VTC 系列智能数字温控模块通迅手册

【RS-485】通迅（部份特殊型号有 RS-232 接口）

1. 通迅参数

●支持传输速度（波特率）4800，9600，19200，38400bps;出厂设定为 38400bps（可在产品上侧指拨开关②设置）;

- 设备地址出厂为 1（可在产品上侧指拨开关①设置）
- 数据位：8 位；
- 偶校验；
- 停止位：1 位；

2. 数据寄存器地址及内容（ModbusRTU 协议保持寄存器 4x 区）

表一：设定与调试寄存器

寄存器地址		默认值	数据	名称	说明
16#格式	10#格式	浮点格式	长度		
4 路温度值（只读）					
046CH	1132		2	第一路温度值	第一路传感器测量温度
046EH	1134		2	第二路温度值	第二路传感器测量温度
0470H	1136		2	第三路温度值	第三路传感器测量温度
0472H	1138		2	第四路温度值	第四路传感器测量温度
基本参数（可读可写）					
0474H	1140	200.50	2	温度上限	温度限定的最大值
0476H	1142	50.00	2	设定温度	设定需要的控制温度
0478H	1144	30.00	2	温度下限	温度限定的最小值
047AH	1146	65.00	2	比例系数	比例增益 1-9999（正向控制）；-1 至-9999（反向控制）；
047CH	1148	45.00	2	积分系数	0-9999
047EH	1150	12.00	2	微分系数	0-9999
高级参数（可读可写）					
0480H	1152	1.00	2	比例因子	0.1-1.0 弱化比例系数的参数（一般不作调整）
0482H	1154	5.00	2	微分因子	0.1-10.0（一般不作调整）
0484H	1156	10.00	2	控制带值	1.0-100.0；PID 在偏差值正负此值范围内工作(-10>偏差值<10)；条件：控制带开关打开
0486H	1158	3000.00	2	激励源	1-控制周期；自整定时脉宽以此值输出；
0488H	1160	0.00	2	手动值	手动控制时以此值输出占空比
048AH	1162	0.00	2	死区宽度	0.0-10.0； PID 在偏差值正负此值范围内不工作(一般都是为 0)
048CH	1164	4000.00	2	控制周期	输出值灵敏度(脉宽周期)4 路必须一致，重新上电生效!
调试观察参数（只读）					
048EH	1166		2	传感器温度	当前传感器温度；
0490H	1168		2	传感器滤波后温度	经过 PID 后的当前传感器温度；

0492H	1170		2	调节变量	PID 运算后的脉宽输出量；
0494H	1172		2	偏差值	当前温度与设定温度的偏差；
0496H	1174		2	过程值 P	运算过程中比例参数；
0498H	1176		2	过程值 I	运算过程中积分参数；
049AH	1178		2	过程值 D	运算过程中微分参数；
049CH	1180		2	传感器温度	冷端补偿前温度（热电偶传感器用）
硬件误差修正（可读可写）					
049EH	1182		2	误差修正	修正由传感器或接线端引起的温度误差值

表二：状态寄存器

寄存器地址		默认值	数据	名称	说明
16#格式	10#格式	(32 位无符号整数)	长度		
自整定观察参数 （只读）					
04CCH	1228	0	2	自整定阶段	（0-7）整定开始为 2；整定结束为 0
04CEH	1230	0	2	自整定结果	用此值来判断自定的理想程度(结合经验值) 一般为系统自用
04D0H	1232	0	2	控制器类型	用此值来判断加热器的类型(结合经验值) 一般为系统自用

表三：开关控制寄存器

寄存器地址		默认值	数据	名称	说明
16#格式	10#格式	(16 位无符号整数)	长度		
控制参数 （可读可写）					
04E0H	1248	0	1	手动开关	置 1 时为手动控制； 置 0 为自动控制
04E1H	1249	0	1	复位开关	值置 1 时复位自整定与相关参数后自动归 0
04E2H	1250	0	1	控制带开关	置 1 时控制带值生效；置 0 时控制带值无效
04E3H	1251	0	1	自整定开关	置 1 时自整定开始，整定结束自动归 0；
04E5H	1253	0	1	参数永久保存开关	置 1 时启动基本参数与高级参数写入 EEPROM；写入完成自动归 0；
04E6H	1254	0	1	传感器类型选择	热电偶 （0：K 型； 1：J 型） 热电阻 （0：PT100； 1：PT1000）
04E7H	1255	1	1	控温开关	0：关闭输出； 1：打开输出

注意：以上所有寄存器地址都是为第一路传感器的参数地址 如需操作同一模块第二路传感器参数时只需要把地址值加上 128； 第三路加上 256；第四路加上 384；即可。

表四：1 至 4 路传感器报警输出设置

寄存器地址： 1780(十进制) ；6F4H（十六进制）；数据长度：1(16Bit)；可读可写； 以下是 0-16Bit 说明															
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
第四路	第三路	第二路	第一路	第四路	第三路	第二路	第一路	第四路	第三路	第二路	第一路	第四路	第三路	第二路	第一路
状态值	状态值	状态值	状态值	状态值	状态值	状态值	状态值	设置值	设置值	设置值	设置值	设置值	设置值	设置值	设置值
1 为已发生断线报警； 0 为无发生断线报警：	1 为已发生断线报警； 0 为无发生断线报警：	1 为已发生断线报警； 0 为无发生断线报警：	1 为已发生断线报警； 0 为无发生断线报警：	1 为已发生超限报警； 0 为无发生超限报警：	1 为已发生超限报警； 0 为无发生超限报警：	1 为已发生超限报警； 0 为无发生超限报警：	1 为已发生超限报警； 0 为无发生超限报警：	1 为开启传感器断线检测； 0 为关闭传感器掉线检测；	1 为开启传感器断线检测； 0 为关闭传感器掉线检测；	1 为开启传感器断线检测； 0 为关闭传感器掉线检测；	1 为开启传感器断线检测； 0 为关闭传感器掉线检测；	1 为超出上限报警输出； 0 为低于下限报警输出；	1 为超出上限报警输出； 0 为低于下限报警输出；	1 为超出上限报警输出； 0 为低于下限报警输出；	1 为超出上限报警输出； 0 为低于下限报警输出；

3. 通迅功能码与通迅格式

1、本模块支持的 ModbusRTU 协议功能码

表五：

功能码	名称	说明
03H	读多个保持寄存器	读取一个或多个 16 位二进制保持寄存器；（本模块作为从站是被读取）
06H	写一个保持寄存器	写入一个 16 位二进制保持寄存器；（本模块作为从站是被写入）
10H	写多个保持寄存器	写入多个 16 位二进制保持寄存器；（本模块作为从站是被写入）

2、RTU 通迅格式

表六：

上位机读取(03H 功能码)		模块回复数据(03H 功能码)		上位机写入(10H 功能码)		模块回复数据(10H 功能码)	
模块地址	01H	模块地址	01H	模块地址	01H	模块地址	01H
功能码	03H	功能码	03H	功能码	03H	功能码	03H
读取数据起始地址	04H	回复数据长度：（字节）	04H	写入数据地址	04H	写入数据地址	04H
	74H				74H		74H
读取数据长度单位:(16bit/字)	00H	回复数据：（高位字节在前；低位字节在后）	43H	写入数据长度单位:(16bit/字)	00H	写入数据长度单位:(16bit/字)	00H
	02H		48H		02H		02H
CRC 低字节	85H		80H	写入数据长度：（字节）	04H	**CRC** 低字节	00H
CRC 高字节	21H		00H	写入数据内容：（高位字节在前；低位字节在后）	43H	**CRC** 高字节	E2H
		CRC 低字节	0EH		48H		
		CRC 高字节	61H		80H		
					00H		
				CRC 低字节	33H		
				CRC 高字节	EAH		

上位机写入(06H 功能码)		模块回复数据(06H 功能码)	
模块地址	01H	模块地址	01H
功能码	06H	功能码	06H
写入数据地址	04H	写入数据地址	04H
	E0H		E0H
写入数据内容	00H	写入数据内容	00H
	00H		00H
CRC 低字节	89H	**CRC** 低字节	89H
CRC 高字节	0CH	**CRC** 高字节	0CH

以上三条指令都是根据本公司产品实际应用测试出来的真实数据，可作为参考！

第一条：（03H 功能码）是读模块的温度上限值，地址是 0474H，数据长度是 0002H，根据模块回复的数据可以看到得到的数据内容是 43H、48H、80H、00H 共 4 个字节，由于我们模块数据格式采用的是浮点数（出于高精度考虑）所以把 43H、48H、80H、00H 按高字节在前低子节在后转换成浮点数的数据是 200.5 度；本案例为了便于理解只读了一个浮点数（温度上限值），实际应用也可以一次读出多组连续地址的值；

第二条：(10H 功能码) 是设置模块的温度上限值；地址与数据与 03H 指令一样，就是把 200.5 度数据写入到模块地址为 0474H 的寄存器里；

第三条：(06H 功能码) 是写模块手动开关，地址是 04E0H，因为 06H 功能码是固定写一个字长度（16bit）所以就

没用了数据长度这一项 ;此例是把数据 0000H 写入到模块,从表三寄存器说明可以得出这条指令是把模块改为 PID 控制模式;

CRC 校验码:

CRC（Cyclical Redundancy Check）校验码是由以下方法计算得出

步骤一：加载一值为FFFFH 的16 位寄存器，称为CRC 寄存器。

步骤二：数据的第一字节和CRC 寄存器的低字节作异或门运算，并将运算结果放回CRC 寄存器。

步骤三：将CRC 寄存器位右移并将最高位填零，并检查移出之最低字节。

步骤四：如果移出的最低字节为0 重复步骤三，否则将CRC 寄存器与值A001H 作异或门运算，并将运算结果放回CRC 寄存器。

步骤五：重复步骤三及四，直到8 个位皆完成右移。如此一个字节便完成。

步骤六：重复步骤二及五，将数据内所有字节计算一次便可得出CRC 校验码。

请特别注意传收数据格式中CRC 寄存器的高、低字节传送顺序。

CRC校验 C 语言程序范例:

```
unsigned int   reg_crc   =   0xffff;
byte j, i   =   0;
while (length--)
{
  reg_crc ^= RTUData[i];
     i ++;
  for (j = 0; j < 8; j++)
  {
    if (reg_crc & 0x01)
        reg_crc = (reg_crc >> 1) ^ 0xA001;
    else
        reg_crc = reg_crc >> 1;
    }
}
return(reg_crc);
```

浮点数转 字节 C语言程序范例:

```
float   Tem = 200.5 ;//温度上限值
unsigned   int * pTem = (unsigned   int *) Tem; //定义一个无符合 32 位整数指针指向 Tem（温度上限）
byte   Tbyte[4];//定义一个字节数组用来存放浮点数转过来的 4 个字节（一个 float 数据类型点用 4 个 byte）
Tbyte [0] =   (* pTem ) >> 24;//取最高 8 位
Tbyte [1] =   (* pTem ) >> 16 & 0xff;
Tbyte [2] =   (* pTem ) >> 8 & 0xff;
Tbyte [3] =   (* pTem ) & 0xff; //取最低 8 位
//转完得出结果是 Tbyte [0] = 0x43; Tbyte [1] = 0x48; Tbyte [2] = 0x80; Tbyte [3] = 0x00;
```

字节 转 浮点数 C语言程序范例:

```
float   Tem;//温度上限值
unsigned   int * pTem = (unsigned   int *) Tem; //定义一个无符合 32 位整数指针指向 Tem（温度上限）
byte   Tbyte[] = {0x43,0x48,0x80,0x00};
* pTem = Tbyte [0]<<24 | Tbyte [1]<<16 | Tbyte [2]<<8 | Tbyte [3];
//转完得出结果是 Tem = 200.5;
```

了解更多资讯请常关注本公司官网
Http://www.szviena.com
E-mail:liaozlei@szviena.com
公司电话：0755-29866780
技术服务电话：13590449918