Internal PLC (CODESYS)

User Manual

Subject to change without notice

The information contained in this document is provided for informational purposes only. While efforts were made to verify the accuracy of the information contained in this documentation, it is provided 'as is' without warranty of any kind.

Third-party brands and names are the property of their respective owners.

Microsoft®, Win32, Windows®, Windows XP, Windows Vista, Windows 7, Windows 8, Visual Studio are either registered trademarks or trademarks of the Microsoft Corporation in the United States and other countries. Other products and company names mentioned herein may be the trademarks of their respective owners.

The example companies, organizations, products, domain names, e-mail addresses, logo, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred.

Contents

Internal PLC (CODESYS V3)

This manual describes:

- How to program internal CODESYS V3 PLC available in Exor HMIs
- How to configure a CODESYS V3 project to get communication with Local/Remote I/O and fieldbusses

Note: For detailed information regarding the use of CODESYS V3 communication protocol in Internal PLC (CODESYS) related to the communication with CODESYS V3 PLC Runtime please refer to the specific CODESYS V3 communication protocol chapter of Internal PLC (CODESYS) help.

Note: This manual is not intended as CODESYS V3 programming manual, for detailed documentation regarding CODESYS V3 Development System please refer to the CODESYS web site www.codesys.com and to and to its on-line help.

Control Solutions with HMI and CODESYS V3

Exor products can deliver effective HMI and control solutions based on Internal PLC (CODESYS) and CODESYS V3.

HMI products shown in table below have been designed to include the CODESYS V3 PLC Runtime.

HMI series	OS	Device Name	Optional Plugins	Communication Stacks					
				CANopen Master	Modbus TCP/RTU Master/Slave	EtherNet/IP Scanner	Profinet I/O	EtherCAT	PowerLink
eTOP500, eTOP500G, eTOP600	WCE	500/600 (WCE)	Y	Y *1	Y	Y	Y	N/A	Y
eTOP600L	Linux	600L/700 (Linux)	Y	Y *2	Y	Y	Y	Y	Y
eX700 / eXware	Linux	600L/700 (Linux)	Y	Y *3	Y	Y	Y	Y	Y
eSMART	Linux	eSMART (Linux)	N/A	N/A	Y	Y	Y	N/A	Y

*1 = 600 MHz CPU HMIs (UN31) support only 1 CAN interface. 1 GHz CPU HMIs (UN30) support up to 2 CAN interfaces.

*2 = Support up to 2 CAN interfaces.

*3 = eX705 and eXware support only 1 CAN interface. All other HMIs support up to 2 CAN interfaces.

CODESYS V3 Runtime is automatically transferred to the HMI by Internal PLC (CODESYS) as part of HMI Runtime and it is running with the support of the HMI operating system.

CODESYS V3 Development System is required to develop and debug PLC applications and transfer them to Exor HMI.

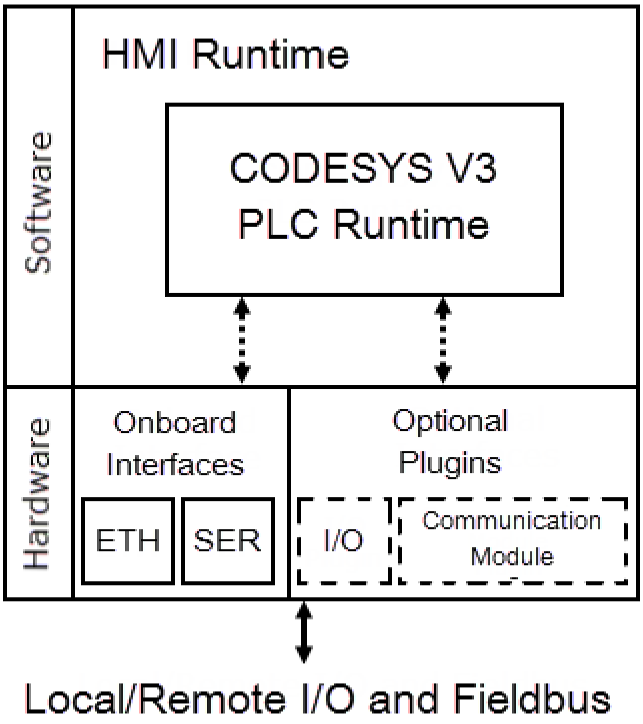
CODESYS V3 Runtime requires a license activation to be used. CODESYS V3 license is activated by Internal PLC (CODESYS), the license code is uniquely matched with the HMI MAC-ID.

Note: License activation is required for the operation of CODESYS V3 Runtime.
CODESYS V3 Runtime will communicate with CODESYS V3 Development System only after license activation.

System Configuration

HMI and PLC based on Internal PLC (CODESYS) and CODESYS V3 can be applied in different configurations.

The picture below shows how HMI and PLC can be configured.



Note: Refer to table above to know which optional plugins and fieldbusses are available for your hardware configuration

Connectivity in HMI and Control Systems

Even when adding the control option with the CODESYS V3 PLC, the HMI still retain its full communication capabilities based on communication drivers configurable from Internal PLC (CODESYS).

Note: Communication capabilities over Serial network are limited by the amount of serial interfaces available, each serial interface supports a single communication protocol. Specific optional add-on serial interface modules are available to increment the amount of serial interfaces at disposal.

Requirements and Limitations

The following firmware and software versions are required to work with the CODESYS V3 PLC Runtime:

Element	WCE	Linux
Internal PLC (CODESYS)	1.91 SP1 or higher	2.6 or higher
CODESYS V3 Development System	3.5.7 patch 5	

Note: The HMI internal PLC is compatible with CODESYS V3.5 Development System only.
CODESYS V2.3 Development System is NOT compatible with CODESYS V3 Runtime.

Note: It is strongly recommended to install CODESYS V3 Development System version 3.5.7 patch 5 for programming Exor HMIs.
If a newer version of CODESYS V3 Development System is already installed, make sure to use 3.5.7 versions of libraries and fieldbus to program Exor HMIs.

Timer resolution

The resolution of CODESYS V3 timers is 1 millisecond. When a timer value is defined it is internally translated to the corresponding number of milliseconds.

The resolution of the internal Real Time Clock is 1 millisecond allowing the maximum resolution of timers. Note that the execution time of the PLC program may apparently affect the resolution of timers.

Getting Started

This chapter provides the necessary information on how to set-up the HMI + CODESYS PLC system.

The required operations are listed below and explained in the forthcoming chapters.

1. CODESYS V3 Development System installation
2. Exor CODESYS Package installation
3. HMI Runtime installation with Internal PLC (CODESYS)
4. Activation of the CODESYS V3 license on the HMI
5. Creation of a new PLC project
6. Download PLC application
7. Symbol File configuration with CODESYS V3 Development System
8. Communication Setup in Internal PLC (CODESYS)

CODESYS V3 Development System installation

CODESYS V3 Development System 3.5.7 patch 5 is to be used to program Exor HMIs internal PLC.

The software can be downloaded from EXOR websites under *Support -> JMobile -> Documents and Download* section.

Exor CODESYS Package installation

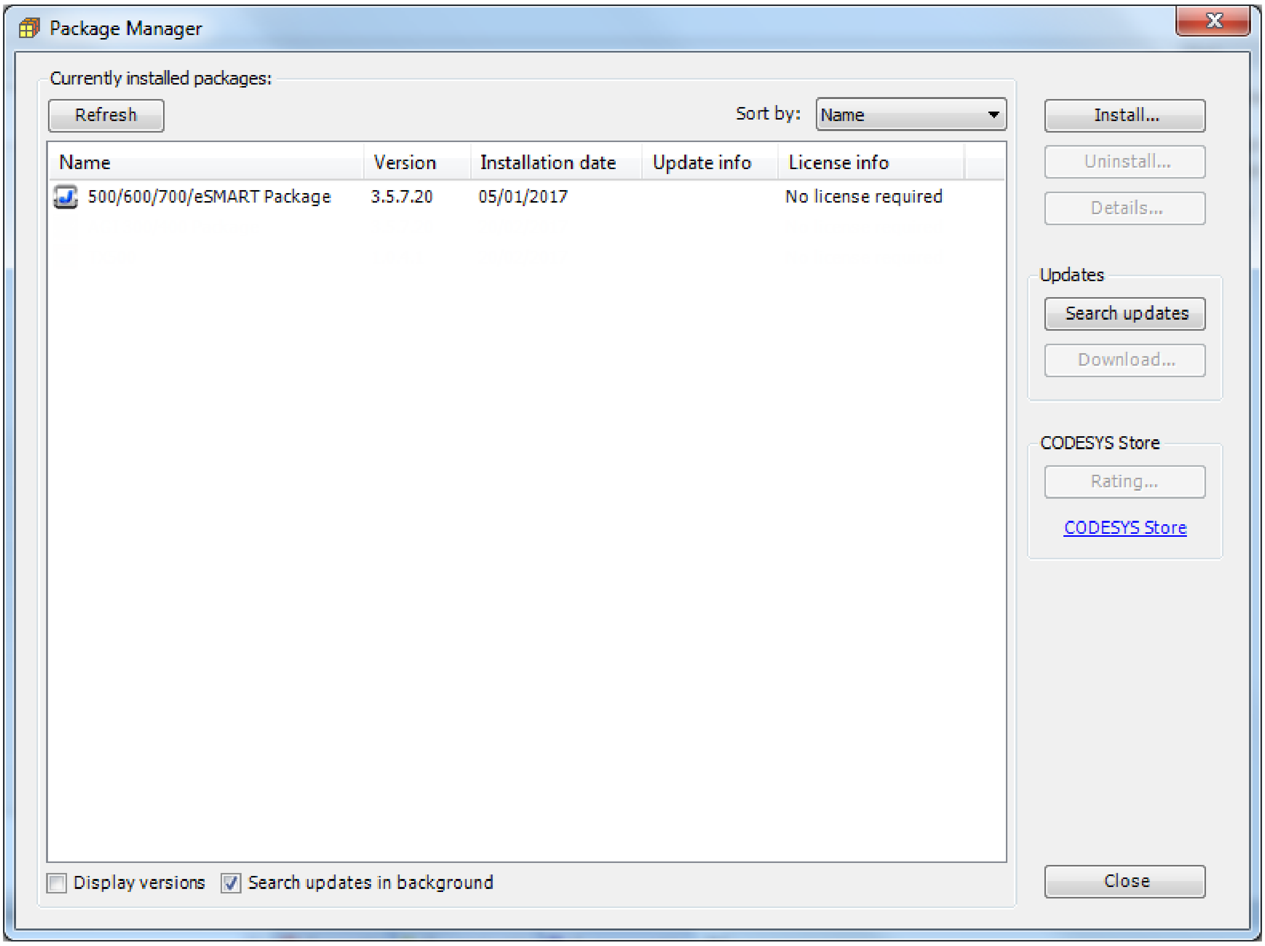
A dedicated CODESYS Package is required to allow the standard CODESYS V3 programming software to program Exor HMIs internal PLC.

Exor CODESYS Package is included into Internal PLC (CODESYS) since version 1.91 SP1 under "\CODESYS\V3\" folder.

The package can be also downloaded from EXOR websites under *Support -> JMobile -> Documents and Download* section.

To install the package double click on .package file. The installation procedure will start automatically, the system asks if to perform a Complete or Typical setup, any of these will install all the required files for a complete support of the CODESYS V3 PLC Runtime features.

The Package Manager dialog is visible in the following figure, showing the installed Package.



Currently installed packages:

Sort by: Name

Name	Version	Installation date	Update info	License info
500/600/700/eSMART Package	3.5.7.20	05/01/2017		No license required

☐ Display versions ☑ Search updates in background

Note: CODESYS Development System includes a tool called "Package Manager" for the installation of the CODESYS Package.
Package Manager tool can be launched from CODESYS Tools menu, selecting the proper menu item. This tool can be used both for checking the installed Packages and for installing new ones.
To install the Exor CODESYS Package launch Package Manager and click on "Install" button. Browse for the file with .package extension, confirming with "Open".

Updating Old CODESYS Packages

When a new version of the CODESYS Package is available, it can be installed to update the Package information accordingly to the latest CODESYS V3 PLC Runtime features.

When a new version of Package is released Exor always tries to guarantee the complete compatibility with the previous versions.

If the new Package is not fully compatible with the previous version some changes to adapt the PLC application to the new Package may be required.

The Package update procedure is identical to the first installation of the CODESYS Package, please refer to the proper chapter for specific information.

Note: CODESYS V3 keeps the older Package versions instead of replacing them in order to have all the installed Packages at disposal if required.

HMI Runtime installation with Internal PLC (CODESYS)

The HMI and control system is composed by two main subsystems, the HMI Runtime and the CODESYS V3 PLC Runtime.

As the CODESYS PLC Runtime is part of the HMI Runtime it is necessary to install the HMI Runtime on the HMI to have the PLC Runtime running.

For further information on installing the HMI Runtime, please refer to Internal PLC (CODESYS) User's Manual.

Activation of CODESYS V3 license on the HMI

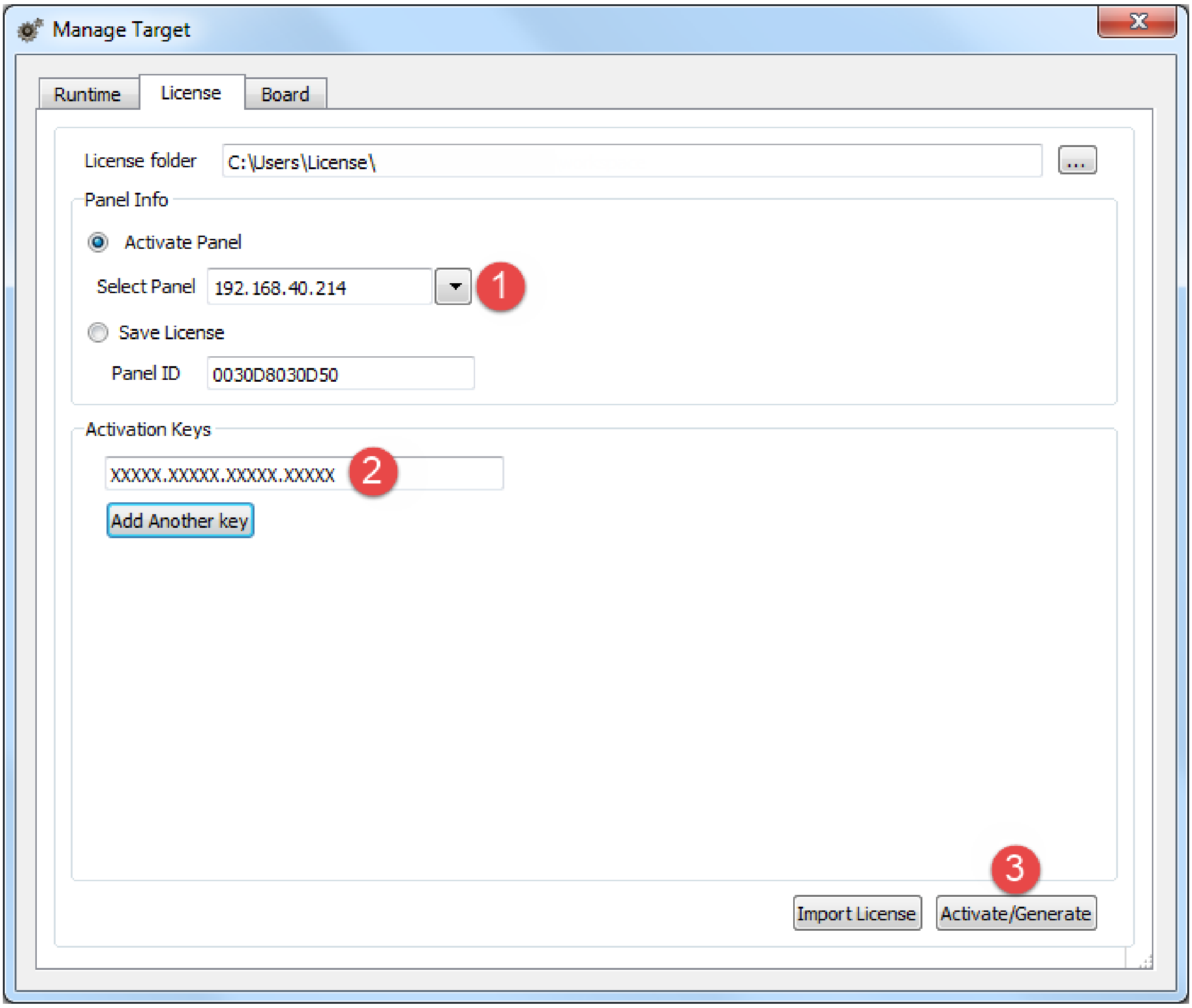
Note: For testing purposes it is possible to use a temporary demo license that will activate the CODESYS V3 PLC Runtime for 120 minutes. To activate the demo mode enter the license code CODESYS_DEMO.

CODESYS V3 PLC Runtime license is activated on the HMI through an Ethernet connection using Internal PLC (CODESYS), each license code can be used on a single HMI, once activated the license is uniquely matched with the Mac Address of the HMI.

To activate the license follow the below steps:

- In Internal PLC (CODESYS) select Run > Manage Target
- Into the License tab select, from the drop-down menu in the Panel Info section, the IP address of the HMI where the license will be activated from the list of HMIs available into the network.
- Type in the Activation keys section the license code.
- Click on the Activate/Generate button.
- Once the system confirms the license to be successfully activated reboot the HMI to complete the activation procedure.

Note: The CODESYS V3 license activation procedure requires an active internet connection on the PC.



License folder: C:\Users\License\

● Activate Panel

Select Panel: 192.168.40.214

○ Save License

Panel ID: 0030D8030D50

Activation Keys: XXXXX.XXXXX.XXXXX.XXXXX

Note: CODESYS V3 licenses, once activated cannot be paired with a different device or de-activated.

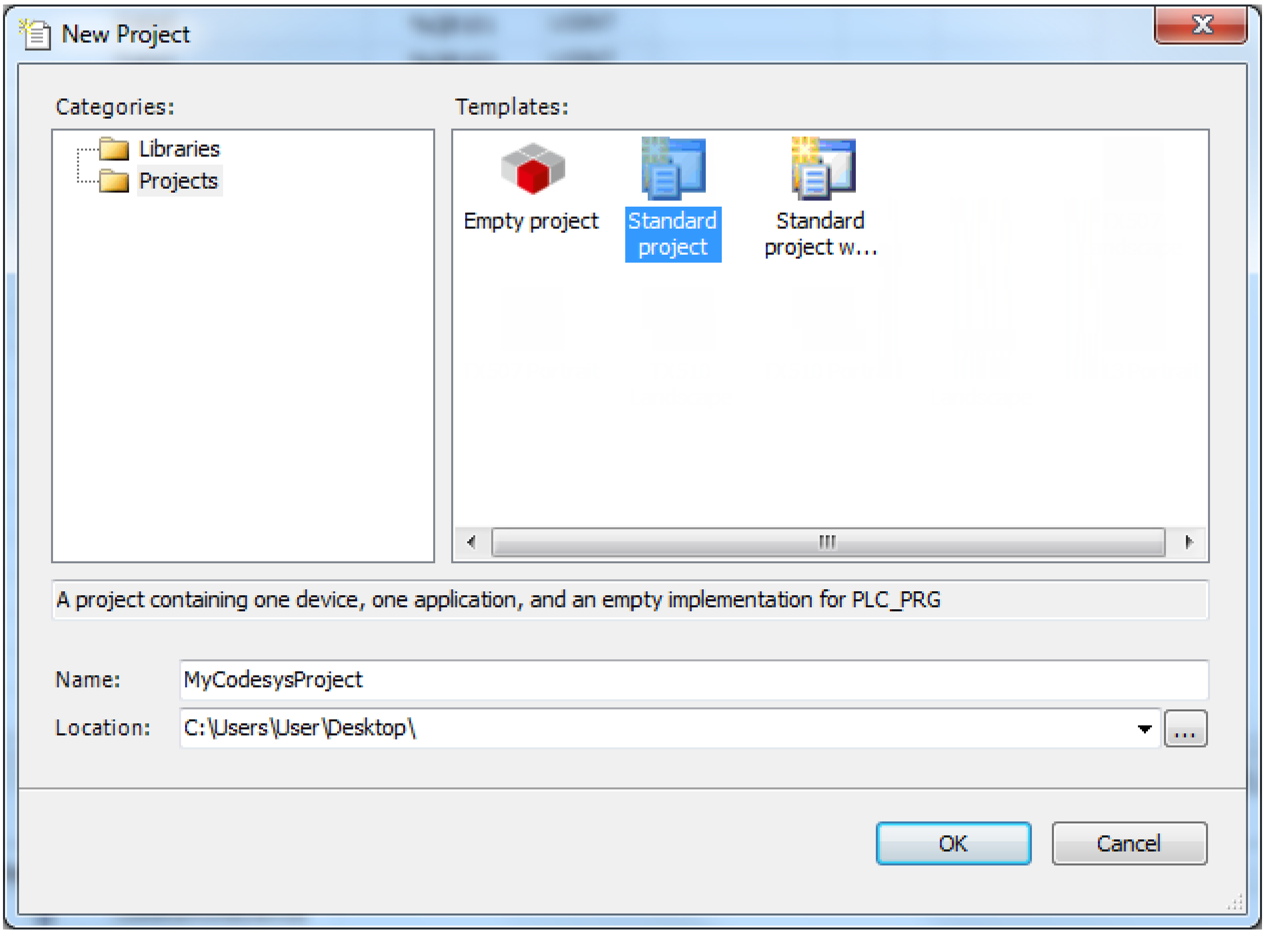
In case it is necessary to make sure whether a license has been activated on an HMI it is possible to verify the system log of the HMI.

From the context menu on the HMI select the option “Log at boot”, and then reboot the HMI. On restart the Log window will be displayed on screen, if a valid CODESYS V3 license is found from the system the string “CODESYS Module: CODESYS V3 license found: CODESYS V3 is running” will be present among the HMI boot logging information.

Creation of a new PLC project

To create a new CODESYS V3 project select File > New Project or click on the icon from the upper tools bar.

Select the “Standard project” template, define then Project Name and Location, then confirm with OK.

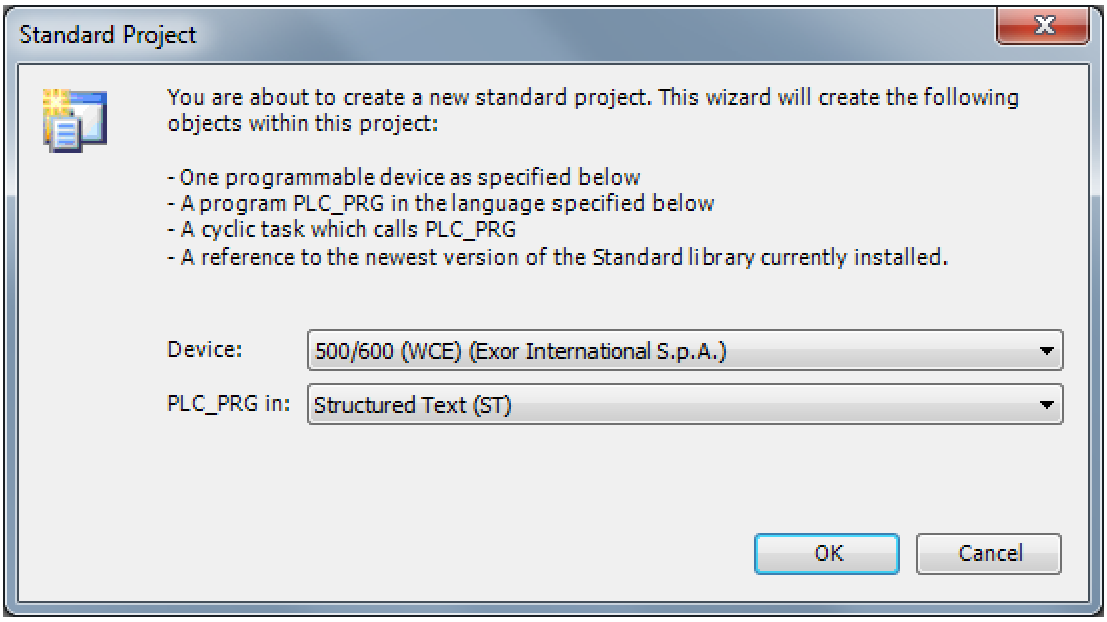


A project containing one device, one application, and an empty implementation for PLC_PRG

Name: MyCodesysProject

Location: C:\Users\User\Desktop\

In the next window select the Device and the programming language to use, as shown in picture below.



You are about to create a new standard project. This wizard will create the following objects within this project:

- One programmable device as specified below
- A program PLC_PRG in the language specified below
- A cyclic task which calls PLC_PRG
- A reference to the newest version of the Standard library currently installed.

Device: 500/600 (WCE) (Exor International S.p.A.)

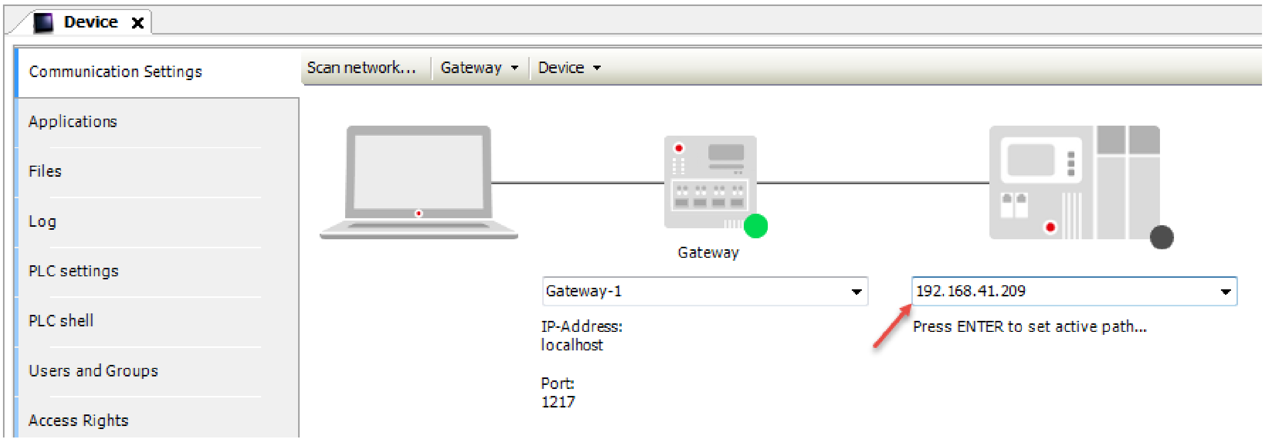
PLC_PRG in: Structured Text (ST)

Download PLC application

The selection of the PLC where to download the project must be done from Device communication settings tab before proceeding with the download operation.

Double click on "**Device (<*device_name*>)**" in the project tree to display the Device properties in the work area.

Insert the IP address of the PLC and press ENTER key.



Gateway-1

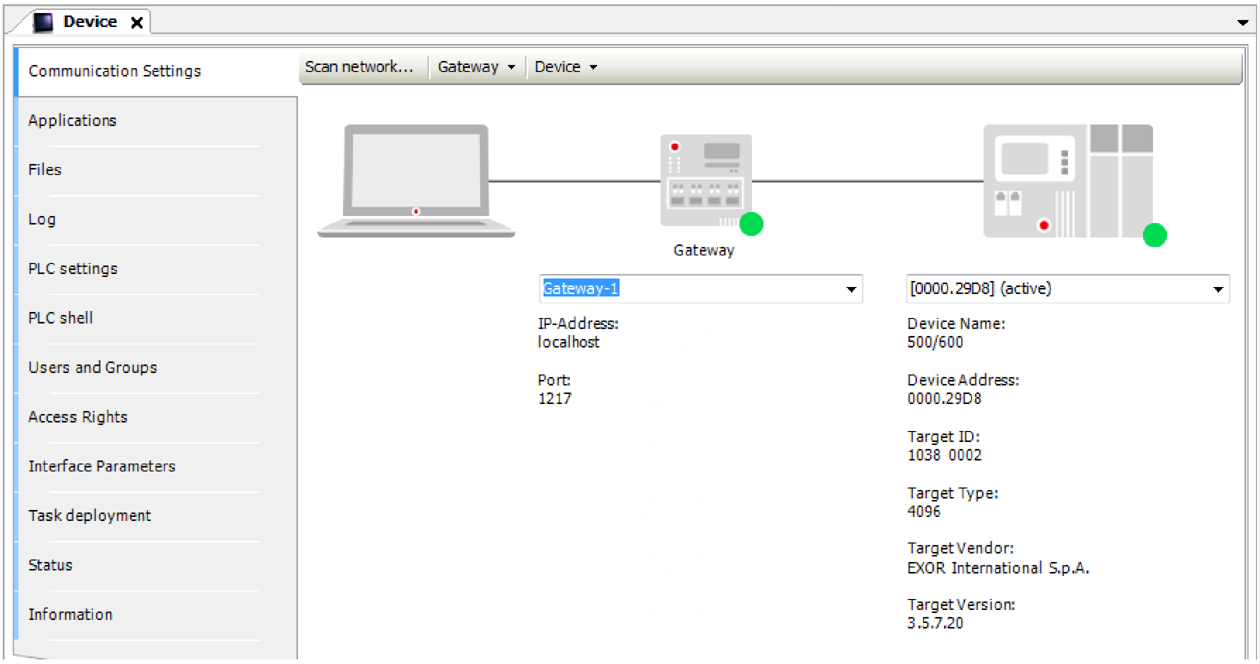
IP-Address:
localhost

Port:
1217

192.168.41.209

Press ENTER to set active path...

A green dot over the device graphical representation informs that the device is correctly recognized and available into the network.



Gateway-1

IP-Address:
localhost

Port:
1217

[0000.29D8] (active)

Device Name:
500/600

Device Address:
0000.29D8

Target ID:
1038 0002

Target Type:
4096

Target Vendor:
EXOR International S.p.A.

Target Version:
3.5.7.20

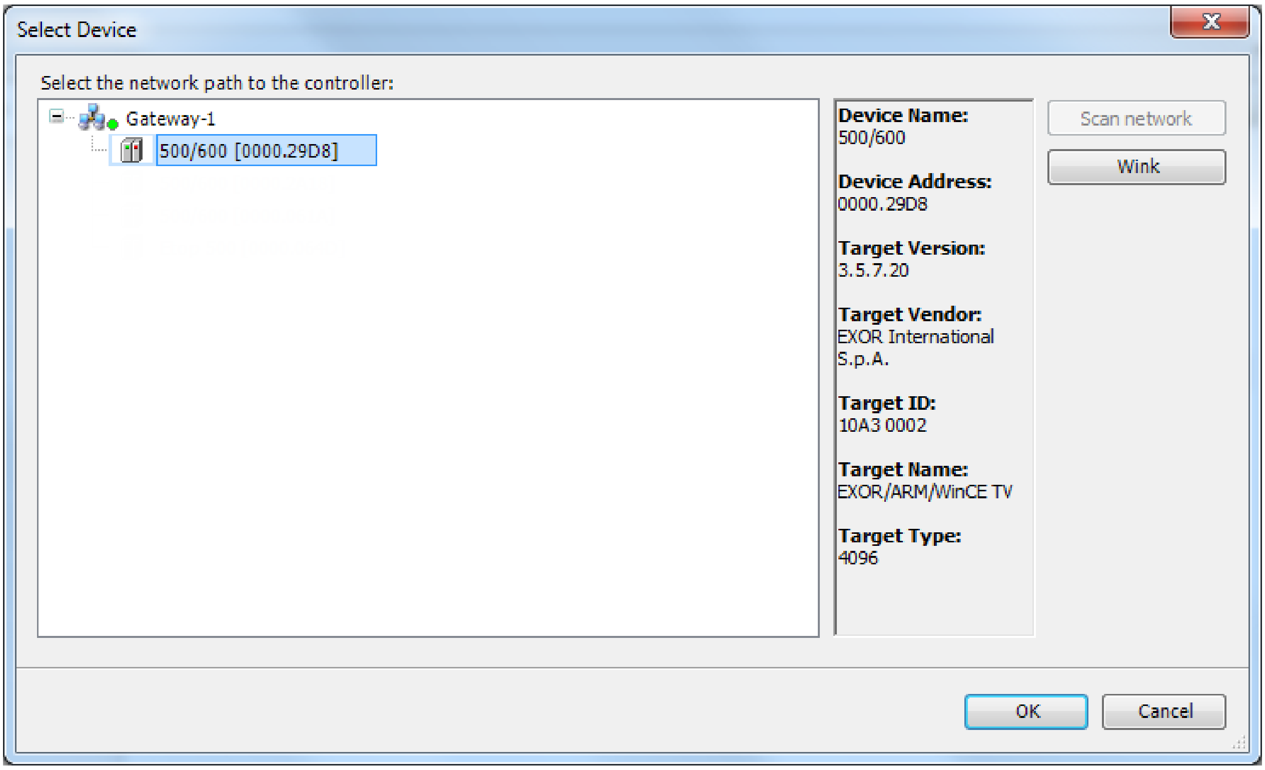
Note: Communication with the available devices is established through a Gateway, a default Gateway is available, and it is generally not needed to change the standard Gateway settings. For more information about the Gateway set-up please refer to CODESYS V3 documentation.

If PLC IP Address is unknown, select the "Communication Settings" tab then click on the "Scan Network..." button.

The Select Device dialog will be displayed. This dialog lists all the compatible devices available in the network. Select from the list the HMI to be programmed then press on “OK”.

In case more HMIs using CODESYS V3 PLC Runtime are present into the network each HMI is recognized by a different string between square brackets reported after Device name.

In the next figure the string is “0000.29D8”, the last part of the string “29D8” corresponds to the last 2 bytes of the HMI IP Address in Hex format so, in this case, the corresponding HMI is the one with IP address xxx.xxx.41.216 as 29Hex corresponds to 41Dec and D8 Hex corresponds to 216 Dec.


Select Device

Select the network path to the controller:

- Gateway-1
  - 500/600 [0000.29D8]

Device Name:
500/600

Device Address:
0000.29D8

Target Version:
3.5.7.20

Target Vendor:
EXOR International S.p.A.

Target ID:
10A3 0002

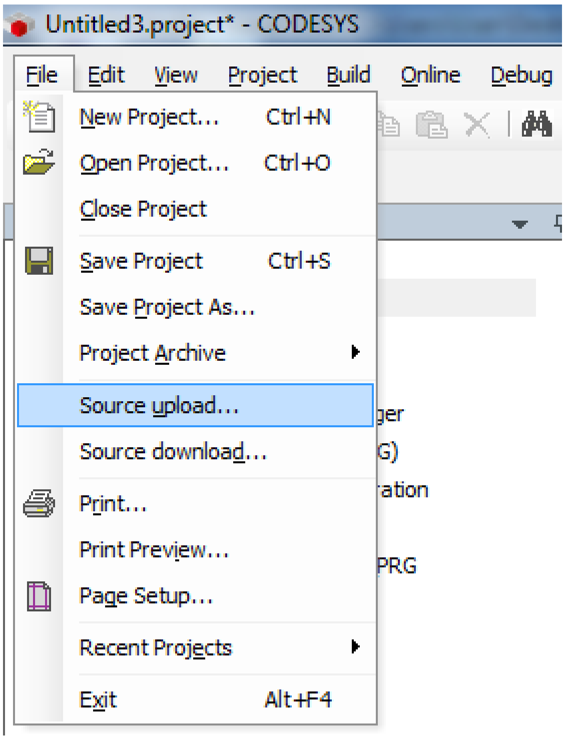
Target Name:
EXOR/ARM/WinCE TV

Target Type:
4096

Upload PLC Application

The upload of the PLC project is possible only if the project Source has been previously downloaded into the PLC. To download the project source, while Online, select the Online > Source download to connected device command.

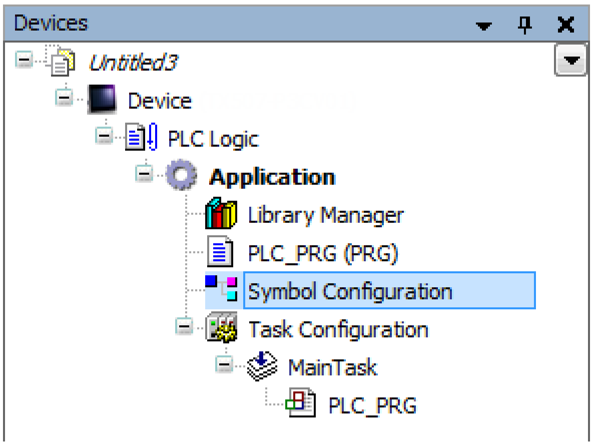
To upload a PLC project from the HMI, select the Source upload command from the File menu, as shown in the following figure, then select the HMI from Select Device window.



Symbol File configuration with CODESYS V3 Development System

When creating the project using CODESYS V3, properly configure the symbol file to contain the required variables.

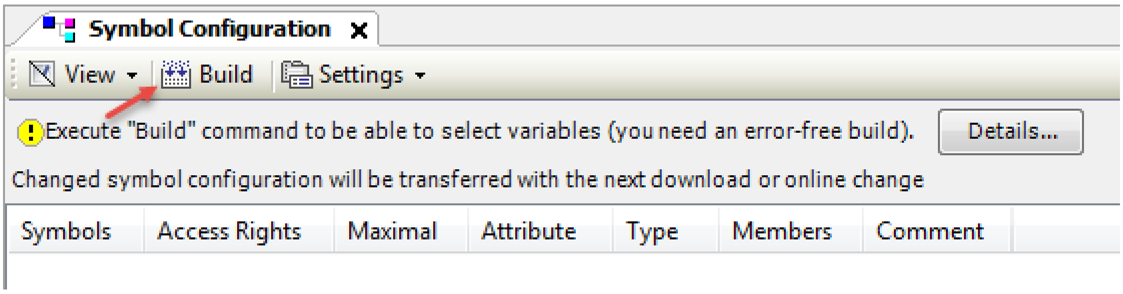
1. To add the Symbol configuration in CODESYS V3 project, right click on the Application item from the project tree, then into the context menu select Add Object > Symbol configuration. The symbol configuration item will be added to the project tree.



Devices

- Untitled3
  - Device
    - PLC Logic
      - Application
        - Library Manager
        - PLC_PRG (PRG)
        - Symbol Configuration
        - Task Configuration
          - MainTask
            - PLC_PRG

2. Double click on Symbol configuration item, then click on "Build" button.



Symbol Configuration

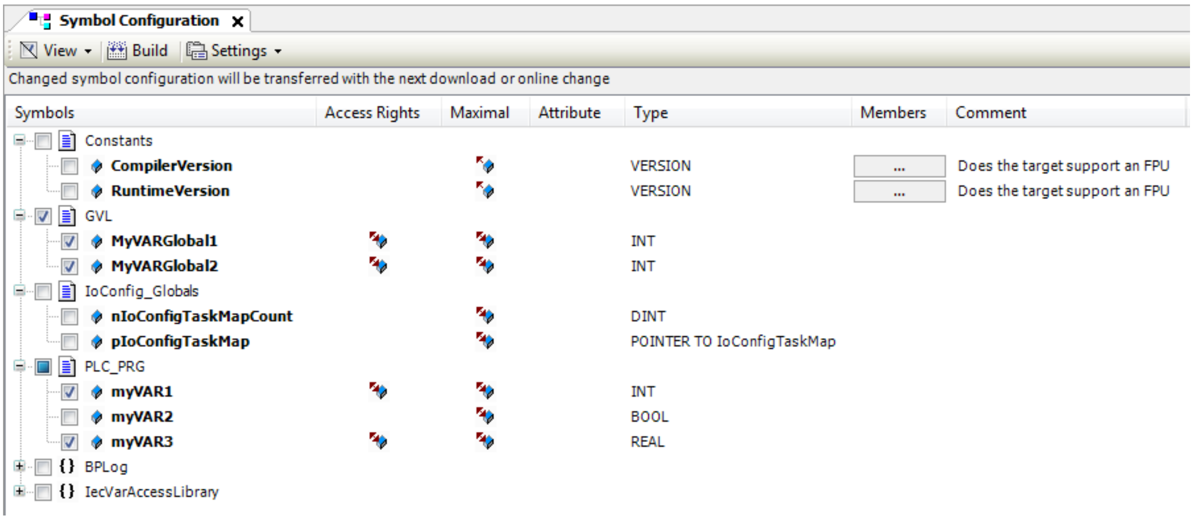
View | Build | Settings

Execute "Build" command to be able to select variables (you need an error-free build). Details...

Changed symbol configuration will be transferred with the next download or online change

Symbols	Access Rights	Maximal	Attribute	Type	Members	Comment

3. Symbol configuration item contains a list of all the variables available into the CODESYS V3 project, single variables or groups of variables can be selected by checking the corresponding item in the list.



Symbol Configuration

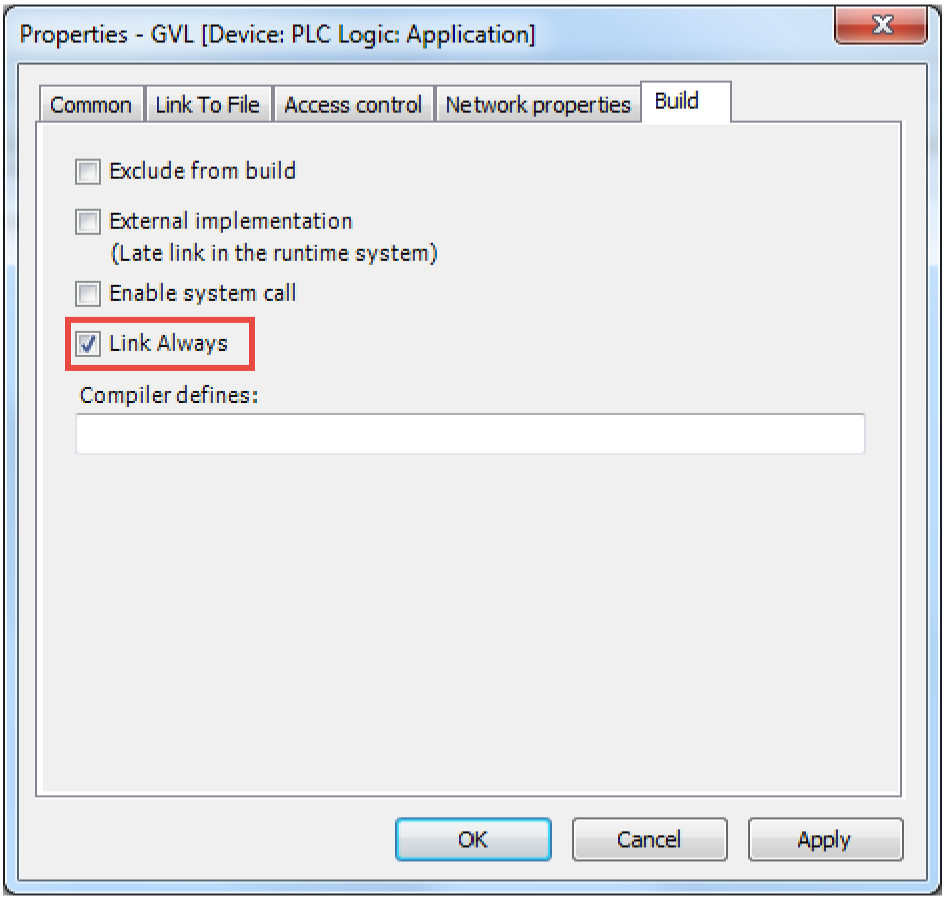
View | Build | Settings

Changed symbol configuration will be transferred with the next download or online change

Symbols	Access Rights	Maximal	Attribute	Type	Members	Comment
□ Constants						
□ **CompilerVersion**		⇄		VERSION	...	Does the target support an FPU
□ **RuntimeVersion**		⇄		VERSION	...	Does the target support an FPU
☑ GVL						
☑ **MyVARGlobal1**	⇄	⇄		INT		
☑ **MyVARGlobal2**	⇄	⇄		INT		
□ IoConfig_Globals						
□ **nIoConfigTaskMapCount**		⇄		DINT		
□ **pIoConfigTaskMap**		⇄		POINTER TO IoConfigTaskMap		
■ PLC_PRG						
☑ **myVAR1**	⇄	⇄		INT		
□ **myVAR2**		⇄		BOOL		
☑ **myVAR3**	⇄	⇄		REAL		
□ {} BPLog						
□ {} IecVarAccessLibrary						

4. After the symbols have been configured, download the project or use the **Generate code** function (Build > Generate code) to create an .xml file containing all the variables read to be imported in the Tag Editor.

Note: GVL global variables are listed in Symbols Configuration only if they are used in PLC program.
To always list global variables right click on GVL and select "Properties". From "Build" tab check "Link Always" option.


Properties - GVL [Device: PLC Logic: Application]

Common | Link To File | Access control | Network properties | Build

□ Exclude from build

□ External implementation
(Late link in the runtime system)

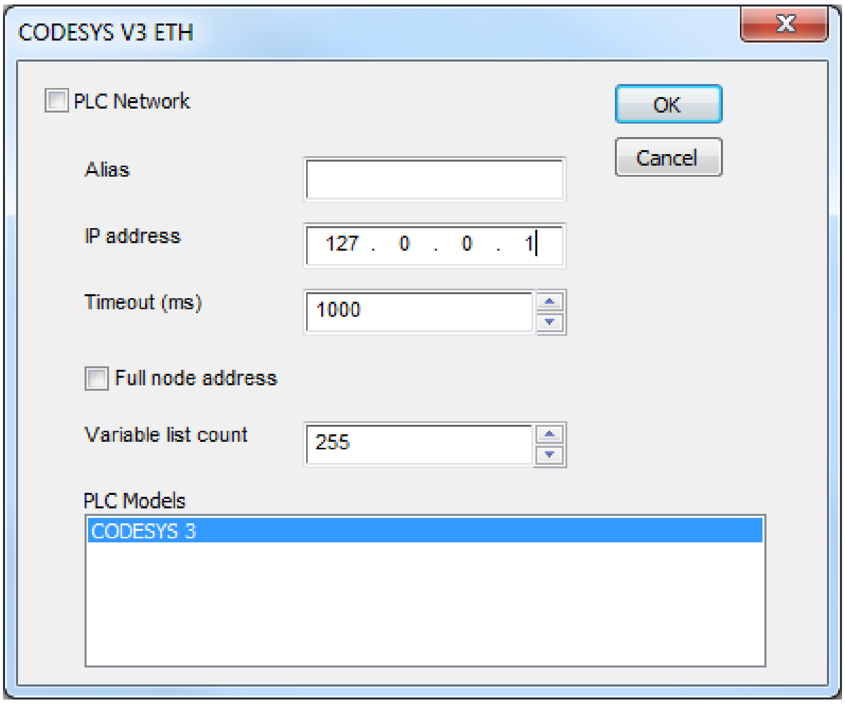
□ Enable system call

☑ Link Always

Compiler defines:

Communication Setup in Internal PLC (CODESYS)

HMI Runtime communicates with the internal CODESYS V3 Runtime using the CODESYS V3 ETH protocol. Localhost 127.0.0.1 should be entered in IP Address parameter, this identifies the PLC as internal CODESYS V3 Runtime.



CODESYS V3 ETH

☐ PLC Network

Alias

IP address 127 . 0 . 0 . 1

Timeout (ms) 1000

☐ Full node address

Variable list count 255

PLC Models

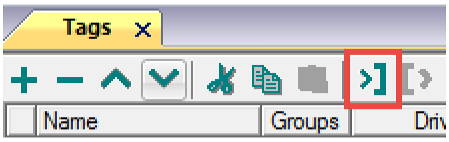
CODESYS 3

Additional information regarding CODESYS V3 ETH driver are available in Internal PLC (CODESYS) *Help* > *Communication Drivers* section.

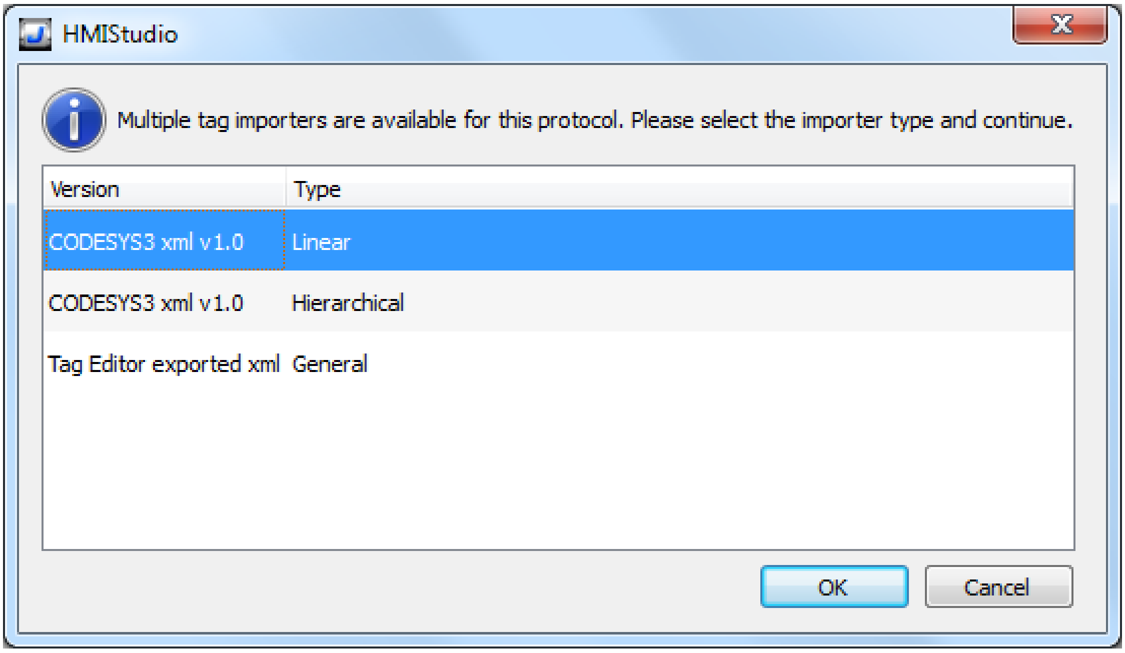
Import Symbol Files in Internal PLC (CODESYS)

Internal PLC (CODESYS) Tag Editor requires direct import of CODESYS V3 symbol file for the definition of the Tags.

Select the driver in Tag Editor and click on the **Import Tags** button to start the importer.



The following dialog shows which importer type can be selected.



Multiple tag importers are available for this protocol. Please select the importer type and continue.

Version	Type
CODESYS3 xml v1.0	Linear
CODESYS3 xml v1.0	Hierarchical
Tag Editor exported xml	General

Note: Additional information regarding CODESYS V3 ETH Tag import are available inInternal PLC (CODESYS) *Help > Communication Drivers* section.

CODESYS V3 Development System generates a new version of the ".xml" file each time the PLC project is built.

CODESYS V3 ETH communication driver supports automatic symbol file (SDB) upload from the PLC; the HMI can upload the symbol table from the PLC using the communication protocol.

Any change in the tag offset information due to a new compilation of the PLC program does not require importing again the symbol file.

Symbol file must be imported again when:

- Tags have been renamed
- Tags have changed data format
- New tags have been added

Memory Layout

The table below shows detailed information about maximum allowed size of CODESYS V3 project.

Memory Type	Maximum Size
Data, Input, Output, Memory and Code	20 MB
Retain	16 KB
Persistent	16 KB

Maximum number of connections

CODESYS V3 internal PLC supports up to 4 concurrent connections to other devices.

Communication Diagnostic

HMI Runtime reports communication diagnostic information also for the communication with the internal PLC Runtime. To display communication diagnostic information use the proper System Variables or the System Logger. For further information please refer to Internal PLC (CODESYS) Help.

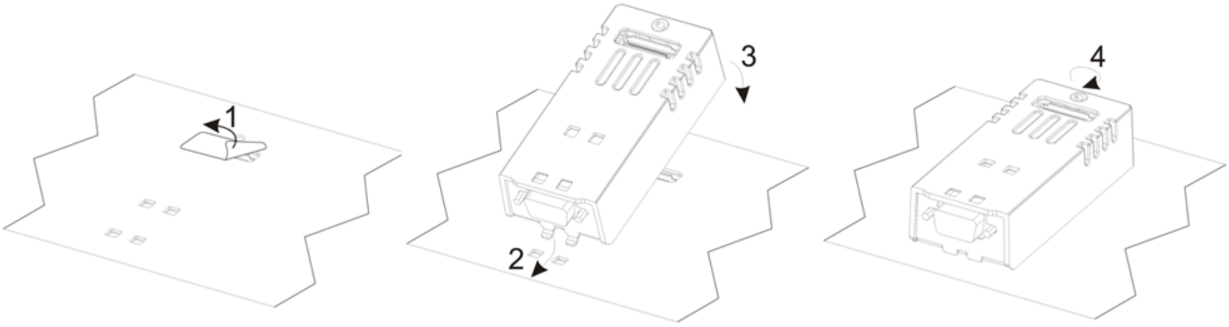
Installing Optional Modules

Optional modules (Communication Modules and I/O Modules) can be mounted on the back of HMIs.

Note: Refer to table in "Control Solution with HMI and CODESYS V3" chapter, under "Optional Plugins" column to know if your HMI support optional modules.

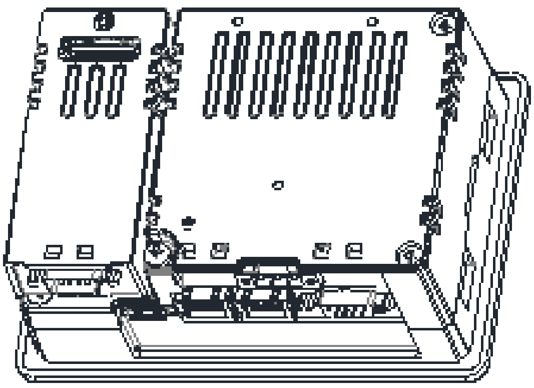
The steps to follow for the installation of the optional module are shown in the next figure and are the following:

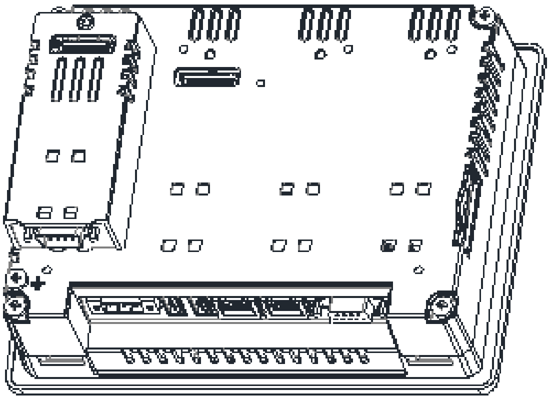
1. If present, remove the sticker covering the plug-in slot
2. Insert the guides of the module into the holes in the enclosure
3. Press the module down into the connector
4. Lock the module using the screw



Note: Do not plug or remove optional modules when the HMI device is powered.

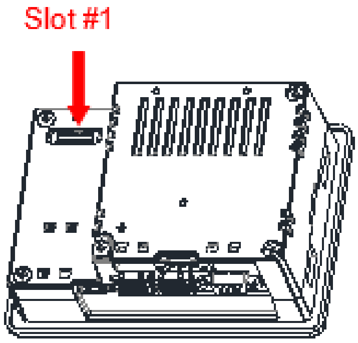
The following figure shows HMIs with optional module mounted.

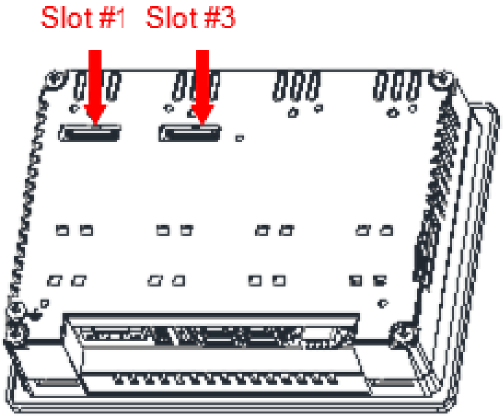


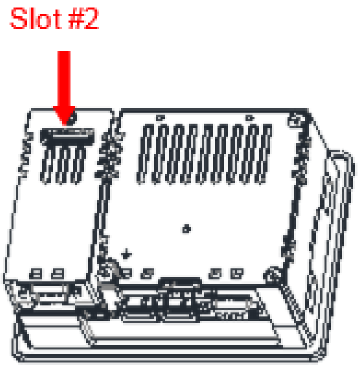


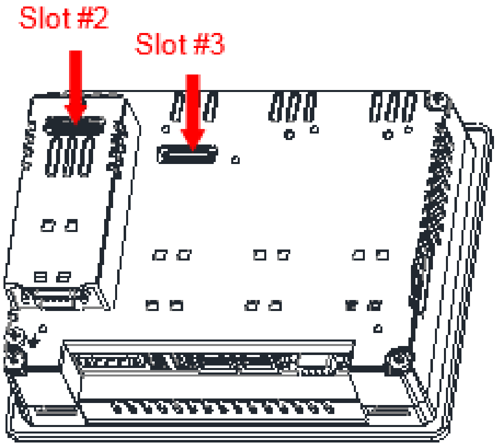
Slot Numbers

The following picture shows numbering of Slots in HMIs.







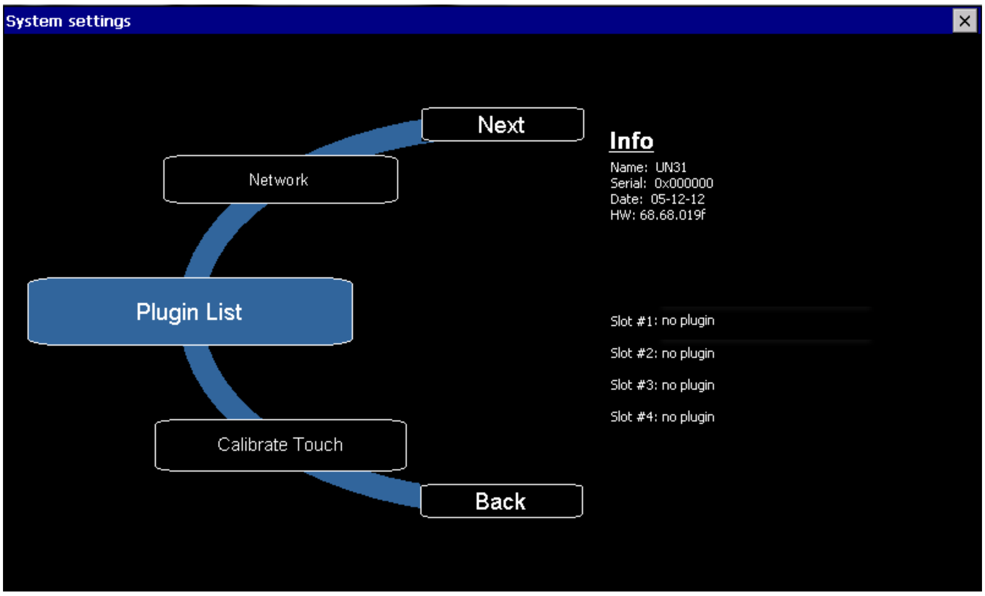


Note: Slot #4 is located on the top of the option module plugged into Slot #3.

Recognizing optional modules installed on HMI

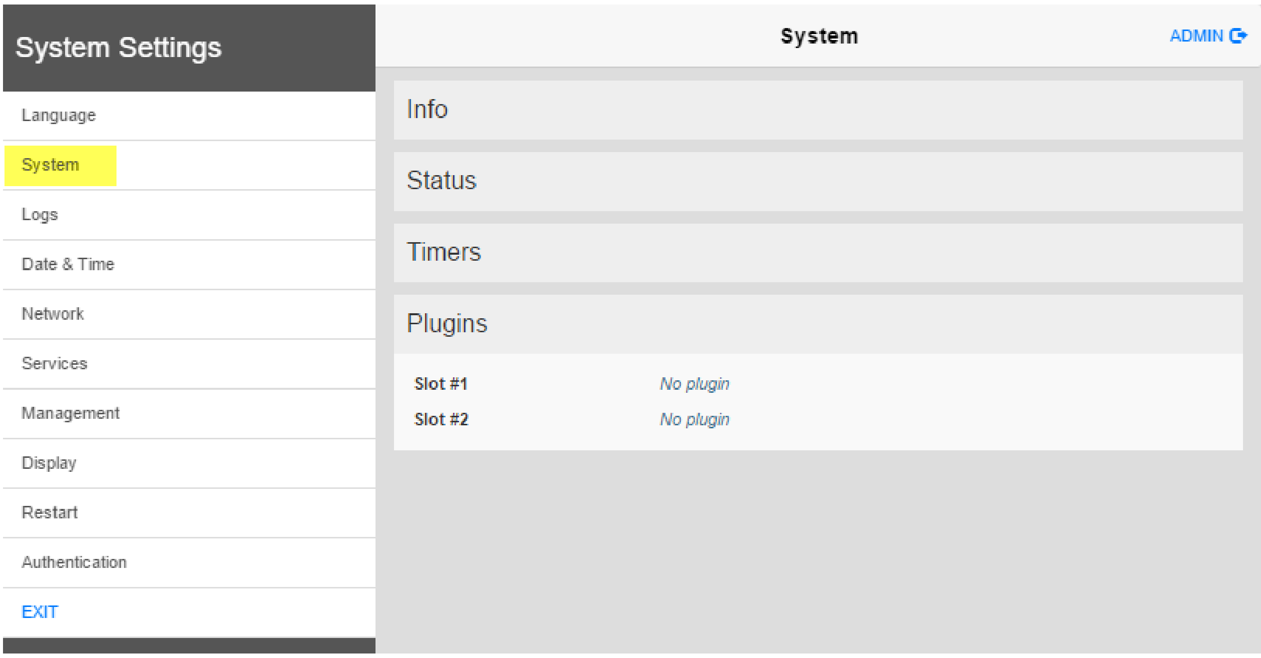
If correctly installed on the operator panel and recognized by the system the optional modules are listed into the Plugin List.

The Plugin List is an item of System Settings menu.



Name: UN31
Serial: 0x000000
Date: 05-12-12
HW: 68.68.019f

Slot #1: no plugin
Slot #2: no plugin
Slot #3: no plugin
Slot #4: no plugin

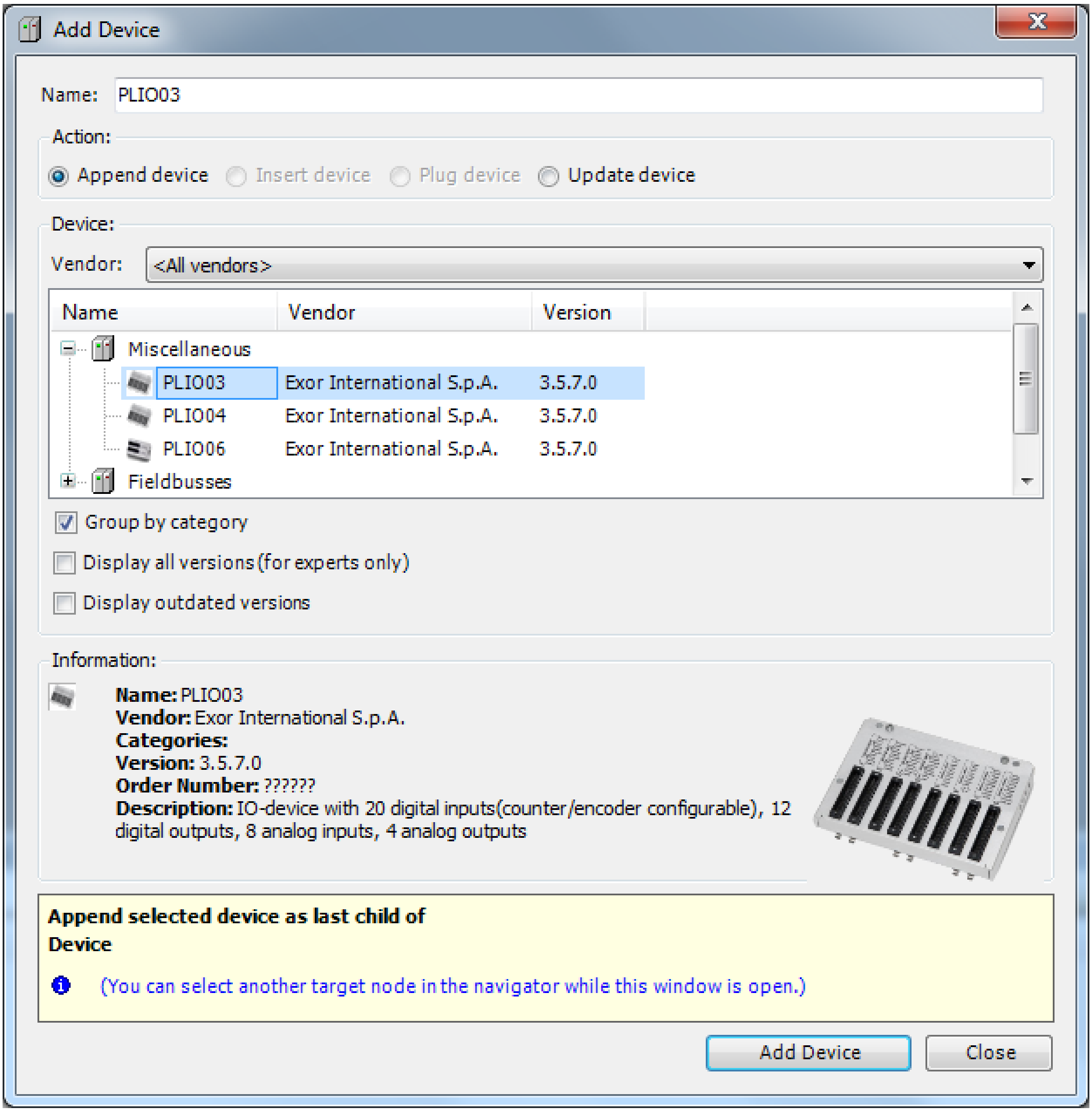


Slot #1 *No plugin*
Slot #2 *No plugin*

Configuring Local I/O

Exor HMI allows creating control systems with integrated I/O. Integrated I/O systems based on optional modules can be easily configured using CODESYS V3 PLC Configuration.

To add an I/O optional module, right click on **Device (<*device_name*>)** in Project tree and select “Add Device”.
I/O optional modules are located under the *Miscellaneous* category.
Choose I/O optional module and click on “Add Device” to add it to the current PLC configuration.



Add Device

Name: PLIO03

Action:

● Append device ○ Insert device ○ Plug device ○ Update device

Device:

Vendor: <All vendors>

Name	Vendor	Version
Miscellaneous		
PLIO03	Exor International S.p.A.	3.5.7.0
PLIO04	Exor International S.p.A.	3.5.7.0
PLIO06	Exor International S.p.A.	3.5.7.0
Fieldbusses		

☑ Group by category

□ Display all versions (for experts only)

□ Display outdated versions

Information:

Name: PLIO03
Vendor: Exor International S.p.A.
Categories:
Version: 3.5.7.0
Order Number: ??????
Description: IO-device with 20 digital inputs(counter/encoder configurable), 12 digital outputs, 8 analog inputs, 4 analog outputs

Append selected device as last child of
Device

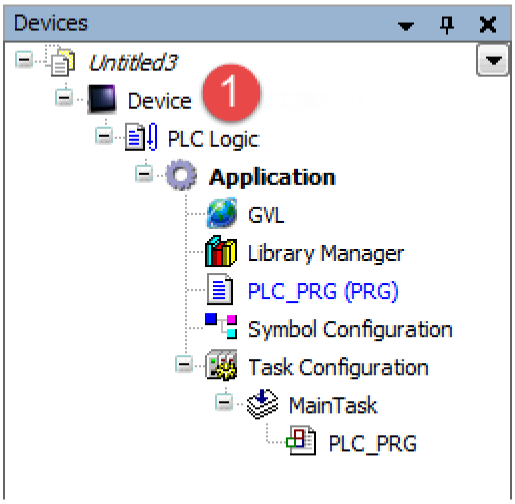
(You can select another target node in the navigator while this window is open.)

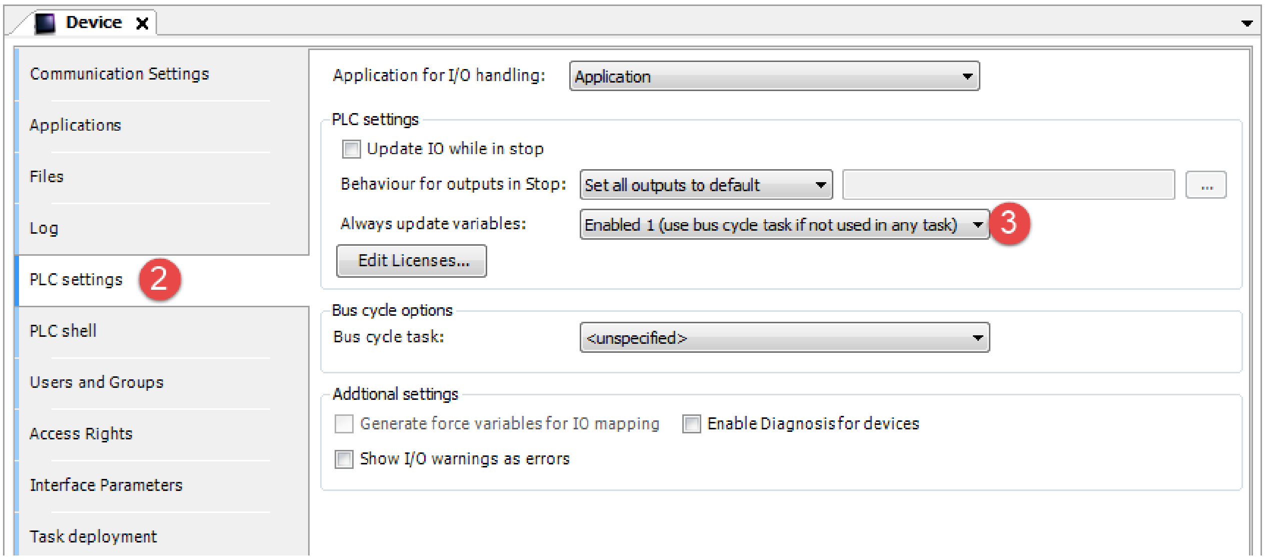
The parameters of all I/O optional modules are collected in single tab of the PLC Configuration tree.

Update I/O variables

To always update I/O variables:

1. Double click on **Device (<*device_name*>)**
2. Select PLC settings tab
3. Set "Enabled 1 (use bus cycle task if not used in any task)" in "Always update variables" option





Application for I/O handling: Application

PLC settings

□ Update IO while in stop

Behaviour for outputs in Stop: Set all outputs to default

Always update variables: Enabled 1 (use bus cycle task if not used in any task)

Bus cycle options

Bus cycle task: <unspecified>

Addtional settings

□ Generate force variables for IO mapping □ Enable Diagnosis for devices

□ Show I/O warnings as errors

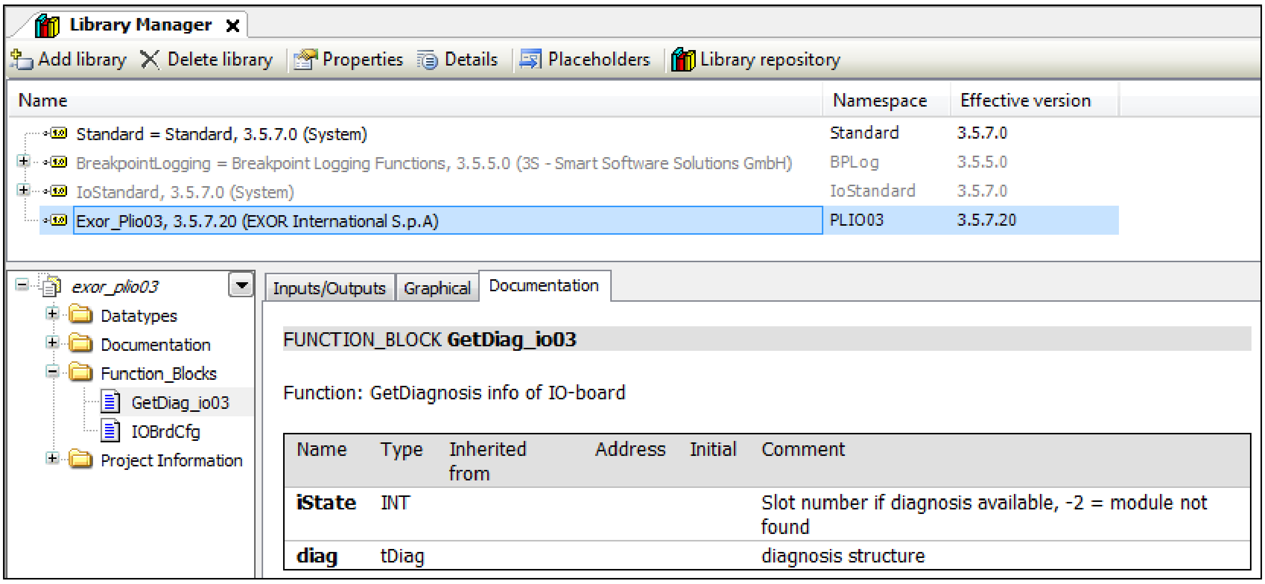
Available options are:

Option	Description
Disabled	I/O variables are updated only if used in a task
Enabled 1	I/O variables are always updated by using bus cycle task if not used in any task
Enabled 2	I/O variables are always updated by using bus cycle task

The above option will be inherited by a specific I/O item if "Always update variables" option is set as "Use parent device setting", from *I/O Mapping* tab.

I/O Modules Diagnostic

I/O module diagnostic can be check by using the function block **GetDiag_io0x** included into I/O module library, as shown in picture below.



Name	Namespace	Effective version
Standard = Standard, 3.5.7.0 (System)	Standard	3.5.7.0
BreakpointLogging = Breakpoint Logging Functions, 3.5.5.0 (3S - Smart Software Solutions GmbH)	BPLog	3.5.5.0
IoStandard, 3.5.7.0 (System)	IoStandard	3.5.7.0
Exor_Plio03, 3.5.7.20 (EXOR International S.p.A)	PLIO03	3.5.7.20

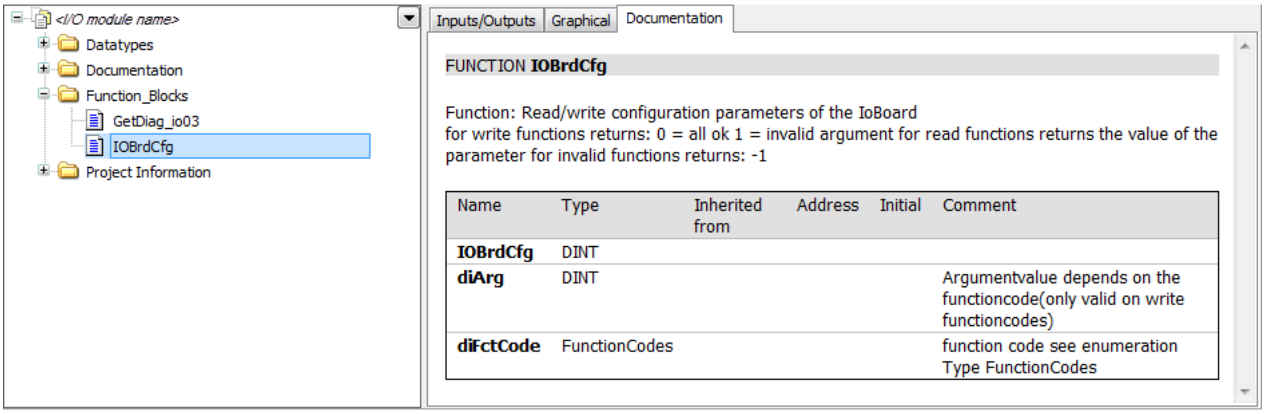
FUNCTION_BLOCK **GetDiag_io03**

Function: GetDiagnosis info of IO-board

Name	Type	Inherited from	Address	Initial	Comment
iState	INT				Slot number if diagnosis available, -2 = module not found
diag	tDiag				diagnosis structure

IOBrdCfg Function Block

Integrated I/O boards can be configured at design-time from the I/O specific configuration page.
The I/O board configuration can be modified also in runtime, to do this use the **IOBrdCfg** function block.



FUNCTION **IOBrdCfg**

Function: Read/write configuration parameters of the IoBoard
for write functions returns: 0 = all ok 1 = invalid argument for read functions returns the value of the parameter for invalid functions returns: -1

Name	Type	Inherited from	Address	Initial	Comment
IOBrdCfg	DINT				
diArg	DINT				Argumentvalue depends on the functioncode(only valid on write functioncodes)
diFctCode	FunctionCodes				function code see enumeration Type FunctionCodes

This function block is included in I/O module library and has the following input parameters:

Element	Description
diFctCode	This parameter is passed to the I/O board; it significance changes from board to board.
diArg	This parameter is passed to the I/O board; its significance varies from board to board.

The function has the following output parameters:

Element	Description
IOBrdCfg	This is the return value; its significance varies from board to board and from FunctionCode to FunctionCode.

Note: The available parameters are described in the specific I/O module chapter.

I/O Modules Points Address Mapping

CODESYS V3 allows three different Mapping modes:

- When no specific Mapping Variable is specified the system automatically assigns a physical address to each I/O point.
- A new Variable can be created by double clicking on the icon of the corresponding I/O point and writing the Variable name in the editing box. Such mapped I/O are marked with symbol.
- An existing variable created in the project can be assigned to the I/O point, by double clicking on the icon of the corresponding I/O point and selecting then the Variable from the list of available by clicking on the ... button. Such mapped I/O are marked with symbol. Using this method the I/O point will not refer anymore to the Physical address assigned by default from the system.

PLIO03

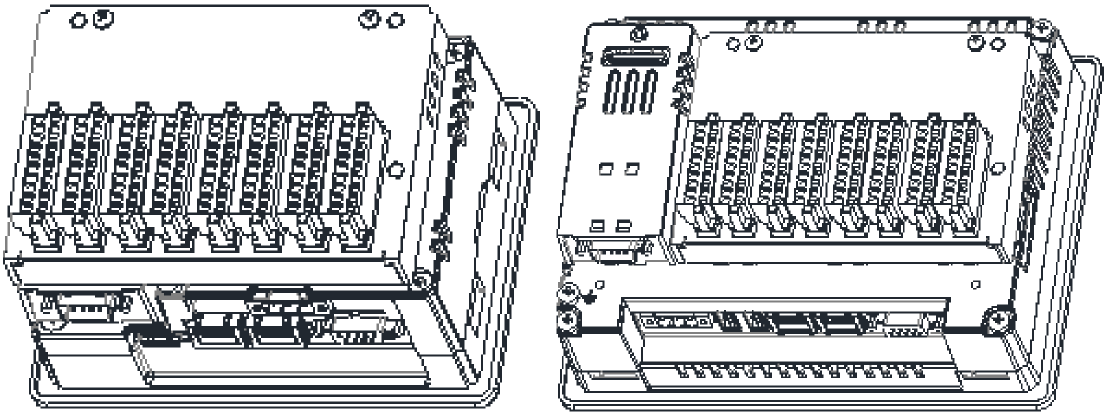
PLIO03 is a multifunction I/O module.

Thanks to the high level of integration and configurability, PLIO03 offers a “one-board solution” for most typical I/O configurations in simple applications.

PLIO03 specifications are the following:

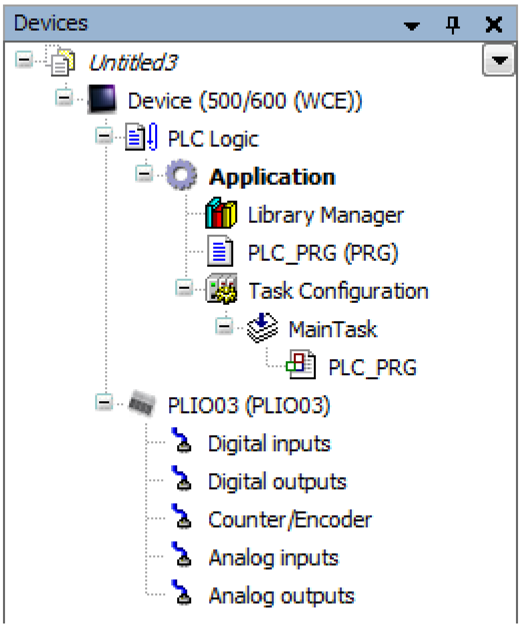
- 20 optically isolated digital inputs; configurable as counter/encoder channels
- 2 optically isolated digital outputs
- 4 non-isolated 12 bit analog inputs configurable as 4 differential channels for voltage, current, resistance, temperature (PT100 and thermocouple) measurement or configurable as 8 single ended channels for voltage measurement
- 4 non-isolated 12 bit analog outputs configurable for voltage or current
- 1 dedicated PT100 input for cold junction compensation of thermocouples

The next figure shows a configuration for HMIs, using a communication module and PLIO03 module.



Note: The above pictures shows two types of HMI.
HMIs of the first type require the use of a Bus extender module for the installation of the PLIO03 I/O module.

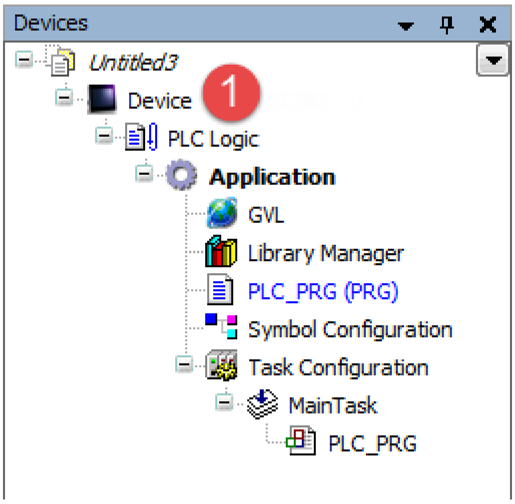
After including the PLIO03 module in the PLC configuration, as shown in the following figure, each I/O type of the module can be configured separately.

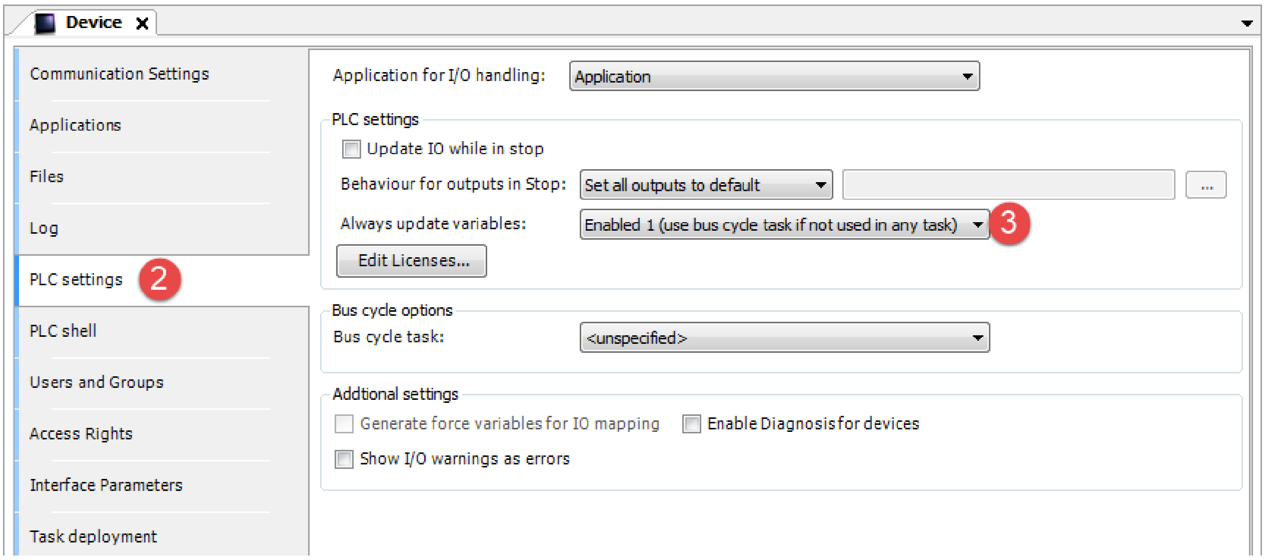


Update I/O variables

To always update I/O variables:

1. Double click on **Device (<*device_name*>)**
2. Select PLC settings tab
3. Set "Enabled 1 (use bus cycle task if not used in any task)" in "Always update variables" option





Application for I/O handling: Application

PLC settings

□ Update IO while in stop

Behaviour for outputs in Stop: Set all outputs to default

Always update variables: Enabled 1 (use bus cycle task if not used in any task)

Bus cycle options

Bus cycle task: <unspecified>

Addtional settings

□ Generate force variables for IO mapping □ Enable Diagnosis for devices

□ Show I/O warnings as errors

Available options are:

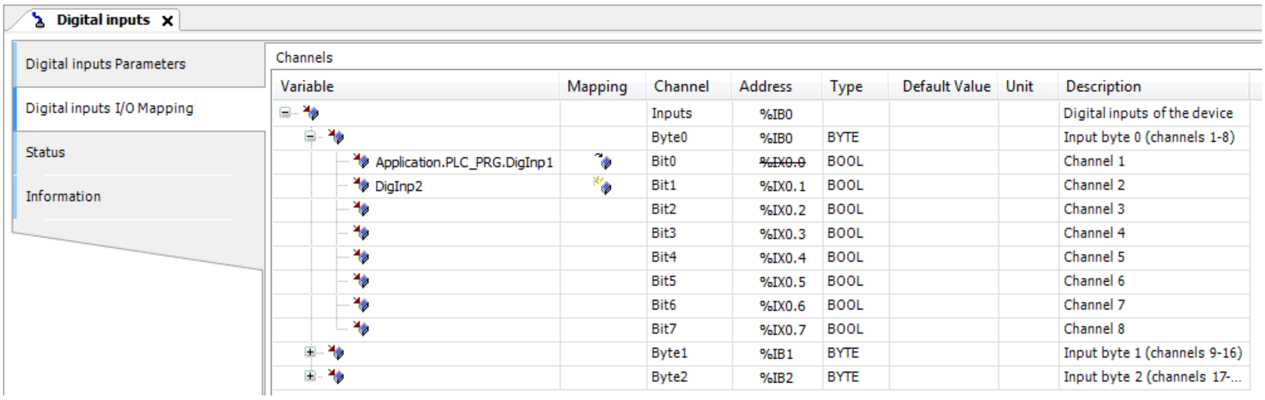
Option	Description
Disabled	I/O variables are updated only if used in a task
Enabled 1	I/O variables are always updated by using bus cycle task if not used in any task
Enabled 2	I/O variables are always updated by using bus cycle task

The above option will be inherited by a specific I/O item if "Always update variables" option is set as "Use parent device setting", from *I/O Mapping* tab.

Configuring Digital Inputs

PLIO03 includes 20 programmable digital inputs.

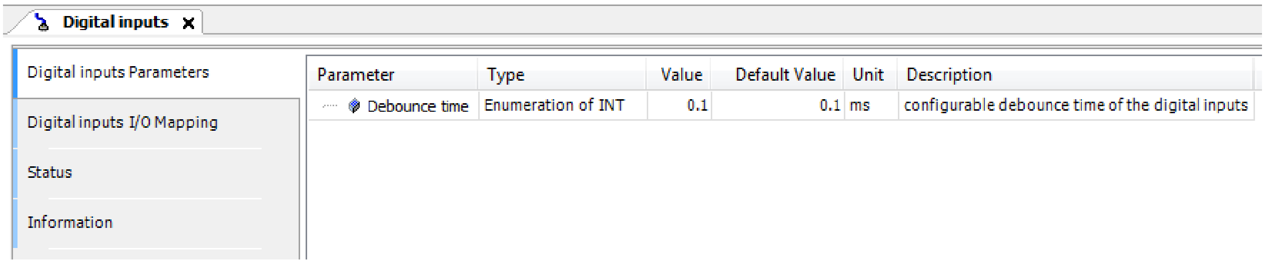
The Mapping configuration of the PLIO03 Digital Inputs is displayed in the Work area by selecting the Digital Inputs I/O Mapping tab after a double click on PLIO03 > Digital Inputs in the project tree.



Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Inputs	%IB0				Digital inputs of the device
		Byte0	%IB0	BYTE			Input byte 0 (channels 1-8)
Application.PLC_PRG.DigInp1		Bit0	~~%IX0.0~~	BOOL			Channel 1
DigInp2		Bit1	%IX0.1	BOOL			Channel 2
		Bit2	%IX0.2	BOOL			Channel 3
		Bit3	%IX0.3	BOOL			Channel 4
		Bit4	%IX0.4	BOOL			Channel 5
		Bit5	%IX0.5	BOOL			Channel 6
		Bit6	%IX0.6	BOOL			Channel 7
		Bit7	%IX0.7	BOOL			Channel 8
		Byte1	%IB1	BYTE			Input byte 1 (channels 9-16)
		Byte2	%IB2	BYTE			Input byte 2 (channels 17-...

Debounce time parameter of the PLIO03 Digital Inputs can be set from Digital Inputs Parameters tab.



Digital inputs

- Digital inputs Parameters
- Digital inputs I/O Mapping
- Status
- Information

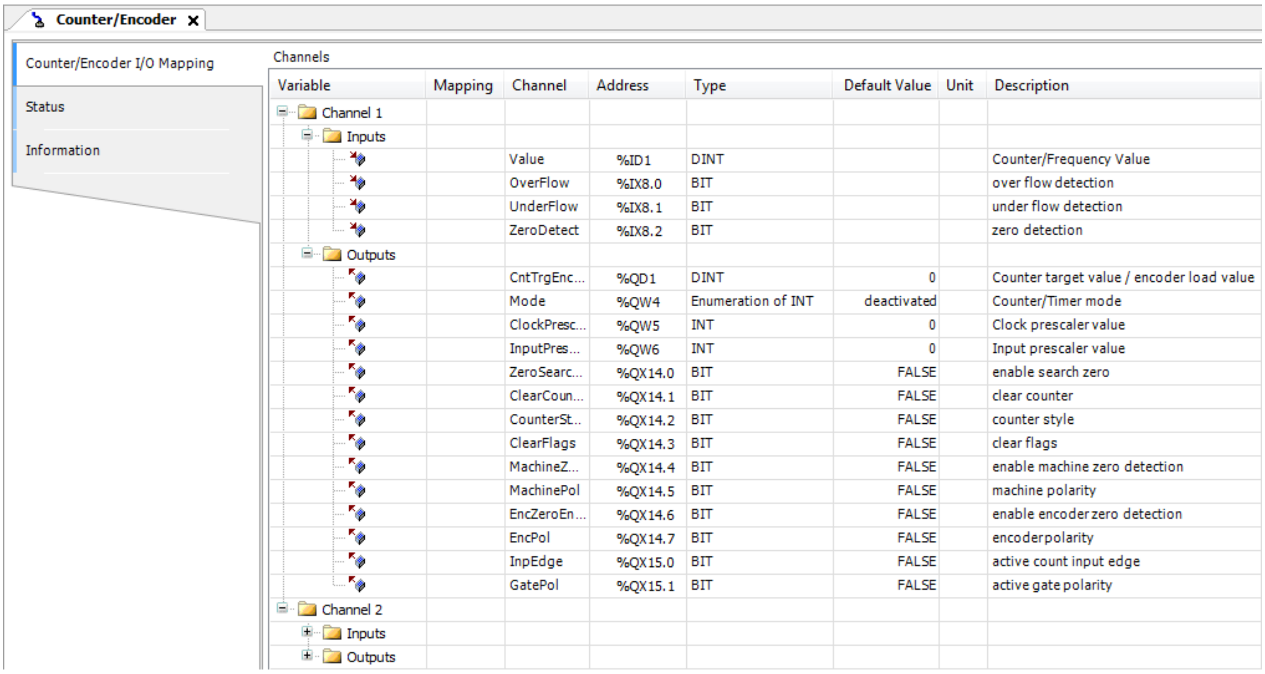
Parameter	Type	Value	Default Value	Unit	Description
Debounce time	Enumeration of INT	0.1	0.1	ms	configurable debounce time of the digital inputs

Counter/Encoder Inputs

PLIO03 includes 2 programmable Counter/Encoder channels.

Each Counter/Encoder channel is associated to a group of 4 digital inputs. When a channel is enabled, the associated digital inputs cannot be used as normal inputs. In case all the 2 Counter/Encoder channels are enabled 12 digital inputs are still available for normal operation. Please refer to PLIO03 hardware manual for detailed specifications and wiring diagrams.

Configuration of Counter/Encoder inputs can be done by selecting the Counter/Encoder I/O Mapping tab after a double click on PLIO03 > Counter/Encoder in the project tree. The following figure shows the list of PLIO03 Counter/Encoder parameters as it appears in the CODESYS software.



Counter/Encoder

- Counter/Encoder I/O Mapping
- Status
- Information

Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
Channel 1							
Inputs							
		Value	%ID1	DINT			Counter/Frequency Value
		OverFlow	%IX8.0	BIT			over flow detection
		UnderFlow	%IX8.1	BIT			under flow detection
		ZeroDetect	%IX8.2	BIT			zero detection
Outputs							
		CntTrgEnc...	%QD1	DINT	0		Counter target value / encoder load value
		Mode	%QW4	Enumeration of INT	deactivated		Counter/Timer mode
		ClockPresc...	%QW5	INT	0		Clock prescaler value
		InputPres...	%QW6	INT	0		Input prescaler value
		ZeroSearc...	%QX14.0	BIT	FALSE		enable search zero
		ClearCoun...	%QX14.1	BIT	FALSE		clear counter
		CounterSt...	%QX14.2	BIT	FALSE		counter style
		ClearFlags	%QX14.3	BIT	FALSE		clear flags
		MachineZ...	%QX14.4	BIT	FALSE		enable machine zero detection
		MachinePol	%QX14.5	BIT	FALSE		machine polarity
		EncZeroEn...	%QX14.6	BIT	FALSE		enable encoder zero detection
		EncPol	%QX14.7	BIT	FALSE		encoderpolarity
		InpEdge	%QX15.0	BIT	FALSE		active count input edge
		GatePol	%QX15.1	BIT	FALSE		active gate polarity
Channel 2							
Inputs							
Outputs							

The input signals associated to each Counter/Encoder channel have the following meaning:

Name	Data Type	Description
Value	DINT	Counter/frequency value
UnderFlow	BOOL	Underflow flag
OverFlow	BOOL	Overflow flag
ZeroDetect	BOOL	Zero found flag

Board parameters have different options that can be selected at runtime using the PLC program.

Description of parameters is given in the next table.

Name	Data Type	Value	Description
CntTrgEncLd	DINT		Target value in Counter mode
Mode	INT	0	
		1	Quadrature incremental encoder counter
		2	Normal Counting
		3	Gated Counting
		4	Frequency Measurement
ClockPrescaler	INT		Clock frequency prescaler
InputPrescaler	INT		Input frequency prescaler
ZeroSearchEnable	BOOL		Enable zero search when in encoder mode. When true the counter resets to zero. ZeroFoundF reports when zero has been reached.
ClearCounter	BOOL		Reset counter value when in counter mode
CounterStyle	BOOL		Valid in counter mode, when true the count value resets when the counter reaches the target value
ClearFlags	BOOL		Reset overflow and underflow flags
MachineZeroEnable	BOOL	false	Corresponding terminal block is used as normal digital input
		true	Corresponding terminal block is used as Machine Zero Input
MachinePol	BOOL	false	Means Input is active LOW
		true	Means Input is active HIGH
EncZeroEnable	BOOL	false	Corresponding terminal block is used as normal digital input
		true	Corresponding terminal block is used as Encoder Zero Input
EncPol	BOOL	false	Means Input is active LOW
		true	Means Input is active HIGH
InpEdge	BOOL	false	Active count input edge negative
		true	Input edge positive
GatePol	BOOL	false	Defines signal level for Gate LOW
		true	Signal level for Gate HIGH

When the input is configured for use with an incremental encoder, then the Encoder Zero Search procedure is usually performed at power-up. PLC program should start this procedure enabling the Zero Search using the parameterZeroSearchE for the selected channel.

Frequency measurement (Mode 4) is based on the following formula:

$$\text{Frequency}=\frac{\text{Internal Clock Frequency}\cdot\text{Number of counts}\cdot(\text{InputScaler}+1)}{\text{Number of counts Internal Clock}\cdot(\text{ClockPresc}+1)}$$

The I/O driver calculates the frequency that is returned in the input signal Value.

One frequency measurement cycle terminates when there is at least one input count and at least 65536 counts in the internal clock.

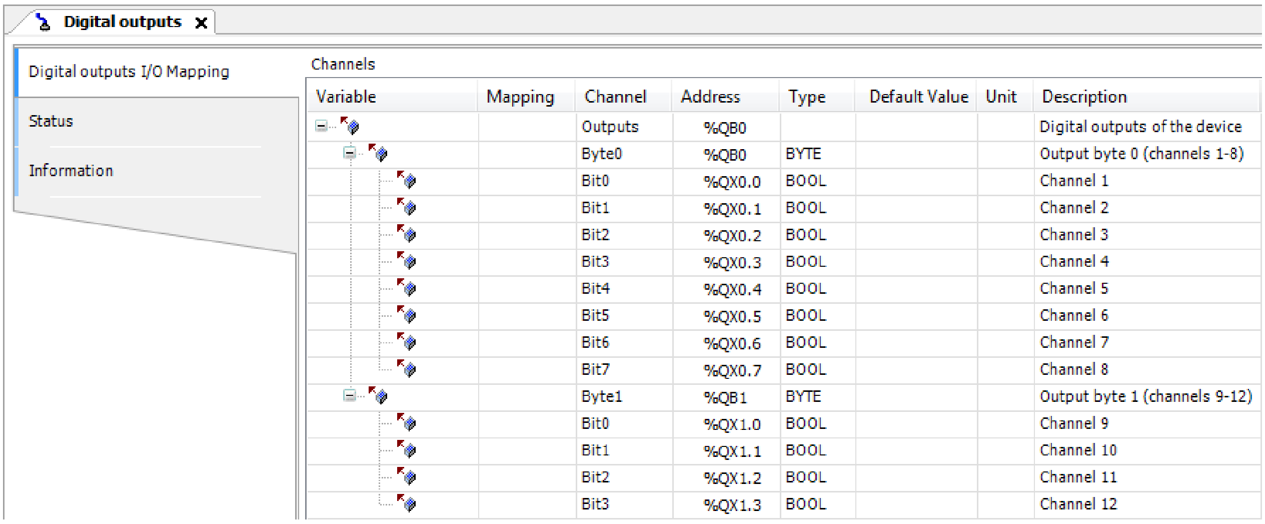
As an example, to cover the frequency range 1Hz to 20KHz you have to set:

Range	ClockPrescaler	InputPrescaler
1Hz to 2Hz	0	1
2Hz to 20KHz	0	0

Configuring Digital Outputs

PLIO03 includes 12 digital outputs.

The Mapping configuration of the PLIO03 Digital Outputs is displayed in the Work area by selecting the Digital Outputs I/O Mapping tab after a double click on PLIO03 > Digital Outputs in the project tree.



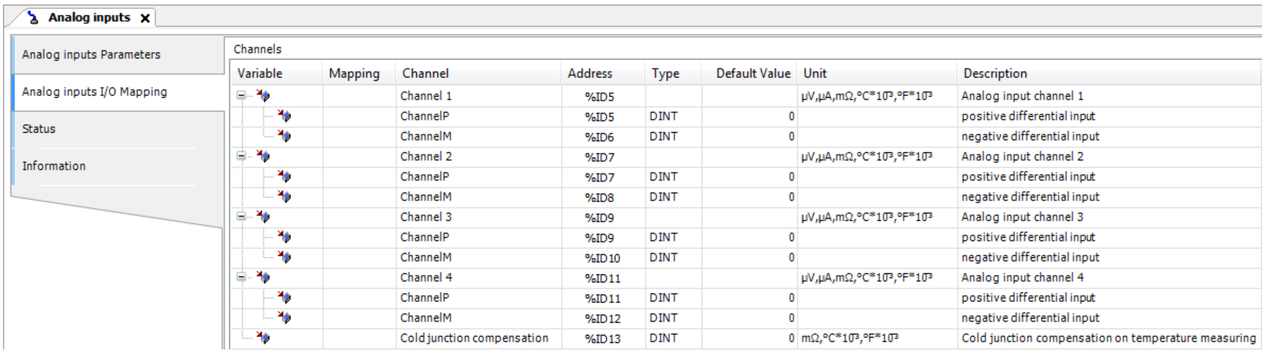
Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Outputs	%QB0				Digital outputs of the device
		Byte0	%QB0	BYTE			Output byte 0 (channels 1-8)
		Bit0	%QX0.0	BOOL			Channel 1
		Bit1	%QX0.1	BOOL			Channel 2
		Bit2	%QX0.2	BOOL			Channel 3
		Bit3	%QX0.3	BOOL			Channel 4
		Bit4	%QX0.4	BOOL			Channel 5
		Bit5	%QX0.5	BOOL			Channel 6
		Bit6	%QX0.6	BOOL			Channel 7
		Bit7	%QX0.7	BOOL			Channel 8
		Byte1	%QB1	BYTE			Output byte 1 (channels 9-12)
		Bit0	%QX1.0	BOOL			Channel 9
		Bit1	%QX1.1	BOOL			Channel 10
		Bit2	%QX1.2	BOOL			Channel 11
		Bit3	%QX1.3	BOOL			Channel 12

Configuring Analog Inputs

PLIO03 includes 4 differential programmable analog input channels plus a dedicated PT100 channel for cold Junction compensation.

Each channel can be configured as single-ended voltage measurement resulting in having 8 voltage analog inputs.

The Mapping configuration of the PLIO03 Analog Inputs is displayed in the Work area by selecting the Analog Inputs I/O Mapping tab after a double click on PLIO03 > Analog Inputs in the project tree. Up to 9 analog values can be produced by the module depending on the configuration.



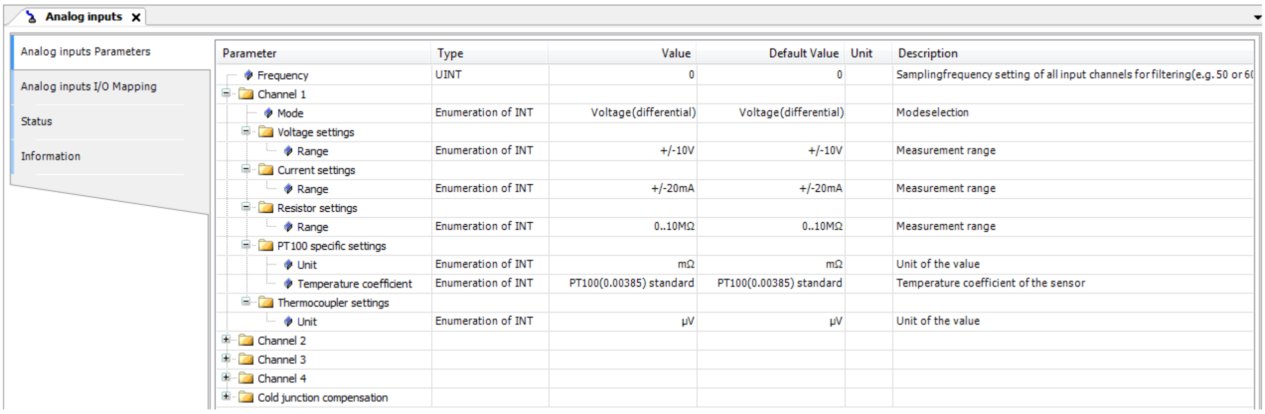
Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Channel 1	%ID5			µV,µA,mΩ,°C*10^3,°F*10^3	Analog input channel 1
		ChannelP	%ID5	DINT	0		positive differential input
		ChannelM	%ID6	DINT	0		negative differential input
		Channel 2	%ID7			µV,µA,mΩ,°C*10^3,°F*10^3	Analog input channel 2
		ChannelP	%ID7	DINT	0		positive differential input
		ChannelM	%ID8	DINT	0		negative differential input
		Channel 3	%ID9			µV,µA,mΩ,°C*10^3,°F*10^3	Analog input channel 3
		ChannelP	%ID9	DINT	0		positive differential input
		ChannelM	%ID10	DINT	0		negative differential input
		Channel 4	%ID11			µV,µA,mΩ,°C*10^3,°F*10^3	Analog input channel 4
		ChannelP	%ID11	DINT	0		positive differential input
		ChannelM	%ID12	DINT	0		negative differential input
		Cold junction compensation	%ID13	DINT	0	mΩ,°C*10^3,°F*10^3	Cold junction compensation on temperature measuring

The following table shows the values provided by the system depending on the Analog Inputs configuration.

	Module Signal	Mode			
		Voltage Differential	**Voltage Single-ended**	**Resistance 2/3 wires**	**Current**
1	AI1 Channel P	Value	Value	Value	Value
2	AI1 Channel M	-	Value	-	-
3	AI2 Channel P	Value	Value	Value	Value
4	AI2 Channel M	-	Value	-	-
5	AI3 Channel P	Value	Value	Value	Value
6	AI3 Channel M	-	Value	-	-
7	AI4 Channel P	Value	Value	Value	Value
8	AI4 Channel M	-	Value	-	-
9	COMP	Resistance	-	-	-

Channel 9 is an input channel dedicated to cold junction compensation of thermocouples. Please refer to the hardware description manual for additional details.

Analog inputs mode can be configured from the apposite table shown in the following figure. This can be reached by selecting the Analog Inputs Parameters tab.



Parameter	Type	Value	Default Value	Unit	Description
Frequency	UINT	0	0		Samplingfrequency setting of all input channels for filtering(e.g. 50 or 6(
Channel 1					
Mode	Enumeration of INT	Voltage(differential)	Voltage(differential)		Modeselection
Voltage settings					
Range	Enumeration of INT	+/-10V	+/-10V		Measurement range
Current settings					
Range	Enumeration of INT	+/-20mA	+/-20mA		Measurement range
Resistor settings					
Range	Enumeration of INT	0..10MΩ	0..10MΩ		Measurement range
PT100 specific settings					
Unit	Enumeration of INT	mΩ	mΩ		Unit of the value
Temperature coefficient	Enumeration of INT	PT100(0.00385) standard	PT100(0.00385) standard		Temperature coefficient of the sensor
Thermocoupler settings					
Unit	Enumeration of INT	µV	µV		Unit of the value
Channel 2					
Channel 3					
Channel 4					
Cold junction compensation					

Configuring Channel Measurement mode

Each channel is independently programmable; six different types of measurements can be executed. The type of measure is defined by the Channel Mode setting, the next table shows the available modes.

Parameter value	Measurement Mode	Unit
0	Voltage (Differential)	µV
1	Voltage (Single Ended)	µV
2	Current (4-20mA)	µA
3	Resistance (2-wire)	mΩ
4	Resistance (3-wire)	mΩ
5	Resistance (4-wire)	mΩ
6	Thermocoupler (Differential)	µV
7	Thermocoupler (Single Ended)	µV
8	Current (0-20mA)	µA
9	PT100 (2-wire)	mΩ
10	PT100 (3-wire)	mΩ
11	PT100 (4-wire)	mΩ

Configuring Channel Full Scale

Measurement range parameter value changes depending on the Channel Mode value set for the Channel, for Channel Mode values from 0 to 5 the Full Scale parameter can be set at programming time. Possible values are reported in the following tables.

Voltage	
Parameter value	**Range**
0	±100mV
1	±500mV
2	±1V
3	±5V
4	±10V
5	0 - 1V
6	0 - 10V

Current	
Parameter value	**Range**
0	±2mA
1	±10mA
2	±20mA

Resistance	
Parameter value	**Range**
0	0 - 80Ω
1	0 - 400Ω
2	0 - 900Ω
3	0 - 8KΩ
4	0 - 10MΩ
5	0 - 1MΩ

For Channel Mode values from 6 to 8 the Range parameter value is set as default by the system.

Parameter value	Measurement Mode	Range
6	Thermocoupler (Differential)	+/- 100mV
7	Thermocoupler (Single Ended)	+/- 100mV
8	Current (0-20mA)	0 - 1V

For Channel Mode values from 9 to 11 the Range value is the one reported in the following table.

Parameter value	Full Scale Value
0	0-157Ω

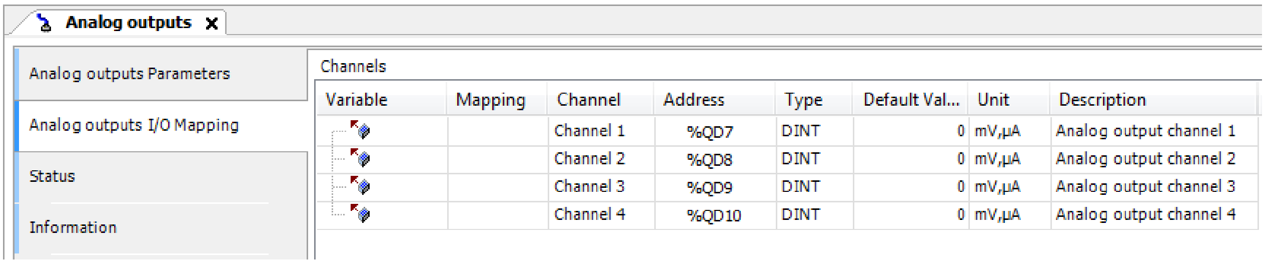
Frequency parameter

The Frequency parameter may be used to synchronize analog measurement to reduce powerline noise. Value 0 means free running measurement, value 50 will take measurements synchronized with powerline at 50Hz. This option is useful with high-gain measurements as in thermocouple channels.

Configuring Analog Outputs

PLIO03 includes 4 programmable analog output channels.

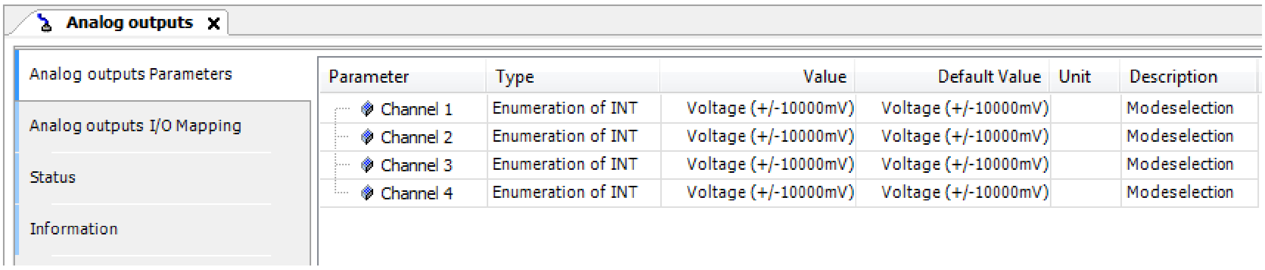
The Mapping configuration of the PLIO03 Analog Outputs is displayed in the Work area by selecting the Analog Outputs I/O Mapping tab after a double click on PLIO03 > Analog Outputs in the project tree.



Channels

Variable	Mapping	Channel	Address	Type	Default Val...	Unit	Description
		Channel 1	%QD7	DINT	0	mV,µA	Analog output channel 1
		Channel 2	%QD8	DINT	0	mV,µA	Analog output channel 2
		Channel 3	%QD9	DINT	0	mV,µA	Analog output channel 3
		Channel 4	%QD10	DINT	0	mV,µA	Analog output channel 4

Analog outputs mode can be configured from the apposite table shown in the following figure this can be reached by selecting the Analog Outputs Parameters tab.



Parameter	Type	Value	Default Value	Unit	Description
Channel 1	Enumeration of INT	Voltage (+/-10000mV)	Voltage (+/-10000mV)		Modeselection
Channel 2	Enumeration of INT	Voltage (+/-10000mV)	Voltage (+/-10000mV)		Modeselection
Channel 3	Enumeration of INT	Voltage (+/-10000mV)	Voltage (+/-10000mV)		Modeselection
Channel 4	Enumeration of INT	Voltage (+/-10000mV)	Voltage (+/-10000mV)		Modeselection

Each of the 4 channels can be independently programmed to be used as voltage or current output. The Channel Value parameter can be configured with the values reported in the following table.

Parameter value	Value
0	Voltage (+/-10000mV)
1	Current (0..20000µA)

The output range is fixed as specified in the table. The value written by the PLC program to the output channels is an integer value between 0 and 10.000 in case of voltage and between 0 and 20.000 in case of current.

PLIO04

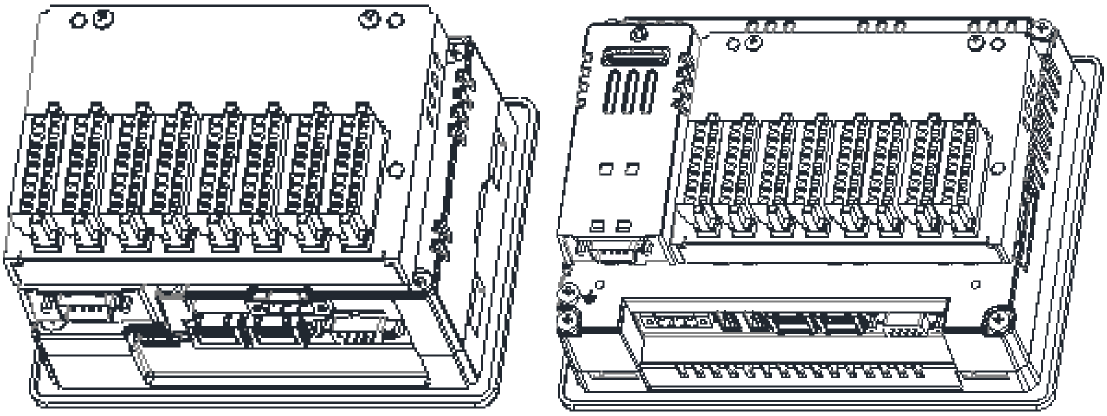
PLIO04 is a multifunction I/O module.

Thanks to the high level of integration and configurability, PLIO04 offers a “one-board solution” for most typical I/O configurations in simple applications.

PLIO04 specifications are the following:

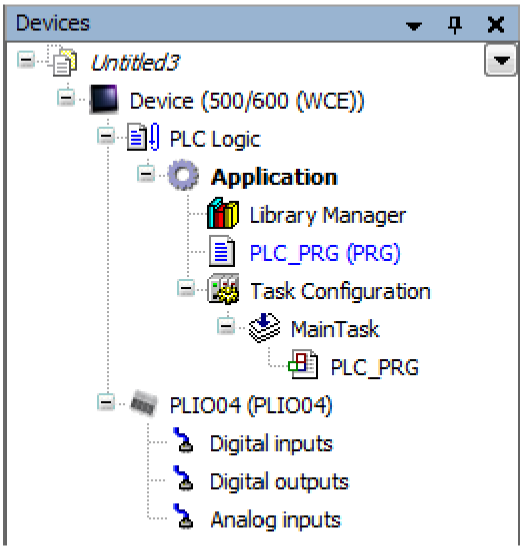
- 10 optically isolated digital inputs
- 12 optically isolated digital outputs SSR
- 4 non-isolated 12 bit analog inputs configurable as 4 differential channels for voltage, resistance or temperature (PT100 and thermocouple) measurement
- 4 non-isolated 12 bit analog inputs for ratiometric voltage only
- 1 dedicated PT100 input for cold junction compensation of thermocouples

The next figure shows a configuration for HMIs, using a communication module and PLIO04 module.



Note: The above pictures shows two types of HMI.
HMIs of the first type require the use of a Bus extender module for the installation of the PLIO04 module.

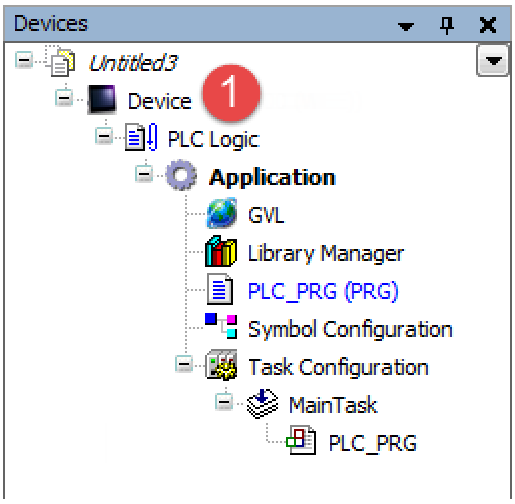
After including the PLIO04 module in the PLC configuration, as shown in the following figure, each I/O type of the module can be configured separately.

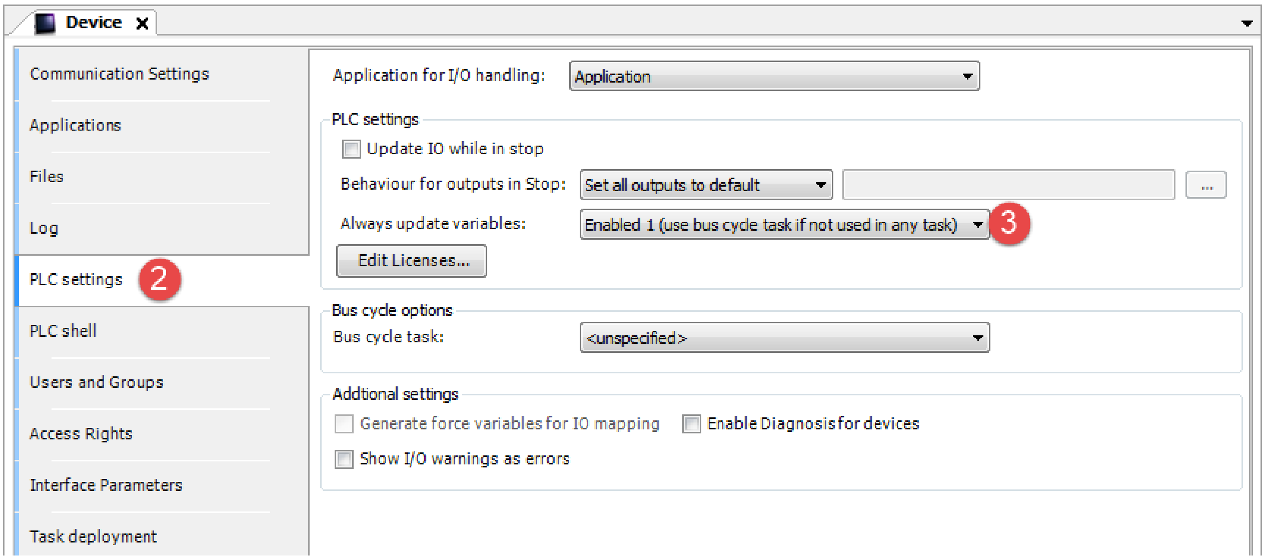


Update I/O variables

To always update I/O variables:

1. Double click on **Device (<*device_name*>)**
2. Select PLC settings tab
3. Set "Enabled 1 (use bus cycle task if not used in any task)" in "Always update variables" option





Application for I/O handling: Application

PLC settings

□ Update IO while in stop

Behaviour for outputs in Stop: Set all outputs to default

Always update variables: Enabled 1 (use bus cycle task if not used in any task)

Bus cycle options

Bus cycle task: <unspecified>

Addtional settings

□ Generate force variables for IO mapping □ Enable Diagnosis for devices

□ Show I/O warnings as errors

Available options are:

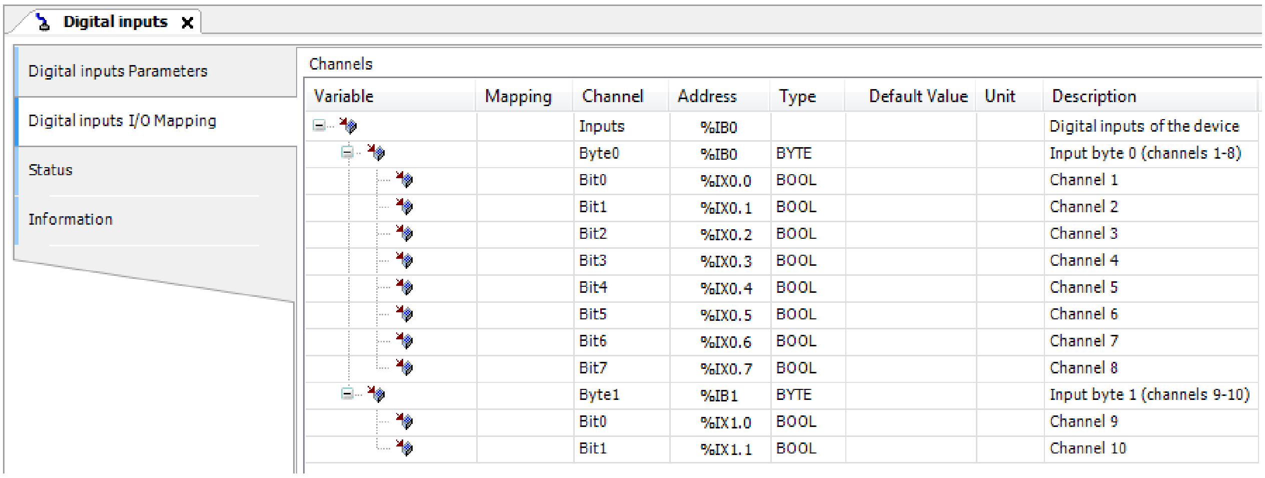
Option	Description
Disabled	I/O variables are updated only if used in a task
Enabled 1	I/O variables are always updated by using bus cycle task if not used in any task
Enabled 2	I/O variables are always updated by using bus cycle task

The above option will be inherited by a specific I/O item if "Always update variables" option is set as "Use parent device setting", from *I/O Mapping* tab.

Configuring Digital Inputs

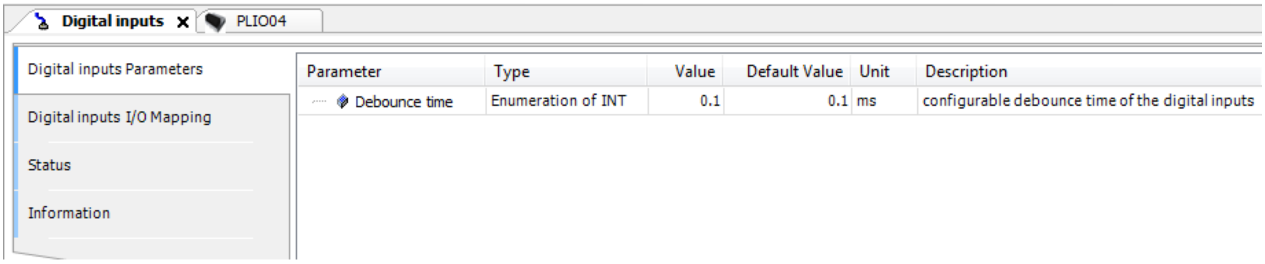
PLIO04 includes 10 digital inputs.

The Mapping configuration of the PLIO04 Digital Inputs is displayed in the Work area by selecting the Digital Inputs I/O Mapping tab after a double click on PLIO04 > Digital Inputs in the project tree.



Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Inputs	%IB0				Digital inputs of the device
		Byte0	%IB0	BYTE			Input byte 0 (channels 1-8)
		Bit0	%IX0.0	BOOL			Channel 1
		Bit1	%IX0.1	BOOL			Channel 2
		Bit2	%IX0.2	BOOL			Channel 3
		Bit3	%IX0.3	BOOL			Channel 4
		Bit4	%IX0.4	BOOL			Channel 5
		Bit5	%IX0.5	BOOL			Channel 6
		Bit6	%IX0.6	BOOL			Channel 7
		Bit7	%IX0.7	BOOL			Channel 8
		Byte1	%IB1	BYTE			Input byte 1 (channels 9-10)
		Bit0	%IX1.0	BOOL			Channel 9
		Bit1	%IX1.1	BOOL			Channel 10

Debounce time parameter of the PLIO04 Digital Inputs can be set by selecting the **Digital Inputs Parameters** tab after a double click on PLIO04 > Digital Inputs in the project tree.

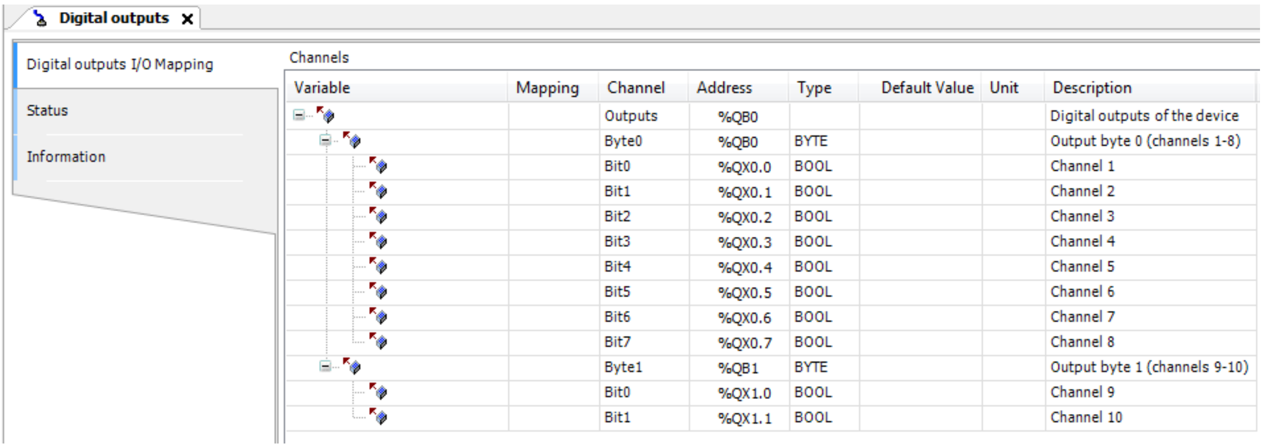


Parameter	Type	Value	Default Value	Unit	Description
Debounce time	Enumeration of INT	0.1	0.1	ms	configurable debounce time of the digital inputs

Configuring Digital Outputs

PLIO04 includes 12 digital outputs.

The Mapping configuration of the PLIO04 Digital Outputs is displayed in the Work area by selecting the Digital Outputs I/O Mapping tab after a double click on PLIO04 > Digital Outputs in the project tree.



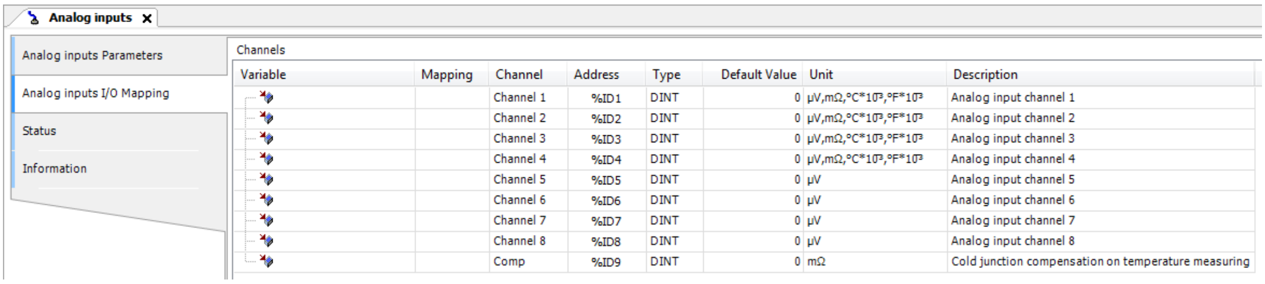
Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Outputs	%QB0				Digital outputs of the device
		Byte0	%QB0	BYTE			Output byte 0 (channels 1-8)
		Bit0	%QX0.0	BOOL			Channel 1
		Bit1	%QX0.1	BOOL			Channel 2
		Bit2	%QX0.2	BOOL			Channel 3
		Bit3	%QX0.3	BOOL			Channel 4
		Bit4	%QX0.4	BOOL			Channel 5
		Bit5	%QX0.5	BOOL			Channel 6
		Bit6	%QX0.6	BOOL			Channel 7
		Bit7	%QX0.7	BOOL			Channel 8
		Byte1	%QB1	BYTE			Output byte 1 (channels 9-10)
		Bit0	%QX1.0	BOOL			Channel 9
		Bit1	%QX1.1	BOOL			Channel 10

Configuring Analog Inputs

PLIO04 includes 4 differential programmable analog input channels, 4 analog input (voltage only) channels plus a dedicated PT100 channel for cold Junction compensation.

First four channels can be configured as single-ended voltage measurement resulting in having 8 voltage analog inputs.

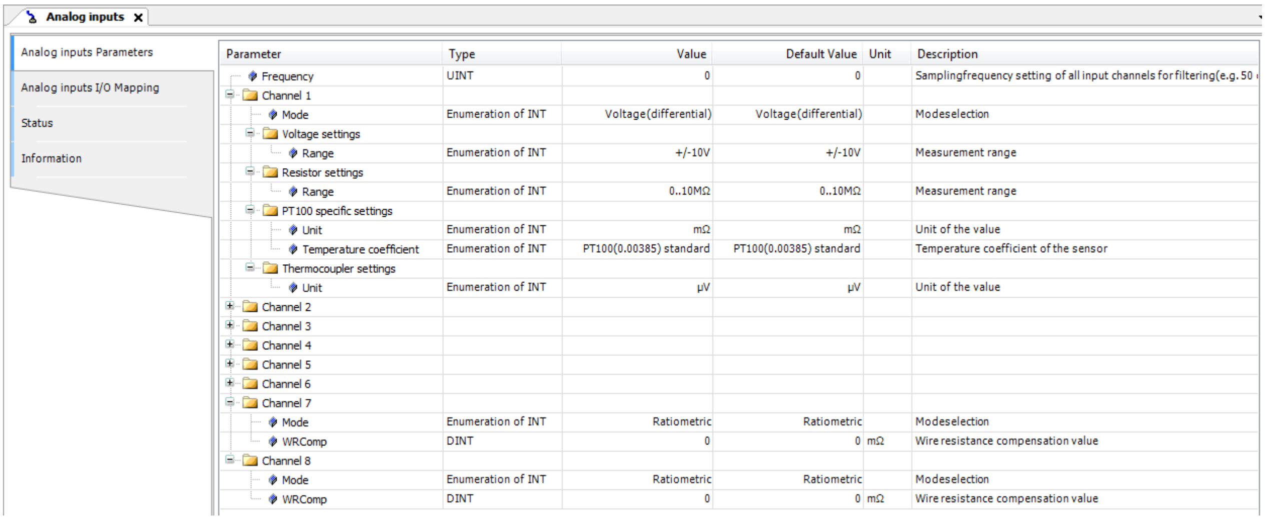
The Mapping configuration of the PLIO04 Analog Inputs is displayed in the Work area by selecting the Analog Inputs I/O Mapping tab after a double click on PLIO04 > Analog Inputs in the project tree. Up to 9 analog values can be produced by the module depending on the configuration.



Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Channel 1	%ID1	DINT	0	µV,mΩ,°C*10³,°F*10³	Analog input channel 1
		Channel 2	%ID2	DINT	0	µV,mΩ,°C*10³,°F*10³	Analog input channel 2
		Channel 3	%ID3	DINT	0	µV,mΩ,°C*10³,°F*10³	Analog input channel 3
		Channel 4	%ID4	DINT	0	µV,mΩ,°C*10³,°F*10³	Analog input channel 4
		Channel 5	%ID5	DINT	0	µV	Analog input channel 5
		Channel 6	%ID6	DINT	0	µV	Analog input channel 6
		Channel 7	%ID7	DINT	0	µV	Analog input channel 7
		Channel 8	%ID8	DINT	0	µV	Analog input channel 8
		Comp	%ID9	DINT	0	mΩ	Cold junction compensation on temperature measuring

Channel 9 is an input channel dedicated to cold junction compensation of thermocouples. Please refer to the hardware description manual for additional details.

PLIO04 Analog Input channels have specific parameters that must to be properly configured according to the operation mode requested for each channel: the operating mode and the full-scale range. Configuration of Analog inputs, shown in the following figure can be reached by selecting the Analog Inputs Configuration tab after a double click on PLIO04 > Analog Input in the project tree.



Parameter	Type	Value	Default Value	Unit	Description
Frequency	UINT	0	0		Samplingfrequency setting of all input channels for filtering(e.g. 50
Channel 1					
Mode	Enumeration of INT	Voltage(differential)	Voltage(differential)		Modeselection
Voltage settings					
Range	Enumeration of INT	+/-10V	+/-10V		Measurement range
Resistor settings					
Range	Enumeration of INT	0..10MΩ	0..10MΩ		Measurement range
PT100 specific settings					
Unit	Enumeration of INT	mΩ	mΩ		Unit of the value
Temperature coefficient	Enumeration of INT	PT100(0.00385) standard	PT100(0.00385) standard		Temperature coefficient of the sensor
Thermocoupler settings					
Unit	Enumeration of INT	µV	µV		Unit of the value
Channel 2					
Channel 3					
Channel 4					
Channel 5					
Channel 6					
Channel 7					
Mode	Enumeration of INT	Ratiometric	Ratiometric		Modeselection
WRComp	DINT	0	0	mΩ	Wire resistance compensation value
Channel 8					
Mode	Enumeration of INT	Ratiometric	Ratiometric		Modeselection
WRComp	DINT	0	0	mΩ	Wire resistance compensation value

Configuring Channel 1-4

Mode

Channels 1-4 are independently programmable; four different types of measurements can be executed. The type of measure is defined by the Channel Mode setting, the next table shows the values that can be configured.

Parameter value	Measurement Mode	Unit
0	Voltage (Differential)	µV
1	Resistance (2-wire)	mΩ
2	Resistance (3-wire)	mΩ

Parameter value	Measurement Mode	Unit
3	Resistance (4-wire)	mΩ
4	Thermocoupler (Differential)	µV
5	PT100 (2-wire)	mΩ
6	PT100 (3-wire)	mΩ
7	PT100 (4-wire)	mΩ

Measurement range parameter value changes depending on the Channel Mode value set for the Channel, for Channel Mode values from 0 to 3 the Full Scale parameter can be set at programming time. Possible values are reported in the following tables.

Voltage	
Parameter value	**Range**
0	±100mV
1	±500mV
2	±1V
3	±5V
4	±10V
5	0 - 1V
6	0 - 10V

Resistance	
Parameter value	**Range**
0	0 - 80Ω
1	0 - 400Ω
2	0 - 900Ω
3	0 - 8KΩ
4	0 - 10MΩ
5	0 - 1MΩ

For Channel Mode value 4 the Range parameter value is set as default by the system.

Thermocoupler		
Parameter value	**Measurement Mode**	**Range**
0	Thermocoupler (Differential)	+/- 100mV

For Channel Mode values from 5 to 7 the Range value is the one reported in the following table.

PT100	
Parameter value	**Full Scale Value**
0	0-157Ω

Configuring Channel 5-8

Channels 5-8 are not programmable ratiometric channels. The voltage input range is fixed and it depends on a reference voltage (5V).

The **WRComp** parameter is the wire resistance compensation value and it can be set at programming time.

Frequency parameter

The Frequency parameter may be used to synchronize analog measurement to reduce powerline noise. Value 0 means free running measurement, value 50 will take measurements synchronized with powerline at 50Hz. This option is useful with high-gain measurements as in thermocouple channels.

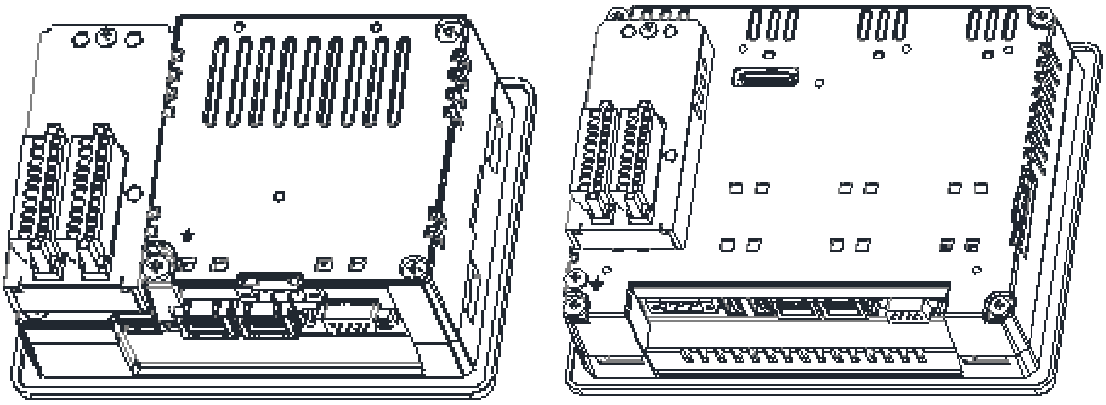
PLIO06

PLIO06 is a compact I/O module. It has been designed for creating simple applications with a limited number of digital I/O signals.

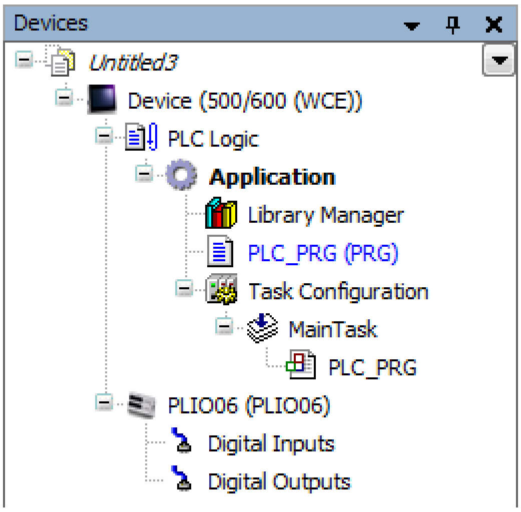
PLIO06 specifications are the following:

- 8 optically isolated digital inputs
- 6 optically isolated digital outputs
- 1 relay output

The next figure shows a configuration for HMIs, using PLIO06 module.



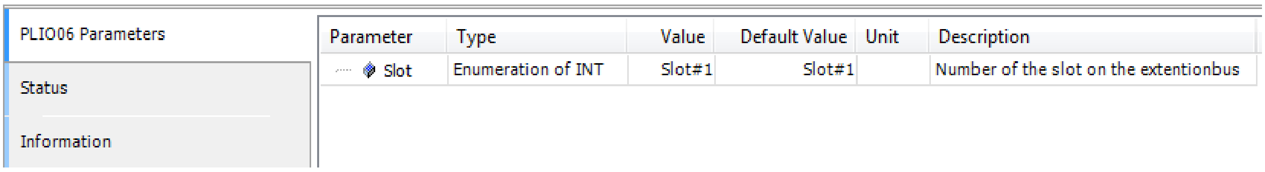
PLIO06 module must be included in the PLC configuration, as shown in the following figure.



Devices

- Untitled3
  - Device (500/600 (WCE))
    - PLC Logic
      - Application
        - Library Manager
        - PLC_PRG (PRG)
        - Task Configuration
          - MainTask
            - PLC_PRG
    - PLIO06 (PLIO06)
      - Digital Inputs
      - Digital Outputs

The contemporary use of 2 PLIO06 modules on the HMI is possible, in this case it is necessary to configure the Slot number parameter for each module. Slot number valid values are 1 to 4. The Slot number configuration is available in the PLIO06 Parameters tab as shown in next figure.



PLIO06 Parameters

Status

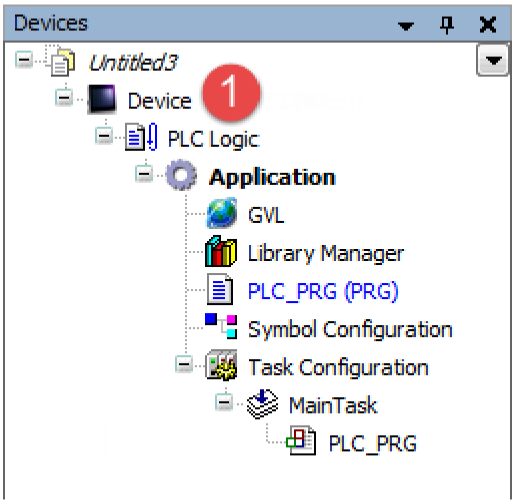
Information

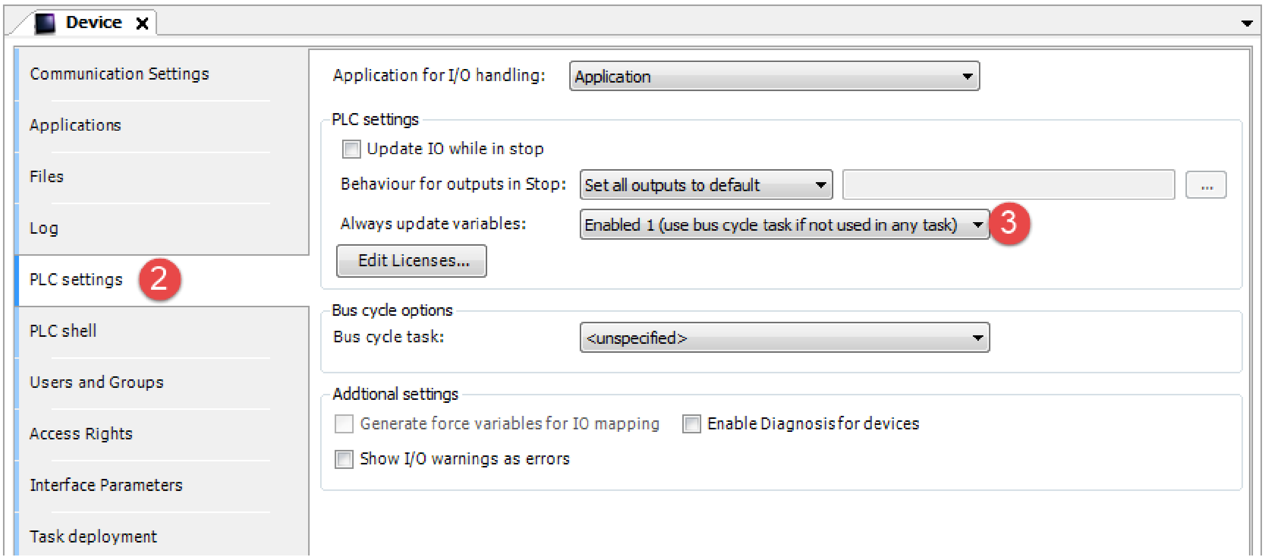
Parameter	Type	Value	Default Value	Unit	Description
Slot	Enumeration of INT	Slot#1	Slot#1		Number of the slot on the extentionbus

Update I/O variables

To always update I/O variables:

1. Double click on **Device (<*device_name*>)**
2. Select PLC settings tab
3. Set "Enabled 1 (use bus cycle task if not used in any task)" in "Always update variables" option





Application for I/O handling: Application

PLC settings

☐ Update IO while in stop

Behaviour for outputs in Stop: Set all outputs to default

Always update variables: Enabled 1 (use bus cycle task if not used in any task)

Bus cycle options

Bus cycle task: <unspecified>

Addtional settings

☐ Generate force variables for IO mapping ☐ Enable Diagnosis for devices

☐ Show I/O warnings as errors

Available options are:

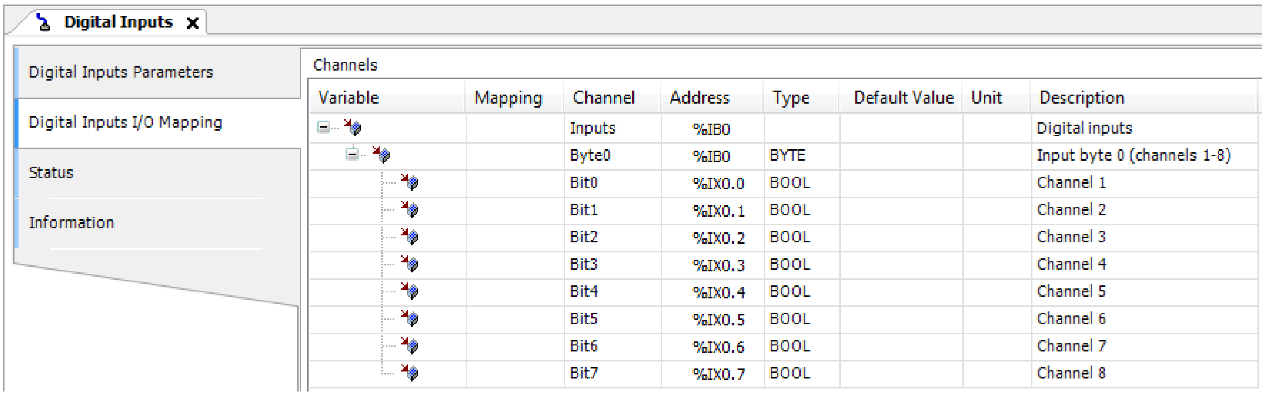
Option	Description
Disabled	I/O variables are updated only if used in a task
Enabled 1	I/O variables are always updated by using bus cycle task if not used in any task
Enabled 2	I/O variables are always updated by using bus cycle task

The above option will be inherited by a specific I/O item if "Always update variables" option is set as "Use parent device setting", from *I/O Mapping* tab.

Configuring Digital Inputs

PLIO06 includes 8 digital inputs.

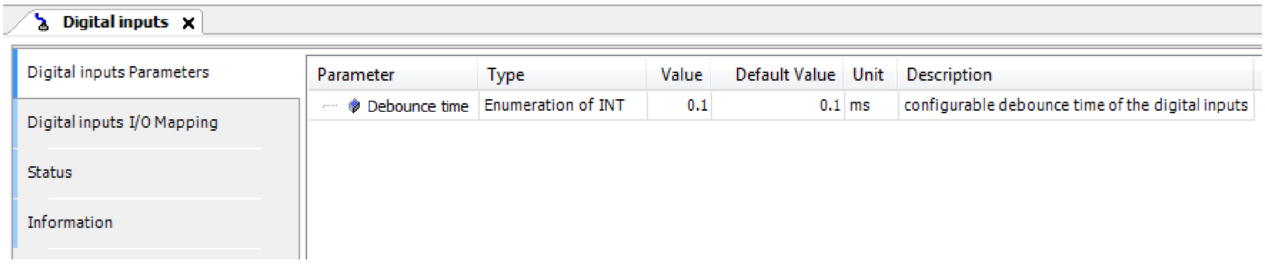
Mapping configuration of the PLIO06 Digital Inputs is displayed in the Work area by selecting the Digital Inputs I/O Mapping tab after a double click on the PLIO06 > Digital Inputs item in the project tree.



Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Inputs	%IB0				Digital inputs
		Byte0	%IB0	BYTE			Input byte 0 (channels 1-8)
		Bit0	%IX0.0	BOOL			Channel 1
		Bit1	%IX0.1	BOOL			Channel 2
		Bit2	%IX0.2	BOOL			Channel 3
		Bit3	%IX0.3	BOOL			Channel 4
		Bit4	%IX0.4	BOOL			Channel 5
		Bit5	%IX0.5	BOOL			Channel 6
		Bit6	%IX0.6	BOOL			Channel 7
		Bit7	%IX0.7	BOOL			Channel 8

Debounce time parameter of the PLIO06 Digital Inputs can be set by selecting the Digital Inputs Parameters tab.

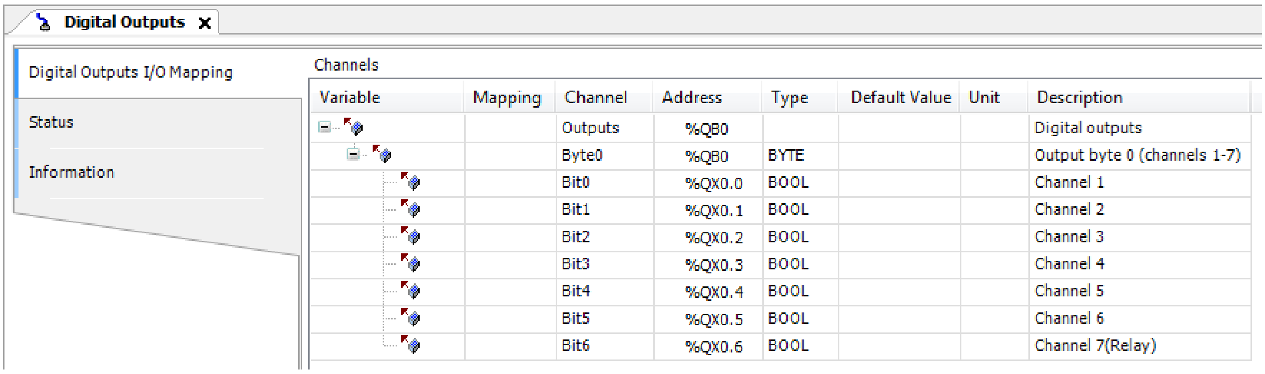


Parameter	Type	Value	Default Value	Unit	Description
Debounce time	Enumeration of INT	0.1	0.1	ms	configurable debounce time of the digital inputs

Configuring Digital Outputs

PLIO06 includes 6 digital outputs plus 1 relay output.

The Mapping configuration of the PLIO06 Digital Outputs is displayed in the Work area by selecting the Digital Outputs I/O Mapping tab after a double click on the PLIO06 > Digital Outputs item in the project tree.



Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Outputs	%QB0				Digital outputs
		Byte0	%QB0	BYTE			Output byte 0 (channels 1-7)
		Bit0	%QX0.0	BOOL			Channel 1
		Bit1	%QX0.1	BOOL			Channel 2
		Bit2	%QX0.2	BOOL			Channel 3
		Bit3	%QX0.3	BOOL			Channel 4
		Bit4	%QX0.4	BOOL			Channel 5
		Bit5	%QX0.5	BOOL			Channel 6
		Bit6	%QX0.6	BOOL			Channel 7(Relay)

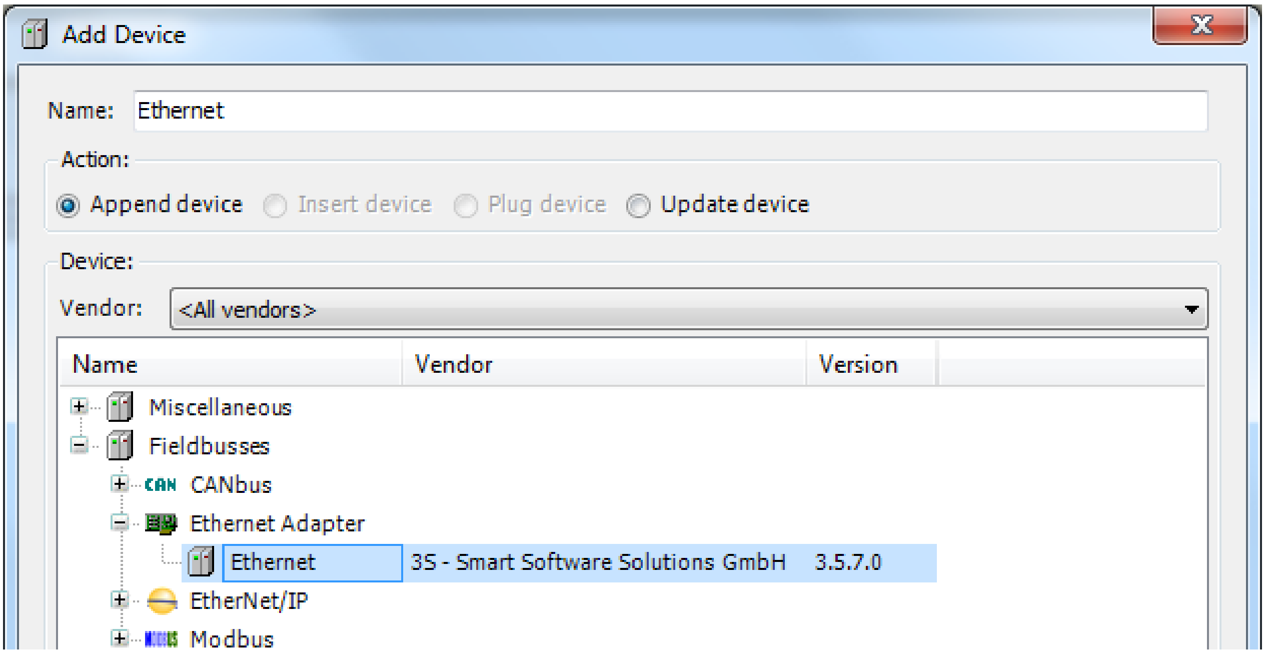
Modbus TCP

CODESYS V3 Runtime can use the built-in Ethernet interface of the operator panels for the distributed Modbus TCP network. No additional hardware is required.

The system can act as Modbus TCP Master or Slave; both configurations are available at the same time.

To add an Ethernet Modbus TCP interface two steps are required.

Right click on **Device (<*device_name*>)** in project tree and select “Add Device”. Ethernet device is located under the *Fieldbusses* > *Ethernet Adapter* category. Choose *Ethernet* device from list and click on “Add Device” to add it to the current PLC configuration.



Add Device

Name: Ethernet

Action:

● Append device ○ Insert device ○ Plug device ○ Update device

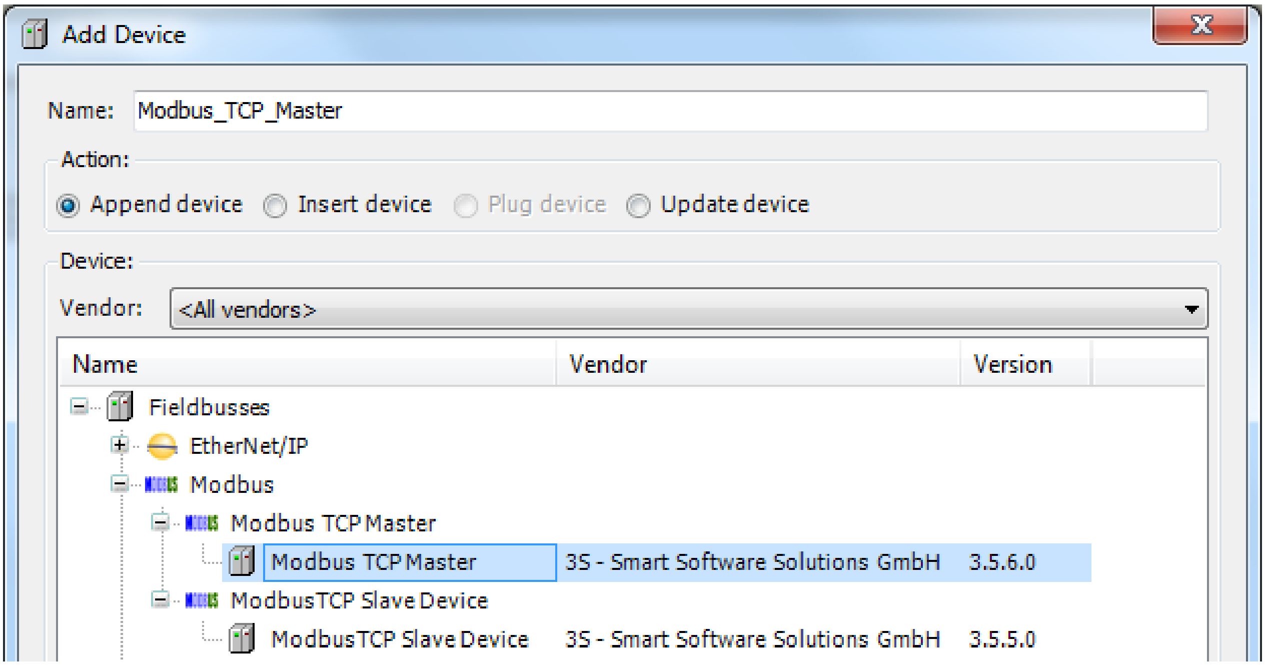
Device:

Vendor: <All vendors>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
CANbus		
Ethernet Adapter		
Ethernet	3S - Smart Software Solutions GmbH	3.5.7.0
EtherNet/IP		
Modbus		

Right click on *Ethernet (Ethernet)* item in the Project tree and select “Add Device”.

Modbus TCP Master and Slave devices are located under the categories *Modbus* > *Modbus TCP Master/Slave*. Select the required device from the list and click on “Add Device” to add it to the current PLC configuration.



Add Device

Name: Modbus_TCP_Master

Action:

● Append device ○ Insert device ○ Plug device ○ Update device

Device:

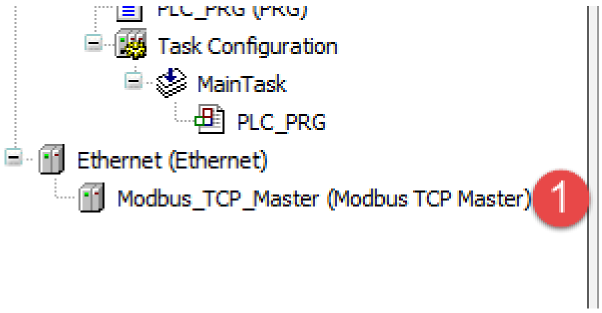
Vendor: <All vendors>

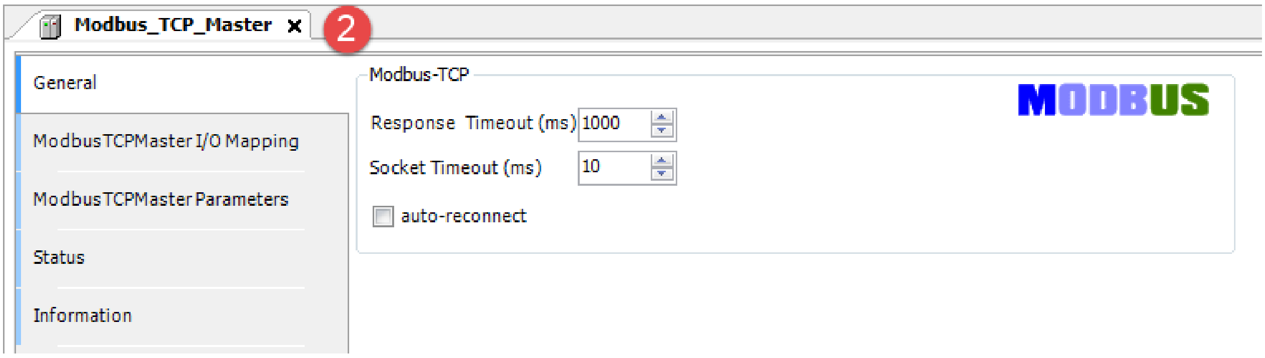
Name	Vendor	Version
Fieldbusses		
EtherNet/IP		
Modbus		
Modbus TCP Master		
Modbus TCP Master	3S - Smart Software Solutions GmbH	3.5.6.0
ModbusTCP Slave Device		
ModbusTCP Slave Device	3S - Smart Software Solutions GmbH	3.5.5.0

Configuring PLC as Modbus TCP Master

Once Modbus TCP Master has been added in project tree double click on *Ethernet* > *Modbus_TCP_Master* (1).

Modbus TCP Master configuration window will be displayed (2).





Response Timeout (ms) 1000

Socket Timeout (ms) 10

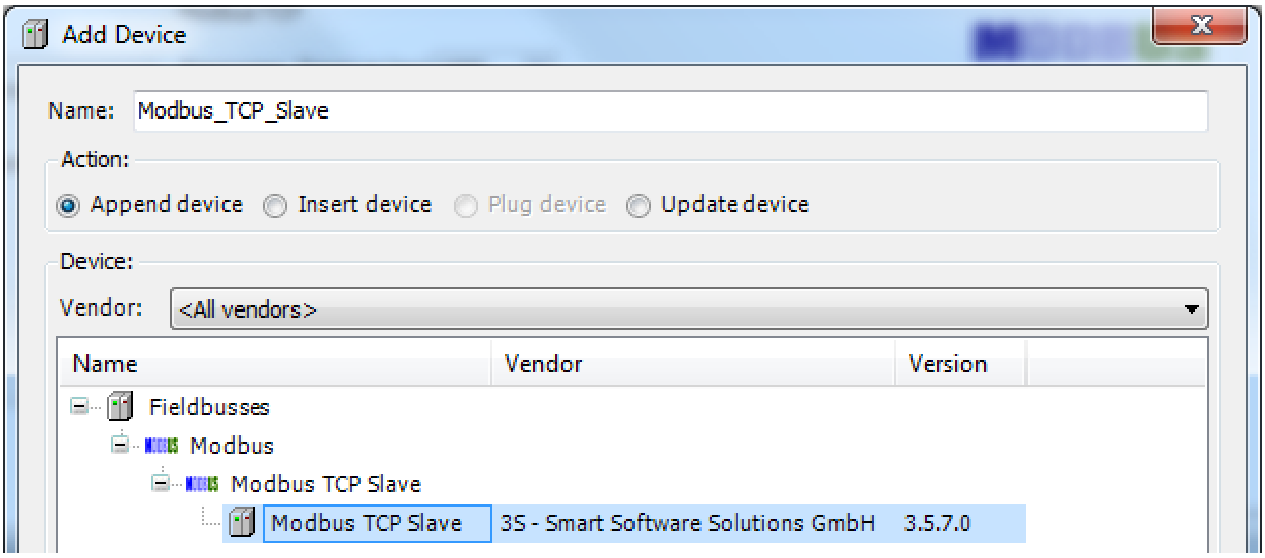
□ auto-reconnect

Available parameters are:

Element	Description
Response Timeout (ms)	Timeout for Modbus slaves reply, given in milliseconds.
Socket Timeout (ms)	Timeout for Socket reply, given in milliseconds.
Auto-reconnect	If set auto-confirm error and re-establish TCP connection.

Add and Configure Remote Modbus TCP Slave Devices

To add a remote Modbus TCP Slave Device, right click on *Ethernet* > *Modbus TCP Master* in the project tree and select “Add Device”. Modbus TCP Slave Devices are located under *Modbus* > *Modbus TCP Slave* category. Choose the device from the list and click on “Add Device” to add it to the current PLC configuration.


Add Device

Name: Modbus_TCP_Slave

Action: ● Append device ○ Insert device ○ Plug device ○ Update device

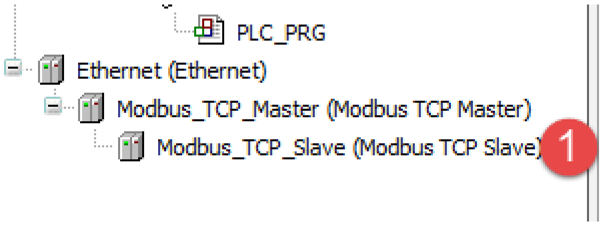
Device:

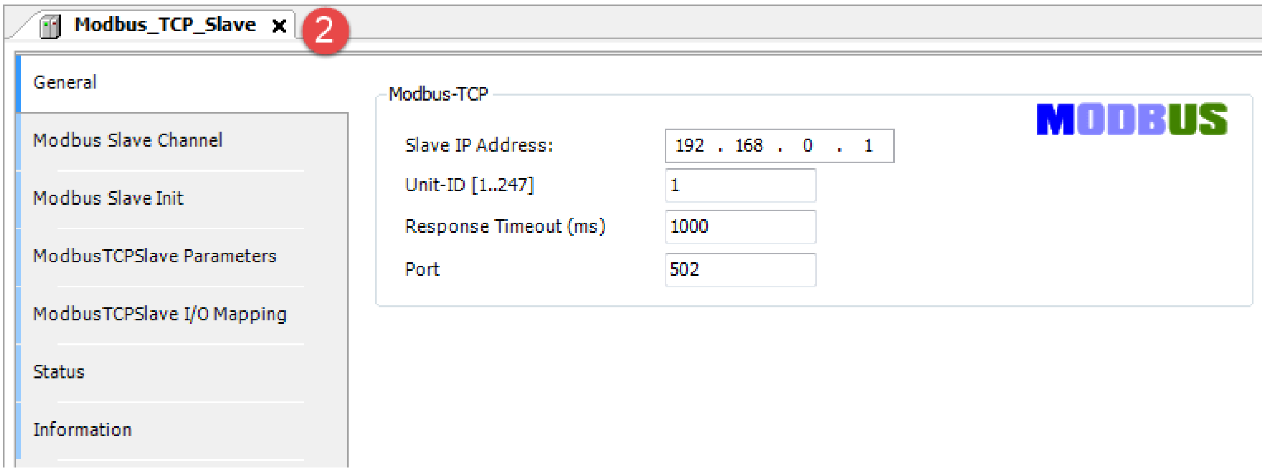
Vendor: <All vendors>

Name	Vendor	Version
Fieldbusses		
Modbus		
Modbus TCP Slave		
Modbus TCP Slave	3S - Smart Software Solutions GmbH	3.5.7.0

Once Modbus TCP Slave has been added in project tree double click on *Ethernet* > *Modbus_TCP_Slave* (1).

Modbus TCP Slave configuration window will be displayed (2).


PLC_PRG
Ethernet (Ethernet)
Modbus_TCP_Master (Modbus TCP Master)
Modbus_TCP_Slave (Modbus TCP Slave)
1



Modbus-TCP

Slave IP Address: 192 . 168 . 0 . 1

Unit-ID [1..247] 1

Response Timeout (ms) 1000

Port 502

Available parameters are:

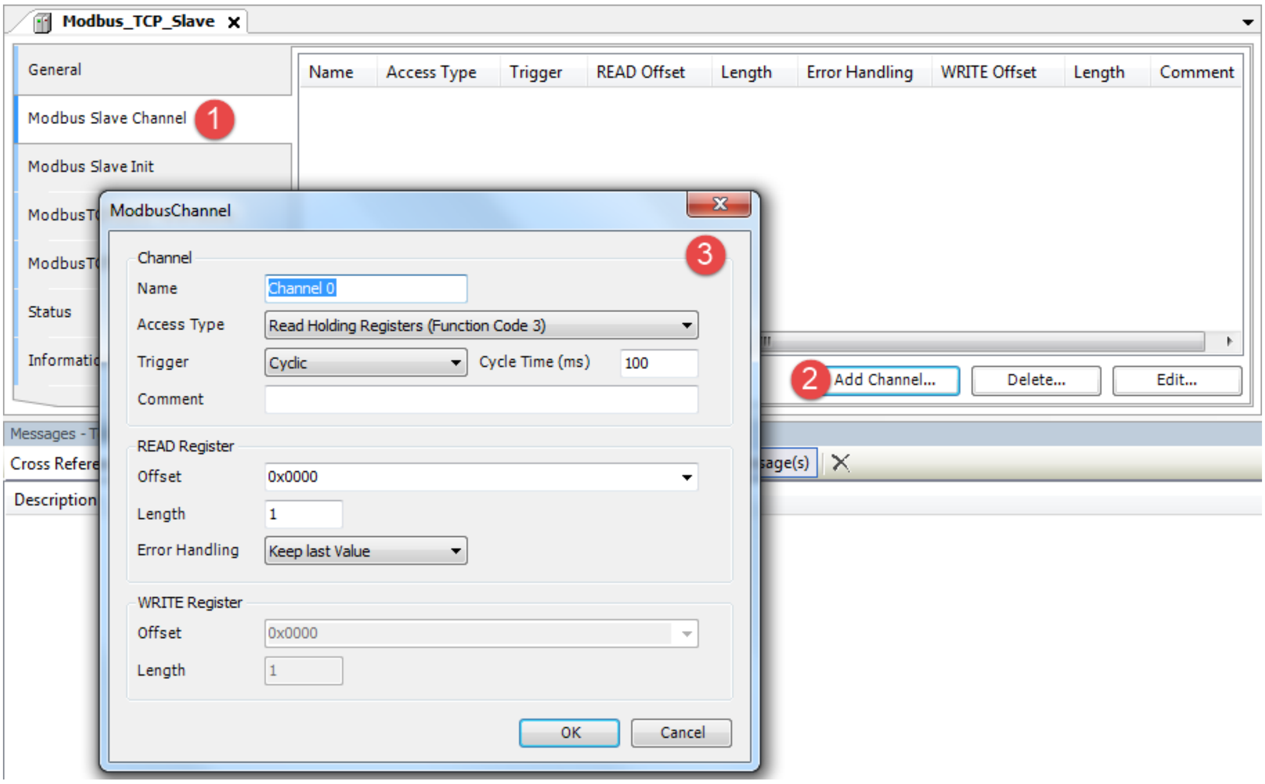
Element	Description
Slave IP Address	IP Address of the Modbus TCP Slave Device
Unit-ID [1..247]	Modbus Node ID of the Modbus TCP Slave Device
Response Timeout (ms)	Timeout for Modbus slaves reply, given in milliseconds.
Port	TCP port used for the communication with the Modbus TCP Slave Device

Configuring Modbus Data Exchange

To configure Modbus data exchange select the Modbus Slave Channel tab (1)

The configuration is based on Channels. For each channel it can be configured a Modbus command that will be sent to the Slave.

To add a new Channel click on Add Channel button (2), as shown in the following figure, the ModbusChannel dialog will be displayed (3).



Channel
Name: Channel 0
Access Type: Read Holding Registers (Function Code 3)
Trigger: Cyclic
Cycle Time (ms): 100
Comment:

READ Register
Offset: 0x0000
Length: 1
Error Handling: Keep last Value

WRITE Register
Offset: 0x0000
Length: 1

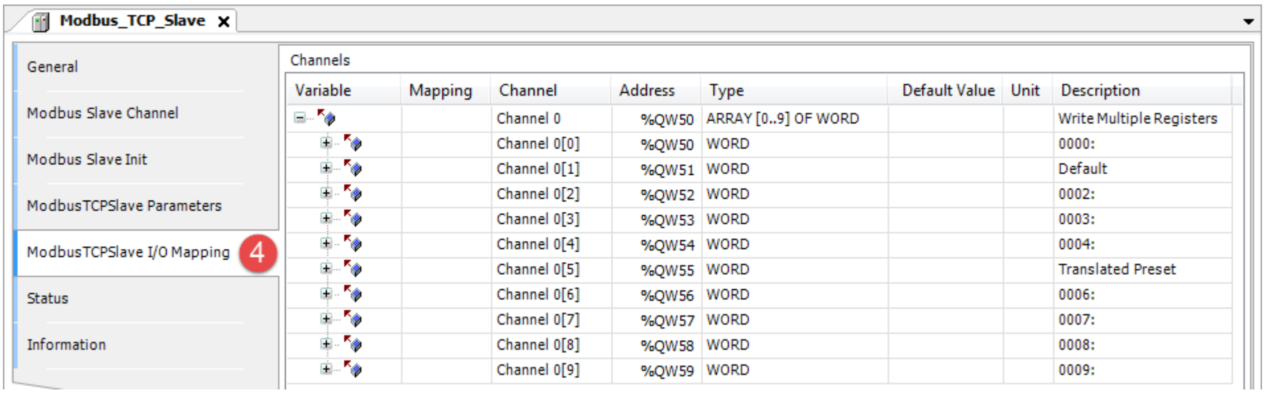
Available parameters are:

Element	Description
Name	Channel Name
Access Type	Selection of the Modbus command
Trigger	Determines if the command execution is cyclic, in this case the Cycle Time must be specified, or Rising Edge, in this case the command is launched on the rising edge event of a boolean variable defined in the Modbus TCP Slave I/O Mapping
Comment	User comment if required
Offset	Determines if the command execution is cyclic, in this case the Cycle Time must be specified, or Rising Edge, in this case the command is launched on the rising edge event of a boolean variable defined in the Modbus TCP Slave I/O Mapping
Length	Number of registers to be Read/Write

Modbus TCP Slave I/O mapping is available by selecting the Modbus TCP Slave I/O Mapping tab (4).

The Mapping shows a list of all Modbus read/write resources in the configured Channels.

In case the configured Channel uses a Rising Edge triggered command the Trigger bit is listed into the Mapping too.

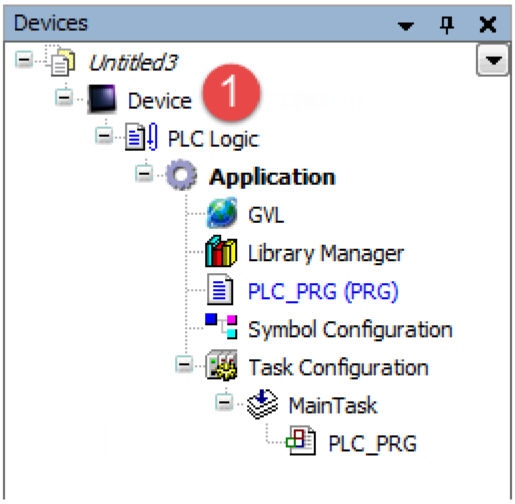


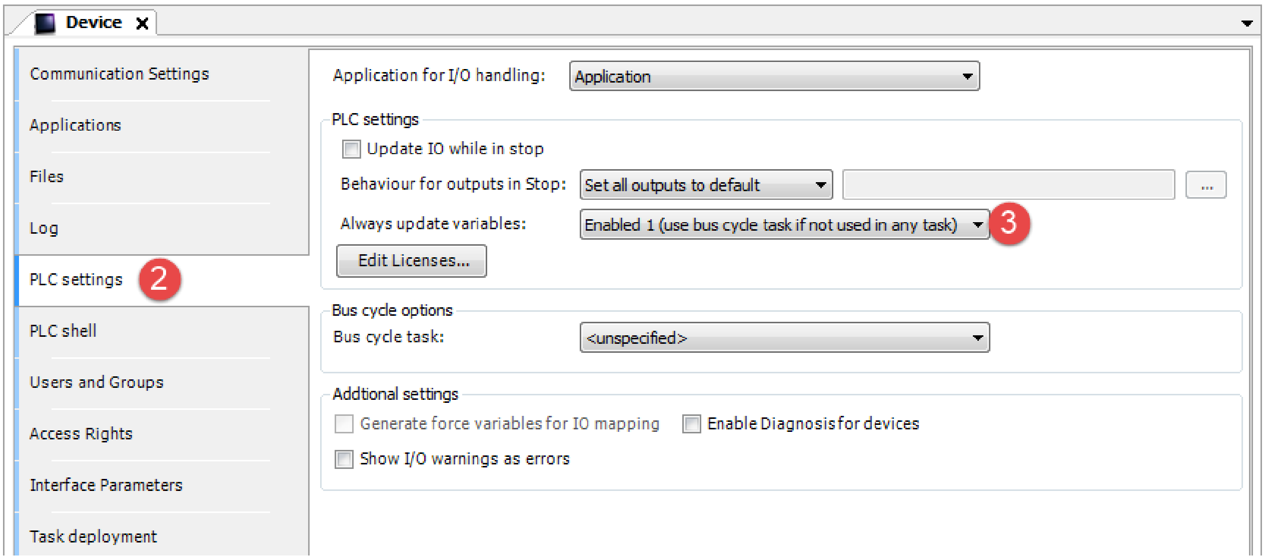
Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Channel 0	%QW50	ARRAY [0..9] OF WORD			Write Multiple Registers
		Channel 0[0]	%QW50	WORD			0000:
		Channel 0[1]	%QW51	WORD			Default
		Channel 0[2]	%QW52	WORD			0002:
		Channel 0[3]	%QW53	WORD			0003:
		Channel 0[4]	%QW54	WORD			0004:
		Channel 0[5]	%QW55	WORD			Translated Preset
		Channel 0[6]	%QW56	WORD			0006:
		Channel 0[7]	%QW57	WORD			0007:
		Channel 0[8]	%QW58	WORD			0008:
		Channel 0[9]	%QW59	WORD			0009:

Update I/O variables

To always update I/O variables:

1. Double click on **Device (<*device_name*>)**
2. Select PLC settings tab
3. Set "Enabled 1 (use bus cycle task if not used in any task)" in "Always update variables" option





Application for I/O handling: Application

PLC settings

□ Update IO while in stop

Behaviour for outputs in Stop: Set all outputs to default

Always update variables: Enabled 1 (use bus cycle task if not used in any task)

Bus cycle options

Bus cycle task: <unspecified>

Addtional settings

□ Generate force variables for IO mapping □ Enable Diagnosis for devices

□ Show I/O warnings as errors

Available options are:

Option	Description
Disabled	I/O variables are updated only if used in a task
Enabled 1	I/O variables are always updated by using bus cycle task if not used in any task
Enabled 2	I/O variables are always updated by using bus cycle task

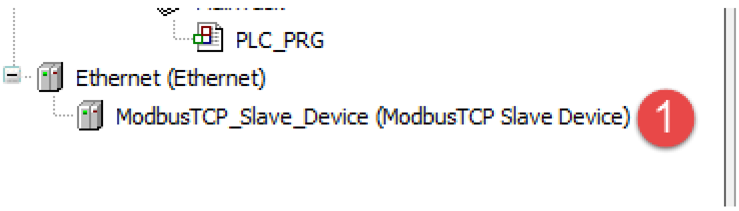
The above option will be inherited by a specific I/O item if "Always update variables" option is set as "Use parent device setting", from *I/O Mapping* tab.

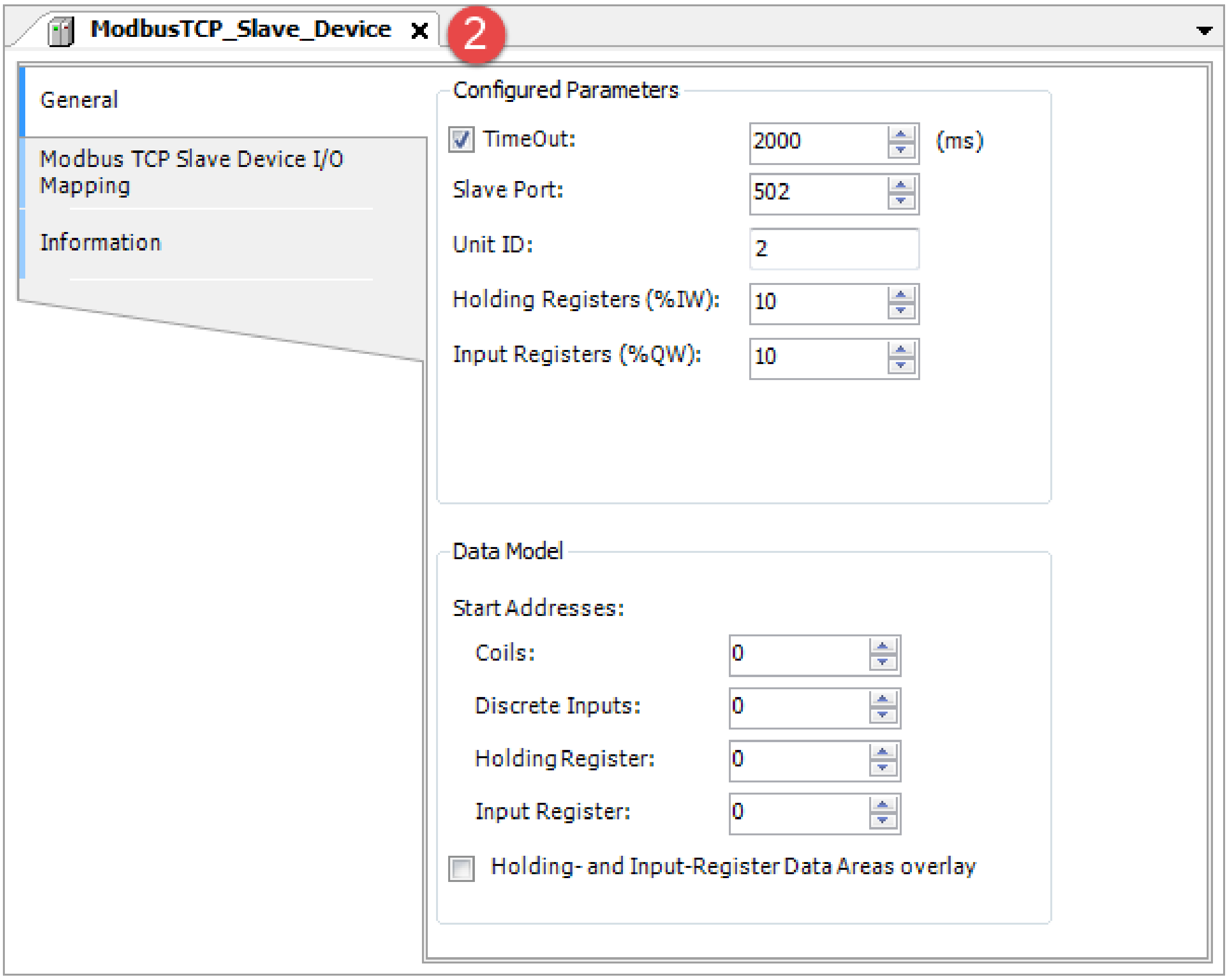
Configuring PLC as Modbus TCP Slave Device

Note: When programmed as explained in this chapter the CODESYS V3 PLC will act as a Modbus TCP Slave device. To configure the device for communication with remote Modbus TCP I/O slaves please refer to *Configuring PLC as Modbus TCP Master* chapter

Once Modbus TCP Slave has been added in project tree double click on *Ethernet* > *Modbus_TCP_Slave_Device* (1).

Modbus TCP Slave configuration window will be displayed (2).





Configured Parameters

☑ TimeOut: 2000 (ms)

Slave Port: 502

Unit ID: 2

Holding Registers (%IW): 10

Input Registers (%QW): 10

Data Model

Start Addresses:

Coils: 0

Discrete Inputs: 0

Holding Register: 0

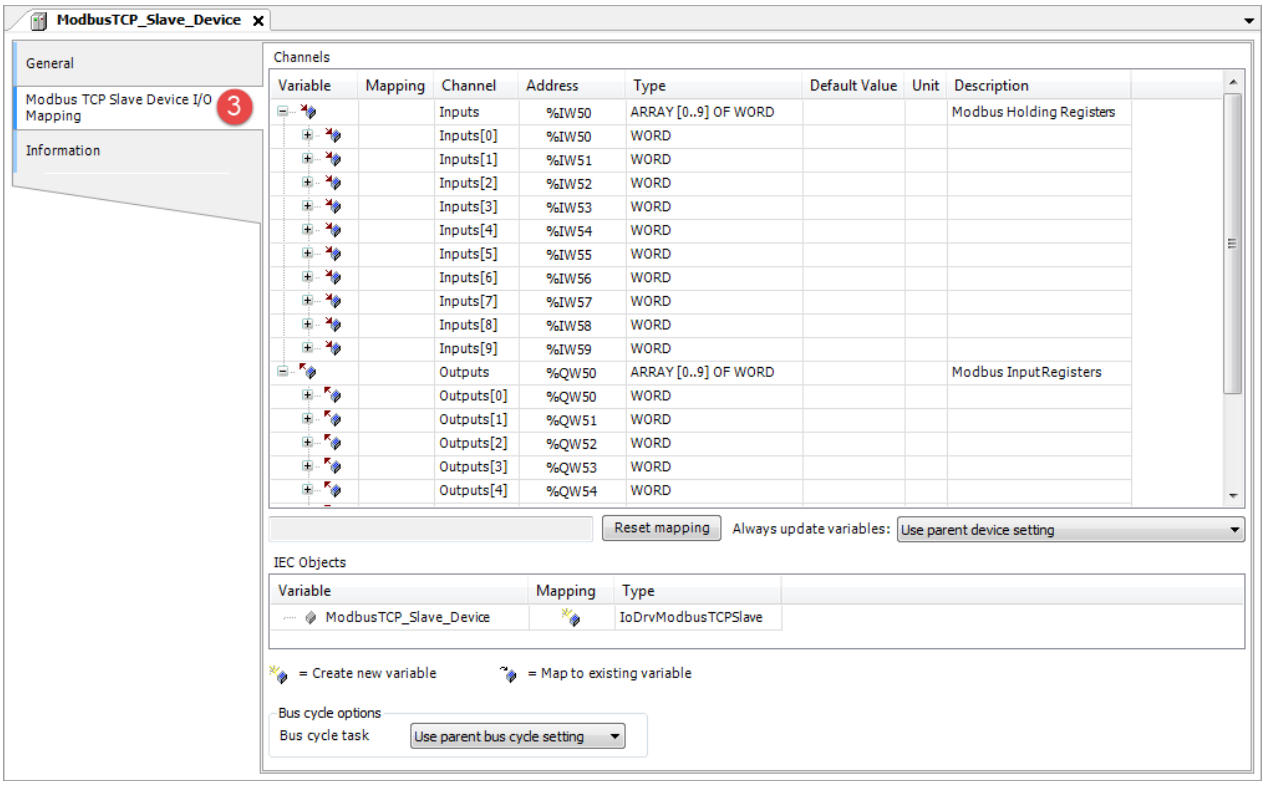
Input Register: 0

□ Holding- and Input-Register Data Areas overlay

Available parameters are:

Element	Description
Timeout	If selected, timeout for Modbus Master queries, given in milliseconds.
Slave port	TCP port used for the communication with the Modbus TCP Slave Device
Unit-ID	Modbus Node ID of the Modbus TCP Slave Device
Holding Registers (%IW)	Number of holding registers available in PLC memory
Input Registers (%QW)	Number of input registers available in PLC memory
Start address	Starting address for Modbus resources in PLC

Modbus TCP Slave mapping is available by selecting the Modbus TCP Slave Device I/O Mapping tab (3).



Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Inputs	%IW50	ARRAY [0..9] OF WORD			Modbus Holding Registers
		Inputs[0]	%IW50	WORD			
		Inputs[1]	%IW51	WORD			
		Inputs[2]	%IW52	WORD			
		Inputs[3]	%IW53	WORD			
		Inputs[4]	%IW54	WORD			
		Inputs[5]	%IW55	WORD			
		Inputs[6]	%IW56	WORD			
		Inputs[7]	%IW57	WORD			
		Inputs[8]	%IW58	WORD			
		Inputs[9]	%IW59	WORD			
		Outputs	%QW50	ARRAY [0..9] OF WORD			Modbus InputRegisters
		Outputs[0]	%QW50	WORD			
		Outputs[1]	%QW51	WORD			
		Outputs[2]	%QW52	WORD			
		Outputs[3]	%QW53	WORD			
		Outputs[4]	%QW54	WORD			

Always update variables: Use parent device setting

IEC Objects

Variable	Mapping	Type
ModbusTCP_Slave_Device		IoDrvModbusTCPSlave

= Create new variable   = Map to existing variable

Bus cycle options

Bus cycle task: Use parent bus cycle setting

Modbus RTU

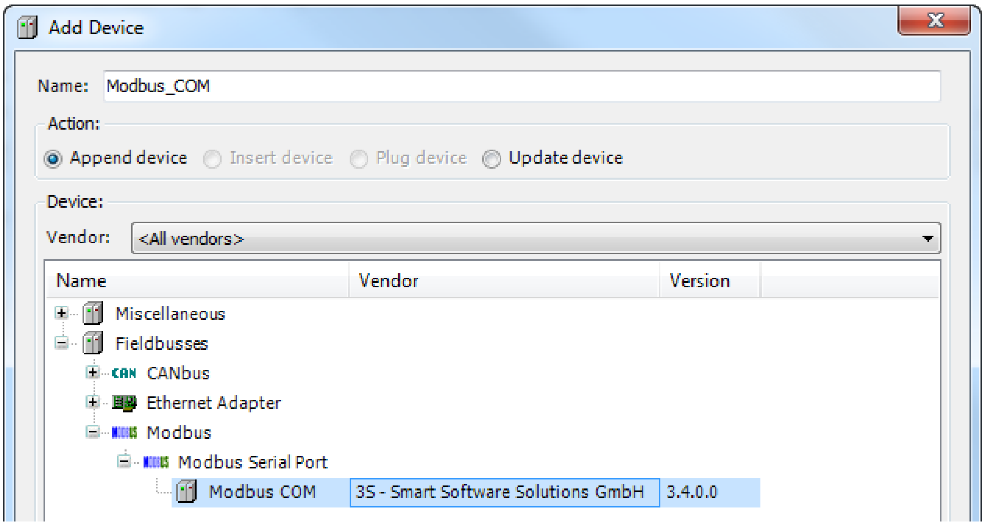
CODESYS V3 Runtime can use the built-in serial interface of the HMI for the distributed Modbus RTU network. One single serial interface is available as built-in option for Exor HMIs. With the use of optional plug-in modules it is possible to have up to 3 serial interfaces on the device.

Note: Availability of optional plug-in modules depends on the HMI model, please check *Control Solutions with HMI and CODESYS V3* chapter to verify availability for your HMI.

The system can act as Modbus RTU Master or Slave.

To add a Modbus RTU interface two consecutive steps are required.

Right click on **Device (<*device_name*>)** in project tree and select “Add Device”. Modbus COM device is located under *Fieldbusses* > *Modbus* > *Modbus Serial port* category. Choose *Modbus COM* device from list and click on “Add Device” to add it to the current PLC configuration.



Add Device

Name: Modbus_COM

Action: ● Append device ○ Insert device ○ Plug device ○ Update device

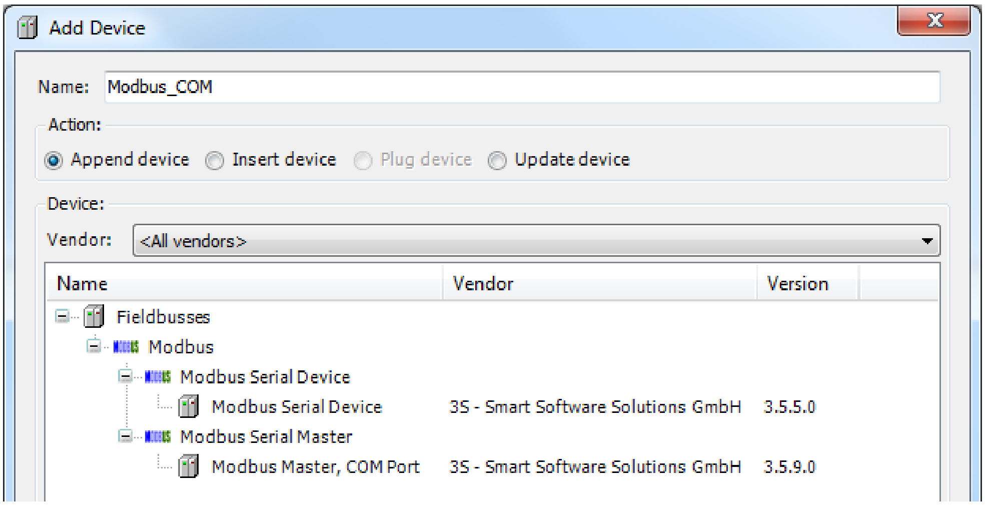
Device:

Vendor: <All vendors>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
CANbus		
Ethernet Adapter		
Modbus		
Modbus Serial Port		
Modbus COM	3S - Smart Software Solutions GmbH	3.4.0.0

Right click on *Modbus COM* item in project tree and select “Add Device”.

Modbus RTU Master and Slave devices are located under *Fieldbusses* > *Modbus* > *Modbus Serial Master/Device* categories. Select the required device from the list and click on “Add Device” to add it to the current PLC configuration.



Add Device

Name: Modbus_COM

Action: ● Append device ○ Insert device ○ Plug device ○ Update device

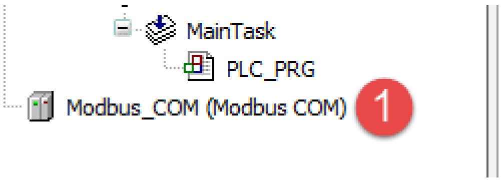
Device:

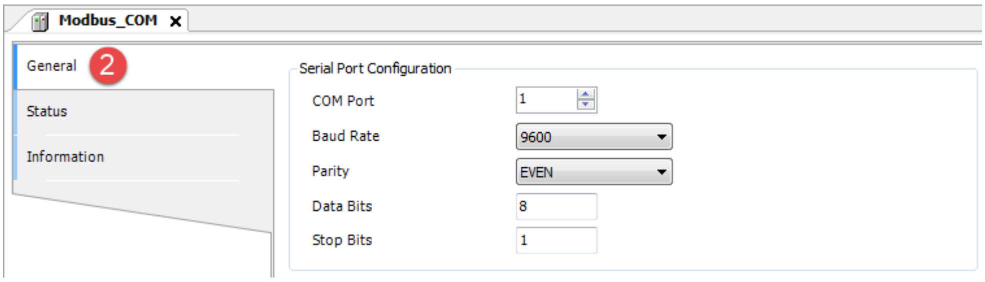
Vendor: <All vendors>

Name	Vendor	Version
Fieldbusses		
Modbus		
Modbus Serial Device		
Modbus Serial Device	3S - Smart Software Solutions GmbH	3.5.5.0
Modbus Serial Master		
Modbus Master, COM Port	3S - Smart Software Solutions GmbH	3.5.9.0

Modbus RTU Serial Port Configuration

To configure Modbus RTU Serial port double click on Modbus COM item (1) and select General tab (2).





Serial Port Configuration

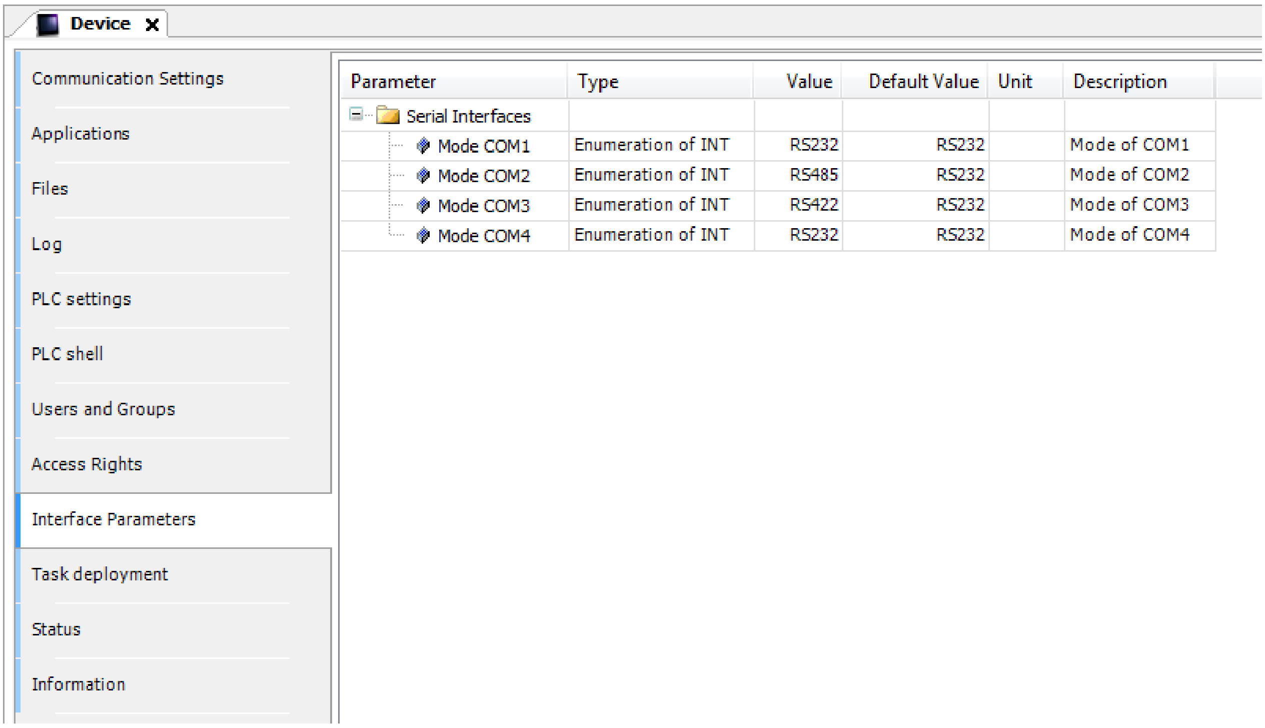
COM Port: 1
Baud Rate: 9600
Parity: EVEN
Data Bits: 8
Stop Bits: 1

Available parameters are:

Element	Description
COM Port	Serial COM Port number. **1** = COM1, for onboard serial port **2** = COM2, for optional serial plugin mounted on Slot#1 or Slot#2 * **3** = COM3, for optional serial plugin mounted on Slot#3 or Slot#4 * * check *Installing Optional Modules > Slot Numbers* for Slot# details
Baud Rate	Serial line parameters
Parity	
Data Bits	
Stop Bits	

To set serial COM mode double click on **Device (<*device_name*>)** and click on Interface Parameters tab, as shown in picture below.

For each port modify *Value* column and choose between RS232, RS485 and RS422.

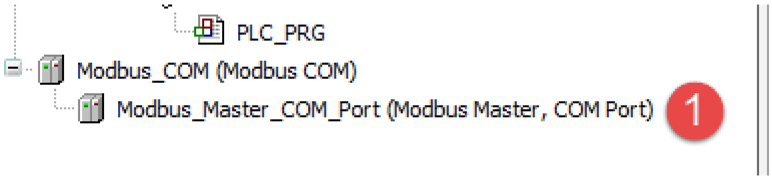


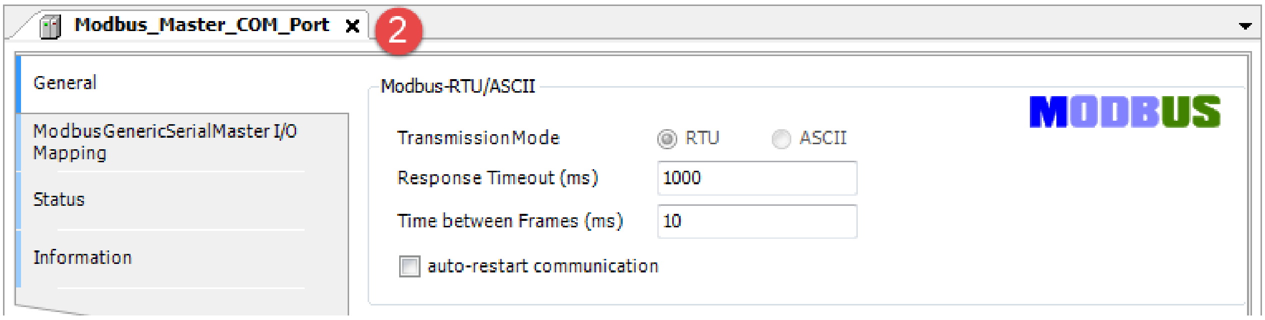
Parameter	Type	Value	Default Value	Unit	Description
Serial Interfaces					
Mode COM1	Enumeration of INT	RS232	RS232		Mode of COM1
Mode COM2	Enumeration of INT	RS485	RS232		Mode of COM2
Mode COM3	Enumeration of INT	RS422	RS232		Mode of COM3
Mode COM4	Enumeration of INT	RS232	RS232		Mode of COM4

Configuring PLC as Modbus RTU Master

Once Modbus Master COM Port as been added in project tee double click on *Modbus_COM> Modbus_Master_COM_Port* (1).

Modbus Master COM Port configuration window will be displayed (2).





Modbus-RTU/ASCII

Transmission Mode ● RTU ○ ASCII

Response Timeout (ms) 1000

Time between Frames (ms) 10

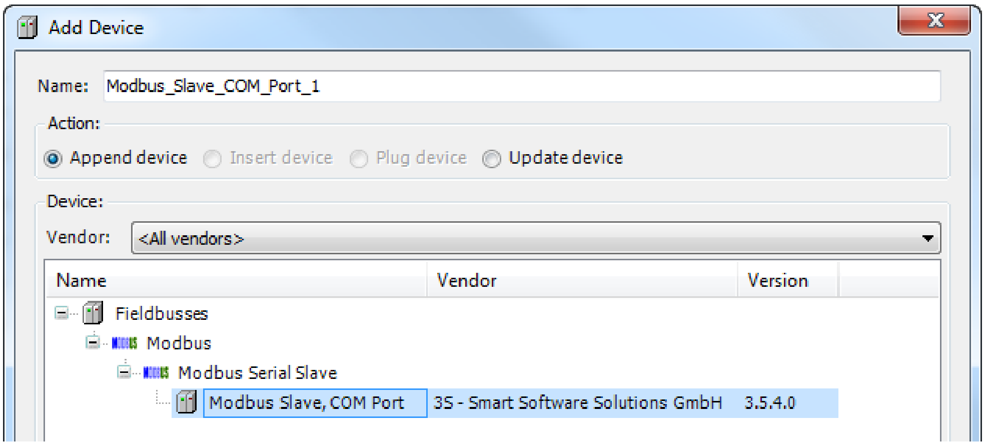
□ auto-restart communication

Available parameters are:

Element	Description
Response Timeout (ms)	Timeout for Modbus slaves reply, given in milliseconds.
Time between frames (ms)	Wait time between Slave reply and next Master query.
Auto-restart communication	If set auto-confirm error and re-establish communication.

Add and Configure Remote Modbus RTU Slave Devices

To add a remote Modbus RTU Slave Device, right click on *Modbus_COM* > *Modbus_Master_COM_Port* in the project tree and select “Add Device”. Modbus RTU Slave Devices are located under *Fieldbusses* > *Modbus* > *Modbus Serial Slave* category. Choose the device from the list and click on “Add Device” to add it to the current PLC configuration.



Add Device

Name: Modbus_Slave_COM_Port_1

Action: ● Append device ○ Insert device ○ Plug device ○ Update device

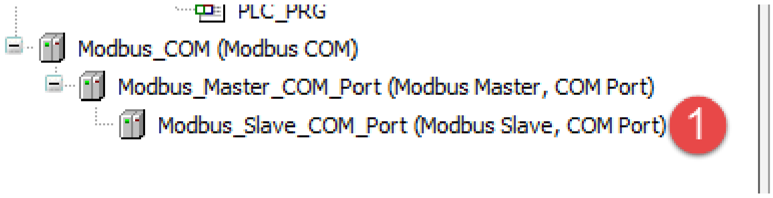
Device:

Vendor: <All vendors>

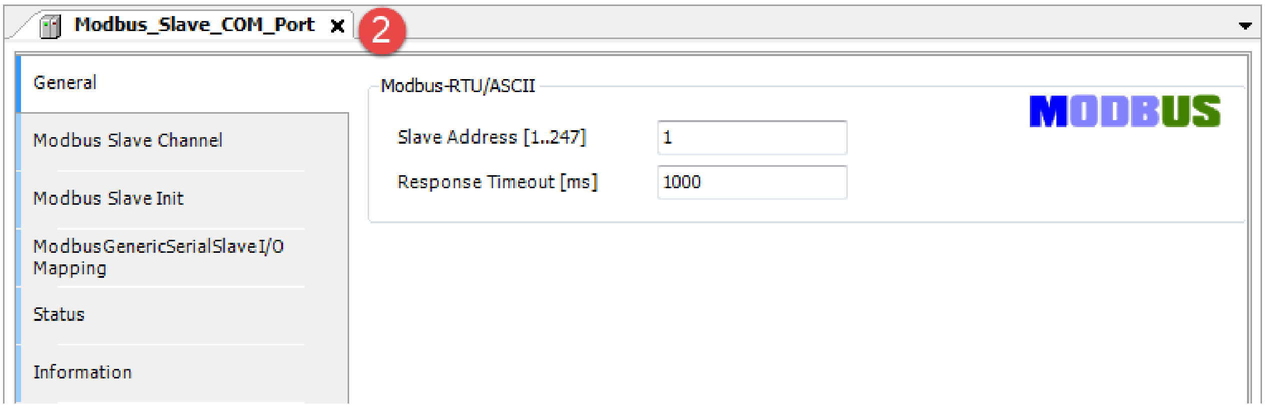
Name	Vendor	Version
Fieldbusses		
Modbus		
Modbus Serial Slave		
Modbus Slave, COM Port	3S - Smart Software Solutions GmbH	3.5.4.0

Once Modbus TCP Slave as been added in project tee double click on *Modbus_COM* > *Modbus_Master_COM_Port* > *Modbus_Slave_COM_Port* (1).

Modbus Slave COM Port configuration window will be displayed (2).



PLC_PRG
Modbus_COM (Modbus COM)
Modbus_Master_COM_Port (Modbus Master, COM Port)
Modbus_Slave_COM_Port (Modbus Slave, COM Port)
1



Modbus-RTU/ASCII

Slave Address [1..247]: 1

Response Timeout [ms]: 1000

Available parameters are:

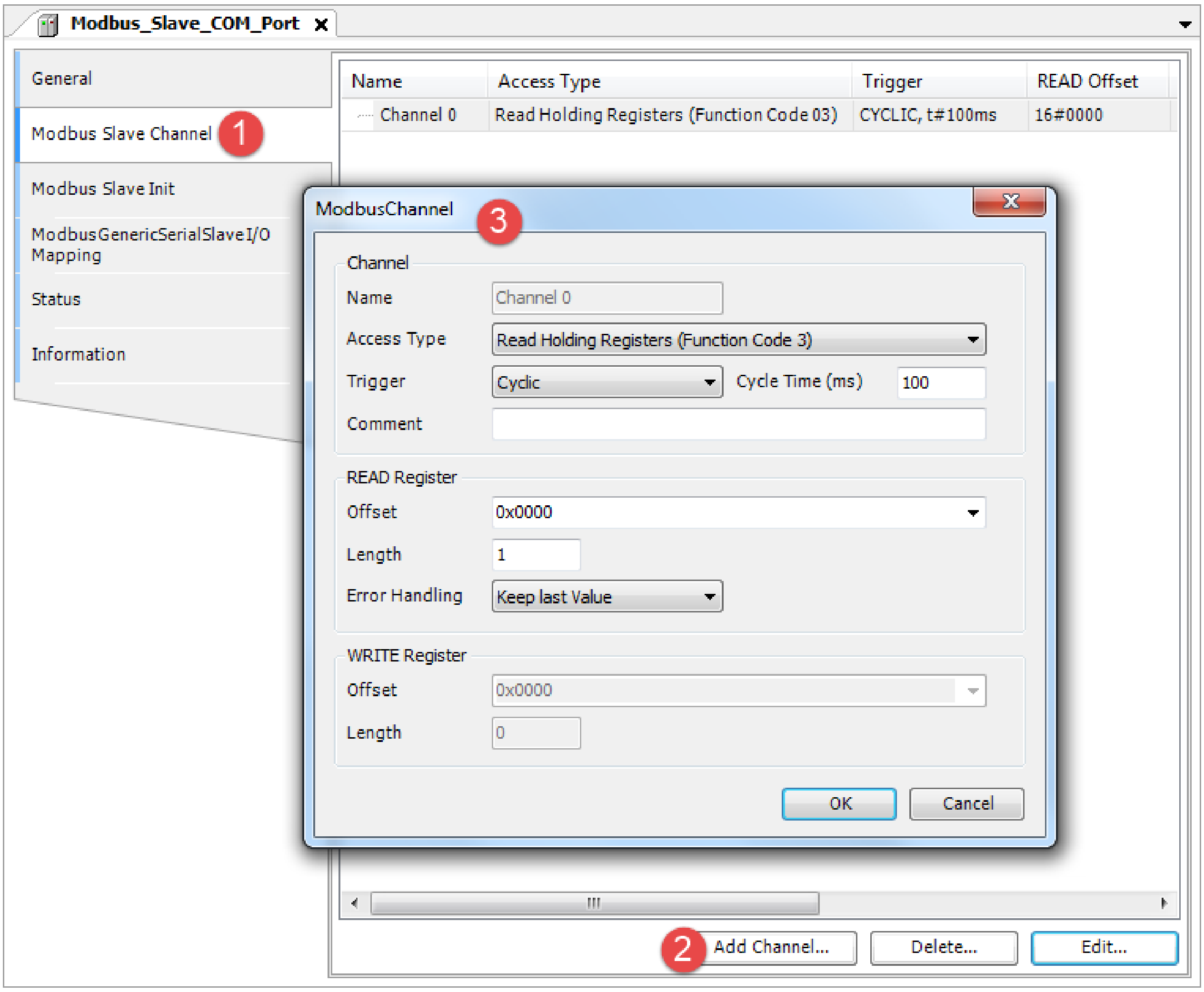
Element	Description
Slave Address [1..247]	Modbus Node ID of the Modbus RTU Slave Device
Response Timeout [ms]	Timeout for Modbus slaves reply, given in milliseconds.

Configuring Modbus Data Exchange

To configure Modbus data exchange select the Modbus Slave Channel tab (1)

The configuration is based on Channels. For each channel it can be configured a Modbus command that will be sent to the Slave.

To add a new Channel click on Add Channel button (2), as shown in the following figure, the ModbusChannel dialog will be displayed (3).



Name	Access Type	Trigger	READ Offset
Channel 0	Read Holding Registers (Function Code 03)	CYCLIC, t#100ms	16#0000

Channel
Name: Channel 0
Access Type: Read Holding Registers (Function Code 3)
Trigger: Cyclic   Cycle Time (ms): 100
Comment:

READ Register
Offset: 0x0000
Length: 1
Error Handling: Keep last Value

WRITE Register
Offset: 0x0000
Length: 0

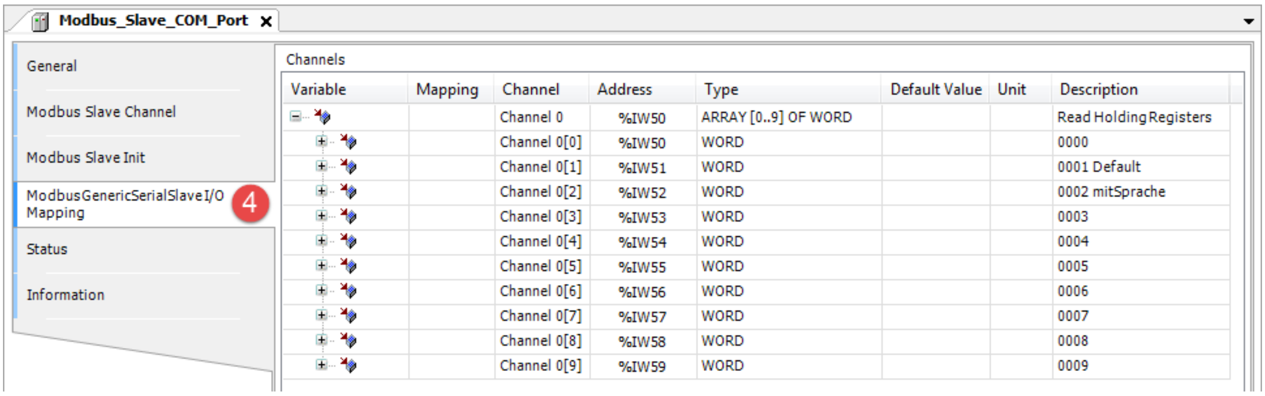
Available parameters are:

Element	Description
Name	Channel Name
Access Type	Selection of the Modbus command
Trigger	Determines if the command execution is cyclic, in this case the Cycle Time must be specified, or Rising Edge, in this case the command is launched on the rising edge event of a boolean variable defined in the Modbus RTU Slave I/O Mapping
Comment	User comment if required
Offset	The starting Modbus address given in HEX format. Note: the addressing is zero based, in case the Slave device has Modbus addressing 1 based the starting address provided must be calculated with -1 offset
Length	Number of registers to be Read/Write

Modbus RTU Slave I/O mapping is available by selecting the ModbusGenericSerialSlave I/O Mapping tab (4).

The Mapping shows a list of all Modbus read/write resources in the configured Channels.

In case the configured Channel uses a Rising Edge triggered command the Trigger bit is listed into the Mapping too.

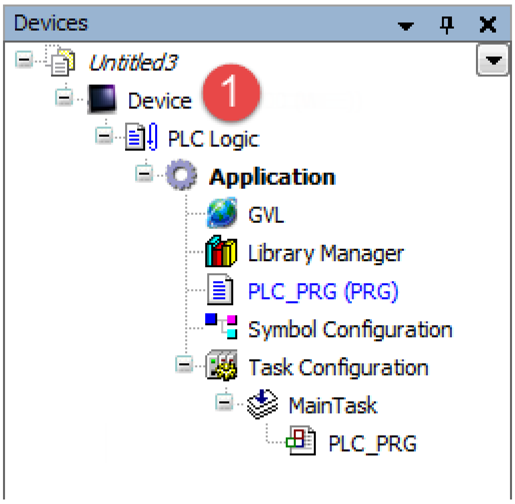


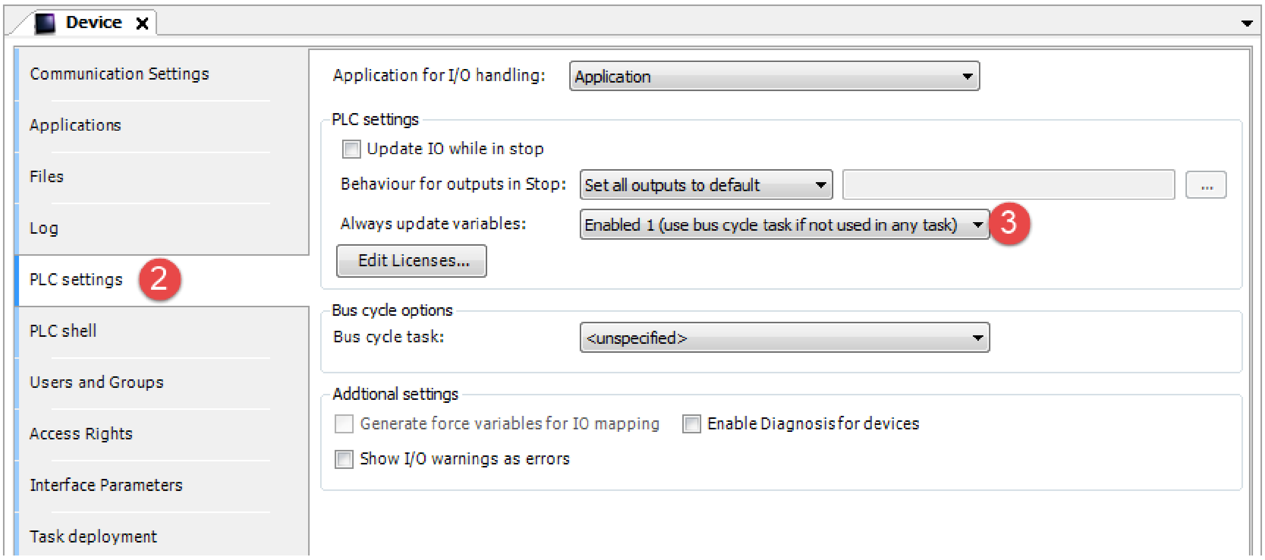
Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Channel 0	%IW50	ARRAY [0..9] OF WORD			Read Holding Registers
		Channel 0[0]	%IW50	WORD			0000
		Channel 0[1]	%IW51	WORD			0001 Default
		Channel 0[2]	%IW52	WORD			0002 mitSprache
		Channel 0[3]	%IW53	WORD			0003
		Channel 0[4]	%IW54	WORD			0004
		Channel 0[5]	%IW55	WORD			0005
		Channel 0[6]	%IW56	WORD			0006
		Channel 0[7]	%IW57	WORD			0007
		Channel 0[8]	%IW58	WORD			0008
		Channel 0[9]	%IW59	WORD			0009

Update I/O variables

To always update I/O variables:

1. Double click on **Device (<*device_name*>)**
2. Select PLC settings tab
3. Set "Enabled 1 (use bus cycle task if not used in any task)" in "Always update variables" option





Application for I/O handling: Application

PLC settings

□ Update IO while in stop

Behaviour for outputs in Stop: Set all outputs to default

Always update variables: Enabled 1 (use bus cycle task if not used in any task)

Bus cycle options

Bus cycle task: <unspecified>

Addtional settings

□ Generate force variables for IO mapping □ Enable Diagnosis for devices

□ Show I/O warnings as errors

Available options are:

Option	Description
Disabled	I/O variables are updated only if used in a task
Enabled 1	I/O variables are always updated by using bus cycle task if not used in any task
Enabled 2	I/O variables are always updated by using bus cycle task

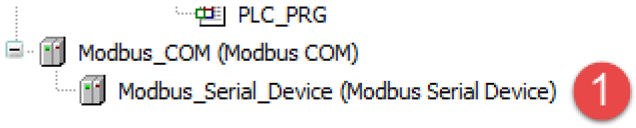
The above option will be inherited by a specific I/O item if "Always update variables" option is set as "Use parent device setting", from *I/O Mapping* tab.

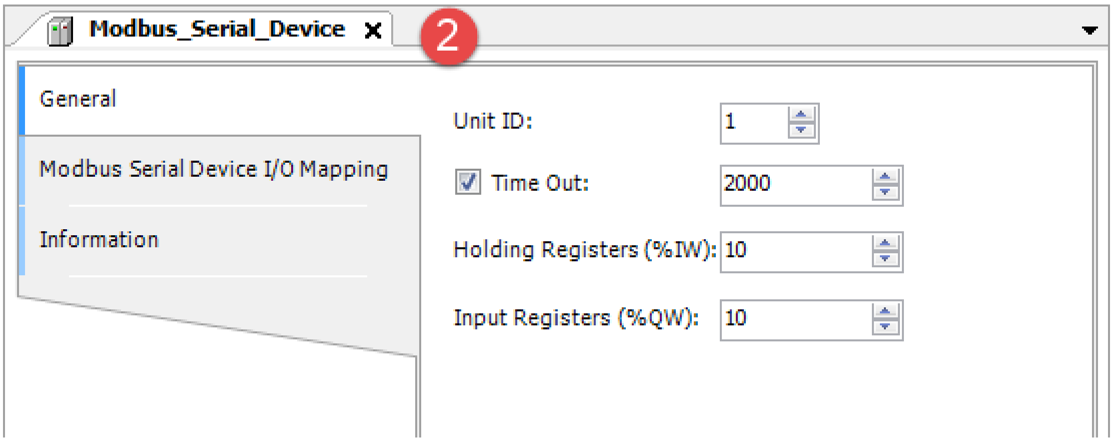
Configuring PLC as Modbus RTU Slave Device

Note: When programmed as explained in this chapter the CODESYS V3 PLC will act as a Modbus RTU Slave device. To configure the device for communication with remote Modbus RTU I/O slaves please refer to *Configuring PLC as Modbus RTU Master* chapter

Once Modbus Serial Device as been added in project tee double click on *Modbus_COM* > *Modbus_Serial_Device* (1).

Modbus Serial Device configuration window will be displayed (2).





Unit ID: 1

☑ Time Out: 2000

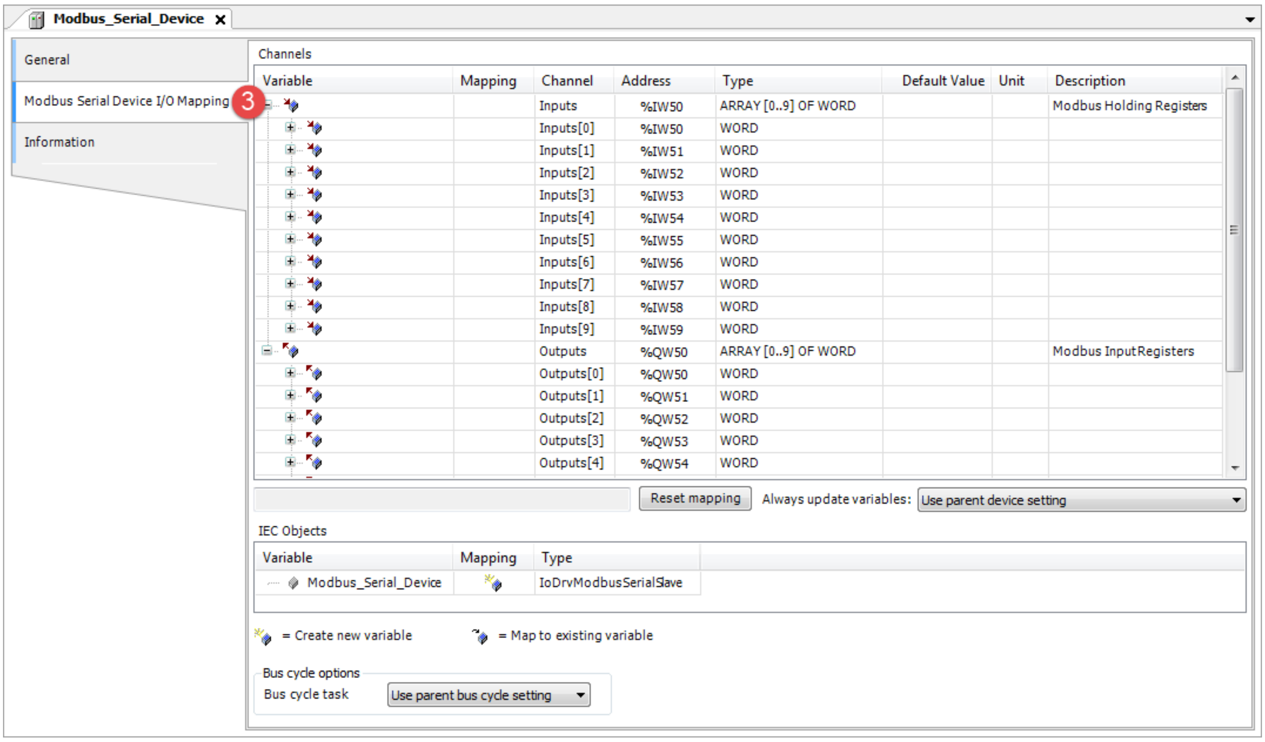
Holding Registers (%IW): 10

Input Registers (%QW): 10

Available parameters are:

Element	Description
Unit ID	Modbus Node ID of the Modbus RTU Slave Device
Timeout	If selected, timeout for Modbus Master queries, given in milliseconds
Holding Registers (%IW)	Number of holding registers available in PLC memory
Input Registers (%QW)	Number of input registers available in PLC memory

Modbus RTU Slave Device I/O mapping is available by selecting the Modbus Serial Device I/O Mapping tab (3).



Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Inputs	%IW50	ARRAY [0..9] OF WORD			Modbus Holding Registers
		Inputs[0]	%IW50	WORD			
		Inputs[1]	%IW51	WORD			
		Inputs[2]	%IW52	WORD			
		Inputs[3]	%IW53	WORD			
		Inputs[4]	%IW54	WORD			
		Inputs[5]	%IW55	WORD			
		Inputs[6]	%IW56	WORD			
		Inputs[7]	%IW57	WORD			
		Inputs[8]	%IW58	WORD			
		Inputs[9]	%IW59	WORD			
		Outputs	%QW50	ARRAY [0..9] OF WORD			Modbus InputRegisters
		Outputs[0]	%QW50	WORD			
		Outputs[1]	%QW51	WORD			
		Outputs[2]	%QW52	WORD			
		Outputs[3]	%QW53	WORD			
		Outputs[4]	%QW54	WORD			

Reset mapping   Always update variables: Use parent device setting

IEC Objects

Variable	Mapping	Type
Modbus_Serial_Device		IoDrvModbusSerialSlave

= Create new variable   = Map to existing variable

Bus cycle options

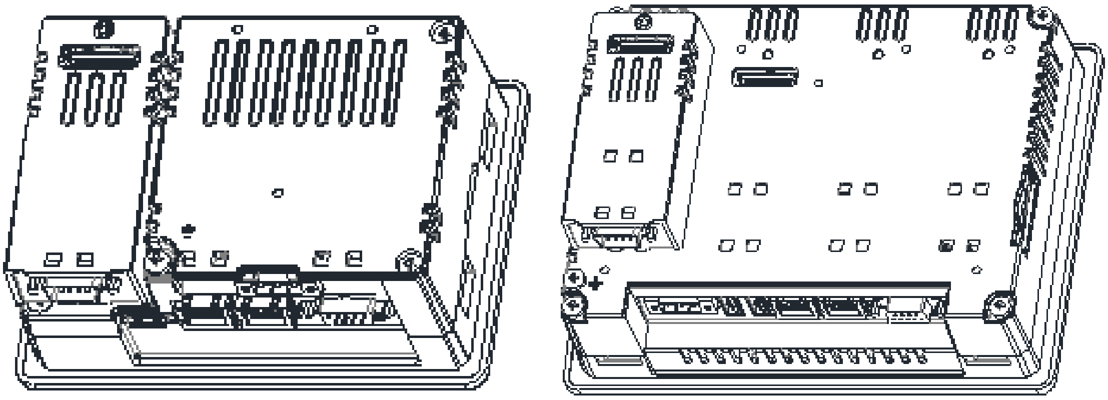
Bus cycle task: Use parent bus cycle setting

CAN Master

CODESYS V3 Runtime can act as a CAN Master to allow the use of distributed CANopen I/O points.
To interface the HMI with a CAN network one of the available CAN optional modules must be installed.

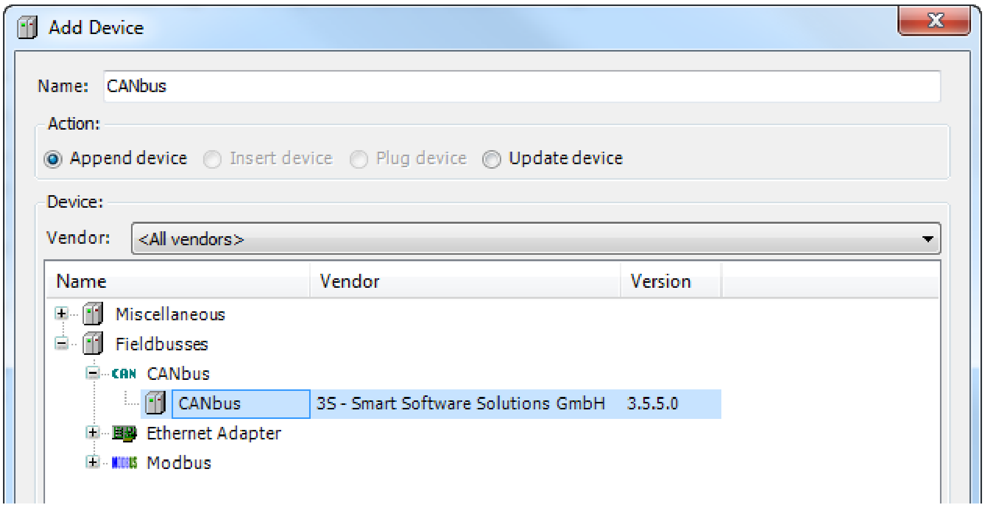
Note: Availability of optional plug-in modules depends on the HMI model, please check *Control Solutions with HMI and CODESYS V3* chapter to verify availability for your HMI.

The following figure shows HMIs with CAN optional module mounted.



To add a CANopen Master interface two steps are required.

Right click on **Device (<*device_name*>)** in the Project tree and select “Add Device”. CANbus device is located under *Fieldbusses > CANbus* category. Choose CANbus device and click on “Add Device” to add it to the current PLC configuration.



Name: CANbus

Action:

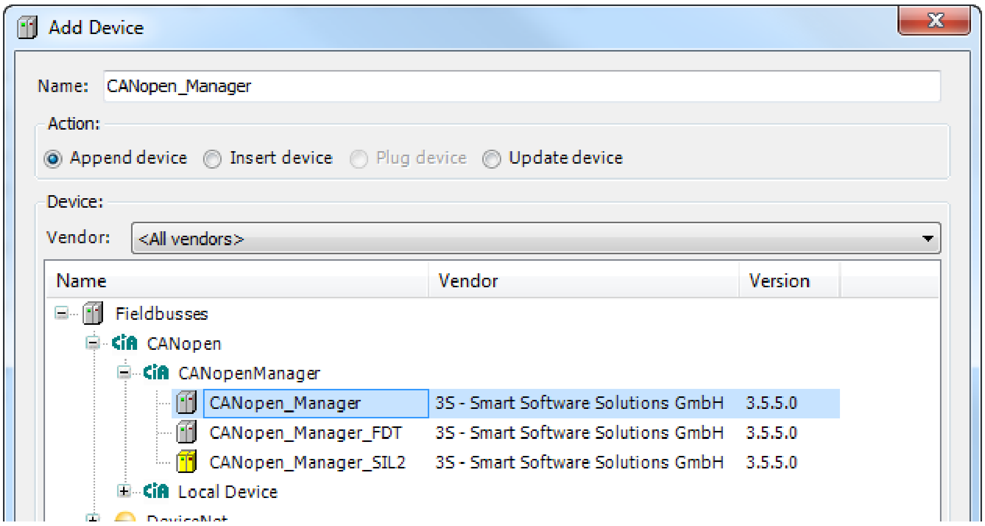
● Append device ○ Insert device ○ Plug device ○ Update device

Device:

Vendor: <All vendors>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
CANbus		
CANbus	3S - Smart Software Solutions GmbH	3.5.5.0
Ethernet Adapter		
Modbus		

Right click on CANbus in the project tree and select “Add Device”.
CANopen Manager device is located under *Fieldbusses* > *CiA CANopen* > *CiA CANopen Manager* category. Choose CANopen Manager device and click on “Add Device” to add it to the current PLC configuration.



Name: CANopen_Manager

Action:

● Append device ○ Insert device ○ Plug device ○ Update device

Device:

Vendor: <All vendors>

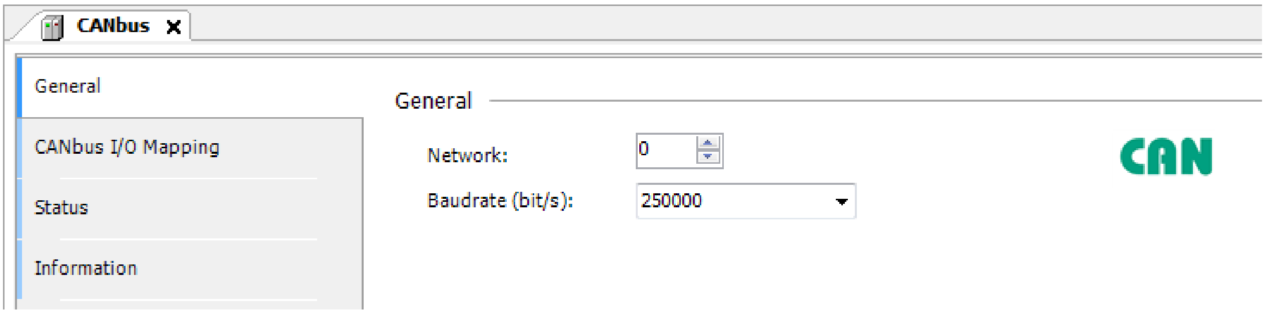
Name	Vendor	Version
Fieldbusses		
CANopen		
CANopenManager		
CANopen_Manager	3S - Smart Software Solutions GmbH	3.5.5.0
CANopen_Manager_FDT	3S - Smart Software Solutions GmbH	3.5.5.0
CANopen_Manager_SIL2	3S - Smart Software Solutions GmbH	3.5.5.0
Local Device		
DeviceNet		

Parameters of CAN interface are grouped in three tabs accessible on the right part of PLC Configuration tool when the Can Master element has been added to the configuration tree.

Note: A complete and detailed description on the configuration of CAN controllers and on the configuration of CAN slave devices is included in CODESYS Development System User Manual

CAN Bus Network Configuration

CAN bus configuration is displayed in the Work area by selecting CANbus tab after a double click on *CANbus* in project tree.



General

Network: 0

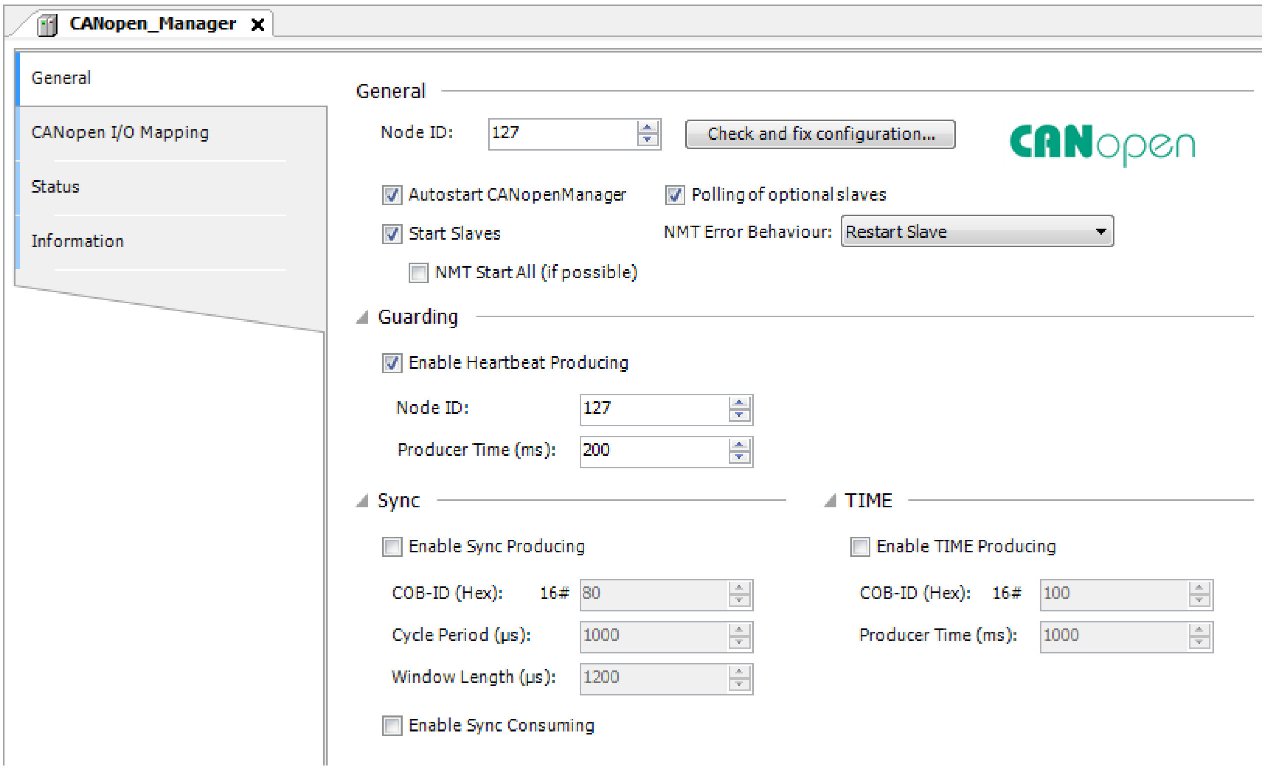
Baudrate (bit/s): 250000

Available parameters are:

Element	Description
Network	CAN network number 0 is default, in case of double CAN network configuration Network 1 is used for the second CAN network.
Baudrate (bit/s)	CAN network baudrate.

CANopen Master Configuration

CANopen Master configuration is displayed in the Work area by selecting CANopen Manager tab after a double click on *CANbus > CANopen Manager* in project tree.



General

Node ID: 127

☑ Autostart CANopenManager

☑ Polling of optional slaves

☑ Start Slaves

NMT Error Behaviour: Restart Slave

□ NMT Start All (if possible)

Guarding

☑ Enable Heartbeat Producing

Node ID: 127

Producer Time (ms): 200

Sync

□ Enable Sync Producing

COB-ID (Hex): 16# 80

Cycle Period (µs): 1000

Window Length (µs): 1200

□ Enable Sync Consuming

TIME

□ Enable TIME Producing

COB-ID (Hex): 16# 100

Producer Time (ms): 1000

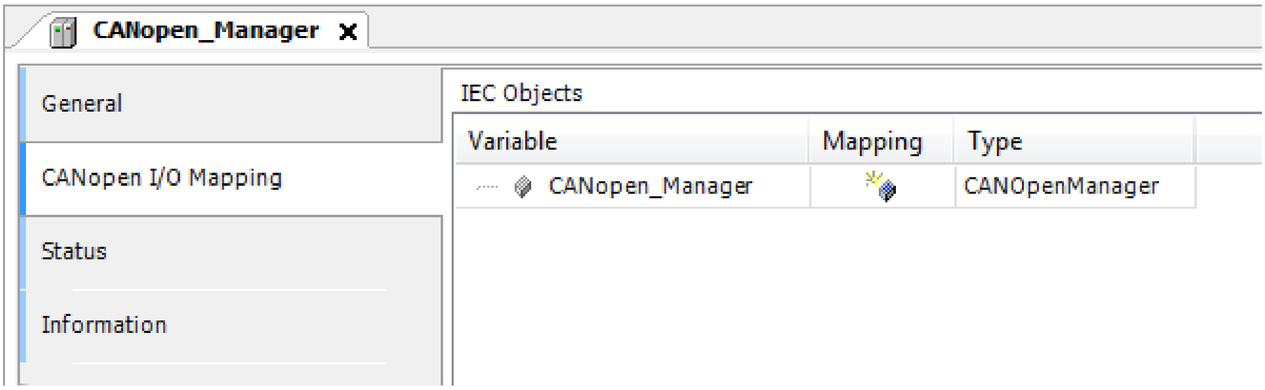
Available parameters are:

Element	Description
Node ID	CAN node number assigned to the CAN master.
Autostart CANopen Manager	If selected, the CANopen Manager starts automatically if all mandatory slaves are ready. If not selected, the manager must be started by the application, using the CiA405 NMT function block for this purpose.
Polling of optional slaves	If an optional slave does not respond during start-up sequence the system polls the slave every second until the slave responds successfully.
Start Slaves	If selected the CAN Master starts the slaves automatically, otherwise the start operation must be done into the application.
NMT Start All	If Start Slaves option is selected it is possible to enable NMT Start All function, this function will start all the slaves at the same time when all the slaves are ready. If not enabled each slave is started separately.
NMT Error Behavior	This option allows to determine the behavior on a guard event, the available options are Restart Slave or Stop Slave
Enable Sync Producing	Enable the sending of Sync telegrams on the CAN bus
COB-ID (Hex)	COB-ID of the Sync message, standard ID is 128 (80 Hex)
Cycle Period (µs)	Time interval between two Sync messages, given in microseconds
Window length (µs)	Length of the time window for synchronous PDOs, given in microseconds
Enable Sync consuming	If selected the Sync messages are supposed to be produced by a different device on the CAN network, the CANopen Manager will receive such messages.
Enable Heartbeat producing	If selected, the Master sends Heartbeat messages on the CAN network.
Node ID	CAN Identifier of the Heartbeat messages producer (1-127)
Producer Time (ms)	Time interval between two Heartbeat messages, given in milliseconds.
Enable TIME Producing	If selected the Master sends TIME messages on the CAN network.
COB-ID (Hex)	COB-ID of the TIME messages, default value is 256 (100 Hex)
Producer Time (ms)	Time between two TIME messages, given in milliseconds. Must be a multiple of the task cycle time.

Diagnostic Mapping

When a CAN master device is added to the PLC configuration, the system automatically creates a Variable of type CANOpenManager into the project. This variable contains the diagnostic information about the CAN Master.

The Variable name is assigned by default but can be changed by changing the I/O module name into the PLC configuration. The Variable is reported into the CANopen I/O Mapping tab, displayed in the work area by double clicking on the CANbus > CANopen Manager in the project tree, as shown in the following figure.

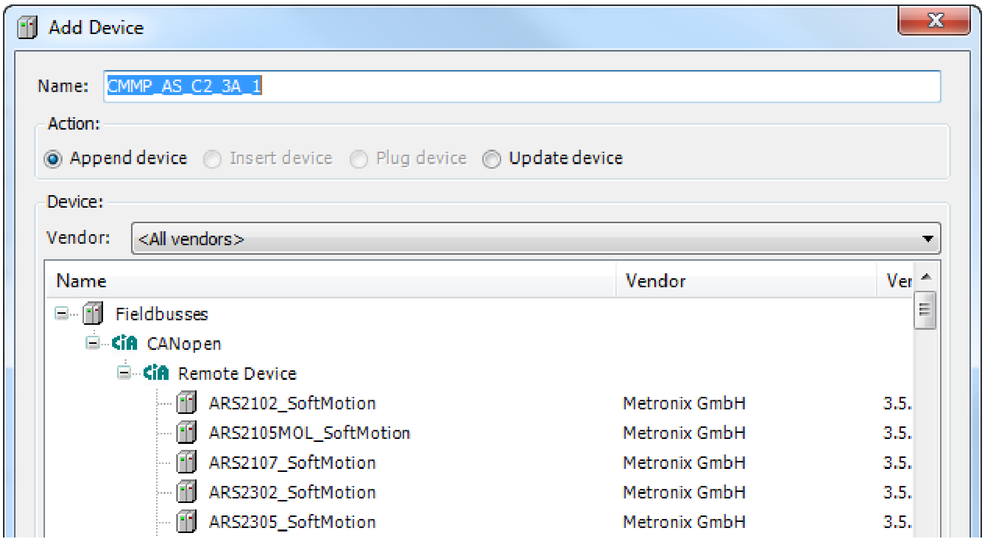


Variable	Mapping	Type
CANopen_Manager		CANOpenManager

Definition of CAN I/O Slaves in the PLC Configuration.

After the set-up of the CAN Master the structure of available CAN I/O Slaves can be defined.

To add a CANopen I/O Slave, right click on CANopen Manager in Project tree and select “Add Device”.
The list of available CANopen I/O slaves is located under *Fieldbusses* > *CiA CANopen* > *CiA Remote Device* category. Choose the device and click on “Add Device” to add it to the current PLC configuration.



Name: CMMP_AS_C2_3A_1

Action: ● Append device ○ Insert device ○ Plug device ○ Update device

Device:
Vendor: <All vendors>

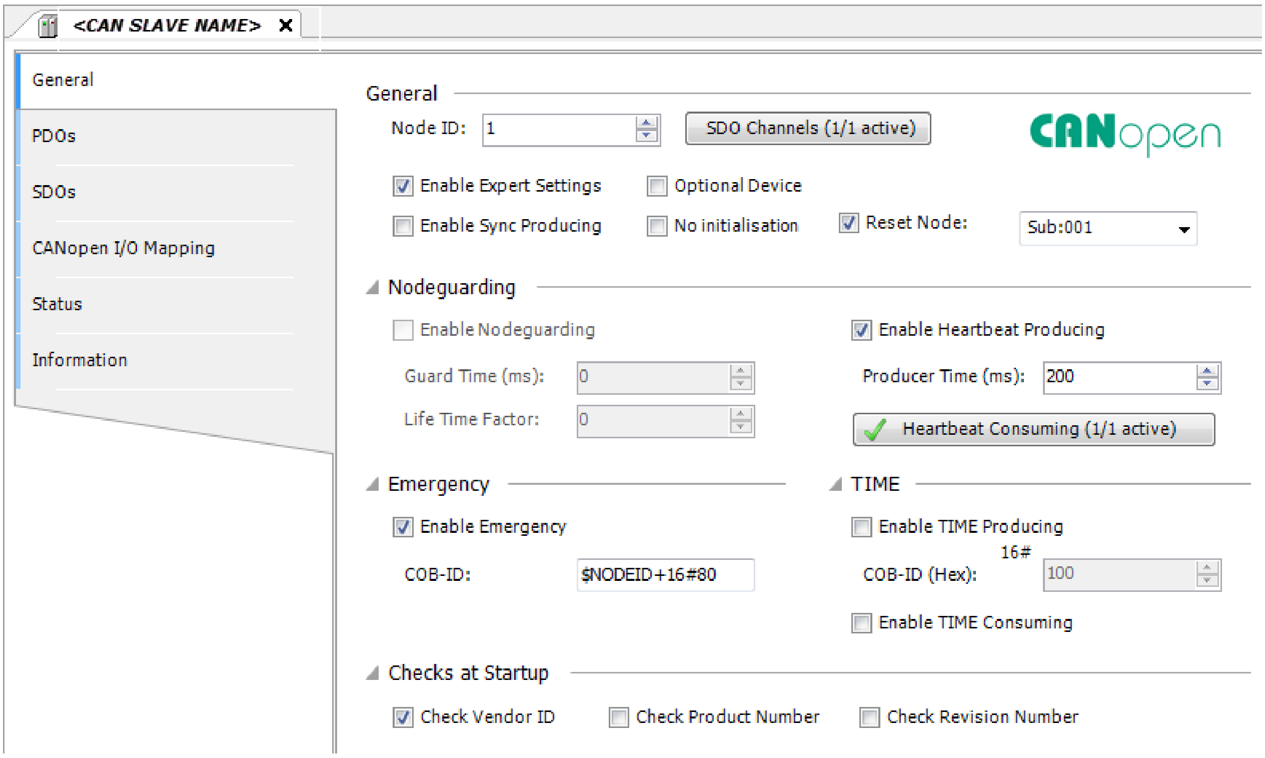
Name	Vendor	Ver
Fieldbusses		
CANopen		
Remote Device		
ARS2102_SoftMotion	Metronix GmbH	3.5.
ARS2105MOL_SoftMotion	Metronix GmbH	3.5.
ARS2107_SoftMotion	Metronix GmbH	3.5.
ARS2302_SoftMotion	Metronix GmbH	3.5.
ARS2305_SoftMotion	Metronix GmbH	3.5.

Note: Installation of third part CAN Slaves into CODESYS V3 requires a specific EDS descriptor file, provided by the CAN Slave manufacturer. Refer to the CODESYS Development System User Manual for detailed information regarding EDS file installation.

Settings for CAN Slaves

CAN slaves configuration has a common part, which is independent from the EDS file. The next figure shows the CANopen Remote Device tab of a CAN Slave displayed in the work area by double clicking on *CANbus* > *CANopen Manager* > *CAN SLAVE NAME* in the project tree.
Parameters shown in the figure can be displayed by selecting *Enable Expert Settings* option.

Note: For specific information regarding the set-up of the CAN Slaves please refer to the documentation provided from the CAN Slave manufacturer.



General

Node ID: 1

☑ Enable Expert Settings □ Optional Device

□ Enable Sync Producing □ No initialisation ☑ Reset Node: Sub:001

Nodeguarding

□ Enable Nodeguarding ☑ Enable Heartbeat Producing

Guard Time (ms): 0 Producer Time (ms): 200

Life Time Factor: 0

Emergency

☑ Enable Emergency

COB-ID: $NODEID+16#80

TIME

□ Enable TIME Producing

COB-ID (Hex): 16# 100

□ Enable TIME Consuming

Checks at Startup

☑ Check Vendor ID □ Check Product Number □ Check Revision Number

Available parameters are:

Element	Description
Node ID	CAN node number of the CAN Slave device (1 – 127).
Enable Sync Producing	If selected this Device send Sync messages on the CAN network. This option is selectable only if the Sync messages production at CANopen Master side is disabled.

Element	Description
No initialization	If selected the sequence of SDO messages required for the device initialization (PDO mapping) will not be created.
Optional device	If selected the current device is considered as optional into the bus. At start-up the CAN controller will check if it is present applying the following rules: • If the device is present since start-up and correctly replies to the CANopen mandatory object “Device Type” query (matching the EDS file specification), then it is started. The master will continue with the next device. • If the device is present since start-up and it does not reply as expected to the “Device type” query, it is not started. The master stops then, reporting a mismatching error in the CAN configuration; if the “Optional device” with not-matching “Device Type” is inserted in the bus after start-up, the master will skip it and continue to scan the other devices. • If the device is not present since start-up, it is simply skipped. The master will continue with the next device.

CAN Interface

CAN optional module includes a CAN bus interface implemented according to CAN protocol specifications 2.0 A.

This CAN controller supports only Standard frame format (2.0 A) with bit rates up to 1 Mbit/s.

The following transfer functions have been implemented:

- Transfer rate and timing
- Message framing (Part A)
- Arbitration accordingly to Part A specifications
- Automatic retransmission in case of lost arbitration or error detection
- Acknowledgment
- Message validation
- Error detection and error signaling
- Global Identifier masking (for 11-bit and 29-bit long identifiers)
- Interrupt or data polling driven software supported
- Automatic transfer of data frame (prepared in SDRAM buffer) triggered by one bit setting
- Automatic receive of data packets with the allowed frame identifier
- 32 separated SDRAM memory buffers for data packets having the node corresponding ID
- Fully implemented CAN error fault confinement
- Automatic detection of Bus off state
- Detection of the heavily disturbed CAN bus and warning

Programming the parameter baudRateKbps at the value 0 enables the use of custom timing.

The resulting baud rate is calculated using the formula:

Bit frequency = 8 MHz / (Prescaler * (1 + Tsetup + Thold))

Valid values for parameters are:

Prescaler: 1 to 64

Tsetup: 1 to 8

Thold: 1 to 4

Other two parameters can affect the behavior of the CAN controller:

SyncJumpWidth: defines the number of time quanta (8 MHz / Prescaler) allowed to accept a SYNC pulse. Valid values are 1 to 4.

SampleMode: defines the number of times the bit is sampled before is considered valid. Valid values are 0 (1 sample) and 1 (3 samples).

Internal PLC (CODESYS V2)

This manual describes the Exor control system based on the CODESYS V2 PLC software.

The documentation covers:

- CODESYS V2 running on eTOP Series 400, 500 and 600 products
- Use of integrated I/O optional modules
- Use of remote Modbus/TCP and CANopen optional modules

Note: This manual is not intended as CODESYS V2 programming manual, for specific documentation regarding CODESYS V2 Development System please refer to the CODESYS web site www.codesys.com and to its on-line help.

Control Solutions with Exor and CODESYS V2

Exor products can deliver effective HMI and control solutions based on JMobile and CODESYS V2.

Exor eTOP Series 400, 500 and 600 HMI products have been designed to include the CODESYS V2 PLC runtime.

The PLC runtime is automatically transferred to the device by JMobile Studio as part of the JMobile Runtime and it is running with the support of the operating system of the device.

CODESYS V2 Development System suite is required to develop and debug PLC applications and transfer them to the Exor HMI device.

The CODESYS V2 runtime requires a license activation to be used. CODESYS V2 license is activated by plugging on the operator panel one of the optional modules that includes CODESYS V2 license. The modules are plug & play, no further steps are required to activate the license.

The following table contains a list of optional plug-in modules that include the CODESYS V2 license.

Module	Description
TCM-19	Plug-in module CAN with CODESYS V2 license (only for eTOP Series 400)
PLCM01-CDS	Plug-in module CAN with CODESYS V2 license
PLCM01-NEC	Plug-in module CAN with CODESYS V2 license without Bus Extension connector
PLCM02-CDS	Plug-in module KNX TP with CODESYS V2 license
PLCM05-CDS	Plug-in module with only CODESYS V2 license
PLIO03-CDS	Analog/Digital I/O Plug-in module with CODESYS V2 license
PLIO06-CDS	Digital I/O Plug-in module with CODESYS V2 license
PLIO07-CDS	Relay Output Plug-in module with CODESYS V2 license

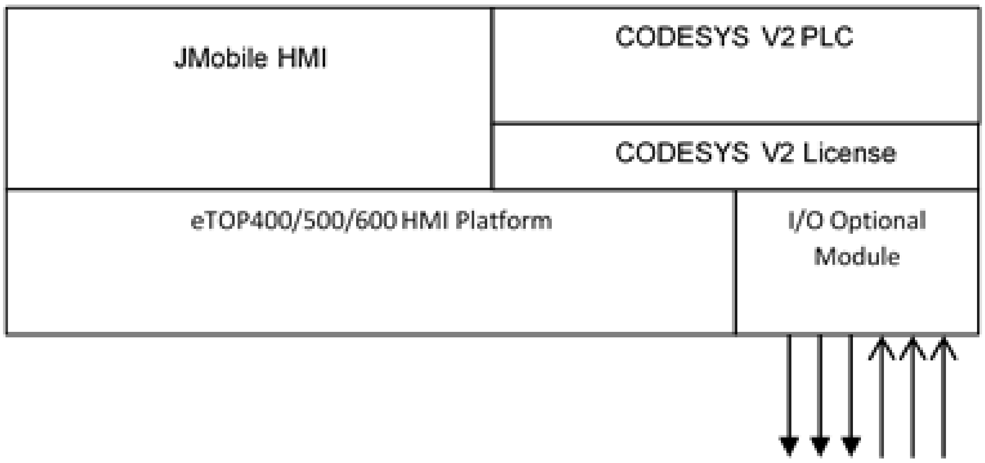
Note: License is required for the operation of CODESYS V2 runtime. CODESYS V2 runtime will communicate with the Development System only with a module containing a license.

System Configuration

The HMI and control solution based on JMobile and CODESYS V2 can be applied in different configurations.

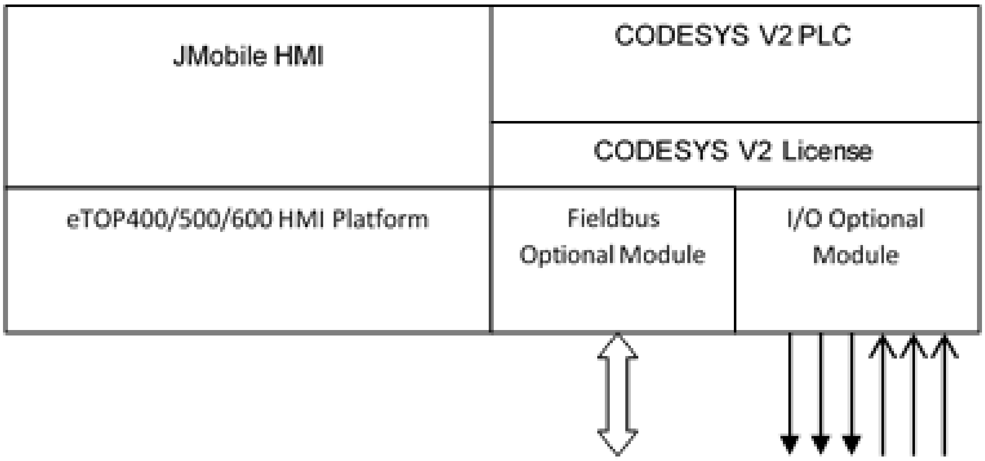
Compact Stand-alone Controller

The HMI and control system can be used to build very compact standalone systems. Input/output is available using the integrated I/O optional modules.



Controller with Remote I/O

A fieldbus interface (either built-in or with an optional module) is available for the HMI and control solutions. Configurations with local and distributed I/O are possible.



Connectivity in HMI and Control Systems

Even when adding the control option with the CODESYS V2 PLC, the HMI still retain its full communication capabilities based on JMobile communication drivers.

Note: The communication capabilities over Serial network are limited by the amount of serial interfaces available, each serial interface supports a single communication protocol. Specific optional add-on serial interface modules are available to increment the amount of serial interfaces at disposal.

Requirements and Limitations

The following firmware and software versions are required to work with the CODESYS HMI and control systems:

Element	Version
JMobile Studio and Runtime	1.80 SP1 or higher
Operator panels BSP version	UN20: 2.65 or higher UN30: 1.66 or higher UN31: 1.66 or higher
CODESYS V2 Development System	2.3.2 or higher

Getting Started

This chapter provides the necessary informations on how to set-up the HMI + CODESYS Plc system. The required operations are listed below and explained in the forthcoming chapters.

1. CODESYS V2 Development System installation
2. Exor Target Support Package installation
3. JMobile Runtime installation into the operator panel
4. Mounting optional Plug-in modules into operator panel to enable CODESYS V2 plc
5. CODESYS V2 Target Settings
6. Download plc application
7. Symbol File configuration with CODESYS V2 Development System
8. Communication Setup in JMobile Studio

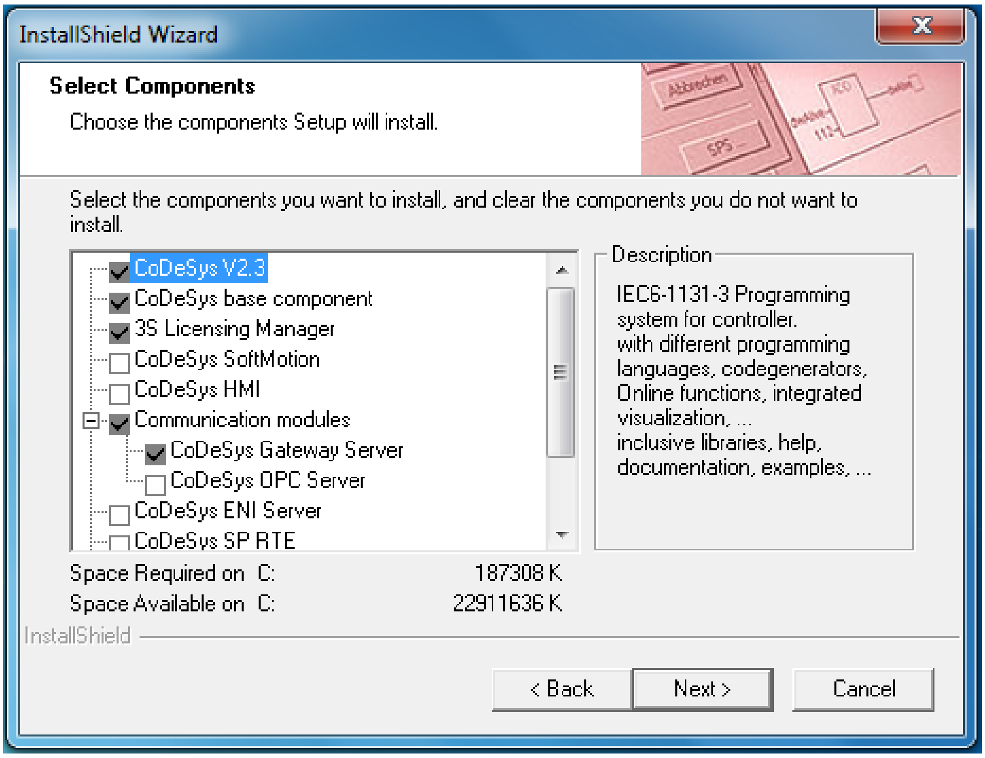
CODESYS V2 Development System installation

The CODESYS V2 Development System can be downloaded for free from the CODESYS web site

at http://www.codesys.com/download.html.

You need to register before you can download the software.

Below the required componets to be selected:



Select Components

Choose the components Setup will install.

Select the components you want to install, and clear the components you do not want to install.

- ☑ CoDeSys V2.3
- ☑ CoDeSys base component
- ☑ 3S Licensing Manager
- □ CoDeSys SoftMotion
- □ CoDeSys HMI
- ☑ Communication modules
  - ☑ CoDeSys Gateway Server
  - □ CoDeSys OPC Server
- □ CoDeSys ENI Server
- □ CoDeSys SP RTE

Description

IEC6-1131-3 Programming system for controller. with different programming languages, codegenerators, Online functions, integrated visualization, ... inclusive libraries, help, documentation, examples, ...

Space Required on C:	187308 K
Space Available on C:	22911636 K

Note: The operator panels internal PLC is compatible with CODESYS V2.3 Development System only. CODESYS V3.5 Development System is NOT compatible with CODESYS V2 Runtime.

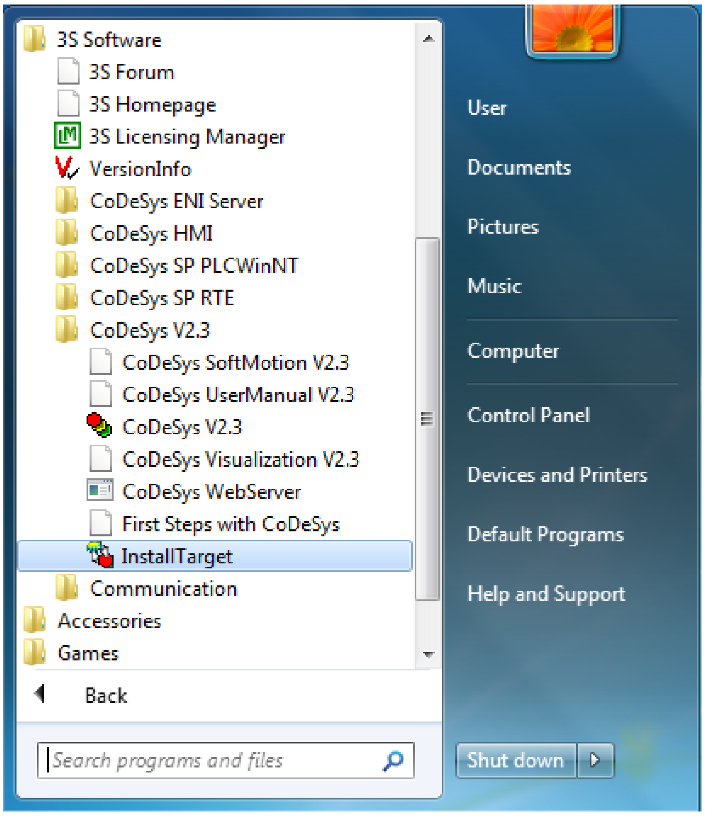
Exor Target Support Package installation

A dedicated Target Support Package (TSP) is required to allow the standard CODESYS V2 Development System to program control systems based on eTOP Series 400, 500 and 600 products.

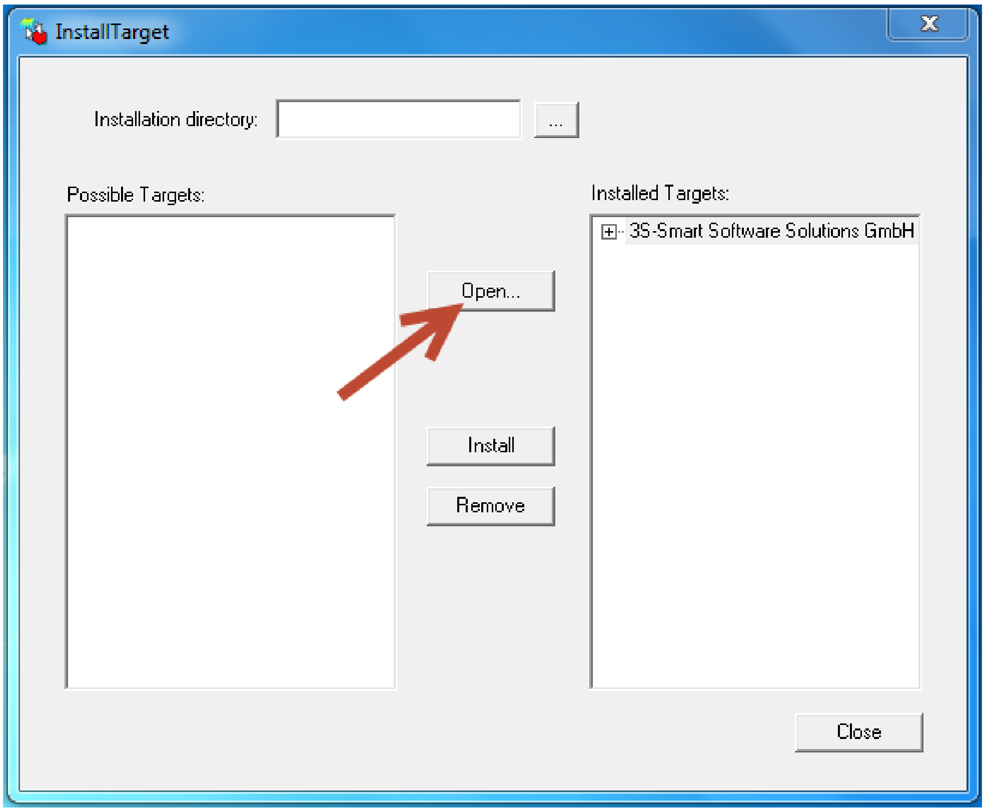
The EXOR TSP is included into JMobile Suite since version 1.91 SP1 under “\CODESYS\V2\” folder, and is also available for download from the web site www.exorint.net.

The TSP is provided as a zip file that must be unzipped to a folder respecting its internal directory structure.

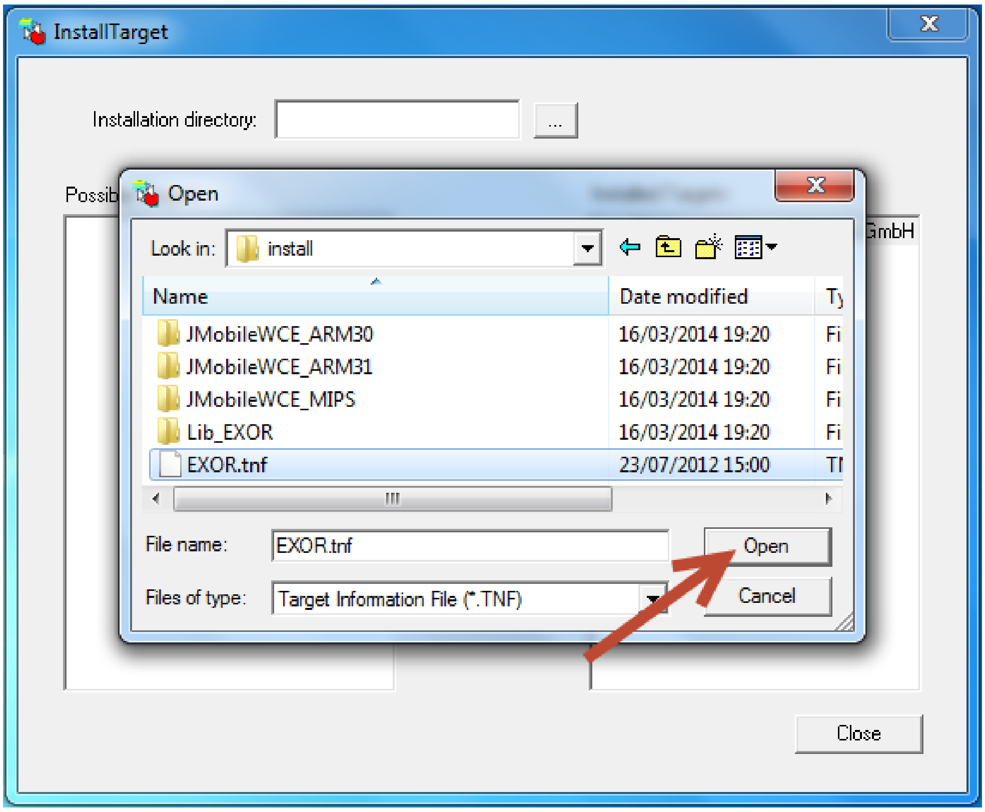
The CODESYS Development System includes a tool called “Install Target” for the installation of the Target Support Package. The Install Target tool is part of the CODESYS program group that can be found in the Start Menu after the installation of the software. This tool can be used both for checking the installed TSP and for installing new ones.



To install the Target Support Package for JMobile click on the "Open" button:

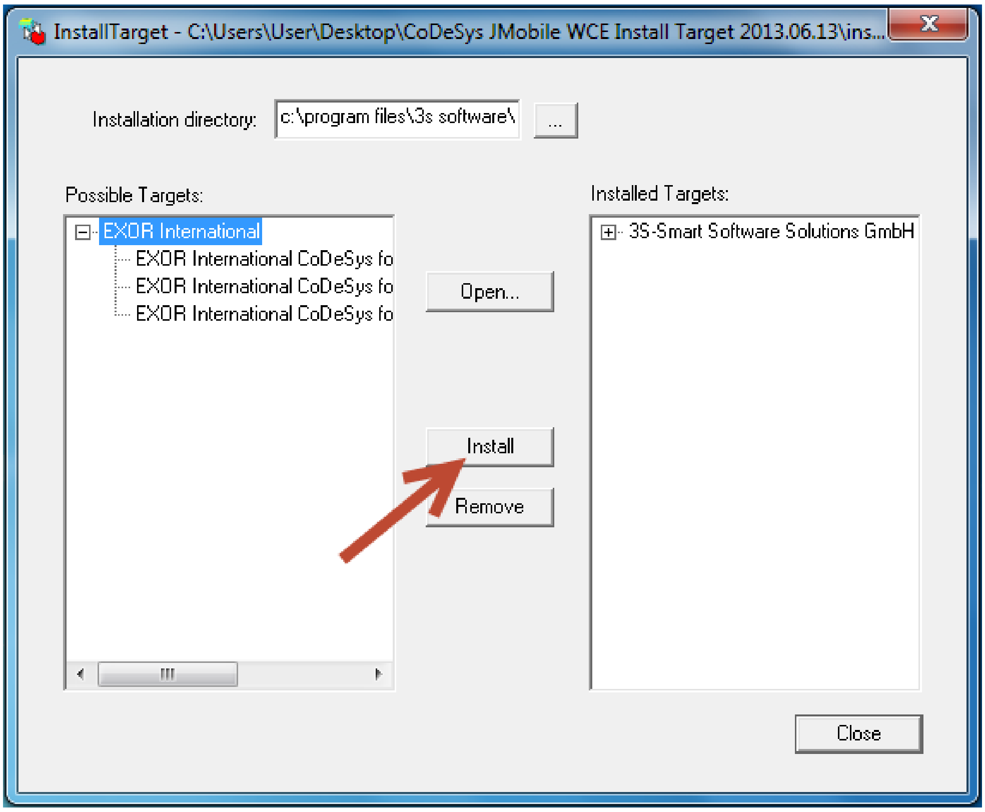


browse for the file with .tnf extension present into the TSP folder extracted from the zip package:

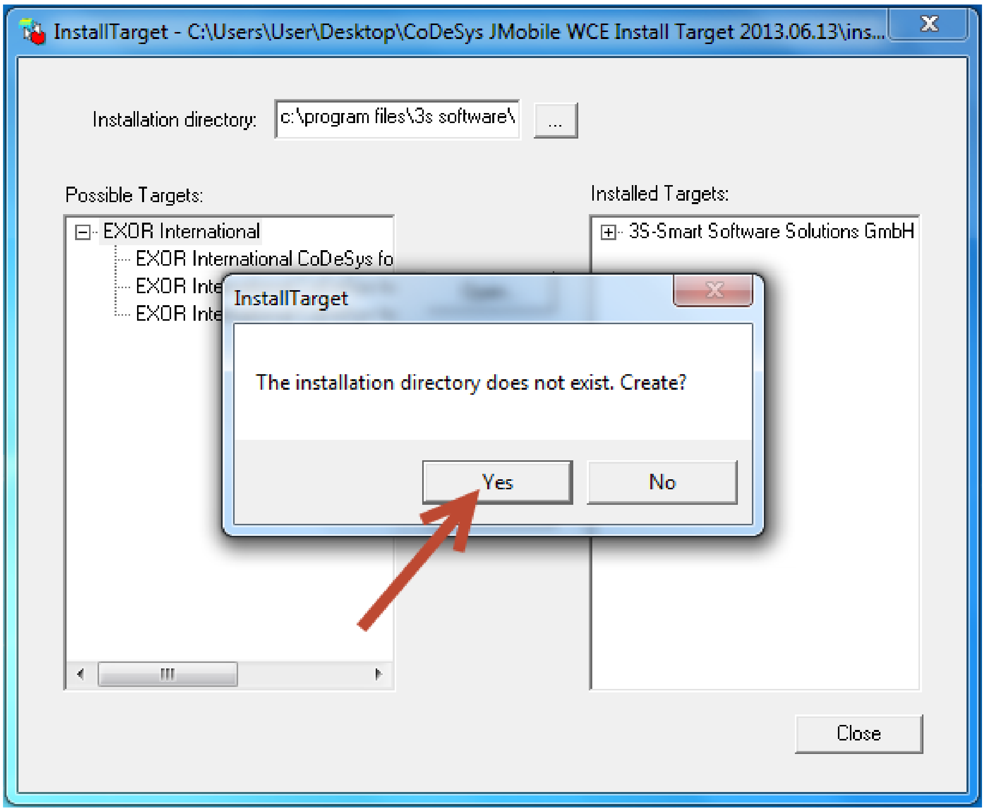


InstallTarget
Installation directory:
Possib
Open
Look in: install
Name
Date modified
Ty
JMobileWCE_ARM30 16/03/2014 19:20 Fi
JMobileWCE_ARM31 16/03/2014 19:20 Fi
JMobileWCE_MIPS 16/03/2014 19:20 Fi
Lib_EXOR 16/03/2014 19:20 Fi
EXOR.tnf 23/07/2012 15:00 TI
File name: EXOR.tnf
Files of type: Target Information File (*.TNF)
Open
Cancel
GmbH
Close

Select an item from the “Possible Targets" list and click on “Install” button to install the TSP into the System:



InstallTarget - C:\Users\User\Desktop\CoDeSys JMobile WCE Install Target 2013.06.13\ins...
Installation directory: c:\program files\3s software\
Possible Targets:
EXOR International
EXOR International CoDeSys fo
EXOR International CoDeSys fo
EXOR International CoDeSys fo
Open...
Install
Remove
Installed Targets:
3S-Smart Software Solutions GmbH
Close

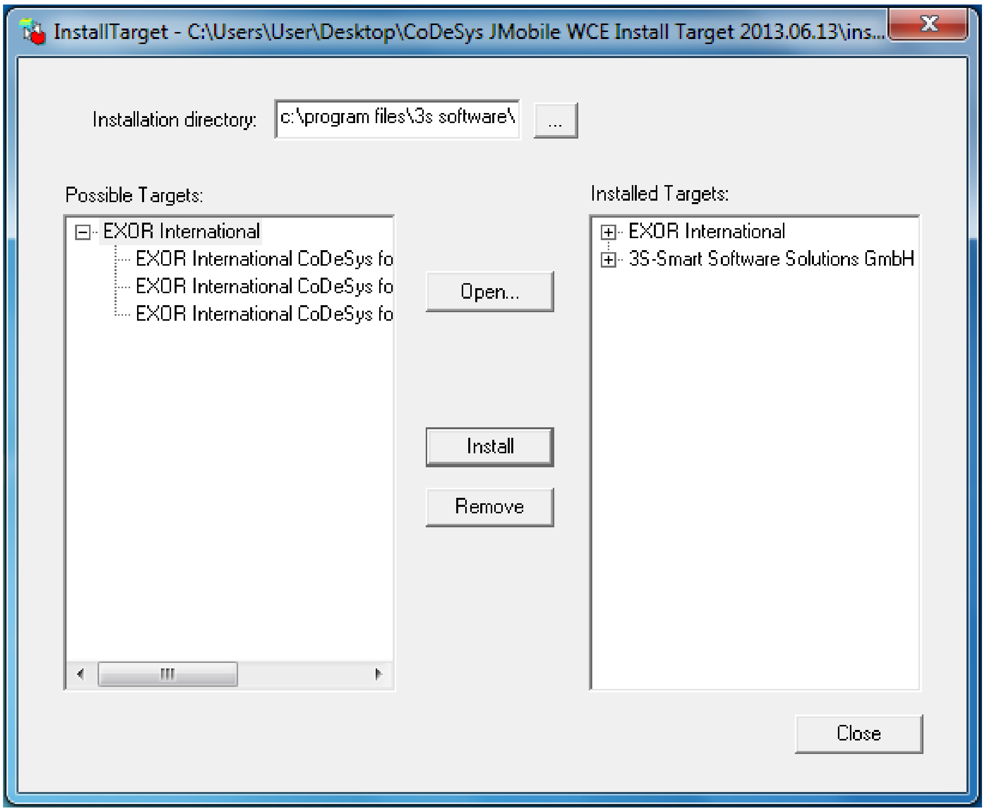

InstallTarget - C:\Users\User\Desktop\CoDeSys JMobile WCE Install Target 2013.06.13\ins...

Installation directory: c:\program files\3s software\

InstallTarget

The installation directory does not exist. Create?

The result will be as shown in the following figure with an example of TSP data.


InstallTarget - C:\Users\User\Desktop\CoDeSys JMobile WCE Install Target 2013.06.13\ins...

Installation directory: c:\program files\3s software\

...
Possible Targets:
EXOR International
EXOR International CoDeSys fo
EXOR International CoDeSys fo
EXOR International CoDeSys fo
Open...
Install
Remove
Installed Targets:
EXOR International
3S-Smart Software Solutions GmbH
Close

The following Targets are included into the EXOR TSP for JMobile:

Target	Platform
EXOR CODESYS for JMobileWCE (WCE/ARM UN31)	eTOP Series 500 and Series 600 products with 600MHz CPU
EXOR CODESYS for JMobileWCE (WCE/ARM UN30)	eTOP Series 500 and Series 600 products with 1GHz CPU
EXOR CODESYS for JMobileWCE (WCE/MIPS)	eTOP Series 400 products

Updating Old Target Support Packages

When a new version of the Target Support Package is available, it can be installed replacing the older one. EXOR always tries to guarantee the complete compatibility with the previous TSP versions.

If the new TSP is not fully compatible with the previous version some changes to adapt the PLC application to the new TSP may be required. In these cases, the updates to the Target Support Package, will come with proper instructions for the conversion of existing projects.

The TSP update procedure is identical to the first installation of the TSP, please refer to the proper chapter for specific informations.

JMobile Runtime installation

The HMI and control system is composed by two main subsystems, the JMobile HMI runtime and the CODESYS V2 PLC runtime.

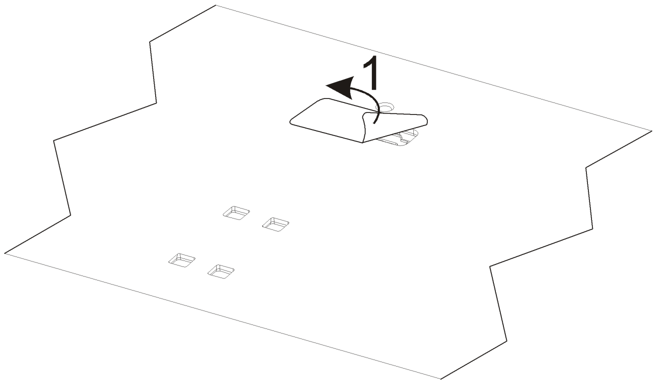
As the CODESYS PLC runtime is part of the JMobile HMI runtime it is necessary to install the JMobile Runtime on the operator panel to have the PLC runtime running. For further information on installing the runtime, please refer to JMobile Studio help.

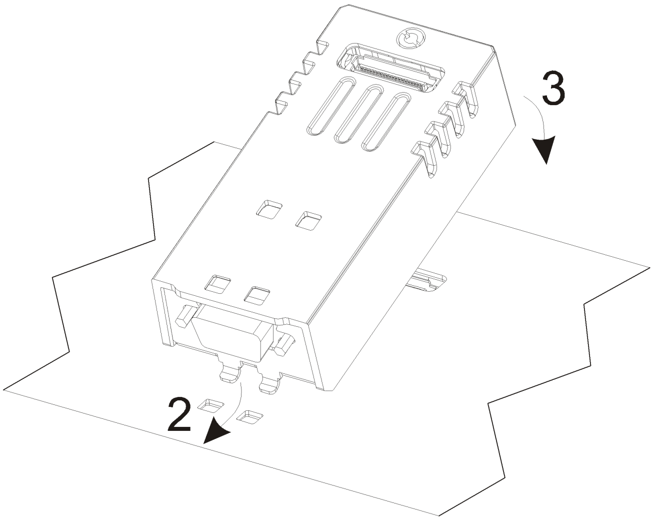
Mounting Plug-in modules to enable CODESYS V2 plc

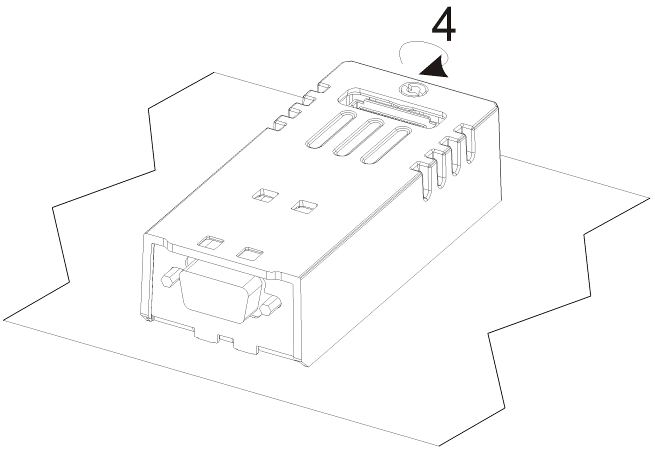
Optional modules of type PLCM (Communication Modules) and PLIO (I/O modules) with CODESYS V2 license can be installed in all eTOP Series 500 and 600 HMI products.

The steps to follow for the installation of the optional module are shown in the following figure and are the following:

1. If present, remove the sticker covering the plug-in slot
2. Insert the guides of the module into the holes in the enclosure
3. Press the module down into the connector
4. Lock the module using the screw

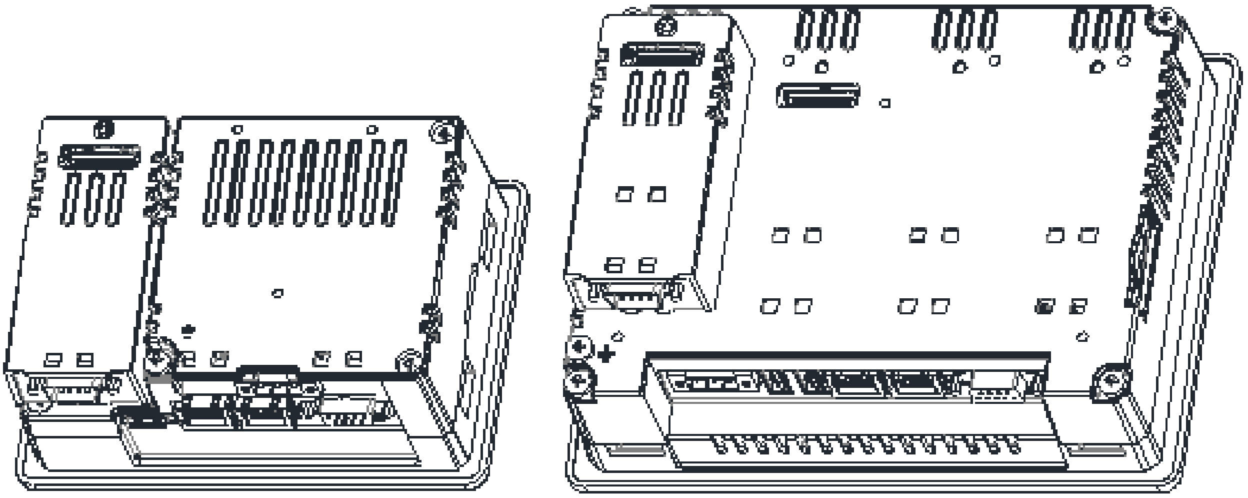






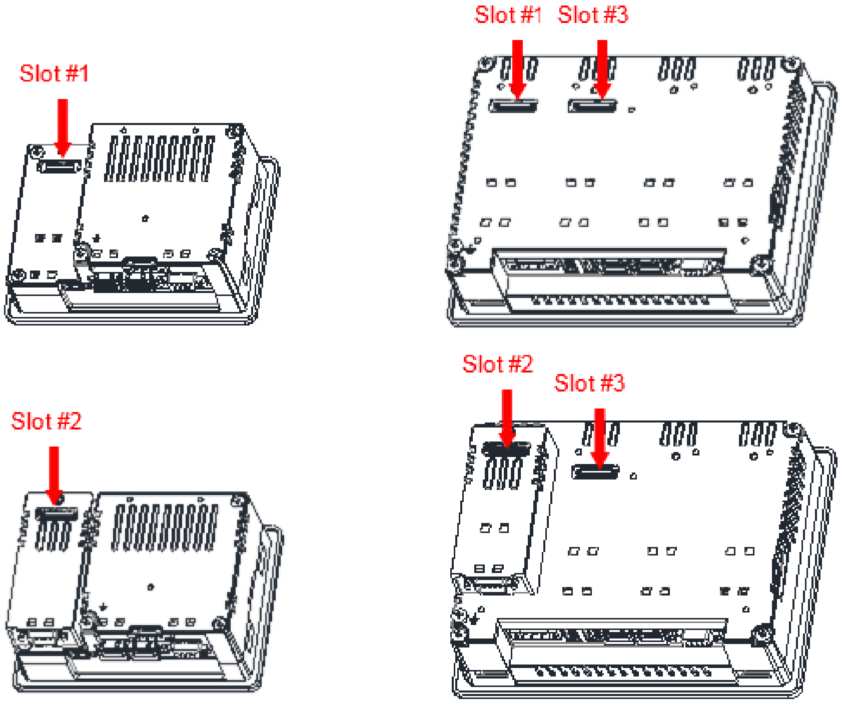
Note: Do not plug or remove optional modules when the HMI device is powered.

The following figure shows eTOP504 and eTOP507 with option modules PLCM01-CDS mounted.



Slot Numbers

eTOP Series 500 and 600 products have up to 4 slots available for optional modules. Slots are numbered from 1 to 4. Numbering of the slots is shown in the following figure.



Slot #2 is located on the top of the option module plugged into Slot #1.

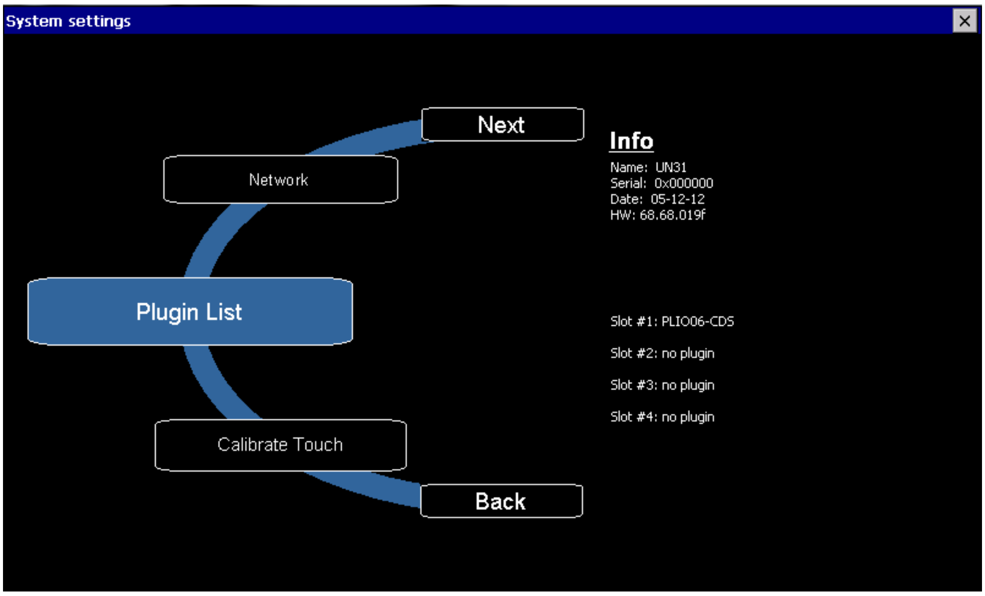
Slot #4 is located on the top of the option module plugged into Slot #3.

Recognizing Plug-In modules mounted on HMI

If correctly installed on the operator panel and recognized by the system, the optional modules are listed into the Plugin List.

The Plugin List is an item of the System Settings rotating menu.

The following figure shows a PLIO06-CDS plugged into Slot #1.

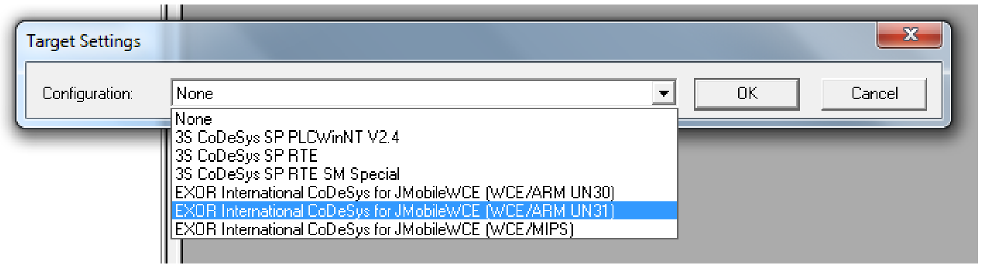


Name: UN31
Serial: 0x000000
Date: 05-12-12
HW: 68.68.019f

Slot #1: PLIO06-CDS
Slot #2: no plugin
Slot #3: no plugin
Slot #4: no plugin

CODESYS V2 Target Settings

When a new PLC program is created, as first step it is necessary to specify the Target by choosing one of the available configurations from the list of available TSP as shown in the following figure.



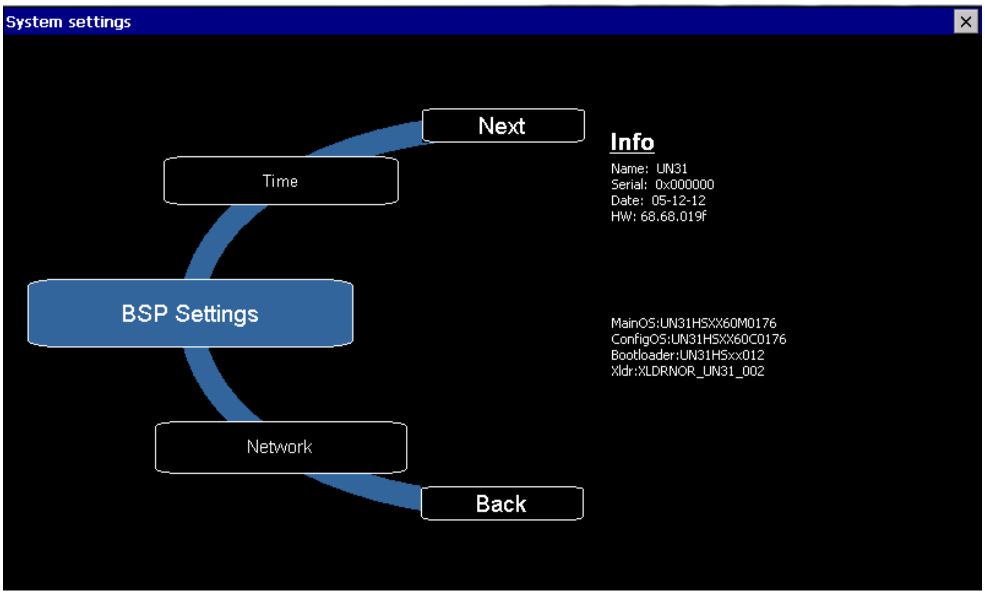
None

None
3S CoDeSys SP PLCWinNT V2.4
3S CoDeSys SP RTE
3S CoDeSys SP RTE SM Special
EXOR International CoDeSys for JMobileWCE (WCE/ARM UN30)
EXOR International CoDeSys for JMobileWCE (WCE/ARM UN31)
EXOR International CoDeSys for JMobileWCE (WCE/MIPS)

For Exor HMI Series products you have the following Target choices:

- EXOR International CODESYS for JMobileWCE (WCE/MIPS)
- EXOR International CODESYS for JMobileWCE (WCE/ARM UN30)
- EXOR International CODESYS for JMobileWCE (WCE/ARM UN31)

The Target selection depends on the type of processor used in the HMI model. The processor type is shown in the System Settings page of the HMI, when BSP Settings voice is selected, into the Info section under Name. See an example in the following figure.



Info

Name: UN31
Serial: 0x000000
Date: 05-12-12
HW: 68.68.019f

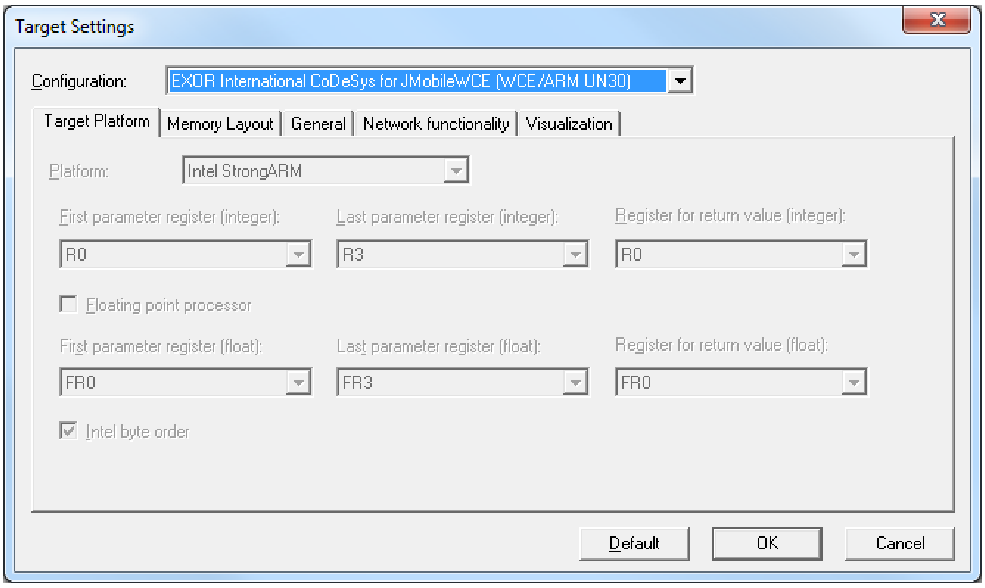
MainOS:UN31HSXX60M0176
ConfigOS:UN31HSXX60C0176
Bootloader:UN31HSxx012
Xldr:XLDRNOR_UN31_002

A list of HMI models with the indication of the Target Settings is given in the following table:

HMI Model	Target Settings	Platform
eTOP406	UN20	600MHz MIPS CPU
eTOP407	UN20	600MHz MIPS CPU
eTOP408	UN20	600MHz MIPS CPU
eTOP410	UN20	600MHz MIPS CPU
eTOP412	UN20	600MHz MIPS CPU
eTOP413	UN20	600MHz MIPS CPU
eTOP415	UN20	600MHz MIPS CPU
eTOP504	UN31	600MHz ARM CPU
eTOP506	UN31	600MHz ARM CPU
eTOP507	UN31	600MHz ARM CPU
eTOP507G	UN31	600MHz ARM CPU
eTOP605	UN31	600MHz ARM CPU
eTOP507M	UN30	1GHz ARM CPU
eTOP507MG	UN30	1GHz ARM CPU
eTOP510	UN30	1GHz ARM CPU
eTOP510G	UN30	1GHz ARM CPU
eTOP512	UN30	1GHz ARM CPU

HMI Model	Target Settings	Platform
eTOP513	UN30	1GHz ARM CPU
eTOP515	UN30	1GHz ARM CPU
eTOP515G	UN30	1GHz ARM CPU
eTOP607M	UN30	1GHz ARM CPU
eTOP610	UN30	1GHz ARM CPU

After selecting the Configuration, the Target Settings dialog box will appear as shown in figure.



Configuration: EXOR International CoDeSys for JMobileWCE (WCE/ARM UN30)

Platform: Intel StrongARM

First parameter register (integer): R0

Last parameter register (integer): R3

Register for return value (integer): R0

☐ Floating point processor

First parameter register (float): FR0

Last parameter register (float): FR3

Register for return value (float): FR0

☑ Intel byte order

These settings can also be reached from the “Resources” tab of the CODESYS V2 Development System.

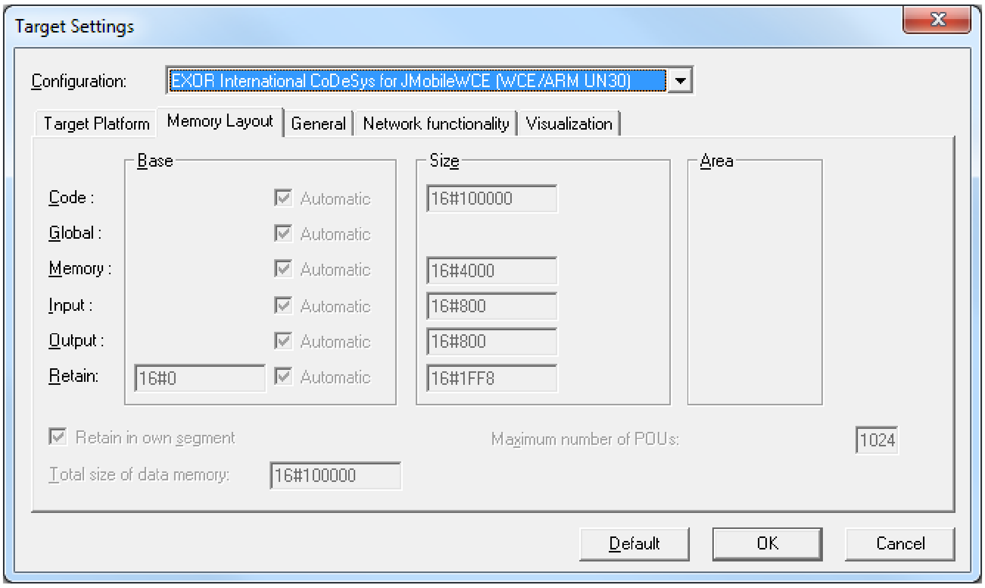
Below you can find basic explanation of the options available in the EXOR International Target Settings. A detailed explanation of all the options is available in the CODESYS V2 online help.

Note: Some options available into the Target Settings are fixed for the EXOR International Target Support Package and cannot be changed by the end user.
Some other options described in the online help are not available for use in the EXOR International Target Support Package.

Target Settings: Memory Layout

The Size table contains the memory size available for Code and memory areas, the size are reported in Byte (Hex). For example the amount of memory for PLC Application code is 1MB. The application files are stored on the Operator panel internal flash memory.

Maximum number of POUs specifies the max number of POU allowed in a project



Target Settings

Configuration: EXOR International CoDeSys for JMobileWCE (WCE/ARM UN30)

Target Platform | Memory Layout | General | Network functionality | Visualization

	Base		Size	Area
Code :		☑ Automatic	16#100000	
Global :		☑ Automatic		
Memory :		☑ Automatic	16#4000	
Input :		☑ Automatic	16#800	
Output :		☑ Automatic	16#800	
Retain:	16#0	☑ Automatic	16#1FF8	

☑ Retain in own segment

Maximum number of POUs: 1024

Total size of data memory: 16#100000

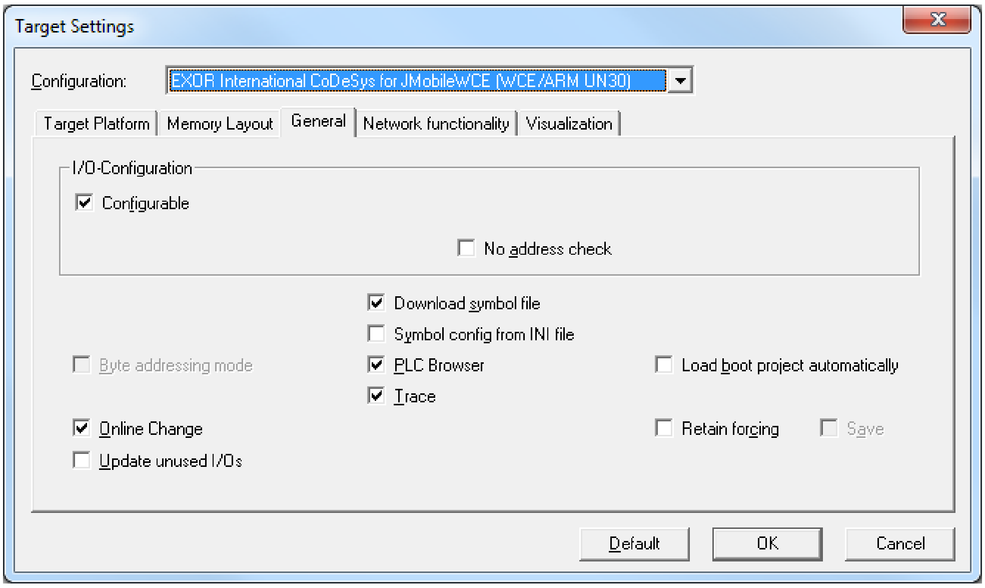
Target Settings: General

The next figure shows the General tab. Among the options available the most commonly used options are:

Download symbol file: when set, symbol file is downloaded into PLC. This is required for the automatic symbol file upload procedure. We suggest to set this option.

Load bootproject automatically: when set, the boot project is automatically created and downloaded into the PLC together with application.

For all the other options we suggest to refer to the CODESYS manual.



Target Settings

Configuration: EXOR International CoDeSys for JMobileWCE (WCE/ARM UN30)

Target Platform | Memory Layout | General | Network functionality | Visualization

I/O-Configuration

☑ Configurable

☐ No address check

☑ Download symbol file

☐ Symbol config from INI file

☐ Byte addressing mode

☑ PLC Browser

☐ Load boot project automatically

☑ Trace

☑ Online Change

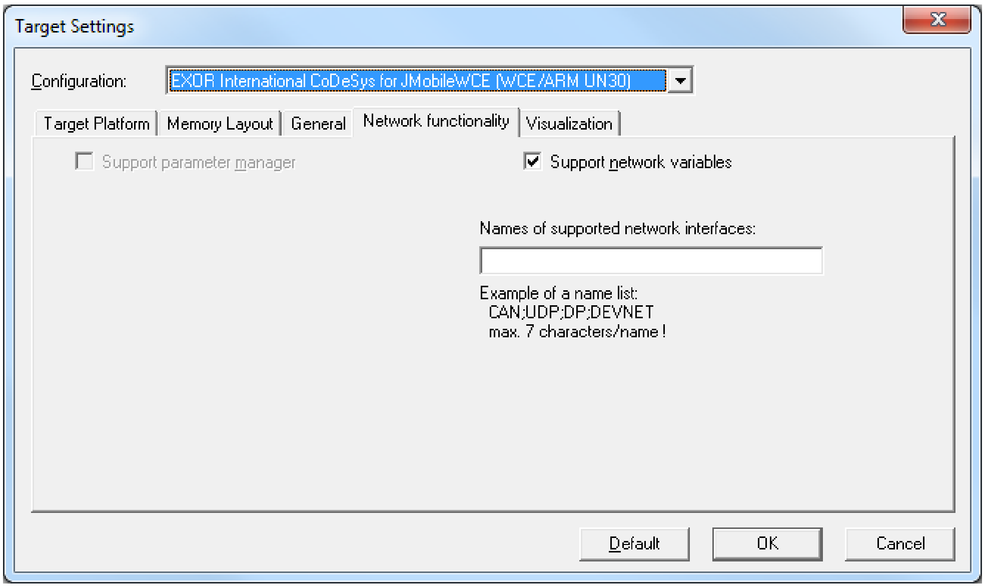
☐ Retain forcing

☐ Save

☐ Update unused I/Os

Target Settings: Network functionality

Network functionality tab enable the use of Network variables, the system allows exclusively the use of UDP network variables.


Target Settings

Configuration: EXOR International CoDeSys for JMobileWCE (WCE/ARM UN30)

□ Support parameter manager

☑ Support network variables

Names of supported network interfaces:

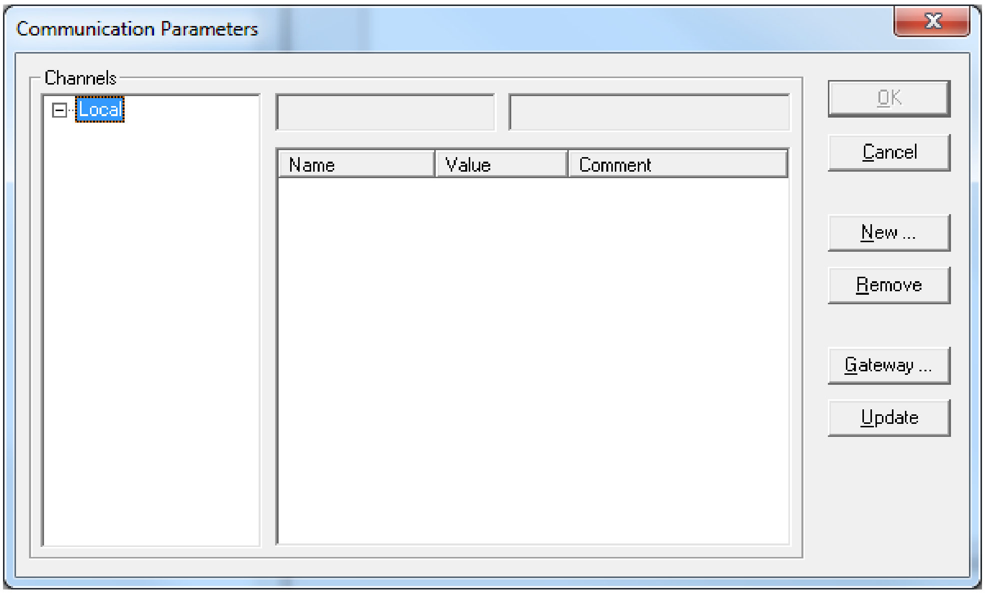
Example of a name list:
CAN;UDP;DP;DEVNET
max. 7 characters/name !

Note: Visualization is not supported from the Exor CODESYS Runtime version. The corresponding tab in Target Settings has no use.

Download PLC application

Before to go online and download the PLC application it is necessary to set Communication Parameters selecting in the Development System: Online > Communication Parameters.

The first time this dialog is opened, the user will be requested to specify the channel for the connection to the PLC runtime.


Communication Parameters

Channels

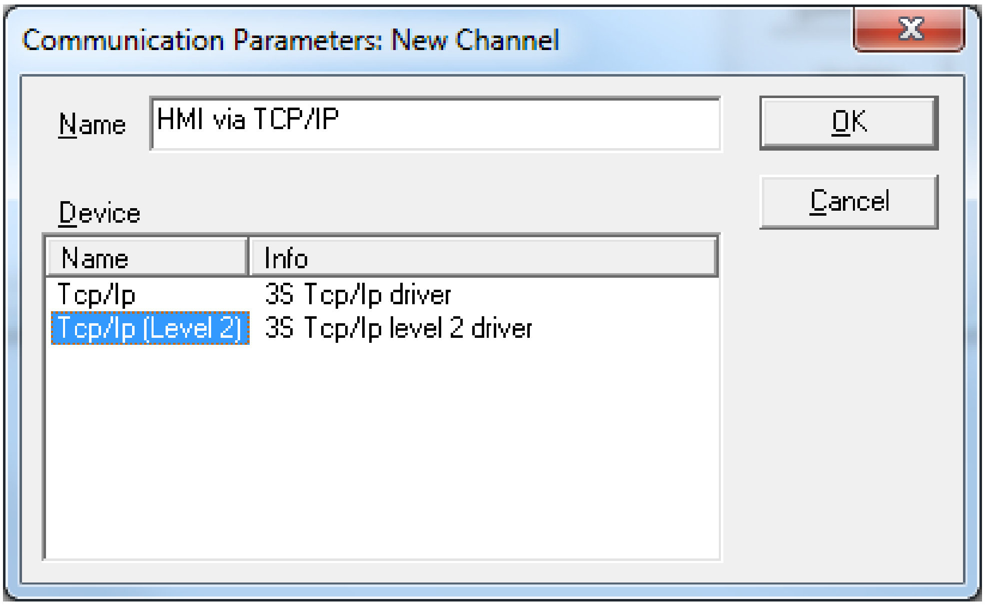
Local

Name	Value	Comment

Click on “New…” button to create a new Communication Channel.

When creating a new Channel, you have to define the type and all the relevant parameters in the dialog box.

Communication is possible by Ethernet using level 2 driver, selecting “Tcp/Ip (Level 2)” from the device list:



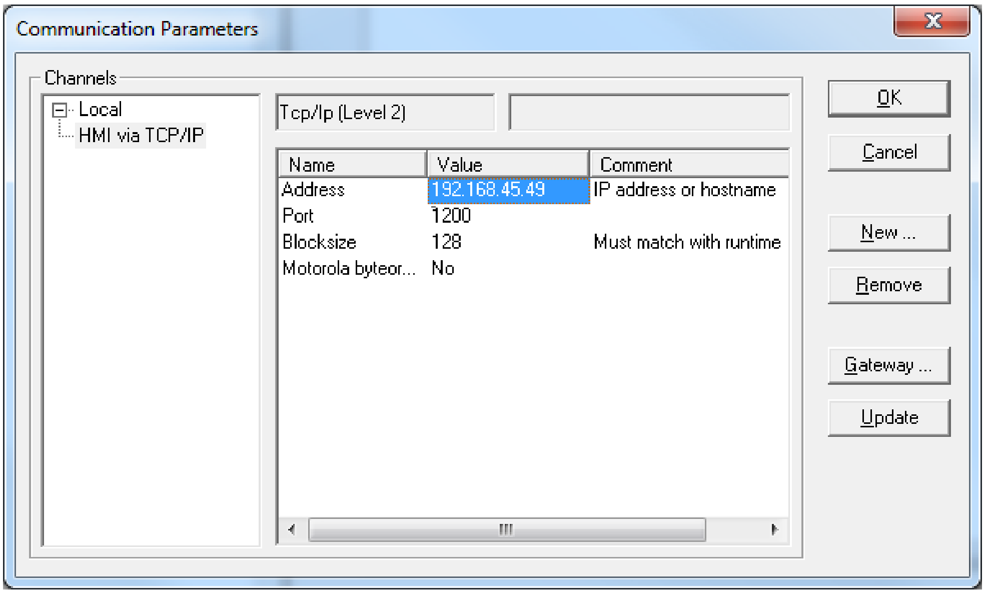
Name: HMI via TCP/IP

Device

Name	Info
Tcp/Ip	3S Tcp/Ip driver
Tcp/Ip (Level 2)	3S Tcp/Ip level 2 driver

Once confirmed, the new Channel is added to the Channels list into the Communication Parameters dialog. Selecting the new Channel it is possible to specify the IP address of the PLC runtime that must be entered in the “Address” parameter; see the following figure as example.

The other parameters: Port, Blocksize and Motorola byteorder are correct for operation with Exor and do not require to be modified.



Channels
- Local
  - HMI via TCP/IP

Tcp/Ip (Level 2)

Name	Value	Comment
Address	192.168.45.49	IP address or hostname
Port	1200	
Blocksize	128	Must match with runtime
Motorola byteor...	No	

Then to download the PLC application it is necessary to enter Online mode;

to do this select Online > Login command from CODESYS workbench. The system will verify the presence of an application into the PLC, asking the confirmation before proceeding with the download.

After the download operation the CODESYS application is stored into a volatile memory, this means that if the operator panel is restarted, the CODESYS application is lost.

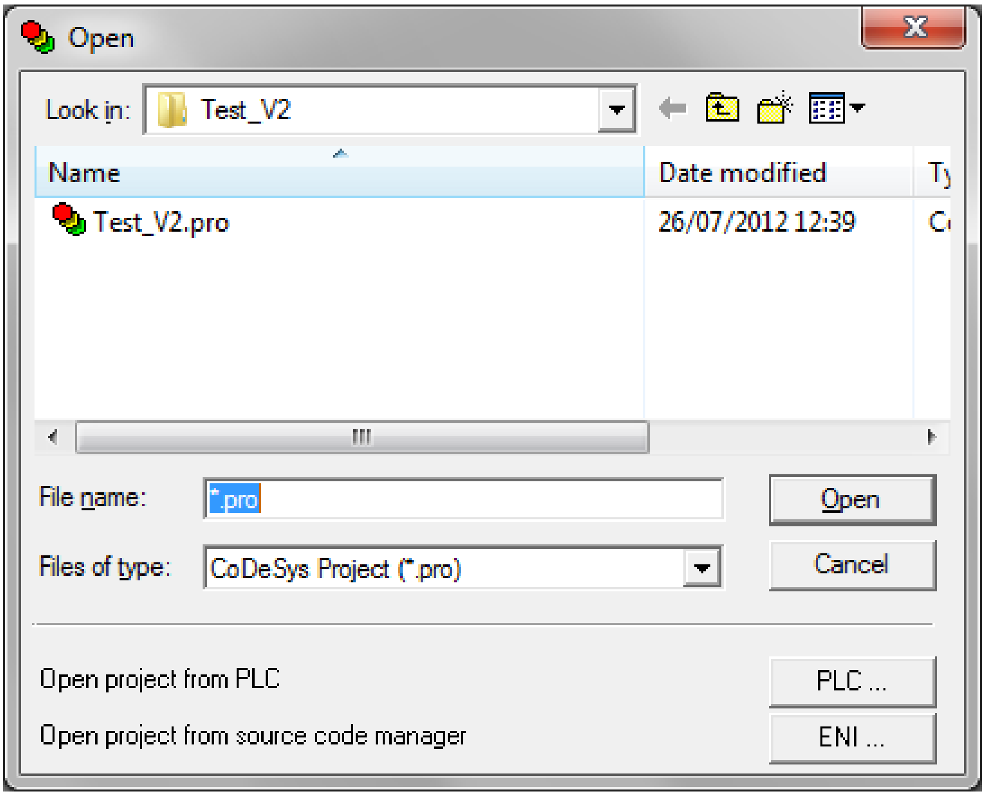
To store the application into the non-volatile flash memory of the panel it is necessary, while Online, to launch the command “Online > Create Boot Project”.

Note: It is possible to download the Boot Project automatically at project download by selecting the “Load bootproject automatically” option available among the Target Settings options. For more information please refer to the Target Settings chapter.

PLC Project Upload

PLC project upload is supported in the Exor implementation.

To upload a PLC project from the HMI, select the command Open from the File menu and click on the “PLC” button to specify the source of the open action.



When opening a program from PLC, CODESYS V2 requires the specification of the Target Settings.

PLC Project upload can be executed only if the project source code has been downloaded to the target device.

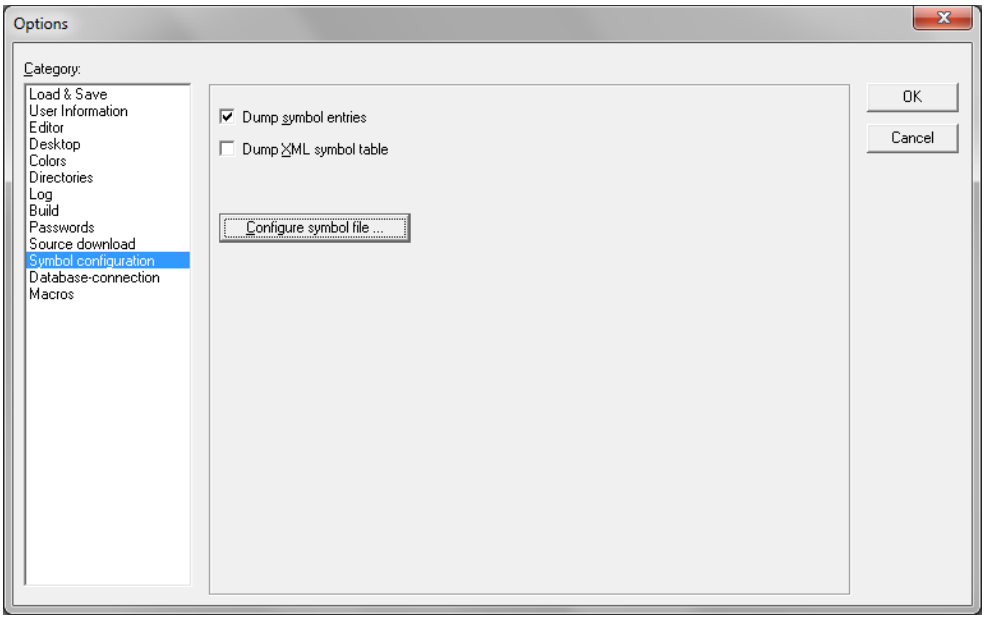
The “Source code download” command is available from the “Online” menu of the CODESYS V2 Development System.

Symbol File configuration with CODESYS V2 Development System

The CODESYS V2 Development System can create a list of all the variables available into the PLC program in form of a file with “.sym” extension. This file is generated when the application is Build and it is stored in the application folder.

Going to “Project > Options…” it’s possible to configure symbol file.

To enable the generation of the Symbol file check the option “Dump symbol entries” as shown in the following figure.

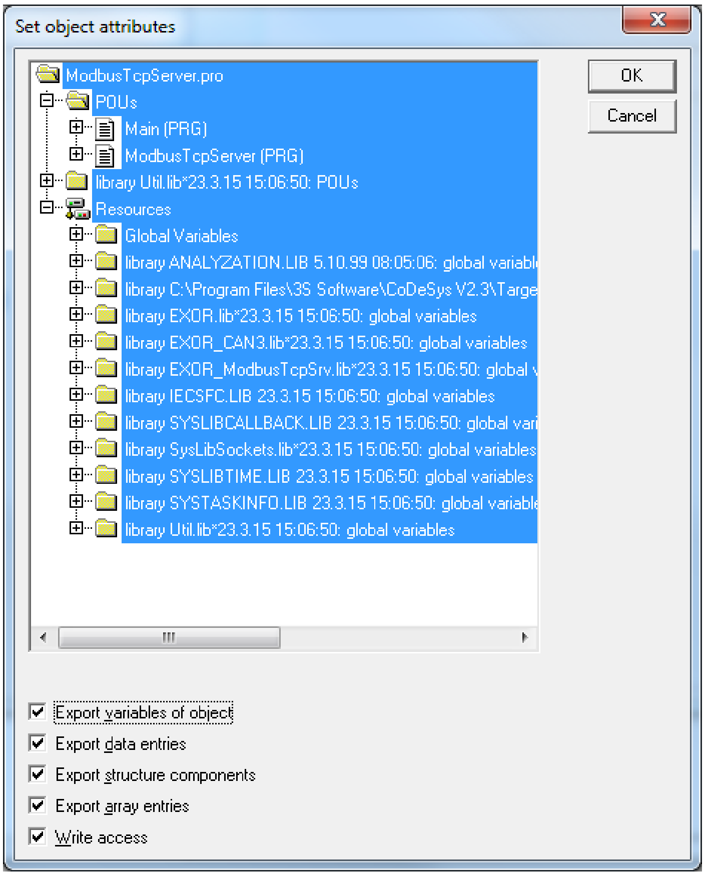


☑ Dump symbol entries

□ Dump XML symbol table

To make sure that Symbols are created for all variables in all POUs, for Global Variables and for variables defined in the PLC Configuration click on the “Configure symbol file…” button.

Check the option “Export variables of object” in the Set Object Attributes dialog box, as shown in the following figure.



ModbusTcpServer.pro
POUs
Main (PRG)
ModbusTcpServer (PRG)
library Util.lib*23.3.15 15:06:50: POUs
Resources
Global Variables
library ANALYZATION.LIB 5.10.99 08:05:06: global variabl
library C:\Program Files\3S Software\CoDeSys V2.3\Targe
library EXOR.lib*23.3.15 15:06:50: global variables
library EXOR_CAN3.lib*23.3.15 15:06:50: global variables
library EXOR_ModbusTcpSrv.lib*23.3.15 15:06:50: global v
library IECSFC.LIB 23.3.15 15:06:50: global variables
library SYSLIBCALLBACK.LIB 23.3.15 15:06:50: global vari
library SysLibSockets.lib*23.3.15 15:06:50: global variables
library SYSLIBTIME.LIB 23.3.15 15:06:50: global variables
library SYSTASKINFO.LIB 23.3.15 15:06:50: global variabl
library Util.lib*23.3.15 15:06:50: global variables

☑ Export variables of object

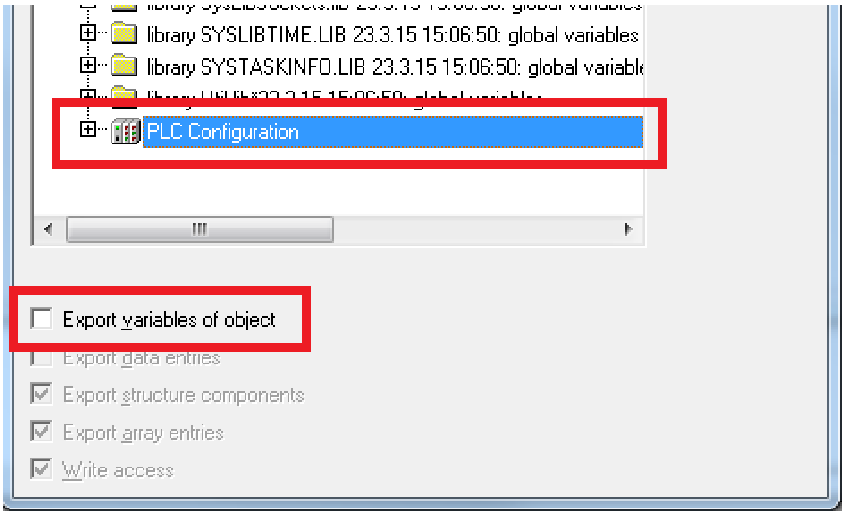
☑ Export data entries

☑ Export structure components

☑ Export array entries

☑ Write access

In some cases, duplication of symbols for variables associated to integrated I/O modules in the “.sym” file may be experienced. To remove the duplication selected the “PLC Configuration” voice from the objects list and uncheck the option “Export variables of object”.



library SYSLIBTIME.LIB 23.3.15 15:06:50: global variables

library SYSTASKINFO.LIB 23.3.15 15:06:50: global variable

PLC Configuration

☐ Export variables of object

☐ Export data entries

☑ Export structure components

☑ Export array entries

☑ Write access

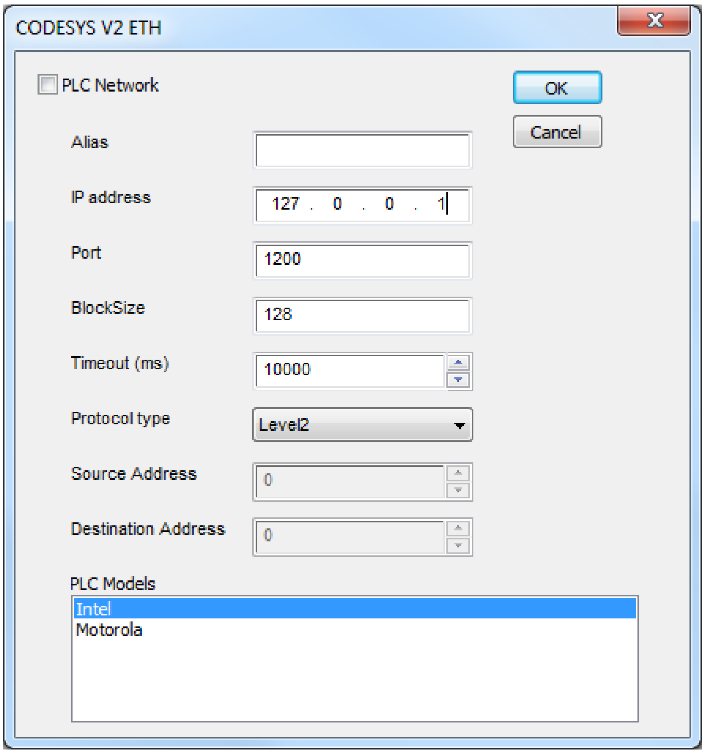
Please refer to CODESYS V2 documentation for additional information.

Communication Setup in JMobile Studio

JMobile communicates with the internal CODESYS V2 runtime using the CODESYS V2 ETH protocol. Localhost 127.0.0.1 should be entered in the IP Address parameter, this identifies the PLC as internal CODESYS V2 runtime.

BlockSize parameter can be increased up to 1024.

The system supports Level 2 and Level 4 protocol types for the communication with the CODESYS runtime.



CODESYS V2 ETH

☐ PLC Network

Alias	
IP address	127 . 0 . 0 . 1
Port	1200
BlockSize	128
Timeout (ms)	10000
Protocol type	Level2
Source Address	0
Destination Address	0

PLC Models

Intel

Motorola

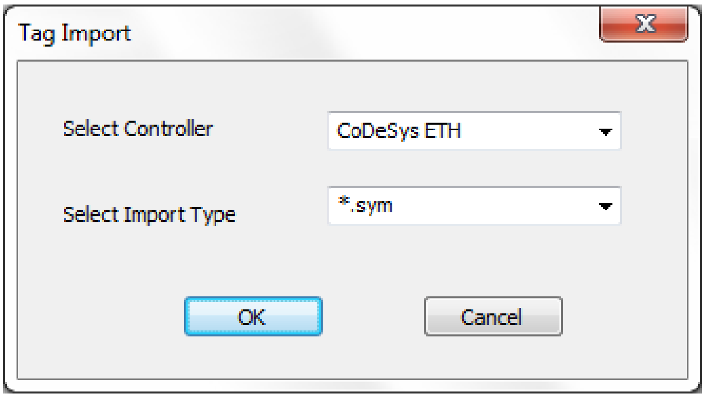
Additional informations regarding the CODESYS V2 ETH driver are available in JMobile help Communication drivers section.

Import Symbol Files in JMobile Studio

JMobile Tag Editor supports direct import of CODESYS V2 symbol file for an automatic and faster definition of the Tags.

Use the command “Import Tags” in JMobile Tag Editor to import the symbols generated by the Development System.

Select *.sym as Import Type in the JMobile Tag Import dialog, as shown in the following figure.



Select Controller: CoDeSys ETH

Select Import Type: *.sym

CODESYS V2 Development System generates a new version of the “.sym” file each time the PLC project is built.

CODESYS V2 Ethernet communication driver supports automatic symbol file (SDB) upload from the PLC; the HMI can upload the symbol table from the PLC using the communication protocol.

Any change in the tag offset information due to a new compilation of the PLC program does not require importing again the symbol file.

Symbol file must be imported again when:

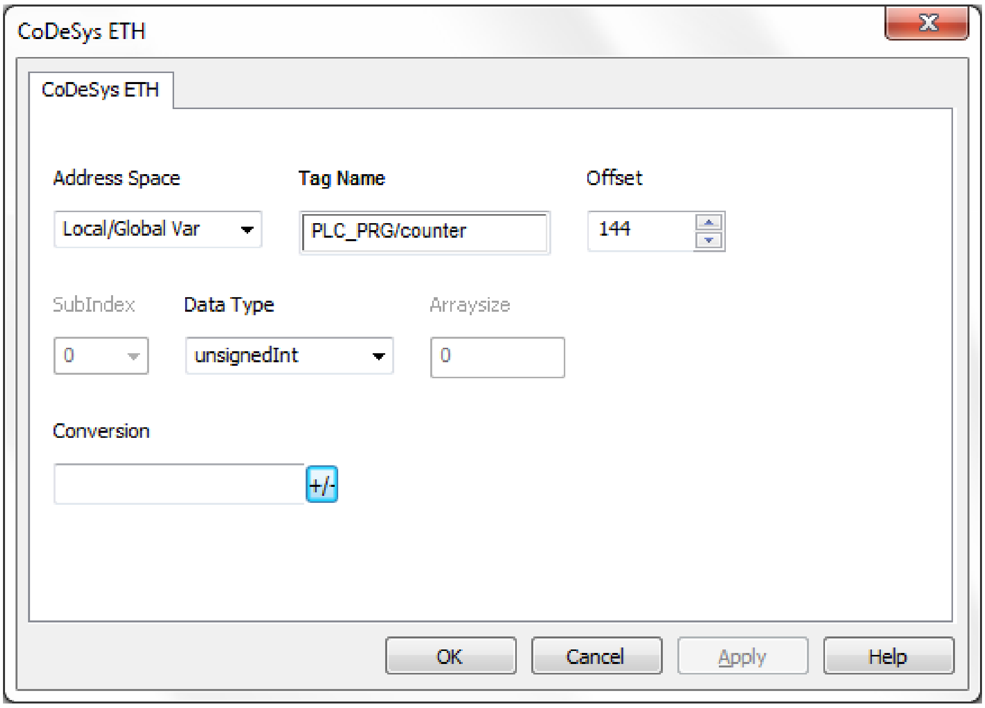
- tags have been renamed
- tags have changed data format or
- new tags have been added.

Tag Addressing

Tag addressing informations are visible in the Tag Address dialog. See an example in the following figure.

The following Address Spaces options are available:

- PLC Memory
- PLC Memory Bit
- PLC Input
- PLC Input Bit
- PLC Output
- PLC Output Bit
- Retentive Memory
- Local/Global Var
- Local/Global Var Bit
- Local/Global Var(5)
- IP Override IP
- ProjectId



CoDeSys ETH

Address Space: Local/Global Var

Tag Name: PLC_PRG/counter

Offset: 144

SubIndex: 0

Data Type: unsignedInt

Arraysize: 0

Conversion:

Retain variables are in segment 3, Global and POU (Program Organization Unit) local variables without direct address are in the subsequent segments.

The reference to variables in the CODESYS system consists of “POUref” (the segment), Offset and size.

Maximum number of connections

CODESYS V2 internal PLC supports up to 4 concurrent connections to other devices.

Communication Diagnostic

JMobile reports communication diagnostic information also for the communication with the internal PLC runtime. To display communication diagnostic information use the proper System Variables or the System Logger. For further information please refer to JMobile Studio help.

Configuring I/O

CODESYS V2 implementation for eTOP Series 500 and 600 products support configuration of integrated and remote I/O.

Remote I/O can be based on CANOpen or Modbus Ethernet.

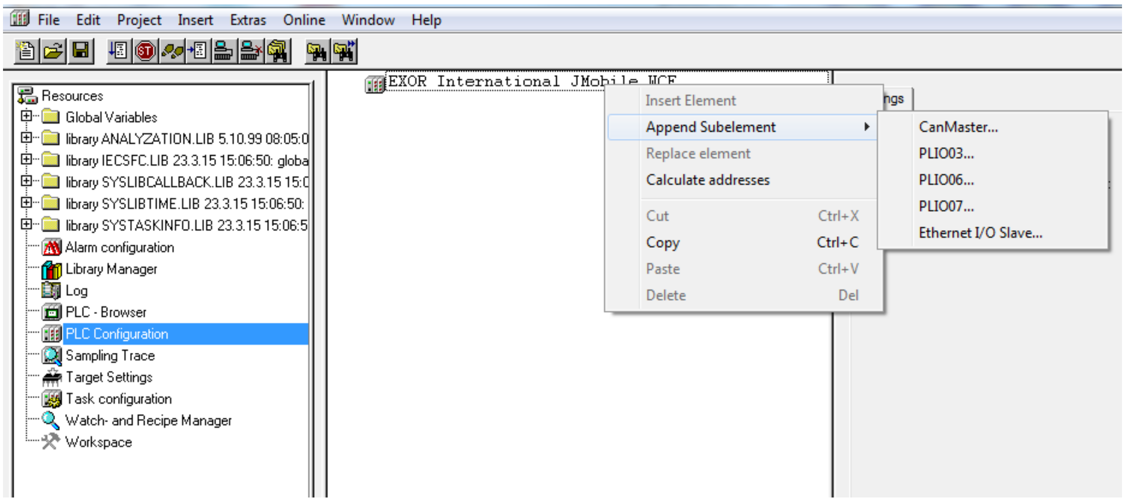
Simple configurations using CANOpen bus for remote I/O can be realized adding the CAN optional module PLCM01 to the HMI.

Modbus Ethernet I/O uses the integrated ethernet port of the HMI as interface to the network. In this case you can use PLCM05-CDS, a module that enable the CODESYS V2 license but without bus interface.

Integrated I/O systems are based on optional plug-in modules named PLIO.

To configure an I/O optional module, select “Resources” tab into CODESYS V2 Development System then click on “PLC Configuration”. Use the “Insert Element/Append Subelement” commands from Insert menu or from context menu on right click on the Root voice of configuration tree. The list of available elements will appear as sub-menu.

Choose the I/O type from the list to add it to the current PLC configuration.



The parameters of all integrated I/O modules are collected in single tab of the PLC Configuration tree.

PLIO03

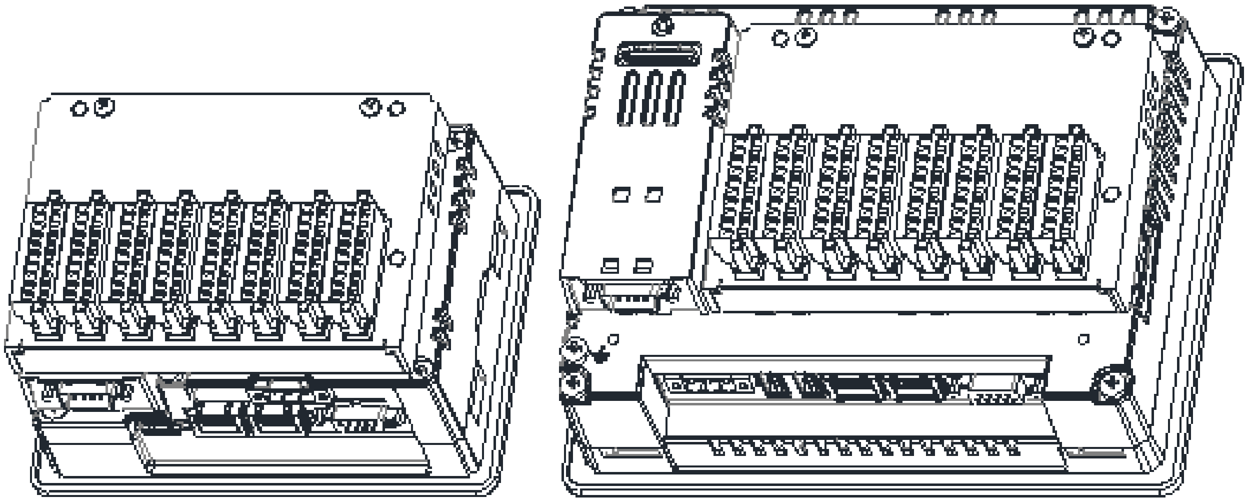
PLIO03 is a multifunction I/O module.

Due to the high level of integration and configurability, PLIO03 offers a “one-board solution” for most typical I/O configurations in simple applications.

PLIO03 specifications are the following:

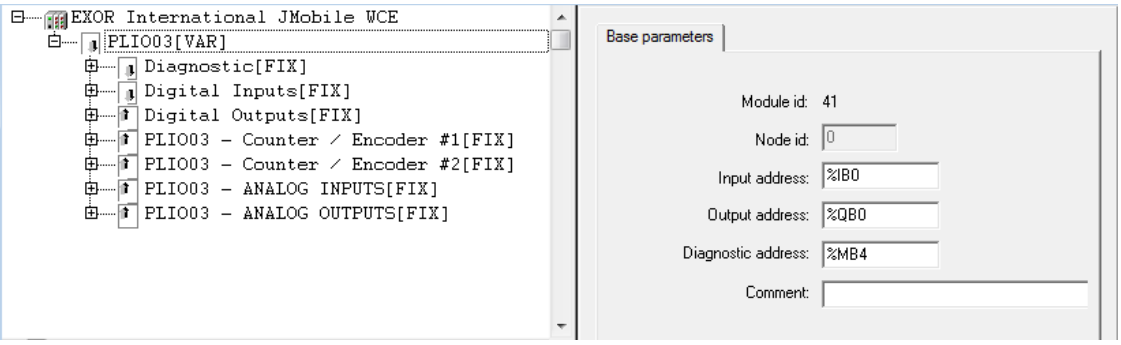
- 20 optically isolated digital inputs; configurable as counter/encoder channels
- 12 optically isolated digital outputs
- 4 non-isolated (4 differential or 8 single ended channels) 12bit analog inputs configurable for voltage, current, resistance or temperature measurement
- 4 non-isolated 12 bit analog outputs configurable for voltage or current
- 1 dedicated PT100 channel-input for general usage or compensation of thermocouples.

The following figure shows a configuration for eTOP504 and eTOP507, using the PLIO03 module.



Note: eTOP504 and eTOP605 panel models require the use of a Bus extender module for the installation of the PLIO03 I/O module.

After including the PLIO03 module in the PLC configuration, as shown in the following figure, Specific configuration steps must be performed for different parts of the module.



```
EXOR International JMobile WCE
  PLIO03[VAR]
    Diagnostic[FIX]
    Digital Inputs[FIX]
    Digital Outputs[FIX]
    PLIO03 - Counter / Encoder #1[FIX]
    PLIO03 - Counter / Encoder #2[FIX]
    PLIO03 - ANALOG INPUTS[FIX]
    PLIO03 - ANALOG OUTPUTS[FIX]
```

Module id: 41

Node id: 0

Input address: %IB0

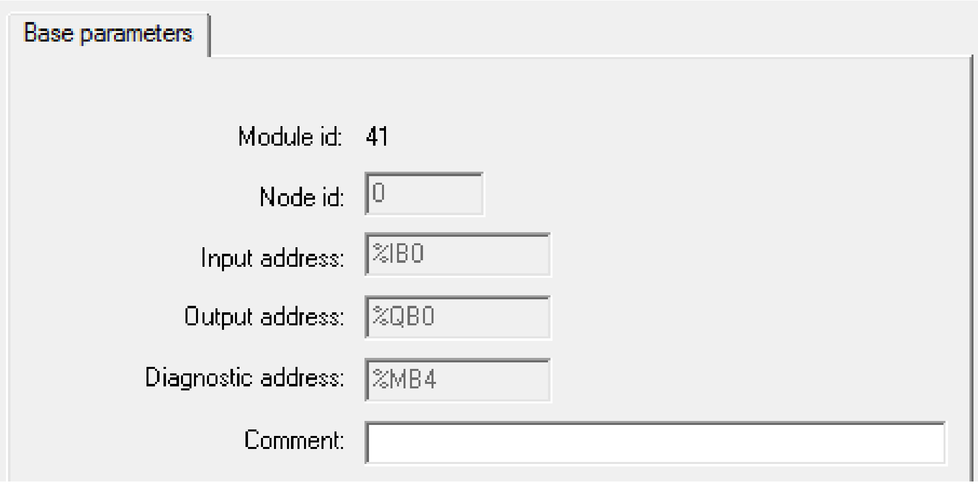
Output address: %QB0

Diagnostic address: %MB4

Comment:

Base parameters

The following figure shows the Base Parameters tab for integrated I/O modules.



Module id: 41

Node id: 0

Input address: %IB0

Output address: %QB0

Diagnostic address: %MB4

Comment:

Element	Description
Module id	CODESYS internal identifier used to recognize the board; it is a read only parameter.
Node id	CODESYS internal identifier assigned by the Development System depending on the order of the board in the PLC Configuration layout; it is a read only parameter.
Input address	Starting address for input information
Output address	Starting address for output information
Diagnostic address	Memory address for the storage of diagnostic information of the I/O module

Configuring Digital Inputs

PLIO03 includes 20 programmable digital inputs.

Normal use of digital inputs does not require any additional configuration.

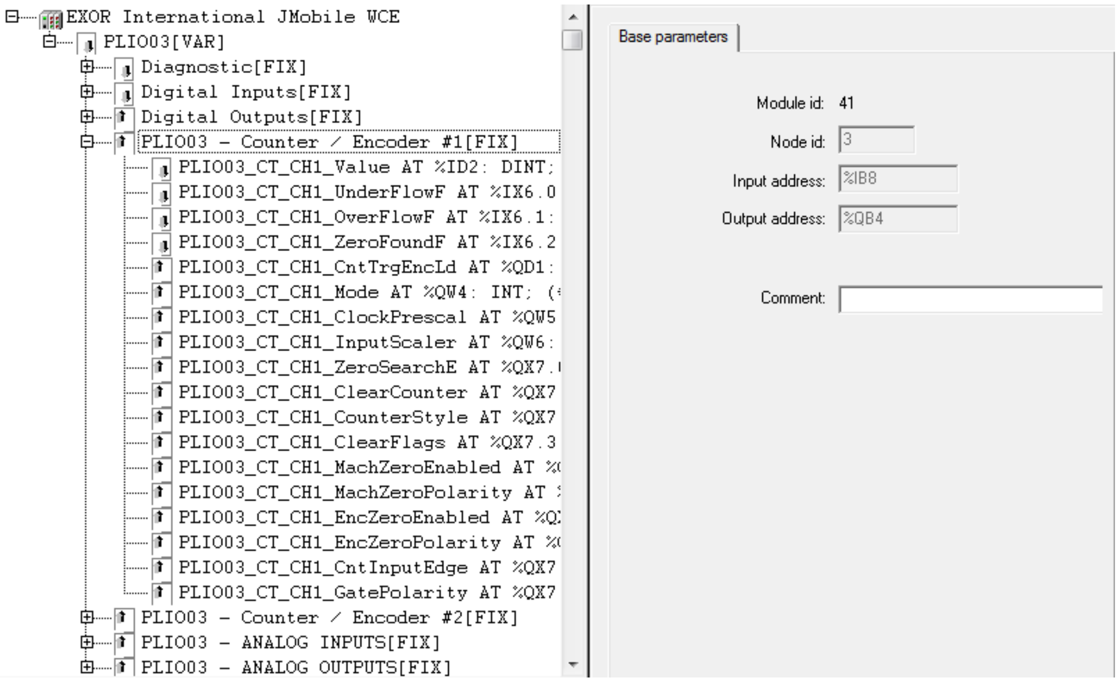
Counter/Encoder Inputs

PLIO03 includes 2 programmable Counter/Encoder channels.

Each Counter/Encoder channel is associated to a group of 4 digital inputs. When a Counter/Encoder channel is enabled, the associated digital inputs cannot be used as normal inputs. In case all the 2 Counter/Encoder channels are enabled, only 12 digital inputs are still available for normal operation. Please refer to PLIO03 hardware manual for detailed specifications and wiring diagrams.

Configuration of Counter/Encoder inputs must be done in the PLC program, based on the parameters defined for the PLIO03 module.

The following figure shows the list of PLIO03 Counter/Encoder #1 parameters as it appears in the PLC Configuration.



```
⊟ EXOR International JMobile WCE
   ⊟ PLIO03[VAR]
      ⊞ Diagnostic[FIX]
      ⊞ Digital Inputs[FIX]
      ⊞ Digital Outputs[FIX]
      ⊟ PLIO03 - Counter / Encoder #1[FIX]
           PLIO03_CT_CH1_Value AT %ID2: DINT;
           PLIO03_CT_CH1_UnderFlowF AT %IX6.0
           PLIO03_CT_CH1_OverFlowF AT %IX6.1:
           PLIO03_CT_CH1_ZeroFoundF AT %IX6.2
           PLIO03_CT_CH1_CntTrgEncLd AT %QD1:
           PLIO03_CT_CH1_Mode AT %QW4: INT; (
           PLIO03_CT_CH1_ClockPrescal AT %QW5
           PLIO03_CT_CH1_InputScaler AT %QW6:
           PLIO03_CT_CH1_ZeroSearchE AT %QX7.
           PLIO03_CT_CH1_ClearCounter AT %QX7
           PLIO03_CT_CH1_CounterStyle AT %QX7
           PLIO03_CT_CH1_ClearFlags AT %QX7.3
           PLIO03_CT_CH1_MachZeroEnabled AT %
           PLIO03_CT_CH1_MachZeroPolarity AT
           PLIO03_CT_CH1_EncZeroEnabled AT %Q
           PLIO03_CT_CH1_EncZeroPolarity AT %
           PLIO03_CT_CH1_CntInputEdge AT %QX7
           PLIO03_CT_CH1_GatePolarity AT %QX7
      ⊞ PLIO03 - Counter / Encoder #2[FIX]
      ⊞ PLIO03 - ANALOG INPUTS[FIX]
      ⊞ PLIO03 - ANALOG OUTPUTS[FIX]
```

Module id: 41

Node id: 3

Input address: %IB8

Output address: %QB4

Comment:

The input signals associated to each Counter/Encoder channel have the following meaning:

Name	Data Type	Description
Value	DINT	Counter/frequency value
UnderFlowF	BOOL	Underflow flag
OverFlowF	BOOL	Overflow flag
ZeroFoundF	BOOL	Zero found flag

Board parameters have different options that can be selected at runtime using the PLC program.

Description of parameters is given in the following table.

Name	Data Type	Value	Description
Mode	INT	0	Counter/Encoder not active
		1	Quadrature incremental encoder counter
		2	Normal Counting
		3	Gated Counting
		4	Frequency Measurement
ClockPrescal	INT		Clock frequency prescaler
InputScaler	INT		Input frequency prescaler
CntTrgEncLd	DINT		Target value in Counter mode
ZeroSearchE	BOOL		Enable zero search when in encoder mode. When true the counter resets to zero. ZeroFoundF reports when zero has been reached.
ClearCounter	BOOL		Reset counter value when in counter mode
CounterStyle	BOOL		Valid in counter mode, when true the count value resets when the counter reaches the target value
ClearFlags	BOOL		Reset overflow and underflow flags
MachZeroEnabled	BOOL	false	Corresponding terminal block is used as normal digital input
		true	Corresponding terminal block is used as Machine Zero Input
MachZeroPolarity	BOOL	false	Means Input is active LOW
		true	Means Input is active HIGH
EncZeroEnabled	BOOL	false	Corresponding terminal block is used as normal digital input
		true	Corresponding terminal block is used as Encoder Zero Input
EncZeroPolarity	BOOL	false	Means Input is active LOW
		true	Means Input is active HIGH
CntInputEdge	BOOL	false	Active count input edge negative
		true	Input edge positive
GatePolarity	BOOL	false	Defines signal level for Gate LOW
		true	Signal level for Gate HIGH

When the input is configured for use with an incremental encoder, then the Encoder Zero Search procedure is usually performed at power-up. The PLC program should start this procedure enabling the Zero Search using the parameterZeroSearchE for the selected channel.

Frequency measurement (Mode 4) is based on the following formula:

$$\text{Frequency=}\quad\frac{\text{Internal Clock Frequency}\cdot\text{Number of counts}\cdot(\text{InputScaler}+1)}{\text{Number of counts Internal Clock}\cdot(\text{ClockPresc}+1)}$$

The I/O driver calculates the frequency that is returned in the input signal Value.

One frequency measurement cycle terminates when there is at least one input count and at least 65536 counts in the internal clock.

As an example, to cover the frequency range 1Hz to 20KHz you have to set:

Range	ClockPresc	InputScaler
1Hz to 2Hz	0	1
2Hz to 20KHz	0	0

Configuring Digital Outputs

PLIO03 includes 12 digital outputs.

Use of digital outputs does not require any additional configuration.

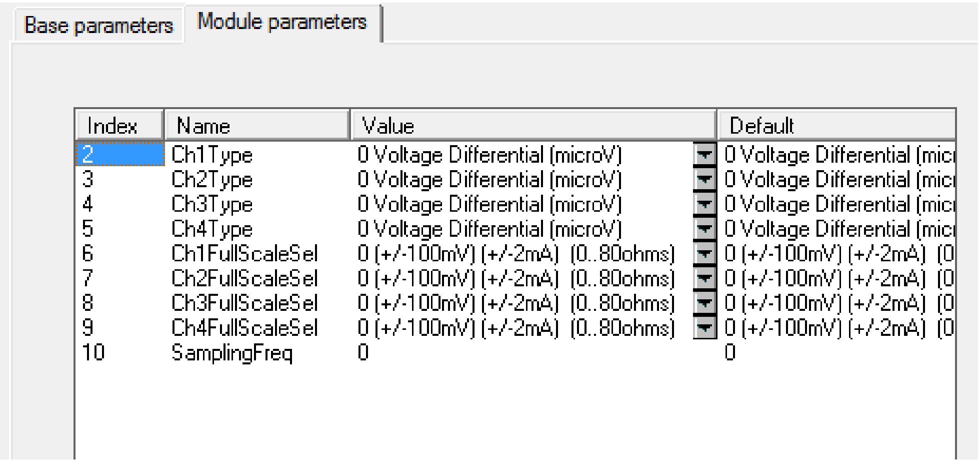
Configuring Analog Inputs

PLIO03 includes 4 differential programmable analog input channels.

PLIO03 Analog Input channels have specific parameters that must to be properly configured according to the operation mode requested for each channel: the operating mode and the full-scale range.

Each channel can be configured as single-ended voltage measurement resulting in having 8 voltage analog inputs.

The following figure shows the configuration interface of the Analog Inputs. Note that 9 analog values are produced by the module.



Index	Name	Value	Default
2	Ch1Type	0 Voltage Differential (microV)	0 Voltage Differential (mici
3	Ch2Type	0 Voltage Differential (microV)	0 Voltage Differential (mici
4	Ch3Type	0 Voltage Differential (microV)	0 Voltage Differential (mici
5	Ch4Type	0 Voltage Differential (microV)	0 Voltage Differential (mici
6	Ch1FullScaleSel	0 (+/-100mV) (+/-2mA) (0..80ohms)	0 (+/-100mV) (+/-2mA) (0
7	Ch2FullScaleSel	0 (+/-100mV) (+/-2mA) (0..80ohms)	0 (+/-100mV) (+/-2mA) (0
8	Ch3FullScaleSel	0 (+/-100mV) (+/-2mA) (0..80ohms)	0 (+/-100mV) (+/-2mA) (0
9	Ch4FullScaleSel	0 (+/-100mV) (+/-2mA) (0..80ohms)	0 (+/-100mV) (+/-2mA) (0
10	SamplingFreq	0	0

The following table shows the different meaning assumed by the 9 values produced by the Analog Input module depending on the measurement mode selected with the configuration parameters.

	Module Signal	Mode			
		Voltage Differential	Voltage Single-ended	Resistance 2/3 wires	Current
1	AI1_CH1	Value	Value	Value	Value
2	AI1_CH2	-	Value	-	-
3	AI2_CH1	Value	Value	Value	Value
4	AI2_CH2	-	Value	-	-
5	AI3_CH1	Value	Value	Value	Value
6	AI3_CH2	-	Value	-	-
7	AI4_CH1	Value	Value	Value	Value
8	AI4_CH2	-	Value	-	-
9	COMP	Resistance	-	-	-

Channel 9 is an input channel dedicated to cold junction compensation of thermocouples. Please refer to the hardware description manual for additional details.

Configuring Channel Measurement Mode

Each channel is independently programmable; six different types of measurements can be executed. The type of measure is defined by the ChxType setting, the following table shows the values that can be configured.

Parameter value	Measurement Mode	Unit
0	Voltage Differential	µV
1	Voltage Single Ended	µV
2	Current	µA
3	Resistance 2 wires	mΩ
4	Resistance 3 wires	mΩ
5	Resistance 4 wires	mΩ
6	Voltage Thermocouple Differential	µV
7	Voltage Thermocouple Single Ended	µV
8	Current (0-20)	µA
9	PT100 2 wires	mΩ
10	PT100 3 wires	mΩ
11	PT100 4 wires	mΩ

Configuring Channel Full Scale

Full Scale parameter value behavior changes depending on the ChxType value set for the Channel, for ChxType values from 0 to 5 the Full Scale parameter can be set at programming time, the available values are reported in the following table.

Parameter value	**Full Scale Value**		
	Voltage	**Current**	**Resistance**
0	±100mV	±2mA	0 - 80Ω
1	±500mV	±10mA	0 - 400Ω
2	±1V	±20mA	0 - 900Ω
3	±5V	±20mA	0 - 8KΩ
4	±10V	±20mA	0 - 1MΩ
5	1V	±20mA	0 - 900Ω
6	0 – 10V	±20mA	0 - 1MΩ
7	PT100		
8	Thermocouple		

For ChxType values from 6 to 8 the Full Scale parameter value is set as default by the system.

Parameter value	**Measurement Mode**	**Full Scale Value**
6	Voltage Thermocouple Differential	+/- 100mV
7	Voltage Thermocouple Single Ended	+/- 100mV
8	Current (0-20)	0 - 1V

For ChxType values from 9 to 11 the Full Scale parameter can be modified during runtime using the EXOR_IO_CTRL function. The table below reports the Full Scale values available, by default the Full Scale parameter value is 0.

Parameter value	**Full Scale Value**
0	0-157Ω
1	0-530Ω
2	0-1020Ω
3	0-8800Ω
4	0-10MΩ

SamplingFreq Parameter

The parameter SamplingFreq available into may be used to synchronize analog measurement to reduce powerline noise. The value 0 for SamplingFreq means free running measurement. The value 50 will make measurements synchronized with powerline at 50Hz. This option is useful with high-gain measurements as in thermocouple channels

Runtime Change of Channels Settings Using EXOR_IO_CTRL

Configuration parameters of the Analog Inputs can be changed at runtime using the EXOR_IO_CTRL function with the following Function Codes:

Function Code	Argument	Description
0	0 ÷8	Set Full Scale for Channel 1
1	0 ÷8	Set Full Scale for Channel 2
2	0 ÷8	Set Full Scale for Channel 3
3	0 ÷8	Set Full Scale for Channel 4
4	0 ÷11	Set Analog Input Mode for Channel 1
5	0 ÷11	Set Analog Input Mode for Channel 2
6	0 ÷11	Set Analog Input Mode for Channel 3
7	0 ÷11	Set Analog Input Mode for Channel 4

Other Function Codes are reserved must not be used.

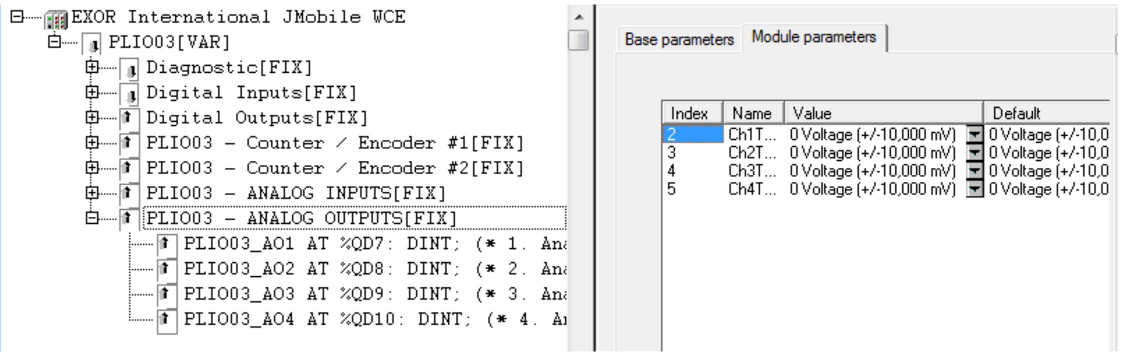
Configuring Analog Outputs

PLIO03 includes 4 programmable analog output channels.

PLIO03 Analog Outputs have several parameters that must be properly configured according to the operation mode you need for each channel.

For each of the 4 channels it is required to specify the operating mode and the full-scale range.

The following figure shows the configuration of the 4 Analog Outputs channels.



```
EXOR International JMobile WCE
    PLIO03[VAR]
        Diagnostic[FIX]
        Digital Inputs[FIX]
        Digital Outputs[FIX]
        PLIO03 - Counter / Encoder #1[FIX]
        PLIO03 - Counter / Encoder #2[FIX]
        PLIO03 - ANALOG INPUTS[FIX]
        PLIO03 - ANALOG OUTPUTS[FIX]
            PLIO03_AO1 AT %QD7: DINT; (* 1. Ana
            PLIO03_AO2 AT %QD8: DINT; (* 2. Ana
            PLIO03_AO3 AT %QD9: DINT; (* 3. Ana
            PLIO03_AO4 AT %QD10: DINT; (* 4. An
```

Index	Name	Value	Default
2	Ch1T...	0 Voltage (+/-10,000 mV)	0 Voltage (+/-10,0
3	Ch2T...	0 Voltage (+/-10,000 mV)	0 Voltage (+/-10,0
4	Ch3T...	0 Voltage (+/-10,000 mV)	0 Voltage (+/-10,0
5	Ch4T...	0 Voltage (+/-10,000 mV)	0 Voltage (+/-10,0

Each of the 4 channels can be independently programmed to be used as voltage or current output. The parameters Ch1Type, Ch2Type, Ch3Type and Ch4Type can be configured as shown in the following table.

Parameter value	Channel Mode	Output Range
0	Voltage Output	-10000 mV – 10000 mV
1	Current Output	0-20000 µA

The output range is fixed as specified in the table. The value written by the PLC program to the output channels is an integer value between 0 and 10.000 in case of voltage and between 0 and 20.000 in case of current.

Configuration parameters of the Analog Outputs can be changed at runtime using the EXOR_IO_CTRL function with the following Function Codes:

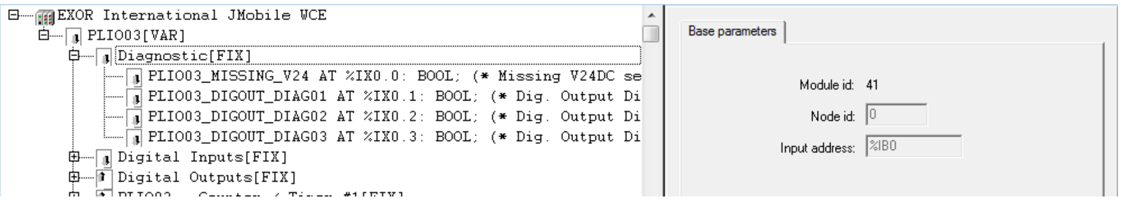
Function Code	Argument	Description
10	0 -1	Set Analog Output Mode Channel 1
11	0 -1	Set Analog Output Mode Channel 2
12	0 -1	Set Analog Output Mode Channel 3
13	0 -1	Set Analog Output Mode Channel 4

Other Function Codes are reserved for different uses and must not be used.

Diagnostic

PLIO03 reports diagnostic information in the Diagnostic section of the configuration.

Parameter	Description
PLIO03_MISSING_V24V	TRUE means 24V missing
PLIO03_DIGOUT_DIAG01	TRUE means digital outputs 1-4 are ok
PLIO03_DIGOUT_DIAG02	TRUE means digital outputs 5-8 are ok
PLIO03_DIGOUT_DIAG03	TRUE means digital outputs 9-12are ok



```
EXOR International JMobile WCE
    PLIO03[VAR]
        Diagnostic[FIX]
            PLIO03_MISSING_V24 AT %IX0.0: BOOL; (* Missing V24DC se
            PLIO03_DIGOUT_DIAG01 AT %IX0.1: BOOL; (* Dig. Output Di
            PLIO03_DIGOUT_DIAG02 AT %IX0.2: BOOL; (* Dig. Output Di
            PLIO03_DIGOUT_DIAG03 AT %IX0.3: BOOL; (* Dig. Output Di
        Digital Inputs[FIX]
        Digital Outputs[FIX]
```

Module id: 41

Node id: 0

Input address: %IB0

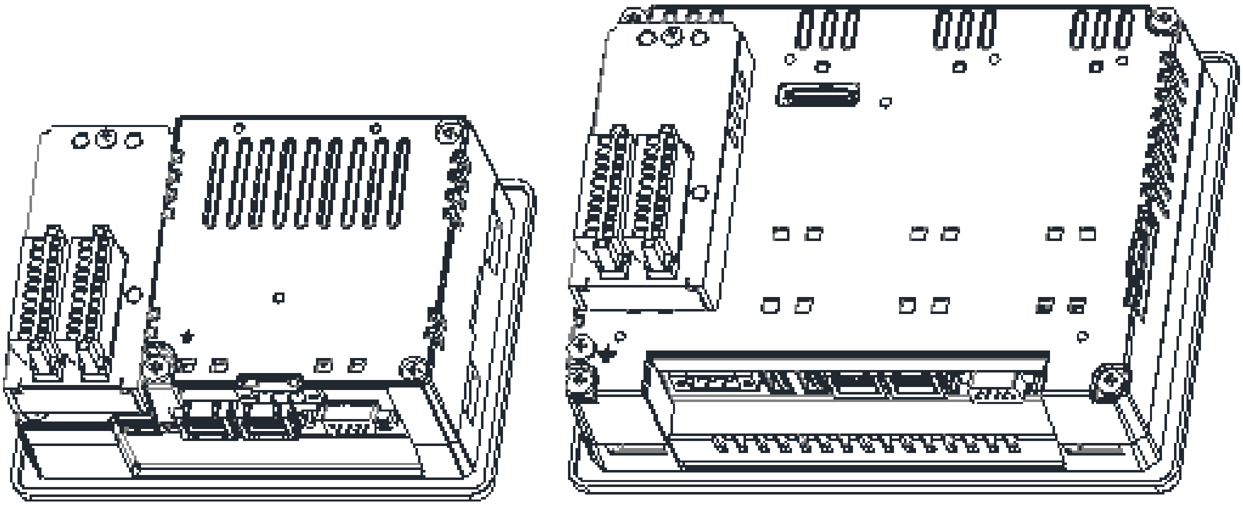
PLIO06

PLIO06 is a compact I/O module. It has been designed for creating simple applications with a limited number of digital I/O signals.

PLIO06 specifications are the following:

- 8 optically isolated digital inputs
- 6 optically isolated digital outputs
- 1 relay output

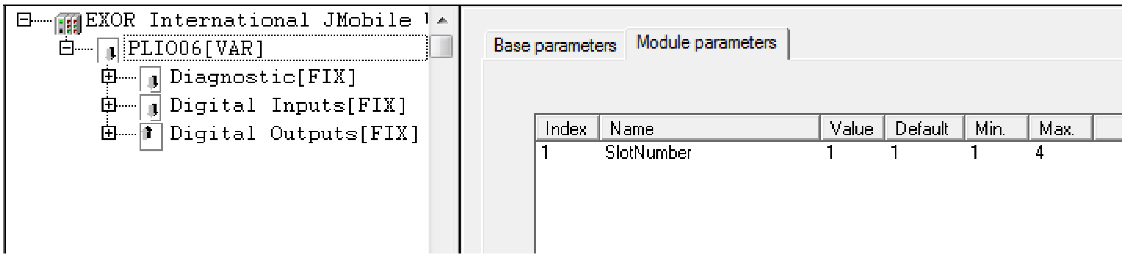
The following figure shows a configuration for eTOP504 and eTOP507 using the PLIO06 module.



The PLIO06 module must be included in the PLC configuration, as shown in the following figure.

Double PLIO06 configuration is allowed. When used is is necessary to specify the Slot number for each module. The SlotNumber parameter is available in the tab Module parameters.

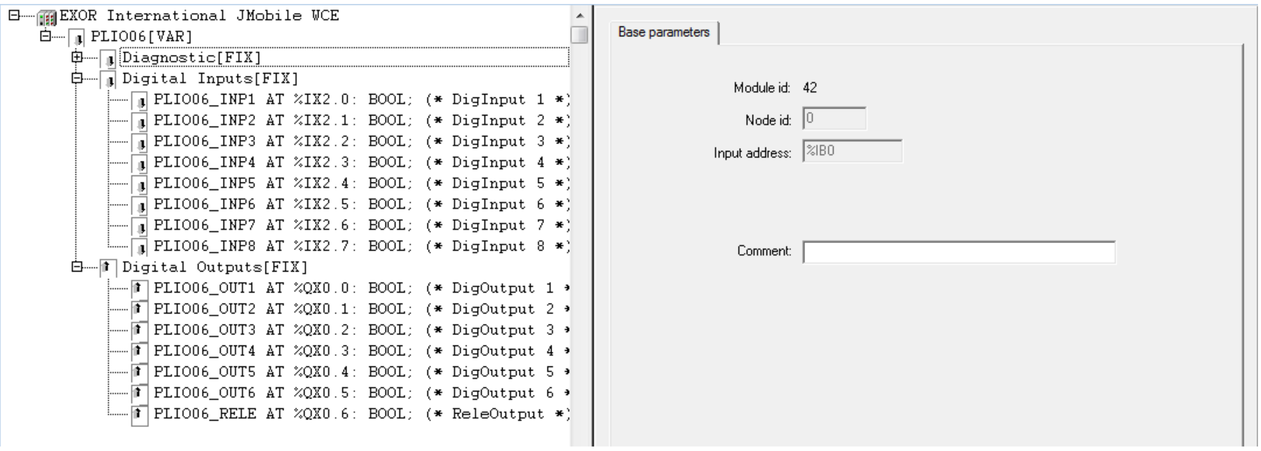
PLIO06 must be assigned a SlotNumber from 1 to 4. An error will be reported at run-time if the I/O module has not been plugged in the slot defined in the PLC configuration.



```
EXOR International JMobile l
  PLIO06[VAR]
    Diagnostic[FIX]
    Digital Inputs[FIX]
    Digital Outputs[FIX]
```

Index	Name	Value	Default	Min.	Max.
1	SlotNumber	1	1	1	4

No further configuration steps are required for using the module.



```
EXOR International JMobile WCE
  PLIO06[VAR]
    Diagnostic[FIX]
    Digital Inputs[FIX]
      PLIO06_INP1 AT %IX2.0: BOOL; (* DigInput 1 *)
      PLIO06_INP2 AT %IX2.1: BOOL; (* DigInput 2 *)
      PLIO06_INP3 AT %IX2.2: BOOL; (* DigInput 3 *)
      PLIO06_INP4 AT %IX2.3: BOOL; (* DigInput 4 *)
      PLIO06_INP5 AT %IX2.4: BOOL; (* DigInput 5 *)
      PLIO06_INP6 AT %IX2.5: BOOL; (* DigInput 6 *)
      PLIO06_INP7 AT %IX2.6: BOOL; (* DigInput 7 *)
      PLIO06_INP8 AT %IX2.7: BOOL; (* DigInput 8 *)
    Digital Outputs[FIX]
      PLIO06_OUT1 AT %QX0.0: BOOL; (* DigOutput 1 *
      PLIO06_OUT2 AT %QX0.1: BOOL; (* DigOutput 2 *
      PLIO06_OUT3 AT %QX0.2: BOOL; (* DigOutput 3 *
      PLIO06_OUT4 AT %QX0.3: BOOL; (* DigOutput 4 *
      PLIO06_OUT5 AT %QX0.4: BOOL; (* DigOutput 5 *
      PLIO06_OUT6 AT %QX0.5: BOOL; (* DigOutput 6 *
      PLIO06_RELE AT %QX0.6: BOOL; (* ReleOutput *)
```

Module id: 42

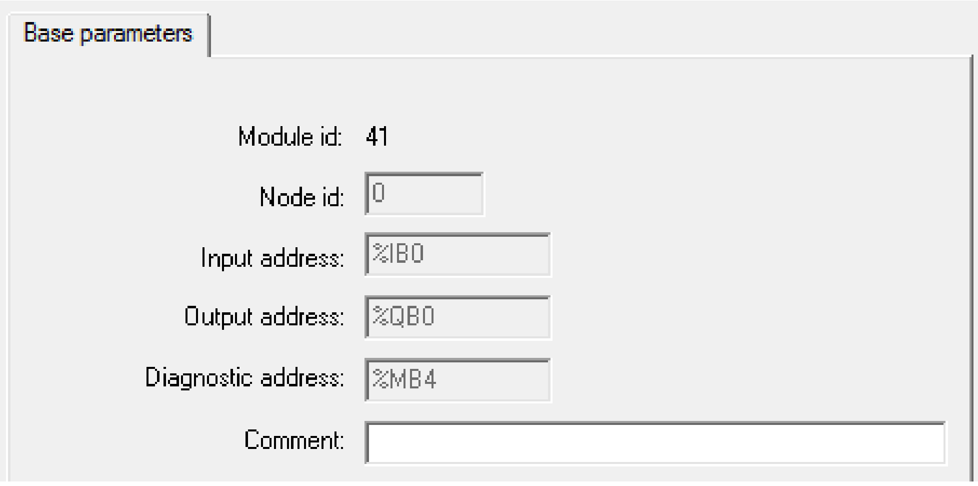
Node id: 0

Input address: %IB0

Comment:

Base parameters

The following figure shows the Base Parameters tab for integrated I/O modules.



Module id: 41

Node id: 0

Input address: %IB0

Output address: %QB0

Diagnostic address: %MB4

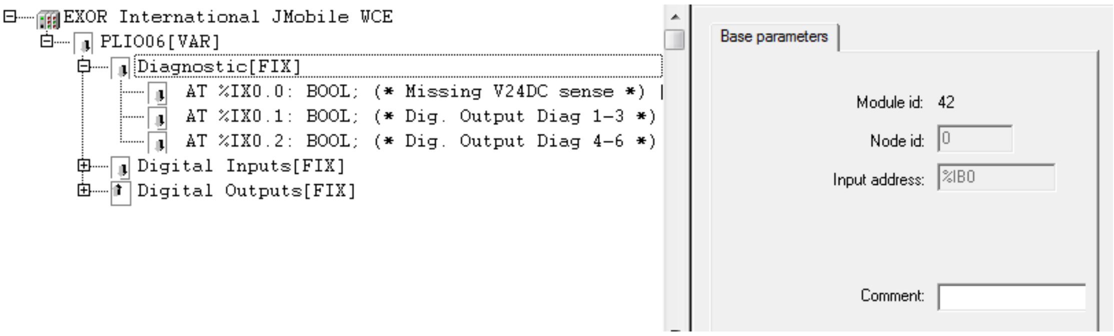
Comment:

Element	Description
Module id	CODESYS internal identifier used to recognize the board; it is a read only parameter.
Node id	CODESYS internal identifier assigned by the Development System depending on the order of the board in the PLC Configuration layout; it is a read only parameter.
Input address	Starting address for input information
Output address	Starting address for output information
Diagnostic address	Memory address for the storage of diagnostic information of the I/O module

Diagnostic

PLIO06 reports diagnostic information in the Diagnostic section of the configuration.

Parameter	Description
DIAG_24VOK	TRUE means 24V missing
DIGOUT1_DIAG01	TRUE means digital outputs 1-3 are ok
DIGOUT1_DIAG02	TRUE means digital outputs 4-6 are ok



```
EXOR International JMobile WCE
  PLIO06[VAR]
    Diagnostic[FIX]
      AT %IX0.0: BOOL; (* Missing V24DC sense *)
      AT %IX0.1: BOOL; (* Dig. Output Diag 1-3 *)
      AT %IX0.2: BOOL; (* Dig. Output Diag 4-6 *)
    Digital Inputs[FIX]
    Digital Outputs[FIX]
```

Module id: 42

Node id: 0

Input address: %IB0

Comment:

Ethernet I/O Slave

CODESYS V2 runtime can use the built-in Ethernet interface of the operator panels for the distributed Modbus/TCP I/O network. No additional hardware is required.

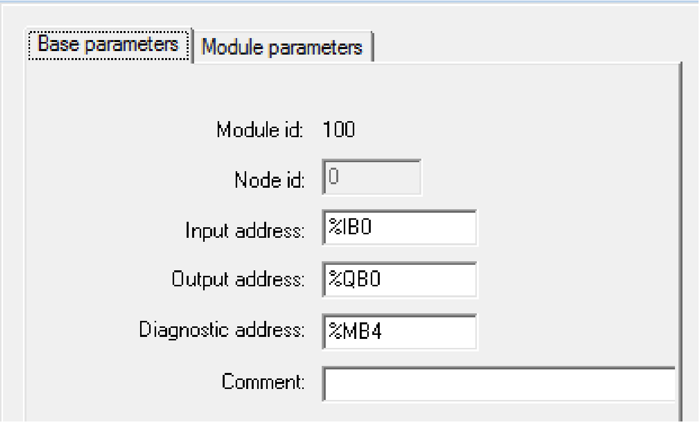
To add an Ethernet I/O module, into the PLC Configuration use the Insert element/Append subelement commands from Insert menu or from context menu on right click on the Root voice of configuration tree. The list of available elements will appear as sub-menu.

Choose the Ethernet I/O Slave module from the list to add it to the current PLC configuration.

The parameters of the Ethernet interface are grouped in two tabs accessible on the right part of the PLC Configuration tool when the Ethernet I/O Slave element has been added to the configuration tree.

Base parameters

The following figure shows the “Base Parameters” tab.



Module id: 100

Node id: 0

Input address: %IB0

Output address: %QB0

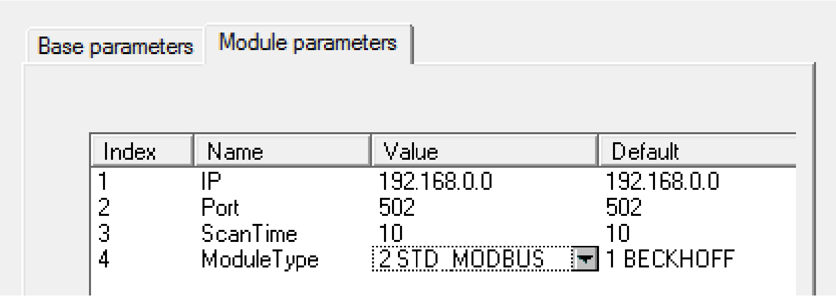
Diagnostic address: %MB4

Comment:

Element	Description
Module id	CODESYS internal identifier used to recognize the board; it is a read only parameter.
Node id	CODESYS internal identifier assigned by the Development System depending on the order of the board in the PLC Configuration layout; it is a read only parameter.
Input address	Starting memory address for input information
Output address	Starting memory address for output information
Diagnostic address	Memory address for the storage of diagnostic information of the Ethernet network

Module Parameters

The following figure shows the “Module Parameters” tab. This tab contains the basic information for setting up Ethernet slave nodes.



Index	Name	Value	Default
1	IP	192.168.0.0	192.168.0.0
2	Port	502	502
3	ScanTime	10	10
4	ModuleType	2 STD_MODBUS	1 BECKHOFF

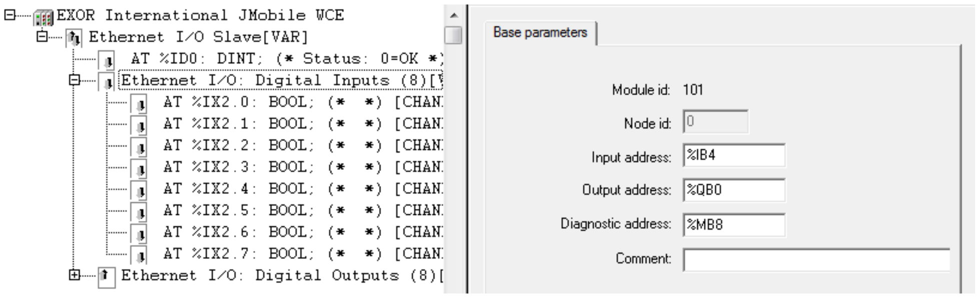
Element	Description
IP	IP Address of the Ethernet slave node
Port	TCP Port Number
Scan Time	Scan time in ms
Module Type	Type of the slave node. The following types are supported: WAGO BECKHOFF STD_ MODBUS WEIDMULLER

After the set-up of the Ethernet I/O Slave the structure of available I/O points can be defined.

To add an I/O module, select the Ethernet I/O Slave from the PLC configuration tree then use the Append subelement command from Insert menu or from context menu on right click on the Ethernet I/O Slave voice of configuration tree. The list of available I/O points will appear as sub-menu.

Choose the I/O point from the list to add it to the current PLC configuration.

The following figure shows an example configuration including 8 digital inputs and 8 digital outputs.



```
EXOR International JMobile WCE
  Ethernet I/O Slave[VAR]
    AT %ID0: DINT; (* Status: 0=OK *)
    Ethernet I/O: Digital Inputs (8)[
      AT %IX2.0: BOOL; (*  *) [CHAN
      AT %IX2.1: BOOL; (*  *) [CHAN
      AT %IX2.2: BOOL; (*  *) [CHAN
      AT %IX2.3: BOOL; (*  *) [CHAN
      AT %IX2.4: BOOL; (*  *) [CHAN
      AT %IX2.5: BOOL; (*  *) [CHAN
      AT %IX2.6: BOOL; (*  *) [CHAN
      AT %IX2.7: BOOL; (*  *) [CHAN
    Ethernet I/O: Digital Outputs (8)[
```

Module id: 101

Node id: 0

Input address: %IB4

Output address: %QB0

Diagnostic address: %MB8

Comment:

Use of EXOR_IO_CTRL for Setup of STD_MODBUS Module Type

When using the STD_MODBUS module type the Modbus physical addressing of the I/O resources can be modified if require, to set a specific Modbus starting address for the I/O.

This is done by using the EXOR_IO_CTRL CODESYS Function block.

The below table specifies the correspondence between the I/O type and the specific Modbus resource.

I/O	Modbus memory resource	Data type
Analog Input	Input Register	16 bit
Analog Output	Holding Register	16 bit
Digital Input	Input bit	1 bit
Digital Output	Output Coil	1 bit

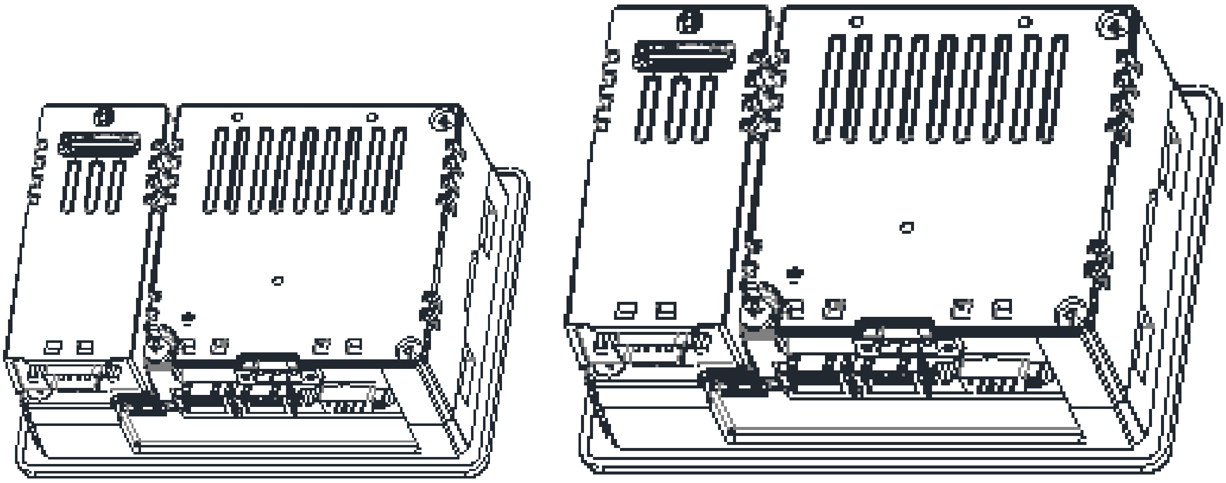
Below you can find the parameters to be passed to the EXOR_IO_CTRL function to modify the Modbus addressing for the I/O

Function Code	Argument	Description
1	0=disabled 1=read-only 2=full access (default)	Set access control (for multi master fault tolerant systems)
2	0-65535	Set CoilModbusOffset start address of digital outputs in Modbus Slave
3	0-65535	Set InputModbusOffset start address of digital inputs in Modbus Slave
4	0-65535	Set HRegModbusOffset start address of analogue outputs in Modbus Slave
5	0-65535	Set HRegModbusOffset start address of analogue intputs in Modbus Slave

Can Master

CODESYS V2 runtime can act as a CAN Master to allow the use of distributed CANopen I/O points. To interface the operator panel with CAN network one of the available CAN optional modules must be installed.

The following figure shows eTOP504 and eTOP507 with option modules PLCM01-CDS mounted.



To add a CAN Master, into the PLC Configuration use the Insert element/Append subelement commands from Insert menu or from context menu on right click on the Root voice of configuration tree. The list of available elements will appear as sub-menu.

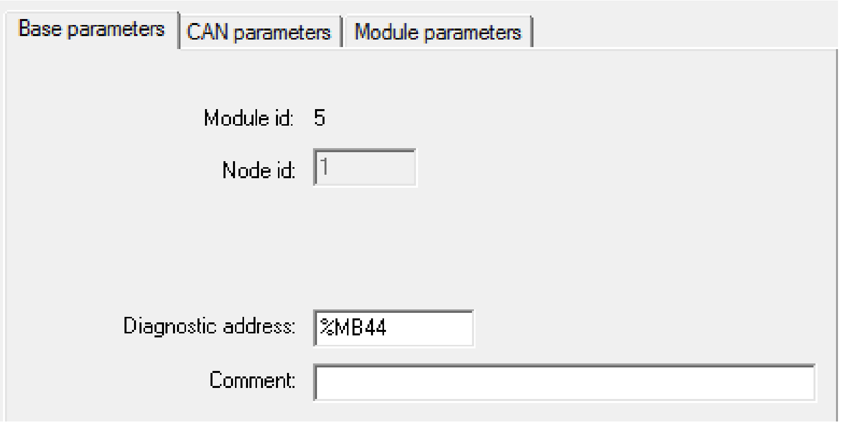
Choose the CanMaster module from the list to add it to the current PLC configuration.

The parameters of the CAN interface are grouped in three tabs accessible on the right part of the PLC Configuration tool when the CanMaster element has been added to the configuration tree.

Note: A complete and detailed description on the configuration of CAN controllers and on the configuration of CAN slave devices is included in the CODESYS User Manual

Base parameters

The following figure shows the “Base Parameters” tab.



Module id: 5

Node id: 1

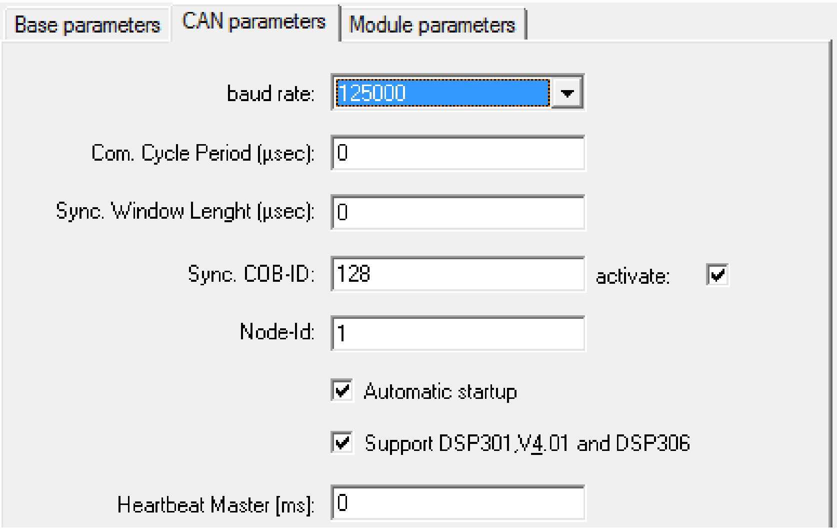
Diagnostic address: %MB44

Comment:

Element	Description
Module id	CODESYS internal identifier used to recognize the board; it is a read only parameter.
Node number	CODESYS internal identifier assigned by the Development System depending on the order of the board in the PLC Configuration layout; it is a read only parameter.
Diagnostic address	Memory address for the storage of diagnostic information of the CAN master

CAN parameters

The following figure shows the “CAN Parameters” tab; it contains all the parameters related to the bus configuration.



baud rate: 125000

Com. Cycle Period [µsec]: 0

Sync. Window Lenght [µsec]: 0

Sync. COB-ID: 128 activate: ☑

Node-Id: 1

☑ Automatic startup

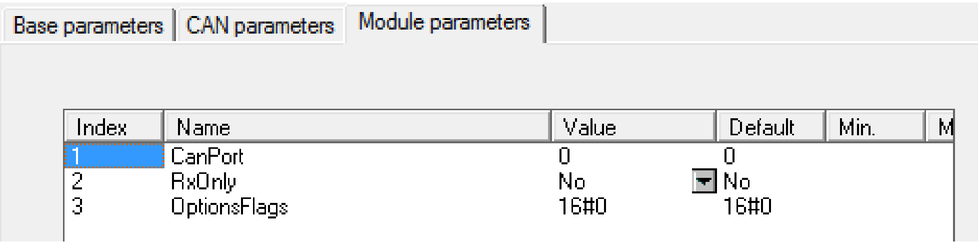
☑ Support DSP301,V4.01 and DSP306

Heartbeat Master [ms]: 0

Element	Description
Baud rate	Bus aped Baud rate setting.
Com. Cycle Period	Specifies the communication cycle interval related to the PDO messages exchange; if set to 0, the default value for communication cycle is 20ms. Minimum allowed value is 1ms.
Sync. Window Length	Not supported.
Sync. COB-ID	Default value 128 is assigned following the CiA CANopen standard, the COB-ID can be changed depending on installation requirements.
Activate	Activate/deactivate the Sync function.
Node-Id	CAN node number assigned to the CAN master.
Automatic Startup	If set the NMT startup command is sent automatically to all configured slaves at power-up.
Support DSP301…	Enables the controls for the heartbeat and of the Modular Slaves; Modular Slaves is a concept defined by CiA and refers to the usage of a bus coupler that allows connecting several combination of modules (please refer to CiA documentation for further details).
Heartbeat Master	Defines the timing, in milliseconds between two heartbeat messages sent by Master on CAN network.

Module parameters

The following figure shows the “Module Parameters” tab. The meaning of the available parameters is the following:



Index	Name	Value	Default	Min.	M
1	CanPort	0	0		
2	RxOnly	No	No		
3	OptionsFlags	16#0	16#0		

Element	Description

CanPort	Select slot for the optional CAN module.
	Value=0 Slot 1 or Slot 2
	Value=1 Slot 3 or Slot 4

RxOnly	Select receive-only mode for the CAN interface
OptionsFlags	These are board parameters. Value=16#80000000 enables the sync mode of master in which CAN scan cycle is driven by PLC cycle and starts immediately after the update of Output Process Image

Installation of EDS descriptor files for third part CAN Slave devices

The use of third part CAN Slave devices requires a specific EDS descriptor file provided by the device manufacturer to be installed into the system. Once installed the file the CAN Slave device will be listed and selectable into the PLC configuration.

To install an EDS file is enough to copy the file provided by the device manufacturer into the Target specific folder.

For example to install an EDS file to be used into an application made for UN31 Target the file must be copied into the C:\Program Files\3S Software\CoDeSys V2.3\targets\EXOR\JMobileWCE_ARM31 folder.

Definition of CAN I/O Slaves in the PLC Configuration.

After the set-up of the CAN Master the structure of available CAN I/O Slaves can be defined.

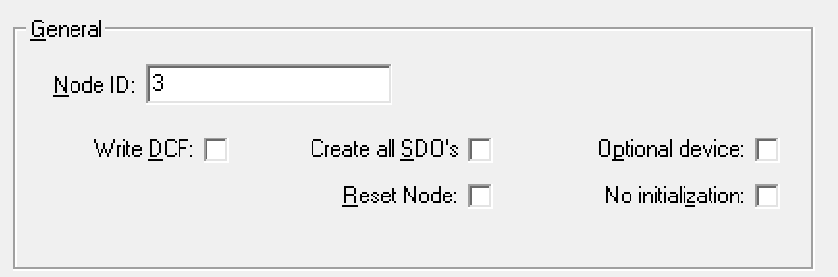
To add a CAN Slave module, select the CanMaster from the PLC configuration tree then use the Append subelement command from Insert menu or from context menu on right click on the CanMaster voice of configuration tree. The list of available I/O Slaves will appear as sub-menu.

Choose the I/O point from the list to add it to the current PLC configuration.

Settings for CAN Slaves

The configuration for the CAN slaves has a common part, which is independent from the EDS file. The following figure shows the “CAN Parameters” tab of a generic CAN Slave.

Note: For specific information regarding the set-up of the CAN Slaves please refer to the documentation provided from the CAN Slave manufacturer.



General

Node ID: 3

Write DCF: □ Create all SDO's □ Optional device: □

Reset Node: □ No initialization: □

Element	Description
Node ID	CAN node number of the CAN Slave device (1 – 127).
Write DCF	Create DCF file.
Create all SDO's	When selected the SDO messages for the slave configuration, depending on the PDO mapping are created for all objects. When not selected the SDO messages for the slave configuration are created only for the modified objects. In this latter case, please make sure the EDS file loaded in CODESYS V2 is matching the hardware device features, otherwise some required SDO messages will be erroneously skipped.
No initialization	If selected the sequence of SDO messages required for the device initialization (PDO mapping) will not be created.
Optional device	If selected the current device is considered as optional into the bus. At start-up the CAN controller will check if it is present applying the following rules: • If the device is present since start-up and correctly replies to the CANopen mandatory object “Device Type” query (matching the EDS file specification), then it is started. The master will continue with the next device. • If the device is present since start-up and it does not reply as expected to the “Device type” query, it is not started. The master stops then, reporting a mismatching error in the CAN configuration; if the “Optional device” with not-matching “Device Type” is inserted in the bus after start-up, the master will skip it and continue to scan the other devices. • If the device is not present since start-up, it is simply skipped. The master will continue with the next device.

The CAN Interface

The PLCM01 CAN option module include a CAN bus interface implemented according to the CAN protocol specifications 2.0 A.

This CAN controller supports only Standard frame format (2.0 A) with bit rates up to 1 Mbit/s.

The following transfer functions have been implemented:

- Transfer rate and timing
- Message framing (Part A)
- Arbitration accordingly to Part A specifications
- Automatic retransmission in case of lost arbitration or error detection
- Acknowledgement
- Message validation
- Error detection and error signaling
- Global Identifier masking (for 11-bit and 29-bit long identifiers)
- Interrupt or data polling driven software supported
- Automatic transfer of data frame (prepared in SDRAM buffer) triggered by one bit setting
- Automatic receive of data packets with the allowed frame identifier
- 32 separated SDRAM memory buffers for data packets having the node corresponding ID
- Fully implemented CAN error fault confinement
- Automatic detection of Bus off state
- Detection of the heavily disturbed CAN bus and warning

Programming the parameter baudRateKbps at the value 0 enables the use of custom timing

The resulting baud rate is calculated using the formula:

Bit frequency = 8 MHz / (Prescaler * (1 + Tsetup + Thold))

Valid values for parameters are:

Prescaler: 1 to 64

Tsetup: 1 to 8

Thold: 1 to 4

Other two parameters can affect the behavior of the CAN controller:

SyncJumpWidth: defines the number of time quanta (8 MHz / Prescaler) allowed to accept a SYNC pulse. Valid values are 1 to 4.

SampleMode: defines the number of times the bit is sampled before is considered valid. Valid values are 0 (1 sample) and 1 (3 samples).

Programming the CANopen Interface

Connection to special CANopen devices may require direct access to some CAN commands. Function blocks are available for this purpose. This chapter describes the most important cases.

CAN Master Operate Functions control

The function “EXOR_IO_CTRL” can be used to control special features of the CAN master.

The following tables show the various Functions Codes for EXOR_IO_CTRL; depending if the IO_CTRL is directed to CanMaster, CAN Slave Node, Rx PDO or Tx PDO, a different set of Functions is available.

Functions Code Directed to CanMaster Module

Function Code	Argument	Description
1	Any	Return the QuickStatus, i.e. the global status: 0 means all ok, otherwise the error code (for a list of error codes please see below).
2	Any	This function fetches one error from the Error FIFO. All the errors are collected in an Error FIFO, so that no error event is lost. The 32 bits integer returned contains the error code (see the list below) in the lower 16 bits and the Node Id in the higher 16 bits. If there are no error events it returns 0.

Functions Code Directed to CAN Slave Node

For those functions directed to the CAN Slaves the SubNode value is used to calculate the Argument parameter.

Each Sub Node in the CODESYS V2 PLC Configuration editor has a progressive Node Number starting from zero; this is assigned by CODESYS and can be seen in the board “Base Parameters”.

Function Code	Argument	Description
0+10000*(SubNode+1)	Any	Returns the current status of the device to which the specified I/O variable is connected. 0 means no errors, otherwise the error code is reported (see error codes list below).

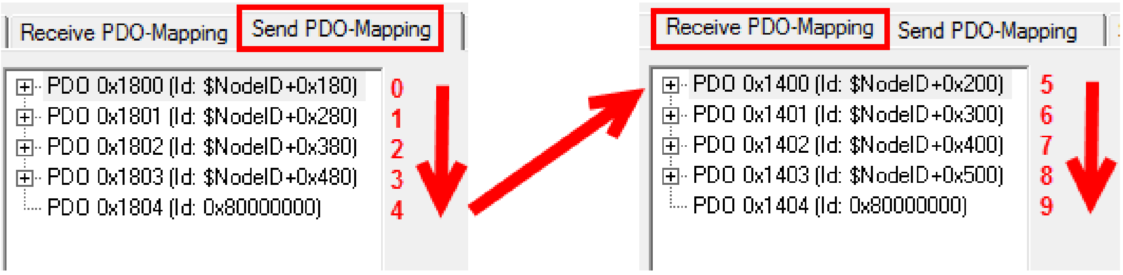
Functions Code Directed to TX PDOs

Function Code	Argument	diloCtrlResult	Description
5+10000*(SubNode+1)	PDO Number	Number of Scansions	Sets the transmission frequency (expressed in number of Scansions), by default a PDO is sent every scansion.
6+10000*(SubNode+1)	PDO Number	Number of Milliseconds	Sets the transmission frequency (expressed in number Milliseconds)
7+10000*(SubNode+1)	PDO Number	Length * 65535+COB ID	Set new COB-ID (11 bits) and LENGHT of PDO. The passed Value must contain in lower 16 bits the new COB-ID, in upper 16 bits the new LENGTH.

The PDO Number is CAN Slave node depending, the numbering is calculated from the PDO Mapping that can be found into the CAN Slave properties in CODESYS.

The numbering starts from 0 which corresponds to the first Send PDO increasing by 1 for each PDO, the numbering continues with the Receive PDOs. Please refer to the image below as example for the calculation of PDO numbering.

PDO Numbering



Receive PDO-Mapping
Send PDO-Mapping
PDO 0x1800 (Id: $NodeID+0x180)
PDO 0x1801 (Id: $NodeID+0x280)
PDO 0x1802 (Id: $NodeID+0x380)
PDO 0x1803 (Id: $NodeID+0x480)
PDO 0x1804 (Id: 0x80000000)
0
1
2
3
4
Receive PDO-Mapping
Send PDO-Mapping
PDO 0x1400 (Id: $NodeID+0x200)
PDO 0x1401 (Id: $NodeID+0x300)
PDO 0x1402 (Id: $NodeID+0x400)
PDO 0x1403 (Id: $NodeID+0x500)
PDO 0x1404 (Id: 0x80000000)
5
6
7
8
9

Functions Code Directed to RX PDOs

Function Code	Argument	diloCtrlResult	Description
3+10000* (SubNode+1)	PDO Number	Any	Returns TRUE if a PDO has been received
7+10000* (SubNode+1)	PDO Number	COB-ID	Set new COB-ID (11 bits) of PDO. The passed Value must contain the new COB-ID.
8+10000* (SubNode+1)	PDO Number	Any	Forces to send the selected PDO; PDO selection depends form the ucPDOnr parameter.

Error Codes

The CODESYS Development System allows showing in its status bar some diagnostic message related to the PLC operation. The following table contains a list of all the possible error codes. Errors marked as “Fatal” are those preventing the operation of the CANopen I/O driver.

Element	Description
0	OK, no errors
1	EXOR.CANopen.CANCFG: (Fatal) [SlotNr] Invalid baudrate
2	EXOR.CANopen: (Fatal) [SlotNr] Invalid board (perhaps an old board version ?)
3	EXOR.CANopen: (Fatal) [SlotNr] Too many boards defined
4	EXOR.CANopen: (Fatal) [SlotNr] Missing Configuration board (it should be before any CANopen board)
5	EXOR.CANopen: (Fatal) [SlotNr] Invalid Node ID (1..127)
6	EXOR.CANopen: (Fatal) [SlotNr] Invalid PDO Length (0..8)
7	EXOR.CANopen: Node [NodeId] has Guarding Error (toggling bit or status incorrect)
8	EXOR.CANopen: Node [NodeId] is Dead (Node does not reply)
9	EXOR.CANopen: Node [NodeId] restarted (Node is alive again)
10	EXOR.CANopen.CANM????: (Fatal) [SlotNr] Missing preceding CANMICFG or CANMOCFG

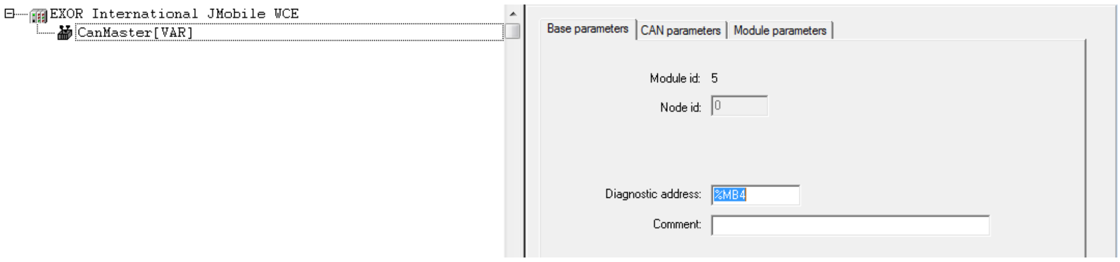
Element	Description
	board (see tech note)
11	EXOR.CANopen.CANM????: (Fatal) [SlotNr] Invalid Offset
12	EXOR.CANopen.CANSDO??: Function Block CANSDORD/WR: Too many SDO/PDO
13	EXOR.CANopen.CANSDO??: Function Block CANSDORD/WR: Invalid parameter/s
14	EXOR.CANopen.CANSDO??: Function Block CANSDORD/WR: Invalid reply from remote SDO server
15	EXOR.CANopen.CANSDORD: Function Block CANSDORD: Returned size differs from requested size
16	EXOR.CANopen.CANSDO??: Function Block CANSDORD/WR: No reply timeout
17	EXOR.CANopen: CAN Error Bus Offstate
18	EXOR.CANopen: CAN Error STAT_ERR_PASSIV
19	EXOR.CANopen: CAN Error STAT_WARN
20	EXOR.CANopen: CAN Error STAT_STUFF_ERR
21	EXOR.CANopen: CAN Error STAT_FORM_ERR
22	EXOR.CANopen: CAN Error STAT_ACK_ERR
23	EXOR.CANopen: CAN Error STAT_BIT_ERR
24	EXOR.CANopen: CAN Error STAT_CRC_ERR
25	EXOR.CANopn: CANOPEN_ERR_SCAN_TOO_FAST: value of Com.CyclePeriod too low or problems on CANbus

Error code 25

In the case the value of Com.CyclePeriod is too low and not all of the PDOs of the previous scan have been transmitted when a new scan is started, the new scan will only send the SYNC message, while new PDOs and NodeGuarding messages will not be transmitted. This is to ensure that all previous PDOs can be transmitted. New scans will only transmit SYNC message until all of the PDOs have been transmitted.

CAN Master Diagnostic Information

When adding a “CAN Master” element in the PLC Configuration, an area of 144 bytes is allocated starting at the indicated diagnostic address, as shown in the following figure.



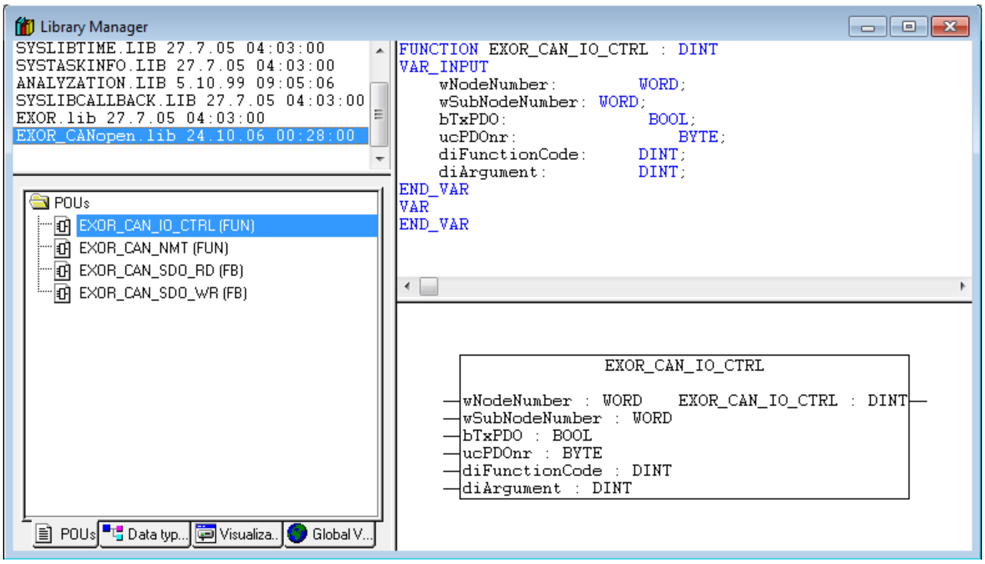
Module id: 5

Node id: 0

Diagnostic address: %MB4

Comment:

When the “CAN Master” board is added to project, after first compilation, the library “EXOR_CANopen.lib” is automatically included, as shown in the following figure.



```
SYSLIBTIME.LIB 27.7.05 04:03:00
SYSTASKINFO.LIB 27.7.05 04:03:00
ANALYZATION.LIB 5.10.99 09:05:06
SYSLIBCALLBACK.LIB 27.7.05 04:03:00
EXOR.lib 27.7.05 04:03:00
EXOR_CANopen.lib 24.10.06 00:28:00
```

```
FUNCTION EXOR_CAN_IO_CTRL : DINT
VAR_INPUT
    wNodeNumber:        WORD;
    wSubNodeNumber: WORD;
    bTxPDO:              BOOL;
    ucPDOnr:                BYTE;
    diFunctionCode:     DINT;
    diArgument:         DINT;
END_VAR
VAR
END_VAR
```

```
                 EXOR_CAN_IO_CTRL
─┤wNodeNumber : WORD    EXOR_CAN_IO_CTRL : DINT├─
─┤wSubNodeNumber : WORD
─┤bTxPDO : BOOL
─┤ucPDOnr : BYTE
─┤diFunctionCode : DINT
─┤diArgument : DINT
```

The user data type GETBUSSTATE is defined inside this library.

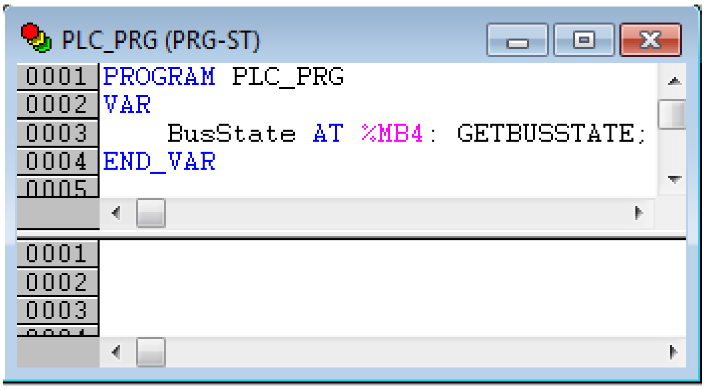
```
TYPE GETBUSSTATE :
STRUCT
     BOLDENABLE: BOOL;
     ENABLE: BOOL;
     DRIVERNAME:POINTER TO STRING;
     DEVICENUMBER:INT;1\
     READY:BYTE;
     STATE:INT;
     EXTENDEDINFO:ARRAY[0..129] OF BYTE;
END_STRUCT
END_TYPE
```

where:

Element	Description
BOLDENABLE	Always TRUE
ENABLE	Always TRUE
DRIVERNAME	"CANopen Master"
DEVICENUMBER	CanPortchoosen in the CANMaster configuration
READY	TRUE if running

Element	Description
STATE	Quick Status: 0 (zero) means OK, other values are error codes, see “ERROR CODES” table
EXTENDEDINFO	Elements 0, 128 and 129 are not used. Elements 1 to 127 reports the state of nodes 1 to 127. Meaning of the bits of each byte is: Bit 0: Node exists in PLC configuration. Bit 1: Node is available in bus system. Bit 2: Node reports error. Bit 3: Node is initialized and without errors (i.e. it's OFF during initialization and configuration). A node will return the value 11 (0x0b) to indicate correct operation.

To read the CAN Diagnostic informations in the PLC program define a variable of type GETBUSSTATE pointing to the Diagnostic Address of the CAN Master element, as shown in the following figure.



```
PROGRAM PLC_PRG
VAR
    BusState AT %MB4 : GETBUSSTATE;
END_VAR
```

This GETBUSSTATE user data type complies with the CODESYS V2 standard method of obtaining diagnostic information from bus I/O, as explained in the CODESYS V2 help.

Access to Remote Data Using SDO Protocol

The SDO protocol can be used to access any remote variable, defined according to the CANopen standard. The CANopen interface in Exor works as SDO client and remote nodes are servers.

While in PDO mode the transmission is normally cyclic and automatic, in SDO mode the data exchange is normally done on purpose. Each session can normally transfer only one data item. This means that the SDO protocol is much slower than the PDO protocol.

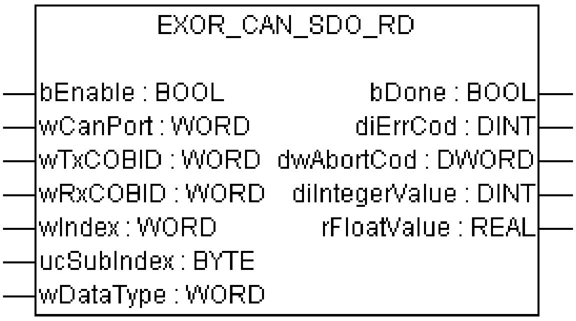
Two Function Blocks are available to configure communication via the SDO protocol.

EXOR_CAN_SDO_RD read remote variables

EXOR_CAN_SDO_WR write remote variables

These Function Blocks are available into the library “EXOR_CANopen.lib”.

EXOR_CAN_SDO_RD



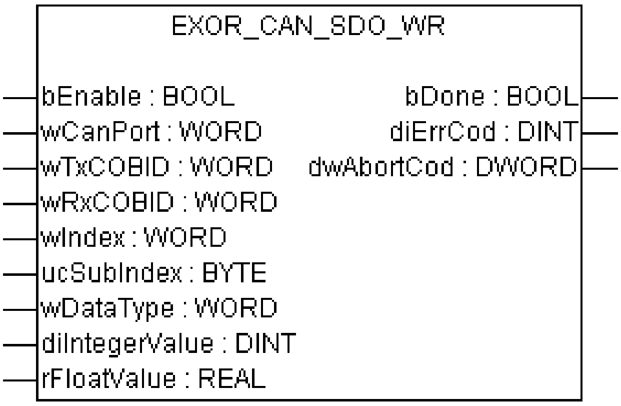
Parameters for the EXOR_CAN_SDO_RD function block are:

Element	Description
wIndex	Address of CAN object inside the remote node, as defined by the manufacturer.
ucSubindex	Address of single variable inside the object.
wDataType	One of the data types supported by CAN.
wCanPort	Identifies the CAN controller channel.
wTxCOBID	Specifies the COB ID of the transmit SDO.
wRxCOBID	Specifies the COB ID of the receiving SDO.
bEnable	Enable bit.

Return values are:

Element	Description
bDone	Flag indicating the operation has been executed.
diErrCod	Error code of the operation. It is generated by the client and it is valid only after operation has been completed. Value 0 means a successful operation.
dwAbortCode	Code sent by the server in case the operation is aborted. It is returned by the function block as received from the remote device, so refer to CAN standard definition or to specific technical description of the server (remote device).
diIntegerValue	Value of the read variable in integer format.
rFloatValue	Value of the read variable in float format.

EXOR_CAN_SDO_WR



The parameters for the EXOR_CAN_SDO_WR function block are:

Element	Description
bEnable	Enable bit.
wCanPort	Identifies the CAN controller channel.
wTxCOBID	Specifies the COB ID of the transmit SDO.
wRxCOBID	Specifies the COB ID of the receiving SDO.
wIndex	Address of the CAN object inside the remote node, as defined by the manufacturer.
ucSubindex	Address of single variable inside the object.
wDataType	One of the data types supported by CAN.
diIntegerValue	Is the value to be written in integer format.
rFloatValue	Is the value to be written in float format.

Return values are:

Element	Description
bDone	Is a flag indicating the operation has been executed.
diErrCod	Error code of the operation. It is generated by the client and it is valid only after operation is completed. Value 0 means a successful operation.
dwAbortCode	Is the code sent by the server in case the operation is aborted. It is reported as received, so refer to CAN standard definition or to specific technical description of the server.

The possible values for the abort code (see return value dwAbortCod) are shown in the table below.

Element	Description
0503 0000h	Toggle bit not alternated.
0504 0000h	SDO protocol timed out.
0504 0001h	Client/server command specifier not valid or unknown.
0504 0002h	Invalid block size (block mode only).
0504 0003h	Invalid sequence number (block mode only).
0504 0004h	CRC error (block mode only).
0504 0005h	Out of memory.
0601 0000h	Unsupported access to an object.
0601 0001h	Attempt to read a write only object.
0601 0002h	Attempt to write a read only object.
0602 0000h	Object does not exist in the object dictionary.
0604 0041h	Object cannot be mapped to the PDO.
0604 0042h	The number and length of the objects to be mapped would exceed PDO length.
0604 0043h	General parameter incompatibility reason.
0604 0047h	General internal incompatibility in the device.
0606 0000h	Access failed due to an hardware error.
0607 0010h	Data type does not match, length of service parameter does not match
0607 0012h	Data type does not match, length of service parameter too high

Element	Description
0607 0013h	Data type does not match, length of service parameter too low
0609 0011h	Sub-index does not exist.
0609 0030h	Value range of parameter exceeded (only for write access).
0609 0031h	Value of parameter written too high.
0609 0032h	Value of parameter written too low.
0609 0036h	Maximum value is less than minimum value.
0800 0000h	general error
0800 0020h	Data cannot be transferred or stored to the application.
0800 0021h	Data cannot be transferred or stored to the application because of local control.
0800 0022h	Data cannot be transferred or stored to the application because of the present device state.
0800 0023h	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of an file error).

Allowed values for data type (refer to input parameter wDataType) are listed below.

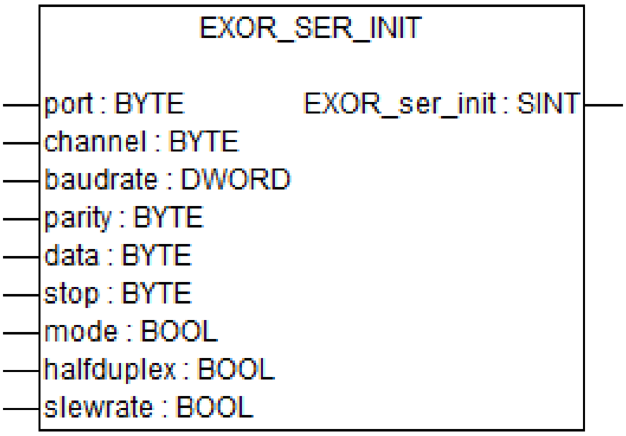
Element	Description
0001	BOOLEAN
0002	INTEGER8
0003	INTEGER16
0004	INTEGER32
0005	UNSIGNED8
0006	UNSIGNED16
0007	UNSIGNED32
0008	REAL32

Control Serial Ports trough the CODESYS V2 plc

The operator panel serial ports can be controlled trough the CODESYS internal plc, both panel integrated and add-on optional 232/485/422 port Plug-in modules can be controlled.

The control of the serial ports is done trough some specific Function blocks included into the EXOR_serial library, the library is installed as part of the Exor Target Support Package.

The CODESYS library manager includes a short help for each Function block included into the library. This chapter explains EXOR_ser_init Function block, this allows to initialize the serial port for the use trough the internal CODESYS Plc.



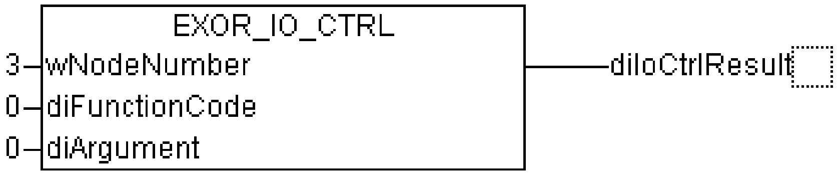
The parameters for the EXOR_ser_init function block are:

Element	Description
Port	Defines the port in use and is hardware depending: eTOP400 serie 1 = PLC port 2 = PC/Printer port eTOP500 and eTOP600 serie 1 = COM1 (integrated serial port) 2 = COM2 (optional plug-in module mounted on slot 1 or 2) 3 = COM3 (optional plug-in module mounted on slot 3 or 4)
Channel	Must be set to 255 (FF Hex)
Baudrate	Communication baudrate. Allowed values are: 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400
Parity	Communication parity. 0 = None

Element	Description
	1 = Even 2 = Odd
Data	Number of data bits. Allowed values are 7 or 8.
Stop	Number of Stop bits. Allowed values are 1 or 2
Mode	Mode of operation 0 = RS232 1 = RS485/422
Halfduplex	Defines use of RTS signal, default value is false, determines the use of RS485 or RS422 0 = Full Duplex (RS422) 1 = Half Duplex (RS485)
Slewrate	Not used, can be left to default value.

The EXOR_IO_CTRL Function

I/O boards can have special functions that need to be configured at design-time using specific Module Parameters in the CODESYS V2 PLC Configuration editor or at run-time using a function named EXOR_IO_CTRL



This function is included in the library “EXOR.lib” and has the following input parameters:

Element	Description
wNodeNumber	Each board in the CODESYS V2 PLC Configuration editor has a progressive Node Number starting from zero; this is assigned by CODESYS and can be seen in the board “Base Parameters”. This parameter identifies the board to which the “diFunctionCode” and “diArgument” parameters will be passed.
diFunctionCode	This parameter is passed to the specified board; it significance changes from board to board.
diArgument	This parameter is passed to the specified board; its significance varies from board to board.

The function has the following output parameters:

Element	Description
diIoCtrlResult	This is the return value; its significance varies from board to board and from FunctionCode to FunctionCode.

Internal Controller Hardware

This chapter describes some implementation-specific issues in the CODESYS V2 runtime kernel developed for use with the Series 500 and 600 HMI products.

Timer Resolution

The resolution of CODESYS V2 timers is 1 millisecond. When a timer value is defined it is internally translated to the corresponding number of milliseconds.

The resolution of the internal Real Time Clock is 1 millisecond allowing the maximum resolution of timers. Note that the execution time of the PLC program may apparently affect the resolution of timers.

Exor International S.p.A.
Via Monte Fiorino, 9
37057 San Giovanni Lupatoto (Verona)
Italy

info@exorint.com
phone: +39 045 8750404
fax: +39 045 8779023

Internal PLC (CODESYS)
User Manual

2017-10-13