# 振弦采集仪 Modbus RTU 协议说明书

版　　本　　V1.0.7

更新日期　　2021-03-11

适用型号:

AR-SS-CJYVW100X

## 一、通讯说明

默认 9600bps，8b N1 None Parity 通讯采用标准 Modbus RTU RS485 协议；数据采用 16 进制，校验采用 CRC-16 (Modbus)，低位在前，高位在后。

## 二、通讯格式

### 2.1、读取寄存器值 功能码 03

上位机发送 ： 请求两个寄存器的数据一共四个字节表示一个浮点数。具体浮点型简析看附录 2。

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 地址 | 功能码 | 起始地址高位 | 起始地址低位 | 寄存器数量高位 | 寄存器数量低位 | CRC16 低位 | CRC16 高位 |
| Address | 03 | 00 | 00 | 00 | 02 | CRC16L | CRC16H |

第 1 字节 Address　　：　下位机地址码，1~254，（注：247 除外），默认地址为 2

第 2 字节 03　　：　功能码

第 3、4 字节　　：　读取寄存器开始地址

第 5、6 字节　　：　读取寄存器数量

第 7、8 字节　　：　字节 1 到 6 的 CRC16 校验和

下位机回复:

| 1 | 2 | 3 | 4、5 | 6、7 | … | M-1、M | M+1 | M+2 |
|---|---|---|---|---|---|---|---|---|
| 地址 | 功能码 | 字节个数 | 数据位 | 数据位 | 数据位 | 数据位 | CRC16 低位 | CRC16 高位 |
| Address | 04 | 04 | 42 | C8 | 00 | 00 | CRC16L | CRC16H |

第 1 字节 Address　　：　下位机地址码，1~254，（注：247 除外）

第 2 字节 03　　：　功能码

第 3 字节　　：　从 4 到 M(含 4 和 M)的字节总数

第 4 到 M 字节　　：　寄存器数据

第 M+1、M+2 字节　：　从字节 1 到 M 的 CRC16 校验和

## 2.2、写入寄存器值 功能码 10

上位机发送:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8+N*2 | 倒数第 2 字节 | 倒数第 1 字节 |
|---|---|---|---|---|---|---|---|---|---|
| 地址 | 功能码 | 起始地址高位 | 起始地址低位 | 寄存器数量高位 | 寄存器数量低位 | 字节数 | 数据内容 | CRC 高字节 | CRC 低字节 |
| Address | 03 | 00 | 00 | 00 | 01 | 04 | 00 | CRC16H | CRC16L |

第 1 字节 Address　　：　下位机地址码，1~254，（注：247 除外），默认地址为 2

第 2 字节 10　　：　功能码

第 3、4 字节　　：　写入寄存器开始地址

第 5、6 字节　　：　写入寄存器数量

第 7 字节　　：　写入字节数

第 8+N*2 字节　　：　写入内容

最后 2 字节　　：　前面所有字节的 CRC16 校验和

下位机回复:

| 地址 | 功能码 | 起始地址高位 | 起始地址低位 | 寄存器数量高位 | 寄存器数量低位 | CRC16 低位 | CRC16 高位 |
|---|---|---|---|---|---|---|---|
| Address | 10 | 00 | 00 | 00 | 02 | CRC16L | CRC16H |

第 1 字节 Address　　：　下位机地址码，1~254，（注：247 除外）

第 2 字节 10　　：　功能码

第 3、4 字节　　：　寄存器开始地址

第 5、6 字节　　：　寄存器数量

第 7、8 字节　　：　从字节 1 到 M 的 CRC16 校验和

## 2.3、寄存器说明

**VX1001 采集仪**

| 数据名称 | 数据类型 | 只读 | 寄存器地址 | 寄存器数 | 字节顺序 | 说明 |
|---|---|---|---|---|---|---|
| 通道 1 | float | √ | 0x00 | 2 | 2-1-4-3 | |
| 温度 | float | √ | 0x02 | 2 | 2-1-4-3 | |
| 通信地址 | uint16 | | 0x0A | 1 | 2-1 | |
| 采集周期（秒） | uint16 | | 0x0B | 1 | 2-1 | 0 表示不定时采集 |
| 调零 | uint16 | | 0x0C | 1 | 2-1 | 写 1 调零，写 0 取消。 |

**VW1003 传感器**

| 数据名称 | 数据类型 | 只读 | 寄存器地址 | 寄存器数 | 字节顺序 | 说明 |
|---|---|---|---|---|---|---|
| 通道 1 | float | √ | 0x00 | 2 | 2-1-4-3 | |
| 通道 2 | float | √ | 0x02 | 2 | 2-1-4-3 | |
| 通道 3 | float | √ | 0x04 | 2 | 2-1-4-3 | |
| 温度 | float | √ | 0x06 | 2 | 2-1-4-3 | |
| 通信地址 | uint16 | | 0x0A | 1 | 2-1 | |
| 采集周期（秒） | uint16 | | 0x0B | 1 | 2-1 | 0 表示不定时采集 |
| 调零 | uint16 | | 0x0C | 1 | 2-1 | 写 1 调零，写 0 取消。 |

VW1004 传感器

| 数据名称 | 数据类型 | 只读 | 寄存器地址 | 寄存器数 | 字节顺序 | 说明 |
|---|---|---|---|---|---|---|
| 通道 1 | float | √ | 0x00 | 2 | 2-1-4-3 | |
| 通道 2 | float | √ | 0x02 | 2 | 2-1-4-3 | |
| 通道 3 | float | √ | 0x04 | 2 | 2-1-4-3 | |
| 通道 4 | float | √ | 0x06 | 2 | 2-1-4-3 | |
| 温度 | float | √ | 0x08 | 2 | 2-1-4-3 | |
| 通信地址 | uint16 | | 0x0A | 1 | 2-1 | |
| 采集周期（秒） | uint16 | | 0x0B | 1 | 2-1 | 0 表示不定时采集 |
| 调零 | uint16 | | 0x0C | 1 | 2-1 | 写 1 调零，写 0 取消。 |

## 2.4、示例

读取传感器 1 通道的值

上位机发送信息: 02 03 00 00 00 02 C4 38

| 数据 | 字节数 | 说明 | 备注 |
|---|---|---|---|
| 02 | 1 | 传感器地址 | |
| 03 | 1 | 功能码 | 读寄存器 |
| 00 00 | 2 | 寄存器地址 | 0 |
| 00 02 | 2 | 寄存器数量 | 2 个（一个 float 由两个寄存器构成） |
| C4 38 | 2 | CRC | |

下位机应答: 02 03 04 32 F1 44 3F E5 68

| 数据 | 字节数 | 说明 | 备注 |
|---|---|---|---|
| 02 | 1 | 传感器地址 | |
| 03 | 1 | 功能码 | 读寄存器 |
| 04 | 1 | 返回数据的字节数 | |
| 32 F1 44 3F | 4 | 返回的数据 | 浮点数，解析结果为 764.795959，PC 上位机将字节 [1 与 2]，[3 与 4]交换后强制转为浮点数即可，即 PC 端要将字节数组{0xF1, 0x32, 0x3F, 0x44} 强制转换为浮点数 |
| E5 68 | 2 | CRC | |

注意! 当采集周期设置为 0 时，中间可能会收到忙应答信息：02 83 05 71 33

| 数据 | 字节数 | 说明 | 备注 |
|---|---|---|---|
| 02 | 1 | 传感器地址 | |
| 83 | 1 | 功能码 | 读寄存器错误 |
| 05 | 1 | 错误码 | 忙，需等待（此时主站反复重试读命令即可） |
| 71 33 | 2 | CRC | |

# 附录 1

ModBus CRC 校验的 C 语言代码:

```
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
    };
```

```
unsignedcharauchCRCLo[]=
    {
        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
    };

unsigned short CRC16ModBus(unsigned char * puchMsg, unsigned short usDataLen)
{
    unsigned char uchCRCHi = 0xFF;
    unsigned char uchCRCLo = 0xFF;
    unsigned char uIndex;
    unsigned short i = 0;
        while (usDataLen-- > 0)
        {
        uIndex = (uint8)(uchCRCHi ^ puchMsg[i++]);
        uchCRCHi = (uint8)(uchCRCLo ^ auchCRCHi[uIndex]);
        uchCRCLo = auchCRCLo[uIndex];
         }
```

```
    return (unsigned short)(( unsigned short)uchCRCHi << 8 | uchCRCLo);
}
```

## 附录 3

将读数结果转换为浮点数的 C/C++语言代码:

```
//收到的数据字节为 “32 F1 44 3F”
//按协议说明交换顺序为 “F1 32 3F 44”
//然后强制转换为 float
    unsigned char bytes[4] = { 0xF1, 0x32,0x3F,0x44 };
    float value = *(float*) bytes; //得到 value 值为 764.79596
```