



<SF4101 Modbus\_Tcp Master>

作者	张娜	 <b>中科时代</b>   基于PC技术的工智机新时代 深圳市南山区粤海街道百度国际大厦西塔楼 官网: <a href="http://www.sinsegye.com.cn">www.sinsegye.com.cn</a> 邮箱: Sales@sinsegye.com.cn 热线电话: 400-013-2158
日期	2025/7/1	
版本	V1.6	
Email		

**更新说明:**

本文记录了<产品名称>的变更情况。

**<2025.7.1> / <V1.6>**

功能	变更类型	说明	相关文档
/	优化		
/	优化	增加插件工具安装 RTE 组件方式、补充并优化程序以及预期结果。	
/	优化		

## 目录

一、前言	4
1. 文件说明	4
2. 安全声明	4
二、概述	6
1. 使用场景	6
2. 整体架构	6
三、安装卸载	7
1. 安装要求	7
2. 安装过程	7
3. 卸载过程	11
四、技术说明	13
1. 快速启动	13
2. 本例实验操作步骤	14
3. 实验注意点	28
五、功能介绍	30
1. 主站配置读取从站多线圈	30
2. 主站配置读取从站输入寄存器	31
3. 主站配置读取从站离散输入	32
4. 主站配置读取从站保持寄存器	34
5. 主站配置写从站多线圈	35
6. 主站配置写从站多保持寄存器	36
7. 主站配置写从站单线圈	37
8. 主站配置写从站单保持寄存器	38
9. 主站配置连接 32 从站	39
六、附录	40
1. Linux 指令安装/卸载指南	40
2. 错误处理	41
3. 支持与服务	41

## 一、前言

### 1. 文件说明

本说明专为熟悉相关国家标准且经过专业培训的控制与自动化技术专家而制定。  
在安装与调试部件时，务必仔细审阅所有相关文件及以下说明。

合格人员应始终采用最新的有效文档进行操作。

责任人员必须确保所述产品的应用或使用完全符合所有安全要求，涵盖所有相关法律法规、指导原则及标准。

#### 1.1 . 免责声明

本文件经过精心编制，但鉴于所描述产品处于持续的开发与升级过程中，中科时代（深圳）计算机系统有限公司保留随时对文件进行修改和更新的权利，且无需事先通知。请注意，禁止依据数据图及本文件描述对已交付的产品进行任何改动。

对于因使用或信赖本手册所载明或未明示的信息而造成的任何损失或损害，中科时代计算机系统有限公司不承担任何责任。

#### 1.2 . 版权所有

本手册的所有权归中科时代计算机系统有限公司所有。未经书面许可，任何人不得以任何形式复制、分发、翻译或以其他方式使用本手册的全部或部分内容。

本手册受版权法保护。任何对本手册内容的复制、分发、翻译、展示、表演、演绎或使用，无论出于何种目的，均需得到中科时代计算机系统有限公司的明确许可。未经许可，任何行为均视为侵犯中科时代计算机系统有限公司的版权。

## 2. 安全声明

### 2.1 . 安全规程

为了您的安全，请阅读以下说明。始终遵守产品特定的安全说明，您可以在本文档的适当位置找到这些说明。

### 2.2 . 责任免除

所有组件都提供了硬件和软件配置。不允许对文件中所述以外的硬件或软件配置进行修改，中科时代不对文件所述外的硬件或软件负责。

### 2.3 . 人员资格

本说明仅适用于熟悉适用国家标准的经过培训的控制、自动化和驱动技术专家。

## 2.4 . 信号词

文档中使用的信号词分类如下。为了防止人员和财产受到伤害和损害，请阅读并遵守安全和警告通知。

## 2.5 . 个人伤害警示

<div></div> <div><div>□ 警告</div><div>危险的类型 说明不避开危险后果 说明如何避免危险的发生</div></div>	警告表示一种潜在的危险情况，如果不加以避免，可能会导致严重的伤害或死亡。
<div></div> <div><div>□ 注意</div><div>危险的类型 说明不避开危险后果 说明如何避免危险的发生</div></div>	注意表示潜在的危险情况，如果不避免，可能会导致轻度受伤或中度受伤，或导致设备损坏。
<div><div>提醒</div><div>危险的类型 说明不避开危险后果 说明如何避免危险的发生</div></div>	注意表示一种潜在的危险情况，如果不加以避免，可能只导致设备的损坏。

## 2.6 . 对财产或环境造成损坏的警告

<div><div>注意</div><div>危险的类型 说明不避开危险后果 说明如何避免危险的发生</div></div>	环境、设备或数据可能会被损坏。
--	-----------------

## 2.7 . 产品处理信息

例如，这些信息包括：行动建议、援助或有关产品的进一步信息。

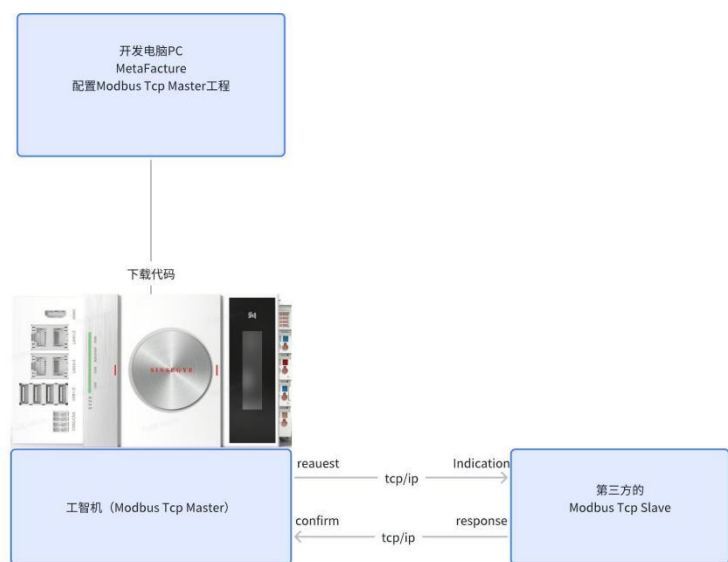
## 二、概述

Modbus 本身是一种信息交换的规范，Modbus TCP 则是透过 TCP/IP 来实现 Modbus 的一种方式，因此所有的信息都是通过 TCP/IP 来传输的；Modbus 协议属于 C/S 架构，Modbus Tcp Master 可以读写 Rtu Slave 的地址，实现数据交互；

### 1. 使用场景

- 设备间数据采集和监控（如 PLC 与传感器、HMI 的连接）。
- 过程自动化中的数据传输。
- 远程监控和控制系统。

### 2. 整体架构



- 注：本手册中用到的中科时代的软件包，均可以从官网的子页面获取。官网提供的版本可能比本手册中提到的版本更高，一般情况下这不会影响您按照本手册的例子执行相应的操作
- Modbus Tcp 是透过 TCP/IP 来实现 Modbus 的一种方式，因此所有讯息是透过 TCP/IP 来传输的，Modbus Tcp Master 可以对 Slave 发送读或者写的指令，Slave 收到指令后会回复确认信息，整个 Modbus 的沟通建立在一来一回的讯息交换上；
- 下表概述了各个产品组件

	产品组件	描述说明
主站	simodbusmaster_1.1.4_a md64.deb	Modbus tcp Master RTE 组件
	SF4100_ModbusMaster_ 1.0.0.2.library	Modbus tcp Master Metafacture 库文件
从站	modbuslave_1.1.3_amd 64.deb	Modbus tcp Slave RTE 组件
	sf4100_modbuslave_1.0. 0.1.library	Modbus tcp Slave Metafacture 库文件

### 三、安装卸载

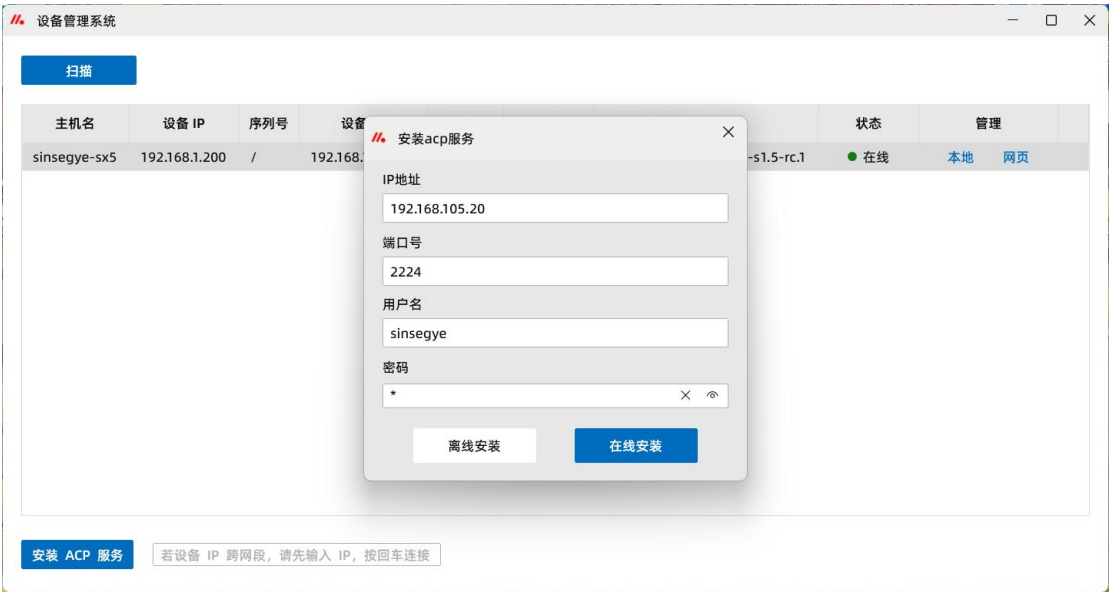
#### 1. 安装要求

- 中科时代出厂的工控机；
- 安装 DeviceManager.exe (0.0.2.4)软件

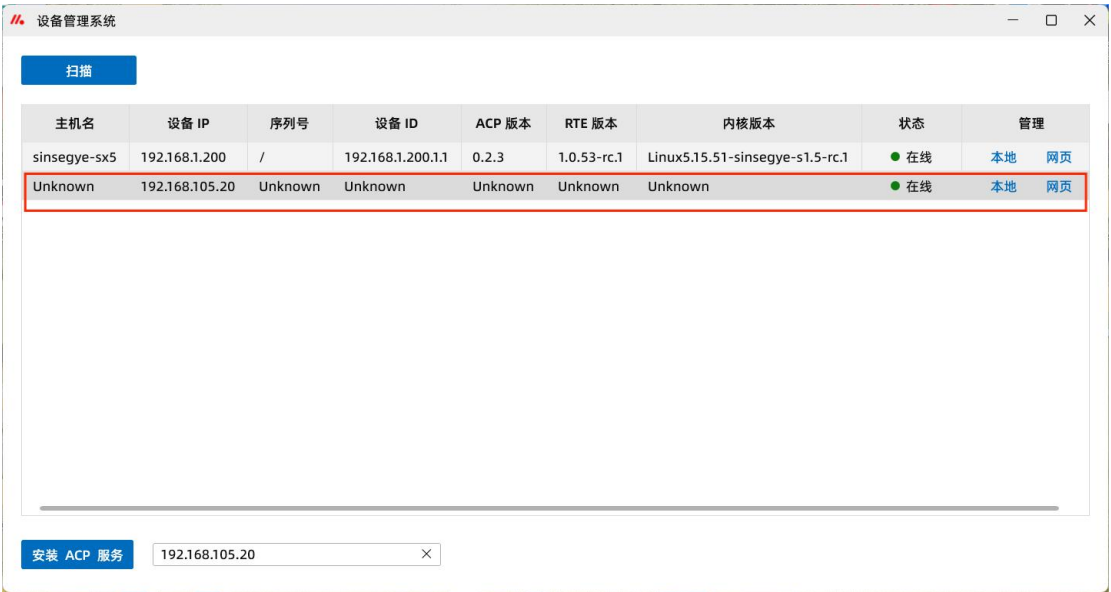
#### 2. 安装过程

## 2.1 . 工控机端安装 Modbus Tcp Master RTE 组件

- 运行 DeviceManager.exe 软件
- 安装 ACP 服务（以工控机（192.168.105.20）为例），选择“在线安装”

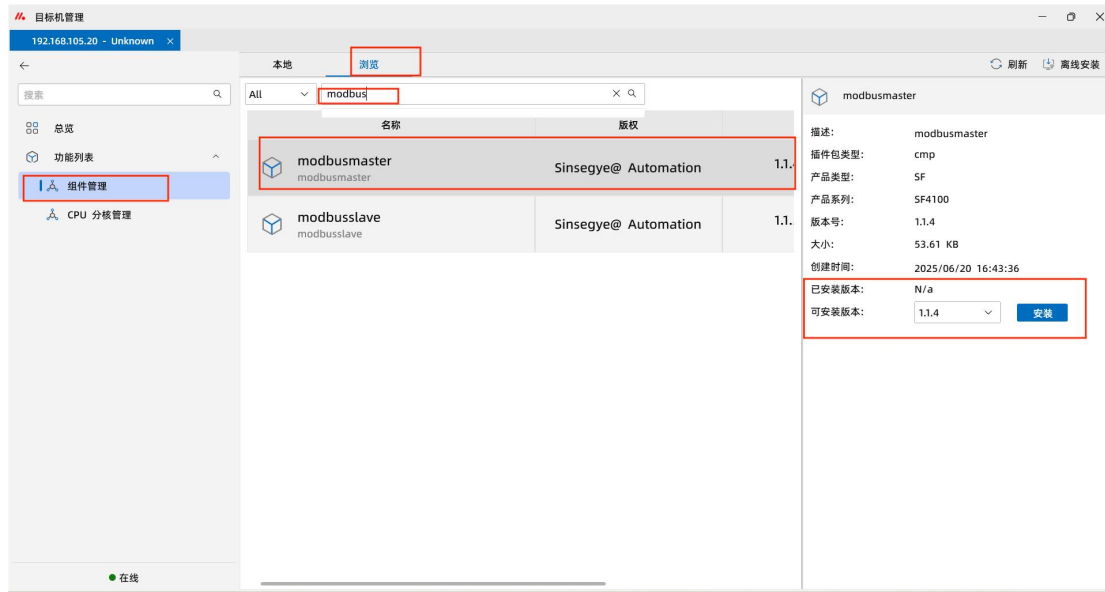


- 选择工控机（192.168.105.20），点击“本地”，进入本地安装界面；

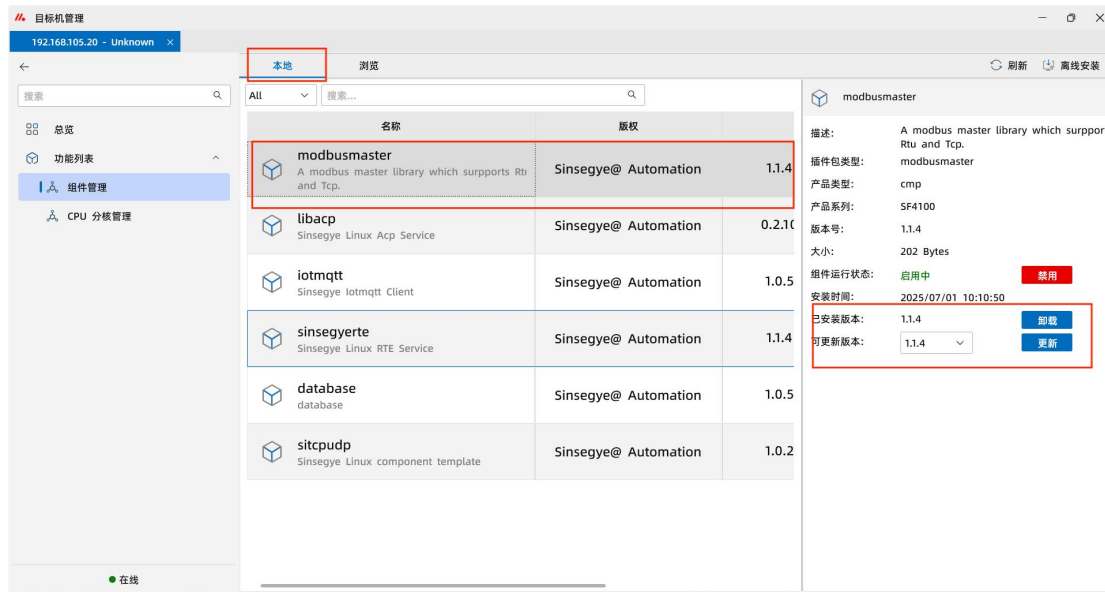


- 进入工控机设备管理器界面，点击"软件"下拉选择“组件管理”，进入“浏览页面”，选择"modbusmaster"查看安装版本并点击“安装”；





- 传输完成，弹窗选择确定安装；安装完成后，选择确认重启组件；
- 安装完成后，本地新增 1.1.4 版本的"modbusmaster"组件，

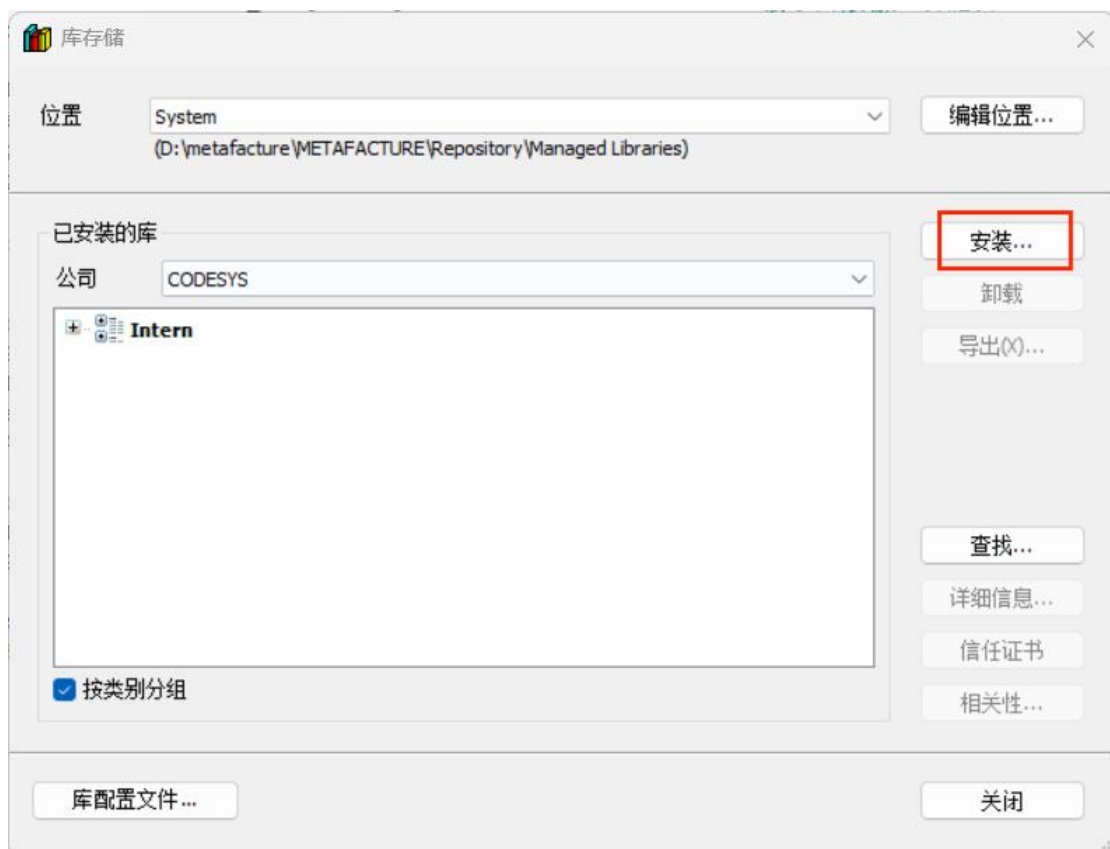


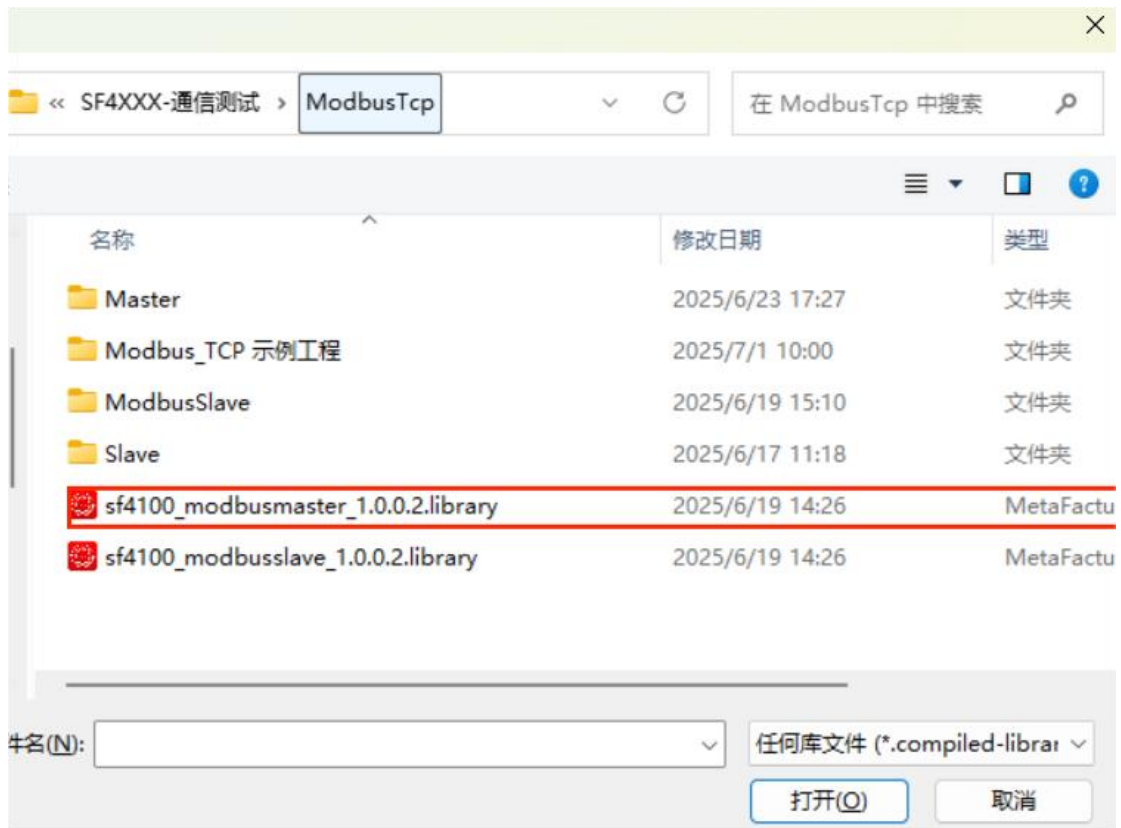
## 2.2 . Metafacture 安装 library

- 打开 Metafacture, 点击“工具” -- “库存储”

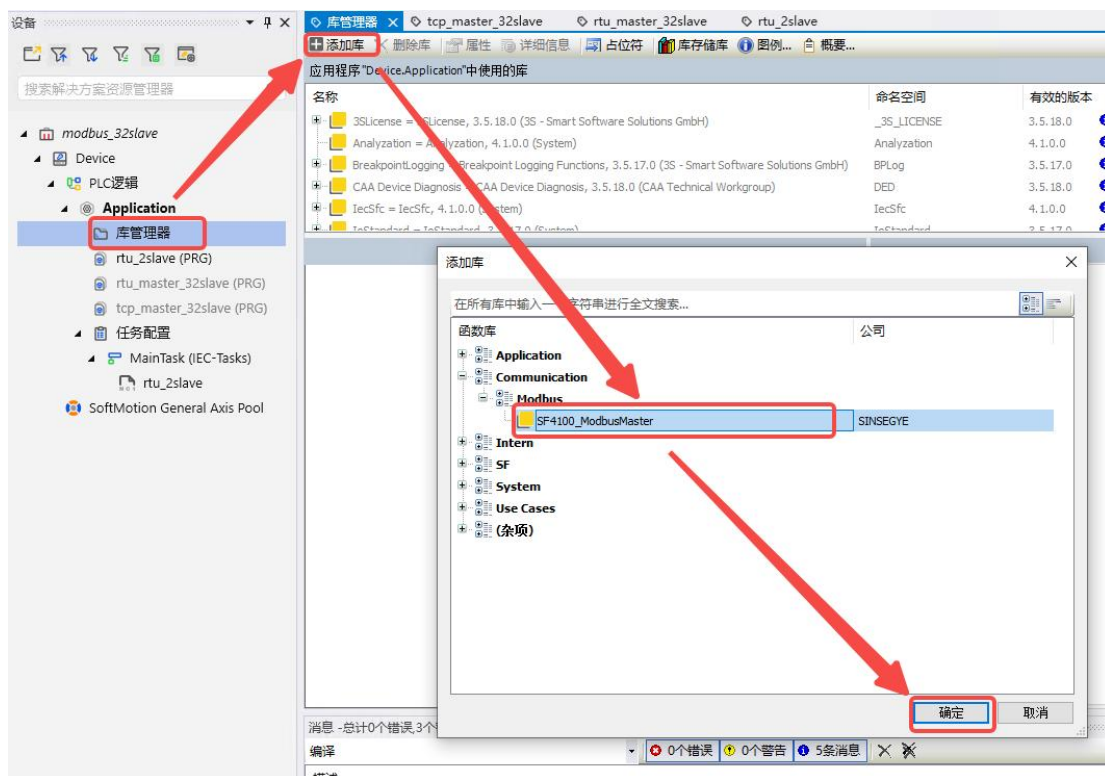


- 点击“安装” -- 选中 modbus master 的库文件, 点击“打开”





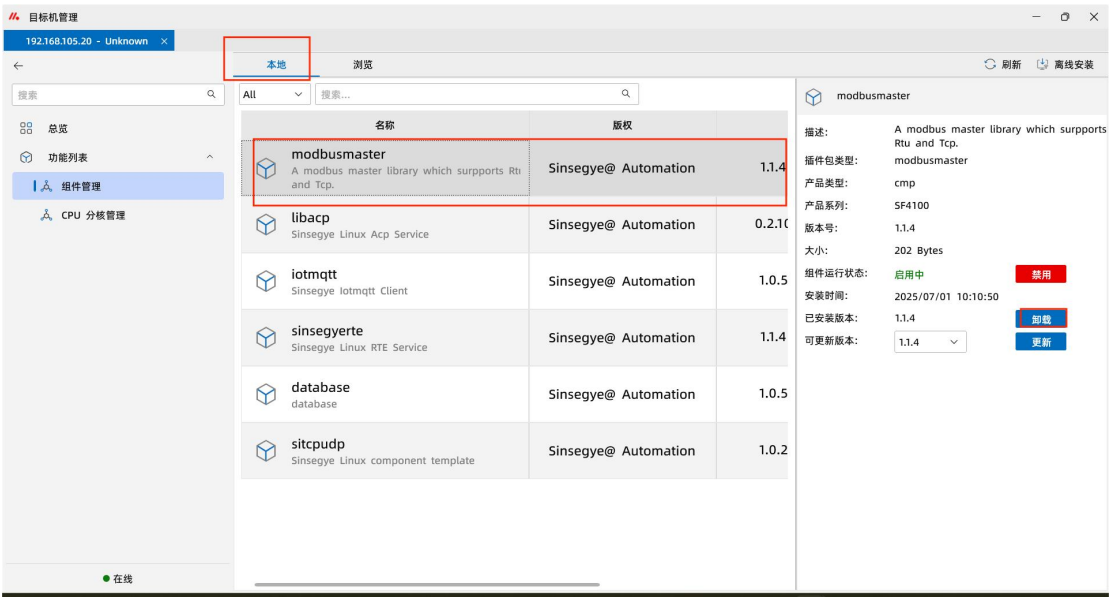
- 工程中点击“库管理器” -- “添加库” -- 选中 modbus 库点击“确定”



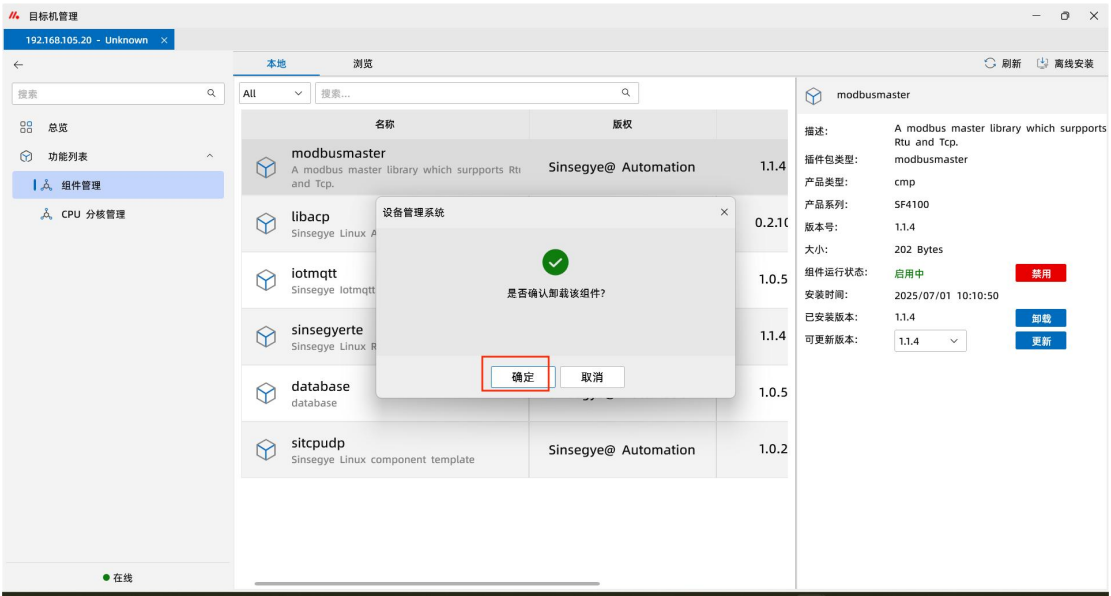
### 3. 卸载过程

### 3.1 . 工控机端卸载 Modbus Tcp Master RTE 组件

- 进入工控机设备管理器->“组件管理界面”->“本地”，选择组件"modbusmaster" 点击“卸载”；



- 确认卸载组件且确定重新加载；



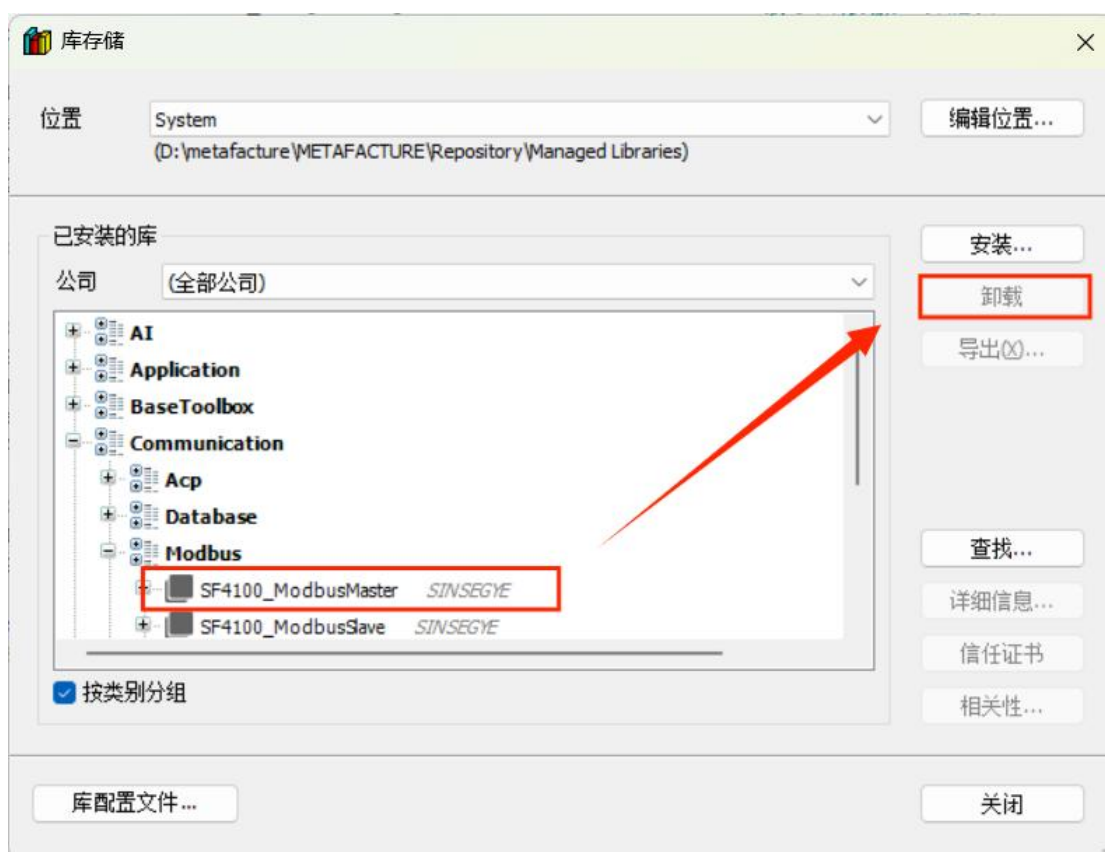
- 加载完成后，"modbusmaster" 组件被删除；

### 3.2 . 卸载 MetaFactory 侧的 Modbus Tcp master library

- MetaFacture 界面点击“工具”--“库存储”



- 对话框中选中安装的 modbus master 的库，点击“卸载”



## 四、技术说明

### 1. 快速启动

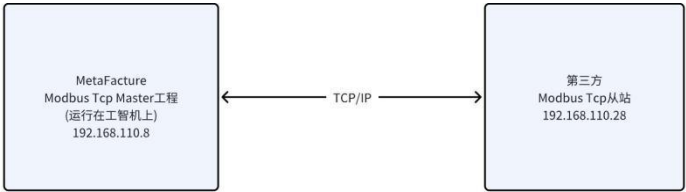
## 1.1 . 本例软、硬件配置

<b>硬件：</b>  1. SX20 工控机 2. Win10 PC 3. SX21 工控机	<b>软件：</b>  1. MetaFacutre V1.0.7.1 2. Modbus slave 从站工具
---	---

### 本例实验要求及原理

实验要求：按照“安装卸载”部分中的“安装过程”配置完成 Modbus Tcp Master 环境

#### 1、实验原理



- Metafactory Modbus Tcp 主站通过 TCP 连接向第三方 Modbus Tcp 从站发送请求，主要包含：功能码、目标寄存器的地址和数量、写操作时的数据；从站收到请求后先检查数据的合法性，然后执行请求的操作，响应主站请求；
- 上位机和工控机使用 EtherNet 连接；
- 上位机上，MetaFactory 下装工程到工控机。工程中会包含下面的试验操作步骤中的配置；
- 工控机与第三方 TCP/IP 客户端通过 EtherNet 连接；

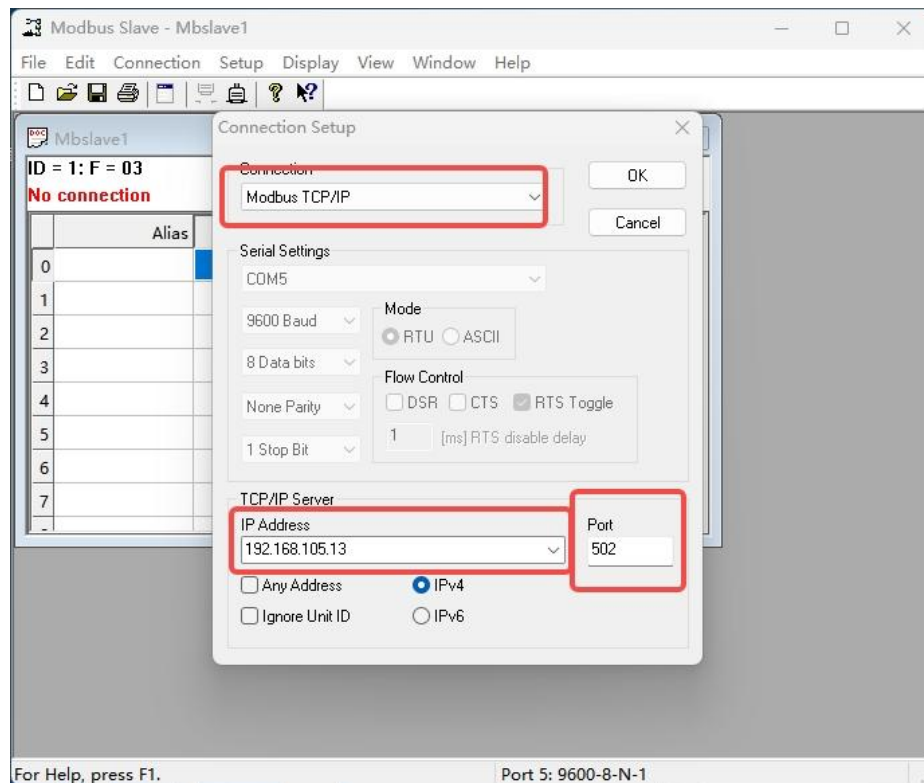
## 2 . 本例实验操作步骤

### 2.1 . 读操作

#### 2.1.1 . 测试环境

SX21 工智机作为主站，助手作为从站：

从站助手配置：



Connection 选择：Modbus TCP/IP

Ip Address：本机电脑 IP 地址（与工智机在同一网段）

Port :502（示例使用默认端口号）

### 2.1.2. 操作步骤

- 1、创建工程 ModbusTCPMaster 工程；
  - 2、按照如下操作完成读功能操作；
- 读多线圈的实验步骤如下：

#### a. POU 中声明区域调用 modbus tcp master 读线圈功能块

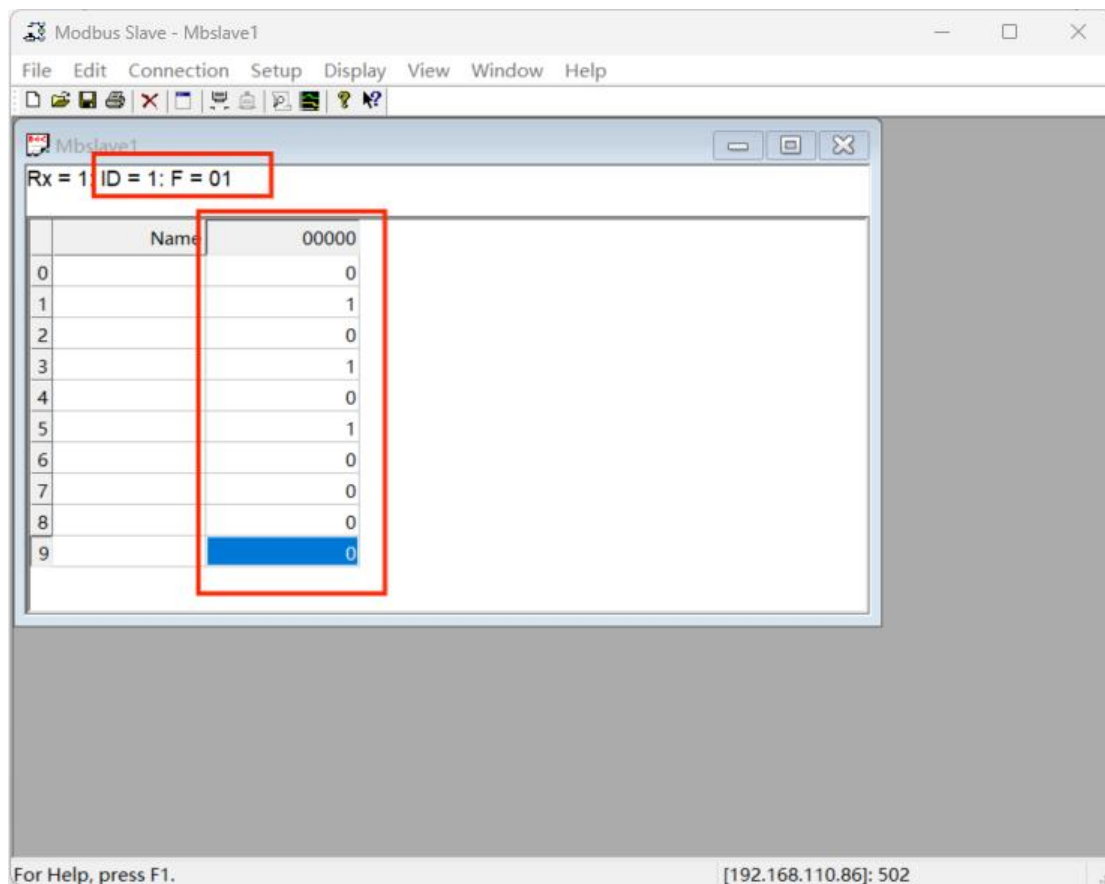
```
Shell
bRead          : BOOL ;
ReadCoils      : FB_MBTcpReadCoils;
ReadCoilData   : ARRAY[0..9] OF BYTE;
```

- POU 中程序区域调用 ReadCoils

Shell

```
ReadCoils(slPAddr:=ipaddr, //从站 IP 地址
          nTCPPort:=502, //从站端口
          nUnitID:=1, //从站 ID
          nQuantity:=10, //读取数量
          nMBAAddr:=0, //读取地址偏移量
          cbLength:=SIZEOF(ReadCoilData), //读取个数
          pDestAddr:=ADR(ReadCoilData), //存放数据的指针
          bExecute:=bRead, //启动读取, 上升沿
          tTimeout:=, bBUSY=>, bError=>, nErrId=> );
```

- 工程运行后触发 ReadCoils 中的 bExecute 上升沿执行读取线圈的值，即写入 bRead 为 True;
- 工程中调用功能块的 nUnitID 必须写为当前从站设置的 ID 值，不然会报错，并且读不到数据，后面调用其他功能块也是必须写 nUnitID。
- Modbus Slave 工具配置模式如下：



- MetaFacture 配置程序暂停并重新运行，获取结果如下：



表达式	类型	值	准备值	地址	注释
ipaddr	STRING...	'192.16...			设定从...
bRead	BOOL	FALSE			启动读取
ReadCoilData	ARRAY ...				读取从...
ReadCoilData[0]	BYTE	0			
ReadCoilData[1]	BYTE	1			
ReadCoilData[2]	BYTE	0			
ReadCoilData[3]	BYTE	1			
ReadCoilData[4]	BYTE	0			
ReadCoilData[5]	BYTE	1			
ReadCoilData[6]	BYTE	0			
ReadCoilData[7]	BYTE	0			
ReadCoilData[8]	BYTE	0			
ReadCoilData[9]	BYTE	0			
* ReadInputData	ARRAY ...				读取从...
* ReadInputRegsData	ARRAY ...				读取从...

## b. 读取离散输入的实验步骤如下：

- POU 中声明区域调用 modbus tcp master 读离散输入功能块

```

Shell
bRead                : BOOL ;
ReadInputs            : FB_MBTcpReadInputs;
ReadInputData         : ARRAY[0..9] OF BYTE;

```

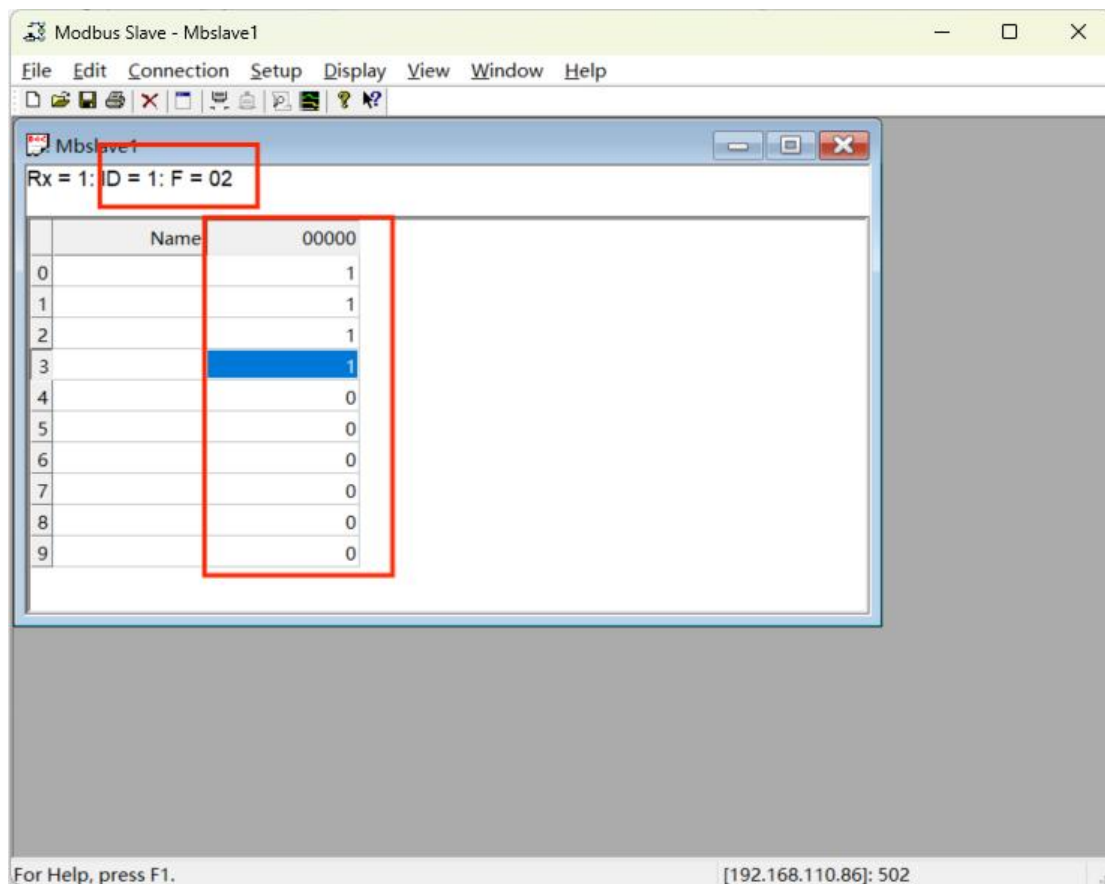
- POU 中程序区域调用 ReadInputs

```

Shell
ReadInputs(slPAddr:=ipaddr ,
           nTCPPort:=502 ,
           nUnitID:= 1,
           nQuantity:=10 ,
           nMBAAddr:=0 ,
           cbLength:=SIZEOF(ReadInputData) ,
           pDestAddr:=ADR(ReadInputData) ,
           bExecute:=bRead ,
           tTimeout:= , bBUSY=> , bError=> , nErrId=> );

```

- 工程运行后触发 ReadInputs 中的 bExecute 上升沿执行读取离散输入的值，即写入 bRead 为 True;
- Modbus Slave 工具配置模式如下：



- MetaFactory 配置程序暂停并重新运行，获取结果如下：

表达式	类型	值	准备值	地址	注释
bRead	BOOL	FALSE			启动读取
ReadCoilData	ARRAY ...				读取从 ...
ReadInputData	ARRAY ...				读取从 ...
ReadInputData[0]	BYTE	1			
ReadInputData[1]	BYTE	1			
ReadInputData[2]	BYTE	1			
ReadInputData[3]	BYTE	1			
ReadInputData[4]	BYTE	0			
ReadInputData[5]	BYTE	0			
ReadInputData[6]	BYTE	0			
ReadInputData[7]	BYTE	0			
ReadInputData[8]	BYTE	0			
ReadInputData[9]	BYTE	0			
ReadInputRegsData	ARRAY ...				读取从 ...
ReadRegsData	ARRAY ...				读取从 ...

#### c. 读取保持寄存器的实验步骤如下：

- POU 中声明区域调用 modbus tcp master 读保持寄存器功能块

```

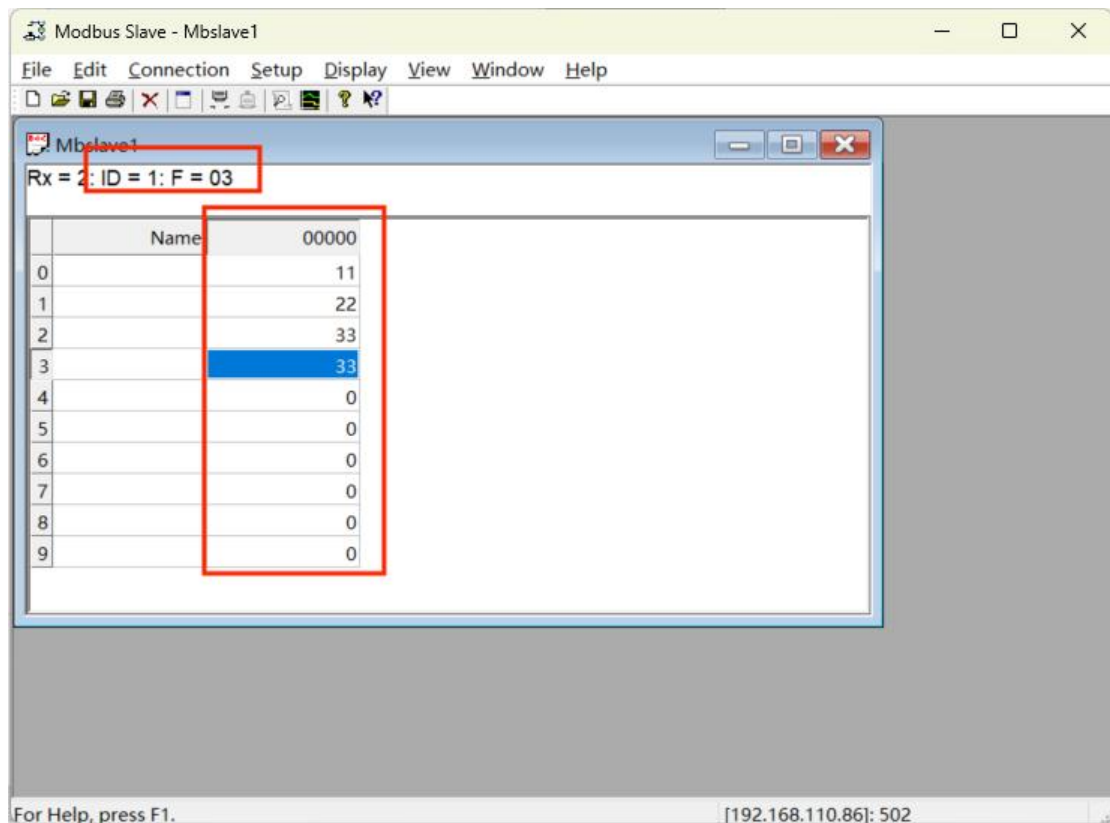
Shell
bRead          : BOOL ;
ReadRegs       : FB_MBTcpReadRegs;
ReadRegsData   : ARRAY[0..9] OF WORD;
  
```

- POU 中程序区域调用 ReadRegs

Shell

```
ReadRegs(sIPAddr:=ipaddr ,  
        nTCPPort:=502 ,  
        nUnitID:= 1,  
        nQuantity:=10 ,  
        nMBAAddr:=0 ,  
        cbLength:=SIZEOF(ReadRegsData) ,  
        pDestAddr:=ADR(ReadRegsData) ,  
        bExecute:=bRead ,  
        tTimeout:= , bBUSY=> , bError=> , nErrId=> );
```

- 工程运行后触发 ReadRegs 中的 bExecute 上升沿执行读取保持寄存器的值，即写入 bRead 为 True;
- Modbus Slave 工具配置模式如下：



- MetaFacture 配置程序暂停并重新运行，获取结果如下

表达式	类型	值	准备值	注释
bRead	BOOL	FALSE		启动读取
ReadCoilData	ARRAY ...			读取从站线圈数据
ReadInputData	ARRAY ...			读取从站输入数据
ReadInputRegsData	ARRAY ...			读取从站输入型寄存...
ReadRegsData	ARRAY ...			读取从站保持型寄存...
ReadRegsData[0]	WORD	11		
ReadRegsData[1]	WORD	22		
ReadRegsData[2]	WORD	33		
ReadRegsData[3]	WORD	33		
ReadRegsData[4]	WORD	0		
ReadRegsData[5]	WORD	0		
ReadRegsData[6]	WORD	0		
ReadRegsData[7]	WORD	0		
ReadRegsData[8]	WORD	0		

d. 读取输入寄存器的实验步骤如下：

- POU 中声明区域调用 modbus tcp master 读输入寄存器功能块

```

Shell
bRead          : BOOL ;
ReadInputRegs  : FB_MBTcpReadInputRegs;
ReadInputRegsData : ARRAY[0..9] OF WORD;

```

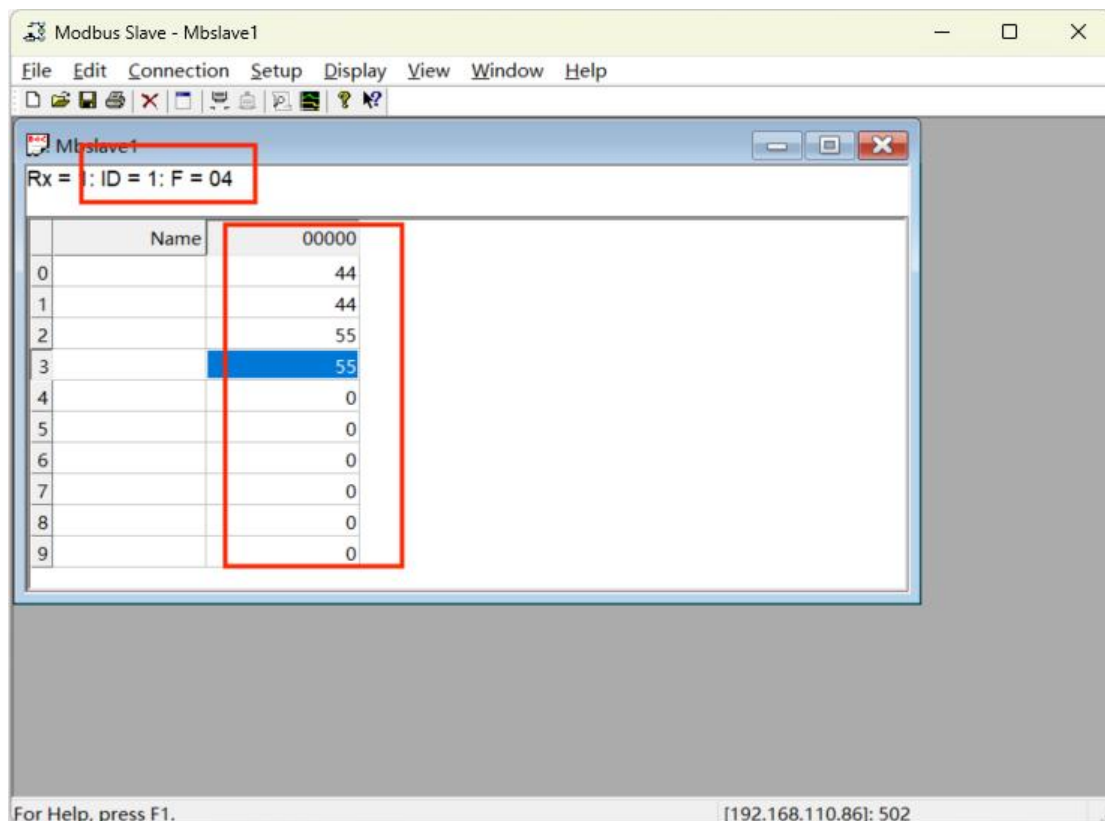
- POU 中程序区域调用 ReadInputRegs

```

Shell
ReadInputRegs(sIPAddr:=ipaddr ,
               nTCPPort:=502 ,
               nUnitID:=1 ,
               nQuantity:=10 ,
               nMBAAddr:=0 ,
               cbLength:=SIZEOF(ReadInputRegsData) ,
               pDestAddr:=ADR(ReadInputRegsData) ,
               bExecute:=bRead ,
               tTimeout:= , bBUSY=> , bError=> , nErrId=> );

```

- 工程运行后触发 ReadInputRegs 中的 bExecute 上升沿执行读取输入寄存器的值，即写入 bRead 为 True;
- Modbus Slave 工具配置模式如下：



- MetaFactory 配置程序暂停并重新运行，获取结果如下：

表达式	类型	值	准备值	注释
ipaddr	STRING...	'192.168.110.86'		设定从站IP地址
bRead	BOOL	FALSE		启动读取
ReadCoilData	ARRAY ...			读取从站线圈数据
ReadInputData	ARRAY ...			读取从站输入数据
ReadInputRegsData	ARRAY ...			读取从站输入寄存器数据
ReadInputRegsData[0]	WORD	44		
ReadInputRegsData[1]	WORD	44		
ReadInputRegsData[2]	WORD	55		
ReadInputRegsData[3]	WORD	55		
ReadInputRegsData[4]	WORD	0		
ReadInputRegsData[5]	WORD	0		
ReadInputRegsData[6]	WORD	0		
ReadInputRegsData[7]	WORD	0		
ReadInputRegsData[8]	WORD	0		
ReadInputRegsData[9]	WORD	0		
ReadRetsData	ARRAY			读取从站保持寄存器数据

## 2.2 . 写操作

### 2.2.1 . 测试环境

SX21 工控机作为主站，助手作为从站。

### 2.2.2. 操作步骤

- 1、创建工程 ModbusTCP Master 工程；
- 2、按照如下操作完成写功能操作；

a. 写多线圈的实验步骤如下：

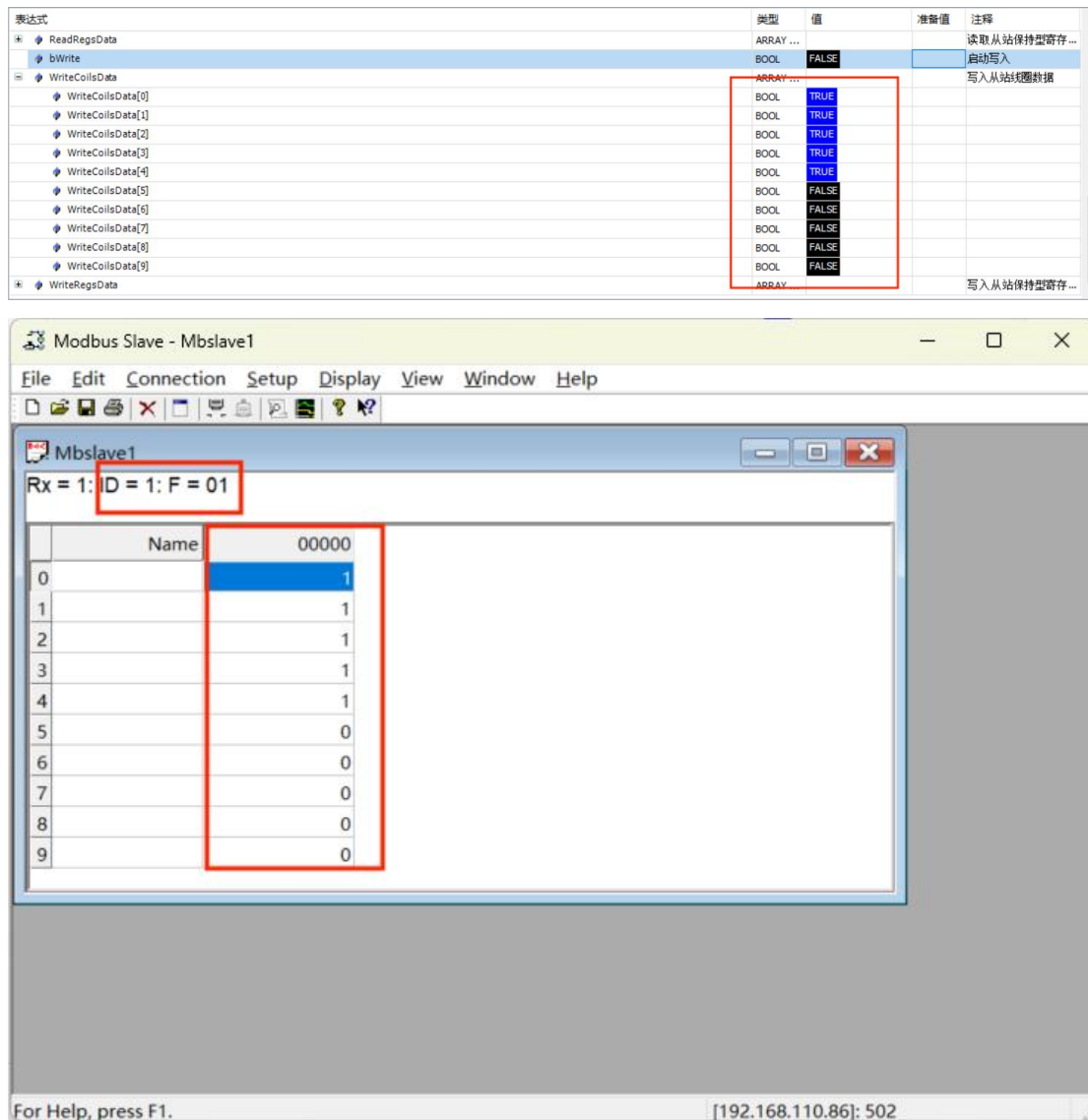
- POU 中声明区域调用 ModbusTCP Master 写线圈功能块

```
Shell
bWrite          : BOOL ;
WriteCoils       : FB_MBTcpWriteCoils;
WriteCoilsData   : ARRAY[0..9] OF BOOL:= [1,1,1,1,1];
```

- POU 中程序区域调用 WriteCoils 功能块

```
Shell
WriteCoils(sIPAddr:=ipaddr ,
           nTCPPort:=502 ,
           nUnitID:=1 ,
           nQuantity:=10 ,
           nMBAAddr:=0 ,
           cbLength:=SIZEOF(WriteCoilsData) ,
           pSrcAddr:=ADR(WriteCoilsData) ,
           bExecute:=bWrite ,
           tTimeout:= , bBUSY=> , bError=> , nErrId=> );
```

- 工程运行后触发 WriteCoils 中的 bExecute 上升沿执行写多线圈的值，即写入 bWrite 为 True;
- 从站显示结果：



b. 写多保持寄存器的实验步骤如下：

- POU 中声明区域调用 modbus tcp master 写多保持寄存器功能块

Shell

```
bWrite          : BOOL ;
WriteRegs       : FB_MBTcpWriteRegs;

WriteRegsData   : ARRAY[0..9] OF
WORD:=[16#FF,16#FE,16#FD,16#FC,16#FB,16#FA,16#F9,16#F8,1
6#F7,16#F6];
```

- POU 中程序区域调用 WriteRegs

## Shell

```
WriteRegs(sIPAddr:=ipaddr ,  
          nTCPPort:=502 ,  
          nUnitID:=1 ,  
          nQuantity:=10 ,  
          nMBAAddr:=0 ,  
          cbLength:=SIZEOF(WriteRegsData) ,  
          pSrcAddr:=ADR(WriteRegsData) ,  
          bExecute:=bWrite ,  
          tTimeout:= , bBUSY=> , bError=> , nErrId=> );
```

- 工程运行后触发 WriteRegs 中的 bExecute 上升沿执行写多保持寄存器的值，即写入 bWrite 为 True;
- 从站显示结果：

The screenshot displays the Modbus Slave software interface. The top window, titled '表达式' (Expressions), shows a list of variables and their values. The 'bWrite' variable is highlighted with a red box, showing a value of 'FALSE'. Below it, the 'WriteRegsData' array is shown, with values ranging from 255 down to 246. The bottom window, titled 'Modbus Slave - Mbslave1', shows the 'Rx = 1: ID = 1: F = 03' status. Below this, a table of register data is displayed, with values ranging from 255 down to 246. The table is highlighted with a red box.

名称	值
0	255
1	254
2	253
3	252
4	251
5	250
6	249
7	248
8	247
9	246



## 2.3. 两台工控机相互访问

### 2.3.1 . 测试环境

SX20 工控机作为主站， SX21 工控机作为从站。

Modbus Slave 端使用说明：

- Modbus slaver 默认 502 端口
- 新增 M 区 、 Q 区配置：
  - Q 区---功能码： 01 ， 02 、 05 位变量操作（读写）
  - M 区---功能码： 03 ， 04 、 06 寄存器变量操作（读写）
- 使用同一 Slave 功能块最多支持 15 个从站的创建。

### 2.3.2. 操作步骤

1、创建工程 ModbusTCP Slave 工程；

2、配置从站工程声明变量；

- POU 中声明区域调用 ModbusTCP Slave 功能块

```
PROGRAM ModbusTCP Slave
VAR
    TCPslave          :   MetaCore_ModbusSlave.FB_ModbusTCP Slave;
                        //Modbus TCP 从站功能块

    Coil              :   ARRAY[0..9] OF BOOL:=[1,0,1,0,1,0,1,0,1,0];
                        //初始化线圈数据

    DiscreteInput     :   ARRAY[0..9] OF BOOL:=[0,1,0,1,0,1,0,1,0,1];
                        //初始化离散输入数据

    InputRegister     :   ARRAY[0..9] OF
WORD:=[11,22,33,44,55,66,77,88,99,16#AA];
                        //初始化输入型寄存器数据

    HoldingRegister   :   ARRAY[0..9] OF
WORD:=[16#AA,99,88,77,66,55,44,33,22,11];
                        //初始化保持型寄存器数据
END_VAR
```

- POU 中程序区域调用 MetaCore\_ModbusSlave.FB\_ModbusTCP Slave

```

Shell
TCPslave(
    bEnable:=TRUE ,                //Modbus TCP 从站功能块使能
    ePort:= ,                      //Modbus RTU 参数, Modbus TCP 不填
    slpAddr:='192.168.105.11' ,    //Modbus TCP 从站 IP 地址
    uilpPort:=503 ,                //Modbus TCP 从站端口, 本例使用 503 端口
    udiTimeOut:= ,                 //超时时间
    pbyCoil:=ADR(Coil) ,           //线圈指针
    pbyDiscreteInput:=ADR(DiscreteInput) , //离散输入指针
    pbyInputRegister:=ADR(InputRegister) , //输入型寄存器指针
    pbyHoldingRegister:=ADR(HoldingRegister) , //保持型寄存器指针
    udiRwBitSize:=SIZEOF(Coil) ,   //线圈的个数
    udiInBitSize:=SIZEOF(DiscreteInput) , //离散输入的个数
    udiInWordSize:=SIZEOF(InputRegister) , //输入型寄存器的个数
    udiRwWordSize:=SIZEOF(HoldingRegister) , //保持型寄存器的个数
    psMbTest:= ,                   //测试用, 不可填
    bBusy=> , bActive=> , bConnect=> , bError=> , eError=> , ulRecvCounter=> ,
    ulSendCounter=> , sLocalIp=> , sClientIp=> );

```

3、创建工程 ModbusTCPMaster 工程;

4、按照如下操作完成写功能操作;

**a. 写多线圈的实验步骤如下:**

- POU 中声明区域调用 ModbusTCP Master 写线圈功能块

```

Shell
bWrite          : BOOL ;
WriteCoils      : FB_MBTcpWriteCoils;
WriteCoilsData  : ARRAY[0..9] OF BOOL:=[1,1,1,1,1];

```

- POU 中程序区域调用 WriteCoils 功能块

## Shell

```
WriteCoils(sIPAddr:=ipaddr ,
          nTCPPort:=503 ,
          nUnitID:=1 ,
          nQuantity:=10 ,
          nMBAAddr:=0 ,
          cbLength:=SIZEOF(WriteCoilsData) ,
          pSrcAddr:=ADR(WriteCoilsData) ,
          bExecute:=bWrite ,
          tTimeout:= , bBUSY=> , bError=> , nErrId=> );
```

- 工程运行后触发 WriteCoils 中的 bExecute 上升沿执行写多线圈的值，即写入 bWrite 为 True;
- 主站数据

表达式	类型	值	准备值	地址	注释
* ReadCoilData	ARRAY ...				读取从 ...
* ReadInputData	ARRAY ...				读取从 ...
* ReadInputRegsData	ARRAY ...				读取从 ...
* ReadRegsData	ARRAY ...				读取从 ...
* bWrite	BOOL	FALSE			启动写入
WriteCoilsData	ARRAY ...				写入从 ...
WriteCoilsData[0]	BOOL	TRUE			
WriteCoilsData[1]	BOOL	TRUE			
WriteCoilsData[2]	BOOL	TRUE			
WriteCoilsData[3]	BOOL	TRUE			
WriteCoilsData[4]	BOOL	TRUE			
WriteCoilsData[5]	BOOL	FALSE			
WriteCoilsData[6]	BOOL	FALSE			
WriteCoilsData[7]	BOOL	FALSE			
WriteCoilsData[8]	BOOL	FALSE			
WriteCoilsData[9]	BOOL	FALSE			
* WriteRegsData	ARRAY ...				写入从 ...

- 从站显示结果：

表达式	类型	值	准备值	地址	注释
* TCPSlave	MetaCo...				Modbus ...
Coil	ARRAY ...				初始化 ...
Coil[0]	BOOL	TRUE			
Coil[1]	BOOL	TRUE			
Coil[2]	BOOL	TRUE			
Coil[3]	BOOL	TRUE			
Coil[4]	BOOL	TRUE			
Coil[5]	BOOL	FALSE			
Coil[6]	BOOL	FALSE			
Coil[7]	BOOL	FALSE			
Coil[8]	BOOL	FALSE			
Coil[9]	BOOL	FALSE			
* DiscreteInput	ARRAY ...				初始化 ...
* InputRegister	ARRAY ...				初始化 ...
* HoldingRegister	ARRAY ...				初始化 ...

## b. 写多保持寄存器的实验步骤如下：

- POU 中声明区域调用 modbus tcp master 写多保持寄存器功能块

## Shell

```
bWrite          : BOOL ;
WriteRegs       : FB_MBTcpWriteRegs;

WriteRegsData   : ARRAY[0..9] OF
WORD:=[16#FF,16#FE,16#FD,16#FC,16#FB,16#FA,16#F9,16#F8,16#F7,16#F6];
```

- POU 中程序区域调用 WriteRegs

## Shell

```
WriteRegs(sIPAddr:=ipaddr ,  
          nTCPPort:=503 ,  
          nUnitID:=1 ,  
          nQuantity:=10 ,  
          nMBAAddr:=0 ,  
          cbLength:=SIZEOF(WriteRegsData) ,  
          pSrcAddr:=ADR(WriteRegsData) ,  
          bExecute:=bWrite ,  
          tTimeout:= , bBUSY=> , bError=> , nErrId=> );
```

- 工程运行后触发 WriteRegs 中的 bExecute 上升沿执行写多保持寄存器的值，即写入 bWrite 为 True;
- 主站数据

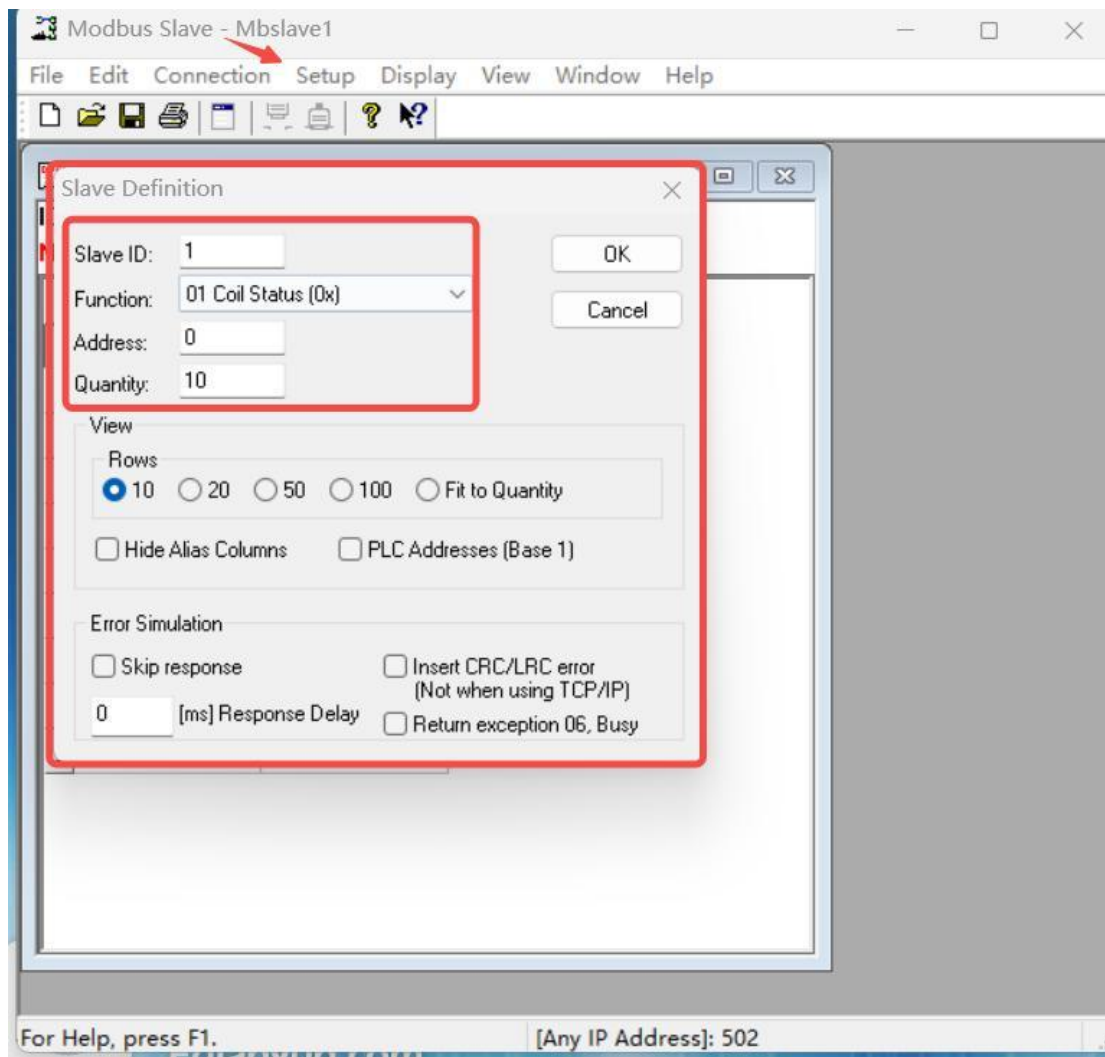
表达式	类型	值	准备值	地址	注释
* ReadCoilData	ARRAY ...				读取从 ...
* ReadInputData	ARRAY ...				读取从 ...
* ReadInputRegsData	ARRAY ...				读取从 ...
* ReadRegsData	ARRAY ...				读取从 ...
bWrite	BOOL	FALSE			启动写入
* WriteCoilsData	ARRAY ...				写入从 ...
WriteRegsData	ARRAY ...				写入从 ...
WriteRegsData[0]	WORD	255			
WriteRegsData[1]	WORD	254			
WriteRegsData[2]	WORD	253			
WriteRegsData[3]	WORD	252			
WriteRegsData[4]	WORD	251			
WriteRegsData[5]	WORD	250			
WriteRegsData[6]	WORD	249			
WriteRegsData[7]	WORD	248			
WriteRegsData[8]	WORD	247			
WriteRegsData[9]	WORD	246			

- 从站显示结果:

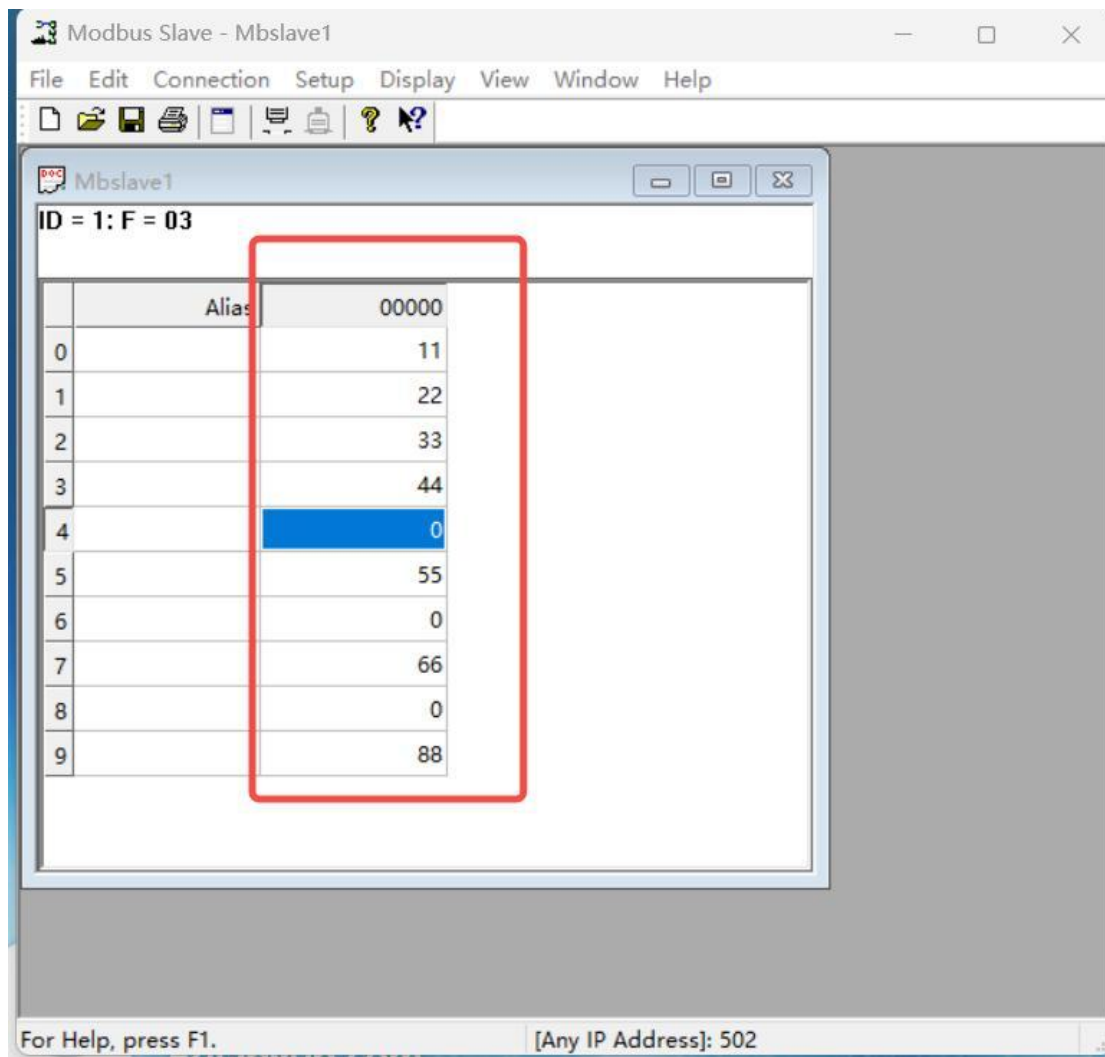
表达式	类型	值	准备值	地址	注释
* TCPSlave	MetaCo...				Modbus ...
* Coil	ARRAY ...				初始化 ...
* DiscreteInput	ARRAY ...				初始化 ...
* InputRegister	ARRAY ...				初始化 ...
HoldingRegister	ARRAY ...				初始化 ...
HoldingRegister[0]	WORD	255			
HoldingRegister[1]	WORD	254			
HoldingRegister[2]	WORD	253			
HoldingRegister[3]	WORD	252			
HoldingRegister[4]	WORD	251			
HoldingRegister[5]	WORD	250			
HoldingRegister[6]	WORD	249			
HoldingRegister[7]	WORD	248			
HoldingRegister[8]	WORD	247			
HoldingRegister[9]	WORD	246			

## 3. 实验注意点

- 1、实验中用到的从站的 IP、端口请改成实际使用的从站 ip、端口；
- 2、实验中使用的从站配置案例如下：
  - 配置 PC 为从站：
  - 安装工具 ModbusSlave
  - 配置模式：SlaveID、功能码（01，02，03，04）开始地址、队列



- 设置寄存器的值/状态



## 五、功能介绍

### 1. 主站配置读取从站多线圈

#### 1.1. 功能块 FB\_MBTcpReadCoils 介绍



#### 1.2. 参数介绍

- 输入参数

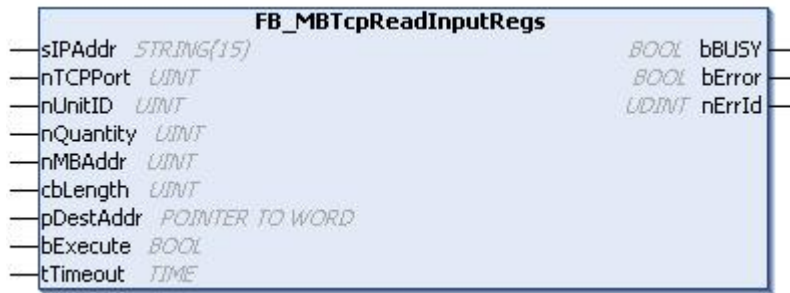
参数名称	参数类型	描述
sIPAddr	string	从站的 IP
nTCPPort	UINT	从站的端口
nUnitID	BYTE	从站 ID
nQuantity	WORD	从线圈起始位置读取的长度,一次最多读取 125 位
nMBAAddr	WORD	线圈起始位置
cbLength	UDINT	存储读取的字节大小
pDestAddr	POINTER TO BYTE	要读的存放数据的存放地址
bExecute	BOOL	上升沿触发执行
tTimeout	Time	超时

- 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 2 . 主站配置读取从站输入寄存器

### 2.1 . 功能块 FB\_MBTcpReadInputRegs 介绍



## 2.2 . 参数介绍

### • 输入参数

参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nQuantity	UINT	读取多少位输入寄存器，一次最多读取 125 位
nMBAAddr	UINT	读取的输入寄存器起始位置
cbLength	UINT	存储读取的字节大小
pDestAddr	POINTER TO WORD	要读的存放数据的存放地址
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

### • 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 3 . 主站配置读取从站离散输入



### 3.1 . 功能块 FB\_MBTcpReadInputs 介绍



### 3.2 . 参数介绍

• 输入参数

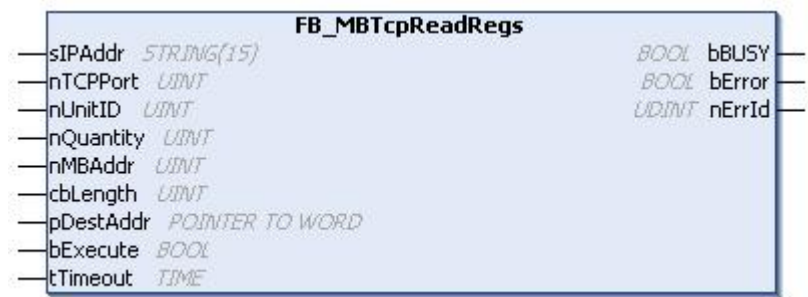
参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nQuantity	UINT	读取多少位离散输入
nMBAAddr	UINT	读取的离散输入起始位置
cbLength	UINT	存储读取的字节大小
pDestAddr	POINTER TO BYTE	要读的存放数据的存放地址
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

• 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 4 . 主站配置读取从站保持寄存器

### 4.1 . 功能块 FB\_MBTcpReadRegs 介绍



### 4.2 . 参数介绍

- 输入参数

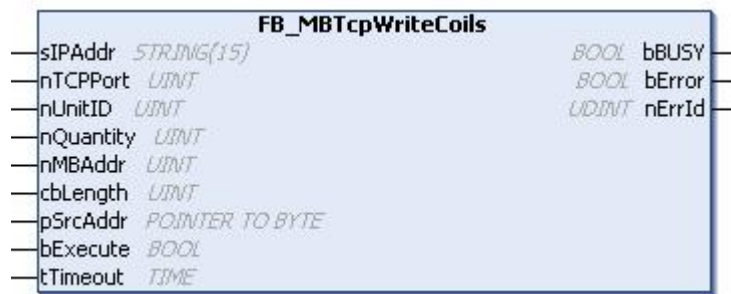
参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nQuantity	UINT	读取多少位保持寄存器，一次最多读取 125 位
nMBAAddr	UINT	读取的保持寄存器起始位置
cbLength	UINT	存储读取的字节大小
pDestAddr	POINTER TO word	要读的存放数据的存放地址
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

- 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 5．主站配置写从站多线圈

### 5.1．功能块 FB\_MBTcpWriteCoils 介绍



## 5.2 . 参数介绍

### • 输入参数

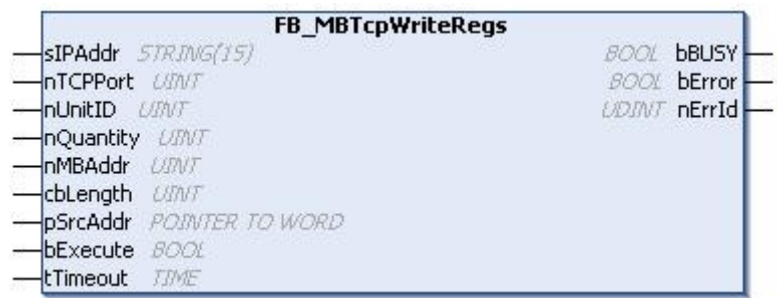
参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nQuantity	UINT	写多少位线圈
nMBAAddr	UINT	要写入的线圈起始位置
cbLength	UINT	存储写入的字节大小
pSrcAddr	POINTER TO BYTE	要写入的数据的存放地址
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

### • 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 6 . 主站配置写从站多保持寄存器

## 6.1 . 功能块 FB\_MBTcpWriteRegs 介绍



## 6.2 . 参数介绍

- 输入参数

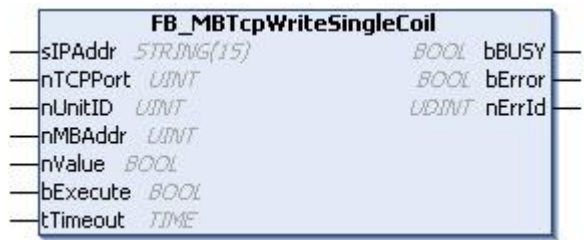
参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nQuantity	UINT	写多少位线圈，一次最多写 123 位
nMBAAddr	UINT	要写入的保持寄存器起始位置
cbLength	UINT	存储写入的字节大小
pSrcAddr	POINTER TO WORD	要写入的数据的存放地址
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

- 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 7 . 主站配置写从站单线圈

## 7.1 . 功能块 FB\_MBTcpWriteSingleCoil 介绍



## 7.2 . 参数介绍

### • 输入参数

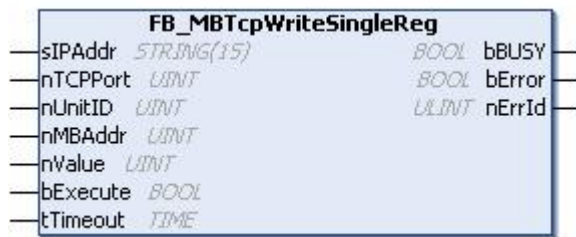
参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nMBAddr	UINT	要写入的单线圈的位置
nValue	BOOL	要写入的值
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

### • 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 8 . 主站配置写从站单保持寄存器

### 8.1 . 功能块 FB\_MBTcpWriteSingleReg 介绍



## 8.2 . 参数介绍

- 输入参数

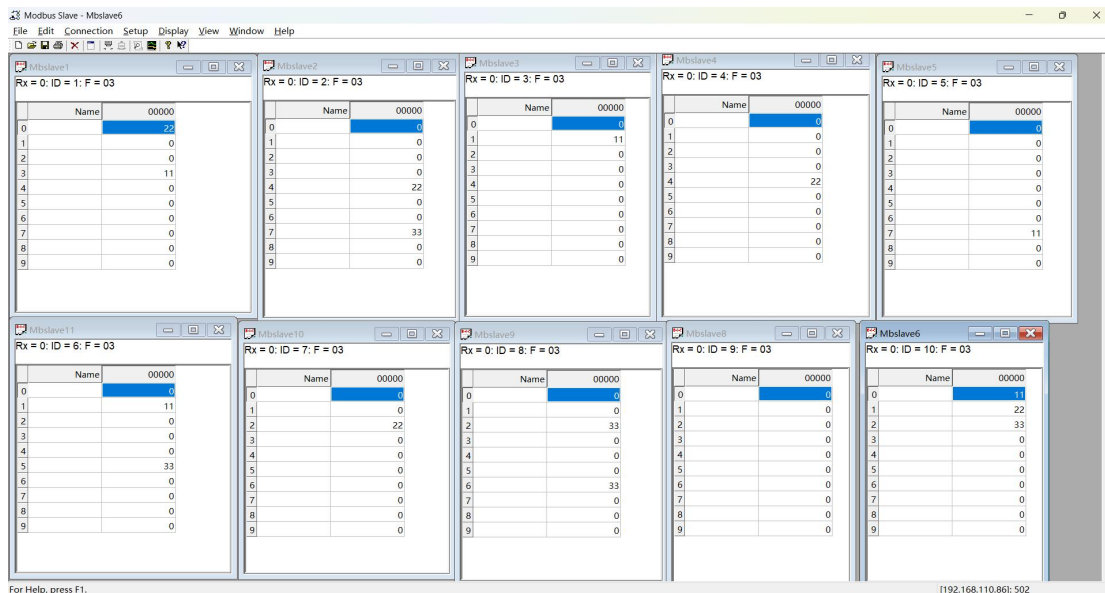
参数名称	参数类型	描述
sIPAddr	string	从站 ip
nTCPPort	UINT	从站端口
nUnitID	UINT	从站 id
nMBAddr	UINT	要写入的单保持寄存器的位置
nValue	UINT	要写入的值
bExecute	BOOL	上升沿触发执行
tTimeout	TIME	超时

- 输出参数

参数名称	参数类型	描述
bBUSY	BOOL	功能块激活时设置，直到确认接收
bError	BOOL	如果命令在传输过程中出现错误，则设置 true 直到 bBusy 输出被重置
nErrId	UDINT	当 bError 输出被设置，提供错误号

## 9 . 主站配置连接 10 从站

- 使用模拟软件配置 10 个从站（各个从站 ID 配置不一样），参考如下：



- 主站配置连接 10 个从站进行读写操作，测试读写正常

## 六、附录

### 1. Linux 指令安装/卸载指南

#### 1.1 . 安装要求

- 中科时代出厂的工控机；
- 熟悉基础的 Linux 操作命令

开始安装前，请熟读 [linux 基础操作](#) 中的操作示例

#### 1.2 . 安装过程

工控机端安装 Modbus Tcp Master RTE 组件

- 上传 deb 包到工控机 Linux 环境的 /home/sinsegye 目录下
- 上传完成后在工控机上执行命令安装（参考下方截图，如果模块文件名发生变化则命令行中的文件名做相应更改）

Shell

```
cd $HOME
sudo dpkg -i modbusmaster_1.1.4_amd64.deb
```

- 修改 RTE 的配置文件，ComponentManger 模块下加入 modbusmaster

Shell

```
[ComponentManager]
Component.0=retainDeamon
Component.1=CmpCanBusUtils
Component.2=CmpSinsegyeLibs
Component.3=SinsegyeCmp
Component.4=modbusmaster
```

- 重启 RTE 服务，使新加入的 modbusmaster 被调用



```
Shell
sudo systemctl restart sinsegerte.service
```

### 1.3 . 更新安装

工智机端升级 Modbus Tcp Master RTE 组件

- 上传 deb 包到工智机 Linux 环境的/home/sinsegye 目录下
- 上传完成后在工智机上执行命令安装（参考下方截图，如果模块文件名发生变化则命令行中的文件名做相应更改）

```
Shell
cd $HOME
sudo dpkg -i modbusmaster_1.1.4_amd64.deb
```

- 重启 RTE 服务，使新升级的 modbusmaster 生效

```
Shell
sudo systemctl restart sinsegerte.service
```

### 三、卸载过程

工智机端卸载 Modbus Tcp Master RTE 组件

- 工智机上执行命令卸载 modbusmaster

```
Shell
sudo dpkg -r simodbusmaster
```

- 修改 RTE 的配置文件，ComponentManger 模块下去掉 modbusmaster

```
Shell
sudo vim /usr/local/etc/SinsegyeRTE/SinsegyeRTE.cfg
```

- 重启 RTE 服务

```
Shell
sudo systemctl restart sinsegerte.service
```

## 2 . 错误处理

1. 需要先确定网络通讯正常，确定主站要请求的从站地址都存在；
2. 确保主/从站的 deb 包与 Library 文件包一致；

## 3 . 支持与服务

中科时代为公司产品及解决方案提供全方位支持与服务，确保针对相关问题给予快速且专业的响应。

### **资料下载**

我们的资料下载专区涵盖了丰富的文件资源，包括应用案例、技术文档、产品介绍等，满足您的多样化需求。

资料下载地址：<https://help.sinsegye.com.cn/>

### **获取支持**

如需中科时代产品的本地支持与服务，请随时联系我们。您可以通过访问我们的官方网站 [www.sinsegye.com.cn](http://www.sinsegye.com.cn)，查找中科时代的分公司地址，并获取更多关于中科时代的信息。

此外，您还可以通过以下方式联系我们：

- 热线电话：400-013-2158
- 邮箱地址：[support@sinsegye.com.cn](mailto:support@sinsegye.com.cn)