRCHi;

CRC\_DData<<= 8;

CRC\_DData |=CRCLo;

return CRC\_DData;

}

测试流程

注意：设备通电后默认是本地模式，测试MODBUS通讯其他命令前必须先配置成远程模式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 |  | 主机发送 | 电源回复 | 备注 |
| 1 | 配置成远程模式 | 01 06 00 00 00 01 48 0A | 01 06 00 00 00 01 48 0A | 连接设备 |
| 2 | 单写电压 | 01 10 00 01 00 02 04 41 40 00 00 27 8B | 01 10 00 01 00 02 10 08 |  |
| 3 | 单写电流 | 01 10 00 03 00 02 04 41 00 00 00 A7 86 | 01 10 00 03 00 02 B1 CB |  |
| 4 | 连续写电压电流 | 01 10 00 01 00 04 08 41 70 00 00 42 0C 00 00 2B F9 | 01 10 00 01 00 04 90 0A |  |
| 5 | 连续读5个寄存器(包含电流、电压、CC/CV状态) | 01 03 00 1D 00 05 15 CF | 01 03 0A 41 70 00 00 3F FA C0 83 00 00 02 96 | 新版本 |
| 6 | 连续读6个寄存器(包含电流、电压、CC/CV状态) | 01 03 00 1D 00 06 55 CE | 01 03 0C 41 20 00 00 40 3E B8 52 00 00 00 00 BD 0B | 旧版本 |
| 7 | 打开电源 | PC发送：01 06 00 1B 00 01 38 0D | 01 06 00 1B 00 01 38 0D |  |
| 8 | 关闭电源 | PC发送：01 06 00 1B 00 00 F9 CD | 01 06 00 1B 00 00 F9 CD |  |
| 9 | 配置成本地模式 | PC发送：01 06 00 00 00 00 89 CA | 01 06 00 00 00 00 89 CA | 断开设备 |