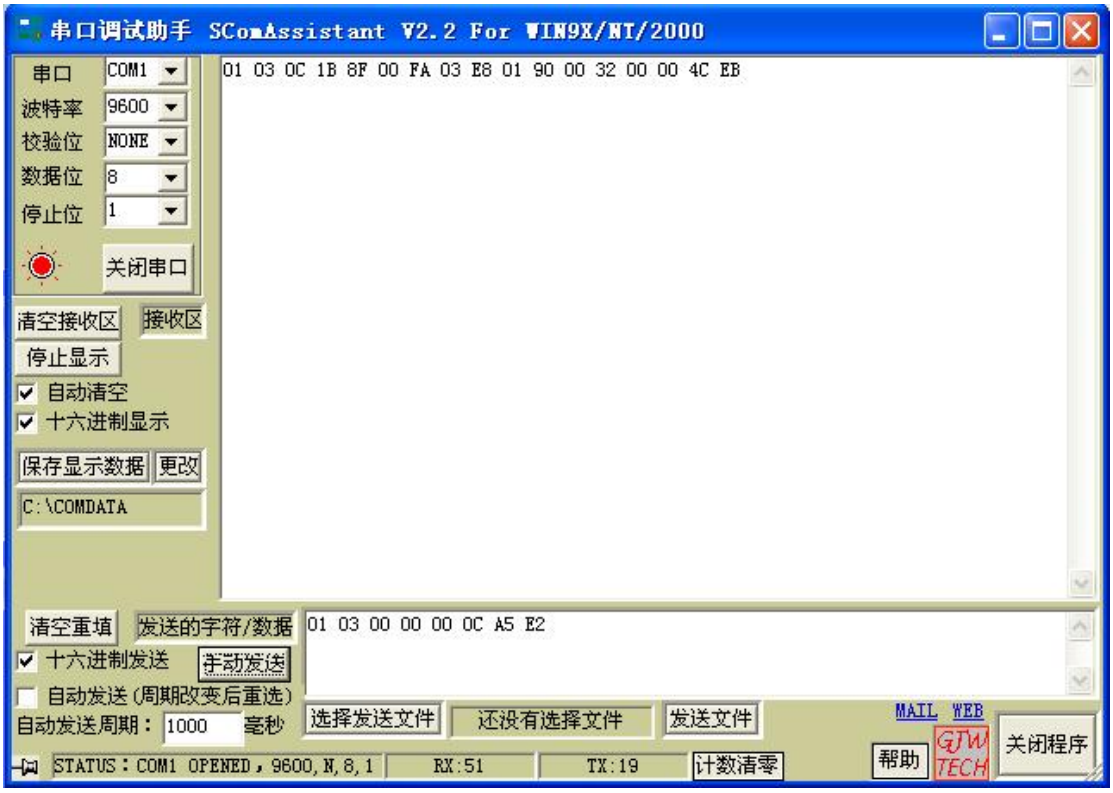
台湾 HANTON 在线 PH/ORP 检测仪

RS-485 接口及 MODBUS 协议

串口调试助手



PH 仪与上位机之间采用 MODBUS 协议进行通信，波特率固定为 9600bps。具体协议如下：

1.1 读取测量信息和状态

命令格式：

定义	地址	功能码	起始地址	数据个数	CRC 校验
数据	ADDR	0x03	0x 0000	0x 000C	CRC 16
字节数	1	1	2	2	2

返回格式：

定义	地址	功能码	数据个数	数据	CRC 校验
数据	ADDR	0x 03	0x 0C	Data	CRC 16
字节数	1	1	1	12	2

注释：

（1） 数据字节长度：为返回数据的实际字节长度。

（2） CRC 16,低 8 位在前，高 8 位在后。

（3） DATA 数据格式如下：

PH 模式的 DATA 数据

1	2	3
PH 值高 8 位字节	PH 值低 8 位字节	温度值高 8 位字节
4	5	6
温度值低 8 位字节	高点报警值高 8 位字节	高点报警值低 8 位字节
7	8	9
低点报警值高 8 位字节	低点报警值低 8 位字节	滞后值高 8 位字节
10	11	12
滞后值低 8 位字节	报警状态	PH/ORP 模式

例，发送命令：01 03 00 00 00 0C A5 E2

返回：01 03 0C 1B 8F 00 FA 03 E8 01 90 00 32 00 00 4C EB

则数据部分为：1B 8F 00 FA 03 E8 01 90 00 32 00 00

PH 值：由整数部分和小数部分组成，整数的有效数字是 2 位，小数有效数字是 3 位。比如 PH 值的高 8 位字节为 0x1B，PH 值的低 8 位字节为 0x8F，则 PH 值为 0x1B8F，转化成 10 进制为 7055，PH 值为 7.055，2 位整数，3 位小数，整数前面的零省去。

温度值：为 0x09C4，转化成 10 进制为 2500，温度值为 25.00，保留两位小数。

高点报警值：为 0x03E8，转化成 10 进制为 1000，值为 10.00，保留两位小数。

低点报警值：为 0x0190，转化成 10 进制为 400，值为 4.00，保留两位小数。

滞后值：为 0x0032，转化成 10 进制为 50，值为 0.50，保留两位小数。

报警状态：为 0x00，无报警。（注：0x00 为无报警，0x01 为低点报警，0x02 为高点报警）

PH/ORP 模式：为 0x00，PH 模式。（注：0x00 为 PH 模式，0x01 为 ORP 模式）

ORP 模式的 DATA 数据

1	2	3
ORP 值高 8 位字节	ORP 值低 8 位字节	温度值高 8 位字节
4	5	6
温度值低 8 位字节	高点报警值高 8 位字节	高点报警值低 8 位字节
7	8	9
低点报警值高 8 位字节	低点报警值低 8 位字节	滞后值高 8 位字节
10	11	12
滞后值低 8 位字节	报警状态	PH/ORP 模式

例，发送命令：01 03 00 00 00 0C A5 E2

返回：01 03 0C FF 30 00 FA 03 E8 FC 18 00 0A 00 01 54 4B

则数据部分为：FF 30 00 FA 03 E8 FC 18 00 0A 00 01

ORP 值：为 0xFF30，当 ORP 值为负数时，数据以补码形式运算，0xFFFF-0xFF30+1=0x00D0，转化成 10 进制为 208，ORP 值为-208mV。ORP 若为 0x011E，转化成 10 进制为 286，ORP 值为 286mV。

温度值：为 0x00FA，转化成 10 进制为 250，温度值为 25.0，保留一位小数。

高点报警值：为 0x03E8，转化成 10 进制为 1000，值为 1000 mV。

低点报警值：为 0x FC18，数据以补码形式运算，值为-1000 mV。

滞后值：为 0x000A，转化成 10 进制为 10，值为 10 mV。

报警状态：为 0x00，无报警。（注：0x00 为无报警，0x01 为低点报警，0x02 为高点报警）

PH/ORP 模式：为 0x01，ORP 模式。（注：0x00 为 PH 模式，0x01 为 ORP 模式）

1.2.错误响应

如果监测器不能正确执行上位机命令，返回如下格式：

定义	地址	功能码	CODE	CRC 校验
数据	ADDR	COM+80H	1	CRC 16
字节数	1	1	1	2

- CODE：　01 – 功能码错
  02 – 起始地址错
  03 – 数据错(数据个数错)
- COM：　接收到的功能码
- CRC 校验，低 8 位在前，高 8 位在后。

例如：

1、发送命令：01 01 00 00 00 0C A5 E2
　　返回：01 81 01 80 7E　　说明功能码错误

2、发送命令：01 03 00 06 00 0C A5 E2
　　返回：01 83 02 81 3E　　起始地址错

3、发送命令：01 03 00 00 00 08 A5 E2
　　返回：01 83 03 41 FF　　数据错(数据个数错)

2.1.写数据寄存器

命令格式：

定义	地址	功能码	起始地址	寄存器数量	字节数
数据	ADDR	0x10	0x 0000	0x 0003	0x 06
字节数	1	1	2	2	1

高报高 8	高报低 8	低报高 8	低报低 8	滞后高 8	滞后低 8	CRC 校验
自定义	自定义	自定义	自定义	自定义	自定义	CRC 16
1	1	1	1	1	1	2

返回格式：

定义	地址	功能码	起始地址	寄存器数量	CRC 校验
数据	ADDR	0x 10	0x 0000	0x 0003	CRC 16
字节数	1	1	2	2	2

例，发送命令：01 10 00 00 00 03 06 03 E8 01 70 00 32 C1 3F

返回： 01 10 00 00 00 03 25 40

附：PH：高点报警(范围 0~14.00)、低点报警(范围 0~14.00)、滞后值(范围 0~9.90)数据格式同上。

ORP：高点报警(范围-1999~1999)、低点报警(范围-1999~1999)、滞后值(范围 0~1000)数据格式同上。

2.2.错误响应

如果监测器不能正确执行上位机命令，返回如下格式：

定义	地址	功能码	CODE	CRC 校验
数据	ADDR	COM+80H	1	CRC 16
字节数	1	1	1	2

- CODE： 01 – 功能码错
  02 – 起始地址错
  03 – 数据错(数据个数错)
  04 – 字节数错或从机无法应答
- COM： 接收到的功能码
- CRC 校验，低 8 位在前，高 8 位在后。

例如：

1、发送命令：01 16 00 00 00 03 06 03 E8 02 70 00 32 C1 3F
返回：01 96 01 80 7E 说明功能码错误

2、发送命令：01 10 00 01 00 03 06 03 E8 01 90 00 32 C1 3F
返回：01 90 02 81 3E 起始地址错

3、发送命令：01 10 00 00 00 05 06 03 E8 01 90 00 32 C1 3F
返回：01 90 03 41 FF 数据错(数据个数错)

3.CRC 检测

CRC 添加到命令中时，低字节先加入，然后高字节。

CRC 简单函数如下：

```
unsigned short CRC16(puchMsg, usDataLen)
```

```
unsigned char *puchMsg ; /* 要进行 CRC 校验的消息 */
unsigned short usDataLen ; /* 消息中字节数 */
{
   unsigned char uchCRCHi = 0xFF ; /* 高 CRC 字节初始化 */
   unsigned char uchCRCLo = 0xFF ; /* 低 CRC 字节初始化 */
   unsigned uIndex ; /* CRC 循环中的索引 */

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

```
/* CRC 高位字节值表 */

static 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 低位字节值表*/

static 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, 0x6 F,
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
} ;