

# OPERATION MANUAL

## 2311 EtherCAT Integration into TwinCAT

© 2024 burster  
präzisionsmesstechnik gmbh & co kg  
All rights reserved

Manufacturer:  
burster  
präzisionsmesstechnik gmbh & co kg  
Talstr. 1 - 5 P.O. Box 1432  
76593 Gernsbach 76587 Gernsbach  
Germany Germany

Valid from: **20.02.2024**  
Applies to: **burster 2311 V0001**

Tel.: +49-7224-645-0  
Fax.: +49-7224-645-88  
Email: [info@burster.com](mailto:info@burster.com)  
[www.burster.com](http://www.burster.com)

4516-BA2311ETHERCATEN-5799-021526

## Table of Contents

<b>Introduction</b> .....	<b>3</b>
1. Creating new project .....	4
2. Installation of ESI description files .....	6
3. Scan EtherCAT devices .....	6
4. Create a sample program .....	9
5. Further Examples .....	15
5.1 Read and Write of 'real' data types .....	15
5.2 Read of 'string' data types.....	19

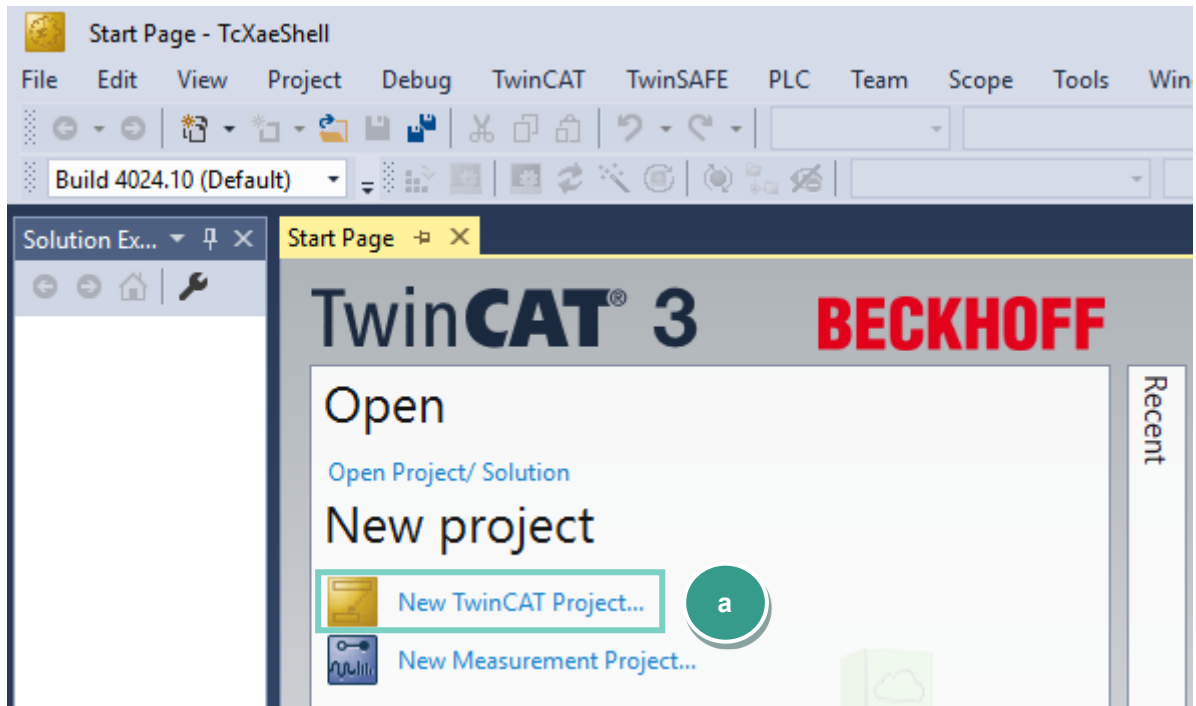
## Introduction

This quick start guide describes an approach how you can configure the 2311 via Beckhoff TwinCAT using a Beckhoff PCI-Ethernet Card. Please note that the samples here cannot be directly used in your production line because they have been extremely simplified to reach a better understanding. Therefore, you may have to complete them by checking of status, error, length values etc.

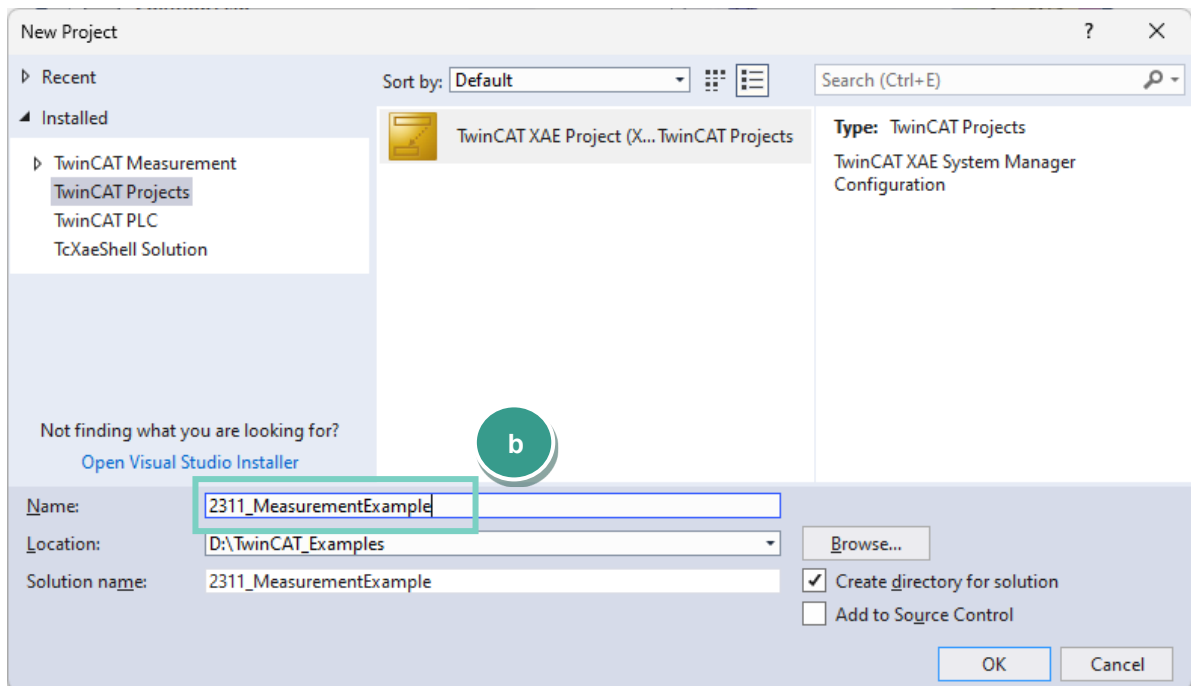
***Please also note that you will have to use the 2311 operational manual as well as 2311 EtherCAT operation manual to get further information about input and output parameters (PDO as well as SDO data transfer)***

## 1. Creating new project

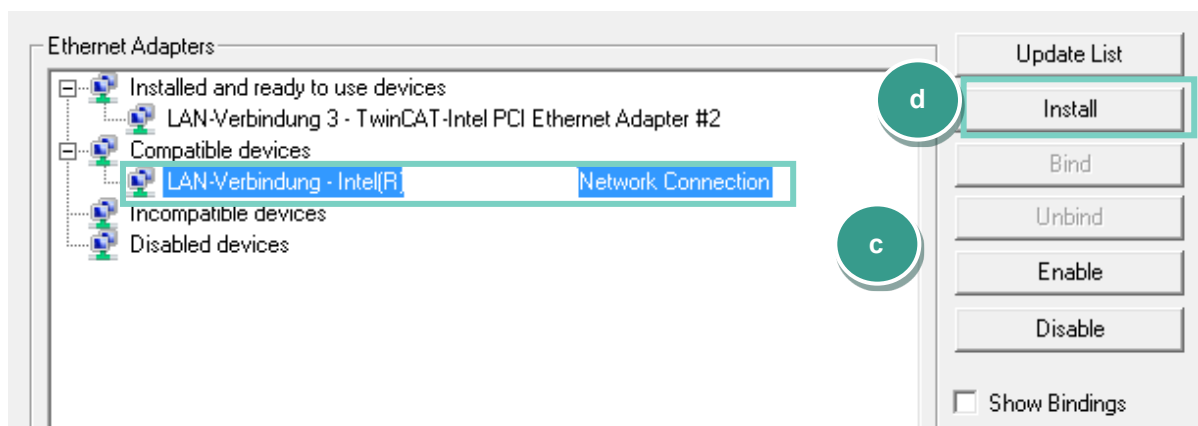
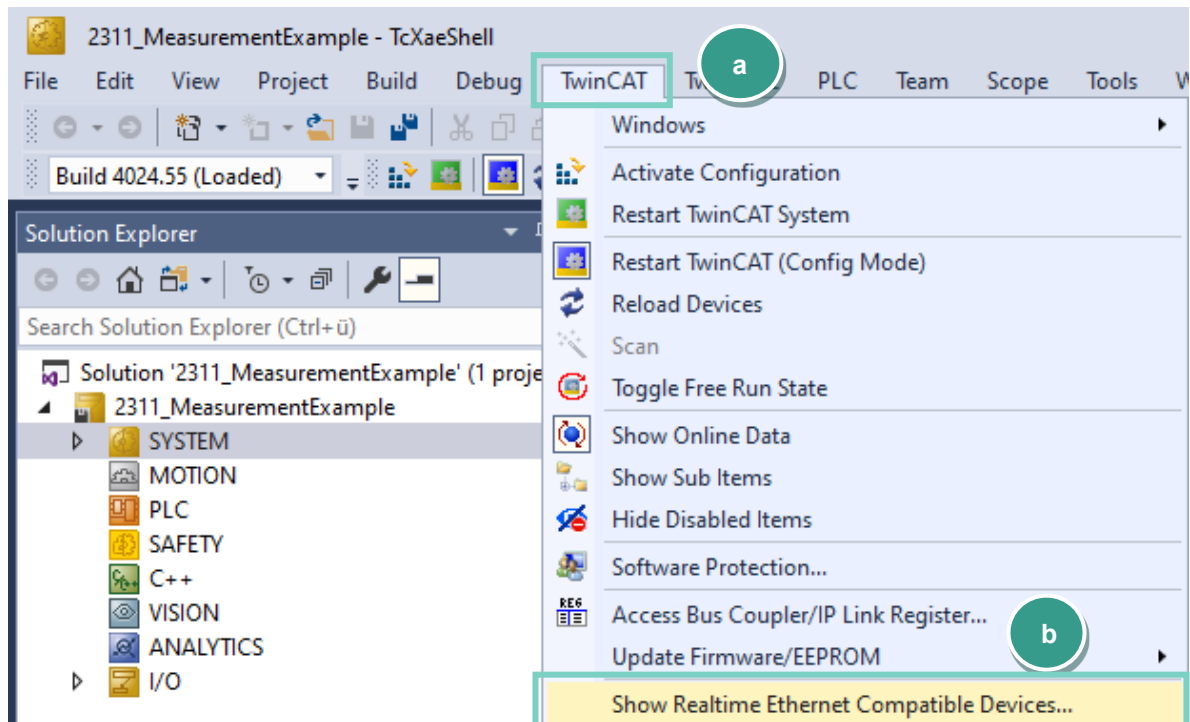
- Start the **TwinCAT XAE Shell** and click on **New TwinCAT Project** (a) (or via **File** → **New Project**)



- Select **TwinCAT XAE Project**, assign a project a name (b) and click **OK**



- Go to **TwinCAT** (a), select **Show Real Time Ethernet Compatible Devices...** (b) and look for your a EtherCAT Master device under Compatible devices\* (c). Afterwards click the **Install** button (d).



\*You can find information of supported network controllers on:

[https://infosys.beckhoff.com/english.php?content=../content/1033/tc3\\_overview/9309844363.html&id](https://infosys.beckhoff.com/english.php?content=../content/1033/tc3_overview/9309844363.html&id)

## 2. Installation of ESI description files

**Note:** Please make sure that your ESI file is compatible to the field bus firmware in the 2311.

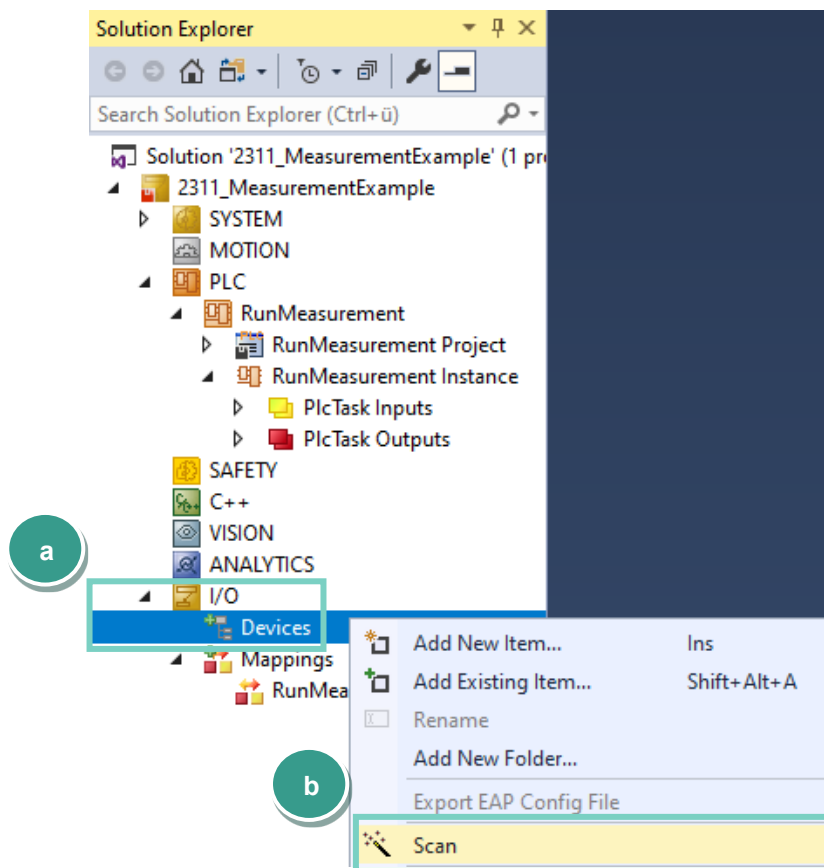
- Copy the ESI file into directory **C:\TwinCAT\3.1\Config\Io\OnboardIo** and additionally into **C:\TwinCAT\3.1\Config\Io\EtherCAT**

**Note:** you will find the corresponding ESI files on burster.com

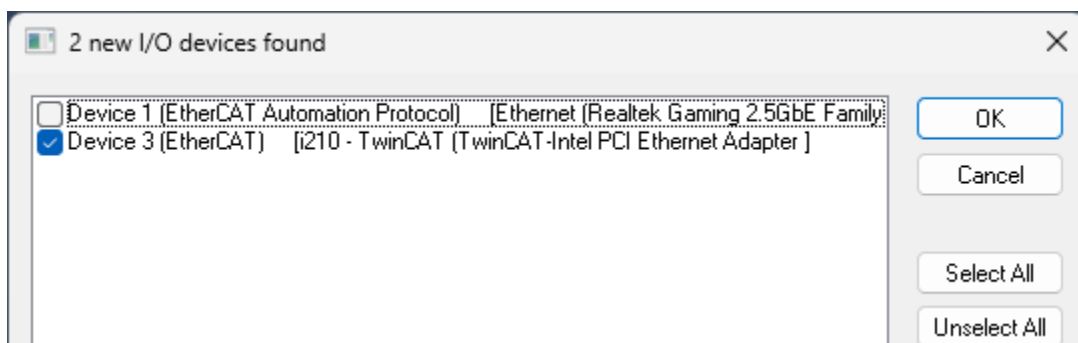
**Note:** If you use the TwinCAT 2 The ESI directory would be **C:\TwinCAT\Io\EtherCAT**

## 3. Scan EtherCAT devices

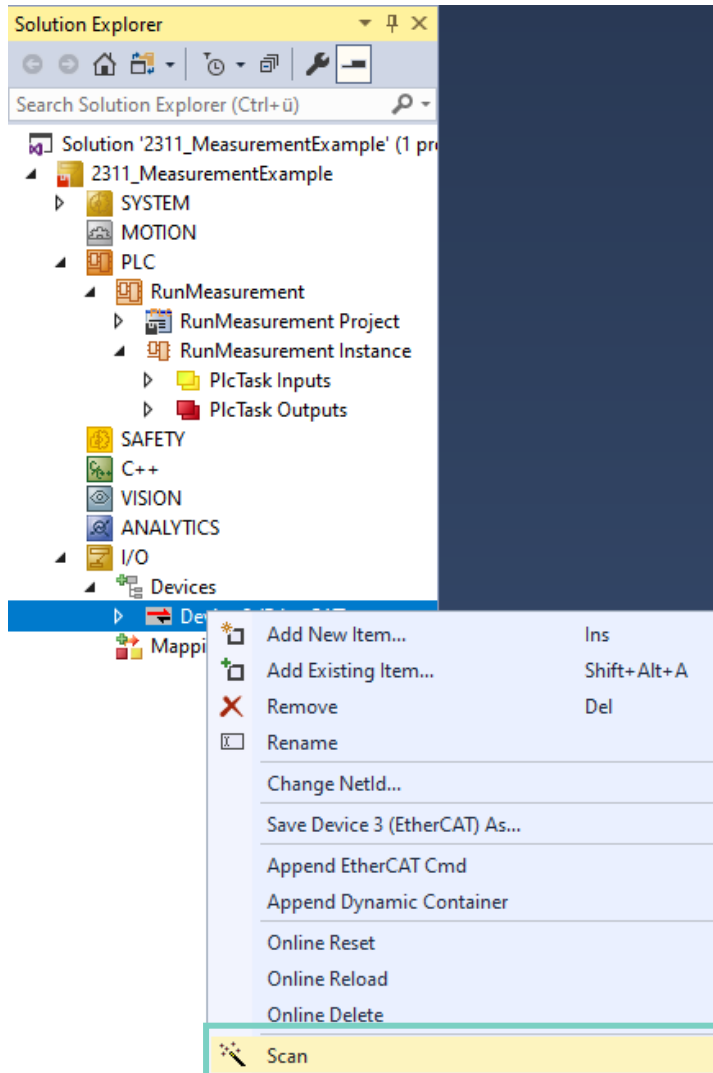
- Connect the 2311 to your EtherCAT master, right click **I/O** → **Devices** (a) in the project tree und select **Scan** (b):



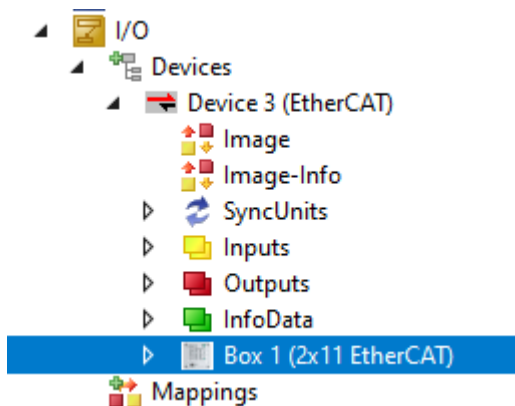
- Now, you can select an EtherCAT compatible device in the new window and click OK:



- At this point you are to perform a device search by confirming the **Scan for boxes** request or later by right-clicking on the found EtherCAT device and selecting **Scan** in the context menu as shown below:



- If the TwinCAT was not able to find the ESI file, confirm the question to use online description and after a while you should be able to see the 2311 device in the project tree:



- Confirm the request to activate **Free Run**

- To see the process data, please click on the 2311 in the project tree (a) and select the **Process Data** tab (b):

The screenshot shows the TwinCAT configuration environment for a 2311 device. The 'Process Data' tab is active, displaying the following data:

**Sync Manager:**

SM	Size	Type	Flags
0	128	MbxOut	
1	128	MbxIn	
2	4	Outputs	
3	12	Inputs	

**PDO List:**

Index	Size	Name	Flags	SM	SU
0x1A00	12.0	Inputs	MF	3	0
0x1600	4.0	Outputs	MF	2	0

**PDO Assignment (0x1C12):**

0x1600

**PDO Content (0x1A00):**

Index	Size	Offs	Name	Type
0x2001:01	0.1	0.0	2311_OUT_READY	BIT
---	0.1	0.1	---	---

**Download:**

PDO Assignment  
 PDO Configuration

**Predefined PDO Assignment:** (none)  
Load PDO info from device  
Sync Unit Assignment...

**Device List:**

Name	Online	Type	Size	>Addr...	In/Out	User ID
2311_OUT_READY	0	BIT	0.1	55.0	Input	0
2311_OUT_MEASUREMENT_ENDED	1	BIT	0.1	55.2	Input	0
2311_OUT_MEASUREMENT_ERROR	0	BIT	0.1	55.3	Input	0
2311_OUT_ERROR	0	BIT	0.1	55.4	Input	0
2311_OUT_PROG0	1	BIT	0.1	56.0	Input	0
2311_OUT_PROG1	0	BIT	0.1	56.1	Input	0
2311_OUT_PROG2	0	BIT	0.1	56.2	Input	0
2311_OUT_PROG3	0	BIT	0.1	56.3	Input	0
2311_OUT_PROG4	0	BIT	0.1	56.4	Input	0

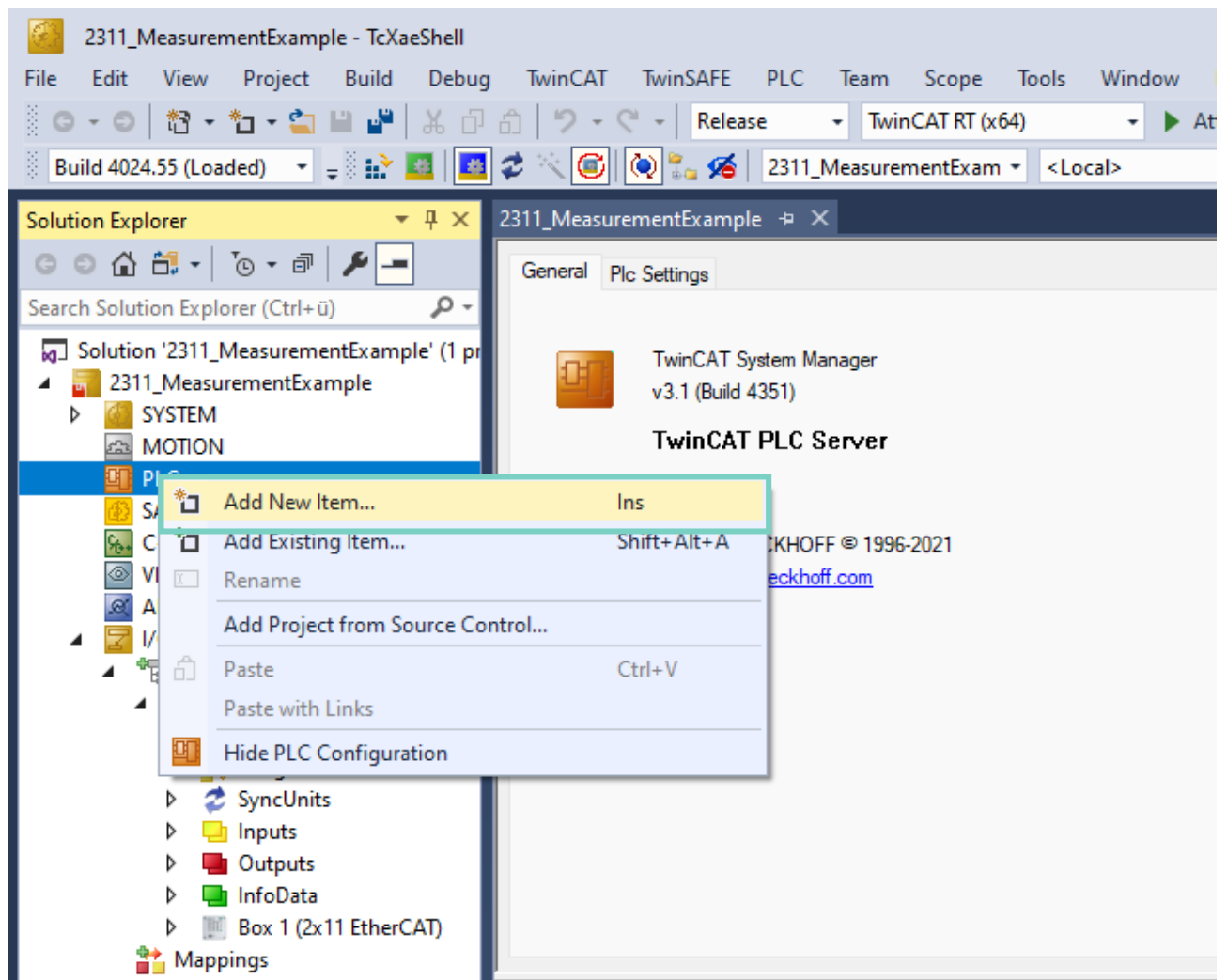
**Error List:** Entire Solution | 0 Errors | 0 Warnings | 0 Messages | Clear | Build + IntelliSense



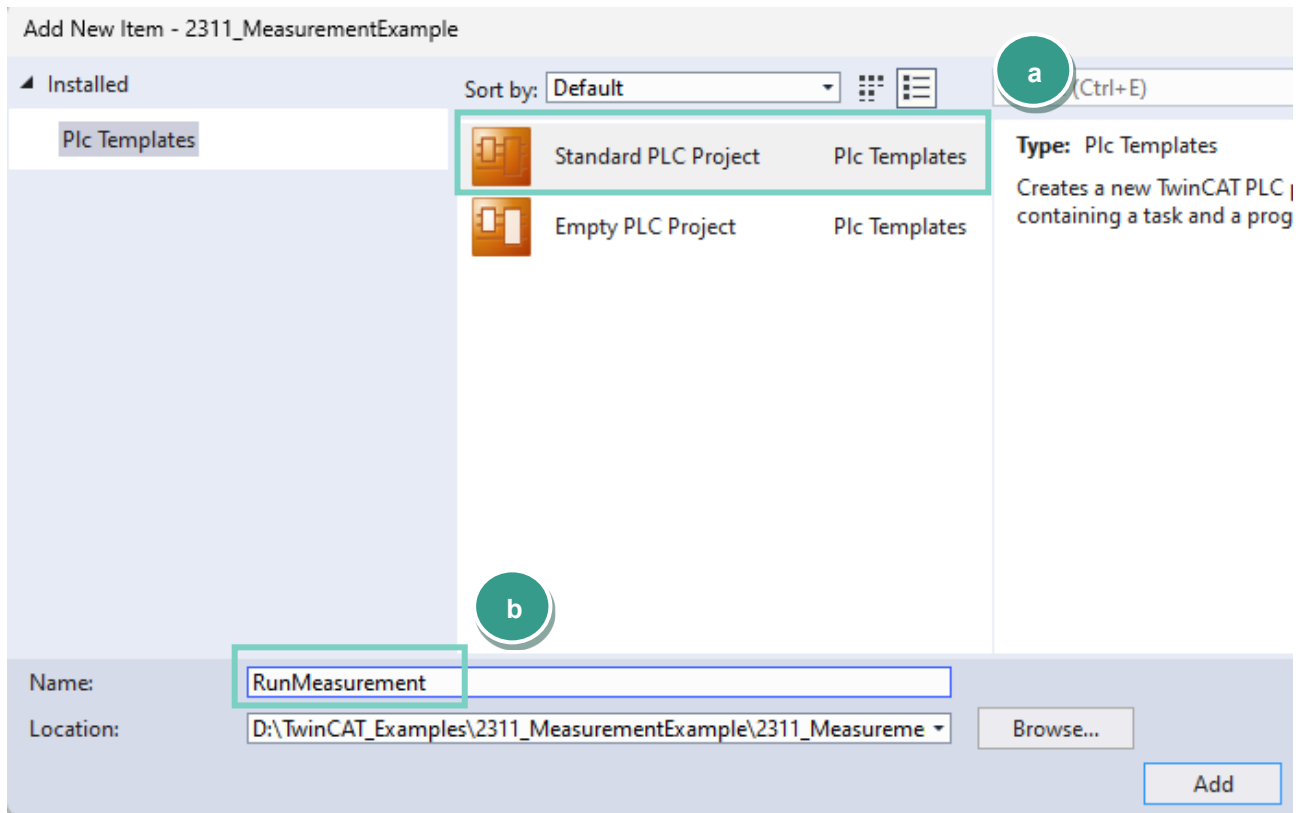
## 4. Create a sample program

In this section, you will learn how to create a simple PLC program to perform a resistance measurement and read the measurement result via PDO (Process Data Object). You will need to refer to section 7 *EtherCAT data protocol* in 2311 EtherCAT operation manual to understand the meaning of input bytes.

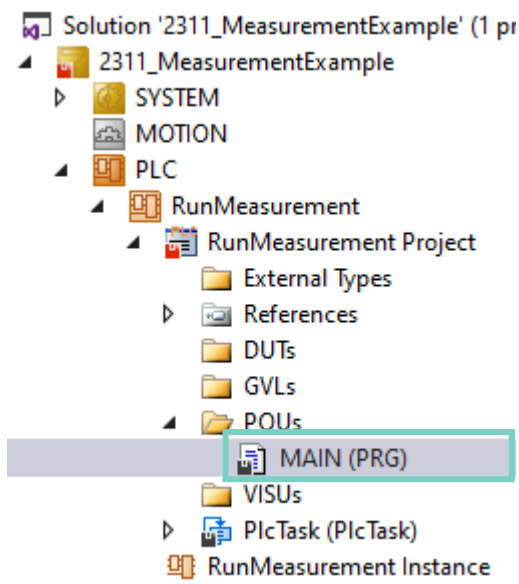
- Right-click **PLC** in the project tree and select **Add New Item...**



- Select **Standard PLC Project** (a) in the **Add New Item** dialog, enter **RunMeasurement** as project name (b) and click **Add**

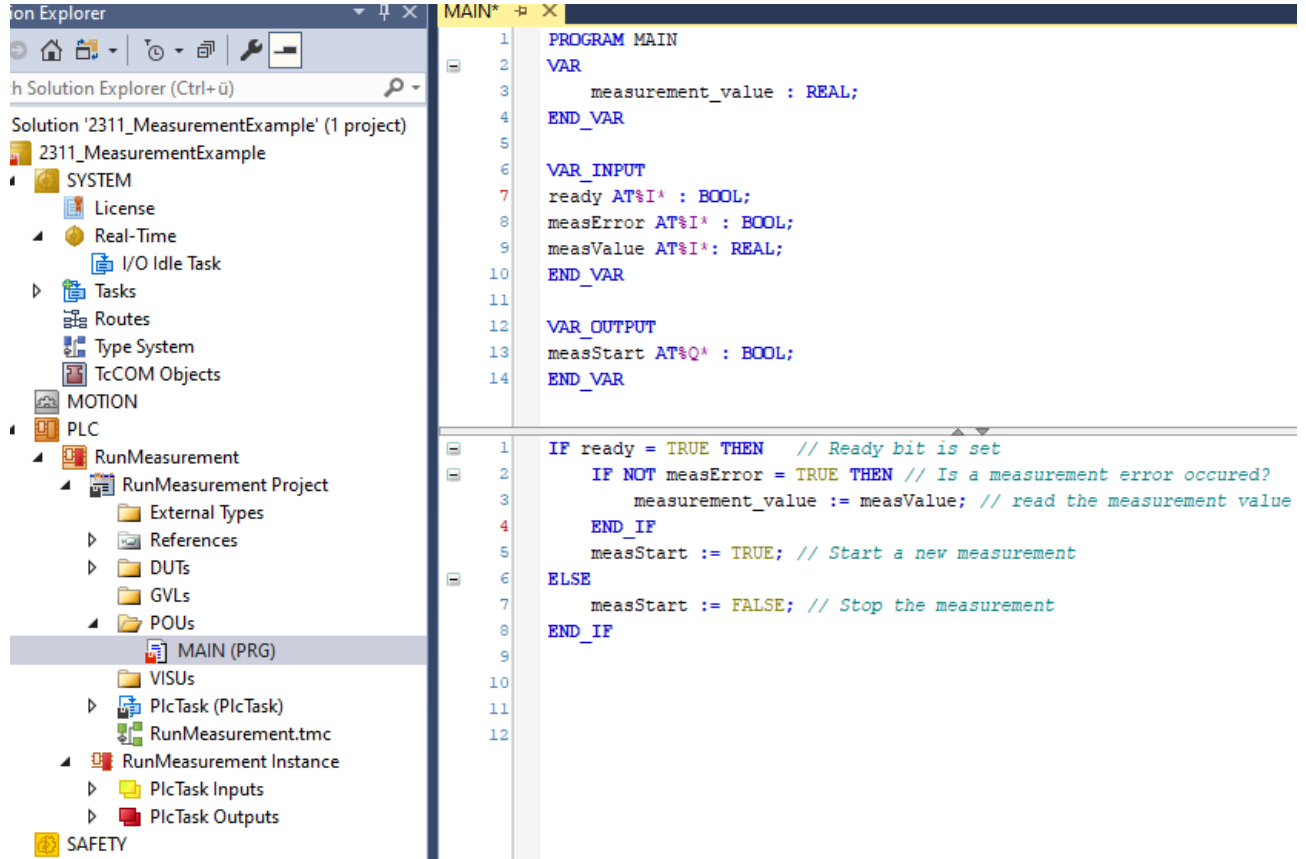


- Next, open the **MAIN (PRG)** file from **PLC** → **RunMeasurement** → **POUs** with double click on it:



## Example 1: Reading and Writing of PDOs

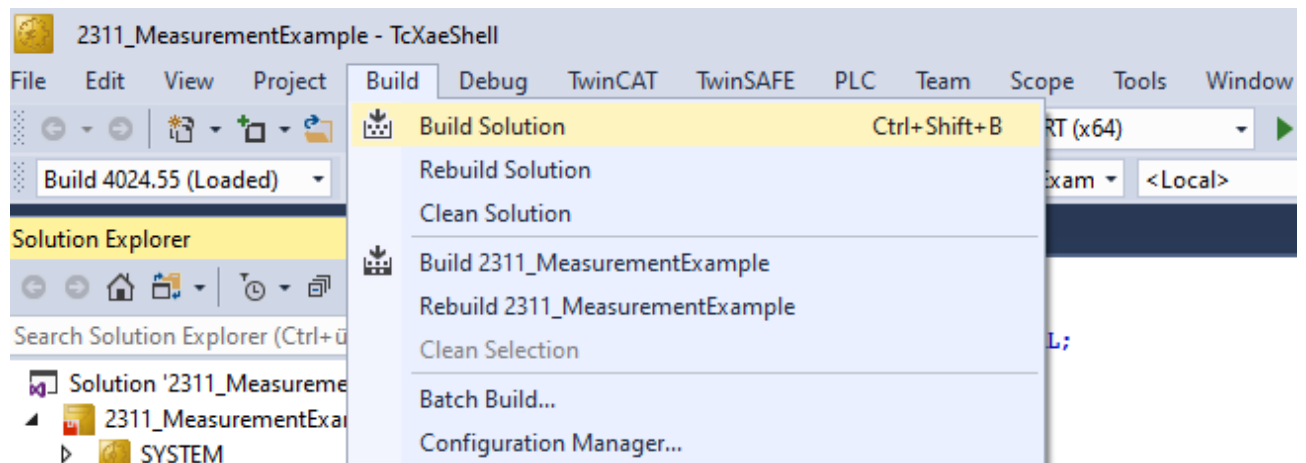
- Type in the following text in the **MAIN** block



```

1 PROGRAM MAIN
2 VAR
3     measurement_value : REAL;
4 END_VAR
5
6 VAR_INPUT
7     ready AT%I* : BOOL;
8     measError AT%I* : BOOL;
9     measValue AT%I* : REAL;
10 END_VAR
11
12 VAR_OUTPUT
13     measStart AT%Q* : BOOL;
14 END_VAR
15
16 IF ready = TRUE THEN // Ready bit is set
17     IF NOT measError = TRUE THEN // Is a measurement error occurred?
18         measurement_value := measValue; // read the measurement value
19     END_IF
20     measStart := TRUE; // Start a new measurement
21 ELSE
22     measStart := FALSE; // Stop the measurement
23 END_IF
    
```

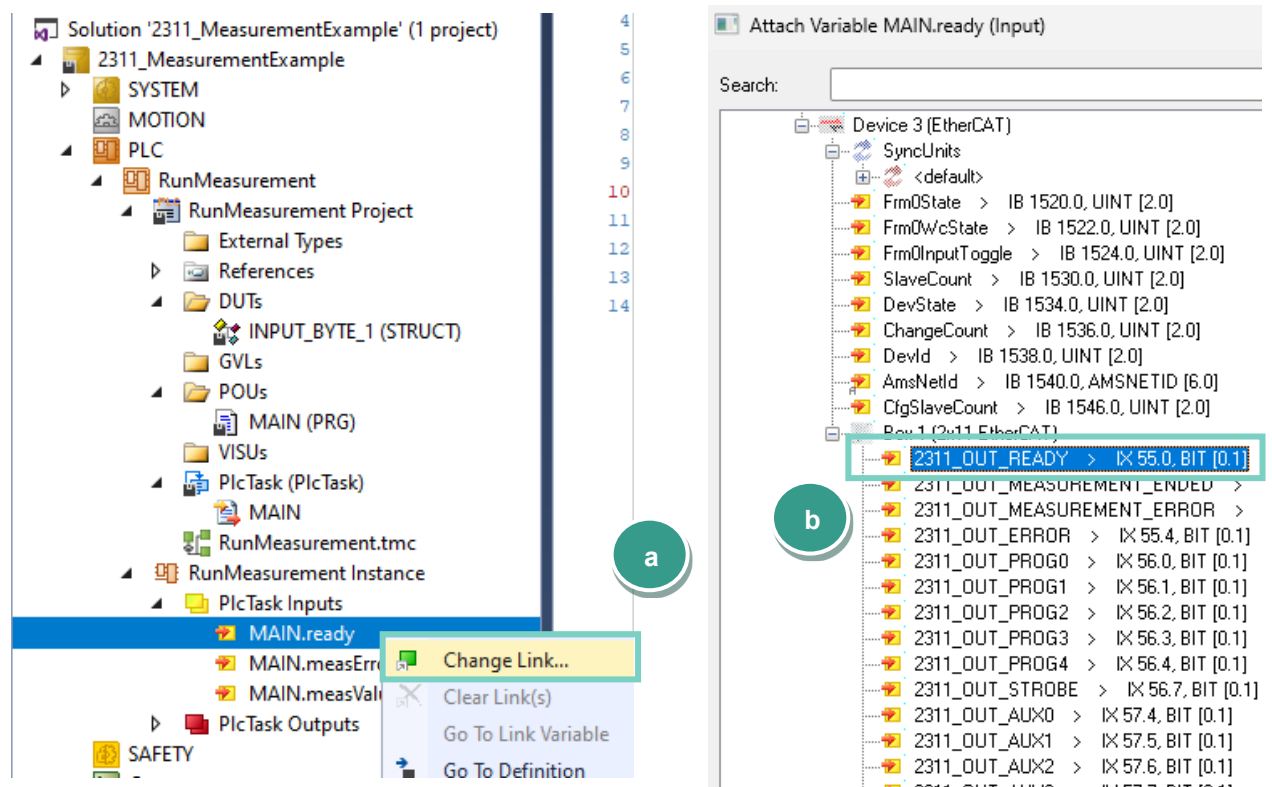
- Goto **Build** → **Build Solution**



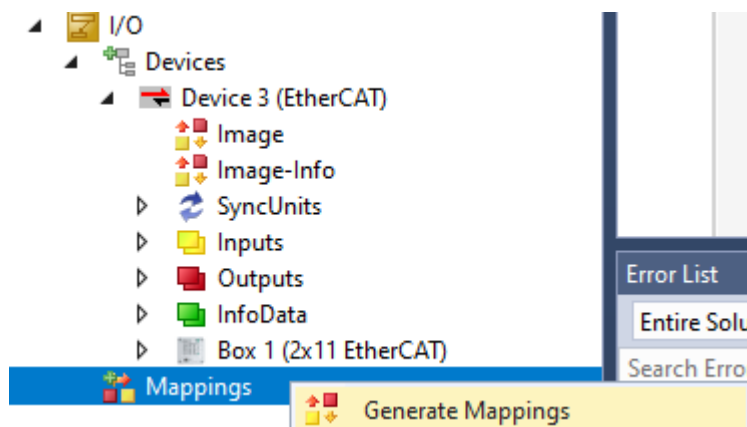
- Assign the **input** and **output** variables to the corresponded PDOs with the right-click on a variable and select **Change Link...**(a) from the context menu, select a corresponded PDO (b)

**Assignment:**

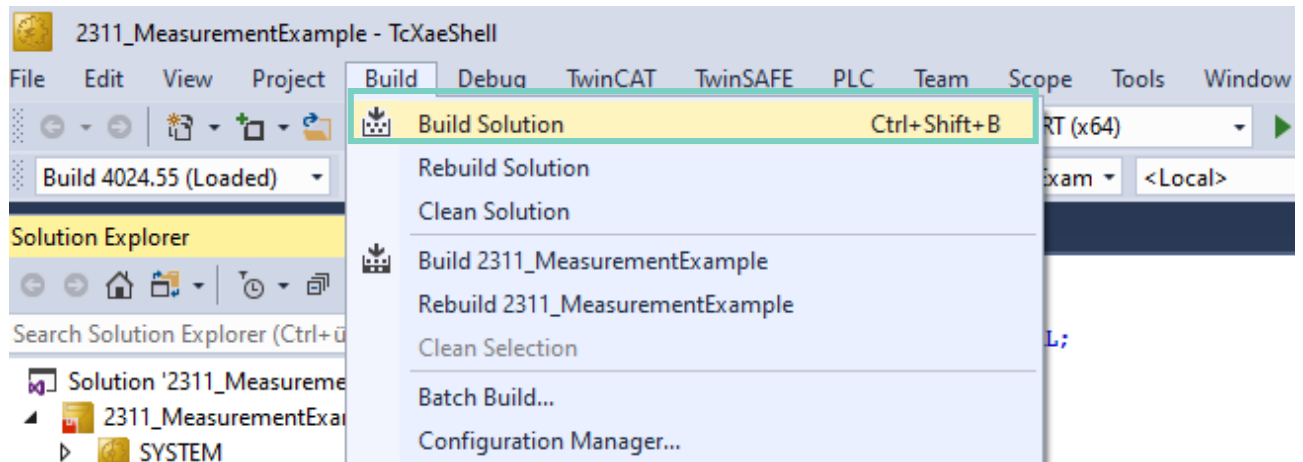
MAIN.ready → 2311\_OUT\_READY  
 MAIN.measError → 2311\_OUT\_MEASUREMENT\_ERROR  
 MAIN.measValue → 2311\_OUT\_MEAS\_VALUE  
 MAIN.measStart → 2311\_IN\_MEAS\_START



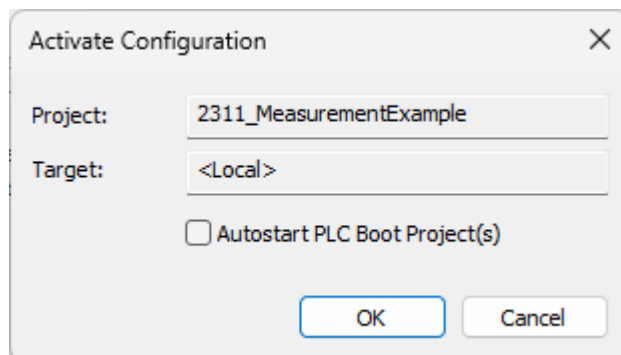
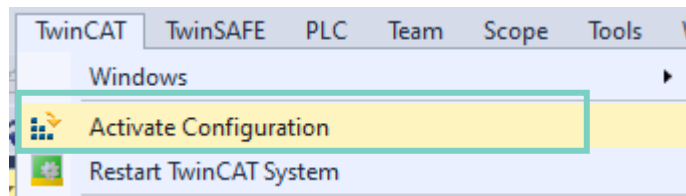
- Right-click **Mappings** → **Generate Mapping**:



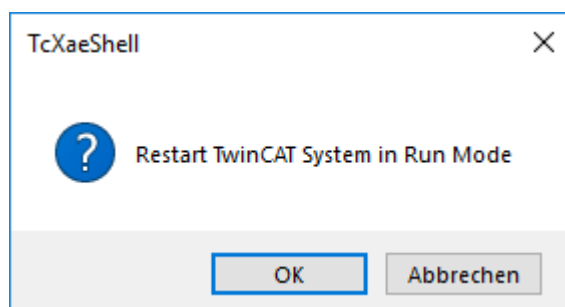
Goto **Build** → **Build Soution** to build the project:



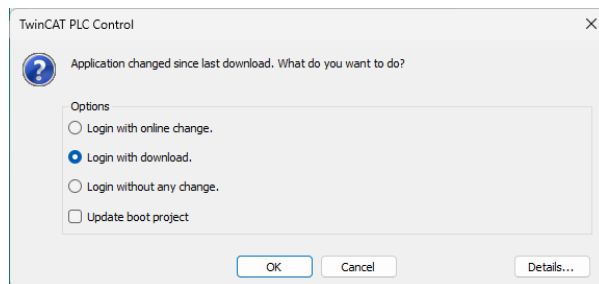
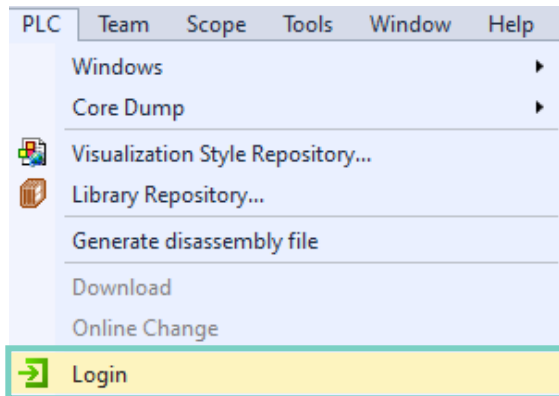
➤ Activate configuration via **TwinCAT** → **Activate Configuration**



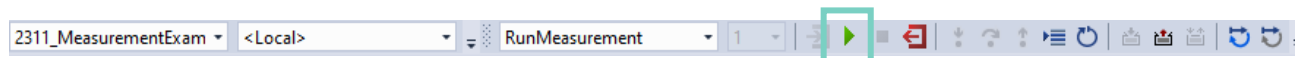
➤ Confirm starting in **Run Mode**:



- Goto **PLC** → **Login**: and if asked, confirm that program should be downloaded into the controller



- Press the F5 key or click on the green start symbol to start the program execution



When the resistomat performs a new measurement (*measStart* = TRUE) it resets its output **ready**, PLC stops after that the measurement by setting *measStart* = FALSE. The 2311 **ready** output becomes TRUE again and when no error occurs (*measError* = FALSE) the measurement result can be copied into variable *measurement\_value*. When the measurement value has been read, a new measurement will be performed and so on. Measurement value will be displayed in variable window:

Expression	Type	Value	Prepared value	Address
measurement_value	REAL	0.100039274		
ready	BOOL	TRUE		%I*
measError	BOOL	FALSE		%I*
measValue	REAL	0.100039274		%I*
measStart	BOOL	TRUE		%Q*

```

1 IF ready TRUE = TRUE THEN // Ready bit is set
2   IF NOT measError FALSE = TRUE THEN // Is a measurement error occurred?
3     measurement_value 0.1 := measValue 0.1; // read the measurement
4   END_IF
5   measStart TRUE := TRUE; // Start a new measurement
6 ELSE
7   measStart TRUE := FALSE; // Stop the measurement
8 END_IF
  
```

**Note:** If measurement does not start: please make sure that **Control via EtherCAT** is selected in device

## 5. Further Examples

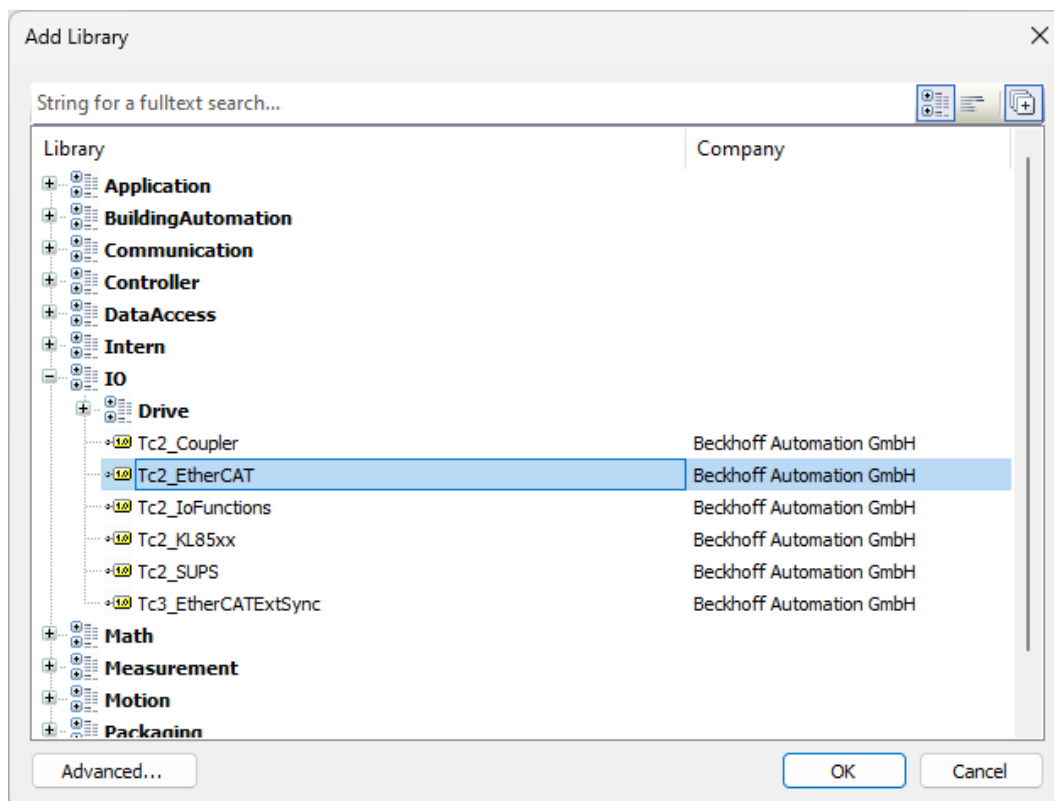
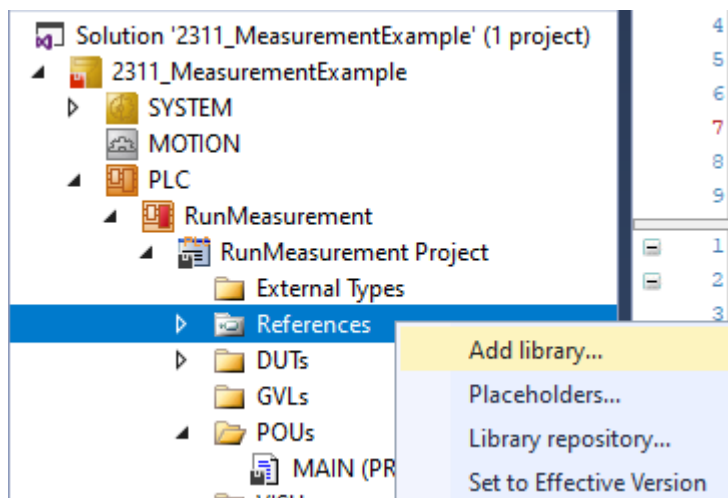
In this chapter, we will perform read & write operations on SDO (**S**ervice **D**ata **O**bjects). These are described in section 8 *SDO – Service Data Objects* of the 2311 EtherCAT operation manual.

### 5.1 Read and Write of 'real' data types

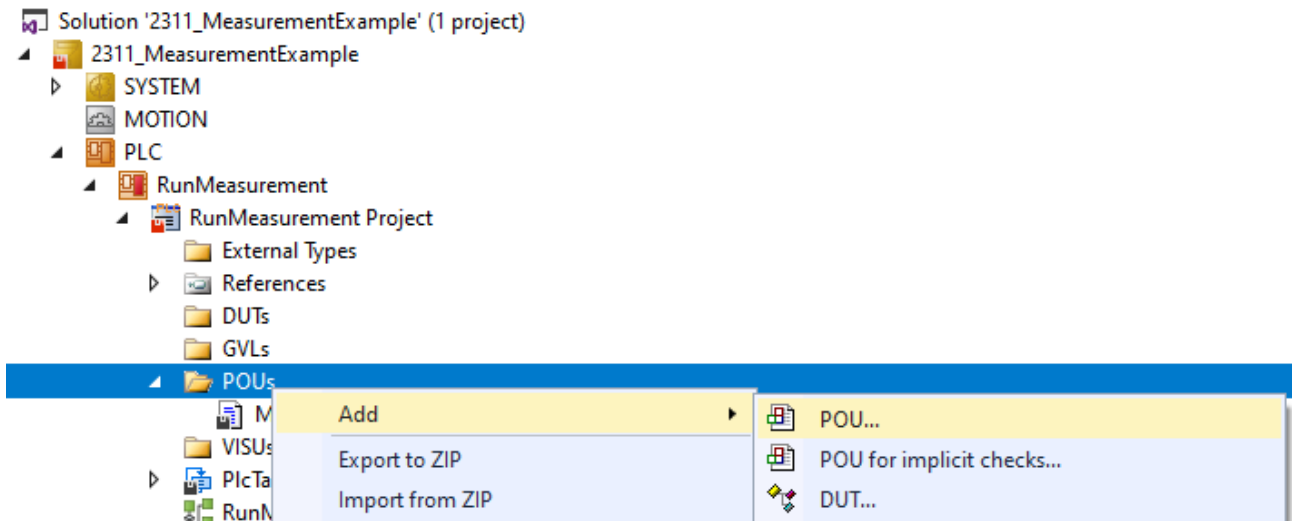
**Example 2:** Set and Get the reference temperature value (Index 0x2045, Subindex 12)

This example shows you how to write and read the reference temperature value.

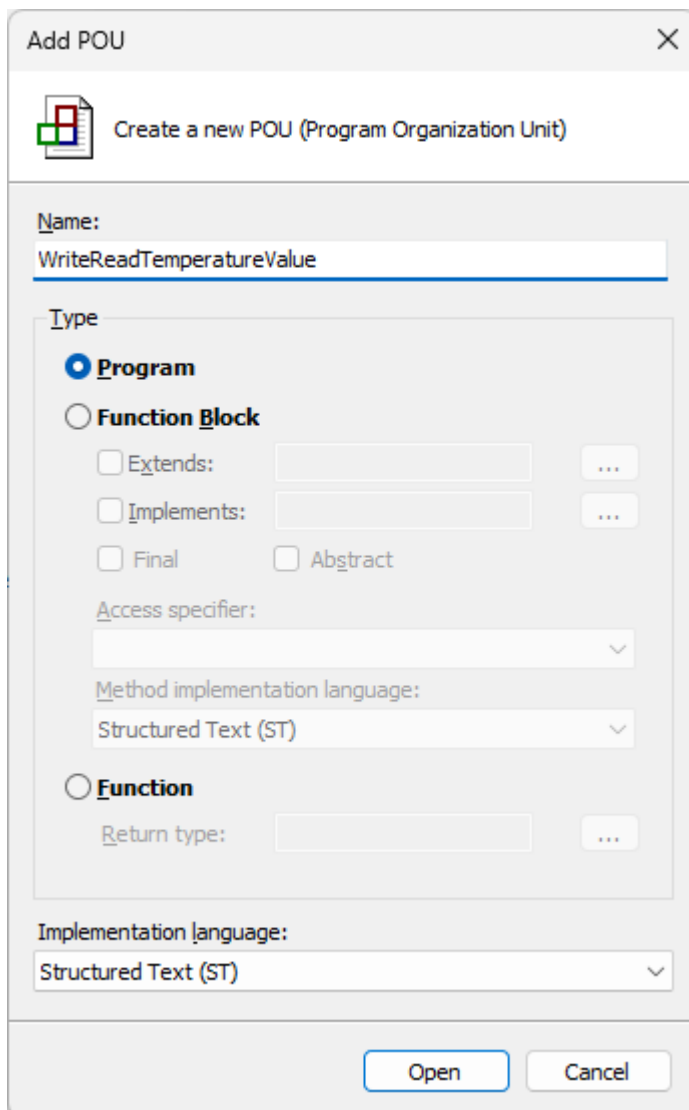
- Add the **Tc2\_EtherCAT** library to your project to be able to use *FB\_EcCoESdoRead* and *FB\_EcCoESdoWrite* function blocks via **References** → **Add library**



- Add a new **POU** (Program Organization Unit)



- Rename it to **WriteReadTemperatureValue** and click **OK**:





- Insert the call of the **WriteReadTemperatureValue** in the **MAIN** POU:

```

3      IF NOT measError = TRUE THEN // Is a measurement error occurred?
4          measurement_value := measValue; // read the measurement value
5      END_IF
6      measStart := TRUE; // Start a new measurement
7  ELSE
8      measStart := FALSE; // Stop the measurement
9  END_IF
10  (*)
11
12  WriteReadTemperatureValue();

```

- Type in the following code into the created **WriteReadTemperatureValue** POU

#### Source code:

```

PROGRAM WriteReadTemperatureValue
VAR
    fbSdoWrite    : FB_EcCoESdoWrite;
    fbSdoRead     : FB_EcCoESdoRead;
    sNetId        : T_AmsNetId := '192.168.19.1.4.1'; // see note 1 below
    nSlaveAddr    : UINT := 1001; // see note 2 below
    nIndex        : WORD := 16#2045; // CoE Object - Temperature value
    nSubIndex     : BYTE := 12; // is always 0
    fTemperature  : REAL := 23.5; // data to be written to 2311
    bExecute      : BOOL := TRUE;
    bError        : BOOL;
    nErrId       : UDINT;
END_VAR

fbSdoWrite(
    sNetId      := sNetId,
    nSlaveAddr  := nSlaveAddr,
    nIndex      := nIndex,
    nSubIndex   := nSubIndex,
    pSrcBuf     := ADR(fTemperature),
    cbBufLen    := SIZEOF(fTemperature),
    bExecute    := bExecute
);

IF NOT fbSdoWrite.bBusy THEN
    bExecute := FALSE;
    IF NOT bError THEN (* write successful *)
        bError := FALSE;
        nErrId := 0;
    ELSE
        (* write failed *)
        bError := fbSdoWrite.bError;
        nErrId := fbSdoWrite.nErrId;
    END_IF

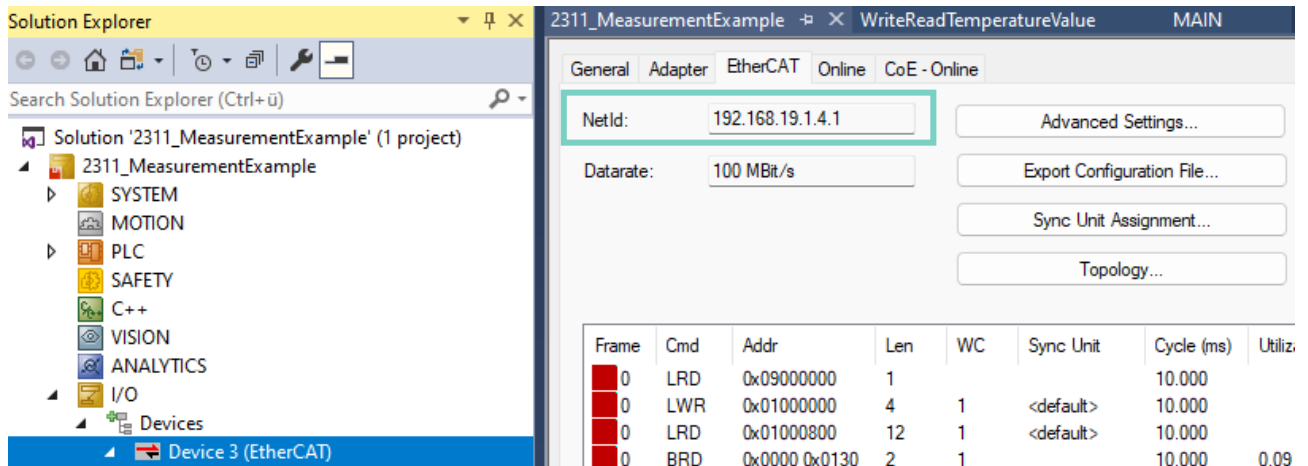
    fbSdoWrite(bExecute := FALSE);
END_IF

fTemperature := 0.0;

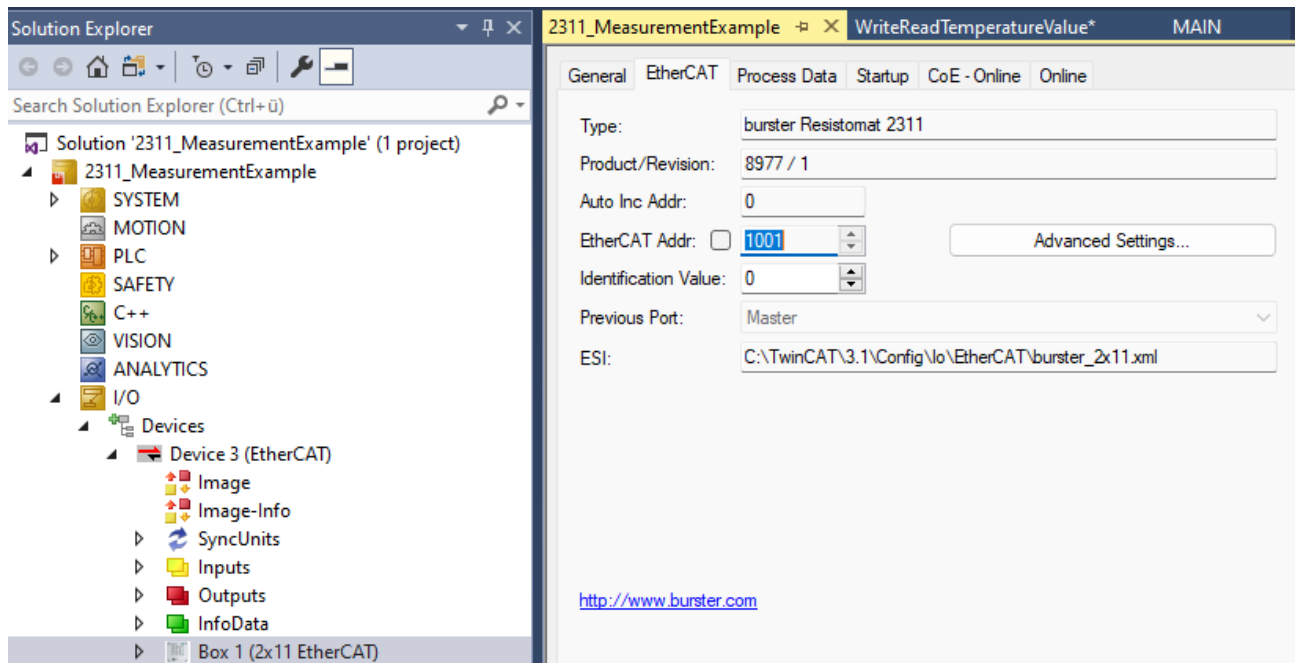
```

```
fbSdoRead(sNetId:= sNetId,nSlaveAddr :=nSlaveAddr, nIndex:=nIndex, nSubIndex :=nSubIndex,
pDstBuf:= ADR(fTemperature), cbBufLen:=SIZEOF(fTemperature), bExecute:=TRUE);
bError:=fbSdoRead.bError;
nErrId:=fbSdoRead.nErrId;
```

**Note 1:** You will find the *NetId* if you click your EtherCAT master device in the project tree and select the tab **EtherCAT**:



**Note 2:** You will find the EtherCAT slave address if you click the 2311 device in the project tree and select the tab **EtherCAT**:



- Build the project via **Build** → **Build Solution**, click on the **Login** symbol and set a break point (F9) in the first code line:

```

1  ● fbSdoWrite (
2      sNetId '192.168.19 ▶           := sNetId '192.168.19 ▶,
3      nSlaveAddr 1001 := nSlaveAddr 1001,
4      nIndex 8261           := nIndex 8261,
5      nSubIndex 12           := nSubIndex 12,
6      pSrcBuf 18446689045179605788 := ADR(fTemperature 23.5),
7      cbBufLen 4             := SIZEOF(fTemperature 23.5),
8      bExecute TRUE          := bExecute TRUE
9  );

```

- Start the program execution with the **F5** key or via **PLC** → **Start** and go step for step (**F10**) through the whole program until you reach the last line. Check if the written and read values are identical:

```

26 ● fTemperature 23.5 := 0.0;
27
28 ● fbSdoRead(sNetId '192.168.19 ▶ := sNetId '192.168.19 ▶, nSlaveAddr 1001 := nSlaveAddr 1001,
29 ● bError FALSE := fbSdoRead.bError FALSE;
30 ● nErrId 0 := fbSdoRead.nErrId 0;
31 ➔ RETURN

```

## 5.2 Read of 'string' data types

**Example 3:** Read the serial number of 2311 (Index 0x2030, Subindex 11):

- Create a new POU as described above and name it **ReadSerial**:

The screenshot shows a dialog box titled 'Add POU' with a close button (X) in the top right corner. Below the title bar is a document icon and the text 'Create a new POU (Program Organization Unit)'. There are two input fields: 'Name:' containing the text 'ReadSeial' and 'Type:' with a radio button selected for 'Program'.

- Write or copy the following source code into the new POU:

```

PROGRAM ReadSerial
VAR
    fbSdoRead : FB_EcCoESdoRead;
    sNetId    : T_AmsNetId := '192.168.19.1.4.1'; // see note 1 in the previous section
    nSlaveAddr : UINT := 1001; // see note 2 in the previous section
    nIndex     : WORD := 16#2030; // CoE Object – Serial number
    nSubIndex  : BYTE := 11; // is always 0
    abSerial   : STRING;
    bExecute   : BOOL := TRUE;
    bError     : BOOL;
    nErrId     : UDINT;

```

```

END_VAR

fbSdoRead(sNetId:= sNetId,
          nSlaveAddr :=nSlaveAddr,
          nIndex:=nIndex,
          nSubIndex :=nSubIndex,
          pDstBuf:= ADR(abSerial),
          cbBufLen:=11,
          bExecute:=TRUE);

bError:=fbSdoRead.bError;
nErrId:=fbSdoRead.nErrId;

```

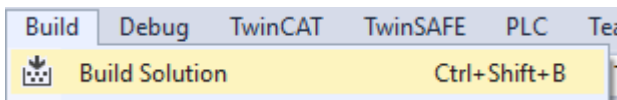
- Insert a call for the POU in the **MAIN** block:

```

END_VAR
5
6 ReadSerial ();

```

- Build the Project via **Build** → **Build Solution**:



- Log in **PLC** → **Login**, set a break point to the last line and click **PLC** → **Start (F5)** to run the program

```

1 fbSdoRead(sNetId: '192.168.19' := sNetId: '192.168.19',
2           nSlaveAddr: 1001 := nSlaveAddr: 1001,
3           nIndex: 8240 := nIndex: 8240,
4           nSubIndex: 11 := nSubIndex: 11,
5           pDstBuf: 18446699045179604560 := ADR(abSerial: '123456789A'),
6           cbBufLen: 20 := 20,
7           bExecute: TRUE := TRUE);
8
9 bError: FALSE := fbSdoRead.bError: FALSE;
10 nErrId: 0 := fbSdoRead.nErrId: 0;
11 RETURN

```

The serial number is read into the variable 'abSerial'