接口说明书

中科新松有限公司

May 17, 2024

CONTENTS

CHAPTER

ONE

TCP 及 IP 接口

机器人在上电状态下会作为 TCP Server 监听端口：2000 和 2001 的连接。端口 2000 可接收对机器人的控制命令及返回状态。端口 2001 会以 10Hz 的频率向外部发送机器人当前的状态信息。

1.1 2000 端口

2000 端口采用请求-响应式的应答机制，可以接受系统指令。Client 和 Server 间的通讯协议如下：

接收的指令	返回值	说明
run(程序名) 或 run(程序名, 速度百分比)	开始时反馈：run start 执行失败反馈：run fail	机器人会执行相应程序，不设置速度百分比或者 速度百分比参数无效时，使用当前速度运行。
speed(速度百分比)	执行成功反馈：set success 执行失败反馈：set fail	可修改程序运行的速度百分比。参数范围在（0,100], 若修改成功，会返回 set success，若不成功， 会返回 set fail
stop	执行成功反馈：stop success 执行失败反馈：stop fail	机器人会停止当前执行的指令。停止后会发送 stop success 回执。若机器人处于非运行状态，接收到 stop 命令后会发送 stop fail 回执
pause	执行成功反馈：pause success 执行失败反馈：pause fail	机器人会暂停当前执行的指令。停止时会发送 pause success 回执。若机器人处于非运行状态，接收到 pause 命令后会发送 pause fail 回执。
resume	执行成功反馈：resume success 执行失败反馈：resume fail	机器人在暂停时会继续运行，程序运行时会发送 resume success 回执。若机器人不在暂停状态，接收到 resume 命令后会发送 resume fail 回执
poweron	执行成功反馈：poweron success 执行失败反馈：poweron fail	机器人上电，若上电成功，建立通讯链接，返回 poweron success 若失败返回 poweron fail，若当前是已上电状态，返回 already poweron
poweroff	执行成功反馈：poweroff success 执行失败反馈：poweroff fail	机器人下电，若下电成功，返回 poweroff success， 若失败返回 poweroff fail，若当前是下电状态，返回 already poweroff
enable	执行成功反馈：enable success 执行失败反馈：enable fail	机器人上使能，若上使能成功，返回 enable success 若失败返回 enable fail，若当前是使能状态， 返回 already enable
disable	执行成功反馈：disable success 执行失败反馈：disable fail	机器人下使能，若下使能成功，返回 disable success 若失败返回 disable fail，若当前是已上电且未使能 状态，返回 already disable
shutdown	执行成功反馈：shutdown success 执行失败反馈：shutdown fail	机器人已处于 poweroff 状态，机器人系统会自动关闭 返回 shutdown success 回执。若机器人未 下电，返回 shutdown fail 回执

1.2 2001 端口

端口 2001 会以 10Hz 的频率向外部发送机器人当前的状态信息，数据总长度 1468 个字节。

数据	类型	数量	字节大小	地址	注释
关节实际位置	float	7	28	0-27	6 关节各自实际位置，第 7 位预留。单位为 rad
关节实际速度	float	7	28	28-55	6 关节各自实际速度，第 7 位预留. 单位为 rad/s
关节实际加速度	float	7	28	56-83	6 关节各自实际加速度，第 7 位预留. 单位为 rad/
关节实际力矩	float	7	28	84-111	6 关节各自实际力矩，第 7 位预留。单位为 Nm
关节期望位置	float	7	28	112-139	6 关节各自设定位置，第 7 位预留。单位为 rad
关节期望速度	float	7	28	140-167	6 关节各自设定速度，第 7 位预留。单位为 rad/s
关节期望加速度	float	7	28	168-195	6 关节各自设定加速度，第 7 位预留。单位为 rad/
关节期望力矩	float	7	28	196-223	6 关节各自设定力矩，第 7 位预留。单位为 Nm 单位为 Nm
关节实际温度	float	7	28	224-251	7 关节各自实际温度。数据预留
关节实际电流	float	7	28	252-279	6 关节各自实际电流，第 7 位预留。单位为额定电流千分比。
伺服驱动错误 id	uint	7	28	280-307	6 关节伺服驱动错误 id，第 7 位预留。

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
伺服驱动状态字	uint	7	28	308-335	6 关节伺服驱动状态字，第 7 位预留
预留	byte	32	32	336-367	
TCP 实际位置	float	6	24	368-391	末端笛卡尔空间实际位姿。TCP 相对基坐标系的值。数据定义：X，Y，Z,Rx,Ry,Rz，单位为 m，rad
TCP 实际速度	float	6	24	392-415	末端笛卡尔空间实际速度。TCP 相对基坐标系的值。数据定义：X，Y，Z,Rx,Ry,Rz，单位为 m，rad/s
TCP 实际加速度	float	6	24	416-439	末端笛卡尔空间实际位姿。TCP 相对基坐标系的值。数据定义：X，Y，Z,Rx,Ry,Rz，单位为 m/s2，rad/s2
法兰实际外力	float	6	24	440-463	机器人末端法兰受到的外力，当配置末端力矩传感器时，该值为传感器数据在机器人末端法兰面下的力描述。据定义:X，Y，Z,Rx,Ry,Rz，单位为 N，Nm

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
TCP 期望位置	float	6	24	464-487	末端笛卡尔空间实际位姿。TCP 相对基坐标系的值。数据定义：X，Y, Z,Rx,Ry,Rz, 单位为 m，rad
TCP 期望速度	float	6	24	488-511	末端笛卡尔空间实际速度。TCP 相对基坐标系的值。数据定义：X，Y, Z,Rx,Ry,Rz, 单位为 m/s，rad/s
TCP 期望加速度	float	6	24	512-535	末端笛卡尔空间实际速度。TCP 相对基坐标系的值。数据定义：X，Y, Z,Rx,Ry,Rz, 单位为 m/s2，rad/s2
法兰理论外力	float	6	24	536-559	机器人末端法兰在当前运动状态下，理论收到的外力大小。数据定义：X，Y, Z,Rx,Ry,Rz, 单位为 N，Nm

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
基座实际外力	float	6	24	560-583	底座实际受力在机器人基座下的力描述。当配置底座力矩传感器时，该值为力矩传感器测量值，数据定义：X，Y，Z,Rx,Ry,Rz，单位为N，Nm
基座理论外力	float	6	24	584-607	在当前运动状态下，底座理论受力在机器人基座下的力描述。数据定义：X，Y，Z,Rx,Ry,Rz，单位为N，Nm
当前激活的工具坐标系	float	6	24	608-631	数据定义：X，Y，Z,Rx,Ry,Rz，单位为m，rad
当前激活的工件坐标系	float	6	24	632-655	不包含当前工具坐标系偏移量. 数据定义：X，Y，Z,Rx,Ry,Rz，单位为m，rad
合线速度	float	1	4	656-659	末端合线速度，单位为m/s
全局速度	byte	1	1	660	全局速度百分比
Jog 速度	byte	1	1	661	Jog 速度百分比
预留	byte	58	58	662-719	

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
功能数字 IO 输入	byte	8	8	720-727	以一个 byte 表示一个 bool 量。True=1,false=0。对应硬件 FDI1-FDI8
功能数字 IO 输出	byte	8	8	728-735	以一个 byte 表示一个 bool 量。True=1,false=0。对应硬件 FDO1-FDO8
数字 IO 输入	byte	16	16	736-751	对应 DI 接口普通 DI 输入 1-16。对应硬件 DI1-DI16
数字 IO 输出	byte	16	16	752-767	对应 DO 接口普通 DO 输出 1-16。对应硬件 DO1-DO16
模拟量输入	float	8	32	768-799	8 个模拟量输入，前 4 路为电流、后 4 路为电压。对应硬件 AI_C1-AI_C4,AI_V1-AI_V4。取值范围详见硬件说明书。
模拟量输出	float	8	32	800-831	预留
float 寄存器输入	float	32	128	832-959	可通过 modbus、Profinet 等外部接口修改
float 寄存器输出	float	32	128	960-1087	可通过脚本函数修改
功能 bool 寄存器输入	byte	16	16	1088-1103	可通过 modbus、Profinet 等外部接口修改
功能 bool 寄存器输出	byte	16	16	1104-1119	可通过脚本函数修改

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
bool 寄存器输入	byte	64	64	1120-1183	可通过 modbus、Profinet 等外部接口修改
bool 寄存器输出	byte	64	64	1184-1247	可通过脚本函数修改
word 寄存器输入	char	32	64	1248-1311	可通过 modbus、Profinet 等外部接口修改
word 寄存器输出	char	32	64	1312-1375	可通过脚本函数修改
预留	byte	32	31	1376-1406	
仿真/真机	byte	1	1	1407	机器人仿真/真机模式
工具 IO 输入	byte	8	8	1408-1415	工具数字输入（1-2 末端输入，3-8 预留）。对应硬件工具 IO 数字输入 1-2
工具 IO 输出	byte	8	8	1416-1423	工具数字输出（1-2 末端输出，3-8 预留）。对应硬件工具 IO 数字输出 1-2
工具模拟量输入	float	2	16	1424-1431	工具末端电压模拟量输入。取值范围 0V-10V
工具模拟量输出	float	2	16	1432-1439	预留
工具按钮状态	byte	2	2	1440-1441	以一个 byte 表示一个 bool 量。True=1,false=0。地址 1440 表示 S 键的状态，地址 1441 表示 T 键的状态。
预留	byte	6	6	1442-1447	
机器人操作模式	char	1	1	1448	0: kManual,1: kAuto,2: kRemote,

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
机器人状态	char	1	1	1449	0: SR_Start,1: SR_Initialize, 2:SR_Logout, 3: SR_Login, 4: SR_PowerOff, 5: SR_Disable/ SR_ PowerOn（上电未使能状态）6: SR_Enable
机器人程序运行状态	char	1	1	1450	0: SP_Stopped, 1: SP_Stopping, 2: SP_Running, 3: SP_Paused, 4: SP_Pausing5: SP_TaskRunning
安全监控状态	char	1	1	1451	0: SS_INIT2: SS_WAIT3: SS_CONFIG 4: SS_POWER_OFF5: SS_RUN6: SS_RECOVERY 7: SS_STOP28: SS_STOP19: SS_STOP0 10: SS_MODEL12: SS_REDUCE13: SS_BOOT 14: SS_FAIL15: SS_UPDATE
碰撞检测触发信号	char	1	1	1452	为 1 表示触发碰撞检测

continues on next page

Table 1 – continued from previous page

数据	类型	数量	字节大小	地址	注释
碰撞轴	char	1	1	1453	返回值为数字。1~6 表示发生碰撞的关节，11~16 表示基于底座力传感器碰撞发生的方向。顺序为 X , Y, Z,Rx,Ry,Rz . 20 表示基于末端力传感器碰撞发生
预留	byte	2	2	1454-1455	
机器人错误代码	uint	1	4	1456-1459	机器人最新报错的错误代码，当同时出现多个错误时，可能返回其中的任何一个错误代码。
预留	byte	8	8	1460-1467	

CHAPTER

TWO

MODBUS TCP SERVER

本章描述机器人端作为 Modbus/TCP server 端口时的可交互数据。线圈（Colis）：读（功能码：1）写（功能码：5 单个/15 多个）

Addr	节点说明	数量	读写权限	功能
0-15	功能 bool 寄存器输入	16	R/W	外部修改寄存器，触发功能
16-31	功能 bool 寄存器输出	16	R	寄存器值改变，表示对应系统状态
32-95	bool 寄存器 输入	64	R/W	外部修改寄存器，在脚本中读取
96-159	bool 寄存器 输出	64	R	脚本或 UI 中修改寄存器，输出到外部
160-167	功能数字 IO 输入	8	R	外部物理 IO 输入，触发功能
168-175	功能数字 IO 输出	8	R	物理 IO 输出，表示对应系统状态
176-191	数字 IO 输入	16	R	外部物理 IO 输入，在（脚本/UI/外部端口）中读取
192-207	数字 IO 输出	16	R/W	（脚本/UI/外部端口）中修改 IO，输出到外部物理 IO
208-215	工具 IO 输入	8	R	外部物理 IO 输入，在（脚本/UI/外部端口）中读取
216-223	工具 IO 输出	8	R/W	（脚本/UI/外部端口）中修改 IO，输出到外部物理 IO
224-231	工具末端按钮	8	R	工具末端按钮输入，触发功能

Holding Registers: 读（功能码：3）写（功能码：6 单个/16 多个）

Addr	节点说明	读写权限
0	个位：机器人操作模式 0：kManual 1：kAuto 2：kRemote 百位：仿真/真机模式 0：真机模式 1：仿真模式	R

Addr	节点说明	读写权限
1	机器人状态 0：SR_Start 1：SR_Initialize 2：SR_Logout 3：SR_Login 4：SR_PowerOff 5：SR_Disable/SR_PowerOn（上电未使能状态） 6：SR_Enable	R
2	程序状态 0：SP_Stopped 1：SP_Stopping 2：SP_Running 3：SP_Paused 4：SP_Pausing 5：SP_TaskRunning	R
3	机器人安全状态 0：SS_INIT 2：SS_WAIT 3：SS_CONFIG 4：SS_POWER_OFF 5：SS_RUN 6：SS_RECOVERY 7：SS_STOP2 8：SS_STOP1 9：SS_STOP0 10：SS_MODEL 12：SS_REDUCE 13：SS_BOOT 14：SS_FAIL 15：SS_UPDATE	R
4-6	保留	R

Addr	节点说明	读写权限
7	末端合线速度	R
8-39	World 输入寄存器	R/W
40-71	World 输出寄存器	R
72-79	控制柜模拟输入,采用 count 表示，实际电流值为 count/100，电压值为 count/1000。	R
80-81	控制柜模拟电流输出采用 count 表示，实际电流值为 count/100。	R/W
82-83	预留	R/W
84-85	控制柜模拟电压输出,采用 count 表示，实际电压值为 count/1000。	R/W
86-87	预留	R/W
88-89	末端模拟输入,采用 count 表示，实际电流值为 count/100，电压值为 count/1000。	R
90	全局速度百分比	R/W
91	Jog 速度百分比	R
92-98	关节实际位置，取值-32767-32767, 单位 1/1000rad。	R
99-105	关节实际速度，取值-32767-32767, 单位 1/1000rad/s。	R
106-111	空间实际位置，取值-32767-32767,单位 1/10000m；1/10000 rad	R
112-117	空间实际速度，取值-32767-32767,单位 1/10000(m/s);1/10000(rad/s)	R

CHAPTER

THREE

配方接口

配方分类:

（1）根据功能，分为用户自定义配方（非实时配方）和系统数据配方（实时配方）：自定义配方：由用户自己定义想要接收和发送的数据格式，由配方管理器提取用户的输入有效数据和对输出数据进行组帧。接收和发送的过程均由用户在脚本程序中控制。系统数据配方：用户从系统提供的系统数据中选择想要的数据通过输出端进行实时读取，或者通过输入端实时修改机器人提供的系统变量，用户可配置的变量均由系统提供，自用无法自定义，数据的接收法发送不受用户控制，由系统实时控制，控制周期（发送和接收的最快周期）为 4ms。

（2）根据协议，分为流式协议配方和 can 协议配方：流式协议配方为流式数据（485、tcp 等）服务。

can 协议配方仅提供对 can 的支持。

3.1 系统数据映射说明

配方的数据类型:

（1）byte(无符号 8 位整形)

（2）char(有符号 8 位整形)

（3）word(无符号 16 位整形)

（4）short(有符号 16 位整形)

（5）dword(无符号 32 位整形)

（6）int(有符号 32 位整形)

（7）float(32 位浮点数)

（8）double(64 位双精度浮点数)

系统数据输入:

数据方向：用户 ? 机器人

配方变量	类型	对应的接口	说明
DigitalOutputCommand[1..2]	byte	控制柜通用输出 DO[1..16]	用户控制机器人的 DO。一个 byte 顺序对应 8 个 DO。
DigitalOutputCommand[3]	byte	机械臂末端工具输出 Tool_DO[1..2]	byte 的前两位对应工具 DO1 和 DO2，后六位预留。
BitInputReg[1..2]	byte	寄存器功能输入 fun_reg_in[1..16]	一个 byte 顺序对应 8 个寄存器功能输入
BitInputReg[3..10]	byte	寄存器 Bool 输入 bool_reg_in[1..64]	一个 byte 顺序对应 8 个寄存器 Bool 输入
WordInputReg[1..32]	word	寄存器 Word 输入 word_reg_in[1..32]	按顺序对应寄存器 Word 输入 1-32
FloatInputReg[1..32]	float	寄存器 Float 输入 float_reg_in[1..32]	按顺序对应寄存器 Float 输入 1-32
ForceEnd[1..6]	float	末端力传感器数据 控制柜通用输	末端力传感器数据在机器人末端 TCP 下的力描述。数据定义：X , Y, Z,Rx,Ry,Rz, 单位为 N，Nm
ForceBase[1..6]	float	基座力传感器数据	底座力传感器数据在机器人基座下的力描述。数据定义：X , Y, Z,Rx,Ry,Rz, 单位为 N，Nm

系统数据输出：

数据方向：机器人 ? 用户

配方变量	类型	对应的接口	说明
DigitalSignalStatus[1..2]	byte	控制柜通用输入 DI[1..16]	机器人向用户反馈当前机器人的输入信号。一个 byte 顺序对应 8 个 DI
DigitalSignalStatus[3]	byte	机械臂末端工具输入 Tool_DI[1..2]	byte 的前两位对应工具 DI1 和 DI2，后六位预留
DigitalSignalStatus[4..5]	byte	控制柜通用输出 DO[1..16]	机器人向用户反馈当前机器人的输出信号。一个 byte 顺序对应 8 个 DO
DigitalSignalStatus[6]	byte	工具输出 Tool_DO [1..2]	byte 的前两位对应工具 DO1 和 DO2，后六位预留
DigitalSignalStatus[7]	byte	IO 功能输入 fun_io_in[1..8]	一个 byte 顺序对应功能输入的 1-8
DigitalSignalStatus[8]	byte	IO 功能输出 fun_io_out [1..8]	一个 byte 顺序对应功能输出的 1-8
RobotState[1..7]	float	关节实际位置	6 关节各自实际位置，第 7 位预留。单位为 rad
RobotState[8..13]	float	末端在基坐标系下的实际笛卡尔位姿	末端笛卡尔空间实际位姿。TCP 相对基坐标系的值。数据定义：X,Y,Z,Rx,Ry,Rz, 单位为 m，rad
RobotState[14..19]	float	笛卡尔实际力矩	末端力传感器数据在机器人末端法兰面下的力描述。数据定义：X,Y,Z,Rx,Ry,Rz, 单位为 N，Nm
RobotState[20..25]	float	工具坐标系偏移量	工具坐标系下的位姿偏移量
RobotState[26..29]	float	负载质量和质心	末端负载的质量和质心，单位 kg,m
RobotState[30..32]	float	预留	无
BitOutputReg[1..2]	byte	寄存器功能输出 fun_reg_out[1..16]	一个 byte 顺序对应 8 个寄存器功能输出
BitOutputReg[3..10]	byte	寄存器 Bool 输出 bool_reg_out[1..64]	一个 byte 顺序对应 8 个寄存器 Bool 输出
WordOutputReg[1..32]	word	寄存器 Word 输出 word_reg_out[1..32]	按顺序对应寄存器 Word 输入 1-32
FloatOutputReg[1..32]	float	寄存器 Float 输出 float_reg_out[1..32]	按顺序对应寄存器 Float 输入 1-32

3.2 配方的创建

3.2.1 流式数据配方

用户自定义配方

1、在 ui 界面选择任意流式端口（485、tcp 等）配方管理（这里选择 485 端口）

2、点击添加，输入配方名，选择非实时配方

3、添加输入数据项

4、添加输出数据项

点击确认，配方创建完成。注意：输入与输出的帧头帧尾，是相对独立的，需单独配置，互不影响。

系统数据配方

1、在 ui 界面选择任意流式端口（485、tcp 等）配方管理（这里选择 485 端口）

2、点击添加，输入配方名，选择非实时配方

3、添加输入数据项

4、添加输出数据项

5、点击确认，配方创建完成。

3.2.2 CAN 协议配方

用户自定义配方

1、在 ui 界面选择 can 端口配方管理

2、点击添加，输入配方名，选择非实时配方

3、添加输入数据项

4、添加输出数据项

点击确认，配方创建完成。

系统数据配方

1、在 ui 界面选择 can 端口配方管理

2、点击添加，输入配方名，选择实时配方

3、添加输入数据项

4、添加输出数据项

5、点击确认，配方创建完成。

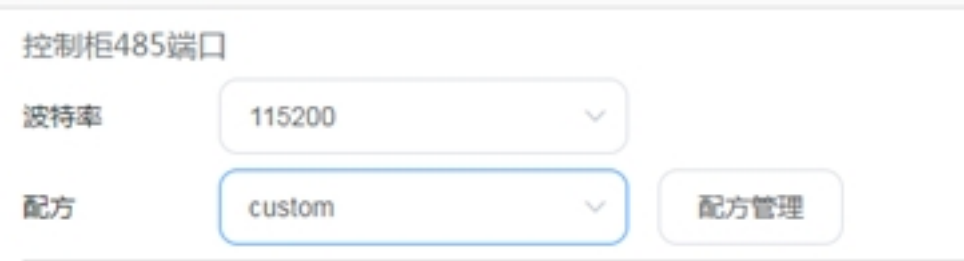
3.3 配方的使用

3.3.1 485 端口配方的使用

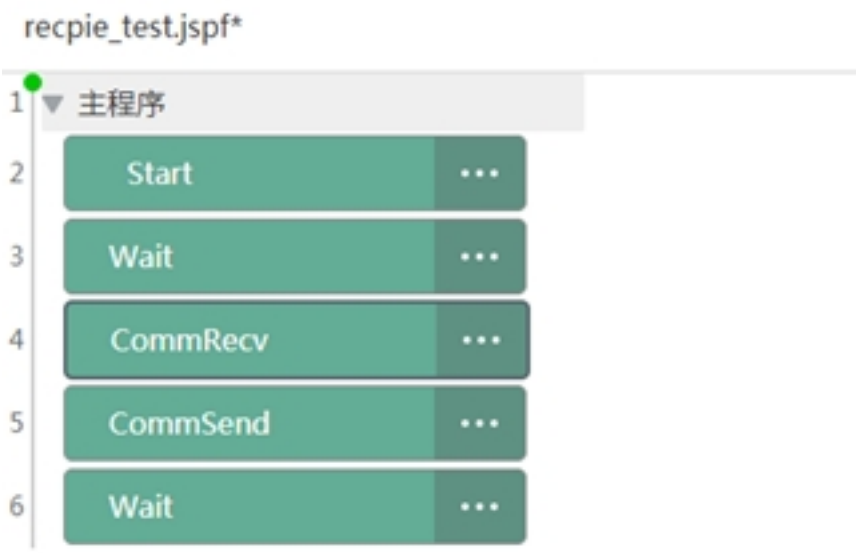
自定义配方使用

1、使用 485 转串口调试助手将电脑和机器人 485 端口连接

2、为 485 端口选择配方文件



3、编写程序流程控制



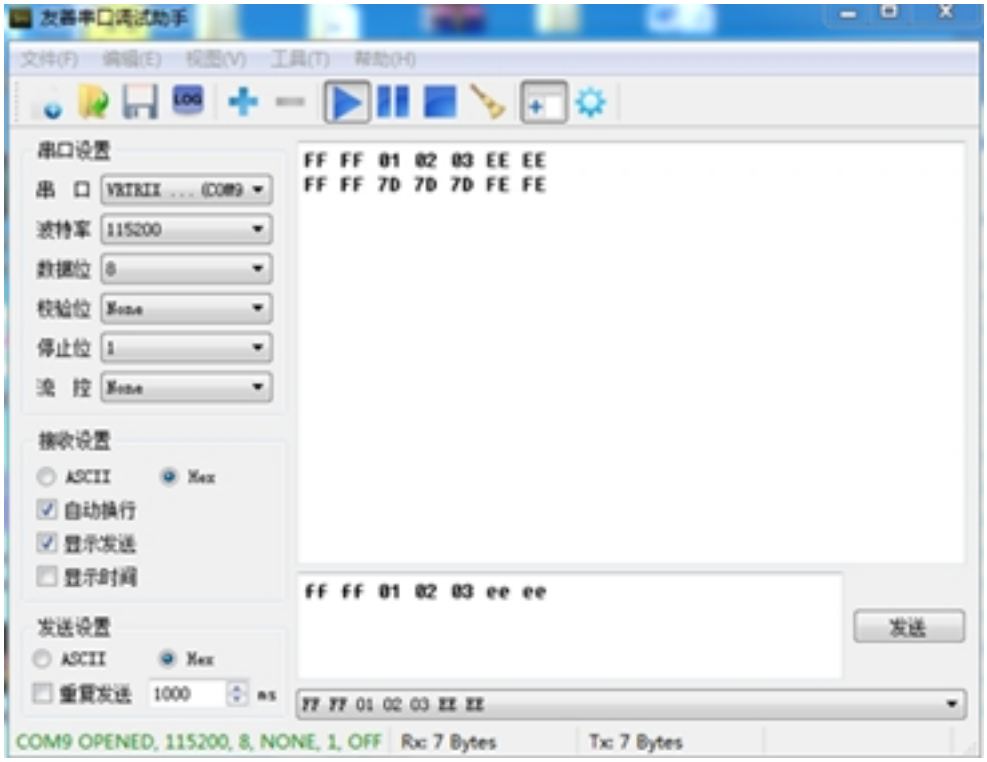
CommRecv:

CommSend:

4、运行程序，串口调试助手发送 FF FF 01 02 03 EE EE

5、程序变量界面显示

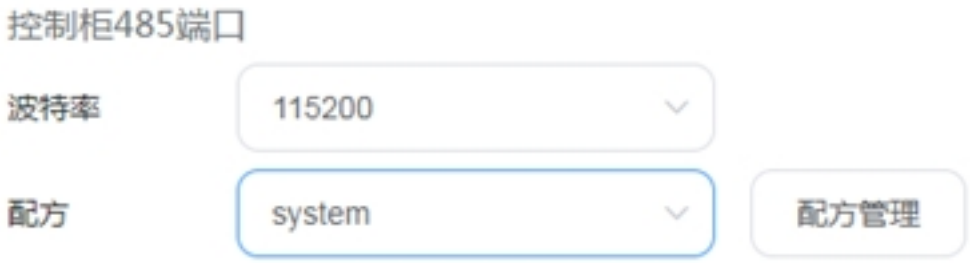
6、在串口调试助手端将会收到：FF FF 7D 7D 7D FE FE



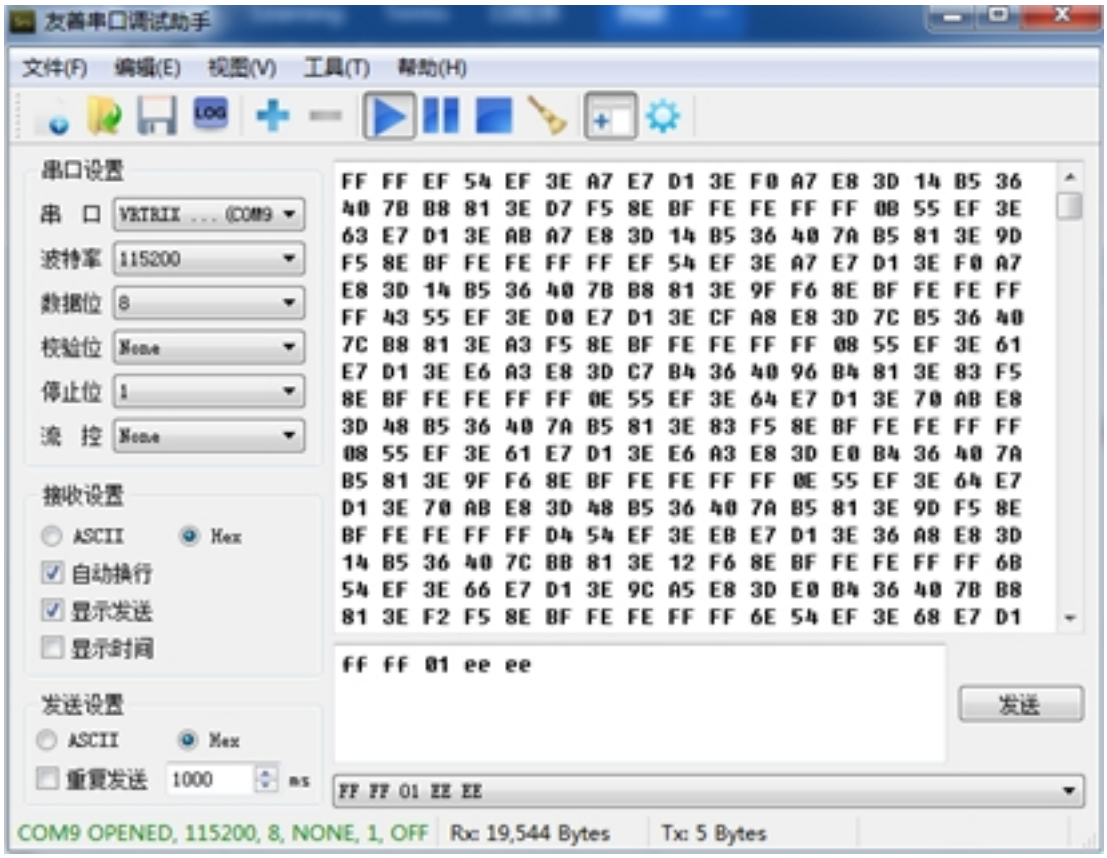
系统配方使用

1、使用 485 转串口调试助手将电脑和机器人 485 端口连接

2、为 485 端口选择配方文件



3、由于选择的是实时系统配方，串口调试助手将会周期性的收到机器人位姿数据:



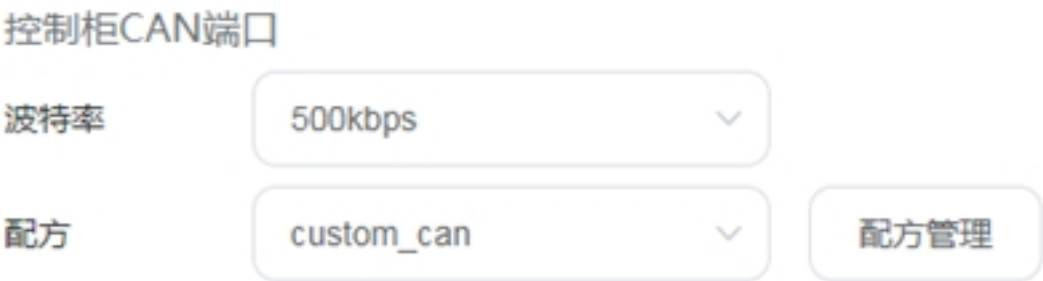
4、当通过串口调试助手发送 FF FF 01 EE EE 时，通过配方管理器将会控制对应的 IO 端口

3.3.2 CAN 端口配方的使用

自定义配方使用

1、使用 can 盒将电脑和机器人连接，使用 CanTest 调试 can

2、为 can 端口选择配方文件



3、编写程序流程控制

CommRecv:

CommSend:

4、运行程序，CanTest 发送数据帧 id 1, 数据 01 02 03

5、程序变量界面显示

程序变量

名称	类型	当前值
list1	num_list	{1,1,2,3}

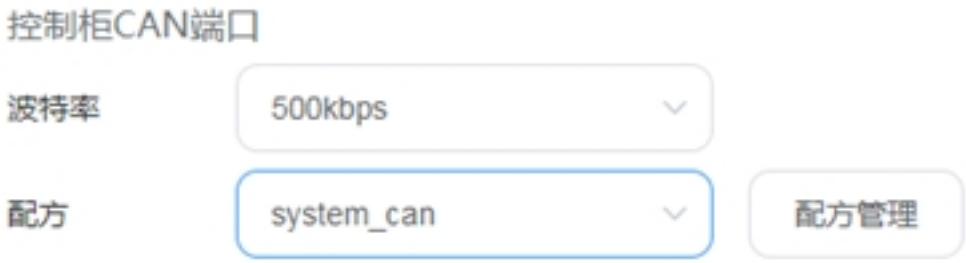
列表第一个变量为收到的数据帧 id，后面的为数据。

6、在 CanTest 端将会收到帧 id 为 1：7D 7D 7D(125 125 125)

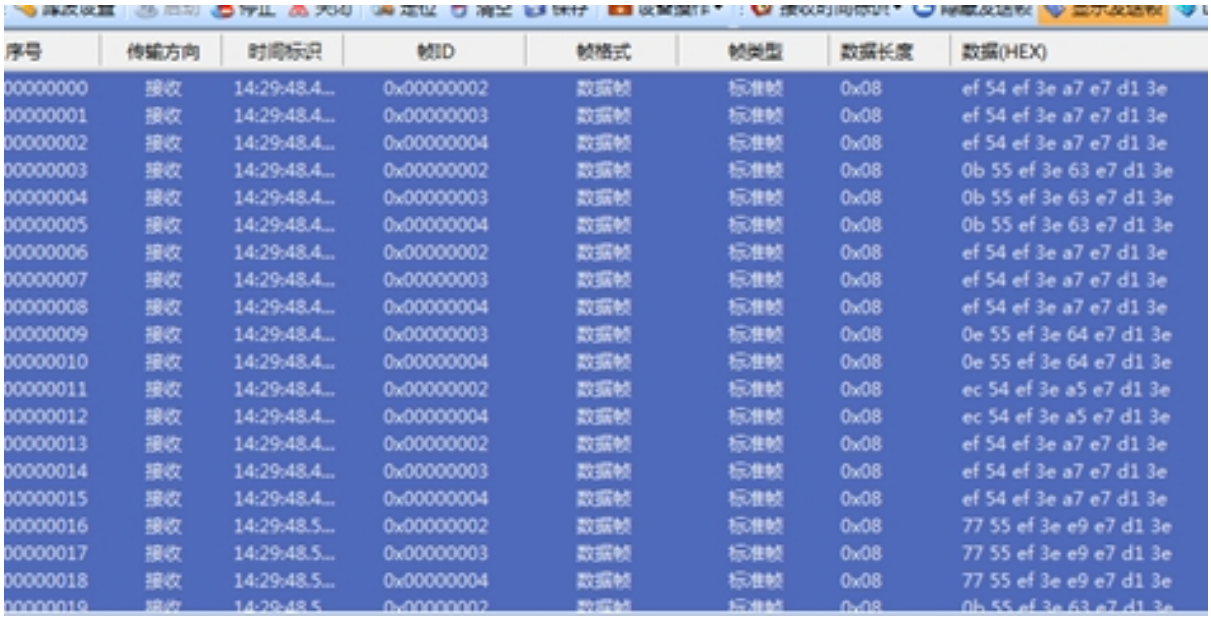
系统配方使用

1、使用 can 盒将电脑和机器人连接，使用 CanTest 调试 can

2、为 can 端口选择配方文件



由于选择的是实时系统配方，CanTest 将会周期性的收到机器人位姿数据：



序号	传输方向	时间标识	帧ID	帧格式	帧类型	数据长度	数据(HEX)
00000000	接收	14:29:48.4...	0x00000002	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000001	接收	14:29:48.4...	0x00000003	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000002	接收	14:29:48.4...	0x00000004	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000003	接收	14:29:48.4...	0x00000002	数据帧	标准帧	0x08	0b 55 ef 3e 63 e7 d1 3e
00000004	接收	14:29:48.4...	0x00000003	数据帧	标准帧	0x08	0b 55 ef 3e 63 e7 d1 3e
00000005	接收	14:29:48.4...	0x00000004	数据帧	标准帧	0x08	0b 55 ef 3e 63 e7 d1 3e
00000006	接收	14:29:48.4...	0x00000002	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000007	接收	14:29:48.4...	0x00000003	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000008	接收	14:29:48.4...	0x00000004	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000009	接收	14:29:48.4...	0x00000003	数据帧	标准帧	0x08	0e 55 ef 3e 64 e7 d1 3e
00000010	接收	14:29:48.4...	0x00000004	数据帧	标准帧	0x08	0e 55 ef 3e 64 e7 d1 3e
00000011	接收	14:29:48.4...	0x00000002	数据帧	标准帧	0x08	ec 54 ef 3e a5 e7 d1 3e
00000012	接收	14:29:48.4...	0x00000004	数据帧	标准帧	0x08	ec 54 ef 3e a5 e7 d1 3e
00000013	接收	14:29:48.4...	0x00000002	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000014	接收	14:29:48.4...	0x00000003	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000015	接收	14:29:48.4...	0x00000004	数据帧	标准帧	0x08	ef 54 ef 3e a7 e7 d1 3e
00000016	接收	14:29:48.5...	0x00000002	数据帧	标准帧	0x08	77 55 ef 3e e9 e7 d1 3e
00000017	接收	14:29:48.5...	0x00000003	数据帧	标准帧	0x08	77 55 ef 3e e9 e7 d1 3e
00000018	接收	14:29:48.5...	0x00000004	数据帧	标准帧	0x08	77 55 ef 3e e9 e7 d1 3e

当通过 CanTest 发送数据帧 id 1，数据为 01 时，通过配方管理器将会控制对应的 IO 端口：

3.3.3 TCP 及 UDP 端口配方的使用

tcp/ip 接口同样可以使用流式配方接口，由于历史原因，tcp 和 udp 仅仅能够使用配方中的系统数据配方（实时配方）。

机器人控制器提供了两种 tcp/ip 接口使用配方的方式：

1、udp server，端口号 2011。

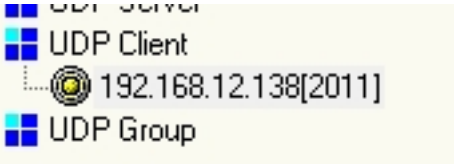
2、tcp/udp client，ip 和端口由用户自定义。

udp server

1、使用网线将电脑和控制器连接，使用网络调试助手进行测试

2、为 udp server 配置配方文件

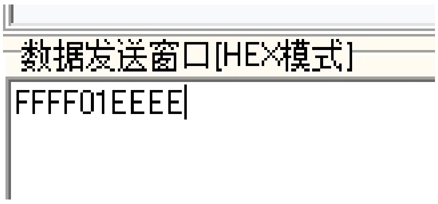
3、使用调试助手创建 udp client



4、连接成功后将会收到周期性的机器人位姿数据

```
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据: {FF FF 39 2F 0B BF 86 87 13 3F BF BD 9D 3F 37 FC C8 BF 17 A0 1B BF 59 FC EA BE FE FE }
17:03:53收到数据
```

5、向服务器发送 FF FF 01 EE EE, 将会控制配置的对应的 IO。

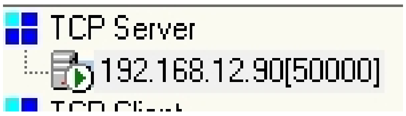


```
FFFF01EEEE
```

tcp/udp client

1、使用网线将电脑和控制器连接，使用网络调试助手进行测试

2、使用调试助手创建 tcp/udp 服务器



3、通过 ui 创建 tcp/udp 客户端

名称: tcp1

类型: TCP

IP地址: 192.168.12.90

端口: 50000

4、为客户端选择配方文件

5、调试助手端将会收到周期性的机器人位姿数据

6、向客户端发送 FF FF 01 EE EE, 将会控制配置的对应的 IO

CHAPTER

FOUR

PROFINET DEVICE 接口操作手册

4.1 Profinet device 接口配置

4.1.1 接口说明

关于硬件接口说明，请参照 DUCO CORE 硬件手册。

Profinet 数据交互的周期为 8ms, Watchdog 设置为 5ms

4.1.2 脚本函数

机器人提供了相应的脚本函数，可读取或者写入数据接口的值。

---

write_reg(number:num,number:type,val)

函数说明:

该函数可修改内部寄存器的值。

参数说明:

num: 内部寄存器序号。

type: 修改的寄存器类型 1 为 bool 寄存器，2 为 word 寄存器，3 为 float 寄存器。

val: 根据 type 类型确定 val 类型。

当 type 为 1 时，val 类型为 boolean,true 表示真，false 表示假,num 范围为 1-64

当 type 为 2 时，val 类型为 number, 范围 0-65535，num 范围为 1-32

当 type 为 3 时，val 类型为 number，num 范围为 1-32

参数错误时函数不改变内部寄存器数值。

返回值:

无

示例:

write_reg(5, 1,true)

---

read_reg (number:num,number:type,number:in_out)

函数说明:

该函数可读取内部寄存器的值。

参数说明:

num: 内部 bool 寄存器序号。

type: 寄存器类型 1 为 bool 寄存器，2 为 word 寄存器，3 为 float 寄存器。

当 type 为 1 时，num 范围为 1-64。

当 type 为 2 时，num 范围为 1-32。

当 type 为 3 时，num 范围为 1-32。

参数错误时函数不改变内部寄存器数值。

in_out: 为 0 时代表读取输入寄存器，1 代表读取输出寄存器。

返回值:

当 type 为 1 时，返回值类型为 boolean,true 表示真，false 表示假。

当 type 为 2 时，返回值类型为 number，范围为 0-65535。

当 type 为 3 时，返回值类型为 number。

示例:

ret=read_reg(10,1,1)

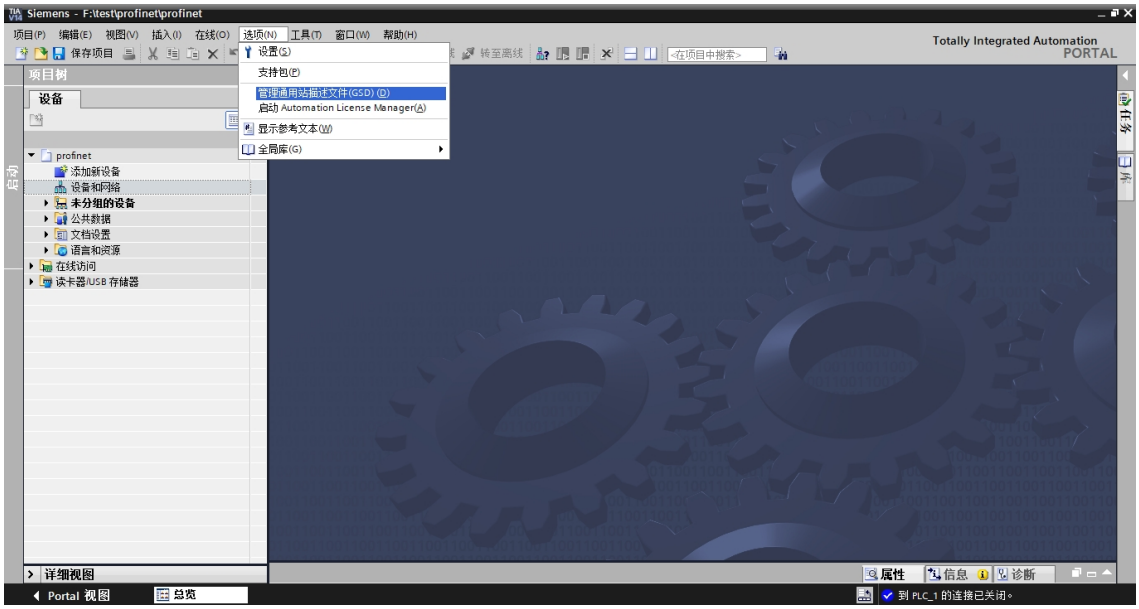
4.1.3 配置流程

下面以西门子 PLC 为例，使用 TIA 14 软件简述 Profinet 接口的配置流程:

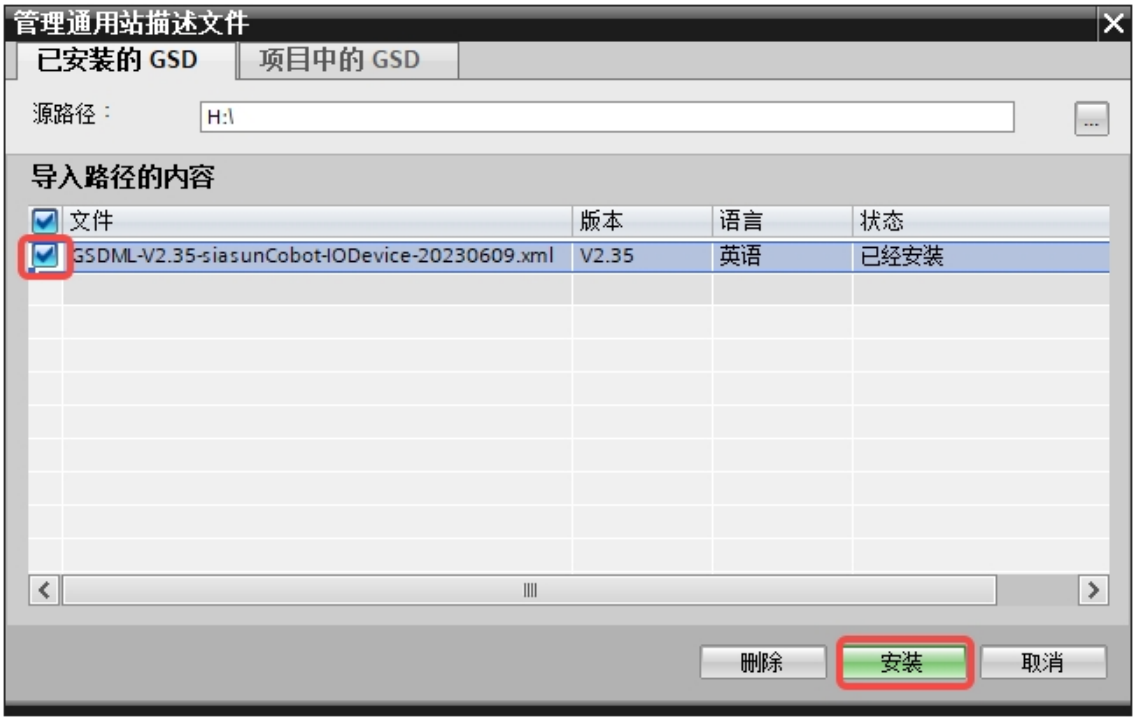
- 如果需新建项目，单击“创建新项目”，如果需打开已有项目，单击“打开现有项目”:

- 单击“打开项目视图”：

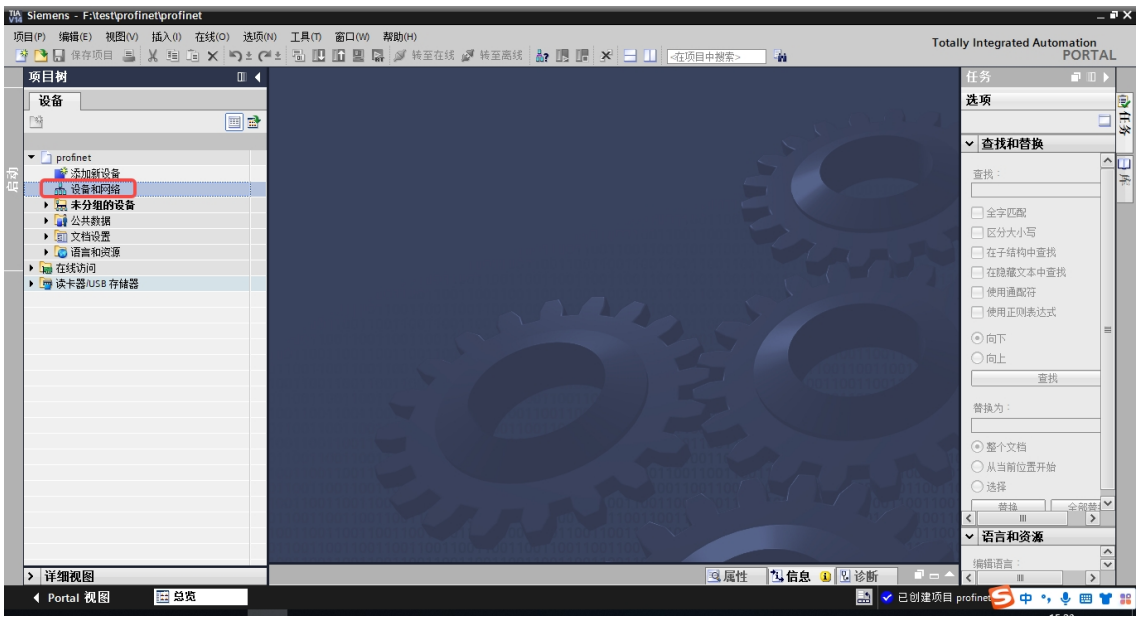
- 单击菜单栏“选项”里的“安装设备描述文件（GSD）”：



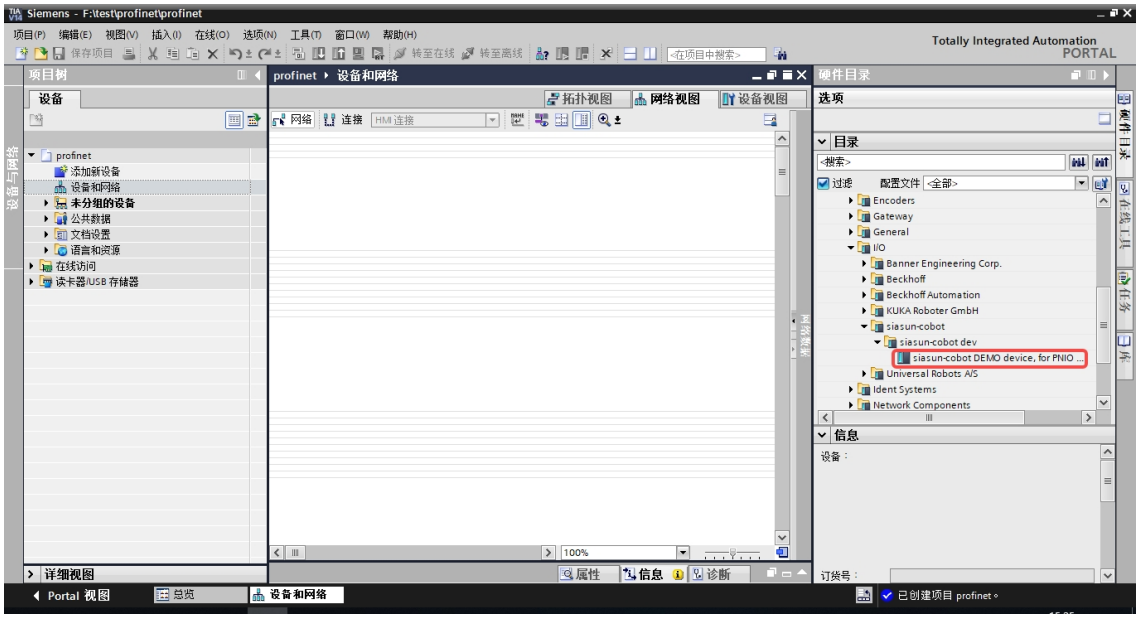
- 按照下图标号顺序，单击 1 处按钮，选择 GSD 文件的路径，选择 2 处复选框选择要导入 GSDML-V2.35-siasunCobot-IODevice-20200615.xml 文件，最后单击 3 处的“安装”



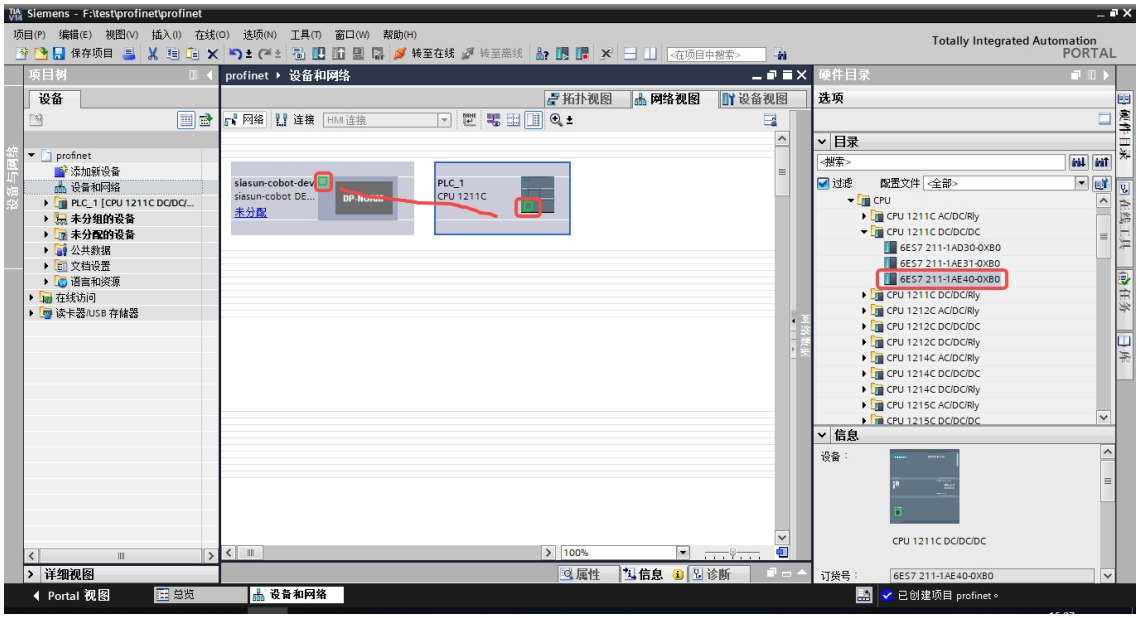
- 双击“设备和网络”，在最左边一栏选择“硬件目录”：



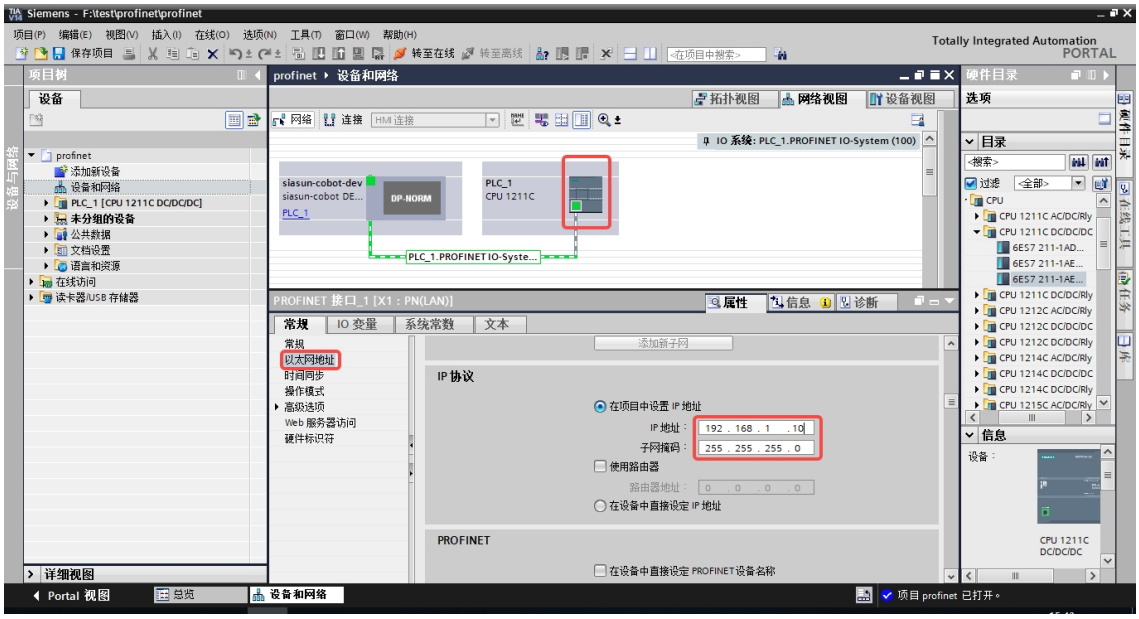
- 双击“其他现场设备->PROFINET IO->I/O->siasun-cobot dev”下的“siasun-cobot Demo dev”，注意选择版本为刚刚载入的 GSDML-V2.35-siasunCobot-IODevice-20200615.xml



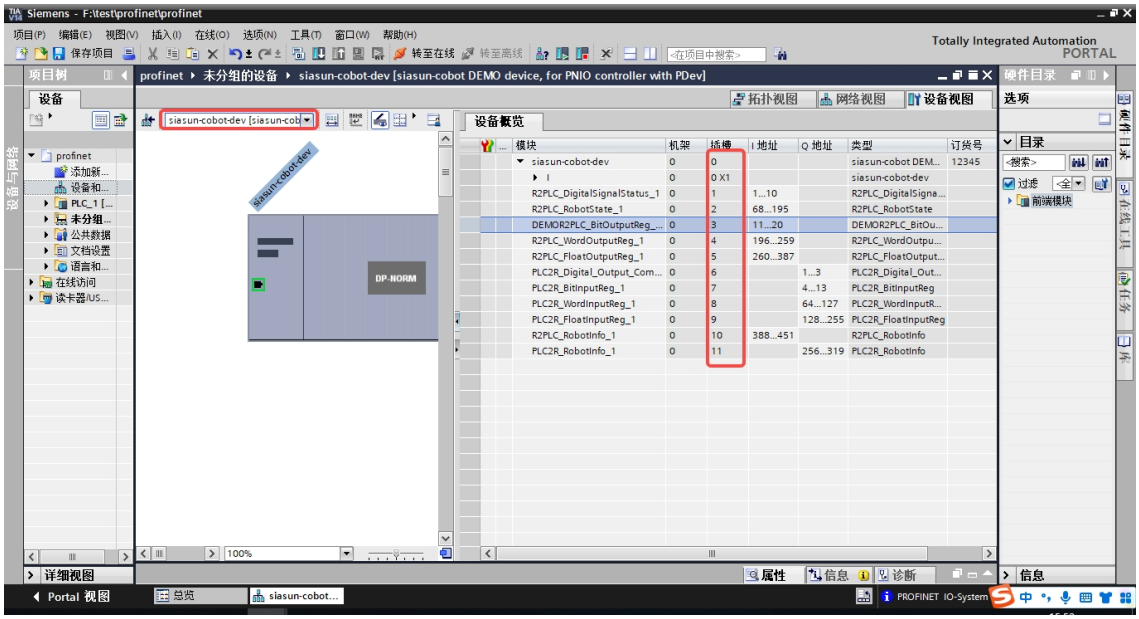
- 作为 Profinet Device 设备，组态还需要 controller，这里用上图同样的方式加入 CPU1211C DC/DC/DC 作为控制器。左击点中一个设备的绿方框不松开然后向另一个设备的绿方框拖动，实现设备连接



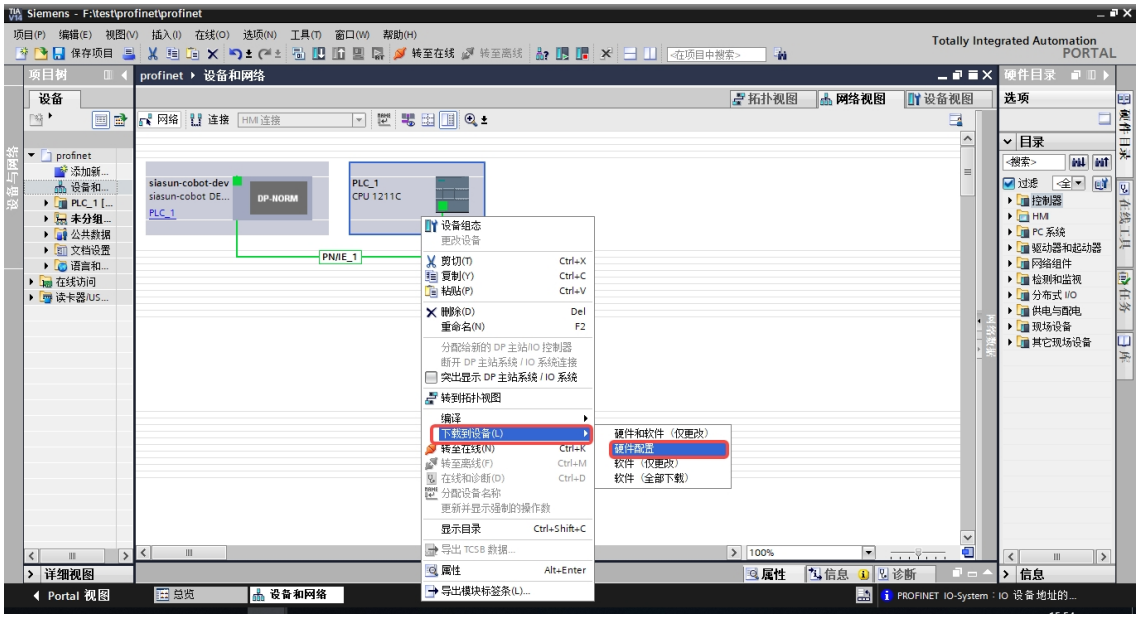
- 选择一个设备，在“属性->PROFINET 接口-> 以太网地址”中选择“在项目中设置 IP 地址”，填写“IP 地址”和“子网掩码”，另外一个设备也用同样的方式去配置。



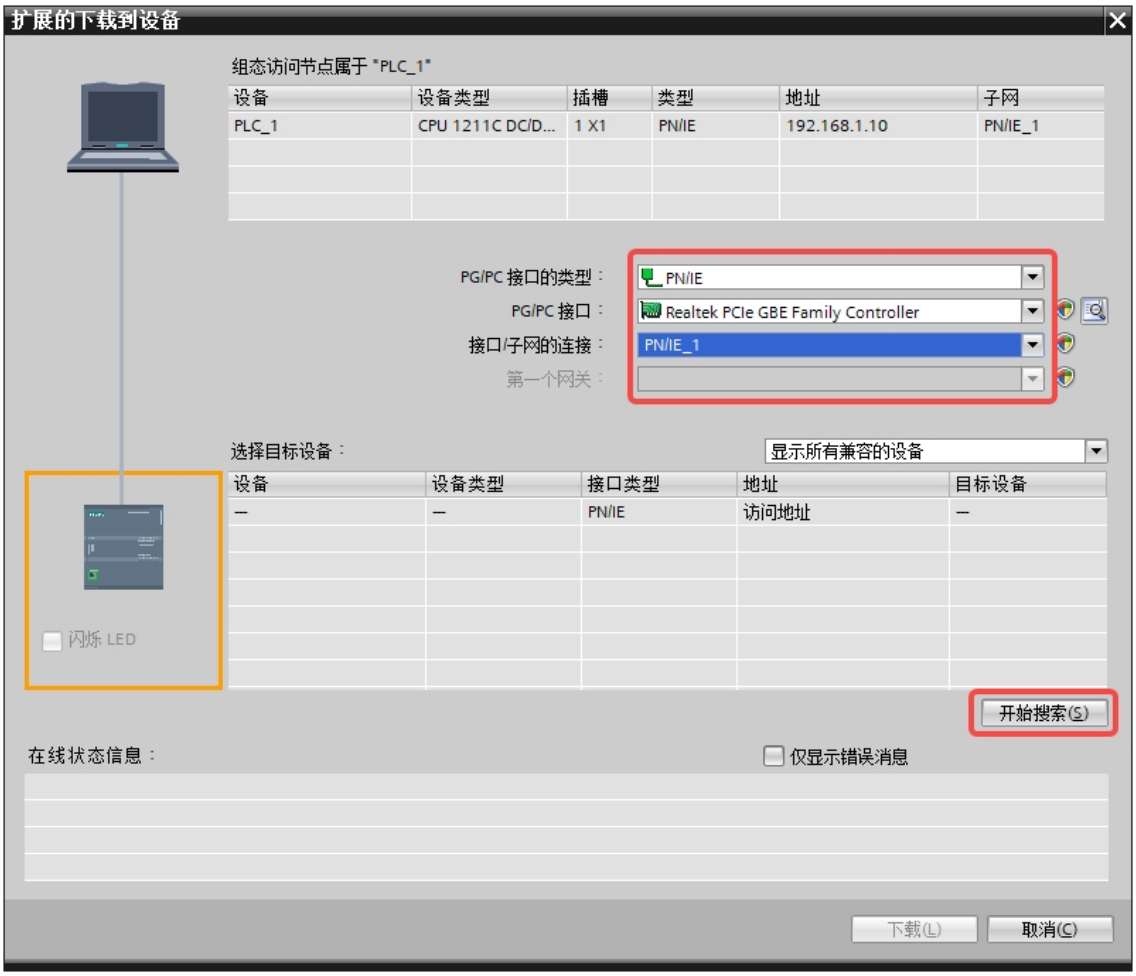
- 在“siasun-cobot dev”设备的“属性->PROFINET 接口-> 高级选项-> 实时设定->IO 周期”中选择“手动设定”，刷新时间为 8ms
- 单击“设备视图”，
- 选择“siasun-cobot dev”设备，在不同编号的插槽内双击“模块“的不同变量类型添加模块：



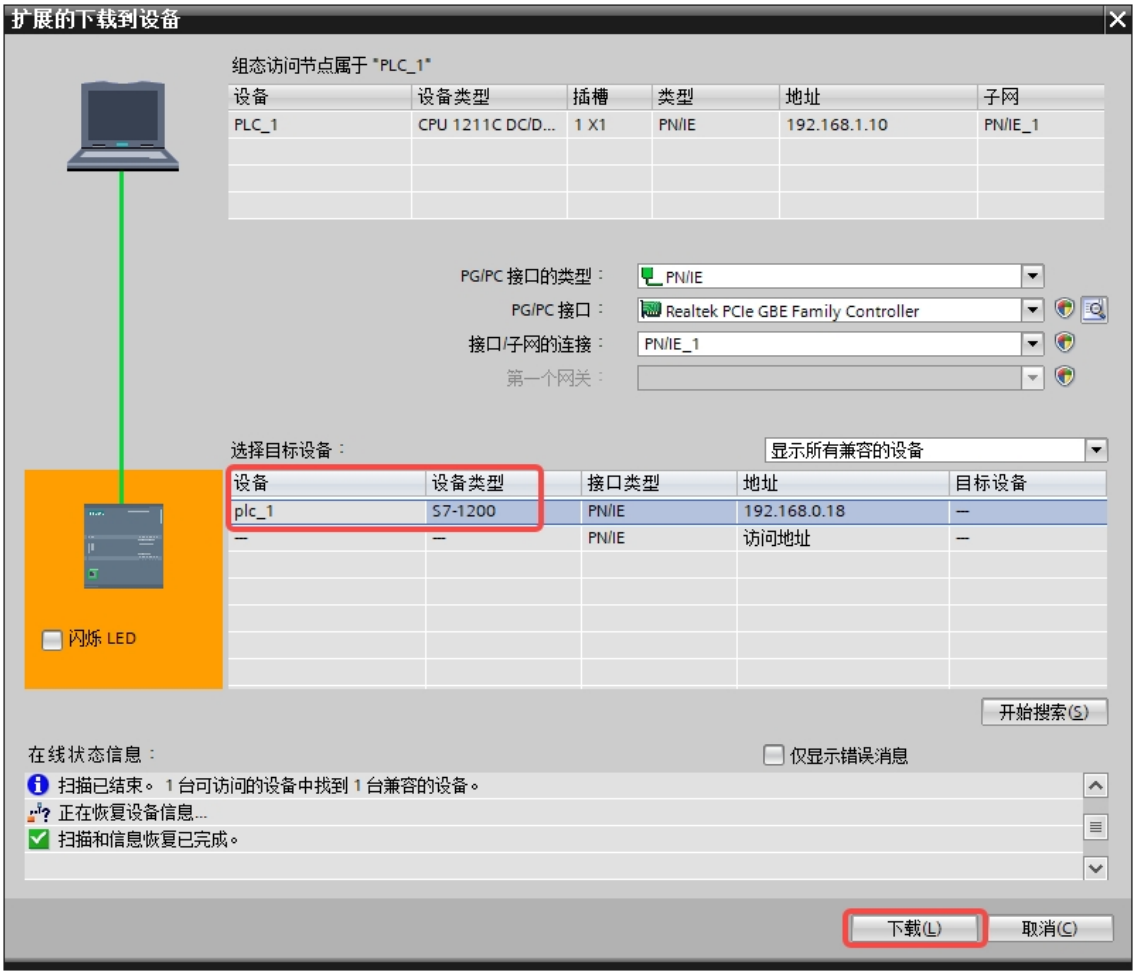
- 各插槽的数据类型为下图所示，具体含义详见下文：
- 添加完毕后，右击 controller，这里为 CPU1211C，选择“下载到设备”的“硬件配置”



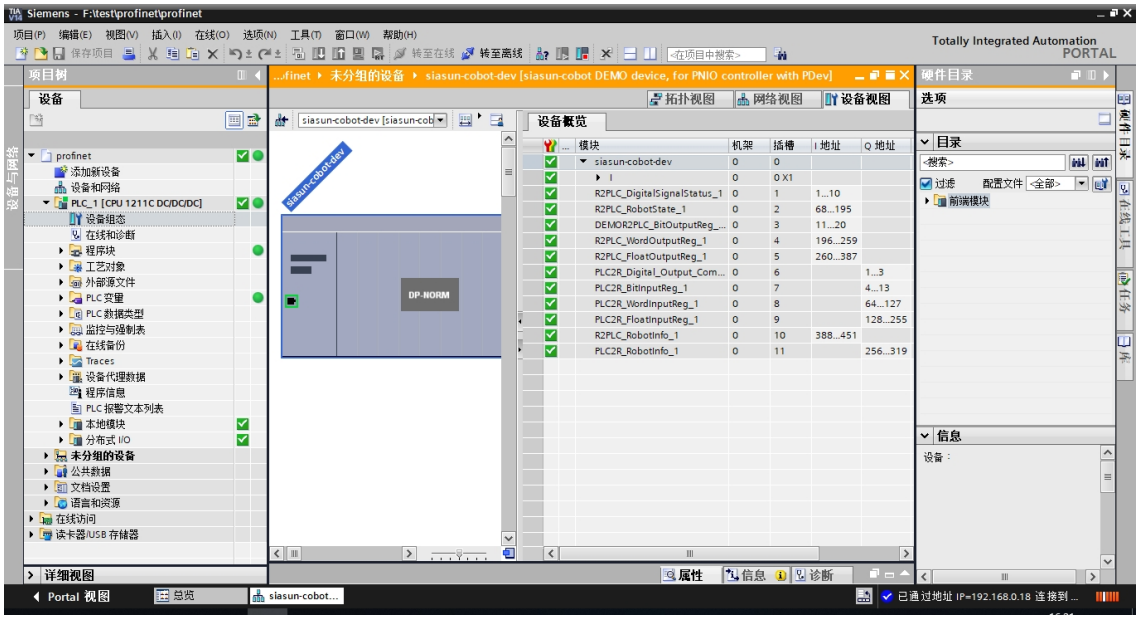
- 选择“PG/PC 接口的类型”，“PG/PC 接口”以及“接口/子网的连接”，之后单击“开始搜索”（注意：运行 TIA 的配置电脑需与 controller 设备可连接，例如处于同一网段）



- 选中搜索到的 controller 设备，这里为 CPU 1121C，单击“下载”：



- 一切正常，单击“下载”，等待下载完成：
- 单击“完成”，等待启动：
- 单击“转到在线”，在“诊断”选项卡中如果没有错误，代表通信正常。



- 通信正常情况下，所有模块为“绿色底白色对勾”

4.2 Slot 数据结构

注：各插槽（slot）的数据，WORD、FLOAT 类型的数据，已经进行了字节转换处理。即转换成低位高字节、高位低字节。

（1）Slot1_R2PLC_DigitalSignalStatus（10 byte）：机器人数字信号状态 数据方向：机器人——» PLC

	bit0	bit1	bit2	bit3	bit4	bit5	bit6	bit7
BYTE1	general_digital_input[0..7]							
BYTE2	general_digital_input[8..15]							
BYTE3	tool_digitial_input[0..7]							
BYTE4	general_digital_output[0..7]							
BYTE5	general_digital_output[8..15]							
BYTE6	tool_digital_output[0..7]							
BYTE7	fun_digital_input[0..7]							
BYTE8	fun_digital_output[0..7]							
BYTE9	safety_state_input							
BYTE10	safety_state_output							

端子定义详见 DUCO CORE 硬件手册

general_digital_input: 控制柜普通 DI 状态。

tool_digitial_input：机械臂末端 DI 状态

general_digital_output：控制柜普通 DO 状态

tool_digital_output：机械臂末端 DO 状态

fun_digital_input：控制柜功能 DI 状态

fun_digital_output：控制柜功能 DO 状态

安全控制器状态 safety_state 结构说明

	safety_state_1	safety_state_2
bit0	ystem_emergency_stop	config_safety_output0
bit1	external_emergeny_stop	config_safety_output1
bit2	protective_stop_input	reserved
bit3	operation_mode_input	reserved
bit4	3_position_enable_input	reserved
bit5	config_safety_input0	reserved
bit6	config_safety_input1	reserved
bit7	reserved	reserved

config_safety_input：由界面设置的安全 input

config_safety_output：由界面设置的安全 output

（2）Slot2_R2PLC_RobotState(29 float): 机器人状态信息输出 数据方向：机器人——» PLC

Float1..7	joint_pos（rad）
Float8..13	tcp_pose(tcp 相对于基坐标系的值)
Float14..19	tcp_force
Float20..25	tcp_offset
Float26..29	tcp_load (质心、质量)

（3）Slot3_R2PLC_BitOutputReg（10 byte）: 位输出寄存器，输出机器人的当前位寄存器状态信息 数据方向：机器人——》PLC

	bit0	bit1	bit2	bit3	bit4	bit5	bit6	bit7
BYTE1	fun_registers_output[1..8]							
BYTE2	fun_registers_output[9..16]							
BYTE3	bool_registers_output[1..8]							
BYTE4	bool_registers_output[9..16]							
BYTE5	bool_registers_output[17..24]							
BYTE6	bool_registers_output[25..32]							
BYTE7	bool_registers_output[33..40]							
BYTE8	bool_registers_output[41..48]							
BYTE9	bool_registers_output[49..56]							
BYTE10	bool_registers_output[57..64]							

（4）Slot4_R2PLC_WordOutputReg（64 byte）:Word 输出寄存器 数据方向：机器人——》PLC

Word1..32	word_output_register [1..32]

（5）Slot5_R2PLC_FloatOutputReg（32 float）: 浮点输出寄存器 数据方向：机器人——》PLC

Float1..32	float_output_register [1..32]

（6）Slot6_PLC2R_Digital_Output_Command (3 byte) : 机器人相关控制指令输入 数据方向：PLC——》机器人

	bit0	bit1	bit2	bit3	bit4	bit5	bit6	bit7
BYTE1	general_digital_ouput[1..8]							
BYTE2	general_digital_ouput[9..16]							
BYTE3	tool_digital_output[1..8]							

（7）Slot7_PLC2R_BitInputReg(10 byte): 通用位输入寄存器 数据方向：PLC——》机器人

	bit0	bit1	bit2	bit3	bit4	bit5	bit6	bit7
BYTE1	fun_input_register[1..8]							
BYTE2	fun_input_register[9..16]							
BYTE3..10	bit_input_register[1..64]							

（8）Slot8_PLC2R_WordInputReg (64 byte): Word 输入寄存器 数据方向：PLC——》机器人

Word1..32	word_input_register[1..32]

（9）Slot9_PLC2R_FloatInputReg(32 float): 通用浮点输入寄存器 数据方向: PLC ——》机器人

Float1..32	float_input_register[1..32]

（10）Slot10_R2PLC_RobotInfo(16 float): 机器人速度等信息 数据方向: 机器人——》PLC

Float1	读取全局速度百分比
Float2	读取 Jog 速度百分比
Float3	读取末端合线速度
Float4	读取仿真/真机模式
Float5	读取错误 ID（系统最后一次错误 ID）
Float6..16	预留

（11）Slot11_PLC2R_RobotInfo(16 float): 机器人速度等信息 数据方向: PLC ——》机器人

Float1	读取全局速度百分比
Float2..16	预留

CHAPTER

FIVE

ETHERNET IP 接口操作手册

5.1 Ethernet 及 IP 接口配置

5.1.1 接口说明

关于硬件接口说明，请参照 DUCO CORE 硬件手册。

Ethernet/IP 数据交互的周期为 4ms。

5.1.2 脚本函数

机器人提供了相应的脚本函数，可读取或者写入数据接口的值。

---

write_reg(number:num,number:type,val)

函数说明:

该函数可修改内部寄存器的值。

参数说明:

num: 内部寄存器序号。

type: 修改的寄存器类型 1 为 bool 寄存器，2 为 word 寄存器，3 为 float 寄存器。

val: 根据 type 类型确定 val 类型。

当 type 为 1 时，val 类型为 boolean,true 表示真，false 表示假,num 范围为 1-64

当 type 为 2 时，val 类型为 number, 范围 0-65535，num 范围为 1-32

当 type 为 3 时，val 类型为 number，num 范围为 1-32

参数错误时函数不改变内部寄存器数值。

返回值:

无

示例:

write_reg(5, 1,true)

---

read_reg (number:num,number:type,number:in_out)

函数说明:

该函数可读取内部寄存器的值。

参数说明:

num: 内部 bool 寄存器序号。

type: 寄存器类型 1 为 bool 寄存器，2 为 word 寄存器，3 为 float 寄存器。

当 type 为 1 时，num 范围为 1-64。

当 type 为 2 时，num 范围为 1-32。

当 type 为 3 时，num 范围为 1-32。

参数错误时函数不改变内部寄存器数值。

in_out: 为 0 时代表读取输入寄存器，1 代表读取输出寄存器。

返回值:

当 type 为 1 时，返回值类型为 boolean,true 表示真，false 表示假。

当 type 为 2 时，返回值类型为 number，范围为 0-65535。

当 type 为 3 时，返回值类型为 number。

示例:

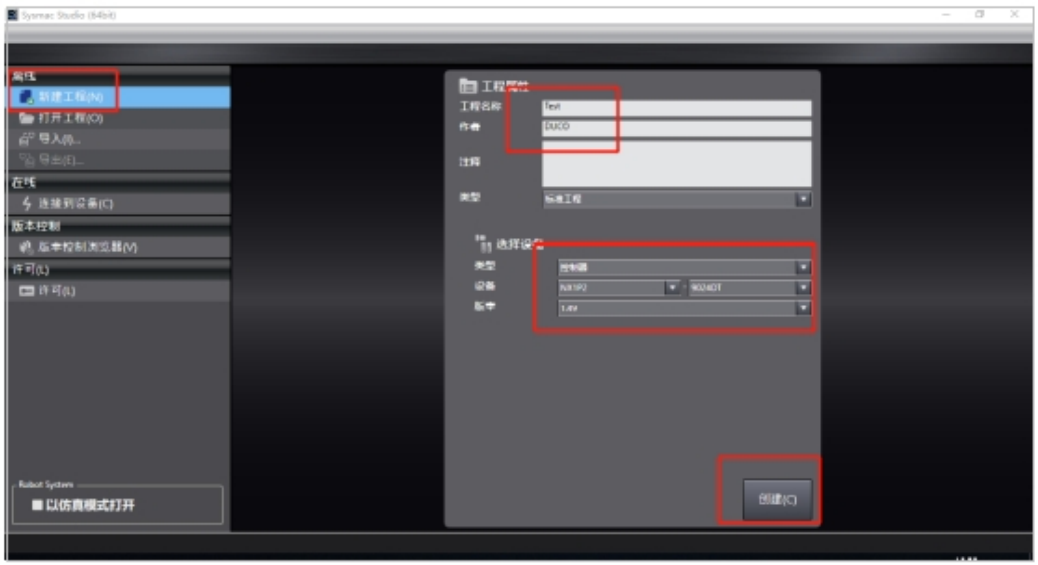
ret=read_reg(10,1,1)

5.1.3 配置流程

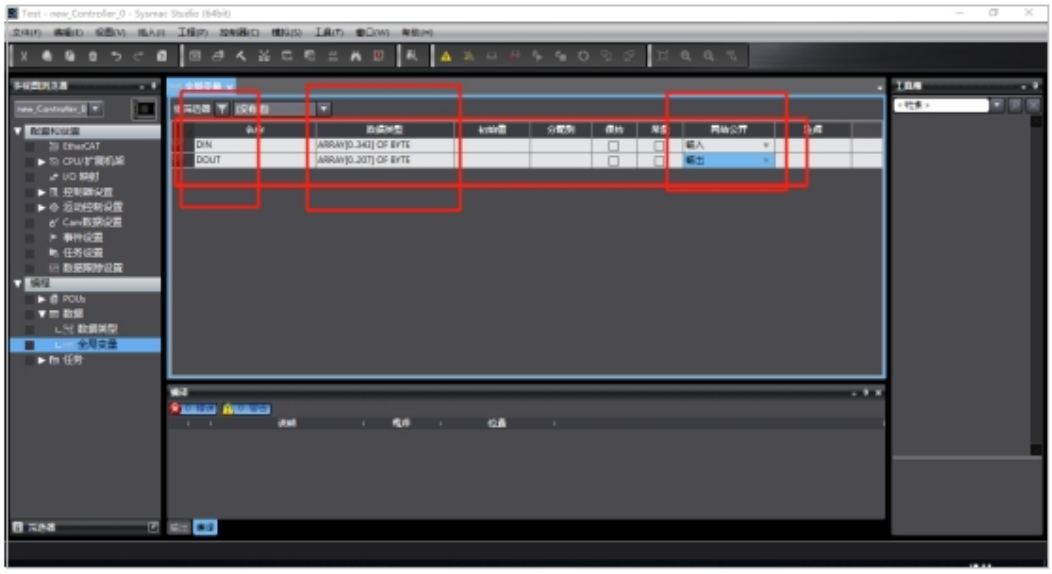
下面以欧姆龙 PLC 为例，使用 Sysmac Studio 软件简述 Ethernet/IP 接口的配置流程:

- 单击“新建工程”:

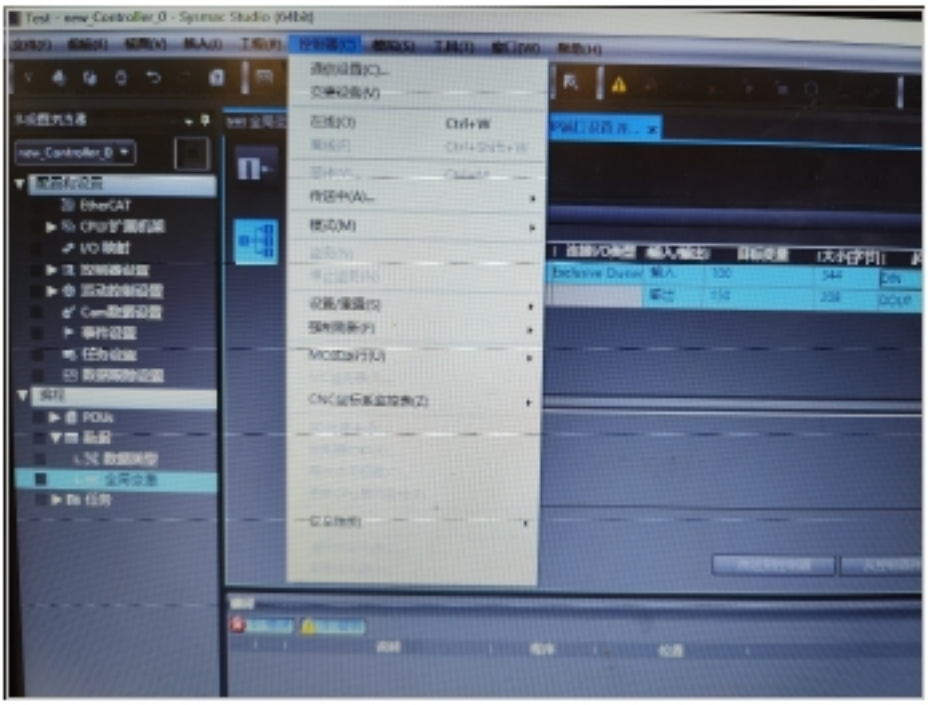
- 输入工程名称和作者，选择当前设备的型号，然后点击“创建”:

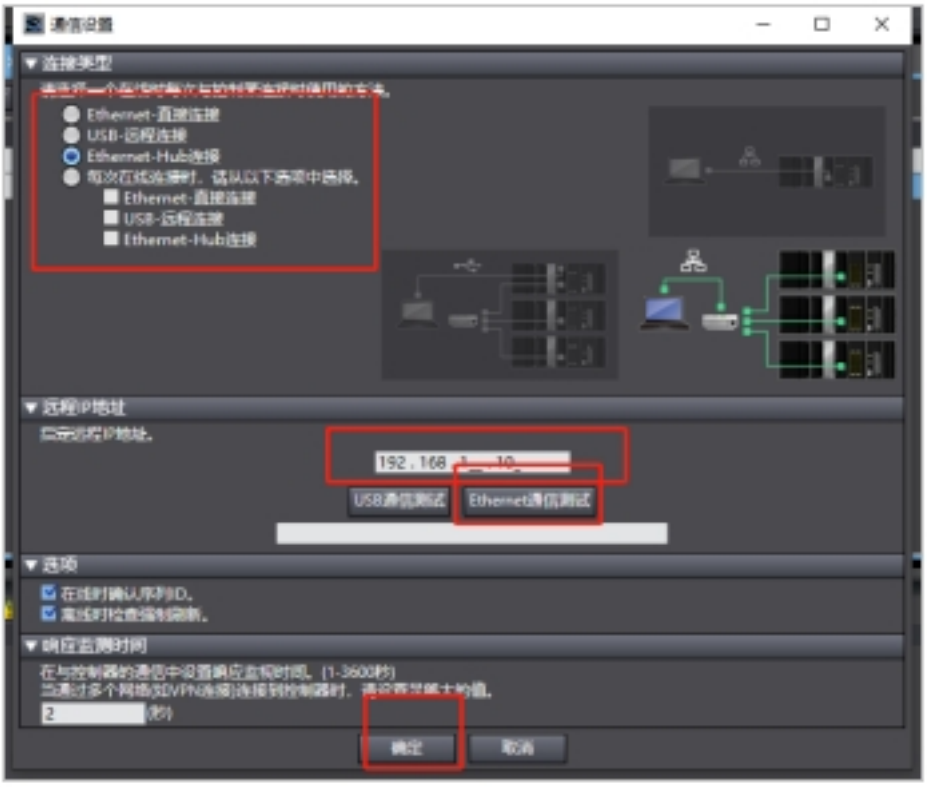


- 点击“编程”下拉框，点击“全局变量”；在“全局变量”页面里，鼠标右键，点击“新建”，创建用于接收和发送的变量（robot→PLC 的字节大小是 344，PLC→robot 的字节大小是 208），并选择变量网络公开类型：

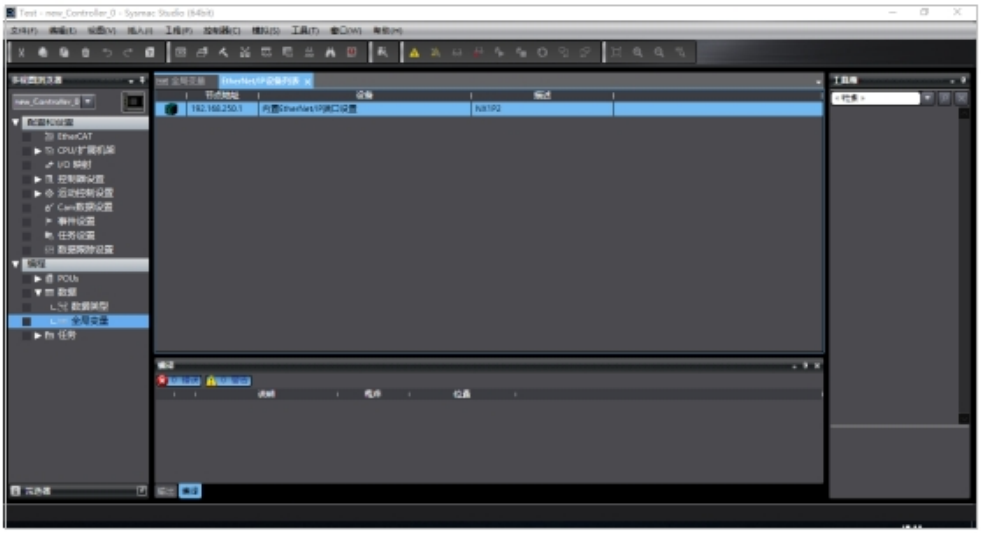


- 点击“控制器”、“通信设置”，在弹出的界面里选择连接类型，输入 PLC 的 IP，并点击“Ethernet 通信测试”，检查电脑和 PLC 的通信是否正常：

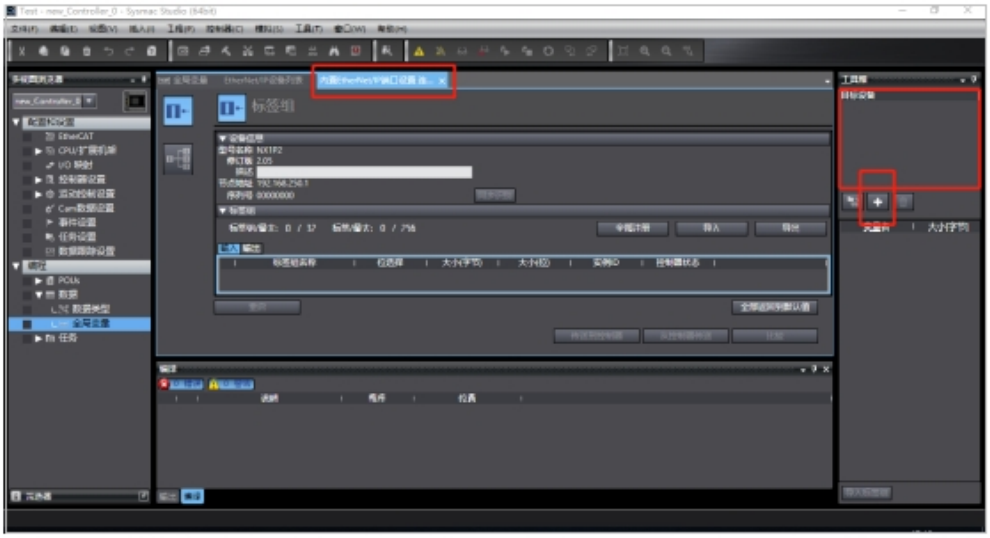




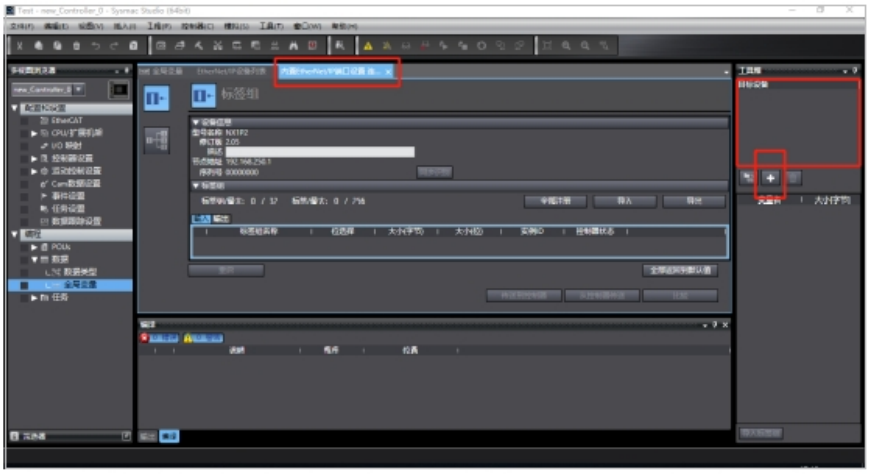
- 点击“工具”、“EtherNet/IP 连接设置 (N)”，加载“EtherNet/IP 设备列表”窗口:



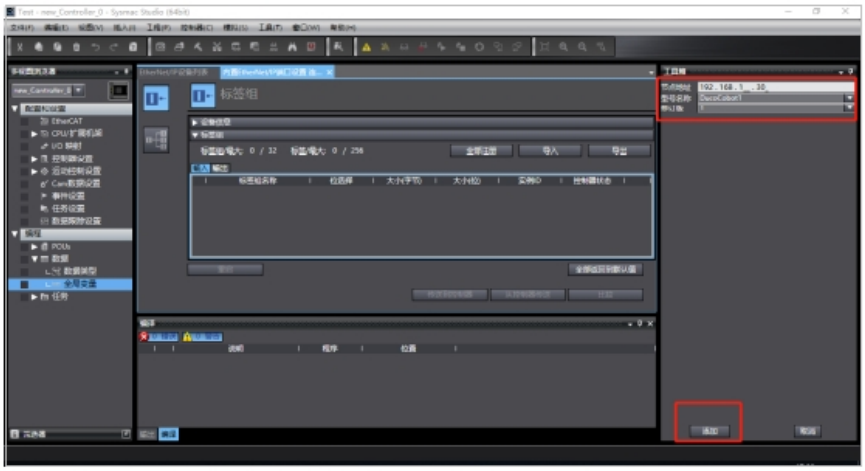
- 双击“EtherNet/IP 设备列表”中条目，加载“内置 EtherNet/IP 端口设置连接设置”窗口:



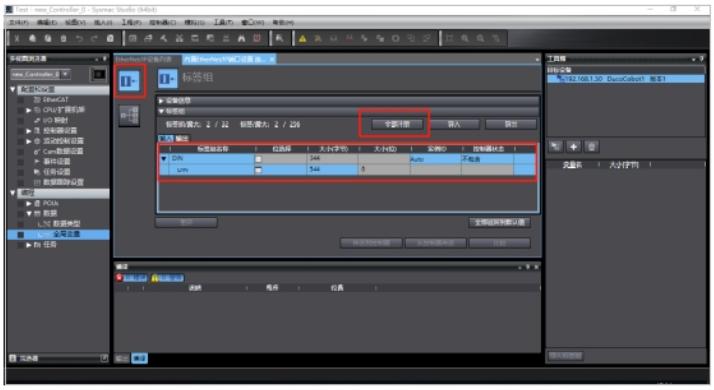
- 在软件右侧的“目标设备”窗口，鼠标右键，点击“显示 EDS”，弹出“EDS 库”列表，点击“安装”，选择 DUCO COBOT 的 EDS 文件：



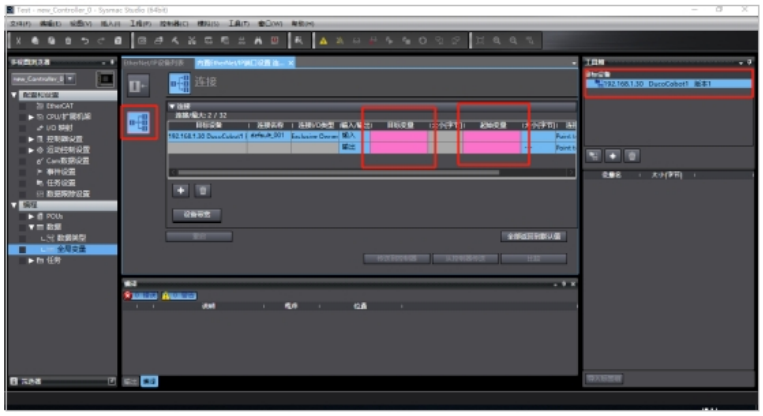
- 单击“目标设备”窗口下方的“+”按键，添加机器人的 IP，选择 DUCO COBOT 的 EDS，并选择修订版本，然后点击添加：



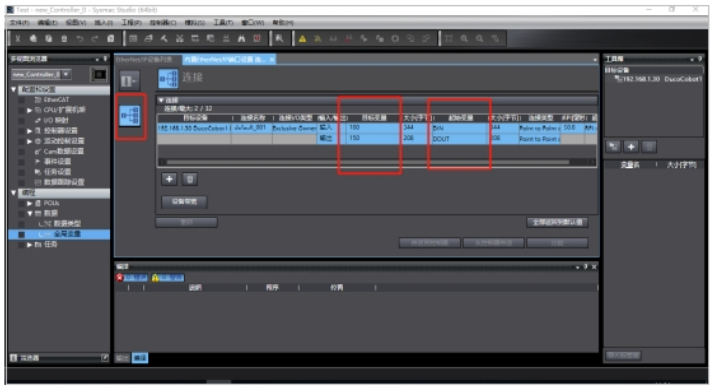
- 点击“标签组”窗口里的“全部注册”按键，注册创建的全局变量：



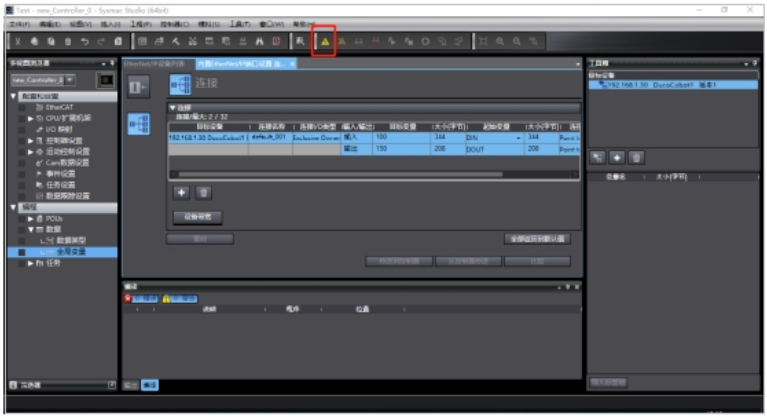
- 点击“连接”图标，然后双击创建的目标设备（或鼠标左键长按目标设备，并拖拽到“连接”窗口），将机器人的输入输出变量添加到“连接”窗口：



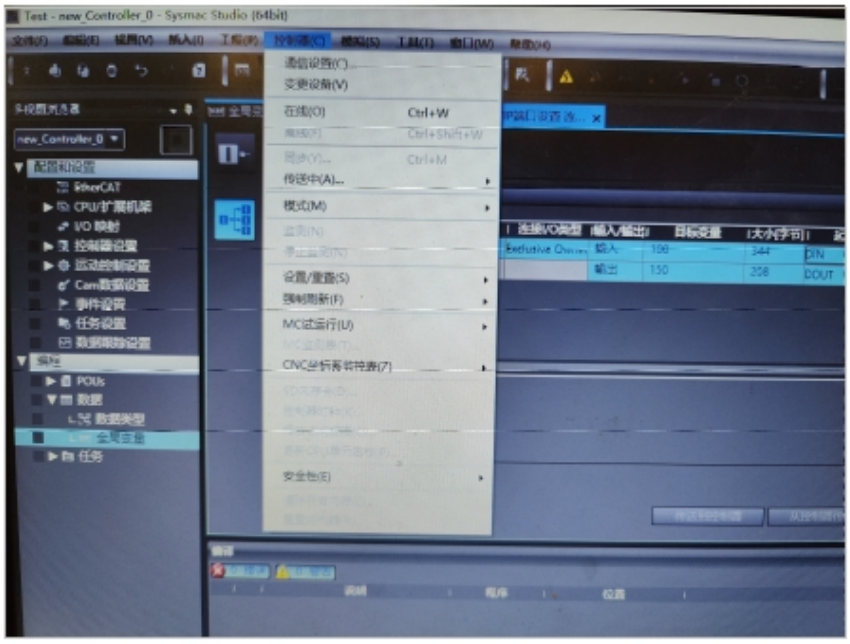
- 输入目标变量填写“100”，输入起始变量选择全局变量的输入“DIN”；输出目标变量填写“150”，输出起始变量选择全局变量的输出“DOUT”：



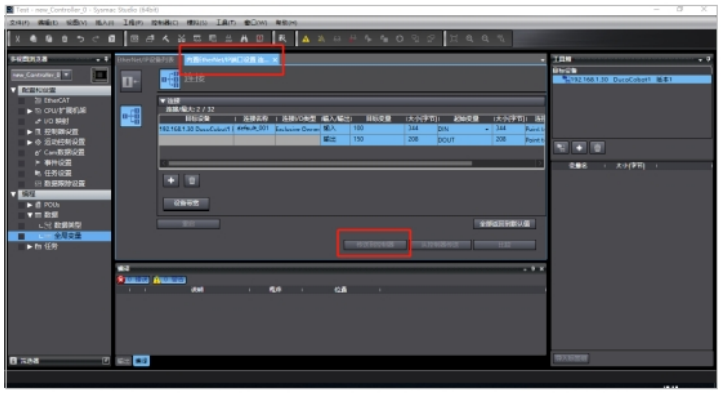
- 点击“在线”图标，连接 PLC：



- 连接成功后，点击“控制器”、“同步”，在弹出的对话框里，勾选所有选项，并点击“传送到设备”，将 EDS 下载到 PLC：

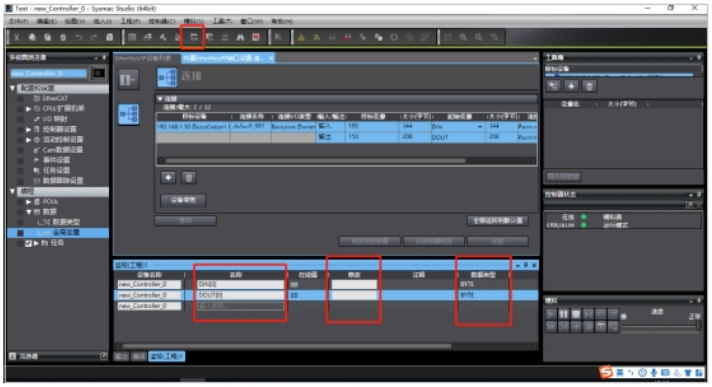


- 点击“内置 EtherNet/IP 端口设置连接设置”窗口里的“传送到控制器”，在弹出的对话框里选择“编辑模式传送”：



- 点击“控制器”、“模式”，选择“运行模式”。然后，点击“监控”图标，加载“监视”窗口，添加输入、输出变量，监控（或设置）PLC 接收和输出的数据（接收的

数据不能修改）：



5.2 数据结构

（1）R2PLC_DigitalSignalStatus（10 byte）：机器人数字信号状态 数据方向：机器人 →PLC

	数据	
BYTE1	general_digital_input[0..7]	DIN[0]
BYTE2	general_digital_input[8..15]	DIN[1]
BYTE3	tool_digitial_input[0..7]	DIN[2]
BYTE4	general_digital_output[0..7]	DIN[3]
BYTE5	general_digital_output[8..15]	DIN[4]
BYTE6	tool_digital_output[0..7]	DIN[5]
BYTE7	fun_digital_input[0..7]	DIN[6]
BYTE8	fun_digital_output[0..7]	DIN[7]
BYTE9	safety_state_input	DIN[8]
BYTE10	safety_state_output	DIN[9]

注：以上文示例中全局变量 DIN 为例。

端子定义详见 DUCO CORE 硬件手册

general_digital_input: 控制柜普通 DI 状态。

tool_digitial_input：机械臂末端 DI 状态

general_digital_output：控制柜普通 DO 状态

tool_digital_output：机械臂末端 DO 状态

fun_digital_input：控制柜功能 DI 状态

fun_digital_output：控制柜功能 DO 状态

安全控制器状态 safety_state 结构说明

	safety_state_1	safety_state_2
bit0	ystem_emergency_stop	config_safety_output0
bit1	external_emergeny_stop	config_safety_output1
bit2	protective_stop_input	reserved
bit3	operation_mode_input	reserved
bit4	3_position_enable_input	reserved
bit5	config_safety_input0	reserved
bit6	config_safety_input1	reserved
bit7	reserved	reserved

config_safety_input：由界面设置的安全 input

config_safety_output：由界面设置的安全 output

（2）R2PLC_Reserved（6 byte）：机器人数字信号状态 数据方向：机器人 →PLC

BYTE1..6	reserved [0..5]	DIN[10···15]

（3）R2PLC_RobotState(32float): 机器人状态信息输出 数据方向：机器人 →PLC

Float1..7	joint_pos（rad）	DIN[16···43]
Float8..13	tcp_pose(tcp 相对于基坐标系的值)	DIN[44···67]
Float14..19	tcp_force	DIN[68···91]
Float20..25	tcp_offset	DIN[92···115]
Float26..29	tcp_load (质心、质量)	DIN[116···131]
Float30..32	reserved	DIN[132···143]

注：以上文示例中全局变量 DIN 为例。

（4）R2PLC_BitOutputReg（10 byte）: 位输出寄存器，输出机器人的当前位寄存器状态信息 数据方向：机器人 →PLC

BYTE1	fun_registers_output[1..8]	DIN[144]
BYTE2	fun_registers_output[9..16]	DIN[145]
BYTE3	bool_registers_output[1..8]	DIN[146]
BYTE4	bool_registers_output[9..16]	DIN[147]
BYTE5	bool_registers_output[17..24]	DIN[148]
BYTE6	bool_registers_output[25..32]	DIN[149]
BYTE7	bool_registers_output[33..40]	DIN[150]
BYTE8	bool_registers_output[41..48]	DIN[151]
BYTE9	bool_registers_output[49..56]	DIN[152]
BYTE10	bool_registers_output[57..64]	DIN[153]

注：以上文示例中全局变量 DIN 为例。

（5）R2PLC_WordOutputReg（64 byte）:Word 输出寄存器 数据方向：机器人 →PLC

Word1..32	word_output_register [1..32]	DIN[154···217]

注：以上文示例中全局变量 DIN 为例。

（6）R2PLC_FloatOutputReg（32 float）：浮点输出寄存器　数据方向：机器人 →PLC

Float1..32	float_output_register [1..32]	DIN[220···347]

注：以上文示例中全局变量 DIN 为例。

（7）R2PLC_RobotInfo（16 float）：末端合线速度等信息　数据方向：机器人 →PLC

Float1	读取全局速度百分比	DIN[348···351]
Float2	读取 Jog 速度百分比	DIN[352···355]
Float3	读取末端合线速度	DIN[356···359]
Float4	读取真机/仿真模式	DIN[360···363]
Float5	读取错误 ID（系统最后一次的错误 ID）	DIN[8]
Float6..16	Reserved	DIN[368···411]

注：以上文示例中全局变量 DIN 为例。

（8）PLC2R_Digital_Output_Command (3 byte) : 机器人相关控制指令输入　数据方向：PLC→ 机器人

BYTE1	general_digital_ouput[1..8]	DOUT[0]
BYTE2	general_digital_ouput[9..16]	DOUT[1]
BYTE3	tool_digital_output[1..8]	DOUT[2]

注：以上文示例中全局变量 DOUT 为例。

（9）PLC2R_Reserved (7 byte) : 预留　数据方向：PLC→ 机器人

BYTE1..7	Reserved	DOUT[3···9]

注：以上文示例中全局变量 DOUT 为例。

（10）BitInputReg(10 byte): 通用位输入寄存器　数据方向：PLC→ 机器人

BYTE1	fun_input_register[1..8]	DOUT[10]
BYTE2	fun_input_register[9..16]	DOUT[11]
BYTE3..10	bit_input_register[1..64]	DOUT[12···19]

注：以上文示例中全局变量 DOUT 为例。

（11）PLC2R_WordInputReg (64 byte): Word 输入寄存器　数据方向：PLC→ 机器人

BYTE1..32	word_input_register[1..32]	DOUT[20···83]

注：以上文示例中全局变量 DOUT 为例。

（12）PLC2R_FloatInputReg(32 float): 通用浮点输入寄存器 数据方向: PLC → 机器人

Float1..32	float_input_register[1..32]	DOUT[84···211]

注：以上文示例中全局变量 DOUT 为例。

（13）PLC2R_RobotInfo (16 float): 全局速度百分比等信息 数据方向: PLC → 机器人

Float1	设置全局速度百分比	DOUT[212···215]
Float2..16	Reserved	DOUT[216···275]

注：以上文示例中全局变量 DOUT 为例。