**Matching SCE Trainer Packages for these Learn-/Training Document**

Learn-/Training Document  
  
Siemens Automation Cooperates with Education (SCE) | From Version V14 SP1

**siemens.com/sce**

TIA Portal Module 031-500

Analog Values for SIMATIC S7-1200

* **SIMATIC S7-1200 AC/DC/RELAY (set of 6) "TIA Portal"**  
  Order no.: 6ES7214-1BE30-4AB3
* **SIMATIC S7-1200 DC/DC/DC (set of 6) "TIA Portal"**  
  Order no.: 6ES7214-1AE30-4AB3
* **Upgrade SIMATIC STEP 7 BASIC V14 SP1 (for S7-1200) (set of 6) "TIA Portal"**  
  Order no.: 6ES7822-0AA04-4YE5

Please note that these trainer packages are replaced with successor packages when necessary.

An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](http://www.siemens.com/sce/tp)

**Continued training**

For regional Siemens SCE continued training, please contact your regional SCE contact [siemens.com/sce/contact](http://www.siemens.com/contact)

**Additional information regarding SCE**

[siemens.com/sce](http://www.siemens.com/sce)  
  
  
**Information regarding use**

The SCE Learn-/Training Document for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational facilities and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems, which means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this Learn-/Training Document and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact person: Roland Scheuerer roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is explicitly not permitted. We do not consent to commercial use of the Learn-/Training Document.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this Learn-/Training Document.

Table of contents

[1 Goal 4](#_Toc489014114)

[2 Prerequisite 4](#_Toc489014115)

[3 Required hardware and software 5](#_Toc489014116)

[4 Theory 6](#_Toc489014117)

[4.1 Analog signals 6](#_Toc489014118)

[4.2 Measuring transducers 7](#_Toc489014119)

[4.3 Analog modules – A/D converter 7](#_Toc489014120)

[4.4 Data types of the SIMATIC S7-1200 8](#_Toc489014121)

[4.5 Reading/outputting analog values 9](#_Toc489014122)

[4.6 Normalizing analog values 11](#_Toc489014123)

[5 Task 12](#_Toc489014124)

[6 Planning 12](#_Toc489014125)

[6.1 Analog control of the conveyor speed 12](#_Toc489014126)

[6.2 Technology diagram 13](#_Toc489014127)

[6.3 Reference list 14](#_Toc489014128)

[7 Structured step-by-step instructions 15](#_Toc489014129)

[7.1 Retrieve an existing project 15](#_Toc489014130)

[7.2 Create the "MOTOR\_SPEEDCONTROL" function 17](#_Toc489014131)

[7.3 Configuration of the analog output channel 24](#_Toc489014132)

[7.4 Expand the tag table to include analog signals 25](#_Toc489014133)

[7.5 Call the block in the organization block 26](#_Toc489014134)

[7.6 Save and compile the program 29](#_Toc489014135)

[7.7 Download the program 30](#_Toc489014136)

[7.8 Monitor program blocks 31](#_Toc489014137)

[7.9 Archive the project 33](#_Toc489014138)

[8 Checklist 34](#_Toc489014139)

[9 Exercise 35](#_Toc489014140)

[9.1 Task – Exercise 35](#_Toc489014141)

[9.2 Technology diagram 36](#_Toc489014142)

[9.3 Reference list 37](#_Toc489014143)

[9.4 Planning 37](#_Toc489014144)

[9.5 Checklist – Exercise 38](#_Toc489014145)

[10 Additional information 39](#_Toc489014146)

Analog Values for SIMATIC S7-1200

# Goal

In this chapter, you will become acquainted with the analog value processing of the SIMATIC S7-1200 with the TIA Portal programming tool.

The module explains the acquisition and processing of analog signals and gives a step-by-step description of read and write access to analog values in the SIMATIC S7-1200.

The SIMATIC S7 controllers listed in Chapter 3 can be used.

# Prerequisite

This chapter builds on the chapter IEC Timers and Counters with the SIMATIC S7 CPU1214C. You can use the following project for this chapter, for example: SCE\_EN\_031-300\_IEC\_Timers\_Counters\_S7-1200.zap14

# Required hardware and software

**1** Engineering station: requirements include hardware and operating system   
(for additional information, see Readme on the TIA Portal Installation DVDs)

**2** SIMATIC STEP 7 Basic software in TIA Portal – as of V14 SP1

**3** SIMATIC S7-1200 controller, e.g. CPU 1214C DC/DC/DC with ANALOG OUTPUT SB1232 signal board, 1 AO – Firmware as of V4.2.1

Note: The digital inputs and analog inputs and outputs should be fed out to a control panel.

**4** Ethernet connection between engineering station and controller



**2** SIMATIC STEP 7 Basic (TIA Portal) as of V14 SP1



**1** Engineering station

**4** Ethernet connection



**3** SIMATIC S7-1200 controller



Control panel

# Theory

## Analog signals

In contrast to a binary signal, which can assume only two signal states (“Voltage present +24 V” and “Voltage not present 0 V”), analog signals can assume any value within a defined range. A typical example of an analog sensor is a potentiometer. Depending on the position of the knob, any resistance can be set, up to the maximum value.

Examples of analog quantities in control engineering:

* + Temperature -50 to +150 °C
  + Flow rate 0 to 200 l/min
  + Speed -500 to +50 rpm
  + etc.

## Measuring transducers

These quantities are converted to electrical voltages, currents or resistances with the help of a measuring transducer. If, for example, a speed is to be measured, the speed range of 500 to 1500 rpm can be converted to a voltage range of 0 to +10 V using a measuring transducer. At a measured speed of 865 rpm, the measuring transducer would output a voltage value of +3.65 V.

**1500 rpm**



1000 rpm

**10 V: 1000 rpm = 0.01 V/rpm**

**365 rpm x 0.01 V/rpm = 3.65 V**

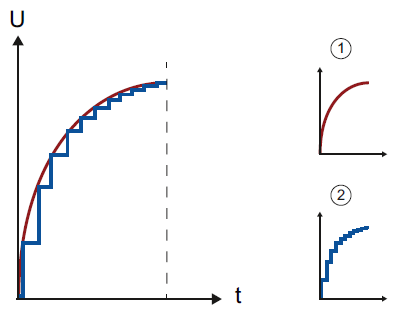
**+10 V**

## Analog modules – A/D converter

These electrical voltages, currents or resistances are then connected to an analog module that digitizes this signal for further processing in the PLC.

If analog quantities will be processed with a PLC, the read-in voltage, current or resistance value must be converted to digital information. The analog value is converted to a bit pattern. This conversion is referred to as analog-to-digital conversion (A/D conversion). This means, for example, that the voltage value of 3.65 V is stored as information in a series of binary digits.

The result of this conversion is always a 16-bit word for SIMATIC products. The integrated ADC (analog-to-digital converter) of the analog input module digitizes the analog signal being acquired and approximates its value in the form of a stepped curve. The most important parameters of an ADC are its resolution and conversion rate.

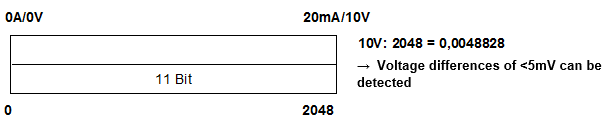


1: Analog value

2. Digital value

The more binary digits the digital representation uses, the finer the resolution is. For example, if only 1 bit was available for the voltage range of 0 to +10 V, you would only know whether the measured voltage is between 0 and +5 V or between +5 V and +10 V. With 2 bits, the range can be divided into 4 individual ranges, i.e., 0 to 2.5 / 2.5 to 5 / 5 to 7.5 / 7.5 to 10 V. Conventional A/D converters in control engineering use 8 bits, 11 bits or more for converting.

With 8 bits you have 256 individual ranges, while 11 bits provide a resolution of 2048 individual ranges.



11-bit

**2048**

**0A/0V**

**20mA/10V**

**10 V: 2048 = 0.0048828**

🡪 **Voltage differences of <5 mV can be detected**

## Data types of the SIMATIC S7-1200

The SIMATIC S7-1200 has many different data types for representing different numerical formats. A list of some of the elementary data types is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **Size (bits)** | **Range** | **Example of constant entry** |
| Bool | 1 | 0 to 1 | TRUE, FALSE, O, 1 |
| Byte | 8 | 16#00 to 16#FF | 16#12, 16#AB |
| Word | 16 | 16#0000 to 16#FFFF | 16#ABCD, 16#0001 |
| DWord | 32 | 16#00000000 to 16#FFFFFFFF | 16#02468ACE |
| Char | 8 | 16#00 to 16#FF | 'A', ‘r’, ‘@’ |
| Sint | 8 | -128 to 127 | 123,-123 |
| **Int** | **16** | **-32,768 to 32,767** | **123, -123** |
| Dint | 32 | -2,147,483,648 to 2,147,483,647 | 123, -123 |
| USInt | 8 | 0 to 255 | 123 |
| Ulnt | 16 | 0 to 65,535 | 123 |
| UDInt | 32 | 0 to 4,294,967,295 | 123 |
| **Real** | **32** | **+/-1.18 x 10 -38 to +/-3.40 x 10 38** | **123.456, -3.4, -1.2E+12, 3.4E-3** |
| LReal | 64 | +/-2.23 x 10 -308 to +/-1.79 x 10 308 | 12345.123456789  -1.2E+40 |
| Time | 32 | T#-24d\_20h\_31 m\_23s\_648ms to T#24d\_20h\_31 m\_23s\_647ms  Saved as: -2,147,483,648 ms to +2,147,483,647 ms | T#5m\_30s  5#-2d  T#1d\_2h\_15m\_30x\_45ms |
| String | Variable | 0 to 254 characters in byte size | 'ABC' |

**Note:** The **'INT'** and **'REAL'** data types play a large role in analog value processing. This is because read-in analog values exist as 16-bit integers in the '**INT'** format, and in order to ensure exact further processing only **'REAL'** floating-point numbers should be used due to rounding errors in the case of '**INT'**.

## Reading/writing analog values

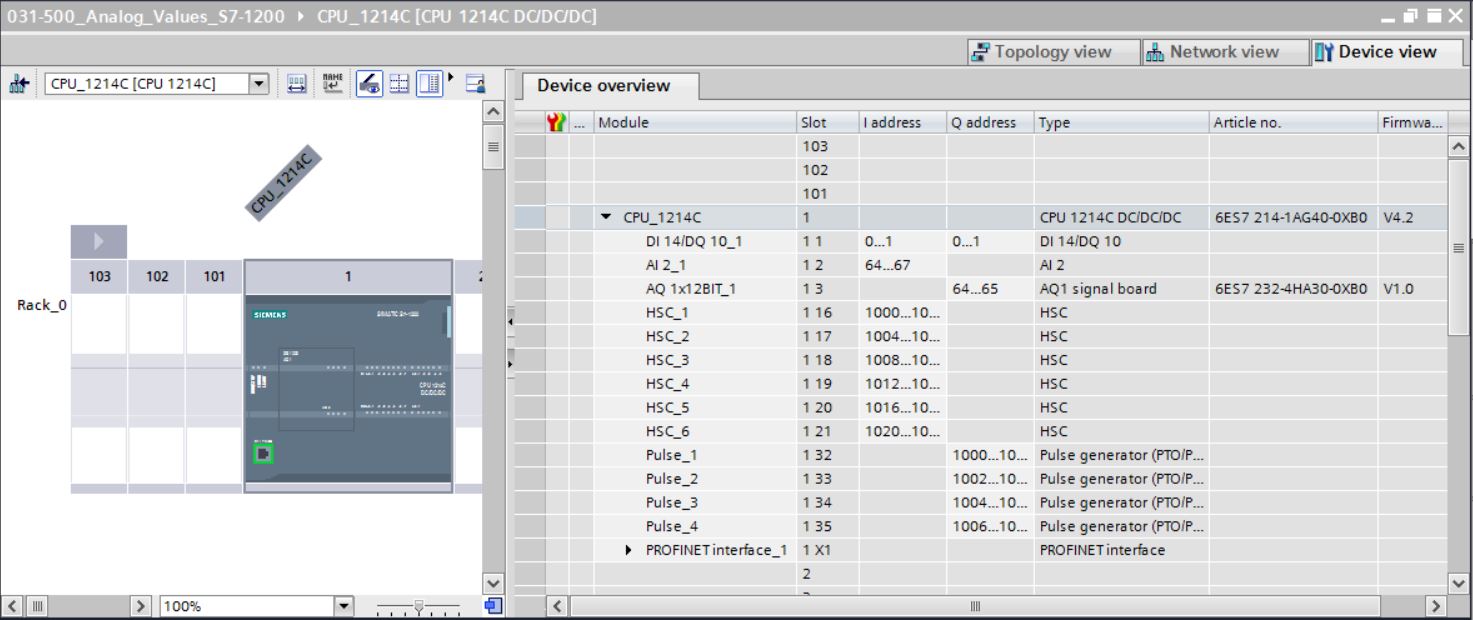
Analog values are read into the PLC or output from the PLC as word information. These words are accessed, for example, with the following operands:

%IW 64 Analog input word 64

%QW 64 Analog output word 64

Each analog value (“channel”) occupies one input or output word. The format is **‘Int’**, an integer.

The addressing of input and output words conforms to the addressing in the device overview.   
For example:

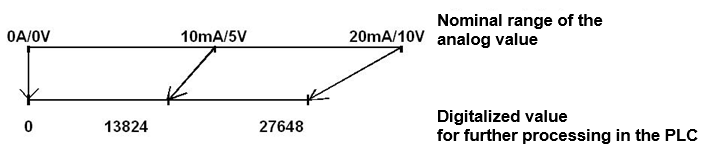


Here, the address of the first analog input would be %IW 64, and the address of the second analog input would be %IW 66.

The address of the analog output would be %QW 64.

The analog value transformation for further processing in the PLC is the same for analog inputs and analog outputs.

The digitized value ranges are as follows:



**10 mA/5 V**

**13824**

**27648**

**Digitalized value for further processing in the PLC**

**20mA/10V**

**Nominal range of the analog value**

**0A/0V**

Often, these digitized values still have to be normalized by further processing them in the PLC in an appropriate manner.

## Normalizing analog values

If an analog input value exists as a digitized value in the range +/- 27648, it must usually still be normalized so that the numerical values correspond to the physical quantities in the process.

Likewise, the analog output usually results from setting of a normalized value that then still has to be scaled to the output value +/- 27648.

In the TIA Portal, ready-made blocks or arithmetic operations are used for normalizing and scaling.

For this to be carried out as exactly as possible, the values for the normalizing must be converted to the REAL data type to minimize rounding errors.

# Task

In this chapter, a function for analog control of the conveyor speed will be added to the program from chapter "SCE\_EN\_031-300 IEC Timers and Counters S7-1200".

# Planning

The analog control of the conveyor speed will be programmed in the "MOTOR\_SPEEDCONTROL" [FC10] function as an expansion of the "SCE\_EN\_031-300 IEC Timers and Counters S7-1200" project. This project must be retrieved from the archive in order to add this function. The "MOTOR\_ SPEEDCONTROL" [FC10] function will be called in the "Main“ [OB1]" organization block and wired. The control of the conveyor motor must be changed to –Q3 (conveyor motor -M1 variable speed).

## Analog control of the conveyor speed

The speed will be set at an input of the "MOTOR\_SPEEDCONTROL" [FC10] function in revolutions per minute (range: +/- 50 rpm). The data type is 32-bit floating-point number (Real).

First, the function will be checked for correct entry of the speed setpoint in the range +/- 50 rpm.

If the speed setpoint is outside the range +/- 50 rpm, the value 0 with data type 16-bit integer (Int) will be output at the output. The return value of the function (Ret\_Val) will then be assigned the value TRUE (1).

If the speed setting is within the range +/- 50 rpm, this value will first be normalized to the range 0…1 and then scaled to +/- 27648 with data type 16-bit integer (Int) for output as the speed manipulated value at the analog output.

The output will then be connected with signal U1 (manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm).

## Technology diagram

Here you see the technology diagram for the task.



Figure 1: Technology diagram



Figure 2: Control panel

## Reference list

The following signals are required as global operands for this task.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DI** | **Type** | **Identifier** | **Function** | **NC/NO** |
| I 0.0 | BOOL | -A1 | Return signal emergency stop OK | NC |
| I 0.1 | BOOL | -K0 | Main switch "ON" | NO |
| I 0.2 | BOOL | -S0 | Mode selector manual (0)/ automatic (1) | Manual = 0  Auto = 1 |
| I 0.3 | BOOL | -S1 | Pushbutton automatic start | NO |
| I 0.4 | BOOL | -S2 | Pushbutton automatic stop | NC |
| I 0.5 | BOOL | -B1 | Sensor cylinder -M4 retracted | NO |
| I 1.0 | BOOL | -B4 | Sensor part at slide | NO |
| I 1.3 | BOOL | -B7 | Sensor part at end of conveyor | NO |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DO** | **Type** | **Identifier** | **Function** |  |
| Q 0.2 | BOOL | -Q3 | Conveyor motor -M1 variable speed |  |
| QW 64 | BOOL | -U1 | Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm |  |

***Legend for reference list***

|  |  |
| --- | --- |
| DO | Digital Output |
| AO | Analog Output |
| Q | Output |

|  |  |
| --- | --- |
| DI | Digital Input |
| AI | Analog Input |
| I | Input |
| NC | Normally Closed |
| NO | Normally Open |

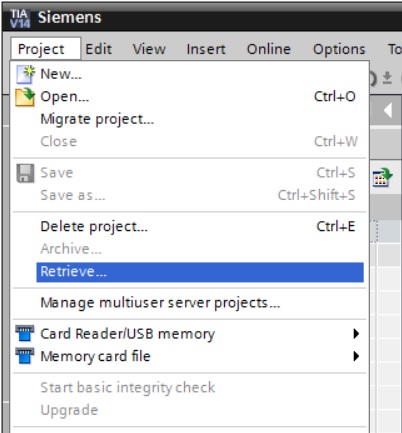
# Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps.   
Otherwise, simply follow the detailed steps in the instructions.

## Retrieve an existing project

* Before we can expand the "SCE\_EN\_031-300\_IEC\_Timers\_Counters\_S7-1200.zap14" project from chapter "SCE\_EN\_031-300\_IEC\_Timers\_Counters\_S7-1200", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.

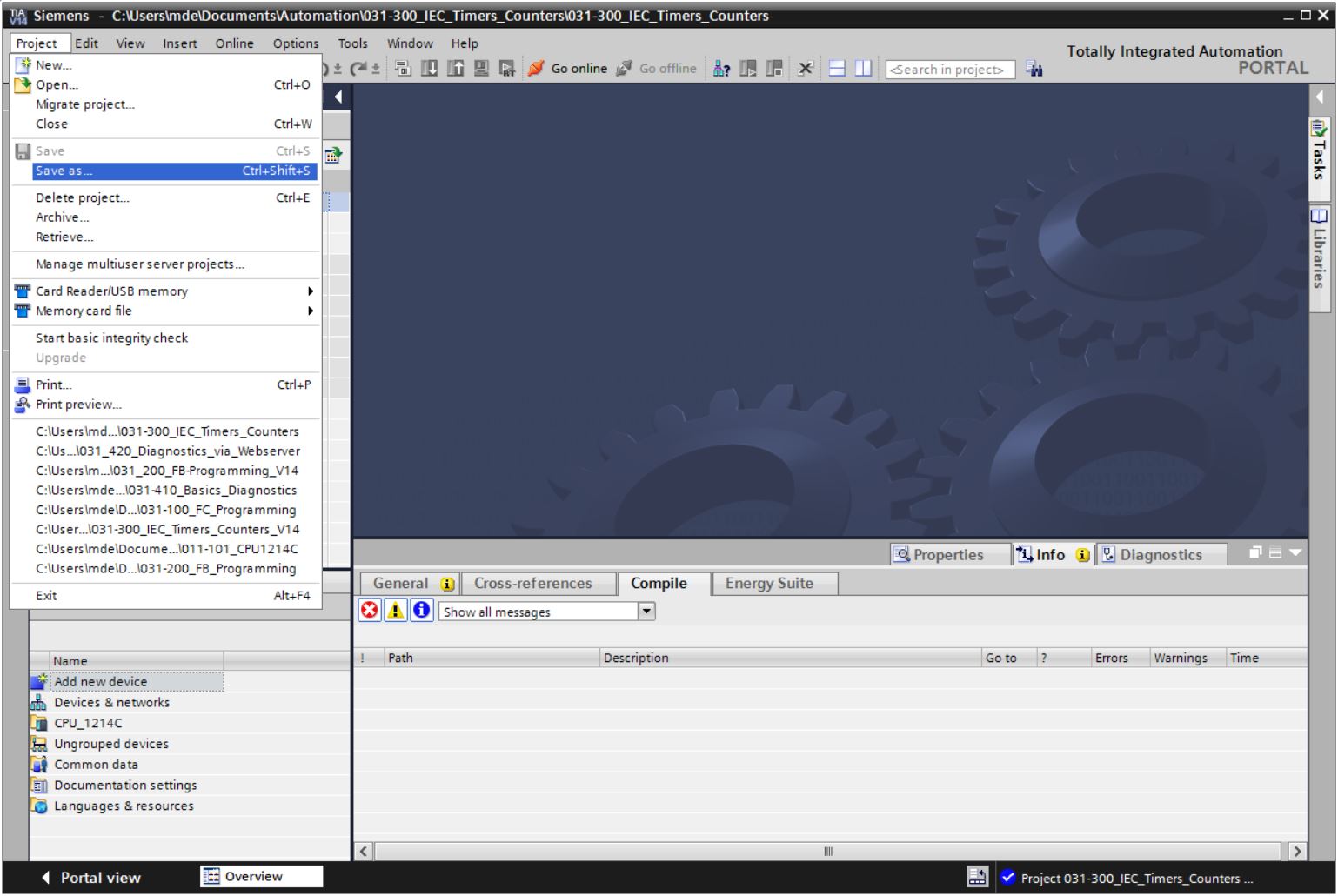
(→ Project → Retrieve → Select a .zap archive → Open)



* The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".

(→ Target directory → OK)

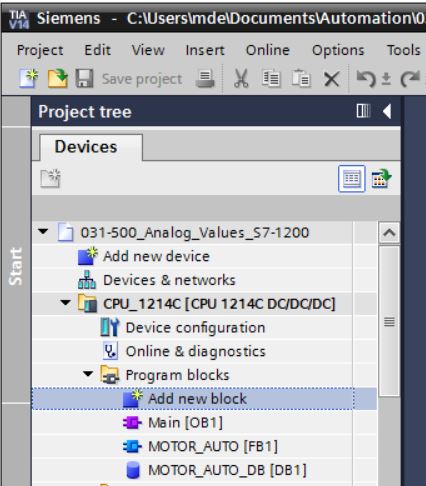
* Save the opened project under the name 031-500\_Analog\_Values\_S7-1200.   
  (→ Project → Save as … → 031-500\_Analog\_Values\_S7-1200 → Save)



## Create the "MOTOR\_SPEEDCONTROL" function

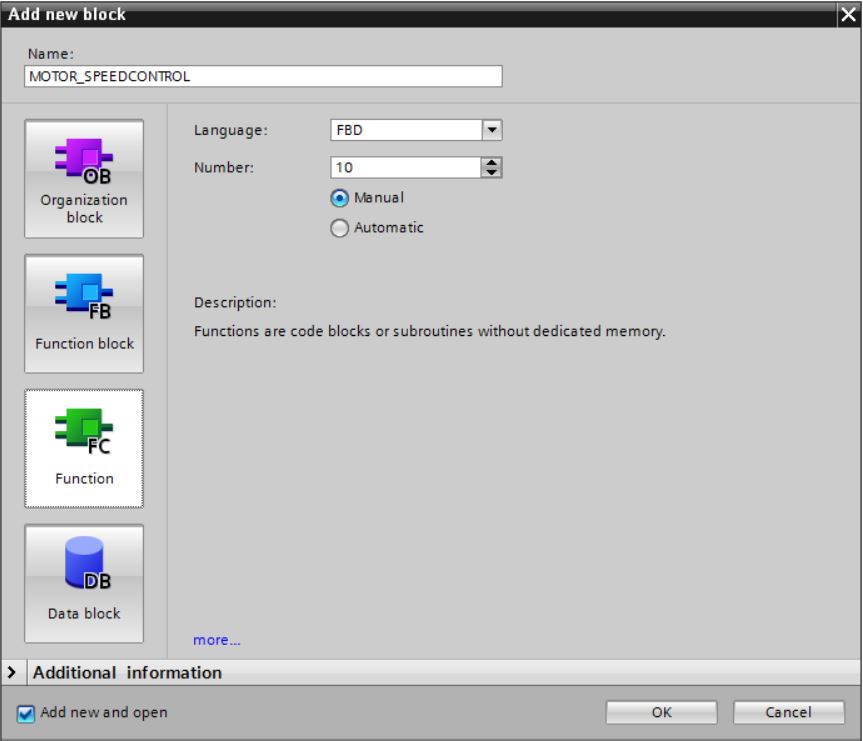
* Select the 'Program blocks' folder of your CPU\_1214C and then click "Add new block" to create a new function there.

(→ CPU\_1214C [CPU 1214C DC/DC/DC] → Add new block)

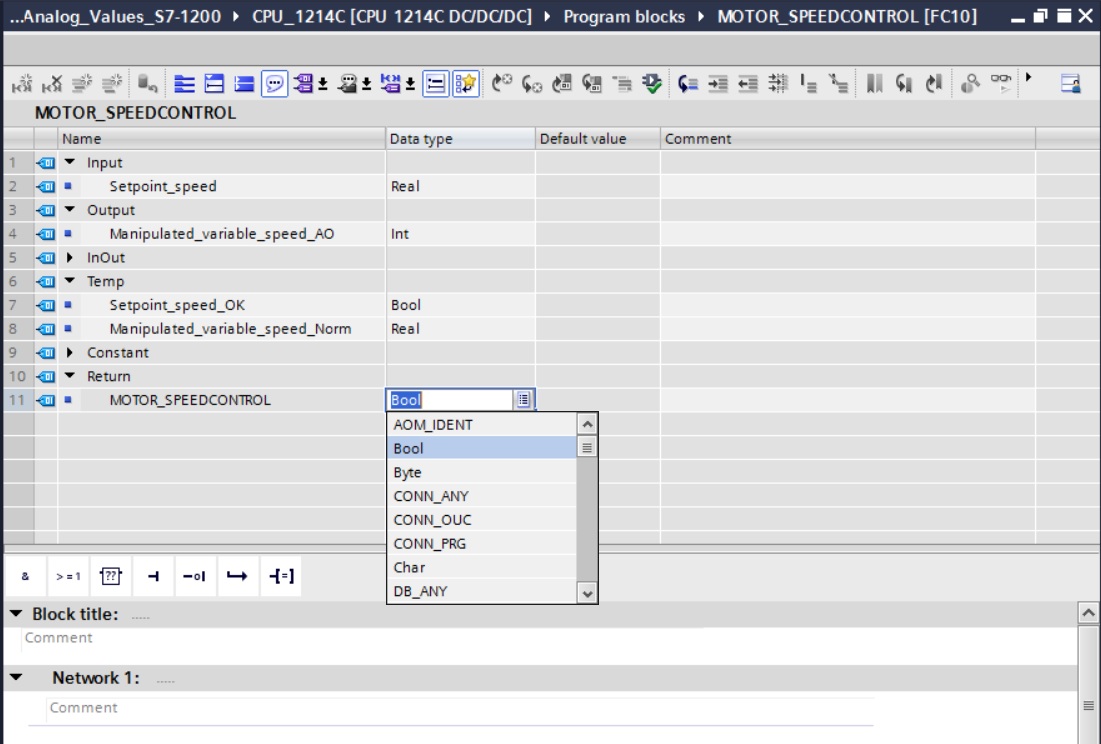


* Select  in the next dialog and rename your new block to: "MOTOR\_SPEEDCONTROL". Set the language to FBD and manually assign the number "10". Select the "Add new and open" check box. Click "OK".

(→→ Name: MOTOR\_SPEEDCONTROL → Language: FBD → Number: 10 Manual →  Add new and open → OK)



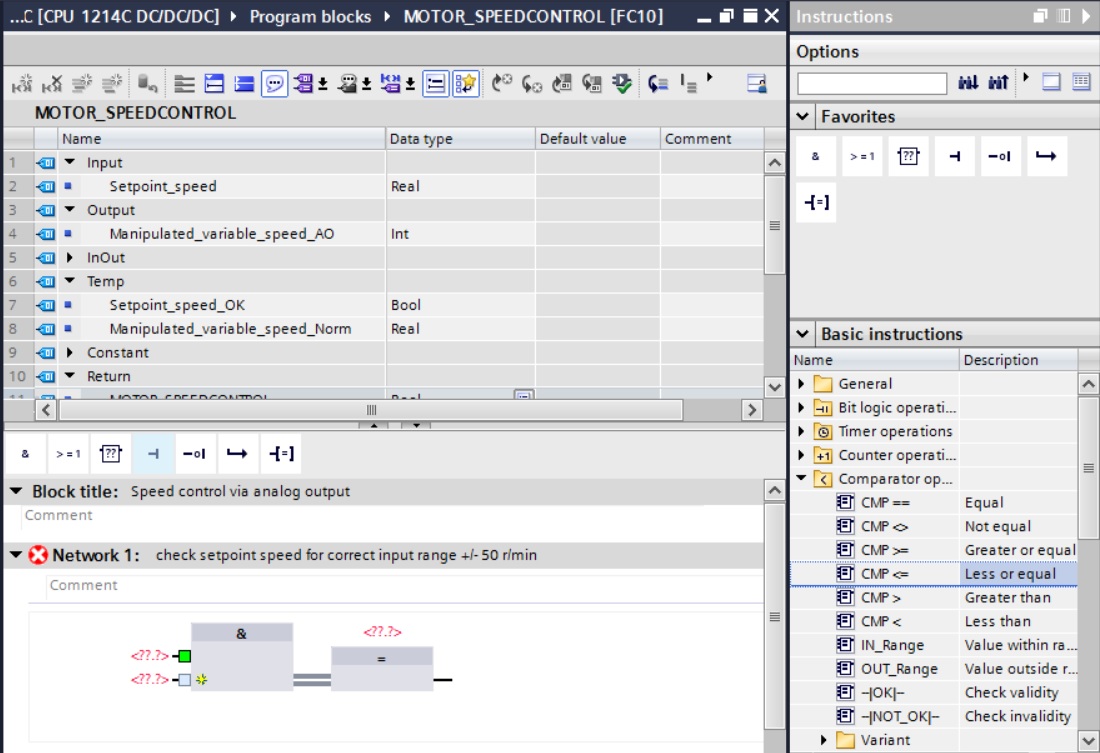
* Create the local tags with their comments as shown here and change the data type of the 'Return' tag from 'Void' to 'Bool'. (→ Bool)



**Note:** Be sure to use the correct data types.

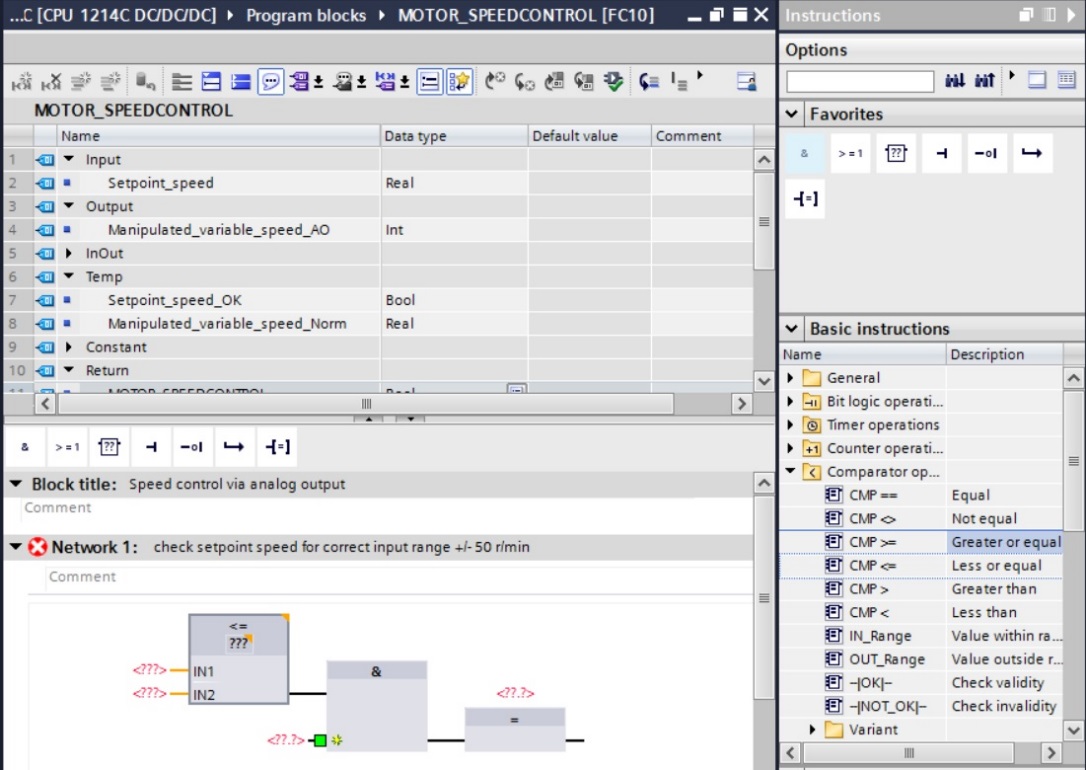
* Insert an Assignment '' in the first network and an 'And'' in front of it. Then use drag & drop to move the 'Comparator operation' 'Less or equal' from the 'Basic instructions' onto the first input of the  AND logic operation.

(→  →  → Basic instructions → Comparator operations→ CMP<=)

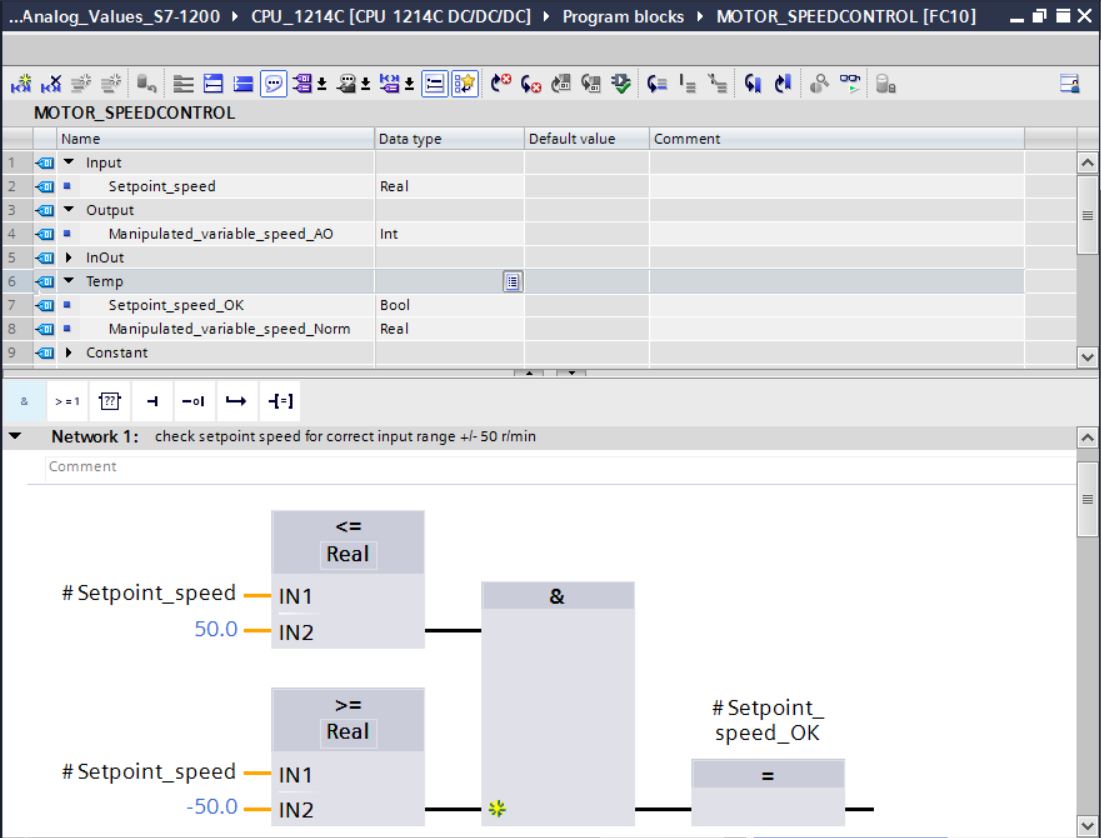


* Next use drag & drop to move the 'Comparator operation' 'Greater or equal' onto the second input of the  AND logic operation.

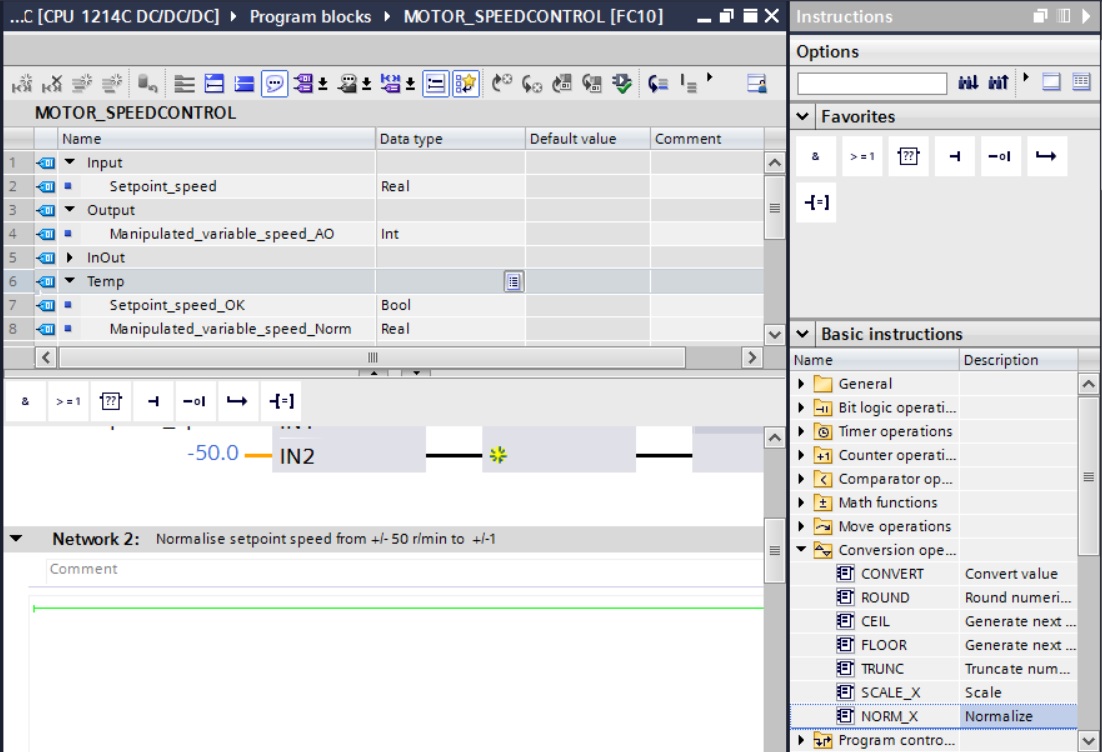
(→ Basic instructions → Comparator operations→ CMP>=)



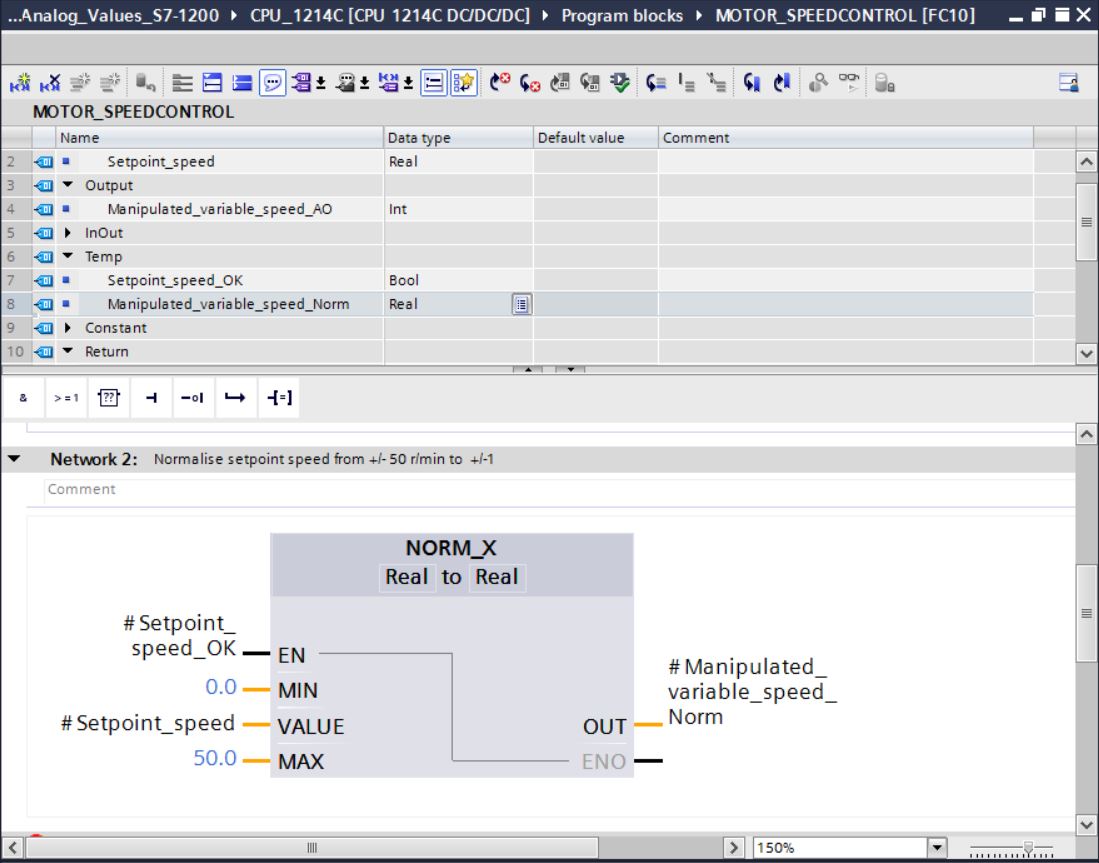
* Connect the contacts in Network 1 with the constants and local tags as shown here. The data types in the comparator operations are automatically adapted to 'Real'.



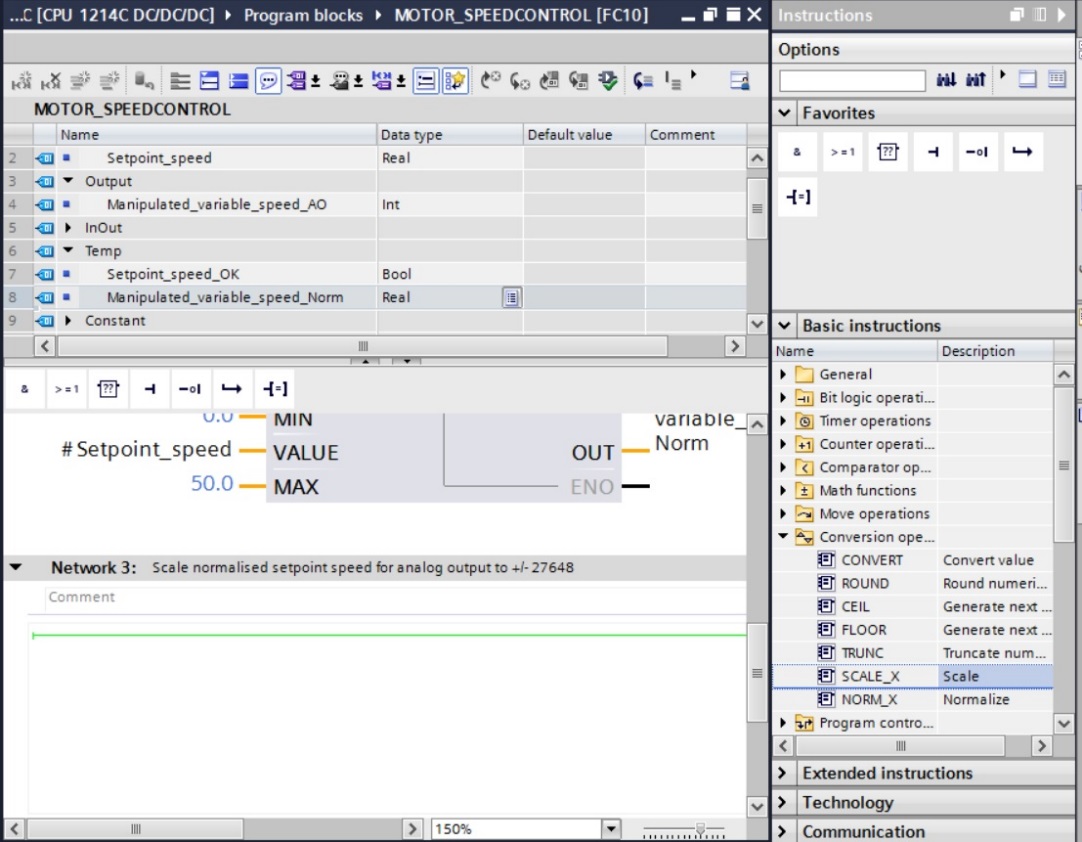
* Use drag & drop to move the 'Conversion operation' 'NORM\_X' into Network 2 in order to normalize the speed setpoint of +/- 50 rpm to +/- 1. (→ Basic instructions → Conversion operations → NORM\_X)



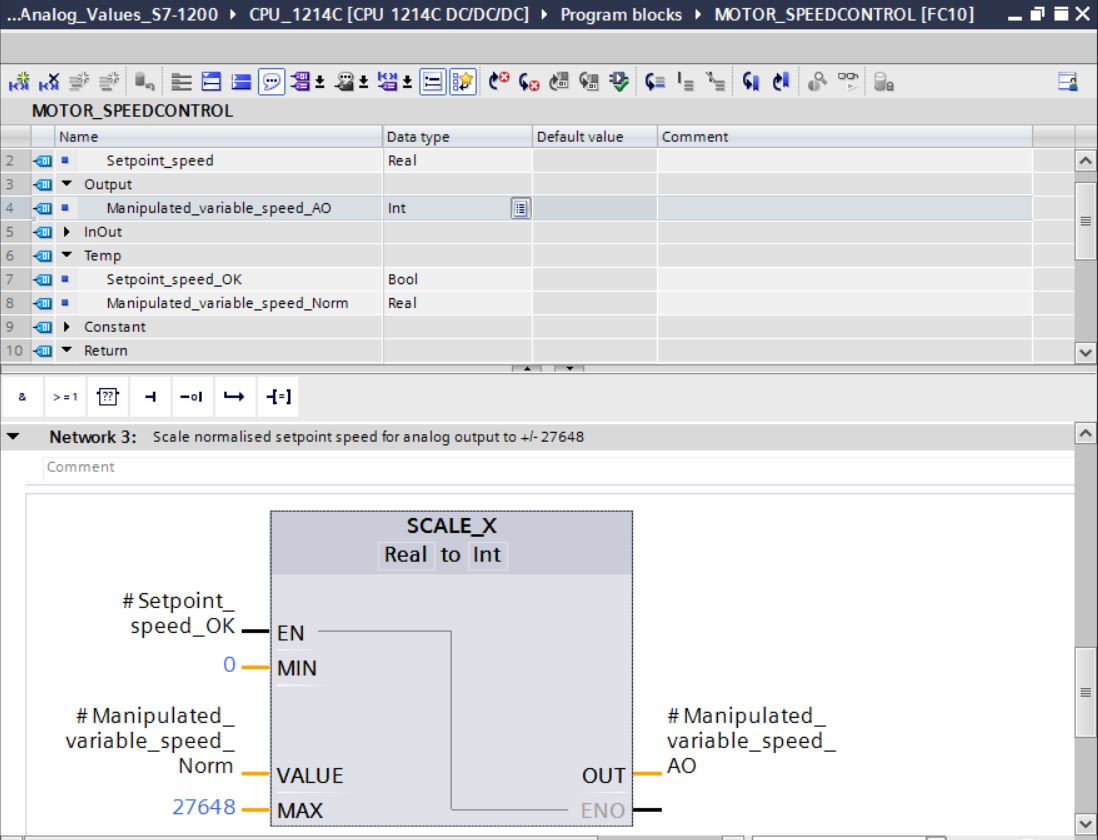
* Connect the contacts in Network 2 with the constants and local tags as shown here.   
  The data types in 'NORM\_X' are automatically adapted to 'Real'.



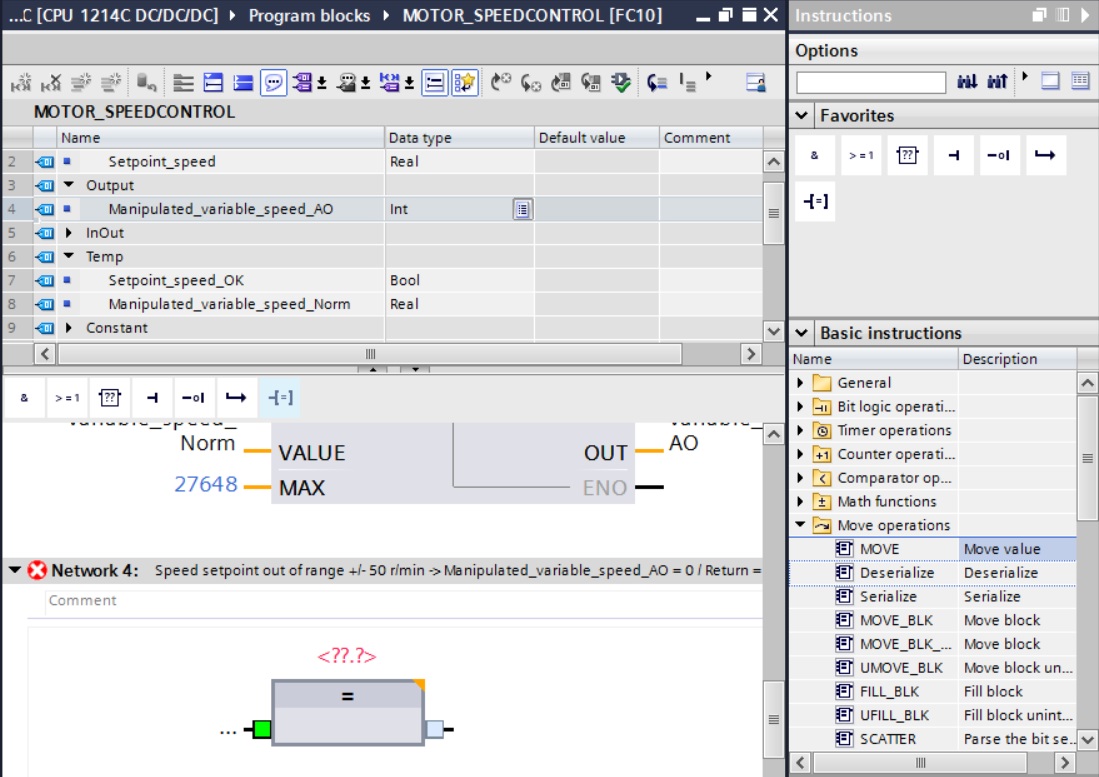
* Use drag & drop to move the 'Conversion operation' 'SCALE\_X' into Network 3 in order to scale the speed setpoint from the normalized +/- 1 onto the range for the analog output +/- 27468.   
  (→ Basic instructions → Conversion operations → SCALE\_X)



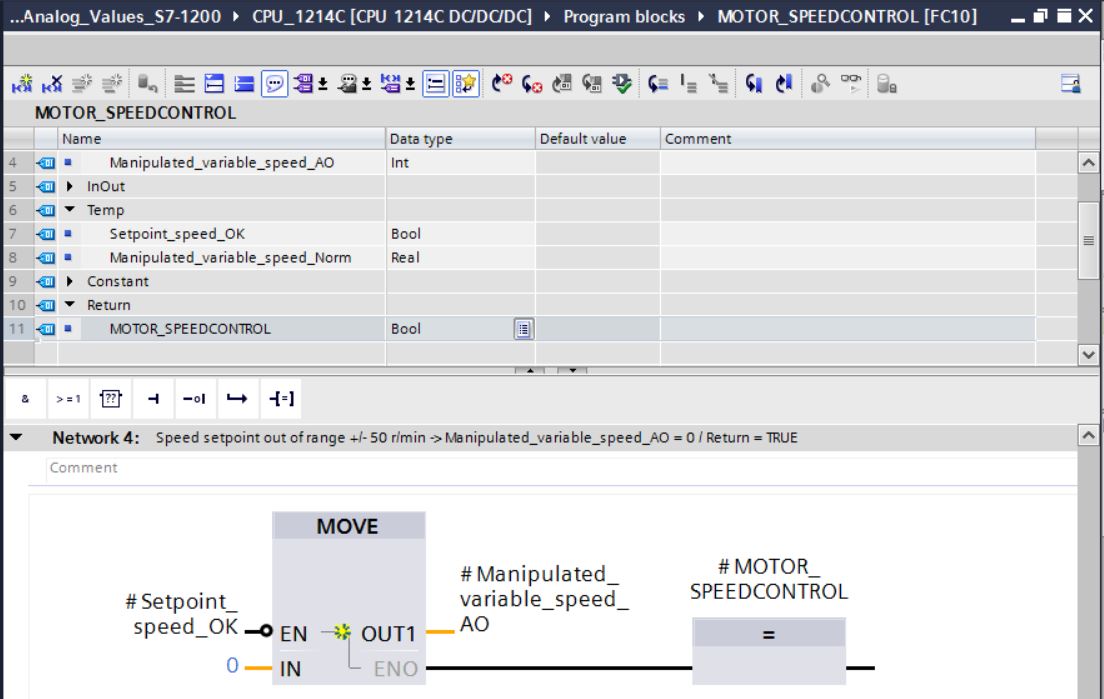
* Connect the contacts with the constants and local tags in Network 3 as well, as shown here. The data types in 'SCALE\_X' are automatically changed to 'Real' or 'Int'.



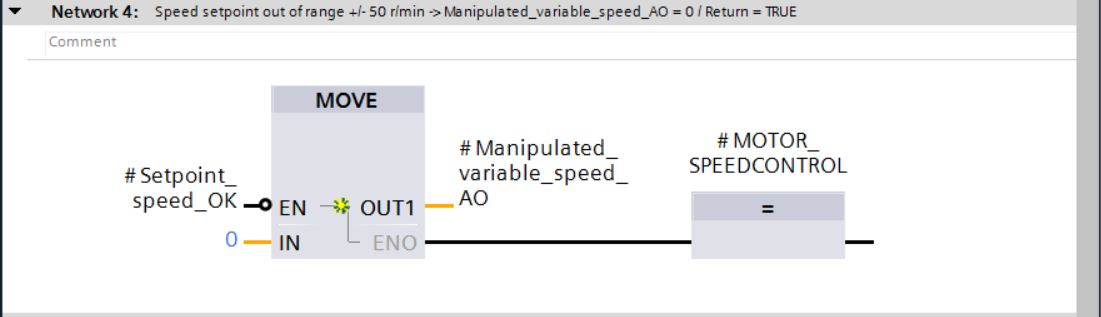
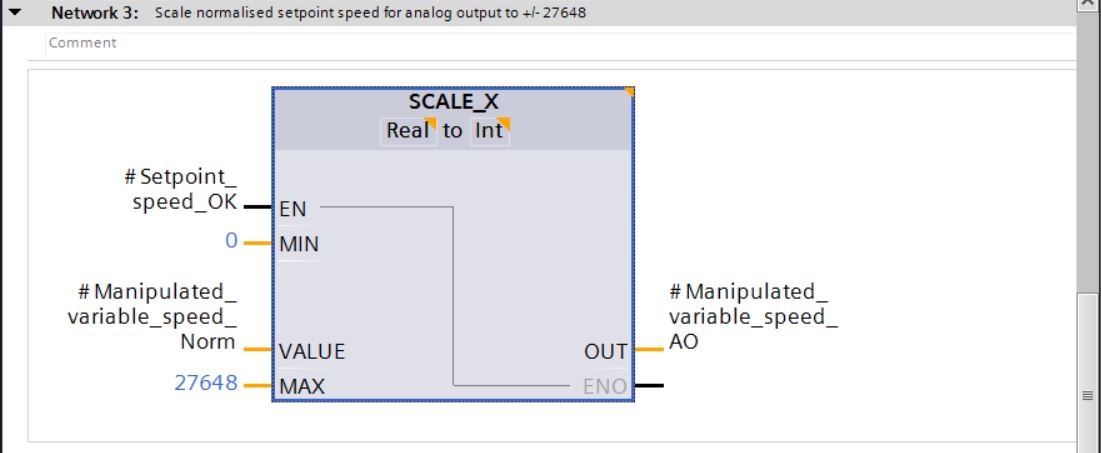
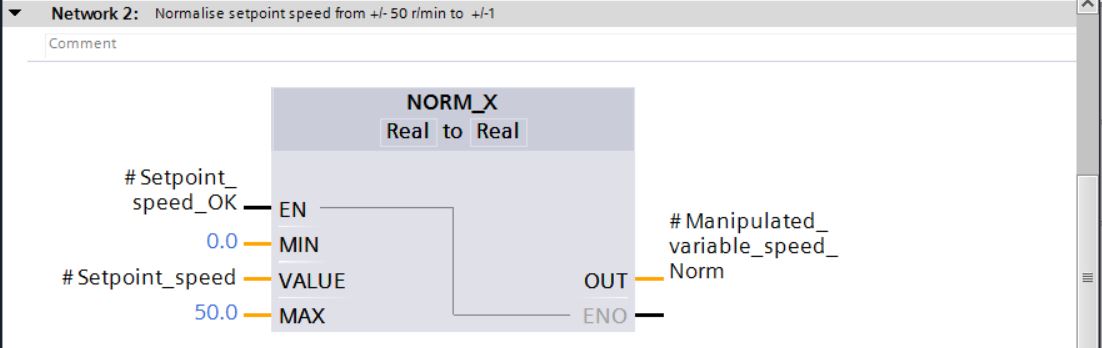
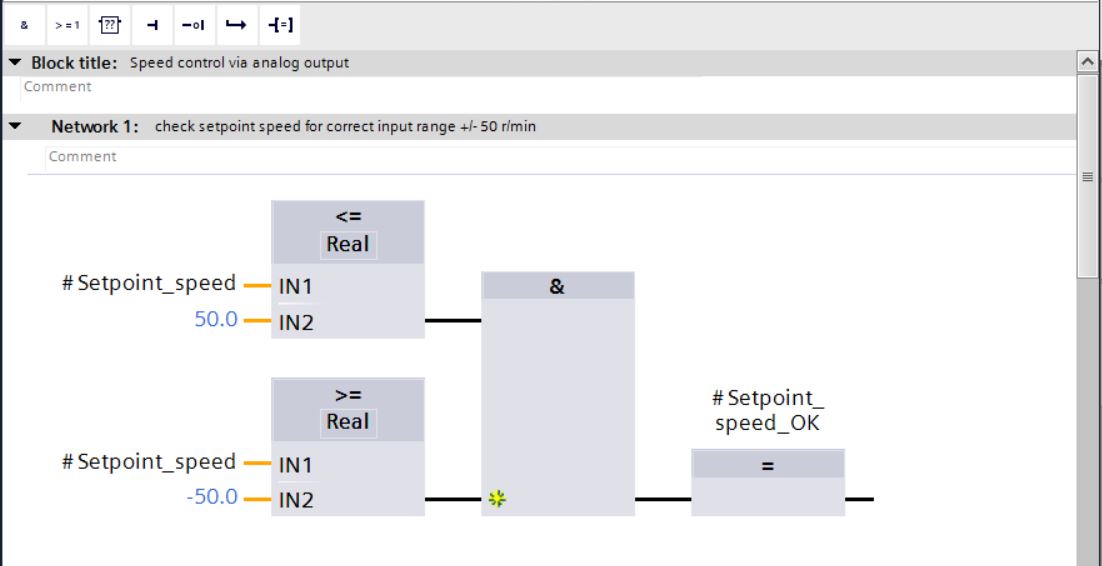
* Insert an Assignment '' in the fourth network. Use drag & drop to move the 'Move' command from the 'Move operations' folder under 'Basic instructions' in front of the Assignment.   
  (→  → Basic instructions → Move operations → MOVE)



* The contacts in Network 4 will now be connected with constants and local tags as shown here. If the speed setpoint is not within the range +/- 50 rpm, the value '0' is output at the analog output and the value TRUE is assigned to the return value (Return) of the "MOTOR\_SPEEDCONTROL“ function.

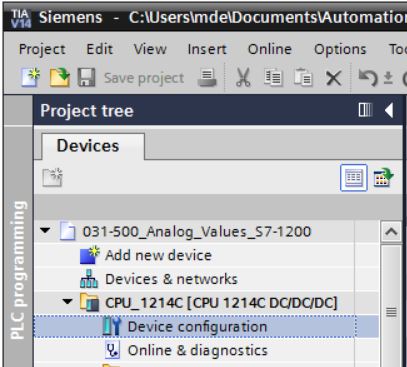


* Do not forget to click . The finished function "MOTOR\_SPEEDCONTROL" [FC10] in FBD is shown below.

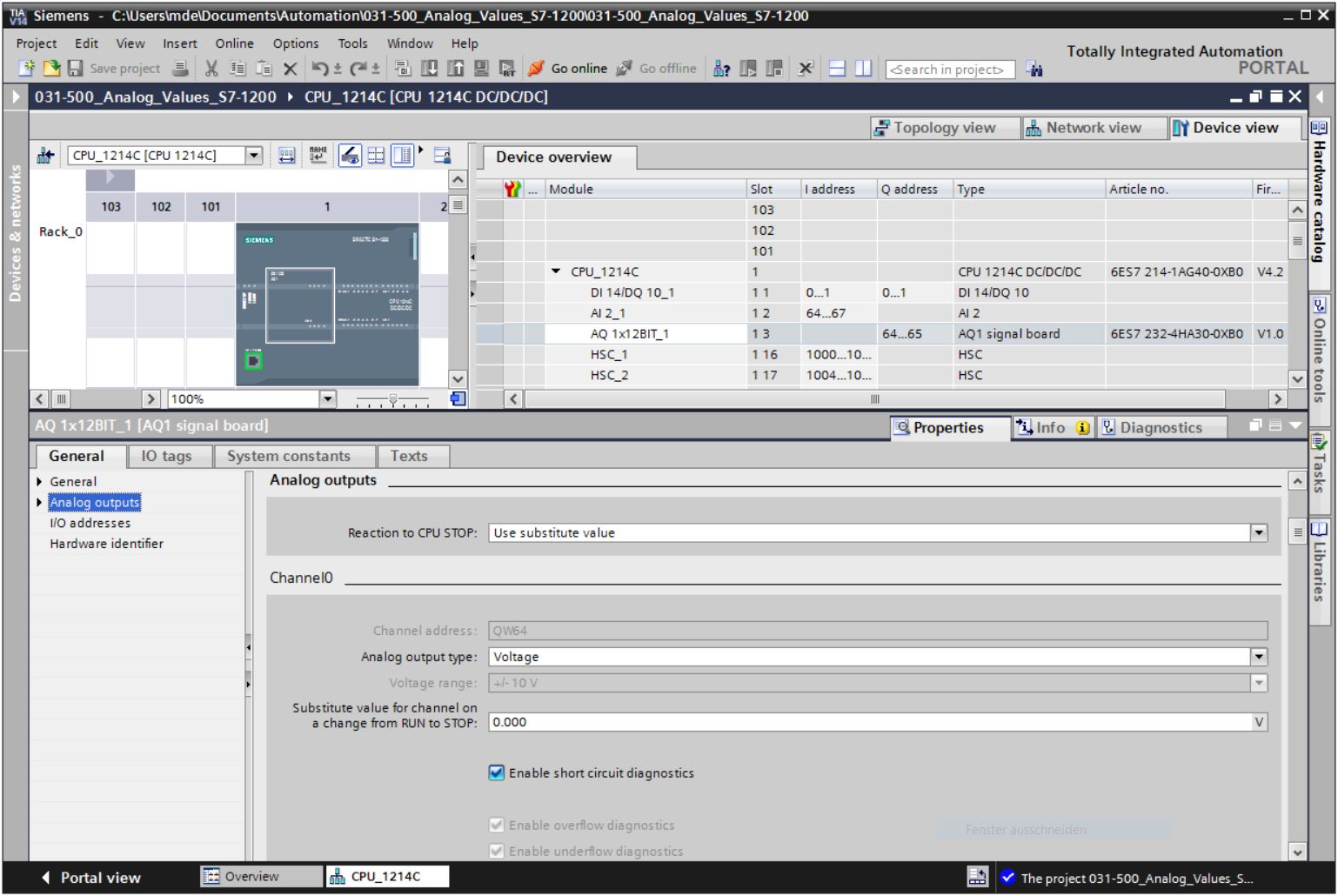


## Configuration of the analog output channel

* Double-click the 'Device configuration' to open it.

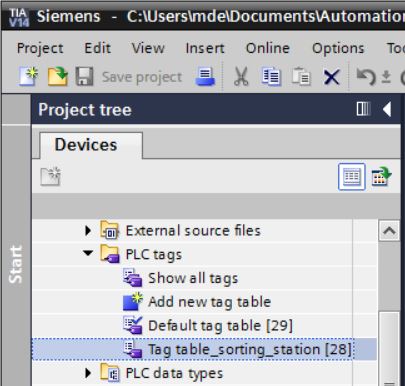


* Check the address setting and the configuration of the analog output channel 0.   
  (→ Q address: 64…65 →Properties → General → Analog outputs → Reaction to CPU STOP: Use substitute value → Channel 0 → Analog output type: Voltage → Substitute value for channel on a change from RUN to STOP: 0.000 V →  Enable short circuit diagnostics)



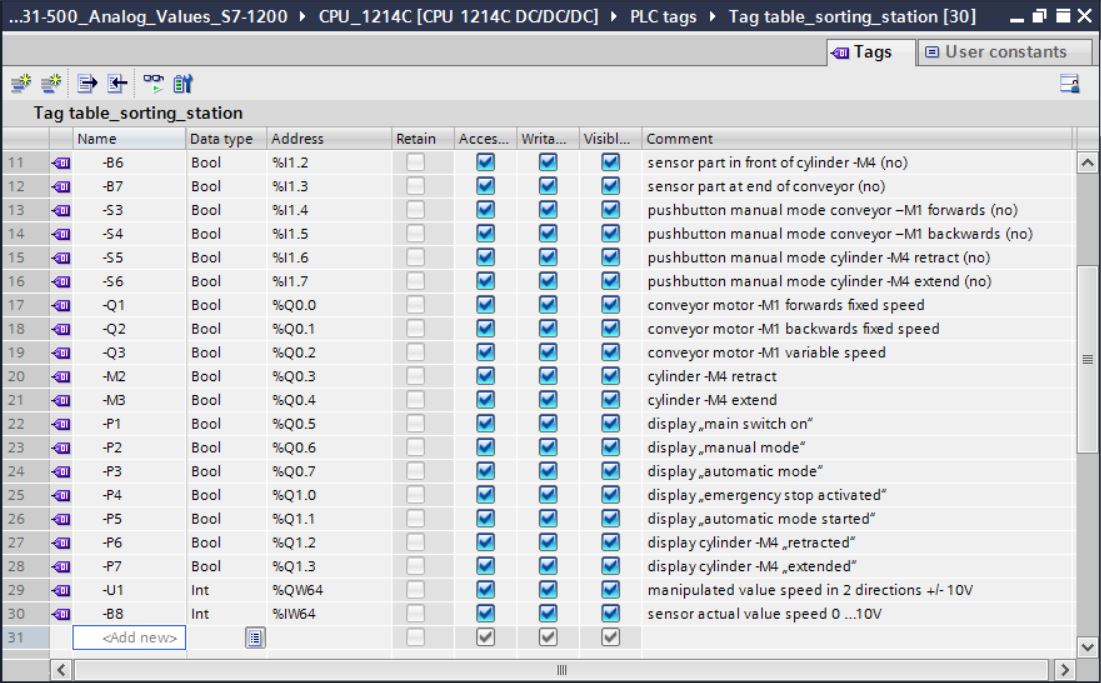
## Expand the tag table to include analog signals

* Double-click the 'Tag table\_sorting station' to open it.



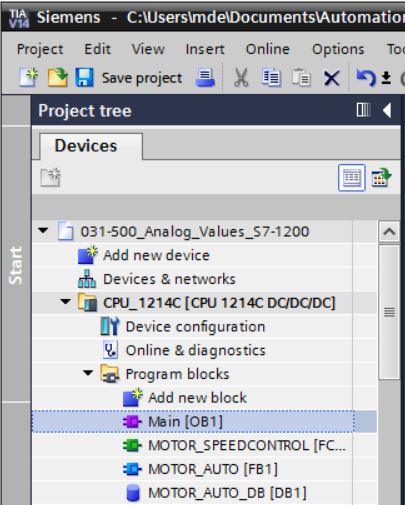
* Add the global tags for the analog value processing to the "Tag table\_sorting station". An analog input B8 and an analog output U1 must be added.

(→ U1 → %QW64 → B8 → %IW64)

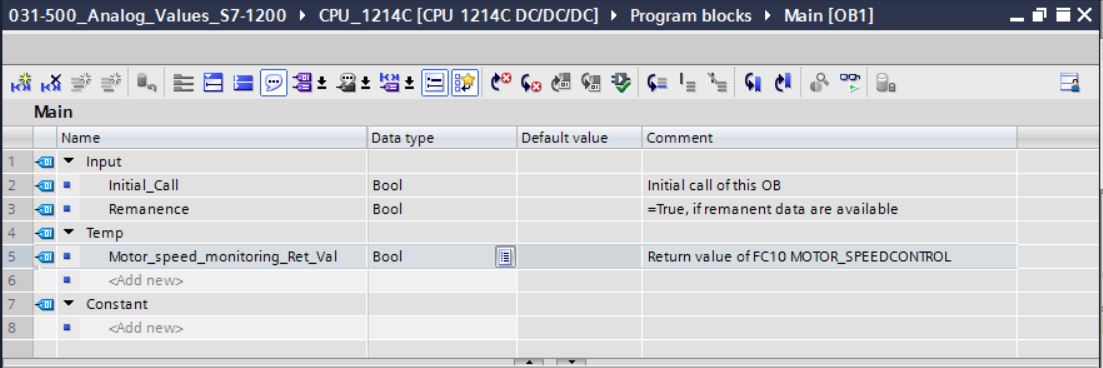


## Call the block in the organization block

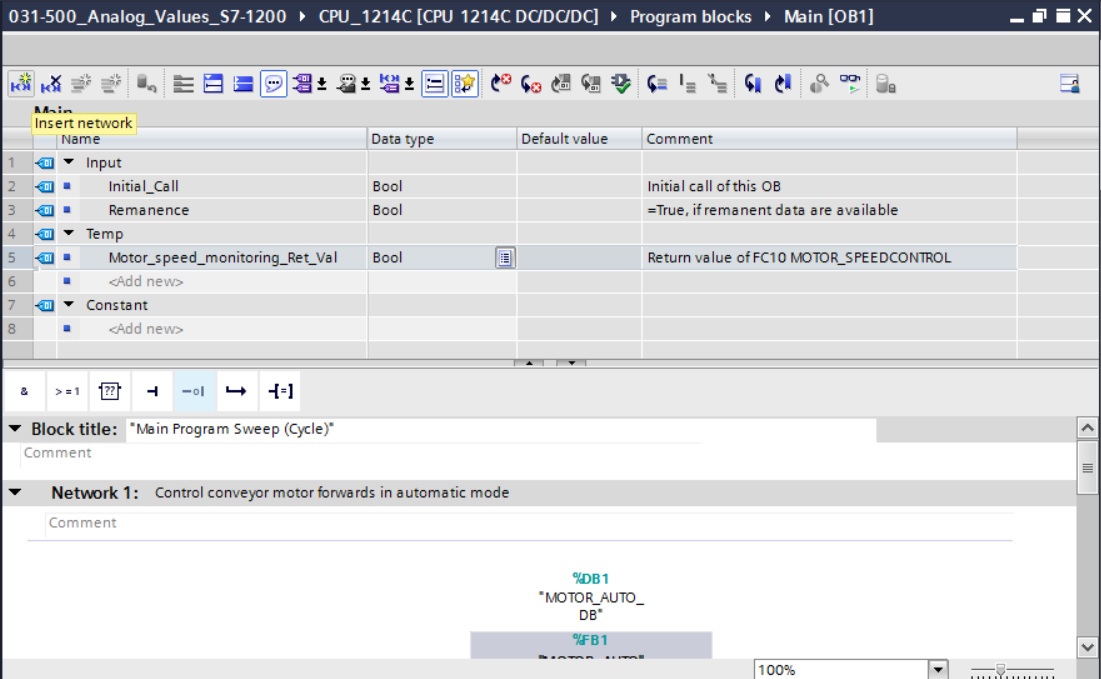
* Open the "Main [OB1]" organization block with a double-click.



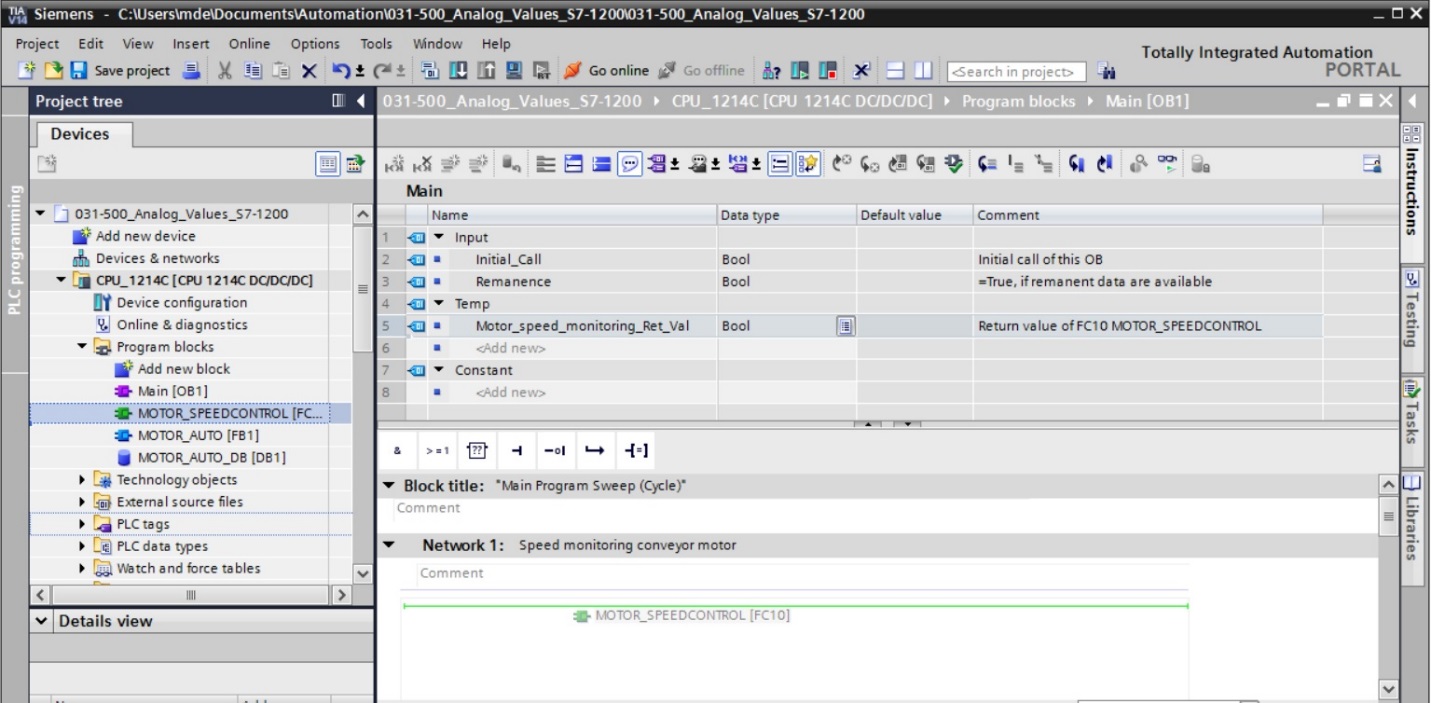
* Add the temporary tag 'Motor\_speed\_monitoring\_Ret\_Val' to the local tags of OB1. These will be needed in order to interconnect the return value of the "MOTOR\_SPEEDCONTROL" function.   
  (→ Temp → Motor\_speed\_monitoring\_Ret\_Val → Bool)



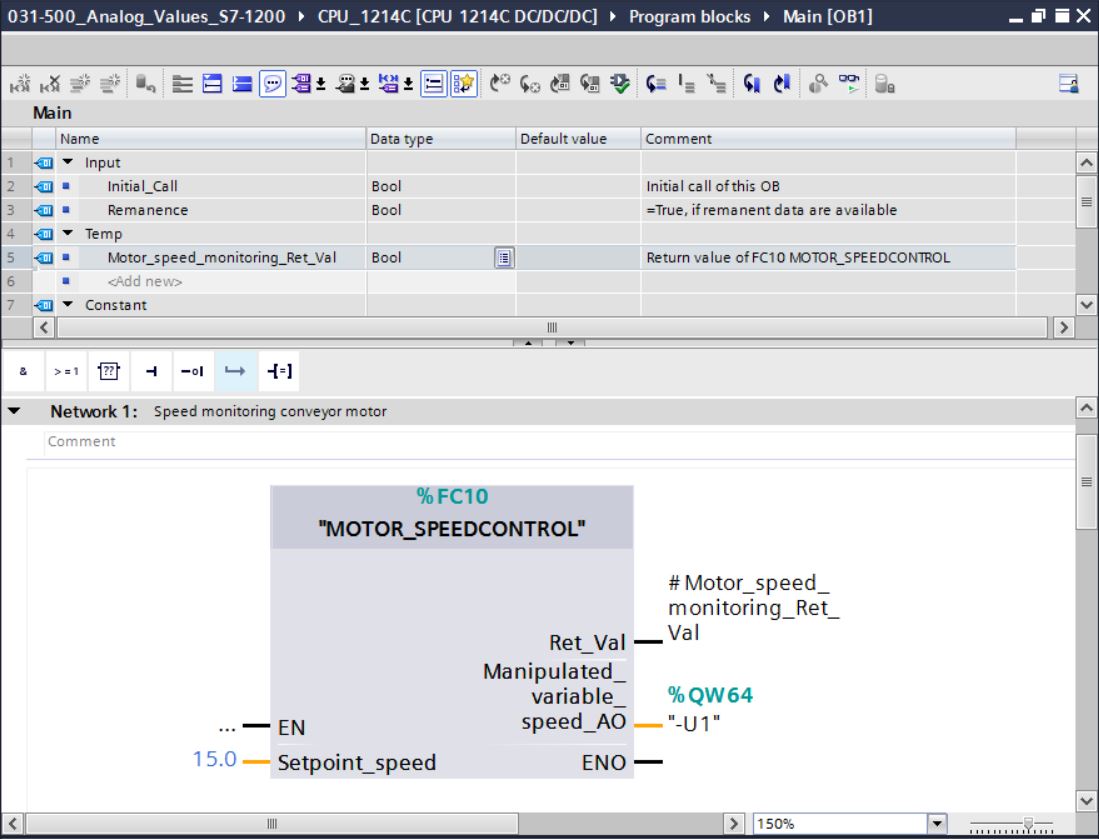
* Select the block title of OB1 and then click '' to insert a new Network 1 in front of the other networks (→ )



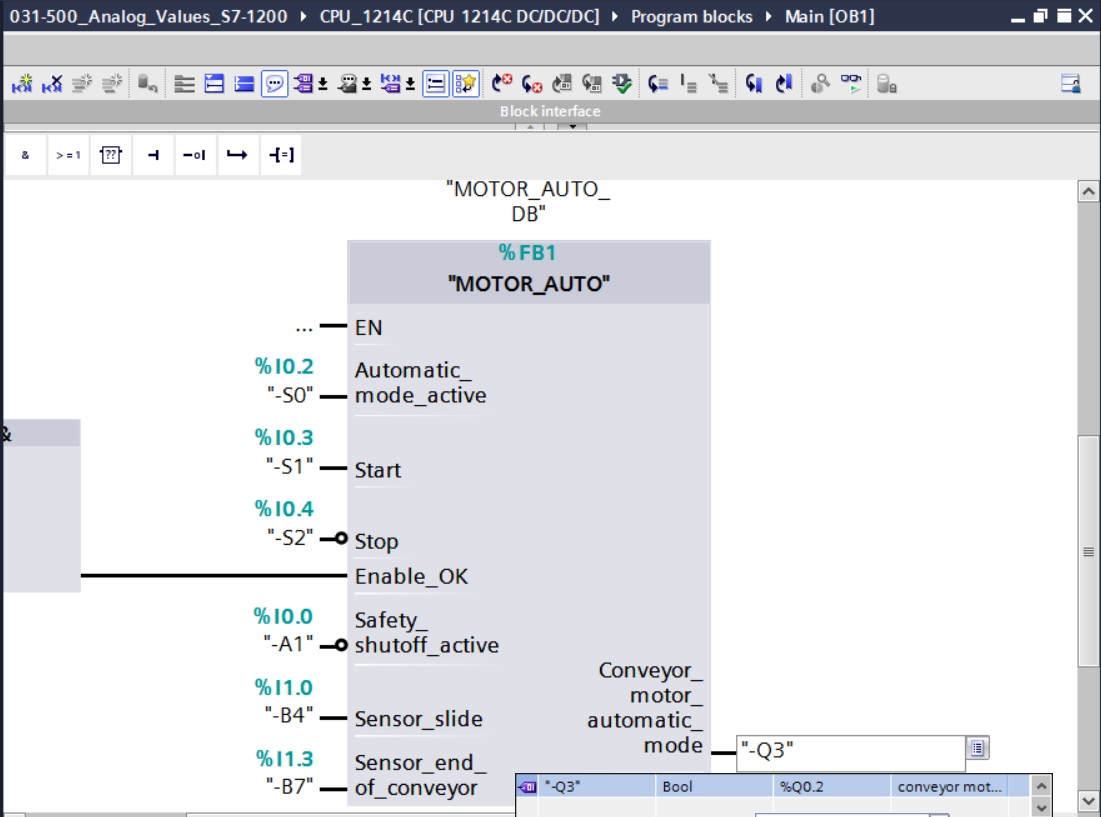
* Use drag & drop to move your "MOTOR\_SPEEDCONTROL [FC10]" function onto the green line in Network 1.



* Connect the contacts with the constants and global and local tags here as shown.

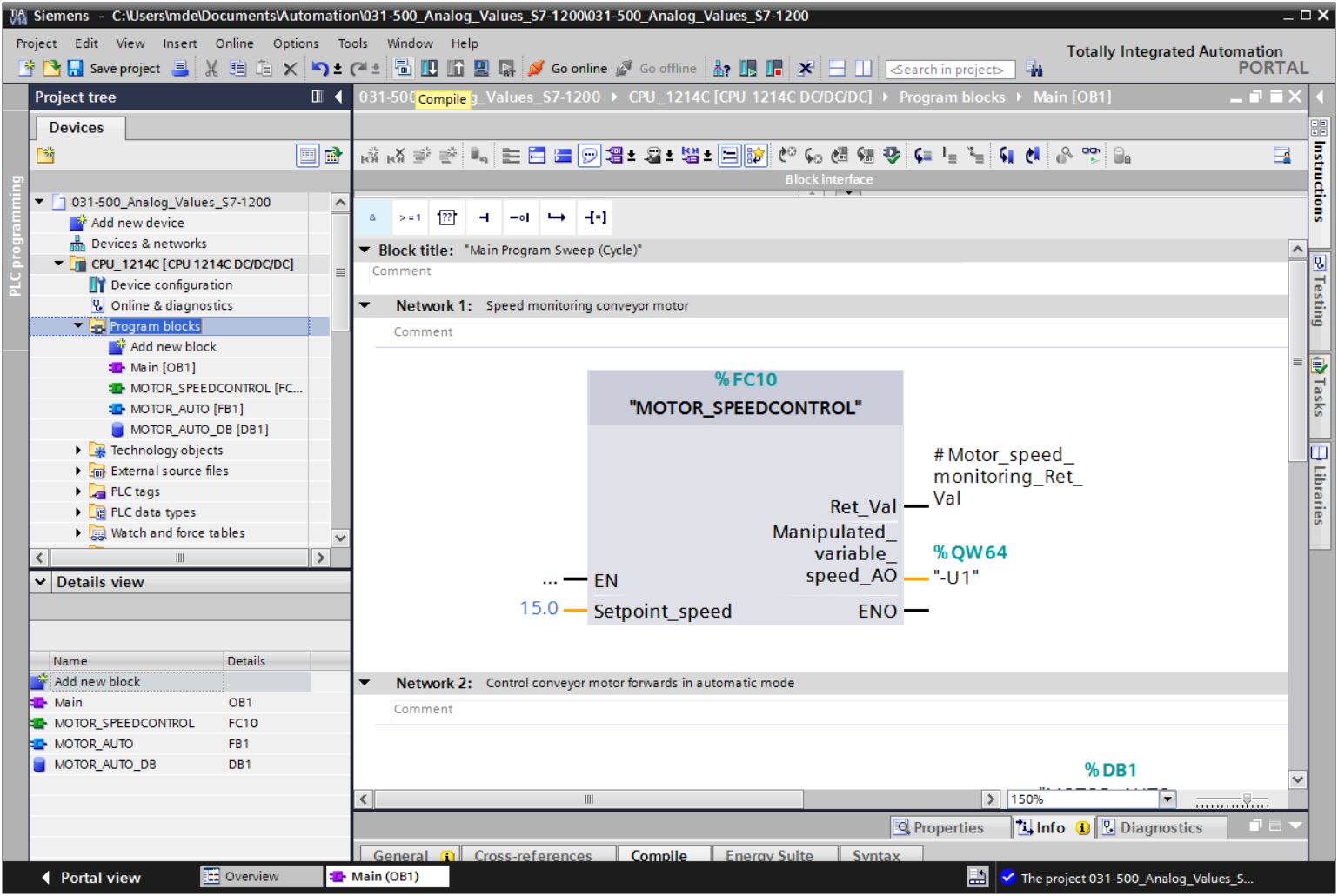


* Change the connection of output tag "Conveyor\_motor\_automatic\_mode" in Network 2 to '-Q3' (Conveyor motor -M1 variable speed) so that the conveyor motor is controlled taking   
  the analog speed setting into consideration. (→ -Q3)

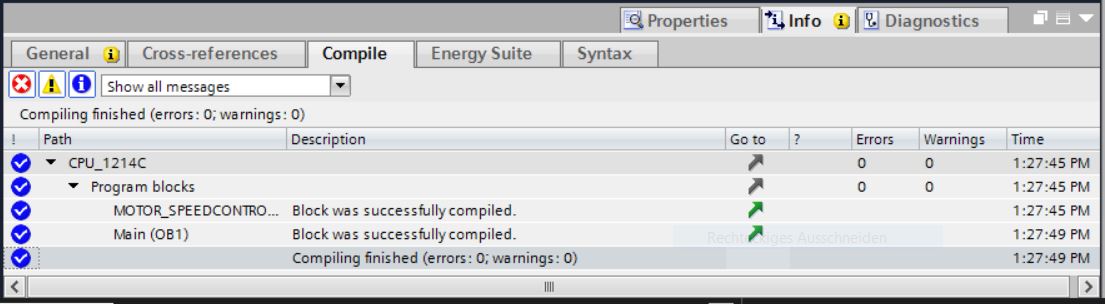


## Save and compile the program

* To save your project, select the  button in the menu. To compile all blocks, click the "Program blocks" folder and select the D:\00_DATA\SIEMENS\Unterlagen\08_Ausbildungsunterlage_TIA-Portal_R1502_dt\032-100 FC-Programmierung\pics\052.jpg icon for compiling in the menu.   
  (→ → Program blocks → D:\00_DATA\SIEMENS\Unterlagen\08_Ausbildungsunterlage_TIA-Portal_R1502_dt\032-100 FC-Programmierung\pics\052.jpg)

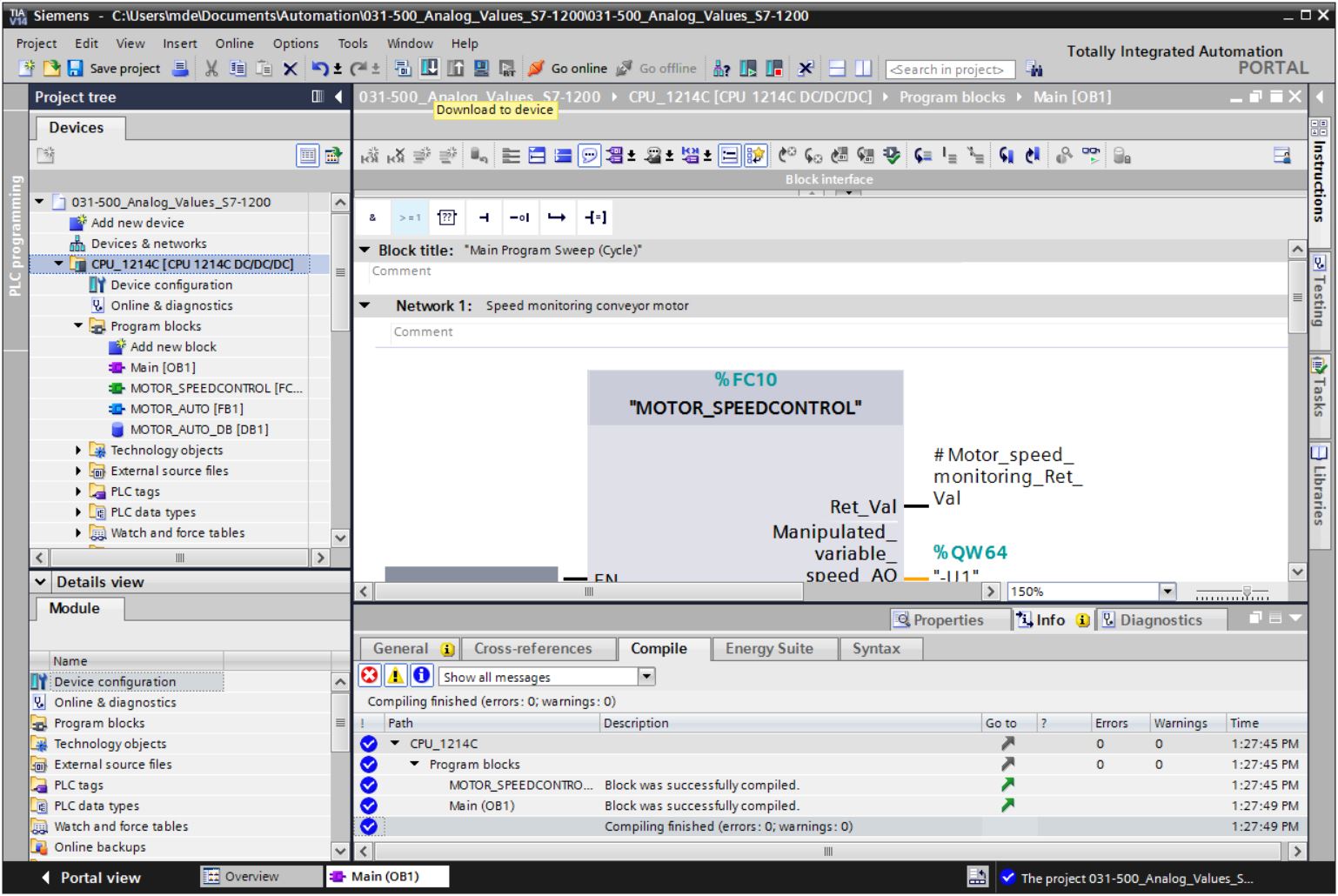


* The "Info", "Compile" area shows which blocks were successfully compiled.



## Download the program

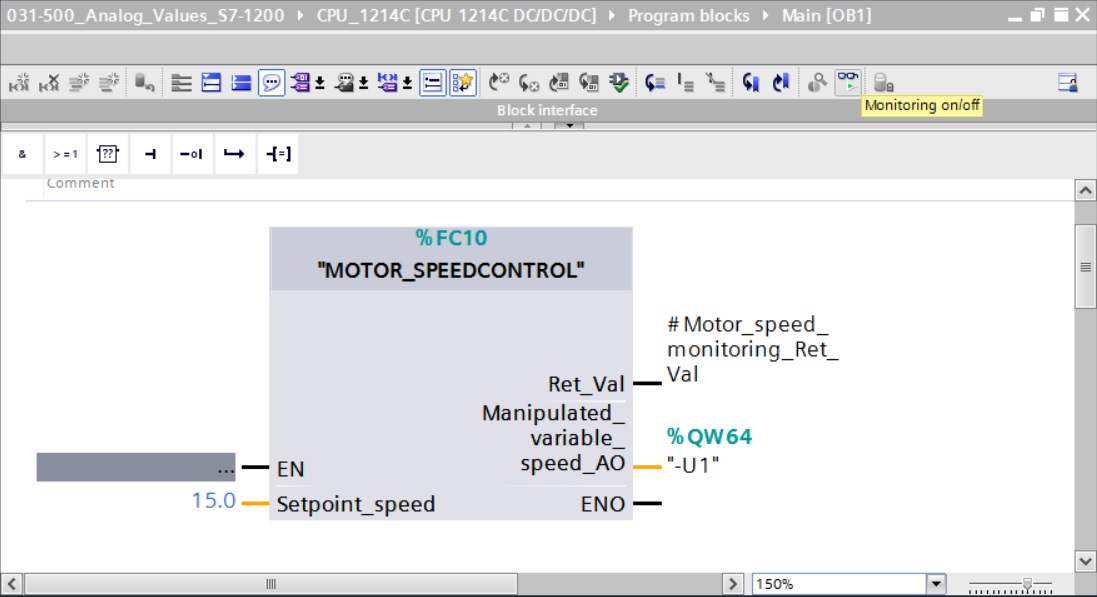
* After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded. (→ )

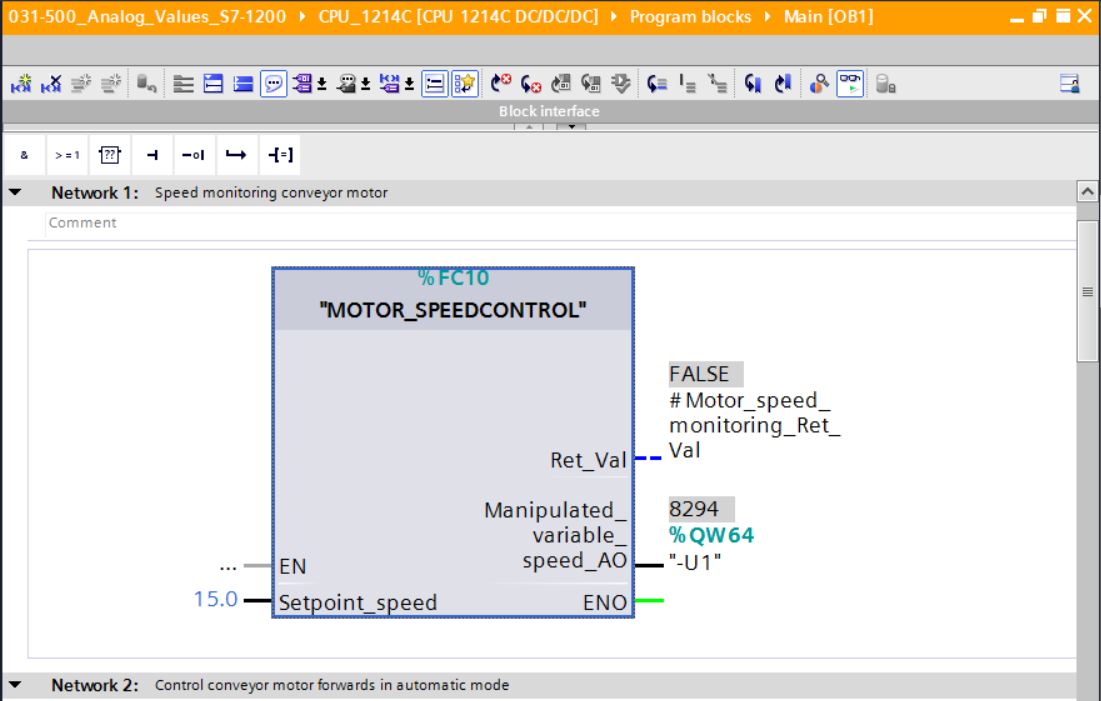


## Monitor program blocks

* The desired block must be open for monitoring the downloaded program. The monitoring can now be activated/deactivated by clicking the D:\00_DATA\SIEMENS\Unterlagen\08_Ausbildungsunterlage_TIA-Portal_R1502_dt\032-100 FC-Programmierung\pics\055b.jpg icon.

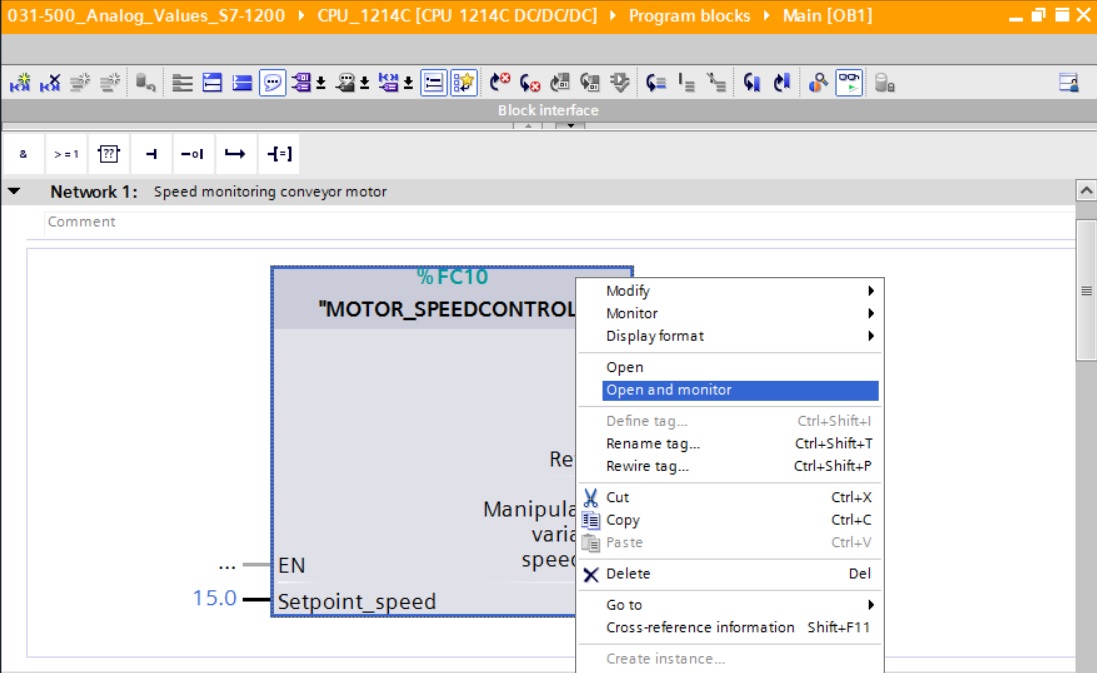
(→ Main [OB1] → D:\00_DATA\SIEMENS\Unterlagen\08_Ausbildungsunterlage_TIA-Portal_R1502_dt\032-100 FC-Programmierung\pics\055b.jpg)

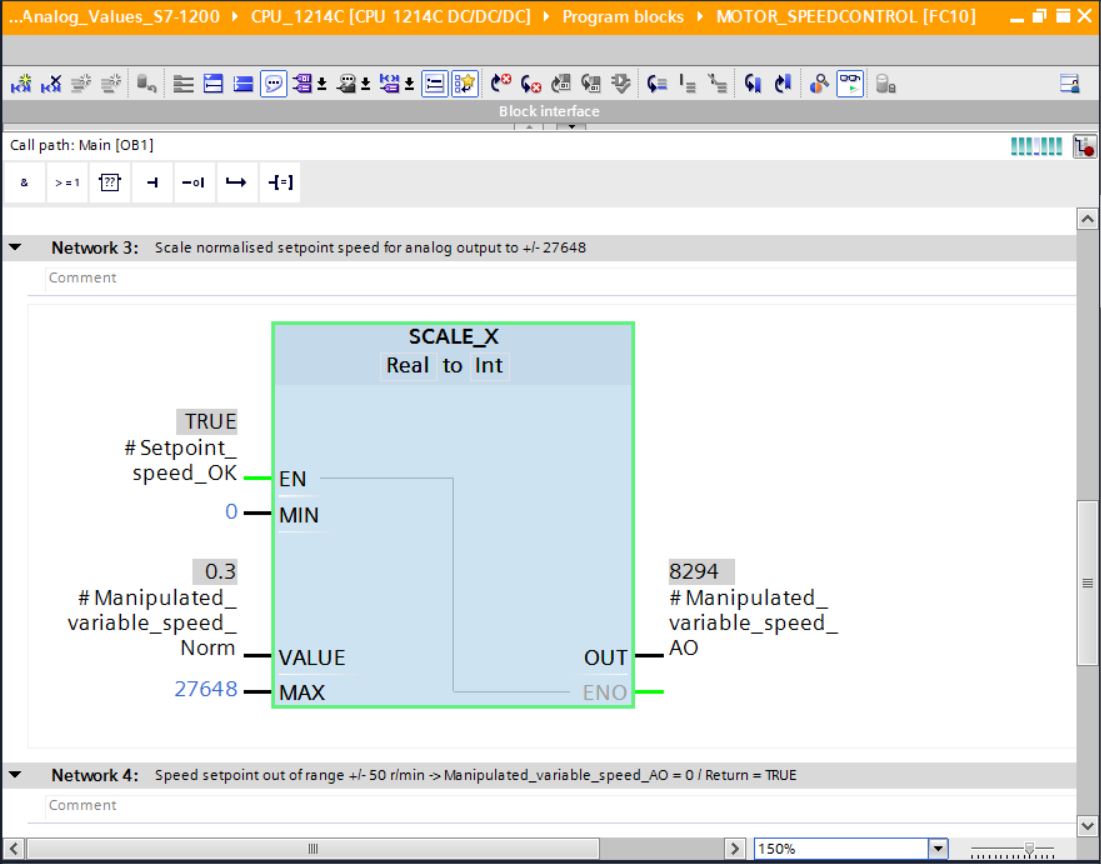




* The "MOTOR\_SPEEDCONTROL" [FC10] function called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking and the program code in the function can thus be monitored.

(→ "MOTOR\_SPEEDCONTROL" [FC10] → Open and monitor)

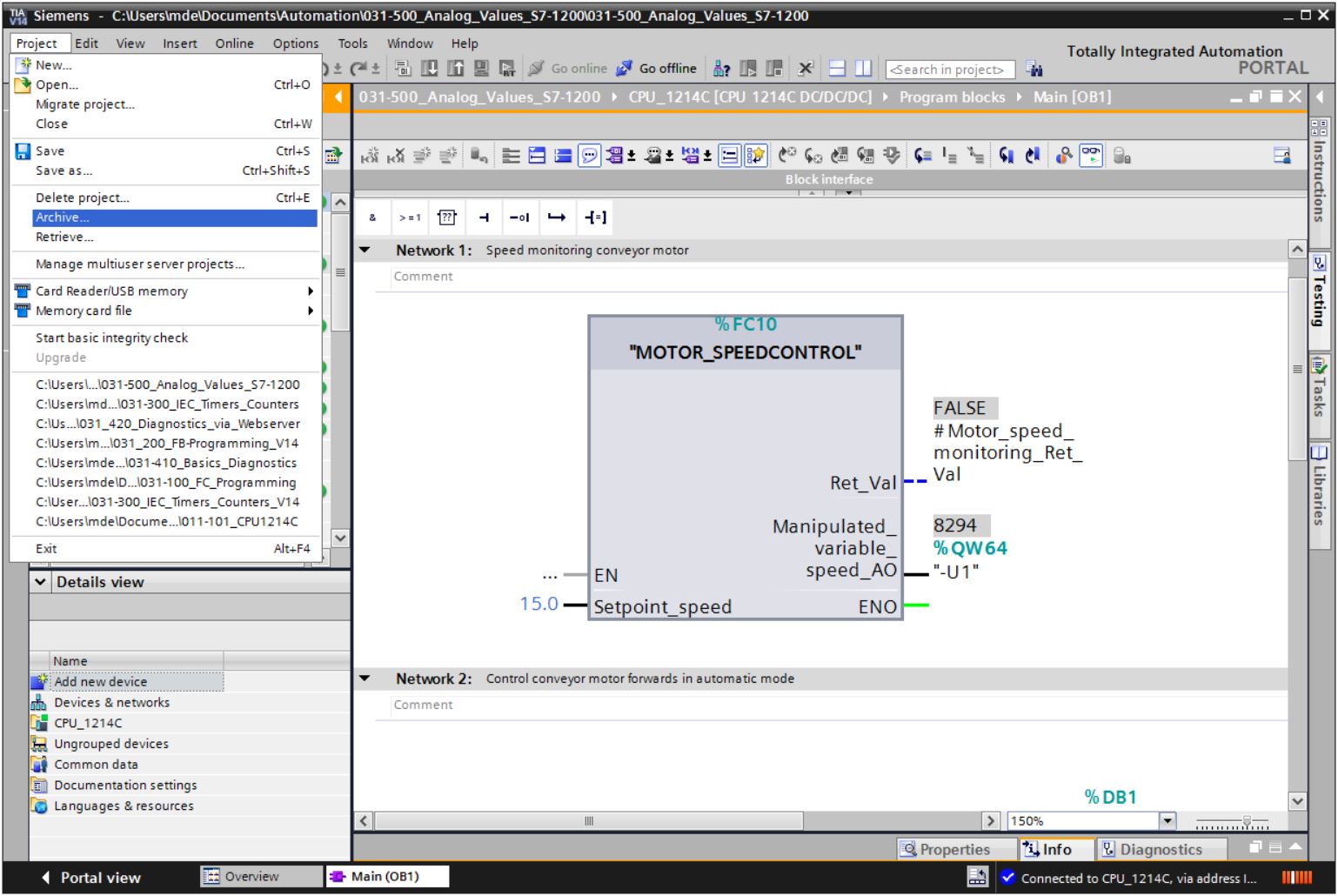




## Archive the project

* As the final step, we want to archive the complete project. Select the → 'Archive ...' command in the → 'Project' menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive".

(→ Project → Archive → TIA Portal project archive → 031-500\_Analog\_Values\_S7-1200…. → Save)



# Checklist

|  |  |  |
| --- | --- | --- |
| No. | Description | Completed |
| 1 | Compiling successful and without error message |  |
| 2 | Download successful and without error message |  |
| 3 | Switch on station (-K0 = 1)  Cylinder retracted / Feedback activated (-B1 = 1)  EMERGENCY OFF (-A1 = 1) not activated  AUTOMATIC mode (-S0 = 1)  Pushbutton automatic stop not actuated (-S2 = 1)  Briefly press the automatic start pushbutton (-S1 = 1)  Sensor part at slide activated (-B4 = 1)  then Conveyor motor -M1 variable speed (-Q3 = 1)  switches on and stays on.  The speed corresponds to the speed setpoint in the range +/- 50 rpm |  |
| 4 | Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds) |  |
| 5 | Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0 |  |
| 6 | Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0 |  |
| 7 | Manual mode (-S0 = 0) → -Q3 = 0 |  |
| 8 | Switch off station (-K0 = 0) → -Q3 = 0 |  |
| 9 | Cylinder not retracted (-B1 = 0) → -Q3 = 0 |  |
| 10 | Project successfully archived |  |

# Exercise

## Task – Exercise

In this exercise a "MOTOR\_SPEEDMONITORING" [FC11] function will be created additionally.

The actual value will be made available to B8 (Sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm) as an analog value and queried at an input of the "MOTOR\_SPEEDMONITORING" [FC11] function. The data type is 16-bit integer (Int).

This actual speed value will first be normalized to the range +/- 1 as 32-bit floating-point number (Real) in the function.

The normalized actual speed value will then be scaled to revolutions per minute (range: +/- 50 rpm) as 32-bit floating-point number (Real) and made available at an output.

The following 4 limit values can be specified as 32-bit floating-point numbers (Real) at the block inputs in order to monitor them in the function:

Speed > Motor\_speed\_monitoring\_error\_max

Speed > Motor\_speed\_monitoring\_warning\_max

Speed < Motor\_speed\_monitoring\_warning\_min

Speed < Motor\_speed\_monitoring\_error\_min

If a limit value is exceeded or fallen below, the value TRUE (1) is assigned to the corresponding output bit.

If a fault is present, the protective tripping of the "MOTOR\_AUTO" [FB1] function block will be tripped.

## Technology diagram

Here you see the technology diagram for the task.



Figure 3: Technology diagram



Figure 4: Control panel

## Reference list

The following signals are required as global operands for this task.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DI** | **Type** | **Identifier** | **Function** | **NC/NO** |
| I 0.0 | BOOL | -A1 | Return signal emergency stop OK | NC |
| I 0.1 | BOOL | -K0 | Main switch "ON" | NO |
| I 0.2 | BOOL | -S0 | Mode selector manual (0)/ automatic (1) | Manual = 0  Auto = 1 |
| I 0.3 | BOOL | -S1 | Pushbutton automatic start | NO |
| I 0.4 | BOOL | -S2 | Pushbutton automatic stop | NC |
| I 0.5 | BOOL | -B1 | Sensor cylinder -M4 retracted | NO |
| I 1.0 | BOOL | -B4 | Sensor part at slide | NO |
| I 1.3 | BOOL | -B7 | Sensor part at end of conveyor | NO |
| IW64 | BOOL | -B8 | Sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DO** | **Type** | **Identifier** | **Function** |  |
| Q 0.2 | BOOL | -Q3 | Conveyor motor -M1 variable speed |  |
| QW 64 | BOOL | -U1 | Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm |  |

***Legend for reference list***

|  |  |
| --- | --- |
| DO | Digital Output |
| AO | Analog Output |
| Q | Output |

|  |  |
| --- | --- |
| DI | Digital Input |
| AI | Analog Input |
| I | Input |
| NC | Normally Closed |
| NO | Normally Open |

## Planning

Plan the implementation of the task on your own.

## Checklist – Exercise

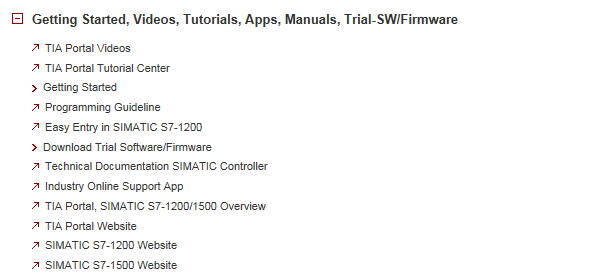
|  |  |  |
| --- | --- | --- |
| No. | Description | Completed |
| 1 | Compiling successful and without error message |  |
| 2 | Download successful and without error message |  |
| 3 | Switch on station (-K0 = 1)  Cylinder retracted / Feedback activated (-B1 = 1)  EMERGENCY OFF (-A1 = 1) not activated  AUTOMATIC mode (-S0 = 1)  Pushbutton automatic stop not actuated (-S2 = 1)  Briefly press the automatic start pushbutton (-S1 = 1)  Sensor part at slide activated (-B4 = 1)  then Conveyor motor M1 variable speed (-Q3 = 1)  switches on and stays on.  The speed corresponds to the speed setpoint in the range +/- 50 rpm |  |
| 4 | Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds) |  |
| 5 | Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0 |  |
| 6 | Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0 |  |
| 7 | Manual mode (-S0 = 0) → -Q3 = 0 |  |
| 8 | Switch off station (-K0 = 0) → -Q3 = 0 |  |
| 9 | Cylinder not retracted (-B1 = 0) → -Q3 = 0 |  |
| 10 | Speed > Motor\_speed\_monitoring\_error\_max → -Q3 = 0 |  |
| 11 | Speed < Motor\_speed\_monitoring\_error\_min → -Q3 = 0 |  |
| 12 | Project successfully archived |  |

# Additional information

More information for further practice and consolidation is available as orientation, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software / firmware, under the following link:

[www.siemens.com/sce/s7-1200](http://www.siemens.com/sce/s7-1200)

Preview „Additional information“



Further Information

Siemens Automation Cooperates with Education  
siemens.com/sce

SCE Learn-/Training Documents  
**siemens.com/sce/documents**

SCE Trainer Packages  
**siemens.com/sce/tp**

SCE Contact Partners   
**siemens.com/sce/contact**

Digital Enterprise  
**siemens.com/digital-enterprise**

Industrie 4.0   
**siemens.com/future-of-manufacturing**

Totally Integrated Automation (TIA)  
**siemens.com/tia**

TIA Portal  
**siemens.com/tia-portal**

SIMATIC Controller  
**siemens.com/controller**

SIMATIC Technical Documentation   
**siemens.com/simatic-docu**

Industry Online Support  
**support.industry.siemens.com**

Product catalogue and online ordering system Industry Mall   
**mall.industry.siemens.com**

Siemens AG  
Digital Factory   
P.O. Box 4848  
90026 Nuremberg  
Germany

Subject to change and errors  
© Siemens AG 2018

**siemens.com/sce**