

GigaDevice Semiconductor Inc.

EtherCAT 协议栈代码移植参考手册

应用笔记

AN246

1.2 版本

（2025 年 9 月）

目录

目录.....	2
图索引	3
表索引	4
1. 前言	5
2. 使用 SSC TOOL 生成协议栈代码	6
3. 协议栈代码编入工程	11
4. EEPROM 更新方法.....	11
5. COE 添加 SDO & PDO 方法.....	15
5.1 添加 SDO	16
5.2 添加 PDO	19
6. 版本历史	26

图索引

图 2-1. SSC Tool Options 页面	6
图 2-2. Configurations 页面	7
图 2-3. 创建工程页面	7
图 2-4. 选择 GD 工程页面	8
图 2-5. 保存路径页面	8
图 2-6. EXCEL 文件页面	9
图 2-7. 导入页面	9
图 2-8. 生成协议栈代码页面	10
图 2-9. 保存路径页面	10
图 3-1. 源文件目录页面	11
图 3-2. 编译工程页面	11

表索引

表 4-1. 版本历史	26
-------------------	----

1. 前言

本文是专为使用 GD32 MCU 产品，需要使用 EtherCAT 协议栈代码移植做简要说明移植步骤和移植方法。

本应用笔记分为两个部分描述第一部分描述使用 SSC TOOL 工具生成协议栈代码过程，第二部分说明协议栈代码嵌入工程编译过程。

在使用本应用前，需要使用者已成为 ETG 组织会员，可以获取到 SSC TOOL 工具用于生成协议栈代码。

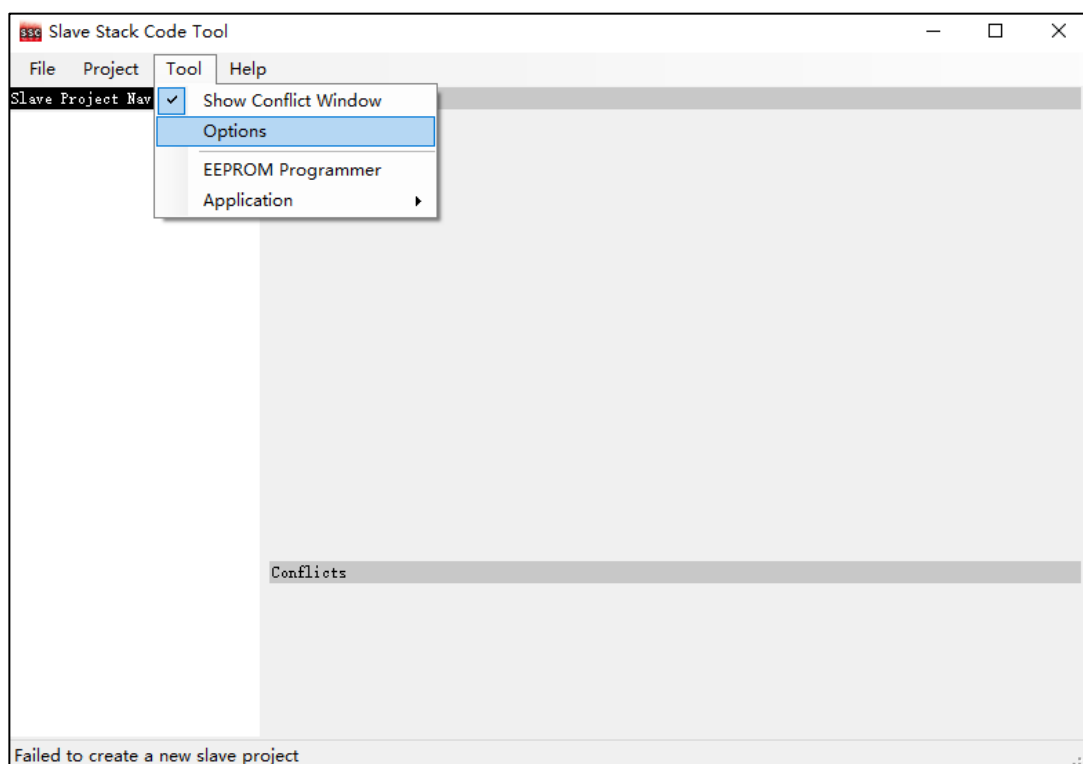
2. 使用 SSC TOOL 生成协议栈代码

使用者首先已完成 SSC TOOL 工具下载，下载方法参考 ETG 官方：
https://www.ethercat.org/en/downloads/downloads_01DCC32A10294F2EA866F7E46FB0285F.htm。

参考以下步骤生成协议栈代码

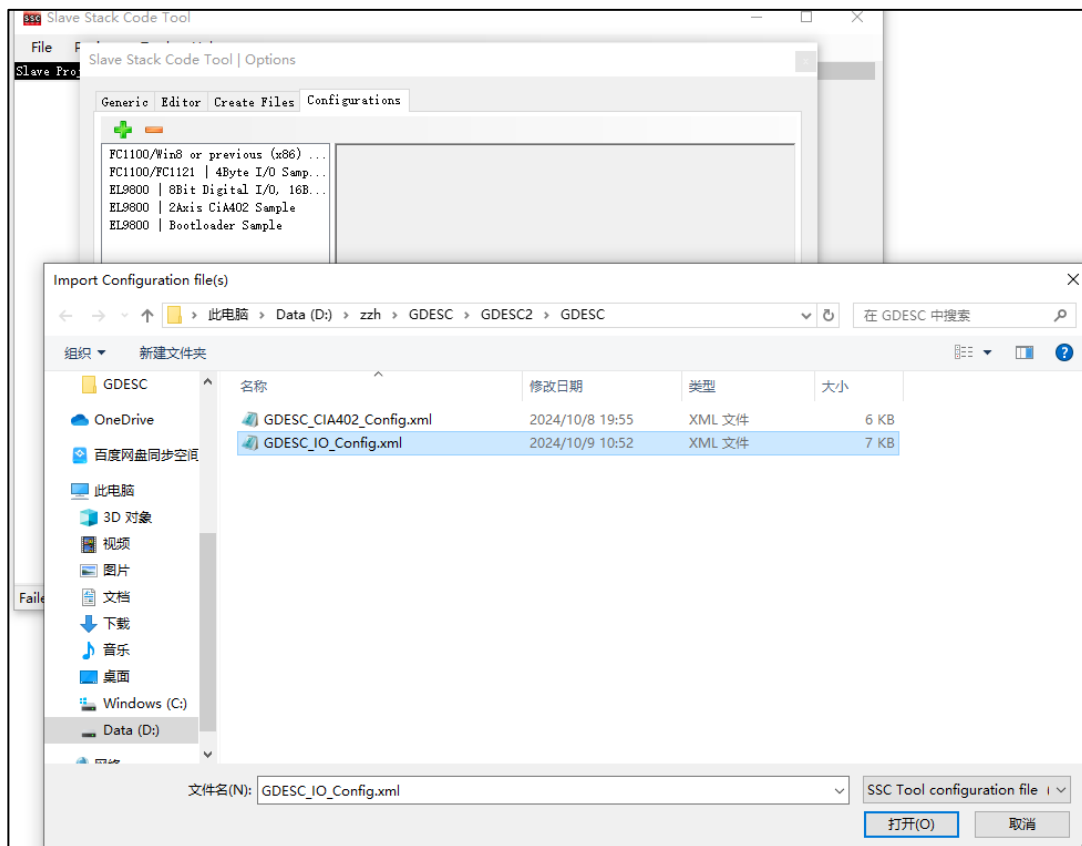
1. 打开 SSC Tool, 选择**Tool->Options**

图 2-1. SSC Tool Options 页面



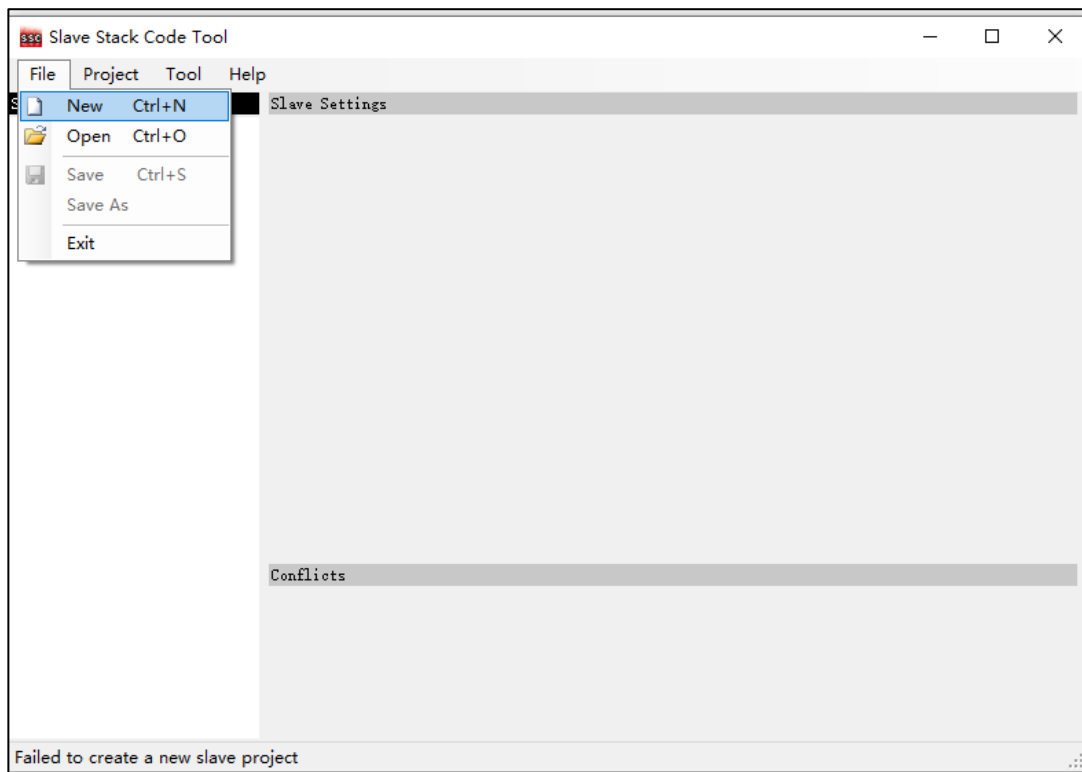
2. 点击 Configurations, 选择添加配置文件 GDESC_XXX_Config.xml

图 2-2. Configurations 页面



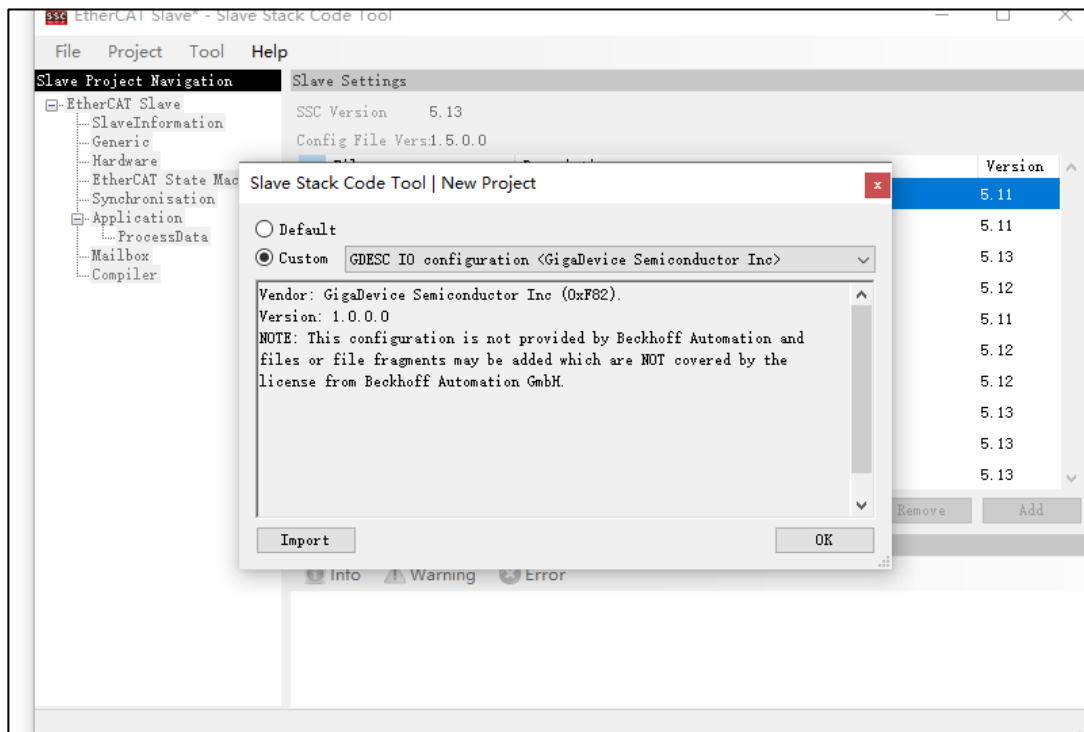
3. SSC Tool 中创建新的工程, 点击**File->New**

图 2-3. 创建工程页面



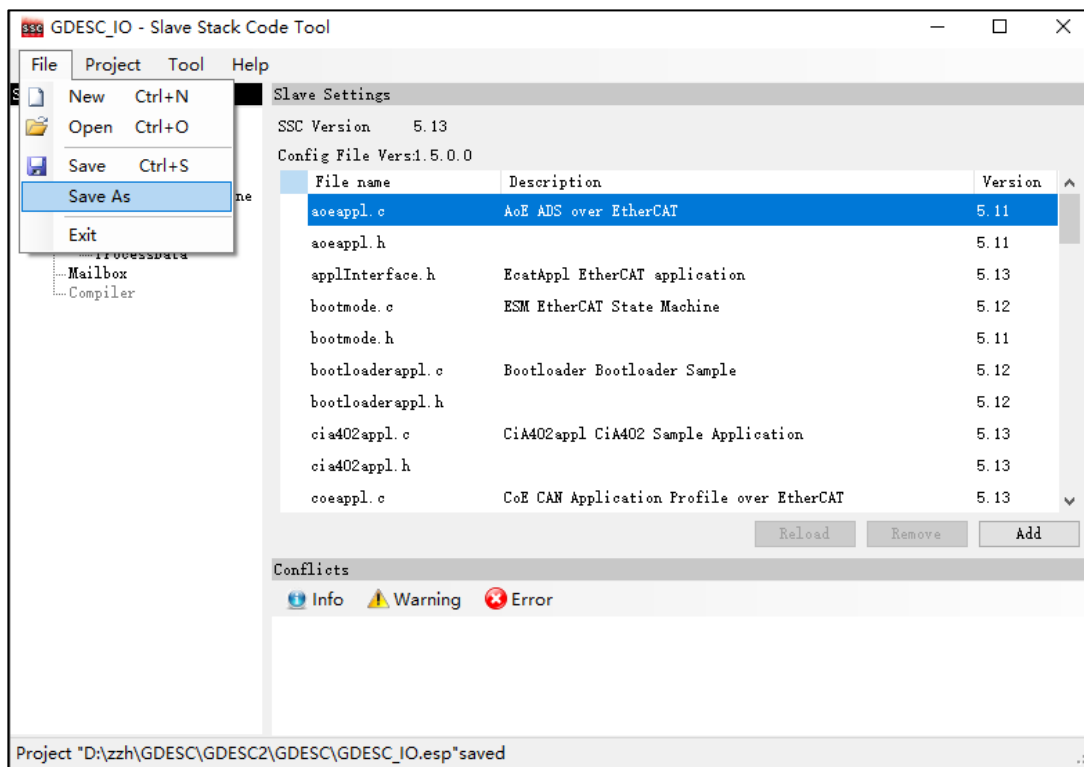
4. 点击****Custom****选项，并在下拉列表中选择****GDESC IO configuration <GigaDevice Semiconductor Inc >****

图 2-4. 选择 GD 工程页面



5. 保存工程文件，指定保存路径

图 2-5. 保存路径页面



6. 导入应用 EXCEL 文件，若导入的为 GDESC_CIA402_Config_Demo.xml 则忽略此步骤

图 2-6. EXCEL 文件选择页面

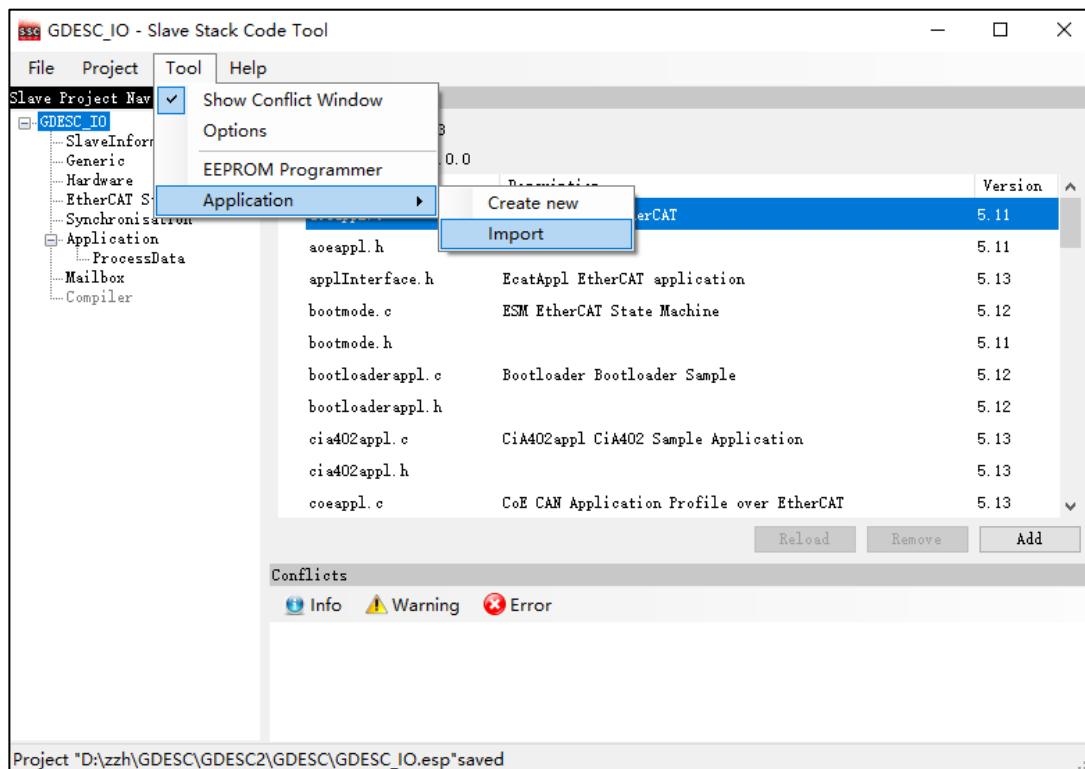
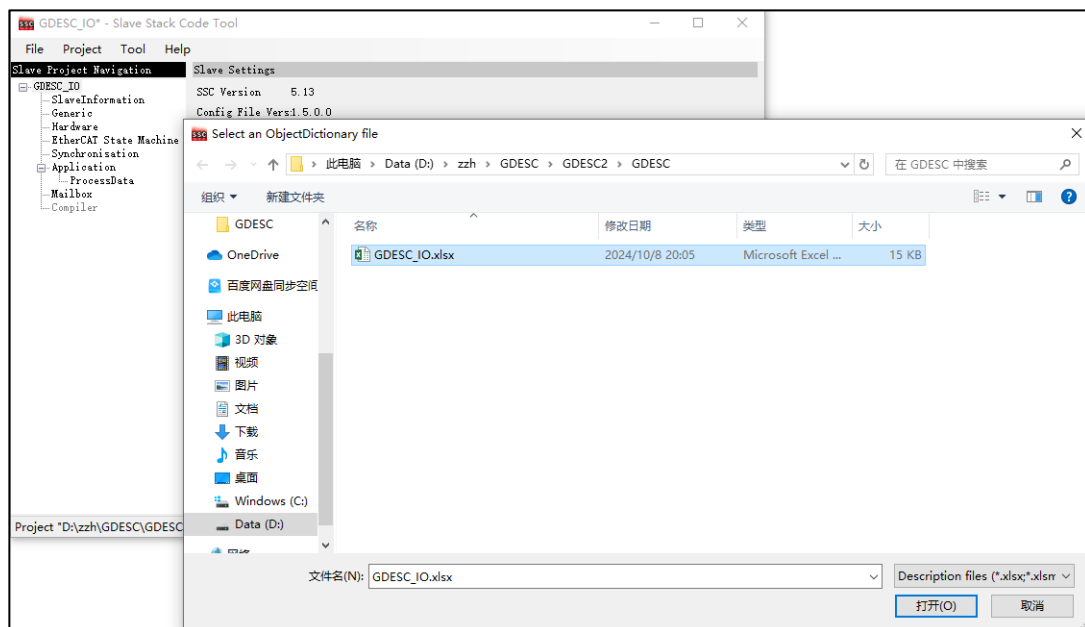
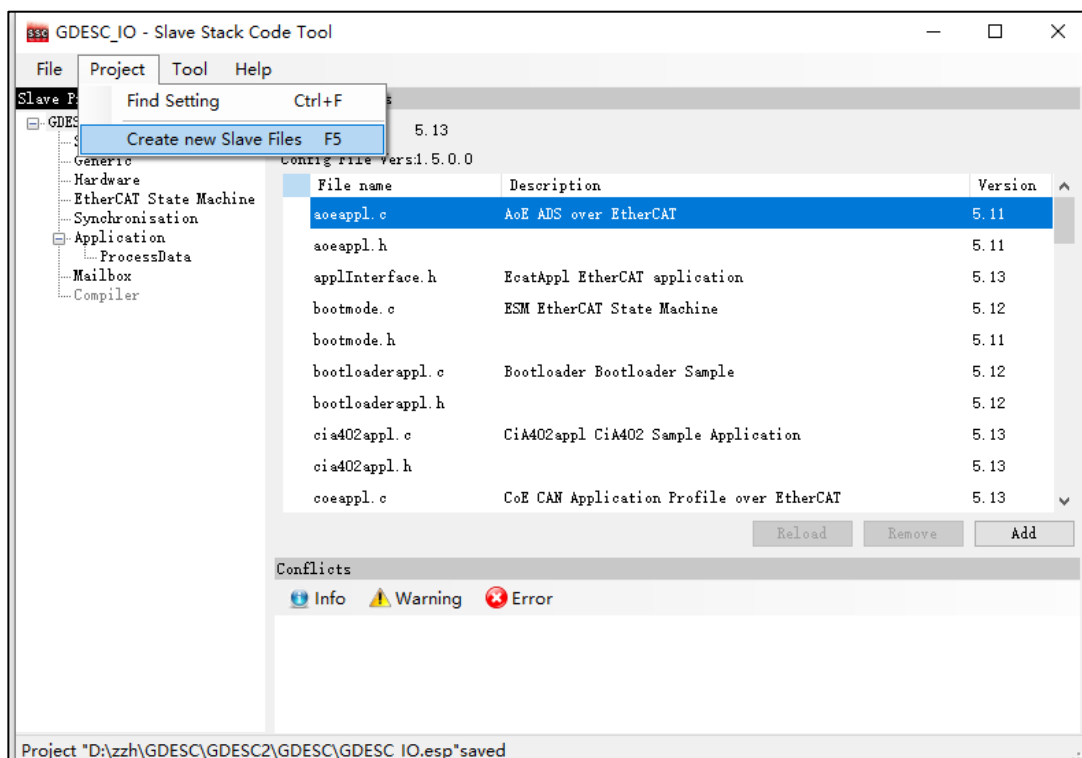


图 2-7. 导入页面



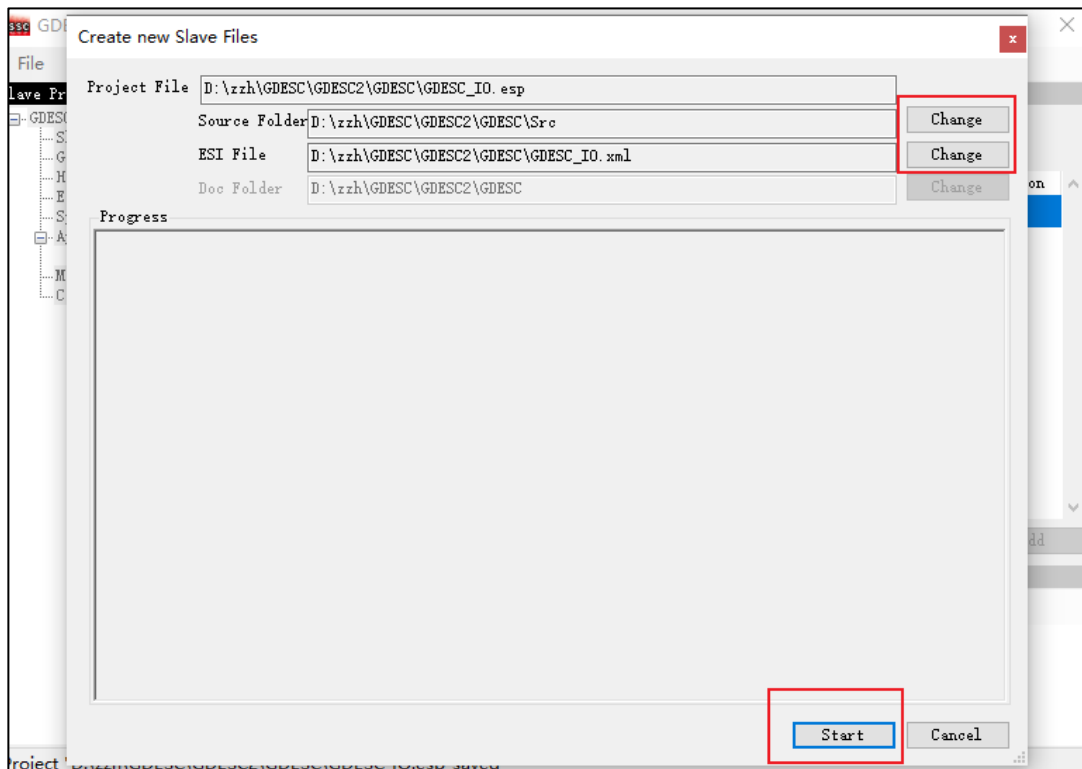
7. 生成协议栈代码

图 2-8. 生成协议栈代码页面



8. 选择路径，保存协议栈代码和配置 xml 文件

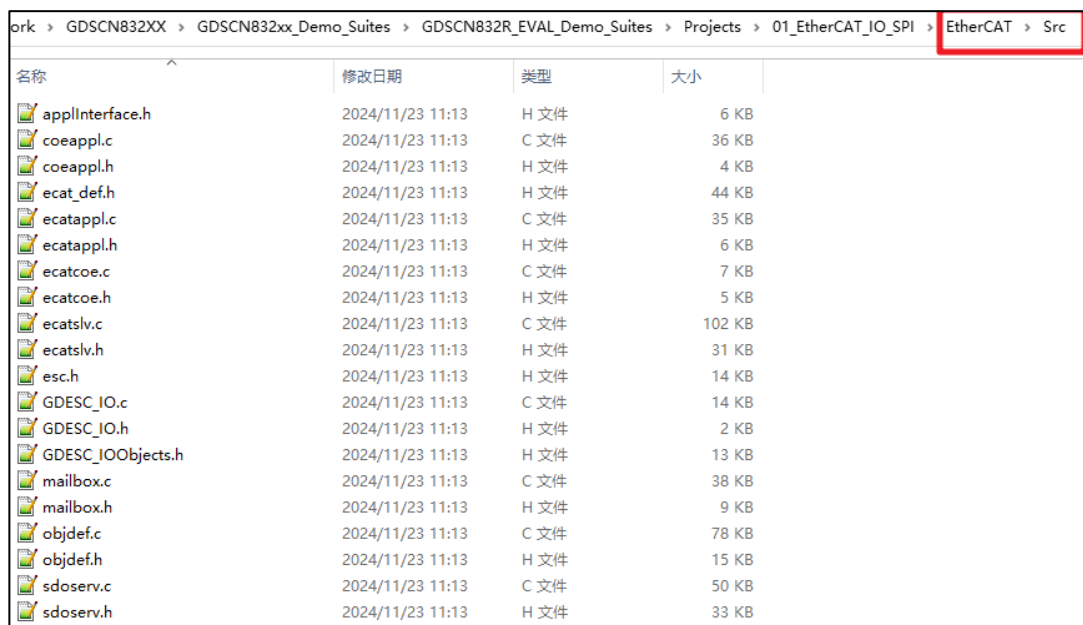
图 2-9. 保存路径页面



3. 协议栈代码编入工程

1. 拷贝生成的协议栈代码到工程目录下的 EtherCAT/Src 目录

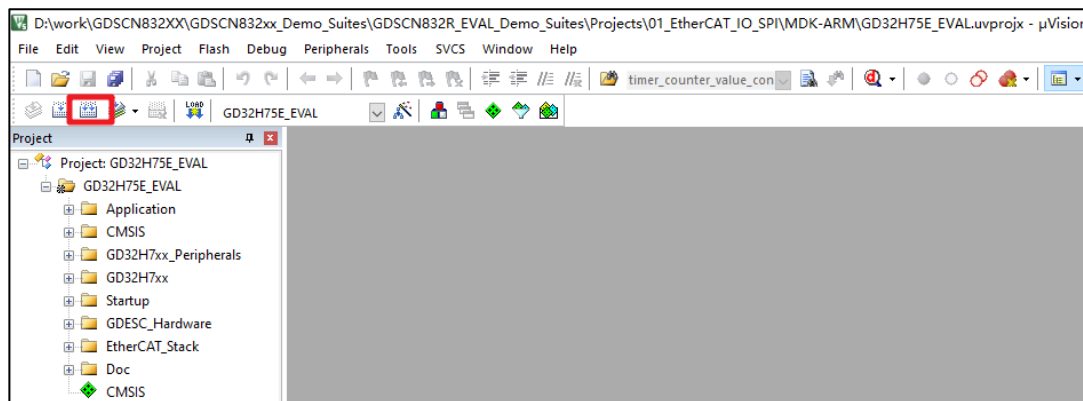
图 3-1. 源文件目录页面



名称	修改日期	类型	大小
applInterface.h	2024/11/23 11:13	H 文件	6 KB
coeappl.c	2024/11/23 11:13	C 文件	36 KB
coeappl.h	2024/11/23 11:13	H 文件	4 KB
ecat_def.h	2024/11/23 11:13	H 文件	44 KB
ecatappl.c	2024/11/23 11:13	C 文件	35 KB
ecatappl.h	2024/11/23 11:13	H 文件	6 KB
ecatcoe.c	2024/11/23 11:13	C 文件	7 KB
ecatcoe.h	2024/11/23 11:13	H 文件	5 KB
ecatslv.c	2024/11/23 11:13	C 文件	102 KB
ecatslv.h	2024/11/23 11:13	H 文件	31 KB
esc.h	2024/11/23 11:13	H 文件	14 KB
GDESC_IO.c	2024/11/23 11:13	C 文件	14 KB
GDESC_IO.h	2024/11/23 11:13	H 文件	2 KB
GDESC_IOObjects.h	2024/11/23 11:13	H 文件	13 KB
mailbox.c	2024/11/23 11:13	C 文件	38 KB
mailbox.h	2024/11/23 11:13	H 文件	9 KB
objdef.c	2024/11/23 11:13	C 文件	78 KB
objdef.h	2024/11/23 11:13	H 文件	15 KB
sdoserv.c	2024/11/23 11:13	C 文件	50 KB
sdoserv.h	2024/11/23 11:13	H 文件	33 KB

2. 直接编译工程文件

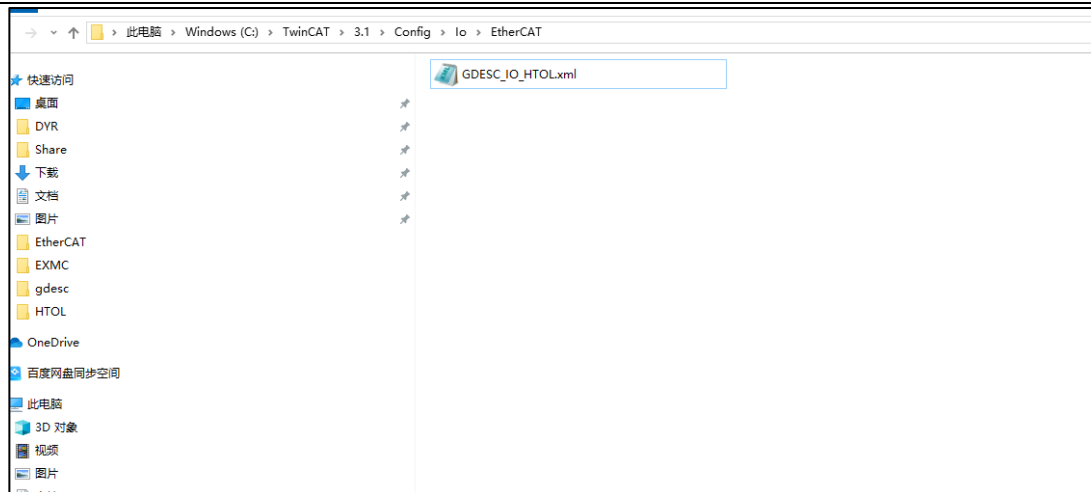
图 3-2. 编译工程页面



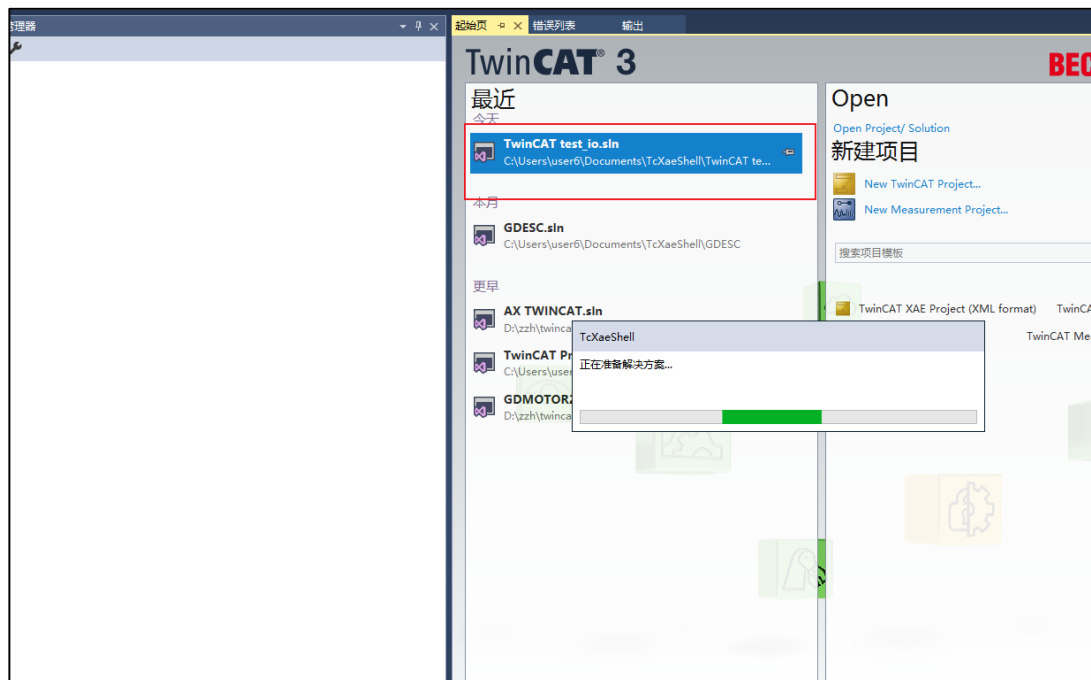
4. EEPROM 更新方法

下面将使用 Twincat 主站软件举例如何更新 EEPROM 的方法步骤：

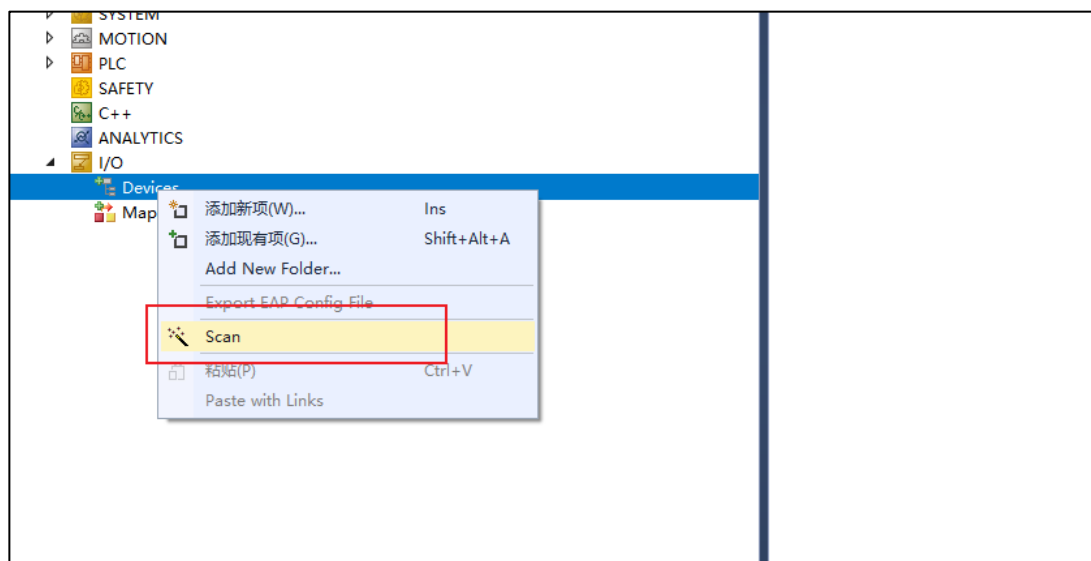
1. 将需要更新的 XML 文件拷贝到 Twincat 软件制定目录下，如 XXX\TwinCAT\3.1\Config\I o\EtherCAT 目录下



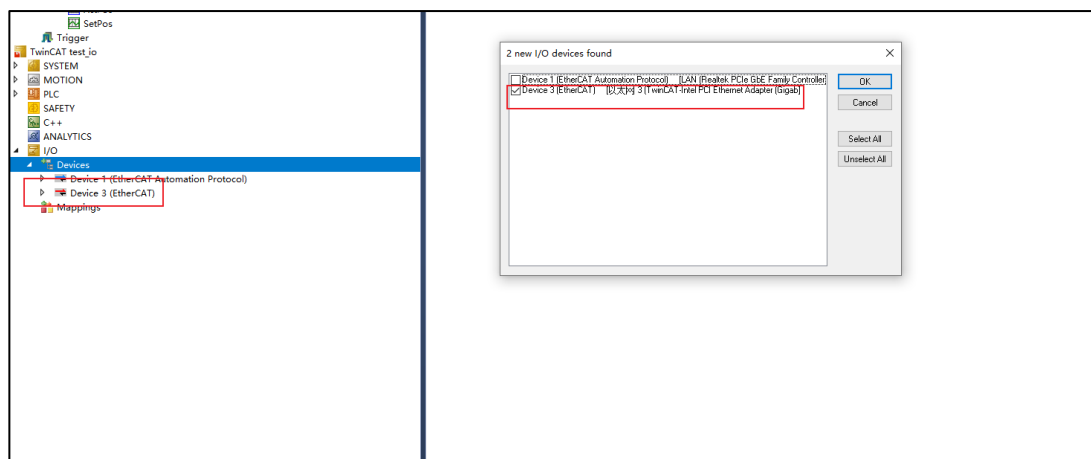
2. 打开 TwinCAT 软件，点击打开任意工程文件



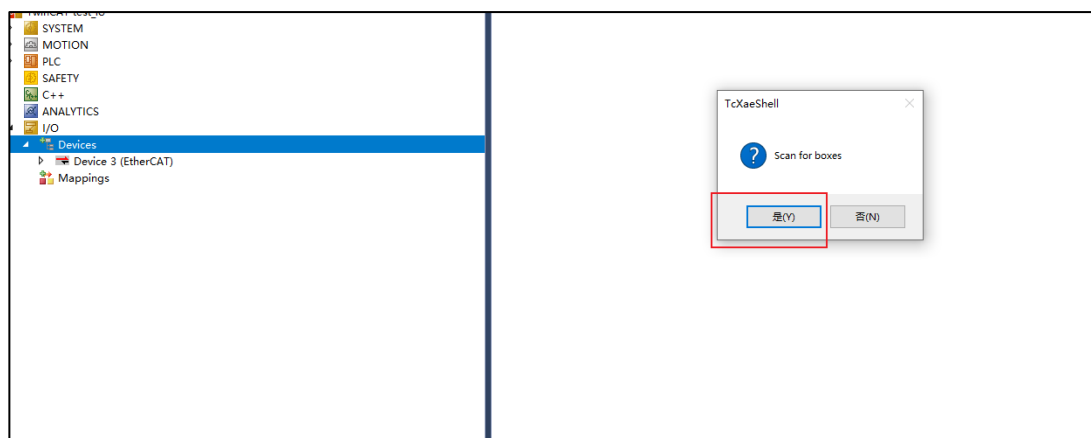
3. 打开工程后，点击 Device，右击选择 Scan

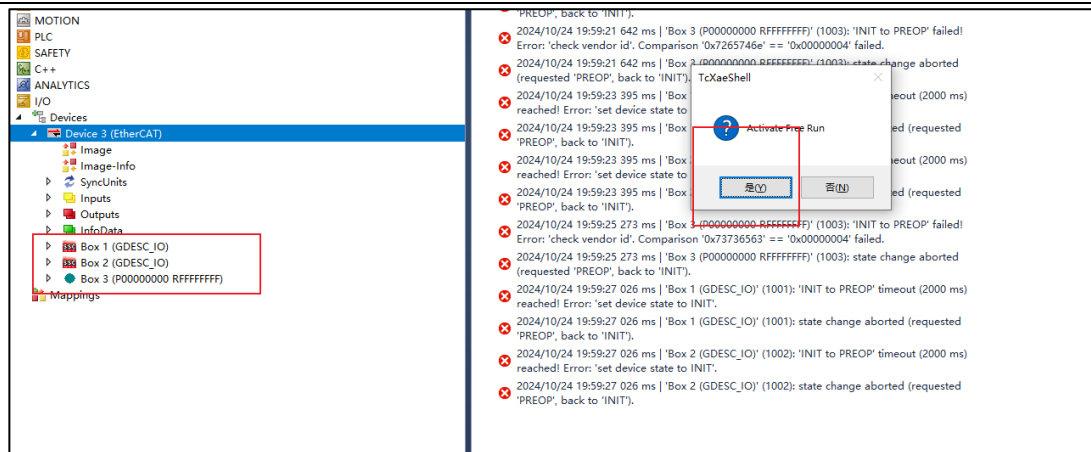


4. 选择对应的网卡连接，正常识别到从站后，勾选对应的网卡，点击 OK

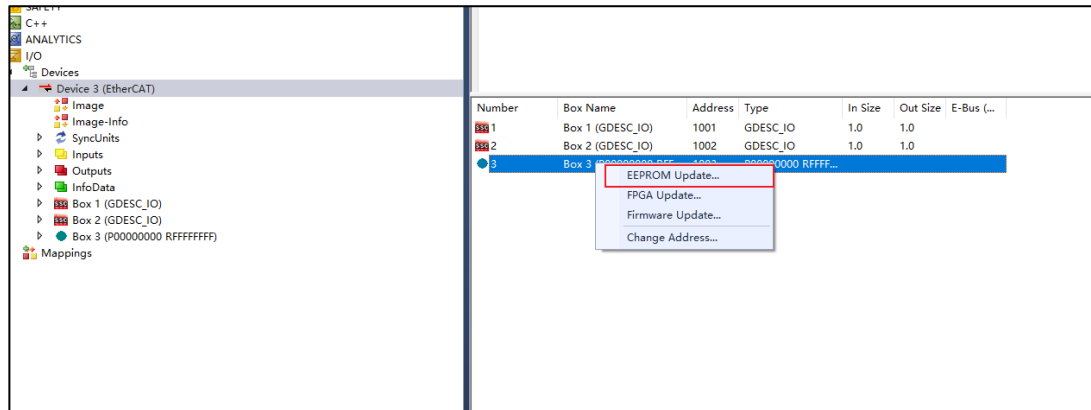


5. 依次点击“是”直至扫描结束





- 扫描结束后，在单击两次 Device，右侧出现对应的 Box，右击 BOX 选择 EEPROM 进行更新



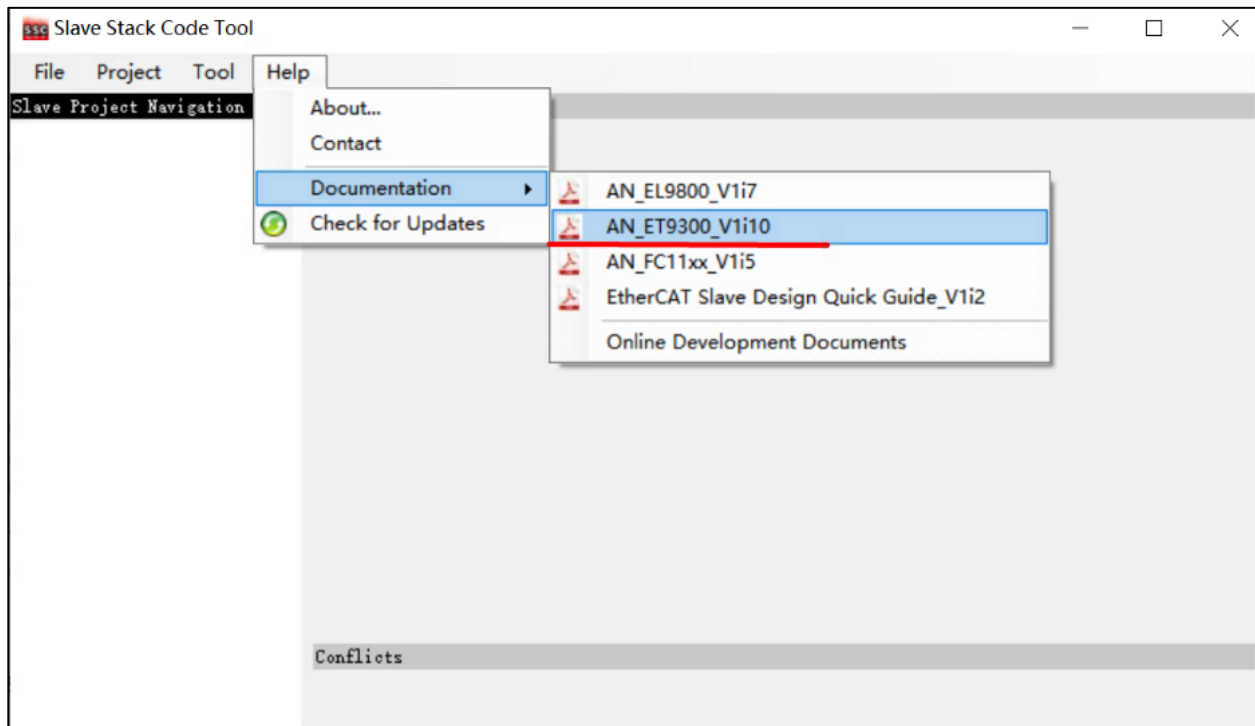
5. COE 添加 SDO & PDO 方法

通常我们可以选择两种方式增加 CoE 对象字典内容：

一种是基于 SSC Tool 中的 OD 工具，直接创建新的对象字典；

一种是基于现有的例程，通过修改从站设备 xml 文件和修改从站设备程序，实现添加新的对象字典内容。

《AN_ET9300》是 SSC Tool 安装后的帮助文档。该文档是关于 SSC Tool 的使用介绍，不仅有助于学习使用 SSC Tool，还有助于理解从站协议栈代码处理流程，在指导如何添加 CoE 字典内容方面也非常有帮助。



第一种方法在对于对象字典使用不是很多，映射比较简单的应用程序，使用 SSC TOOL 添加对象字典，需要按照实际应用修改 GD 提供的 DEMO 例程中(GDESC_IO.xlsx)的 EXCEL 文件，再将文件重新导入 SSC TOOL 中生成新的协议栈代码（可参考本文 2.6 中的步骤），即可添加对应的对象字典参数。在添加对应的对象字典时需要注意参数命名的高字节和低字节的排列顺序，确保参数命名符合规范。

参考《AN_ET9300》中的 6.4 Create an own Application 和 13 OD Tool 章节，编辑 excel，添加需要的对象字典内容。

Device Profile:

5001

Modular Device Profile

Modul Profile:

0

IndexIncrement:

0

PdoIndexIncrement:

0

Usage Notes:

The PDO mapping object and SynManager assignment object doesn't need to be defined. In that case they are created automatically.

The following objects are fixed included in the SSC and shall not be defined in the file : 0x1000, 0x1001, 0x1008, 0x1009, 0x100a, 0x1010, 0x1011, 0x1018, 0x10F0, 0x10F1, 0x10F3, 0x1c32, 0x1c33

Entries less or equal one SBit shall not overlap byte borders.

Entries greater SBit shall always start at an exact word border

The object dictionary defined here shall be used complementary with ETC.5001 and ETC.1000

Index	ObjectCode	SI	Data Type	Name	Default V.	M/O/C	B/S	Access	rx/tx	CoeRead	CoeWrite	Description
//0x6xxx Input Data of the Module (0x6000 - 0x6FFF)												
0x6000	RECORD		1 BOOL	SWITCH1	0		M	ro	tx			
			2 BOOL	SWITCH2	0		M	ro	tx			
//0x7xxx Output Data of the Module (0x7000 - 0x7FFF)												
0x7010	RECORD		1 BOOL	LED1	0		M	rw	rx			
			2 BOOL	LED2	0		M	rw	rx			
//0x8xxx Configuration Data of the Module (0x8000 - 0x8FFF)												
//0x9xxx Information Data of the Module (0x9000 - 0x9FFF)												
//0xAxxx Diagnosis Data of the Module (0xA000 - 0xAFFF)												
//0xFxxx Device Objects (0xF000 - 0xFFFF)												
0xF000	RECORD		1 UINT16	Modular Device Profile	0x10		M	ro				
			2 UINT16	Index distance			M	ro				
				Maxiaum number of modules			M	ro				

PDO

SDO

第二种方法手动添加 SDO & PDO 的方法适用于应用程序较为复杂的应用工程，参数映射较多，已经比较成熟的应用中，可以手动进行添加。在使用 CIA402 的例程时，使用配置的 xxx config.xml 会生成对应的代码工程，此时不支持再通过导入 excel 创建新的 application 的方式。也就是如想在 cia402 例程上增加新的 CoE 对象时，需要在原有基础上手动修改 xml 和从站代码，没办法通过编辑 excel 导入的办法完成。在手动修改时需要注意：声明相关的对象字典定义，并添加到对象字典中；代码修改与 xml 修改的一致性；如果是 PDO 对象，还需要修改对应的 RxPDO 和 TxPDO 中的定义，添加对象到 PDO 的映射。

5.1 添加 SDO

以 GD32H75EY_EVAL 板的 23_EtherCAT 例程为例，使用的是 CIA402 的工程，基于上面的步骤完成协议栈的移植和编译后，需要新增一组参数 Target Torque Value (0x6071), Current Actual Value (0x6078) 用于通信使用，通信采用 SDO 方式。

1. 修改 XML 文件 在 GDESC_CIA402.xml 中添加对应的对象字典内容

```
<Object>
  <Index>#x6071</Index>
  <Name>Target Torque Value</Name>
  <Type>INT</Type>
  <BitSize>16</BitSize>
  <Info>
    <SubItem>
      <Name>Target Torque Value</Name>
      <Info>
        <DefaultData>0</DefaultData>
      </Info>
    </SubItem>
  </Info>
  <Flags>
    <Access>rw</Access>
    <Category>o</Category>
    <PdoMapping>R</PdoMapping>
  </Flags>
</Object>
```

```
<Object>
  <Index>#x6078</Index>
  <Name>Current Actual Value</Name>
  <Type>INT</Type>
  <BitSize>16</BitSize>
  <Info>
    <SubItem>
      <Name>Current Actual Value</Name>
      <Info>
        <DefaultData>0</DefaultData>
      </Info>
    </SubItem>
  </Info>
  <Flags>
    <Access>ro</Access>
    <Category>o</Category>
    <PdoMapping>T</PdoMapping>
  </Flags>
</Object>
```

2. 修改协议栈代码中的 cia402appl.h 和 cia402appl.c 两个文件
先修改 cia402appl.c 添加字典数据初值赋值的代码

```
case 0x606C:
  pDiEntry->pVarPtr = &LocalAxes[AxisCnt].Objects.objVelocityActualValue;
  break;
case 0x6071:
  pDiEntry->pVarPtr = &LocalAxes[AxisCnt].Objects.objTargetTorqueValue;
  break;
case 0x6077:
  pDiEntry->pVarPtr = &LocalAxes[AxisCnt].Objects.objTorqueActualValue;      CiA402_Init函数
  break;
case 0x6078:
  pDiEntry->pVarPtr = &LocalAxes[AxisCnt].Objects.objCurrentActualValue;
  break;
case 0x607A:
  pDiEntry->pVarPtr = &LocalAxes[AxisCnt].Objects.objTargetPosition;
  break;
```

再修改 cia402appl.h 文件在结构体中添加具体字典声明和定义

```
typedef struct OBJ_STRUCT_PACKED_START
{
    T0B11600 sRxPDOMap0; /**< \brief csv/csp RxPDO (0x1600)*/
    T0B11601 sRxPDOMap1; /**< \brief csv/csp RxPDO (0x1601)*/
    T0B11602 sRxPDOMap2; /**< \brief csv/csp RxPDO (0x1602)*/
    T0B11A00 sTxPDOMap0; /**< \brief csv/csp TxPDO (0x1A00)*/
    T0B11A01 sTxPDOMap1; /**< \brief csv/csp TxPDO (0x1A01)*/
    T0B11A02 sTxPDOMap2; /**< \brief csv/csp TxPDO (0x1A02)*/

    UINT16 objErrorCode; /**< \brief Error Code (0x603F)*/
    UINT16 objControlWord; /**< \brief Control Word (0x6040)*/
    UINT16 objStatusWord; /**< \brief Status Word (0x6041)*/
    INT16 objQuickStopOptionCode; /**< \brief Quick Stop Option Code (0x605A)*/
    INT16 objShutdownOptionCode; /**< \brief Shutdown Option Code (0x605B)*/
    INT16 objDisableOperationOptionCode; /**< \brief Disable Operation Option Code (0x605C)*/
    INT16 objHaltOptionCode; /**< \brief Halt Option Code (0x605D)*/
    INT16 objFaultReactionCode; /**< \brief Fault Reaction Code (0x605E)*/
    INT16 objModesOfOperation; /**< \brief Modes of Operation (0x6060)*/
    INT16 objModesOfOperationDisplay; /**< \brief Mode of Operation Display (0x6061)*/
    INT32 objPositionDemandValue; /**< \brief Position Demand Value (0x6062)*/
    INT32 objPositionActualInternalValue; /**< \brief Position Actual Internal Value (0x6063)*/
    INT32 objPositionActualValue; /**< \brief Position Actual Value (0x6064)*/
    UINT32 objFollowingErrorWindow; /**< \brief Following Error Window (0x6065)*/
    UINT16 objFollowingErrorTimeOut; /**< \brief Following Error Time Out (0x6066)*/
    UINT32 objPositionWindow; /**< \brief Position Window (0x6067)*/
    UINT16 objPositionWindowTime; /**< \brief Position Window Time (0x6068)*/
    INT32 objVelocityActualValue; /**< \brief Actual Velocity Value (0x606C)*/
    INT16 objTargetTorqueValue; /**< \brief Target Torque Value (0x6071)*/
    INT16 objTorqueActualValue; /**< \brief Torque Actual Value (0x6077)*/
    INT16 objCurrentActualValue; /**< \brief Current Actual Value (0x6078)*/
    INT32 objTargetPosition; /**< \brief Target Position (0x607A)*/
    INT32 objHomeOffset; /**< \brief Home Offset (0x607C)*/
    T0B1607D objSoftwarePositionLimit; /**< \brief Software Position Limit (0x607D)*/
    UINT8 objInvertDir; /**< \brief Invert Dir (0x607E)*/
    UINT32 objMaxProfileVelocity; /**< \brief Max Profile Velocity (0x607F)*/
    UINT32 objMaxMotorSpeed; /**< \brief Max Motor Speed (0x6080)*/
    UINT32 objProfileVelocity; /**< \brief Profile Velocity (0x6081)*/
    UINT32 objProfileAcceleration; /**< \brief Profile Acceleration (0x6083)*/
    UINT32 objProfileDeceleration; /**< \brief Profile Deceleration (0x6084)*/
    UINT32 objQuickStopDeclaration; /**< \brief Quick Stop Declaration (0x6085)*/
    INT8 objHomingMethod; /**< \brief Homing Method (0x6098)*/
    T0B70699 objHomingSpeeds; /**< \brief Homing Speeds (0x6099)*/
    UINT32 objHomingAcceleration; /**< \brief Homing Acceleration (0x609A)*/
    T0B706C2 objInterpolationTimePeriod; /**< \brief Interpolation Time Period (0x60C2)*/
    INT32 objPosDemand; /**< \brief Pos Demand (0x60FC)*/
    INT32 objTargetVelocity; /**< \brief Target Velocity (0x60FF)*/
    UINT32 objSupportedDriveModes; /**< \brief Supported Drive Modes (0x6502)*/
}
```

```
/** \brief Object 0x606C (Velocity Actual Value) entry description*/
OBJCONST TSDOINFOENTRYDESC OBJ3MEM sEntryDesc0x606C = {DEFTYPE_INTEGER32, 0x20, (ACCESS_READ | OBJACCESS_TXPDO_MAPPING)};

/** \brief Object 0x606C (Velocity Actual Value) object name*/
OBJCONST UCHAR OBJ3MEM aName0x606C[] = "Velocity Actual Value";

/** \brief Object 0x6071 (Target Torque Value) entry description*/
OBJCONST TSDOINFOENTRYDESC OBJ3MEM sEntryDesc0x6071 = {DEFTYPE_INTEGER16, 0x10, (ACCESS_READWRITE | OBJACCESS_RXPDO_MAPPING)};
/** \brief Object 0x6071 (Target Torque Value) object name*/
OBJCONST UCHAR OBJ3MEM aName0x6071[] = "Target Torque Value";
/** \brief Object 0x6077 (Torque Actual Value) entry description*/
OBJCONST TSDOINFOENTRYDESC OBJ3MEM sEntryDesc0x6077 = {DEFTYPE_INTEGER16, 0x10, (ACCESS_READ | OBJACCESS_TXPDO_MAPPING)};
/** \brief Object 0x6077 (Torque Actual Value) object name*/
OBJCONST UCHAR OBJ3MEM aName0x6077[] = "Torque Actual Value";

/** \brief Object 0x6078 (Current actual value) entry description*/
OBJCONST TSDOINFOENTRYDESC OBJ3MEM sEntryDesc0x6078 = {DEFTYPE_INTEGER16, 0x10, (ACCESS_READ | OBJACCESS_TXPDO_MAPPING)};
/** \brief Object 0x6078 (Current actual value) object name*/
OBJCONST UCHAR OBJ3MEM aName0x6078[] = "Current Actual Value";

/** \brief Object 0x607A (Target Position) entry description*/
OBJCONST TSDOINFOENTRYDESC OBJ3MEM sEntryDesc0x607A = {DEFTYPE_INTEGER32, 0x20, (ACCESS_READWRITE | OBJACCESS_RXPDO_MAPPING)};
/** \brief Object 0x607A (Target Position) object name*/
OBJCONST UCHAR OBJ3MEM aName0x607A[] = "Target Position";
```

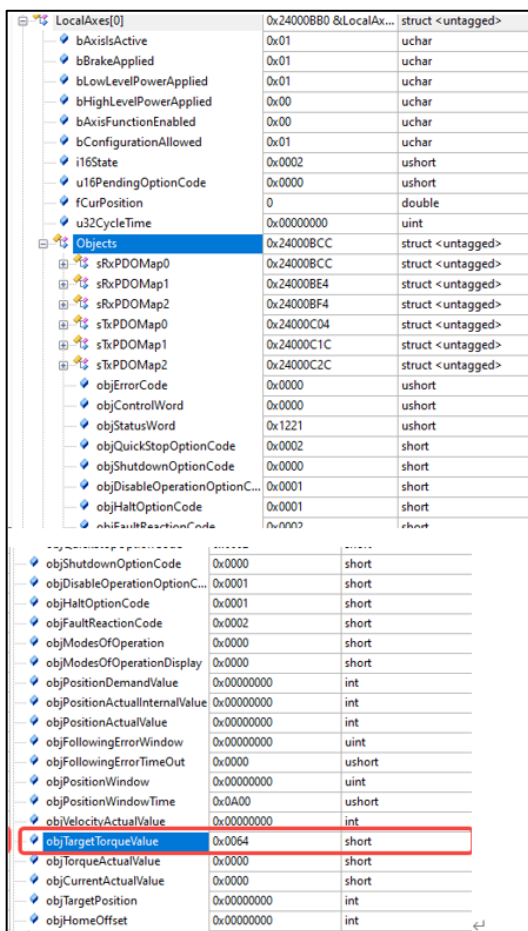
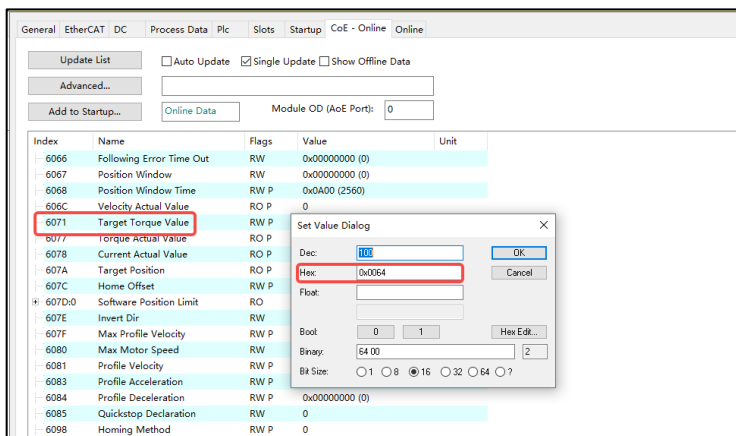
设置参数默认值

```
#ifndef CIA402_
#define CIA402_
{
    {0x00000010, 0x00000000, 0x00000000, 0x00000000}, /* T0B11600*/
    {0x00000010, 0x00000000, 0x00000000, 0x00000000}, /* T0B11601*/
    {0x00000010, 0x00000000, 0x00000000, 0x00000000}, /* T0B11602*/
    {0x00000010, 0x00000000, 0x00000000, 0x00000000}, /* T0B11A00*/
    {0x00000010, 0x00000000, 0x00000000, 0x00000000}, /* T0B11A01*/
    {0x00000010, 0x00000000, 0x00000000, 0x00000000}, /* T0B11A02*/
    0x00, /*(UINT16) ErrorCode 0x603F*/
    0x00, /*(UINT16) ControlWord 0x6040*/
    0x00, /*(UINT16) StatusWord 0x6041*/
    0x00, /*(INT16) QuickStopOptionCode 0x605A*/
    0x00, /*(INT16) ShutdownOptionCode 0x605B*/
    0x00, /*(INT16) DisableOperationCode 0x605C*/
    0x00, /*(INT16) HaltOptionCode 0x605D*/
    0x00, /*(INT16) FaultReactionCode 0x605E*/
    0x00, /*(INT16) ModesOfOperation 0x6060*/
    0x00, /*(INT16) Mode of Operation Display 0x6061*/
    0x00, /*(INT32) Position Demand Value 0x6062*/
    0x00, /*(INT32) Position Actual Internal Value 0x6063*/
    0x00, /*(INT32) Position Actual Value 0x6064*/
    0x00, /*(UINT32) Following Error Window 0x6065*/
    0x00, /*(UINT16) Following Error Time Out 0x6066*/
    0x00, /*(UINT32) Position Window 0x6067*/
    0x00, /*(UINT16) Position Window Time 0x6068*/
    0x00, /*(INT32) Velocity Actual Value 0x606C*/
    0x00, /*(INT16) Target Torque Value (0x6071)*/
    0x00, /*(INT16) Torque Actual Value 0x6077*/
    0x00, /*(INT16) Current Actual Value (0x6078)*/
    0x00, /*(INT32) Home Offset 0x607C*/
    {0x00000000, 0x00000000, 0x00000000, 0x00000000}, /*T0B1607D Software Position Limit (minLimit: -2000000000 / maxLimit: 2000000000)*/
    0x00, /*(UINT8) Invert Dir 0x607E*/
    0x00, /*(UINT32) Max Profile Velocity 0x607F*/
    0x00, /*(UINT32) Max Motor Speed 0x6080*/
    0x00, /*(UINT32) Profile Velocity 0x6081*/
    0x00, /*(UINT32) Profile Acceleration 0x6083*/
    0x00, /*(UINT32) Profile Deceleration 0x6084*/
    0x00, /*(UINT32) QuickStopDeclaration 0x6085*/
    0x00, /*(INT8) Homing Method 0x6098*/
    {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, /*T0B70699 Homing Speeds*/
    0x00, /*(UINT32) QuickStopDeclaration 0x6085*/
    {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, /*T0B706C2 Interpolation Time Period*/
    0x00, /*(INT32) Target Velocity 0x60FF*/
    0x00, /*(INT32) Target Velocity 0x60FF*/
    0x00, /*(UINT32) Supported Drive Modes 0x6502*/
}
#endif
```

将新增字典成员添加到字典中

```
/* Object 0x6068 */
{NULL,NULL, 0x6068, {DEFTYPE_UNSIGNED16, 0 | (OBJCODE_VAR << 8)}, &EntryDesc0x6068, aName0x6068, NULL, NULL, NULL, 0x0000 },
/* Object 0x606C */
{NULL,NULL, 0x606C, {DEFTYPE_INTEGER32, 0 | (OBJCODE_VAR << 8)}, &EntryDesc0x606C, aName0x606C, NULL, NULL, NULL, 0x0000 },
/* Object 0x6071 */
{NULL,NULL, 0x6071, {DEFTYPE_INTEGER16, 0 | (OBJCODE_VAR << 8)}, &EntryDesc0x6071, aName0x6071, NULL, NULL, NULL, 0x0000 },
/* Object 0x6077 */
{NULL,NULL, 0x6077, {DEFTYPE_INTEGER16, 0 | (OBJCODE_VAR << 8)}, &EntryDesc0x6077, aName0x6077, NULL, NULL, NULL, 0x0000 },
/* Object 0x6078 */
{NULL,NULL, 0x6078, {DEFTYPE_INTEGER16, 0 | (OBJCODE_VAR << 8)}, &EntryDesc0x6078, aName0x6078, NULL, NULL, NULL, 0x0000 },
/* Object 0x607D */
{NULL,NULL, 0x607D, {DEFTYPE_INTEGER32, 2 | (OBJCODE_ARR << 8)}, &EntryDesc0x607D, aName0x607D, NULL, NULL, NULL, 0x0000 },
/* Object 0x607A */
{NULL,NULL, 0x607A, {DEFTYPE_INTEGER32, 0 | (OBJCODE_VAR << 8)}, &EntryDesc0x607A, aName0x607A, NULL, NULL, NULL, 0x0000 },
```

- 主站对 0x6071 写入 0x64，在从站的对应的 objTargetTorqueValue 会接收到对应的数据，用户层可以读取调用对应的数据结构，获取变量值



4. 对于主站只读的 TX 数据，修改从站中 objCurrentActualValue 数据的值，主站同样可以读取到数据

objModesOfOperationDisplay	0x0000	short
objPositionDemandValue	0x00000000	int
objPositionActualInternalValue	0x00000000	int
objPositionActualValue	0x00000000	int
objFollowingErrorWindow	0x00000000	uint
objFollowingErrorTimeOut	0x0000	ushort
objPositionWindow	0x00000000	uint
objPositionWindowTime	0x0A00	ushort
objVelocityActualValue	0x00000000	int
objTargetTorqueValue	0x0064	short
objTorqueActualValue	0x0000	short
objCurrentActualValue	0x0080	short
objTargetPosition	0x00000000	int
objHomeOffset	0x00000000	int
objSoftwarePositionLimit	0x2400C80	struct <untagged>
objInvertDir	0x00	uchar
objMaxProfileVelocity	0x00000000	uint
objMaxMotorSpeed	0x00000000	uint
objProfileVelocity	0x00000000	uint

6066	Following Error Time Out	RW	0x00000000 (0)
6067	Position Window	RW	0x00000000 (0)
6068	Position Window Time	RW P	0x0A00 (2560)
606C	Velocity Actual Value	RO P	0
6071	Target Torque Value	RW P	100
6077	Torque Actual Value	RO P	0
6078	Current Actual Value	RO P	128
607A	Target Position	RO P	0
607C	Home Offset	RW P	0
607D:0	Software Position Limit	RO	> 2 <
607E	Invert Dir	RW	0x00 (0)
607F	Max Profile Velocity	RW P	0x00000000 (0)
6080	Max Motor Speed	RW	0x00000000 (0)
6081	Profile Velocity	RW P	0x00000000 (0)
6083	Profile Acceleration	RW P	0x00000000 (0)

5.2 添加 PDO

同样以 GD32H75EY_EVAL 板的 23_EtherCAT 例程为例，修改 xml 和协议栈代码，新增一组 Target Torque Value (0x6071), Current Actual Value (0x6078)参数采用 PDO 通信方式。

首先按照添加 SDO 的操作方法完成对 XML 文件修改，将变量添加到 XML 文件中，再将变量添加到对应的 RXPDO 和 TXPDO 中。

1. 在对应的 RXPDO(0x1600)和 TXPDO(0x1A00)增加对应索引

DT1600 中增加对应的索引

```
<Name>DT1600</Name>
<BitSize>160</BitSize>

<SubItem>
  <SubIdx>4</SubIdx>
  <Name>SubIndex 004</Name>
  <Type>INT</Type>
  <BitSize>16</BitSize>
  <BitOffs>128</BitOffs>
  <Flags>
    <Access>ro</Access>
    <Category>o</Category>
  </Flags>
</SubItem>

<Index>#x1600</Index>
<Name>csp/csv/pp RxPDO</Name>
<Type>DT1600</Type>
<BitSize>160</BitSize>
<Info>
  <SubItem>
    <Name>SubIndex 000</Name>
    <Info>
      <DefaultData>05</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 001</Name>
    <Info>
      <DefaultData>10004060</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 002</Name>
    <Info>
      <DefaultData>00006060</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 003</Name>
    <Info>
      <DefaultData>20007A60</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 004</Name>
    <Info>
      <DefaultData>2000FF60</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 005</Name>
    <Info>
      <DefaultData>10007160</DefaultData>
    </Info>
  </SubItem>
</Info>
```

DT1A00 中增加对应的索引

```
<Name>DT1A00</Name>
<BitSize>160</BitSize>

<SubItem>
  <SubIdx>5</SubIdx>
  <Name>SubIndex 005</Name>
  <Type>INT</Type>
  <BitSize>16</BitSize>
  <BitOffs>128</BitOffs>
  <Flags>
    <Access>ro</Access>
    <Category>o</Category>
  </Flags>
</SubItem>

<Index>#x1A00</Index>
<Name>csp/csv/pp TxPDO</Name>
<Type>DT1A00</Type>
<BitSize>160</BitSize>
<Info>
  <SubItem>
    <Name>SubIndex 000</Name>
    <Info>
      <DefaultData>05</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 001</Name>
    <Info>
      <DefaultData>10004160</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 002</Name>
    <Info>
      <DefaultData>00006160</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 003</Name>
    <Info>
      <DefaultData>20006460</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 004</Name>
    <Info>
      <DefaultData>20006C60</DefaultData>
    </Info>
  </SubItem>
  <SubItem>
    <Name>SubIndex 005</Name>
    <Info>
      <DefaultData>10007160</DefaultData>
    </Info>
  </SubItem>
</Info>
```

- 在需要的 Module 中添加对应字典成员，本例添加在"#x119800"中，在 RX 和 TX 中添加对应字典成员

```
<Type ModuleIdet="#x119800">csv,csp,pp - axis</Type>
<Name>dynamic switch bewteen csp/csv/pp</Name>
<RxPdo Fixed="true" Sm="2">
  <Index DependOnSlot="true">#x1600</Index>
  <Name>Outputs</Name>
  <Entry>
    <Index DependOnSlot="true">#x6040</Index>
    <SubIndex>0</SubIndex>
    <BitLen>16</BitLen>
    <Name>Control Word</Name>
    <Comment>object 0x6040:0</Comment>
    <DataType>UINT</DataType>
  </Entry>
  <Entry>
    <Index DependOnSlot="true">#x607A</Index>
    <SubIndex>0</SubIndex>
    <BitLen>32</BitLen>
    <Name>TargetPosition</Name>
    <Comment>object 0x607A:0</Comment>
    <DataType>DINT</DataType>
  </Entry>
  <Entry>
    <Index DependOnSlot="true">#x60FF</Index>
    <SubIndex>0</SubIndex>
    <BitLen>32</BitLen>
    <Name>TargetVelocity</Name>
    <Comment>object 0x60FF:0</Comment>
    <DataType>DINT</DataType>
  </Entry>
  <Entry>
    <Index DependOnSlot="true">#x60FF</Index>
    <SubIndex>0</SubIndex>
    <BitLen>16</BitLen>
    <Name>TargetTorqueValue</Name>
    <Comment>object 0x6071:0</Comment>
    <DataType>INT</DataType>
  </Entry>
</Type>
```

```
<Index DependOnSlot="true">#x1a00</Index>
<Name>Inputs</Name>
<Entry>
  <Index DependOnSlot="true">#x6041</Index>
  <SubIndex>0</SubIndex>
  <BitLen>16</BitLen>
  <Name>Status Word</Name>
  <Comment>object 0x6041:0</Comment>
  <DataType>UINT</DataType>
</Entry>
<Entry>
  <Index DependOnSlot="true">#x6064</Index>
  <SubIndex>0</SubIndex>
  <BitLen>32</BitLen>
  <Name>ActualPosition</Name>
  <Comment>object 0x6064:0</Comment>
  <DataType>DINT</DataType>
</Entry>
<Entry>
  <Index DependOnSlot="true">#x606C</Index>
  <SubIndex>0</SubIndex>
  <BitLen>32</BitLen>
  <Name>ActualVelocity</Name>
  <Comment>object 0x606C:0</Comment>
  <DataType>DINT</DataType>
</Entry>
<Entry>
  <Index DependOnSlot="true">#x6077</Index>
  <SubIndex>0</SubIndex>
  <BitLen>16</BitLen>
  <Name>TorqueActualValue</Name>
  <Comment>object 0x6078:0</Comment>
  <DataType>INT</DataType>
</Entry>
```

同时添加对应字典到 Module 中的 Dictionary 下的 Objects 中

```
<Object>
  <Object>
    <Index DependOnSlot="true">#x6071</Index>
    <Name>target torque value</Name>
    <Type>INT</Type>
    <BitSize>16</BitSize>
    <Flags>
      <Access>rw</Access>
      <PdoMapping>r</PdoMapping>
    </Flags>
  </Object>
</Object>
<Object>
  <Index DependOnSlot="true">#x6077</Index>
  <Name>Torque actual value</Name>
  <Type>INT</Type>
  <BitSize>16</BitSize>
  <Flags>
    <Access>ro</Access>
    <PdoMapping>t</PdoMapping>
  </Flags>
</Object>
<Object>
  <Index DependOnSlot="true">#x6078</Index>
  <Name>Current actual value</Name>
  <Type>INT</Type>
  <BitSize>16</BitSize>
  <Flags>
    <Access>ro</Access>
    <PdoMapping>t</PdoMapping>
  </Flags>
</Object>
</Object>
```

3. 在上面添加完代码的基础上继续添加支持 PDO 访问的代码，首先修改 cia402appl.c 文件先增加 TXPDO 的 Current Actual Value (0x6078)数据访问代码

```
void APPL_InputMapping(UINT16* pData)
{
    UINT16 j = 0;
    UINT16 *pTmpData = (UINT16 *)pData;
    UINT8 AxisIndex;

    for (j = 0; j < sTxPDOassign.u16SubIndex0; j++)
    {
        /*The Axis index is based on the PDO mapping offset (0x10)*/
        AxisIndex = ((sTxPDOassign.aEntries[j] & 0xF0) >> 4);

        switch ((sTxPDOassign.aEntries[j] & 0x000F))
        {
            case 0: //copy csp/csv TxPDO entries
            {
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objStatusWord;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objPositionActualValue & 0xFFFF;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objPositionActualValue >> 16;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objVelocityActualValue & 0xFFFF;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objVelocityActualValue >> 16;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objCurrentActualValue & 0xFFFF;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objModesOfOperationDisplay & 0xFF;
            }
            break;
            case 1: //copy csp TxPDO entries
            {
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objStatusWord;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objPositionActualValue & 0xFFFF;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objPositionActualValue >> 16;
            }
            break;
            case 2: //copy csv TxPDO entries
            {
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objStatusWord;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objVelocityActualValue & 0xFFFF;
                *pTmpData++ = LocalAxes[AxisIndex].Objects.objVelocityActualValue >> 16;
            }
            break;
        }
        //switch TxPDO entry
    }
}
```

再增加 RX 的 Target Torque Value (0x6071)数据访问代码

```
void APPL_OutputMapping(UINT16* pData)
{
    UINT16 j = 0;
    UINT16 *pTmpData = (UINT16 *)pData;
    UINT8 AxisIndex;

    for (j = 0; j < sRxPDOassign.u16SubIndex0; j++)
    {
        /*The Axis index is based on the PDO mapping offset (0x10)*/
        AxisIndex = ((sRxPDOassign.aEntries[j] & 0xF0) >> 4);

        switch ((sRxPDOassign.aEntries[j] & 0x000F))
        {
            case 0: //csp/csv RxPDO entries
            {
                LocalAxes[AxisIndex].Objects.objControlWord = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetPosition = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetPosition += (SWAPWORD(*pTmpData++) << 16);
                LocalAxes[AxisIndex].Objects.objTargetVelocity = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetVelocity += (SWAPWORD(*pTmpData++) << 16);
                LocalAxes[AxisIndex].Objects.objTargetTorqueValue = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objModesOfOperation = SWAPWORD(*pTmpData++) & 0xFF;
            }
            break;
            case 1: //csp RxPDO entries
            {
                LocalAxes[AxisIndex].Objects.objControlWord = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetPosition = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetPosition += (SWAPWORD(*pTmpData++) << 16);
            }
            break;
            case 2: //csv RxPDO entries
            {
                LocalAxes[AxisIndex].Objects.objControlWord = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetVelocity = SWAPWORD(*pTmpData++);
                LocalAxes[AxisIndex].Objects.objTargetVelocity += (SWAPWORD(*pTmpData++) << 16);
            }
            break;
        }
    }
}
```

4. 修改 cia402appl.h 的代码

增加实体数目

```

/** \brief 0x1600 (csp/csv RxPDO) data structure*/
typedef struct OBJ_STRUCT_PACKED_START {
    UINT16 u16SubIndex0; /**< \brief SubIndex 0*/
    UINT32 aEntries[6]; /**< \brief Entry buffer*/
} OBJ_STRUCT_PACKED_END
TOBJ1600;

/** \brief 0x1601 (csp RxPDO) data structure*/
typedef struct OBJ_STRUCT_PACKED_START {
    UINT16 u16SubIndex0; /**< \brief SubIndex 0*/
    UINT32 aEntries[3]; /**< \brief Entry buffer*/
} OBJ_STRUCT_PACKED_END
TOBJ1601;

/** \brief 0x1602 (csv RxPDO) data structure*/
typedef struct OBJ_STRUCT_PACKED_START {
    UINT16 u16SubIndex0; /**< \brief SubIndex 0*/
    UINT32 aEntries[3]; /**< \brief Entry buffer*/
} OBJ_STRUCT_PACKED_END
TOBJ1602;

/** \brief 0x1A00 (csp/csv TxPDO) data structure*/
typedef struct OBJ_STRUCT_PACKED_START {
    UINT16 u16SubIndex0; /**< \brief SubIndex 0*/
    UINT32 aEntries[4]; /**< \brief Entry buffer*/
} OBJ_STRUCT_PACKED_END
TOBJ1A00;

```

增加具体映射成员定义

```

/** \brief Data structure to handle the process data transmitted via 0x1A00 (csp/csv TxPDO)*/
typedef struct STRUCT_PACKED_START {
    UINT16 ObjStatusWord; /**< \brief Status word (0x6041)*/
    INT32 ObjPositionActualValue; /**< \brief Actual position (0x6064)*/
    INT32 ObjVelocityActualValue; /**< \brief Actual velocity (0x6066)*/
    INT16 ObjCurrentActualValue; /**< \brief Current Actual Value (0x6078)*/
    INT16 ObjModesOfOperationDisplay; /**< \brief Current Mode of operation (0x6061)*/
}STRUCT_PACKED_END
TCIA402PDO1A00;

/** \brief Data structure to handle the process data transmitted via 0x1A01 (csp TxPDO)*/
typedef struct STRUCT_PACKED_START {
    UINT16 ObjStatusWord; /**< \brief Status word (0x6041)*/
    INT32 ObjPositionActualValue; /**< \brief Actual position (0x6064)*/
    INT16 Padding16Bit; /**< \brief 16bit padding*/
}STRUCT_PACKED_END
TCIA402PDO1A01;

/** \brief Data structure to handle the process data transmitted via 0x1A02 (csv TxPDO)*/
typedef struct STRUCT_PACKED_START {
    UINT16 ObjStatusWord; /**< \brief Status word (0x6041)*/
    INT32 ObjPositionActualValue; /**< \brief Actual position (0x6064)*/
    INT16 Padding16Bit; /**< \brief 16bit padding*/
}STRUCT_PACKED_END
TCIA402PDO1A02;

/** \brief Data structure to handle the process data transmitted via 0x1600 (csp/csv RxPDO)*/
typedef struct STRUCT_PACKED_START {
    UINT16 ObjControlWord; /**< \brief Control word (0x6040)*/
    INT32 ObjTargetPosition; /**< \brief Target position (0x607A)*/
    INT32 ObjTargetVelocity; /**< \brief Target velocity (0x6065)*/
    INT16 ObjTargetTorqueValue; /**< \brief Target Torque Value (0x6071)*/
    INT16 ObjModesOfOperation; /**< \brief Mode of operation (0x6060)*/
}STRUCT_PACKED_END
TCIA402PDO1600;

```

增加对应的实体索引

```

/**
 * \brief Object 0x1600 (csp/csv RxPDO) entry descriptions
 */
OBJCONST TSDOINFOENTRYDESC OBJMEM asEntryDesc0x1600[] = {
    {DEFTYPE_UNSIGNED8, 0x8, ACCESS_READ }, /* Subindex 000 */
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 001*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 002*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 003*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 004*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 005*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}; /* SubIndex 006*/
}

```

```
/**
 * \brief Object 0x1A00 (csp/csv TxPDO) entry descriptions
 */
OBJCONST TSDOINFOENTRYDESC OBJMEM asEntryDesc0x1A00[] = {
    {DEFTYPE_UNSIGNED8, 0x8, ACCESS_READ }, /* SubIndex 000 */
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 001*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 002*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 003*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 004*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}, /* SubIndex 005*/
    {DEFTYPE_UNSIGNED32, 0x20, ACCESS_READ}; /* SubIndex 006*/
}
```

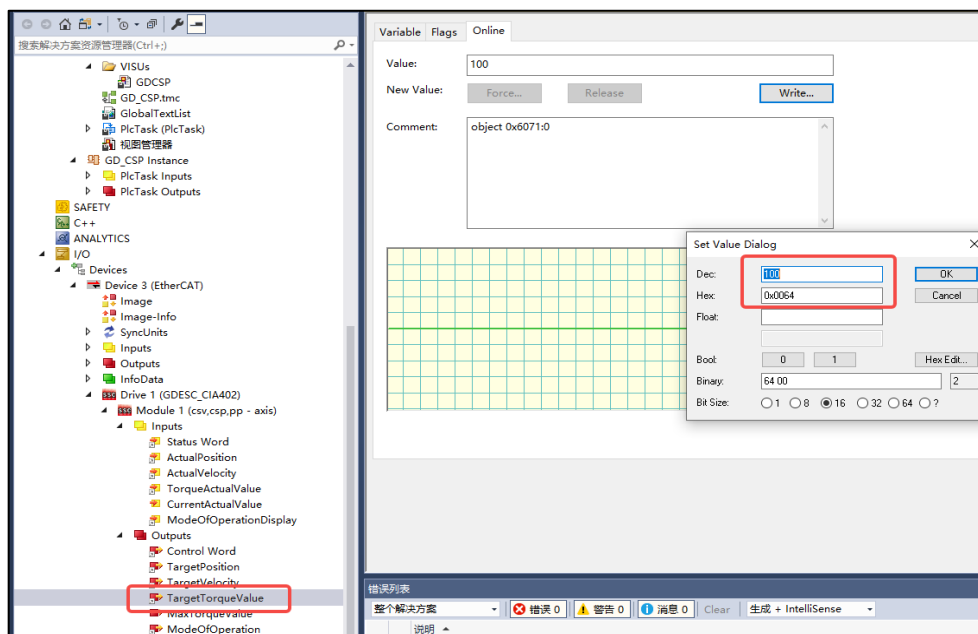
将字典成员添加到对应的映射中

```
PROTO CiA402Objects DefCiA402ObjectValues
#ifdef _CiA402_
= {
    {6, {0x60400010, 0x607A0020, 0x60FF0020, 0x60710010, 0x60600008, 0x00000008}}, /* TOBJ1600*/
    {3, {0x60400010, 0x607A0020, 0x00000010}}, /*TOBJ1601*/
    {3, {0x60400010, 0x60FF0020, 0x00000010}}, /*TOBJ1602*/
    {6, {0x60410010, 0x60640020, 0x606C0020, 0x60780010, 0x60610008, 0x00000008}}, /*TOBJ1A00*/
}
```

修改 CiA402 Axis 字典中映射中成员的数目

```
/*\brief Object dictionary related to on CiA402 Axis
 */
PROTO TOBJECT OBJMEM DefCiA402AxisObjDic[]
#ifdef _CiA402_
= {
    /* Object 0x1600 */
    {NULL, NULL, 0x1600, {DEFTYPE_PDOMAPPING, 6 | (OBJCODE_REC << 8)}, asEntryDesc0x1600, aName0x1600, NULL, NULL, NULL, 0x0000},
    /* Object 0x1601 */
    {NULL, NULL, 0x1601, {DEFTYPE_PDOMAPPING, 3 | (OBJCODE_REC << 8)}, asEntryDesc0x1601, aName0x1601, NULL, NULL, NULL, 0x0000},
    /* Object 0x1602 */
    {NULL, NULL, 0x1602, {DEFTYPE_PDOMAPPING, 3 | (OBJCODE_REC << 8)}, asEntryDesc0x1602, aName0x1602, NULL, NULL, NULL, 0x0000},
    /* Object 0x1A00 */
    {NULL, NULL, 0x1A00, {DEFTYPE_PDOMAPPING, 6 | (OBJCODE_REC << 8)}, asEntryDesc0x1A00, aName0x1A00, NULL, NULL, NULL, 0x0000},
}
```

- 主站通过 RXPDO(0x6071)写入 0x64，在从站的对应的 objTargetTorqueValue 会接收到对应的数据，用户层可以读取调用对应的数据结构，获取变量值



The screenshot displays a debugger interface with a C++ code window on the left and a 'Name' vs 'Value' watch window on the right.

Code Window:

```

For [j = 0; j < sRxPDOAssign.ul6SubIndex0; j++]
{
    /*The Axis Index is based on the PDO mapping offset (0x10)*/
    AxisIndex = ((sRxPDOAssign.sEntries[j] & 0x0F0) >> 4);

    switch ((sRxPDOAssign.sEntries[j] & 0x000F))
    {
        case 0: //csp/csw RxFDO...entries
        {
            LocalAxes[AxisIndex].Objects.objControlWord = SWAPWORD(*pTmpData++);
            LocalAxes[AxisIndex].Objects.objTargetPosition = SWAPWORD(*pTmpData++);
            LocalAxes[AxisIndex].Objects.objTargetPosition = SWAPWORD(*pTmpData++) << 16;
            LocalAxes[AxisIndex].Objects.objTargetVelocity = SWAPWORD(*pTmpData++);
            LocalAxes[AxisIndex].Objects.objTargetVelocity = SWAPWORD(*pTmpData++) << 16;
            LocalAxes[AxisIndex].Objects.objTargetTorqueValue = SWAPWORD(*pTmpData++);
            LocalAxes[AxisIndex].Objects.objCurrentActualValue = SWAPWORD(*pTmpData++);
            LocalAxes[AxisIndex].Objects.objModeOnOperation = SWAPWORD(*pTmpData++) << 0x0FF;

        }
        break;
        case 1: //csp RxFDO...entries
        {
            LocalAxes[AxisIndex].Objects.objControlWord = SWAPWORD(*pTmpData++);
            LocalAxes[AxisIndex].Objects.objTargetPosition = SWAPWORD(*pTmpData++);

```

Watch Window:

Name	Value
LocalAxes[0].Objects.objTargetTorqueValue	0.0064
LocalAxes[0].Objects.objCurrentActualValue	0.0000

The watch window also includes a prompt: "< Enter expression >".

6. 对于主站只读的的 TXPDO(0x6078)数据，修改从站中 objCurrentActualValue 数据的值，主站同样可以读取到数据

The screenshot shows a C# code editor with a switch statement. Line 1374 is highlighted, showing a case for 0x000F. To the right, a variable declaration table is visible, showing LocalAxes[0].Objects.objCurrentActualValue with a value of 0x0000.

Name	Value
LocalAxes[0].Objects.objTargetTimeValue	0x0000
LocalAxes[0].Objects.objCurrentActualValue	0x0000

[illegible]

6. 版本历史

表 6-1. 版本历史

版本号.	说明	日期
1.0	首次发布	2024 年 11 月 29 日
1.1	1.增加 EEPROM 更新方法	2025 年 7 月 10 日
1.2	1.增加 COE 添加 SDO & PDO 方法	2025 年 9 月 8 日

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.