

docs

ATOM / Fieldbus / PROFINET

Author: Control Concepts, Inc.

Date: 2025-10-30

Contents

• ATOM / Fieldbus / PROFINET / Overview	2
○ GSDML	2
○ Control Panel Communication Settings	2
○ Control Panel and PLC software	4
○ IP Address Conflict Detection	4
○ Hardware considerations	4
○ Daisy chaining	5
○ Parameters	5
■ Overview	5
■ Table	5
■ Additional parameter descriptions	9
■ Data types	14
○ Advanced	16
• ATOM / Fieldbus / PROFINET / TIA Portal V18	17
○ Requirements	17
○ Hardware setup	17
○ Creating a project in TIA Portal V18	19
○ Importing Atom's GSDML file into TIA	20
○ Adding and configuring your PLC	22
○ Adding and configuring Atom	29
○ Network provisioning	34
○ A basic test	38
■ Download to your PLC	38
■ Use a simulator	43
■ Monitor the heatsink temperature	44
○ Building a simple PLC program to control Atom	52
■ Writing some ladder logic	52
■ Running on your PLC	54
○ Troubleshooting	58
■ Download to PLC fails	58

● ATOM / Fieldbus / PROFINET / Codesys	62
○ Prerequisites	62
○ Hardware setup	62
○ Configure Windows firewall	63
○ Create a Codesys project	70
○ Adding a Profinet Controller	72
○ Adding ATOM to the controller	77
○ Create a program	84
○ Example: Ladder logic	85
■ Creating the program	85
■ Setting up visualization	95
■ Wiring up the controls	100
■ Mapping variables	102
○ Example: Structured text	106
■ Creating the program	106
■ Mapping variables	112
○ Running the program with SoftPLC	115

ATOM / Fieldbus / PROFINET / Overview

ⓘ INFO



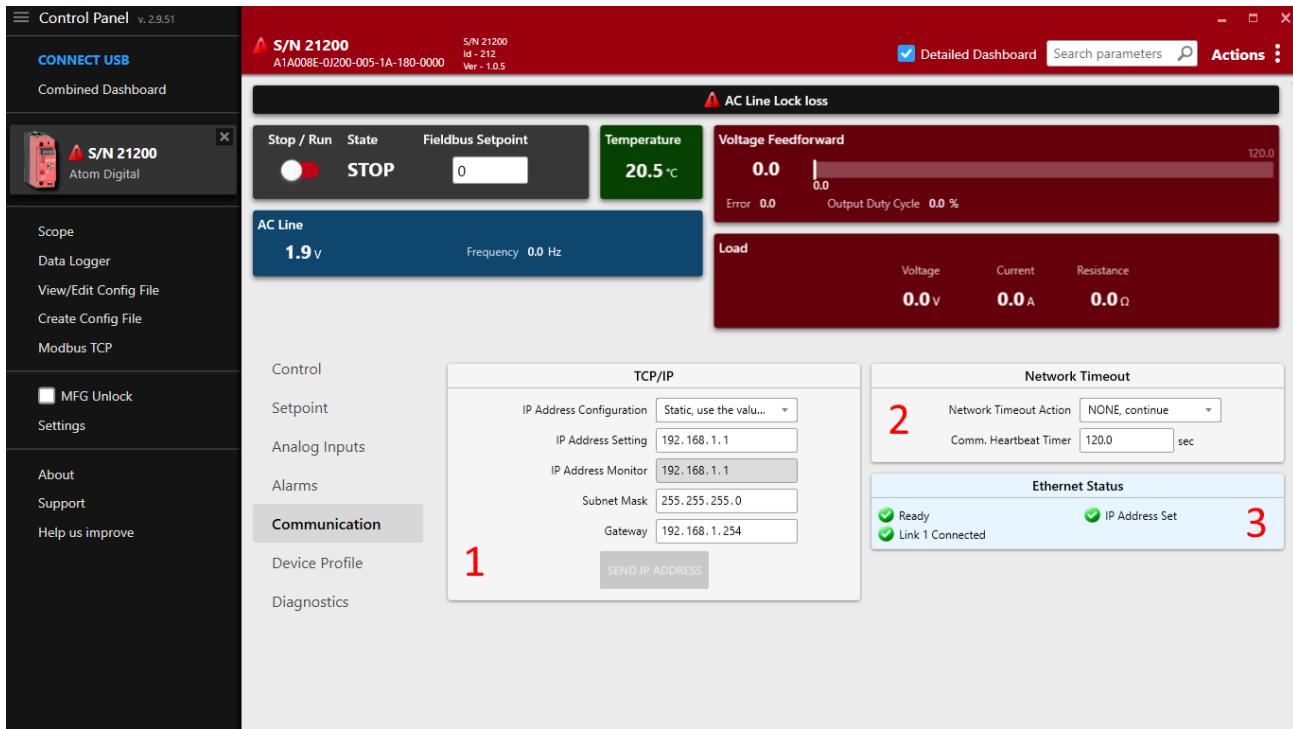
GSDML

A general station description markup language file (**gsdml**) for ATOM is available. This file can be imported into most PLC software to easily interface with ATOM.

ⓘ INFO

You can download the GSDML file for ATOM [here](#).

Control Panel Communication Settings



Some communication settings can be configured in the **Communication** tab in **Control Panel**.

- Section 1: TCP/IP settings
 - **IP Address Configuration**
 - **Static**: Use the IP address, subnet mask, and gateway specified below.
 - **DHCP**: Use DHCP to obtain an IP address.
 - **IP Address Setting**: The IP address of the ATOM controller.
 - **IP Address Monitor**: The current IP address of the ATOM controller.
 - **Subnet Mask**: The subnet mask of the ATOM controller.
 - **Gateway**: The gateway address for the ATOM controller.
- Section 2: Network Timeout
 - The EtherNet/IP heartbeat timeout (Encapsulation Inactivity Timeout) in seconds.
 - You can configure a network timeout action to perform when the device loses communication with the PLC:
 - **None**: Do nothing

- **STOP, fault shutdown:** STOP the controller, disabling output
 - **Use network timeout setpoint:** Configure an alternative setpoint to use when the controller loses communication with the PLC.
- Section ③: Ethernet status
 - Indicates the status of both RJ45 ports, IP address configuration, conflict detection, and any other errors with the EtherNet/IP connection.

ⓘ INFO

Control Panel and PLC software

These settings are synchronized with your PLC environment. You do not have to use Control Panel to change these settings - you can stay in your PLC software. Control Panel merely provides them as an alternative way to configure ATOM's EtherNet/IP settings.

You can use Control Panel simultaneously with your PLC software without issues.

⚠ WARNING

IP Address Conflict Detection

ATOM uses **IP Address Conflict Detection** to detect IP address conflicts on the network. If ATOM detects another device using the same IP address, it will disable all network communication until the conflict is resolved.

Please ensure all devices on the network are assigned unique a IP address.

Hardware considerations

⚠️ WARNING

Daisy chaining

As ATOM has two RJ45 ports, it can be easily daisy-chained. When daisy-chaining ATOM, take care to avoid a loop in the network. In some loop configurations, ATOM is susceptible to network broadcast storms, which can cause the controller to become unresponsive. If you are daisy-chaining ATOM, ensure that the network is loop-free.

ATOM works with both unmanaged and managed switches. We recommend a managed switch for larger networks to give you more control over the network topology.

Parameters

Overview

ATOM makes 30 parameters accessible to Profinet. These parameters are made available through the input and output modules.

Table

#	Name	Type	Description	Read/Write
1	Digital setpoint	DINT	A value between 0 and 10,000 indicating the desired output current. The value is scaled to the output range of ATOM. For example, if the output range is 0-	Read/Write

#	Name	Type	Description	Read/Write
			100A, a value of 5000 would set the output to 50A.	
2	Digital run enable	BOOL	Enables or disables the output current. When disabled, the output current is set to 0A.	Read/Write
3	Inhibit Alarm Status	BYTE	A bitfield indicating alarms that are preventing controller operation. See Inhibit Alarm Status .	Read
4	Warning Alarm Status	BYTE	A bitfield indicating warning alarms. See Warning Alarm Status .	Read
5	Feedback Read Status	BOOL	A bitfield indicating if controller has acquired feedback. See Feedback Read Status .	Read
6	AC Line Frequency	REAL	The AC line frequency in Hz.	Read
7	AC Line Voltage	REAL	The AC line voltage in volts.	Read
8	Load Voltage	REAL	The load voltage in volts.	Read
9	Load Current	REAL	The load current in amps.	Read
10	Load Resistance	REAL	The load resistance in ohms.	Read

#	Name	Type	Description	Read/Write
11	Heatsink Temperature	REAL	Heatsink temperature, in degrees celsius.	Read
12	Output Duty Cycle %	REAL	Indicates the amount, in percent, that the output of the controller is ON	Read
13	Setpoint reference	REAL	Reference input to control compensation loop in units determined by "feedback type"	Read
14	Feedback	REAL	The control output supplied to the load in units determined by "feedback type"	Read
15	Partial Load Fault Target Resistance	REAL	Expected nominal resistance, in Ohms, of the load. Used for partial load fault detection.	Read
16	Partial Load Fault Resistance	REAL	The actual load resistance in Ohms. Compared with #15 to determine if a partial load fault has occurred.	Read
17	Partial Load Fault Resistance Deviation	REAL	The tolerable percentage that parameter #15 and #16 may differ by until a partial load fault will be triggered.	Read
18	Firmware ID	DINT	Indicates the version of firmware that is loaded, dictating which	Read

#	Name	Type	Description	Read/Write
			features are available.	
19	Firmware major revision	DINT	Indicates which revision of the firmware is loaded. Major revisions fix critical bugs or add significant new features.	Read
20	Firmware minor revision	DINT	Indicates which minor revision of the firmware is loaded. Minor revisions fix minor issues and/or add minor improvements.	Read
21	Full Scale Voltage	DINT	The expected output voltage when the controller output is fully on.	Read
22	Full Scale Current	REAL	The expected current when the controller output is fully on.	Read
23	AC Line Status	BYTE	A bitfield indicating the status of the connected AC Line. See AC Line Status .	Read
24	Load Status	BYTE	A bitfield indicating the load status. See Load status .	Read
25	Controller Status	BYTE	A value indicating the operational status of the controller. See Controller status .	Read

#	Name	Type	Description	Read/Write
26	Controller State	BYTE	A value indicating the controller state. See Controller state .	Read
27	EEPROM Status	WORD	A bitfield indicating the EEPROM status. See EEPROM Status .	Read
28	EEPROM Status 2	WORD	Identical to parameter #27	Read
29	Error Latch	BYTE	A bitfield used for diagnostic troubleshooting. See Error Latch .	Read
30	Miscellaneous Status	BYTE	A bitfield indicating miscellaneous status information. See Miscellaneous Status .	Read

Additional parameter descriptions

Inhibit Alarm Status

Inhibit alarm status is a 8-bit bitfield:

7	6	5	4	3	2	1
Reserved	Reserved	Reserved	Reserved	Feedback Loss	Over Temperature	Over Current Trip

If any bit is set to 1, the controller will *not* be allowed to run.

Warning Alarm Status

Warning alarm status is a 8-bit bitfield:

7	6	5	4	3	2	1	0
Reserved	Reserved	High temperature	Shorted SCR	Open Load	Partial Load Fault	Current Limit	Voltage Limit

Warning alarms are not considered critical and will not prevent the controller from running.

Feedback Read Status

Feedback status is a 8-bit bitfield:

7	6	5	4	3	2	1	
Reserved	Ti						

Indicates whether the controller has acquired feedback on the line. If any bit is set to 1, then the controller has lost feedback.

AC Line Status

AC Line status is a 8-bit bitfield:

7	6	5	4	3	2	1	0
Reserved	Reserved	Sync-Locked (to AC Line)	Pre-Lock 2	Pre-Lock 1	Reserved	AC Line B OK	AC Line A OK

Bits 5 must be set to 1 before the controller can provide power to the load.

Load Status

Load status is a 8-bit bitfield:

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Open Load	Reserved	Reserved	Reserved	Short SCR

Controller Status

Controller status is one of:

Value	Description
0	Disabled
1	Initialization
2	Normal, operating
3	Calibration

Value	Description
4	Diagnostic

Controller State

Controller state is one of:

Value	State	Description
0	STOP	The state the controller is in when AC Line voltage is not present.
1	RUN	The state the controller is in when AC Line voltage is present and the controller is synchronized to the AC line.
2	FAULT	A latching state of output shutdown caused by over current or over temperature alarms. A power cycle or processor reset is required to clear this state.
3	FAULT RESET	Used as a temporary state to transition from FAULT to RUN once again.

EEPROM Status

EEPROM status is an 16-bit bitfield. EEPROM is used to store controller configuration and calibration data. Any errors in EEPROM may indicate that the firmware is corrupted.

Bit	Description
0	EEPROM Initialization

Bit	Description
1	SP Table Error
2	MFG CP Table Error
3	Calibration Table Error
4	Reserved
5	Reserved
6	Backup Calibration Table Error
7	Bottom Board Calibration Table Error
8	SP Definition Table needs updating
9	Bottom Board Calibration Backup Error
10	Reserved
11	Reserved
12	EEPROM is write protected
13	Reserved
14	Reserved
15	Feedback Calibration Table has changed, store to EEPROM

Error Latch

Error latch is a 8-bit bitfield:

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Feedback loss	SCR timing loss	Line Frequency failure	Phase loss or missing cycle	Line Lock Loss

Error latch is provided as a diagnostic troubleshooting aid.

Miscellaneous Status

Miscellaneous status is an 8-bit bitfield:

7	6	5	4	3	2	1
Reserved	Initialization in progress	Reserved	Reserved	Waiting for ENTER key during initialization	Reserved	USB Powerrec

Data types

The data types listed above in the parameter table are defined in the CIP standard as:

Type	Size	Description
BOOL	1 byte	Boolean value

Type	Size	Description
BYTE	1 byte	8-bit bitmap
WORD	2 bytes	16-bit bitmap
DWORD	4 bytes	32-bit bitmap
LWORD	8 bytes	64-bit bitmap
USINT	1 byte	Unsigned 8-bit integer
UINT	2 bytes	Unsigned 16-bit integer
UDINT	4 bytes	Unsigned 32-bit integer
ULINT	8 bytes	Unsigned 64-bit integer
SINT	1 byte	Signed 8-bit integer
INT	2 bytes	Signed 16-bit integer
DINT	4 bytes	Signed 32-bit integer
LINT	8 bytes	Signed 64-bit integer
REAL	4 bytes	32-bit floating point number
LREAL	8 bytes	64-bit floating point number

Advanced

ATOM has many more parameters beyond the 30 made available through Profinet. The default profile listed above should be sufficient for the majority of use cases.

If this is not the case, you can use [Control Panel](#) to adjust or monitor all parameters.

In the rare case that you need more parameters available through ATOM's Profinet profile, Control Concepts does have the ability to make additional parameters available or to change the data type of included parameters. Please [contact us](#) if you would like a custom Profinet profile. There may be a service fee for custom Profinet profiles as they require new GSDML files, device-reconfiguration and testing.

ATOM / Fieldbus / PROFINET / TIA Portal V18

In this tutorial, you'll learn how to connect Atom to a Siemens PLC over Profinet. You'll learn how to update the run/stop and setpoint parameters, and monitor the heatsink temperature.

If you haven't yet, please review ATOM's [Profinet Profile](#).

If you'd like to skip the tutorial, you can download a completed example project in TIA Portal V18:

- Download [AtomExample.zip](#)

Requirements

1. TIA Portal V18
2. A Siemens PLC (optional, you can still follow along by simulating the PLC on your PC).
 - i. We use a `S7-1511-1 6ES7 511-1AK02-0BA0` in this example.
 - ii. You can easily follow along with a different PLC if you have a different one.
3. Download Atom's [GSDML file](#)

Hardware setup

ⓘ INFO

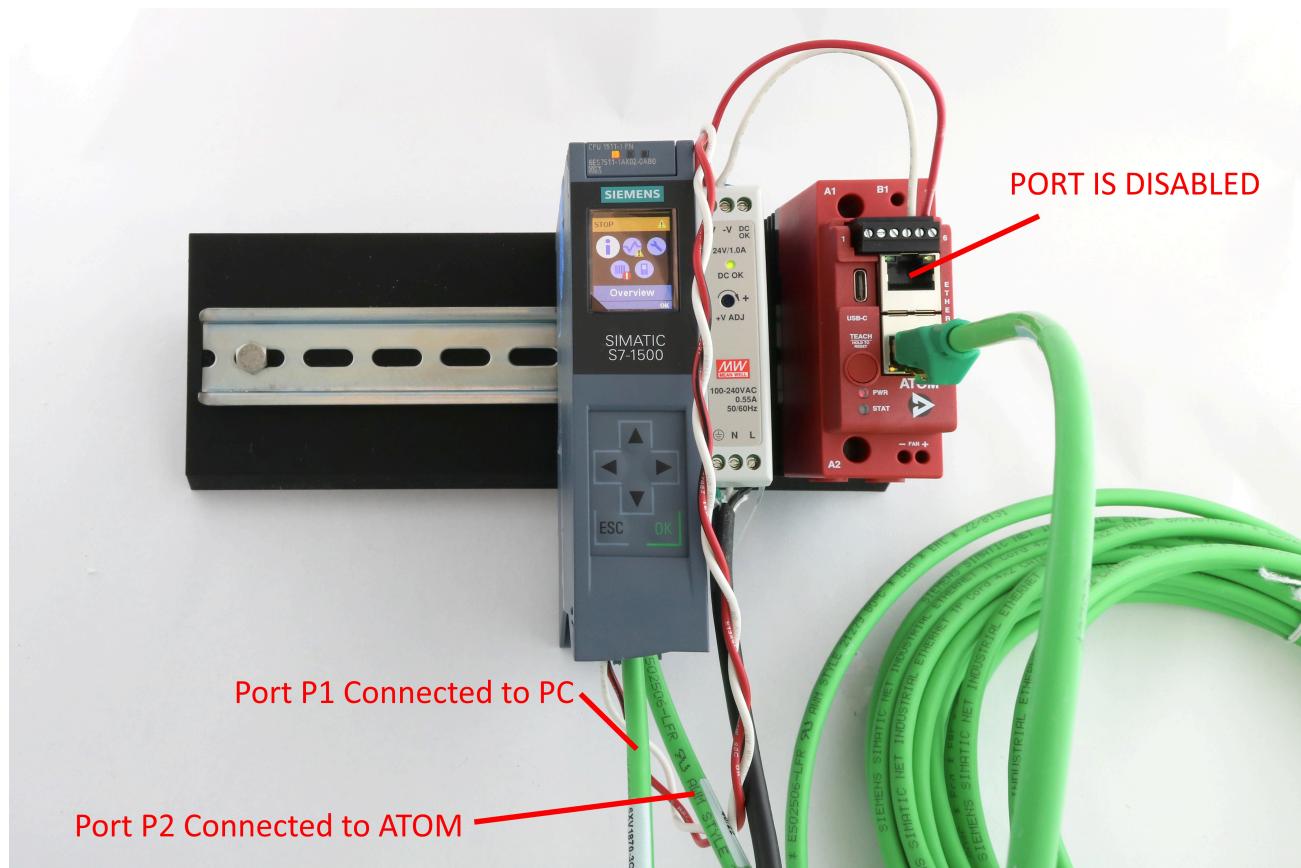
If you're using the PLC simulator, skip to the [next section](#).

⚠️ IMPORTANT

When Atom is configured for Profinet, the Ethernet port closest to the 24V power connector is **disabled**.

Connect 24V to your PLC and Atom unit with the provided power cable. Connect two Ethernet cables to the PLC:

- P1: Connect to your PC
- P2: Connect to your Atom unit



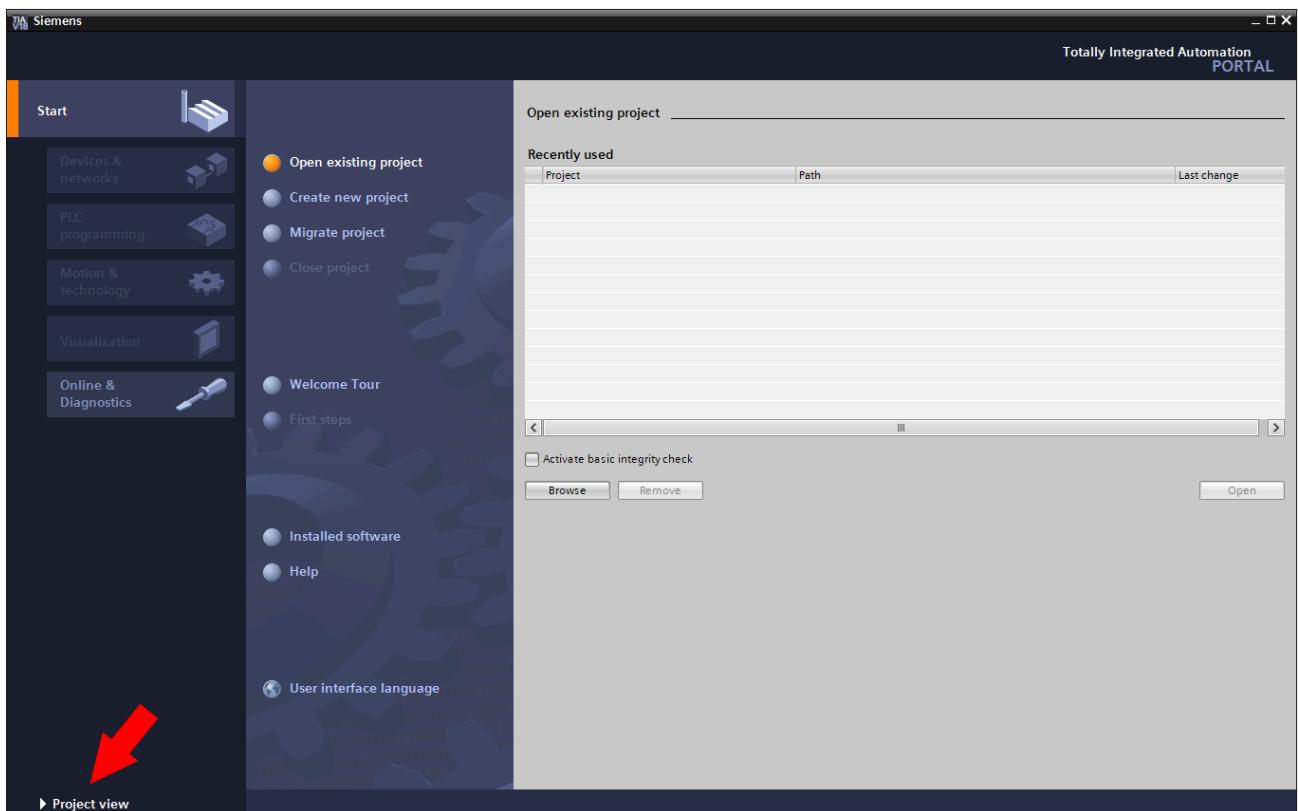
ⓘ INFO

To simplify this diagram, we have not connected a load to Atom.

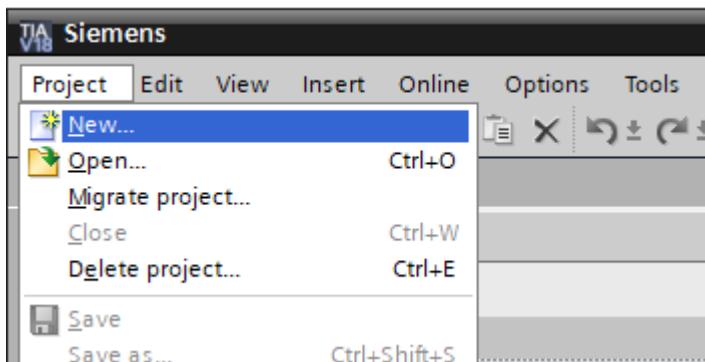
If you do not connect a load, you can still verify your PLC is working by connecting a USB cable to Atom and using Control Panel to watch the parameters change/verify the PLC is receiving the correct monitor data.

Creating a project in TIA Portal V18

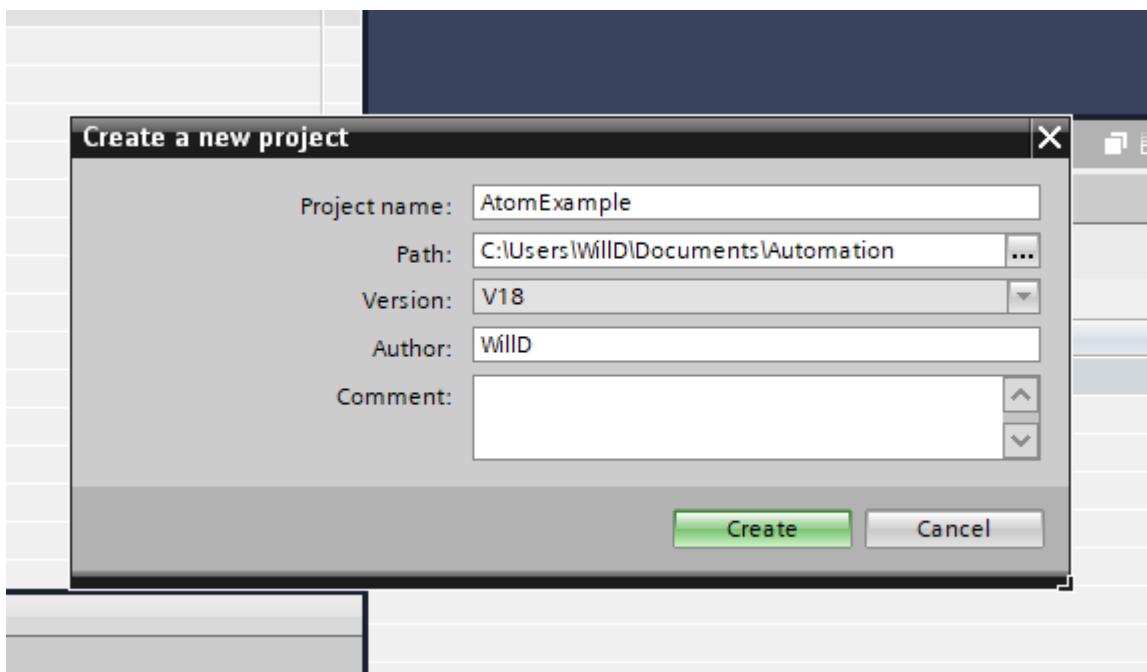
Open TIA Portal V18. If you're in the *Portal view* (shown below), switch to *Project view* by clicking in the lower left:



Next, select **Project > New**:



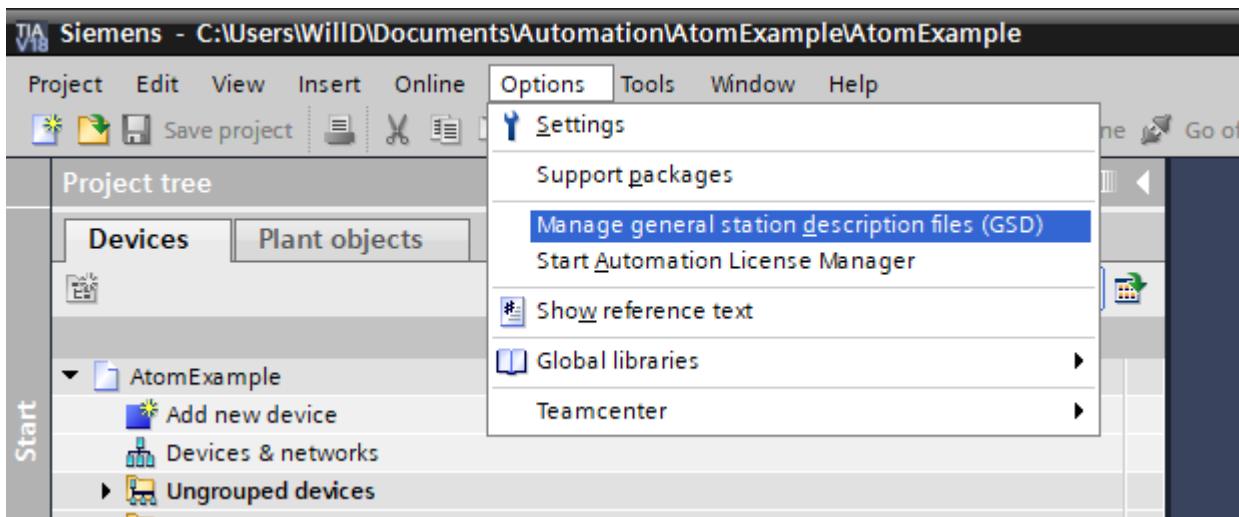
Give your project a name, like AtomExample, then click **Create**:



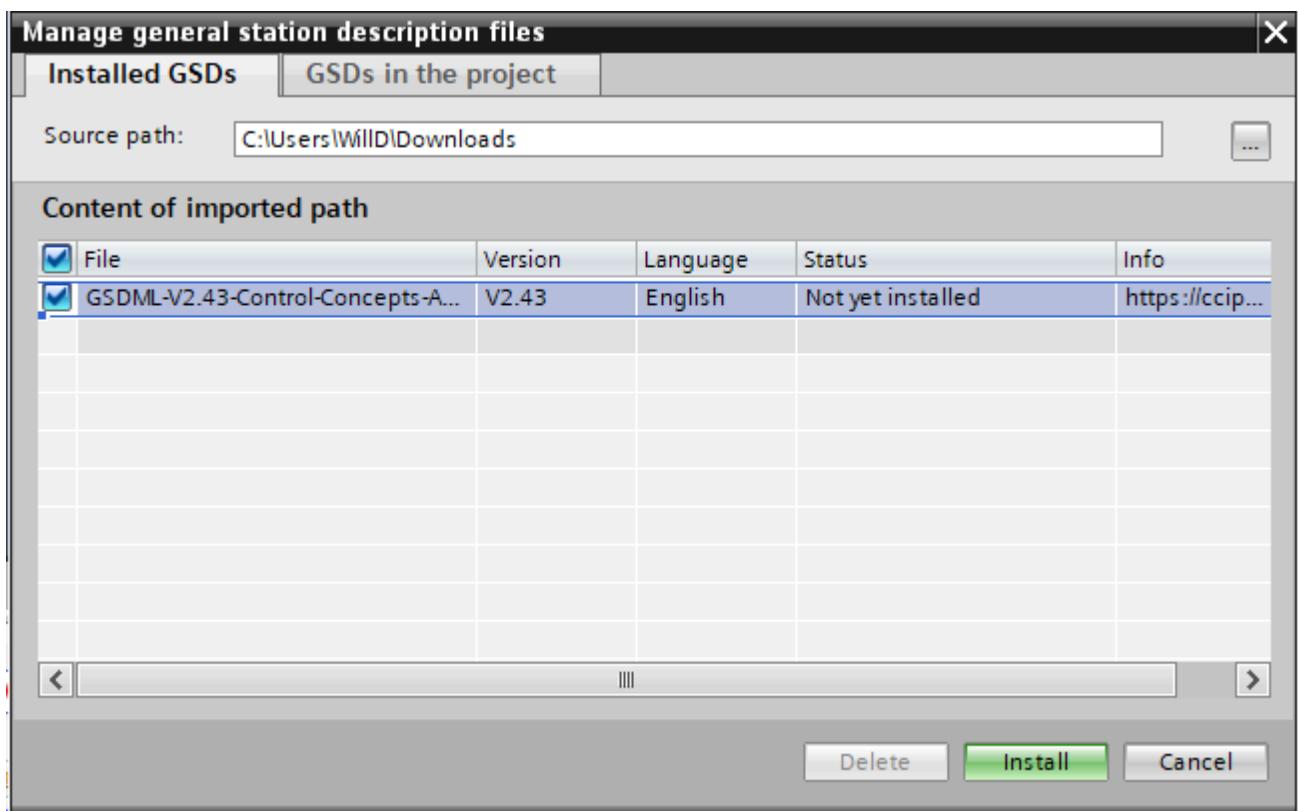
Importing Atom's GSDML file into TIA

A GSDML file describes the capabilities of a Profinet device — the parameters, communication modes, diagnostics (and more) that the device supports. We'll import Atom's GSDML file into TIA so that TIA knows how to talk to Atom.

Select **Options > Manage general station description files**:



Enter the path of the folder containing your GSDML file, check `GSDML-V2.43-Control-Concepts-ATOM-20231108.xml`, then click **Install**:



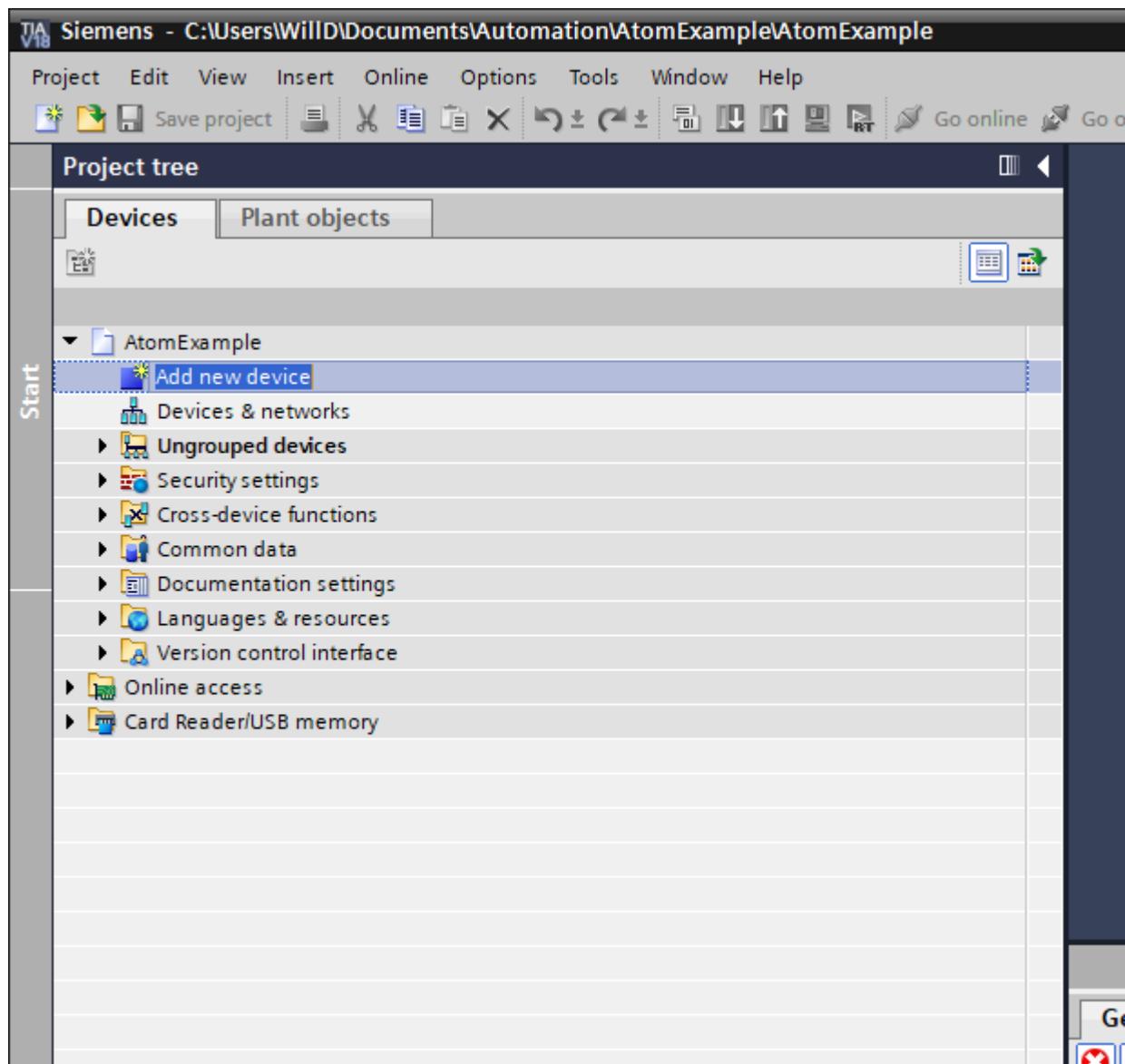
If everything goes well, a dialog should appear "Installation was completed successfully". Click **Close**.

Adding and configuring your PLC

INFO

Even if you're using the PLC simulator, still follow this section and add a Siemens PLC to your project. TIA is capable of simulating the S7-1500 series PLCs.

In the **Project tree** pane, within the **Devices** tab, double click **Add new device**:



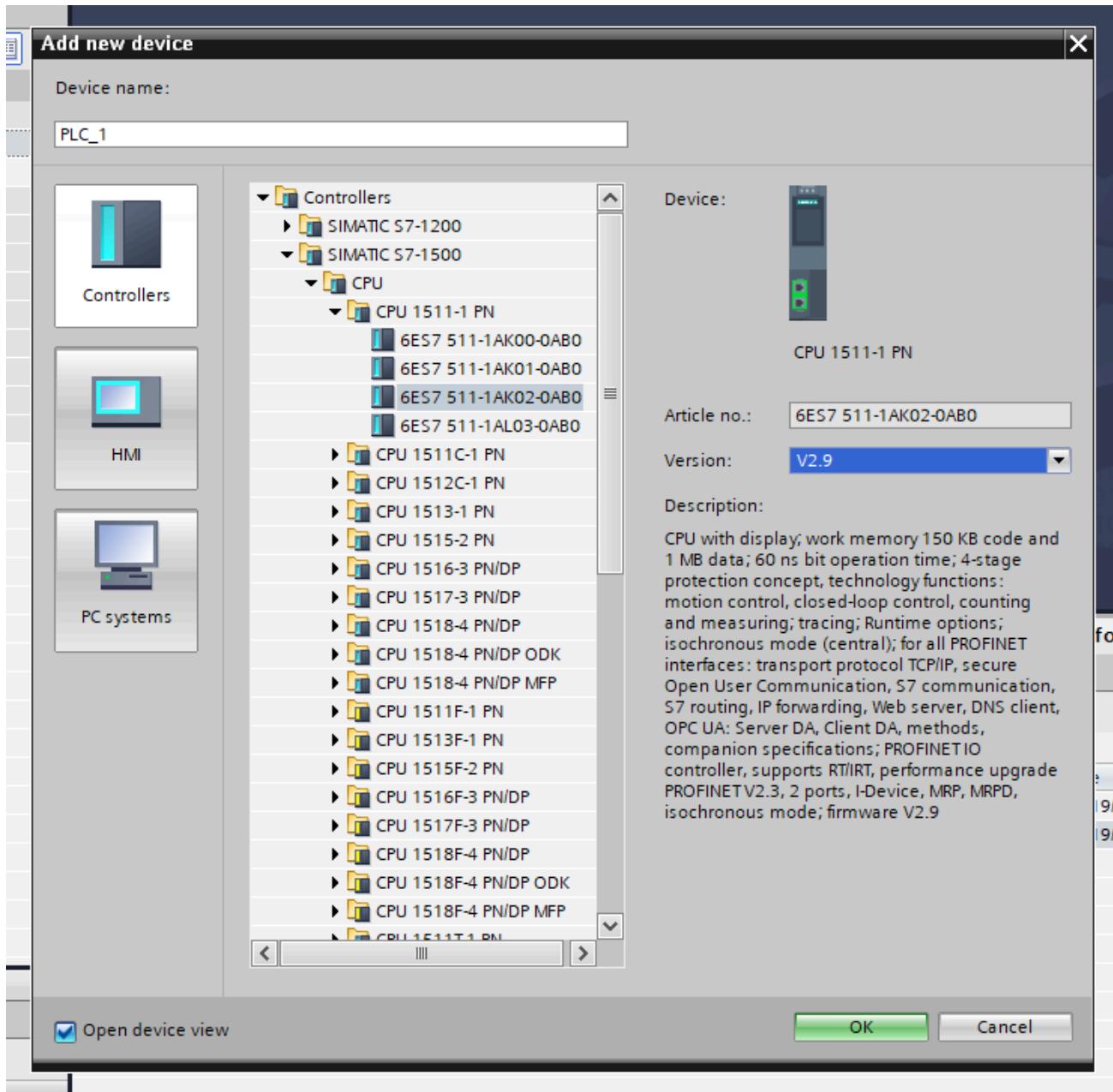
Select the **Controllers** tab, and browse to **Controllers > SIMATIC S7-1500 > CPU > CPU 1511-1 PN > 6ES7 511-1AK02-0AB0** (if you're using a different PLC, select that PLC instead):

ⓘ INFO

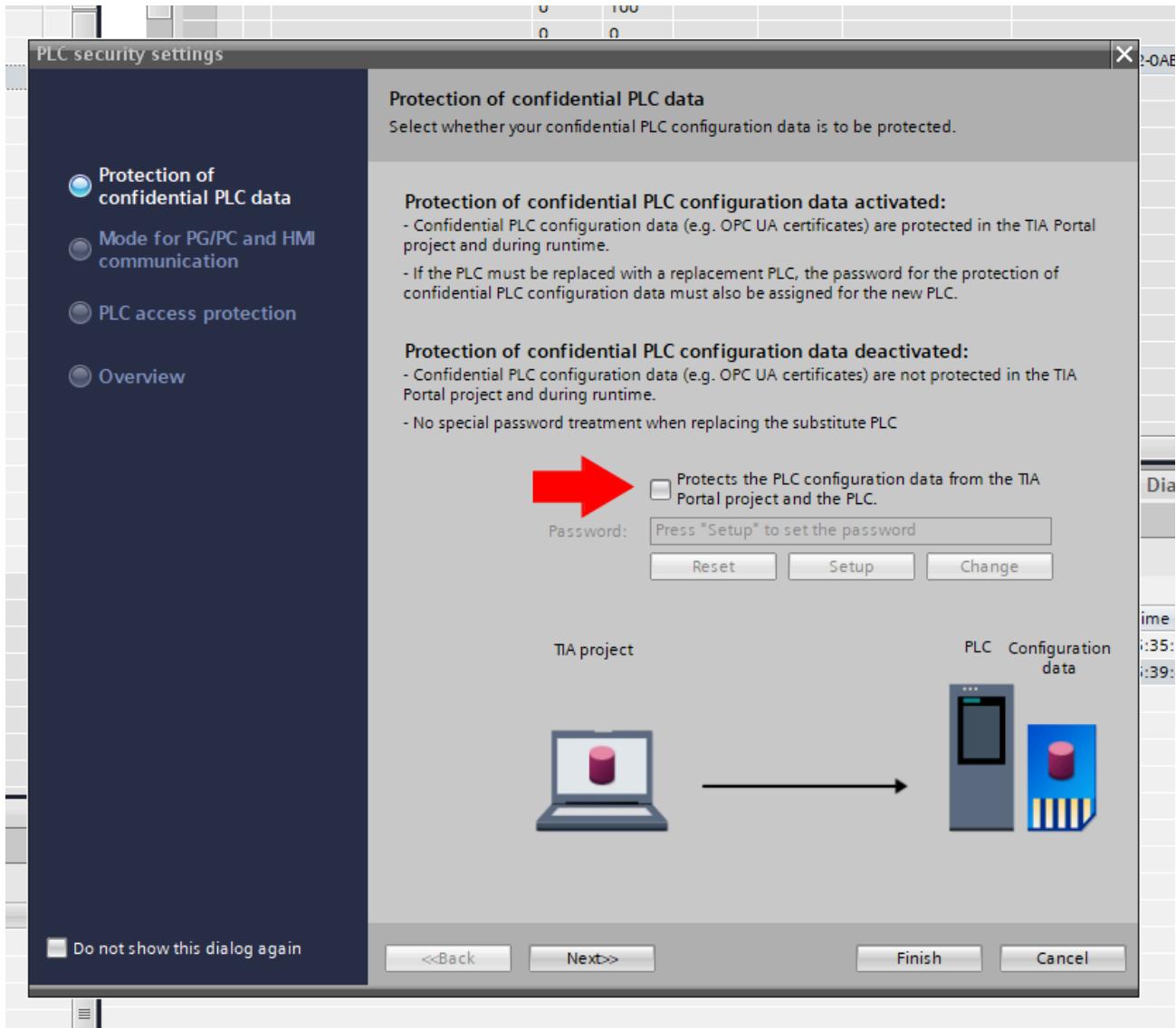
You can generally find your PLC's model number on the frame of your PLC. You can also check the firmware version (and model number) within the **Overview** page on your PLC:



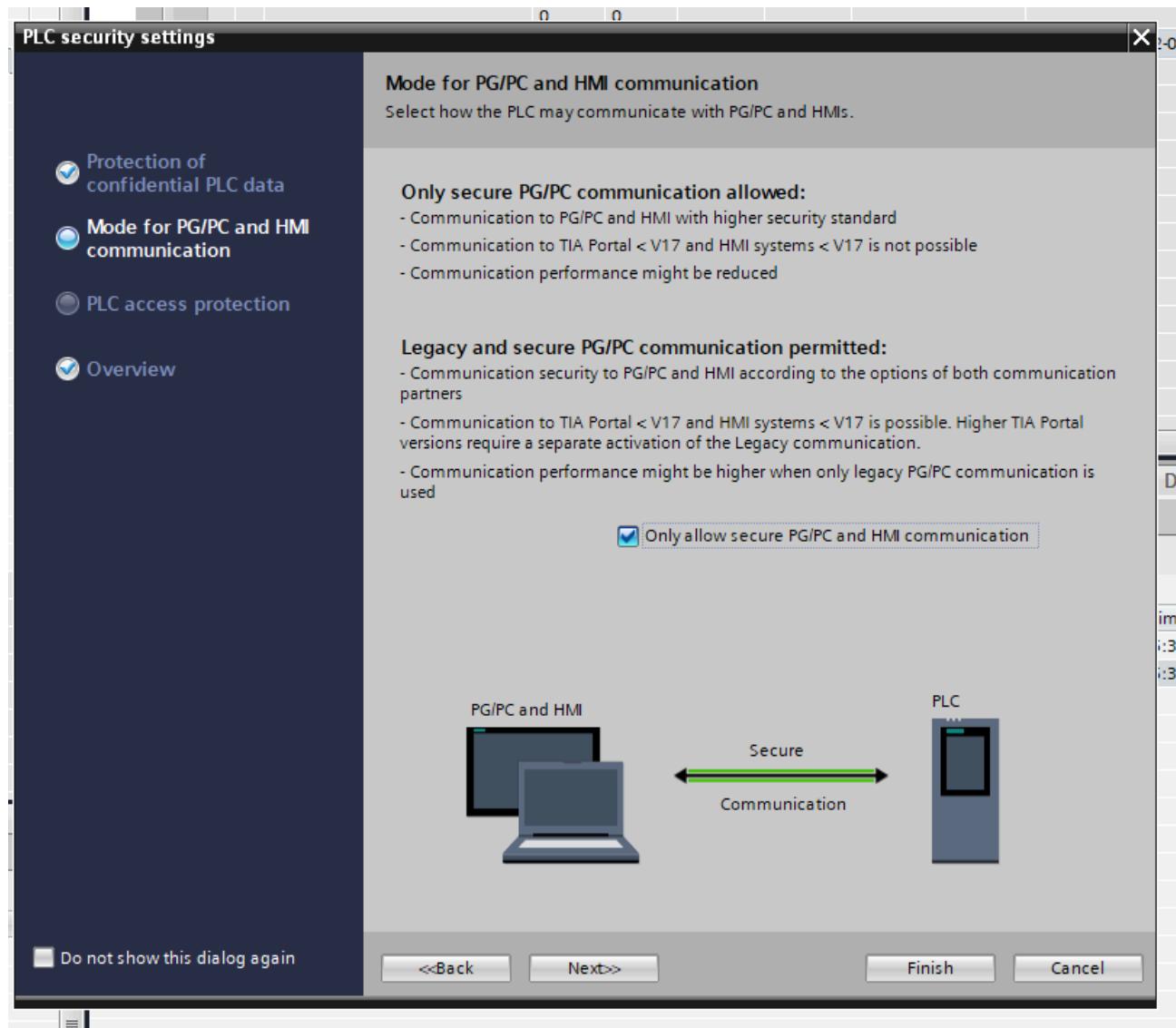
Then, click **Ok**:



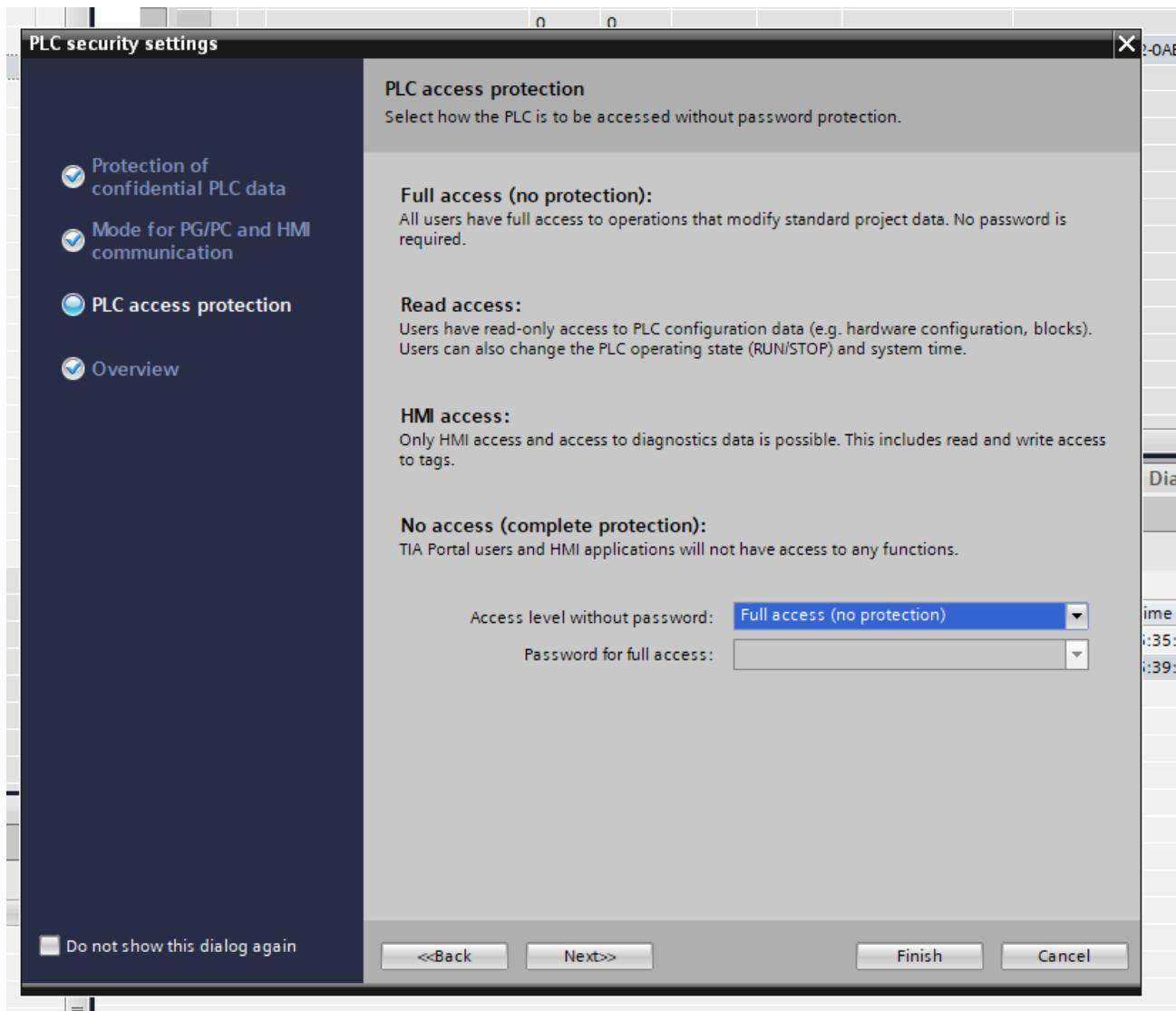
A **PLC security settings** dialog will popup. For this example, we'll disable the PLC password (unchecked):



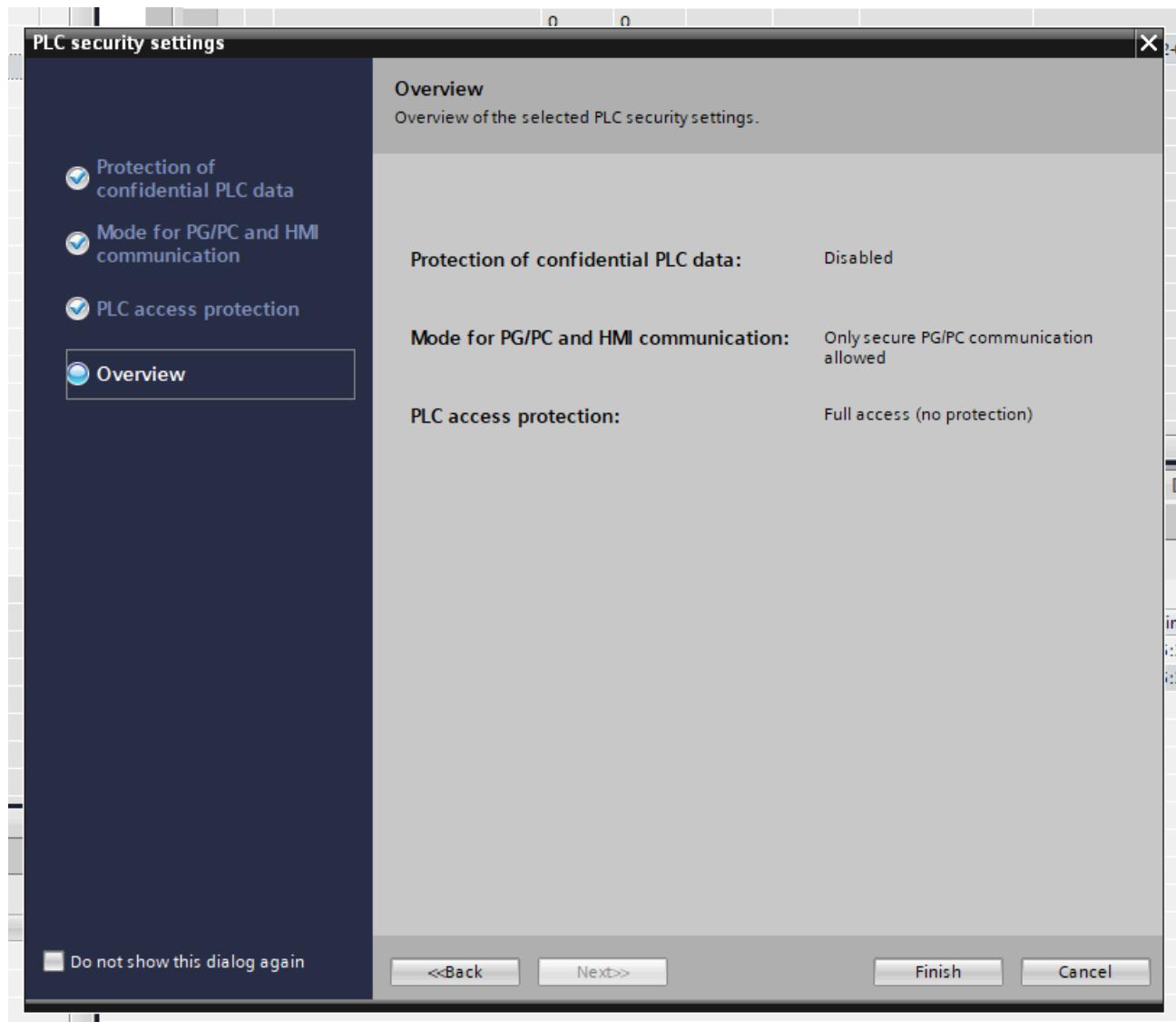
Leave **Only allow secure PG/PC and HMI communications** checked, then click **Next>>**:



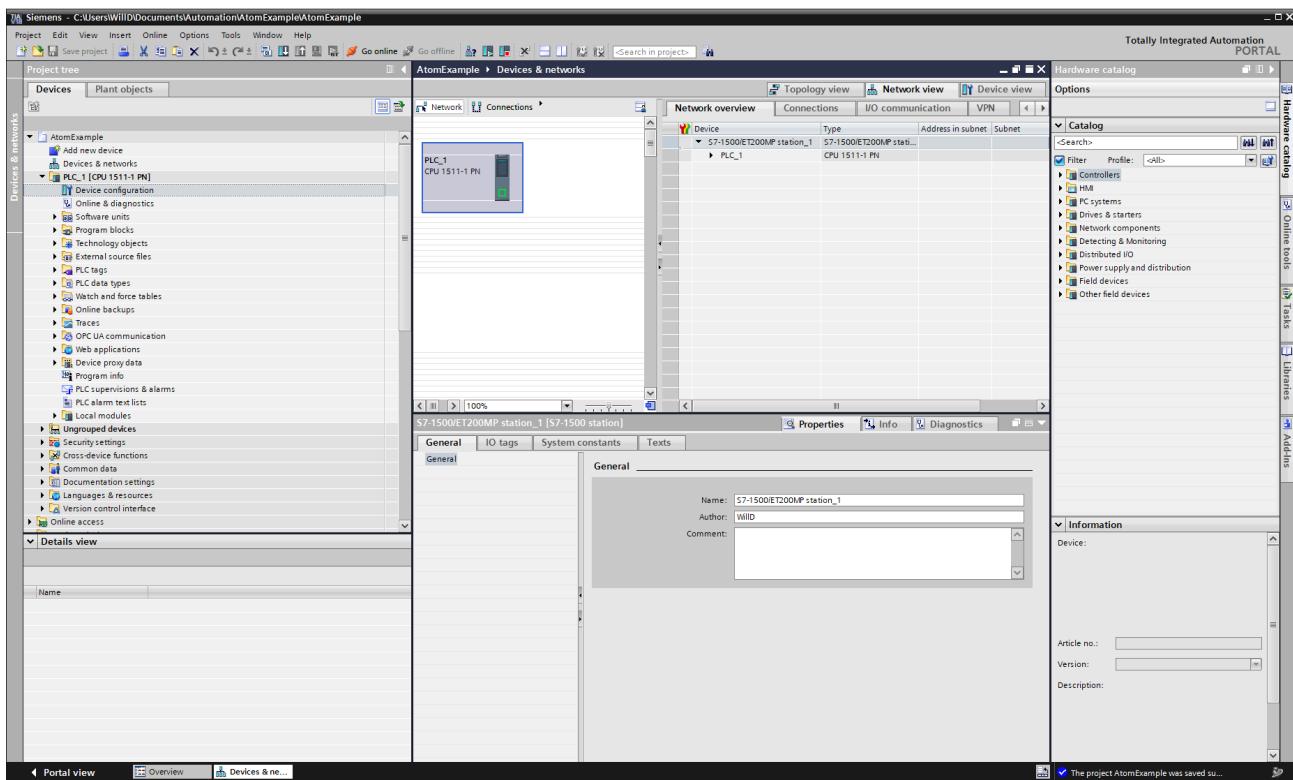
For this example, change **Access level without password** to **Full access (no protection)**, then click **Next>>**:



Then, click **Finish**:

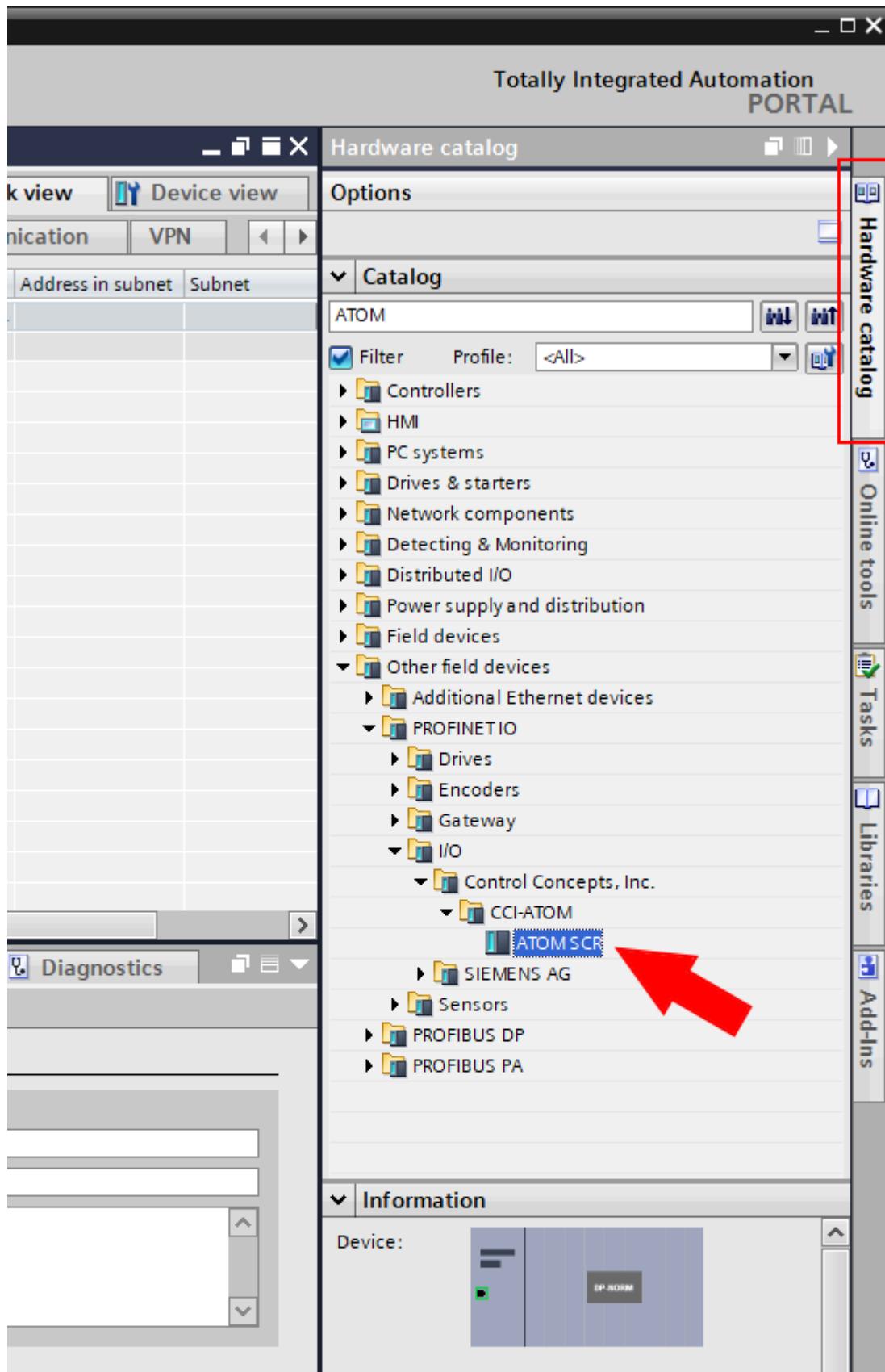


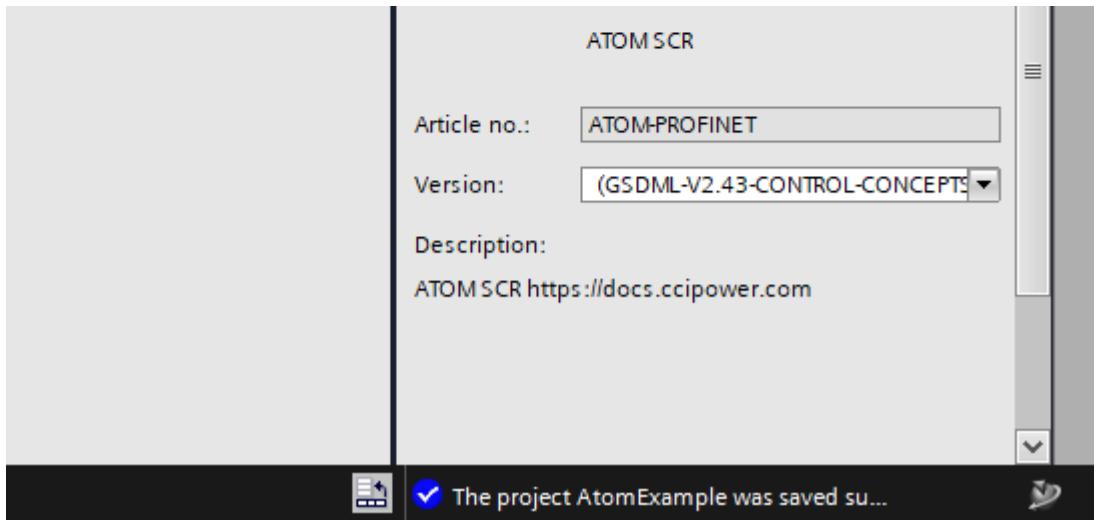
The PLC should appear in the **Network view**:



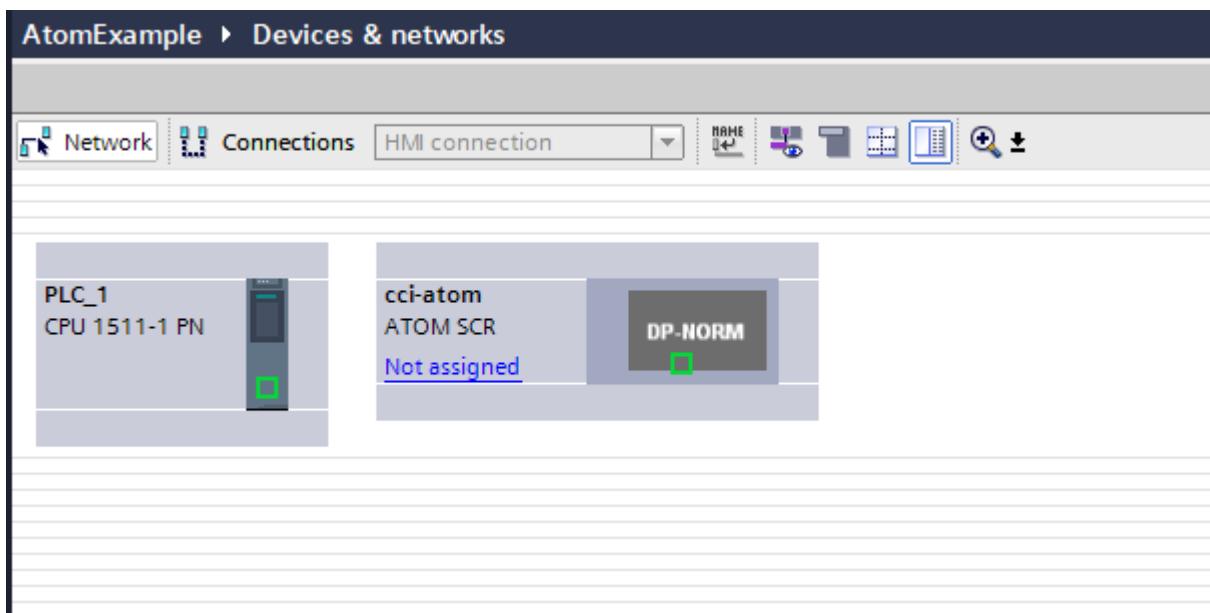
Adding and configuring Atom

To add Atom to your project, select the **Hardware catalog** tab on the right side of the screen. Enter **ATOM** in the search box and double click **ATOM SCR**:

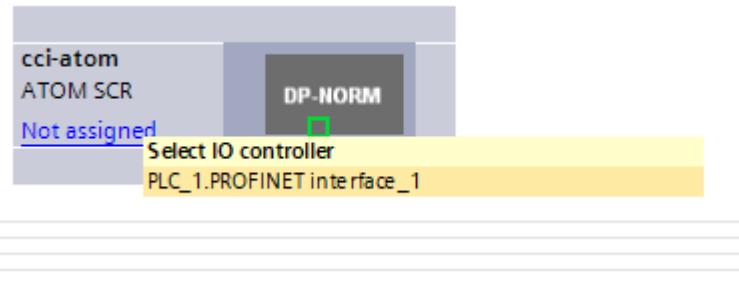




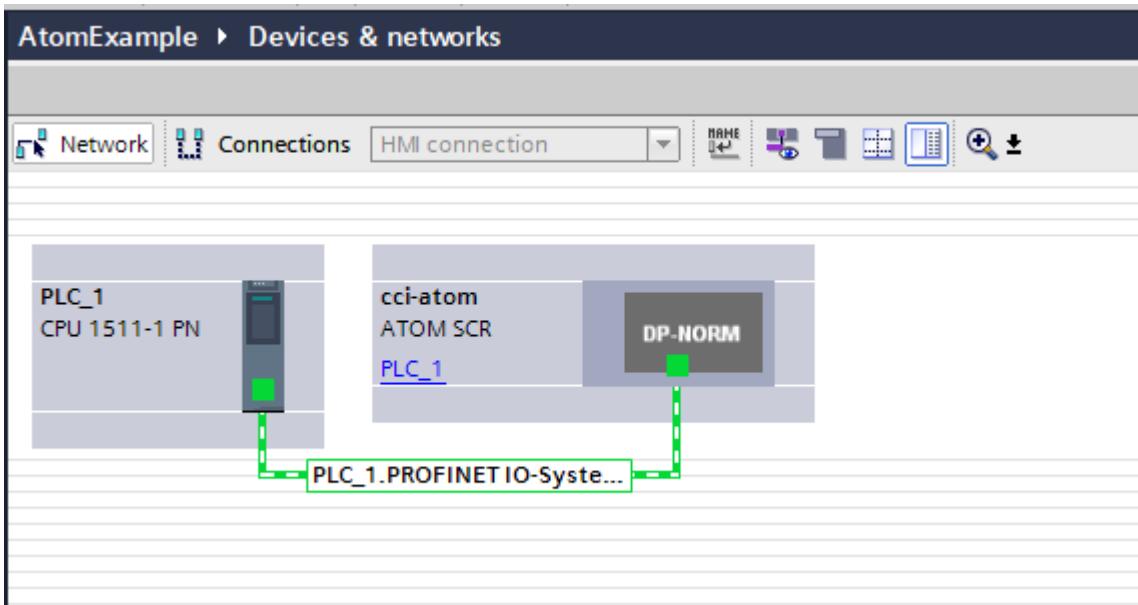
Atom should appear in the **Network view** of your project:



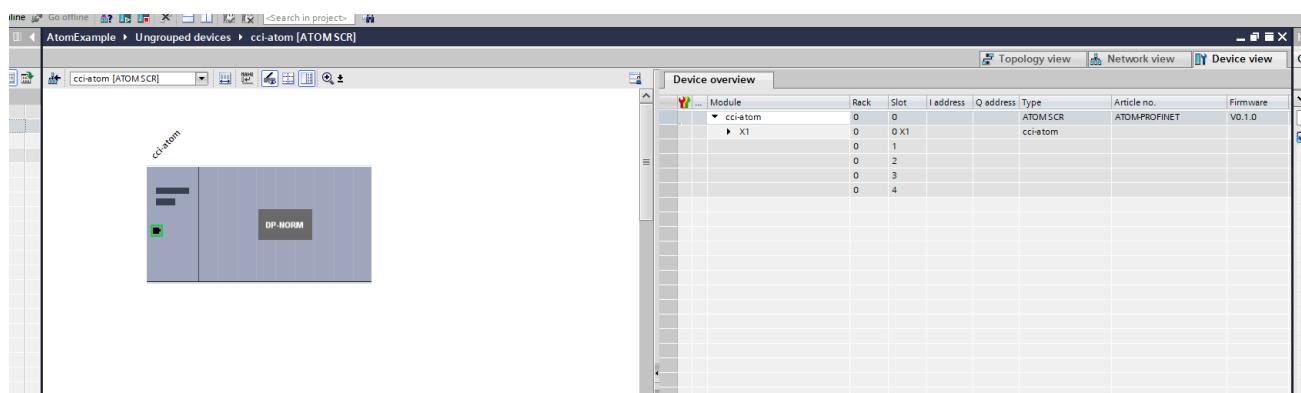
Click **Not assigned** on the Atom block, and select **PLC_1.PROFINET interface_1**:



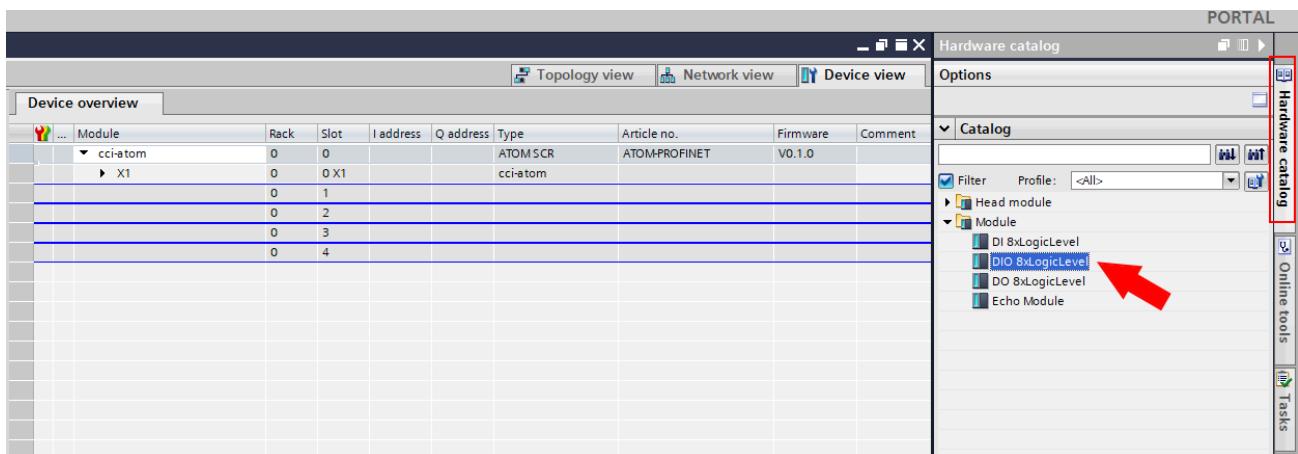
After doing so, a network connection should appear between the PLC and Atom:



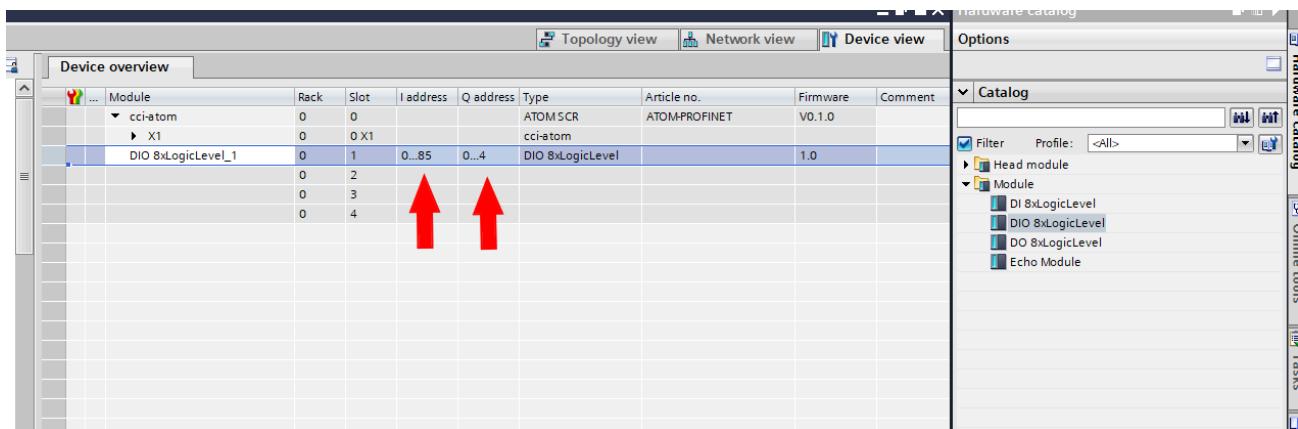
Next, double click on the Atom block to open its **Device view**:



Select the **Hardware catalog** tab on the right side, expand **Module** and select drag the **DIO 8xLogicLevel** module into **Slot 1** of Atom:



Your Atom module configuration should look like this:



INFO

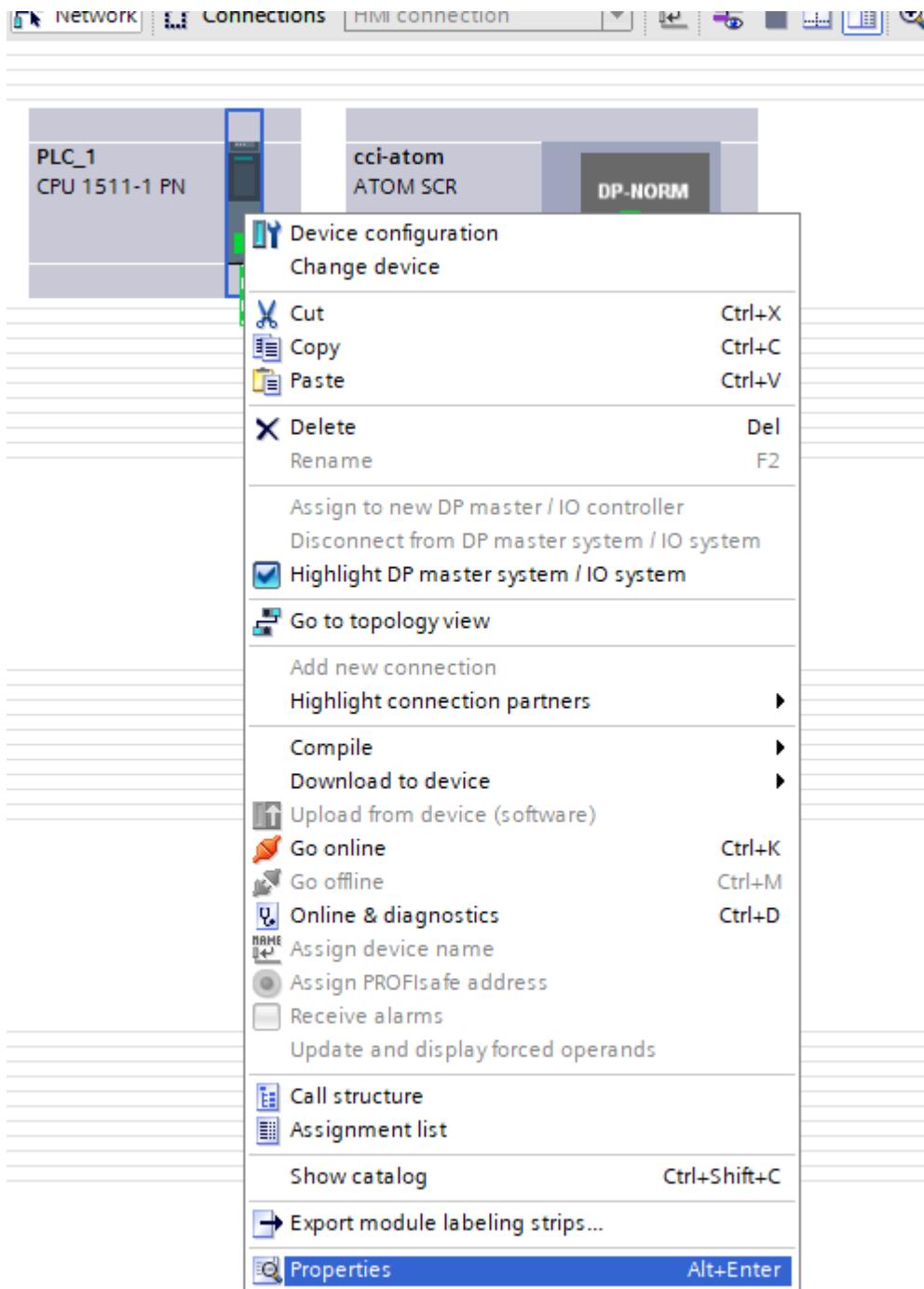
The two red arrows indicate the number of input and output parameters that Atom supports. In this case, the addresses `IW0` to `IW85` are input parameters and `QW0` to `QW85` are output parameters. Check out the [Profinet Profile](#) for more information on the available parameters.

Network provisioning

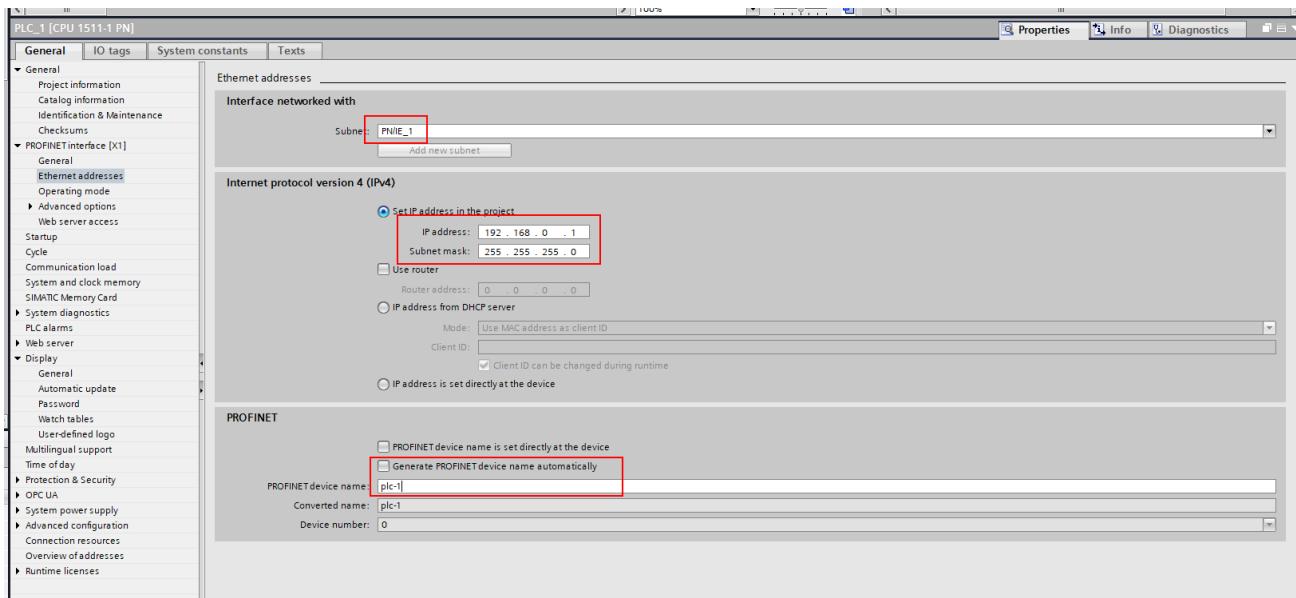
Let's set up a basic Profinet network. We'll provision a basic network like so:

- PC
 - IP address `192.168.0.25`
 - Subnet mask `255.255.255.0`
 - Only used for loading the PLC program
- Siemens PLC
 - Station name `plc-1`
 - IP address `192.168.0.1`
 - Subnet mask `255.255.255.0`
 - Port P1 connected to PC
 - Port P2 connected to Atom
- Atom
 - Station name `atom-1`
 - IP address `192.168.0.2`
 - Subnet mask `255.255.255.0`

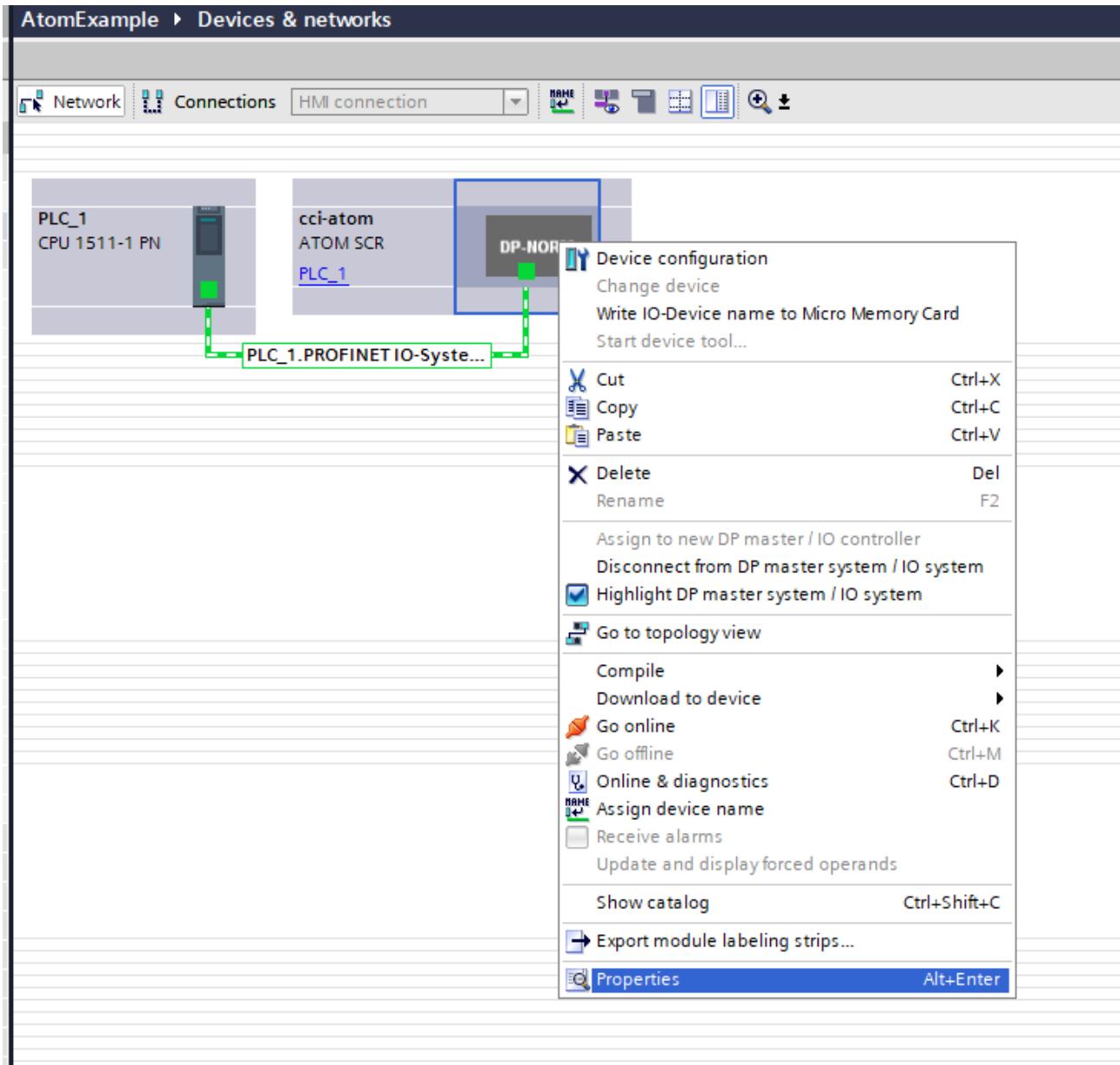
Return to the **Network view**, right click the PLC block (make sure to right click directly on the graphic) and select **Properties**:



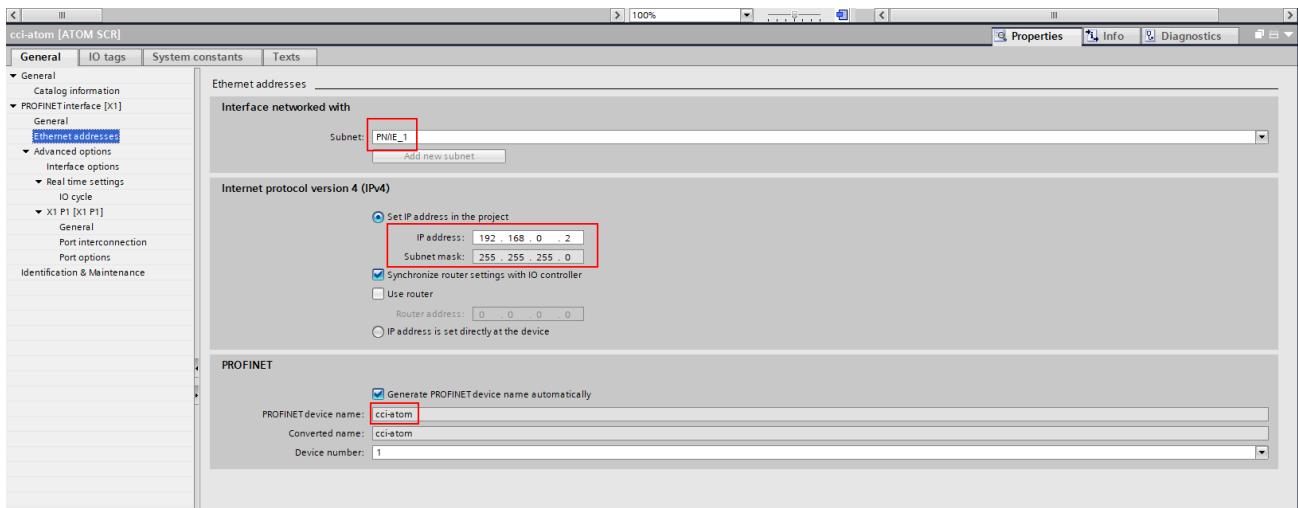
Navigate to **PROFINET interface [X1]** and select **Ethernet addresses**. Change these to match our network provisioning above:



Navigate back to the **Network view**, right click the Atom block (make sure to right click directly on the graphic) and select **Properties**:



Navigate to **PROFINET interface [X1]** and select **Ethernet addresses**. Check these to match our network provisioning above:

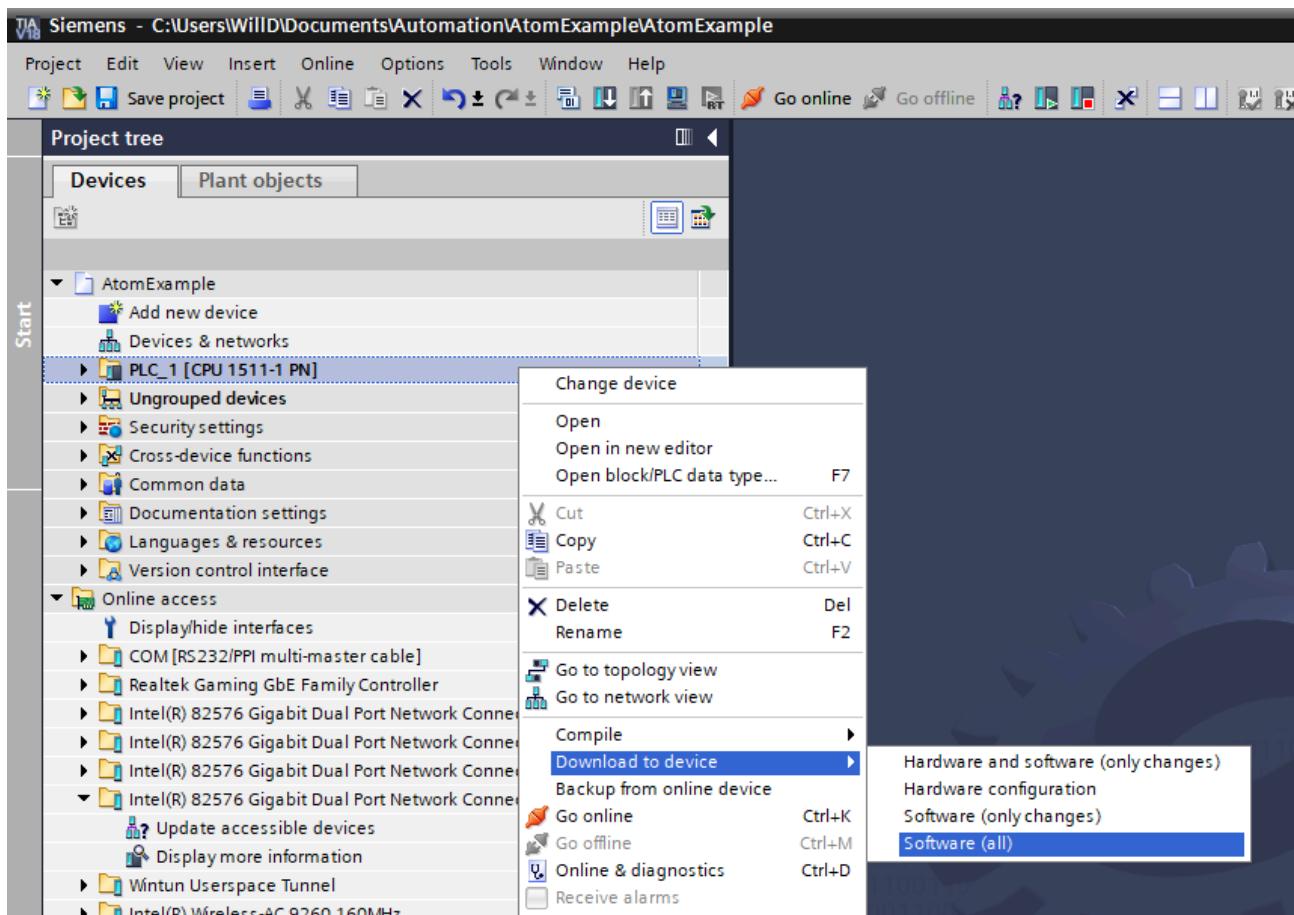


A basic test

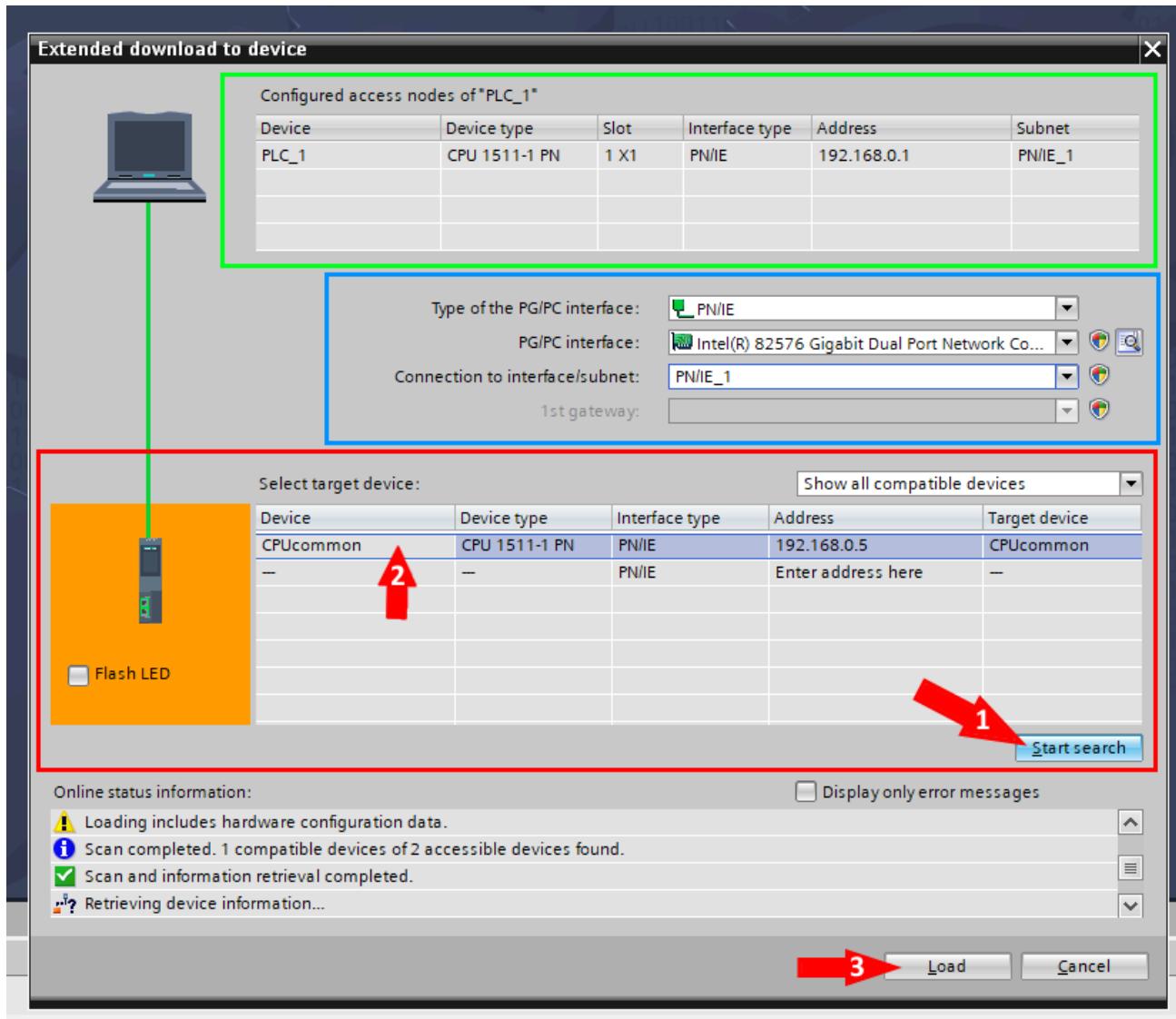
Now that we've configured our network, we should be able to run a simple test to make sure everything is working.

Download to your PLC

Right click on **PLC_1** in the device tree and select **Download to device > Software (all)**:



The **Extended download to device dialog** will appear.



ⓘ INFO

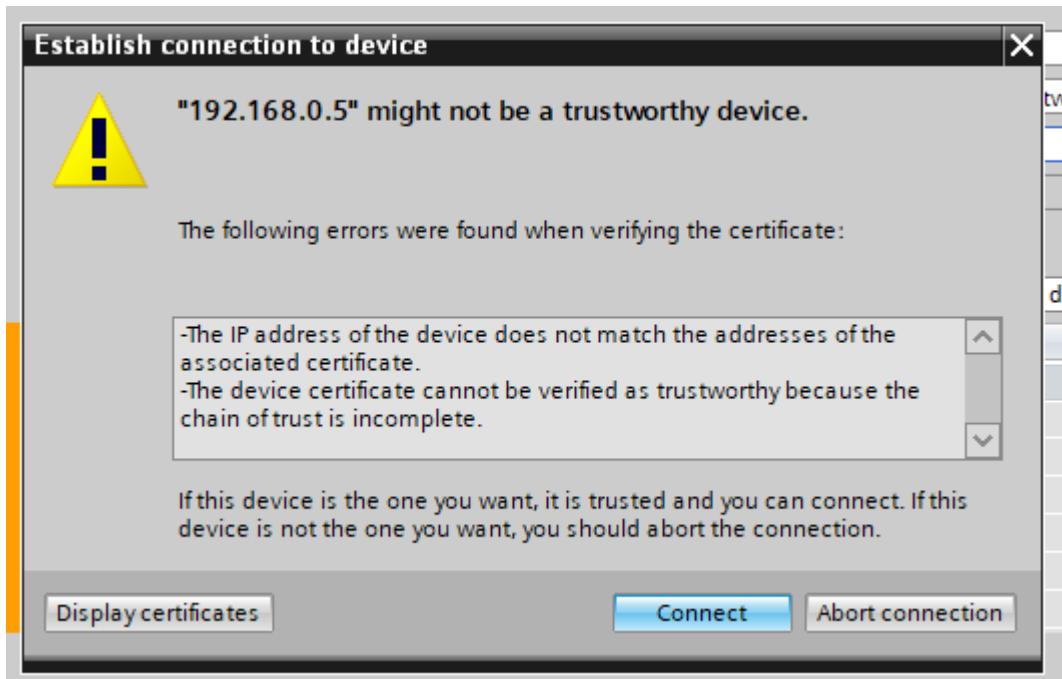
- The green box shows your provisioned network settings, in other words, what you *want* them to be.
- The red box shows the actual (current) network settings — the actual IP addresses of the PLC, Atom, etc.
- The blue box lets you configure the Ethernet interface on your PC and Profinet subnet to search for your PLC on.

Your PLC will automatically update the station names, IP addresses, and subnet masks that you set up in [network provisioning](#) when you start your PLC program, so you don't need to change them manually. The reason this dialog shows up is so that you can tell TIA which PLC to load the program on to.

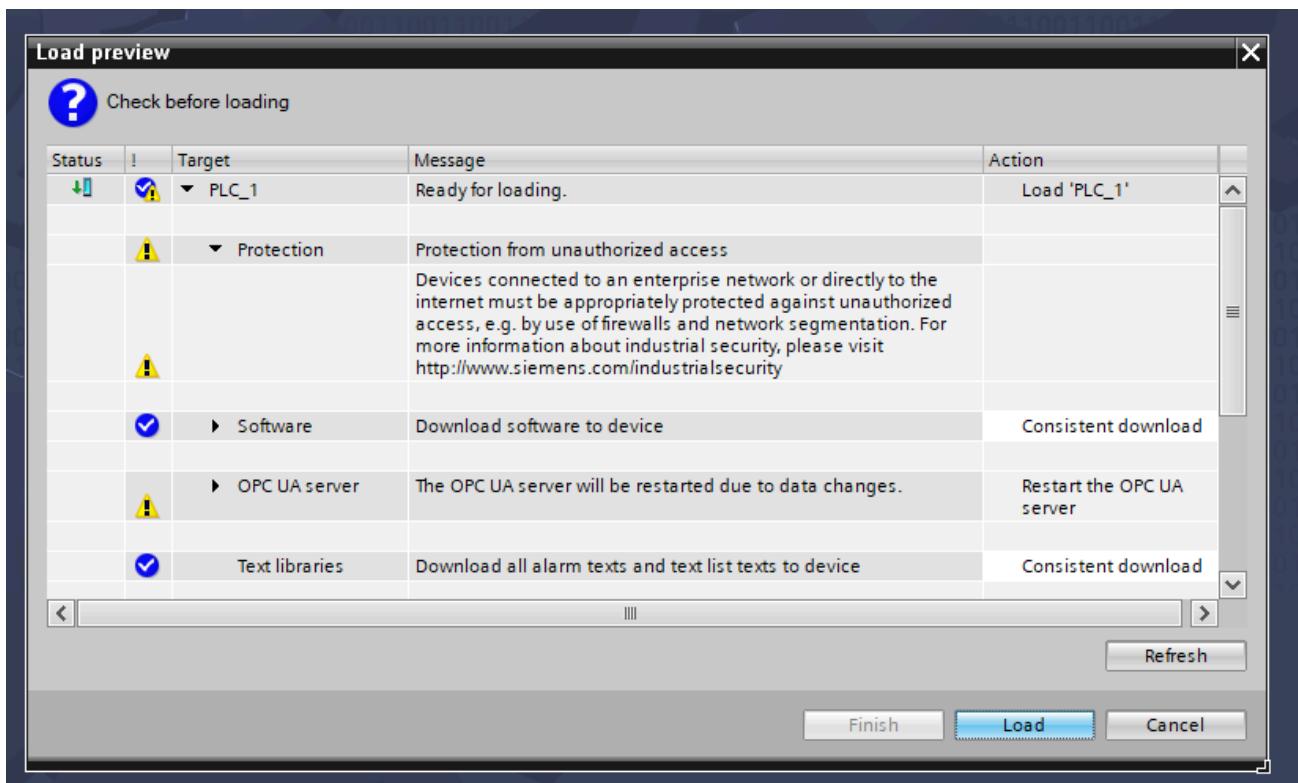
Make sure **Show compatible devices** is selected, click **Start search**, wait for your PLC to appear, select it, then click **Load**. Notice that our PLC's actual IP address [\(192.168.0.5\)](#) isn't the same as the IP address we provisioned for it [\(192.168.0.1\)](#).

This is fine, because we've linked our provisioned PLC to an actual PLC on the network, TIA knows whichs PLC to program and subsequently will update its network settings along with the network settings of all other Profinet devices on your provisioned network.

You may get a warning dialog like "**X.X.X.X** might not be a trustworthy device", click **Connect:**



The **Load preview** dialog will open, when it finishes preparing, click **Load**.

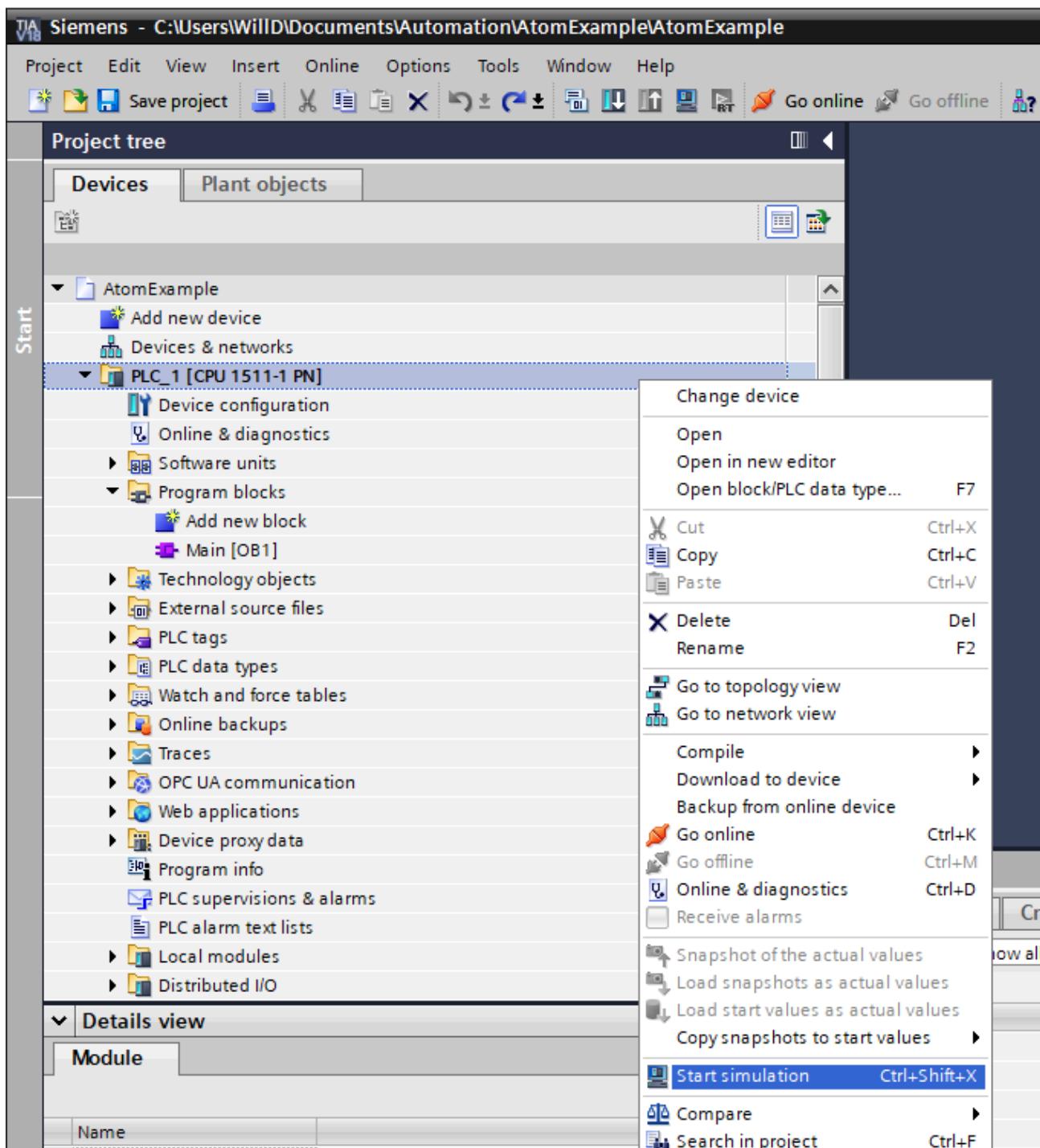


ⓘ INFO

If the load fails, see [Troubleshooting](#).

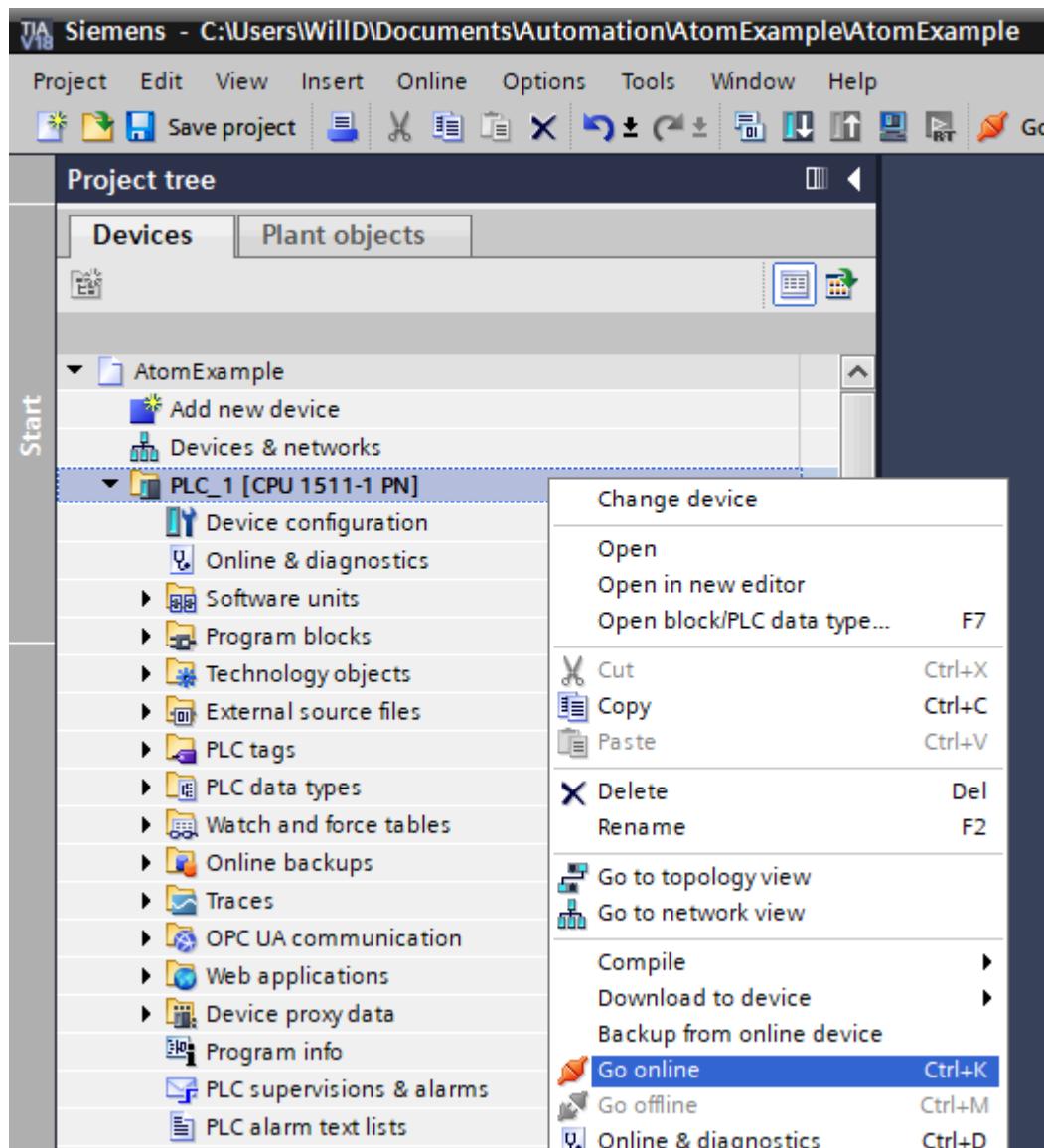
Use a simulator

If you don't have a real PLC handy, you can instead start the PLC simulator by right clicking **PLC_1** in the devices tree and selecting **Start simulation**:

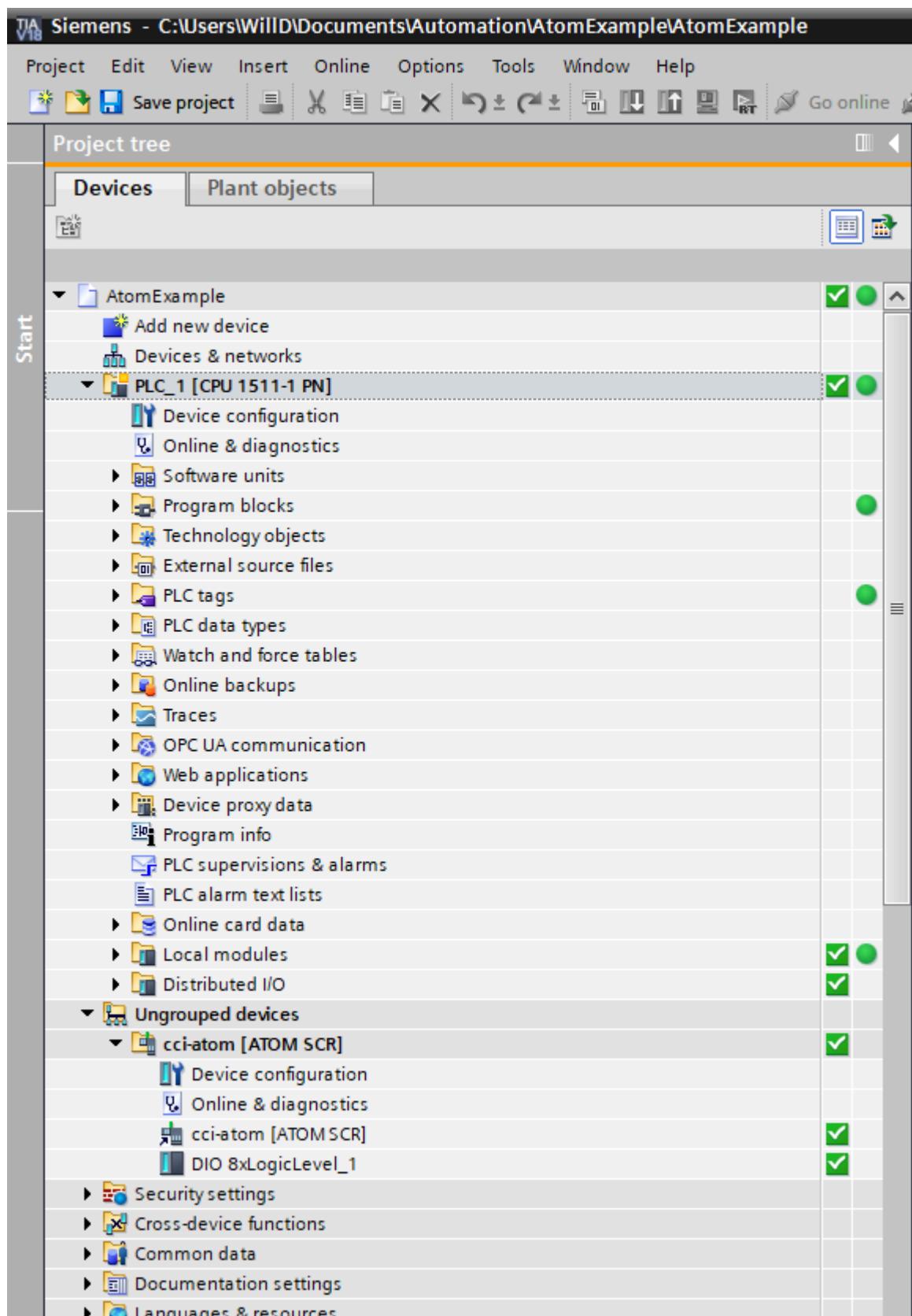


Monitor the heatsink temperature

Next, right click **PLC_1** in the device tree and select **Go online**:

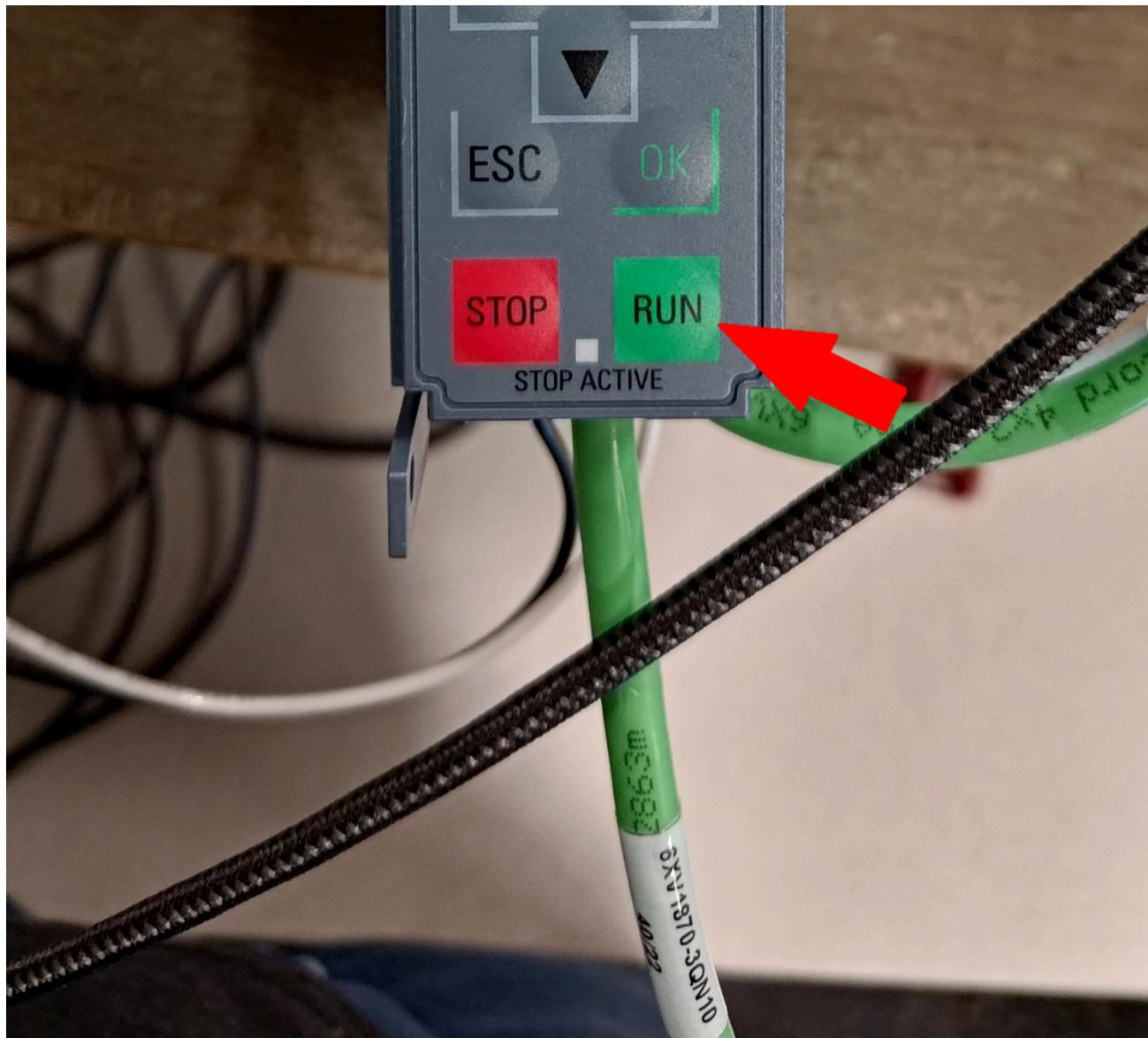


If everything goes well, your device tree should display green checkmarks over each device and their associated modules:

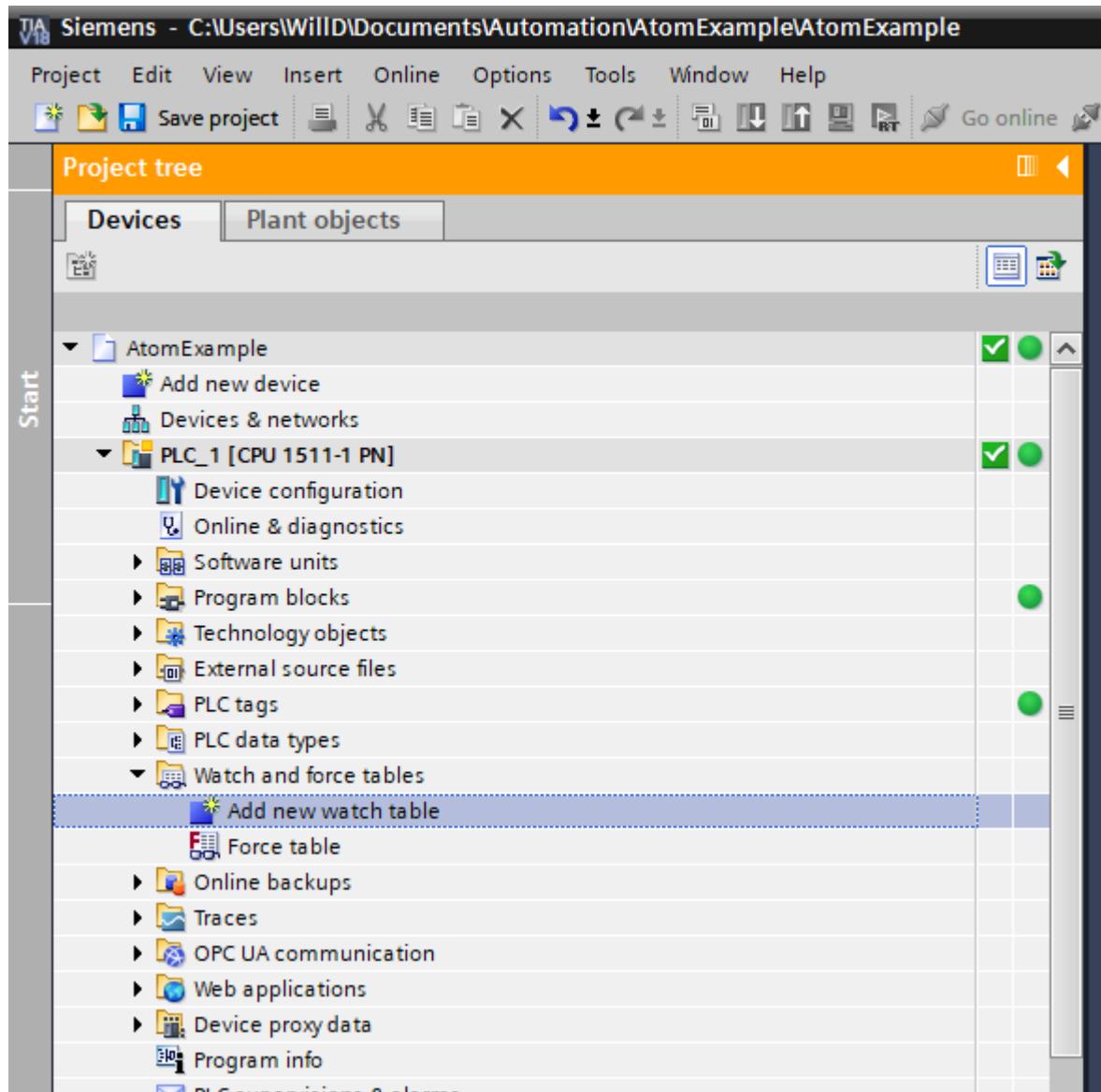


Open the case of your PLC and put it into **RUN** mode (or, if you're using the simulator, click **RUN** in the PLC simulator popup):

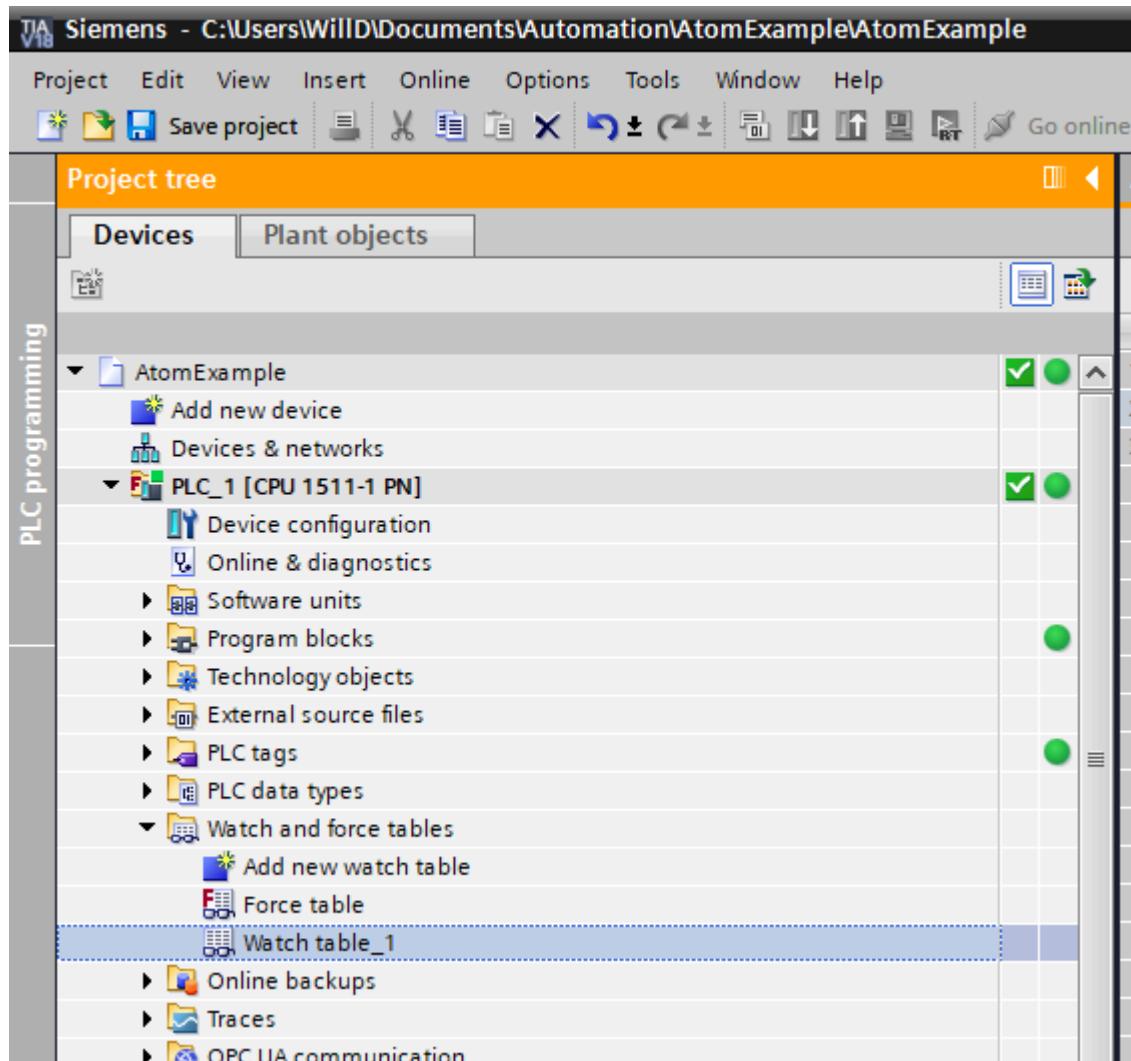




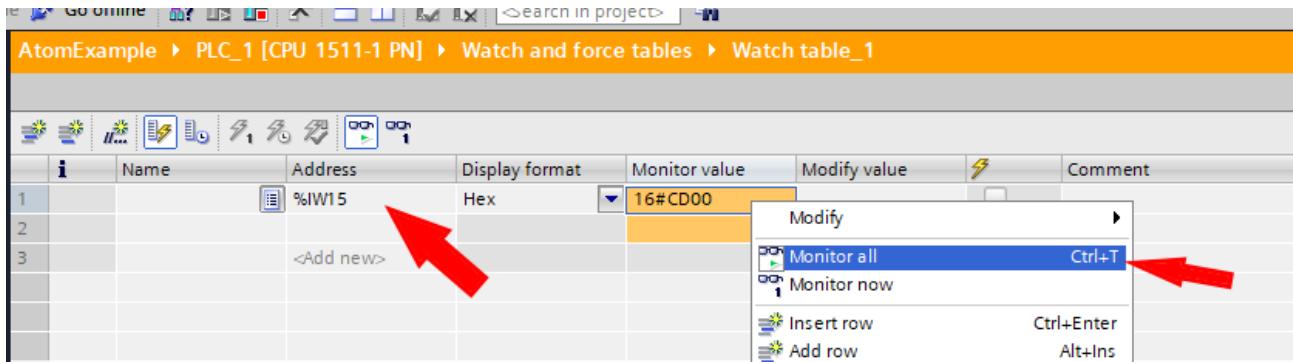
Next, expand **Watch and force tables** under **PLC_1** in the device tree and double click **Add new watch table**:



Double click the newly created watch table **Watch table_1** to open it:



On the first row, enter `%IW15`, then right click the row and select **Monitor all**. `%IW15` is short for input word #15, which corresponds to the low-order word of the heatsink temperature parameter. After clicking **Monitor all**, you should see the value update to a non-zero value. If it does, this means your PLC is successfully talking to Atom over Profinet!

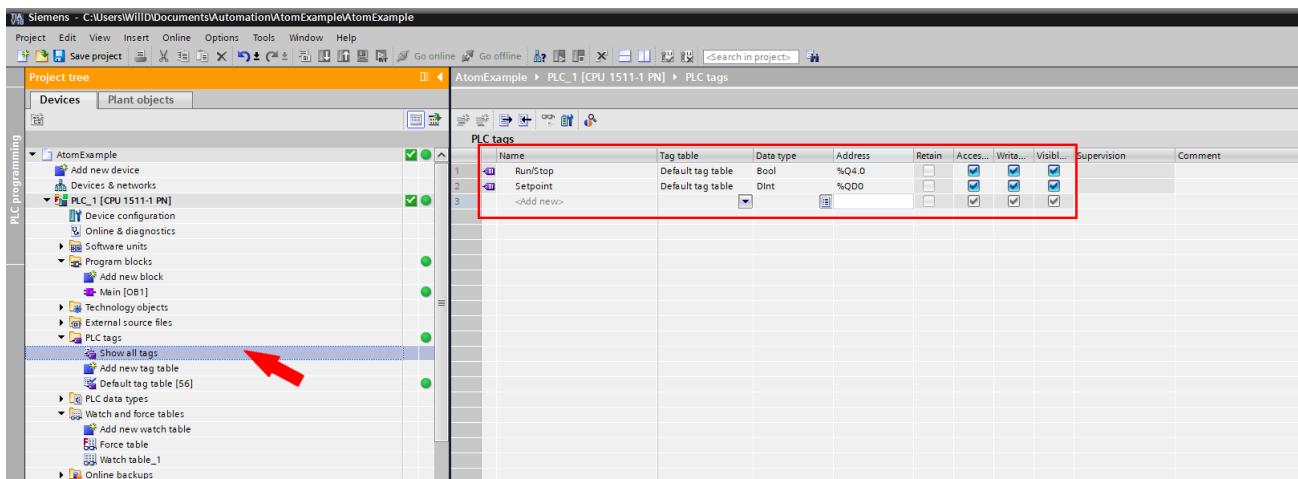


Building a simple PLC program to control Atom

Writing some ladder logic

In the devices tree, expand **PLC_1 > PLC tags** and double click **Show all tags**. Create two tags:

- Tag #1
 - Name: Run/Stop
 - Data type: Bool
 - Address: %Q4.0
- Tag #2
 - Name: Setpoint
 - Data type: Dint
 - Address: %QD0



Next, in the device tree, expand **PLC_1 > Program blocks** and double click **Main [OB1]**.

First, create two constants:

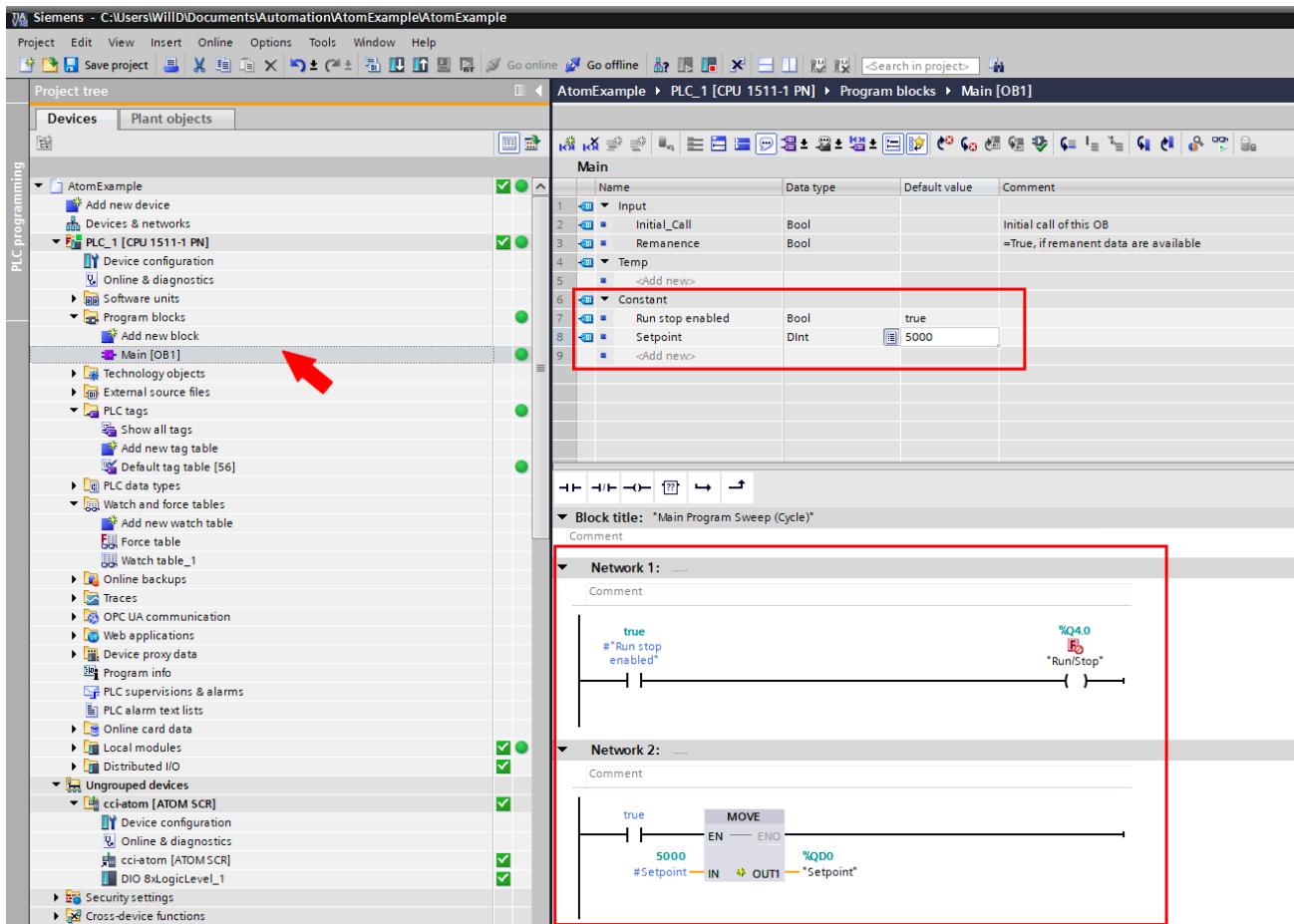
- Name: `Run_stop_enabled`, Data type: `Bool`, Default value: `true`
- Name: `Setpoint`, Data type: `Dint`, Default value: `5000`

These are the values we will ultimately set to Atom.

Next, create two networks.

- Network #1
 - Insert a **contact** on the left, and drag the `Run_stop_enabled` constant into it.
 - Insert an **assignment** on the right, and enter `%Q4.0`
- Network #2
 - Insert a **MOVE** block
 - Insert a **contact** on the **EN** input, and set its value to `true`
 - Drag the `Setpoint` constant into the **IN** input
 - Enter `%QD0` into the **OUT1** output

After you're done, your PLC program should look like this (you can also download the [example project](#) with the completed program):

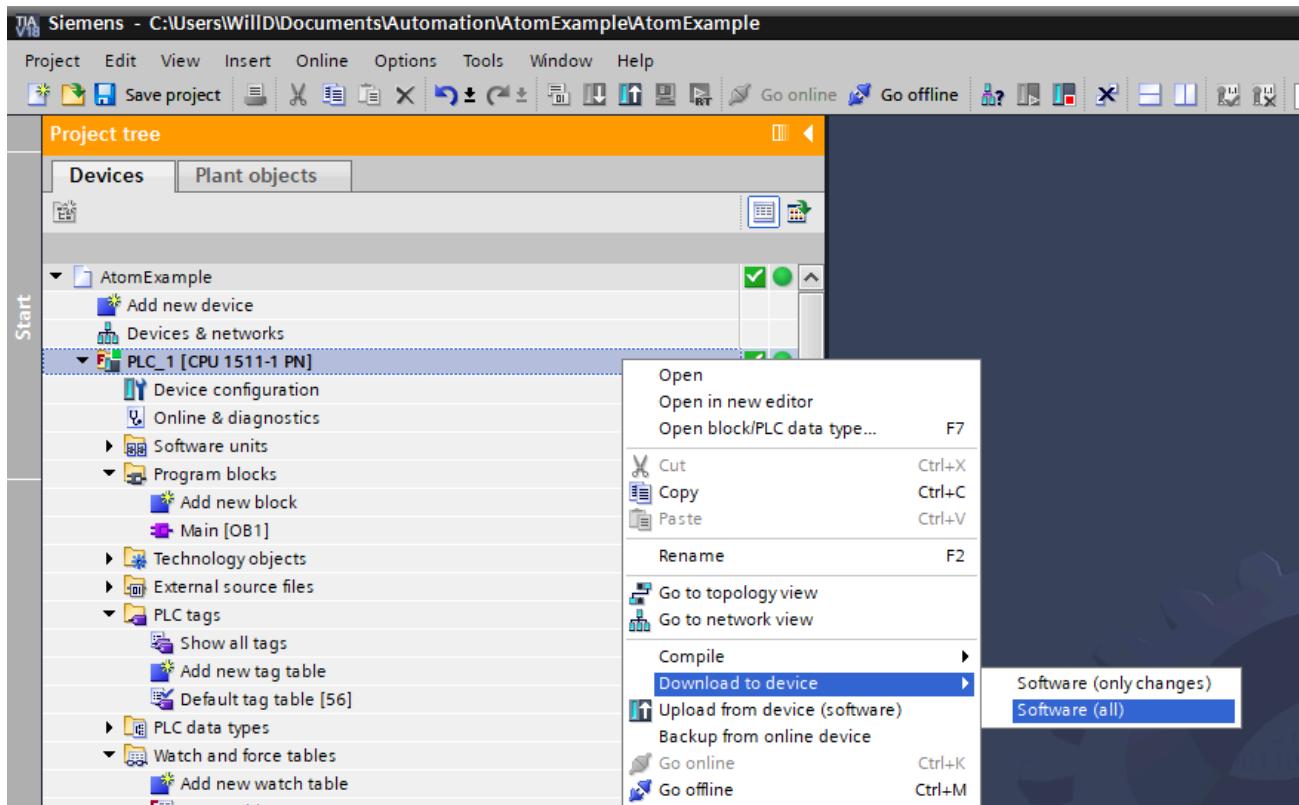


Running on your PLC

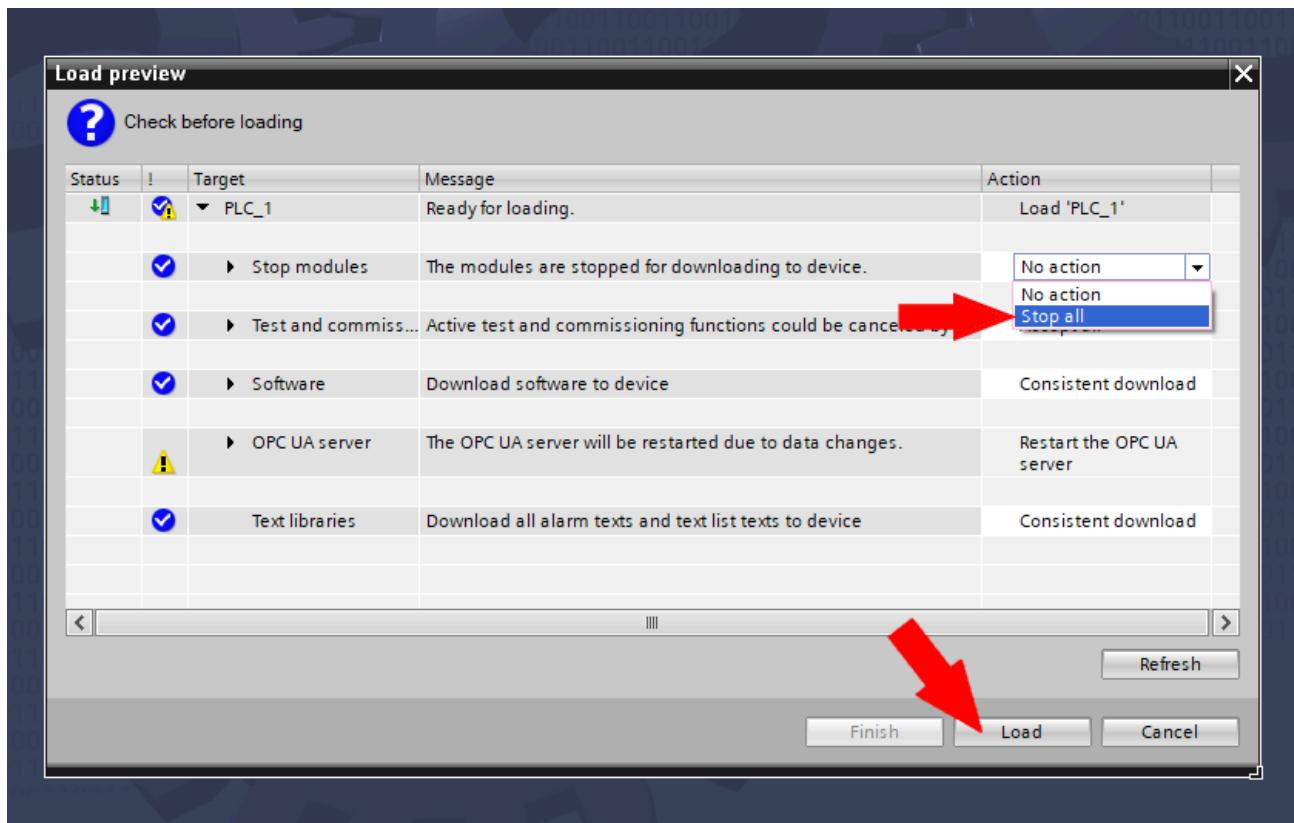
INFO

This will still work if you're running a simulator, make sure you started the simulation as shown in [Use a simulator](#).

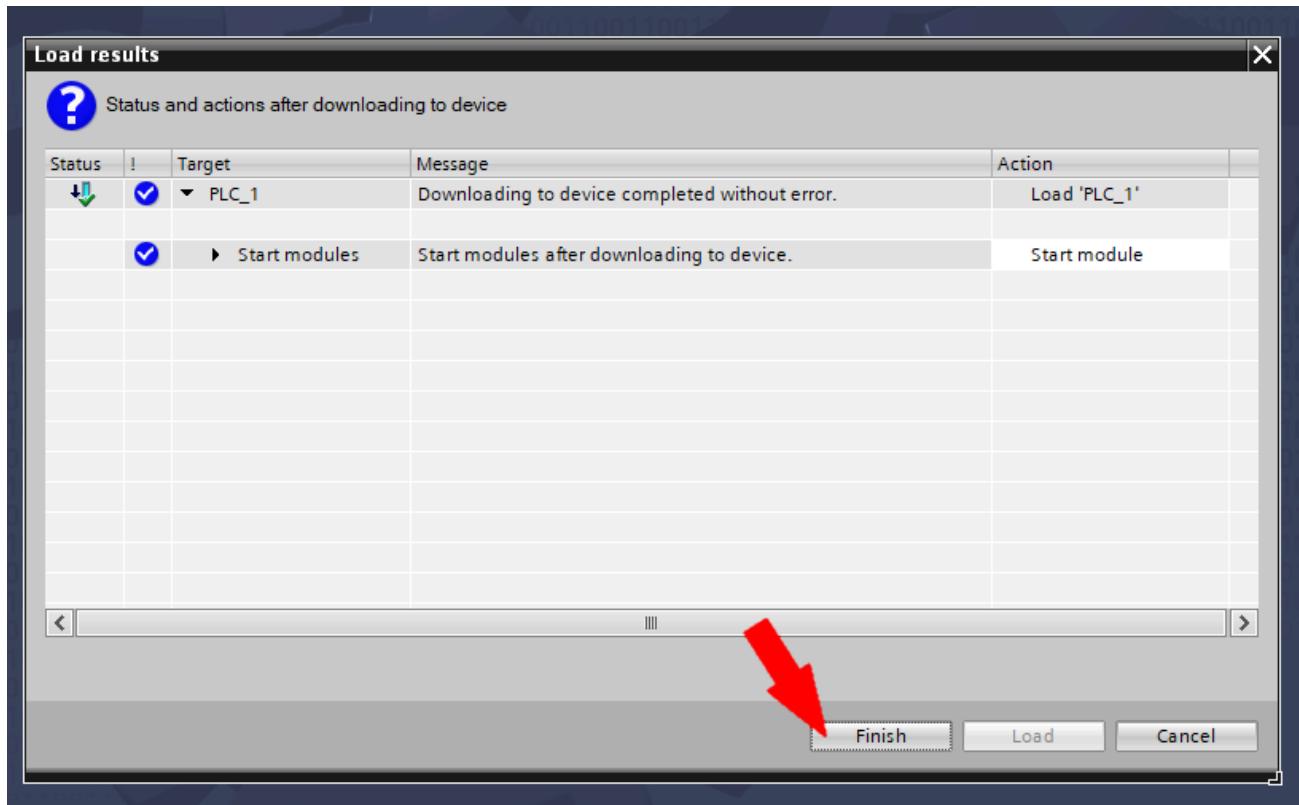
Right click **PLC_1** in the devices tree, select **Download to device > Software (all)**:



A **Load preview** dialog will show up, if your PLC was previously running, you may have to set the **Stop modules** action to **Stop all**. Then, click **Load**:



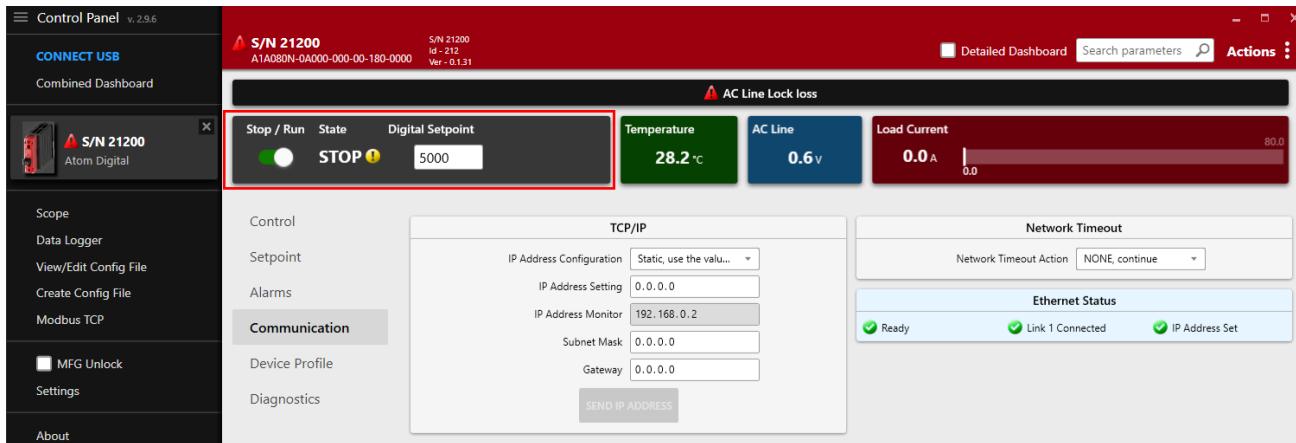
If the load fails, check out [Troubleshooting](#). Otherwise, click **Finish**:



Make sure your PLC is set to RUN. If everything worked, the PLC will put Atom into RUN with a setpoint of 5000. If you connect a USB cable to your Atom and look in Control Panel, you should see the setpoint and run/stop parameters update.

INFO

Try switching the **Stop / Run** switch off in Control Panel or updating the setpoint. Notice that the PLC immediately sets the run/stop and setpoint parameters back to their original values.

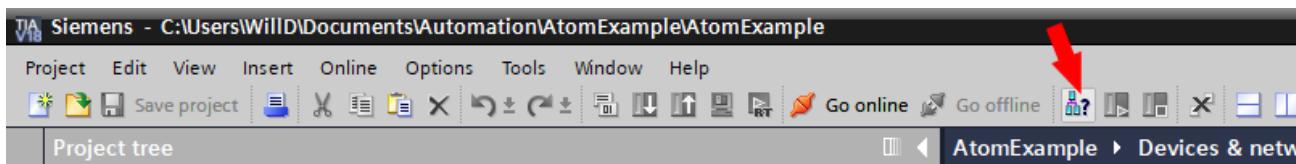


Troubleshooting

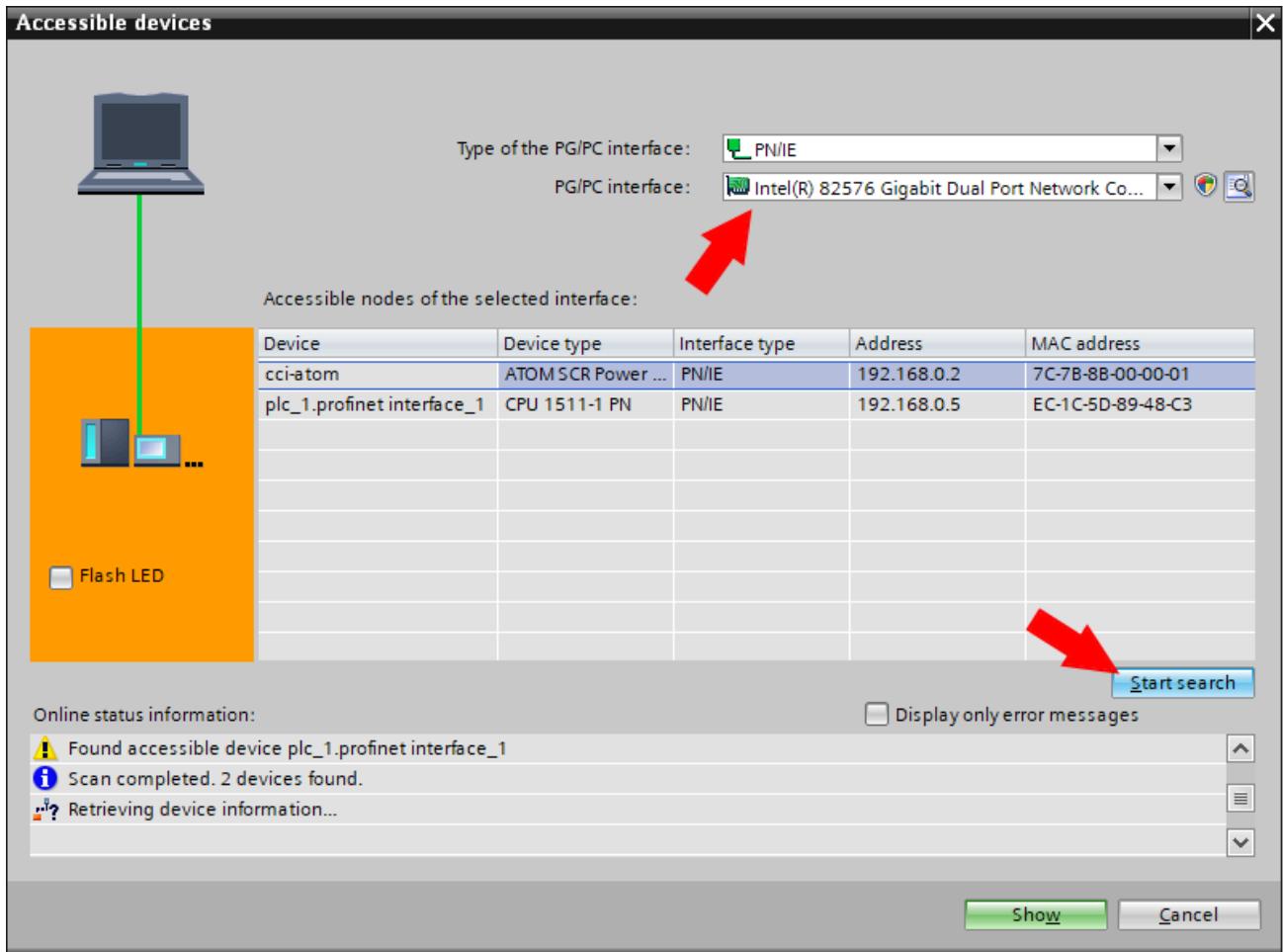
Download to PLC fails

If the download to your PLC fails, you can try resetting your PLC.

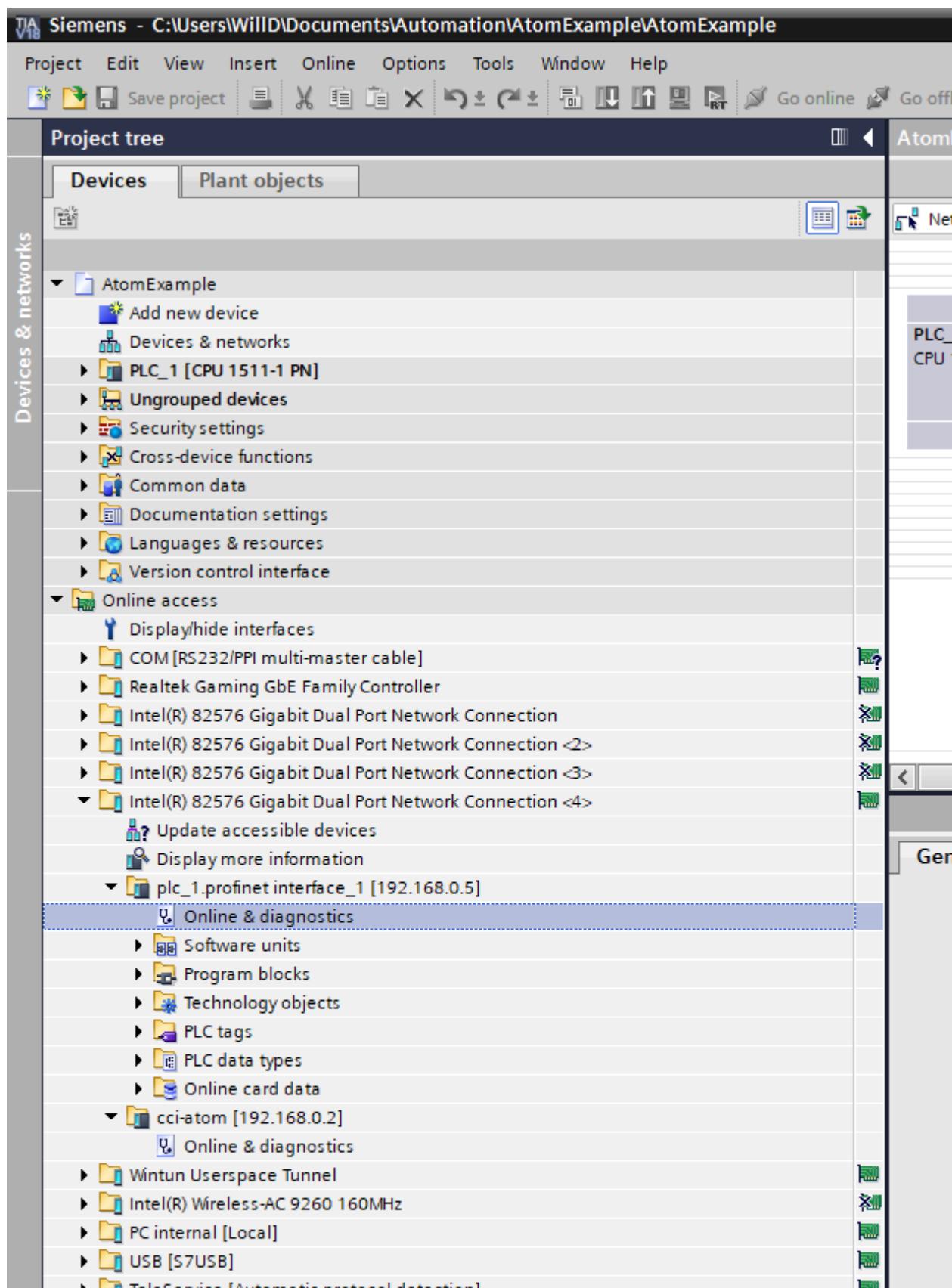
First, click the **Accessible devices** icon in the menu bar:



When the **Accessible devices** dialog appears, select the network adapter on your PC that is connected to your PLC and click **Start search**:

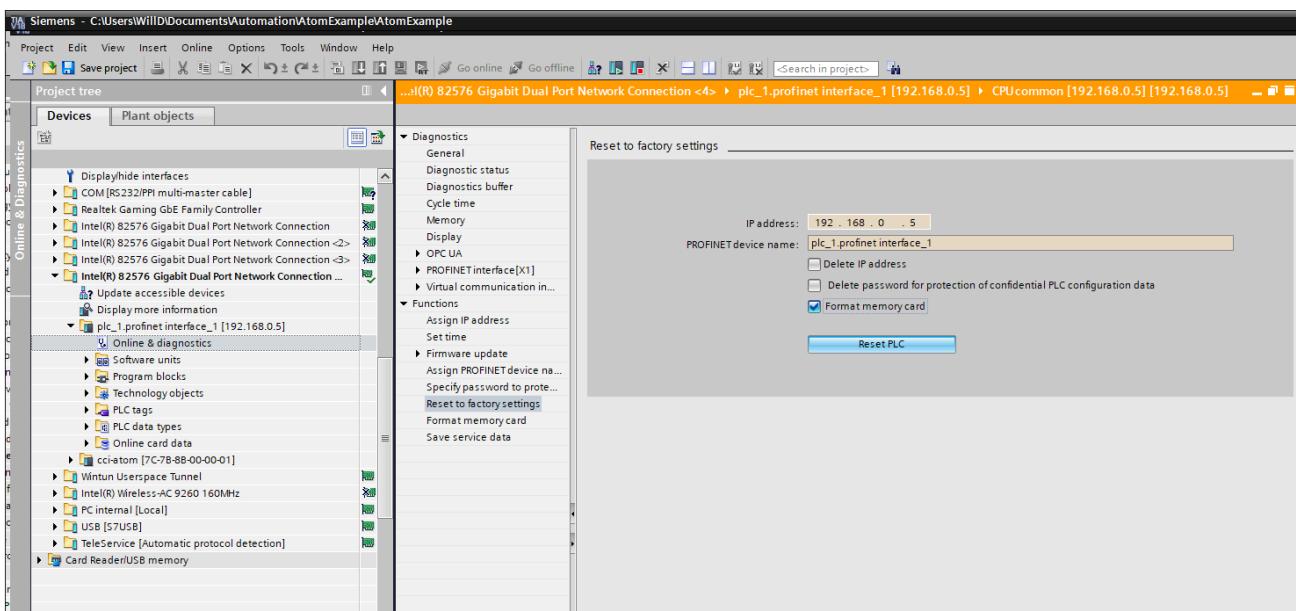


In the devices tree, expand **Online access** > Your PC's network adapter connected to your PLC > **plc_1.profinet interface_1 (192.168.0.05)** (may be different for you) > **Online & diagnostics:**





Expand **Functions > Reset to factory settings**, check **Format memory card**, then click **Reset PLC**:



ATOM / Fieldbus / PROFINET / Codesys

In this tutorial, you'll learn how to use Codesys with the SoftPLC emulator to connect to ATOM using Profinet and perform some basic operations and monitor data. You can follow along using the SoftPLC emulator or your own PLC.

We provide examples for both ladder logic and structured text.

If you haven't yet, please review ATOM's [Profinet Profile](#).

If you'd like to skip the tutorial, you can download a completed example project:

- Download [ATOM_Codesys_Profinet_LadderLogic_Example.zip](#)
- Download [ATOM_Codesys_Profinet_StructuredText_Example.zip](#)

Prerequisites

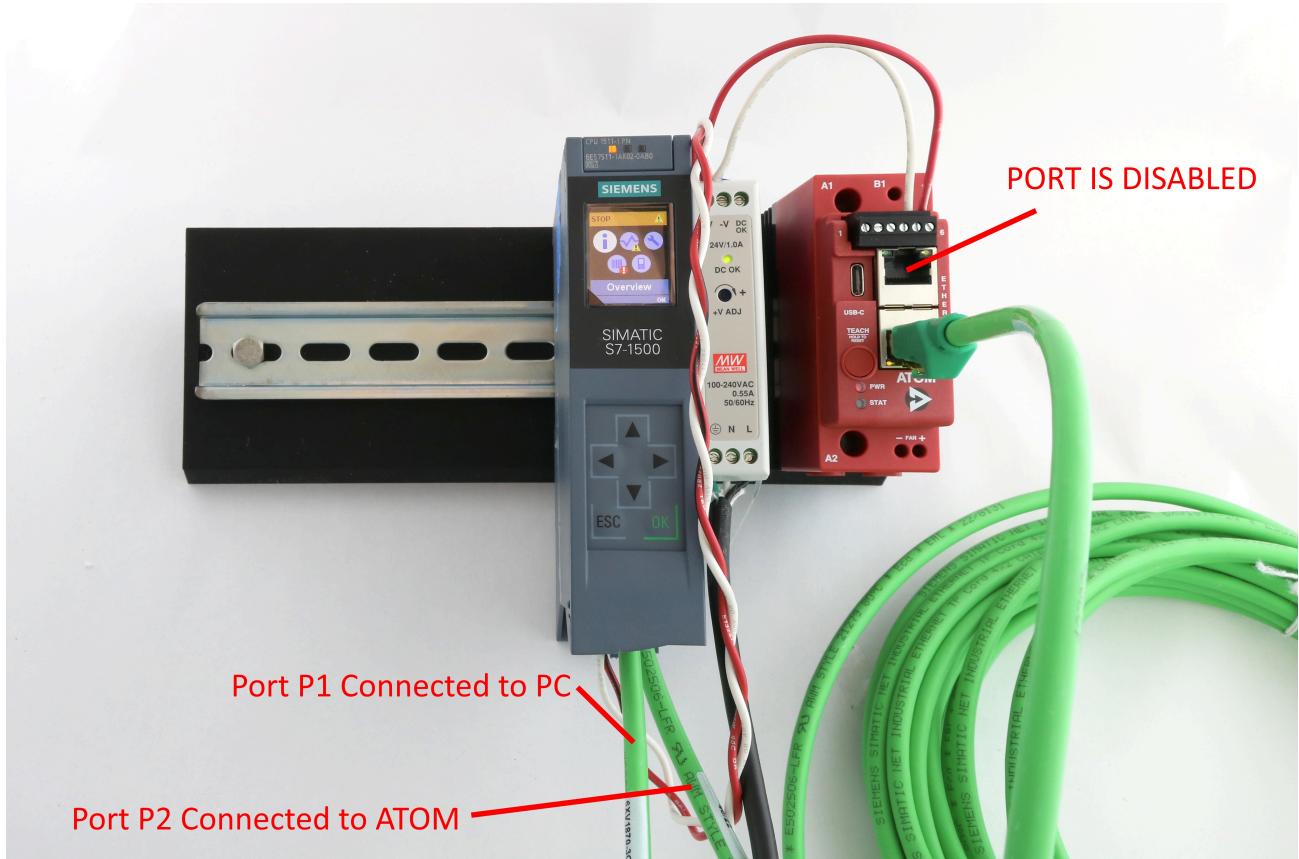
1. Install [Codesys](#)
2. Download ATOM's [GSDML file](#)

Hardware setup

IMPORTANT

When Atom is configured for Profinet, the Ethernet port closest to the 24V power connector is **disabled**. You must use opposing Ethernet port nearest the reset button as shown below or the PLC won't be able to connect to Atom.

Connect 24V to your PLC and Atom unit with the provided power cable. Connect Atom to your PC with an Ethernet cable.



ⓘ INFO

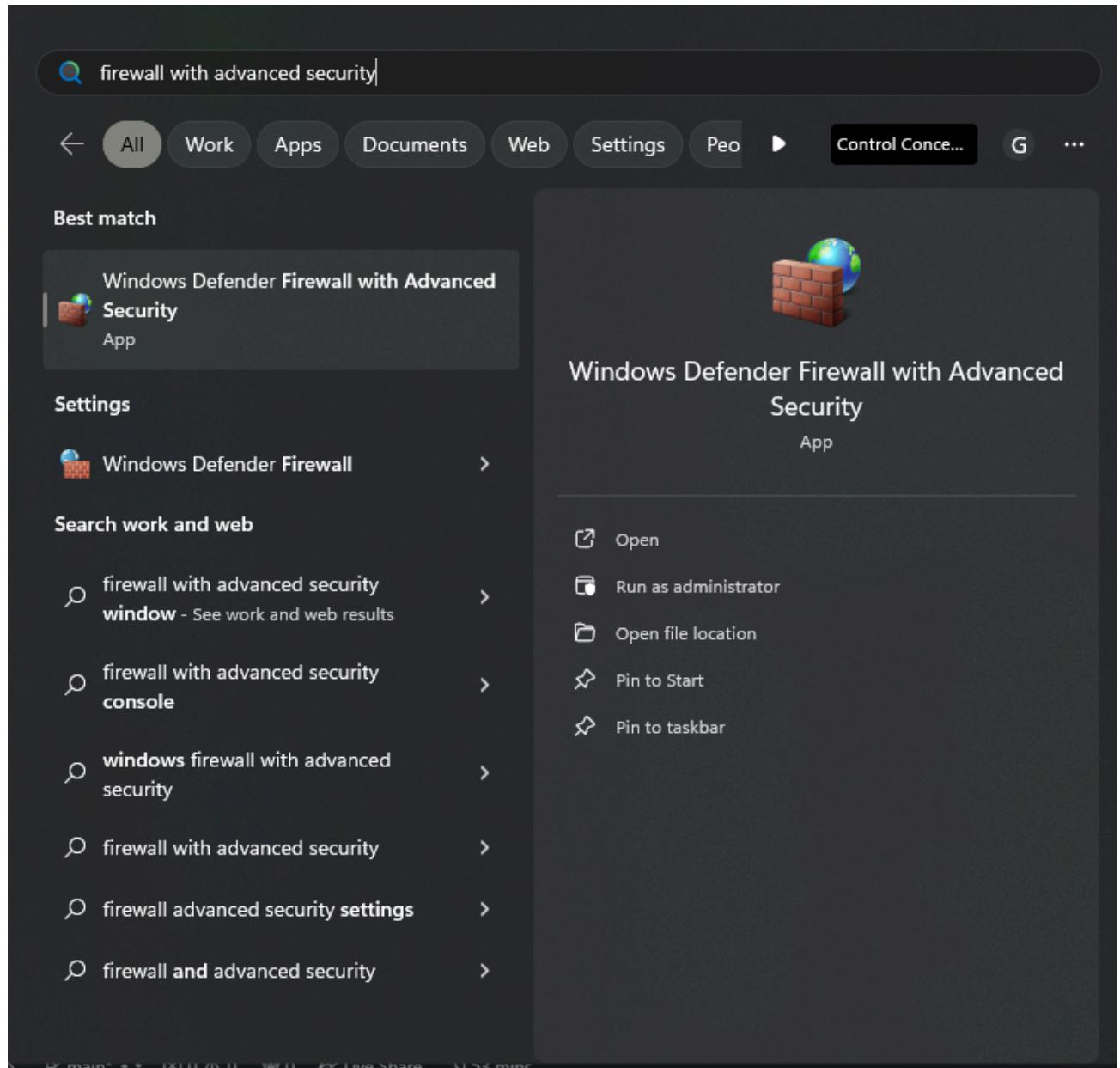
To simplify this diagram, we have not connected a load to Atom. You may connect a load or leave it disconnected, either way is fine for the purposes of this tutorial.

If you do not connect a load, you can still verify your PLC is working by connecting a USB cable to Atom and using Control Panel to watch the parameters change/verify the PLC is receiving the correct monitor data.

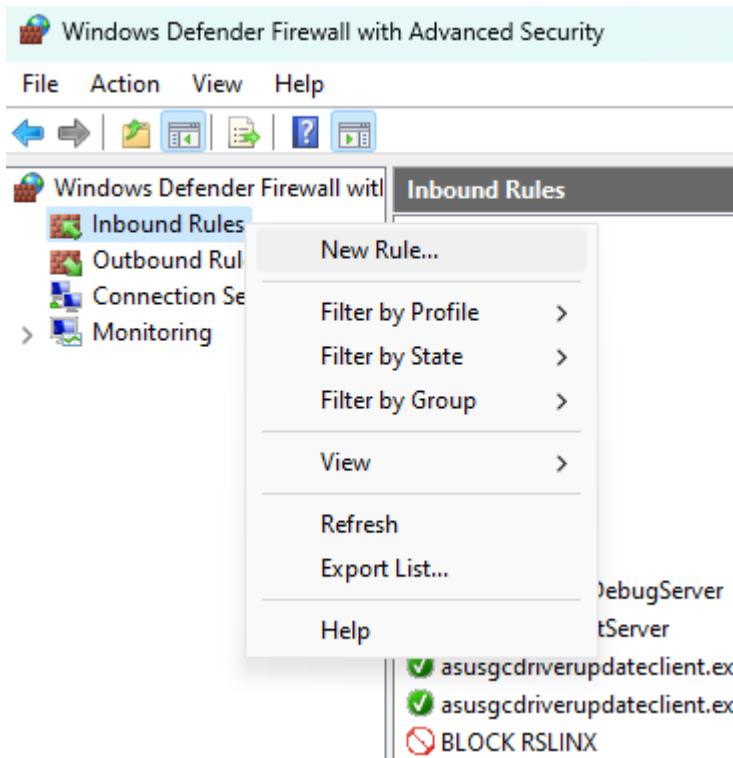
Configure Windows firewall

Codesys requires you to allow incoming Profinet UDP packets through the Windows firewall so that the SoftPLC is able to receive UDP Profinet requests from Atom.

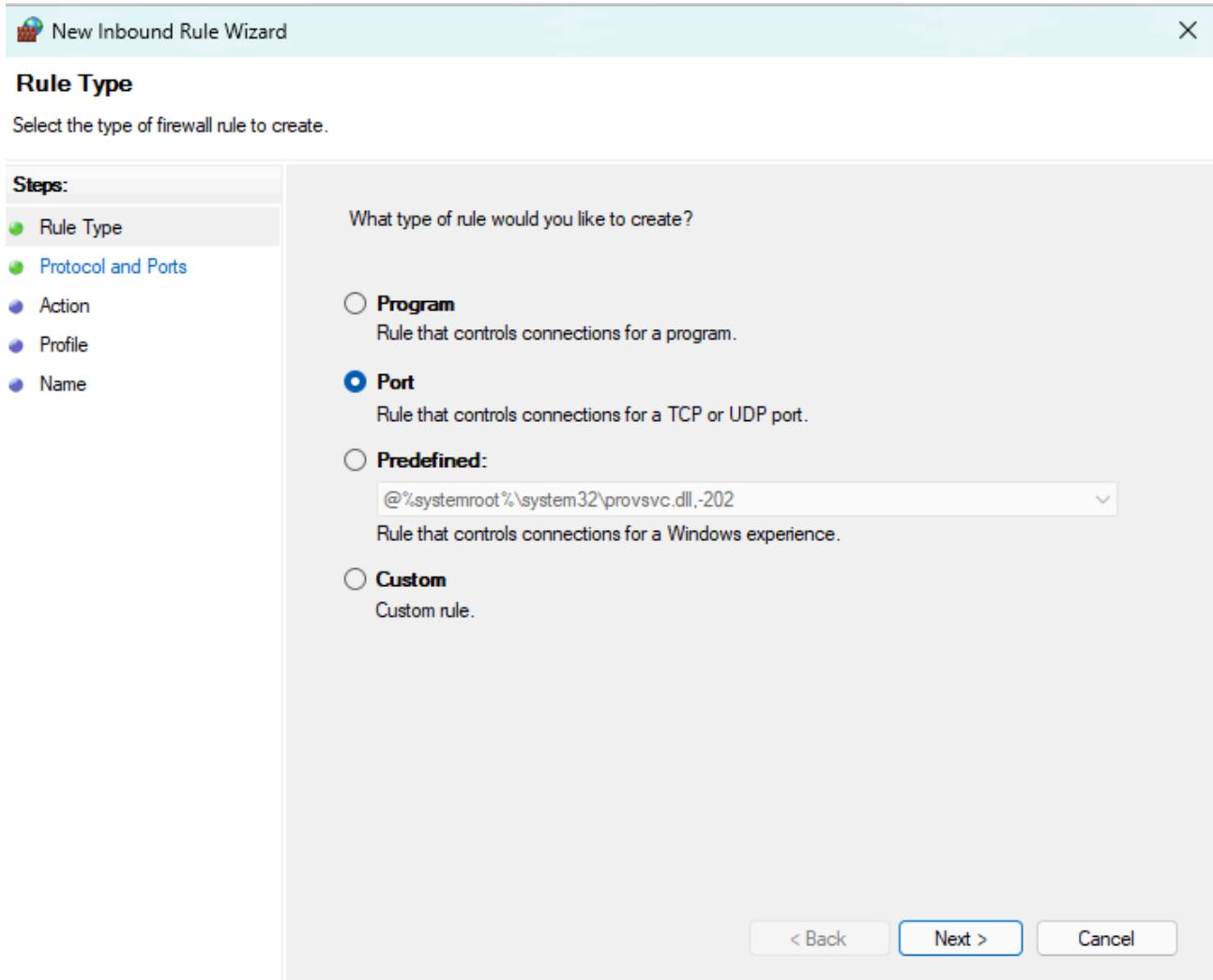
1. First, search for **Windows Defender Firewall with Advanced Security** and open it:



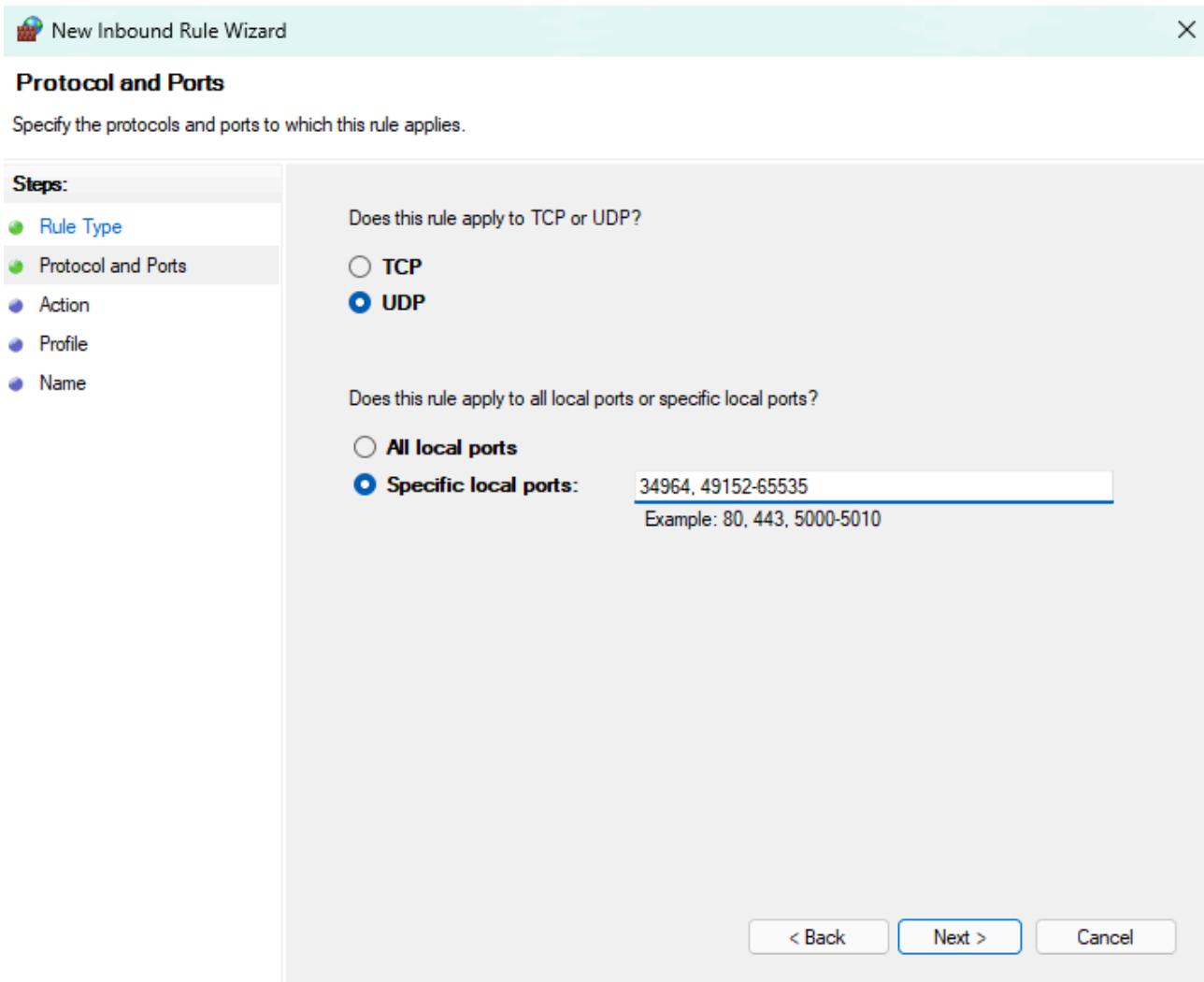
2. Right click on **Inbound Rules** and select **New Rule**:



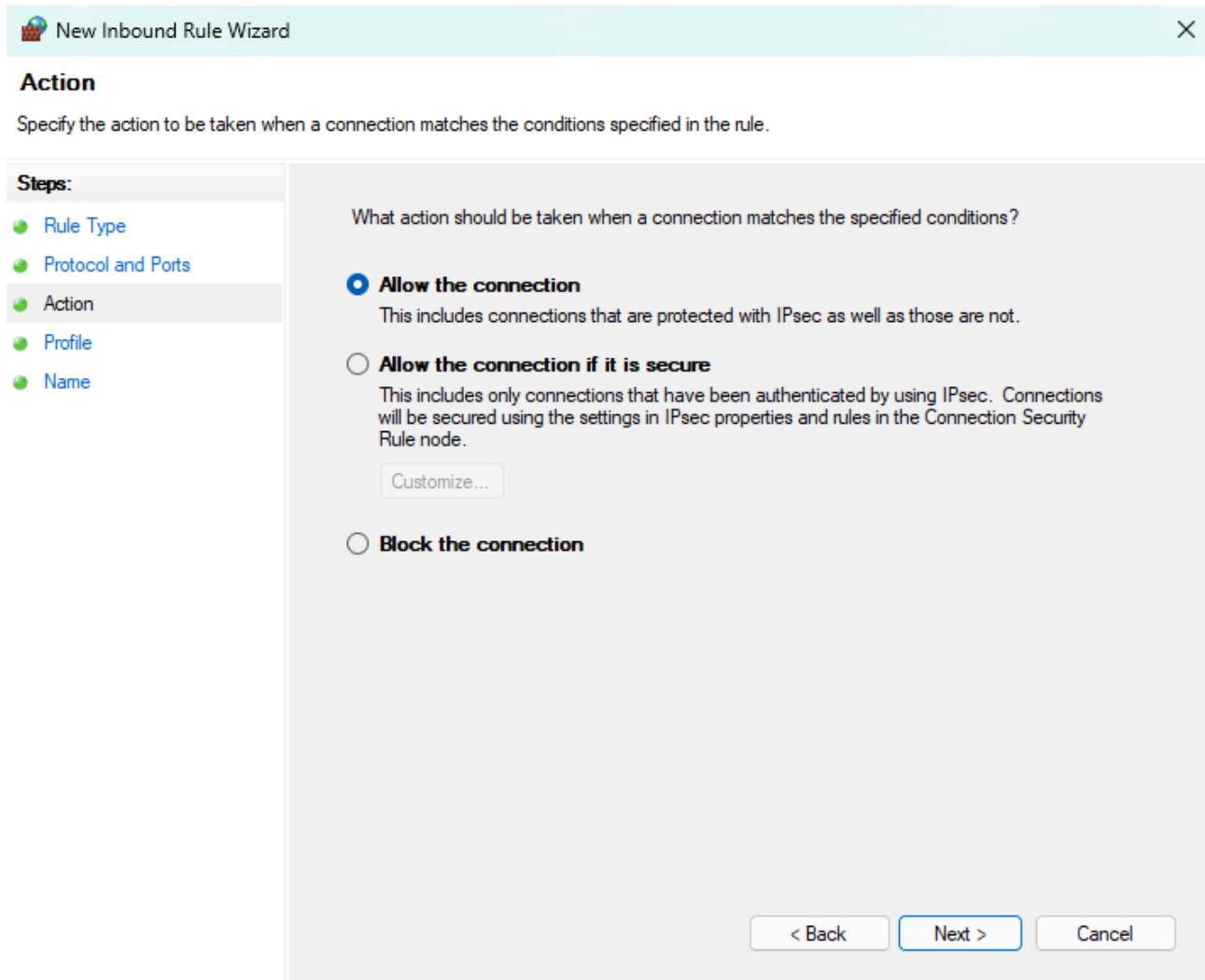
3. Select **Port**, then click **Next**:



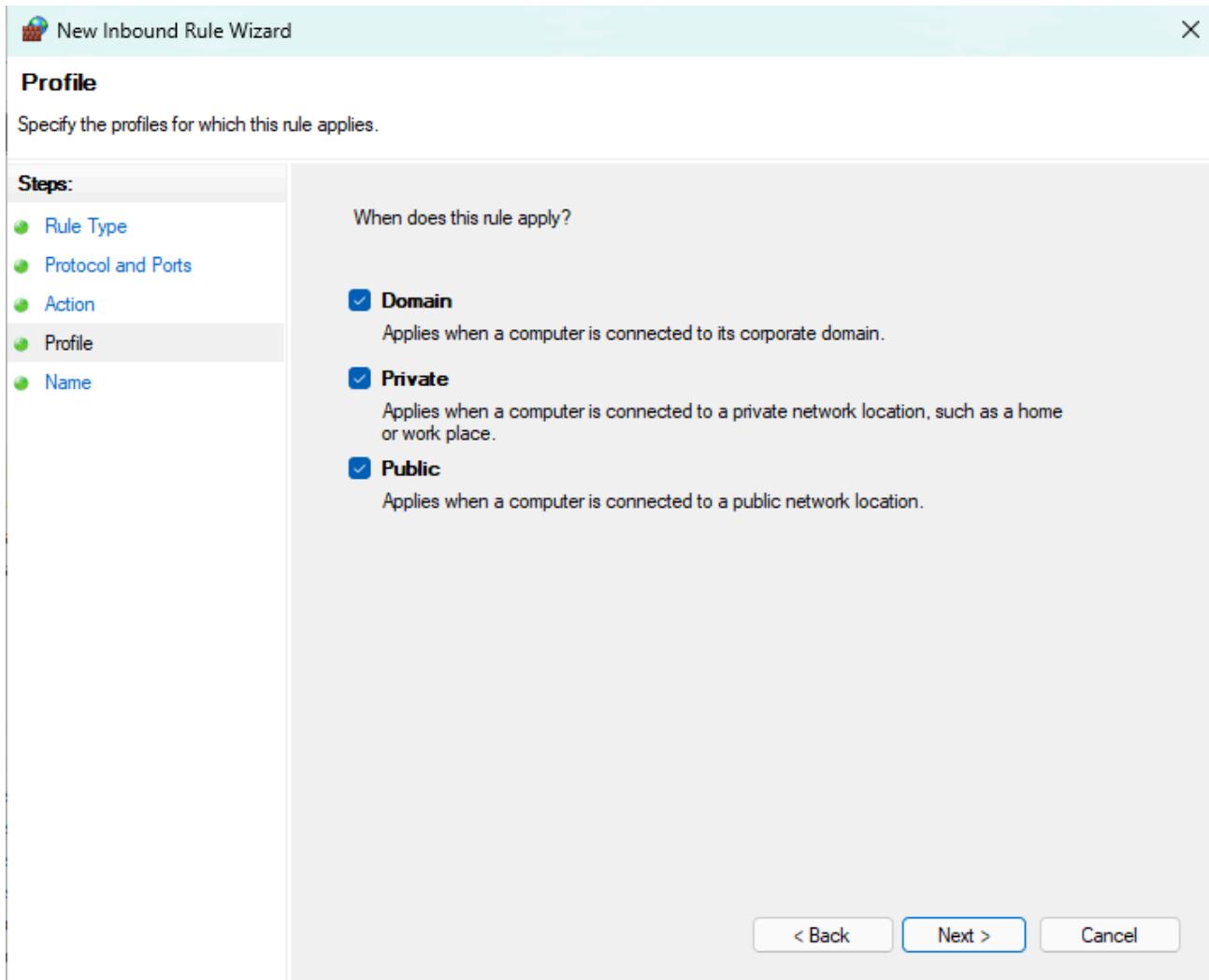
4. Select **UDP**, then select **Specific local ports** and enter **34964, 49152-65535**, then click **Next**:



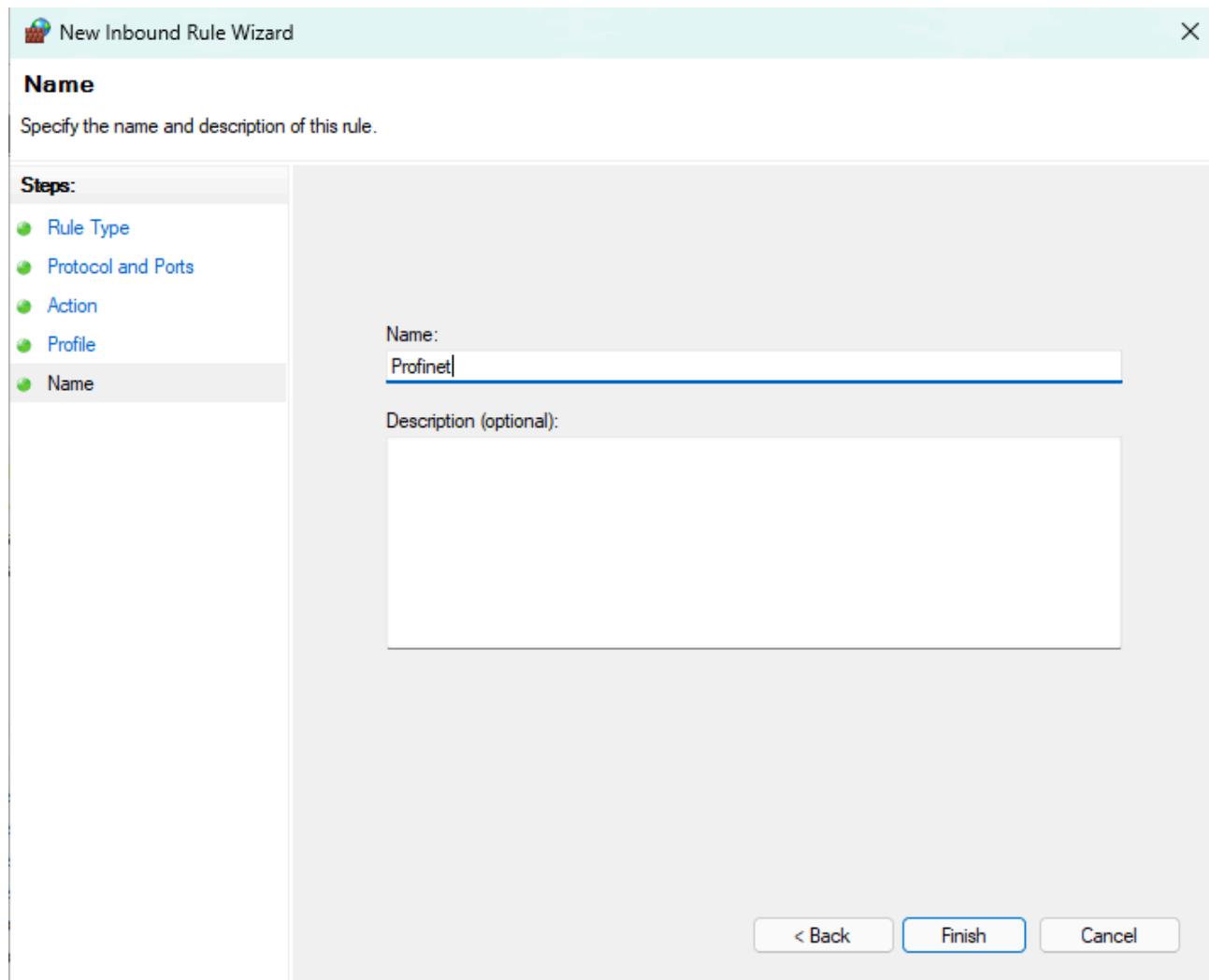
5. Select **Allow the connection**, then click **Next**:



6. Select which network types this rule applies to, then click **Next**:

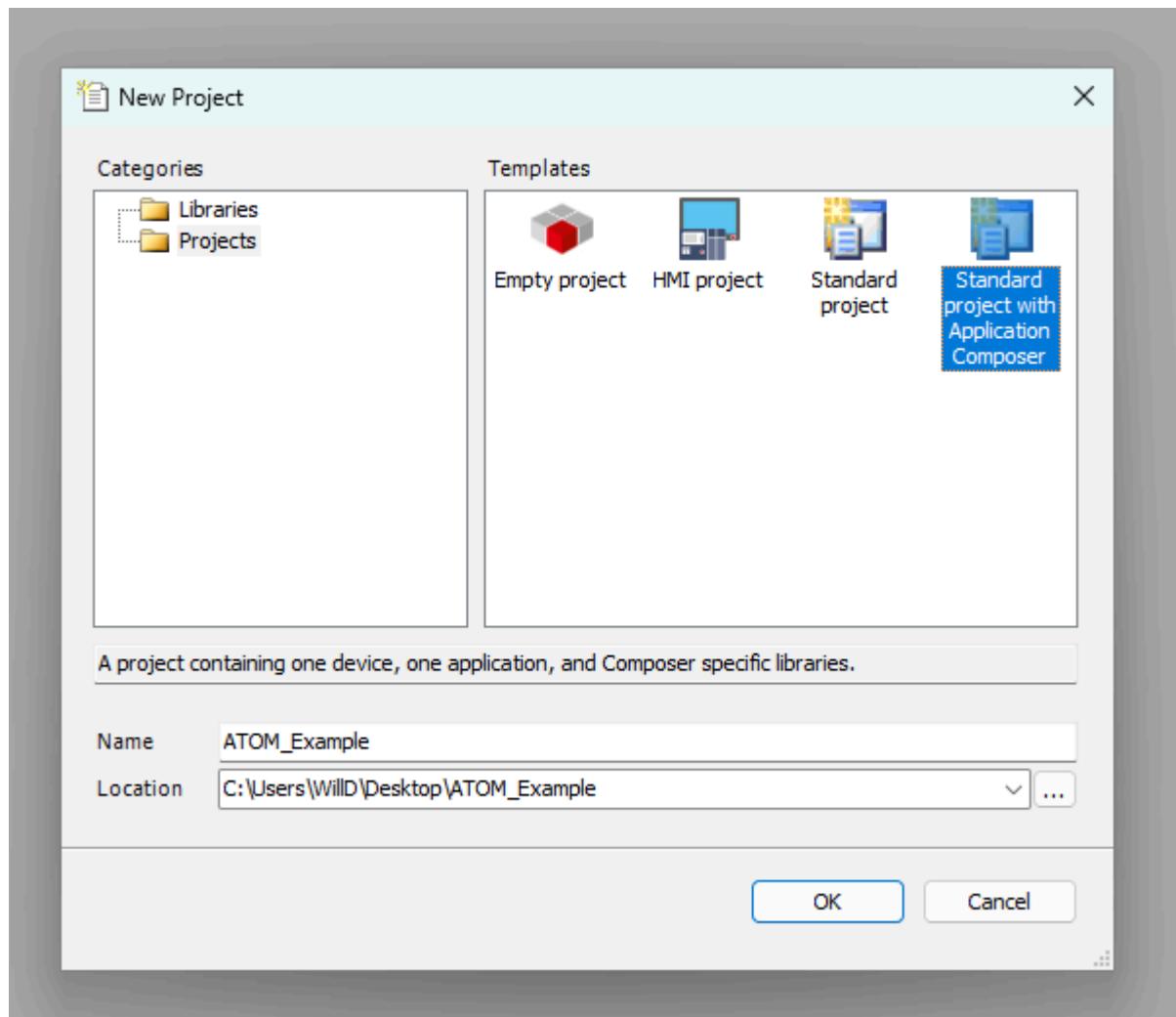


7. Name the rule **Profinet**, then click **Finish**:

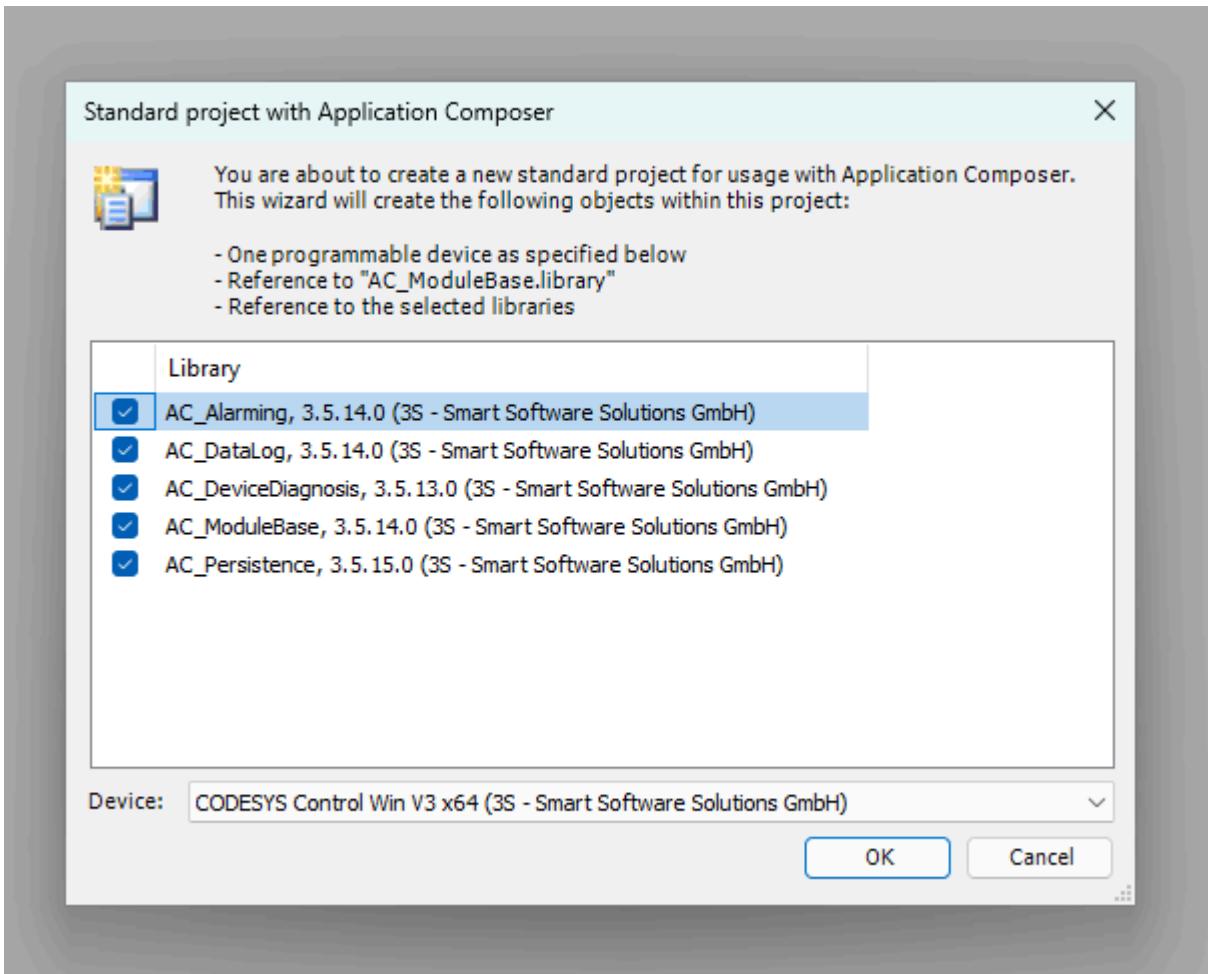


Create a Codesys project

Create a new Codesys project using the **Standard project with Application Composer** template:



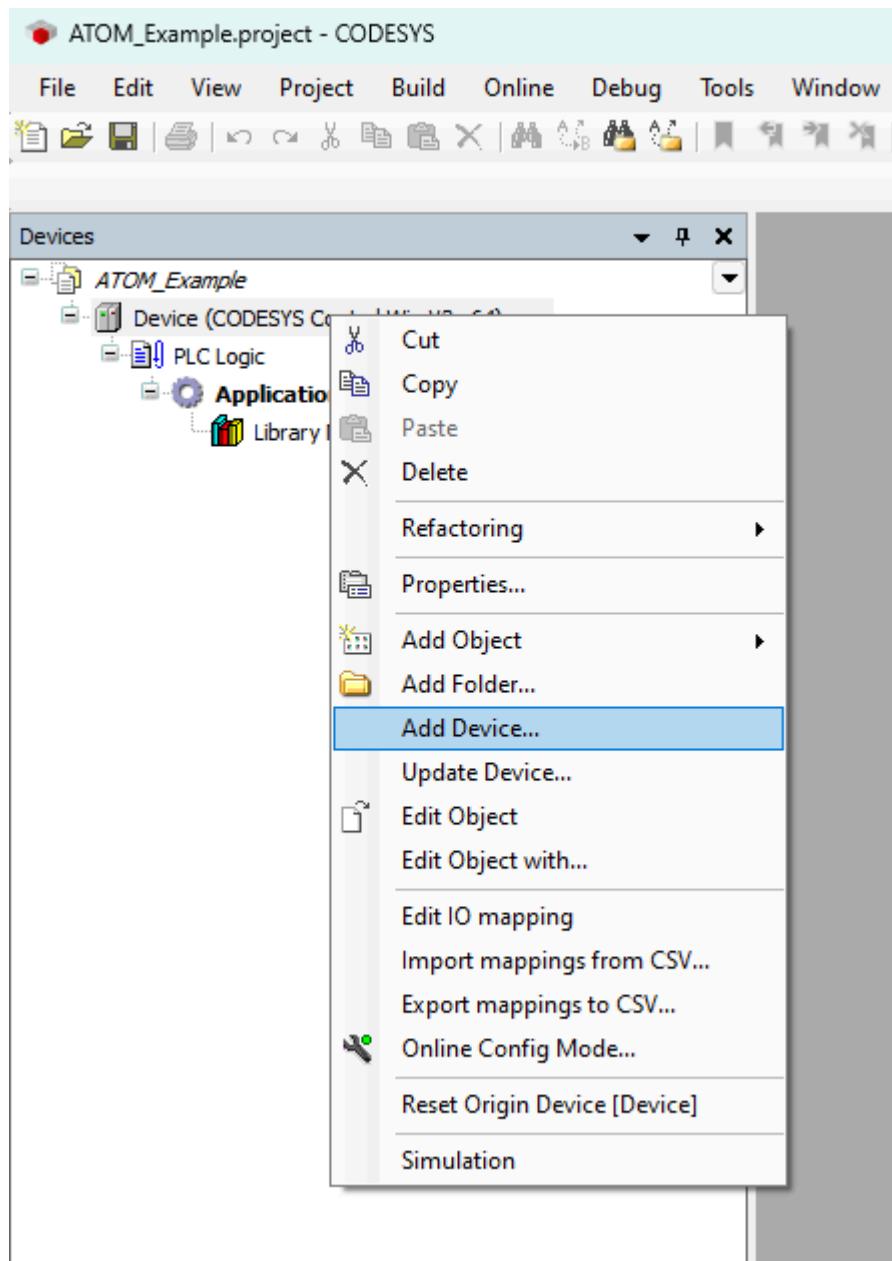
Check each library to include it in the project and select **CODESYS Control WIN V3 x64** as the device:



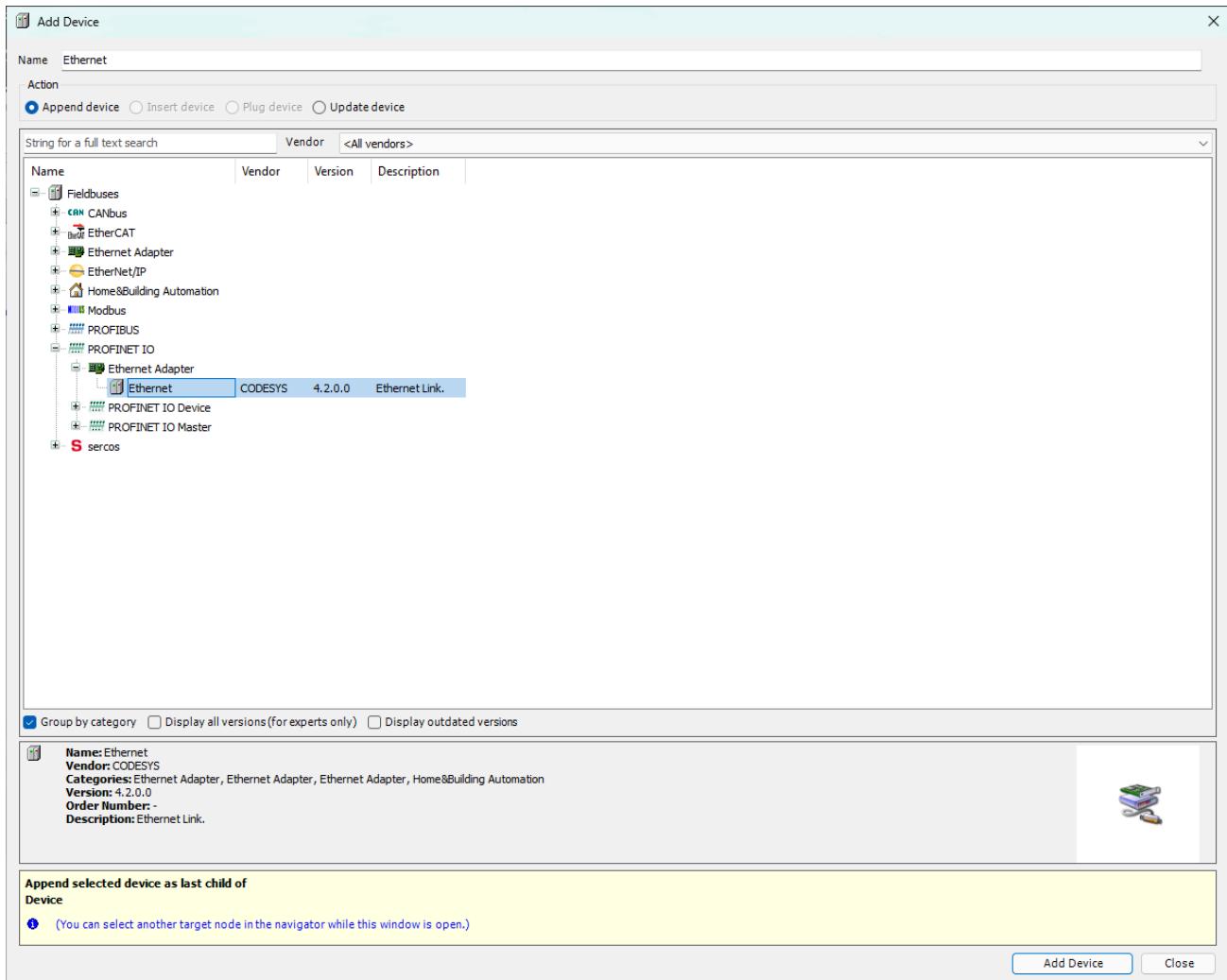
Adding a Profinet Controller

Next we'll add a Profinet Controller device. This allows the SoftPLC to discover Profinet I/O devices on the network (in our case, ATOM) and establish a connection with them.

First, right click **Device** and select **Add Device**:



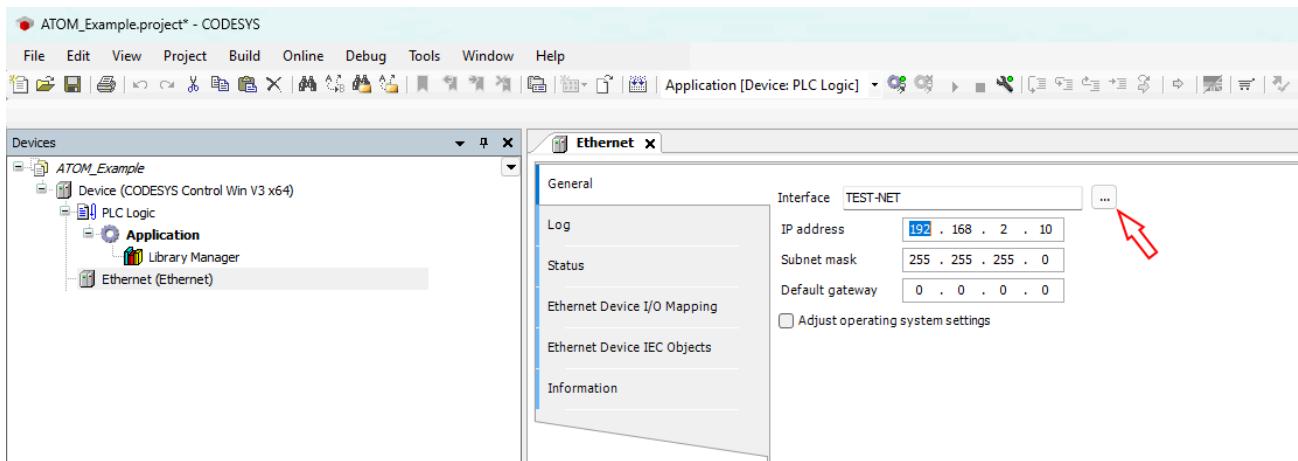
Next, expand **PROFINET IO > Ethernet Adapter** and select **Ethernet**, then click **Add Device**:



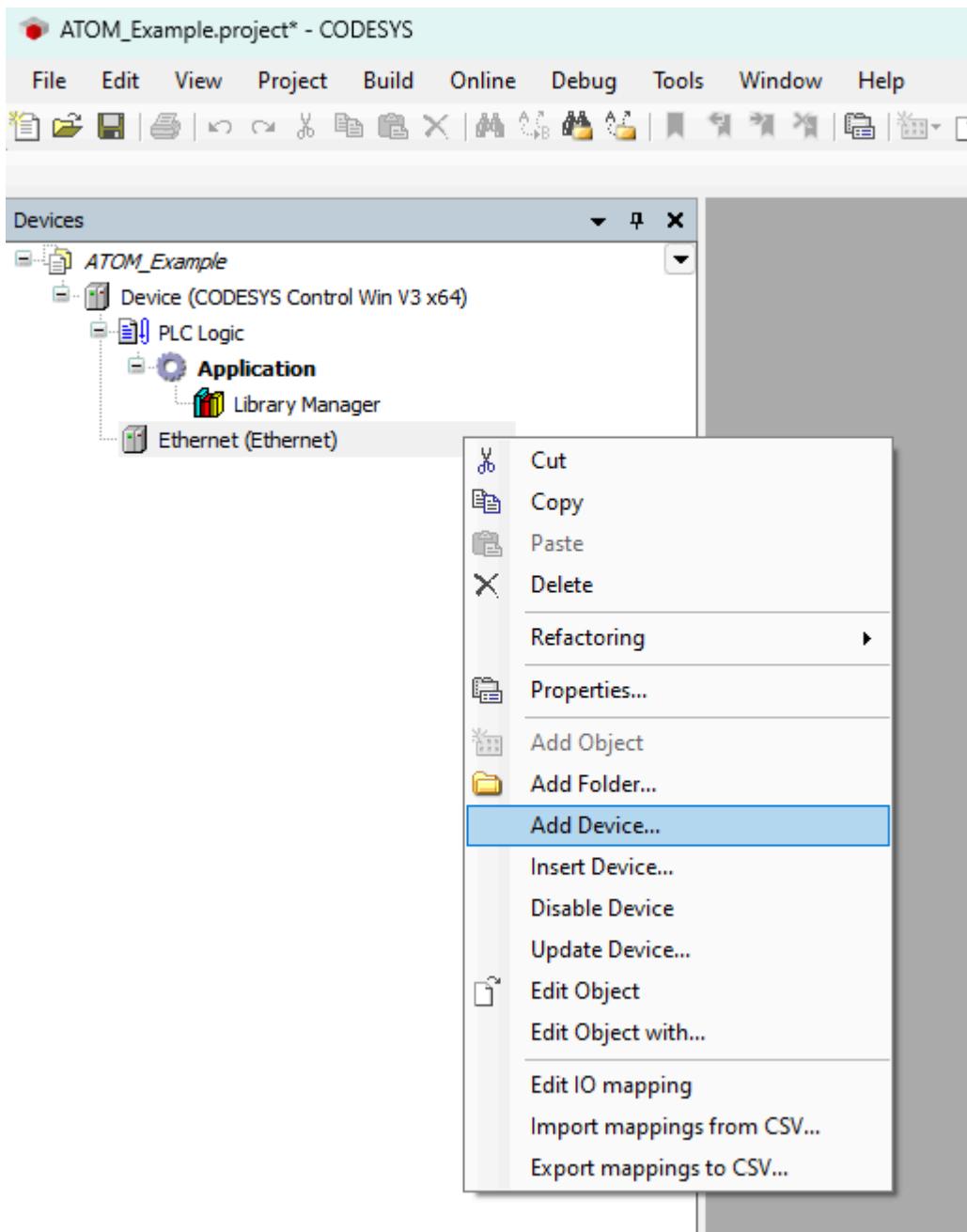
The newly added **Ethernet** device will now appear in the device tree. Double click **Ethernet (Ethernet)** to open its configuration tab. Within the **General** configuration tab, use the button indicated by the red arrow to select the network interface of the host machine that will be used to communicate with ATOM. In our case, we have a **TEST-NET** interface but this will be different for you.

ⓘ INFO

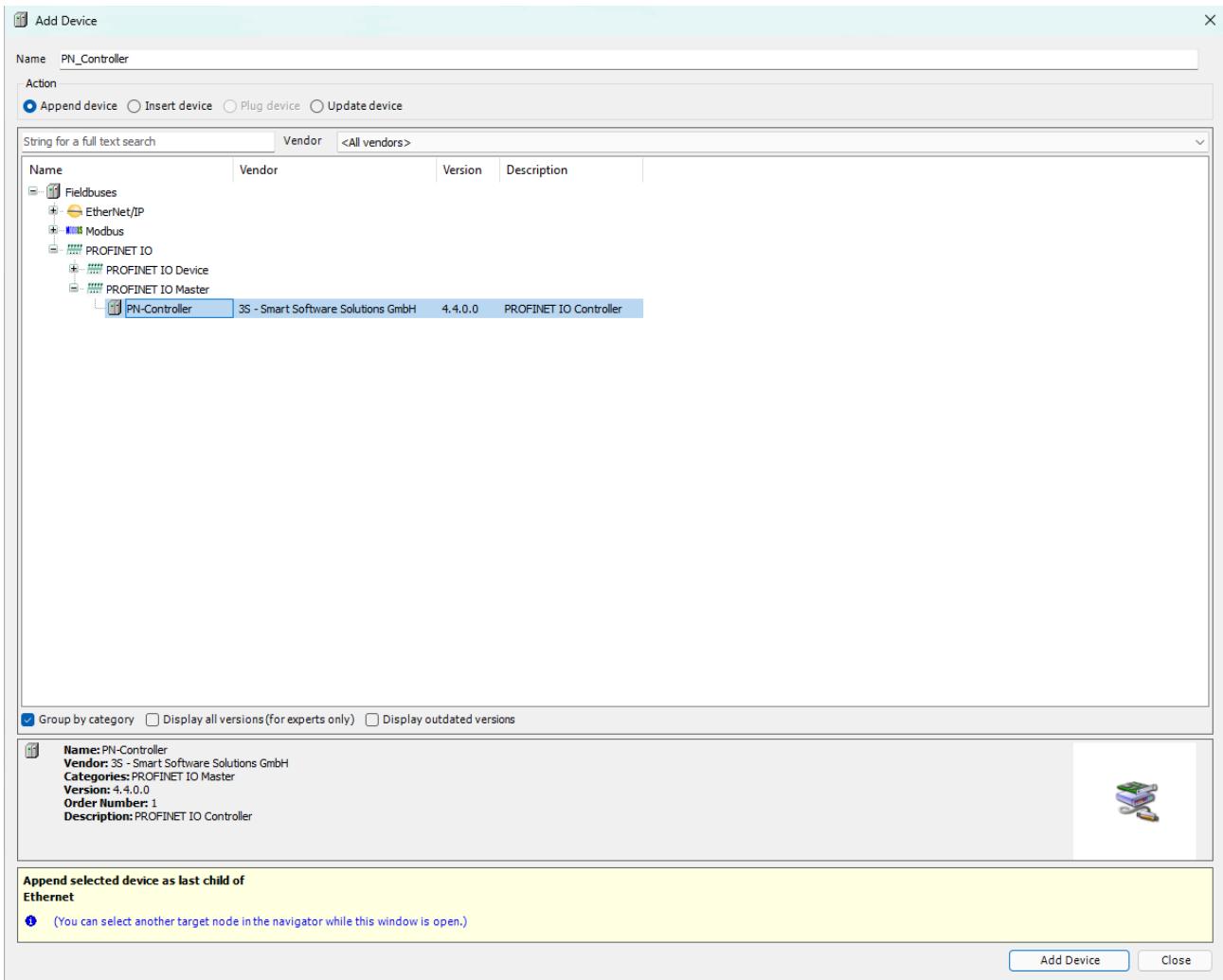
Note, you may get an error dialog displaying "Gateway not configured properly". If this is the case, make sure your SoftPLC is online by right-clicking the Codesys Win SysTray icon and starting the PLC. Navigate to the CODESYS Control Win V3 device in Codesys and use **Scan Network** to make sure the gateway is detected. Then, you can select the network interface.



Next, right click **Ethernet (Ethernet)** and select **Add Device**:



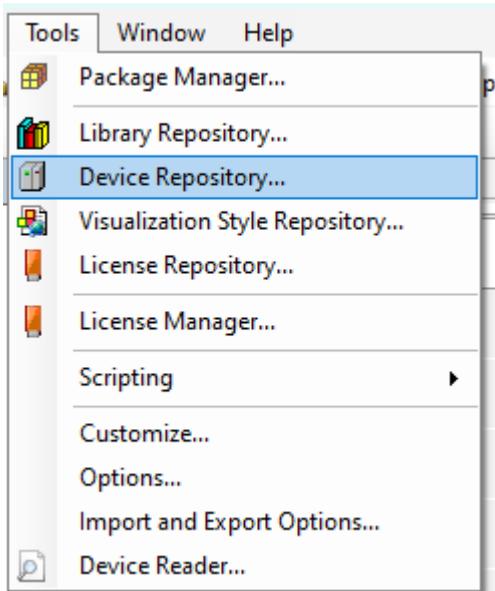
Expand **PROFINET IO > PROFINET IO Master**, select **PN-Controller** then click **Add Device:**



Your device tree should update to include the **PN-Controller** device.

Adding ATOM to the controller

First, we'll import ATOM's GSDML file you downloaded [earlier](#) into our Codesys device library. Open the tools menu and select **Device repository**:



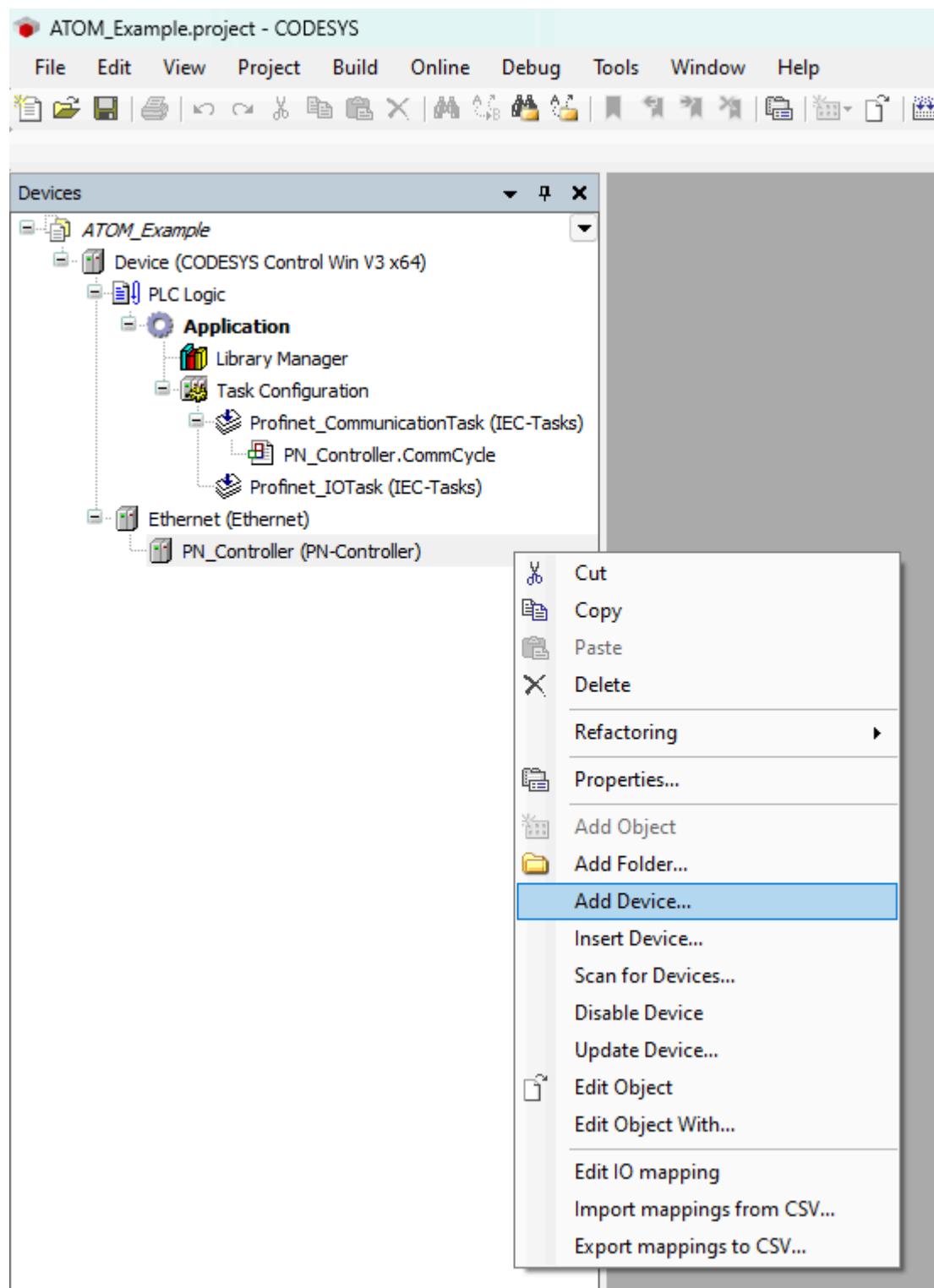
Next, click **Install** and select the `GSDML-V2.43-Control-Concepts-ATOM_20231108.xml` file. After you click install, **Atom** will appear under the **PROFINET IO > PROFINET IO Slave > I/O** category. Click **Close** to dismiss the dialog:

The screenshot shows the 'Device Repository' dialog. The 'Installed Device Descriptions' table lists the following entries:

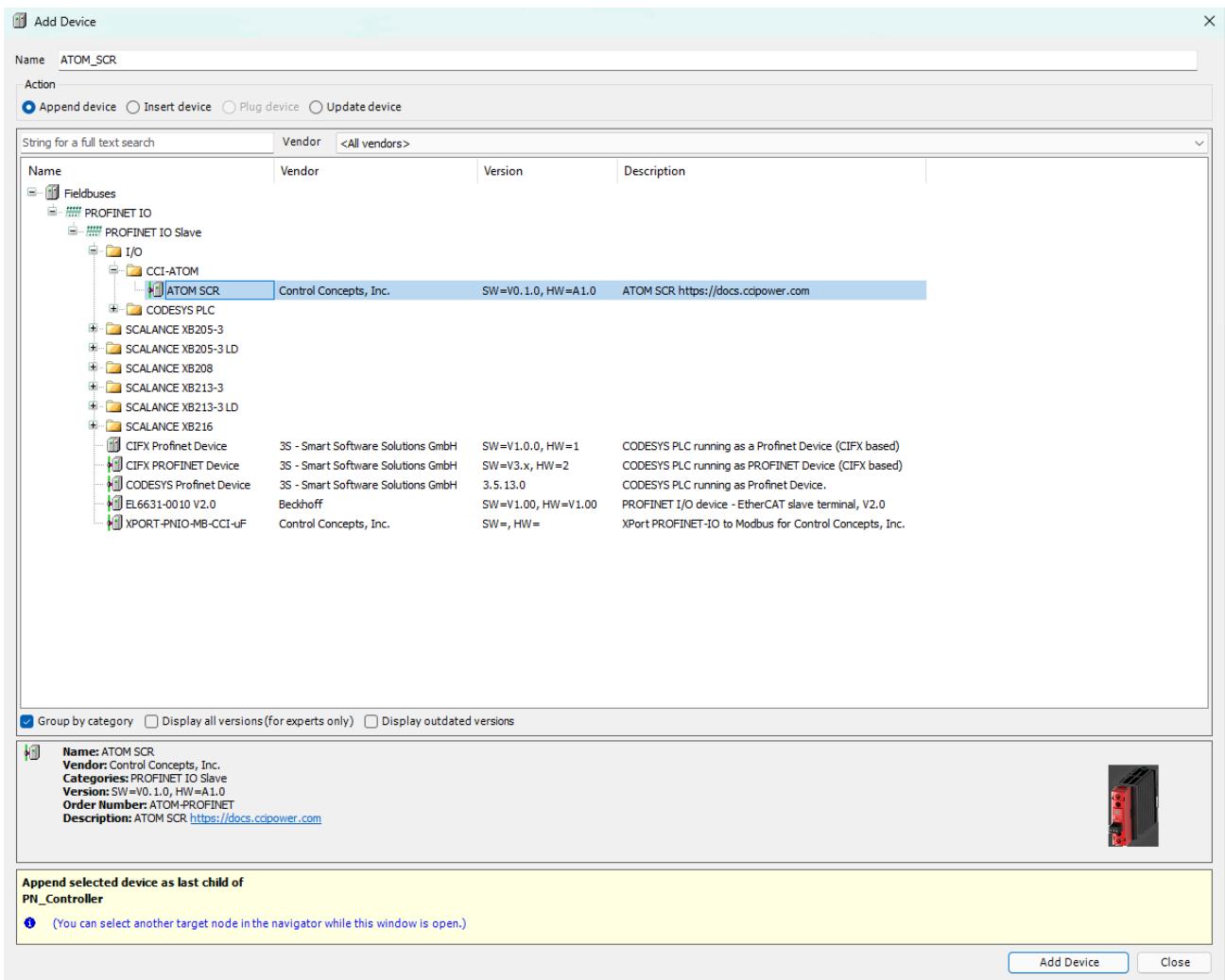
Name	Vendor	Version	Description
Modbus			
PROFIBUS			
PROFINET IO			
Ethernet Adapter			
PROFINET IO Device			
PROFINET IO Master			
PROFINET IO Slave			
I/O			
CCI-ATOM	Control Concepts, Inc.	SW=V0.1.0, HW=A1.0	ATOM SCR https://docs.ccipower.com
CODESYS PLC			
SCALANCE XB205-3			
SCALANCE XB205-3 LD			
SCALANCE XB208			
SCALANCE XB213-3			
SCALANCE XB213-3 LD			
SCALANCE XB216			
CIFX Profinet Device	3S - Smart Software Solutions GmbH	SW=V1.0.0, HW=1	CODESYS PLC running as a Profinet Device (CIFX based)
CIFX PROFINET Device	3S - Smart Software Solutions GmbH	SW=V3.x, HW=2	CODESYS PLC running as PROFINET Device (CIFX based)
CODESYS Profinet Device	3S - Smart Software Solutions GmbH	3.5.13.0	CODESYS PLC running as Profinet Device.
ELE631-0010 V2.0	Beckhoff	SW=V1.00, HW=V1.00	PROFINET I/O device - EtherCAT slave terminal, V2.0
YDNET-DANTO-MP-FCU-E	Control Concepts, Inc.	SW= - HW= -	YDNET PROFINET I/O to Modbus for Control Concepts, Inc.

On the right side of the dialog, there are buttons for **Install...**, **Uninstall**, **Export..**, **Renew Device Repository**, **Details...**, and **Close**.

Now, we'll add ATOM to the PN-Controller. Right click **EtherNet/IP Scanner (EtherNet/IP Scanner)** and select **Add Device**:



Expand **PROFINET IO > PROFINET IO Slave > I/O > CCI-ATOM** and select **ATOM SCR**, then click **Add Device**:



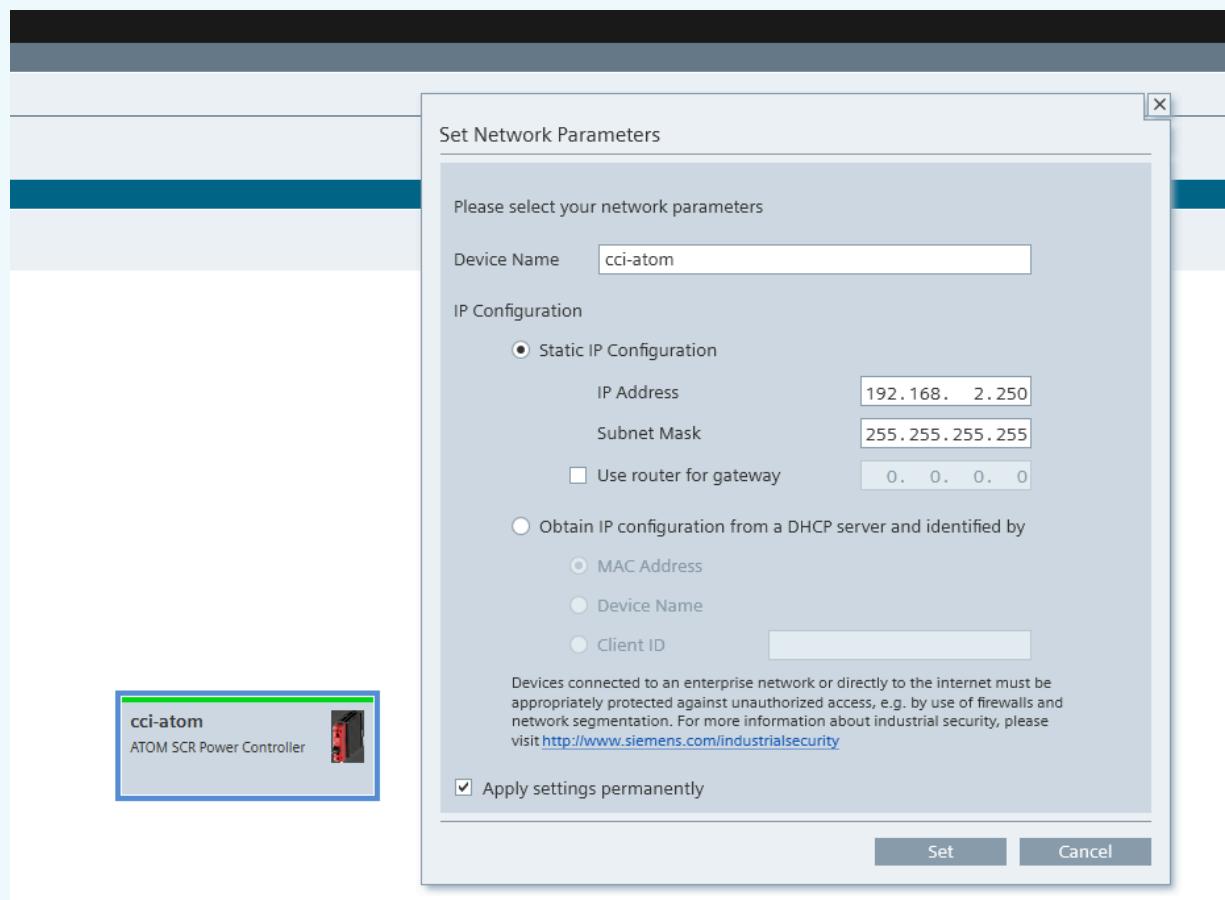
Finally, double click **Atom (Atom)** to open its configuration tab. In the **General** tab, set the **Station name**, **IP Address**, and **Subnet mask** for your ATOM SCR:

ⓘ INFO

You can find or change these parameters in Control Panel, or using a tool like [Proneta](#). Make sure your station name and IP settings on Atom are properly set to the same values you enter here so that Codesys can connect to ATOM.

Proneta

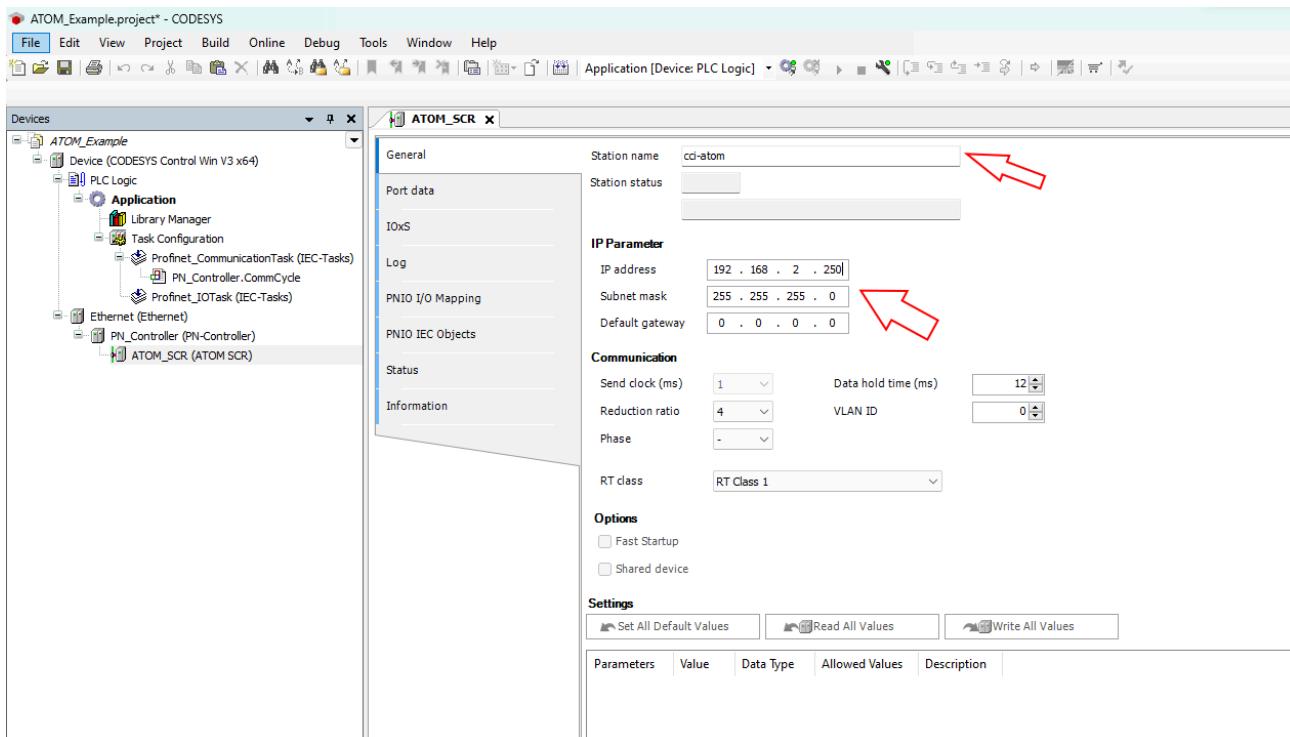
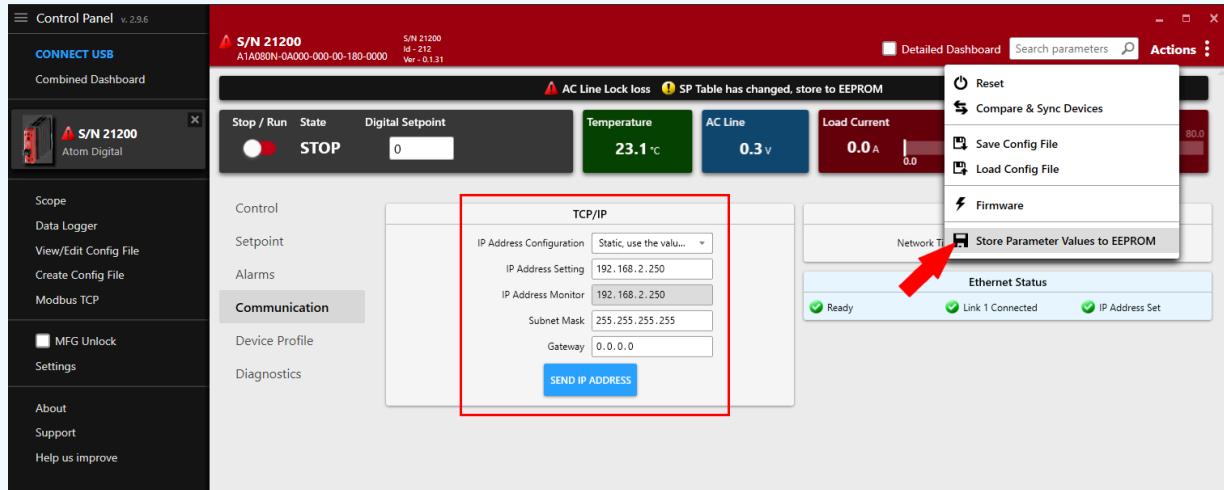
If you're using Proneta, make sure to change the IP settings with **Store permanently** checked.



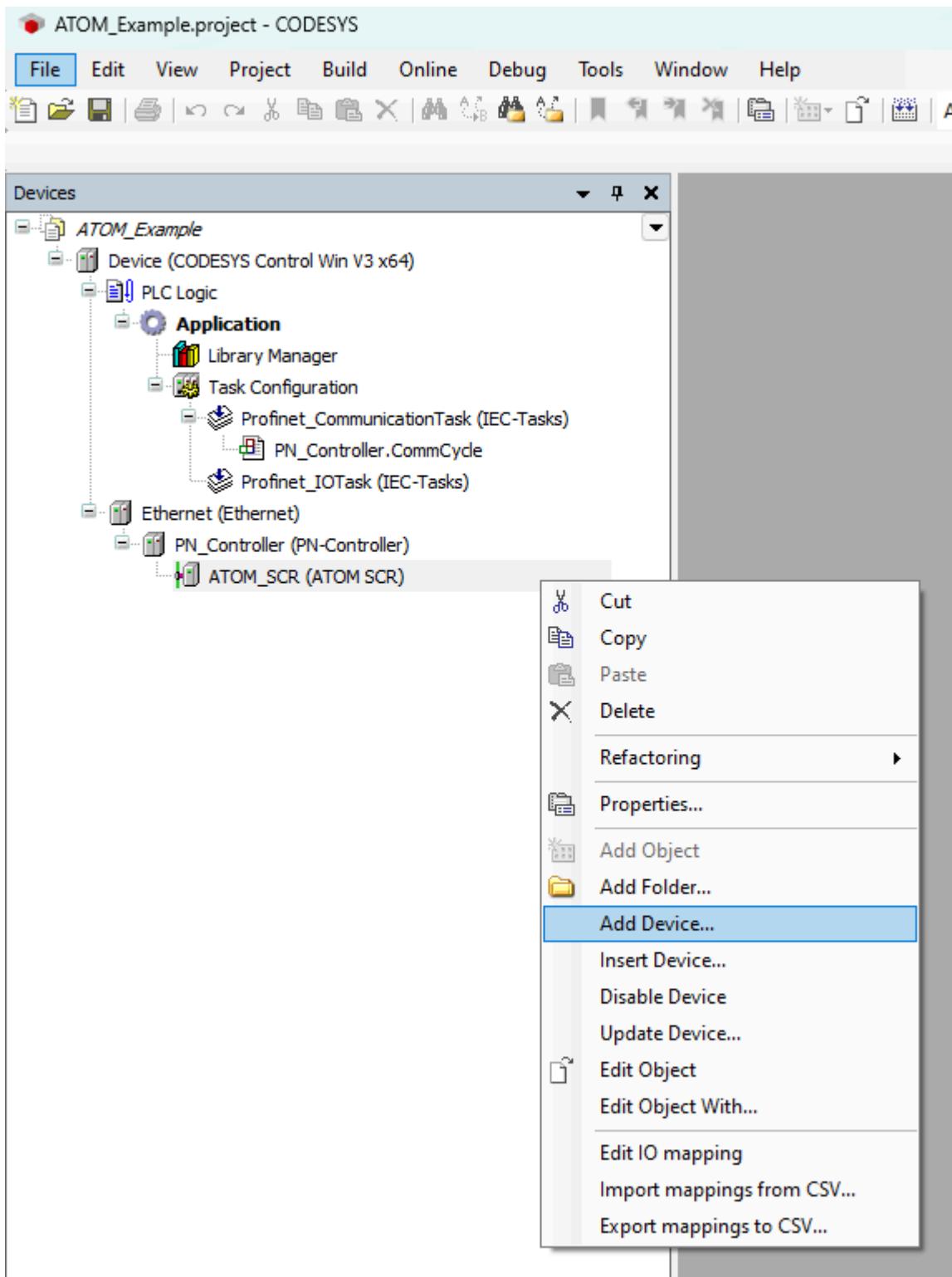
Control Panel

Connect your Atom unit to your PC using a USB cable. Open Control Panel and update your Atom's communication parameters. When you're finished, click **Send IP**

Address, then go to **Actions** in the upper right and select **Store Parameter Values to EEPROM**:

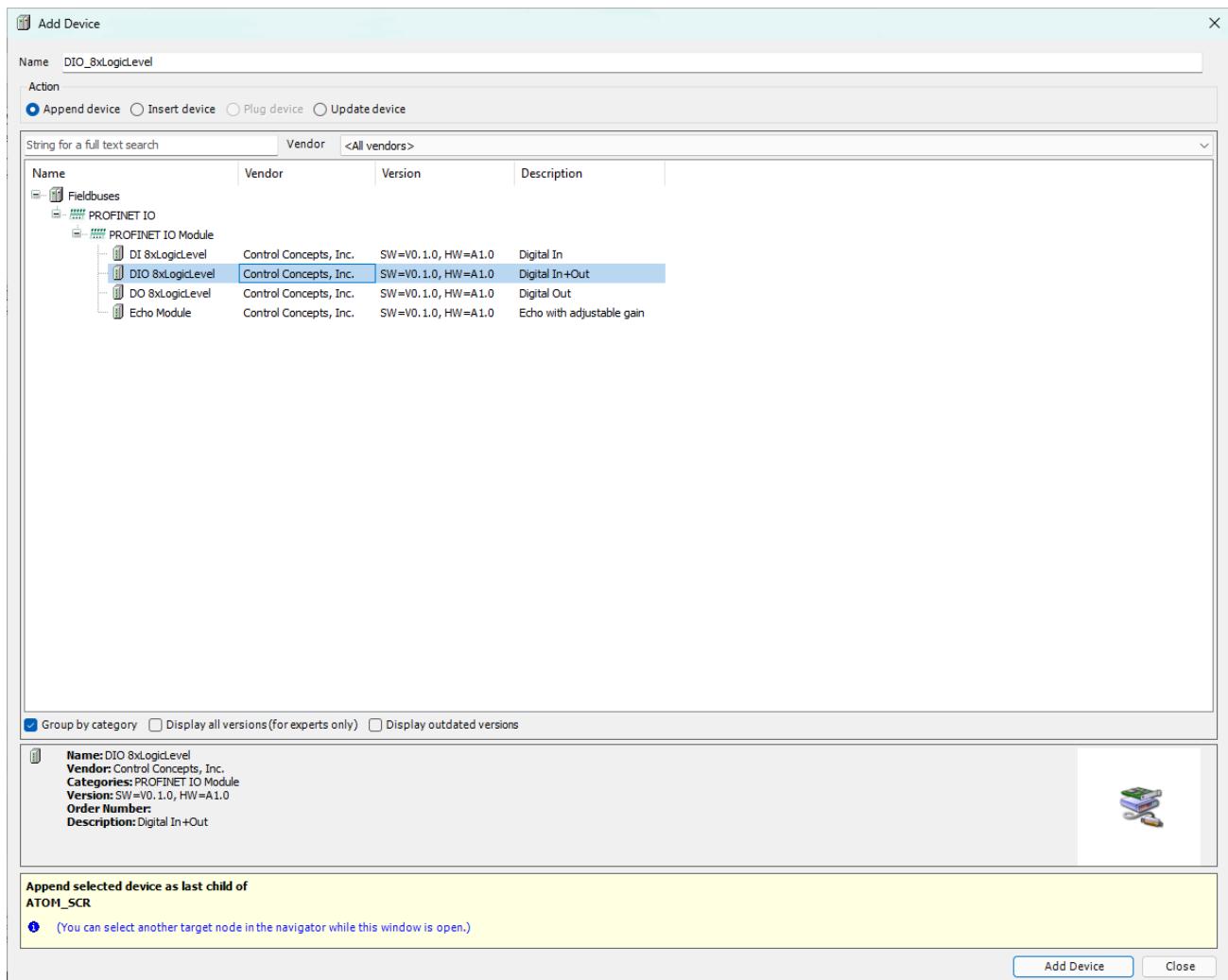


Next, right click **ATOM_SCR (ATOM SCR)** and select **Add Device**:



Here, you can choose which Profinet I/O modules to enable for your Atom. Select **DIO 8xLogicLevel** which allows both input and output of data to/from Atom. You can add other

I/O modules if needed. Then, click **Add Device**:



Create a program

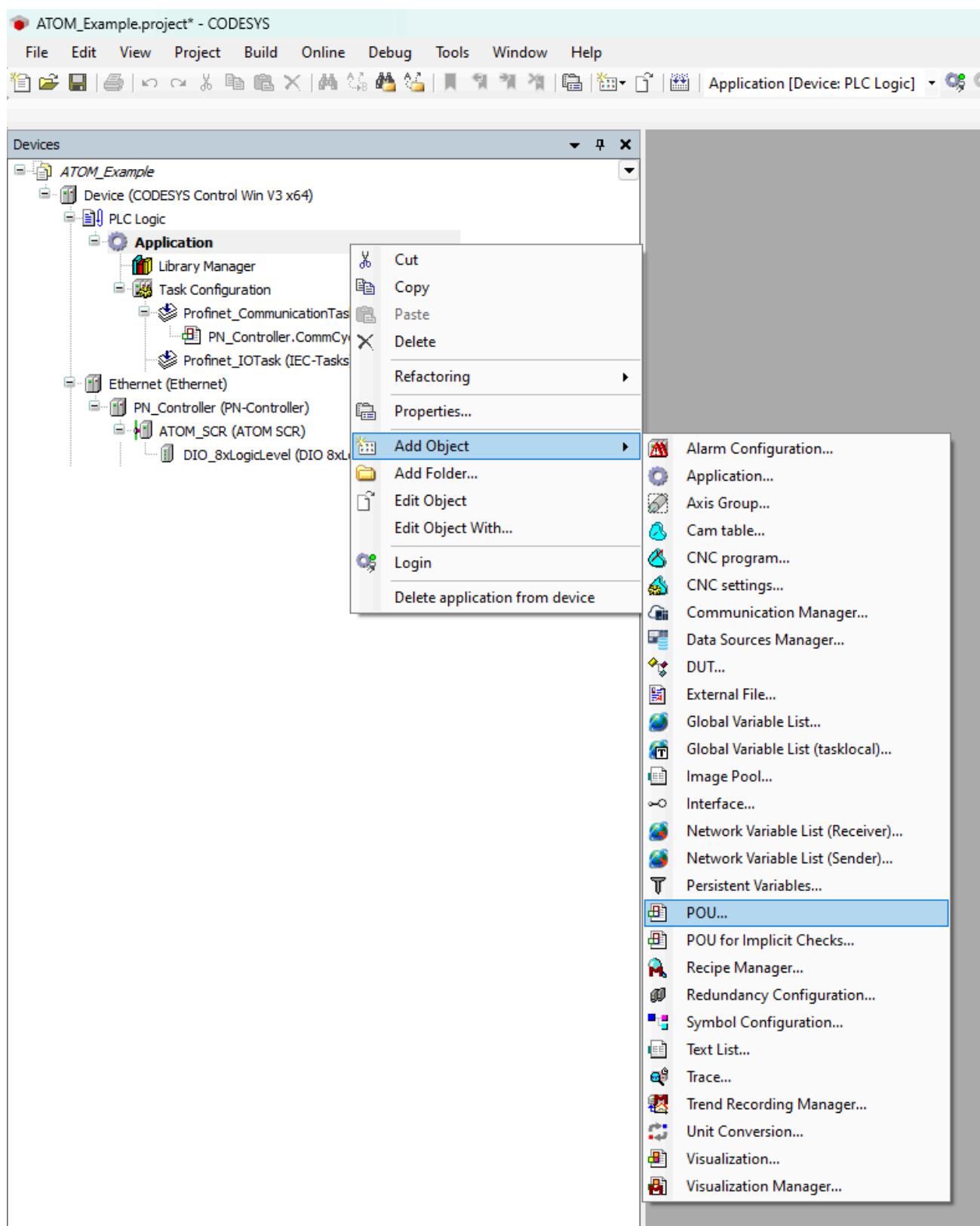
Next, we'll create a PLC program. We provide examples for both ladder logic and structured text:

- Program with ladder logic
- Program with structured text

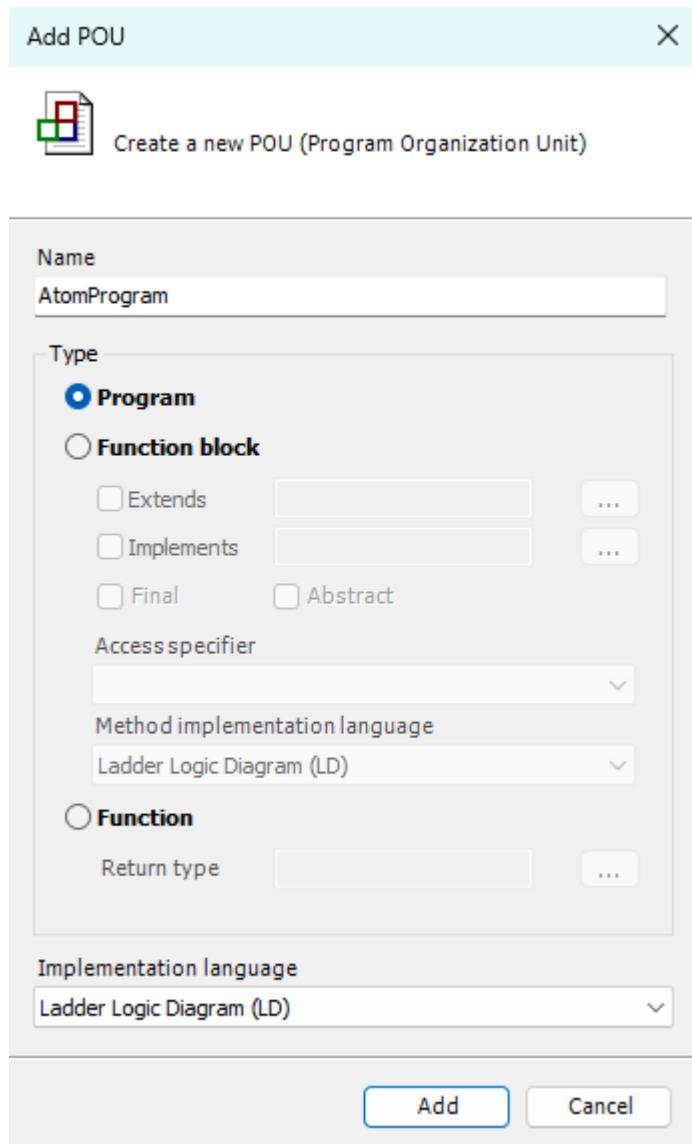
Example: Ladder logic

Creating the program

Right click **Application** and select **Add Object > POU:**



Set the name to **AtomProgram** and select **Ladder Diagram (LD)** as the Implementation language:



Copy the following code into the top panel of the **AtomProgram** editor:

```
PROGRAM AtomProgram
```

```
VAR
```

```
RUN_SWITCH: BOOL;
```

```
SETPOINT: DINT;
```

```
TEMP: REAL;
```

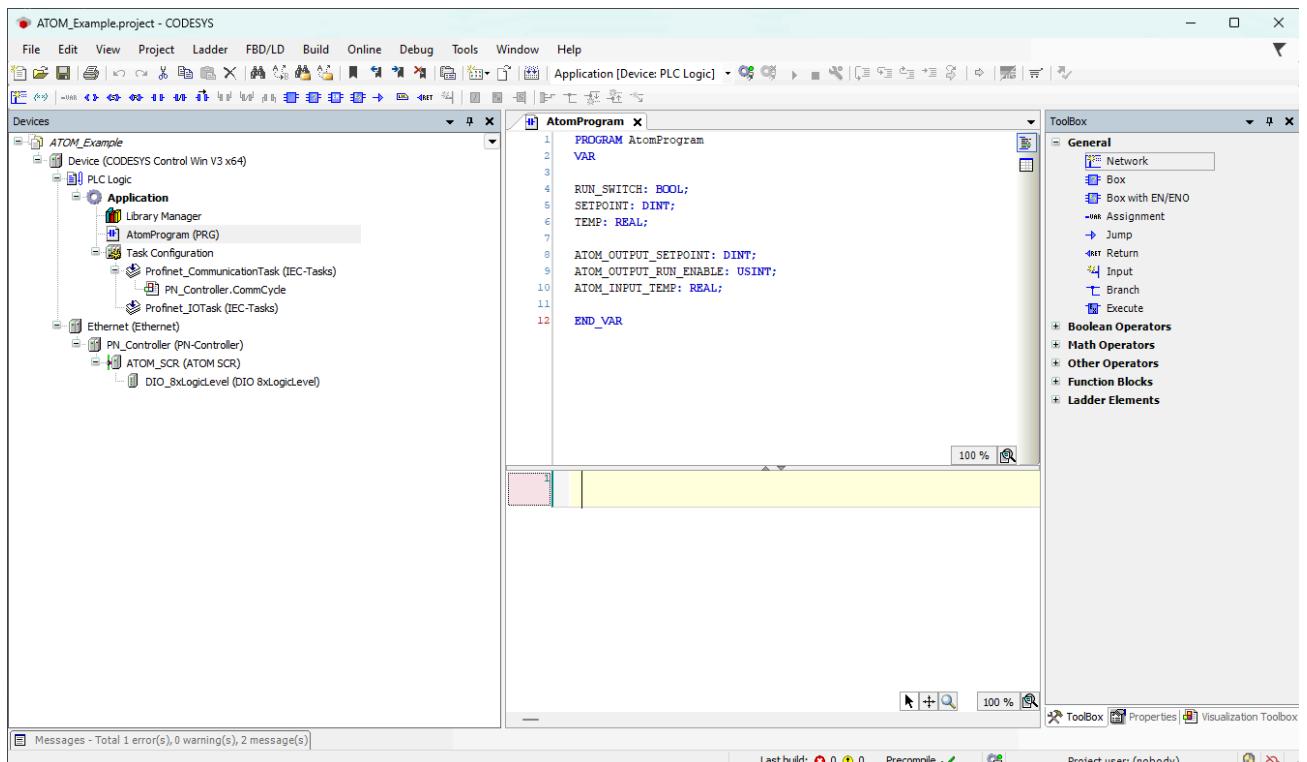
```
ATOM_OUTPUT_SETPOINT: DINT;
```

```
ATOM_OUTPUT_RUN_ENABLE: USINT;
```

```
ATOM_INPUT_TEMP: REAL;
```

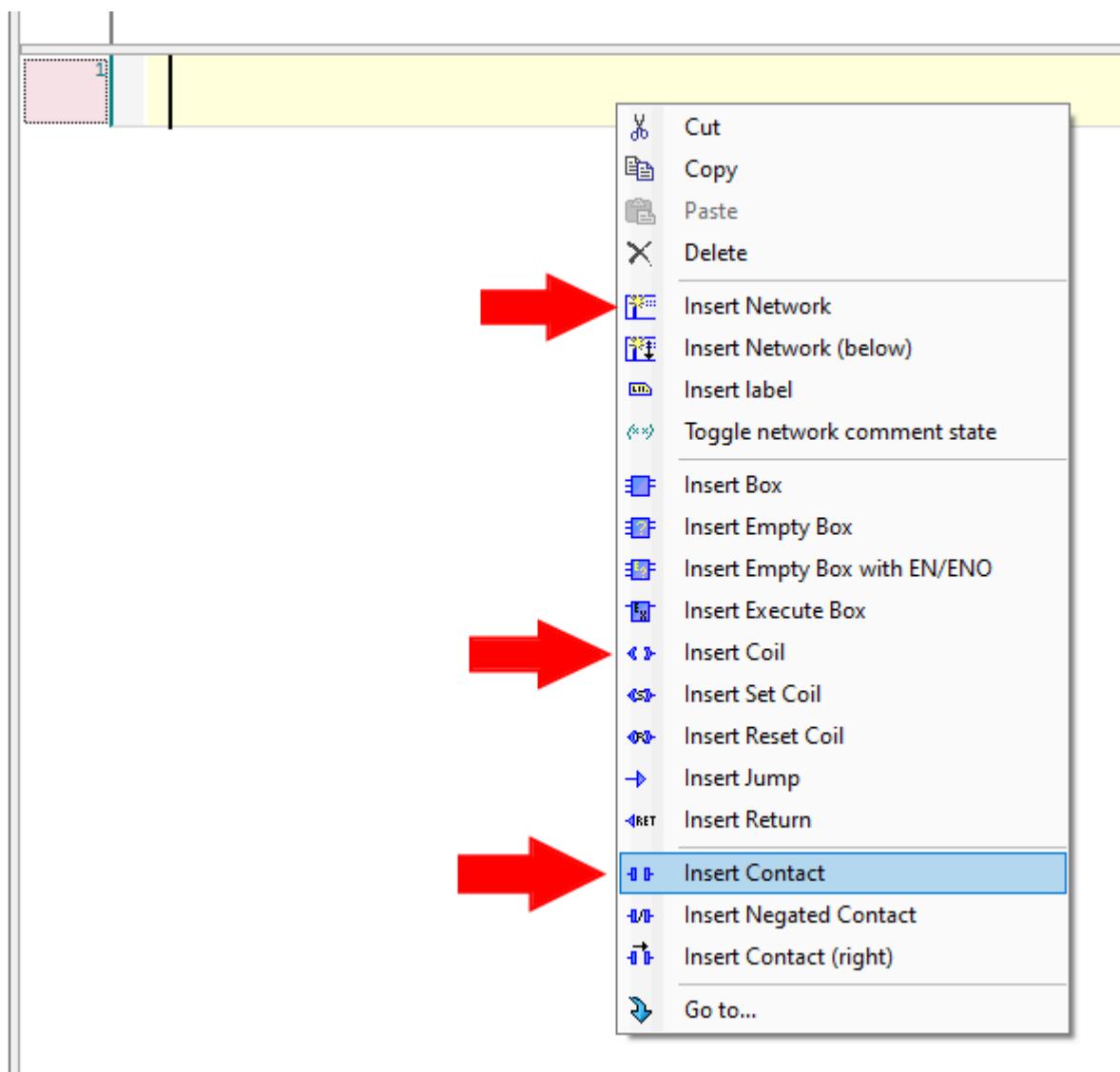
```
END_VAR
```

After you've copied the code over, the editor for **AtomProgram** should look like this:

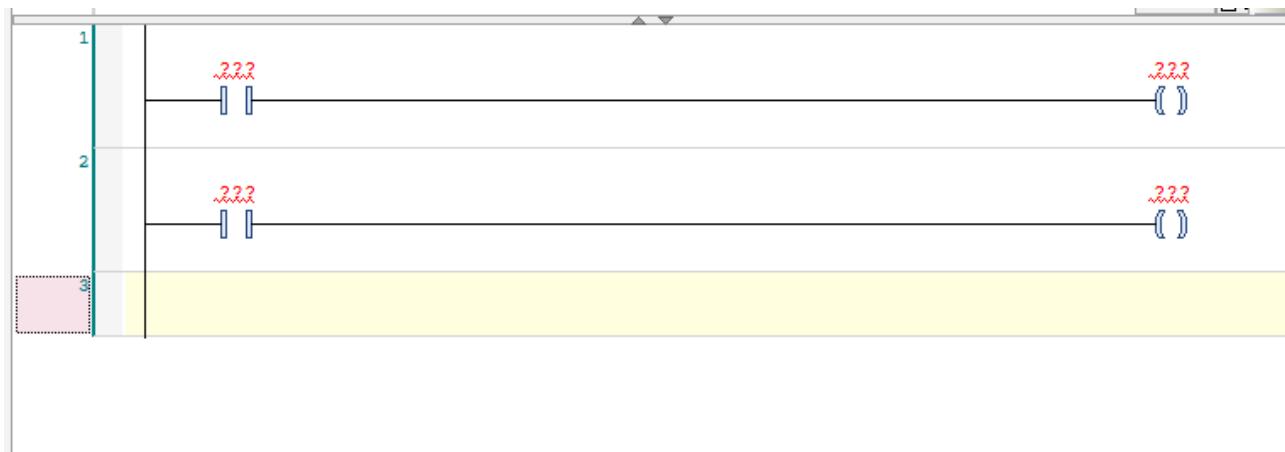


In the bottom panel of the editor, we'll create a simple ladder logic program using the variables we just added above.

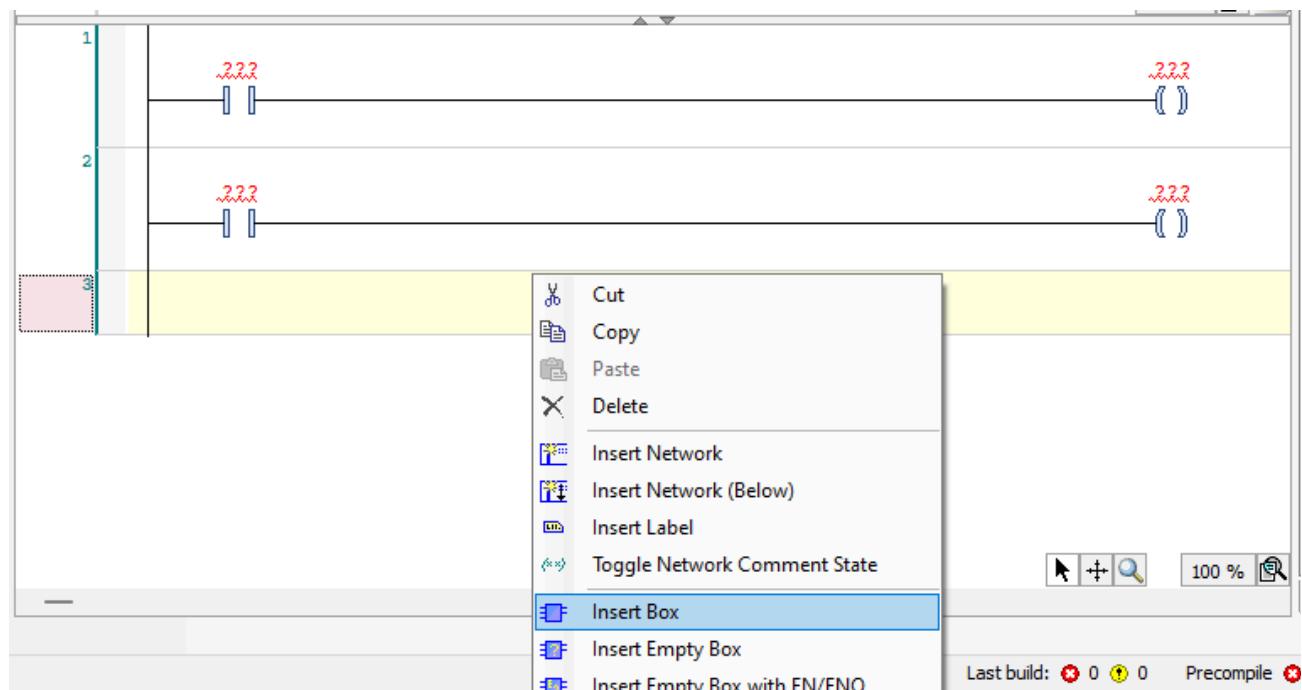
1. Create **3** networks total by right-clicking and selecting **Insert Network** three times.
2. For the first two rungs (networks), insert a contact and a coil.



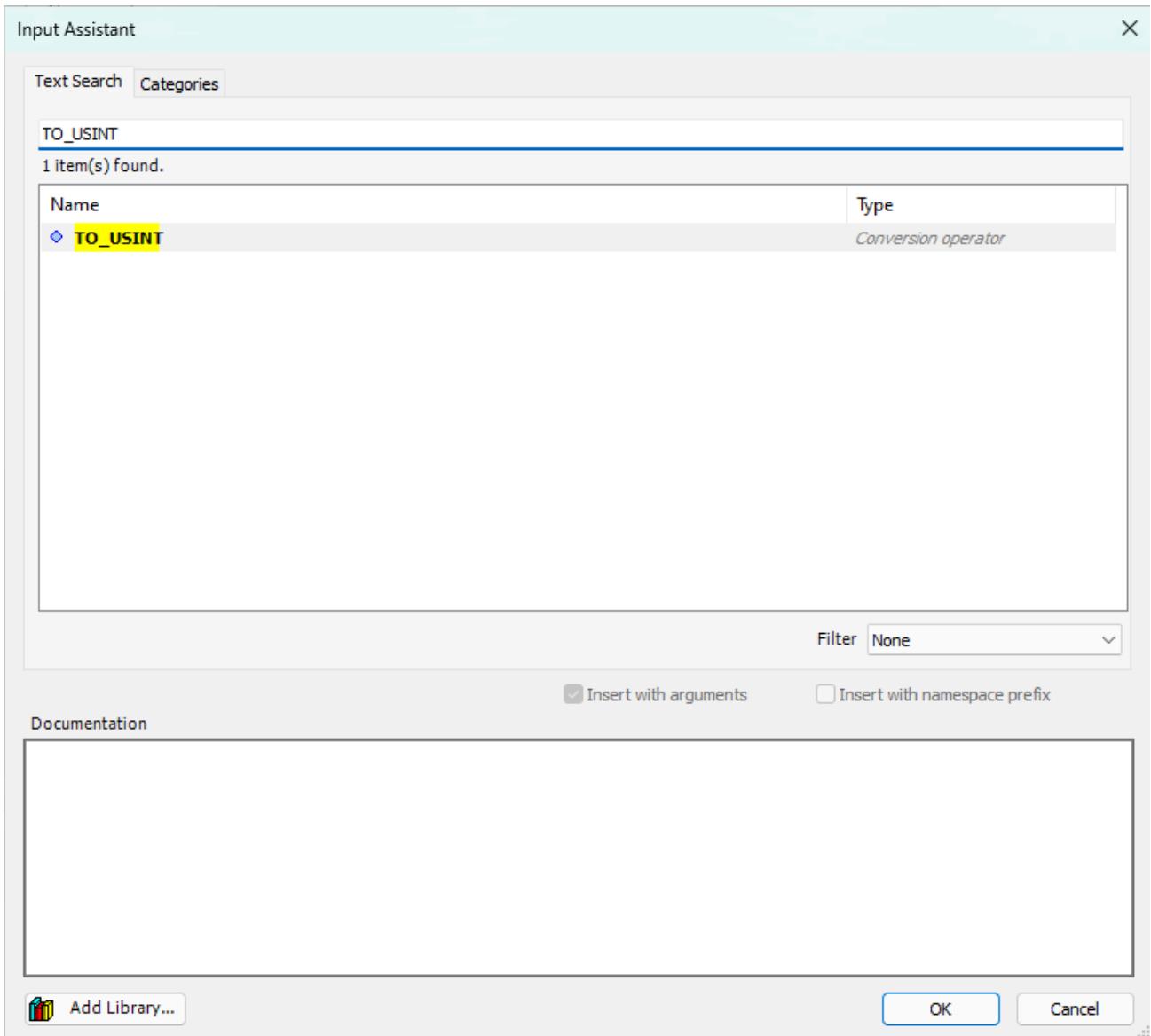
After you're finished, your ladder logic program should look like:



On the third rung, right click and select **Insert Box**:



Add a **TO_USINT** box:



For the first two rungs, replace the `???` with the corresponding variables:

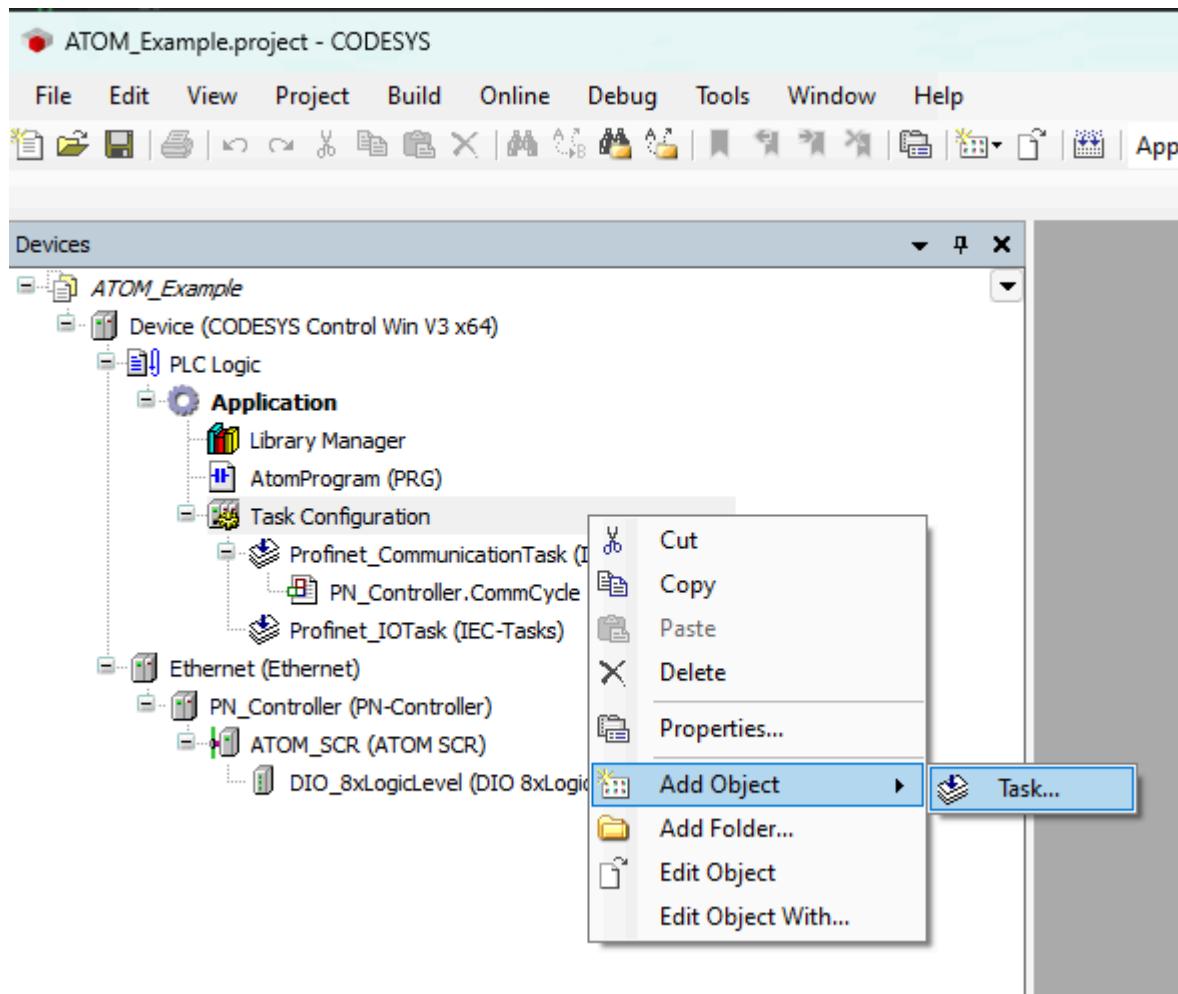
1. **Rung #1** - `ATOM_INPUT_TEMP` and `TEMP`
2. **Rung #2** - `SETPOINT` and `ATOM_OUTPUT_SETPOINT`

On the third rung, set the input to `EN` to `TRUE` and set the input parameter to `RUN_SWITCH` and output parameter to `ATOM_OUTPUT_RUN_ENABLE`. After you're finished, your ladder logic program should look like:

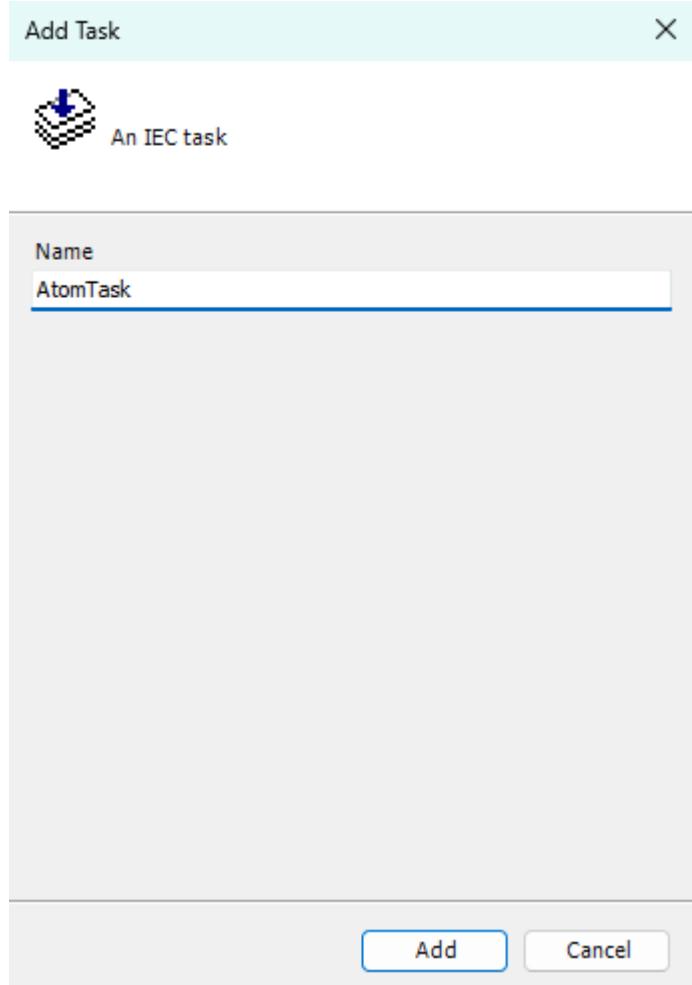


Finally, we'll add a task to call **AtomProgram** from the PLC's control loop:

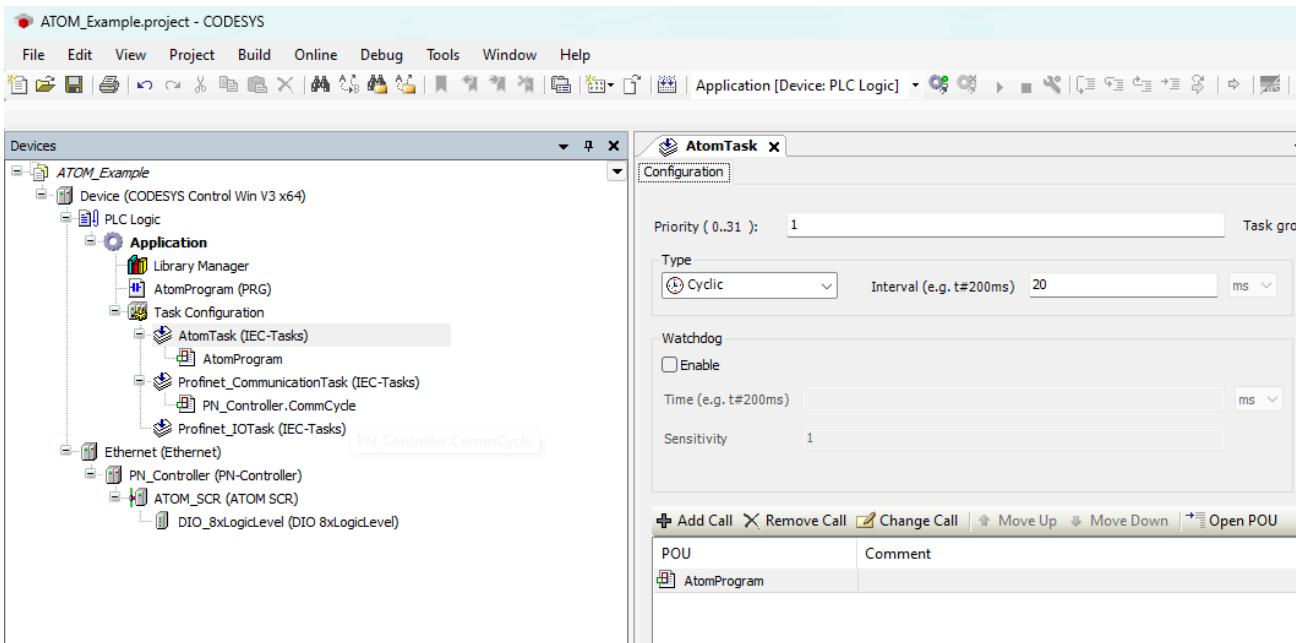
Right click **Task Configuration** and select **Add Object > Task**:



Name your task **AtomTask** and click **OK**:



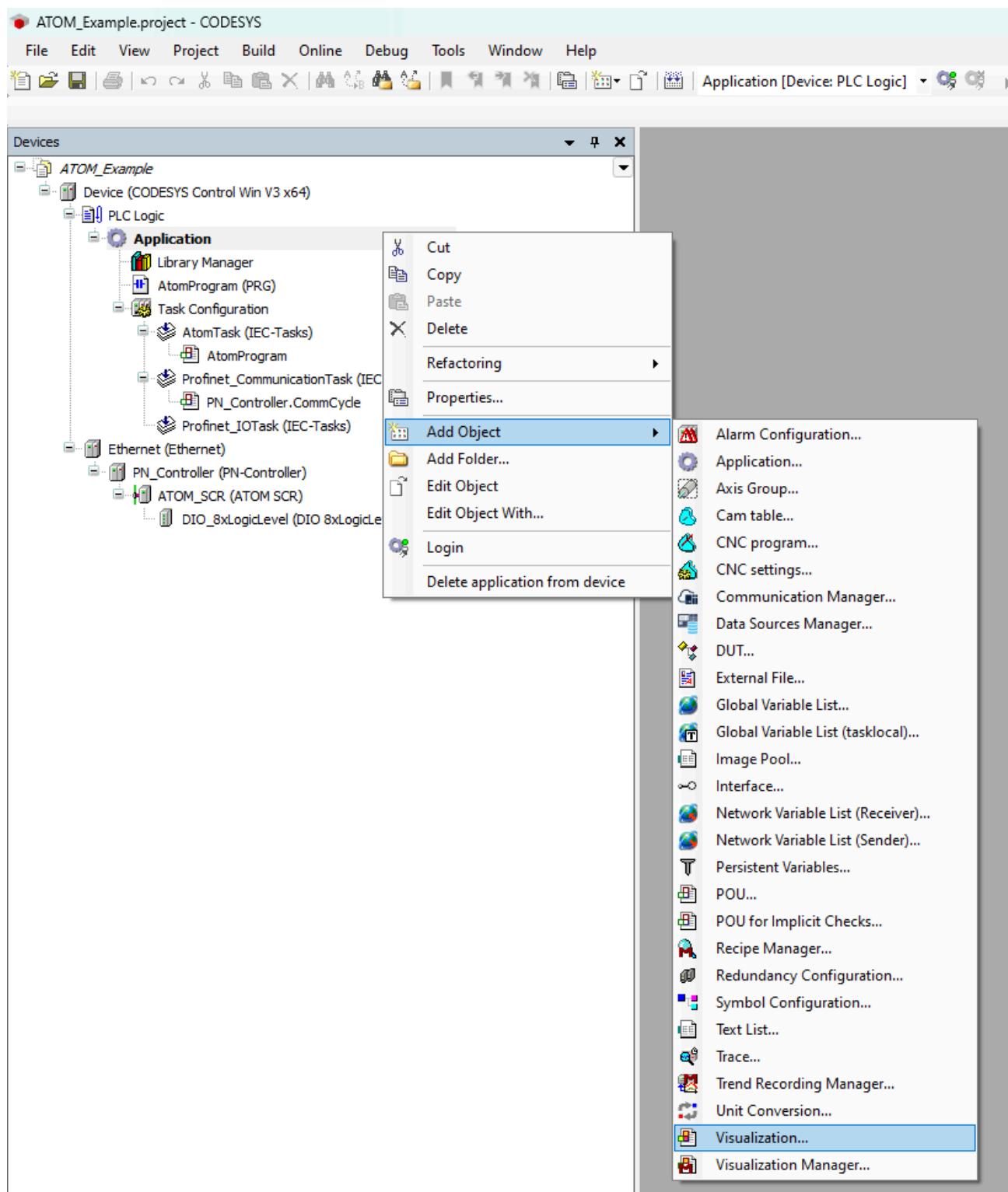
Next, double click **AtomTask (IEC-Tasks)** to open its configuration tab. Click **Add Call** and select **Application > AtomProgram**. After doing so, AtomTask's configuration should look like:



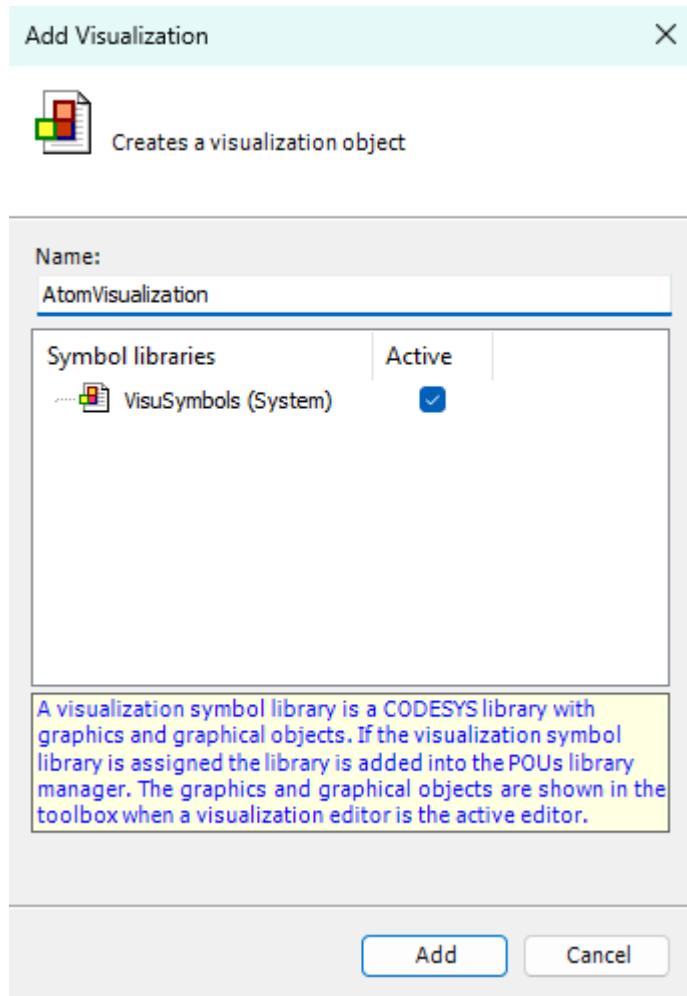
Setting up visualization

Next, we'll set up a simple visualization display to control and monitor ATOM.

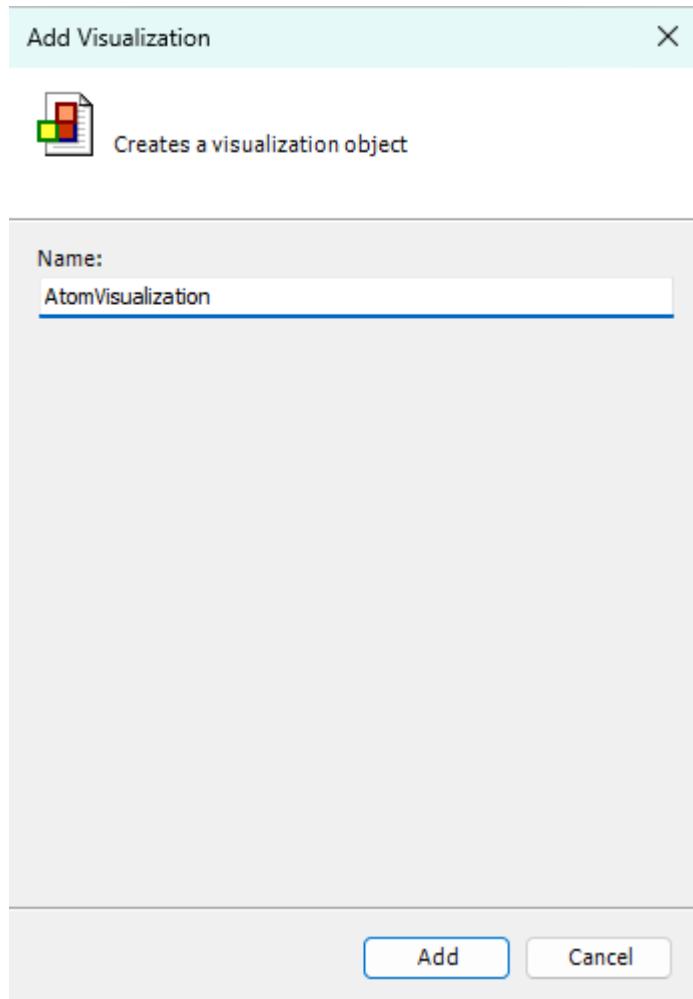
Right click **Application** and select **Add Object > Visualization**:



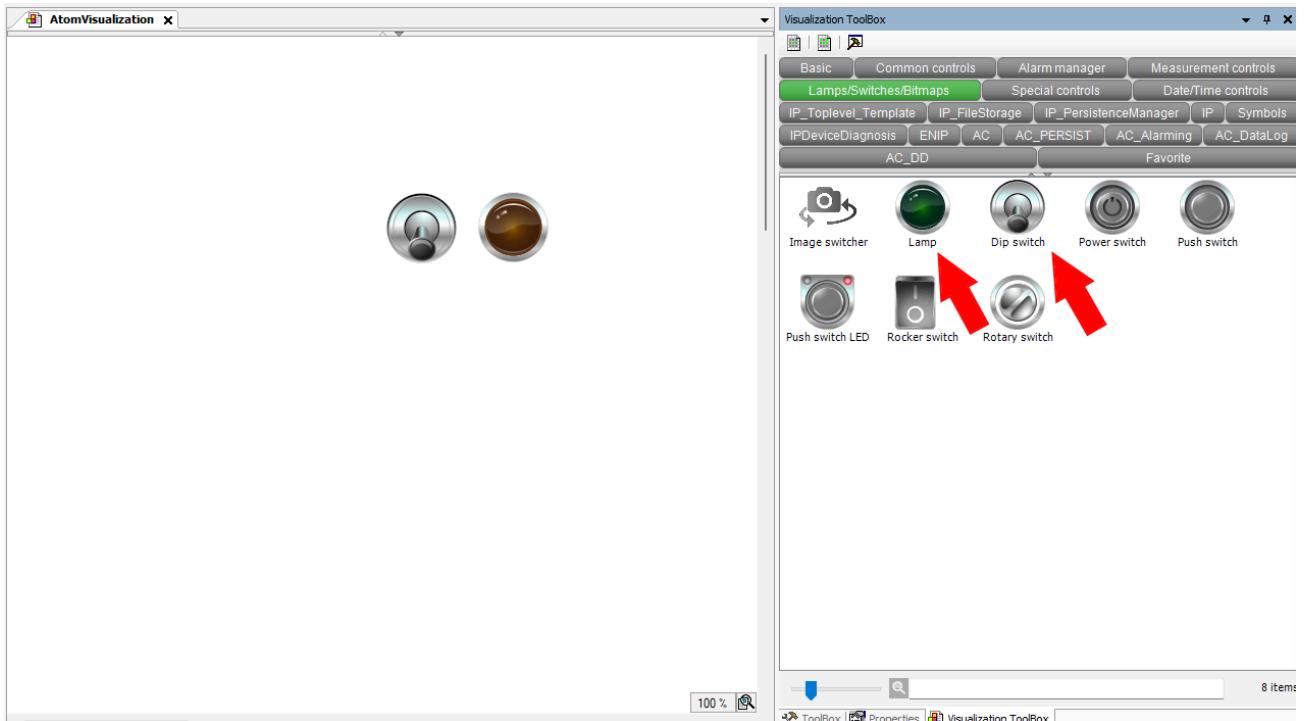
Make sure to check **Active** for **VisuSymbols (System)**, then click **Add**:



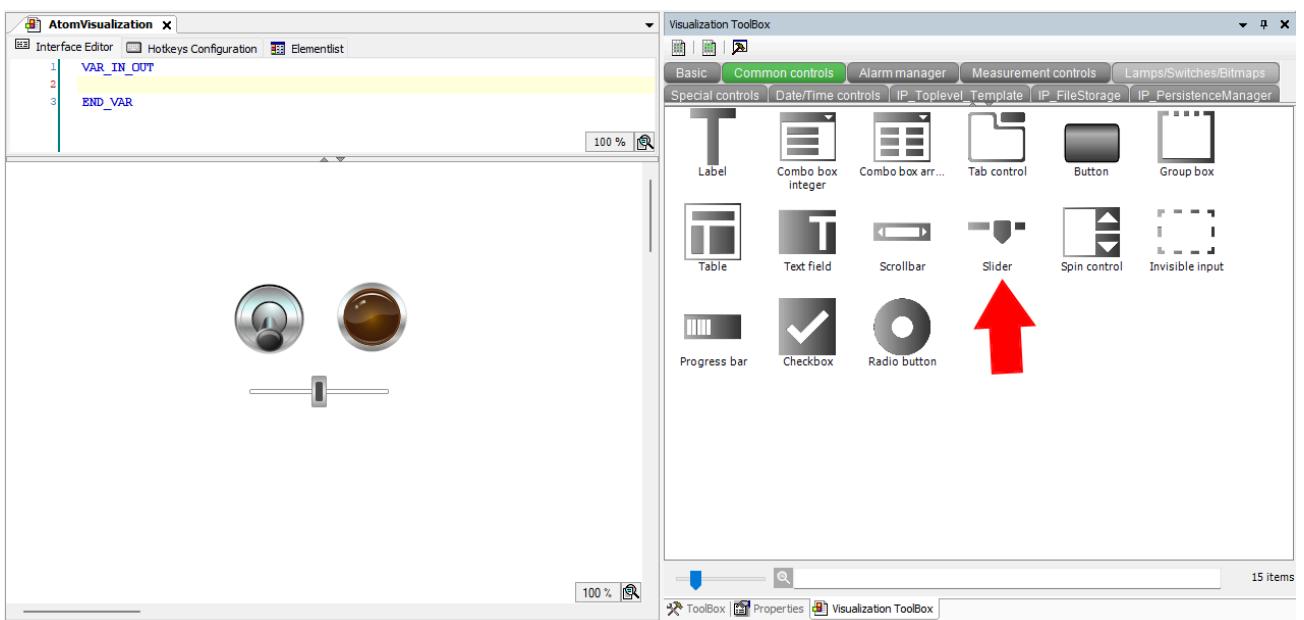
Name your visualization **AtomVisualization** and click **Add**:



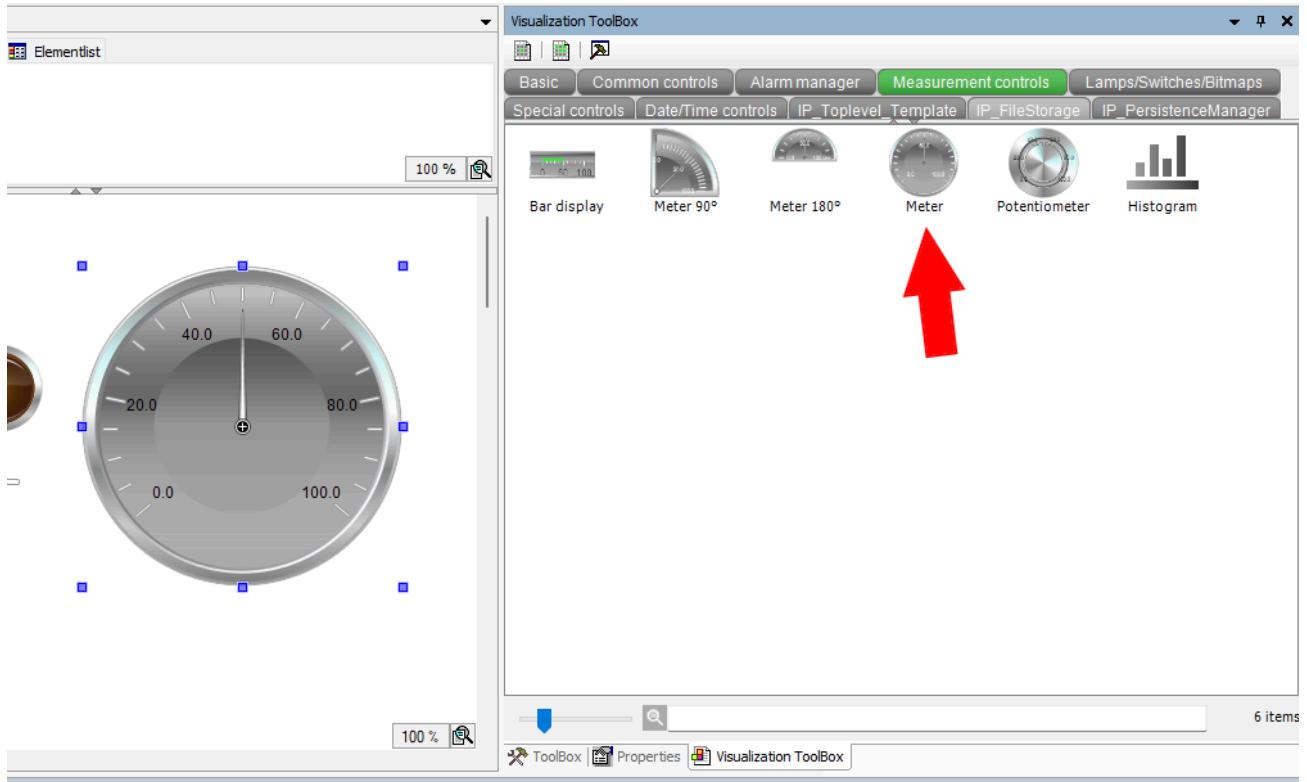
Double click **AtomVisualization** to open its configuration editor. From the **Visualization ToolBox** panel on the right, select the **Lamps/Switches/Bitmaps** category and add a lamp and a dip switch:



Next, in the **Common controls** category, add a slider:

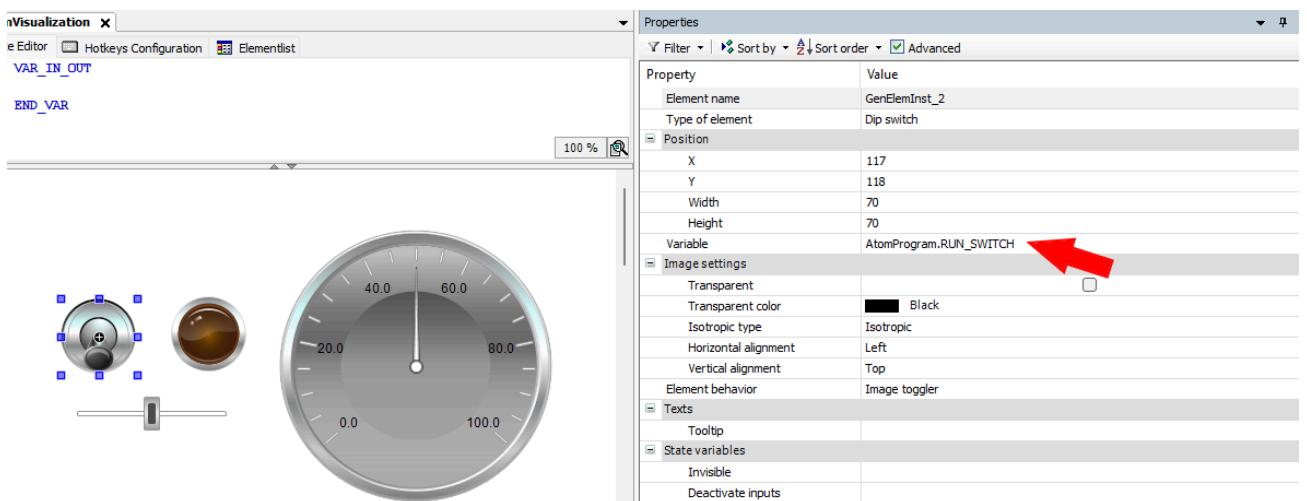


Finally, in the **Measurement controls** category, add a meter:

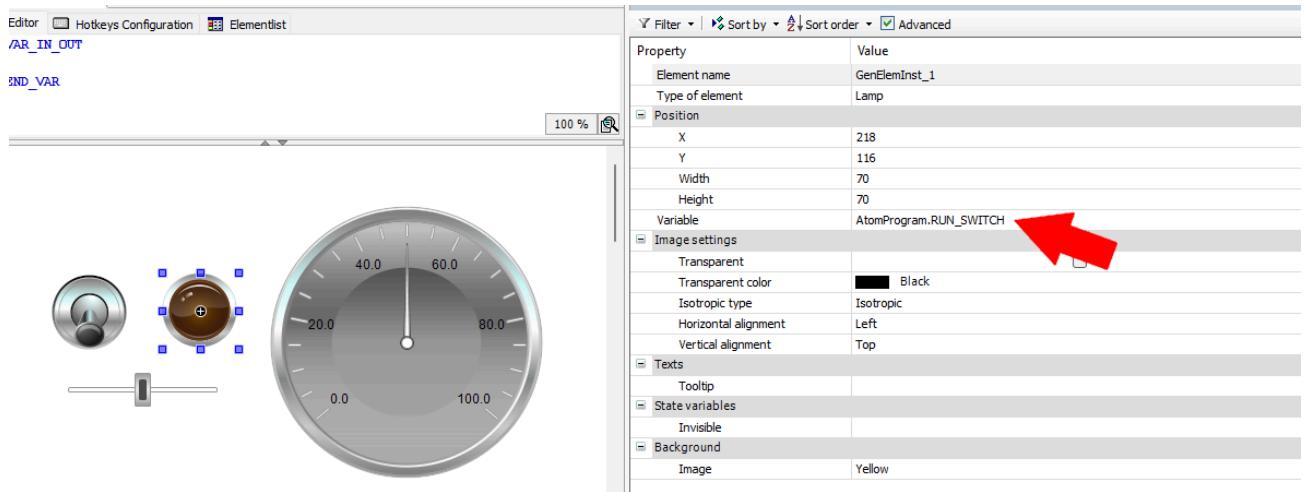


Wiring up the controls

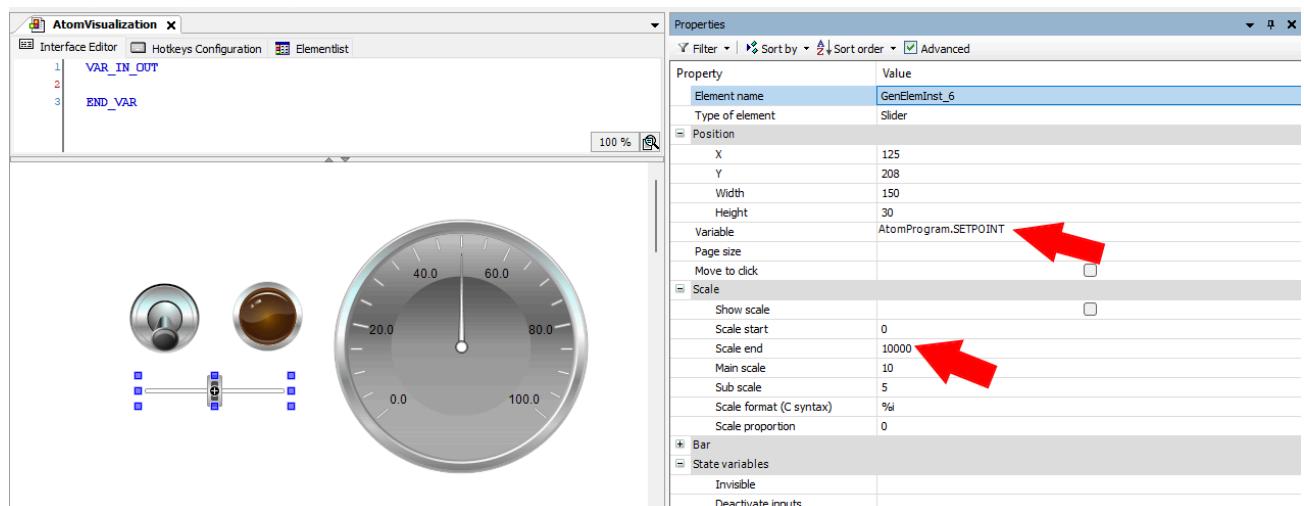
Next, we'll connect the controls to our PLC program. Select the dip switch and set the **Variable** field to `AtomProgram.RUN_SWITCH` as indicated by the red arrow:



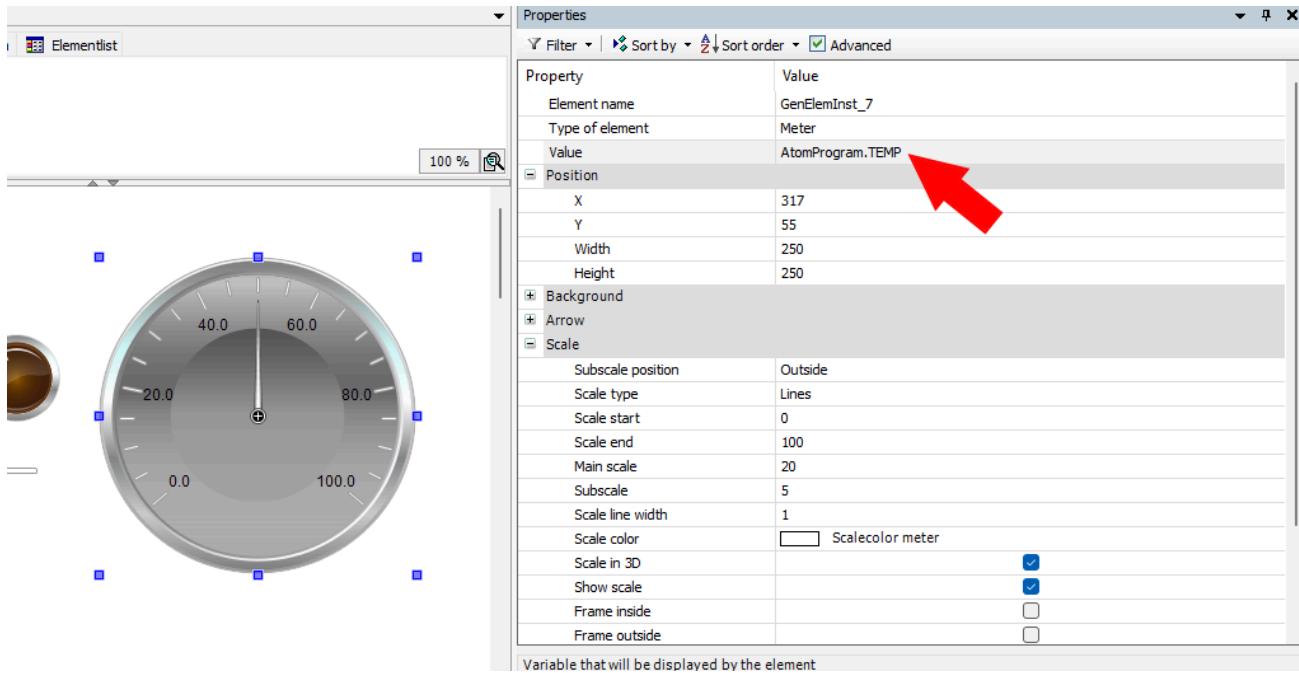
Select the lamp and set the **Variable** field to `AtomProgram.RUN_SWITCH` as indicated by the red arrow:



Select the slider and set the **Variable** field to `AtomProgram.SETPOINT` and set **Scale end** to `10000`:

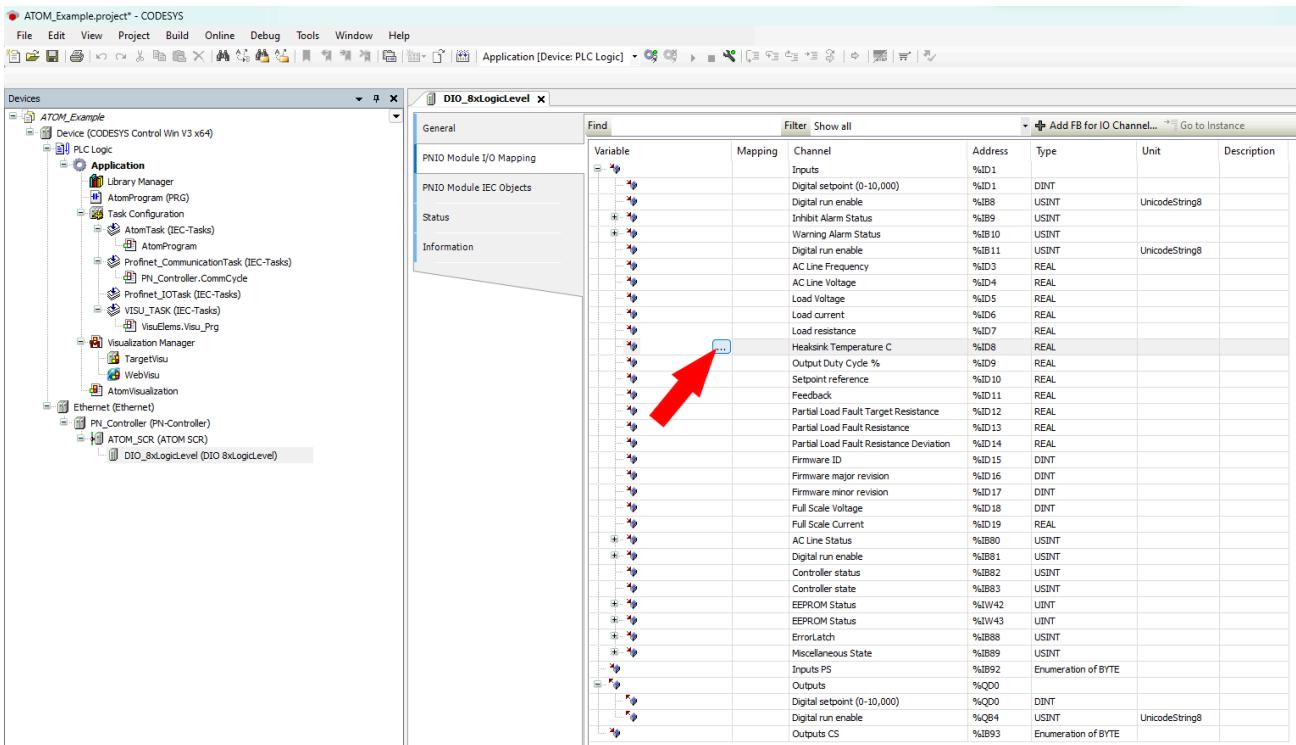


Select the meter and set the **Variable** field to `AtomProgram.TEMP`:



Mapping variables

Finally, we'll map our PLC variables to ATOM. Double click **DIO_8xLogicLevel (DIO 8xLogicLevel)** in the device tree to open its configuration window. Select the **PNIO Module I/O Mapping** tab:



Above, select the button indicated by the red arrow. This will open the **Input Assistant** dialog. Select **Application > AtomProgram > ATOM_INPUT_TEMP** and click **Add**:

Input Assistant

Text Search Categories

Variables

Name	Type	Address	Origin
{ AC	Library		AC_ModuleBase, 4....
{ Application	Application		
AtomProgram	PROGRAM		
ATOM_INPUT_TEMP	REAL		
ATOM_OUTPUT_RUN_ENABLE	USINT		
ATOM_OUTPUT_SETPOINT	DINT		
RUN_SWITCH	BOOL		
SETPOINT	DINT		
TEMP	REAL		
IoConfig_Globals	VAR_GLOBAL		
{ IoDrvEthernet	Library		IoDrvEthernet, 4.2....

ATOM_INPUT_TEMP: REAL(VAR)

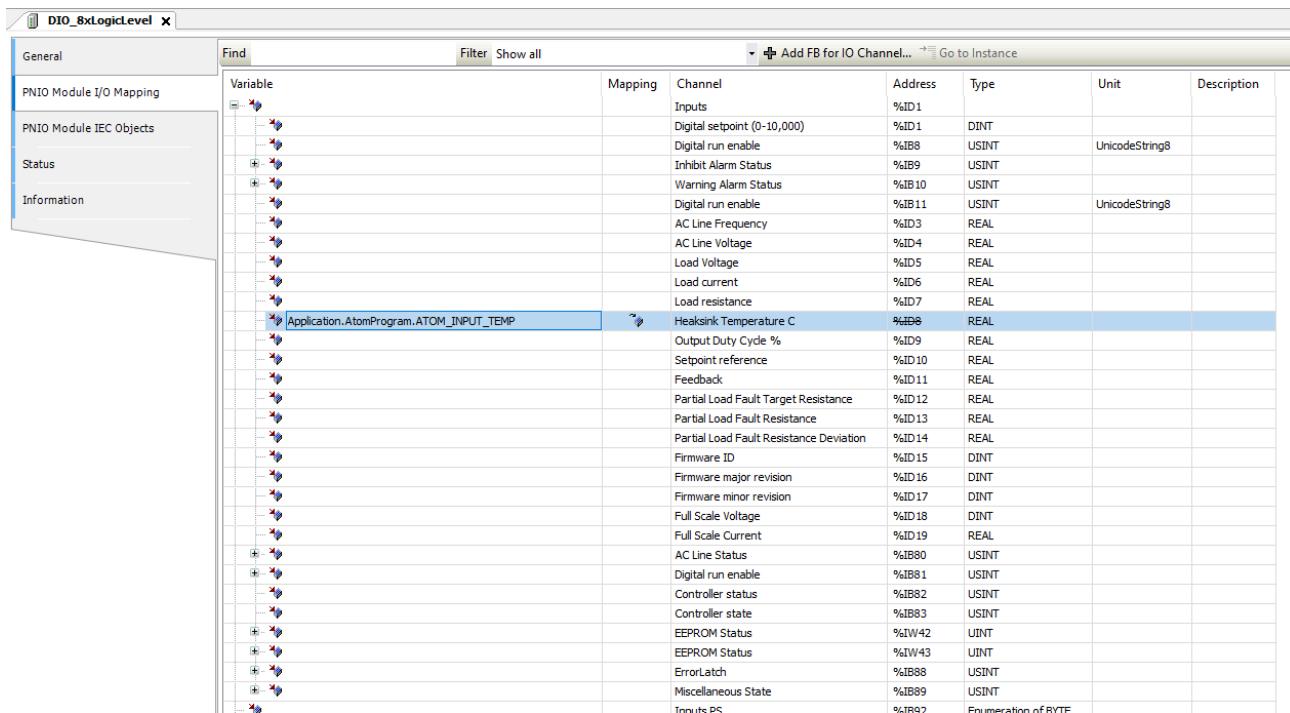
Structured view Filter None

Documentation

Insert with arguments Insert with namespace prefix

Add Library... OK Cancel

After doing so, your input I/O mappings should look like:



The screenshot shows the 'DIO_8xLogicLevel' configuration window. On the left, there's a sidebar with tabs: General, PNIO Module I/O Mapping (selected), PNIO Module IEC Objects, Status, and Information. The main area has a 'Find' bar with 'Filter Show all' and a toolbar with 'Add FB for IO Channel...' and 'Go to Instance'. A tree view under 'Variable' shows various I/O objects. A table lists the mapping from these objects to physical addresses and data types. One row is highlighted: 'Application.AtomProgram.ATOM_INPUT_TEMP' is mapped to channel %ID8, address %ID8, type REAL, unit none, and description 'Heatsink Temperature C'. Other rows include digital inputs like Digital setpoint (0-10,000) at %ID1, digital run enable at %IB8, and inhibit alarm status at %IB9. There are also analog inputs for AC Line Frequency (%ID3), AC Line Voltage (%ID4), Load Voltage (%ID5), Load current (%ID6), and Load resistance (%ID7). The table continues with various digital and analog outputs, status bits, and EEPROM data.

Variable	Mapping	Channel	Address	Type	Unit	Description
Digital setpoint (0-10,000)		%ID1	DINT			
Digital run enable		%IB8	USINT	UnicodeString8		
Inhibit Alarm Status		%IB9	USINT			
Warning Alarm Status		%IB10	USINT			
Digital run enable		%IB11	USINT	UnicodeString8		
AC Line Frequency		%ID3	REAL			
AC Line Voltage		%ID4	REAL			
Load Voltage		%ID5	REAL			
Load current		%ID6	REAL			
Load resistance		%ID7	REAL			
Application.AtomProgram.ATOM_INPUT_TEMP		%ID8	REAL			
Output Duty Cycle %		%ID9	REAL			
Setpoint reference		%ID10	REAL			
Feedback		%ID11	REAL			
Partial Load Fault Target Resistance		%ID12	REAL			
Partial Load Fault Resistance		%ID13	REAL			
Partial Load Fault Resistance Deviation		%ID14	REAL			
Firmware ID		%ID15	DINT			
Firmware major revision		%ID16	DINT			
Firmware minor revision		%ID17	DINT			
Full Scale Voltage		%ID18	DINT			
Full Scale Current		%ID19	REAL			
AC Line Status		%IB80	USINT			
Digital run enable		%IB81	USINT			
Controller status		%IB82	USINT			
Controller state		%IB83	USINT			
EEPROM Status		%IW42	UINT			
EEPROM Status		%IW43	UINT			
ErrorLatch		%IB88	USINT			
Miscellaneous State		%IB89	USINT			
Intrinsic PC		%TR02	Enumeration of RYTF			

Repeat this for your output I/O mappings:

1. Map **Digital setpoint** to **Application.AtomProgram.ATOM_OUTPUT_SETPOINT**
2. Map **Digital run enable** to **Application.AtomProgram.ATOM_OUTPUT_RUN_ENABLE**

Change the **Filter** to **Show only outputs** and repeat the process for the outputs. Map **Digital setpoint** to **Application.AtomProgram.ATOM_OUTPUT_SETPOINT** and **Digital RUN Enable** to **Application.AtomProgram.ATOM_OUTPUT_RUN_ENABLE**.

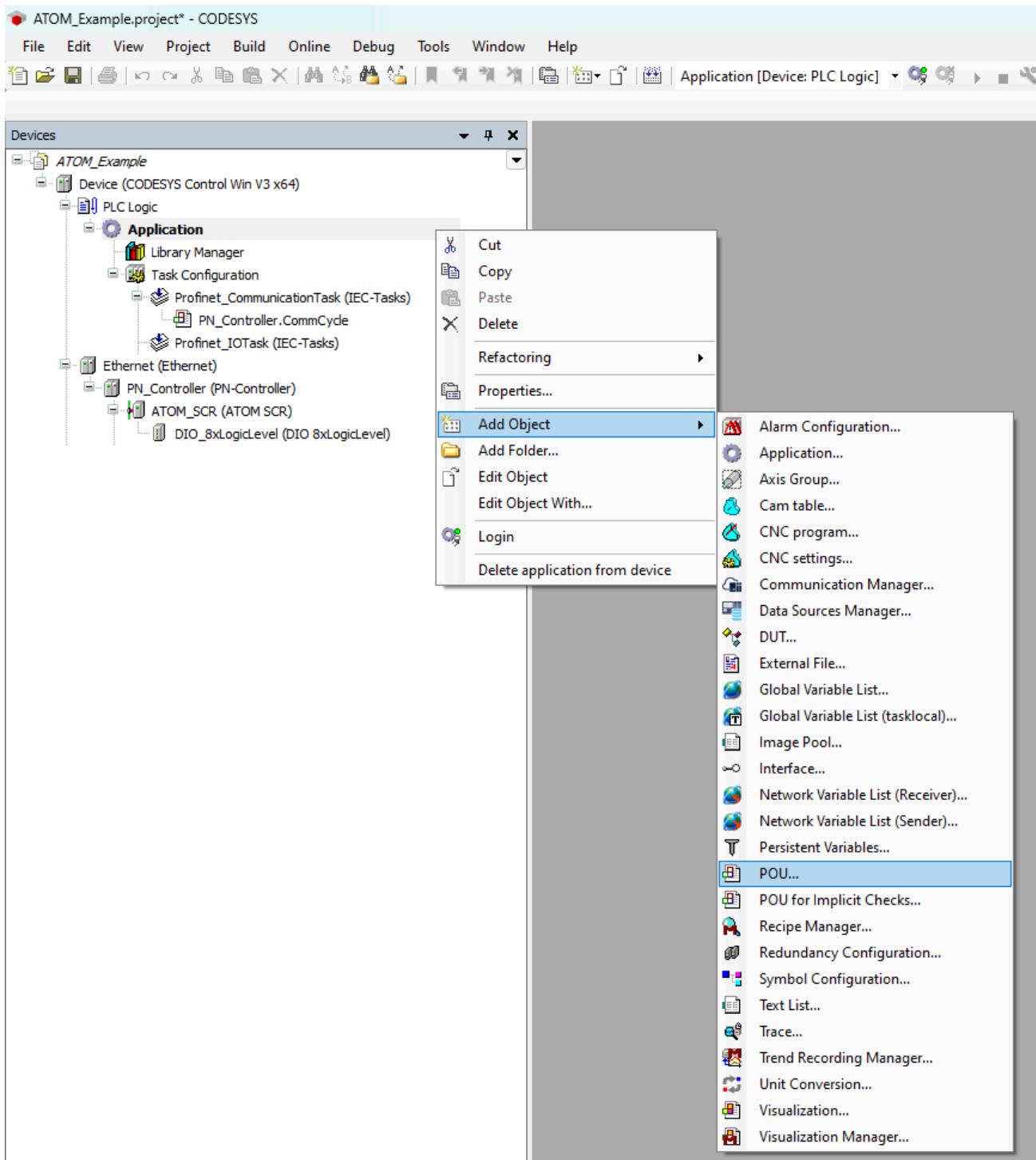
DIO_BxLogicLevel x								
General	Find	Filter Show all	Add FB for IO Channel...	Go to Instance				
PNIO Module I/O Mapping	Variable	Mapping	Channel	Address	Type	Current Value	Prepared Value	Unit
PNIO Module IEC Objects								
Status	Digital setpoint (0-10,000)	%ID1	DINT		Only subelements updated			
Information	Digital run enable	%ID8	USINT		Not updated (check tooltip)			
	Inhibit Alarm Status	%IB9	USINT		Not updated (check tooltip)		UnicodeString8	
	Warning Alarm Status	%IB10	USINT		Not updated (check tooltip)			
	Digital run enable	%IB11	USINT		Not updated (check tooltip)		UnicodeString8	
	AC Line Frequency	%ID3	REAL		Not updated (check tooltip)			
	AC Line Voltage	%ID4	REAL		Not updated (check tooltip)			
	Load Voltage	%ID5	REAL		Not updated (check tooltip)			
	Load current	%ID6	REAL		Not updated (check tooltip)			
	Load resistance	%ID7	REAL		Not updated (check tooltip)			
	Application.AtomProgram.ATOM_INPUT_TEMP							
	Heatsink Temperature C	%EB8	REAL	29.2				
	Output Duty Cycle %	%ID9	REAL		Not updated (check tooltip)			
	Setpoint reference	%ID10	REAL		Not updated (check tooltip)			
	Feedback	%ID11	REAL		Not updated (check tooltip)			
	Partial Load Fault Target Resistance	%ID12	REAL		Not updated (check tooltip)			
	Partial Load Fault Resistance	%ID13	REAL		Not updated (check tooltip)			
	Partial Load Fault Resistance Deviation	%ID14	REAL		Not updated (check tooltip)			
	Firmware ID	%ID15	DINT		Not updated (check tooltip)			
	Firmware major revision	%ID16	DINT		Not updated (check tooltip)			
	Firmware minor revision	%ID17	DINT		Not updated (check tooltip)			
	Full Scale Voltage	%ID18	DINT		Not updated (check tooltip)			
	Full Scale Current	%ID19	REAL		Not updated (check tooltip)			
	AC Line Status	%EB80	USINT		Not updated (check tooltip)			
	Digital run enable	%IB81	USINT		Not updated (check tooltip)			
	Controller status	%EB82	USINT		Not updated (check tooltip)			
	Controller state	%EB83	USINT		Not updated (check tooltip)			
	EEPROM Status	%IV42	UDINT		Not updated (check tooltip)			
	EEPROM Status	%IV43	UDINT		Not updated (check tooltip)			
	ErrorLatch	%EB88	USINT		Not updated (check tooltip)			
	Miscellaneous State	%EB89	USINT		Not updated (check tooltip)			
	Inputs PS	%IB92	Enumeration of BYTE		Not updated (check tooltip)			
	Outputs	%QD0			Only subelements updated			
	Digital setpoint (0-10,000)	%QB96	DINT	5000				
	Digital run enable	%QB94	USINT	1			UnicodeString8	
	Outputs CS	%IB93	Enumeration of BYTE		Not updated (check tooltip)			

You're all set! Go to the [Running the program with SoftPLC](#) section to run your program.

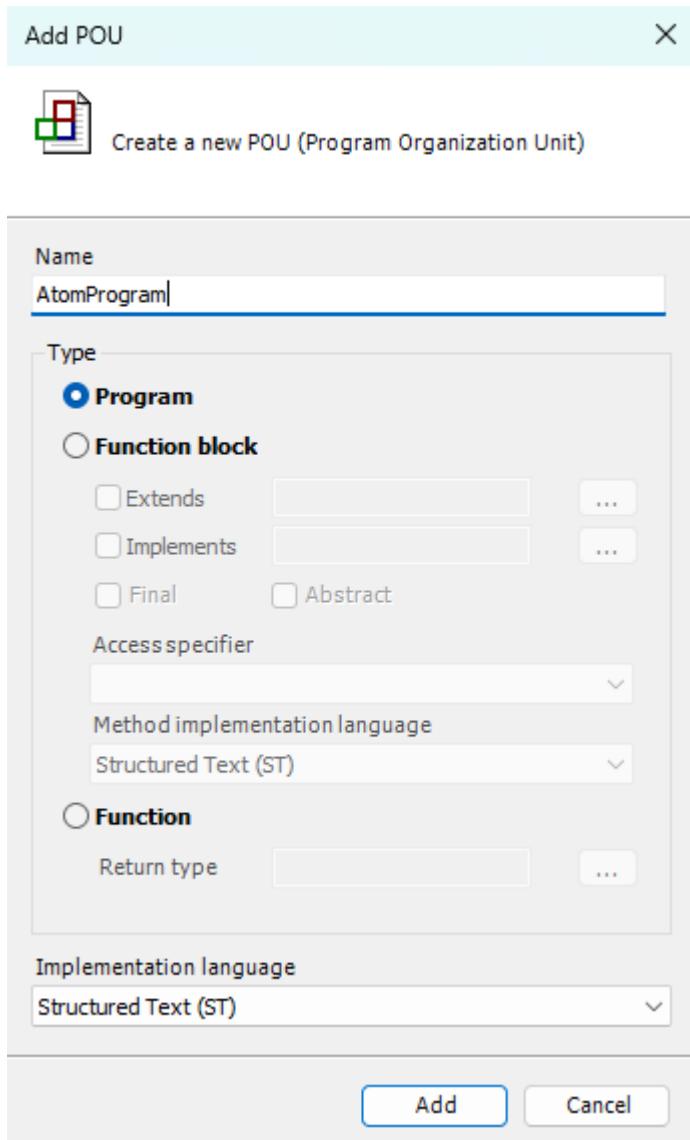
Example: Structured text

Creating the program

Right click **Application** and select **Add Object > POU:**



Name your **POU** **AtomProgram** and select **Structured Text (ST)** as the language:



Next, let's create a basic program. We'll check to make sure no alarms are active and then write a setpoint value of `8000` and set run enable to `true`.

Copy the following code into the top panel of the **AtomProgram** editor:

```
PROGRAM AtomProgram
VAR

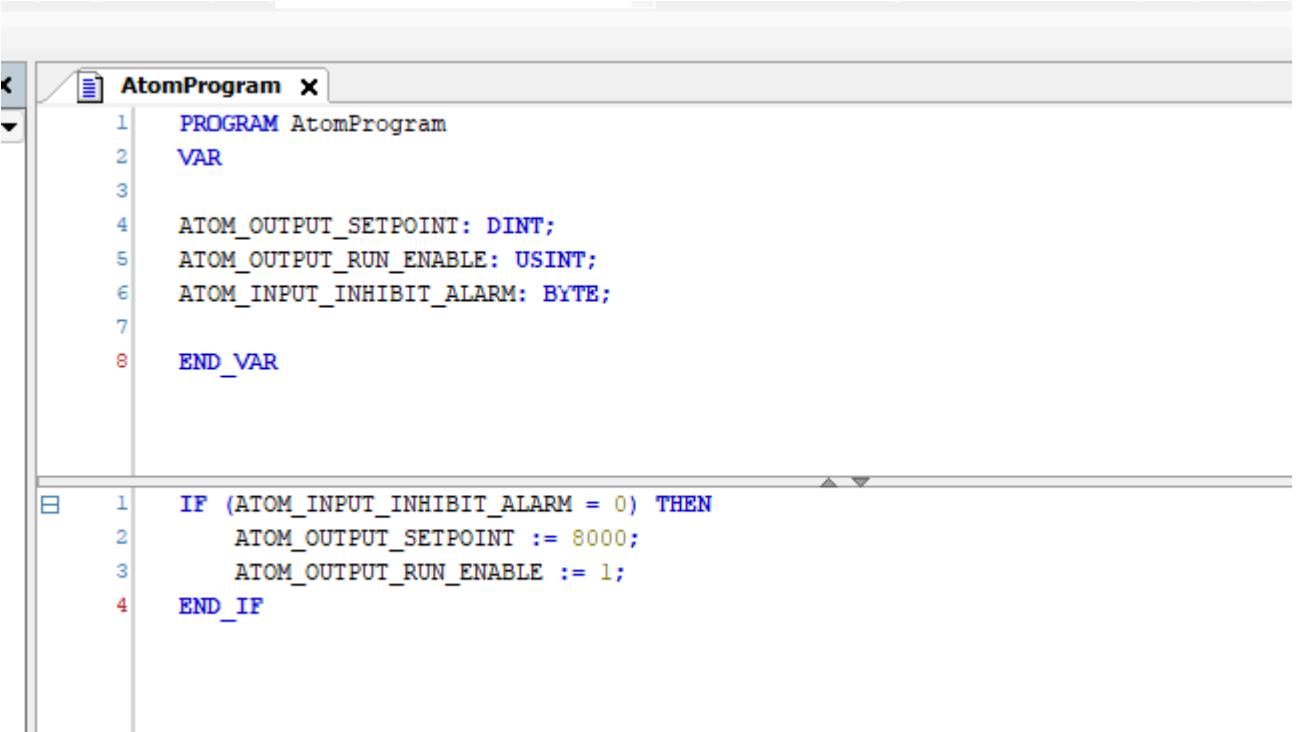
ATOM_OUTPUT_SETPOINT: DINT;
ATOM_OUTPUT_RUN_ENABLE: USINT;
ATOM_INPUT_INHIBIT_ALARM: BYTE;

END_VAR
```

Copy the following code into the main program section:

```
IF (ATOM_INPUT_INHIBIT_ALARM = 0) THEN
    ATOM_OUTPUT_SETPOINT := 8000;
    ATOM_OUTPUT_RUN_ENABLE := 1;
END_IF
```

Your editor should look like:



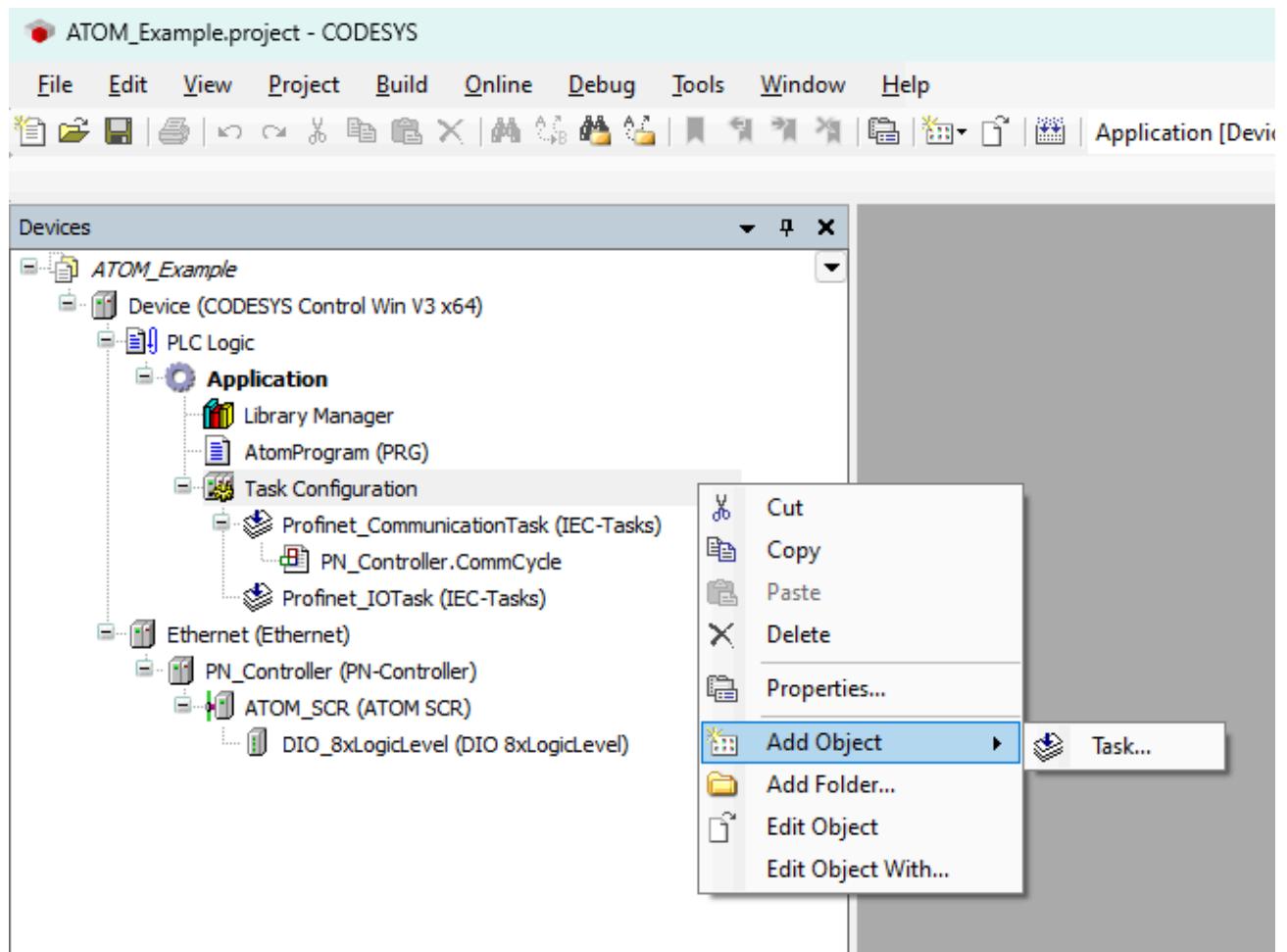
The screenshot shows a software interface for editing PLC programs. The window title is "AtomProgram". The code area contains the following program structure:

```
PROGRAM AtomProgram
VAR
ATOM_OUTPUT_SETPOINT: DINT;
ATOM_OUTPUT_RUN_ENABLE: USINT;
ATOM_INPUT_INHIBIT_ALARM: BYTE;
END_VAR

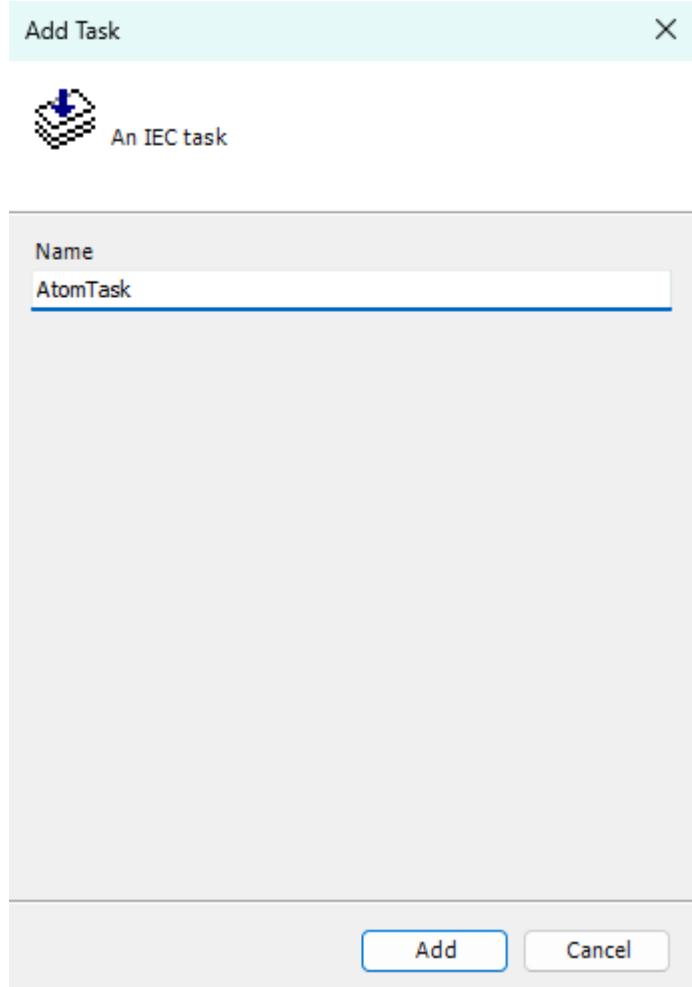
IF (ATOM_INPUT_INHIBIT_ALARM = 0) THEN
    ATOM_OUTPUT_SETPOINT := 8000;
    ATOM_OUTPUT_RUN_ENABLE := 1;
END_IF
```

The code is color-coded: blue for reserved words like `PROGRAM`, `VAR`, `IF`, `THEN`, `END_IF`, etc.; black for variable names; and grey for comments. The `END_VAR` line is also greyed out. The `IF` block is indented under the `END_VAR` line, indicating it is part of the main program's variable declarations.

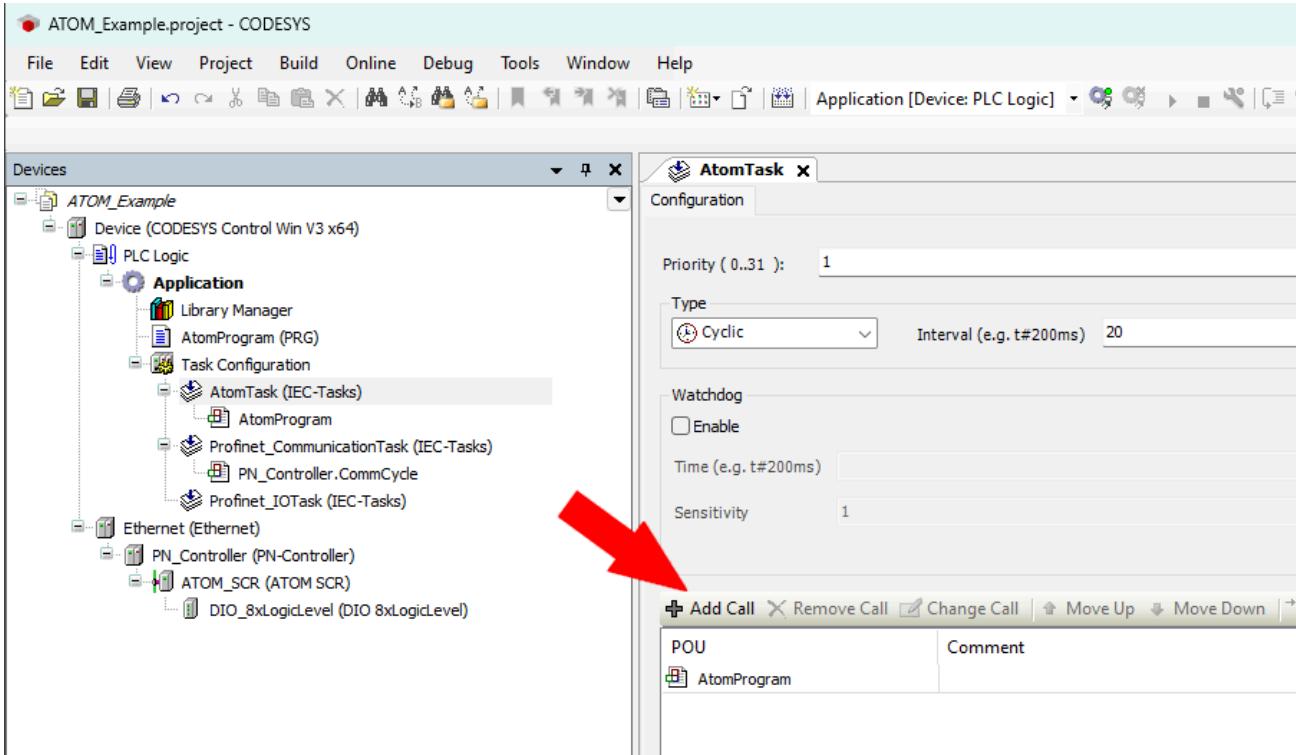
Next, we'll add a new task to call our program. Right click **Task Configuration** and Select **Add Object > Task**:



Name your task **AtomTask** and click **Add**:

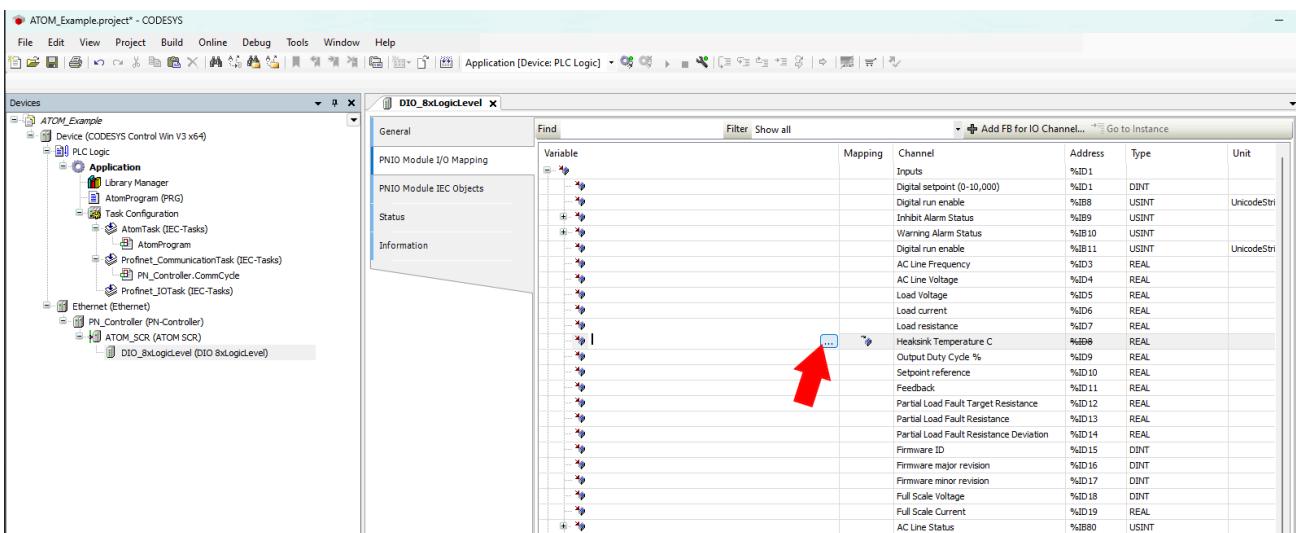


Next, double click **AtomTask (IEC-Tasks)** to open its configuration tab. Click **Add Call** and select **Application > AtomProgram**. After doing so, **AtomTask**'s configuration should look like:

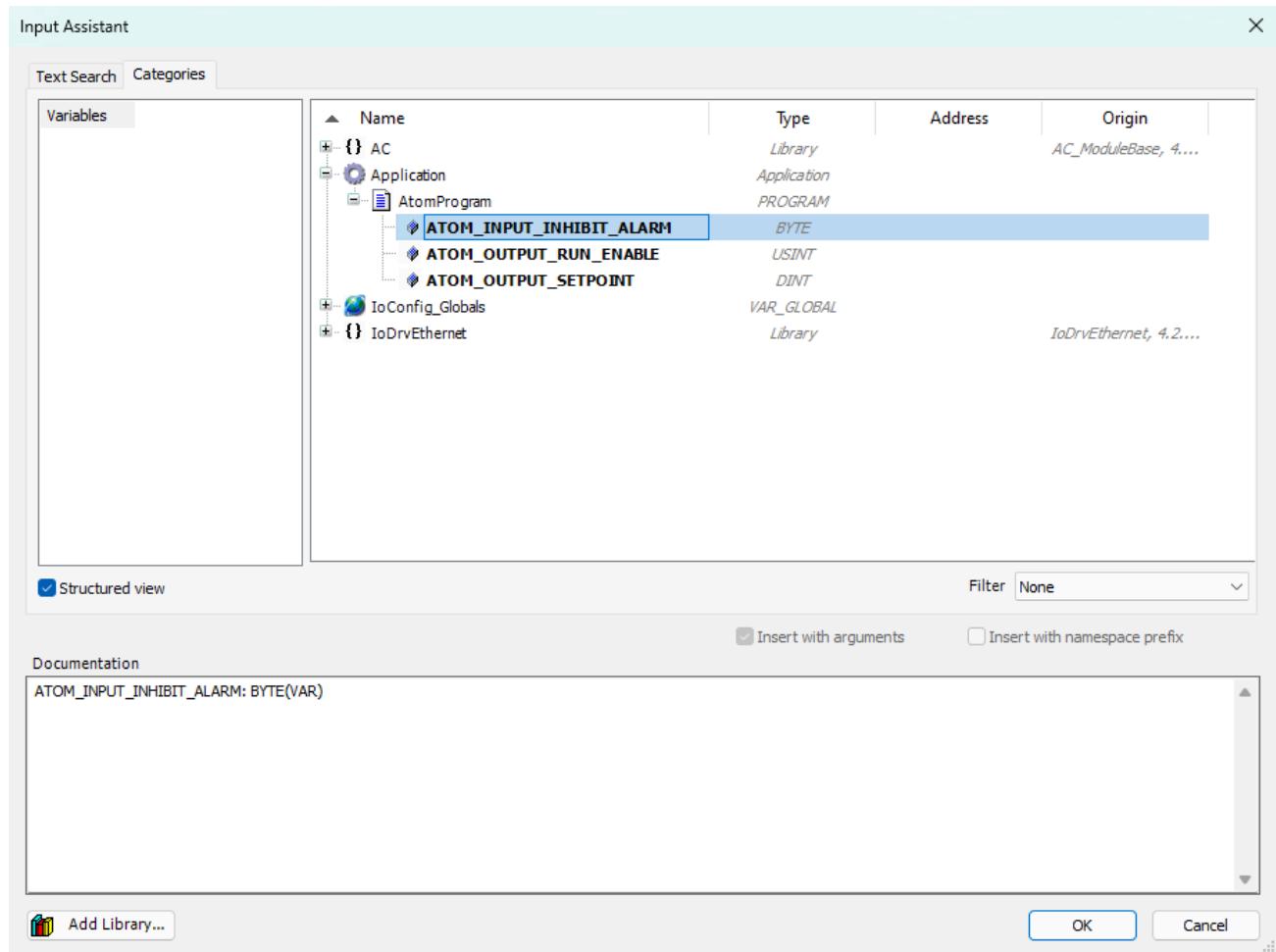


Mapping variables

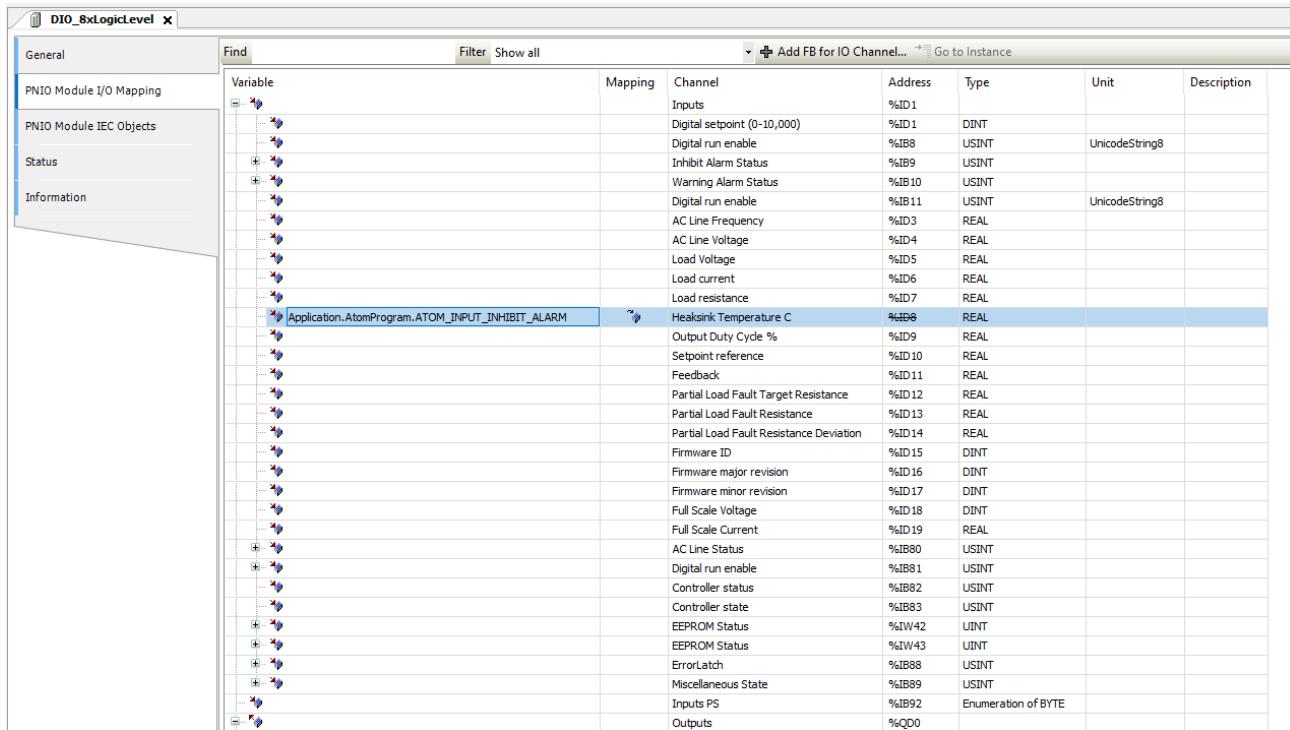
Finally, we'll map our PLC variables to ATOM. Double click **DIO_8xLogicLevel (DIO 8xLogicLevel)** in the device tree to open its configuration window. Select the **PNIO Module I/O Mapping** tab:



Above, select the button indicated by the red arrow. This will open the **Input Assistant** dialog. Select **Application > AtomProgram > ATOM_INPUT_INHIBIT_ALARM** and click **Add:**



After doing so, your input I/O mappings should look like:



The screenshot shows the 'DIO_8xLogicLevel' configuration window. On the left, there's a sidebar with tabs for 'General', 'PNIO Module I/O Mapping' (which is selected), 'PNIO Module IEC Objects', 'Status', and 'Information'. The main area is a table titled 'Variable' with columns for 'Mapping', 'Channel', 'Address', 'Type', 'Unit', and 'Description'. A search bar at the top of the table area includes 'Find', 'Filter', 'Show all', and buttons for 'Add FB for IO Channel...' and 'Go to Instance'. The table lists various variables, many of which are collapsed under a '+' icon. One row is highlighted in blue, showing 'Application.AtomProgram.ATOM_INPUT_INHIBIT_ALARM' in the 'Mapping' column and '%ID0' in the 'Address' column. Other visible rows include 'Digital setpoint (0-10,000)', 'Digital run enable', 'Inhibit Alarm Status', 'Warning Alarm Status', 'Digital run enable', 'AC Line Frequency', 'AC Line Voltage', 'Load Voltage', 'Load current', 'Load resistance', 'Heatsink Temperature C', 'Output Duty Cycle %', 'Setpoint reference', 'Feedback', 'Partial Load Fault Target Resistance', 'Partial Load Fault Resistance', 'Partial Load Fault Resistance Deviation', 'Firmware ID', 'Firmware major revision', 'Firmware minor revision', 'Full Scale Voltage', 'Full Scale Current', 'AC Line Status', 'Digital run enable', 'Controller status', 'Controller state', 'EEPROM Status', 'EEPROM Status', 'ErrorLatch', 'Miscellaneous State', 'Inputs PS', and 'Outputs'.

Variable	Mapping	Channel	Address	Type	Unit	Description
Digital setpoint (0-10,000)	%ID1	DINT				
Digital run enable	%IB8	USINT		UnicodeString8		
Inhibit Alarm Status	%IB9	USINT				
Warning Alarm Status	%IB10	USINT				
Digital run enable	%IB11	USINT		UnicodeString8		
AC Line Frequency	%ID3	REAL				
AC Line Voltage	%ID4	REAL				
Load Voltage	%ID5	REAL				
Load current	%ID6	REAL				
Load resistance	%ID7	REAL				
Application.AtomProgram.ATOM_INPUT_INHIBIT_ALARM	%ID0	REAL				
Digital setpoint	%ID9	REAL				
Output Duty Cycle %	%ID10	REAL				
Setpoint reference	%ID11	REAL				
Feedback	%ID12	REAL				
Partial Load Fault Target Resistance	%ID13	REAL				
Partial Load Fault Resistance	%ID14	REAL				
Partial Load Fault Resistance Deviation	%ID15	DINT				
Firmware ID	%ID16	DINT				
Firmware major revision	%ID17	DINT				
Firmware minor revision	%ID18	DINT				
Full Scale Voltage	%ID19	REAL				
Full Scale Current	%IB80	USINT				
AC Line Status	%IB81	USINT				
Digital run enable	%IB82	USINT				
Controller status	%IB83	USINT				
Controller state	%IW42	UINT				
EEPROM Status	%IW43	UINT				
ErrorLatch	%IB88	USINT				
Miscellaneous State	%IB89	USINT				
Inputs PS	%IB92	Enumeration of BYTE				
Outputs	%QD0					

Repeat this for your output I/O mappings:

1. Map **Digital setpoint** to `Application.AtomProgram.ATOM_OUTPUT_SETPOINT`
2. Map **Digital run enable** to `Application.AtomProgram.ATOM_OUTPUT_RUN_ENABLE`

Change the **Filter** to **Show only outputs** and repeat the process for the outputs. Map **Digital setpoint** to `Application.AtomProgram.ATOM_OUTPUT_SETPOINT` and **Digital RUN Enable** to `Application.AtomProgram.ATOM_OUTPUT_RUN_ENABLE`.

Variable	Mapping	Channel	Address	Type	Unit	Description
Inputs			%ID1	DINT		
Digital setpoint (0-10,000)			%ID1	DINT		
Digital run enable			%IB8	USINT	UnicodeString8	
Inhibit Alarm Status			%IB9	USINT		
Warning Alarm Status			%IB10	USINT		
Digital run enable			%IB11	USINT	UnicodeString8	
AC Line Frequency			%ID3	REAL		
AC Line Voltage			%ID4	REAL		
Load Voltage			%ID5	REAL		
Load current			%ID6	REAL		
Load resistance			%ID7	REAL		
Application.AtomProgram.ATOM_INPUT_INHIBIT_ALARM						
Heatsink Temperature C			%ID9	REAL		
Output Duty Cycle %			%ID9	REAL		
Setpoint reference			%ID10	REAL		
Feedback			%ID11	REAL		
Partial Load Fault Target Resistance			%ID12	REAL		
Partial Load Fault Resistance			%ID13	REAL		
Partial Load Fault Resistance Deviation			%ID14	REAL		
Firmware ID			%ID15	DINT		
Firmware major revision			%ID16	DINT		
Firmware minor revision			%ID17	DINT		
Full Scale Voltage			%ID18	DINT		
Full Scale Current			%ID19	REAL		
AC Line Status			%IB80	USINT		
Digital run enable			%IB81	USINT		
Controller status			%IB82	USINT		
Controller state			%IB83	USINT		
EEPROM Status			%IW42	UINT		
EEPROM Status			%IW43	UINT		
ErrorLatch			%IB88	USINT		
Miscellaneous State			%IB89	USINT		
Inputs PS			%IB92	Enumeration of BYTE		
Outputs			%QD0			
Application.AtomProgram.ATOM_OUTPUT_SETPOINT						
Digital setpoint (0-10,000)			%QB0	DINT		
Digital run enable			%QB4	USINT	UnicodeString8	
Outputs CS			%IB93	Enumeration of BYTE		

You're all set! Go to the [Running the program with SoftPLC](#) section to run your program.

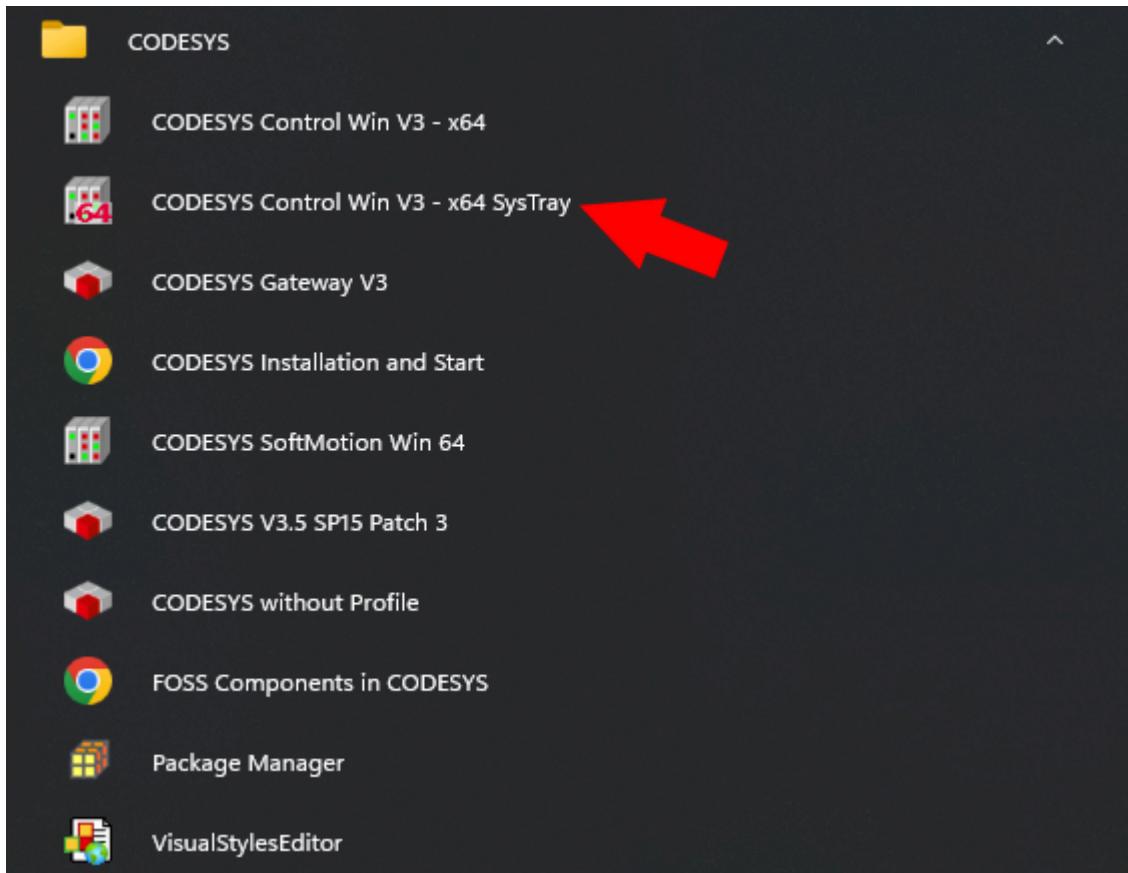
Running the program with SoftPLC

INFO

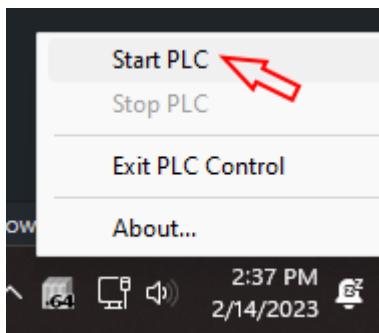
The instructions to run your program are the same regardless of whether you are using ladder logic or structured text.

The only difference is that in the ladder logic example, a visualization window will open that allows you to control ATOM.

To debug the program, first make sure you start **Codesys WIN Control V3 - x64 SysTray**

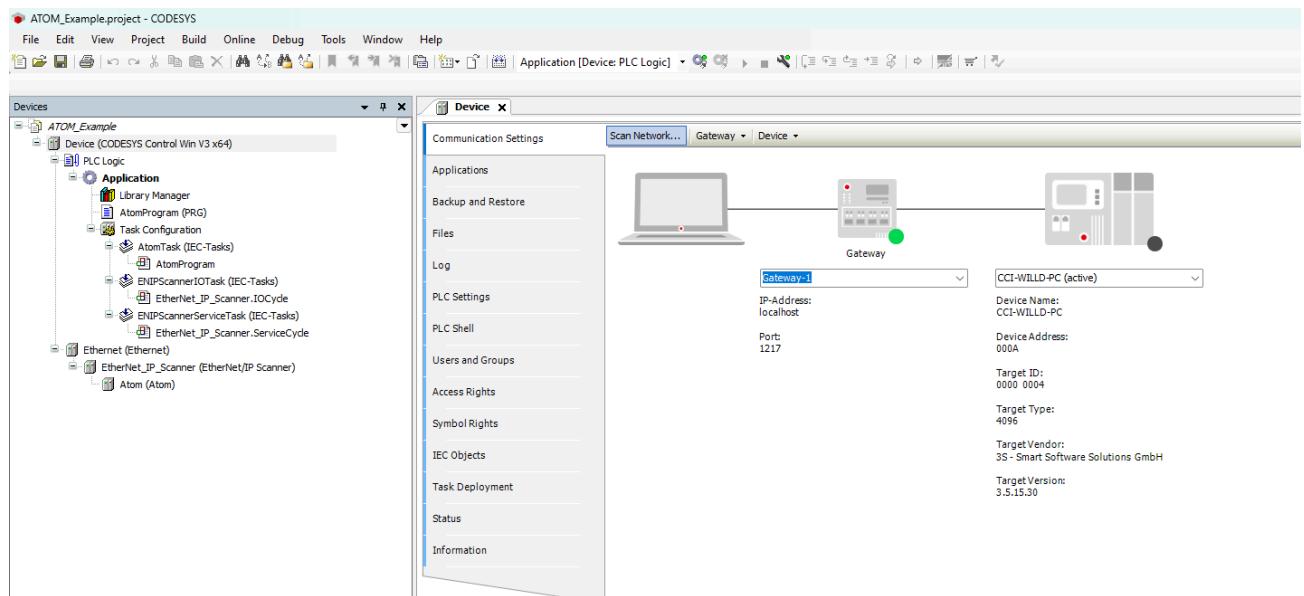


This will launch the Codesys SoftPLC. You should see an icon appear in your systray and you can right click it and select **Start PLC** to start the SoftPLC:

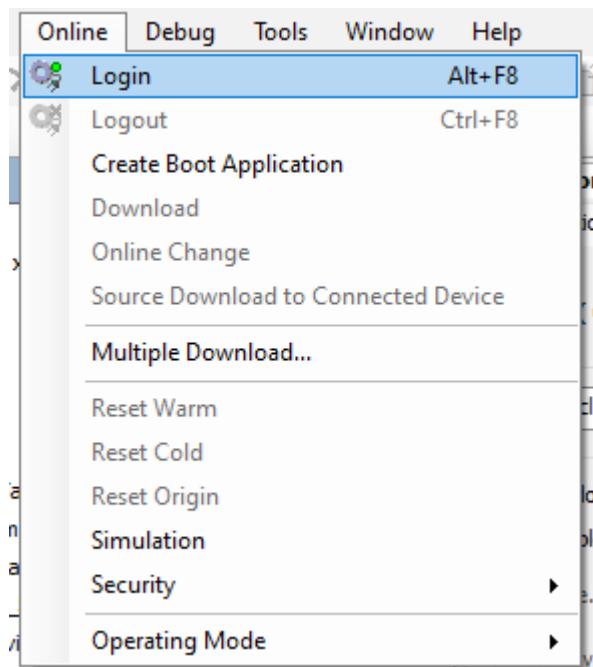


Next, connect your Atom to your PC via an Ethernet cable, ensuring to use the network interface you specified in the [Adding a Profinet controller](#) section.

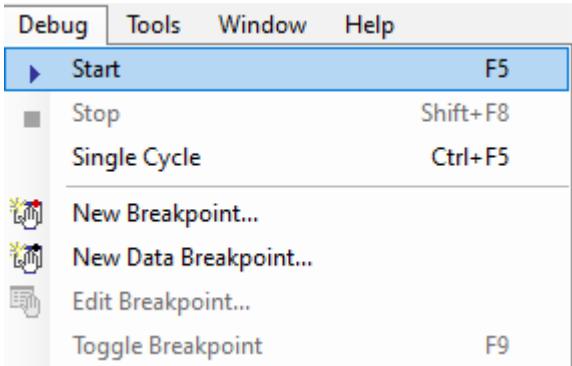
Next, in Codesys double click **Application** to open its configuration window. Here you can select **Scan Network** to discover your SoftPLC:



Finally, **Login** to your SoftPLC:



Then you can start debugging the program:



If you use Control Panel to monitor ATOM, you should see the **Stop / Run** state and the **Digital Setpoint** values change to reflect the PLC program's instructions. If you followed the structured text example, the values will change once and remain fixed. If you followed the ladder logic example, a visualization control panel will appear. Flipping the dip switch or adjusting the slider will immediately update ATOM and the changes should reflect in real-time:

