Rev1.0

GDS20 系列可编程开关直流电源 Modbus 编程指南

版权所有翻印必究 爱科赛博 ACTIONPOWER 如有变更恕不通知

1.通信规约

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

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，CRC16 校验结果为 0X0D54）

电源返回：01 03 04 40 A0 00 00 EF D1 （0X40A00000 为 5.000V，0XD1EF 为 CRC16 校验结果）

举例二：读输出电压电流值(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　　（0X40A00000 表示 5.00V，0X40000000 表示 2.0000A，0X2D24 为 CRC16 校验结果）

功能码 0X06

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：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 0X0D38）

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 0X0A48）

功能码 0X10

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，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)

2. 通信协议

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	Y	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
0X0005	V_MIN	Y	N	Y	最小电压设定寄存器 float 型
0X0007	V_MAX	Y	N	Y	最大电压设定寄存器 float 型
0X0009	A_MIN	Y	N	Y	最小电流设定寄存器 float 型
0X000B	A_MAX	Y	N	Y	最大电流设定寄存器 float 型
0X000D	OVP_SET	Y	N	Y	过压参数设定寄存器 float 型
0X000F	OCP_SET	Y	N	Y	过流参数设定寄存器 float 型
0X0011	OVP_STATE	Y	Y	Y	过压状态设定寄存器 0:关闭保护 1:打开保护
0X0012	OCP_STATE	Y	Y	Y	过流状态设定寄存器 0:关闭保护 1:打开保护
0X0013	OUTPUT	Y	Y	Y	输出状态设定寄存器 0:关闭输出 1:打开输出
0X0014	STATE	Y	Y	Y	状态读取清除寄存器 u16 型，参考状态寄存器位定义表
0X0015	V_OUT	Y	N	N	输出实际电压寄存器 float 型
0X0017	A_OUT	Y	N	N	输出实际电流寄存器 float 型
0X0019	CV/CC	Y	N	N	输出工作状态寄存器 0:CV 状态 1:CC 状态

状态寄存器(0X0014)位定义表：

Bit	名称	属性	说明
2	OTP_FLAG	W/R	1 为过温保护标记， 写 1 清除过温标记（温度回到安全值时）
1	OCP_FLAG	W/R	1 为过流保护标记， 写 1 清除过流标记
0	OVP_FLAG	W/R	1 为过压保护标记， 写 1 清除过压标记

注意：所有 float 型参数都占用两个寄存器地址

附录

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 wIndex;
   unsigned int CRC_DData;
   while(len--)
    {
    wIndex = CRCLo ^ *DData++;
    CRCLo = CRCHi ^ CRCHTalbe[wIndex];
    CRCHi = CRCLTalbe[wIndex];
    }
CRC_DData = CRCHi;
CRC_DData<<= 8;
CRC_DData |=CRCLo;
   return CRC_DData;
}
```