SIEMENS

SIMATIC

STEP 7 Professional / WinCC Advanced V11 for Sample project Filling Station

Getting Started

10/2011

A5E03728923-01

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠DANGER
indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠WARNING
indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠CAUTION
with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

CAUTION
without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

NOTICE
indicates that an unintended result or situation can occur if the relevant information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠WARNING
Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E03728923-01
℗ 11/2011

Table of contents

1 Overview of the Getting Started

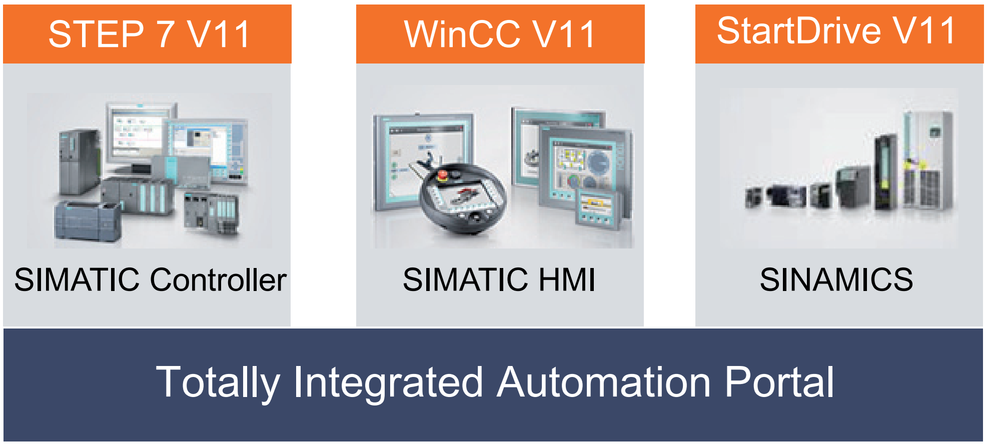
1.1 Introduction to the TIA Portal

Introduction

The Totally Integrated Automation Portal, referred to as TIA Portal in the following, offers all the functions you need for implementing your automation task assembled in a single, cross-software platform.

The TIA Portal is the first shared working environment for integrated engineering with the various SIMATIC systems made available within a single framework. The TIA Portal therefore also enables reliable, convenient cross-system collaboration for the first time.

All required software packages, from hardware configuration and programming to visualization of the process are integrated in a comprehensive engineering framework.



Advantages of working with the TIA Portal

The following features provide efficient support during the realization of your automation solution when working with the TIA Portal:

- **Integrated engineering with a uniform operating concept**

  Process automation and process visualization go "hand-in-hand".

- **Consistent, centralized data management with powerful editors and universal symbols**

  Data created once is available in all editors. Changes and corrections are automatically applied and updated within the entire project.

- **Comprehensive library concept**

  Use the ready-made instructions and pre-existing parts of the project again and again.
- **Multiple programming languages**

  Five different programming languages are available for implementing your automation task.

1.2 Views in the TIA Portal

Introduction

Two different views available for a differentiated introduction to the TIA Portal: the portal view and the project view.

Below you will find an explanation of the functions of the portal view and the project view.

Note

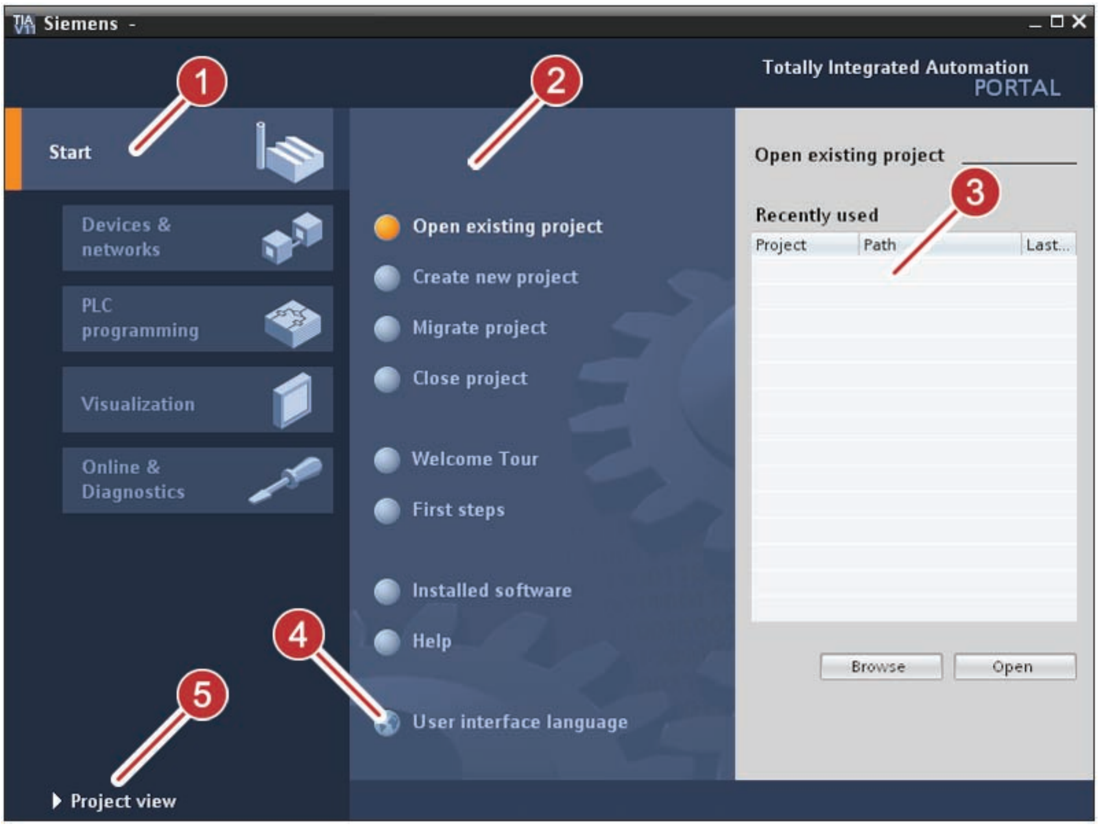
You can find additional information on this topic in the information system of the TIA Portal.

The portal view

The portal view provides an overview of all configuration steps and enables a task-based start to your automation solution.

The individual portals ("Start", "Devices & Networks", "PLC programming", "Visualization", "Online & Diagnostics", etc.) show all required steps to solve an automation task clearly structured. Here, you can quickly decide what you want to do and start the tool required for it.

The following figure shows the layout of the portal view:



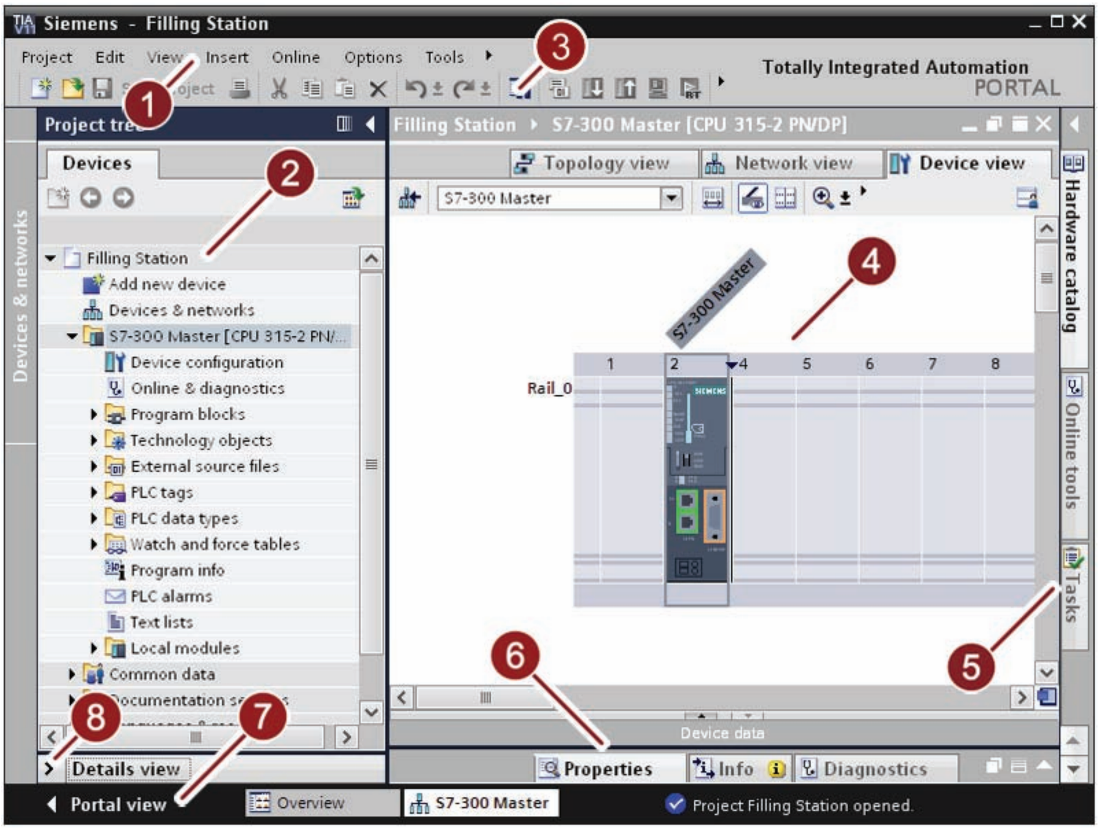
①	Portals for the various tasks: The portals provide the basic functions for the individual task areas. The portals that are provided in the portal view depend on the products that have been installed.
②	Actions for the selected portal: Here, you will find the actions available to you in the portal you have selected. You can open context-sensitive help in every portal.
③	Selection window for the selected action: The selection window is available in all portals. The content of the window adapts to your current selection.
④	Select user interface language.
⑤	Change to project view.

The project view

The project view is a hierarchically structured view of all components in a project. The project view enables quick and intuitive access to all objects in the project, the relevant work areas and editors. Using the available editors, you can create and edit all the objects required in the project.

The various work windows show you all the corresponding data for the selected objects.

The following figure shows the layout of the project view:



①	Menu bar: The menu bar contains all the commands that you require for your work.
②	Project tree: The project tree provides access to all components and project data.
③	Toolbar: The toolbar provides you with buttons for commands you will use frequently. This setup gives you faster access to these commands compared to the menus in the menu bar.
④	Work area: The objects that you can open for editing purposes are displayed in the work area.
⑤	Task cards: Task cards are available depending on the edited or selected object. You can find the task cards in a bar at the right edge of the screen. You can collapse and reopen them at any time.
⑥	Inspector window: The inspector window displays additional information on a selected object or on executed actions
⑦	Portal view: Changing to the portal view
⑧	Detailed view: The detailed view shows specific content of a selected object. This might include text lists or tags.

Note

Setting the work area in the TIA Portal

You can close the task cards, the project tree and the inspector window with one click. This increases the size of the work area. To return to the previous view, you can maximize the window again at any time.

1.3 Introduction to the Getting Started

Introduction to the Getting Started

Based on the example of this Getting started, you will see how you can implement a comprehensive automation task step-by-step using TIA Portal V11.0 Professional.

Each individual configuration step is explained in detail in the Getting Started. Illustrations are provided to make each step easy to understand and follow.

Along the way, you will easily learn how to work with the TIA Portal, because the steps you will take can also be applied to your own automation task.

Requirements

The following hardware and software is required to work with the Getting Started:

- Hardware:

  You require no additional hardware except your functional computer, because the module being used and the HMI Panel are simulated by the software for testing the project.

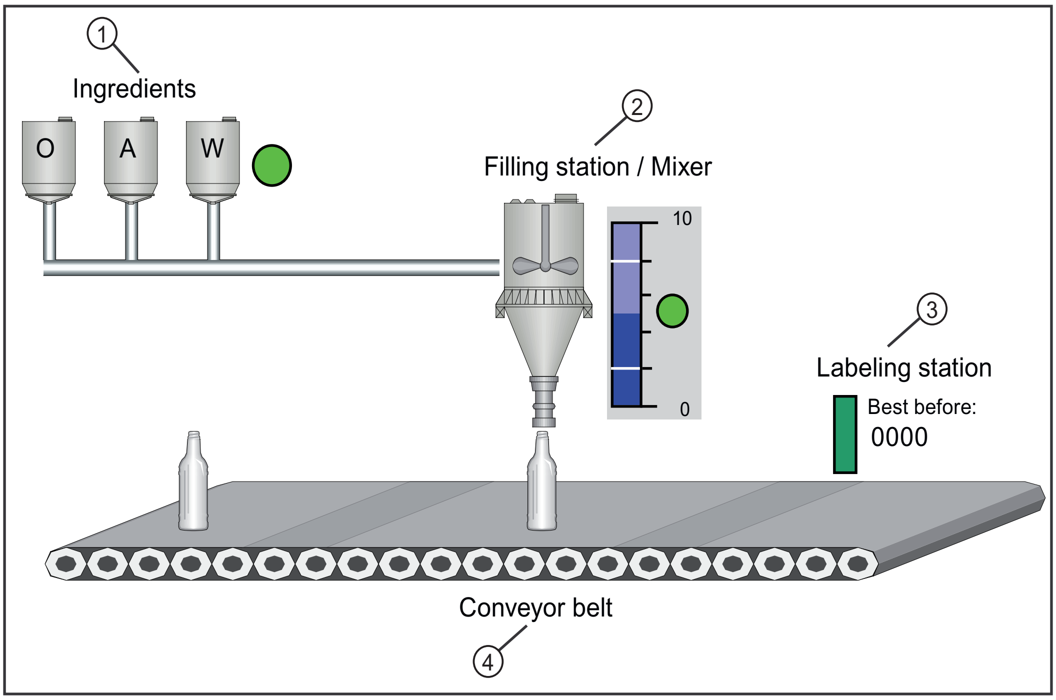
- Software:

  The following software packages must be installed and executable on your computer:

  - "STEP 7 Professional V11"
  - "WinCC Advanced V11"
  - The simulation software "S7-PLCSIM" and "WinCC Runtime Advanced Simulator"

The example project in the Getting Started

The "Filling Station" example project is implemented as an industrial filling plant for various fruit juices and fruit juice mixtures as shown below:



Components of the "Filling Station"	
①	Containers for the various ingredients: • A tank for orange juice concentrate • A tank for apple juice concentrate • A tank for water
②	Filling station with mixer for combining the respective recipe ingredients
③	Labeling station for labeling the fruit juice bottles and printing the respective expiration date
④	Conveyor belt for transportation of bottles

1.4 Organization of the Getting Started

Introduction

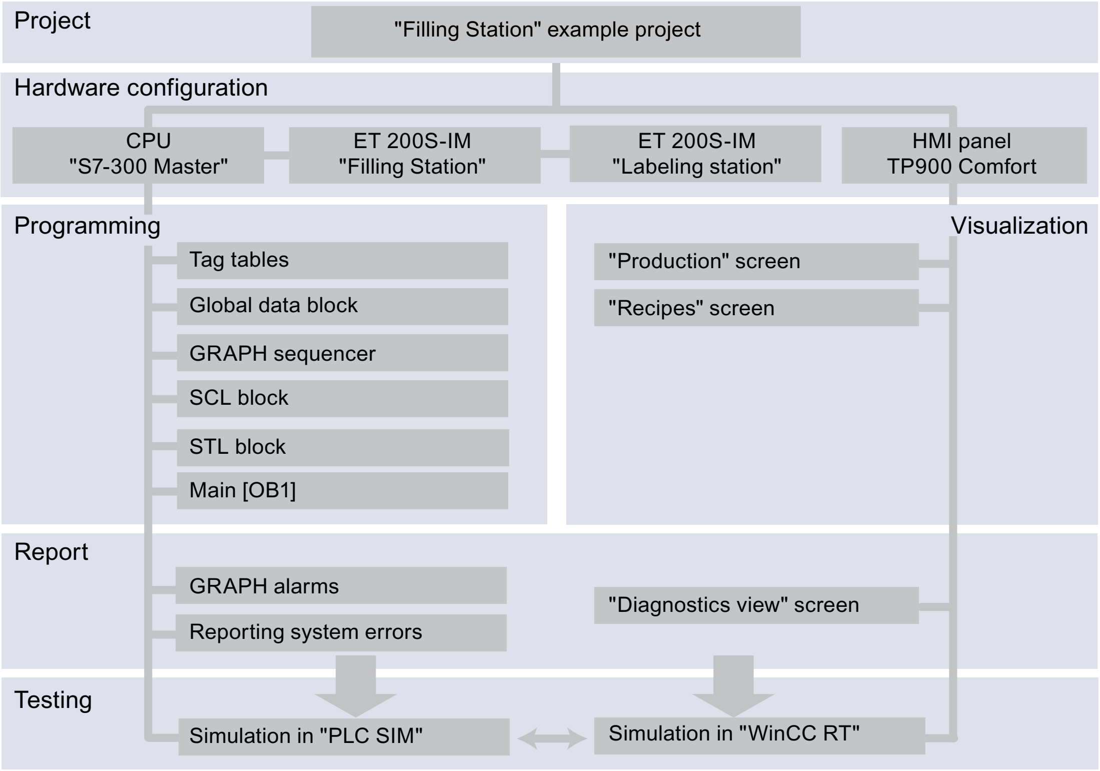
The following section provides an overview of the individual configuration steps and the objects you will create for the Filling Station example project within the TIA Portal.

Organization of the "Filling Station" project

The example project is divided into the following configuration steps:

- Create the "Filling Station" example project
- Add and configure hardware
- Program the PLC
- Visualize the process
- Configure alarms
- Test example project online

The figure below shows these configuration steps with the objects to be created:



A detailed list of the individual configuration steps can be found in the following table. You can use the links to navigate directly to the required task.

Step	Task	Implementation
1	Create "Filling Station" example project (Page 17)	• Start the TIA Portal • Create a new project
2	Add and configure hardware (Page 21)	• Add the CPU • Display the CPU in the device view • Configure the interface of the CPU • Add the power supply and signal modules • Add "Filling Station" DP slave • Pack the addresses • Add "Labeling Station" DP slave
3	Program the PLC (Page 45)	• Create PLC tag tables • Create global data block • Create sequence control with GRAPH function block • Calculate expiration date with SCL block • Control conveyor belt with STL block • Call program blocks in the Main block [OB1]
4	Visualize the process (Page 163)	• Configure HMI Comfort Panel • Create main screen "Production" • Create screen "Recipes"
5	Configure alarms (Page 249)	• Alarms in GRAPH • Report system errors
6	Test example project online (Page 263)	• Test program • Test process visualization

1.5 How do I work with the Getting Started?

Introduction

This Getting Started shows you how to use TIA Portal V11.0 Professional to implement the "Filling Station" example project step-by-step. A few explanations are provided below to help you better understand the idea of the Getting Started.

Notes on the work process

The following information is intended to facilitate working with the Getting Started.

- Linear structure

  The Getting Started has a linear structure, which means that the work process is also linear. In other words, you begin with the first chapter and work through all subsequent chapters in the specified sequence. Of course, you can interrupt the work at any time, but do not forget to save your working version. This will save your results and lets you continue your work at any time without any problems.

- Contents of the individual chapters

  The individual chapters of the Getting Started differ in size because each configuration step is dealt with in a separate chapter. There are shorter and longer chapters, depending on the respective task.

- Text and illustrations

  The introductory chapters provide you with an overview of the contents of the Getting Started. As you work with the Getting Started, each individual configuration step is explained in detail with comprehensive instructions and corresponding illustrations in the following chapters. You can find your bearings at any time using the figures in the user interface of the TIA Portal.

- Mouse symbols

  The mouse symbols placed in the figures are numbered sequentially and indicate the order of the individual operating steps. They also show whether an object is to be selected with the right or the left mouse button, with a single-click or a double-click. The display of the symbols changes for text input and for drag&drop.

- Notes

  Further instructions and tips for working with the TIA Portal are sometimes available between the individual tasks.

- Project progress

  As you work through the Getting started, a "project progress" graphic at the beginning of each chapter shows you where you are, which task is next, and the configuration steps you have successfully completed.

- Functionality

  In this Getting Started, we only show you the functions required to implement the example project. There are other numerous functions and options within the TIA Portal that are not described here.

Note

You can find additional information about the functions used in the Getting Started in the TIA Portal information system.

Create "Filling Station" example project

2

2.1 Start the TIA Portal

Introduction

The first step for working with theTIA Portal is to start the software.

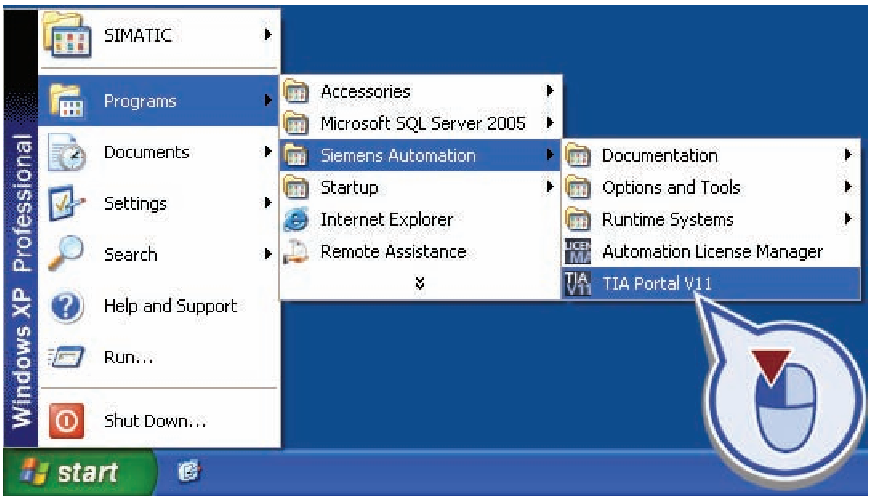
Requirement

You have installed the "TIA Portal V11.0 Professional" software.

Procedure

Follow the steps below to start the TIA Portal:

1. Click Start > Programs > Siemens Automation > TIA Portal V11.



Result

This selection starts the TIA Portal and opens the portal view (Page 8).

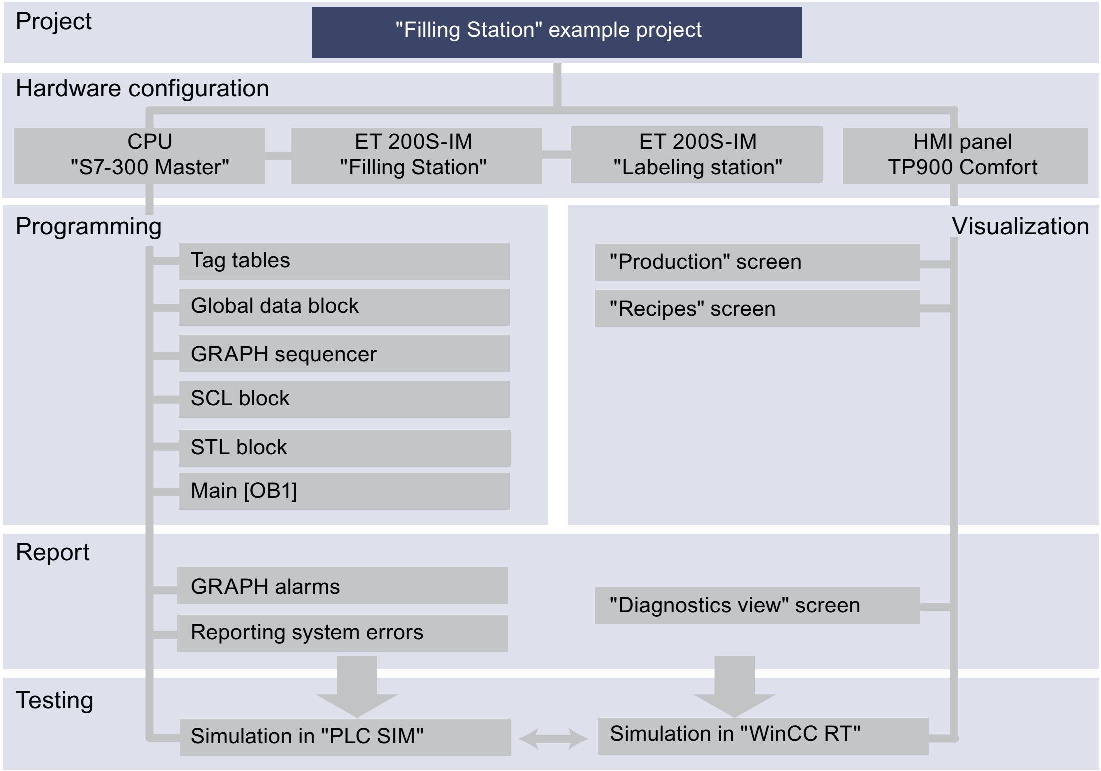
2.2 Create a new project

Introduction

In the following, we create a new project. All automation tasks are performed within a project, for example, the hardware configuration and the programming of the PLC.

Project progress in the Getting Started

The following graphic shows you which configuration step you perform next:



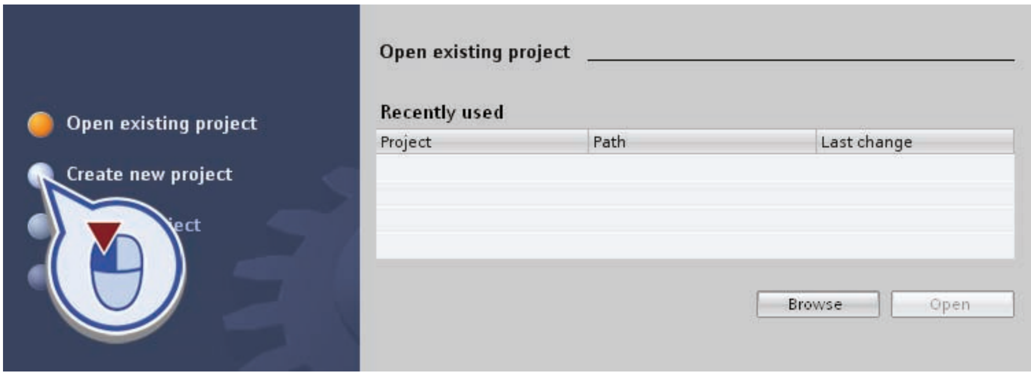
Requirement

You have started the "TIA Portal V11.0 Professional" software.

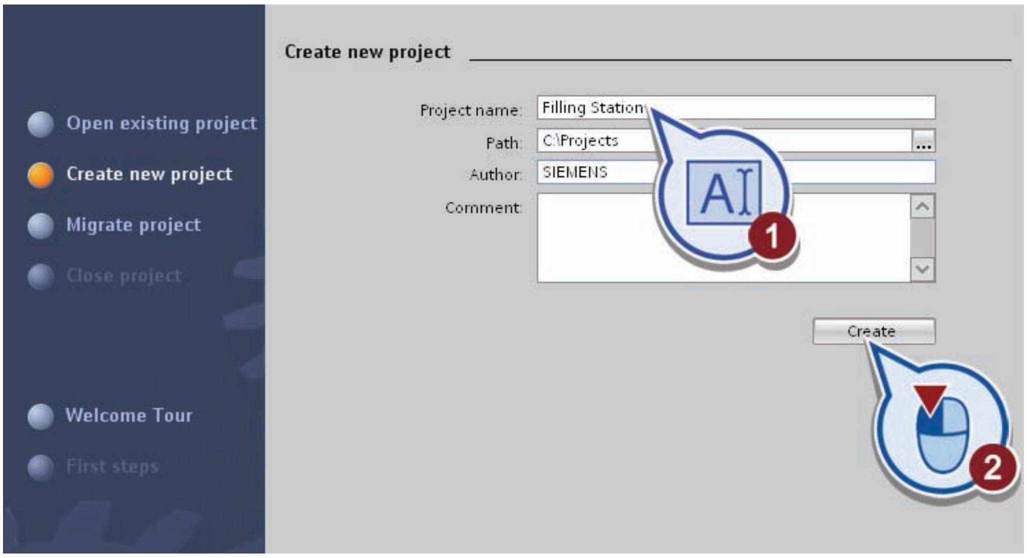
Procedure

Follow these steps to create the "Filling Station" example project:

1. Click "Create new project".



2. Enter the name "Filling Station" in the "Project name" text field and click the "Create" button.



Project name: Filling Station
Path: C:\Projects
Author: SIEMENS

Result

You have successfully created the "Filling Station" example project.

Inserting and configuring hardware 3

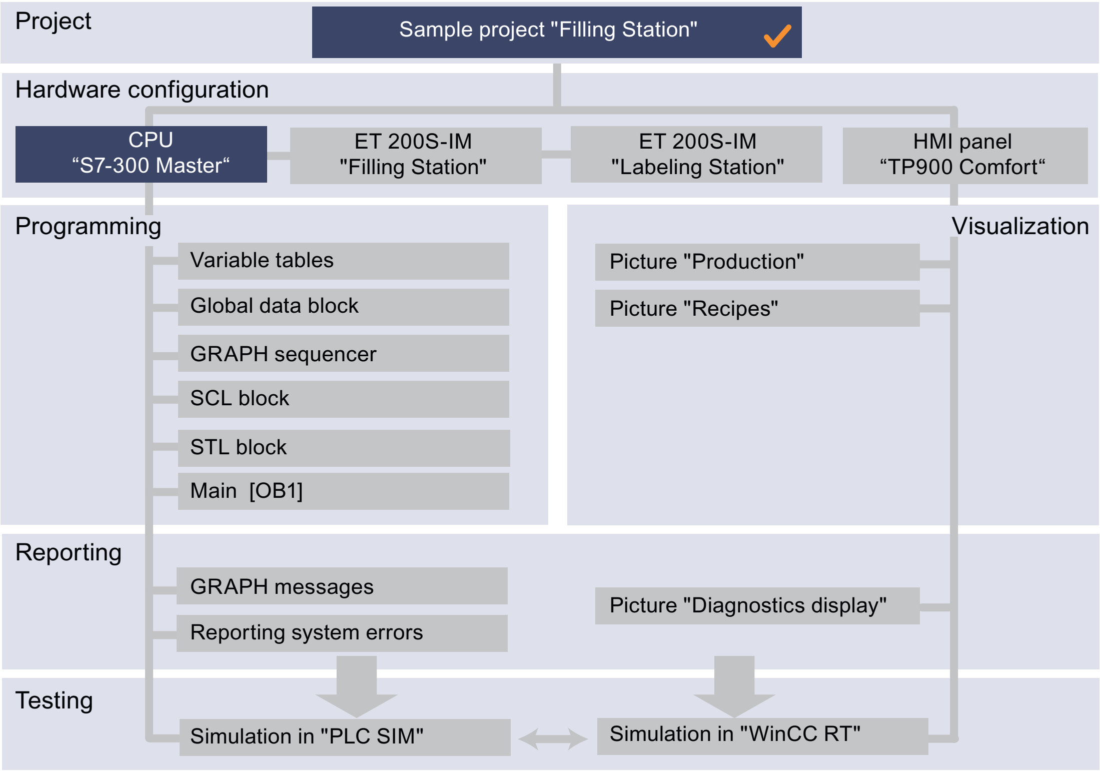
3.1 Insert the CPU

Introduction

In the following, insert the CPU "315-2 PN/DP" in the sample project "Filling Station". In the further course of the project the CPU as DP Master will control the DP-Slaves (distributed I/O).

Progress of project

The following graphic shows you which step you perform next:



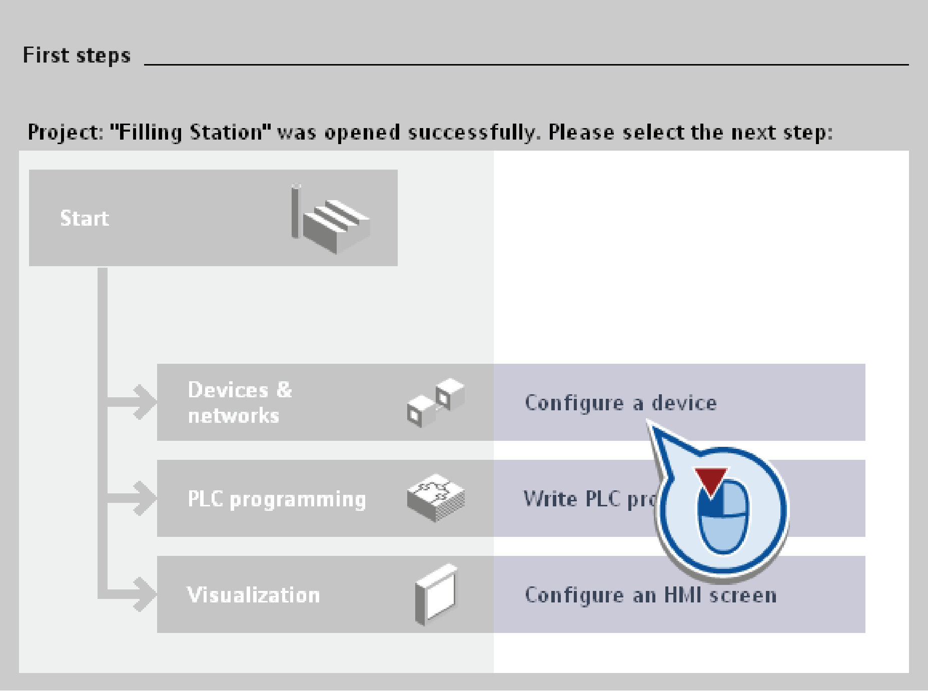
Requirement

You have created and opened the sample project "Filling Station".

Procedure

To insert the CPU, proceed as follows:

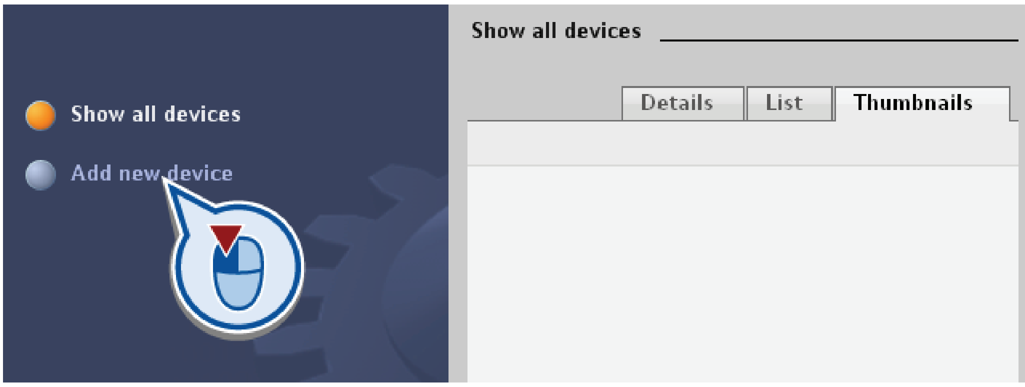
1. Click on "Configure a device".



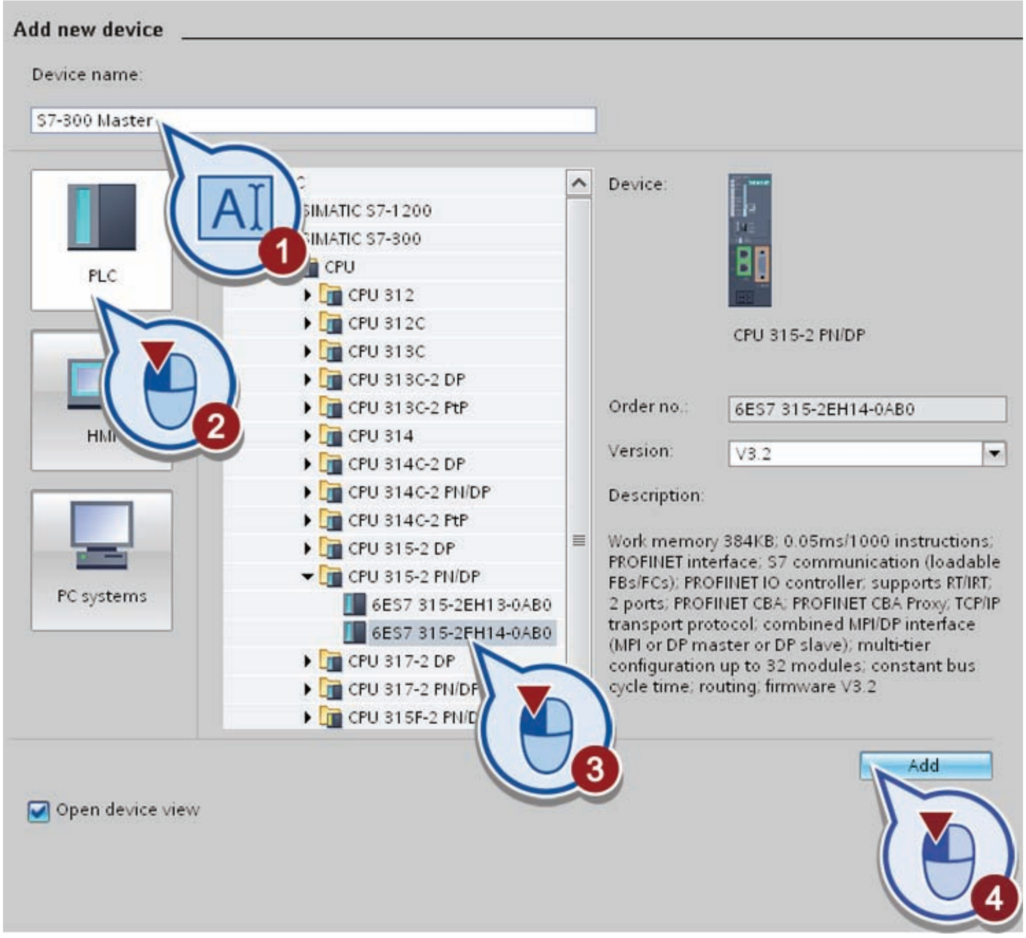
First steps

Project: "Filling Station" was opened successfully. Please select the next step:

2. Click "Add new device".



3. To create the CPU:
   - Enter in the designation "S7-300 Master" in the "Device name" text field.
   - Select the CPU "315-2 PN/DP": To do this, click "PLC" and open the folder "PLC" > "SIMATIC S7-300" > "CPU" > "CPU 315-2 PN/DP" and select the second version with the No. "6ES7 315-2EH14-0AB0".
   - Make sure that the "Open device view" option is selected. If not, select it.
   - Click "Add".



Add new device

Device name: S7-300 Master

Device: CPU 315-2 PN/DP

Order no.: 6ES7 315-2EH14-0AB0

Version: V3.2

Description:

Work memory 384KB; 0.05ms/1000 instructions; PROFINET interface; S7 communication (loadable FBs/FCs); PROFINET IO controller; supports RT/IRT; 2 ports; PROFINET CBA; PROFINET CBA Proxy; TCP/IP transport protocol; combined MPI/DP interface (MPI or DP master or DP slave); multi-tier configuration up to 32 modules; constant bus cycle time; routing; firmware V3.2

☑ Open device view

Result

You have successfully inserted the CPU "315-2 PN/DP" in the sample project "Filling Station". The TIA Portal then switches automatically from the portal view to the project view.

3.2 Displaying the CPU in the device view

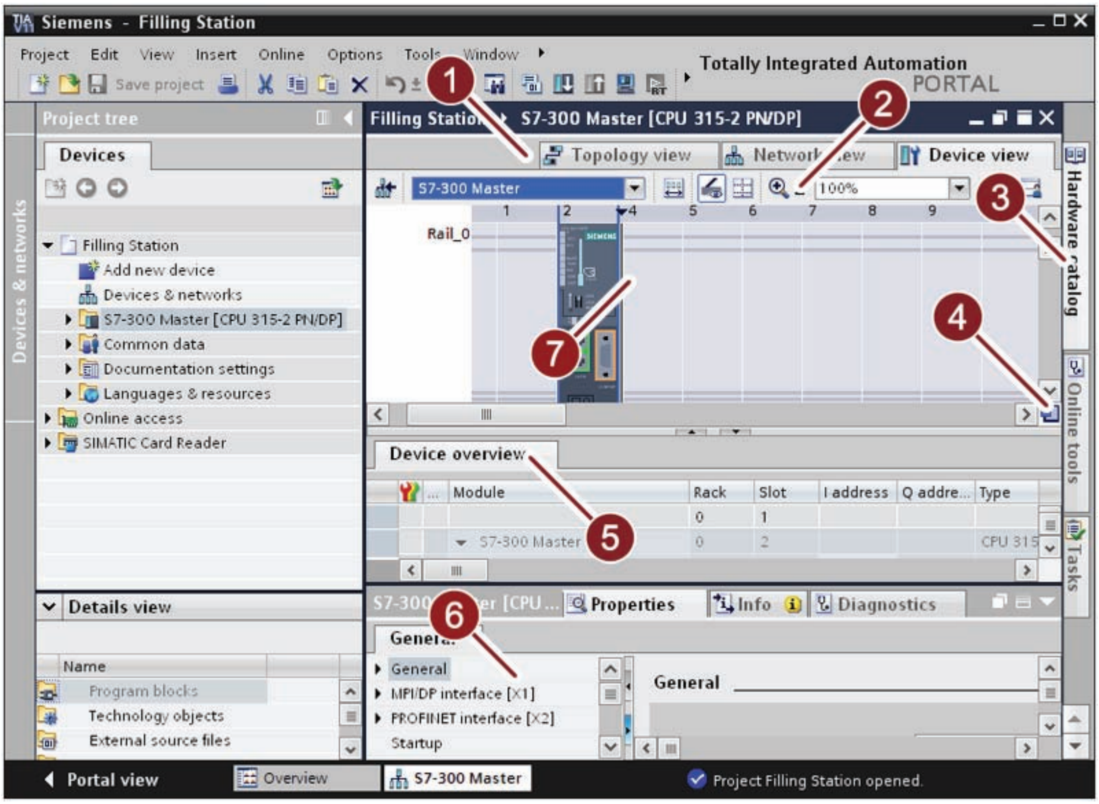
Introduction

The CPU you inserted in the sample project "Filling Station" is displayed in the device view of the hardware and network editor.

The hardware and network editor in the TIA Portal

The device view is one of the three working ranges of the hardware and network editor in which you can configure and set the parameters of devices and modules.

The following figure shows the structure of the device view:



①	Tab for toggling between topology view, network view and device view
②	Device view toolbar: You can use the toolbar to switch between the various devices and to show and hide certain information. Use the zoom function to change the representation in the graphic area.
③	"Hardware catalog" task card: The hardware catalog gives you easy access to various hardware components. Drag the devices and modules you need for your automation task from the hardware catalog to the graphic area of the device view.

④	Overview navigation: Click in the overview navigation to obtain an overview of the created objects in the graphic area. By holding down the mouse button, you can quickly navigate to the desired objects and display them in the graphic area.
⑤	Table area of the device view: The table area of the device view gives you an overview of the modules used with the most important components and technical data.
⑥	Inspector window: The inspector window shows information about the currently selected objects. You can edit the settings of the selected objects in the "Properties" tab of the inspector window.
⑦	Graphic area of the device view: The graphic area of the device view displays hardware components and if necessary the associated modules that are assigned to each other via one or more racks. For devices with racks you can drag additional hardware objects from the hardware catalog (3) into the slots of the rack and configure these objects.

Note

Setting working range in the TIA Portal

You can close the task cards, project tree and inspector window with a single click. This increases the size of the work area. To return to the previous view, you can maximize the window again at any time.

3.3 Configuring the CPU interface

Introduction

In the following section you will configure the Ethernet interface of the CPU "315-2 PN/DP". You can use this interface to network the DP slaves (distributed I/O stations), which are still to be inserted in the further course of the project, to the CPU.

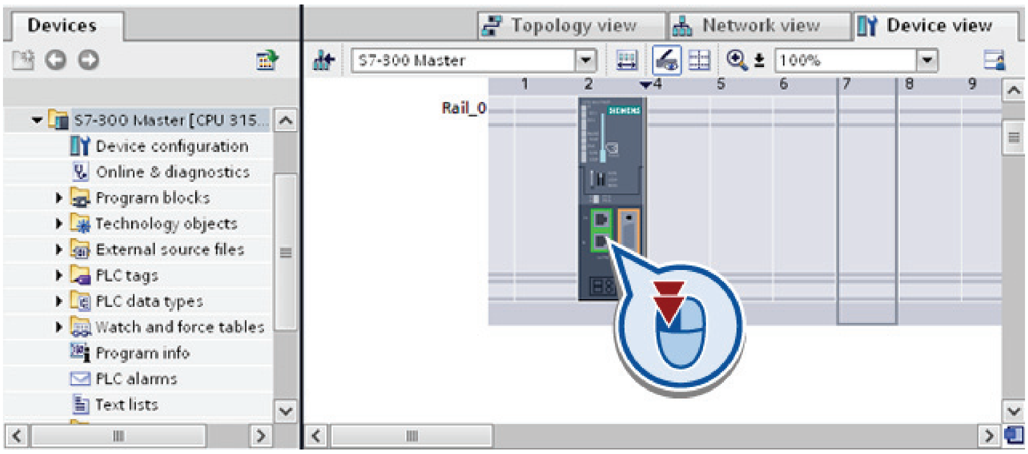
Requirement

You have the opened the CPU "S7-300 Master" in the device view of the hardware and network editor.

Procedure

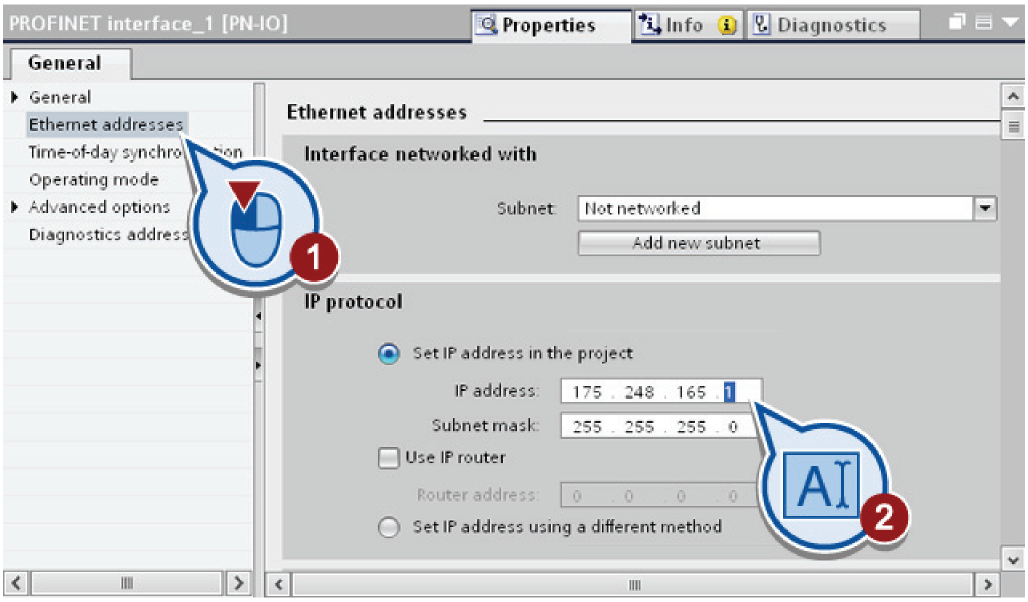
To configure the Ethernet interface of the CPU, proceed as follows:

1. Double-click the Ethernet interface of the CPU.



   The properties of the Ethernet interface are displayed in the inspector window.

2. In the "Properties" tab in the inspector window click the "Ethernet address" dialog box. Enter the following IP addresses in the "IP protocol" under "Set IP address in the project": "175.248.165.1".



Subnet: Not networked

● Set IP address in the project

IP address: 175 . 248 . 165 . 1

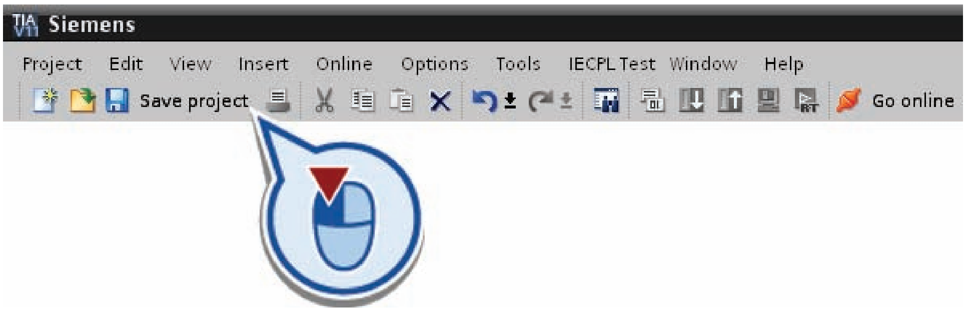
Subnet mask: 255 . 255 . 255 . 0

□ Use IP router

Router address: 0 . 0 . 0 . 0

○ Set IP address using a different method

3. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.


Siemens
Project Edit View Insert Online Options Tools IECPL Test Window Help
Save project
Go online

Result

You have successfully configured the Ethernet interface of the CPU.

3.4 Insert power supply and signal modules

Introduction

In the following section you insert the power supply "PS 307 5A" and the digital input/output module "DI8/DO8 x 24VDC / 0.5 A" in the device configuration. The power supply (PS) provides the load power supply. You can use the digital input/output module to condition the ingoing and outgoing signals in the CPU.

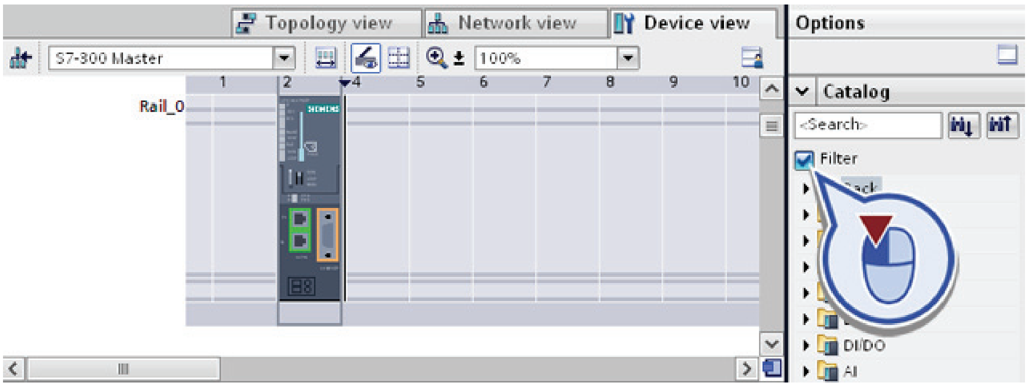
Requirement

You have the opened the CPU "S7-300 Master" in the device view of the hardware and network editor.

Procedure

To insert the power supply and the digital input/output module, proceed as follows:

1. Open the hardware catalog by clicking Task Card "Hardware catalog".
2. Check whether the "Filter" option is selected in the hardware catalog. If not, set the check mark in the check box to select it.

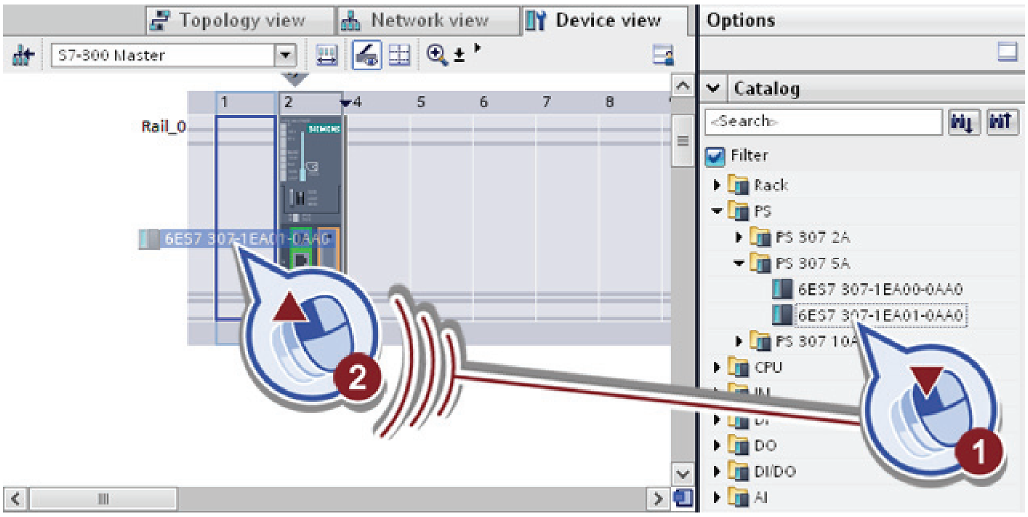


Note

You can use the "Filter" option to restrict the number of hardware components displayed.

- When the "Filter" is selected only those components which can currently be selected are displayed in the hardware catalog.
- When the "Filter" option is disabled, the entire hardware catalog is displayed.

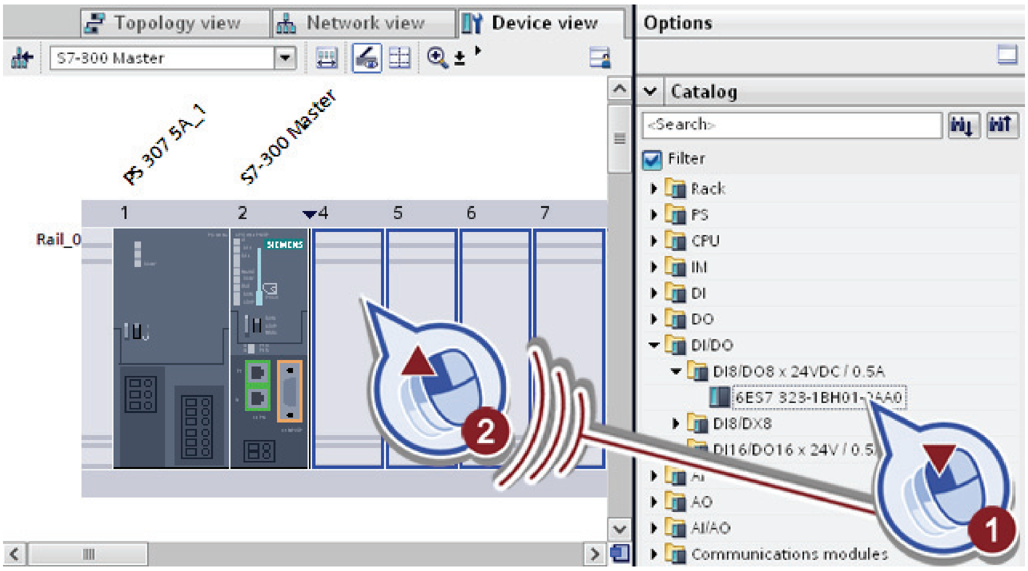
3. Drag the power supply "PS 307 5A" with the No. "6ES7 307-1EA01-0AA0" from the hardware catalog to the first slot on the mounting rail.



Note

When you click a module in the hardware catalog, the permitted slots are displayed with a blue border in the device view. For more information about the rules for slots, refer to the information system of the TIA portal.

4. Drag the digital inputs/output module "DI8/DO8 x 24VDC / 0.5 A" with the No. "6ES7 323-1BH01-0AA0" from the hardware catalog to slot 4.



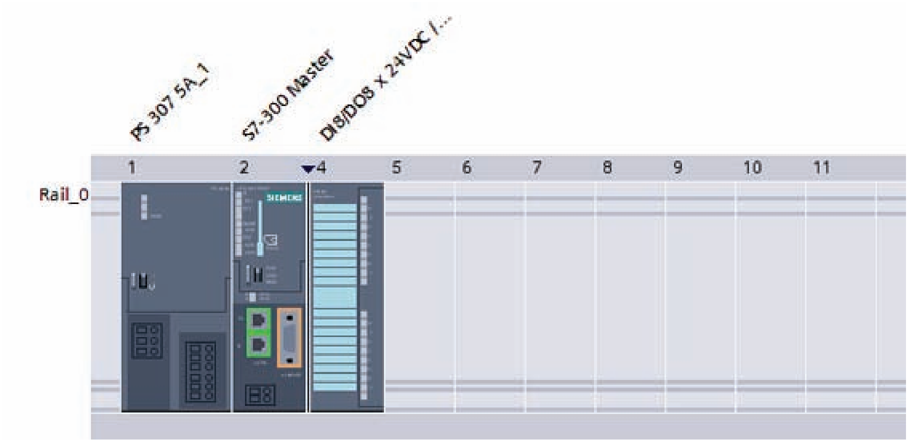
Note

Instead of navigating in the menu of the desired hardware components, you can also enter the designation or the order number of the hardware component in the search field of the hardware catalog.

5. Save the project.

Result

You have successfully inserted the power supply "PS 307 5A" and the digital input/output module "DI8/DO8 x 24VDC / 0.5A" in the sample project "Filling Station". The automatically preset input and output address areas of the module can be adjusted in the device view.



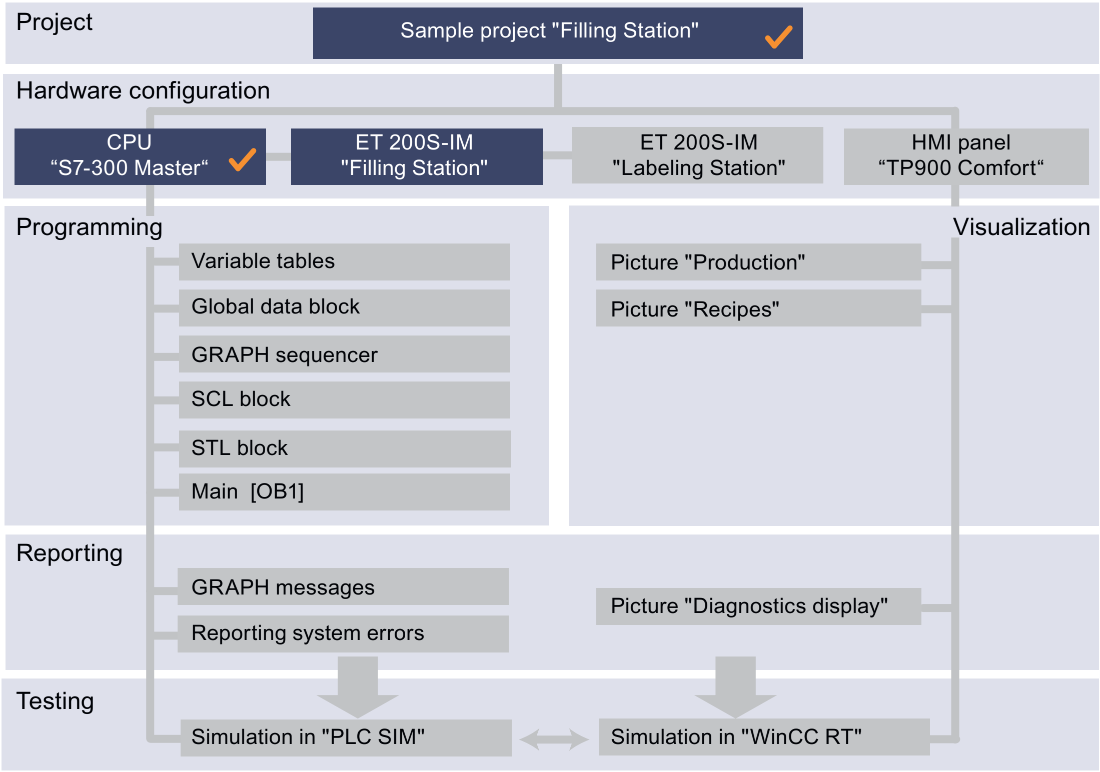
3.5 Insert DP slave "Filling Station"

Introduction

In the following section you will insert the distributed I/O station (DP slave) "Filling Station" with an appropriate power module and digital input/output modules. DP slaves can be used to process locally all input/output signals required for controlling the filling process.

Progress of project

The following graphic shows you which step you perform next:



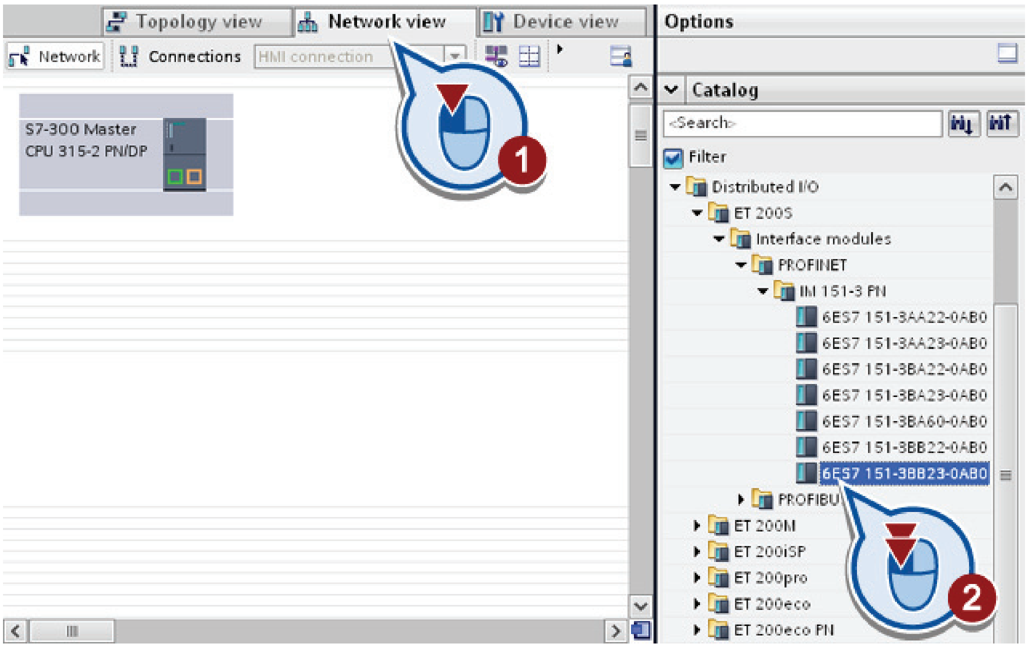
Requirement

You have the opened the CPU "S7-300 Master" in the network view of the hardware and network editor.

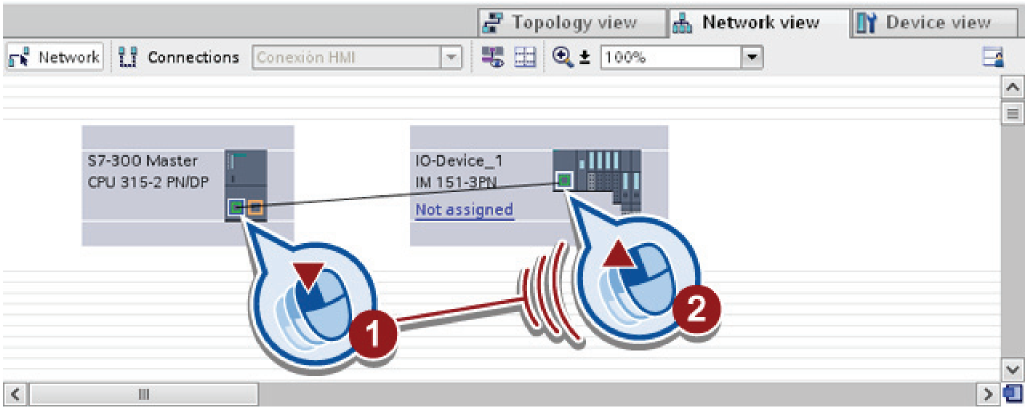
Procedure

To insert the DP slave "Filling Station", proceed as follows:

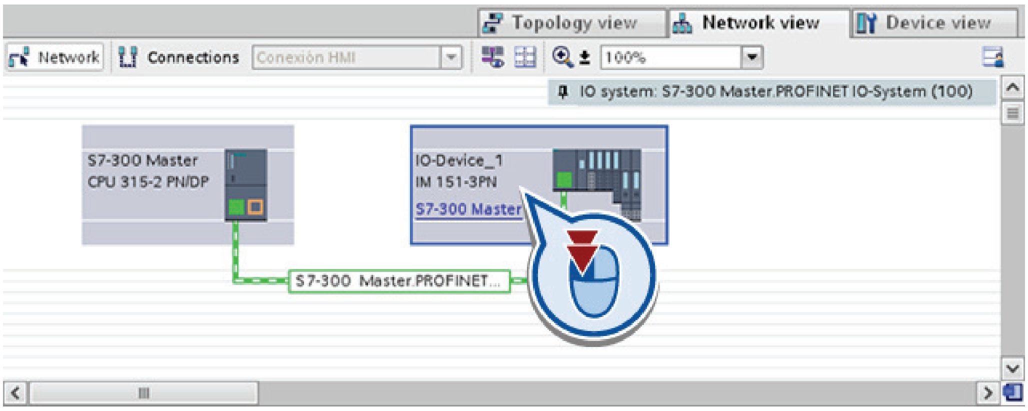
1. Drag the DP Slave "IM 151-3 PN" with the No. "6ES7 151-3BB23-0AB0" from the hardware catalog to the editor area.



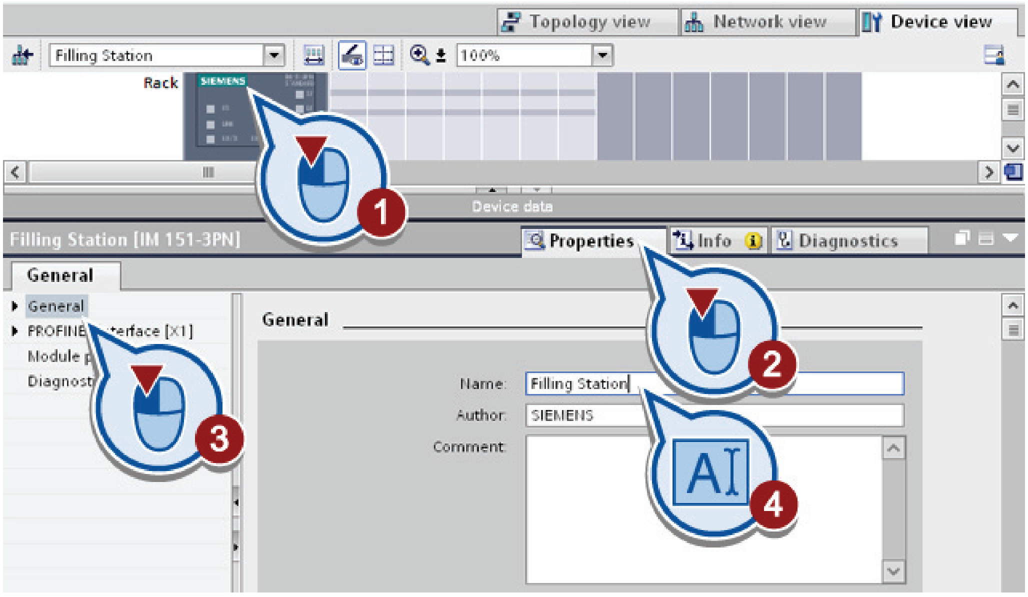
2. Create a PROFINET connection between the DP Slave "IM 151-3 PN" and the CPU "S7-300 Master".



3. Double-click the DP slave to open this in the device view. The name displayed in the network view corresponds to the name of the device. This can be modified as required in the device view of the module.

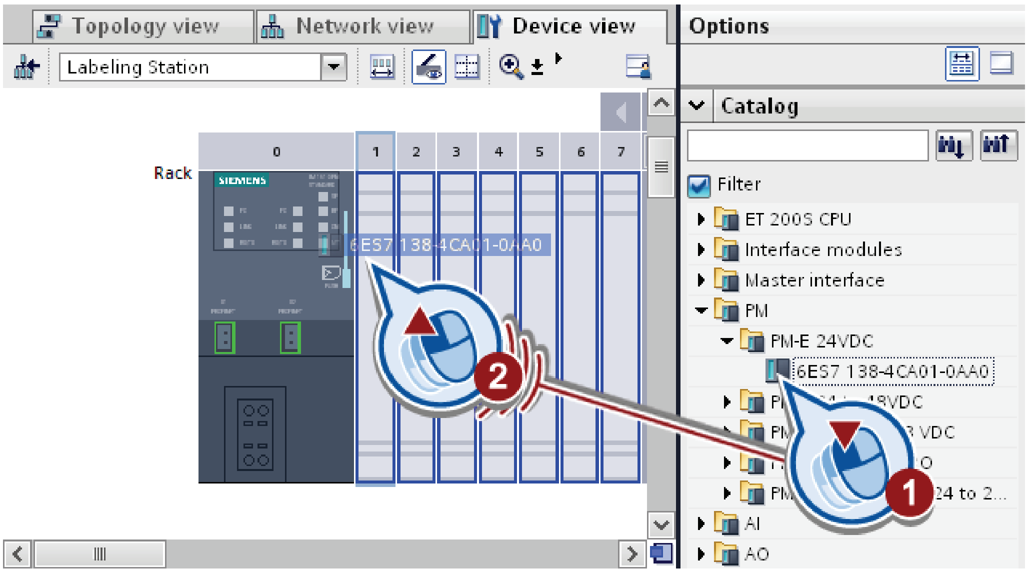


4. Select the DP Slave and modify the name of the module to "Filling Station" in the "General" dialog box.

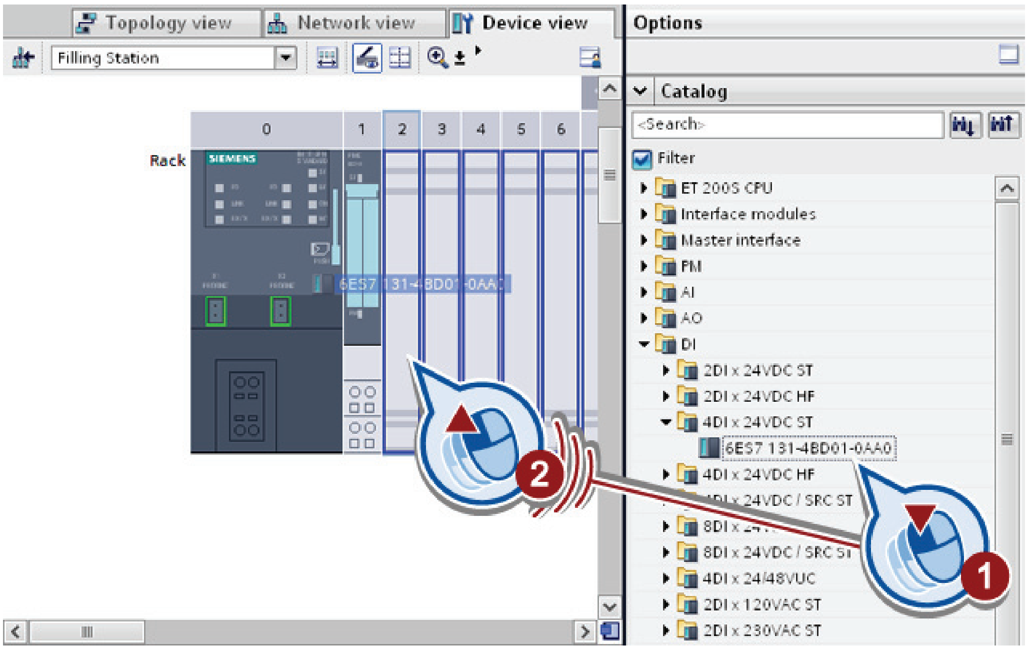


Name: Filling Station
Author: SIEMENS
Comment:

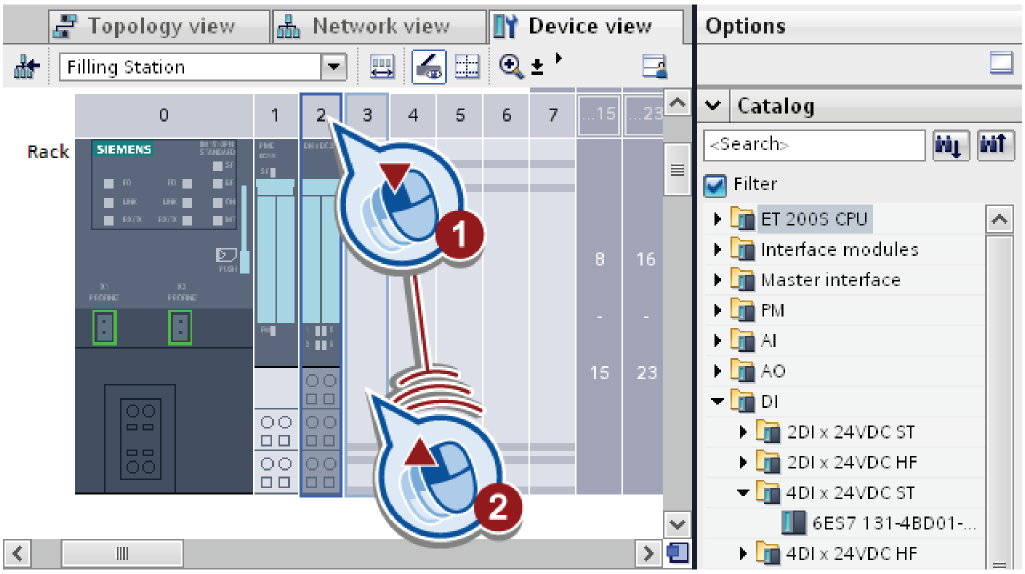
5. Select the power module "PM-E 24 V DC" from the hardware catalog. Drag the module onto slot 1.



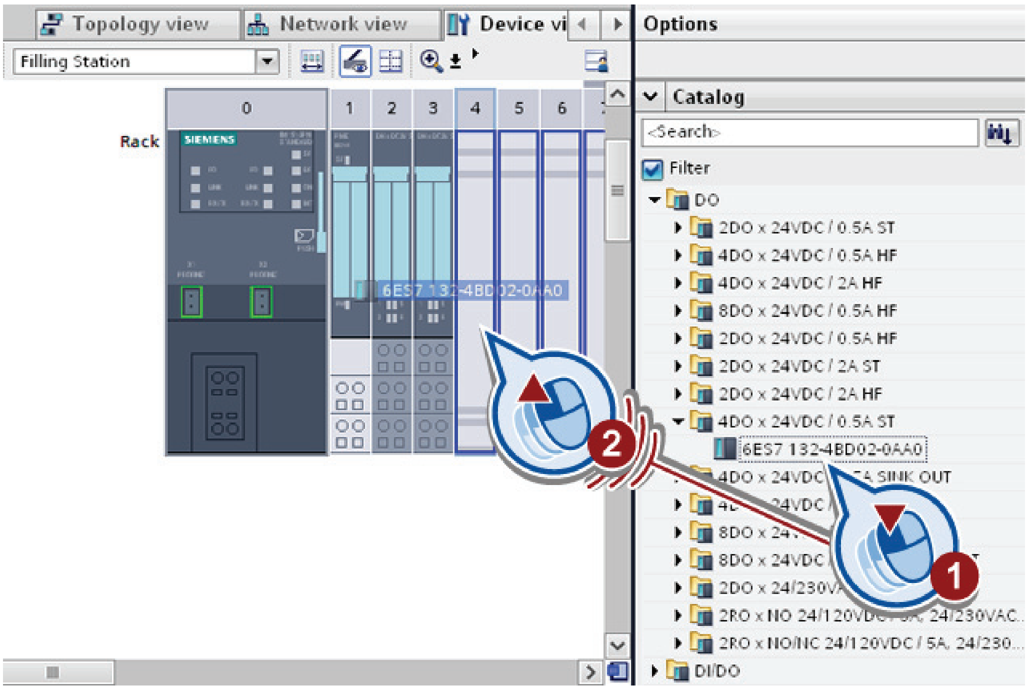
6. Select the digital input module "4 DI x 24 V DC ST" with the No. "6ES7 131-4BD01-0AA0" and drag the module onto slot 2.



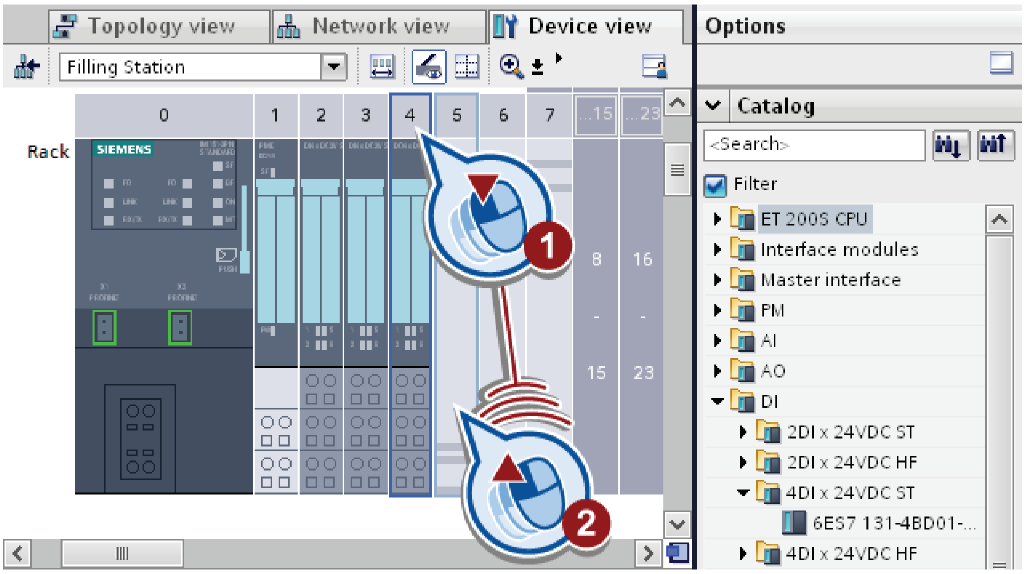
7. The digital input module is required twice for the DP slave. To copy the module, hold down the <CTRL> key and drag it from slot 2 on the empty slot 3.



8. Select the digital output module "4 DO x 24 V DC / 0.5A ST" with the No. "6ES7 132-4BD02-0AA0" and drag the module onto slot 4.



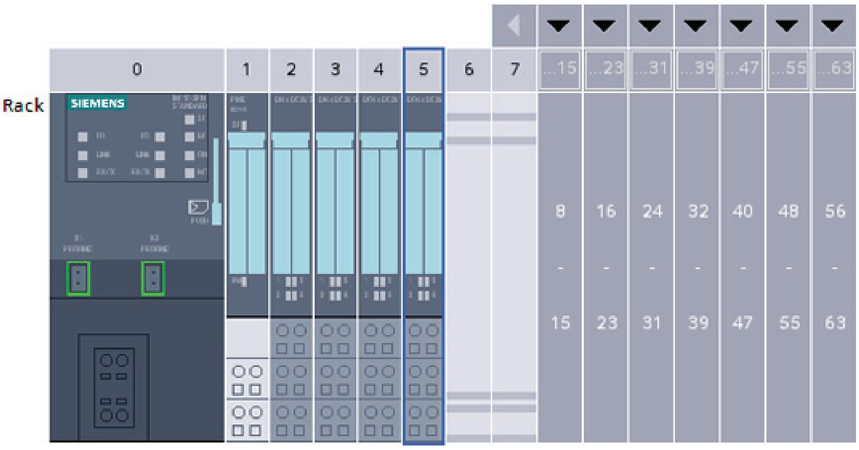
9. The digital output module is required twice for the DP slave. To copy the module, hold down the <CTRL> key and drag it from slot 4 on the empty slot 5.



10. Save the project.

Result

You have successfully created the DP slave "Filling Station" with a power module and digital input/output modules.



3.6 Packing addresses

Introduction

Two digital input modules "4 DI x 24 V DC ST" are inserted on slot 2 and 3 of the DP slave "Filling Station". Each of the two modules has 4 digital inputs. Therefore, they each require an address area of 4 bits. However, each slot is automatically assigned an address area of a full byte, as other modules have up to 8 inputs or outputs. This means that only 4 of the reserved 8 bits are required for the digital input module "4 DI x 24 V DC ST" used.

In the following section you will pack the input addresses of the two modules to reduce the assigned address range of 2 bytes in total, to 1 byte. The "Pack addresses" function also causes the 4 bit long address areas to be grouped into one byte. The following table shows how the application of the "Pack addresses" function affects the address areas of the module:

Module	Pre-assigned address area	After "Pack addresses"
4 DI x 24 V DC ST On slot 2	4 bits from byte 1 Address area: I1.0 to I1.3	4 bits from byte 1 Address area: I1.0 to I1.3
4 DI x 24 V DC ST On slot 3	4 bits from byte 2 Address area: I2.0 to I2.3	4 bits from byte 1 Address area: I1.4 to I1.7

Note

For more information on the "Pack addresses" function, refer to the information system of the TIA portal.

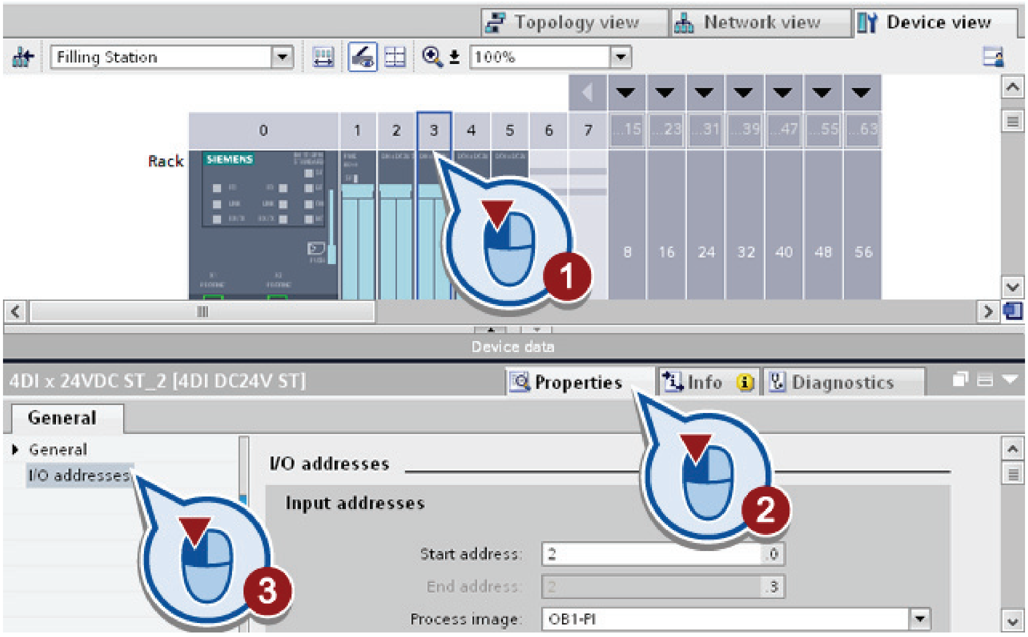
Requirement

You have opened the DP slave "Filling Station" in the device view.

Procedure

To pack the addresses of the modules, proceed as follows:

1. Select the module in slot 3 and open the properties of the I/O addresses in the inspector window.



I/O addresses

Input addresses

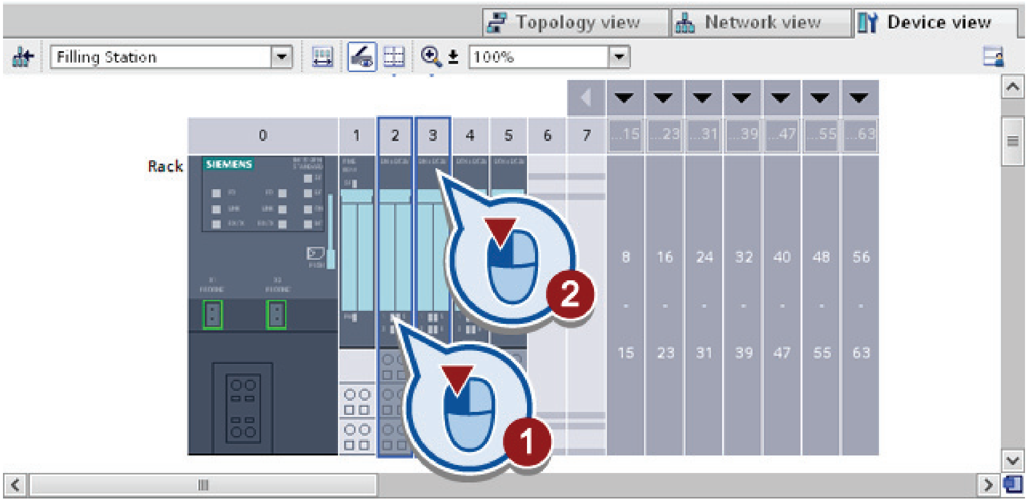
Start address: 2 .0

End address: 2 .3

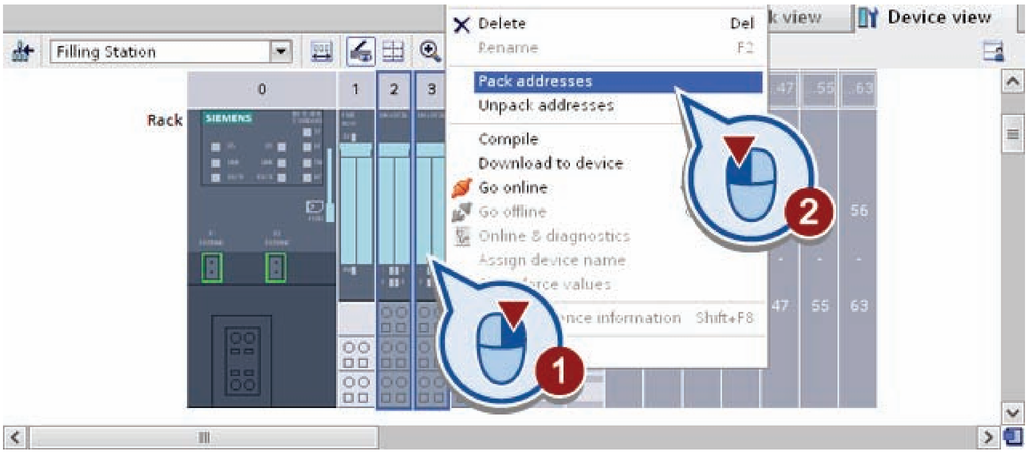
Process image: OB1-PI

As you can see, the address range begins with the start address I2.0 and ends with I2.3. Therefore, the module requires 4 bits within byte 2.

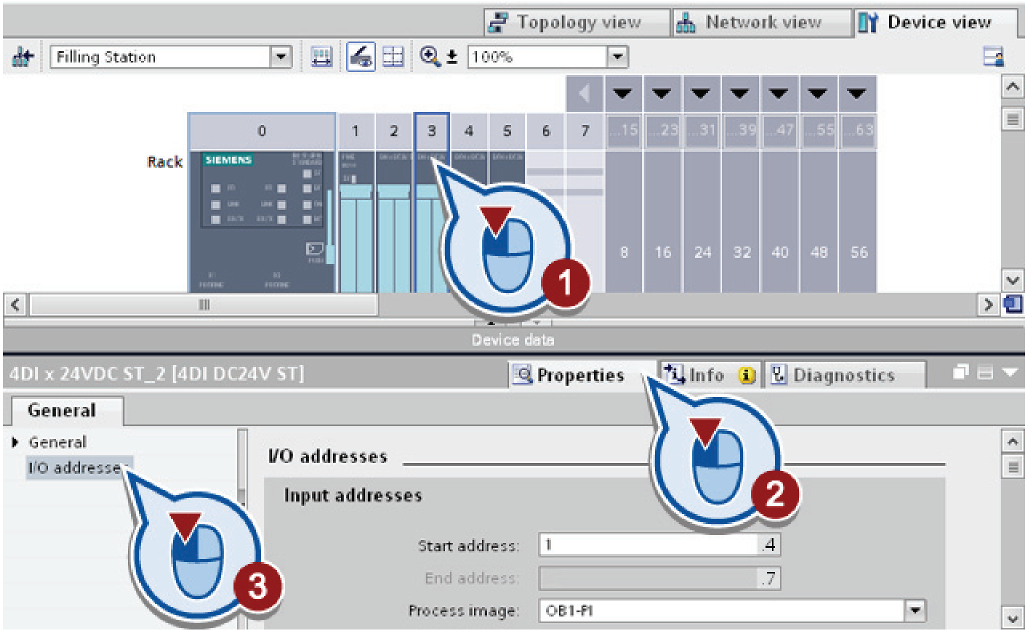
2. Select the two modules on slot 2 and 3 by clicking these while simultaneously pressing the <Shift> key.



3. Right click a module and select "Pack addresses" from the shortcut menu.



4. Click the module on slot 3 and display the properties of the I/O addresses.



4DI x 24VDC ST_2 [4DI DC24V ST]

General
- General
- I/O addresses

I/O addresses

Input addresses

Start address: 1 .4

End address: .7

Process image: OB1-PI

5. Save the project.

Result

The "I/O addresses" dialog box displays the packed input addresses. After the "Pack addresses" function has been executed, the digital input module is assigned the address area I1.4 to I1.7 on slot 3. The "Pack addresses" function has reduced by half the address area occupied by the module.

Note

Packing outputs

If required, you can you also pack outputs. Use the procedure described above for the configured output modules.

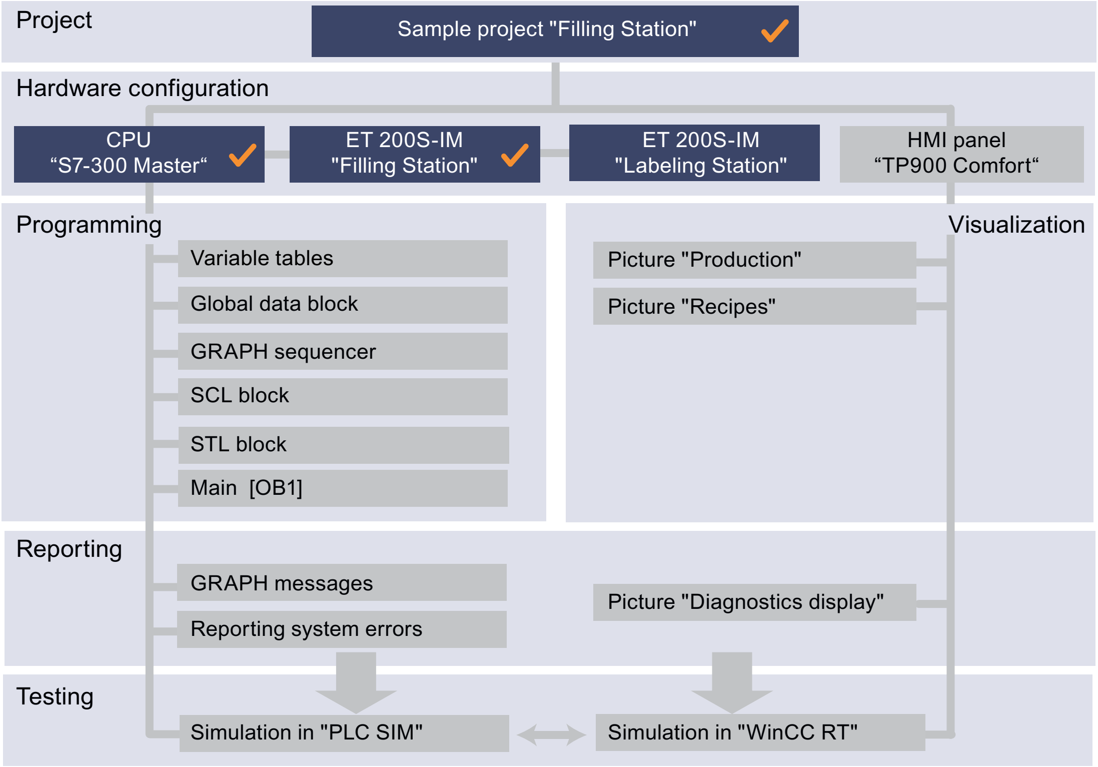
3.7 Insert DP slave "Filling Station"

Introduction

In the following section you will insert the second DP slave "Labeling Station" by copying the DP Slave "Filling Station ". You can use the second DP slave to locally process all input/output signals required for controlling the labeling process. Then network the DP slave "Labeling Station" to the CPU "S7-300 Master".

Progress of project

The following graphic shows you which step you perform next:



Requirement

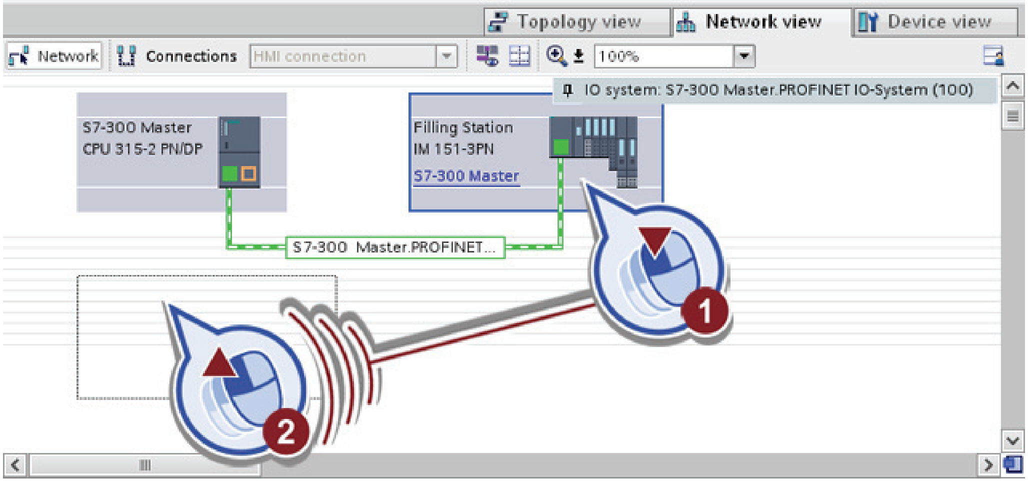
You have opened the network view of the hardware and network editor.

Procedure

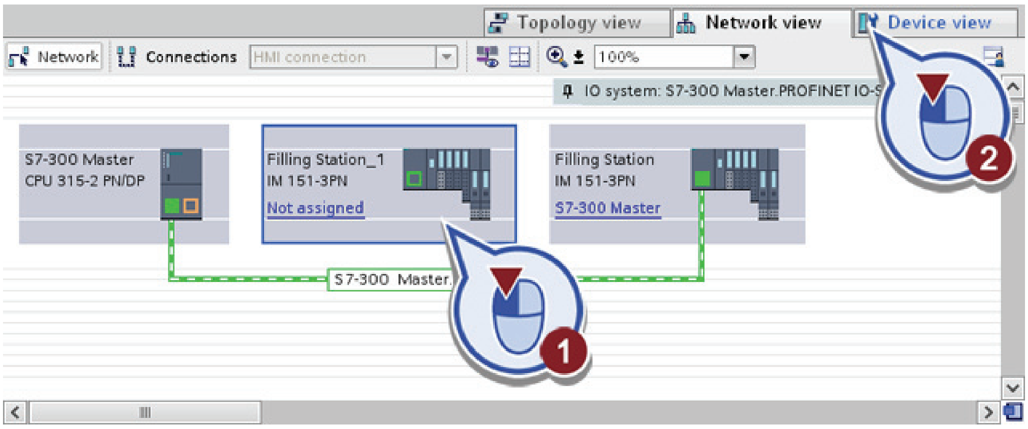
The same interface module "IM 151-3 PN" with the same configuration as for the DP slave "Filling Station" is used for the DP-Slave "Labeling Station". For this reason, you can copy the already configured DP Slave "Filling Station".

To copy the DP slave, follow these steps:

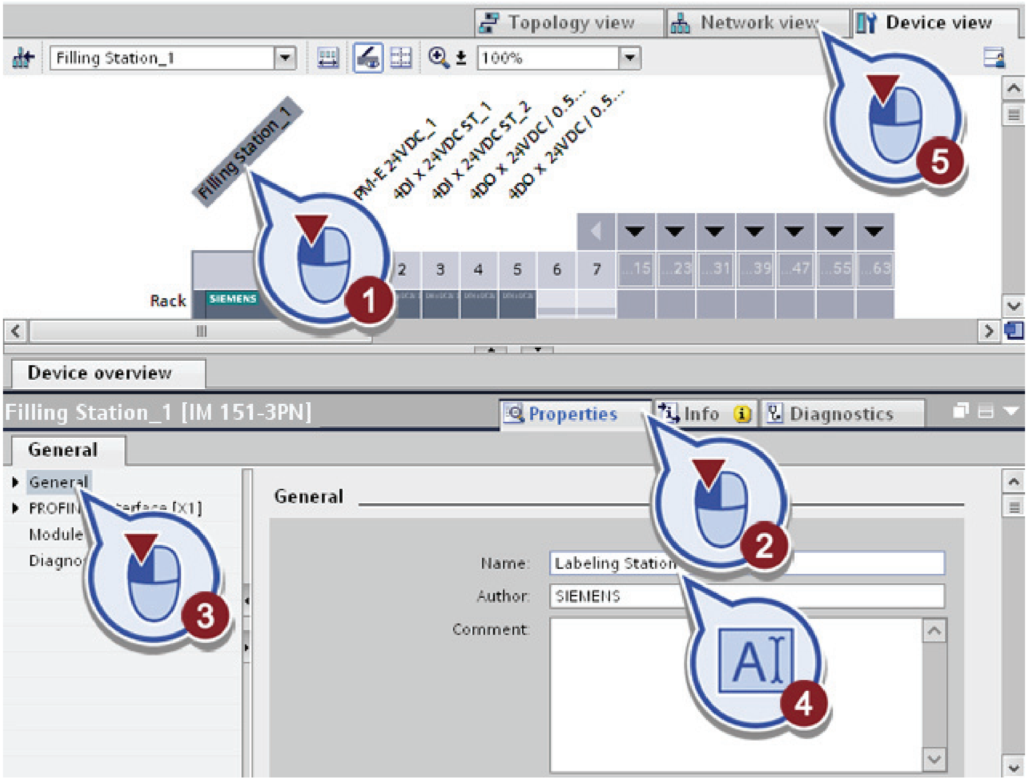
1. Select the DP Slave "Filling Station" and copy this by moving it while holding down the <CTRL> key.



2. Select the copied DP slave "Filling Station_1" and switch to the device view.



3. Open the properties of the IM module in the inspector window and rename the module to "Labeling Station". Then return to the network view.

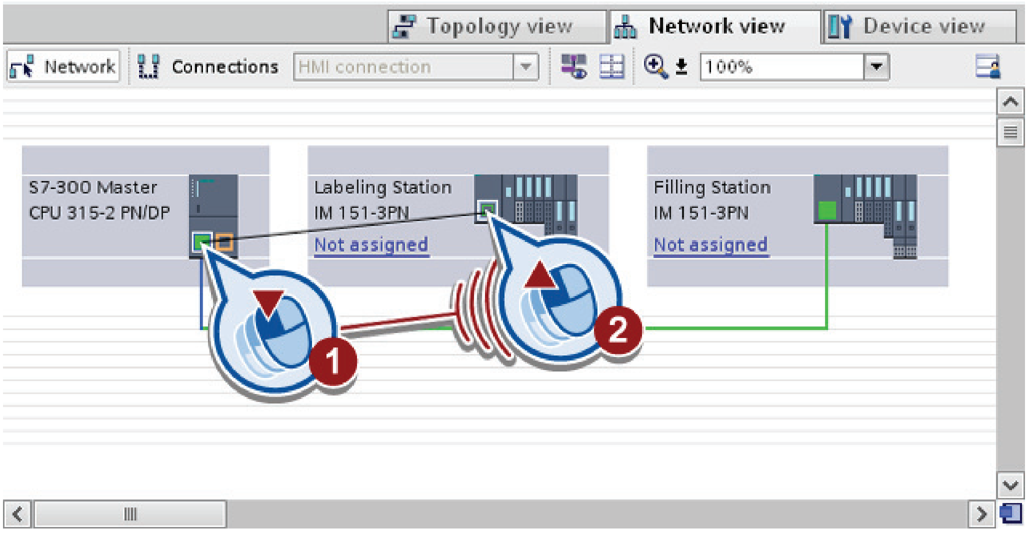


Name: Labeling Station

Author: SIEMENS

Comment:

4. In the network view link the DP slave "Labeling Station" to the existing PROFINET connection.

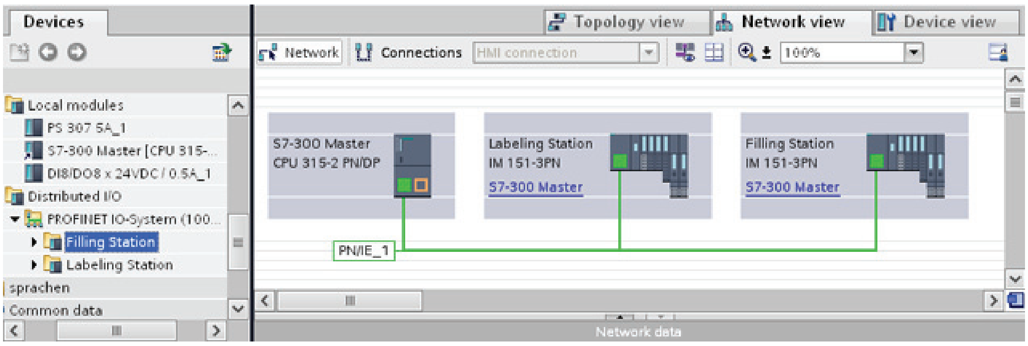


5. Save the project.

Result

You have successfully created the second DP Slave "Labeling Station". Apart from the designation, both DP slaves have the same configuration from the copying process.

The assignment of the DP slaves below the CPU "S7-300 Master" is shown in the Network view. In the project tree the DP slaves appear below CPU "S7-300 Master" in the "Distributed I/O" folder.



Programming the PLC

4

4.1 Creating PLC tag tables

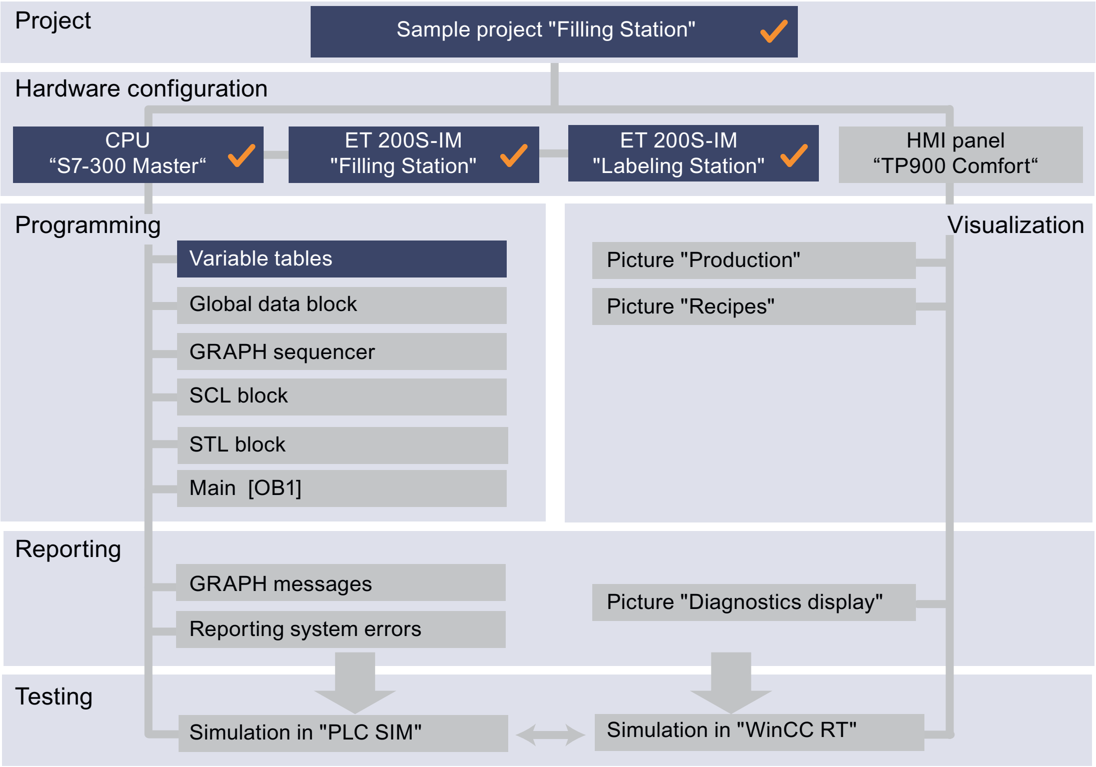
Introduction

In the following section you will create a new PLC tag table. In addition to the default tag table you can create several user-defined PLC tag tables for each CPU in the TIA portal.

For the project "Filling Station", you create four additional PLC tag tables. You can use these tag tables to organize the defined PLC tags clearly and by project component and to access these from each program editor.

Progress of project

The following graphic shows you which step you perform next:



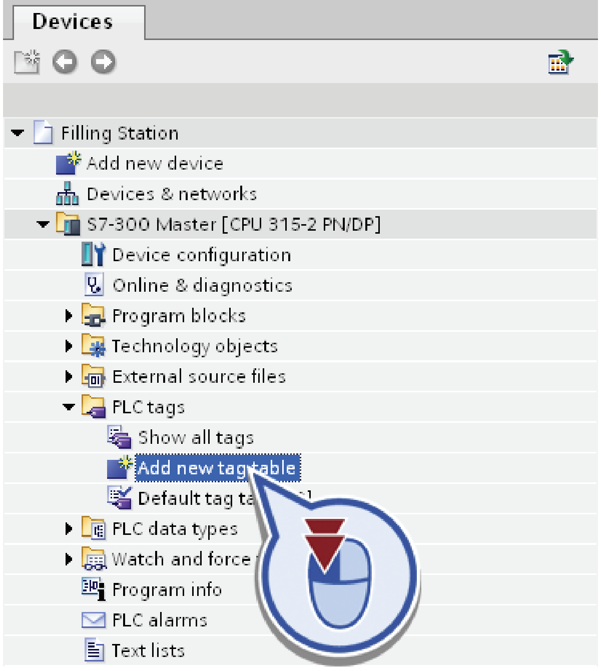
Requirement

You have configured the hardware.

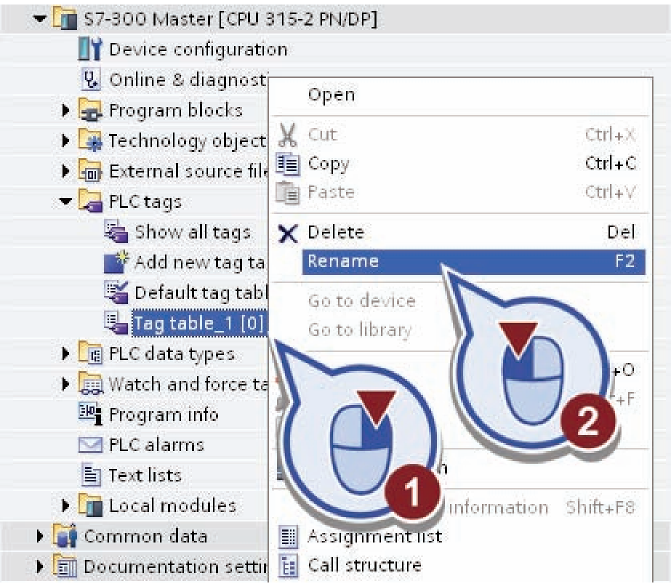
Procedure

To create the four new tag tables, proceed as follows:

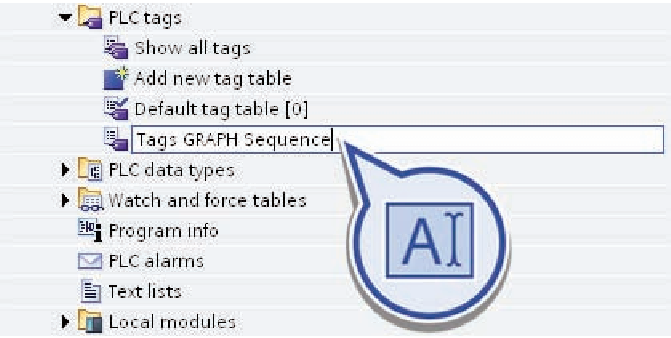
1. In the project tree, open the "PLC tags" folder below the CPU "S7-300 Master".
2. Double-click the entry "Add new tag table".



3. Right-click on the newly created "Tag table_1" and select "Rename" from the shortcut menu.



4. Assign "Tags GRAPH Sequence" as new name.

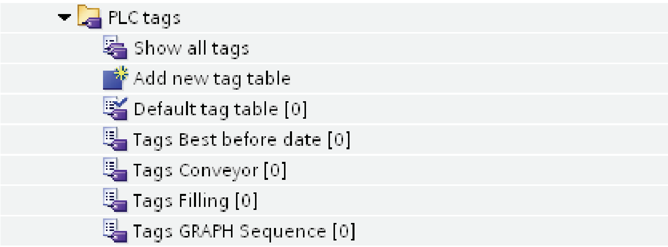


5. Repeat steps 2 to 4 to create 3 additional tag tables. Assign the following names:
   - "Tags Filling"
   - "Tags Conveyor"
   - "Tags Best before date"
6. Save the project.

Result

You have successfully created four PLC tag tables. The tags which are still to be defined in the course of the project will be created in these PLC tag tables. The default tag table remains available.

The number of tags contained is displayed in the square bracket behind the designation of a PLC tag table. You can us the "Show all tags" function to show all tags in a window and to edit these centrally.



Note

PLC tag tables

It makes no difference in which editor you modify the tag properties within a project. All changes are automatically applied at all corresponding points of use.

You can also group user-defined PLC tag tables of a CPU by storing them in a folder.

4.2 Creating a global data block

Introduction

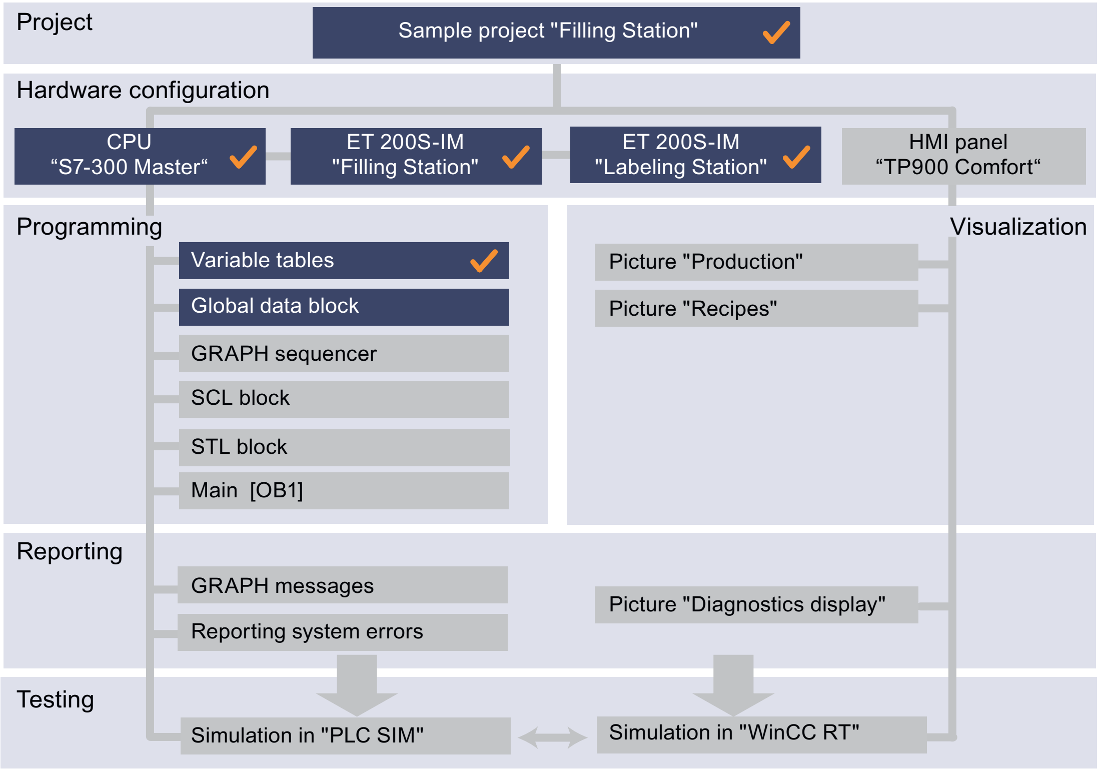
In the following section you will create a global data block. This data block provides you with the option of managing all program data of the sample project "Filling Station" at a central storage location.

Definition: Data block

Data blocks are used to store program data. Each block, regardless of whether it is a function block, a function or an organization block, can access a global data block in read or write mode. The program data remain in the global data block until they are overwritten.

Progress of project

The following graphic shows you which step you perform next:



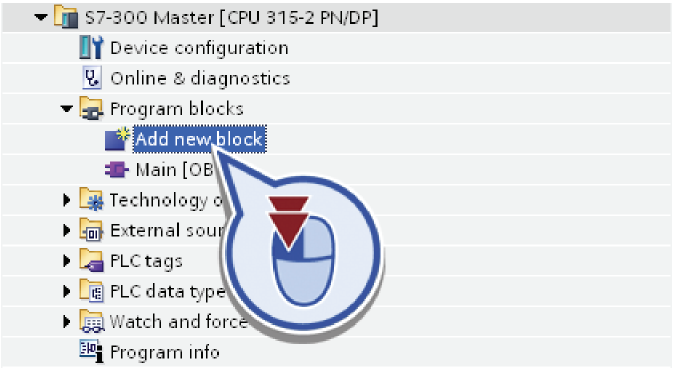
Requirement

You have configured the hardware.

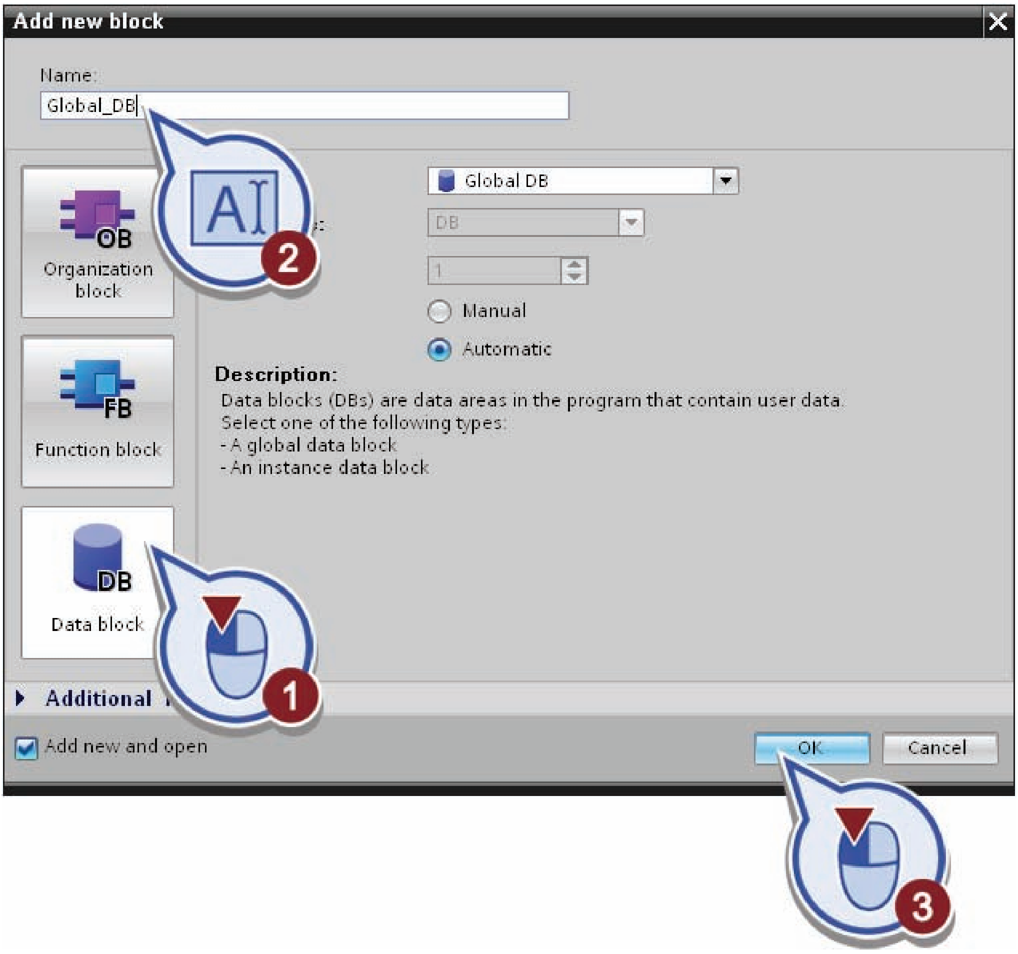
Procedure

To create a global data block, follow these steps:

1. Open the "Program blocks" folder.
2. Double-click "Add new block".



3. To add a new data block:
   - Click "Data block".
   - Assign the block name "Global_DB".
   - Select "Global_DB" as type.
   - Click "OK".



Add new block

Name: Global_DB

Global DB

DB

1

○ Manual

● Automatic

Description:
Data blocks (DBs) are data areas in the program that contain user data.
Select one of the following types:
- A global data block
- An instance data block

☑ Add new and open

4. Save the project.

Result

You have successfully created the global data block "Global_DB", in which you will later manage the recipe data for the sample project.

4.3 Using GRAPH function block to create sequence control

4.3.1 Introduction to GRAPH

Introduction

GRAPH is a graphical programming language for creating sequential control systems with the help of sequencers.

Sequential control systems can be programmed quickly and easily. The process is hereby broken down into individual steps with a clear scope of functions. The actions to be executed are defined in the individual steps. Transitions form the links between the steps. These contain conditions for switching to the next step.

Overview of sample project "Filling Station"

In the sample project "Filling Station", create a GRAPH function block (GRAPH FB) in which you program the complete process from the mixing of the beverages to the labeling of the bottles.

The following program blocks are required in addition to GRAPH FB:

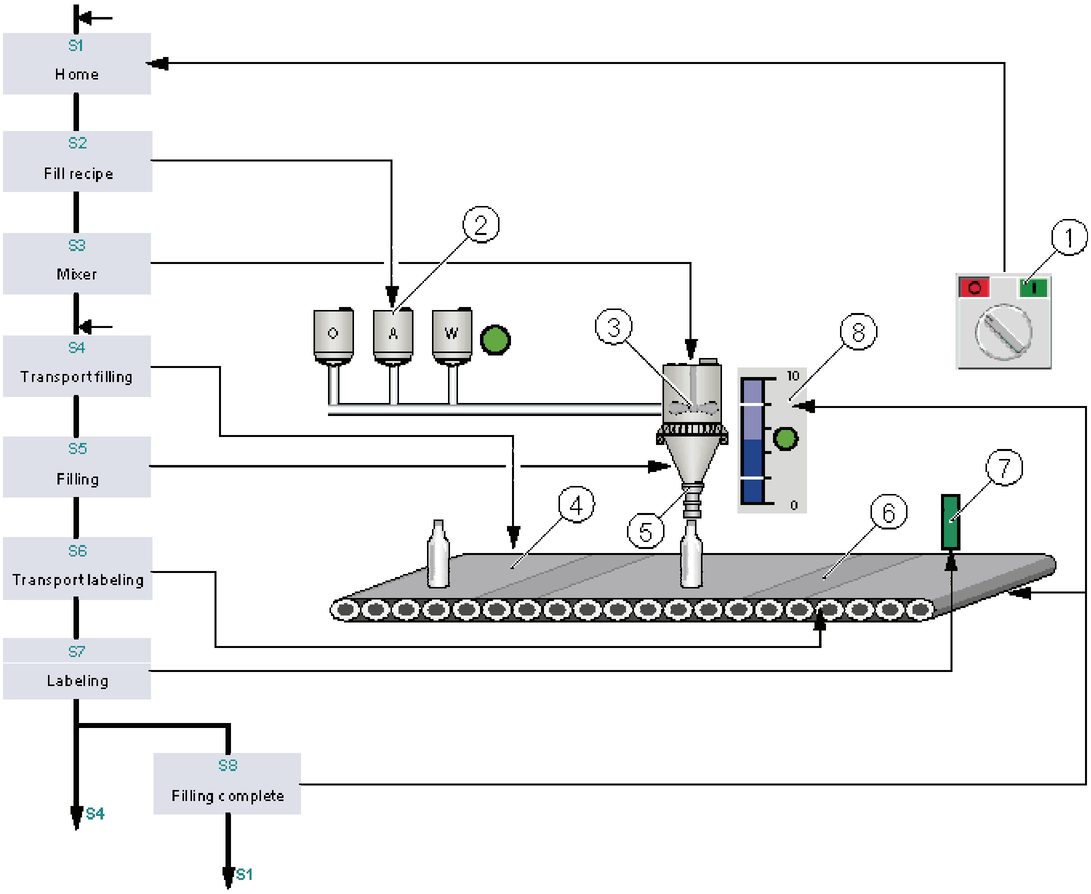
- An AWL block, which activates the conveyor belt and transports the bottles

  This block is called indirectly in GRAPH FB as soon as the corresponding tag is set in the "S4 Transport Filling" or in the "S6 Transport Labeling" step.

- An SCL block which calculates the best-before-date duration of the beverages

Structure of the GRAPH sequencer

The sequencer to be created reflects the exact sequence in which the program will be executed. The following figure shows in detail the individual steps within the GRAPH sequencer:



①	**Step 1 "Home" - Initial step** The initial step is always the first step when a GRAPH sequencer is called. During the execution of the initial step a counter for detecting the number of filled bottles is reset.
②	**Step 2 "Fill recipe ingredients" - Filling of the ingredients** The valves for each filling of the ingredients are opened for the duration defined using the HMI recipe function in the further course of the project. Depending on filling duration, there are different fill quantities for the respective ingredients.
③	**Step 3 "Mixer" - Mixing the ingredients** The output to activate the mixer is set. After 4 seconds, the output is reset and the mixer deactivated.
④	**Step 4 "Transport filling" - Transport of a bottle for filling** An STL block is activated via GRAPH interface; this block controls the conveyor belt and transports the bottles to the filling station.

⑤	**Step 5 "Filling" - Filling of the particular beverage** During filling the valve is in each case opened for 3 seconds to fill the bottle. In each filling process a counter, which detects the number of bottles already filled, increments by 1 per executed step. A maximum of 10 bottles can be filled.
⑥	**Step 6 "Transport labeling" - Transport of the bottles for labeling** The STL block is activated again via the GRAPH sequencer to get the conveyor belt to transport the filled bottle to the labeling station.
⑦	**Step 7 "Labeling" - Labeling the bottle** As soon as a bottle has been filled and transported, the output to activate the labeling station is set. At the labeling station, a label with the best-before date is affixed to each bottle. • When the beverage filling process has been completed, the sequencer starts again from the beginning (initial step "S1 Home"). • If the filling process is not yet completed, the steps S4 to S7 are repeated until all 10 bottles are filled and the filling process is completed. The best-before date is calculated via an SCL block. The best-before value is hereby calculated from the specific system time set on the CPU and the best-before-date duration of the produced beverages.
⑧	**Step 8 "Filling complete" - Filling completed** This step is only executed after the 10 bottles have been filled.

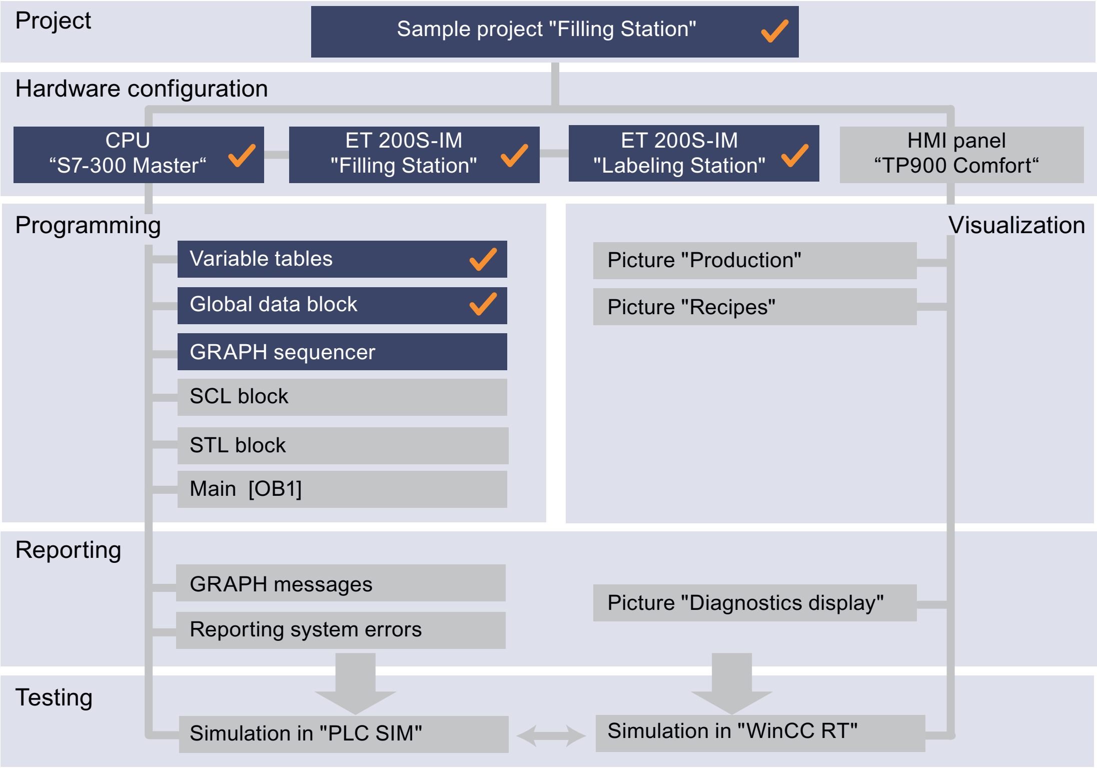
4.3.2 Create GRAPH function block

Introduction

In the following section you will create the GRAPH FB "GRAPH_Sequence". The GRAPH FB is used to quickly and easily program all program steps of the sample project and, if necessary, to control each step individually.

Progress of project

The following graphic shows you which step you perform next:



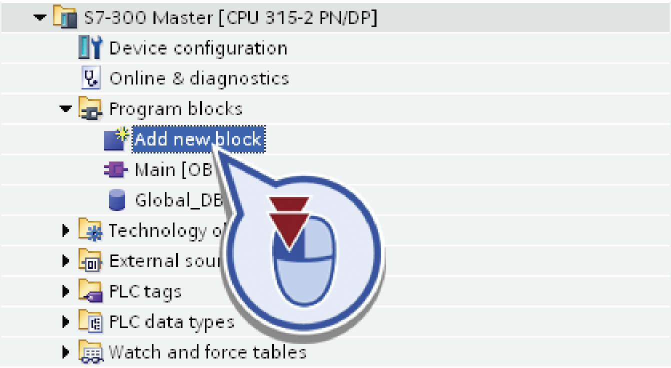
Requirement

You have created the global data block "Global_DB".

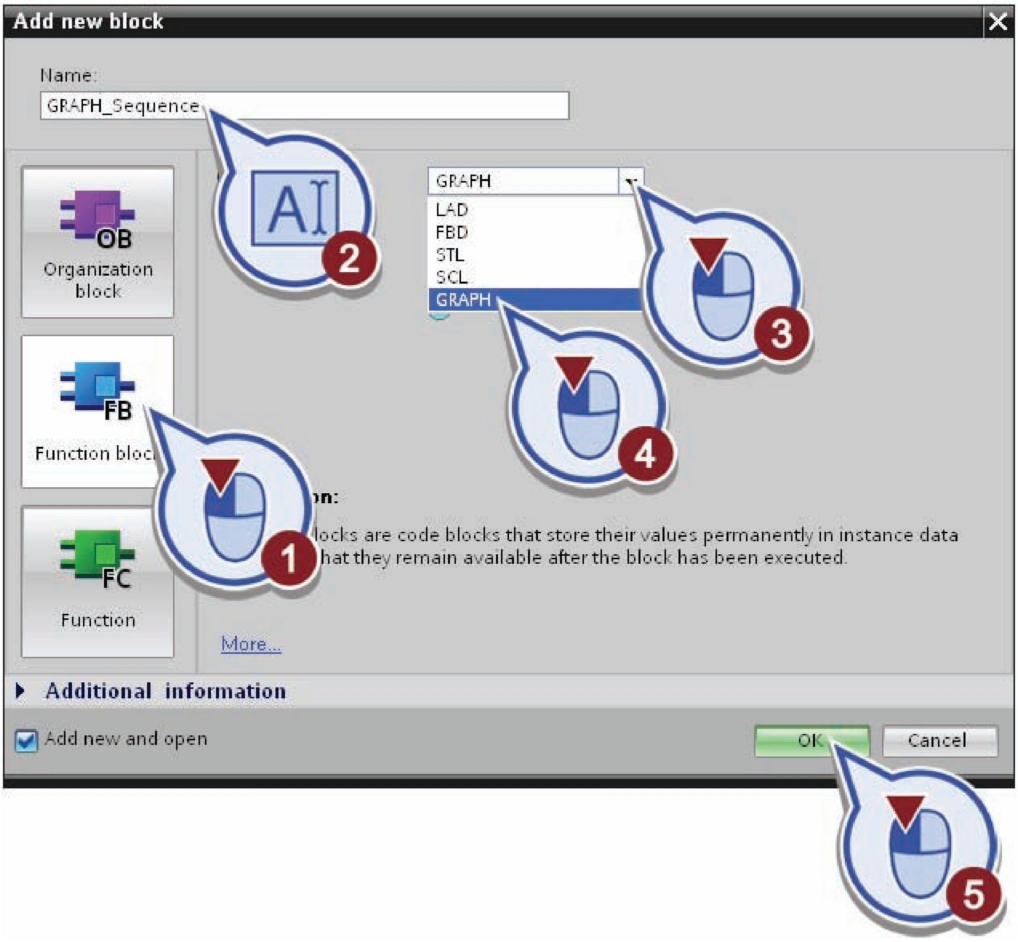
Procedure

To create the GRAPH FB, proceed as follows:

1. Open the "Program blocks" folder.
2. Double-click "Add new block".



3. To add a function block:
   – Click "Function block".
   – Assign the block name "GRAPH_Sequence".
   – Select the type "GRAPH".
   – Click "OK".



Add new block

Name: GRAPH_Sequence

…on:

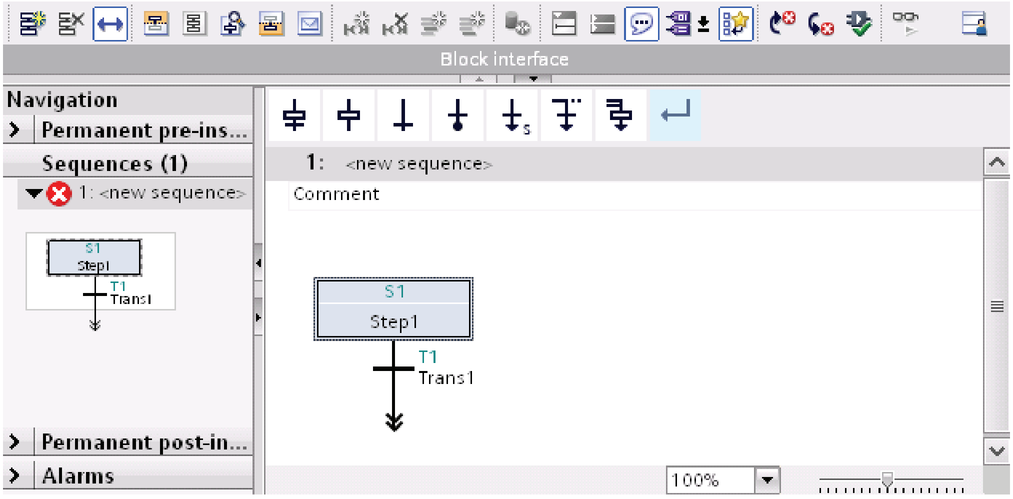
…locks are code blocks that store their values permanently in instance data …hat they remain available after the block has been executed.

☑ Add new and open

4. Save the project.

Result

You have successfully created the GRAPH FB "GRAPH_Sequence". After creating the GRAPH FB, the program editor opens automatically.



One step and one transition are already specified in the GRAPH FB. This first step is the initial step of the GRAPH sequencer.

The initial step can be recognized by its double border and is used to activate the sequencer.

4.3.3 Create sequencer

4.3.3.1 Structure of a sequencer

Introduction

In the following section, you will find an explanation of the elements of a sequencer, which you use to program the sample project "Filling Station".

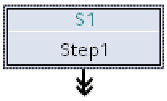
In the sample project "Filling Station" you work with the following elements:

- Step and transition
- Alternative branch
- Jump

Definition: Step

The tasks of a sequencer are broken down into separate steps. In the steps, you formulate instructions which are to be executed by the CPU under certain defined conditions. During the execution of the program each step is processed consecutively.

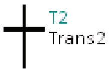
The following figure shows the graphic representation of a step:



Definition: Transition

Transitions contain the conditions for switching the sequencer from one step to the next. A transition is valid when all the conditions defined in it are satisfied. When the conditions in a transition are satisfied, it switches to the next step. The step or steps belonging to the transition are hereby deactivated and the next step activated.

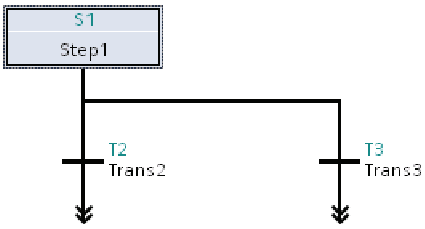
The following figure shows the graphic representation of a transition:



Definition: Alternative branch

If several transitions follow a step, the entry point for an alternative branch is located at this point. An alternative branch is an OR logic operation and consists of several parallel branches, each of which begins with a transition. If several transitions are satisfied simultaneously at the start of different branches, the transition farthest to the left has the highest priority in each case.

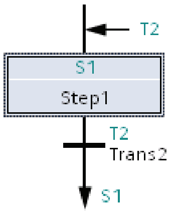
The following figure shows the graphic representation of an alternative branch:



Definition: Jump

A jump is the transition from one transition to any step within the sequencer. In addition to allowing you to run through parts of the sequencer again, the jump also permits the repeated processing of parts of the GRAPH FB sequencer. Jump and jump destination are each represented as arrows.

The following figure shows the graphic representation of a jump:



You are now familiar with all required elements for programming the GRAPH FB.

4.3.3.2 Insert steps and transitions

Introduction

In the following section you will insert further steps and transitions in the GRAPH PB "GRAPH_Sequence".

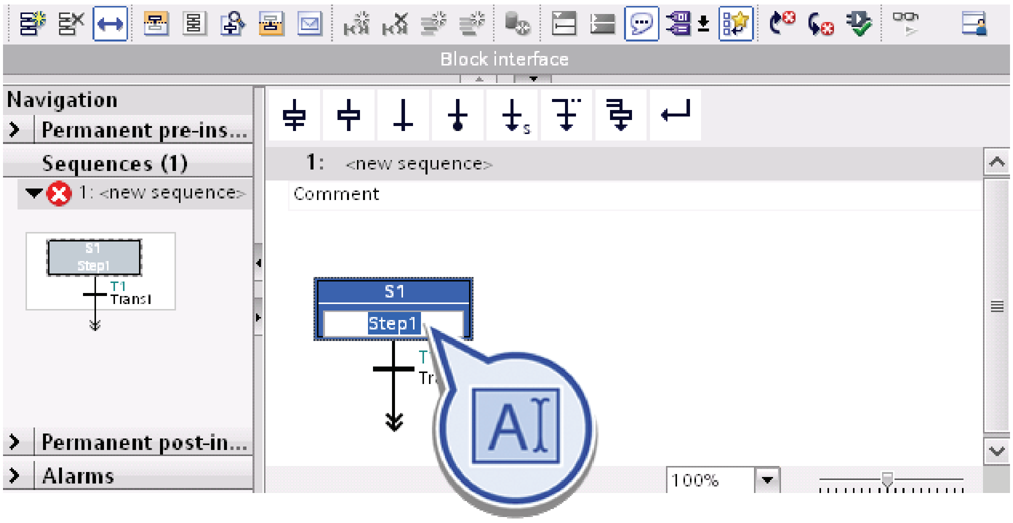
Requirement

You have the GRAPH FB opened in the program editor.

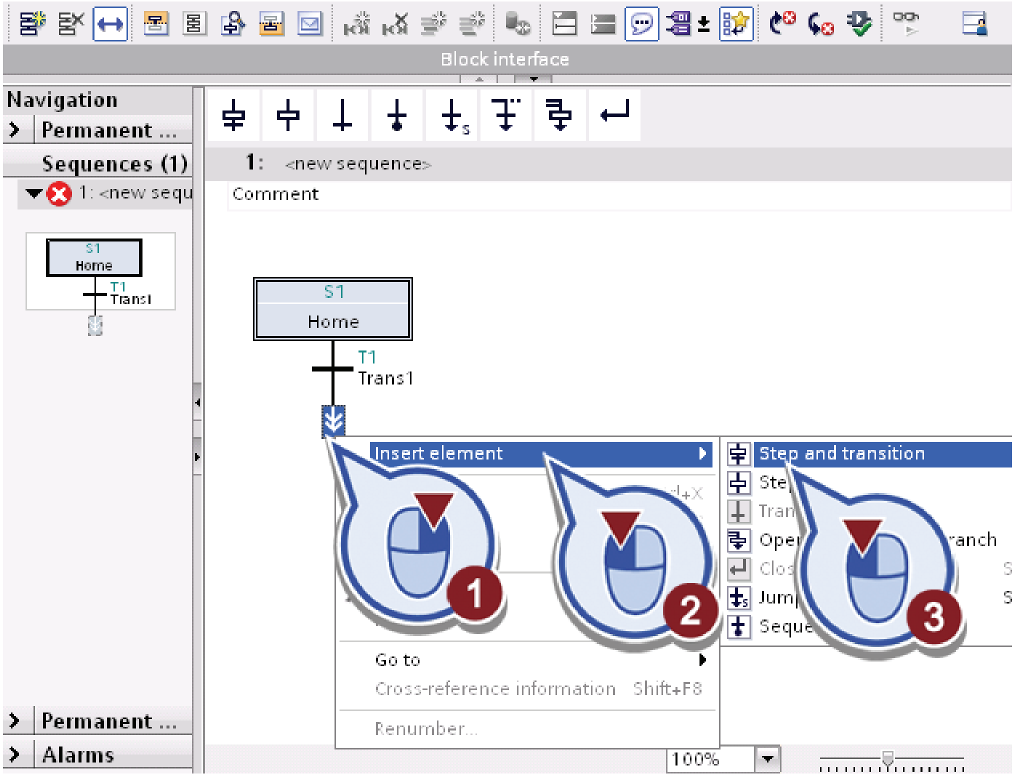
Procedure

To insert additional steps, proceed as follows:

1. Rename the step "Step1" to "Home".



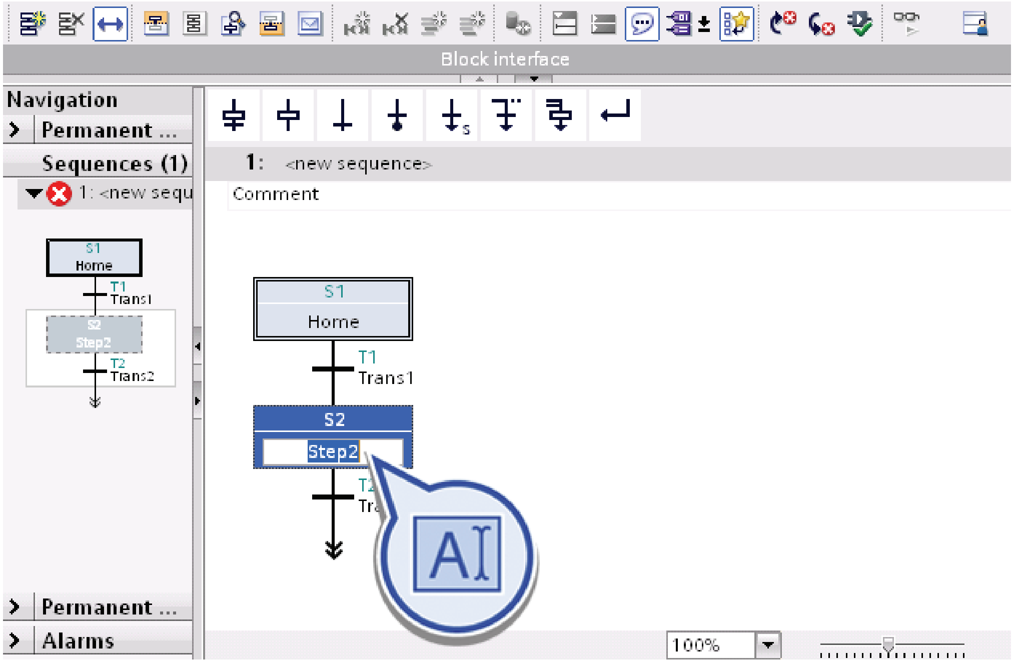
2. Add an additional step and a transition by a right-clicking the end of the branch and selecting "Insert element" > "Step and transition" from the shortcut menu.



3. Rename the step "Step2" to "Fill recipe ingredients".

Note

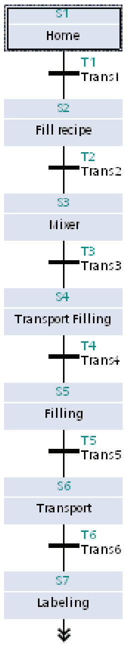
Please note that not more than 11 characters are displayed for the designation in the graphic representation of the sequencer. Step S2 is therefore displayed as "Fill recipe".



4. You require five additional steps and transitions for the sequencer. To insert these steps, proceed as described in the steps 2 and 3 . Rename the steps as follows:
   - Step3 > Mixer
   - Step4 > Transport Filling
   - Step5 > Filling
   - Step6 > Transport Labeling
   - Step7 > Labeling
5. Save the project by clicking "Save project" on the toolbar, or by pressing <Ctrl + S>.

Result

You have successfully inserted all the required steps and transitions in the GRAPH sequencer. The structure of the sequencer appears as follows:



However the steps and transitions still don't contain any actions or conditions for the transition. If you called the sequencer now all steps would be call successively starting with the initial step "S1 Home", without this having any effect on the conditions of the input and outputs of the CPU.

4.3.3.3 Insert alternative branch

Introduction

The created steps and transitions are processed linearly during the initialization of the GRAPH sequencer in the sample project and the processing is terminated after the step "S7 Labeling". In the following section, you insert an alternative branch in the sequencer. You can use the alternative branch to insert several transitions after a single step. Depending on whether a transition's condition is satisfied or not, another branch is run through during the execution of the sequencer.

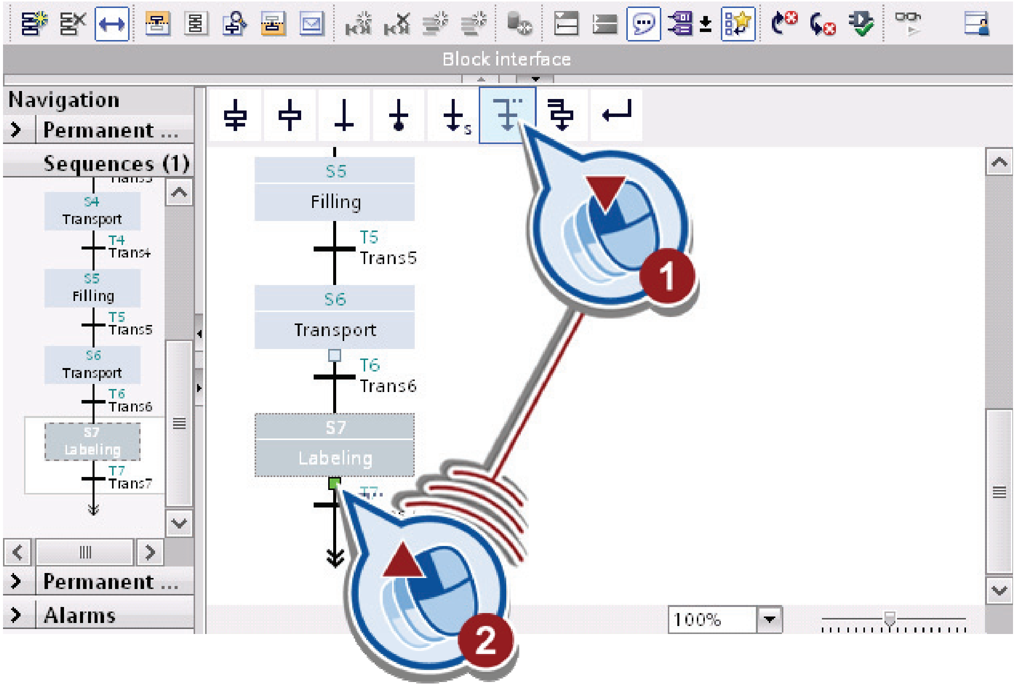
Requirement

You have created the sequence up to step "S7 Labeling".

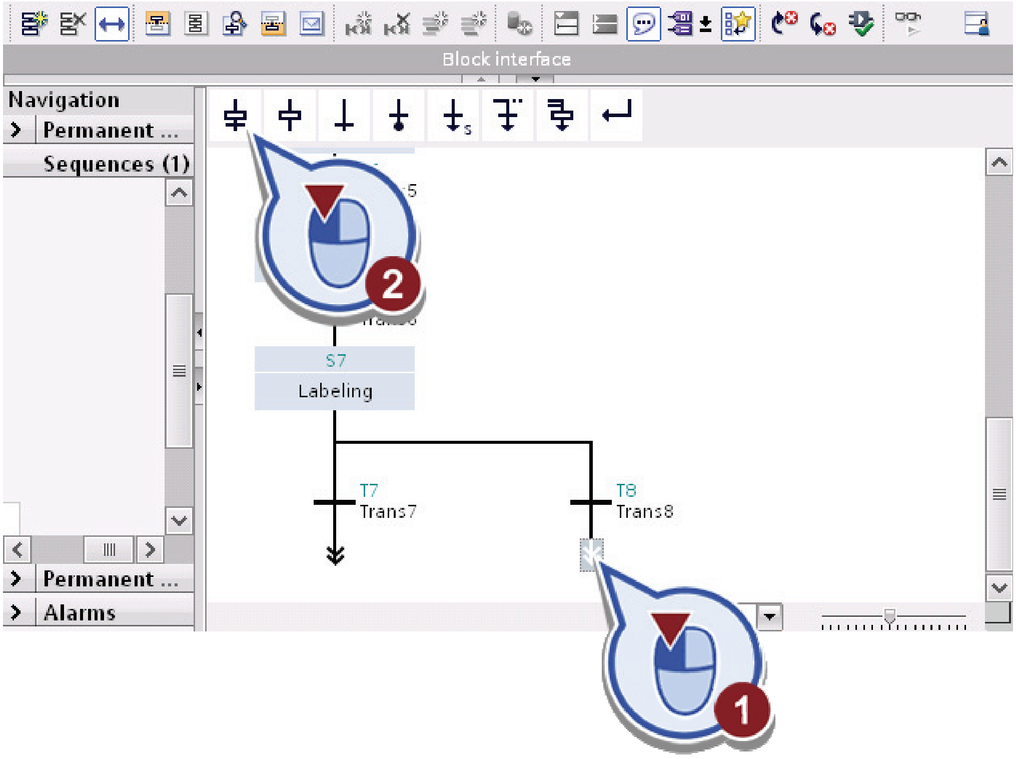
Procedure

To insert an alternative branch in the GRAPH FB, proceed as follows:

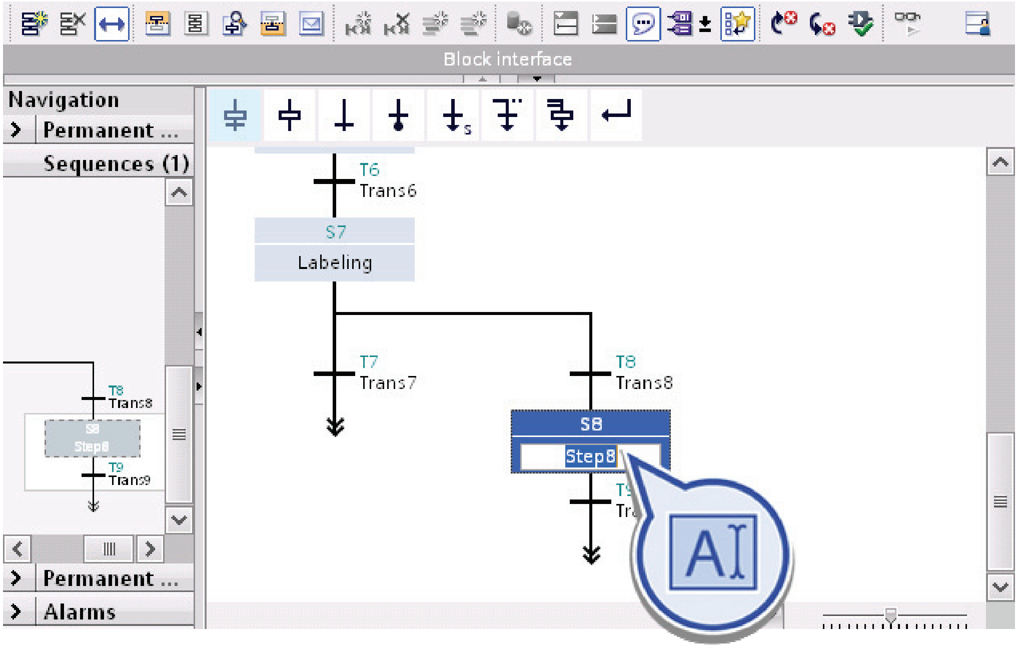
1. Insert an alternative branch after the last step "S7 Labeling".



2. Click the end of the alternative branch and insert another step and a transition.



3. Rename the step "Step8" to "Filling Complete".



4. Save the project.

Result

You have successfully inserted an alternative branch in the sequencer. The alternative branch permits two transitions ("Trans7" and "Trans8") to follow the step "S7 Labeling". Depending on the transition conditions, either the left or right branch of the sequencer is processed.

4.3.3.4 Inserting jumps

Introduction

In the following section you will insert a jump after the step "S7 Labeling" and another jump after the step "S8 Filling Complete":

- With less than 10 bottles the filling process of the mixed quantity is not yet completed. In this case all steps after the mixing of the ingredients (step "S3 Mixer") must be performed again after the "S7 Labeling" until the filling process is completed, i.e. until a total of 10 bottles has been filled.
- When 10 bottles are filled, the step "S8 Filling Complete" is executed. After this step there is a jump to the beginning of the sequencer (step "S1 Home") to run through the program again.

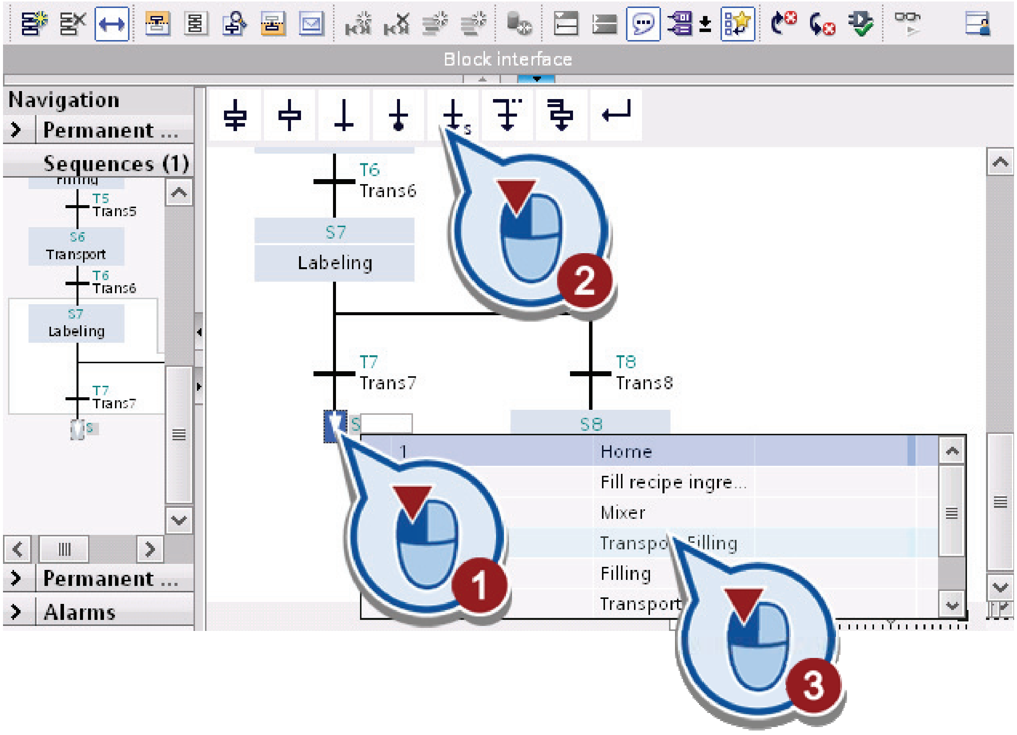
Requirement

You have created the step "S7 Labeling" and "S8 Filling Complete".

Procedure

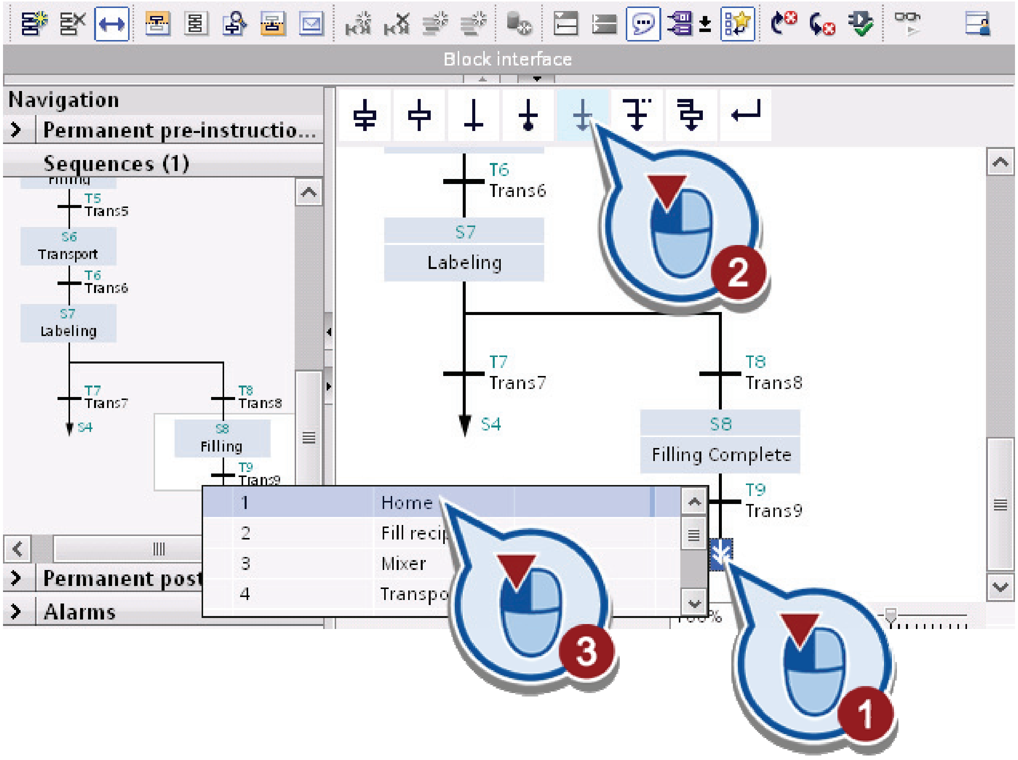
To insert a jump in the GRAPH FB, proceed as follows:

1. Click the end of the left branch and insert a jump. A list with the created steps appears during the insertion process. Select the step "S4 Transport Filling".



To recognize that you inserted the jump successfully, check that the double arrow at the end of the sequence has become a single arrow and that the designation "S4" is on the right-hand side of it. This means that at this point of the sequencer a jump is made to step "S4 Transport Filling" and the sequencer is run through again as of step "S4".

2. Insert the second jump by clicking on the right end of the sequencer and selecting the corresponding icon. Select the step "S1 Home".

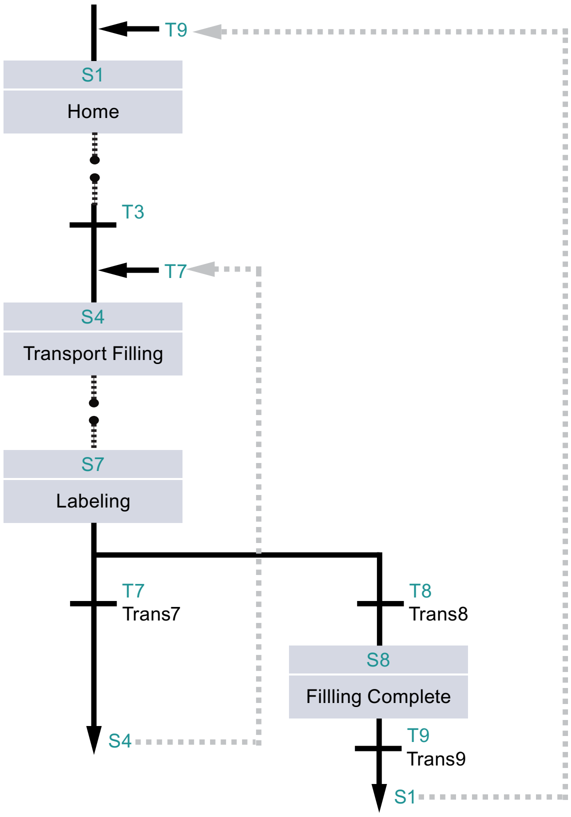


3. Save the project.

Result

You have successfully inserted two jumps in the sequence.

The following figure shows a shortened representation of the sequence:



The inserted jumps affect the sequencer as follows:

- If the transition condition "Trans7" is satisfied after the step "S7 Labeling", a jump is executed from step "S4 Transport Filling". The steps from "S4 Transport Filling" to "S7 Labeling" are repeated until the transition condition "Trans8" is satisfied in place of transition condition "Trans7".
- If the transition condition "Trans8" is satisfied after the step "S7 Labeling" and the step "S8 Filling Complete" are completed, a jump is executed back to "S1 Home". In other words, the sequencer is executed again from the beginning.

Therefore, the sequencer has no end, but is executed again and again after the first call.

4.3.3.5 Compile a project

Introduction

In the following section you will compile the project "Filling Station", to check whether the project is error-free up to the current configuration status. Projects always have to be compiled before they are downloaded to the CPU.

Compiling project data

The following project data must be compiled:

- Hardware project data

  For example, configuration data of the devices or networks and connections

- Software project data

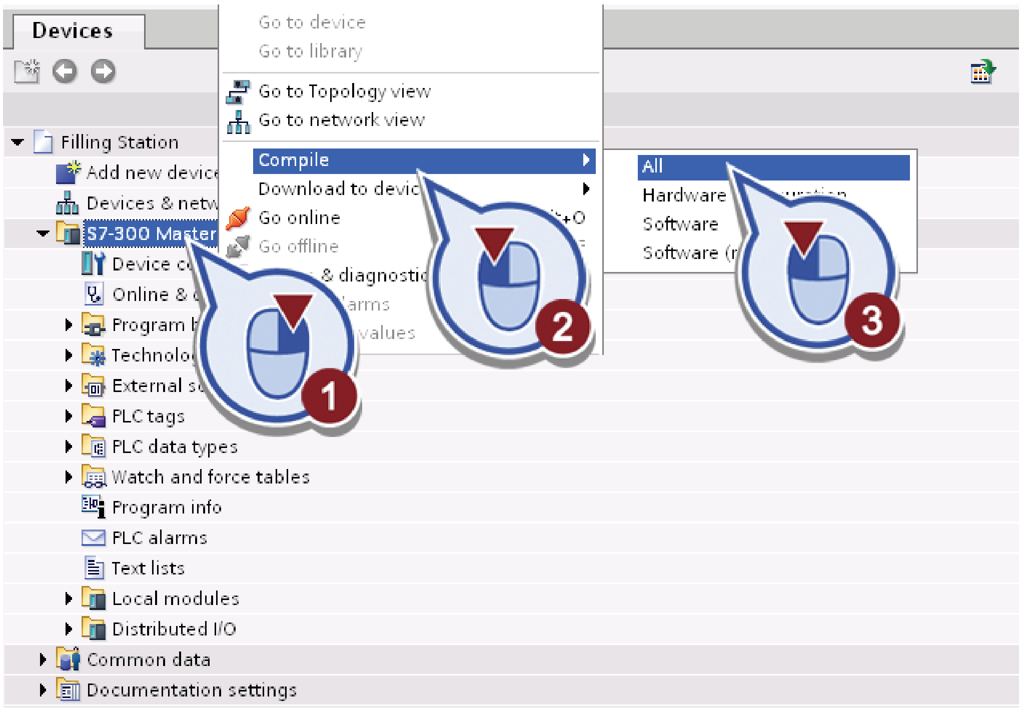
  For example program blocks or HMI screens

You can compile hardware configuration data and program data separately or together. Which data is compiled depends on the position of the project tree in which you trigger the compiling process.

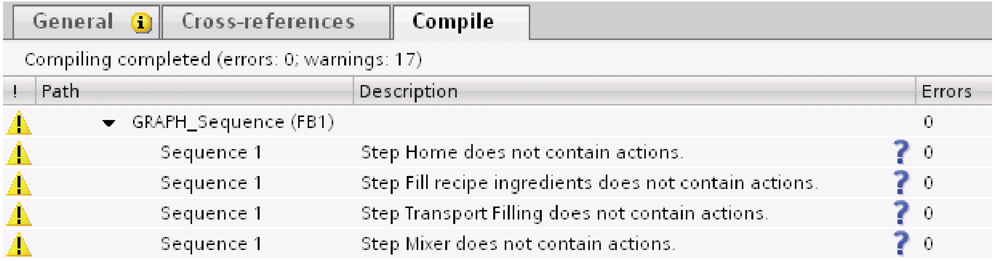
Procedure

To compile all of the currently created hardware and software data of the project "Filling Station", proceed as follows:

1. In the Project tree, click CPU "S7-300 Master" and right-click in the shortcut menu to call the command "Compile > All". In this way, all project data previously created in the CPU is compiled.



2. Go to "Info > Compile" in the inspector window to check whether compiling was completed successfully.



Compiling completed (errors: 0; warnings: 17)

!	Path	Description		Errors
⚠	GRAPH_Sequence (FB1)			0
⚠	Sequence 1	Step Home does not contain actions.	?	0
⚠	Sequence 1	Step Fill recipe ingredients does not contain actions.	?	0
⚠	Sequence 1	Step Transport Filling does not contain actions.	?	0
⚠	Sequence 1	Step Mixer does not contain actions.	?	0

– If an error occurs, such as an address conflict between two tags, the project cannot be downloaded to the CPU. If an error occurs, double-click the corresponding error message to automatically navigate to the object at which the error occurred to correct this error.

– In the event of warnings due to missing actions or conditions, for example, as shown in the illustration, the subsequent download to the simulated CPU cannot be prevented and they can be ignored for execution of the sample project.

3. Save the project.

Result

You have successfully compiled the currently created project data in the sample project "Filling Station".

4.3.4 Programming steps

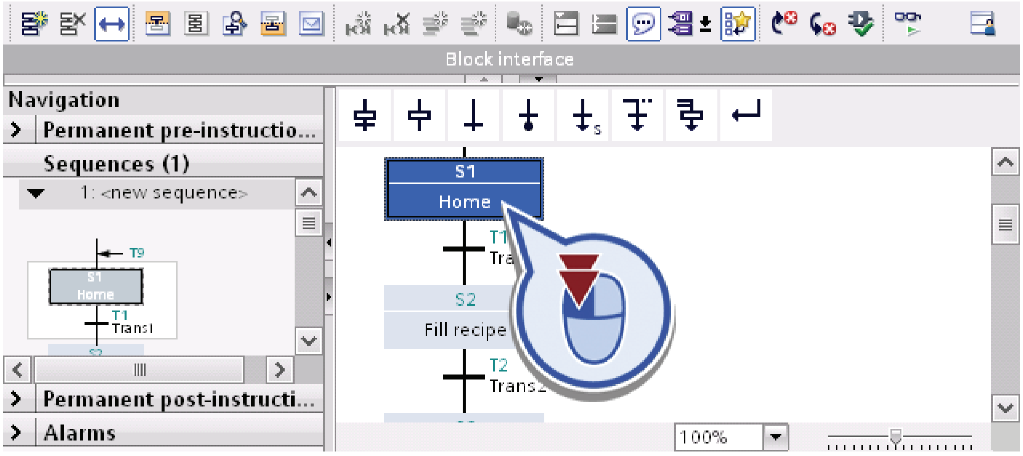
4.3.4.1 Step elements

Introduction

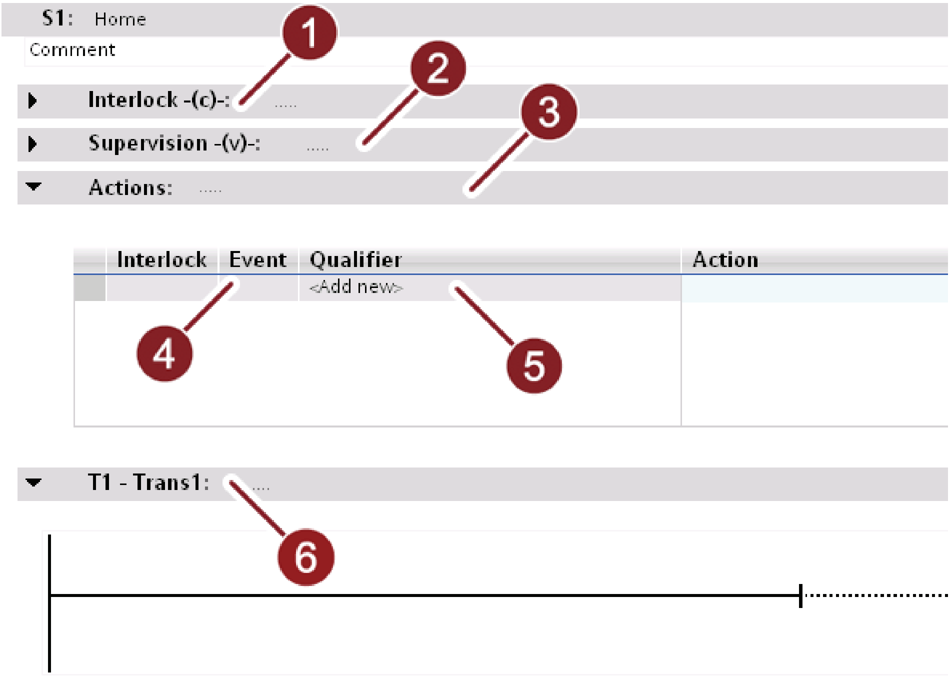
The following section will provide you with information about the elements of a step, with the help of which you can execute various functions in a step. The use of the individual elements is optional. You can also define conditions (transitions), which have to be satisfied in order for the step to be processed.

Step elements

To display the elements of a step, double-click the desired step in the sequencer:



The following figure shows the elements of a step:



The individual elements have the following functions:

①	Interlock: An interlock is a programmable interlocking condition within a step which blocks the execution of the step. If the condition is satisfied, this is the best outcome: There are no faults. If, under certain circumstances (when an error occurs, for example) the step is not to be executed, you can define this in the Interlock. • If all the conditions in the Interlock are satisfied, the actions linked to the interlock are executed. • If the conditions defined in the Interlock are not satisfied, the sequencer stops and the next step is not executed. You can also configure messages to issue a corresponding error message.
②	Supervision: A supervision (step monitoring) is a programmable condition within a step, with which you monitor the execution of a step. If the condition is **not** satisfied, this is the best outcome: There are no faults. If a fault occurs and therefore a monitoring error, the switching to the next step is blocked. In online mode, the occurrence of a fault in a step is indicated with a "V" to the left of the sequencer view. You can also configure messages to issue a corresponding error message.
③	Actions: An action contains the actual instructions for the process control. You can make the execution of the instructions dependent on the occurrence of an interlock or the occurrence of other events which are to be defined. The ID of an action is used to define the type of action to be executed. Yon can program instructions in an action, such as value assignments, block calls or a counter call.

④	**Event:** An event is the change in the signal state of a step, a supervision, or an interlock, or the acknowledgment of a message or the coming of a registration. An event can be recorded and processed with an action.
⑤	**Identifier:** The identifier is used to specify the type of action in which the GRAPH step is to be executed. Automatically predefined placeholders are created during the selection of certain standard actions (for example, when calling a counter). In the sample project, use the identifier "N", via which you assign a tag to the value while the step is active.
⑥	**Transitions:** Transitions contain the conditions for switching to the next step. When the conditions for a transition are satisfied, the process switches to the next step.

4.3.4.2 Insert multistep transition condition

Introduction

In the following section you will insert a multistep transition condition. You can use this transition condition to prevent the continued processing of a sequencer when a group error occurs, regardless of the position of the program in the sequencer.

The multistep transition condition in the sample project "Filling Station" is implemented using an NC contact, which is interconnected with a tag for a group error.

Definition: NC contact

An NC contact is represented with the symbol "---| / |---" in the program.

- The activation of an NC contact depends on the signal state of the associated operand. When the operand has signal state "1," the contact is opened and the current flow to the right power rail is interrupted. The output of the instruction in this case results in the signal state "0".
- When the operand has signal state "0," the NC contact remains closed. The current flows through the NC contact and the output of the instruction is set to the signal state "1".

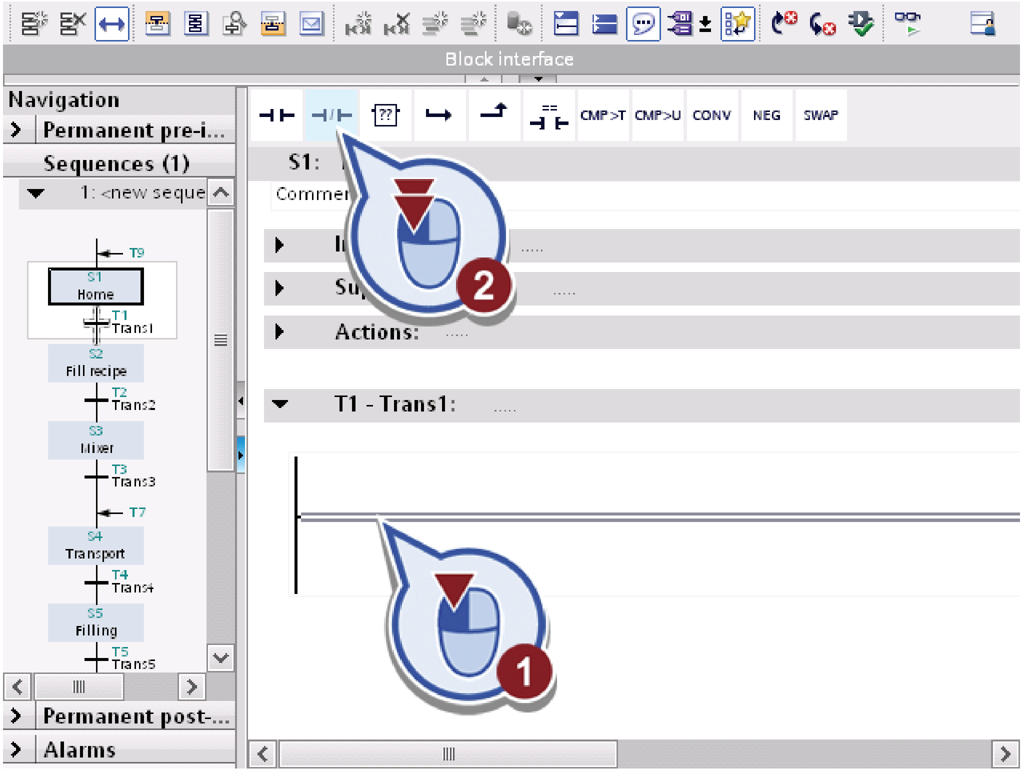
Requirement

The step "S1 Home" is opened in the program editor.

Procedure

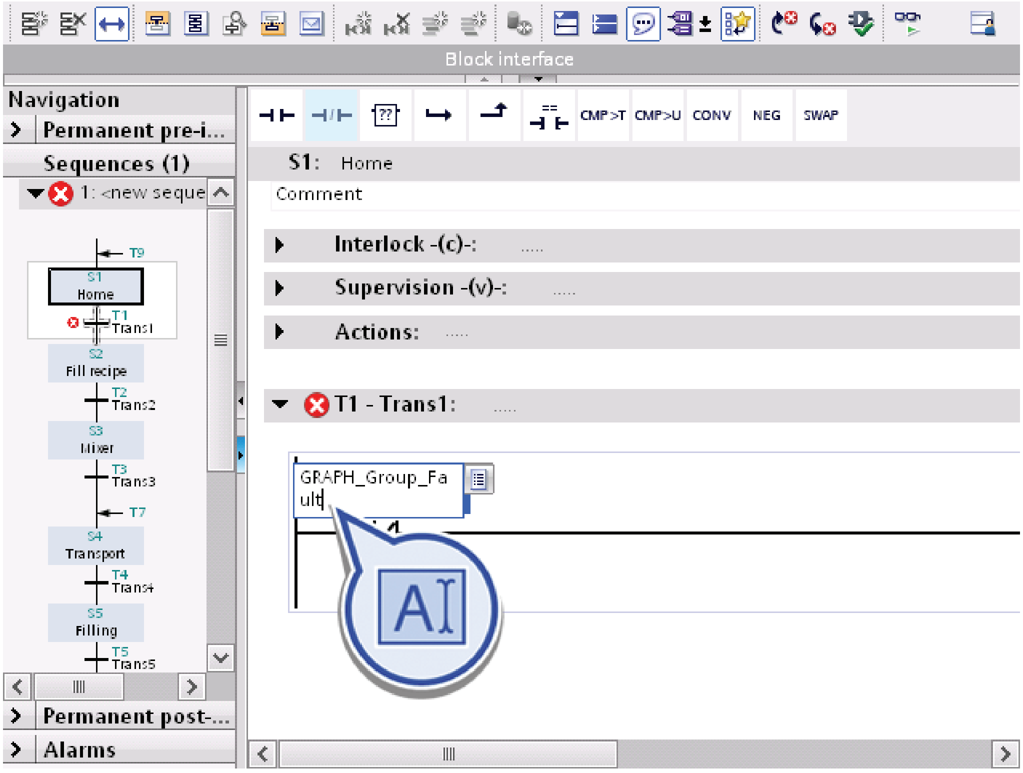
To insert the transition condition, proceed as follows:

1. At "T1 – Trans1" in the work area, click the power rail and then click the "NC contact" on the favorites bar.



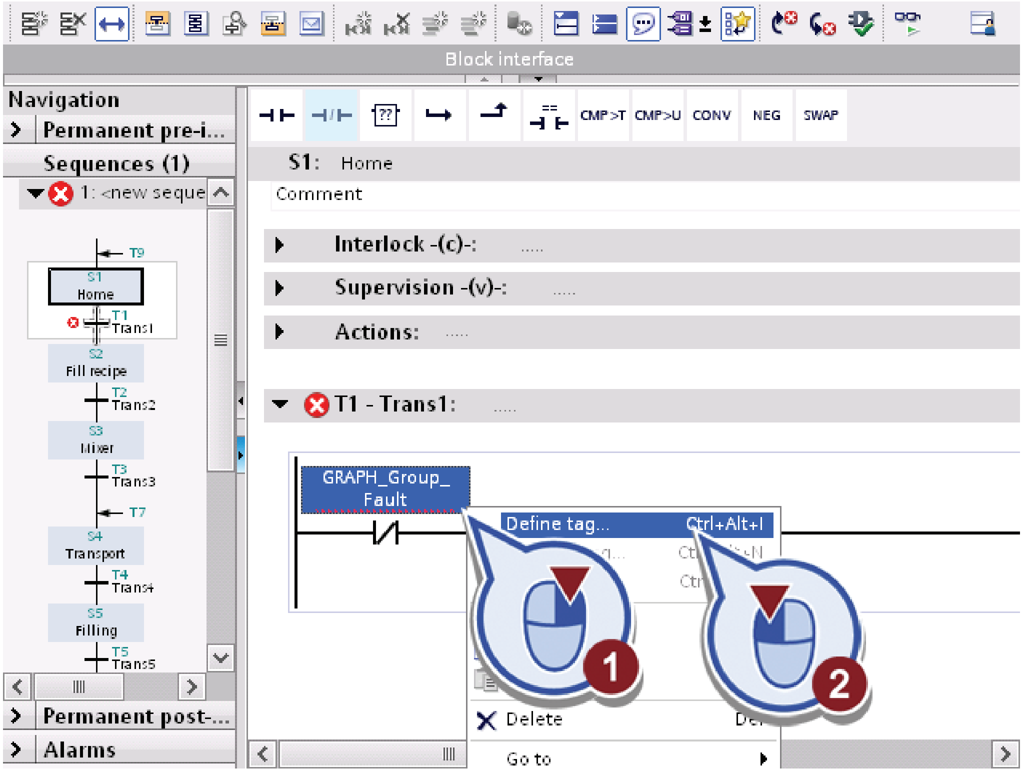
An NC contact is inserted. The characters "<??.?>" represent an operand placeholder.

2. Double-click the operand placeholder, start entering the name of the tag and rename it to "GRAPH_Group_Fault".



GRAPH_Group_Fault

3. Right-click the operand and select "Define tag" from the shortcut menu.



The "Define tag" dialog box appears.

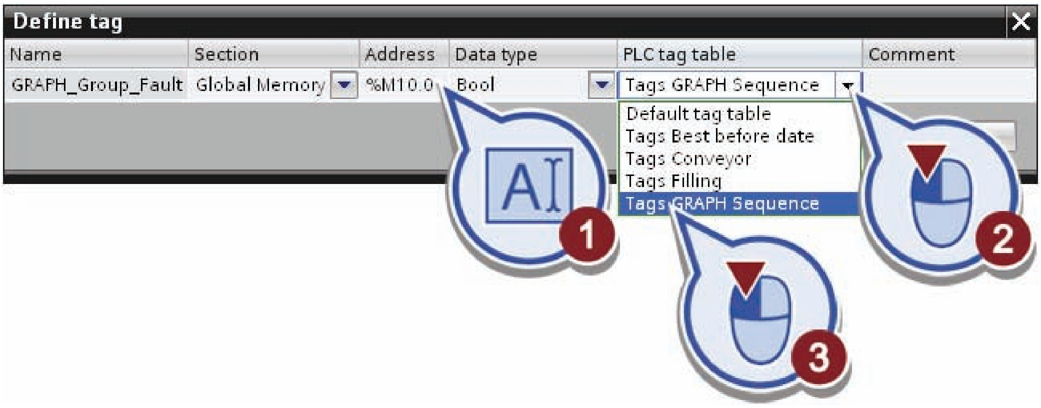
4. Define the tag with the following properties:

   – Section: "Global Memory"

   – Address: "M10.0"

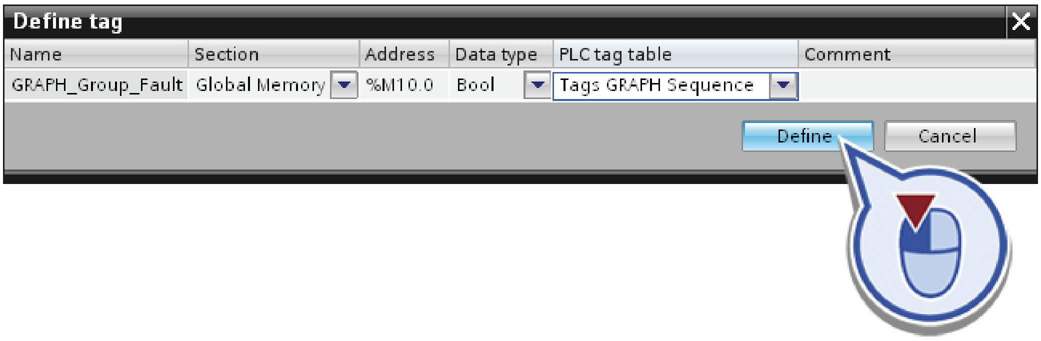
   – Date type: "Bool"

   – PLC tag table: "Tags GRAPH Sequence"


Define tag

Name	Section	Address	Data type	PLC tag table	Comment
GRAPH_Group_Fault	Global Memory	%M10.0	Bool	Tags GRAPH Sequence	

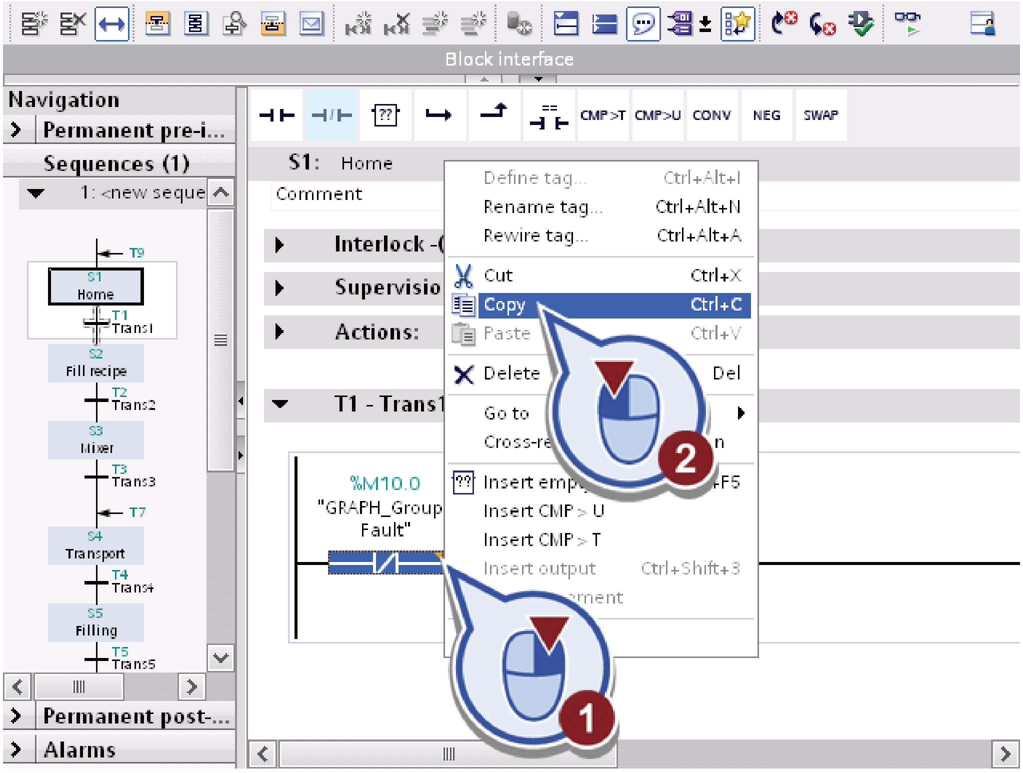
5. Confirm the dialog by clicking "Define".



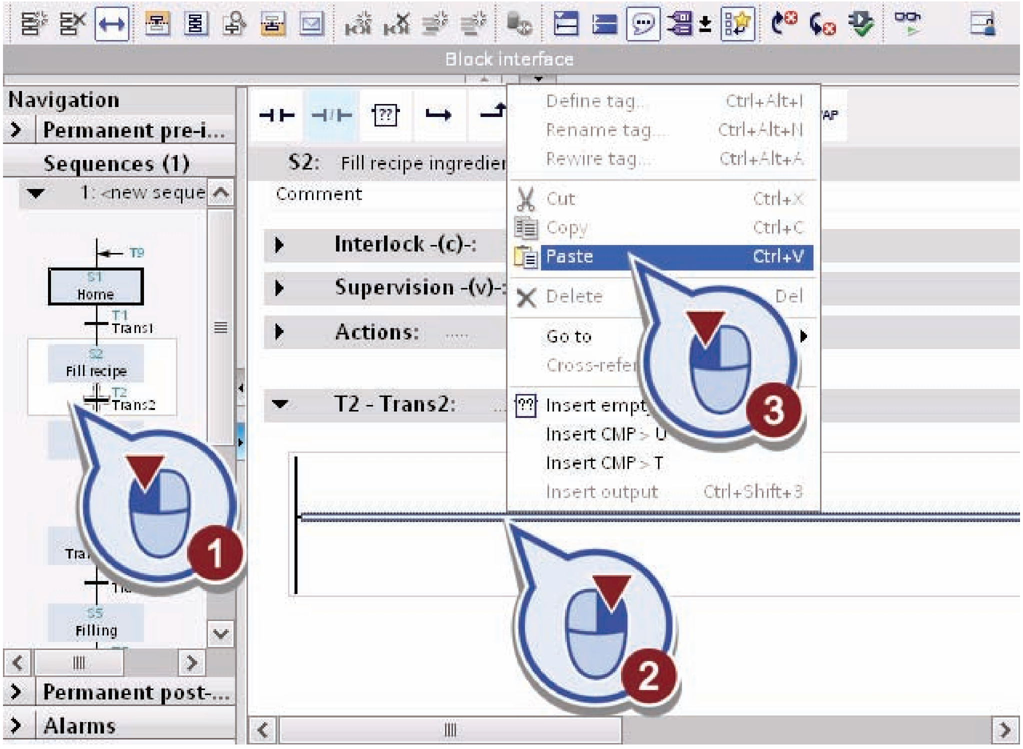
Define tag

Name	Section	Address	Data type	PLC tag table	Comment
GRAPH_Group_Fault	Global Memory	%M10.0	Bool	Tags GRAPH Sequence	

6. Right-click the NC contact and select "Copy" from the shortcut menu.



7. Click the step "S2 Fill Recipe". At "T2 - Trans2", right-click the power rail and select "Paste" from the shortcut menu.



8. To assign the transition condition to all steps, repeat step 7 for each of the remaining transitions in the sequencer.

9. Save the project.

Result

You have successfully added the same multistep transition condition to each step of the sequencer. The power rail of the transition is interrupted as soon as the tag "GRAPH_Group_Fault" has signal state "1". This prevents switching to the next step.

4.3.4.3 Program Step S1 Home

Introduction

In the following section you will program an action for the step "S1 Home".

- The action causes the counter tag "GRAPH_Count_Bottle " (number of already filled bottles) to be set to the value "0".
- The action should be executed as soon as the step is executed. You define this using the event identifier "S1".
- The action should be executed as soon as the step is activated. You use the action qualifier "N" to define this.

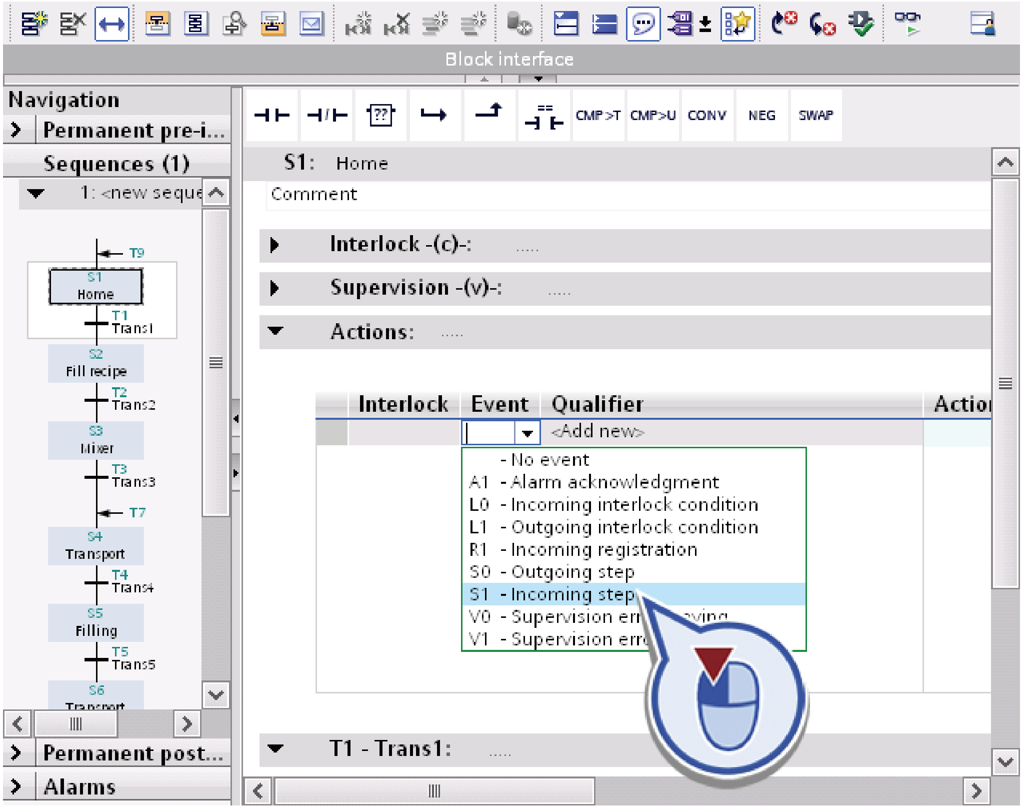
Requirement

You have opened the step "S1 Home".

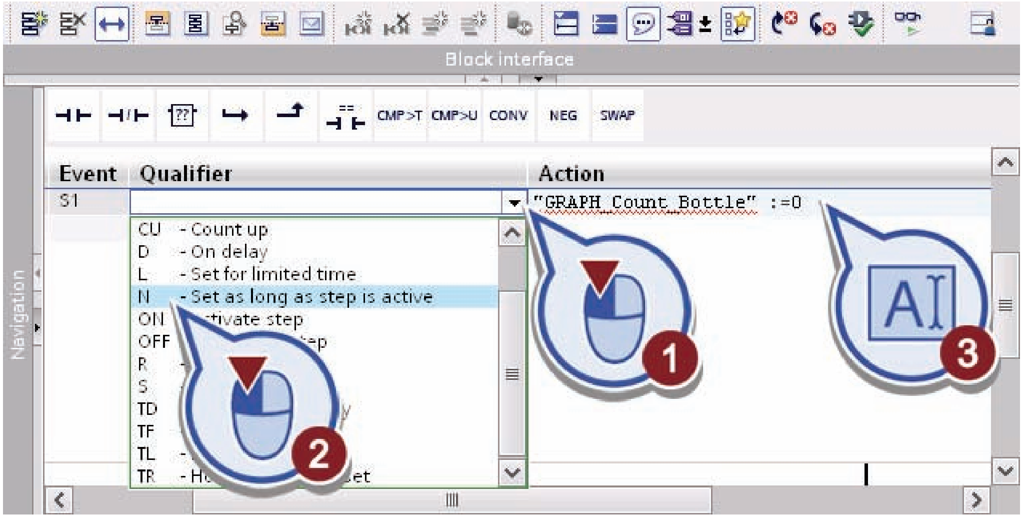
Procedure

To program the action, proceed as follows:

1. Select the event "S1 - Incoming step" from the "Actions" dialog.

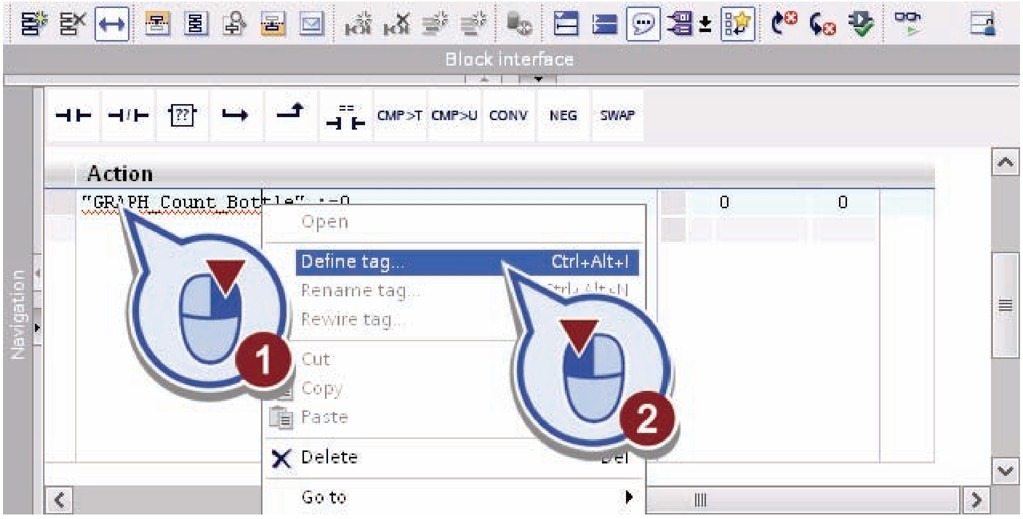


2. Select the qualifier "N - Set as long as step is active". Enter the text "GRAPH_Count_Bottle" :=0 in the "Action" column.



Event	Qualifier	Action
S1		"GRAPH_Count_Bottle" :=0

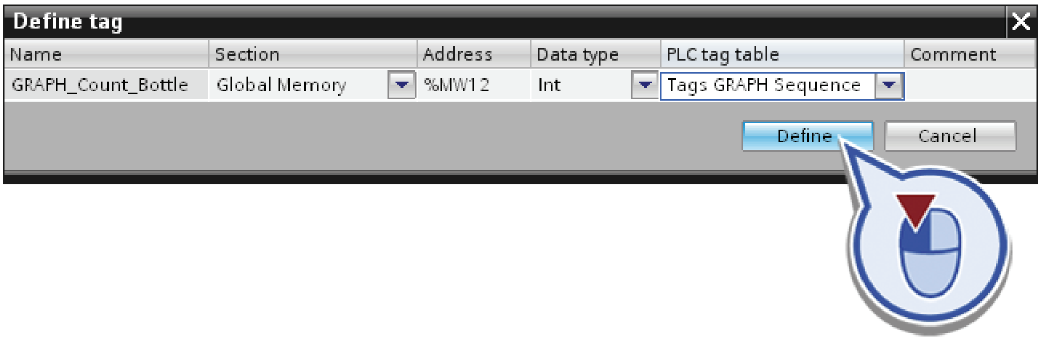
3. Right-click the text "GRAPH_Count_Bottle" and select "Define tag" from the shortcut menu.



Action		
"GRAPH_Count_Bottle" :=0	0	0

4. Define the tag with the following properties:
   - Section: "Global Memory"
   - Address: "MW12"
   - Date type: "Int"
   - PLC tag table: "Tags GRAPH Sequence"

   Confirm the dialog by clicking "Define".



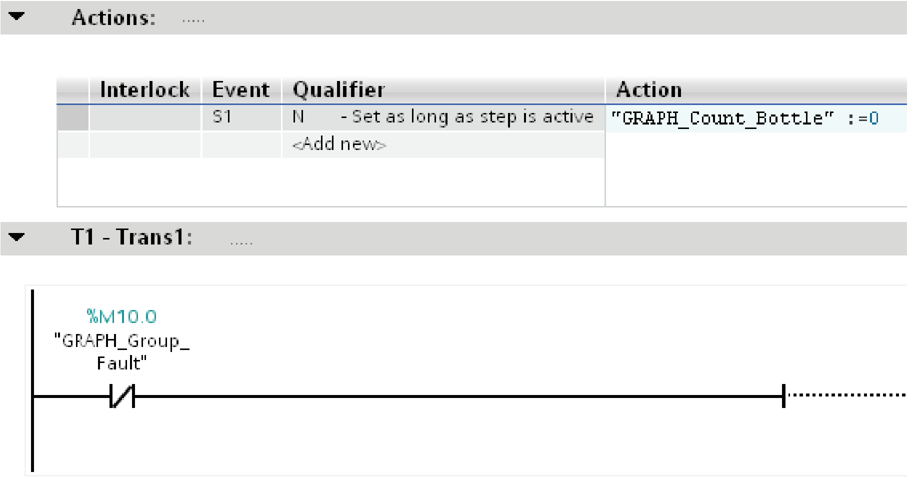
Define tag

Name	Section	Address	Data type	PLC tag table	Comment
GRAPH_Count_Bottle	Global Memory	%MW12	Int	Tags GRAPH Sequence	

5. Save the project.

Result

You have successfully programmed an action for the step "S1 Home". Each time this step is called, the integer tag "Graph_Count_Bottle" is set to "0". If no error occurs while the step is being executed, the transition condition is satisfied and the sequencer switches to step "S2 Fill recipe ingredients".



Actions: .....

Interlock	Event	Qualifier	Action
	S1	N - Set as long as step is active	"GRAPH_Count_Bottle" :=0
		<Add new>	

T1 - Trans1: .....

4.3.4.4 Step S2 Fill recipe ingredients - Programming actions

Introduction

In the following section you will program the actions for mixing the ingredients in the step "S2 Fill recipe ingredients". You can use the tags to program the opening and closing of the valves, depending on the specific recipe. The tags are defined in the global data block "Global_DB".

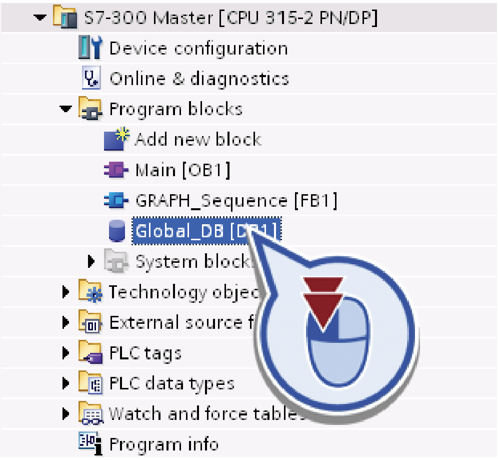
Requirement

You have created the global data block "Global_DB" and the step "S2 Fill recipe ingredients".

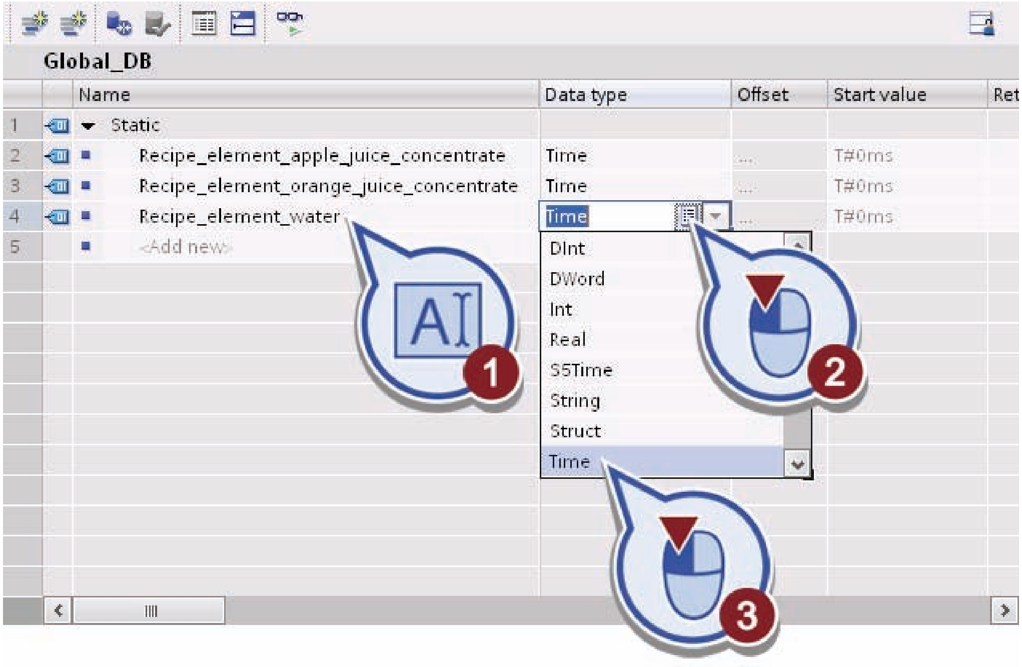
Procedure

To program the actions, proceed as follows:

1. Open the "Program blocks" folder.
2. Double-click the global data block "Global_DB".

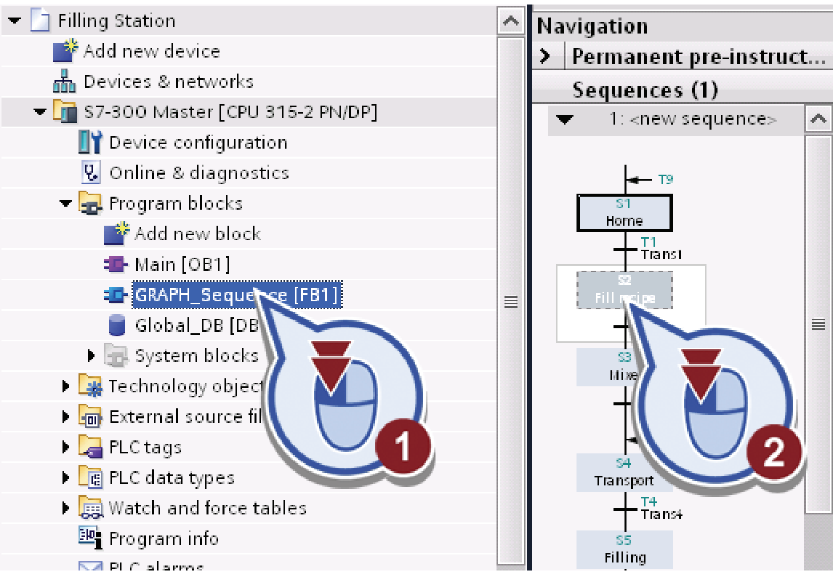


3. Define the following tags each with the data type "Time":
   - Name: "Recipe_element_apple_juice_concentrate"
   - Name: "Recipe_element_orange_juice_concentrate"
   - Name: "Recipe_element_water"

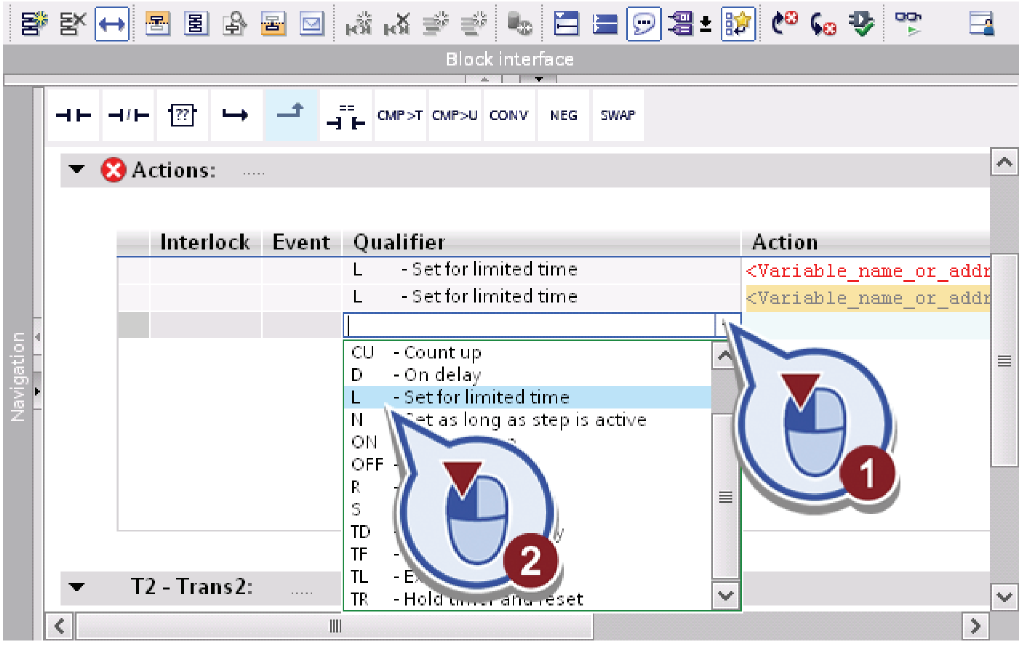

Global_DB

	Name	Data type	Offset	Start value	Ret
1	Static				
2	Recipe_element_apple_juice_concentrate	Time	...	T#0ms	
3	Recipe_element_orange_juice_concentrate	Time	...	T#0ms	
4	Recipe_element_water	Time	...	T#0ms	
5	<Add new>				

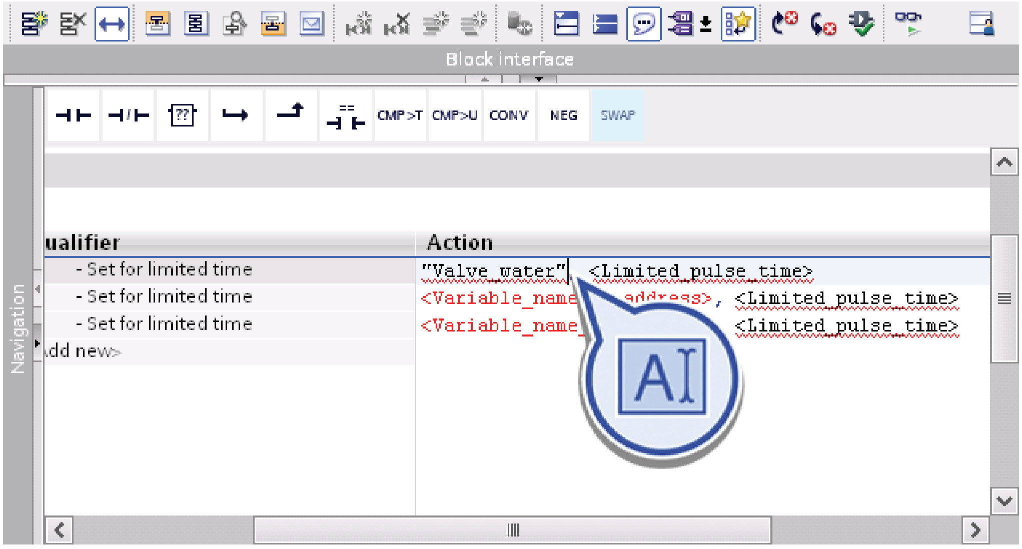
4. Open the step "S2 Fill recipe ingredients" and interconnect it with the defined tags.



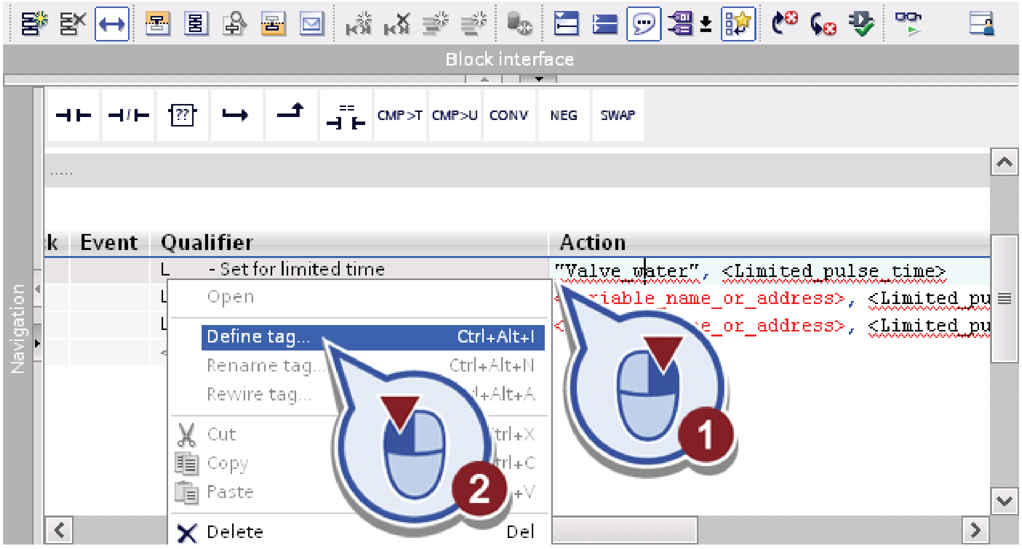
5. Under "Actions" in the column "Qualifier" select the qualifier "L - Set for limited time" three times.



6. In the first line, replace the text <Variable_name_or_address> with "Valve_water".



7. Right-click the text "Valve_water" and select "Define tag" from the shortcut menu.



8. Define the tag with the following properties:

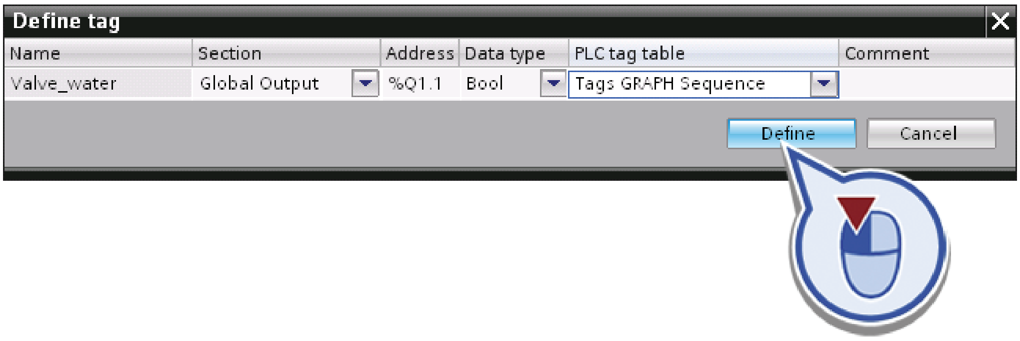
   – Section: "Global Output"

   – Address: "A1.1"

   – Date type: "Bool"

   – PLC tag table: "Tags GRAPH_Sequence"

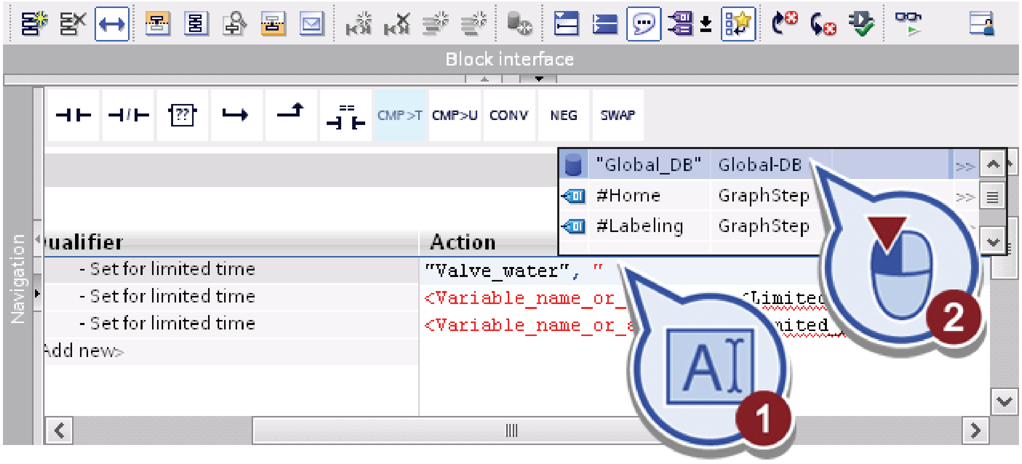
   Confirm the dialog by clicking "Define".



Define tag

Name	Section	Address	Data type	PLC tag table	Comment
Valve_water	Global Output	%Q1.1	Bool	Tags GRAPH Sequence	

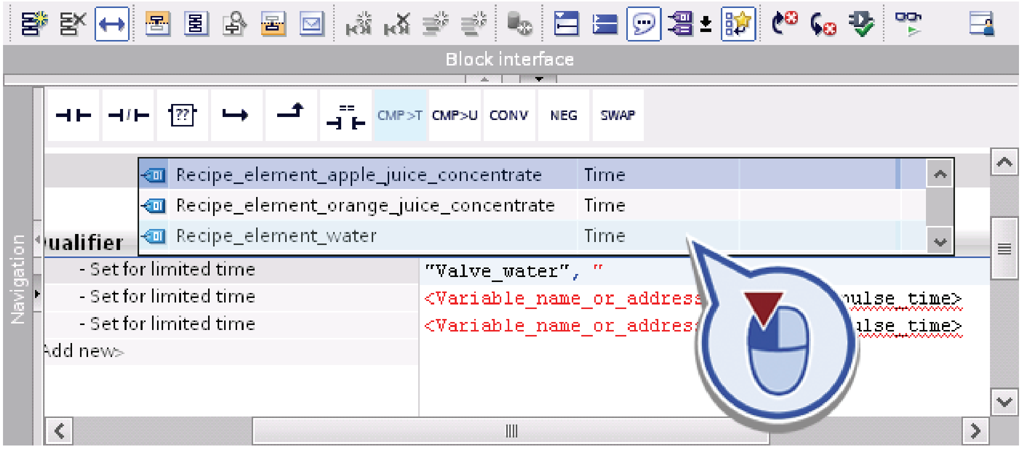
9. Replace the text <Limited_pulse_time> with "Global_DB". As soon as you enter the inverted commas a window automatically opens for selecting the already created tags and blocks. Select the global data block "Global_DB".



"Global_DB"	Global-DB	>>
#Home	GraphStep	>>
#Labeling	GraphStep	

ualifier	Action
- Set for limited time	"Valve_water", "
- Set for limited time	<Variable_name_or_ <Limited
- Set for limited time	<Variable_name_or_a nited
Add new>	

10. Press the "Return key" to confirm the selection of the "Global_DB". A window now opens for selecting data block tags which have already been created. Select the tag "Recipe_element_water".



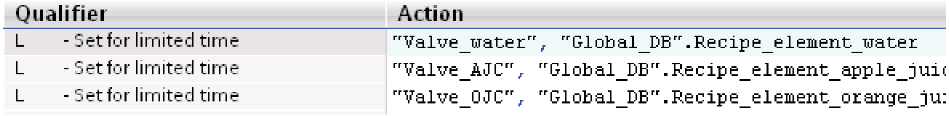
Recipe_element_apple_juice_concentrate	Time
Recipe_element_orange_juice_concentrate	Time
Recipe_element_water	Time

ualifier	
- Set for limited time	"Valve_water", "
- Set for limited time	<Variable_name_or_address ulse_time>
- Set for limited time	<Variable_name_or_address lse_time>
Add new>	

11. Repeat steps 6 to 10 for the second line with the following properties:
    - Replace the text <Variable_name_or_address> with "Valve_AJC".
    - Define the tag with the following properties:
    - Section: "Global Output"
    - Address: "A1.2"
    - Date type: "Bool"
    - PLC tag table: "Tags GRAPH_Sequence".
    - Replace the text <Limited_pulse_time> with quotation marks.
    - Select the global data block "Global_DB".
    - Press the "Return key" to confirm and select the tag "Recipe_element_apple_juice_concentrate".
12. Repeat steps 6 to 10 for the third line with the following properties:
    - Replace the text <Variable_name_or_address> with "Valve_OJC".
    - Define the tag with the following properties:
    - Section: "Global Output"
    - Address: "A1.3"
    - Date type: "Bool"
    - PLC tag table: "Tags GRAPH_Sequence".
    - Replace the text <Limited_pulse_time> with quotation marks.
    - Select the global data block "Global_DB".
    - Press the "Return key" to confirm and select the tag "Recipe_element_orange_juice_concentrate".
13. Save the project.

Result

You have successfully programmed the required actions for mixing the beverages. You can use tags to open and close the valves which control the individual ingredient tanks.



Qualifier		Action
L	- Set for limited time	"Valve_water", "Global_DB".Recipe_element_water
L	- Set for limited time	"Valve_AJC", "Global_DB".Recipe_element_apple_jui
L	- Set for limited time	"Valve_OJC", "Global_DB".Recipe_element_orange_ju

4.3.4.5 Step S2 Fill recipe ingredients - Programming a transition

Introduction

In the following section you will program the transition conditions for switching from step "S2 Fill recipe ingredients" to step "S3 Mixer". In addition to the already programmed multistep transition condition "GRAPH_Group_Fault", you will program three other conditions which are only valid inside the step "S2 Fill recipe ingredients".

You can use these step-specific transition conditions to define the states of the individual valves. If one of the valves is still open it is not permitted to switch to the next step "S3 Mixer". The transition conditions are implemented with NC contacts.

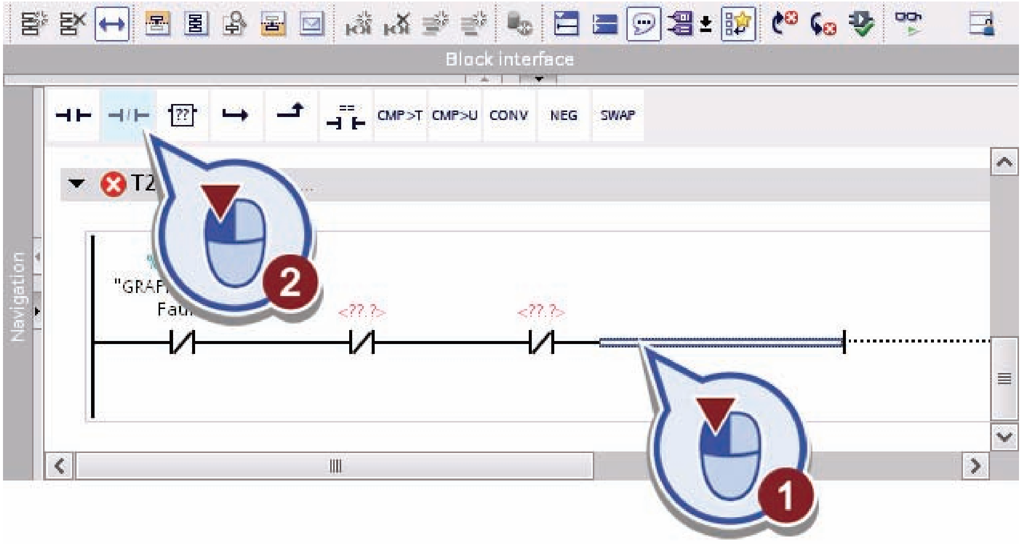
Requirement

You have opened the step "S2 Fill recipe ingredients" and programmed the actions.

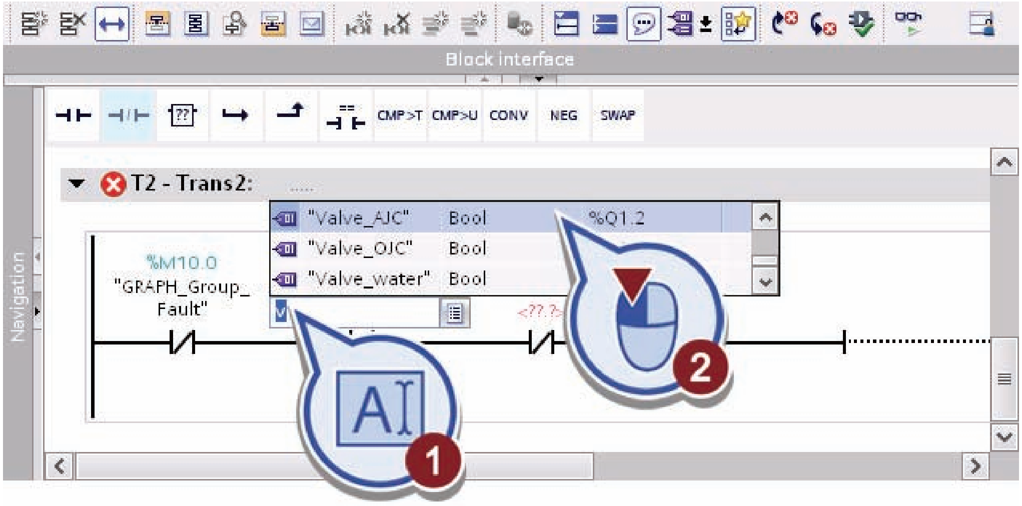
Procedure

To program the transition conditions, proceed as follows:

1. Insert three NC contacts at "T2 – Trans2".



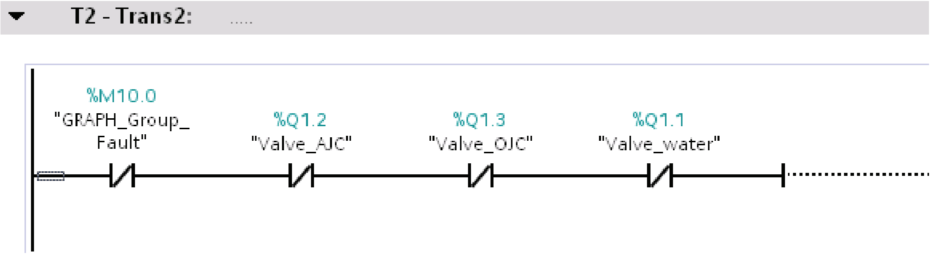
2. Double-click the operand placeholder, start to enter the tag name and select the tag "Valve_AJC" from the drop-down list box.



3. Repeat step 2 for the two other NC contacts and select the tags "Valve_OJC" and "Valve_water" for each of these contacts.
4. Save the project.

Result

You have successfully created three step-specific transition conditions "Valve_AJC", "Valve_OJC" and "Valve_water". The process does **not** switch to the next step "S3 Mixer" as long as one of these conditions has the signal state "1", i.e. as long as a valve is open or a group error occurred.



4.3.4.6 Step S3 Mixer - Programming actions and transitions

Introduction

In the following section you will program an action and a transition condition for the operation of the mixer in the step "S3 Mixer".

The mixer is only permitted to go into operation when all valves are closed and no group errors exist.

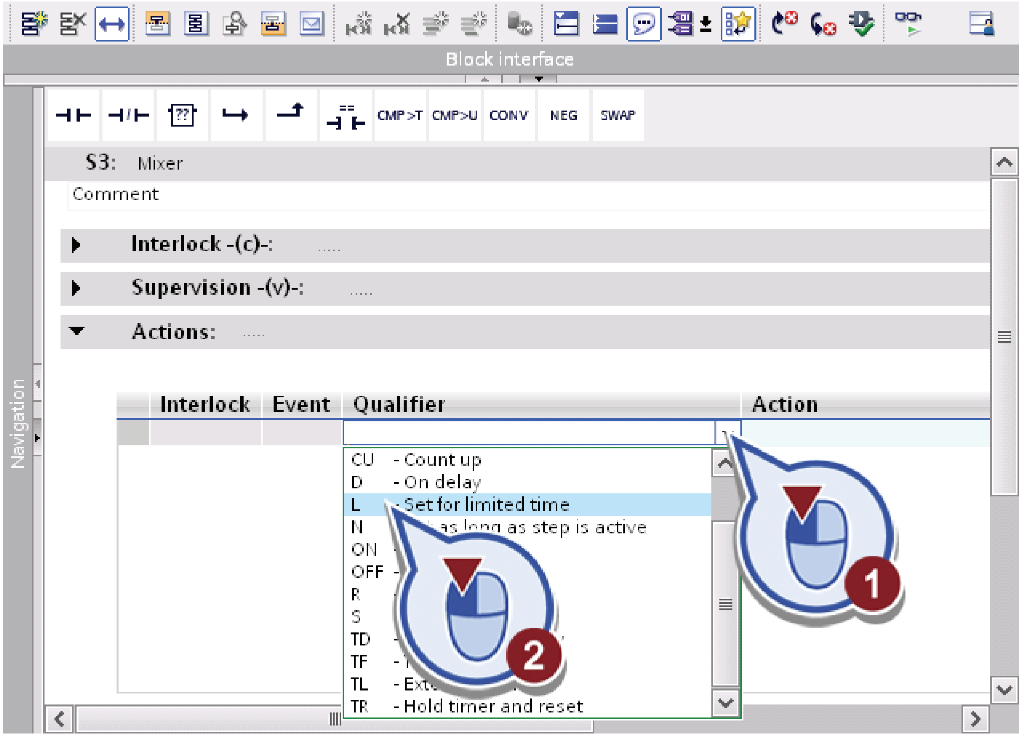
Requirement

You have opened the step "S3 Mixer".

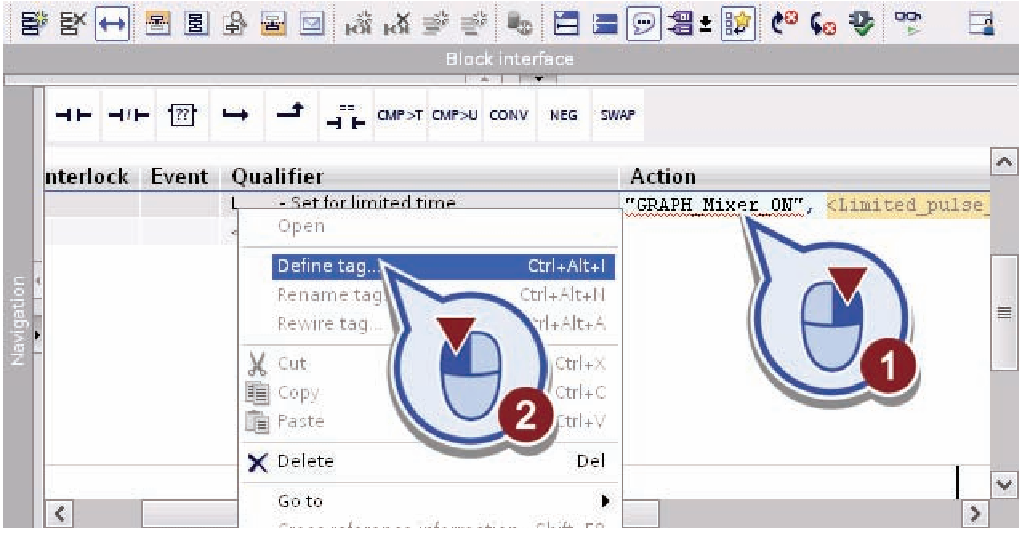
Procedure

To program the action and the transition condition, proceed as follows:

1. In the "Actions" dialog box, select the qualifier "L - Set for limited time".

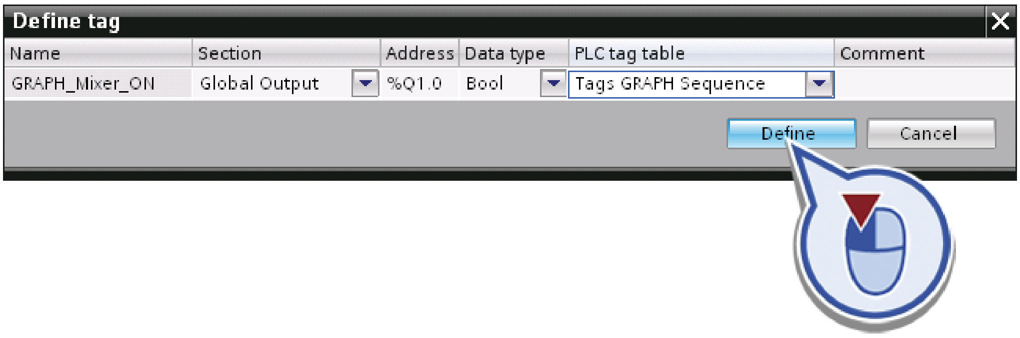


2. Replace the text "<Variable_name_or_address>" with "GRAPH_Mixer_ON".
3. Right-click the text "GRAPH_Mixer_ON" and select "Define tag" from the shortcut menu.



4. Define the tag with the following properties:
   - Section: "Global Output"
   - Address: "Q1.0"
   - Date type: "Bool"
   - PLC tag table: "Tags GRAPH Sequence"

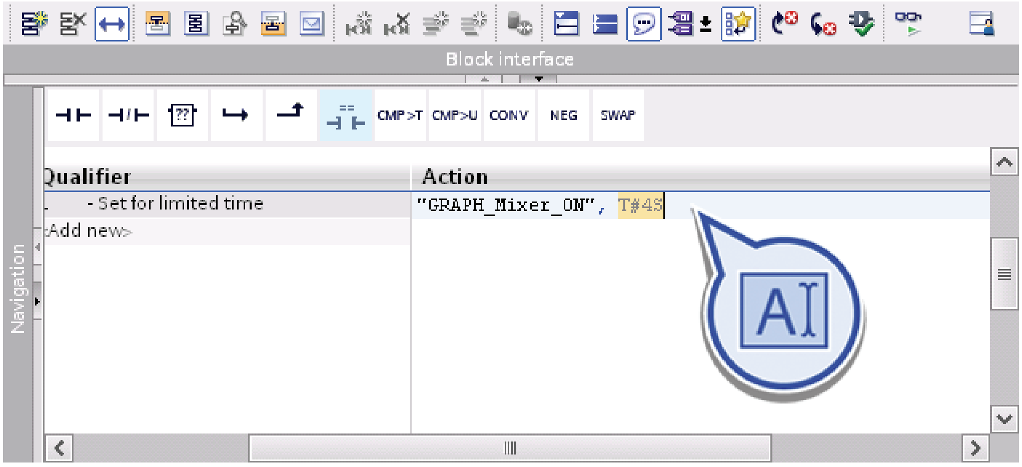
   Confirm the dialog by clicking "Define".



Define tag

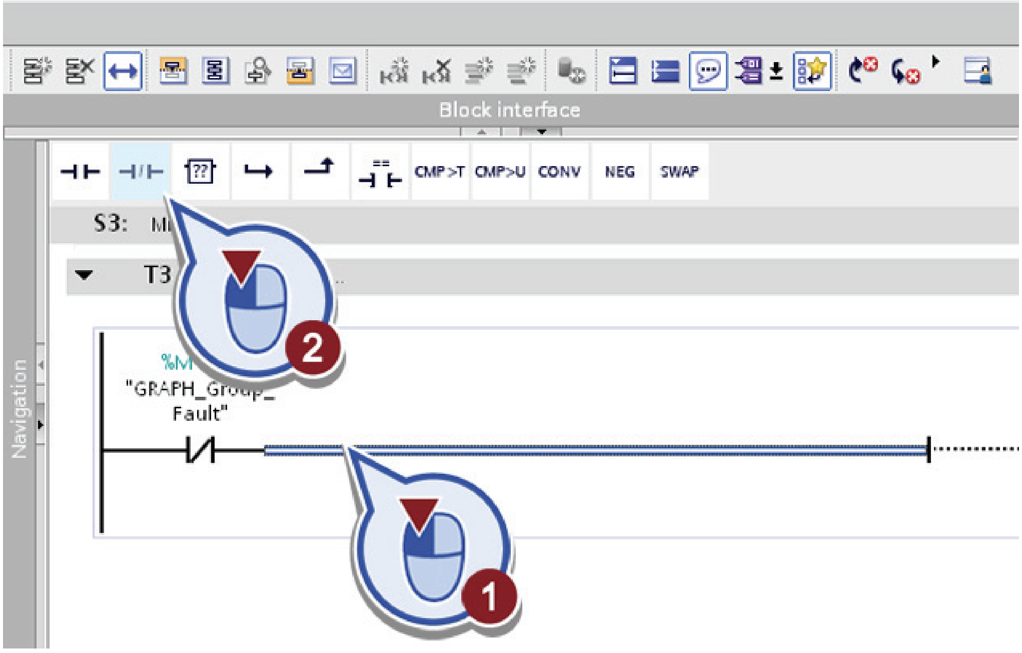
Name	Section	Address	Data type	PLC tag table	Comment
GRAPH_Mixer_ON	Global Output	%Q1.0	Bool	Tags GRAPH Sequence	

5. Replace the text "<Limited_pulse_time>" with the time specification "T#4S" (4 seconds).

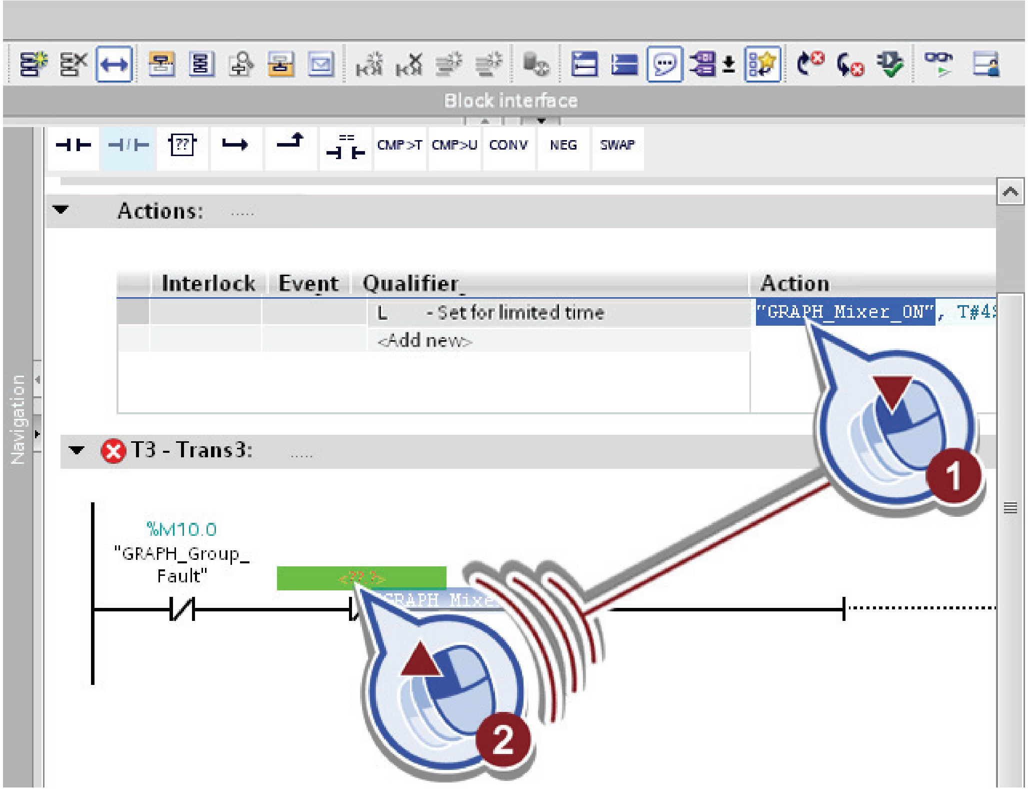


Qualifier	Action
- Set for limited time	"GRAPH_Mixer_ON", T#4S
<Add new>	

6. Insert an NC contact at "T3 – Trans3".



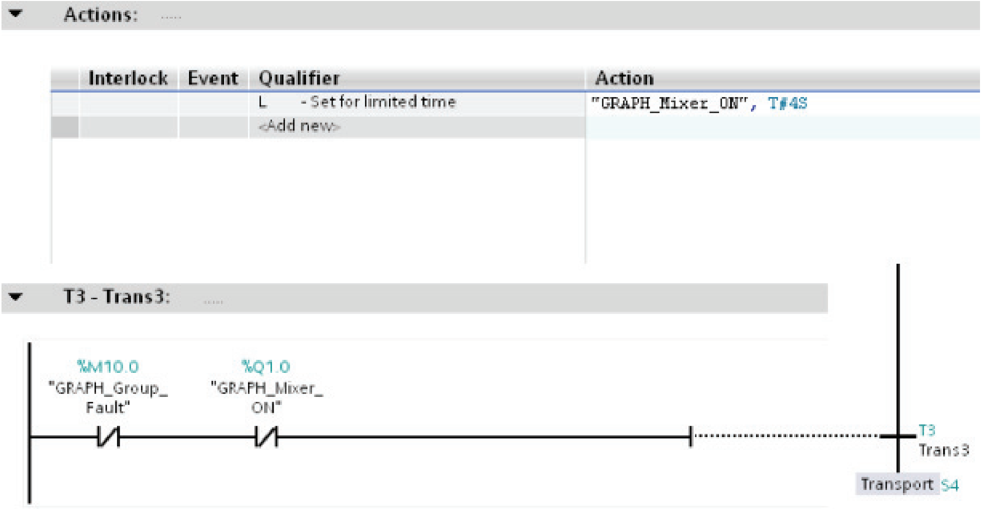
7. Select the tag "GRAPH_Mixer_ON" in the "Action" field and drag it to the operand placeholder of the NC contact.



8. Save the project.

Result

You have successfully inserted the action ""GRAPH_Mixer_ON", T#4S" and the transition condition "GRAPH_Mixer_ON".



Actions:

Interlock	Event	Qualifier	Action
		L - Set for limited time	"GRAPH_Mixer_ON", T#4S
		<Add new>	

T3 - Trans3:

As soon as the step "S3 Mixer" is enabled, the tag "GRAPH_Mixer_ON" is set to signal state "1" for 4 seconds. The mixer is in operation for 4 seconds. After 4 seconds the timer has elapsed and the tag is set to the signal state "0". The transition condition is only satisfied now and the sequencer can switch to the next step "S4 Transport Filling".

4.3.4.7 Step S4 Transport Filling - Programming actions and transitions

Introduction

In the following section you will program the conveyor belt. The conveyor belt transports the empty bottles to the filling system, stops there for the filling process and transports the filled bottles to the labeling station. In the step "S4 Transport Filling", you program the transport section to the filling station.

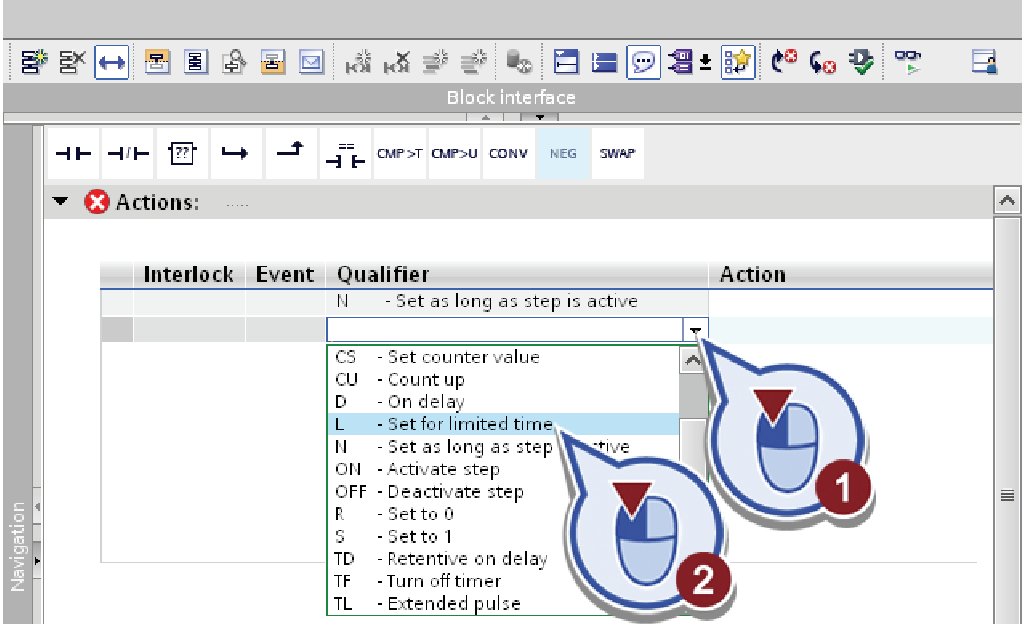
Requirement

You have opened the step "S4 Transport Filling".

Procedure

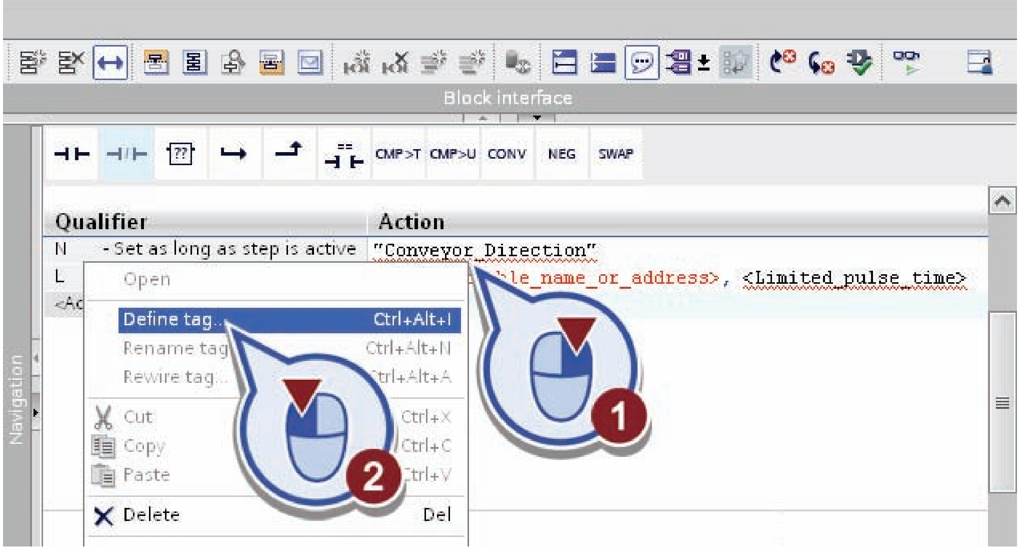
To program the conveyor belt, proceed as follows:

1. Create two qualifiers in the "Actions" dialog box:
   - "N - Set as long as step is active"
   - "L - Set for limited time"



Interlock	Event	Qualifier	Action
		N - Set as long as step is active	

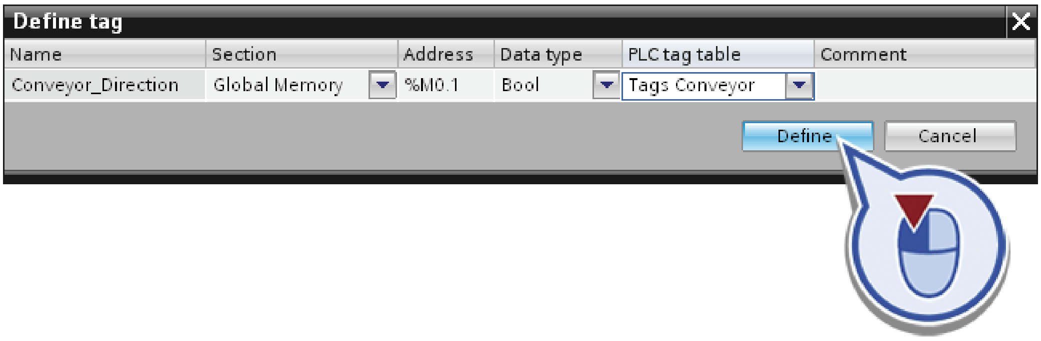
2. Enter the text "Conveyor_Direction" in the "Action" column.
3. Right-click the text "Conveyor_Direction" and select "Define tag" from the shortcut menu.



Qualifier	Action
N - Set as long as step is active	"Conveyor_Direction"
L	le_name_or_address>, <Limited_pulse_time>
<Ac	

4. Define the tag with the following properties:
   - Section: "Global Memory"
   - Address: "M0.1"
   - Date type: "Bool"
   - PLC tag table: "Tags Conveyor"

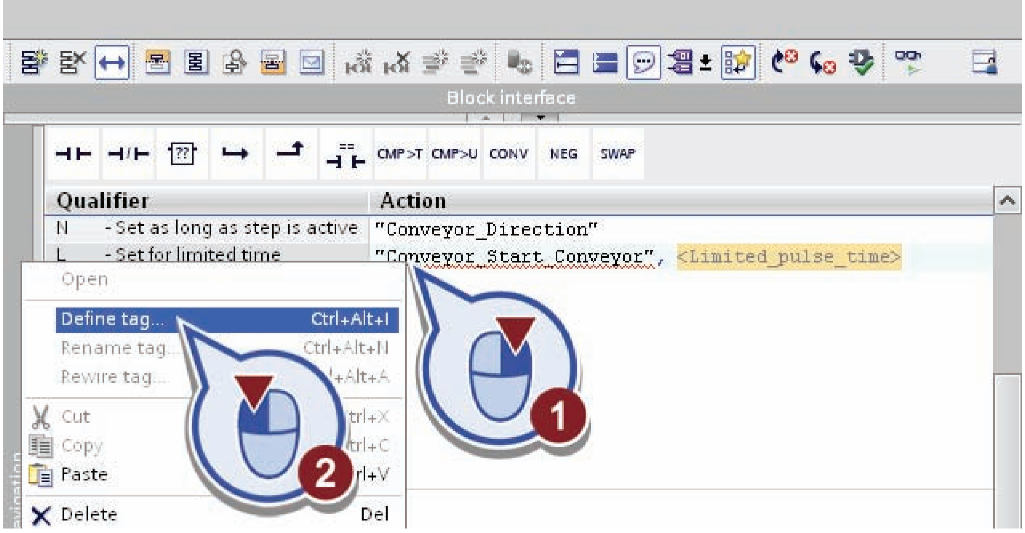
   Confirm the dialog by clicking "Define".



Define tag

Name	Section	Address	Data type	PLC tag table	Comment
Conveyor_Direction	Global Memory	%M0.1	Bool	Tags Conveyor	

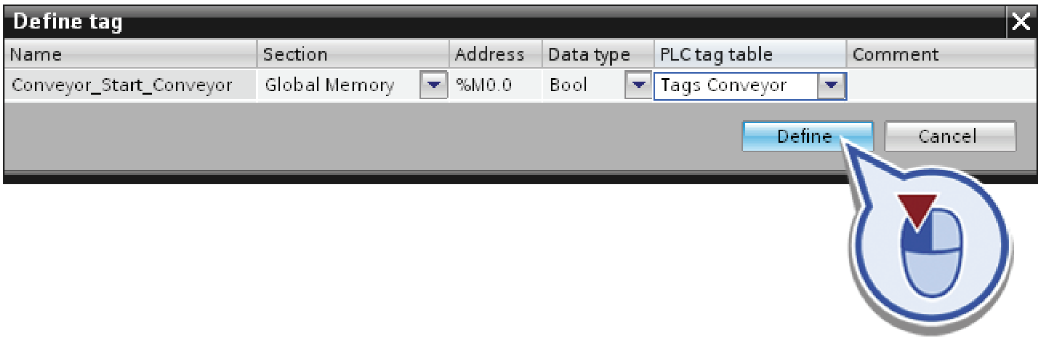
5. In the second line, replace the text <Variable_name_or_address> with "Conveyor_Start_Conveyor".
6. Right-click the text "Conveyor_Start_Conveyor" and select "Define tag" from the shortcut menu.



Qualifier	Action
N - Set as long as step is active	"Conveyor_Direction"
L - Set for limited time	"Conveyor_Start_Conveyor", <Limited_pulse_time>

7. Define the tag with the following properties:
   – Section: "Global Memory"
   – Address: "M0.0"
   – Date type: "Bool"
   – PLC tag table: "Tags Conveyor"

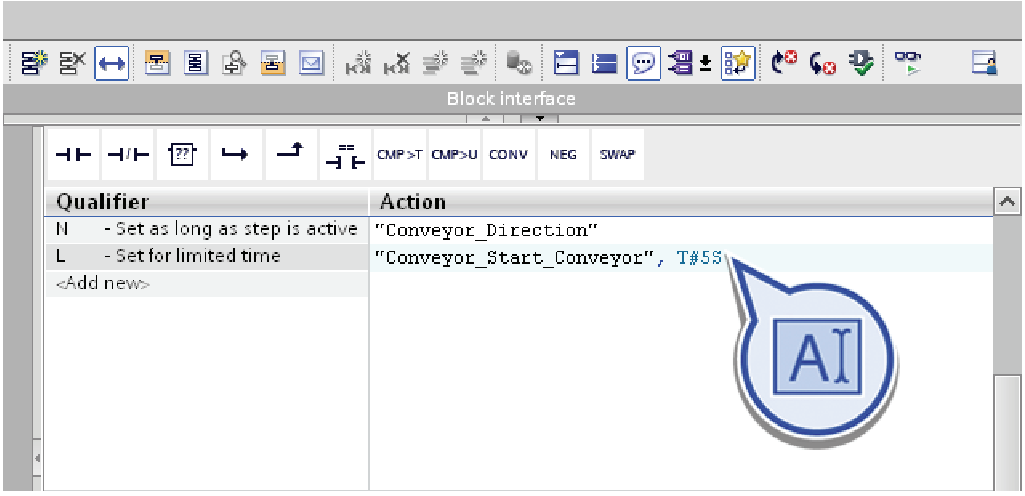
   Confirm the dialog by clicking "Define".



Define tag

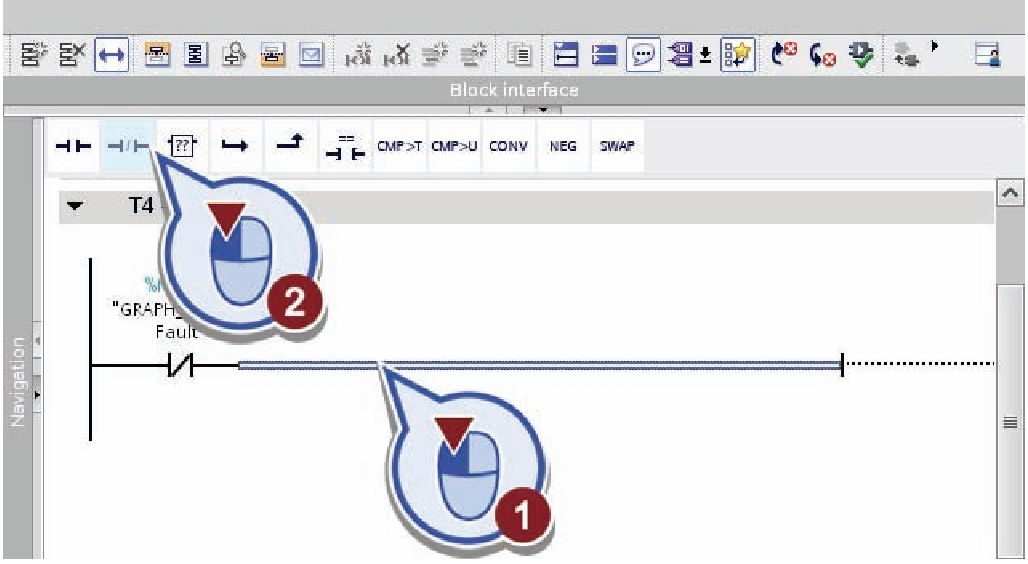
Name	Section	Address	Data type	PLC tag table	Comment
Conveyor_Start_Conveyor	Global Memory	%M0.0	Bool	Tags Conveyor	

8. In the second line, replace the text <Limited_pulse_time> with the time specification "T#5S" (5 seconds).

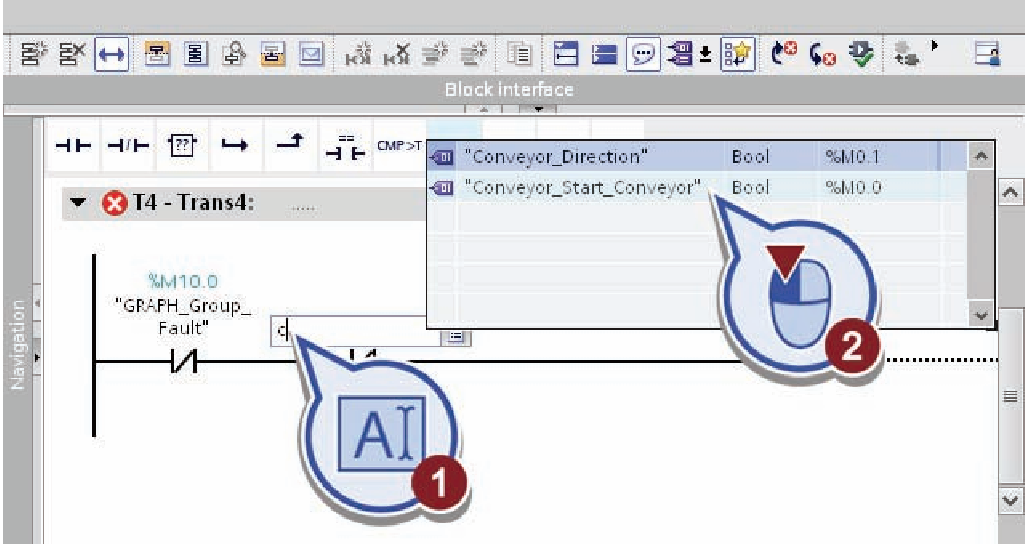


Qualifier	Action
N - Set as long as step is active	"Conveyor_Direction"
L - Set for limited time	"Conveyor_Start_Conveyor", T#5S
<Add new>	

9. Insert an NC contact at "T4 – Trans4".



10. Double-click the operand placeholder, start to enter the name of the tag "Conveyor_Start_Conveyor", and select the tag from the drop-down list box.

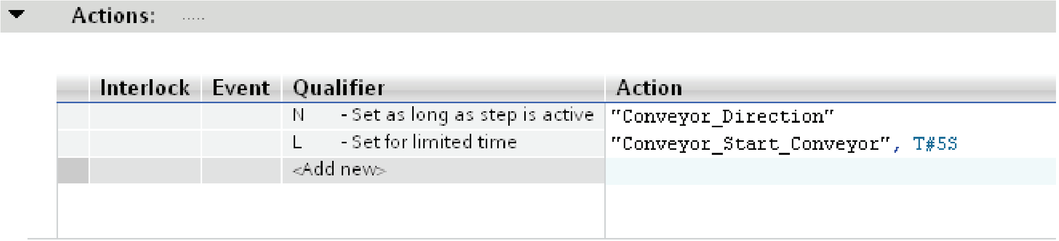


"Conveyor_Direction"	Bool	%M0.1
"Conveyor_Start_Conveyor"	Bool	%M0.0

11. Save the project.

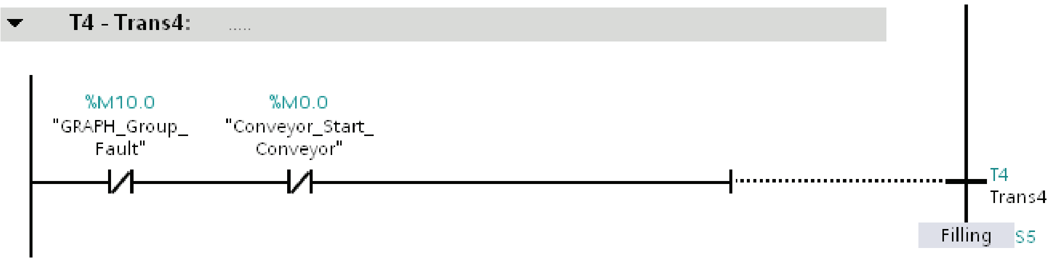
Result

You have successfully inserted the actions "Conveyor_Direction" and "Conveyor_Start_Conveyor" ,T#5S, and the transition condition "Conveyor_Start_Conveyor". As soon as the step "S4 Transport Filling" is enabled, the timer "T#5S" starts to count down. The timer is set to 5 seconds, since the conveyor belt requires this period to transport an empty bottle to the filling station. After 5 seconds the empty bottle is in position and the timer has the signal state "0". The transition condition "Conveyor_Start_Conveyor" is only satisfied now and the sequencer can switch to the next step "S5 Filling".



Actions: .....

Interlock	Event	Qualifier	Action
		N - Set as long as step is active	"Conveyor_Direction"
		L - Set for limited time	"Conveyor_Start_Conveyor", T#5S
		<Add new>	



T4 - Trans4: .....

4.3.4.8 Step S5 Filling - Programming actions and transitions

Introduction

In the following section you will program the filling process. The empty bottle is positioned under the filling station and the valve opens for 3 seconds. To detect the number of already filled bottles, you program a counter which increments by one each time this step is completed.

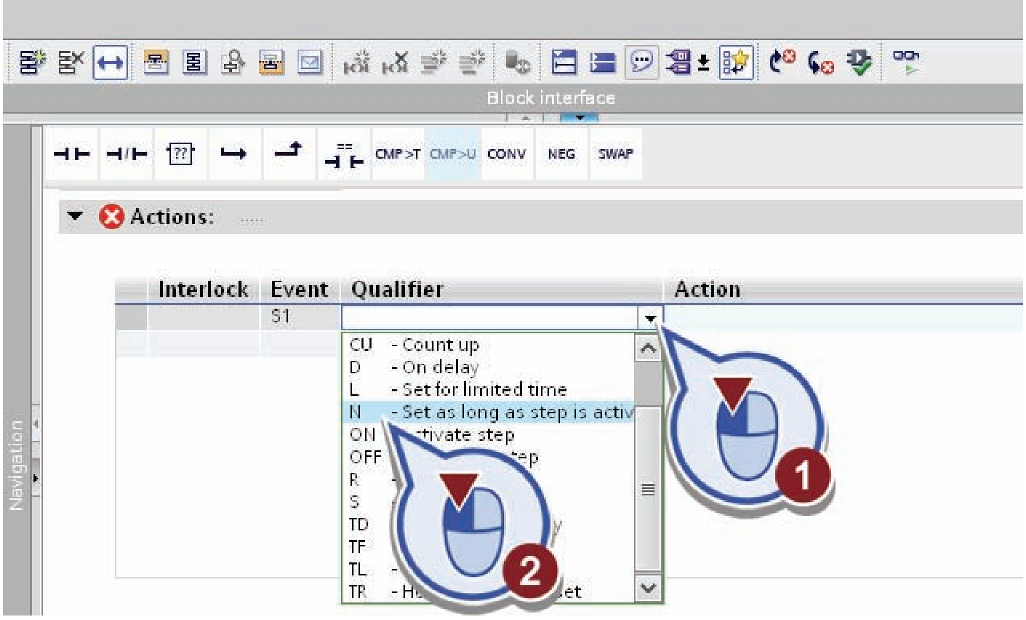
Requirement

You have opened the step "S5 Filling".

Procedure

To program the filling process, proceed as follows:

1. In the "Actions" dialog box, select the event "S1 – Incoming step" and the qualifier "N – Set as long as step is active".

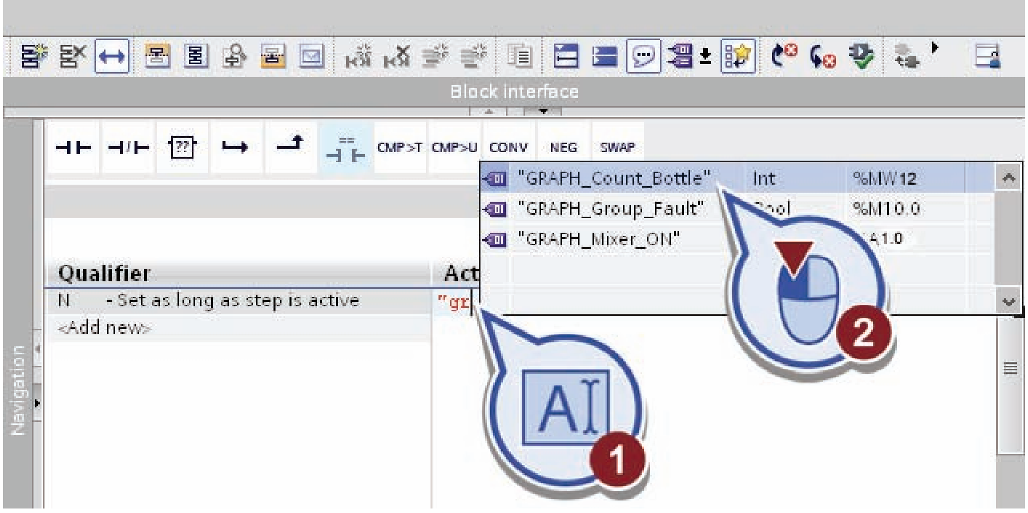


Actions:

Interlock	Event	Qualifier	Action
	S1		

CU - Count up
D - On delay
L - Set for limited time
N - Set as long as step is activ
ON ...tivate step
OFF
R
S
TD
TF
TL
TR - H...

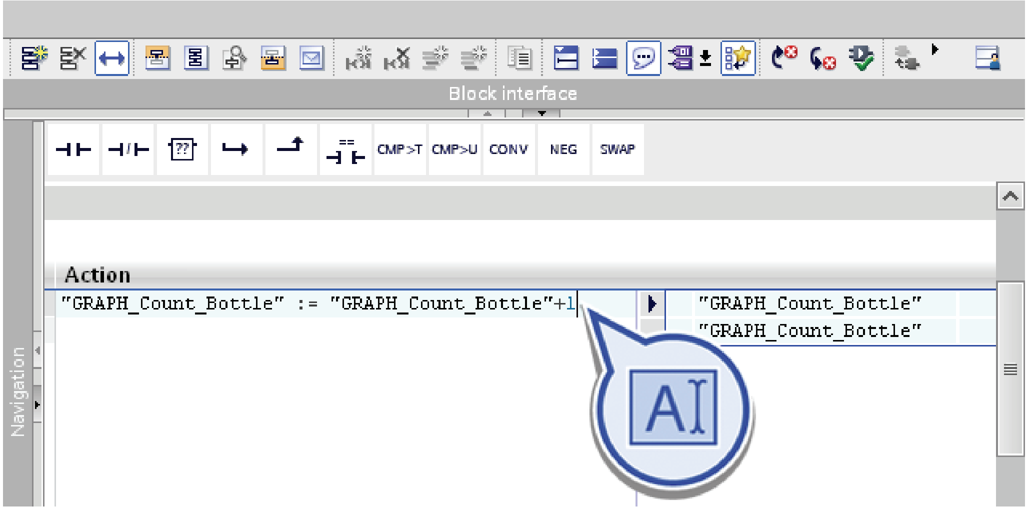
2. Enter the tag name "GRAPH_Count_Bottle" in the "Action" column.



Qualifier	Act
N - Set as long as step is active	"gr
<Add new>	

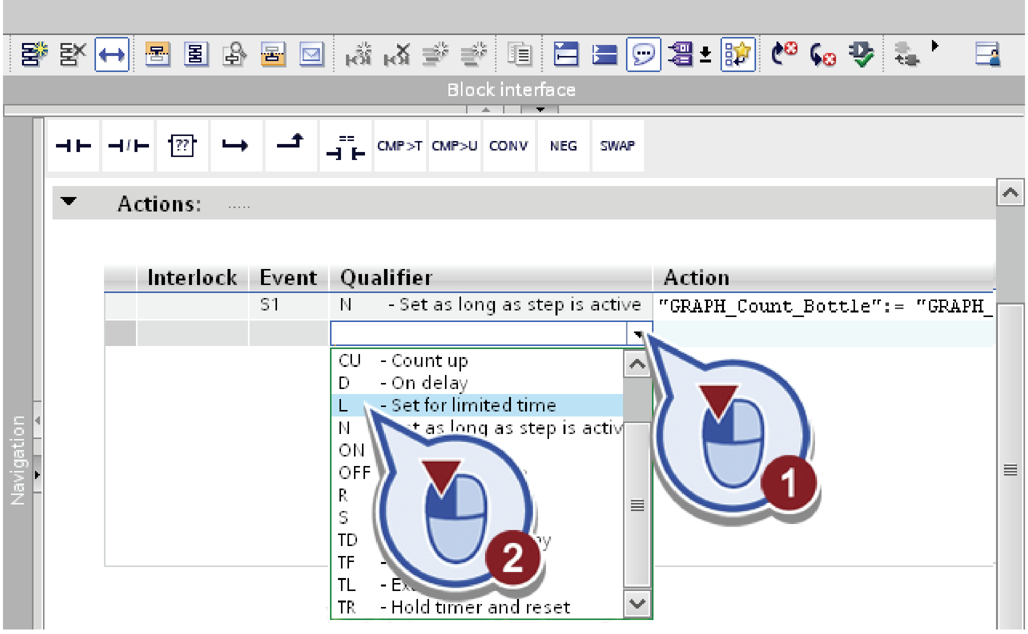
"GRAPH_Count_Bottle"	Int	%MW12
"GRAPH_Group_Fault"	Bool	%M10.0
"GRAPH_Mixer_ON"		%A1.0

3. Insert the addition ":= "GRAPH_Count_Bottle"+1" after the tag.



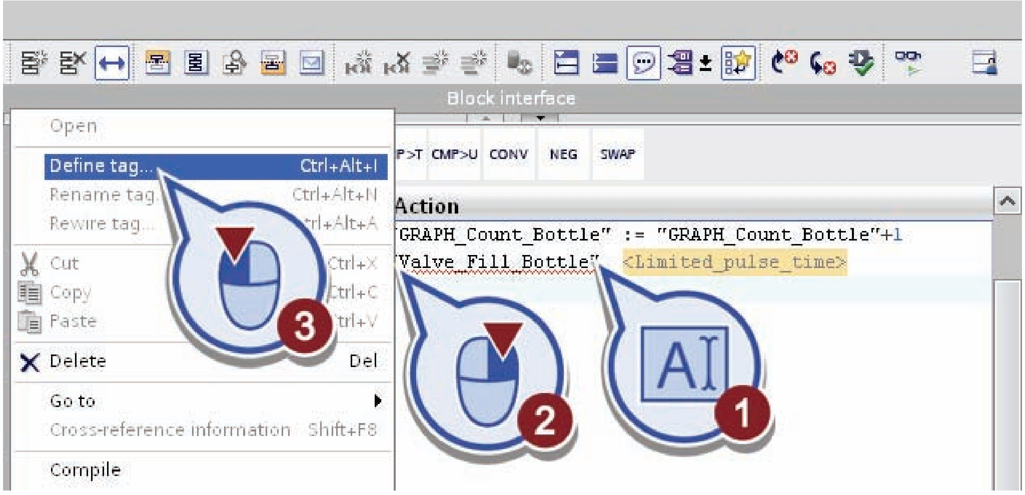
"GRAPH_Count_Bottle" := "GRAPH_Count_Bottle"+1

4. In the second line, select the qualifier "L - Set for limited time".



Interlock	Event	Qualifier	Action
	S1	N - Set as long as step is active	"GRAPH_Count_Bottle":= "GRAPH_

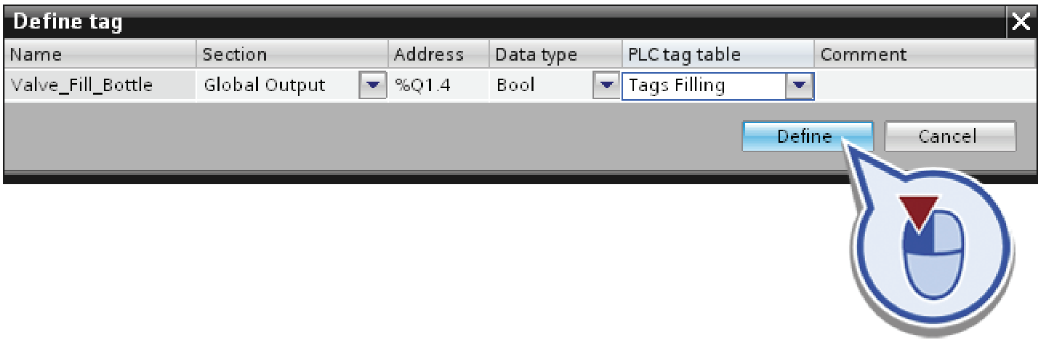
5. Replace the text <Variable_name_or_address> with "Valve_Fill_Bottle". Right-click the text "Valve_Fill_Bottle" and select "Define tag" from the shortcut menu.



```
"GRAPH_Count_Bottle" := "GRAPH_Count_Bottle"+1
"Valve_Fill_Bottle"  <Limited_pulse_time>
```

6. Define the tag with the following properties:
   - Section: "Global Output"
   - Address: "Q1.4"
   - Date type: "Bool"
   - PLC tag table: "Tags Filling"

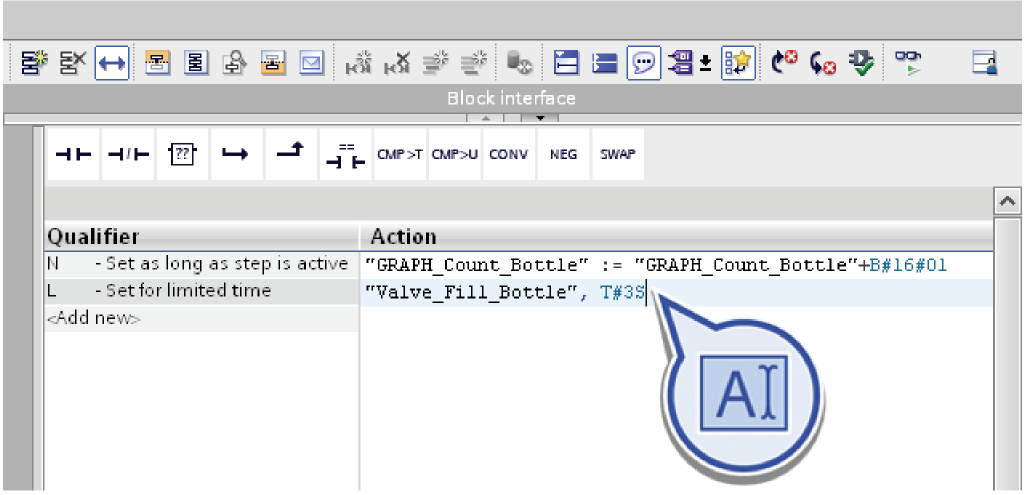
   Confirm the dialog by clicking "Define".



Define tag

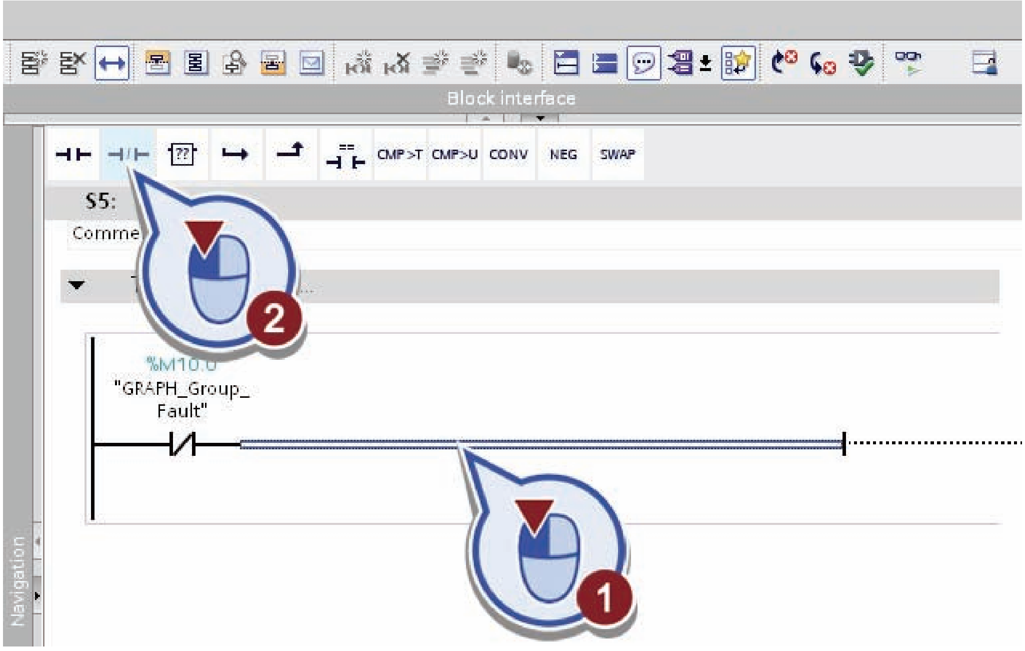
Name	Section	Address	Data type	PLC tag table	Comment
Valve_Fill_Bottle	Global Output	%Q1.4	Bool	Tags Filling	

7. Replace the text <Limited_pulse_time> with "T#3S".

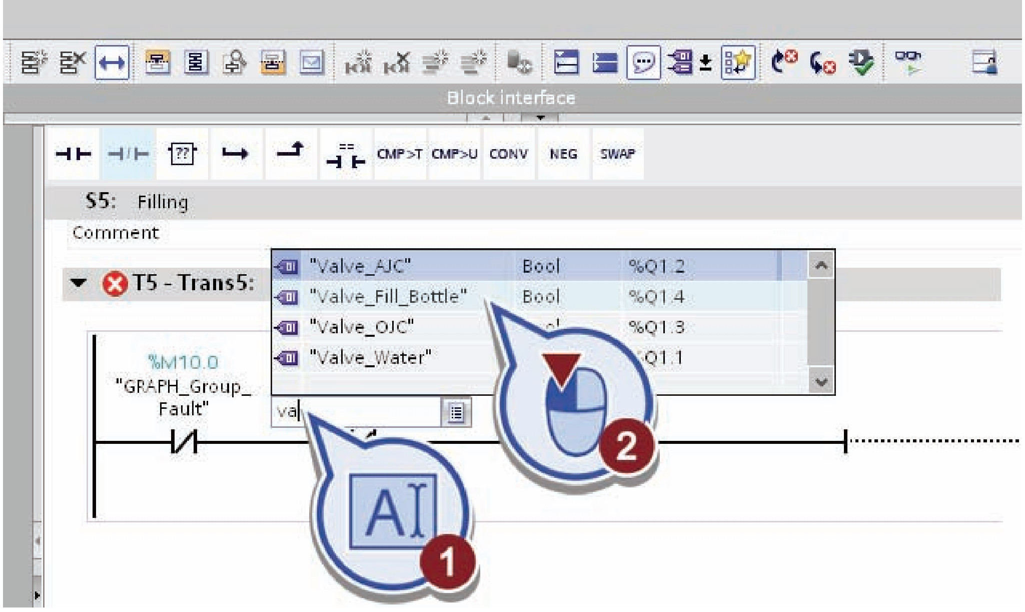


Qualifier	Action
N - Set as long as step is active	"GRAPH_Count_Bottle" := "GRAPH_Count_Bottle"+B#16#01
L - Set for limited time	"Valve_Fill_Bottle", T#3S
<Add new>	

8. Insert an NC contact at "T5 – Trans5".



9. Double-click the operand placeholder, start to enter the name of the tag "Valve_Fill_Bottle", and select the tag from the drop-down list box.

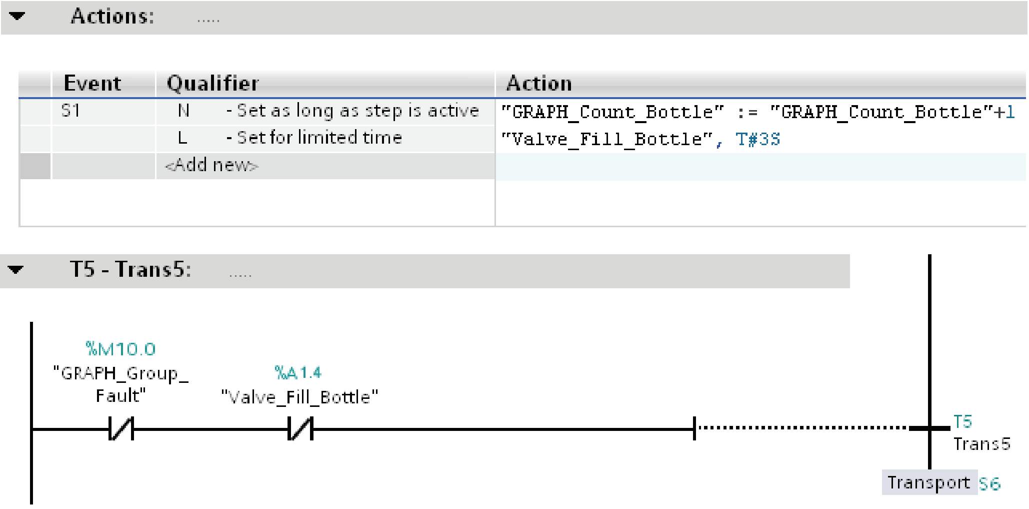


"Valve_AJC"	Bool	%Q1.2
"Valve_Fill_Bottle"	Bool	%Q1.4
"Valve_OJC"		%Q1.3
"Valve_Water"		Q1.1

10.Save the project.

Result

You have successfully inserted the actions "GRAPH_Count_Bottle" and "Valve_Fill_Bottle, T#3S" and the transition condition "Valve_Fill_Bottle, T#3S". As soon as the step "S5 Filling" is enabled, the empty bottle is filled for 3 seconds with the beverage mix. As soon as a bottle has been filled and transported to the labeling station, the counter is incremented by one. You need the counter to determine when the maximum number of 10 bottles has been filled. After 3 seconds the bottle has been filled and the timer has the signal state "0". The transition condition "Valve_Fill_Bottle" is only satisfied now and the sequencer can switch to the next step "S6 Transport Labeling".



Actions: .....

Event	Qualifier	Action
S1	N - Set as long as step is active	"GRAPH_Count_Bottle" := "GRAPH_Count_Bottle"+1
	L - Set for limited time	"Valve_Fill_Bottle", T#3S
	<Add new>	

4.3.4.9 Step S6 Transport Labeling - Actions and transitions

Introduction

In the following section you will program the conveyor belt to transport the bottles to the labeling station after the filling process. Since the conveyor belt is activated for the same period (5 seconds) and the same direction as in step "S5 Transport Filling", you can copy the actions and transition conditions from this step.

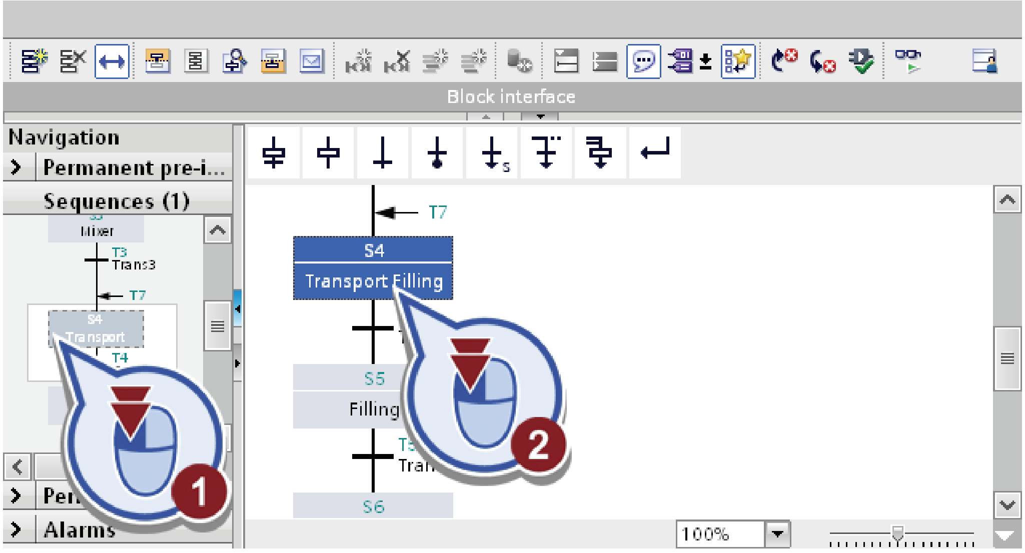
Requirement

You have created the step "S6 Transport Filling" and the actions and transitions of the step "S4 Transport Filling".

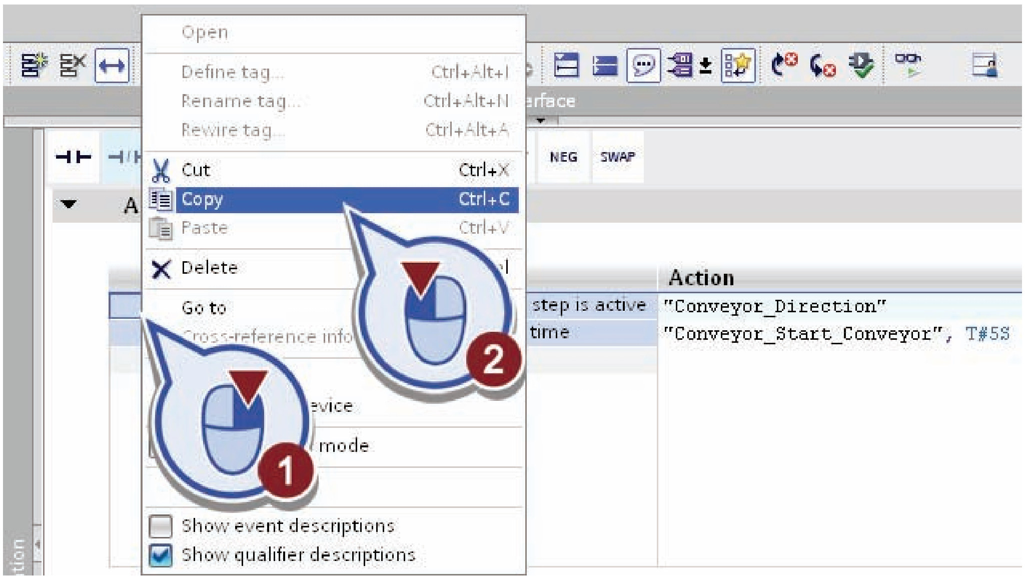
Procedure

To program the conveyor belt, proceed as follows:

1. Double-click on the step "S4 Transport Filling" in the project tree and in the work area.

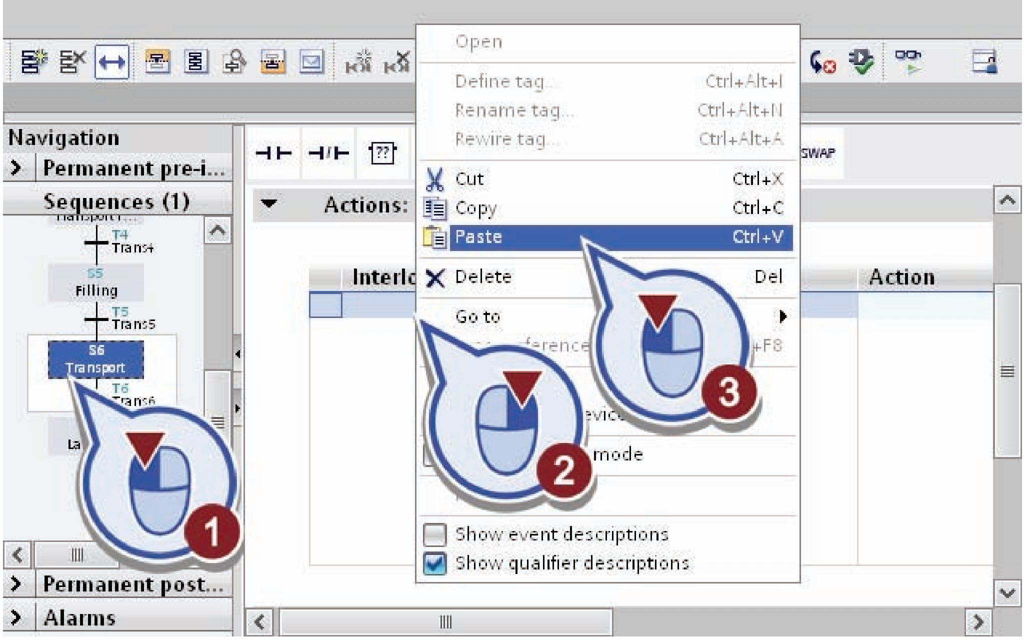


2. In the "Actions" dialog box, select the first cells of both lines by holding down the <Shift> key. Right-click the selected cells and select "Copy" from the shortcut menu.

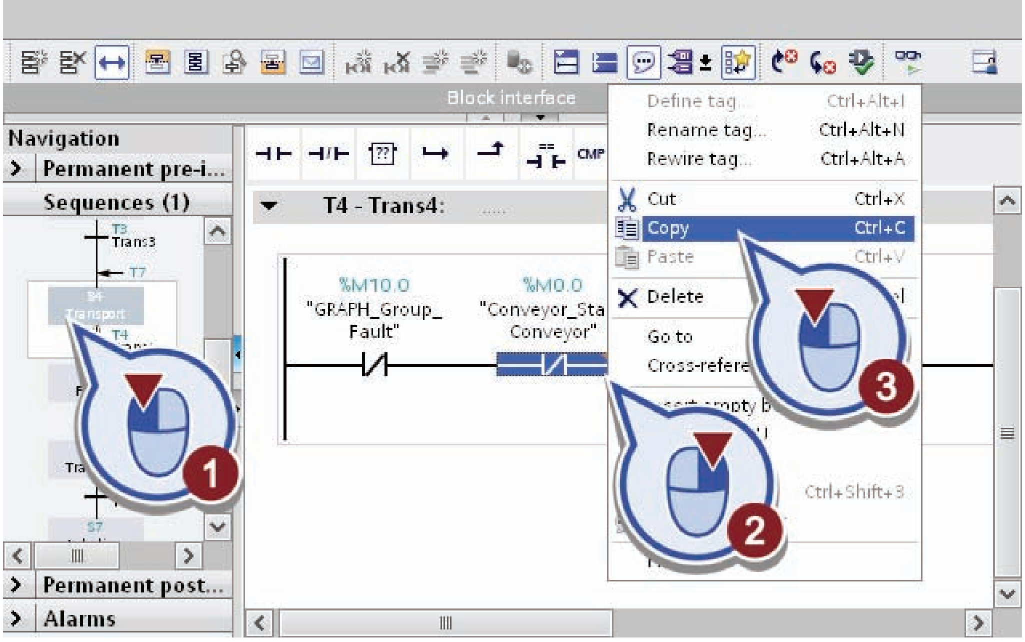


"Conveyor_Direction"
"Conveyor_Start_Conveyor", T#5S

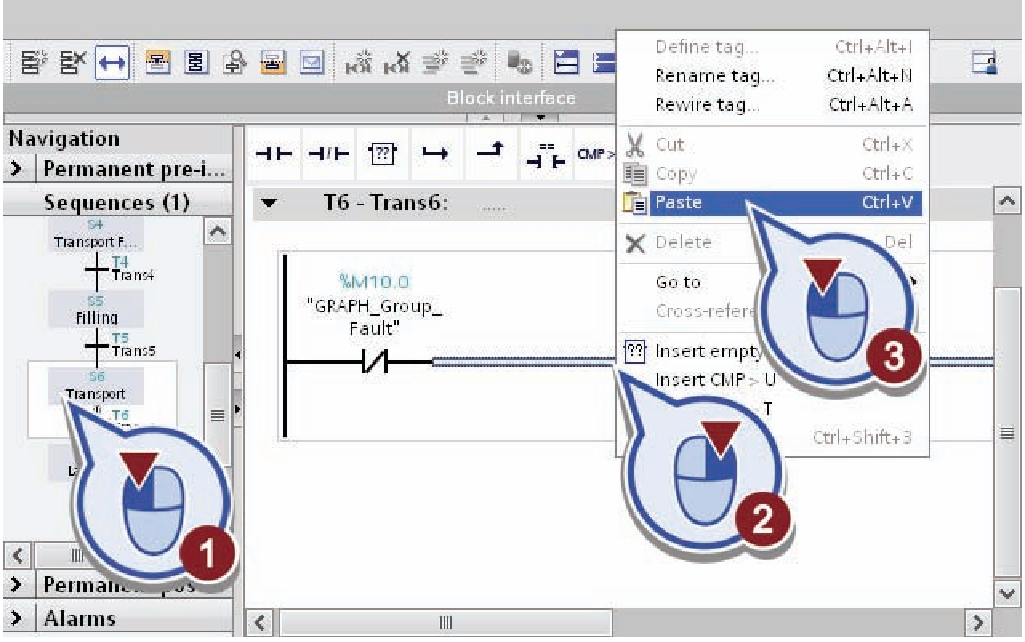
3. Click the step "S6 Transport Labeling" and insert the copied actions by right-clicking in the first line of the "Actions" dialog box and selecting "Paste" from the shortcut menu.



4. Click the step "S4 Transport Filling" and copy the transition condition "Conveyor_Start_Conveyor" at "T4 – Trans4".



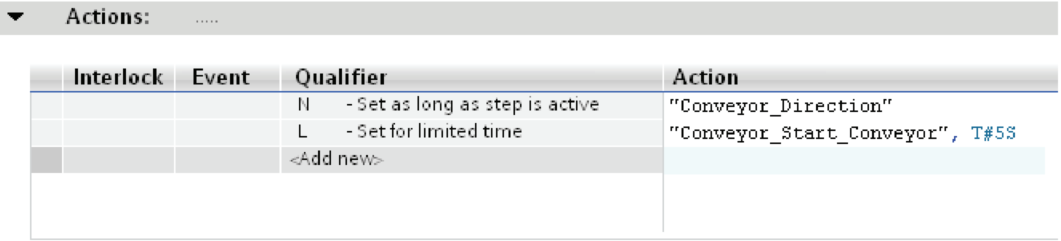
5. Click the step "S6 Transport Labeling" and insert the copied transition condition at "T6 - Trans6".



6. Save the project.

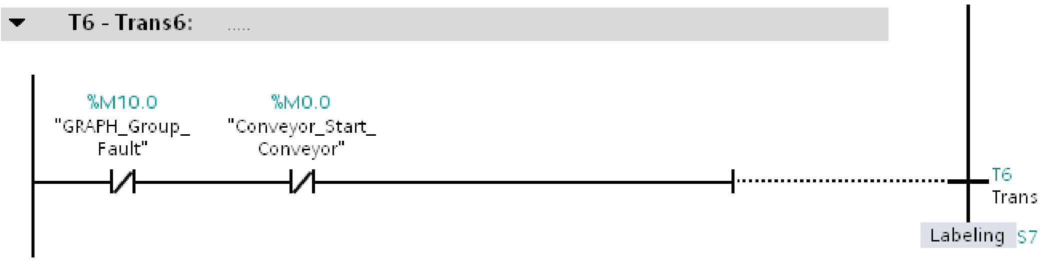
Result

You have successfully copied the actions "Conveyor_Direction" and "Conveyor_Start_Conveyor", T#5S and the transition condition "Conveyor_Start_Conveyor" from the step "S4 Transport Filling" and inserted them in the step "S6 Transport Labeling".



Actions: .....

Interlock	Event	Qualifier		Action
		N	- Set as long as step is active	"Conveyor_Direction"
		L	- Set for limited time	"Conveyor_Start_Conveyor", T#5S
		<Add new>		



As in step "S4 Transport Filling", the timer "T#5S"starts to count down as soon as the step "S6 Transport Labeling" is activated. The timer is set to 5 seconds, as the conveyor belt requires this long to transport a filled bottle from the filling station to the labeling station. After 5 seconds, the filled bottle has arrived at the end of the conveyor belt and the timer has the signal state "0". The transition condition "Conveyor_Start_Conveyor" is only satisfied now and the sequencer can switch to the next step "S7 Labeling".

4.3.4.10 Step S7 Labeling - Programming actions

Introduction

In the following section you will program the labeling process. To do this, an output to start the labeling process is set to two seconds.

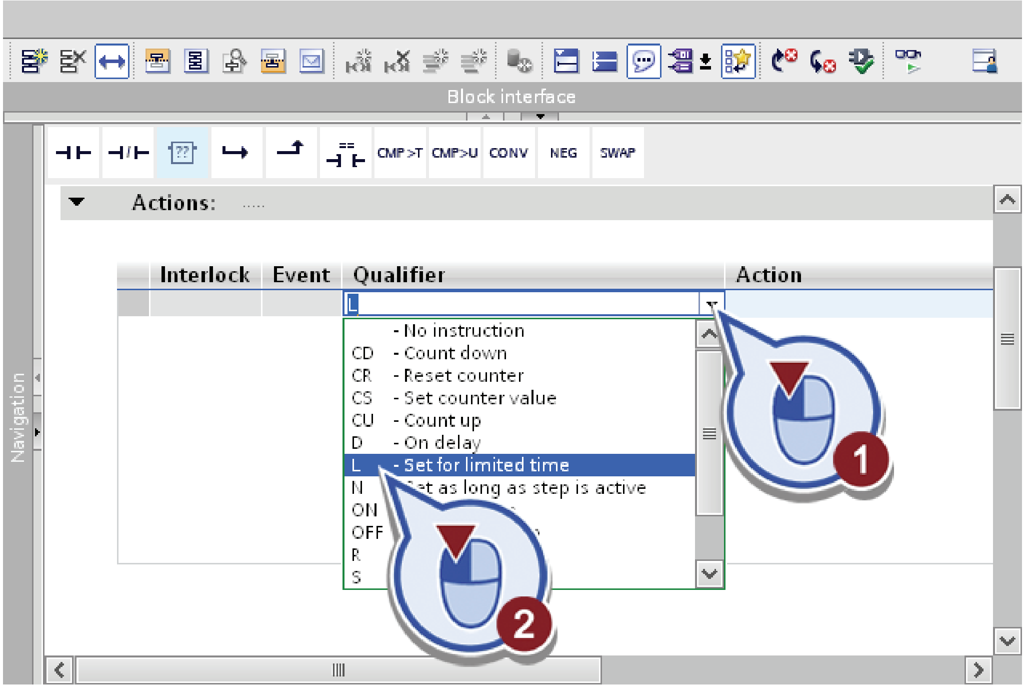
Requirement

You have opened the step "S7 Labeling".

Procedure

To program the labeling process, proceed as follows:

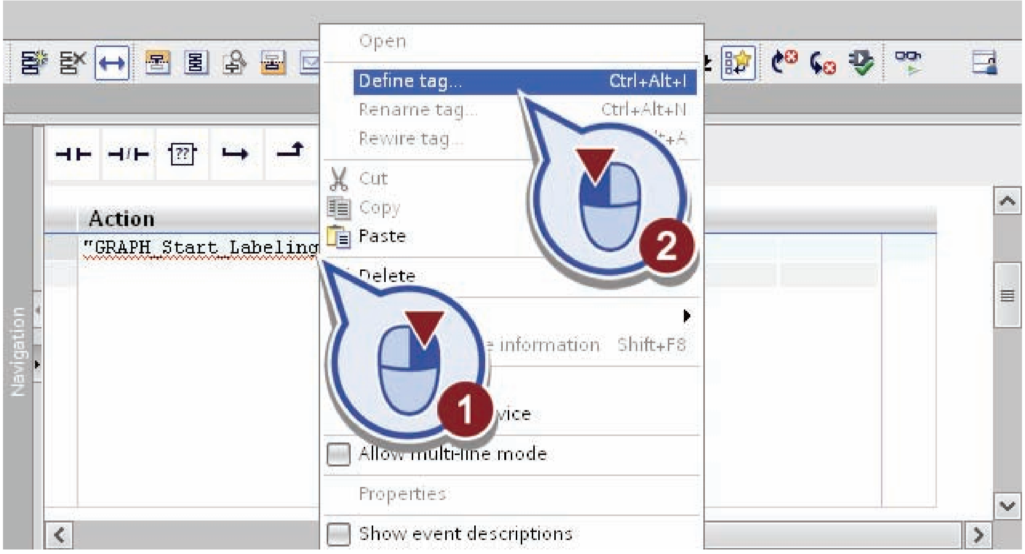
1. Select the qualifier "L - Set for limited time".



Interlock	Event	Qualifier	Action
		L	

	- No instruction
CD	- Count down
CR	- Reset counter
CS	- Set counter value
CU	- Count up
D	- On delay
L	- Set for limited time
N	- Set as long as step is active
ON	
OFF	
R	
S	

2. Replace the text <Variable_name_or_address> with "GRAPH_Start_Labeling". Right-click the text and select "Define tag" from the shortcut menu.



3. Define the tag with the following properties:

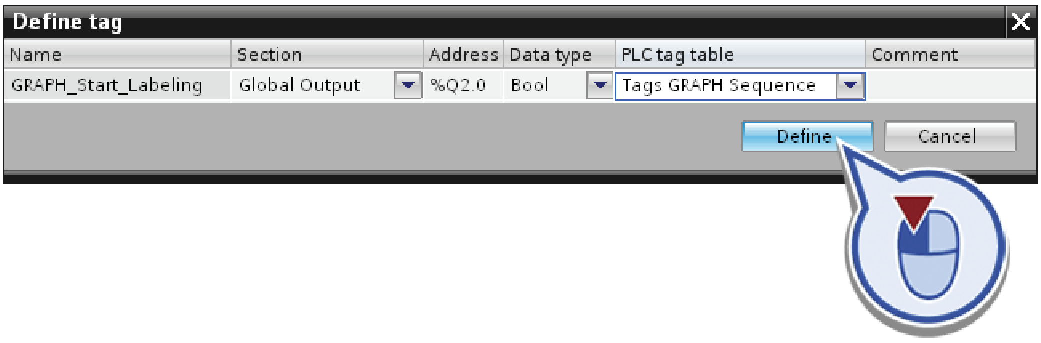
– Section: "Global Output"

– Address: "Q2.0"

– Date type: "Bool"

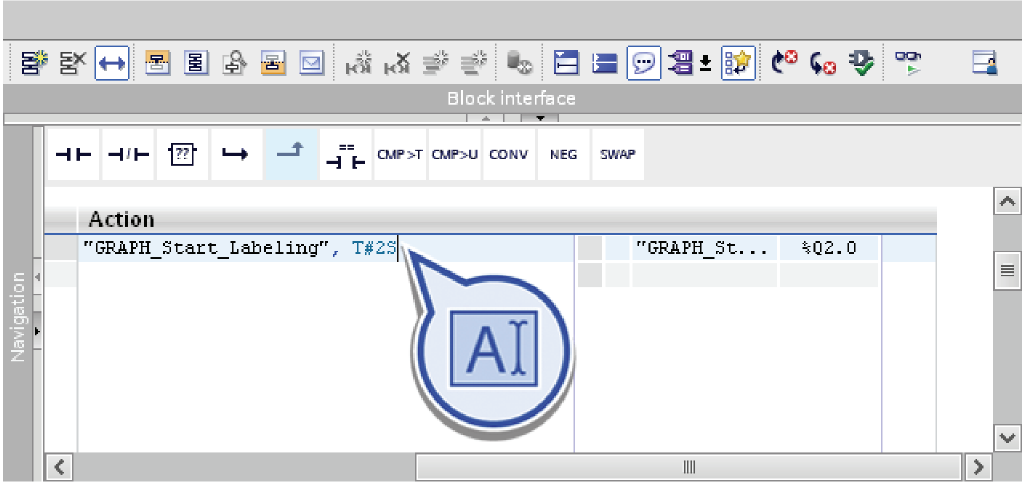
– PLC tag table: "Tags GRAPH Sequence"

Confirm the dialog by clicking "Define".


Define tag

Name	Section	Address	Data type	PLC tag table	Comment
GRAPH_Start_Labeling	Global Output	%Q2.0	Bool	Tags GRAPH Sequence	

4. Replace the text <Limited_pulse_time> with "T#2S".

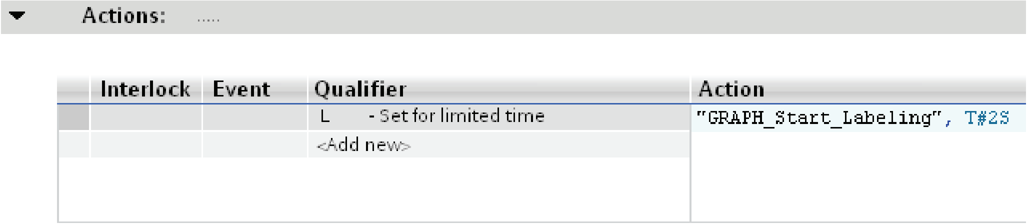


```
"GRAPH_Start_Labeling", T#2S
"GRAPH_St...   %Q2.0
```

5. Save the project.

Result

You have successfully programmed an action to start the labeling process.



Actions:

Interlock	Event	Qualifier	Action
		L - Set for limited time	"GRAPH_Start_Labeling", T#2S
		<Add new>	

4.3.4.11 Step S7 Labeling - Programming transitions

Introduction

In the following section you will program the transition conditions for the step "S7 Labeling". As soon as the labeling process has been completed, there are two alternatives that can run through the sequencer:

- If less than 10 bottles have been filled and labeled, the sequencer should be run through again starting from the step "S4 Transport Filling".
- If 10 bottles have already been filled and labeled, the sequencer should be continued via the alternative branch with the step "S8 Filling Complete".

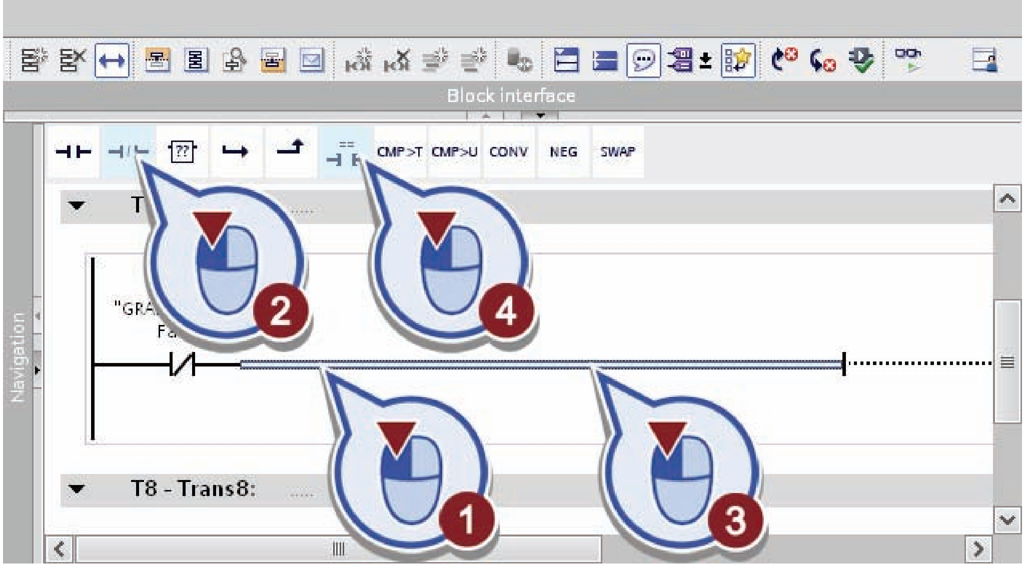
Requirement

You have opened the step "S7 Labeling" and programmed the action "L - Set for limited time".

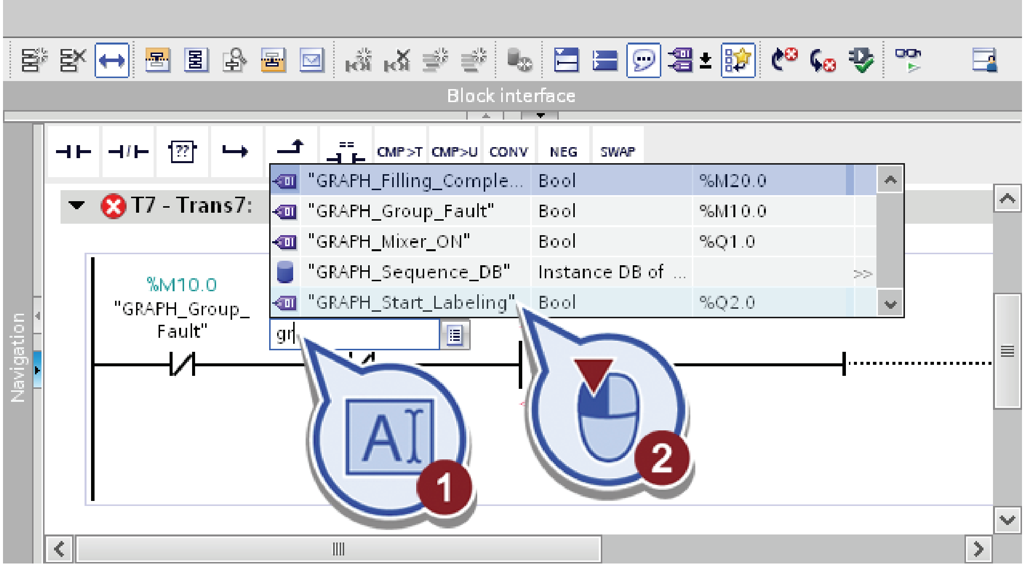
Procedure

To program the transition conditions, proceed as follows:

1. Insert an NC contact and a comparator at "T7 – Trans7".



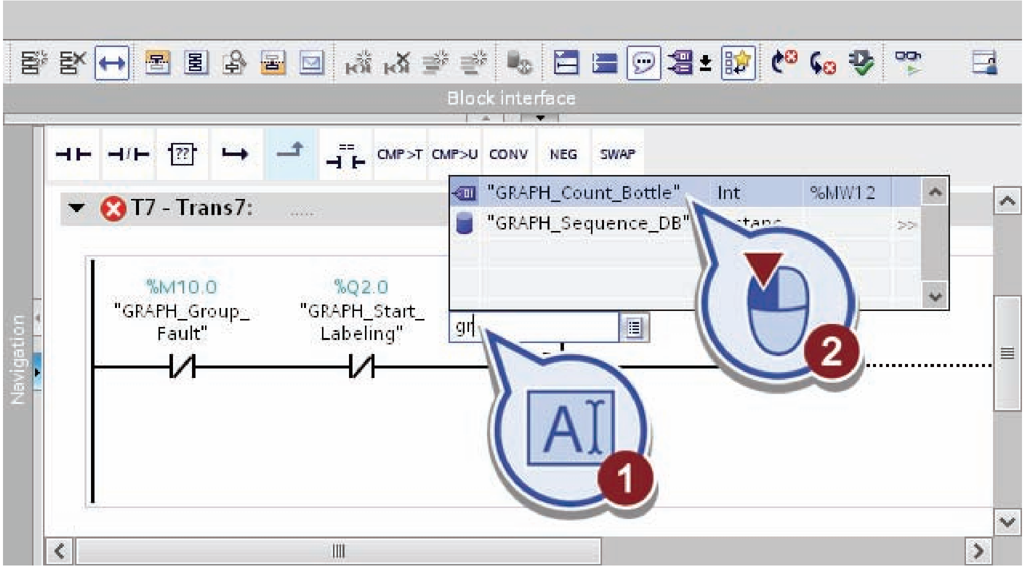
2. Double-click the operand placeholder of the NC contact, start to enter the name of the tag "GRAPH_Start_Labeling", and select the tag from the drop-down list box.



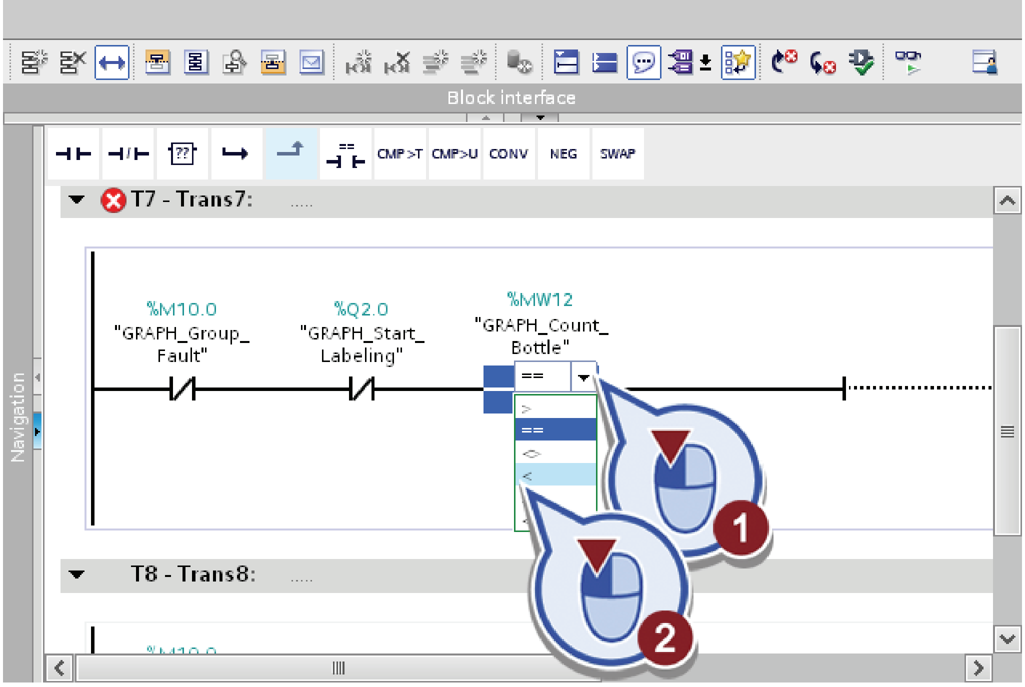
"GRAPH_Filling_Comple...	Bool	%M20.0
"GRAPH_Group_Fault"	Bool	%M10.0
"GRAPH_Mixer_ON"	Bool	%Q1.0
"GRAPH_Sequence_DB"	Instance DB of ...	>>
"GRAPH_Start_Labeling"	Bool	%Q2.0

gr

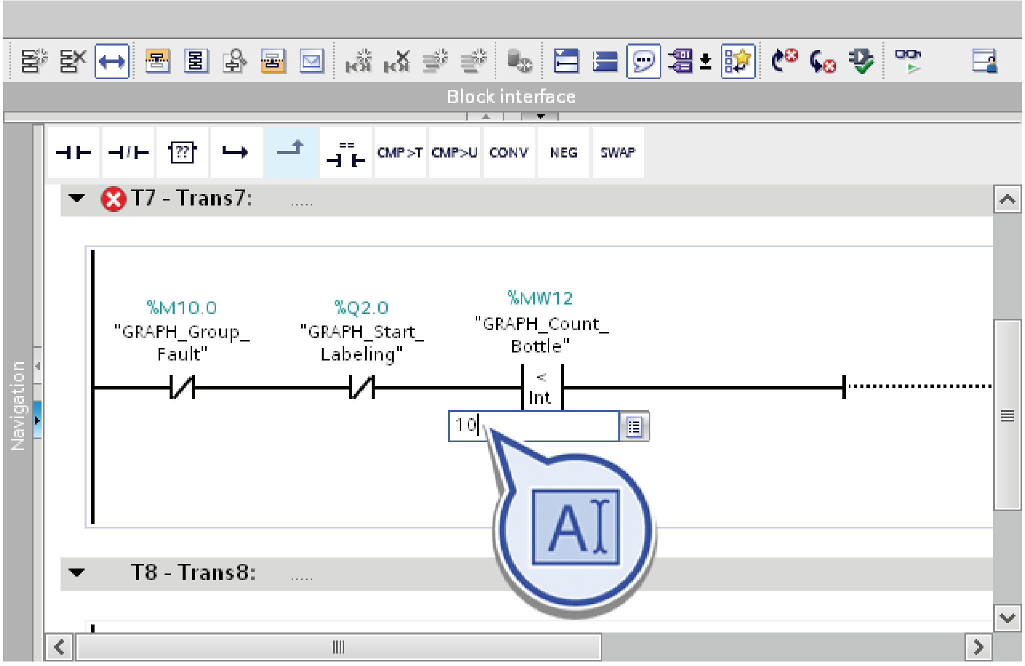
3. Double-click the above operand placeholder of the comparator and select the tag "GRAPH_Count_Bottle".



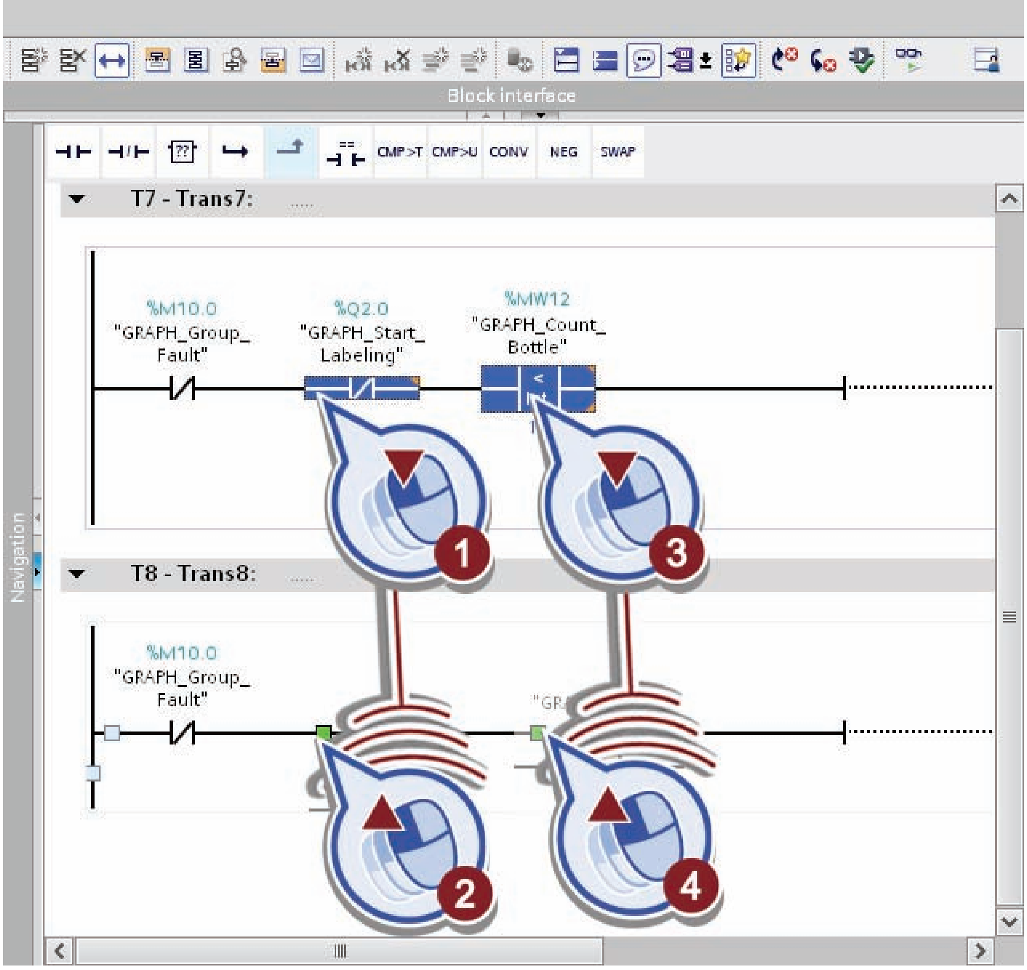
4. Double-click the characters "==" of the comparator and select the character "<". This converts the comparator "Equal to" to "Less than".



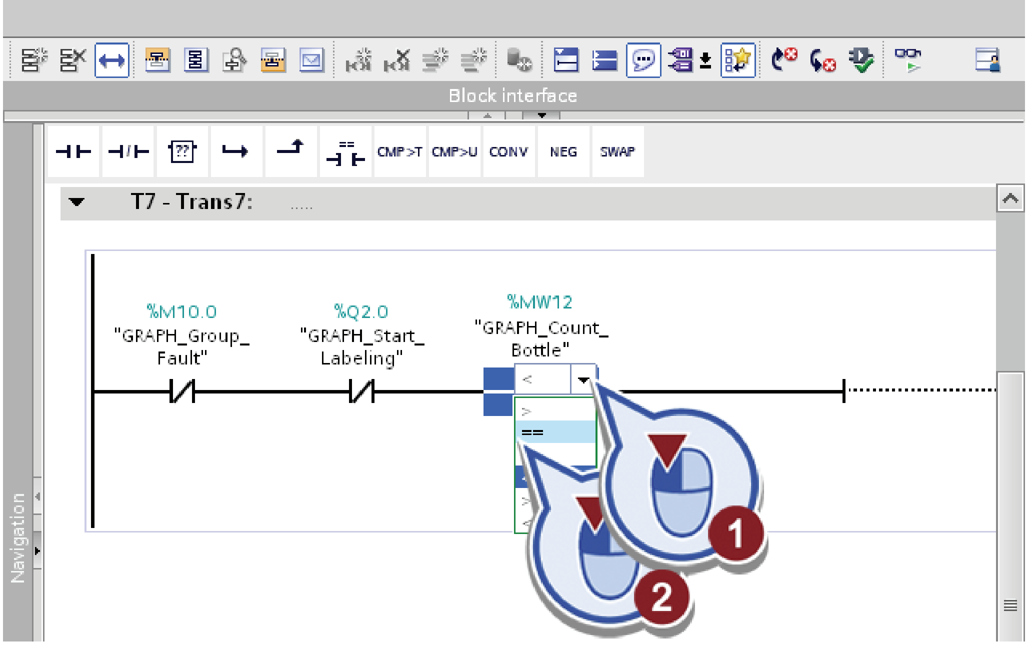
5. Double-click the operand placeholder below the comparator shown and assign the value "10" to it.



6. Select the transition conditions "GRAPH_Start_Labeling" and "GRAPH_Count_Bottle" and drag each of the selected conditions onto the power rail of the transition condition "T8 - Trans8" while holding down the <Ctrl> key.



7. Double-click the character "<" of the comparator and select the character "==". This converts the comparator "Less than" into the comparator "Equal to".

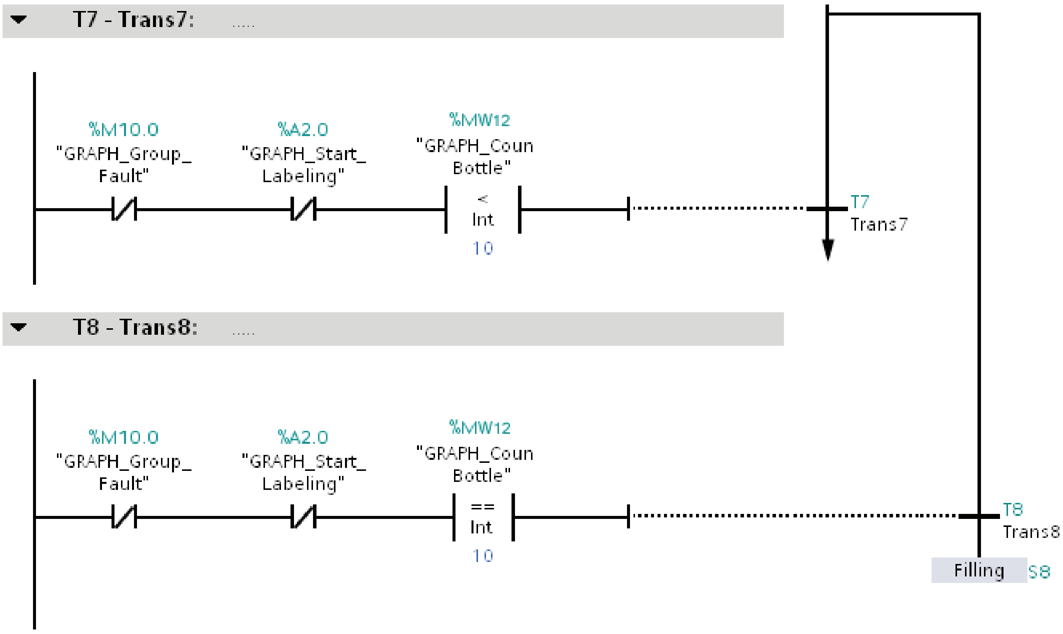


8. Save the project.

Result

You have successfully programmed the transition conditions of the step "S7 Labeling".

- If the transition condition "T7 - Trans7" is satisfied, the processing of the sequencer jumps back to the step "S4 Transport Filling" and another empty bottle is transported to the filling process.
- As soon as the transition condition "T7 - Trans7" is no longer satisfied, i.e. the counter "GRAPH_Count_Bottle" has reached the value "10", the sequencer switches in the alternative branch to the step "S8 Filling Complete".



4.3.4.12 Step S8 Filling Complete - Programming actions and transitions

Introduction

In the following section you will program the end of the GRAPH sequencer and jump back to the beginning. By means of the jump within the sequencer back to the beginning, the processing of the program is continued in a loop and doesn't have to be manually initiated.

In the step itself, a status bit is set, which detects the completion of the filling process.

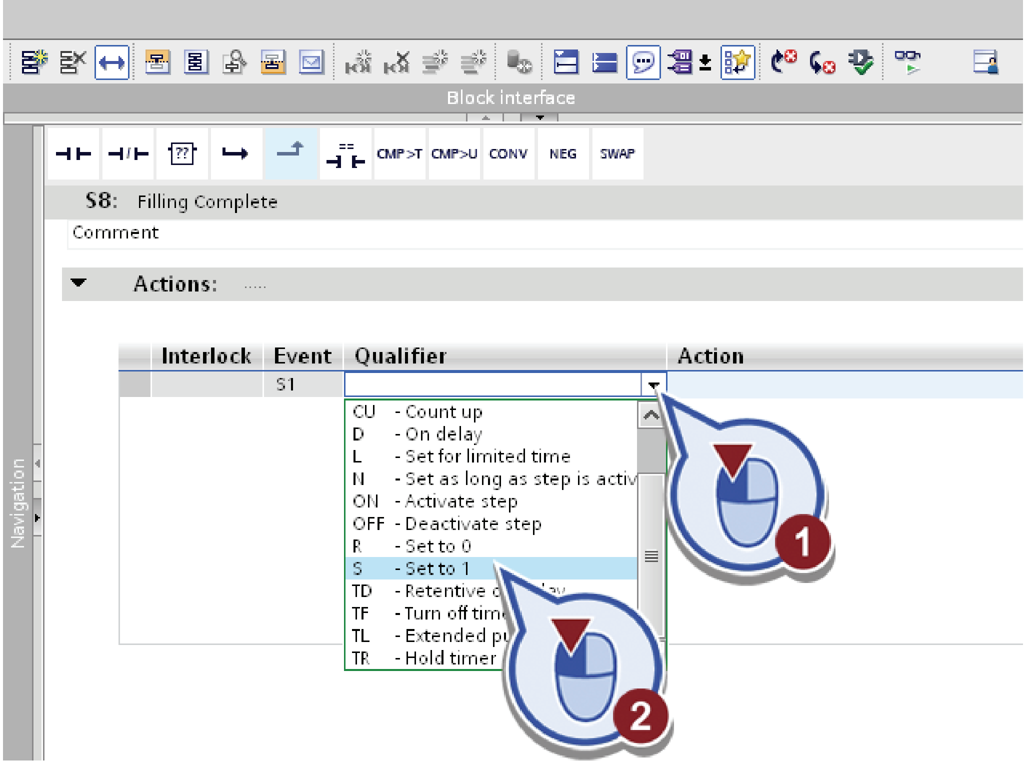
Requirement

You have opened the step "S8 Filling Complete".

Procedure

To program the last step of the sequencer, proceed as follows:

1. In the event "S1 - Incoming step" select the qualifier "S – Set to 1".



S8: Filling Complete

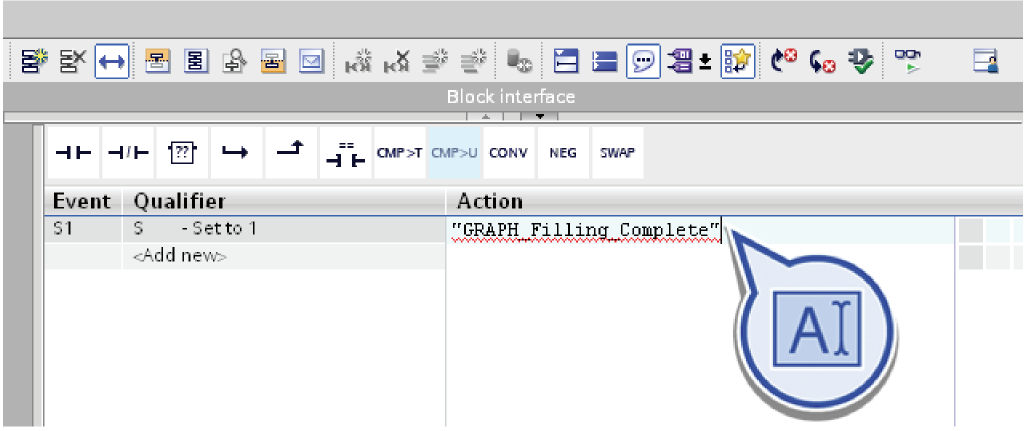
Comment

Actions:

Interlock	Event	Qualifier	Action
	S1		

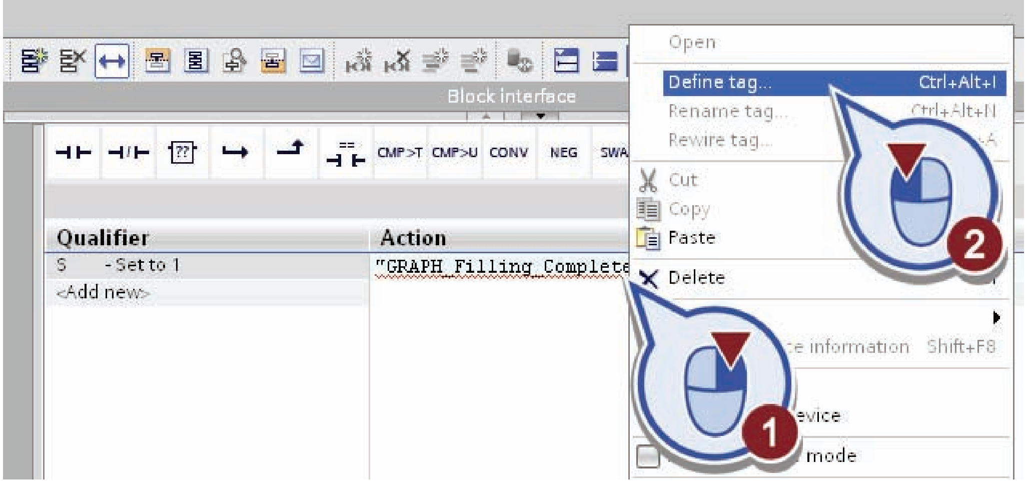
CU - Count up
D - On delay
L - Set for limited time
N - Set as long as step is activ
ON - Activate step
OFF - Deactivate step
R - Set to 0
S - Set to 1
TD - Retentive d
TF - Turn off tim
TL - Extended pu
TR - Hold timer

2. Replace the text <Variable_name_or_address> with "GRAPH_Filling_Complete".



Event	Qualifier	Action
S1	S - Set to 1	"GRAPH_Filling_Complete"
	<Add new>	

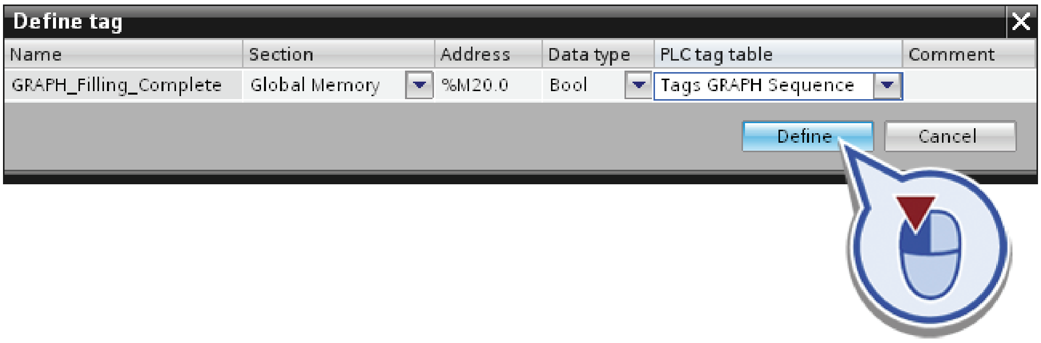
3. Right-click the text "GRAPH_Filling_Complete" and select "Define tag" from the shortcut menu.



Qualifier	Action
S - Set to 1	"GRAPH_Filling_Complete
<Add new>	

4. Define the tag with the following properties:
   – Section: "Global Memory"
   – Address: "M20.0"
   – Date type: "Bool"
   – PLC tag table: "Tags GRAPH Sequence"

   Confirm the dialog by clicking "Define".



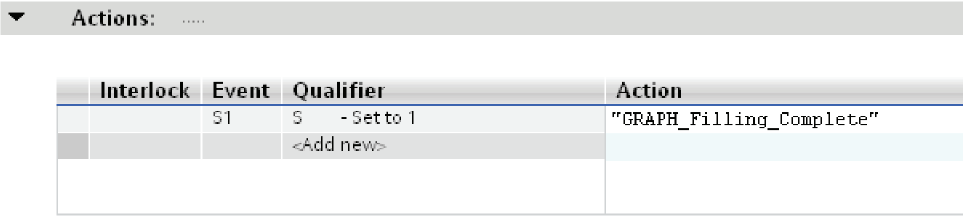
Define tag

Name	Section	Address	Data type	PLC tag table	Comment
GRAPH_Filling_Complete	Global Memory	%M20.0	Bool	Tags GRAPH Sequence	

5. Save the project.

Result

You have successfully programmed the action GRAPH_Filling_Complete at the end of the sequencer. After this step the sequencer jumps back to the start, to step "S1 Home", as long as the multistep transition condition is satisfied. With the completion of this last step, you have completed the programming of the GRAPH sequencer.



	Interlock	Event	Qualifier	Action
		S1	S - Set to 1	"GRAPH_Filling_Complete"
			<Add new>	

4.4 Calculating freshness with the SCL block

4.4.1 Overview

Introduction

SCL (Structured Control Language) is a PASCAL oriented high-level programming language for programmable controllers.

SCL is particularly suitable for calculations and for program control, for example to program branches, loops or jumps. SCL is thus particularly suitable for the application areas of data management, process optimization and recipe management, or for mathematical/statistical tasks.

Language elements

SCL also contains high-level programming languages in addition to the typical PLC elements, such as inputs, outputs, timers or bit memories.

- Expressions

  Expressions are computed during the program runtime (with the exception of the constants) and return a value. Expressions can, for example, comprise the following:

  – Operands (e.g. from constants, tags or function calls)

  – Optionally of operators which link the expressions or nest them inside each other. These operators can include arithmetic operators (such as +; -; *; MOD), relational operators (such as >; <; >=) or logical operators (such as NOT; XOR; OR).

- Value assignments

  You can use a value assignment to assign the value of an expression to a tag. On the left side of the assignment is the tag that takes the value of the expression on the right. The value assignments are always closed with a ";".

Overview SCL programming example

In the project "Filling Station" you created an SCL function block to calculate the best-before date for the labeling process.

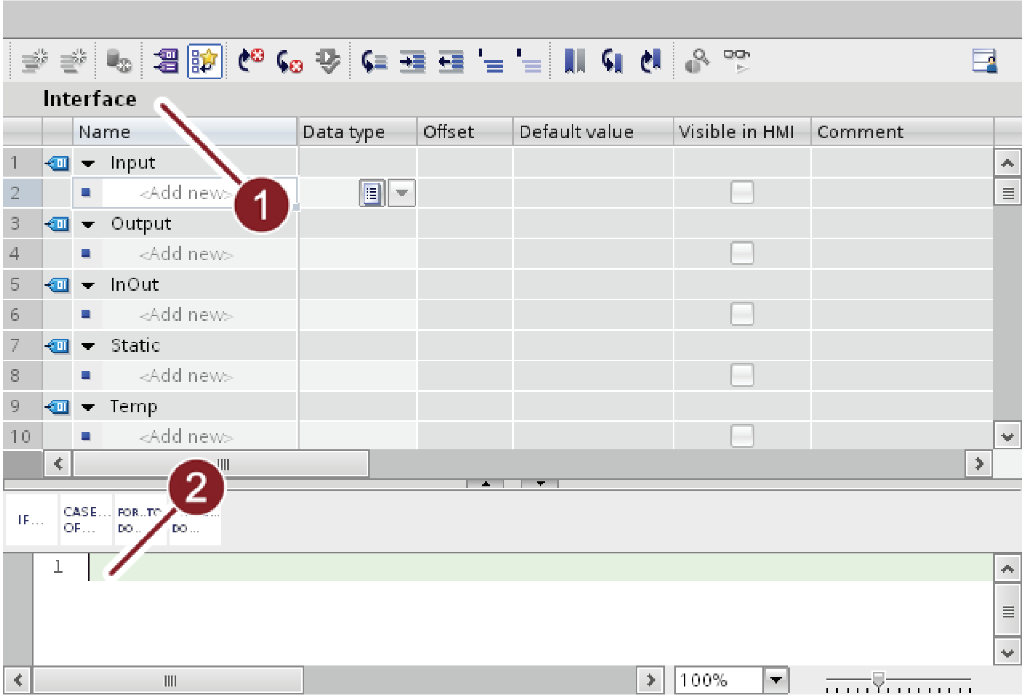
Definition: Function block

Function blocks are logic blocks that store their input, output and in-out parameters permanently in instance data blocks, so that they remain available even after the block has been processed. This is why they are also referred to as "blocks with memory".

Function blocks can work with both static and temporary tags. Static tags allow you to store multi-cycle information. Temporary tags are not saved and are thus available for one cycle only.

Structure of an SCL function block

The following figure shows the structure of an SCL function block:



	Name	Data type	Offset	Default value	Visible in HMI	Comment
1	Input					
2	<Add new>				□	
3	Output					
4	<Add new>				□	
5	InOut					
6	<Add new>				□	
7	Static					
8	<Add new>				□	
9	Temp					
10	<Add new>				□	

①	**Interface** This area of the editor is used to define the input and output parameters with which the SCL function block is interconnected.
②	**Programming** The actual programming of the SCL function block is performed in this area of the editor. The favorites bar contains the most common SCL-specific instructions for creating the program.

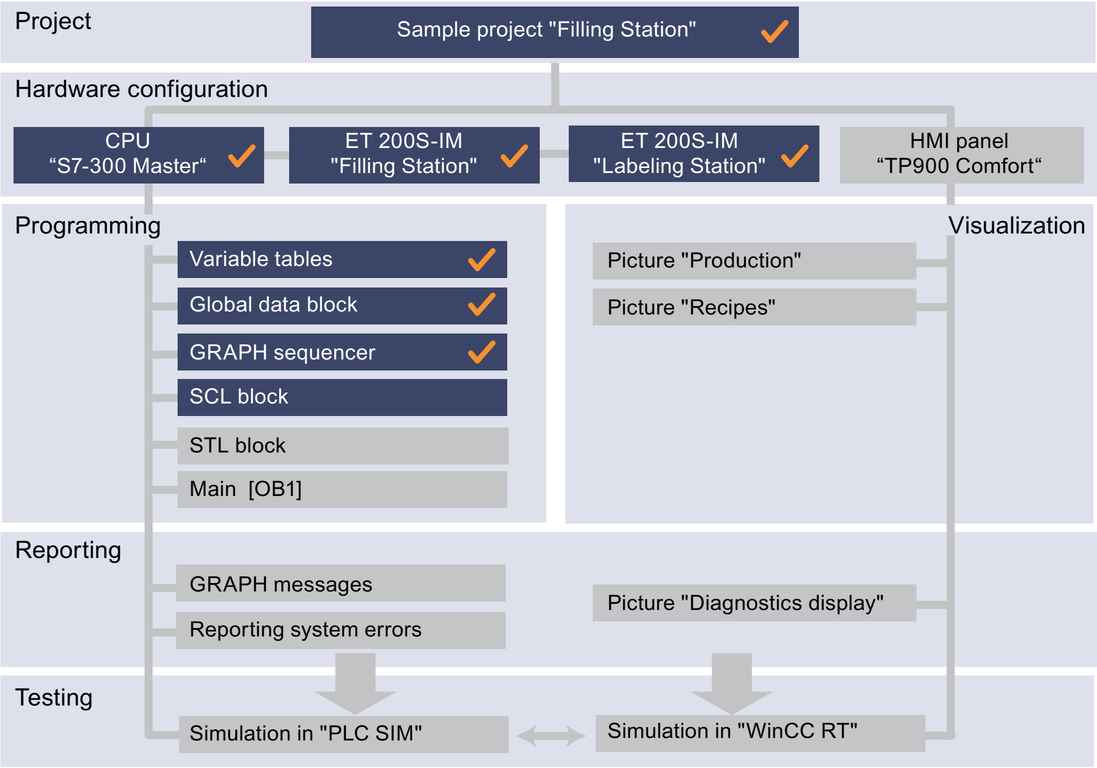
4.4.2 Create SCL function block

Introduction

In the following section you will create the SCL function block "SCL_Best_before_date". You can use the SCL function block to calculate the best-before date.

Progress of project

The following graphic shows you which step you perform next:



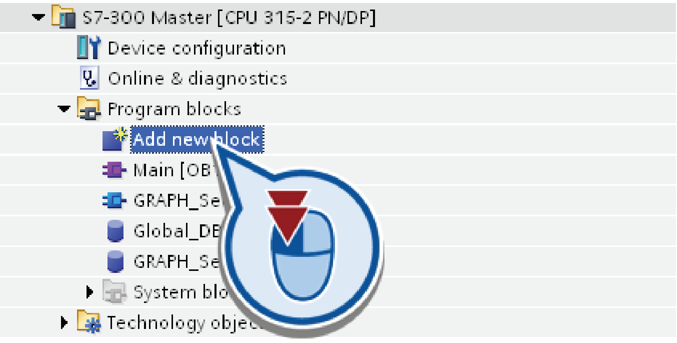
Requirement

You have configured the hardware in the project.

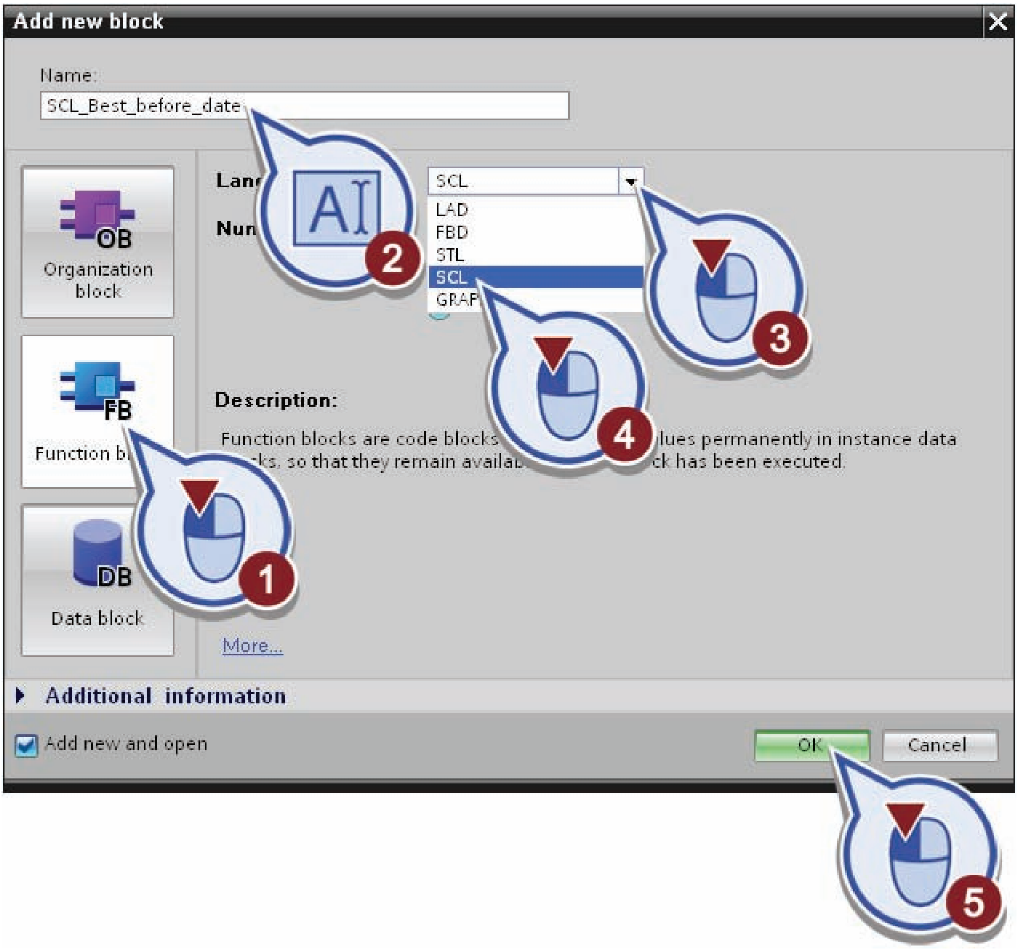
Procedure

To create the SCL function block, proceed as follows:

1. Open the "Program blocks" folder.
2. Double-click "Add new block".



3. To add a function block:
   - Click "Function block".
   - Assign the block name "SCL_Best_before_date"
   - Select the language "SCL".
   - Click "OK"



4. Save the project.

Result

You have successfully created the SCL function block "SCL_Best_before_date". The program editor opens automatically.

4.4.3 Defining the interface of the SCL function block

Introduction

In the following section, you will define the interface of the SCL function block "SCL_Best_before_date". You can use input and output parameters and the temporary tags to program the calculation of the best-before date, which the SCL function block carries out internally. Use a temporary tag to save the current value of the system time.

Definition: Temporary tag

Tags that are used to store temporary intermediate results. Temporary local data are retained for only one cycle. If you use temporary local data you must ensure that the values are written within the cycle before you attempt to read them. Otherwise the values will be random.

Since the system time is read in the format (DT) Date_And_Time but cannot be used for arithmetic operations, you have to convert this format to a different format. To do this, use an AT keyword to overlay the temporary tag of the read system time with an Array of Byte.

Definition: (DT) Date_And_Time

The data type (DT) Date_And_Time contains a date and time-of-day specification in the BCD format with a length of 8 bytes. You will find a graphic on the next page. The byte structure of the data type for the value DT#2011-07-04-10:30:40.201 is shown on the left side of this graphic (July 4, 2011, 10:30, 40 seconds 201 milliseconds). The four least significant bits of byte 7 (bits 0 to 3) store the day of the week as well.

Definition: Array of byte

The ARRAY data type represents a field that consists of a fixed number of components of the same data type.

The field components are addressed by means of an index. The index limits are defined in square brackets (Array [0 .. 7] of Byte) during the declaration of the field after the keyword ARRAY. The low limit value must be smaller than or equal to the high limit value. The index value must be entered directly; tags cannot be entered. A field can contain up to six dimensions, the limits of which are specified in each case separated by a comma.

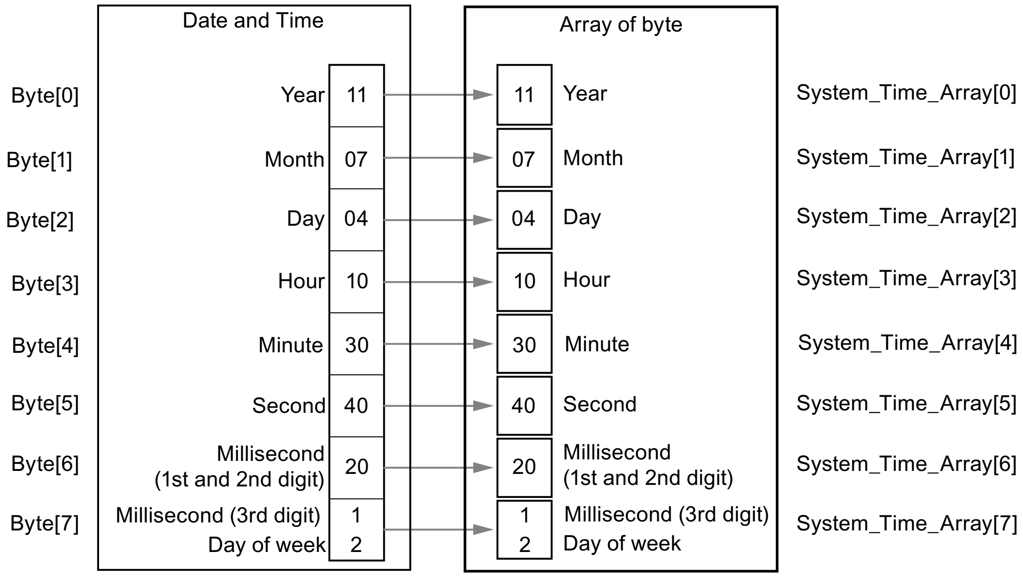
Using AT keyword

To access data areas within a declared tag, you can overlay the declared tags with an additional declaration. This provides you with the option of addressing an already declared tag with a different data type.

To overlay a tag declare another tag directly after the overlaying tag and label this with the keyword "AT" by entering "AT" as data type.

The following figure shows the overlaying of a data type "DT" by an "Array of Byte".

The overlaid values of the data type DT are shown on the right-hand side, distributed over fields 0 to 7 of the data type "Array of Byte".



Requirement

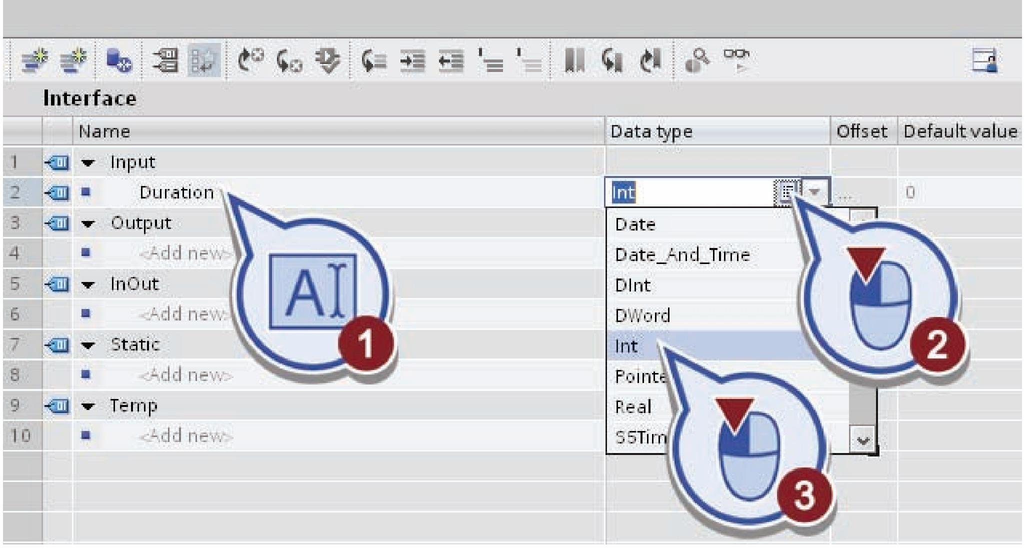
You have successfully created the SCL function block "SCL_Best_before_date".

Procedure

To define the interface, follow these steps:

1. Define an input parameter with the following properties in the "Input" section:
   – Name: "Duration"
   – Date type: "Int"

   You can specify the best-before period in years in this parameter.

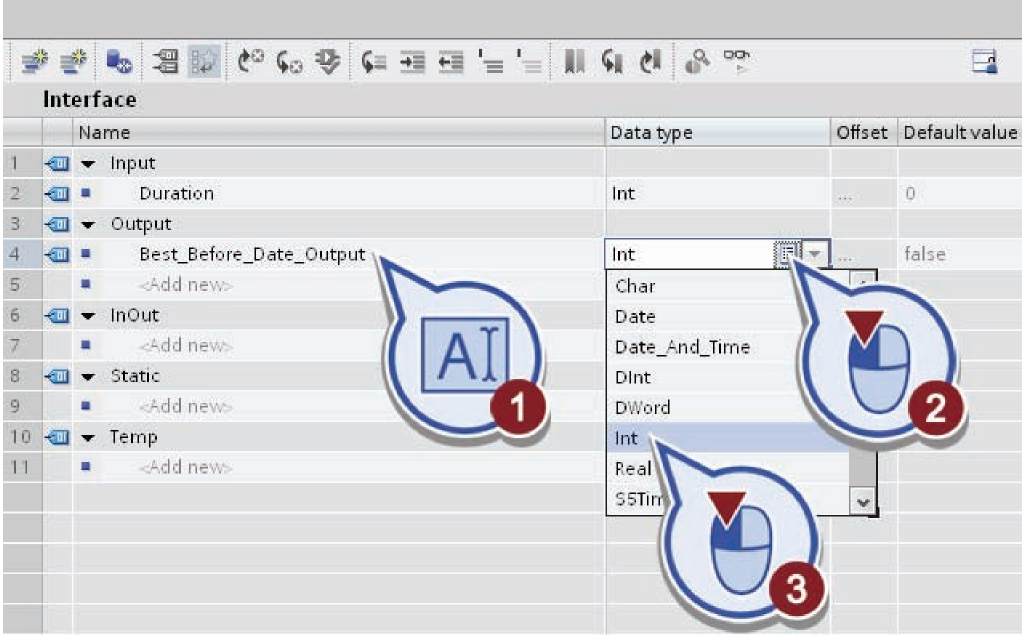


Interface

	Name	Data type	Offset	Default value
1	Input			
2	Duration	Int	...	0
3	Output			
4	<Add new>			
5	InOut			
6	<Add new>			
7	Static			
8	<Add new>			
9	Temp			
10	<Add new>			

2. Define an output parameter with the following properties in the "Output" section:
   – Name: "Best_before_date_Output"
   – Date type: "Int"

   You need this parameter to output the calculated best-before date.

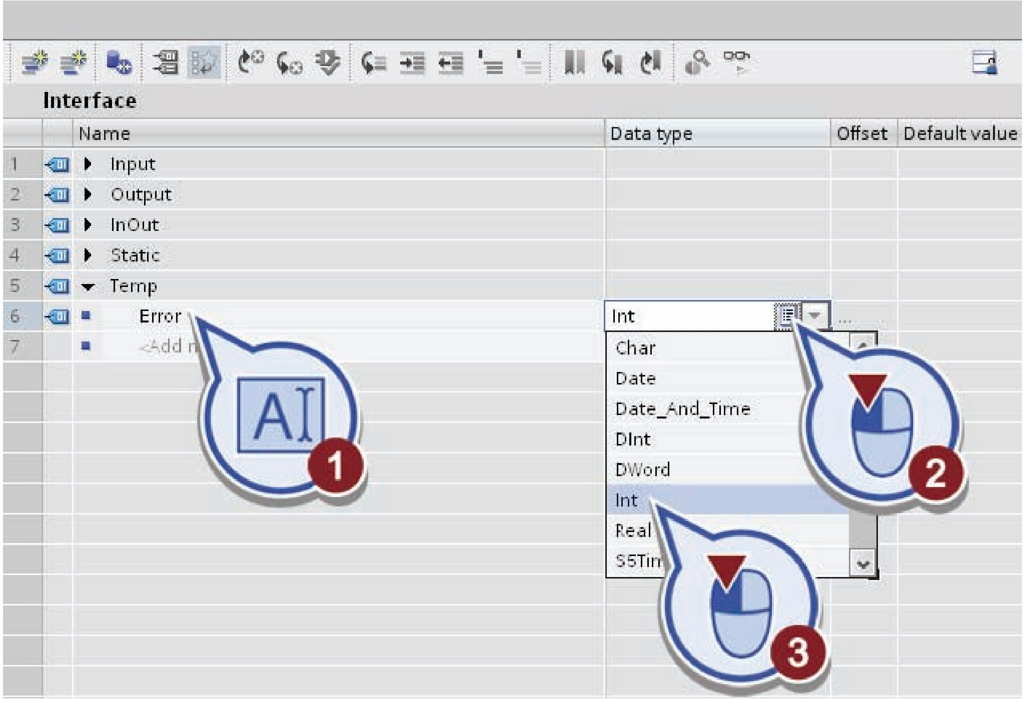


Interface

	Name	Data type	Offset	Default value
1	Input			
2	Duration	Int	...	0
3	Output			
4	Best_Before_Date_Output	Int	...	false
5	<Add new>			
6	InOut			
7	<Add new>			
8	Static			
9	<Add new>			
10	Temp			
11	<Add new>			

3. Define a parameter with the following properties in the "Temp" section:
   - Name: "Error"
   - Date type: "Int"

   You need this temporary parameter to provisionally store the return value of the instruction "RD_SYS_T", which you will program later in the course of the project.

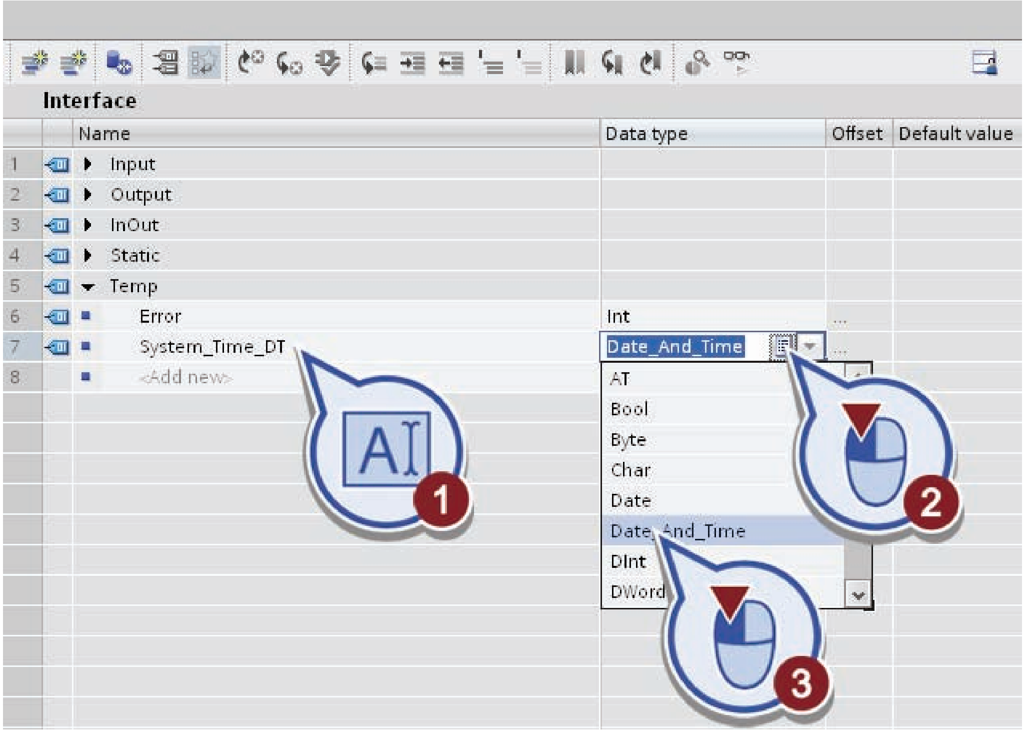


Interface

	Name	Data type	Offset	Default value
1	Input			
2	Output			
3	InOut			
4	Static			
5	Temp			
6	Error	Int		
7	<Add n			

4. Define a second parameter with the following properties in the "Temp" section:
   - Name: "System_Time_DT"
   - Date type: "Date_And_Time"

   You need this temporary parameter to provisionally store the system time of the instruction "RD_SYS_T".

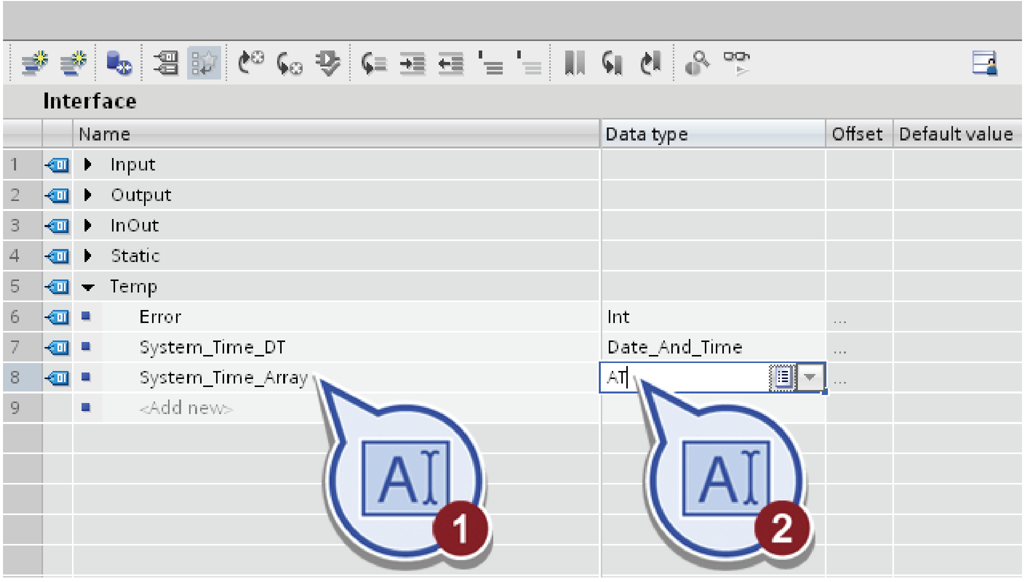


Interface

	Name	Data type	Offset	Default value
1	Input			
2	Output			
3	InOut			
4	Static			
5	Temp			
6	Error	Int	...	
7	System_Time_DT	Date_And_Time	...	
8	<Add new>			

5. Define a third parameter with the following properties in the "Temp" section:
   - Name: "System_Time_Array"
   - Date type: "AT"

   You need this temporary parameter to overlay the interface "System_Time_DT".

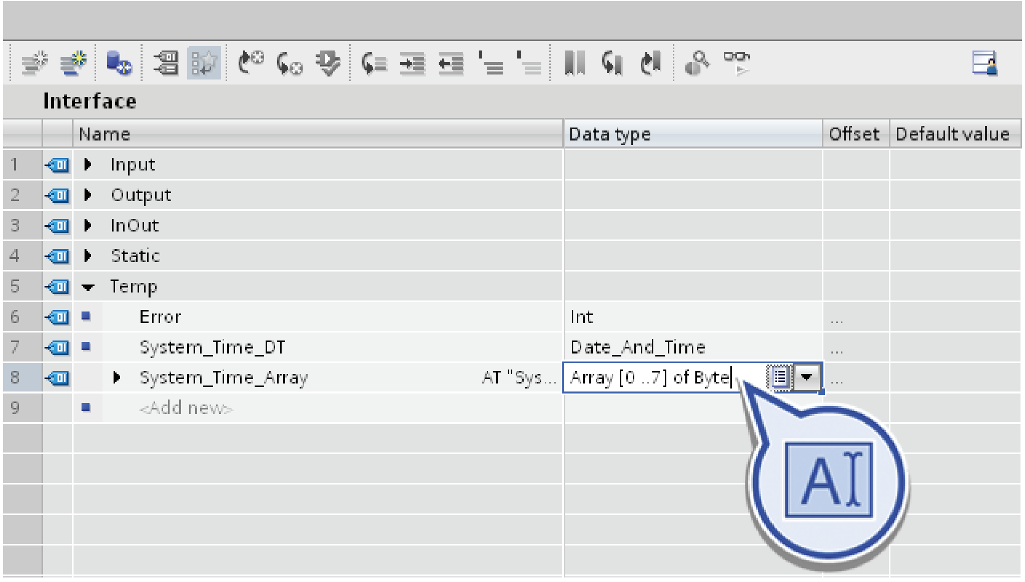


Interface

	Name	Data type	Offset	Default value
1	▸ Input			
2	▸ Output			
3	▸ InOut			
4	▸ Static			
5	▾ Temp			
6	Error	Int	...	
7	System_Time_DT	Date_And_Time	...	
8	System_Time_Array	AT	...	
9	<Add new>			

   The suffix 'AT "System_Time_DT"' is automatically added to the section name, and the data type "AT" converted into "Date_And_Time".

6. Replace the text "Date_And_Time" with "Array [0 .. 7] of Byte".

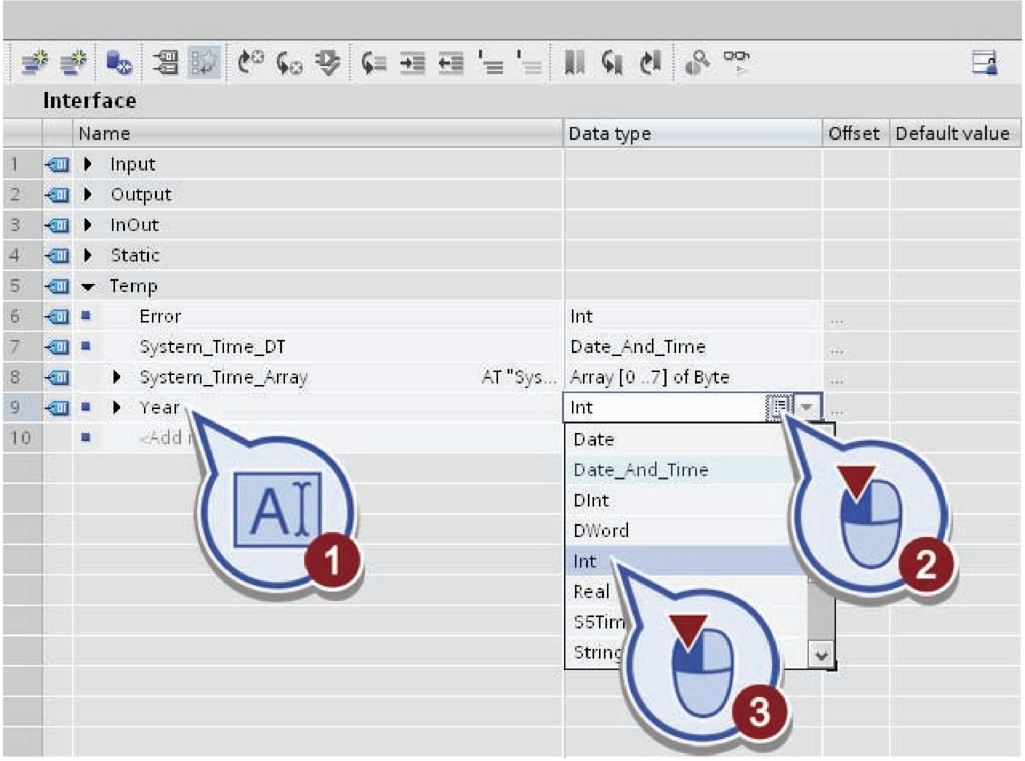


Interface

	Name	Data type	Offset	Default value
1	▸ Input			
2	▸ Output			
3	▸ InOut			
4	▸ Static			
5	▾ Temp			
6	Error	Int	...	
7	System_Time_DT	Date_And_Time	...	
8	▸ System_Time_Array AT "Sys...	Array [0 ..7] of Byte	...	
9	<Add new>			

7. Define a fourth parameter with the following properties in the "Temp" section:
   - Name: "Year"
   - Date type: "Int"

   You need this temporary parameter to provisionally store the year of the system time read.



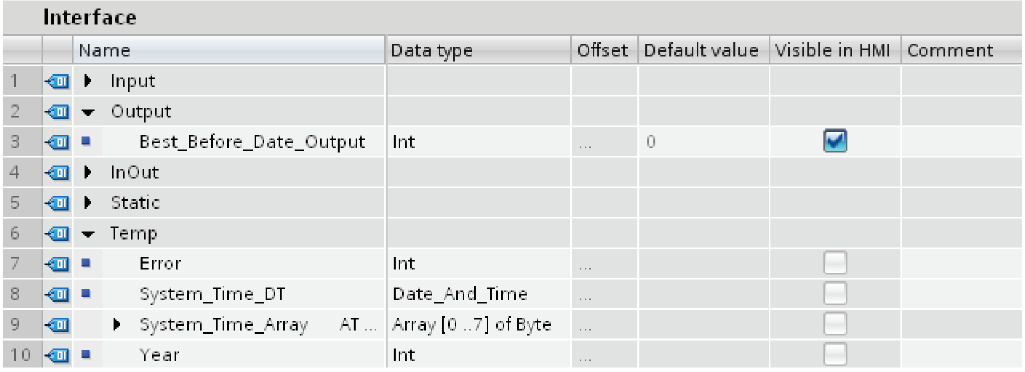
Interface

	Name		Data type	Offset	Default value
1	Input				
2	Output				
3	InOut				
4	Static				
5	Temp				
6	Error		Int	...	
7	System_Time_DT		Date_And_Time	...	
8	System_Time_Array	AT "Sys...	Array [0 ..7] of Byte	...	
9	Year		Int	...	
10	<Add n				

8. Save the project.

Result

You have successfully defined all the required parameters in the interface of the SCL function block.



Interface

	Name	Data type	Offset	Default value	Visible in HMI	Comment
1	▶ Input					
2	▼ Output					
3	■ Best_Before_Date_Output	Int	...	0	☑	
4	▶ InOut					
5	▶ Static					
6	▼ Temp					
7	■ Error	Int	...		☐	
8	■ System_Time_DT	Date_And_Time	...		☐	
9	▶ System_Time_Array AT ...	Array [0 ..7] of Byte	...		☐	
10	■ Year	Int	...		☐	

4.4.4 Programming the calculation of the best-before duration

Introduction

In the following section, you will program the SCL function block to calculate the best-before date.

For the project "Filling Station", only the year should be output:

- To do this, read the current system time of the CPU clock using the instruction "RD_SYS_T". The readout data are stored in the format DT (Date_And_Time) in the temporary interface "System_Time_DT" of the instruction.
- Overlay the temporary parameter "System_Time_DT" with the temporary parameter "System_Time_Array". This distributes the individual values of the data type "DT" on the corresponding bytes of the array.
- The first byte of the array contains the current year. Add the value "2000" to the year, as the format DT (Date_And_Time) only specifies the last two digits of the year (example: the value "11" corresponds to the year "2011"). In addition to the current year add the duration specified on the input interface "Duration".

Requirement

You have defined the interface for the SCL function block.

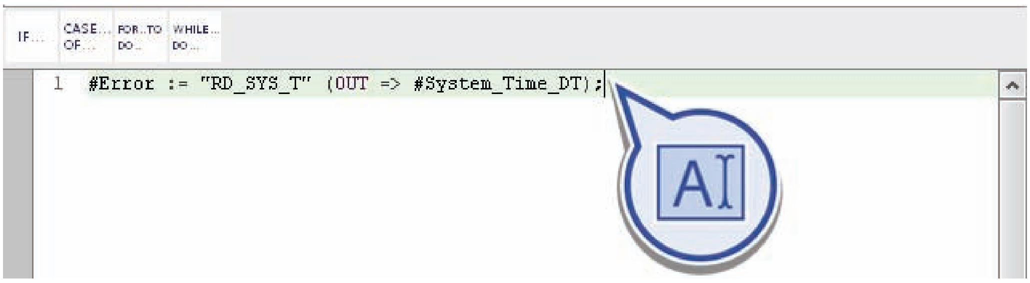
Procedure

To program the SCL function block, proceed as follows:

1. Define the first line of the program code:
   - Write before the instruction: "#Error :="
   - Write the instruction: "RD_SYS_T"
   - Write after the instruction: "(OUT => #System_Time_DT);"

   ### Note

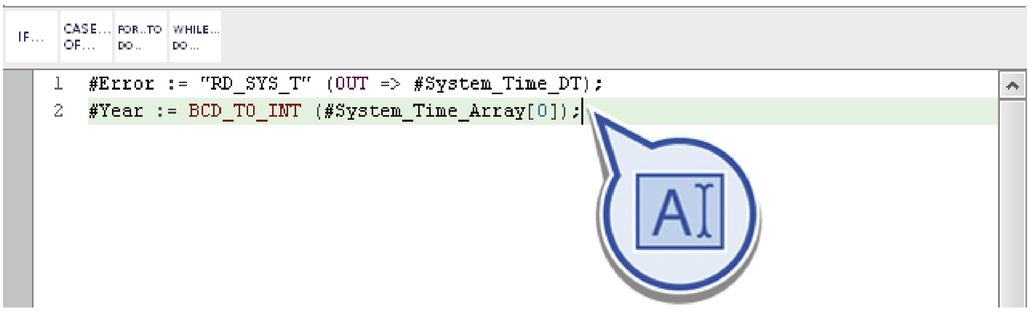
   A list with the possible instructions to be selected appears after you have entered the first letters of the instruction.



```
#Error := "RD_SYS_T" (OUT => #System_Time_DT);
```

2. Define the second line of the program code:
   - Write before the instruction: "#Year :="
   - Write the instruction: "BCD_TO_INT"
   - Write after the instruction: "(#System_Time_Array[0]);"

   This instruction is used to convert the format "BCD" into the format "INT".

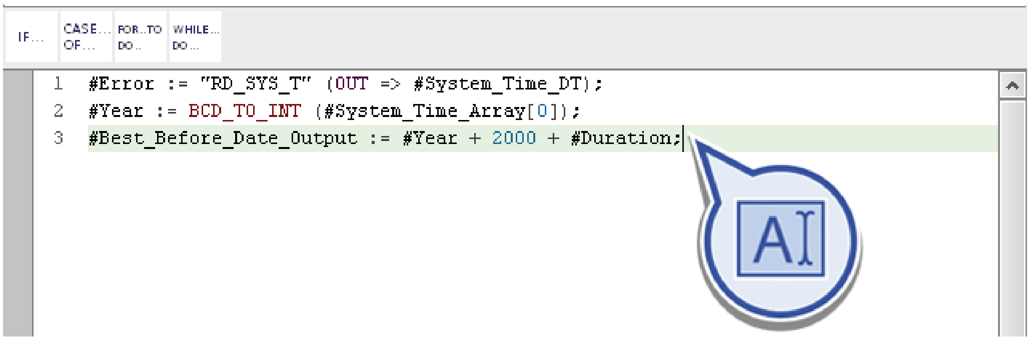


```
#Error := "RD_SYS_T" (OUT => #System_Time_DT);
#Year := BCD_TO_INT (#System_Time_Array[0]);
```

3. Define the third line of the program code:

   – "#Best_Before_Date_Output := #Year + 2000 + #Duration;"

   This instruction is used to add the value "2000" and the current value on the input interface "Duration" to the value "Year".

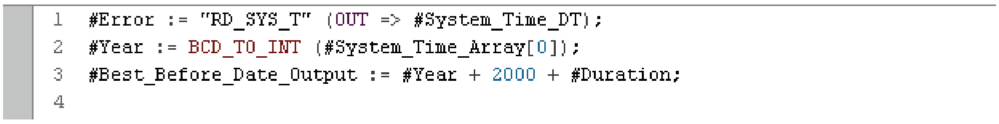


```
#Error := "RD_SYS_T" (OUT => #System_Time_DT);
#Year := BCD_TO_INT (#System_Time_Array[0]);
#Best_Before_Date_Output := #Year + 2000 + #Duration;
```

4. Save the project.

Result

You have successfully programmed the SCL function block for calculating the best-before date.



```
#Error := "RD_SYS_T" (OUT => #System_Time_DT);
#Year := BCD_TO_INT (#System_Time_Array[0]);
#Best_Before_Date_Output := #Year + 2000 + #Duration;
```

4.5 Controlling the conveyor belt with STL function

4.5.1 Overview

Introduction

STL (statement list) is a text-based programming language you can use to program logic blocks. The STL program is organized in networks. The individual STL instructions are programmed in the lines of a network, where only one STL instruction can be specified per line. Each instruction represents a work regulation for the CPU, which is processed in linear manner.

Overview STL programming example

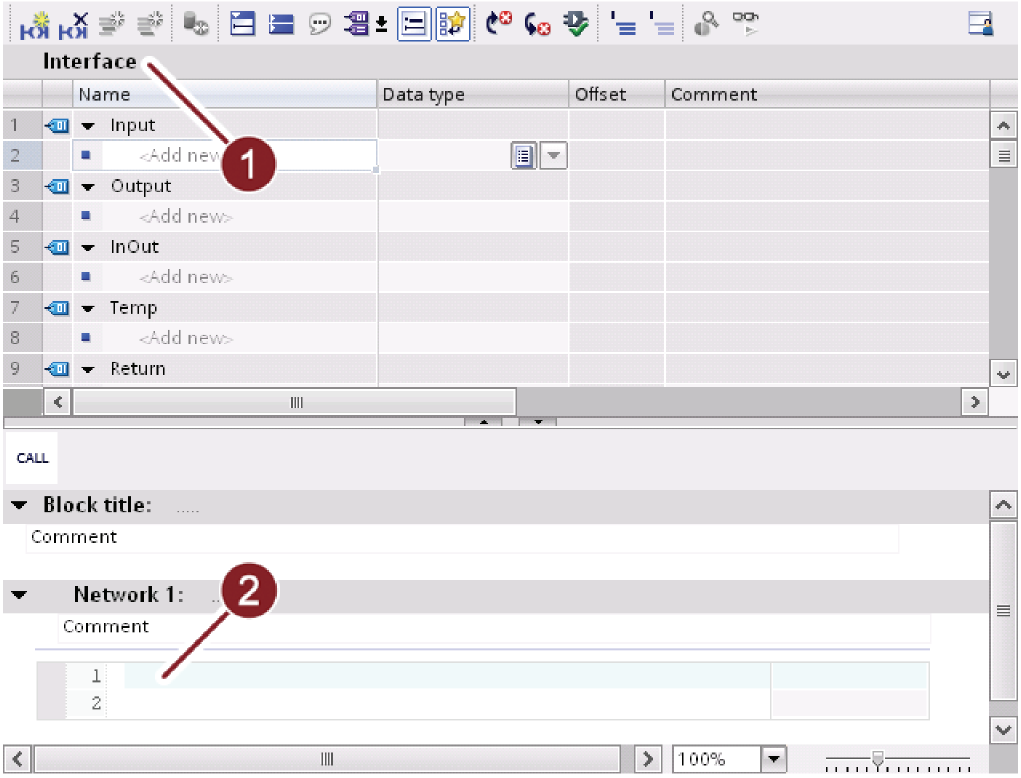
In the sample project "Filling Station", you create a function in STL to control the conveyor belt. The conveyor belt transports the empty bottles to the filling station, stops there for the filling process, and then transports the filled bottles to the labeling station.

Definition: Function

A function (FC) is a logic block without memory. A function provides you with the option of transferring parameters via the function's interface. Functions cannot save data permanently. If a function requires certain data permanently, it should be replaced by a function block.

Structure of the STL editor

The following figure shows the structure of the STL editor:



Interface

	Name	Data type	Offset	Comment
1	Input			
2	<Add new>			
3	Output			
4	<Add new>			
5	InOut			
6	<Add new>			
7	Temp			
8	<Add new>			
9	Return			

①	**Interfaces** This area of the editor is used to define the input and output parameters with which the STL function is interconnected.
②	**Programming** This area of the editor is used to perform the actual programming of the function. You can store frequently required programming instructions in the favorites bar.

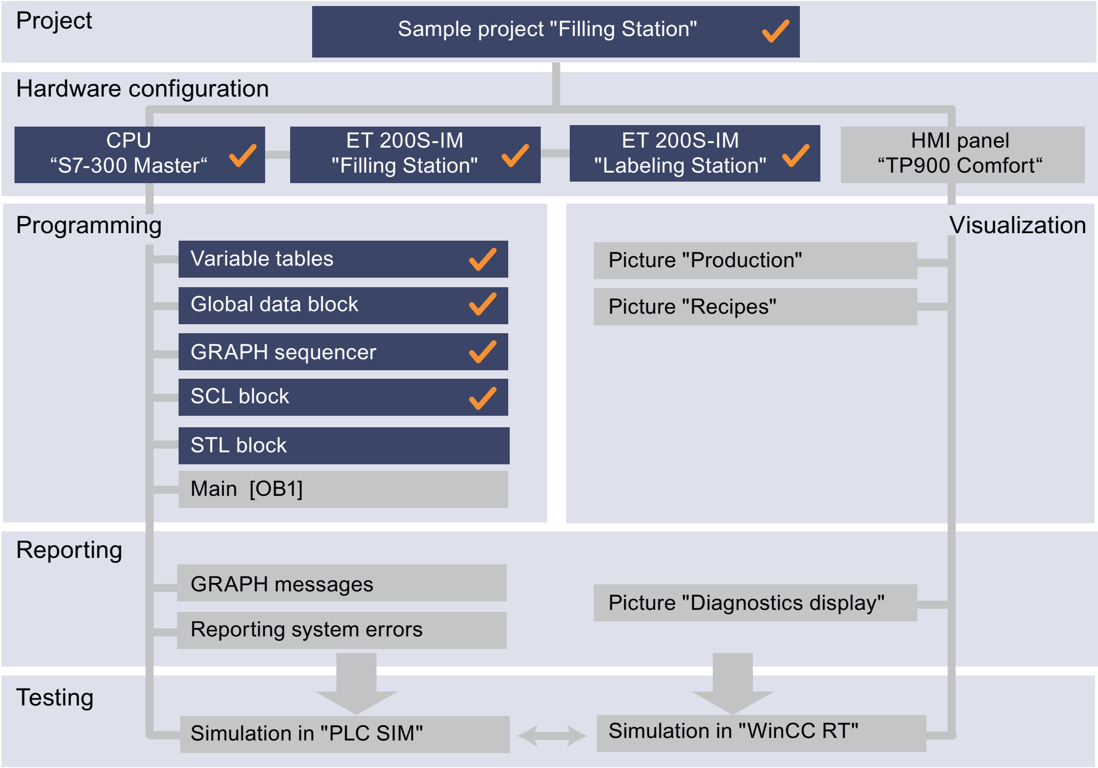
4.5.2 Create STL function

Introduction

In the following section you will create the STL function "STL-Conveyor". The STL function is used to control the conveyor belt.

Progress of project

The following graphic shows you which step you perform next:



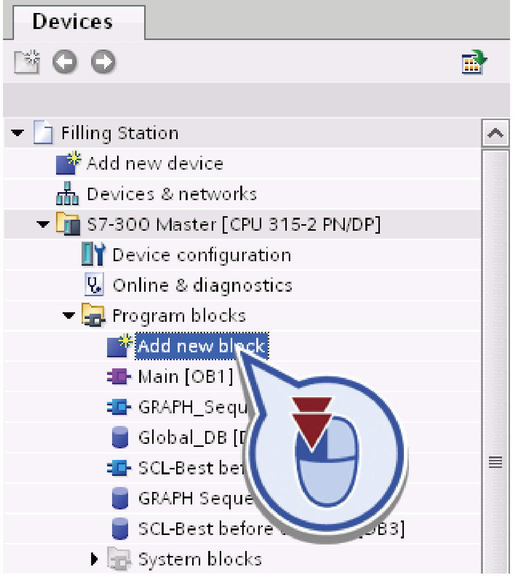
Requirement

You have configured the hardware in the project.

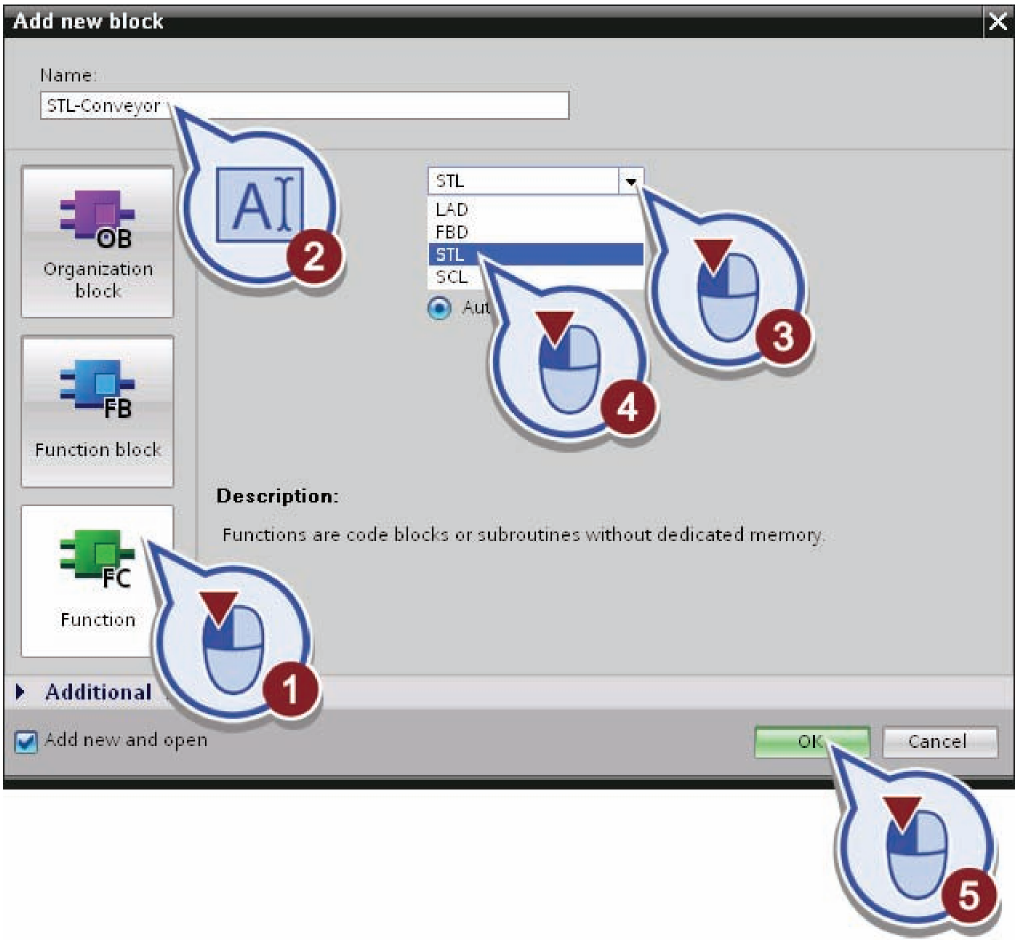
Procedure

To create the STL function "STL-Conveyor", proceed as follows:

1. Open the "Program blocks" folder in the project tree.
2. Double-click "Add new block".



3. To add to a function:
   - Click "Function"
   - Assign the block name "STL-Conveyor"
   - Select the type "STL"
   - Click "OK"



Add new block

Name: STL-Conveyor

STL

Description:

Functions are code blocks or subroutines without dedicated memory.

☑ Add new and open

4. Save the project.

Result

You have successfully created the STL function "STL-Conveyor". The program editor opens automatically.

4.5.3 Defining the interface of the STL function

Introduction

In the following section, you will define the interface of the STL function "STL-Conveyor". The interface is used to transfer the tag values for the program. You can use these tags to control the conveyor belt. The actual programming is performed within the function.

Requirement

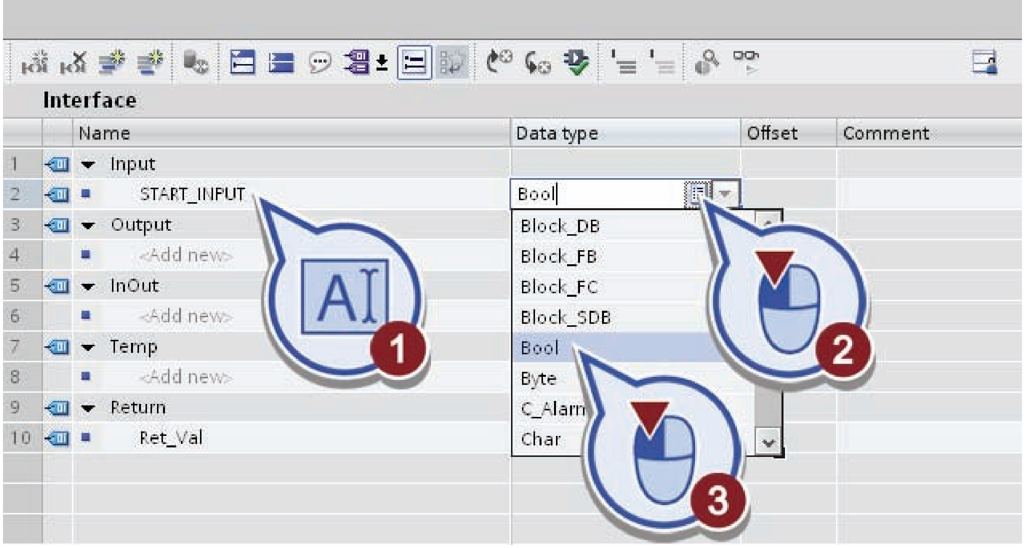
You have created the STL function "STL-Conveyor".

Procedure

To define the interface, follow these steps:

1. Define an input parameter with the following properties in the "Input" section:
   - Name: "START_INPUT"
   - Date type: "Bool"

   You use this parameter to activate the conveyor belt.

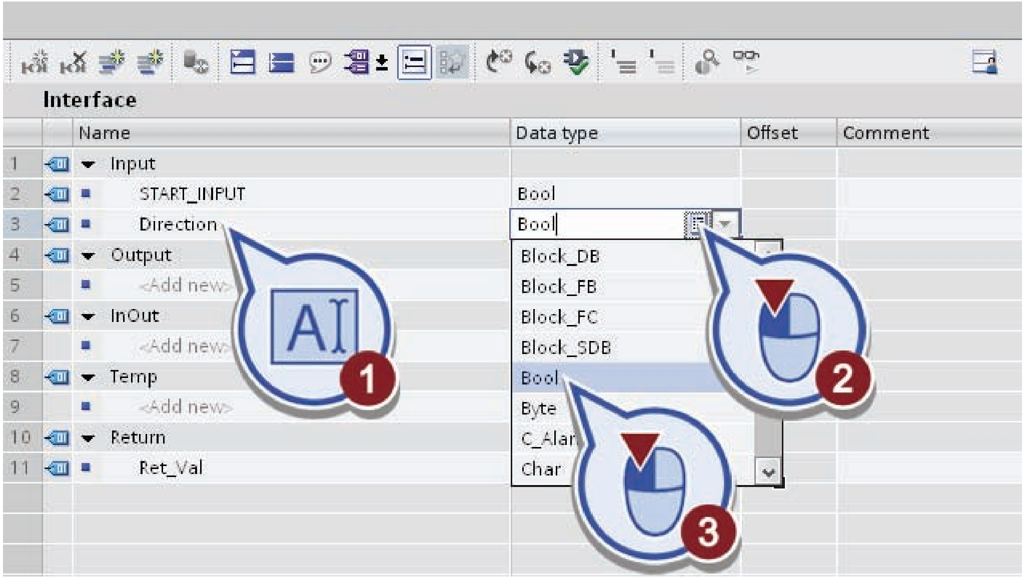


Interface

	Name	Data type	Offset	Comment
1	Input			
2	START_INPUT	Bool		
3	Output			
4	<Add new>			
5	InOut			
6	<Add new>			
7	Temp			
8	<Add new>			
9	Return			
10	Ret_Val			

2. Define a second input parameter with the following properties in the "Input" section:
   - Name: "Direction"
   - Date type: "Bool"

   This parameter is used to query the direction in which the conveyor belt is to travel.



Interface

	Name	Data type	Offset	Comment
1	Input			
2	START_INPUT	Bool		
3	Direction	Bool		
4	Output			
5	<Add new>			
6	InOut			
7	<Add new>			
8	Temp			
9	<Add new>			
10	Return			
11	Ret_Val			

3. Define an output parameter with the following properties in the "Output" section:
   - Name: "Conveyor_DONE"
   - Date type: "Bool"

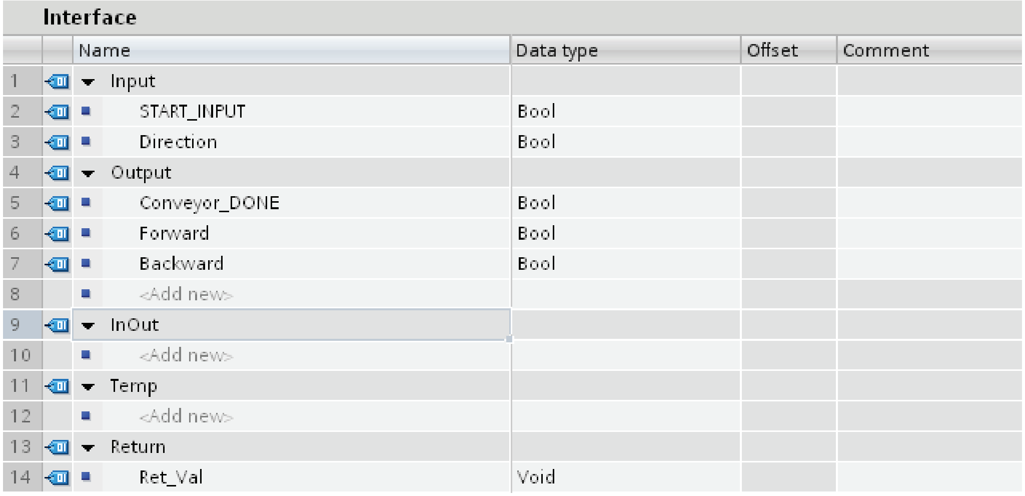
   You use this parameter to query whether the conveyor belt has been activated.
4. Define a second output parameter with the following properties in the "Output" section:
   - Name: "Forward"
   - Date type: "Bool"

   This parameter is used to control the forward motion of the conveyor belt.
5. Define a third output parameter with the following properties in the "Output" section:
   - Name: "Backward"
   - Date type: "Bool"

   This parameter is used to control the backward motion of the conveyor belt.
6. Save the project.

Result

You have successfully defined all the required parameters in the interface of the STL function.



Interface

	Name	Data type	Offset	Comment
1	Input			
2	START_INPUT	Bool		
3	Direction	Bool		
4	Output			
5	Conveyor_DONE	Bool		
6	Forward	Bool		
7	Backward	Bool		
8	<Add new>			
9	InOut			
10	<Add new>			
11	Temp			
12	<Add new>			
13	Return			
14	Ret_Val	Void		

4.5.4 Programming the control of the conveyor belt

Introduction

In the following section you will program the STL function for controlling the conveyor belt. To do this, you will need three networks:

- In the first network you query whether the conveyor belt is to run forward.
  - For this, you have to set the two input parameters "START_INPUT" and "Direction".
  - If this is the case, the output "Forward" should be set.
  - The output "Conveyor_DONE" should be reset to indicate that the conveyor belt is active.
- In the second network you query whether the conveyor belt is to run backward.
  - To do this, you first query whether the input parameter "START_INPUT" is set and whether the input parameter "Direction" is not set.
  - If this is the case, the output "Backward" should be set.
  - The output "Conveyor_DONE" should be reset to indicate that the conveyor belt is active.
- In the third network, you query whether the input parameter "START_INPUT" is not set. If this is the case, both outputs for controlling the conveyor belt should be reset and the output "Conveyor_DONE" should be set.

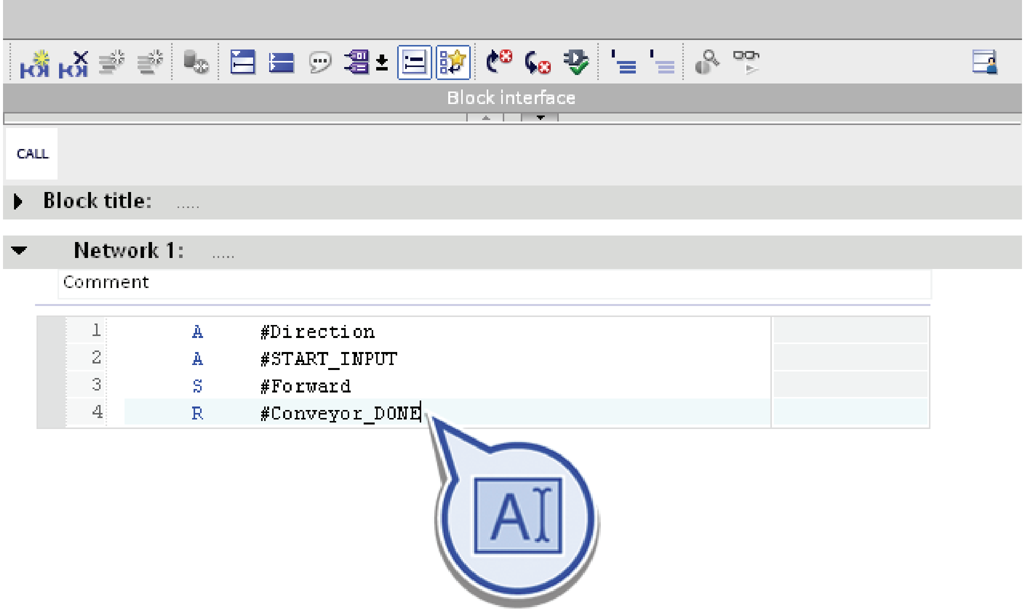
Requirement

You have the defined the interfaces for the STL function.

Procedure

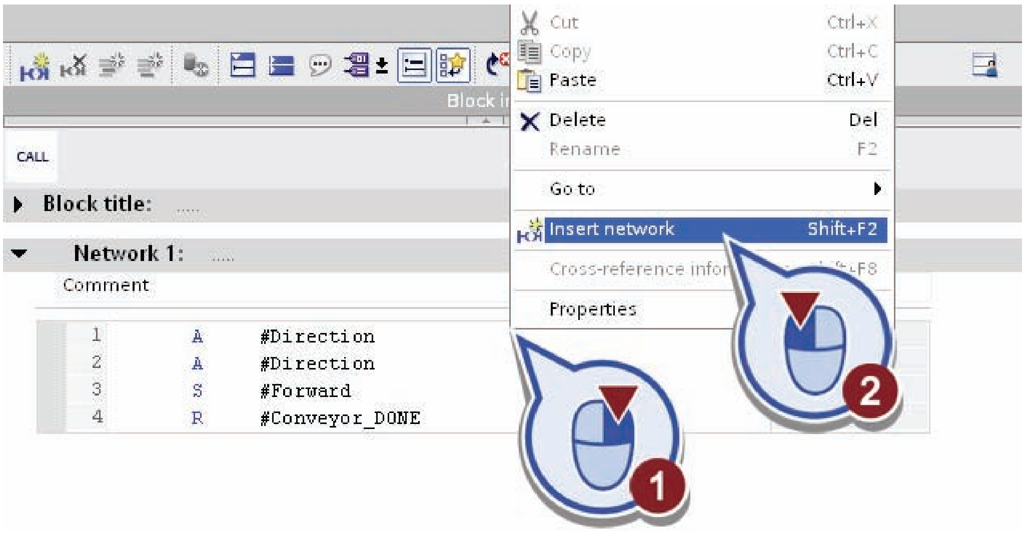
To program the STL function, proceed as follows:

1. Define the program code of network 1:
   - 1. line: "A #Direction"
   - 2. line: "A #START_INPUT"
   - 3. line: "S #Forward"
   - 4. line: "R #Conveyor_DONE"



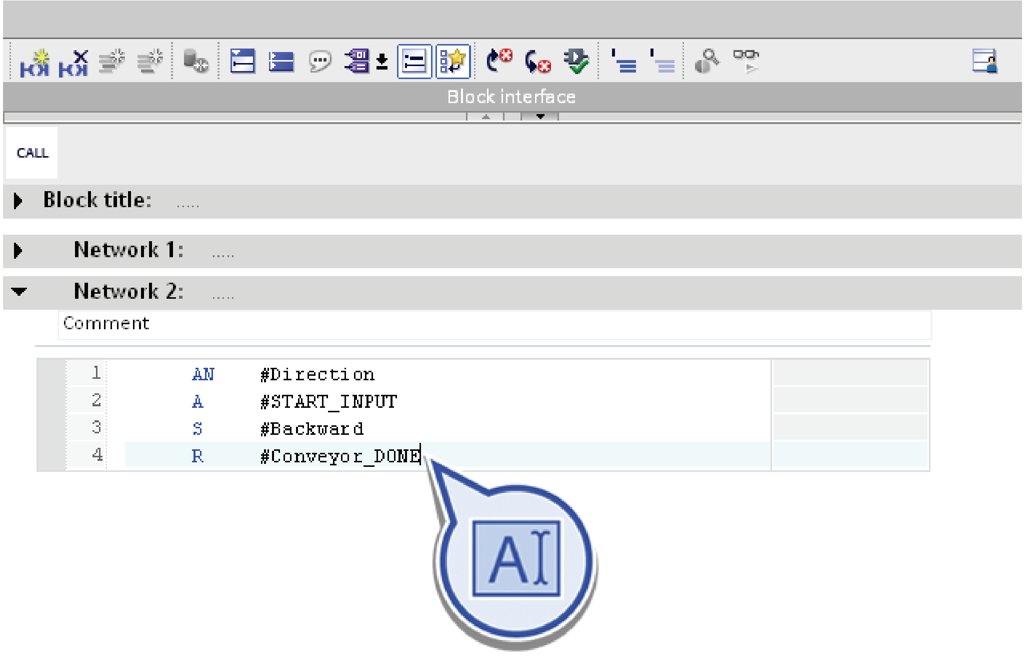
```
1     A     #Direction
2     A     #START_INPUT
3     S     #Forward
4     R     #Conveyor_DONE
```

2. Insert a second network in the editor by right-clicking in a free area of the program editor and selecting "Insert network" from the shortcut menu.



```
1     A     #Direction
2     A     #Direction
3     S     #Forward
4     R     #Conveyor_DONE
```

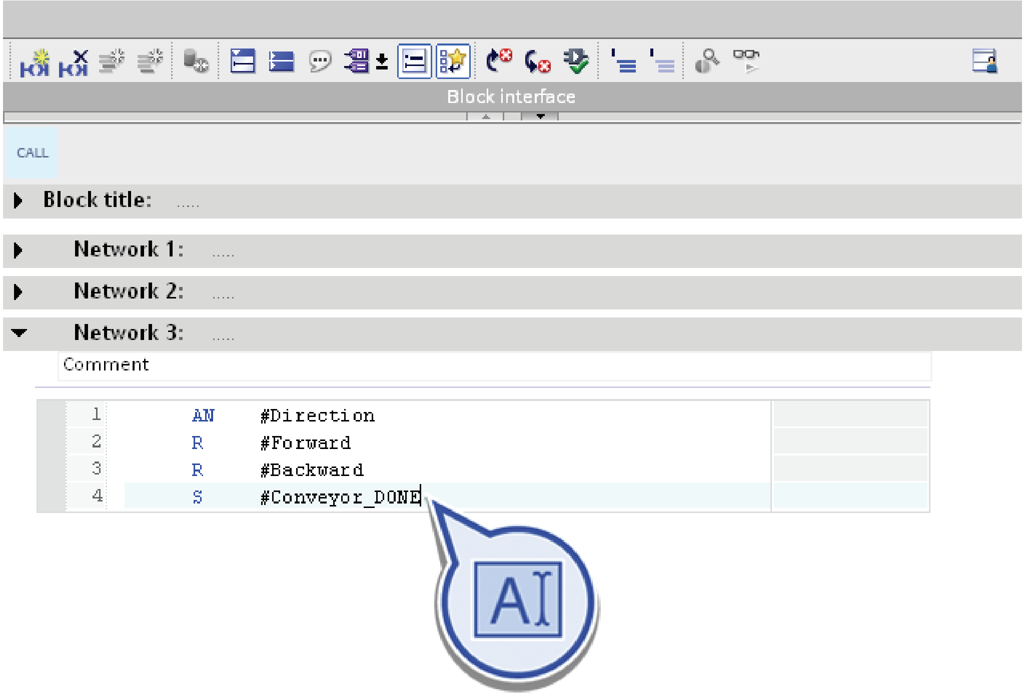
3. Define the program code of network 2:
   - 1. line: "AN #Direction"
   - 2. line: "A #START_INPUT"
   - 3. line: "S #Backward"
   - 4. line: "R #Conveyor_DONE"



```
AN    #Direction
A     #START_INPUT
S     #Backward
R     #Conveyor_DONE
```

4. Create a third network by pressing the key combination <Shift>+<F2>.

5. Define the program code of network 3:
   - 1. line: "AN #START_INPUT"
   - 2. line: "R #Forward"
   - 3. line: "R #Backward"
   - 4. line: "S #Conveyor_DONE"

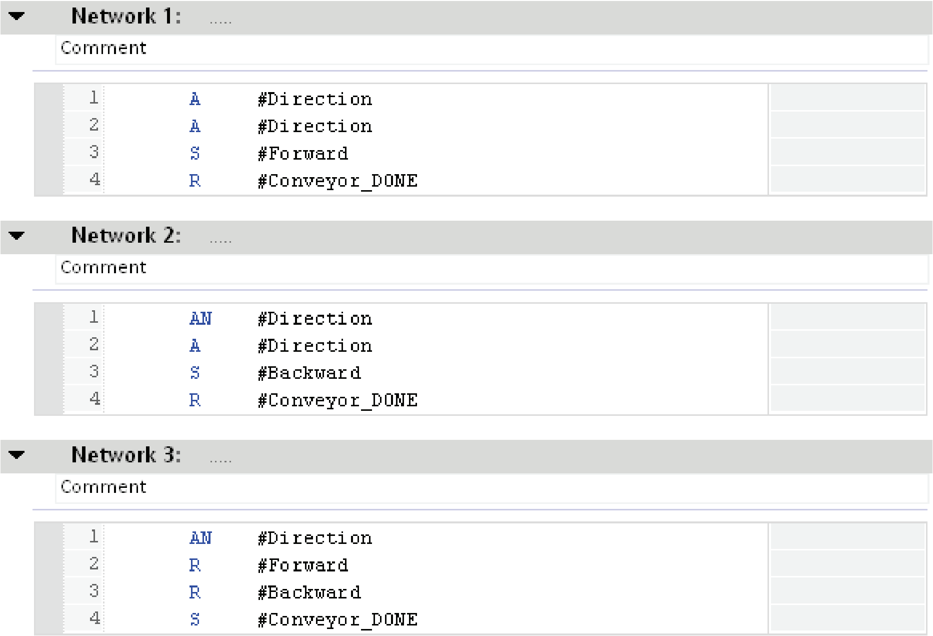


```
AN      #Direction
R       #Forward
R       #Backward
S       #Conveyor_DONE
```

6. Save the project.

Result

You have successfully programmed the STL function for controlling the conveyor belt.



Network 1: .....

Comment

```
A     #Direction
A     #Direction
S     #Forward
R     #Conveyor_DONE
```

Network 2: .....

Comment

```
AN    #Direction
A     #Direction
S     #Backward
R     #Conveyor_DONE
```

Network 3: .....

Comment

```
AN    #Direction
R     #Forward
R     #Backward
S     #Conveyor_DONE
```

4.6 Calling program blocks in the "Main" organization block

4.6.1 Overview of the call structure

Introduction

If you want your program blocks to be executed in the user program, you have to call them from another block. In the sample project "Filling Station" you use the "Main" organization block to do this. When the "Main" organization block calls a program block, the instructions of the called block are executed. Parallel to this the "Main" organization block is run cyclically.

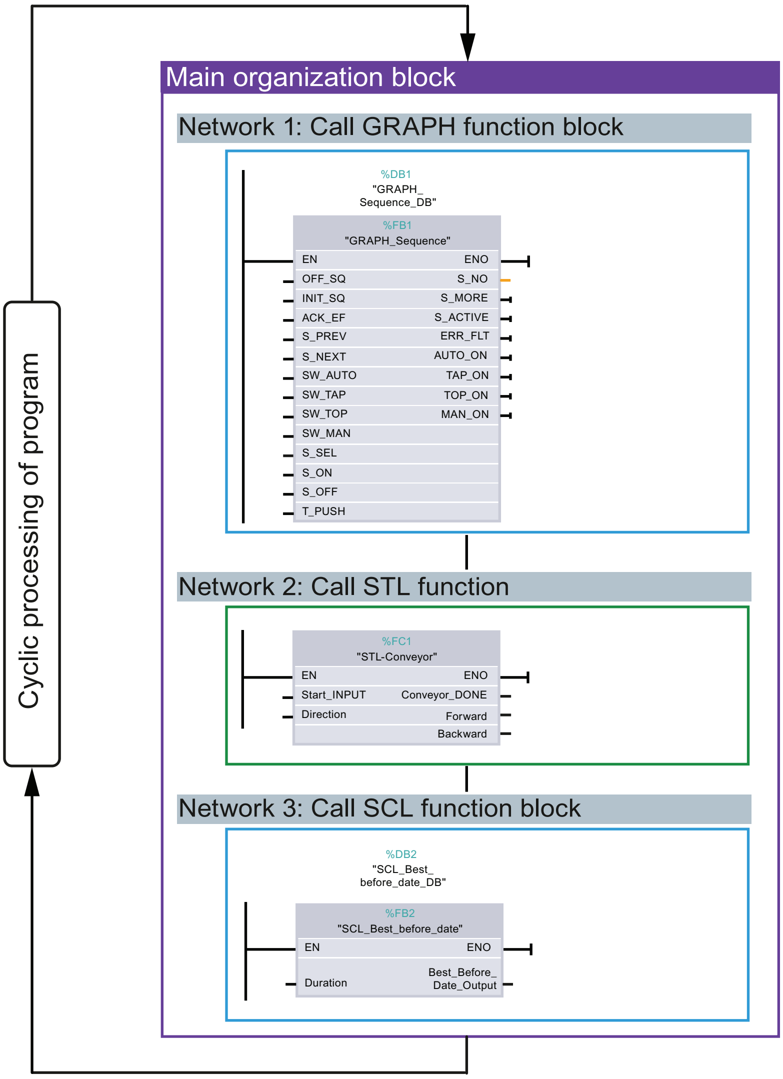
Definition of cyclic organization block

The "Main" organization block is an organization block for cyclic program processing and was created automatically in the "Program blocks" folder in the project tree when the CPU was created . For the program execution to start, at least one program cycle OB must be present in the project. The operating system calls this OB and thus starts the processing of the user program.

The program blocks are called in the following sequence within the "Main" organization block:

1. Network 1: GRAPH FB "GRAPH_Sequence"
2. Network 2: STL function "STL-Conveyor"
3. Network 3: SCL function block "SCL_Best_before_date"

The following figure shows the cyclic program processing of the programmed blocks:



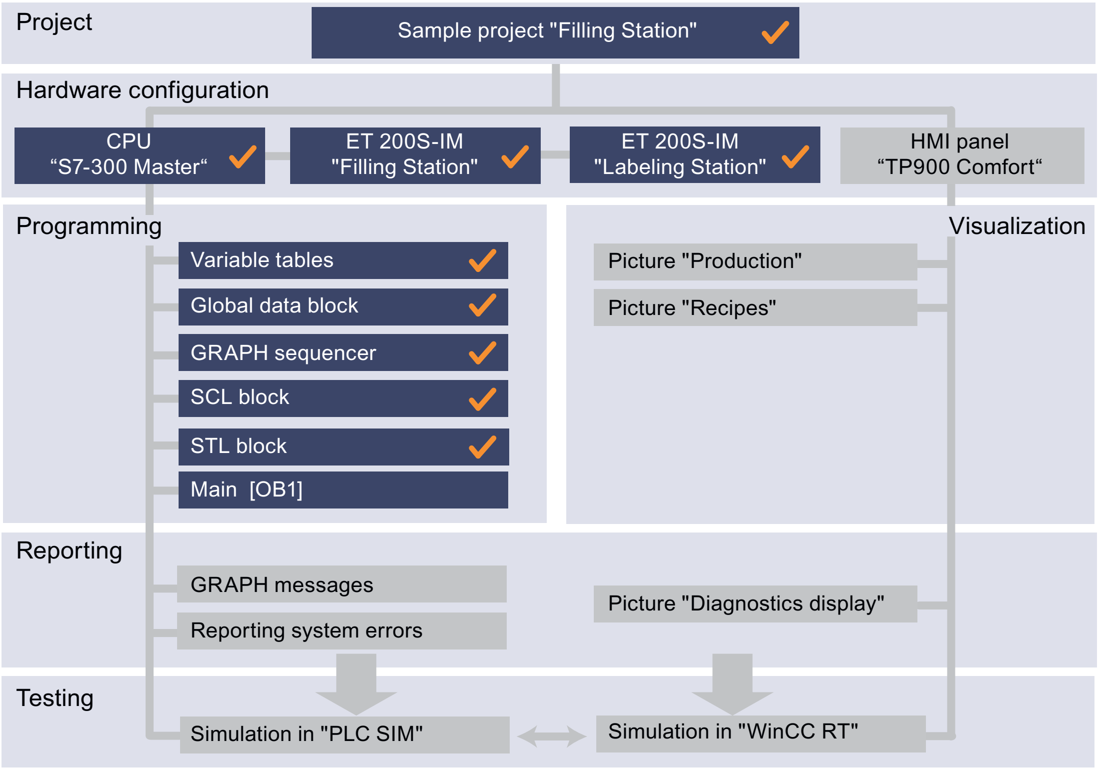
4.6.2 Calling GRAPH sequencer

Introduction

In the following section you will call the GRAPH FB "GRAPH_Sequence" in the organization block "Main" and supply the two input parameters with data. As soon as the CPU switches to RUN mode, the "Main" organization block is called. This in turn calls the GRAPH FB "GRAPH_Sequence".

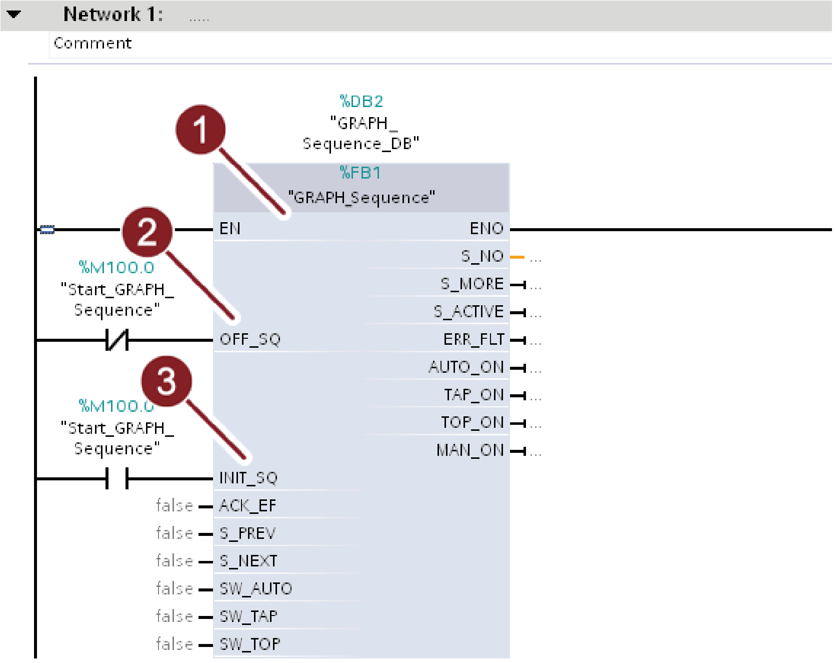
Progress of project

The following graphic shows you which step you perform next:



Calling the GRAPH FB

The following figure shows the calling of the GRAPH FB:



①	The status information of the sequencer and the individual parameters, as well as the status information of the individual steps and transitions are stored in the "GRAPH_Sequence_DB".
②	You can use the input parameter "OFF_SQ" to switch off the GRAPH sequencer. All steps are deactivated when the GRAPH sequencer is switched off.
③	You can use the input parameter "INIT_SQ" to activate the GRAPH sequencer with the initial step. The processing status of all steps is reset in the event of a re-activation of the sequencer via this parameter.

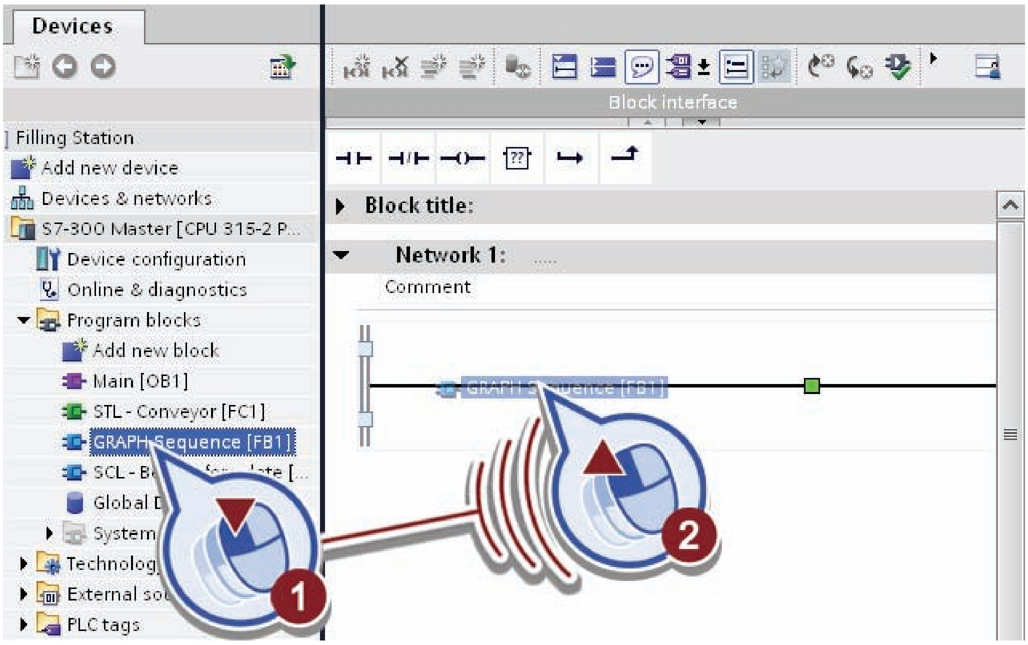
Requirement

You have programmed the program block "GRAPH_Sequence" and opened the organization block "Main".

Procedure

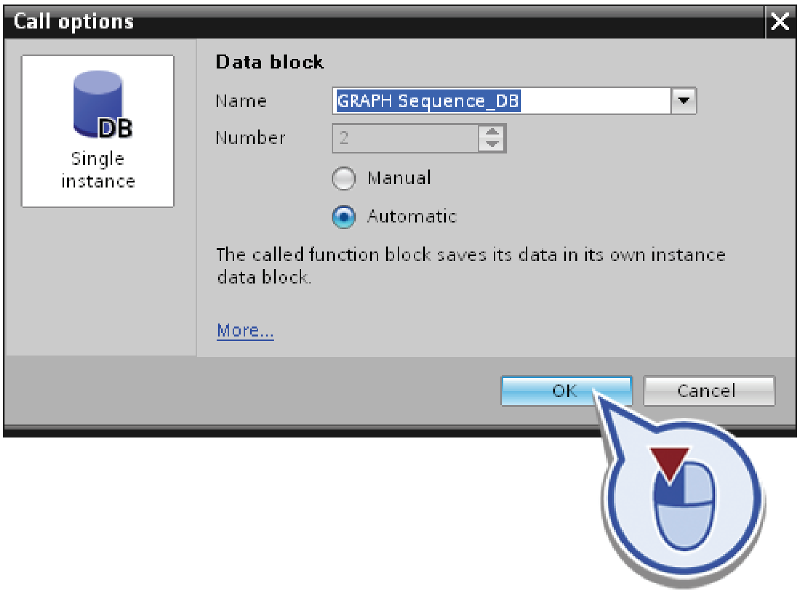
To call the program block, proceed as follows:

1. Drag the GRAPH FB "GRAPH_Sequence" to network 1 of the organization block "Main".



The "Call options" dialog box appears.

2. Click "OK".



Data block

Name: GRAPH Sequence_DB

Number: 2

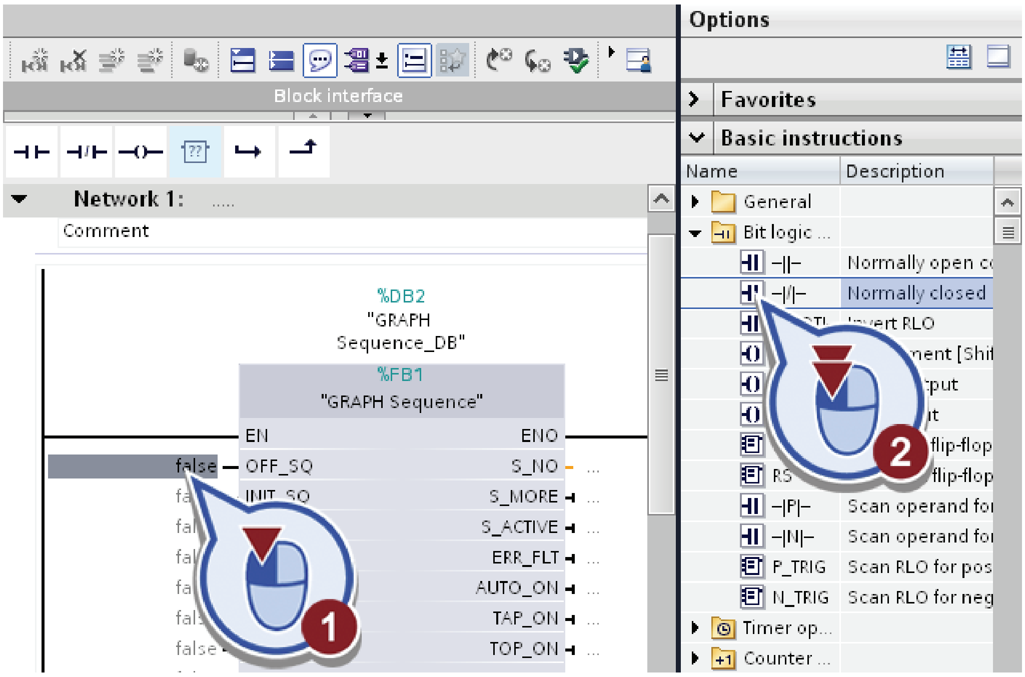
○ Manual

● Automatic

The called function block saves its data in its own instance data block.

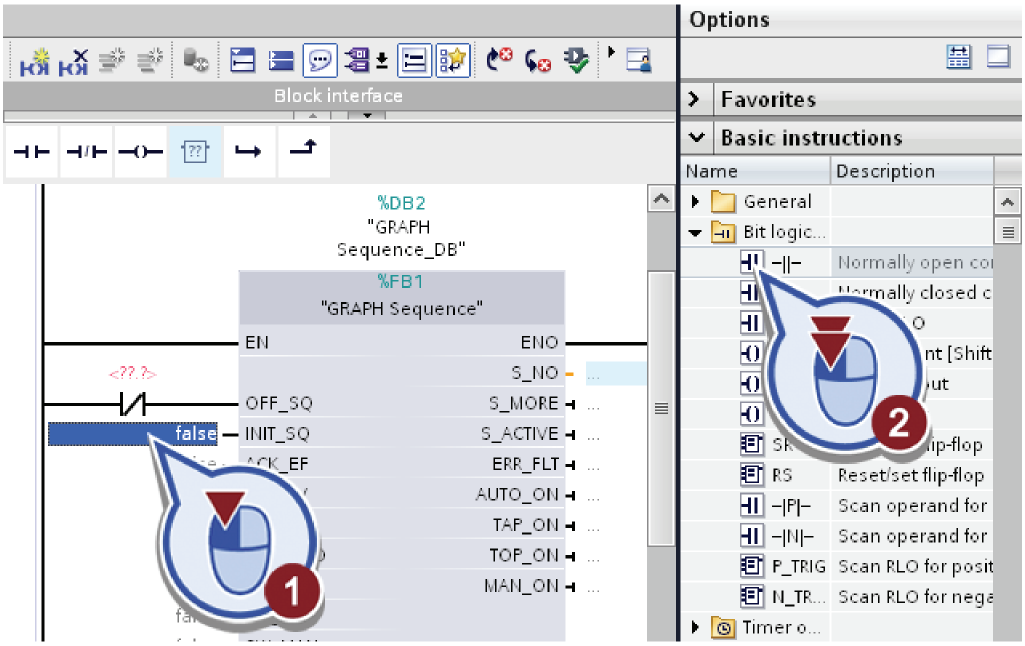
With this step, you create an instance data block for the GRAPH FB.

3. Link the input parameter "OFF_SQ":
   - Click the input.
   - Double-click the "NC contact" from the Task-Card.



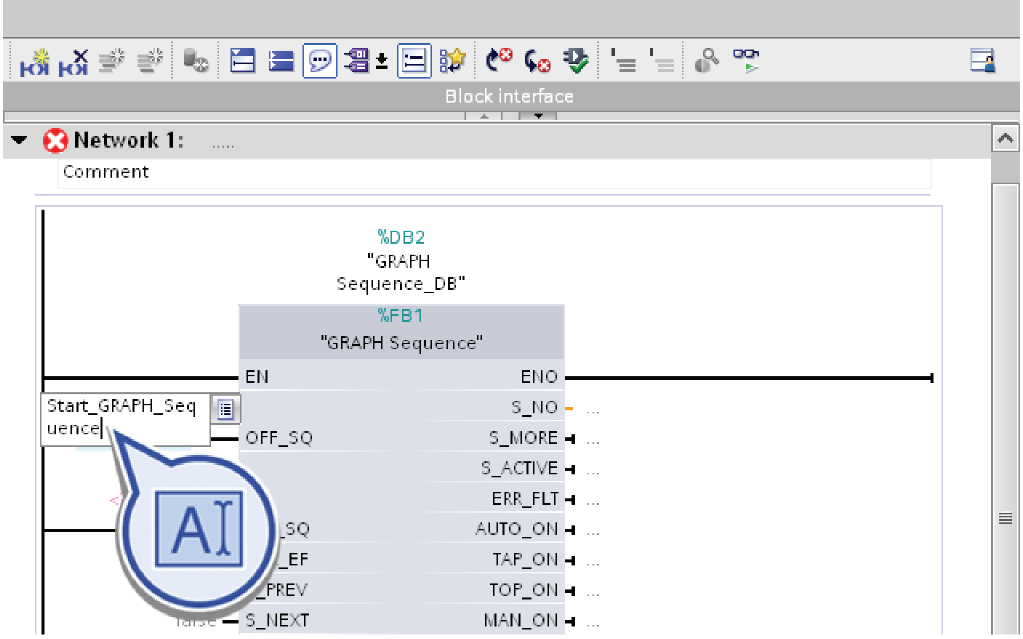
You use the NC contact at the parameter "OFF_SQ" to deactivate the execution of the sequencer. If the NC contact has the signal state "0", the sequencer is terminated and all steps are deactivated.

4. Link the input parameter "INIT_SQ":
   - Click the input.
   - Double-click the "NO contact" in the Task-Card

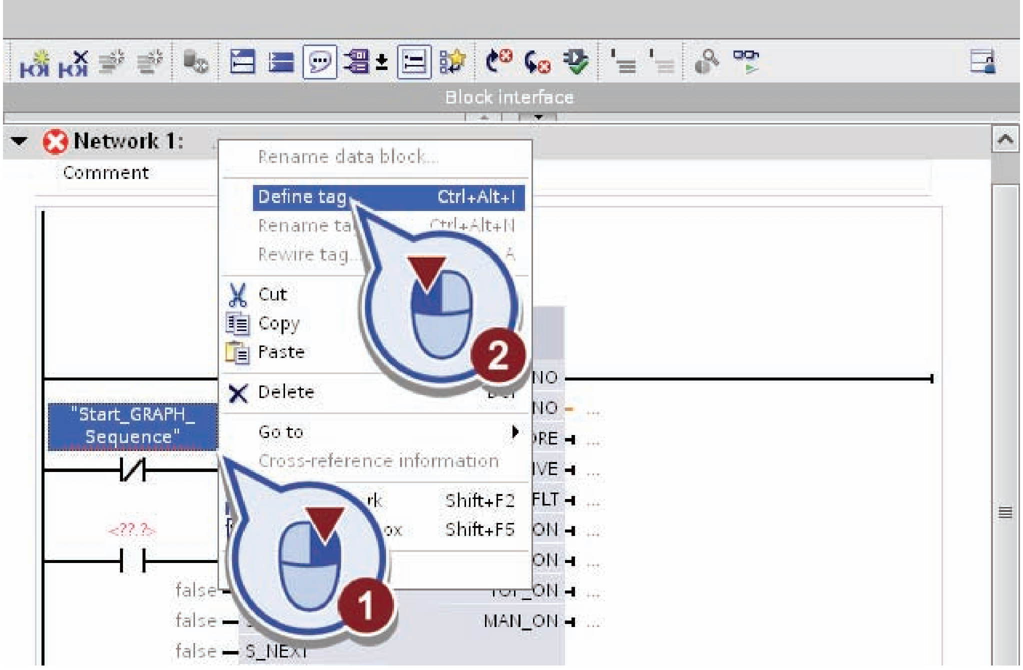


You use the NO contact at the parameter "INIT_SQ" to initialize the execution of the sequencer. If the NO contact has the signal state "1", the sequencer is reset and executed starting from the initial step "S1 Home".

5. Click the operand placeholder of the parameter "OFF_SQ" and enter "Start_GRAPH_Sequence" as tag name.



6. Right-click the text "Start_GRAPH_Sequence" and select "Define tag" from the shortcut menu.



7. Define the tag with the following properties:

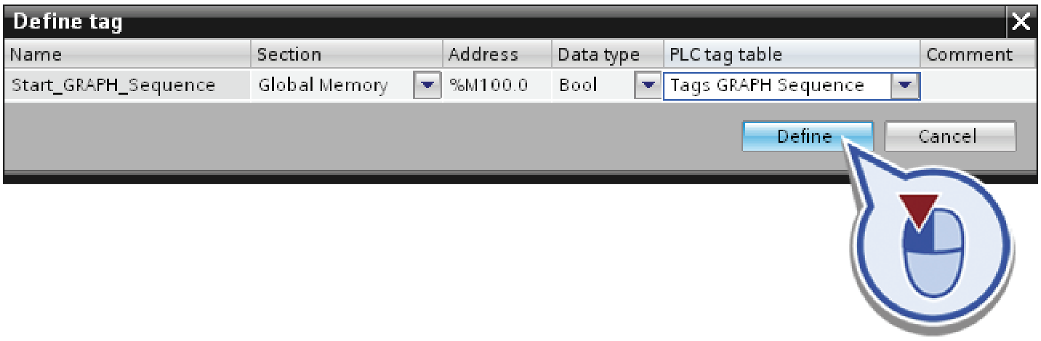
– Section: "Global Memory"

– Address: "M100.0"

– Date type: "Bool"

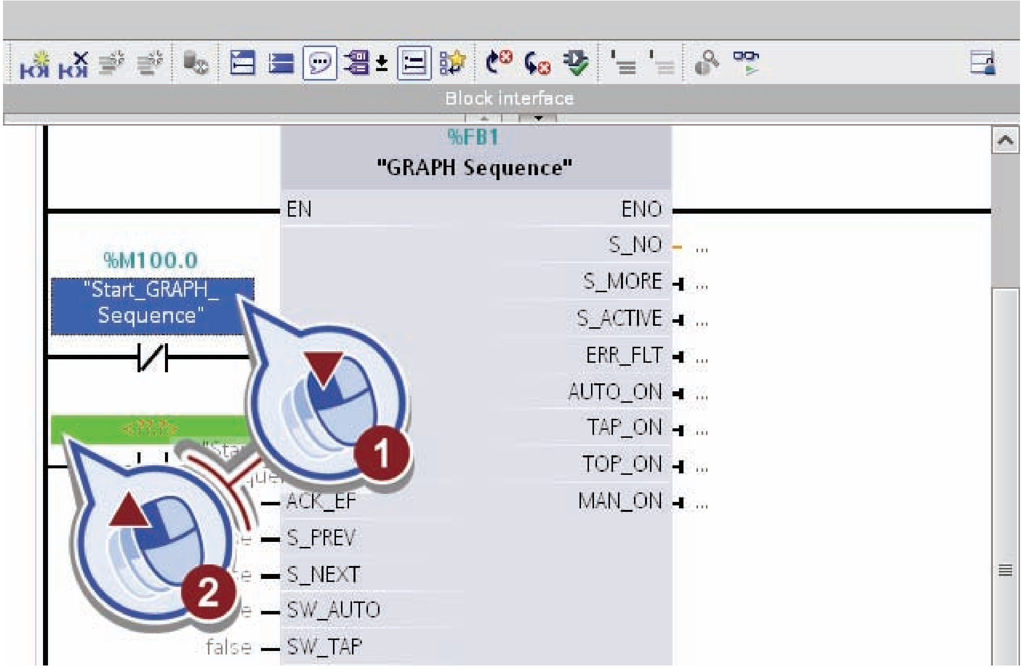
– PLC tag table: "Tags GRAPH Sequence"

Confirm the dialog by clicking "Define".


Define tag

Name	Section	Address	Data type	PLC tag table	Comment
Start_GRAPH_Sequence	Global Memory	%M100.0	Bool	Tags GRAPH Sequence	

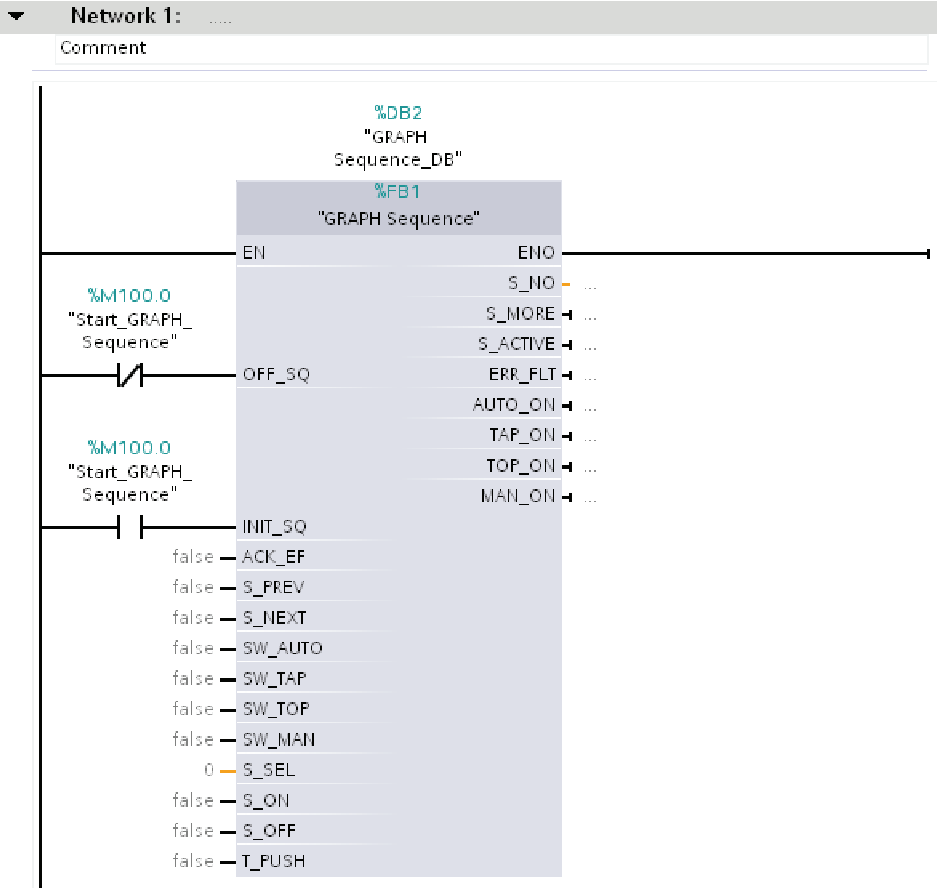
8. Click the tag "Start_GRAPH_Sequence" while pressing the <Ctrl> key, and drag the tag to the operand placeholder of the NO contact.



9. Save the project.

Result

You have successfully created the call for the GRAPH FB "GRAPH_Sequence" in the organization block "Main".



You can use the "Start_GRAPH_Sequence" tag to control the execution of the entire sequencer.

- If the signal state of the tag to is set to "0", the sequencer will be deactivated and the current program terminated. This happens regardless of which step is currently being processed.
- If the signal state of the tag is set to "1", the sequencer is initialized. This happens regardless of whether the sequencer has been activated for the first time or has been activated again after a deactivation.

Note

Additional sequencer control options

The GRAPH function block offers even more options for controlling the sequencer via appropriate input parameters. In the case of complicated production processes in particular, it is useful to control the initialization and deactivation of the sequencer and the interruption and restarting via the specific individual tags.

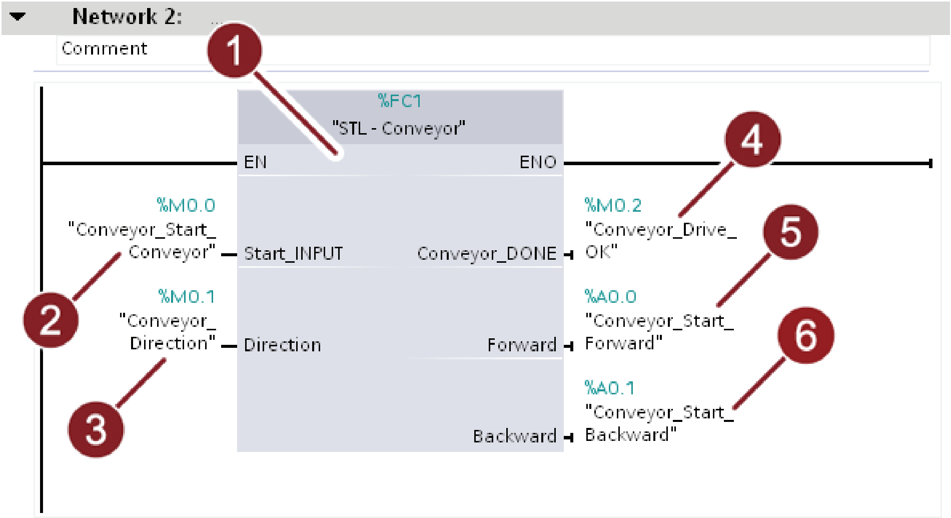
4.6.3 Calling the STL function

Introduction

In the following section you will call the STL function "STL-Conveyor" in the organization block "Main" and interconnect the input and output parameters.

Calling the STL function

The following figure shows the call of the STL function:



①	The function itself queries the direction specification of the conveyor belt and sets the output parameters accordingly.
②	If the signal state of the tag "Conveyor_Start_Conveyor" is set to "1" in the GRAPH interface, the condition stipulating that one of the two output parameters "Forward" or "Backward" has to be activated is satisfied in the function.
③	The direction of motion of the conveyor belt is specified using the tag "Conveyor_Direction", i.e. which of the two output parameters "Forward" or "Backward" is activated.
④	The tag "Conveyor_Drive_OK" is used to indicate whether the conveyor belt is currently activated. As long as the function is active, the output is reset. If the conveyor belt is not activated, the output is set.
⑤	You can use the tag "Conveyor_Start_Forward" to activate the output for the forward motion of the conveyor belt.
⑥	You can use the tag "Conveyor_Start_Backward" to activate the output for the backward motion of the conveyor belt.

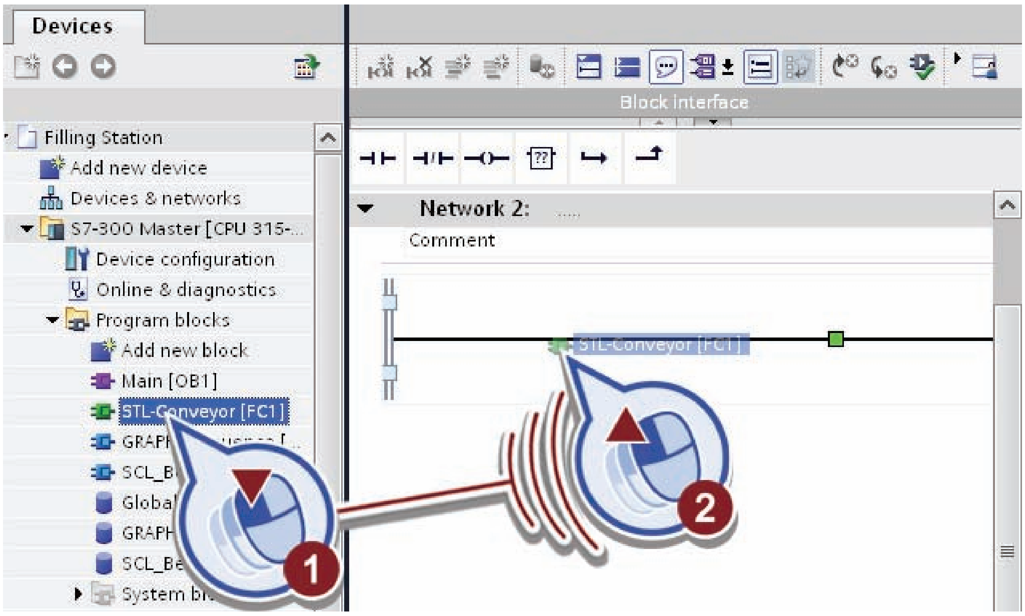
Requirement

You have programmed the program block "STL–Conveyor" and opened the organization block "Main".

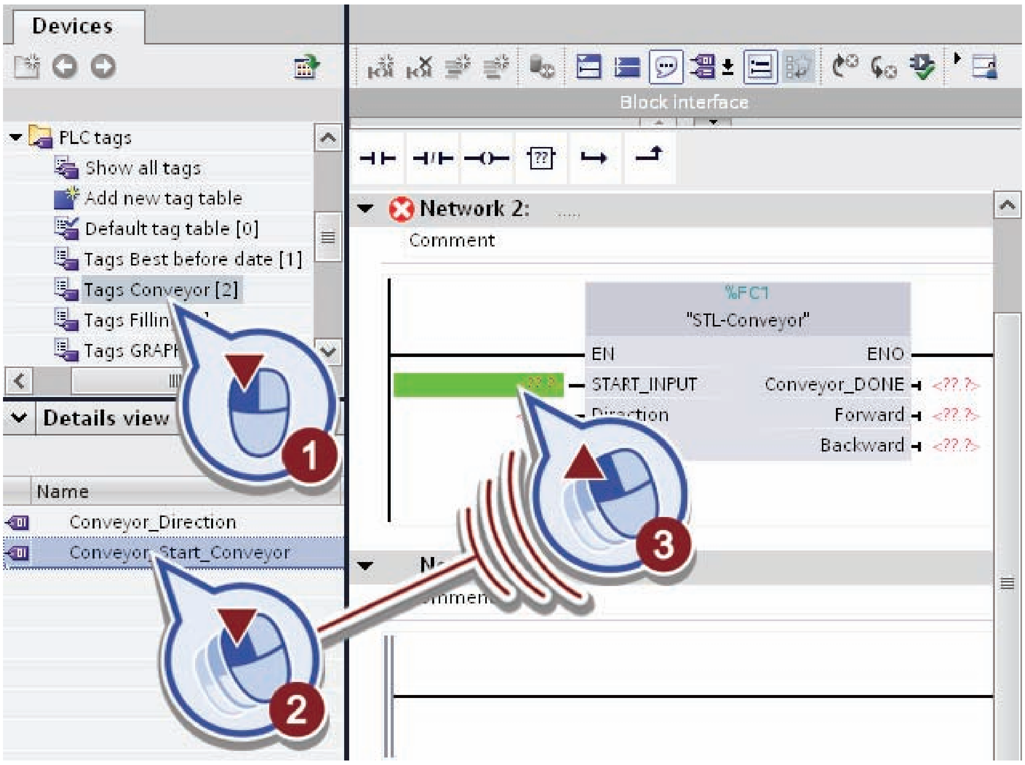
Procedure

To call the program block, proceed as follows:

1. Drag the STL function "STL–Conveyor" to network 2 of the organization block "Main".

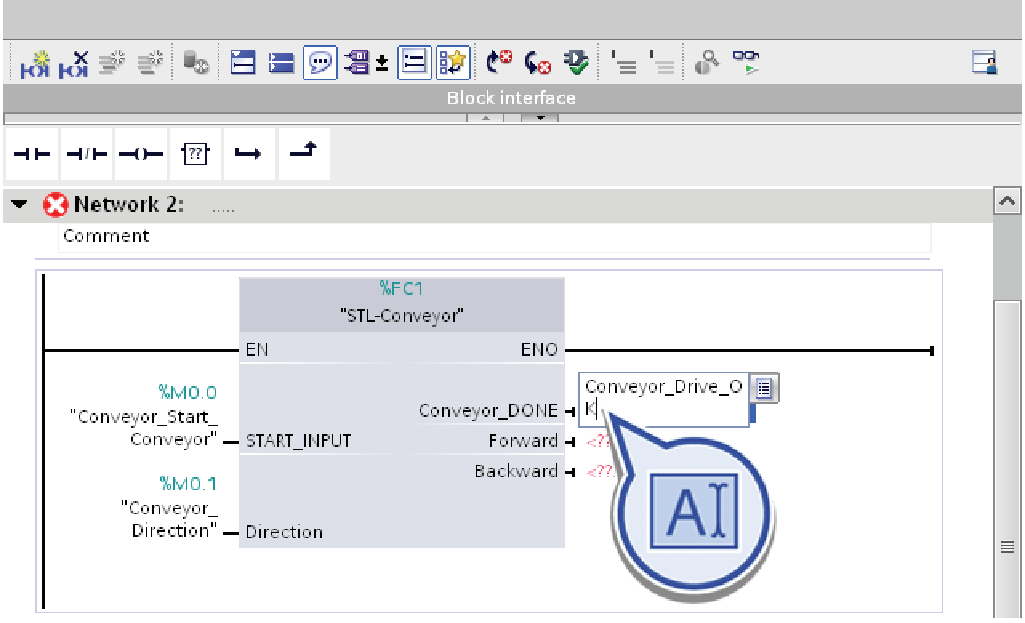


2. Link the input parameter "START_INPUT" by:
   - Opening the tag table "Tags Conveyor" in the details view.
   - Drag the tag "Conveyor_Start_Conveyor" from the details view to the operand placeholder of the parameter "START_INPUT".



3. As in the last step, link the input parameter "Direction" to the tag "Conveyor_Direction".

4. Click the operand placeholder of the parameter "Conveyor_DONE". Enter the text "Conveyor_Drive_OK".



5. To define the tag, select the operand placeholder and press the key combination <Ctrl+Alt+I>.

6. Define the tag with the following properties:

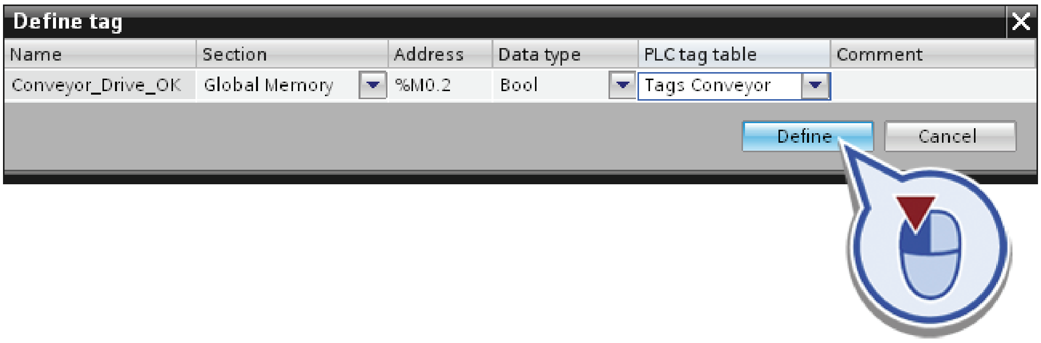
   – Section: "Global Memory"

   – Address: "M0.2"

   – Date type: "Bool"

   – PLC tag table: "Tags Conveyor"

   Confirm the dialog by clicking "Define".


Define tag

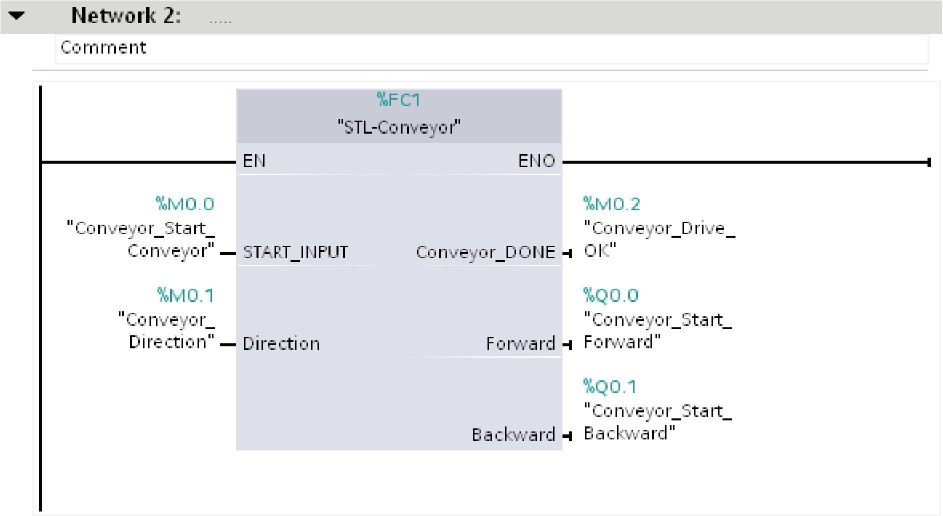
Name	Section	Address	Data type	PLC tag table	Comment
Conveyor_Drive_OK	Global Memory	%M0.2	Bool	Tags Conveyor	

7. Click the operand placeholder of the parameter "Forward". Enter the text "Conveyor_Start_Forward".

8. Define the tag with the following properties:
   – Section: "Global Output"
   – Address: "Q0.0"
   – Date type: "Bool"
   – PLC tag table: "Tags Conveyor"
9. Click the operand placeholder of the parameter "Backward". Enter the text "Conveyor_Start_Backward".
10. Define the tag with the following properties:
    – Section: "Global Output"
    – Address: "Q0.1"
    – Date type: "Bool"
    – PLC tag table: "Tags Conveyor"
11. Save the project.

Result

You have successfully inserted the call for the function "STL–Conveyor" in the organization block "Main".



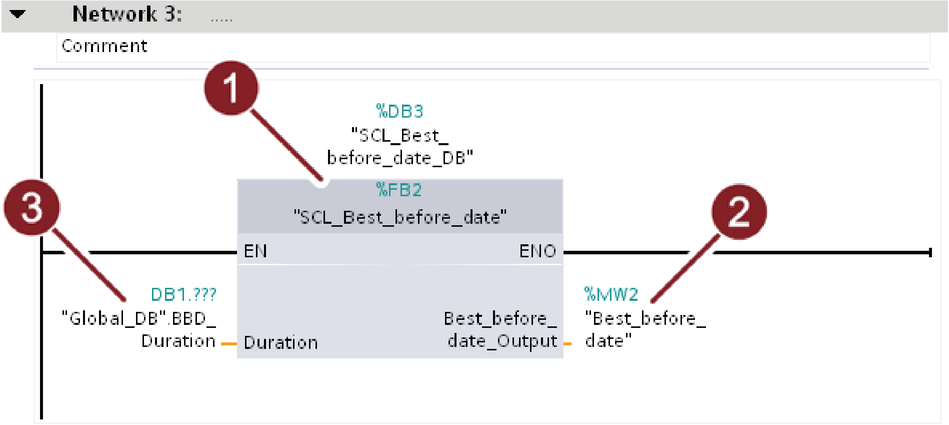
4.6.4 Calling the SCL function block

Introduction

In the following section you will call the SCL function block "SCL_Best_before_date" in the organization block "Main" and interconnect the input and output parameters.

Calling the SCL function block

The following figure shows the call of the SCL function block:



①	The SCL function block reads the system time of the CPU internally and calculates the year of the "best-before date" from the year of the current date and the entered best-before duration in years.
②	The year of the calculated best-before date is output as integer value on the output parameter. Store the calculated value store in the tag "Best_before_date".
③	The best-before duration in years is entered on the input parameter. You store the value for the best-before-date duration in the tag BBD_Duration.

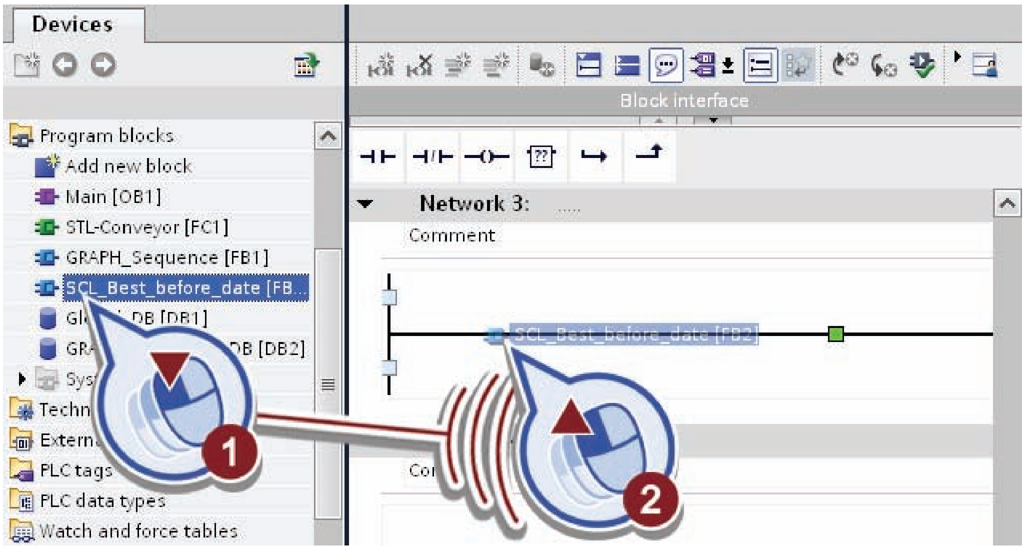
Requirement

You have programmed the Program block "SCL_Best_before_date" and opened the organization block "Main".

Procedure

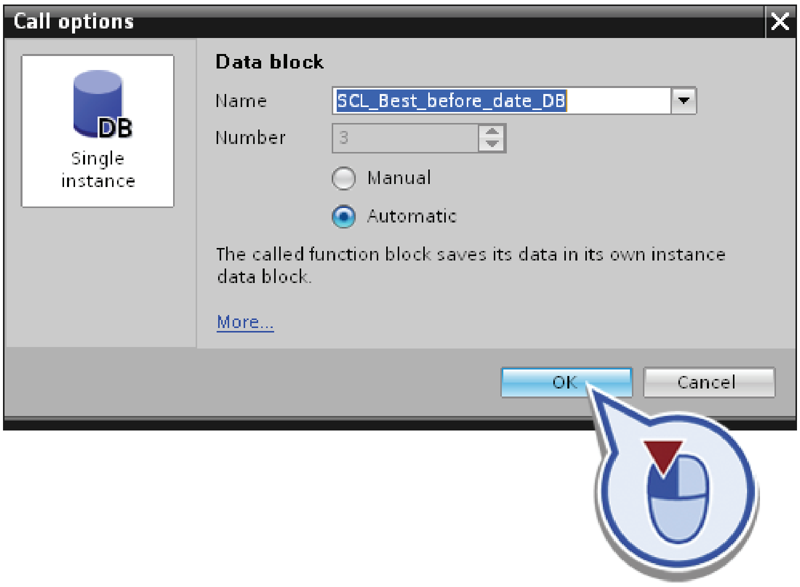
To call the program block, proceed as follows:

1. Drag the SCL function block "SCL_Best_before_date" to network 3 of the organization block "Main".



The "Call options" dialog box appears.

2. Click "OK" to confirm the creation of the instance data block.



Data block

Name: SCL_Best_before_date_DB

Number: 3

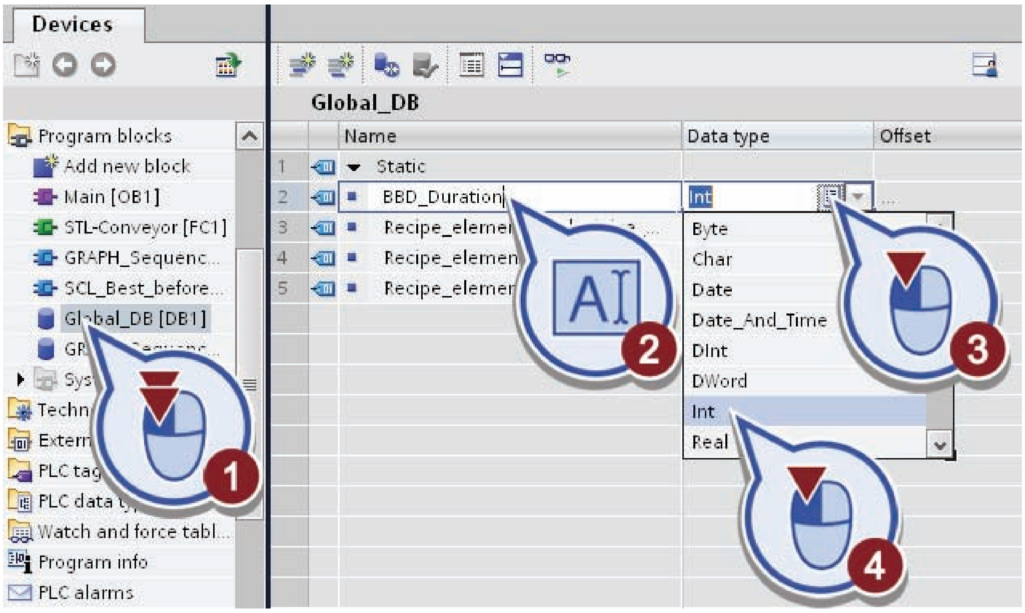
○ Manual

● Automatic

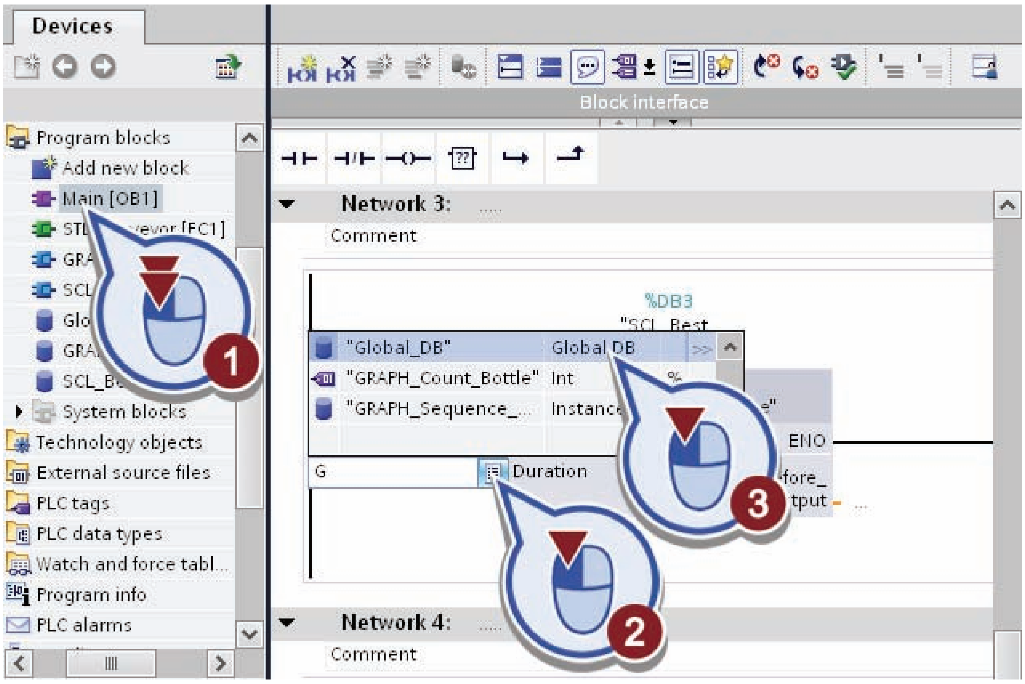
The called function block saves its data in its own instance data block.

More...

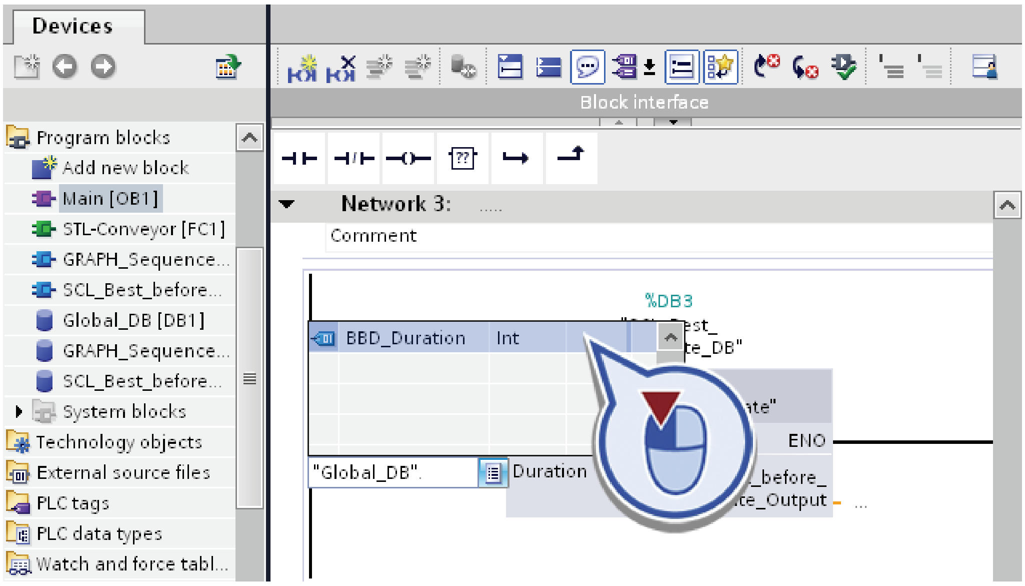
3. Double-click the data block "Global_DB" and define the integer tag "BBD_Duration".



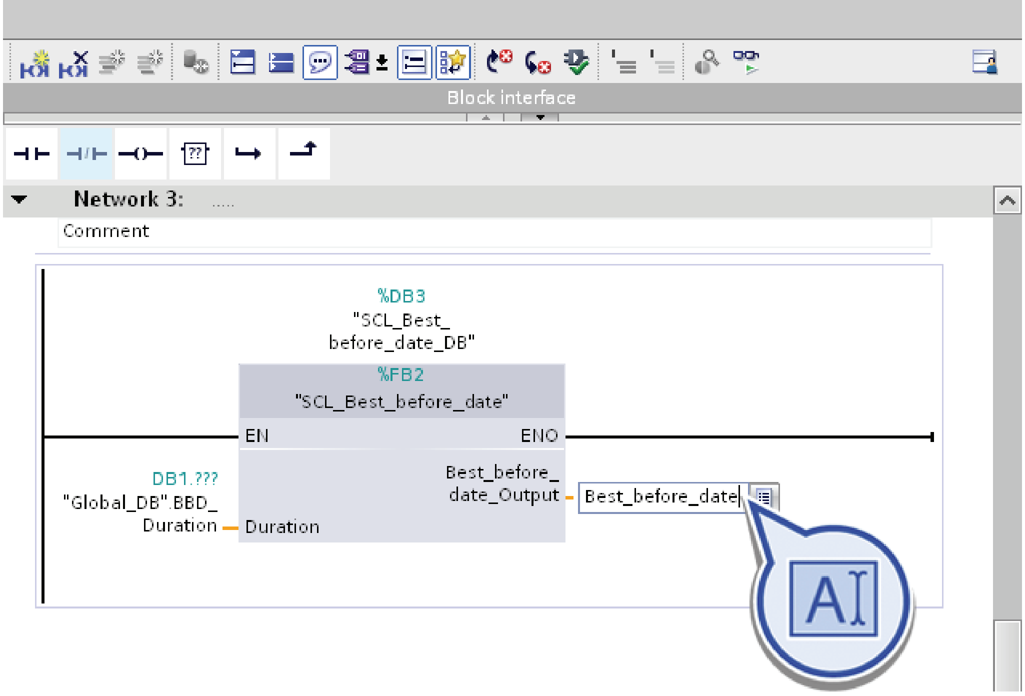
4. Click the input parameter "Duration" in the organization block "Main" and select the "Global_DB".



5. Assign the integer tag "BBD_Duration" to the input parameter.



6. Enter the text "Best_before_date" in the operand placeholder on the output parameter "Best_before_date_Output".

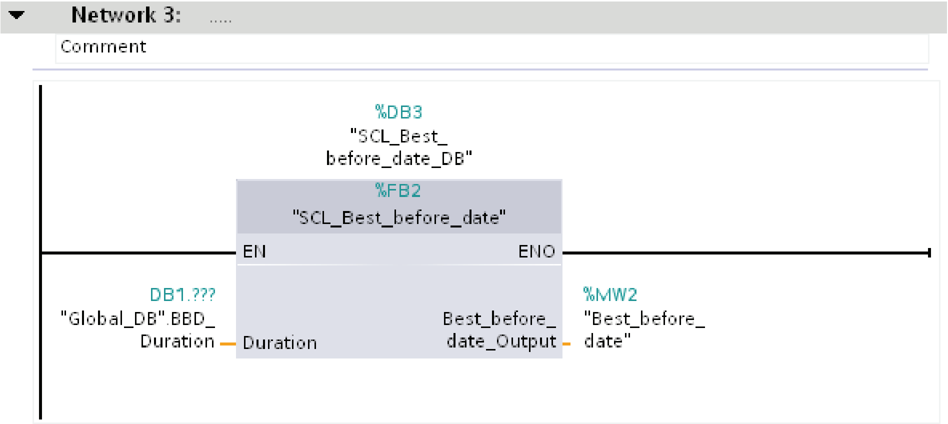


7. To define the tag, select the operand placeholder with the text "Best_before_date" and press the key combination <Ctrl+Alt+I>.

8. Define the tag with the following properties:
   - Section: "Global Memory"
   - Address: "MW2"
   - Date type: "Int"
   - PLC tag table: "Tags Best before date"
9. Save the project.

Result

You have successfully inserted the call for the program block "SCL_Best_before_date" in the organization block "Main".



Visualizing the process 5

5.1 Basics principles of HMI

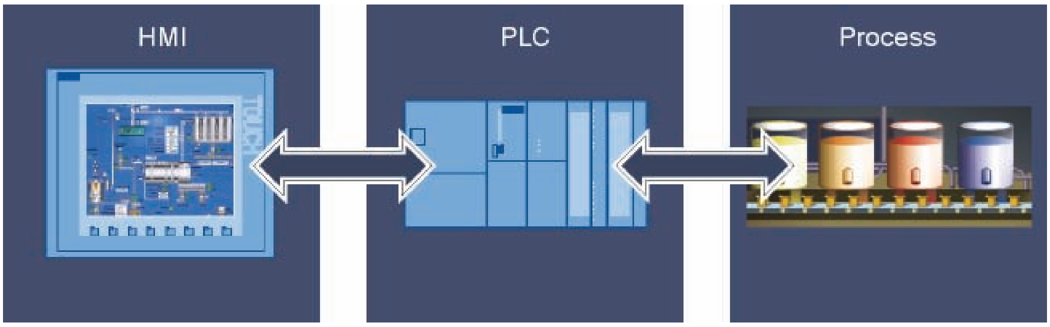
Introduction

In the "Filling Station" example project you create several HMI screens. You use these screens to visualize the entire production process, which you have programmed in the previous chapter.

To create the three required screens, pre-defined objects are provided, which allow you to simulate the production process. They enable you to display the process flow and enter the process values. The functions of the "TP900 Comfort" HMI device determine the visualization options of the project and the functional scope of the graphic objects.

Definition of Human Machine Interface (HMI)

The Human Machine Interface (HMI) system represents the interface between the user and the process. The process flow is controlled by the CPU. The operator can use the HMI device to monitor the process or intervene in the active process.



The following are some of the options provided for operating and monitoring machines and plants:

- Display processes
- Control processes
- Generate alarms
- Manage process parameters and recipes

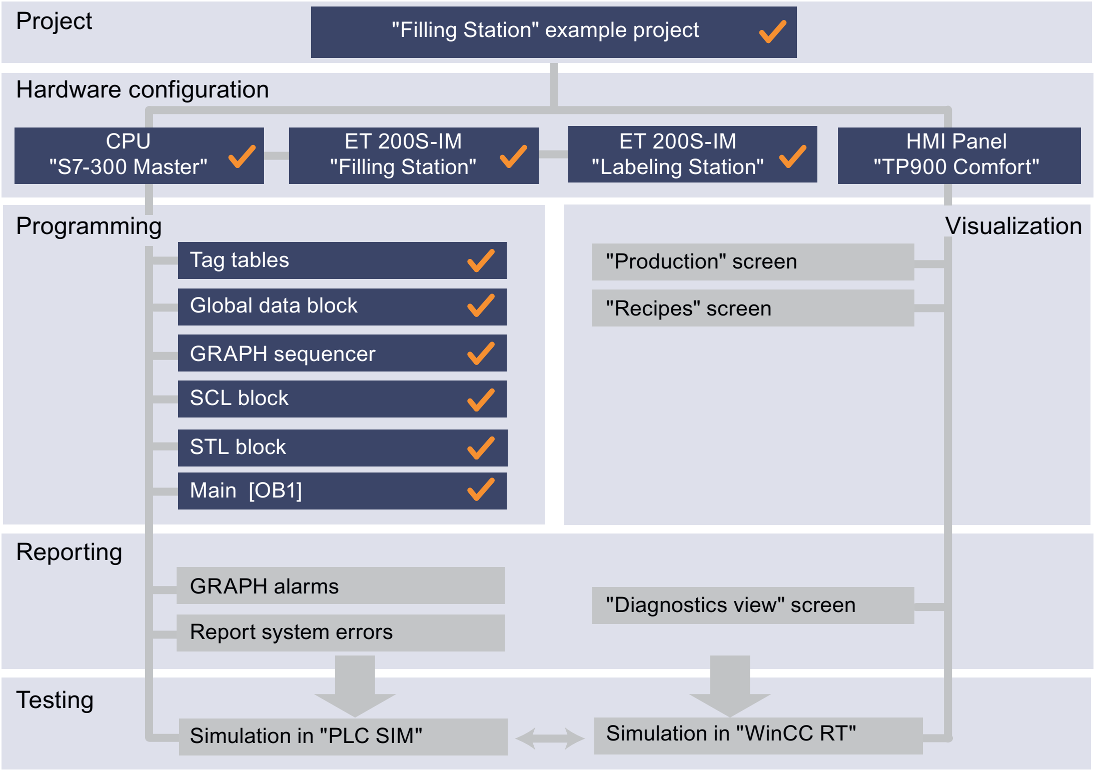
5.2 Configuring the HMI Comfort Panel

Introduction

In the following, you will create the "TP900 Comfort" HMI panel and use the HMI device wizard to create the configuration and a template for the HMI screens. You can use these HMI screens to visualize each phase of the filling process in the next steps.

Progress of progress

The following graphic shows you the configuration step you need to take:



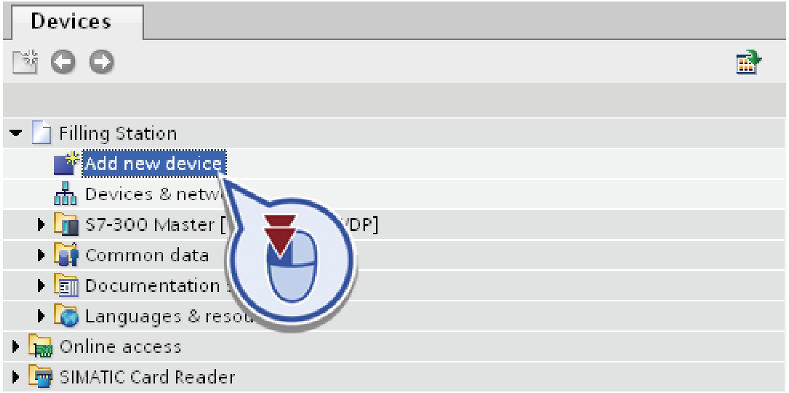
Requirement

You have created the "Filling Station" project and the "S7-300 Master" CPU.

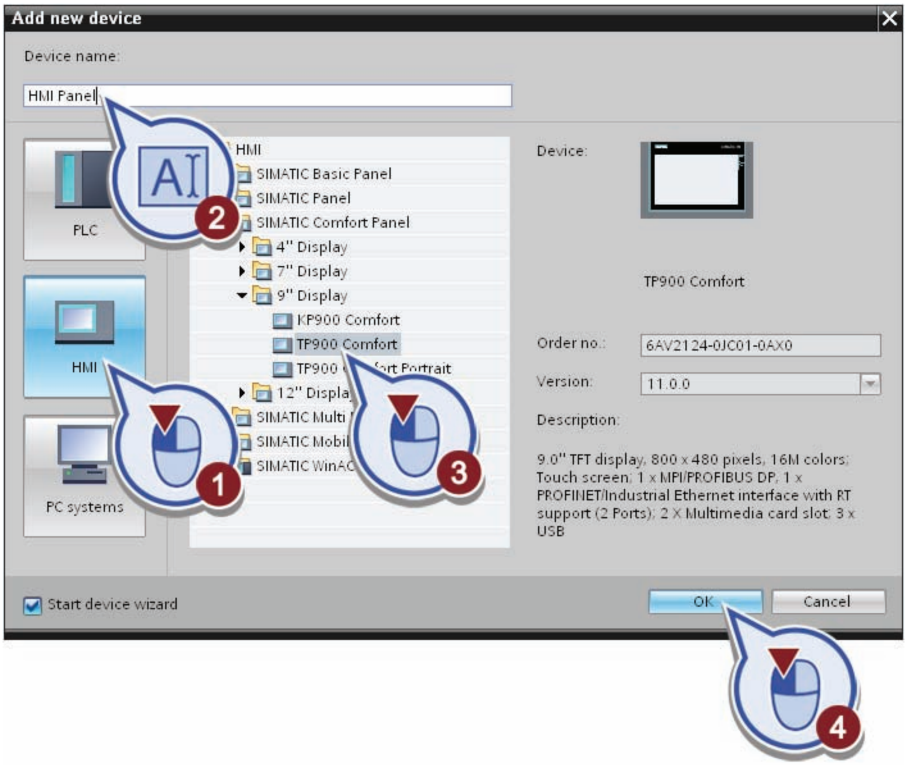
Procedure

To create the HMI panel, follow these steps:

1. Double-click "Add new device" in the project tree.



2. Make the following settings in the "Add new device" dialog:
   – Click "HMI".
   – Enter "HMI panel" as the device name.
   – Select the HMI panel "TP900 Comfort".
   – Check that the "Start device wizard" function is activated and confirm the creation of the HMI panel with "OK".



Device name: HMI Panel

Order no.: 6AV2124-0JC01-0AX0

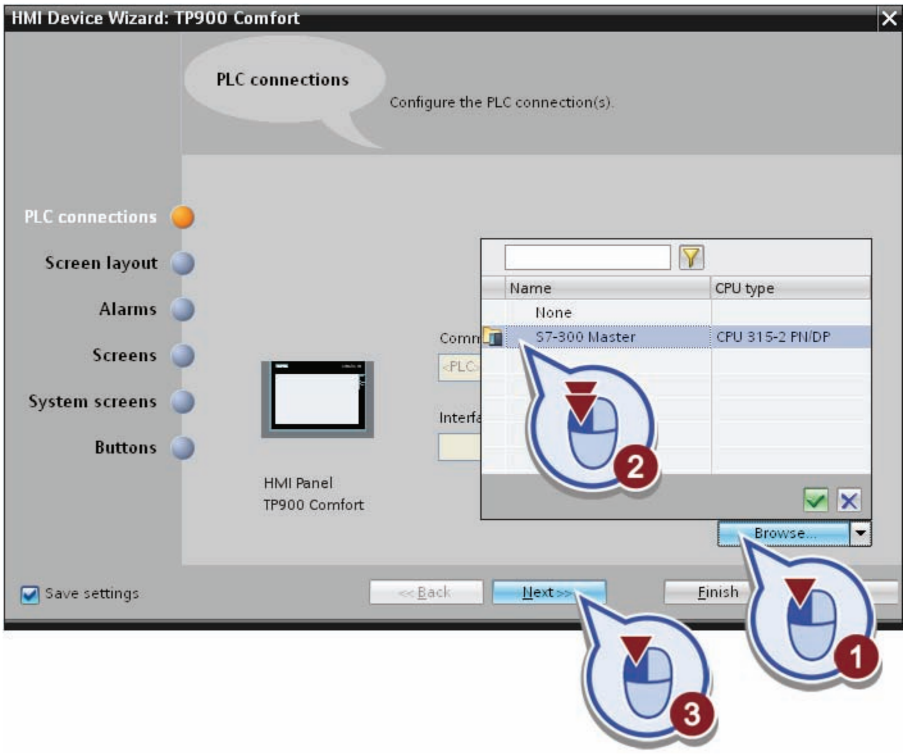
Version: 11.0.0

Description: 9.0" TFT display, 800 x 480 pixels, 16M colors; Touch screen; 1 x MPI/PROFIBUS DP, 1 x PROFINET/Industrial Ethernet interface with RT support (2 Ports); 2 X Multimedia card slot; 3 x USB

☑ Start device wizard

The "HMI Device Wizard" dialog opens.

3. Configure the connection from the CPU to the HMI panel as follows:
   - Click "Browse".
   - Select the "S7-300 Master" CPU.
   - Open the next dialog window with "Next".



HMI Device Wizard: TP900 Comfort

PLC connections

Configure the PLC connection(s).

Name	CPU type
None	
S7-300 Master	CPU 315-2 PN/DP

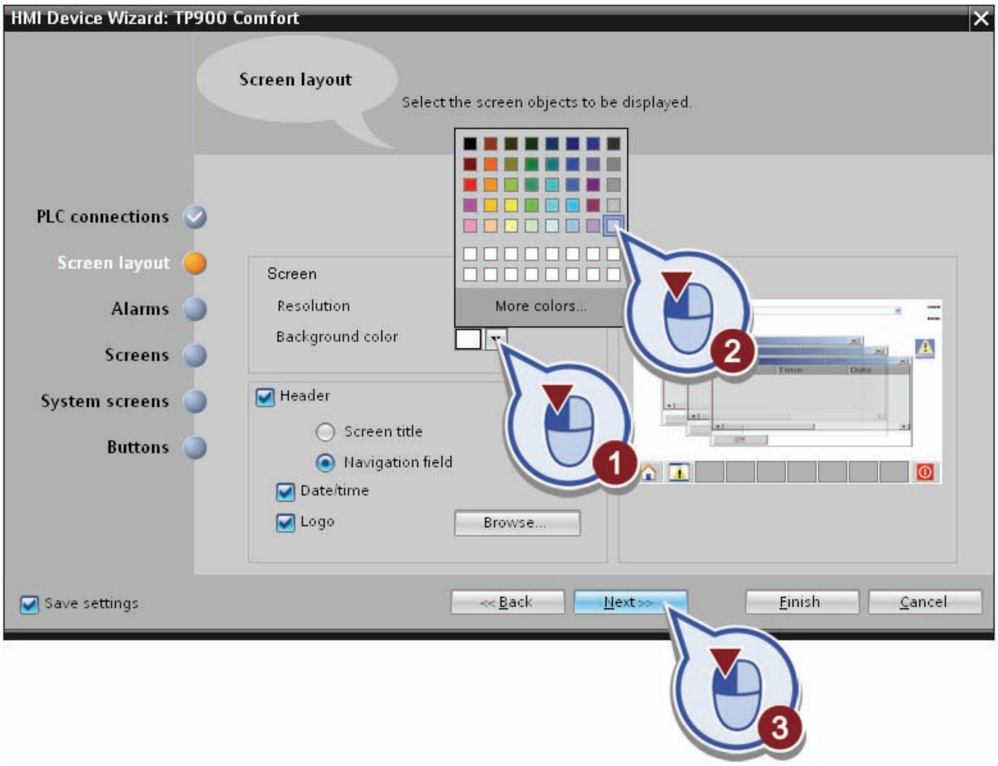
Note

Configuring the connection to the CPU later

You can also configure the connection between the HMI Comfort Panel and the CPU later under "Devices & Networks".

---

4. Select the background color for the HMI screens as follows:
   - Open the color palette.
   - Select white as the color.
   - Open the next dialog window with "Next".



HMI Device Wizard: TP900 Comfort

Screen layout

Select the screen objects to be displayed.

Screen

Resolution

Background color

☑ Header

○ Screen title

● Navigation field

☑ Date/time

☑ Logo

☑ Save settings

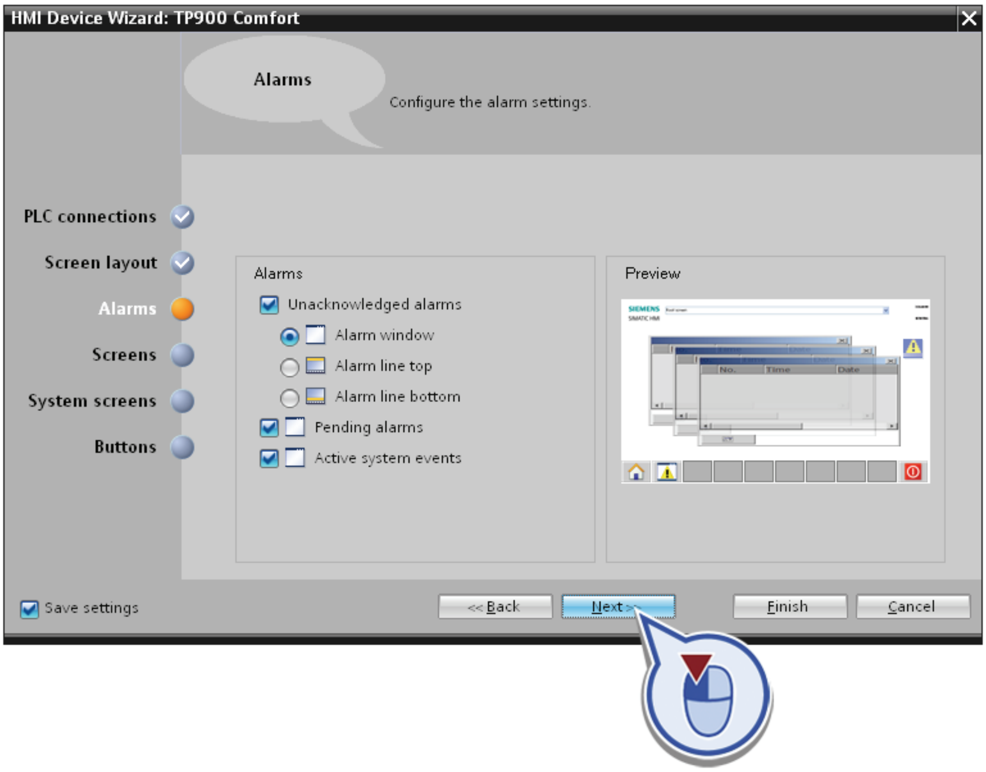
Note

Subsequently changing the display

You can change the display settings made here later in the template of the HMI screen.

---

5. Ensure that the displayed settings are activated in the "Alarms" dialog and click "Next".



HMI Device Wizard: TP900 Comfort

Alarms

Configure the alarm settings.

Alarms

☑ Unacknowledged alarms

● Alarm window

○ Alarm line top

○ Alarm line bottom

☑ Pending alarms

☑ Active system events

Preview

☑ Save settings

Note

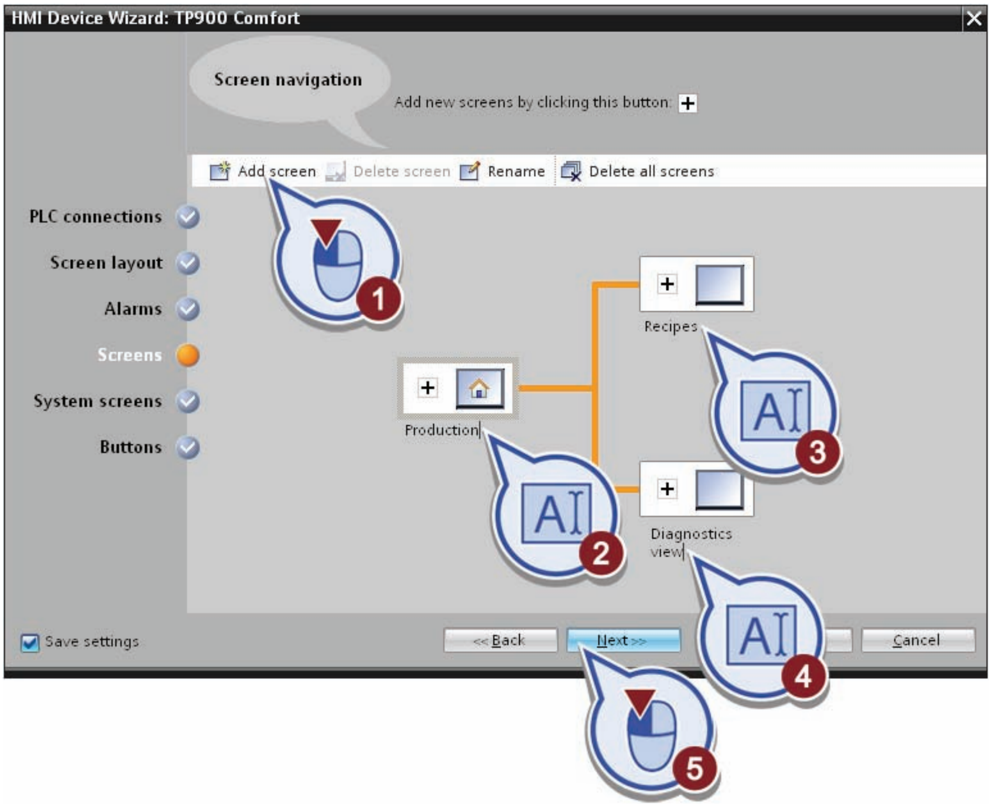
Alarms

Alarms can be supplemented by additional information, for example, to make it easier to localize faults in the system. A basic distinction is made between user-defined alarms and system events:

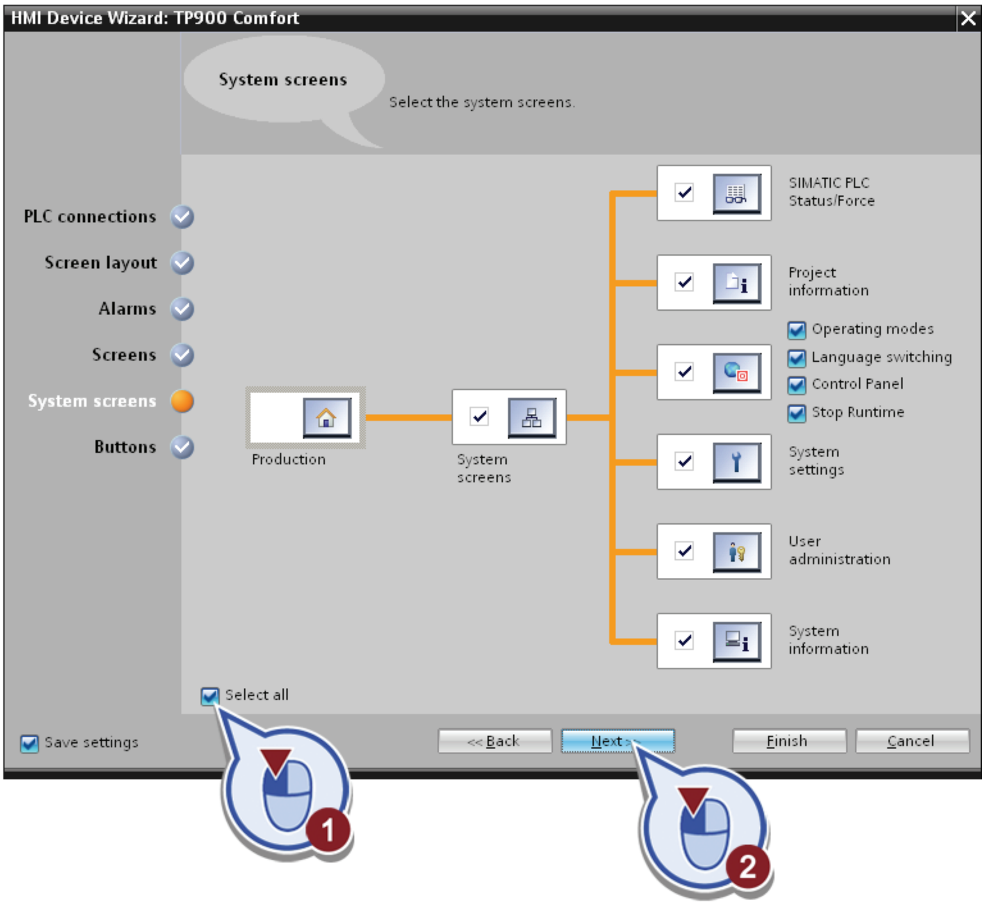
- User-defined alarms are used to monitor the machine process.
- System events are imported into the project and contain status information of the HMI device used.

---

6. Create the screen navigation as follows:
   - Double-click "Add screen". If the button is not enabled, select the previously available root screen.
   - Rename the screens as follows: "Production", "Recipes" and "Diagnostics view".
   - Open the next dialog window with "Next".



7. Select the "Select all" option in the "System screens" dialog. Click "Next".

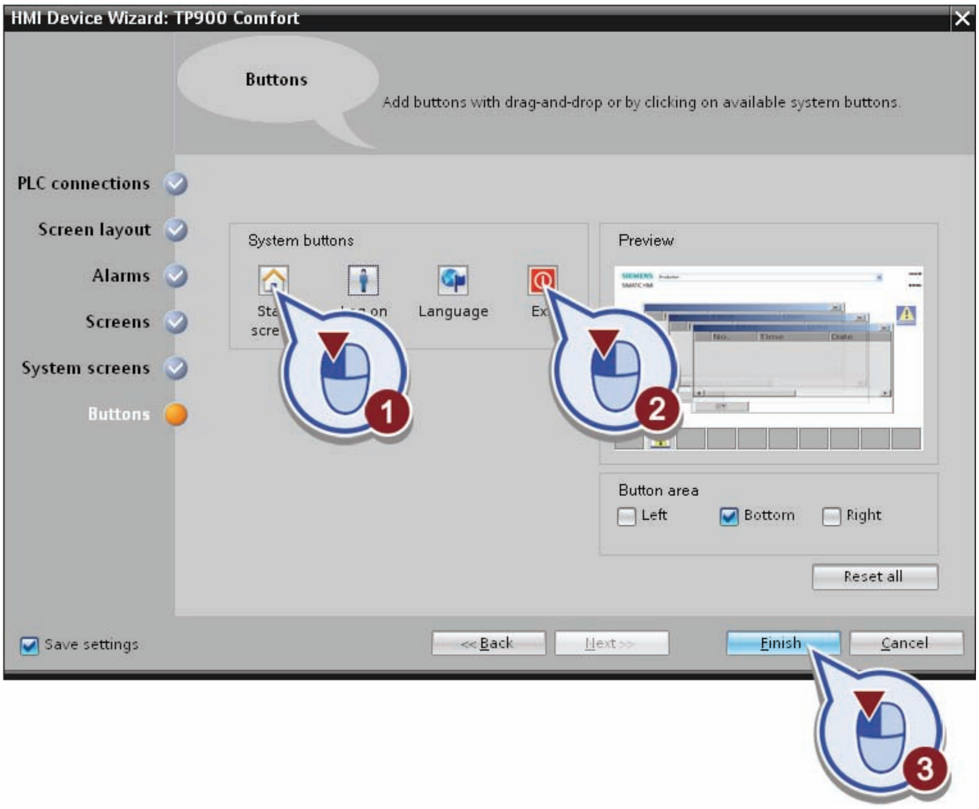


Note

System screens

You can use the system screens to set up project, system and operating information as well as the user administration as HMI screens. The buttons for navigating between the root screen "Production" and the system screens are created automatically.

8. Add the "Start screen" and "Exit" buttons and close the configuration of the HMI panel by clicking "Finish".



HMI Device Wizard: TP900 Comfort

Buttons

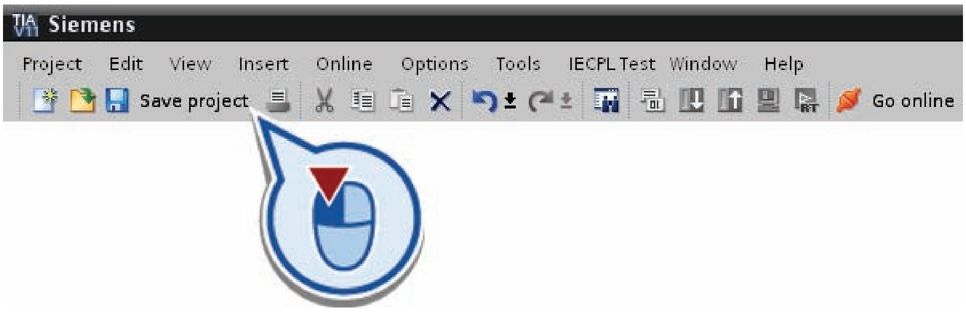
Add buttons with drag-and-drop or by clicking on available system buttons.

Button area

□ Left ☑ Bottom □ Right

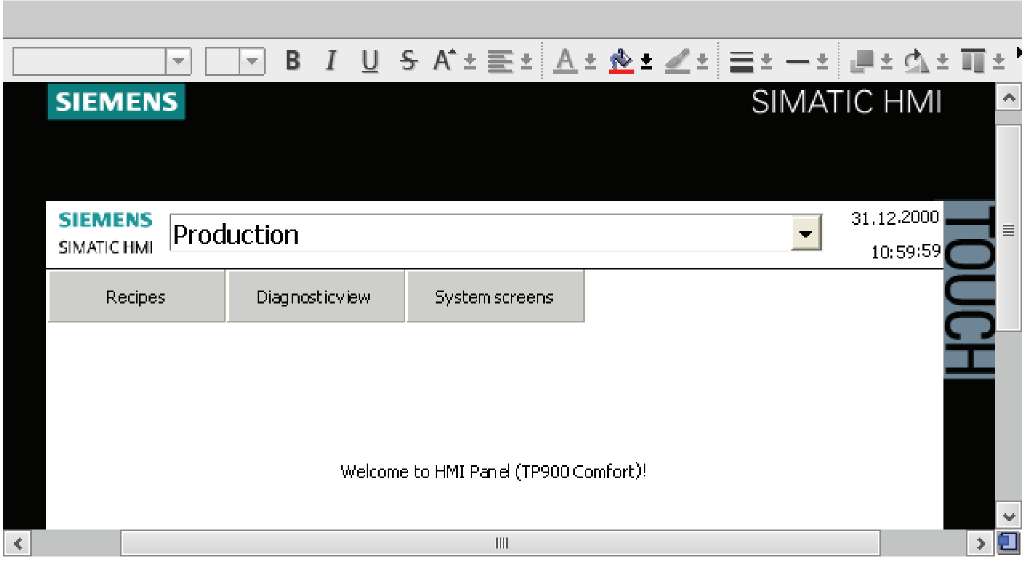
☑ Save settings

9. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.



Result

You successfully created the HMI Comfort Panel and the template for the HMI screens. Once you close the HMI Device Wizard, the "Production" root screen opens automatically.



The root screen is the initial screen that opens when the runtime software is launched. It is highlighted by a green arrow in the "Screens" folder in the project tree. You can later navigate in runtime from the root screen to the other HMI screens. The other HMI screens you have created using the HMI Device Wizard are also displayed in the "Screens" folder in the project tree.

5.3 Creating the "Production" root screen

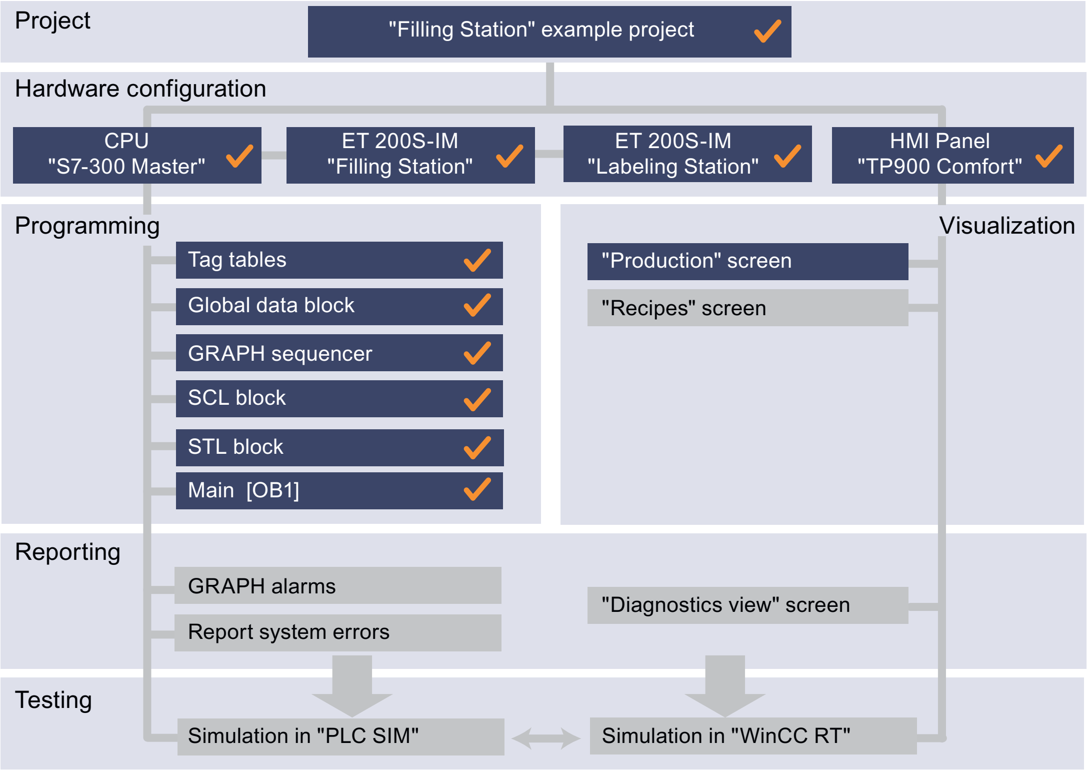
5.3.1 Overview

Introduction

In the following section, you will create the first HMI screen, the "Production" root screen. This screen shows which step of the GRAPH sequencer is currently running. Accordingly, in the following section you will define for the steps the individual screen objects that show the current processing state in the HMI panel.

Project progress

The following graphic shows you which step you perform next:



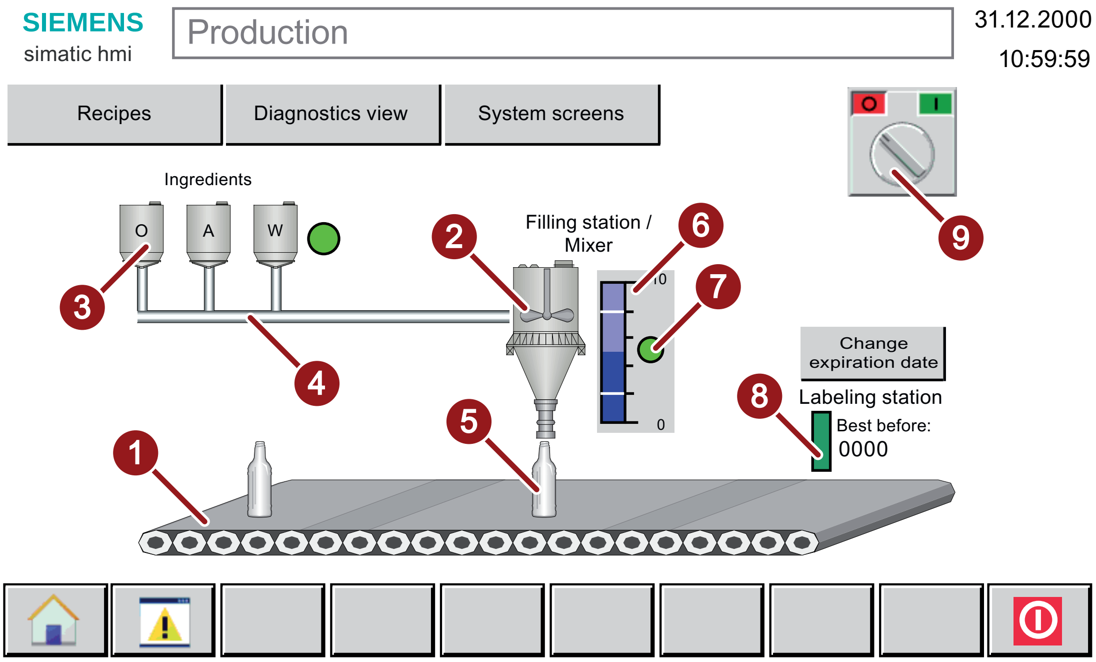
Static and dynamic screen objects

You use static and dynamic elements to implement the "Production" root screen:

- Static screen objects, such as the beverage tanks, pipelines or the labels, do not change in runtime. The appearance in the screen is independent of the processing state of the program.
- Dynamic screen objects, such as a bar graph or the bottles change depending on the process values. The dynamic screen objects are always associated with process values of the controller or with internal tags of the HMI panel. Depending on the linked values, the form or the location of each screen object also changes. The appearance in the screen is therefore dependent on the processing state of the program.

Screen objects of the "Production" root screen

The figure shows an overview of the "Production" root screen objects, which you create one after the other in the next section.



①	Conveyor belt
②	Filling station with mixer
③	Beverage tanks
④	Pipelines
⑤	Bottles on the conveyor belt
⑥	Bar graph
⑦	Indicator lights
⑧	Labeling machine
⑨	Switch to activate the sequencer

5.3.2 Visualizing the conveyor belt

Introduction

In the following, you will insert the first graphic object, the conveyor belt, in the "Production" root screen.

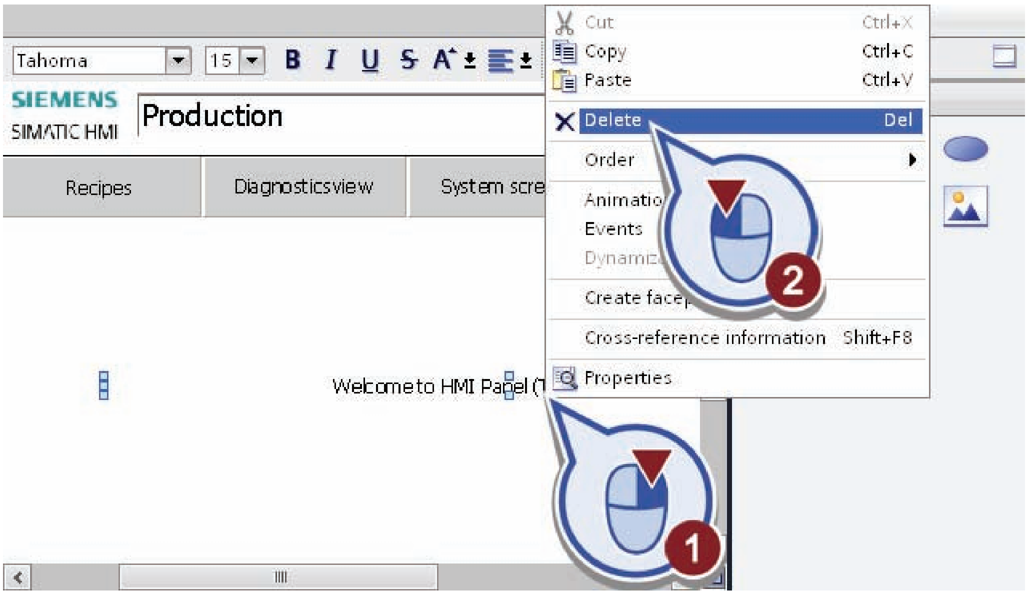
Requirement

You have created the "Production" root screen.

Procedure

To insert the graphic, follow these steps:

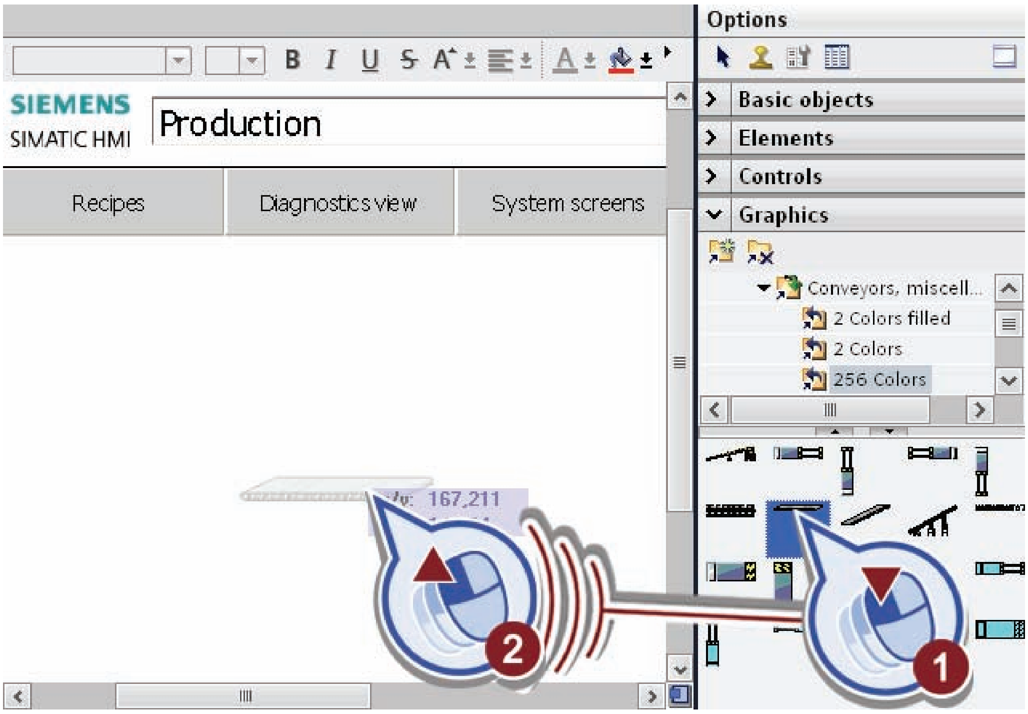
1. Right-click the text "Welcome to...". Select "Delete" from the shortcut menu.



2. Open the "Graphics" palette in the "Toolbox" task card.

   The WinCC graphics folder contains several graphics and is sorted thematically according to areas of application.

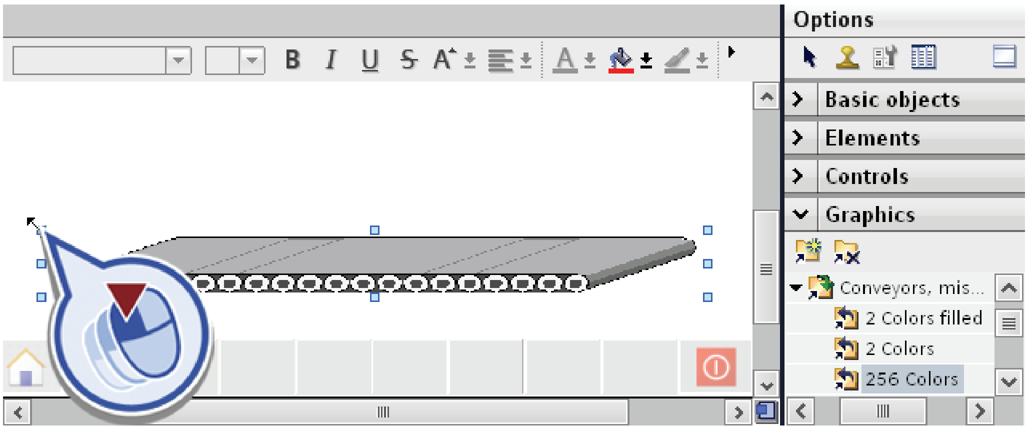
3. Add a graphic for the conveyor belt:
   - Open the directory "WinCC graphics folder" > "Automation equipment" > "Conveyors, miscellaneous".
   - Click on the "256 Colors" folder.
   - Drag the "Horizontal conveyor with perspective" graphic into the "Production" root screen.



Note

To display the names of the graphics in the folder, right-click in the listed graphics and clear the "Large icons" check box.

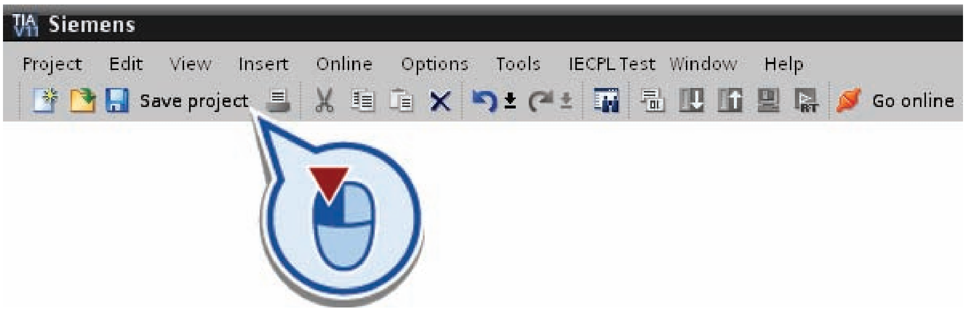
4. Position the conveyor belt in the center of the lower section of the screen and scale it to two-thirds of the total screen width.



Note

Make sure that screen objects do not protrude beyond the screen area, otherwise the graphics will not be displayed in runtime.

5. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.


Siemens
Project Edit View Insert Online Options Tools IECPL Test Window Help
Save project
Go online

Result

You have successfully inserted the conveyor belt in the "Production" root screen.

5.3.3 Visualizing the filling station with mixer

Introduction

In the following, you will add a graphical representation of the filling station to the "Production" root screen.

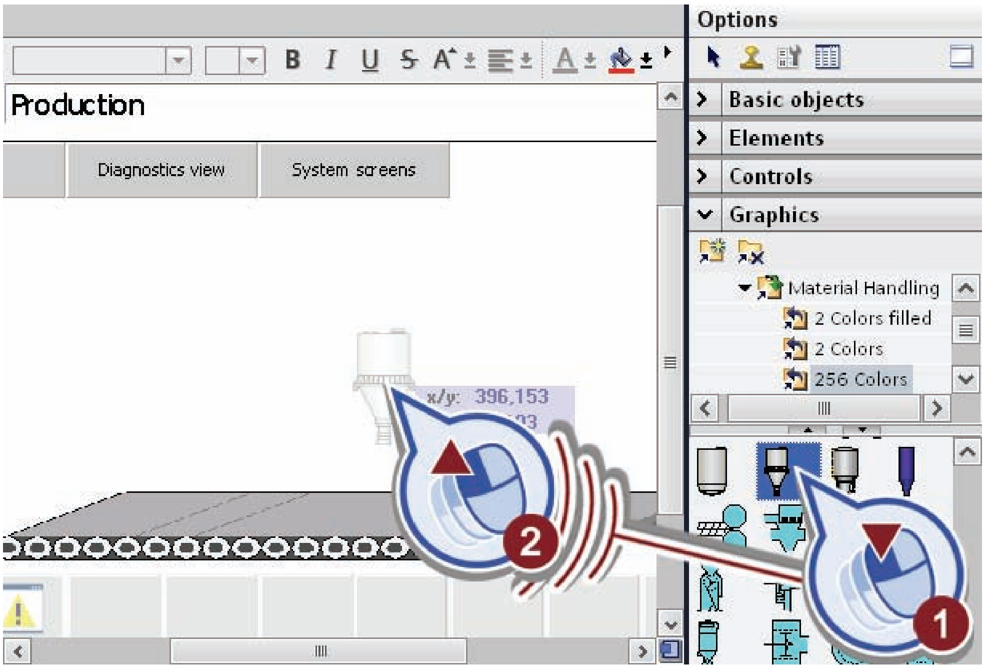
Requirement

You have created the "Production" root screen and the "GRAPH_Sequence_DB" instance data block of the GRAPH sequencer.

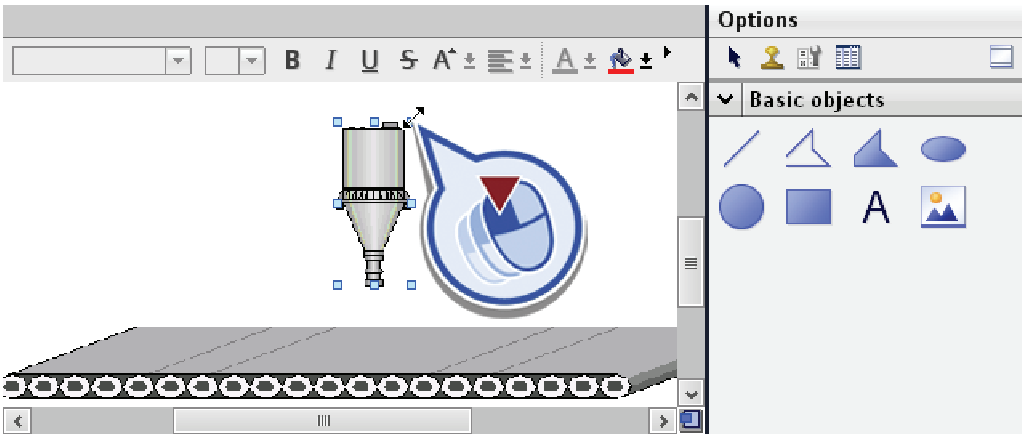
Procedure

To visualize the filling station, follow these steps:

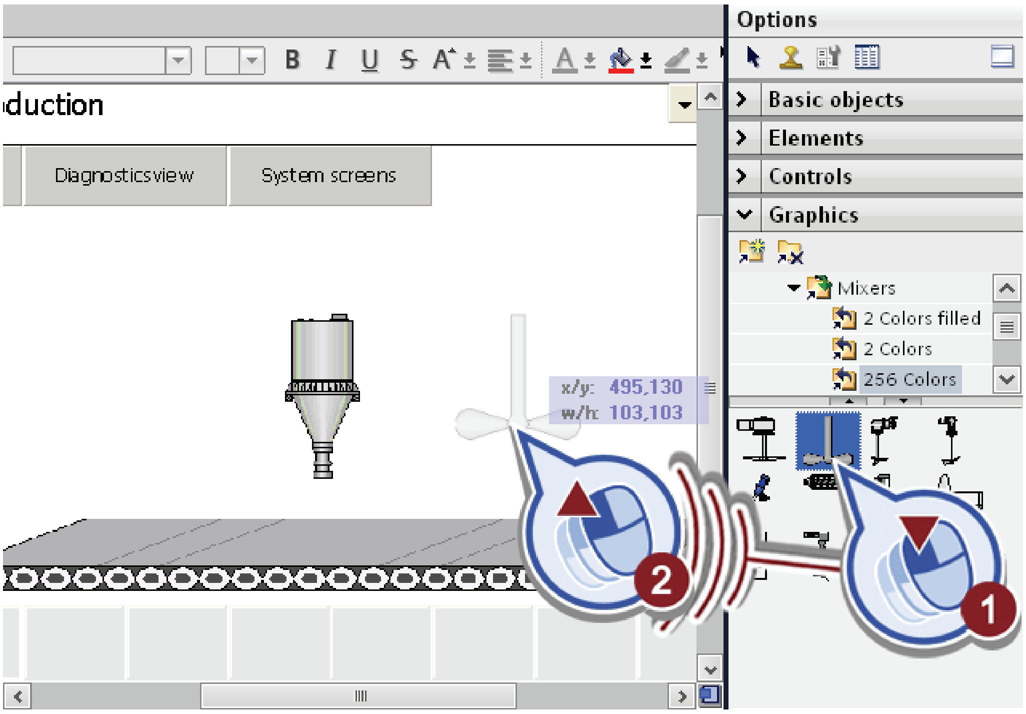
1. Insert the "Silo2" graphic as follows:
   - Open the directory "WinCC graphics folder" > "Industries" > "Material Handling".
   - Click on the "256 Colors" folder.
   - Drag the "Silo2" graphic into the "Production" root screen.



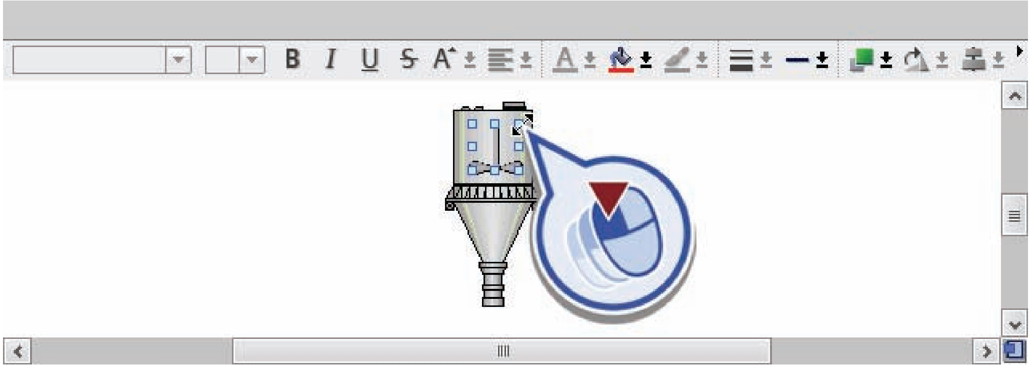
2. Position the graphic in the center above the conveyor belt and scale it to half its original size.



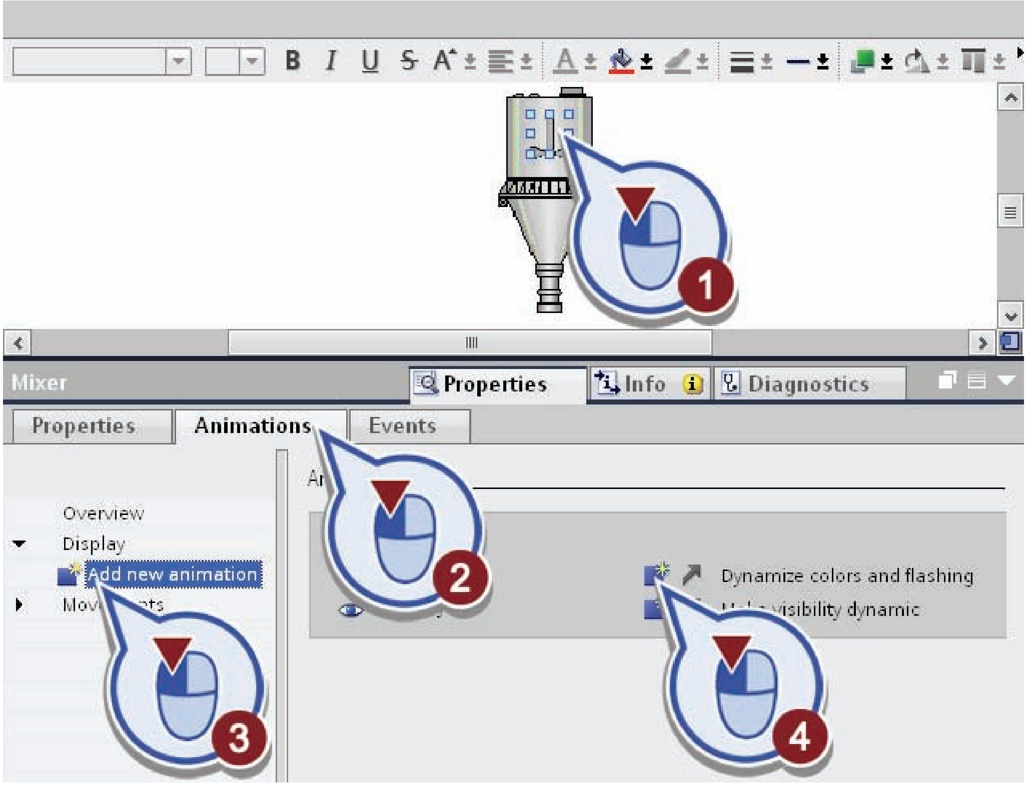
3. Insert the "Mixer blade" graphic as follows:
   - Open the directory "WinCC graphics folder" > "Automation equipment" > "Mixers".
   - Click on the "256 Colors" folder.
   - Drag the "Mixer blade" graphic into the "Production" root screen.



4. Scale the graphic to the width of the filling station and place it on the filling station.

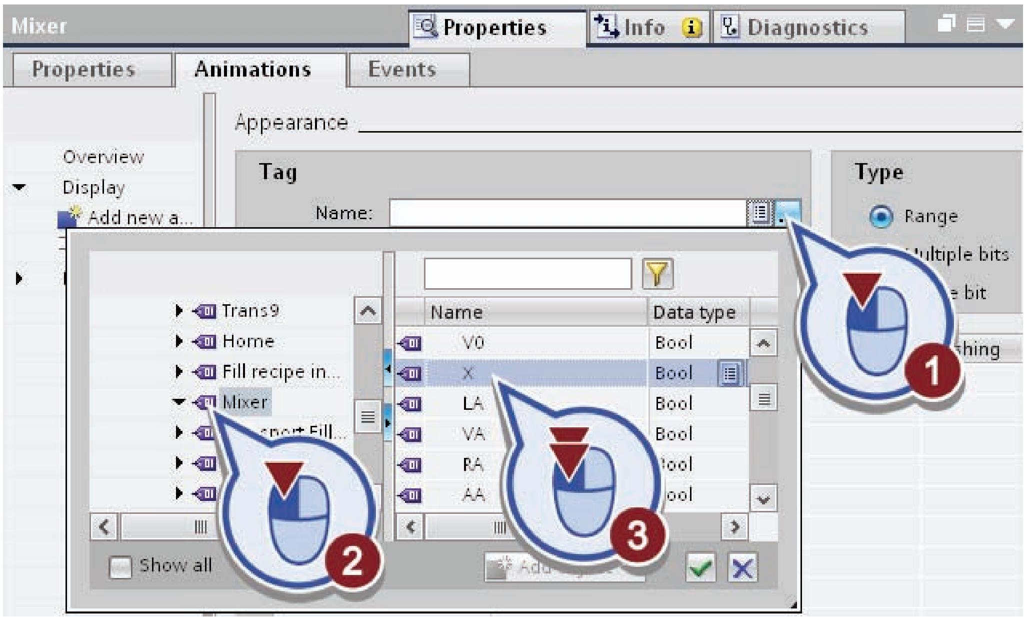


5. Animate the mixer as follows:
   - Select the graphic.
   - Open the "Animations" tab.
   - Click on "Add new animation" in the "Display" folder.
   - Click on the "Dynamize colors and flashing" function in the "Animations" dialog.



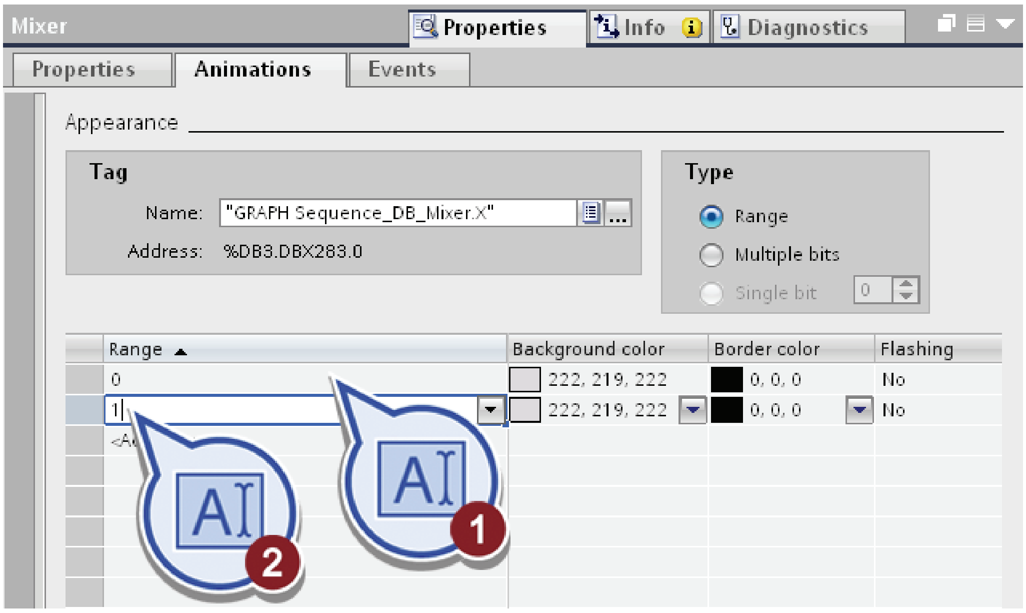
The "Appearance" dialog opens.

6. Link the "X" status tag of the "Mixer" step in the GRAPH sequencer to the animation as follows:
   - Open the tab overview.
   - Select the "Mixer" step under "S7-300 Master" > "Program blocks" > "GRAPH Sequence DB".
   - Link the "X" tag to the animation with a double click.



Name	Data type
V0	Bool
X	Bool
LA	Bool
VA	Bool
RA	..ool
AA	..ool

7. Enter "0" and "1" as the value range for the tag.



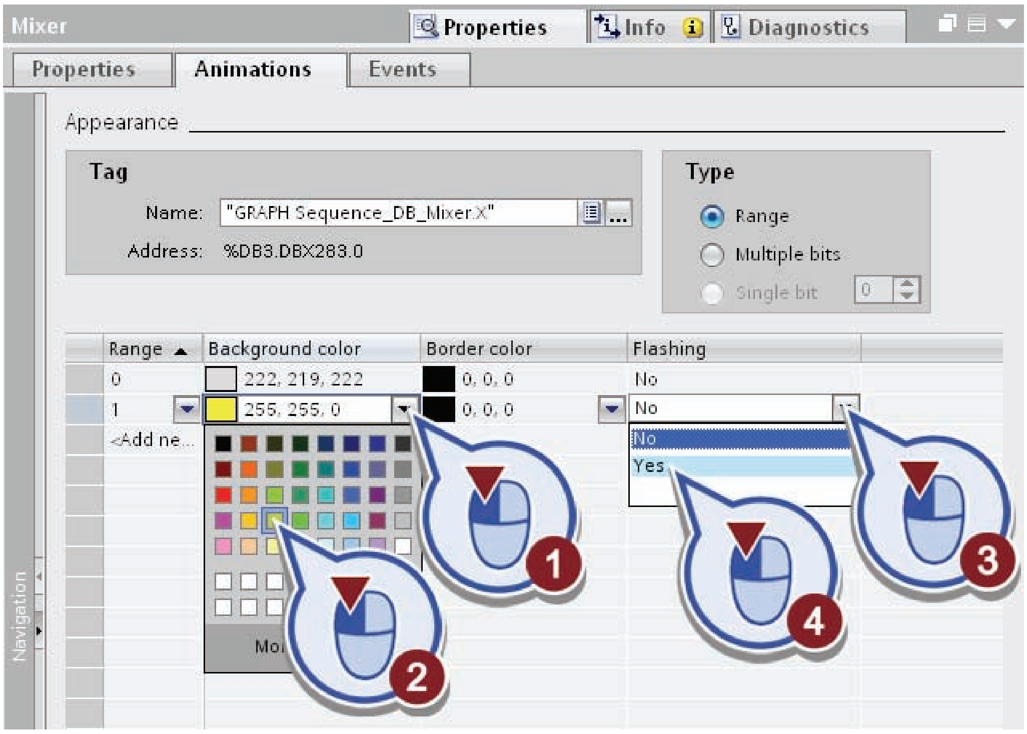
Name: "GRAPH Sequence_DB_Mixer.X"

Address: %DB3.DBX283.0

Type: ● Range ○ Multiple bits ○ Single bit 0

Range ▲	Background color	Border color	Flashing
0	222, 219, 222	0, 0, 0	No
1	222, 219, 222	0, 0, 0	No
<A...			

8. For "1", select a different background color and enable the "Flashing" function.



Mixer

Appearance

Tag

Name: "GRAPH Sequence_DB_Mixer.X"

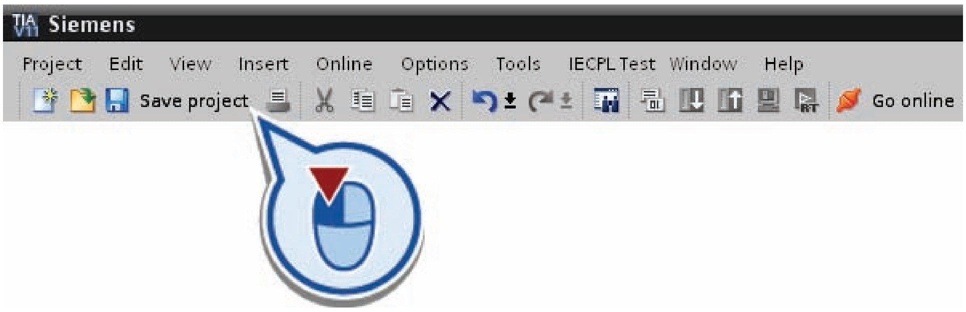
Address: %DB3.DBX283.0

Type

● Range
○ Multiple bits
○ Single bit 0

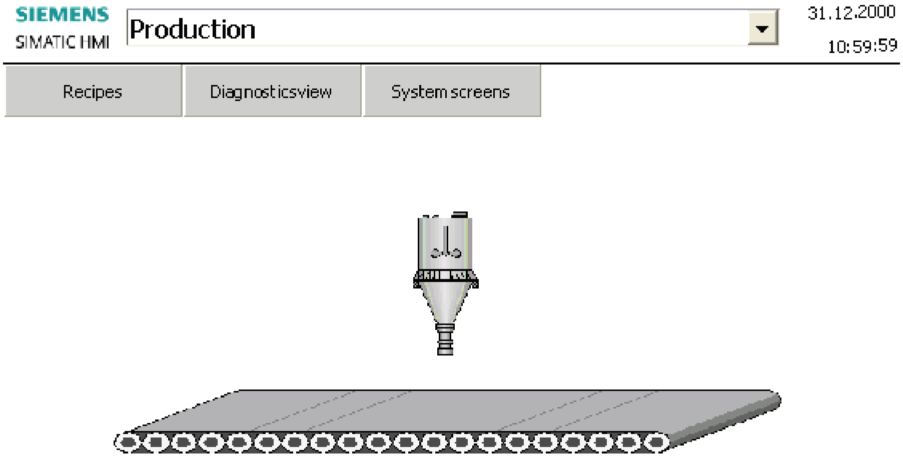
Range	Background color	Border color	Flashing
0	222, 219, 222	0, 0, 0	No
1	255, 255, 0	0, 0, 0	No
<Add ne...			

9. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.



Result

You have successfully inserted the graphics for the filling station with a mixer into the "Production" root screen. You have also animated the mixer, so that the background flashes yellow while the "S3 Mixer" GRAPH step is being executed.



5.3.4 Visualizing beverage tanks

Introduction

In the following, you will add three graphics to represent the beverage tank.

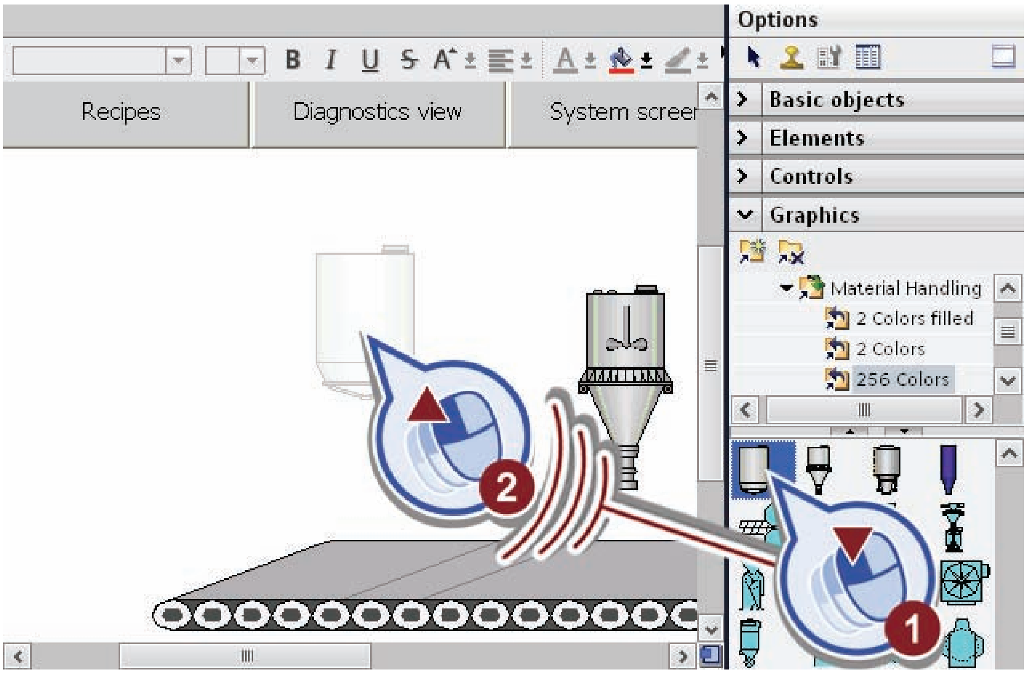
Requirement

You have created the "Production" root screen.

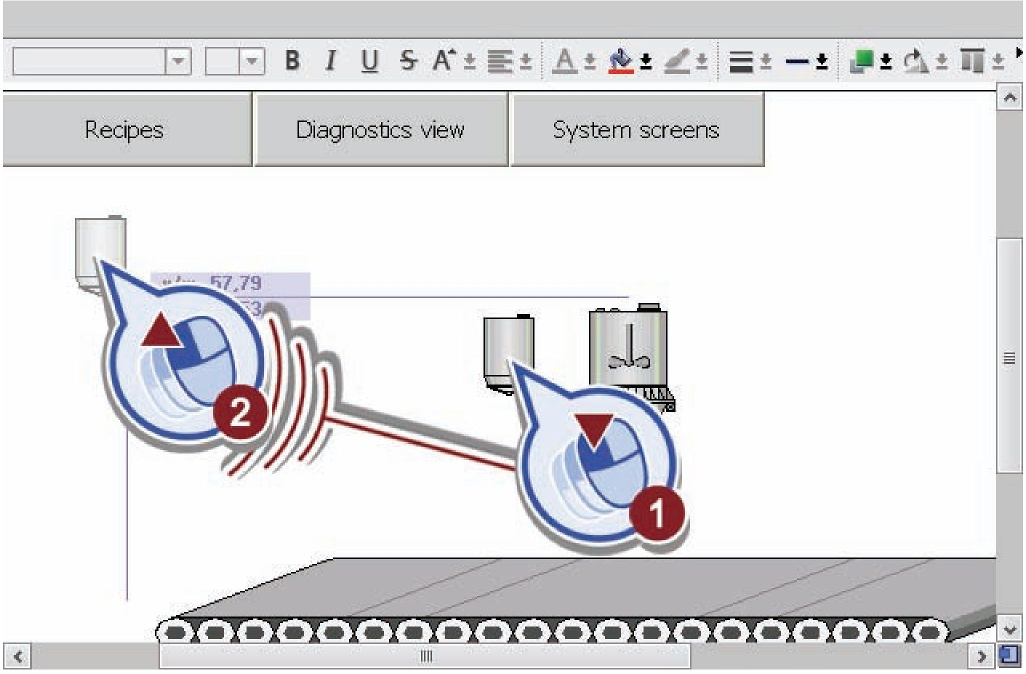
Procedure

To visualize the beverage tanks, follow these steps:

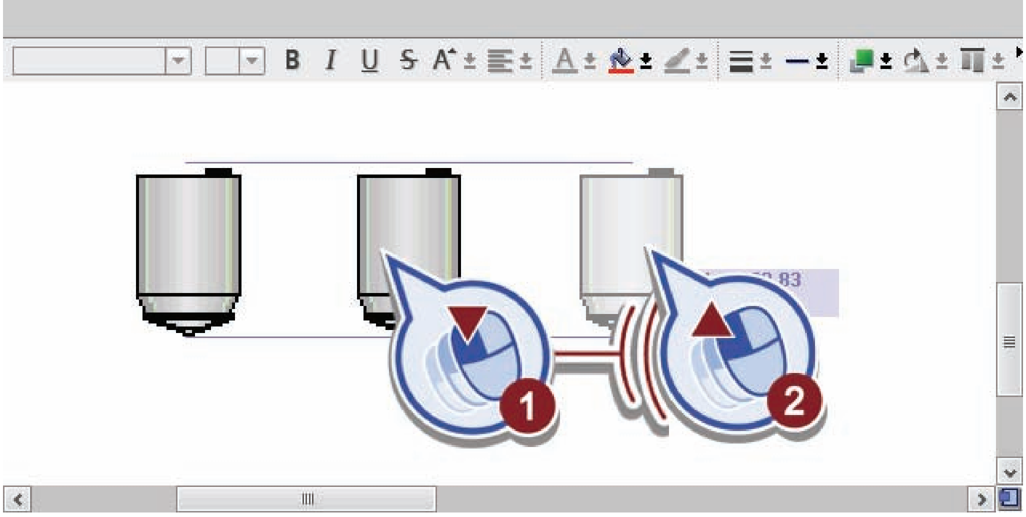
1. Insert the "Silo1" graphic as follows:
   – Open the directory "WinCC graphics folder" > "Industries" > "Material Handling".
   – Click on the "256 Colors" folder.
   – Drag the "Silo1" graphic into the "Production" root screen.



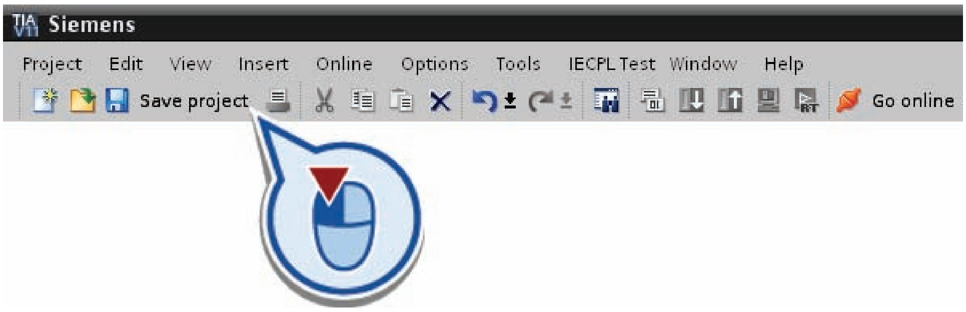
2. Scale the tank to the same size as the upper section of the filling station and place it in the upper left corner of the root screen.



3. Copy the tank twice, by moving it while holding down the <Ctrl> key.

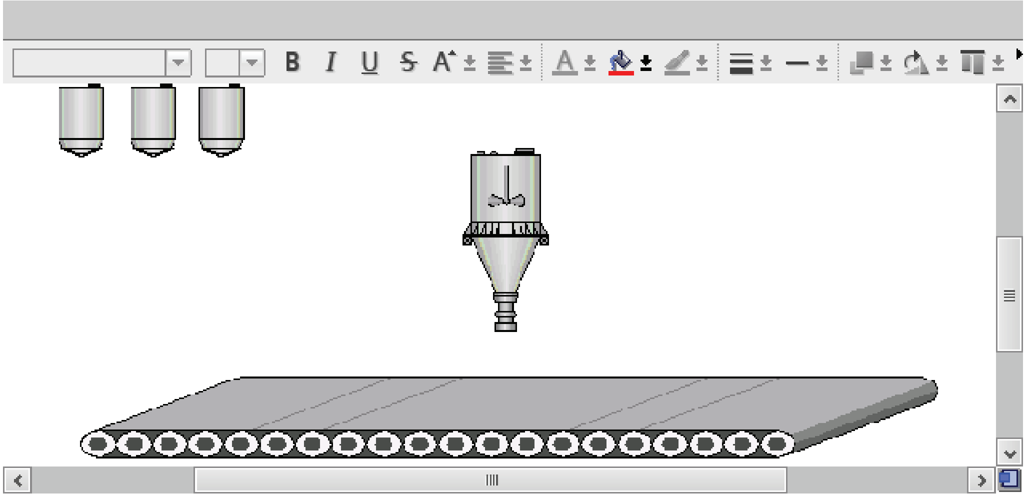


4. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.



Result

You have successfully inserted and scaled three beverage tanks in the "Production" root screen.



5.3.5 Visualizing pipelines

Introduction

In the following, you will add graphics to the display of pipelines from the beverage tank to the filling station in the "Production" root screen.

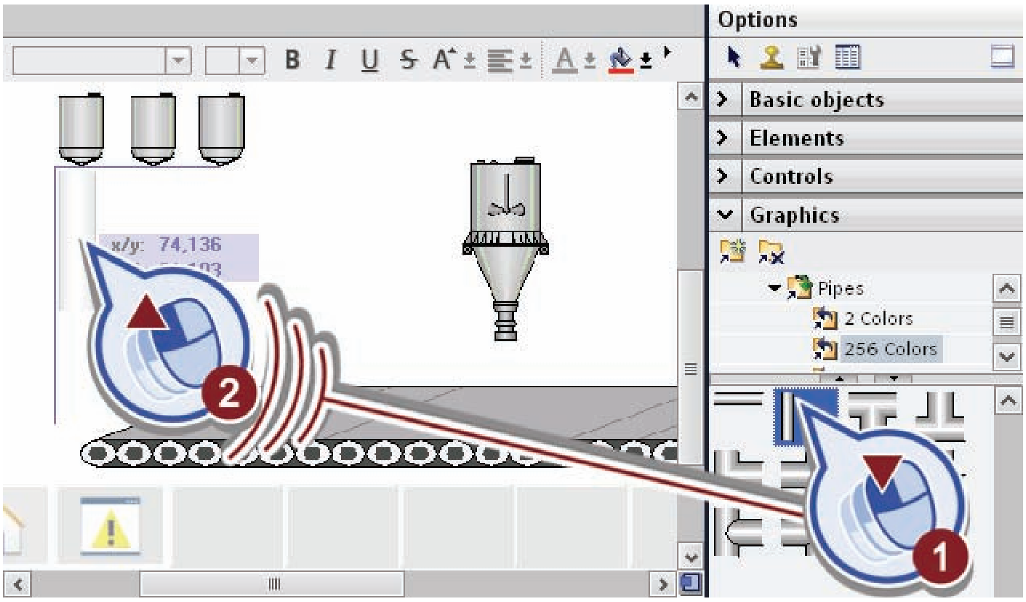
Requirement

You have created the "Production" root screen.

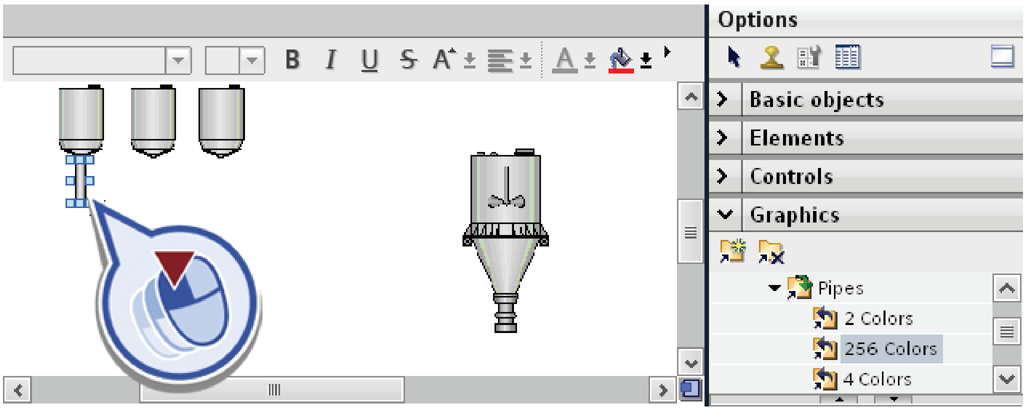
Procedure

To visualize the pipelines, follow these steps:

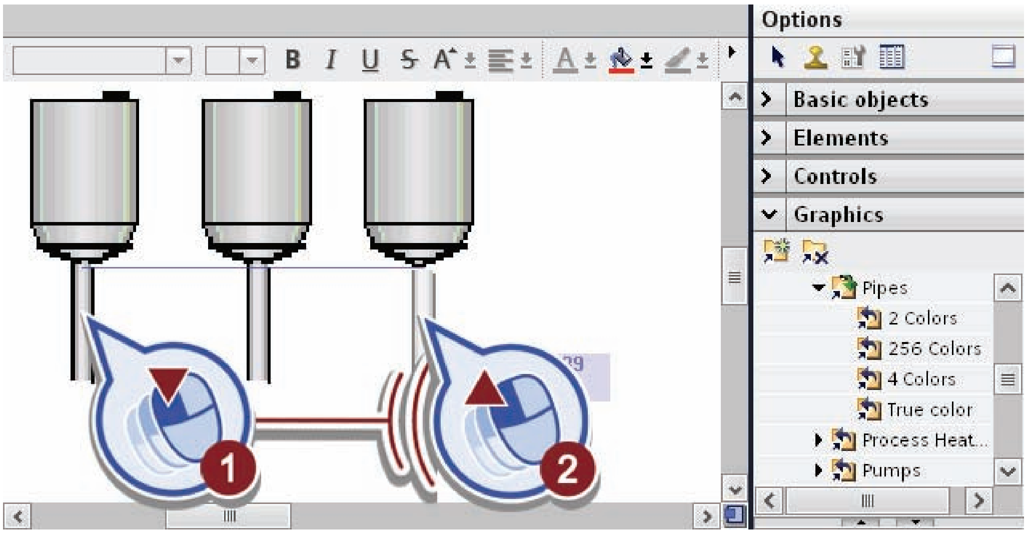
1. Insert the "Short vertical pipe" graphic as follows:
   - Open the directory "WinCC graphics folder" > "Automation equipment" > "Pipes".
   - Click on the "256 Colors" folder.
   - Drag the "Short vertical pipe" graphic into the "Production" root screen.



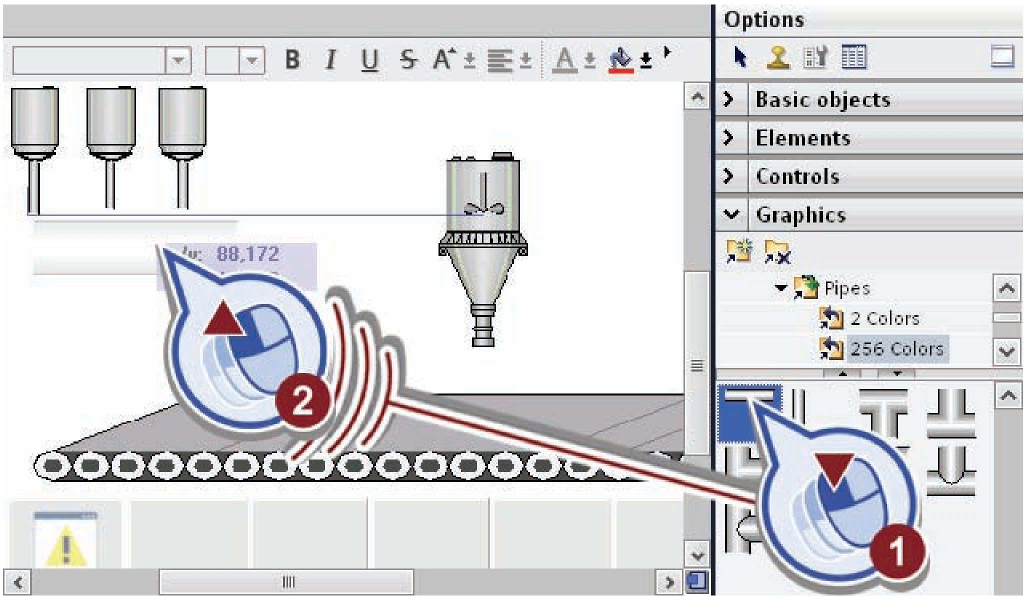
2. Scale the "Short vertical pipe" graphic to a matching size and place it below the first beverage tank.



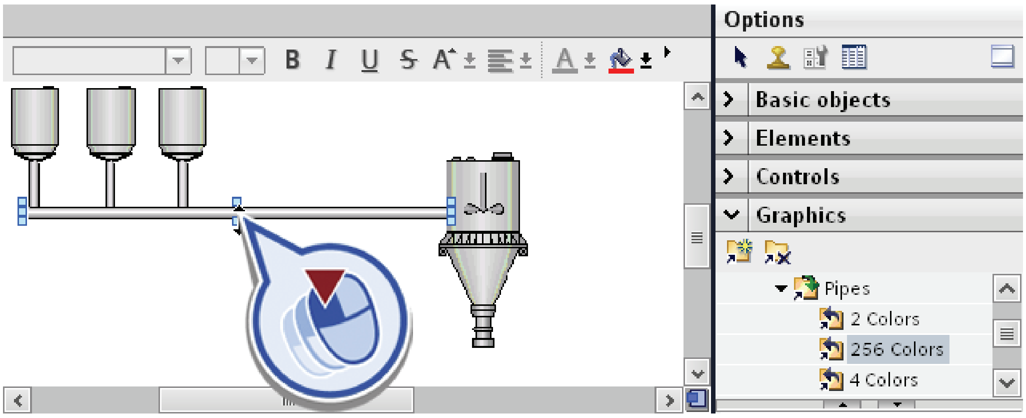
3. Copy the pipe twice, by moving it while holding down the <Ctrl> key. Position the vertical pipes below each of the beverage tanks.



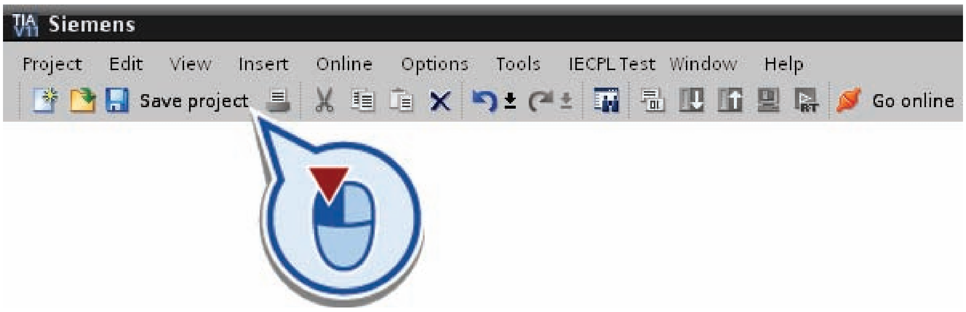
4. Insert the "Short horizontal pipe" graphic as follows:
   - Open the directory "WinCC graphics folder" > "Automation equipment" > "Pipes".
   - Click on the "256 Colors" folder.
   - Drag the "Short horizontal pipe" graphic into the "Production" root screen.



5. Scale the graphic to the same width as the vertical pipes and position it below the pipes of the beverage tank.



6. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.



Result

You have successfully inserted and scaled the pipelines from the beverage tanks to the filling station in the "Production" root screen.

5.3.6 Visualizing bottles on the conveyor belt

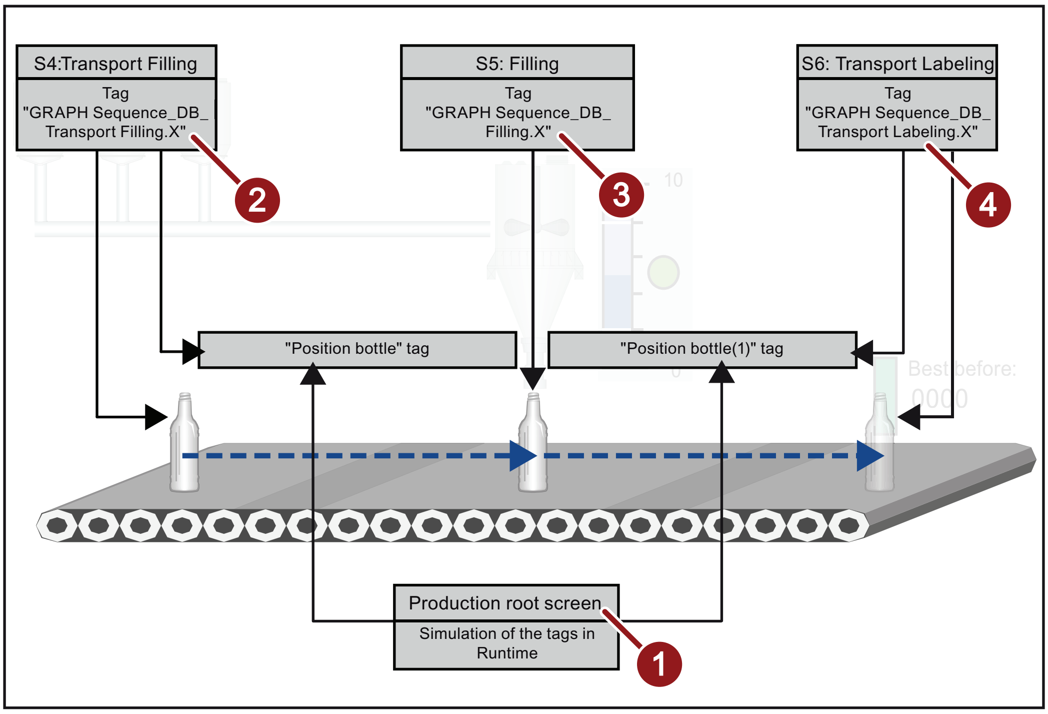
5.3.6.1 Overview of the visualization of the bottles

Introduction

The following provides an overview of the sequence of the movement animation for the bottles on the conveyor belt. There is a graphic for visualizing a bottle and a corresponding animation for each step of the GRAPH sequencer:

- The first bottle visualizes the "S4 Transport Filling" step with a motion animation from the left end of the conveyor belt to the filling station in the center of the conveyor belt.
- The second bottle visualizes the "S5 Filling" step. During filling, the bottle stops below the filling station.
- The third bottle visualizes the "S6 Transport Labeling" step with a motion animation from the filling station in the center of the conveyor belt to the labeling station at the right end of the conveyor belt.

The following graphic shows how the individual tags in the "Production" root screen are controlled as soon as you start the simulation:



The relationships can be explained as follows:

①	**Simulation of the HMI tags** The values of the two tags, "Position Bottle" and "Position Bottle(1)", are incremented by 2 each runtime cycle when WinCC Runtime starts. The motion animation depends on the values of these two tags. The higher the value of the two tags, the farther right the visualized bottle are located within the range of the motion animation. This range is represented by the dashed blue arrow. The bottles are only visible, however, when the corresponding GRAPH step of the animation is being executed.
②	**Visualizing the GRAPH step "S4 Transport Filling"** • As soon as the GRAPH step "S4 Transport Filling" becomes active, the "GRAPH Sequence_DB_Transport Filling.X" tag assumes signal state "1". This makes the first bottle visible with the motion animation. • The "Position Bottle" tag is then set to "0", so that the bottle at the beginning of the GRAPH step is shown at the start position. • While the "S4 Transport Filling" step is being executed, the value of the "Position Bottle" takes continues to be incremented by the value 2 every WinCC Runtime cycle and the bottle moves from left to the center of the conveyor belt. • The first bottle is no longer visible after execution of the "S4 Transport Filling" step.
③	**Visualizing the GRAPH step "S5 Filling"** • As soon as the GRAPH step "S5 Filling" becomes active, the "GRAPH Sequence_DB_Filling.X" tag switches to signal state "1". • This makes second bottle visible, and the bottle is positioned below the filling station. • A tag for motion animation is not needed here because the bottle stops below the filling stations during the execution of the "S5 Filling" step.
④	**Visualizing the GRAPH step "S6 Transport Labeling"** • As soon as the GRAPH step "S6 Transport Labeling" becomes active, the "GRAPH Sequence_DB_Transport Labeling.X" tag assumes signal state "1". This makes the third bottle visible with the motion animation. • The "Position Bottle(1)" tag is then set to "0", so that the bottle at the beginning of the GRAPH step is shown at the start position. • While the "S6 Transport Labeling" step is being executed, the value of the "Position Bottle" takes continues to be incremented by the value 2 every WinCC Runtime cycle and the bottle moves from center of the conveyor belt to the labeling station. • The third bottle is no longer visible after execution of the "S6 Transport Labeling" step.

5.3.6.2 Creating the animation for the "S4 Transport Filling" GRAPH step

Introduction

In the following, you will add the animation for the first bottle of the "S4 Transport Filling" GRAPH step.

- The bottle moves during the execution of the step from the left end of the conveyor belt to the position below the filling station. You can realize the motion animation using an internal HMI tag containing the position information of the bottle.
- The bottles should only be visible when the "S4 Transport Filling" GRAPH step is being executed.

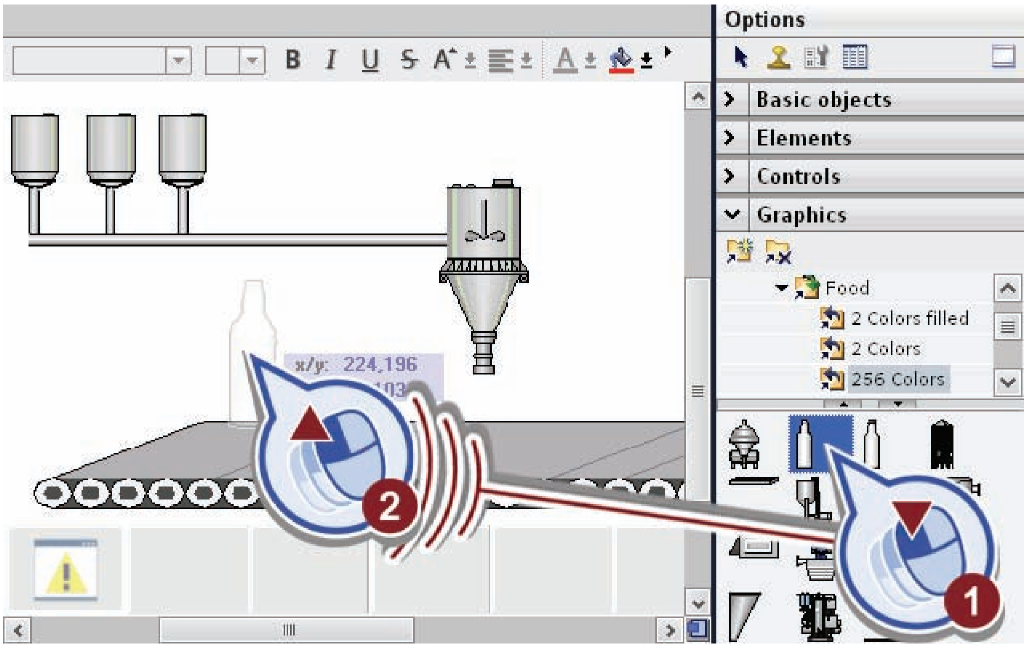
Requirement

You have created the "S4 Transport Filling" GRAPH step and the "Production" root screen.

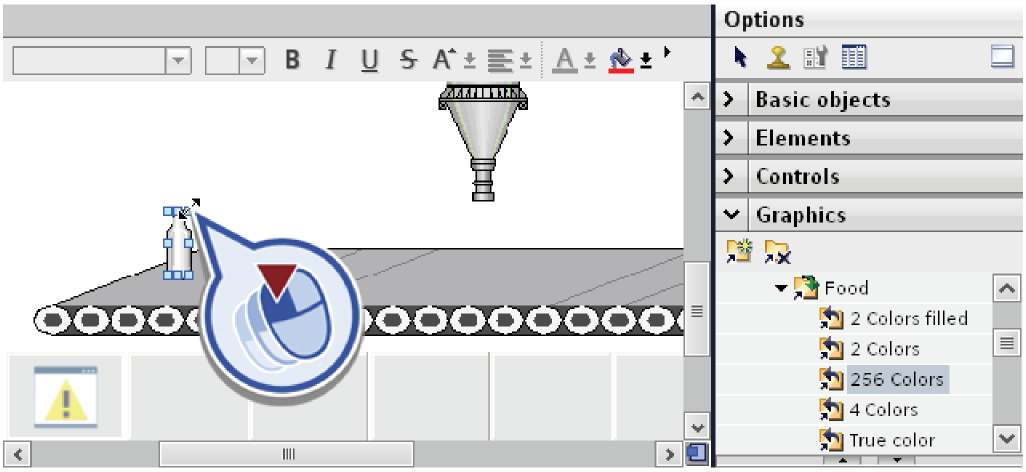
Procedure

To visualize the first bottle on the conveyor belt, follow these steps:

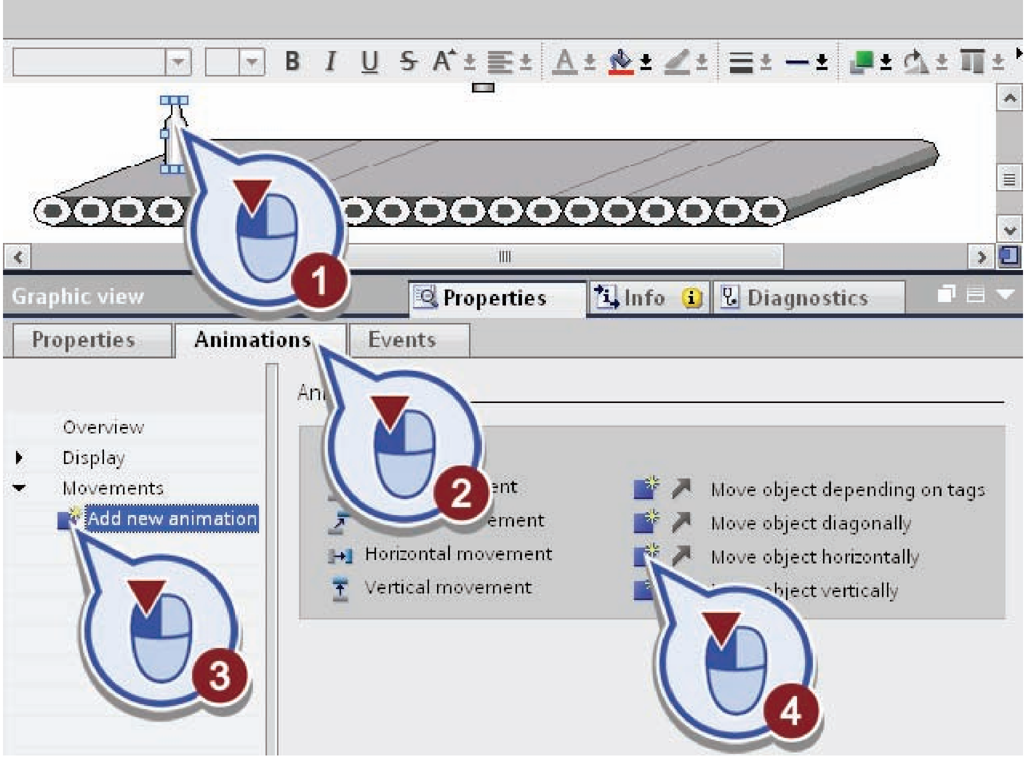
1. Insert the "Glass bottle (no cap)" graphic as follows:
   – Open the directory "WinCC graphics folder" > "Industries" > "Food".
   – Click on the "256 Colors" folder.
   – Drag the "Glass bottle (no cap)" graphic into the "Production" root screen.



2. Scale the graphic to the distance between the conveyor belt and the lower end of the filling station. Place the bottle on the left edge of the conveyor belt.

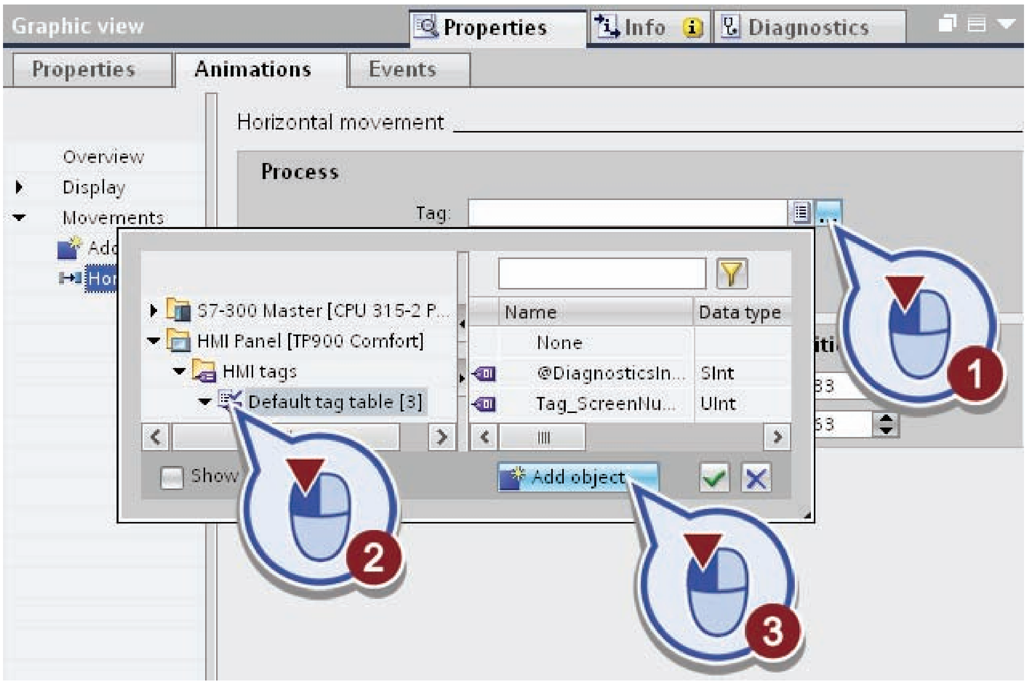


3. Create the animation as follows:
   - Select the graphic.
   - Open the "Animations" tab.
   - Click on "Add new animation" in the "Movements" folder.
   - Select the "Move object horizontally" function.

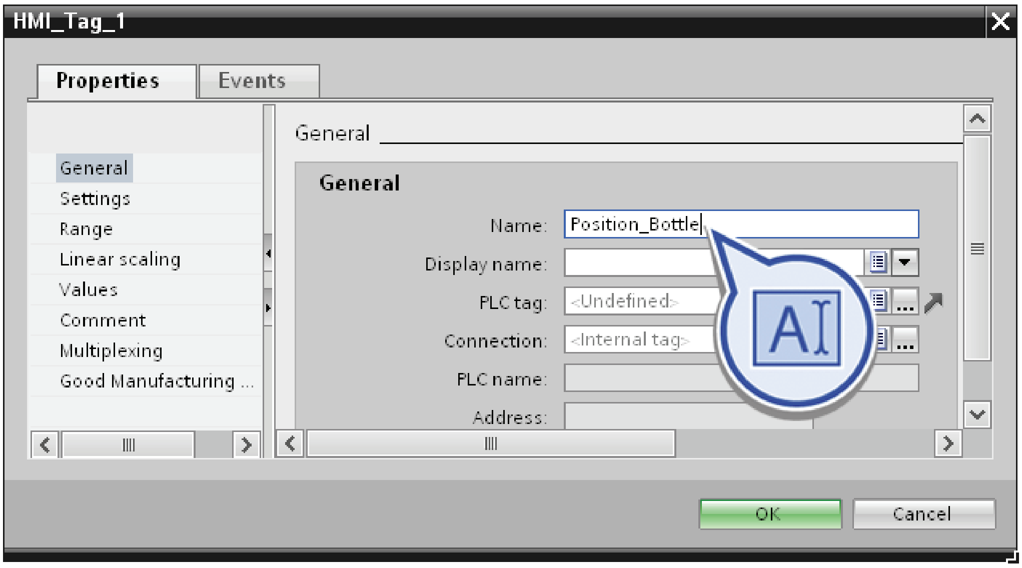


The "Horizontal movement" dialog opens.

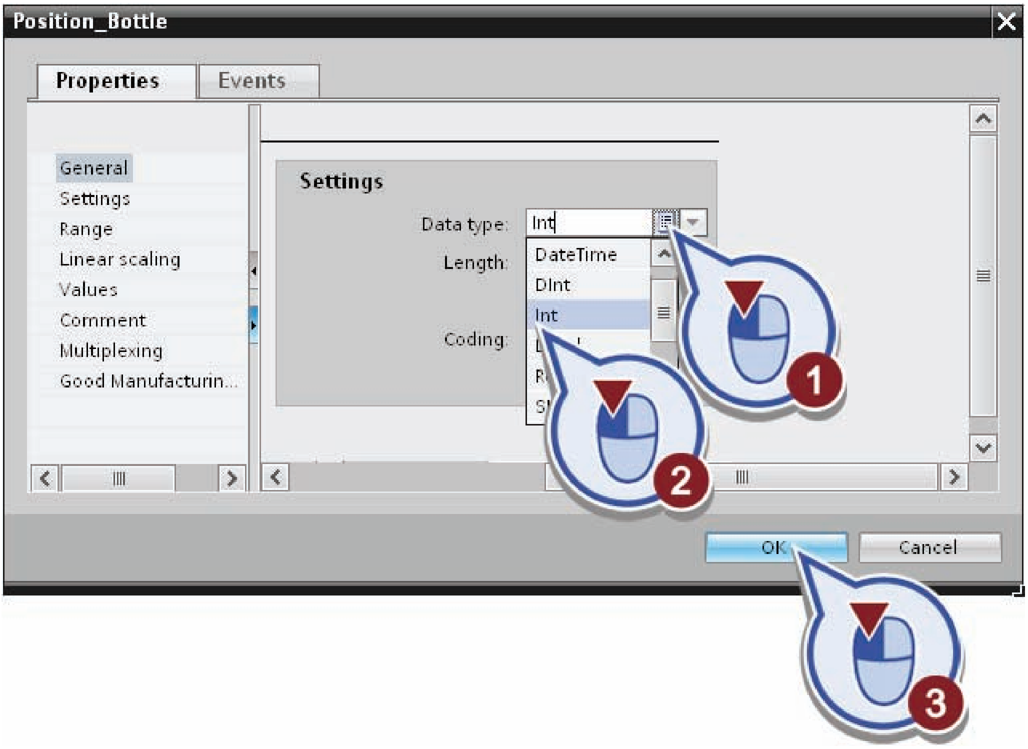
4. Create a new tag as follows:
   - Open the tab overview.
   - Select the "Default tag table" under "HMI Panel" > "HMI Tags".
   - Click on "Add object".



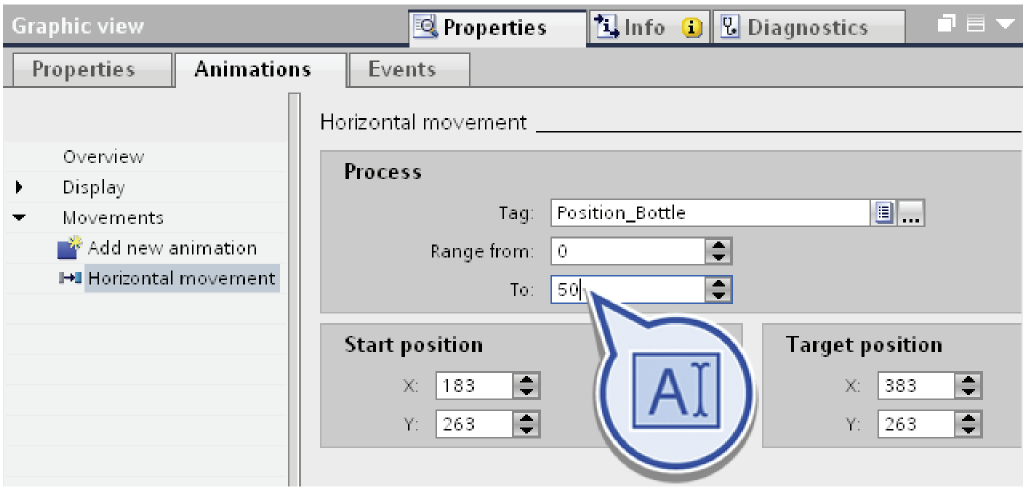
5. Enter "Position_Bottle" as the name for the new tag.



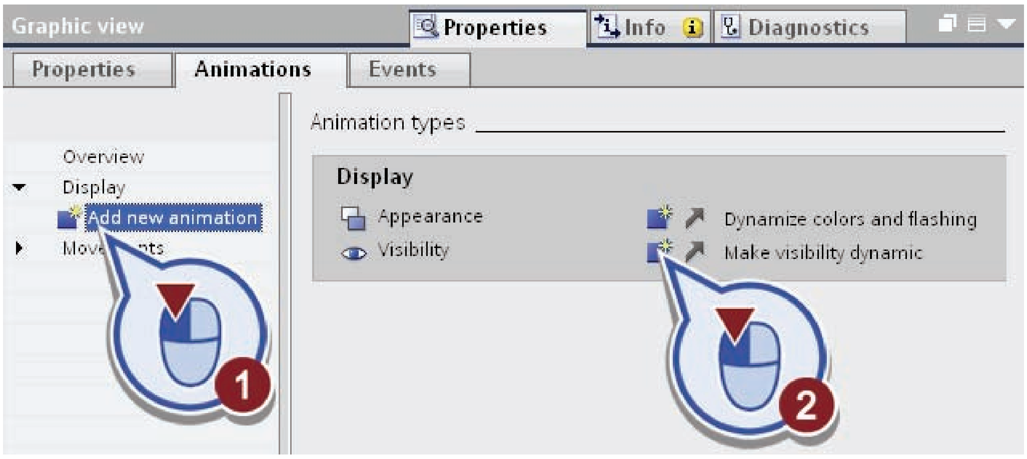
6. Change the data type of the tags as follows:
   - Open the "Data type" drop-down list.
   - Select the data type "Int".
   - Confirm the creation of the tag with "OK".



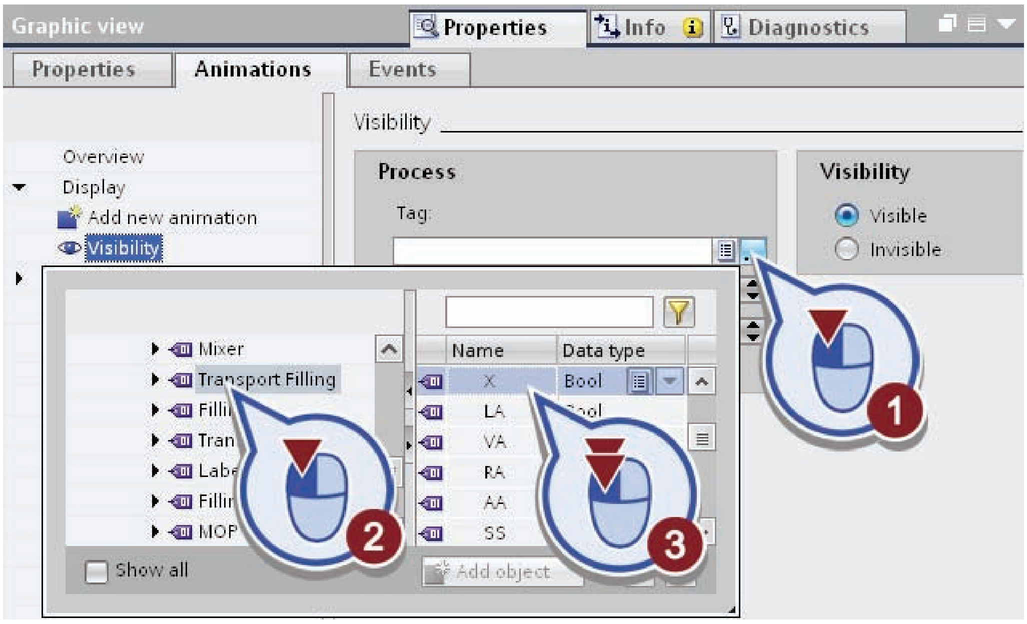
7. Enter "50" as the high limit of the tag's value range in the "Process" dialog.



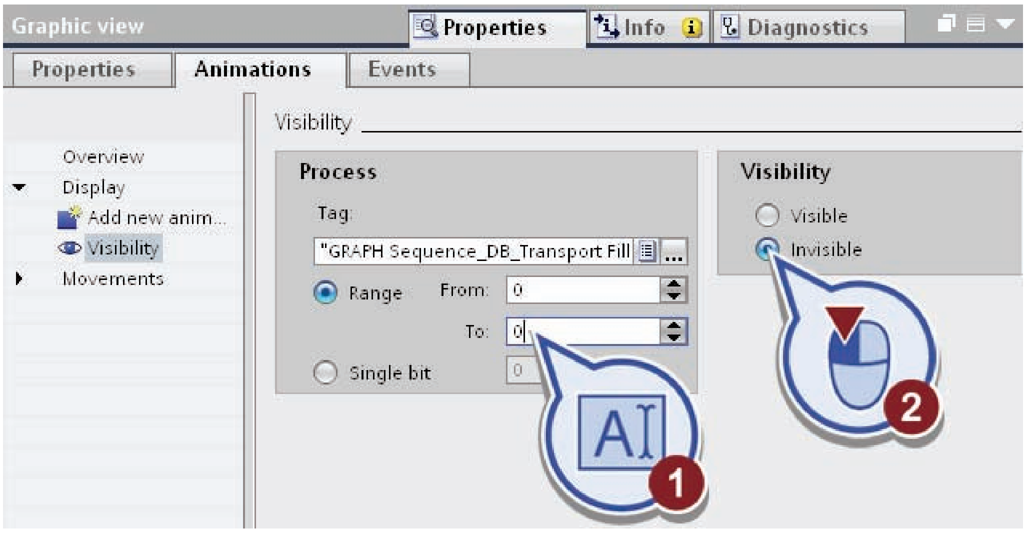
8. Create an animation as follows:
   - Click on "Add new animation" function under "Display" in the "Animations" tab.
   - Select the "Make visibility dynamic" function under "Animation types".



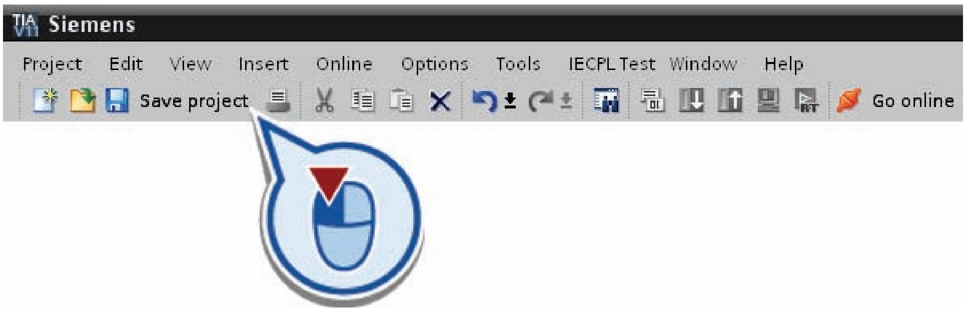
9. Link the animation to the "X" status tag of the "Transport Filling" step as follows:
   - Open the tab overview.
   - Select the "Transport Filling" step under "S7-300 Master" > "Program blocks" > "GRAPH Sequence DB".
   - Link the "X" tag to the animation with a double click.



10. Enter "0" to "0" as the Range and select the "Invisible" setting under "Visibility".



11. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.


Siemens
Project Edit View Insert Online Options Tools IECPL Test Window Help
Save project
Go online

Result

You have successfully inserted and animated the first bottle.

- The bottle changes its position according to the value of the "Position_Bottle" tab.
- The bottle is now only visible when the "S4 Transport Filling" GRAPH step is being executed, i.e., when the "X" tag in "GRAPH Sequence DB" has signal state "1".

5.3.6.3 Creating the animation for the "S5 Filling" GRAPH step

Introduction

In the following, you will insert the second bottle on the conveyor belt in the "Production" root screen and animate it. The bottles should only be visible when the "S5 Filling" GRAPH step is being executed. During the "S5 Filling" step, the bottle stops below the filling station.

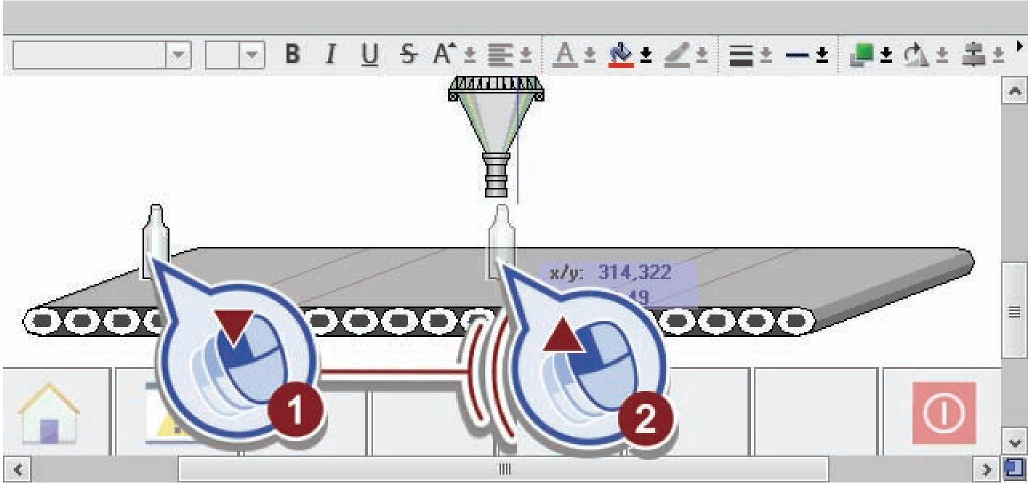
Requirement

You have created the "S5 Filling" GRAPH step, the "GRAPH_Sequence_DB" data block, and the "Production" root screen.

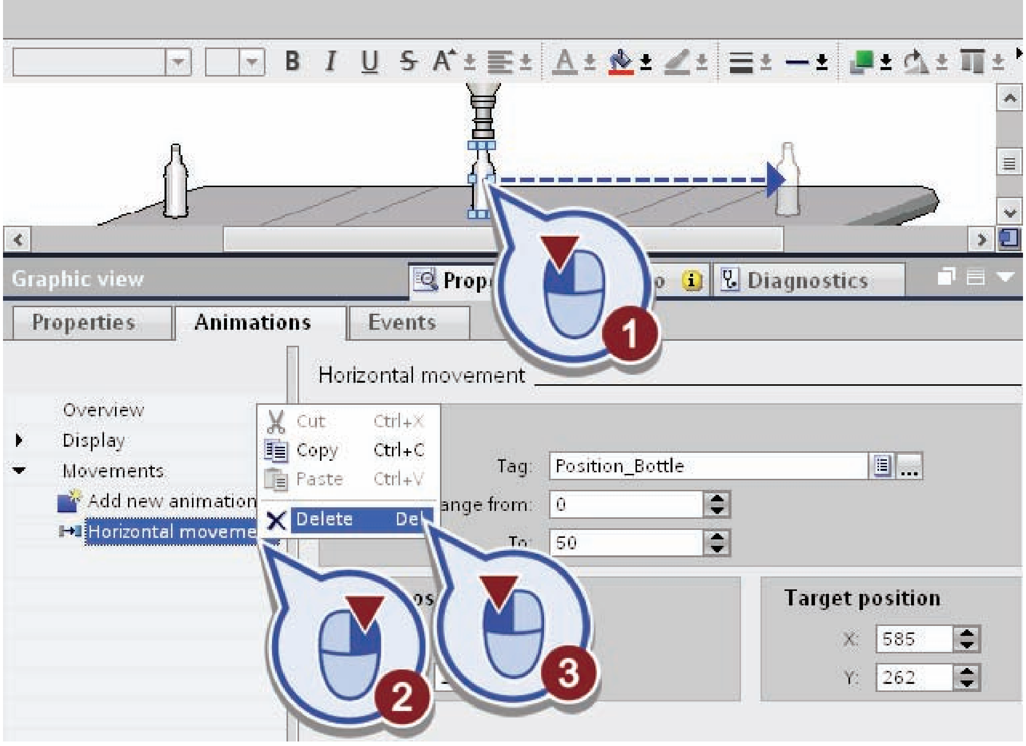
Procedure

To visualize the second bottle on the conveyor belt, follow these steps:

1. Copy the first bottle, by dragging it to the area below the filling station while holding down the <Ctrl> key. The properties of the bottle previously defined are copied as well.



2. No motion animation is required for the "S5 Filling" step. Therefore, delete the "Horizontal movement" as follows:
   – Select the second bottle.
   – In the "Animations" tab, right-click on the "Horizontal movement" function.
   – Select "Delete" in the shortcut menu.



Horizontal movement

Tag: Position_Bottle

Range from: 0

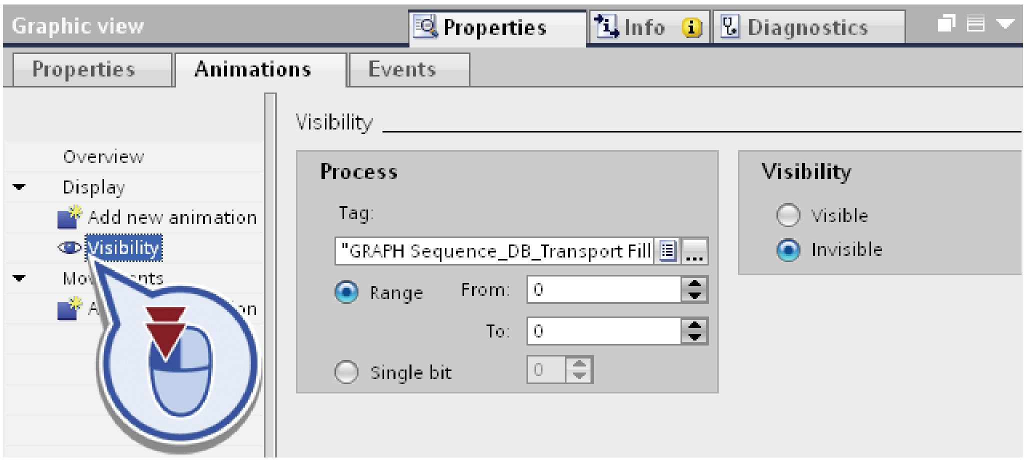
To: 50

Target position

X: 585

Y: 262

3. Open the "Visibility" function with a double click.



Visibility

Process

Tag: "GRAPH Sequence_DB_Transport Fill

● Range From: 0

To: 0

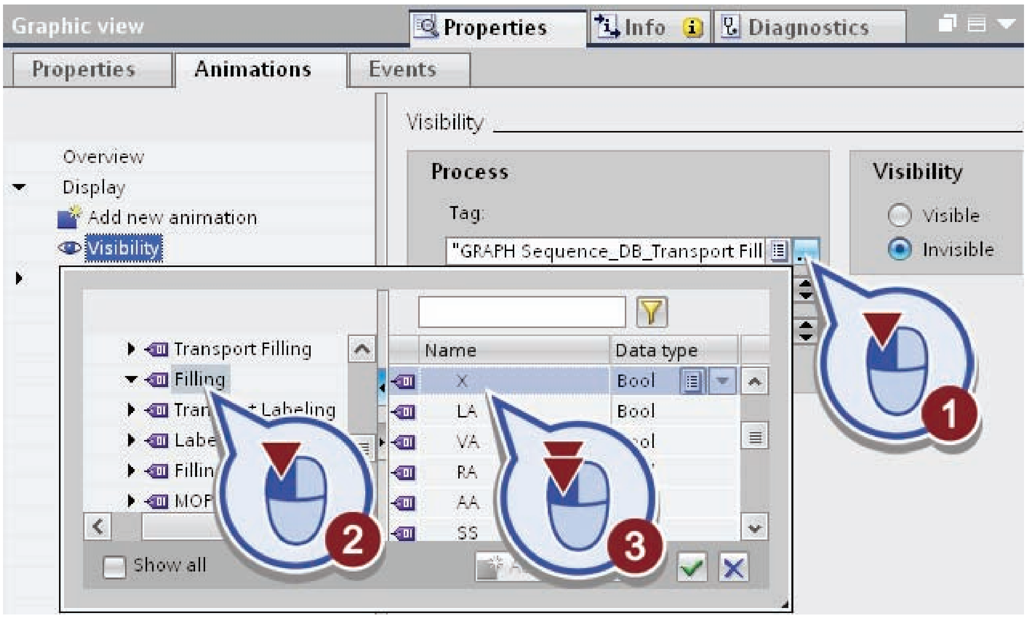
○ Single bit 0

Visibility

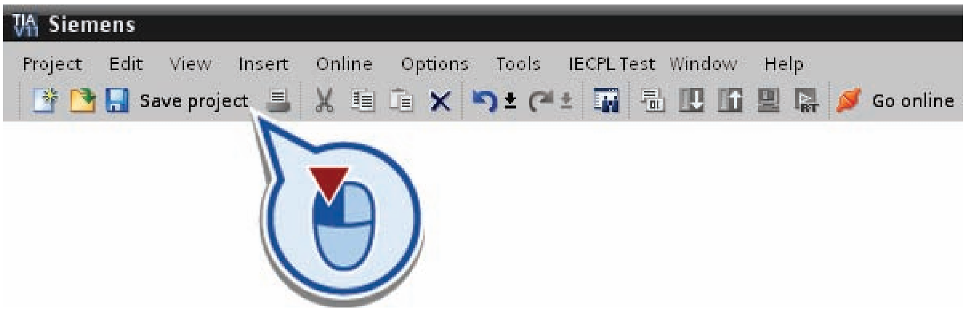
○ Visible

● Invisible

4. To change the tag link in the "X" status tag of the "Filling" step, follow these steps:
   – Open the tab overview.
   – Select the "Filling" step under "S7-300 Master" > "Program blocks" > "GRAPH Sequence DB".
   – Link the "X" tag to the animation with a double click.



5. Click the "Save project" button on the toolbar or press <Ctrl + S> to save the project.



Result

You have successfully inserted the second bottle in the "Production" root screen.

5.3.6.4 Creating the animation for the "S6 Transport Labeling" GRAPH step

Introduction

In the following, you will add another graphic to represent the third bottle on the conveyor belt and animate it.

- The bottle should move horizontally to the right from the filling station to the labeling machine.
- The bottle should only be visible when the "S6 Transport Labeling" GRAPH step is being executed.

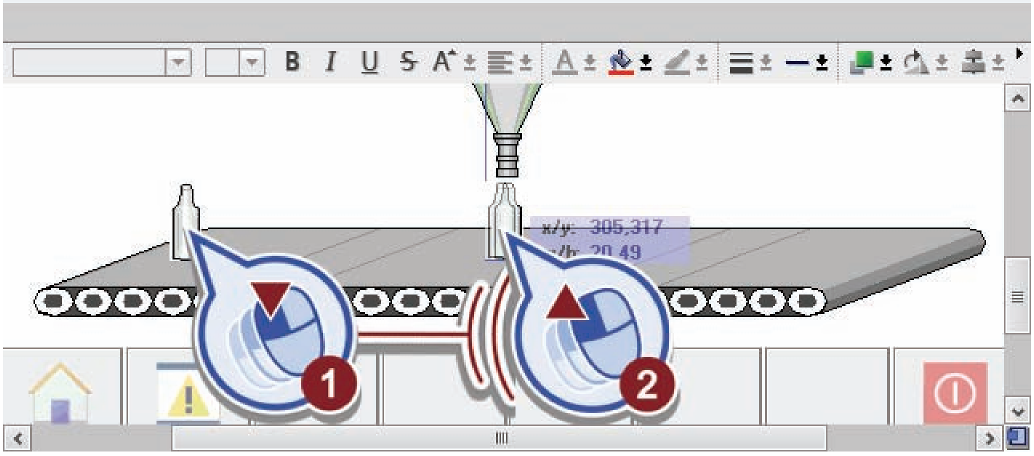
Requirement

You have created the "S6 Transport Labeling" GRAPH step, the "GRAPH_Sequence_DB" data block, and the "Production" root screen.

Procedure

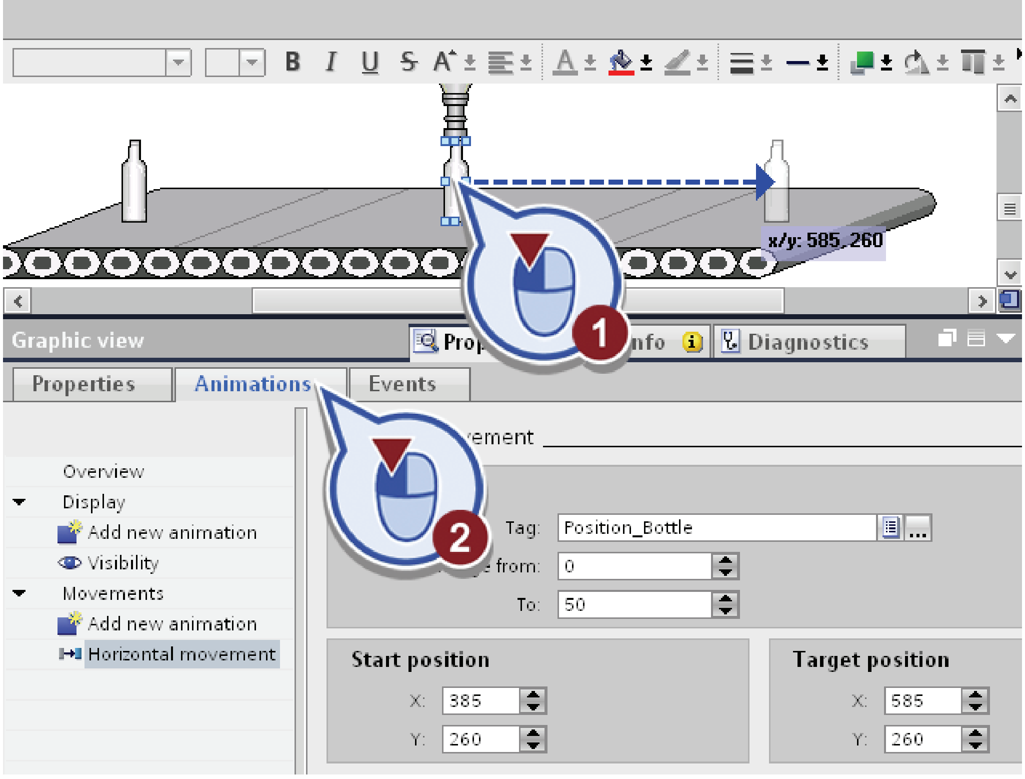
To visualize the third bottle on the conveyor belt, follow these steps:

1. Copy the first bottle, by dragging it to the area below the filling station while holding down the <Ctrl> key.



The properties of the bottle previously defined are copied as well.

2. Open the "Animations" tab.



Tag: Position_Bottle

from: 0

To: 50

Start position

X: 385

Y: 260

Target position

X: 585

Y: 260

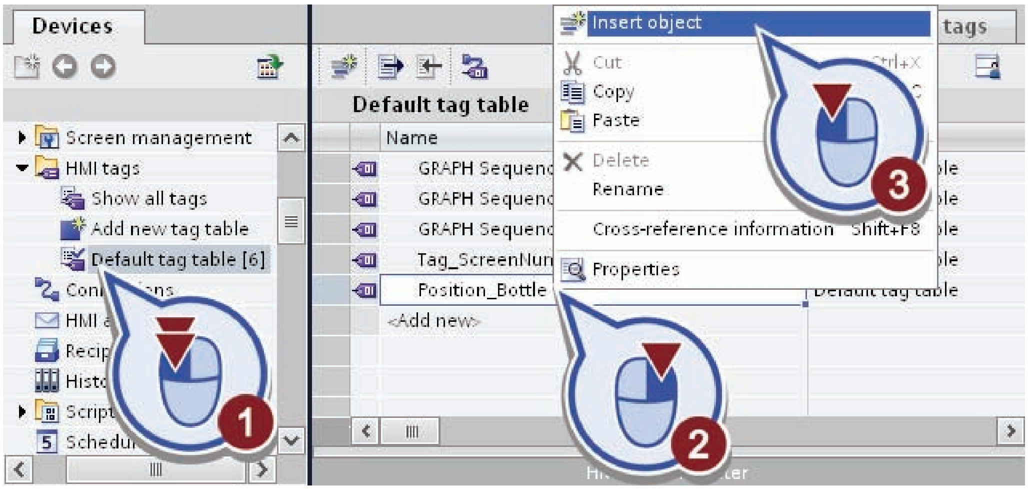
For motion animation of this bottle, you need an additional tag to animate the horizontal movement.

3. Copy the "Position_Bottle" tag as follows:

   – Open the "Default tag table" with a double click in the "HMI Tags" folder in the project tree.

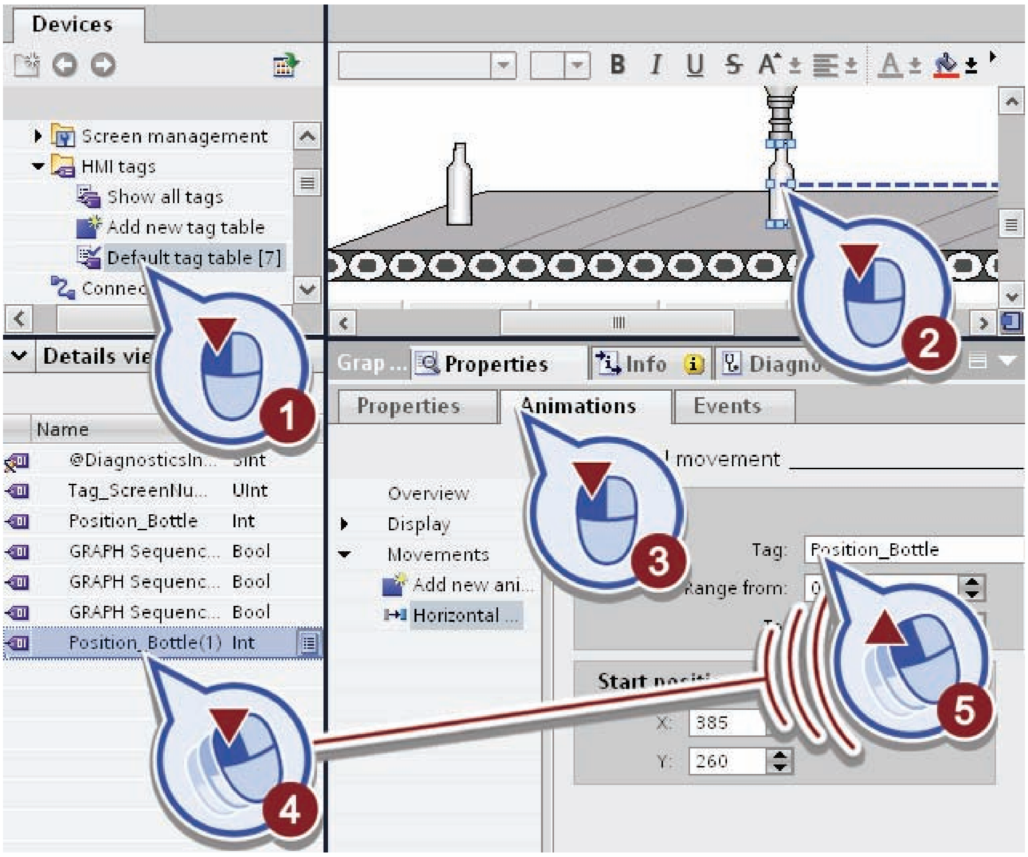
   – Right-click the "Position_Bottle" tag.

   – Select the "Insert object" command in the shortcut menu.

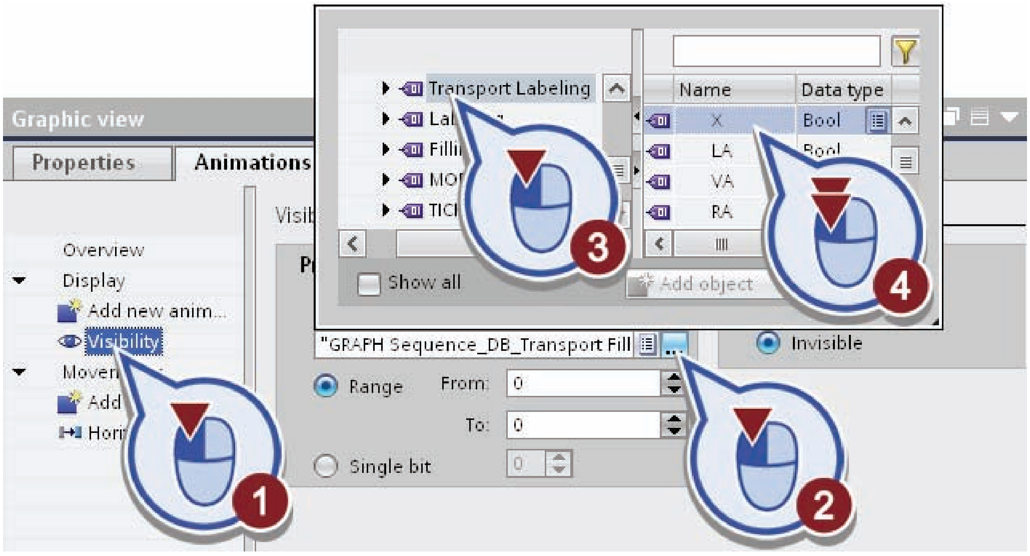


A new "Position_Bottle(1)" tag is created in the Default tag table.

4. First, select the "Default tag table". Then switch to the "Production" root screen, select the third bottle and replace the "Position_Bottle" tag with the "Position_Bottle(1)" tag in the horizontal motion animation.



5. Change the tag link for the "Visibility" animation as follows:
   - Open the "Visibility" dialog.
   - Open the tab overview.
   - Select the "Transport Labeling" step under "S7-300 Master" > "Program blocks" > "GRAPH_Sequence_DB".
   - Link the "X" tag to the animation with a double click.



6. Save the project.

Result

You have successfully inserted the third bottle in the root screen.

- As soon as the "S6 Transport Labeling" step in the GRAPH sequences becomes active, the bottle is visible.
- The position of the bottle depends on the integer value of the "Position_Bottle(1)" tag.

5.3.6.5 Simulating tags for the horizontal movement of the bottles

Introduction

In the following, you will simulate the values of the tags "Position Bottle" and "Position Bottle(1)".

- When Runtime starts, the value of the tags should be automatically increased by 2 in each cycle. When the value of the tags increase, the first and third bottles move from left to right down the conveyor belt, but remain invisible as long as the corresponding step is not being executed.
- When the "S4 Transport Filling" and "S6 Transport Labeling" steps are executed, the bottles should be at the initial position of the respective step.
  - When the "S4 Transport Filling" step is activate, the value of the "Position Bottle" tag should be set to "0".
  - When the "S6 Transport Labeling" step is activate, the value of the "Position Bottle" tag should be set to "0".

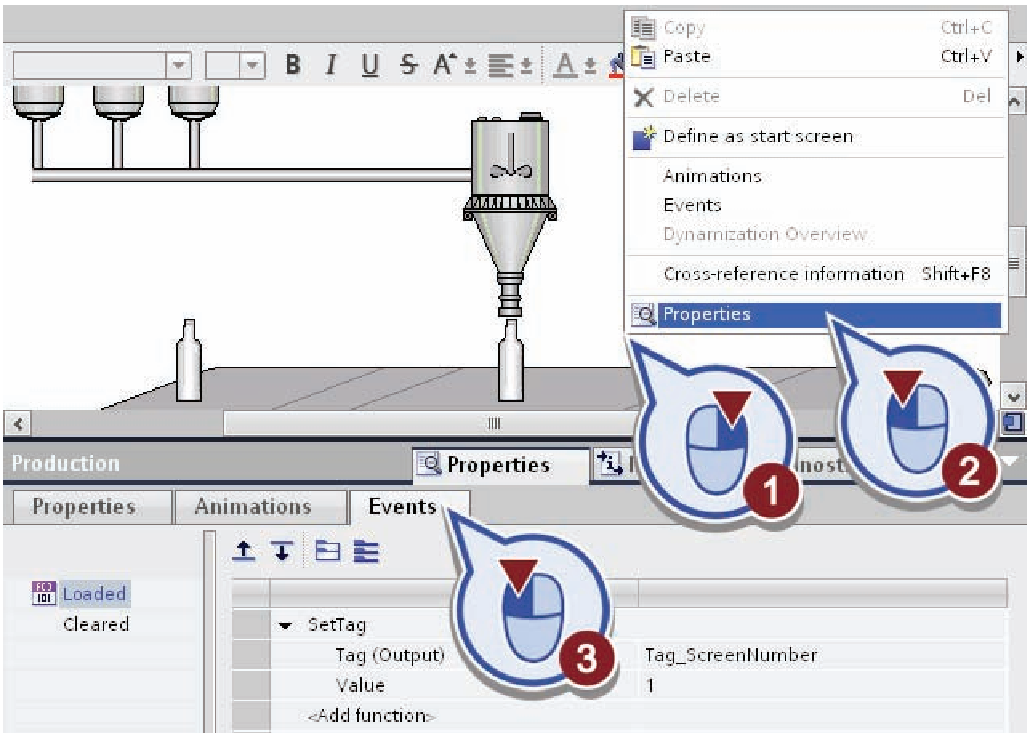
Requirement

You have added the graphics for displaying the bottles and created the motion animations.

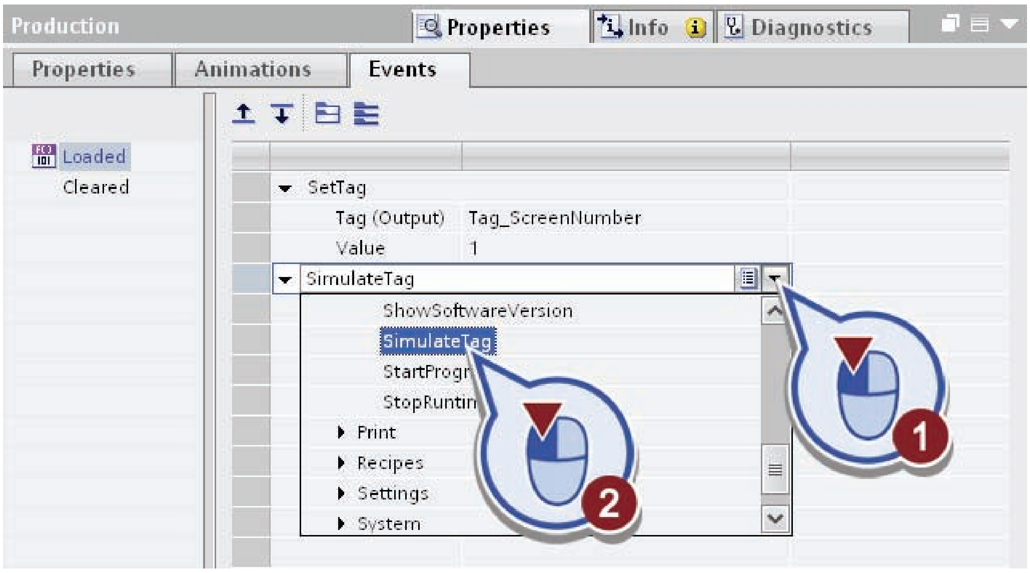
Procedure

To simulate the values of the tags, follow these steps:

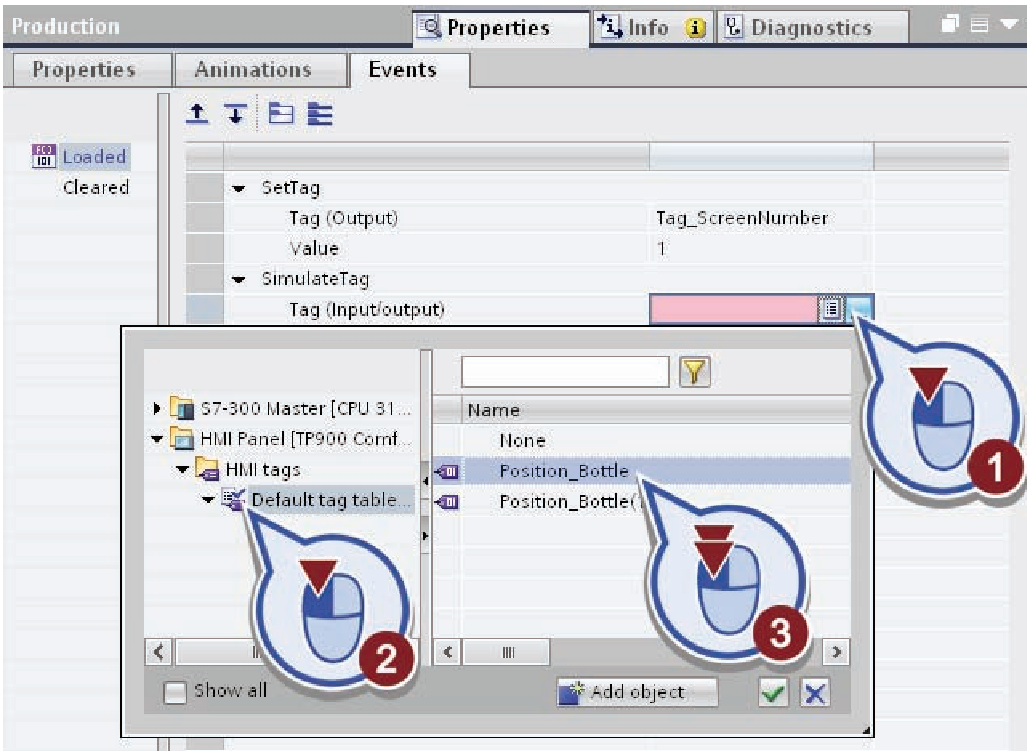
1. Create a new event in the properties of the "Production" root screen as follows:
   - Right-click on an empty area of the root screen.
   - Select "Properties" in the shortcut menu.
   - Open the "Events" tab.



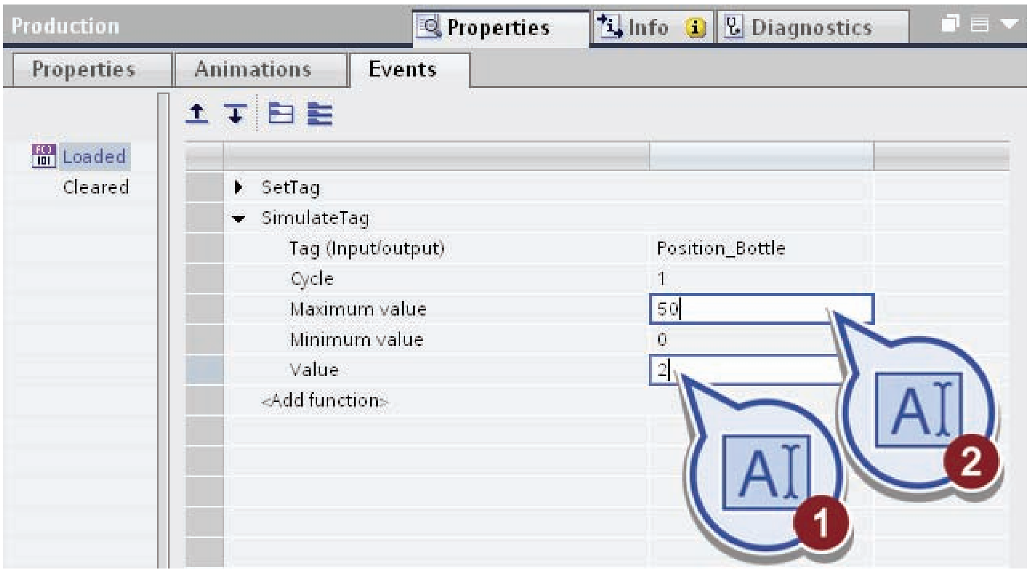
2. Insert the "SimulateTag" function as follows:
   - Open the drop-down list of the <Add function> entry.
   - Select the "SimulateTag" function under "System functions" > "Other functions".



3. Link the "SimulateTag" function to the "Position_Bottle" tag as follows:
   - Open the tab overview.
   - Select the "Default tag table" under "HMI Panel" > "HMI Tags".
   - Assign "Position_Bottle" to the HMI tag with a double click.

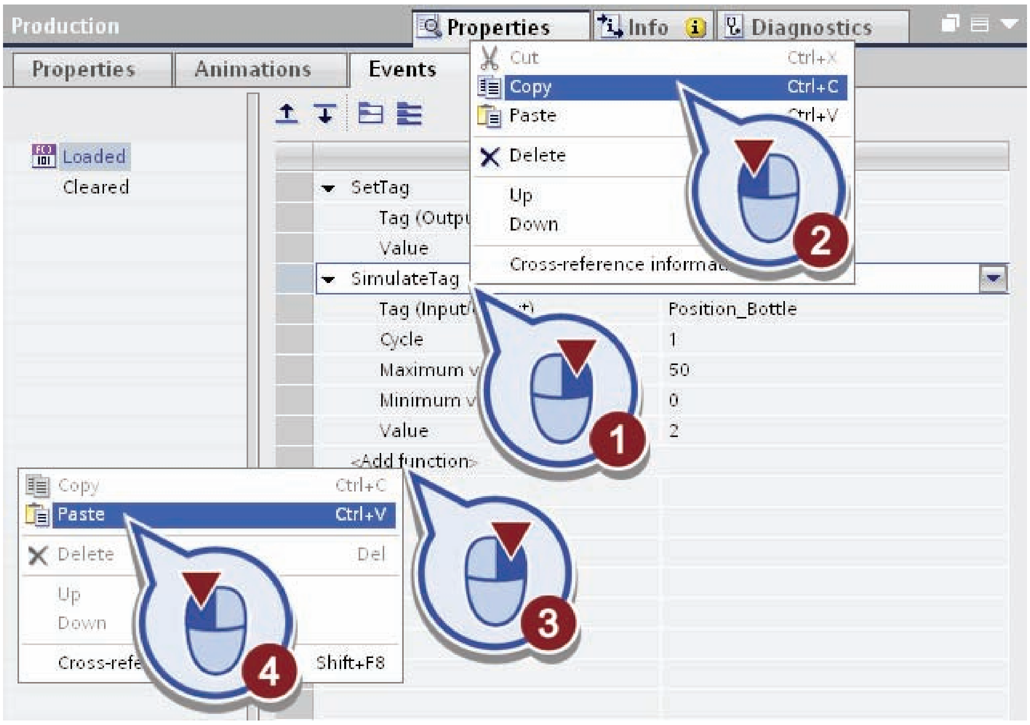


4. Change the maximum value for the tag simulation according to the range of horizontal motion simulation to "50" and set the value for incrementing the tag value per cycle to "2".



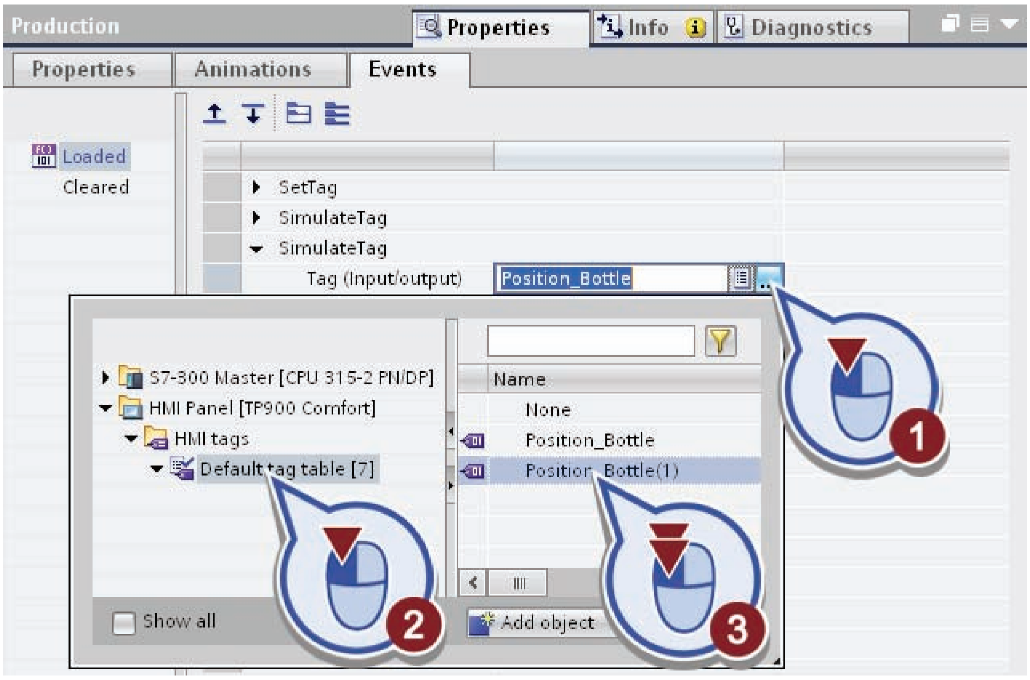
SetTag
SimulateTag
Tag (Input/output) Position_Bottle
Cycle 1
Maximum value 50
Minimum value 0
Value 2
<Add function>

5. Copy the event as follows:
   - Right-click on the "SimulateTag" line.
   - Select "Copy" in the shortcut menu.
   - Right-click on the "Add function" line.
   - Select the "Paste" command in the shortcut menu.



SetTag
Tag (Outpu
Value
SimulateTag
Tag (Input/) Position_Bottle
Cycle 1
Maximum v 50
Minimum v 0
Value 2
<Add function>

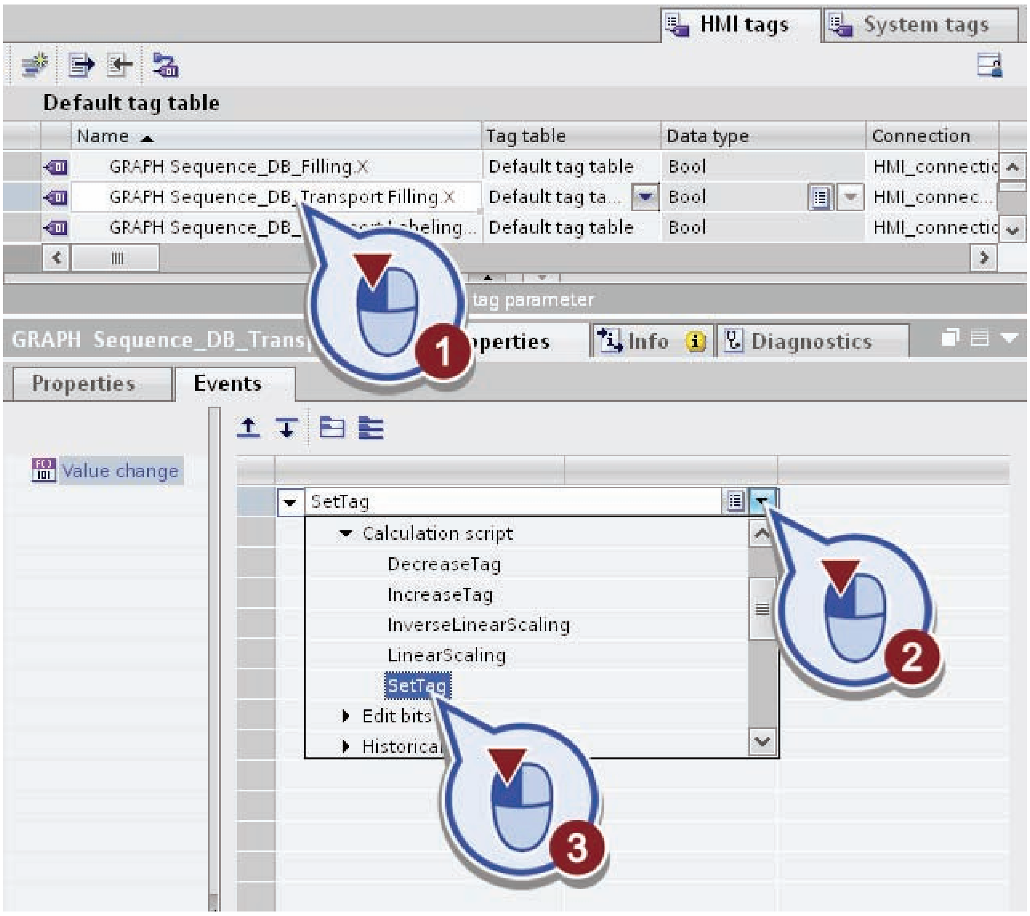
6. Link the copied function to the "Position_Bottle(1)" tag as follows:
   - Open the tab overview.
   - Select the "Default tag table" under "HMI Panel" > "HMI Tags".
   - Assign the "SimulateTag" function the HMI tag "Position_Bottle(1)" with a double-click.



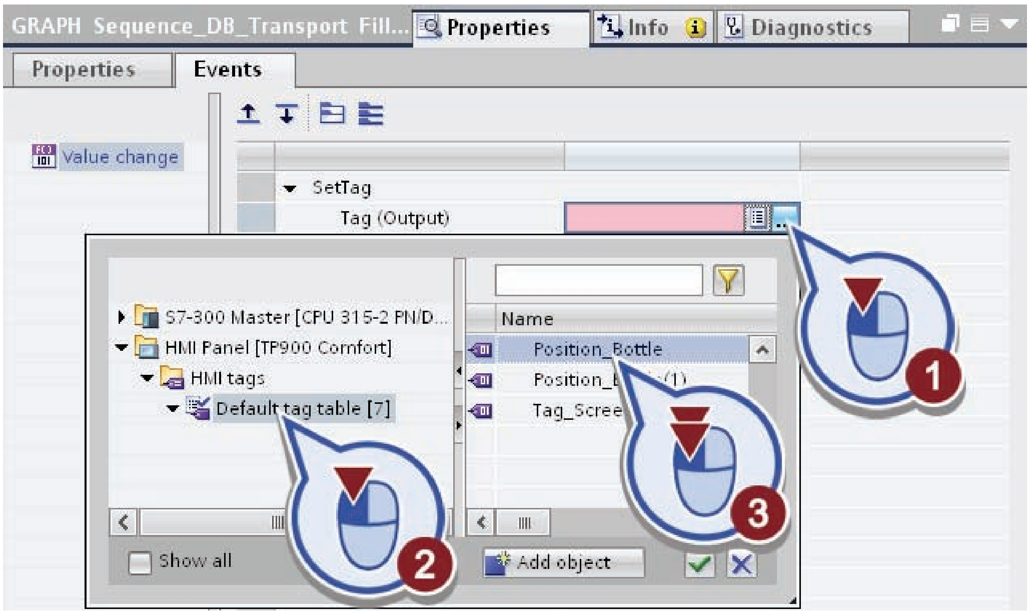
You have created the simulation of the two tags. When Runtime starts, the values of the two tags are automatically incremented each cycle. To ensure that the tags are set "0" when the appropriate step is launched, you will create the corresponding events for the status tags of the steps in the following.

7. Open the "Default tag table" under "HMI Panel" in the "HMI Tags" folder in the project tree.

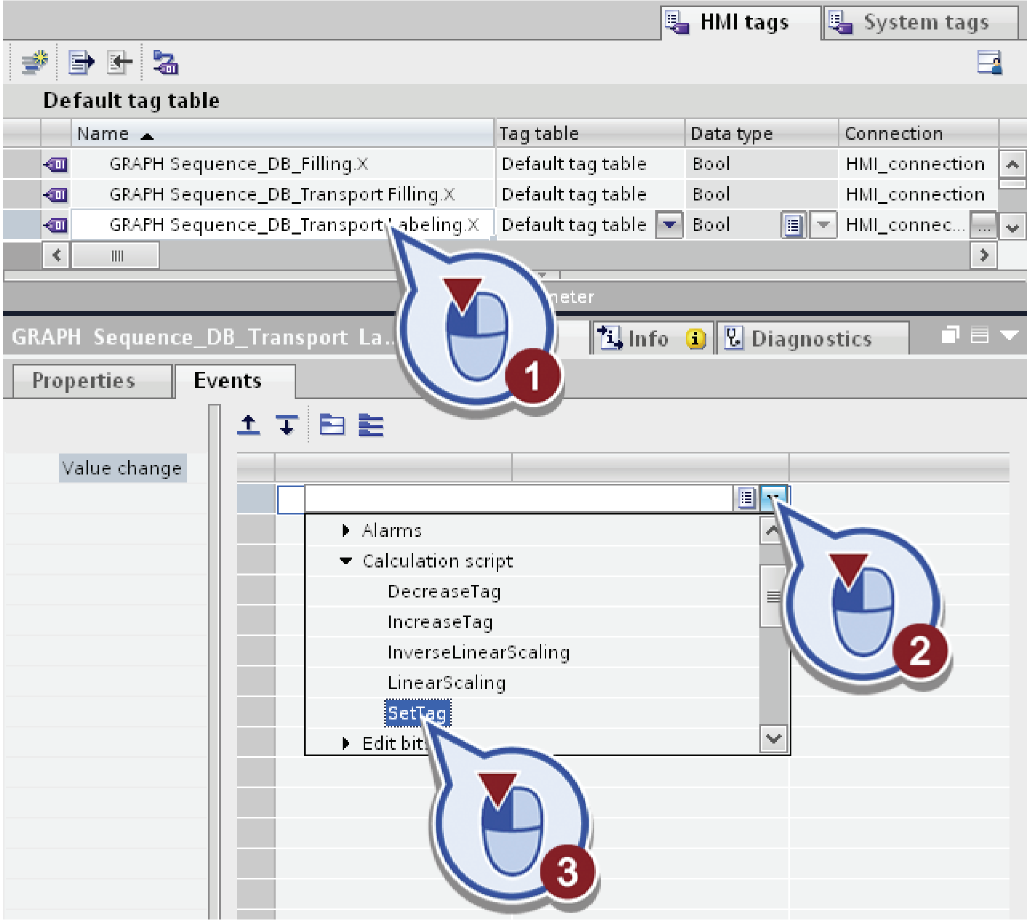
8. Create an event for the "GRAPH Sequence_DB_Transport Filling.X" HMI tag as follows:
   - Select the "GRAPH Sequence_DB_Transport Filling.X" tag.
   - Open the drop-down list of the "<Add function>" line in the "Events" tab.
   - Under "Calculation script", select the "SetTag" function.



9. Assign the "Position_Bottle" tag to the function as follows:
   – Open the tab overview.
   – Select the "Default tag table" under "HMI Panel" > "HMI Tags".
   – Assign the function the HMI tag "Position_Bottle" with a double click.



10. Create another event for the "GRAPH Sequence_DB_Transport Labeling.X" HMI tag as follows:
    - Select the "GRAPH Sequence_DB_Transport Labeling.X" HMI tag in the Default tag table.
    - Open the drop-down list of the "<Add function>" line in the "Events" tab.
    - Under "Calculation script", select the "SetTag" function.



Default tag table

Name	Tag table	Data type	Connection
GRAPH Sequence_DB_Filling.X	Default tag table	Bool	HMI_connection
GRAPH Sequence_DB_Transport Filling.X	Default tag table	Bool	HMI_connection
GRAPH Sequence_DB_Transport Labeling.X	Default tag table	Bool	HMI_connec...

11. Assign the "Position_Bottle(1)" tag to the function as shown in Step 9 as follows:
    - Open the tab overview.
    - Select the "Default tag table" under "HMI Panel" > "HMI Tags".
    - Assign the function the HMI tag "Position_Bottle(1)".

12. Save the project.

Result

You have successfully created the simulation for the tags.

- When Runtime starts, the value of the "Position_Bottle" and "Position_Bottle(1)" tags are incremented by 2 every cycle, independent of the progress of the CPU's user program.
- When the "S4 Transport Filling" step is started or stopped, the value of the "Position_Bottle" tag for specifying the position is set to "0".
- When the "S6 Transport Labeling" step is started or stopped, the value of the "Position_Bottle(1)" tag for specifying the position is set to "0".

For the display in runtime, this means that the bottles always appear at the initial position of the horizontal motion animation at the beginning of each step.

5.3.7 Creating a bar graph

Introduction

In the following, you will insert the "Bar" object in the "Production" root screen. From the bar graph, you can read the number of bottles already filled to see which has been recorded in the "GRAPH_Count_Bottle" PLC tag.

The "GRAPH_Count_Bottle" tag is always incremented by 1 during the filling of a bottle in the "S5 Filling" step of the GRAPH sequencer. After 10 bottling operations ("S5 Filling" step performed ten times), the sequencer starts again with the initial step.

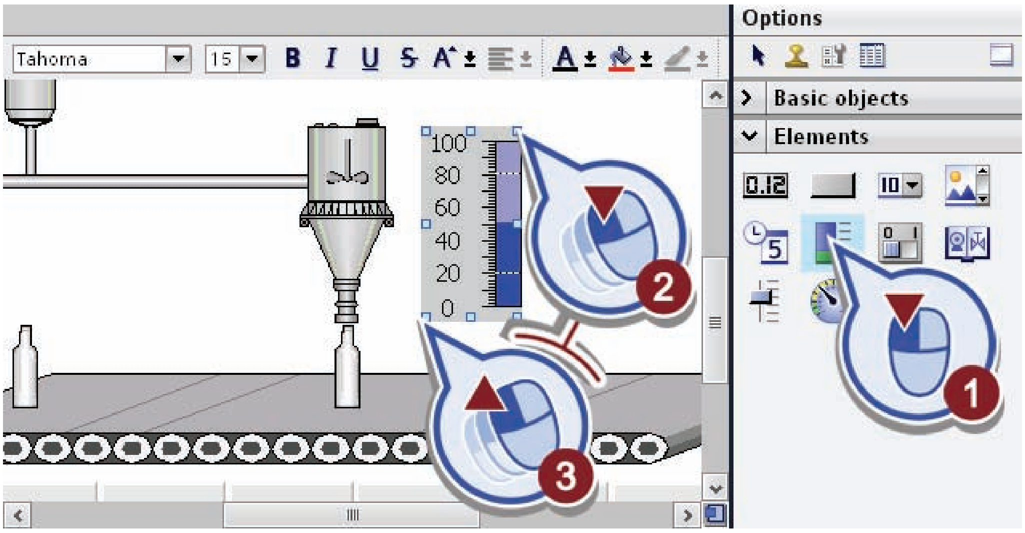
Requirement

You have created the "Production" root screen and the "GRAPH_Count_Bottle" tag.

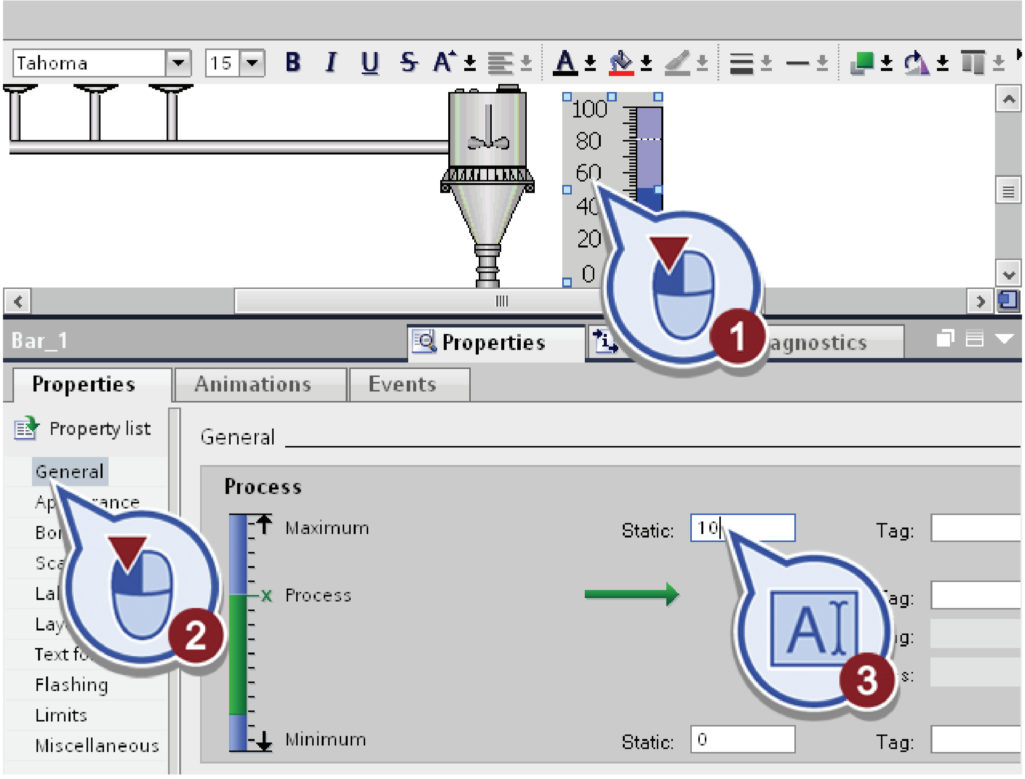
Procedure

To visualize the bar graph, follow these steps:

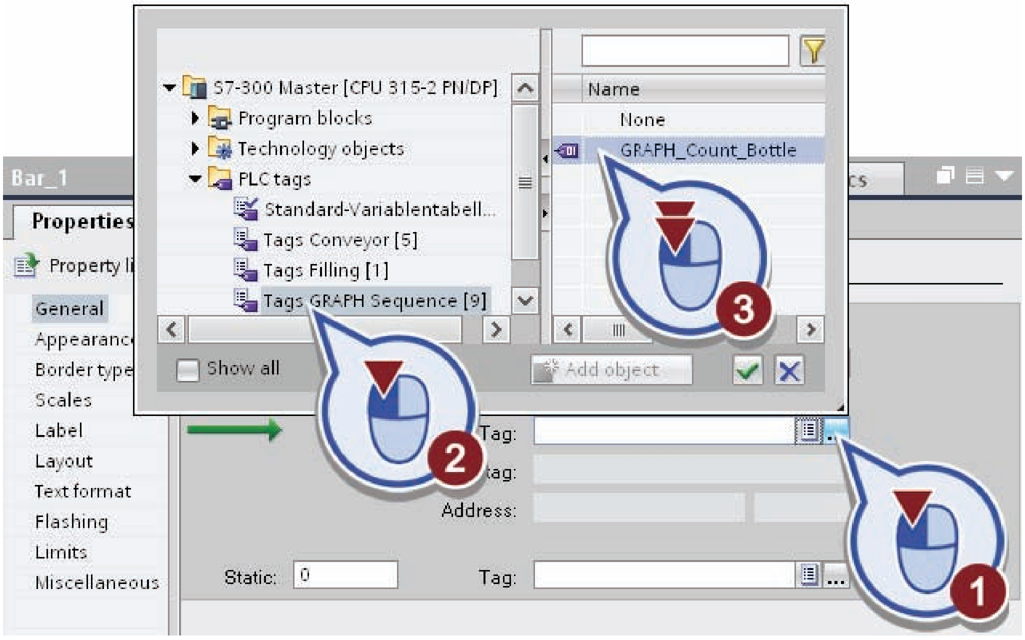
1. Open the "Elements" palette in the task card.
2. Insert the "Bar" object as follows:
   - Select the "Bar" object.
   - Scale the bar graph to approximately the same height as the filling station in the "Production" screen.



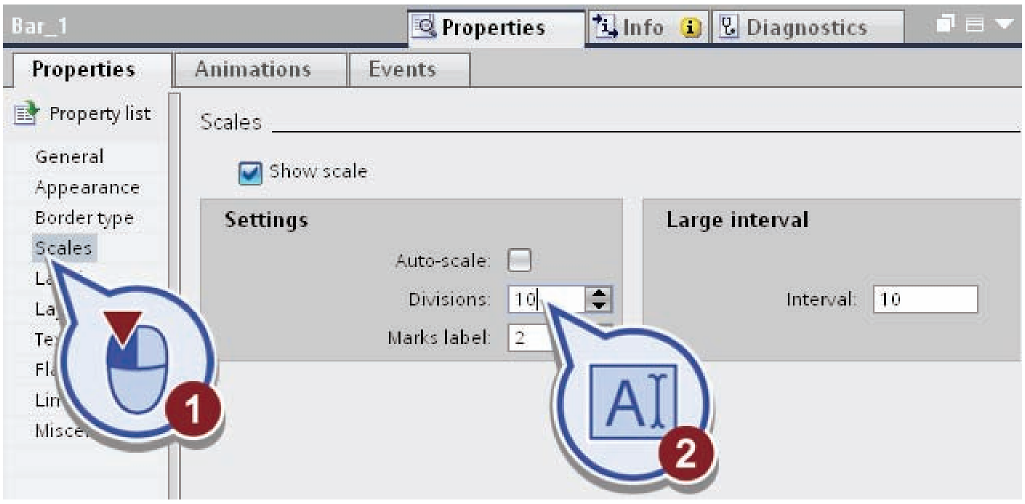
3. Set the maximum process value of the scale to "10" as follows:
   - Select the bar graph.
   - Open the "Properties" tab in the Inspector window.
   - Enter "10" as the maximum process value in the general properties.



4. Link the indicator value to the "GRAPH_Count_Bottle" tag as follows:
   - Open the tab overview.
   - Double-click to select the "GRAPH_Count_Bottle" tag under "S7-300 Master" > "PLC Tags" > "Tags GRAPH Sequence".



5. Open "Scales" dialog in the properties and change the number of scale divisions to "10".



☑ Show scale

Auto-scale: □

Divisions: 10

Marks label: 2

Interval: 10

6. Save the project.

Result

You have successfully inserted the bar graph in the "Production" root screen. When Runtime is active, the current value of the "GRAPH_Count_Bottle" PLC tag is shown in the bar graph.

5.3.8 Visualizing the pilot lights

Introduction

In the following, you will add two graphics objects for visualization of the pilot lights that indicate whether the "S2 Fill recipe ingredients" and "S5 Filling" steps are currently running:

- You position a pilot light next to the beverage tanks with the ingredients. It should flash when the "S2 Fill recipe ingredients" step is running.
- You position the second pilot next to the bar graph. It should flash when a bottle is being filled, i.e., when the "S5 Filling" step is running.

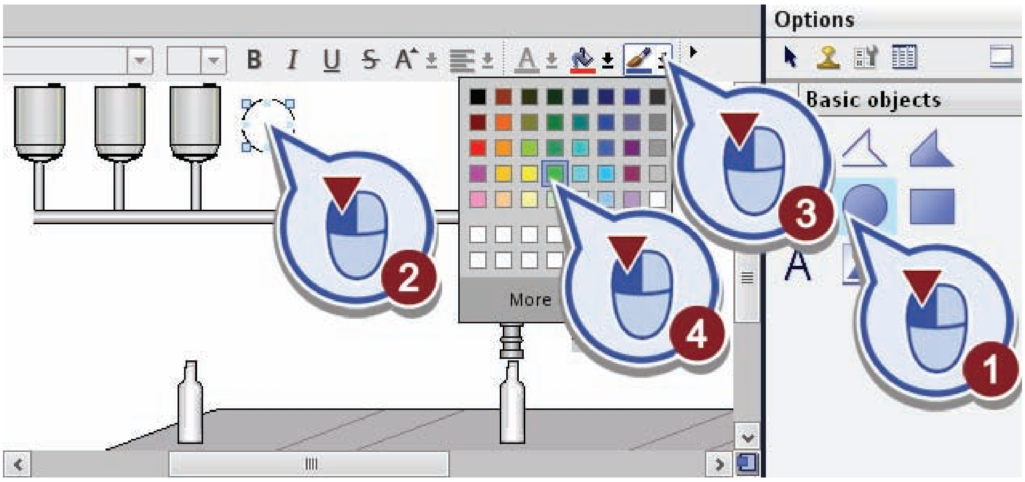
Requirement

You have created the "GRAPH_Sequence_DB" data block and the "Production" root screen.

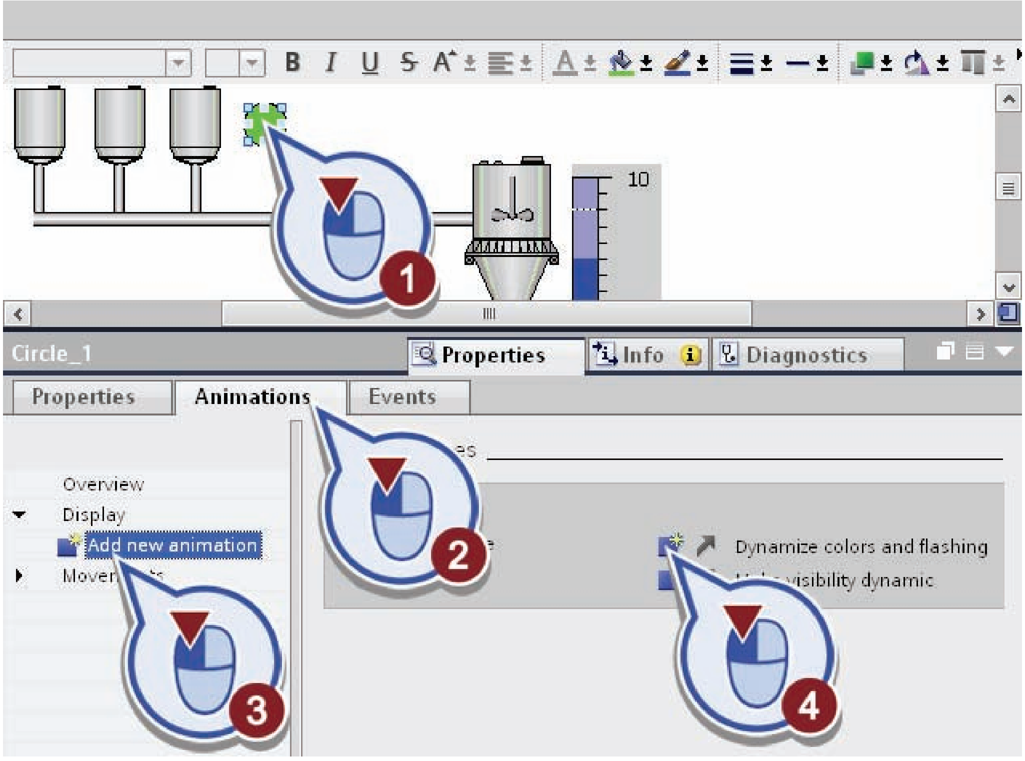
Procedure

To visualize the pilot lights, follow these steps:

1. Open the "Toolbox" palette in the "Basic objects" task card.
2. Insert a circle for visualization the first pilot light as follows:
   – Select the "Circle" basic object and drag it next to the ingredient tank in the root screen.
   – Select green as the background color for the circle.



3. Animate the first pilot lights as follows:
   - Select the circle.
   - Open the "Animations" tab in the Inspector window.
   - Select the "Add new animation" command under "Display".
   - Select the "Dynamize colors and flashing" function.



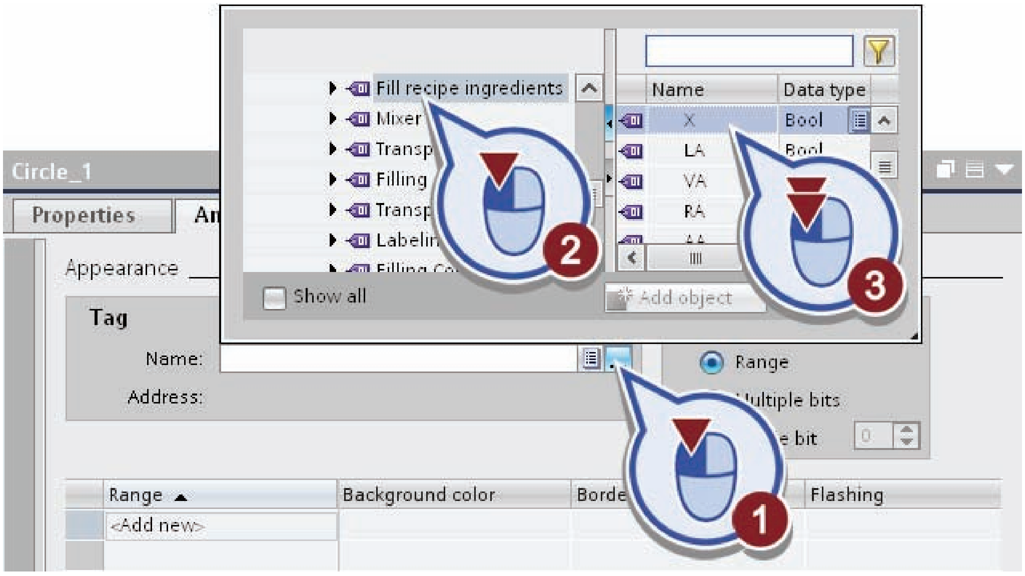
The "Appearance" dialog opens.

4. Link the object animation to the "X" status tag of the "Fill recipe ingrendients" GRAPH step as follows:

   – Open the tab overview.

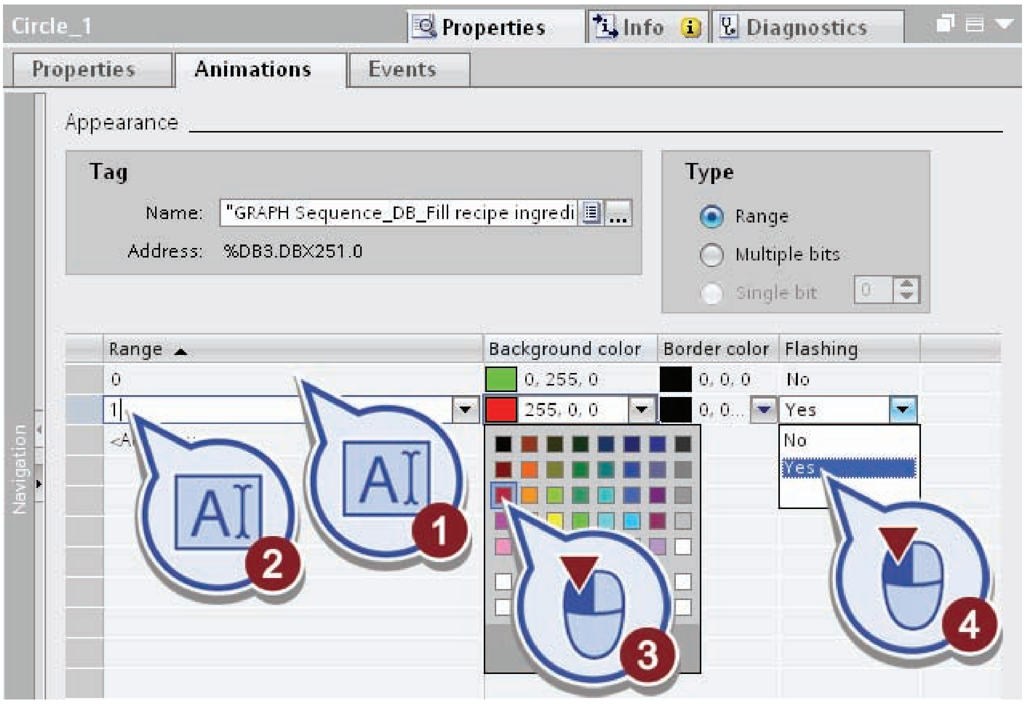
   – Select the "S2 Fill recipe ingredients" step under "S7-300 Master" > "Program blocks" > "GRAPH_Sequence_DB".

   – Link the "X" tag to the animation with a double click.



Name	Data type
X	Bool
LA	Bool
VA	
RA	
AA	

5. Enter "0" and "1" as the value range for the tag. For "1", select a different background color and enable the "Flashing" function.



Tag

Name: "GRAPH Sequence_DB_Fill recipe ingredi

Address: %DB3.DBX251.0

Type

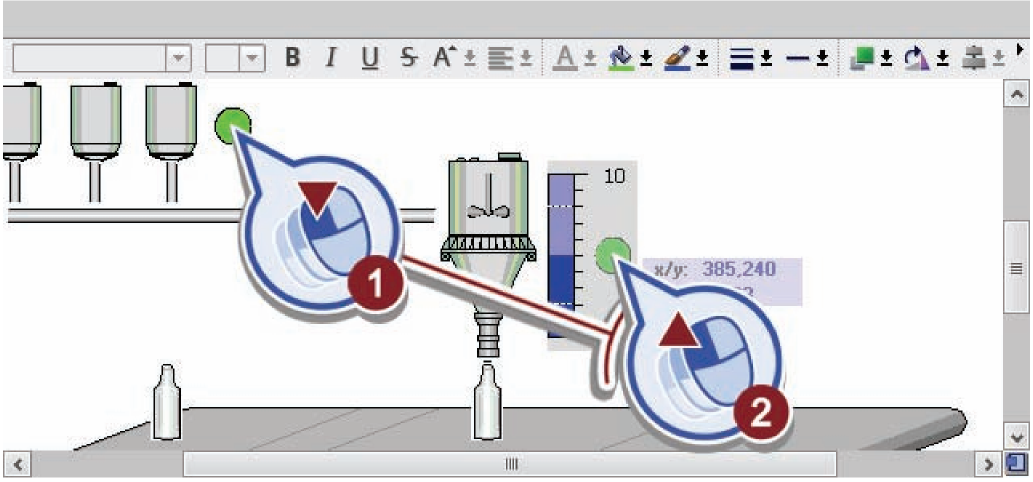
● Range

○ Multiple bits

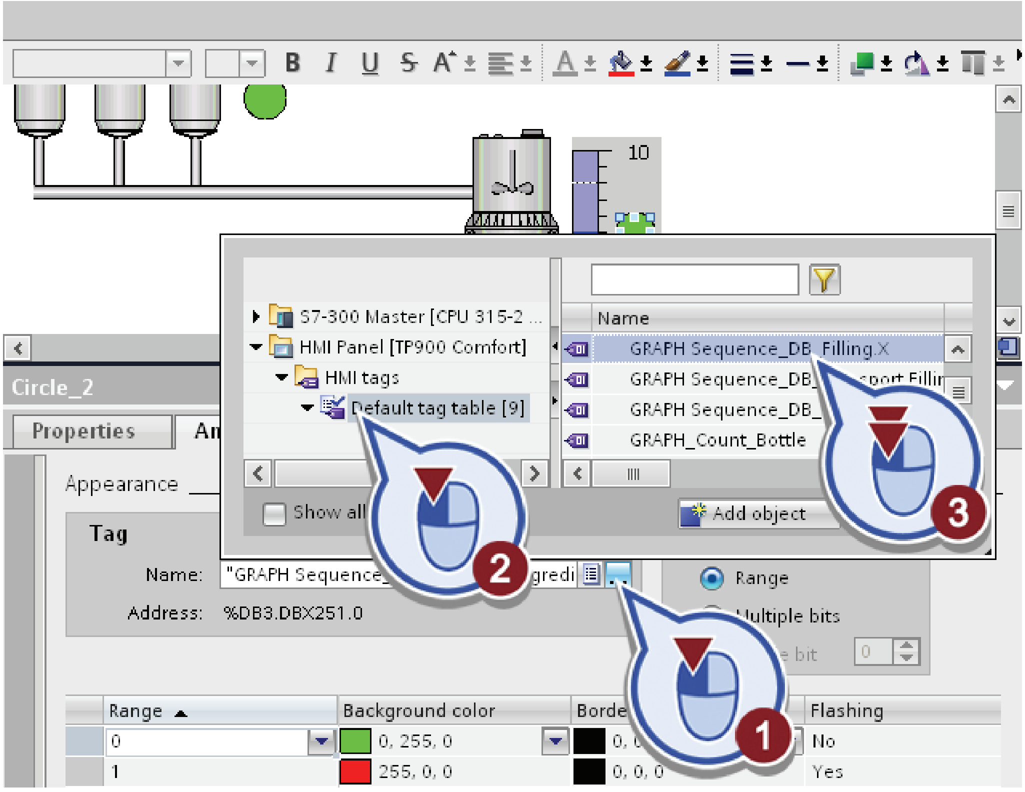
○ Single bit 0

Range	Background color	Border color	Flashing
0	🟩 0, 255, 0	■ 0, 0, 0	No
1	🟥 255, 0, 0	■ 0, 0...	Yes

6. Copy the first pilot light, by dragging it to the middle of the bar graph while holding down the <Ctrl> key.



7. Link the second pilot light to the "GRAPH Sequence_DB_Filling.X" HMI tag as follows:
   - Open the tab overview.
   - Double-click to select the "GRAPH Sequence_DB_Filling.X" HMI tag under "HMI Panel" > "HMI Tags" > "Default tag table".



Tag

Name: "GRAPH Sequence_...gredi

Address: %DB3.DBX251.0

Range	Background color	Borde	Flashing
0	🟩 0, 255, 0	0, 0	No
1	🟥 255, 0, 0	0, 0, 0	Yes

8. Save the project.

Result

You have successfully inserted the two pilot lights in the "Production" root screen.

- When the "S2 Fill recipe ingredients" step is executed, the first pilot light flashes next to the tanks with the ingredients.
- When the "S5 Filling" step is executed, the second pilot light flashes within the bar graph.

5.3.9 Visualizing the labeling machine

Introduction

In the following, you will add two objects for visualizing the labeling machine:

- A rectangle that flashes when the "S7 Labeling" step is running.
- An I/O field displaying the minimum date of expiration calculated by the "SCL - Best before date" program block.

Definition: I/O field

The "I/O field" object is used to enter and display process values.

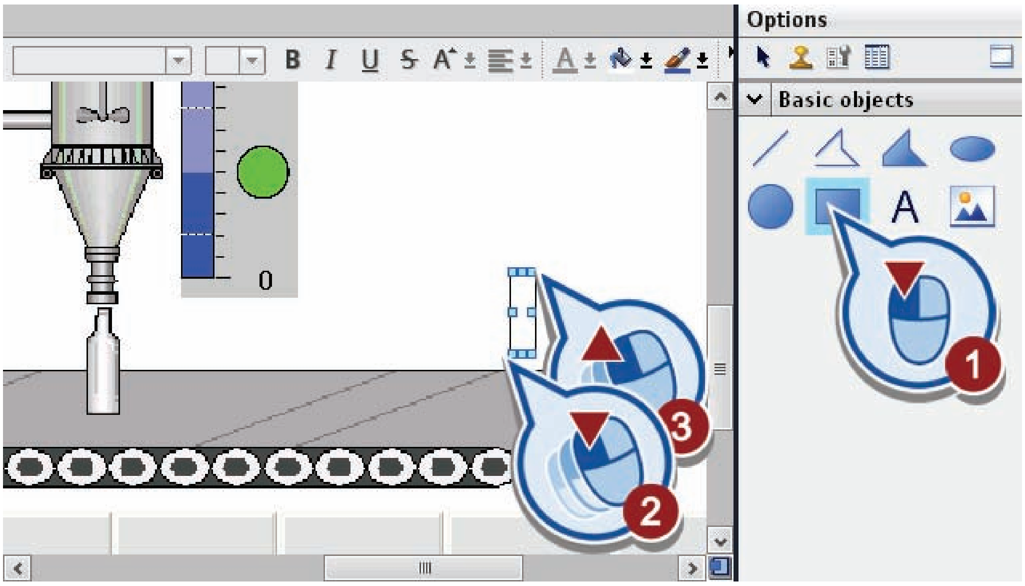
Requirement

You have created the "GRAPH_Sequence_DB" data block, the "Best_Before_Date" tag and the "Production" root screen.

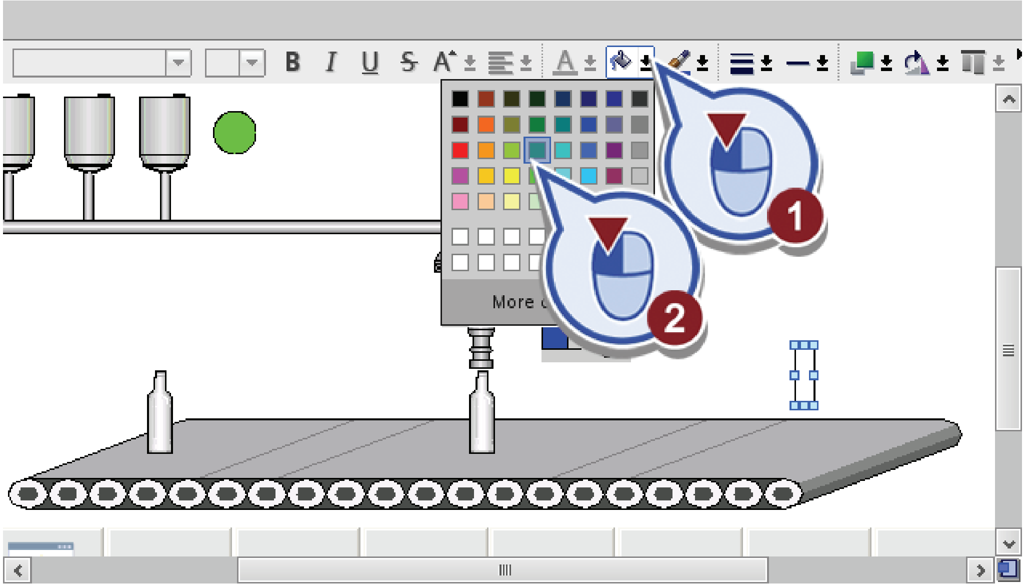
Procedure

To visualize the labeling machine, follow these steps:

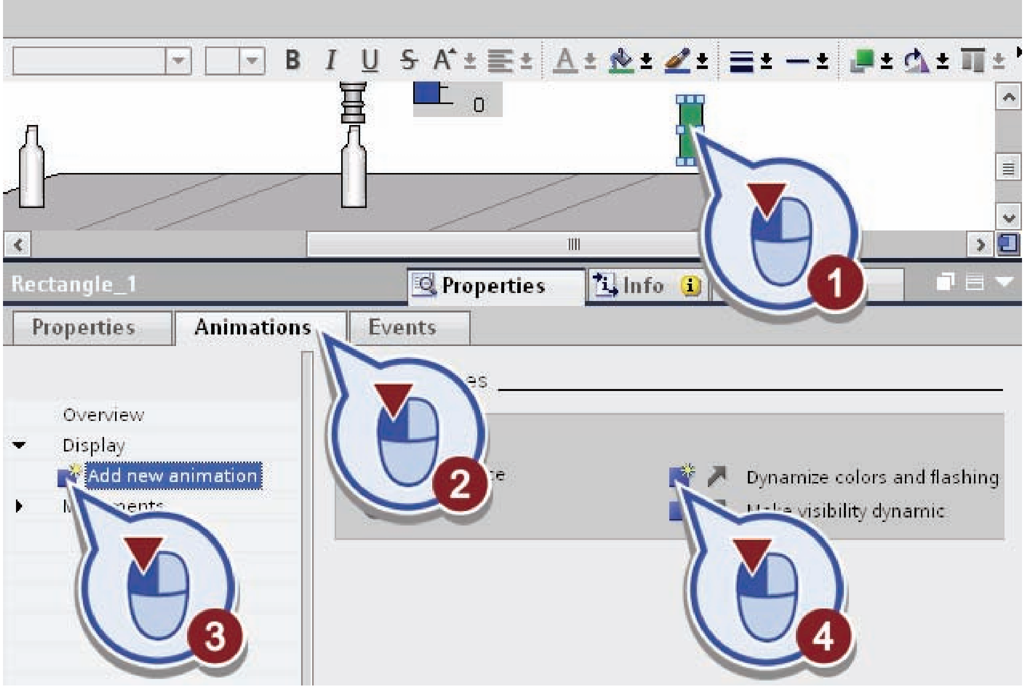
1. Open the "Basic objects" palette in the "Toolbox" task card.
2. Insert a rectangle as follows:
   – Select the "Rectangle" basic object.
   – Drag the rectangle to the right end of the conveyor belt.



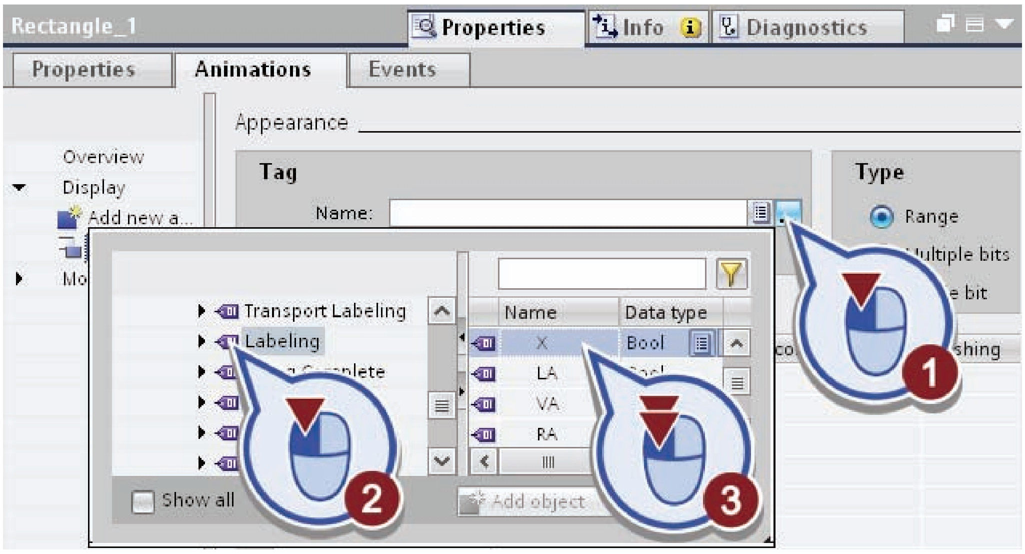
3. Select green as the background color.



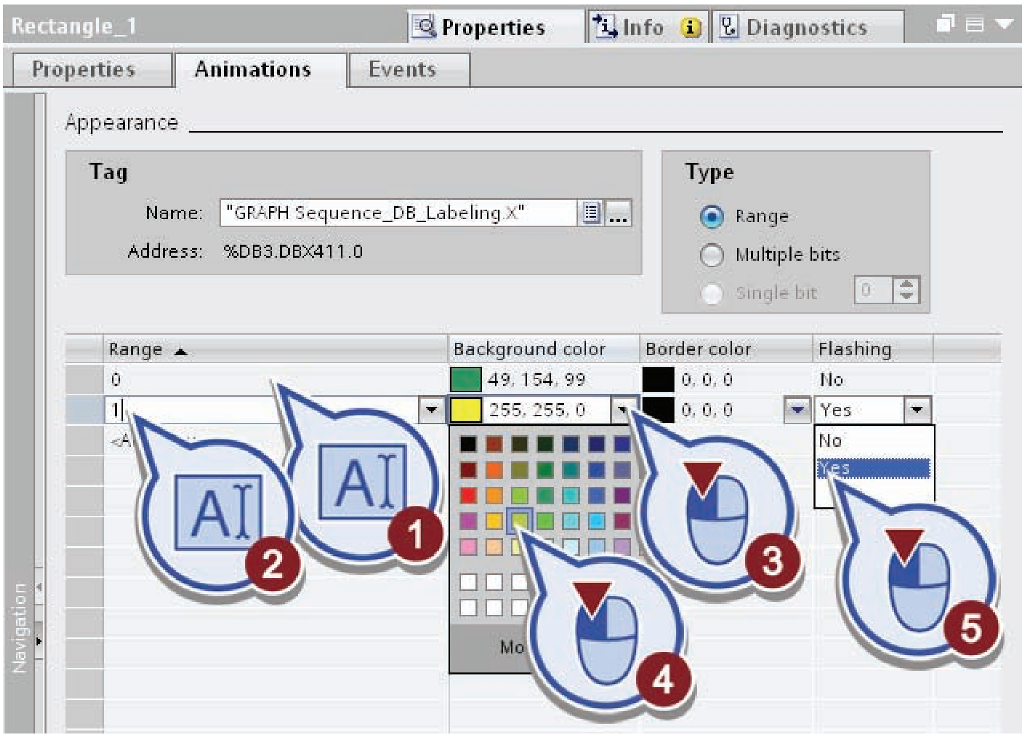
4. Animate the rectangle as follows:
   - Select the rectangle.
   - Open the "Animations" tab.
   - Click on "Add new animation" under "Display".
   - Select the "Dynamize colors and flashing" command.



5. Link the animation to the status tag of the "Labeling" step as follows:
   - Open the tab overview.
   - Select the "Labeling" step under "S7-300 Master" > "Program blocks" > "GRAPH_Sequence_DB".
   - Link the "X" status tag to the animation with a double click.



6. Enter "0" and "1" as the value range for the tag. For "1", select a different background color and enable the "Flashing" function.



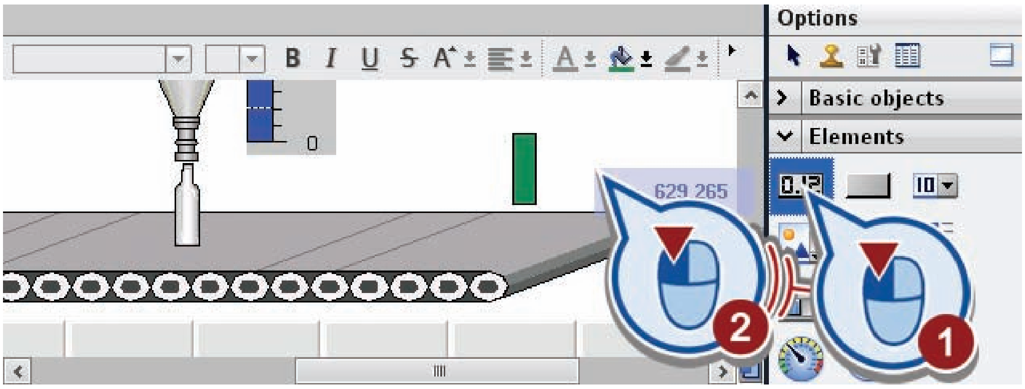
Name: "GRAPH Sequence_DB_Labeling.X"

Address: %DB3.DBX411.0

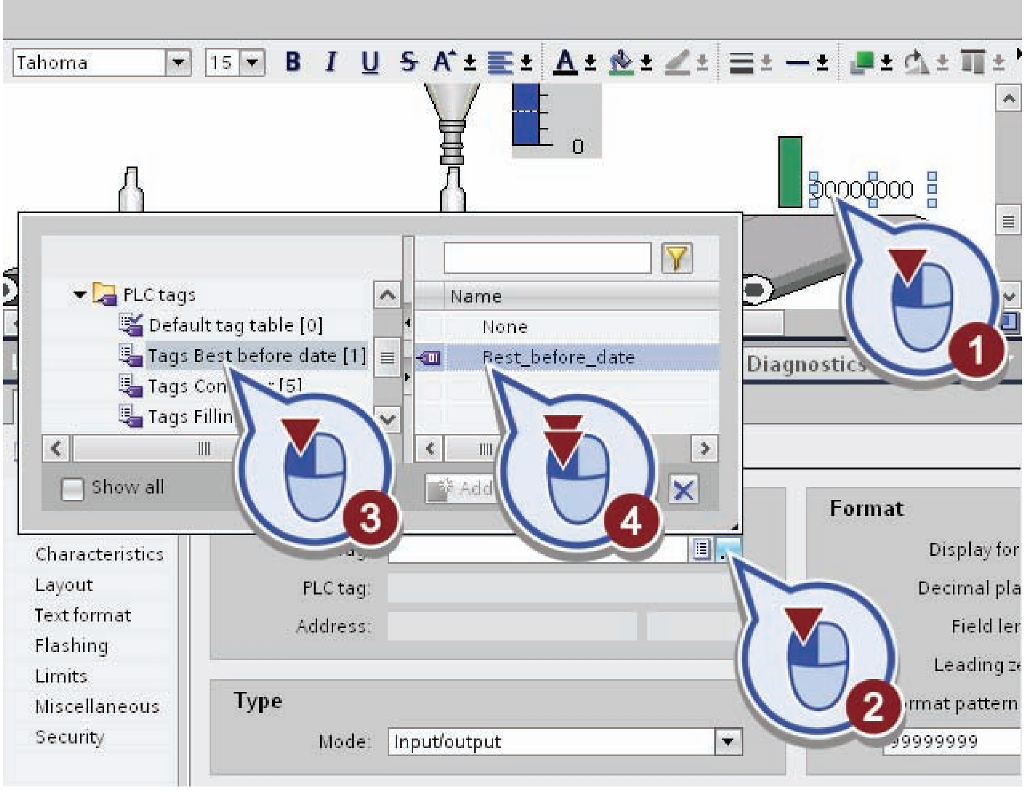
● Range
○ Multiple bits
○ Single bit 0

Range	Background color	Border color	Flashing
0	49, 154, 99	0, 0, 0	No
1	255, 255, 0	0, 0, 0	Yes

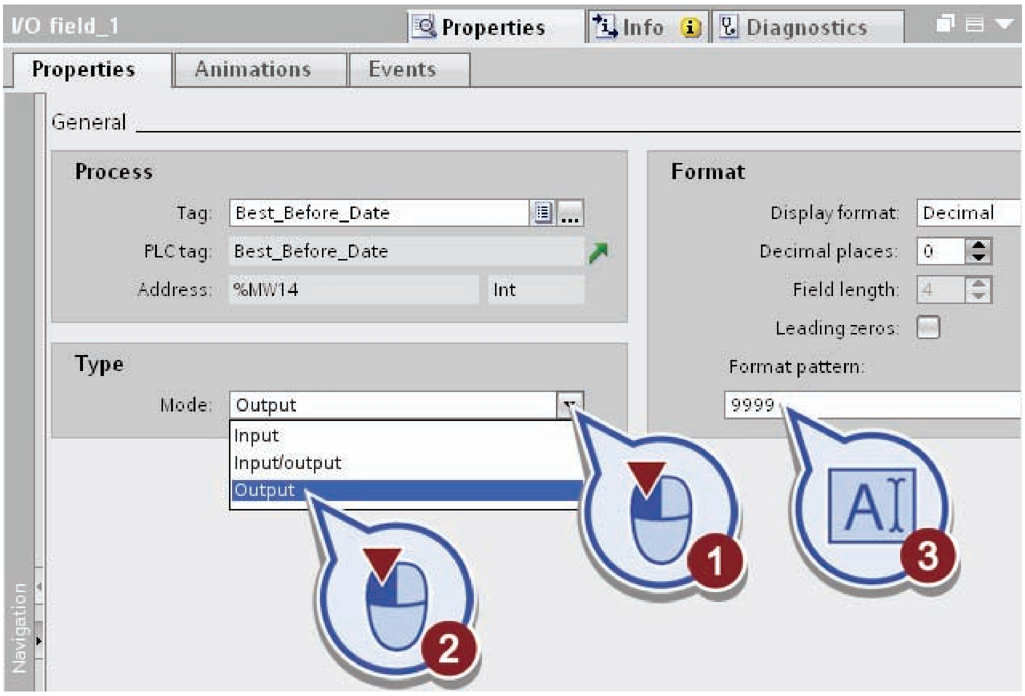
7. In the "Elements" tab, select the "I/O field" object and drag it into the root screen.



8. Link the I/O field to the tag of the calculated expiration date as follows:
   - Select the I/O field.
   - Open the tag overview under "General" in the "Properties" tab.
   - Select the "Tags Best_Before_Date" tag table under "S7-300 Master" > "PLC Tags".
   - Select the "Best_Before_Date" tag with a double click.



9. In the "Properties" tab, change the type of the I/O field to "Output" and the display format to four digits.



I/O field_1

Process
Tag: Best_Before_Date
PLC tag: Best_Before_Date
Address: %MW14 Int

Type
Mode: Output

Format
Display format: Decimal
Decimal places: 0
Field length: 4
Leading zeros: □
Format pattern: 9999

10. Save the project.

Result

You have successfully visualized the labeling machine.

- The rectangle flashes when the "S7 Labeling" step is active.
- The expiration date that is calculated by the "SCL – Best before date" program block based on the current system time now appears in the output field when you start Runtime.

5.3.10 Switch for activating the sequencer

Introduction

In the following, you will add a graphic object from the global library to visualize a switch, which will allow you to activate the GRAPH sequencer. The GRAPH FB called in the "OB1" program block is interconnected to the input parameters "OFF_SQ" and "INIT_SQ" by the "Start_GRAPH_Sequence" tag. This tag is used to activate and deactivate the sequencer:

- When there is a positive signal edge at the "OFF_SQ" input parameter, the sequencer is aborted, regardless of the state of execution.
- When there is a positive signal edge at the "INIT_SQ" input parameter, the sequencer is started with the initial step, regardless of whether another step has already been executed.

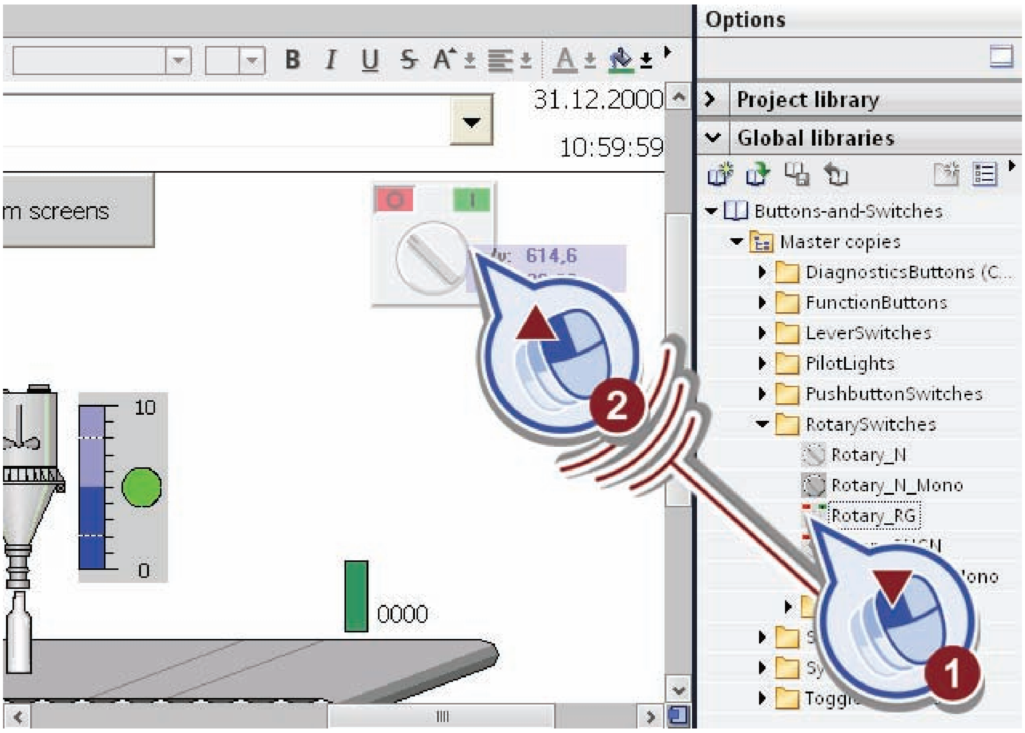
Requirement

You have programmed the GRAPH sequencer, called it in the organization block "Main [OB1]" and created the "Production" root screen.

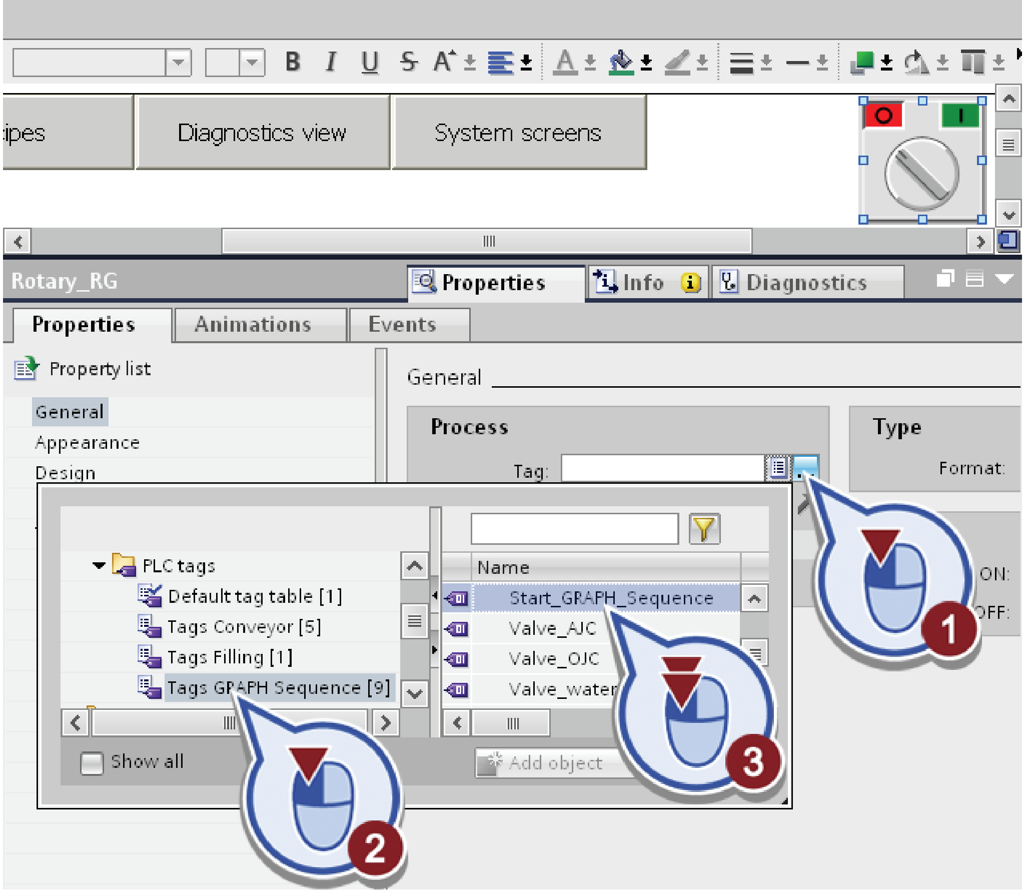
Procedure

To visualize the switch, follow these steps:

1. Open the "Global libraries" palette in the "Libraries" task card.
2. Insert the switch as follows:
   - Open the "RotarySwitches" folder in the "Buttons-and-Switches" > "Master copies" directory.
   - Insert the "Rotary_RG" switch into the upper right corner of the HMI screen.



3. Link the switch to the "Start_GRAPH_Sequence" tag as follows:
   - Select the switch.
   - Open the tag overview under "General" in the "Properties" tab.
   - Double-click the "Start_GRAPH_Sequence" tag under "S7-300 Master" > "PLC Tags" > "Tags_GRAPH_Sequence".



4. Save the project.

Result

You have inserted a switch and interconnected it with the "Start_GRAPH_Sequence" tag. When Runtime starts, the sequencer is initially not activated. When the switch is engaged, the "Start_GRAPH_Sequence" obtains the value "1". There is a positive signal edge at the "INIT_SQ" input parameter of the GRAPH FB and the sequencer is started.

5.3.11 Labeling objects in the HMI screen

Introduction

In the following, you will create the text fields to label the various objects in the "Production" root screen.

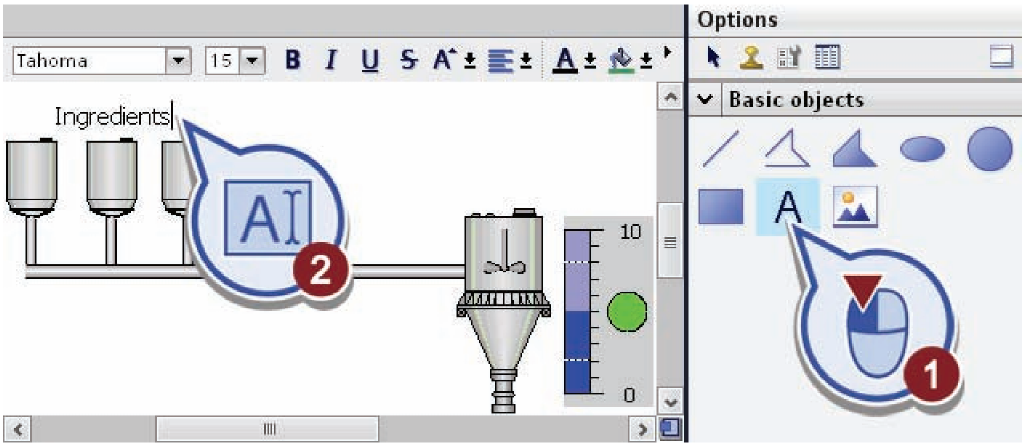
Requirement

You have inserted the required graphics in the "Production" root screen.

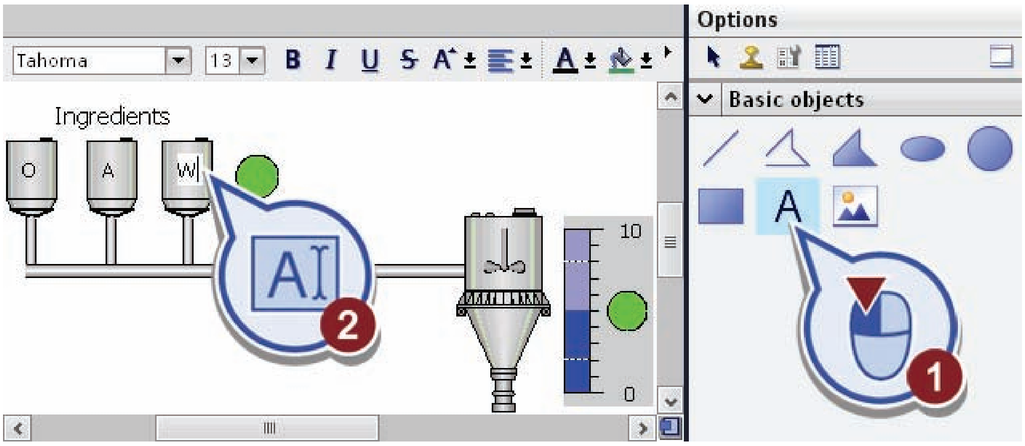
Procedure

To create the labels for the text fields, follow these steps:

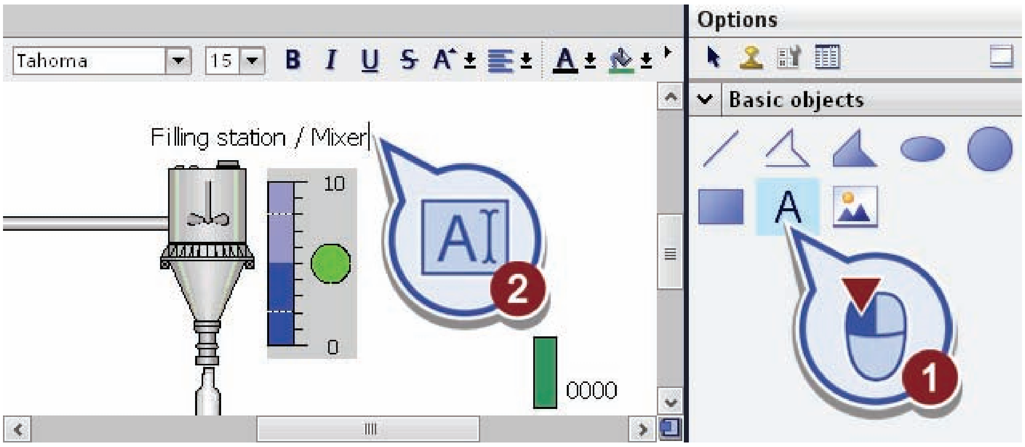
1. Open the "Basic objects" palette in the "Toolbox" task card.
2. Insert a text field as follows:
   - – Select the "Text field" object and insert it over the beverage tank in the "Production" root screen with click of the mouse.
   - – Enter "Ingredients" as text.



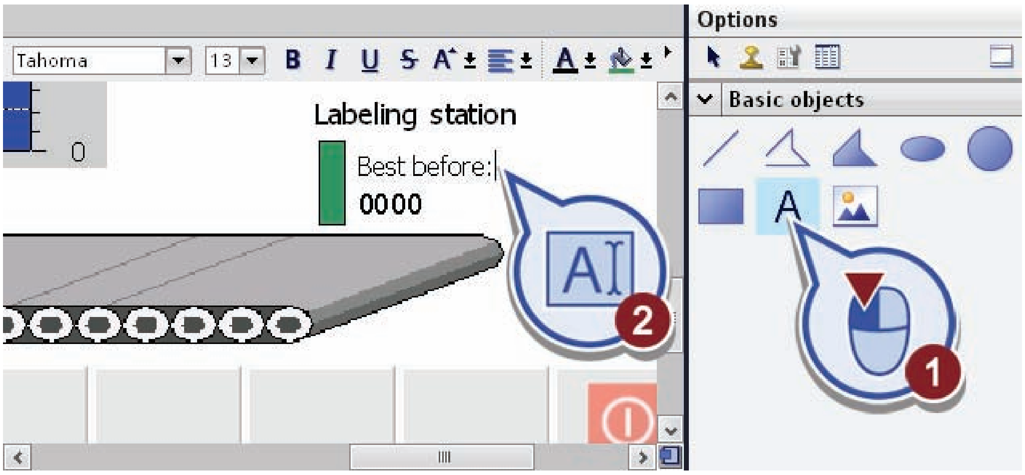
3. Create a text field for each beverage tank and label the tanks with the letters "O", "A" and "W".



4. Create a text field over the filling station and label it with "Filling station / mixer".



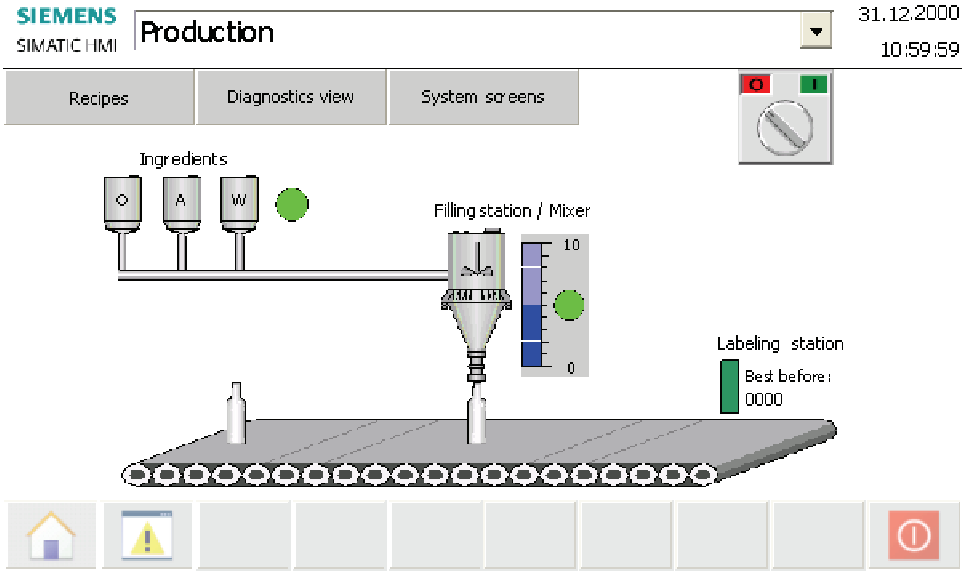
5. Create two text fields, one above the rectangle and one above the I/O field. Label them with "Labeling station" and "Best before:".



6. Save the project.

Result

You have successfully completed the "Production" root screen. You can use this screen to visualize the entire process, which you have programmed in the previous chapter.



5.4 Creating the "Recipes" screen

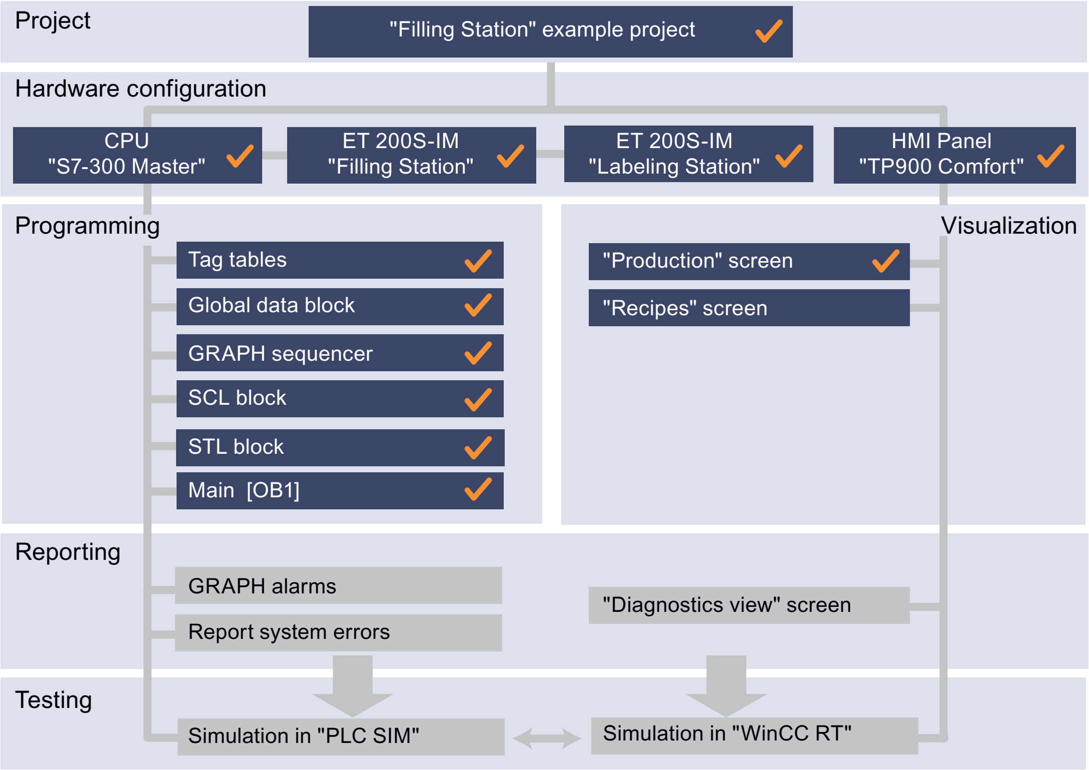
5.4.1 Basics of using recipes

Introduction

In the following, the structure of a recipe is explained as it is used in the "Filling Station" example project.

Project progress

The following graphic shows you which step you perform next:

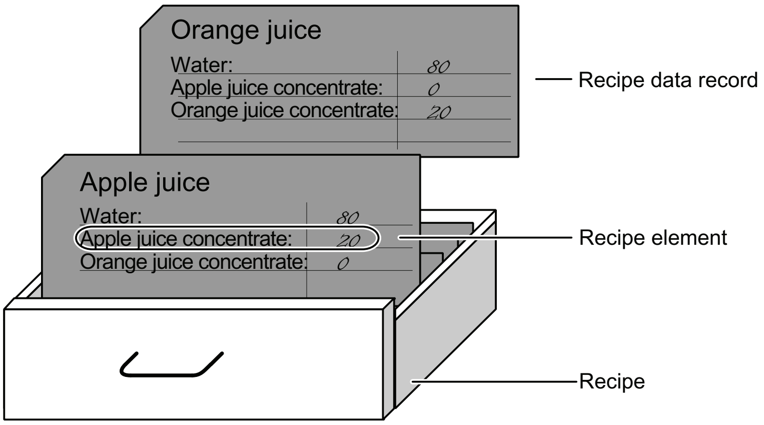


Recipe

There may be several different recipes in an HMI device. A recipe can be compared to an index card box that contains several index cards. All the data for each production variant is contained on a single index card.

In the "Filling Station" example project, a recipe with three recipe records is created for the production of each beverage variant. The beverage variants are apple juice, orange juice and water.

The recipe contains three recipe records that are needed for the production of the various beverage variants.


Orange juice
Water: 80
Apple juice concentrate: 0
Orange juice concentrate: 20
Recipe data record
Apple juice
Water: 80
Apple juice concentrate: 20
Orange juice concentrate: 0
Recipe element
Recipe

Recipe data record

Each index card represents a recipe data record needed to produce a beverage variant. The records differ in the value of the individual recipe elements. In the "Filling Station" example project, the recipe record for apple juice contains for example the recipe elements water (80%), apple juice concentrate (20%) and orange juice concentrate (0%).

Recipe elements

Each line on the index card corresponds to one recipe element. In the "Filling Station" example project, three recipes elements are used (water, apple juice concentrate and orange juice concentrate). The recipe elements are filled in the "S2 Fill recipe ingredients" step of the GRAPH sequencer.

5.4.2 Creating a recipe

Introduction

In the following, you will create a recipe for the production of the three beverage variants, water, apple juice and orange juice. The recipe elements and the recipe records within this recipe are defined in the following sections.

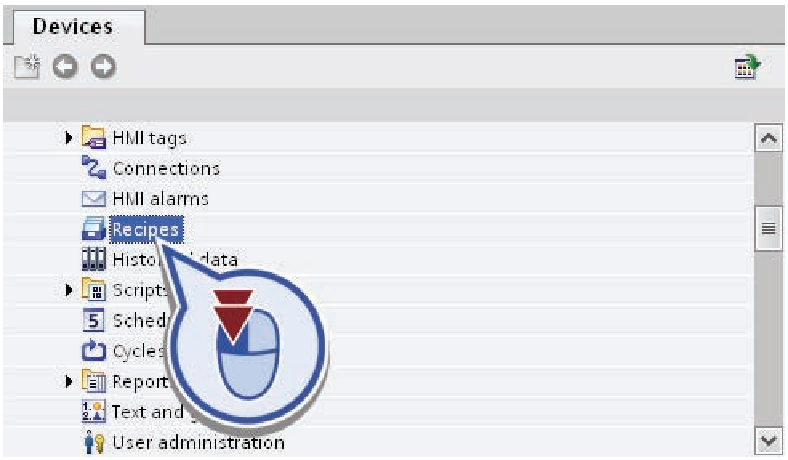
Requirement

You have created the "TP900 Comfort" HMI panel.

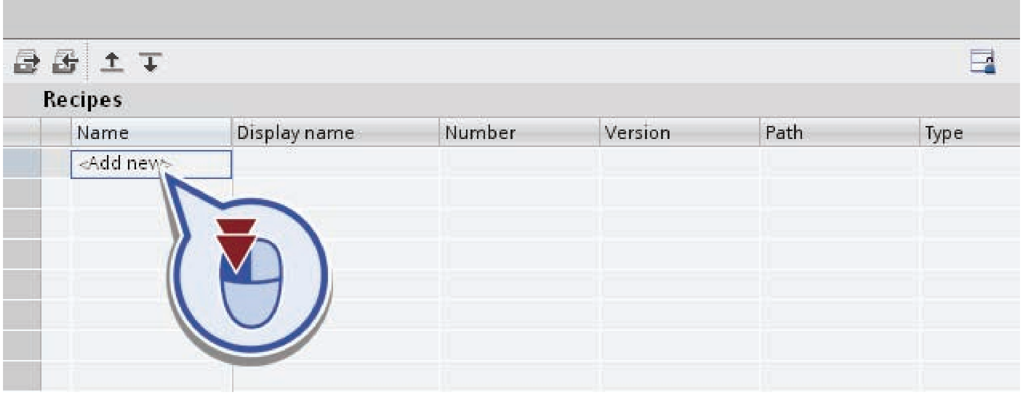
Procedure

To create the recipe follow these steps:

1. Open the "Recipes" dialog in the project tree.

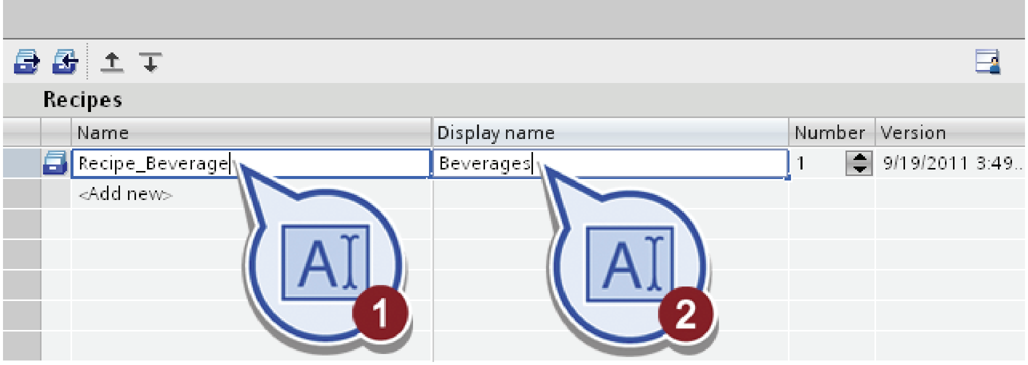


2. Double-click "<Add new>" in the "Recipes" dialog.


Recipes

Name	Display name	Number	Version	Path	Type
<Add new>					

3. Enter "Recipe_Beverage" in the "Name" column and "Beverages" in the "Display name" column.


Recipes

Name	Display name	Number	Version
Recipe_Beverage	Beverages	1	9/19/2011 3:49..
<Add new>			

4. Save the project.

Result

You have successfully created a recipe. The name of the recipe is used to identify the recipe in the process. The display name is shown in the process above the recipe view.

5.4.3 Creating a recipe element

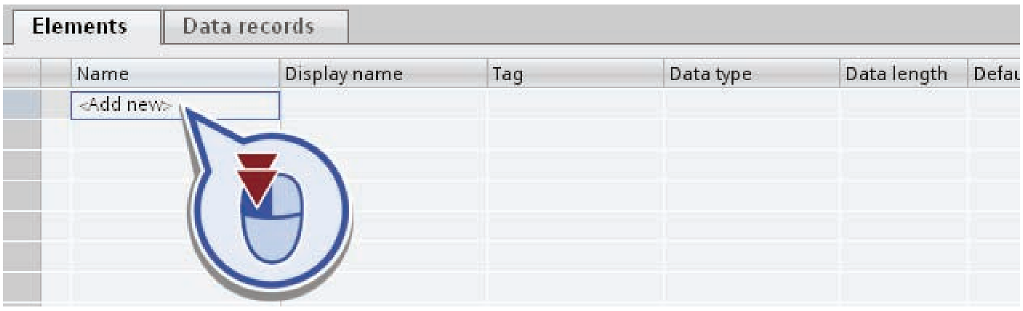
Introduction

In the following, you will create the three elements for the recipe data records. The "Water", "Concentrate_Apple_Juice" and "Concentrate_Orange_Juice" recipe elements are the ingredients that are used in the recipe data records. The values of the recipe elements are stored in the global data block of the CPU and are linked to the recipe data records via tags.

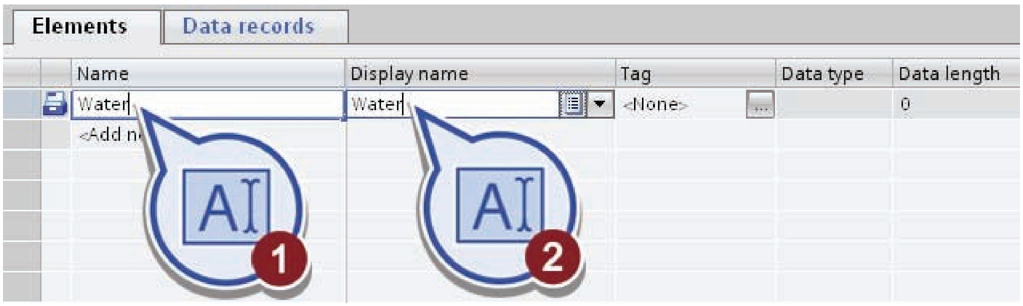
Procedure

To create the recipe elements follow these steps:

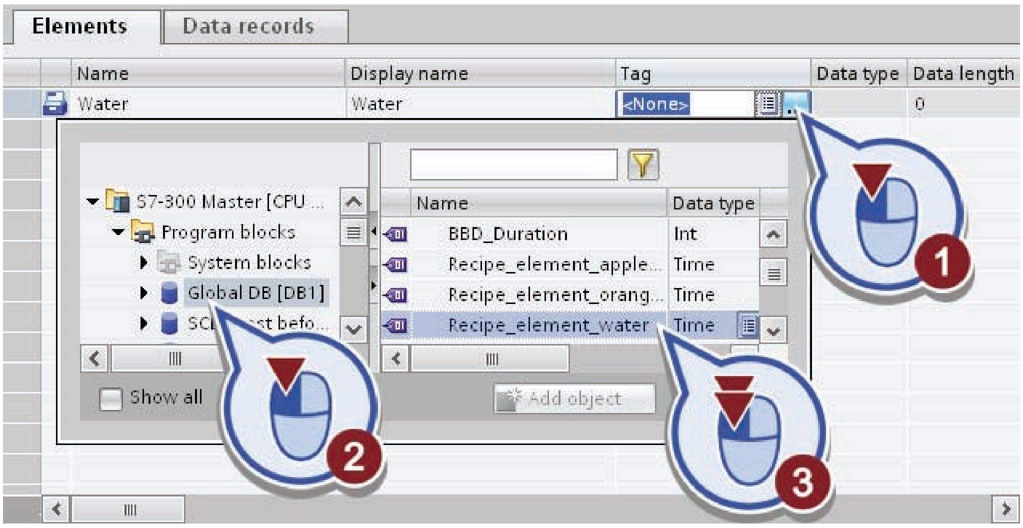
1. Double click "<Add new>" in the "Name" column of the "Elements" tab.



2. Enter "Water" in the "Name" column and "Water" in the "Display name" column.



3. Link the recipe element using the according tag of the global data block as follows:
   – Open the tag overview in the "Tag" column.
   – Select the "Global_DB" data block under "S7-300 Master" > "Program blocks".
   – Link the "Recipe_element_water" tag to the recipe element with a double click.



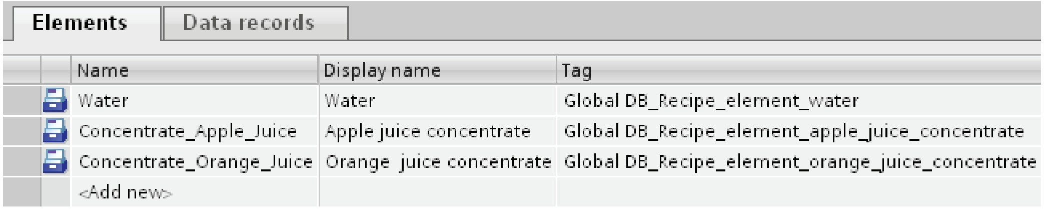
Name	Display name	Tag	Data type	Data length
Water	Water	<None>		0

Name	Data type
BBD_Duration	Int
Recipe_element_apple...	Time
Recipe_element_orang...	Time
Recipe_element_water	Time

4. Create a second recipe element with the following properties:
   – Name: "Concentrate_Apple_Juice"
   – Display name: "Apple juice concentrate"
   – Tag: "Recipe_element_concentrate_apple_juice" of the "Global_DB" global data block
5. Create a third recipe element with the following properties:
   – Name: "Concentrate_Orange_Juice"
   – Display name: "Orange juice concentrate"
   – Tag: "Recipe_element_concentrate_orange_juice" of the "Global_DB" global data block
6. Save the project.

Result

You have successfully created the three recipe elements for the recipe record. The following figure shows the defined recipe elements.



Name	Display name	Tag
Water	Water	Global DB_Recipe_element_water
Concentrate_Apple_Juice	Apple juice concentrate	Global DB_Recipe_element_apple_juice_concentrate
Concentrate_Orange_Juice	Orange juice concentrate	Global DB_Recipe_element_orange_juice_concentrate
<Add new>		

Each of the three recipe elements are linked with a tag of a global data block. This forms the connection between the recipe function of the HMI panel and the program of the PLC. The values for each recipe element are read in the "S2 Fill recipe ingredients" GRAPH step to control the fill amount at the valves of the tanks with the ingredients.

5.4.4 Creating recipe data records

Introduction

In the following, you will create the three recipe data records.

- The "Recipe_Water", "Recipe_Apple_Juice" and "Recipe_Orange_Juice" recipe data records contain the mixing ratios for the individual beverage variants. The recipe elements created in the previous section are listed in separate columns in the "Data records" tab. Enter values for the mixing ratios of the beverage variants in these columns.
- The various mixing ratios result from the different durations for which a valve is opened at the tank of the respective ingredient during the filling process. The values for the individual recipe elements are stored as a "Time" data type in the corresponding global data block. A value of "1000" in the "Time" data type corresponds to a filling time of one second.

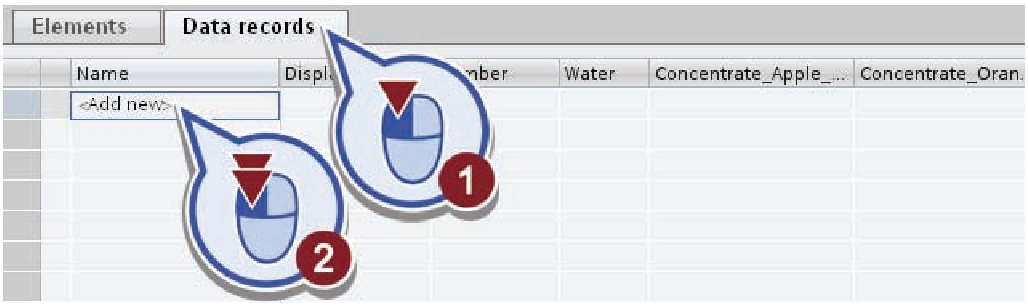
Requirement

You have created the three recipe elements.

Procedure

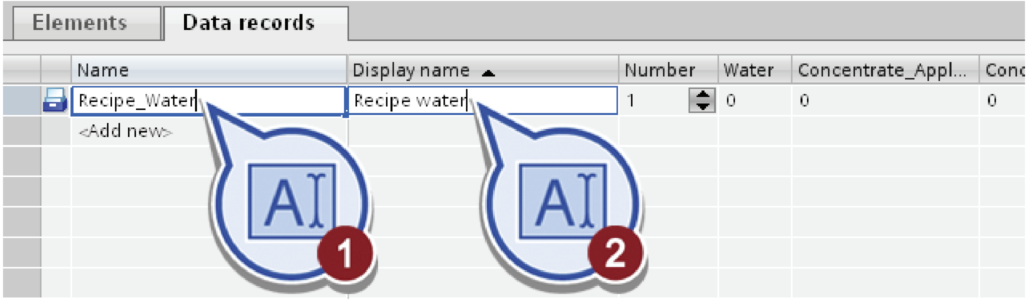
To create the recipe records follow these steps:

1. Open the "Data records" tab and double-click "<Add new>" in the "Name" column.



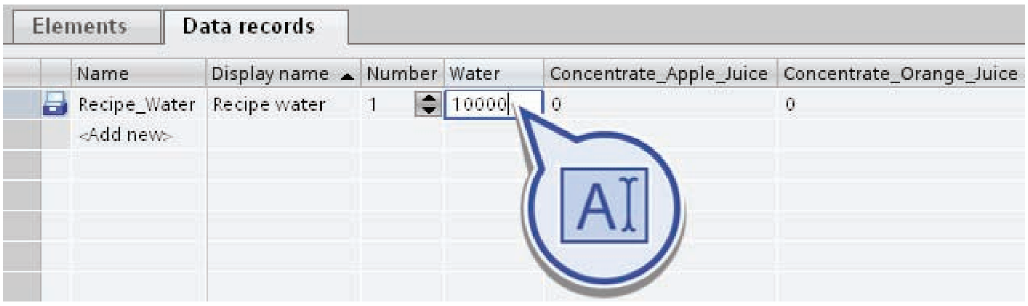
Name	Displ	mber	Water	Concentrate_Apple_...	Concentrate_Oran.
<Add new>					

2. Enter "Recipe_Water" in the "Name" column and "Recipe water" in the "Display name" column.



Name	Display name ▲	Number	Water	Concentrate_Appl...	Conc
Recipe_Water	Recipe water	1	0	0	0
<Add new>					

3. Assign a value of "10000" to the "Water" recipe element. Leave the values for the remaining recipe elements, "Concentrate_Apple_Juice" and "Concentrate_Orange_Juice", at "0".



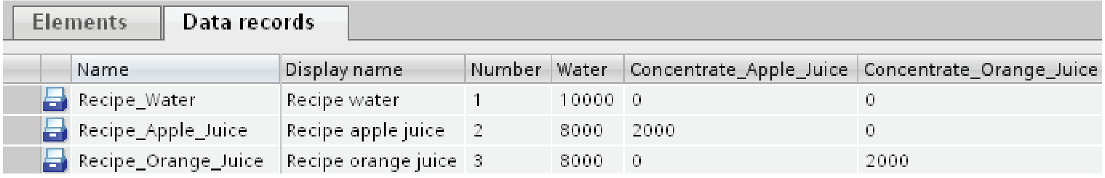
Name	Display name ▲	Number	Water	Concentrate_Apple_Juice	Concentrate_Orange_Juice
Recipe_Water	Recipe water	1	10000	0	0
<Add new>					

4. Click "<Add new>" to create a second recipe record with the following properties:
   - Name: "Recipe_Apple_Juice"
   - Display name: "Recipe apple juice"
   - Water: "8000"
   - Concentrate_Apple_Juice: "2000"
   - Concentrate_Orange_Juice: "0"

5. Click "<Add new>" to create a third recipe record with the following properties:
   - Name: "Recipe_Orange_Juice"
   - Display name: "Recipe orange juice"
   - Water: "8000"
   - Concentrate_Apple_Juice: "0"
   - Concentrate_Orange_Juice: "2000"
6. Save the project.

Result

You have successfully created the three recipe data records and entered the appropriate values for the mixing ratios.



Name	Display name	Number	Water	Concentrate_Apple_Juice	Concentrate_Orange_Juice
Recipe_Water	Recipe water	1	10000	0	0
Recipe_Apple_Juice	Recipe apple juice	2	8000	2000	0
Recipe_Orange_Juice	Recipe orange juice	3	8000	0	2000

- When the "Recipe orange juice" data record is selected later in Runtime, the value "8000" is written for the "Water" recipe element and the value "2000" is written for the "Concentrate_Orange_Juice" recipe element in the respective tag of the global data block.
- Accordingly, the value at the tank for the "Water" ingredient is opened for 8 seconds and the valve at the tank for the "Orange juice concentrate" ingredient is opened for 2 seconds in the "S2 Fill recipe ingredients" GRAPH step.

5.4.5 Creating a recipe view

Introduction

In the following, you will create a recipe view in the "Recipes" HMI screen and link it to recipe you have previously created. You can use the recipe view to select the recipe data records in Runtime. Based on this selection, the respective values of the recipe elements are written in the global data block.

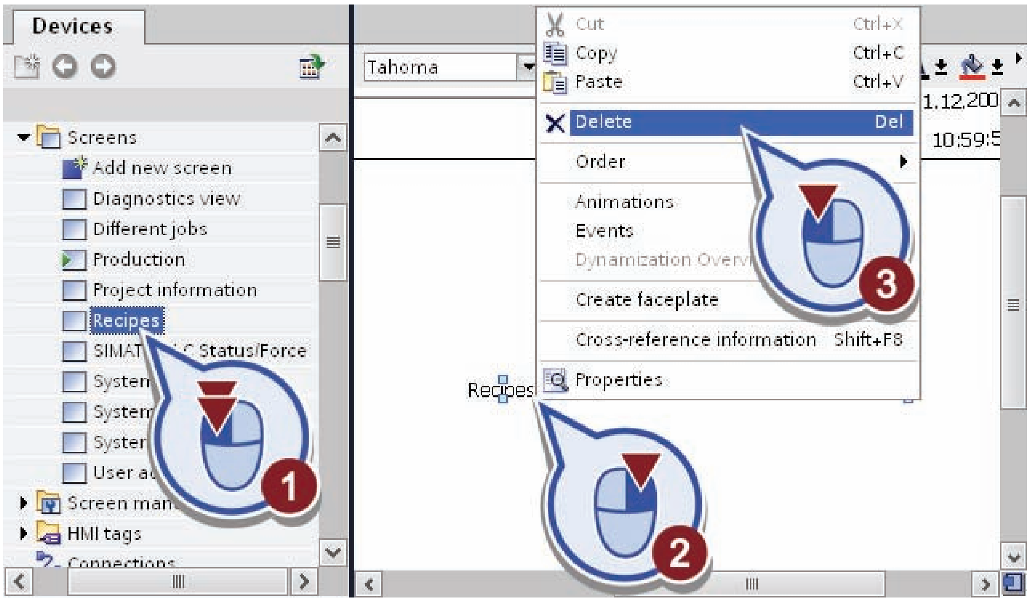
Requirement

You have created the recipe with the recipe data records and elements.

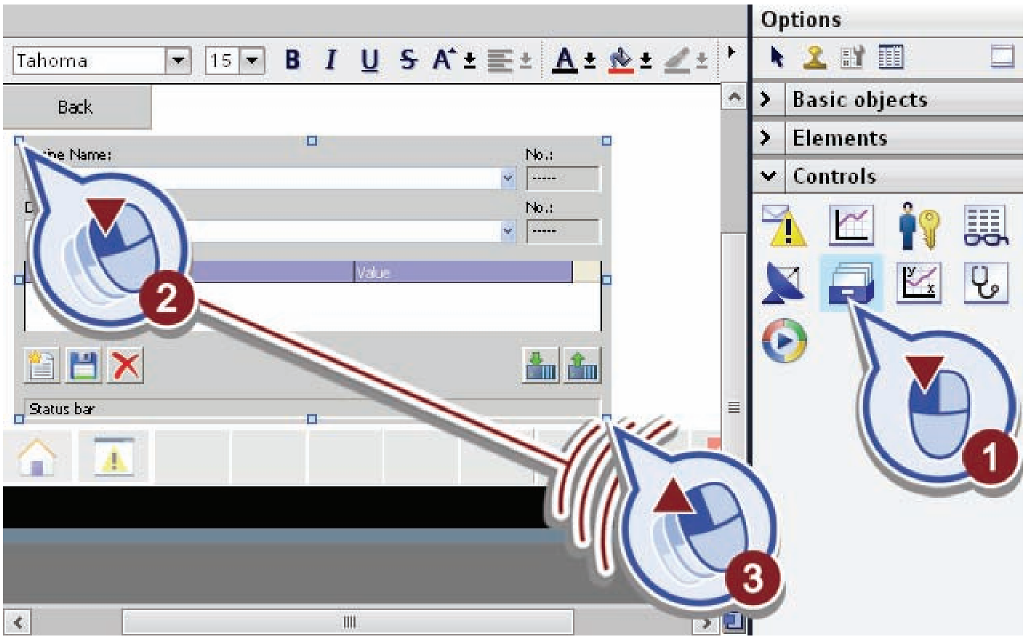
Procedure

To create the recipe view, follow these steps:

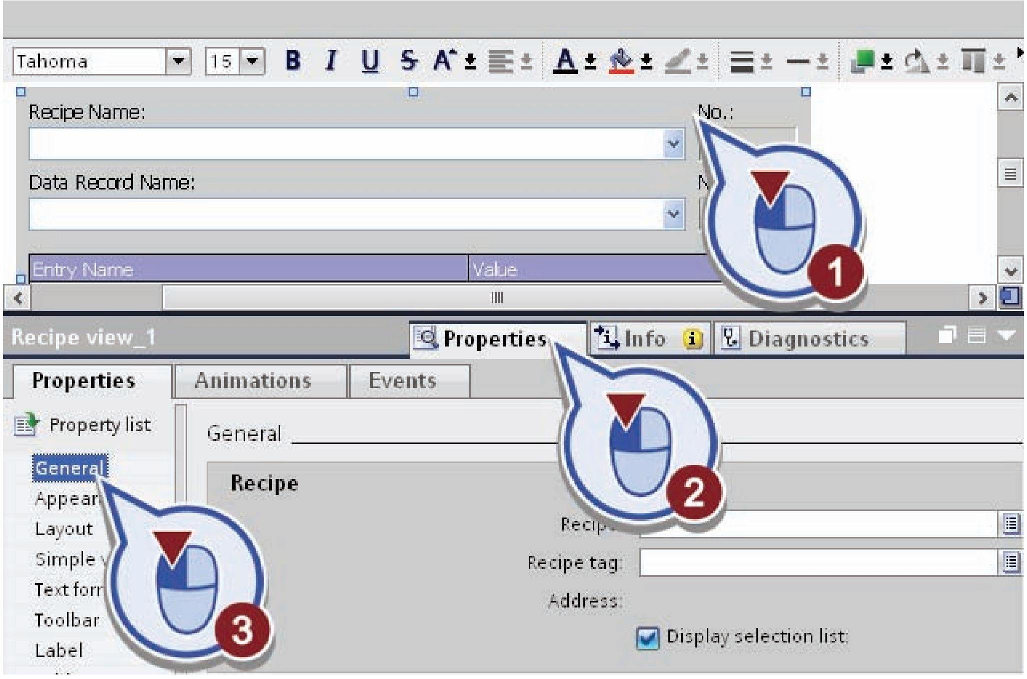
1. Open the "Recipes" HMI screen in the "Screens" folder of the project tree and delete the text field.



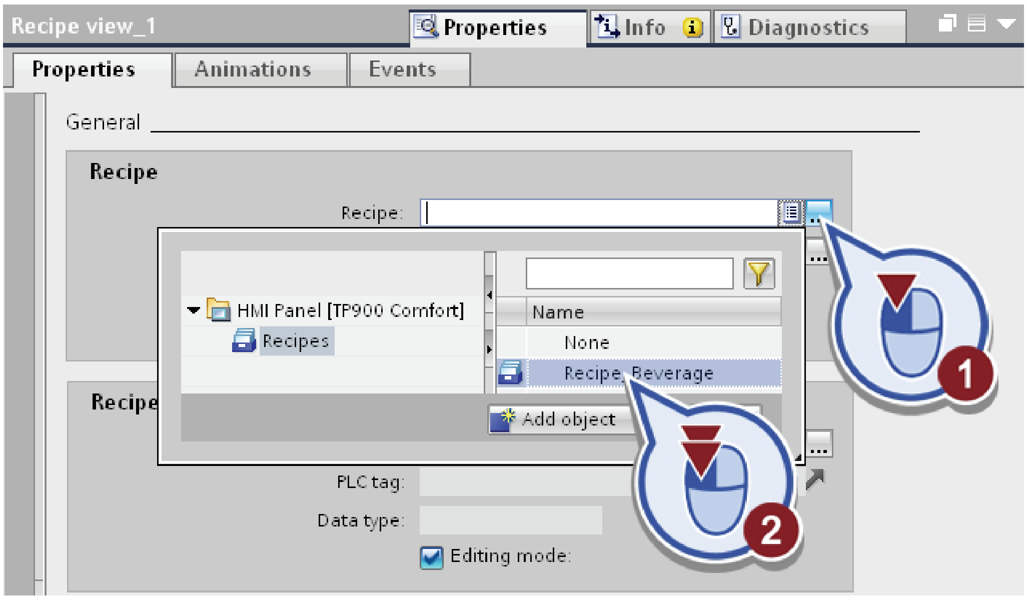
2. Open the "Controls" palette in the "Toolbox" task card.
3. Insert the recipe view by selecting "Recipe view" and dragging into the screen.



4. Select the Recipe view and open the "General" dialog in the Inspector window under the "Properties" tab.



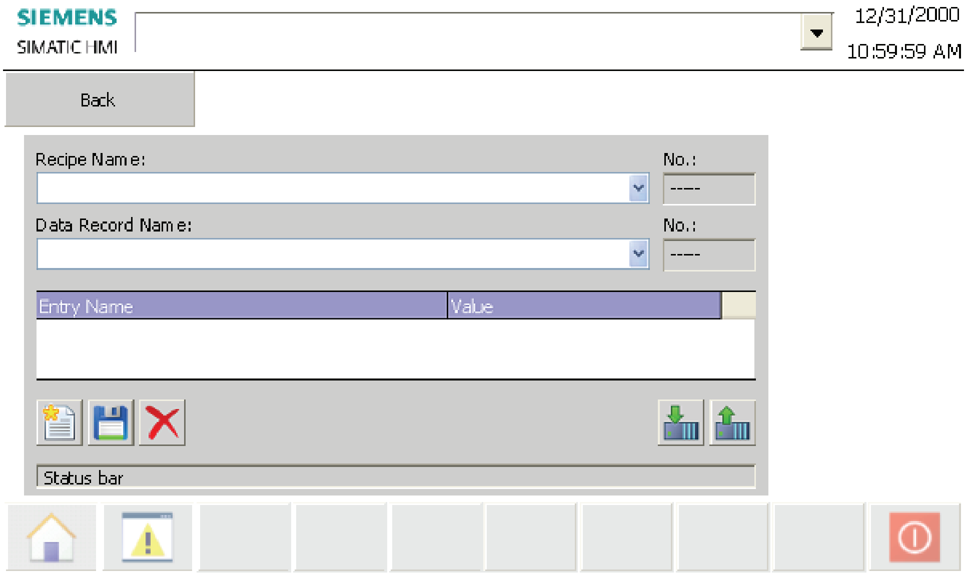
5. Select the "Recipe_Beverage" recipe from the "Recipe" drop-down list.



6. Save the project.

Result

You have successfully created the recipe view in the "Recipes" HMI screen.



- As soon as Runtime is started, you can navigate in this HMI screen using the "Recipes" button in the "Production" root screen.
- Select the desired recipe record in the "Recipes" HMI screen.
- You can navigate back to the "Production" root screen using the "Back" button.

5.4.6 Creating the specification for the expiration date

Introduction

In the following, you will create a slider control next to the recipe view to specify the expiration date. The expiration date is written to the global data block of the CPU with a tag and used to calculate the expiration date in the "SCL- Best before date" data block. The duration is added to the year of the system clock and displayed in the "Production" root screen.

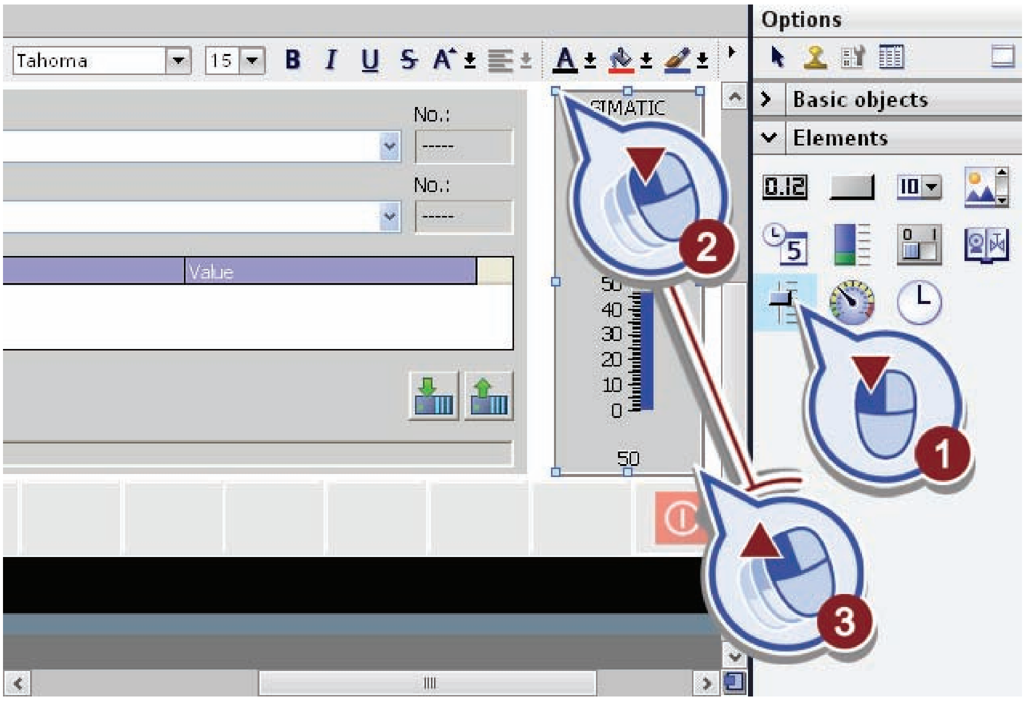
Requirement

You have created the "Recipes" HMI screen.

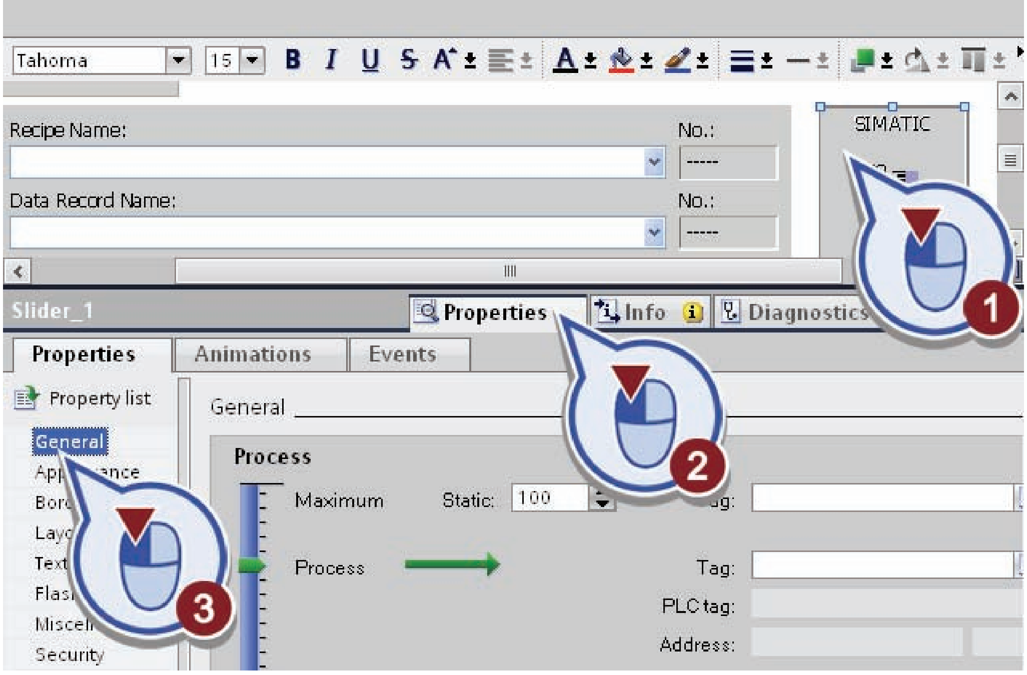
Procedure

To create the slider control, follow these steps:

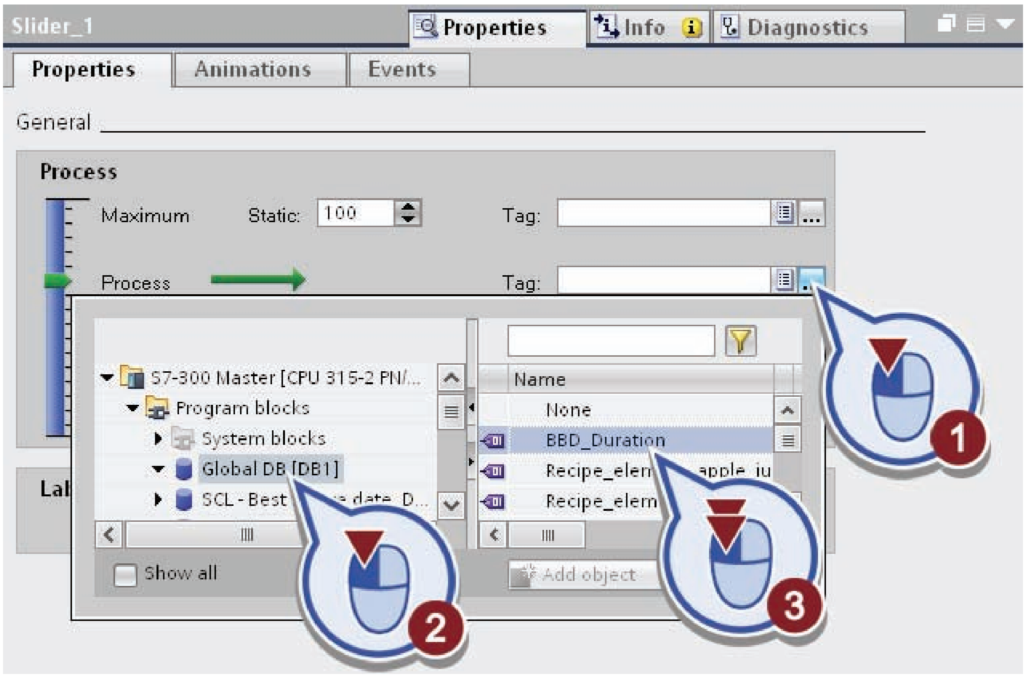
1. Create a slider control next to the recipe view from the "Elements" palette.



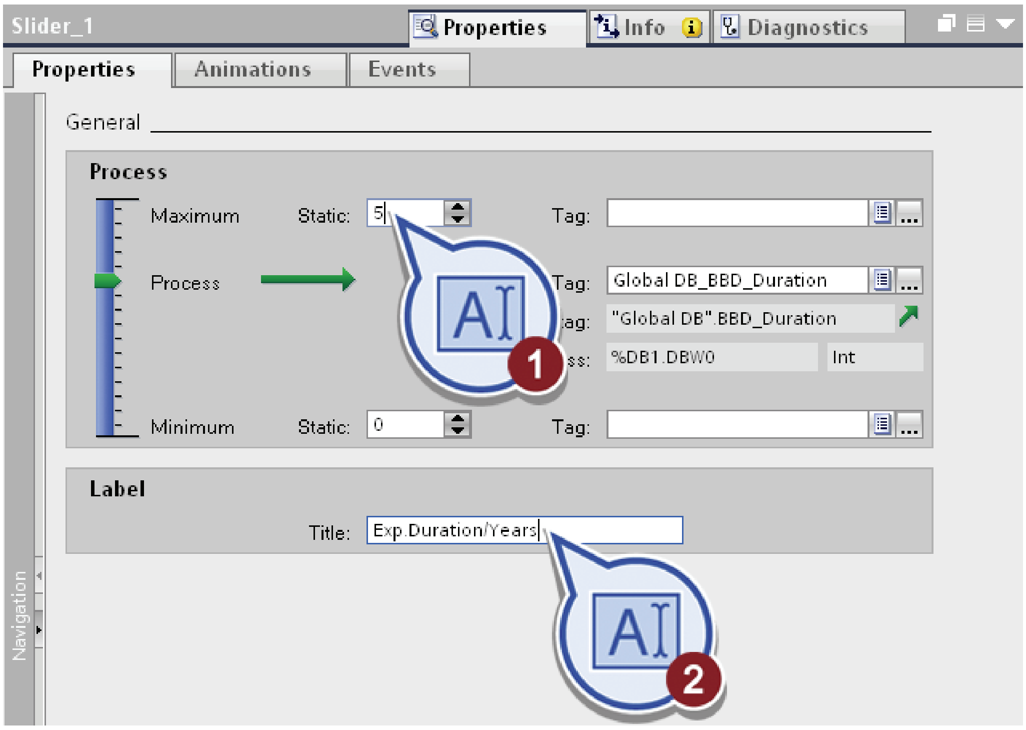
2. Select the slider control and open the "General" dialog in the Inspector window under the "Properties" tab.



3. Assign the slider control the "BBD_Duration" PLC tag of the global data block.



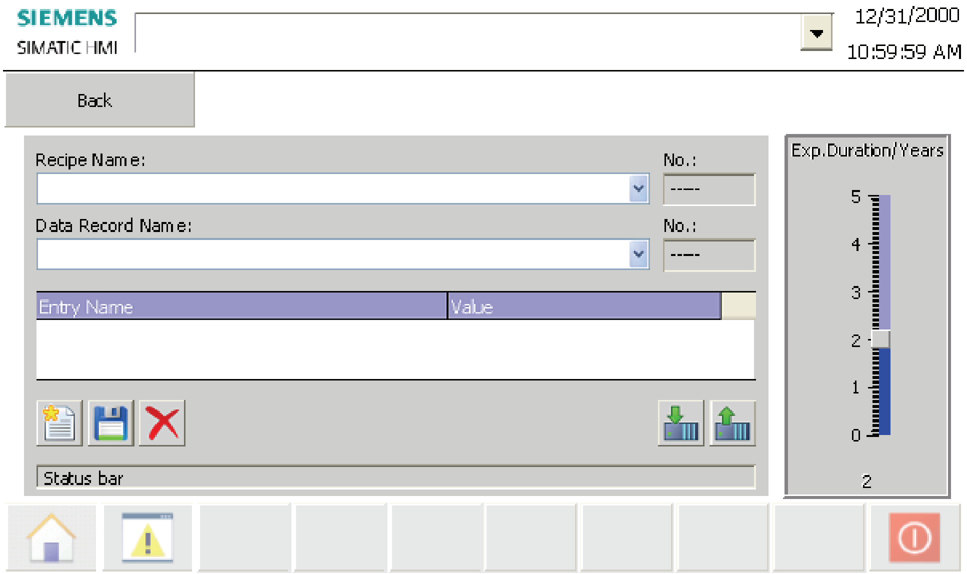
4. Assign the scale of the slider control the maximum value of "5" and change the label to "Exp. Duration/Years".



5. Save the project.

Result

You have created the slider control for setting the expiration date.



- You can set the expiration date to a value between 0 and 5 years.
- The current setting is displayed at the lower section of the slider control.
- If you change the value in Runtime, the newly set value is written to the data block and used at the input of the SCL function to calculate the expiration date.

5.4.7 Creating a navigation button

Introduction

In the following, you will create a button in the "Production" root screen, which will enable you to navigate in the "Recipes" HMI screen in Runtime.

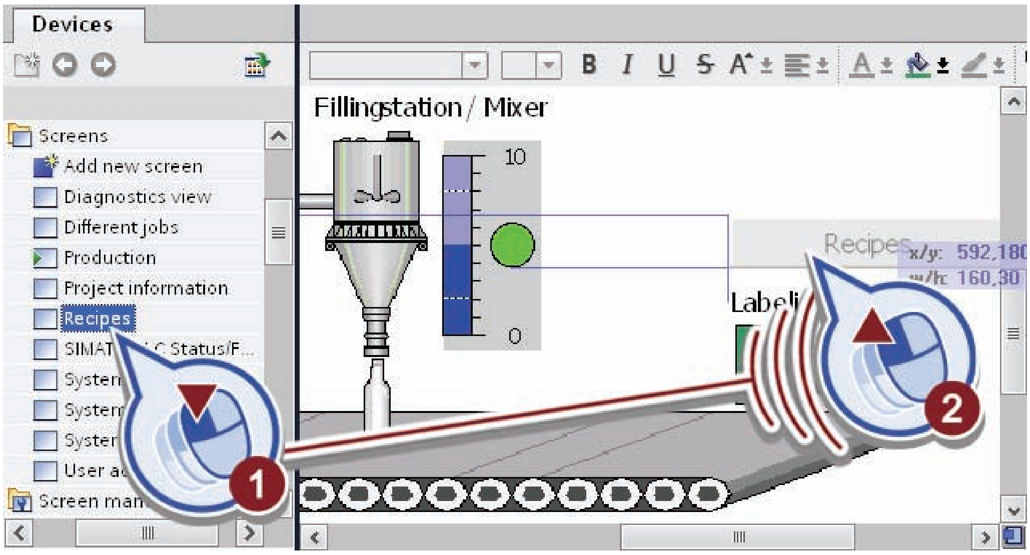
Requirement

You have created the "Production" root screen.

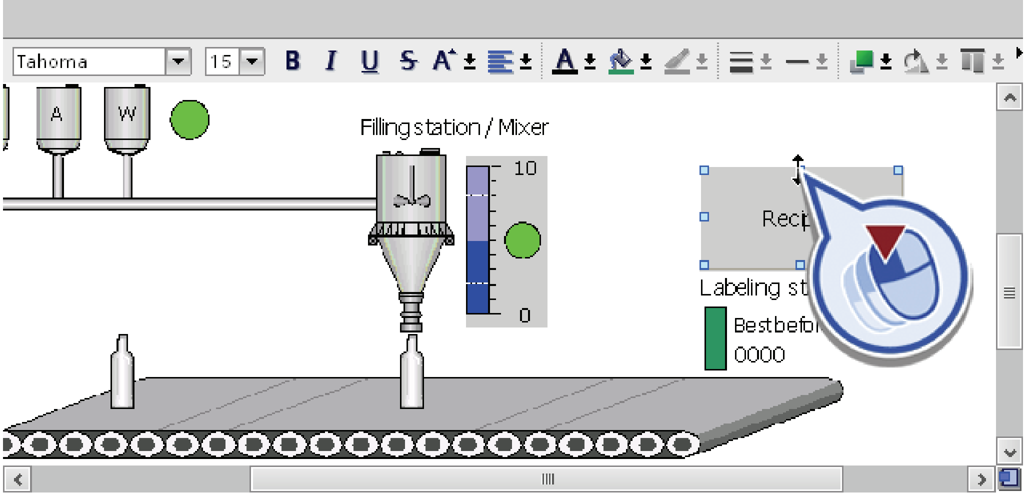
Procedure

To create the navigation button, follow these steps:

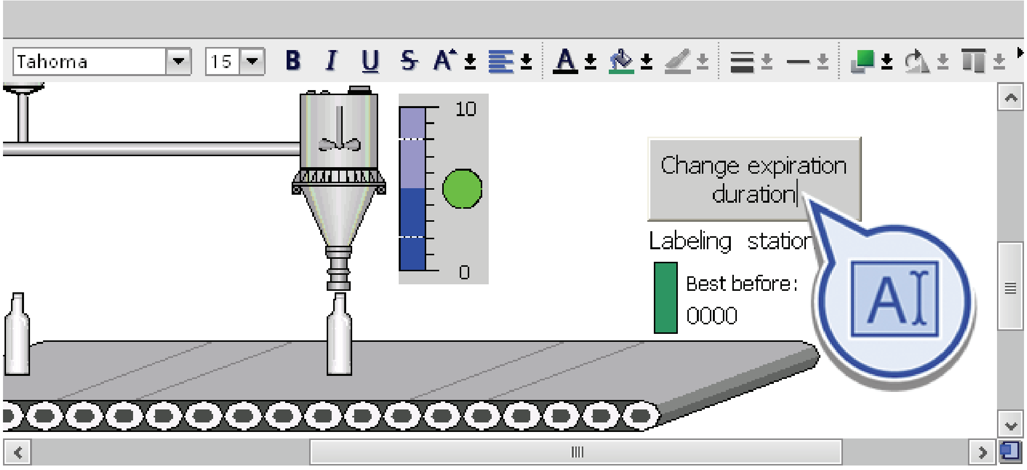
1. Open the "Production" HMI screen.
2. Click the "Recipes" HMI screen in the project tree and drag it onto an empty area above the labeling station.



3. Scale the button to about twice the height.



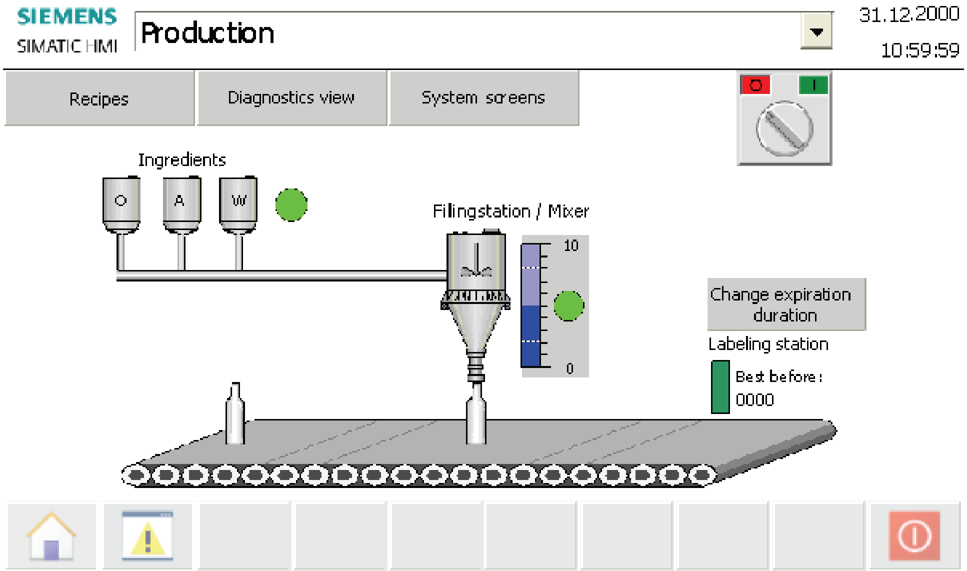
4. Change the label of the button to "Change expiration duration". Use the key combination <Shift> and <Enter> to insert a line break.



5. Save the project.

Result

You have successfully created the "Expiration duration" button in the "Production" root screen. An event is automatically created along with the button. This event executes the "ActivateScreen" function for the "Production" root screen when the button is clicked.



Configure alarms 6

6.1 Alarms in GRAPH

6.1.1 Create supervision

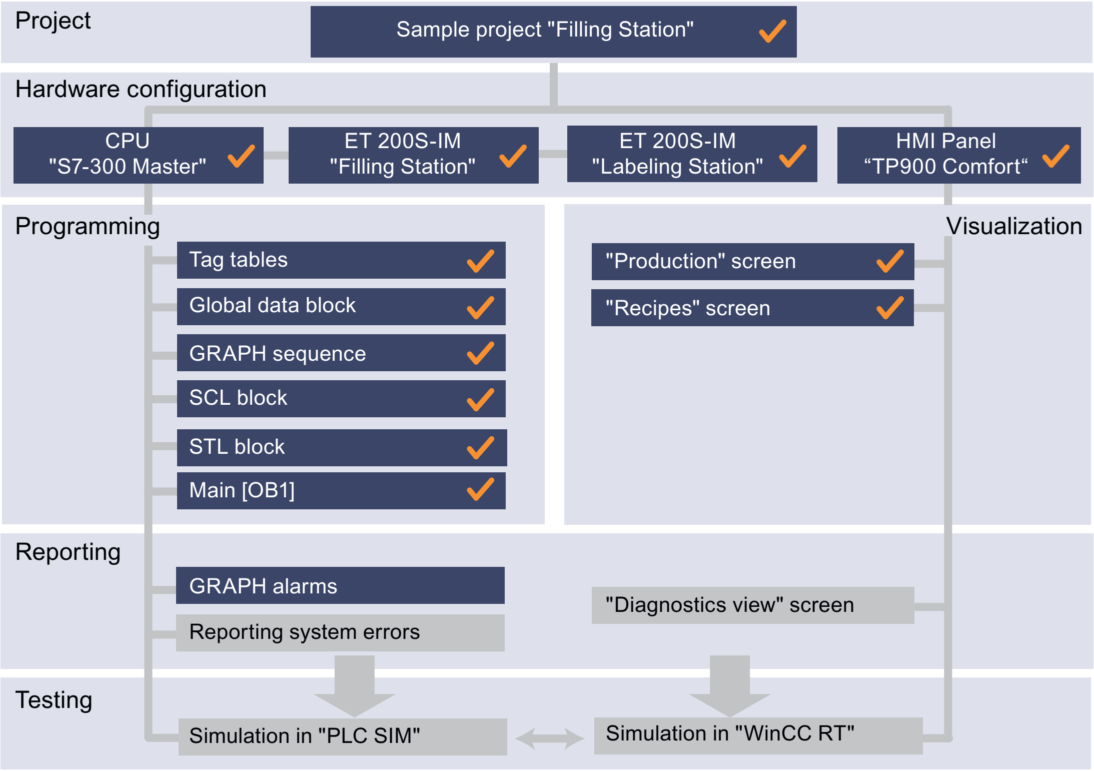
Introduction

In the following, you will create a sequence monitoring for the step "S5 Filling" of the sequence. In this case you should monitor how long the step executed. Because the filling process takes 3 seconds per bottle in this step, the sequence should stop and an error message should be generated as soon as this time is exceeded by half a second.

To monitor the step, you first create a supervision that you use to define when the execution time is exceeded.

Progress of project

The following graphic shows you which step you perform next:



Definition: Supervision

A supervision is a programmable monitoring condition within a step.

- If the condition is not met, this the good case.
- A met supervision leads to an error message.

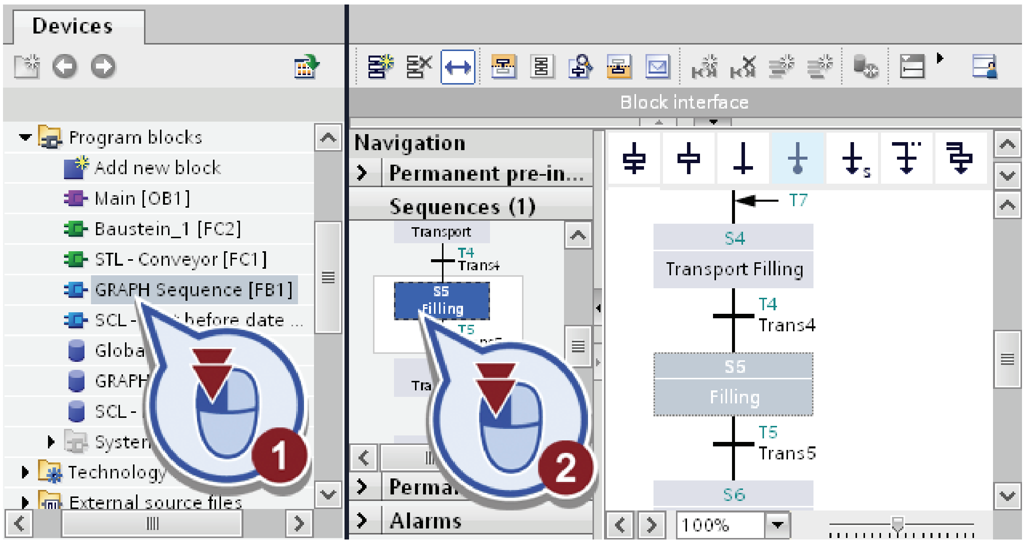
You can use the "Alarms" pane in the area navigation of the programming window to define the properties and contents of alarms. The next step is only enabled if the monitoring error is no longer pending and the following transition is satisfied.

Requirement

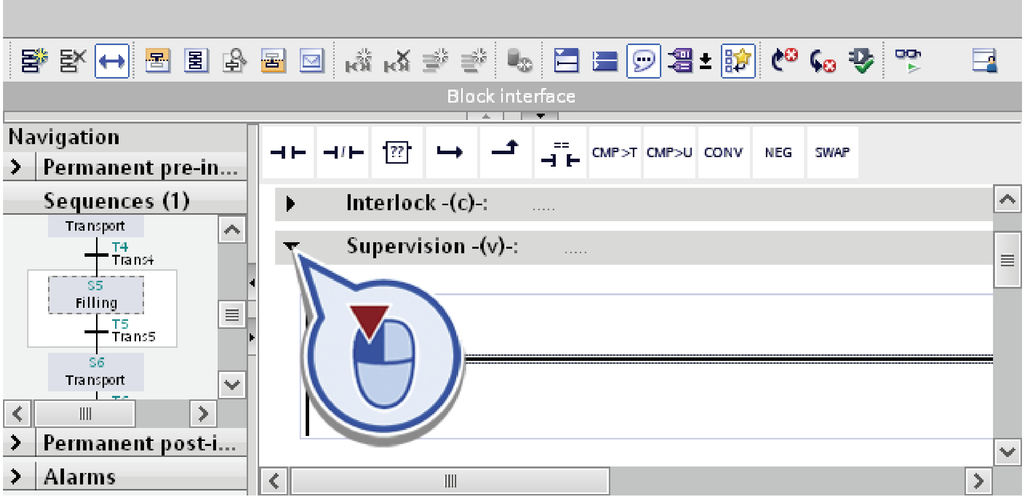
You have created the GRAPH sequence and programmed the step "S5 Filling".

Procedure

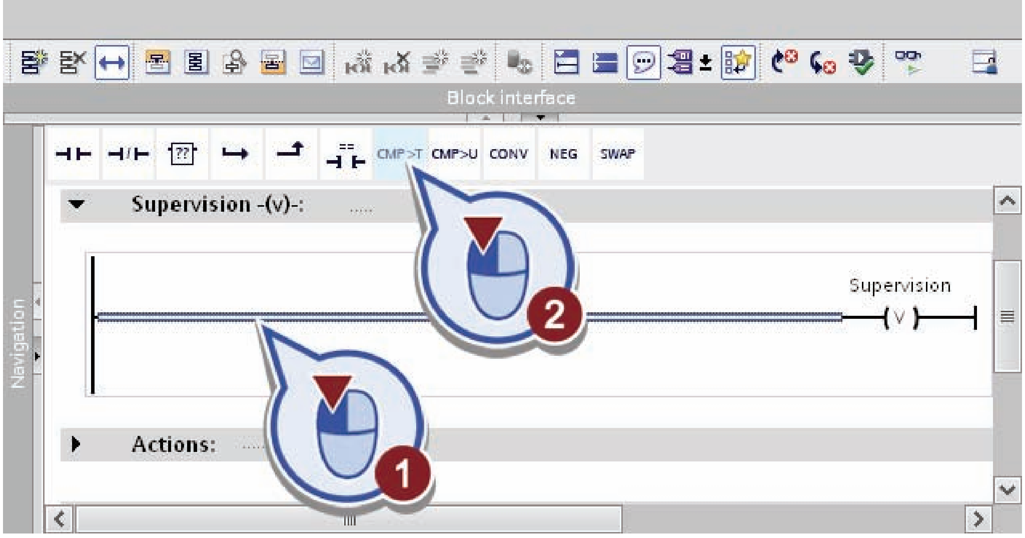
1. Open the step "S5 Filling" in the GRAPH FB.



2. Open the section "Supervision" in step "S5 Filling".



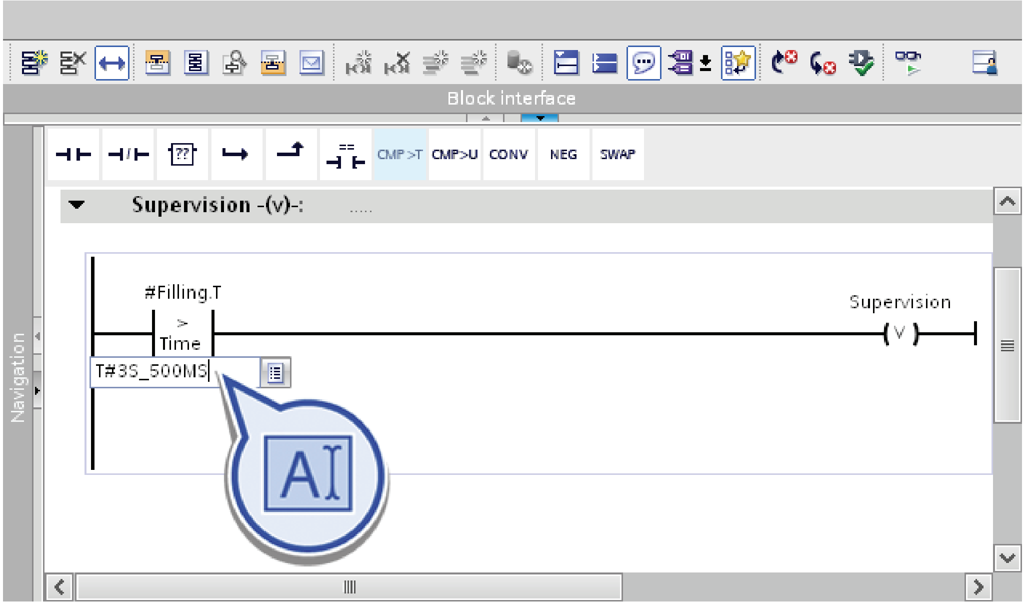
3. Insert the comparator "CMP >T" in the " Supervision".



The comparator is automatically assigned the tag "#Filling.T" as high value. The value of this tag of data type "Time" specifies how long the step has already been executed. The value is reset with every call of the step.

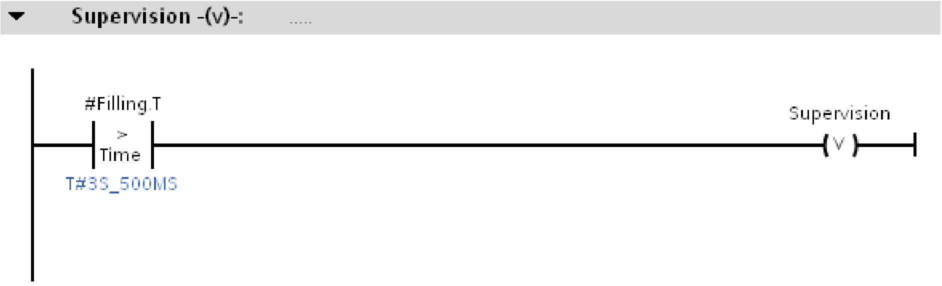
The comparator is automatically assigned the time "T#100MS" (0.1 seconds) in the format "Time" as the low value. This means the supervision condition is met as soon as the step takes longer than 0.1 seconds.

4. Increase the lower value of the comparator to "T#3S_500MS" (3.5 seconds).



Result

You have successfully created a supervision for the step "S5 Filling".



If the transition condition of the previous step has been met and the step "S5 Filling" is activated, the timer of the tag "T" of the step is started automatically:

- If the step is completed within 3.5 seconds, the supervision condition has not been met and the sequence is continued without interruption.
- If the step is not completed within 3.5 seconds, the supervision condition has been met. The status bit of the "V1" tag (incoming monitoring error) is set to "1" for the step in the instance data block of the sequence. The execution of the sequence is stopped.

In the next step, you will create an error message for the event that the condition for "Supervision" has been met.

6.1.2 Create alarm for sequence monitoring

Introduction

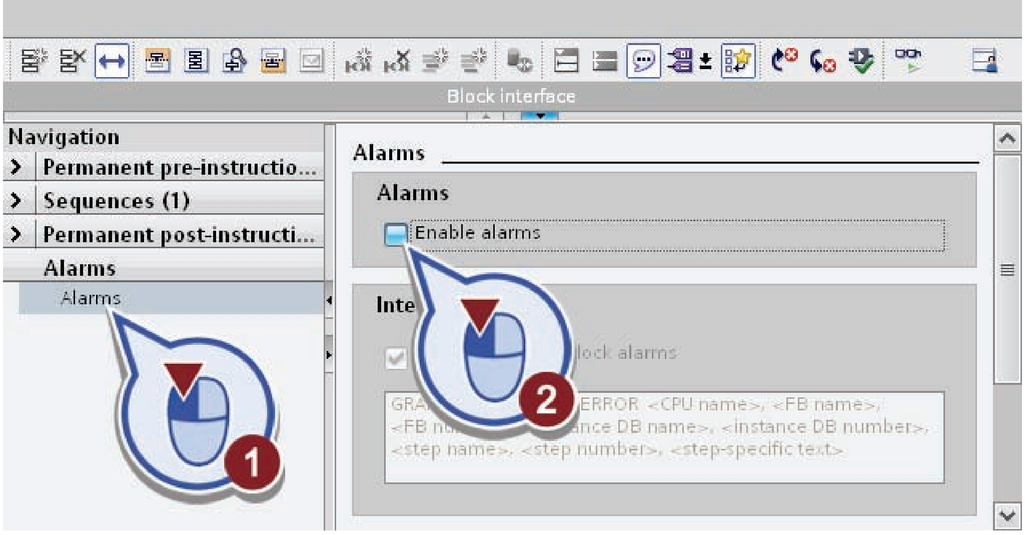
You are going to activate the alarm generation for supervisions and create an alarm text that is displayed on the HMI Panel if the previously programmed condition of the supervision has been met.

Requirement

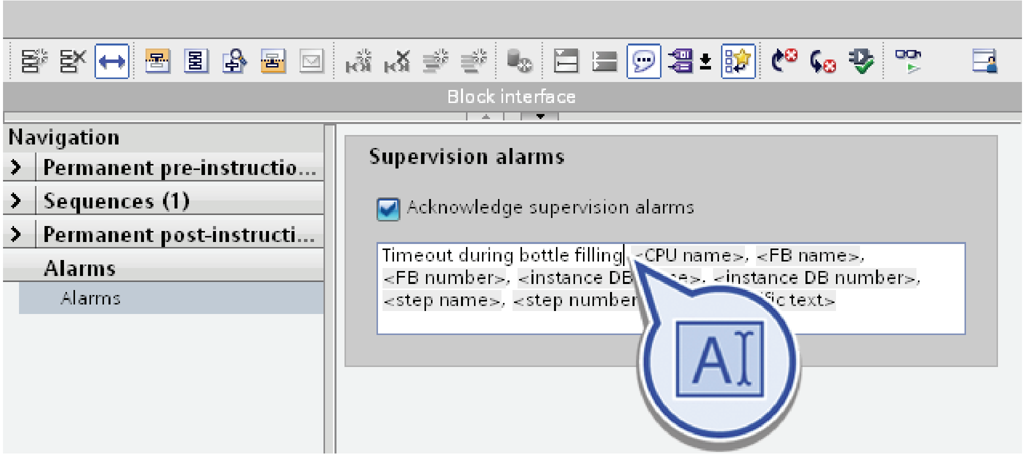
You have created the supervision for the step "S5 Filling".

Procedure

1. Open the "Alarms" pane in the navigation of the step and enable the alarms for the step.


Block interface
Navigation
Permanent pre-instructio...
Sequences (1)
Permanent post-instructi...
Alarms
Alarms
Alarms
Alarms
Enable alarms
Inte
lock alarms
GRA ERROR <CPU name>, <FB name>,
<FB n ance DB name>, <instance DB number>,
<step name>, <step number>, <step-specific text>
1
2

2. In the field "Supervision alarms" replace the alarm text "GRAPH7_SUPERVISION_FAULT" with "Timeout during bottle filling".



Supervision alarms

☑ Acknowledge supervision alarms

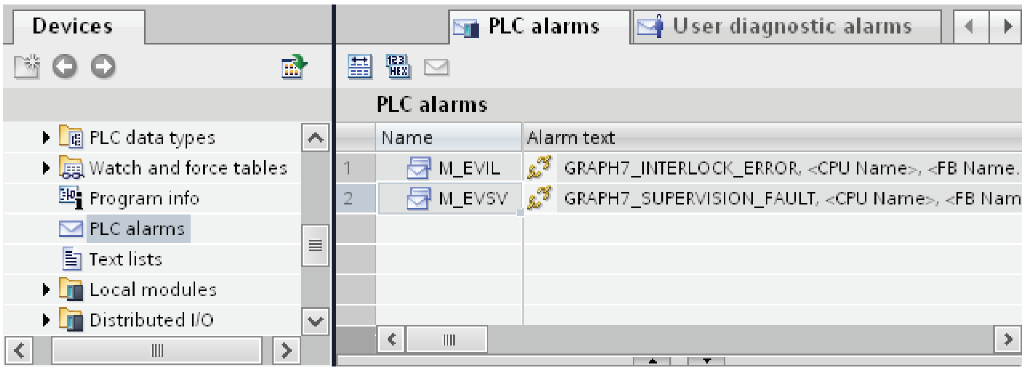
Timeout during bottle filling, <CPU name>, <FB name>, <FB number>, <instance DB e>, <instance DB number>, <step name>, <step number fic text>

3. Compile the complete programming of the CPU by clicking the "S7-300 Master" CPU with the right mouse button in the project tree on and selecting "Compile" > "All" from the shortcut menu.

Result

While compiling the blocks, the alarms and alarm classes for the newly created GRAPH alarm are created automatically:

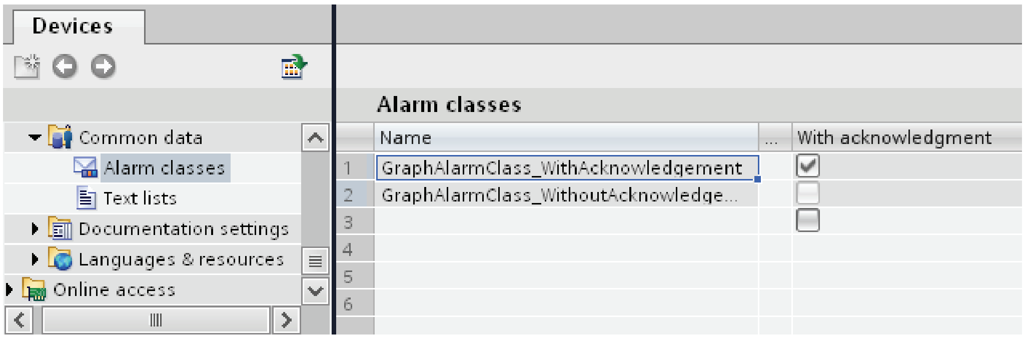
- Two new alarms with their own alarm classes have been created in the project tree under "S7-300 Master" > "PLC alarms". The PLC alarm "M_EVSV" contains the alarm of the Supervision.



PLC alarms

	Name	Alarm text
1	M_EVIL	GRAPH7_INTERLOCK_ERROR, <CPU Name>, <FB Name.
2	M_EVSV	GRAPH7_SUPERVISION_FAULT, <CPU Name>, <FB Nam

- Alarm classes are created for the enabled alarms in the project tree under "Common data" > "Alarm classes". The setting "With acknowledgment" is set automatically. If the alarm is output on the HMI Panel, the alarm text will be displayed until the alarm has been acknowledged.



Alarm classes

	Name	...	With acknowledgment
1	GraphAlarmClass_WithAcknowledgement		☑
2	GraphAlarmClass_WithoutAcknowledge...		□
3			□
4			
5			
6			

6.2 Reporting system errors

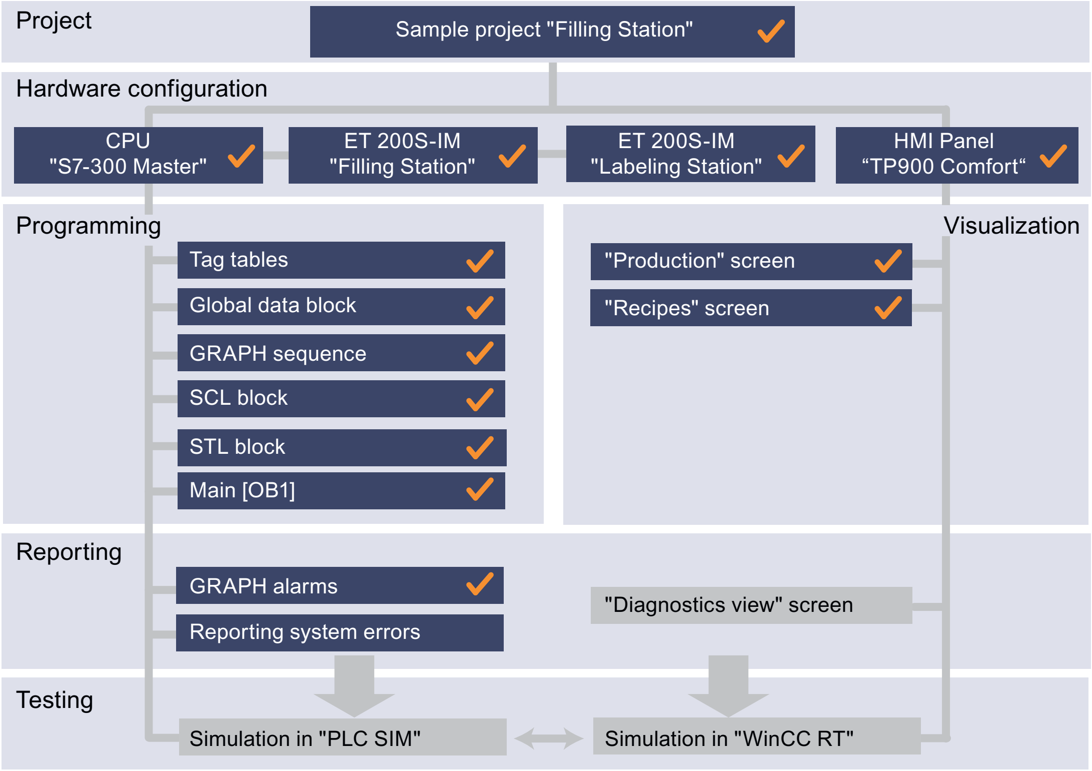
6.2.1 System diagnostics with "Report System Errors"

Introduction

In the following section you create blocks with the system diagnostics "Report system errors" that analyzes errors in the system and create alarms with an error description and the error location. You then create a System diagnostics view to output the error messages on the HMI panel.

Progress of project

The following graphic shows you which step you perform next:



Functionality of the system diagnostics "Report System Errors"

Hardware components can trigger calls of organization blocks in case of a system error and provide information on the system error that has occurred. The system diagnostics with "Report system errors" (RSE) offers the user-friendly option to evaluate this diagnostic information and display it in the form of alarms. The required blocks and alarm texts are created in the properties of the PLC. You must only download the created blocks to the CPU.

To display diagnostic events on an HMI device, you can generate one or more status DBs. These status DBs are updated by the system diagnostics blocks and then contain information on the current state of the system.

Note

If you use system diagnostics, the system response of the plant may change if an error occurs. For example, the CPU may not change to "STOP" mode as it would without system diagnostics.

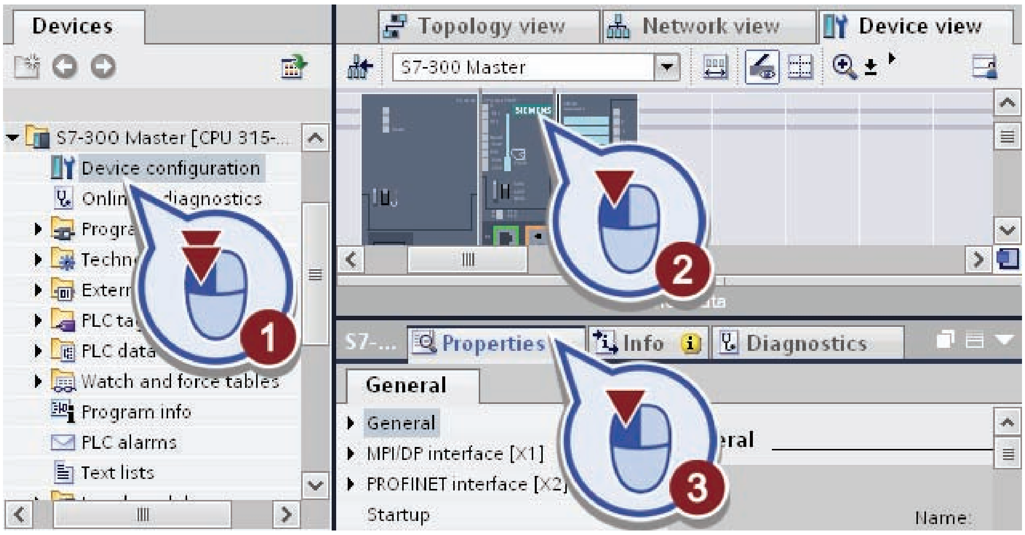
6.2.2 Activating system diagnostics of the CPU

Introduction

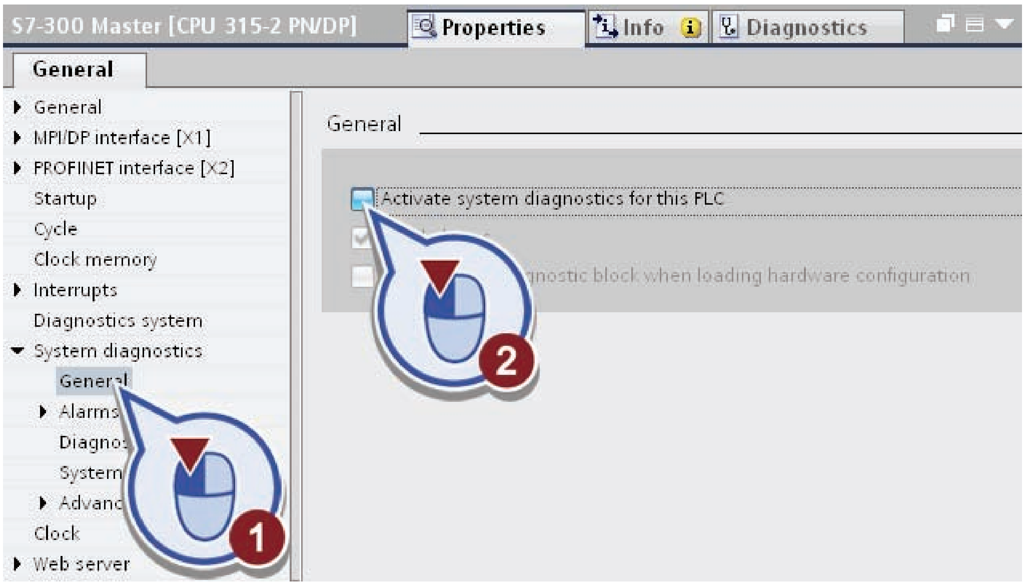
You will now activate the system diagnostics for the "S7-300 Master" CPU. If the system diagnostics is activated, all alarms and blocks required for diagnostics are automatically generated during the next compilation of the hardware.

Procedure

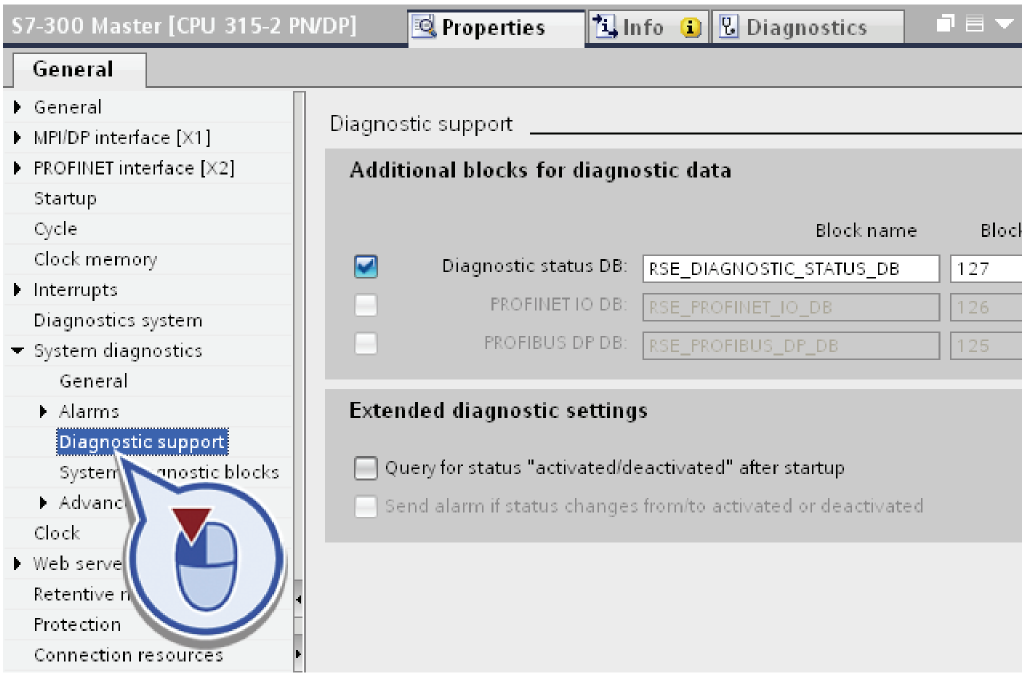
1. Select the CPU in the device configuration and open the "Properties" tab in the Inspector window.



2. Activate the system diagnostics for the CPU under "System diagnostics" > "General".



3. Check whether the functions Send alarms" and "Load system diagnostic block when loading hardware configuration" were enabled during activation of the system diagnostics (default setting). If necessary, enable both functions.
4. Open the diagnostic support.



Diagnostic support

Additional blocks for diagnostic data

		Block name	Block
☑	Diagnostic status DB:	RSE_DIAGNOSTIC_STATUS_DB	127
□	PROFINET IO DB:	RSE_PROFINET_IO_DB	126
□	PROFIBUS DP DB:	RSE_PROFIBUS_DP_DB	125

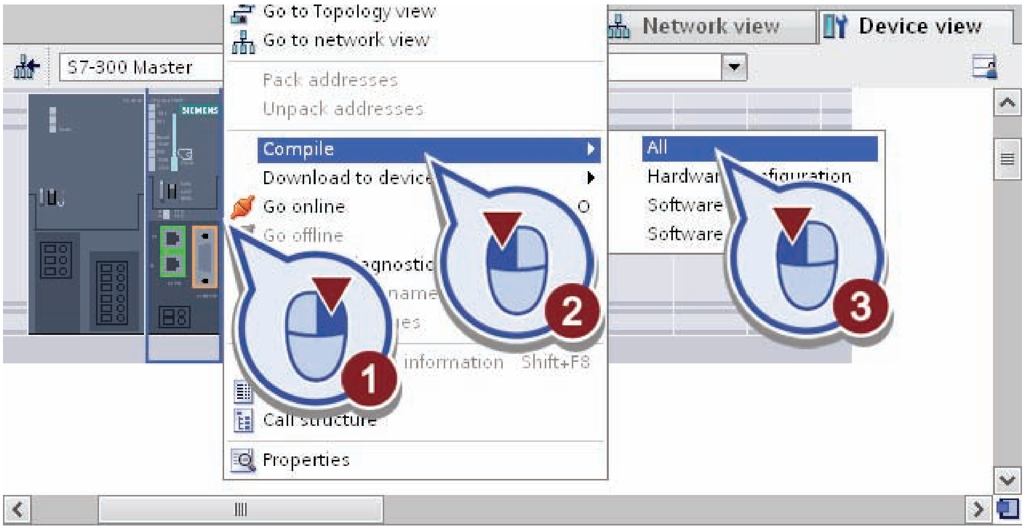
Extended diagnostic settings

□ Query for status "activated/deactivated" after startup

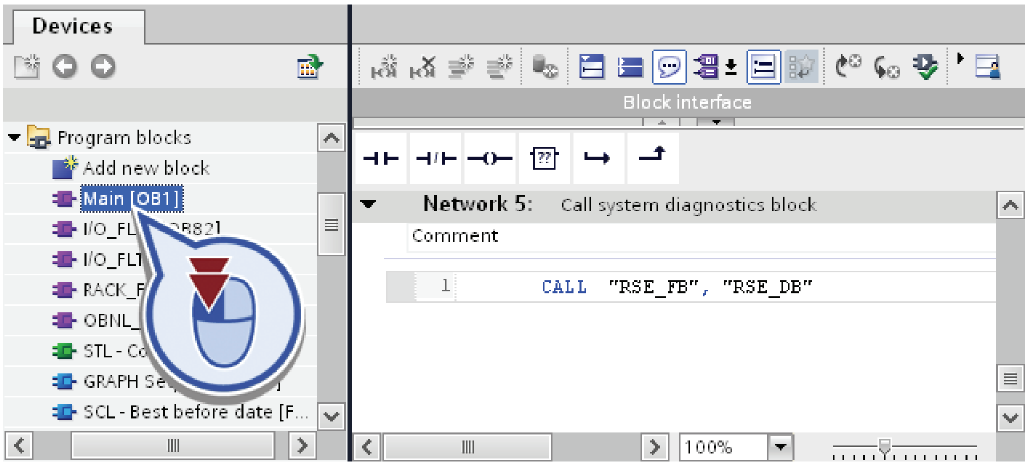
□ Send alarm if status changes from/to activated or deactivated

With the activation of the system diagnostics, the data block "RSE_DIAGNOSTIC_STATUS_DB" for acquisition of diagnostic data has been activated automatically.

5. To generate the required system diagnostics blocks, switch to the work area of the device view and compile the hardware configuration of the CPU.



6. In the project tree, open the organization block "Main".



```
CALL  "RSE_FB", "RSE_DB"
```

Result

In the final network of the organization block, a call of the system diagnostics block has been added automatically during compilation of the hardware configuration.

In addition to the entry in the organization block "Main", additional organization blocks were generated for different error cases. If an error occurs within the module, the corresponding organization block is called automatically (for example, OB 83 in the event of a remove/insert alarm). A call of the "RSE_FB" system diagnostics block has been added in each organization block which reads out the error information.

The error information is stored in the diagnostic status DB "RSE_DIAGNOSTIC_STATUS_DB".

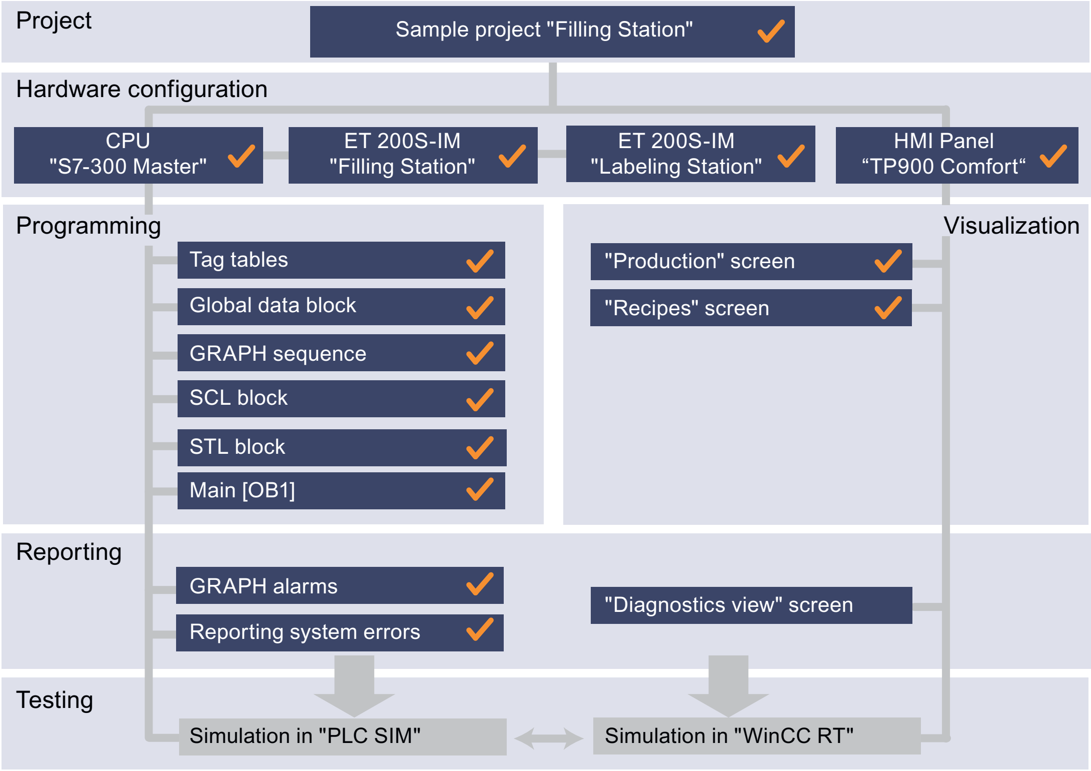
6.2.3 Creating a diagnostics view in HMI

Introduction

You will now create a "System diagnostics view" in the HMI screen "Diagnostics view". You can use it to output the diagnostic data of the devices configured in the "Devices & Networks" editor.

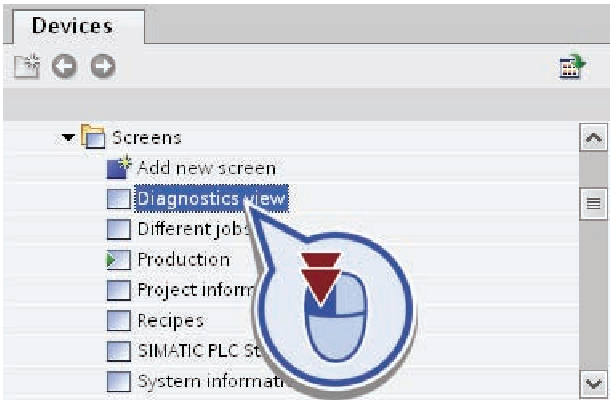
Progress of project

The following graphic shows you which step you perform next:

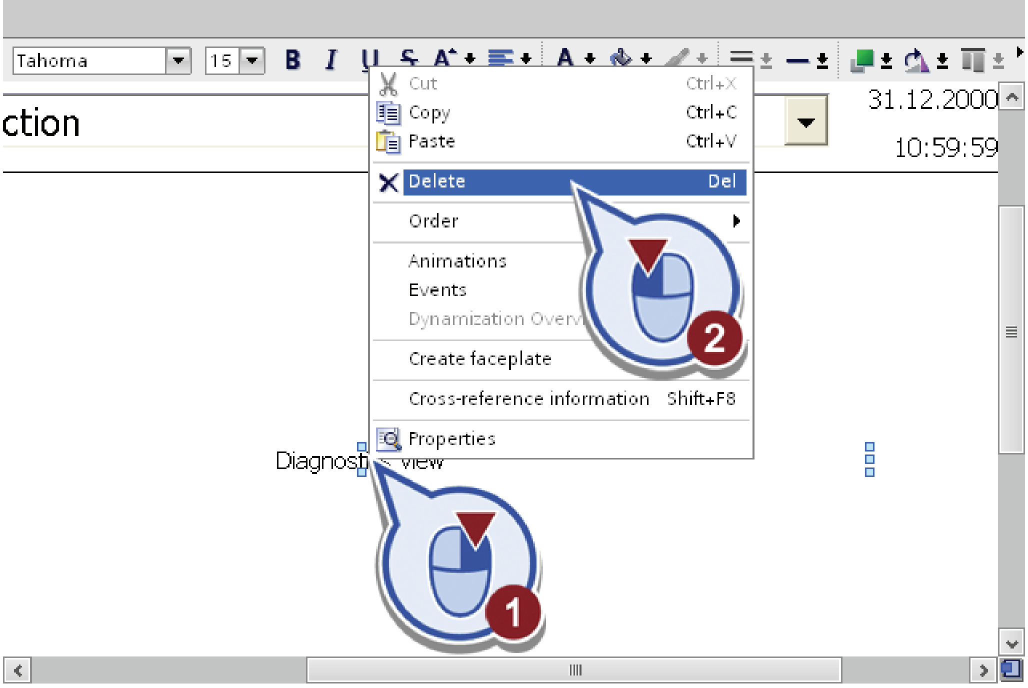


Procedure

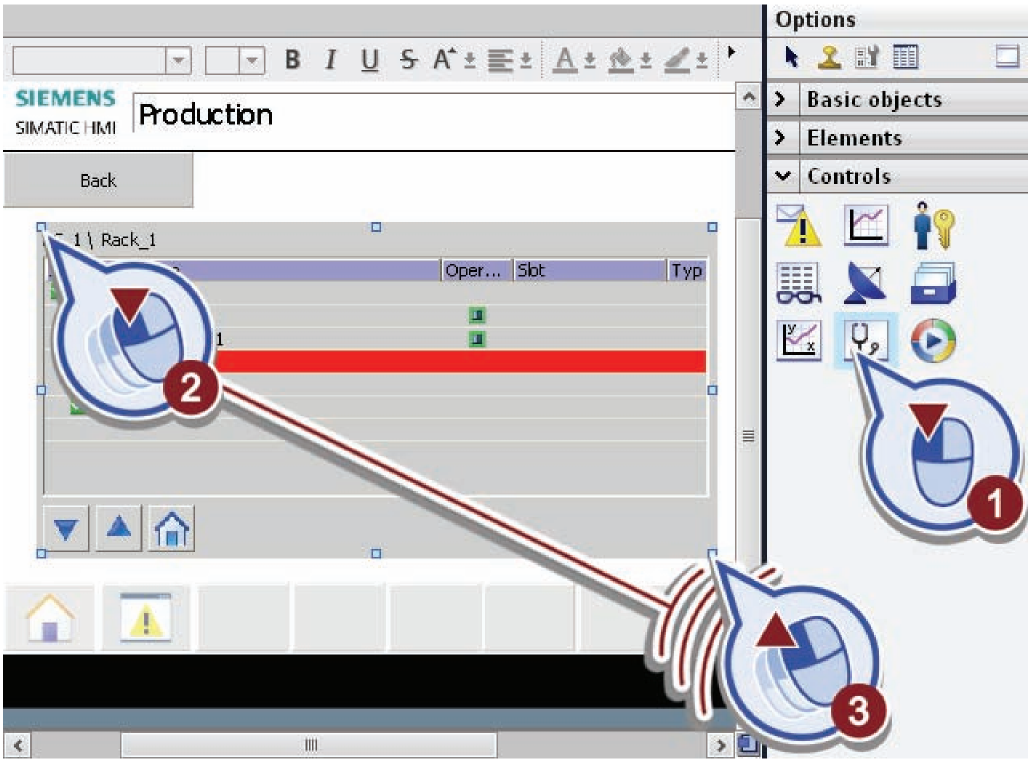
1. In the project tree, open the HMI screen "Diagnostics view".



2. Delete the automatically generated text field in the HMI screen.



3. Select the "System diagnostics view" from the "Controls" palette in the "Toolbox" task card and drag open the diagnostics view in the HMI screen.



Result

You have successfully created the system diagnostics view. After the start of Runtime, you can use the "Diagnostics view" button in the "Production" root screen to open the system diagnostics view and check the diagnostic status of the devices used.

Testing the sample project online 7

7.1 Test program

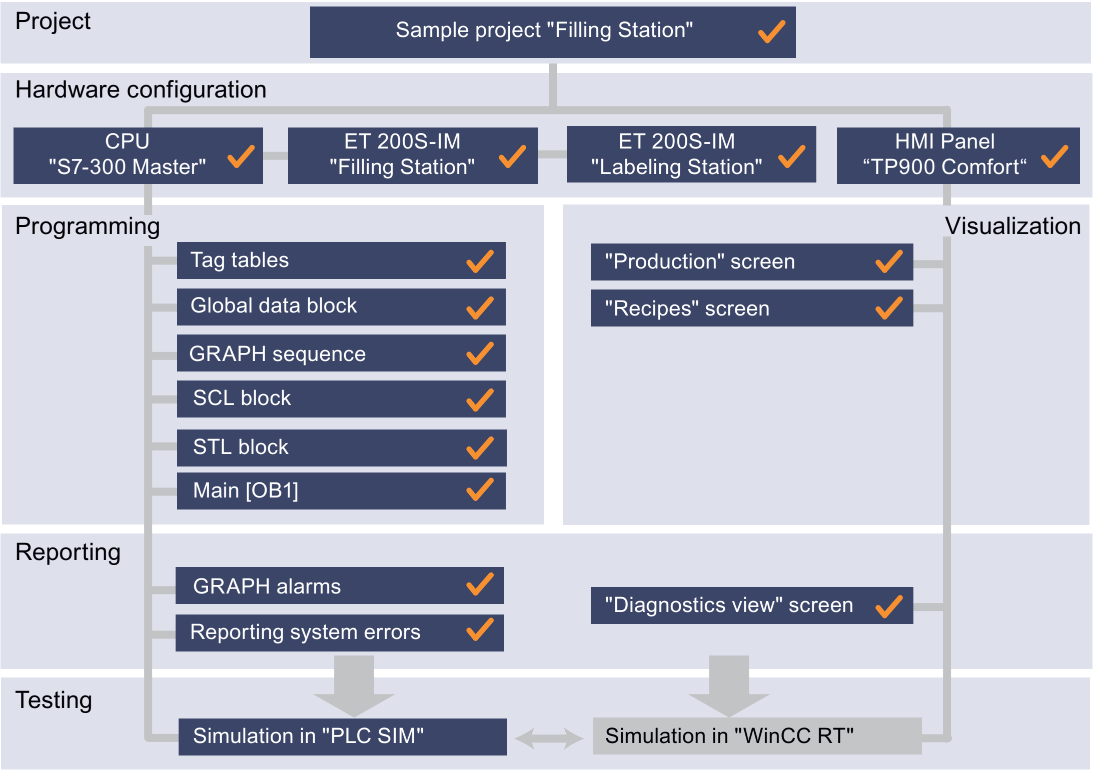
7.1.1 Start simulation in PLCSIM

Introduction

You will now test the functionality of the program with the PLCSIM simulation software. You can use this simulation to test the program for errors before you start the production. First download the configuration and the user program into the simulated module and insert view objects for monitoring and control of outputs and bit memories.

Progress of project

The following graphic shows you which configuration step you carry out next:



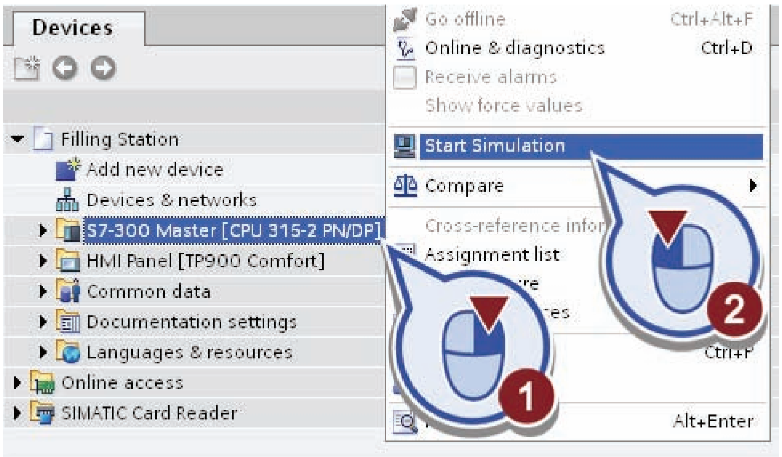
Requirement

You have created the hardware configuration and the user program.

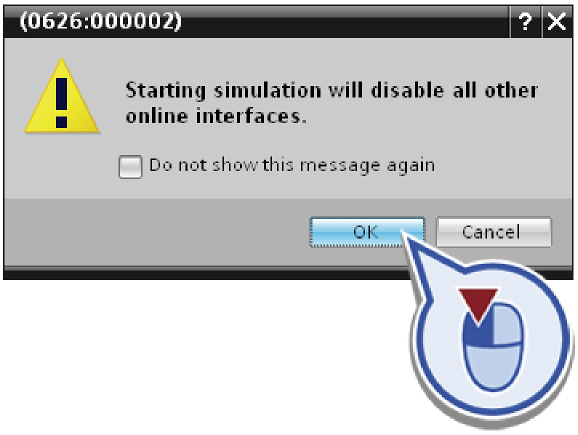
Procedure

To start the PLCSIM software, follow these steps:

1. Right-click the S7-300 Master CPU in the project tree. From the shortcut menu select the menu item "Start Simulation".



2. Confirm the following dialog with "OK".


(0626:000002)

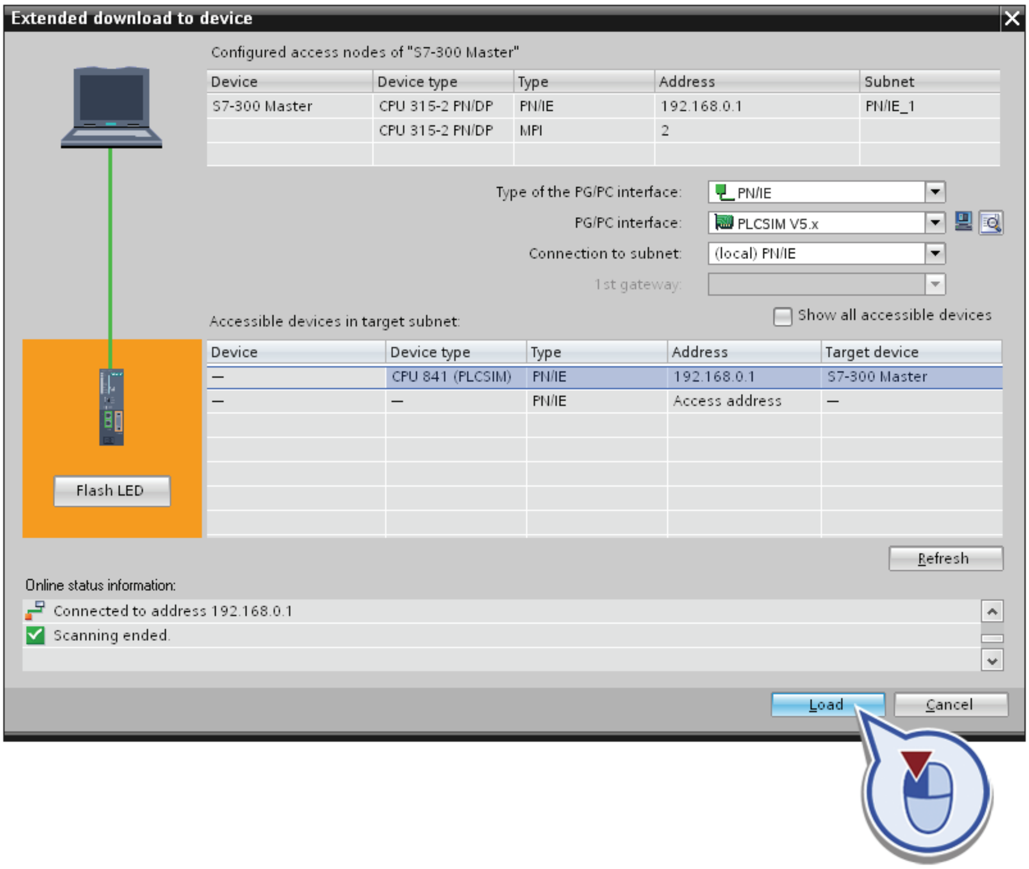
Starting simulation will disable all other online interfaces.

□ Do not show this message again

The PLCSIM software starts in the background and the dialog "Extended download to device" opens.

3. In the "Extended download to device" dialog window, select the following settings:
   - Type of PG/PC interface: PN/IE
   - PG/PC interface: PLCSIM V5.x
   - Connection to subnet: PN/IE_1

   Then click the "Load" button.



Extended download to device

Configured access nodes of "S7-300 Master"

Device	Device type	Type	Address	Subnet
S7-300 Master	CPU 315-2 PN/DP	PN/IE	192.168.0.1	PN/IE_1
	CPU 315-2 PN/DP	MPI	2	

Type of the PG/PC interface: PN/IE
PG/PC interface: PLCSIM V5.x
Connection to subnet: (local) PN/IE
1st gateway:

Accessible devices in target subnet:

☐ Show all accessible devices

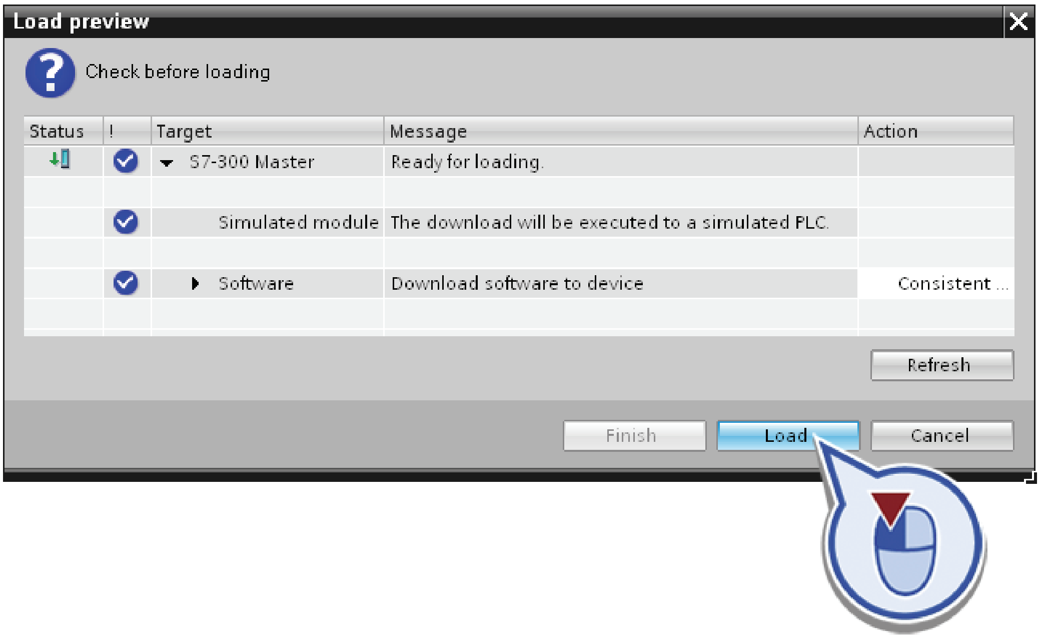
Device	Device type	Type	Address	Target device
–	CPU 841 (PLCSIM)	PN/IE	192.168.0.1	S7-300 Master
–	–	PN/IE	Access address	–

Online status information:

Connected to address 192.168.0.1
✓ Scanning ended.

The "Compile" dialog is processed.

4. Confirm the load procedure in the module simulated with PLCSIM.

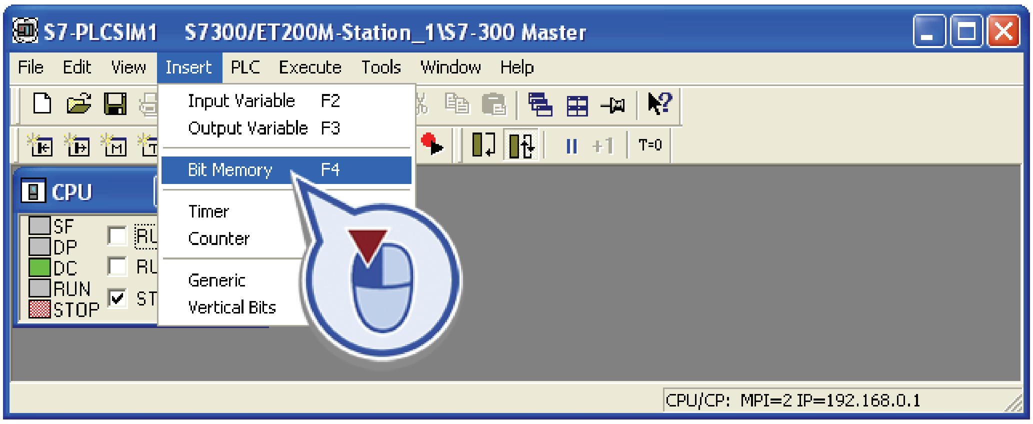


Load preview

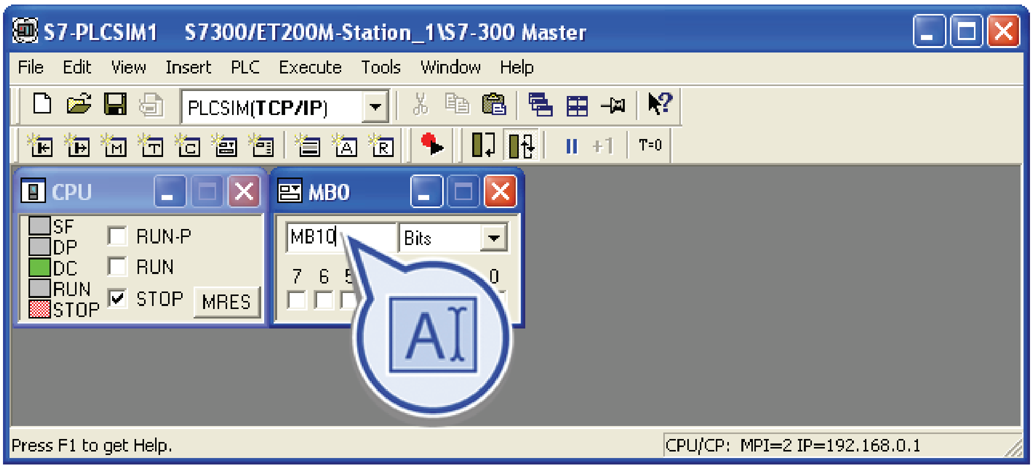
Check before loading

Status	!	Target	Message	Action
	✓	S7-300 Master	Ready for loading.	
	✓	Simulated module	The download will be executed to a simulated PLC.	
	✓	Software	Download software to device	Consistent ...

5. Insert a "Bit Memory" view object to control and output the PLC tags with memory bits in the address area M10.0 to M10.7.

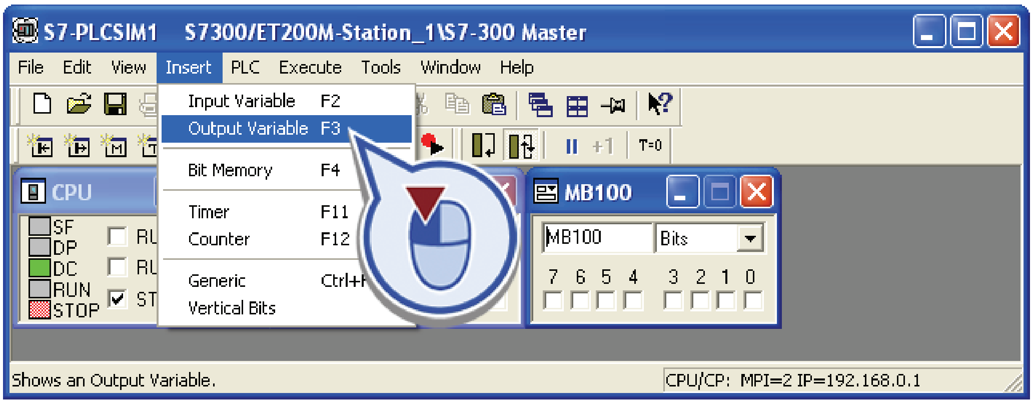


6. Enter the memory byte MB10 as address area. Leave the format set to "Bits".


S7-PLCSIM1 S7300/ET200M-Station_1\S7-300 Master
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(TCP/IP)
CPU
SF DP DC RUN STOP
RUN-P RUN STOP MRES
MB0
MB10
Bits
7 6 5 0
Press F1 to get Help.
CPU/CP: MPI=2 IP=192.168.0.1

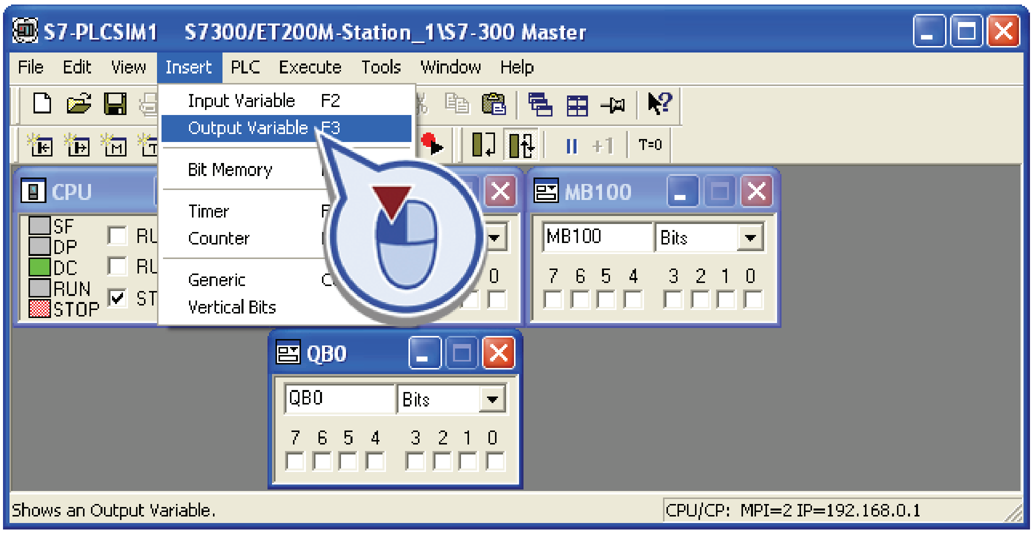
7. Create an additional "Bit Memory" view object as shown in the last two steps to control and output the memory bits in the address area M100.0 to M100.7. Enter the memory byte MB100 as address area. Leave the format set to "Bits".

8. Insert an "Output Variable" view object to output the outputs in the address area Q0.0 to Q0.7.


S7-PLCSIM1 S7300/ET200M-Station_1\S7-300 Master
File Edit View Insert PLC Execute Tools Window Help
Input Variable F2
Output Variable F3
Bit Memory F4
Timer F11
Counter F12
Generic Ctrl+F
Vertical Bits
CPU
SF DP DC RUN STOP
MB100
MB100
Bits
7 6 5 4 3 2 1 0
Shows an Output Variable.
CPU/CP: MPI=2 IP=192.168.0.1

Leave the address area set to QB0. This area is used by default when you create a new view object "Output Variable".

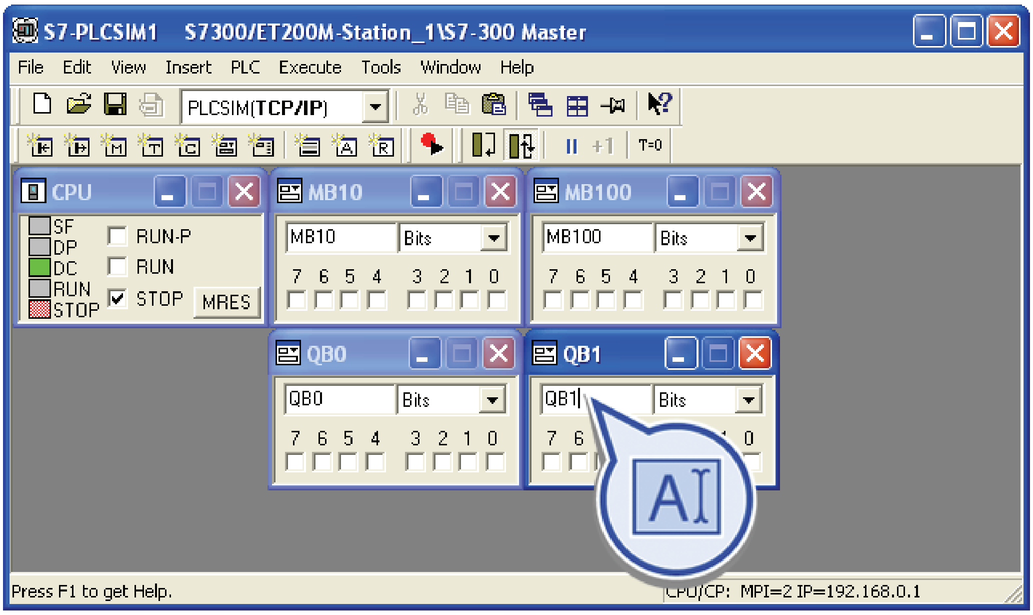
9. Create another "Output Variable" view object to output the outputs in the address area Q1.0 to Q1.7.



Shows an Output Variable.

CPU/CP: MPI=2 IP=192.168.0.1

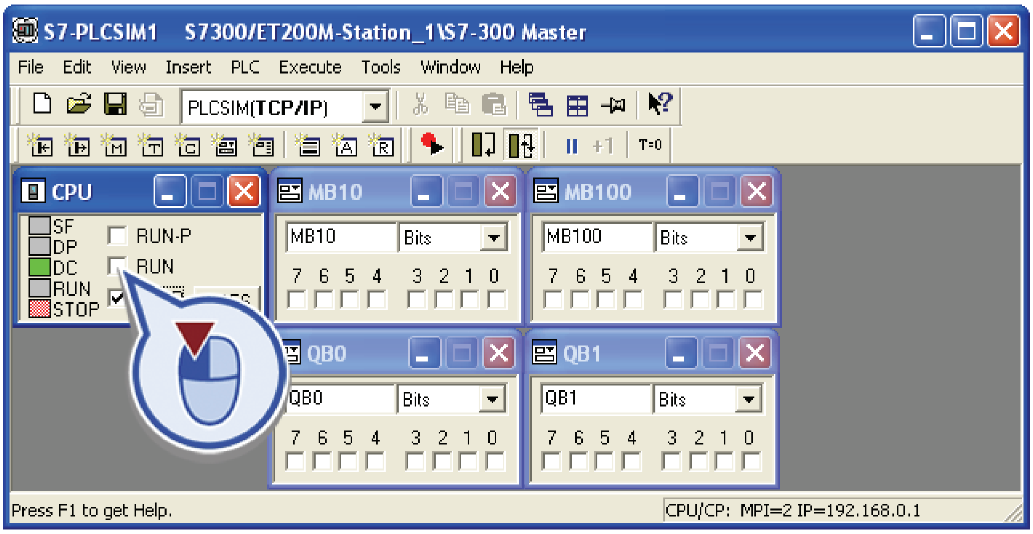
10. In the second "Output Variable view object, set the address area to the output byte QB1. Leave the format set to "Bits".


S7-PLCSIM1 S7300/ET200M-Station_1\S7-300 Master
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(TCP/IP)
CPU
SF
DP
DC
RUN
STOP
☐ RUN-P
☐ RUN
☑ STOP
MRES
MB10
MB100
QB0
QB1
Bits
7 6 5 4 3 2 1 0
T=0

Press F1 to get Help.

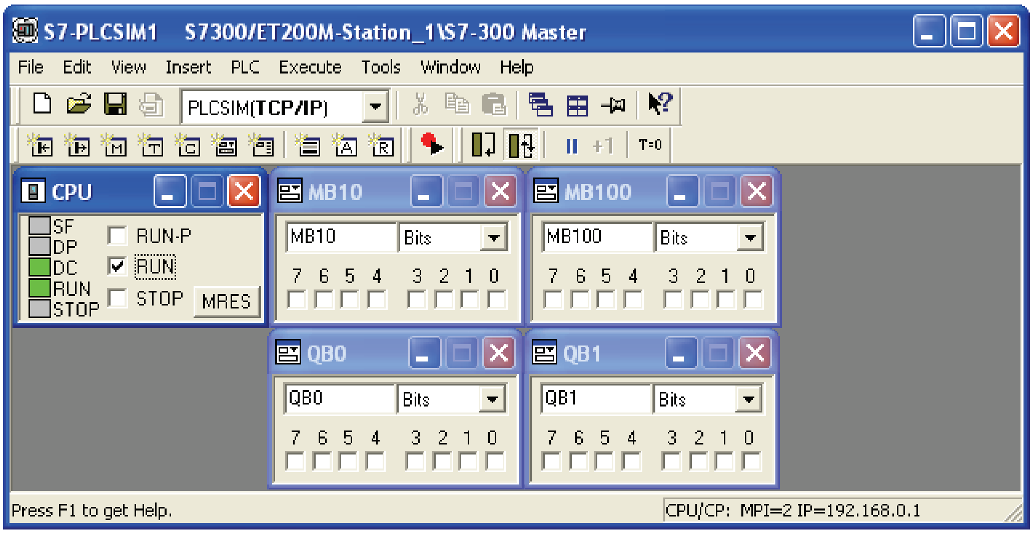
CPU/CP: MPI=2 IP=192.168.0.1

11. Switch the simulation of the CPU to RUN mode.


S7-PLCSIM1 S7300/ET200M-Station_1\S7-300 Master
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(TCP/IP)
CPU
SF
DP
DC
RUN
STOP
RUN-P
RUN
MB10
Bits
7 6 5 4 3 2 1 0
MB100
QB0
QB1
Press F1 to get Help.
CPU/CP: MPI=2 IP=192.168.0.1

Result

You have successfully set up the simulation of the CPU in PLCSIM. After the transition to RUN, the two LEDs "RUN" (operating mode) and "DC" (power supply) are lit in green.


S7-PLCSIM1 S7300/ET200M-Station_1\S7-300 Master
File Edit View Insert PLC Execute Tools Window Help
PLCSIM(TCP/IP)
CPU
SF
DP
DC
RUN
STOP
□ RUN-P
☑ RUN
□ STOP
MRES
MB10
Bits
7 6 5 4 3 2 1 0
MB100
QB0
QB1
Press F1 to get Help.
CPU/CP: MPI=2 IP=192.168.0.1

Leave the PLCSIM window open. You will now use the simulation to test the sample project "Filling Station" with the online and diagnostics functions.

7.1.2 Testing execution of the GRAPH sequence

Introduction

You will now test the call of the GRAPH sequence with PLCSIM. For this purpose, you will first establish an online connection with the simulated module. In the online view, start the call of the GRAPH block and check the sequence.

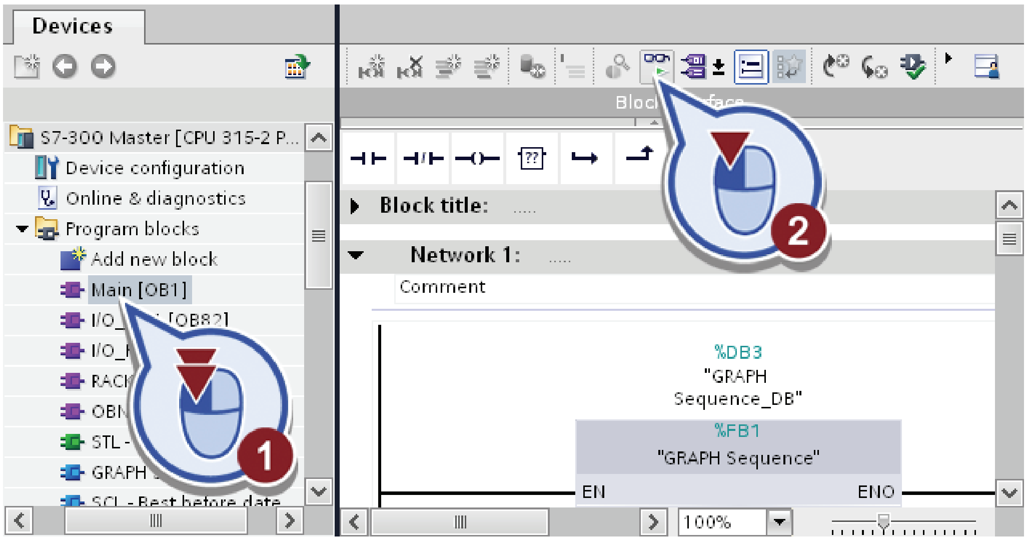
Requirement

You have downloaded the hardware configuration and the program blocks to PLCSIM.

Procedure

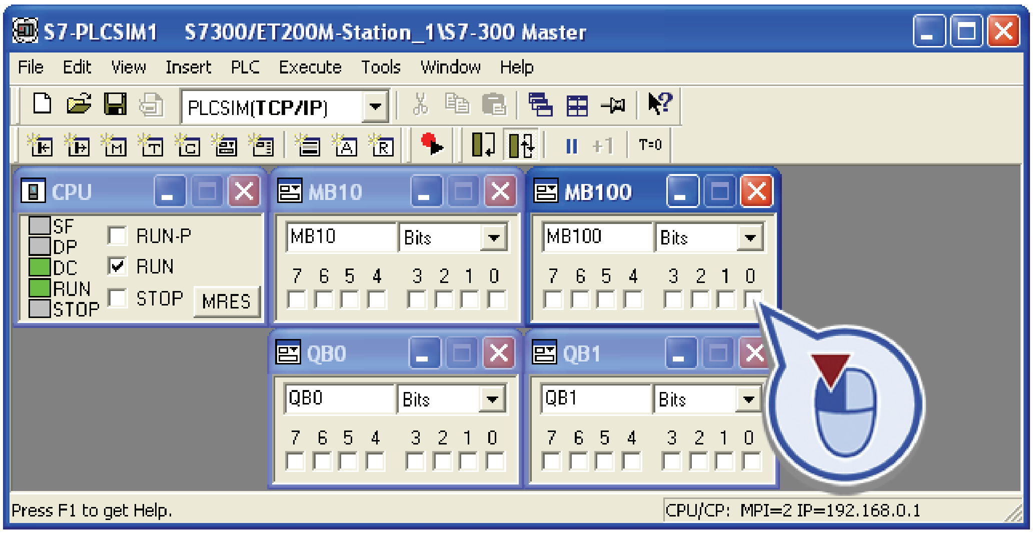
To test the GRAPH sequence, follow these steps:

1. Open the "Main" organization block with double-click and click the "Monitor" icon in the toolbar.



The title bar of the project tree is highlighted in orange. This indicates that the online view for this window is enabled. An additional column with diagnostic icons is displayed to the right of the blocks. The green circle indicates that the blocks between the online view on the simulated module and the configuration in the TIA Portal are identical.

2. The GRAPH FB is not called yet. To start the sequence, you open the simulation in PLCSIM and set the memory bit with the address M100.0. To do this, check the "0" (Bit 0) of the memory byte "100" (MB100).

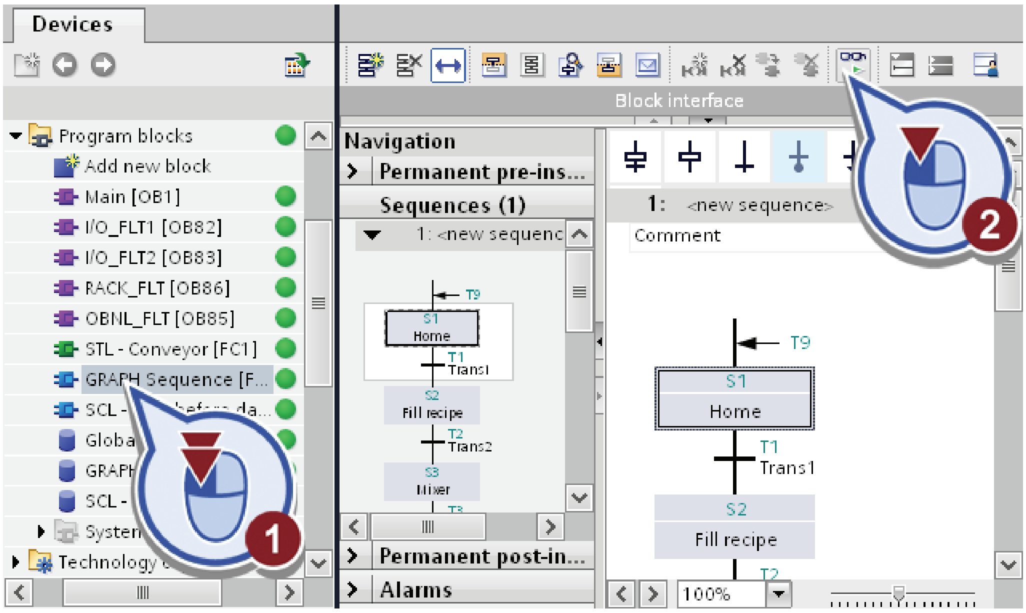


After setting the memory bit, you can monitor in PLCSIM how individual outputs are set and reset by the user program.

3. Go back to the TIA Portal.

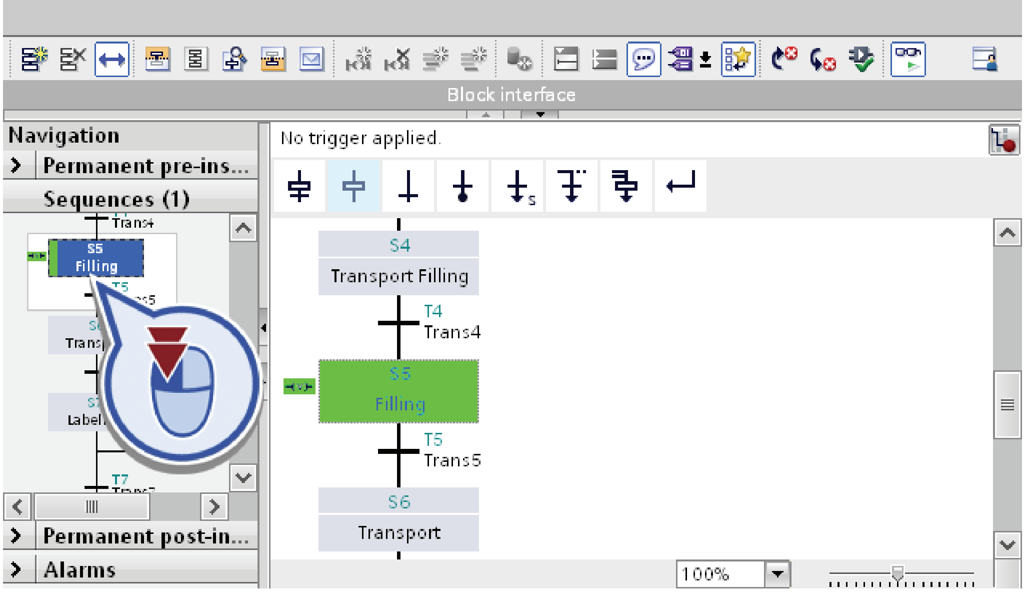
   In the "Main" organization block, the input "OFF_SQ" is reset and the input "INIT_SQ" is set when you call the GRAPH FB.

4. Open the "GRAPH-Sequence" block and activate the "Monitor" function.



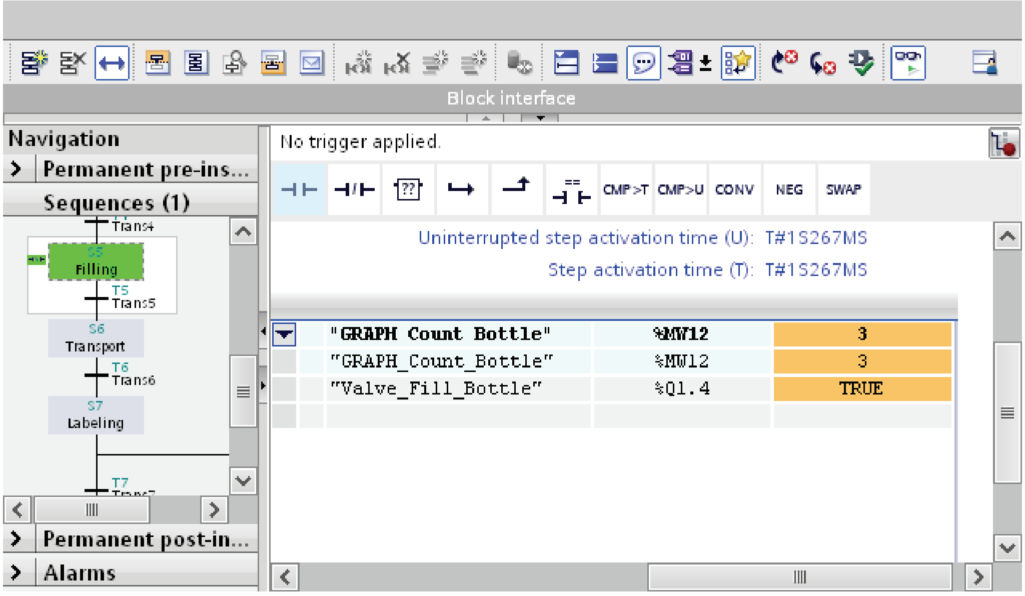
5. Confirm the prompt for switching to test mode with "Yes".

6. The CPU switches to test mode. The sequence was enabled with activation of the M100.0 memory bit. The currently active step is highlighted in green in test mode.
7. Open the "Filling" step to monitor the changes of the used tags.



Result

The current value of the tag "GRAPH_Count_Bottle" as well as the state of the output "Valve_Fill_Bottle" is output in the right-hand column of the section "Actions".



Uninterrupted step activation time (U): T#1S267MS
Step activation time (T): T#1S267MS

```
"GRAPH_Count_Bottle"    %MW12    3
"GRAPH_Count_Bottle"    %MW12    3
"Valve_Fill_Bottle"     %Q1.4    TRUE
```

7.1.3 Testing with sequence control

Introduction

You will now test the execution of the GRAPH sequence with the manual mode of the sequence control. Different operating modes are available in sequence control.

- Automatic mode

  In this operating mode the sequence does automatically switch to the next step as soon as the transition is satisfied.

- Semi-automatic mode

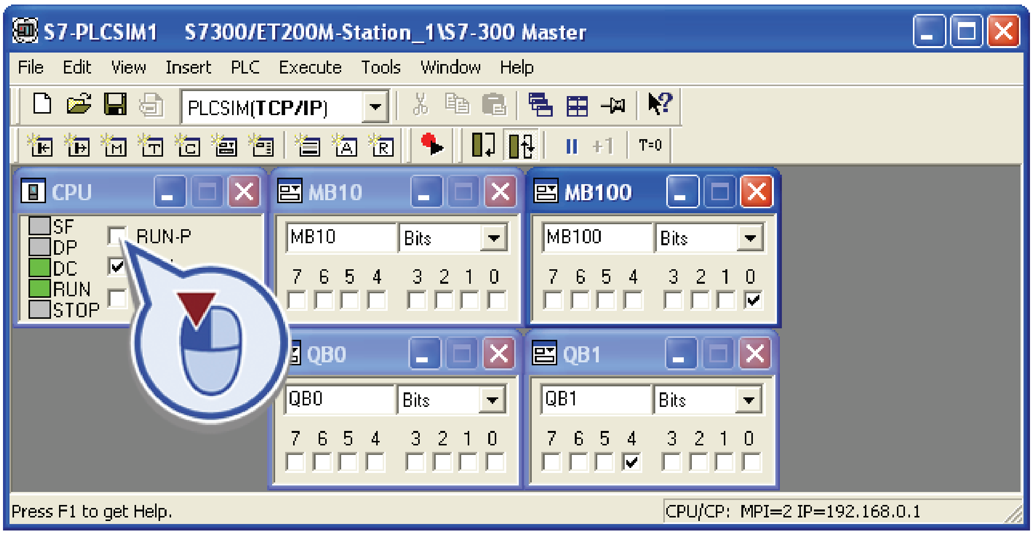
  In this mode, the sequence switches to the next step if the transition is met or the "Ignore transition" button is pressed.

- Manual mode

  In this operating mode the system does not automatically switch to the next step when the transition is satisfied. Instead you select the step to be tested manually.

Procedure

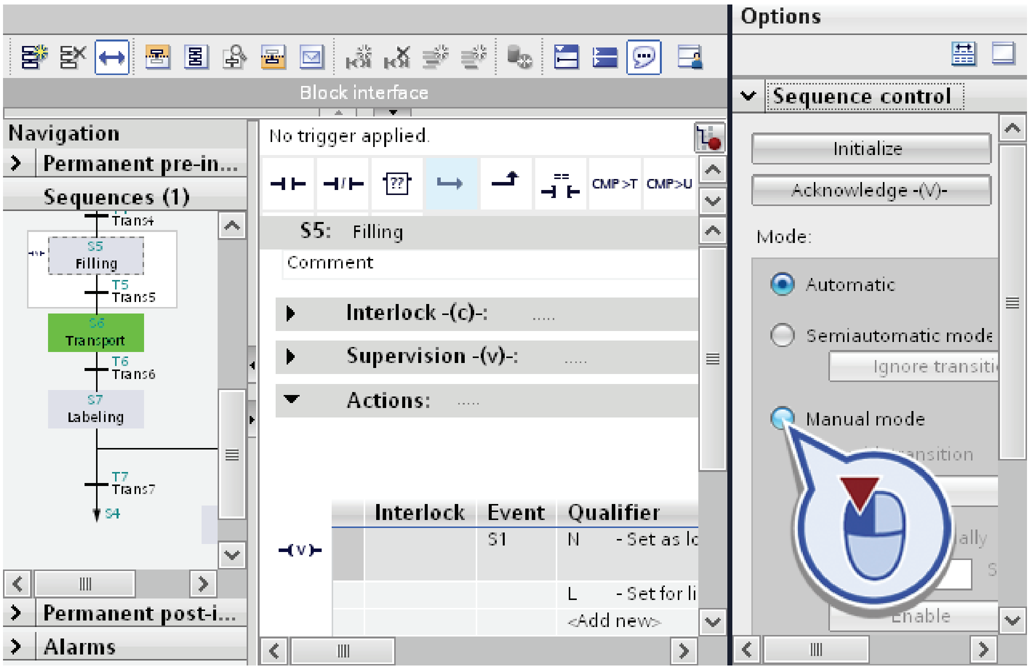
1. Certain interventions in the execution of the program, such as control of the sequence in manual mode, are blocked in the RUN operating mode. To control the sequence in manual mode, set the CPU to RUN-P operating mode in PLCSIM.



2. Go back to the TIA Portal and open the "Testing" tab in the task card.

   "Automatic" is set as operating mode.

3. Switch the sequence control to manual mode.

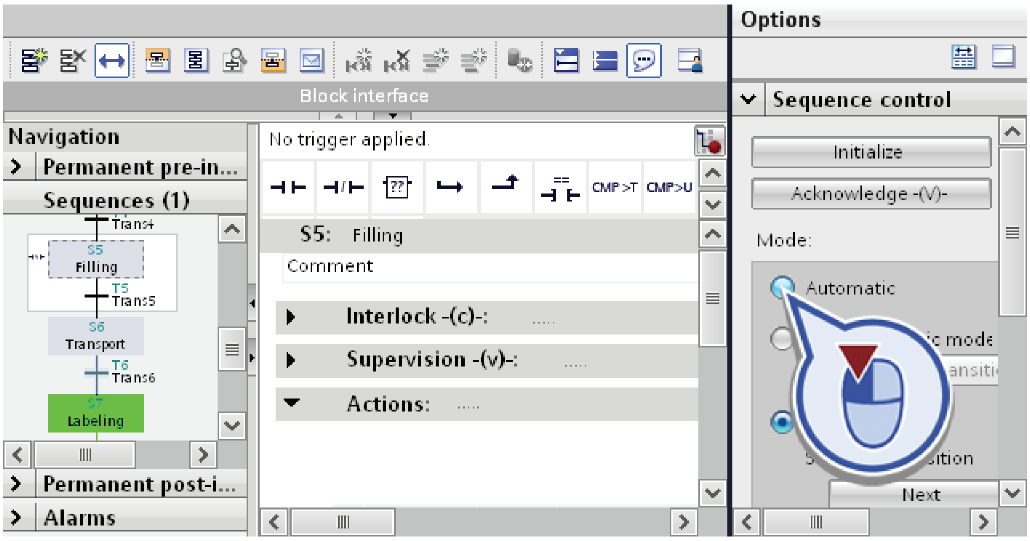


4. Manual mode has been enabled for testing of the sequence. The dialog gives you the following options for testing the sequence:

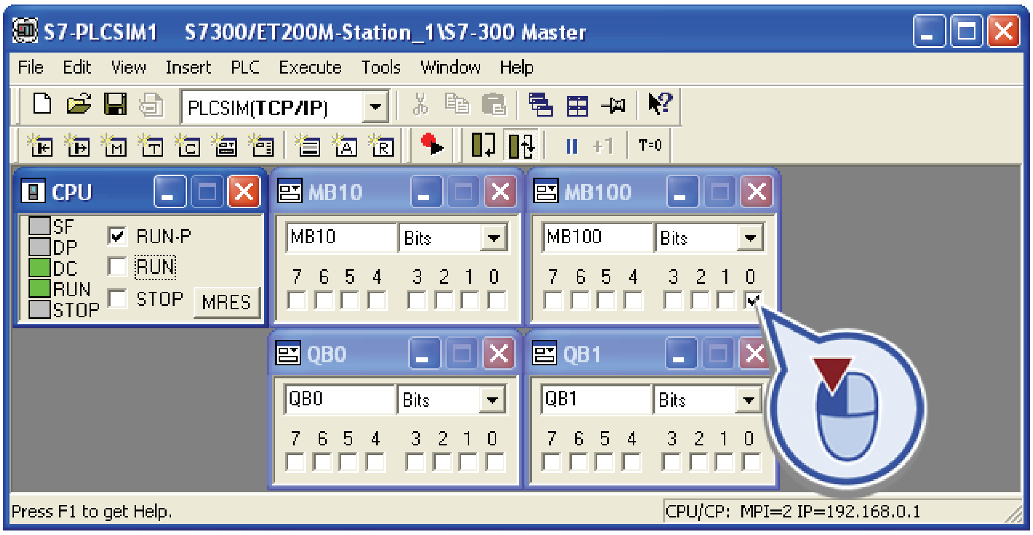
   – You can switch to the next step with the "Next" button. This button will let you check the sequence.

   – Use the "Step number" input field to select any step regardless of the processing status of the sequence. Use the respective buttons to enable or disable the previously selected step.

5. Once you have completed testing in manual mode, reset the sequence back to automatic mode.



6. Then switch back to PLCSIM and reset the memory bit M100.0.



Result

You have successfully tested calling the sequence. Leave the PLCSIM running and retain the online connection after testing is complete to continue with the test of visualization.

7.2 Testing process visualization

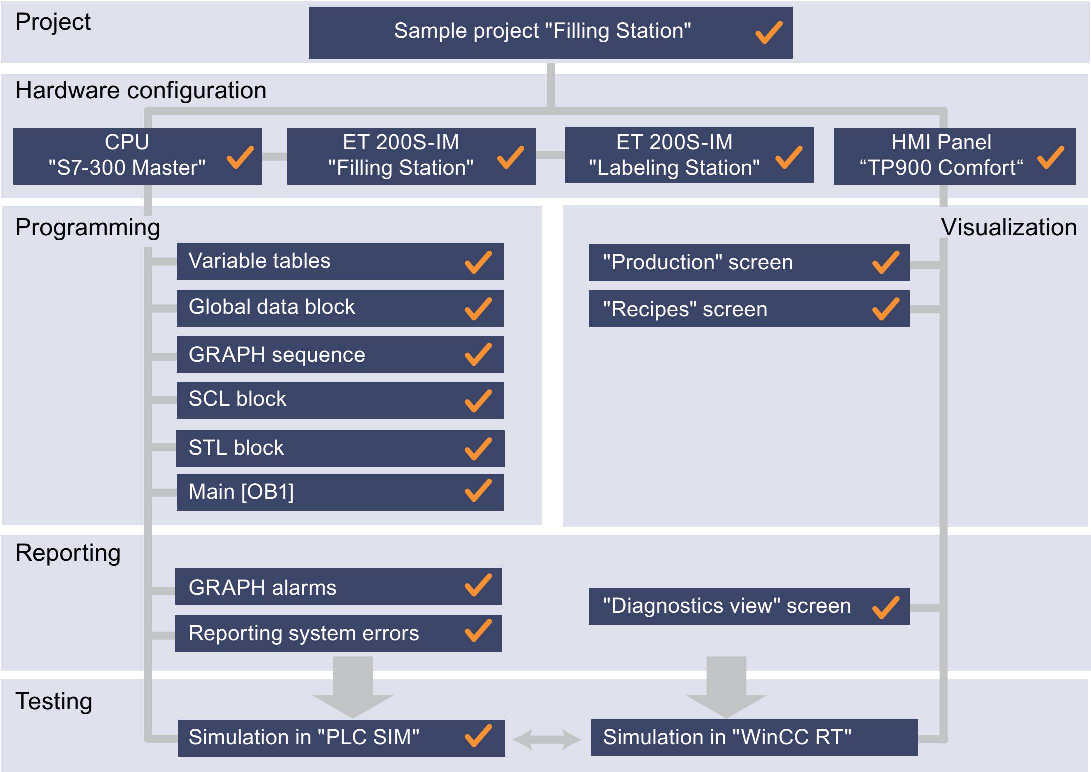
7.2.1 Starting WinCC Runtime

Introduction

You can test the functionality of visualization with the simulation software "WinCC Runtime Advanced". You will now start simulation of the HMI Panel. Using this simulation, you can test whether visualization works without errors before you start production.

Progress of project

The following graphic shows you which step you perform next:

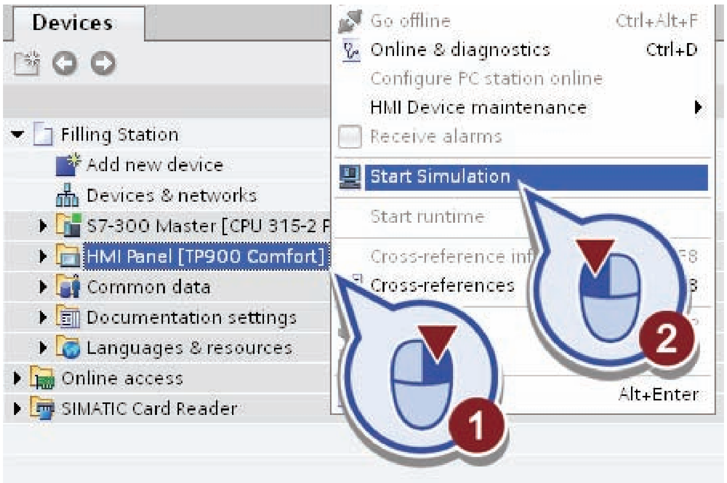


Requirements

The hardware configuration and the program blocks of the CPU are downloaded to PLCSIM and the TIA Portal is in online view. The "WinCC Runtime Advanced" software was installed with the TIA Portal.

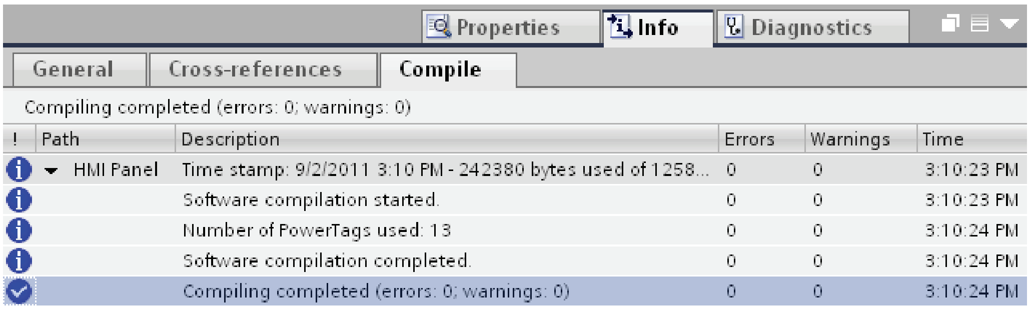
Procedure

1. Right-click on the HMI panel in the project tree and start the simulation of Runtime from the shortcut menu.



Before the start of Runtime, the configured elements of the HMI Panel are automatically compiled. The compilation status is displayed in the Inspector window in the "Info" tab.

2. Check in the Inspector window if the compilation was executed without errors. For an programming error, you can navigate to the affected object with a double-click on the output alarm and make any necessary changes.



Compiling completed (errors: 0; warnings: 0)

!	Path	Description	Errors	Warnings	Time
ℹ	HMI Panel	Time stamp: 9/2/2011 3:10 PM - 242380 bytes used of 1258...	0	0	3:10:23 PM
ℹ		Software compilation started.	0	0	3:10:23 PM
ℹ		Number of PowerTags used: 13	0	0	3:10:24 PM
ℹ		Software compilation completed.	0	0	3:10:24 PM
✓		Compiling completed (errors: 0; warnings: 0)	0	0	3:10:24 PM

Result

The "WinCC Runtime Advanced" software is started. The "Production" root screen is the first screen that is displayed.

7.2.2 Testing the recipes screen

Introduction

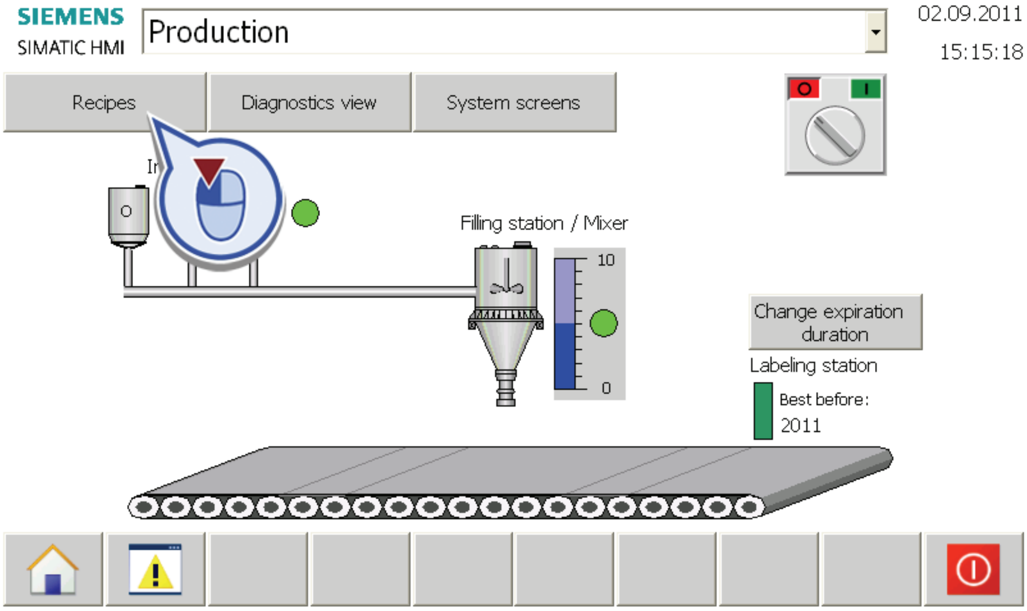
You will now test the functions of the recipe view in Runtime. Then you will test the Production root screen, because the recipe you want to produce must be specified before you start the sequence. To do this, select the recipe "Orange juice" and download the data to the simulated CPU. Then specify the expiration date for the recipe.

Requirement

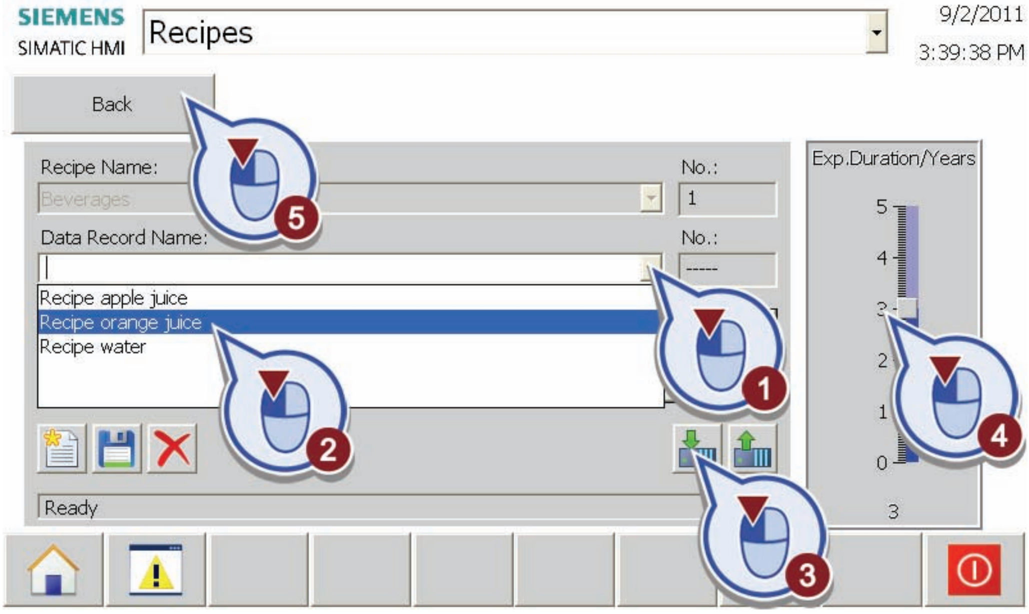
The PLC program was downloaded to PLCSIM and the CPU is RUN-P operating mode. The HMI Panel was loaded in the "WinCC Runtime Advanced" simulation software.

Procedure

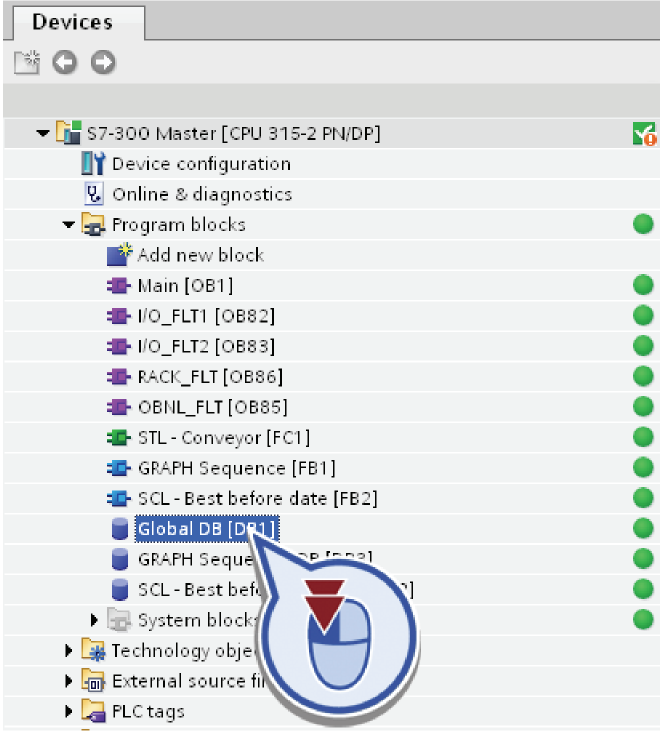
1. Open the "Recipes" screen in Runtime.



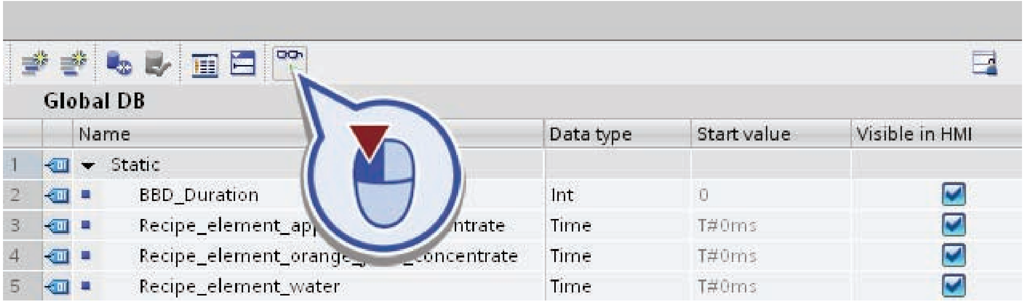
2. Select the recipe data record "Recipe orange juice" and transfer the data record to the CPU. Select an expiration duration of three years using the slide ruler. To return to the root screen "Production", click on the "Back" button.



3. Change to the project tree of the TIA portal and open the "Global_DB" data block.



4. Activate the function "Monitor all" in the data block.

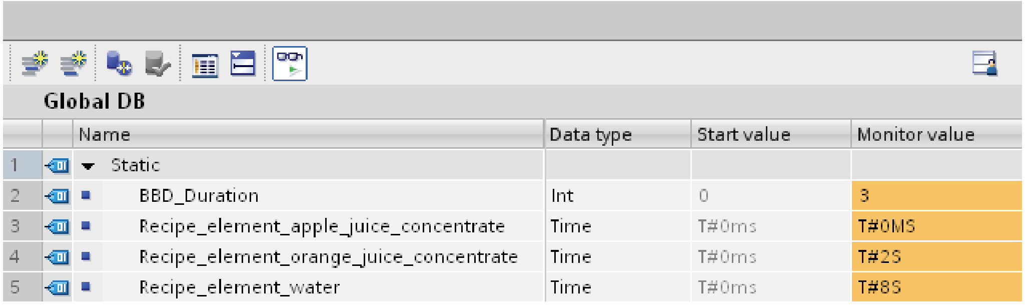


Global DB

	Name	Data type	Start value	Visible in HMI
1	Static			
2	BBD_Duration	Int	0	☑
3	Recipe_element_ap ntrate	Time	T#0ms	☑
4	Recipe_element_orange_ concentrate	Time	T#0ms	☑
5	Recipe_element_water	Time	T#0ms	☑

Result

By selecting the recipe "Orange Juice" in WinCC Runtime and downloading it to the CPU, the duration for filling of the ingredients was transferred to the CPU. The corresponding values for the fill times of the ingredients of the recipe data record are displayed in the data block.



Global DB

	Name	Data type	Start value	Monitor value
1	▼ Static			
2	▪ BBD_Duration	Int	0	3
3	▪ Recipe_element_apple_juice_concentrate	Time	T#0ms	T#0MS
4	▪ Recipe_element_orange_juice_concentrate	Time	T#0ms	T#2S
5	▪ Recipe_element_water	Time	T#0ms	T#8S

7.2.3 Testing production screen

Introduction

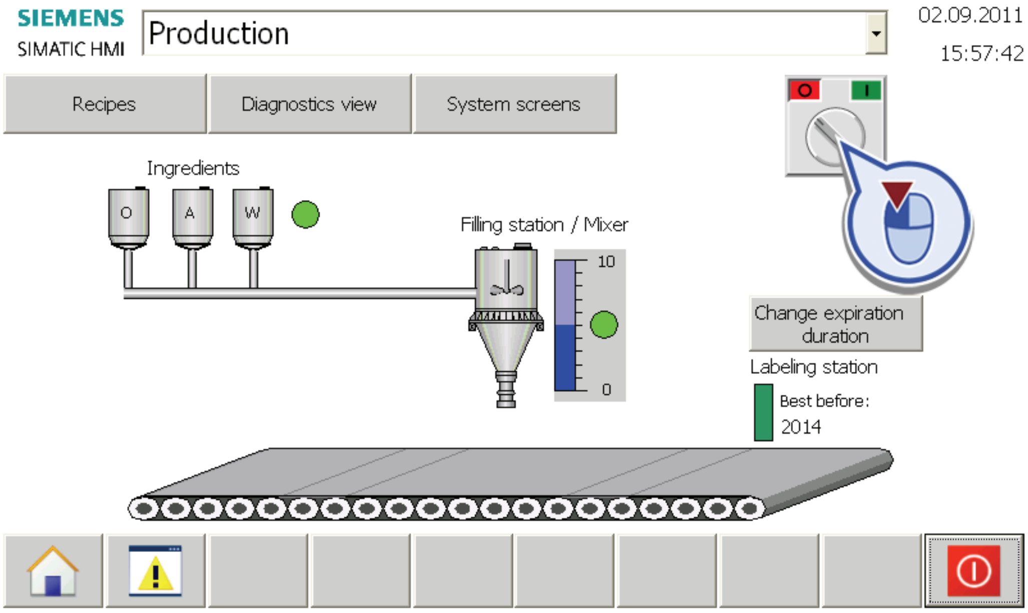
You will now test the visualization of the individual GRAPH steps in the root screen "Production".

Requirement

The PLC program was downloaded to PLCSIM and the CPU is in RUN mode. The HMI Panel was downloaded to "WinCC Runtime Advanced".

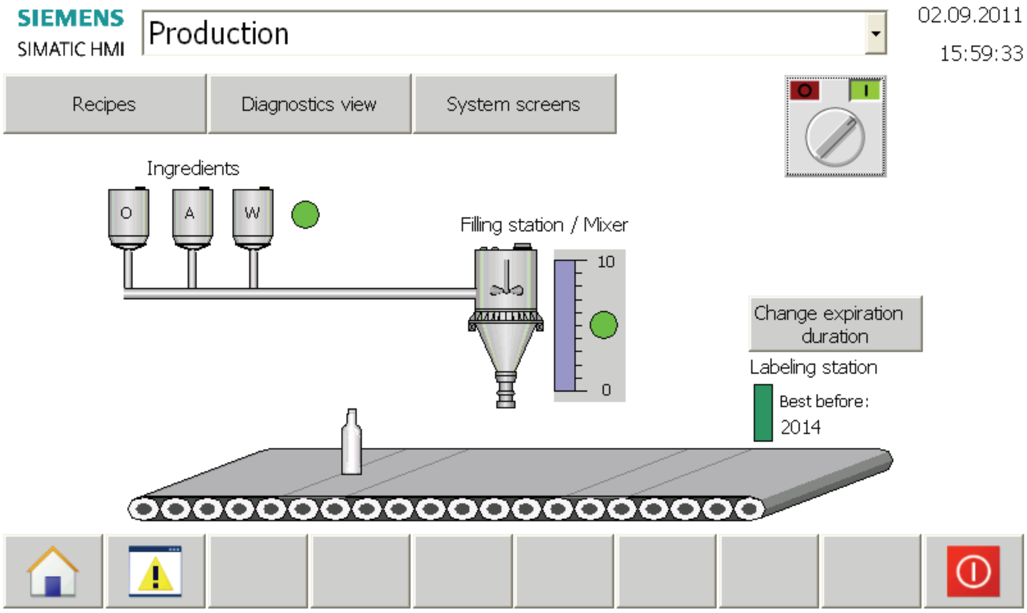
Procedure

1. In the TIA Portal open the GRAPH sequence in the "GRAPH Sequence" function block.
2. Make sure that the function "Monitor on/off" is enabled and that the GRAPH sequence is not being executed at the moment.
3. Switch to the "WinCC Runtime Advanced" software and start the sequence with the rotary switch.



Result

After starting the sequence, the animations for the respective steps are activated.



Parallel to the visualization in WinCC Runtime, you can monitor the processing status of the sequence in the online view of the TIA Portal.

7.2.4 Testing the diagnostics view screen

Introduction

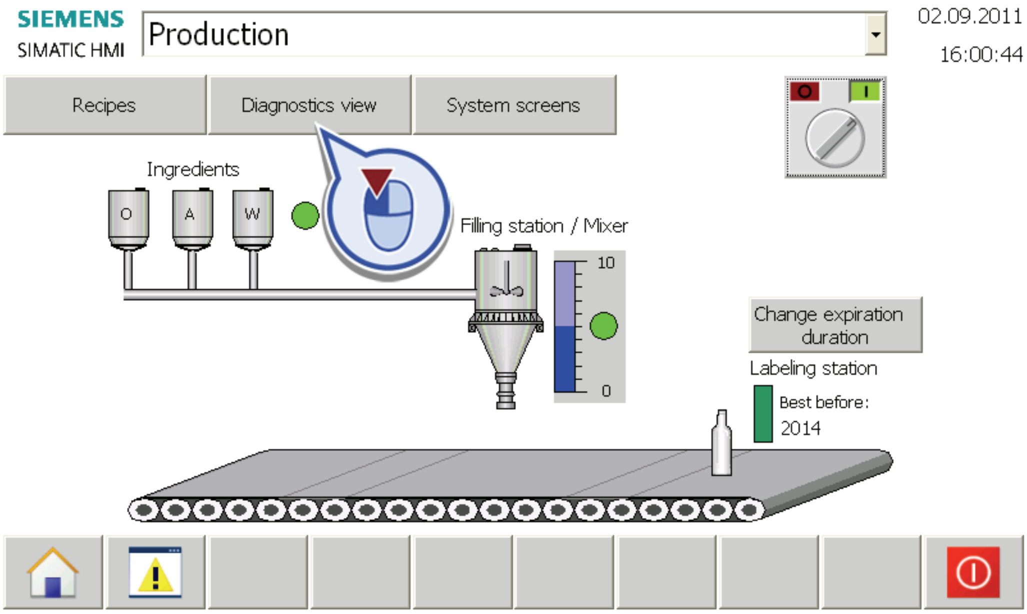
The system diagnostics window gives you an overview of all available devices in your plant. You will first open the device view of the system diagnostics window to check the current status of the module. Then you open the detailed view to display detailed information on the selected device.

Requirement

The PLC program was downloaded to PLCSIM and the CPU is RUN operating mode. The HMI Panel was downloaded to "WinCC Runtime Advanced".

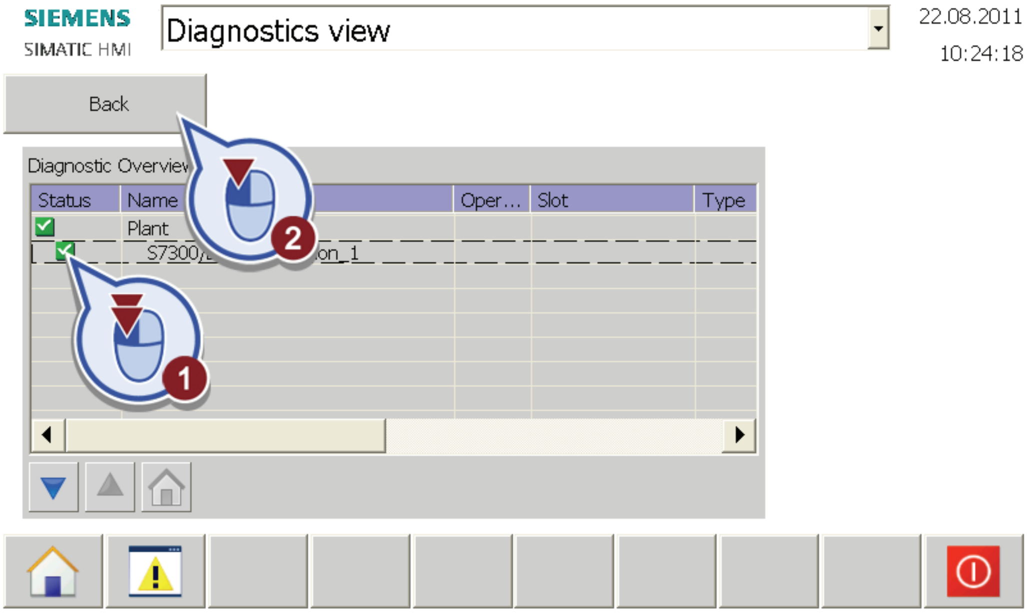
Procedure

1. Open the "Diagnostics view" screen.



In the open diagnostics view, a green check mark against a green backdrop is displayed next to the module. It indicates that the device is currently in operation and that no error has occurred.

2. Open the detail view of the module to display the detailed diagnostics status of the module. Then use the "Back" button to return to the root screen "Production".



Result

You have successfully tested the status of the module. If an error should occur during operation, if the module enters STOP or if there is a maintenance request pending, it will be indicated by a corresponding symbol in the system diagnostics windows.

7.2.5 Testing system screens

Introduction

At the end of the Getting Started, you will test the function "SIMATIC PLC Status/Force" to simulate a group error via the HMI Panel for the entire sequence.

In the GRAPH sequence, an NC contact has been added during program creation for the transition of each individual step and was interconnected with the tag "GRAPH_Group_Fault". You will now set the "GRAPH_Group_Fault" tag to "1" and disable the execution of the sequence.

System screens

When creating the HMI Panel, the following system screens were automatically created with the HMI Device Wizard:

- SIMATIC PLC Status/Force

  You can use this system screen to control and monitor the values of different data areas via HMI connection.

- Project information

  This system screen includes the general project information.

- Different jobs

  You use this system screen to execute basic functions of the HMI Panel, such as changing the language or shutting down Runtime.

- System settings

  This system screen includes the functions for calibrating the screen and a clean screen. The clean screen is used for brief disabling of all operator controls to clean the HMI Panel.

- User administration

  If several users with different rights are defined in the project, you can use this function to change the user. Additional information regarding user administration is available in the online help of the TIA Portal.

- System information

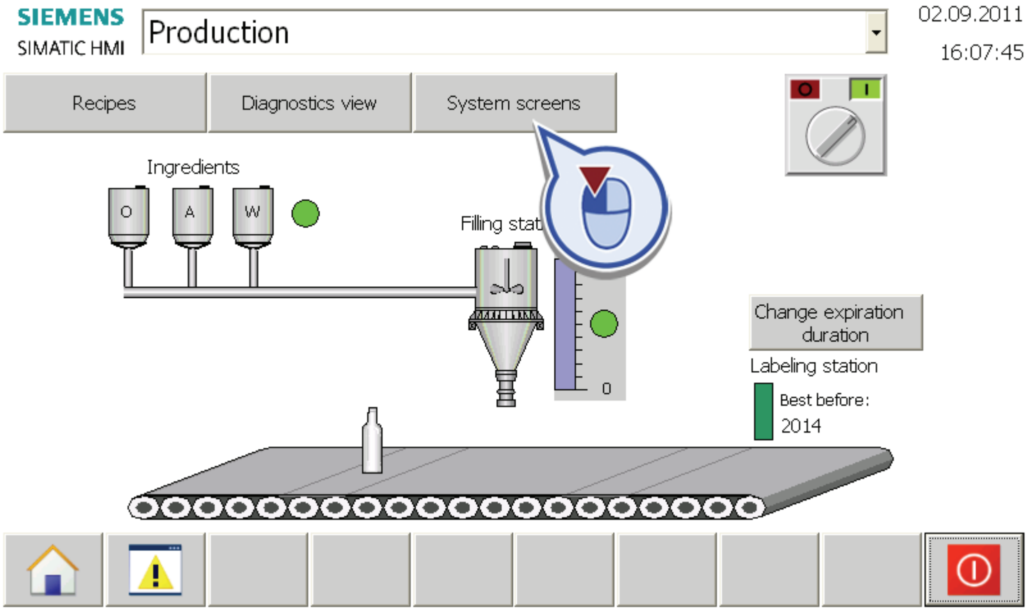
  The system Information contains information on the used HMI Panel, the connections and the connected PLC.

Requirement

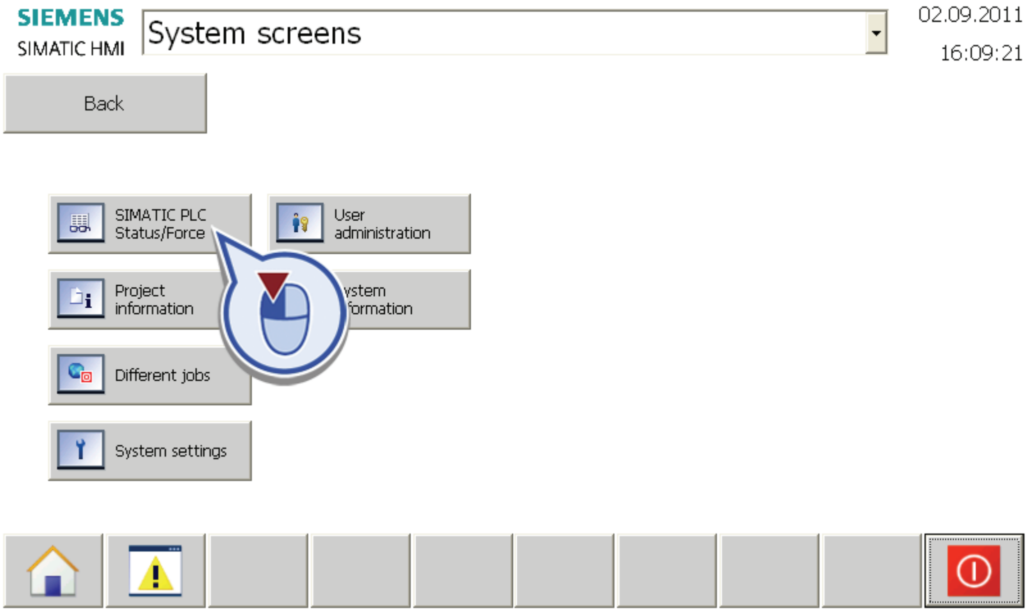
The PLC program was downloaded to PLCSIM and the CPU is RUN operating mode. The online connection is enabled in the TIA Portal and the sequence is switched to test mode. The HMI Panel was downloaded to "WinCC Runtime Advanced".

Procedure

1. Open the system screens in the simulation in "WinCC Runtime Advanced".



2. Select the system screen "SIMATIC PLC Status/Force" from the overview.



3. Select the "HMI connection" to the PLC under "Connection" and enter the following values in the other columns:

   – Type: M (bit memory)

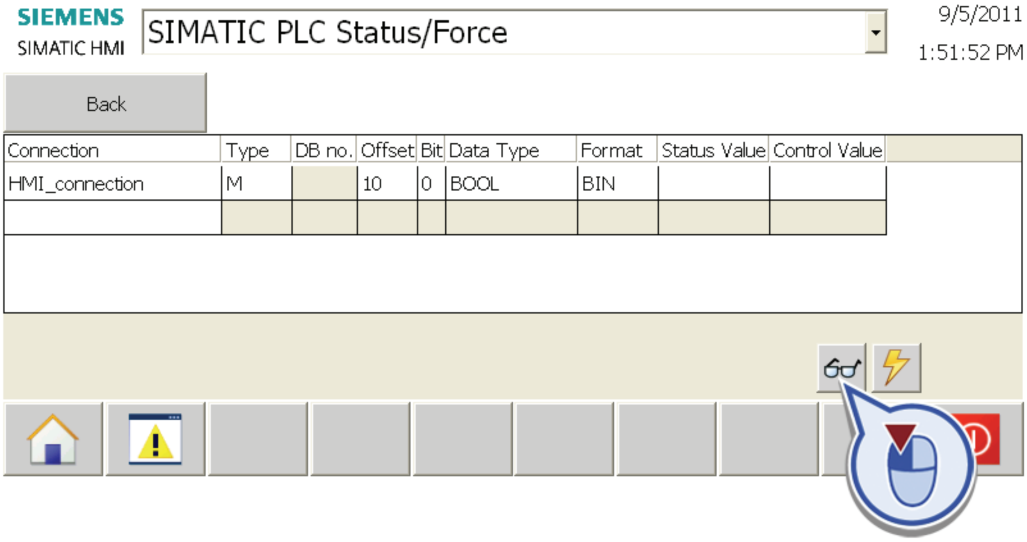
   – Offset: 10 (Memory Byte 10)

   – Bit: 0
     The bit number can only be entered after you have selected "BOOL" as data type.

   – Data type: BOOL

   – Format: BIN

   Then click on the button for monitoring the status value.



SIEMENS SIMATIC HMI

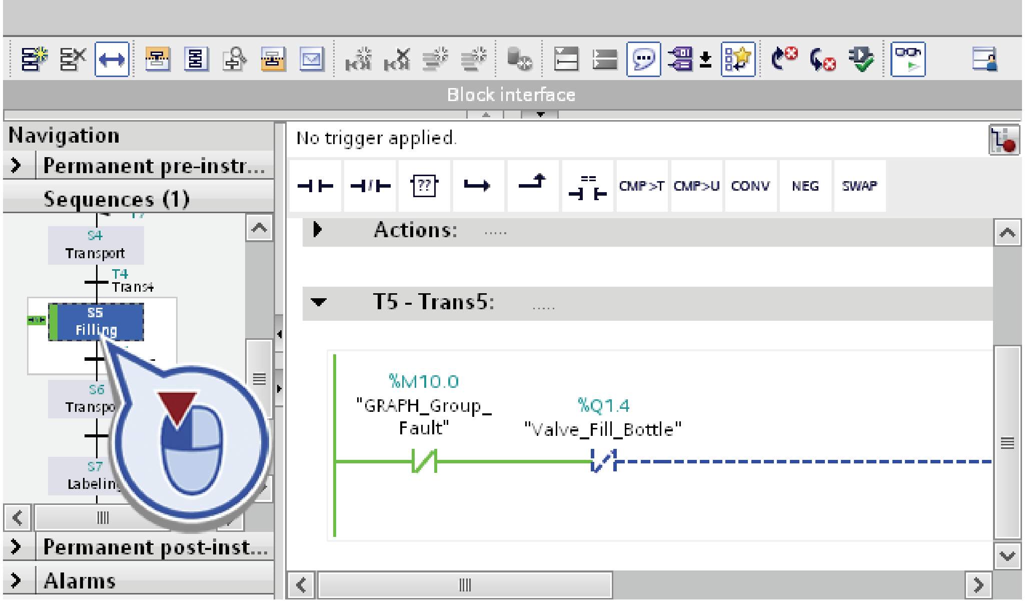
SIMATIC PLC Status/Force

9/5/2011
1:51:52 PM

Connection	Type	DB no.	Offset	Bit	Data Type	Format	Status Value	Control Value
HMI_connection	M		10	0	BOOL	BIN		

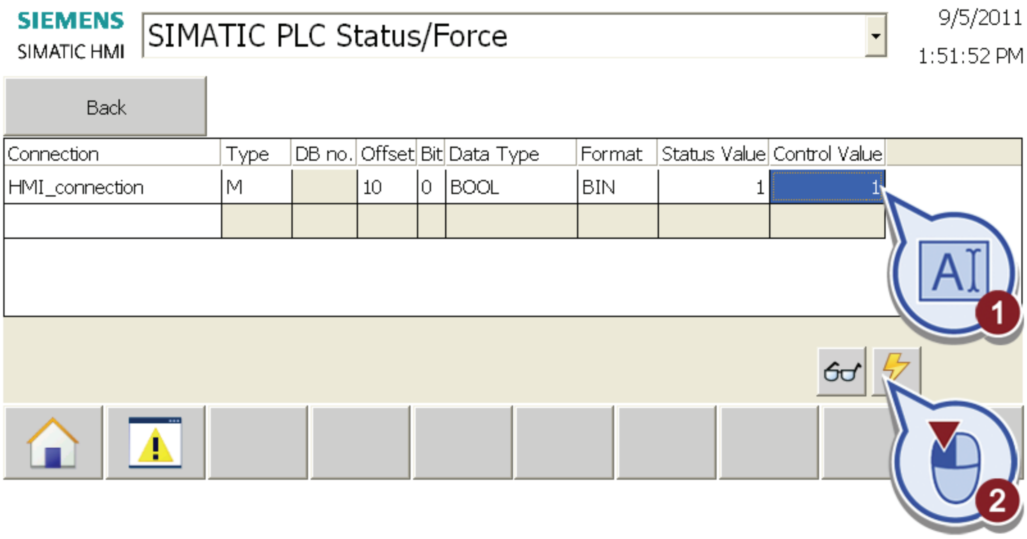
The column "Status Value" outputs a "0".

4. Change to the TIA Portal and open the step "S5 Filling" in the online view of the GRAPH sequence.



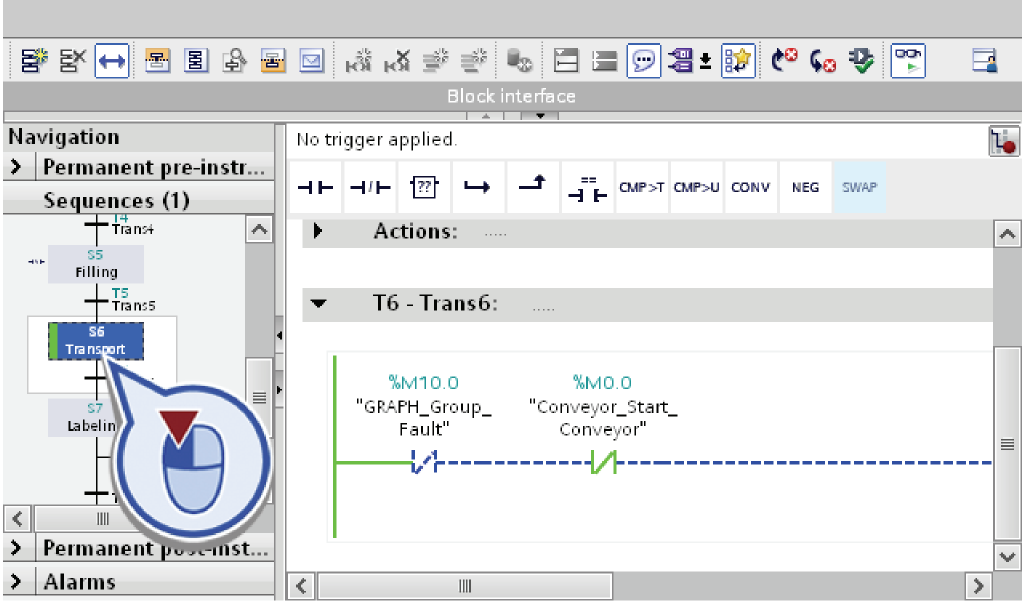
The bit of the NC contact for the tag "GRAPH_Group_Fault" in not set in the section "Transitions". After execution of the actions, the sequence switches to the next step "Transport Filling".

5. Switch to the "WinCC Runtime Advanced" software and stop the ongoing monitoring of the status value. In the Control Value column of the memory bit of the "GRAPH_Group_Fault" tag enter "1" as value and activate the control of the memory bit.



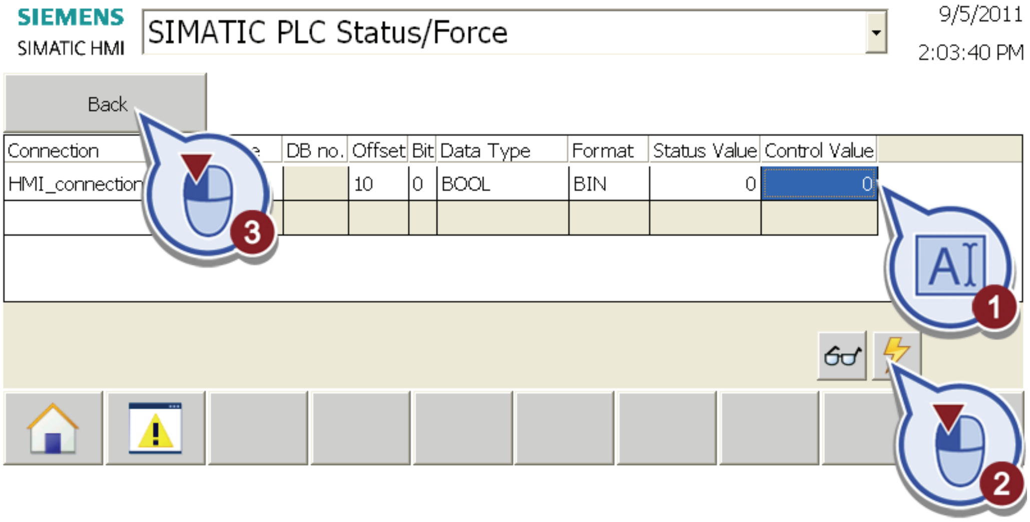
Connection	Type	DB no.	Offset	Bit	Data Type	Format	Status Value	Control Value
HMI_connection	M		10	0	BOOL	BIN	1	1

6. Change to the TIA Portal and open the step at which the sequence was stopped in the online view of the GRAPH sequence. The last active step is shown in green in the left section of the overview of the sequence.



The "Transition" section shows that the step enabling condition is not met as long as the bit for the "GRAPH_Group_Fault" tag is set to "1".

7. Change once again to the "WinCC Runtime Advanced" software and set the control value for the "GRAPH_Group_Fault" tag back to "0".



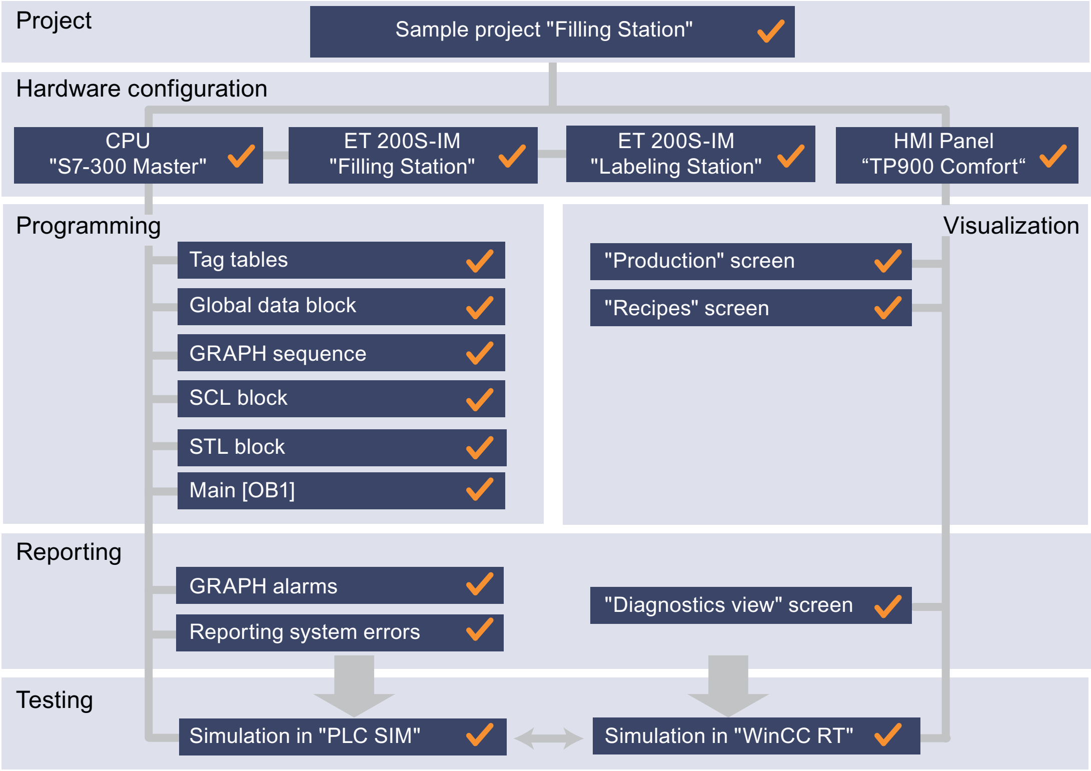
Connection	e	DB no.	Offset	Bit	Data Type	Format	Status Value	Control Value
HMI_connection			10	0	BOOL	BIN	0	0

Result

You have successfully tested the function SIMATIC PLC Status/Force" of the system screens.

Project completion

You have successfully completed this Getting Started by testing the visualization.


Project
Sample project "Filling Station" ✓
Hardware configuration
CPU "S7-300 Master" ✓
ET 200S-IM "Filling Station" ✓
ET 200S-IM "Labeling Station" ✓
HMI Panel “TP900 Comfort“ ✓
Programming
Tag tables ✓
Global data block ✓
GRAPH sequence ✓
SCL block ✓
STL block ✓
Main [OB1] ✓
Visualization
"Production" screen ✓
"Recipes" screen ✓
Reporting
GRAPH alarms ✓
Reporting system errors ✓
"Diagnostics view" screen ✓
Testing
Simulation in "PLC SIM" ✓
Simulation in "WinCC RT" ✓

Downloading the sample project A

A.1 Downloading the sample project

Available project files

The following project versions are available as a ZIP file:

- Sample_Project_Programming.zip

  The project file contains the status up to the end of the chapter "Programming a PLC". If you want to start directly at the chapter "Visualizing the process", download the file and open it in the TIA Portal.

- Sample_Project_Complete.zip

  The project file contains the status at the end of the chapter "Configuring alarms". If you simply want to work through the chapter "Testing online", download the file and open it in the TIA Portal.

Copying project files

Both files are available at the Internet address of the Service&Support Portal http://support.automation.siemens.com/ (http://support.automation.siemens.com/WW/view/en/28919804/133300). To copy the ZIP files, follow these steps:

1. Open the link to the following Internet address:

   http://support.automation.siemens.com/WW/view/en/28919804/133300 (http://support.automation.siemens.com/WW/view/en/28919804/133300)

2. In the Entry list, select the category "Manuals".
3. Open the entry "SIMATIC STEP 7 Professional V11 Getting Started".
4. In the entry, click the "Info" link.
5. Copy the ZIP file "Sample_project.ZIP".
6. Extract the file in a local folder. The file contains both project versions.

⚠ WARNING

Use the sample project only for test purposes

The sample project is only intended to provide you with an introduction to the functions of the TIA Portal.

- Use the sample project only in a test environment and not in an operational plant.
- Loading the sample project in an operational plant can cause malfunctions, program errors and serious property damage and personal injuries!

A.2 Downloading the sample project

Introduction

To download the project file, first open it. Then make the following language settings:

- Setting the project language

  The project language involves the textual content of the sample project. Certain elements such as text fileds, display names and button labels can be created in multiple languages. Regardless of the user interface language, you can choose the language in which the project texts are displayed.

- Language in the runtime settings

  By selecting the language in the runtime settings, you specify which languages will be loaded in WinCC Runtime. You also decide which language will be displayed first when you start the runtime.

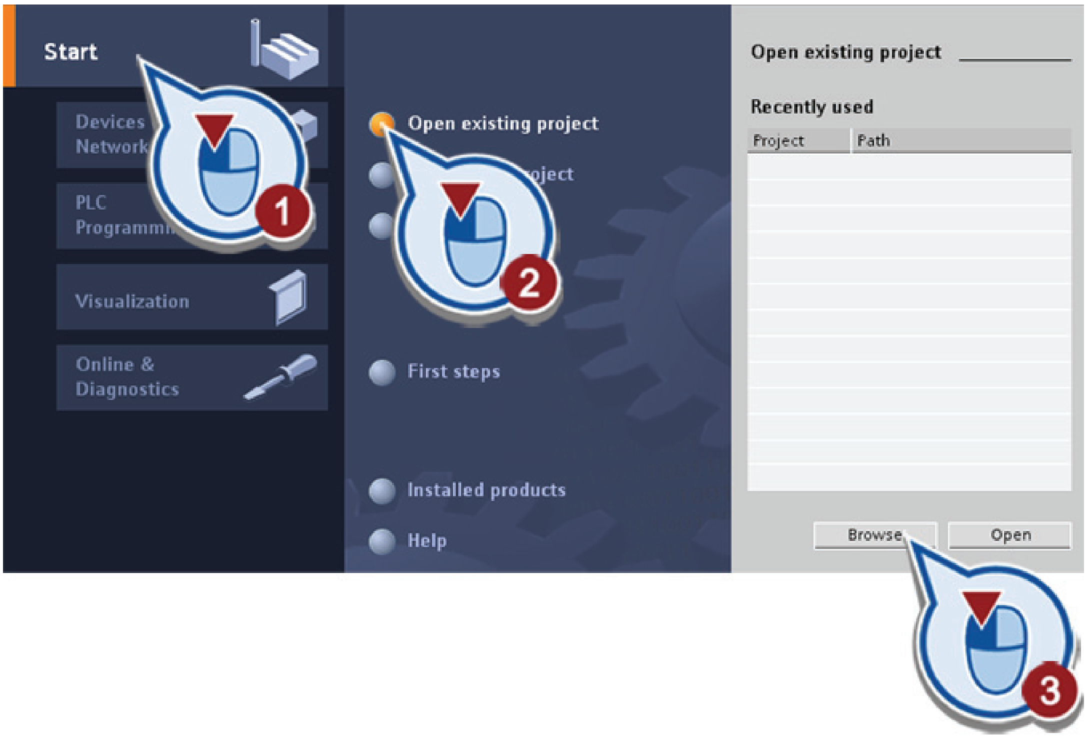
Requirement

You have copied the required project status and extracted it in a local folder.

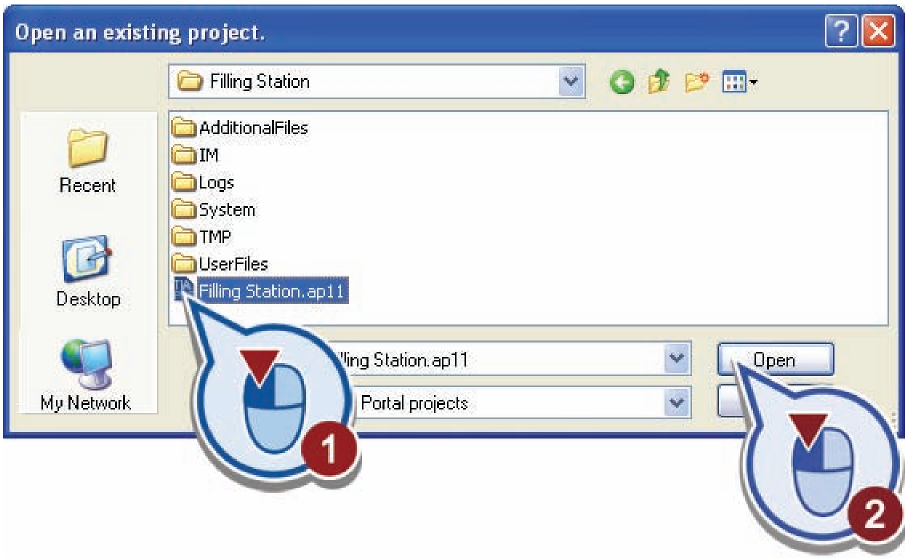
Loading a project

To load a project, follow these steps:

1. In the Portal view, click "Open existing project" and click "Browse".

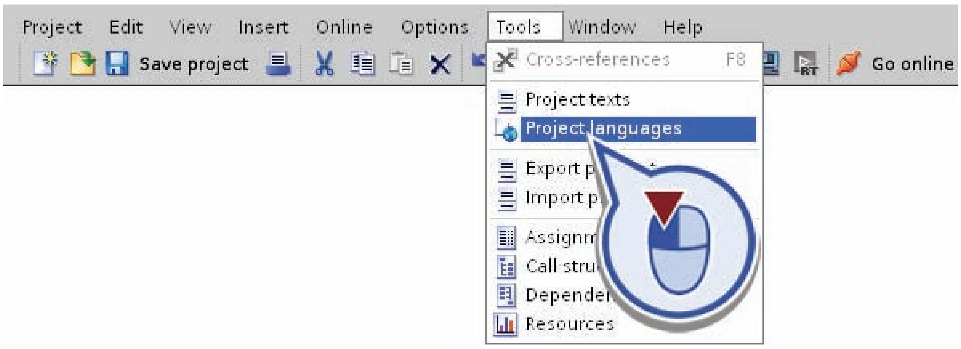


2. Select the folder in which you stored the project and double-click on the project file to open it.

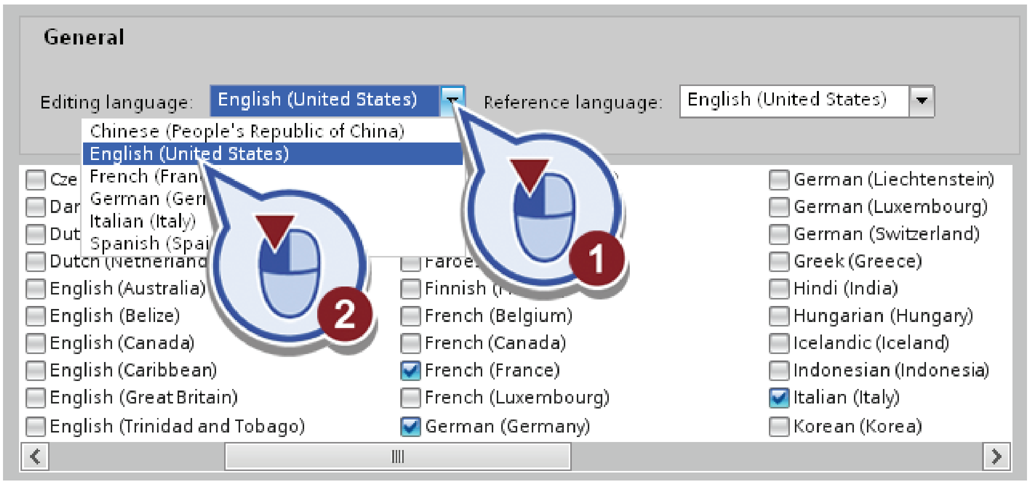


3. Change to the project view.

4. From the "Tools" menu, select the "Project languages" function. The function is enabled only after you have selected an element of the loaded project in the project tree.



5. Select the editing language you want to use. The drop-down list contains only the languages selected with the check boxes in the lower area. The selected reference language is displayed for the language-dependent texts if there are no texts in the editing language.

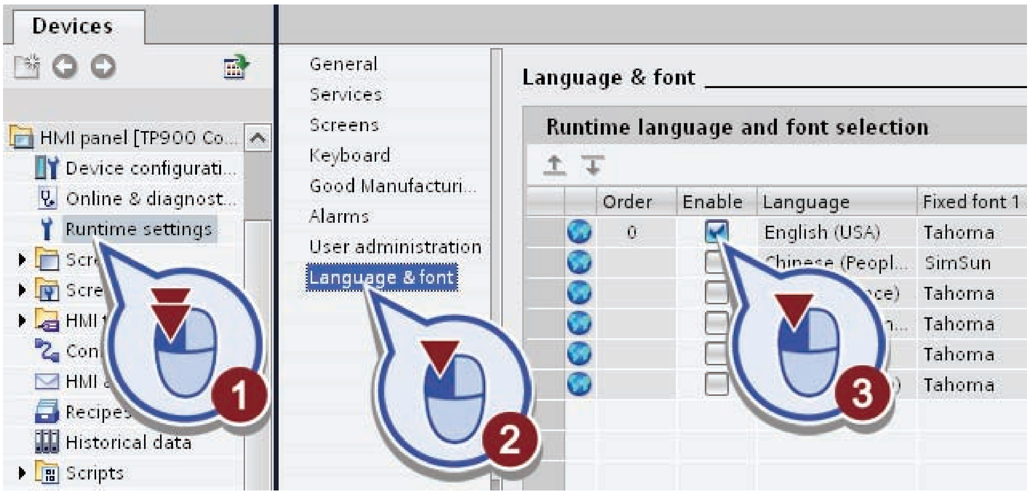


General

Editing language: English (United States)

Reference language: English (United States)

6. If you have loaded the sample project of the "Sample_Project_Complete" ZIP file, select the language you require to display the user interface texts of the HMI screens in the runtime settings of the HMI panel.



Order	Enable	Language	Fixed font 1
0	☑	English (USA)	Tahoma
	□	Chinese (Peopl...	SimSun
	□	ce)	Tahoma
	□	n...	Tahoma
	□		Tahoma
	□	)	Tahoma

The selected language with number "0" in the order is used for the user interface elements when runtime is started.

Result

You have loaded the project and made all the necessary language settings.

- All texts that can be created in multiple languages in a project are displayed in the selected language.
- Texts that are not language-dependent and that uniquely identify individual objects within the project such as tag names or module names are not affected if the language is changed.

Glossary

Address

Designation of a certain address in the input, output or bit memory area of the CPU.

Addressing

Assignment of an address in the user program. Addresses can be assigned specific operands or ranges of operands. Examples: Input I12.1; memory word MW25.

Automation system

An automation system is a programmable logic controller (PLC) consisting of a central controller, a CPU, and various input/output modules.

Bit memory

Memory area in the system memory of a CPU. This can be written and read (in bits, bytes, words, and double words). The bit memory is available to the user to save interim results.

Block

Structures the user program in independent sections. Parts of the user program can be subdivided into blocks that can be reused at various locations or to make the structure of the user program more straightforward.

Block parameters

Placeholder within blocks that can be used more than once and that are supplied with current values when the relevant block is called.

Box

Boxes are program elements with complex functions. The empty box is an exception. You can use the empty box as a placeholder in which you can select the required operation.

Coil

You can use coils to modify binary operands. Coils can set or reset a binary operand depending on the signal state of the result of logic operation.

Configuring

"Configuring" is understood to mean arranging, setting and networking devices and modules within the device or network view. Racks are represented symbolically. Just like "real" racks, they allow you to plug in a defined number of modules.

Contact

You can use contacts to create or interrupt a current-carrying connection between two elements. The current is relayed from left to right. You can use contacts to query the signal state or the value of an operand and control it depending on the result of the current flow.

CPU

The user program is stored and executed in the central processing unit (CPU) of an automation system. It contains the operating system, processing unit, and communication interfaces.

CPU operating system

The operating system organizes all functions and sequences of the CPU that are not associated with a specific control task.

Cycle time

The cycle time is the time that the CPU requires to execute the user program once.

Cyclic interrupt

Cyclic interrupt OBs serve to start program in periodic time intervals independently of the cyclic program execution. The start times of a cyclic interrupt OB are specified using the time base and the phase offset.

Data block (DB)

Block in the user program that is used for storing values or character strings. There are global data blocks that can be accessed by all code blocks and instance data blocks that are assigned to a specific FB call.

Data type

Specifies how the value of a tag or constant is to be used in the user program. A tag of the BOOL data type can, for example, only take the value 1 or 0.

Function block (FB)

According to IEC 1131-3, a function block is a code block with static data. An FB allows you to pass parameters in the user program. This makes function blocks ideally suited for programming frequently recurring complex functions, such as closed-loop control or operating mode selection. Because a function block has a memory (instance data block), its parameters can be accessed at any time and at any point in the user program.

HMI device

Screen device for displaying the status, progress and operation of the user program.

I/O field

The I/O field is an input and output field that serves for displaying and changing the tag values.

Input

Memory area in the system memory of the CPU (process image input) or connection to an input module.

Instance data block

An instance data block stores the formal parameters and static data of function blocks. An instance data block can be assigned to an FB call or to a call hierarchy of function blocks.

Library

Collection of elements that can be used more than once.

Modify tags

Using the "modify tags" function, you can modify the tags of a user program and assign permanent values to individual tags at a specified point during execution of the user program.

Network

The program of a block is divided into networks. The networks are used to structure programs.

Organization block

Organization blocks (OBs) are the interface between the CPU operating system and the user program. The order in which the user program executes is specified in organization blocks.

Output

Memory area in the system memory of the CPU (process image of outputs) or connection to an output module.

PLC

Programmable logic controllers (PLCs) are electronic controllers whose functionality is stored as a program on the control device. The installation and wiring of the device do not therefore depend on the function of the PLC. A PLC is made up of at least one power supply module, one CPU, and input and output modules.

PLC tag table

A table used to define tags valid in the entire CPU.

Process image

The signal states of the digital input and output modules are stored on the CPU in a process image. A distinction is made between the process image input (PII) and the process image output (PIQ).

The process image output (PIQ) is transferred by the operating system to the output modules before the execution of the user program and before the reading of the process image input.

The process image input (PII) is read from the input modules by the operating system before the user program is started.

Program

A program solves a self-contained control task. It is assigned to a programmable module and can be structured in smaller units, for example blocks.

Programming language

A programming language is used to create user programs and provides a defined language subset for this purpose in the form of graphical or textual instructions. These instructions are entered by the user with an editor and then compiled into an executable user program.

Runtime

The runtime software executes the project in process mode and allows processes to be operated and monitored.

Subnet

A subnet includes all the network devices interconnected without gateways. It can include repeaters.

Tag

A tag is made up of an address and a symbolic name that is generally used multiple times in the project. The address (for example, of an input or bit memory) is used in communication with the automation system. Tags are used to perform a change of address (for an input, for example) centrally instead of throughout the entire user program.

Target system

Automation system on which the user program runs.

Watch table

The purpose of the watch table is to assemble the tags from the user program that are to be monitored, modified, and/or forced.