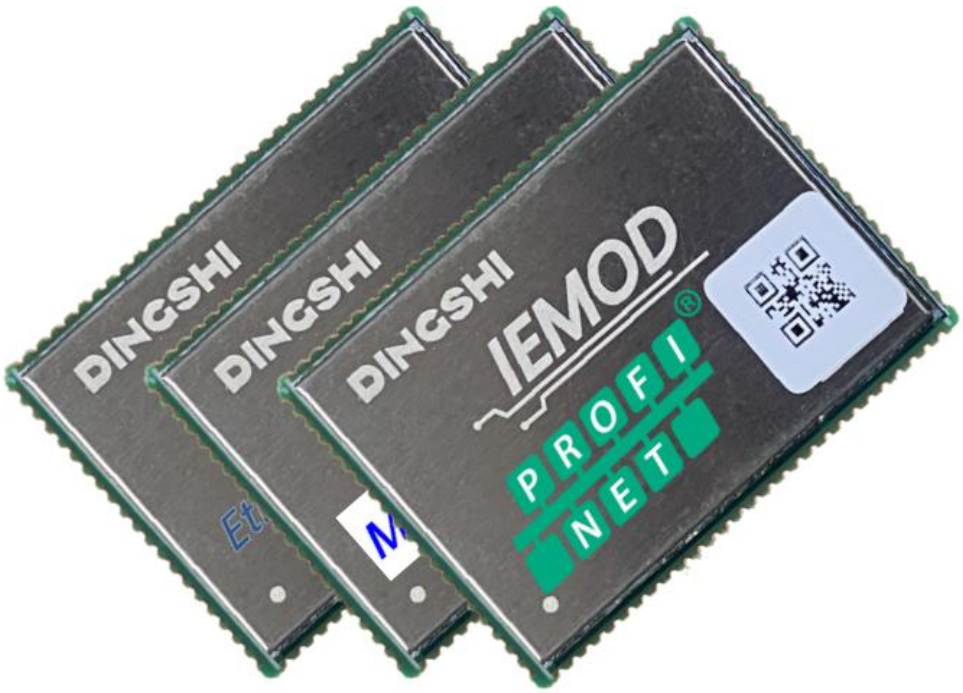
鼎实 IEMOD 系列产品-Profinet 从站

（V0.0.3）



北京鼎实创新科技股份有限公司

2022 年 9 月

目录

第一章 产品概述

1.1 产品简介

1.1.1 经验与业绩

鼎实有近 20 年的网络通信接口,嵌入式板卡技术产品经验，拥有几百个固定客户。产品应用遍布电力，能源，冶金，地铁，石油，化工，汽车制造，家电制造，电子制造，食品制造等行业。产品应用在国内所有地区，国际包括南亚，中亚，中东等许多国家。

1.1.2 支持国产替代

在“缺芯”的大变局之下，国产设备替代需求显得重要而迫切，在 PROFINET，ETHERNET/IP，MODBUS/TCP，CC-LINK IE，PROFIBUS-DP/PA 技术领域，鼎实在自主开发基于国产 MCU 的通信协议栈方面取得实质成果。现推出工业以太网多协议通信模组（DS-IEMOD）产品，为实现工业以太网设备国产化替代提供了自主解决方案。

1.1.3 模组系列产品

鼎实系列模组产品涵盖现有主流的工业以太网协议和工业现场总线协议，不需要用户了解各种工业通信协议实现快速接入主流控制器系统。模组具有使用简便，对用户硬件平台要求低，本地化支持，供货及时，支持定制功能开发。

所有模组封装相同，分为工业以太网模组和工业现场总线模组两大类，每一类具有完全相同的封装，便于用户实现单一用户硬件平台快速支持多种通信协议。对比板卡类产品具有集成度更高，方便产品升级为多协议产品等优势。

1.1.4 多协议以太网模组 DS-IEMOD

鼎实 DS-IEMOD 产品是多协议以太网通信模组，同样硬件（封装，管脚定义），通过固件升级可实现多种协议，包括:

一期（已开发）：从站：Profinet(RT)，Ethernet/IP，Modbus/TCP。

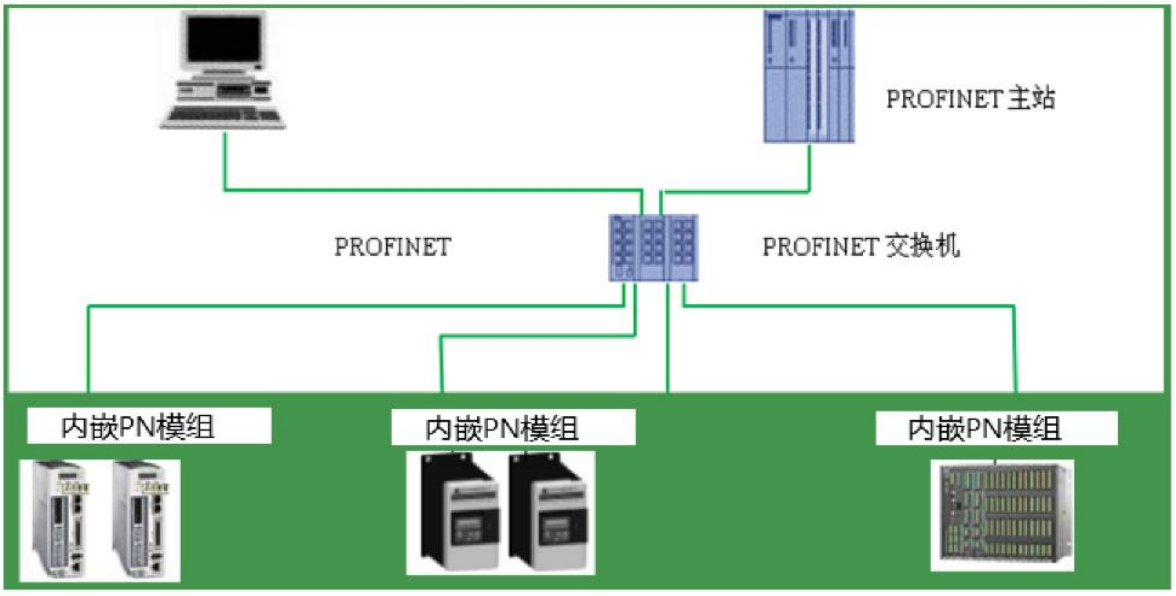
二期（开发中）：从站：Ethercat，Profinet(IRT/TSN)，Ethernet/IP(TSN)，CC-LINK IE(TSN)

主站：Profinet(Master)。

1.1.5 Profinet（RT）从站

Profinet（RT）从站，以下简称 PN 模组。

PN 模组+RJ45/M12 接头作为一个 PROFINET 从站,通过 SPI/UART/MEM 接口将用户产品接入 PROFINET 网络,与 PROFINET 网络实现互联互通。



1.2 PN 模组产品特点及优势

国际标准：PROFINET V2.42 一致性等级 CLASS B；

保证供货：不受国际芯片断货影响，采用国产通用 MCU+自主协议栈方案，供货更稳定；

自主研发：基于国产芯片的硬件设计，自主知识产权协议栈；

工业环境：采用工业级芯片，器件及工业等级软硬件设计，保证在工业环境下可靠运行；

多种协议：同一硬件，不同协议只需要固件升级；

快速上手：模组不需要用户复杂编程，编程调试一周内搞定；

技术服务：资深的产品经理对客户产品提供全生命周期的技术支持；

资料免费：鼎实提供详实的开发资料，提供用户侧开发例程，评估板原理图等，帮助用户最快开发产品；

1.3 PN 模组技术指标

1.3.1 PROFINET 指标

一致性等级	PROFINET V2 一致性等级 CLASS B；
IO DATA	Max Input Bytes ≤1440 Bytes； Max Output Bytes ≤1440 Bytes；
数据交换	RT 级 循环数据交换

1.3.2 用户接口指标

用户接口	SPI/UART/MEM;
接口速率	SPI 最高 25M;
固件升级	通过以太网接口;
握手信号	提高通信效率，节省等待时间;

1.3.3 硬件指标

模组封装	TQFP96 封装；
供电电压	DC 3.3V（3.1~3.5V）；
模块功耗	1.4W
工作温度	-20~+55℃
储存温度	-40~+70℃
工作湿度	0~95%无凝露
模块尺寸	38 mm x 28 mm x 3.7 mm（ L X W X H ）

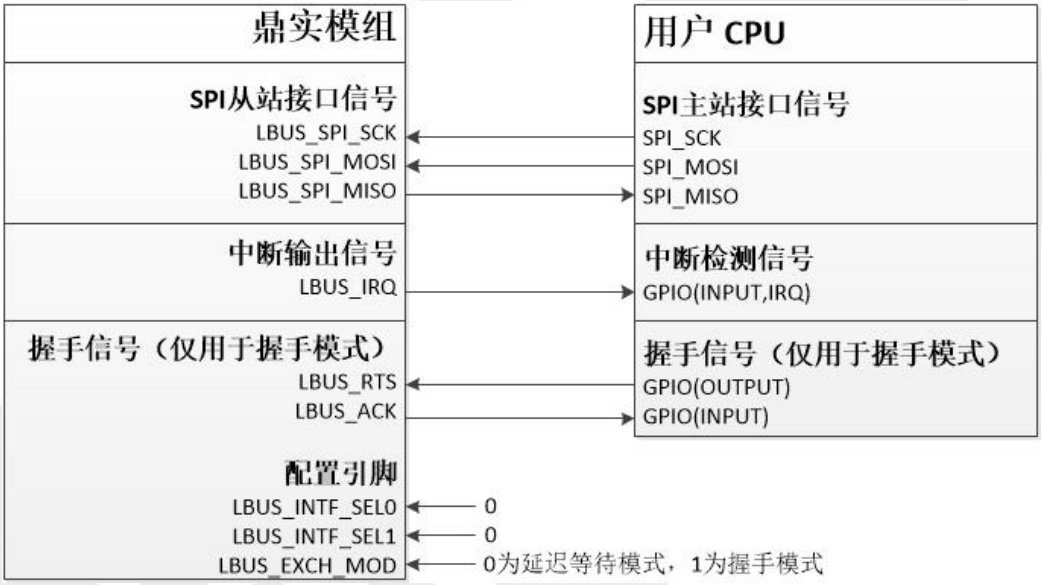
1.3.4 SPI 接口

模组 SPI 接口做为从设备，包括 LBUS_SPI_SCK，LBUS_SPI_MOSI，LBUS_SPI_MISO 三个信号。

SPI 接口参考设计

对于高速串行应用场合，SPI 是比较合适的通信接口。基于 SPI 接口使用模组，需要用到 SPI 标准信号 SPI_SCK，SPI_MOSI，SPI_MISO，不需要片选信号，中断检测信号 LBUS_IRQ，使用握手模式进行交互还需要 LBUS_RTS 和 LBUS_ACK 信号。参考连接方式见下图。

备注:目前仅支持 SPI 握手模式，LBUS-IRQ/配置模式引脚可以预留，也可以悬空。



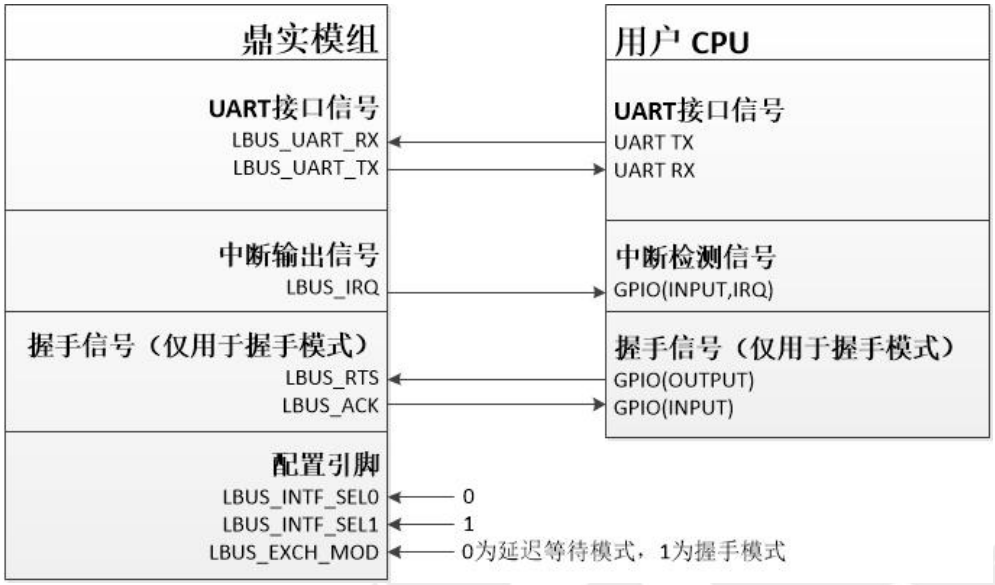
1.3.5 UART 接口

对于低速模组支持 3.3V 电平的 UART 接口。模组支持固定的 8 档波特率：9.6k，19.2k，38.4k，57.6k，115.2k，230.4k，460.8k，2250k。模组支持波特率自适应，上电初始化时，用户使用模组支持的几档波特率中的一种连续发送 0x68 字符，直到收到模组返回 0xE5 模组波特率自适应结束。模组的数据位，奇偶校验位，停止位固定为 0xE1。

UART 接口参考设计

基于 UART 接口使用模组，需要用到 UART 标准信号中的 TXD 和 RXD，不需流控信号，中断检测信号 LBUS_IRQ，使用握手模式进行交互还需要 LBUS_RTS 和 LBUS_ACK 信号。参考连接方式见下图。

备注:目前仅支持 SPI 握手模式，

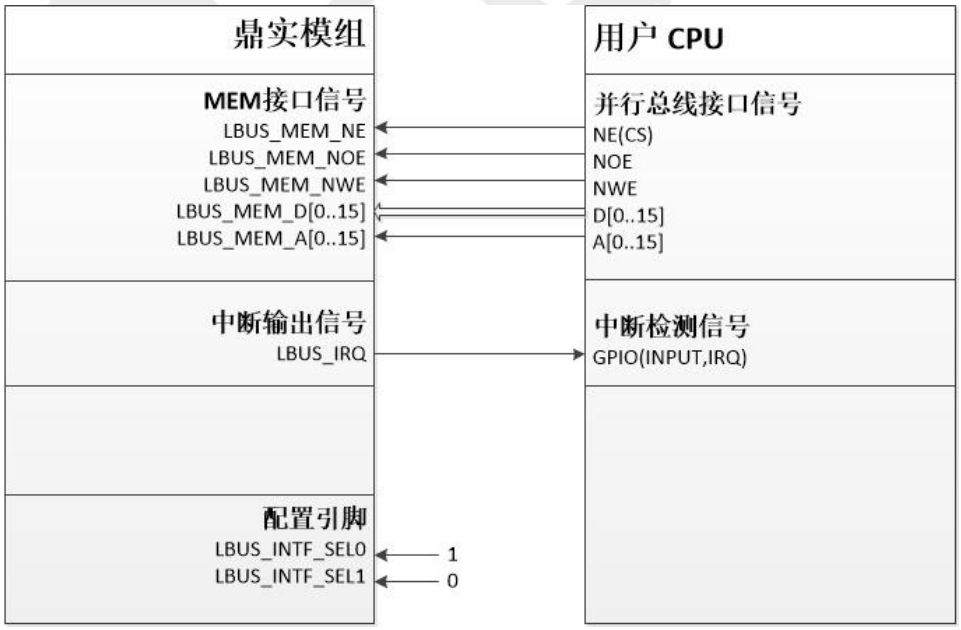


1.3.6 MEM 接口

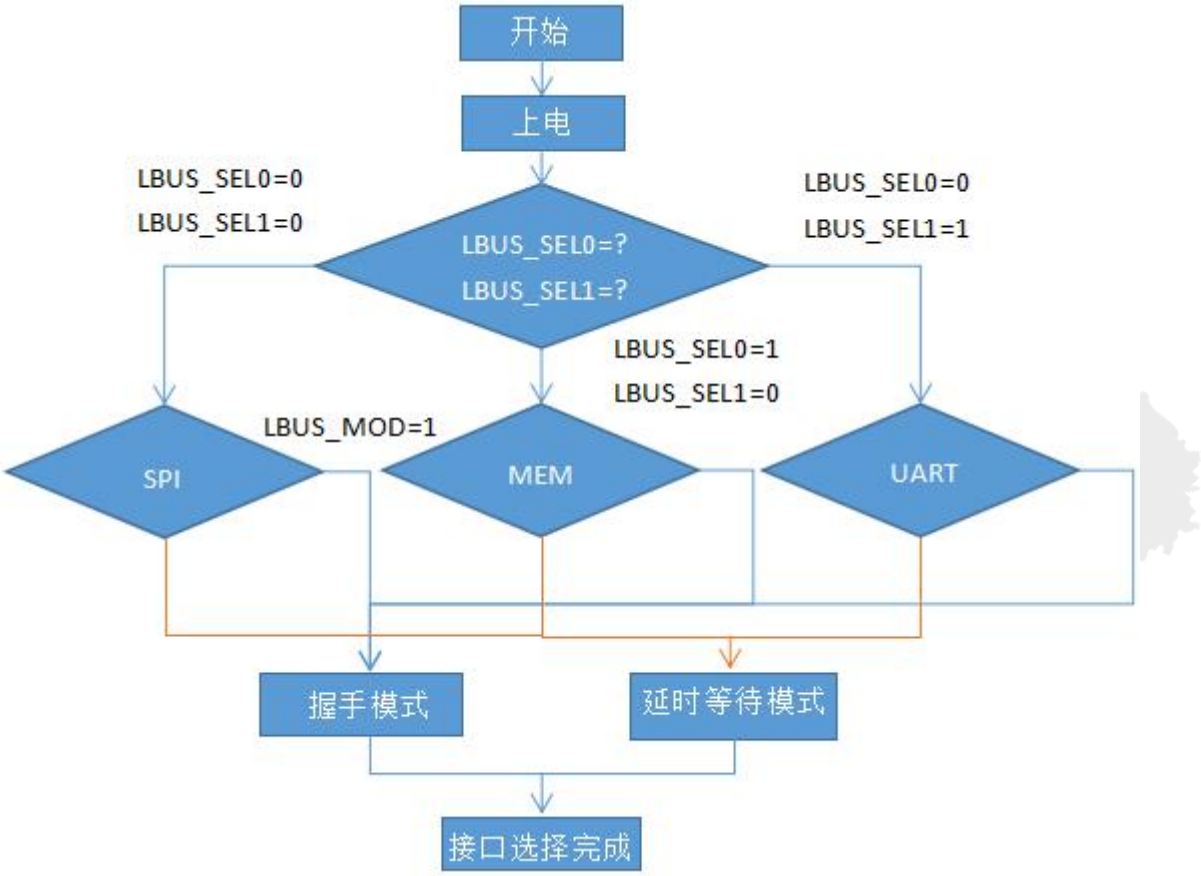
对于上面的串行通信接口无法满足用户数据带宽要求的场合，可以使用 MEM 接口。MEM 接口是一个 16 位宽数据总线的通信接口，类似于一个异步静态 RAM，在模组内部 MEM 接口对应着一片同享内存区。用户通过写 MEM 接口将数据放到内部输入共享数据区中，这些数据会由模组内部 CPU 读取，模组内部的 CPU 会将应答数据写到内部输出共享数据区中，，用户通过读 MEM 接口将数据读出。常用的带并行总线接口的 MCU，MPU 或 FPGA 都可以通过 MEM 接口与模组交互。

备注:目前仅支持 SPI 握手模式，

MEM 接口连接图



1.3.7 接口选择流程图


开始
上电
LBUS_SEL0=0
LBUS_SEL1=0
LBUS_SEL0=?
LBUS_SEL1=?
LBUS_SEL0=0
LBUS_SEL1=1
LBUS_SEL0=1
LBUS_SEL1=0
LBUS_MOD=1
SPI
MEM
UART
握手模式
延时等待模式
接口选择完成

1.3.8 开发流程

第一步：硬件设计

根据后面第二章的模组硬件外形尺寸、接口管脚定义指标等，进行用户板的原理图及 PCB 设计。

第二步：软件设计

参考后面第三章模组软件设计说明，编写用户板与 PN 板卡用户接口部分的通讯规约， 可参考鼎实提供的应用例程函数，即可快速实现接口通讯。

第三步：修改 GSDML 文件

根据用户最终产品的 IO 指标的需求，参考鼎实提供的 GSDML 文件模板，用户实现自己 GSDML 文件的编写。

第四步：用 PN 主站配置调试

选用一台 PN 主站，例如用西门子的 300 或 1500PLC，使用博图软件或 STEP-7 进行 PN 从站组态及测试。

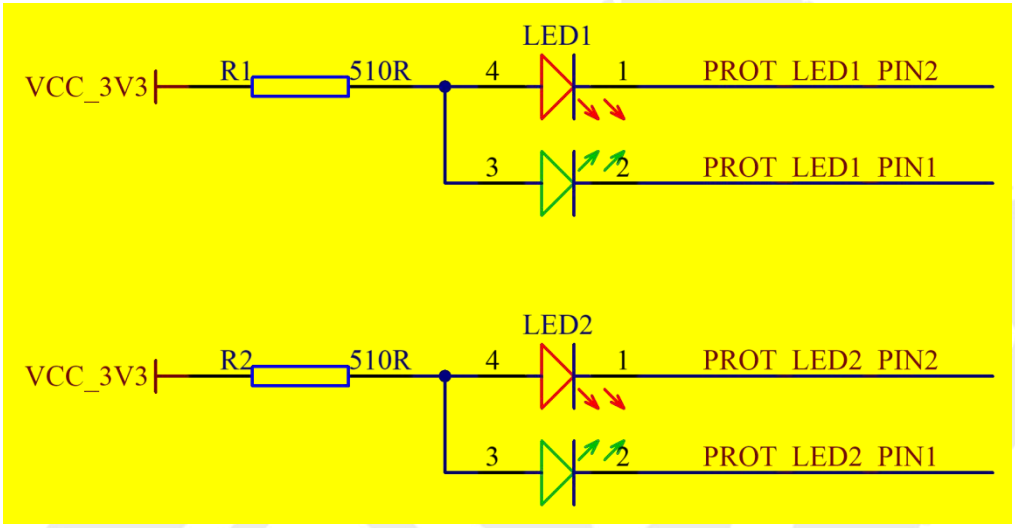
第二章 硬件设计说明

2.1 指示灯状态

模组外扩两个指示协议栈运行状态的指示灯，用于指示协议栈运行状态

协议栈 LED 指示灯参考设计如下。

指示灯连接图



指示灯状态定义

指示灯	功能说明
LED1	熄灭:PN 通讯没有总线错误 红色常亮:PN 通讯有总线错误
LED2	熄灭:PN 通讯没有系统错误 红色常亮:PN 通讯有系统错误

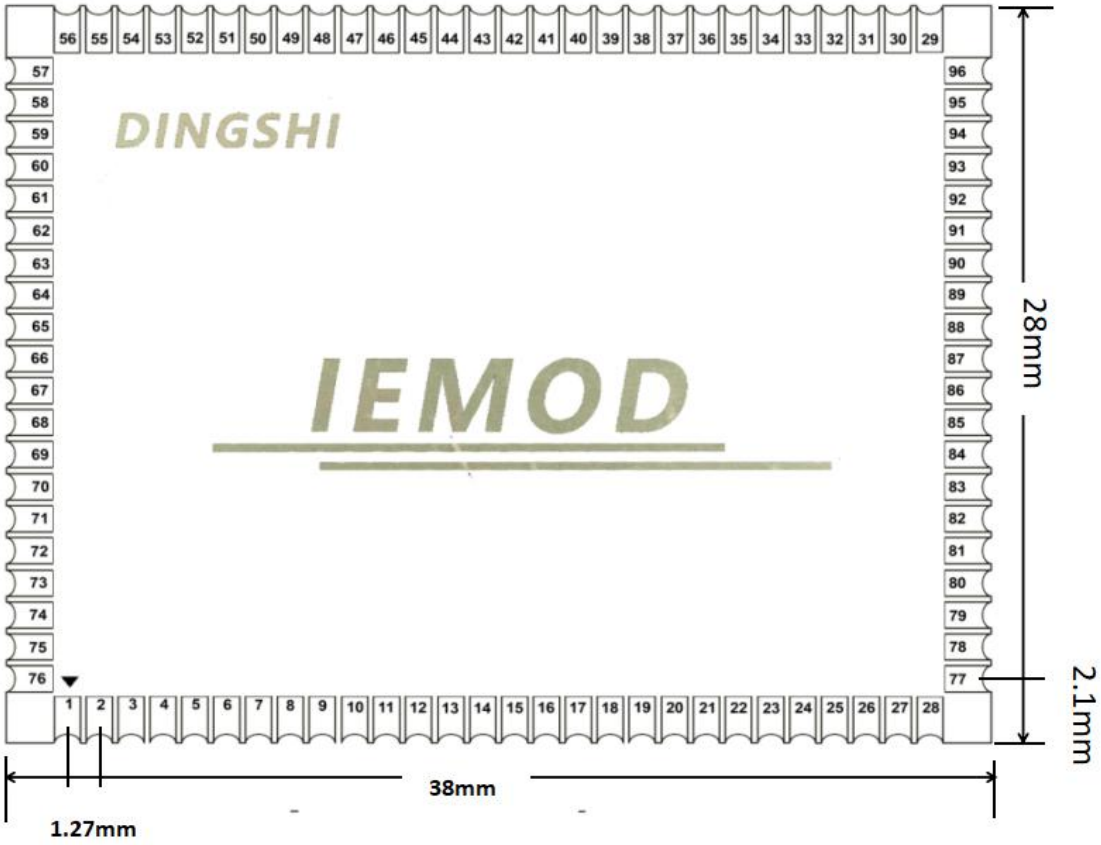
2.2 结构尺寸

模块尺寸：38 mm x 28 mm x 3.7 mm

焊盘尺寸：宽 0.8mm，半孔内径 0.6mm，中心间距 1.27mm，外侧焊盘距边缘 2.1mm

封装 ：TQFP-96

模块共有 96 个引脚，焊盘分布于模组四周



2.3 引脚描述

引脚编号	名称	IO 类型	功能
1	GND	PWR	电源地
2	VCC_3V3	PWR	3.3V 电源输入
3	LBUS_MEM_NE	DI	LBUS MEM 接口片选信号
4	LBUS_MEM_NOE	DI	LBUS MEM 接口读使能信号

5	LBUS_MEM_NWE	DI	LBUS MEM 接口写使能信号
6	LBUS_MEM_D0	DIO	LBUS MEM 接口数据位 0 信号
7	LBUS_MEM_D1	DIO	LBUS MEM 接口数据位 1 信号
8	LBUS_MEM_D2	DIO	LBUS MEM 接口数据位 2 信号
9	LBUS_MEM_D3	DIO	LBUS MEM 接口数据位 3 信号
10	LBUS_MEM_D4	DIO	LBUS MEM 接口数据位 4 信号
11	LBUS_MEM_D5	DIO	LBUS MEM 接口数据位 5 信号
12	LBUS_MEM_D6	DIO	LBUS MEM 接口数据位 6 信号
13	LBUS_MEM_D7	DIO	LBUS MEM 接口数据位 7 信号
14	RSV	RSV	预留管脚
15	RESTORE_FACTORY	LATCH PULLUP	恢复出厂设置，悬空或为高运行于正常工作模式，为低持续 3s 以上恢复工作固件为出厂固件（代码 flash 中放不下两个固件，只能用出厂固件将工作固件覆盖）
16	LBUS_EXCH_MOD	LATCH PULLDOWN	LBUS 交互模式，上电锁存信号，悬空或低电平工作在延时等待模式，高电平工作在握手模式
17	LBUS_INTF_SEL0	LATCH PULLDOWN	LBUS 接口选择位，上电锁存信号，LBUS_INTF_SEL[0..1]取值对应接口选择：00：SPI 接口；01：UART 接口；10：FIFO 接口；11：预留
18	LBUS_INTF_SEL1		
19	NC	NC	悬空

20	GND	PWR	电源地
21	LBUS_IRQ	DO	LBUS 中断信号，模组产生异步事件由该引脚提交中断信号，上升沿有效
22	LBUS_ACK	DO	LBUS 应答信号，模组应答数据就绪时将该引脚切换为高电平，应答数据发送完成后将该引脚切换为低电平
23	LBUS_SPI_MISO/ LBUS_UART_RX	DO	LBUS SPI 接口 MISO 信号/LBUS UART 接收数据信号
24	LBUS_RTS	DI	LBUS 请求发送信号，当模组工作在握手模式时，在 LBUS 发送数据期间用户需要将该信号设置为高电平，非发送数据期间维持低电平
25	LBUS_SPI_SCK/ LBUS_UART_TX	DI/DO	LBUS SPI 接口时钟信号/LBUS UART 发送数据信号
26	LBUS_SPI_MOSI	DI/DO	LBUS SPI 接口 MOSI 信号
27	VCC_3V3	PWR	3.3V 电源输入
28	GND	PWR	电源地
29	GND	PWR	电源地
30	NC	NC	悬空
31	NC	NC	悬空
32	DBG_COM1_RX	DI	模组调试串口 1 接收数据信号
33	DBG_COM1_TX	DO	模组调试串口 1 发送数据信号

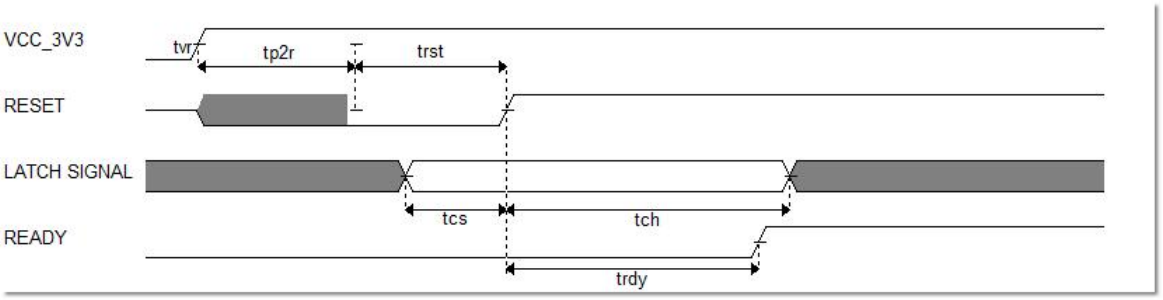
34	READY	DO	模组启动就绪信号，高电平有效，上电后用户需要等待直到该信号为高电平才能通过 LBUS 与模组交互
35	RESET	DI	复位输入信号，低电平有效
36	PROT_LED2_PIN2	DO	协议栈双色指示灯 LED2 引脚 2 控制信号
37	PROT_LED2_PIN1	DO	协议栈双色指示灯 LED2 引脚 1 控制信号
38	PROT_LED1_PIN2	DO	协议栈双色指示灯 LED1 引脚 2 控制信号
39	PROT_LED1_PIN1	DO	协议栈双色指示灯 LED1 引脚 1 控制信号
40	VCC_3V3	PWR	3.3V 电源输入
41	GND	PWR	电源地
42	ETH_PORT1_ACT	DO	以太网端口 1 活动指示灯控制信号，网络有数据时该信号为低电平
43	ETH_PORT1_LINK	DO	以太网端口 1 链路指示灯控制信号，网络物理层链路建立后该信号为低电平
44	ETH_PORT2_ACT	DO	以太网端口 2 活动指示灯控制信号，网络有数据时该信号为低电平
45	ETH_PORT2_LINK	DO	以太网端口 2 链路指示灯控制信号，网络物理层链路建立后该信号为低电平
46	GND	PWR	电源地
47	ETH_PORT1_RX_N	AI	以太网端口 1 接收数据差分信号负极
48	ETH_PORT1_RX_P	AI	以太网端口 1 接收数据差分信号正极

49	ETH_PORT1_TX_N	AO	以太网端口 1 发送数据差分信号负极
50	ETH_PORT1_TX_P	AO	以太网端口 1 发送数据差分信号正极
51	GND	PWR	电源地
52	ETH_PORT2_RX_N	AI	以太网端口 2 接收数据差分信号负极
53	ETH_PORT2_RX_P	AI	以太网端口 2 接收数据差分信号正极
54	ETH_PORT2_TX_N	AO	以太网端口 2 发送数据差分信号负极
55	ETH_PORT2_TX_P	AO	以太网端口 2 发送数据差分信号正极
56	GND	PWR	电源地
57	PHY_MODE	LATCH PULLDOWN	以太网物理层接口选择。悬空或拉低为电口，拉高为光口；
58	RSV	RSV	预留管脚
59	RSV	RSV	预留管脚
60	RSV	RSV	预留管脚
61	RSV	RSV	预留管脚
62	RSV	RSV	预留管脚
63	RSV	RSV	预留管脚
64	RSV	RSV	预留管脚
65	RSV	RSV	预留管脚

66	SYNC_LATCH_0	DIO	运动控制用同步输出或锁存输入引脚 0
67	SYNC_LATCH_1	DIO	运动控制用同步输出或锁存输入引脚 1
68	GND	PWR	电源地
69	LBUS_MEM_D8	DIO	LBUS MEM 接口数据位 8 信号
70	LBUS_MEM_D9	DIO	LBUS MEM 接口数据位 9 信号
71	LBUS_MEM_D10	DIO	LBUS MEM 接口数据位 10 信号
72	LBUS_MEM_D11	DIO	LBUS MEM 接口数据位 11 信号
73	LBUS_MEM_D12	DIO	LBUS MEM 接口数据位 12 信号
74	LBUS_MEM_D13	DIO	LBUS MEM 接口数据位 13 信号
75	LBUS_MEM_D14	DIO	LBUS MEM 接口数据位 14 信号
76	LBUS_MEM_D15	DIO	LBUS MEM 接口数据位 15 信号
77	LBUS_MEM_A0	DI	LBUS MEM 接口地址位 0 信号
78	LBUS_MEM_A1	DI	LBUS MEM 接口地址位 1 信号
79	LBUS_MEM_A2	DI	LBUS MEM 接口地址位 2 信号
80	LBUS_MEM_A3	DI	LBUS MEM 接口地址位 3 信号
81	LBUS_MEM_A4	DI	LBUS MEM 接口地址位 4 信号
82	LBUS_MEM_A5	DI	LBUS MEM 接口地址位 5 信号
83	LBUS_MEM_A6	DI	LBUS MEM 接口地址位 6 信号

84	LBUS_MEM_A7	DI	LBUS MEM 接口地址位 7 信号
85	GND	PWR	电源地
86	LBUS_MEM_A8	DI	LBUS MEM 接口地址位 8 信号
87	LBUS_MEM_A9	DI	LBUS MEM 接口地址位 9 信号
88	LBUS_MEM_A10	DI	LBUS MEM 接口地址位 10 信号
89	LBUS_MEM_A11	DI	LBUS MEM 接口地址位 11 信号
90	LBUS_MEM_A12	DI	LBUS MEM 接口地址位 12 信号
91	LBUS_MEM_A13	DI	LBUS MEM 接口地址位 13 信号
92	LBUS_MEM_A14	DI	LBUS MEM 接口地址位 14 信号
93	LBUS_MEM_A15	DI	LBUS MEM 接口地址位 15 信号
94	LBUS_MEM_WAIT	DO	LBUS MEM 接口等待信号
95	DBG_COM2_TX	DO	模组调试串口 2 发送数据信号
96	DBG_COM2_RX	DI	模组调试串口 2 接收数据信号

2.4 上电及复位时序



注：LATCH SIGNAL 表示上电锁存信号，参见引脚定义中 IO 类型为 LATCH 的引脚。

符号	描述	最小值	典型值	最大值	单位
tvr	VCC_3V3 电源上升时间	—	—	100	us
tp2r	上电到复位信号生效的时间	0	—	—	us
trst	复位信号持续时间	—	2	—	ms
tcs	锁存信号建立时间	0	—	—	us
tch	锁存信号保持时间	—	—	—	ms
trdy	模组就绪时间	—	300	—	ms

2.5 LBUS 接口

LBUS 接口是鼎实模组与用户进行本地数据交互的信号集合。为适应不同的应用场合，LBUS 支持不同的工作模式和子接口类型。LBUS 包含 SPI，UART 等接口类型，可以运行在延时等待模式和握手模式，握手模式需要通过一些辅助握手信号来实现。

LBUS 的子接口选择通过上电锁存 LBUS_INTF_SEL0 和 LBUS_INTF_SEL1 信号来实现，工作模式通过上电锁存 LBUS_EXCH_MOD 信号选择。

LBUS_INTF_SEL0	LBUS_INTF_SEL1	接口选择
0	0	SPI
0	1	UART
1	0	MEM
1	1	无效

2.6 工作模式

LBUS 接口是一种主从交互模式，用户侧做为主设备向模组发送一条请求消息，模组回复一条应答消息。根据发送请求消息和应答消息的时机不同，工作模式分为延迟等待模式和握手模式两种

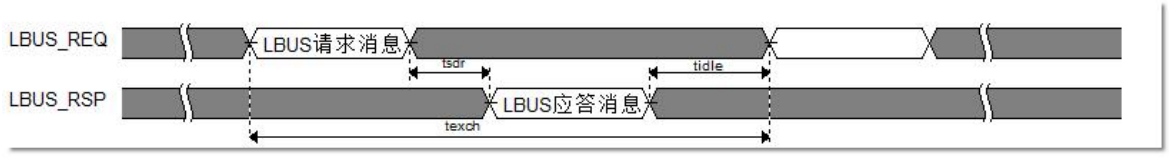
LBUS_EXCH_MOD	工作模式
0	延迟等待
1	握手模式

2.7 延迟等待模式

延迟等待模式下，用户侧发送请求消息后，获取应答消息的时机基于确定的延迟时间，启动。

相较于握手模式，延迟等待模式优点是对用户 MCU 需求资源更小，占用很少的 IO，缺点是固定的延迟时间会导致用户输入输出数据刷新较慢。

延迟等待模式时序图



时序参数描述如下

符号	描述	最小值	典型值	最大值	单位
tsdr	请求消息和应答消息之间的间隔时间	2	—	—	ms
tidle	应答消息和下一次请求消息之间的空闲时间	1	—	—	ms
texch	一次 LBUS 访问的时间	—	12	—	ms

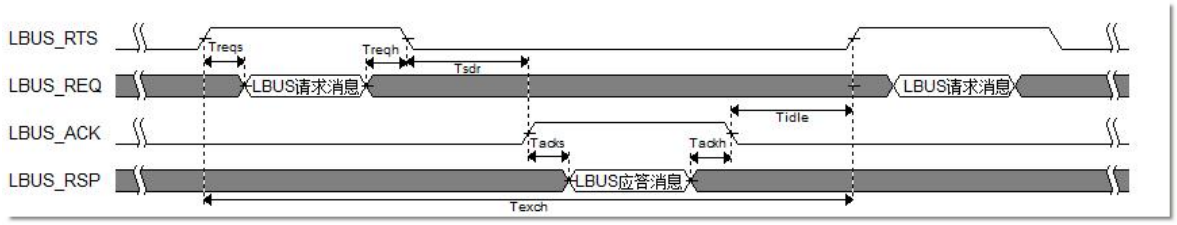
2.8 握手模式

握手模式下，用户侧发送请求消息的开始和结束，模组应答消息的开始和截止都通过辅助握手

信号的操控。通过检测这些信号可以立即启动消息接收动作，减小不必要的延迟等待，提高 LBUS 接口的通信效率。

相较于延时等待模式，握手模式优点是用户输入输出数据刷新较快，缺点是握手模式占用的 MCU 资源较大，需要用户 MCU 连接 LBUS_RTS 与 LBUS_ACK 两个握手信号。

握手模式时序图



时序参数描述如下

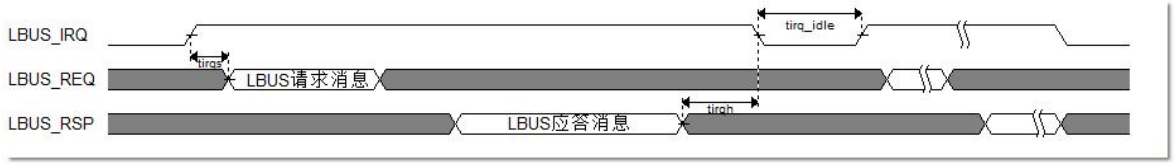
符号	描述	最小值	典型值	最大值	单位
treqs	LBUS_RTS 信号建立时间	0	10	—	us
treqh	LBUS_RTS 信号保持时间	0	—	—	us
tsdr	请求消息和应答消息之间的间隔时间	—	—	—	us
tacks	LBUS_ACK 信号的建立时间	—	—	—	us
tackh	LBUS_ACK 信号的保持时间	—	—	—	us
tidle	应答消息和下一次请求消息之间的空闲时间	1	—	—	us
texch	一次 LBUS 访问的时间	—	12	—	us

2.9 异步中断机制

LBUS 接口还包含一个 LBUS_IRQ 信号，当模组有异步事件时可以通过该信号告知用户。随后用

户会发送异步事件读取消息，获取来自用户的异步事件响应。获取一条异步事件应答后 LBUS_IRQ 信号变低，如果模组内部又有新的异步事件，则 LBUS_IRQ 信号再次变高。

异步中断信号时序图



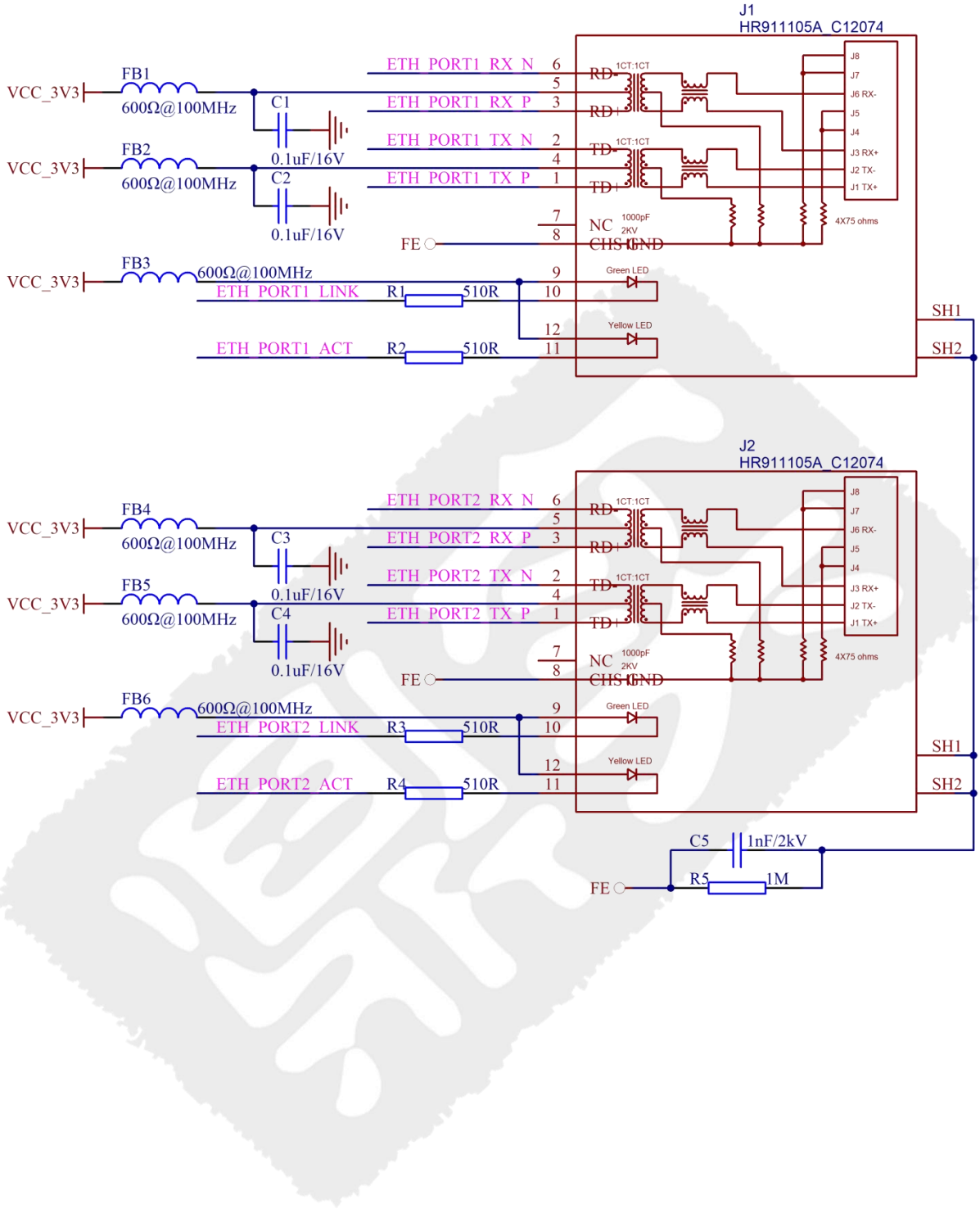
时序参数描述如下

符号	描述	最小值	典型值	最大值	单位
tirqs	LBUS_IRQ 建立时间，LBUS_REQ 必须在 LBUS_IRQ 信号产生后，在晚于该时间之后才能发送 LBUS_REQ	0	—	—	ns
tirqh	LBUS_IRQ 保持时间，LBUS_RSP 结束后，会持续该时间之后才释放 LBUS_IRQ 信号	0	—	—	ns
tirq_idle	相邻两个 LBUS_IRQ 有效信号之间的最小间隔	1	—	—	us

2.10 以太网接口

模组内置交换芯片，对外提供两个 100Mpbs 交换机端口 PORT1 和 PORT2，每个接口有反映物理链路状态的指示灯 LINK 和 ACT。

以太网接口的参考设计如下:


J1
HR911105A_C12074
VCC_3V3
FB1
600Ω@100MHz
C1
0.1uF/16V
FB2
600Ω@100MHz
C2
0.1uF/16V
FB3
600Ω@100MHz
ETH_PORT1_RX_N
ETH_PORT1_RX_P
ETH_PORT1_TX_N
ETH_PORT1_TX_P
ETH_PORT1_LINK
ETH_PORT1_ACT
R1
510R
R2
510R
FE
RD-
RD+
TD-
TD+
1CT:1CT
NC
1000pF
2KV
CHS GND
4X75 ohms
Green LED
Yellow LED
SH1
SH2
J2
HR911105A_C12074
FB4
600Ω@100MHz
C3
0.1uF/16V
FB5
600Ω@100MHz
C4
0.1uF/16V
FB6
600Ω@100MHz
ETH_PORT2_RX_N
ETH_PORT2_RX_P
ETH_PORT2_TX_N
ETH_PORT2_TX_P
ETH_PORT2_LINK
ETH_PORT2_ACT
R3
510R
R4
510R
C5
1nF/2kV
R5
1M

第三章 软件设计说明

购买模组的的同时，鼎实会提供 DSMODLib 库文件，提供用户侧开发例程源代码等并提供产品全生命周期的技术支持。

3.1 DSMODLib 库函数说明

函数名称	功能描述
dsmod_opt_init();	用户通过 LBUS 操作模组的接口初始化
dsmod_opt_get_dev_info(&dev_info);	获取模组描述信息
dsmod_opt_get_dev_state(&dev_state);	获取模组当前运行状态
dsmod_opt_get_slave_state(&slave_state);	获取从站协议栈当前运行状态
dsmod_opt_get_slave_cfg(&slave_cfg);	获取从站协议栈当前配置数据
dsmod_opt_get_net_param(&net_param);	获取从站协议栈网络参数
dsmod_opt_exch_io(&slave_cfg, &slave_data, &error);	与主站进行周期性 IO 数据交互

3.2 入门例程

下面是一个使用 DSMODLib 初始化模组并进入 IO 数据交换过程的例程，用户仅需很少的操作步骤就可以实现 PROFINET 周期性 IO 数据通信。用户代码流程如下：

```
/** Step 1. 本地 DSMOD 操作接口初始化 */
    ret = dsmod_opt_init();
```

备注:主要包含复位/SPI 初始化/IO 口初始化；

```
/** Step 2. 获取设备描述信息（设备型号，MAC 地址等信息） */
ret = dsmod_opt_get_dev_info(&dev_info);
```

备注:主要包含设备序列号，固件版本等；

/** **Step 3**. 获取设备状态 slave_state */

ret = dsmod_opt_get_dev_state(&dev_state); ///< 周期获取设备状态信息

备注:主要包含设备状态,网口状态等；

/** **Step 4**. 获取从站状态 slave_state */

ret = dsmod_opt_get_slave_state(&slave_state);

备注:主要包含从站配置状态，掉线次数等；

/** **Step 5**. 检测从站配置是否被 IO 控制器更新 */

ret = dsmod_opt_get_slave_cfg(&slave_cfg);

备注:主要包含从站配置输入输出数据长度等；

/** **Step6**. 获取从站网络参数 */

ret = dsmod_opt_get_net_param(&net_param);

备注:主要包含从站 MAC 地址，当前 IP 等；

/** **Step 7**. 与从站进行周期性 IO 数据通信 */

ret = dsmod_opt_exch_io(&slave_cfg, &slave_data, &error);

备注:输入输出数据交换等；

3.3 数据结构

设备描述信息数据结构

```
typedef struct
{
  /* 硬编码字段，直接从芯片读取 */
    unsigned char sn[DEV_SN_LEN];  //设备序列号，对于 OEM 产品，代表鼎实的序列号，不是客户产品的序列号
    unsigned int ver;              //PN 接口的软件及硬件版本号
    unsigned int firmware_id;      //固件 ID
  /* 硬编码字段，值由上位软件改写固件文件生成 */
    char dflt_ip[16];              //默认 IP 地址
    char dflt_gw[16];              //默认网关地址
    char dflt_nm[16];              //默认子网掩码
```

```
    unsigned int rsv[4]; //预留
  /* 可配置字段，值由上位软件在目标设备运行时写入 Data Flash 中 */
    Dev_Info_Param_Type param;      //Dev_Info 中的可设置字段部分，内容保存在 Data Flash
中
} Dev_Info_Type;
```

设备状态信息数据结构

```
typedef struct {
    unsigned short state;          //< 设备状态
    unsigned char firm_type;       //< 当前运行的固件类型
    unsigned char rst_cnt;         //< 重启次数寄存器
    unsigned port_state[2];        //< 两个网络端口的状态 1 为连通，0 为断开
    unsigned int rsv[4];           //< 预留
} Dev_State_Type;
```

从站状态信息数据结构

```
typedef struct {
    unsigned int cfg_id;           //< 配置数据 id，每次配置信息更新该 id 加 1
    unsigned char cfg_stat;        //< 设备配置状态
    unsigned char fsm_state;       //< 从站状态机当前工作状态
    unsigned short offline_cnt;    //< 从站掉线次数统计
    unsigned int rsv[4];           //< 预留
} Slave_State_Type;
```

从站配置信息数据结构

```
typedef struct
{
    unsigned int param_fcs;        //< 参数数据校验值
    unsigned short input_upd_ms;   //< 当前配置的 PN 输入数据更新周期，单位为 ms
    unsigned short output_upd_ms;  //< 当前配置的 PN 输出数据更新周期，单位为 ms
    unsigned short watchdog_ms;    //< 看门狗超时时间，单位为 ms
```

```
    unsigned short dout_len;            //< 从站输出数据长度
    unsigned short din_len;             //< 从站输入数据长度
    unsigned short param_len;           //< 从站参数数据长度
    unsigned int rsv[4];                //预留
    char pn_station_name[PN_STATION_NAME_SIZE];      //  当前 PN 设备名
    Io_Cfg_Tab_T io_cfg_tab[PND_MAX_MODULE_NUM];     //< IO 配置数据结构
    unsigned char io_cfg_param[IO_CFG_PARAM_SIZE];   //< IO 配置参数数据
} Slave_Cfg_Type;
```

从站网络参数数据结构

```
typedef struct {
    char cur_ip[16];                    //当前 IP 地址
    char cur_gw[16];                    //当前网关地址
    char cur_nm[16];                    //当前子网掩码
    char mac[ETH_PORT_NUM][8];          //MAC 地址
    unsigned int rsv[4];                //预留
} Net_Param_Type;
```

周期性 IO 数据数据结构

```
typedef struct {
    unsigned char din_buf[SLAVE_IO_SIZE];
    unsigned char dout_buf[SLAVE_IO_SIZE];
} Slave_IO_Data_Type;
```

3.4 LBUS 接口协议介绍

LBUS 接口是指模组与户的本地通信接口。LBUS 接口基于主从消息通信实现用户和模组的数据交互。用户为主设备，模组为从设备，每次通信由用户请求报文发起，以模组应答报文结束。LBUS 消息中数据以大端模式进行传递。

3.4.1 LBUS 用户请求报文定义

用户与模组间交互报文的格式为如下。整个报文的最大长度不超过*字节。

SC	FC	LEN	DATA	EC

各个字段的含义如下：

名称	长度	含义
SC	1	消息起始码，取值固定为 0x5E
FC	1	bit7：固定为 0 bit[6:0]：消息功能码，取值见下表
LEN	2	消息长度，取值为随后数据字段的长度，最大长度为*字节
DATA	0~*	消息数据部分
EC	1	消息结束码，取值固定为 0xE5

FC 功能码的取值：

取值	名称	含义
1	GET_DEVINFO	获取模组描述信息
2	GET_DEVSTAT	获取模组当前状态
3	GET_NETPARAM	获取设备网络参数
4	GET_SLAVSTAT	获取从站状态
5	GET_SLAVCFG	获取从站当前配置

6	EXCH_USRIO	与协议栈进行周期性 IO 数据交互
7	GET_REC_REQ	从协议栈获取非周期性 IO 数据请求消息
8	SET_REC_RSP	向协议栈发送非周期性 IO 数据应答消息
9	GET_ASYNEVEN	获取异步事件
10	SET_ALARM	向协议栈发送报警消息
11	SET_LOG	向协议栈发送日志信息
12	GET_USRPIPE_REQ	从协议栈获取用户管道数据请求消息
13	SET_USRPIPE_RSP	向协议栈发送用户管道数据应答消息
14	GET_USRPARAM	获取用户参数数据
15	SET_USRPARAM	设置用户参数数据
16	GET_USRFIRMWARE	获取用户升级固件文件

3.4.2 LBUS 模组应答报文定义

应答报文格式与请求报文相同。

SC	FC	LEN	DATA	EC

各个字段的含义如下：

名称	长度	含义
SC	1	消息起始码，取值固定为 0x5E
FC	1	bit7：如果 DATA 部分为正确应答内容，取值为 0，如果 DATA

		部分为错误码信息，取值为 1 bit[6:0]：消息功能码，与请求报文中该字段取值相同
LEN	2	消息长度，取值为随后数据字段的长度，最大长度为*字节
DATA	0~*	消息数据部分，正确应答数据或应答错误码信息。
EC	1	消息结束码，取值固定为 0xE5

正常情况下，模组对于用户的请求会返回正确的应答数据。如果模组内部存在错误，会使用错误数据填充 DATA 字段进行应答。

错误数据的格式为：

ERRCODE	DECODE	CODE1	CODE2

各字段含义如下：

名称	长度	含义
ERRCODE	1	错误代码
DECODE	1	错误子代码
CODE1	1	扩展代码 1
CODE2	1	扩展代码 2

第四章 设备描述信息初始化

DSMOD 设备描述信息初始化采用了以太网口的配置方式，抛弃了以往的用户 MCU 对板卡初始化过程。用户只需在鼎实提供的 TXT 文档上直接修改，节省用户 MCU 资源，用户程序变得更简洁，理解起来更容易。

4.1 描述信息文档介绍

购买模组产品鼎实会提供一份名为“factory_cfg”的 TXT 文档，具体内容如下:

其中红色部分为用户可修改内容，其余保留即可。

####出厂参数配置文件

####格式说明：#开头为注释行，等号前为参数名，等号后为参数值，参数值必须以;结束，= 前后不能有空格，;后回车另起一行

#固件版本号，从首到尾 4 字节的含义：主版本号.次版本号.补丁版本号.编译版本号

soft_ver=2.0.0.0

#硬件版本号，2 字节整数

hard_ver=1;

#设备型号，最长 31 字节字符串

####用户需要设置该参数

model_name=model_name;

#默认 PN 设备名，最长 31 字节字符串

####用户需要设置该参数

dflt_pn_station_name=demo-device-name;

#供应商 ID，2 字节整数

####用户需要设置该参数

vendor_id=0x0A0A;

#设备 ID，2 字节整数

####用户需要设置该参数

device_id=0x0300;

#订货号，最长 20 字节字符串

order_num=order_num;

#设备描述信息，最长 53 字节字符串

####用户需要设置该参数

descriptor=dev_description;

4.2 描述信息文档修改

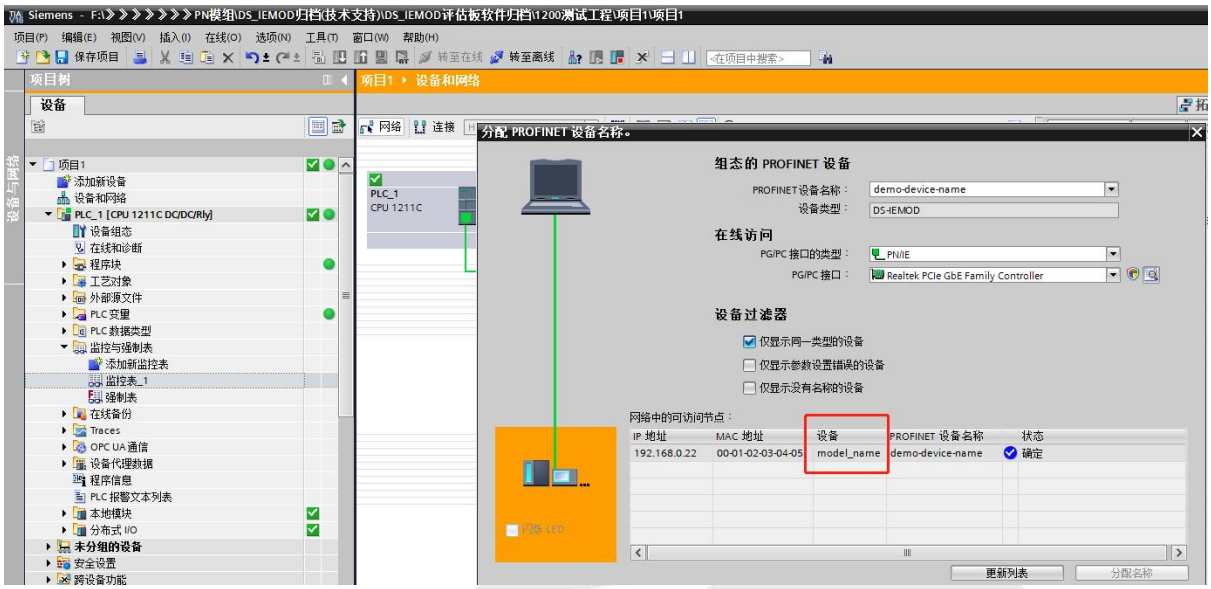
描述文档中，标有用户需要设置该参数需用户根据实际使用改写。

4.2.1 设备型号

修改规则：最长 31 字节字符串;

鼎实默认：model_name=model_name;

鼎实提供的设备型号默认为 model_name，用户可根据自己产品定义,下图为博图 V16 为从站设备分配设备名展示界面:

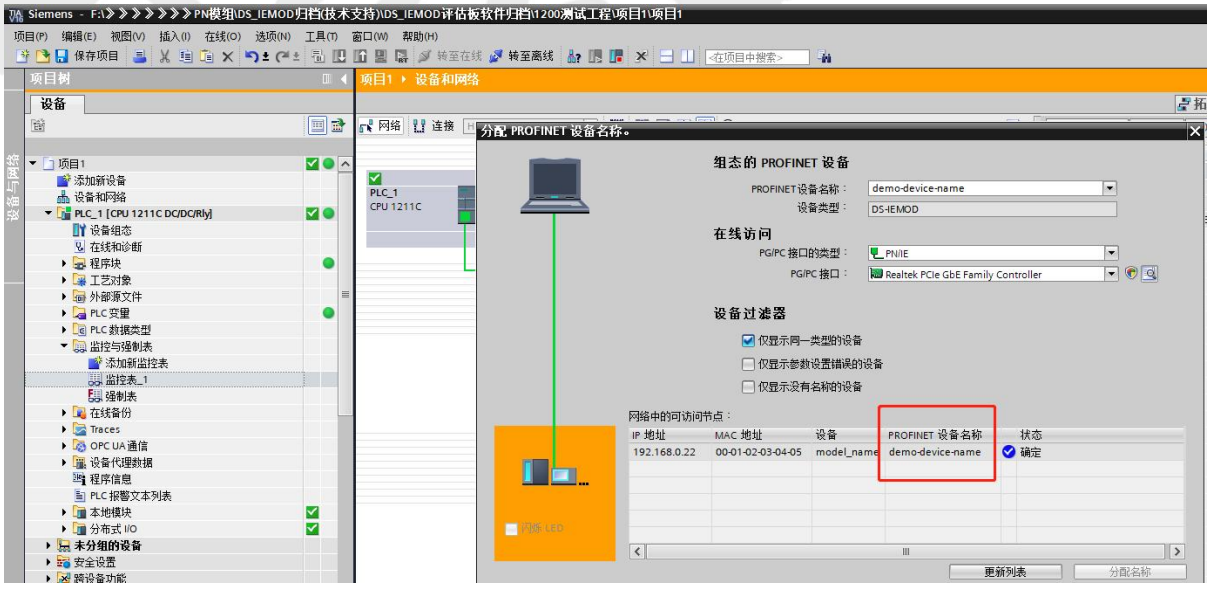


4.2.2 PN 设备名

修改规则：最长 31 字节字符串

鼎实默认：dflt_pn_station_name=demo-device-name

鼎实提供的设备名称默认为 demo-device-name，用户可根据自己产品定义,下图为博图 V16 为从站设备分配设备名展示界面（图中设备名为分配完界面）：



4.2.3 供应商 ID/设备 ID

修改规则：均为 2 字节整数

鼎实默认：vendor_id=0x0A0A / device_id=0x0300;

鼎实提供的 vendor/device_id 为鼎实自己产品所有，0CC9 是鼎实的 id，不能用于认证。每一家去认证的制造商，要向国际 pi 申请自己的制造商 id 和被认证产品的产品 id。

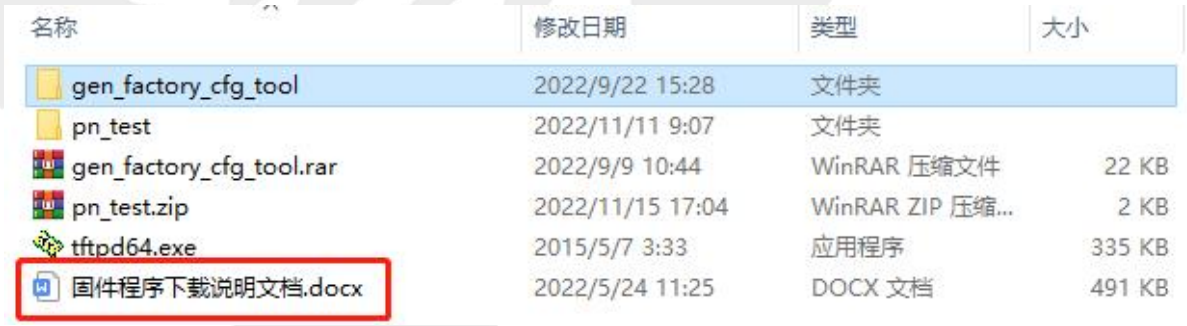
4.3 描述信息文档下载

第一步、

用户描述信息文档根据自己需求修改完成后，需要点击如下应用程序生成一个后缀为“.CFG”文件，“.CFG”为最终下载文件。

第二步、

鼎实提供的用户信息描述文档，还会包含一个如下的升级软件，选中该软件执行升级操作。



第三步、

打开该软件，界面如下图:

1、“Current Directory”按照下图选择即可;

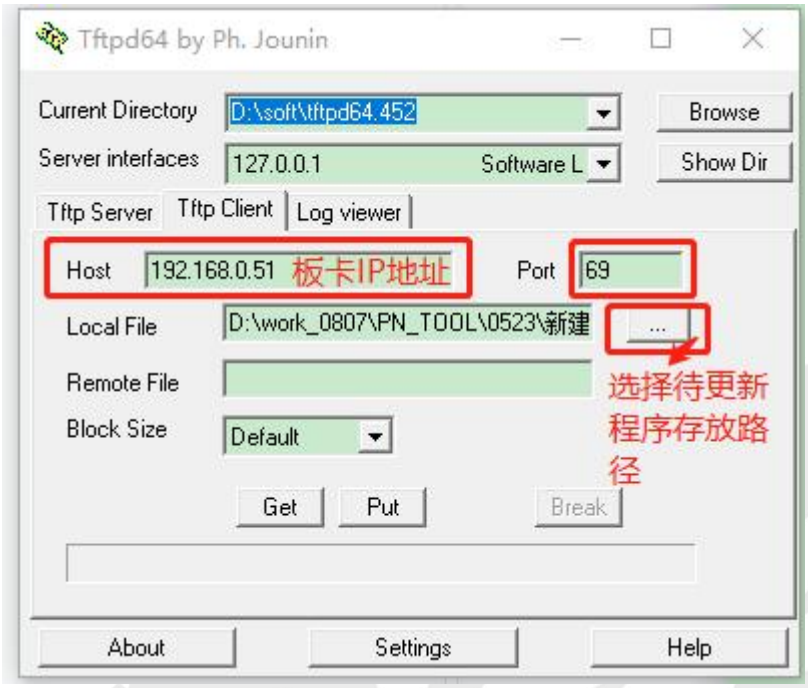
2、“Server interfaces”按照下图选择即可;

3、“Host”为 PN 模组当前 IP（确认 IP 后将 PLC 断开）;

4、“Port”按照下图选择即可;

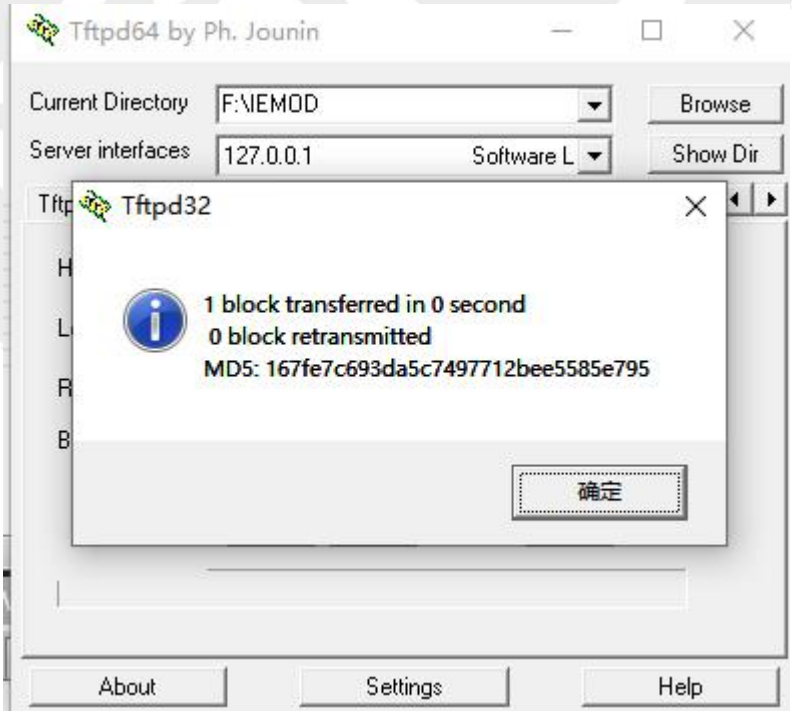
5、“Local File”为下载文件所在目录;

6、“Put”选项开始执行下载操作;



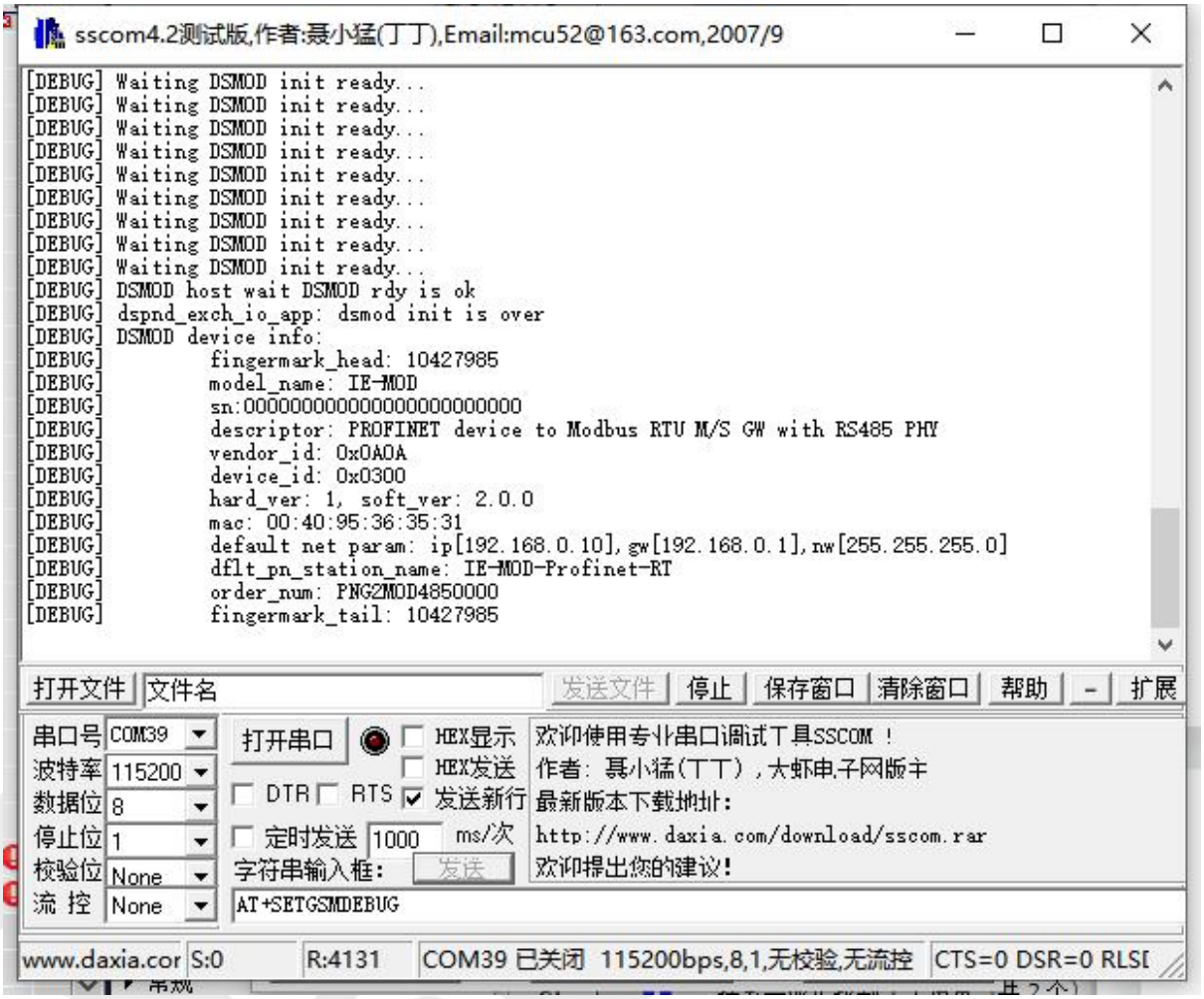
第四步、

固件升级完成界面如下:



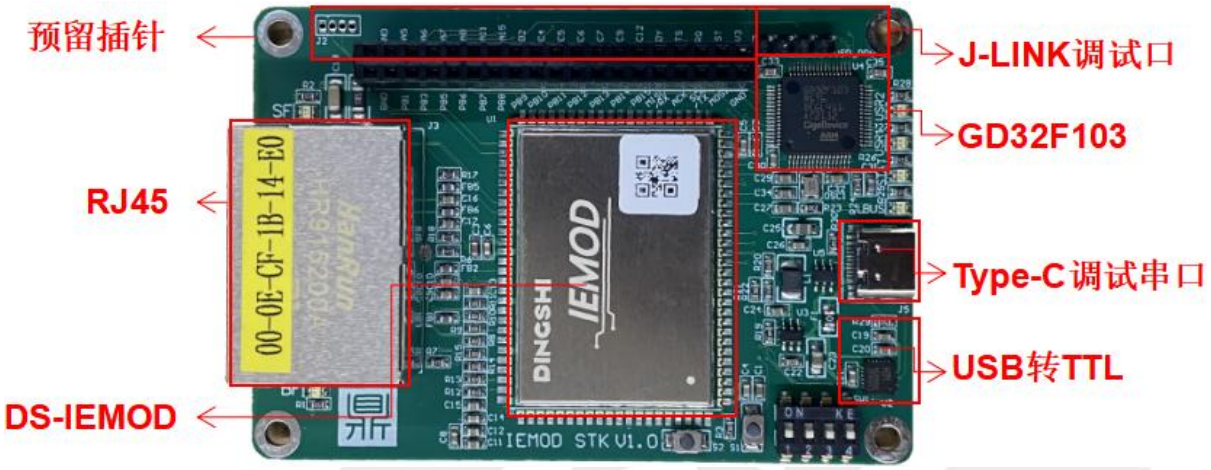
第五步、

通过串口调试软件，查看修改内容:



第五章 DSMOD 评估板

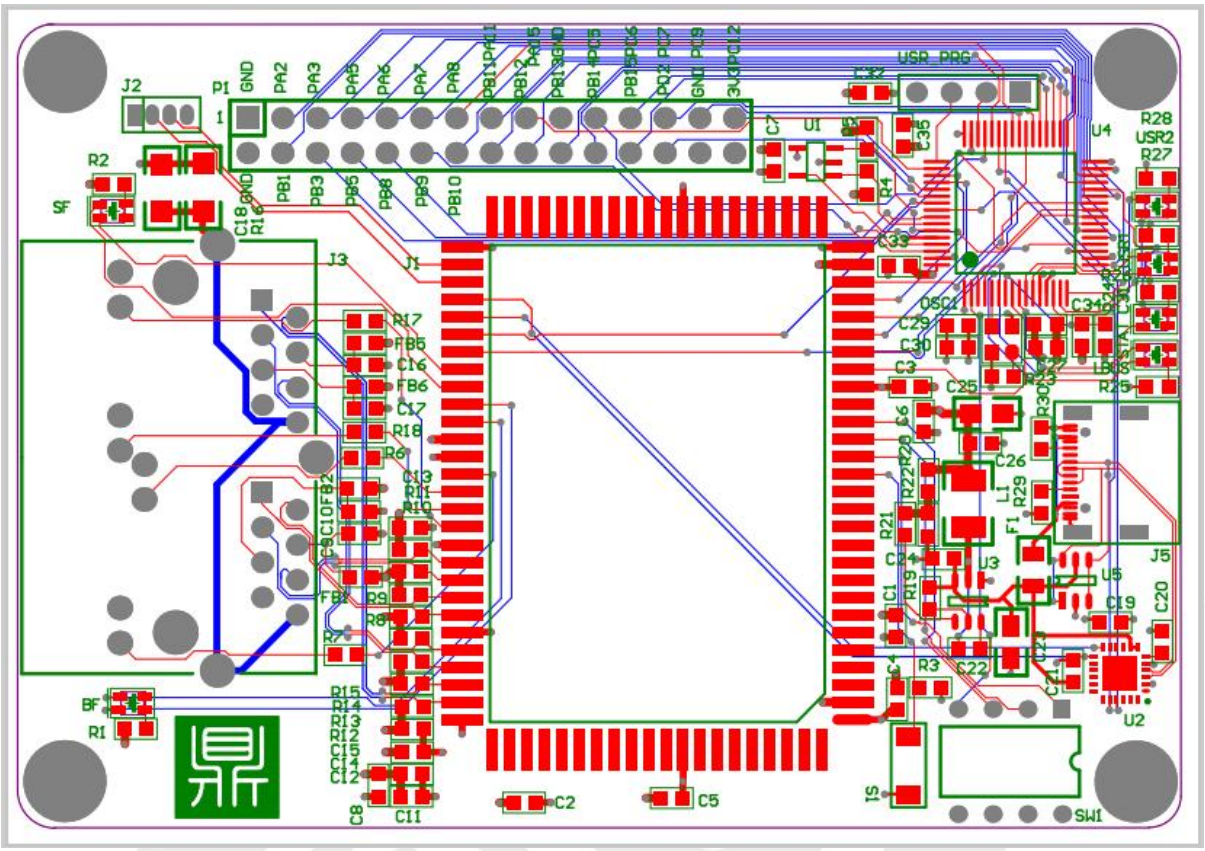
5.1 DSMOD 评估板简介



DSMOD 评估板用来展示模组的主要功能，通过评估板提供的示例程序，可以让用户快速掌握模组的各种使用方法，用户参考示例程序进行修改可以加快产品开发进程。

北京鼎实提供各种主流工业以太网和现场总线协议的模组解决方案，按这两大类划分为工业以太网模组评估板（以下简称 IE-MOD）和工业现场总线模组评估板。每一类都具有完全相同的封装和引脚定义，用户使用两种评估板通过固件升级模组程序可实现多种协议的快速评估。每种模组提供配套的用户侧库函数源代码，不同协议的库函数基本相同，用户在一致的程序流程下完成支持多重协议的产品开发。

5.2 IE-MOD 评估板主要特性



- 尺寸 83mm*46mm（信用卡大小）
- 支持多种工业以太网通信协议的模组；
- 板载与模组通信的用户可编程 CPU，型号 GD32F103RGT6，Cortex-M3 内核，72MHz 主频，1024kB 程序 Flash， 96kB SRAM；
- 外扩 100Mbps 以太网接口；
- 用户 CPU SWDIO 编程口；
- 通过 USB 接口提供 5V 电源，同时板载 USB 转 TTL UART 芯片作为模组和用户 CPU 的调试串口；
- 4 个 LED(USR1/USR2/STAT/LBUS)；
- 通过插针引出其它 IO 供用户使用；
- 功率：小于 3W

5.3 系统需要

- Windows 操作系统（XP，7，8，10）
- USB Type- C 电缆

5.4 开发工具

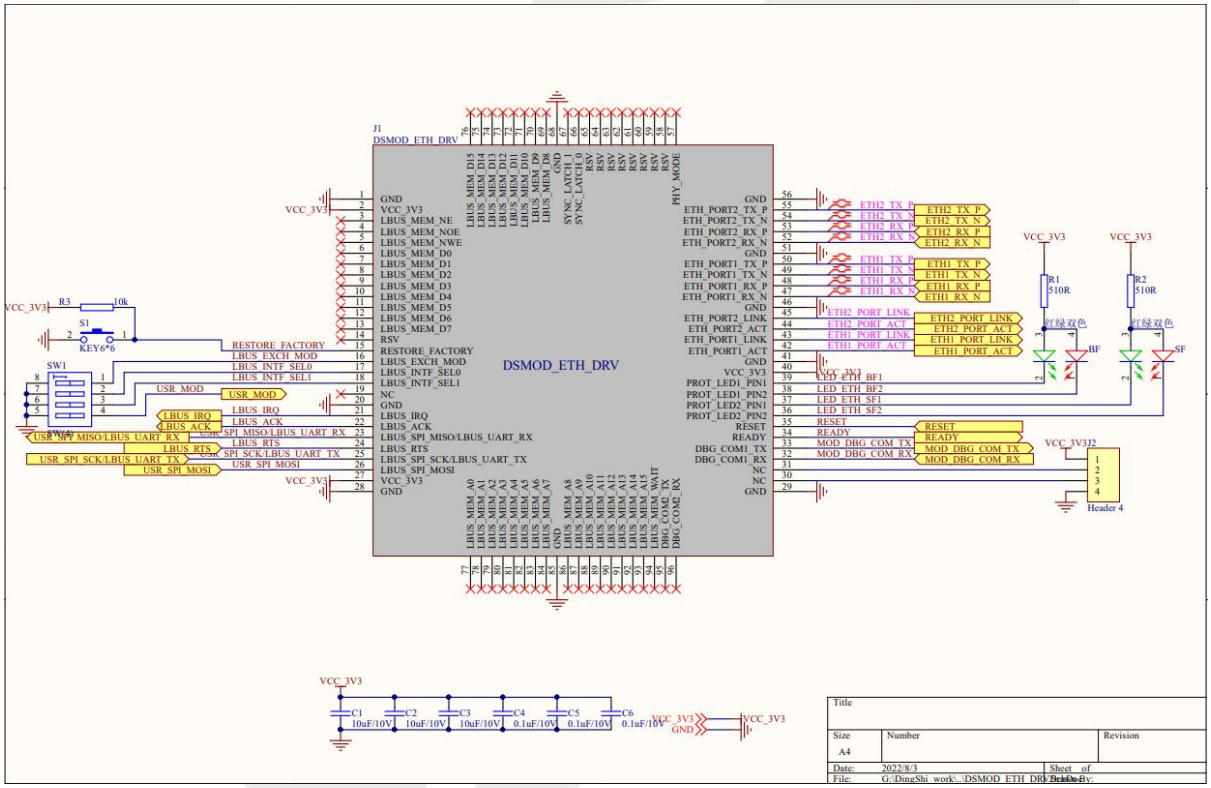
- ARM Keil

5.5 订货信息

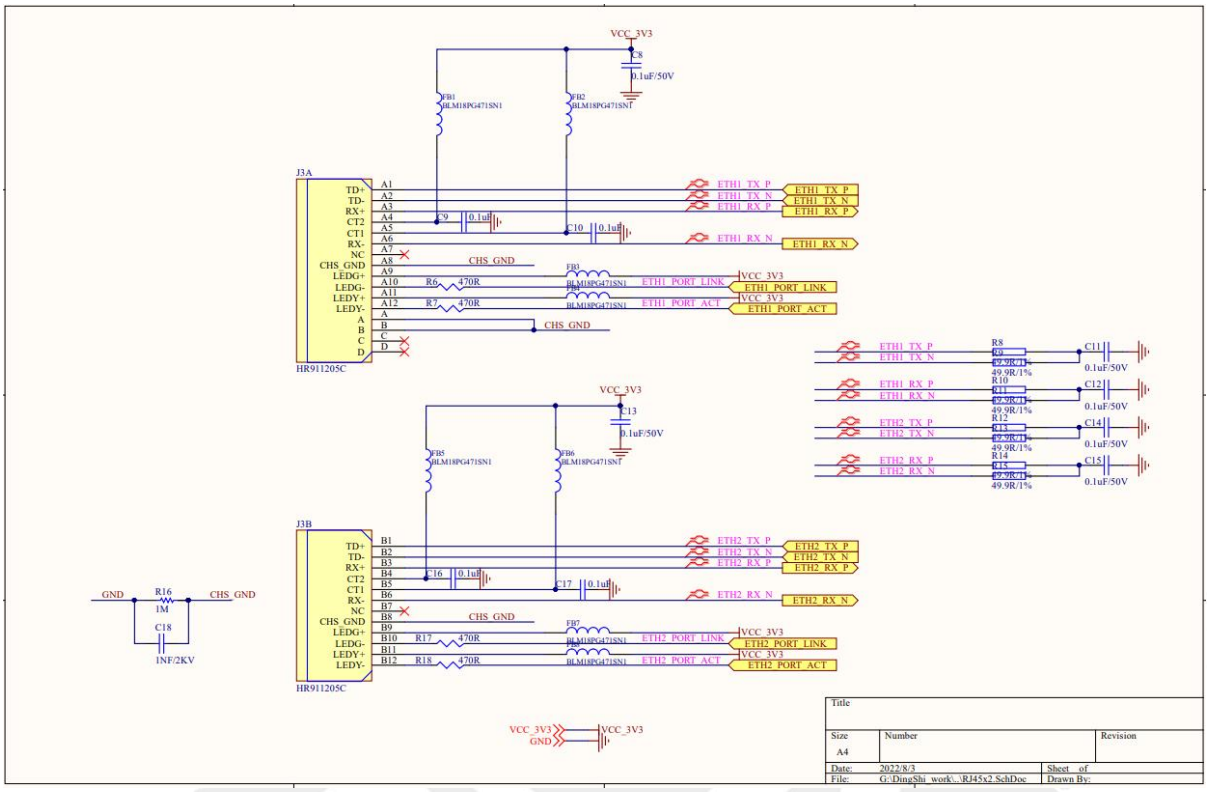
可联系鼎实销售。

5.6 IE-MOD 评估板参考原理图

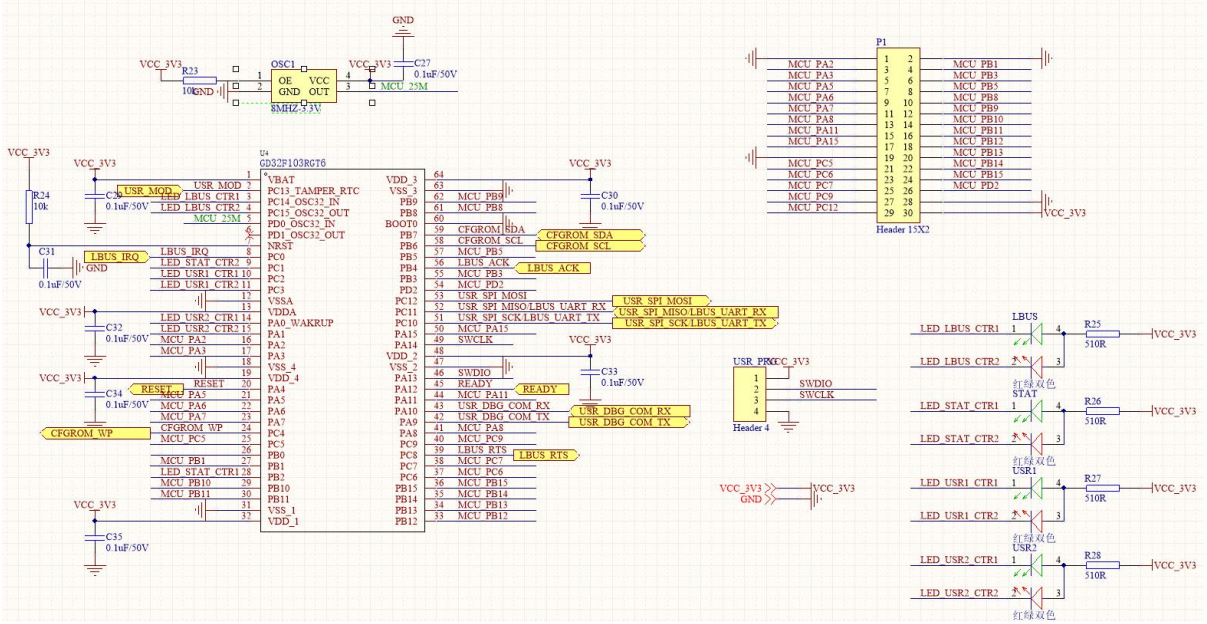
5.6.1 模组参考设计电路



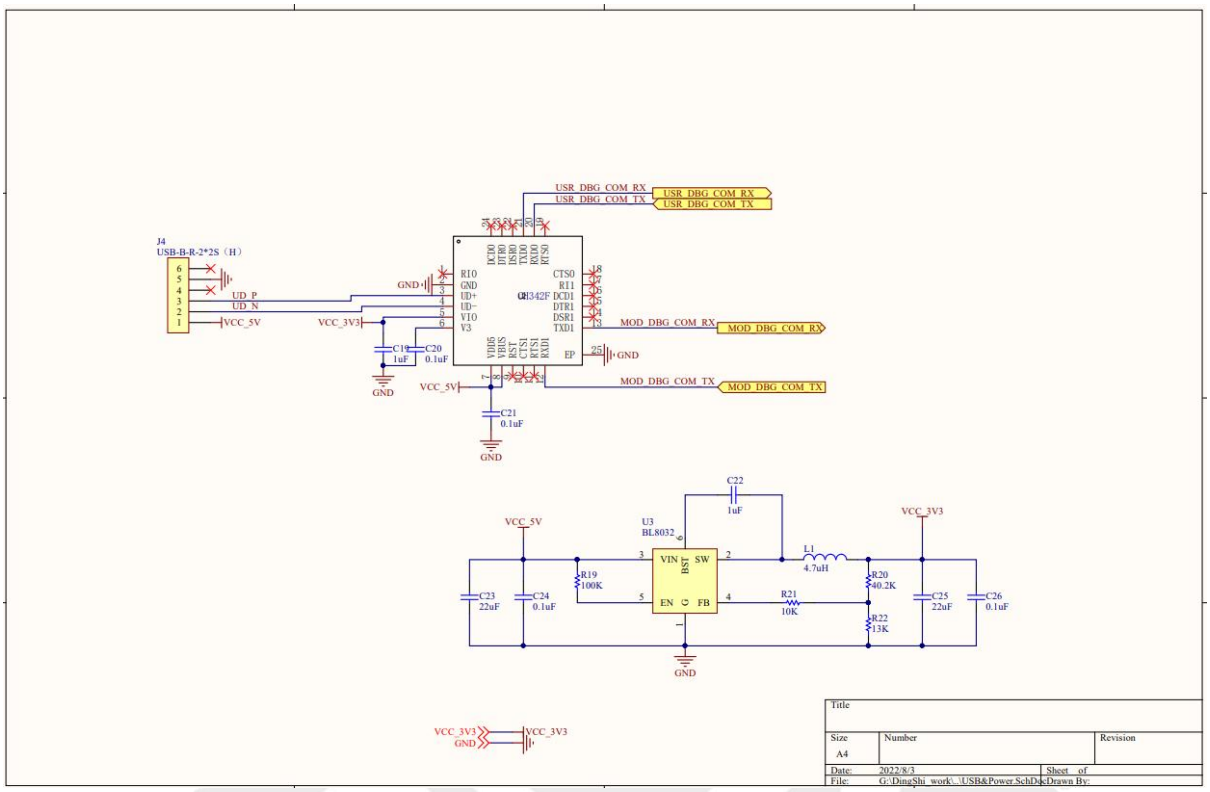
5.6.2 网口参考设计电路



5.6.3 MCU 参考设计电路



5.6.4 USB 调试串口电路



5.6.5 IE-MOD 参考程序设计

联系鼎实技术支持或官网下载。

第六章 PROFINET GXML 文件编写

6.1 PROFINET 的 GXML 文件简介

PROFINET 从站设备与 PROFIBUS-DP 从站类似，需要一个描述文件来描述设备信息，以供主站来获取从站相关信息，并进行组态配置。原来 PROFIBUS-DP 从站用的是 GSD 描述文件，而 PROFINET 从站设备的秒描述文件是 GXML 文件。

鼎实提供了 GXML 文件模板，GSDML-V2.3-DingShi-PND-20220915.xml，可配合鼎实的评估板使用，用户可在此文件模板的基础上修改完成自己产品的 GXML 描述文件。

用户编写完自己的描述文件后，可使用 PI 组织提供的文件核对工具 PROFINET XML Viewer 检查编写的文件是否有问题。

6.2 PROFINET 的 GXML 文件编写

用户在改写 GXML 文件时，只需按需求在模板上修改以下几个部分：

1、用户公司信息

例如鼎实模板中是鼎实的信息：VendorName Value="DS co. ltd"

用户可改成自己公司的相关 logo 信息。

2、产品信息

VendorID 和 DeviceID，例如模板中

<DeviceIdentity VendorID="0x0A0A" DeviceID="0x0300">

用户可自己修改，但这里需要特别注意，这两个 ID 必须和用户对 PN 板卡的基础配置那里的设置一致，否则 PN 不能联通。

3、PN 的插槽个数及 IO 长度及用户参数等信息

此部分是 GXML 文件的主体部分，也是描述设备在 PROFINET 侧组态时的形态及入出数据长度等核心信息。

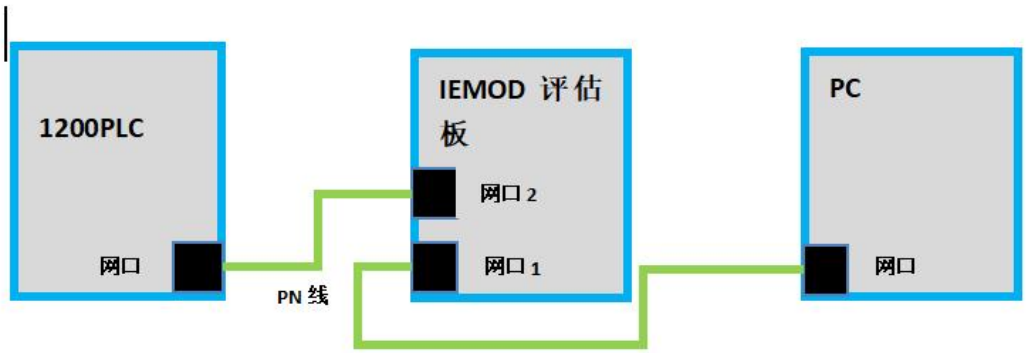
用户需参照模板按照自身需求，进行修改，对于这里有难度的用户，可以直接联系鼎实技术人员辅助完成 GXML 文件的编写。

鼎实公司自主的 GXML 生成工具在开发中，后续推出后，用户只需输入一些信息即可自动生成 GXML 文件，不用在自己去逐条修改 XML 文件。

第七章 Profinet 组网指南

说明：在 PROFINET 侧，本手册以西门子 CPU1200 作为 PROFINET 主站，以鼎实公司 IE-MOD 评估板配置 PROFINET 从站，PC 软件以博图 V16 为例。

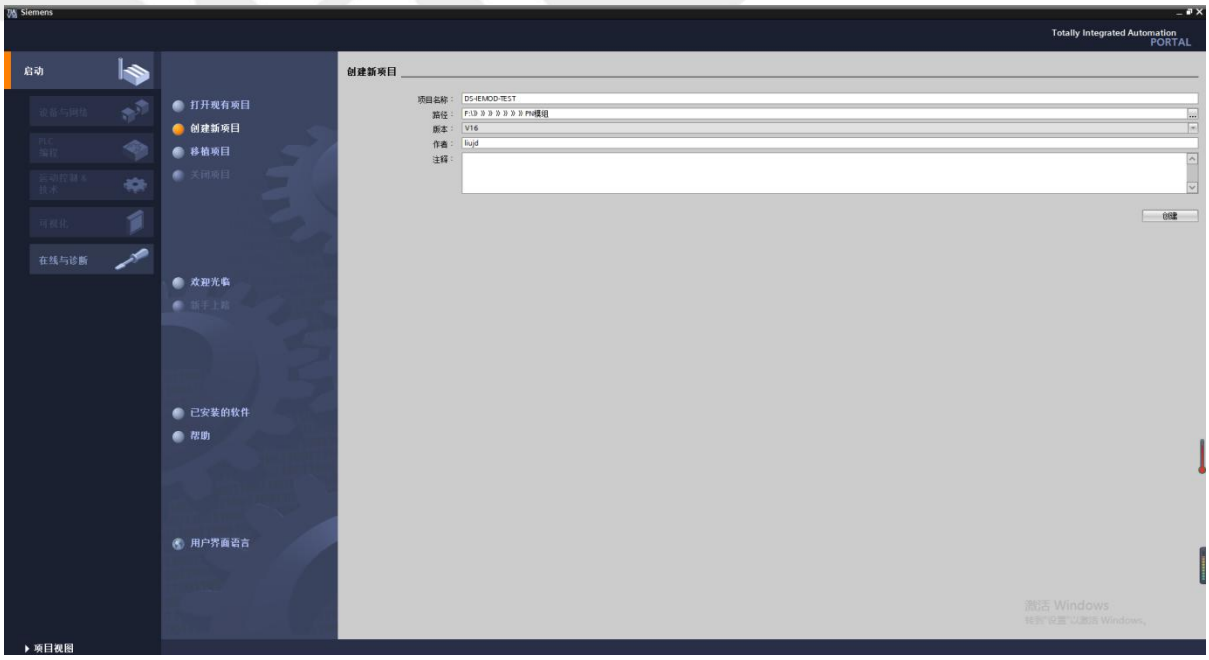
7.1 测试连接图



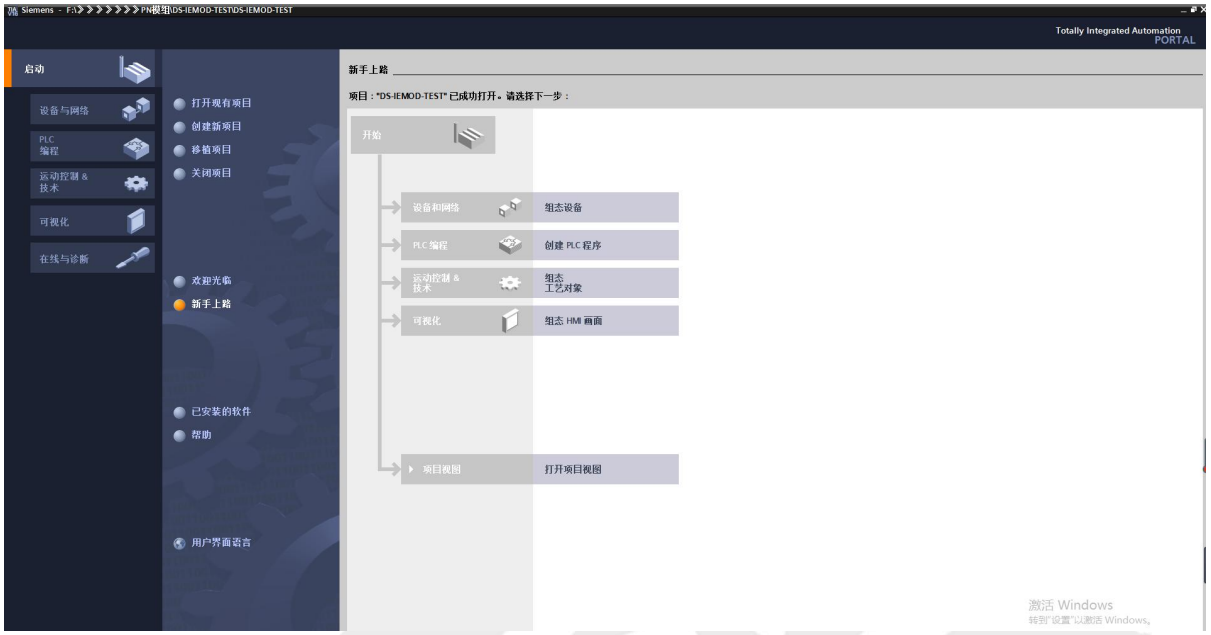
7.2 博图 V16 创建新项目

打开博图软件>创建新项目>选择自己的工程路径>输入自己的项目名称>点击创建。

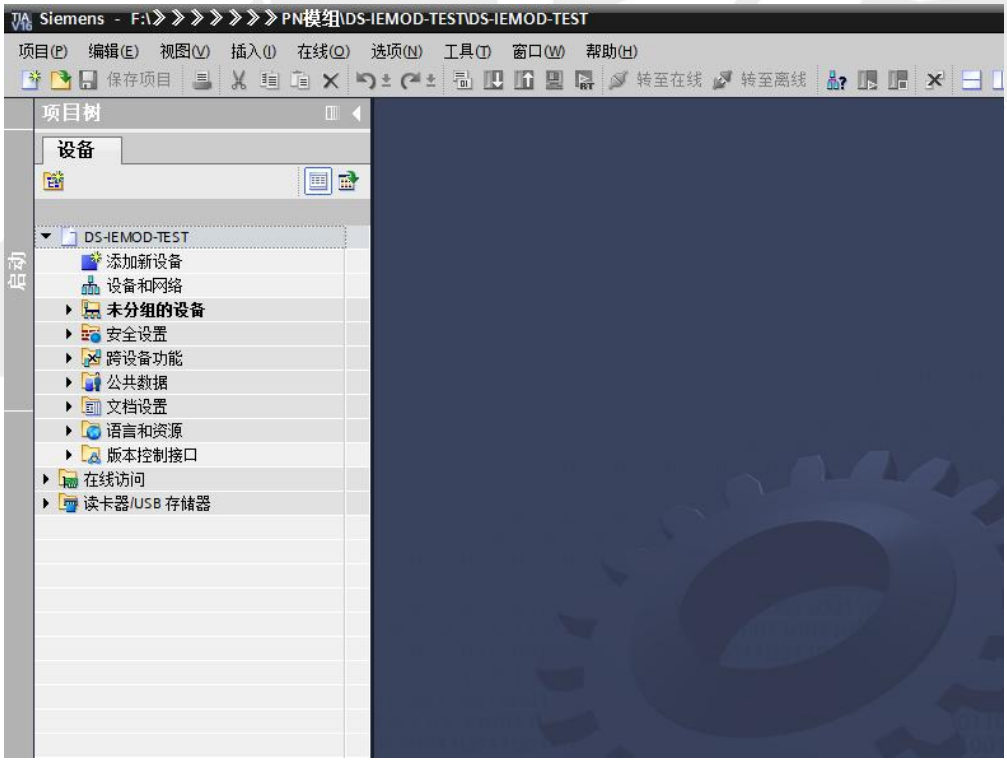
如下图 5-1 所示:



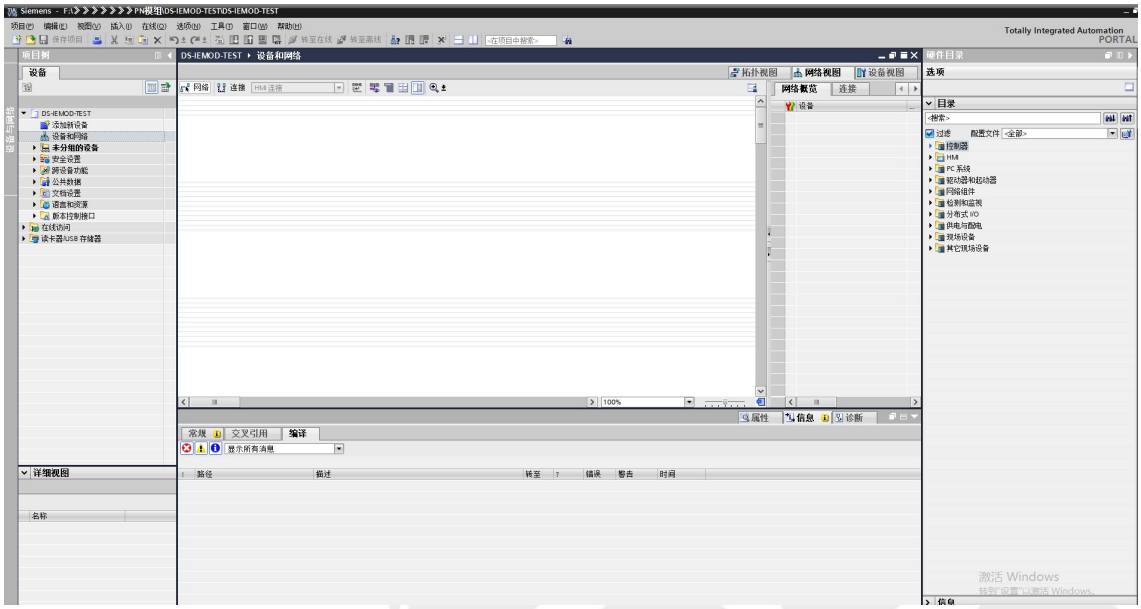
下一步，打开项目视图



下一步，打开设备和网络



进入主界面



7.3 博图安装 GSDML 文件

打开博图菜单>选项>管理通用站描述文件>源路径>选择 GSD 所在文件夹>点击安装。

如下图 5-1 所示:

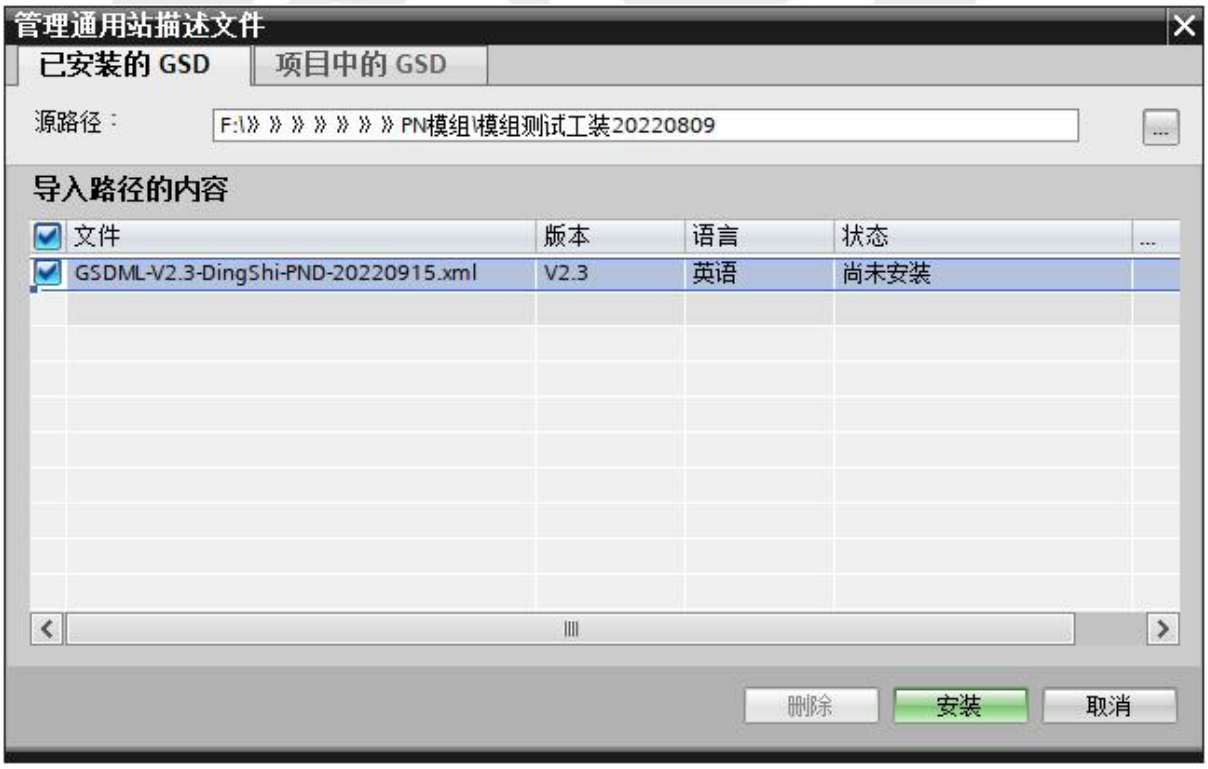


图 5-1

7.4 添加站点

7.4.1 添加主站

硬件目录>选择控制器>CPU>选择对应的 CPU>双击。

如下图 5-2 所示:

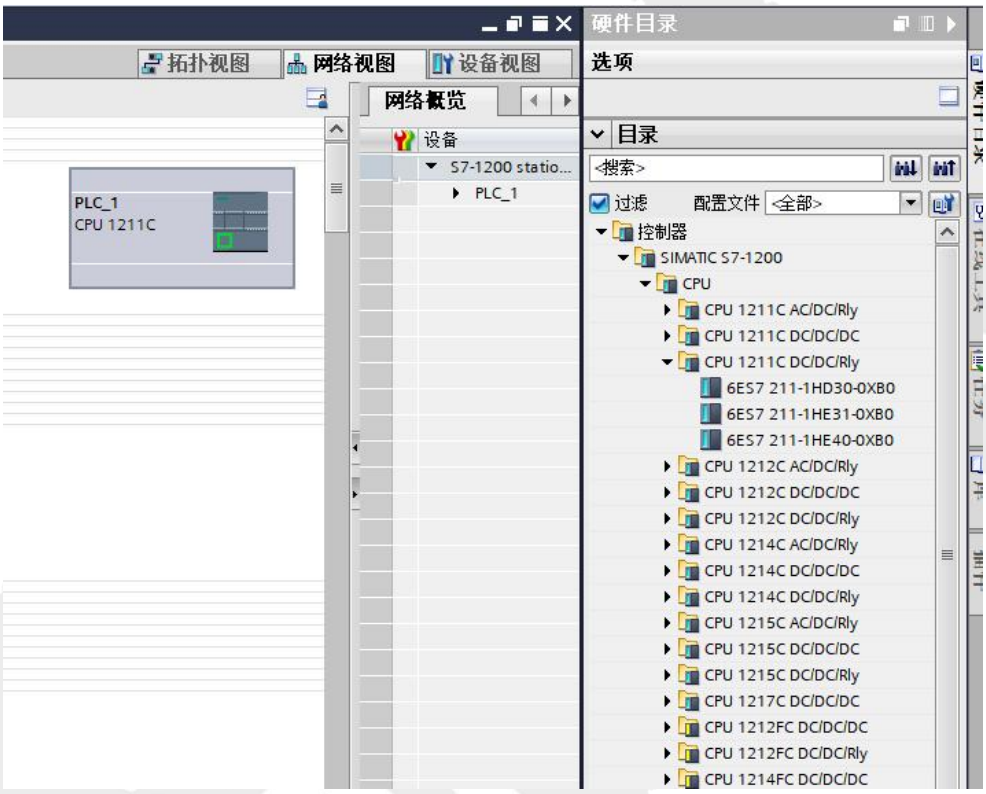


图 5-2

7.4.2 添加从站

硬件目录>其它现场设备>PROFINET IO>I/O>DS Co.Ltd>DEMO-DEVICE-TYPE>DS-IEMOD>双击。

如下图 5-3 所示:

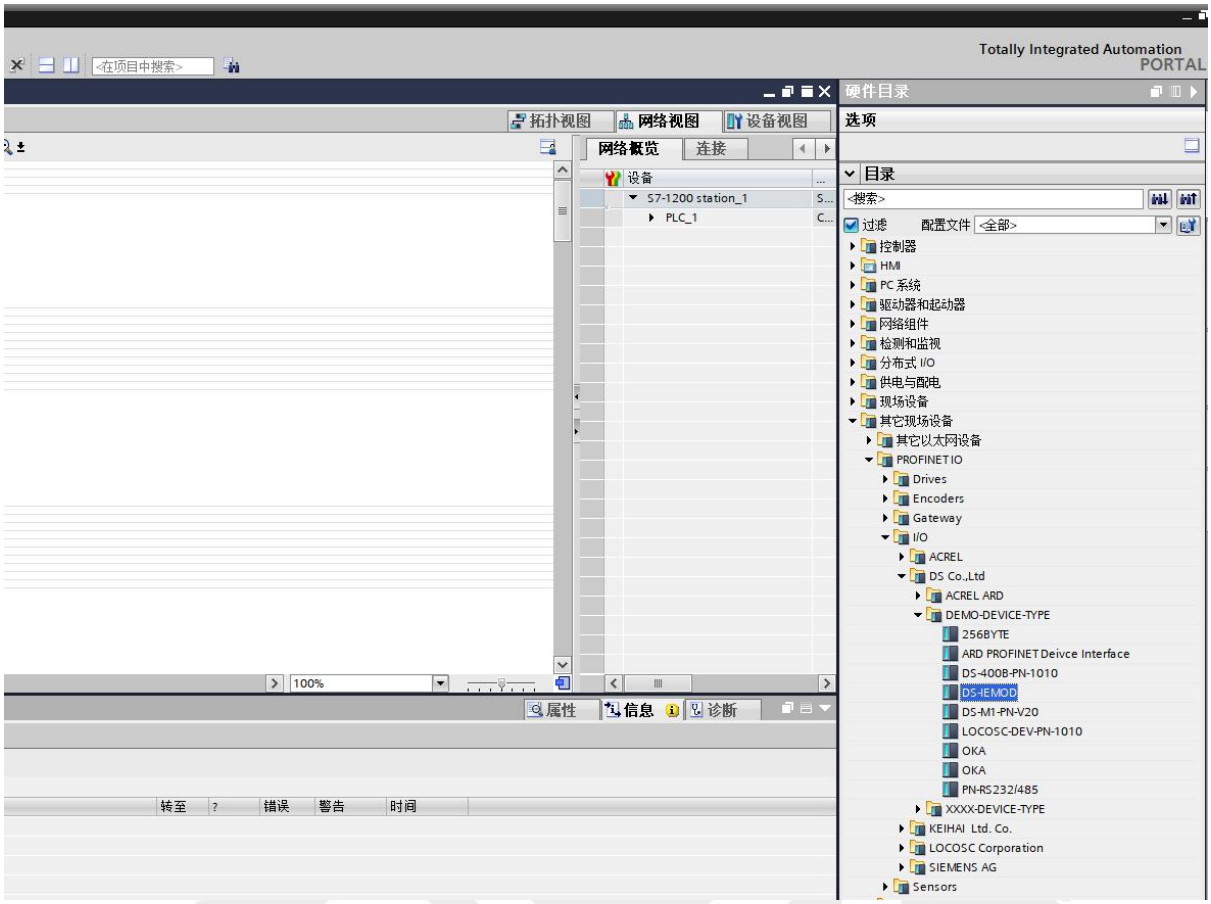


图 5-3

7.4.3 连接 PROFINET 网络

1、连接 PLC 与 PN 模组

如下图 5-4 所示:

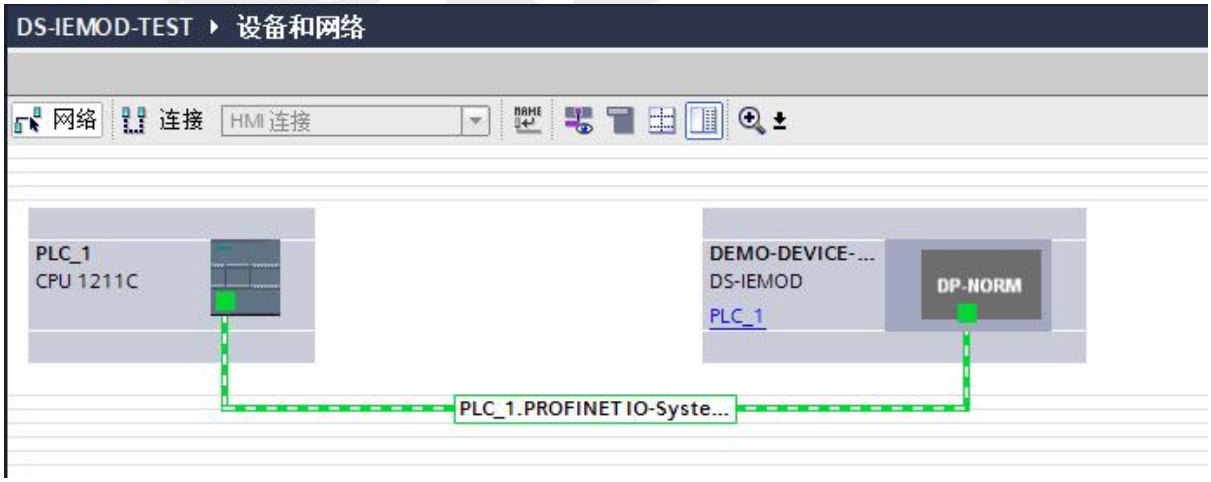


图 5-4

2、配置输入输出字节，插槽数

双击 DS-IEMOD>硬件目录>模块>DI/DO>选择自己需要的输入输出字节数量、插槽。

如下图 5-5 所示：

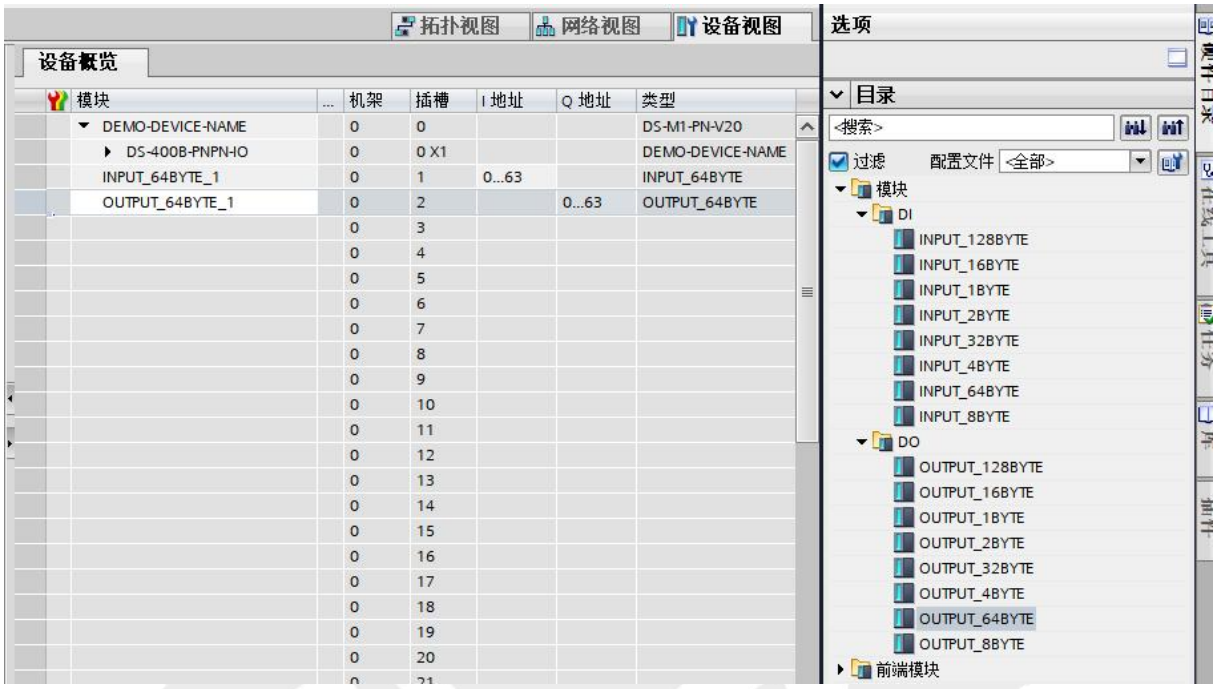


图 5-5

3、分配设备名

右键选择 DS-IEMOD>分配设备名>更新列表>分配名称。

如下图图 5-6 所示:

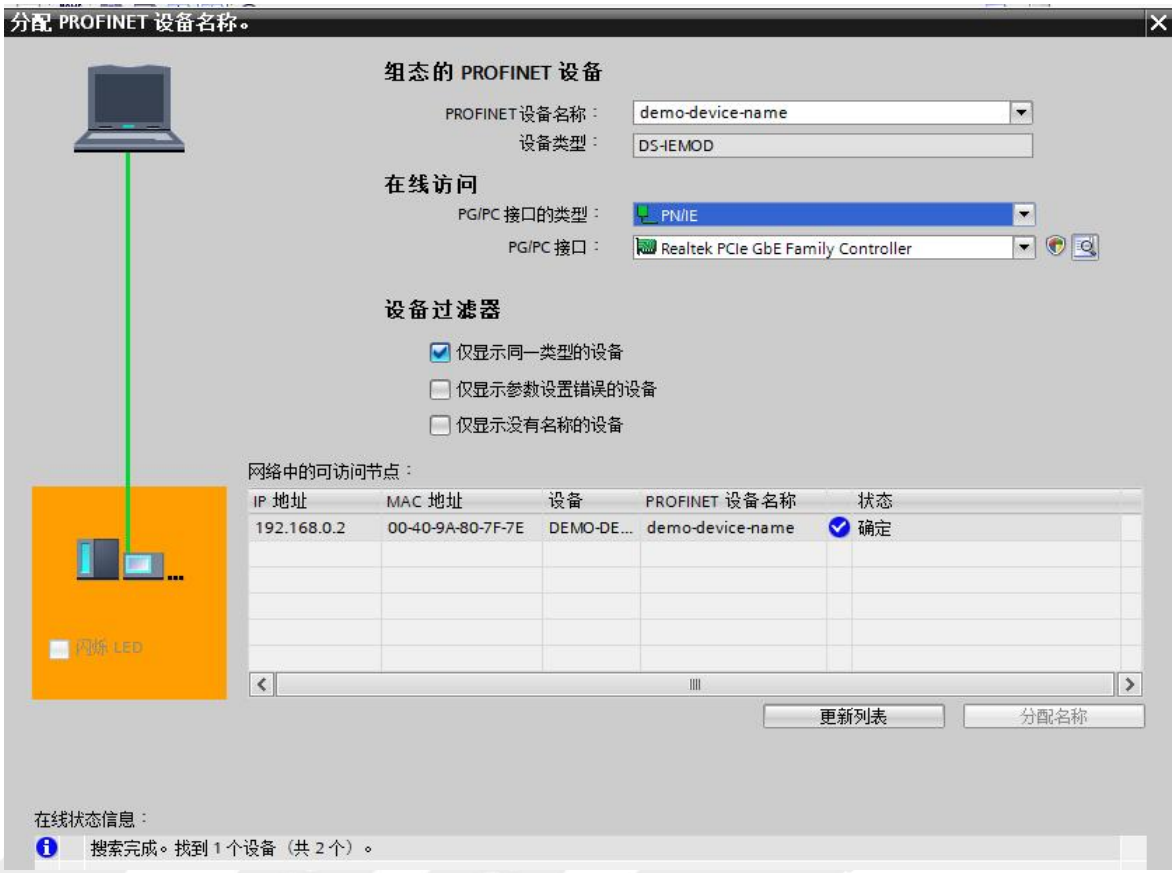
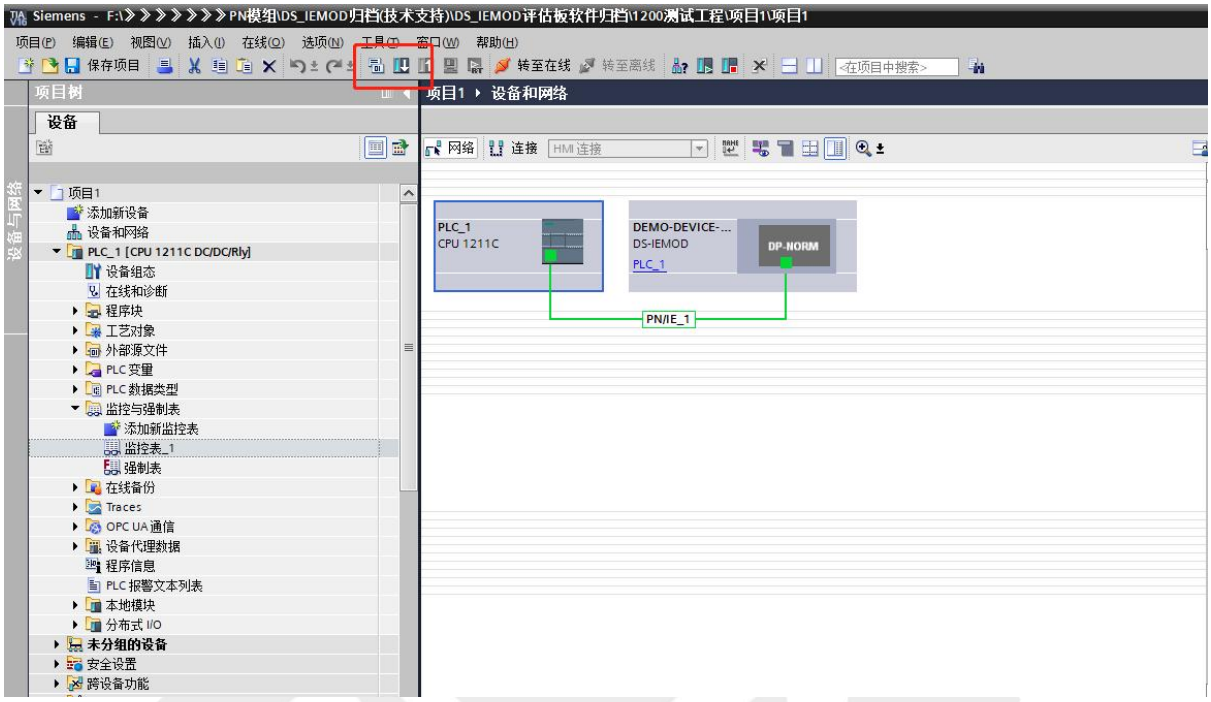


图 5-6

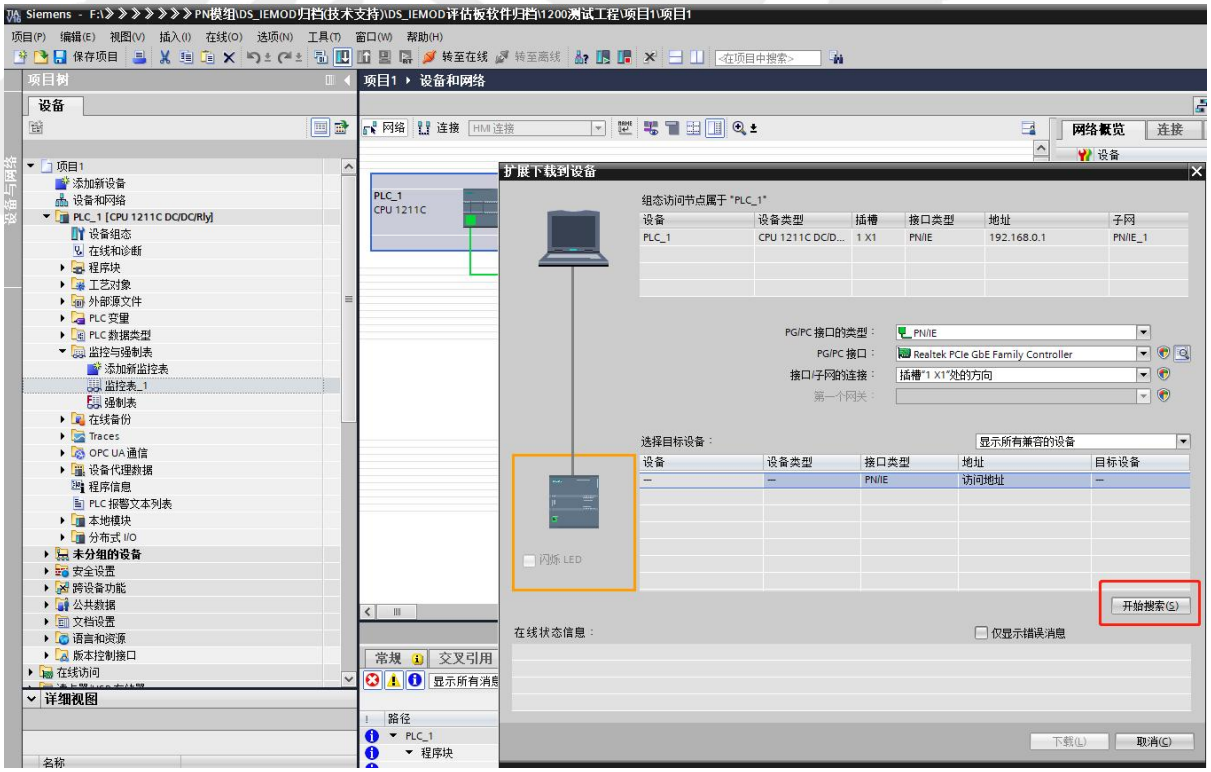
4、下载到设备

选中 PLC>点击下载程序图标>点击开始搜索>选中下载

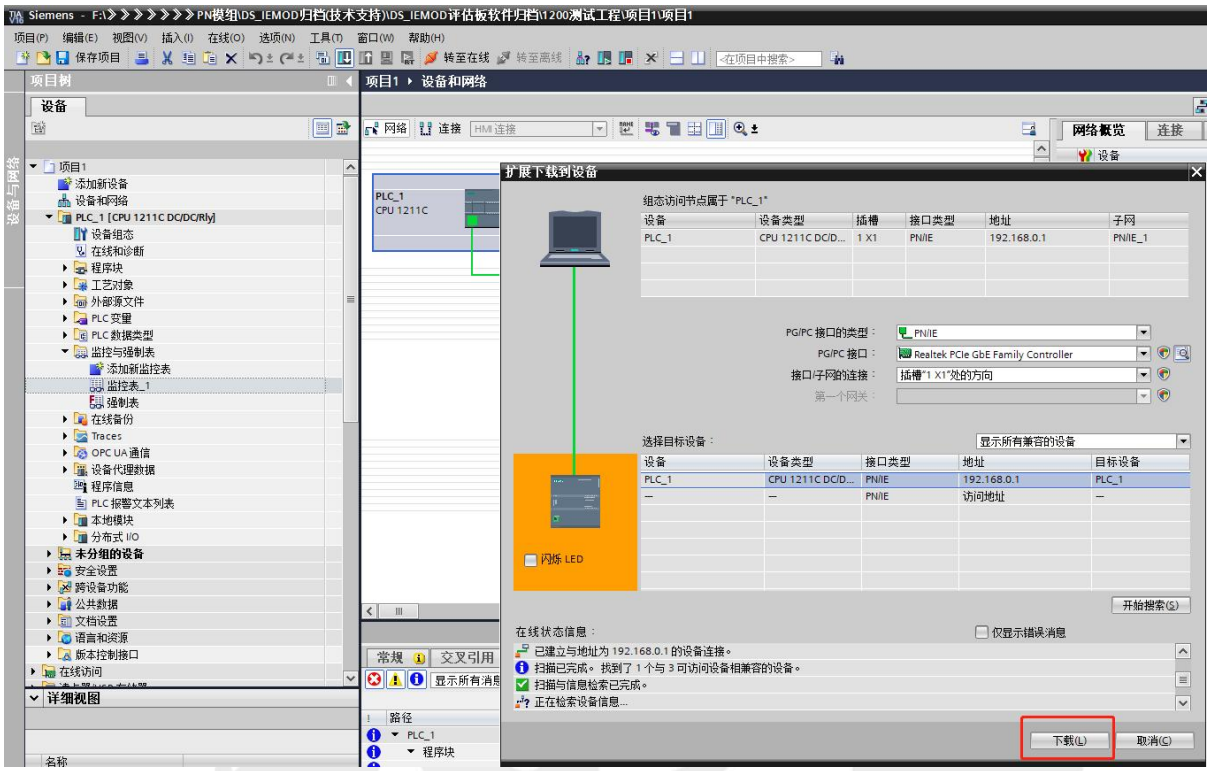
如下图 5-7 所示



下一步，点击开始搜索

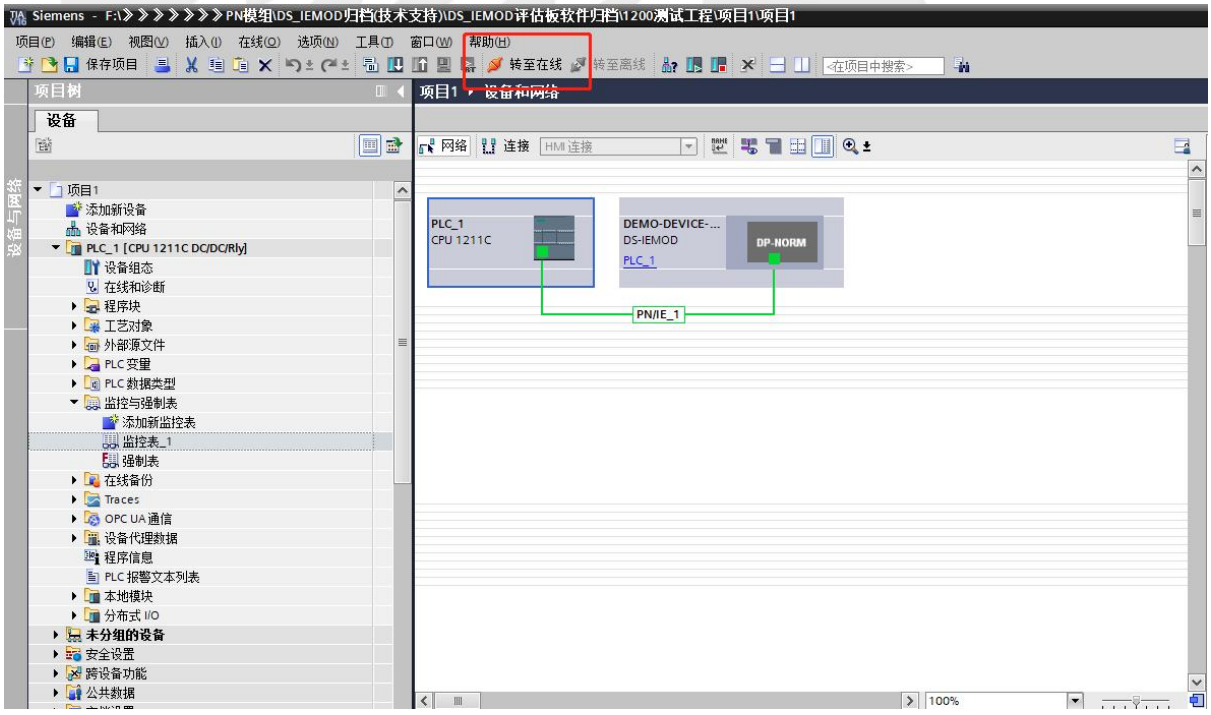


下一步，点击下载



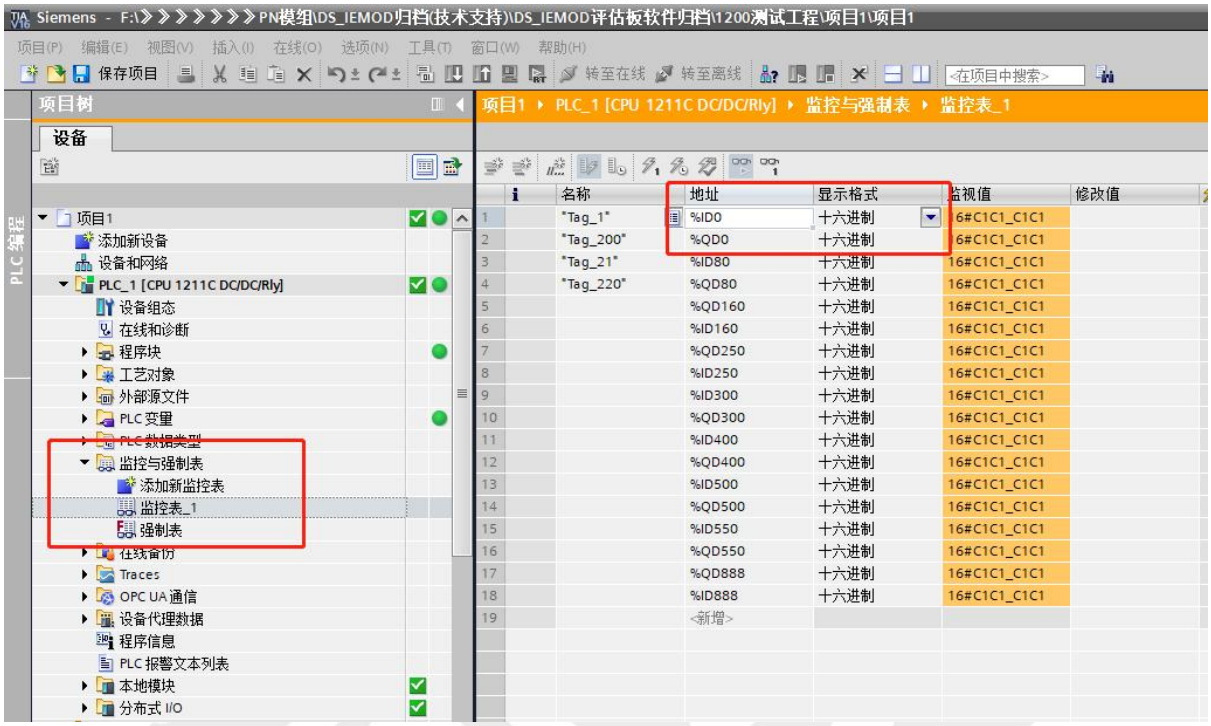
5、转至在线

如下图 5-7 所示

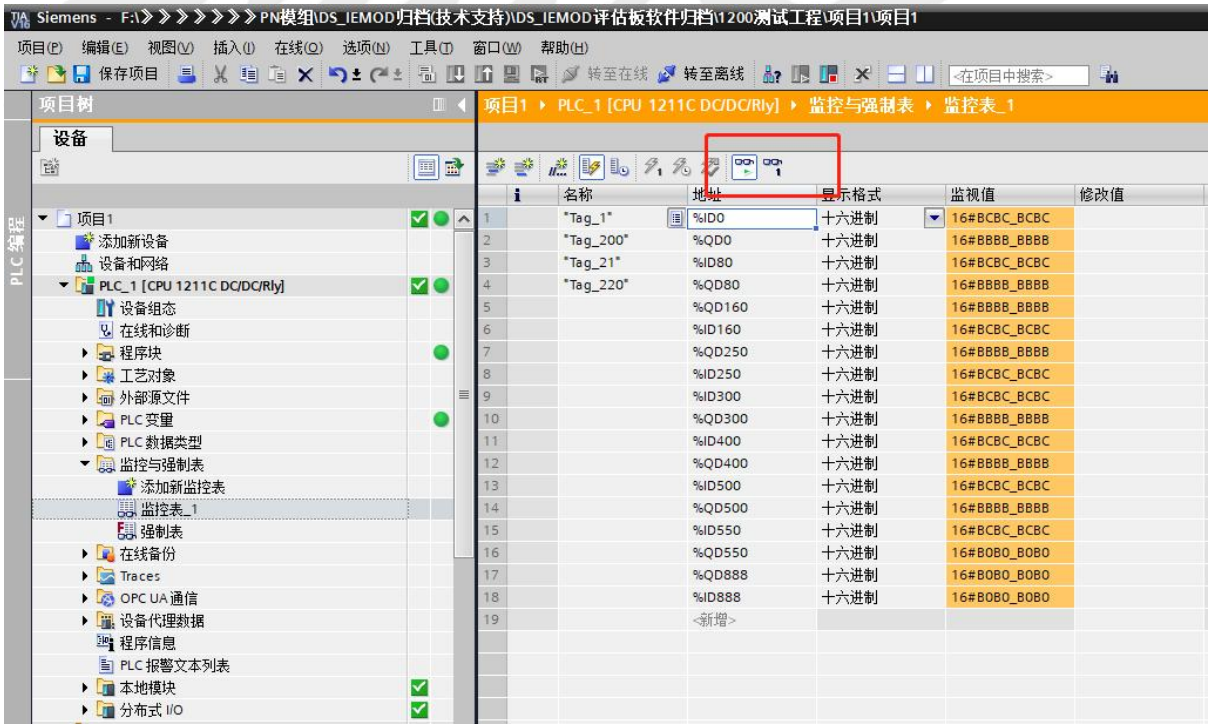


7.5 输入输出闭环测试

打开监控与强制表>添加新监控表>输入如图测试 Q 区 I 区地址

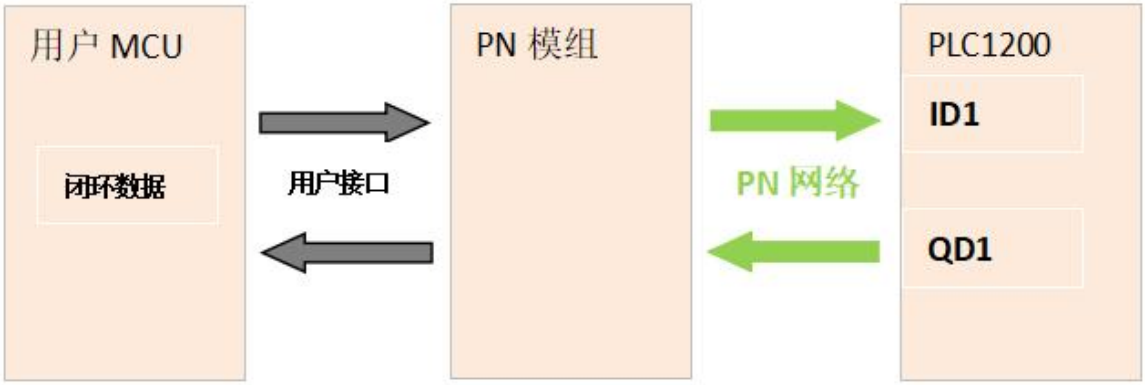


下一步，点击全部监视



7.6 PN 模组测试原理

QDxx 与对应的 IDxx 监视值规律变化，说明数据传输成功，输入输出数据流程图如下：



第八章 有毒有害物质表

部件名称	有毒有害物质和元素					
	铅（Pb）	汞（Hg）	镉（Cd）	六价铬（Cr（VI））	多溴联苯（PBB）	多溴二苯醚（PBDE）
塑料外壳	0	0	0	0	0	0
电路板	X	0	0	0	0	0
铜螺柱	0	0	0	0	0	0
贴膜	0	0	0	0	0	0
插座/插头	X	0	0	0	0	0
拨码开关	X	0	0	0	0	0

根据中国《电子信息产品污染控制管理办法》的要求出台

0：表示在此部件所用的所有同类材料中，所含的此有毒或有害物质均低于 SJ/T1163-2006 的限制要求；

X：表示在此部件所用的所有同类材料中，至少一种所含的此有毒或有害物质高于 SJ/T1163-2006 的限制要求。

注明：引用的“环保使用期限”是根据在正常温度和湿度条件下操作使用产品而确定的。

现场总线 PROFIBUS（中国）技术资格中心
北京鼎实创新科技股份有限公司
电话：010-82066344、010-82066355、010-82066377
地址：北京德胜门外新风街 2 号天成科技大厦 B 座 6001-6004　　邮编：100120
Web:www.c-profibus.com.cn　　Email: tangjy@c-profibus.com.cn