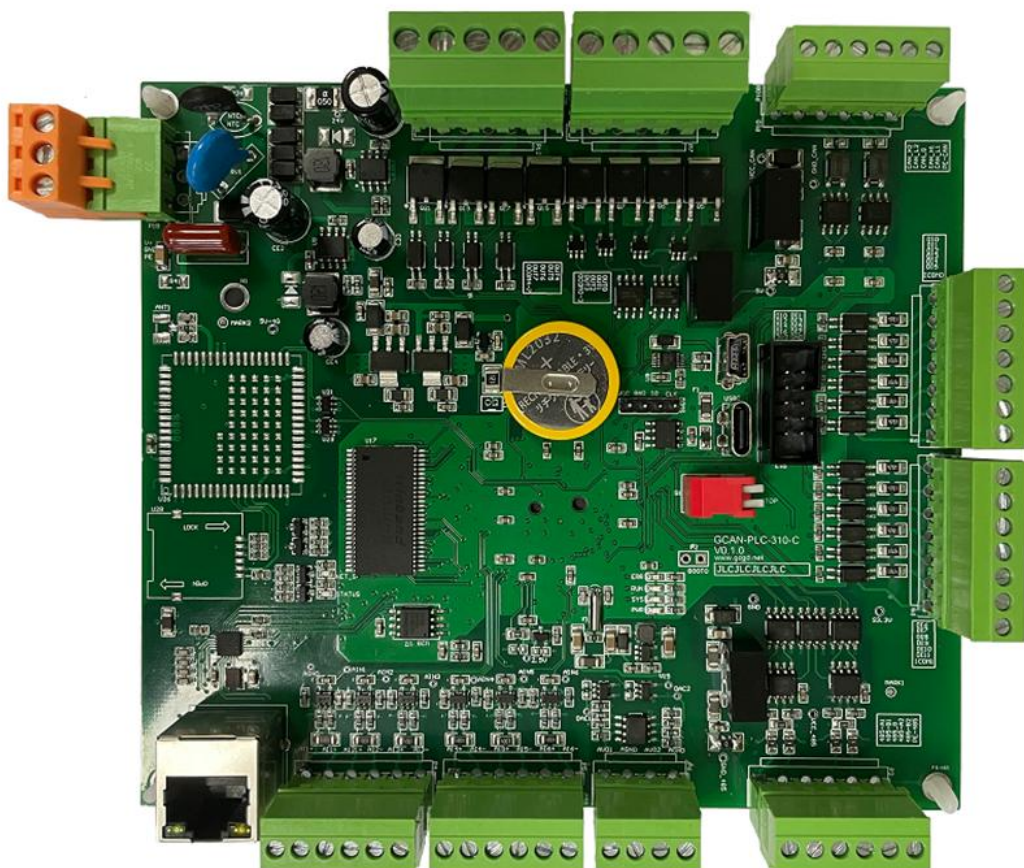


# GCAN-PLC

## 可编程逻辑控制器 (GCAN-PLC-310)

# 用户手册



文档版本: V1.01 (2024/08/13)

## 修订历史

| 版本    | 日期         | 原因          |
|-------|------------|-------------|
| V1.00 | 2023/03/01 | 创建文档        |
| V1.01 | 2024/08/13 | 问题修正，联系方式更新 |

# 目 录

|                             |    |
|-----------------------------|----|
| 1.功能简介 .....                | 4  |
| 1.1 功能概述 .....              | 4  |
| 1.2 硬件接口 .....              | 4  |
| 1.3 性能特点 .....              | 5  |
| 1.4 典型应用 .....              | 5  |
| 2.设备安装尺寸与使用 .....           | 6  |
| 2.1 模块外观及尺寸 .....           | 6  |
| 2.2 模块接口定义 .....            | 6  |
| 2.2.1 电源供电接口 .....          | 6  |
| 2.2.2 数字量输入（源/漏）接口 .....    | 7  |
| 2.2.3 数字量输出（PNP）接口 .....    | 9  |
| 2.2.4 模拟量输入接口 .....         | 10 |
| 2.2.5 模拟量输出接口 .....         | 11 |
| 2.2.6 RS485 总线接口 .....      | 12 |
| 2.2.7 CAN 总线接口 .....        | 13 |
| 2.3 系统指示灯 .....             | 14 |
| 3.通信连接 .....                | 15 |
| 3.1 串口连接 .....              | 15 |
| 3.2 CAN 连接 .....            | 15 |
| 3.3 CAN 总线终端电阻 .....        | 16 |
| 4. PLC 程序运行开关及复位按键 .....    | 17 |
| 5.技术规格 .....                | 18 |
| 附录 A: CANopen 协议简介 .....    | 20 |
| 附录 B: Modbus 协议简介 .....     | 27 |
| B.1 Modbus RTU 协议数据格式 ..... | 27 |
| B.2 Modbus TCP 协议数据格式 ..... | 28 |
| B.3 Modbus 常用功能码 .....      | 30 |
| 6.免责声明 .....                | 38 |
| 销售与服务 .....                 | 39 |

# 1.功能简介

## 1.1 功能概述

GCAN-PLC-310 是集成有总线控制功能的可编程逻辑控制器（PLC）。其不仅具有简约、高性价比的特点，还可以方便的连接入 CAN 总线系统及 Modbus 系统等。

GCAN-PLC-310 产品将微处理器、集成电源、输入电路和输出电路组合到一起，形成功能强大的 PLC。下载用户程序后，PLC 将包含监控应用中的输入和输出设备所需的逻辑。

GCAN-PLC-310 可使用 OpenPCS 软件或 CODESYS V3.5 SP16 及以上版本软件对其编程。软件支持符合 IEC 61131-3 标准中规定的五种标准编程语言，这使得程序的可移植性和复用性很强，而且软件还具有多种调试功能（断点、单步等），调试程序更加方便。

GCAN-PLC-310 不仅可完成各种数字/模拟量的输入/输出，还集成了多种常用的工业现场总线，如：CAN 总线、RS485 总线、以太网总线，并支持常见的通信协议如：CANopen、Modbus RTU、Modbus TCP 等。

## 1.2 硬件接口

GCAN-PLC-310 的 IO 接口包含 12 路数字量输入，8 路数字量输出，4 路模拟量输入，2 路模拟量输出；通讯接口包含 1 路以太网接口，2 路 485 接口，2 路 CAN 接口。

- 12 路源/漏型数字量输入 DI，输入电压 24V DC。输入电压最大值为 25V DC，输入电流最大值为 5mA（单通道）；
- 8 路晶体管 PNP 数字量输出 DO，输出电压 24V DC。输出电流最大值为 3A（单通道）；
- 4 路模拟量输入 AI，0V~+10V，映像位宽 16Bit（0~65535）；
- 2 路模拟量输出 AO，0V~+10V，映像位宽 12Bit（0~4095）；

### 1.3 性能特点

- 1路以太网接口，2路CAN总线接口，2路RS485接口；
- 编程软件：OpenPCS / CODESYS（编程语言符合IEC 61131-3标准）；
- 电源采用 24V DC（-15%/+20%）；输入电流最大 1.5A；
- 电气隔离为 1500 VDC；
- 工作温度范围：-40℃~+85℃；
- 防护等级：IP20；

### 1.4 典型应用

- 工业自动化控制核心
- 模拟量闭环控制系统
- 开关量开环控制，逻辑运算
- 工业 CAN 总线、ModBUS 总线网络通讯

## 2.设备安装尺寸与使用

本章节将详细说明 GCAN-PLC-310 的安装尺寸、接口的含义与指示灯状态的含义。

### 2.1 模块外观及尺寸

GCAN-PLC-310 控制器外观如图 2.1 所示：(长)141.60mm\*(宽)125.02mm。IO 接口包含 12 路数字量输入，8 路数字量输出，4 路模拟量输入，2 路模拟量输出；通信接口包含 1 路以太网接口，2 路 485 接口，2 路 CAN 接口。其中以太网接口，可用作以太网通讯及控制器编程。

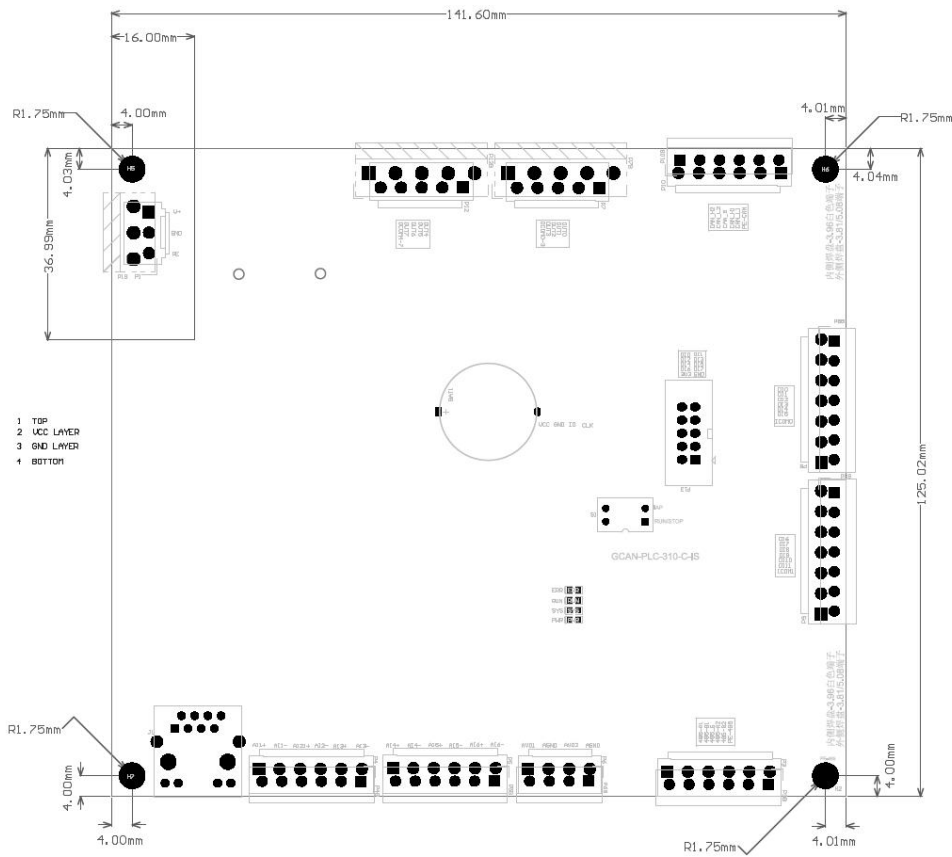


图 2.1 GCAN-PLC-310 外观尺寸图

### 2.2 模块接口定义

#### 2.2.1 电源供电接口

GCAN-PLC-310 主控模块的电源供电接口引脚定义如下图及下表。

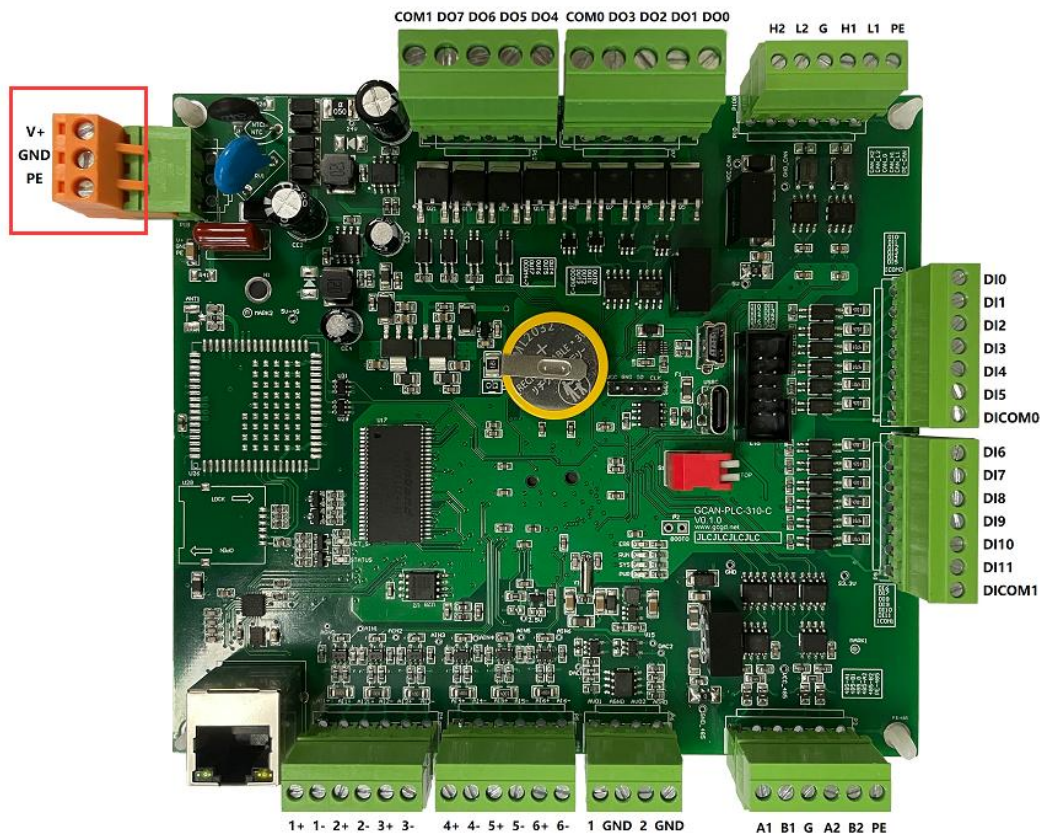


图 2.2 GCAN-PLC-310 的电源供电接口样式

| 接口定义（从上到下） | 含义         |
|------------|------------|
| V+         | 直流24V电源输入正 |
| GND        | 直流24V电源输入负 |
| PE         | -          |

表 2.1 GCAN-PLC-310 的电源接口定义

### 2.2.2 数字量输入（源/漏）接口

GCAN-PLC-310 主控模块的数字量输入接口引脚定义如下图及下表。

其中 COM0 为 DI0~DI5 前 6 路公共端；COM1 为 DI6~DI11 后 6 路公共端；当公共 COM 端接入低电平时，公共端对应的该组 DI 可作为源型数字量输入使用；当公共 COM 端接入高电平时，公共端对应的该组 DI 可作为漏型数字量输入使用。

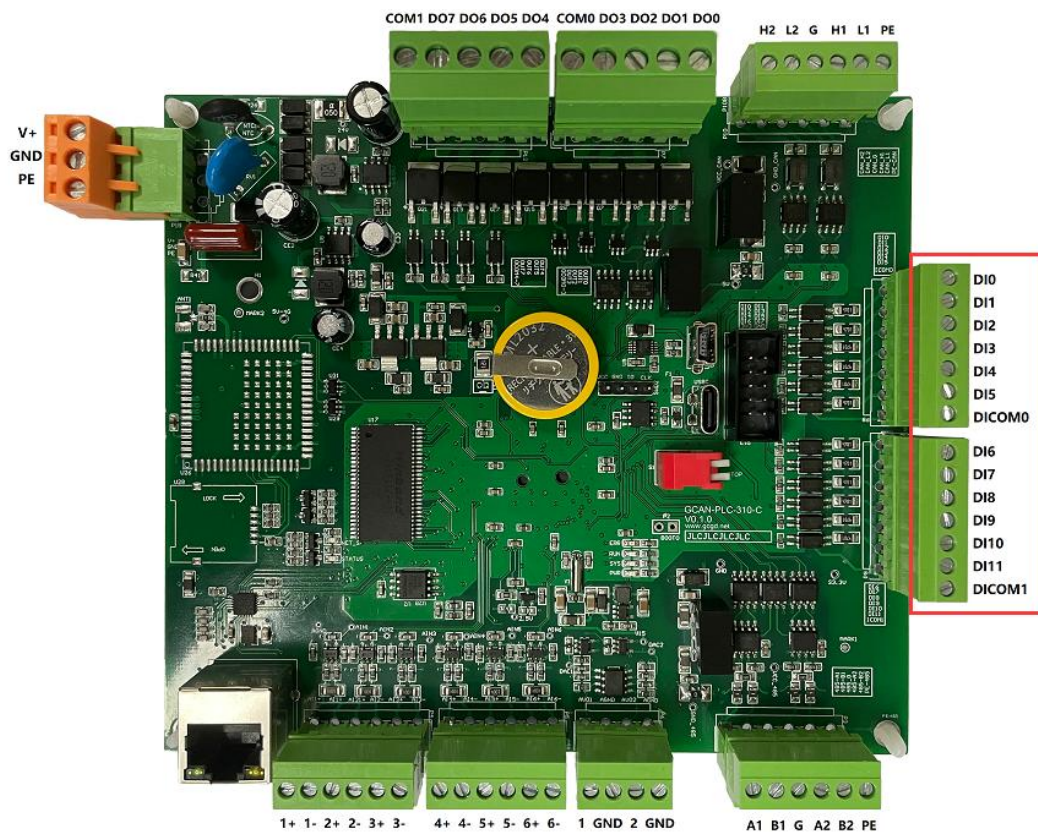


图 2.3 GCAN-PLC-310 的数字量输入 DI 接口样式

| 接口定义（从上到下） | 含义               |
|------------|------------------|
| DI0        | 数字量输入0-组态地址I0.0  |
| DI1        | 数字量输入1-组态地址I0.1  |
| DI2        | 数字量输入2-组态地址I0.2  |
| DI3        | 数字量输入3-组态地址I0.3  |
| DI4        | 数字量输入4-组态地址I0.4  |
| DI5        | 数字量输入5-组态地址I0.5  |
| DICOM0     | DI0~DI5公共端       |
| DI6        | 数字量输入6-组态地址I0.6  |
| DI7        | 数字量输入7-组态地址I0.7  |
| DI8        | 数字量输入8-组态地址I1.0  |
| DI9        | 数字量输入9-组态地址I1.1  |
| DI10       | 数字量输入10-组态地址I1.2 |
| DI11       | 数字量输入11-组态地址I1.3 |
| DICOM1     | DI6~DI11公共端      |

表 2.2 GCAN-PLC-310 的数字量输入 DI 接口定义及 OpenPCS 软件地址

### 2.2.3 数字量输出（PNP）接口

GCAN-PLC-310 主控模块的数字量输出接口引脚定义如下图及下表。

其中 COM0 为 DO0~DO3 前 4 路公共端；COM1 为 DO4~DO7 后 4 路公共端；数字量输出接口只能作为晶体管 PNP 输出，公共端需接入 DC 24V 正。

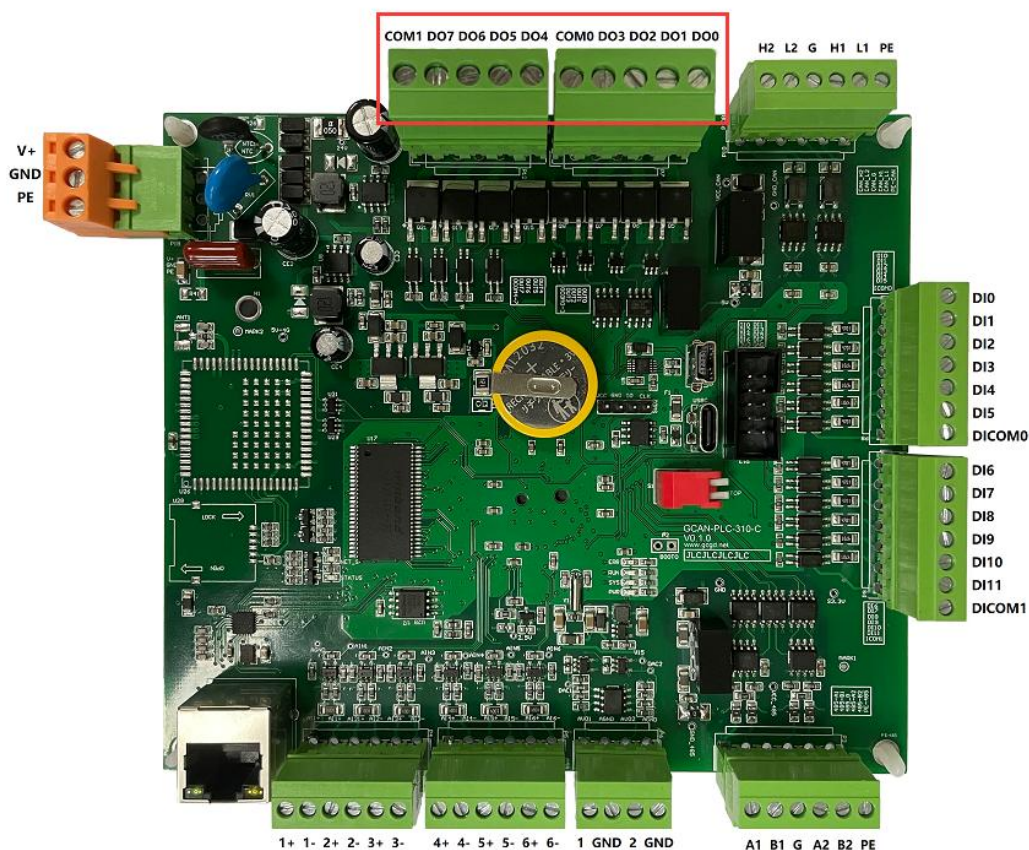


图 2.4 GCAN-PLC-310 的数字量输出 DO 接口样式

| 接口定义（从左到右） | 含义              |
|------------|-----------------|
| COM4-7     | DO4~DO7公共端      |
| DO7        | 数字量输出7-组态地址Q0.7 |
| DO6        | 数字量输出6-组态地址Q0.6 |
| DO5        | 数字量输出5-组态地址Q0.5 |
| DO4        | 数字量输出4-组态地址Q0.4 |
| COM0-3     | DO0~DO3公共端      |
| DO3        | 数字量输出3-组态地址Q0.3 |
| DO2        | 数字量输出2-组态地址Q0.2 |
| DO1        | 数字量输出1-组态地址Q0.1 |
| DO0        | 数字量输出0-组态地址Q0.0 |

表 2.5 GCAN-PLC-310 的数字量输出 DO 接口定义及 OpenPCS 软件地址

## 2.2.4 模拟量输入接口

GCAN-PLC-310 主控模块的模拟量输入接口引脚定义如下图及下表。输入范围为 0~10V，分辨率是 16 位，共 4 路。

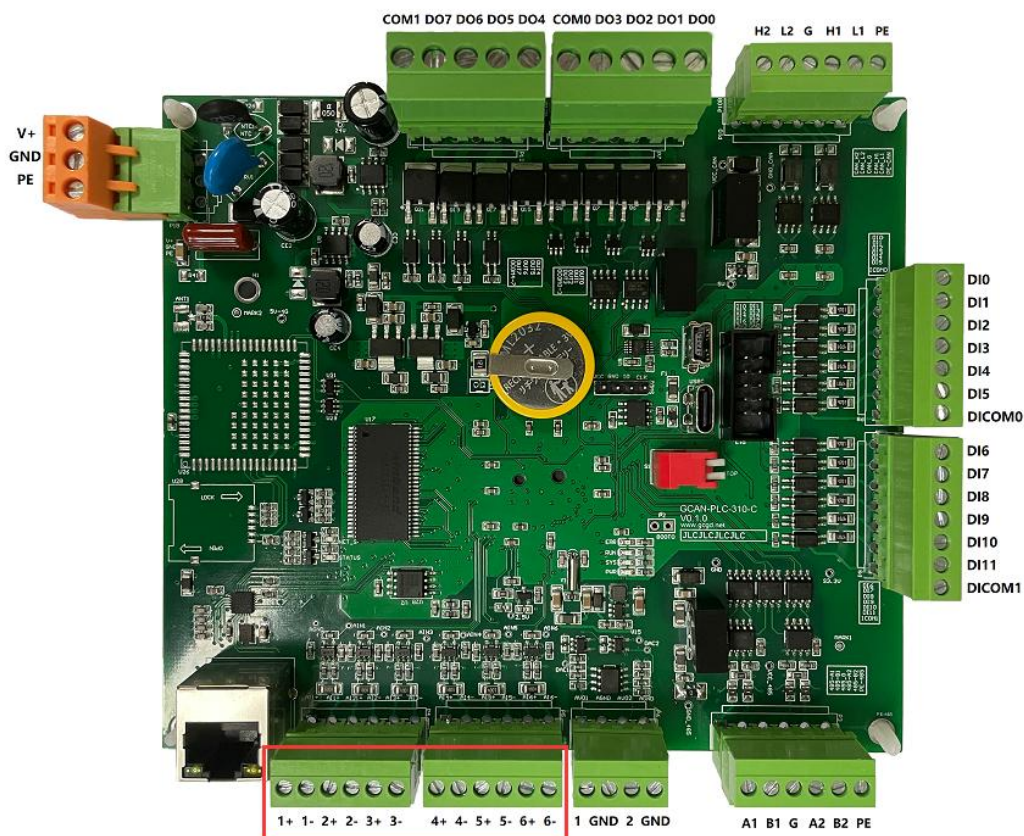


图 2.5 GCAN-PLC-310 的模拟量输入 AI 接口样式

| 接口定义（从左到右） | 含义                  | 地址长度 |
|------------|---------------------|------|
| AI1+       | 第1路模拟量输入正-组态地址I4.0  | 2字节  |
| AI1-       | 第1路模拟量输入负           |      |
| AI2+       | 第2路模拟量输入正-组态地址I6.0  | 2字节  |
| AI2-       | 第2路模拟量输入负           |      |
| AI3+       | 第3路模拟量输入正-组态地址I8.0  | 2字节  |
| AI3-       | 第3路模拟量输入负           |      |
| AI4+       | 第4路模拟量输入正-组态地址I10.0 | 2字节  |
| AI4-       | 第4路模拟量输入负           |      |
| AI5+       | -                   | -    |
| AI5-       | -                   | -    |

|      |   |   |
|------|---|---|
| AI6+ | - | - |
| AI6- | - | - |

表 2.6 GCAN-PLC-310 的模拟量输入 AI 接口定义及 OpenPCS 软件地址

## 2.2.5 模拟量输出接口

GCAN-PLC-310 主控模块的模拟量输出接口引脚定义如下图及下表。输出范围为 0~10V，分辨率是 12 位，共 2 路。

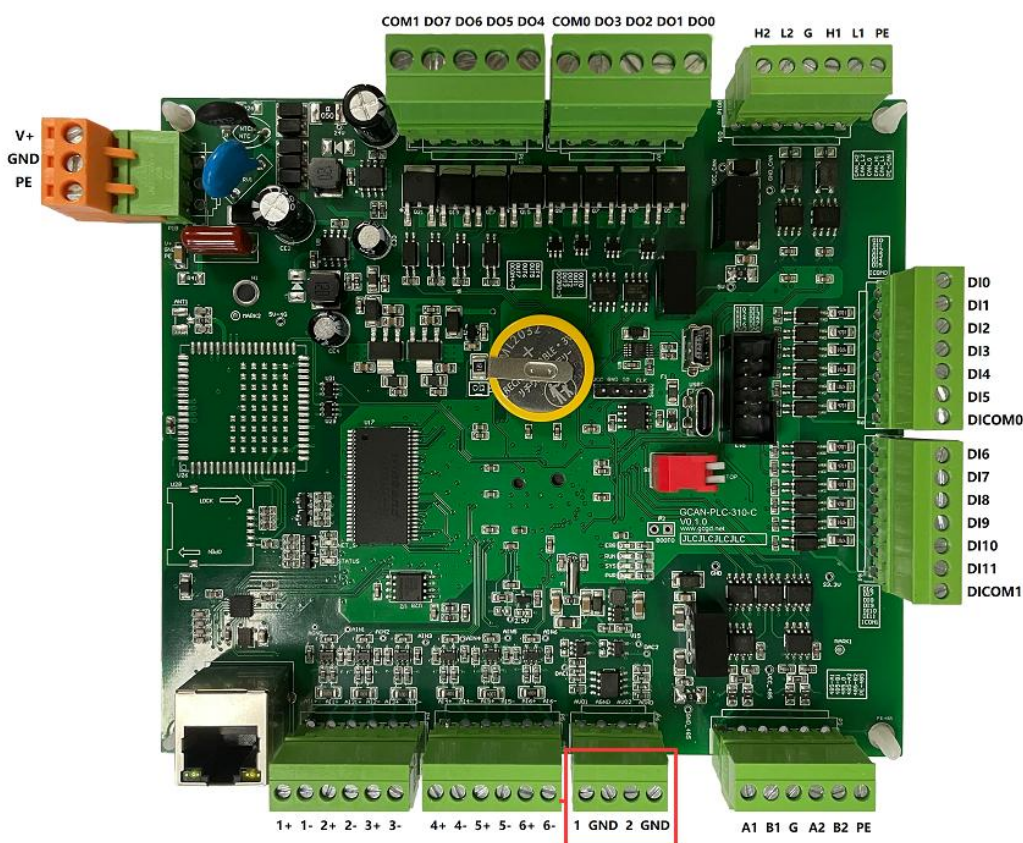


图 2.6 GCAN-PLC-310 的模拟量输出 AO 接口样式

| 接口定义（从左到右） | 含义                 | 地址长度 |
|------------|--------------------|------|
| AO1        | 第1路模拟量输出正-组态地址Q4.0 | 2字节  |
| AGND1      | 第1路模拟量输出负          |      |
| AO2        | 第2路模拟量输出正-组态地址Q6.0 | 2字节  |
| AGND2      | 第2路模拟量输出负          |      |

表 2.7 GCAN-PLC-310 的模拟量输出 AO 接口定义及 OpenPCS 软件地址

## 2.2.6 RS485 总线接口

GCAN-PLC-310 主控模块的 RS485 总线接口引脚定义如下图及下表。支持自由协议、Modbus RTU 主/从站协议。

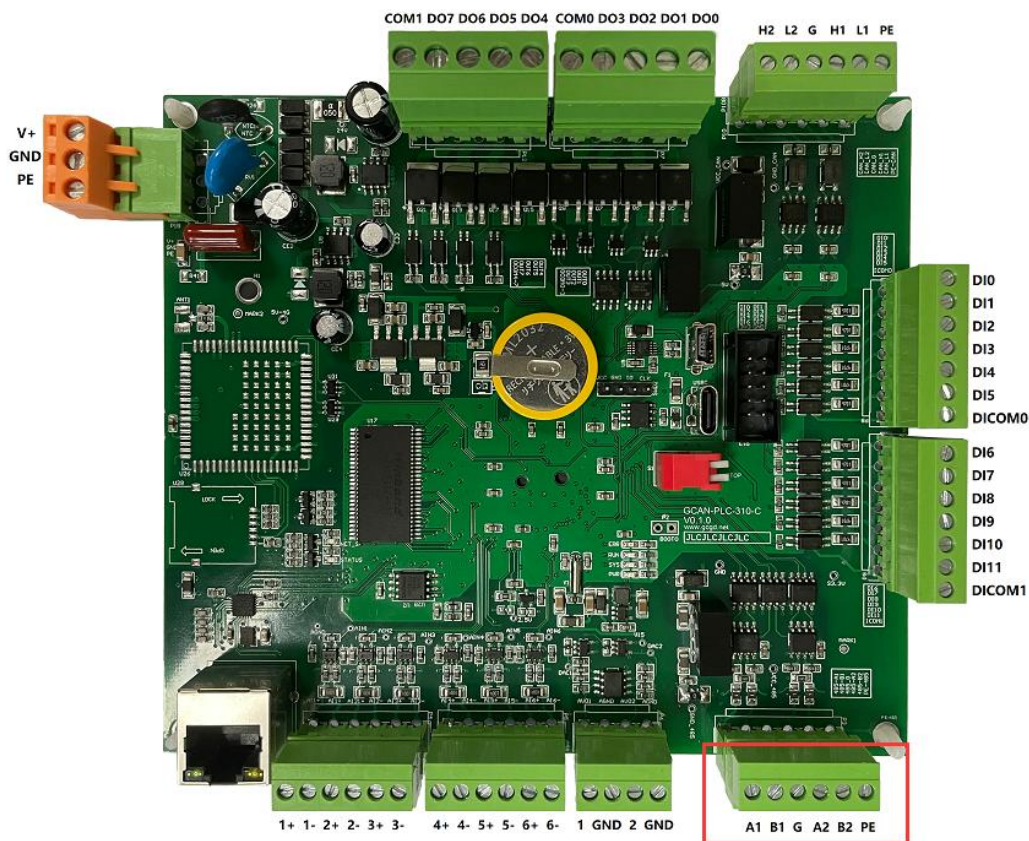


图 2.7 GCAN-PLC-310 的 RS485 接线图示

| 接口定义（从左到右） | 含义        |
|------------|-----------|
| 485_A1     | 第1路485_A1 |
| 485_B1     | 第1路485_B1 |
| 485_G      | GND       |
| 485_A2     | 第2路485_A1 |
| 485_B2     | 第2路485_B1 |
| 485_PE     | PE        |

表 2.8 GCAN-PLC-310 的 RS485 总线接口定义

## 2.2.7 CAN 总线接口

GCAN-PLC-310 主控模块的 CAN 总线接口引脚定义如下图及下表。支持 CAN 2.0A、CAN 2.0B 自由协议，CANopen 主/从站协议。

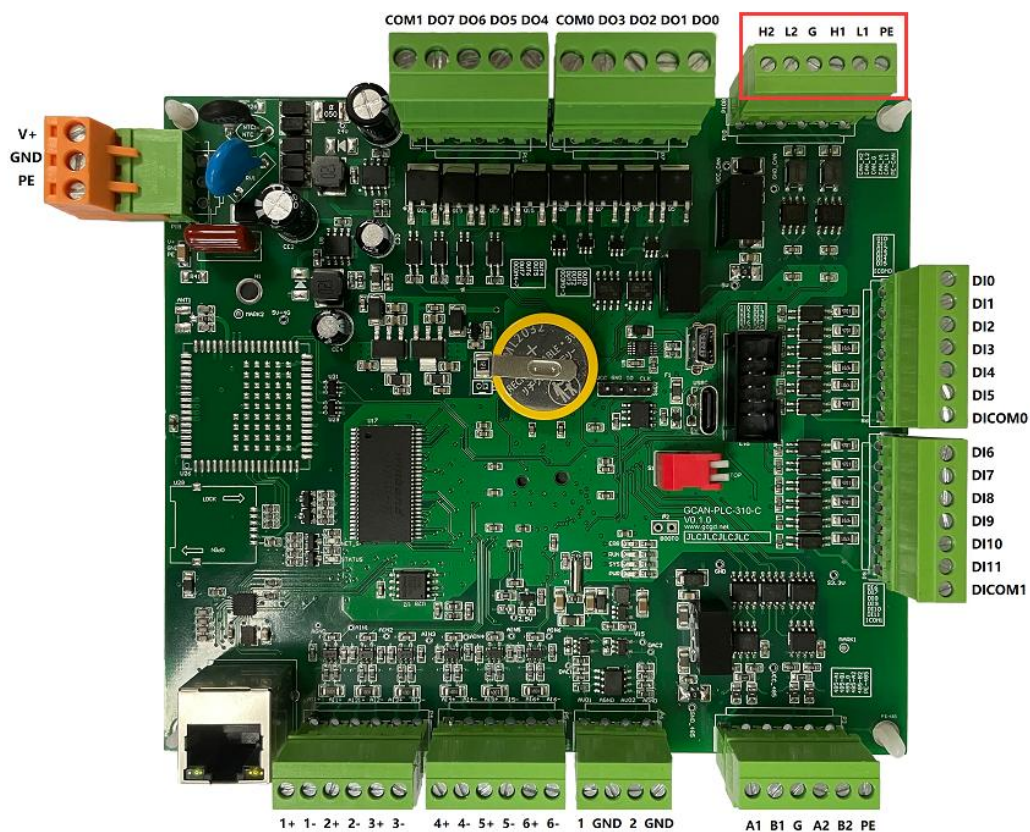


图 2.8 GCAN-PLC-310 的 CAN 总线接线图示

| 接口定义（从左到右） | 含义       |
|------------|----------|
| CAN_H2     | 第2路CAN_H |
| CAN_L2     | 第2路CAN_L |
| CAN_GND    | GND      |
| CAN_H1     | 第1路CAN_H |
| CAN_L1     | 第1路CAN_L |
| CAN_PE     | PE       |

表 2.9 GCAN-PLC-310 的 CAN 总线接口定义

## 2.3 系统指示灯

GCAN-PLC-310包含4个系统状态指示灯。指示灯的具体指示功能见表2.10。

| 系统状态指示灯 | 颜色 | 状态 | 状态含义    |
|---------|----|----|---------|
| ERR     | 红  | 不亮 | 无错误     |
|         |    | 常亮 | 运行错误    |
| RUN     | 绿  | 不亮 | 用户程序未运行 |
|         |    | 闪烁 | 用户程序运行  |
| SYS     | 绿  | 其他 | 系统错误    |
|         |    | 闪烁 | 系统正常运行  |
| PWR     | 绿  | 不亮 | 供电异常    |
|         |    | 常亮 | 系统供电正常  |

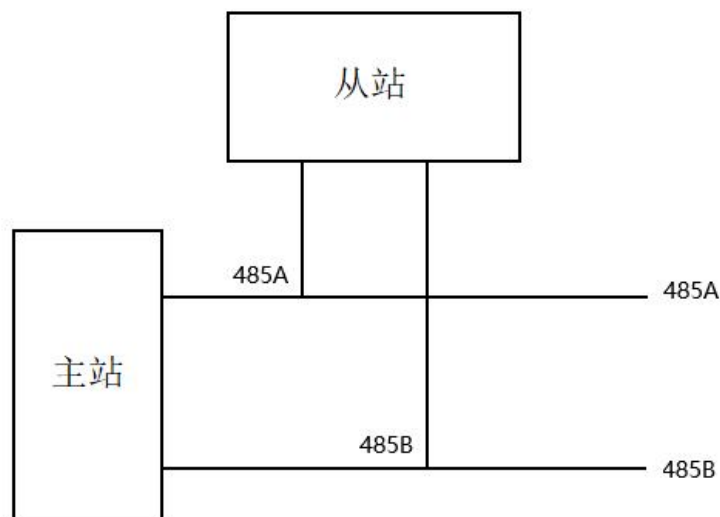
表 2.10 GCAN-PLC-310 系统指示灯定义

## 3.通信连接

### 3.1 串口连接

GCAN-PLC-310模块使用标准串口电平，因此该模块可以直接与带有RS485接口的设备进行连接。

RS485总线的连接如图所示：



### 3.2 CAN 连接

GCAN-PLC-310模块接入CAN总线时仅需要将CAN\_H连CAN\_H，CAN\_L连CAN\_L 即可建立通信。

CAN-bus网络采用直线拓扑结构，总线最远的2个终端需要安装120Ω的终端电阻；如果节点数目大于2，中间节点不需要安装120Ω的终端电阻。对于分支连接，其长度不应超过3米。CAN-bus总线的连接如图3.1所示。

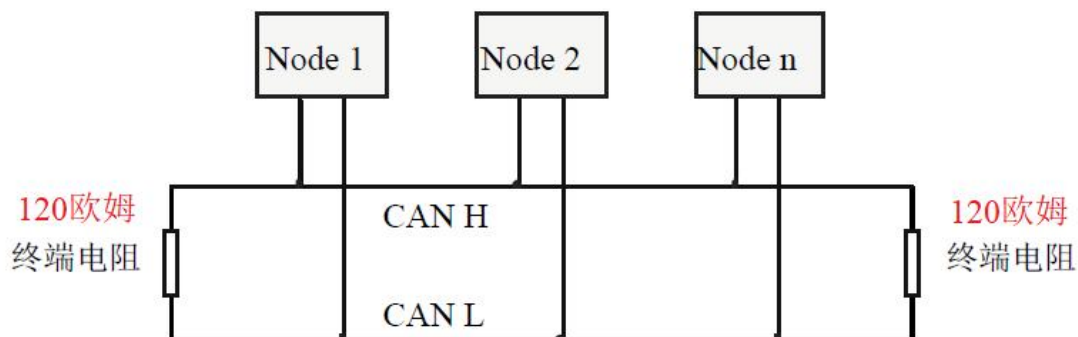


图 3.1 CAN-bus 网络的拓扑结构

请注意：CAN-bus电缆可以使用普通双绞线、屏蔽双绞线。理论最大通信距离主要取决于总线波特率，最大总线长度和波特率关系详见表3.1。若通讯距

离超过1km，应保证线的截面积大于 $\Phi 1.0\text{mm}^2$ ，具体规格应根据距离而定，常规是随距离的加长而适当加大。

| 波特率        | 总线长度  |
|------------|-------|
| 1 Mbit/s   | 25m   |
| 500 kbit/s | 100m  |
| 250 kbit/s | 250m  |
| 125 kbit/s | 500m  |
| 50 kbit/s  | 1.0km |
| 20 kbit/s  | 2.5km |
| 10 kbit/s  | 5.0km |
| 5 kbit/s   | 13km  |

表 3.1 波特率与最大总线长度参照表

### 3.3 CAN 总线终端电阻

为了增强CAN通讯的可靠性，消除CAN总线终端信号反射干扰，CAN总线网络最远的两个端点通常要加入终端匹配电阻，如图3.2所示。终端匹配电阻的值由传输电缆的特性阻抗所决定。例如双绞线的特性阻抗为 $120\Omega$ ，则总线上的两个端点也应集成 $120\Omega$ 终端电阻。如果网络上其他节点使用不同的收发器，则终端电阻须另外计算。

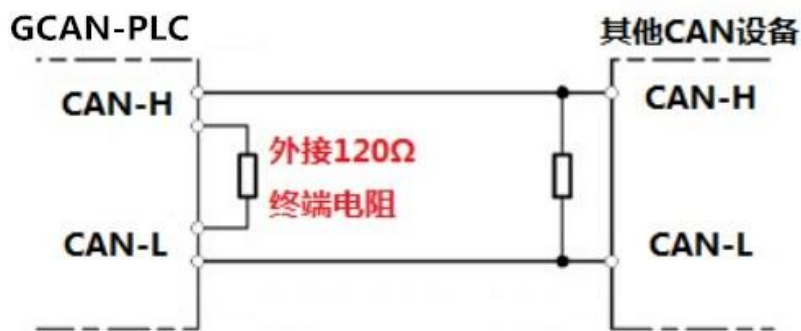


图 3.2 GCAN-PLC-310 模块与其他 CAN 节点设备连接

请注意：GCAN-PLC-310模块的CAN总线内部未集成 $120\Omega$ 终端电阻。如果节点数目大于2，中间节点不需要安装 $120\Omega$ 的终端电阻。需要使用时，将电阻两端分别接入CAN\_H、CAN\_L即可，如图3.2所示。

## 4. PLC 程序运行开关及复位按键

GCAN-PLC-310 提供 1 个复位开关和 1 个程序运行开关。开关位置如图 4.1 所示。

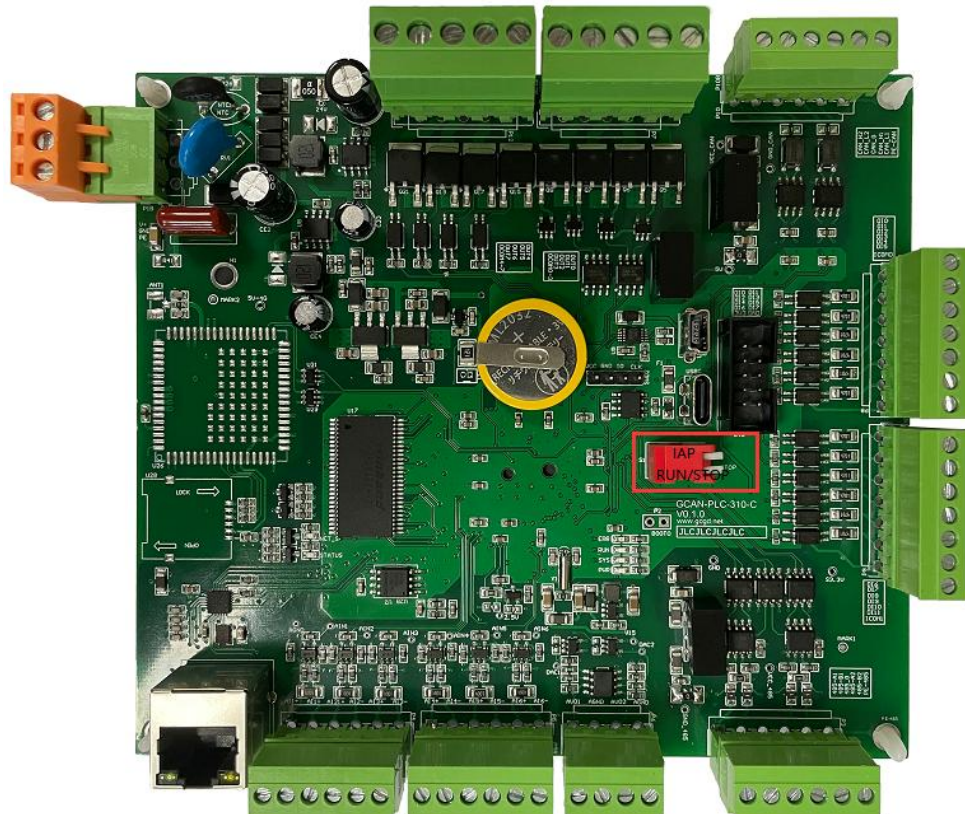


图 4.1 运行开关及复位键实物图

图 4.1 中：

- ① “IAP” 为复位拨码开关
- ② “RUN/STOP” 为运行/停止拨码开关。

**注意：**

- 1.运行/停止开关拨至 ON 端即 RUN 时，用于启动 PLC 程序运行；运行/停止按钮的拨码位于数字端即 STOP 位置时，用于停止 PLC 程序运行。
- 2.上电后复位开关位于 ON 端时 ERR 灯闪烁，用于恢复 PLC 出厂 IP 地址：**192.168.1.30**。

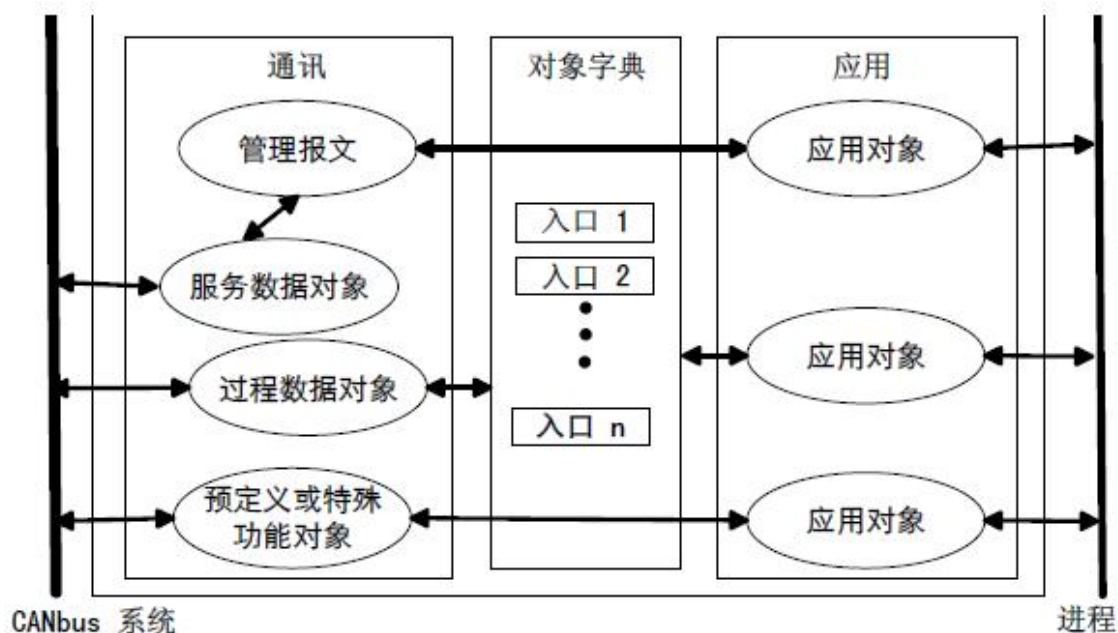
## 5.技术规格

|                         |                                                                                                                                                   |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>OpenPCS内核</b>        |                                                                                                                                                   |
| 程序存储空间                  | 8M 字节                                                                                                                                             |
| 数据存储空间                  | 256k 字节                                                                                                                                           |
| 掉电保存空间                  | 2k 字节                                                                                                                                             |
| <b>CodeSys内核</b>        |                                                                                                                                                   |
| 程序存储空间                  | 8M 字节                                                                                                                                             |
| 数据存储空间                  | 8M 字节                                                                                                                                             |
| 掉电保存空间                  | 4k 字节                                                                                                                                             |
| CANopen Master-3S库      | 支持                                                                                                                                                |
| CANopen RTU Master-3S库  | 支持                                                                                                                                                |
| CANopen Slave-GCAN库     | 支持                                                                                                                                                |
| Modbus RTU Master-GCAN库 | 支持                                                                                                                                                |
| Modbus RTU Slave-GCAN库  | 支持                                                                                                                                                |
| Modbus TCP Master-GCAN库 | 支持                                                                                                                                                |
| Modbus TCP Slave-GCAN库  | 支持                                                                                                                                                |
| <b>PLC参数</b>            |                                                                                                                                                   |
| Run-Time系统              | 多个PLC任务                                                                                                                                           |
| PLC周期时间                 | 1000条指令约需要0.5ms（I/O周期不计）                                                                                                                          |
| 程序在线修改                  | 不支持                                                                                                                                               |
| 编程语言执行标准                | IEC 61131-3                                                                                                                                       |
| 编程语言种类                  | SFC（顺序功能图）、LD（梯形图）、FBD（功能块）、ST（结构化文本）、IL（指令表）                                                                                                     |
| 本机I/O                   | 12路（源/漏型）DI 24V DC，单通道输入电压最大25V，单通道输入电流最大5mA；<br>8路（PNP型）DO 24V DC，单通道输出电流最大3A；<br>4路AI 0~10V 映像位宽16Bit（0~65535）；<br>2路AO 0~10V 映像位宽12Bit（0~4095） |
| 实时时钟                    | 内置                                                                                                                                                |
| 浮点数运算                   | 支持                                                                                                                                                |
| <b>通信接口特点</b>           |                                                                                                                                                   |
| 通信形式                    | 1路以太网接口，2路CAN接口， 2路RS485接口                                                                                                                        |
| CANopen 主/从站            | 支持                                                                                                                                                |
| Modbus RTU 主/从站         | 支持                                                                                                                                                |
| Modbus TCP 主/从站         | 支持                                                                                                                                                |
| <b>电气参数</b>             |                                                                                                                                                   |
| 电源                      | 24V DC                                                                                                                                            |
| 输入电流                    | 300mA（24V DC）                                                                                                                                     |
| 最大电流                    | 1.5A                                                                                                                                              |
| 电气隔离                    | 1500 V DC                                                                                                                                         |
| <b>环境试验</b>             |                                                                                                                                                   |

|               |                                     |
|---------------|-------------------------------------|
| 工作温度          | -40℃~+85℃                           |
| 工作湿度          | 95%RH, 无凝露                          |
| EMC测试         | EN 55024:2011-09 ; EN 55022:2011-12 |
| 抗振/抗冲击性能      | EN 60068-2-6 / EN 60068-2-27/29     |
| 抗电磁干扰/抗电磁辐射性能 | EN 61000-6-2 / EN 61000-6-4         |
| 防护等级          | IP 20                               |

## 附录 A: CANopen 协议简介

CANopen协议是在20世纪90年代末，由CiA组织（CAN-in-Automation）在CAL（CAN Application Layer）的基础上发展而来，一经推出便在欧洲得到了广泛的认可与应用。经过对CANopen协议规范文本的多次修改，使得CANopen协议的稳定性、实时性、抗干扰性都得到了进一步的提高。并且CiA在各个行业不断推出设备子协议，使CANopen协议在各个行业得到更快的发展与推广。目前CANopen协议已经在运动控制、车辆工业、电机驱动、工程机械、船舶海运等行业得到广泛的应用。



图A1 CANopen设备结构

图A1所示为CANopen设备结构，CANopen协议通常分为用户应用层、对象字典、以及通讯三个部分。

### A.1 相关名词解释和书写规则

#### 1. 名词解释：

PDO: Process Data Object, 过程数据对象。

TPDO: Transmit Process Data Object, 发送过程数据对象。

RPDO: Receive Process Data Object, 接收过程数据对象。

SDO: Service Data Object, 服务数据对象。

NMT: Network Management, 网络管理。

SYNC: Synchronization Objects, 同步报文对象。

EMCY: Emergency Objects, 紧急对象报文。

OD: Object Dictionary, 对象字典。

EDS: Electronic Data Sheet, 电子数据文档。

CAN-ID: Controller Area Network-Identify, 控制器局域网标识符。

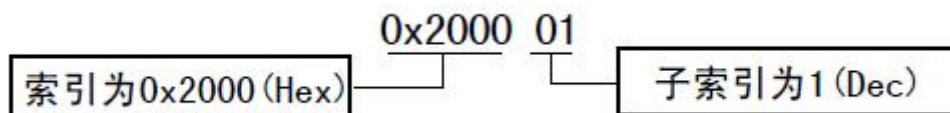
COB-ID: Communication Object-Identify, 通信对象标识符。

SSDO: Servers Service Data Object, 服务数据服务器。

DS: Draft Standard, 标准草案。

## 2. 书写规则

本手册中, 对象字典索引与子索引的书写遵循如下图A2所示的规则, 其中索引为16进制表示, 子索引为10进制表示, 索引与子索引中间用空格隔开。



图A2 索引/子索引书写规则

## A.2 预定义CAN标识符

| Object对象                        | 功能代码  | CAN-ID范围  |
|---------------------------------|-------|-----------|
| NMT网络管理命令                       | 0000b | 000h      |
| Sync同步报文                        | 0001b | 080h      |
| Time Stamp时间戳报文                 | 0010b | 100h      |
| Emergency紧急报文                   | 0001b | 081h-0FFh |
| TPDO1发送过程数据对象1                  | 0011b | 181h-1FFh |
| RPDO1接收过程数据对象1                  | 0100b | 201h-27Fh |
| TPDO2发送过程数据对象2                  | 0101b | 281h-2FFh |
| RPDO2接收过程数据对象2                  | 0110b | 301h-37Fh |
| TPDO3发送过程数据对象3                  | 0111b | 381h-3FFh |
| RPDO3接收过程数据对象3                  | 1000b | 401h-47Fh |
| TPDO4发送过程数据对象4                  | 1001b | 481h-4FFh |
| RPDO4接收过程数据对象4                  | 1010b | 501h-57Fh |
| SDO Server-to-Client 服务数据对象 (答) | 1011b | 581h-5FFh |
| SDO Client-to-Server 服务数据对象 (问) | 1100b | 601h-67Fh |
| NMT error control 网络管理错误控制      | 1110b | 701h-77Fh |

## A.3 CANopen对象字典

CANopen对象字典(OD: Object Dictionary)是CANopen协议最为核心的概念。所谓的对象字典就是一个有序的对象组, 每个对象采用一个16位的索引值来寻址, 这个索引值通常被称为索引, 其有效范围在0x1000到0x9FFF之间。为了允许访问数据结构中的单个元素, 同时也定义了一个8位的索引值, 这个索引值通常被称为子索引。每个CANopen设备都有一个对象字典, 对象字典包含了描述这

个设备和它的网络行为的所有参数，对象字典通常用电子数据文档（EDS: Electronic Data Sheet）来记录这些参数，而不需要把这些参数记录在纸上。对于CANopen网络中的主节点来说，不需要对CANopen从节点的每个对象字典项都访问。

CANopen对象字典中的项由一子协议来描述。子协议为对象字典中的每个对象都描述了它的功能、名字、索引、子索引、数据类型，以及这个对象是否必需、读写属性等等，这样可保证不同厂商的同类型设备兼容。CANopen协议的核心描述子协议是DS301，其包括了CANopen协议应用层及通信结构描述，其它子协议都是对DS301协议描述文本的补充与扩展。CANopen协议包含了许多子协议，其主要划分为以下类型。

#### 1. 通讯子协议（Communication Profile）

通讯子协议，描述对象字典的主要形式和对象字典中的通讯对象以及参数。这个子协议适用所有的CANopen设备，其索引值范围从0x1000~0x1FFF。

#### 2. 制造商自定义子协议（Manufacturer-specific Profile）

制造商自定义子协议，对于在设备子协议中未定义的特殊功能，制造商可以在此区域根据需求定义对象字典对象。因此这个区域对于不同的厂商来说，相同的索引的对象字典项定义不一定相同，其索引值范围为0x2000~0x5FFF。

#### 3. 设备子协议(Device Profile)

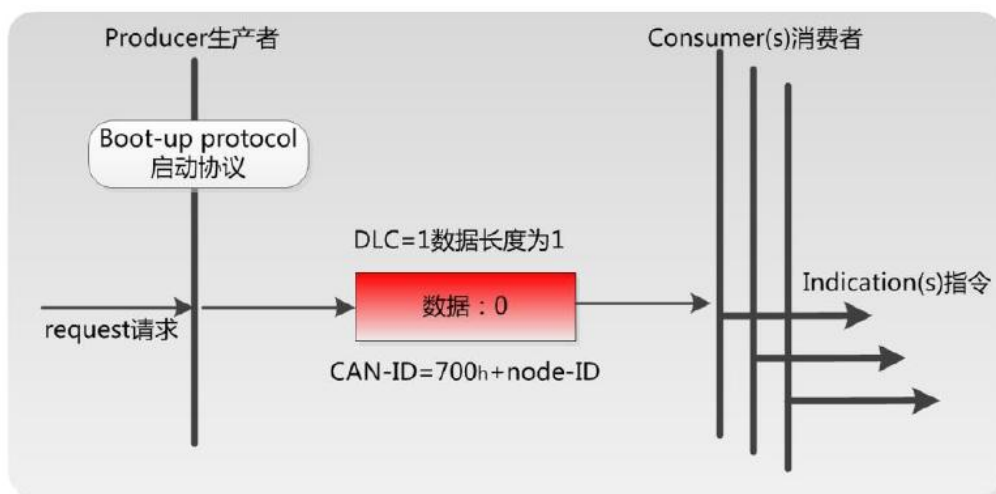
设备子协议，为各种不同类型的设备定义对象字典中的对象。目前已有十几种为不同类型的设备定义的子协议，例如DS401、DS402、DS406 等，其索引值范围为0x6000~0x9FFF。

### A.4 CANopen通讯

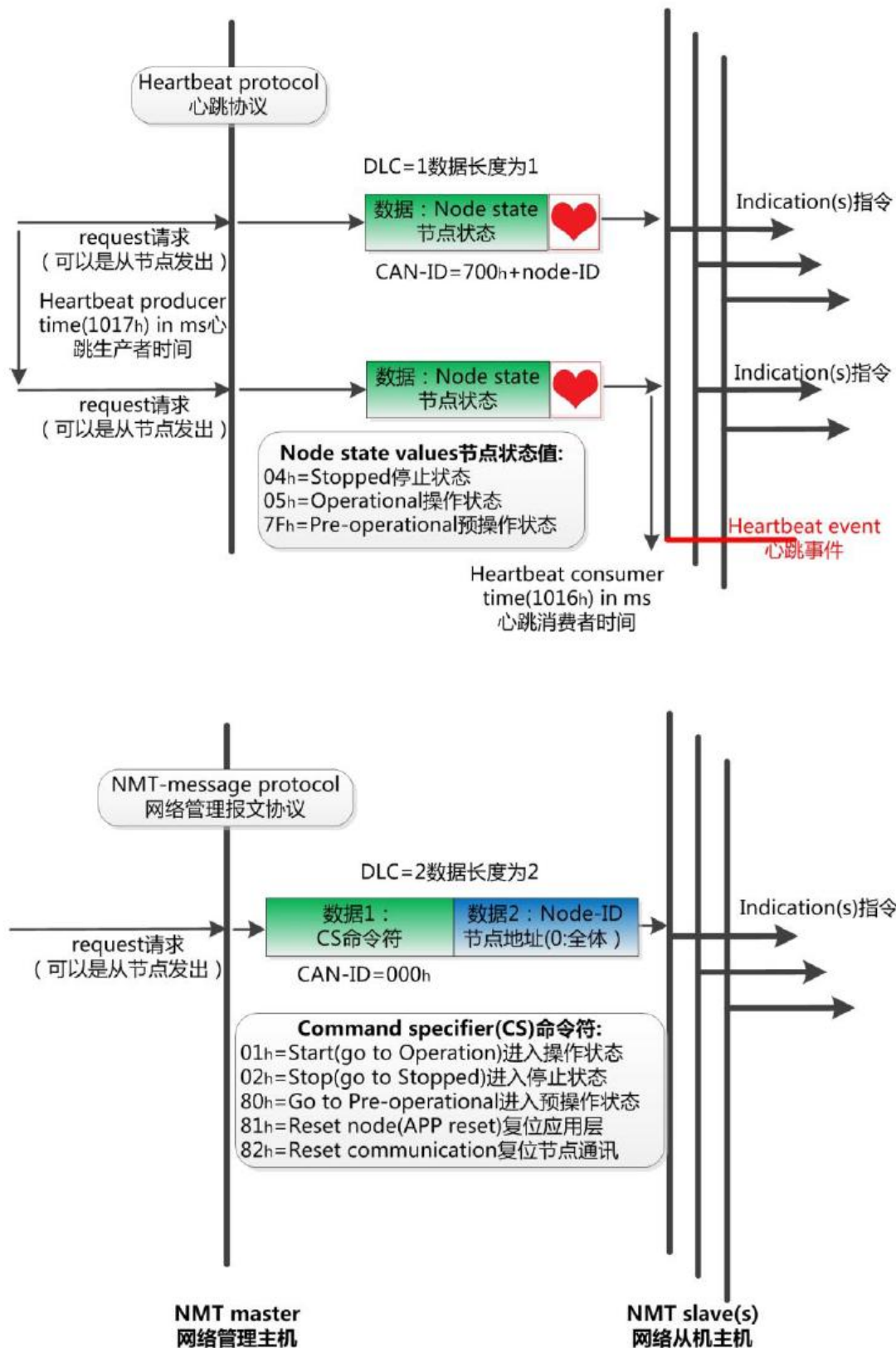
在CANopen协议中主要定义了管理报文对象NMT（Network Management）、服务数据对象SDO(Service Data Object)、过程数据对象PDO(Process Data Object)、预定义报文或特殊功能对象等四种对象。

#### 1. 网络管理NMT（Network Management）

管理报文负责层管理、网络管理和ID分配服务，例如，初始化、配置和网络管理（其中包括节点保护）。网络管理中，同一个网络中只允许有一个主节点、一个或多个从节点，并遵循主从模式。通过NMT服务，我们可以对节点进行初始化、运行、监控、复位和停止。所有节点都被认为是NMT从站。



如上图所示，举个例子，某 CANopen 从站设备上电之后将发送一个帧 ID 为 0x702，数据为 0x00 的数据；说明该设备已启动，且节点号为 2。

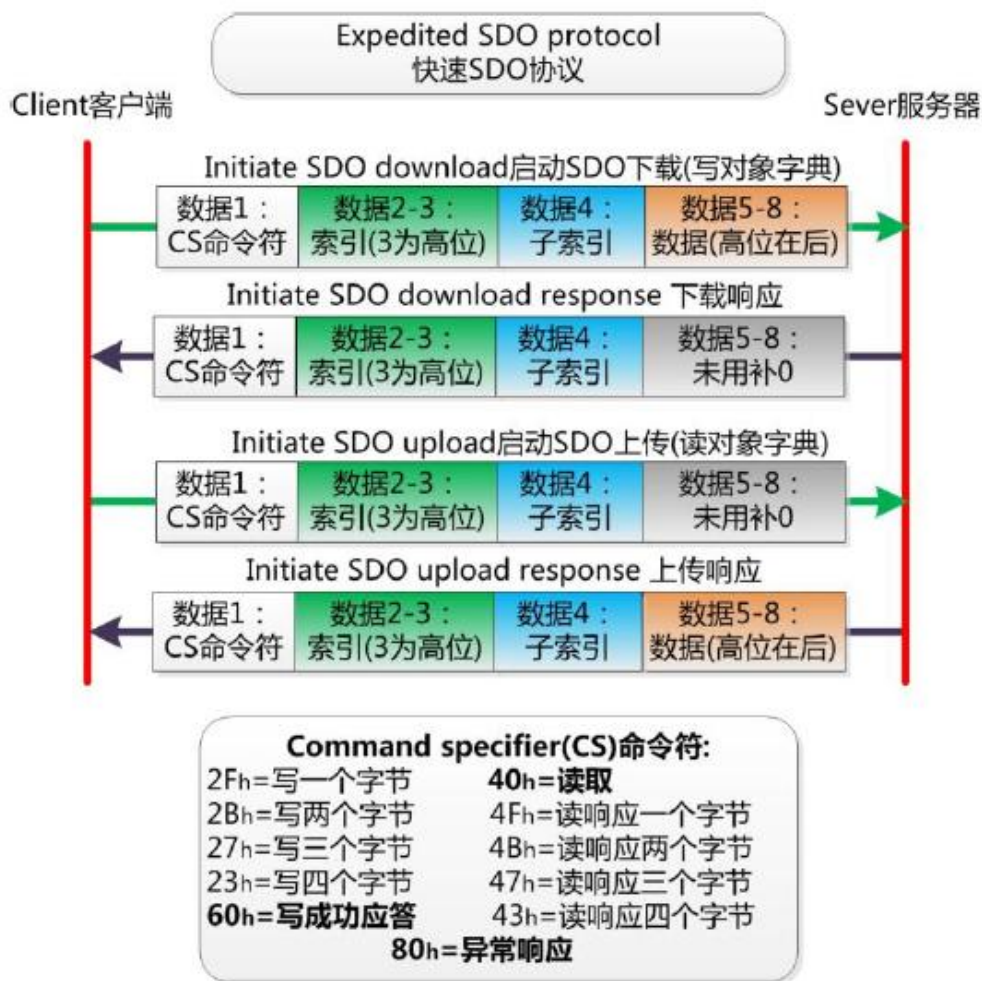


如上图所示，举个例子，某 CANopen 主站向从站发送一帧数据，帧 ID 为 0x000，帧数据为 0x01、0x02，则该指令可使节点号为 2 的 CANopen 从站设备

进入操作状态。

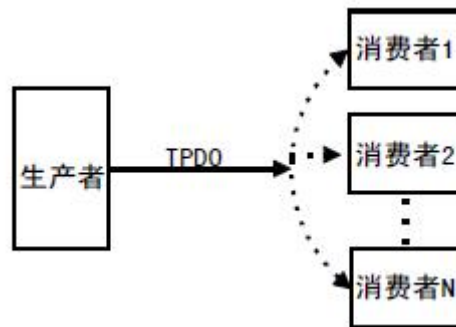
## 2. 服务数据对象SDO（Service Data Object）

SDO主要用于主节点对从节点的参数配置。服务确认是SDO的最大的特点，为每个消息都生成一个应答，确保数据传输的准确性。在一个CANopen系统中，通常CANopen从节点作为SDO服务器，CANopen主节点作为客户端。客户端通过索引和子索引，能够访问数据服务器上的对象字典。这样CANopen主节点可以访问从节点的任意对象字典项的参数，并且SDO也可以传输任何长度的数据（当数据长度超过4个字节时就拆分成多个报文来传输）。



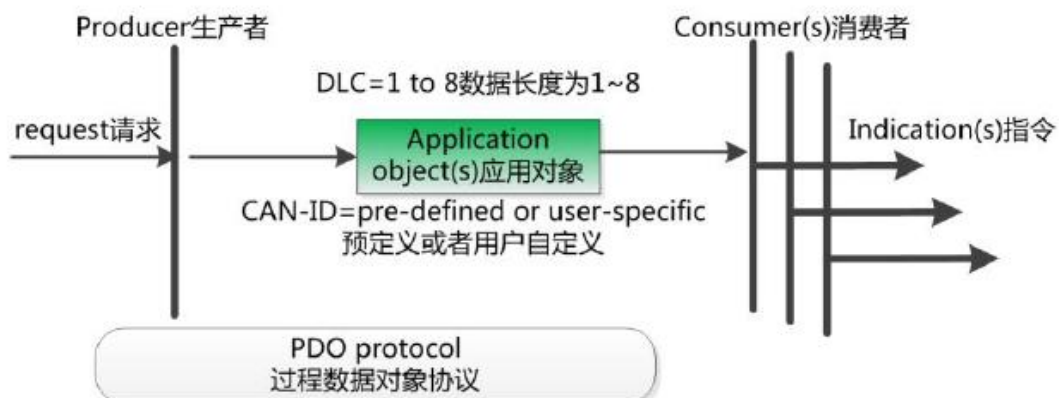
## 3. 过程数据对象PDO（Process Data Object）

PDO用来传输实时数据，其传输模型为生产者-消费者模型，如图A3所示。数据长度被限制为1~8字节。PDO通信对象具有如下的特点：



图A3 生产者消费者模型

- PDO通讯没有协议规定，PDO数据内容由它的CAN-ID（也可称为COB-ID）定义；
- 每个PDO在对象字典中用2个对象描述：
  - ◆ PDO通讯参数，该通讯参数定义了设备所使用的COB-ID、传输类型、定时周期；
  - ◆ PDO映射参数，映射参数包含了一个对象字典中的对象列表，这些对象映射到相应的PDO，其中包括数据的长度（单位：位），对于生产者和消费者都必须要知道这个映射参数，才能够正确的解释PDO内容。
- PDO消息内容是预定义的，如果PDO支持可变PDO映射，那么该PDO是可以通过SDO进行配置；
- PDO可以有多种的传输方式：
  - ◆ 同步传输（通过接收同步对象实现同步），同步传输又可分为非周期和周期传输。非周期传输是由远程帧预触发或者由设备子协议中规定的对象特定事件预触发传送。周期传输则是通过接收同步对象（SYNC）来实现，可以设置1~240个同步对象触发；
  - ◆ 异步传输（由特定事件触发），其触发方式可有两种，第一种是通过发送与PDO的COB-ID相同的远程帧来触发PDO的发送，第二种是由设备子协议中规定的对象特定事件来触发（例如，定时传输，数据状态变化传输等）。



#### 4. 预定义报文或特殊功能对象

预定义报文或特殊功能对象为CANopen设备提供特定的功能，方便CANopen主站对从站管理。在CANopen协议中，已经为特殊的功能预定义了

COB-ID, 其主要有以下几种特殊报文:

- 同步 (SYNC), 该报文对象主要实现整个网络的同步传输, 每个节点都以该同步报文作为PDO同步触发参数, 因此该同步报文的COB-ID具有较高的优先级以及最短的传输时间;
- 时间标记对象 (Time Stamp), 为各个节点提供公共的时间参考;
- 紧急事件对象 (Emergency), 当设备内部发生错误触发该对象, 即发送设备内部错误代码;
- 节点/寿命保护 (Node/Life Guarding), 主节点可通过节点保护方式获取从节点的状态。从节点可通过寿命保护方式获取主节点的状态;
- 启动报文对象 (Boot-up), 从节点初始化完成后向网络中发送该对象, 并进入到预操作状态。

### A.5 CANopen网络配置

在CANopen协议描述文本DS305中定义了一种网络配置协议即网络配置服务 LSS (Layer Setting Service), 其通过CAN总线, 用具有LSS 主机功能的CANOpen模块来查询或修改具有LSS 从机的CANOpen模块的某些参数。

通过使用LSS, 可以对下面的参数进行查询或修改:

- CANopen 从站的Node-ID;
- 物理层的位定时参数 (波特率);
- LSS地址 (特征对象1018h)。

## 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。

Modbus TCP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 Modbus RTU 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 Modbus TCP 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 Modbus TCP 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 Modbus TCP ADU 中，MBAP 头部占 7 个字节（含 4 个子域），及交易标识符 TI(Transaction Identifier)、协议标识符 PI(Protocol Identifier)，长度标识符 L(Length)(占用 2 字节，指明 Protocol Identifier 和 Data 域的总长度)和单元标识符

UI(Unit Identifier)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。

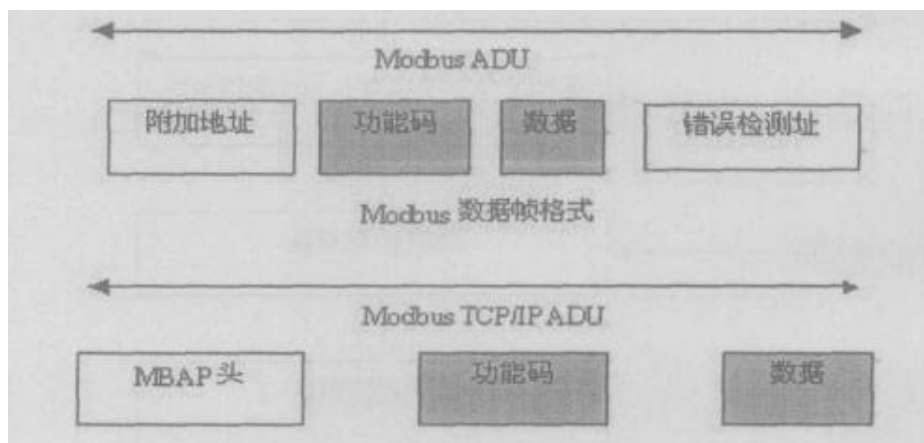


图 B.1 Modbus 与 Modbus TCP/IP 帧格式

Modbus TCP 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.2 Modbus TCP 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 Modbus TCP 的连接端口。Modbus TCP 连接的建立通常由 TCP/IP Socket 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.3 Modbus TCP 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 Modbus RTU 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 | 3D CC | 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 0000 0010, DO1 为 ON |
| CRC   | 2 字节 | D0 49   | 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 1000 0001, 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 | 0F FB   | 0000H 寄存器中的数据    |
| CRC   | 2 字节 | FD 43   | 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 | FF 00   | ON 状态            |
| CRC   | 2 字节 | DD FA   | 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 | FF 00   | ON 状态            |
| CRC   | 2 字节 | DD FA   | 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 | C7 6F   | 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 | C7 6F   | 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 Modbus RTU 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 0000 0010, 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 1000 0001, 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 | 0F FB   | 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 | FF 00   | 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 | FF 00   | 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      |

## 6.免责声明

感谢您购买广成科技的 GCAN 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

