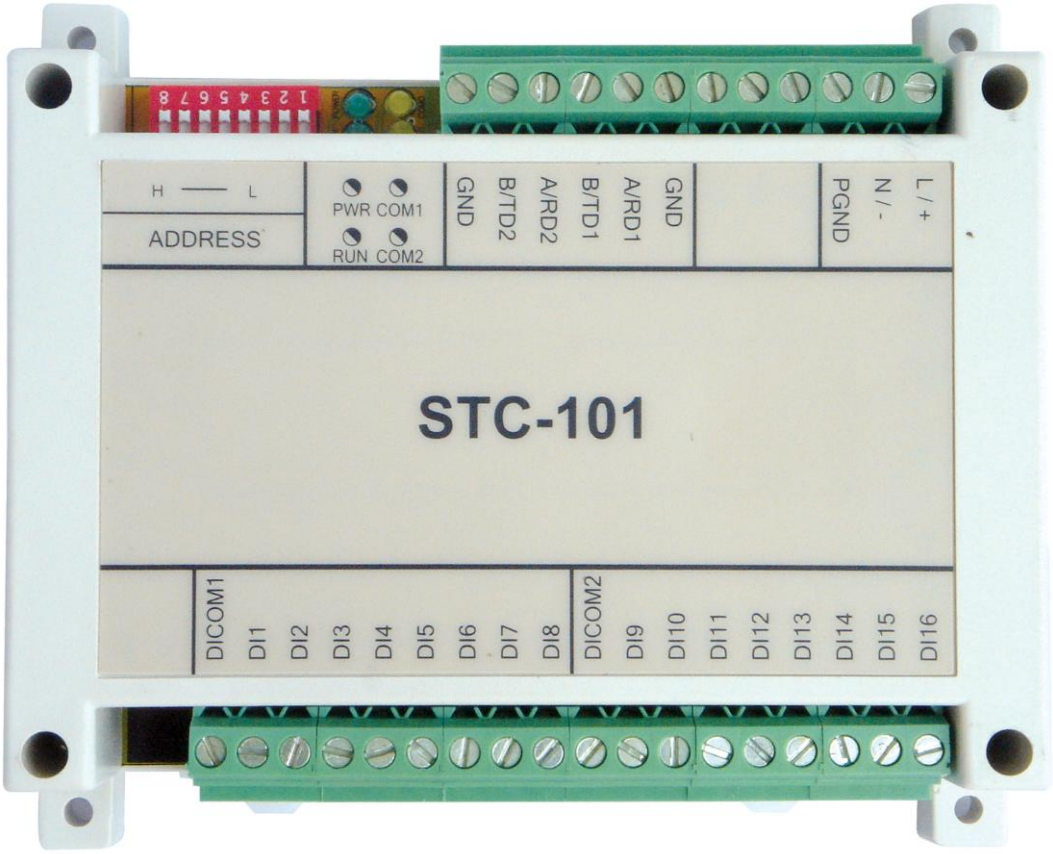
STC-101 微型 RTU

用户手册(V1.3)



北京易控微网科技有限公司

V1.2 修改说明：修改了部分错误和增加了 STC-101 测量频率功能

V1.3 修改说明：支持 MODBUS 规约的 RTU 和 ASCII 两种方式，可修通讯参数。

1. 产品介绍

1.1. 产品设计说明

STC-101 微型 RTU（以下简称 STC-101）是我公司针对各种应用场合，研发的 STC 系列 RTU 之一，广泛应用于消防、供水、石化、环保、电力等各个行业，为大多数系统集成商和自动化公司、研究所采用，是一种具有极高性价比、稳定可靠的数据采集产品。

STC－101 模块可以单独使用，也可以进行扩展，建议在一个 485 网络内，模块数量小于 32。

1.2. 功能特点

- 16 路开关量光电隔离输入， 支持脉冲计数及 SOE（事件顺序记录）。
- 1 个标准 485 或 232 通信口，支持 MODBUS 规约的 ASCII 和 RTU 两种方式。可通过计算机设定 RTU 还是 ASCII 方式，波特率（最高 115200），奇偶校验。
- 高可靠性高，较强抗干扰能力。
- 卡式导轨或螺丝固定，安装简单。

1.3. 系统参数

1.3.1. 开关量输入

容量： 每个模块 16 路

额定输入信号（订货时选择）：

a. 输入直流 110－400V 或交流 110V/220V/380V

b. 输入直流 12V 或 24V 或 48V

输入方式： 光电隔离

扫描方式： 中断方式

SOE 分辨率： 1ms

1.3.2. 脉冲计数

作为脉冲计数使用，要求输入必须为直流。

扫描方式：中断方式

最高计数频率： 1KHz 16 路

单路最高计数频率：4KHZ

1.3.3. 通信接口

容量： 1 路
接口方式： RS485/RS232 接口
规约： MODBUS 规约或者其他规约

1.4. 安装使用环境

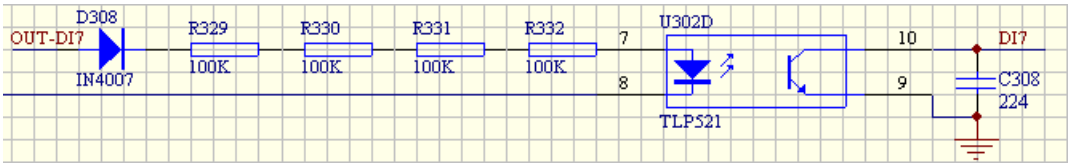
安装方式： 卡式导轨安装或者底部螺丝固定
温度范围： -10℃～ 55℃
存贮温度： -20℃～70℃
相对湿度： ＜85%（20±5℃条件）
大气压力： 86～108Kpa
安装尺寸： 143×90×40mm
电 源： 交流 165～265V 50HZ~60HZ
或者 直流 24V ± 30 %
功 耗： 小于 1W
工作环境： 无爆炸，无腐蚀性气体及导电尘埃，无严重霉菌存在，无剧烈振动，无冲击源；如果需要在此类环境下工作，请采取相应的防护措施。

2. 原理说明

STC-101 单元采用 16 位超低功耗单片机，单片机内部集成了几乎计算机所能外围电路，设计该产品时没有进行任何的扩展。提高了系统的可靠性和抗干扰能力，缩小了产品的体积。

2.1. 开关量输入

系统的数字输入采用光电隔离，原理图如下：



输入信号为高时，光耦导通，DI7=0。

计算机监测到信号的变化，产生中断。单片机记下产生中断的时间，并启动一个软定时器计数。如果没有到达设定的时间信号又发生变换，则清除中断时间和事件记录。如果在设定的时间内信号没有发生变化，则作为输入改变，并记录改变的时间作为 SOE 记录。

开关量的输入具有公共端子，而且是无源的，使用时需要外接电源。每路输入的输入电流不超过 1mA，功率消耗很小。

一般情况下，输入的公共端子应该接开关量输入电源的负端，输入电源的正端接待测开关量的一端，另外一端接单元的输入端。

开关量输入可以采用交流电源或者直流电源作为开关量输入电源。

3. 通信协议说明

我们的 STC-1 采用 MODBUS 规约，原因是该规约文本容易得到，协议本身简单,而且该规约是开放的，有着众多国内厂商和国际厂商的支持。

MODBUS 规约是 MODICOM 公司开发的，版权归其所有。

我们的接口采用 RS485 接口，比 RS232 具有更高的通信速率和更远的通信距离。

根据我们设备的情况，我们仅仅实现了 MODBUS 的一个小型子集，没有完全实现其所有内容，已经能够满足我们所有的需要。

3.1. MODBUS 规约简介

MODBUS 规约是 MODICOM 公司开发的一个为很多厂商支持的开放规约。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，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 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。

3.1.1. 在 Modbus 网络上转输

标准的 Modbus 口是使用 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做出相应反应。典型的主设备：主机和可编程仪表。典型的从设备：可编程控制器。

主设备可单独和从设备通信，也能以广播方式和所有从设备通信。如果单独通信，从设备返回消息作为回应，如果是以广播方式查询的，则不作任何回应。Modbus 协议建立了主设备查询的格式：设备（或广播）地址、功能代码、所有要发送的数据、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、和错误检测域。如果在消息接收过程中发生错误，或从设备不能执行其命令，从设备将建立错误消息并把它作为回应发送出去。

3.1.2. 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”。如果控制器发送消息，它只是作为主设备，并期望从从设备得到回应。同样，当控制器接收到消息，它将建立一从设备回应格式并返回给发送的控制器。

3.1.3. 查询—回应周期

.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。

.回应

如果从设备产生正常的回应，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：像寄存器值或状态。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。

3.2. 帧格式说明

控制器能设置为两种传输模式（ASCII 或 RTU）中的任何一种在标准的 Modbus 网络通信。用户选择想要的模式，包括串口通信参数（波特率、校验方式等），在配置每个控制器的时候，在 Modbus 网络上的所有设备都必须选择相同的传输模式和串口参数。

ASCII 模式

:	地址	功能代码	数据数量	数据 1	...	数据 n	LRC 高字节	LRC 低字节	回车	换行

RTU 模式

地址	功能代码	数据数量	数据 1	...	数据 n	CRC 高字节	CRC 低字节

所选的 ASCII 或 RTU 方式仅适用于标准的 Modbus 网络，它定义了在这些网络上连续传输的消息段的每一位，以及决定怎样将信息打包成消息域和如何解码。

当控制器设为在 Modbus 网络上以 ASCII（美国标准信息交换代码）模式通信，在消息中的每个 8Bit 字节都作为两个 ASCII 字符发送。这种方式的主要优点是字符发送的时间间隔可达到 1 秒而不产生错误。

代码系统

. 十六进制，ASCII 字符 0...9，A...F

. 消息中的每个 ASCII 字符都是一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

. 1 个奇偶校验位，无校验则无

. 1 个停止位（有校验时），1 个 Bit（无校验时）

错误检测域

. LRC(纵向冗长检测)

地址域

消息帧的地址域包含两个字符（ASCII）或 8Bit（RTU）。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做出回应。

地址 0 是用作广播地址，以使所有的从设备都能认识。当 Modbus 协议用于更高水准的网络，广播可能不允许或以其它方式代替。

功能域

消息帧中的功能代码域包含了两个字符（ASCII）或 8Bits（RTU)。可能的代码范围是十进制的 1...255。当然，有些代码是适用于所有控制器，有此是应用于某种控制器，还有些保留以备后用。

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。

我们目前所支持的功能码非常有限，主要包括：

01 READ COIL STATUS

02 READ INPUT STATUS

03 READ HOLDING REGISTERS

04 READ INPUT REGISTERS

05 FORCE SINGLE COIL

06 PRESET SINGLE REGISTERS

24 READ FIFO QUEUE

3.3. 数据定义

STC-101 通信数据定义：

我们采用 MODBUS 规约的 ASCII 方式或 RTU 方式，通讯出厂默认设定为：RTU，波特率 9600BPS，1 位起始位，8 位数据位，1 位停止位，无校验。

帧格式：

上位机发送例

读地址为模块 1 的输入状态，从第一个开始读，读 8 个开关量输入点的值

ASCII 方式

：单元地址 功能码 起始地址 读取点数 LRC 校验 CR LF

：　01　　02　　0000　　0008　　F5　　CR LF

计算机串口发出以上 ASCII 字符，冒号为帧起始标志，CR　LF 为帧结束标志

RTU 方式

单元地址 功能码 起始地址 读取点数 CRC 校验

01　　02　　0000　　0008　　79CC

计算机串口发出以上十六进制字符

LRC 校验为和校验，占用两个字节。计算方法可以参考如下的 C 源程序，需要注意的是首先计算 LRC 的值,然后把结果转换成为相应的 ASCII 字符串。比如 LRC 结果为 0X5F，则在规约中 LRC 的值为 0X35, 0X46

```
unsigned char lrc(unsigned char *str,int lenth)
{
     unsigned char tmp;
     tmp=0;
     while (lenth-- )
     {
         tmp+= *str++;
    }

      return ((unsigned char)(-((char)tmp)));
}
```

RTU 方式　CRC 校验计算：

```
const 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
} ;

const 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
} ;
```

```
unsigned short crc(unsigned char *puchMsg , unsigned short usDataLen)
{
      unsigned char uchCRCHi = 0xFF ;  /* high byte of CRC initialized */
      unsigned char uchCRCLo = 0xFF ;  /* low byte of CRC initialized */
      unsigned uIndex ;  /* will index into CRC lookup table */
      while (usDataLen--) /* pass through message buffer */
      {
          uIndex = uchCRCHi ^ *puchMsg++ ;  /* calculate the CRC */
          uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex];
          uchCRCLo = auchCRCLo[uIndex] ;
      }
      return (uchCRCHi << 8 | uchCRCLo) ;
}
```

在通信中,假设是各种不同类型的量的第一个起始地址为 0000,按照顺序排列的,不同的功能码实际读取的是不同类型的量。上位机按照程序可以顺序读取所有的量。

下位机响应例

地址为的模块回应计算机自己的 8 个开关量输入的输入状态

ASCII 方式

单元地址	功能码	字节数	数据	LRC 校验	CR LF
:	01	02	01	00	FC	CR LF

模块发出的为 ASCII 字符，冒号为帧起始标志，CR LF 为帧结束标志

字节数为 1，则其后跟 1 个数据，为 2，则其后跟 2 个数据

RTU 方式

单元地址	功能码	字节数	数据	CRC 校验
01	02	01	00	A188

模块发出的为十六进制字符

功能码： 02

数据起始地址：00

数据长度：不大于 16（因为只有 16 路输入）

功能： 读取输入开关量的状态

说明： 当读取数据长度小于等于 8 时，模块返回数据的第 7 位对应输入开关量的第

8 路，第 4 位对应第 5 路，……，第 0 位对应第 1 路。当读取数据长度大于 8 并小于等于 16 时，则返回两个数据，每个数据的每个位对应 1 个输入点的状态，第 1 个数据的第一位至第八位对应第一个至第八个输入开关量状态，第 2 个数据的第一位对应第 9 个至第十六个输入开关量。开关量有信号输入时，经过 0.320S 滤波抗干扰后位的值为 1，无信号输入时位的值为 0。

ASCII 方式计算机发送

: 单元地址 02 起始地址 读取点数 LRC 校验 CR LF

ASCII 方式模块响应

: 单元地址 02 字节数 数据 LRC 校验 CR LF

RTU 方式计算机发送

单元地址 02 起始地址 读取点数 CRC 校验

RTU 方式模块响应

单元地址 02 字节数 数据 CRC 校验

功能码： 03

数据起始地址：00－63

数据长度：不大于 32 （因为通信缓冲区的限制）

开关量采用交流信号输入时，没有 32 位计数。

功能： 读取保持寄存器的值。

说明： 读取的是 16 位整数或者无符合整数。

ASCII 方式计算机发送

: 单元地址 03 起始地址 读取点数 LRC 校验 CR LF

ASCII 方式模块响应

: 单元地址 03 字节数 数据 LRC 校验 CR LF

RTU 方式计算机发送

单元地址 03 起始地址 读取点数 CRC 校验

RTU 方式模块响应

单元地址 03 字节数 数据 CRC 校验

保持寄存器定义：

地址（十六进制）	数据描述
0000	系统实际时间的低 16 位 （0000－0032 为无符号整数）
0001	系统实际时间的高 16 位
0002	第一路开关量 32 位计数器低 16 位
0003	第一路开关量 32 位计数器高 16 位
0004	第二路开关量 32 位计数器低 16 位
0005	第二路开关量 32 位计数器高 16 位
0006	第三路开关量 32 位计数器低 16 位
0007	第三路开关量 32 位计数器高 16 位
0008	第四路开关量 32 位计数器低 16 位
0009	第四路开关量 32 位计数器高 16 位
000A	第五路开关量 32 位计数器低 16 位
000B	第五路开关量 32 位计数器高 16 位
000C	第六路开关量 32 位计数器低 16 位
000D	第六路开关量 32 位计数器高 16 位
000E	第七路开关量 32 位计数器低 16 位
000F	第七路开关量 32 位计数器高 16 位
0010	第八路开关量 32 位计数器低 16 位
0011	第八路开关量 32 位计数器高 16 位
0012	第九路开关量 32 位计数器低 16 位
0013	第九路开关量 32 位计数器高 16 位
0014	第十路开关量 32 位计数器低 16 位
0015	第十路开关量 32 位计数器高 16 位
0016	第十一路开关量 32 位计数器低 16 位
0017	第十一路开关量 32 位计数器高 16 位
0018	第十二路开关量 32 位计数器低 16 位
0019	第十二路开关量 32 位计数器高 16 位
001A	第十三路开关量 32 位计数器低 16 位
001B	第十三路开关量 32 位计数器高 16 位
001C	第十四路开关量 32 位计数器低 16 位
001D	第十四路开关量 32 位计数器高 16 位
001E	第十五路开关量 32 位计数器低 16 位

001F　　第十五路开关量 32 位计数器高 16 位
001A　　第十六路开关量 32 位计数器低 16 位
001B　　第十六路开关量 32 位计数器高 16 位

0030　　串口 1 通讯设定，001C 内容用 16 进制表示为 0XPQRS　R 保留
P 高 4 位为奇偶校验设置。P=D(13)为 1,8,E,　P=1 为 1,8,N
Q ＝1，通讯为 ASCII 方式；Q ＝0，通讯为 RTU 方式
S = 0　　波特率 9600
S = 1　　波特率 300
S = 2　　波特率 600
S = 3　　波特率 1200
S = 4　　波特率 2400
S = 5　　波特率 4800
S = 6　　波特率 9600
S = 7　　波特率 19200
S = 8　　波特率 38400
S = 9　　波特率 57600
S = A　　波特率 76800
S = B　　波特率 115200
S = C ~F　波特率 9600
默认通讯方式设定：
通电前把地址开关拨成地址 0，然后重新开电。这时通讯为默认方式：模块地址为 1，通讯规约为 RTU，波特率为 9600　1，8，N

0031　　测频闸门时间

功能码： 04

数据起始地址：00－15
数据长度：不大于 16　（因为通信缓冲区的限制）

功能：　读取输入寄存器的值。
说明：　读取的是 16 位整数或者无符合整数。

ASCII 方式计算机发送

: 单元地址 04 起始地址 读取点数 LRC 校验 CR LF

ASCII 方式模块响应

: 单元地址 04 字节数 数据 LRC 校验 CR LF

RTU 方式计算机发送

单元地址 04 起始地址 读取点数 CRC 校验

RTU 方式模块响应

单元地址 04 字节数 数据 CRC 校验

输入寄存器定义：

地址（十六进制）	数据描述
0000	第 1 路频率
0001	第 2 路频率
0002	第 3 路频率
0003	第 4 路频率
0004	第 5 路频率
0005	第 6 路频率
0006	第 7 路频率
0007	第 8 路频率
0008	第 9 路频率
0009	第 10 路频率
000A	第 11 路频率
000B	第 12 路频率
000C	第 13 路频率
000D	第 14 路频率
000E	第 15 路频率
000F	第 16 路频率

功能码： 06

数据起始地址：0000－0080

数据长度：

功能： 设置保持寄存器的值。

说明： 设置的是 16 位整数。

各个寄存器的说明参考上面保持寄存器定义

ASCII 方式计算机发送

: 单元地址 06 起始地址 数据 LRC 校验 CR LF

ASCII 方式模块响应

: 单元地址 06 起始地址 数据 LRC 校验 CR LF

数据为 16 位整数

RTU 方式计算机发送

单元地址 06 起始地址 数据 CRC 校验

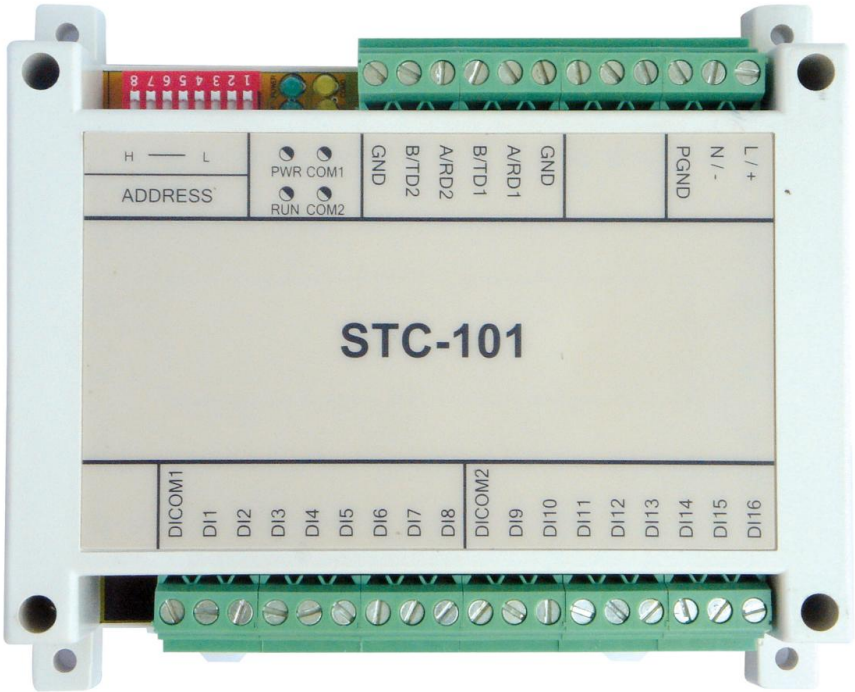
RTU 方式模块响应

单元地址 06 起始地址 数据 CRC 校验

详细说明参考 MODBUS 规约。

4. 安装说明

STC-101 安装的方法：如果是有 IEC 标准导轨，直接卡装在导轨上即可。如果没有导轨，需要用四个螺丝把 STC-101 安装在机柜上。



参照下面说明接线端子定义和接线

如上图所示，端子定义按序对应关系如下：

L/+　　交流电源输入/直流电源正

N/-　　交流电源输入/直流电源负

PGND 保护地

GND	第一路 232 接口的 GND
A/RD1	第一路 232 接口的 RXD 或 485 接口的 A(485 正)
B/TD1	第一路 232 接口的 TXD 或 485 接口的 B(485 负)
A/RD2	第二路 232 接口的 RXD 或 485 接口的 A(485 正)
B/TD2	第二路 232 接口的 TXD 或 485 接口的 B(485 负)
GND	第二路 232 接口的 GND
DICOM1	第一组开关量输入端公共端
DI1	第一路开关量输入
DI2	第二路开关量输入
DI3	第三路开关量输入
DI4	第四路开关量输入
DI5	第五路开关量输入
DI6	第六路开关量输入
DI7	第七路开关量输入
DI8	第八路开关量输入
DICOM2	第二组开关量输入端公共端
DI9	第九路开关量输入
DI10	第十路开关量输入
DI11	第十一路开关量输入
DI12	第十二路开关量输入
DI13	第十三路开关量输入
DI14	第十四路开关量输入
DI15	第十五路开关量输入
DI16	第十六路开关量输入

8 位拨码开关　　选择本模块的地址 （也就是 MODBUS 规约中的单元地址），便于组网。拨码开关标有数字‘1’的为最低位，标有‘8’的为最高位，

ON 的位置为‘0’，单元地址按照二进制表示。仅仅使用最低的五位，高三位系统保留。

比如： 1 ON 2 ON 3 OFF 4 ON 5 ON 对应的单元地址为 04 。

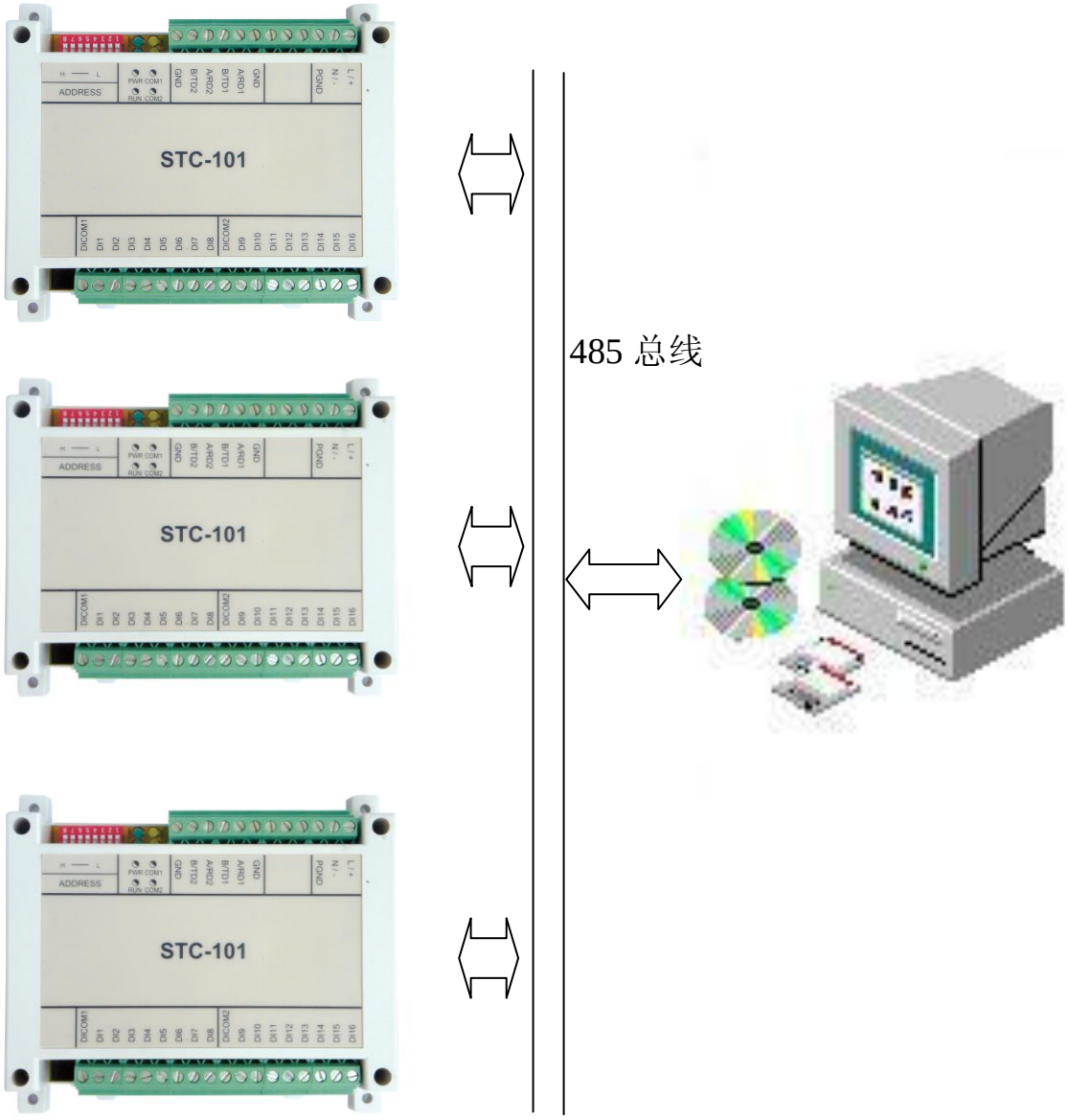
0 0 1 0 0

单元地址：00 或者 FF 是广播地址， 请不要使用， 否则会引起系统故障。

- 安装时需要小的一字起子
- 导线的线径除了电流引线外，不要超过 2.5 平方毫米，否则不易安装
- 如果工作场合干扰严重，请把 STC-101 放入铁制的机箱内，并对电源加电源滤波器
- 如果需要 IPC55 的工作条件，可以把 STC-101 安装在 IPC55 防护的机箱内

5. 典型方案

STC-1xx 作为测控终端，直接和计算机连接见下图



计算机可以采用一般的工业控制计算机，软件可以采用多种工控软件（如国产的组态王，国外的 INTOUCH 等）或其他的软件。

6. 订货须知

为了方便订货，让我们更好地为您服务，也避免差错和不必要地麻烦，在订货前我们希望您能够给我们提供如下信息：

数字量输入电压：

第一组

☐ DC110V ☐ DC220V ☐ AC110V ☐ AC220V ☐ AC380V

☐ DC12V ☐ DC24V ☐ DC48V ☐ 其他_____

第二组

☐ DC110V ☐ DC220V ☐AC110V ☐ AC220V ☐ AC380V

☐ DC12V ☐ DC24V ☐ DC48V ☐ 其他_____

电源电压

☐ DC 5V ☐ DC 12V ☐ DC 24V ☐ DC 48V

☐ AC 110V ☐ AC 220V ☐ 其他_____

通信接口

☐ RS232 ☐ RS485 ☐ 其他_____

注意！

本产品不防爆，不允许在有爆炸危险的环境下使用；本品不允许在医院、核电、航空等可能对生命有危险或者有着巨大安全风险的场合使用；本公司不对任何使用者或任何第三方遭受的间接的、偶然的、特殊的、相应而生的信用或投保的损害，包括利润、收入、数据或使用的损失负责。

公司地址：北京市海淀区紫竹院路广源闸 5 号广源大厦 312

邮政编码：100081

电话：010-59790086　传真：68703551

开户行：华夏银行北京紫竹桥支行

帐号：　801924046

网址：http://www.tengcon.com

技术：tech@tengcon.com　　技术支持：　support@ tengcon.com

市场：market@tengcon.com　销售：　　sales@ tengcon.com

地图：公司位置


广源大厦
万寿桥
丹龙大厦
苏州街会馆
光华写字楼
中友集团
韦伯豪
大慈寺路
金德嘉园
京密引水渠
汇景阁
南长河
北科大厦
民族大学南路
车道沟桥
紫竹院路
车道沟东里
信弘大厦
金属大厦
香格里拉写字楼
紫竹花园
科贸商务中心
楠迪大厦
中海紫金苑
首体溜冰馆公司写字楼
紫竹桥
紫竹院公园
荷花池
车道沟南里
紫竹院商住综合楼
华澳中心
车道沟南路
泛华商务楼
双紫渠
中建大厦
方园大厦
中海雅园
腾达大厦
万景公寓
百胜村
新世纪写字楼
北洼路西里
正泰办公大厦
花园写字楼
西苑路
又一村路
七贤村路
玲珑路
花园桥
车公庄西路
华通大厦
国兴家园
花园村小区
中经大厦
科源大厦
中海馥园
北京鑫桥房地产
玲珑公园
增光路