OPERATION MANUAL

2511
PROFINET Integration into TIA Portal

Manufacturer:
burster
präzisionsmesstechnik gmbh & co kg
Talstr. 1 - 5 P.O. Box 1432
76593 Gernsbach 76587 Gernsbach
Germany Germany

Tel.: +49-7224-645-0
Fax.: +49-7224-645-88
Email: info@burster.com
www.burster.com

Valid from: **22.04.2025**
Applies to: **2511-VXXX**

4678-BA2511PROFINETTIAEN-5799-041529

Table of Contents

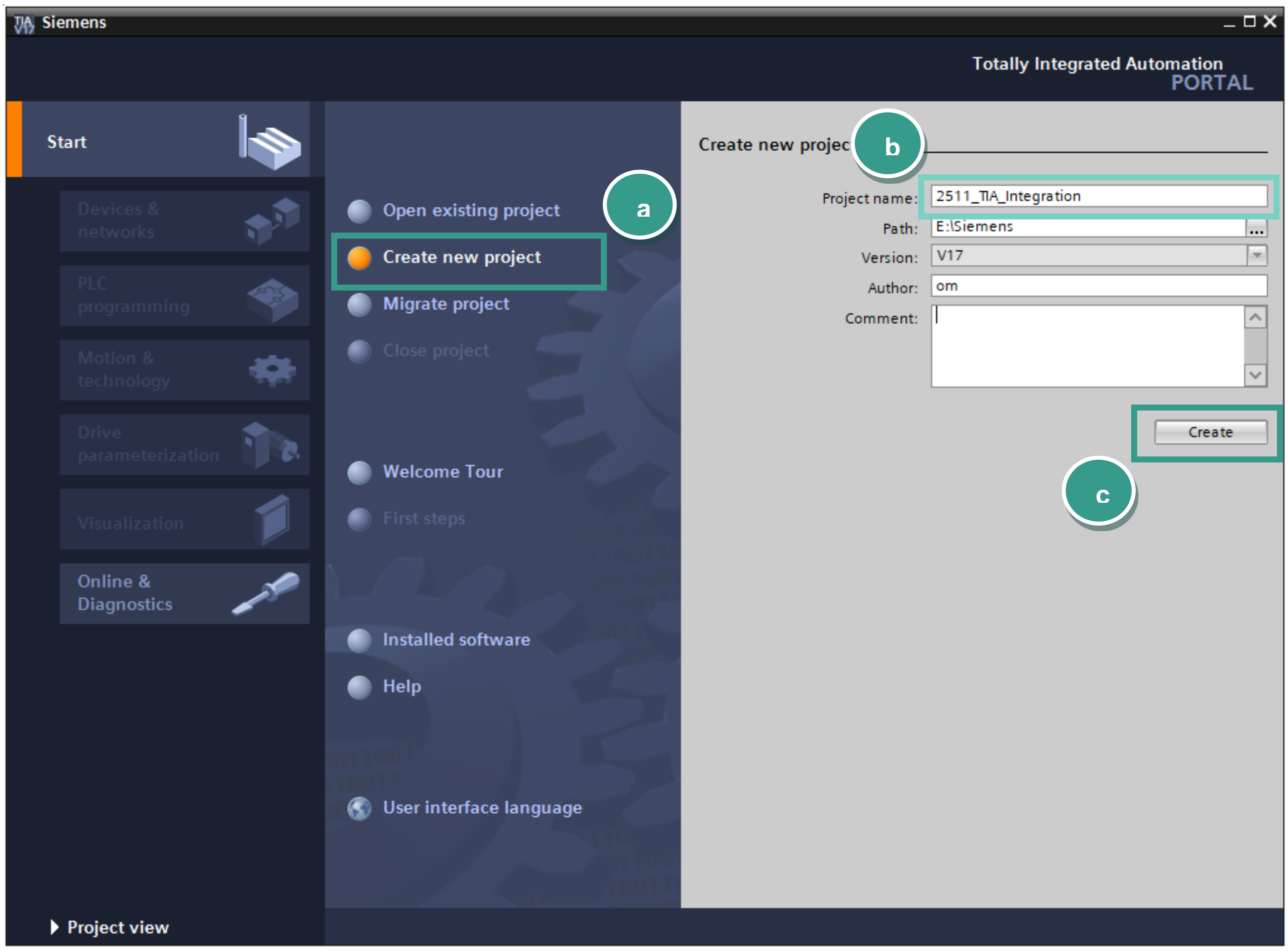
1. Introduction

This quick start guide describes an approach how you can configure the 2511 via TIA Portal using the example of S7-1511 CPU. Please note that the samples here cannot be directly used in your production line because they have beed extremely simplified to reach a better understanding. Therefore, you may have to complete them by checking of status, error, length values etc.

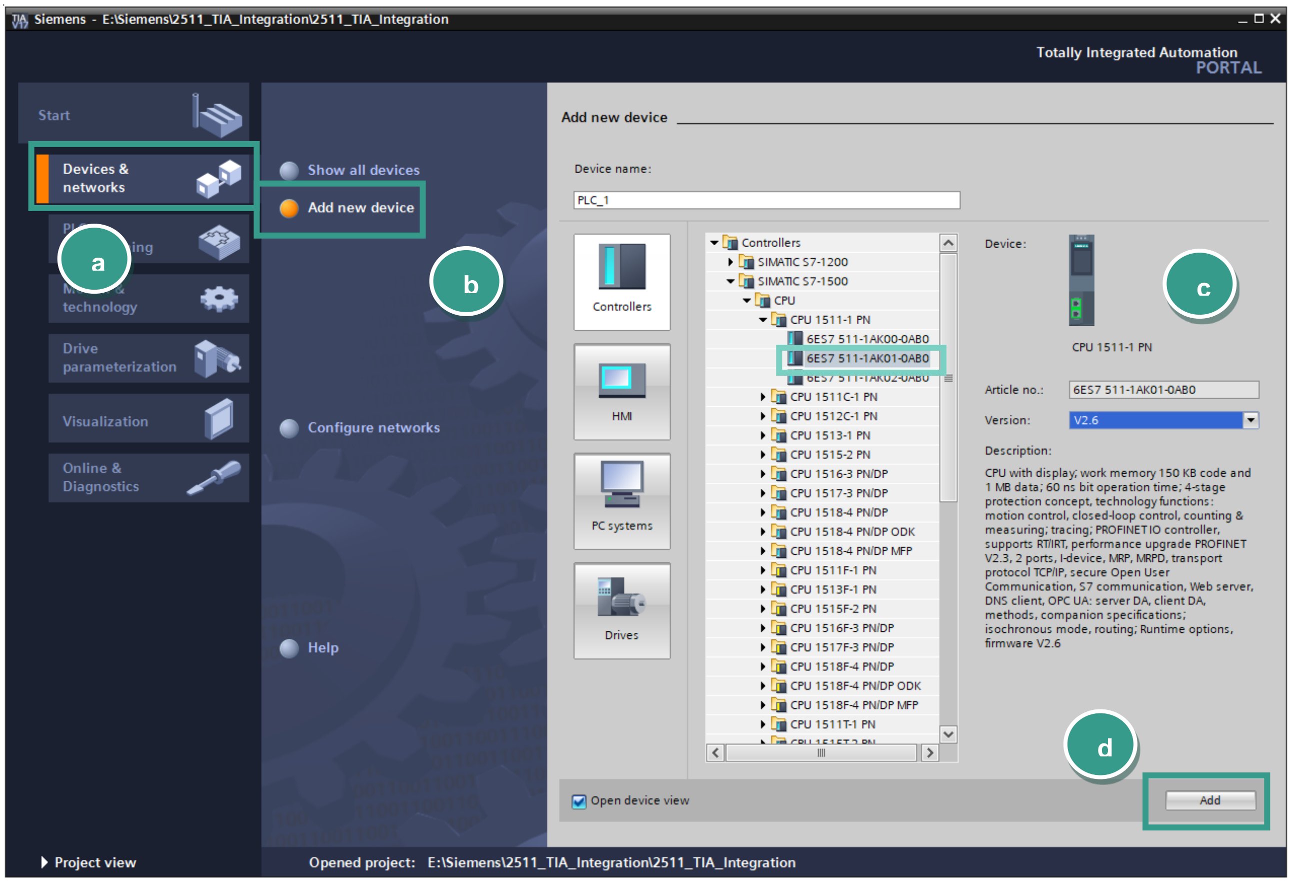
!	NOTE
	Please also note that you will have to use the 2511 PROFINET manual to get futher information about input and output parameters (cyclic as well acyclic data transfer).

2. Creating new project

- Start the **Totally Integrated Automation Protal**, select ***Create New Project*** (a), assign the project a name (b) and click ***Create*** (c):



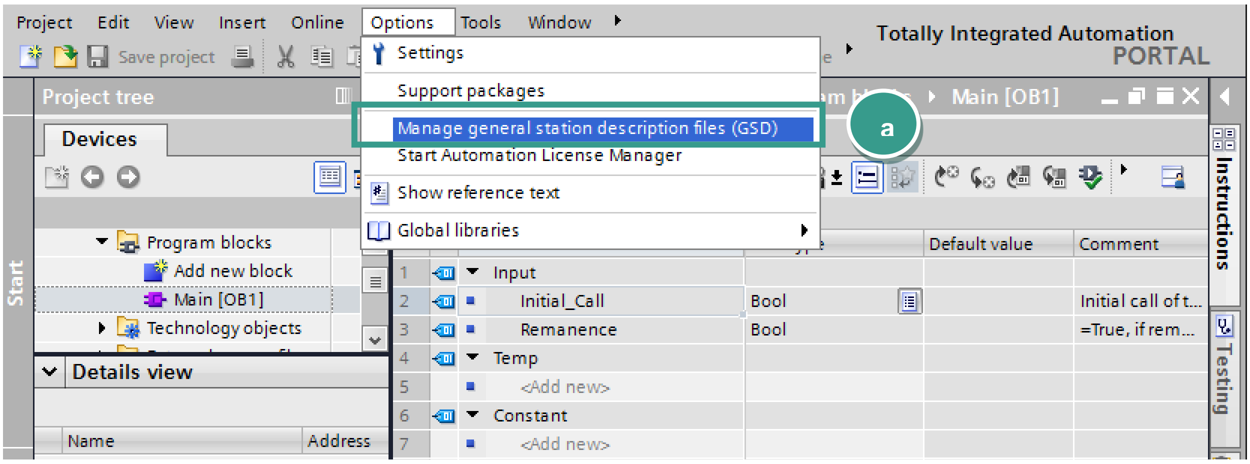
- Go to ***Devices & networks*** (a) on the left side select ***Add new device*** *(b)* and look for your CPU (c). Afterwards click the **Add** button (d).



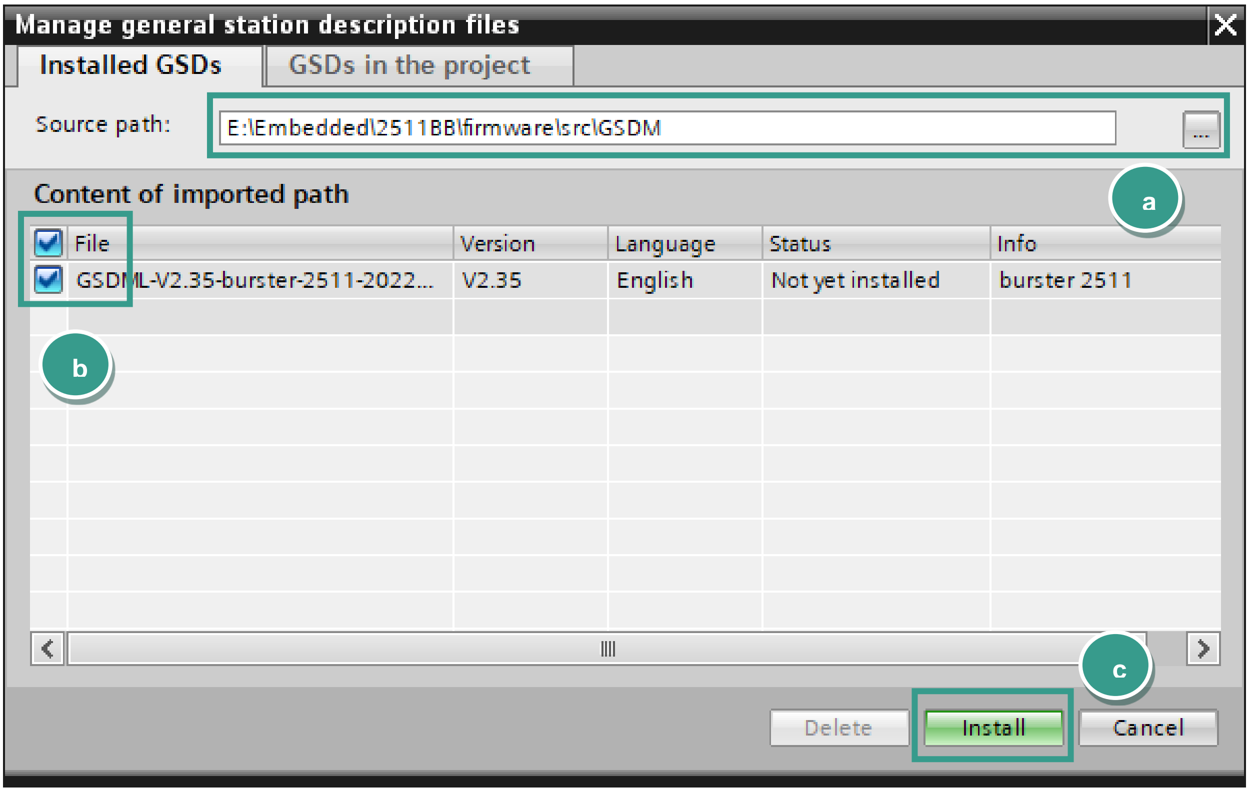
3. Installation of GSDML files

Note: Please make sure that your GSDML file is compatible to the field bus firmware in the 2511. The latest GSDML file is available for download on www.burster.com/en/download-area.Also for compatibility reasons, uninstall all previous GSDML files of particular device if you have any!

- Go to Options → Manage general station description files (GSD)



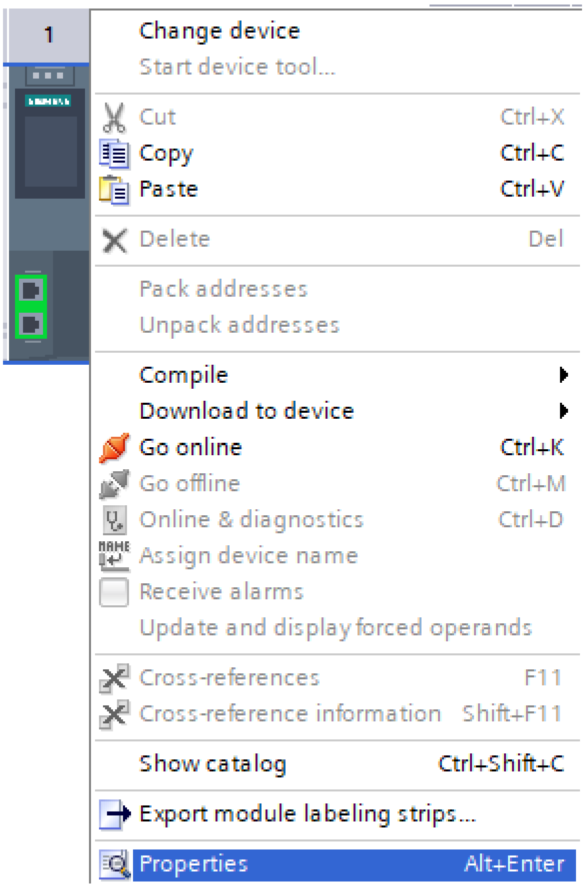
- Navigate to your 2511 GSDML directory (a)(you will find the GSD files on www.burster.com/en/download-area select the GSDML file (b) and click ***Install*** (c)

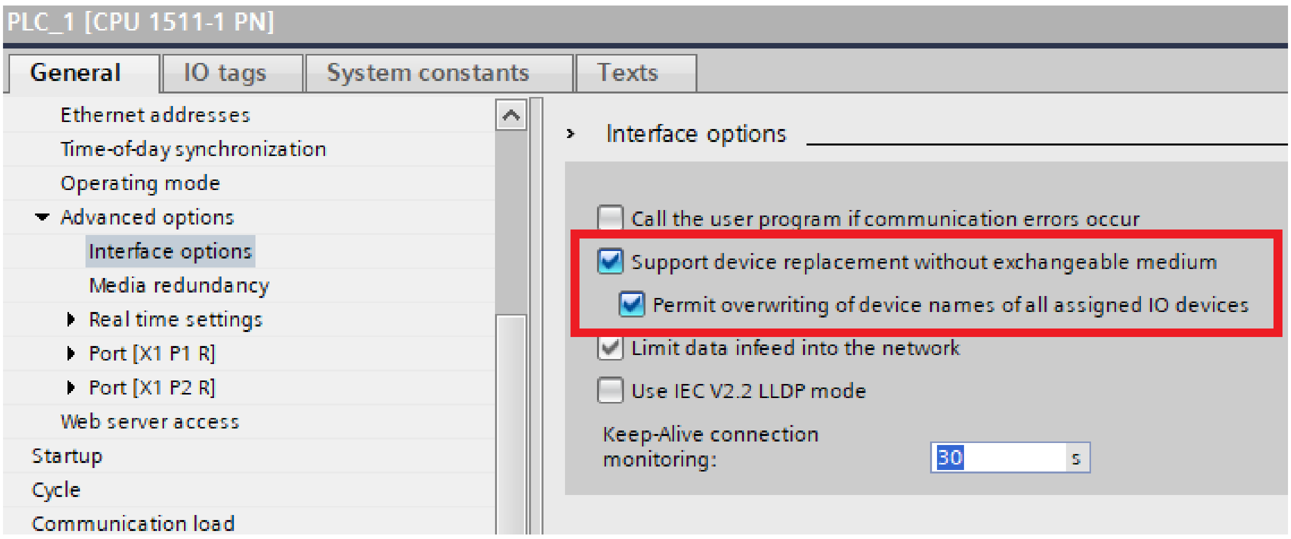


4. Device replacement without exchangeable medium

To be able to exchange a device with another one later without setting the PROFINET station name of the new device manually you can allow overwriting of device names by the PLC-Controller.

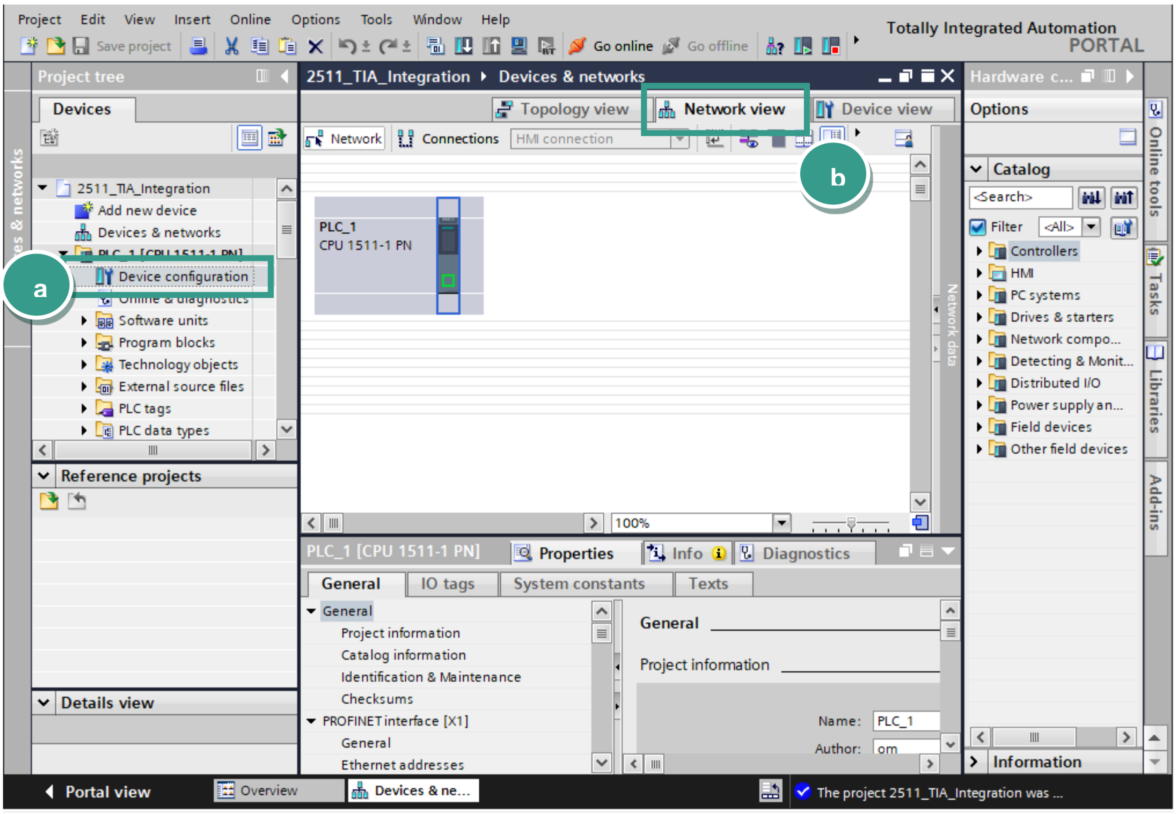
Go to the the properties of your PLC Controller and then to *Interface options* and *Permit overwriting of device names…*



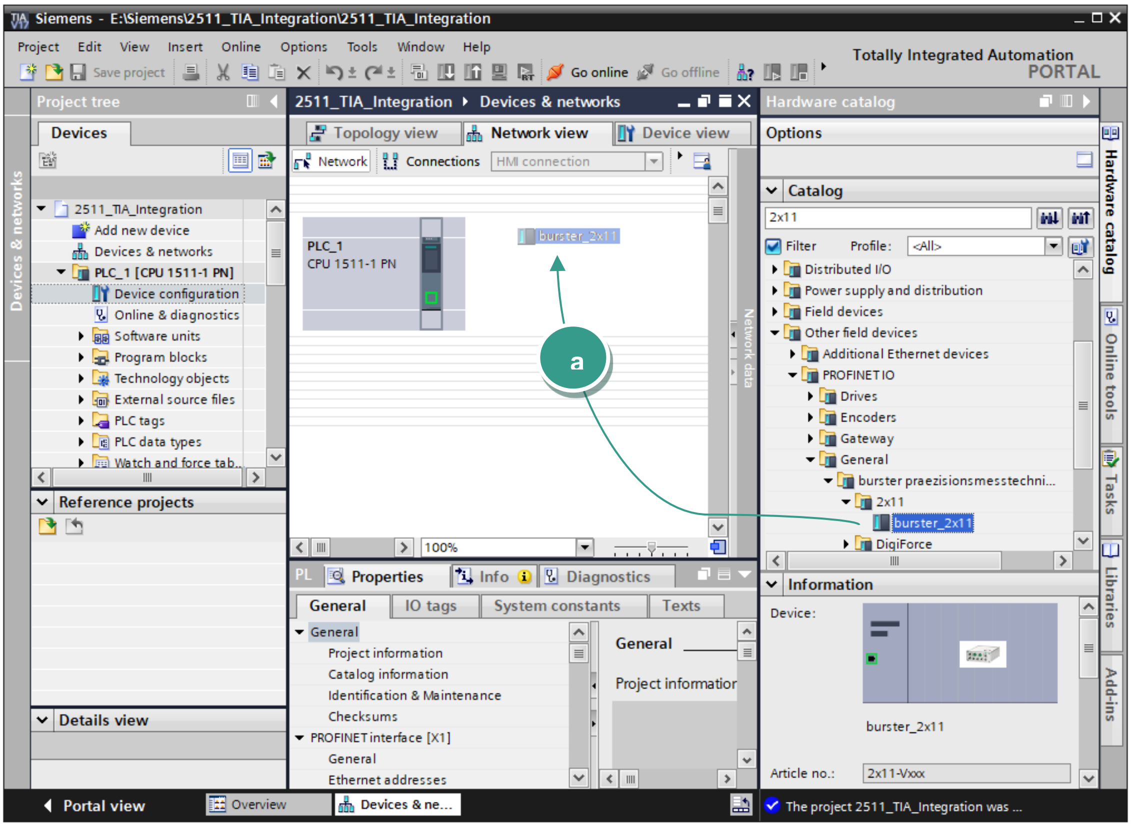


5. Creation of network connections

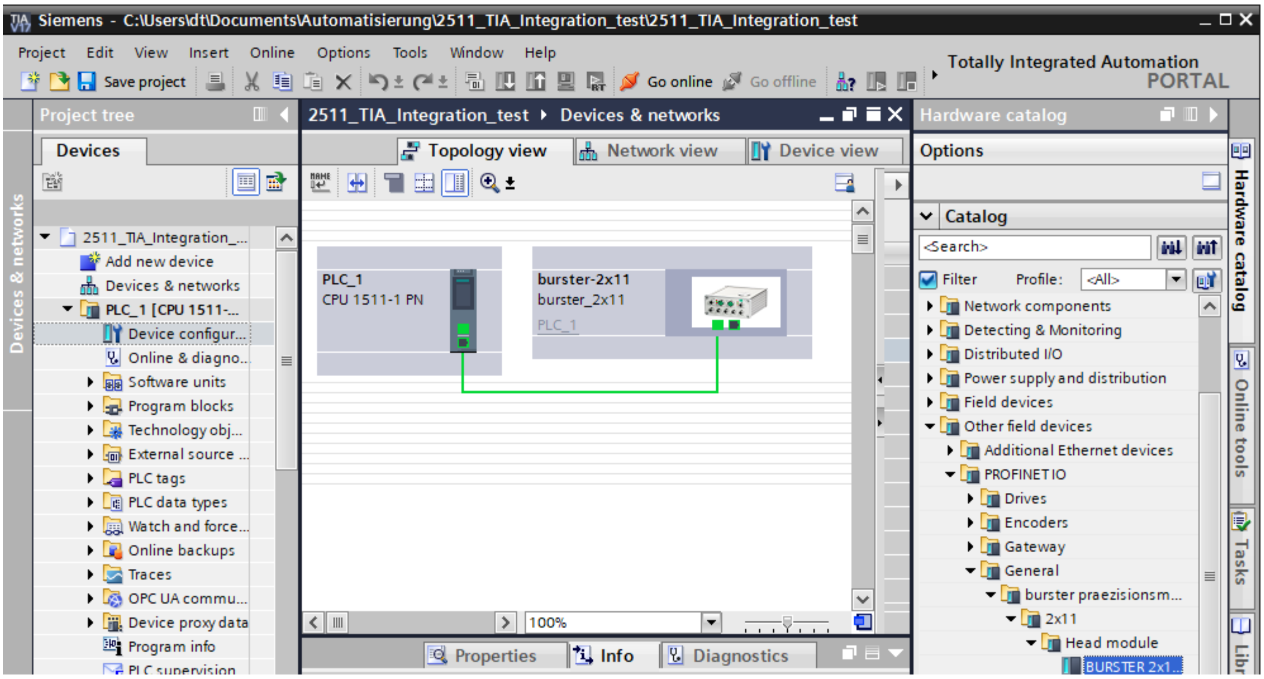
- Double click ***Device Configuration*** (a) in the project tree und switch to **Network view** (b) :



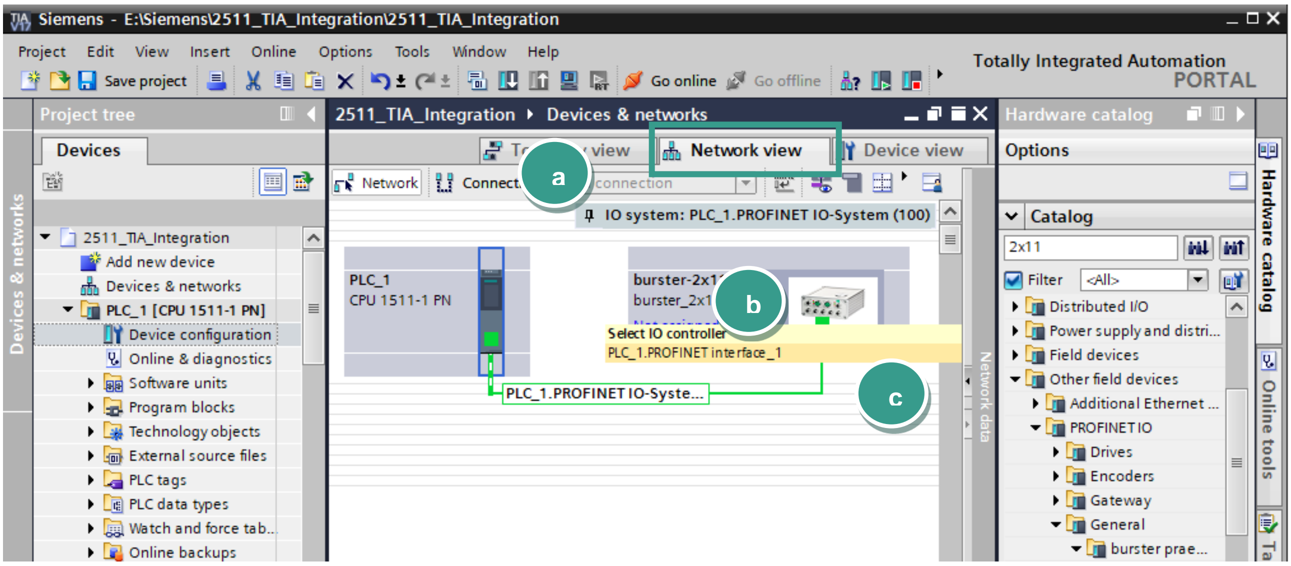
- Now select the **burster_2x11** device in the catalog and drag & drop it into the working area (a):



- Select an Ethernet port on the S7 and hold the left mouse button down to connect the S7 with 2511:

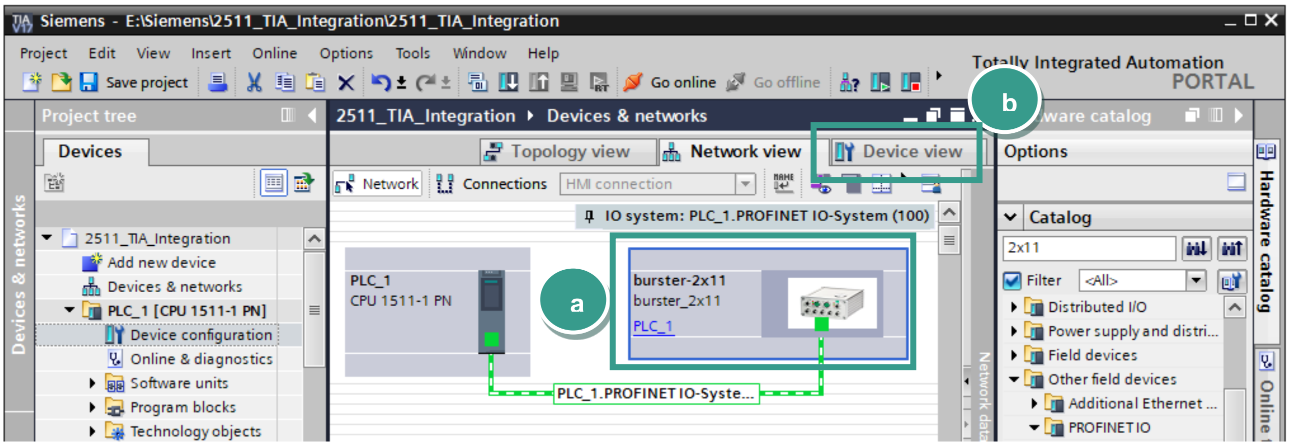


- If device is not automatically assigned, change now to **Network view** (a) to assign a controller to the 2511. Click on the link "Not assigned" (b) of 2x11 and select your controller (c):

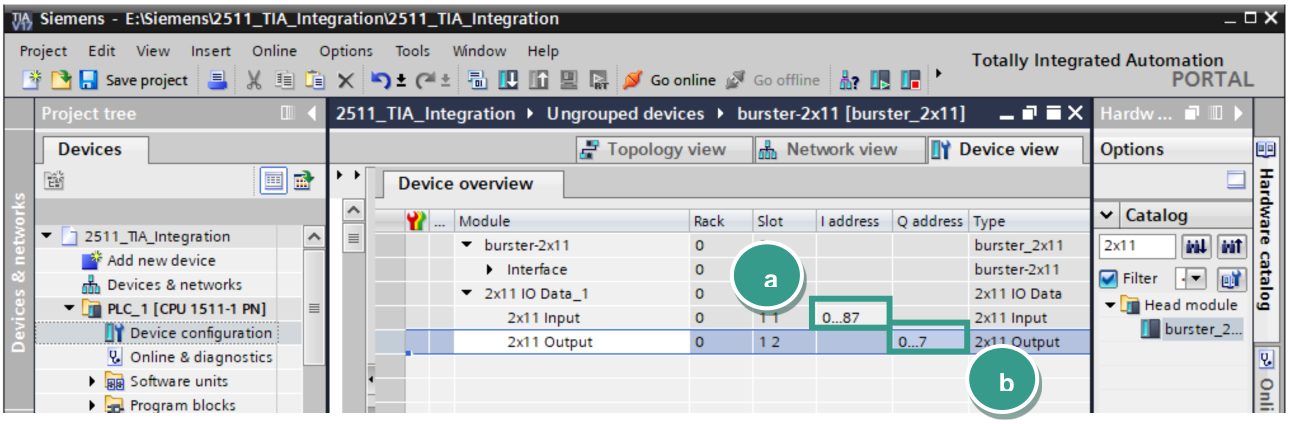


Note: Check if devices also connected physically to the right ports. You find the port number assignment directly on the device front side.

- Next, click on the *burster-2x11* device (a) and then switch to **Device view** (b)



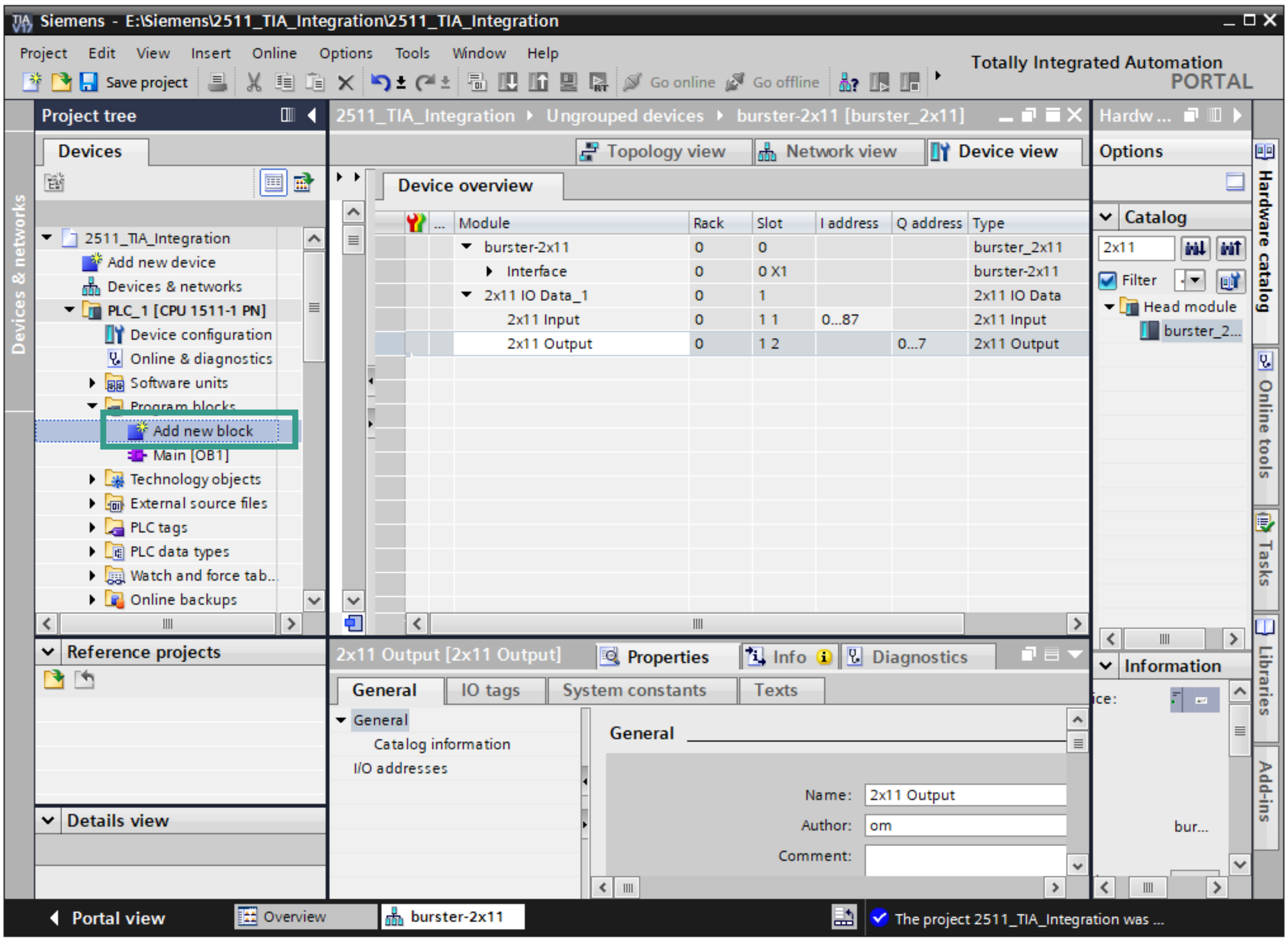
- Make a note of Input (a) and Output (b) addresses



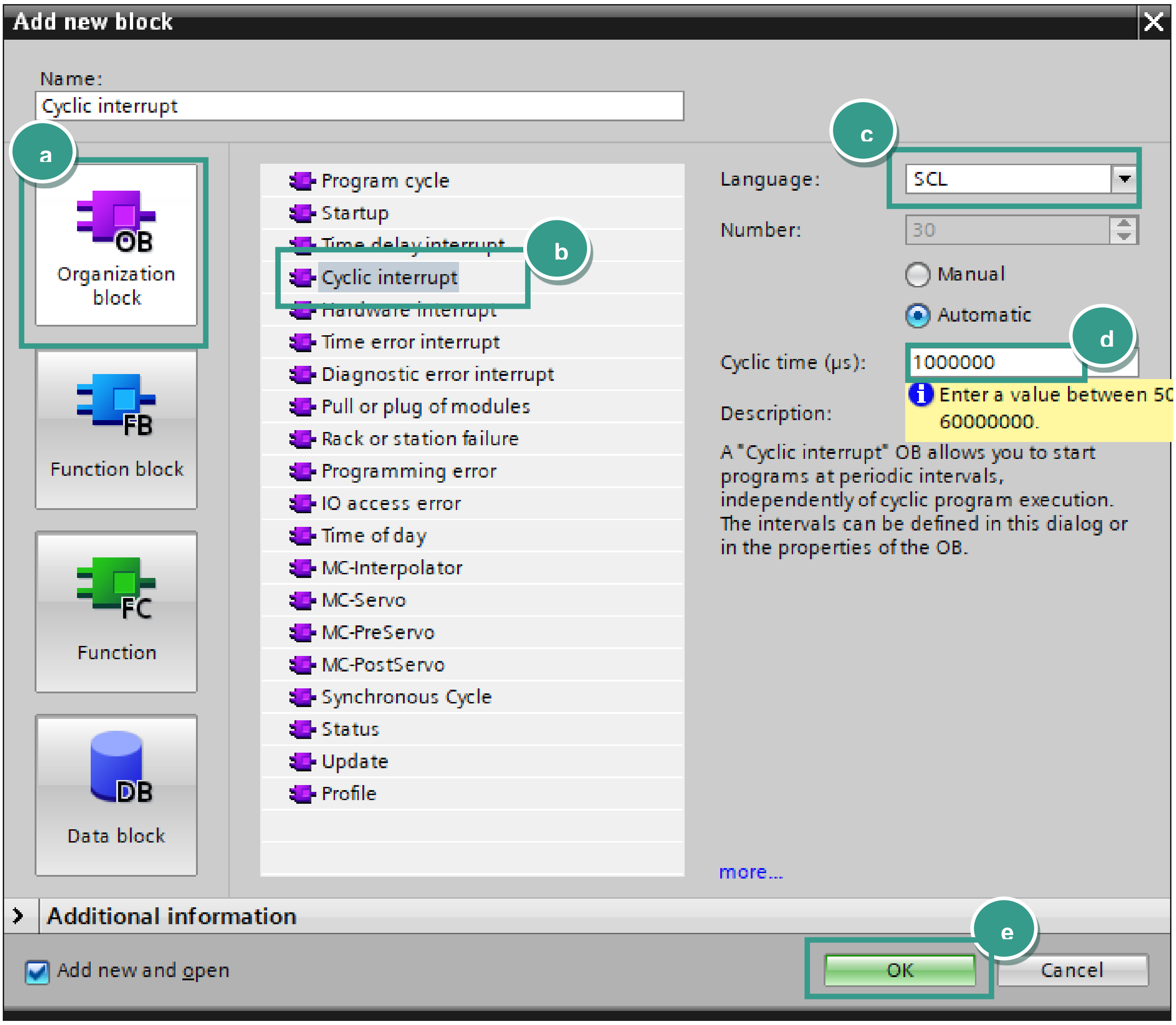
6. Create a sample program

In this section, you will learn how to create a simple program to start a measurement process and read the measurement value from 2511. You will need to refer to section *PROFINET data protocol* of the **2511 ProfiNet Interface Manual** to understand the meaning of input and output bytes.

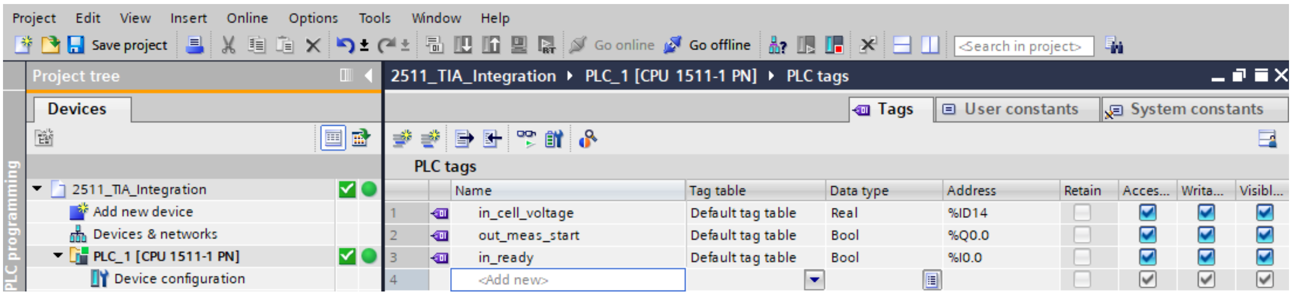
- Expand the tree node ***Program blocks*** in the **Project tree** and double click ***Add new block:***



- Select in the new window ***Organization block*** (a) and then ***Cyclic interrupt*** (b). As language set SCL (c), change the cyclic time to 1.000.000 µs (d) and click OK (e):



- Declare 3 new variables in_cell_voltage, out_meas_start and in_ready under PLC Tags:

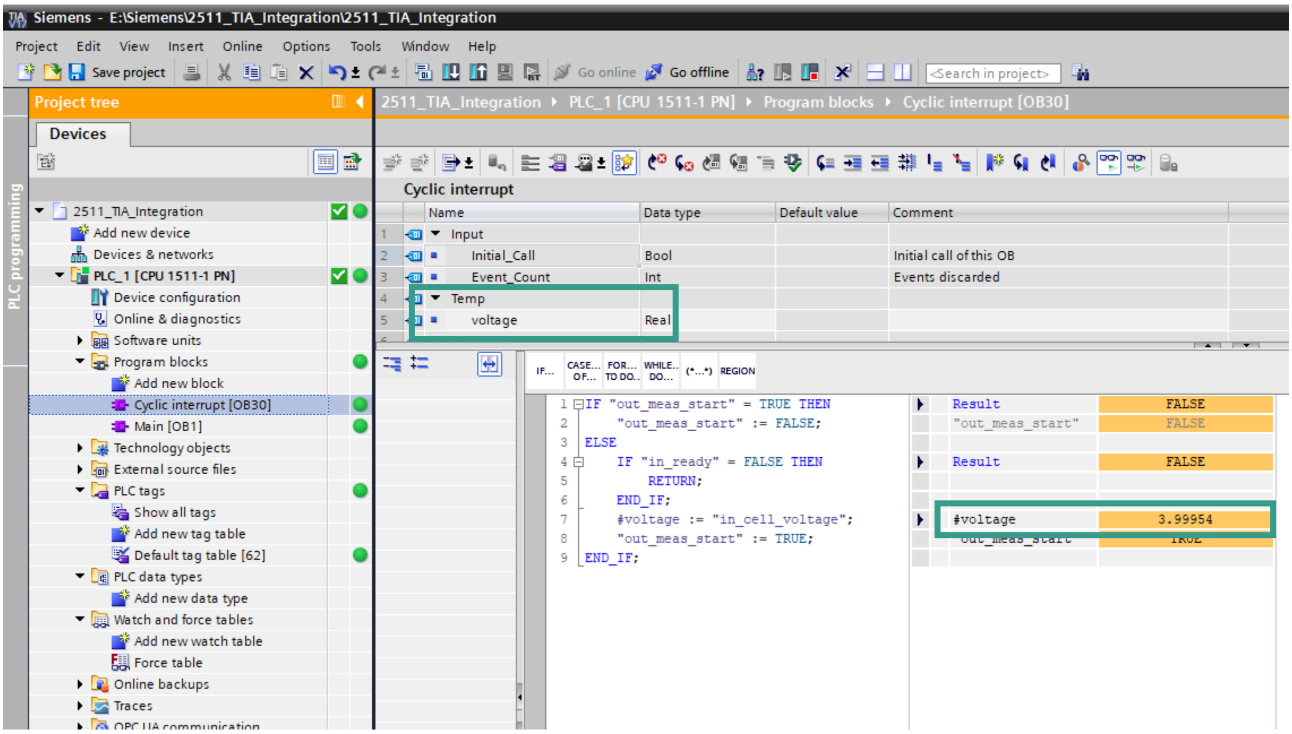


Note: Please use your note from Chapter ***Creation of network connections*** of Input and Output addresses, as they can be diffentent in your PLC. Alternatively check them in the Device view → Device overview of the 2511. The meaning of the addresses is described in *2511 Profinet Interface Manual.*

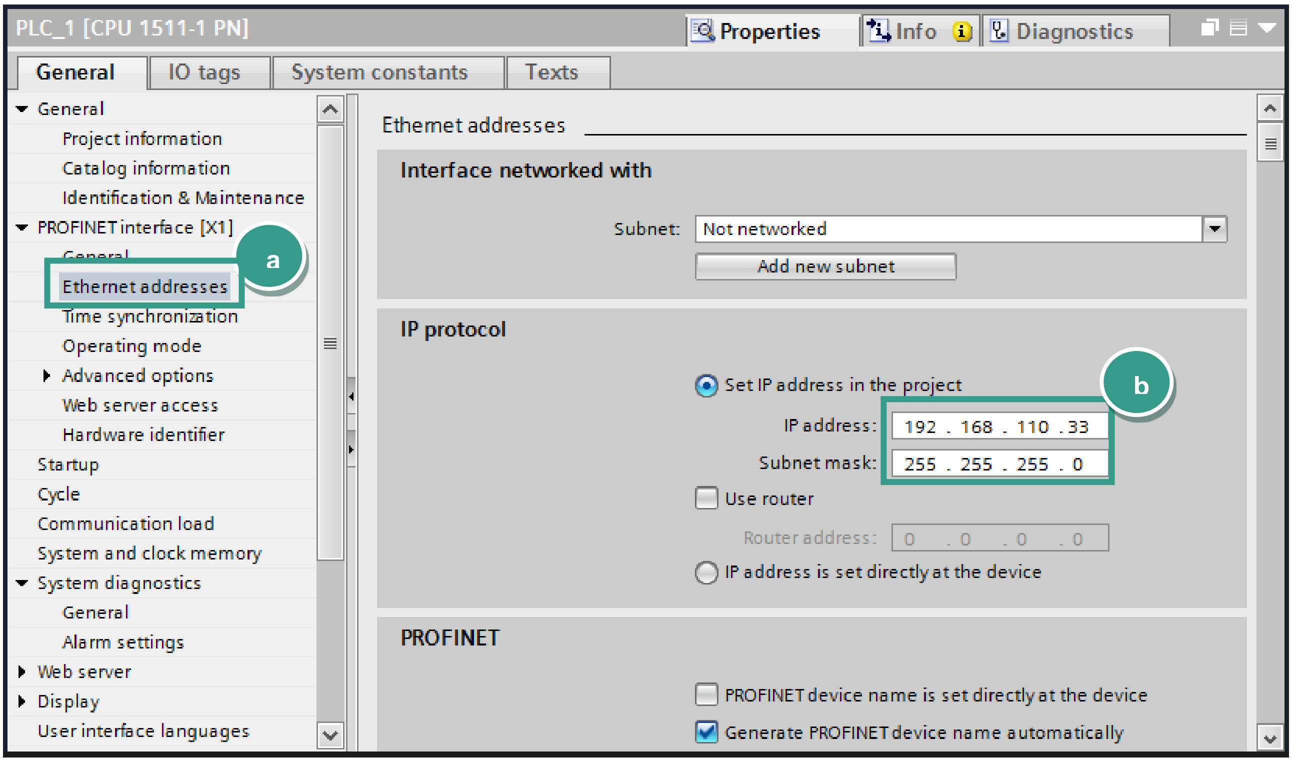
- Then type in the following source code in the code field of the new block:

```
IF "out_meas_start" = TRUE THEN             // is the output bit 'measuremet start set'
   "out_meas_start" := FALSE;               // then reset ist
ELSE                                        // is devide ready
   IF "in_ready" = FALSE THEN               // if not, return
      RETURN;
   END_IF;
#voltage := "in_cell_voltage";              // read and strore voltage value
"out_meas_start" := TRUE;                   // start a new measurement
END_IF;
```

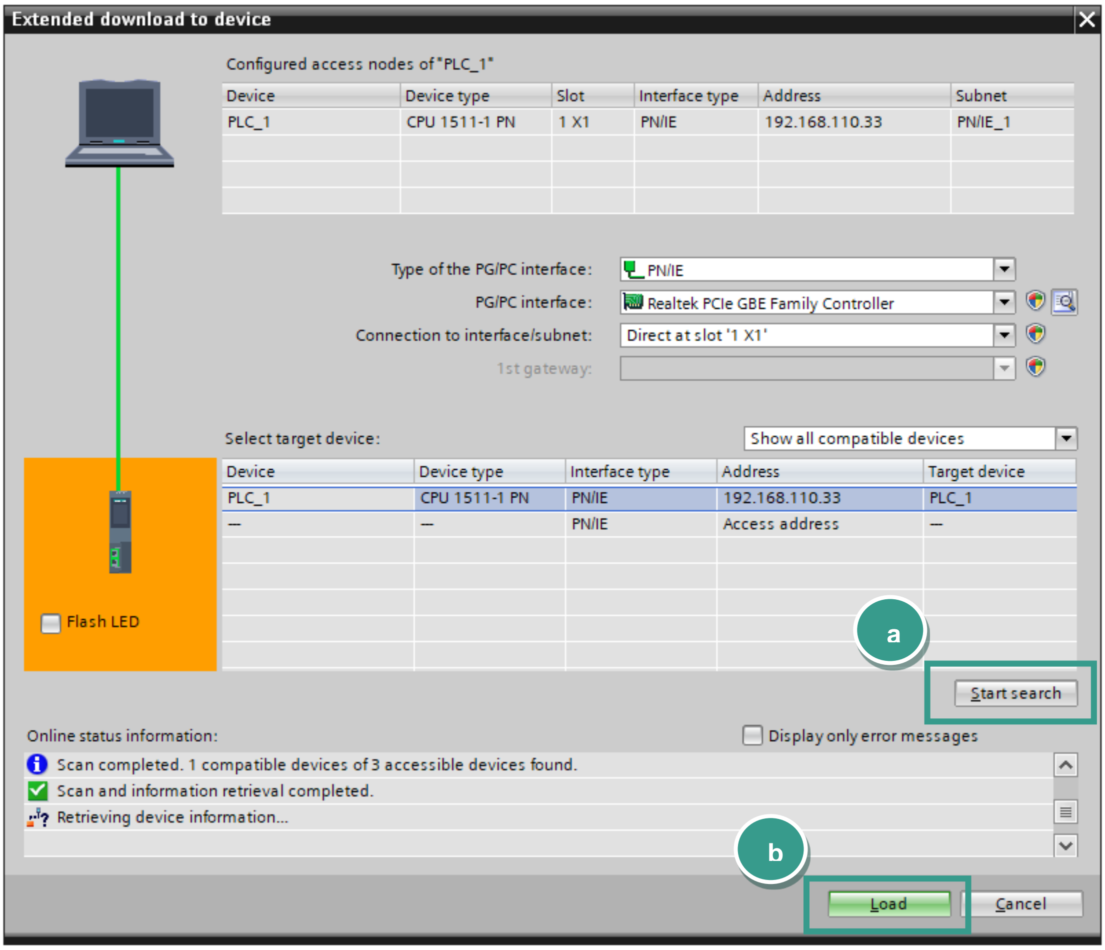
Note: You only need the '**voltage**' variable if your PLC supports debug mode, so you can watch the measured voltage directly in the source code, like this:



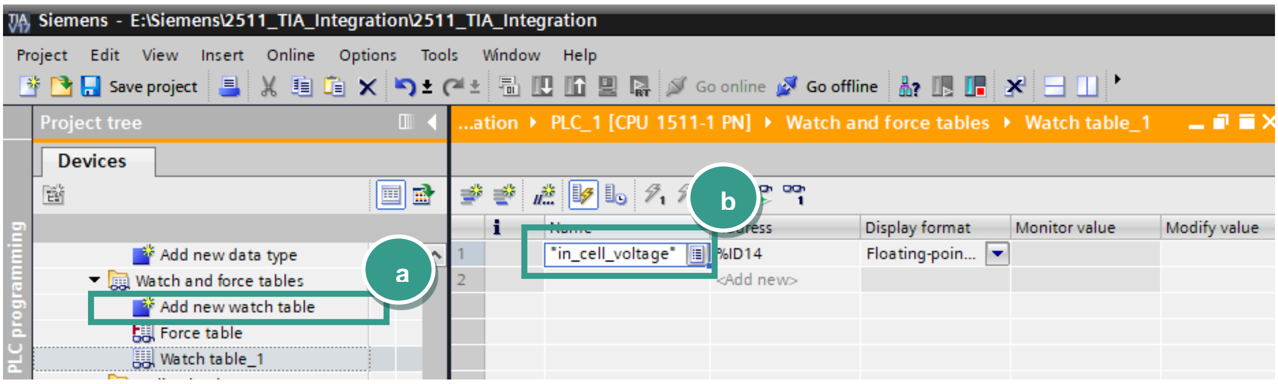
- Before you load the project into the CPU you have to set the IP address of your CPU. To do this please go to **Device view** and select ***Ethernet addresses*** (a) in ***General*** tab. Set now the IP-Address and a subnet mask(b) for your PLC:



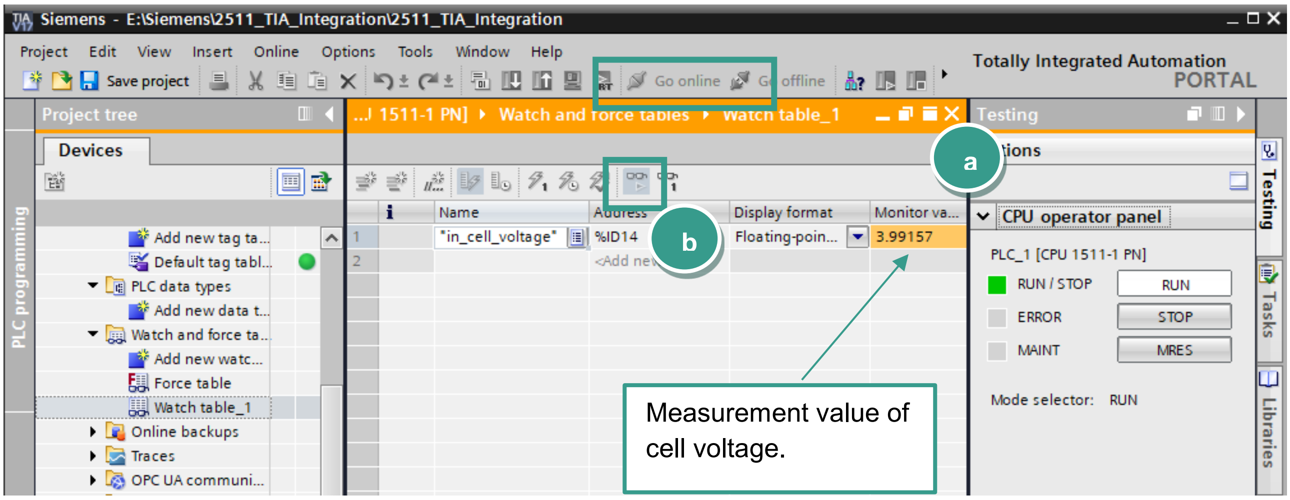
- To load the configuration into the CPU select it first go to ***Online*** → ***Download to device*** and click on ***Start search*** (a) to look for your controller. Then select the controller and click on ***Load*** (b):



- To watch the measurement value, go to **Watch and force tables** → **Add new watch table** (a) and add the variable **in_cell_voltage** to the table (b):

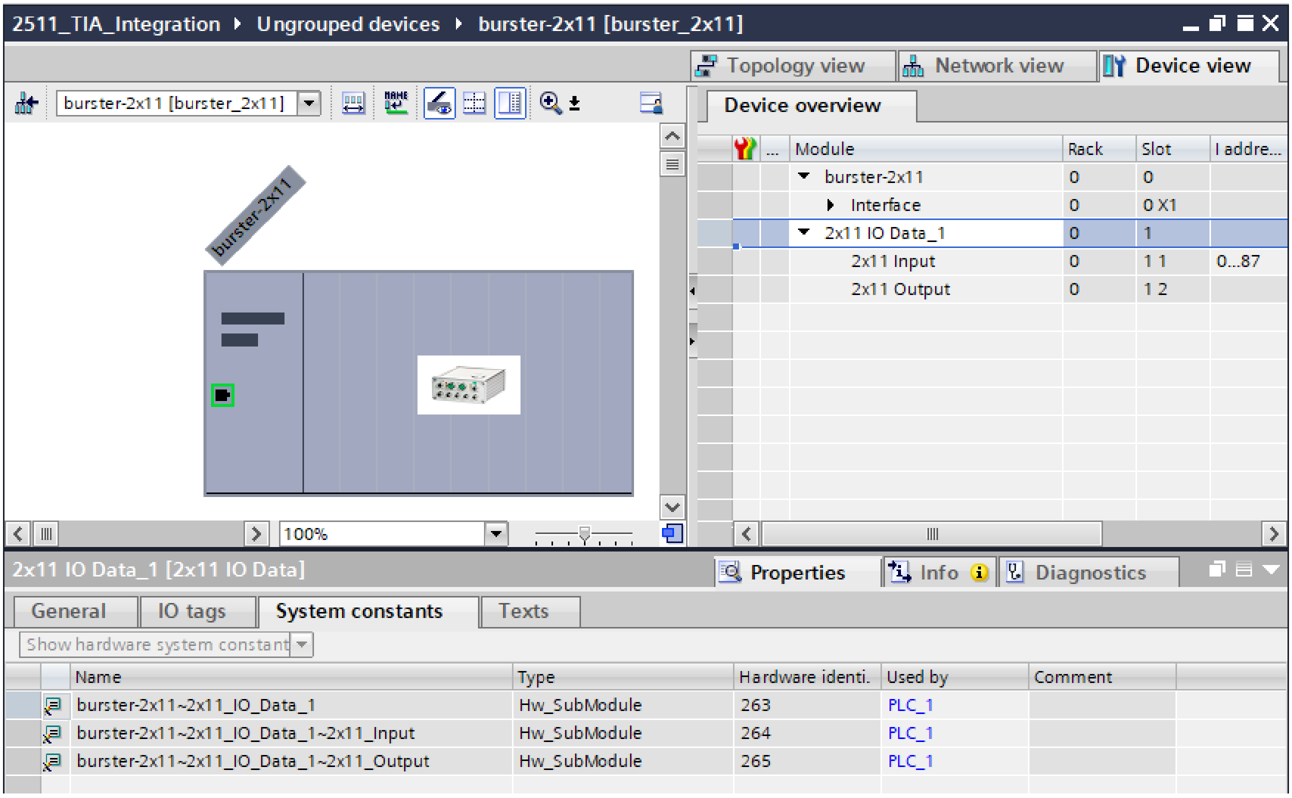


- Now click on "**Go online**" (a) and then "**Monitor all**" (b) to watch the value of **in_cell_voltage**

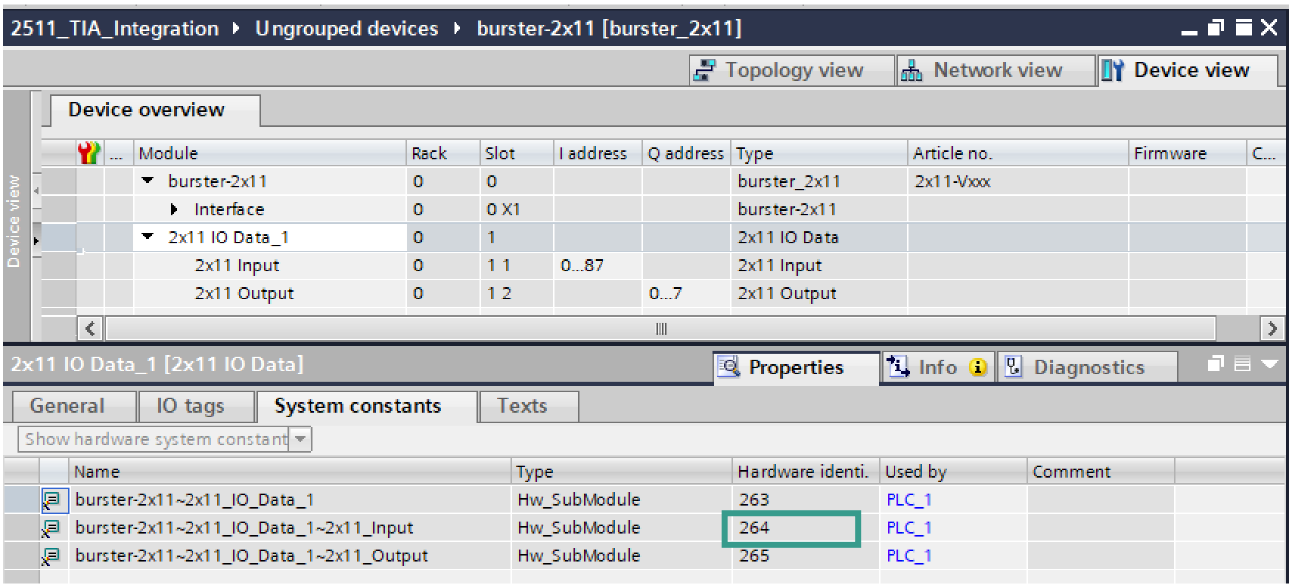


7. Further Examples

In the followed examples, a *Hardware-ID* is used to access a certain slot. To find this, please select a **burster-2x11** device in **Topology view** or **Network view** and then switch to **Device view**. Click with the right mouse button on the desired module, e.g. *BusCoupler Data extended* and select **Properties**:



- You will see the hardware identifier in the tab **System constants**:

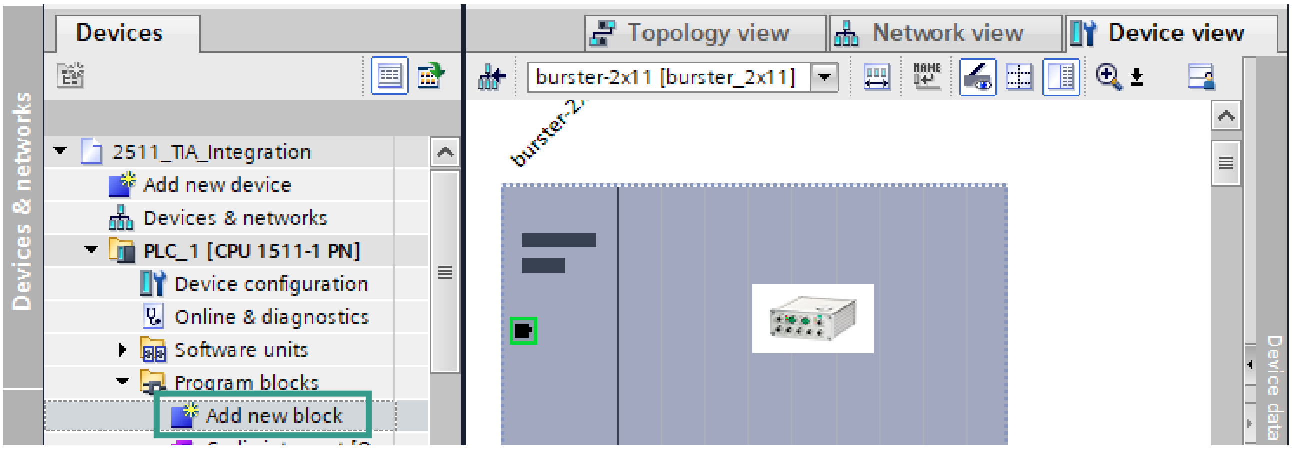


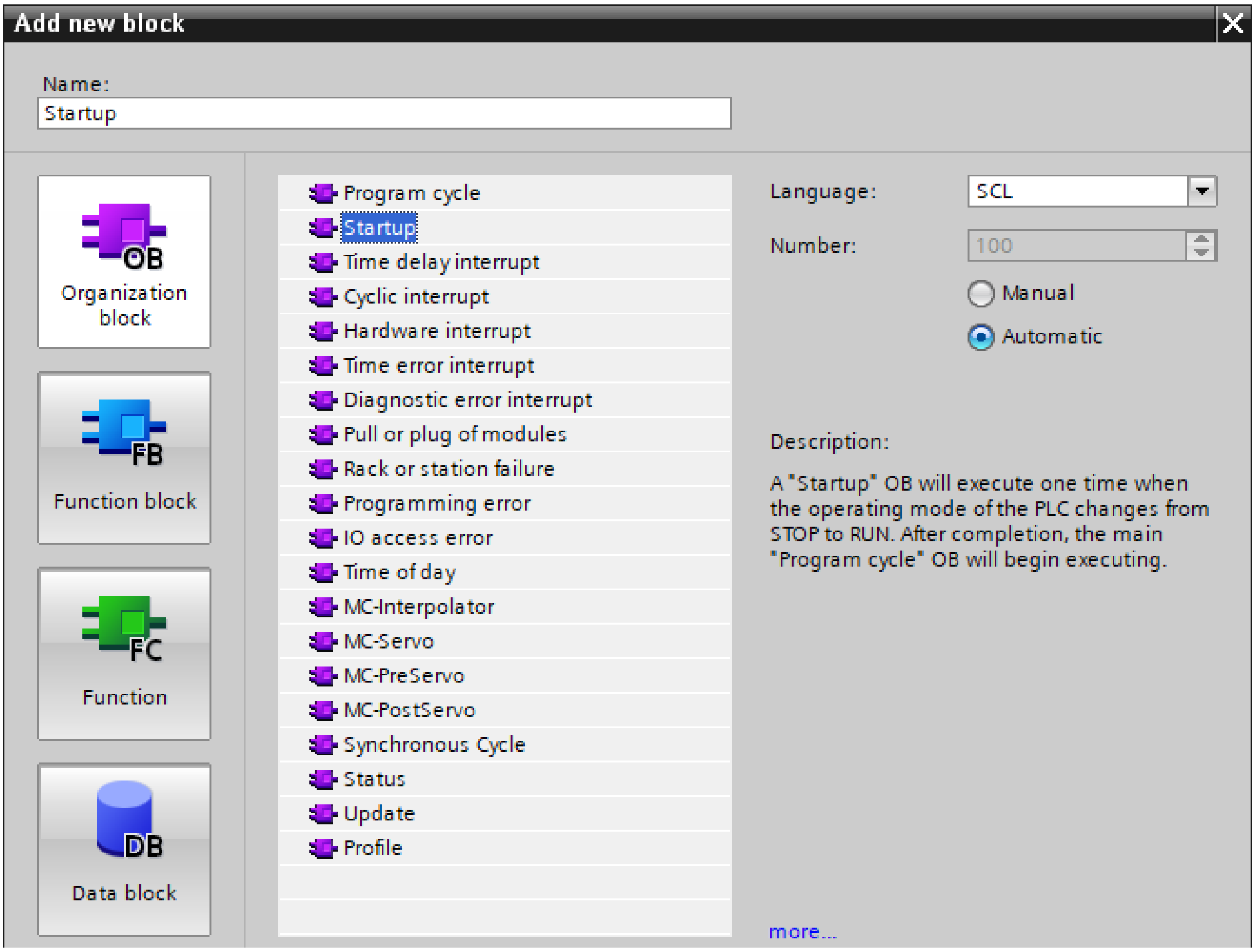
7.1. Reading of ‘string’ data types

Example 1: Reading the serial number of the digital board of 2511

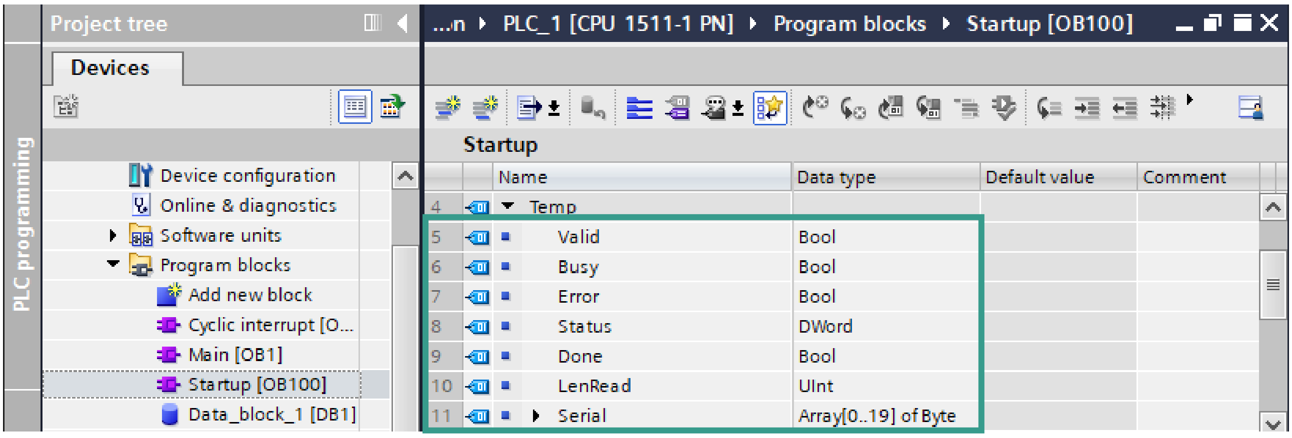
In this example, we perform a read access on index 11 to get the serial number of 2511. For these acyclic operations, you will need an instance of a RDREC block.

Add a **Startup block** to the Program blocks using **Add new block**:

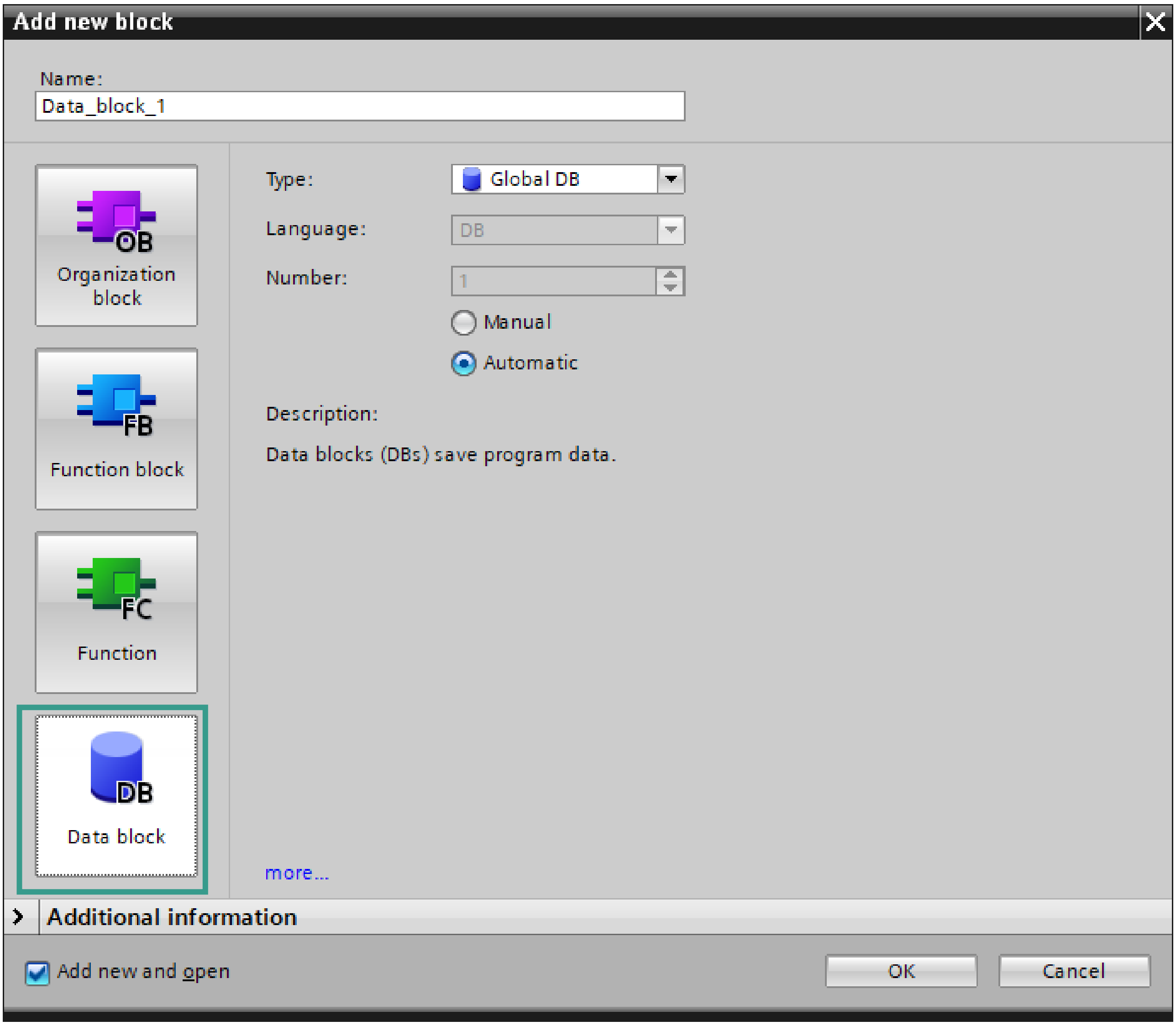




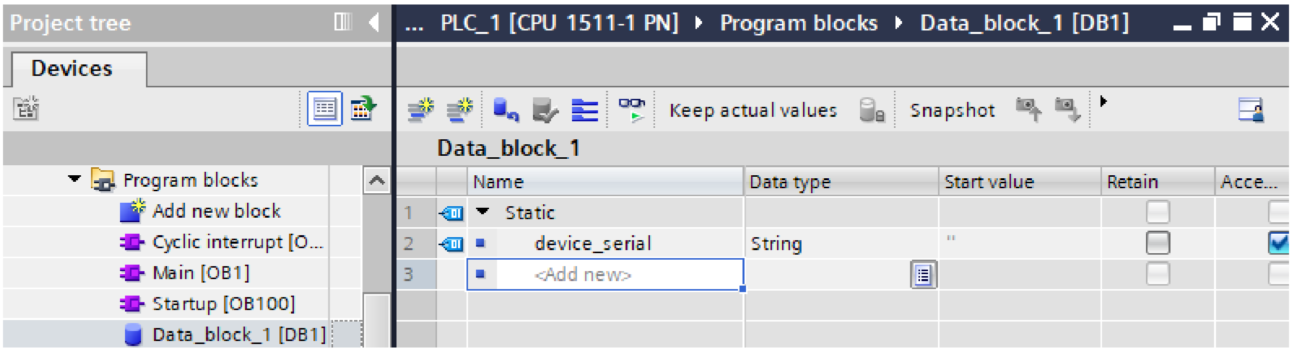
- Add variables to the **Startup** block:



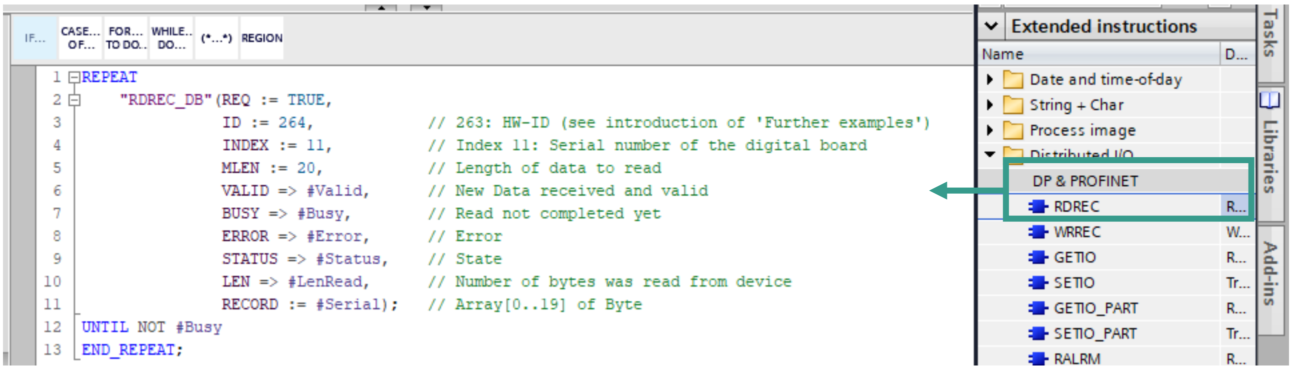
- Then add a new **Data block:**



- In addition, insert the variable **device_serial** into this new block:



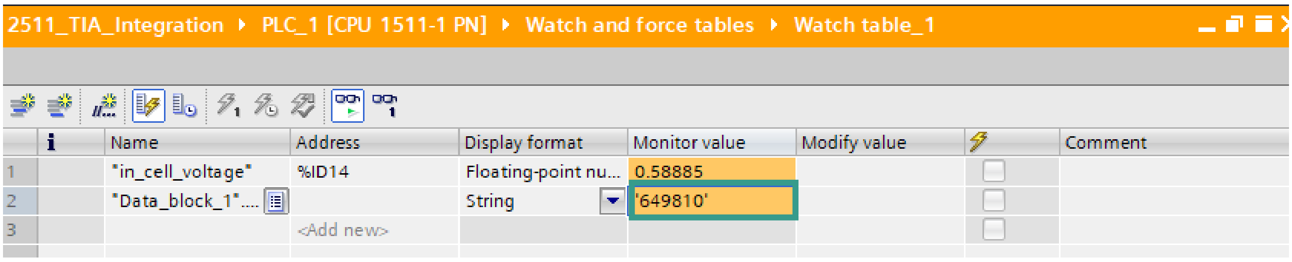
- Drag & Drop the **RDREC** instruction into the editor field and use the following source code to get the serial number from the device:



Source code:

```
REPEAT
"RDREC_DB"(REQ:=TRUE,
        ID:=264,                // 264: HW-ID (see introduction of 'Further examples')
        INDEX:=11,              // Index 11: Serial number of the digital board
        MLEN:=20,               // Length of data to read
        VALID=>#Valid,          // New Data received and valid
        BUSY=>#Busy,            // Read not completed yet
        ERROR=>#Error,          // Error
        STATUS=>#Status,        // State
        LEN=>#lenRead,          // Number of bytes was read from device
        RECORD:= #serial);      // Array[0..19] of Byte
UNTIL NOT #Busy
END_REPEAT;

IF #Error = TRUE OR #Status <> 0 THEN
  RETURN;
END_IF;

Chars_TO_Strg(Chars:=#serial,  // Convert to string
         pChars:=0,
         Cnt:=19,
         Strg=>"Data_block_1".device_serial);
```

- Add now the variable to your **Watch table** and set the PLC into the **RUN** mode:


2511_TIA_Integration ▸ PLC_1 [CPU 1511-1 PN] ▸ Watch and force tables ▸ Watch table_1

	i	Name	Address	Display format	Monitor value	Modify value	⚡	Comment
1		"in_cell_voltage"	%ID14	Floating-point nu...	0.58885			
2		"Data_block_1"....		String	'649810'			
3			<Add new>					

Note: Datatype **String** in TIA Portal contains two additional bytes, which represent the length of the string. To cut off these two bytes, use the function '*Chars_TO_Strg'* to convert the byte array to a String data type.

7.2. Read and Write of ‘real’ data types

Example 2: Set and Get the Limit Rdc – Lower Value

This example shows you how to write and read the Lower Limit Rdc in [mΩ]

- Add the needed variables to the **parameters table:**

2511_TIA_Integration ▸ PLC_1 [CPU 1511-1 PN] ▸ Program blocks ▸ Startup [OB100]

Startup

	Name	Data type	Default value	Comment
4	▼ Temp			
5	Valid	Bool		
6	Busy	Bool		
7	Error	Bool		
8	Status	DWord		
9	Done	Bool		
10	LenRead	UInt		
11	Limit_Rdc_Low_To_Set	Real		

- Add the variable **Limit_Rdc_Low_Read** to a data block as shown in the example 1 above

2511_TIA_Integration ▸ PLC_1 [CPU 1511-1 PN] ▸ Program blocks ▸ Data_block_1 [DB1]

Keep actual values Snapshot Copy snapshots to start values Load start values a

Data_block_1

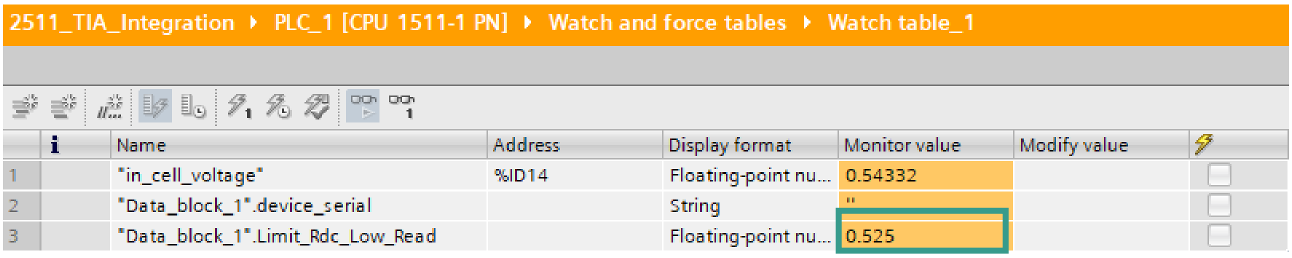
	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	▼ Static			☐	☐	☐	☐	☐
2	Limit_Rdc_Low_Read	Real	0.0	☐	☑	☑	☑	☐
3	device_serial	String	''	☐	☑	☑	☑	☐

- Use the following source code to write the value 0.525 as Limit A Lower Value and then read it back:

```
# Limit_Rdc_Low_To_Set := 0.525;

REPEAT
    "WRREC_DB"(REQ := TRUE,                              // Write access
            ID := 264,                                   // Hardware-ID  (see section 'Further examples')
            INDEX := 33,                                 // Index 33 to set the Lower Limit Rdc
            LEN := 4,                                    // Length in bytes to write
            DONE => #Done,
            BUSY => #Busy,
            ERROR => #Error,
            STATUS => #Status,
            RECORD := # Limit_Rdc_Low_To_Set);
UNTIL NOT #Busy AND #Done
END_REPEAT;

IF #Error = TRUE OR #Status <> 0 THEN                    // If write failed → return
  RETURN;
END_IF;

REPEAT
    "RDREC_DB"(REQ := TRUE,                              // Read access
            ID := 264,                                   // Hardware-ID (see 'Further examples')
            INDEX := 33,                                 // Index 33 to read the Lower Limit Rdc
```

```
            MLEN := 4,                                // Max. length of bytes to read
            VALID => #Valid,
            BUSY => #Busy,
            ERROR => #Error,
            STATUS => #Status,                        // Number of bytes read
            LEN => #lenRead,                          // Lower Limit Rdc
            RECORD := "Data_block_1".
Limit_Rdc_Low_Read);
UNTIL NOT #Busy
END_REPEAT;
```

- Check that the value of the variable **Limit_Rdc_Low_Read** is identical to the value of **Limit_Rdc_Low_To_Set**:



	i	Name	Address	Display format	Monitor value	Modify value	
1		"in_cell_voltage"	%ID14	Floating-point nu...	0.54332		
2		"Data_block_1".device_serial		String	''		
3		"Data_block_1".Limit_Rdc_Low_Read		Floating-point nu...	0.525		