

**科士达科技股份有限公司**

文件名称：Modbus\_RTU外部通讯协议

文件编号：

版本号：

修订日期：2012.5.31

目 录

[1.协议总论 2](#_Toc395688703)

[1.1范围 2](#_Toc395688704)

[1.2通信接口及设置 2](#_Toc395688705)

[1.3 监控内容 2](#_Toc395688706)

[2. Modbus-RTU通信协议 3](#_Toc395688707)

[3.监控地址表 11](#_Toc395688708)

1.协议总论

1.1范围

本文制定了我公司为实现精密空调集中监控而使用的通信协议规范，规定了精密空调和监控单元之间的通信协议。本协议使用于精密空调的MODBUS-RTU监控系统。

1.2通信接口及设置

串行通信口采用RS-485。

波特率：9600 ；默认ID号为1 ；通讯方式：串行异步半双工；数据位长度：8位(LSB在前)；奇偶校验位：无；起始位：1位；停止位：1位。

1.3 监控内容

风机，压缩机，电加热，加湿器，温湿度传感器等器件的运行状态及报警。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2. Modbus-RTU通信协议**2.1.Modbus协议简介**Modbus协议是应用于电子控制器上的一种通用语言。通过此协议，控制器相互之间、控制器经由网络（例如以太网）和其它设备之间可以通信。它已经成为一通用工业标准。有了它不同厂商生产的控制设备可以连成工业网络，进行集中监控。 此协议定义了一个控制器能认识使用的消息结构,而不管它们是经过何种网络进行通信的。它描述了一控制器请求访问其它设备的过程，如果回应来自其它设备的请求，以及怎样侦测错误并记录。它制定了消息域格局和内容的公共格式。 当在一Modbus网络上通信时，此协议决定了每个控制器须要知道它们的设备地址，识别按地址发来的消息，决定要产生何种行动。如果需要回应，控制器将生成反馈信息并用Modbus协议发出。在其它网络上，包含了Modbus 协议的消息转换为在此网络上使用的帧或包结构。这种转换也扩展了根据具体的网络解决节地址、路由路径及错误检测的方法。 **2.1.1. 在Modbus 网络上转输** 标准的Modbus口是使用一RS-232C兼容串行接口，它定义了连接口的针脚、电缆、信号位、传输波特率、奇偶校验。控制器能直接或经由Modem组网。 控制器通信使用主—从技术，即仅一设备（主设备）能初始化传输（查询）。其它设备（从设备）根据主设备查询提供的数据作出相应反应。 Modbus协议建立了主设备查询的格式：设备地址、功能代码、所有要发送的数据、一错误检测域。 从设备回应消息也由Modbus协议构成，包括确认要行动的域、任何要返回的数据、和一错误检测域。如果在消息接收过程中发生一错误，或从设备不能执行其命令，从设备将建立一错误消息并把它作为回应发送出去。 **2.1.2. 在其它类型网络上转输** 在其它网络上，控制器使用对等技术通信，故任何控制都能初始和其它控制器的通信。这样在单独的通信过程中，控制器既可作为主设备也可作为从设备。提供的多个内部通道可允许同时发生的传输进程。 在消息位Modbus协议仍提供了主从原则，尽管网络通信方法是“对等”。如果一控制器发送一消息，它只是作为主设备，并期望从从设备得到回应。同样，当控制器接收到一消息，它将建立一从设备回应格式并返回给发送的控制器。 **2.1.3．查询回应** ① 查询 查询消息中的功能代码告之被选中的从设备要执行何种功能。数据段包含了从设备要执行功能的任何附加信息。例如功能代码03是要求从设备读保持寄存器并返回它们的内容。数据段必须包含要告之从设备的信息：从何寄存器开始读及要读的寄存器数量。错误检测域为从设备提供了一种验证消息内容是否正确的方法。 ② 回应 如果从设备产生一正常的回应，在回应消息中的功能代码是在查询消息中的功能代码的回应。数据段包括了从设备收集的数据：象寄存器值或状态。如果有错误发生，功能代码将被修改以用于指出回应消息是错误的，同时数据段包含了描述此错误信息的代码。错误检测域允许主设备确认消息内容是否可用。   **2.2. 传输方式** 控制器设置的传输模式：RTU模式，适用于机器语言编程的计算机和PC主机。可在标准的Modbus 网络通信，报文字符以连续数据流的形式传送。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址  | 功能代码  | 数据数量 | 数据1 | … | 数据n | CRC高字节 | CRC低字节  |

**RTU 模式** 下面就来介绍RTU模式： 当控制器设为在Modbus网络上以RTU（远程终端单元）模式通信，在消息中的每个8Bit字节包含两个4Bit的十六进制字符。这种方式的主要优点是：在同样的波特率下，可比ASCII方式传送更多的数据。 代码系统 8位二进制十六进制数0...9 A...F  消息中的每个8位域都是一个两个十六进制字符组成 每个字节的位  1个起始位  8个数据位最小的有效位先发送  无校验  1个停止位错误检测域 CRC(循环冗长检测)  **2.3. Modbus消息帧** RTU传输模式中，传输设备以将Modbus消息转为有起点和终点的帧，这就允许接收的设备在消息起始处开始工作，读地址分配信息，判断哪一个设备被选中，判知何时信息已完成。部分的消息也能侦测到并且错误能设置为返回结果。 **2.3.1. RTU 帧**使用RTU模式，消息发送至少要以3.5个字符时间的停顿间隔开始。在网络波特率下多样的字符时间，这是最容易实现的(如下图的T1-T2-T3-T4 所示)。传输的第一个域是设备地址。可以使用的传输字符是十六进制的0...9,A...F。 网络设备不断侦测网络总线，包括停顿间隔时间内。当第一个域（地址域）接收到，每个设备都进行解码以判断是否发往自己的。在最后一个传输字符之后，一个至少3.5个字符时间的停顿标定了消息的结束。一个新的消息可在此停顿后开始。 整个消息帧必须作为一连续的流转输。如果在帧完成之前有超过1.5个字符时间的停顿时间，接收设备将刷新不完整的消息并假定下一字节是一个新消息的地址域。同样地，如果一个新消息在小于3.5个字符时间内接着前个消息开始，接收的设备将认为它是前一消息的延续。这将导致一个错误因为在最后的CRC域的值不可能是正确的一典型的消息帧如下所示：  图2 RTU消息帧 **2.3.2. 地址域** 消息帧的地址域包含8Bit；可能的从设备地址是0...247(十进制)。单个设备的地址范围是1...247。主设备通过将要联络的从设备的地址放入消息中的地址域来选通从设备。当从设备发送回应消息时，它把自己的地址放入回应的地址域中以便主设备知道是哪一个设备作出回应。 **2.3.3.如何处理功能域** 消息帧中的功能代码域包含了8Bits，可能的代码范围是十进制的1...255。 当消息从主设备发往从设备时，功能代码域将告之从设备需要执行哪些行为。例如去读取输入的开关状态，读一组寄存器的数据内容，读从设备的诊断状态，允许调入、记录、校验在从设备中的程序等。 当从设备回应时，它使用功能代码域来指示是正常回应(无误)还是有某种错误发生（称作异议回应）。对正常回应，从设备仅回应相应的功能代码。对异议回应，从设备返回一等同于正常代码的代码，但最重要的位置为逻辑1。 例如：一从主设备发往从设备的消息要求读一组保持寄存器，将产生如下功能代码： 0 0 0 0 0 0 1 1（十六进制03H） 对正常回应，从设备仅回应同样的功能代码。对异议回应，它返回： 1 0 0 0 0 0 1 1（十六进制83H） 除功能代码因异议错误作了修改外，从设备将一独特的代码放到回应消息的数据域中，这能告诉主设备发生了什么错误。 主设备应用程序得到异议的回应后，典型的处理过程是重发消息，或者诊断发给从设备的消息并报告给操作员。 **2.3.4.数据域** 数据域是由两个十六进制数集合构成的，范围00...FF。这是由一对RTU字符组成。 从主设备发给从设备消息的数据域包含附加的信息：从设备必须用于进行执行由功能代码所定义的所为。这包括了象不连续的寄存器地址，要处理项的数目，域中实际数据字节数。 例如：如果主设备需要从设备读取一组保持寄存器（功能代码03），数据域指定了起始寄存器以及要读的寄存器数量。如果主设备写一组从设备的寄存器（功能代码10 十六进制），数据域则指明了要写的起始寄存器以及要写的寄存器数量，数据域的数据字节数，要写入寄存器的数据。 如果没有错误发生，从从设备返回的数据域包含请求的数据。如果有错误发生，此域包含一异议代码，主设备应用程序可以用来判断采取下一步行动。 在某种消息中数据域可以是不存在的（0长度）。例如：主设备要求从设备回应通信事件记录（功能代码0B 十六进制），从设备不需任何附加的信息。 **2.3.5. 错误检测域** 当选用RTU 模式作字符帧，错误检测域包含一16Bits值(用两个8 位的字符来实现)。 错误检测域的内容是通过对消息内容进行循环冗长检测方法得出的。CRC域附加在消息的最后，添加时先是低字节然后是高字节。故CRC的高位字节是发送消息的最后一个字节。 **2.3.6. 字符的连续传输** 当消息在标准的Modbus系列网络传输时，每个字符或字节以如下方式发送（从左到右）。 最低有效位...最高有效位 使用RTU字符帧时位的序列是：  有奇偶校验

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  启始位  | 1  | 2  | 3 | 4 | 5 | 6 | 7 | 8 | 奇偶位 | 停止位 |

 无奇偶校验

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  启始位  | 1  | 2  | 3 | 4 | 5 | 6 | 7 | 8 | 停止位 | 停止位 |

**2.4. 错误检测方法** 标准的Modbus串行网络采用两种错误检测方法。奇偶校验对每个字符都可用，帧检测（LRC 或CRC）应用于整个消息。它们都是在消息发送前由主设备产生的，从设备在接收过程中检测每个字符和整个消息帧。 用户要给主设备配置一预先定义的超时时间间隔，这个时间间隔要足够长，以使任何从设备都能作为正常反应。如果从设备测到一传输错误，消息将不会接收，也不会向主设备作出回应。这样超时事件将触发主设备来处理错误。发往不存在的从设备的地址也会产生超时。 **2.4.1. 奇偶校验** 用户可以配置控制器是奇或偶校验，或无校验。这将决定了每个字符中的奇偶校验位是如何设置的。如果指定了奇或偶校验，“1”的位数将算到每个字符的位数中，RTU中8个数据位。 例如：RTU 字符帧中包含以下8个数据位 1 1 0 0 0 1 0 1 整个“1”的数目是4个。如果使用了偶校验，帧的奇偶校验位将是0，使得整个“1”的个数仍是4个。如果使用了奇校验，帧的奇偶校验位将是1，使得整个“1”的个数是5个。 如果没有指定奇偶校验位，传输时就没有校验位，也不进行校验检测。代替一附加的停止位填充至要传输的字符帧中。 **2.4.2. CRC 检测** 使用RTU模式，消息包括了一基于CRC方法的错误检测域。CRC域检测了整个消息的内容。 CRC域是两个字节，包含一16位的二进制值。它由传输设备计算后加入到消息中。接收设备重新计算收到消息的CRC，并与接收到的CRC域中的值比较，如果两值不同，则有误。 CRC是先调入一值是全“1”的16位寄存器，然后调用一过程将消息中连续的8位字节各当前寄存器中的值进行处理。仅每个字符中的8Bit数据对CRC有效，起始位和停止位以及奇偶校验位均无效。 CRC产生过程中，每个8位字符都单独和寄存器内容相或（OR），结果向最低有效位方向移动，最高有效位以0填充。LSB被提取出来检测，如果LSB为1，寄存器单独和预置的值或一下，如果LSB为0，则不进行。整个过程要重复8次。在最后一位（第8位）完成后，下一个8位字节又单独和寄存器的当前值相或。最终寄存器中的值，是消息中所有的字节都执行之后的CRC值。 CRC添加到消息中时，低字节先加入，然后高字节。 /\* 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, 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 **2.5. Modbus的数据校验方式（附注）** CRC-16（循环冗余错误校验） CRC-16错误校验程序如下：报文（此处只涉及数据位，不指起始位、停止位和任选的奇偶校验位）被看作是一个连续的二进制，其最高有效位（MSB）首选发送。报文先与X↑16相乘（左移16位），然后看X↑16+X↑15+X↑2+1除X↑16+X↑15+X↑2+1可以表示为二进制数11000000000000101。整数商位忽略不记，16位余数加入该报文（MSB先发送），成为2个CRC校验字节。余数中的1全部初始化，以免所有的零成为一条报文被接收。经上述处理而含有CRC字节的报文，若无错误，到接收设备后再被同一多项式（X↑16+X↑15+X↑2+1） 除，会得到一个零余数（接收设备核验这个CRC字节，并将其与被传送的CRC比较）。全部运算以2为模（无进位）。 习惯于成串发送数据的设备会首选送出字符的最右位（LSB-最低有效位）。而在生成CRC情况下，发送首位应是被除数的最高有效位MSB。由于在运算中不用进位，为便于操作起见，计算CRC时设MSB 在最右位。生成多项式的位序也必须反过来，以保持一致。多项式的MSB 略去不记，因其只对商有影响而不影响余数。 生成CRC-16校验字节的步骤如下： ① 装如一个16位寄存器，所有数位均为1。 ② 该16位寄存器的高位字节与开始8位字节进行“异或”运算。运算结果放入这个16 位寄存器。 ③ 把这个16寄存器向右移一位。 ④ 若向右（标记位）移出的数位是1，则生成多项式1010000000000001 和这个寄存器进行“异或”运算；若向右移出的数位是0，则返回③。 ⑤ 重复③和④，直至移出8位。 ⑥ 另外8位与该十六位寄存器进行“异或”运算。 ⑦ 重复③~⑥，直至该报文所有字节均与16位寄存器进行“异或”运算，并移位8次。 ⑧ 这个16位寄存器的内容即2字节CRC错误校验，被加到报文的最高有效位。  **2.6. Modbus界面与功能代码** Modbus 功能码(表1)

|  |  |  |
| --- | --- | --- |
|  代码  | 功能（英文）  | 功能（中文）  |
| 01  | READ COIL STATUS  | 线圈状态  |
| 03  | READ HOLDING REGISTER  | 保存寄存器  |
| 05  | WRITE SINGLE COIL  | 写单个线圈  |
| 06  | WRITE SINGLE REGISTER  | 写单个寄存器  |
| 16  | WRITE MULTIPLE REGISTER  | 写多个寄存器  |

Modbus 功能与数据类型对应表(表2)

|  |  |  |
| --- | --- | --- |
|  代码  | 功能  | 数据类型  |
| 01  | 读  | 位  |
| 03  | 读  | 整型、字符型、状态字、浮点型  |
| 05  | 写  | 位  |
| 06  | 写  | 整型、字符型、状态字、浮点型  |
| 16  | 写  | 整型、字符型、状态字、浮点型  |

例子: a. 读主板: (主板地址0x01) 读开关量输入输出与报警状态 （功能代码01）  主机 从地址 0x01 功能代码 0x01 起始地址高字节 0x00 起始地址低字节 0x02 数据数量高字节 0x00 数据数量低字节 0x10 校验和 ---  从机 从地址 0x01 功能代码 0x01 数据数量 0x10 数据（8-1） 0x02 数据（16-9） 0x00 校验和 ---  b. 读设置参数，模拟量输入输出（功能代码03） 在一个远程设备中，使用该功能码读取保持寄存器连续块的内容。请求PDU说明了起始寄存器地址和寄存器数量。从零开始寻址寄存器。因此，寻址寄存器 1-16 为0-15。 将响应报文中的寄存器数据分成每个寄存器有两字节，在每个字节中直接地调整二进制内容。 对于每个寄存器，第一个字节包括高位比特，并且第二个字节包括低位比特。 本协议最多提供长度为8的连续读取。主机 从地址 0x01 功能代码 0x03 起始地址高字节 0x00 起始地址低字节 0x00 数据数量高字节 0x00 数据数量低字节 0x02校验和 ---  从机 从地址 0x01 功能代码 0x03 数据数量 0x04数据（1H） 0x02 数据（1L） 0x00 数据（2H） 0x02 数据（2L） 0x00 校验和 --- c. 设置线圈（功能代码05）在一个远程设备上，使用该功能码写单个输出为 ON 或OFF。 请求数据域中的常量说明请求的ON/OFF状态。十六进制值FF 00请求输出为 ON。十六进制值00 00 请求输出为OFF。其它所有值均是非法的，并且对输出不起作用。 请求PDU说明了强制的线圈地址。从零开始寻址线圈。因此，寻址线圈1为0。线圈值域的常量说明请求的ON/OFF状态。十六进制值0XFF00请求线圈为ON。十六进制值0X0000请求线圈为OFF。其它所有值均为非法的，并且对线圈不起作用。 正常响应是请求的应答，在写入线圈状态之后返回这个正常响应。主机 从地址 0x01 功能代码 0x05 起始地址高字节 0x00 起始地址低字节 0x3F 数据数量高字节 0XFF 数据数量低字节 0x00校验和 ---  从机 从地址 0x01 功能代码 0x05 起始地址高字节 0x00 起始地址低字节 0x3F 数据数量高字节 0XFF 数据数量低字节 0x00校验和 --- D. 设置参数（功能代码16）在一个远程设备中，使用该功能码写单个保持寄存器。 请求PDU 说明了被写入寄存器的地址。从零开始寻址寄存器。因此，寻址寄存器1 为 0。 正常响应是请求的应答，在写入寄存器内容之后返回这个正常响应。主机 从地址 0x01 功能代码 0x10 起始地址高字节 0x00 起始地址低字节 0x00 数据数量高字节 0x00 数据数量低字节 0x01字节数 0x02数据数量高字节 0x00 数据数量低字节 0x01校验和 ---  从机 从地址 0x01 功能代码 0x10 起始地址高字节 0x00 起始地址低字节 0x00 数据数量高字节 0x00 数据数量低字节 0x01校验和 ---  |

3.监控地址表

以下参数地址只适用于科士达小机StationAir恒温恒湿系列

|  |
| --- |
| 故障地址定义 |
| 序号 | 地址 | 名称 | 功能号 |
| 1 | 1 | 新风温度探头故障; | 0x01 |
| 2 | 2 | 吸气温度探头故障; | 0x01 |
| 3 | 3 | 吸气压力探头故障; | 0x01 |
| 4 | 4 | 新风湿度探头故障; | 0x01 |
| 5 | 5 | 回风温度探头故障; | 0x01 |
| 6 | 6 | 回风湿度探头故障; | 0x01 |
| 7 | 8 | 水浸报警; | 0x01 |
| 8 | 9 | 低湿报警; | 0x01 |
| 9 | 10 | 高湿报警; | 0x01 |
| 10 | 11 | 低温报警; | 0x01 |
| 11 | 12 | 高温报警; | 0x01 |
| 12 | 13 | 压缩机一高压; | 0x01 |
| 13 | 14 | 压缩机一低压; | 0x01 |
| 14 | 19 | 外风机一过载; | 0x01 |
| 15 | 21 | 压缩机二高压; | 0x01 |
| 16 | 22 | 压缩机二低压; | 0x01 |
| 17 | 27 | 外风机二过载; | 0x01 |
| 18 | 45 | 制热保护; | 0x01 |
| 19 | 48 | 冷却水泵过载; | 0x01 |
| 20 | 49 | 冷却水流开关; | 0x01 |
| 21 | 58 | 送风温度过高过低; | 0x01 |
| 22 | 63 | 电源故障; | 0x01 |
| 23 | 64 | 失风报警; | 0x01 |
| 24 | 65 | 滤网堵塞; | 0x01 |
| 25 | 66 | 高电流报警; | 0x01 |
| 26 | 67 | 低电流报警; | 0x01 |
| 27 | 70 | 水质软报警; | 0x01 |
| 28 | 71 | 手动排水报警; | 0x01 |
| 29 | 60 | 电源故障/风机过载 | 　 |
| 30 | 7 | 火灾报警（电流一故障） | 　 |
| 继电器输出地址定义 |
| 序号 | 地址 | 名称(风冷) | 功能号 |
| 1 | 128 | 送风机; | 0x01 |
| 2 | 129 | 压缩机一; | 0x01 |
| 3 | 130 | 压缩机二; | 0x01 |
| 4 | 131 | 电加热一; | 0x01 |
| 5 | 132 | 电加热二; | 0x01 |
| 6 | 133 | 除湿阀; | 0x01 |
| 7 | 134 | 加湿电极; | 0x01 |
| 8 | 135 | 冷凝风机一; | 0x01 |
| 9 | 136 | 冷凝风机二; | 0x01 |
| 10 | 137 | 进水阀; | 0x01 |
| 11 | 138 | 排水阀; | 0x01 |
| 12 | 139 | 故障/轮值输出; | 0x01 |
| 故障输入地址定义 |
| 序号 | 地址 | 名称(风冷) | 功能号 |
| 25 | 152 | 失风报警 | 0x01 |
| 26 | 153 | 滤网堵塞 | 0x01 |
| 27 | 154 | 压缩机一高压 | 0x01 |
| 28 | 155 | 压缩机一低压 | 0x01 |
| 29 | 156 | 外风机一过载 | 0x01 |
| 30 | 157 | 压缩机二高压 | 0x01 |
| 31 | 158 | 压缩机二低压 | 0x01 |
| 32 | 159 | 外风机二过载 | 0x01 |
| 33 | 160 | 远控/轮值输入 | 0x01 |
| 34 | 161 | 电源故障/风机过载 | 0x01 |
| 35 | 162 | 制热保护 | 0x01 |
| 36 | 163 | 高水位 | 0x01 |
| 37 | 164 | 火灾报警 | 0x01 |
| 38 | 165 | 水浸报警 | 0x01 |
| 厂家设置参数寄存器地址定义 |
| 序号 | 地址 | 名称 | 范围 | 功能号 |
| 0 | 645 | 温度校正(℃); | -9.9~9.9 | 0x10 |
| 1 | 646 | 湿度校正(%); | -9.9~9.9 | 0x10 |
| 2 | 663 | 低压报警延时(s); | 1~240 | 0x10 |
| 3 | 640 | 启动延时(s); | 1~240 | 0x10 |
| 4 | 666 | 最短运行时间(s); | 1~240 | 0x10 |
| 5 | 662 | 最短停机时间(s); | 1~240 | 0x10 |
| 6 | 544 | 故障报警时间(s); | 1~10 | 0x10 |
| 7 | 602 | 一级制冷上限(℃); | 0~5.0 | 0x10 |
| 8 | 603 | 一级制冷下限(℃); | -2.0~5.0 | 0x10 |
| 9 | 604 | 二级制冷上限(℃); | 0~5.0 | 0x10 |
| 10 | 605 | 二级制冷下限(℃); | -2.0~5.0 | 0x10 |
| 11 | 606 | 一级制热上限(℃); | 0~5.0 | 0x10 |
| 12 | 607 | 一级制热下限(℃); | -2.0~5.0 | 0x10 |
| 13 | 608 | 二级制热上限(℃); | 0~5.0 | 0x10 |
| 14 | 609 | 二级制热下限(℃); | -2.0~5.0 | 0x10 |
| 15 | 610 | 加湿上限(%); | 0~10.0 | 0x10 |
| 16 | 611 | 加湿下限(%); | -2.0~10.0 | 0x10 |
| 17 | 612 | 除湿上限(%); | 0~10.0 | 0x10 |
| 18 | 613 | 除湿下限(%); | -2.0~10.0 | 0x10 |
| 19 | 601 | 上下限设置; | 0~2 | 0x10 |
| 20 | 644 | 送风温度过高(℃); | 20.0~70.0 | 0x10 |
| 21 | 659 | 送风温度过低(℃); | 0~30.0 | 0x10 |
| 22 | 523 | 切换系统温度; | 20.0~40.0 | 0x10 |
| 23 | 673 | 制冷开冷凝(℃); | 0~99.0 | 0x10 |
| 24 | 522 | 电加热个数; | 0~2 | 　 |
| 25 | 521 | 压缩机个数; | 1~2 | 0x10 |
| 26 | 528 | 制热模式; | 0~2 | 0x10 |
| 27 | 515 | 联网地址; | 1~32 | 0x10 |
| 28 | 524 | 除湿开度(%) | 10~100 | 0x10 |
| 29 | 514 | 正常时开度(%) | 40~100 | 0x10 |
| 30 | 532 | 第二模拟输出; | 1~3 | 0x10 |
| 31 | 529 | 系统工作方式; | 0~2 | 0x10 |
| 32 | 681 | 轮值功能设置; | 0~1 | 0x10 |
| 33 | 688 | 轮值机组个数; | 1~5 | 0x10 |
| 34 | 682 | 机组轮值地址; | 0~5 | 0x10 |
| 35 | 683 | 轮值时间设置; | 0~99 | 0x10 |
| 36 | 684 | 温差报警(℃); | 5.0~20.0 | 0x10 |
| 37 | 686 | 湿差报警(%); | 5.0~40.0 | 0x10 |
| 38 | 639 | 恢复厂家设置; | 0~1 | 0x10 |
| 39 | 534 | 电极选择; | 0~1 | 0x10 |
| 40 | 530 | 加湿器选择; | 0~4 | 0x10 |
| 41 | 535 | 加湿量选择(Kg); | 2~46 | 0x10 |
| 42 | 538 | 二次进水时间(分); | 1~10 | 0x10 |
| 43 | 536 | 排水比例(%); | 5~30 | 0x10 |
| 44 | 537 | 排水时间(s); | 1~99 | 0x10 |
| 45 | 539 | 排水间隔(10分); | 1~99 | 0x10 |
| 46 | 552 | 送风探头设置; | 0~1 | 0x10 |
| 47 | 553 | 新风温度启用/禁用; | 0~1 | 0x10 |
| 48 | 554 | 新风湿度启用/禁用; | 0~1 | 0x10 |
| 49 | 564 | 失风报警常开常闭; | 0~1 | 0x10 |
| 50 | 565 | 滤网堵塞常开常闭; | 0~1 | 0x10 |
| 51 | 566 | 压一高压常开常闭; | 0~1 | 0x10 |
| 52 | 567 | 压一低压常开常闭; | 0~1 | 0x10 |
| 53 | 568 | 风机一过载常开常闭; | 0~1 | 0x10 |
| 54 | 569 | 压二高压常开常闭; | 0~1 | 0x10 |
| 55 | 570 | 压二低压常开常闭; | 0~1 | 0x10 |
| 56 | 571 | 风机二过载常开常闭; | 0~1 | 0x03 |
| 57 | 574 | 制热保护常开常闭; | 0~1 | 0x03 |
| 58 | 540 | 除湿阀常开常闭; | 0~1 | 0x03 |
| 59 | 638 | 高水位电压 | 0-2 | 0x03 |
| 60 | 531 | 第一模拟输出; | 1~3 | 0x03 |
| 61 | 573 | 电子膨胀阀 | 1 | 0x03 |
| 62 | 574 | 电子膨胀阀步数 | 2000 | 0x03 |
| 63 | 575 | 手动调开度 | 0 | 0x03 |
| 64 | 576 | 最低开度 | 200 | 0x03 |
| 65 | 577 | 过热度目标值 | 80 | 0x03 |
| 66 | 578 | *最小过热度* | -50 | 0x03 |
| 67 | 579 | *最小过热度调整步数* | 100 | 0x03 |
| 68 | 580 | P | 20 | 0x03 |
| 69 | 581 | I | 5 | 0x03 |
| 70 | 582 | T | 10 | 0x03 |
| 71 | 583 | 除湿使用 | 1 | 0x03 |
| 72 | 584 | 新风差 | 1~20（5） | 0x03 |
| 73 | 585 | 新风湿度值 | 50~90（80） | 0x03 |
| 74 | 586 | 系统切换时间 | 1~180（30） | 0x03 |
| 75 | 587 | 系统切换温度 | 25~50（30） | 0x03 |
| 76 | 588 | D | 2 | 0x03 |
| 77 | 618 | 新风温度校正 | -9.9~9.9 | 0x03 |
| 78 | 619 | 吸气温度校正 | -9.9~9.9 | 0x03 |
| 79 | 620 | 吸气压力校正 | -9.9~9.9 | 0x03 |
| 80 | 621 | 新风湿度校正 | -9.9~9.9 | 0x03 |
| 用户参数寄存器地址定义 |
| 序号 | 地址 | 名称 | 范围 | 功能号 |
| 1 | 653 | 温度设定; | 18.0~32.0 | 0x10 |
| 2 | 652 | 湿度设定; | 30.0~80.0 | 0x10 |
| 3 | 642 | 来电自启; | 0~1 | 0x10 |
| 4 | 176 | 开关机 | 0~1 | 0x01/0x05 |
| 5 | 177 | 故障复位 | 0~1 | 0x01/0x05 |
| 6 | 178 | 自动手动 | 0~1 | 0x01/0x05 |
| 模拟量显示定义 |
| 序号 | 地址 | 名称 | 功能号 |
| 1 | 700 | 送风探头 | 0x03 |
| 2 | 701 | 翅片一/冷却出水探头 | 0x03 |
| 3 | 702 | 翅片二/冷却回水探头 | 0x03 |
| 5 | 704 | 回风温度探头 | 0x03 |
| 6 | 705 | 回风湿度探头 | 0x03 |
| 7 | 699 | 加湿器电流 | 0x03 |
| 8 | 708 | 模拟输出一 | 0x03 |
| 9 | 709 | 模拟输出二 | 0x03 |