D-8K2 8 路热电偶采集模块使用说明书

通讯协议 MODBUS（RTU 模式）

一、 概览

本产品主要测量 8 路 K 型热电偶传感器的温度，测温从-50℃~1300℃，精度高，可分辨到 0.1℃；高可靠性，输入输出隔离，RS485 和 RS232 输出可自定义选择使用，通讯方式通过拔码开关设定，宽电源供电，使用方面;工业产品设计，输入输出多重输入保护。

二、 产品指标

输入类型：K 型热电偶
测温范围：-50～1300℃
供电电源：9-30V DC；
通讯协议：MODBUS 协议；
通讯速率：9600/19200/38400/115200 bps；
数据更新周期：<1.5 秒；
外形尺寸：115*90*40 mm;

输入路数：8 路；
分 辩 率：0.1℃；
功 耗：<1W；
输 出：RS485/RS232；
校验方式：无/奇/偶校验可选；
工作温度：-40℃－70℃；

三、 端子接线说明

图 1：D-8K2 温度采集模块

接口	功能定义	连接说明	接口	功能定义	连接说明
1	RS232/RS485 通信接口	TXD/485B-	15	电源接口 6~30VDC	电源 VCC
2		RXD/485A+	16		电源地 GND
3		GND	17	空	NC
4	第 1 路	COM 1+	18	第 5 路	COM 5+
5		COM 1-	19		COM 5-
6	空	NC	20	空	NC

7	第 2 路	COM 2+	21	第 6 路	COM 6+
8		COM 2-	22		COM 6-
9	空	NC	23	空	NC
10	第 3 路	COM 3+	24	第 7 路	COM 7+
11		COM 3-	25		COM 7-
12	空	NC	26	空	NC
13	第 4 路	COM 4+	27	第 8 路	COM 8+
14		COM 4-	28		COM8-

特别说明：

1、供电电源使用 9-30V 直流电源，推荐使用 12V 或者 24V。（请一定要注意安全用电）

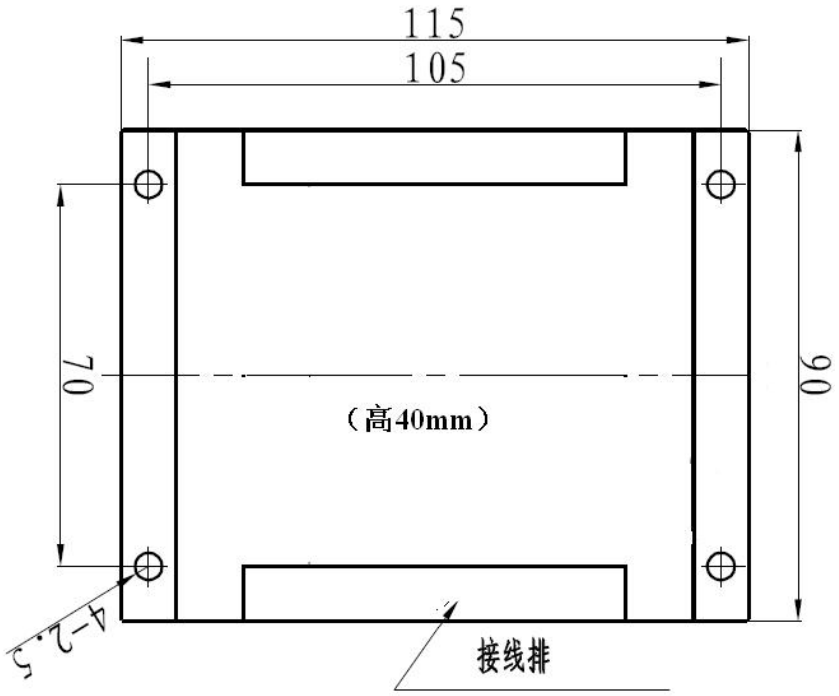

图 4、产品装配尺寸图

四、 通信设置

所有设置在更改后，需要重新开机才生效；拨码开关需打开产品外壳，在产品内部。

拨码开关一共有 10 个位，分别是 B1～B10，具体功能如下：

B1～B5 位用于设置通信地址；

B6～B7 位用于设置波特率；

B8 位用于选择 232/485 通信；

B9～B10 用于设置奇偶校验位；

4.1、通信地址（出厂默认值是 1）

*说明： B1 对应地址的高位，B5 对应地址的低位；“ON”状态表示逻辑“1”， “OFF”状态表示逻辑“0”。

B1	B2	B3	B4	B5	地址十六进制值	十进制值
0	0	0	0	1	01H	1
0	0	0	1	0	02H	2
0	0	0	1	1	03H	3
0	0	1	0	0	04H	4
0	0	1	0	1	05H	5

0	0	1	1	0	06H	6
0	0	1	1	1	07H	7
0	1	0	0	0	08H	8
0	1	0	0	1	09H	9
0	1	0	1	0	0AH	10
0	1	0	1	1	0BH	11
0	1	1	0	0	0CH	12
0	1	1	0	1	0DH	13
0	1	1	1	0	0EH	14
0	1	1	1	1	0FH	15
1	0	0	0	0	10H	16
1	0	0	0	1	11H	17
1	0	0	1	0	12H	18
1	0	0	1	1	13H	19
1	0	1	0	0	14H	20
1	0	1	0	1	15H	21
1	0	1	1	0	16H	22
1	0	1	1	1	17H	23
1	1	0	0	0	18H	24
1	1	0	0	1	19H	25
1	1	0	1	0	1AH	26
1	1	0	1	1	1BH	27
1	1	1	0	0	1CH	28
1	1	1	0	1	1DH	29
1	1	1	1	0	1EH	30
1	1	1	1	1	1FH	31

4.2、波特率（B6、B7：设置波特率，出厂设置默认是 9600）

B6=OFF，B7=OFF，9600bps
B6=OFF，B7=ON，19200bps
B6=ON，B7=OFF，38400bps
B6=ON，B7=ON，115200bps

4.3、通信接口选择（出厂设置默认是 B8=OFF，485 通信模式）

232 通信：拨码开关的第 8 位拨到“ON”位置，并把 2 个跳线帽接到 232 的位置。
485 通信：拨码开关的第 8 位拨到“OFF”位置，并把 2 个跳线帽接到 485 的位置。

4.4、奇偶校验位（B9、B10：设置校验位，出厂设置默认是：无校验）

B9=OFF，B10=OFF，无校验
B9=OFF，B10=ON，奇校验
B9=ON，B10=OFF，偶校验

五、控制与应用

5.1 采集模块作为从设备（从机/从站）使用，上电后 LED 不断闪烁表示模块正常工作，采集模块只有在接收到正确的主机命令时才会返回数据。

5.2 通讯方式：串行通讯，波特率和奇偶校验位的设置请参照“**通信设置**”部分。

5.3 数据保持寄存器长度 16bit，一个寄存器的数据分 2 字节发送，先高位字节后低位字节。

5.4 数据保持寄存器地址及定义：（寄存器长度 16bit）

序号	寄存器地址	数 据 定 义
1 路	0000	1、数据是 16 位（bit）有符号二进制数据（2 字节）。 2、采集的温度精确到 0.1 摄氏度。 3、寄存器的温度数据扩大 10 倍处理（避免浮点数）。 4、温度值 = 寄存器的数据 ÷ 10（请参考下表例子）。
2 路	0001	
3 路	0002	
。。。	。。。。。。	
6 路	0005	
7 路	0006	
8 路	0007	

说明:0008H-000FH 寄存器为温度修正寄存器,以 0.1 度为单位,如需要增加 2 度,写入 20 数据即可,采用 06 功能码,温度减少需写入负数采用补码方式写入.

5.5 保持寄存器的温度数据举例说明：

寄存器的数据(2 字节)	二进制值（16bit）	十进制值	对应温度值
04E2H	0000 0100 1110 0010	1250	+125℃
0293H	0000 0010 1001 0011	659	+65.9℃
0066H	0000 0000 0110 0110	102	+10.2℃
0005H	0000 0000 0000 0101	5	+0.5℃
0000H	0000 0000 0000 0000	0	0℃
FFF8H	1111 1111 1111 1000	-8	-0.8℃
FE81H	1111 1110 1000 0001	-383	-38.3℃
FDDAH	1111 1101 1101 1010	-550	-55℃
8000H	1000 0000 0000 0000	-32768	找不到传感器

5.6 通讯协议数据格式（读数据寄存器功能码：03H）

主站请求帧

地址	1 字节	
功能码	1 字节	03H
起始地址	2 字节	0000H ~ 0007H （0~7）
寄存器数量	2 字节	0001H ~ 0008H （1~8）
CRC 校验	2 字节	

从机响应帧

地址	1 字节	
功能码	1 字节	03H
字节数	1 字节	2×N（N 是寄存器数量）
寄存器值	2×N 字节	
CRC 校验	2 字节	

主站请求帧发送数据为 8 字节：（主机到模块）

第 1 字节	第 2 字节	第 3~4 字节	第 5~6 字节	第 7~8 字节
模块地址	功能码（0x03）	寄存器起始地址	寄存器数量	CRC 校验

*说明：模块只有 8 个数据寄存器，因此必须要满足：寄存器起始地址+寄存器数量<=8。

从机响应帧数据为 2N+5 字节：（模块到主机）

第 1 字节	第 2 字节	第 3 字节	第 3~2N+3 字节	2N+4、2N+5 字节
模块地址	功能码（0x03）	字节数（2×N）	2×N 字节数据	CRC 校验

*说明：字节数（2×N）就是请求发送的 N 个寄存器的数据的字节数，也就是 2×N。

5.7 读寄存器数据示例：（假设模块地址是 01，数据为十六进制格式）

主机请求帧： 01 03 00 00 00 01 84 0A
地址 功能码 起始地址 寄存器数量 CRC 校验码

模块响应帧： 01 03 02 00 CB F9D3
地址 功能码 字节数 电流（电压）数据 CRC 校验码

其中数据： 00 CB（0x00CB 转换为十进制数是 203）
对应的温度值 = 203 ÷ 10 = 20.3
所以得到的温度是：+20.3 ℃

六、使用异常情况说明：

当主机发送的数据帧出错或从机接收不正确时，从机（模块）不应答。
常见出错的原因包括（但不限于）以下几种：
1、 连接异常
2、 通讯设置不正确
3、 地址不对
4、 CRC 校验出错
5、 数据长度出错
6、 操作的数据地址超出范围
7、 数据帧不符合要求

五、MODBUS 校验码和 CRC16 求值函数

/* CRC 高位字节值表*/

```
const unsigned char code 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
} ;
/* CRC 低位字节值表*/
const unsigned char code 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
} ;

/* CRC 求值函数*/
unsigned int crc16(uchar *puchMsg, uint usDataLen)
{
    uchar uchCRCHi = 0xFF ;                    /* 高 CRC 字节初始化 */
    uchar uchCRCLo = 0xFF ;                    /* 低 CRC 字节初始化 */
```

```
    uint    uIndex ;                                /* CRC 循环中的索引  */
    while (usDataLen--)                             /*  传输消息缓冲区  */
    {
        uIndex = uchCRCHi ^ *puchMsg++ ;            /*  计算 CRC */
        uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex] ;
        uchCRCLo = auchCRCLo[uIndex] ;
    }
    return (uchCRCHi << 8 | uchCRCLo) ;
}
```