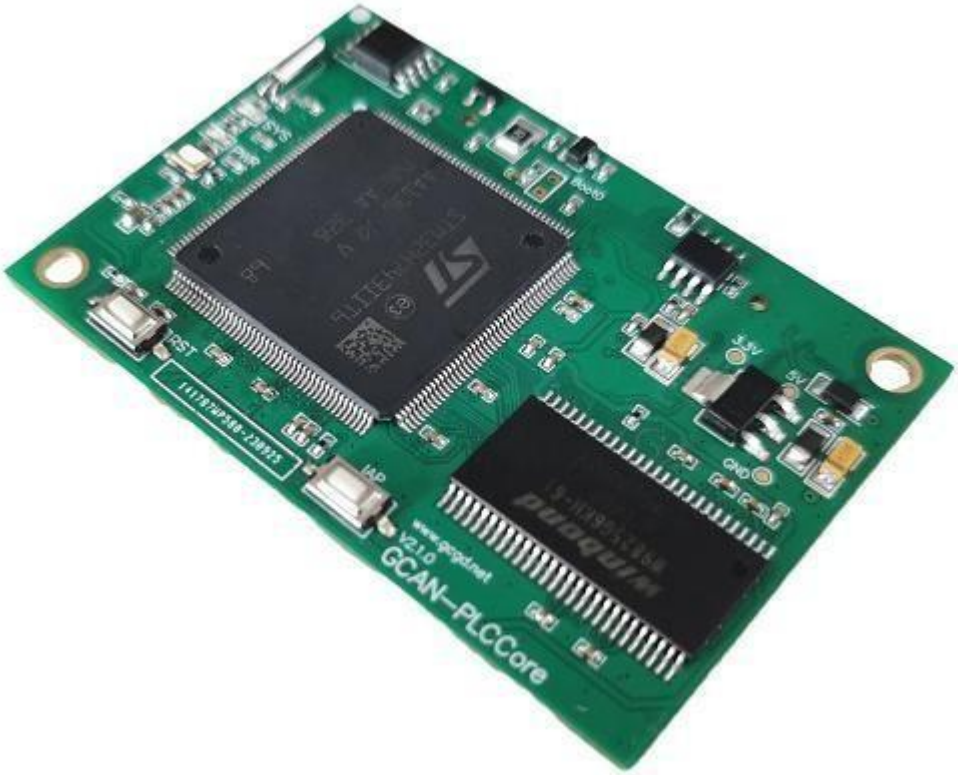
受　控

GCAN-PLCCore-M7

PLC核心模块及开发套件

用户手册



文档版本：V2.02（2026 / 08 / 18）

修订历史

版本	日期	原因
V1.00	2017/06/16	创建文档
V1.02	2017/10/17	修正设备工作参数
V1.03	2018/01/04	新增部分例程说明
V1.20	2018/09/14	调整文档结构
V1.30	2020/05/18	修改部分功能说明
V1.40	2021/10/25	修改部分工作参数
V2.00	2024/01/01	硬件更新 V2.00版本
V2.01	2026/7/22	修正设备工作参数
V2.02	2026/8/18	修正设备工作参数

目 录

1 功能简介.....3
1.1 功能概述.....3
1.2 PLC核心板资源.....3
1.3 典型应用.....3
2 GCAN-PLCCore-M7引脚定义及功能说明.....5
3 GCAN-PLCCore-M7启动策略.....8
4 GCAN-PLCCore-M7核心板参数表.....9
5 PLCCore-EDV-M7开发套件.....10
6 PLCCore-EDV-M7开发套件参数.....12
7 PLCCore-EDV-M7开发套件基本例程.....13
7.1 实验6ModbusTCP实验.....13
8 OpenPCSPLC软件使用.....19
8.1 软件安装.....19
8.2 PLC编程界面简介.....19
8.3 创建项目.....19
附录A：常见问题.....29
附录B：Modbus协议简介.....30
B.1 ModbusRTU协议数据格式.....30
B.2 ModbusTCP协议数据格式.....31
B.3 Modbus常用功能码.....33
9.免责声明.....41
销售与服务.....42

1 功能简介

1.1 功能概述

GCAN-PLCCore-M7 是一种嵌入式 PLC 核心模块。该模块已经集成了 PLC 控制内核，支持用户使用符合 IEC61131-3标准的五种编程语言对其编程，包括：CFC、SFC、LD、ST、IL 并支持在线调试功能包括观看和设置变量、单周期、断点和单步执行等功能。

GCAN-PLCCore-M7非常适合过程信号处理、集中监控、智能网络节点、分布式控制系统使用，还可以作为大系统中的特殊组件，也可以用作运动控制器的核心 PLC。

GCAN-PLCCore-M7核心模块已经包含整个 PLC 的运行环境和一部分存储外设，并且其还将芯片中的各种资源引出，提供多达 4路 UART，2路 CAN，1路 100M以太网，8路 AD，2路 DA，40个数字输入/输出，2个高速计数器输入和 2 路 PWM 输出。用户可以利用以上资源开发特定的程序进行采集与控制。

该模块还可按需定制。

1.2 PLC核心板资源

- CPU：ARM®32-bitCortex®-M7内核，处理器频率 400MHz；
- 4路 UART、2路 CAN、1路 100M以太网；
- 8路 AD、2 路 DA；
- 40个数字输入/输出（复用）；
- 2个高速计数器输入和 2路 PWM输出；
- 双 60pin插座（在底部），方便接入各种底板；
- 1个 5V&3.3V测试点，支持外接电源或输出电源给外部；
- 1个电源指示灯，1个状态指示灯；
- 1个复位按钮，可用于复位 MCU和 LCD。

1.3 典型应用

- 工业级 PLC开发；
- CAN工业自动化控制系统开发。

2 GCAN-PLCCore-M7引脚定义及功能说明

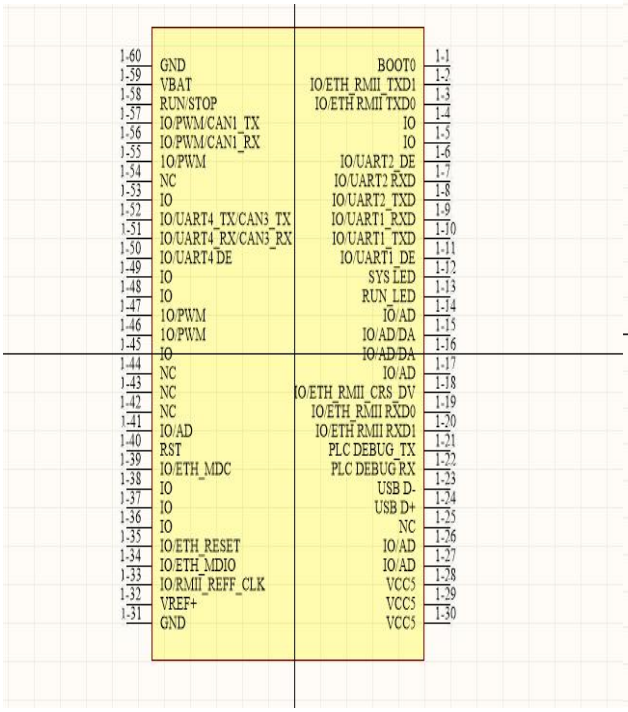
下表中对 GCAN-PLCCore-M7的引脚定义及功能进行了详细的说明。其中，IO表示可以做为 DI和 DO 使用，X 表示禁止使用，引脚悬空即可。

CON1							
引脚序号	引脚编号	功能定义	默认映射	引脚序号	引脚编号	功能定义	默认映射
1-1	P31	BOOT0		1-31	P1	GND	
1-2	P32	IO/ETH_RMII_TXD1	ETH	1-32	P2	VREF+	
1-3	P33	IO/ETH_RMII_TXD0	ETH	1-33	P3	IO/RMII_REFF_CLK	
1-4	P34	IO		1-34	P4	IO/ETH_MDIO	
1-5	P35	IO		1-35	P5	IO/ETH_RESET	
1-6	P36	IO/UART2_DE		1-36	P6	IO	
1-7	P37	IO/UART2_RXD		1-37	P7	IO	
1-8	P38	IO/UART2_TXD		1-38	P8	IO	
1-9	P39	IO/UART1_RXD		1-39	P9	IO/ETH_MDC	
1-10	P40	IO/UART1_TXD		1-40	P10	RST	
1-11	P41	IO/UART1_DE		1-41	P11	IO/AD	
1-12	P42	SYS_LED		1-42	P12	NC	
1-13	P43	RUN_LED		1-43	P13	NC	
1-14	P44	IO/AD		1-44	P14	NC	
1-15	P45	IO/AD/DA		1-45	P15	IO	
1-16	P46	IO/AD/DA		1-46	P16	IO/PWM	
1-17	P47	IO/AD		1-47	P17	IO/PWM	
1-18	P48	IO/ETH_RMII_CRS_DV		1-48	P18	IO	
1-19	P49	IO/ETH_RMII_RXD0		1-49	P19	IO	
1-20	P50	IO/ETH_RMII_RXD1		1-50	P20	IO/UART4_DE	
1-21	P51	PLC_DEBUG_TX		1-51	P21	IO/UART4_RX/CAN3_RX	
1-22	P52	PLC_DEBUG_RX		1-52	P22	IO/UART4_RX/CAN3_TX	
1-23	P53	USB D-		1-53	P23	IO	
1-24	P54	USB D+		1-54	P24	NC	
1-25	P55	NC		1-55	P25	IO/PWM	
1-26	P56	IO/AD		1-56	P26	IO/PWM/CAN1_RX	
1-27	P57	IO/AD		1-57	P27	IO/PWM/CAN1_TX	
1-28	P58	VCC5		1-58	P28	RUN/STOP	
1-29	P59	VCC5		1-59	P29	VBAT	
1-30	P60	VCC5		1-60	P30	GND	

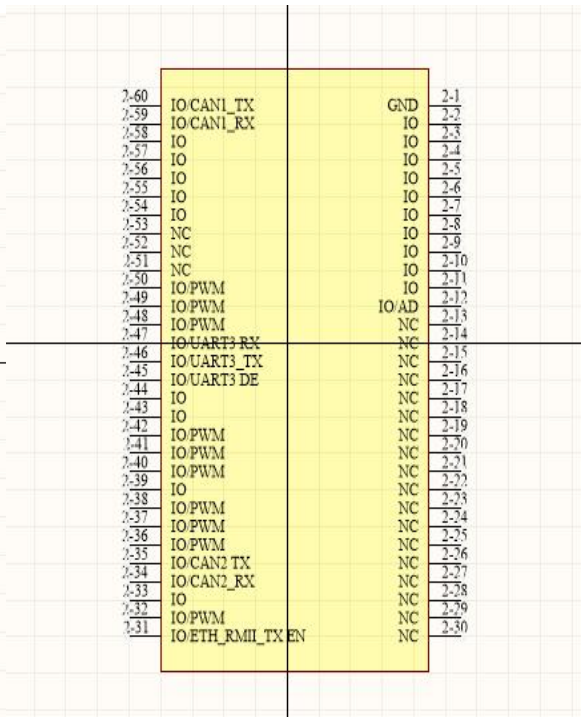
CON2							
引脚序号	引脚编号	功能定义	默认映射	引脚序号	引脚编号	功能定义	默认映射
2-1	P91	GND		2-31	P61	IO/ETH_RMII_TX_EN	
2-2	P92	IO		2-32	P62	IO/PWM	
2-3	P93	IO		2-33	P63	IO	
2-4	P94	IO		2-34	P64	IO/CAN2_RX	
2-5	P95	IO		2-35	P65	IO/CAN2_TX	
2-6	P96	IO		2-36	P66	IO/PWM	
2-7	P97	IO		2-37	P67	IO/PWM	
2-8	P98	IO		2-38	P68	IO/PWM	
2-9	P99	IO		2-39	P69	IO	
2-10	P100	IO		2-40	P70	IO/PWM	
2-11	P101	IO		2-41	P71	IO/PWM	
2-12	P102	IO/AD		2-42	P72	IO/PWM	
2-13	P103	NC		2-43	P73	IO	
2-14	P104	NC		2-44	P74	IO	
2-15	P105	NC		2-45	P75	IO/UART3_DE	
2-16	P106	NC		2-46	P76	IO/UART3_TX	
2-17	P107	NC		2-47	P77	IO/UART3_RX	
2-18	P108	NC		2-48	P78	IO/PWM	
2-19	P109	NC		2-49	P79	IO/PWM	
2-20	P110	NC		2-50	P80	IO/PWM	
2-21	P111	NC		2-51	P81	NC	
2-22	P112	NC		2-52	P82	NC	
2-23	P113	NC		2-53	P83	NC	
2-24	P114	NC		2-54	P84	IO	
2-25	P115	NC		2-55	P85	IO	
2-26	P116	NC		2-56	P86	IO	
2-27	P117	NC		2-57	P87	IO	
2-28	P118	NC		2-58	P88	IO	
2-29	P119	NC		2-59	P89	IO/CAN1_RX	
2-30	P120	NC		2-60	P90	IO/CAN1_TX	

GCAN-PLCCore-M7核心板采用“3710F端子”型号连接器与底板连接，如用户想自己开发底板，则底板参考原理图及引脚定义如下图所示，具体连接器原理图及封装可收到货后向技术支持工程师索要。

CON1：



CON2：



3 GCAN-PLCCore-M7启动策略

默认情况下，GCAN-PLCCore-M7在上电或重启时会加载所有必要的固件程序，之后会开始运行 PLC 程序。因此，GCAN-PLCCore-M7 适合使用于独立型的控制系统。在电源崩溃的情况下，这样的系统能够恢复执行 PLC程序，而无需用户干预。下图演示了系统启动的细节：

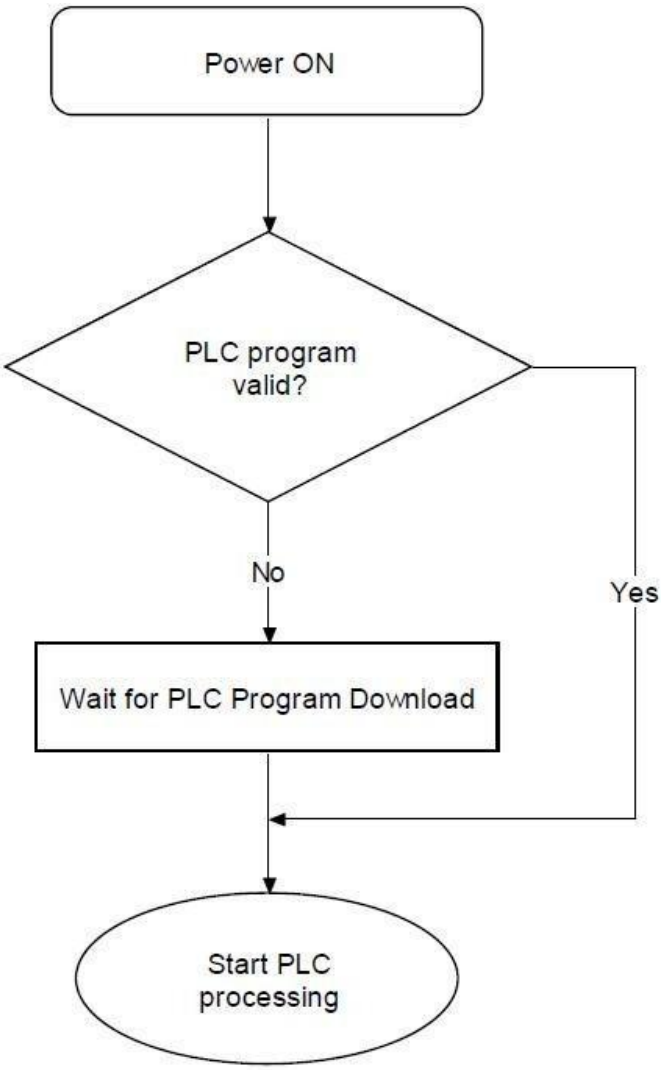


图 4.1GCAN-PLCCore-M7启动策略

如果固件在非易失性存储器中找到了有效的控制程序，则该程序将重新启动和处理。否则将激活程序，进入 GCAN-PLCCore-M7 停止模式。

4 GCAN-PLCCore-M7核心板参数表

控制器特点	
控制器	STM32F7
内核	ARM®32-bitCortex®-M7
处理器频率	400MHz
程序存储空间	32M字节 （OpenPCS） 8M字节（CODESYS）
数据存储器空间	16M字节（OpenPCS） 8M字节（CODESYS）
掉电保存空间	2K字节（OpenPCS） 4K字节（CODESYS）
接口特点	
以太网	1路，10/100Mbps
RS232/RS485	4路，2400bps~115200bps
CAN接口	2路，遵循ISO11898标准，支持CAN2.0A/B
数字量输入/输出	50路DI/DO（复用）
模拟量输入	6路（CODESYS内核支持） 8路（OPENPCS支持）
模拟量输出	2路（仅CODESYS内核支持）
板载接口	双60pin插座，0.8mm管脚间距
软件	内部集成IEC61131-3实时内核
编程接口	TCP、RS485/232(可选)
RTC	板载实时时钟，需要外接时钟电池
供电电源	
供电电压	+5VDC（±5%）
供电电流	200mA
环境试验	
工作温度	-40℃~+85℃
工作湿度	15%~90%RH，无凝露
EMC测试	EN55024:2011-09 EN55022:2011-12
防护等级	IP20
基本信息	
外形尺寸	65mm*45mm
重量	10g

5 PLCCore-EDV-M7开发套件

PLCCore-EDV-M7开发套件是一套高性能低成本的开发套件。基于GCAN-PLCCore-M7核心板，开发套件 PLCCore-EDV-M7能使用户快速完成分散、网络相兼容的自动化项目。此外，它有助于用户了解基于 IEC61131-3PLC编程相比传统编程语言的优点。

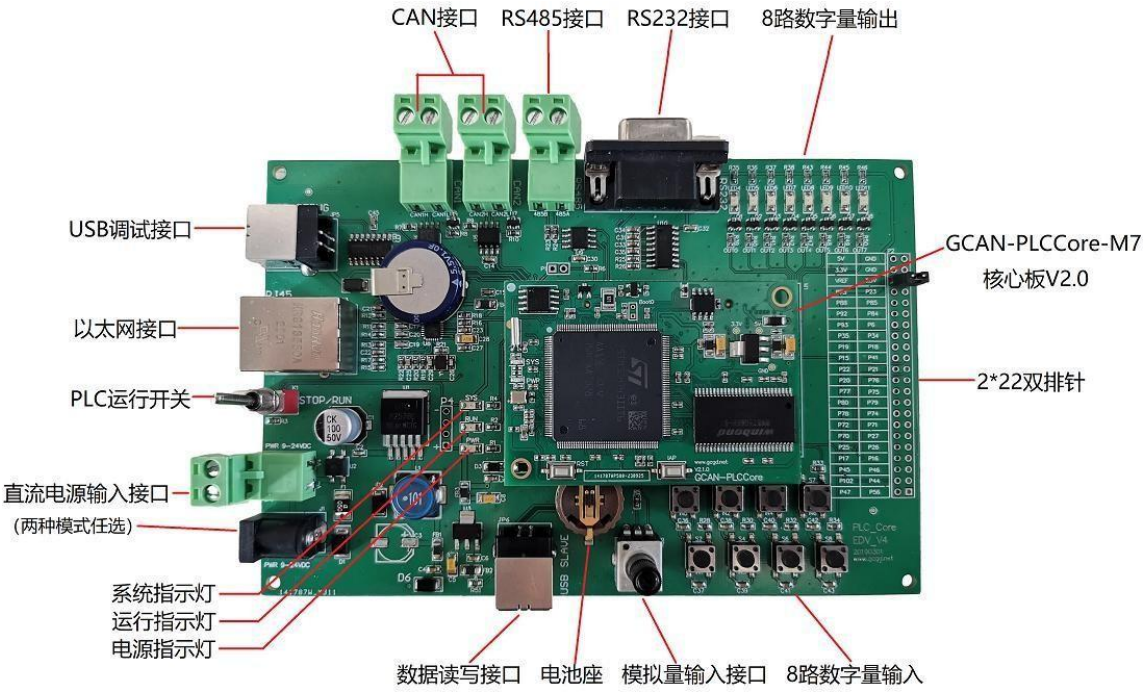


图 6.1PLCCore-EDV-M7开发套件

GCAN-PLCCore-M7核心板配套开发板底板的板载资源如下：

- 1个核心板接口，支持 GCAN-PLCCore-M7核心板；
- 1个电源指示灯，2个状态指示灯；
- 2路 CAN接口，1路以太网接口；
- 1路 RS232接口，1路 RS485接口；
- 8路数字量输入，8路数字量输出；
- 1路 USB串口，可用于 PLC程序下载和代码调试；
- 1路 USBSlave，可用于 PLC内用户数据读取与写入；
- 1个直流电源输入接口（输入电压范围：DC9-30V）；
- 1个 RTC后备电池座；

- 1个开关，用于 PLC的 RUN和 STOP；
- 1组 2x22的双排针，用于引出核心板内其余的引脚；
- 引出 2x22个接口，包含电源及 IO接口，满足各种应用需求；
- 外形尺寸为 150mm*100mm。

6 PLCCore-EDV-M7开发套件参数

PLCCore-EDV-M7 开发套件（GCAN-PLCCore-M7 核心板配套底板）技术参数如下所示：

核心板特点	
核心板	GCAN-PLCCore-M7
内核	ARM®32-bitCortex®-M7
接口特点	
以太网	1路，10/100Mbps
RS232	1路，600bps~115200bps
RS485	1路，600bps~115200bps
CAN接口	2路，遵循ISO11898标准，支持CAN2.0A/B
CAN波特率	1000K、500K、250K、200K、125K、100K、50K、20K
数字量输入	8路DI，按键输入
数字量输出	8路DO，LED指示灯
板载接口	双60pin插座，0.8mm管脚间距
编程接口	RS232
RTC	板载实时时钟，需要外接时钟电池
供电电源	
供电电压	+9-30VDC
环境试验	
工作温度	-40℃~+85℃
工作湿度	15%~90%RH，无凝露
EMC测试	EN55024:2011-09 EN55022:2011-12
防护等级	IP20
基本信息	
外形尺寸	150mm *100mm
重量	100g

7 PLCCore-EDV-M7开发套件基本例程

PLCCore-EDV-M7 开发套件提供例程代码，通过这些例程用户可以快速熟悉GCAN-PLCCore-M7的使用及其特性，用户可以快速开发自己的项目。例程可以登录官网 www.gcgd.net下载，也可以联系技术工程师索要。下面以其中一个例程为例进行讲解。

7.1 实验 6ModbusTCP实验

本实验用于实现 GCAN-PLCCore的 ModbusTCP 从站功能，涉及的功能码主要包含 0203 04 05。Modbus TCP 的数据格式详见附录“表 B.2 Modbus TCP 响应数据格式”。

A.下载程序

请参照“**9.3.4**设置调试连接”及“**9.3.5**下载程序并调试”完成程序的下载。下载完程序之后，您需要点击一下 GCAN-PLCCore 上面的 RESET 键或重新给GCAN-PLCCore 开发板上电，使新的程序生效。

B.主要功能块说明

本实验中使用的功能块有： LAN_INIT、 LAN_ASCII_TO_INET、Modbus_TCP_SLAVE_INIT、Modbus_SLAVE_CTRL 四个功能块。它们的作用如下表所示。功能块原型、操作数定义及描述等信息请参照“GCAN-PLC 扩展功能块说明书”。

功能块	作用
LAN_INIT	用于初始化以太网接口。
LAN_ASCII_TO_INET	将 IP地址的 ASCII码形式转换成 IP地址的整数形式。
Modbus_TCP_SLAVE_INIT	用于初始化 ModbusTCP从站接口。
Modbus_SLAVE_CTRL	用于执行 ModbusTCP从站接口的控制指令。

表 8.1ModbusTCP实验中使用的功能块

C.程序段说明

打开 TEST.ST文件，可查看到如下代码：

iflanInit=falsethen

```
mIP := LAN_ASCII_TO_INET('192.168.1.31');
mNetMask := LAN_ASCII_TO_INET('255.255.0.0');
mGateWay:= LAN_ASCII_TO_INET('192.168.1.1');
    inst0_LAN_INIT(
            ENABLE :=true ,
            HOSTNAME:='PLCCore',
            IP := mIP,
            NETMASK:=mNetMask,
            GATEWAY:=mGateWay,
            NETNUMBER:=1|mConfirm:=CONFIRM,
            mError:= ERROR,
            mErrorinfo:=ERRORINFO
             );
    lanInit:=true;
    else
        ifmModbusInitOK=falsethen

            inst0_MODBUS_TCP_SLAVE_INIT(
            ENABLE:=true,
            MODE := 1,
            ADDRESS:=1,
            NETNUMBER:=1    |   mConfirm:=CONFIRM,
            mError:= ERROR,
            mErrorinfo:=ERRORINFO
            );
            mModbusInitOK:=true;
        end_if;
    end_if;
```

这段代码用于初始化以太网接口和 ModbusTCP从站接口。用户可通过设置IP、NETMASK、GATEWAY 这三个参数，建立以太网连接。可通过将 MODE设置为 1将 Port定义为 Modbus协议并使能该协议，通过将 MODE设置为 0将Port定义为 PPI并禁止 Modbus协议；通过 ADDRESS参数设定 TCP端的本站地址。

```
VAR
lanInit:bool:=false;
inst0_LAN_INIT:LAN_INIT;
mConfirm:bool;
mError:usint;
```

```
mErrorinfo:usint;
mIP:udint;
mNetMask:udint;
mGateWay:udint;
mSocket:INT;
inst0_MODBUS_TCP_SLAVE_INIT:MODBUS_TCP_SLAVE_INIT;
mModbusInitOK:bool:=false;
modbusDOBuf:ARRAY[0..5]OFbyte;
DO_Ptr : POINTER;
modbusDIBuf:ARRAY[0..5]OFbyte;
DI_Ptr:POINTER
modbusAIBuf:ARRAY[0..10]OFint;
AI_Ptr : POINTER;
modbusRegBuf:ARRAY[0..127]OFint;
mPtr : POINTER;
xDONE:BOOL;

inst4_MODBUS_TCP_SLAVE_CTRL:MODBUS_TCP_SLAVE_CTRL;
xError:usint;
mDIat%I0.0:byte;
mAI at %I1.0:int;
mDO      at%Q0.0:byte;
END_VAR

mDO:=modbusDOBuf[0];
modbusDIBuf[0]:=mDI;
modbusAIBuf[0]:=mAI;
modbusRegBuf[0]:=byte_to_int(mDI);      (*此处定义 modbus数字量输入值存放在*)
                                        (*Modbus保持寄存器中的地址*)
modbusRegBuf[1]:=mAI;                   (*模拟量输入值*)
modbusRegBuf[2]:=byte_to_int(mDO);      (*数字量输出值*)

mPtr:=&modbusRegBuf;
DO_Ptr:=&modbusDOBuf;
DI_Ptr:=&modbusDIBuf;
AI_Ptr:=&modbusAIBuf;

if mModbusInitOK=true then
    inst4_MODBUS_TCP_SLAVE_CTRL(
        NETNUMBER:=1,
        DO_ENABLE :=1 ,
        DO_PTR:=DO_Ptr,
        DO_LENGTH :=5 ,                 (*请注意其长度限制*)
        DI_ENABLE :=1 ,
```

```
            DI_PTR:=DI_Ptr,
            DI_LENGTH:=5,
            AI_ENABLE  :=1  ,
            AI_PTR  :=AI_Ptr  ,
            AI_LENGTH:=10,
            REG_ENABLE :=1 ,
            REG_PTR :=mPtr ,
            REG_LENGTH:=127
            |xDONE:=DONE,
            xError:=ERROR);
end_if;
```

这段代码用于执行 ModbusTCP从站接口的控制指令。modbusDIBuf 为输入寄存器（数字量），modbusAIBuf 为输入寄存器（模拟量），modbusRegBuf 为保持寄存器。modbusRegBuf 共包含 127 个地址，您可以通过赋值语句将不同的输入输出值存放到Modbus保持寄存器中的地址。如“RegBuf[3]:=byte_to_int(DO); ”，将数字量输出值存放到 Modbus 寄存器中的 3 号地址。

D.实际测试

您可以通过网络调试助手给 GCAN-PLCCore-M7发送控制指令。

控制指令含义	控制指令及返回指令	功能码及含义
点亮 LED0	发送：0000 00 00 00 0601 **05** 00 00 FF00 返回：0000 00 00 00 0601 **05** 00 00 FF00	05强置单线圈
点亮 LED1	发送：0000 00 00 00 0601 **05** 00 01 FF00 返回：0000 00 00 00 0601 **05** 00 01 FF00	05强置单线圈
熄灭 LED0	发送：00 00 00 00 00 06 01 **05** 00 00 00 00 返回：00 00 00 00 00 06 01 **05** 00 00 00 00	05强置单线圈
读取 DI的值	发送：00 00 00 00 00 06 01 **02** 00 00 00 08 返回：00 00 00 00 00 04 01 **02** 01 0C	02读取输入状态
读取 AI的值	发送：00 00 00 00 00 06 01 **04** 00 00 00 01 返回：00 00 00 00 00 05 01 **04** 02 0FFB	04读取输入寄存器
读取两字节保持寄存器	发送：00 00 00 00 00 06 01 **03** 00 00 00 02 返回：000000000007010**3**040000000C	03读取保持寄存器

表 8.2ModbusTCP实验测试指令

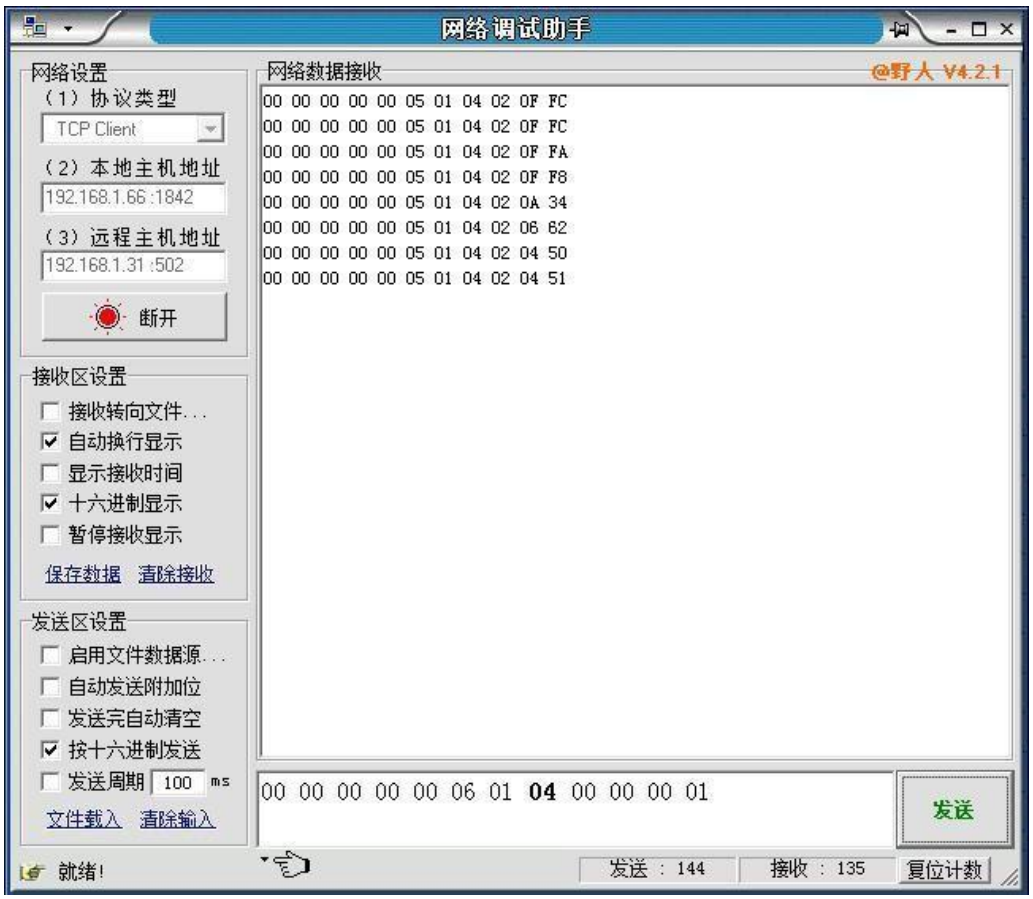


图 8.1ModbusTCP实验数据发送界面

我们以“读取 AI的值”为例，说明一下 ModbusTCP发送数据与接收数据的含义。通过发送如表 8.3显示的指令，主机将读取 PLC（Server）中的输入寄存器，即 modbusAIBuf，从“0000”起始地址开始，读取一个寄存器。PLC（Server）回送数据如表 8.4所示。您也可以通过发送“000000000006010300020001”，通过读取保持寄存器，即 modbusRegBuf，从“0002”起始地址开始，读取一个寄存器，同样可以返回当前 AI 的值。

主机（**Client**）发送	字节	数据（**Hex**）
传输标识码	Byte0、Byte1	00 00
协议标识符	Byte2、Byte3	00 00
数据长度	Byte4、Byte5	00 06
设备地址	Byte6	01
功能码	Byte7	**04**
起始地址	Byte8、Byte9	00 00
读取寄存器数量	Byte10、Byte11	00 01

表 8.3 “读取 AI 的值”发送指令的含义

从机（**Server**）回送	字节	数据
传输标识码	Byte0、Byte1	00 00
协议标识符	Byte2、Byte3	00 00
数据长度	Byte4、Byte5	00 05
设备地址	Byte6	01
功能码	Byte7	04
数据字节数	Byte8	02
寄存器值	Byte9、Byte10	0F0C

表 8.4 “读取 AI 的值”回送指令的含义

8 OpenPCSPLC软件使用

8.1 软件安装

OpenPCS2008编程软件（随货光盘中包含本软件，也可网上下载安装软件）

8.2 PLC编程界面简介

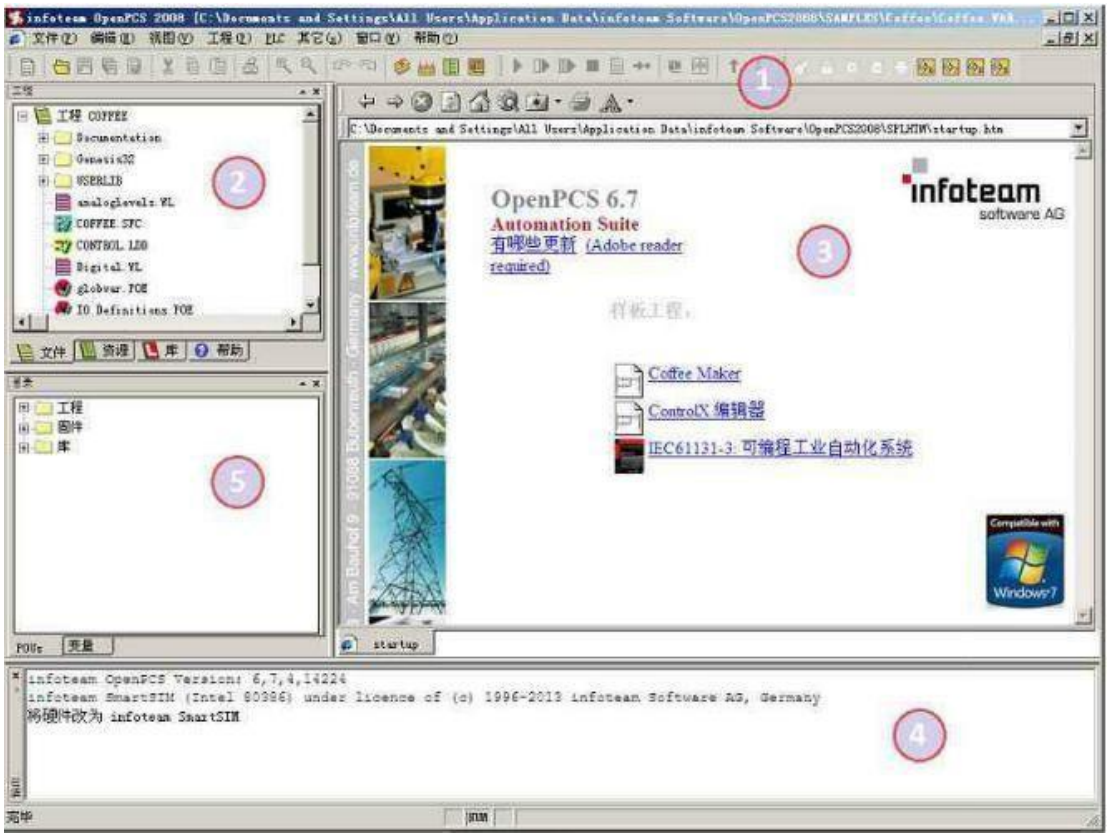


图 8.1OpenPCS编程界面

OpenPCS编程界面中主要包含：

1） 菜单工具栏

2） 工程浏览器

3） 编辑窗口

4） 输出窗口

5） 目录窗口

8.3 创建项目

8.3.1 工程创建

点击文件->新建，创建新项目，如下图 9.2所示。

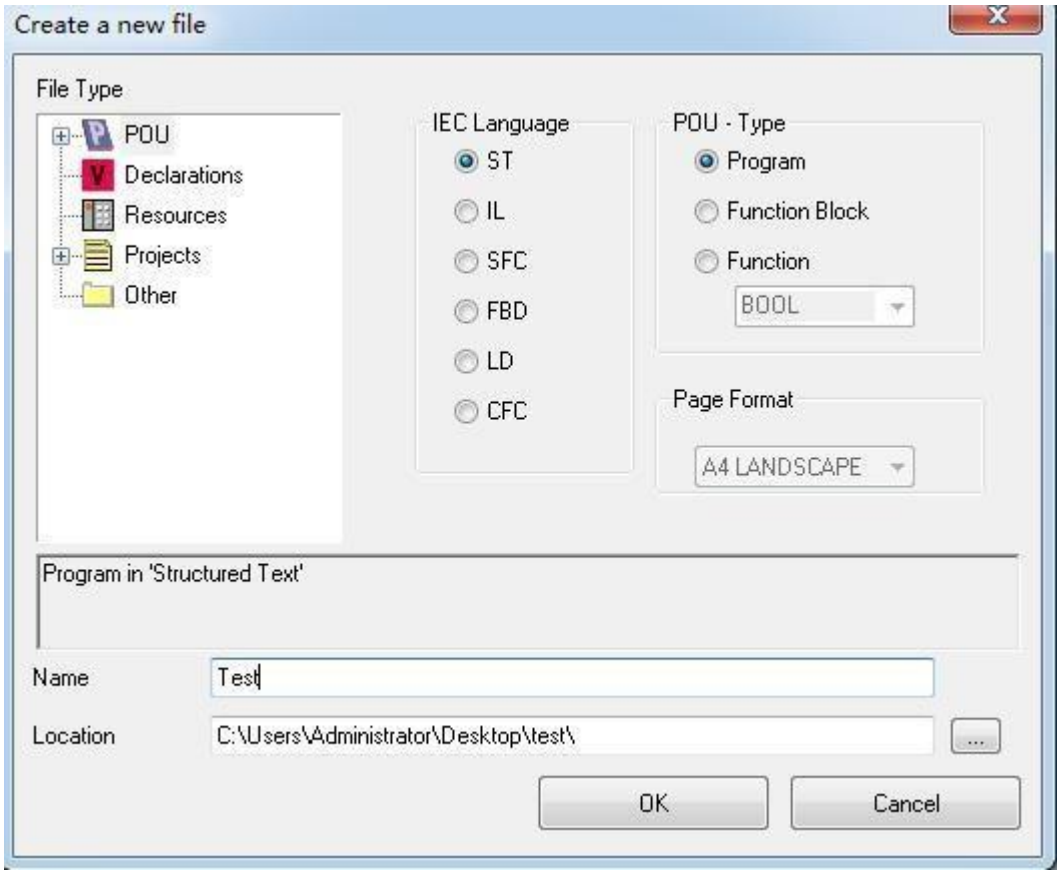


图 8.2创建项目

8.3.2 添加文件

为项目添加文件(例如：添加功能块-FuctionBlock，SampleFB)，如图 9.3所示。请注意，**Name**（名称）一栏中填入的字符串不能以数字为开头。

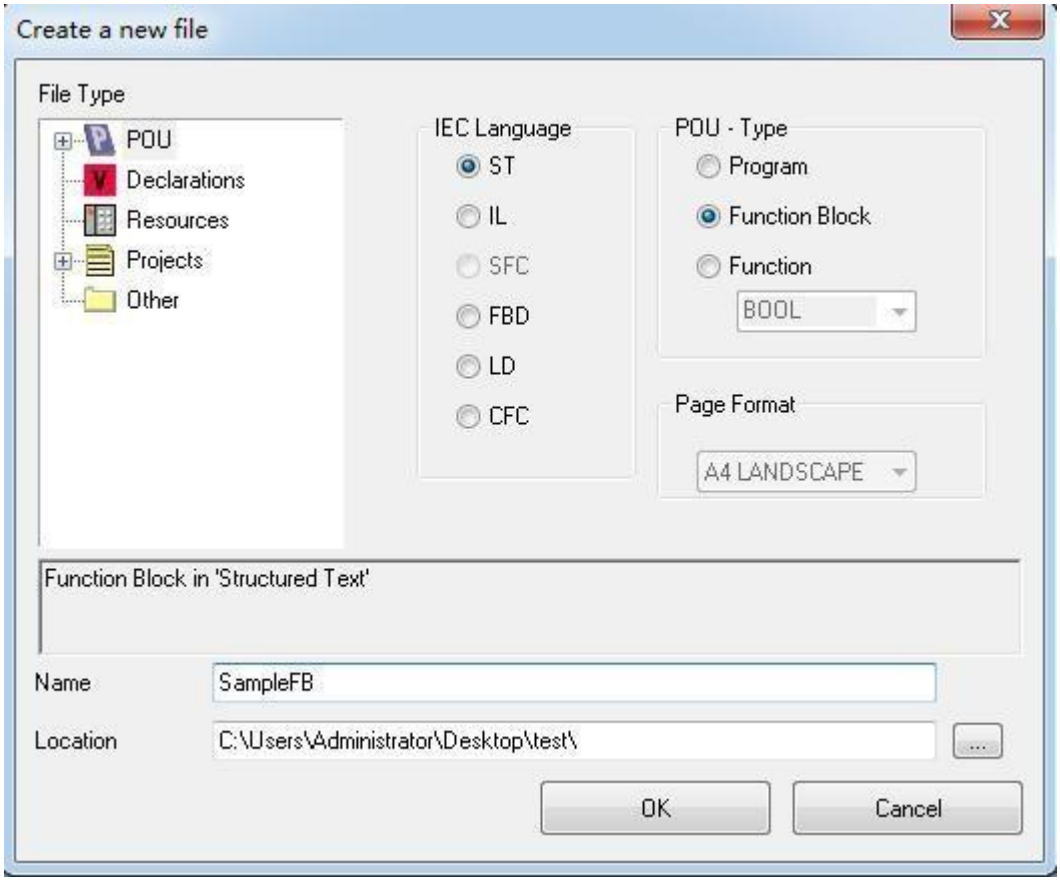


图 8.3创建功能块

8.3.3 程序编写

首先需要在变量区定义变量（VAR到 END_VAR）。

```
VAR                             (*内部变量段开始                                    *)
    v1:INT:=0;                  (*:为变量/类型分隔符，:=为初始化操作符              *)
    v2:INT:=0;
    oled at%Q0.0:Byte;          (*   %Q0.0表示输出0单元第0位，:为变量/类型分割符    *)
                                (*符号变量地址声明。分配Q0.0到字节 OLED             *)
                                (* 如果对变量声明不理解，可参考电子书第49页，变量声明的示例*)
END_VAR
```

完成变量定义后便可在下方的编程界面开始编程了，下面为用 ST编写的简单例程语句：

LED 跑马灯例程：

```
IF v1<100 THEN                  (*v1自加到100时，v1归零。v1越大，闪灯的变化频率越慢   *)
    v1:=v1+1;                   (* := 可表示初始化、输入连接或者赋值                 *)
                                (* 如果对st语言的各种符号不理解可参考电子书第25页，分界符 *)
ELSE
    v1:=0;
    v2:=v2+1;
    if v2>=255 then             (*v2自加到255时，v2归零                             *)
       v2:=0;
    end_if;
    oled:=int_to_byte(v2);      (* int_to_byte 整型转字节。类型转换类函数，电子书57页 *)
                                (* :=这里表示赋值                                    *)
end_if;
```

8.3.4 设置调试连接

1、 点击 PLC->Connections...（连接...）。

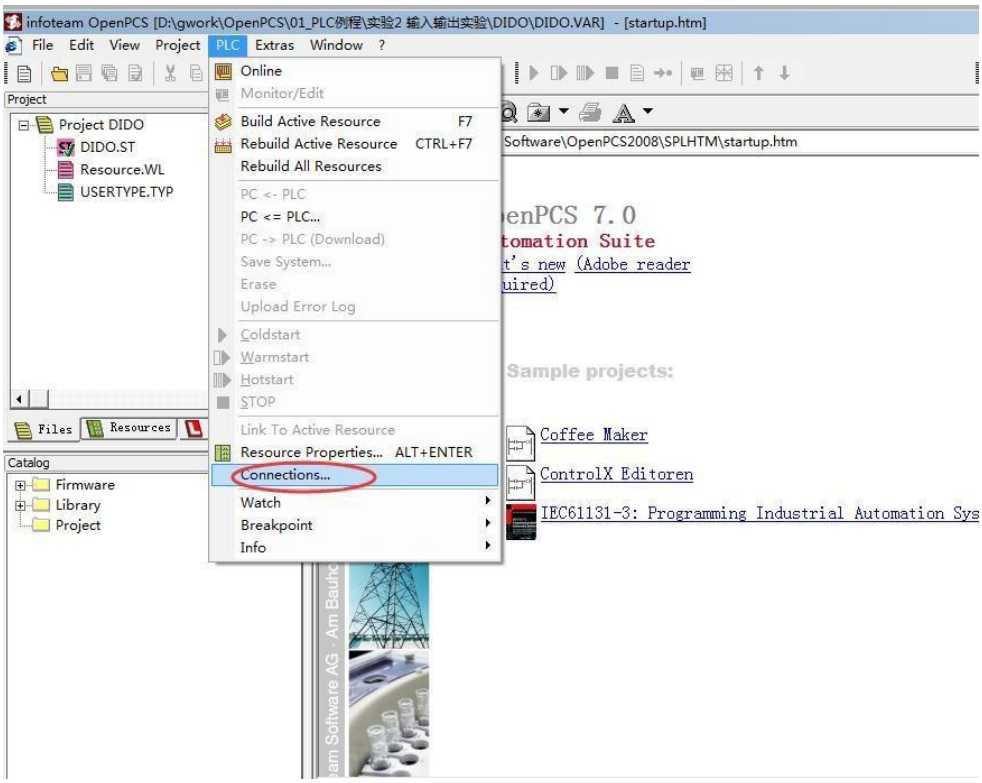


图 8.4调试连接

2、在ConnectionSetup（连接设置）窗口新建连接，设置参数。点击“New”按钮。

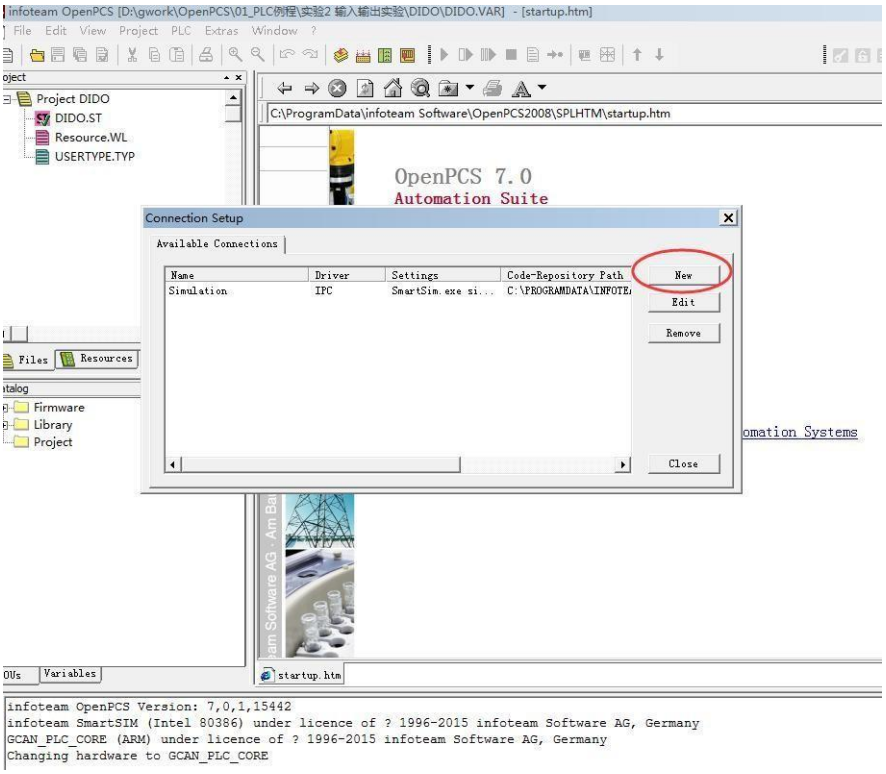


图 8.5点击“New”

3、 在 Name 中输入 TCP，点击 Select按钮。

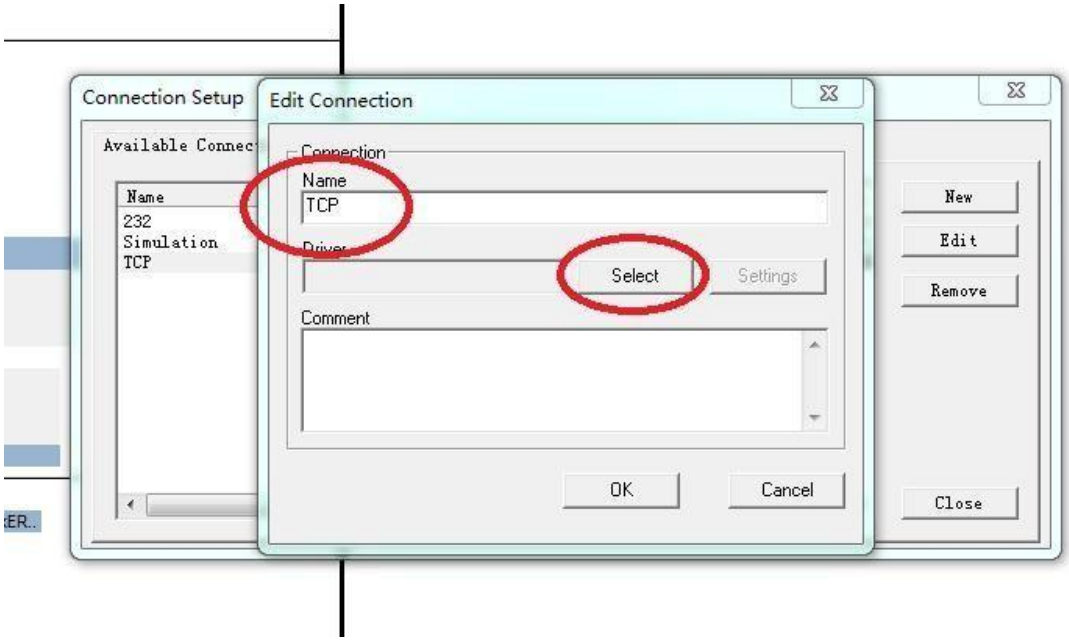


图 8.6点击“Select”按钮

4、 点击TCP432图标 

，之后点击OK。

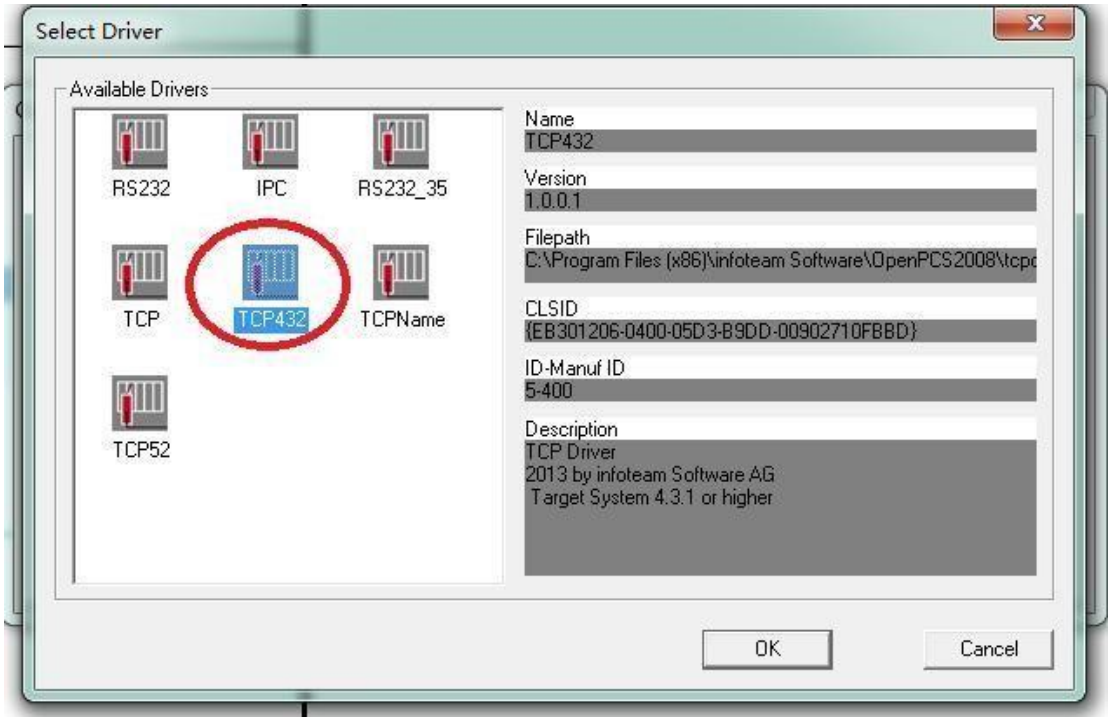


图 8.7选择 TCP432

5、Driver中会显示“TCP432”字样，点击“Settings（设置）”按钮。



图 8.8点击“Settings”按钮

6、Port（端口）请输入 23042。IP地址为 192.168.1.30，设置好后点 OK。

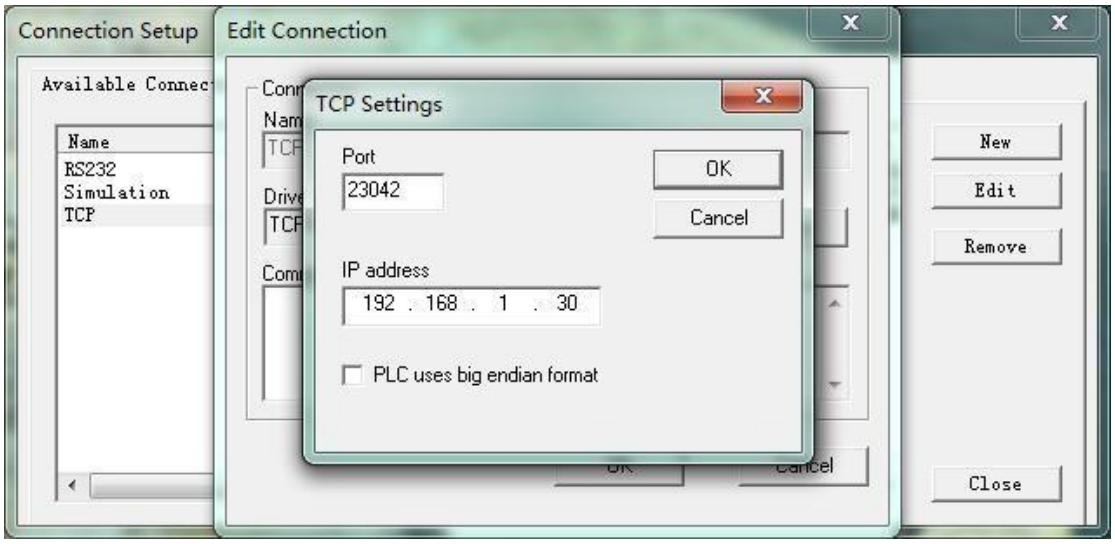


图8.9　IP地址及端口号设置

7、设置好后，返回 ConnectionSetup（连接设置）界面，点击“Close（关闭）”。

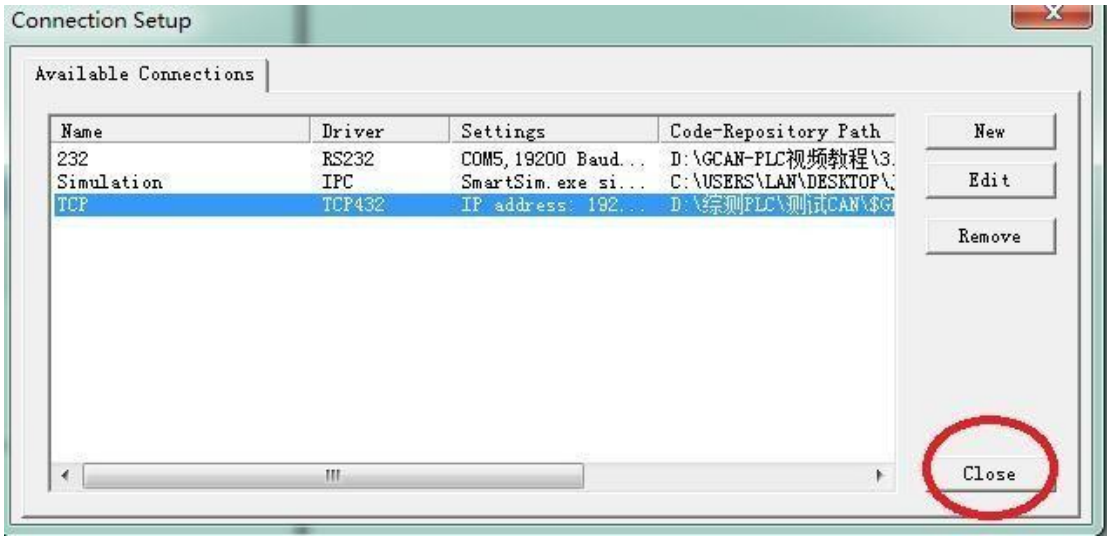


图 8.10点击“Close”

8、 设置 ResourceProperties（资源属性），如下图所示。

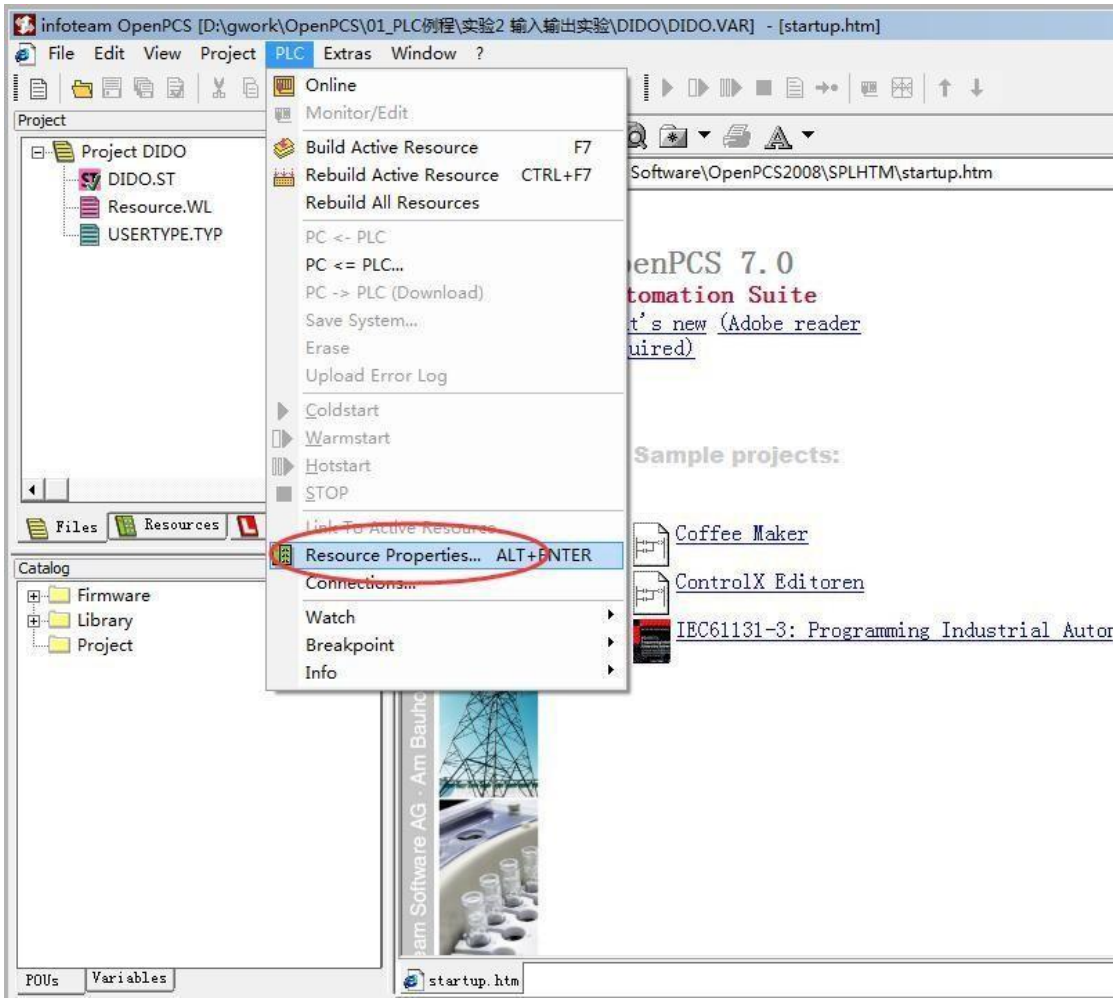


图 8.11设置资源属性

9、 选择 GCAN_PLC和 TCP。

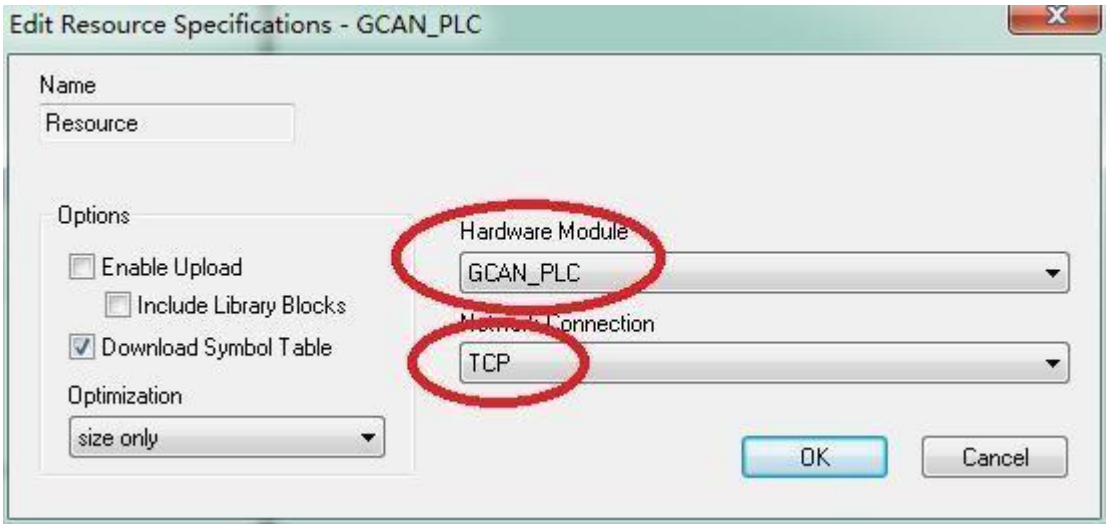


图 8.12选择 GCAN_PLC和 TCP

8.3.5 下载程序并调试

1、 完成程序编写后需点击 BuildActiveResource（生成当前资源）按钮，如图 9.13 所示。

图 8.13点击 BuildActive Resource按钮

2、 编译完成后，提示没有错误。如下图所示。

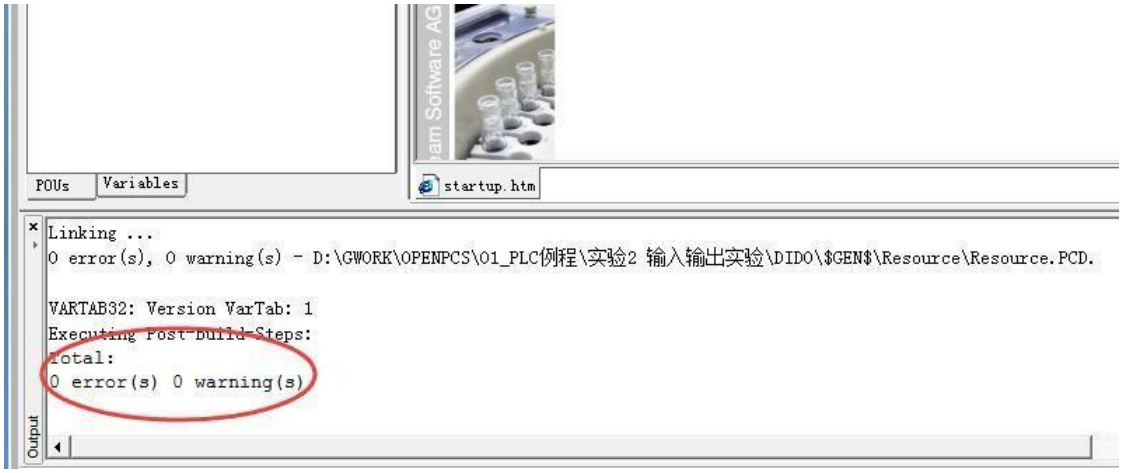


图 8.14编译完成

3、 点击 Online（联机）按钮。

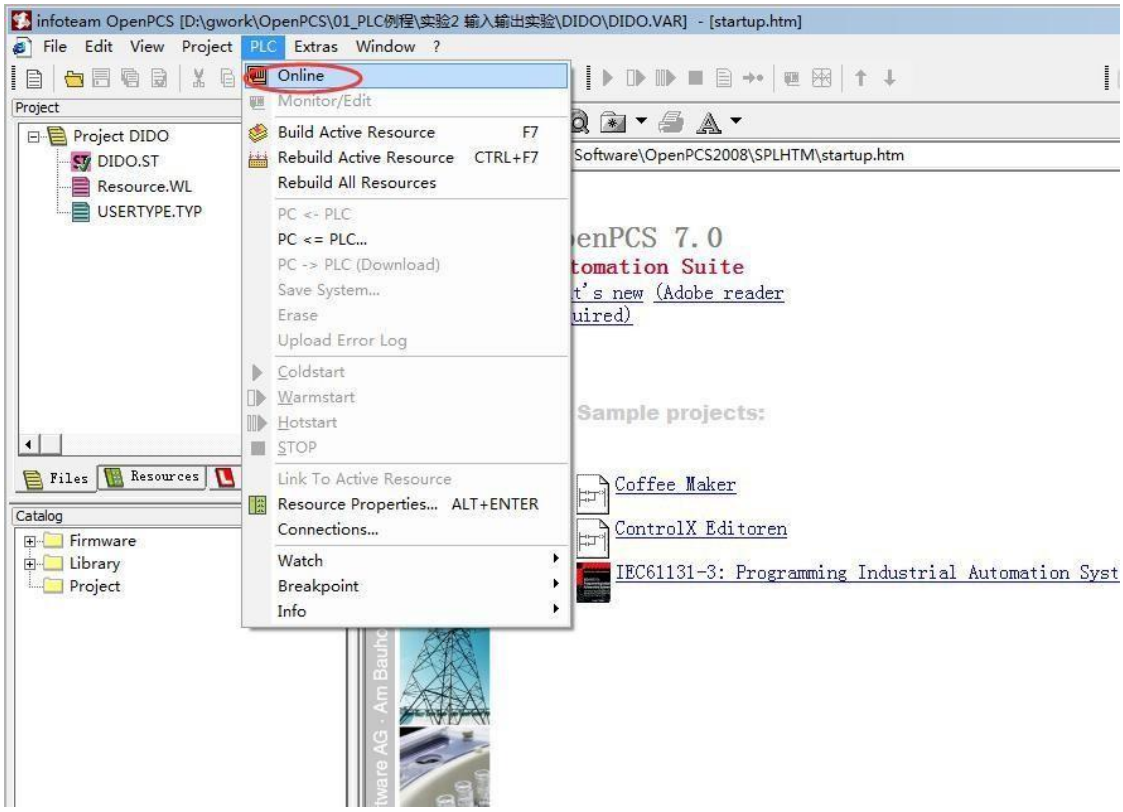


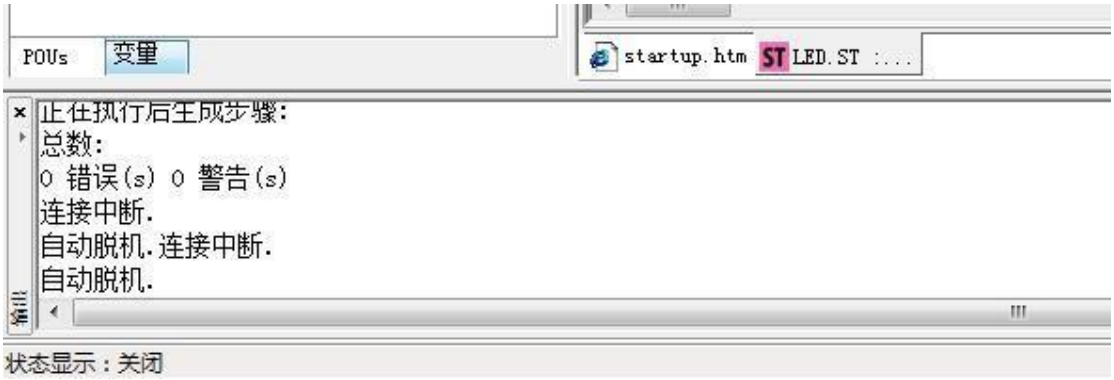
图 8.15点击 Online按钮

4、 在下拉菜单中点击 PC->PLC(Download)下载程序。

图 8.16下载程序

附录 A：常见问题

1.Q: 点击“联机”按钮后，状态栏中显示自动脱机。无法下载程序。



A: 请确定您已按照“图 9.9IP地址及端口号设置”设置好了网口端的通信参数，设置的网口 IP地址与电脑的 IP地址有相同的前三段地址。如 PLC还需要通过网口进行 MODBUS-TCP 等通讯，则可外接交换机实现网口的复用。

附录 B：Modbus协议简介

Modbus通信协议是由Modicon公司开发的应用在PLC或其他工业控制器上的一种通用语言。通过此协议，各控制器之间可以实现串行通信，Modbus通信协议定义了一个控制器能识别使用的消息结构，描述了主控制器访问从站设备的过程，例如规定从站怎样做出应答响应，检查和报告传输错误等。Modbus协议的通信方式为主从方式。主站首先向从站设备发送通信请求指令，从节点根据请求指令中的功能码向主站发回回答数据。网络中的每个从站设备都必须分配给一个唯一的地址，最多可达31个从站设备。通过多达24种总线命令实现主控制器与从站设备之间的信息交换。从站设备只执行发给自己的指令，对于其它从站地址开头的报文不作应答。这种一问一答的通信模式，大大提高了通信的正确率。因其具有操作简单、高效、通信可靠等优点，Modbus协议已成为一个国际通信标准，得到了国际上大多数工控产品生产厂家的支持。该通信协议已广泛应用于机械、水利、电力、环保等行业设备中。

ModbusTCP通信协议可供自动化设备的监控使用。常见的应用是开发基于该协议的网关，通过网关可以将PLC、I/O模块和其它总线连到以太网上。Modbus TCP是在不改变原有的Modbus协议基础上，只是将其作为应用层协议简单的移植到TCP/IP协议上。Modbus TCP协议每一个呼叫都要求一个应答。利用TCP/IP协议，通过网页的形式可以使用户界面更加友好。利用网络浏览器就可以查看企业网内部的设备运行情况。Schneider公司已经为Modbus注册了502端口，这样就可以将实时数据嵌入到网页中，通过在设备中嵌入Web服务器，就可以将Web浏览器作为设备的操作终端。但是Modbus协议本身存在一些缺陷，它不支持诸如基于对象的通信模型等一些正在被广泛采用的网络新技术，用户在使用的时候，不得不手工配置一些参数，比如信息数据类型、寄存器号等等。

B.1 ModbusRTU协议数据格式

Modbus协议有ASCII(美国标准信息交换代码)和RTU(远程终端单元)两种数据传输方式可由用户选择，但在一个 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。其中 RTU模式信息帧中的 8位数据包括两个 4位 16进制字符，相对于 ASCII模式表达相同的信息只需较少的位数，在相同的速率下较ASCII模式具有更大的数据流量。因此，在通常情况下较多使用 RTU模式。GCAN-204 设备也采用 RTU 模式。

RTU模式消息发送至少以 3.5个字符间隔时间(如表B.1的T1-T2-T3-T4)标志开始和结束，信息帧由地址域、功能域、数据域和 CRC校验域构成，所有字符位由 16进制 0-9、A-F组成。整个消息帧必须作为一连续的流传输。如果在帧完成之前有超过 1.5个字符时间的停顿时间，接受设备将刷新不完整的消息并假定下一个字节是一个新消息的地址域。同样的，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。

起始位	设备地址	功能代码	数据	CRC校验	结束符
T1-T2-T3-T4	8Bit	8Bit	N个 8Bit	16Bit	T1-T2-T3-T4

表B.1 RTU消息帧格式

（1） 地址域

指定报文的目的地址，包括 8bit。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。地址 0 用作广播地址，以使所有的从设备都能认识。

（2） 功能域

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。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。

（3） 数据域

数据域是由两个十六进制数集合构成的，范围 00~FF。从主设备发给从设备消息的数据域包含从机执行主机功能代码中所需的参数，如处理对象的寄存器地址，要处理项的数目，域中实际数据字节数。举例说明，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 16，即 10H)，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 H)，从设备不需任何附加的信息。

当传送一个 **2** 个字节的数据时，高字节**(MSB)**将被首先传送，然后传送低字节**(LSB)**。这与 **DeviceNet** 的传送方式刚好相反。

（4） CRC校验域

CRC域检测整个消息的内容，包括两个字节，包含一个 16位的二进制值。它由传输设备计算后加入到消息中。接收设备将重新计算收到消息的 CRC，并与接收到的 CRC域中的值进行比较。如果两值不同，则有误。CRC添加到消息中时，低字节先加入，然后是高字节。

B.2 ModbusTCP协议数据格式

TCP/IP 协议和以太网的链路层校验机制已可保证数据包传递的正确性，因此ModbusTCP报文中不再存在 CRC-16或 LRC校验域，但需要添加一个 Modbus应用帧头(MBAP)。它可对 Modbus的参数及功能进行解释。每个 TCP/IP报文仅可含有一个 Modbus 帧。

在 ModbusTCPADU中，MBAP头部占 7个字节（含 4个子域），及交易标识符 TI(TransactionIdentifier)、协议标识符 PI(Protocol Identifier)，长度标识符 L(Length)(占用 2字节，指明 ProtocolIdentifier和 Data域的总长度)和单元标识符 UI(UnitIdentifier)组成。TI占用 2字节，用来标识 Modbus帧的次序，PI占用 2字节，用于确认应用层协议。UI占 1字节，用于标识 Modbus设备单元。功能码占 1字节，可分为位操作和 16位字操作两类。功能码指出要进行的操作，如功

能码 15代表写多个位寄存器，功能码 06表示对独立的 16位字寄存器进行写操作。数据域最多可达 248字节，其具体格式与功能码相关。当客户机发送请求数据时，数据域给出要操作的寄存器的起始地址（2 字节）和个数（1 字节）；当服务器发送应答数据时，数据域给出被操作的寄存器个数（1 字节）及各寄存器状态值。图 B.1 给出了 Modbus 与 Modbus TCP 数据帧格式比较。

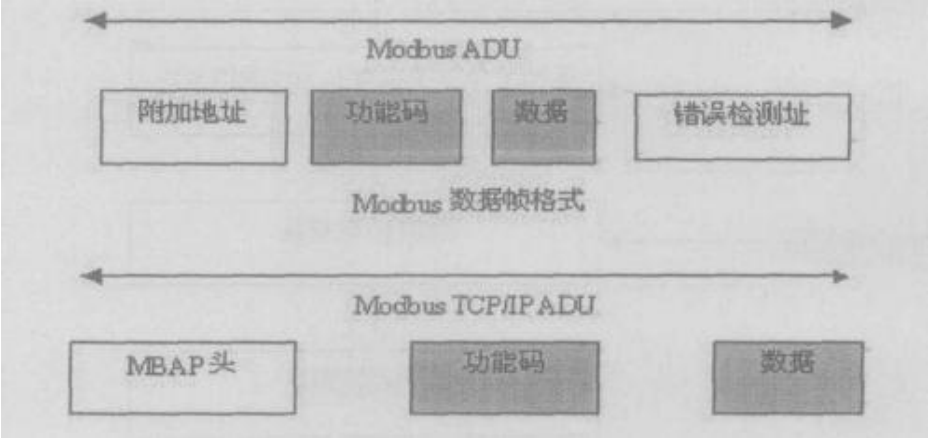


图 B.1Modbus与 ModbusTCP/IP帧格式

ModbusTCP的 ADU 数据单元规范如表 B.1所示。

	描述	所占字节
MBAP头	传输标识码高位 Hi	1
	传输标识码低位 Lo	1
	协议标识符	2
	长度标识符	2
	单元标识符	1
Modbus请求	功能码	1
	开始地址	2
	寄存器数目	2

表 B.2ModbusTCP的 ADU数据单元规范

在通过 Modbus TCP传送数据之前，需要在客户机和服务器之间建立一个TCP/IP连接。服务器使用端口 502作为 ModbusTCP的连接端口。ModbusTCP连接的建立通常由 TCP/IPSocket接口的软件协议自动实现，因此对应用完全透明。一旦客户端和服务器之间的 TCP/IP连接建立，同样的连接可以根据要求的方向用来传输任意数量的用户数据。客户端和服务器还可以同时建立多个TCP/IP 连接，最大的连接数量取决于 TCP/IP 接口的规范。

当某一设备发出请求，则其相应的设备要做出响应。响应的数据格式如表B.2所示。

字节	响应数据
Byte0、Byte1	传输标识码=0（响应时拷贝该数据）
Byte2、Byte3	协议标识符
Byte4	长度标识符高字节=0

Byte5	长度标识符低字节（标识其后有多少个字节）

Byte6	单元标识符（从设备地址）
Byte7	Modbus功能码
Byte8	数据

表 B.3ModbusTCP响应数据格式

B.3 Modbus常用功能码

在 Modbus消息帧的功能码中较常使用的是 01、02、03、04、05、06和 16 功能码，使用它们即可实现对从机的数字量和模拟量的读写操作。

Modbus标准地址与各个功能码的对应关系如下所示。

Modbus标准地址	数据	功能码
00001-0xxxx	DO	01、05、15
10001-1xxxx	DI	02
30001-3xxxx	AI	04
40001-4xxxx	保持寄存器	03、06、16

下面以在 RTU传输模式下通讯为例，对这些功能码进行详细介绍。

功能码	名称	功能说明
01	读取线圈状态	取得一组线圈的当前状态(ON/OFF)
02	读取输入状态	取得一组开关输入的当前状态(ON/OFF)
03	读取保持寄存器	在一个或多个保持寄存器中取得当前的二进制值
04	读取输入寄存器	在一个或多个输入寄存器中取得当前的二进制值
05	强置单线圈	强置一个逻辑线圈的通断状态
06	预置单寄存器	把具体二进制值装入一个保持寄存器
07	读取异常状态	取得 8 个内部线圈的通断状态
08	回送诊断校验	把诊断校验报文送从机，通信诊断
16	预置多寄存器	把具体二进制值装入一串连续的保持寄存器
128~255	保留	用于异常应答

下面是 7个 **ModbusRTU**命令的主从机收发的数据包格式，其余的命令可参照其格式。

（1） 功能码：**01H**

代码功能：读取线圈状态（DO）

说明：读取从机 DO的 ON/OFF状态，不支持广播。

查询：查询信息规定了要读的起始线圈地址和线圈量，线圈的起始地址为 0000H，1-16个线圈的寻址地址分为 0000H-0015H。

主机发送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	01	读取线圈状态
线圈首地址	2字节	00 00	线圈首址为 0000H
线圈数量	2字节	00 08	连续读 8个线圈
CRC	2字节	3DCC	前 6个字节的 CRC校验码

响应：响应信息中的各线圈的状态与数据区的每一位的值相对应，即每个DO占用一位（1=ON,0=OFF）。数据区从高位到低位依次为DO7、DO6.....DO0。

从机回送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	01	读取线圈状态
数据字节数	1字节	01	1个字节
数据	1字节	02	二进制为 00000010，DO1为ON
CRC	2字节	D049	前 4个字节的 CRC校验码

（2） 功能码：**02H**

代码功能：读取输入状态（DI）

说明：读取从机 DI的 ON/OFF状态，不支持广播。

查询：查询信息规定了要读的输入起始地址及输入信号的数量，输入寻址起始地址为 0000H，输入 1-16 所对应的地址分别为 0-15。

主机发送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	02	读取输入状态
输入首地址	2字节	00 00	输入首址为 0000H
寄存器数量	2字节	00 08	连续读 8个输入口
CRC	2字节	79 CC	前 6个字节的 CRC校验码

响应：响应信息中的各输入口的状态与数据区的每一位的值相对应，即每个DI占用一位（1=ON,0=OFF）。数据区从高位到低位依次为DI7、DI6.....DI0。

从机回送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	02	读取输入状态
数据字节数	1字节	01	1个字节
数据	1字节	81	二进制为 10000001，DI7与 DI0为 ON
CRC	2字节	61 E8	前 4个字节的 CRC校验码

（3） 功能码：**03H**

代码功能：读取保持寄存器

说明：读从机保持寄存器的二进制数据，不支持广播。

查询：查询信息规定了要读的寄存器起始地址及寄存器的数量，寄存器寻址起始地址为 0000H，寄存器 1-16 所对应的地址分别为 0-15。

主机发送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	03	读取保持寄存器数据
寄存器首地址	2字节	00 01	寄存器首址为 0001H
寄存器数量	2字节	00 03	连续读 3个寄存器
CRC	2字节	54 0B	前 6个字节的 CRC校验码

响应：响应信息中的寄存器数据为二进制数据，每个寄存器分别对应 2个字

节，第一个字节为高位值数据，第二个字节为低位数据。

从机回送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	03	读取保持寄存器数据
数据字节数	1字节	06	3个寄存器占 6个字节
数据 1	2字节	02 0B	0001H寄存器中的数据
数据 2	2字节	00 00	0002H寄存器中的数据
数据 3	2字节	00 64	0003H寄存器中的数据
CRC	2字节	84 BD	前 9个字节的 CRC校验码

（4） 功能码：**04H**

代码功能：读取输入寄存器（AI）

说明：读取从机输入寄存器(3X类型)中的二进制数据，不支持广播。

查询：查询信息规定了要读的寄存器起始地址及寄存器的数量，寄存器寻址起始地址为 0000H，寄存器 1-16 所对应的地址分别为 0-15。

主机发送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	04	读取输入寄存器数据
寄存器首地址	2字节	00 00	寄存器首址为 0000H
寄存器数量	2字节	00 01	连续读 1个寄存器
CRC	2字节	31 CA	前 6个字节的 CRC校验码

响应：响应信息中的寄存器数据为二进制数据，每个寄存器分别对应 2个字节，第一个字节为高位值数据，第二个字节为低位数据。

从机回送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	04	读取输入寄存器数据
数据字节数	1字节	02	1个寄存器占 2个字节
数据 1	2字节	0FFB	0000H寄存器中的数据
CRC	2字节	FD43	前 5个字节的 CRC校验码

（5） 功能码：**05H**

代码功能：强置单线圈（DO）

说明：强制单个线圈(DO，0X类型)为 ON或 OFF 状态，广播时，该功能可强制所有从机中同一类型的线圈均为 ON 或 OFF 状态。

查询：查询信息规定了需要强制线圈的地址及状态，线圈的起始地址为 0000H，寄存器 1-16 所对应的地址分别为 0-15。查询时，由查询数据区中的一个常量，规定被请求线圈的 ON/OFF状态，FF00H 值请求线圈处于 ON 状态，0000H 值请求线圈处于 OFF 状态，其它值对线圈无效，不起作用。

主机发送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	05	强置单线圈

线圈地址	2字节	00 01	线圈地址为 0001H
线圈状态值	2字节	FF00	ON状态
CRC	2字节	DDFA	前 6个字节的 CRC校验码

响应：对这个命令请求的正常响应是在 DO状态改变以后，原样传送接收到的数据。

从机回送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	05	强置单线圈
线圈地址	2字节	00 01	线圈地址为 0001H
线圈状态值	2字节	FF00	ON状态
CRC	2字节	DDFA	前 6个字节的 CRC校验码

（6） 功能码：**06H**

代码功能：预置单寄存器

说明：把一个值预置到一个保持寄存器（4X 类型）中，广播时，该功能把值预置到所有从机相同类型的寄存器中。该功能可越过控制器的内存保护。使寄存器中的预置值保持有效。只能由控制器的下一个逻辑信号来处理该预置值。若控制逻辑中无寄存器程序时，则寄存器中的值保持不变。

查询：查询信息规定了要预置寄存器的类型，寄存器寻址起始地址为 0000H，寄存器 1-16 所对应的地址分别为 0-15。

主机发送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	06	读寄存器数据
寄存器地址	2字节	00 03	预置寄存器地址为 0003H
寄存器的值	2字节	AB CD	将该值预置到寄存器中
CRC	2字节	C76F	前 6个字节的 CRC校验码

响应：对这个命令请求的正常响应是在寄存器值状态改变以后，原样传送接收到的数据。

从机回送	字节数	例（Hex）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	06	读寄存器数据
寄存器地址	2字节	00 03	预置寄存器地址为 0003H
寄存器的值	2字节	AB CD	将该值预置到寄存器中
CRC	2字节	C76F	前 6个字节的 CRC校验码

（7） 功能码：**10H**（十进制为 **16**）

代码功能：预置多个寄存器

说明：把数据按顺序预置到各(4x类型)寄存器中，广播时该功能代码可把数据预置到全部从机中的相同类型的寄存器中。需要注意的是该功能代码可越过控制器的内存保护，在寄存器中的预置值一直保持有效，只能由控制器的下一个逻

辑来处理寄存器的内容，控制逻辑中无该寄存器程序时，则寄存器中的值保持不变。

查询：信息中规定了要预置的寄存器类型，寄存器寻址的起始地址为 0。查询数据区中指定了寄存器的预置值，M84和 484型控制器使用 10位二进制数据，2 个字节，剩余的高 6位置 0。而其他类型的控制器使用一个 16位二进制数据，每个寄存器 2个字节。

主机发送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	10	预置多个寄存器
寄存器首地址	2字节	10 20	写入寄存器首址为 1020H
寄存器数量	2字节	00 03	连续 3个寄存器
字节数	1字节	06	3个寄存器占 6个字节
数据 1	2字节	02 01	寄存器 1020H中的数据
数据 2	2字节	04 03	寄存器 1021H中的数据
数据 3	2字节	06 05	寄存器 1022H中的数据
CRC	2字节	BD 9B	前 13个字节的 CRC校验码

响应：正常响应返回从机地址、功能代码、起始地址和预置寄存器数。

从机回送	字节数	例（**Hex**）	注释
从机地址	1字节	01	与 01号从机通信
功能码	1字节	10	写寄存器数据
寄存器首地址	2字节	10 20	写入寄存器首址为 1020H
寄存器数量	2字节	00 03	连续 3个寄存器
CRC	2字节	85 02	前 6个字节的 CRC校验码

下面是 7 个 **Modbus TCP** 命令的主从机收发的数据包格式，其余的命令可参照其格式。本部分略去代码功能及说明，相关内容请参考 ModbusRTU部分。

（1） 功能码：**01H**

主机发送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	其后有 6个字节
从机地址	1字节	01	与 01号从机通信
功能码	1字节	01	读取线圈状态
线圈首地址	2字节	00 00	线圈首址为 0000H
线圈数量	2字节	00 08	连续读 8个线圈

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 04	其后有 4个字节
从机地址	1字节	01	与 01号从机通信

功能码	1字节	01	读取线圈状态

数据字节数	1字节	01	1个字节
数据	1字节	02	二进制为 00000010，DO1为ON

（2） 功能码：**02H**

主机发送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	02	读取输入状态
输入首地址	2字节	00 00	输入首址为 0000H
寄存器数量	2字节	00 08	连续读 8个输入口

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 04	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	02	读取输入状态
数据字节数	1字节	01	1个字节
数据	1字节	81	二进制为 10000001，DI7与DI0为ON

（3） 功能码：**03H**

主机发送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	03	读取保持寄存器数据
寄存器首地址	2字节	00 01	寄存器首址为 0001H
寄存器数量	2字节	00 03	连续读 3个寄存器

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 09	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	03	读取保持寄存器数据
数据字节数	1字节	06	3个寄存器占 6个字节
数据 1	2字节	02 0B	0001H寄存器中的数据
数据 2	2字节	00 00	0002H寄存器中的数据

数据 3	2字节	00 64	0003H寄存器中的数据

（4） 功能码：**04H**

主机发送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	04	读取输入寄存器数据
寄存器首地址	2字节	00 00	寄存器首址为 0000H
寄存器数量	2字节	00 01	连续读 1个寄存器

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 05	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	04	读取输入寄存器数据
数据字节数	1字节	02	1个寄存器占 2个字节
数据 1	2字节	0FFB	0000H寄存器中的数据

（5） 功能码：**05H**

主机发送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	05	强置单线圈
线圈地址	2字节	00 01	线圈地址为 0001H
线圈状态值	2字节	FF00	ON状态

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	05	强置单线圈
线圈地址	2字节	00 01	线圈地址为 0001H
线圈状态值	2字节	FF00	ON状态

（6） 功能码：**06H**

主机发送	字节数	例（**Hex**）	注释

传输标识	2字节	00 00	

协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	06	读寄存器数据
寄存器地址	2字节	00 03	预置寄存器地址为 0003H
寄存器的值	2字节	AB CD	将该值预置到寄存器中

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	06	读寄存器数据
寄存器地址	2字节	00 03	预置寄存器地址为 0003H
寄存器的值	2字节	AB CD	将该值预置到寄存器中

（7） 功能码：**10H**（十进制为 **16**）

主机发送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 0D	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	10	预置多个寄存器
寄存器首地址	2字节	10 20	写入寄存器首址为 1020H
寄存器数量	2字节	00 03	连续 3个寄存器
字节数	1字节	06	3个寄存器占 6个字节
数据 1	2字节	02 01	寄存器 1020H中的数据
数据 2	2字节	04 03	寄存器 1021H中的数据
数据 3	2字节	06 05	寄存器 1022H中的数据

从机回送	字节数	例（**Hex**）	注释
传输标识	2字节	00 00	
协议标识	2字节	00 00	
数据长度	2字节	00 06	
从机地址	1字节	01	与 01号从机通信
功能码	1字节	10	写寄存器数据
寄存器首地址	2字节	10 20	写入寄存器首址为 1020H
寄存器数量	2字节	00 03	连续 3个寄存器

9.免责声明

感谢您购买广成科技的 GCAN系列软硬件产品。GCAN是沈阳广成科技有限公司的注册商标。本产品及手册为广成科技版权所有。未经许可，不得以任何形式复制翻印。在使用之前，请仔细阅读本声明，一旦使用，即被视为对本声明全部内容的认可和接受。请严格遵守手册、产品说明和相关的法律法规、政策、准则安装和使用该产品。在使用产品过程中，用户承诺对自己的行为及因此而产生的所有后果负责。因用户不当使用、安装、改装造成的任何损失，广成科技将不承担法律责任。

关于免责声明的最终解释权归广成科技所有。

销售与服务

沈阳广成科技有限公司

地址：辽宁省沈阳市浑南区长青南街 135-21号 5楼

邮编：110000

网址：www.gcgd.net

淘宝官方店：https://shop72369840.taobao.com/

天猫官方店：https://gcan.tmall.com/

京东官方店：https://mall.jd.com/index-684755.html

官方服务热线：13019325660

售前服务电话与微信号：15712411229

售前服务电话与微信号：18309815706

售前服务电话与微信号：18940207426

售后服务电话与微信号：15566084645

售后服务电话与微信号：18609810321

售后服务电话与微信号：17602468871