

BECKHOFF



TwinCAT 3 Function 实验指导手册

ModbusTCP & TCP/IP

Version 1.0

毕孚自动化设备贸易（上海）有限公司

2016年6月



目录

实验一：TwinCAT 3 Function 安装方法介绍	3
实验二：Modbus-TCP Server 使用介绍	11
实验三：Modbus-TCP Client 使用介绍	14
实验四：TCP/IP Client 使用介绍	22
实验五：TCP/IP Server 使用介绍	27

相关软件下载链接：

TwinCAT 3 软件下载路径：

<ftp://ftp.beckhoff.com.cn/TwinCAT3/install/InstallationPackage/>

TF6250 TC3 Modbus TCP Function 下载路径：

<ftp://ftp.beckhoff.com.cn/TwinCAT3/install/Functions/TF6x-Connectivity/TF6250-Modbus-TCP/>

TF6310 TC3 TCP/IP Function 下载路径：

<ftp://ftp.beckhoff.com.cn/TwinCAT3/install/Functions/TF6x-Connectivity/TF6310-TCP-IP/>

Modbus 调试工具下载路径：

<ftp://ftp.beckhoff.com.cn/TwinCAT3/Samples/TF6250-Modbus-TCP/Tools/>

Socket Tool 以太网调试工具下载路径：

<ftp://ftp.beckhoff.com.cn/TwinCAT3/Samples/TF6310-TcpIpServer/Tools/>

实验一：TwinCAT 3 Function 安装方法介绍

一、实验目的：

1. 了解并掌握在 TwinCAT3 环境下安装 Function；
2. 了解并掌握在 TwinCAT3 环境下对相应 Function 的 license 激活。

二、实验器材：

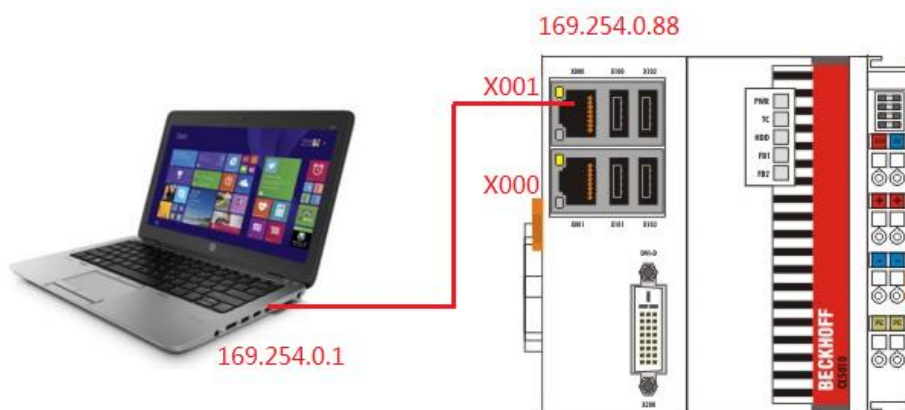
1. 硬件：CX5020-0112（嵌入式 PC），U 盘；
2. 软件：TwinCAT3 软件（Ver. 3.1.4018.26）

CERHOST（CE 系统远程桌面连接工具）

TwinCAT 3 Function 下载路径：

<ftp://ftp.beckhoff.com.cn/TwinCAT3/install/Functions/>

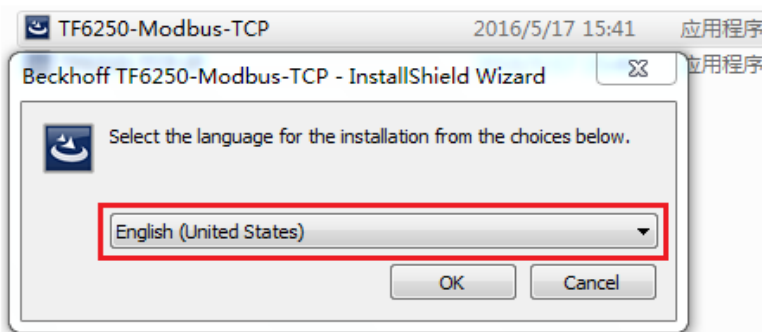
三、实验的系统搭建图



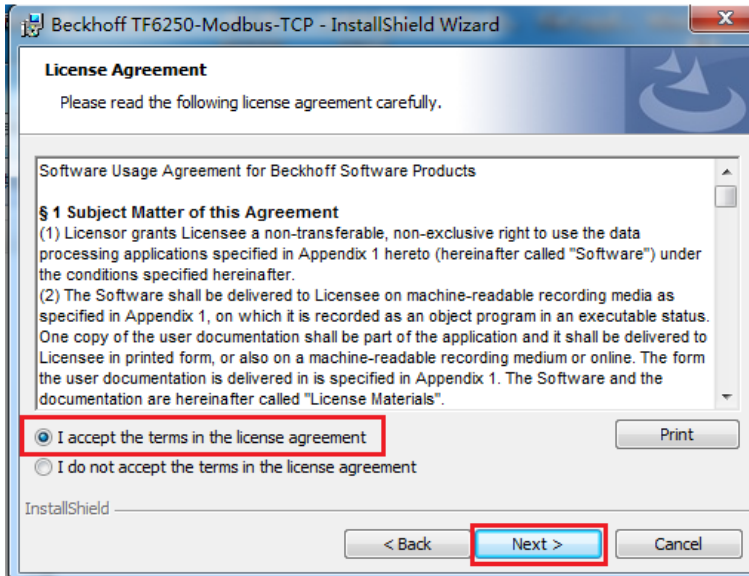
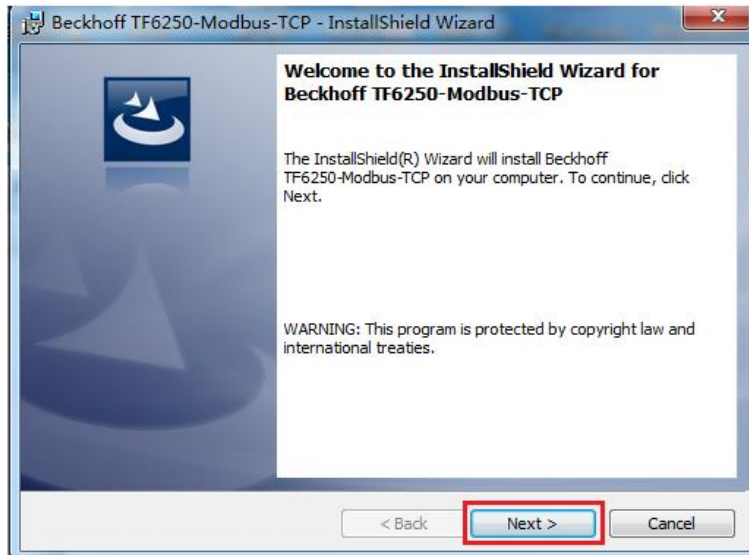
四、实验内容

1. 在 Windows 7 / Windows XP 系统 PC/控制器 Function

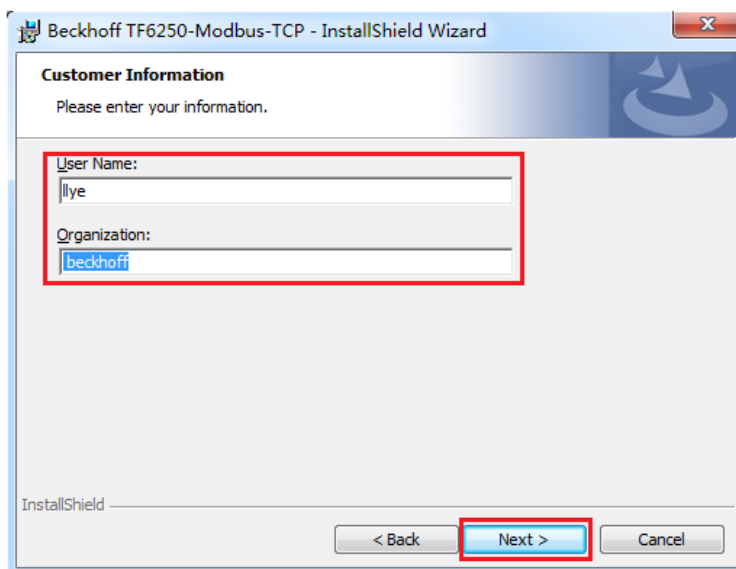
1.1 双击已经下载的安装文件"TFxxxx"。



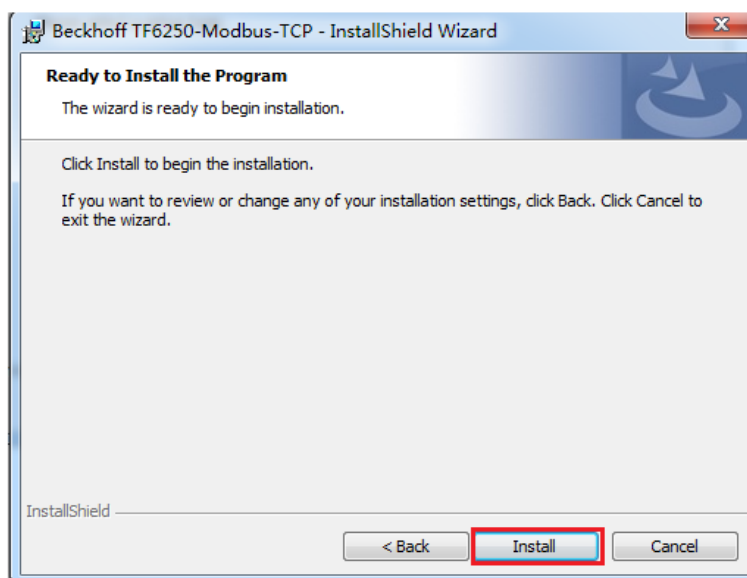
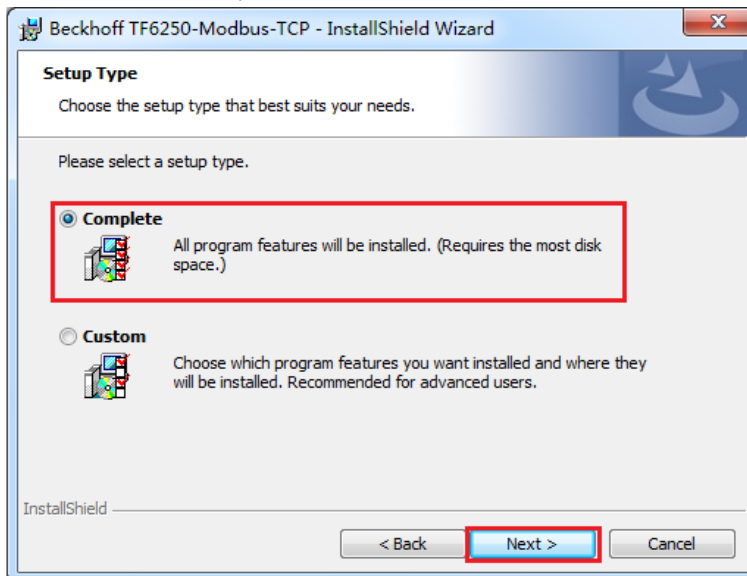
1.2 点击"Next"和选择"I accept the license Agreement"。



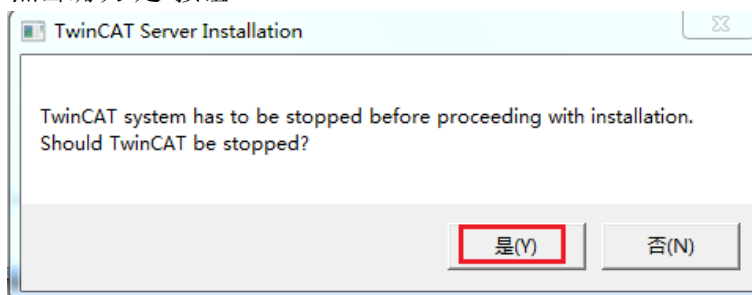
1.3 输入用户信息，点击"Next"。



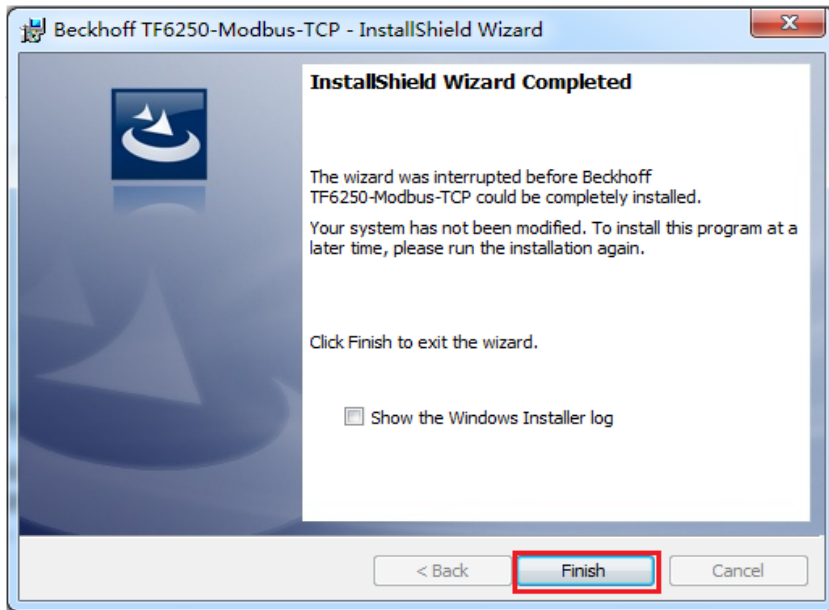
1.4 选择全部安装"Complete", 点击"intall"



1.5 点击确认"是"按钮



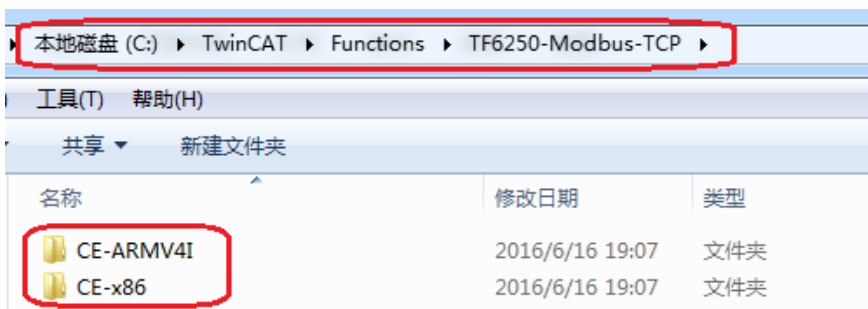
1.6 安装完成, 点击"Finish".



1.7 这样就完成了 WIN7/XP 系统控制器 Function 的安装

2. 对于 CE 系统控制器需要在调试 PC 安装完 Function 后，还需要将相对应的.CAB 文件安装到 CE 设备上。

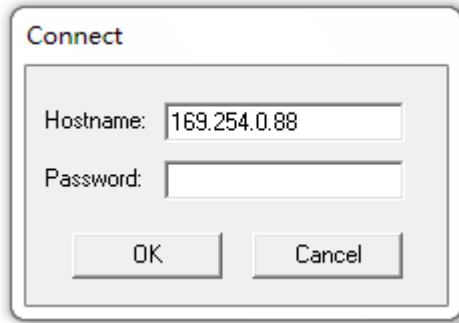
2.1 在 C:\TwinCAT\Functions\TF6250-Modbus-Tcp\CE-86 目录下会生成两个 CAB 文件，*.ARM.CAB 支持 ARM 架构处理器例如：CX8000，CX9020 等系列都是 ARM 处理器，*.I586.CAB 支持 x86 架构处理器例如：CX5000，CX5100，CX2000，C69xx 等是 x86 处理器。



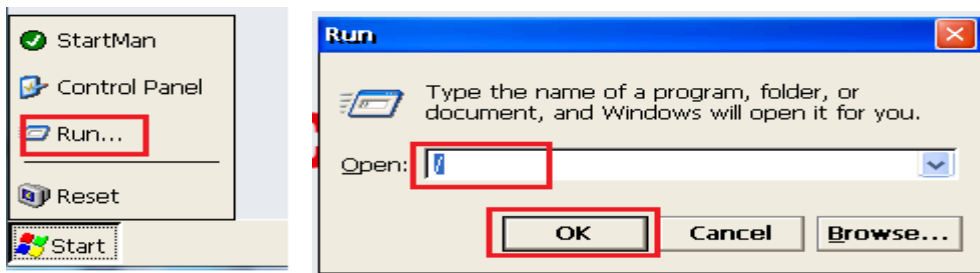
此次实验使用硬件为 CX5020，所以将生成的 CAB 文件 (*.I586.CAB) 拷贝到 U 盘中，连接到 CE 设备上。



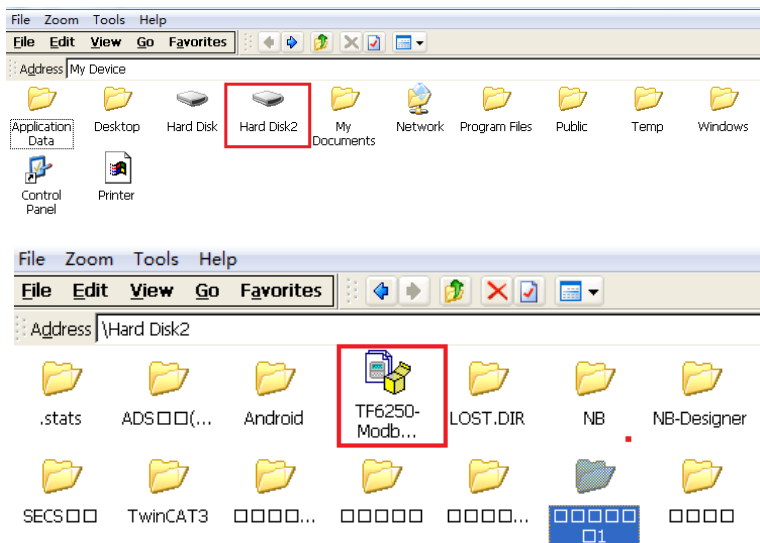
2.2 打开远程桌面,本次样机的网口 1 的 IP 地址是 169.254.0.88。没有 Password。



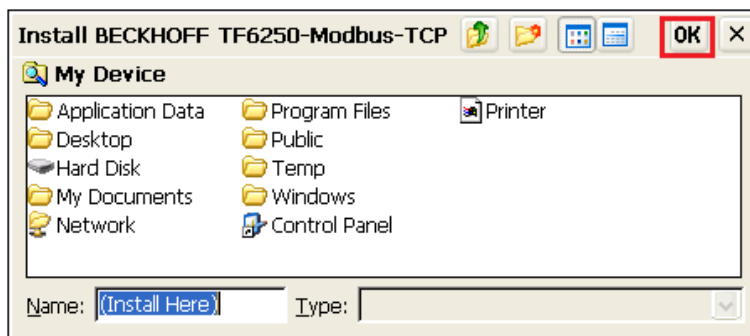
2.3 在 CE 设备中，点击左下角 Start 菜单，选择 RUN，在 OPEN 窗口中输入“/”



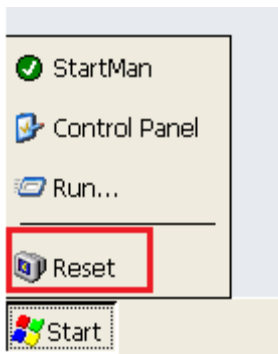
2.4 打开 Hard Disk2 (U 盘) 文件夹，找到 TF6250-Modbus-TCP 双击安装该件；



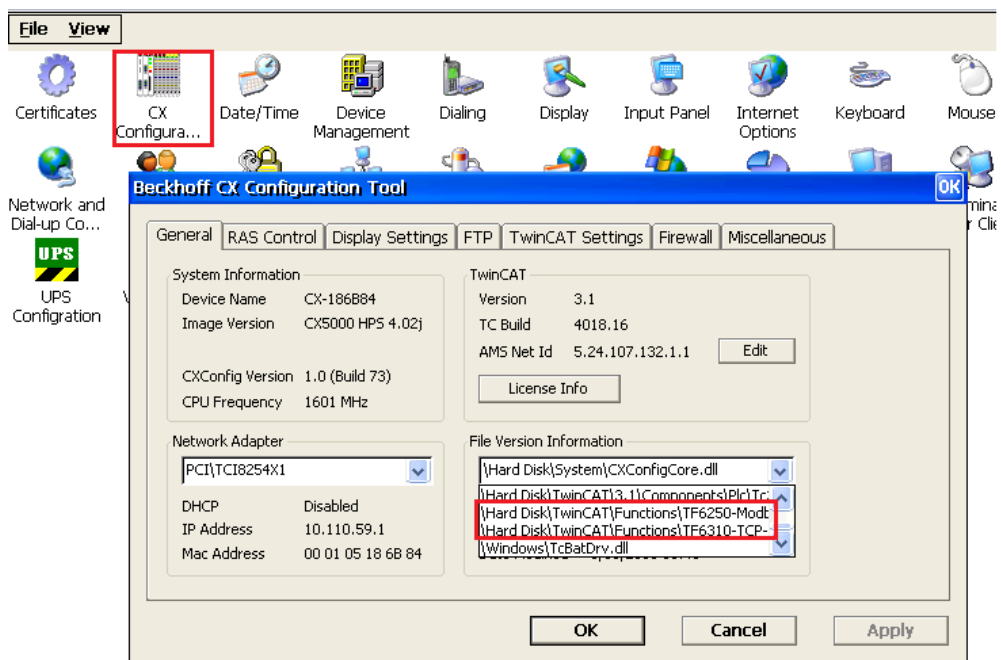
2.5 不要更改安装路径，直接点击 OK



2.5 安装完毕后 Start->Reset



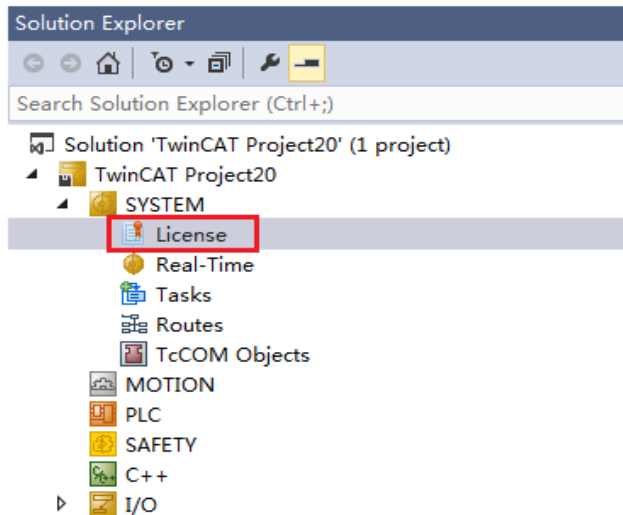
2.6 重启后进入 CE 系统可在 Control Panel->CX Configuration->General 下的 File Version Information 的下拉菜单中看到新装的软件。到此安装软件结束



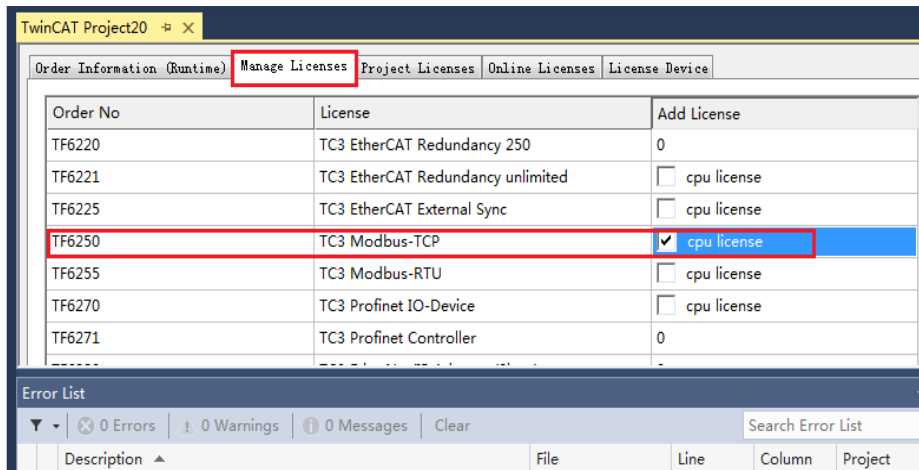
3. 成功安装完对应的 Function 之后，激活相应 Function 的 license。

3.1 打开 TwinCAT3, 新建工程, 添加完路由并成功连上控制后, 双击 SYSTEM 下面的 license。

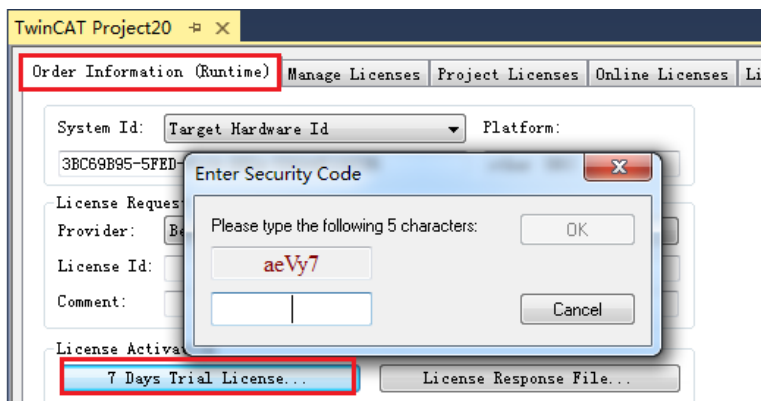




3.2 选择 Manage Licenses, 勾选 TC3 Modbus-TCP 的 licenses



3.3 激活 7 天 licenses, 确认 TC3 Modbus-TCP 的 licenses 已经激活



Order Information (Runtime) | Manage Licenses | Project Licenses | Online Licenses | License Device

System Id: Target Hardware Id Platform: 3BC69B95-5FED-F104-BB5A-55694F3397D5 other (90)

License Request
 Provider: Beckhoff Automation Generate File...
 License Id: Customer
 Comment:

License Activation
 7 Days Trial License... License Response File...

Order No	License	Instances	Current Status
TC1000	TC3 ADS	cpu license	expires on Jun 14, 2016 (t..
TF6250	TC3 Modbus-TCP	cpu license	expires on Jun 14, 2016 (t..

4. 这样就完成了 Function 的安装与授权激活

五、 随堂问答

实验二：Modbus-TCP Server 使用介绍

一、实验目的：

1. 了解并掌握 modscan32 使用方法
2. 了解并掌握 Modbus-TCP Sever 的使用方法。

二、实验器材：

1. 硬件：CX5020-0112（嵌入式 PC）
2. 软件：TF6250-Modbus-TCP 的安装包

modscan32（modbus-TCP Client 调试助手）

TwinCAT3 软件（编程软件）

三、实验的系统搭建图



四、实验内容

(一)实验要求

CX5020 作为 Modbus-TCP Server，modscan32 作为 Modbus-TCP Client，结合 modscan32 对保持寄存器 (CX5020 的 M 区) 进行读写。

(二)实验步骤

1. 安装 TF6250-Modbus-TCP 的 Function, (具体步骤见实验一)，打开 TwinCAT3, 新建工程，激活相应的 license。

License Activation

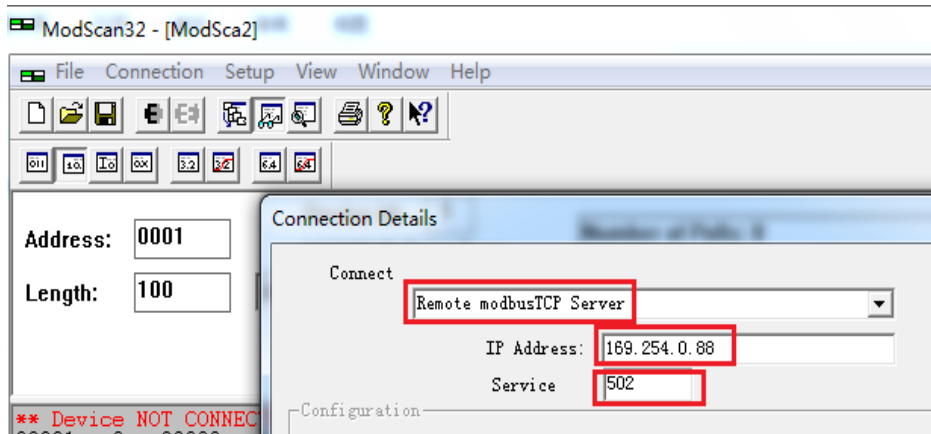
Order No	License	Instances	Current Status
TC1000	TC3 ADS	cpu license	expires on Jun 21, 2016 (tria...
TF6250	TC3 Modbus-TCP	cpu license	expires on Jun 21, 2016 (tria...

2. 建立变量，通过变量“AT%”来进行声明，AT%是关键字，把变量分配给内部地

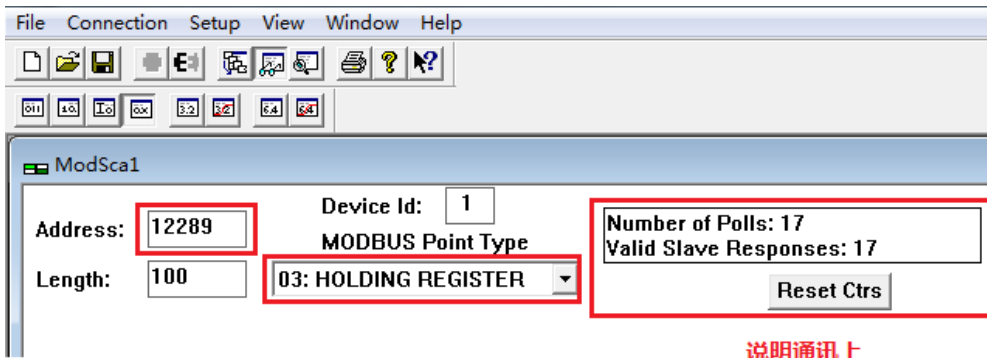
址。

```
PROGRAM MAIN
VAR
arr1 AT %MB0: ARRAY[1..2] OF WORD; (*保持寄存器 起始地址为12289*)
arr2 AT %MB10: ARRAY[1..2] OF BYTE; (*保持寄存器 起始地址为12294*)
END VAR
```

3. 对 modscan32 设置，点击文件—新建，然后点击 connection--connect，确认 Modbus-TCP Sever IP 地址和 Modbus TCP 通讯端口：502。



4. 连接上控制器，激活配置，下载程序，使 Modbus-TCP Sever 处于监听状态。设置 modscan32 的数据类型和起始地址。



5. 如果 Valid Slave Responses 在累加，说明 modscan32 和 CX5020 已成功建立通讯，对 modscan32 的 03: Holding Register 12289, 12290, 12294 分别赋值 16#1111, 16#3333, 16#5678, 在 PLC 程序中便可以看到相应数值。

The screenshot shows the ModSca1 software interface. At the top, the configuration is as follows:

- Address: 12289
- Device Id: 1
- MODBUS Point Type: 03: HOLDING REGISTER
- Number of Polls: 708
- Valid Slave Responses: 708
- Length: 100
- Reset Ctrs button

Below the configuration is a table of Modbus registers:

412289: <1111H>	412309: <0000H>	412329: <0000H>	412349: <0000H>	412369: <0000H>
412290: <3333H>	412310: <0000H>	412330: <0000H>	412350: <0000H>	412370: <0000H>
412291: <0000H>	412311: <0000H>	412331: <0000H>	412351: <0000H>	412371: <0000H>
412292: <0000H>	412312: <0000H>	412332: <0000H>	412352: <0000H>	412372: <0000H>
412293: <0000H>	412313: <0000H>	412333: <0000H>	412353: <0000H>	412373: <0000H>
412294: <5678H>	412314: <0000H>	412334: <0000H>	412354: <0000H>	412374: <0000H>
412295: <0000H>	412315: <0000H>	412335: <0000H>	412355: <0000H>	412375: <0000H>
412296: <0000H>	412316: <0000H>	412336: <0000H>	412356: <0000H>	412376: <0000H>
412297: <0000H>	412317: <0000H>	412337: <0000H>	412357: <0000H>	412377: <0000H>
412298: <0000H>	412318: <0000H>	412338: <0000H>	412358: <0000H>	412378: <0000H>

Below the register table is a table of expressions and values:

Expression	Type	Value	Prepared v...	Address
arr1	ARRAY [1..2] OF WORD			%MB0
arr1[1]	WORD	16#1111		%MB0
arr1[2]	WORD	16#3333		%MB2
arr2	ARRAY [1..2] OF BYTE			%MB10
arr2[1]	BYTE	16#78		%MB10
arr2[2]	BYTE	16#56		%MB11

Red annotations in the image indicate:

- 保持寄存器 起始地址为12289 (Holding register start address is 12289) pointing to the start of the arr1 array.
- 保持寄存器 起始地址为12294 (Holding register start address is 12294) pointing to the start of the arr2 array.

五、 配套 PLC 例程下载链接:

ftp://ftp.beckhoff.com.cn/TwinCAT3/Samples/TF6250-Modbus-TCP/ModbusTCP_SampleCode/ModbusTCP_Server_SampleCodeV1.tzip

六、 随堂问答

Q: TwinCAT 2 中, 如果 PLC 作为 Server 也可以通讯 I 区和 Q 区, 为什么现在无法通讯了?

A: 目前最新 1.0.53.0 不支持 I 区和 Q 区数据的数据通讯, 建议客户开发过程中统一使用 M 区, 避免内存冲突。

Q: x86 32 位的控制器 %MW0 对应定制为 03: Holding Register 12289, 其他类型的控制器是否有差别?

A: 经过测试存在差别, ARM 的控制器 %MW0 对应 12296; x86 64 位的控制器 %MW0 对应 12296。具体对应关系, 建议实际测试。

Q: 资料中举例的 GVL.mb_Input_Registers/ GVL.mb_Output_Registers/ GVL.mb_Input_Coils/ GVL.mb_Output_Coils 是否可以正常使用?

A: 经过测试可以使用, 只需要在全局标量中相同名称的变量, 无需声明地址即可进行通讯。

实验三：Modbus-TCP Client 使用介绍

一、实验目的：

1. 了解并掌握 modsim32 使用方法
2. 了解并掌握 Modbus-TCP 的 Client 的功能块的使用。

二、实验器材：

1. 硬件：CX5020-0112（嵌入式 PC）
2. 软件：TF6250-Modbus-TCP 的安装包

modsim32（modbus-TCP Server 调试助手）

TwinCAT3 软件（编程软件）

三、实验的系统搭建图



四、实验内容

(一)实验要求

CX5020 作为 Modbus-TCP Client，modsim32 作为 Modbus-TCP Server，结合 modsim32，分别对离散量输入，线圈，输入寄存器和保持寄存器的读取，对线圈和保持寄存器写入。

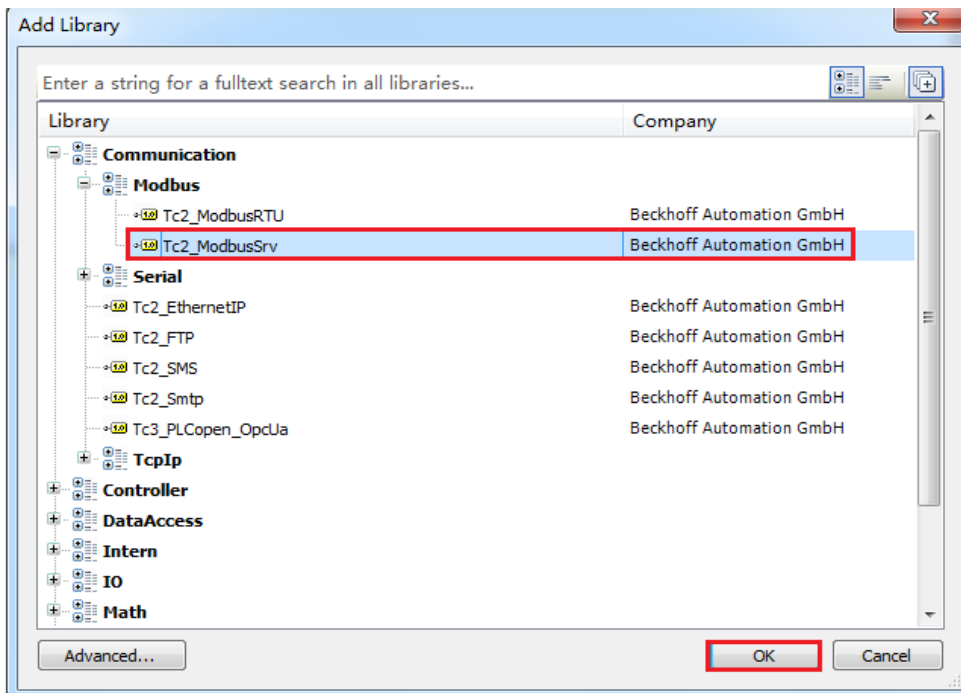
(二)实验步骤

1. 安装 TF6250-Modbus-TCP 的 Function, (具体步骤见实验一)，打开 TwinCAT3，新建工程，激活相应的 license。



Order No	License	Instances	Current Status
TC1000	TC3 ADS	cpu license	expires on Jun 21, 2016 (tria...
TF6250	TC3 Modbus-TCP	cpu license	expires on Jun 21, 2016 (tria...

2. 添加相应的功能库，右键 References, 选择 Add library, 选择 Communication 里面的 Modbus 里面的 Tc2_ModbusSrv。



3. 本次实验共介绍 8 个功能块，分别是 FB_MBReadInputs（读取离散量输入）；FB_MBReadCoils（读取线圈），FB_MBWriteCoils（写入线圈），FB_MBWriteSingleCoil（写入单个线圈）；FB_MBReadInputRegs（读取输入寄存器）；FB_MBReadRegs（读取保持寄存器），FB_MBWriteRegs（写入保持寄存器），FB_MBWriteSingleReg；（写入单个寄存器）。

4. 声明相关功能块和变量
读取离散量输入相关变量

```

////////////////////////////////////02: Input Status 读取////////////////////////////////////
fbReadInputs      : FB_MBReadInputs;          (* 读取离散量输入功能块*)
bReadInputs       : BOOL;                    (* 读取离散量输入执行条件*)
nQuantityinput    : WORD:=1 ;                (* 读取离散量输入个数*)
nMBAddrinput      : WORD:=1 ;                (* 读取离散量输入起始地址*)
arrDatainput      : BYTE;                    (* 存放离散量输入的值*)
////////////////////////////////////

```

读取/写入线圈相关变量

```
////////////////////////////////////01: Coil Status 读取&写入////////////////////////////////////
fbReadCoils      : FB_MBReadCoils;          (*读取线圈功能块*)
bReadCoils      : BOOL;                    (*读取线圈执行条件*)
nQuantitycoils  : WORD :=3;                (*读取线圈个数*)
nMAddrcoils     : WORD :=1;                (*读取线圈起始地址*)
arrDatacoils    : BYTE;                    (*存放线圈的值*)

fbWriteCoils     : FB_MBWriteCoils;         (*写入线圈功能块*)
bWriteCoils     : BOOL;                    (*写入线圈执行条件*)
nQuantityWriteCoils : WORD := 10;         (*写入高数量输入个数*)
nMAddrWriteCoils : WORD := 14;           (*写入高数量输入起始地址*)
arrDataWriteCoils : ARRAY[1..2] OF BYTE := [16#11, 16#33]; (*写入高数量输入的值*)

fbWriteSingleCoil : FB_MBWriteSingleCoil;  (*写入单个线圈功能块*)
bWriteSingleCoil  : BOOL;                  (*写入单个线圈执行条件*)
nMAddrWriteSingleCoil : WORD := 3;        (*写入单个线圈Modbus 地址*)
nValueWriteSingleCoil : WORD := 16#FF00;  (*16#FF00: True; 16#0000: False*)
////////////////////////////////////
```

读取输入寄存器相关变量

```
////////////////////////////////////04: Input Register 读取////////////////////////////////////
fbReadInputRegs  : FB_MBReadInputRegs;     (*读取输入寄存器功能块*)
bReadInputRegs   : BOOL;                   (*读取输入寄存器执行条件*)
nQuantityInputRegs : WORD := 3;           (*读取输入寄存器个数*)
nMAddrInputRegs  : WORD:= 2;              (*读取输入寄存器起始地址*)
arrDataInputRegs : ARRAY [1..3] OF WORD;   (*存放输入寄存器的值*)
////////////////////////////////////
```

读取/写入保持寄存器相关变量

```
////////////////////////////////////03: Holding Register 读取&写入////////////////////////////////////
fbReadRegs      : FB_MBReadRegs;          (*读取保持寄存器功能块*)
bReadRegs      : BOOL;                    (*读取保持寄存器执行条件*)
nQuantityregs   : WORD:=2;                (*读取保持寄存器个数*)
nMAddrregs     : WORD:=24;                (*读取保持寄存器起始地址*)
arrDataregs    : ARRAY [1..2] OF WORD;    (*存放保持寄存器的值*)

fbWriteRegs     : FB_MBWriteRegs;         (*写入保持寄存器功能块*)
bWriteRegs     : BOOL;                    (*写入保持寄存器个数*)
nQuantityWriteRegs : WORD := 4;          (*写入保持寄存器个数*)
nMAddrWriteRegs : WORD := 4;             (*写入保持寄存器起始地址*)
arrDataWriteRegs : ARRAY[1..4] OF WORD := [1122, 3344, 5566, 7788]; (*写入保持寄存器的值*)

fbWriteSingleReg : FB_MBWriteSingleReg;   (*写入单个寄存器功能块*)
bWriteSingleReg  : BOOL;                  (*写入单个寄存器执行条件*)
nMAddrSingleReg  : WORD := 4;             (*写入单个寄存器Modbus 地址*)
nValueSingleReg  : WORD := 16#1234;      (*写入单个寄存器数值*)
////////////////////////////////////
```

5. 调用相关功能块及输入填写

FB_MBReadInputs (读取离散量输入):

```
fbReadInputs (  
    sIPAddr:= '169.254.0.1', //modsim32的IP地址  
    nTCPPort:=100, //Modbus-Tcp端口号  
    nUnitID:=1, //Modbus-Tcp从站号  
    nQuantity:=nQuantityinput, //读取离散量输入个数  
    nMBAAddr:= nMBAAddrinput, //读取离散量输入 Modbus起始地址  
    cbLength:= SIZEOF(arrDatainput), //存放离散量输入变量的个数  
    pDestAddr:=ADR(arrDatainput), //存放离散量输入变量指针起始地址  
    bExecute:=bReadInputs, //读取离散量输入执行条件  
    tTimeout:=T#1S,  
    bBusy=>,   
    bError=>,   
    nErrId=>,   
    cbRead=> );
```

FB_MBReadCoils (读取线圈):

```
fbReadCoils (  
    sIPAddr:='169.254.0.1', //modsim32的IP地址  
    nTCPPort:=100, //Modbus-Tcp端口号  
    nUnitID:=1, //Modbus-Tcp从站号  
    nQuantity:=nQuantitycoils, //读取线圈个数  
    nMBAAddr:=nMBAAddrcoils, //读取线圈 Modbus起始地址  
    cbLength:=SIZEOF(arrDatacoils), //存放线圈变量的个数  
    pDestAddr:=ADR(arrDatacoils), //存放线圈变量指针起始地址  
    bExecute:=bReadCoils, //读取线圈执行条件  
    tTimeout:= T#1S,  
    bBusy=>,   
    bError=>,   
    nErrId=>,   
    cbRead=> );
```

FB_MBWriteCoils (写入线圈):

```
fbWriteCoils (  
    sIPAddr:= '169.254.0.1', //modsim32的IP地址  
    nTCPPort:=100, //Modbus-Tcp端口号  
    nUnitID:=1, //Modbus-Tcp从站号  
    nQuantity:= nQuantityWriteCoils, //写入线圈个数  
    nMBAAddr:=nMBAAddrWriteCoils, //写入线圈Modbus起始地址  
    cbLength:=SIZEOF(arrDataWriteCoils), //写入线圈的变量个数  
    pSrcAddr:=ADR(arrDataWriteCoils), //写入线圈的变量指针起始地址  
    bExecute:=bWriteCoils, //写入线圈的执行条件  
    tTimeout:=T#1S,  
    bBusy=>,   
    bError=>,   
    nErrId=> );
```

FB_MBWriteSingleCoil(写入单个线圈):

```
fbWriteSingleCoil(  
    sIPAddr:= '169.254.0.1',           //modsim32的IP地址  
    nTCPPort:= 100,                   //Modbus-Tcp端口号  
    nUnitID:= 1,                      //Modbus-Tcp从站号  
    nMBAAddr:=nMBAAddrWriteSingleCoil , //写入单个线圈Modbus起始地址  
    nValue:=nValueWriteSingleCoil ,   //写入单个线圈的值:16#FF00: True; 16#0000: False  
    bExecute:=bWriteSingleCoil ,     //写入单个线圈执行条件  
    tTimeout:=T#1S ,  
    bBusy=> ,  
    bError=> ,  
    nErrId=> );
```

FB_MBReadInputRegs (读取输入寄存器):

```
fbReadInputRegs (  
    sIPAddr:='169.254.0.1' ,           //modsim32的IP地址  
    nTCPPort:=100,                   //Modbus-Tcp端口号  
    nUnitID:=1,                     //Modbus-Tcp从站号  
    nQuantity:=nQuantityInputRegs,   //读取输入寄存器个数  
    nMBAAddr:=nMBAAddrInputRegs ,    //读取输入寄存器Modbus起始地址  
    cbLength:= sizeof(arrDataInputRegs), //存放输入寄存器变量的个数和指针起始地址  
    pDestAddr:=ADR(arrDataInputRegs), //存放输入寄存器变量指针起始地址  
    bExecute:= bReadInputRegs ,      //读取输入寄存器执行条件  
    tTimeout:=T#1S ,  
    bBusy=> ,  
    bError=> ,  
    nErrId=> ,  
    cbRead=> );
```

FB_MBReadRegs (读取保持寄存器):

```
fbReadRegs (  
    sIPAddr:='169.254.0.1' ,           //modsim32的IP地址  
    nTCPPort:=100,                   //Modbus-Tcp端口号  
    nUnitID:= 1,                     //Modbus-Tcp从站号  
    nQuantity:=nQuantityregs,        //读取保持寄存器个数  
    nMBAAddr:=nMBAAddrregs ,         //读取保持寄存器Modbus起始地址  
    cbLength:=sizeof(arrDataregs) ,  //存放保持寄存器变量的个数  
    pDestAddr:=ADR(arrDataregs) ,    //存放保持寄存器变量指针起始地址  
    bExecute:=bReadRegs,             //读取保持寄存器执行条件  
    tTimeout:= T#1S ,  
    bBusy=> ,  
    bError=> ,  
    nErrId=> ,  
    cbRead=> );
```

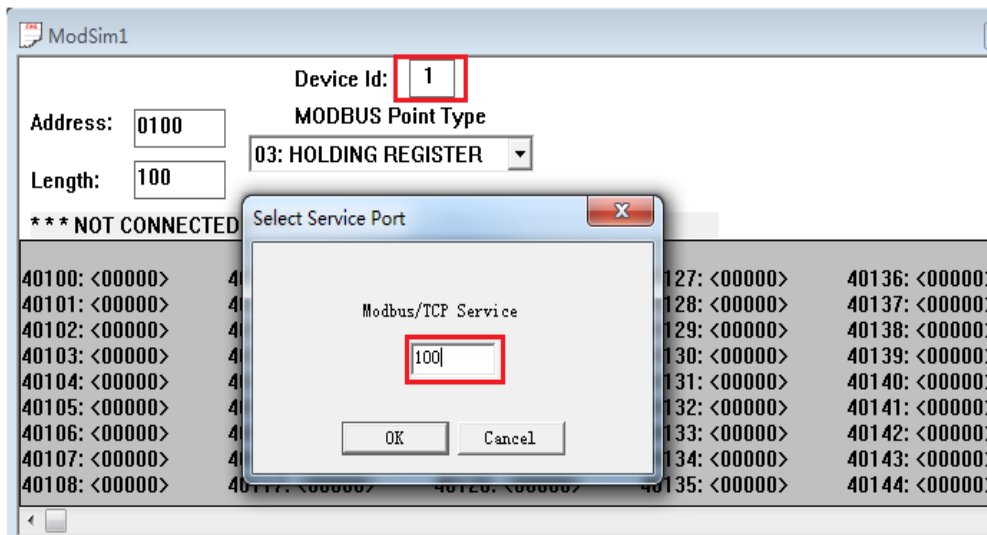
FB_MBWriteRegs (写入保持寄存器):

```
fbWriteRegs(  
    sIPAddr:='169.254.0.1' , //modsim32的IP地址  
    nTCPPort:=100, //Modbus-Tcp端口号  
    nUnitID:=1 , //Modbus-Tcp从站号  
    nQuantity:=nQuantityWriteRegs , //写入保持寄存器个数  
    nMBAAddr:= nMBAAddrWriteRegs , //写入保持寄存器起始地址  
    cbLength:= SIZEOF(arrDataWriteRegs), //写入变量的个数和指针起始地址  
    pSrcAddr:=ADR(arrDataWriteRegs) , //写入变量指针起始地址  
    bExecute:= bWriteRegs , //写入保持寄存器的执行条件  
    tTimeout:=T#1S ,  
    bBusy=> ,  
    bError=> ,  
    nErrId=> );
```

FB_MBWriteSingleReg (写入单个寄存器):

```
fbWriteSingleReg(  
    sIPAddr:='169.254.0.1' , //modsim32的IP地址  
    nTCPPort:=100, //Modbus-Tcp端口号  
    nUnitID:=1 , //Modbus-Tcp从站号  
    nMBAAddr:=nMBAAddrSingleReg, //写入单个保持寄存器起始地址  
    nValue:=nValueSingleReg, //写入单个寄存器数值  
    bExecute:=bWriteSingleReg , //写入单个寄存器的执行条件  
    tTimeout:=T#1S ,  
    bBusy=> ,  
    bError=> ,  
    nErrId=> );
```

6. 对 modsim32 设置，点击文件—新建，然后点击连接----Modbus/Tcp Svr，确认端口号是 100，从站号是 1。



7. 连接上控制器，激活配置，下载程序，已经建立连接。

把 fbReadInputs 功能块的 bReadInputs 变为 True 之后,可以读取 modsim32 的 02: INPUT STATUS 0002

Device Id: 1
 Address: 0001
 Length: 100
 MODBUS Point Type: 02: INPUT STATUS

10001: <0>	10022: <0>	10043: <0>	10064: <0>	10085: <0>
10002: <1>	10023: <0>	10044: <0>	10065: <0>	10086: <0>
10003: <0>	10024: <0>	10045: <0>	10066: <0>	10087: <0>
10004: <0>	10025: <0>	10046: <0>	10067: <0>	10088: <0>
10005: <0>	10026: <0>	10047: <0>	10068: <0>	10089: <0>
10006: <0>	10027: <0>	10048: <0>	10069: <0>	10090: <0>
10007: <0>	10028: <0>	10049: <0>	10070: <0>	10091: <0>
10008: <0>	10029: <0>	10050: <0>	10071: <0>	10092: <0>

fbReadInputs	FB_MBReadInputs	
bReadInputs	BOOL	TRUE
nQuantityinput	WORD	1
nMBAaddrinput	WORD	1
arrDatainput	BYTE	1

8. 把 fbReadCoils 功能块的 bReadCoils 变为 True 之后,可以读取 modsim32 的 01: COIL STATUS 0002/0003/0004

Device Id: 1
 Address: 0001
 Length: 100
 MODBUS Point Type: 01: COIL STATUS

00001: <0>	00022: <0>	00043: <0>	00064: <0>	00085: <0>
00002: <1>	00023: <0>	00044: <0>	00065: <0>	00086: <0>
00003: <1>	00024: <0>	00045: <0>	00066: <0>	00087: <0>
00004: <1>	00025: <0>	00046: <0>	00067: <0>	00088: <0>
00005: <0>	00026: <0>	00047: <0>	00068: <0>	00089: <0>
00006: <0>	00027: <0>	00048: <0>	00069: <0>	00090: <0>
00007: <0>	00028: <0>	00049: <0>	00070: <0>	00091: <0>

fbReadCoils	FB_MBReadCoils	
bReadCoils	BOOL	TRUE
nQuantitycoils	WORD	3
nMBAaddrcoils	WORD	1
arrDatacoils	BYTE	7

9. 把 fbReadInputRegs 功能块的 bReadInputRegs 变为 True 之后,可以读取 modsim32 的 04: INPUT REGISTER 0003/0004/0005

Device Id: 1
 Address: 0001
 Length: 100
 MODBUS Point Type: 04: INPUT REGISTER

30001: <00000>	30022: <00000>	30043: <00000>
30002: <00000>	30023: <00000>	30044: <00000>
30003: <01111>	30024: <00000>	30045: <00000>
30004: <02222>	30025: <00000>	30046: <00000>
30005: <03333>	30026: <00000>	30047: <00000>
30006: <00000>	30027: <00000>	30048: <00000>
30007: <00000>	30028: <00000>	30049: <00000>

fbReadInputRegs	FB_MBReadInputRegs	
bReadInputRegs	BOOL	TRUE
nQuantityInputRegs	WORD	3
nMBAAddrInputRegs	WORD	2
arrDataInputRegs	ARRAY [1..3] OF WORD	
arrDataInputRegs[1]	WORD	1111
arrDataInputRegs[2]	WORD	2222
arrDataInputRegs[3]	WORD	3333

10. 把 fbReadRegs 功能块的 bReadRegs 变为 True 之后,可以读取 modsim32 的 03: HOLDING REGISTER 0025/0026

Device Id: 1
 Address: 0001
 Length: 100
 MODBUS Point Type: 03: HOLDING REGISTER

40001: <00000>	40022: <00000>	40043: <00000>
40002: <00000>	40023: <00000>	40044: <00000>
40003: <00000>	40024: <00000>	40045: <00000>
40004: <00000>	40025: <08888>	40046: <00000>
40005: <00000>	40026: <09999>	40047: <00000>
40006: <00000>	40027: <00000>	40048: <00000>

fbReadRegs	FB_MBReadRegs	
bReadRegs	BOOL	TRUE
nQuantityregs	WORD	2
nMBAaddrregs	WORD	24
arrDataregs	ARRAY [1..2] OF WORD	
arrDataregs[1]	WORD	8888
arrDataregs[2]	WORD	9999

11. 把 fbWriteCoils 功能块的 bWriteCoils 变为 True 之后，可以写入 modsim32 的 01:COIL STATUS 0015~0025

Parameter	Type	Value
fbWriteCoils	FB_MBWriteCoils	
bWriteCoils	BOOL	TRUE
nQuantityWriteCoils	WORD	2#0000000000001010
nMBAAddrWriteCoils	WORD	2#0000000000001110
arrDataWriteCoils	ARRAY [1..2] OF BYTE	
arrDataWriteCoils[1]	BYTE	2#00010001
arrDataWriteCoils[2]	BYTE	2#00110011

写入的值是2#01100010001

Address: 0001 MODBUS Point Type: 01: COIL STATUS

Length: 100

00001: <0>	00022: <0>	00043: <0>	00064: <0>
00002: <0>	00023: <1>	00044: <0>	00065: <0>
00003: <0>	00024: <1>	00045: <0>	00066: <0>
00004: <0>	00025: <0>	00046: <0>	00067: <0>
00005: <0>	00026: <0>	00047: <0>	00068: <0>
00006: <0>	00027: <0>	00048: <0>	00069: <0>
00007: <0>	00028: <0>	00049: <0>	00070: <0>
00008: <0>	00029: <0>	00050: <0>	00071: <0>
00009: <0>	00030: <0>	00051: <0>	00072: <0>
00010: <0>	00031: <0>	00052: <0>	00073: <0>
00011: <0>	00032: <0>	00053: <0>	00074: <0>
00012: <0>	00033: <0>	00054: <0>	00075: <0>
00013: <0>	00034: <0>	00055: <0>	00076: <0>
00014: <0>	00035: <0>	00056: <0>	00077: <0>
00015: <1>	00036: <0>	00057: <0>	00078: <0>
00016: <0>	00037: <0>	00058: <0>	00079: <0>
00017: <0>	00038: <0>	00059: <0>	00080: <0>
00018: <0>	00039: <0>	00060: <0>	00081: <0>
00019: <1>	00040: <0>	00061: <0>	00082: <0>
00020: <0>	00041: <0>	00062: <0>	00083: <0>
00021: <0>	00042: <0>	00063: <0>	00084: <0>

12. 把 fbWriteRegs 功能块的 bWriteRegs 变为 True 之后，可以写入 modsim32 的 03:HOLDING REGISTER 0005/0006/0007/0008。注意是低位在前。

Parameter	Type	Value
fbWriteRegs	FB_MBWriteRegs	
bWriteRegs	BOOL	TRUE
nQuantityWriteRegs	WORD	4
nMBAAddrWriteRegs	WORD	4
arrDataWriteRegs	ARRAY [1..4] OF WORD	
arrDataWriteRegs[1]	WORD	1122
arrDataWriteRegs[2]	WORD	3344
arrDataWriteRegs[3]	WORD	5566
arrDataWriteRegs[4]	WORD	7788

Device Id: 1

Address: 0001 MODBUS Point Type: 03: HOLDING REGISTER

Length: 100

40001: <00000>	40022: <00000>	40043: <00000>
40002: <00000>	40023: <00000>	40044: <00000>
40003: <00000>	40024: <00000>	40045: <00000>
40004: <00000>	40025: <00000>	40046: <00000>
40005: <01122>	40026: <00000>	40047: <00000>
40006: <03344>	40027: <00000>	40048: <00000>
40007: <05566>	40028: <00000>	40049: <00000>
40008: <07788>	40029: <00000>	40050: <00000>
40009: <00000>	40030: <00000>	40051: <00000>

五、 配套 PLC 例程下载链接:

ftp://ftp.beckhoff.com.cn/TwinCAT3/Samples/TF6250-Modbus-TCP/ModbusTCP_SampleCode/ModbusTCP_Client_SampleCodeV1.tpz

六、 随堂问答

Q: Modbus TCP 默认通讯端口是 502，为什么本示例中均采用了通讯端口 100?

A: 由于测试电脑侧已经安装了 ModbusTCP Server，502 端口已被占用，所以使用了非 502 通讯端口进行 Modbus TCP 通讯测试

实验四：TCP/IP Client 使用介绍

一、实验目的：

1. 了解并掌握 Socket Tool 的使用方法。
2. 了解并掌握 TCP/IP 的 Client 的功能块的使用。

二、实验器材：

1. 硬件：CX5020-0112（嵌入式 PC）
2. 软件：TF6310-TCP-IP 的安装包

Socket Tool（以太网调试助手）

TwinCAT3 软件（编程软件）

三、实验的系统搭建图



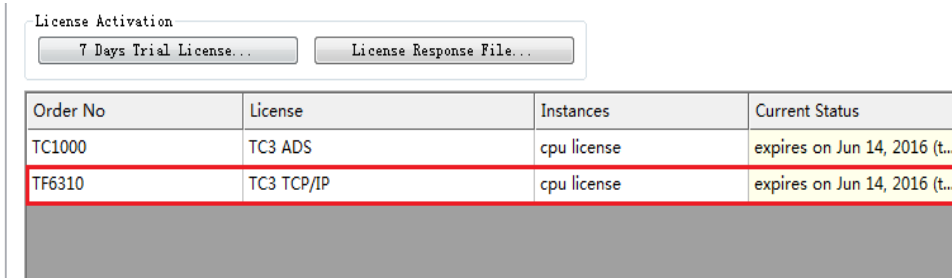
四、实验内容

(一)实验要求

CX5020 作为 Client，Socket Tool 作为 Server，结合 Socket Tool，把 Socket Tool 上面数据发给控制器 CX5020 数组内存 arrRecvData，把 CX5020 数组内存 arrSendData 发送给 Socket Tool。

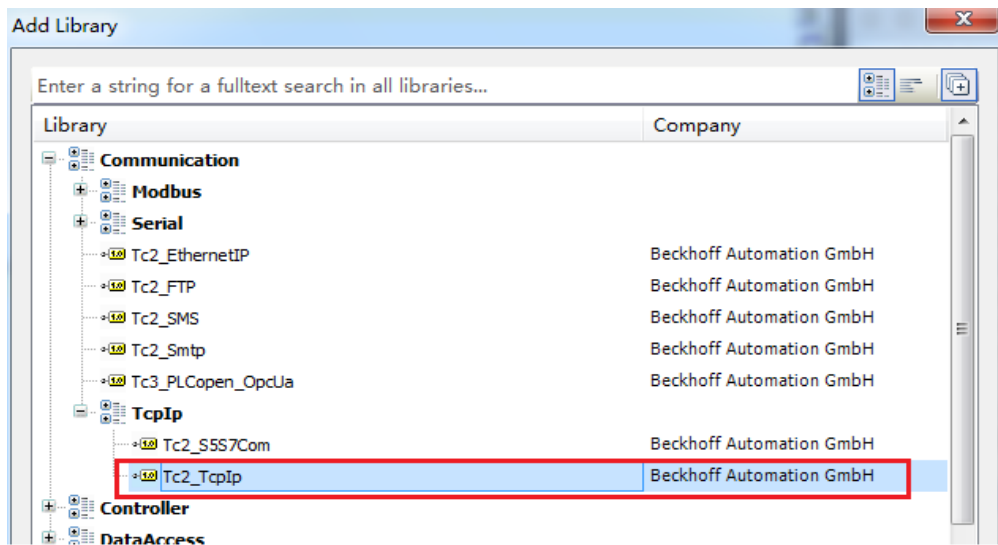
(二)实验步骤

1. 安装 TCP/IP 的 Function, (具体步骤见实验一)，打开 TwinCAT3，新建工程，激活相应的 license。



2. 添加相应的功能库，右键 References, 选择 Add library, 选择 Communication 里

面的 TC2_TCPIP。



3. 本次实验需要用到 4 个功能块，分别是 FB_SocketConnect，FB_SocketSend，FB_SocketReceive，FB_SocketClose，建立相关的变量。

Scope	Name	Address	Data type	Initialization	Comment
VAR	fbSocketConnect		FB_SocketConnect		TCP/IP建立连接的功能块
VAR	fbSocketSend		FB_SocketSend		TCP/IP的发送功能块
VAR	fbSocketReceive		FB_SocketReceive		TCP/IP的写入的功能块
VAR	fbSocketClose		FB_SocketClose		TCP/IP的关闭建立连接的功能块
VAR	bExecute		BOOL		TCP/IP建立连接的执行位
VAR	hSocket		T_HSOCKET		TCP/IP的句柄
VAR	arrSendData		ARRAY[1..4] OF WORD	[16#1111, 16#2222, 16#3333, 16#4444]	TCP/IP发送数组
VAR	bSend		BOOL		TCP/IP发送的执行位
VAR	bRecieve		BOOL		TCP/IP接收的执行位
VAR	arrRecieveData		ARRAY[1..4] OF WORD		TCP/IP接收数组
VAR	bClose		BOOL		TCP/IP关闭连接的执行位

4. 编写实验案例

第一步，编写 TCP/IP 连接功能块，确认 TCP/IP Server 的 IP 地址为 169.254.0.1 和端口号 6000，并将 fbSocketConnect 获取到 hSocket 与声明的 hSocket 进行绑定

```
fbSocketConnect (
  sSrvNetId:= ,
  sRemoteHost:='169.254.0.1', //TCP/IP Server的IP地址169.254.0.1
  nRemotePort:=6000 , //TCP/IP Server的I端口号6000
  bExecute:= bExecute, //TCP/IP连接的执行位*)
  tTimeout:=T#1S ,
  bBusy=> ,
  bError=> ,
  nErrId=> ,
  hSocket=>hSocket ); //通过SocketConnect获取句柄，供后续FB使用
```

第二步，编写 TCP/IP 发送功能块。

```

fbSocketSend(
    sSrvNetId:= ,
    hSocket:=hSocket , //TCP/IP的句柄
    cbLen:=SIZEOF(arrSendData) , //TCP/IP发送数据的长度
    pSrc:=ADR (arrSendData), //TCP/IP发送数据的指针起始地址
    bExecute:=bSend, //TCP/IP发送的执行位
    tTimeout:= T#1S,
    bBusy=> ,
    bError=> ,
    nErrId=> );

```

第三步，编写 TCP/IP 接收功能块

```

fbSocketReceive(
    sSrvNetId:= ,
    hSocket:=hSocket , //TCP/IP的句柄
    cbLen:=SIZEOF(arrRecieveData) , //TCP/IP接收数据的长度
    pDest:=ADR (arrRecieveData) , //TCP/IP接收数据的指针起始地址
    bExecute:=bRecieve, //TCP/IP接收的执行位
    tTimeout:= T#1S ,
    bBusy=> ,
    bError=> ,
    nErrId=> ,
    nRecBytes=> );

```

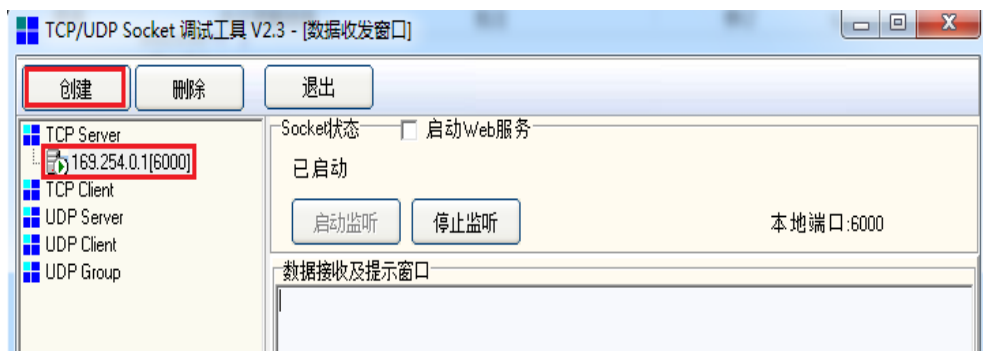
第四步，编写 TCP/IP 关闭功能块

```

fbSocketClose(
    sSrvNetId:= ,
    hSocket:=hSocket , //TCP/IP的句柄
    bExecute:=bClose , //TCP/IP关闭连接执行位
    tTimeout:=T#1S ,
    bBusy=> ,
    bError=> ,
    nErrId=> );

```

5. 对 Socket Tool 设置，选中 TCP Server,点击创建，设置监听端口号为 6000。



6. 连接上控制器，激活配置，下载程序，把 SocketConnect 功能块触发位 bExecute 变为 True 之后，便建立了 TCP/IP 连接。可以看见 hSocket 有值，显示 Server 和 Client 的 IP 地址与通讯端口。其中 Local 指 CX 控制器，Remote 指 PC 侧的 Socket Tool

```
fbSocketConnect (
  sSrvNetId:= ,
  sRemoteHost:= '169.254.0.1',
  nRemotePort:= 6000,
  bExecute:= bExecute TRUE,
  tTimeout:= T#1S,
  bBusy=> ,
  bError=> ,
  nErrId=> ,
  hSocket=> hSocket );
```

hSocket	T_HSOCKET	
handle	UDINT	16#00010001
localAddr	ST_SockAddr	
nPort	UDINT	16#0000C000
sAddr	STRING(15)	'169.254.0.88'
remoteA...	ST_SockAddr	
nPort	UDINT	16#00001770
sAddr	STRING(15)	'169.254.0.1'

Socket状态: 已连接

对方IP: 169.254.0.88 对方端口: 49152

本地端口: 6000

数据接收及提示窗口

7. Socket Tool 写入发送数据 16#1111,16#2222,16#3333,16#4444,然后导通 bReceive 接收数据，在数组 arrrecieveData 接收到 Socket Tool 发送的数据

数据发送窗口 [HEX模式]

1111222233334444

发送数据

重复发送次数: 1

收: 0字节, 发: 0字节

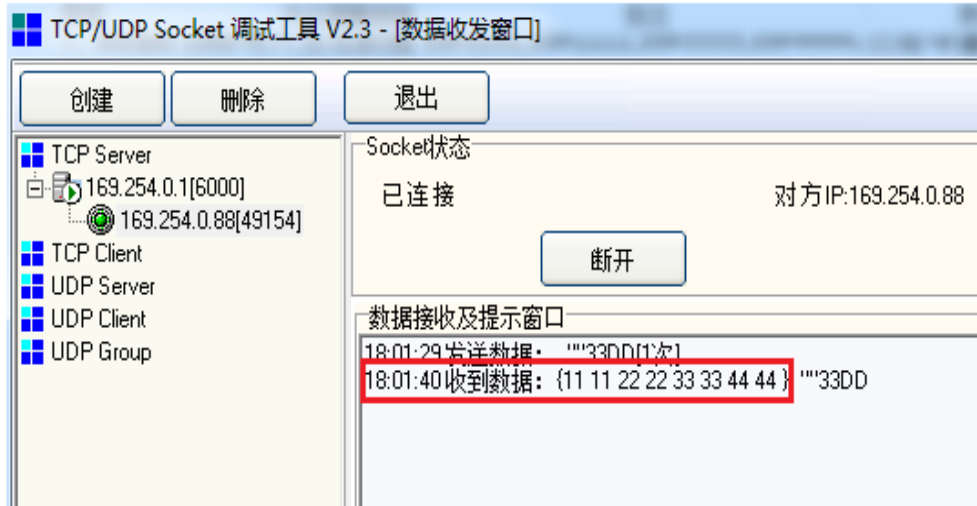
显示十六进制值

统计清零

arrrecieveData	ARRAY [1..4] OF WO...		建立 TCP/IP接收数组
arrreciev...	WORD	16#1111	
arrreciev...	WORD	16#2222	
arrreciev...	WORD	16#3333	
arrreciev...	WORD	16#4444	

8. 接收测试完成之后，导通 bSend，并且把数组 arrsendData，发送给 Socket Tool。

arrsendData	ARRAY [1..4] OF WO...			建立TCP/IP发送数组
arrsend...	WORD	16#1111		
arrsend...	WORD	16#2222		
arrsend...	WORD	16#3333		
arrsend...	WORD	16#4444		



9. 如果不使用 TCP/IP 通讯，建议关掉先前使用的 TCP/IP 通讯端口。

```

fbSocketClose (
  sSrvNetId:= ,
  hSocket:=hSocket ,
  bExecute TRUE :=bclose TRUE ,
  tTimeout T#1s :=T#1S ,
  bBusy=> ,
  bError=> ,
  nErrId=> );RETURN

```

五、 配套 PLC 例程下载链接:

ftp://ftp.beckhoff.com.cn/TwinCAT3/Samples/TF6310-TcpIpServer/TCPIP_SampleCode/TCPCP_Client_SampleCodeV1.tzip

六、 随堂问答

实验五：TCP/IP Server 使用介绍

一、实验目的：

1. 了解并掌握 Socket Tool 的使用方法。
2. 了解并掌握 TCP/IP 的 Server 的功能块的使用。

二、实验器材：

1. 硬件：CX5020-0112（嵌入式 PC）
2. 软件：TF6310-TCP-IP 的安装包
Socket Tool（以太网调试助手）
TwinCAT3 软件（编程软件）

三、实验的系统搭建图

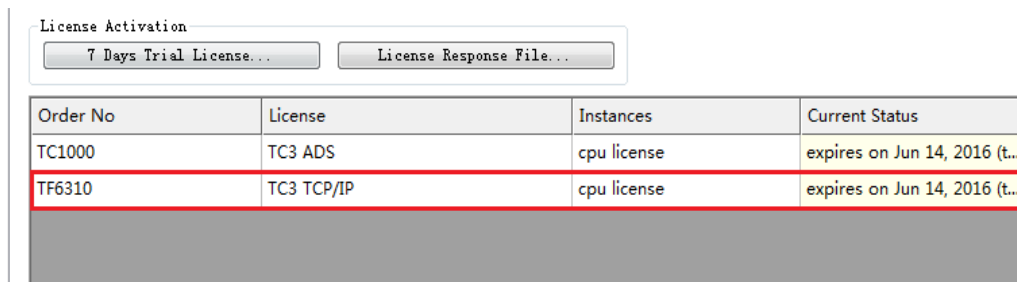


四、实验内容

(一)实验要求

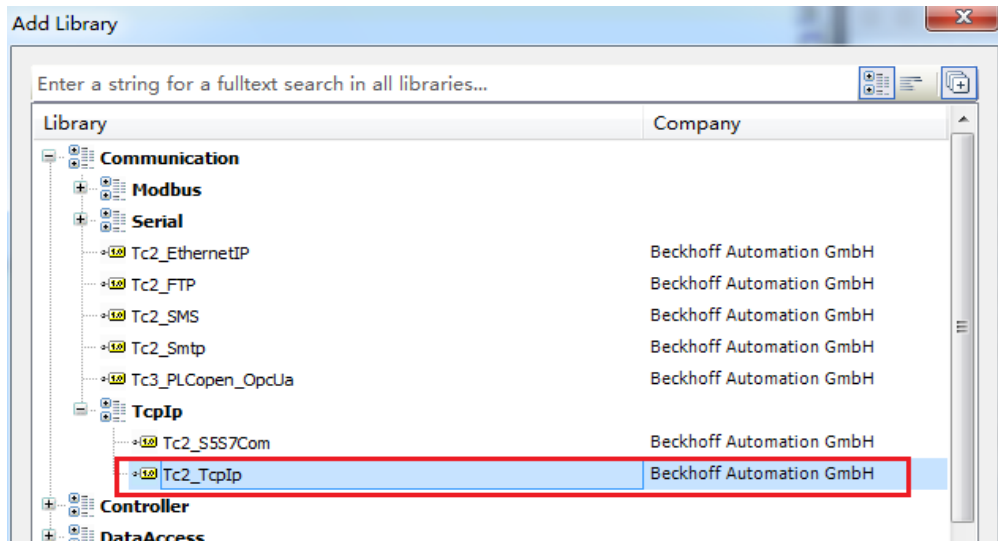
CX5020 作为 Server，Socket Tool 作为 Client，结合 Socket Tool，把 Socket Tool 上面数据发给控制器 CX5020 数组内存 arrReciveData，把 CX5020 数组内存 arrSendData 发送给 Socket Tool。

(二)实验步骤



1. 安装 TCP/IP 的 Function, (具体步骤见实验一)，打开 TwinCAT3，新建工程，激活相应的 license。

2. 添加相应的功能库，右键 References, 选择 Add library, 选择 Communication 里面的 TC2_TCPIP。



3. 本次实验需要用到 5 个功能块，分别是 FB_SocketListen, FB_SocketAccept, FB_SocketSend, FB_SocketReceive, FB_SocketClose，建立相关的变量。

Scope	Name	Address	Data type	Initialization	Comment
VAR	fbSocketListen		FB_SocketListen		TCP/IP Sever建立监听功能块
VAR	fbSocketAccept		FB_SocketAccept		TCP/IP Servershou功能块
VAR	fbSocketSend		FB_SocketSend		TCP/IP的发送功能块
VAR	fbSocketReceive		FB_SocketReceive		TCP/IP的写入的功能块
VAR	fbSocketClose		FB_SocketClose		TCP/IP的关闭建立连接的功能块
VAR	bListen		BOOL		TCP/IP监听端口开启的执行位
VAR	hListener		T_HSOCKET		TCP/IP的监听句柄
VAR	bAccept		BOOL		TCP/IP接受请求的执行位
VAR	hSocket		T_HSOCKET		TCP/IP的句柄
VAR	bSend		BOOL		TCP/IP 发送执行位
VAR	bRecieve		BOOL		TCP/IP 接收执行位
VAR	arrSendData		ARRAY[1..4] OF WORD	[16#1111, 16#2222, 16#3333, 16#4444]	TCP/IP发送数组
VAR	arrRecieveData		ARRAY[1..4] OF WORD		TCP/IP接收数组
VAR	bClose		BOOL		TCP/IP关闭连接端口执行位

4. 编写实验案例

第一步，编写 TCP/IP Server 的监听功能块，确认 TCP/IP Sever 的 IP 地址为 169.254.0.1 和端口号 8000。

```
fbSocketListen(
    sSrvNetId:= ,
    sLocalHost:='169.254.0.88' , //TCP/IP Server的IP地址169.254.0.1
    nLocalPort:= 8000, //TCP/IP Server的通讯端口号8000
    bExecute:= blisten, //TCP/IP建立监听端口的执行位
    tTimeout:=T#1S ,
    bBusy=> ,
    bError=> ,
    nErrId=> ,
    hListener=>hListener); //TCP/IP的监听句柄
```

第二步，编写 TCP/IP Server 接受功能块。

```

fbSocketAccept(
    sSrvNetId:= ,
    hListener:=hListener ,           //TCP/IP的监听句柄
    bExecute:=baccept ,             //TCP/IP接受连接请求的执行位
    tTimeout:= T#1S,
    bAccepted=> ,
    bBusy=> ,
    bError=> ,
    nErrId=> ,
    hSocket=>hSocket );           //获取的TCP/IP句柄

```

第三步，编写 TCP/IP 发送功能块。

```

fbSocketSend(
    sSrvNetId:= ,
    hSocket:=hSocket ,
    cbLen:=SIZEOF(arrSendData) ,
    pSrc:=ADR (arrSendData),
    bExecute:= bSend,               //TCP/IP数据发送的执行位
    tTimeout:= T#1S,
    bBusy=> ,
    bError=> ,
    nErrId=> );

```

第四步，编写 TCP/IP 接收功能块

```

fbSocketReceive(
    sSrvNetId:= ,
    hSocket:=hSocket ,
    cbLen:=SIZEOF(arrRecieveData) ,
    pDest:=ADR (arrRecieveData) ,
    bExecute:=bRecieve,            //TCP/IP数据接收的执行位
    tTimeout:= T#1S ,
    bBusy=> ,
    bError=> ,
    nErrId=> ,
    nRecBytes=> );

```

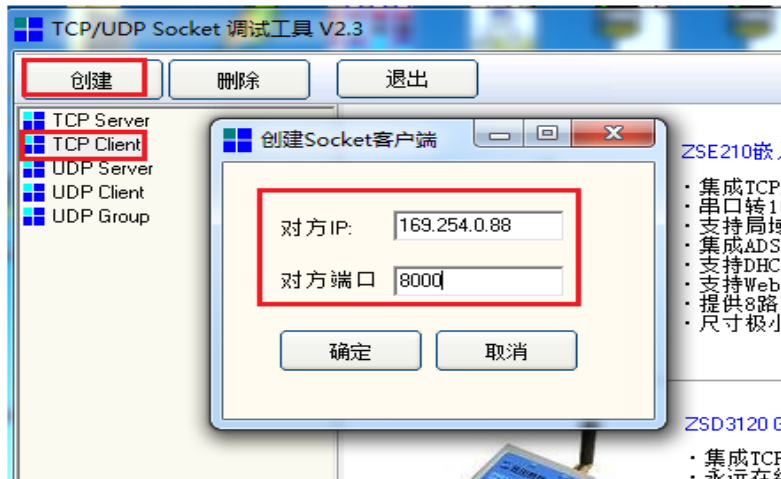
第五步，编写 TCP/IP 关闭功能块

```

fbSocketClose(
    sSrvNetId:= ,
    hSocket:=hSocket ,
    bExecute:=bClose ,            //TCP/IP端口关闭的执行位
    tTimeout:=T#1S ,
    bBusy=> ,
    bError=> ,
    nErrId=> );

```

5. 对 Socket Tool 设置，选中 TCP Client,点击创建，设置对方 IP 为 169.254.0.88，对方端口号为 8000。

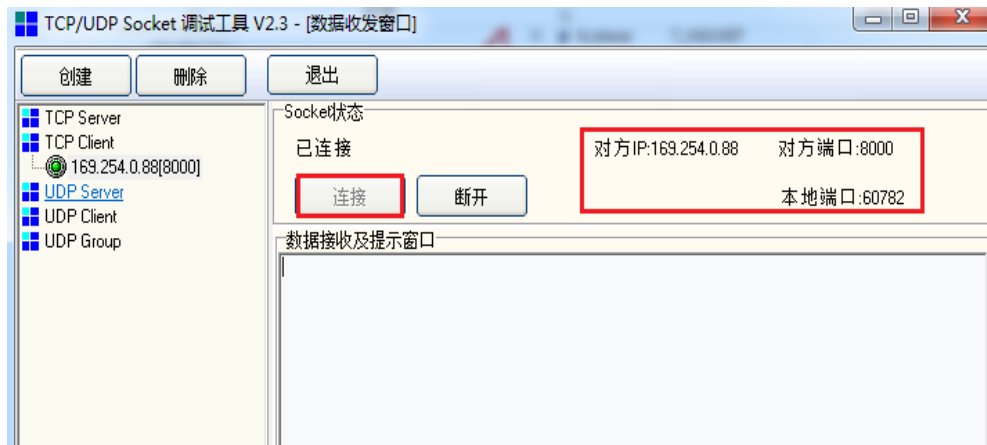


6. 连接上控制器，激活配置，下载程序，把 bListen 变为 True 之后，等待 TCP/IP Client 建立连接。

```
fbSocketListen(
  sSrvNetId:= ,
  sLocalHost:='169.254.0.88' := '169.254.0.88',
  nLocalPort:= 8000 := 8000,
  bExecute TRUE := blisten TRUE,
  tTimeout:= ,
  bBusy=> ,
  bError=> ,
  nErrId=> ,
  hListener=>hListener)
```

hListener	T_HSOCKET	
handle	UDINT	262145
localAddr	ST_SockAddr	
nPort	UDINT	8000
sAddr	STRING(15)	'169.254.0.88'
remoteA...	ST_SockAddr	
nPort	UDINT	0
sAddr	STRING(15)	"

7. 点击 Socket Tool 的连接，再去把 bAccpet 变为 True 之后，TCP/IP Client 和 TCP/IP Server 之间建立连接。

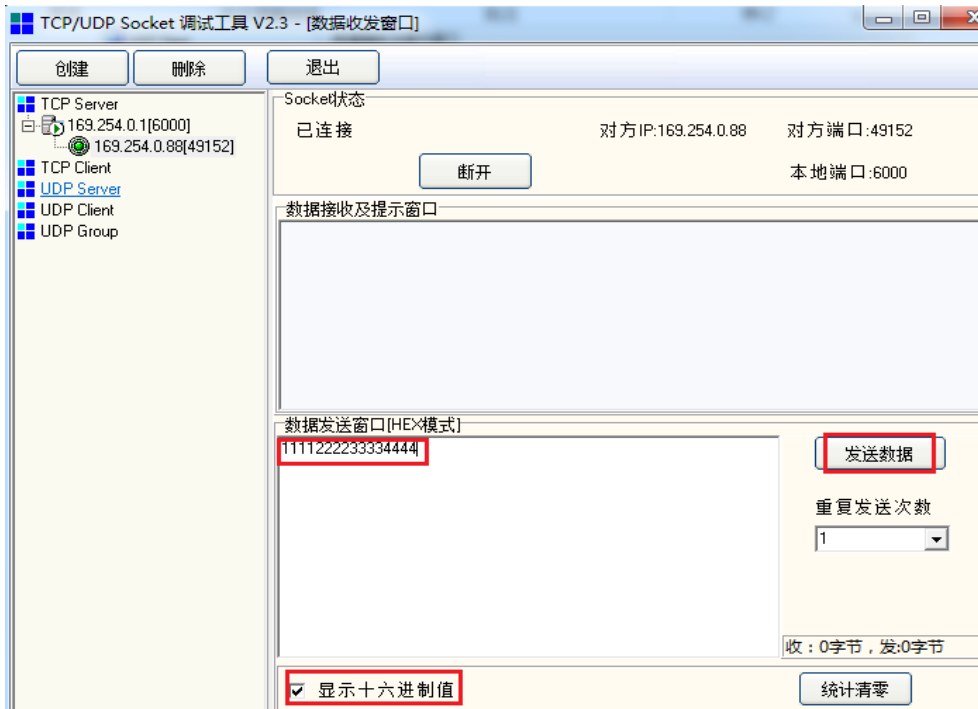


8. 读取 TCP/IP Client 和 TCP/IP Server 连接的句柄 hSocket，可以看见相关信息。

```
fbSocketAccept(
  sSrvNetId:= ,
  hListener:=hListener,
  bExecute TRUE := baccept TRUE,
  tTimeout:= ,
  bAccepted=> ,
  bBusy=> ,
  bError=> ,
  nErrId=> ,
  hSocket=>hSocket)
```

hSocket	T_HSOCKET	
handle	UDINT	131073
localAddr	ST_SockAddr	
nPort	UDINT	8000
sAddr	STRING(15)	'169.254.0.88'
remoteA...	ST_SockAddr	
nPort	UDINT	60782
sAddr	STRING(15)	'169.254.0.1'

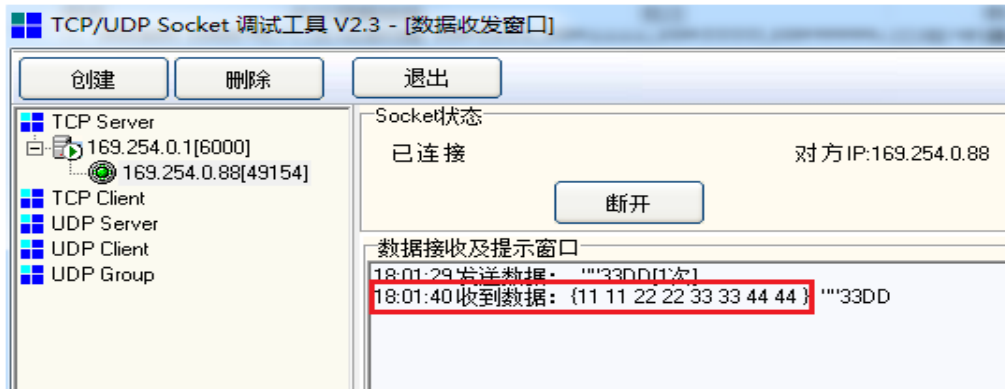
9. Socket Tool 写入发送数据 16#1111,16#2222,16#3333,16#4444，然后点击发送数据。导通 bReceive 之后，在数组 arrRecieveData 中可以接收到 Socket Tool 发送的数据。导通 bReceive，arrRecieveData 接收到 Socket Tool 发送的数据。



arrrecieveData	ARRAY [1..4] OF WO...			建立 TCP/IP接收数组
arrreciev...	WORD	16#1111		
arrreciev...	WORD	16#2222		
arrreciev...	WORD	16#3333		
arrreciev...	WORD	16#4444		

10. 导通 bReceive 之后便把数组 arrSendData，发送给 Socket Tool。

arrsendData	ARRAY [1..4] OF WO...			建立 TCP/IP发送数组
arrsend...	WORD	16#1111		
arrsend...	WORD	16#2222		
arrsend...	WORD	16#3333		
arrsend...	WORD	16#4444		



11. 如果不使用 TCP/IP 通讯，建议关掉 TCP/IP 通讯端口。

```
fbSocketClose(  
  sSrvNetId:= ,  
  hSocket:=hSocket ,  
  bExecuteTRUE:=bclose TRUE ,  
  tTimeoutT#1s:=T#1S ,  
  bBusy=> ,  
  bError=> ,  
  nErrId=> );RETURN
```

五、 配套 PLC 例程下载链接:

ftp://ftp.beckhoff.com.cn/TwinCAT3/Samples/TF6310-TcplpServer/TCPIP_SampleCode/TCPIP_Server_SampleCodeV1.tzip

六、 随堂问答

上海（中国区总部）

德国倍福自动化有限公司

上海市闸北区江场三路 163 号（市北工业园区）5 楼

电话：021-66312666 传真：021-66315696 邮编：200436

北京分公司

德国倍福自动化有限公司

北京市西城区西直门外大街 1 号西环广场 T3 写字楼 1801 - 1803 室

电话：010-58301236/7 传真：010-58301286 邮编：100044

广州分公司

德国倍福自动化有限公司

广州市天河区珠江新城珠江东路16号高德置地G2603室

电话：020-38010300/1/2 传真：020-38010303 邮编：510623

成都分公司

德国倍福自动化有限公司

成都市锦江区东御街18号 百扬大厦2305 房

电话：028-86202581 传真：028-86202582 邮编：610016



扫一扫，关注
倍福官方微信

技术服务热线：400-820-7388

倍福中文官网：

<http://www.beckhoff.com.cn/>

技术资料下载：

<ftp://ftp.beckhoff.com.cn>

倍福虚拟学院：

<http://tr.beckhoff.com.cn/>

招贤纳士：job@beckhoff.com.cn

技术支持：support@beckhoff.com.cn

产品维修：service@beckhoff.com.cn

方案咨询：sales@beckhoff.com.cn