六合一传感器模块（RS485）

功能：检测环境中的，CO2,甲醛，TVOC，PM2.5 及温度和湿度的实时数据

一、产品功能描述

六合一传感模块是一款集 CO2、激光粉尘、温湿度、TVOC　及甲醛于一体的综合型空气质量传感器。可对所处环境进行实时侦测，并通过一定的协议格式输出数据，使用方便，具有良好的稳定性。



图示仅供参考，具体以实物为准

二、产品应用领域

便携式仪表、、车载系统、空气质量监测设备、空气净化机、新风换气系统、空调、智能家居设备等场所。

三、产品技术指标

1、检测参数及分辨率

	测量分辨率	测量范围	测量精度
eCO2		400~5000ppm	±75ppm
PM2.5	0.3 ug/m³	0~1000 ug/m³	±10%
Temperature	0.01℃	0~60℃	±0.1
Humidity	0.04%	0~100%RH	±1%
TVOC	1ppb	0-1000ppb	
CH2O	1ppb	0-400ppb	

模块所测量的 CO2 值为 TVOC 的等效值输出，客户使用或购买时请注意。

2、工作电压：DC5V±0.2V（纹波不得超过 200mv）

3、工作环境：工作温度 0~50℃　　工作湿度：15%RH-90%RH（无凝结）

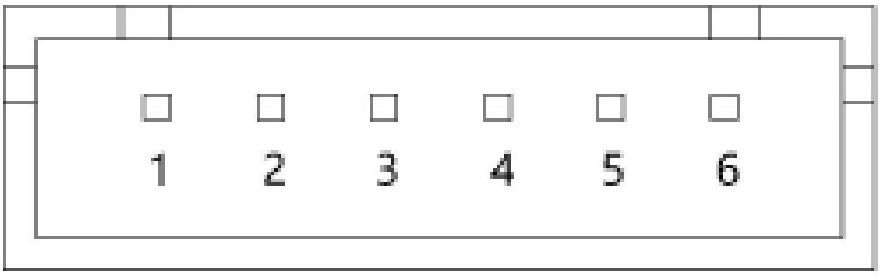
4、通讯方式：ModBus-RTU 485

5、预热时间：≤3 分钟

6、使用寿命：5 年（空气中）

四、产品规格

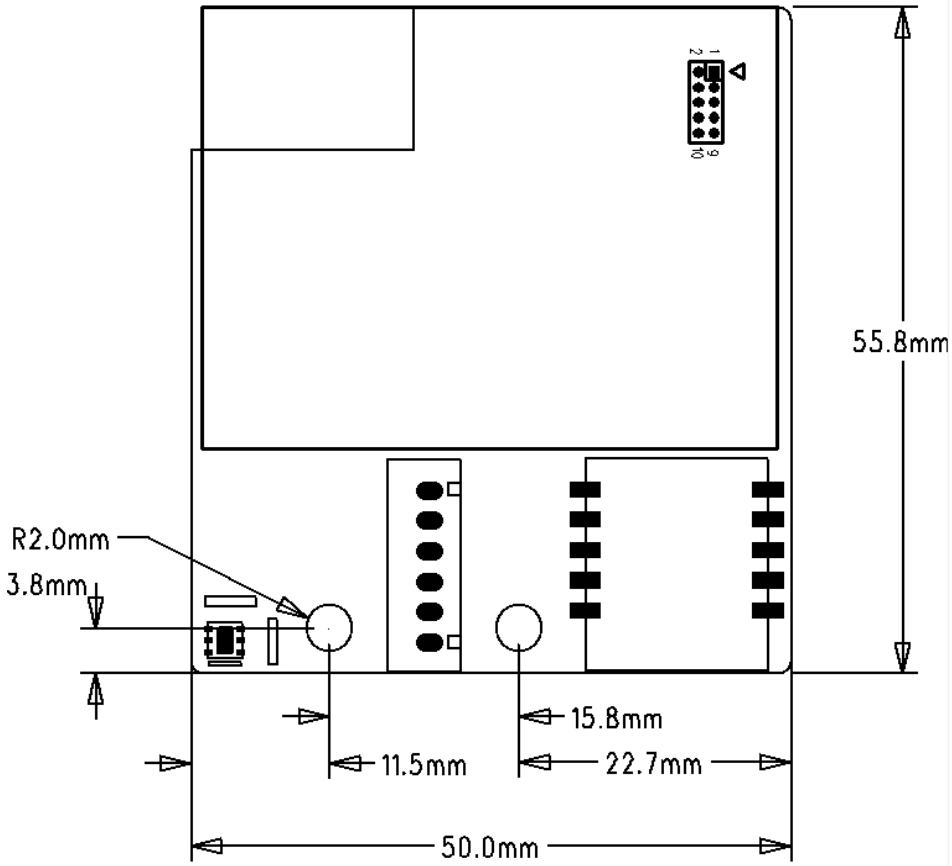
1、引脚定义



序号	名称	描述
1	SET	控制脚（高电平：开启测量-默认状态） （低电平：停止测量）
2	TX/B	串口发送端/RS485_B
3	RX/A	串口接收端/RS485_A
4	5V	DC5V 输入
5	PWM	预留（需要订制,只输出 PM2.5 数据））
6	GND	GND

备注：接口间距采用 2.54mm 间距，无特殊说明采用 90°插针接口。

2、外观尺寸图（55.8*50*15mm）



五、 通讯协议

波特率：9600，校验位，无，停止位，1 位）

返回数据时间：<300ms

读取速度不可超过 350ms

1. 读取地址命令：模块出厂默认地址：01　任意数值不可超过 254

xx	17	00	00	00	00	CRC 校验	
任意	功能码	任意	任意	任意	任意	CRC_L	CRC_H

返回数据：

xx	17	02	00	yy	CRC 校验	
与发送相同	功能码	字节数	保留	当前 ID	CRC_L	CRC_H

例：发送命令：00 17 00 00 00 00 74 18　返回数据：00 17 02 00 01 41 B4

2. 修改地址命令：修改范围为：0~254

yy	06	00	00	00	zz	CRC 校验	
当前地址	功能码	固定	固定	预留	预设地址	CRC_L	CRC_H

返回数据：

yy	06	02	00	zz	CRC 校验	
与发送相同	功能码	字节数	保留	修改后的 ID	CRC_L	CRC_H

例：当前地址为 01 时，预设地址为：02

发送命令：01 06 00 00 00 02 08 0B　返回数据：01 06 02 00 02 39 49

3. 读取数据命令

yy	03	00	MM	00	NN	CRC 校验	
当前地址	功能码	寄存器开始地址		预留	数据长度	CRC_L	CRC_H

返回数据：

yy	03	NN*2	xx xx ……xx xx	CRC 校验	
当前地址	功能码	数据长度	传感器数据	CRC_L	CRC_H

注意：返回的 xx xx ……xx xx 传感器数据可以根据寄存器地址和数据长度来改变。

00 MM 表示传感器的寄存器地址，00 NN 为数据长度。

寄存器地址（00 MM）					
00 00	00 01	00 02	00 03	00 04	00 05
CO2	TVOC	CH2O	PM2.5	H（湿度）	T（温度）

数据长度 00 NN 最低为 00 01，最大为 00 06 。MM 值为 00 时，数据长度 NN 值最大可为 06，此时能读出所有传感器的值，也可为 01，为 01 时只能单独读取粉尘的数据。寄存器靠前的地址，在加大数据长度时，可读取后面传感器的数据，但寄存器靠后的地址，不能读取在此地址前传感器的数据。

详情见下：

00 MM 00 NN 为 00 00 00 01 表示从起始地址 00 00 读取 CO2 数据。
00 MM 00 NN 为 00 00 00 02 表示从起始地址 00 00 读取 CO2，TVOC 数据。
00 MM 00 NN 为 00 00 00 03 表示从起始地址 00 00 读取 CO2，TVOC，CH2O 数据。
00 MM 00 NN 为 00 00 00 04 表示从起始地址 00 00 读取 CO2，TVOC，CH2O，PM2.5 数据。
00 MM 00 NN 为 00 00 00 05 表示从起始地址 00 00 读取 CO2，TVOC，CH2O，PM2.5，H 数据。
00 MM 00 NN 为 00 00 00 06 表示从起始地址 00 00 读取 CO2，TVOC，CH2O，PM2.5，H，T 数据.
00 MM 00 NN 为 00 01 00 01 表示从起始地址 00 01 读取 TVOC 数据。
00 MM 00 NN 为 00 01 00 02 表示从起始地址 00 01 读取 TVOC，CH2O 数据。
00 MM 00 NN 为 00 01 00 03 表示从起始地址 00 01 读取 TVOC，CH2O，PM2.5 数据。
00 MM 00 NN 为 00 01 00 04 表示从起始地址 00 01 读取 TVOC，CH2O，PM2.5，H 数据。
00 MM 00 NN 为 00 01 00 05 表示从起始地址 00 01 读取 TVOC，CH2O，PM2.5，H，T 数据。
00 MM 00 NN 为 00 02 00 01 表示从起始地址 00 02 读取 CH2O 数据。
00 MM 00 NN 为 00 02 00 02 表示从起始地址 00 02 读取 CH2O，PM2.5 数据。
00 MM 00 NN 为 00 02 00 03 表示从起始地址 00 02 读取 CH2O，PM2.5，H 数据。
00 MM 00 NN 为 00 02 00 04 表示从起始地址 00 02 读取 CH2O，PM2.5，H，T 数据.
00 MM 00 NN 为 00 03 00 01 表示从起始地址 00 03 读取 PM2.5 数据。
00 MM 00 NN 为 00 03 00 02 表示从起始地址 00 03 读取 PM2.5，H 数据。
00 MM 00 NN 为 00 03 00 03 表示从起始地址 00 03 读取 PM2.5，H，T 数据。
00 MM 00 NN 为 00 04 00 01 表示从起始地址 00 04 读取 H 数据。
00 MM 00 NN 为 00 04 00 02 表示从起始地址 00 04 读取 H，T 数据。
00 MM 00 NN 为 00 05 00 01 表示从起始地址 00 05 读取 T 数据。

例：地址为 01 时，发送命令：01 03 00 00 00 06 C5 C8 返回数据如下：

序号	名称	说明（十六进制）
1	地址	01

数据计算方法如下：

CO2 有效值 = CO2_H*256+CO2_L

TVOC 有效值 = TVOC_H*256+TVOC_L

CH2O 有效值 = CH2O_H*256+ CH2O_L

PM2.5 有效值 = PM2.5_H*256+ PM2.5_L

湿度有效值 = -6+125*S_{RH}/(2^16)

S_{RH}= Humidity_H*256+ Humidity_L

温度有效值 = -46.85+175.72* S_{Tem}/(2^16)

S_{tem}= Temperature_H*256+ Temperature_L

CRC 校验计算为：

选择 A001 或者 8005 逆序即可。

2	功能码	03
3	数据长度	0C
4	数据 1	CO2_H
5	数据 2	CO2_L
6	数据 3	TVOC_H
7	数据 4	TVOC_L
8	数据 5	CH2O_H
9	数据 6	CH2O_L
10	数据 7	PM2.5_H
11	数据 8	PM2.5_L
12	数据 9	Humidity_H
13	数据 10	Humidity_L
14	数据 11	Temperature_H
15	数据 12	Temperature_L
16	CRC16 校验	CRC16_L
17	CRC16 校验	CRC16_H

CRC 校验计算方法：

/*CRC 校验表高位*/

unsigned char 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 校验表低位   */
unsigned char 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 校验函数，生成 CRC*/
u32 CRC_Compute(u8 * pushMsg,u8 usDataLen)
{
u8 uchCRCHi = 0xff;//高 CRC 字节初始化
u8 uchCRCLo = 0xff;//低 CRC 字节初始化
u8 uIndex; //CRC 循环中的索引
while(usDataLen--)
{
uIndex = uchCRCLo^ *pushMsg++;//计算 CRC
uchCRCLo = uchCRCHi^ auchCRCHi[uIndex];
uchCRCHi = auchCRCLo[uIndex];
```

```
}
return (((u16)uchCRCHi) << 8 | uchCRCLo);
}
```