Lab 0 - E!Cockpit Beginner’s Guide

Asean-Factori 2022

UGA
Université
Grenoble Alpes

FACTORI
4.0
Erasmus +

e!COCKPIT

Dauger Kévin - Jean-Marc Thiriet - Denis Genon-Catalot

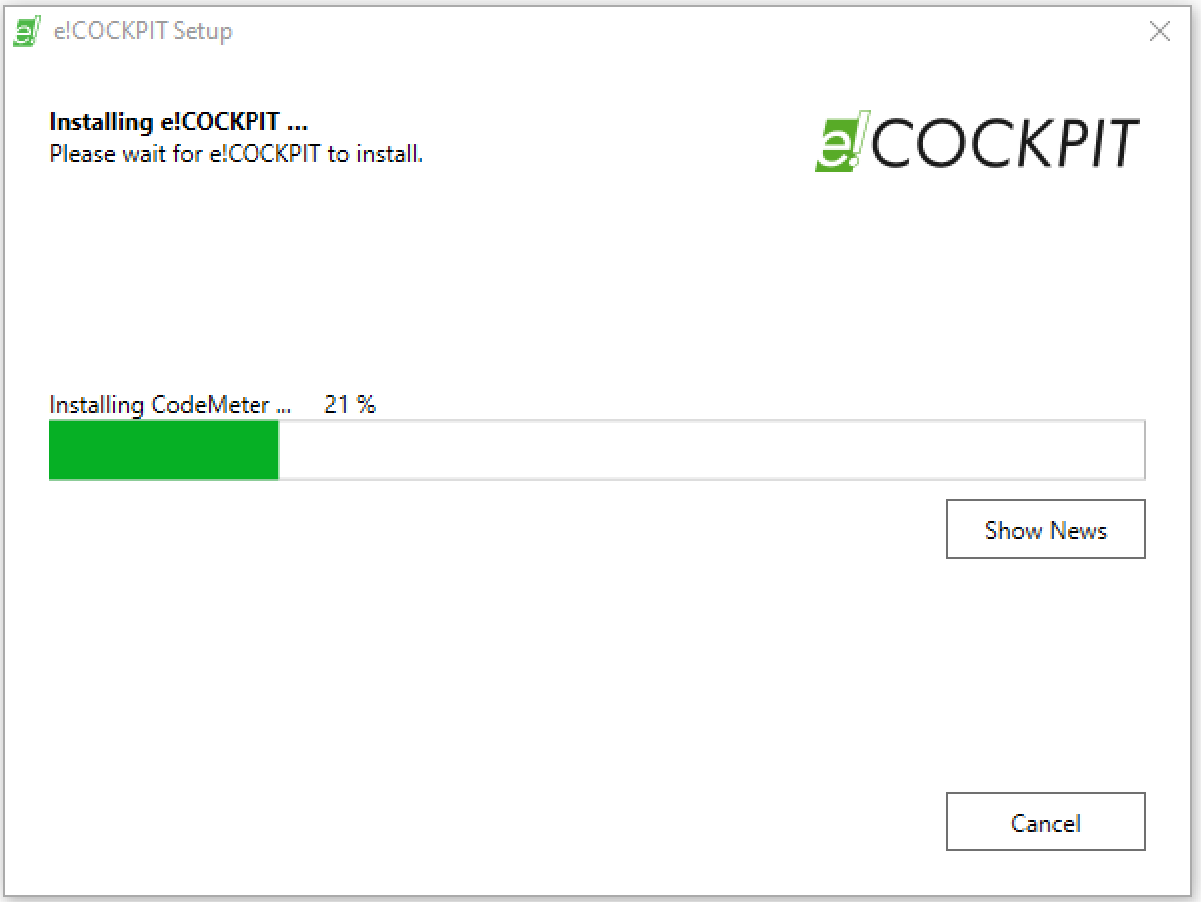
Sommaire

Introduction

This document is a complete guide to e!Cockpit, the software developed by Wago to program their PLCs. This is a really complete software with many features. You should read it before starting the labs and when you need information. Some points of this guide are detailed in the videos that accompany the training.

Software Installation

To install the software, extract the archive "*e-COCKPIT-v1.10*" downloaded before the beginning of the training. Double-click on the file "*e!COCKPIT_setup.exe*" and then on "*OK*" to select the language wanted. On the welcome page, click on "*next*", and accept the license agreement. Remember to check the "*I confirm that I am using the software exclusively for professional purposes*" box. Click on "*next*", select an installation folder, click again on "*next*" and check the box "*create a desktop shortcut*" if you need it. Then click on "*install*". You can now wait for the installation. This can take a long time depending on the power of your computer.



e!COCKPIT Setup

Installing e!COCKPIT ...
Please wait for e!COCKPIT to install.

Installing CodeMeter ... 21 %

Figure 1 : e!Cockpit being installed

When the installation is complete, you can run the software. By default you can use the 30-days evaluation license, which is quite sufficient for this training. The version given is not the last one but don't do any update because we are already using this version in the lab at Grenoble.

Begin with e!Cockpit

Create a project

When e!Cockpit is open, you can create a project by clicking on “Empty project”. You can also click on “Product Series 750” but you will have to manually find the reference of your PLC in a list. We will see why you don’t need to do this later.

If you want to change the software language, you can go to file > options > language settings (you must relaunch the software after).

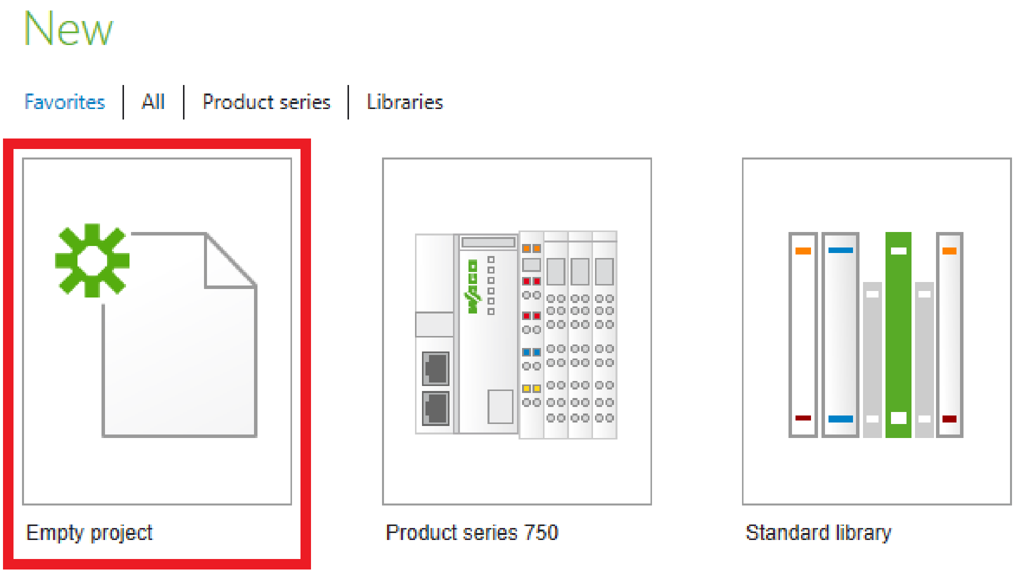


Figure 2 : Choose “Empty project” to start a new project

Organization of the windows

Network/Devices

The first window you see when your project is created is the Network/Device window. It consists of empty squares. This is the window where all the PLCs you will need in your project are represented. You can easily add a PLC (or scan your network to detect your PLCs, we will see how later), and connect the PLCs to each other.

This window is in the center of your screen. When you have a lot of PLCs in a project, it is important to know that you can zoom in or out with the little corresponding tool at the bottom right.

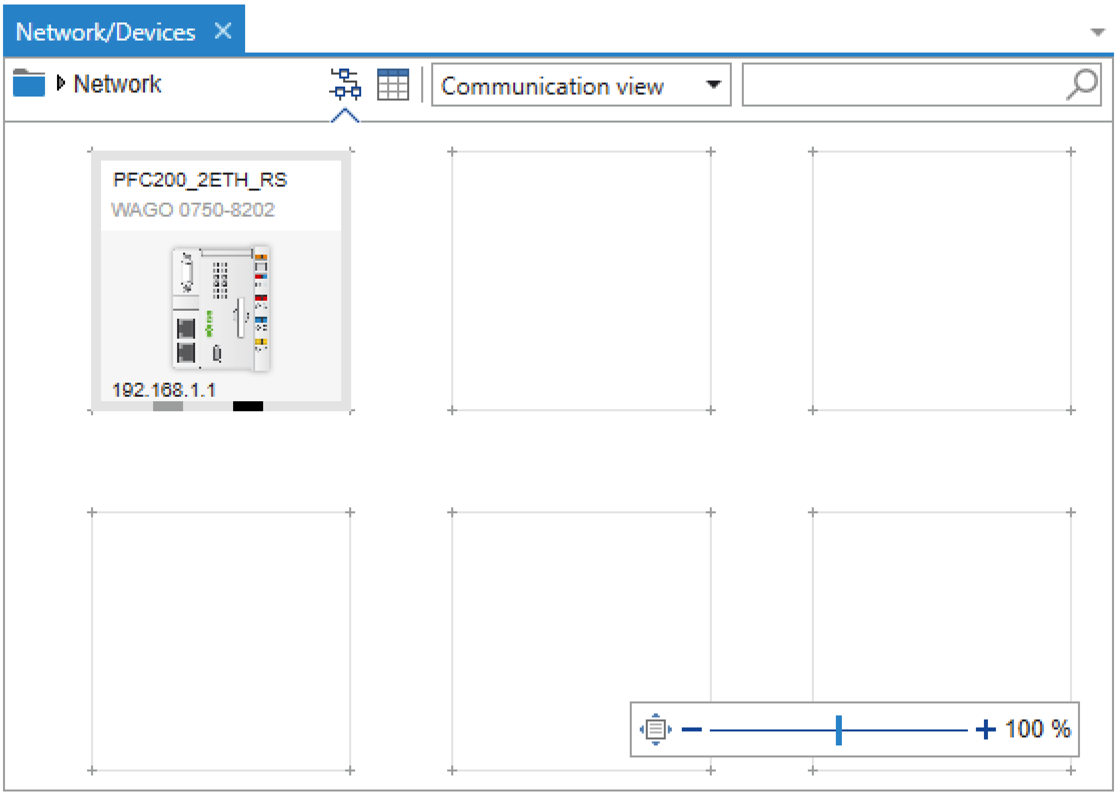


Figure 3 : The Network/Devices window

Left window

To the left of the "*Network/Devices*" window, there is the "Device structure" or the "Program structure" window (you can switch from one to the other at the bottom). The first window shows the PLCs in the project, the second window shows the program structure. The second window is used a lot during programming.

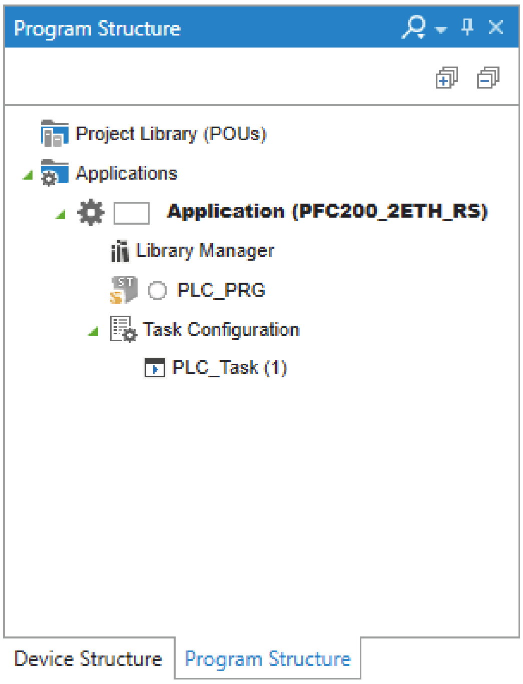


Figure 4 : The Program structure window with a PLC and a ST program (example)

Right window

To the right of the *Network/Devices* window, there is a window used in many cases. Most of the time, it is used for :

- The product catalog (a list of Wago controllers and modules from which you can select one and drag & drop it into the *Network/Devices* window)
- The settings window
- The properties window, which is mostly used for the visualizations
- The visualization tools window, where you can find the tools for creating visualizations

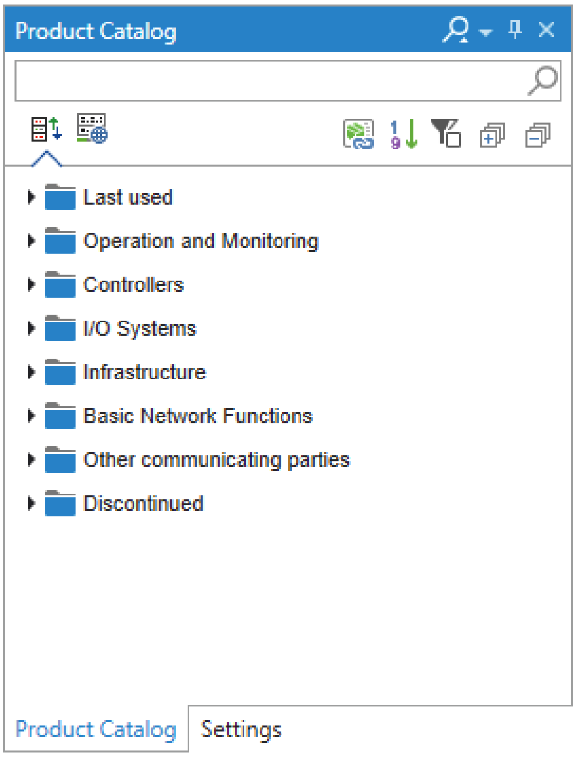


Figure 5 : The Product Catalog window

Messages

The messages window is used to display messages, warnings, and errors during programming and connection to the PLC. You can filter the messages by types, or by searching.

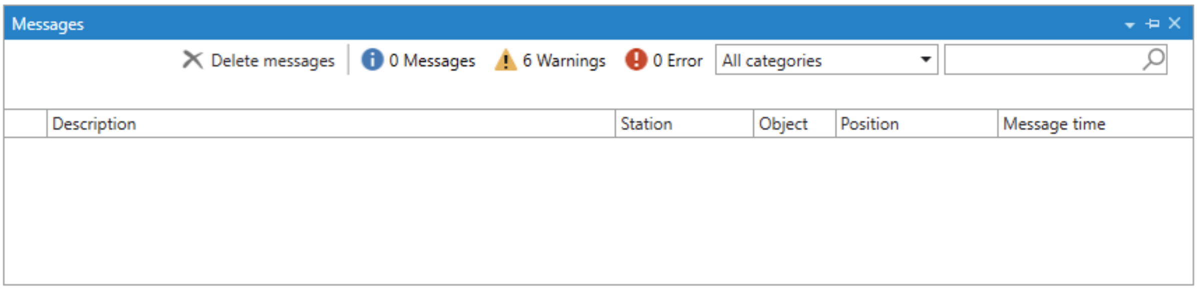


Figure 6 : The messages window

e!Cockpit tabs organization

In addition to all these windows, there are a lot of tabs used in e!Cockpit software.

Home tab

The “Home” tab is used to copy, cut, paste and delete objects and selections, to manage your workspaces and to print your programs.

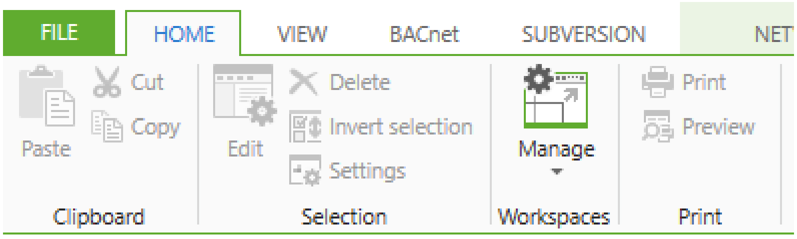


Figure 7 : The home tab

View tab

The “View” tab is used most of the time to manage your windows. You can tile horizontally or vertically different windows to compare them, for example. This is also in this tab that you can manage your licenses.

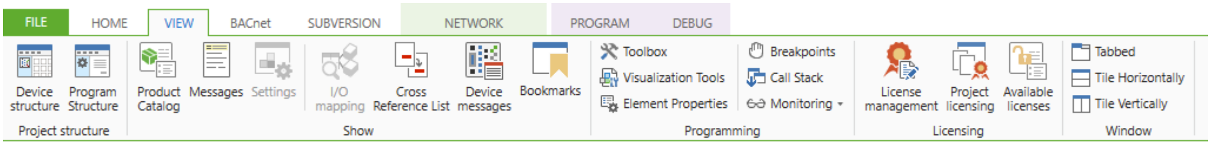


Figure 8 : The view tab

BACnet tab

This tab is used to implement the BACnet configurator. BACnet is a protocol used in the building automation. We will not use it in our labs.

Subversion tab

This tab is used to manage subversioning in your project. We will not use it in our labs.

Network tab

The “Network” tab is used to manage the network, scan your network and connect the software (your PC) to a PLC. We will use this tab in the part “Scan your network”.

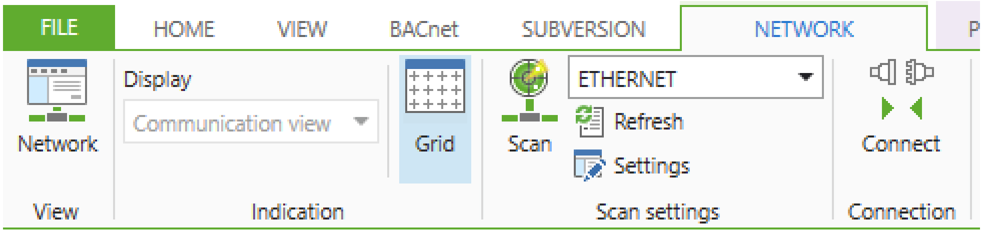


Figure 9 : The Network tab

Device Tab

The “Device” tab is one of the tabs which appears only when you are on a precise window. In this case, it appears when you double-click on a PLC in the *Network/Devices* window. In this tab, you can connect your PC to the PLC, and manage the PLC firmware. You also can get some details about your device.

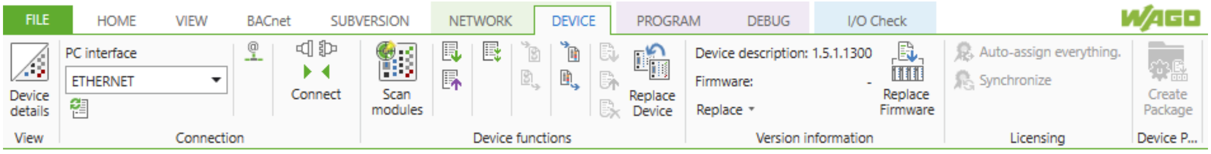

PC interface: ETHERNET
Device description: 1.5.1.1300
Firmware: -

Figure 10 : The Device tab

Modbus Slave tab

This tab is used to manage Modbus Slave. It appears when you double-click on a PLC in the *Network/Devices* window.

Program tab

The “*Program*” tab is where you compile, simulate and download your program to the PLC. You can also import and export *Codesys XML MTP* files (not used in the labs), use editing tools, run analysis and generate profilers.

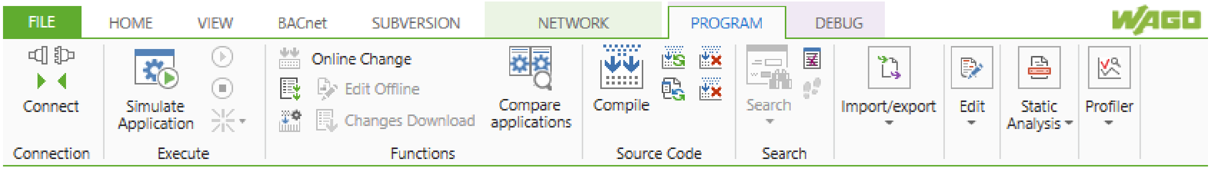


Figure 11 : The Program tab

Debug tab

e!Cockpit includes a debugging mode that can be controlled by the “*Debug*” tab. You can use breakpoints, increment and monitor variables values.

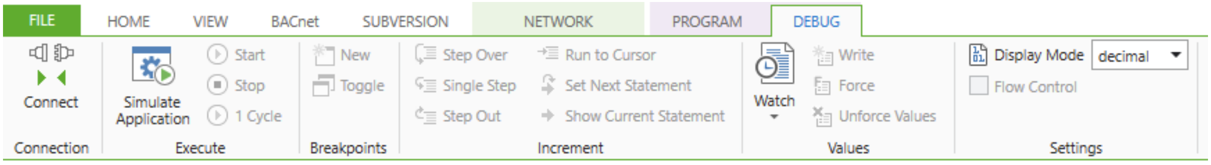

Display Mode: decimal

Figure 12 : The Debug tab

I/O Check

Finally, the “*I/O Check*” tab allows you to check the input and outputs. It appears when you double-click on a PLC in the *Network/Devices* window. We will not use it during our labs.

Scan your network

If you don’t know the IP address of your PLC, you should read the part “How to initialize the IP address of the PLC”. Otherwise, you can scan your network to automatically detect your PLC and his modules.

Settings

The first thing to do is to select “*Ethernet*” in the *Network* tab. After that, you have to select the boundaries of your search. This means you have to choose a start and an end address for your search.

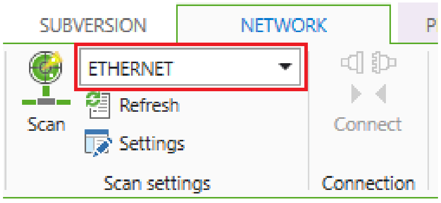
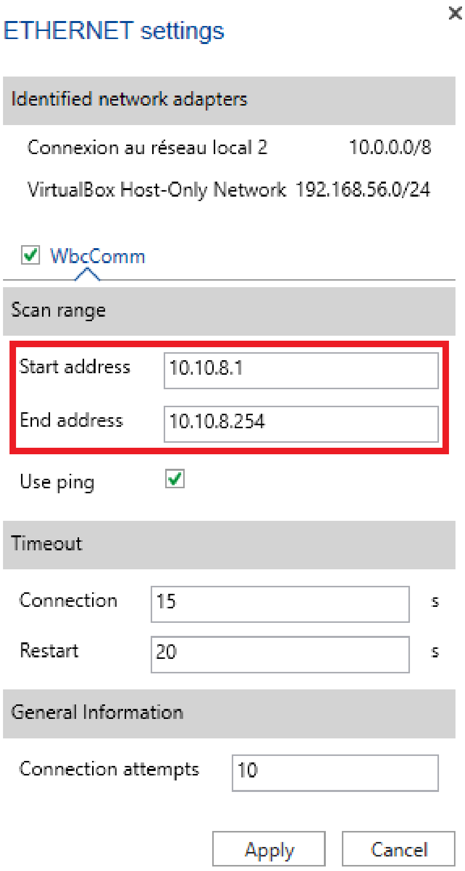


Figure 13 : The Debug tab

If you are not sure of the address of your PLC, you can scan your network from address .1 to address .254. To apply your settings, click on “*apply*”.



ETHERNET settings

Identified network adapters

Connexion au réseau local 2	10.0.0.0/8
VirtualBox Host-Only Network	192.168.56.0/24

☑ WbcComm

Scan range

Start address	10.10.8.1
End address	10.10.8.254
Use ping	☑

Timeout

Connection	15	s
Restart	20	s

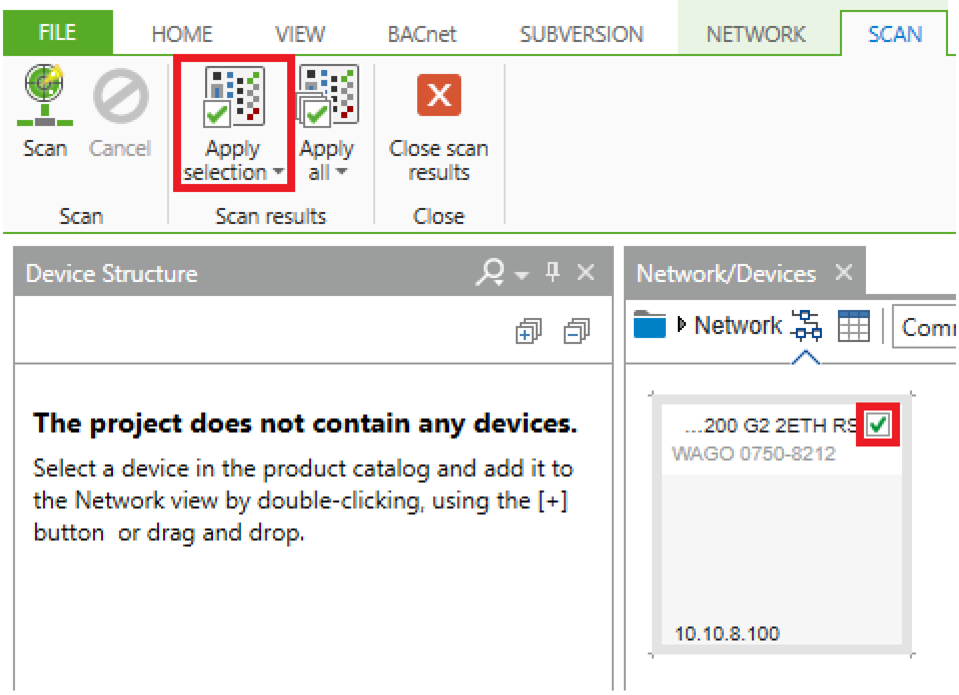
General Information

Connection attempts	10

Figure 14 : The Ethernet settings window, where you can choose the scan range

Scanning

You can now click on the “*scan*” button in the *Network* tab, and let the software search for your PLC. After a few seconds, it should appear in the window. You must select it and then click on “*apply selection*” in the *scan* tab.



The project does not contain any devices.

Select a device in the product catalog and add it to the Network view by double-clicking, using the [+] button or drag and drop.

Figure 15 : You have to apply the selection to validate the scan

After that, you can double-click on the PLC in the *Network/Devices* window. If you are not in this window anymore, you can click on the *Network* tab, and then click on “*network*” in “*view*” at the left of the tab.

Scan your modules

For the moment, the software only gets the PLC, but not his modules. To detect them, you have to click on the “device” tab, then on “scan modules”.

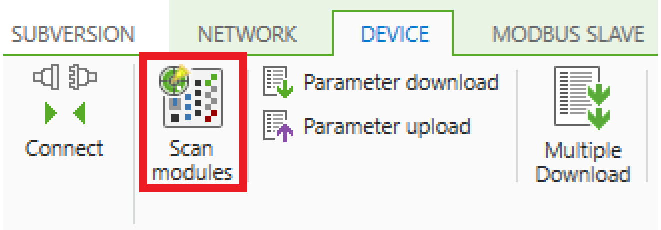
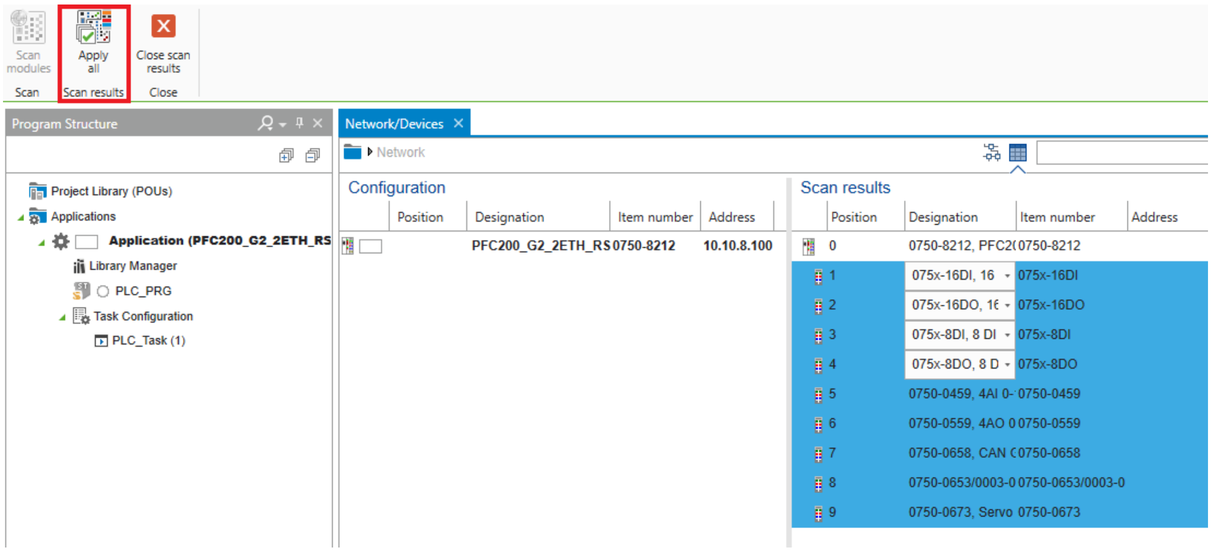


Figure 16 : The scan modules button

When the scan is completed, a new view appears in the *Network/Devices* window. In this view, you can see the list of all your modules by references. To validate the scan, click on “apply all”.



Configuration

Position	Designation	Item number	Address
	PFC200_G2_2ETH_RS	0750-8212	10.10.8.100

Scan results

Position	Designation	Item number	Address
0	0750-8212, PFC2(	0750-8212	
1	075x-16DI, 16	075x-16DI	
2	075x-16DO, 1€	075x-16DO	
3	075x-8DI, 8 DI	075x-8DI	
4	075x-8DO, 8 D	075x-8DO	
5	0750-0459, 4AI 0-	0750-0459	
6	0750-0559, 4AO 0	0750-0559	
7	0750-0658, CAN (	0750-0658	
8	0750-0653/0003-0	0750-0653/0003-0	
9	0750-0673, Servo	0750-0673	

Figure 17 : Always think to validate the scan by clicking on "apply all"

If you click again on the PLC in the normal *Network/Devices* view, you will now see a diagram of your PLC and all its modules (figure 16). If a module is missing, you must check that it is correctly installed and connected to the PLC.

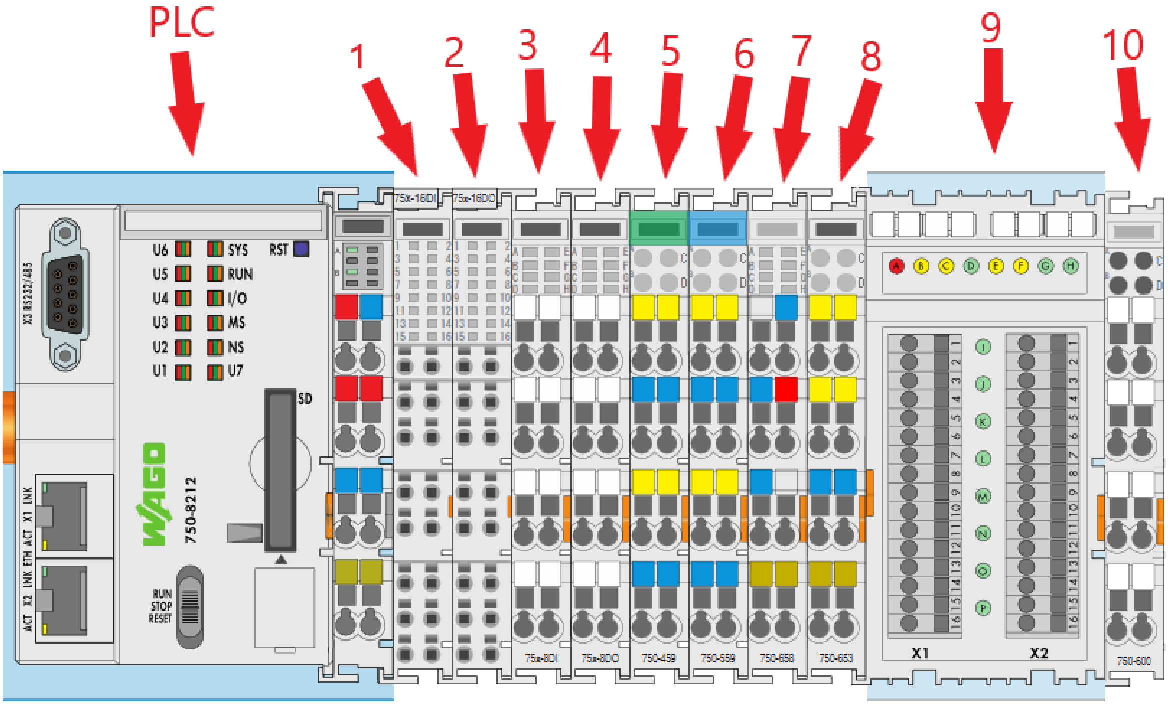


Figure 18 : The schematic view should corresponds to your real PLC and modules

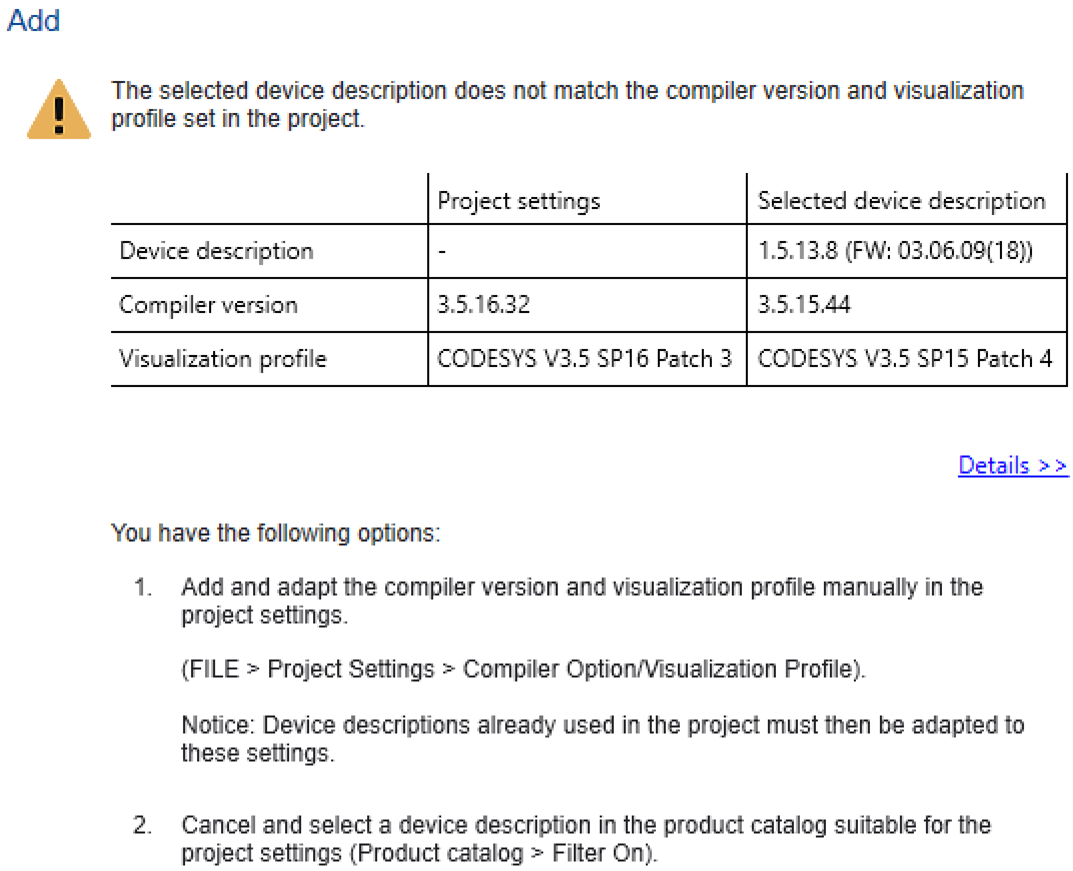
In the figure above, you can see the list of the modules scanned by the software. You can see different types of modules :

1. 16 Inputs 24VDC
2. 16 Outputs 24VDC
3. 8 Inputs 24VDC
4. 8 Outputs 24VDC
5. 4 Analog Inputs

6. 4 Analog Outputs
7. CAN Bus Gateway
8. RS485 Interface
9. Servo Stepper Controller
10. End module

Firmware update

It is possible that there is a difference between the PLC firmware and that used by the software. If there is, you will get a message like the one in figure 17 below after requesting the PLC scan.



Add

The selected device description does not match the compiler version and visualization profile set in the project.

	Project settings	Selected device description
Device description	-	1.5.13.8 (FW: 03.06.09(18))
Compiler version	3.5.16.32	3.5.15.44
Visualization profile	CODESYS V3.5 SP16 Patch 3	CODESYS V3.5 SP15 Patch 4

You have the following options:

1. Add and adapt the compiler version and visualization profile manually in the project settings.

   (FILE > Project Settings > Compiler Option/Visualization Profile).

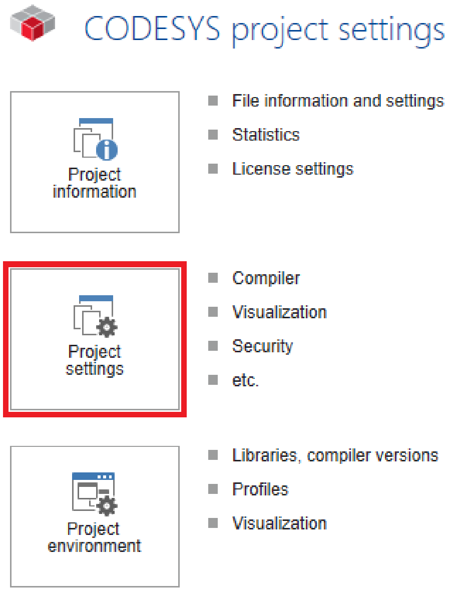
   Notice: Device descriptions already used in the project must then be adapted to these settings.

2. Cancel and select a device description in the product catalog suitable for the project settings (Product catalog > Filter On).

Figure 19 : Firmware error message

In this message, you will get different crucial information. You should take a screenshot or write on a paper the "*selected device description*" column, because you will use these informations to correct this error. You can click on "add" before doing the update.

Click on the "file" tab, then look for "project settings". Click again on "project settings" (figure 18) and a new window will appear (figures 19 and 20).



CODESYS project settings

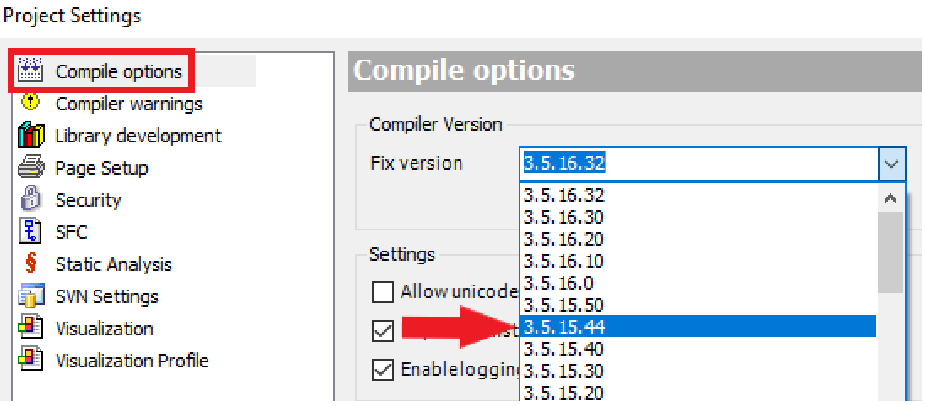
Project information
- File information and settings
- Statistics
- License settings

Project settings
- Compiler
- Visualization
- Security
- etc.

Project environment
- Libraries, compiler versions
- Profiles
- Visualization

Figure 20 : Click on “project settings”

In the project settings window, click on “compile options”. Here, you can set the compiler version for your PLC. You have to search in the list for the corresponding version.



Project Settings

- Compile options
- Compiler warnings
- Library development
- Page Setup
- Security
- SFC
- Static Analysis
- SVN Settings
- Visualization
- Visualization Profile

Compile options

Compiler Version

Fix version: 3.5.16.32

3.5.16.32
3.5.16.30
3.5.16.20
3.5.16.10
3.5.16.0
3.5.15.50
3.5.15.44
3.5.15.40
3.5.15.30
3.5.15.20

Settings

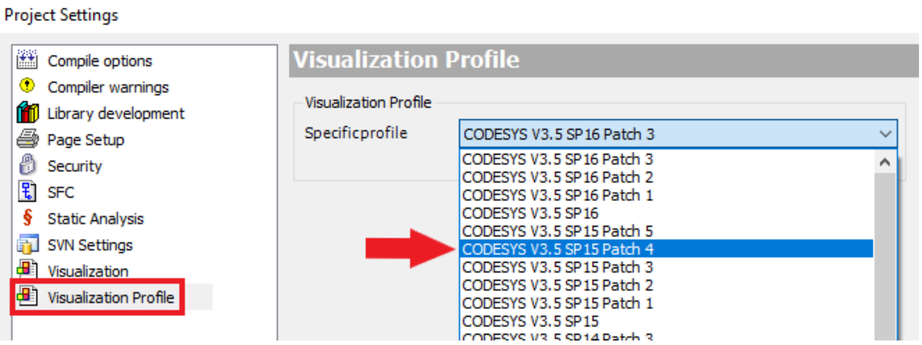
☐ Allow unicode

☑

☑ Enablelogging

Figure 21 : In “compile options”, you can select the corresponding compiler version

Now, click on “visualization profile”. In this tab, you can modify the visualization profile version.



Project Settings

- Compile options
- Compiler warnings
- Library development
- Page Setup
- Security
- SFC
- Static Analysis
- SVN Settings
- Visualization
- Visualization Profile

Visualization Profile

Visualization Profile

Specificprofile: CODESYS V3.5 SP16 Patch 3

CODESYS V3.5 SP16 Patch 3
CODESYS V3.5 SP16 Patch 2
CODESYS V3.5 SP16 Patch 1
CODESYS V3.5 SP16
CODESYS V3.5 SP15 Patch 5
CODESYS V3.5 SP15 Patch 4
CODESYS V3.5 SP15 Patch 3
CODESYS V3.5 SP15 Patch 2
CODESYS V3.5 SP15 Patch 1
CODESYS V3.5 SP15
CODESYS V3.5 SP14 Patch 3

Figure 22 : In “visualization profile”, you can select the corresponding visualization version

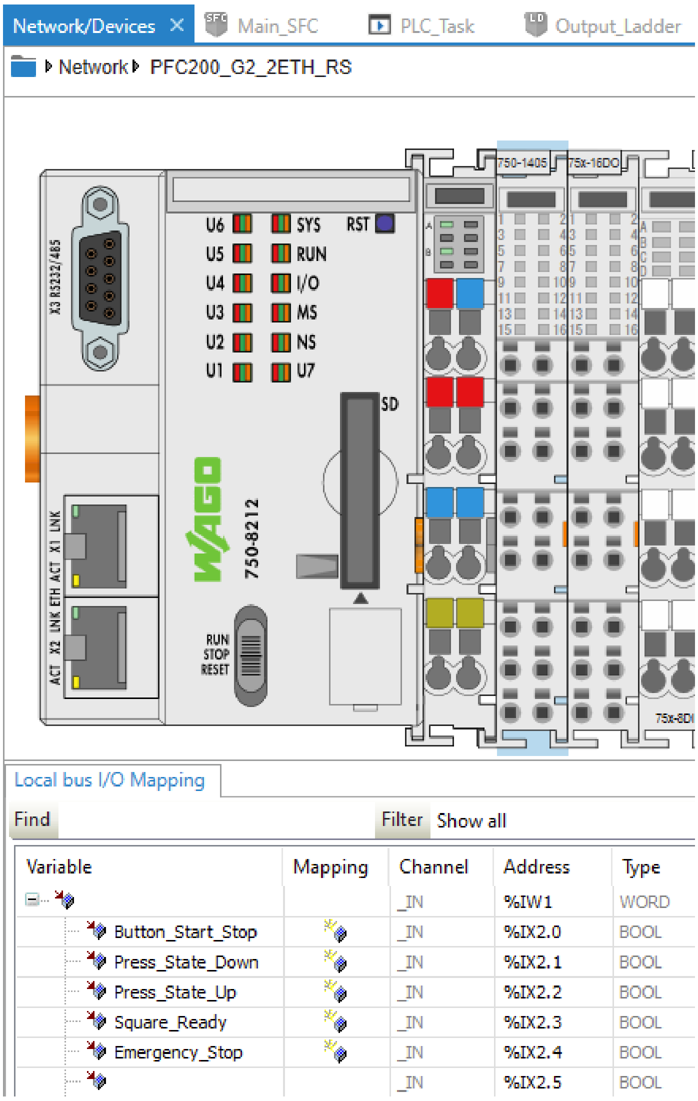
You can check that the update has worked by clicking on “compile” (even without any program) in the “program” tab. If this message no longer appears, the software firmware is in accordance with the one of the PLC.

Variable declaration

At this point, you should have a schematic with your PLC and all your modules without errors. The next point is how to add input and output variables to your projects.

Input Variables

Among your modules, there should be an input module. In the schematic view of your PLC in the *Network/Devices* window, you can click on this module to display the “Local bus I/O Mapping” tab below.

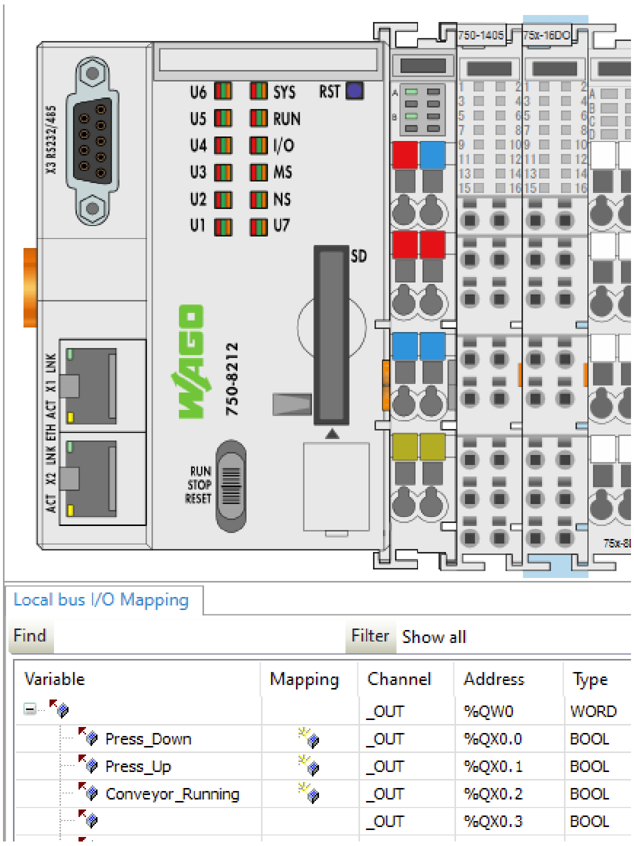


Variable	Mapping	Channel	Address	Type
		_IN	%IW1	WORD
Button_Start_Stop		_IN	%IX2.0	BOOL
Press_State_Down		_IN	%IX2.1	BOOL
Press_State_Up		_IN	%IX2.2	BOOL
Square_Ready		_IN	%IX2.3	BOOL
Emergency_Stop		_IN	%IX2.4	BOOL
		_IN	%IX2.5	BOOL

Figure 23 : You can add your input variables easily

In this tab, there are all the slots available for variables. The addresses are already written, the type too (BOOL), and all you have to do is to name your variables. For example, there are three input variables in this project (figure 21).

Output Variables



Local bus I/O Mapping

Find   Filter Show all

Variable	Mapping	Channel	Address	Type
		_OUT	%QW0	WORD
Press_Down		_OUT	%QX0.0	BOOL
Press_Up		_OUT	%QX0.1	BOOL
Conveyor_Running		_OUT	%QX0.2	BOOL
		_OUT	%QX0.3	BOOL

Figure 24 : You can add your output variables easily

For the outputs, this is the same thing. You only have to click on the output module and name your outputs. For example, there are two boolean outputs in this project (figure 22).

Tips about variables declaration

If you want to delete a variable, you should **NEVER** use the "*delete*" key on your keyboard. The reason is that e!Cockpit thinks that you want to delete the module you selected even if you clicked on a variable. If you click on "*delete*", all your variables will disappear with the module, and you will have to write them again. The proper way to delete a variable is to double-click on its name and use the "*backspace*" key instead of "*delete*".

Perhaps you are wondering how to memorize a value or a state for example. To do that, you can use internal variables. In e!Cockpit, these variables are declared directly in the program in Structured Text language, so we will explain how to use them in the next part of the guide.

Programming

That's it ! You can now start programming something in e!Cockpit. For this part and for the rest of the guide, we will use a simple SFC program as support, in addition to some

Ladder and Structured Text scripts. The process consists of a hydraulic press that is used to compress boxes on a conveyor. We will use these inputs and outputs :

Inputs	Outputs
Button_Start_Stop	Press_Down
Press_State_Down	Press_Up
Press_State_Up	Conveyor_Running
Square_Ready	
Emergency_Stop	

SFC

In the “*device structure*” window (on the left of your screen), you can click on “*program structure*” at the bottom to display the tree structure of your program. By default, in the “applications” folder, a ST file named “*PLC_PRG*” is present. The first task is to delete it with “delete” or by right-clicking on it and selecting “*delete*”. Indeed, we want to program in SFC.

To do this, right-click on “*Application (reference of your PLC)*” and select “*POU*” in “*add a new item*”.

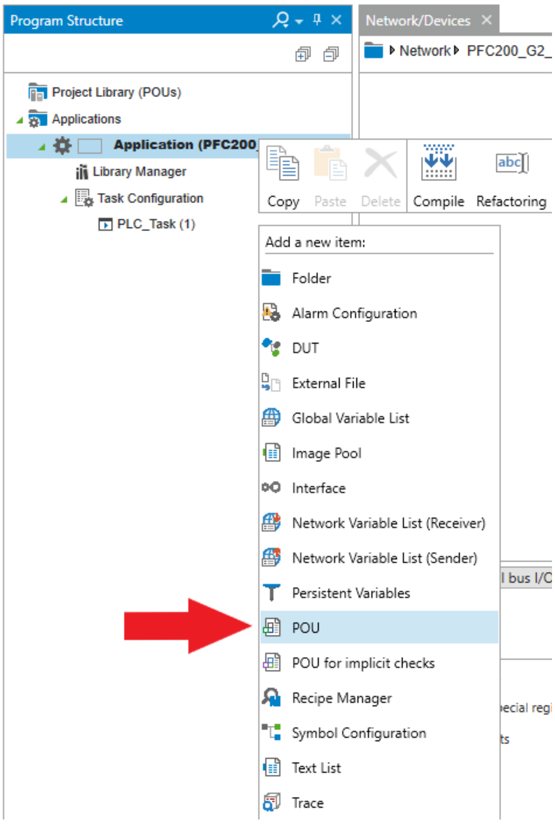
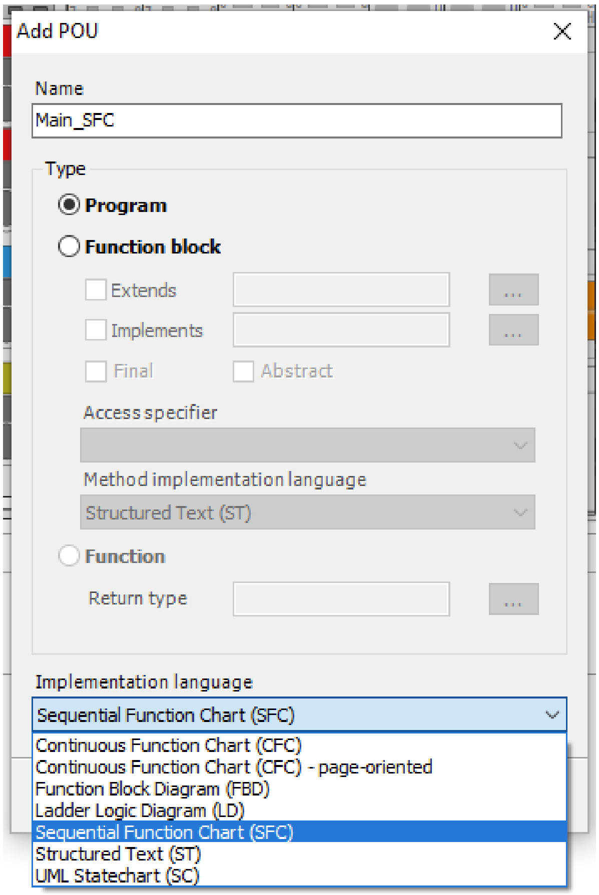


Figure 25 : Select “POU” to add a SFC program

"*POU*" means "*Program Organization Units*". By clicking on it, you can choose the programming language you want to use and the name of the "*POU*". It is advisable to write a clear name, for example, for your main program, write "*Main_SFC*".



Add POU

Name: Main_SFC

Type

● Program

○ Function block

□ Extends

□ Implements

□ Final □ Abstract

Access specifier

Method implementation language: Structured Text (ST)

○ Function

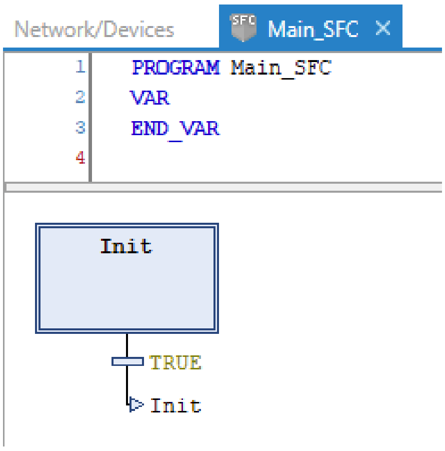
Return type

Implementation language: Sequential Function Chart (SFC)

- Continuous Function Chart (CFC)
- Continuous Function Chart (CFC) - page-oriented
- Function Block Diagram (FBD)
- Ladder Logic Diagram (LD)
- Sequential Function Chart (SFC)
- Structured Text (ST)
- UML Statechart (SC)

Figure 26 : You can change the name and select "Sequential Function Chart"

A new window appears in the center of your screen with the name of your program. It is in this window where you will create your SFC. The window is separated in two parts. At the top, there is a Structured Text program where you can declare internal variables. At the bottom, there is the SFC diagram.



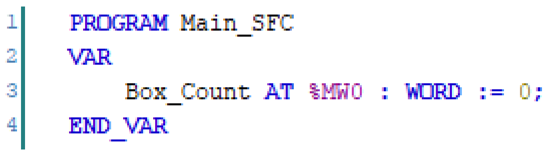
Network/Devices | Main_SFC

```
PROGRAM Main_SFC
VAR
END_VAR

```

Figure 27 : The SFC program window

To declare an internal variable, just write the name of your variable, its %M address, the type and if necessary, a default value.



```
PROGRAM Main_SFC
VAR
    Box_Count AT %MW0 : WORD := 0;
END_VAR
```

Figure 28 : Example of an internal value which gets the number of boxes detected

To realize the SFC program, a tab named “toolbox” should appear at the bottom of the right window. In this window, you have all the tools used to create a SFC program such as steps, transitions, jumps, etc.

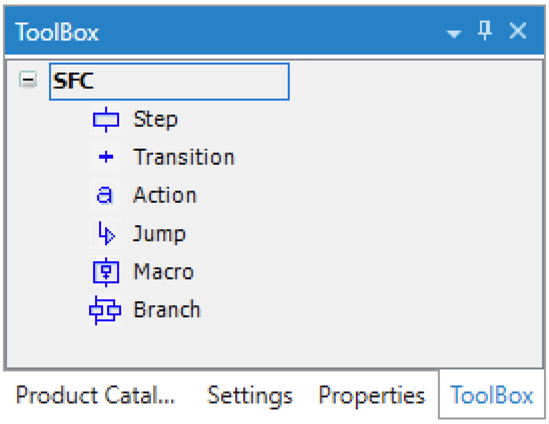


Figure 29 : The SFC toolbox window

To complete your program, you can also right-click directly on a transition or step to select what you want to add.

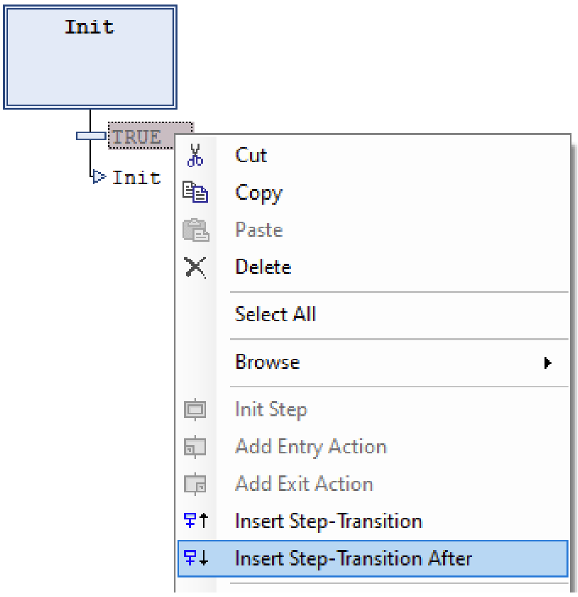


Figure 30 : You can add a Step-Transition block with a right-click on a transition

You can of course implement AND or OR branches, add macros or add a jump to a new state. The transition can be in Ladder, ST, FBD or IL, we will see how to use the 2 first one in e!Cockpit in the next parts. Note that if your transition contains only one variable, you can simply write it as a transition name.

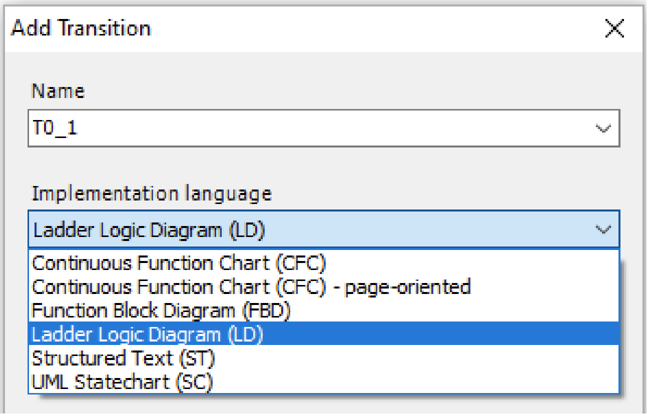
In e!Cockpit, you can associate the actions (the outputs) directly to a step in the SFC chart. This works, but it's not recommended because it is hard to debug for large projects. We recommend controlling the values of the outputs with a ladder program (see the next part).

Ladder

In e!Cockpit, you can use Ladder to control the outputs and create the SFC transitions.

Transitions

To create a Ladder transition, right-click on your SFC program in the "Program Structure" view and select "transition".


Add Transition

Name: T0_1

Implementation language: Ladder Logic Diagram (LD)

Continuous Function Chart (CFC)
Continuous Function Chart (CFC) - page-oriented
Function Block Diagram (FBD)
Ladder Logic Diagram (LD)
Structured Text (ST)
UML Statechart (SC)

Figure 31 : You can write the transitions with a lot of different languages

A new window appears. First of all, you have to add a coil with the name of the transition. You can do this by right-clicking on the top of the window in the network "1" (the line at the top of the window).

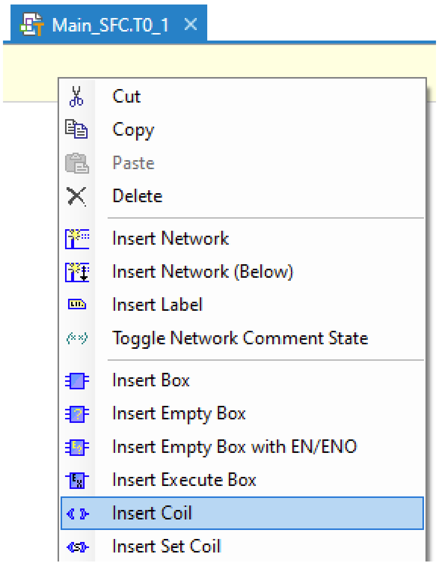


Figure 32 : You have to insert a coil with the name of the transition

To rename the coil, simply click on the "???" above the coil and type the name. A good way to name your transitions is to combine the number of the step before the transition with the step after. In this case, since this is the transition between the step "init" (step 0) to step 1, the name of the transition is "T1_0".

Now, you can add the contacts you need by right-clicking on the wire that appeared and name them with the inputs you declared before. You can also use an output as an input if necessary. You can add contact in parallel or in series to create OR and AND equations. It is also possible to use some special blocks like TON, TOF (timers), CTU, CTD (counters), in "Ladder Elements" in the "toolbox" window.

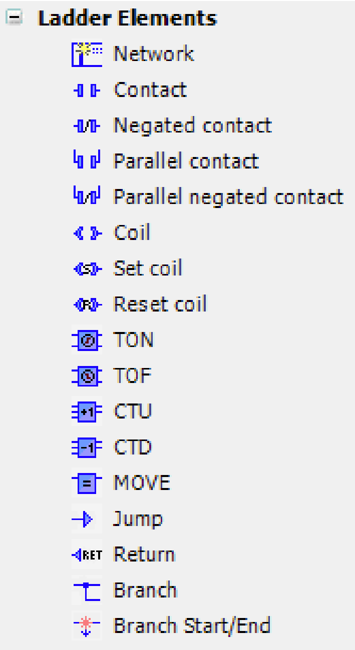

Ladder Elements
- Network
- Contact
- Negated contact
- Parallel contact
- Parallel negated contact
- Coil
- Set coil
- Reset coil
- TON
- TOF
- CTU
- CTD
- MOVE
- Jump
- Return
- Branch
- Branch Start/End

Figure 33 : List of all the blocks disponible in Ladder in e!Cockpit

If you have a lot of inputs and outputs and you don't remember all of them, you can click on the "..." button right next to the "???" and click on "*IoConfig_Globals_Mapping*". A list of all your inputs and outputs will appear. To simplify the visibility of the diagram, remember to uncheck "*Insert with namespace prefix*".

Outputs

It's recommended to create a Ladder diagram dedicated to control the outputs. To do that, you can right-click on "*Application*" in the "*Program Structure*" window, select "*POU*", and name it "*Output_Ladder*" (for example).

Now, you can associate an output to a state of the *Main_SFC* diagram. To do this, simply add a contact and name it "*Main_SFC.Step1.x*". Indeed, you have to name the program, then the step and finally add "*.x*" which means "when this state is activated".

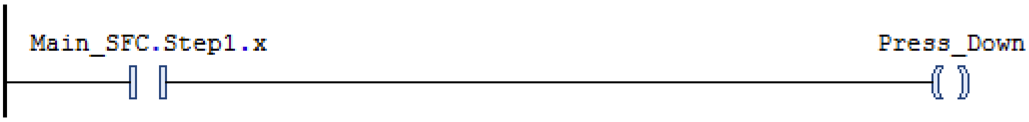
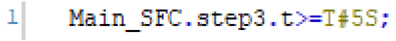


Figure 34 : Example where step 1 activates the press

To control multiple outputs, you can right click in the middle of the window and click on “Insert Network”. Remember that you should **NOT** insert the same output twice.

Structured Text

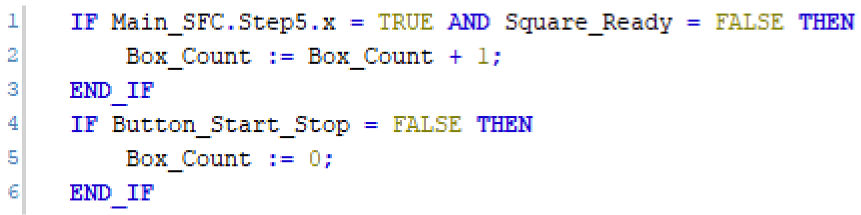
You can also use Structured Text in e!Cockpit to program a PLC. It can be really useful to create timers in a SFC transition for example.



```
Main_SFC.step3.t>=T#5S;
```

Figure 35 : Timer in Structured Text in a transition

In our example, we want to know how many boxes were pressed during the process, and reset the count when the process is stopped. The easiest way to program this is to create a new “*POU*” in Structured Text named “*Box_Counting*”, and create an internal variable of type *word*, and code a small script as seen in figure 34.



```
IF Main_SFC.Step5.x = TRUE AND Square_Ready = FALSE THEN
    Box_Count := Box_Count + 1;
END_IF
IF Button_Start_Stop = FALSE THEN
    Box_Count := 0;
END_IF
```

Figure 36 : Structured Text script to count the number of boxes

How to make different scripts work together ?

To make all these different scripts work together, you have to double-click on “*PLC_Task*” in the “*Program Structure*” window. If you never modified anything in this window, there should be one “*POU*” in the table named “*PLC_PRG*” (the name of the first deleted ST program). You can delete it by clicking on it and then on “Remove Call”.

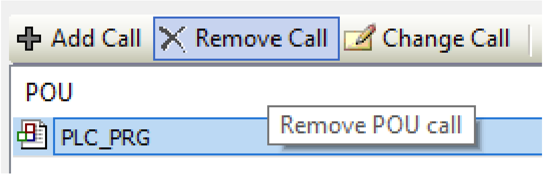


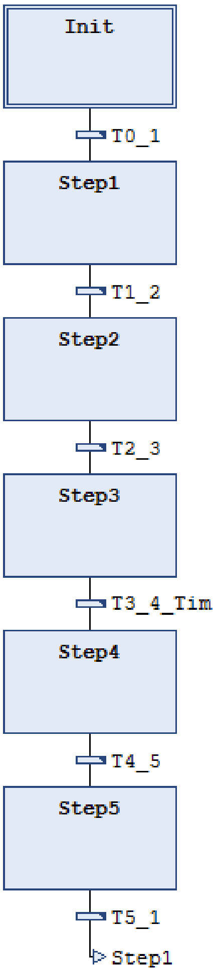
Figure 37 : You should remove the default name to avoid an error

In this window, you should add your diagrams and programs in order of priority. For example, in our case, we should add the main SFC first, before the output Ladder, and finally the box counting script. To do this, click on “*add call*” and select firstly *Main_SFC*, then click again on “*add call*” and select “*Output_Ladder*”, and again for “*Box_Counting*”. After that, you can change the order by selecting a program and clicking on “move up” or “move down”.

Warning : If you have several programs in your project and you forget one of them in the PLC_Task, there will be no error but this program will not be loaded during the process.

Example

To program the process described before, we have to create a little SFC of 6 steps. Each state is separated by a Ladder transition, except for the transition between step 3 and step 4, because it is a 5s timer written in Structured Text. The Output Ladder can be seen in appendix 1 (notice the emergency stop and the conveyor protection : the conveyor can't run if the press is down). There is also a little Structured Text program to count the number of boxes (figure 34).



During step 1, the conveyor carries the box to the press.

When the sensor detects the box, the conveyor stops and the press descends to the Press_State_Down sensor (step 2).

When this sensor is activated, a timer of 5 seconds starts to continue compressing the box (step 3).

After the timer, the program goes to step 4, where the press goes up to the Press_State_Up sensor.

When it is reached, the conveyor runs again and the cycle starts again.

Figure 38 : The SFC program of this project

Compiling

To check your programs for errors, you must compile them by clicking on "*compile*" in the program tab. If there are errors, the corresponding messages will appear in the "*messages*" window. You can click on "*display message view*" in the yellow box that appears at the top of the window in case of messages.

Visualisations

To supervise your process and first of all to simulate your program, you can create visualization. This is a schematic screen representing your process with lights and animations.

Creation

To create a visualization, right-click on "*Application*" and select "*visualization*" at the bottom of the list. You can give a name to your visualization. Two objects should appear when you validate the name, a "*Visualization Manager*", and the visualization itself. The "*Visualization Manager*" is used to configure the "*WebVisu*" (the ability to see the visualization from an internet navigator. Notice that a new task has appeared in "Task Configuration".

Double-click on the visualization to open it. An empty screen appears. On the right of the screen you can see the "Visualization Toolbox", where we will find all the objects that we need. The objects are classified by categories :

- Basic : Basics geometrical shapes
- Common Controls : Basic Human Machine Interface (HMI) objects (button, slider, text field…)
- Alarm Manager : Alarm tools
- Measurement Controls : Configurable displays to check input or output values
- Lamps/Switches/Bitmaps : High quality lamps and switches
- Special Controls : Less common objects like a trace for a time dependent variable, a web browser…
- Date/Time Controls : Calendar, Clock…
- Symbols : Various symbols like arrows, padlocks, speakers…
- WagoIcons : Various icons by Wago
- WagoSysModule_75x_658 : It contains only "tpl_Diagnosis", a screen used to check counter and timers values
- VisuDialogs : Various HMI screens like a numpad, a keypad, a login screen, a messagebox…

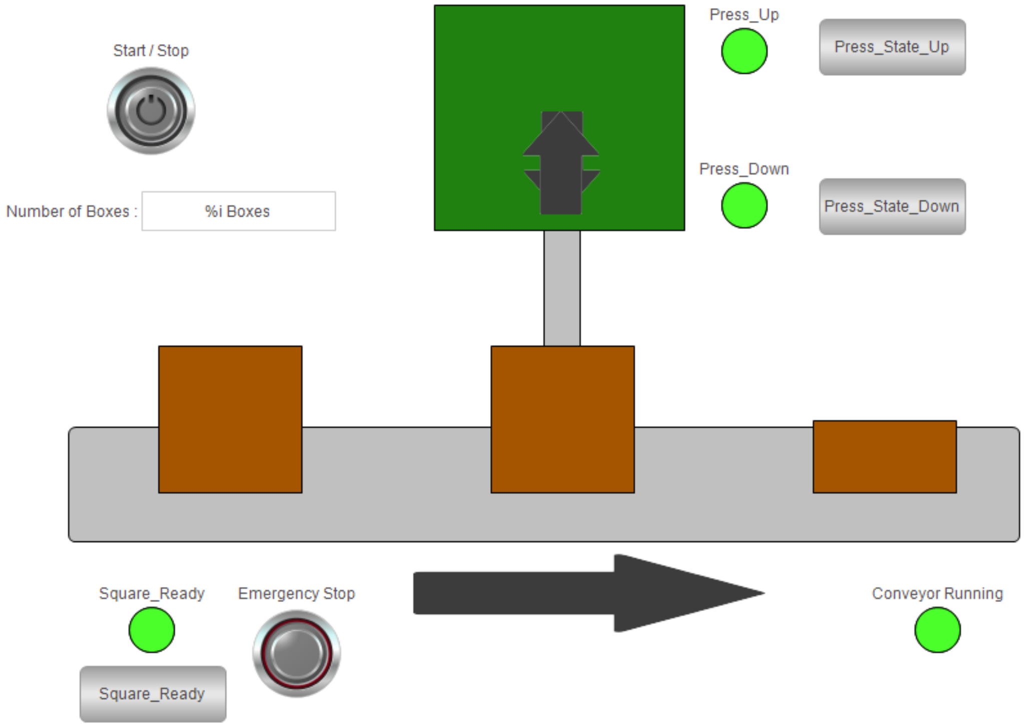


Figure 39 : The visualization made for this process

If you don’t see all the categories, you need to click on file, then project settings, project environment, search for “*Visualization Symbols*” and check all the boxes, and then apply. Keep in mind that other symbols may appear with new versions.

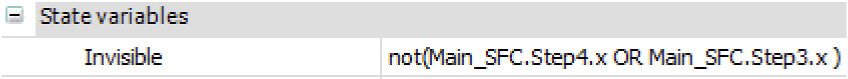
Animations

You can choose any of these objects to schematize your process, but you should consider that the more complex your visualization, the harder it is to debug it when something goes wrong. Most of the time, we use simple shapes to represent machines and processes. For example, you can see the visualization I made for this little project (figure 37).

The conveyor is represented by a simple gray square, and an arrow is visible when it is running. To animate some objects, you can superimpose different shapes. For example, the press is composed of two different gray rectangles. The first one represents the “up” position, so it is always visible. The second one represents the “down” position, it is over the first one and it is only visible when the corresponding variable is true.

To place an object, find it in the visualization toolbox, and drag and drop it to the screen. You can modify its shape and then move it pixel by pixel with the arrow keys (or 10 pixels by 10 pixels by pressing shift and the arrow keys). To, for example, modify the color of an object, select an object, click on the “*properties*” tab of the right window and then on “*Colors*”, “*Normal State*”, and “Fill color”. Double-click on the blank rectangle to choose the color wanted.

To display or not an object, you can select it and then click on the “*Properties*” tab. Look for “*State variables*” (at the bottom of the list). There is a slot where you can write the name of a variable, and the object will be invisible when this variable is true. To do the opposite, you should write “not()” and type your variable between the parenthesis. As you can see in the figure 38, you can combine different variables with “OR” and “AND”.



⊟ State variables	
Invisible	not(Main_SFC.Step4.x OR Main_SFC.Step3.x)

Figure 40 : State variables to animate the press

There are a lot of modifications disponible in the properties tab, we can’t detail them all here because it depends on the selected object. It is quite easy to understand when you try a new feature, you just need time to discover these options.

Simulating

When your visualization is complete, you can simulate your process to confirm that it works without any problem. Click on “*Simulate Application*” in the “*Program*” tab. If the button is grayed out, don't worry, you only have to click on “*Application*” in the “*Program Structure*” window. This sometimes appears when you have different applications, because the software does not know which application should be simulated.

When the software is in simulation mode, you must click on "*Start*" to begin. You can check your visualization, but always remember to check your SFC and the other programs as well. In e!Cockpit, variables values are written next to their name in the scripts when the process is running. This is really useful to observe what happens if there is a problem in the process.

Connecting to the PLC

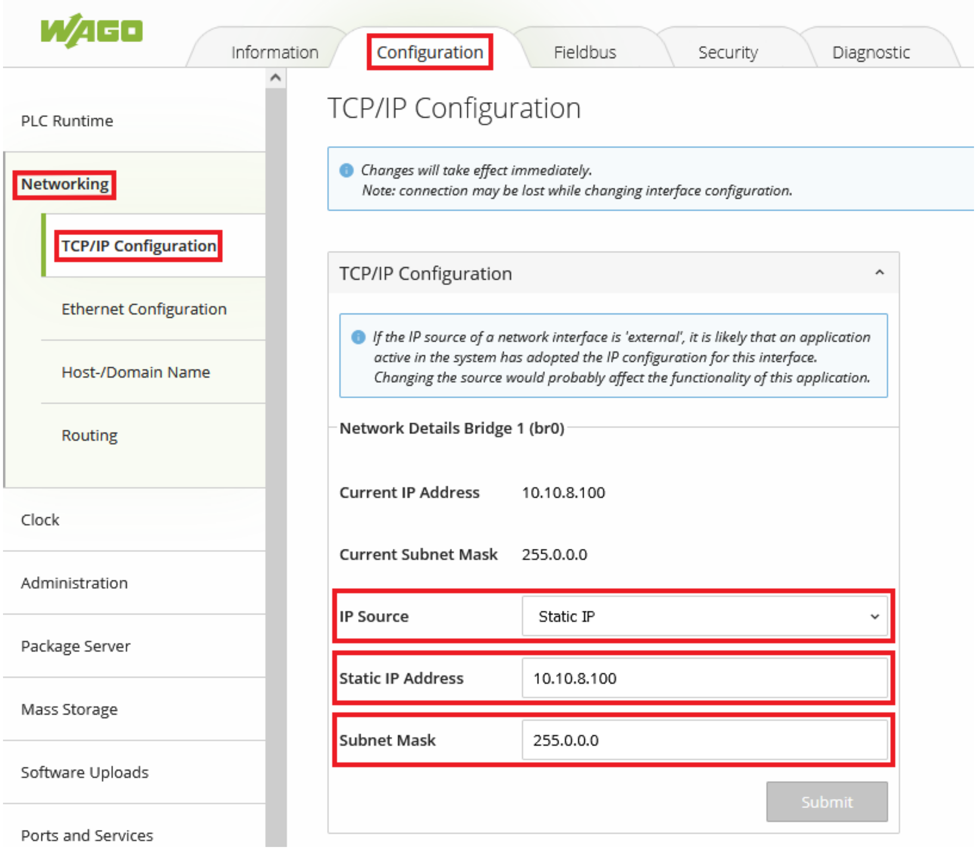
At this stage, if the simulation does not present any problem, you can connect the software to the PLC. Thanks to the first points of this guide, the software already knows the address of the PLC, you only have to click on "Connect" in the "Program" tab, and then on "Start". If the program does not start, check your PLC, you probably forgot to put the switch on "Start" on the front of the PLC.

List of tips

How to initialize the IP address of the PLC ?

If you don't know the IP address of your PLC, you must initialize it. To do this, you have to connect the PLC to your network with a RJ45 cable, set the switch on the PLC to "*Stop*" and press the "*Reset*" button for 8 seconds with a screwdriver. When the "*SYS*" led blinks in yellow, the PLC is in "standard address mode" as long as it is powered. In this mode, the address of the PLC is temporarily "192.168.1.17".

You can now connect to the web configuration interface of your PLC. Just type "https://192.168.1.17/wbm" in your web browser. By default, the login is "admin", and the password is "wago".



TCP/IP Configuration

Changes will take effect immediately.
Note: connection may be lost while changing interface configuration.

TCP/IP Configuration

If the IP source of a network interface is 'external', it is likely that an application active in the system has adopted the IP configuration for this interface. Changing the source would probably affect the functionality of this application.

Network Details Bridge 1 (br0)

Current IP Address 10.10.8.100

Current Subnet Mask 255.0.0.0

IP Source Static IP

Static IP Address 10.10.8.100

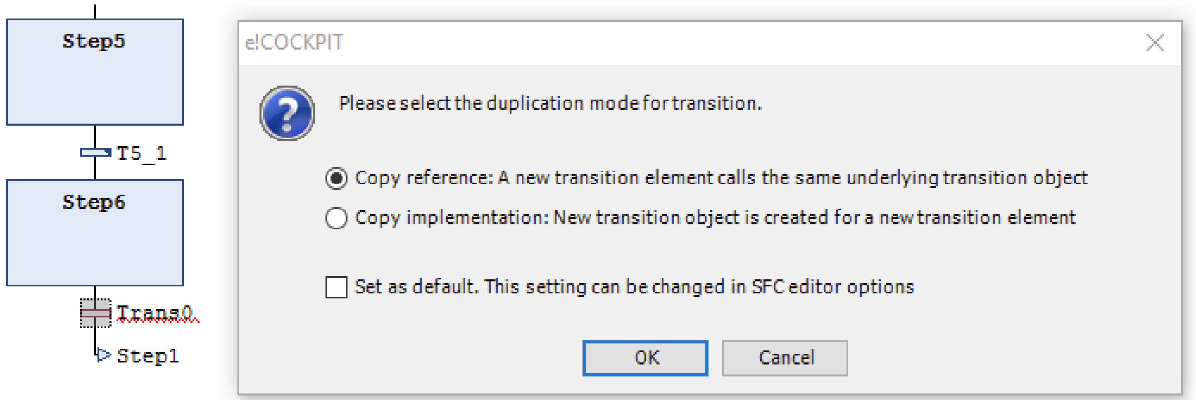
Subnet Mask 255.0.0.0

Figure 41 : View of the web administrator interface

In the interface, click on Configuration > Networking > TCP/IP Configuration and type the address you want for your PLC. Remember that you must be on the same network as your PC to interact with the PLC. When the configuration is complete, click on “*Submit*” and turn off the power to the PLC. Set the switch to “*Run*” before putting it back on. You can check that the connection is working with the “*ping*” command.

Various tips

- With Ladder, you can turn a normal contact into a negative contact by selecting it and clicking on “/”. If you click on “P” or “N”, you can transform the contact into a “Rising Edge” (P) or “falling edge” detection (N).
- In SFC, it is possible to create a Ladder or ST transition directly by double-clicking on a new transition (on the diagram), and selecting “*Copy reference*” (figure 39). After that, a new window appears and you can directly name your transition and select the desired language. Notice that you still have to add the coil manually.



Please select the duplication mode for transition.

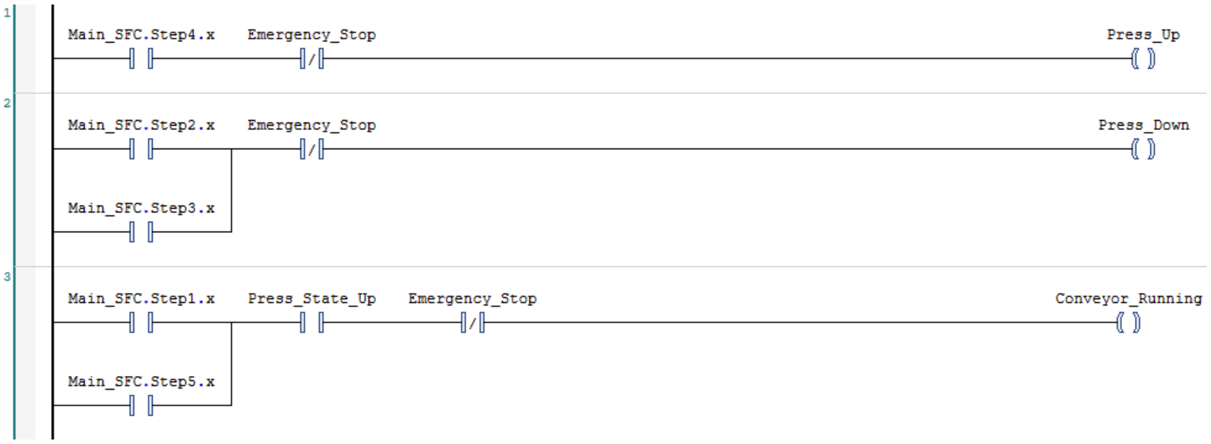
● Copy reference: A new transition element calls the same underlying transition object

○ Copy implementation: New transition object is created for a new transition element

□ Set as default. This setting can be changed in SFC editor options

Figure 42 : You should select “Copy reference”

Appendices



Appendix 1 : The Output Ladder of the example