sysWORXX IO CODESYS

Application Note

Edition March 2025

Document No.: L-2939e_01

SYS TEC electronic AG Am Windrad 2 D-08468 Heinsdorfergrund
Phone: +49 (3765) 38600-0 Fax: +49 (3765) 38600-4100
Web: https://www.systec-electronic.com Mail: info@systec-electronic.com

Changes

Version	Section	Changes
01	-	Initial document

Product names used in this manual which are also registered trademarks have not been marked extra. The missing © mark does not imply that the trade name is unregistered. Nor is it possible to determine the existence of any patents or protection of inventions on the basis of the names used.

The information in this manual has been carefully checked and is believed to be accurate. However, it is expressly stated that SYS TEC electronic AG does not assume warranty or legal responsibility or any liability for consequential damages which result from the use or contents of this user manual. The information contained in this manual can be changed without prior notice. Therefore, SYS TEC electronic AG shall not accept any obligation.

Furthermore, it is expressly stated that SYS TEC electronic AG does not assume warranty or legal responsibility or any liability for consequential damages which result from incorrect use of the hardware or software. The layout or design of the hardware can also be changed without prior notice. Therefore, SYS TEC electronic AG shall not accept any obligation.

Contact	**Direct**	**Your Local Distributor**
Address:	SYS TEC electronic AG Am Windrad 2 D-08468 Heinsdorfergrund GERMANY	Please contact us for information about your local distributor. info@systec-electronic.com
Ordering Information:	+49 (0) 37 65 / 38 600-0 info@systec-electronic.com	
Technical Support:	+49 (0) 37 65 / 38 600-0 support@systec-electronic.com	
Fax:	+49 (0) 37 65 / 38 600 4100	
Web Site:	https://www.systec-electronic.com	

Table of Contents

List of Tables

List of Figures

1 Introduction

The following document provides a comprehensive introduction to working with CODESYS on a sysWORXX device. The goal is to show you step-by-step how to optimally prepare and configure your system for the use of CODESYS. We will first discuss the installation of CODESYS and explain the necessary requirements and preparations on the sysWORXX device.

The rest of the document will show you how to configure a sysWORXX device specifically for use in a CODESYS environment. This includes, among other things, adjusting system settings, installing the required libraries, and setting up communication between hardware and software.

The creation of a simple CODESYS program is also emphasized. You will learn how to build a basic control program – from development and debugging to successful commissioning.

The guide also covers the integration of communication protocols such as EtherCAT, Modbus, and CANopen. This allows you to integrate your sysWORXX device into sophisticated industrial applications and network it with other intelligent devices.

Finally, the installation and use of a special extension is presented, which offers you two different visualization options: local visualization via a directly connected display and web-based visualization that is accessible from anywhere.

This documentation is aimed at beginners and experienced users alike and is intended not only to make getting started easier, but also to enable you to successfully implement your own projects with CODESYS on a sysWORXX device. Immerse yourself in the world of modern control technology and discover the diverse possibilities opened up by this combination.

Table 1 lists up some other relevant manuals for the sysWORXX devices.

Table 1: Overview of relevant manuals

Information about…	**In which manual?**
Basic information about the sysWORXX IO CODESYS	In this manual
Basic information about the device (configuration, administration, process image, connection assignment, firmware update, reference designs et cetera)	• System Manual sysWORXX series – L-2932 • System Manual sysWORXX Pi-AM62x – L-2934
Technical documentation for the specific sysWORXX devices	• sysWORXX CTR-600 – L-2941 • sysWORXX CTR-800 – L-2933
Quick start Guide for sysWORXX devices	• L-2935

For more information, optional products, updates et cetera, we recommend you to visit our website: https://www.systec-electronic.com. The content of this website is updated periodically and provides you with downloads of the latest software releases and manual versions.

2 Installation

2.1 CODESYS

To control the sysWORXX IO, the CODESYS Deployment System V3 (v3.5.21.10 or higher) must be installed on a Windows PC.

Next install the necessary CODESYS RunTime. To set up the project example following packages must be installed on the host machine:

- CODESYS Virtual Control for Linux SL[1]
- CODESYS Control for Linux ARM64 SL[2]

For this, start CODESYS and open the CODESYS installer with: *Tools->CODESYS installer*. Before a new package can be installed, it must be restarted as administrator.

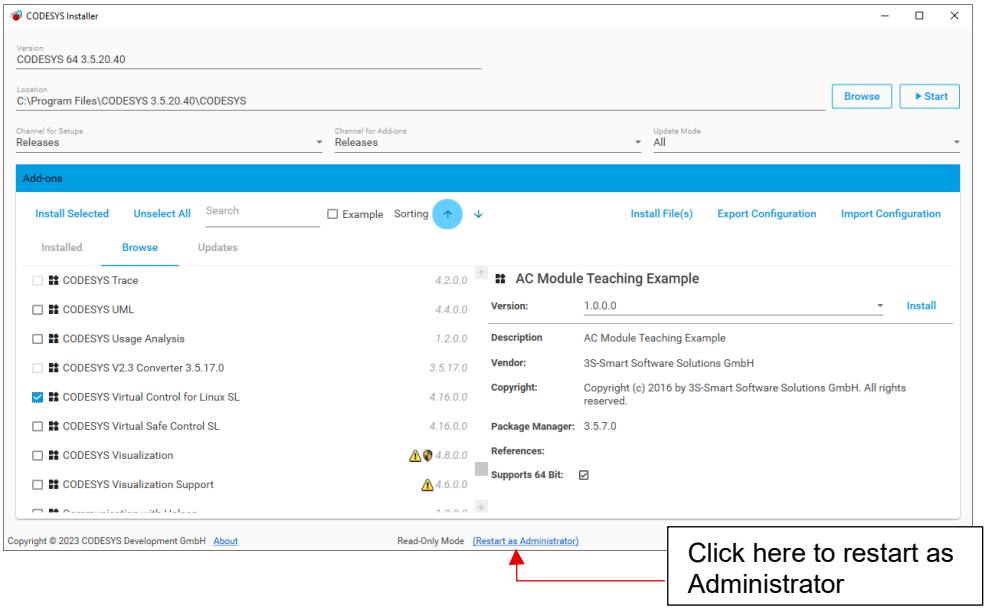


Figure 1: CODESYS Demo: CODESYS installer setup

[1] This package contains a Soft-PLC running in a Docker container. The required Docker runtime is already installed on the device.
[2] This package installs additional libraries required for ARM64 systems.

Then select *Browse* and the listed packages above.

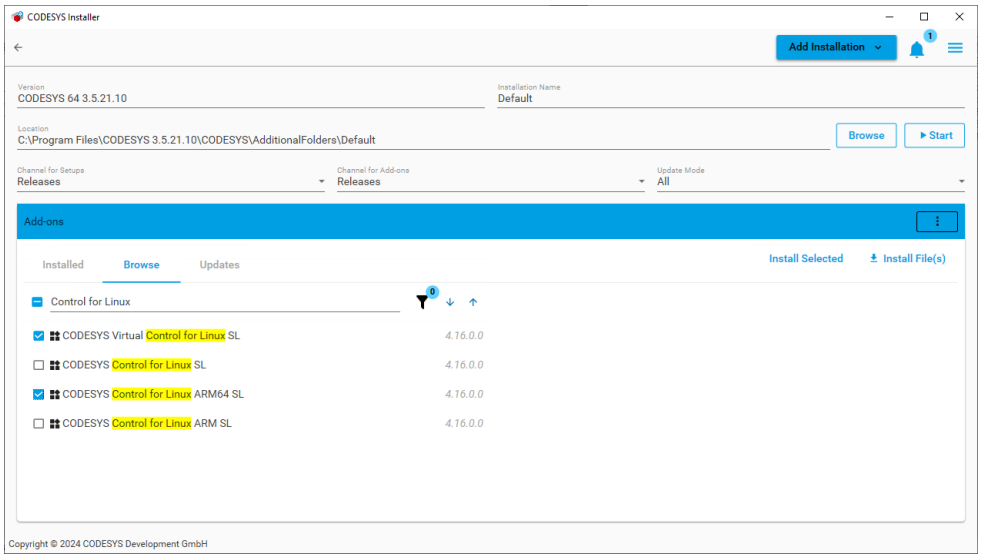
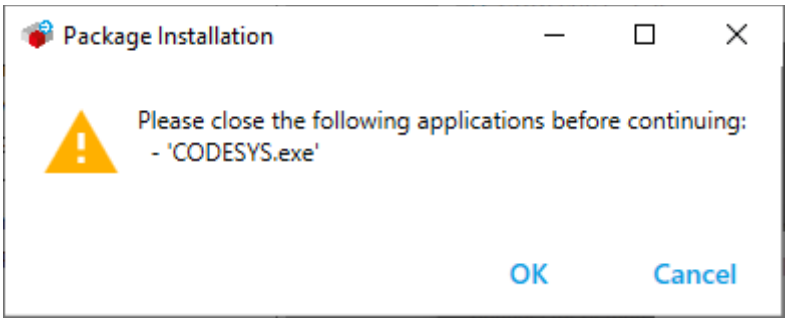


Figure 2: CODESYS Demo: CODESYS installer setup

HINT: CODESYS.exe must be terminated during installation (running CODESYS installer) and can be restarted after installation of the new packages using the *CODESYS installer*.



2.2 sysWORXX IO connector for CODESYS

In modern automation systems, the CODESYS runtime system forms the central element for controlling and regulating processes. To enable this system to communicate with the diverse hardware landscape of sysWORXX devices, a dedicated software component ("*sysWORXX IO CODESYS connector*") exists that enables data exchange between the runtime system and the specific hardware drivers.

The sysWORXX IO CODESYS connector coordinates both read and write accesses, thus ensuring synchronized data exchange. As soon as the runtime system generates a task or instruction, the component receives it, formats it according to the requirements of the target driver, and then forwards it to the corresponding device. Similarly, it receives the feedback and measurement data from the hardware, converts it into a format compatible with the CODESYS runtime system, and makes it available to the control logic.

The sysWORXX IO CODESYS connector has a key advantage: It reduces the complexity of direct hardware connections, enables consistent error handling, and facilitates the integration of new devices, since changes or extensions must primarily be made in the abstraction layer. It thus makes a significant contribution to the operational reliability and maintainability of modern CODESYS-supported applications.

By default, the sysWORXX IO CODESYS connector are preinstalled on each sysWORXX device, but not running. To start the sysWORXX IO CODESYS connector, the *codesys-connector* system service must be running. The steps in chapter 3.1.2 show how to start the service every time the device reboots.

3 Getting started

3.1 Getting started with sysWORXX device

3.1.1 Setup the sysWORXX device

Login to the device with a Terminal program (for example Tera term, PuTTY) The steps below are described in detail in Quick start guide (see chapter 1).

Add a new user:

```
$ useradd codesysuser
```

Add a password for the new user:

```
$ passwd codesysuser
New password:********
Retype new password:********
passwd: password updated successfully
```

Add the new user to sudoers group:

```
$ echo "codesysuser ALL=(ALL:ALL) ALL" > /etc/sudoers.d/00_codesysuser
$ logout

login: codesysuser
Password: **********

codesysuser@sysworxx:~$
```

Connect the sysWORXX device to a network and get the IP address of the device using nmcli-command:

```
codesysuser@sysworxx:~$ nmcli

eth0: connected to Wired connection 1
        "eth0"
        ethernet (am65-cpsw-nuss), 44:D5:F2:A0:5B:AC, hw, mtu 1500
        ip4 default
        inet4 192.168.15.147/24
        route4 192.168.15.0/24 metric 100
        route4 192.168.10.8/32 metric 100
        route4 192.168.115.0/24 via 192.168.10.8 metric 100
        route4 default via 192.168.15.252 metric 100
        inet6 fe80::6030:d3e0:d077:c399/64
        route6 fe80::/64 metric 1024
```

Test whether an SSH connection can be established to the new user. Open PowerShell (under Windows) or Terminal (under Linux):

```
> ssh codesysuser@192.168.15.147
codesysuser@192.168.15.147's password:

codesysuser@sysworxx:~$
```

3.1.2 Start sysWORXX IO CODESYS connector

Login to the device with a command shell via SSH or a Terminal program.

The device offers a connector, to use the IO's of the device inside CODESYS project. For this, there is the service `codesys-connector` and `codesys-generate-xml-devdesc`.

To see the status of each service, the following command must be executed:

```
$ sudo systemctl status codesys-connector
```

If the service is stopped or faulty, it can be (re-)started with:

```
$ sudo systemctl start codesys-connector
```

When using CODESYS, the codesys-connector service must be running. It can be enabled, to start with the system on each boot:

```
$ sudo systemctl enable codesys-connector
```

The command shell is not needed anymore and can be closed.

The service `codesys-generate-xml-devdesc` generates an XML file, which can be used to access the IO's in CODESYS projects. If not enabled, the service has to be started manually as seen above. The XML file for sysWORXX IO is located on the device under*: /tmp/sysworxx-xxxxx.devdesc.xml*. This file must be downloaded from the device and is later integrated into CODESYS.

3.2 Getting started with CODESYS

To use sysWORXX devices with CODESYS, one has set up the project environment to be usable with Linux ARM64 devices. The steps necessary are described below.

3.2.1 Project setup for sysWORXX device

Start CODESYS, choose a workspace and create a new *CODESYS Project*. Select the "*CODESYS Virtual Control for Linux ARM64 SL*" device (Notice: If is not yet installed, see chapter 2.1. This device can be updated after installation.)

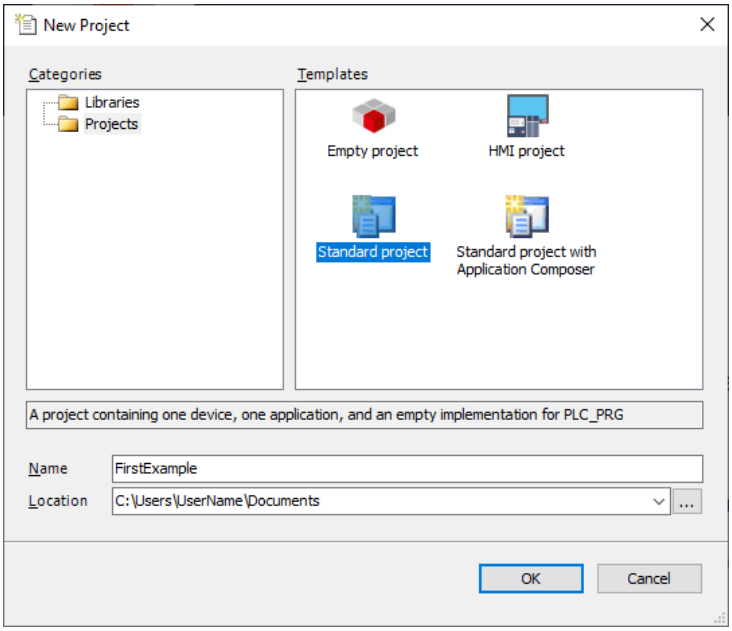


Figure 3: CODESYS Demo: Project setup

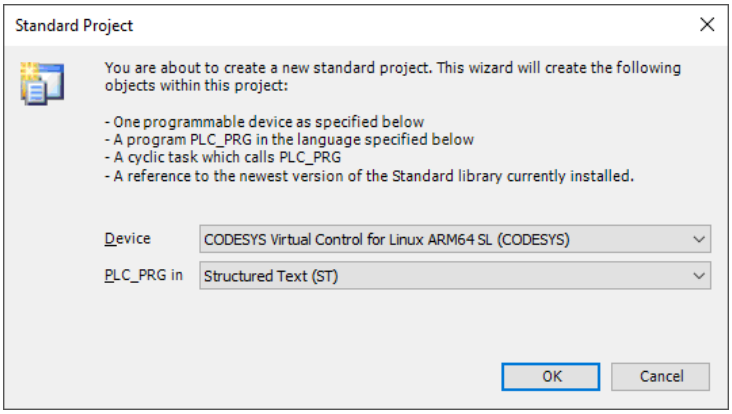


Figure 4: CODESYS Demo: Project setup

3.2.2 Establish communication with sysWORXX device

Next, it need to connect to the device to install the Docker image with the Soft-PLC. To do this, proceed as follows:

Open *Tools->Deploy Control SL* to establish a SSH communication with the sysWORXX device. The IP address, user name and password must be entered in the Communication tab (Linux user name and password, as added in accordance with chapter 3.1.1). One can then connect to the device.

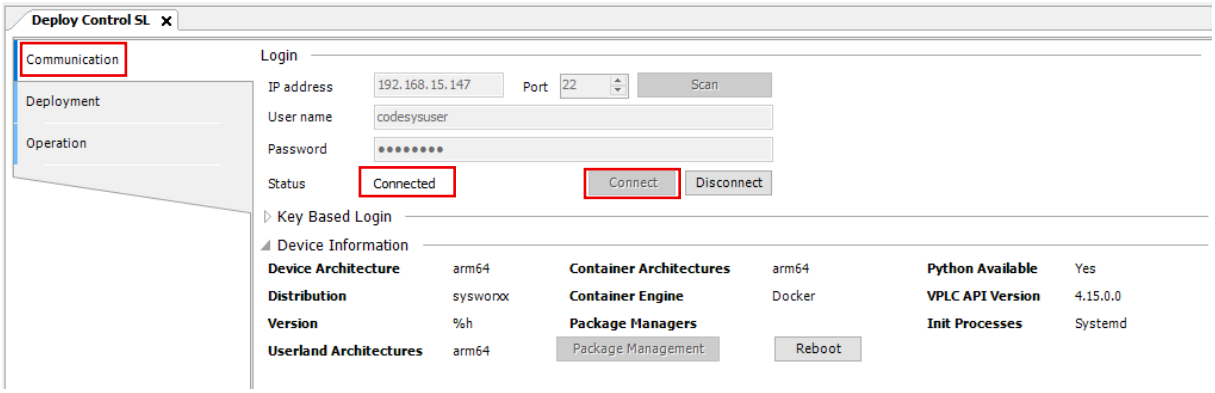


Figure 5: CODESYS Demo: Deploy Control SL (communication)

The next steps involve installing the Soft-SPS as Docker image on the device (create the Docker Image and download to the device). The product and version must be installed in the deployment menu. A new product “CODESYS Virtual Control for Linux SL” then appears under installed images.

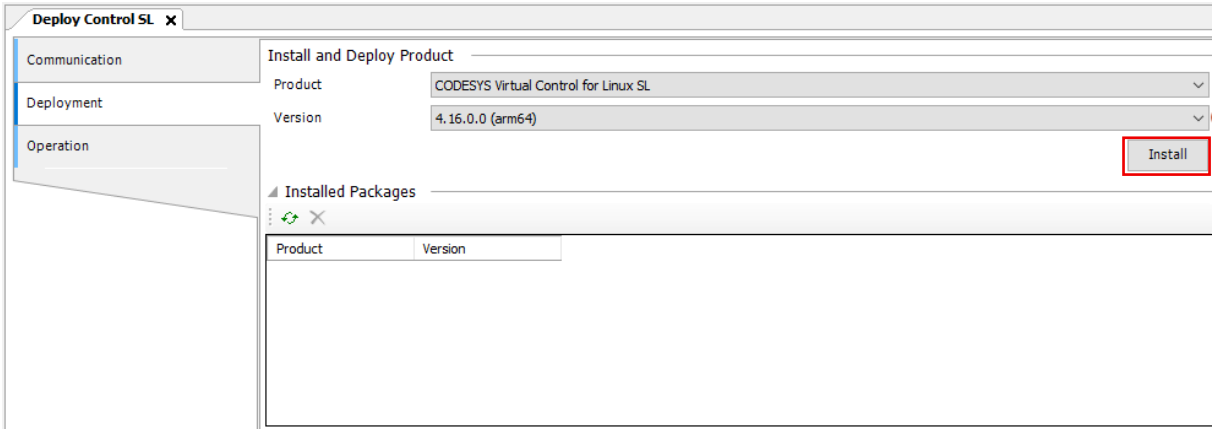


Figure 6: CODESYS Demo: Deploy Virtual Control SL (Docker Image)

Then change to Operation Menu to create a new Docker Container on the device. A new Docker container with a corresponding name can be created via the VPLCs button.

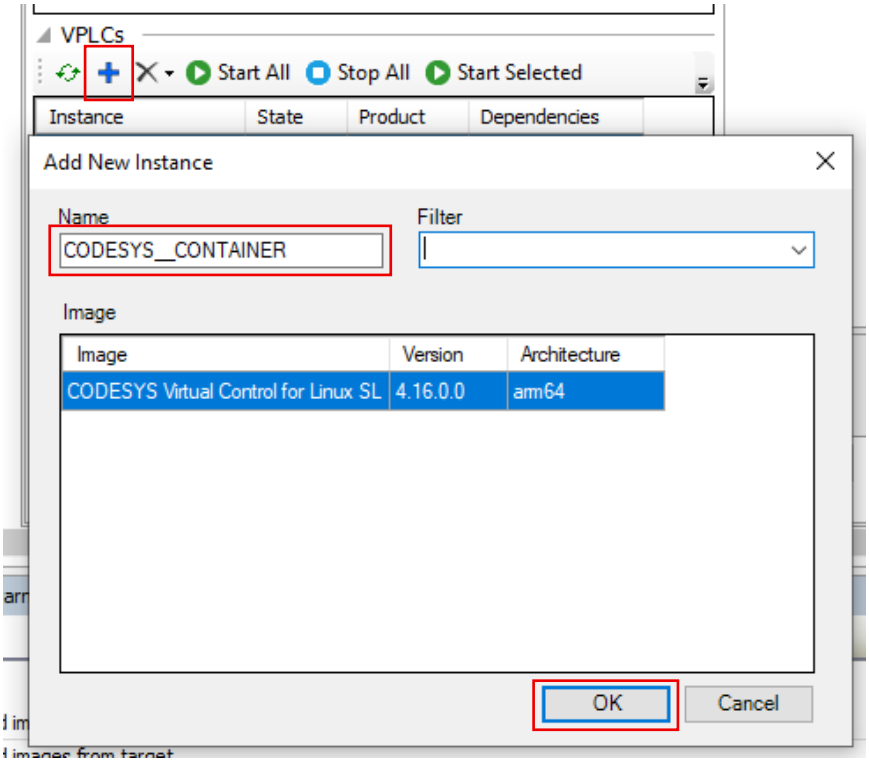


Figure 7: CODESYS Demo: Deploy Virtual Control SL (Create Docker container)

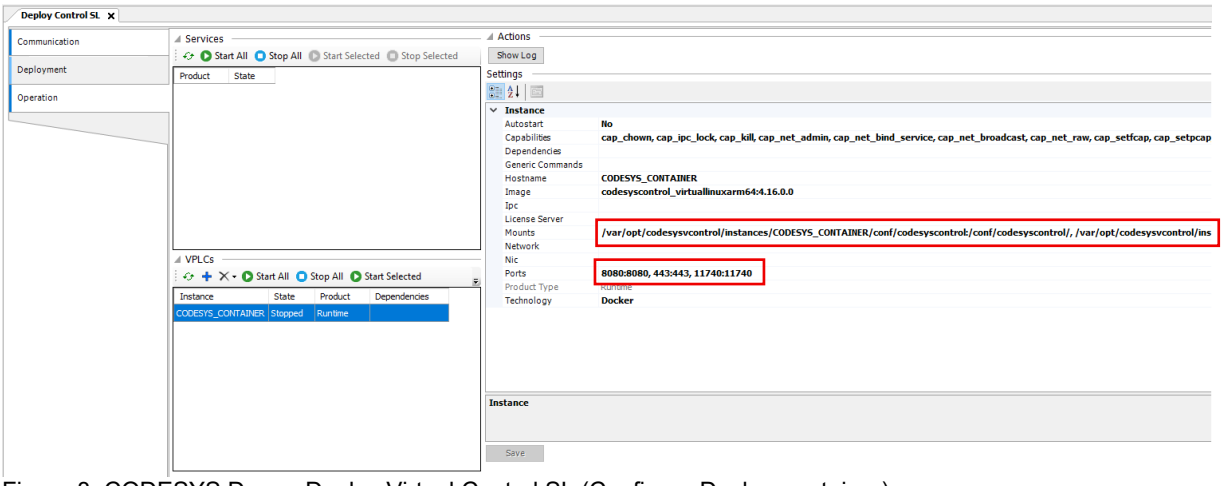


Figure 8: CODESYS Demo: Deploy Virtual Control SL (Configure Docker container)

The following parameters must be added or replaced in the Docker instance:

- **Mounts:**
  `/var/opt/codesysvcontrol/instances/CODESYS_CONTAINER/conf/codesyscontrol:/conf/codesyscontrol/,/var/opt/codesysvcontrol/instances/CODESYS_CONTAINER/data/codesyscontrol:/data/codesyscontrol/,/var/run/codesysextension/extfuncs/:/var/run/codesysextension/extfuncs/`

- **Ports:** `8080:8080, 443:443, 11740:11740`

Hint: In the parameter window, other Docker specific changes can be made, if these are necessary for the runtime. For example, one could one could forego the port declaration and use the keyword "*host*" in the "Network" option instead – this would not only expose the port but every network interface on the device (eth, can, …). More information about the features available can be found in the Docker documentation: https://docs.docker.com/engine/containers/start-containers-automatically/

Once the Docker has been successfully configured, the Docker instance can be started (select the *Instance* and click *Start Selected*).

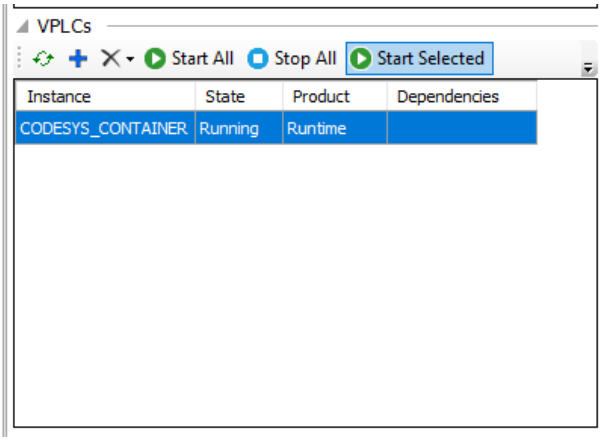


Figure 9: CODESYS Demo: Deploy Virtual Control SL (Start Docker container)

The Docker container, and thus the CODESYS RunTime, is now running on the sysWORXX device. Next, the interface to the process data and the connection to the Soft-PLC in the Docker container must be installed.

3.2.3 Add sysWORXX device description to the project

Implementing a PLC program requires a description of the software interface for accessing the process data (hardware). The next steps describe how to install this interface.

Under *Tools->Device Repository,* the current XML file[3] of the sysWORXX device can be selected via the install button. All sysWORXX IOs for this device are defined in this XML file.

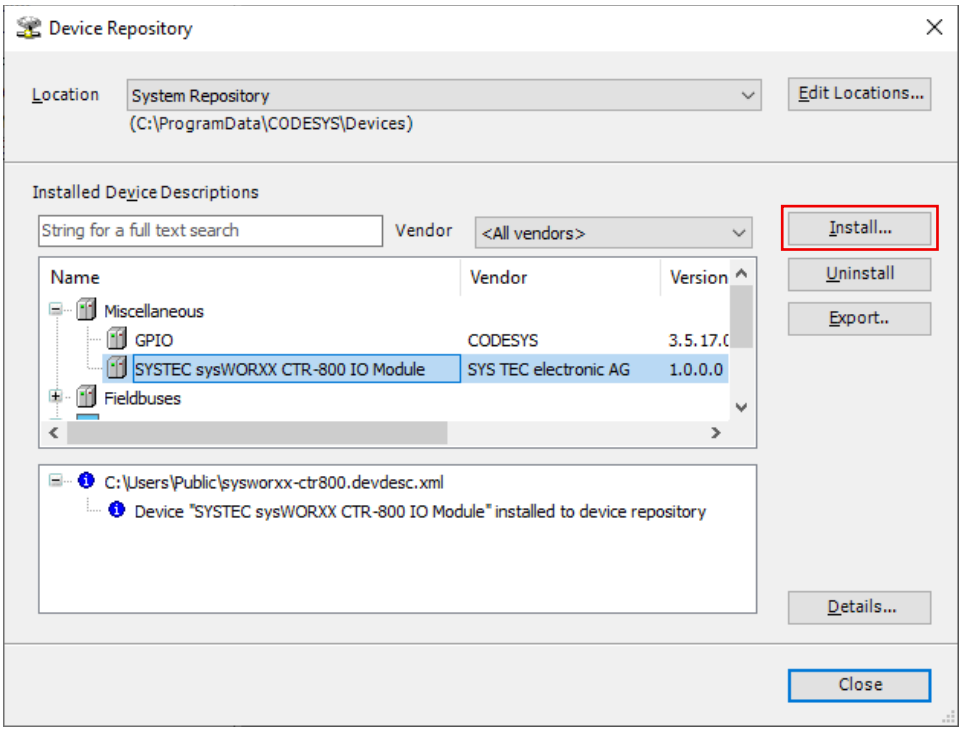


Figure 10: CODESYS Demo: Device repository

This device can be attached and selected under Device -> *right-click on Device*. The corresponding XML file can be found under the various tab.

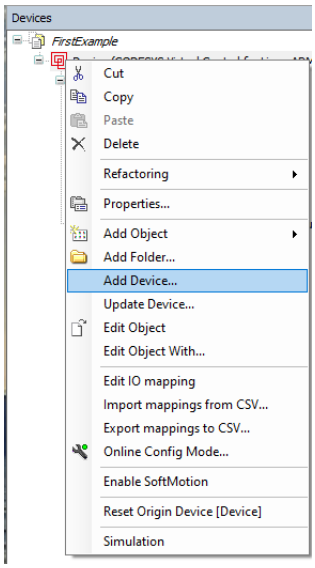


Figure 11: CODESYS Demo: Attach device

[3] The XML file can be loaded from the SYS TEC electronic website.

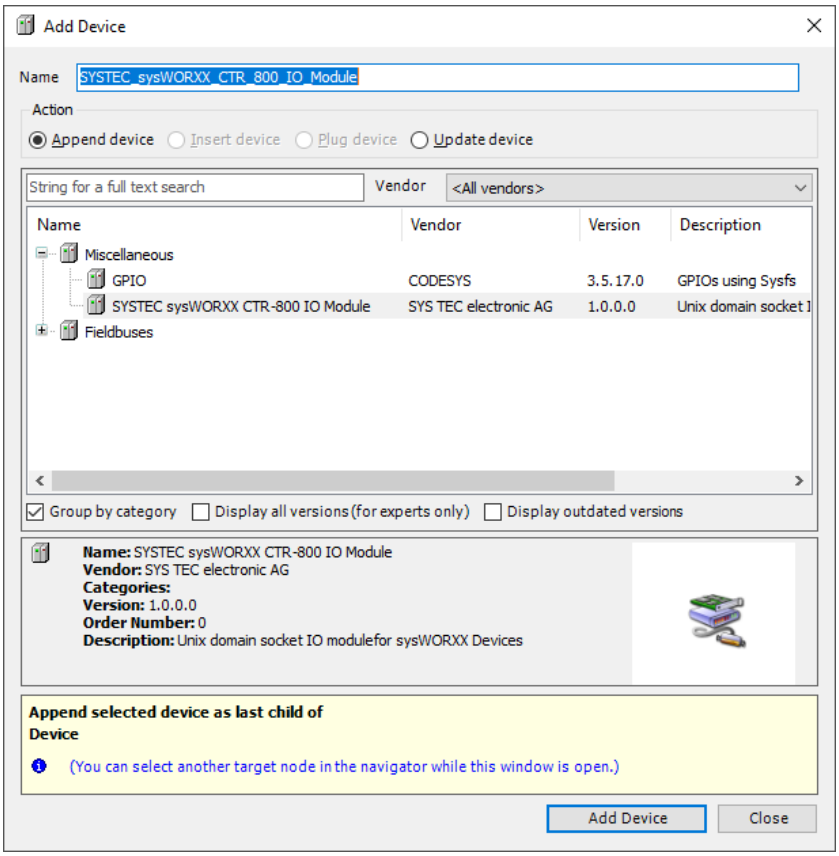


Figure 12: CODESYS Demo: Select the installed device

The corresponding sysWORXX-IO list can be called up by attaching the XML file.

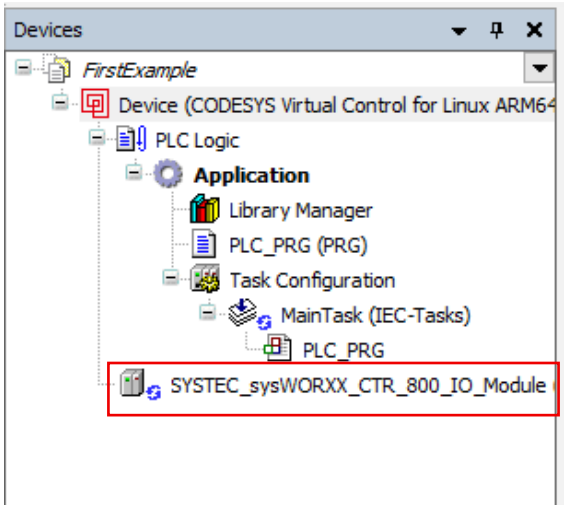


Figure 13: CODESYS Demo: Installed device decription

3.2.4 Establish communication with the Soft-PLC on the device

To load a PLC program, a connection must be established with the RunTime of the Soft-PLC.

Device -> double-click on Device

The IP address of the device must be stored in the communication tab. It is important to enter the port after the IP address. In addition, a user name and password for CODESYS are required for first-time users.

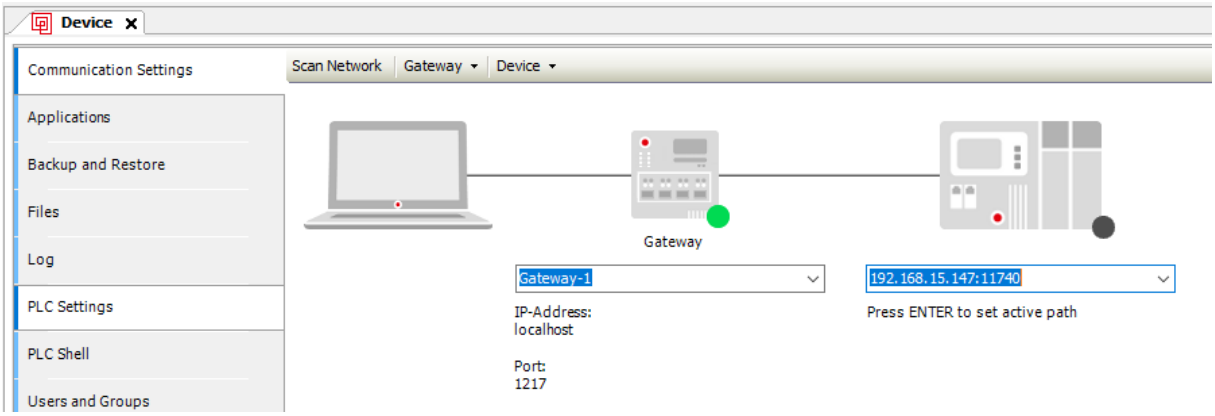


Figure 14: CODESYS Demo: Establish Communication to the Soft-PLC

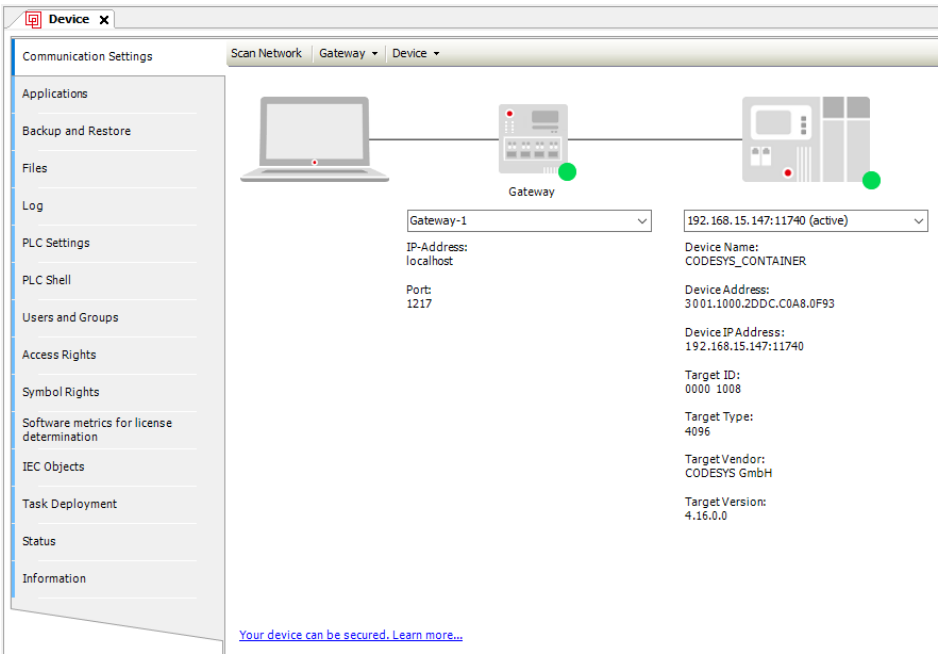


Figure 15: CODESYS Demo: Successful communication to the Soft-PLC

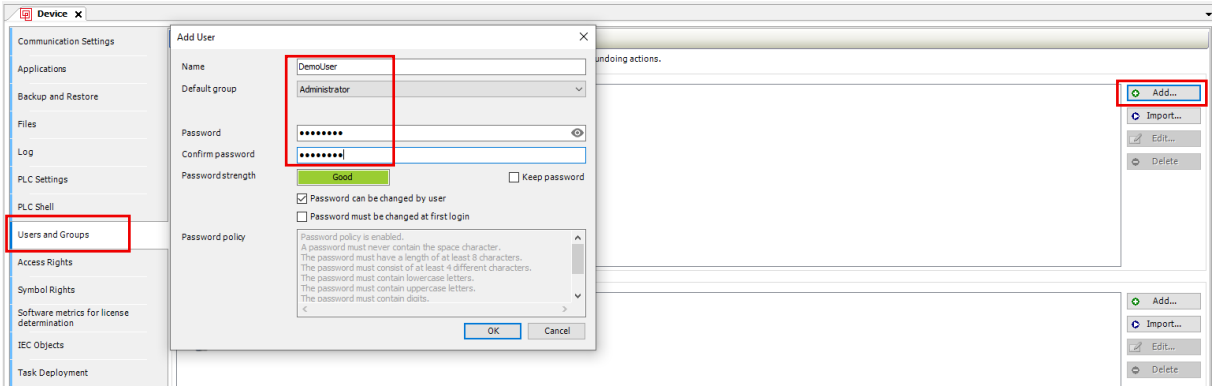


Figure 16: CODESYS Demo: Add user name and password

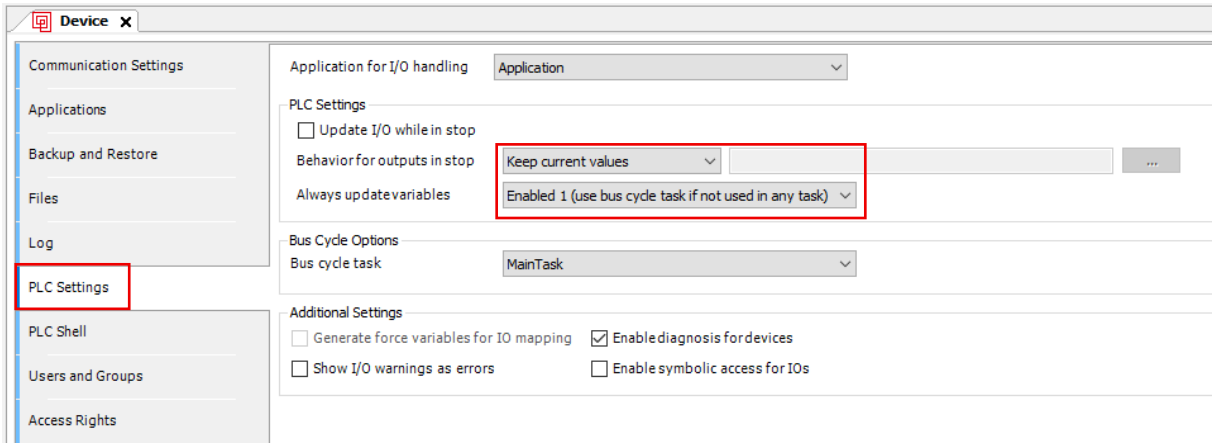


Figure 17: CODESYS Demo: SPS-Settings

Once the configuration of the CODESYS program has been successfully completed, a corresponding program *(PLC_PRG)* can be written for the sysWORXX device. This example program refers to the sysWORXX Pi-AM62x.

3.2.5 Write the first PLC program

PLC_PRG | SYSTEC_sysWORXX_CTR_800_IO_Module

```
1   PROGRAM PLC_PRG
2   VAR
3       Relay_State   AT %QX2.0 : BOOL := FALSE;
4       Relay_Release AT %QX5.0 : BOOL := TRUE;
5       CurrentCycle : USINT := 0;
6   END_VAR
```

```
1   CurrentCycle := CurrentCycle +1;
2
3   IF CurrentCycle >= 0 THEN
4
5       Relay_State := TRUE;
6   END_IF
7
8   IF CurrentCycle >= 150 THEN
9
10      Relay_State := FALSE;
11  END_IF
12
13  RETURN;
```

Figure 18: CODESYS Demo: Code example for switch Relay

In this program, the Relay are switched one after the other according to a specific cycle. To run the code on the device, the code must be generated, logged in and the program started.

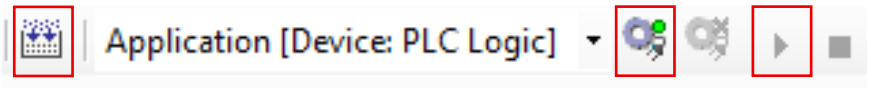


Figure 19: CODESYS Demo: Build, Login and Start

3.2.6 Licensing

If not licensed, the "CODESYS Virtual Control for Linux ARM64 SL" SoftPLC runtime operates 2 hour and then stops. A license is needed to make it run an unlimited time.

We recommend licensing Control Standard S (119€) with CANopen/Modbus Serial, TCP/J1939 (4 instances) and 1 instance EtherCAT, Ethernet/IP, PROFINET. No optional features included such as communication, Softmotion axes, visualization, safety and Softmotion axis groups/CNC.

In CODESYS, however, the currently required license can also be determined automatically using the software metrics. To determine this, open the tab Software metrics for license determination under: *Geräte-> double-click on Device*.

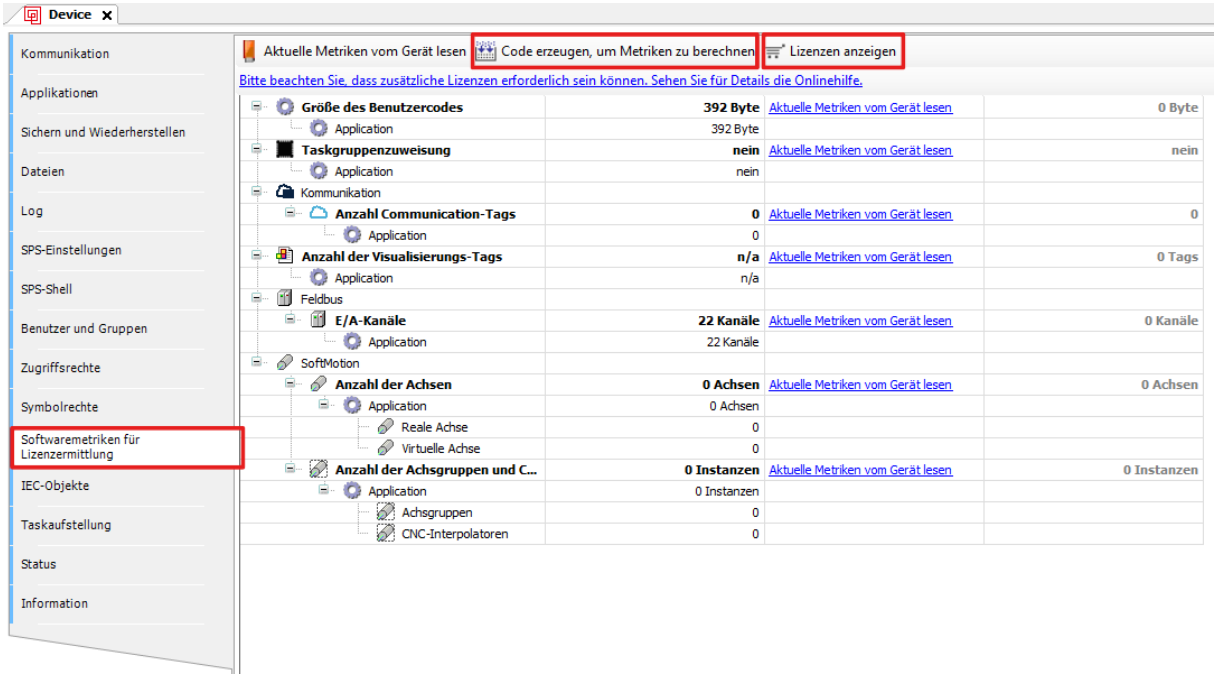


Figure 20: Licensing: Software metrics for license determination

First, the code must be generated in order to calculate the metrics (“Code erzeugen, um Metriken zu berechnen”). One can then press the Display licenses button (“Lizenzen anzeigen”).

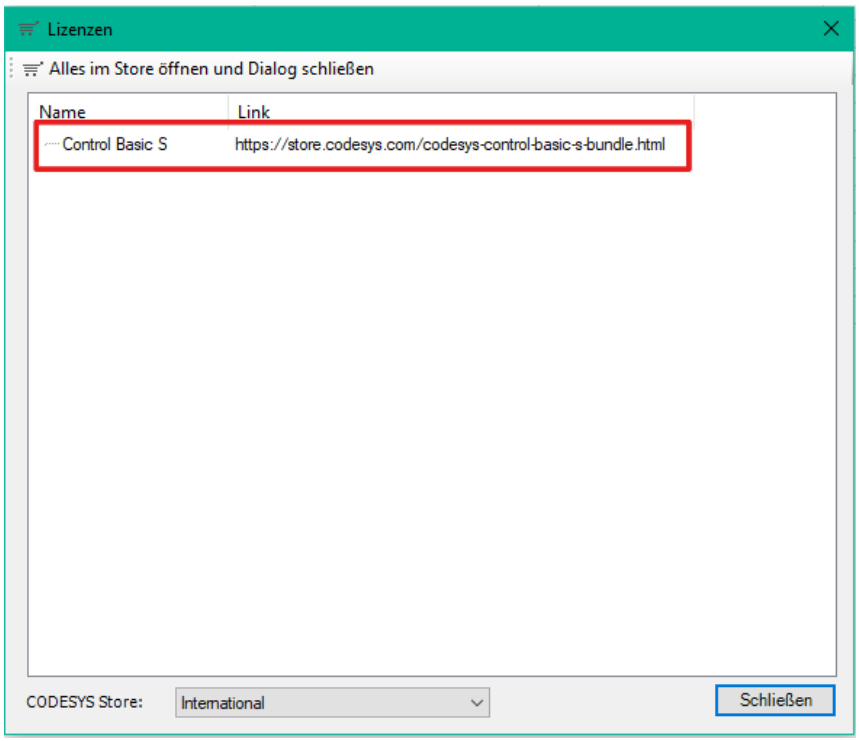


Figure 21: Licensing: License determination

The corresponding recommended license is then displayed. Double-click on the license to go straight to the CODESYS Store to view the price and components. The corresponding license can also be purchased.

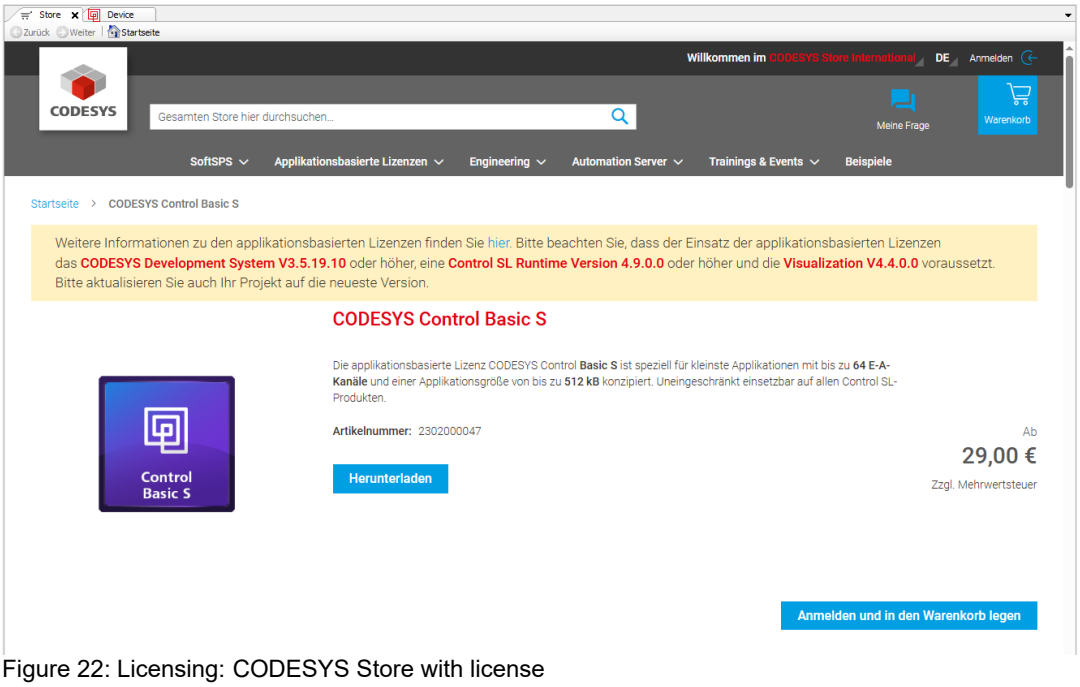


Figure 22: Licensing: CODESYS Store with license

A license purchase follows an email with a ticket number (e.g A78HY-TBVMD-8SVC7-P8BHX-4LED6) granting one the license. The *Tools->License Manager* in the “CODESYS Development System V3” (internet needed) transforms the ticket number in a deployed license. The ticket number can be used only one time and gets invalid during the licensing procedure.

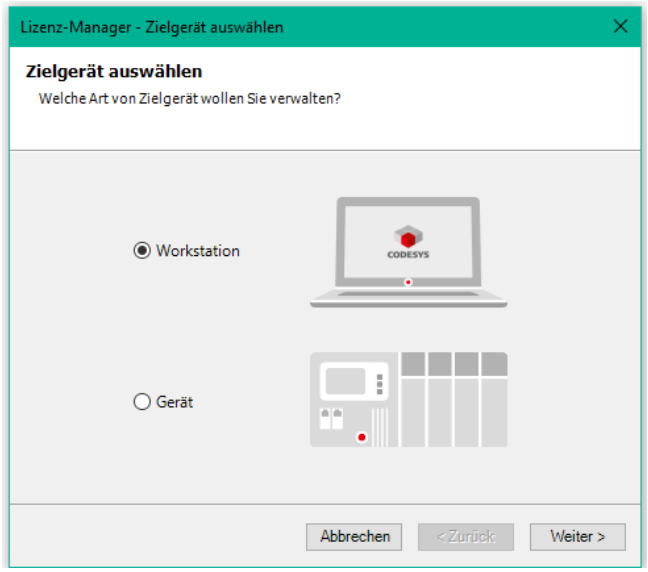


Figure 23: Licensing: Select target device

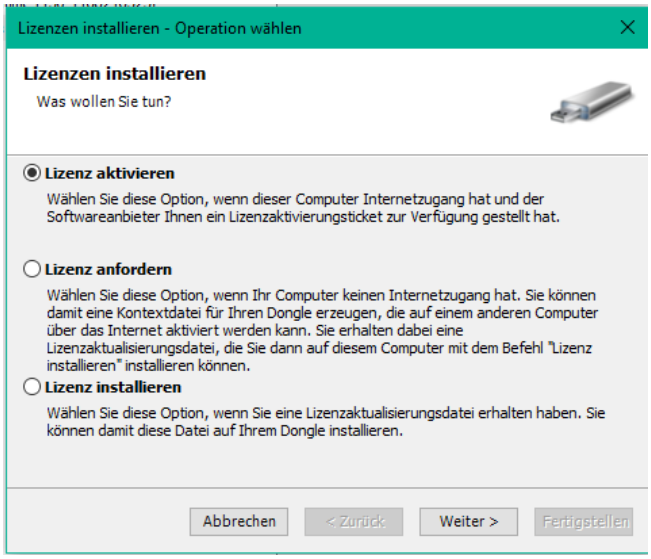


Figure 24: Licensing: Install licenses

It is possible to deploy a license to either a CODESYS Runtime Key (USB dongle) or to a software container.

4 CODESYS fieldbus interface support

In addition to the functionality of an IEC 61131-3 development system for programming industrial controllers, CODESYS also offers an extensive fieldbus support including everything from communication POUs to configurators and portable protocol stacks. The following 3 fieldbuses are described in more detail in their mode of operation.

4.1 EtherCAT

EtherCAT is a powerful real-time Ethernet fieldbus system with an open protocol. With its short cycle times, low jitter values and different network topologies, the system is standard in many industrial automation applications today.

The following integration of EtherCAT in CODESYS is illustrated using a motor (JMC Servermotor mit integr. Servotreiber und EtherCAT). The XML file for the device description is enclosed with the product or can be downloaded from the internet.

Once the project is deployed it needs an upgrade with the following packages one can download from *Tools->CODESYS installer*. Before a new package can be installed, it must be restarted as administrator. The following package must then be installed:

- CODESYS EtherCAT

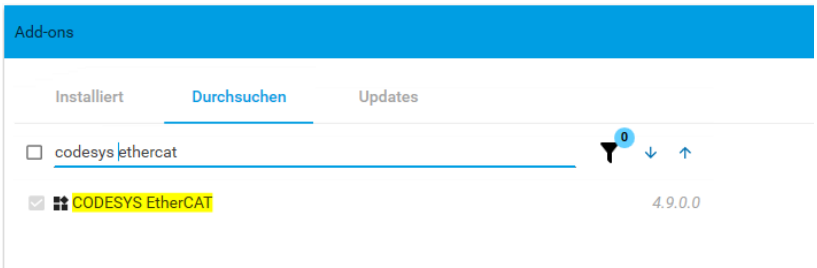


Figure 25: EtherCAT Demo: CODESYS installer

CODESYS must be terminated during installation and can be restarted after installation using the *CODESYS installer*.

If not already done, one has to follow the steps described in section 3.2.1. Also following parameters must be set instead of the ones described in section 3.2.1.

The following parameters must be changed in the Docker instance:

- `Mounts: /var/opt/codesysvcontrol/instances/CODESYS/conf/codesyscontrol:/conf/codesyscontrol/, /var/opt/codesysvcontrol/instances/CODESYS/data/codesyscontrol:/data/codesyscontrol/, /var/run/codesysextension/extfuncs/: /var/run/codesysextension/extfuncs/`
- `Network: host`

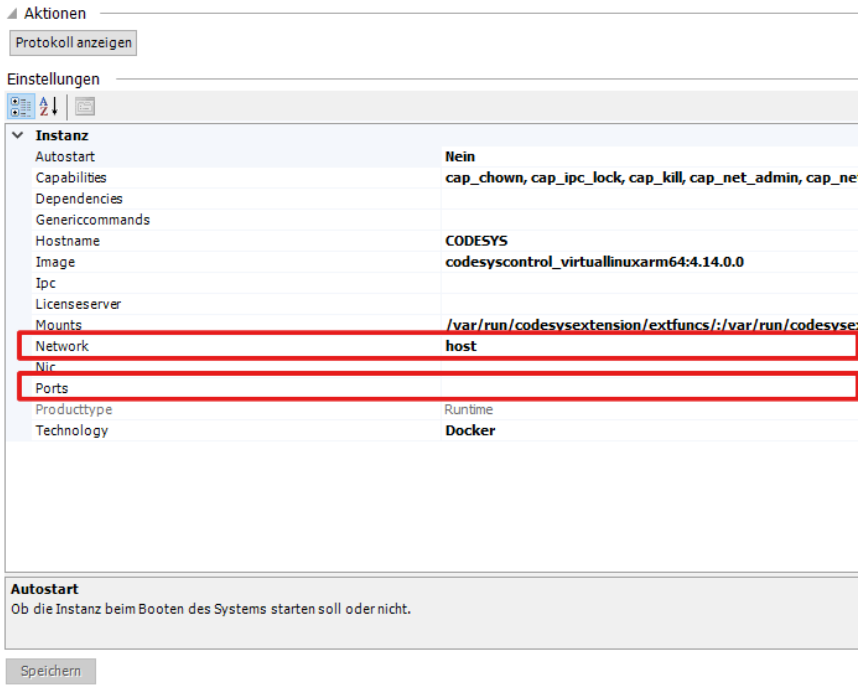


Figure 26: EtherCAT Demo: Control SL roll out Docker script

Once the Docker has been successfully configured, the Docker instance can be started. The EtherCAT fieldbus can be attached and selected under: *Geräte -> right-click on Device -> Geräte anhängen*.

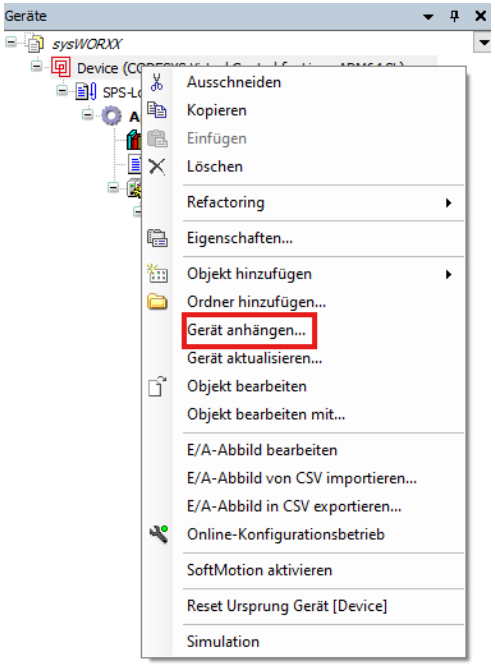


Figure 27: EtherCAT Demo: Attach device

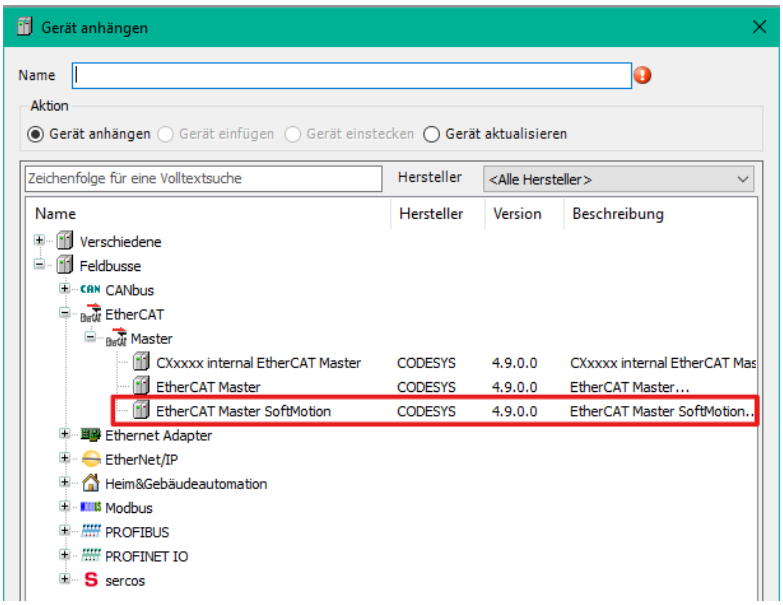


Figure 28: EtherCAT Demo: Select EtherCAT

Under *Tools->Geräte-repository,* the current XML file of the device can be selected via the install button.

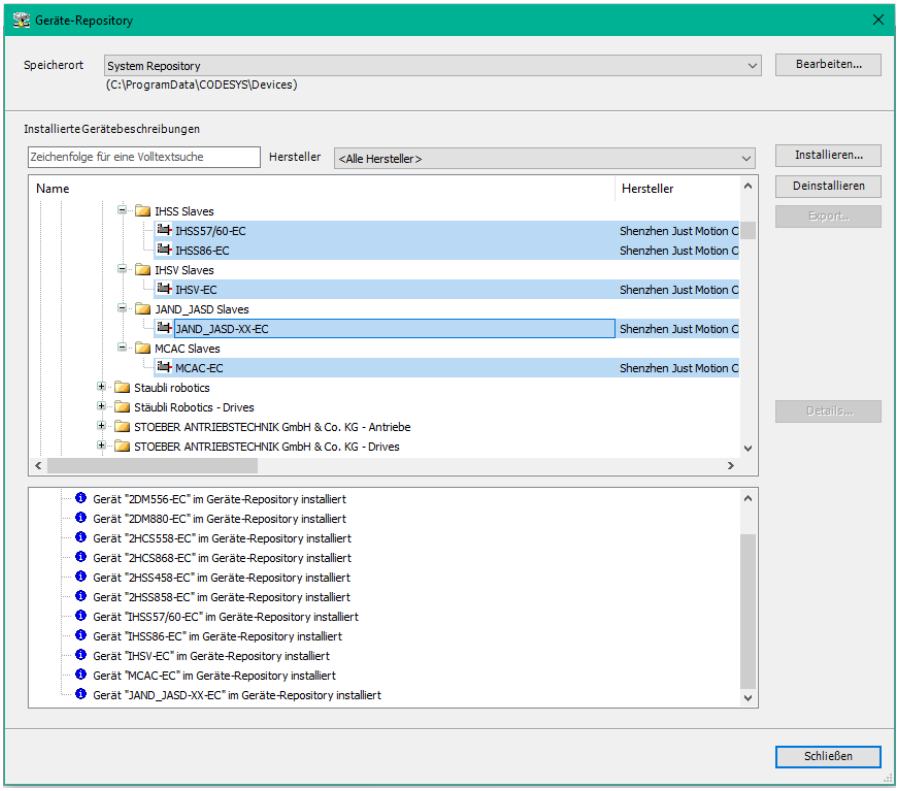


Figure 29: EtherCAT Demo: Device repository

The device can be configured under: *Geräte -> double-click on EtherCAT*. The following settings must be made in the General tab:

- Offset
- Network name
- Source address (MAC), search via network name

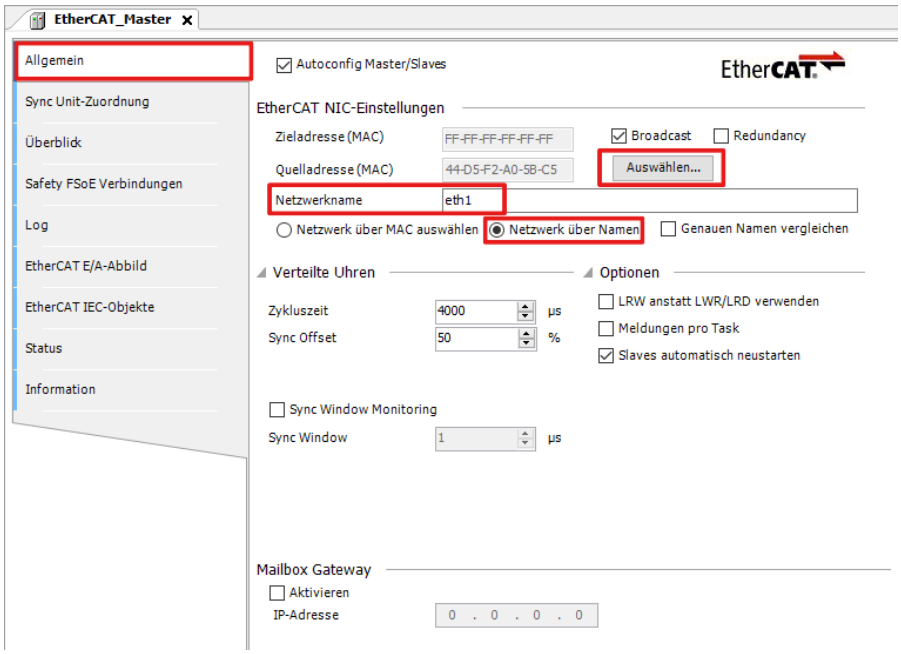


Figure 30: EtherCAT Demo: EtherCAT general tab

After this, the device can be attached (*Geräte-> right-click on EtherCAT ->Gerät anhängen)*. If the device is already connected, the device can be searched for under: *Geräte-> right-click on EtherCAT ->Gerät suchen*.

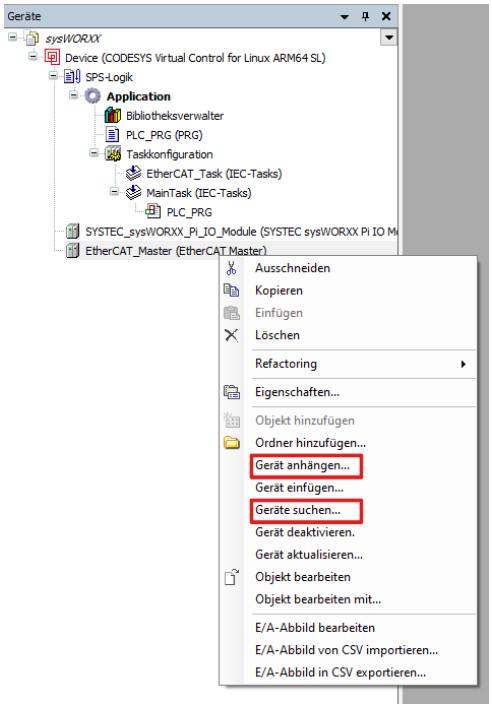


Figure 31: EtherCAT Demo: Attach device to EtherCAT

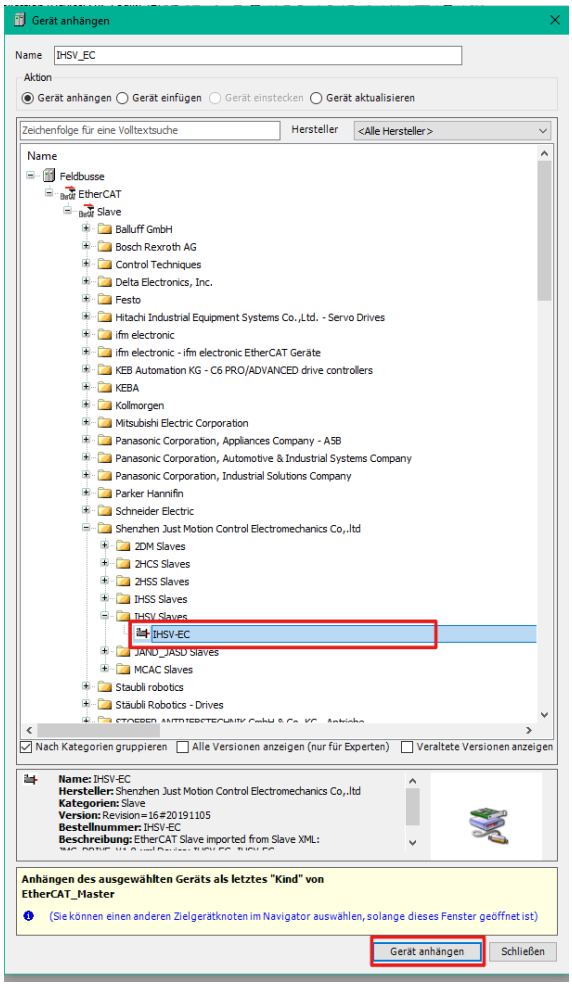


Figure 32: EtherCAT Demo: Select device

The device can be configured under: *Geräte -> EtherCAT -> double-click on IHSV_EC*. The following settings must be made in the General tab:

- Additionally activate exert mode

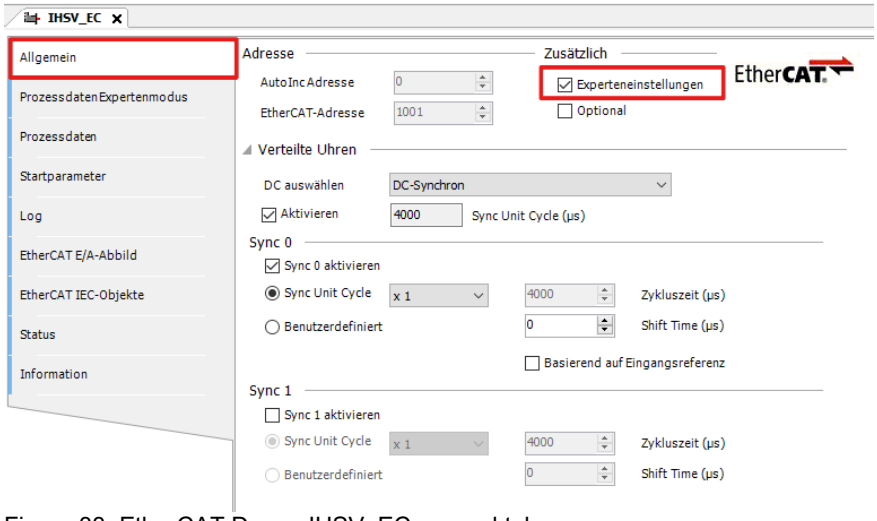


Figure 33: EtherCAT Demo: IHSV_EC general tab

The axis must then be added to the IHSV-EC device. This is done under: *Geräte -> EtherCAT -> right-click on IHSV_EC -> add SoftMotion CiA402 axis.*

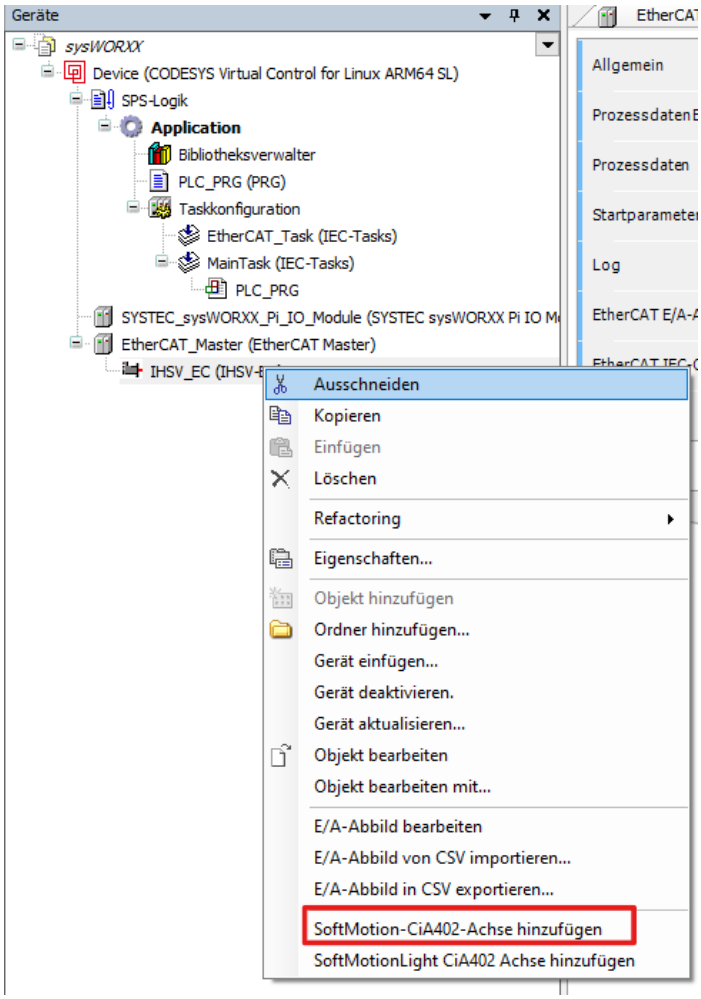


Figure 34: EtherCAT Demo: IHSV_EC SoftMotion-CiA02 Axis

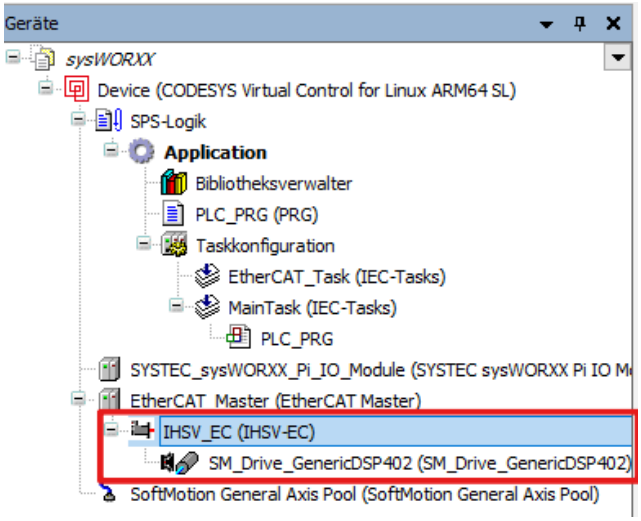


Figure 35: EtherCAT Demo: IHSV_EC with SoftMotion-CiA02 Axis

To integrate a corresponding program for EtherCAT, the following must be entered under: *Geräte -> Device -> right-click on Application -> select Objekt hinzufügen -> add POU.* A name and an implementation language must be assigned. Once the object has been created, a demo code can be implemented.

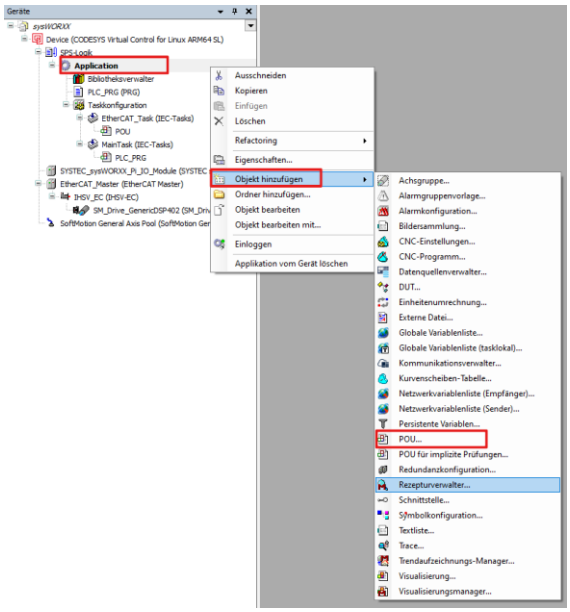


Figure 36: EtherCAT Demo: Application POU

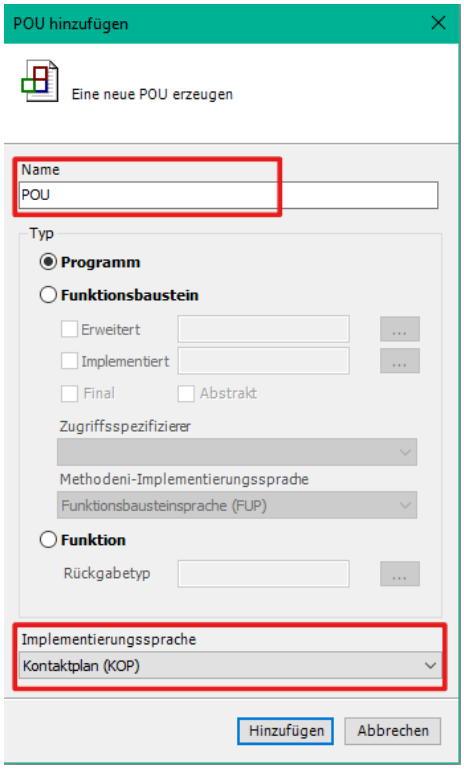


Figure 37: EtherCAT Demo: Add POU

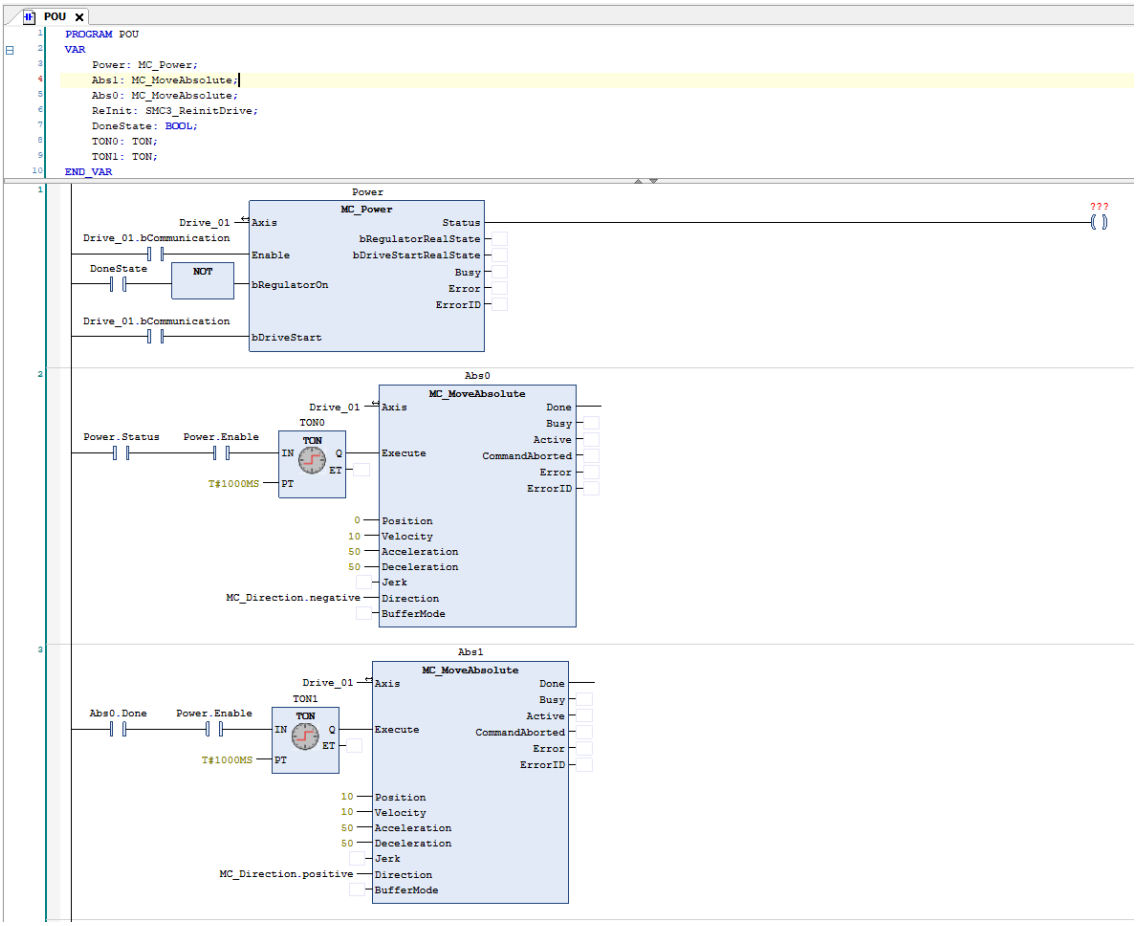


Figure 38: EtherCAT Demo: Demo Code (1)

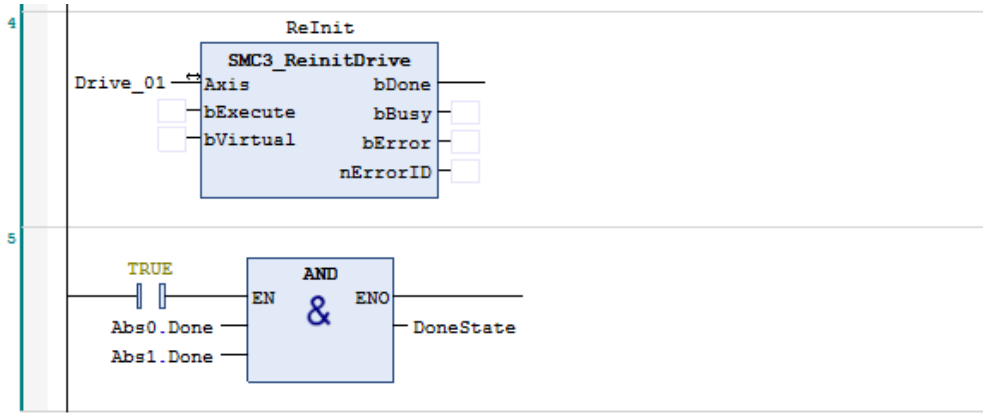


Figure 39: EtherCAT Demo: Demo Code (2)

In this program, the motor is turned to the left for a certain time, then stopped with a short delay and then turned to the right. The written program must be linked to the EtherCAT task. This is done under: *Geräte -> Application ->Taskkonfiguration -> right-click on EtherCAT_Task -> select Object hinzufügen -> add Programmaufruf*

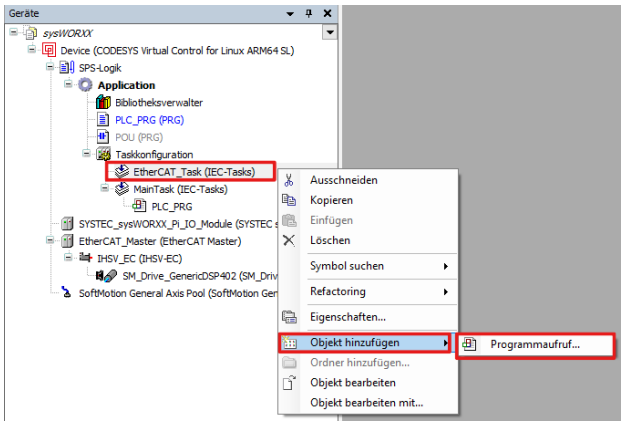


Figure 40: EtherCAT Demo: Linked POU to EtherCAT_Task

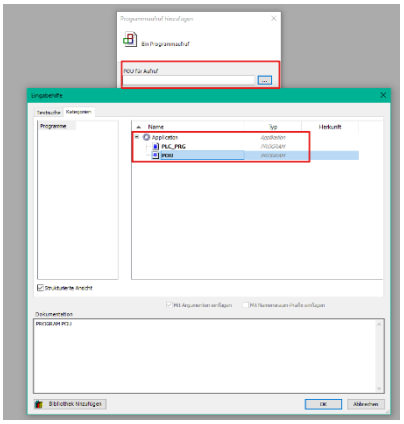


Figure 41: EtherCAT Demo: Select POU from Application

To run the code on the device, the code must be generated, logged in and the program started.

Figure 42: EtherCAT Demo: Login and start

4.2 Modbus

The Modbus communication protocol is based on a client/server architecture. CODESYS offers extensive and integrated support

4.3 CANopen

The asynchronous serial bus system CAN (Controller Area Network) can be used for the most different applications and plays an important role in the automation market. CODESYS offers comprehensive CAN support with integrated functions and products for different layers.

Once the project is deployed it needs an upgrade with the following packages one can download from *Tools->CODESYS installer*. Before a new package can be installed, it must be restarted as administrator. The following package must then be installed:

- CODESYS CANopen
- CAN Bus API

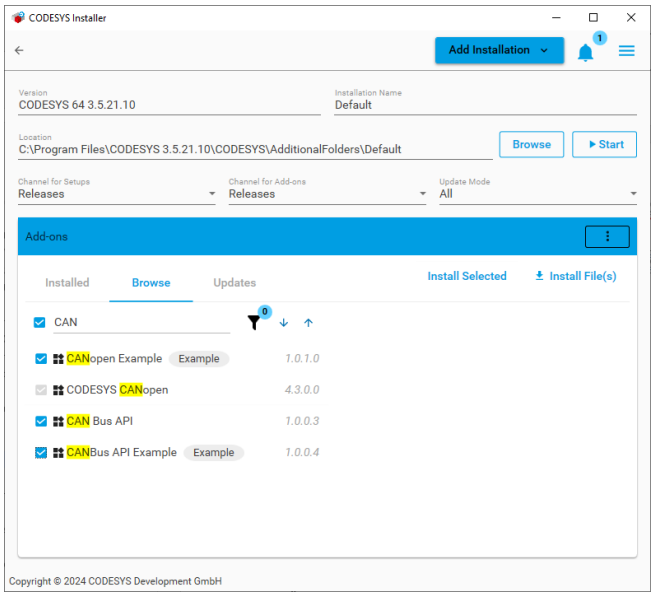


Figure 43: CANopen Demo: CODESYS installer

CODESYS must be terminated during installation and can be restarted after installation using the *CODESYS installer*.

If not already done, one has to follow the steps described in section 3.2.1. Also following parameters must be set instead of the ones described in section 3.2.2.

The following parameters must be changed in the Docker instance:

- `Mounts: /var/opt/codesysvcontrol/instances/CODESYS_CONTAINER/conf/codesyscontrol:/conf/codesyscontrol/,/var/opt/codesysvcontrol/instances/CODESYS_CONTAINER/data/codesyscontrol:/data/codesyscontrol/,/var/run/codesysextension/extfuncs/:/var/run/codesysextension/extfuncs/`
- `Network: host`

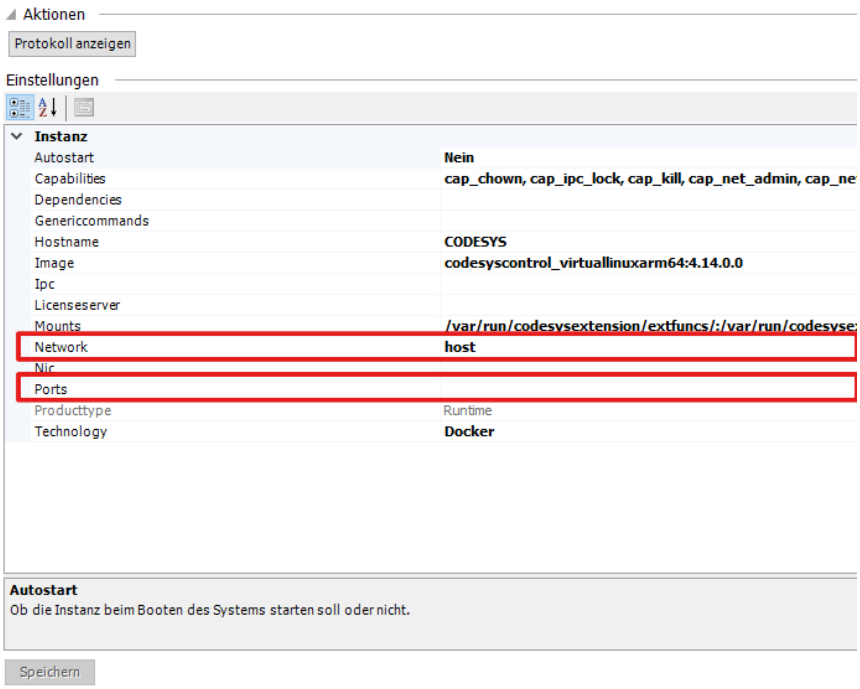


Figure 44: CANopen Demo: Control SL roll out Docker script

Once the Docker has been successfully configured, the Docker instance can be started. The CANopen fieldbus can be attached and selected under: *Geräte -> right-click on Device -> Geräte anhängen*. CANopen should be added twice.

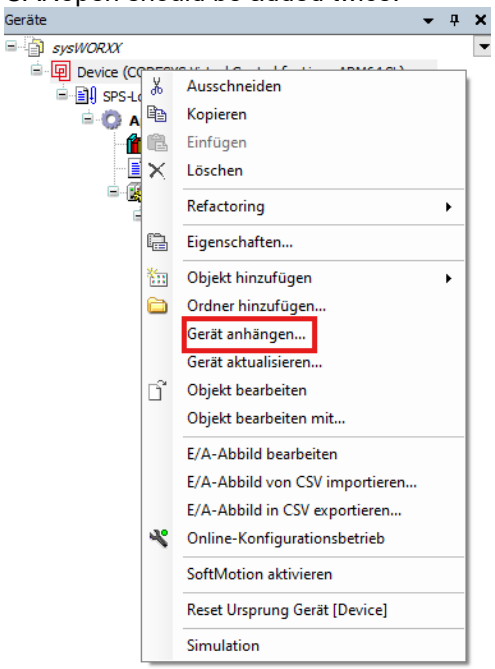


Figure 45: CANopen Demo: Attach device

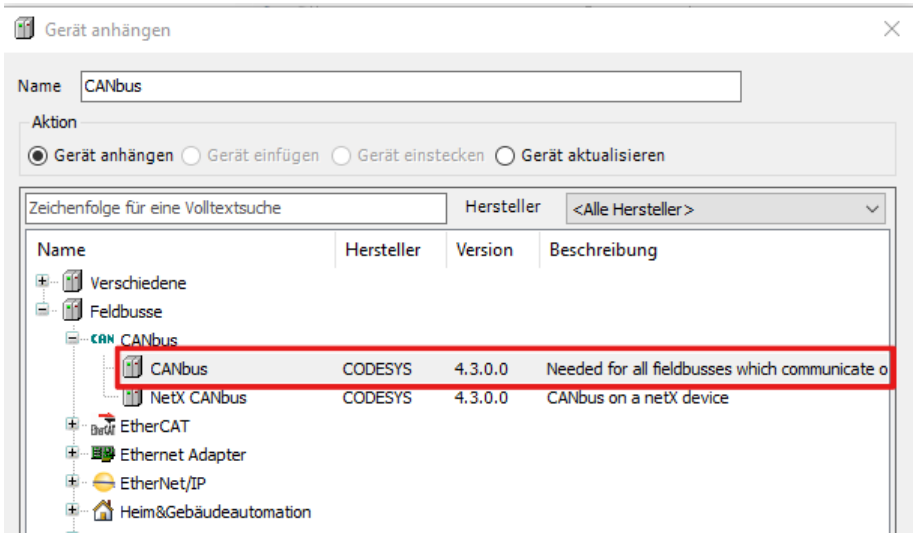


Figure 46: CANopen Demo: Select CANbus

The device can be configured under: *Geräte -> double-click on CANbus_1*. The following settings must be made in the General tab:

- Set Network

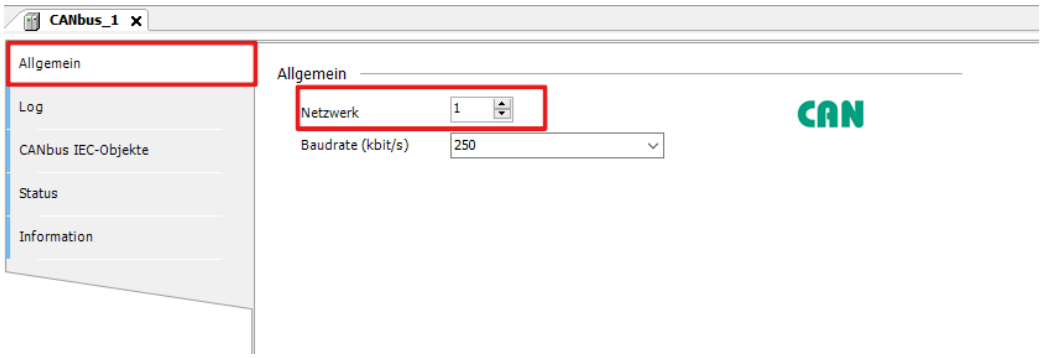


Figure 47: CANopen Demo: CANbus_1 general tab

After this, the device can be attached (*Geräte-> right-click on CANBus_0 ->Gerät anhängen).* CANopen_Manager must be selected for CANBus_0 and CANopen_Device for CANBus_1.

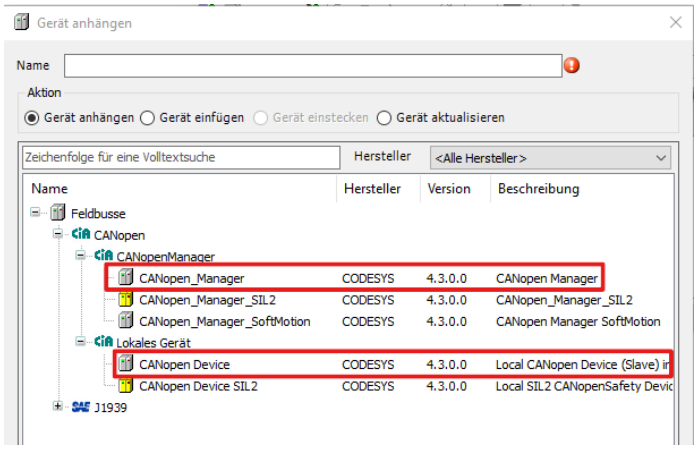


Figure 48: CANopen Demo: Select device

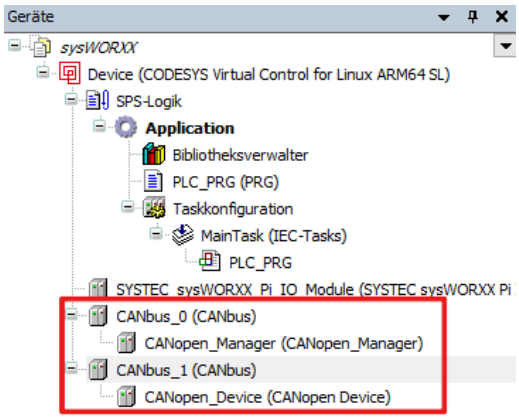


Figure 49: CANopen Demo: Overview of the devices

The device can be configured under: *Geräte -> CANbus_0 -> double-click on CANopen_Manager*. The following settings must be made in the General tab:

- Activate NMT Start All
- Set Node-ID
- Configure Heartbeat

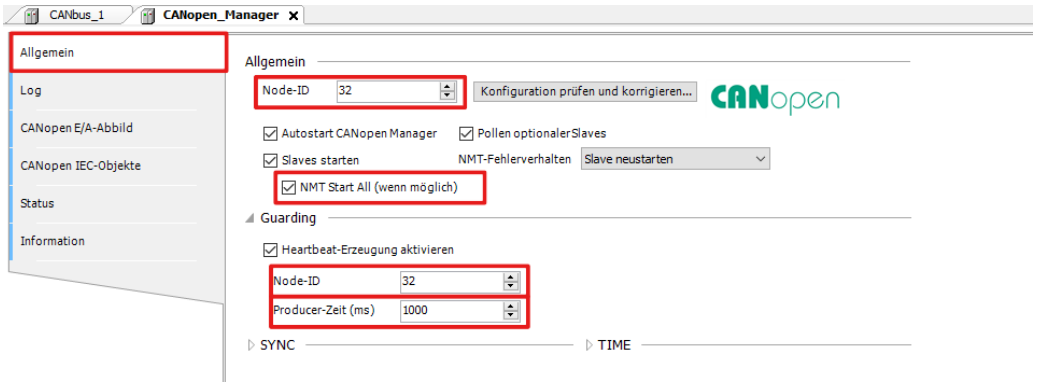


Figure 50: CANopen Demo: CANopen_Manager general tab

The other device can be configured under: *Geräte -> CANbus_1 -> double-click on CANopen_Device*. The following settings must be made in the General tab:

- Set Node-ID and device profile
- Configure ESD-File

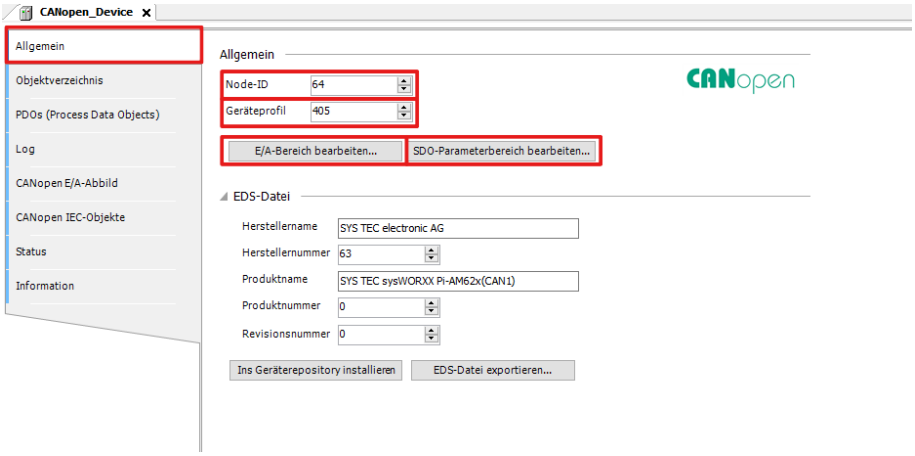


Figure 51: CANopen Demo: CANopen_Device general tab

A corresponding EDS file for the configuration of the device can be inserted in the Object directory tab. The I/O areas and the SDO parameter areas can then be edited.

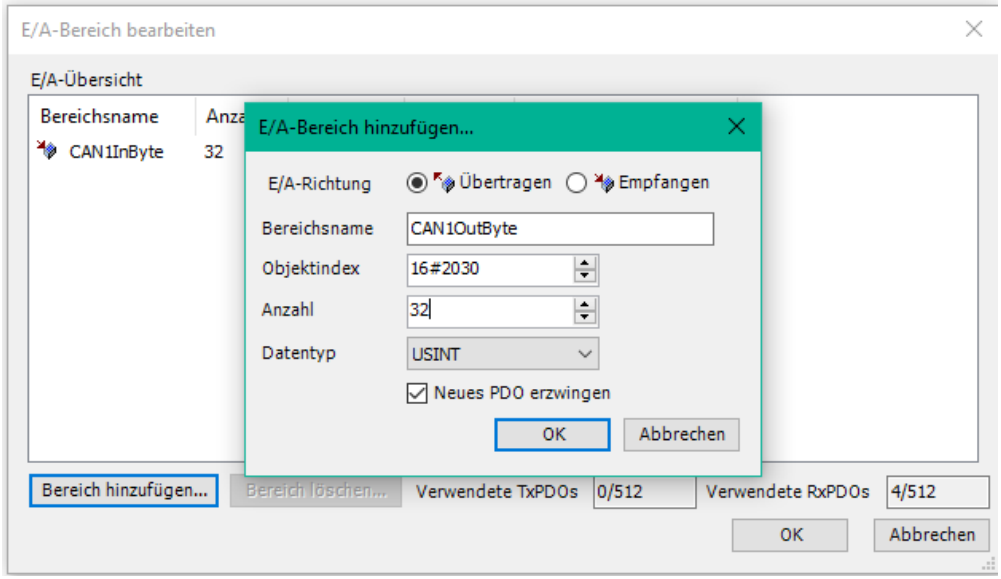


Figure 52: CANopen Demo: CANopen_Device add I/O areas (out)

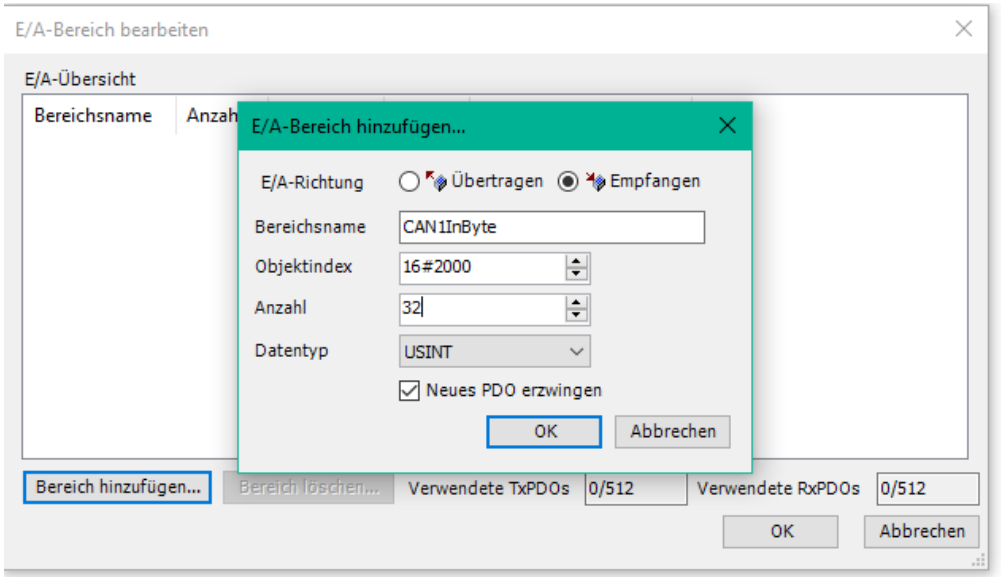


Figure 53: CANopen Demo: CANopen_Device add I/O areas (in)

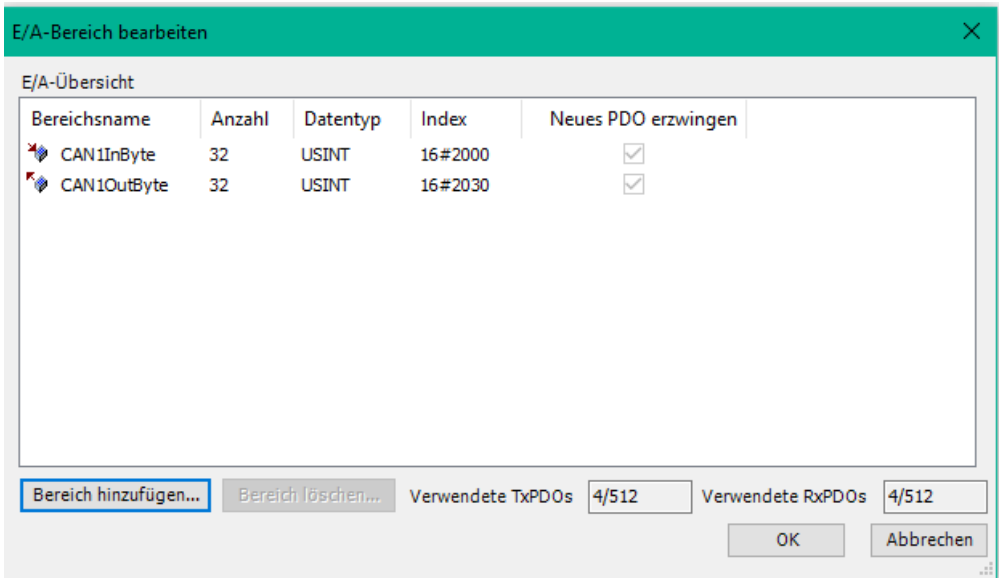


Figure 54: CANopen Demo: CANopen_Device I/O areas

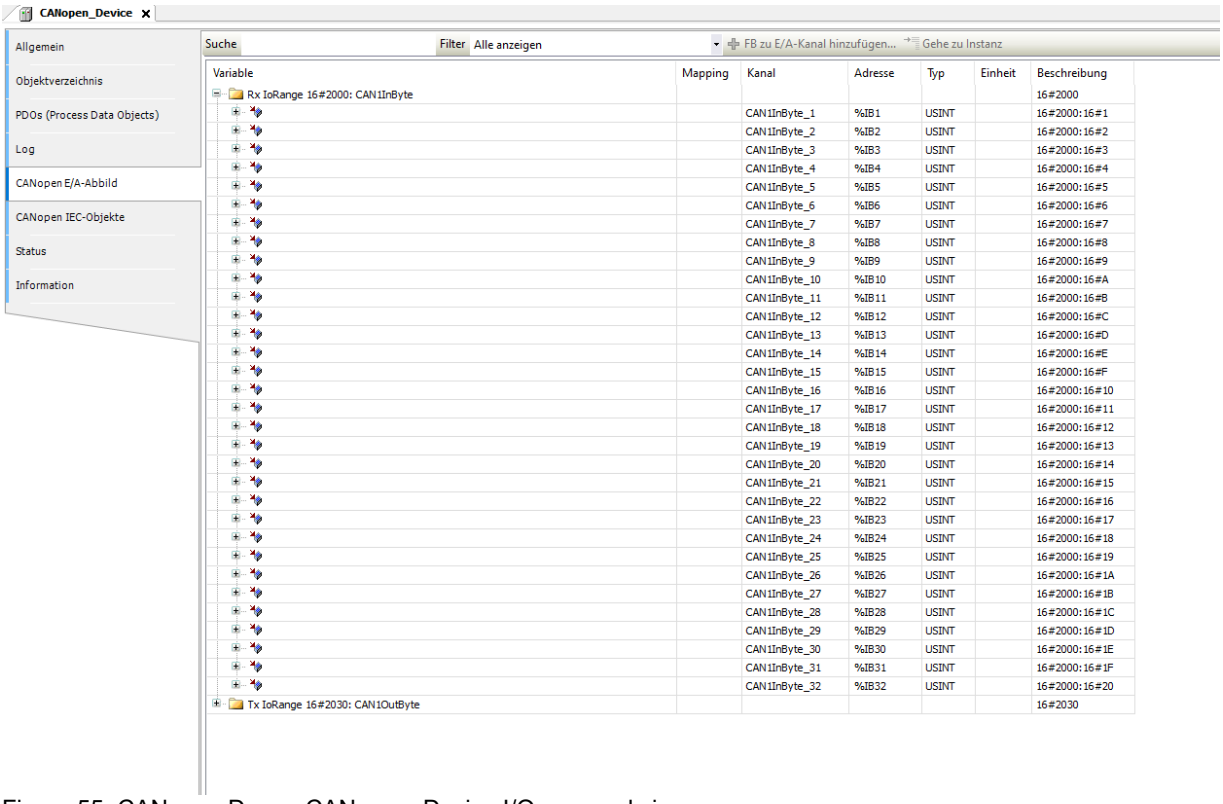


Figure 55: CANopen Demo: CANopen_Device I/Os general view

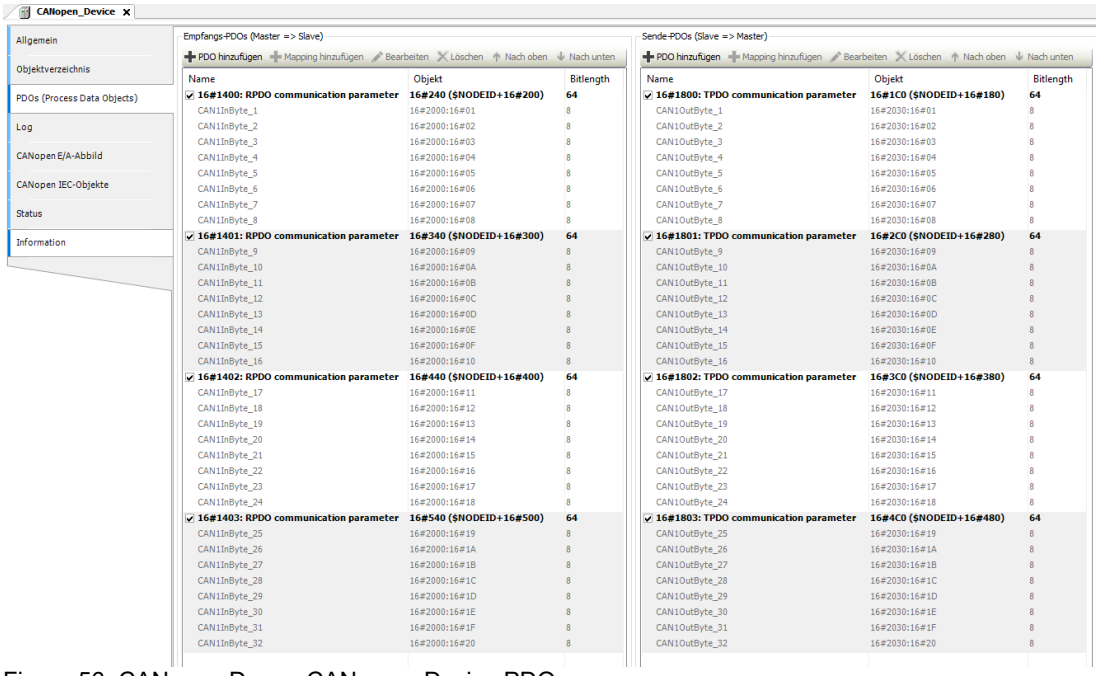


Figure 56: CANopen Demo: CANopen_Device PDOs

To integrate a corresponding program for CANopen, the following must be entered under: *Geräte -> Device -> right-click on Application -> select Objekt hinzufügen -> add POU.* A name and an implementation language must be assigned. Once the object has been created, a demo code can be implemented.

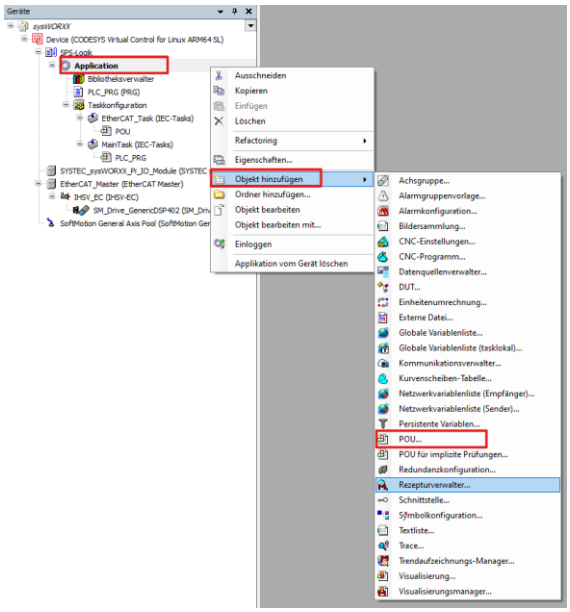


Figure 57: CANopen Demo: Application POU

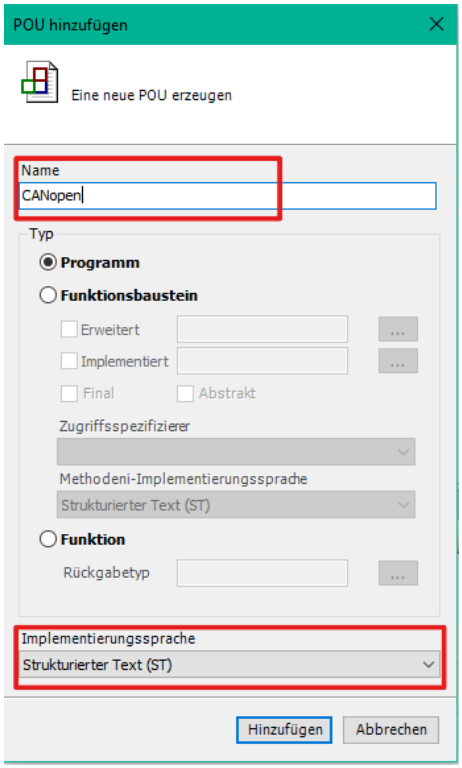


Figure 58: CANopen Demo: Add POU

```
CANopen
PROGRAM CANopen
VAR
    usiNetId        :   USINT;
    uiBaudrate      :   UINT;
    xSupports29Bit  :   BOOL;
    ctMessages      :   CAA.COUNT;
    hDriver         :   CAA.HANDLE;
    hMessage        :   CAA.HANDLE;
    CobID           :   CL2I.COBID;
    pData           :   POINTER TO CL2I.DATA;
    eError          :   CL2.ERROR;
END_VAR


usiNetId        := 0;
uiBaudrate      := 250;
xSupports29Bit  := FALSE;
ctMessages      := 10;

hDriver := CL2.DriverOpenH (usiNetId, uiBaudrate, xSupports29Bit, ctMessages, peError := ADR(eError));
IF( hDriver <> CAA.gc_hINVALID)
THEN

    CobID := 0;

    hMessage := CL2.CreateMessage (hDriver, CobID, 2, FALSE, FALSE, peError := ADR(eError));

    IF (hMessage <> CAA.gc_hINVALID)
    THEN
        //Get message data pointer
        pData := CL2.GetMessageDataPointer(hMessage := hMessage, peError := ADR(eError));
        IF pData <> CAA.gc_pNULL THEN
            //initialize message data
            pData^[0] := 16#81;
            pData^[1] := 16#0;
        END_IF

        //send message
        eError := CL2.Write(hDriver          := hDriver,
                            hMessage         := hMessage,
                            usiPriority      := 1, //highest priority
                            xEnableSyncWindow := FALSE);

        IF (eError <> CL2.ERROR.NO_ERROR)
        THEN
            //sending was not successful ==> release the message
            CL2.FreeMessage(hMessage);
        END_IF

    END_IF

    CL2.DriverClose (hDriver);
END_IF
```

Figure 59: CANopen Demo: Demo Code

The written program must be linked to the CANopen task. This is done under: *Geräte -> Application ->Taskkonfiguration -> right-click on CANopen_Task -> select Object hinzufügen -> add Programmaufruf*

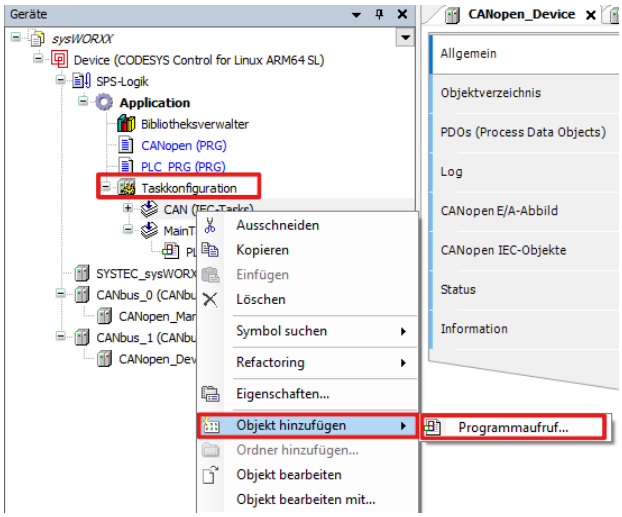


Figure 60: CANopen Demo: Linked POU to CANopen_Task

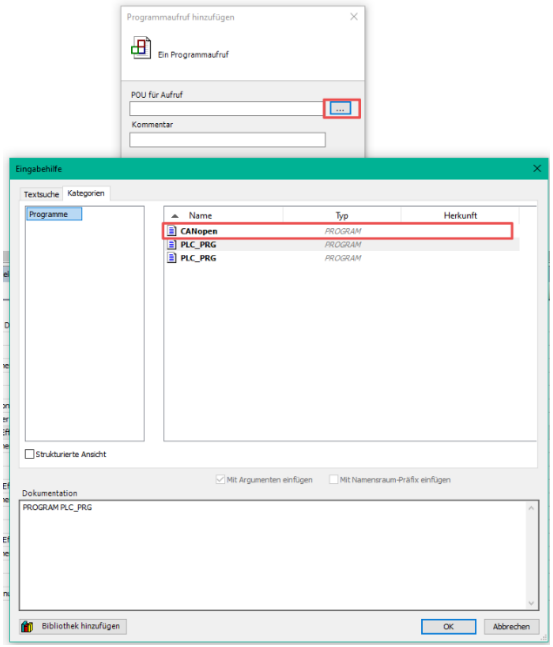


Figure 61: CANopen Demo: Select POU from Application

The following libraries must be installed for this example.

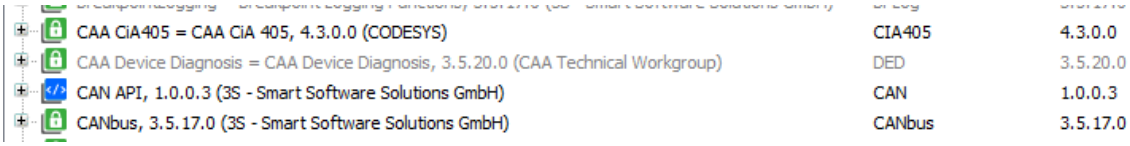


Figure 62: CANopen Demo: Library administrator

To run the code on the device, the code must be generated, logged in and the program started.

Figure 63: CANopen Demo: Login and start

5 CODESYS visualization support

To be able to start any visualization project, one must set up the project just as described in section 3.2.

5.1 Add Visualization to any project

Adding a visualization can be done via the project view: r*ight-click on Application -> Objekt hinzufügen -> Visualisierung.*

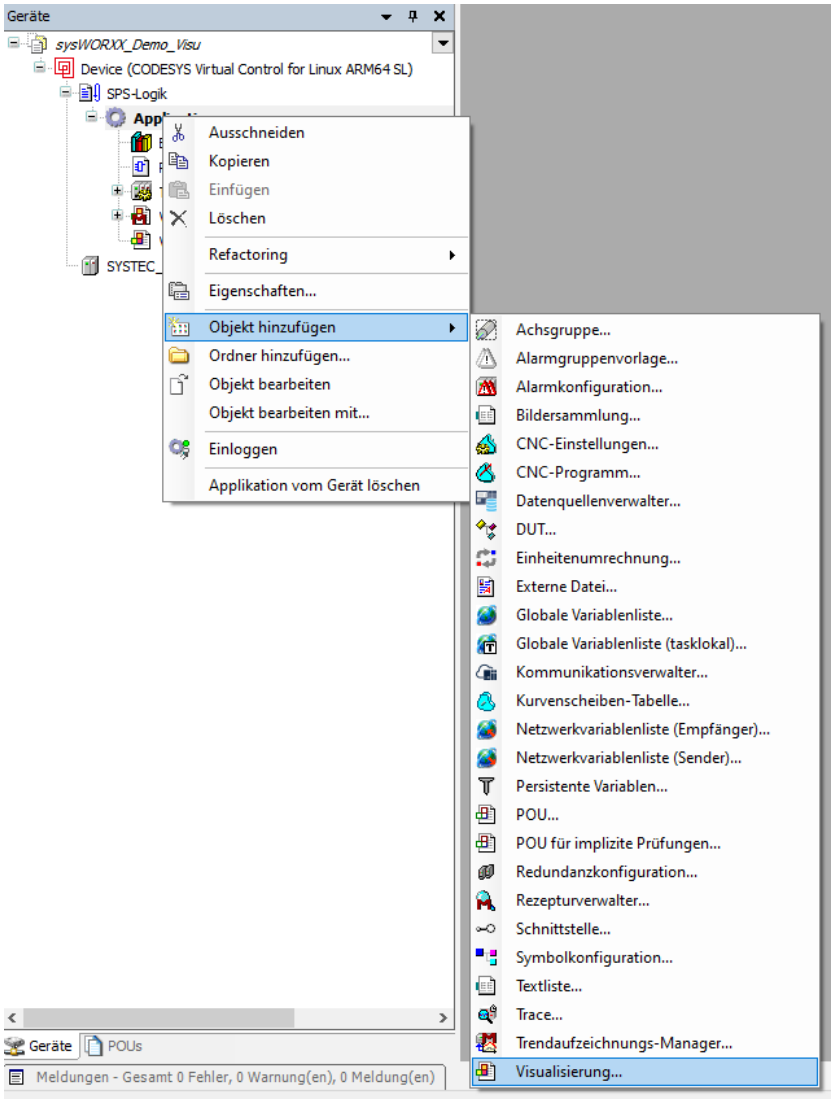


Figure 64: Add visualization to project

A dialog will open, which lets one name the visualization tab. After doing that, the empty visualization tab will be opened.

5.2 Add Elements to the visualization

If one opens the visualization tab, there should be a toolbox opening on the right side of the window:

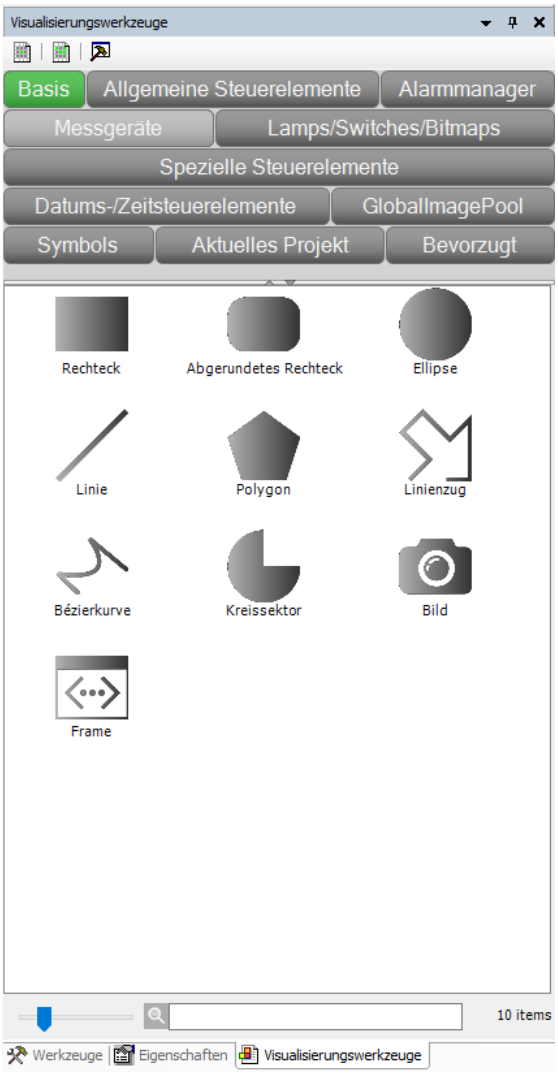


Figure 65: Visualization toolbox

There, one can choose and design the empty visualization tab. CODESYS offers a few simple animated buttons and sliders. These can be put into the visualization with simple drag-and-drop. To move any added elements, one must click them. This way, they can also be resized. To move a group of elements, one can mark multiple at once.

5.3 Add a picture and load it on the device

One can add pictures or images with the standard functionality. For that, go into the toolbox, select the tab “Basis” and drag-and-drop the element “Bild” into the visualization window.

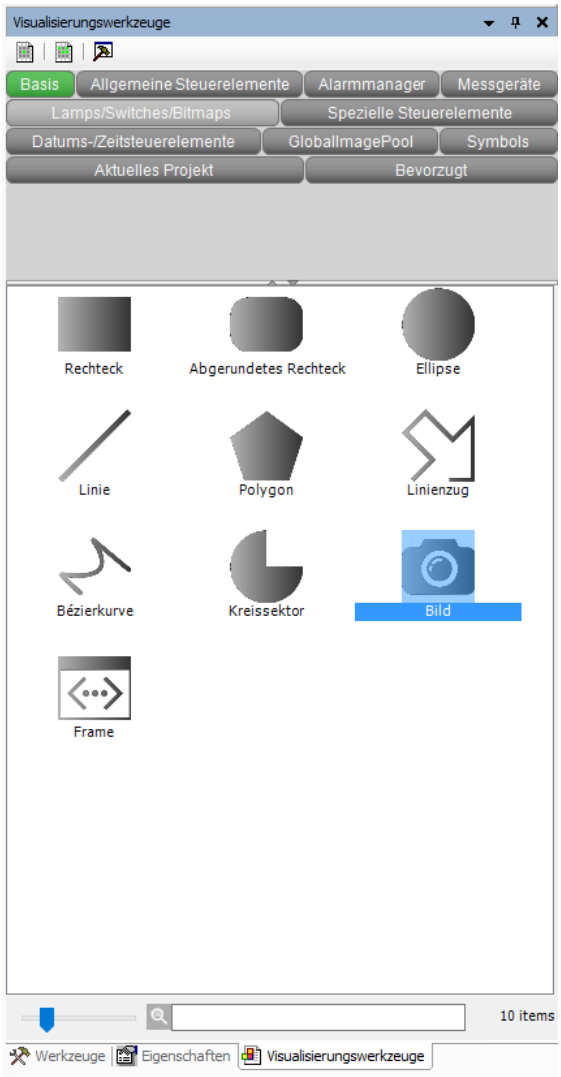


Figure 66: Picture in toolbox

A dialog window will open, where any already added image or picture is shown, as well as any default image provided by CODESYS.

To add a custom image, close this tab and select the empty frame inside the visualization tab. On the right side, the configuration window will open, see 5.4.

In the field “Statische ID” one must set any name for the element and press enter. Doing that will open the windows dialog to add any custom file.

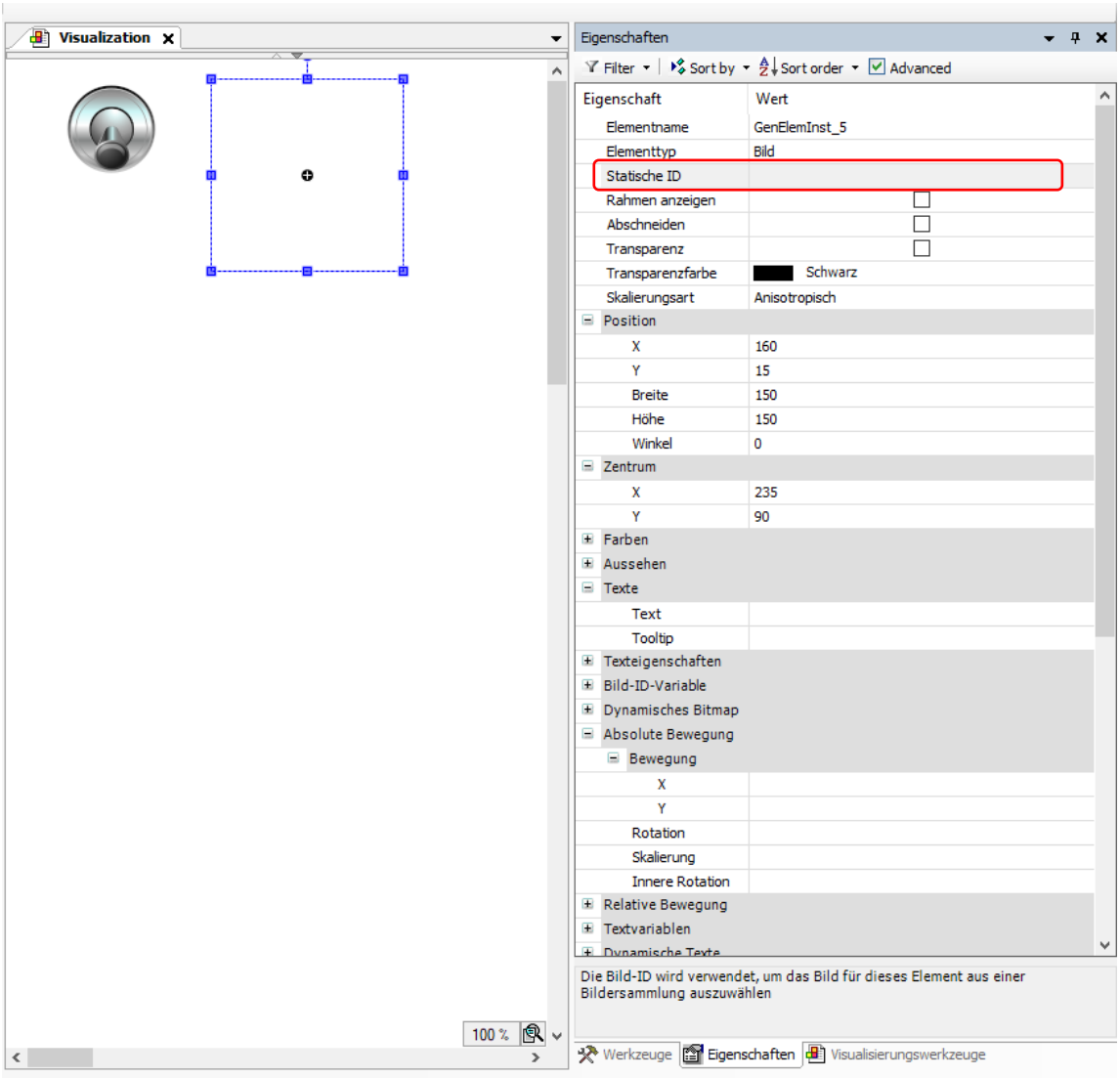


Figure 67: Add static ID

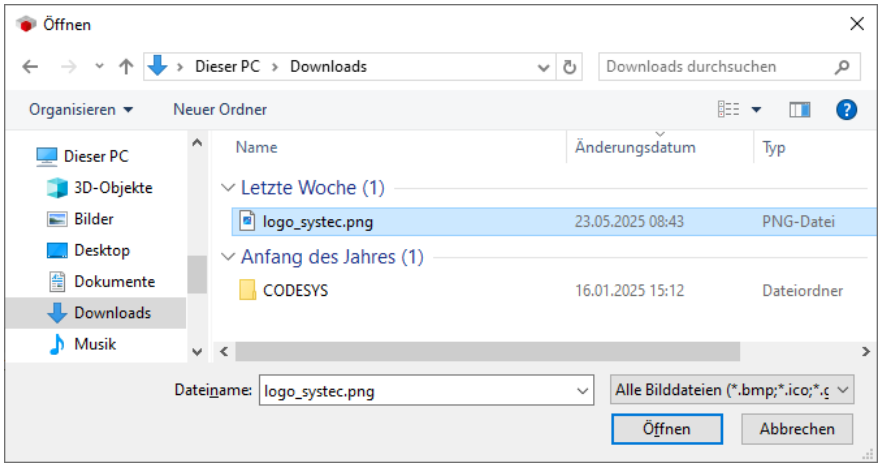


Figure 68: Add custom image

Selecting the file will add it to the project and to the visualization.

By default, this image is not visible with the WebVisu functionality. To enable the transfer to the device, do the following*: Select POUs -> Click GlobalImagePool -> Click on the field Verknüpfungstyp*. In the opening configuration window, select “Ins Projekt einbetten”

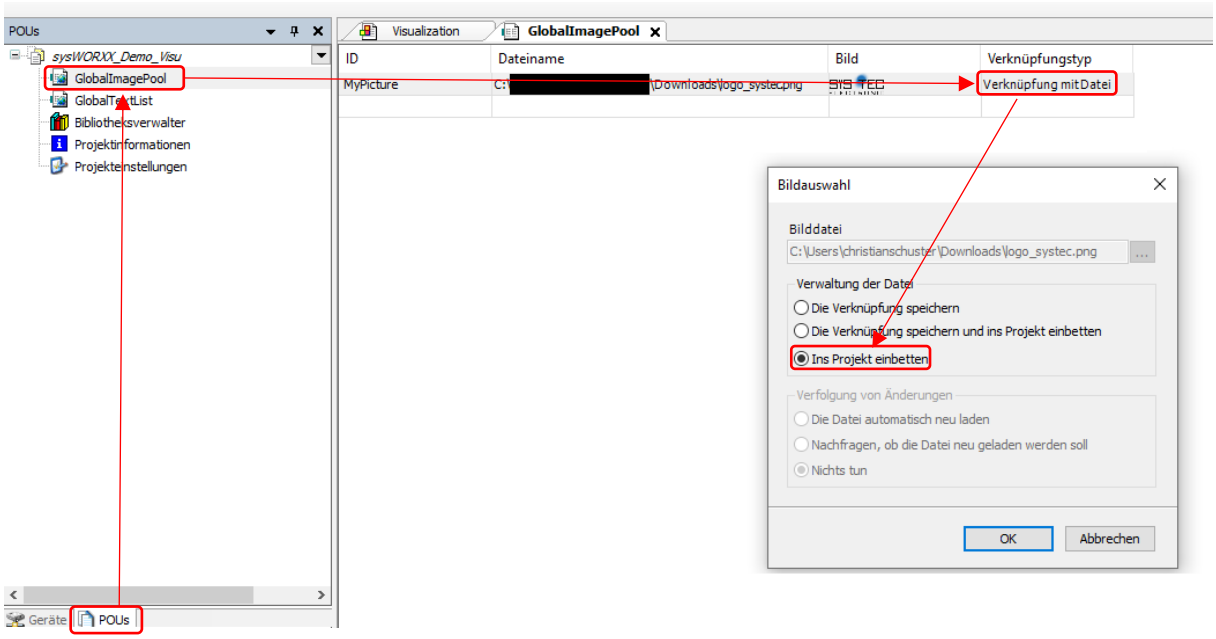


Figure 69: Embed image into project

Now the picture will be loaded onto the target device and is visible in the WebVisu.

5.4 Configure control elements

Besides visualizing any information, one can control the program and its behavior through control elements (e.g. switches, lamps, sliders, …). To add any functionality, click an added element in the visualization tab. A configuration tab should open, where the toolbox was (right side of the screen):

If not open, open the tab “Position” and there in the field “Variable” add any variable one wants to change.

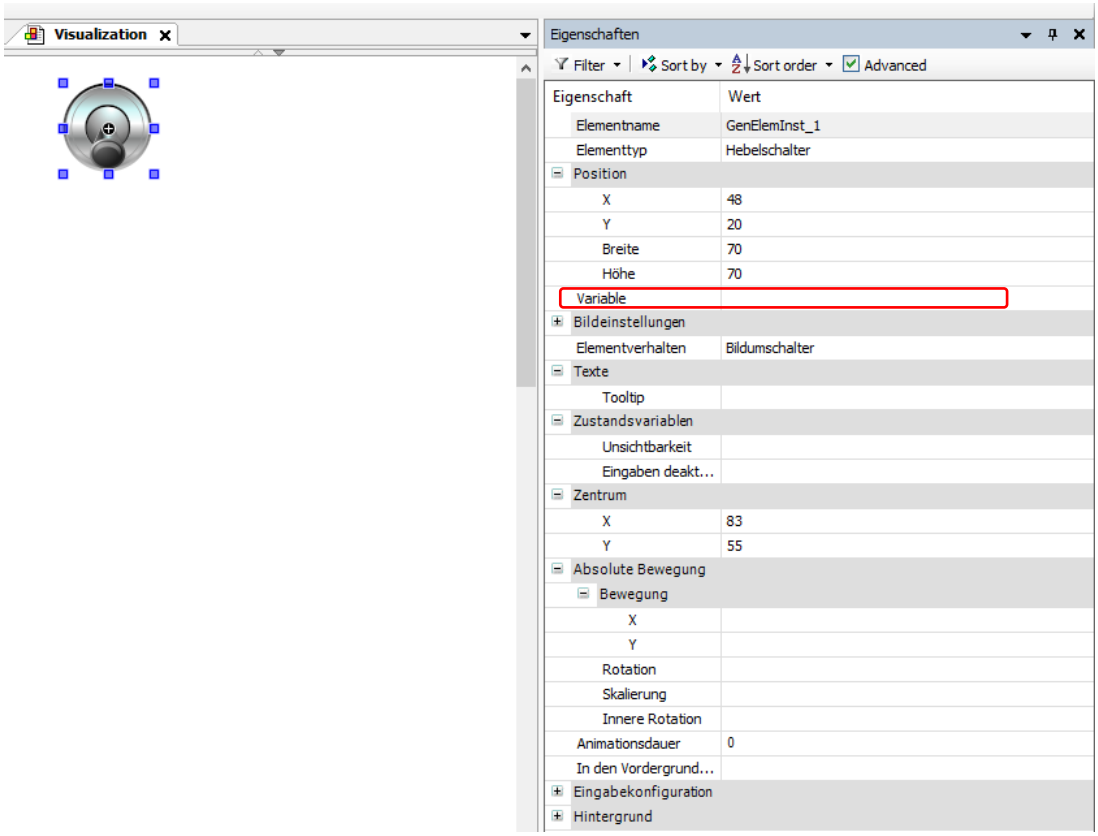


Figure 70: Add variable to element

5.5 Example application

Putting everything mentioned together, one could build the following visualization:

Figure 71: Visualization example

There is an added image, some textboxes, switches, and lamps. Besides the SYS TEC logo, all the elements are provided by default from CODESYS. In this example, the switches control digital outputs, and the lamps are display the digital input states.

There is no code needed for this application, but for each switch and lamp, there has to be a variable defined.

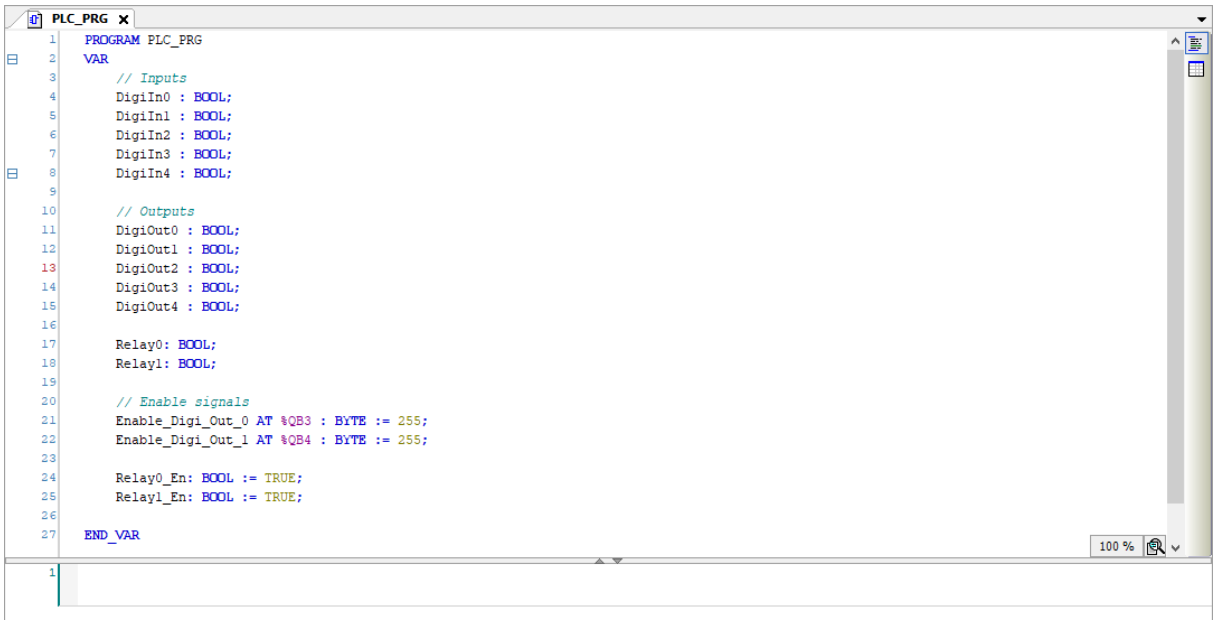


Figure 72: Defined variables

For the digital outputs, there are the variables DigiOut0…3, as well as Relay0…1 and their respective enable signals (must be set TRUE/1 to let the CODESYS connector on the device set the outputs).

These must be mapped to the physical hardware:

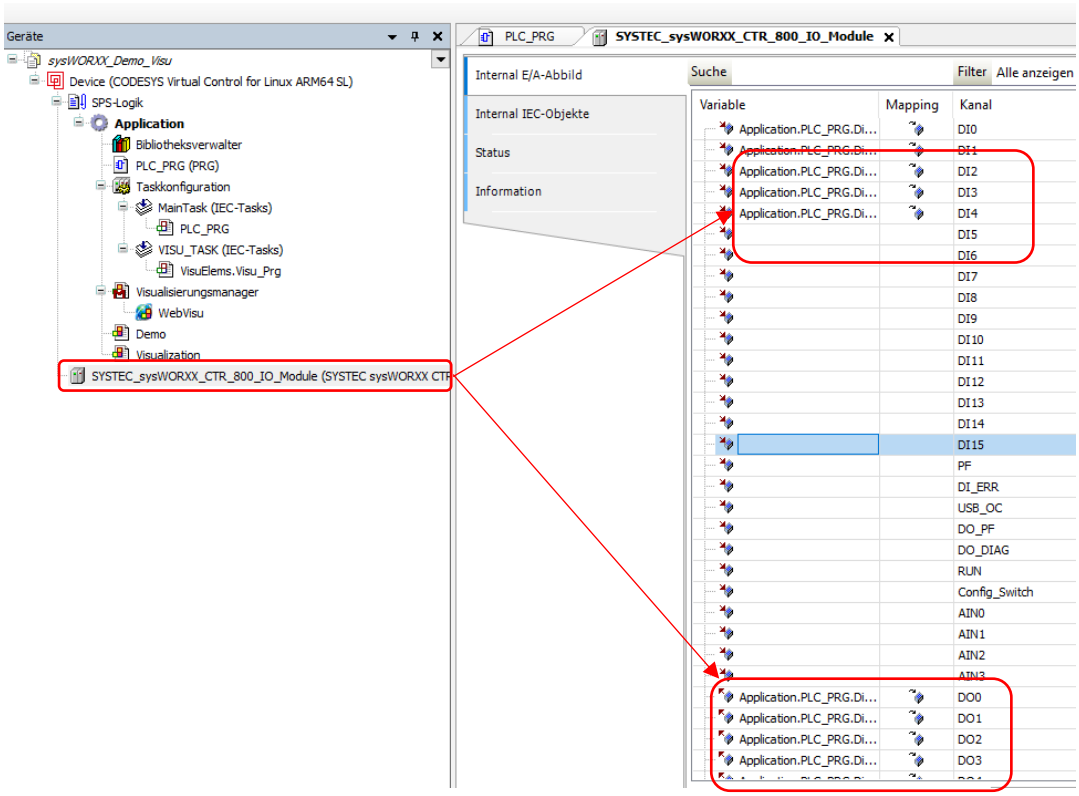


Figure 73: Set variables to hardware

In the last step, each switch and lamp has to be mapped with the corresponding variable:

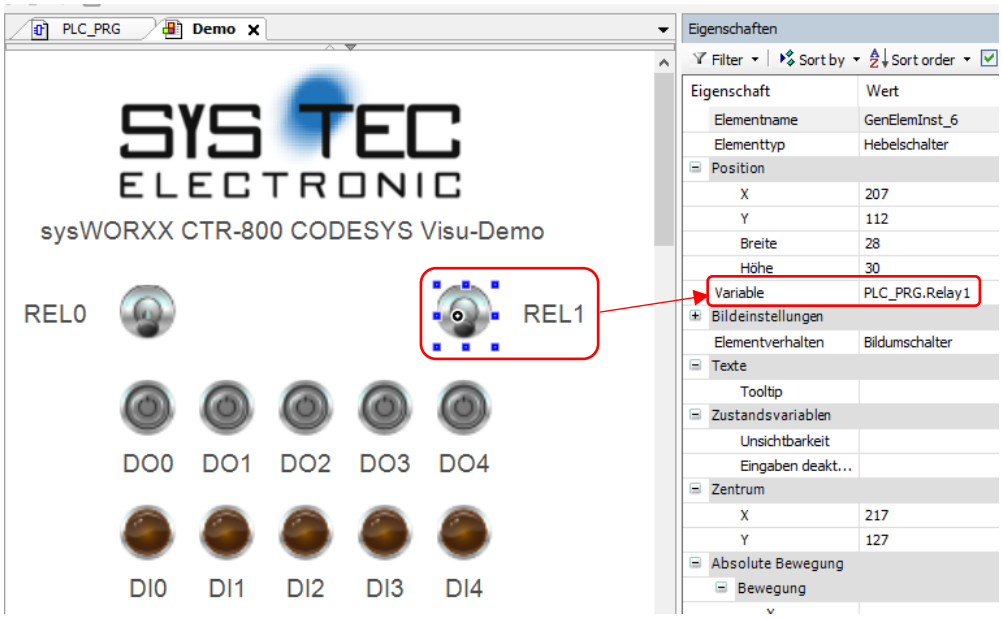


Figure 74: Add variable to visualization elements

Compiling and downloading the application to the device has to be done as usual.

5.6 Web visualization

Additionally, CODESYS will set up a so called WebVisu. The previous created visualization can now be accessed via a Browser (Firefox, Chrome, …). For this access, one has to be in the same network and use the IP address from the device with the port “8080”, e.g. 192.168.15.159:8080:

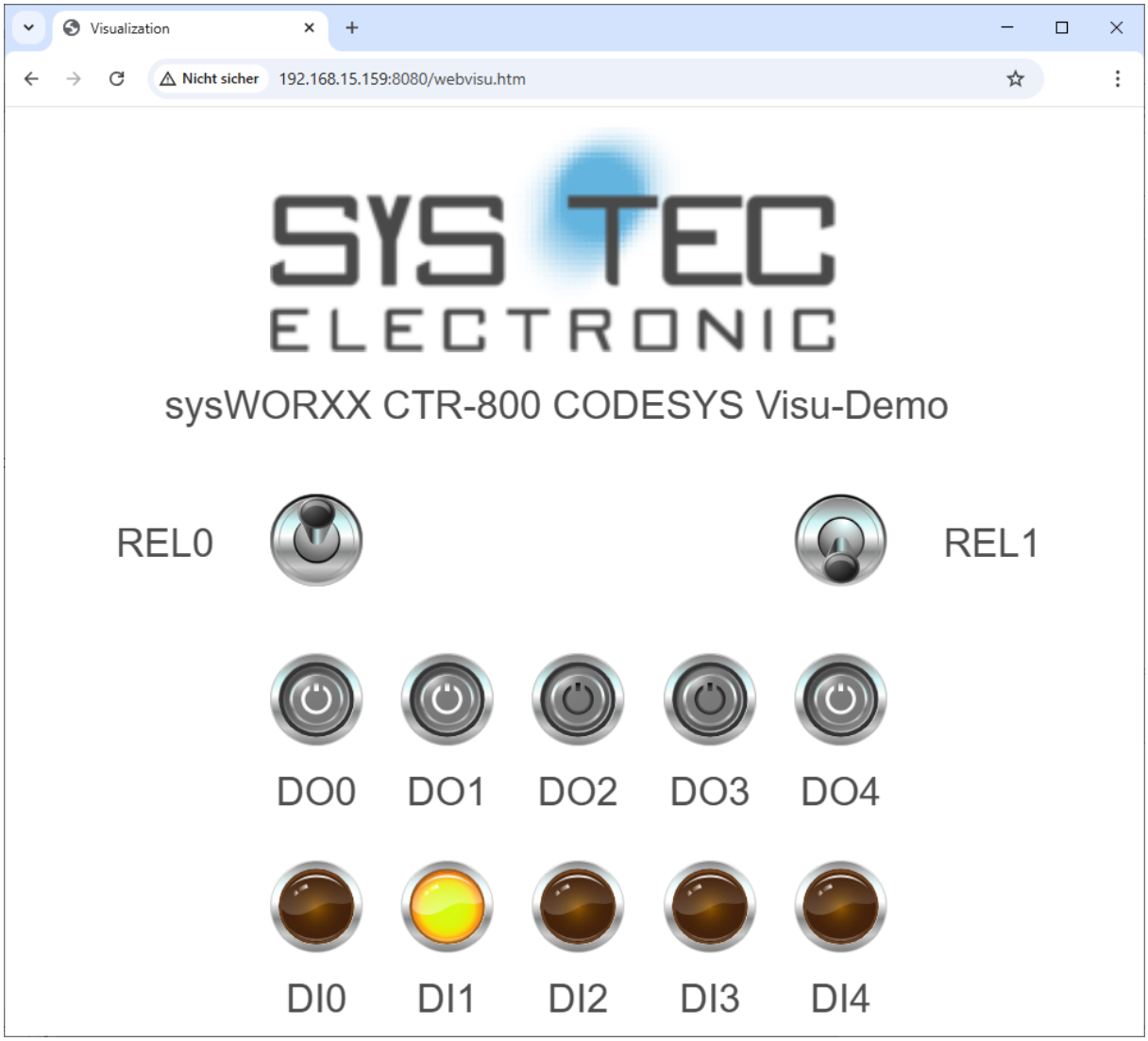


Figure 75: WebVisu example

Hint: If one is working from inside a docker container (see section 3.2.4) the port 8080 must be set!