RS485/RS422-EC 模块使用手册

-- V1.0

帮助 100 万家企业实现智能制造

目录

一、产品概述

1.1、产品简介

RS485/RS422-EC 是一款 EtherCAT 转 MODUBUS RTU 通讯功能的模块，是一款经济稳定、安装简易，适用性强的产品。最多支持 10 个从站模块，每个从站模块可支持 64 位线圈、64 位离散输入、16 个保持寄存器、16 个输入寄存器当用户从站 IO 点数大于单一从站模块可设置 IO 点数时，可进行扩展使用，即支持单一模块最大 640 位线圈，640 位离散输入、160 个保持寄存器、160 个输入寄存器。

1.2、特点功能

- 支持 Modbus RTU 主站功能，采用标准 Modbus RTU 通讯协议，通讯接口可设置
- 支持 RS485 或 RS422 透明传输功能
- 模块波特率、数据格式、从站相关信息可通过 EtherCAT 上位机进行设置
- 相关用户信息可永久保存，上电即可用
- 采用符合 DIN 35MM 标准导轨安装方式，方便安装
- 电源电路采用防反接设计
- 广泛用于工业现场设备的信号采集和控制

1.3、应用场景

RS485/RS422-EC 模块可应用范围很广，如：PLC 控制、工业自动化、楼宇自控、POS 系统、电力监控、门禁医疗、考勤系统、自助银行系统、电信机房监控、信息家电、LED 信息显示设备、测量仪表及环境动力监控系统、售饭系统等需要 RS485 或 RS422 串口转 EtherCAT 总线的设备或系统。

二、产品规格

2.1、产品参数

主要参数	
网口参数（RS422 通讯参数）	
接口类型	RJ45
通讯协议	EtherCAT
串口参数（RS422 通讯参数）	
接口类型	RS422（5.08mm 间距工业级接线端子）
波特率	1200~4.6875Mbps
通信格式	默认 8 位数据，1 位停止，无校验
传输距离	波特率 100kb/s 时，422 串口通讯 1200 米，以实际为准
串口参数（RS485 通讯参数）	
接口类型	RS485（5.08mm 间距工业级接线端子）
波特率	1200~4.6875Mbps
通信格式	默认 8 位数据，1 位停止，无校验
传输距离	波特率 9600 时，485 串口通讯 1200 米，以实际为准
电源参数	
工作电压	DC 24V;带防反接保护
功耗	2W~4W
工作环境	
工作温度	-20°C~+70°C
存储温度	-40°C~+85°C
其他	
安装方式	导轨
尺寸	125MM(长)*80MM(宽)*50MM(高)，以实物为准

三、产品功能

3.1、Modbus RTU 主站功能

本模块 RTU 主站最多可支持 10 个从站模块，每个模块可设置 32 路数字（线圈）输出、32 路离散输入、16 个输入寄存器和 16 个保持寄存器。

使用 Modbus RTU 功能时，每个从站会生成各自读写命名，每条命令以一定周期（可设）进行轮询，当周期过小时，主站会给出警告，但是仍然会在超时后尽快将下一条指令发出。

使用 Modbus RTU 功能时，写功能调用不能小于轮询周期的 2 倍，不然会出现一直进行写功能，回读数据不刷新，详细设置方式见第七章节。

Modbus 通讯可选使用 RS422 或者 RS485 接口。

3.2、透传

当使用透传模式时，本模块在收到 EtherCAT 主站发出的数据后，将直接通过所选串口发出，接收到串口数据后，将数据直接发送到 EtherCAT 主站。

3.3、按键复位功能

模块上电时，按住 Reset 复位按钮，直到 EC_MB 模块 RUN 灯出现双闪后松开按钮，再将模块断电至少 3S 后上电，模块即恢复出厂参数，如下表。

参数名称	参数默认值
串口参数	波特率 9600，校验位 None,停止位 1 位
从站参数	恢复默认，全 0
选用接口	RS485
选用模式	透传模式

3.4、Boot 功能

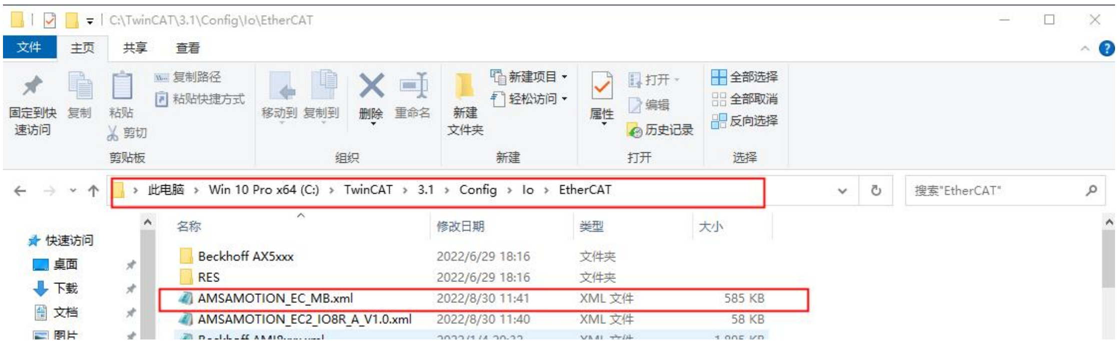
模块上电之前或者上电后 3 秒内，先将 SW1 拨至“1”并保持 3 秒以上，此时 RUN 灯、RS422、RS485 灯均以 1 秒间隔闪烁，3 秒后 RUN 灯由 1 秒间隔闪烁变为 0.2 秒间隔闪烁，进入 boot 状态成功。

四、快速使用

4.1、添加模块

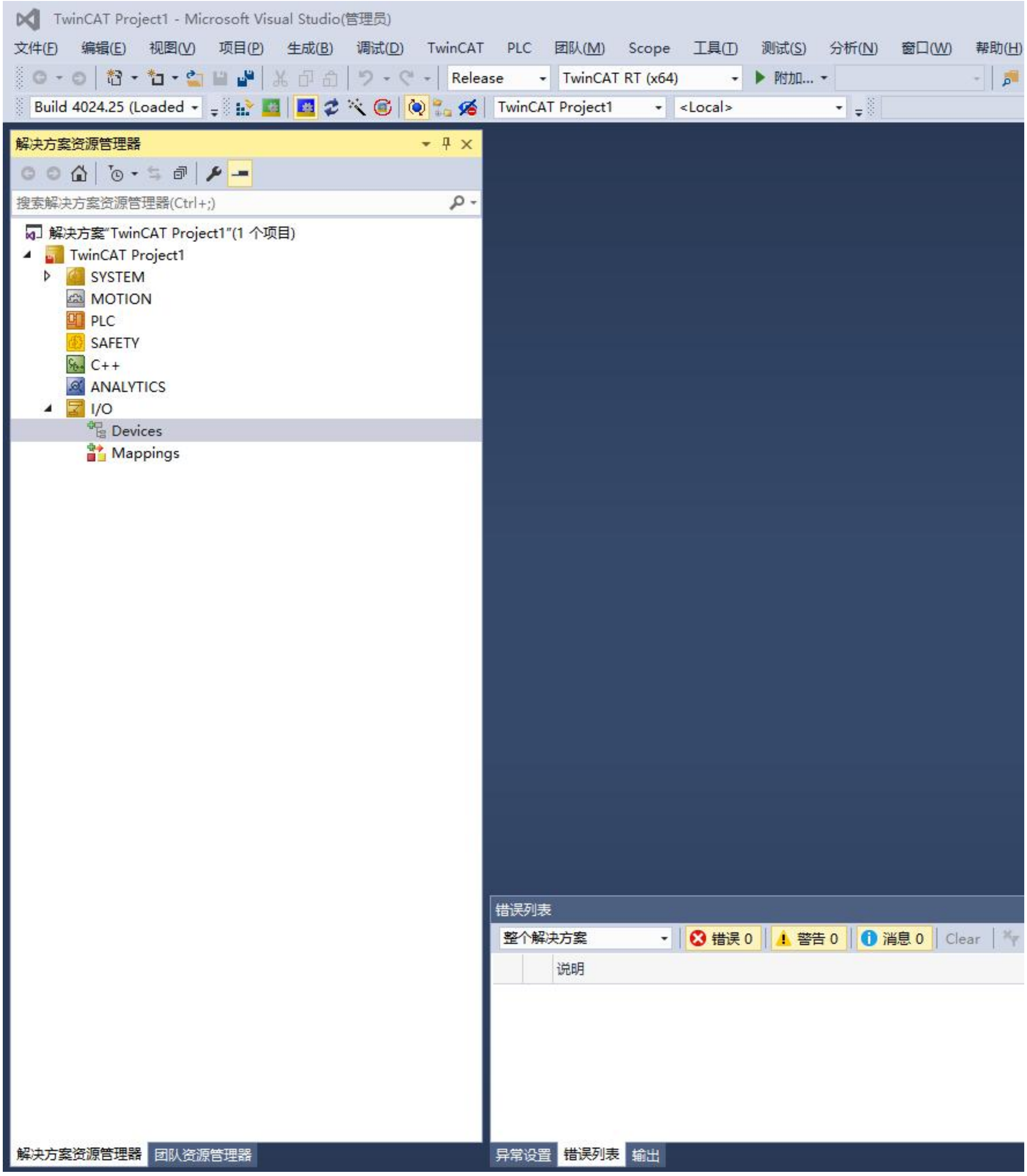
本章节对模块使用进行说明，以实现产品的快速使用，使用 TwinCAT 3 为例。

将 etherCAT 所需 XML 拷贝至主站安装文件夹下的模块 xml 文件夹下，以 TwinCAT 3 为例，如下图：

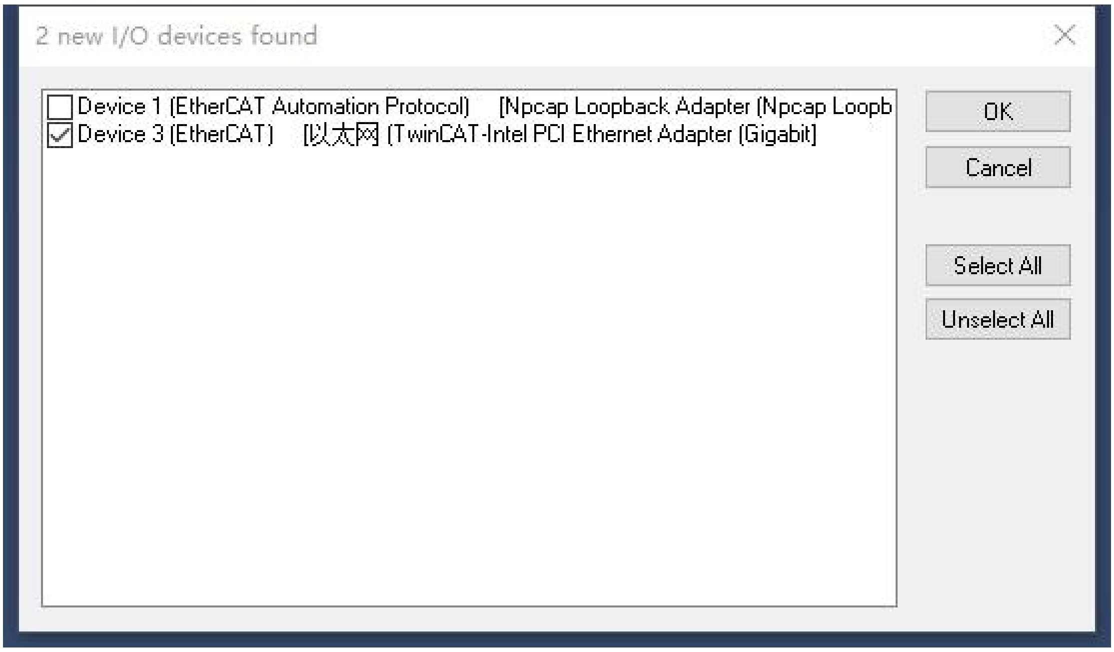


右击桌面右下角 TwinCAT 图标，选择“System Manager”，打开 TwinCAT 软件，如下图所示：

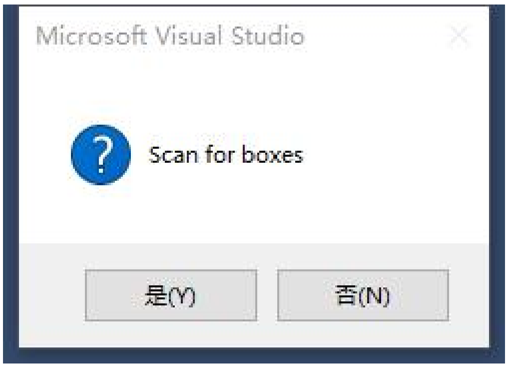
打开 TwinCAT 软件后，新建工程或打开已有工程，在“I/O –> Devices”处右击选择“Scan Devices”



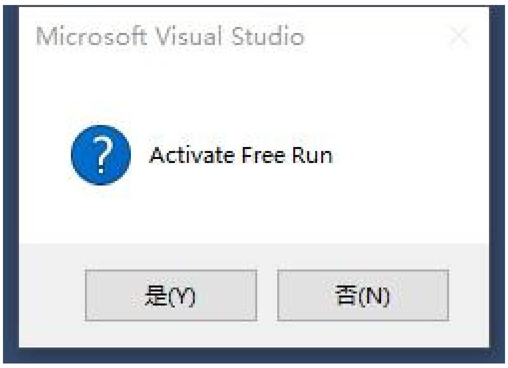
勾选当前扫描到的“本地连接”网卡



弹出窗口“Scan for boxes”选择 “是”



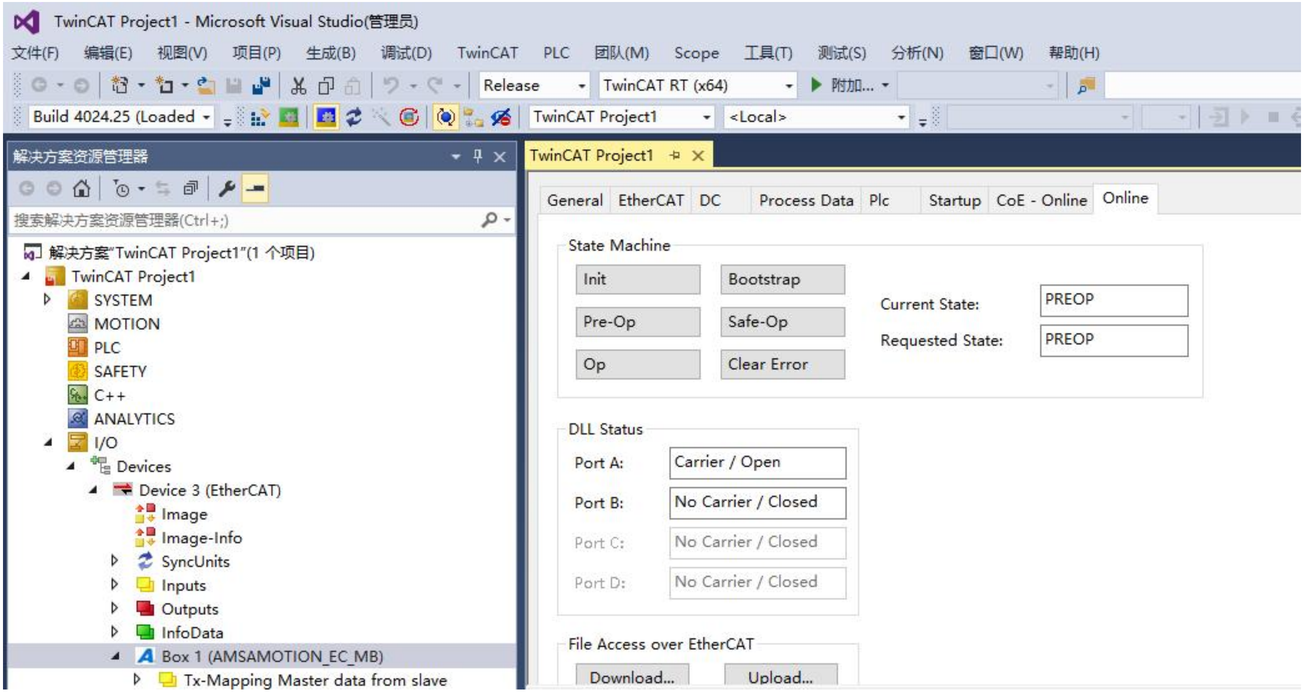
弹出窗口“Activate Free Run”选择“否”



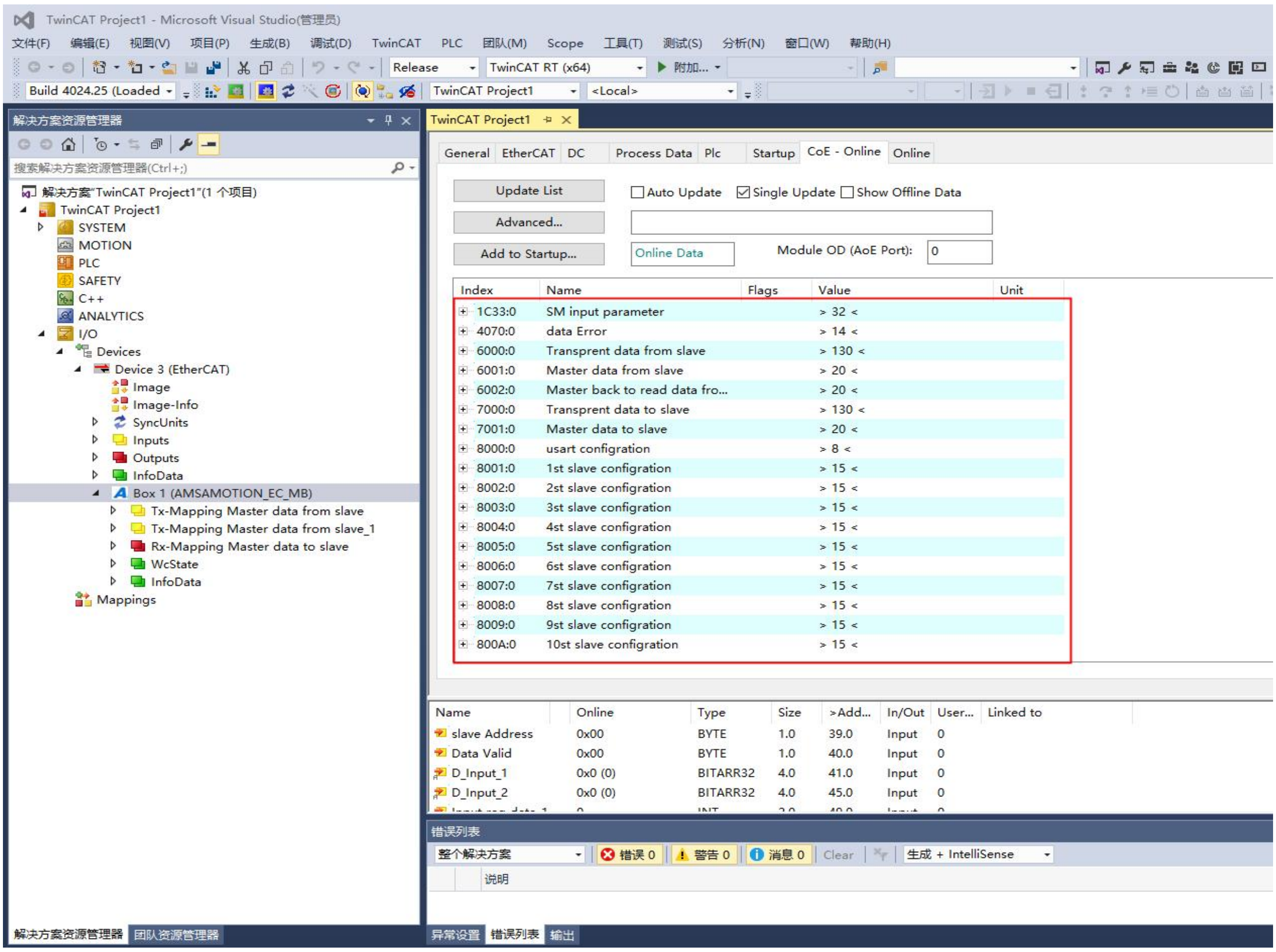
单击 Box1，进入 Online 选项，当前状态 Pre-Op，如果在其他状态，请切回 Pre-Op：

Note：如果不是处于 Pre-Op 模式下，大部分对象字是不可修改的，若修改，TwinCAT

会报错。



单击 CoE-Online，下拉可以见到本模块所有对象字典：



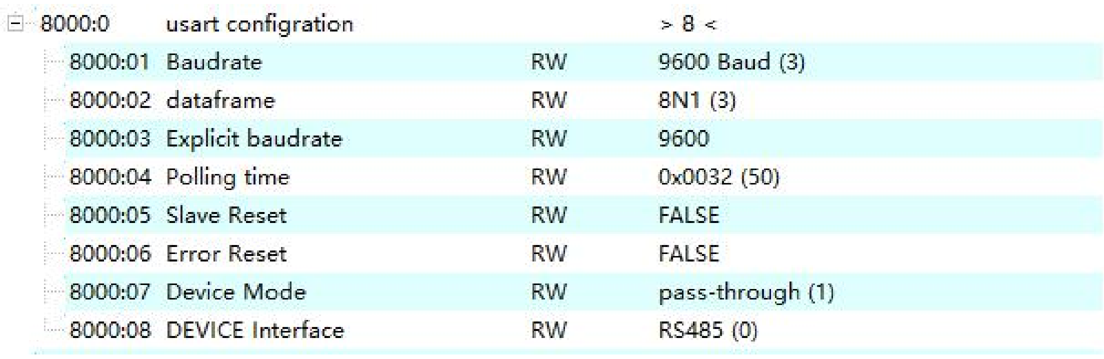
4.2、ModBus 功能

本例将其设置为 Modbus 模式，从站使用一个 16 位数字输入输出模块，示例用模块参数如下：

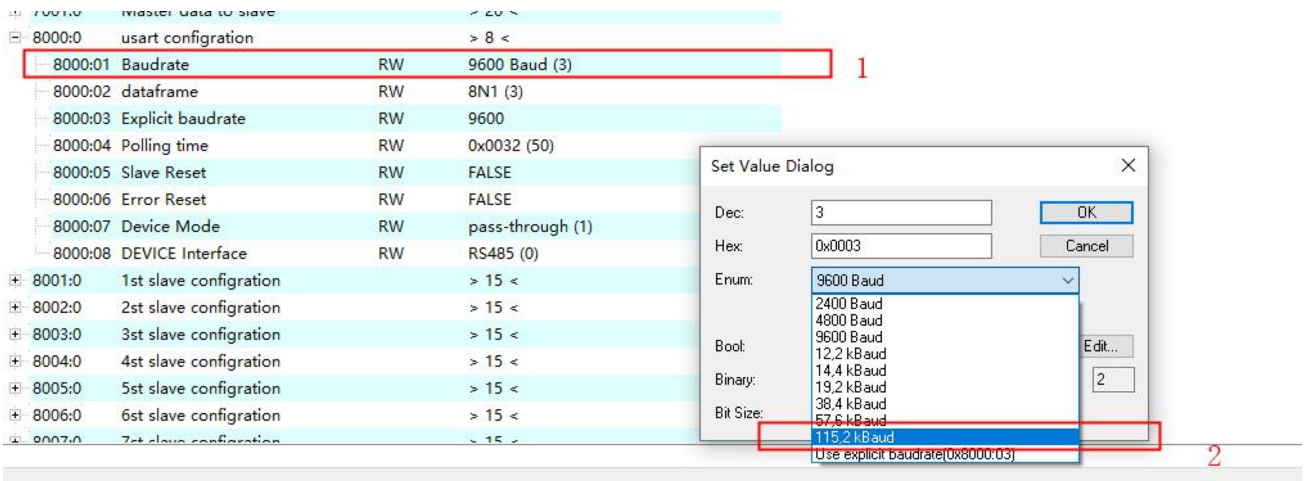
Modbus 从站地址	1
Modbus 从站接口	RS485
Modbus 从站波特率	115200
Modbus 从站数据格式	8N1
Modbus 从站线圈数量	16
Modbus 从站离散输入数量	16

Note：在单一对象字典无法将某一个从站数据读取完毕时，可以使用多个对象字典读取同一个从站中的值，将寄存器和线圈的起始地址进行相应偏移即可。

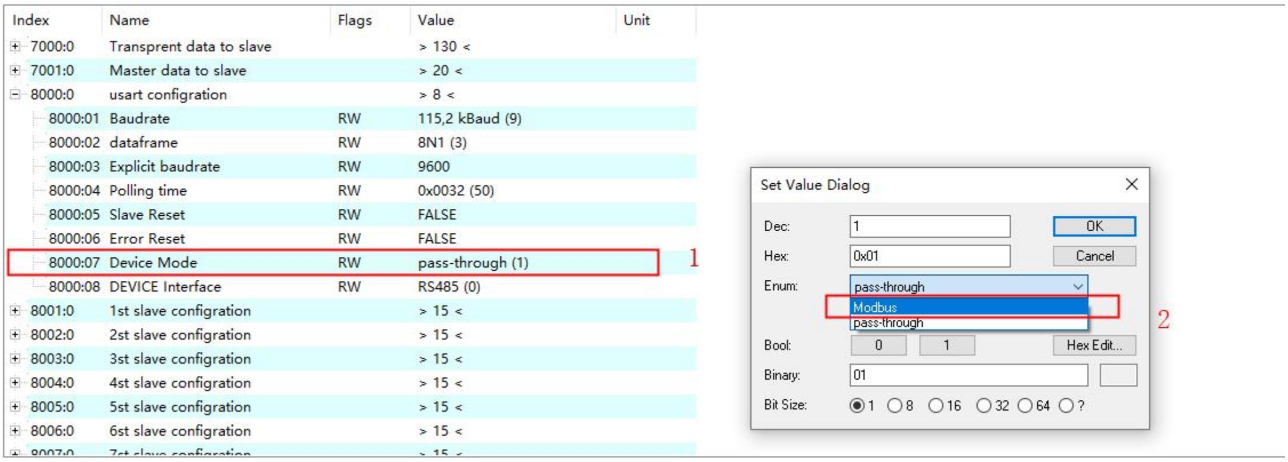
1）单击展开对象字典 8000 对象字：



2)选择对象字 8000：1，双击下图“1”处，然后选择“2”处选项，然后点击 OK：



3）选择对象字 8000：7，双击下图“1”处，然后选择“2”处选项，然后点击 OK：



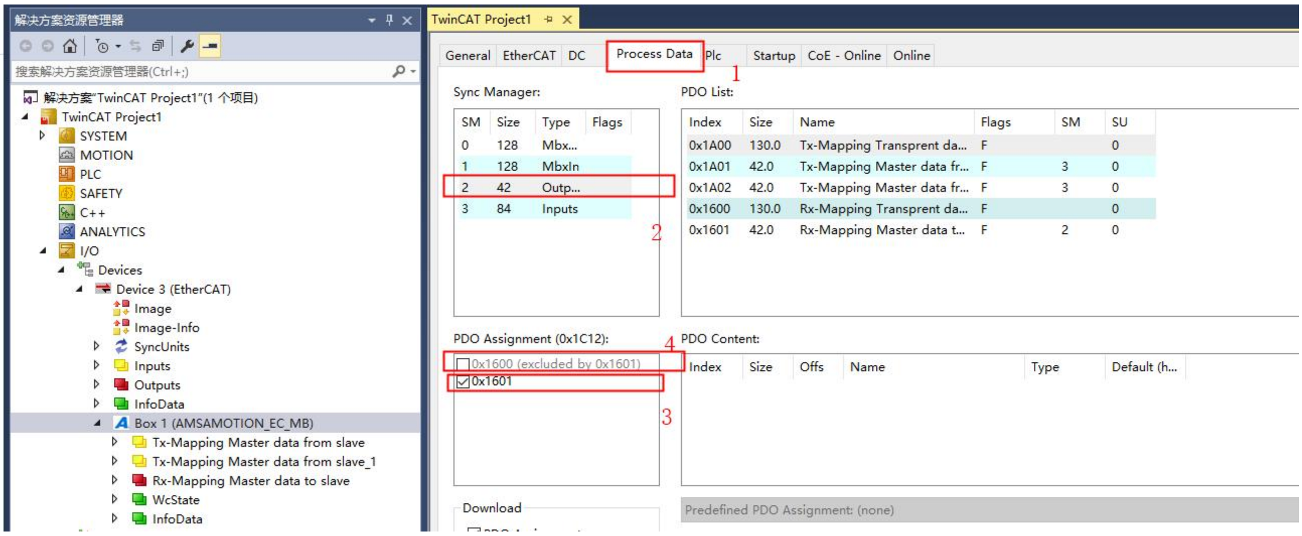
4）完成后对象字 8000 所有子索引如下图：

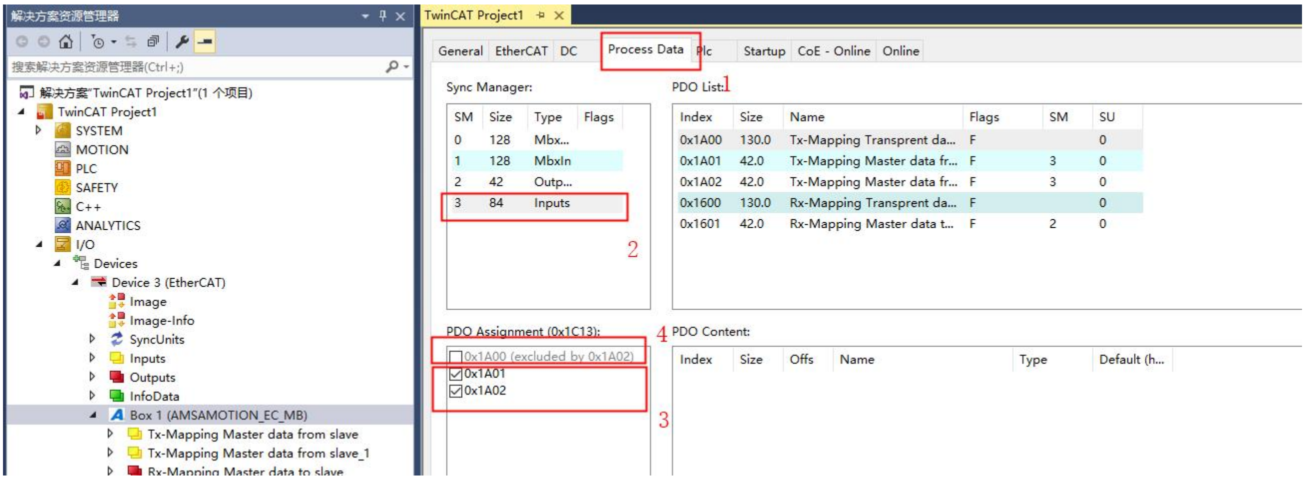
8000:0	usart configration		> 8 <
8000:01	Baudrate	RW	115,2 kBaud (9)
8000:02	dataframe	RW	8N1 (3)
8000:03	Explicit baudrate	RW	9600
8000:04	Polling time	RW	0x0032 (50)
8000:05	Slave Reset	RW	FALSE
8000:06	Error Reset	RW	FALSE
8000:07	Device Mode	RW	Modbus (0)
8000:08	DEVICE Interface	RW	RS485 (0)
8001:0	1st slave configration		> 15 <

5）展开对象字 8001，同样将其设置为下图所示：

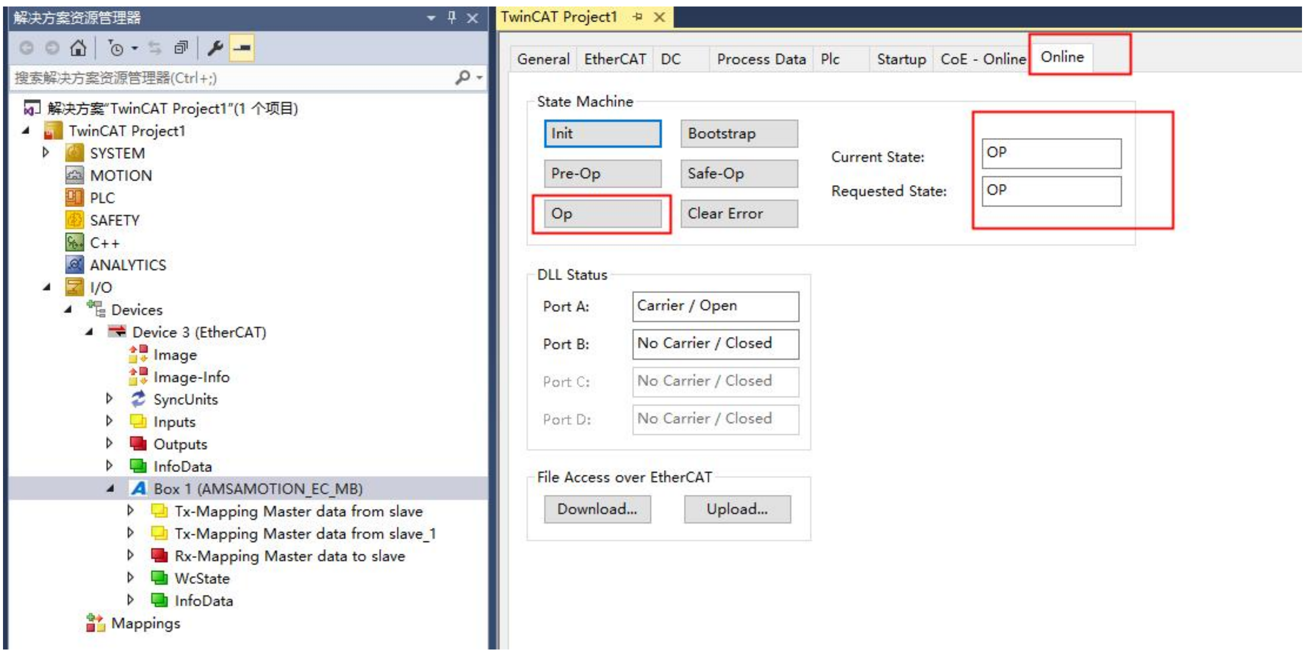
8001:0	1st slave configration		> 15 <
8001:01	slave Addr	RW	0x01 (1)
8001:02	coil is readable	RW	TRUE
8001:03	Keeps the register readable	RW	FALSE
8001:05	Slave coil start address	RW	0x0000 (0)
8001:06	The number of slave coil	RW	0x10 (16)
8001:08	slave Discrete input start address	RW	0x0000 (0)
8001:09	The number of slave Discrete input	RW	0x10 (16)
8001:...	Slave input register start address	RW	0x0000 (0)
8001:...	The number of Slave input register	RW	0x00 (0)
8001:0E	Slave hold register start address	RW	0x0000 (0)
8001:0F	The number of Slave hold register	RW	0x00 (0)

6）单击标题栏中 Process Data（图中“1”处），然后分别单击下两张图中的“2”处，检查图中“3”是否为图中所示，如果不是，先点击图中“4”所示位置，将其取消，然后点击“3”选中：

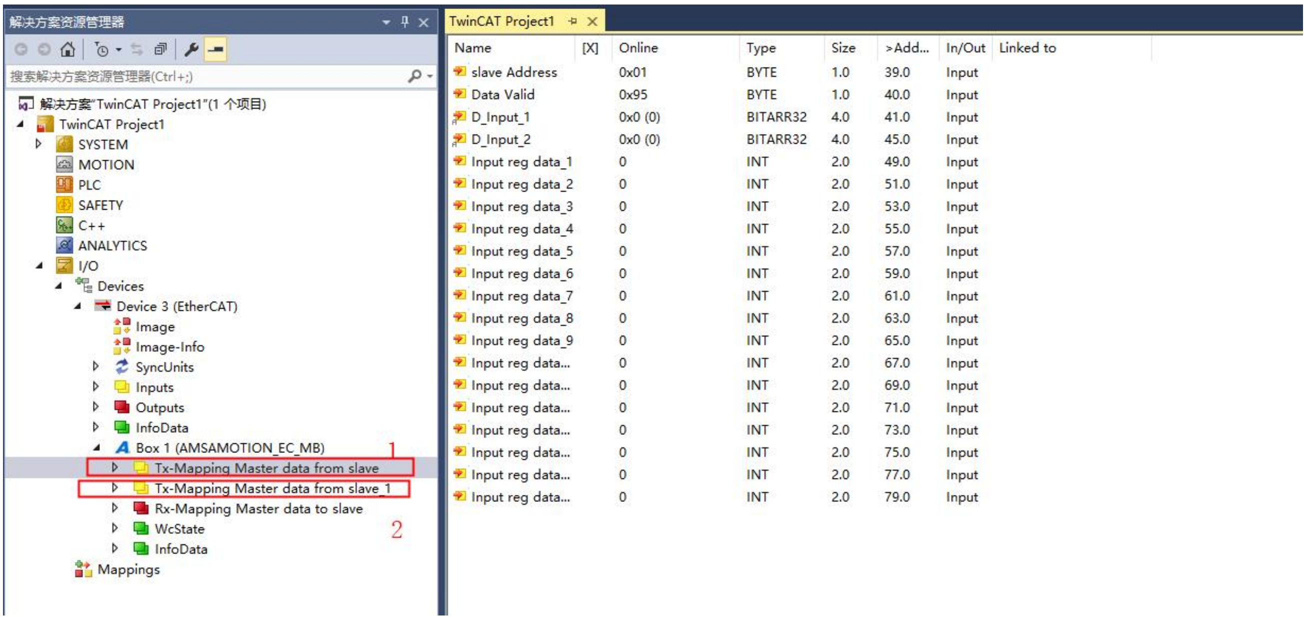


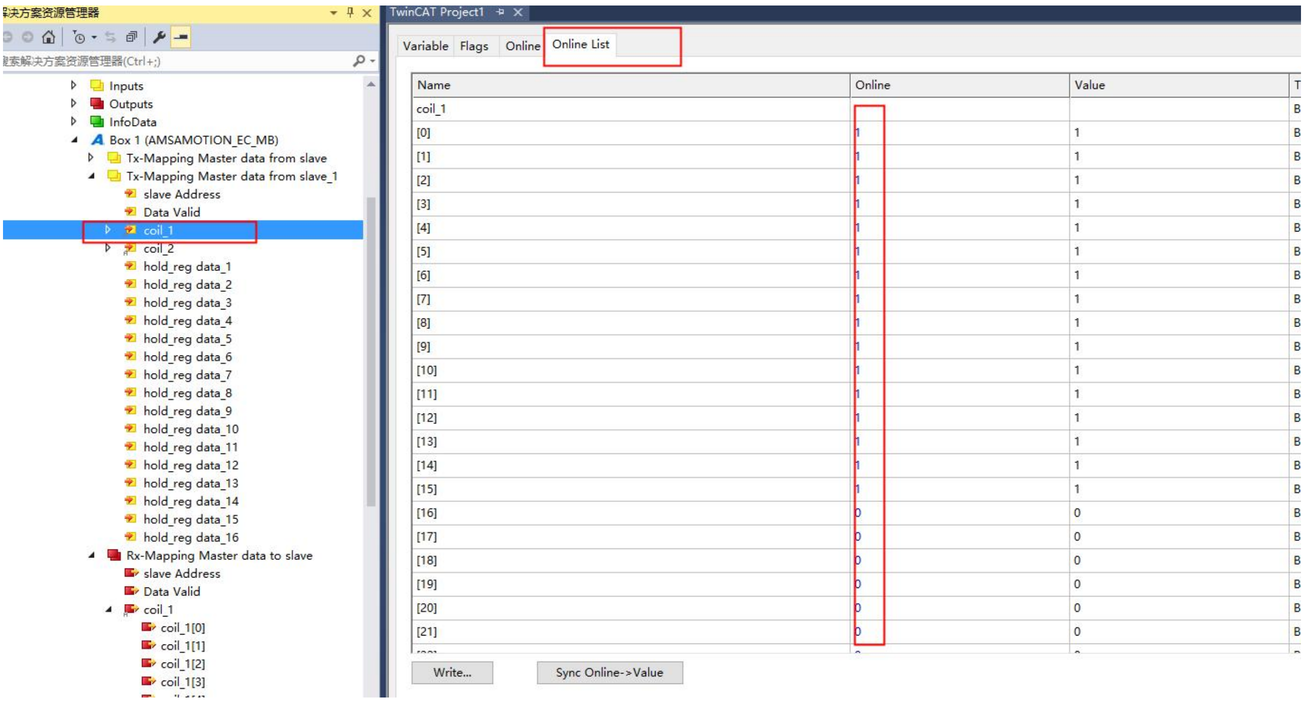


7）点击标题栏，将页面切回 Online，将状态更改为 Op，此时已经在开始运行，RUN 灯以 1 秒周期闪烁，RS485 灯以 0.5 秒周期闪烁：

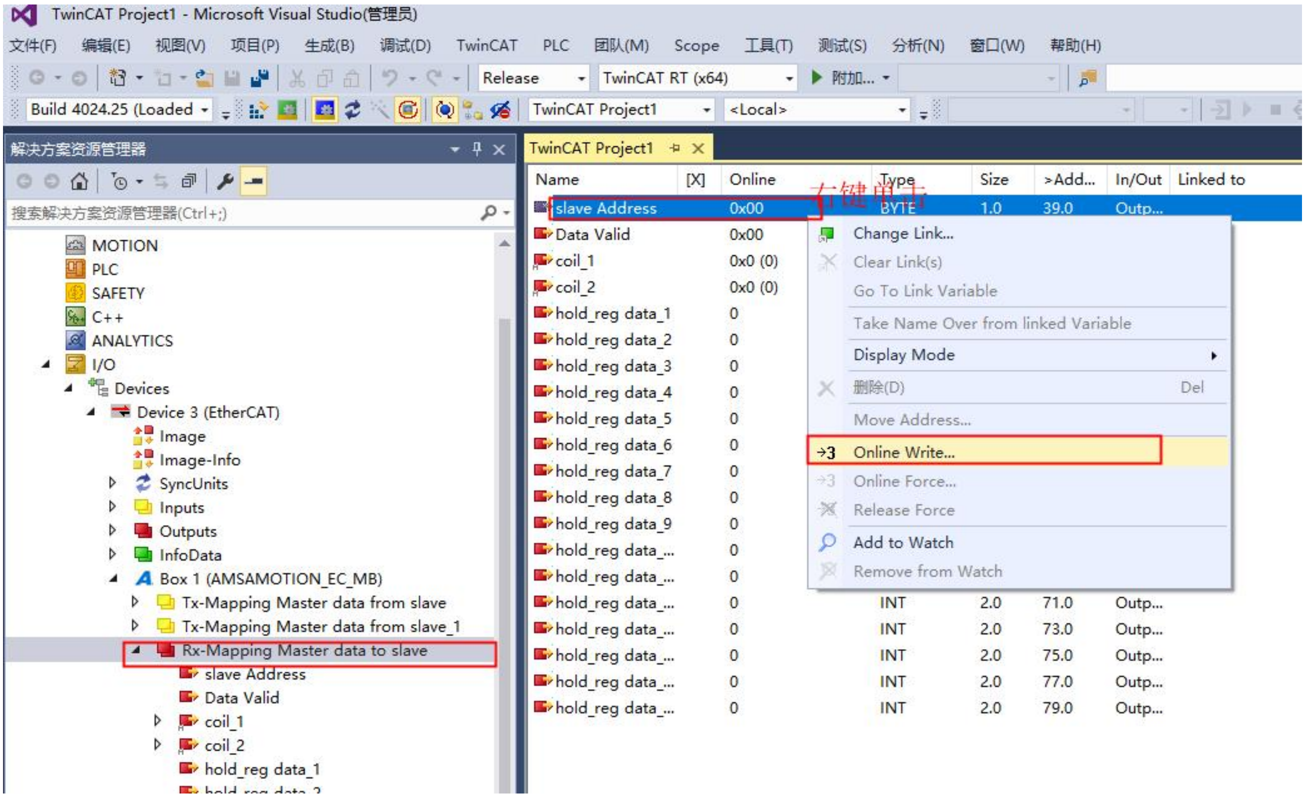


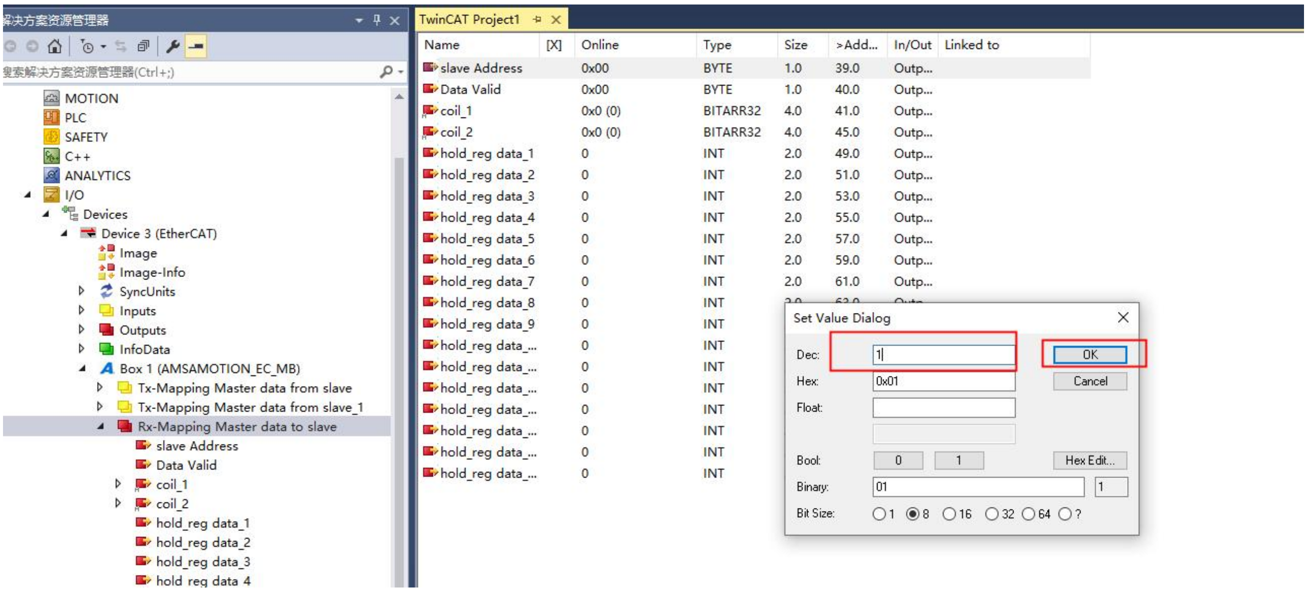
8）此时单击下图中的“1”或者“2”，可以看到 Data Valid 位开始更新，如果有数据，相应的位会置 0 或者置 1，其中 DI 或者回读的 DO 在当前窗口无法直接看到变化，按下图 2 所示进行操作（显示的是回读的 DO），可看到每一位的状态。

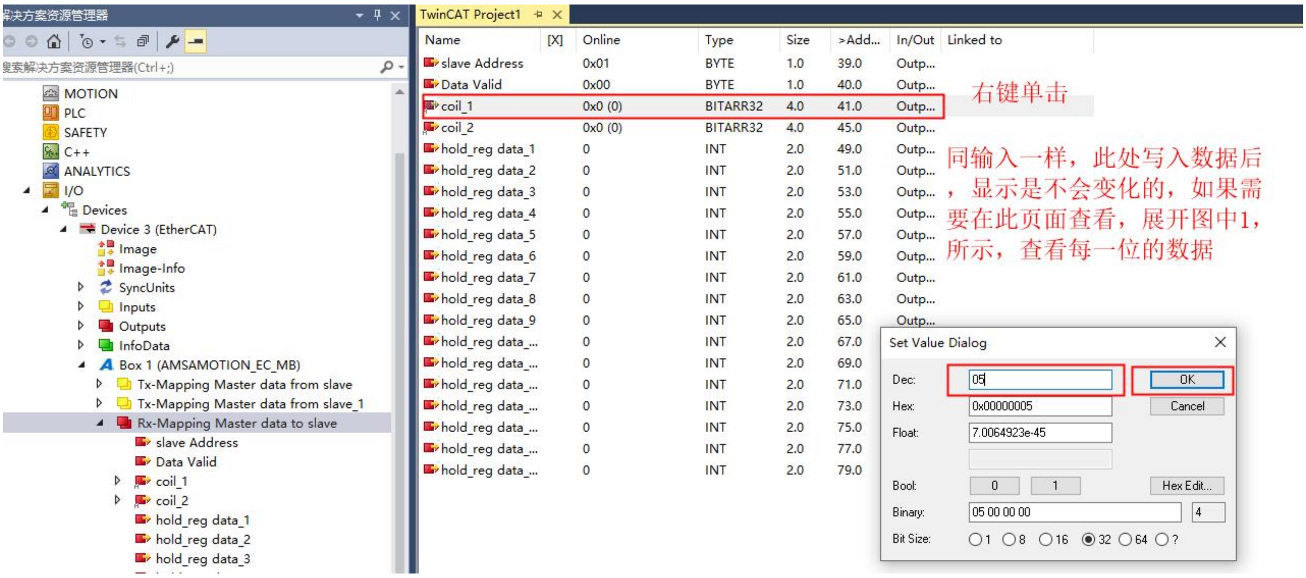


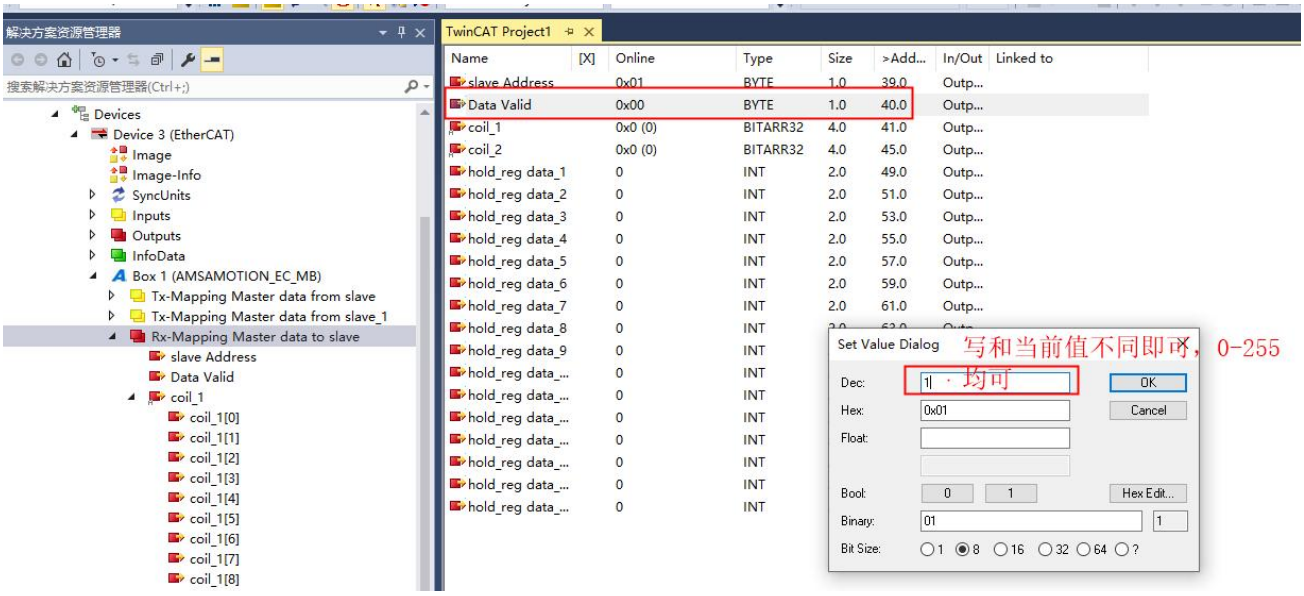


9）如果需要写线圈输出或者写保持寄存器，将分别将要写的从站地址，需要写的数据写入下图中的地方，然后更改 Data Valid 位，modbus 将会把需要更新的从站数据发出。

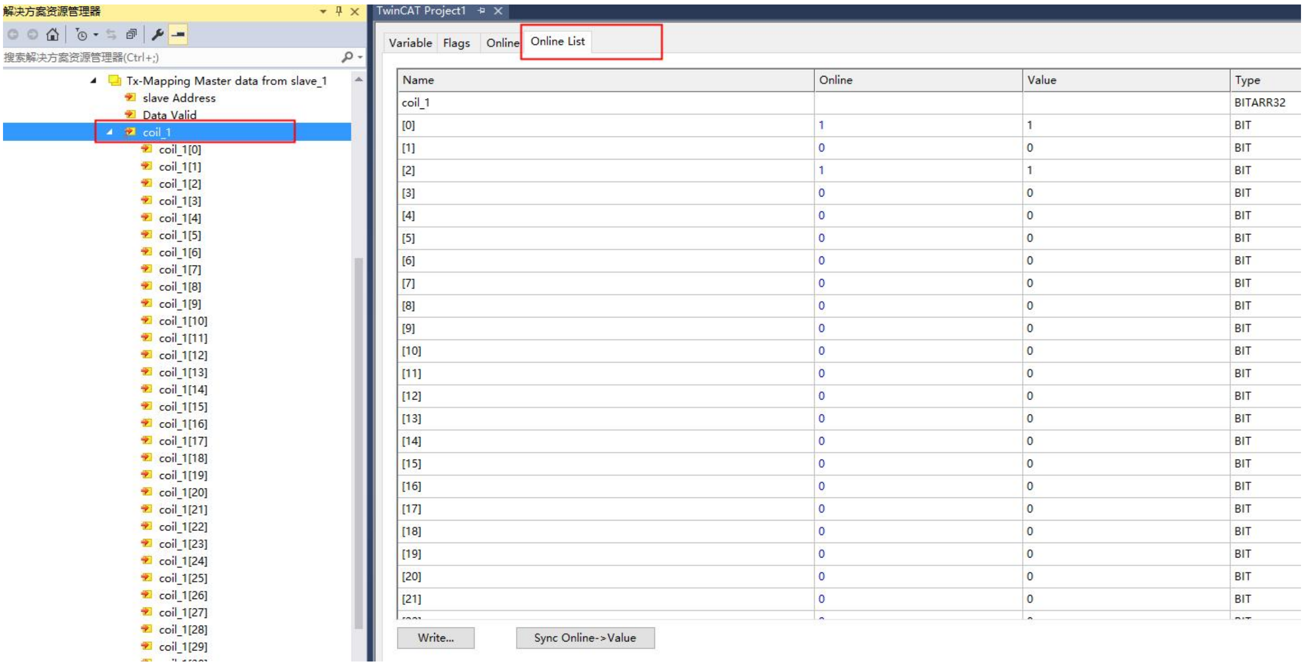

TwinCAT Project1 - Microsoft Visual Studio(管理员)
解决方案资源管理器
TwinCAT Project1
Rx-Mapping Master data to slave
slave Address
右键单击
Change Link...
Clear Link(s)
Go To Link Variable
Take Name Over from linked Variable
Display Mode
删除(D)
Move Address...
Online Write...
Online Force...
Release Force
Add to Watch
Remove from Watch


Set Value Dialog
Dec:
Hex:
Float:
Bool:
Binary:
Bit Size:
OK
Cancel
Hex Edit...





10）按上面写入后，读回线圈状态，第 0 位和第 2 位是 1，其他为均为 0，如下图：

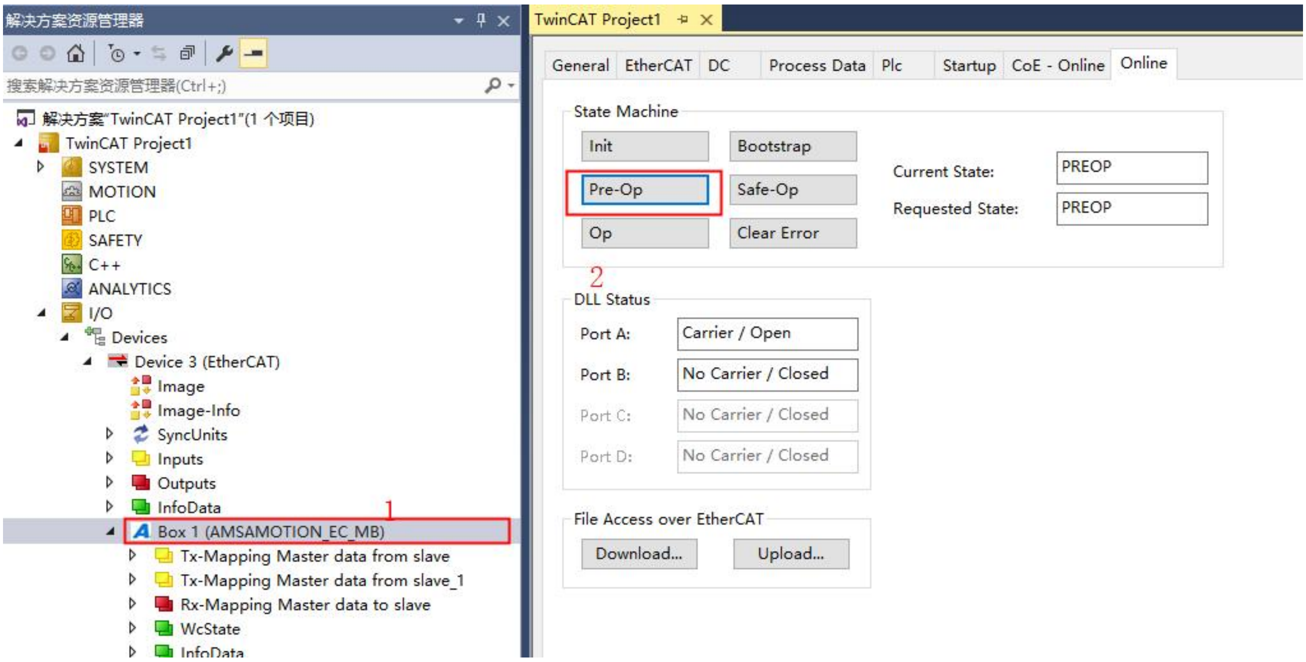


4.3、透明传输功能

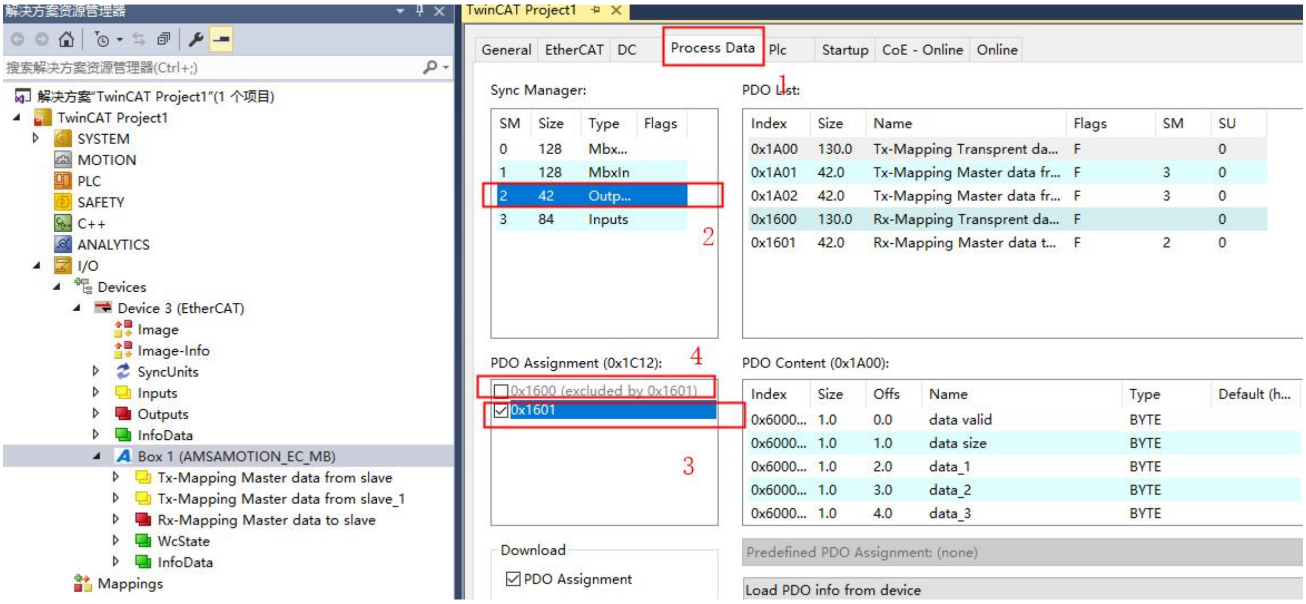
本章节在 4.1 章节基础上进行修改。

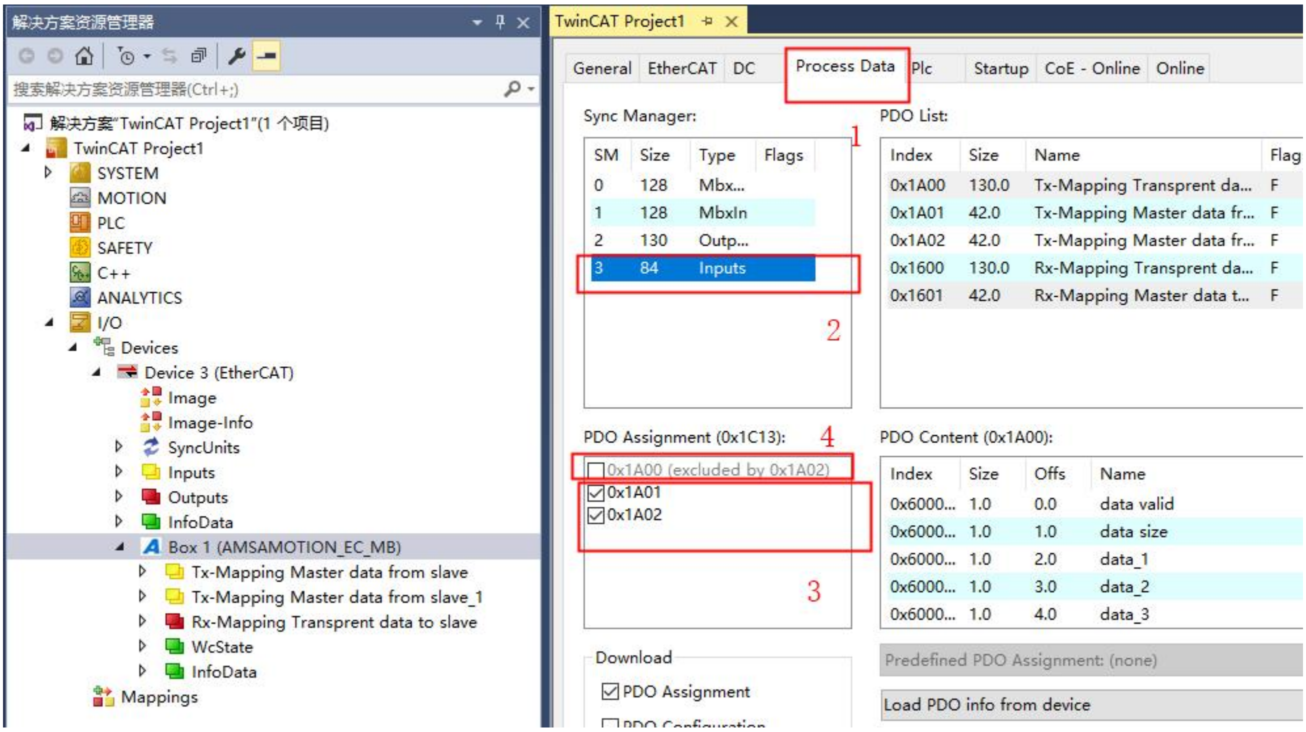
1）按下图单击“1”处，然后单击“2”，将运行模式切换回 Pre-Op。

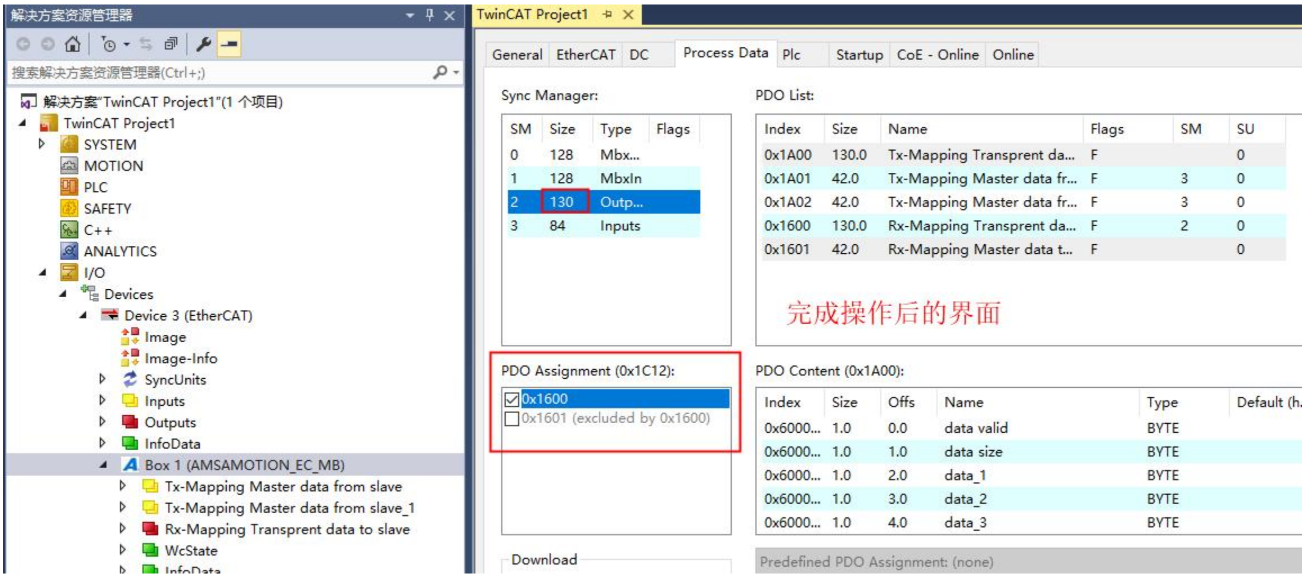
Note：如果不是处于 Pre-Op 模式下，大部分对象字是不可修改的，否则会报错。

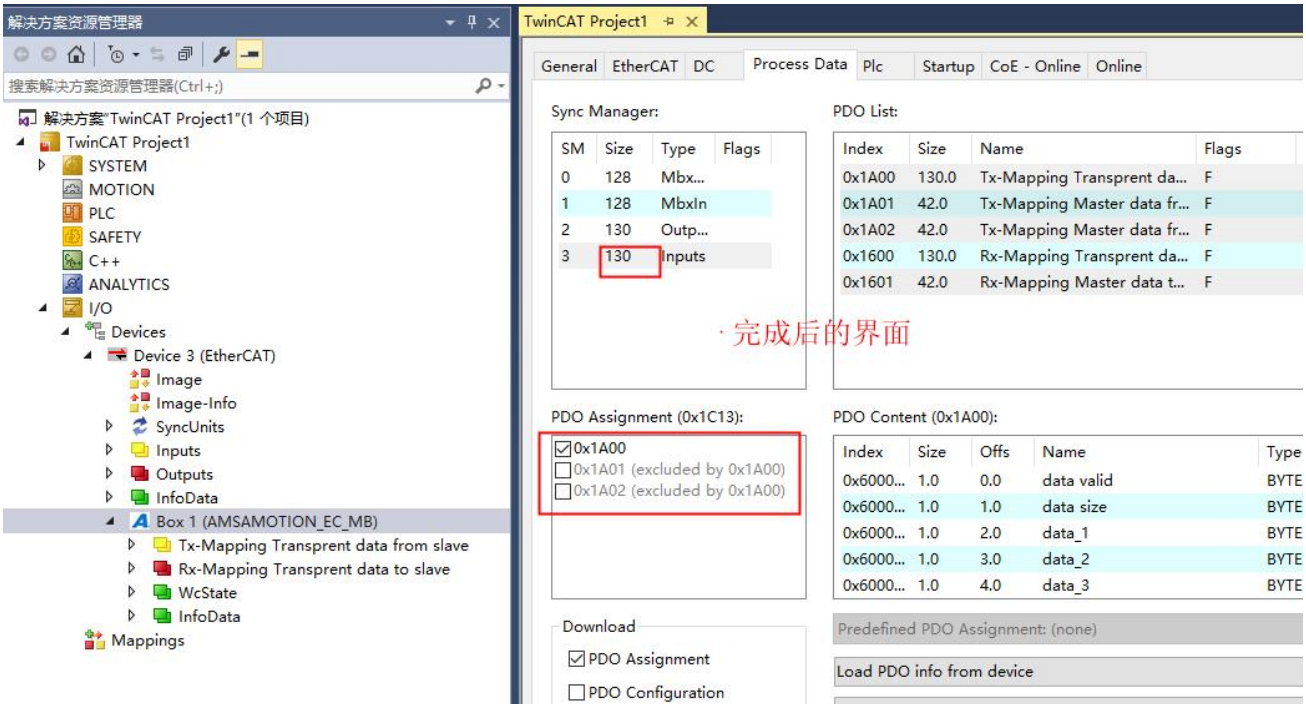


2）按下图将同步管理器输入输出更换，单击“1”处，将界面切换回 Process Data，然后单击“2”处，取消“3”处的勾选，后将“4”勾选上。

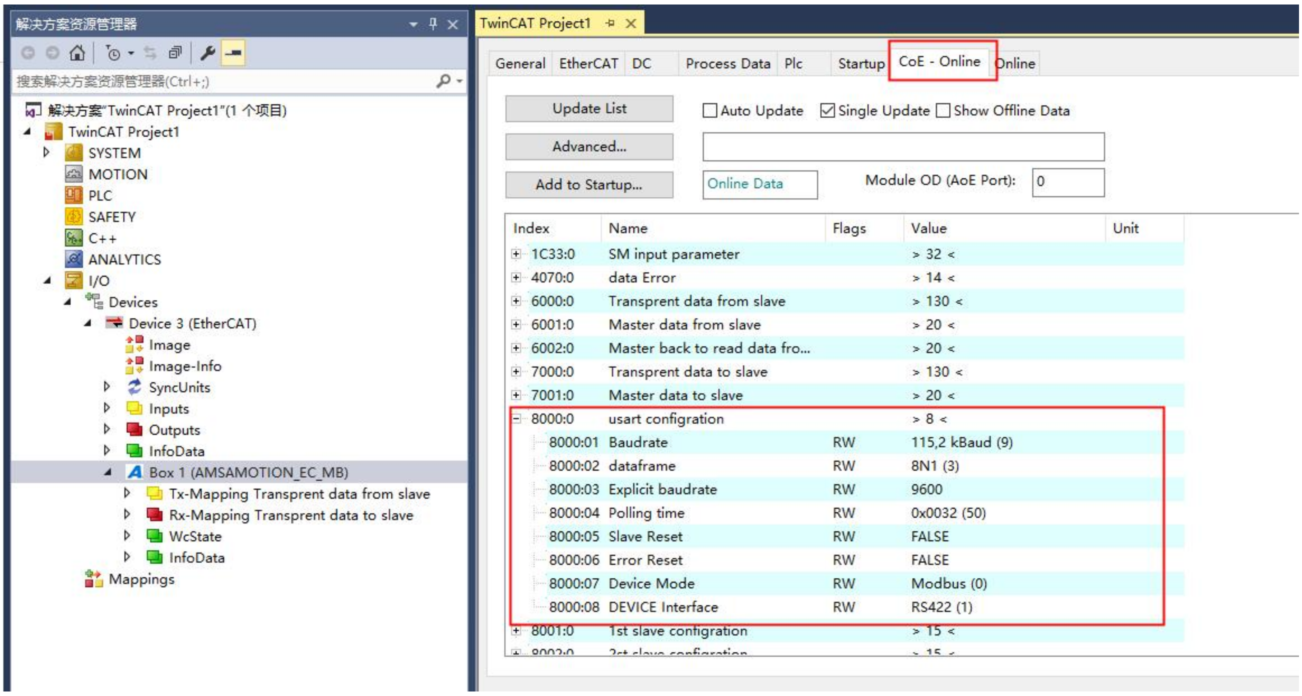

解决方案资源管理器
搜索解决方案资源管理器(Ctrl+;)
解决方案"TwinCAT Project1"(1 个项目)
TwinCAT Project1
SYSTEM
MOTION
PLC
SAFETY
C++
ANALYTICS
I/O
Devices
Device 3 (EtherCAT)
Image
Image-Info
SyncUnits
Inputs
Outputs
InfoData
Box 1 (AMSAMOTION_EC_MB)
Tx-Mapping Master data from slave
Tx-Mapping Master data from slave_1
Rx-Mapping Master data to slave
WcState
InfoData
Mappings
TwinCAT Project1
General
EtherCAT
DC
Process Data
Plc
Startup
CoE - Online
Online
Sync Manager:
PDO List:
PDO Assignment (0x1C12):
0x1600 (excluded by 0x1601)
0x1601
PDO Content (0x1A00):
Download
PDO Assignment
Predefined PDO Assignment: (none)
Load PDO info from device
1
2
3
4


解决方案资源管理器
搜索解决方案资源管理器(Ctrl+;)
解决方案"TwinCAT Project1"(1 个项目)
TwinCAT Project1
Box 1 (AMSAMOTION_EC_MB)
Tx-Mapping Master data from slave
Tx-Mapping Master data from slave_1
Rx-Mapping Transprent data to slave
WcState
InfoData
Mappings
General
EtherCAT
DC
Process Data
Plc
Startup
CoE - Online
Online
Sync Manager:
PDO List:
PDO Assignment (0x1C13):
0x1A00 (excluded by 0x1A02)
0x1A01
0x1A02
PDO Content (0x1A00):
Download
PDO Assignment
Predefined PDO Assignment: (none)
Load PDO info from device
1
2
3
4

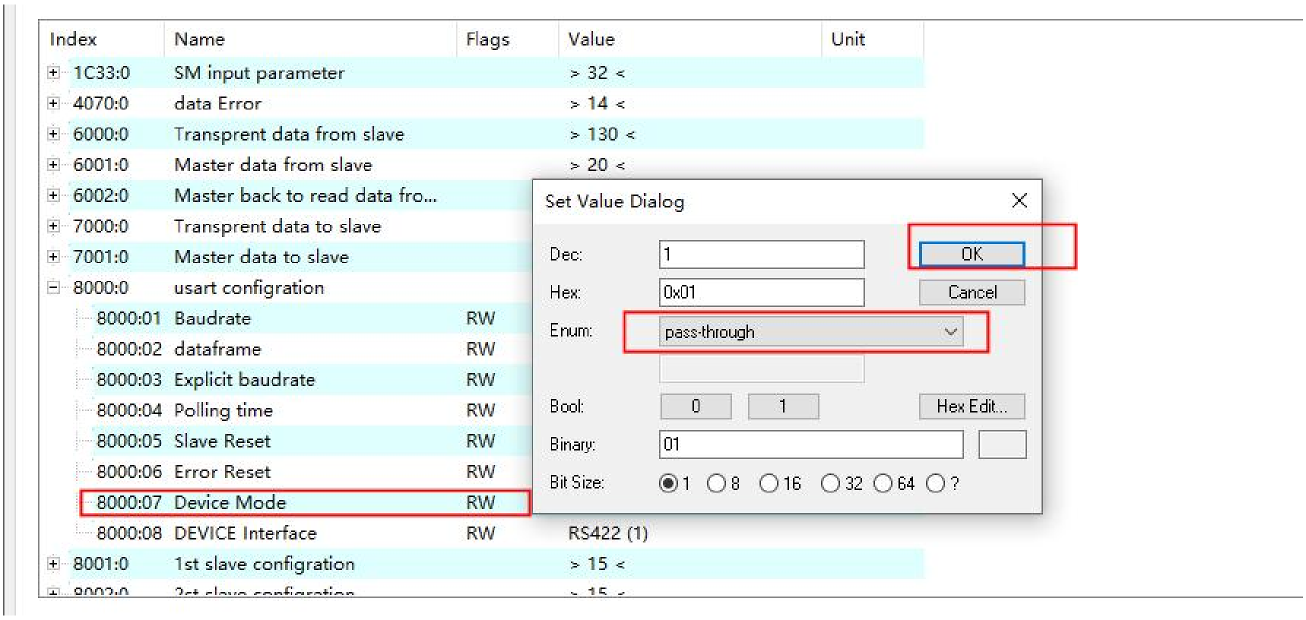




3）按下图点击标题栏，将界面切换到 CoE-Online,找到对象字典索引 8000，展开：

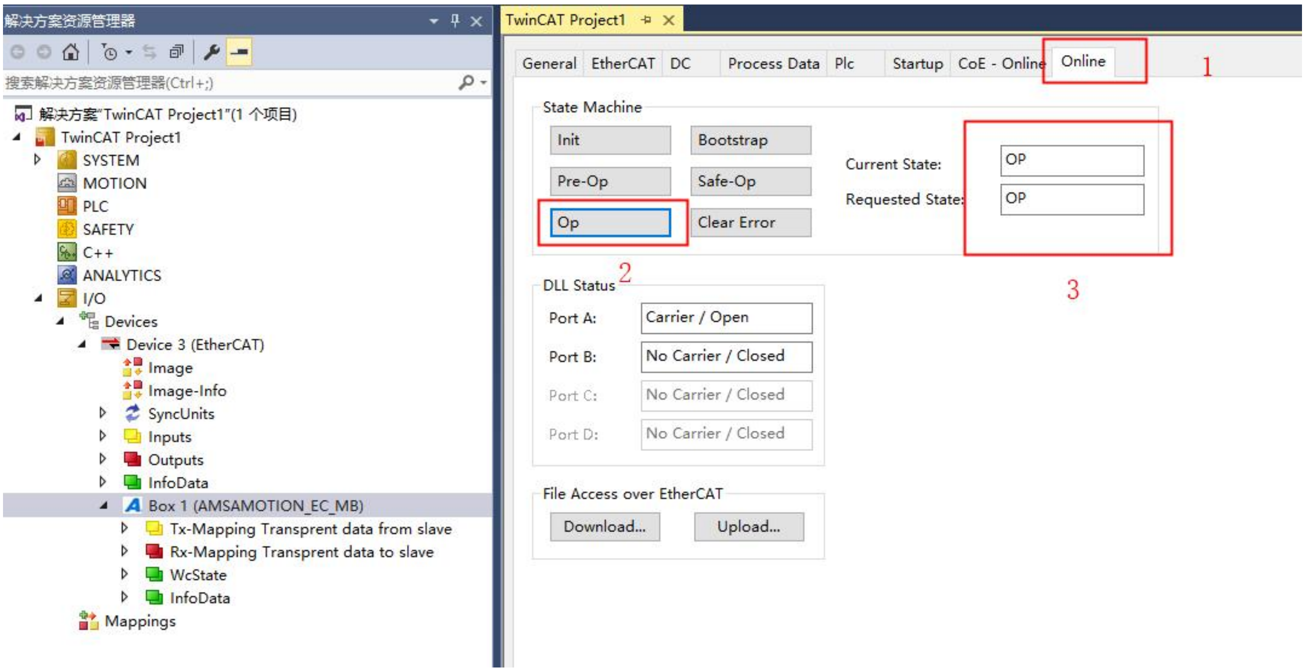


4）将索引 8000 下的子索引 7 修改为 pass-through 模式，如下：

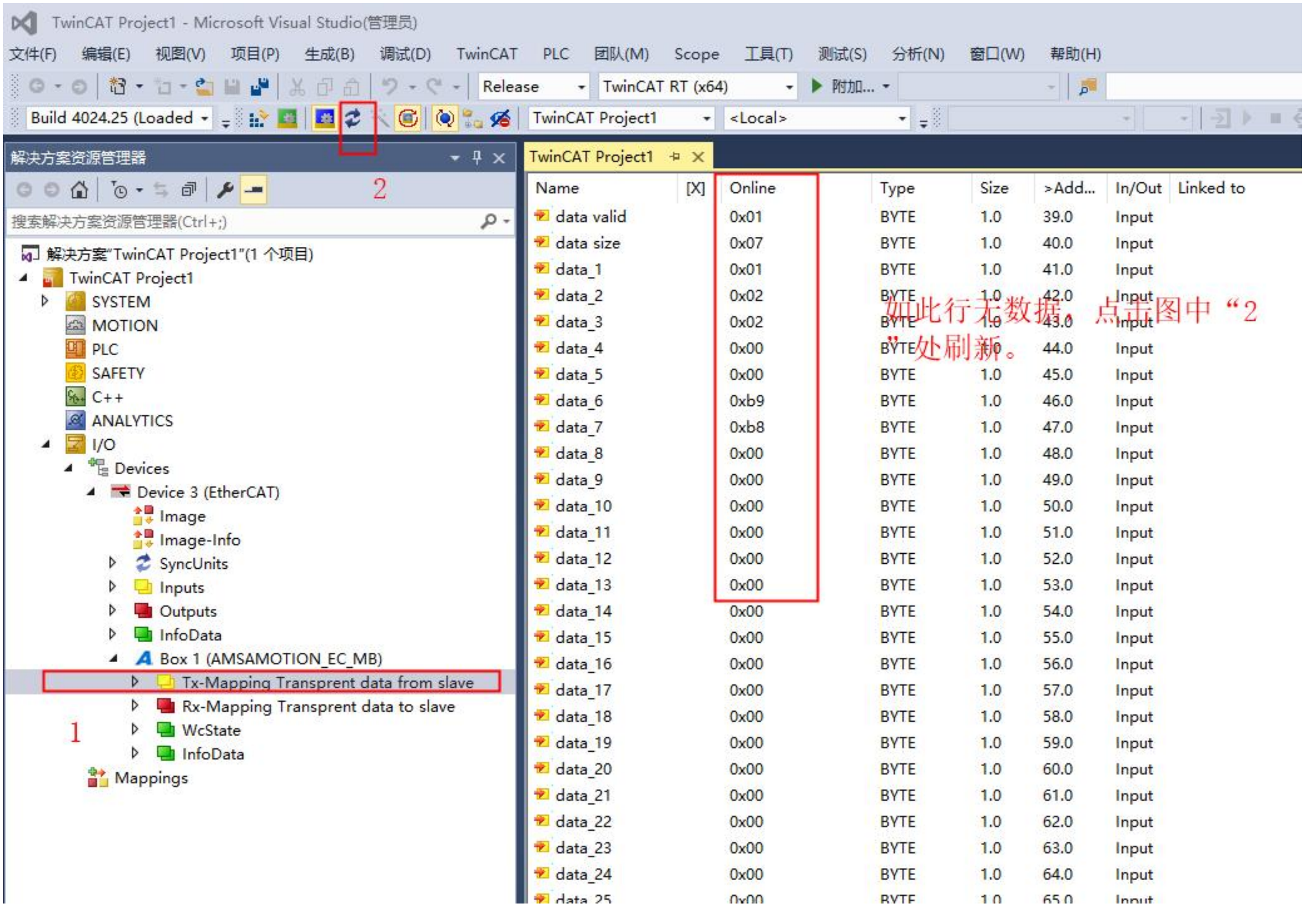


5）波特率、数据格式、接口等按实际需求进行设置，完成后将运行状态切换成 Op，此时 RUN 灯以 1 秒周期闪烁，RS485 或者 RS422 灯以 0.2 秒周期闪烁：

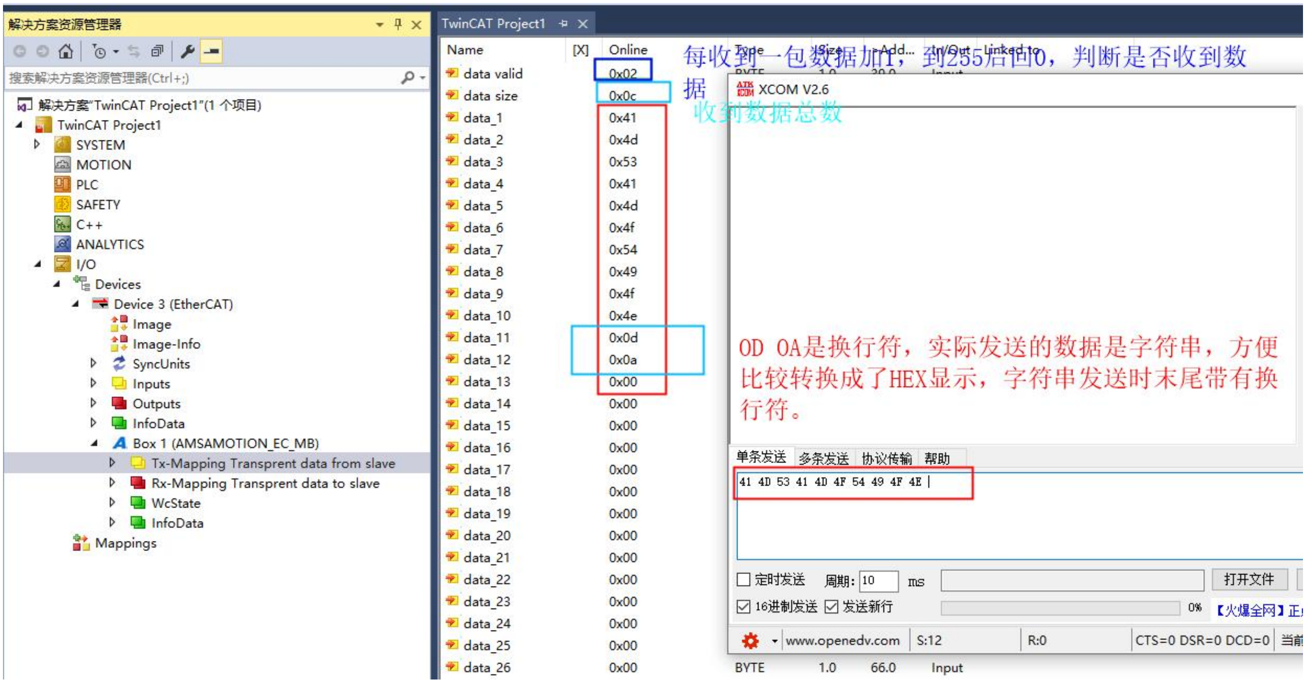
Note：使用透传功能时，索引 8001-800A 从站设置未使用，可以是任意合理的值。



6）将界面切换到输输出，如果 Online 行无数据，按下图操作：

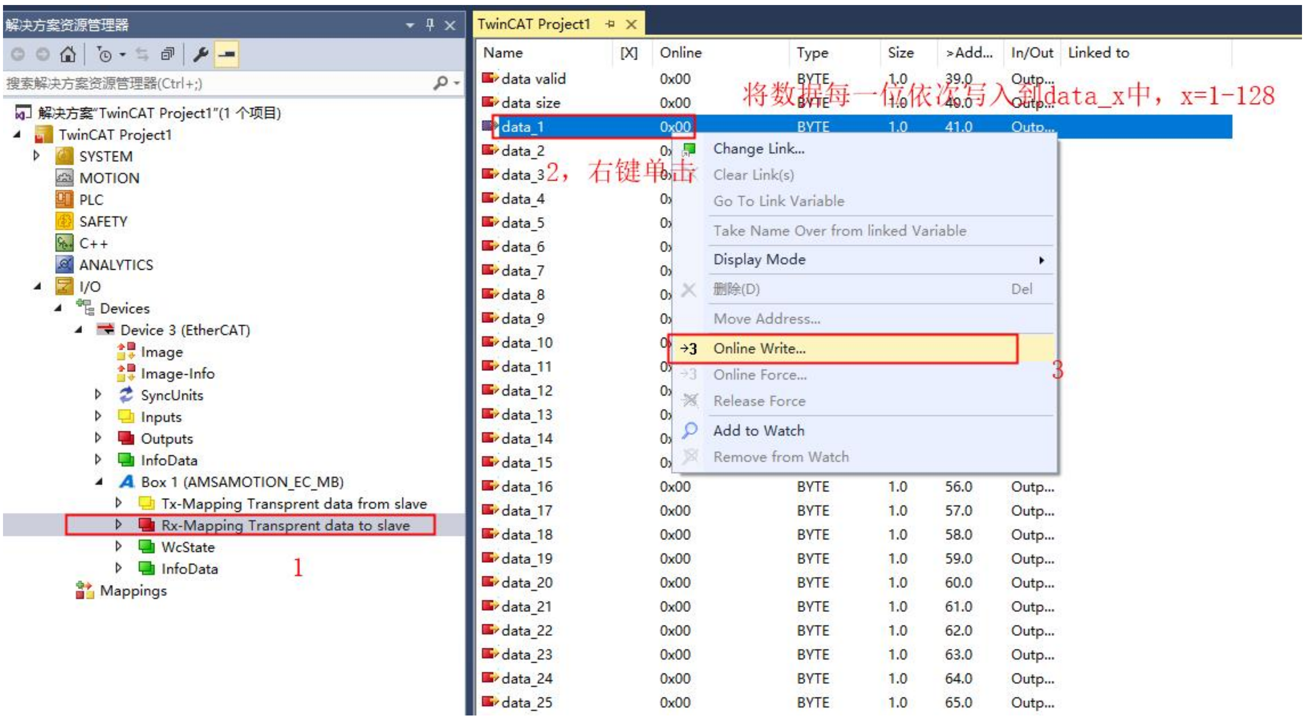


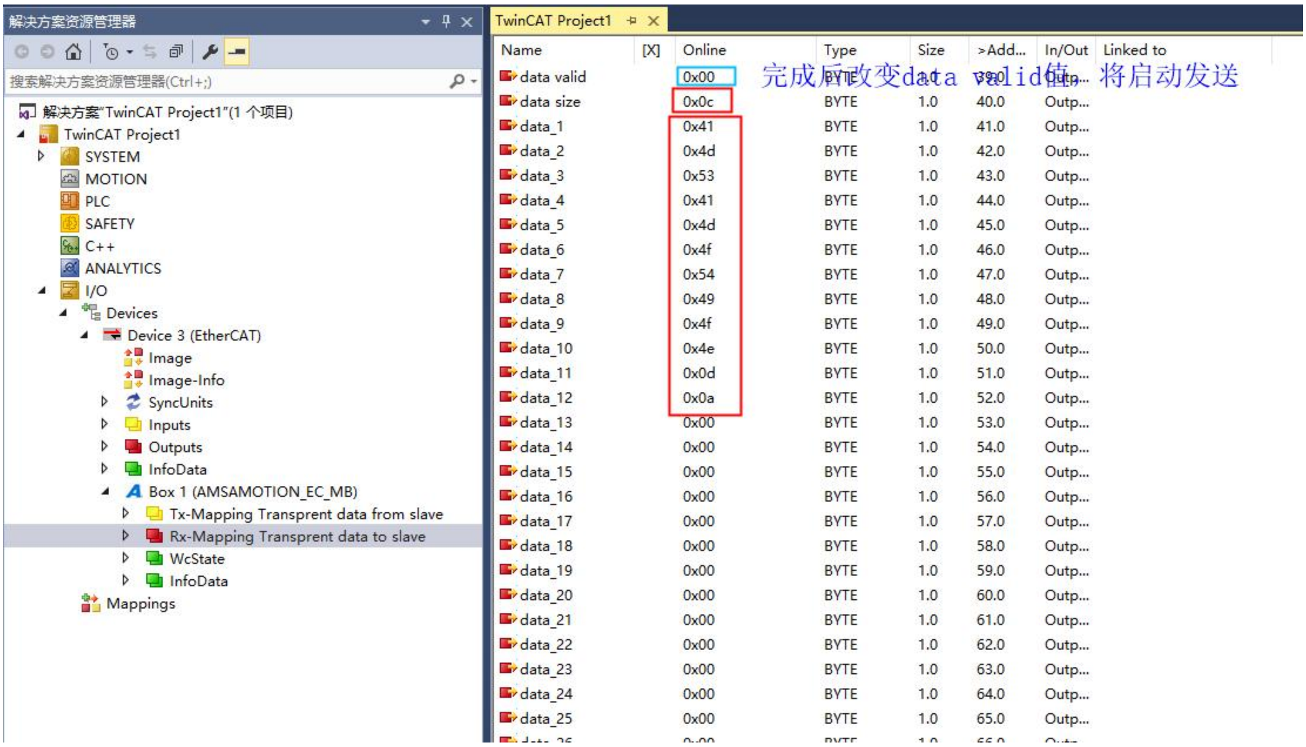
7）打开一个串口助手，连接好硬件，发送任意字符，完成后可以看到模块显示和串口发送的数据一致：



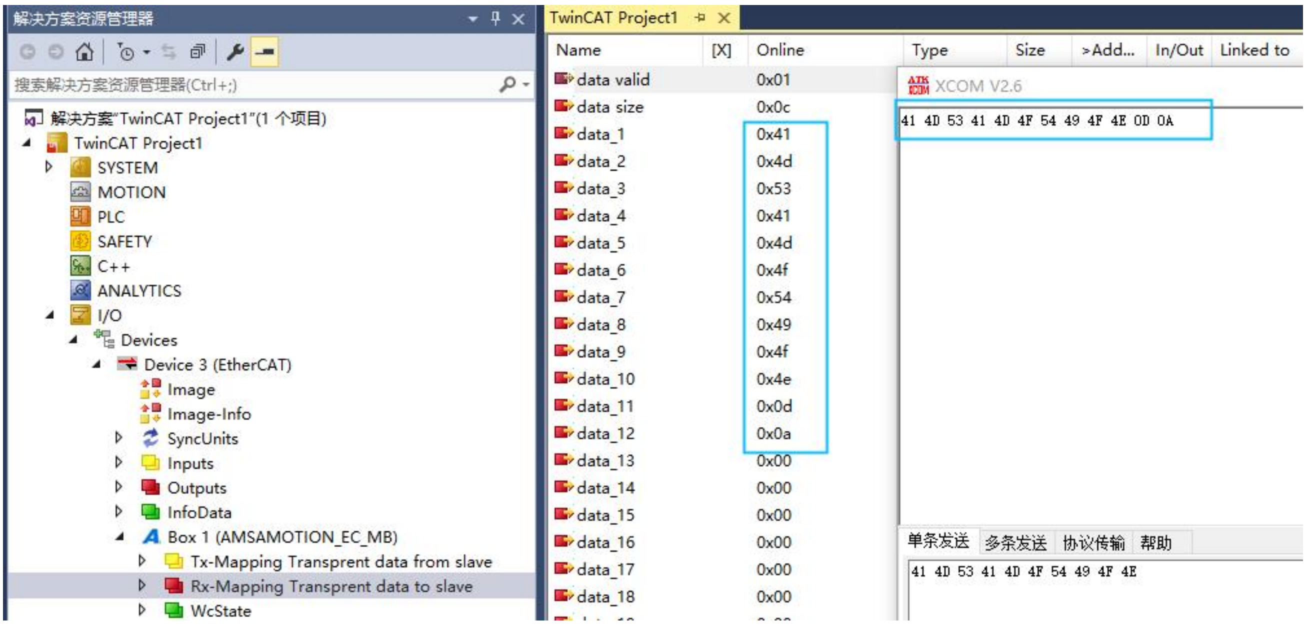
8）发送数据时如下图，切换到输出界面，将每一位数据写入 data_x 中，x=1~128，将要发送的数据长度写入 data_size 中,写完后如下图 2，完成后改变 data valid 位的值，将启动发送。

Note：写入数据时注意进制转换。





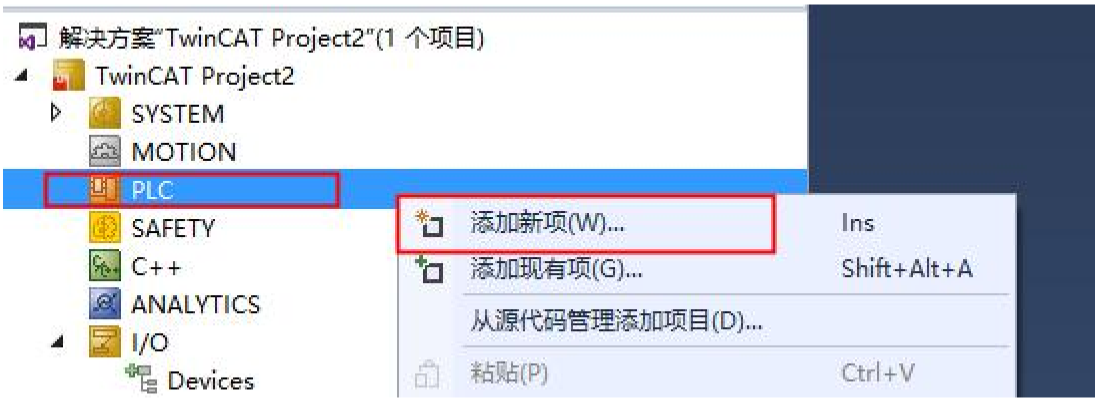
9）发送完成后如下图：



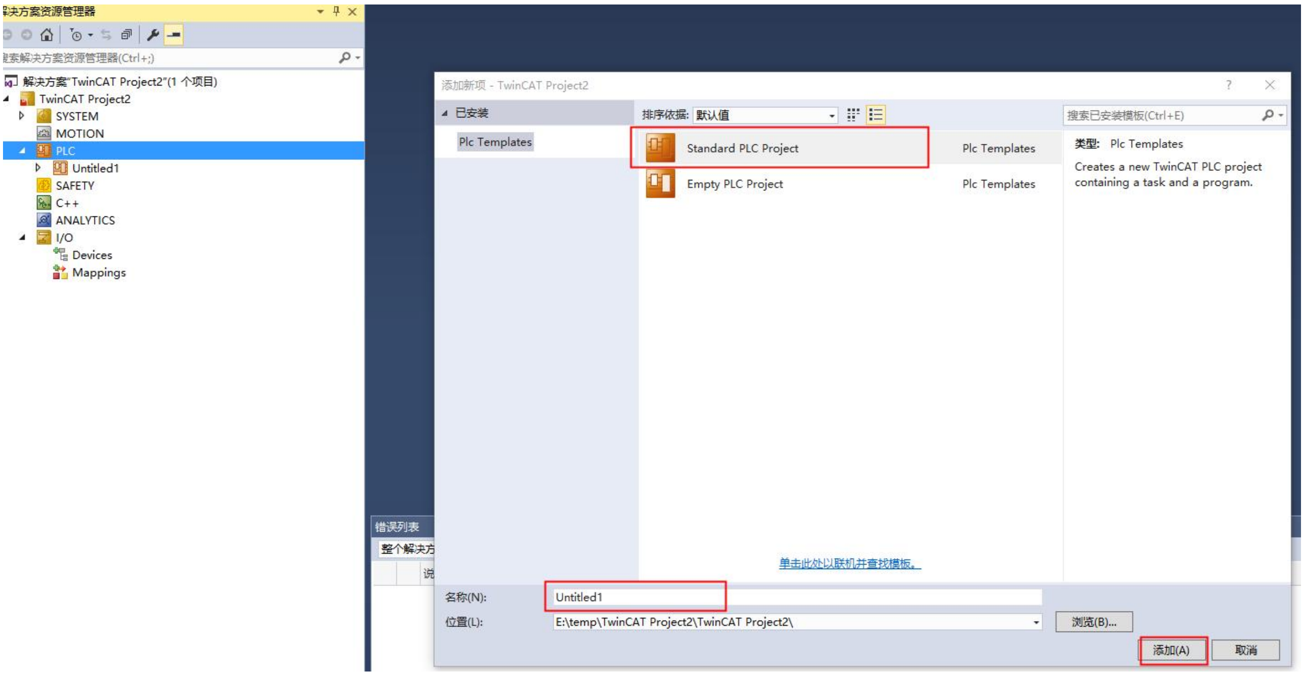
4.4、配合 PLC 使用

本章节利用 10 个 16 位输入输出模块配合本模块，利用简单流水灯工程示例，演示添加 PLC 工程及映射变量，并保存每个从站中回读数据。

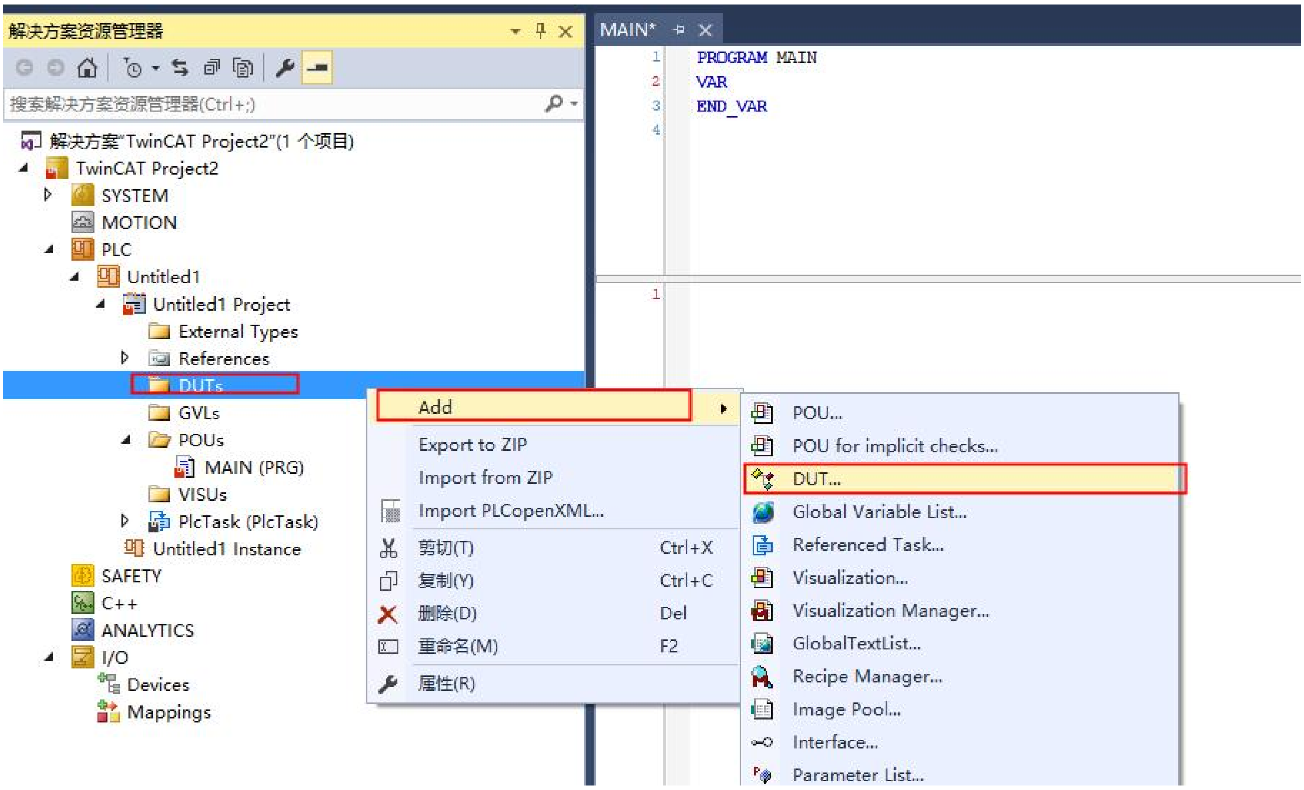
1）如下图，找到 PLC 选项，右键单击，然后选择添加新项：



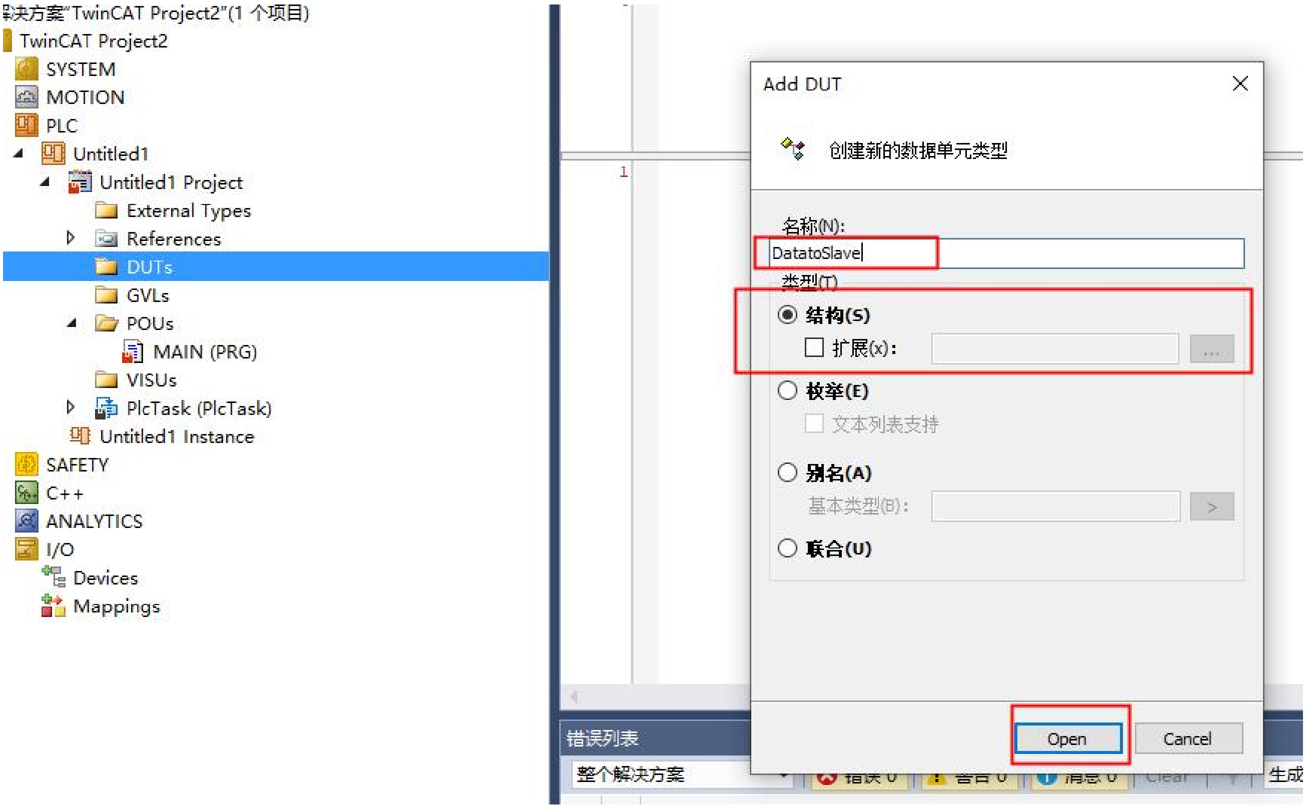
2）选择标准 PLC 项目，输入名称，点击添加：



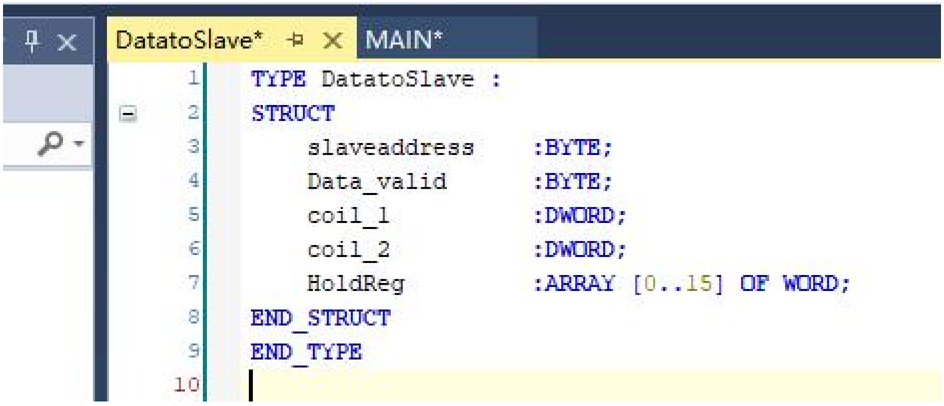
3）找到 DUTs 文件夹，右键单击，天机 DUT 文件，如下图。



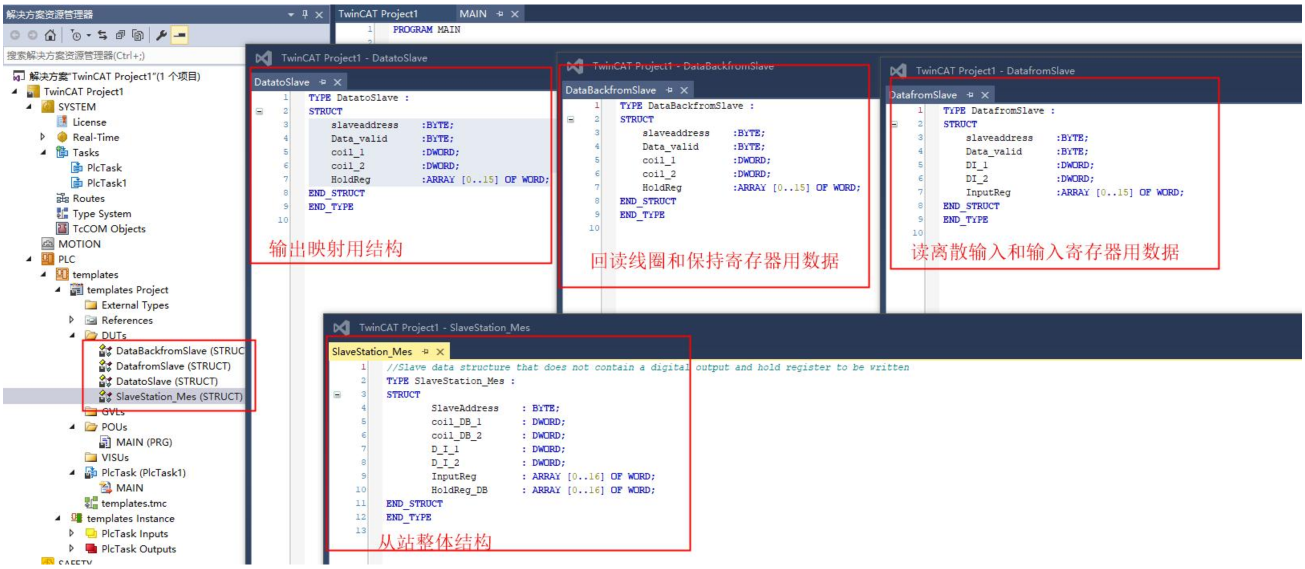
4）输入数据结构体名称“DatatoSlave”,点击打开：



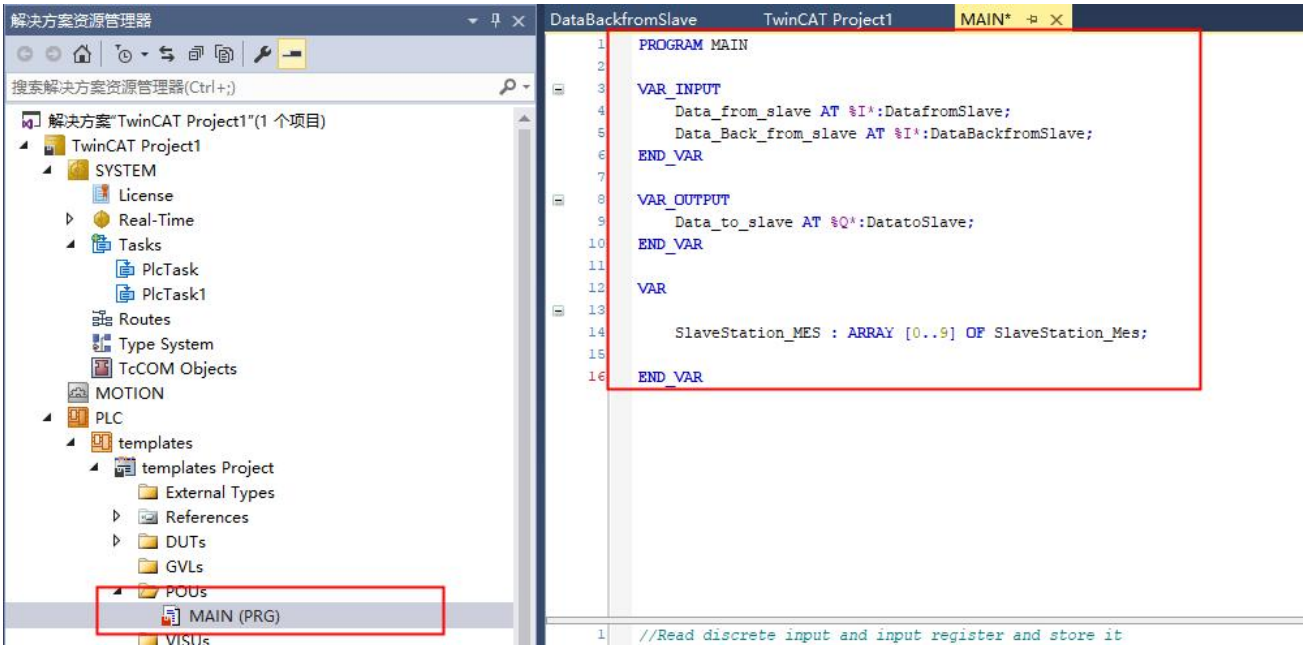
5）输入以下代码：



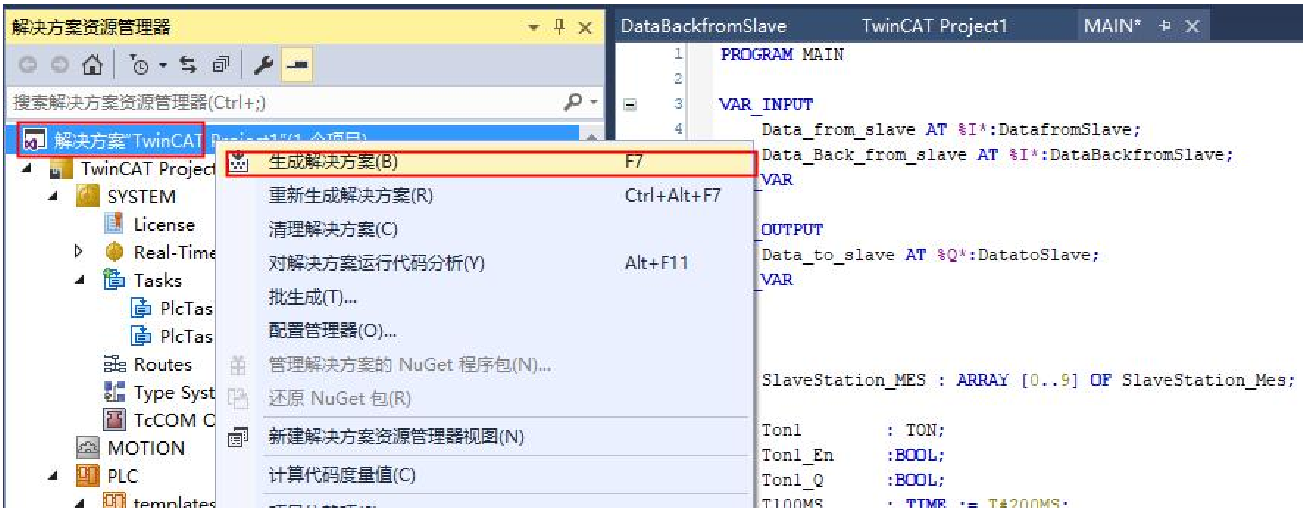
6）按照步骤 3、4、5 新建其他如下图中结构体：



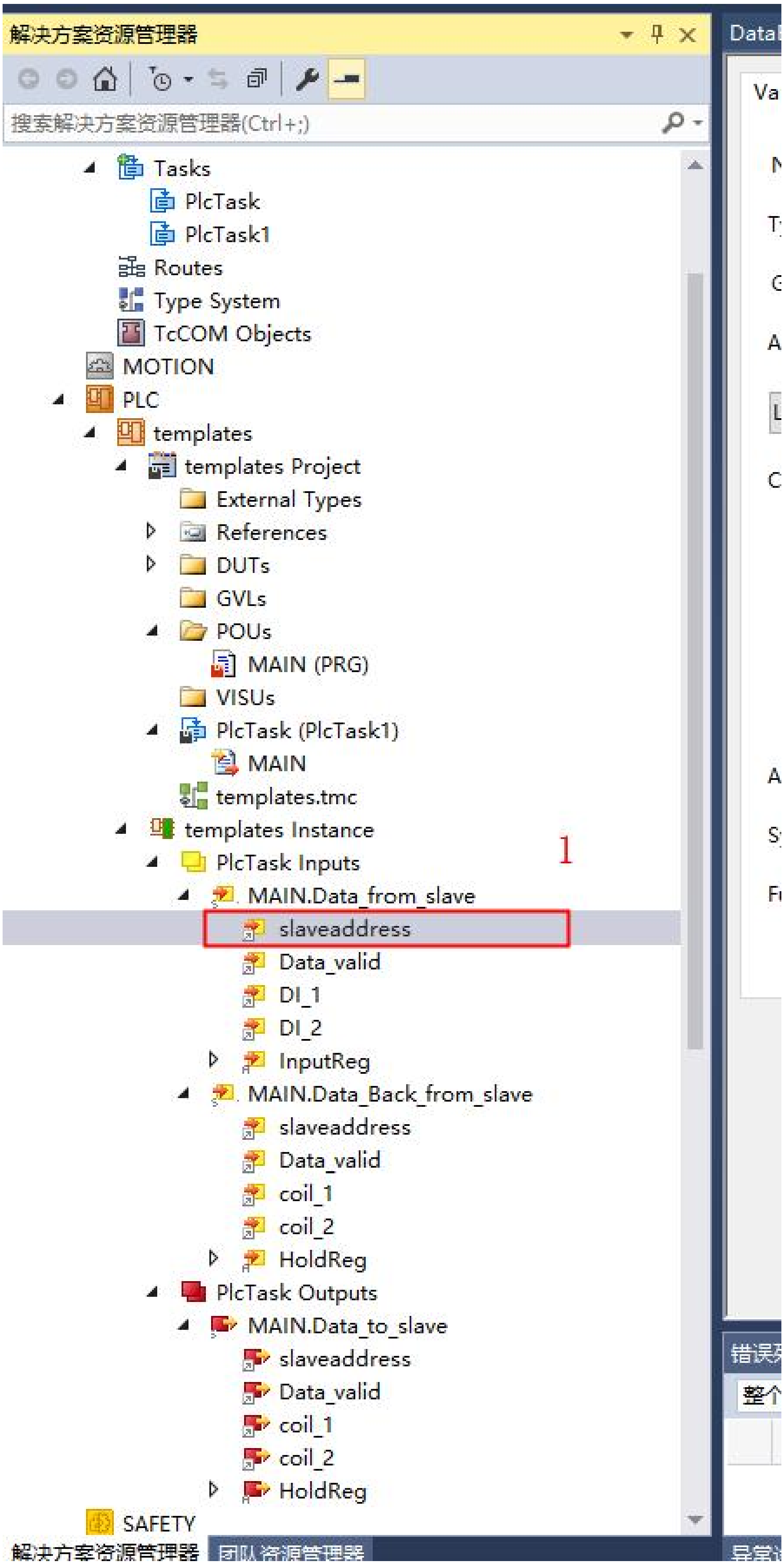
7）找到 MAIN（PRG）文件，双击打开，在声明栏输入以下变量声明，其中“Data_from_slave“用于读输入映射，” Data_Back_from_slave“用于回读输出的映射，” Data_to_slave“用于输出映射，” SlaveStation_MES“保存了 10 个从站的输入和回读数据：



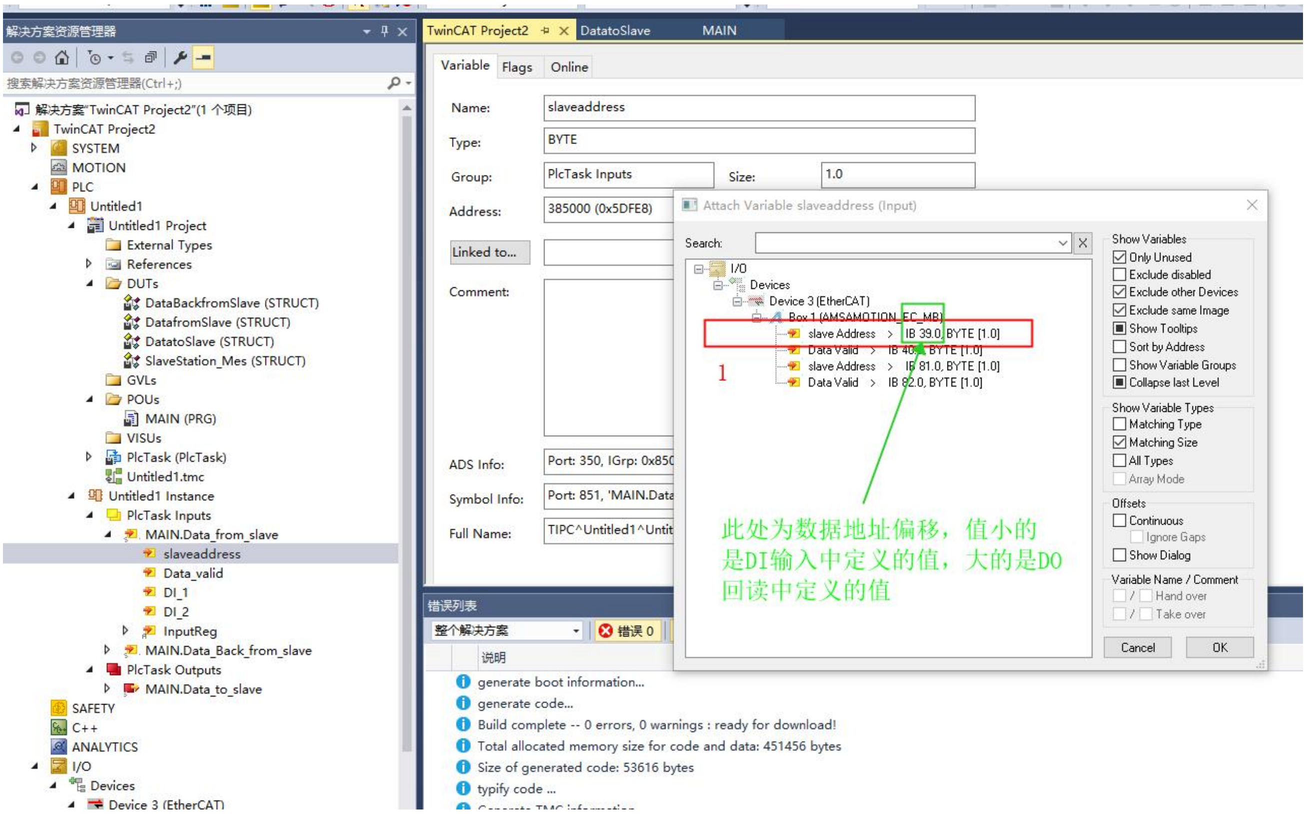
8）按下 F7 或者按下图右键点击“解决方案“后单击”生成解决方案“



9）编译后会生成部分数据，在 PLC 栏目下找到“PLCTask Inputs“和”PLCTask Outputs”并展开，双击图中“1”所示位置。


解决方案资源管理器
搜索解决方案资源管理器(Ctrl+;)
Tasks
PlcTask
PlcTask1
Routes
Type System
TcCOM Objects
MOTION
PLC
templates
templates Project
External Types
References
DUTs
GVLs
POUs
MAIN (PRG)
VISUs
PlcTask (PlcTask1)
MAIN
templates.tmc
templates Instance
PlcTask Inputs
1
MAIN.Data_from_slave
slaveaddress
Data_valid
DI_1
DI_2
InputReg
MAIN.Data_Back_from_slave
slaveaddress
Data_valid
coil_1
coil_2
HoldReg
PlcTask Outputs
MAIN.Data_to_slave
slaveaddress
Data_valid
coil_1
coil_2
HoldReg
SAFETY
解决方案资源管理器
团队资源管理器

10）双击下图所示位置，第 9 步中所示其他变量也是一样设置。



11）如果需要保存回读的每个从站数据，可参考以下代码：

DataBackfromSlave   MAIN

```
PROGRAM MAIN

VAR_INPUT
    Data_from_slave AT %I*:DatafromSlave;
    Data_Back_from_slave AT %I*:DataBackfromSlave;
END_VAR

VAR_OUTPUT
    Data_to_slave AT %Q*:DatatoSlave;
END_VAR

VAR
    SlaveStation_MES : ARRAY [0..9] OF SlaveStation_Mes;

    Ton1        : TON;
    Ton1_En     :BOOL;
    Ton1_Q      :BOOL;
    T100MS      : TIME := T#200MS;

    Data_from_slave_Data_valid :BYTE := 0;
    Data_Back_from_slave_Data_valid :BYTE := 0;

    i: INT := 0;
END_VAR
```

```
//Read discrete input and input register and store it
IF (Data_from_slave.Data_valid <> Data_from_slave_Data_valid) THEN
    SlaveStation_MES[Data_from_slave.slaveaddress-2].SlaveAddress :=  Data_from_slave.slaveaddress;
    SlaveStation_MES[Data_from_slave.slaveaddress-2].D_I_1 := Data_from_slave.DI_1;
    SlaveStation_MES[Data_from_slave.slaveaddress-2].D_I_2 := Data_from_slave.DI_2;

    FOR i:=0 TO 15 BY 1 DO
        SlaveStation_MES[Data_from_slave.slaveaddress-2].InputReg[i] := Data_from_slave.InputReg[i];
    END_FOR
    Data_from_slave_Data_valid := Data_from_slave.Data_valid;
END_IF

//Read back digital output and hold register and store
IF(Data_Back_from_slave.Data_valid <> Data_Back_from_slave_Data_valid)THEN
    SlaveStation_MES[Data_Back_from_slave.slaveaddress-2].SlaveAddress := Data_Back_from_slave.slaveaddress;
    SlaveStation_MES[Data_Back_from_slave.slaveaddress-2].coil_DB_1 := Data_Back_from_slave.coil_1;
    SlaveStation_MES[Data_Back_from_slave.slaveaddress-2].coil_DB_2 := Data_Back_from_slave.coil_2;

    FOR i:=0 TO 15 BY 1 DO
        SlaveStation_MES[Data_Back_from_slave.slaveaddress-2].HoldReg_DB[i] := Data_Back_from_slave.HoldReg[i];
    END_FOR
    Data_Back_from_slave_Data_valid := Data_Back_from_slave.Data_valid;
END_IF

//Running water lamp application
```

12）以下是实现 10 个从站 DO 输出流水灯代码，供参考：

```

//Running water lamp application
Ton1(IN:= Ton1_En, PT:= T100MS, Q=> Ton1_Q, ET=> );

IF Ton1_Q THEN
    Data_to_slave.coil_1 := Data_to_slave.coil_1 * 2 + 1;
    IF (Data_to_slave.coil_1 > 65535) THEN
        Data_to_slave.slaveaddress := Data_to_slave.slaveaddress + 1;
        Data_to_slave.coil_1 := 0;
    END_IF
    IF(Data_to_slave.slaveaddress > 11) OR (Data_to_slave.slaveaddress < 2) THEN
        Data_to_slave.slaveaddress := 2;
    END_IF
    Data_to_slave.Data_valid := Data_to_slave.Data_valid + 1;
    Ton1_En := FALSE;
END_IF

IF NOT(Ton1_Q) AND NOT(Ton1_En) THEN
    Ton1_En :=TRUE;
END_IF

```

五、参数配置说明

本章节针对 RS485/RS422-EC 模块的对象字典进行详细介绍，本章只介绍和模块功能相关对象字，其他不作介绍。

5.1、配置前准备

完成第四章节快速使用中第 4.1 小节添加模块设置。

将 DC 24V 外部电源接入模块并通电，通电前请检查电源正负极是否连接正确

5.2、对象字典

索引	子索引	名称	数据类型	可设置模式	最大/小值	功能
4070	01~14	data error	uint32_t	RO(只读)	--	保存开机后出现的错误代码和出现的时间，单数索引为出现错误的时间，双数索引为错误，错误代码解析见第 6 章节
4071						该对象字保存 17 秒内耗时最大值
	01	Tx time	uint32_t	RO(只读)	--	发送耗时
	02	Wait time	uint32_t	RO(只读)	--	等待接收耗时+接收耗时
	03	Total time	uint32_t	RO(只读)	--	Tx time + Wait time + Modbus 标准间隔等待（3.5T 间隔） + 处理时间
8000	1	Baudrate	enum	Pre-Op	min:1 max:16	枚举类型，用于选择接口波特率，如果枚举中波特率无需要的，可以将其选择为 0x10，此时，使用子索引 3 中定义的值
	2	dataframe	enum	Pre-Op	min:1 max:18	用于选择串口数据格式，第一位是数据长度，第二位是校验，第三位是停止位

	3	explicit baudrate	uint32_t	Pre-Op	--	当 8000：1 设置为 0x10 时，波特率使用本子索引中的值，直接输入十进制数即可，如 115200 对应 115.2kbps 波特率
	4	polling time	uint16_t	Pre-Op	min:50 max:65535	轮询时间，在 modbus 模式下，每一条指令轮询设置的时间，关于轮询时间详细设置方式见第七章节
	5	Slave Reset	bool	Pre-Op	min:0 max:1	将该子索引给 1，见复位索引 8000~800A 为默认值，清除保存的用户参数，该位自复位
	6	Error Reset	bool	Op	min:0 max:1	该子索引给 1，将清除 4070 中保存的错误代码，该位自复位
	7	Device Mode	enum	Pre-Op	min:0 max:1	选择模块工作模式，modbus 模式和透明传输模式
	8	Device Interface	enum	Pre-Op	min:0 max:1	选择模块工作接口，RS422 接口和 RS485 接口
800x（x=1-A)	1	Slave addr	Byte	Pre-Op	min:0 max:255	从站接口地址，1-255，不可为 0，为 0 时从站无效,关闭从站
	2	coil is readable	bool	Pre-Op	min:0 max:1	线圈是否需要回读，如果不需要回读，可关闭回读功能，减少命令条数，以获取更快的循环周期
	3	hold reg readable	bool	Pre-Op	min:0 max:1	保持寄存器是否需要回读，如果不需要回读，可关闭回读功能，减少命令条数，以获取更快的循环周期
	5	coil start address	uint6_t	Pre-Op	min:0 max:65535	从站 modbus 协议中线圈起始地址
	6	the	Byte	Pre-Op	min:0	从站 modbus 协议中线圈

		number of coil			max:64	数量
	8	Discrete input start addr	uint6_t	Pre-Op	min:0 max:65535	从站 modbus 协议中离散输入起始地址
	9	the number of DI	Byte	Pre-Op	min:0 max:64	从站 modbus 协议中离散输入数量
	B	Input register start addr	uint6_t	Pre-Op	min:0 max:65535	从站 modbus 协议中输入寄存器起始地址
	C	the number of Input register	Byte	Pre-Op	min:0 max:16	从站 modbus 协议中输入寄存器数量
	E	Hold register start addr	uint6_t	Pre-Op	min:0 max:65535	从站 modbus 协议中保持寄存器起始地址
	F	the number of Hold register	Byte	Pre-Op	min:0 max:16	从站 modbus 协议中保持寄存器数量

Note：在单一对象字典无法将某一个从站数据读取完毕时，可以使用多个对象字典读取同一个从站中的值，将寄存器和线圈的起始地址进行相应偏移即可。

六、错误

6.2、错误处理

1）从站错误

当出现从站错误，除 0xFF00000F 外，其他均可以先检查从站的状态是否正常，以及本模块的对象字 8001~800A 设置是否合理。

当出现从站错误且主站未停止运行时，0xFF00000F 代码一定会出现。但是，当单独出现 0xFF00000F 时，没有从站错误，请参考第七章内容。

2）主站错误

当出现 0xFF000010 时，先检查 PLC 代码，是否有写从站相应数据，且从站地址未配置到本模块对象字 8001~800A 的子索引 1 中。

当出现 0xFF000011 时，检查是否未配置到本模块对象字 8001~800A 或者设置有误，比如未设置保持寄存器数量，但是 PLC 代码在写相应地址的保持寄存器。

当出现 0xFF000014 时，表明在透明传输模式下，模块接收到的一包长度大于了 128 字节，出现丢包现像。

3）重启

本模块设置有物理 Reset 按钮，当按下该按钮超过三秒时，不管在任何状态，均会见用户数据清除复位成出厂值，此时模块应该需要重新进入 Pre-OP 状态进行初始化，合适的办法是断电 3 秒然后重启模块，如果无法断电，可参考以下两种操作：

1、可将模块运行状态切换回 Pre-OP，然后使用对象字 8000：6 清除故障，然后重新设置对象字 8000~800A，完成后再将运行模式切换回 OP，开始运行。

2、不使用物理 Reset 按钮，直接将模块运行模式切换成 Pre-Op 模式，将对象字 8000：5 置 1，然后重新设置对象字 8000~800A，完成后再将运行模式切换回 OP，开始运行。

6.1、模块状态

本模块状态分为四类，等级从低到高分别为正常、从站错误（警告）、主站错误（错误 1）、重启（错误 2）。

不同等级的状态可由 RUN 灯闪烁进行提示，且在对象字 4070 中可查看错误状态。下

表是状态和对应的出现原因。

<table>
<tr><td>状态</td><td>4070 对象字</td><td>RUN 灯状态</td><td>原因</td></tr>
<tr><td>正常</td><td>0x00000000</td><td>2 秒周期闪烁</td><td>--</td></tr>
<tr><td rowspan="8">从站错误</td><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x01</td><td rowspan="8">1 秒周期闪烁</td><td>不合法功能码</td></tr>
<tr><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x02</td><td>非法的数据地址</td></tr>
<tr><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x03</td><td>非法的数据值或范围</td></tr>
<tr><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x04</td><td>从站设备故障</td></tr>
<tr><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x0D</td><td>接收超时（超过当前波特率接收 300 个字节的时间未接收到返回数据）</td></tr>
<tr><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x0E</td><td>接收 CRC 错误，但数据长度>5</td></tr>
<tr><td>0x00<<24|(从站索引<<16)|（功能码<<8）|0x12</td><td>接收 CRC 错误，且数据长度<5</td></tr>
<tr><td>0xFF00000F</td><td>发送下一条命令时，上一条指令未执行完成</td></tr>
<tr><td rowspan="3">主站错误</td><td>0xFF000010</td><td rowspan="3">亮 0.5 秒、灭 0.5 秒、亮 0.5 秒、灭 2 秒</td><td>错误的写从站地址，未设置该从站</td></tr>
<tr><td>0xFF000011</td><td>选择的从站无法使用写功能，或者进入 OP 模式，但未设置任何从站信息</td></tr>
<tr><td>0xFF000014</td><td>在透传模式下模块接收到的串口消息过快，出现丢包现象</td></tr>
<tr><td>重启</td><td>--</td><td>亮 0.2 秒、灭 0.2 秒、亮 0.2 秒、灭 1 秒</td><td>使用了物理 Reset 按钮复位</td></tr>
</table>

6.3、模块 LED 灯状态

本模块在第 2.2 章节介绍中，有五中 LED 灯，本章节详细介绍各种灯的状态。

1）RUN 灯

在模块正常工作时以 2 秒的周期闪烁；

在模块出现从站错误时以 1 秒周期闪烁；

在模块出现主站错误时（错误 1），会以“亮 0.5 秒、灭 0.5 秒、亮 0.5 秒、灭 2 秒”的周期进行双闪；

在模块需要重启时（错误 2），会以“亮 0.2 秒、灭 0.2 秒、亮 0.2 秒、灭 1 秒”的周期进行双闪。

2)EC RUN 灯和 EC ERR 灯

EC RUN 灯状态	EC ERR 灯状态	模块状态
灭	灭	未连接 EtherCAT 主站
		连接 EtherCAT 主站但处于 Init 状态
0.2 秒周期闪烁	灭	模块处于 Pre-Op 状态
0.5 秒周期闪烁	灭	模块处于 SAFEOP 状态
常亮	灭	模块处于 OP 状态
灭	0.2 秒周期闪烁	EtherCAT 通讯出现错误

3)RS485 灯和 RS422 灯

RS485 灯	RS422 灯	模块状态
1 秒周期闪烁	灭	RS485 接口工作且在 Modbus 模式
0.5 秒周期闪烁	灭	RS485 接口工作且在透传模式
灭	1 秒周期闪烁	RS422 接口工作且在 Modbus 模式
灭	0.5 秒周期闪烁	RS422 接口工作且在透传模式
灭	灭	模块未处于 OP 状态

七、模块轮询时间

在第五章对象字典中有关于轮询时间的设定，具体对象字为 0x8000:4，单位是 ms。该参数设置不当时，即使主站模块未报错，也会出现 0xFF00000F 的错误。

设置时如果不能知道设置多少合适，可先使用默认值运行，在运行时，模块会统计出和从站的通讯时间相关参数，并保存在对象字 4071 中，具体对应见下面说明。

该对象字用于设置模块发送一条指令到发送下一条指令的间隔，设置和应用时应该注意以下几点问题：

1)发送间隔 = 指令最大长度（字节数） * 当前波特率每 ms 发送的字节数 + 接收最大长度（字节数） * 当前波特率每 ms 发送的字节数 + 当前波特率每字节发送消耗的时间 * 3.5（向上取整 ms 数） + 从站响应时间。

部分时间计算在对象字 4071 中已经进行粗略计算，可以进行参考对应关系如下：

4071：3　Total Time：发送间隔

4071：2　Tx Time：指令最大长度（字节数） * 当前波特率每 ms 发送的字节数

4071：1　Wait Time ：接收最大长度（字节数） * 当前波特率每 ms 发送的字节数 + 当前波特率每字节发送消耗的时间 * 3.5（向上取整 ms 数） + 从站响应时间

1) PLC 发送数据的间隔应该大于该对象字设置时间的两倍，否则模块将一直发送写命令，回读命令将暂停，无法得到执行时间；

2) 在使用读取到的状态时，需要注意每个状态更新的时间 = 每个从站命令条数 * 从站总数 * 轮询时间（0x8000:4）,每个从站命令条数由对象字 8001~800A 设置生成。比如如下设置：

索引	子索引	名称	设定值	索引	子索引	名称	设定值
8001	1	Slave addr	2	8004	1	Slave addr	4
	2	coil is readable	1		2	coil is readable	0
	3	hold reg readable	1		3	hold reg readable	0
	5	coil start address	0		5	coil start address	0
	6	the number of coil	64		6	the number of coil	64
	8	Discrete input start addr	0		8	Discrete input start addr	0

	9	the number of DI	64		9	the number of DI	64
	B	Input register start addr	0		B	Input register start addr	0
	C	the number of Input register	16		C	the number of Input register	16
	E	Hold register start addr	0		E	Hold register start addr	0
	F	the number of Hold register	16		F	the number of Hold register	16
8002	1	Slave addr	3	8005	1	Slave addr	5
	2	coil is readable	1		2	coil is readable	1
	3	hold reg readable	1		3	hold reg readable	0
	5	coil start address	0		5	coil start address	0
	6	the number of coil	16		6	the number of coil	64
	8	Discrete input start addr	0		8	Discrete input start addr	0
	9	the number of DI	16		9	the number of DI	64
	B	Input register start addr	0		B	Input register start addr	0
	C	the number of Input register	0		C	the number of Input register	16
	E	Hold register start addr	0		E	Hold register start addr	0
	F	the number of Hold register	0		F	the number of Hold register	16
8003	1	Slave addr	0	8006	1	Slave addr	5
	2	coil is readable	1		2	coil is readable	0
	3	hold reg readable	1		3	hold reg readable	1
	5	coil start address	0		5	coil start address	64
	6	the number of coil	16		6	the number of coil	64
	8	Discrete input start addr	0		8	Discrete input start addr	64
	9	the number of DI	16		9	the number of DI	64
	B	Input register start addr	0		B	Input register start addr	16
	C	the number of Input register	0		C	the number of Input register	16
	E	Hold register start addr	0		E	Hold register start addr	16
	F	the number of Hold register	0		F	the number of Hold register	16

在上图示例中，一共设置 5 个从站，对象字 8003：1 从站地址为空，即使设置从站属

性也不计算。

从站 2 使用 64 个线圈输出、64 个离散输入、16 个保持寄存器、16 个输入寄存器，同时支持线圈回读和保持寄存器回读。这是本模块支持的最大数量，在模块运行时，会有 4 条 modbus 协议指令（读线圈、读离散输入、读输入寄存器、读保持寄存器），如果 PLC 编程有按周期写本模块的线圈和保持寄存器，一共有 6 条 modbus 协议指令（加入写线圈、写保持寄存器）。

从站 3 使用 16 个线圈输出、16 个离散输入，同时支持线圈回读和保持寄存器回读，但是保持寄存器数量为零，回读指令无效。在模块运行时，一共 2 条 modbus 协议指令（读线圈、读离散输入），如果 PLC 编程有按周期写本模块的线圈，一共有 3 条 modbus 协议指令（加入写线圈）。

从站 4 使用 64 个线圈输出、64 个离散输入、16 个保持寄存器、16 个输入寄存器，不支持线圈回读和保持寄存器回读。在模块运行时，会有 2 条 modbus 协议指令（读离散输入、读输入寄存器），如果 PLC 编程有按周期写本模块的线圈和保持寄存器，一共有 4 条 modbus 协议指令（加入写线圈、写保持寄存器）。

从站 5 使用 128 个线圈输出、128 个离散输入、32 个保持寄存器、32 个输入寄存器，支持保持寄存器回读。共占用两个对象字（8005、8006）。在模块运行时，会有 6 条 modbus 协议指令（读离散输入、读输入寄存器、读保持寄存器），如果 PLC 编程有按周期写本模块的线圈和保持寄存器，一共有 10 条 modbus 协议指令（加入写线圈、写保持寄存器）。

综上本次一共使用 14 条读命令，9 条写命令，读命令按照对象字 8000：4 设定时间进行轮询，写命令由 PLC 编程随机插入，当 PLC 需要进行写命令时，本模块在执行完毕上一条读命令后，优先执行写命令。

此时，如果轮询时间（8000：4）设置为 50ms，即使没有任何的写指令，在读到某一个从站状态后，至少需要 50ms*14 = 700ms 才会再次更新这个状态，即假如读取从站 2 离散输入状态后，下一次更新此状态将在 700ms 后，再次期间模块不读此从站离散输入状态。所以在实际使用时，使用 Modbus 功能时，尽量只周期读取需要的从站变量，减少轮询时间和命令条数。

修订历史

版本	修订日期	修订说明	维护人
1.0	2022.10.25	初始版本	Zhang

关于我们

企业名称：东莞市艾莫迅自动化科技有限公司

官方网站：www.amsamotion.com

技术服务：4001-522-518拨1

企业邮箱：sale@amsamotion.com

公司地址：广东省东莞市南城区袁屋边艺展路9号兆炫智造园 B 栋

官方公众号

官方抖音号