




SIEMENS



# SCE Training Curriculum for Integrated Automation Solutions Totally Integrated Automation (TIA)

Siemens Automation Cooperates with Education

## TIA Portal Module 040-020

Startup Programming in High-Level Languages  
with S7-SCL and SIMATIC S7-300

Cooperates  
with Education

Automation



SIEMENS

## Suitable SCE trainer packages for these training curriculums

### SIMATIC controllers

- **SIMATIC S7-300 with CPU 314C-2PN/DP**  
Order no.: 6ES7314-6EH04-4AB3
- **SIMATIC S7-300 with CPU 314C-2PN/DP (upgrade)**  
Order no.: 6ES7314-6EH04-4AB4
- **SIMATIC S7-300 with CPU 315F-2PN/DP**  
Order no.: ES7315-2FH14-4AB1
- **SIMATIC ET 200S with CPU IM151-8 F PN/DP**  
Order no.: 6ES7151-8FB00-4AB1

### SIMATIC STEP 7 software for training

- **SIMATIC STEP 7 Professional V11 - Single license**  
Order no.: 6ES7822-1CC01-4YA5
- **SIMATIC STEP 7 Professional V11 - Classroom license (up to 12 users)**  
Order no.: 6ES7822-1AA01-4YA5
- **SIMATIC STEP 7 Professional V11 - Upgrade license (up to 12 users)**  
Order no.: 6ES7822-1AA01-4YE5
- **SIMATIC STEP 7 Professional V11 - Student license (up to 20 users)**  
Order no.: 6ES7822-1AC01-4YA5

Please note that these trainer packages may be replaced by successor trainer packages.  
An overview of the currently available SCE packages is provided under: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

## Advanced training

Please get in touch with your regional SCE contact for information on regional Siemens SCE advanced training  
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

## Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

## Information regarding usage

This SCE training curriculum for the end-to-end automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for educational purposes for public educational institutions and R&D facilities. Siemens AG does not make any guarantee regarding its contents.

This document may only be used for initial training on Siemens products/systems. That is, it may be copied in whole or in part and handed out to participants for use within the context of their education. Distribution and reproduction of this curriculums and disclosure of its contents are permitted within public education and further education institutions for educational purposes.

Any exceptions require written consent from the Siemens AG contact person: Mr. Roland Scheuerer  
[roland.scheuerer@siemens.com](mailto:roland.scheuerer@siemens.com).

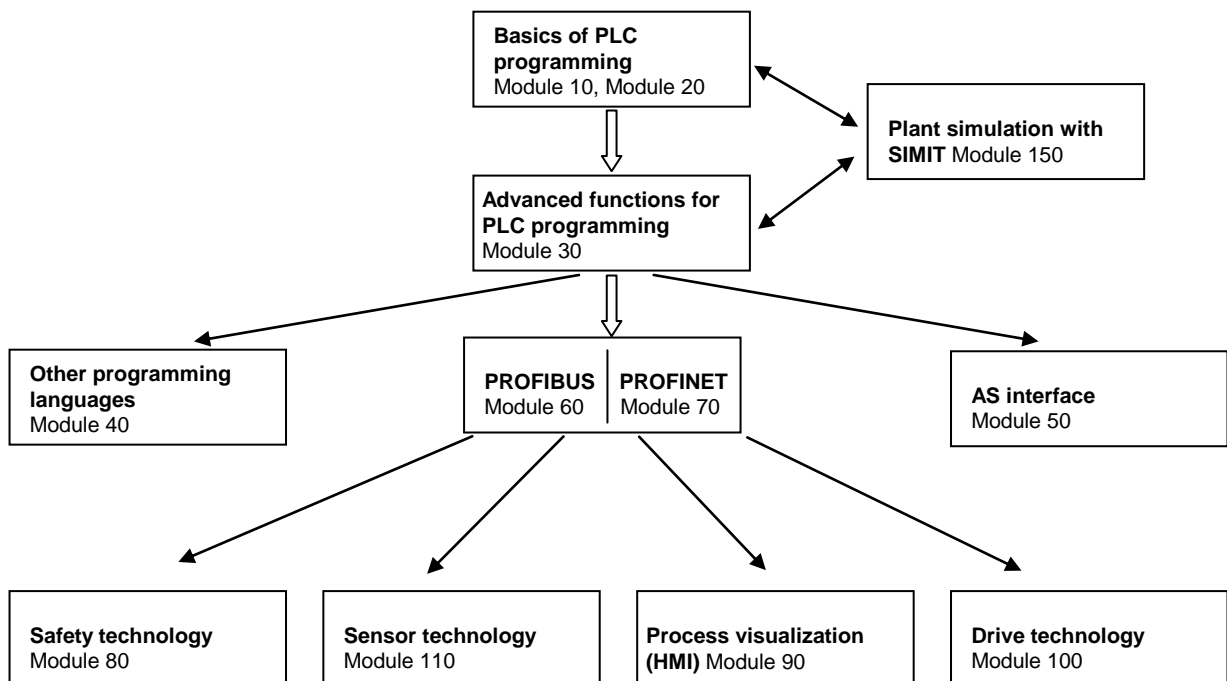
Offenders will be liable for damages. All rights reserved, including those relating to translation and in particular those rights created as a result of a patent being granted or utility model being registered.  
Use for industry customers is expressly prohibited. We do not consent to any commercial use of the training curriculum.

We would like to thank Michael Dziallas Engineering and all those involved for their support in creating this training curriculum.

	<b>PAGE:</b>
1. Preface .....	4
2. Notes on S7-SCL Programming Language .....	6
3. The S7-SCL Development Environment .....	7
4. Example Task Tank Content .....	8
4.1 Description of task .....	8
4.2 Assignment list / tag table .....	8
4.3 Program structure .....	9
4.4 Interface of the block calculate_tankcontent [FC140] .....	10
4.5 Note on solution .....	11
5. Programming the Tank Content Calculation for SIMATIC S7-300 in S7-SCL.....	12
5.1 Creating a project and configuring hardware.....	12
5.2 Creating a program.....	16
5.3 Test program .....	23
5.4 Expanding the program .....	26

## 1. Preface

The content of the SCE\_EN\_040-020 module is part of the **'Other Programming Languages'** training unit and represents a **fast entry point** for programming the SIMATIC S7 300 with the **S7-SCL** programming language with the TIA Portal.



### Learning objective:

This module describes the basic functions of the S7 SCL development environment. Test functions for eliminating logical programming errors will also be shown.

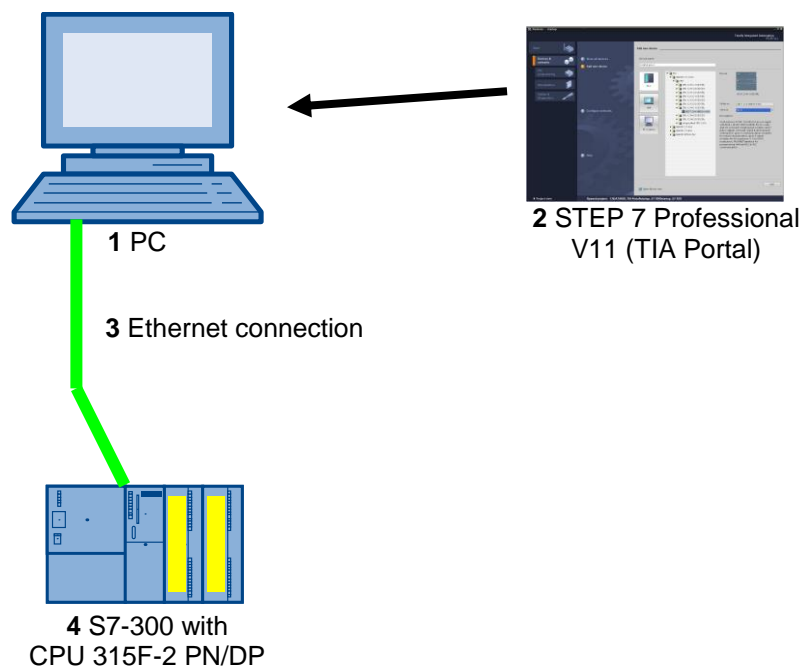
### Requirements:

To successfully work through this module, the following knowledge is required:

- Proficiency in working with Windows
- Basics of PLC programming with STEP 7 Professional V11 (e.g. modules 020- to 030-)
- Basic information on programming in high-level languages, such as Pascal.

**Required hardware and software**

- 1 PC Pentium 4, 1.7 GHz 1 (XP) – 2 (Vista) GB RAM, approx. 2 GB of free hard disk space  
Operating system Windows XP Professional SP3 / Windows 7 Professional / Windows 7 Enterprise / Windows 7 Ultimate / Windows 2003 Server R2 / Windows Server 2008 Premium SP1, Business SP1, Ultimate SP1
- 2 Software: STEP 7 Professional V11 SP1 (Totally Integrated Automation (TIA) Portal V11)
- 3 Ethernet connection between the PC and CPU 315F-2 PN/DP
- 4 SIMATIC S7-300 PLC, e.g., CPU 315F-2PN/DP with 16DI/16DO signal module. The inputs must be fed out to a control panel.



## 2. Notes on S7-SCL Programming Language

S7-SCL (Structured Control Language) is a high-level programming language based on PASCAL, which allows structured programming. The language corresponds to the SFC "Sequential Function Chart" language specified in the standard DIN EN-61131-3 (IEC 61131-3). In addition to high-level programming language elements, S7-SCL contains typical elements of the PLC such as inputs, outputs, timers, bit memories, block calls etc. as language elements. It supports the block concept of STEP 7 and allows data blocks to be programmed according to standard as well as using STL, LAD and FBD. In other words, S7-SCL supplements and extends the STEP 7 programming software with its LAD, FBD and STL programming languages.

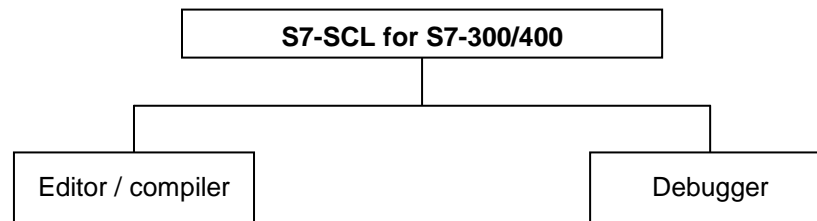
You do not have to create every function yourself. Instead, you can use already prepared blocks such as system functions or system function blocks that are present in the operating system of the CPU.

Blocks that are programmed with S7-SCL can be mixed with STL, LAD and FBD. This means that a block programmed with S7-SCL can call another block programmed in STL, LAD or FBD. In turn, S7-SCL blocks can also be called in STL, LAD and FBD.

The test functions of S7-SCL allow searching for logical programming errors in an error-free compilation.

### 3. The S7-SCL Development Environment

To use and deploy S7-SCL there is a development environment that is oriented to specific features of both S7-SCL and STEP 7. This development environment consists of an editor/compiler and a debugger.



#### Editor / compiler

The S7-SCL editor is a text editor with which texts can be edited. The central task you use the editor for is to create and edit blocks for STEP 7 programs. The syntax of the text you input with the editor is checked thoroughly, making error-free programming simple. Syntax errors are displayed in various colors.

The editor offers the following options:

- Programming a S7 block in the S7-SCL language.
- Convenient insertion of language elements and block calls using drag-and-drop operation.
- Direct syntax check during programming.
- Adjustment of the editor to your requirements by, for example, syntax-oriented coloring of the various language elements.
- Checking of the finished blocks by compiling.
- Display of all errors and warnings that occur during compiling.
- Localizing of the faulty position in the block, optionally with description of error and information on debugging.

#### Debugger

The S7-SCL debugger provides the option of checking how the program runs in the AS and thus finding any possible errors.

S7-SCL offers two different test modes for this:

- Step-by-step monitoring
- Continuous monitoring

With "Step-by-step monitoring" the logical execution of the program is completed. You can execute the program algorithm instruction for instruction and monitor in a result window how the processed tag contents change during the process.

With "Continuous monitoring" you can test a group of instructions within a block. The values of the tags and parameters are displayed in chronological order during the test run and - where possible - updated cyclically.

## 4. Example Task Tank Content

### 4.1 Description of task

Our first program is one to program the calculate the content of tank.

The tank has the shape of an upright cylinder. The fill level of the content is measured by an analog sensor. In the task, the value of the fill level should already existed standardized in the meter unit.

A function FC 140 'calculate\_tank content' is to be programmed in the program. The diameter and the fill level in meter unit are the passed parameters. Result is the tank content in the liter unit.

### 4.2 Assignment list / tag table

Because modern programming uses tags and not absolute addresses, the **global PLC tags** must be defined here.

These global PLC tags are descriptive names with a comment for each input and output used in the program. The global PLC tags can then be accessed later during programming via their names. These global tags can be used in all blocks anywhere in the program.

#### Default tag table

Name	Data type	Address	Comment
FillLevel_tank1	REAL	%MD40	in meters
Diameter_tank1	REAL	%MD44	in meters
Content_tank1	REAL	%MD48	in liters



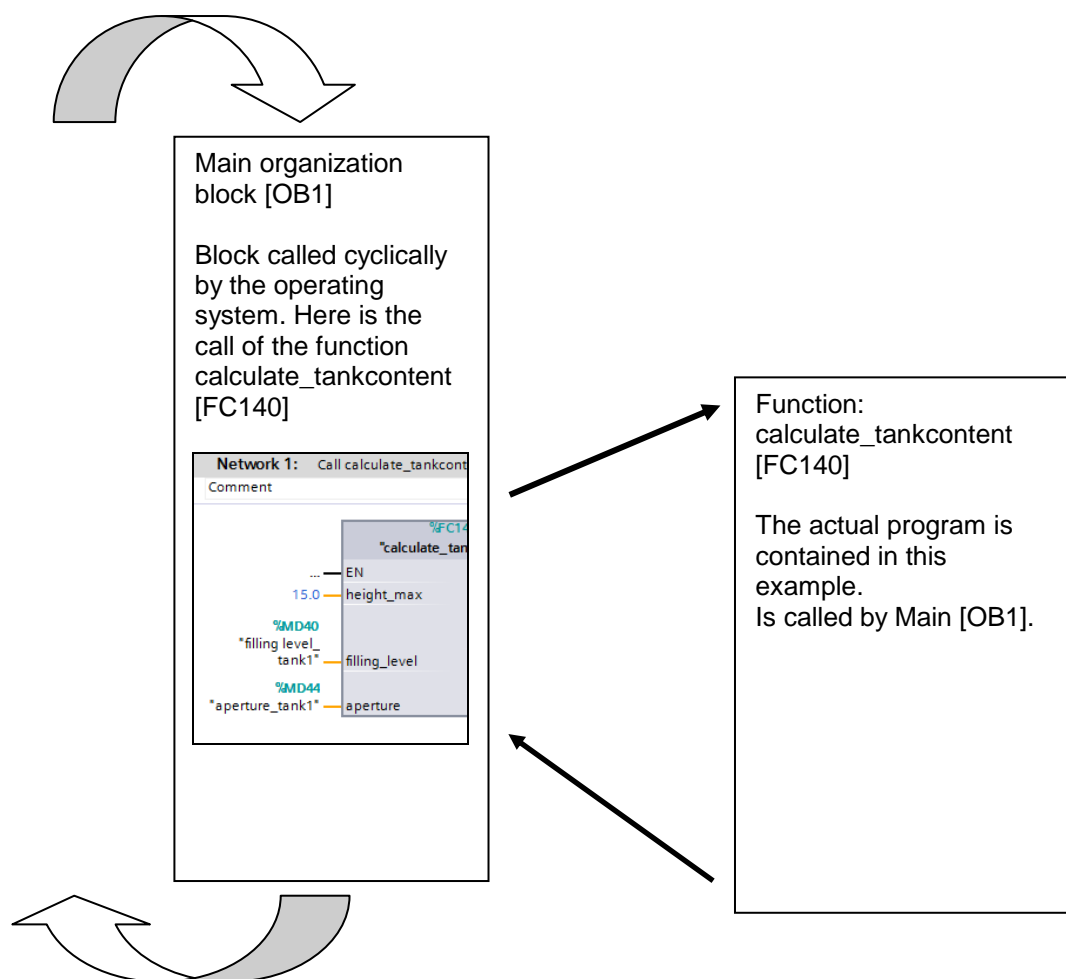
## 4.3 Program structure

Program execution is written in what are referred to as blocks. The Main [OB1] organization block is provided as default. This block represents the interface to the CPU operating system and is automatically called and cyclically processed by this operating system.

From this organization block, additional blocks can be called in turn for structured programming, such as the function calculate\_tankcontent [FC140]

This function is used to break an overall task down into subtasks. These can then be solved more easily and tested in their functionality.

### Structure of the tank content sample task



## 4.4 Interface of the block calculate\_tankcontent [FC140]

The interface of the block must be declared before the program can be written. In the interface declaration, the local tags known only in this block are defined.

The tags or interface parameters are divided into two groups:

- Block parameters that form the block interface for the call in the program.

Type	Designation	Function	Available in
Input parameters	Input	Parameters whose values are read by the block.	Functions, function blocks, and some types of organization blocks
Output parameters	Output / Return	Parameters whose values are written by the block.	Functions and function blocks
In/out parameters	InOut	A parameter whose value is read by the block when it is called and is written back by the block to the same parameter after it is processed.	Functions and function blocks

- Local data that is used for saving intermediate results.

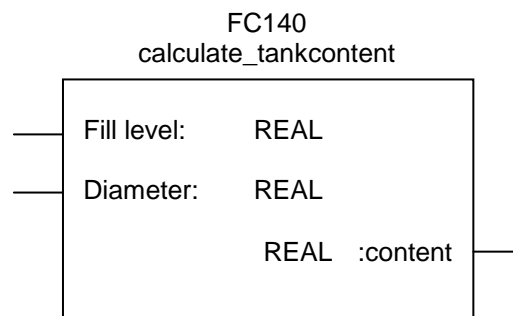
Type	Designation	Function	Available in
Temporary local data	Temp	Tags that are used to store temporary intermediate results. Temporary local data is retained for one cycle only.	Functions, function blocks, and organization blocks
Static local data	Static	Tags that are used for saving static intermediate results in the instance data block. Static data is retained until it is overwritten, which may be after several cycles.	Function blocks

The interface parameters for the 'calculate\_tankcontent [FC140] block used in our sample program are as follows.

Interface of the function FC140:		calculate_tankcontent	
Type	Name	Data type	Comment
IN	Fill level	REAL	in meters
IN	Diameter	REAL	in meters
OUT	Content	REAL	in liters

The function call in a block programmed in FBD appears as follows.

### Function call: Representation in FBD



### 4.5 Note on solution

The formula for calculating the volume of an upright cylinder is used to solve the task. The conversion factor 1000 is used to calculate the result in liters.

$$V = \frac{d^2}{4} \cdot \pi \cdot h \quad \Rightarrow \quad \text{content} = \frac{\text{diameter}^2}{4} \cdot 3.14159 \cdot \text{fill level} \cdot 1000$$

## 5. Programming the Tank Content Calculation for SIMATIC S7-300 in S7-SCL

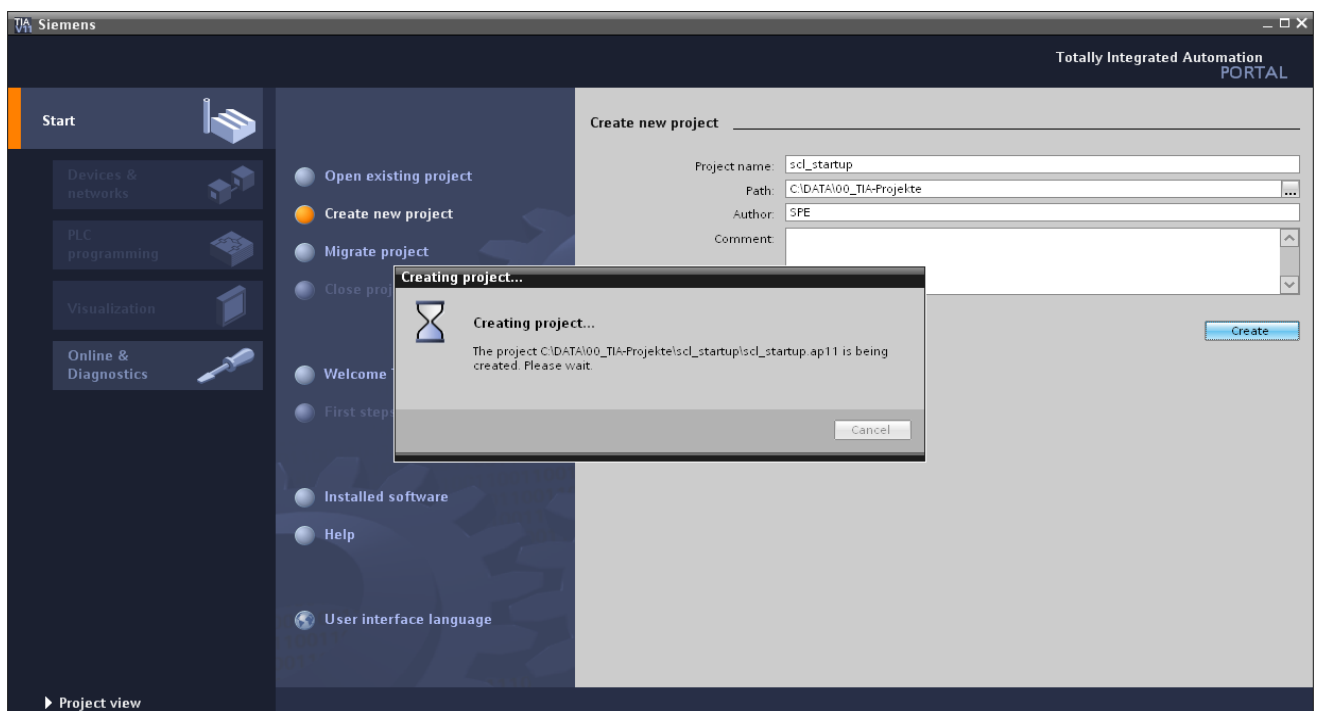
The following steps enable you to create a project for SIMATIC S7-300 and to program the solution for the task:

### 5.1 Creating a project and configuring hardware

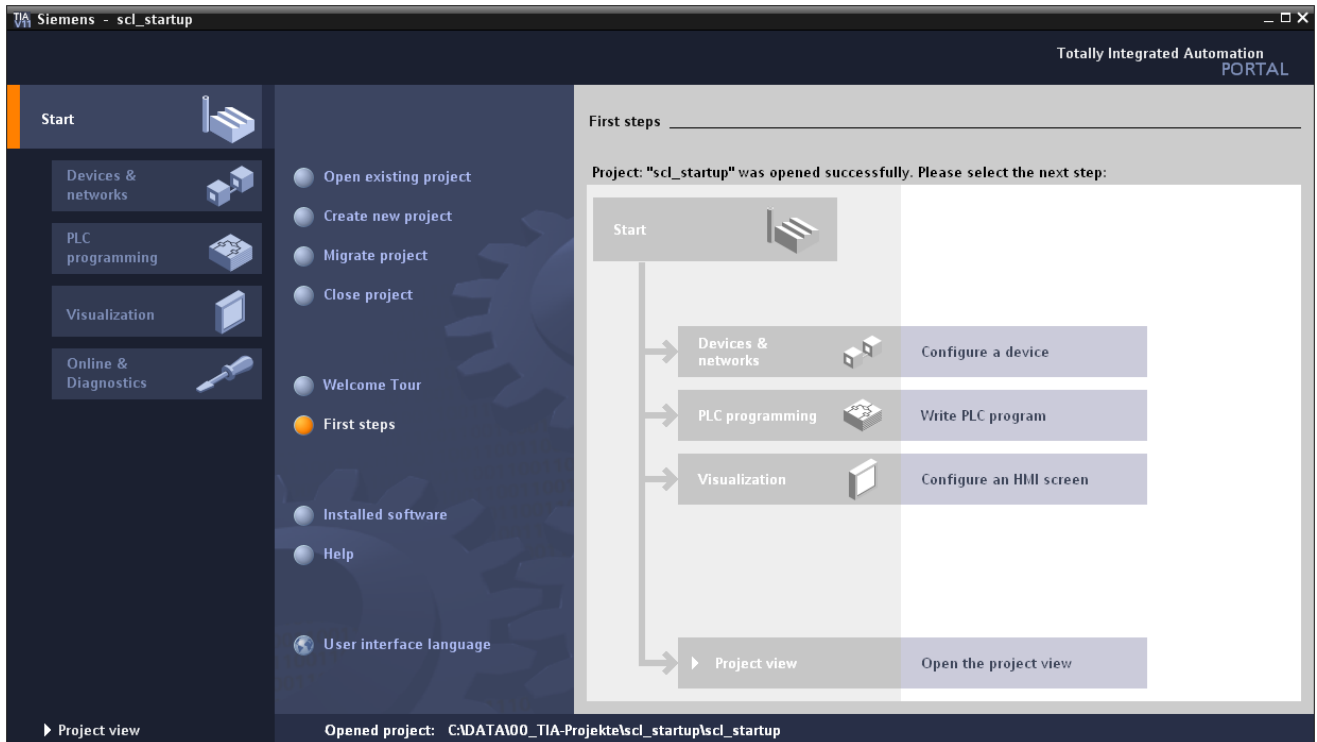
1. The central tool is the **'Totally Integrated Automation Portal'**, which is opened here with a double-click. (→ TIA Portal V11)



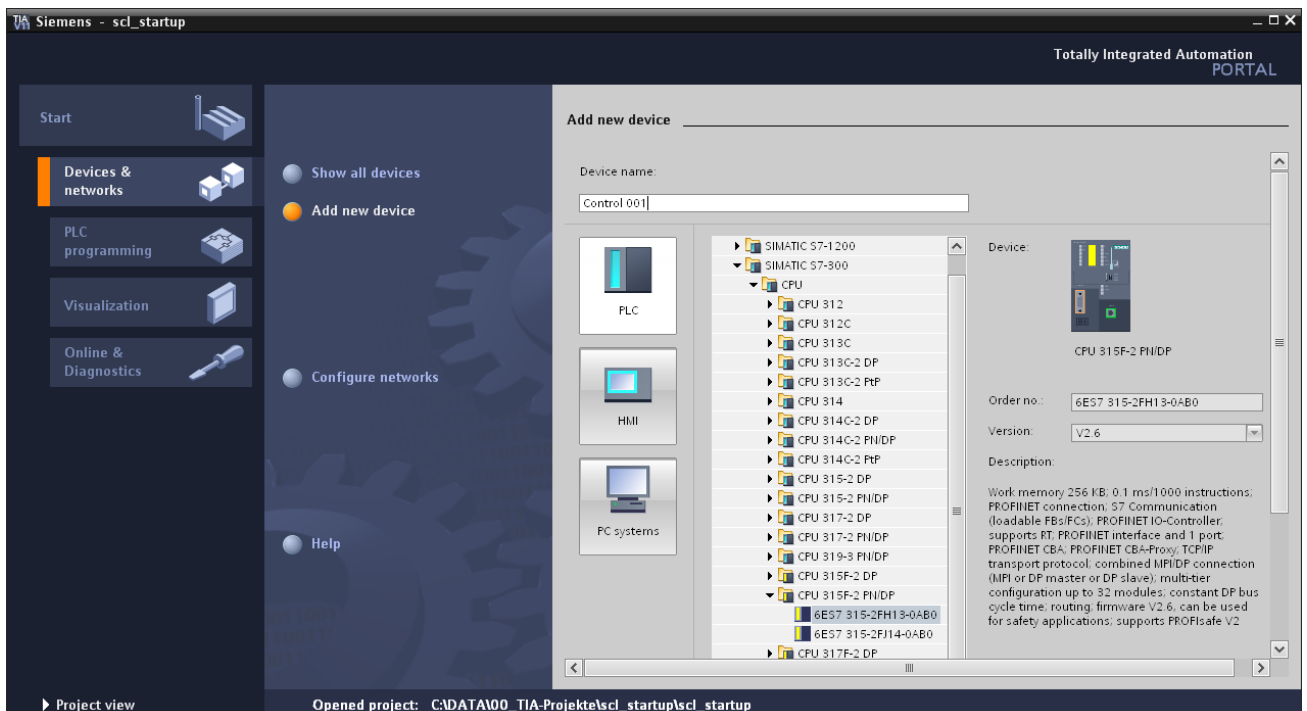
2. Programs for SIMATIC S7-300 are managed in projects. Such a project is now created in the portal view. (→ Create new project → scl\_startup → Create)



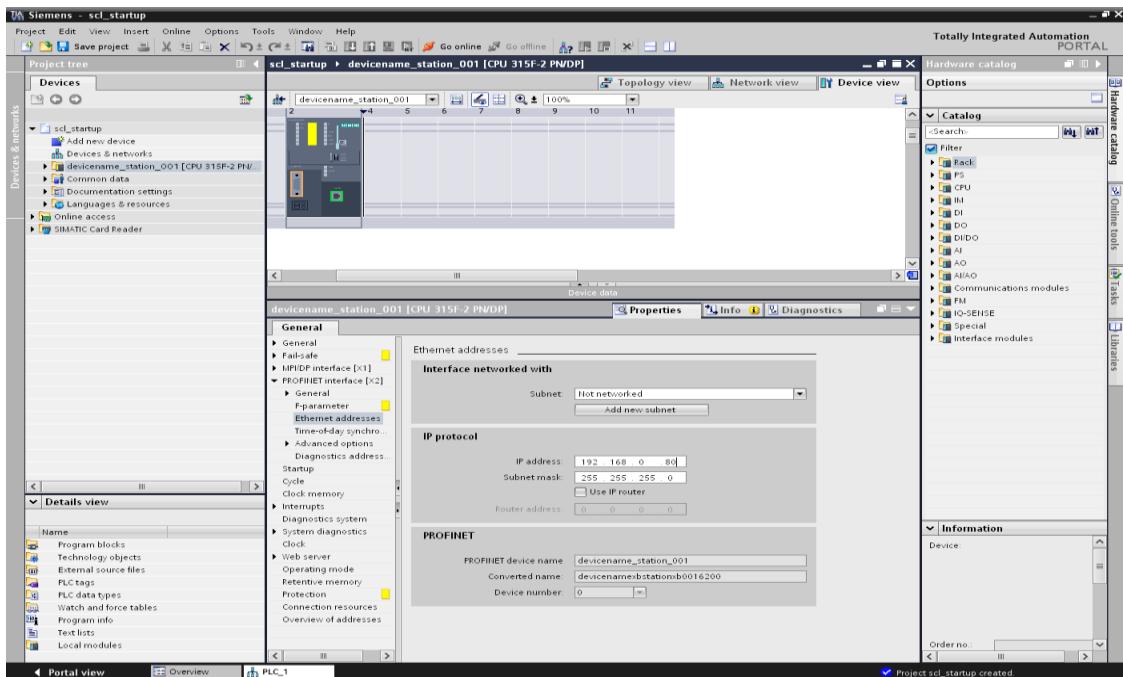
- 'First steps' for configuring are suggested. We want to start with 'Configure a device'.  
(→ First steps → Configure a device)



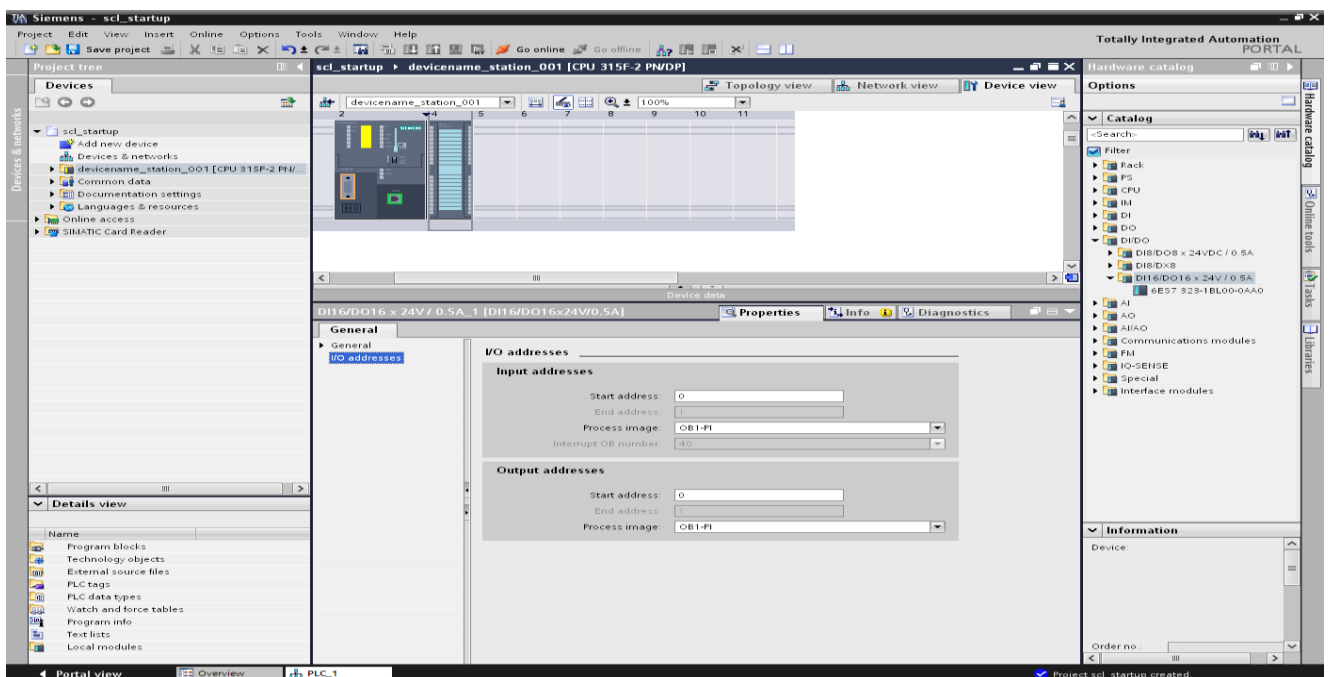
- The next step is 'Add new device' with the 'Device name Controller 001'. Choose the 'CPU 315F-2 PN/DP' with the appropriate order number from the catalog. (→ Add new device → Controller 001 → PLC → SIMATIC S7-300 → CPU → CPU 315F-2 PN/DP → 6ES7 315-2FH13-0AB0 → V2.6 → add)



- The software now switches automatically to the project view containing the opened hardware configuration in the device view. To ensure that the software will access the correct CPU later, the **'ETHERNET address'** of the CPU has to be set.  
 (→ Properties → General → ETHERNET address → IP address: 192.168.0.80 → Subnet mask: 255.255.255.0)



- Additional modules can be added for the hardware catalog (on the right). We select the signal module **'DI16/DO16'** with 16 digital inputs and 16 digital outputs and drag this to slot 4.  
 (→ Hardware catalog → DI/DO → DI16/DO16 x 24V / 0.5A → 6ES7 323-1BL00-0AA0)

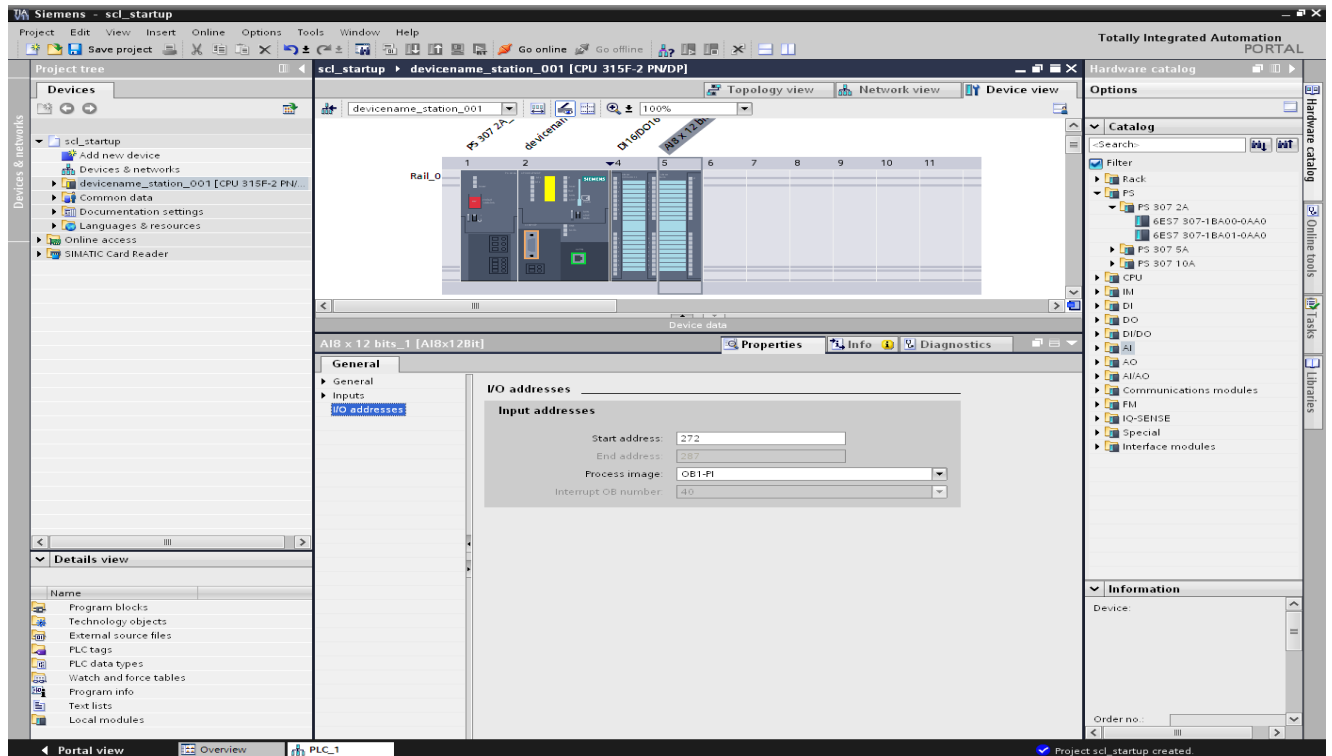


- We also select the '**AI8x12Bit**' signal module with 8 analog input and the '**PS307 2A**' power supply module and drag these to the respective slots 5 and 1.

(→ Hardware catalog

→ AI → AI8 x 12Bit → 6ES7 331-7KF02-0AB0

→ PS → PS 307 2A → 6ES7 307-1BA00-0AA0)



## 5.2 Creating a program

1. Open the **Default tag table** and enter the **'operands'** with their name and data type.  
 (→ Controller 001[CPU315F-2 PN/DP] → PLC-tags→ Default tag table → Enter operands)

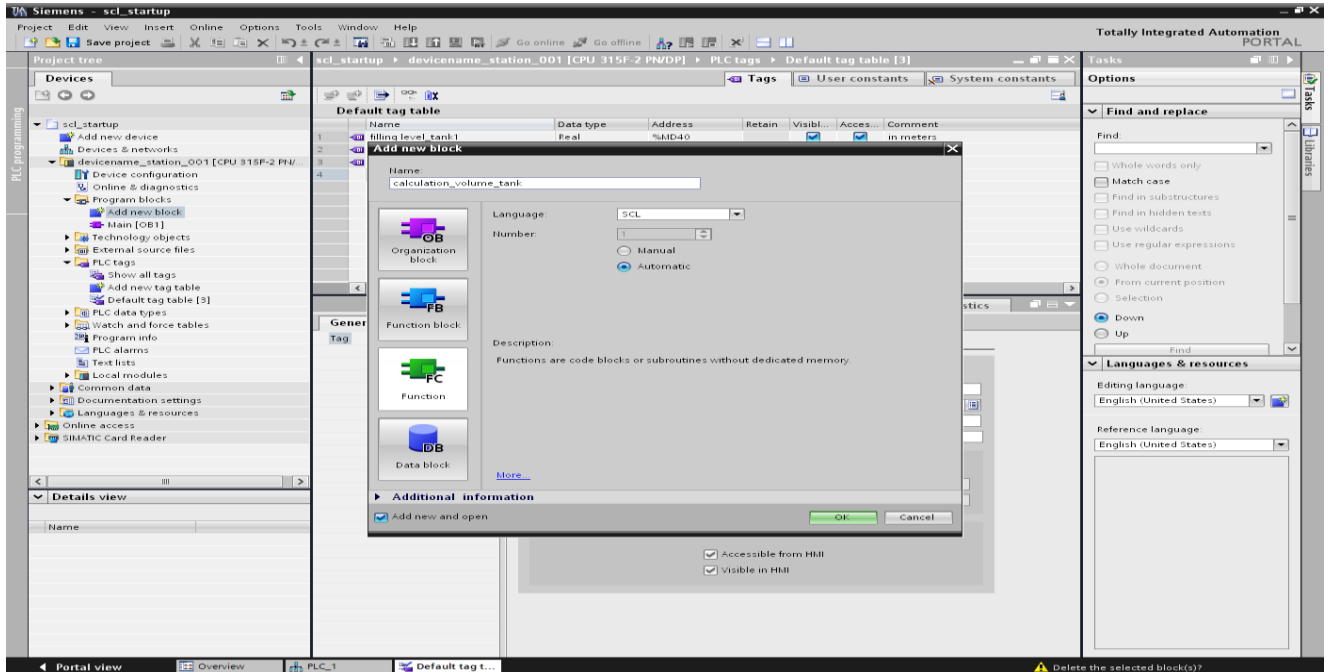
The screenshot shows the Siemens TIA Portal interface. The main window displays the 'Default tag table' configuration for a PLC. The table contains the following data:

	Name	Data type	Address	Retain	Visibl...	Acces...	Comment
1	filling_level_tank1	Real	%MD40		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	in meters
2	aperture_tank1	Real	%MD44		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	in meters
3	volume_tank1	Real	%MD48		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	in liters
4	<Add new>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	



- To create the function calculate\_tankcontent [FC140], select '**Controller 001 [CPU 315F-2 PN/DP]**' in the project tree and then '**Program blocks**'. Then, double-click '**Add new block**'. Select '**Function (FC)**' and assign the name '**calculate\_tankcontent**'. Change the programming language to '**SCL**'. You can change the numbering by switching from automatic to manual. Enter the number 140. Click '**OK**' to accept your entries.

(→ Controller 001 [CPU 315F-2 PN/DP] → Program blocks → Add new block → Function (FC) → calculate\_tankcontent → SCL → manual → 140 → OK)



- The 'calculate\_tankcontent [FC140]' block opens automatically. Enter the **'Input and output parameters'** of the block as specified. All local tags should also be provided with a **'sufficiently descriptive comment'** for better understanding.  
(→ Enlarge interface of block FC140 → Enter interface parameters)

Diagram 0-1

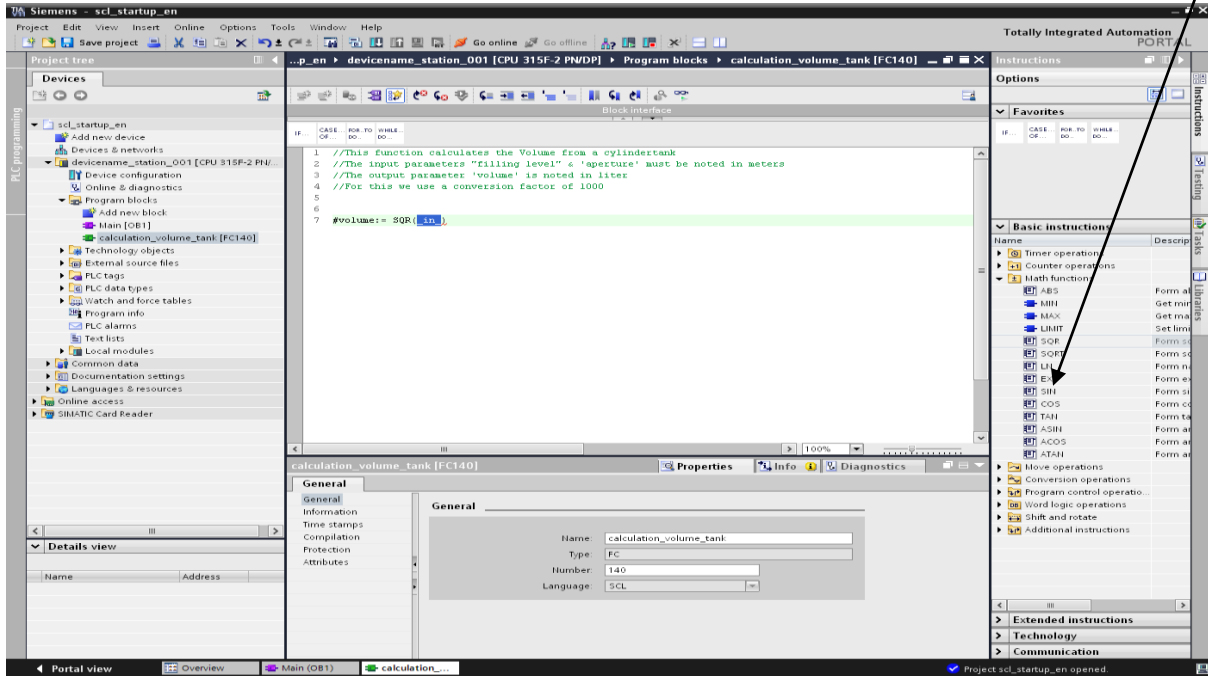
The screenshot shows the Siemens TIA Portal interface for the 'calculate\_volume\_tank [FC1]' block. The 'Interface' table is expanded to show the following parameters:

	Name	Data type	Offset	Comment
1	Input			
2	filling level	Real		in meters
3	aperture	Real		in meter
4	<Add new>			
5	Output			
6	volume	Bool		in liters when failure =-1
7	<Add new>			
8	InOut			
9	<Add new>			

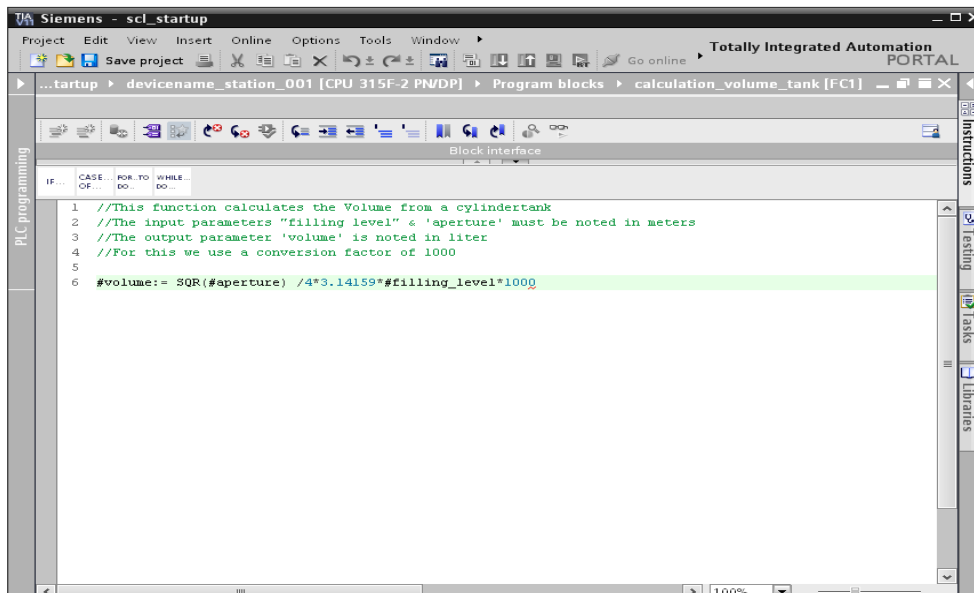
**Note:**

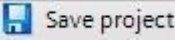

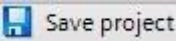

To avoid confusion with the PLC tags it is helpful to write the local tags with small letters.

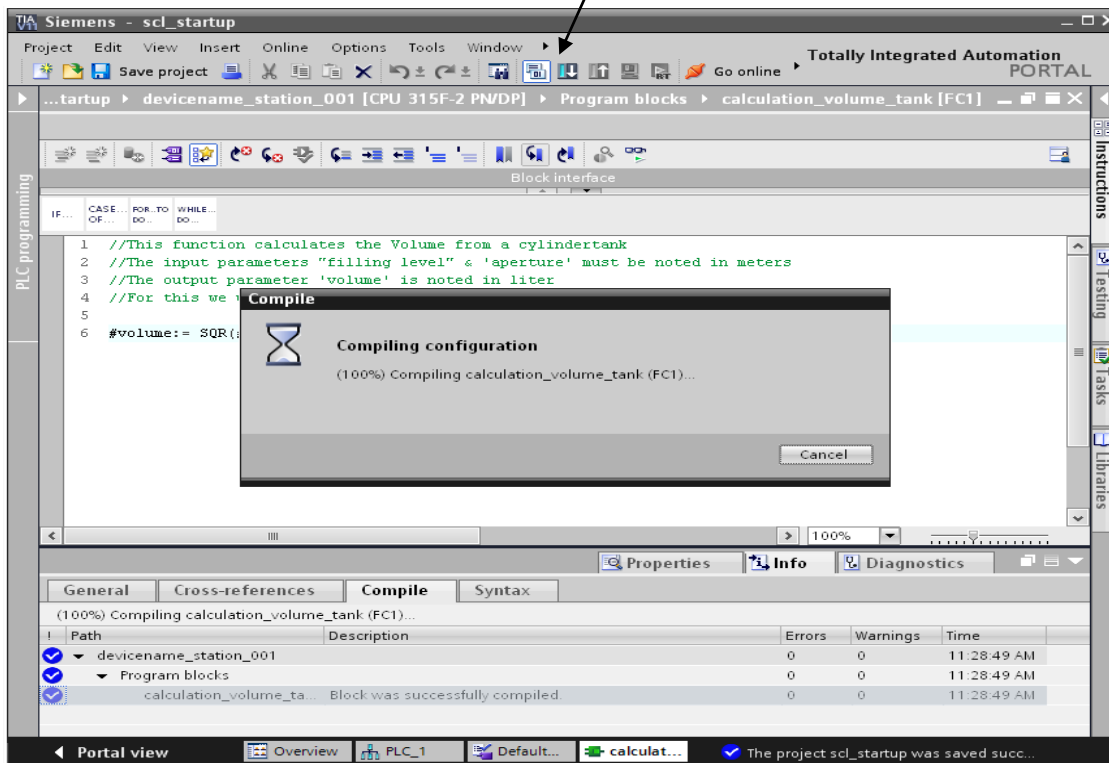
- Once the local tags have been declared, you can start to create the program shown here. The square function 'SQR' is used to square a number for this purpose. The number to be squared is enclosed in brackets. To insert the function, drag it to the position of use in the program.  
 (→ Instructions (menu on the right) → Basic instructions → Mathematical functions → SQR)



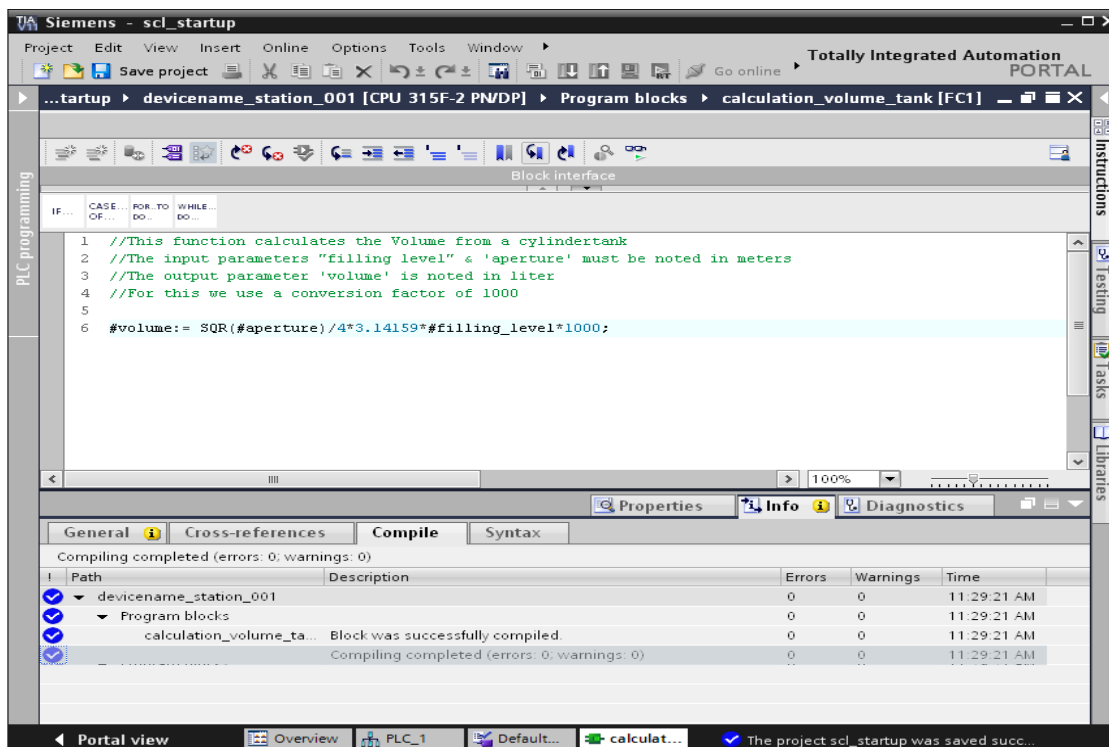
- Supplement the program as specified below.



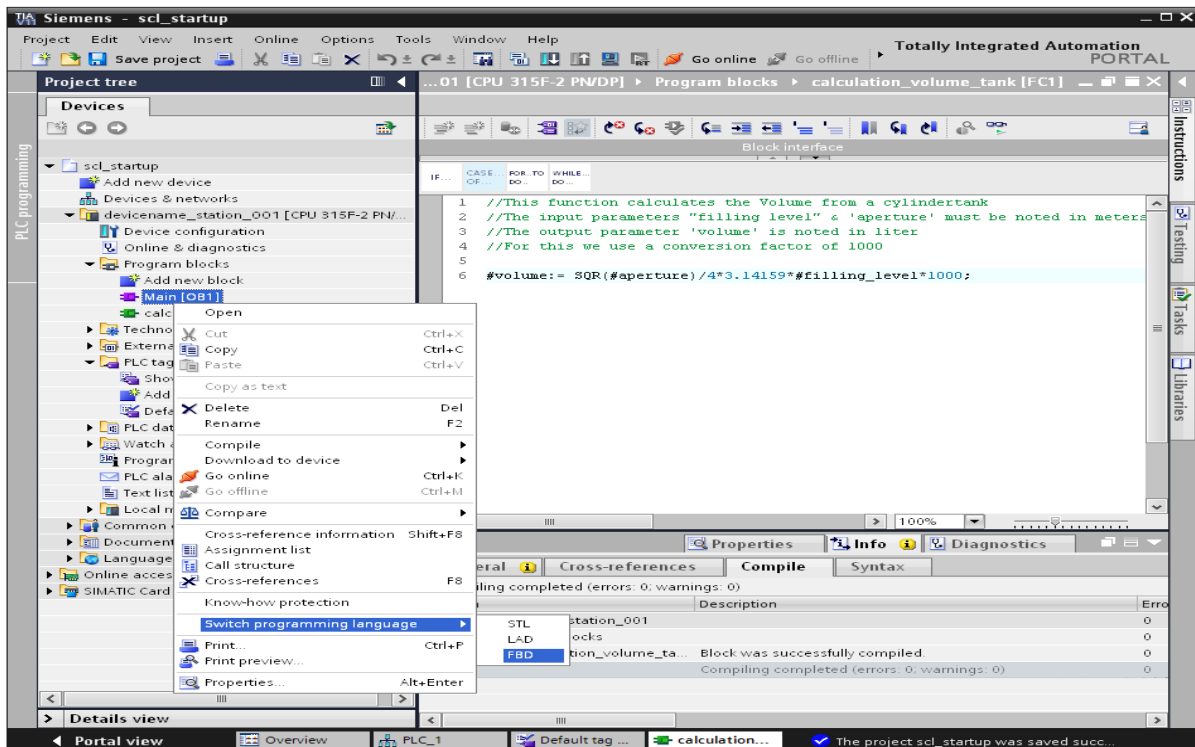
6. The program can now be saved  and compiled .
- (→ Save  → Compile )



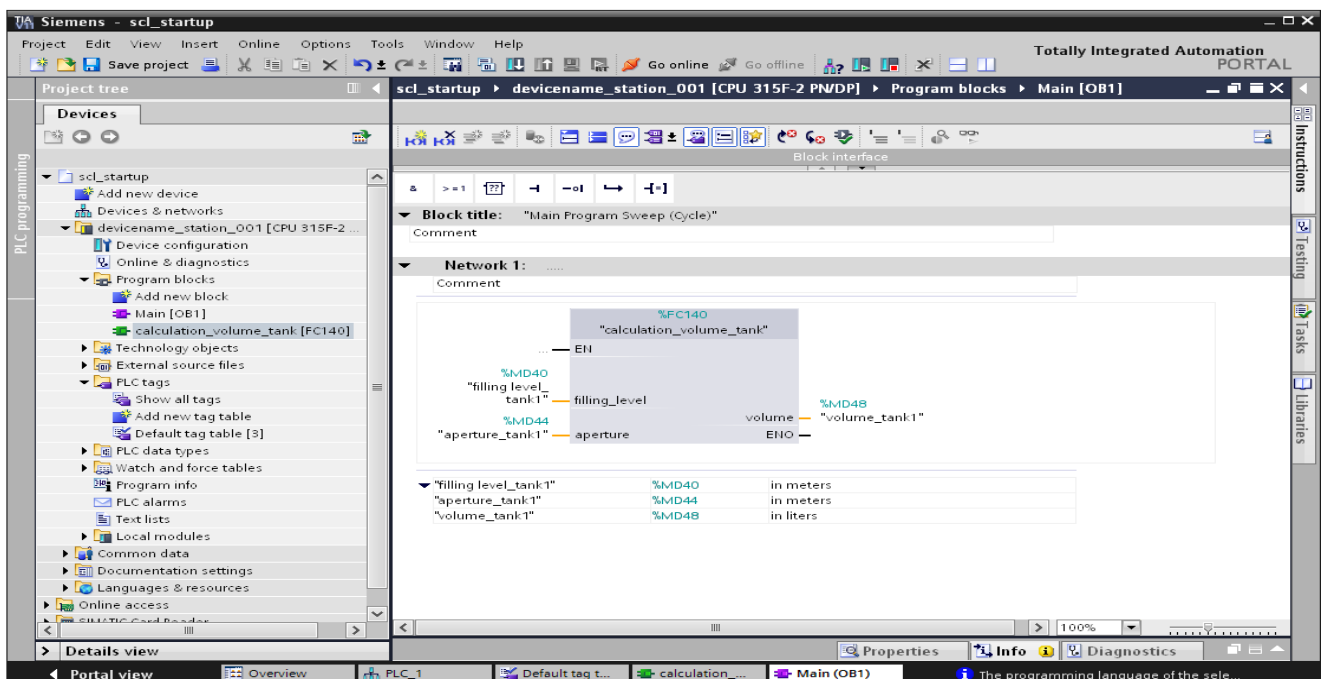
7. Syntax errors that occur are recognized during compilation and display in the 'Menu Info/Compile'.
- (→ Info → Compile)





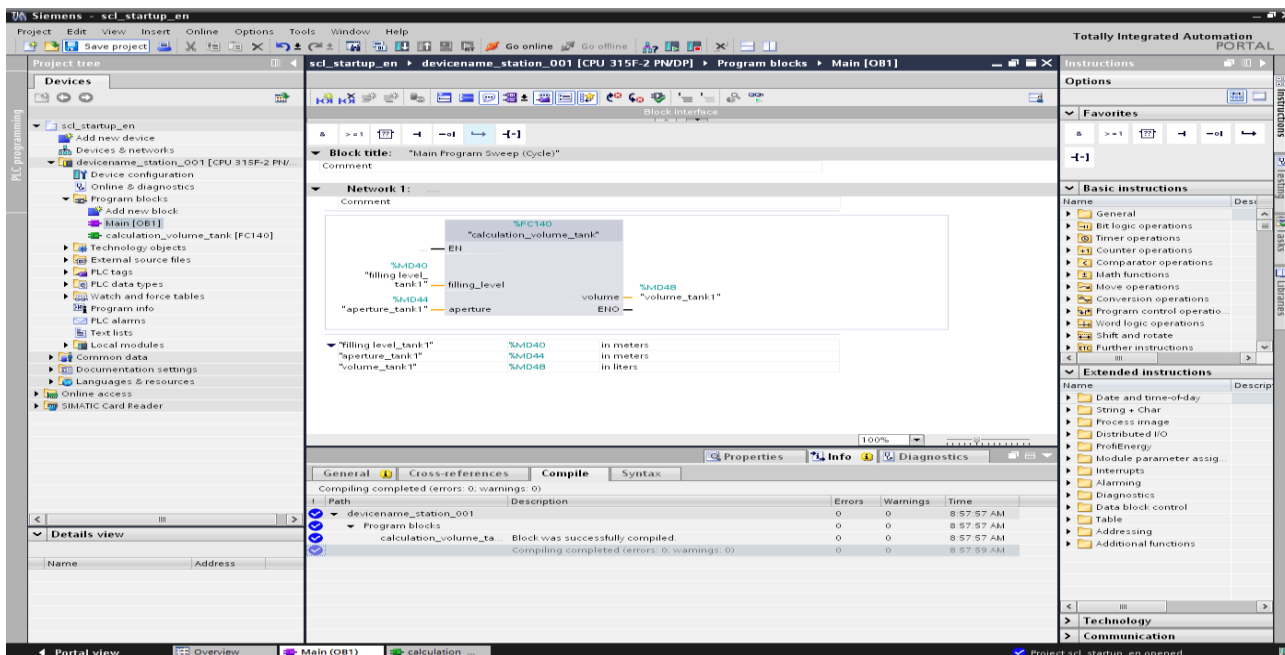
- The function can now be called in the 'Main [OB1]'. Before we double-click to open the 'Main [OB1]' block, we select 'FBP' as its programming language. (→ Main [OB1] → Switch programming language → FBD)





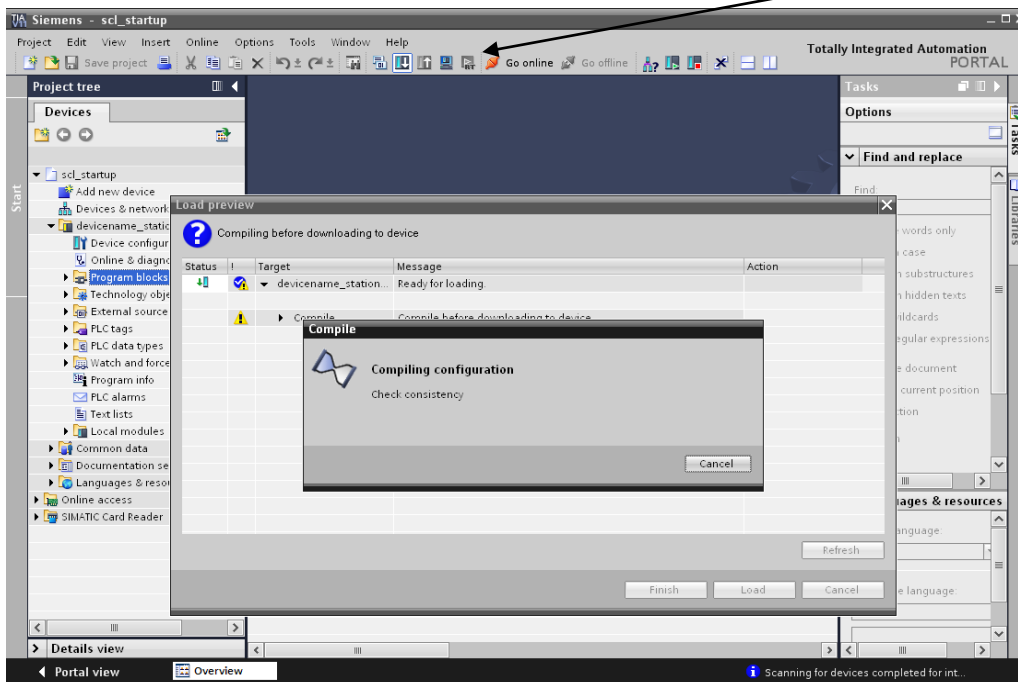
- The 'calculate\_tankcontent [FC140]' function can then be moved into Network 1 of the Main [OB1] block using a drag-and-drop operation. The interface parameters of the 'calculate\_tankcontent [FC140]' function have to be connected to the global PLC tags, as shown here. Don't forget to document the networks in the Main [OB1] block. (→ Main [OB1] → Program blocks → calculate\_content [FC140])



10. The button  Save project is used to resave the project.  
 (→ )

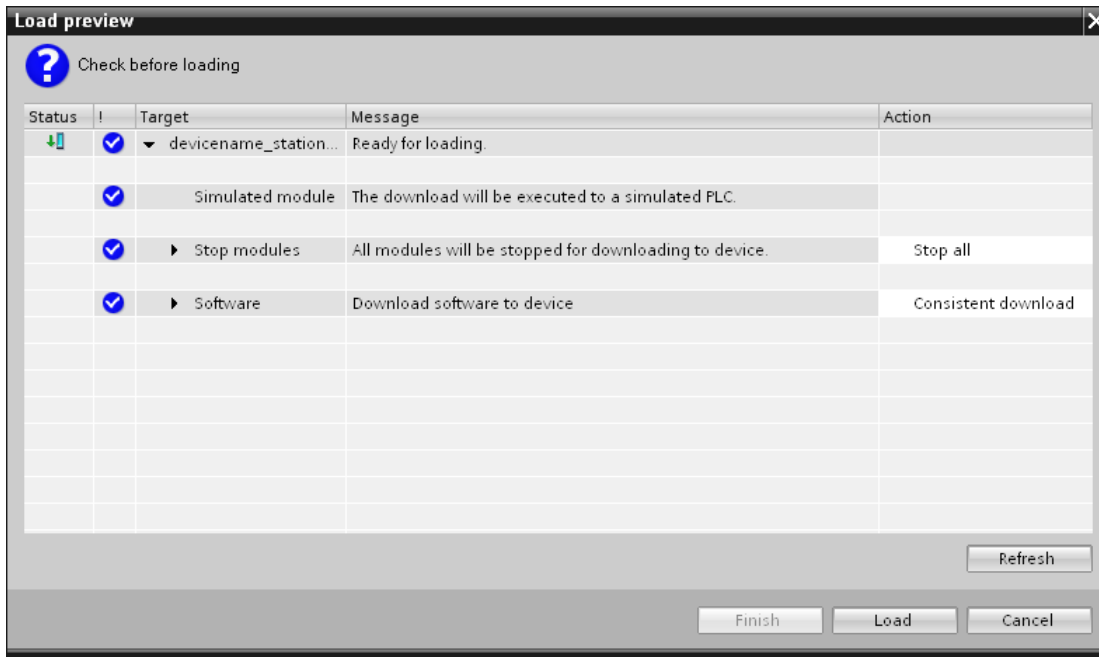


11. To load the program blocks and the device configuration to the CPU, first select the '**Controller 001 [CPU 315F-2 PN/DP]**' folder and then click the Download to device icon . (→ Program blocks → )





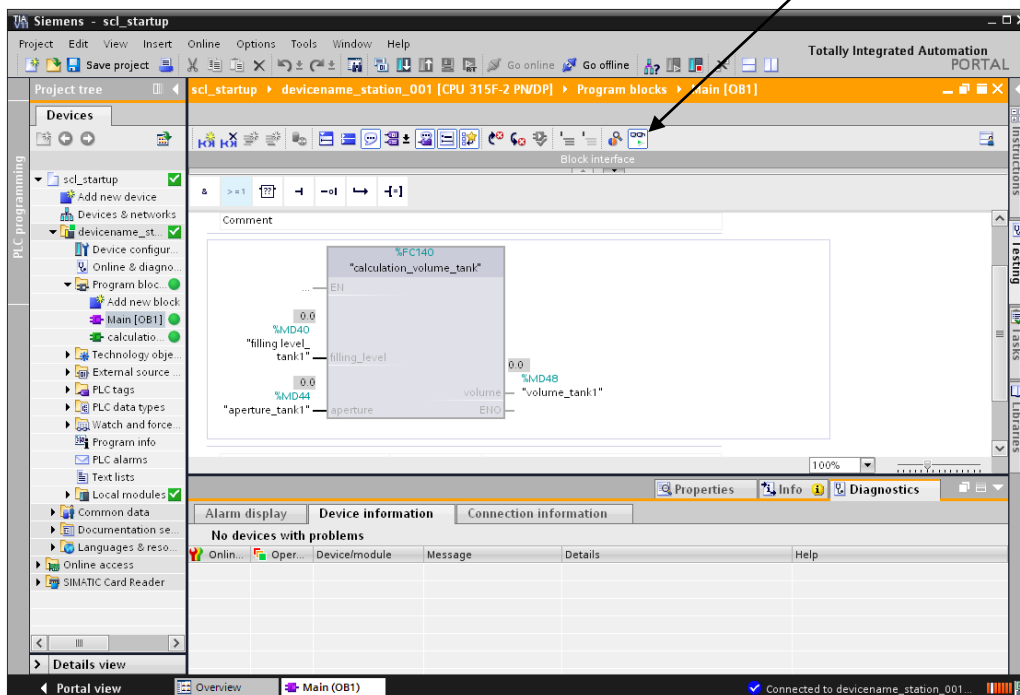
**Note:**  
 During downloading the project is automatically recompiled and checked for errors.

- An overview for checking the steps to be executed is shown before the download process starts. Click **'Load'** to start loading the program. (→ Load)

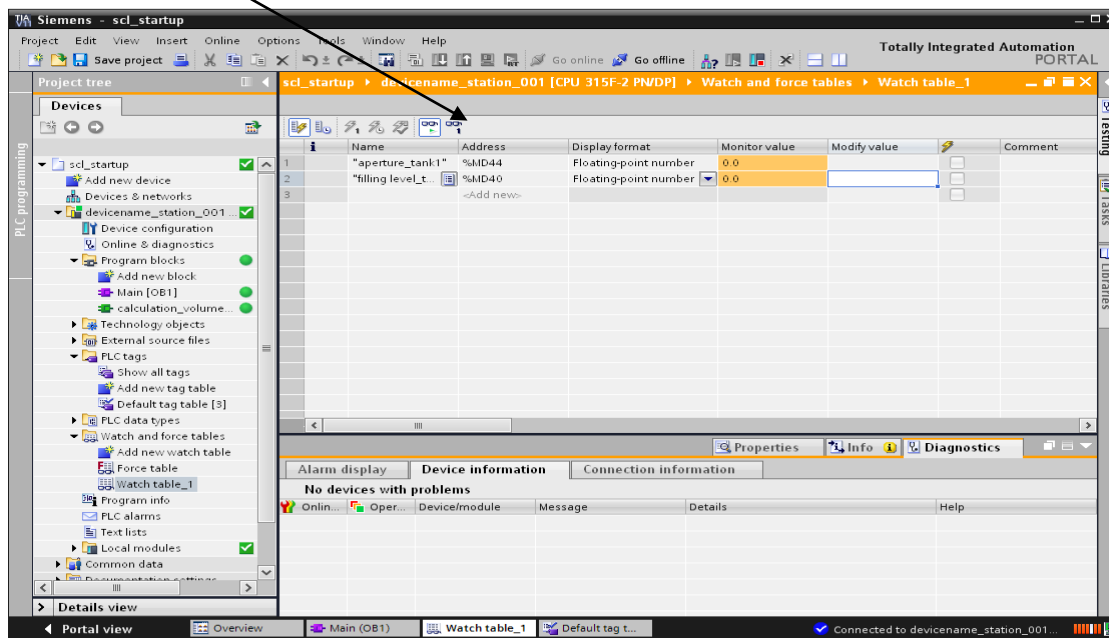


### 5.3 Test program

- Clicking the  icon (Monitoring on/off) allows you to monitor the state of the simulated input and output tags on the **'calculate\_tankcontent'** block. (→ )

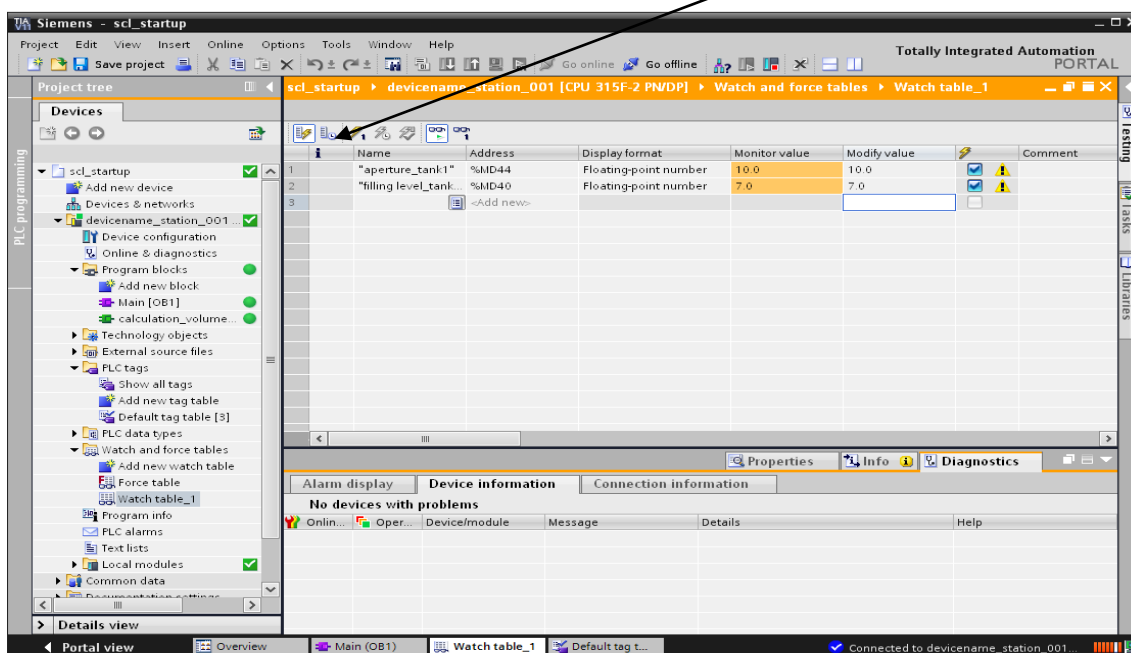


- Since no analog sensor exists and thus no corresponding process value is available, we have to specify the values 'Diameter\_tank1' and 'Fill-level\_tank1' using the watch table. Create a 'new watch table' and enter the two values. To see the current values, switch 'Monitoring mode on'. (→ Controller 001 → Watch and force table → New watch table → Diameter\_tank1, Fill-level\_tank1



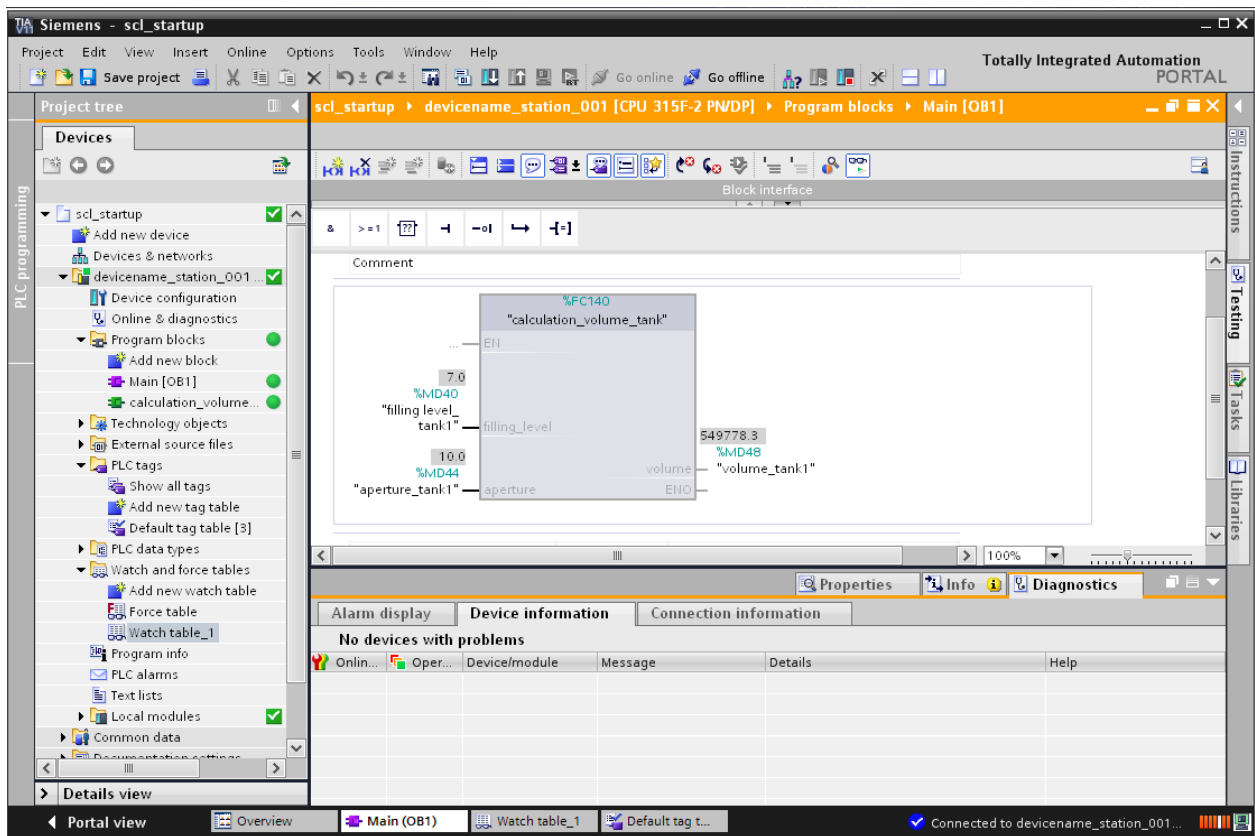
- To specify the values, you have to enter a modify value in the 'Modify value' column. The values are applied in the CPU by clicking the icon 'Modify all selected values once and now'.


(→ diameter\_tank1 = 10.0 → fill level\_tank1 = 7.0 → )

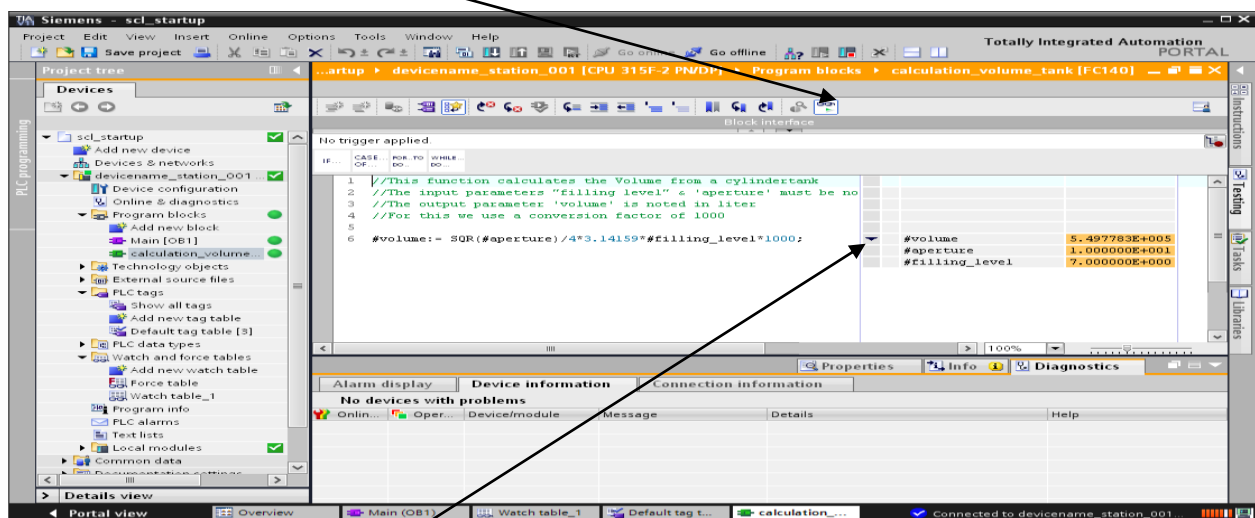





4. The program in OB1 can now be checked. (→ )



5. The values of the individual tags can be monitored in the SCL editor. To do this, switch 'Monitoring mode on'. (→ )



**Note:**

When you click  the current values of the tags programmed in this row are displayed.

## 5.4 Expanding the program

Next, we will check whether faulty information is contained in the input parameters of the **'calculate\_tankcontent'** block. Another value **'high\_max'** is also passed to the block. This indicates the height of the tank.

The block is now to analyze whether the fill-level of the tank is less than zero or greater than the specified tank height. It is also to be checked whether the diameter was specified as less than zero. If there is an error, the Boolean output parameter **'er'** should return TRUE and the value of the **'Content'** parameter should be -1.

### Expanding the assignment list / tag table:

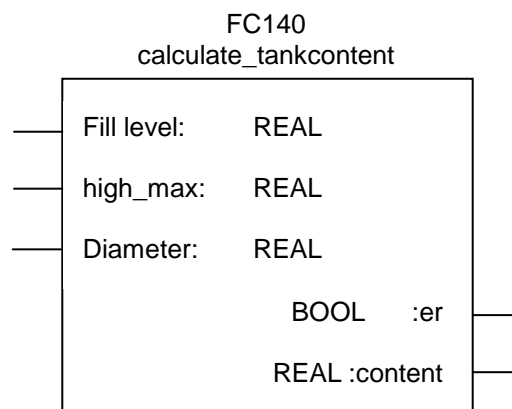
Address	Symbol	Data type	Comment
%Q1.7	Error bit	BOOL	Error, unable to perform calculation

### Expansion of the interface of the function FC140:

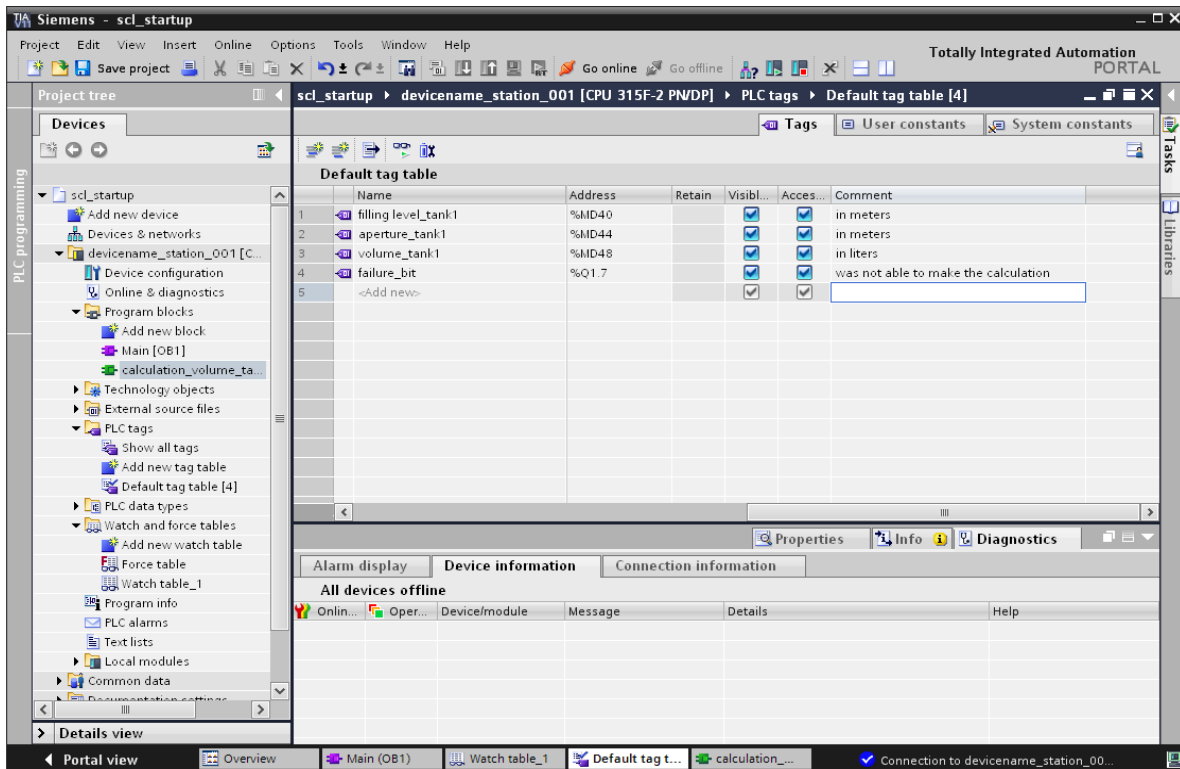
### calculate\_tankcontent

Type	Symbol	Data type	Comment
IN	high_max	REAL	in meters
OUT	er	REAL	er = 1, there is an error, content = -1

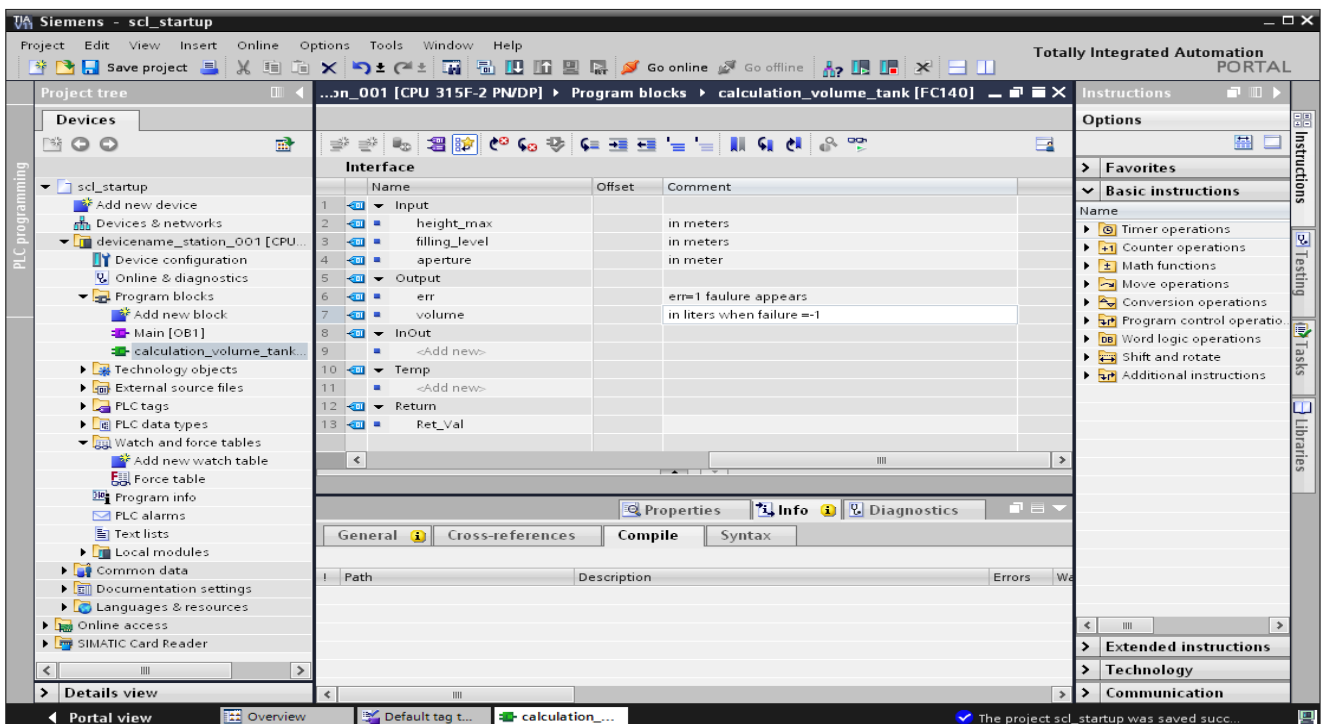
### Function call: Representation in FBD



1. Expand the 'Tag table' of the block as specified above.  
 (→ Controller 001[CPU 315F-2 PN/DP] → PLC- tags → Default tag table → Enter operands)

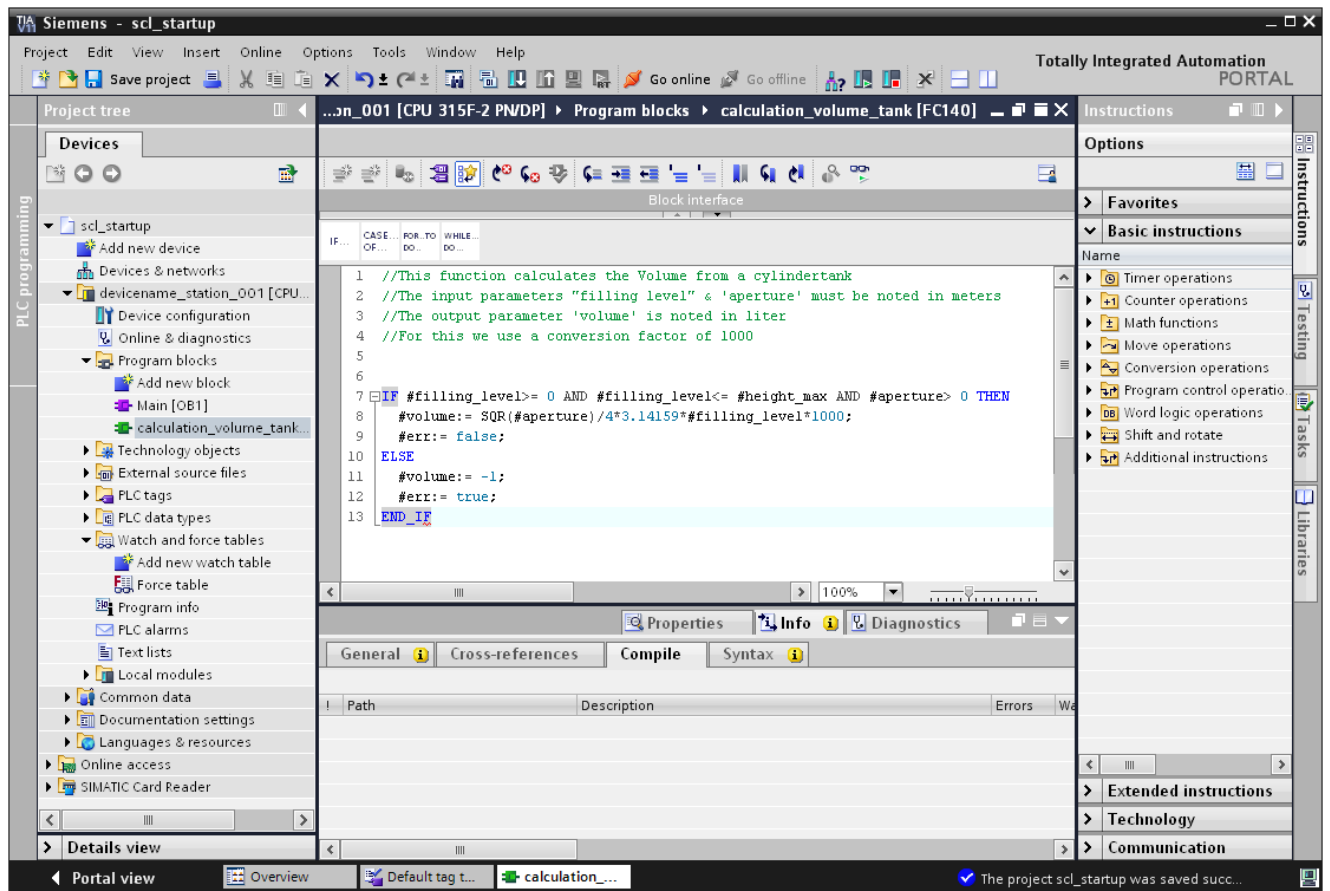


2. Expand the 'Interface parameters' of the block as specified above.  
 (→ Controller 001 [CPU 315F-2 PN/DP] → Program blocks → calculate\_tankcontent) → Enter parameters)

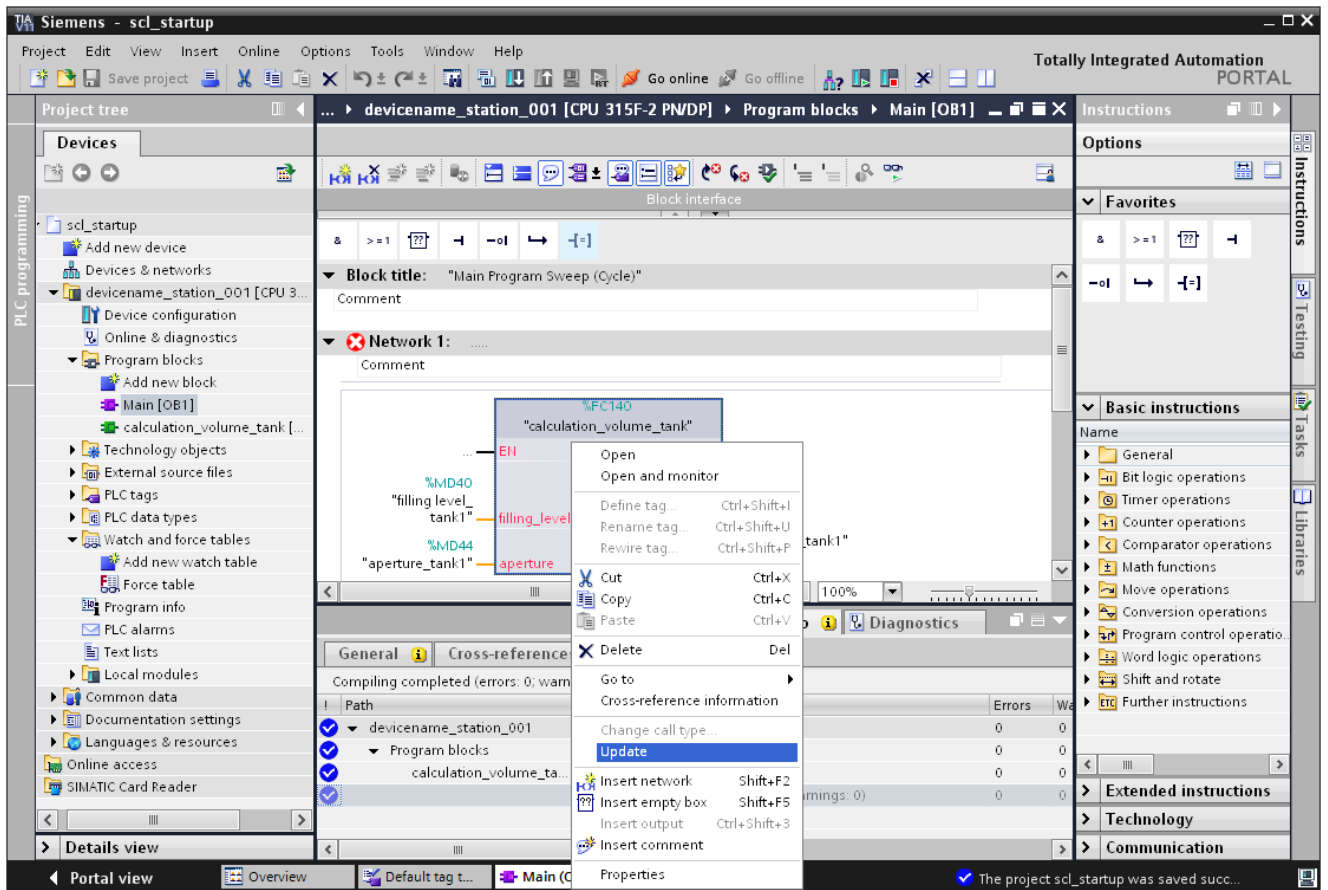


- Next, expand the program as specified below and check it for syntax errors by compiling it. Save the program and load it to the controller.

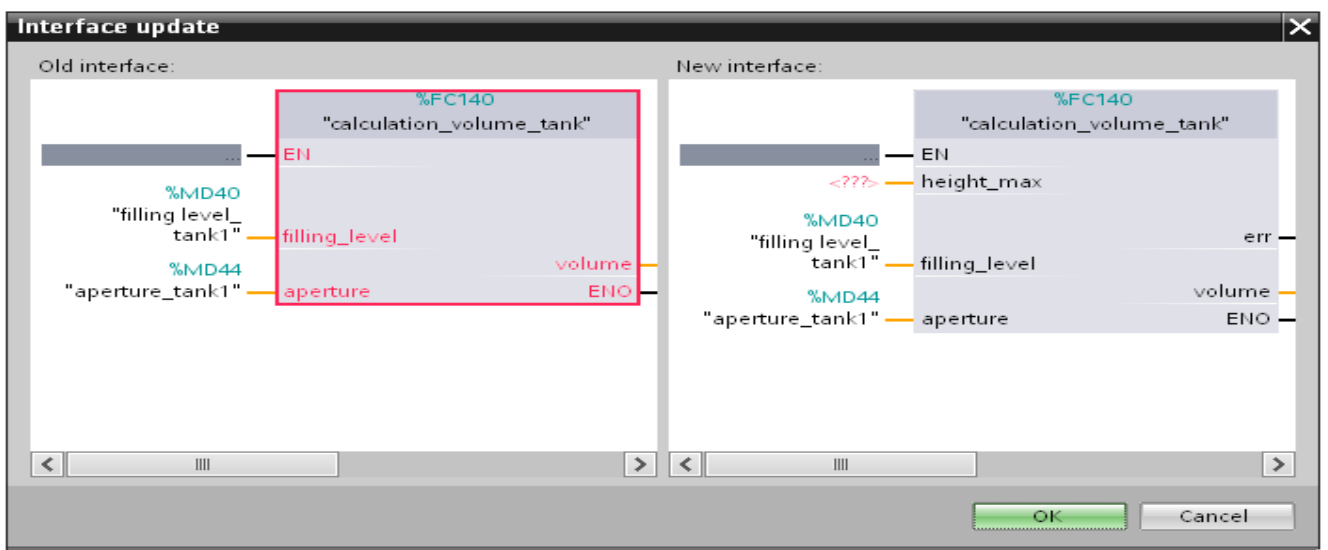
(Write program →  →  Save project → )



- Because the parameters of the block were changed, the call must be updated in OB1. Open OB1 and scroll to the position of the block call. Right-click to open the shortcut menu and then click **'Update'**.  
 (→ Controller 001 [CPU 315F-2 PN/DP] → Program blocks → Main [OB1]  
 → Right mouse button → Update)

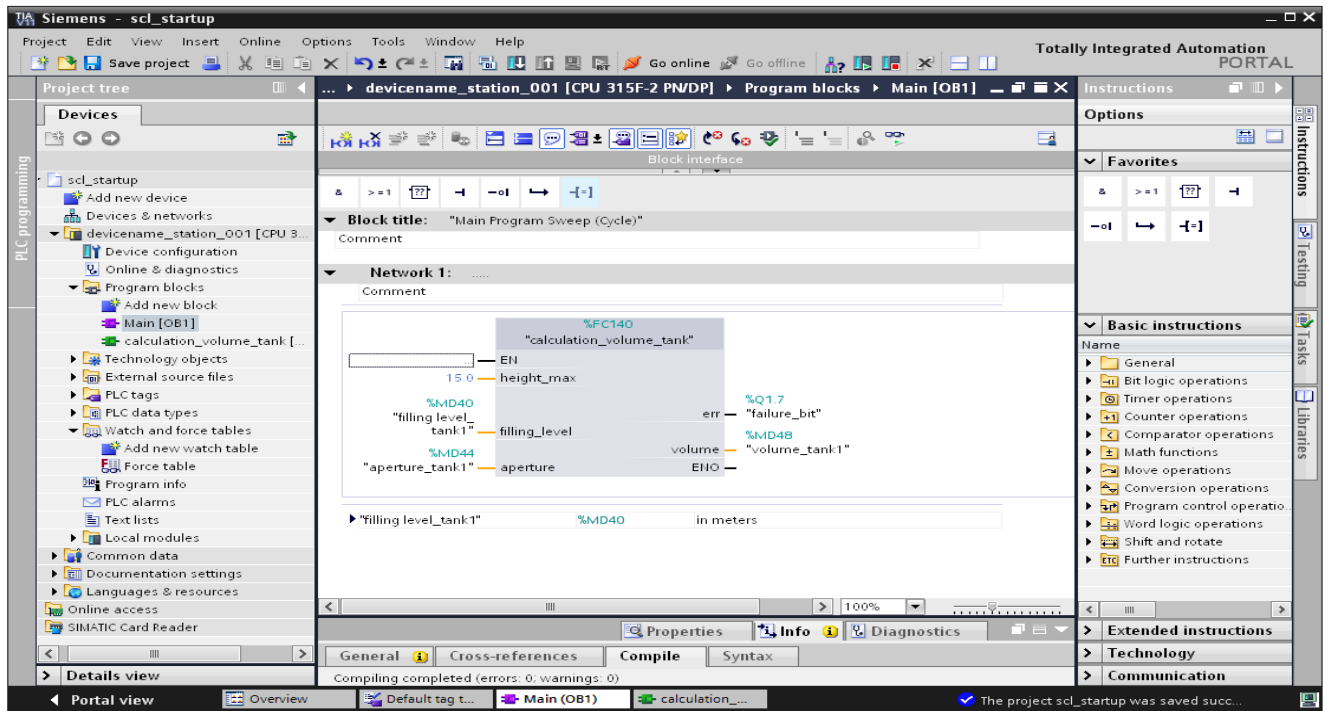


- The old and the new interface will now be displayed. Click **'OK'** to confirm.  
 (→ OK)

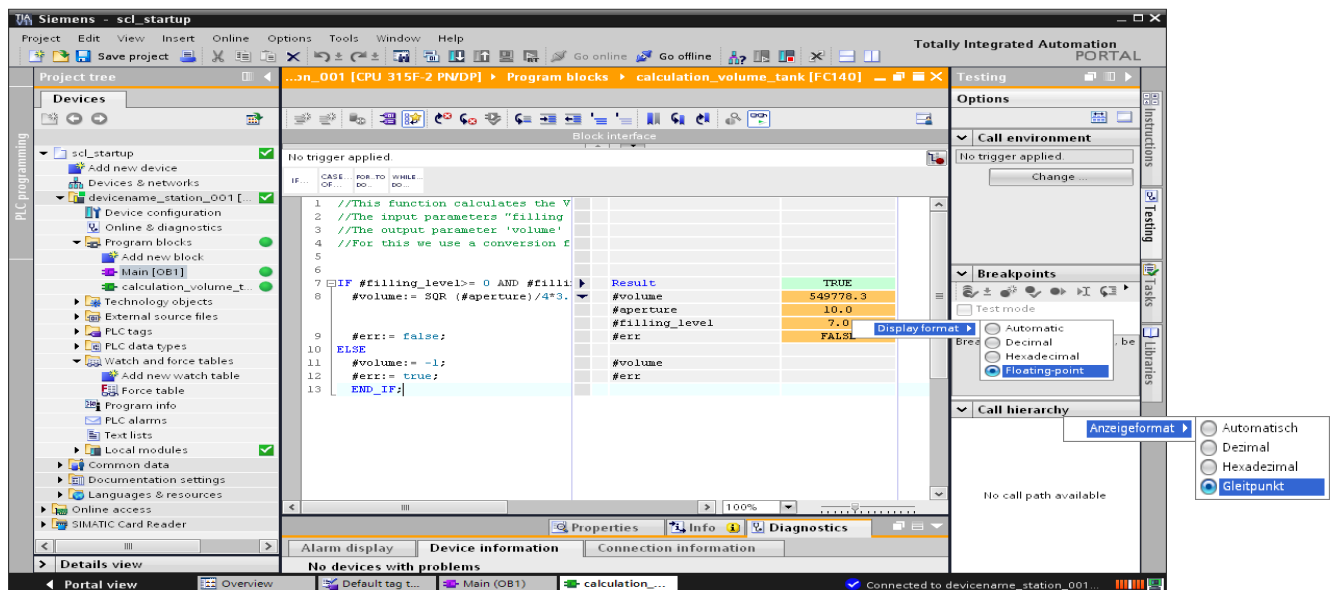


- Complete the values at the input parameter 'high\_max' and output parameter 'er' as specified below. Compile, save and load the program to the controller.

(Add other parameters →  → Save project →  → )



- Check the changes in the 'Monitoring mode' of the 'calculate\_tankcontent' block.



**Note:**

You can change the display format of the current value by right-clicking the value and opening the shortcut menu.