www.jingsudz.com 专业温湿度测量产品的研发生产与销售

RS485 总线的二次编程和调试

只要您的设备有 RS485 主站接口，软件支持 MODBUS-RTU 编程就可以使用我们的设备。阅读以下资料前请先阅读 MODBUS-RTU 协议，这个协议资料有几十页，没有看懂此协议前，看以下资料也可能会看不懂。

本方法需要一个 USB 转 485 或 232 转 485 转换器，用来监控 RS485 总线的数据，可以快速地解决编程上的问题。请下单购买的时候顺便购买一个，可以对设备进行调试，或作为检修工具用，

使用方法：

1：将模块的 485 与 PLC(文本等）和转换器的 485 都并联起来

下载打开模块对应的调试软件，对模块进行通信测试和数据测试，以确定模块工作正常，然后退出软件

2：下载一个串口调试助手

3：打开调试助手后设置好波特率，打开相应的串口号，设置为 HEX（16 进制）格式显示

4：编写您设备的程序并下载，使程序工作正常，这个时候可以从串口调试助手数据框中看到通信过程的数据，如果通信不上或读到的数据不对，请把串口助手显示框的数据截图给我们分析，可以快速地知道哪里错误。如果总线无数据请检查程序。（我们配的调试软件上面全部都有显示发送和接收的数据，显示的数据全部为十六进制，可以把您的软件发送的数据与我们软件的数据进行对比，如果发送的数据不对说明软件编写错误，如果调试助手功能上面不能显示有数据说明您的软件有问题或硬件接线错误）不管什么设备，什么语言写的程序总线的通信的数据一定是一样的，否则就是软硬件有问题。

电脑端软件的开发与调试：如果电脑端的软件与设备通信不上，请先用我们的调试软件调试，如果我们的软件可以通信上，那就一定是你们软件的问题，因为硬件设备不变，有些用户一定说自己软件没问题，我可以肯定的说一定是软件有问题，可能存在的问题有波特率、校验、停止位，命令不对等，

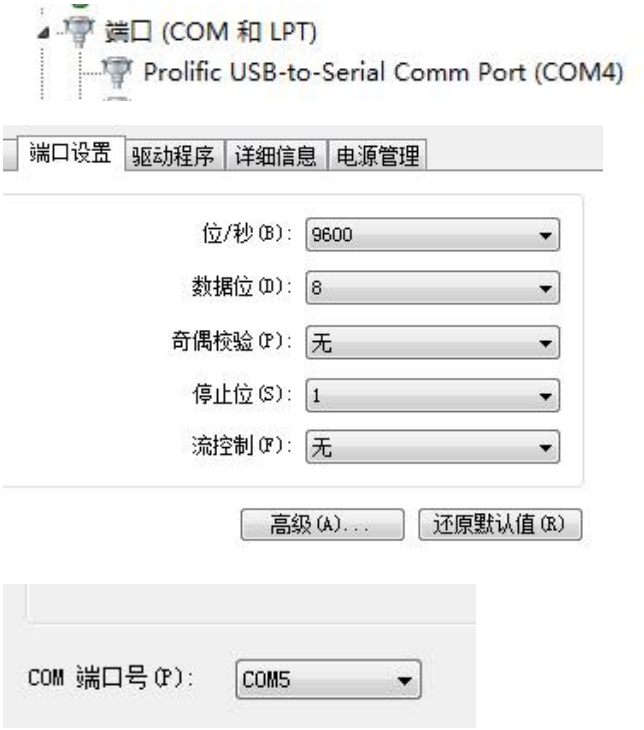
电脑端软件开发可以下载一个串口监控软件，此软件可以监控使用的串口的各项数据，如波特率、校验、停止位、通信中发送和接收的数据等，对比这些参数与我们软件的参数，就能知道是哪些参数不对，这些自行开发的软件问题只能用户自己解决，我们也不懂，因为我们只研究模块硬件和硬件程序和生产，上位机的应用开发是软件公司的专业。如果怀疑是我们软件的问题，可以下载 MODBUS-RTU 调试助手进行调试。

注意事项：为以更清楚地看到通信过程数据，请将上位机通信速度调到 1 秒或更慢。USB 转 485 转换器可能需要安装驱动。

通信不上检测要点：

RS485 总线有数据传输时 485 总线的电压会跳动，用万用表可以直接测量，如果 RS485 总线电压不跳动，说明总线没有数据输传，可检查接线，RS485 两根有正负极这分，不可接错，静态时 RS485 A+电压大于 B-，接反了电压就不对了。

如果是 USB 转 485 的转换器，部分电脑当 COM 号大于 9 时会出现串口无法打开的情况，请到设备管理器中右击 COM 口->属性->端口设置->高级->修改端口号后确定->确定，然后再重启电脑即可，



组态或 PLC 的编程注意事项：

我们一般采用 03 功能码进行数据的读取，对应 PLC 或文本的 400XX 地址。我们资料里给出的寄存地址全部为 10 进制，如 40 号寄存器，那么其 PLC 值为 40001+40=40041，以此类推。

有些软件的地址为 HEX（16 进制）格式那么 40 号寄存器的值为 28，以此类推。

没有 04 功能码的设备不支持 300XX 格式。

PLC 编程样例：

由于 PLC、文本、触摸屏厂类多，型号复杂，不如一一描述，但编程的原理都是一样的，

关于模块的指示灯说明：

绿灯：电源指示，一秒钟闪一次，(需要把传感器全部正确接上，此时不会亮红灯，否则会因为红灯太亮，看不到绿灯闪)

数据接收指示：当上位机有数据发来时会快闪，此灯可以断定 485 线上是否有数据发送过来，如果测试过程中此灯不闪，那么说明上位机无数据过来，需要检查线路、COM 口是否正确，485 是否接反，此灯不管设备地址是否正确，波特率是否对，只要 485 线上有数据就会闪。

红灯：错误指示，常亮，说明输入的传感器错误，断线，线序错，超量程等异常

发送数据指示，当模块接收到正常的数据指令后会返回数据，此时红灯会快闪，说明此时有数据发送到 485，只要指令错误，就不会闪红灯。

具体可参阅各模块的详细资料说明

二次开发编程：

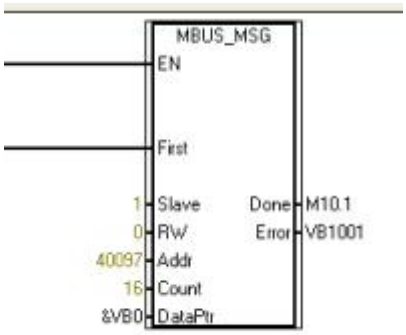
PLC,触摸屏等设备的编程请参考厂家提供的编程资料，最好是向设备原厂提供编程帮助会更快速，阅读 MODBUS-RTU 编程资料，也可以去网上找找同行业的人员的编程经验和编程中遇到的各种问题，搜索工具里输入 XXXMODBUS-RTU 编程，就会出来很多相关的资料，例如 200smart MODBUS-RTU 编程、组态王 MODBUS-RTU 编程等。

与组态软件通信：



在配置向导里选择 COM 口，地址，波特率，起始停止位（模块默认 1，9600，8 N 1)

与 PLC 通信：



如果读取 40~47 号寄存器数据，那么图中地址为 40041 开始，数据为 8 个。

注意：我们模块资料里寄存器表里的寄存器地址全部为十进制，如果需要转换成十六进制发送需要自行转换，

PC 端开发：

PC 端开发先参阅 Modbus-RTU 协议资料，有详细说明哪个字节做什么用，怎么计算 CRC。

例 8 通道的数据起始地址为 40~47，Modbus-RTU 通讯　输入命令示例（例　模块地址：1）

1. 读取 1 个通道（读取通道 0）：　　01 03 00 28 00 01 04 02

2. 读取 4 个通道（读取通道 0 - 3）：01 03 00 28 00 04 C4 01

注：在 RS485 通信过程中会由于线路受到外界的干扰会引起传输字节的错误，由此将会引起本条数据通信失败，如果一次通信失败就认为设备断线或错误，那么将引起很多不必要的问题，解决这个问题的办法是如果通信失败，接下来将再偿试几次（建议两次）通信，如果一直通信不上就认为是设备断线，如果现场干扰比较严重可以使用双绞屏蔽线来屏蔽外界干扰。

如果在通信过程中遇到偶尔会有通信不上的情况，可以将通信的溢出时间适当调长一点，如 0.5 秒，如果时间太短只有几十 MS 的情况下，由于模块要处理一些时间比较长的功能，将造成响应不及时，造成偶尔通信失败

如果购买的模块是非隔离型的，如果 485 总线上的设备不是共用电源，而各电源又没有做共地或接地处理，会引起总线各设备的电势差，此时可能会因大电流烧坏 RS485 总线或造成通信失败，所以在使用 RS485 非隔离的模块时请把各模块的地各部接起来，以保证通信正常,我们的模块不隔离型的 485 地与电源负共用，在多个不隔离设备的总线并联通信时，由于各个设备电源线和 RS485 总线会存在环流的影响，会存在通信的影响，这个时候可使用隔离型的模块来解决这个问题，或使用 RS485 总线隔离器来解决这个问题。

关于 RS485 通信距离的问题，RS485 通用标准距离是 1200 米，但是由于布线、设备数量、现场干扰等问题通距离会不一样，有些用双绞屏蔽线的差不多可达 2000 米，如果用便宜的网线可能就只有几百米了，所以这个问题无法有确切的答案，我们的模块内部全部都没有 120 欧终端负载，可以根据需要加终端负载。

关于两线制热电阻精度问题，热电阻是纯电阻信号，使用中一定需要延长线再接入模块，所有导线都有线阻，关于线长对温度的精度影响问题，以下数据可作为参考，26AWG 的线每 100M 的线阻约 14 欧，PT100 每度约 0.38 欧，(14*2/0.38) 每 100 米线约偏高 73 度。每 10 米线约偏高 7.3 度。PT1000 的影响减少 10 倍，两线建议用 PT1000 传感器，不懂的话请自行网上查资料研究。

使用热电偶时一定要注意周边变频器，变压器等强干扰源时会造成热电偶的温度突变问题，解决这个问题的办法有采用热电阻传感器或者使用带屏蔽的热电偶，并把屏蔽层接入大地，整个机柜和模块也要良好接地，最好购买隔离型的模块，或能更好的解决温度突变问题，因为热电偶是 UV 级信号，很容易受空气电

磁干扰的影响。未屏蔽的热电偶同天线功能类似，会将空气中的信号直接导入到模块造成温度不正常。

关于模块的隔离问题，总有些用户会问那个隔离是什么意思，问到这个的时候真不知道怎么回答，为什么要隔离，请自行网上找资料学习，这个相关的资料非常多，一时半会看不完。一般隔离我们会把 RS485,电源，MCU 部分都隔离开来，这样相互之间电气上不是通的，可以防止很多问题的出现，比如通信问题，通信稳定性，采集稳定性。传感器隔离，增强抗干扰，可以传感器带电测量。还有一些是传感器间隔离，传感器可以测量不同的电位，和增加传感器的抗干扰，使温度更加稳定。

附 1.CRC 校验函数 /*

```
 * 函数介绍：CRC 校验
* 输入参数：
 * 输出参数：
 * 返回值 ：
CRC 校验码 */
u16 CRC16(u8 *bpMsgStartAddress, u8 bDataLen)
{ u8 i, n;
u16 crc = 0xffff;
u8 u8tmp;
u16 u16tmp;
for (i=0; i<bDataLen; i++)
{ crc ^= *bpMsgStartAddress++;
for (n=0; n<8; n++)
      { u8tmp = crc & 1;
      crc >>= 1;
      crc &= 0x7fff;
      if (u8tmp == 1)
      { crc ^= 0xa001; }
}
} return crc;
}
```