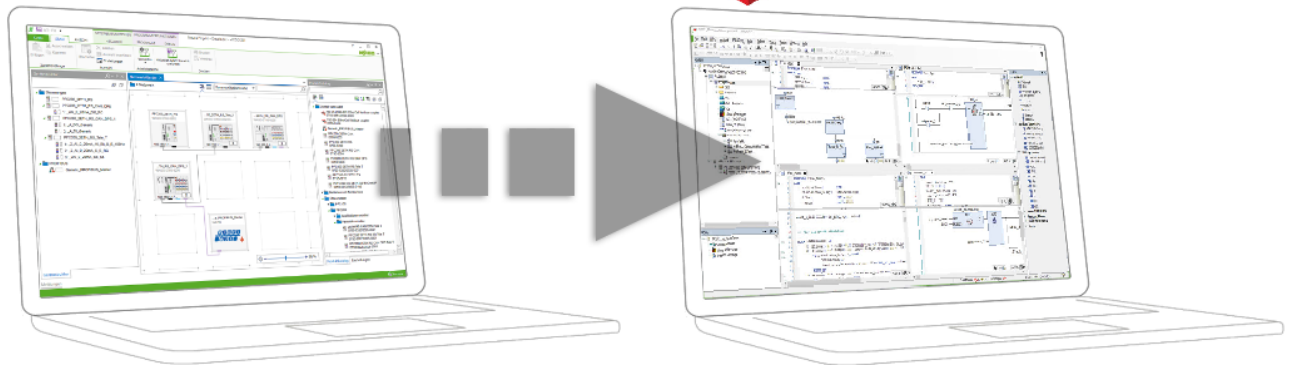


# Migration Guide

Migration from *e!COCKPIT* projects to CODESYS V3.5

 **COCKPIT**

 **CODESYS**



# Table of Contents

<b>Terms .....</b>	<b>3</b>
<b>Overview .....</b>	<b>4</b>
<b>General Information .....</b>	<b>5</b>
<b>Installing and Launching.....</b>	<b>6</b>
4.1 Updating a Project in e!COCKPIT .....	7
4.2 Updating Firmware for Use in CODESYS .....	7
4.3 Installing CODESYS.....	8
4.4 Installing CODESYS Add-ons .....	8
4.5 Installing WAGO Components collectively via Package Files.....	8
4.6 Installing WAGO Device Descriptions .....	8
4.7 Installing WAGO Libraries .....	9
4.8 Installing WAGO Visualization Styles.....	9
4.9 Installing the “WAGO Licensing” Add-on .....	9
4.10 Transferring Licenses with WAGOupload .....	10
4.11 Installing and Using WAGO I/O-CHECK.....	10
<b>Project Conversion .....</b>	<b>11</b>
5.1 Project conversion using the example of Modbus.....	12
5.2 Project conversion using the example of CANopen .....	16
5.3 Project Conversion on the Example of EtherNet/IP .....	18
5.4 Project Conversion on the Example of EtherCAT .....	22
<b>Differences in Workflows and Functions.....</b>	<b>25</b>
<b>Fieldbus Configuration in CODESYS.....</b>	<b>29</b>
7.1 Modbus – First Steps .....	29
7.2 CANopen – First Steps.....	30
7.3 EtherNet/IP – First Steps.....	32
7.4 EtherCAT – First Steps .....	35
<b>Example Project: Creating a New Modbus Project in CODESYS .....</b>	<b>37</b>

# Terms

This document is intended as a guide for the migration of **e!COCKPIT** projects to CODESYS.

The software must only be installed and operated in accordance with the operating instructions. Knowledge of the operating instructions is required for proper use.

The legal information in the document applies accordingly:

 **Product Manual e!COCKPIT** (also available as  **e!COCKPIT Online help**).

You can find more information on **e!COCKPIT** at:  **[www.wago.com](http://www.wago.com)**.

You can find more information on CODESYS V3.5 at:  **[help.codesys.com](http://help.codesys.com)**.

## Limitation of Liability

This documentation describes the use of various hardware and software components in specific example applications. The components may represent products or parts of products from different manufacturers. The respective operating instructions from the manufacturers apply exclusively with regard to intended and safe use of the products. The manufacturers of the respective products are solely responsible for the contents of these instructions.

The sample applications described in this documentation represent concepts, that is, technically feasible application. Whether these concepts can actually be implemented depends on various boundary conditions. For example, different versions of the hardware or software components can require different handling than that described here. Therefore, the descriptions contained in this documentation do not form the basis for assertion of a certain product characteristic.

Responsibility for safe use of a specific software or hardware configuration lies with the party that produces or operates the configuration. This also applies when one of the concepts described in this document was used for implementation of the configuration.

WAGO GmbH & Co. KG is not liable for any actual implementation of the concepts.

# Overview

The CODESYS V3.5 software represents an additional engineering solution alongside **e!COCKPIT**, which is based on an open platform and can already be used for many WAGO controllers.

This migration guide provides an overview of all the important issues for migrating **e!COCKPIT** projects to CODESYS. **e!COCKPIT** is based on CODESYS, so the IEC programming basis is identical for both software products. However, **e!COCKPIT** differs in terms of the handling and integration of software or device components. This guide offers assistance and tips and gives recommendations that must be followed when switching from **e!COCKPIT** to CODESYS in order to continue using projects in CODESYS or to create new ones.

# General Information

## Firmware and Compatibility

Conversion of projects from **e!COCKPIT** to CODESYS is supported for the following devices: 750-8210, 750-8211, 750-8212, 750-8213, 750-8214, 750-8215 and 750-8217 (and all variants of these devices).

Before conversion, **e!COCKPIT** projects must be updated in **e!COCKPIT** to the latest version (**e!COCKPIT** version 1.11, firmware version 22, compiler version 3.5.17); see [🔗 Updating a Project in e!COCKPIT \[▶ 7\]](#).

Devices that are used in the project must be updated to firmware version 23 through a firmware update. CODESYS Runtime is supported on the firmware side for this version and above (see [🔗 Updating Firmware for Use in CODESYS \[▶ 7\]](#)).

The WAGO website offers a 64 bit version of CODESYS V3.5 for download (recommended). If a 32 bit version has already been installed via CODESYS, it can also be used.

Parallel operation of **e!COCKPIT** and CODESYS on one PC is possible.

## Licenses

CODESYS itself can be used free of charge and without a license. However, some additional components require a license.

Licenses for fee-based **CODESYS components** that you have already purchased for **e!COCKPIT** (for example, the “Profiler,” “Static Analysis,” “UML” or “SVN” add-ons) cannot be reused in CODESYS. Licenses cannot be transferred due to the differences between the two software products’ license models. However, CODESYS offers bundles of different components that you can purchase together in one software package. You can transfer and synchronize these licenses using CODESYS mechanisms (“Tools” tab > “License Manager”/“License Repository”).

The required licenses for fee-based **WAGO components** (e.g., certain WAGO libraries, telecontrol configurators or WAGO Solution Builder) are indicated in CODESYS with the help of the CODESYS “WAGO Licensing” add-on. This add-on is included in Software Packages 1 and 2 for the installation files (see [🔗 Installing and Launching \[▶ 6\]](#)). To transfer the required WAGO licenses, use WAGOupload.

# Installing and Launching

The following components are available for starting the migration of your project and working with CODESYS. Download the software packages you need from the links provided:

- **Software Package 1 – Base Installation Including CODESYS**

- 🌐 **CODESYS Development System V3 Including WAGO Device Support Package**  
(Codesys3.5.x.x\_inclusive\_WagoDSP\_vx.x.x.x.zip)

Software package that includes the CODESYS 3.5 software itself. Also included are firmware, device descriptions, libraries, release notes, visualization styles, add-ons, such as Telecontrol and WAGO Solution Builder as well as the CODESYS “WAGO Licensing” add-on.

- **Software Package 2 – Base Installation without CODESYS**

- 🌐 **WAGO Device Support Package for CODESYS Development System V3**  
(WagoDSP\_vx.x.x.x.zip)

Software package that contains all the components from Software Package 1, but without CODESYS itself. Select this package if you have already installed CODESYS. We recommend downloading CODESYS from the WAGO website (see Software Package 1) to ensure compatibility between the components used.

- **Software Package 3 – Optional Add-ons (License-Based/Fee-Based)**

- 🌐 **CODESYS Professional Developer Edition**

Software package that includes CODESYS add-ons, such as Profiler, UML, Static Analysis or SVN, in case you want to use them.

- **WAGOUpload Software**

- 🌐 **WAGOUpload**

You need the WAGOUpload software (or alternatively, *e!COCKPIT* or an SD card) to update the firmware.

You also need WAGOUpload in order to transfer licenses from WAGO components to devices.

- **WAGO I/O-CHECK Software (Fee-Based)**

- 🌐 **WAGO I/O-CHECK**

You need the WAGO I/O-CHECK software in order to configure I/O modules.

## 4.1 Updating a Project in e!COCKPIT

You must first make sure that your project works in **e!COCKPIT** with firmware version 22. Since this firmware already uses the same compiler version (3.5.17) as you will then use to perform the conversion to CODESYS, this preliminary test can be used to minimize sources of error.

- ✓ You need to have opened your project in **e!COCKPIT**.
- 1. In the Backstage view > “Updates & Add-ons” page, check whether you are using the latest **e!COCKPIT** version 1.11. If not, select it and install the update.
- 2. Update the firmware of your devices to firmware version 22, if you have not already. For a description of the firmware changes, see [📖 Downloading the Firmware](#) and [📖 Replacing the Firmware](#) in the **e!COCKPIT** manual.
- 3. Check whether the updated project works as desired.

## 4.2 Updating Firmware for Use in CODESYS

- ✓ You need to have downloaded and unpacked Software Package 1 or 2. Both packages contain the firmware in the form of \*.wup and \*.img files.  
**Note:** Firmware version 23 is required for using WAGO devices with CODESYS Runtime (see [📖 General Information \[▶ 5\]](#)).
- Load the firmware onto your device(s) either with WAGOupload (\*.wup) or via SD card (\*.img):

### Updating Firmware with WAGOupload

1. Click **[Update Firmware]** in WAGOupload. A wizard guides you through the different steps of the firmware update.
2. Enter the IP address of the controller and select **[Find Controller]**.
3. Check the box to the left of the desired controller.
4. Select the update file (\*.wup).
5. Perform the firmware update.

or

### Updating Firmware via SD Card

1. Copy the firmware file (\*.img) to the SD card, for instance using the “Win32 Disk Imager” or “UNetbootin” freeware tool.
2. Switch the controller off, insert the memory card into the SD slot and switch the controller back on.  
 ⇒ The controller boots from the memory card with the firmware image to be installed.
3. After the controller boots up, open the WBM “Administration” page > “Create Boot Image” (you may have to change the IP address temporarily to do so).
4. To create a new boot image on the internal memory, click **[Start Copy]**.

5. After the process is complete, turn the controller off, remove the memory card and turn the controller back on.  
⇒ The controller now starts with the new firmware version.

### 4.3 Installing CODESYS

- ✓ You need to have downloaded and unpacked Software Package 1.

  1. Install CODESYS.
  2. Open CODESYS and install additional components from CODESYS as described below.

### 4.4 Installing CODESYS Add-ons

- ✓ You need to have downloaded and unpacked Software Package 3 (CODESYS Professional Developer Edition).

  1. On the “Tools” tab, click **CODESYS Installer**.  
**Note:** If you cannot find the CODESYS installer (for the relevant the CODESYS version/update) under “Tools,” open it alternatively via the Start menu > “CODESYS” > “CODESYS Installer.”
  2. In the lower “Add-ons” section, click **[Install Files]** and select the downloaded package files.  
**Note:** You must be logged into CODESYS as an administrator to install these files. If you are not, the button is grayed out.

### 4.5 Installing WAGO Components collectively via Package Files

- ✓ You have downloaded and unpacked Software Package 1 or 2.  
The package contains further folders with \*.package files, which contain all necessary components. The components of each package (e.g. several device descriptions) can be installed together in only one step.

  1. On the “Tools” tab, click **CODESYS Installer ...** (via the “Tools” tab or the Start menu > “CODESYS” > “CODESYS Installer”).
  2. In the lower section, click **[Install File]**.  
**Note:** You must be logged into CODESYS as an administrator to install these files. If you are not, the button is grayed out.
  3. Select the respective \*.package file and click **[Open]**.

⇒ All components contained in the \*.package file are installed.  
Installed device descriptions/devices are now automatically displayed for you to select when you create a new project and select “Standard Project” as the template. I/O modules resulting from the installed device descriptions are also displayed when you search for I/O modules (right-click on “kbus” > **Scan for Devices ...**).

### 4.6 Installing WAGO Device Descriptions

If you need devices/device descriptions that are not included in Software Package 1 or 2, you can also load them individually later.



**Tip:** If you do not have the device description file, visit WAGO's website and search for your device (e.g., 750-337). In the lower section of the product page under "Downloads" > "Device Files," you can download the file.

1. On the "Tools" tab, click **Device Repository ....**
  2. Click **[Install ...]**.
  3. Select the device description.
  4. Click **[Open]**.
- ⇒ The device description has been installed.

## 4.7 Installing WAGO Libraries

If you need libraries that are not included in Software Package 1 or 2, you can also load them individually later:

1. On the "Tools" tab, click **Library Repository ....**
  2. Click **[Install ...]**.
  3. Select the library.
  4. Click **[Open]**.
- ⇒ The library is installed.

**Tip:** If you want to use a library that you have already used in *e!COCKPIT*, for example, you can export the library from *e!COCKPIT* as follows: Open the library manager in *e!COCKPIT*. In the manager, right-click the desired library and select **Export Library**. This gives you a compiled library that you can use in CODESYS.

**Note:** If you use a licensed library, e.g. the library "WagoAppPowerPlantcontrol", then you must use the latest version of these libraries. Older versions of licensed libraries from *e!COCKPIT*, which may be further used in a CODESYS project, can lead to errors when compiling the project.

## 4.8 Installing WAGO Visualization Styles

If you need visualization libraries that are not included in Software Package 1 or 2, you can also load them individually later:

1. On the "Tools" tab, click **Visualization Style Repository ....**
  2. Click **[Install ...]**.
  3. Select the visualization styles.
  4. Click **[Open]**.
- ⇒ The visualization styles are installed and available through the visualization manager.

## 4.9 Installing the "WAGO Licensing" Add-on

You need the add-on if you want to use licensed WAGO libraries or fieldbus functions from WAGO in CODESYS (e.g., certain WAGO libraries, the WAGO telecontrol configurator or WAGO Solution Builder). The "WAGO Licensing" add-on indicates the required licenses in CODESYS.

- ✓ You need to have downloaded and unpacked Software Package 1 or 2. The “WAGO Licensing” add-on needs to be included here under the name “WagoLicensing-x.x-signed.package”.

You also need to have installed the WAGOupload software.

1. Open the **CODESYS Installer** .... (via the “Tools” tab or the Start menu > “CODESYS” > “CODESYS Installer”).
2. Click **[Install Files]** in the lower area.
3. Select the \*.package file for the add-on.
4. To install the add-on, click **[Open]**.
  - ⇒ The add-on shows you which licenses you need for your project.

### 4.10 Transferring Licenses with WAGOupload

Use WAGOupload to transfer the required licenses the “WAGO Licensing” add-on indicates to the device:

1. Click **[Manage Licenses]** in the WAGOupload main menu.
2. Click **[Add]** and enter your license data.
3. Go **[Back to the Main Menu]**.
4. Click **[Synchronize Licenses]**.
5. Enter the IP address of your controller, click **[Find Controller]** and select it from the list.
6. The **[Next]** button takes you to the list of licensed products.
7. Next to the licensed product, select the number of licenses to transfer, click **[Next]** and perform the synchronization.
  - ⇒ The device’s licensing requirements are met.

**Note:** The “WAGO Licensing” add-on only applies to WAGO components; license-based CODESYS components are transferred/synchronized via CODESYS mechanisms (“Tools” tab > “License Manager”/“License Repository”).

### 4.11 Installing and Using WAGO I/O-CHECK

I/O modules are not configured using CODESYS, but rather the WAGO I/O-CHECK software. You can obtain a license key and the CD with the software from the following page: <https://www.wago.com/global/software/wago-i-o-check/p/759-920> (fee-based). If you do not have a CD drive, you can obtain the download version after your purchase upon request from WAGO Support (✉ [support@wago.com](mailto:support@wago.com)).

# Project Conversion

The basic steps in converting the project are as follows. First, read through the entire workflow before you begin.

**Note:** Note that an export of archives from **e!COCKPIT** cannot be used for conversion, because the exported device descriptions are not suitable for CODESYS Runtime, which is integrated in the firmware from firmware version 23. Similarly, certain actions cannot be performed in **e!COCKPIT**, certain actions cannot be performed in CODESYS, and you cannot import/export projects between the two tools multiple times. Perform the conversion to CODESYS only once and perform all further actions there exclusively.

The following initial steps apply equally to all fieldbuses. The conversion steps differ according to the fieldbus used. Note the example conversions for Modbus, CANopen, EtherNet/IP and EtherCAT.

## Preparing the Project for Conversion and Opening It in CODESYS

1. Prepare your project and CODESYS installation as described in [🔗 Installing and Launching \[▶ 6\]](#):
  - Update your **e!COCKPIT** version to 1.11.
  - In **e!COCKPIT**, update your projects to the newest version (firmware version 22, compiler version 3.5.17).
  - Check your application.
  - Update your devices' firmware to firmware version 23 (with WAGOupload or an SD card).
  - Install CODESYS and all required components.
2. Make a backup copy of your project in case data is lost during conversion.
3. Now change your **e!COCKPIT** project's file extension from \*.ecp to \*.project.
4. In CODESYS, click **Open Project** on the File tab and select the project file (\*.project).
  - ⇒ The project appears in the device tree. A red circle with a question mark indicates that device descriptions need to be updated.
  - The "Project Environment" dialog also opens. This dialog shows all the settings related to visualization profiles/symbols and placeholders for which a newer version is available.
5. Update the project settings with **[OK]**.

### Note:

- Libraries may be missing in the project during configuration. Install them afterwards if necessary (see [🔗 Installing WAGO Libraries \[▶ 9\]](#)).
- Device descriptions may also be missing in the project. Install them afterwards if necessary (see [🔗 Installing WAGO Device Descriptions \[▶ 8\]](#)).

## 5.1 Project conversion using the example of Modbus

### Initial state:

In this example, an **e!COCKPIT** project is converted, consisting of a PFC200 (750-8212) as the master with a connected CANopen coupler (750-352) as the slave. There are three I/O modules plugged into the coupler: a 2-channel digital input module (750-400), a 2-channel analog input module (750-469) and a 2-channel analog output module (750-550). You can see the data points of these three I/O modules in the fieldbus configurator under “Local Bus Data Points.” These data points were mapped to master variables by dragging and dropping them.

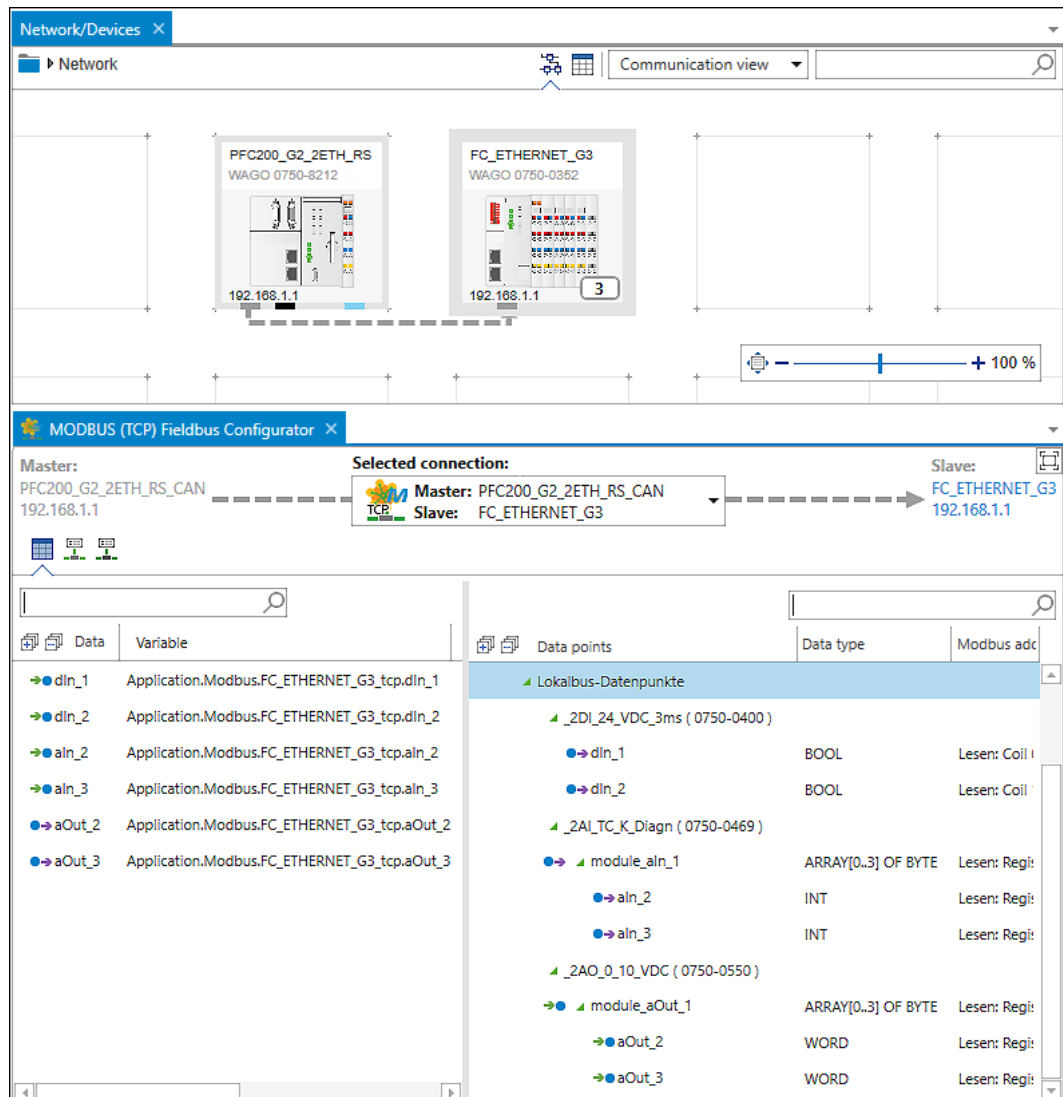


Figure 1: Modbus Project in e!COCKPIT

### In brief:

In the following description, you perform the following steps in CODESYS: You delete the slave device and update the master device. Global Modbus variables are retained. You then recreate the Modbus structure with the master and slave in the device tree and recreate the I/O image (channels, length of input/output data depending on I/O modules). You then reassign the existing global Modbus variables.

1. First delete the slave device. It contains no data you will need.
  - ⇒ The required Modbus variables are generally created globally. In the device tree under “Application” > “Modbus,” you will find a structure that shows these globally stored variables:

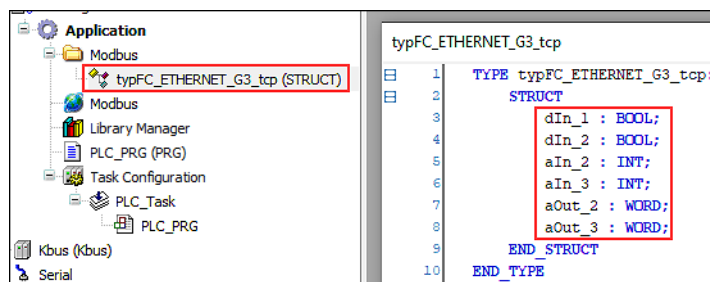


Figure 2: Global Variables via Modbus

- ⇒ For the new slave that you will create, you will reuse these variables and assign them to the input/output data of your I/O modules in CODESYS.
2. First, right-click on the device and select **Update Device ...** to load the device description for the device.
  3. Select your device from the list and click the **[Update Device]** button.
    - ⇒ The fieldbus elements are removed. However, the global variables are retained.
  4. Now recreate the master and the slave. To do so, right-click on the device and select **Insert Device ....**
  5. Leave the dialog open for the next three steps and click **[Insert Device]** to insert the “Ethernet” fieldbus, as well as the master and the slave inserted below it:
    - First select the “Ethernet” fieldbus in the dialog.
    - Click on the new “Ethernet” element in the device tree and select the Modbus master in the dialog.
    - Click on the new “Modbus\_TCP\_Master” element in the device tree and select the Modbus slave in the dialog.

⇒ This procedure creates the following structure in the device tree:

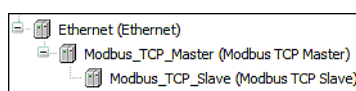


Figure 3: Modbus Master with Modbus Slave

6. Double-click on the slave (“Modbus\_TCP\_Slave” element).
  - ⇒ The configuration opens.
7. Click “Modbus Slave Channel” and click the **[Add Channel ...]** button there on the bottom right.

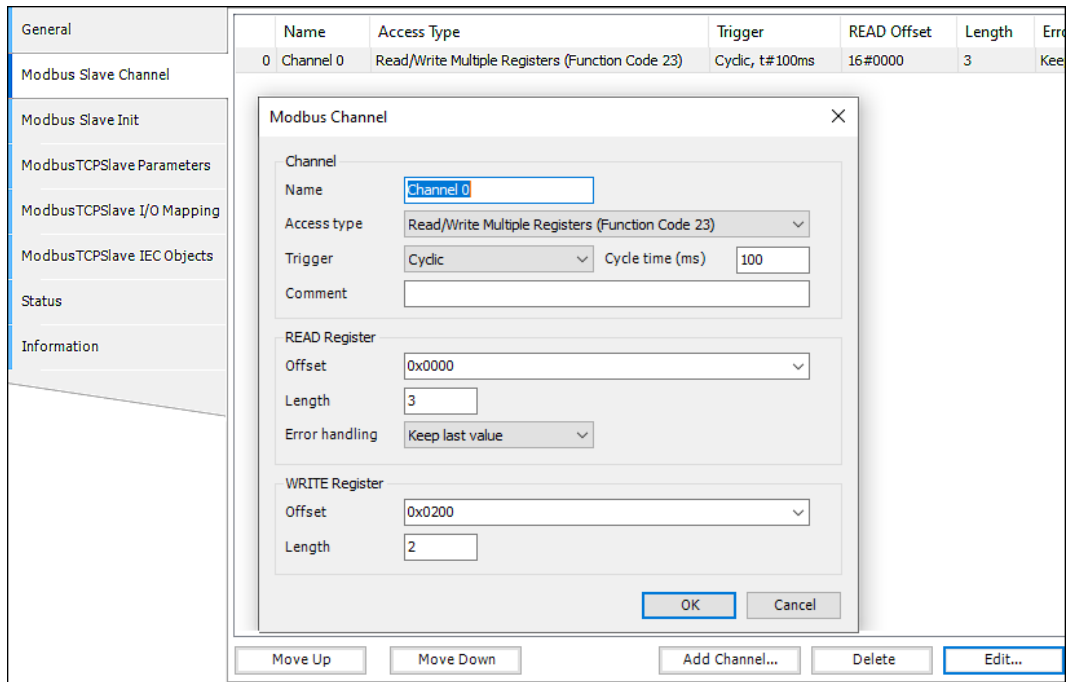


Figure 4: Adding "Modbus Slave Channel"

8. Set the channel, access type and length for reading and writing according to the I/O modules you use.  
Here, for example, the "READ Register" is set to a length of 3 (one 2-channel DI + one 2-channel AI) and the "WRITE Register" to length 2 (2-channel AO).
9. Click "Modbus TCP Slave I/O Image."  
⇒ You now see the channel and variables that were created. You will remap/reassign the global variables of the I/O modules to these.
10. To do so, in the corresponding variable field, click the [...] button that appears to the right to open the Input Assistant.
11. Under "Application" > "Modbus" > "FC\_ETHERNET\_G3\_tcp," you will find the global variables for selection.

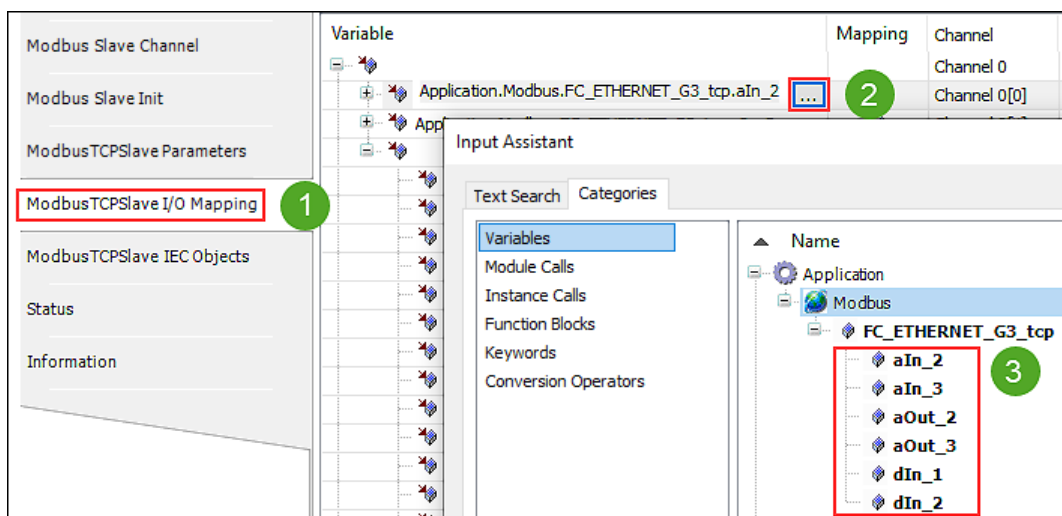


Figure 5: Performing the Mapping

12. Go through these one by one to recreate your slave from the e!COCKPIT project in this way.






























Variable	Mapping	Channel	Address	Type
		Channel 0	%IW1	ARRAY [0..15]
 Application.Modbus.FC_ETHERNET_G3_tcp.aIn_2		Channel 0[0]	%IW1	WORD
 Application.Modbus.FC_ETHERNET_G3_tcp.aIn_3		Channel 0[1]	%IW2	WORD
		Channel 0[2]	%IW3	WORD
 Application.Modbus.FC_ETHERNET_G3_tcp.dIn_1		Bit0	%IX6.0	BOOL
 Application.Modbus.FC_ETHERNET_G3_tcp.dIn_2		Bit1	%IX6.1	BOOL
		Bit2	%IX6.2	BOOL
		Bit3	%IX6.3	BOOL
		Bit4	%IX6.4	BOOL
		Bit5	%IX6.5	BOOL
		Bit6	%IX6.6	BOOL
		Bit7	%IX6.7	BOOL
		Bit8	%IX7.0	BOOL
		Bit9	%IX7.1	BOOL
		Bit10	%IX7.2	BOOL
		Bit11	%IX7.3	BOOL
		Bit12	%IX7.4	BOOL
		Bit13	%IX7.5	BOOL
		Bit14	%IX7.6	BOOL
		Bit15	%IX7.7	BOOL
		Channel 0	%QW0	ARRAY [0..15]
 Application.Modbus.FC_ETHERNET_G3_tcp.aOut_2		Channel 0[0]	%QW0	WORD
 Application.Modbus.FC_ETHERNET_G3_tcp.aOut_3		Channel 0[1]	%QW1	WORD

Figure 6: Assigning Variables

⇒ This completes the conversion.

### Note

**Please note the following restrictions in CODESYS as opposed to e!COCKPIT:**

- Modbus RTU is not supported specifically for PFCs.
- Modbus UDP is not supported (so no multicast requests, for example).
- Specific TCP/IP parameters (Keep Alive, Type of Service) are not supported.
- Function code FC22 (masking of registers) is not supported.
- Nor is function code FC66 (polling of larger amounts of data in one request).
- Device-specific special registers are not supported.
- The MODBUS service is only available when the PLC application is running.

## 5.2 Project conversion using the example of CANopen

### Initial state:

In this example, an **e!COCKPIT** project consisting of a PFC200 (750-8214) as the master with a connected CANopen coupler (750-337) as the slave is converted. For the coupler, a device description (EDS) has been installed in **e!COCKPIT** via the **Backstage View > Product Catalog > [Import Device]**. Three I/O modules are plugged into the coupler: a 4-channel digital input module (750-402), a 2-channel digital output module (750-501) and an 8-channel digital input module (750-449). A program using CAN variables is running on the PFC200.

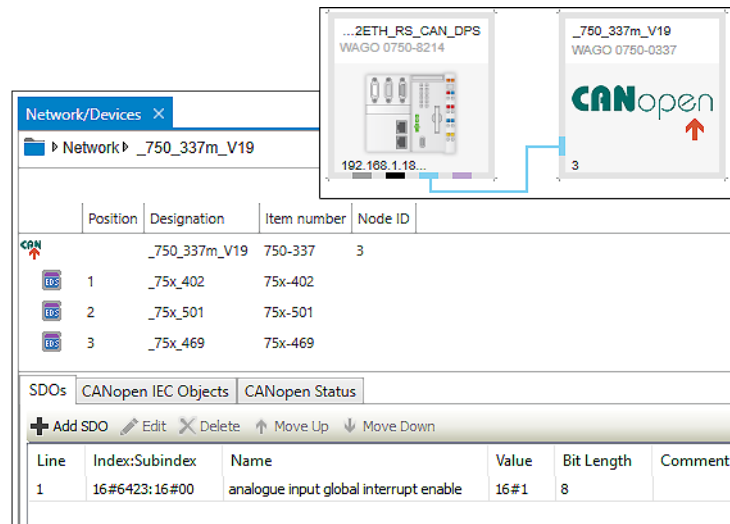


Figure 7: CANopen Project in e!COCKPIT

### In brief:

In the following description, you perform the following steps in CODESYS: You create a copy of the master device to back up the CANopen data. You then update the master device to the current CODESYS device description. The CANopen data (fieldbus, slave and variables) is lost. You now import the slave device via an EDS file. Then you recreate the structure from the CANbus and CANopen manager (master) in the device tree. You then copy the CANopen device (slave) of the device copy with the variables it still contains to under the newly created CANopen master.

1. First create a copy of the controller in the device tree: Right-click on the controller and select **Copy**.
2. Then right-click on the project name at the top of the device tree and select **Insert**.  
⇒ A copy of the controller has been created. You will use the CANopen data of the copy later.
3. Now swap the **e!COCKPIT** device description of the original device that it still contains and the CODESYS device description. To do so, right-click on the device and select **Update Device....**
4. Select your device – in this case, the 750-8214 device – under “Controllers (PFC)” and click **[Update Device]**.  
⇒ The device is updated. In the device tree, the red question mark symbol disappears. However, the substructures with CANopen device and CANopen data are also deleted.



5. Restore the configuration. For this, the EDS file of the CANopen slave must be installed (see [Installing WAGO Device Descriptions \[p 8\]](#)).
6. Now recreate the CANopen structure of your project. In the device tree, right-click on your device and select **Insert Device....**
7. First select the CANbus, click on the empty item below it and select the “WAGO CANopen Manager” in the same way using “Plug Device.”

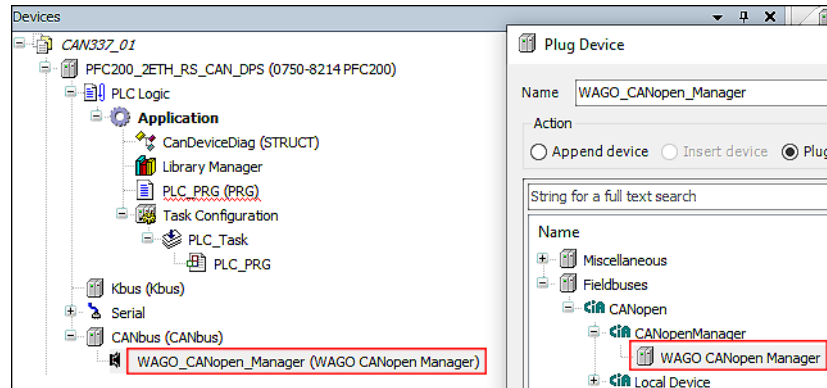


Figure 8: Adding CANopen Manager

8. Now copy the CANopen device (750-337) from the CANopen device you backed up at the beginning and re-insert it under the newly created “WAGO CANopen Manager” (you can also drag and drop it).

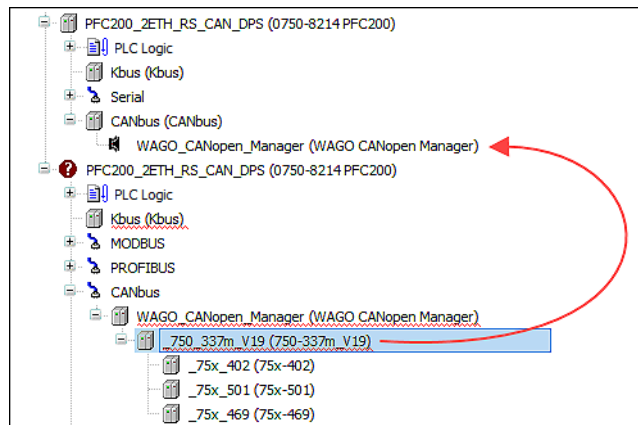


Figure 9: Inserting the Saved CANopen Device under the Newly Created CANopen Manager

9. Delete the device that was copied at the beginning. You can use all the variables in the new device as usual.

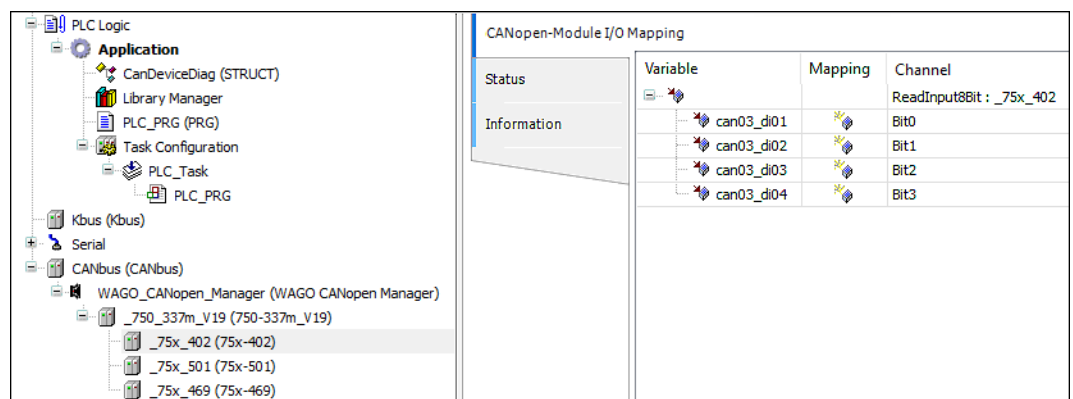


Figure 10: Using Variables

### 5.3 Project Conversion on the Example of EtherNet/IP

#### Initial state:

In this example, an **e!COCKPIT** project consisting of a PFC200 (750-8212) and a coupler (750-363) connected via EtherNet/IP is converted. An analog input module, a digital input module and a digital output module are plugged into the coupler; this yields an input data size of six bytes and an output data size of one byte. Variables corresponding to these data sizes have been created in the lower area of the Data Point Configurator. In the fieldbus configurator, the variables have been mapped to data points.

**Important:** A global variable list has been created in **e!COCKPIT** for the input/output data so the data will be retained in CODESYS throughout the update (Program Structure > right-click on "Application" > "Global Variable List").

**Connection** User parameter

Connection name	Application type:	Input size (T -> O)	Output size (O -> T)	Configuration size (target)	Path
class1 - exclusive owner	Exclusive owner	6	1	0	20 04 24 01 2C 65 2
class1 - input only	Input only	6	0	0	20 04 24 01 2C C6 2
class1 - listen only	Listen only	6	0	0	20 04 24 01 2C C7 2

Add Connection Edit Connection Delete Connection

**Input / Output data** Target configuration

Total input data point size (T -> O): 6 byte  
Total output data point size (O -> T): 1 byte

Direction	Name	Data type	Bit size	Comment
Input (T->O)	Producing Assembly (Input Data)			
	→ Input2	WORD	16	name, data type, bit size and information can be modified
	→ Input3	WORD	16	name, data type, bit size and information can be modified
	→ Input4	BYTE	8	name, data type, bit size and information can be modified
	→ Input5	BYTE	8	name, data type, bit size and information can be modified
Output (O->T)	Consuming Assembly (Output Data)			
	→ Output2	BYTE	8	name, data type, bit size and information can be modified

Figure 11: EtherNet/IP Project in e!COCKPIT

#### In brief:

In the following description, you perform the following steps in CODESYS: You update the device and create an Ethernet fieldbus under the device. You attach an EtherNet/IP scanner to it and the EtherNet/IP device below it. You adjust the EtherNet/IP connection and the length of the input/output data. Then you assign the address for each variable according to the hardware, and you can continue using your project.

1. First delete the slave device (in this case: 750-363). It contains no data you will need.

2. Right-click on the PFC200 and select **Update Device ....**
3. Select your device (in this case: 750-8212) and click **[Update Device]**.  
 ⇒ EtherNet/IP is removed from the tree. However, the variables you saved globally in **e!COCKPIT** are still available.

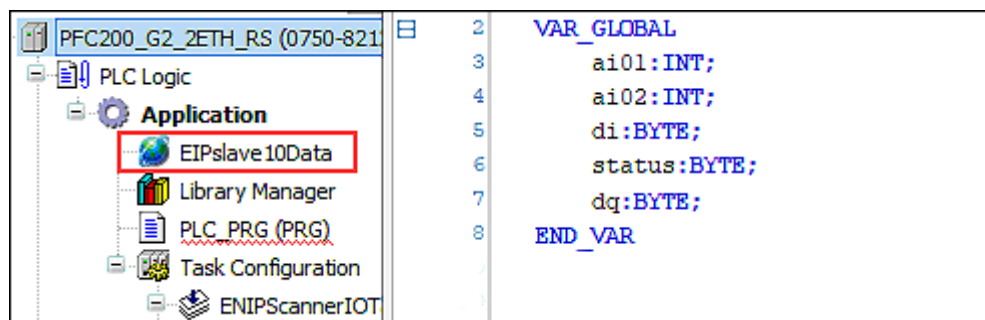


Figure 12: Global Variables

4. Now recreate the master and the slave. To do so, right-click on the PFC200 and select **Insert Device ....**
5. Leave the dialog open for the next three steps and click **[Insert Device]** to insert the “Ethernet” fieldbus, as well as the master and the slave inserted below it:
  - First select the “Ethernet” fieldbus in the dialog.
  - Click on the new “Ethernet” element in the device tree and select the “EtherNet/IP Scanner” in the dialog under “EtherNet/IP.”
  - Click the new “EtherNet\_IP\_Scanner (EtherNet/IP Scanner)” element in the device tree and select the EtherNet/IP device 750-363 in the dialog under “EtherNet/IP Remote Adapter.”

If you do not see the EtherNet/IP device in the list of selections, then you must first install the corresponding device description; see [🔗 Installing WAGO Device Descriptions \[► 8\]](#).

⇒ This procedure creates the following structure in the device tree:

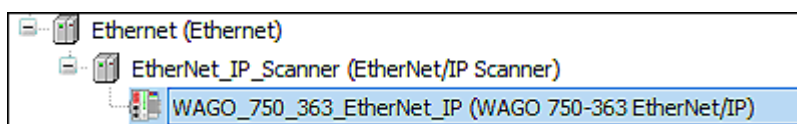


Figure 13: EtherNet/IP Scanner and Adapter

6. Double-click the EtherNet/IP adapter to make further settings:
7. To enter the IP address of the device, open the “General” tab.
8. To adjust the EtherNet/IP connection, open the “Connections” tab.
9. Double-click on the connection that was automatically created before and enter the data length of your input/output data:
  - Scanner to target (output) = data length of the output modules (in this case: one byte)
  - Target to scanner (input) = data length of the input modules (in this case: six bytes)

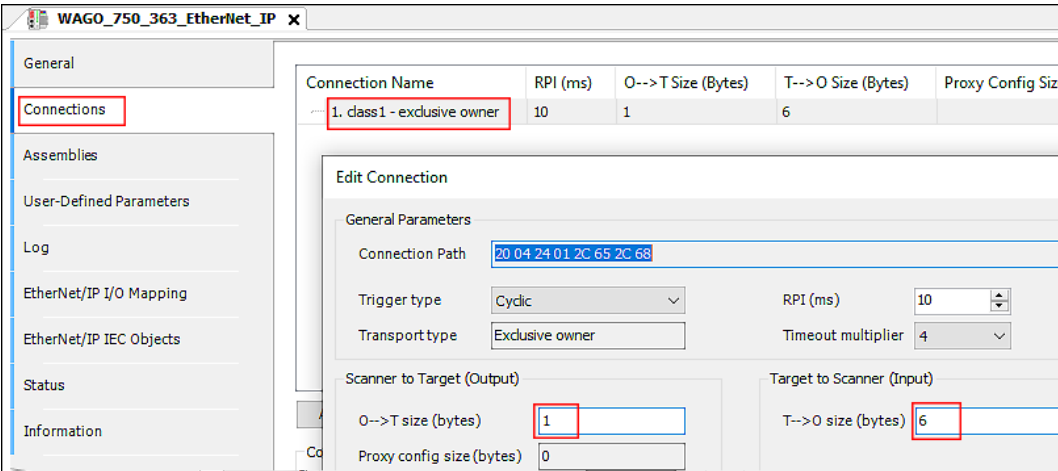


Figure 14: Entering the Input/Output Data Size of the Connection

10. Click **[OK]** to confirm.
- ⇒ Under “EtherNet/IP I/O Image,” you can now see the input/output variables that have been created with the required size. No variables for the respective input/output data of the I/O modules are assigned here yet. You will perform the mapping in what follows.

Connections	Variable	Mapping	Channel	Address	Type
Assemblies	class1 - exclusive owner				
User-Defined Parameters			Producing Assembly (Input Data)_Param0	%IB1	BYTE
Log			Producing Assembly (Input Data)_Param1	%IB2	BYTE
			Producing Assembly (Input Data)_Param2	%IB3	BYTE
			Producing Assembly (Input Data)_Param3	%IB4	BYTE
			Producing Assembly (Input Data)_Param4	%IB5	BYTE
			Producing Assembly (Input Data)_Param5	%IB6	BYTE
EtherNet/IP I/O Mapping			Consuming Assembly (Output Data)_Param0	%QB0	BYTE
EtherNet/IP IEC Objects					

Figure 15: Input/Output Variables That Have Been Created

11. First take a look at your hardware configuration to see which I/O module is the first I/O module to be addressed at the node and what data length it occupies. The representation of the node in WAGO I/O-CHECK can be helpful for this purpose.

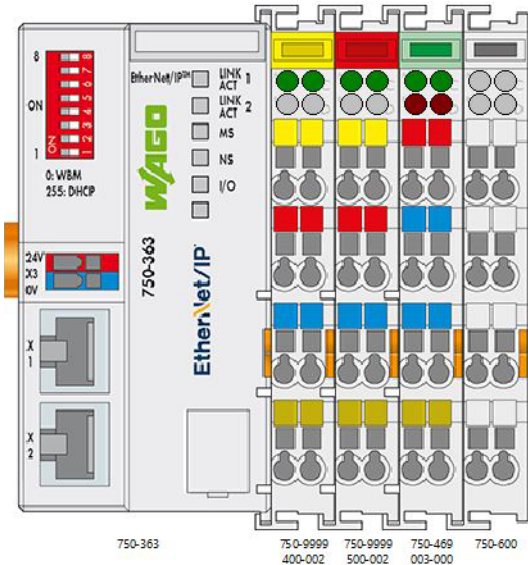


Figure 16: Representation of the Node in WAGO I/O-CHECK

12. Switch to the global variables (cf. item 3).
13. Click on a variable and press **[Shift] + [F2]**. This opens the variable declaration dialog.
14. Now enter the addresses for the input/output variables of the I/O modules in order according to your hardware structure:  
In this example:
  - Analog input starting from byte 1 – address %IB1 → occupies two bytes
  - Analog output starting from byte 3 – address %IB3 → occupies two bytes
  - Digital input starting from byte 5 – address %IB5 → occupies one byte
  - Status byte starting from byte 6 – address %IB6 → occupies one byte
  - Digital output starting from byte 0 – address %QB0 → occupies one byte

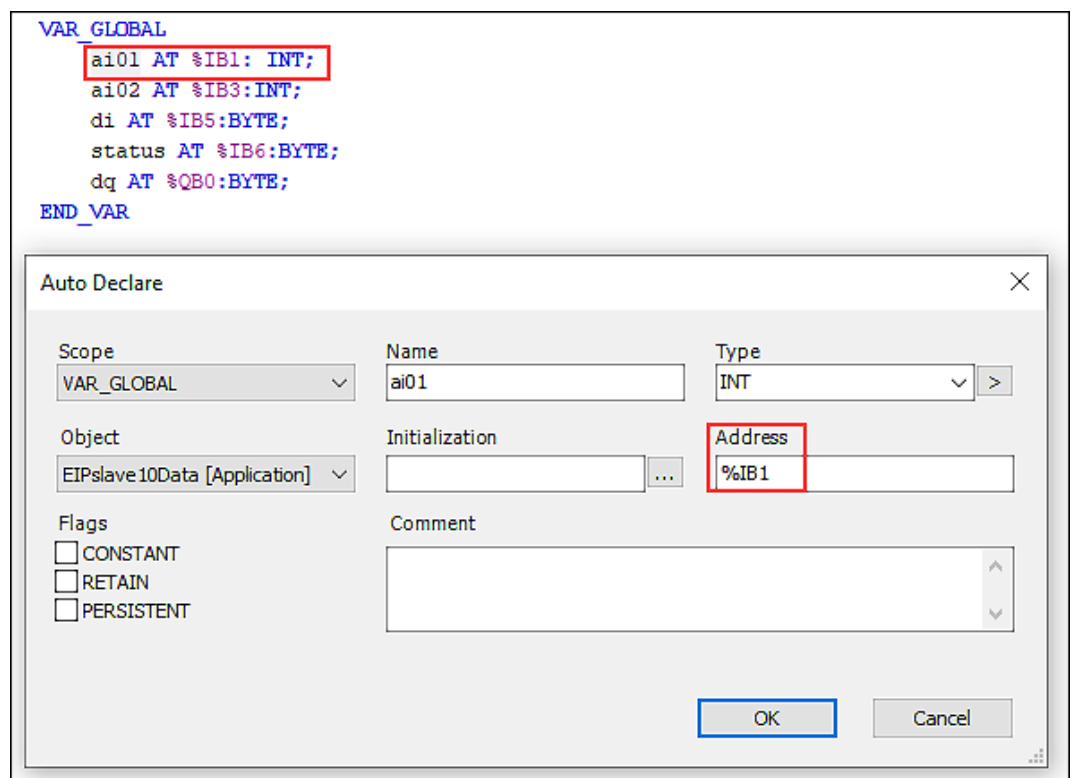


Figure 17: Hardware Addressing

15. Represent your hardware in this way.  
⇒ The full scope of the program can then be used.

## 5.4 Project Conversion on the Example of EtherCAT

### Initial state:

In this example, an **e!COCKPIT** project consisting of a PFC200 (750-8212) as the master with a connected CANopen coupler (750-354) as the slave is converted. For the coupler, a device description (ESI) has been installed in **e!COCKPIT** via the **Backstage View > Product Catalog > [Import Device]**. Four I/O modules are plugged into the coupler.

Variables have been created for the input/output data.

**Important:** In **e!COCKPIT**, right-click on the EtherCAT coupler and use **[Export]** to save a CSV file. The CSV file contains the I/O image with all the variable names; you will reimplement these into CODESYS.

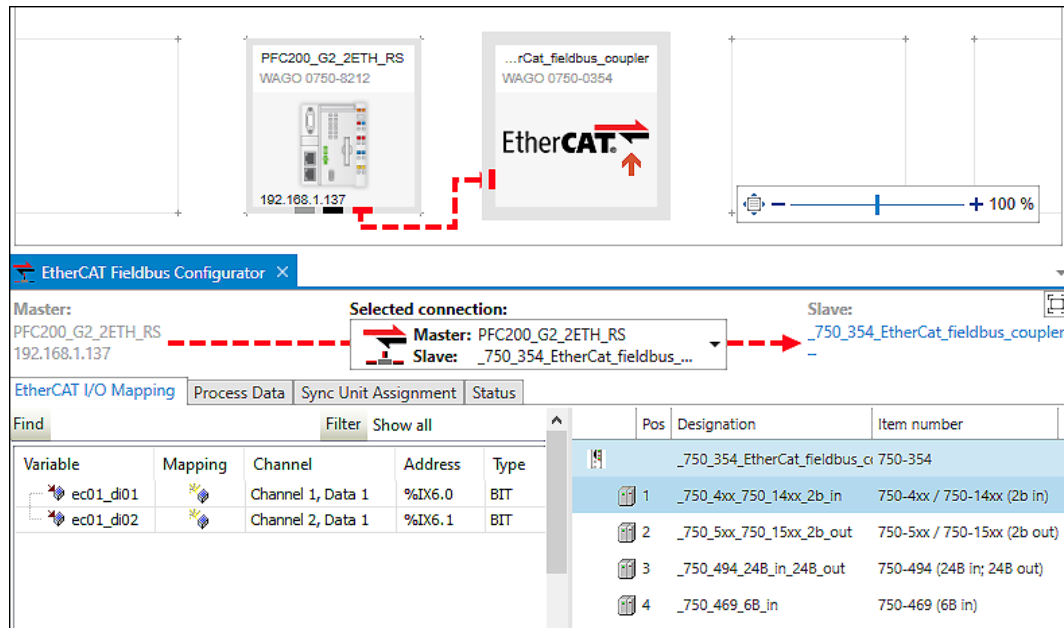


Figure 18: EtherCAT Project in **e!COCKPIT**

### In brief:

In the following description, you perform the following steps in CODESYS: You create a copy of the master device to back up the EtherCAT data. You then update the master device to the current CODESYS device description. The EtherCAT data (fieldbus, slave and variables) is lost. You now import the slave device via an ESI file. You will then insert an EtherCAT master into the device tree and copy the EtherCAT slave from the device copy, with the variables it still contains, under the newly created EtherCAT master. In order to use the variables from **e!COCKPIT**, import the I/O image (CSV) for the EtherCAT slave.

1. First click "Common.PCI." Here you will find the EtherCAT master and slave.
2. Now first create a copy of the controller in the device tree: Right-click on the controller and select **Copy**.
3. Then right-click on the project name at the top of the device tree and select **Insert**.  
⇒ A copy of the controller has been created. You will use the EtherCAT data of the copy later.
4. Now swap the **e!COCKPIT** device description of the original device that it still contains and the CODESYS device description. To do so, right-click on the controller and select **Update Device ...**

5. Select your device – in this case, the 750-8212 device – under “Controllers (PFC)” and click **[Update Device]**.
  - ⇒ The device is updated. In the device tree, the red question mark symbol disappears. However, the substructures with the EtherCAT data (previously under “Common.PCI”) are also deleted.
6. Restore the configuration. For this, the ESI file of the EtherCAT slave must be installed (see [Installing WAGO Device Descriptions \[► 8\]](#)).
7. Now recreate the EtherCAT structure of your project. In the device tree, right-click on your device and select **Insert Device ...**
8. Select the EtherCAT Master.

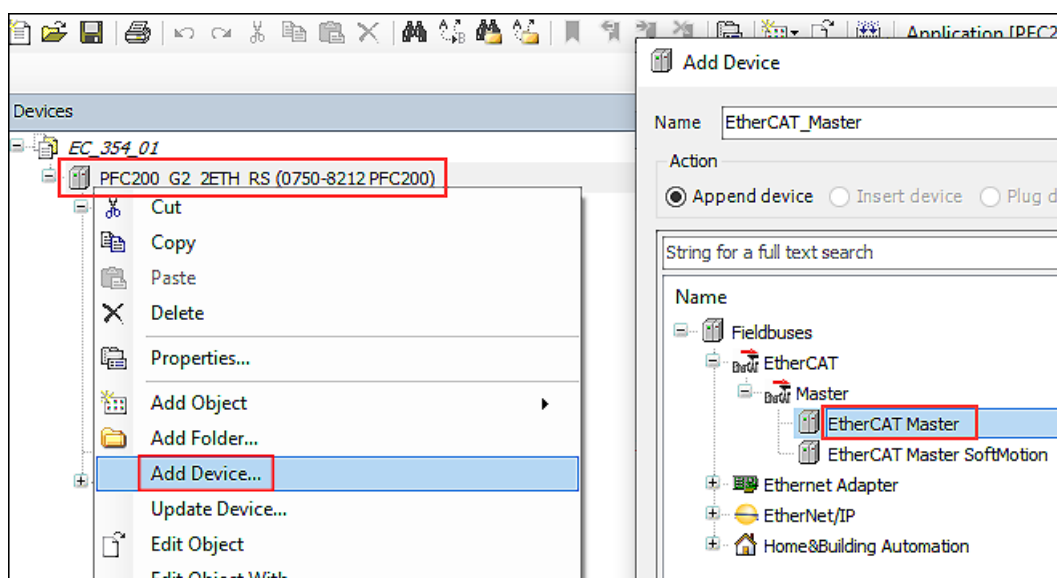


Figure 19: Adding EtherCAT Master

9. Now copy the EtherCAT coupler (750-354) from the device you backed up at the beginning and add it again under the newly created “EtherCAT master” (you can also drag and drop it).



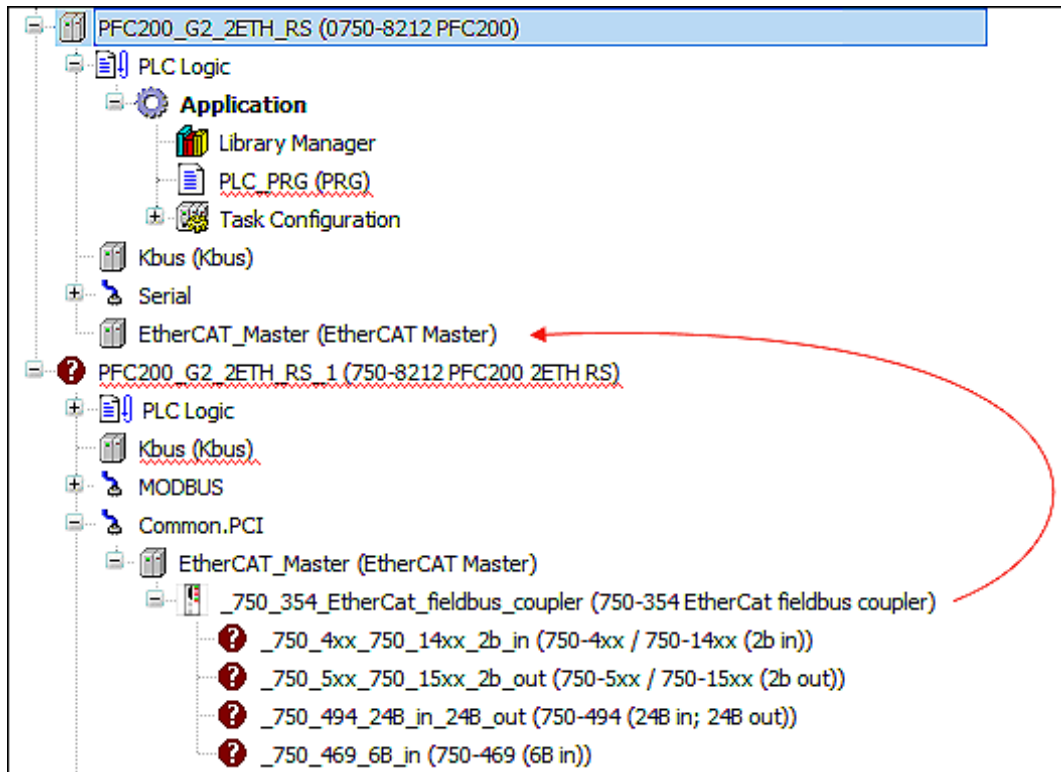


Figure 20: Inserting the Saved EtherCAT Device under the Newly Created EtherCAT Master

10. Delete the device that was copied at the beginning.
11. For all I/O modules marked with a question mark in the device tree, right-click **Update Device ....**
12. In the dialog, select the appropriate device description for the selected device.
13. Click **[Update Device]**.
  - ⇒ The device descriptions are updated, and the question mark symbols disappear. However, the variable names assigned in **e!COCKPIT** for the input/output data are still missing.
14. Right-click on the EtherCAT device and select **Import I/O Image from CSV ....**
  - ⇒ The variable names are carried over. You can use all the variables in the new device as usual.

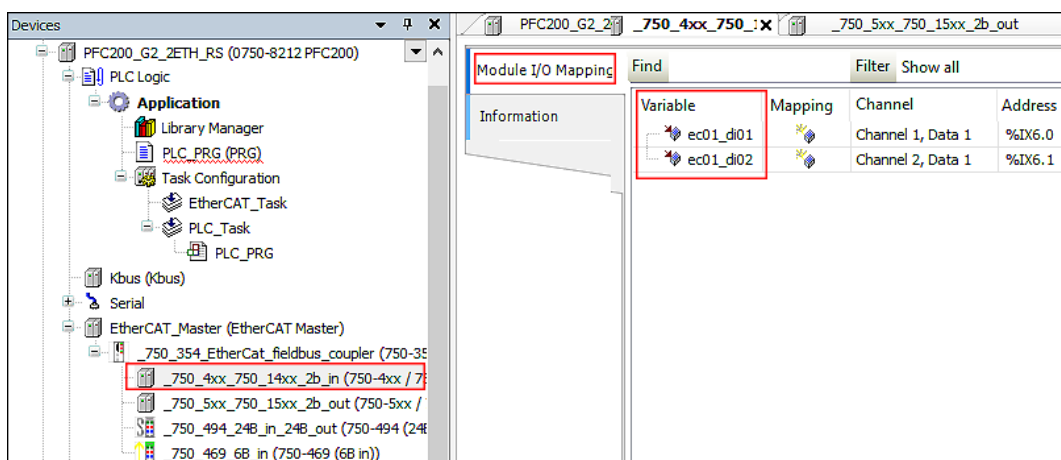


Figure 21: Using Variables



# Differences in Workflows and Functions

The following table gives an overview of the differences in the operation of CODESYS and **e!COCKPIT** and indicates where certain **e!COCKPIT** functions and settings can be found in CODESYS.

The biggest difference between the two tools is the user interface, especially the graphical representation of the network view that **e!COCKPIT** offers. For project engineering, devices can be dragged and dropped into the project in the form of graphical device tiles and connected to each other to form a network by drawing lines. Graphical aids (colored connectors) are also used to specify fieldbuses. Many functions can be accessed directly from the graphical network view through the context menu.

In contrast, in CODESYS, all operations are performed from the device tree. For example, here you can connect a network's devices by entering the IP address, and fieldbuses are also added and configured through the device tree.

The fieldbus configurators are similar because many CODESYS functions and interfaces are used in **e!COCKPIT**. The basic use of various fieldbuses is described in [🔗 Fieldbus Configuration in CODESYS \[▶ 29\]](#). You can find more information on fieldbus configuration in CODESYS in the CODESYS online help at [🌐 https://help.codesys.com/](https://help.codesys.com/).

Table 1: Differences in Workflows and Functions

<b>e!COCKPIT</b>	<b>CODESYS</b>
<b>Licenses</b>	
You can activate the license for <b>e!COCKPIT</b> itself via the Backstage view, "Licensing" page, [ <b>License Management</b> ] > [ <b>Enter Licenses</b> ].	Entering a CODESYS license is not required. CODESYS itself is available free of charge and license-free.
Licenses for additional functions can be purchased, assigned to devices and synchronized on devices via the "Project Licensing" panel.	<p>Licenses for WAGO libraries or additional functions can be displayed in CODESYS via the "WAGO Licensing" add-on and transferred with WAGOupload (see <a href="#">🔗 Installing the "WAGO Licensing" Add-on [▶ 9]</a> and <a href="#">🔗 Transferring Licenses with WAGOupload [▶ 10]</a>).</p> <p><b>Note:</b> Always use the "WAGO Licensing" add-on and WAGOupload for displaying and activating licensed WAGO products.</p> <p>For licensed CODESYS products on the other hand, use the CODESYS functions under "Tools" &gt; "License Manager"/"License Repository."</p>
<b>Downloading Additional Components</b>	
Updates for <b>e!COCKPIT</b> , as well as add-ons and firmware, can be downloaded and installed via the Backstage view on the "Updates & Add-ons" page.	<p>CODESYS updates appear on the software's start page when available.</p> <p>Add-ons can be downloaded via Software Package 3 and installed in CODESYS (see <a href="#">🔗 Installing CODESYS Add-ons [▶ 8]</a>).</p> <p>Firmware can be downloaded via Software Package 1 or 2; however, it is updated not in CODESYS, but rather via WAGOupload or an SD card (see <a href="#">🔗 Updating Firmware for Use in CODESYS [▶ 7]</a>).</p> <p><b>Note:</b> WAGO is currently working on a central download solution via the WAGO Download Center. We will inform you as soon as this is available.</p>

e!COCKPIT	CODESYS
<b>Importing Devices</b>	
Device descriptions for WAGO devices are automatically integrated when <b>e!COCKPIT</b> is installed. Additional devices can be integrated via the Backstage view, "Product Catalog" page > <b>[Import Device]</b> .	Required device descriptions are first downloaded through Software Package 1 or 2 and integrated via the "Tools" tab > <b>CODESYS Installer...</b> (see <a href="#">🔗 Installing WAGO Components collectively via Package Files [p 8]</a> ). Additional devices can be integrated via "Tools" > "Device Repository ..." (see <a href="#">🔗 Installing WAGO Device Descriptions [p 8]</a> ).
<b>Adding Devices in the Project</b>	
Devices can be added to the project by dragging and dropping them from the product catalog.	Devices are offered for selection in a dialog when a new project is created. For this to work, the corresponding device descriptions must already have been installed. In an existing/empty project, devices can be added by clicking on the root node and <b>Insert Device ...</b> .
I/O modules can be connected to the device by dragging and dropping them.	I/O modules can be added by right-clicking on "Kbus (Kbus)" in the device tree and <b>Insert Device ...</b> .
<b>Selecting and Configuring a Fieldbus</b>	
The fieldbus is defined in a graphical network view by drawing connecting lines between fieldbus-specific connectors. For Modbus, the protocol is also specified via a context menu (e.g., TCP or UDP). In addition, the communication direction (master/slave) can be assigned via the context menu.	The fieldbus is added via <b>Insert Device ...</b> in the context menu of the device. Only the fieldbuses that the device supports are displayed in the dialog. <b>Note:</b> For the "Modbus" fieldbus, "Ethernet" is selected first. Modbus can then be selected via <b>Insert Device ...</b> under the Ethernet element in the device tree.
<b>Establishing a Connection between Devices</b>	
In <b>e!COCKPIT</b> , devices are connected by drawing connecting lines between them.	As a general rule, in CODESYS, a connection is established between devices by adding/nesting elements in the device tree and (depending on the fieldbus) by entering the slave IP address for the master device.  <a href="#">🔗 Fieldbus Configuration in CODESYS [p 29]</a> describes how devices are integrated/connected depending on the fieldbus used. An example for commissioning a Modbus node is described in <a href="#">🔗 Example Project: Creating a New Modbus Project in CODESYS [p 37]</a> .
<b>Mapping Data Points</b>	
For Modbus couplers, the I/O image is created automatically.	The I/O image/mapping of a Modbus coupler can be created manually with the help of the CODESYS manual (the Modbus configuration) and the product manuals (the structure of the I/O data).
<b>Replacing Devices</b>	
In <b>e!COCKPIT</b> , devices of a project of the same type can be replaced via the menu ribbon, "DEVICE" tab, or by right-clicking in the device's context menu and clicking <b>[Replace Device]</b> .	In CODESYS, a device can be replaced by right-clicking in the device tree and selecting <b>Update Device ....</b> Functions that the new device does not support are eliminated; additional functions that the new device supports are added (see also "Adjusting Device Type Versions/Device Descriptions" in this table).
<b>Network Scan</b>	
The network scan for devices is performed in <b>e!COCKPIT</b> via the menu ribbon > "NETWORK" tab > <b>[Scan]</b> . Here you can also set the PC interface and IP range.	In CODESYS, scanning the entire network is not supported, but searching for an individual device is. To perform the scan, double-click on the device in the device tree > "Communication" tab > <b>[Scan Network ...]</b> Limiting the IP range, scanning multiple devices and setting different interfaces are not possible.

e!COCKPIT	CODESYS
<b>Module Scan</b>	
You can perform a module scan in e!COCKPIT via the menu ribbon > "DEVICE" tab > <b>[Scan Modules]</b> , or in a device's context menu.	You can find I/O modules by right-clicking on "Kbus (Kbus)" in the device tree and selecting <b>Scan for Devices</b> .
<b>Compiler Version, Visualization Profile, Device Description and Firmware Compatibility</b>	
In e!COCKPIT, message windows appear in the event of incompatibility and provide recommendations for changes. A compatibility list is available for guidance: <a href="https://tech-docs.wago.com/Software/eCOCKPIT/en-US/index.html#1889686667">https://tech-docs.wago.com/Software/eCOCKPIT/en-US/index.html#1889686667</a> .	The following versions are compatible for devices using CODESYS Runtime: see product manuals of the PFC G2 as of FW23 under "CODESYS V3 Compatibility".
<b>Adjusting Compiler Version/Visualization Profile</b>	
The compiler version/visualization profile can be adjusted in the Backstage view, "Project Settings" page > "CODESYS Project Settings" > "Project Settings."	The compiler version and the visualization profile can be adjusted on the "Project" tab under <b>Project Settings ...</b> > <b>Compiler Options</b> or <b>Visualization Profile</b> .
<b>Adjusting Device Type Versions/Device Descriptions</b>	
Device type versions/device descriptions can be modified in the menu ribbon > "DEVICE" tab > "Version Information" tab group with the <b>[Replace]</b> button.	Device type versions/device descriptions are displayed for selection in the "Update Devices" dialog (right-click on the device > <b>Update Devices ...</b> ). To display all available versions of a device, the "Show All Versions (Experts Only)" option must be enabled in the dialog.
<b>Updating Firmware</b>	
The firmware can be downloaded in the Backstage view > "Updates & Add-ons" page and installed via the menu ribbon > "DEVICE" tab > "Replace Firmware."	The firmware is downloaded via Software Package 1 or 2. However, replacing the firmware is done not with CODESYS, but rather with WAGOupload or via SD card (see <a href="#">Updating Firmware for Use in CODESYS [p. 7]</a> ).
<b>Compiling, Connecting and Launching an Application</b>	
You can compile an application via the "PROGRAM" tab > "Compile." You can connect devices online either by right-clicking on a device and selecting "Connect," or alternatively via the button in the menu ribbon. "PROGRAM"/"DEBUG" tab > <b>Start</b> launches the application.	You can compile the application via the "Build" tab > <b>Generate Code</b> . You can connect the application online by double-clicking on the device and the "Online" tab > <b>Login</b> or right-clicking "Application" > <b>Login</b> . Right clicking "Application" > <b>Start</b> launches the application.
<b>Debug Mode</b>	
After login, you can start the execution and verification of the source code via the menu ribbon > "PROGRAM"/"DEBUG" tab > <b>[Simulate Application]</b> .	After login, you can start execution and verification of source code via the "Debug" tab > <b>Start</b> .
<b>Creating a Visualization</b>	
You can create a visualization by right-clicking on an application and selecting the new "Visualization" element.	You can create a visualization by right-clicking on an application and choosing <b>Add Object &gt; Visualization</b> .
<b>Program Download</b>	
A program can be downloaded via the menu ribbon > "PROGRAM" tab > "Program Download" or "Multiple Download," for example.	A program can be downloaded via the "Online" tab > <b>Source Download to Connected Device</b> or <b>Multiple Download</b> .

e!COCKPIT	CODESYS
<b>Adding Libraries</b>	
Libraries are added via the “Program Structure” panel by double-clicking “Library Manager” and <b>[Add Library]</b> .	Required libraries are first downloaded via Software Package 1 or 2 and installed via the “Tools” tab > <b>CODESYS Installer ...</b> (see <a href="#">🔗 Installing WAGO Components collectively via Package Files [▶ 8]</a> ). Additional libraries can be installed via the “Tools” tab > <b>Library Repository ...</b> (see <a href="#">🔗 Installing WAGO Libraries [▶ 9]</a> ) and added from the repository to the project in the device tree by double-clicking “Library Manager” > <b>[Add Library]</b> .
<b>Add-ons</b>	
License-based additional components (add-ons) appear on the “Updates & Add-ons” page of the Backstage view together with other components such as firmware, service packs, libraries and visualization styles. They can be downloaded, installed and licensed there.	License-based additional components (add-ons) can be downloaded via Software Package 3 and installed via the “Tools” tab > <b>CODESYS Installer ...</b> (see <a href="#">🔗 Installing CODESYS Add-ons [▶ 8]</a> ). Unlike e!COCKPIT, CODESYS bundles the add-ons together; they are not available individually.
<b>Configuring I/O Modules</b>	
WAGO I/O-CHECK is already integrated into e!COCKPIT for I/O module configuration.	The stand-alone WAGO I/O-CHECK software is used for I/O module configuration (see <a href="#">🔗 Installing and Using WAGO I/O-CHECK [▶ 10]</a> ).
<b>Importing/Exporting Program Elements (PLCopen XML)</b>	
Program elements can be imported/exported in the Backstage view on the “Import/Export” page.	Program elements can be imported/exported on the “Project” tab > <b>Import PLCopenXML .../Export PLCopenXML ....</b>
<b>Documenting/Printing a Project</b>	
The “Document Project” function can be found in the Backstage view > “Print” page.	The “Document Project” function can be found on the “Project” tab.
<b>Version Management (SVN)</b>	
If the “e!COCKPIT SVN” add-on has been installed and licensed (Backstage view, “Updates & Add-ons” page), different versions of a project can be managed via the “SUBVERSION” tab.	CODESYS version management for projects is included in Software Package 3 (fee-based; see <a href="#">🔗 Installing and Launching [▶ 6]</a> ).
<b>Python</b>	
e!COCKPIT can be operated via Python scripts in the Backstage view, “Scripting” page.	Python scripts can be executed from the Tools tab > <b>Scripting</b> .

# Fieldbus Configuration in CODESYS

Fieldbus configuration in *e!COCKPIT* and CODESYS can differ to varying degrees depending on the fieldbus used. Some CODESYS configurators can be integrated into *e!COCKPIT* with slight adjustments. In this regard, the differences between the use of the two tools are comparatively small. For other configurators (such as Modbus), various modifications have been made for *e!COCKPIT*. This results in bigger differences in the comparison between *e!COCKPIT* and CODESYS. After migrating your project, use the native fieldbus configurators from CODESYS. Note that this renders WAGO fieldbus configuration documentation inapplicable; use the CODESYS online help instead ([help.codesys.com](https://help.codesys.com)).

When you start a new project and add a device, the device tree below the device is initially empty. At the beginning, you create fieldbus elements in the device tree and arrange further devices below them. Devices are generally connected by adding master/slave elements below the respective fieldbus element. For devices of a project, generally only fieldbuses that the device supports can be selected and configured.

## 7.1 Modbus – First Steps

For the “Modbus” fieldbus, note that for the Modbus master and slave device, an “Ethernet” element has to be created first. The Modbus master and slave are inserted under it. A representative of the slave is inserted under the Modbus master and establishes a connection between the master and slave via the slave IP address.

- ✓ You need to have inserted two devices in your project. One of the devices will play the role of the Modbus master and the other that of the Modbus slave.
- 1. First create an “Ethernet” fieldbus element for both devices in the device tree. To do so, right-click on the respective device and select **Insert Device ...**
- 2. Select the “Ethernet” fieldbus under “Ethernet Adapter.”

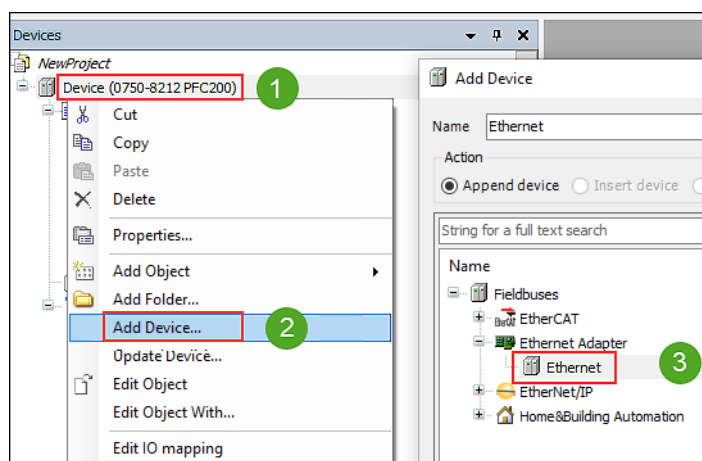


Figure 22: Inserting an “Ethernet” Fieldbus

- 3. Click **[Insert Device]** and leave the dialog open.
  - ⇒ A new “Ethernet (Ethernet)” element is created in the device tree.
- 4. Click on the new “Ethernet (Ethernet)” element.
- 5. Under “Modbus,” select the “Modbus TCP Slave” for the slave device in the dialog that is still open, confirm with **[Insert Device]** and leave the dialog open.

6. Under “Modbus,” select the “Modbus TCP master” for the master device in the dialog that is still open, confirm with **[Insert Device]** and leave the dialog open.

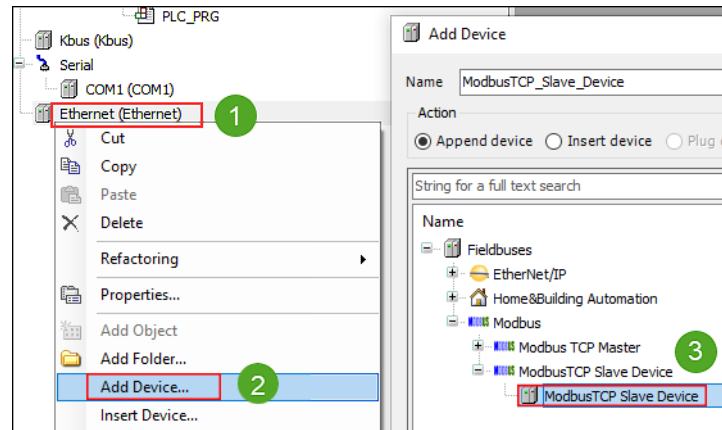


Figure 23: Inserting Devices

7. Click the “Modbus\_TCP\_Master (Modbus TCP Master)” element in the device tree and insert the “Modbus TCP Slave Device” element from the dialog that is still open.  
 ⇒ You have now inserted a slave element under the master in the device tree, through which the connection between the master and the physical slave is established.

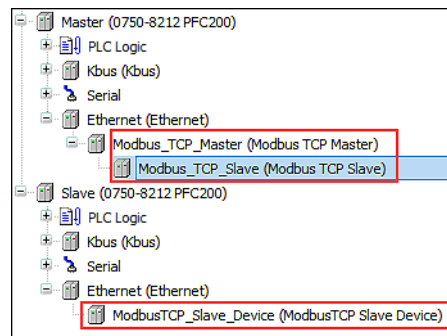


Figure 24: Modbus Master and Slave Created

8. To link the master and slave, now enter the IP address of the slave device for the “Modbus\_TCP\_Slave (Modbus TCP Slave)” slave element below the master. To do so, double-click “Modbus\_TCP\_Slave (Modbus TCP Slave)” to open the configuration and enter the IP address of the slave device on the “General” tab.

You can find a complete example project in [📁 Example Project: Creating a New Modbus Project in CODESYS \[▶ 37\]](#).

## 7.2 CANopen – First Steps

- ✓ You need to have inserted two devices in your project. One of the devices will play the role of the CANopen device (slave) and the other that of the CANopen manager (master).
1. You first create a “CANopen” fieldbus element for both devices in the device tree. To do so, right-click on the respective device and select **Insert Device ...**
  2. Under “CANbus,” select the “CANbus” item and click **[Insert Device]**.

### Configuring CANopen Device (Slave)

1. Right-click on the new, still empty item under the “CANbus (CANbus)” element of the slave device.
2. To insert a slave, select the “WAGO CANopen Device” and confirm with **[Plug Device]**.

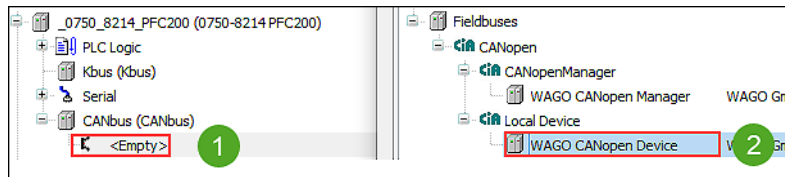


Figure 25: Inserting “WAGO CANopen Device”

3. To open the device settings, double-click on the new “WAGO\_CANopen\_Device (WAGO CANopen Device)” element in the device tree.
  - ⇒ On the “General” tab you can edit the I/O area for the CANopen device, among other things:
4. Click **[Edit I/O Area ...]** and add one or more new areas.
5. Click **[OK]** to confirm.

⇒ After that, you can either export the I/O image and additional data as an “Electronic Data Sheet” (EDS) (and re-import it later) or install the data directly in the device repository:

#### 6. Option 1: Exporting the Slave’s EDS

On the “General” tab, click **[Export EDS File ...]**.

If you are using multiple CANopen devices, save the EDS file with a unique file-name.

#### Option 2: Installing EDS Directly in the Device Repository

Instead of exporting the slave’s EDS file and then re-importing it in the device repository, you can also install the EDS file of a CANopen device (slave) directly in the device repository using the corresponding button. After that, you can find the slave there under the selected product name.

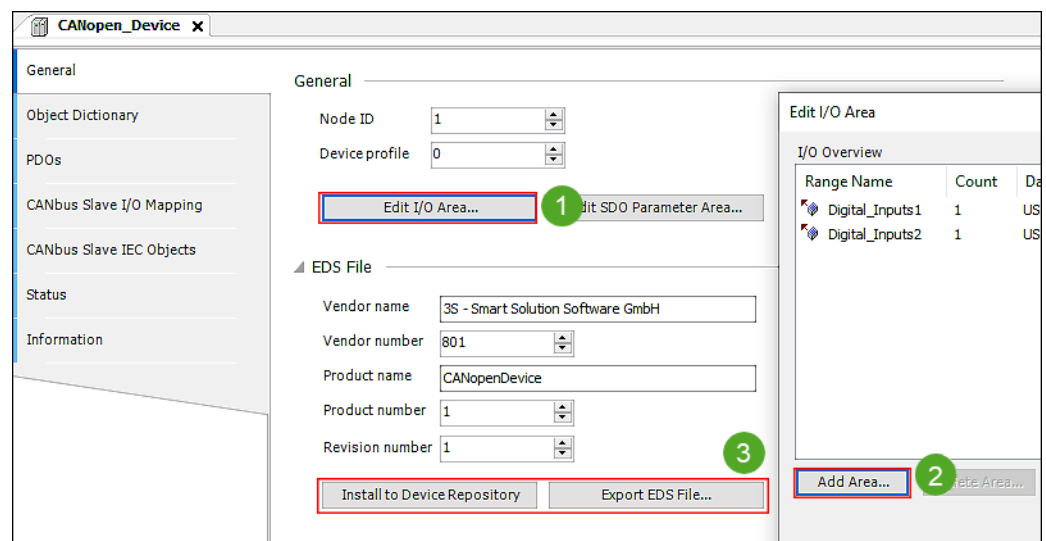


Figure 26: Exporting an EDS File or Installing It Directly in the Device Repository

## Configuring CANopen Manager (Master)



1. Right-click on the new, still empty item under the “CANbus (CANbus)” element of the master device.
2. To insert a master, select the “WAGO CANopen Manager” and confirm with **[Plug Device]**.

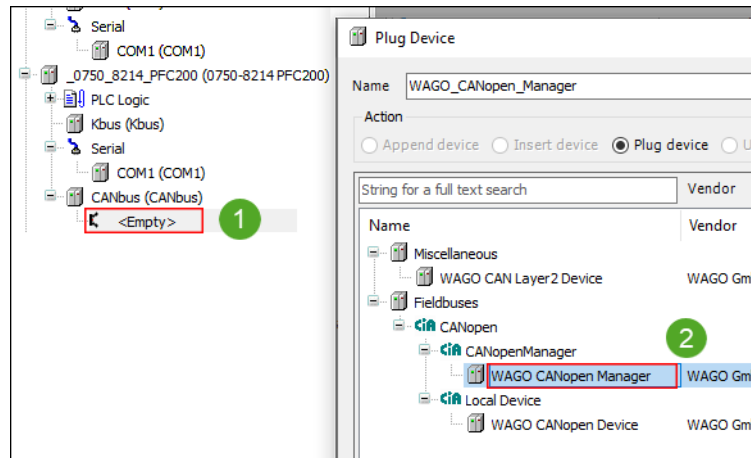


Figure 27: Inserting “WAGO CANopen Manager”

- ⇒ To assign one or more slaves to the master, the EDS files of the slaves must be installed; they can then be attached to the master:
3. To import or install the EDS file of a slave, click **[Device Repository ...]** on the Tools tab and **[Install ...]** in the dialog.  
Select the previously exported EDS file (or another one) to import and click **[Open]**.
  4. To attach the slave to the master, right-click the “WAGO\_CANopen\_Manager (WAGO CANopen Manager)” in the device tree and select **Insert Device ...**.
  5. Select the corresponding slave device from the list and click **[Insert Device]**.
- ⇒ The devices are now created and connected.

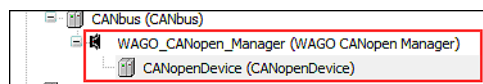


Figure 28: Connected CANopen Devices

### 7.3 EtherNet/IP – First Steps

1. Add a controller that is to be used as an EtherNet/IP master (scanner) to your project.
2. Import the EDS file of a slave device (adapter) via the “Tools” tab > “Device Repository.”
3. Right-click on the controller and click **Insert Device ...**
4. Select the “Ethernet” item under “EtherNet/IP” > “EtherNet/IP Adapter” and click **[Insert Device]**.



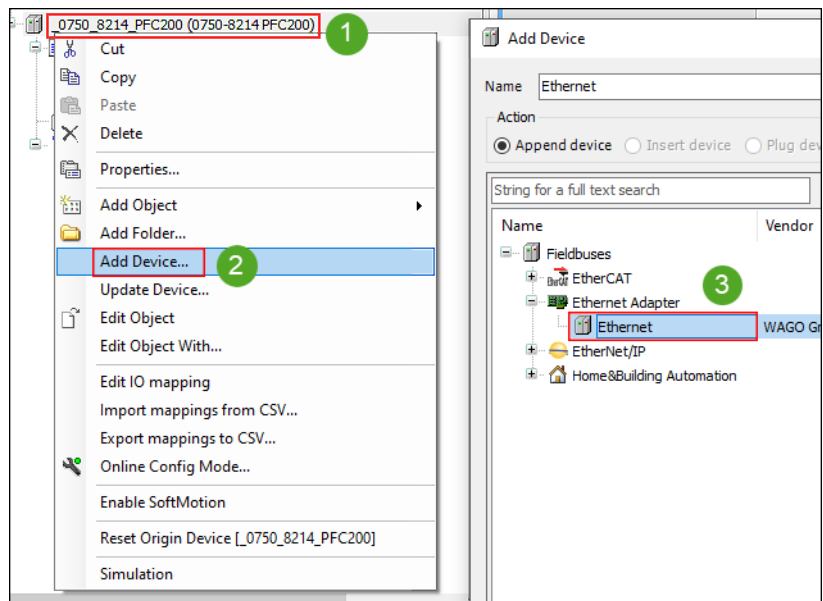


Figure 29: Inserting Ethernet/IP Device

5. Double-click the “Ethernet (Ethernet)” element to configure the Ethernet interface.
6. On the General tab, click **[Browse ...]**, select network adapter “br0” and enter the IP address for the EtherNet/IP interface.

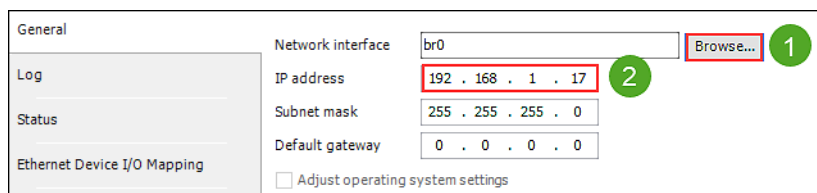


Figure 30: Configuring EtherNet/IP Interface

7. In the device tree, right-click “Ethernet (Ethernet)” and select **Insert Device ...**.
8. Select the “EtherNet/IP Scanner” under “EtherNet/IP.”
9. Select **Insert Device ...** and leave the dialog open.

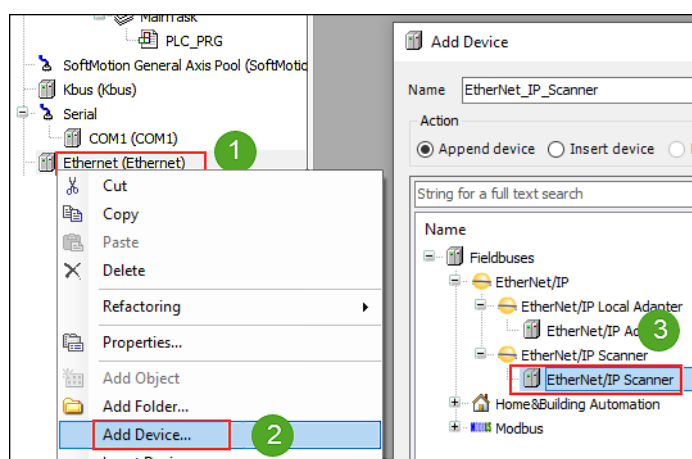


Figure 31: Inserting EtherNet/IP Scanner

10. Click on the “EtherNet\_IP\_Scanner (EtherNet/IP Scanner)” element in the device tree and select the slave device (adapter) imported via the EDS file in the dialog that is still open.

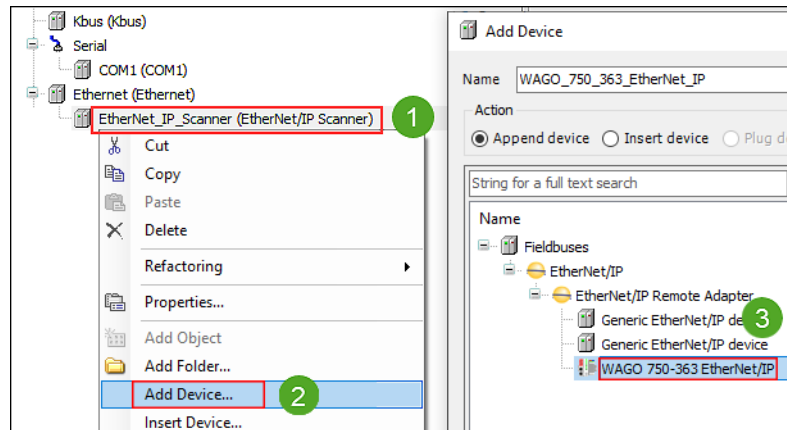


Figure 32: Selecting EtherNet/IP Slave

11. Click **[Insert Device]**.
12. Double-click on the slave device in the device tree to open the slave configuration and specify connections and assemblies.

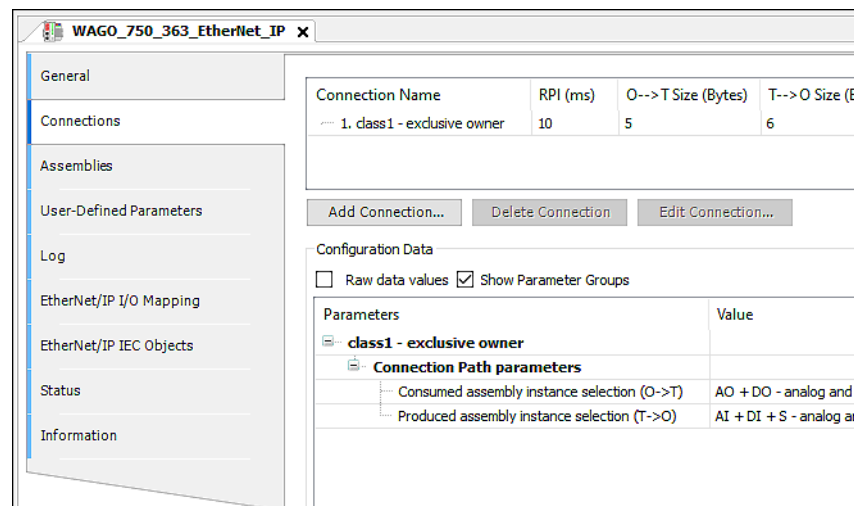


Figure 33: Configuring EtherNet/IP Slave

⇒ In EtherNet/IP, I/O data (input and output data points) is transferred via so-called connections (the term “Connections” is also used in **e!COCKPIT**). When the fieldbus starts up, one or more of these connections can be established between an EtherNet/IP master and an EtherNet/IP slave. The input and output data points of a connection and the configuration data are grouped into so-called assemblies. To see which connections a slave device supports, consult the device description file (EDS file) or the respective product manual of the slave. The connections and the data to be transferred are usually configured automatically using the information contained in the EDS file.

13. On the “Connections” tab, use **[Add Connections ...]** to create new connections according to your product manual.
14. Click on the new connection and then click **[Edit Connection ...]** to set the path in the dialog and to set or change the size of the input/output data ( $T \rightarrow O$  /  $O \rightarrow T$ ) for your connections.
15. Click the Assemblies tab to see how the assemblies were created according to your input/output data.

## 7.4 EtherCAT – First Steps

For EtherCAT communication, the Ethernet ports of the controller that are connected internally via a switch must be disconnected. EtherCAT communication is only permitted via port X2.

1. To separate the ports, first open the WBM of the controller that is to be used as the EtherCAT master. To do so, enter the IP address of the controller in the browser and log in.
2. On the “Configuration” > “Networking” tab of the WBM, click “Ethernet Configuration.”
3. Select port X2 for bridge 2 (also referred to as “br1” in CODESYS) to separate the ports.

Bridge	Port	
	X1	X2
1	<input checked="" type="radio"/>	<input type="radio"/>
2	<input type="radio"/>	<input checked="" type="radio"/>

Figure 34: Separating Ethernet Ports

4. Then reboot the controller.
5. Now add the controller to your CODESYS project.
6. First, double-click on the controller to open the configuration.
7. Configure the gateway and enter the IP address of the controller.
8. Right-click on the controller and click **Insert Device ....**
9. Select the “EtherCAT Master” item in the tree under “EtherCAT” > “Master” and click **[Insert Device]**.

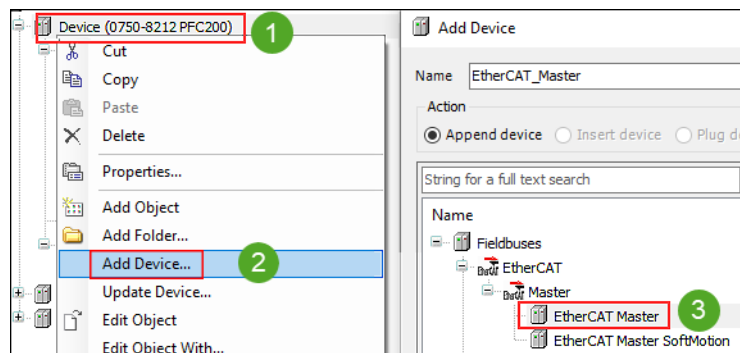


Figure 35: Inserting EtherCAT Master

10. To configure the EtherCAT master, double-click it in the device tree.
11. Click **[Select ...]** on the General tab of the EtherCAT configuration to specify the network adapter that is to be used, i.e. the EtherCAT interface.
12. Select the “br1” item. This identifies the EtherCAT interface on port X2.

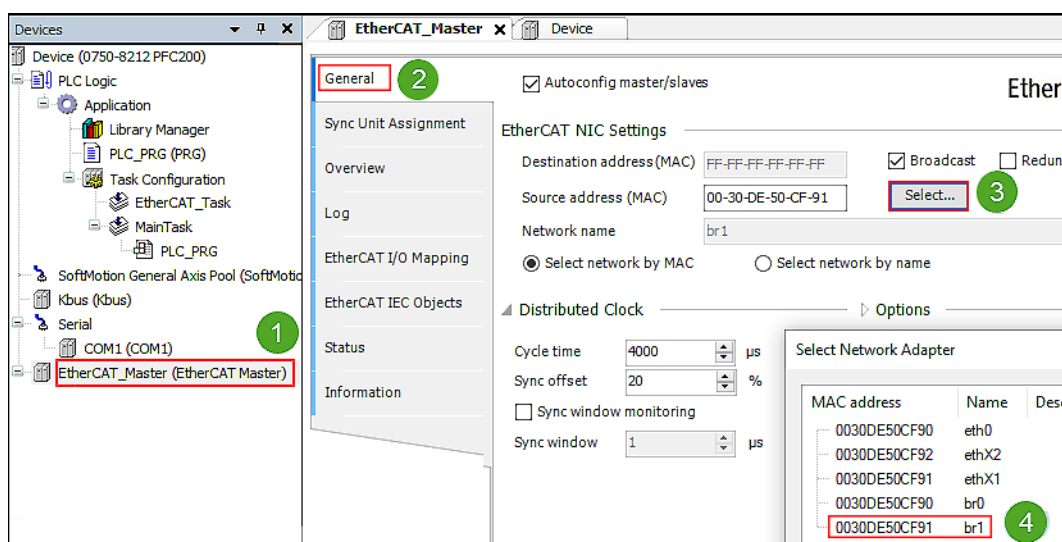


Figure 36: Specifying the Interface

13. Right-click on the “EtherCAT\_Master (EtherCAT\_Master)” in the device tree and select **Insert Device ...**.

14. Select a (previously installed) EtherCAT slave from the list and click **[Insert Device]**.

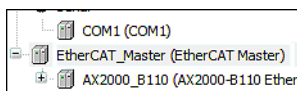


Figure 37: EtherCAT Master with Attached Slave

# Example Project: Creating a New Modbus Project in CODESYS

In this example, you will create a Modbus project with two PFC200 750-8212 Controllers, which are to be used the Modbus master and Modbus slave. You will add I/O modules, create a small program and exchange data between the devices.

Initial state:

- The 750-8212 Controllers have firmware version 23.
- You need to have installed CODESYS and required components, such as device descriptions and libraries (see [🔗 Installing and Launching \[▶ 6\]](#)).

## Create the Project

1. Click **New Project** on the start page or the “File” tab.
2. Select the “Standard Project” as the template and assign a name for the project.  
⇒ The “Standard Project” dialog shows all the available devices. The devices result from the installed device descriptions.
3. Select the first controller (in this example, a PFC200 750-8212) and the programming language you want to use.
4. To add I/O modules, right-click on the “Kbus” element in the device tree and select **Insert Device**.
5. Select the I/O modules of the node from the list and click **[Insert Device]** for each I/O module. In this example, the 750-430 and 750-530 I/O modules are used.  
⇒ The I/O modules of the node are attached below the “Kbus” element.
6. To add the second controller to the project, right-click on the project name (the first element of the device tree) in the device tree and select **Insert Device**.
7. Repeat steps 4 and 5 for this controller.

## Connect to the Device

1. To connect to the controller, double-click it in the device tree.
2. In the graphical view consisting of the PC, gateway and device, enter the IP address of the controller – for example, 192.192.168.1.17 in this case – for the controller that is locally connected to the PC.  
Alternatively, click **[Scan Network]** to search the network for devices.  
**Tip:** To be able to identify the device more easily during a network scan, it is useful to define a unique host name for the device beforehand via the WBM.
3. Select your device and transfer it by clicking **[OK]**.
4. The login screen of the device opens.
5. Log in to the device with your access data (default: user “admin,” password “wago”).
6. Repeat steps 8 to 12 for the second controller.

## Configure the Slave Device

1. First create an “Ethernet” fieldbus element for the controller that is to be used as the slave. To do so, right-click on the device and select **Insert Device ....**

2. Select the “Ethernet” fieldbus under “Ethernet Adapter.”

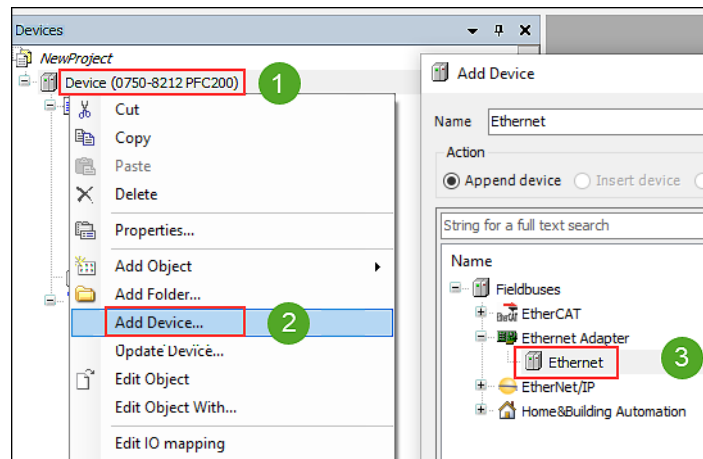


Figure 38: Inserting an “Ethernet” Fieldbus

3. Click **[Insert Device]** and leave the dialog open.
  - ⇒ A new “Ethernet (Ethernet)” element is created in the device tree.
4. Click on the new element and select the “Modbus TCP Slave” for the slave device under “Modbus” in the dialog that is still open.
5. Confirm with **[Insert Device]**.
  - ⇒ In what follows, you will create variables and write a short program to test the data exchange between the master and slave:
6. **Create Variables:** Double click on the slave device to open the configuration.
7. Create two variables on the “Modbus TCP Slave Device I/O Image” tab: one with the name “fromMaster” in the holding register and one with the name “toMaster” in the input register.

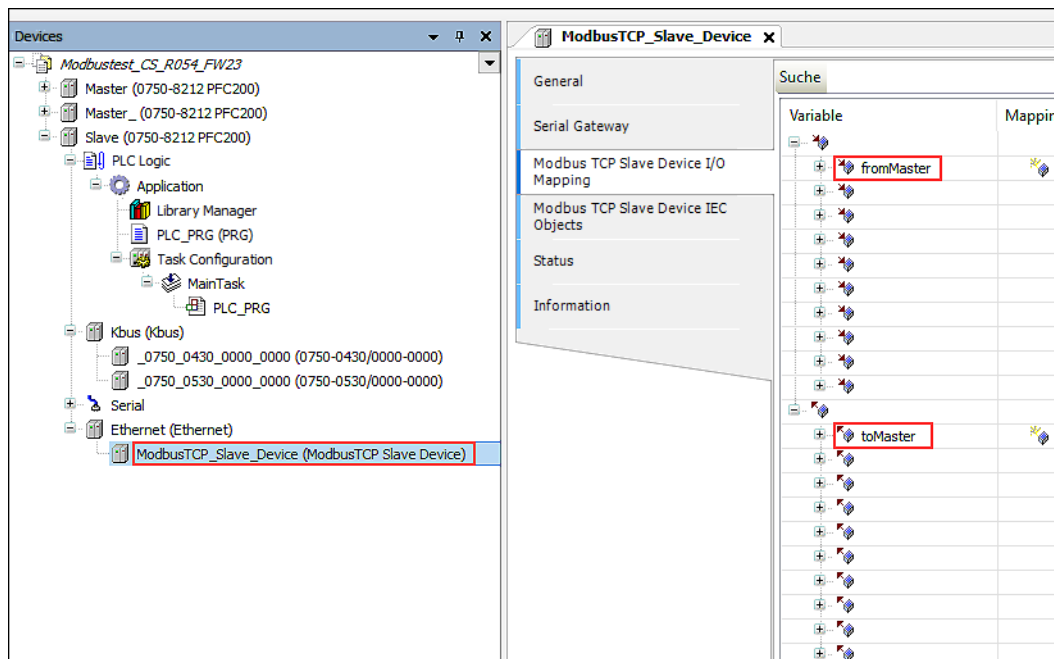


Figure 39: Creating Variables

8. To output variables to the I/O modules, double-click on each I/O module:  
For the 750-430 Digital Input Module, enter “localIn” as the variable.  
For the 750-530 Digital Output Module, enter “localOut” as the variable.

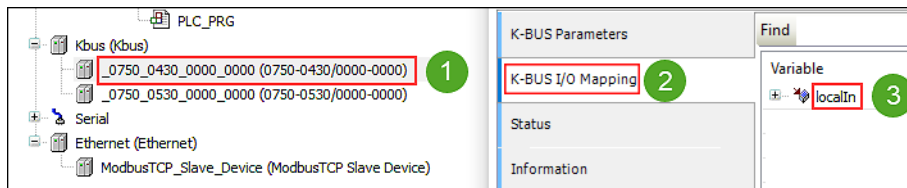


Figure 40: Variable “localIn”

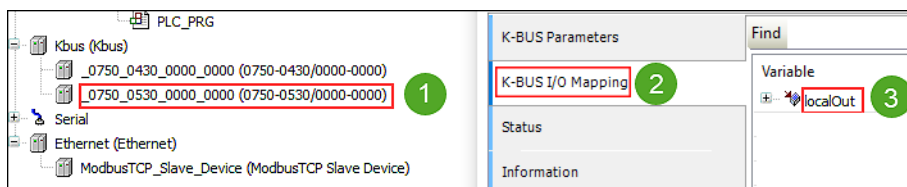


Figure 41: Variable “localOut”

9. **Write the Slave Program:** To open the program editor, double-click on the “PLC\_PRG” slave program in the device tree.
10. Create a simple program:
 

```
localOut := WORD_TO_BYTE(fromMaster);
toMaster := fromMaster / 10;
localIn;
```

The variable value sent from the master to the slave is output once on the I/O module as a test, divided by 10 and sent back to the master.
11. To compile the program, click **Generate Code** on the “Build” tab (alternatively: function key **F11**).

### Configure the Master Device

1. Create an “Ethernet” fieldbus element for the controller that is to be used as the master. To do so, right-click on the device and select **Insert Device ...**.
2. Select the “Ethernet” fieldbus under “Ethernet Adapter.”
3. Click **[Insert Device]** and leave the dialog open.
 

⇒ A new “Ethernet (Ethernet)” element is created in the device tree.
4. Click on the new element and select the “Modbus TCP Master” for the master device under “Modbus” in the dialog that is still open.
5. Confirm with **[Insert Device]** and leave the dialog open.
6. Click the new “Modbus\_TCP\_Master (Modbus TCP Master)” element in the device tree and insert the “Modbus TCP Slave Device” element from the dialog that is still open.
 

⇒ It is placed in the device tree below the Modbus master and represents the slave device.

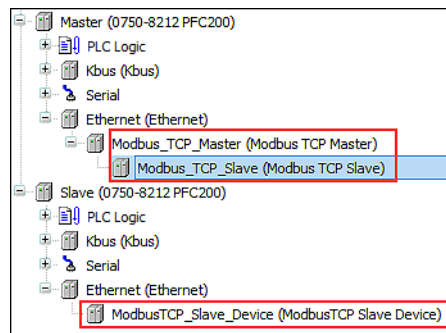


Figure 42: Master and Representative of the Slave

7. To link the master and slave, now enter the IP address of the slave device for the “Modbus\_TCP\_Slave (Modbus TCP Slave)” below the master. To do so, double-click “Modbus\_TCP\_Slave (Modbus TCP Slave)” to open the configuration and enter the IP address of the slave device on the “General” tab.
8. Click the “Modbus Slave Channel” tab and create two channels, which you will use these to specify where and how much to read/write.

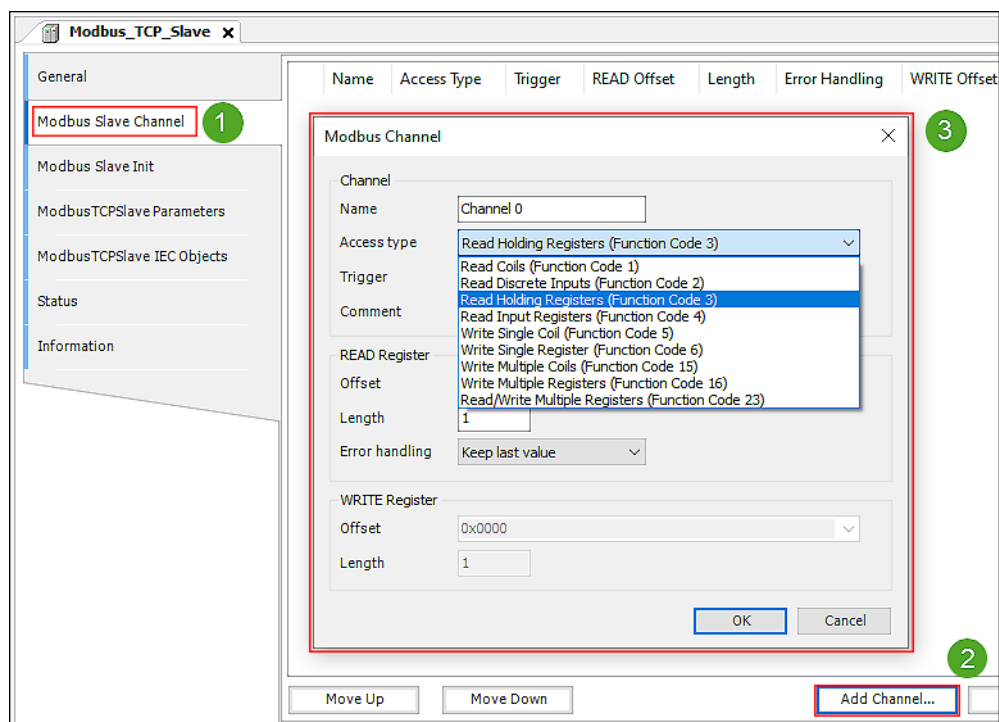


Figure 43: Creating Channels

⇒ Your choice of the access type (or Modbus service) and offset for addressing on what your slave supports. Selecting function codes 4 and 16 is often useful for reading and writing multiple values. For more information about supported Modbus services, refer to the manual of your slave device.

In this example, one channel is created with “WriteSingleRegister” and one channel with “ReadInputRegisters,” since only one value is read and written in this case.

9. Click the “Modbus TCP Slave I/O Image” tab and create two variables here: “toSlave” (WriteSingleRegister) and “fromSlave” (ReadInputRegisters).
10. **Write the Master Program:** To open the program editor, double-click on the “PLC\_PRG” master program in the device tree.



11. Create a simple program:

```
toSlave:=toSlave+1;
fromSlave;
```

One variable value is incremented and one is output.

12. To compile the project, click **Generate Code** on the “Build” tab (alternatively: function key **F11**).

### Connect the Master and Slave Applications Online

1. To connect the application, right-click “Application” for the master and slave respectively and select **[Login]**.
  2. For each, download the program to the device.
  3. Launch the application by right-clicking “Application” and **Start**.
- ⇒ The program is now running.

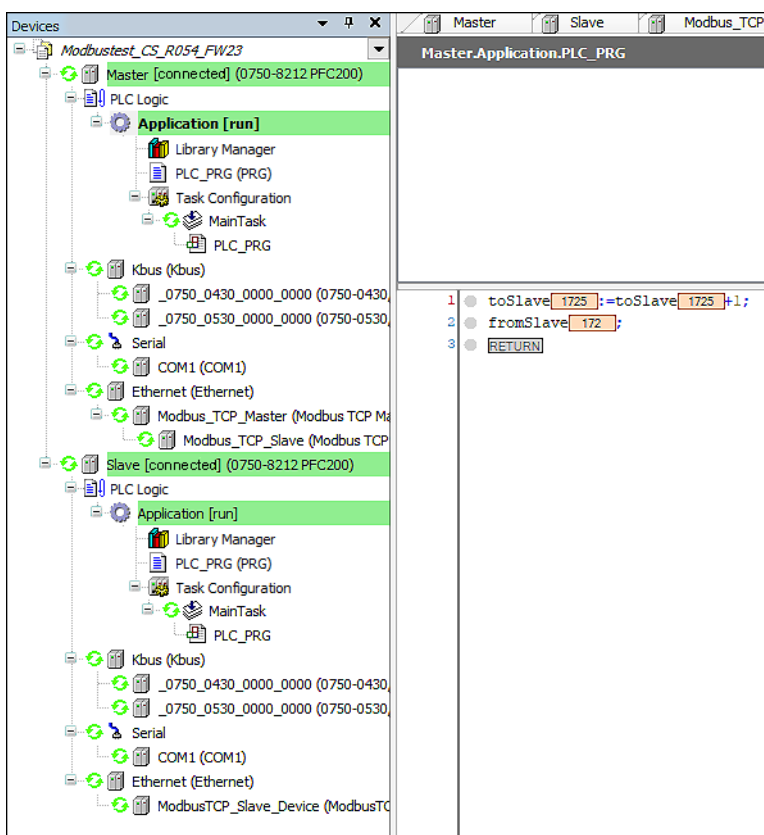


Figure 44: Master Program

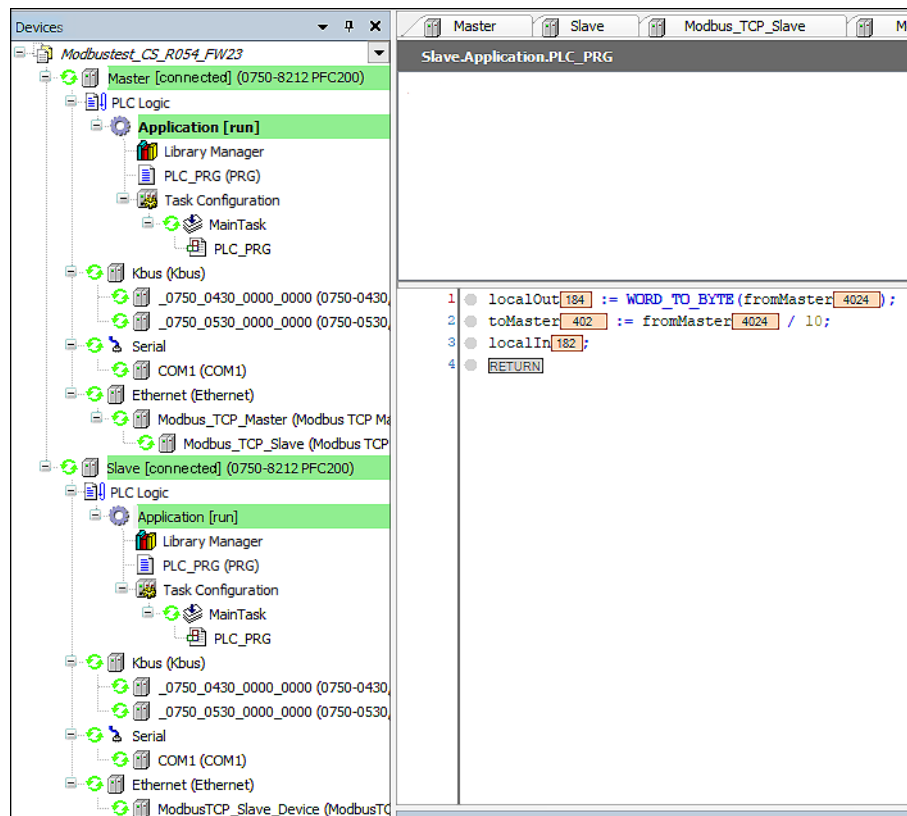


Figure 45: Slave Program

# List of Figures

Figure 1	Modbus Project in <b>e!COCKPIT</b> .....	12
Figure 2	Global Variables via Modbus .....	13
Figure 3	Modbus Master with Modbus Slave .....	13
Figure 4	Adding “Modbus Slave Channel” .....	14
Figure 5	Performing the Mapping.....	14
Figure 6	Assigning Variables .....	15
Figure 7	CANopen Project in <b>e!COCKPIT</b> .....	16
Figure 8	Adding CANopen Manager .....	17
Figure 9	Inserting the Saved CANopen Device under the Newly Created CANopen Manager .	17
Figure 10	Using Variables .....	17
Figure 11	EtherNet/IP Project in <b>e!COCKPIT</b> .....	18
Figure 12	Global Variables.....	19
Figure 13	EtherNet/IP Scanner and Adapter .....	19
Figure 14	Entering the Input/Output Data Size of the Connection .....	20
Figure 15	Input/Output Variables That Have Been Created .....	20
Figure 16	Representation of the Node in WAGO I/O-CHECK .....	20
Figure 17	Hardware Addressing .....	21
Figure 18	EtherCAT Project in <b>e!COCKPIT</b> .....	22
Figure 19	Adding EtherCAT Master .....	23
Figure 20	Inserting the Saved EtherCAT Device under the Newly Created EtherCAT Master ....	24
Figure 21	Using Variables .....	24
Figure 22	Inserting an “Ethernet” Fieldbus.....	29
Figure 23	Inserting Devices.....	30
Figure 24	Modbus Master and Slave Created .....	30
Figure 25	Inserting “WAGO CANopen Device” .....	31
Figure 26	Exporting an EDS File or Installing It Directly in the Device Repository .....	31
Figure 27	Inserting “WAGO CANopen Manager” .....	32
Figure 28	Connected CANopen Devices .....	32
Figure 29	Inserting Ethernet/IP Device .....	33
Figure 30	Configuring EtherNet/IP Interface .....	33
Figure 31	Inserting EtherNet/IP Scanner .....	33
Figure 32	Selecting EtherNet/IP Slave.....	34
Figure 33	Configuring EtherNet/IP Slave .....	34
Figure 34	Separating Ethernet Ports.....	35
Figure 35	Inserting EtherCAT Master.....	35
Figure 36	Specifying the Interface .....	36

Figure 37	EtherCAT Master with Attached Slave.....	36
Figure 38	Inserting an “Ethernet” Fieldbus.....	38
Figure 39	Creating Variables .....	38
Figure 40	Variable “localIn” .....	39
Figure 41	Variable “localOut” .....	39
Figure 42	Master and Representative of the Slave .....	40
Figure 43	Creating Channels .....	40
Figure 44	Master Program .....	41
Figure 45	Slave Program .....	42



**WAGO GmbH & Co. KG**

Postfach 2880 · D - 32385 Minden  
Hansastraße 27 · D - 32423 Minden

✉ [info@wago.com](mailto:info@wago.com)

🌐 [www.wago.com](http://www.wago.com)

Headquarters  
Sales  
Order Service  
Fax

+49 571/887 – 0  
+49 (0) 571/887 – 44 222  
+49 (0) 571/887 – 44 333  
+49 571/887 – 844169

WAGO is a registered trademark of WAGO Verwaltungsgesellschaft mbH.

Copyright – WAGO GmbH & Co. KG – All rights reserved. The content and structure of the WAGO websites, catalogs, videos and other WAGO media are subject to copyright. Distribution or modification of the contents of these pages and videos is prohibited. Furthermore, the content may neither be copied nor made available to third parties for commercial purposes. Also subject to copyright are the images and videos that were made available to WAGO GmbH & Co. KG by third parties.