# docs

## ATOM / Fieldbus / EtherNet/IP

Author: Control Concepts, Inc.

Date: 2025-10-30

# Contents

# ATOM / Fieldbus / EtherNet/IP / Overview

**INFO**

ODVA CONFORMANT

ATOM is ODVA EtherNet/IP CT19 Conformant.

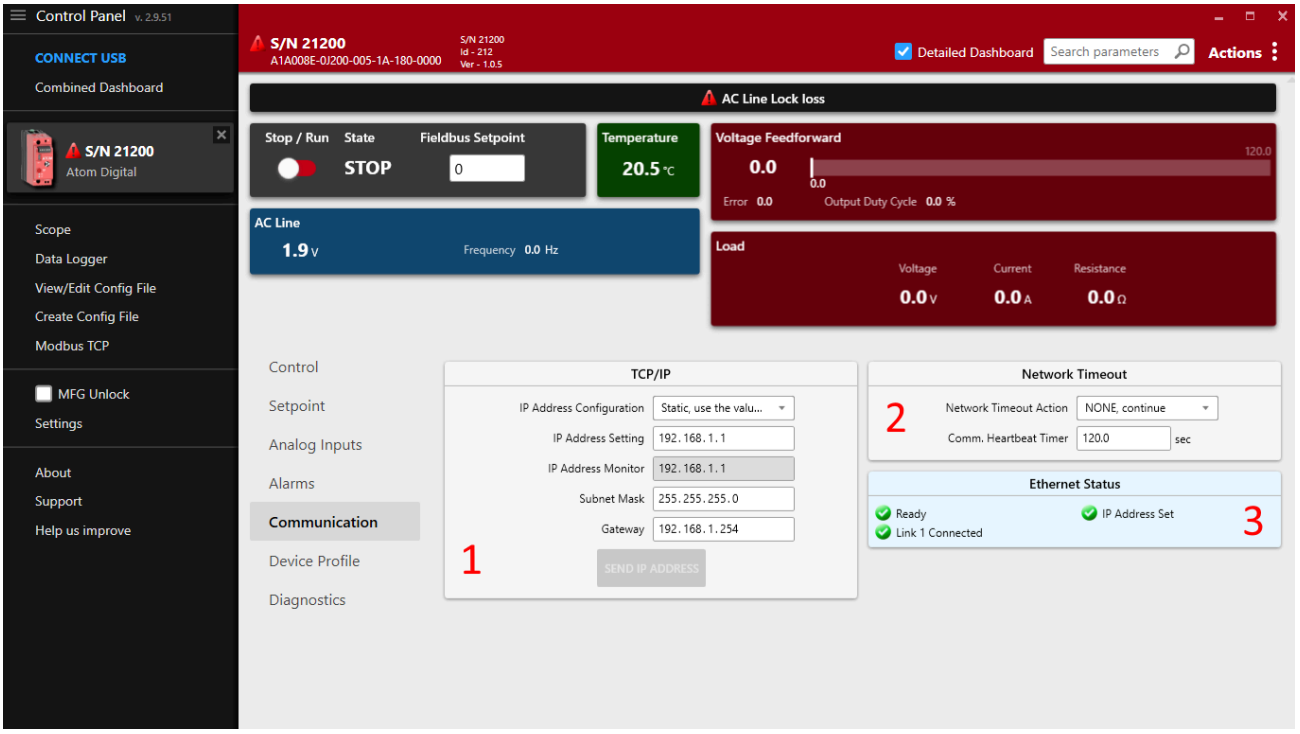
- **Statement of Conformance**
- **Declaration of Conformance**
- **Passing Test Report**
- **ODVA listing**

## EDS

**INFO**

Download the EDS 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 altnerative 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 EtherNet/IP. These parameters are made available through the CIP Assembly Object (code `0x04`) and a custom ParameterLink object (code `0x64`). The assembly object is most commonly used to read and write parameters from a PLC. The ParameterLink object is a custom object defined by Control Concepts that can be used to individually control parameters and is less commonly used.

## Output Assembly (Class `0x04`, Instance `0x01`)

| # | 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, | Read/Write |

| # | Name | Type | Description | Read/Write |
|---|---|---|---|---|
| | | | if the output range is 0-100A, 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 |

## Input Assembly (Class `0x04`, Instance `0x02`)

| # | Name | Type | Description | 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 |

| # | Name | Type | Description | Read/Write |
|---|---|---|---|---|
| 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 |
| 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 |

| # | Name | Type | Description | Read/Write |
|---|---|---|---|---|
| 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 features are available. | Read |
| 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 |

| # | Name | Type | Description | Read/Write |
|---|---|---|---|---|
| 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 |
| 26 | Controller State | BYTE | A value indicating the controller state. See Controller state. | Read |
| 27 | EEPROM Status | WORD | A bitfield indicating the EEPROM status. Seee 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 | Reserved | Reserved | Reserved | Reserved | Reserved | 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 |

| Value | Description |
|---|---|
| 3 | Calibration |
| 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 |
| 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 |

| Bit | Description |
|---|---|
| 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 Powered |

# 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 |
| 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 |

> **INFO**
>
> Rockwell's RSLogix Studio 5000 does not support unsigned integers. Any EDS file that contains unsigned integers will cause issues when it is imported into Studio 5000. To avoid this issue, ATOM uses signed integers for all integer types, regardless of whether the value may be negative or not. For example, parameter #1, *Digital setpoint* is represented as a signed 32-bit integer, but it may never be negative.

# Other resources

> **INFO**
>
> Detailed information about ATOM's EtherNet/IP profile is also available as a **downloadable word document**.

# Advanced

ATOM has many more parameters beyond the 30 made available through EtherNet/IP. 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 EtherNet/IP 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 EtherNet/IP profile. There may be a service fee for custom EtherNt/IP profiles as they require new EDS files, device-reconfiguration and testing.

# ATOM / Fieldbus / EtherNet/IP / RSLogix Studio 5000

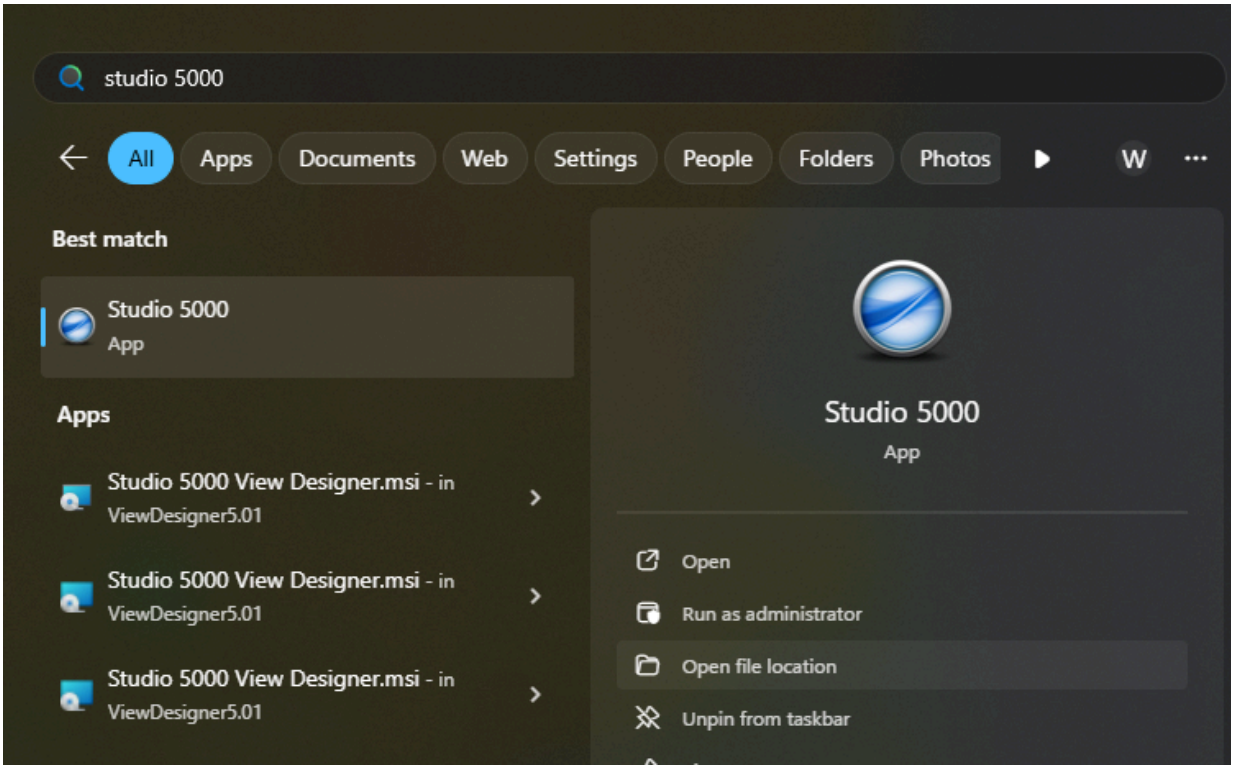
## Overview

In this tutorial, you'll learn how to control ATOM over EtherNet/IP with RSLogix Studio 5000.

> **NOTE**
>
> If you'd like to skip this tutorial, download the completed example project: AtomExampleStudio5000.zip.

## Prerequisites

1. A PC with RSLogix Studio 5000 installed (See Installation Troubleshooting for help with installation issues):

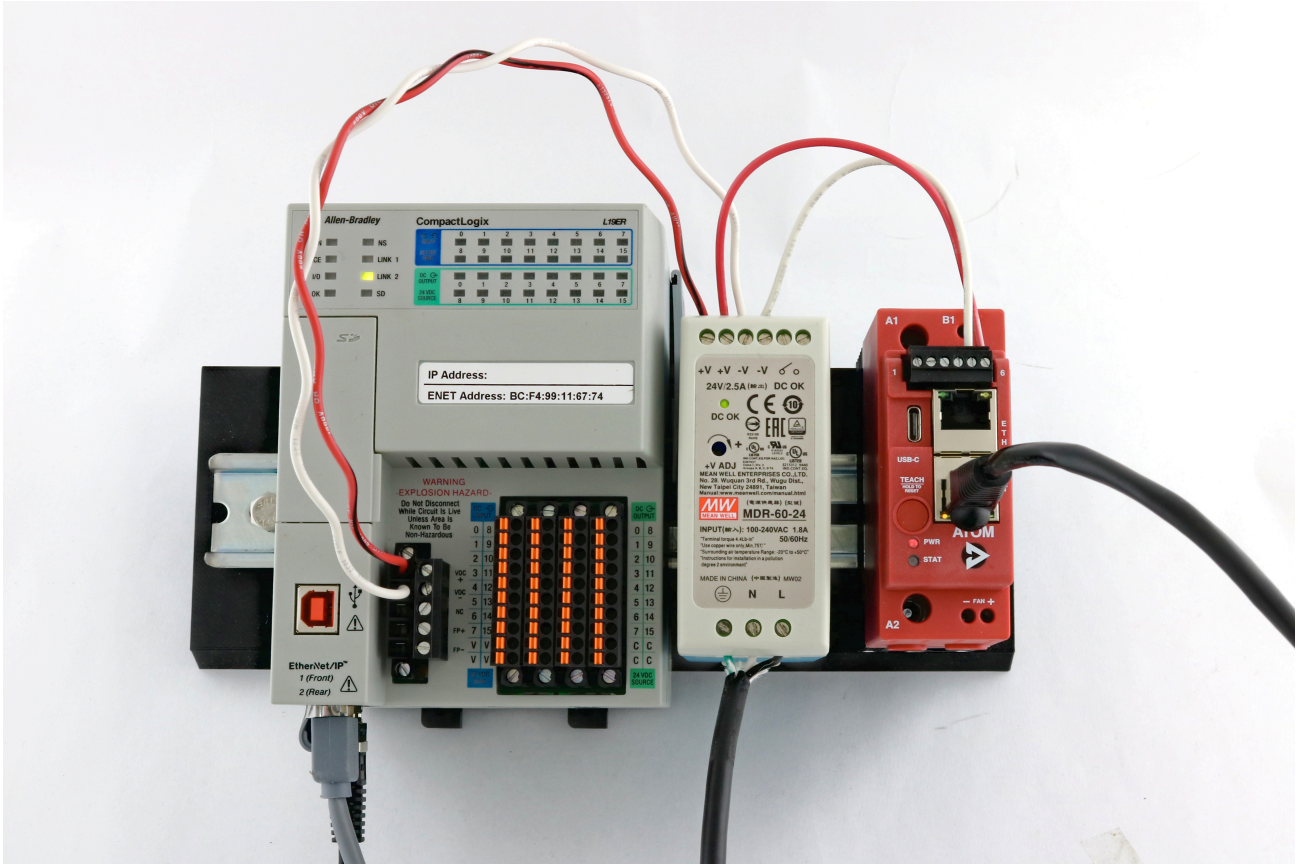



2. A Logix PLC - a CompactLogix `1769-L19ER-BB1B` is used in this example, but you can follow along with any Logix PLC that supports EtherNet/IP.
3. Download ATOM's EDS file: Atom.eds

# Hardware Setup

Connections:

- Connect port `1 (Front)` on your PLC to your *PC*
- Connect port `2 (Rear)` on your PLC to *ATOM* (either port)
- Connect a 24VDC power supply to ATOM and your PLC

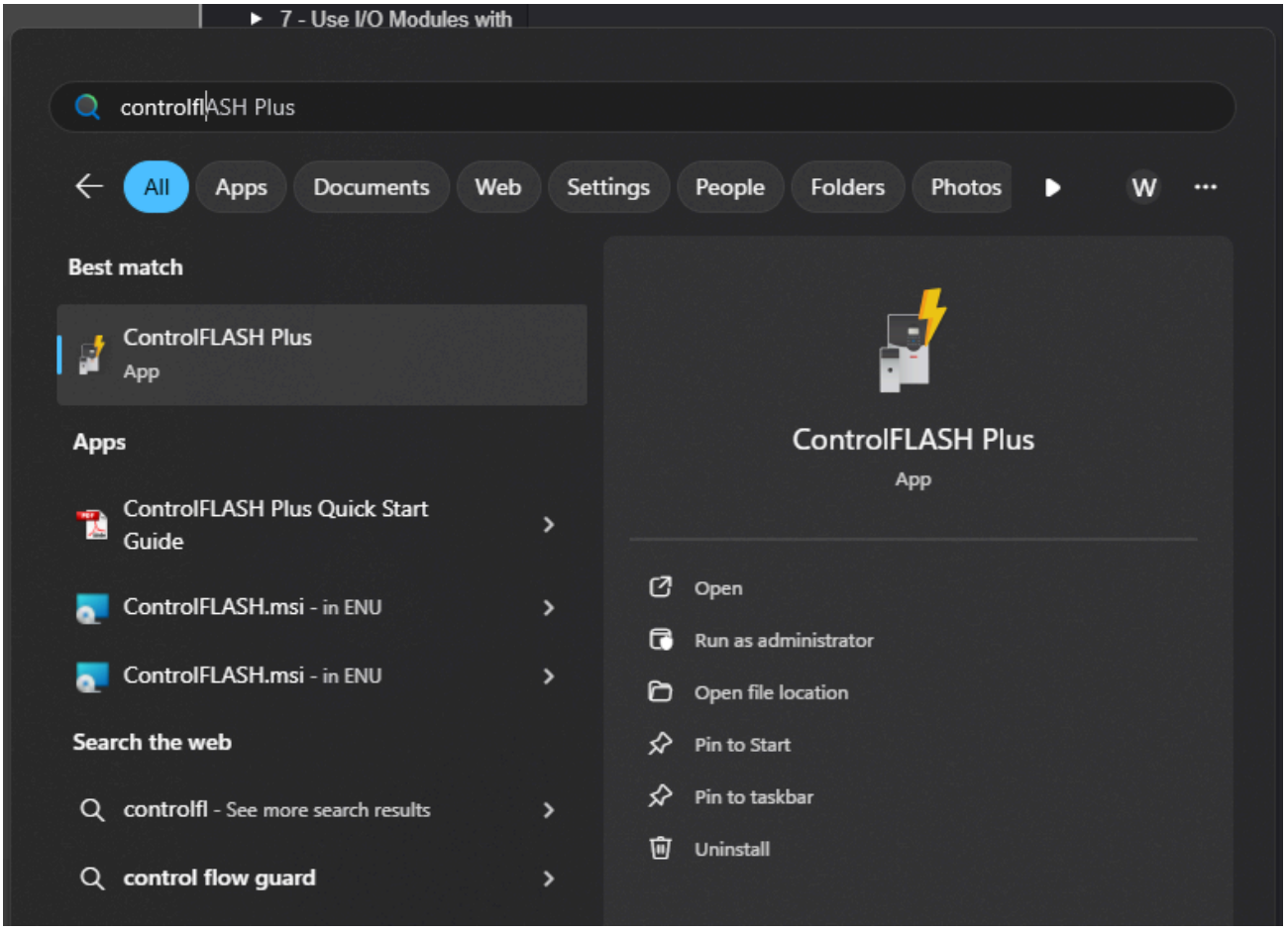



# PLC Configuration

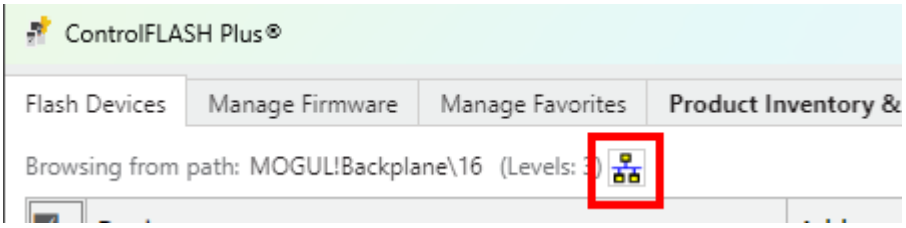
## Upgrading firmware

Make sure to upgrade your PLC firmware to the lastest version to ensure compatibility with Studio 5000.

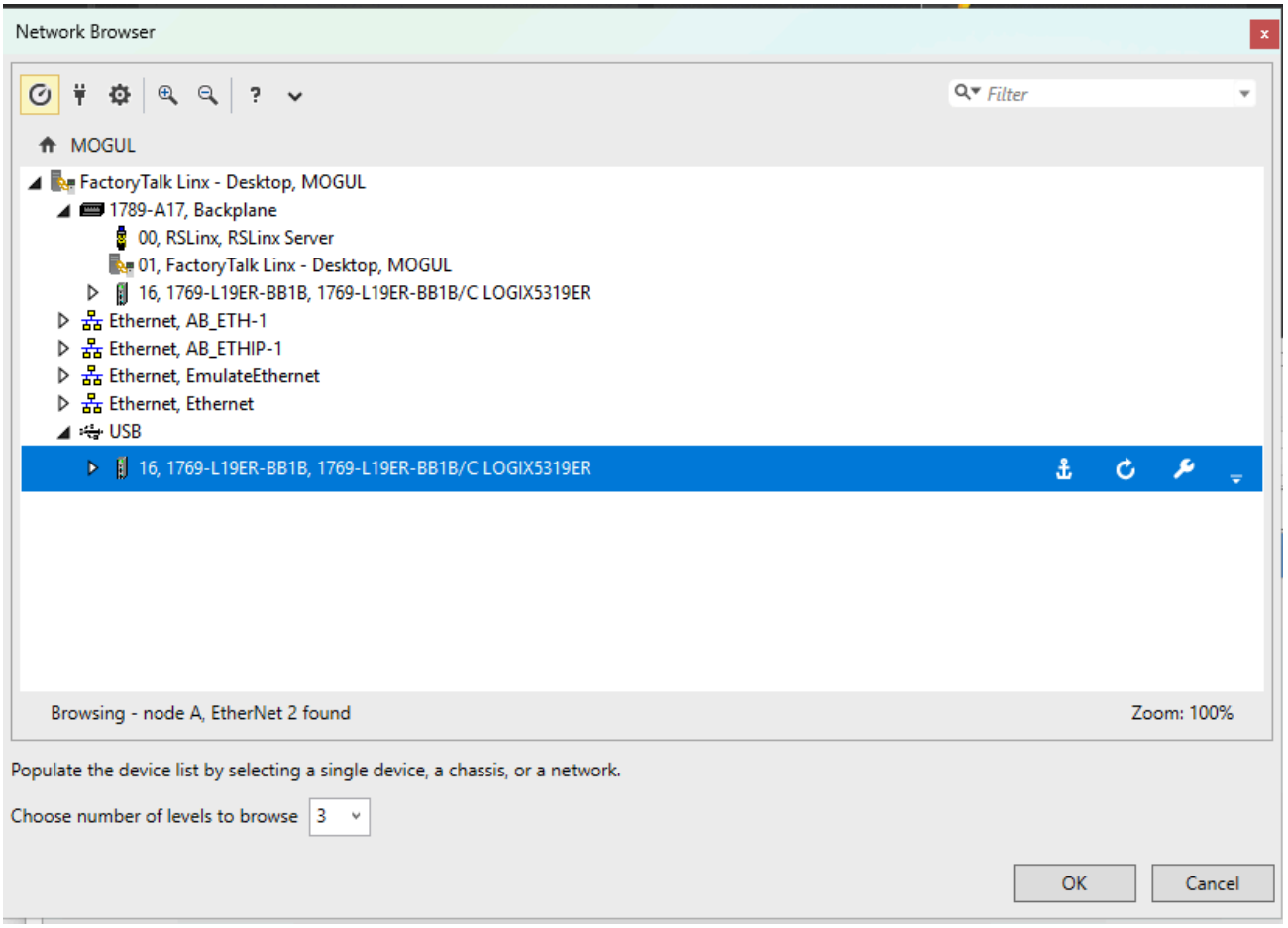
1. Connect your Logix PLC to your PC with a USB cable.
2. Launch `ControlFLASH Plus`:



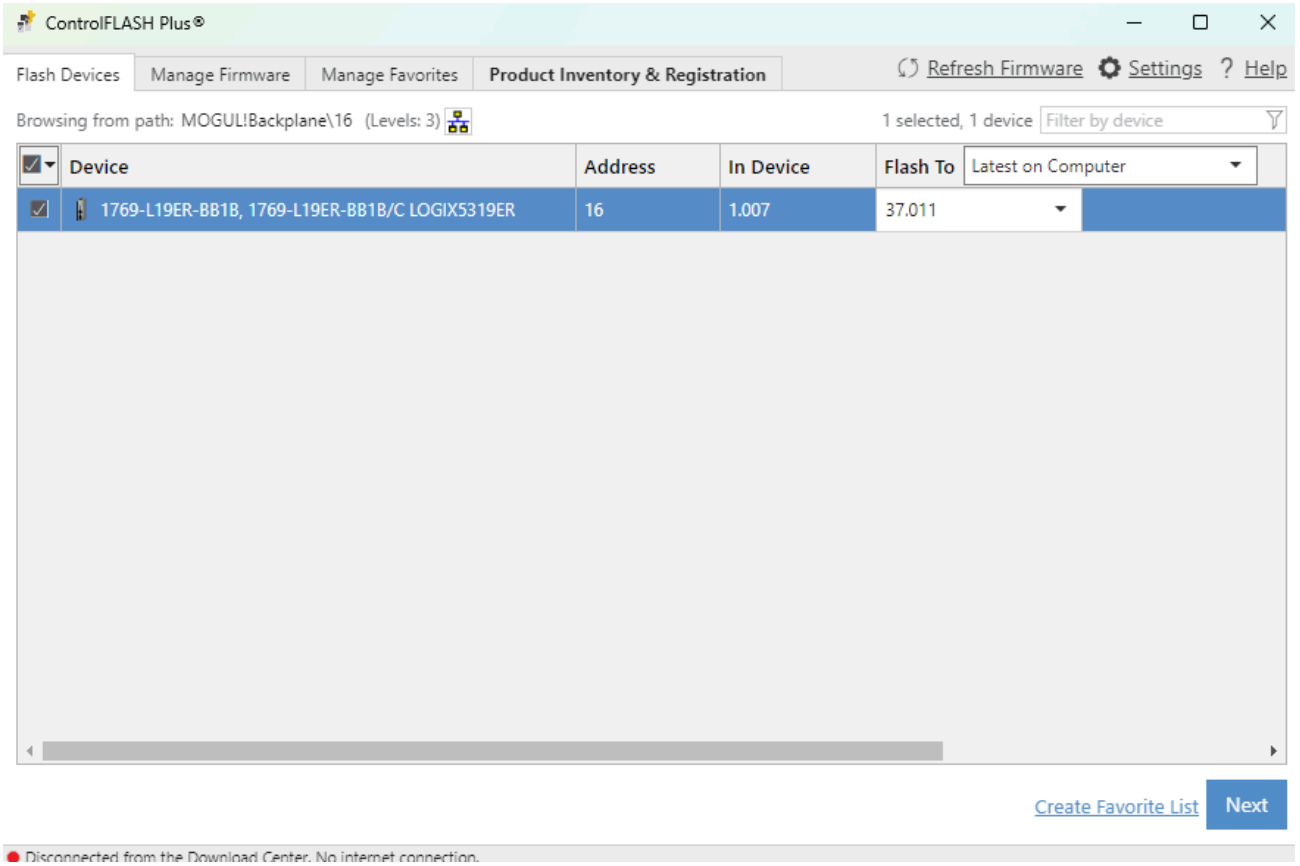

3. Open the network browser:



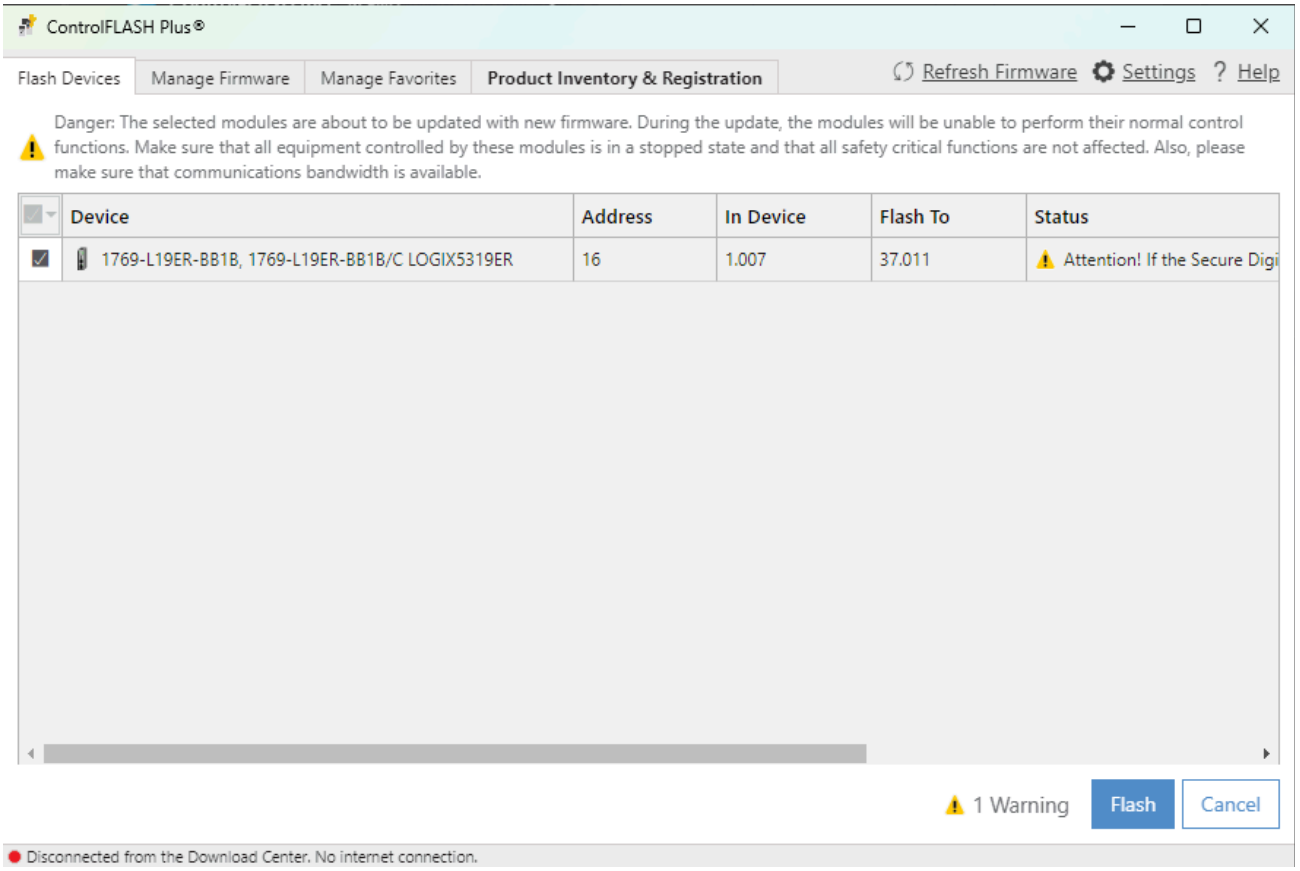

4. Select your PLC under the `USB` category:



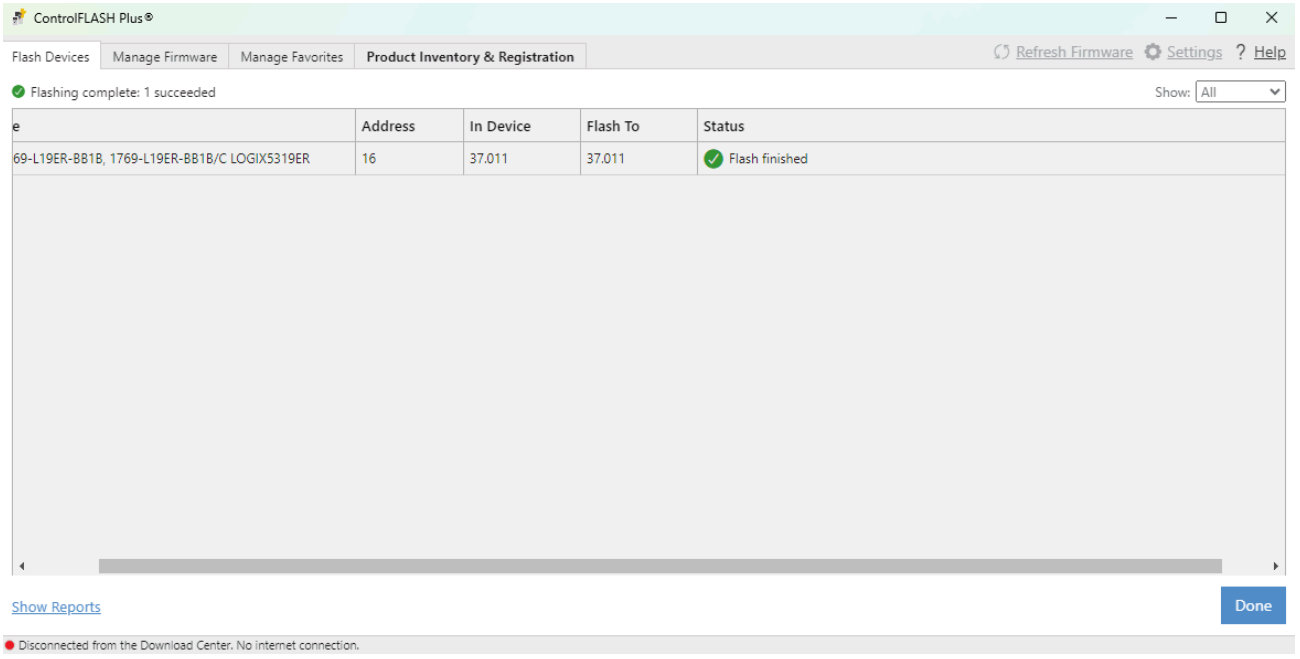

5. Select the device and latest firmware version, then click **Next**:




6. Click **Flash**:

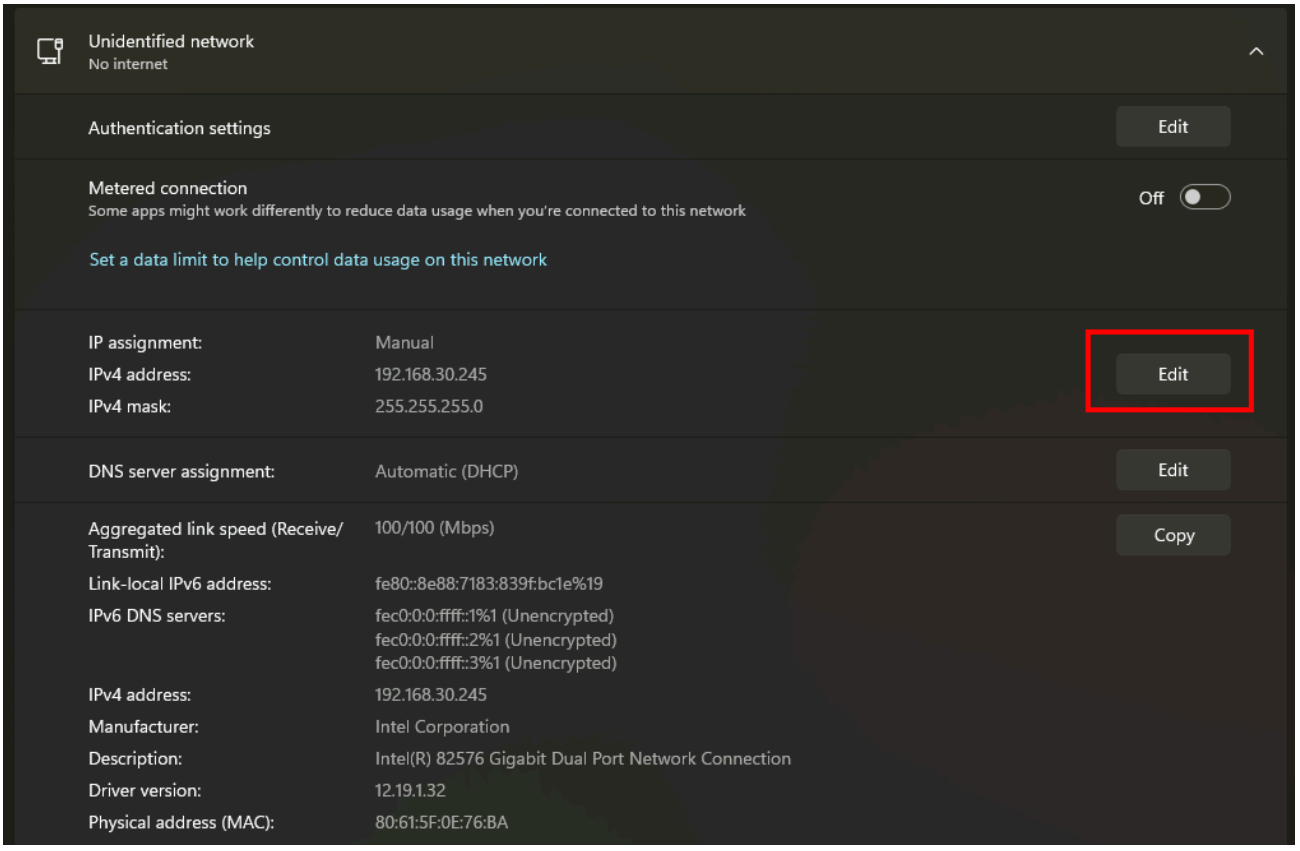



7. After flashing succeeds, reboot your PLC.



# Configuring your PC's network settings

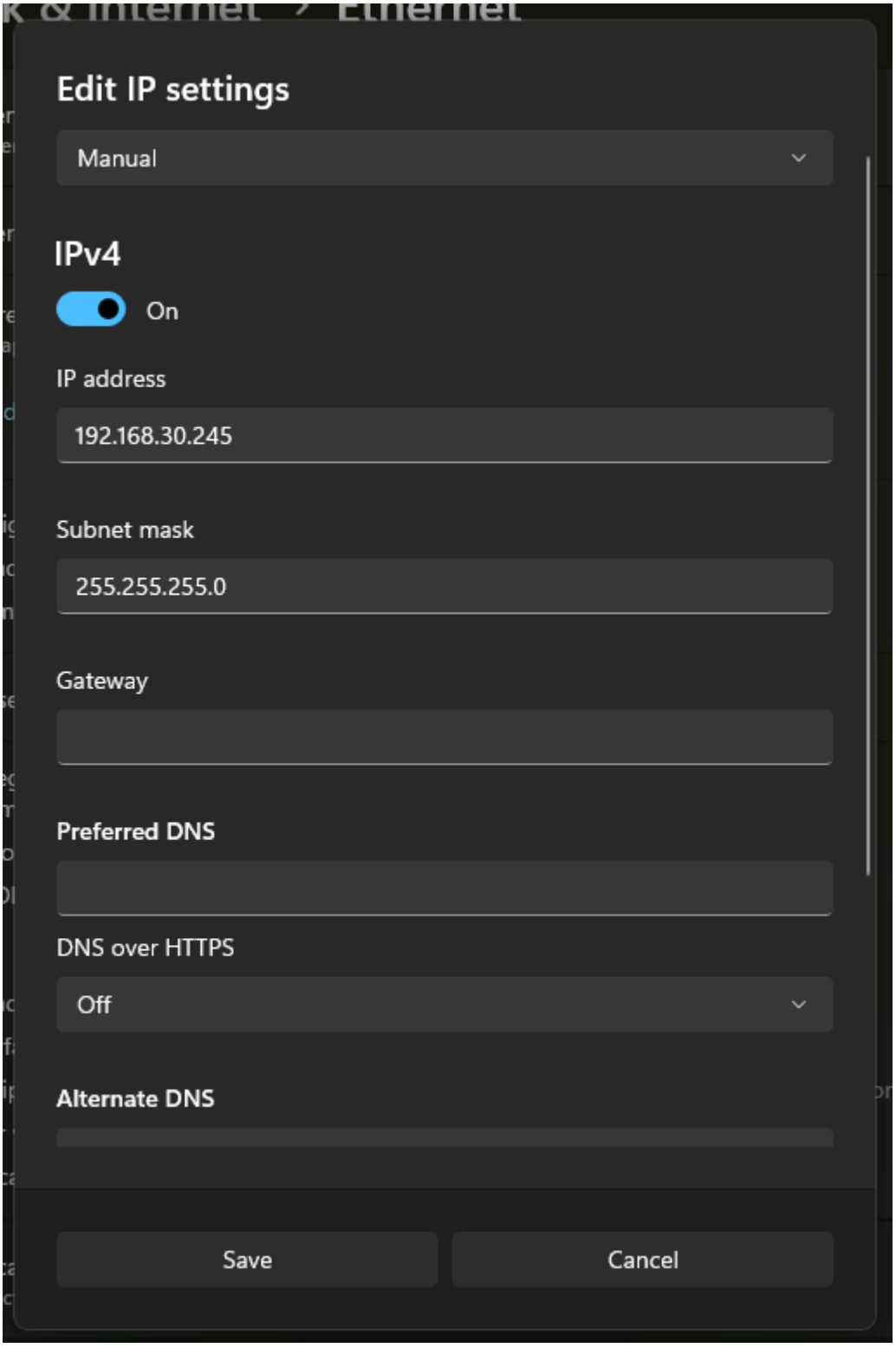
1. Open your PC's network settings and select edit on the Ethernet adapter connected to your PLC:



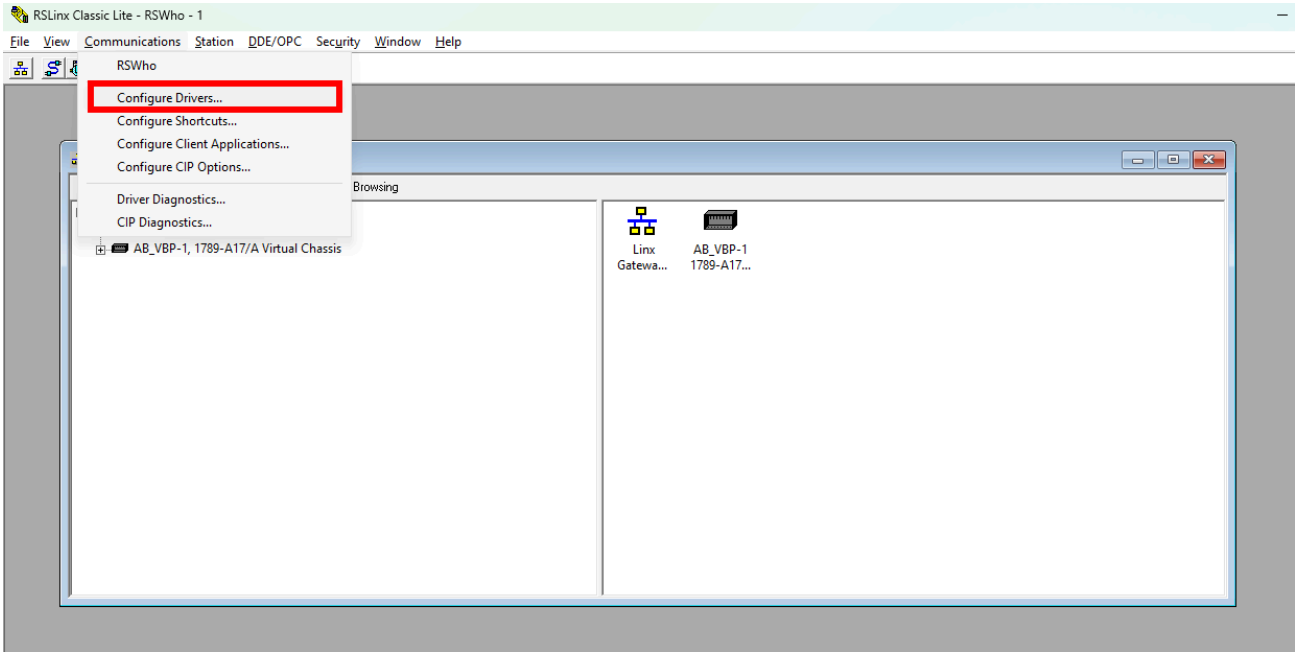

2. In the **Edit IP Settings** dialog, set:

   - **IP Address**: `192.168.30.245`
   - **Subnet Mask**: `255.255.255.0`

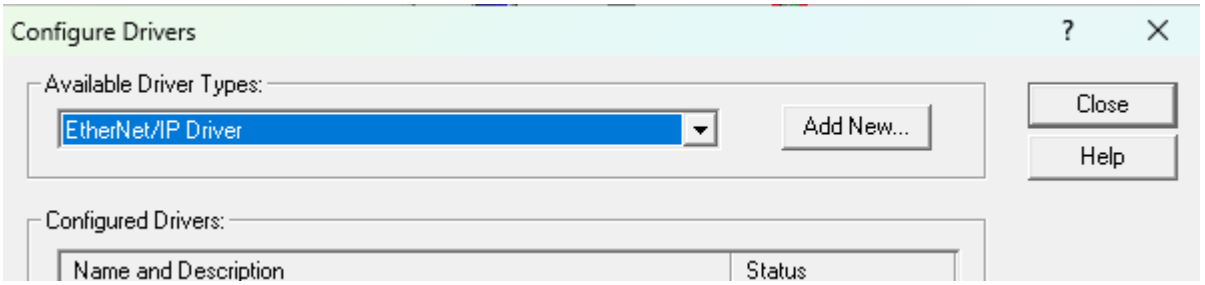
   Click **Save** to apply the settings.



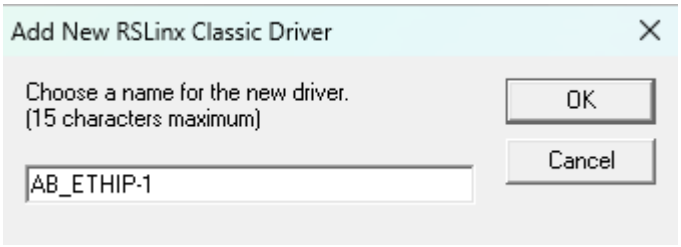

3. Open RSLogix Classic and select **Communications** > **Configure Drivers**:



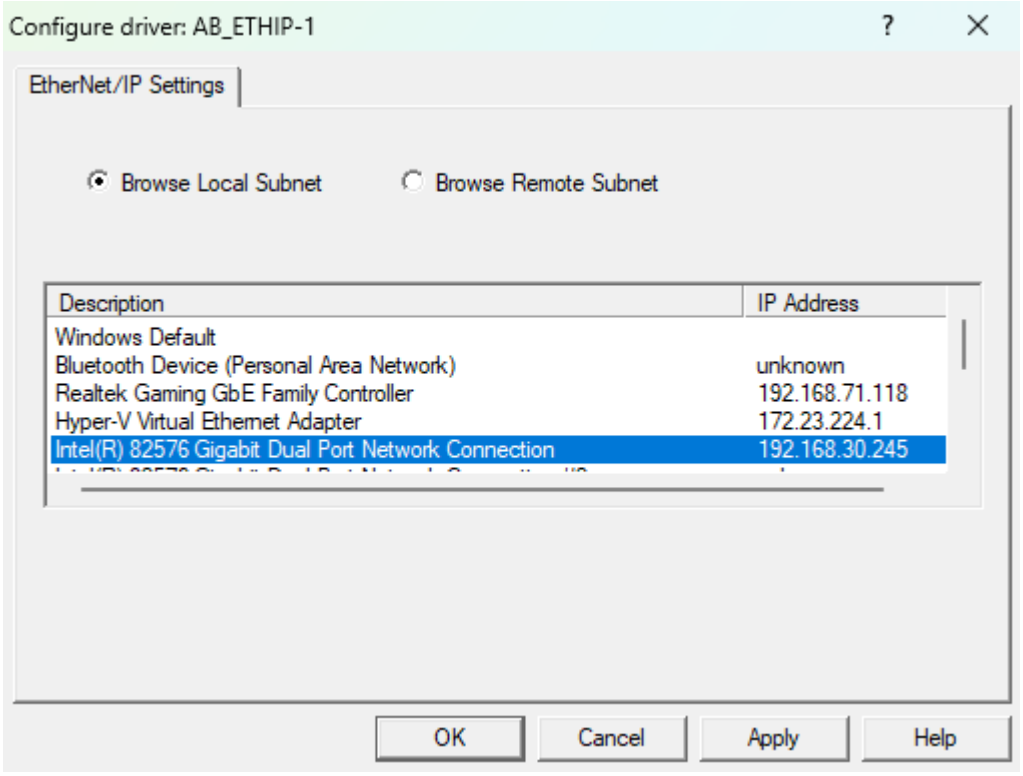

4. Select **EtherNet/IP Driver** and click **Add New**:




5. Click **OK** to add the driver:

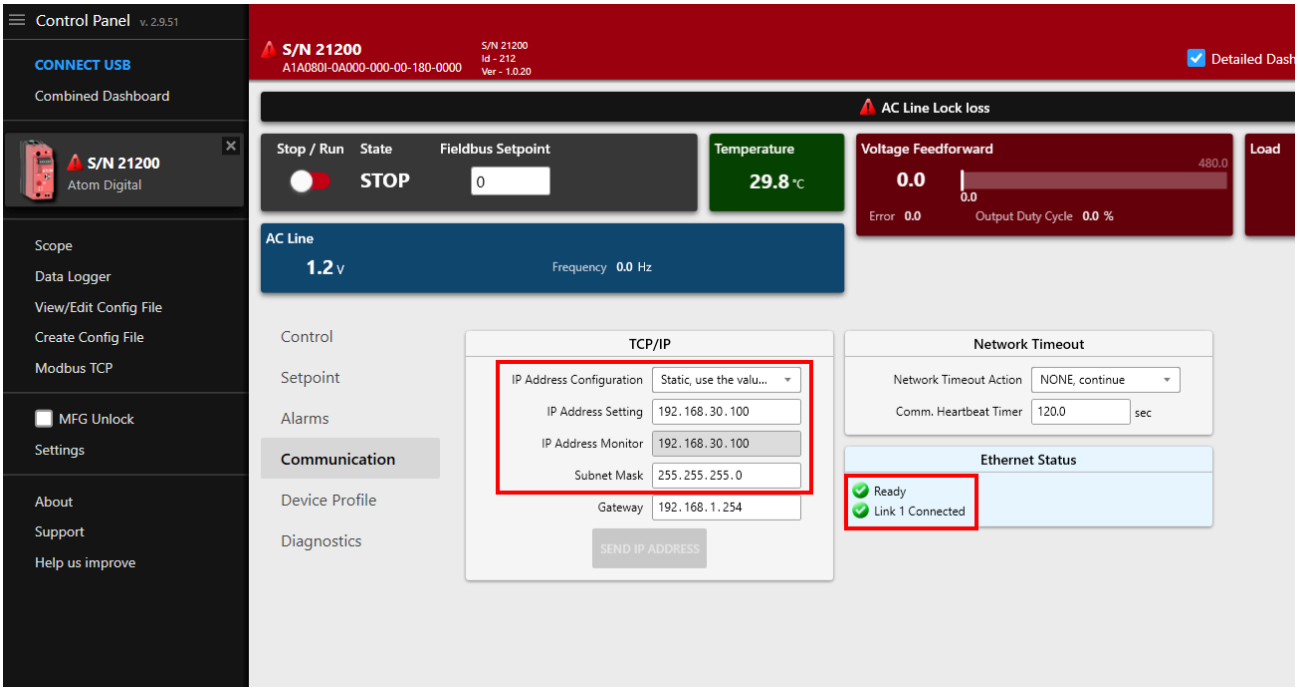



6. Select the adapter with IP address `192.168.30.245`, then hit **Apply** and **OK**:



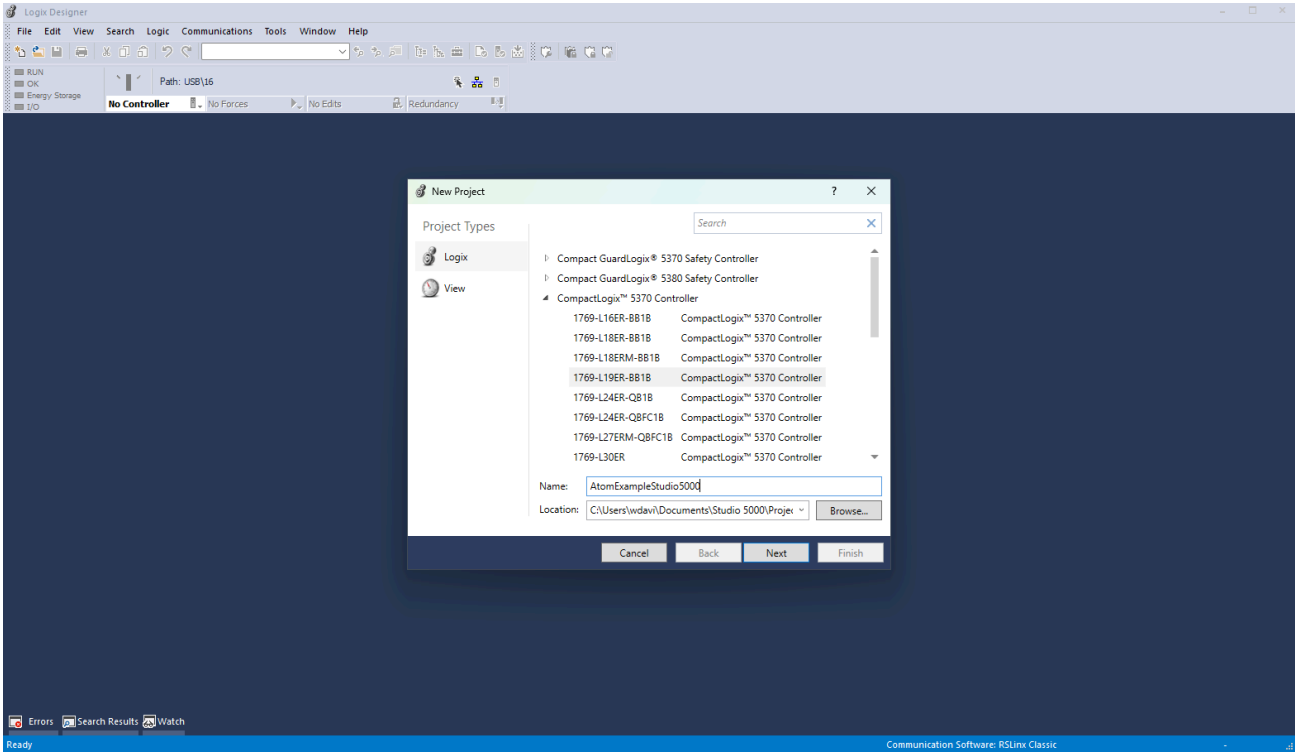

# ATOM Configuration

1. Connect ATOM to your PC using a USB-C cable. Launch Control Panel and connect to your ATOM. In the **Network** tab, set the following:
   - **IP Address Configuration**: `Static`
   - **IP Address**: `192.168.30.100`
   - **Subnet Mask**: `255.255.255.0`

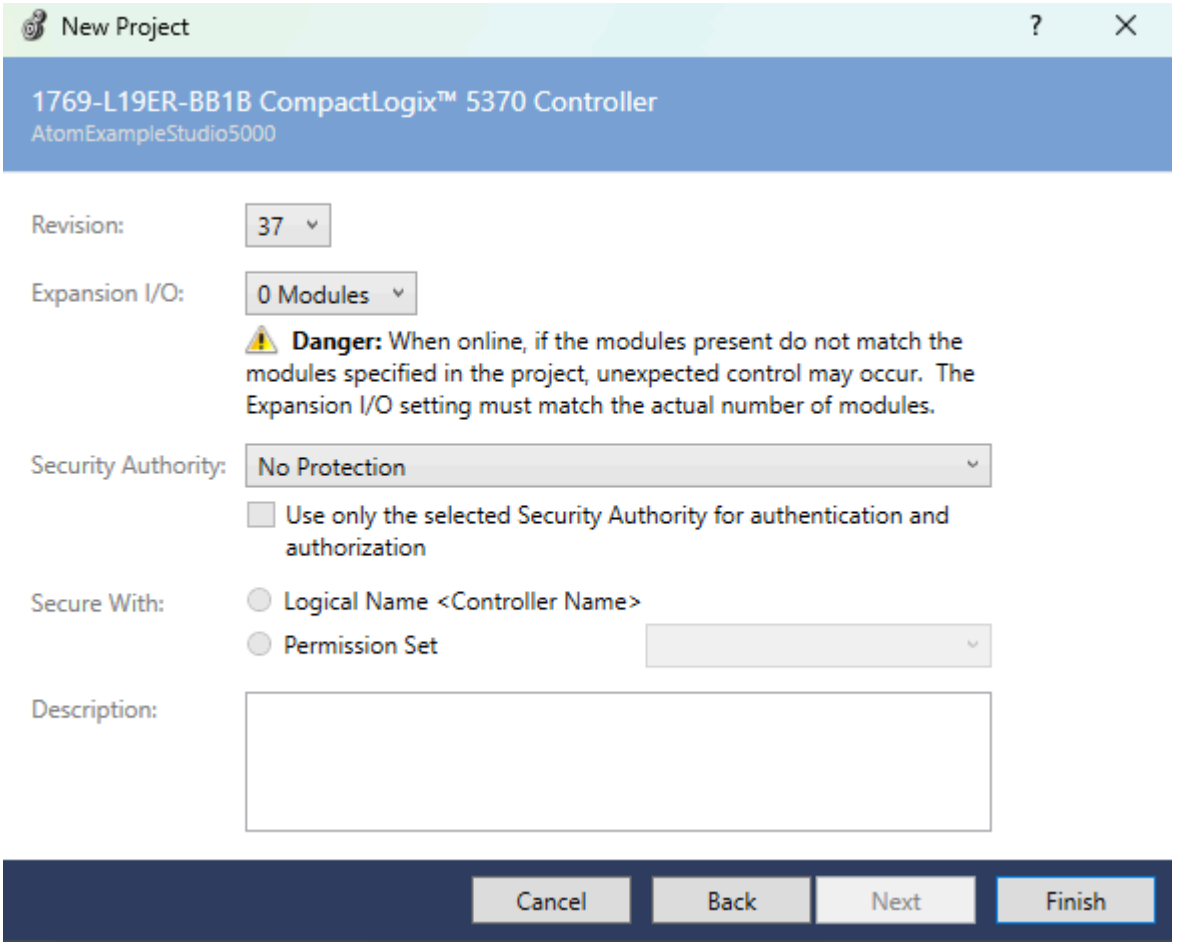



# Create a Studio 5000 project and connect to your PLC

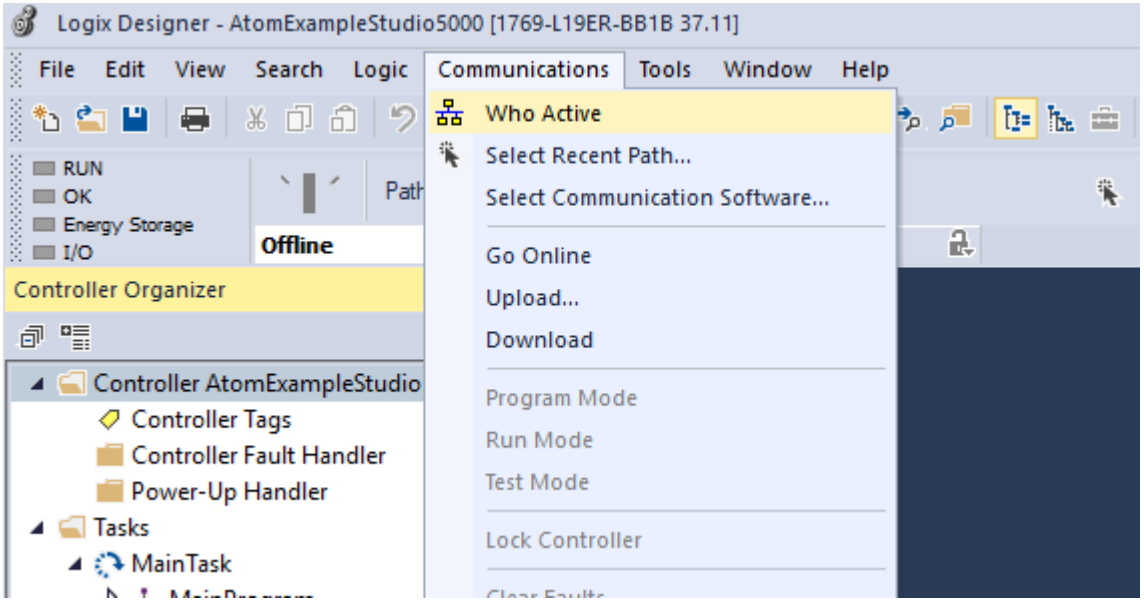
1. Launch Studio 5000, select **File** > **New Project**. Name the project `AtomExampleStudio5000` and select `1769-L19ER-BB1B` (CompactLogix 5370). Click **OK**:



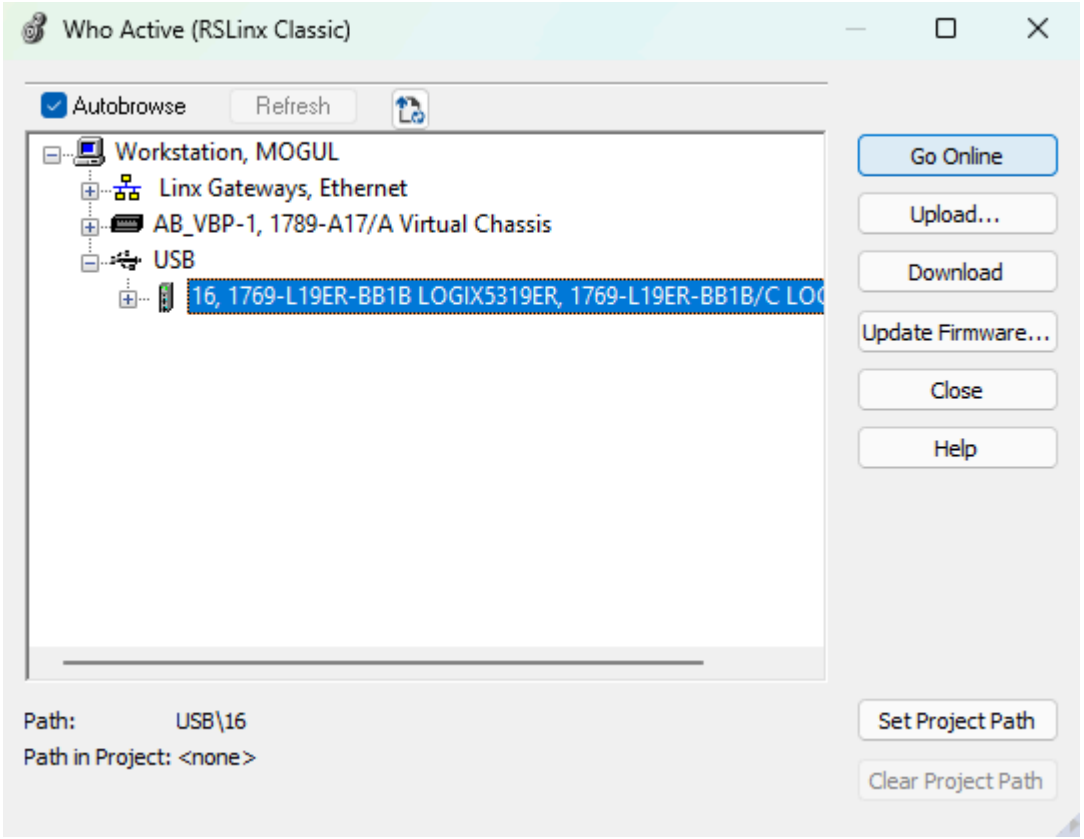

2. Select `0 Modules` under Expansion I/O, then click **Finish**:




3. Connect your PLC to your PC with a USB-B cable. In Studio 5000, select **Communications** > **Who Active**:

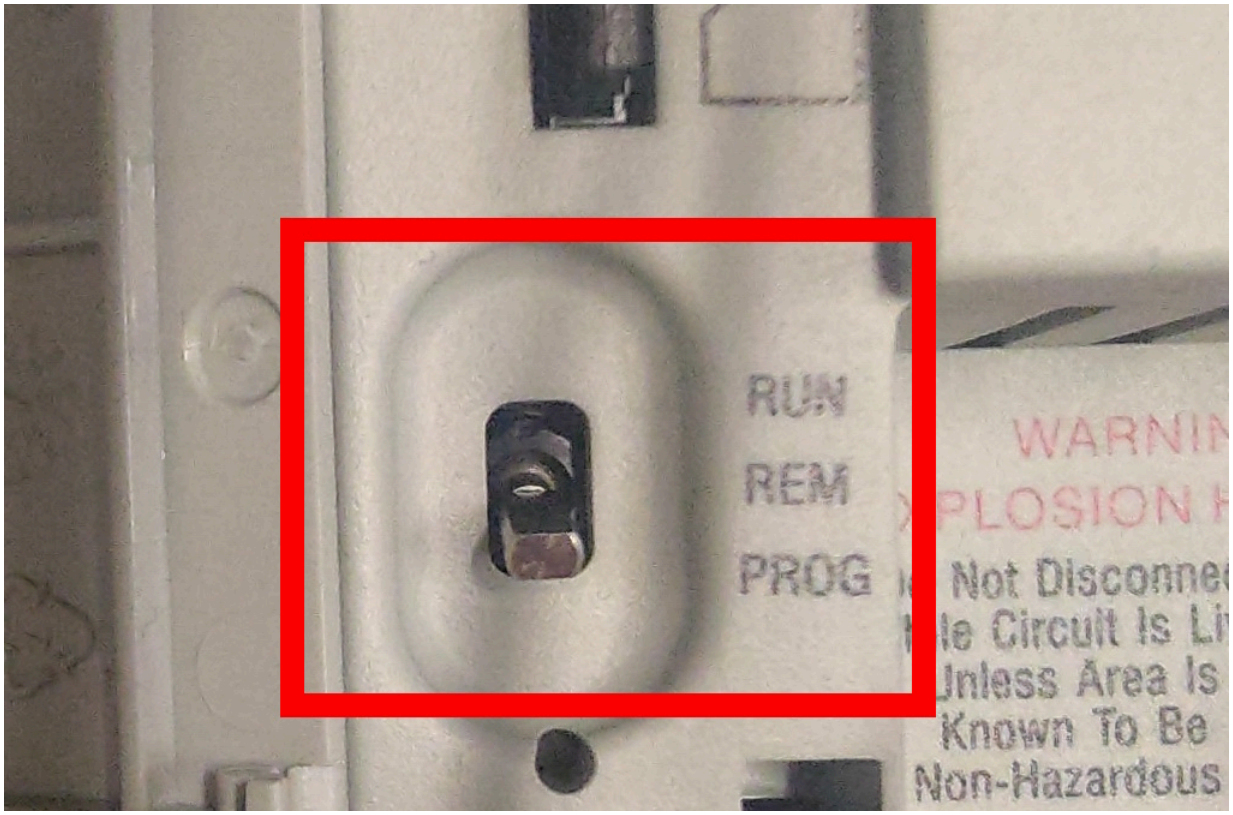



4. Select your PLC under the USB category and click **Go Online**:

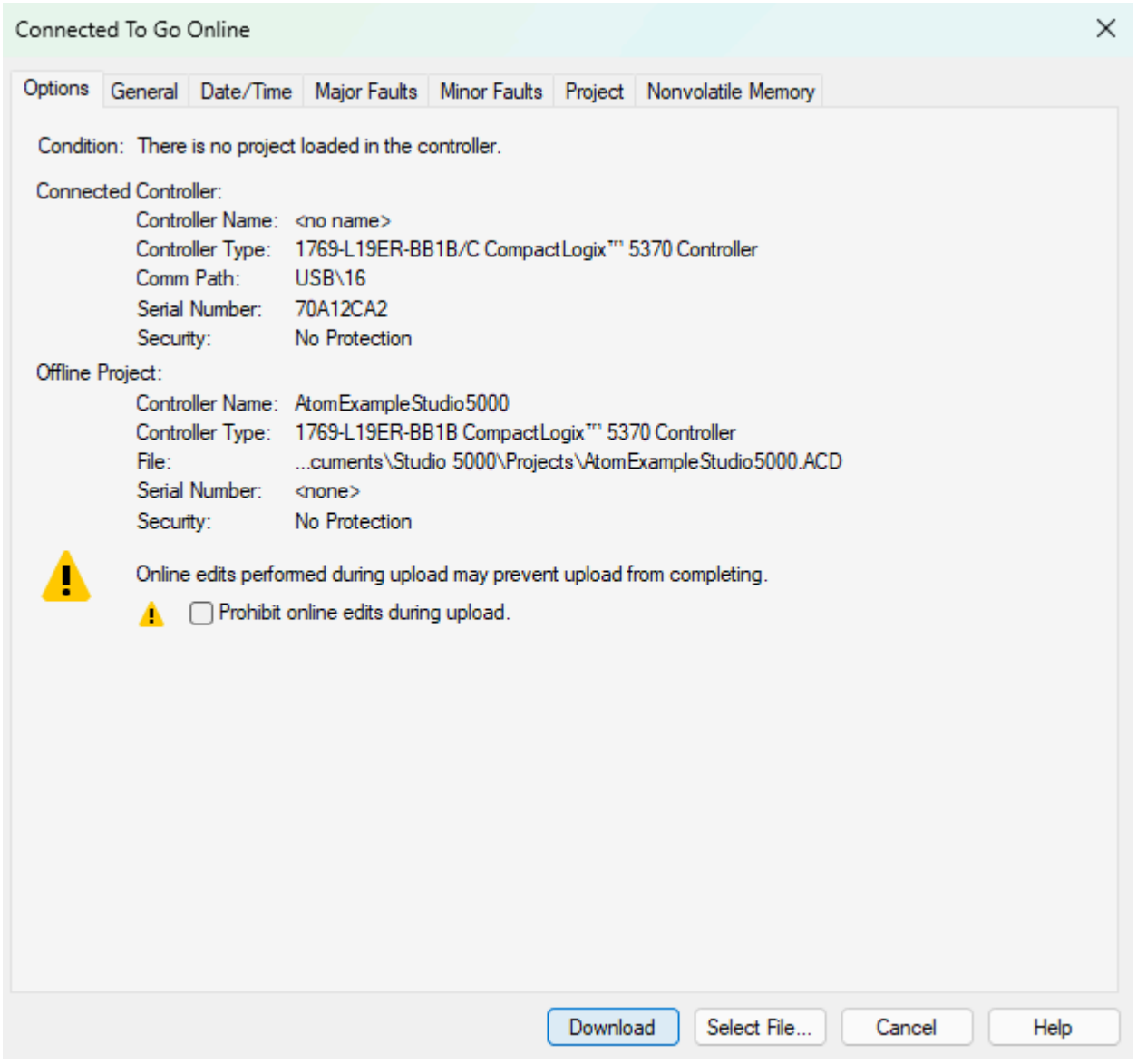


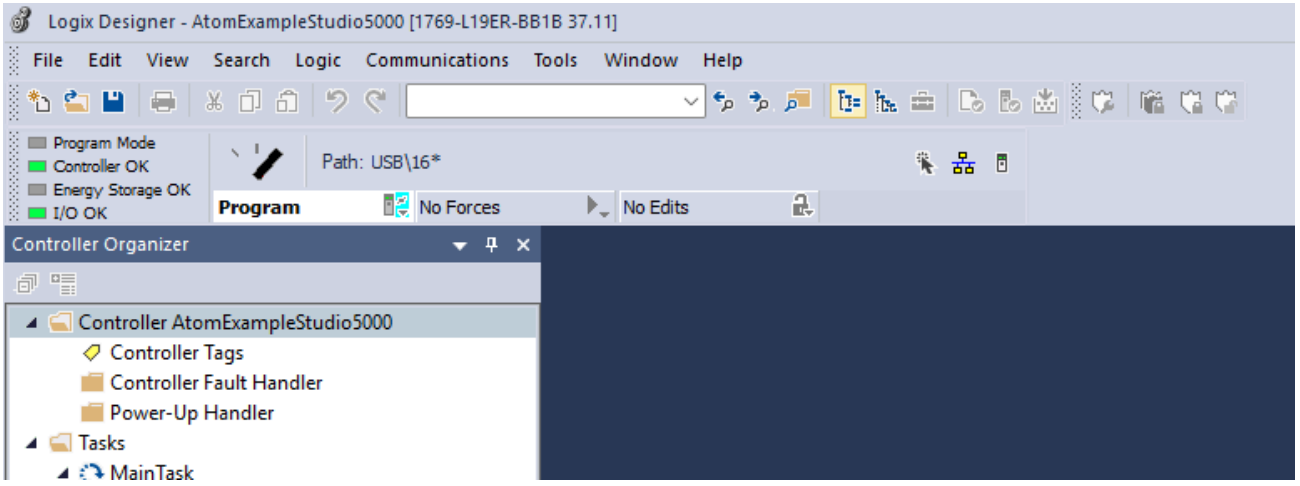
> Ensure the switch on your PLC is set to `PROG` mode before downloading.


Allen-Bradley
RUN
FORCE
I/O
OK
NS
LINK 1
LINK 2
SD

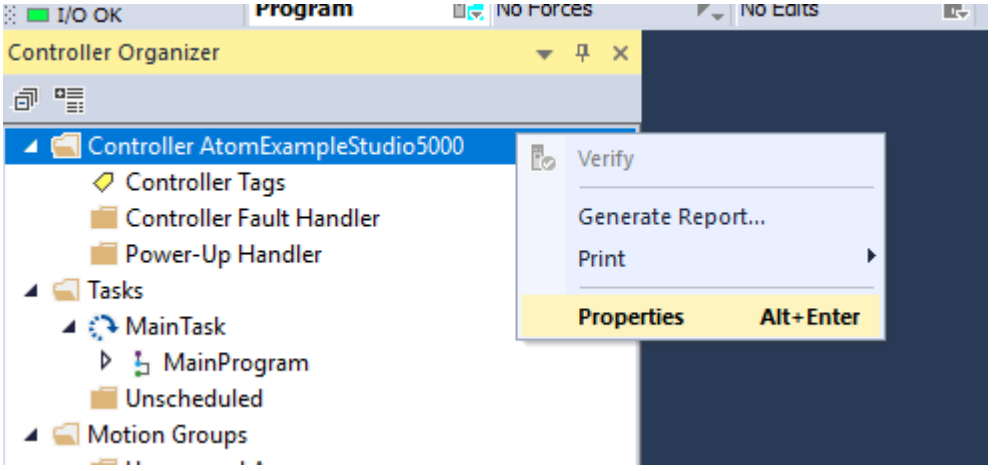



5. Select **Download** and double check that the `Controller OK` indicator light turns green:

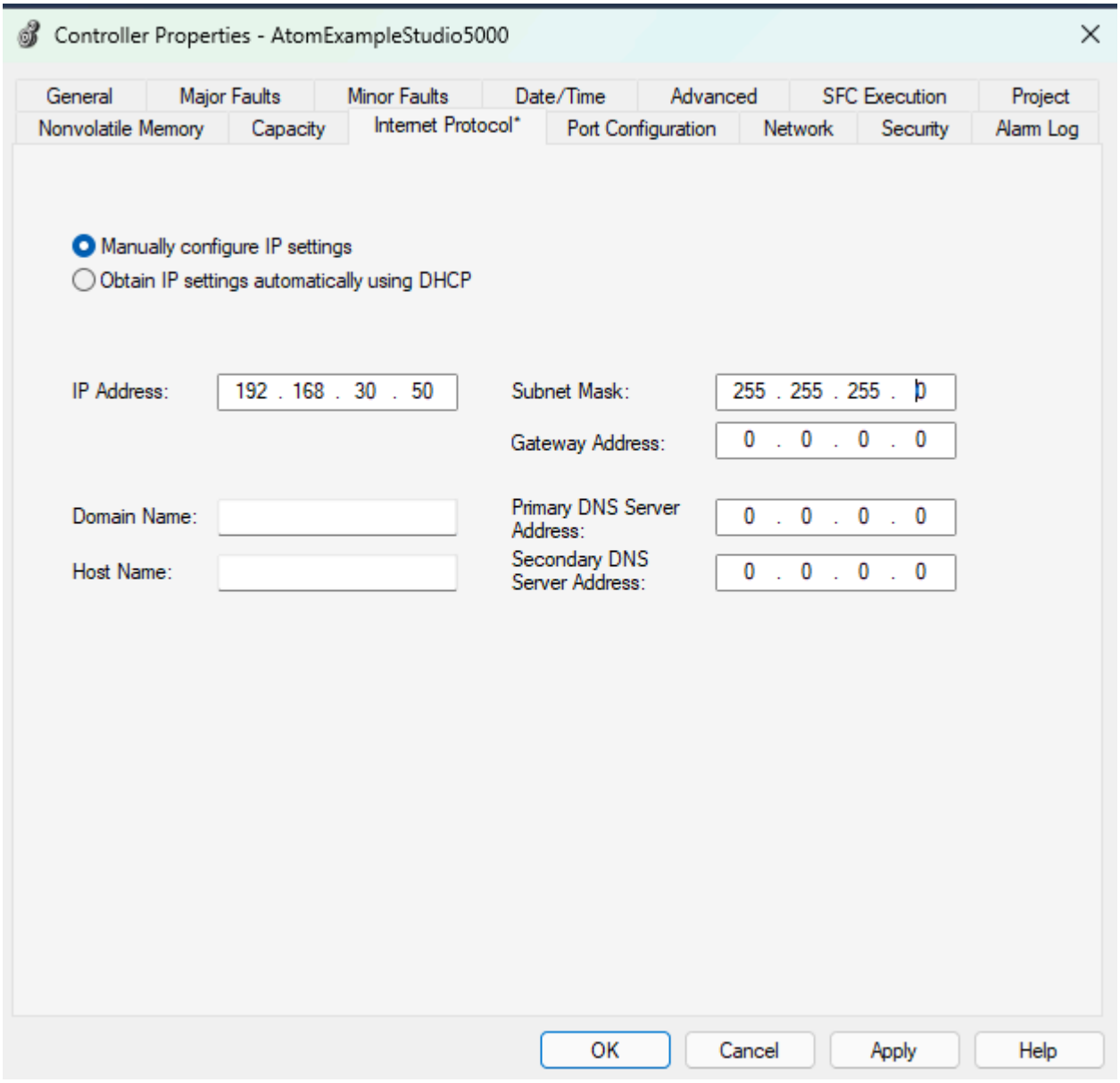

Connected To Go Online
Options
General
Date/Time
Major Faults
Minor Faults
Project
Nonvolatile Memory
Condition: There is no project loaded in the controller.
Connected Controller:
Controller Name: <no name>
Controller Type: 1769-L19ER-BB1B/C CompactLogix™ 5370 Controller
Comm Path: USB\16
Serial Number: 70A12CA2
Security: No Protection
Offline Project:
Controller Name: AtomExampleStudio5000
Controller Type: 1769-L19ER-BB1B CompactLogix™ 5370 Controller
File: ...cuments\Studio 5000\Projects\AtomExampleStudio5000.ACD
Serial Number: <none>
Security: No Protection
Online edits performed during upload may prevent upload from completing.
Prohibit online edits during upload.
Download
Select File...
Cancel
Help



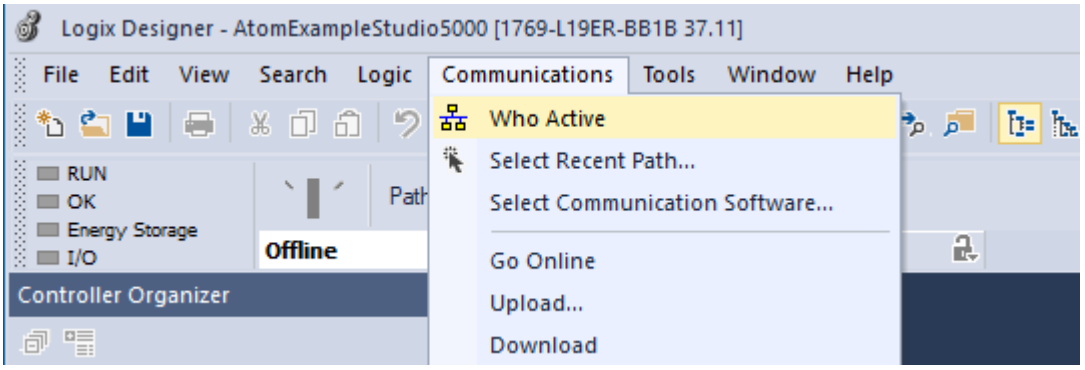

6. Right click `Controller AtomExampleStudio5000` and select **Properties**:

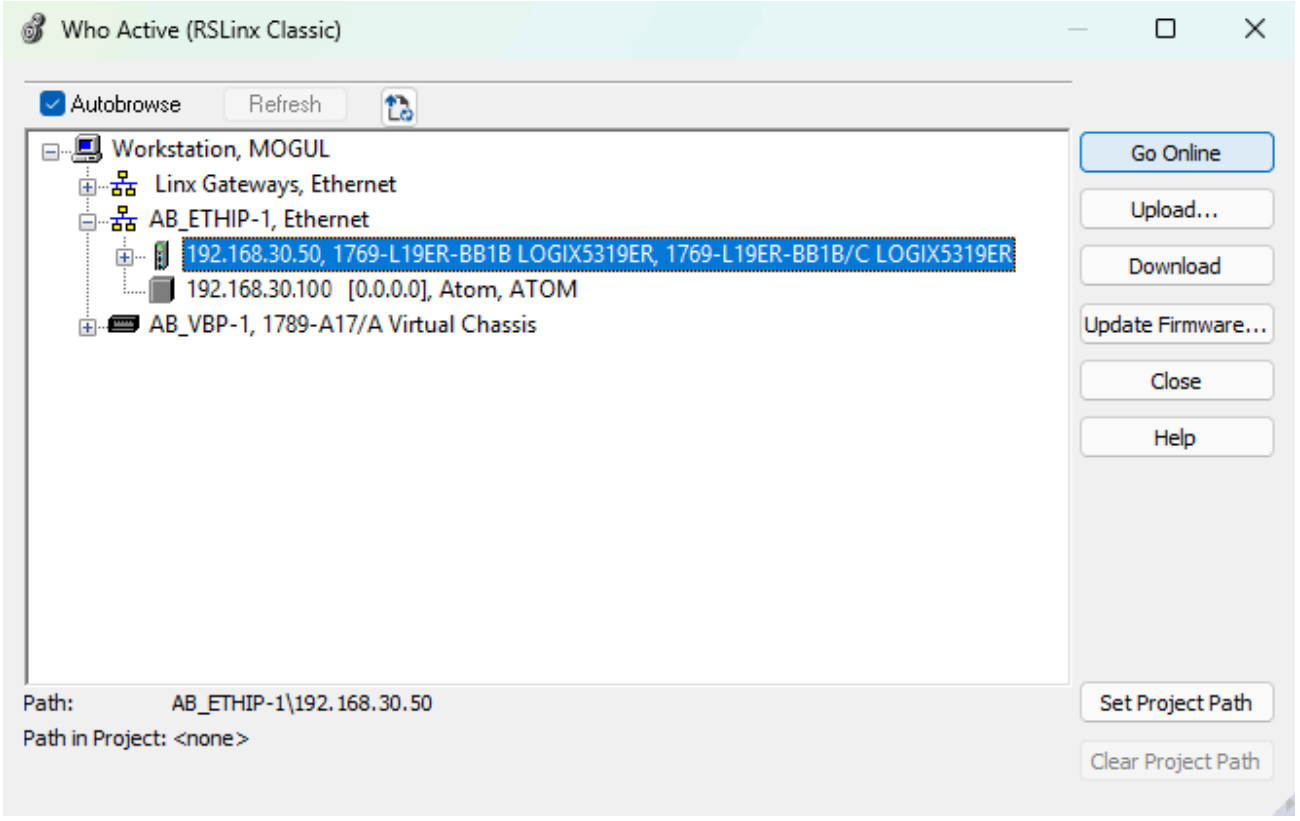



7. In the **Internet Protocol** tab, set the following and hit **Apply** and **OK**:
    - Manually configure IP settings checked
    - **IP Address**: `192.168.30.50`
    - **Subnet Mask**: `255.255.255.0`




8. Disconnect the USB cable from your PLC. In Studio 5000, select **Communications** > **Who Active** again, then select your PLC under the Ethernet category and click **Go Online**:

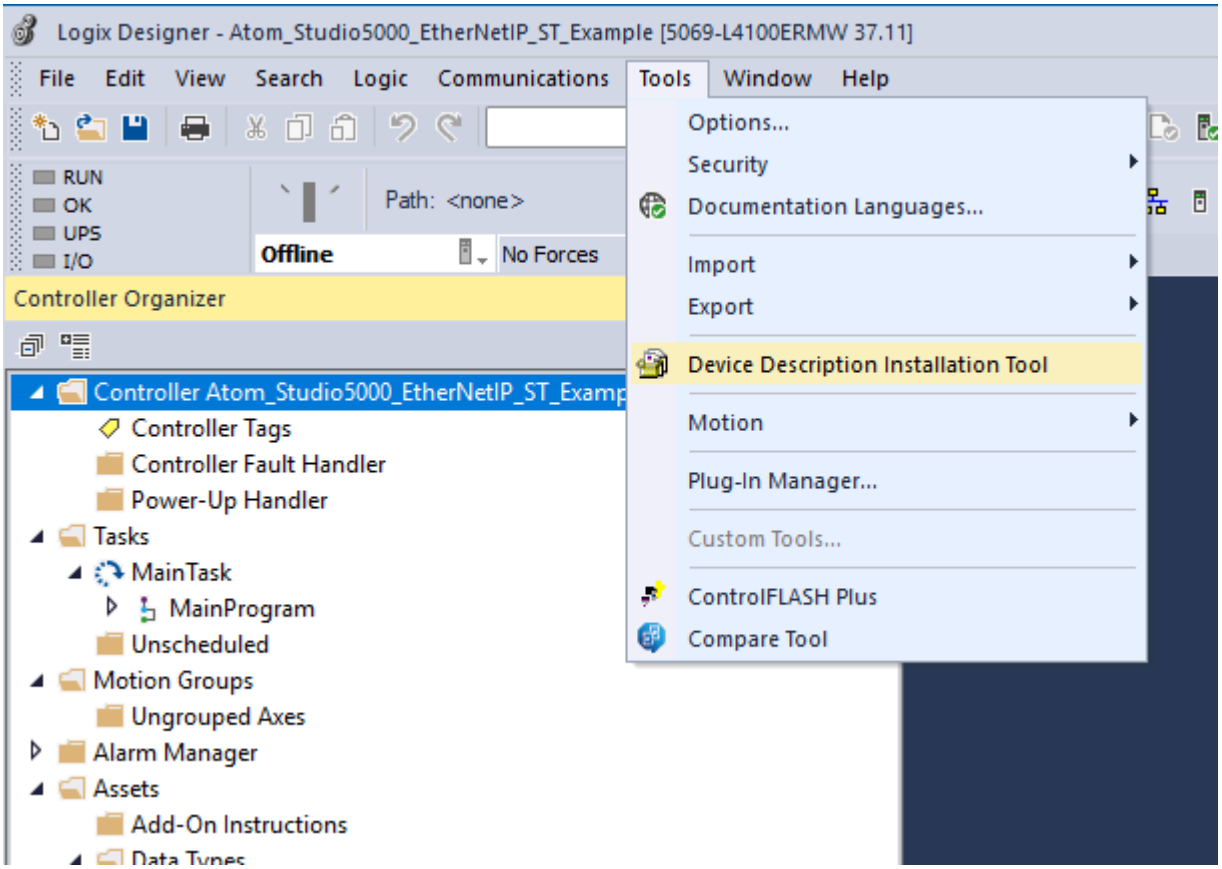


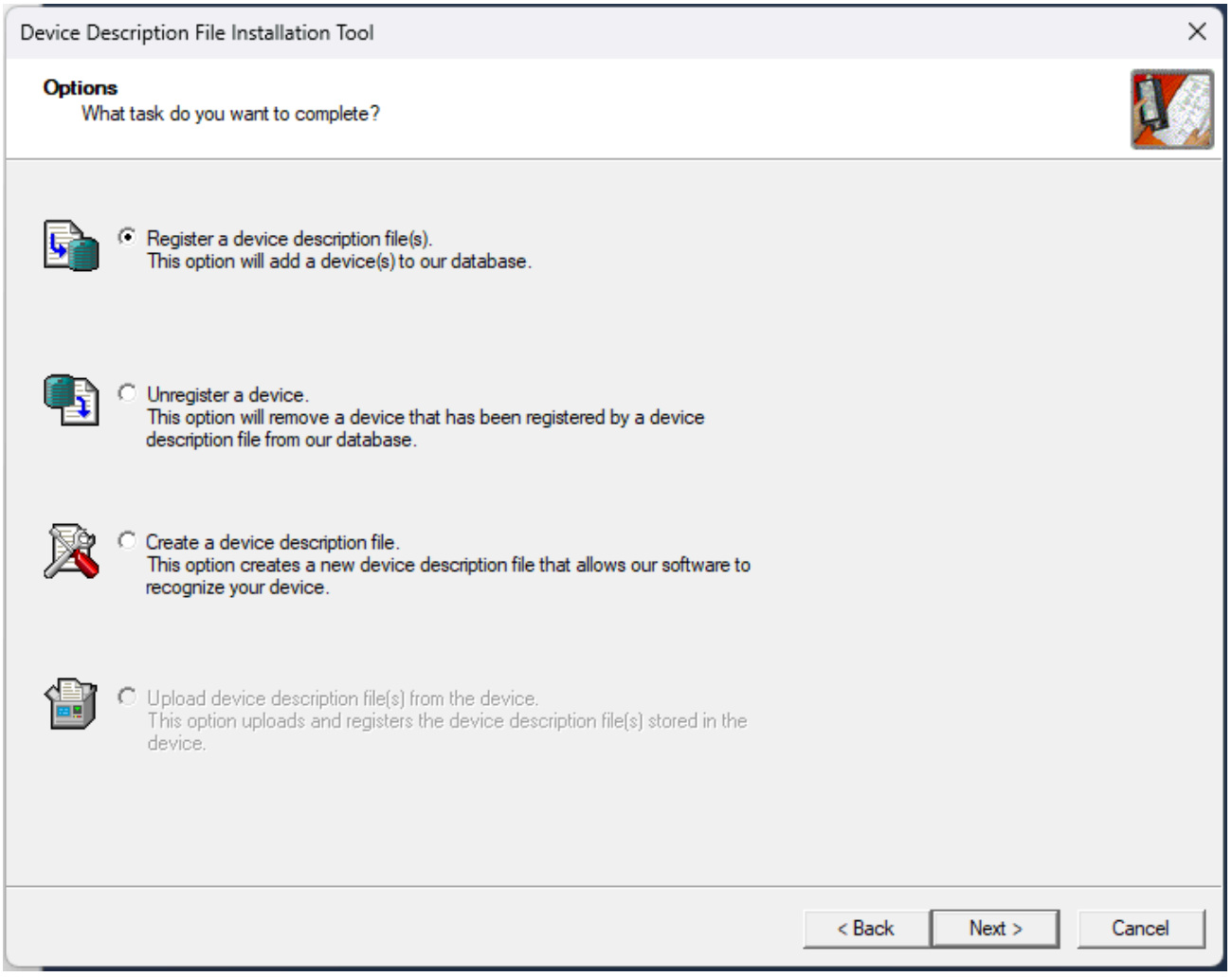



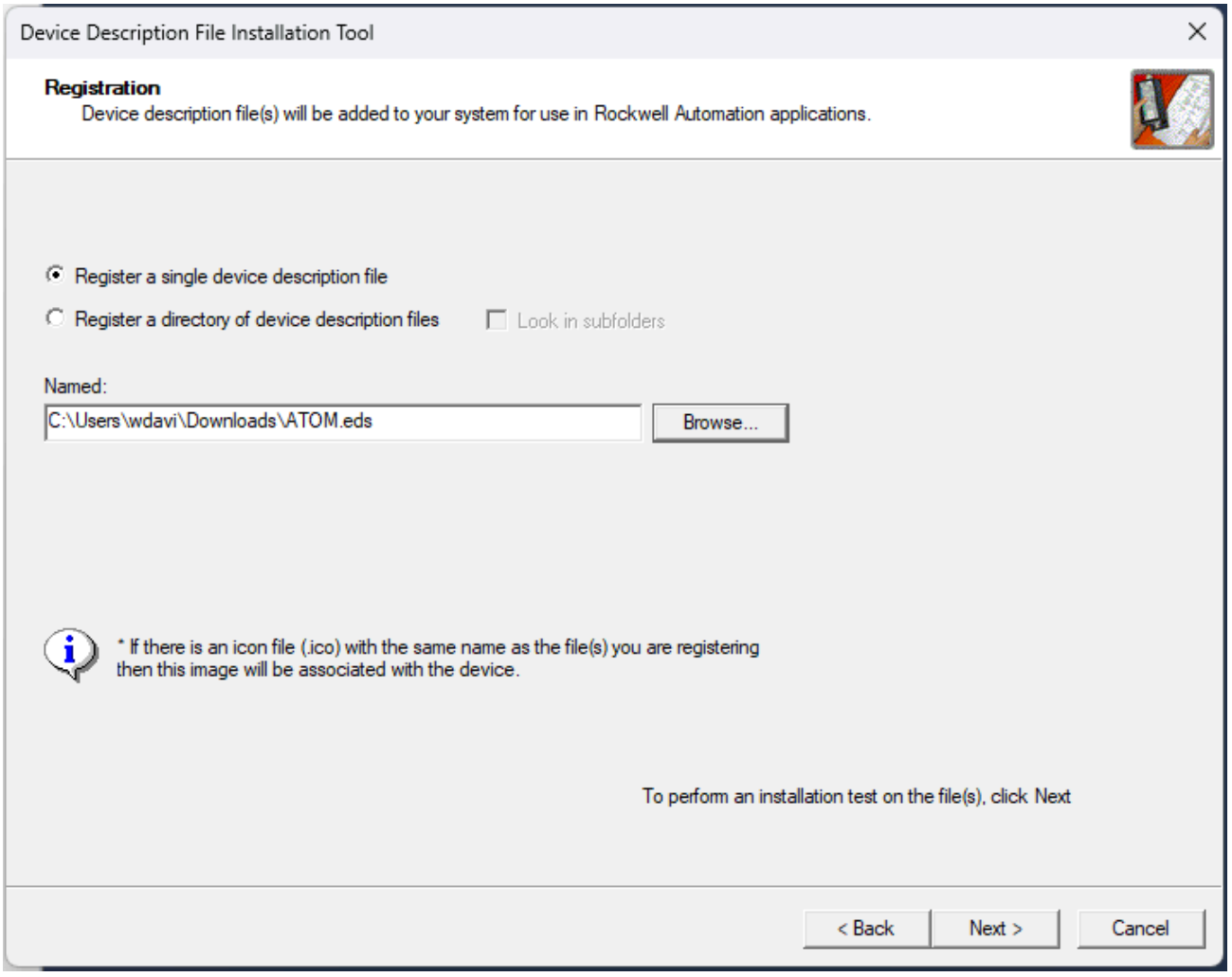

> You should also see ATOM (with IP address `192.168.30.100`) under the `AB_ETHIP-1` category.

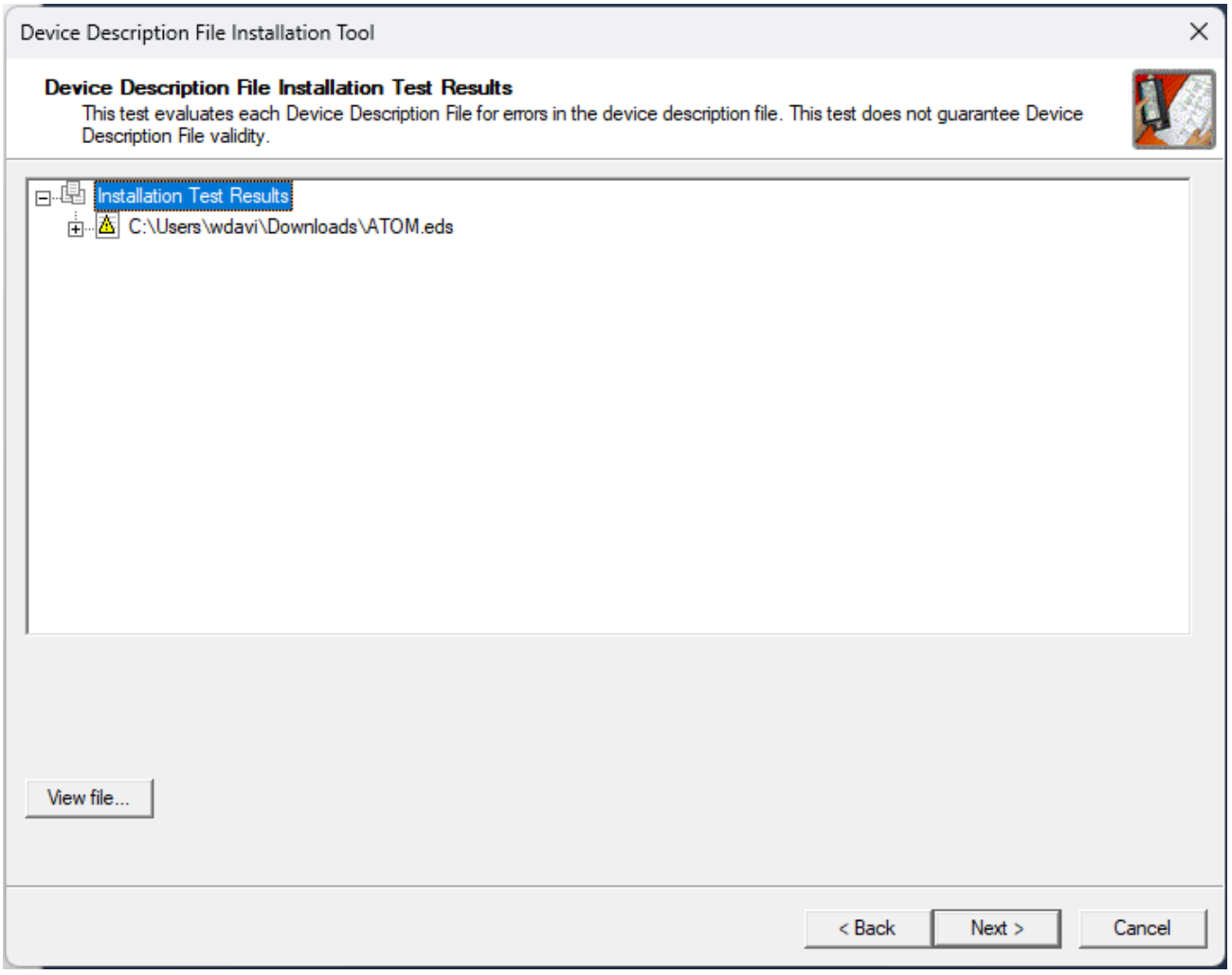
# Import EDS file

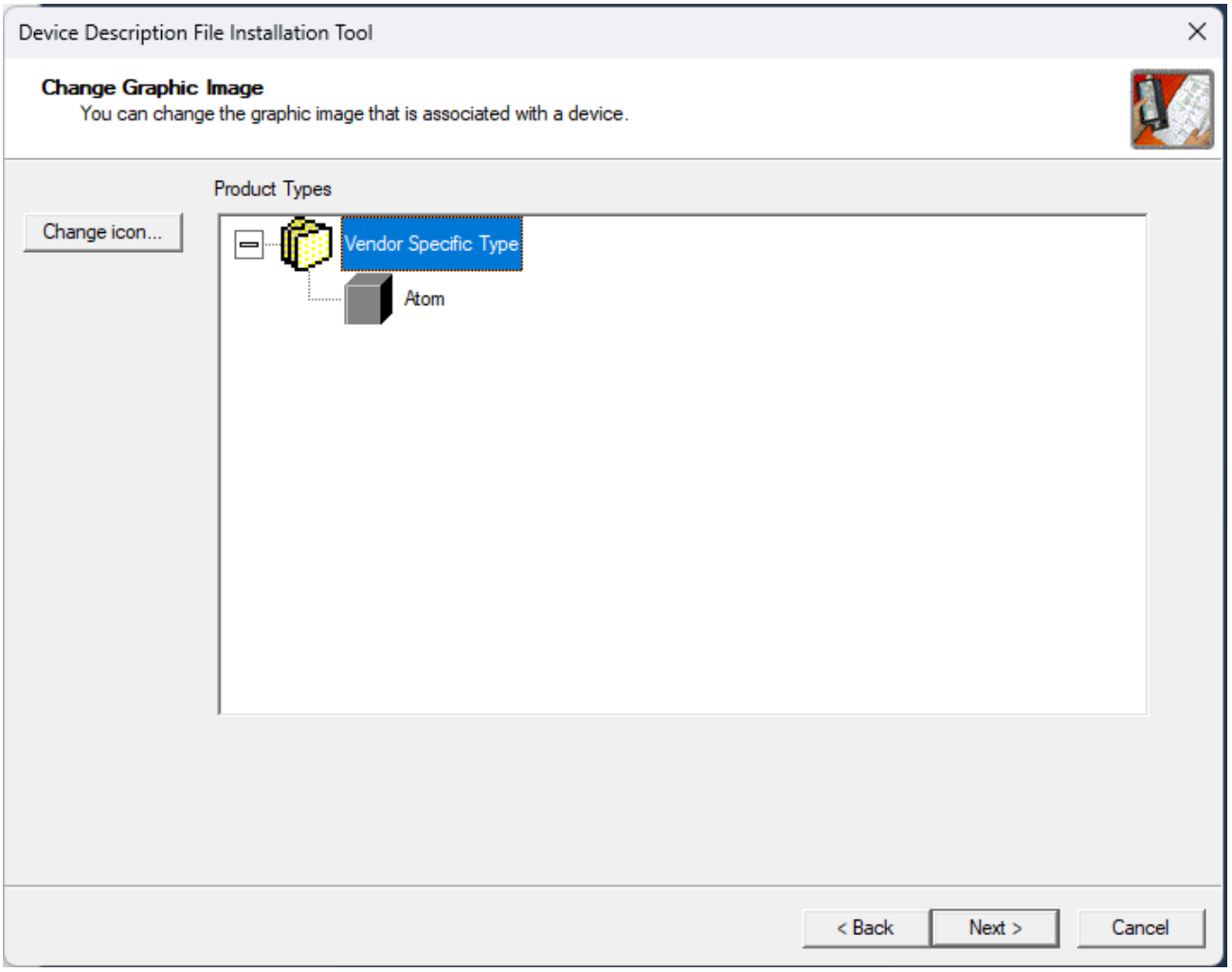
Select **Tools** > **Device Description Installation Tool** (some versions call it **EDS Hardware Installation Tool**)

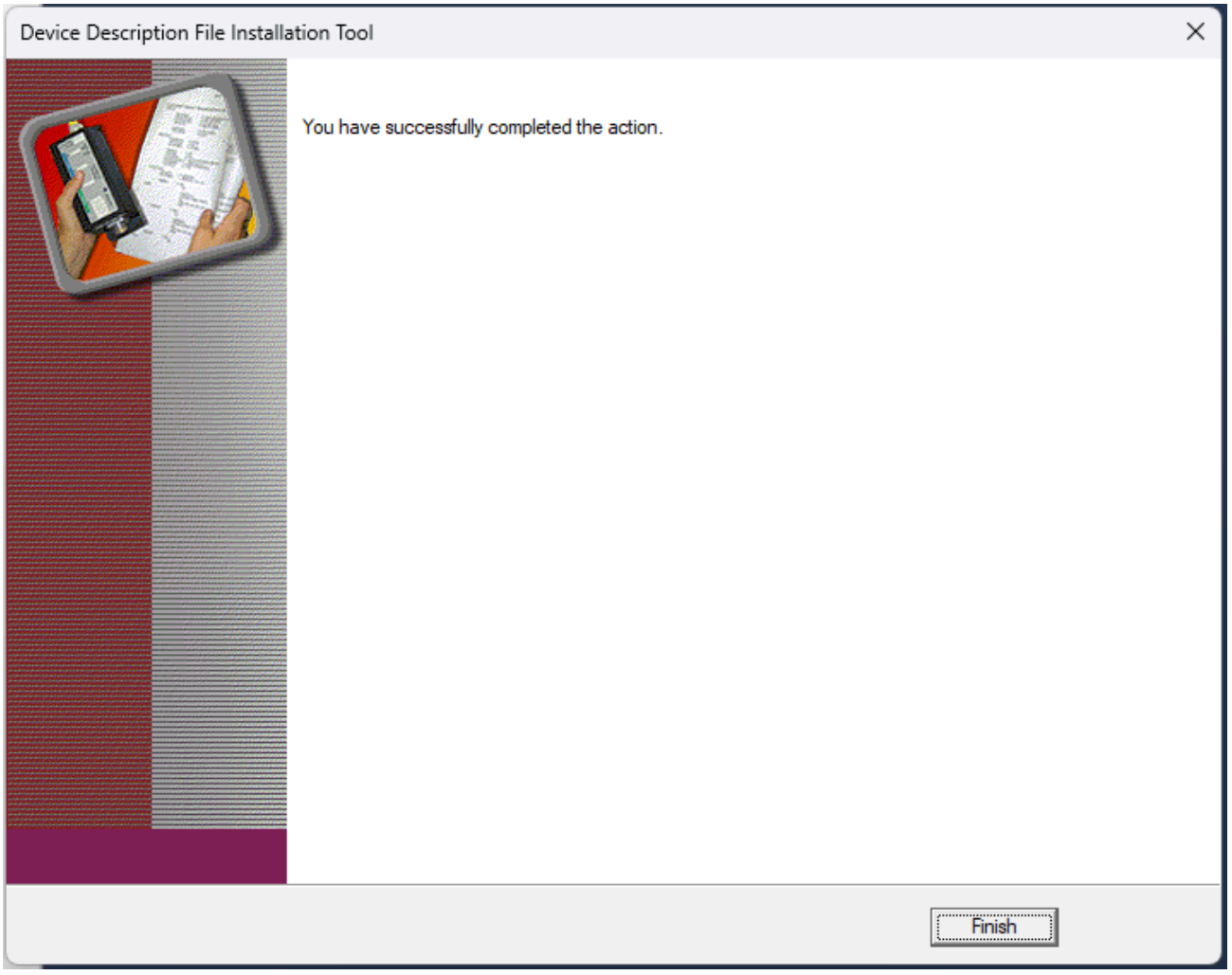

Logix Designer - Atom_Studio5000_EtherNetIP_ST_Example [5069-L4100ERMW 37.11]
File
Edit
View
Search
Logic
Communications
Tools
Window
Help
RUN
OK
UPS
I/O
Path: <none>
Offline
No Forces
Controller Organizer
Controller Atom_Studio5000_EtherNetIP_ST_Examp
Controller Tags
Controller Fault Handler
Power-Up Handler
Tasks
MainTask
MainProgram
Unscheduled
Motion Groups
Ungrouped Axes
Alarm Manager
Assets
Add-On Instructions
Data Types
Options...
Security
Documentation Languages...
Import
Export
Device Description Installation Tool
Motion
Plug-In Manager...
Custom Tools...
ControlFLASH Plus
Compare Tool


Device Description File Installation Tool
Options
What task do you want to complete?
Register a device description file(s).
This option will add a device(s) to our database.
Unregister a device.
This option will remove a device that has been registered by a device description file from our database.
Create a device description file.
This option creates a new device description file that allows our software to recognize your device.
Upload device description file(s) from the device.
This option uploads and registers the device description file(s) stored in the device.
< Back
Next >
Cancel


Device Description File Installation Tool
Registration
Device description file(s) will be added to your system for use in Rockwell Automation applications.
Register a single device description file
Register a directory of device description files
Look in subfolders
Named:
C:\Users\wdavi\Downloads\ATOM.eds
Browse...
* If there is an icon file (.ico) with the same name as the file(s) you are registering then this image will be associated with the device.
To perform an installation test on the file(s), click Next
< Back
Next >
Cancel

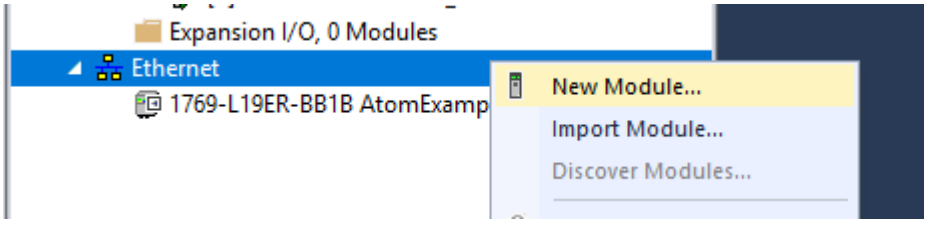

Device Description File Installation Tool
Device Description File Installation Test Results
This test evaluates each Device Description File for errors in the device description file. This test does not guarantee Device Description File validity.
Installation Test Results
C:\Users\wdavi\Downloads\ATOM.eds
View file...
< Back
Next >
Cancel


Device Description File Installation Tool
Change Graphic Image
You can change the graphic image that is associated with a device.
Change icon...
Product Types
Vendor Specific Type
Atom
< Back
Next >
Cancel

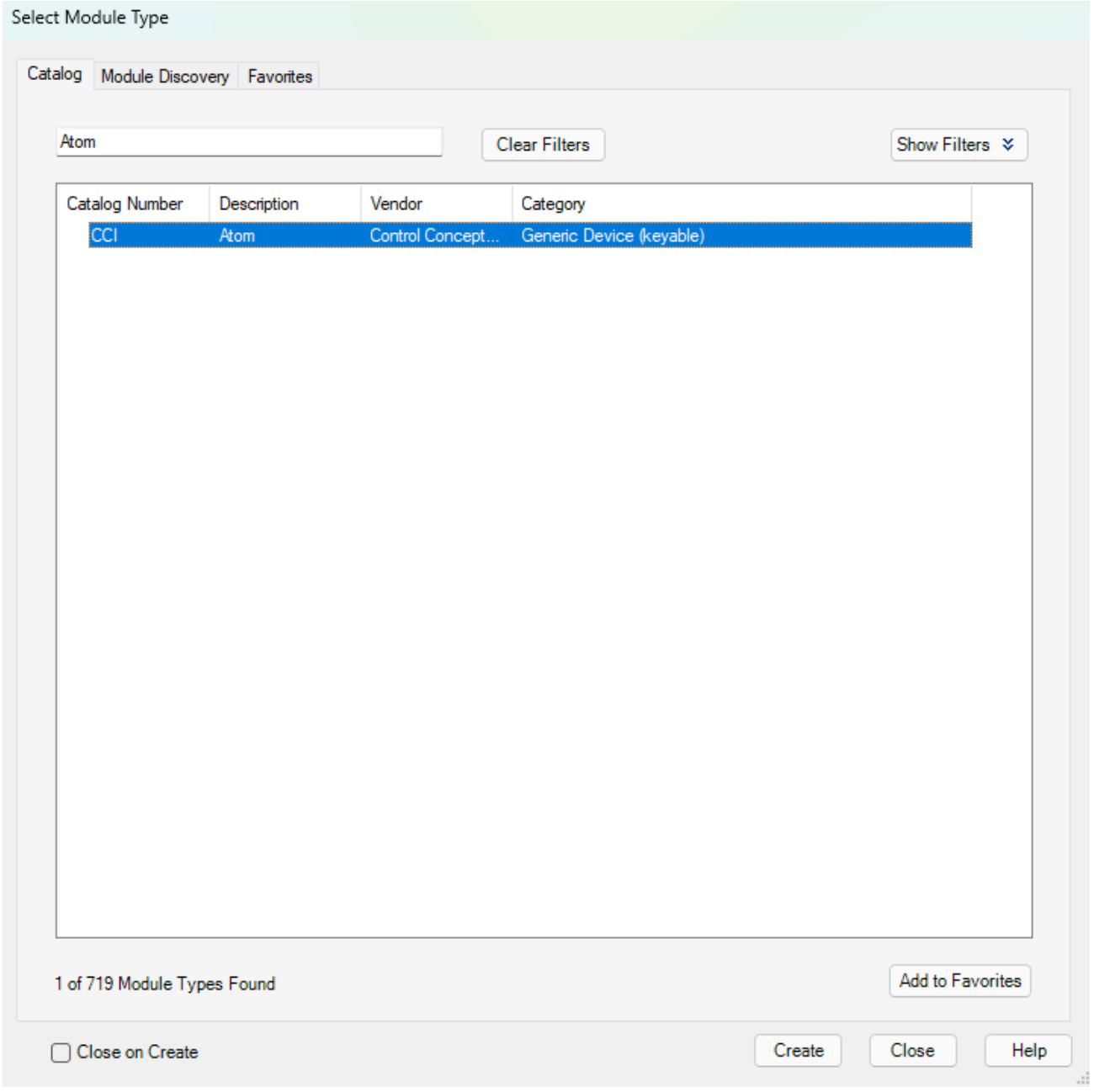



# Add Atom to the project

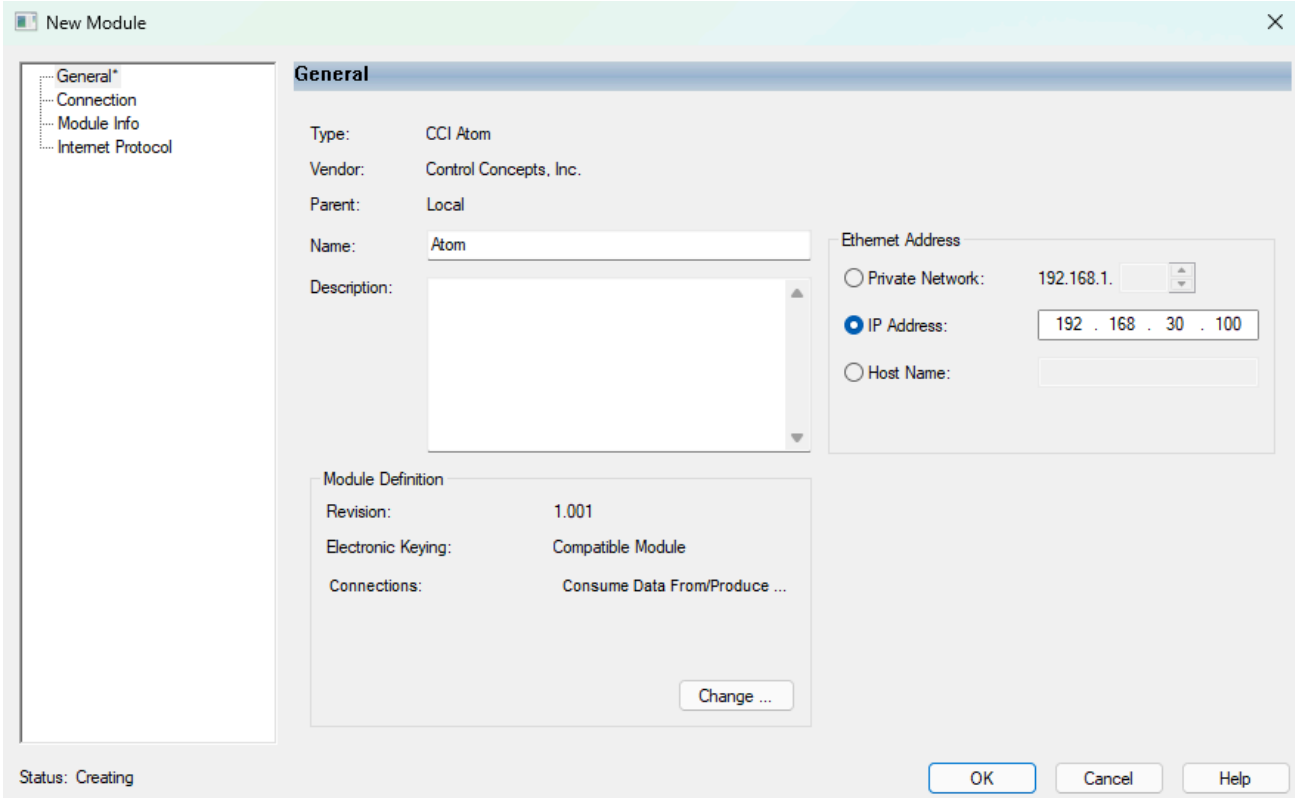
1. Right-click **Ethernet** and select **New Module**:




2. In the **Catalog** tab, search for `Atom`, select it, and click **Create**:

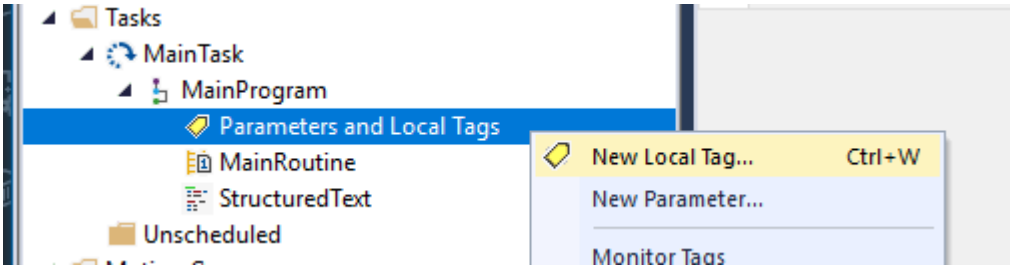



3. In the **General** tab, set the **IP Address** to `192.168.30.100` and click **OK**:



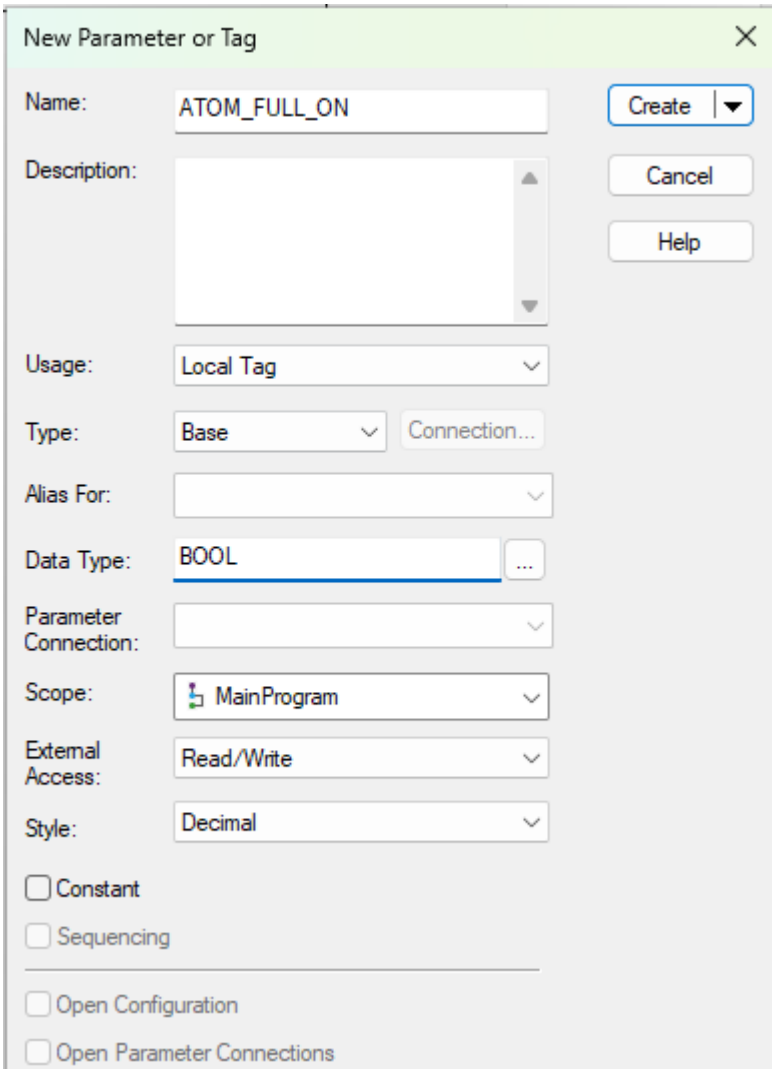

# A basic example program

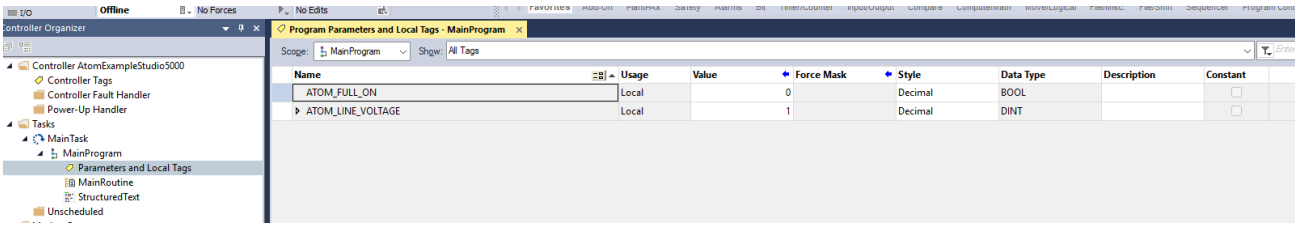
1. Right-click **Parameters and Local Tags** under `MainProgram` and select **New Tag** to create a tag:




2. Create *two*_ new tags:
    - Tag 1
        - **Name**: `ATOM_FULL_ON`
        - **Data Type**: `BOOL`

- Tag 2
  - **Name**: `ATOM_LINE_VOLTAGE`
  - **Data Type**: `DINT`

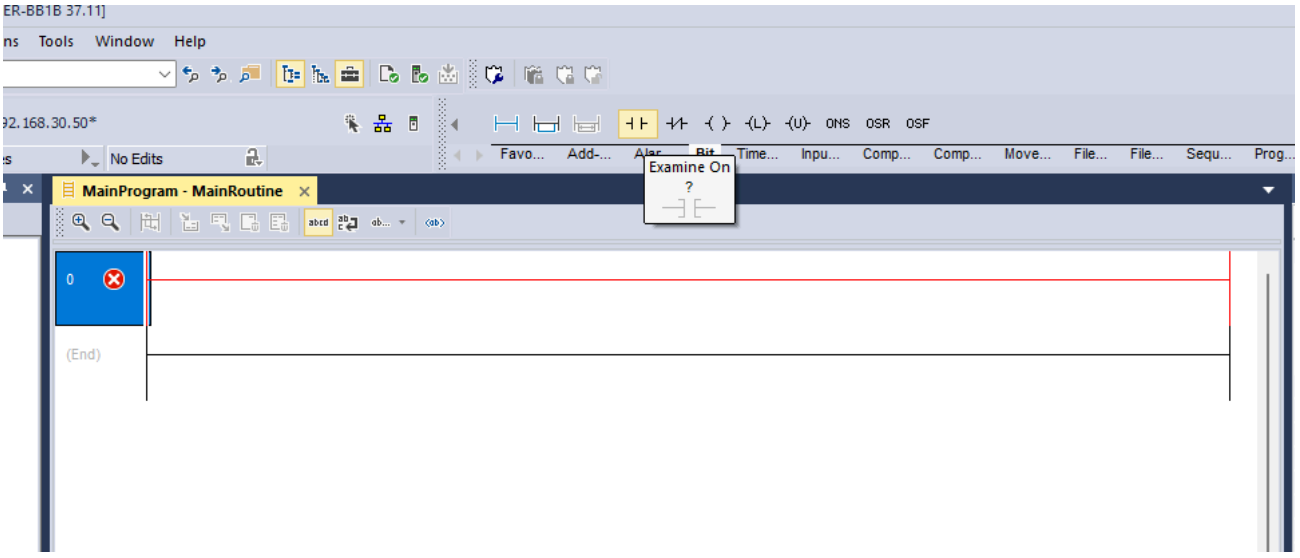


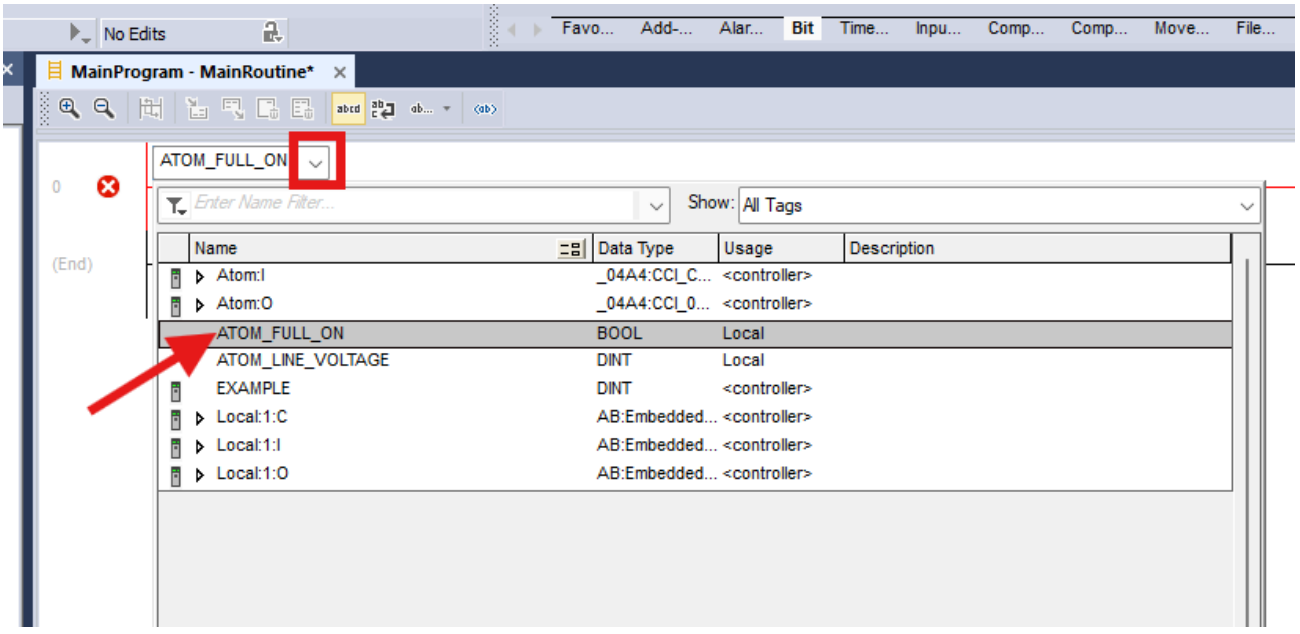

You can follow along with either the Structured Text or Ladder Logic examples below.

# Ladder Logic

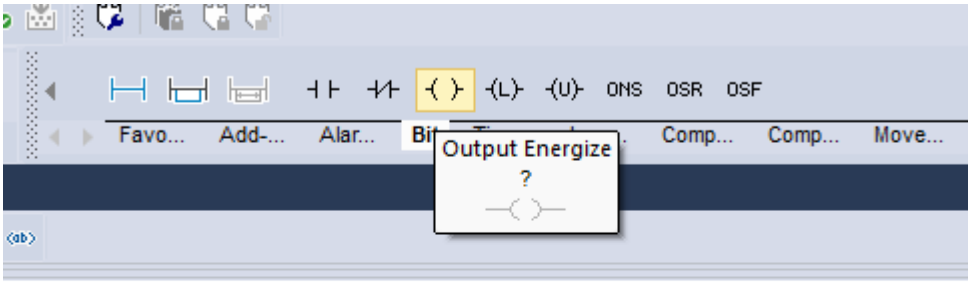
1. In the `MainRoutine` file, select `Ring 0` and add an `Examine On` instruction:

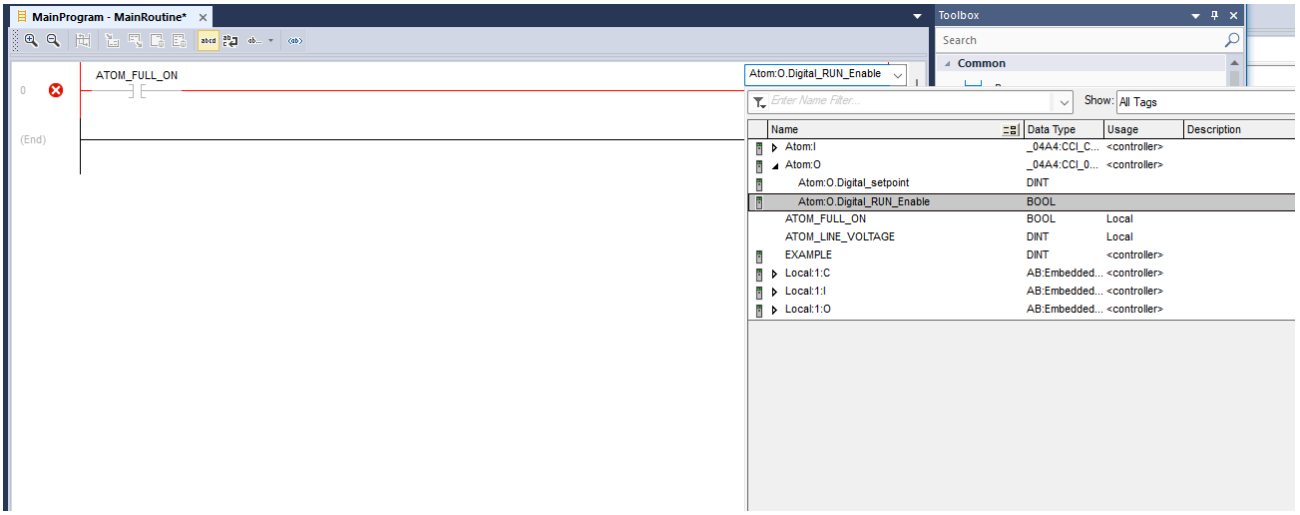



2. Configure this instruction to examine the `ATOM_FULL_ON` tag:

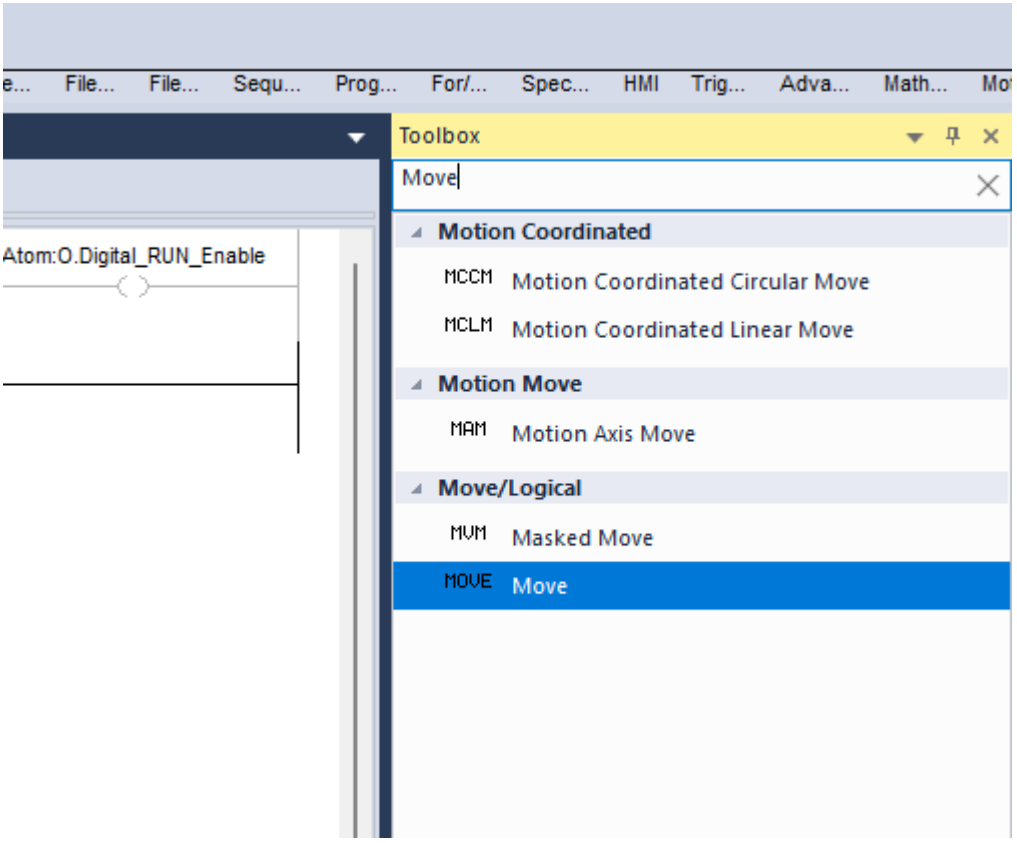


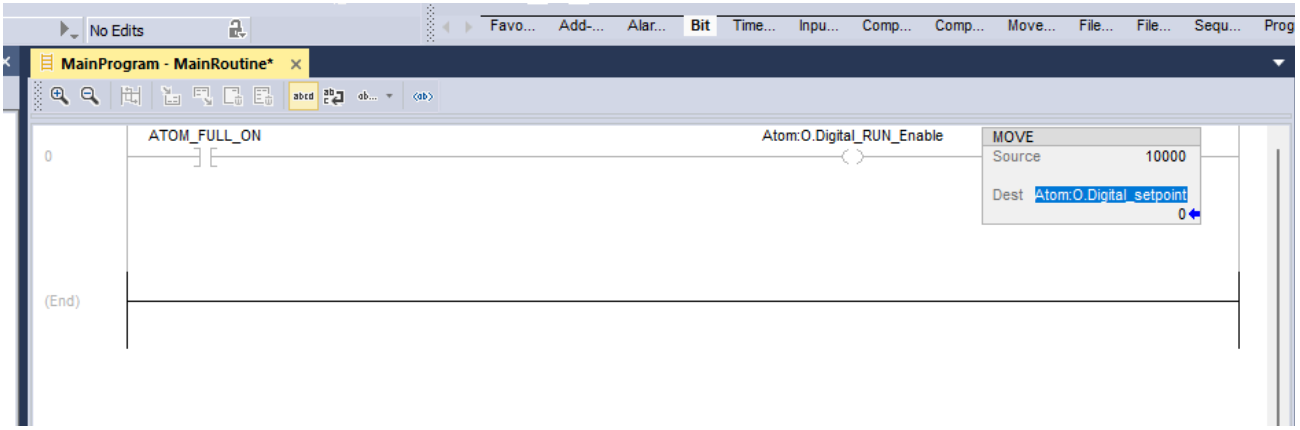

3. Add an `Output Energize` instruction and select `Atom:O.Digital_RUN_Enable`:



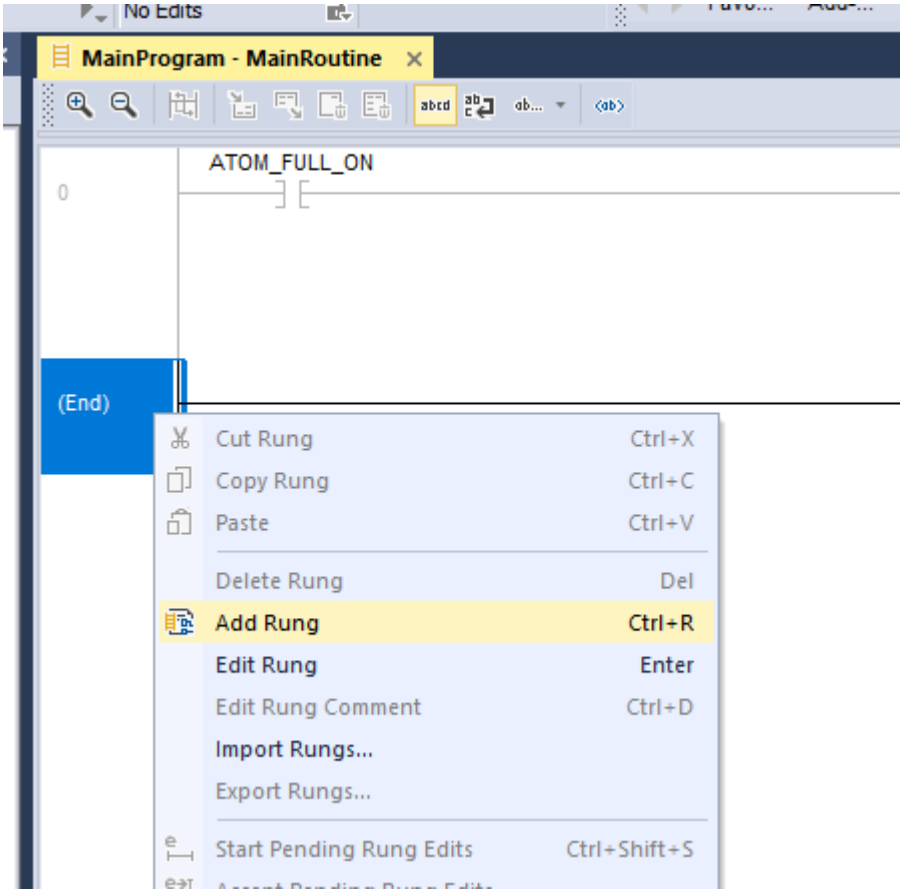



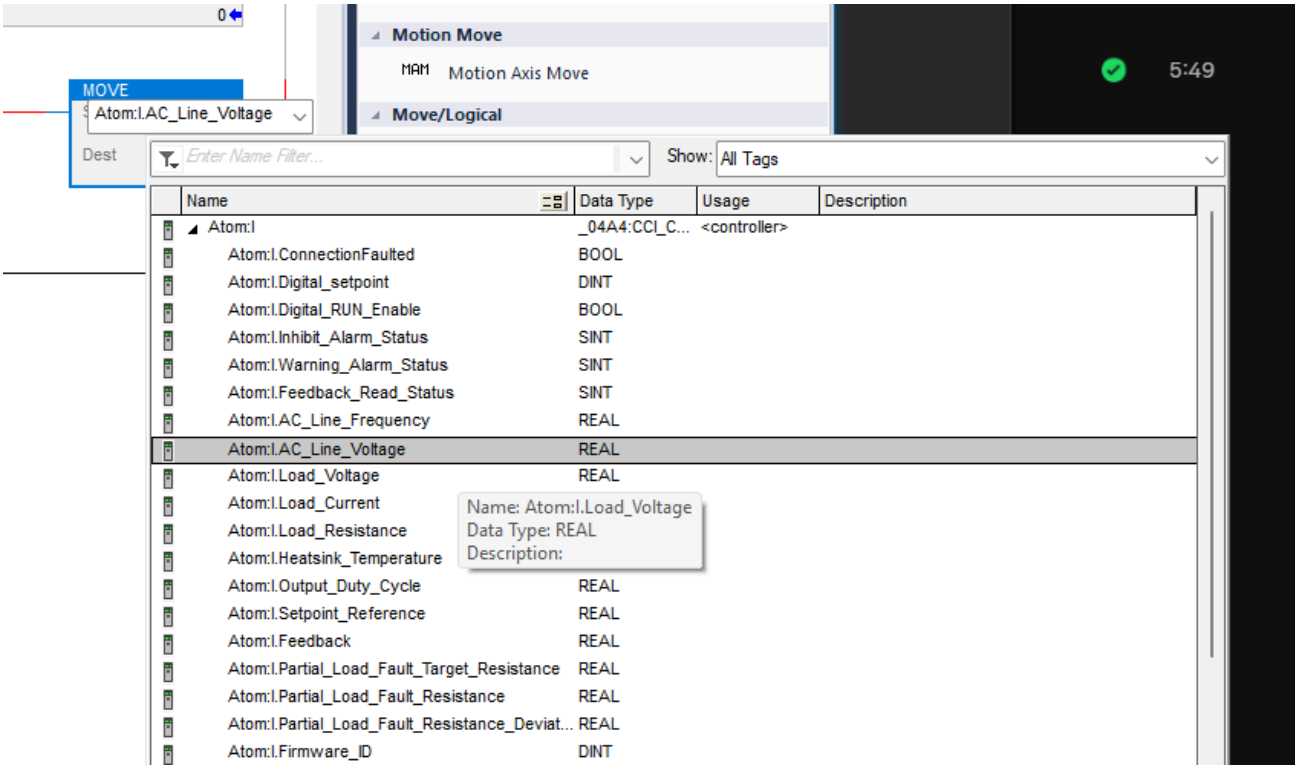

4. Add a `Move` instruction and set *source* to `10000` and *dest* to `Atom:O.Digital_setpoint`.




5. Right-click and select **Add rung**. In this new rung, add a `MOVE` instruction and set *source* to `Atom:I.AC_Line_Voltage` and *dest* to `ATOM_LINE_VOLTAGE`:

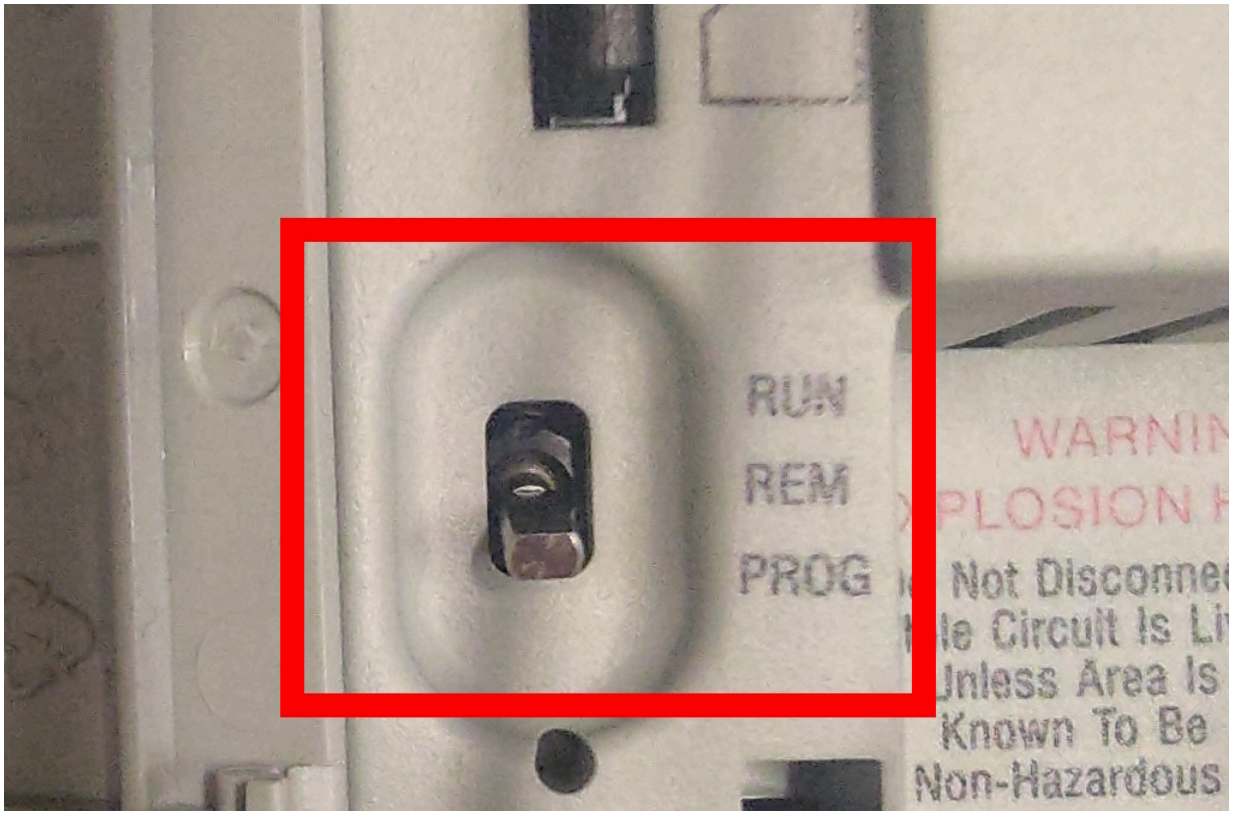

No Edits
MainProgram - MainRoutine
ATOM_FULL_ON
0
(End)
Cut Rung Ctrl+X
Copy Rung Ctrl+C
Paste Ctrl+V
Delete Rung Del
Add Rung Ctrl+R
Edit Rung Enter
Edit Rung Comment Ctrl+D
Import Rungs...
Export Rungs...
Start Pending Rung Edits Ctrl+Shift+S

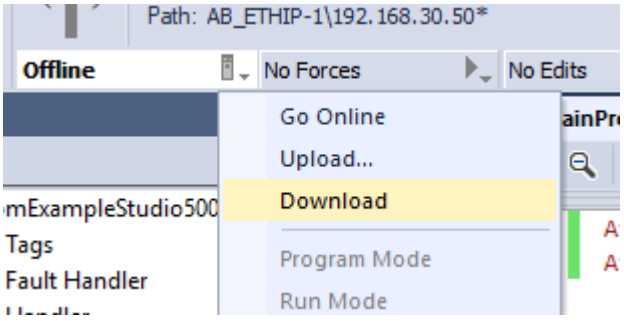

0
MOVE
Atom:I.AC_Line_Voltage
Dest
Motion Move
MAM Motion Axis Move
Move/Logical
5:49
Enter Name Filter...
Show: All Tags
Name
Data Type
Usage
Description
Atom:I _04A4:CCI_C... <controller>
Atom:I.ConnectionFaulted BOOL
Atom:I.Digital_setpoint DINT
Atom:I.Digital_RUN_Enable BOOL
Atom:I.Inhibit_Alarm_Status SINT
Atom:I.Warning_Alarm_Status SINT
Atom:I.Feedback_Read_Status SINT
Atom:I.AC_Line_Frequency REAL
Atom:I.AC_Line_Voltage REAL
Atom:I.Load_Voltage REAL
Atom:I.Load_Current
Atom:I.Load_Resistance
Atom:I.Heatsink_Temperature
Atom:I.Output_Duty_Cycle REAL
Atom:I.Setpoint_Reference REAL
Atom:I.Feedback REAL
Atom:I.Partial_Load_Fault_Target_Resistance REAL
Atom:I.Partial_Load_Fault_Resistance REAL
Atom:I.Partial_Load_Fault_Resistance_Deviat... REAL
Atom:I.Firmware_ID DINT
Name: Atom:I.Load_Voltage
Data Type: REAL
Description:

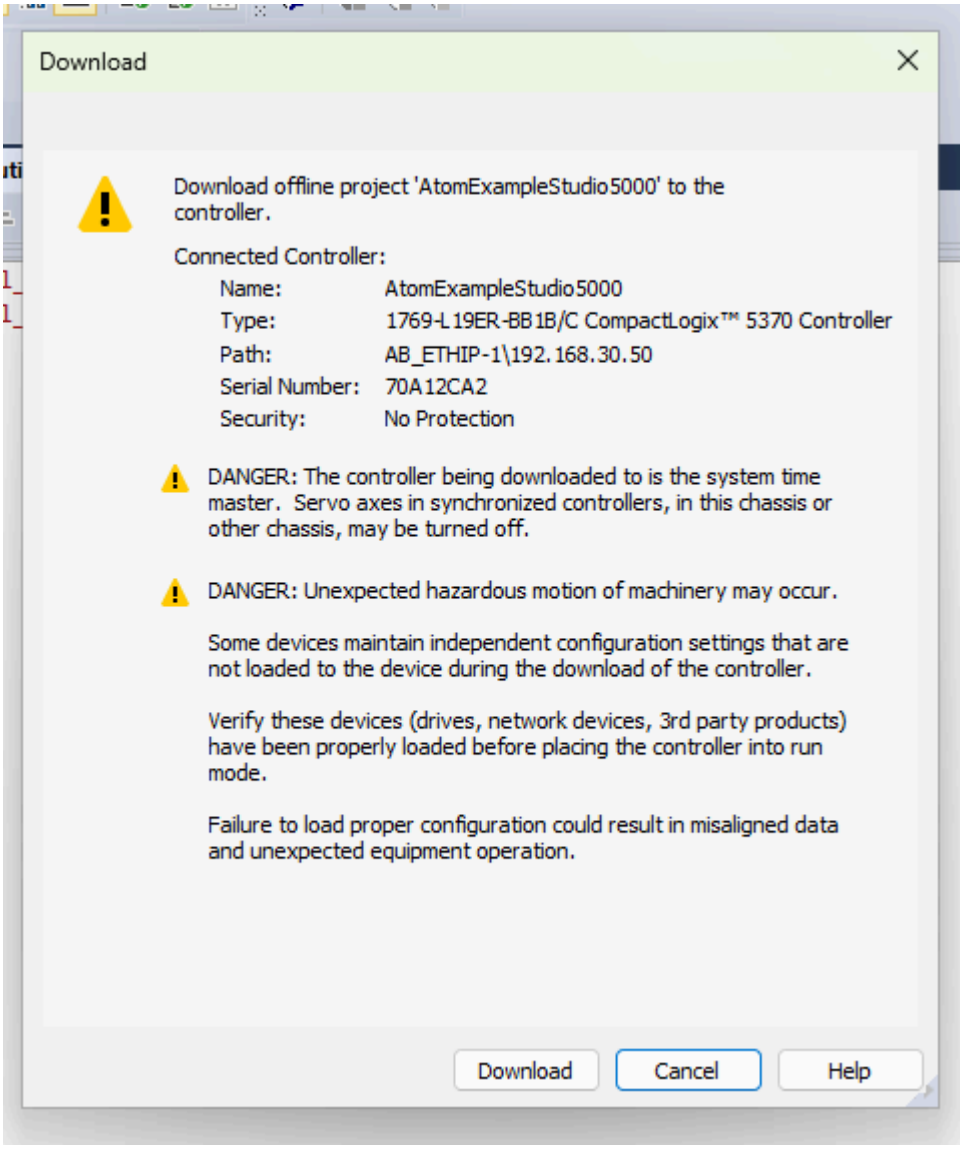
6. Select the PLC dropdown and click **Download** to download the program to your PLC:

> Ensure the switch on your PLC is set to `PROG` mode before downloading.


Allen-Bradley
RUN
FORCE
I/O
OK
NS
LINK 1
LINK 2
SD

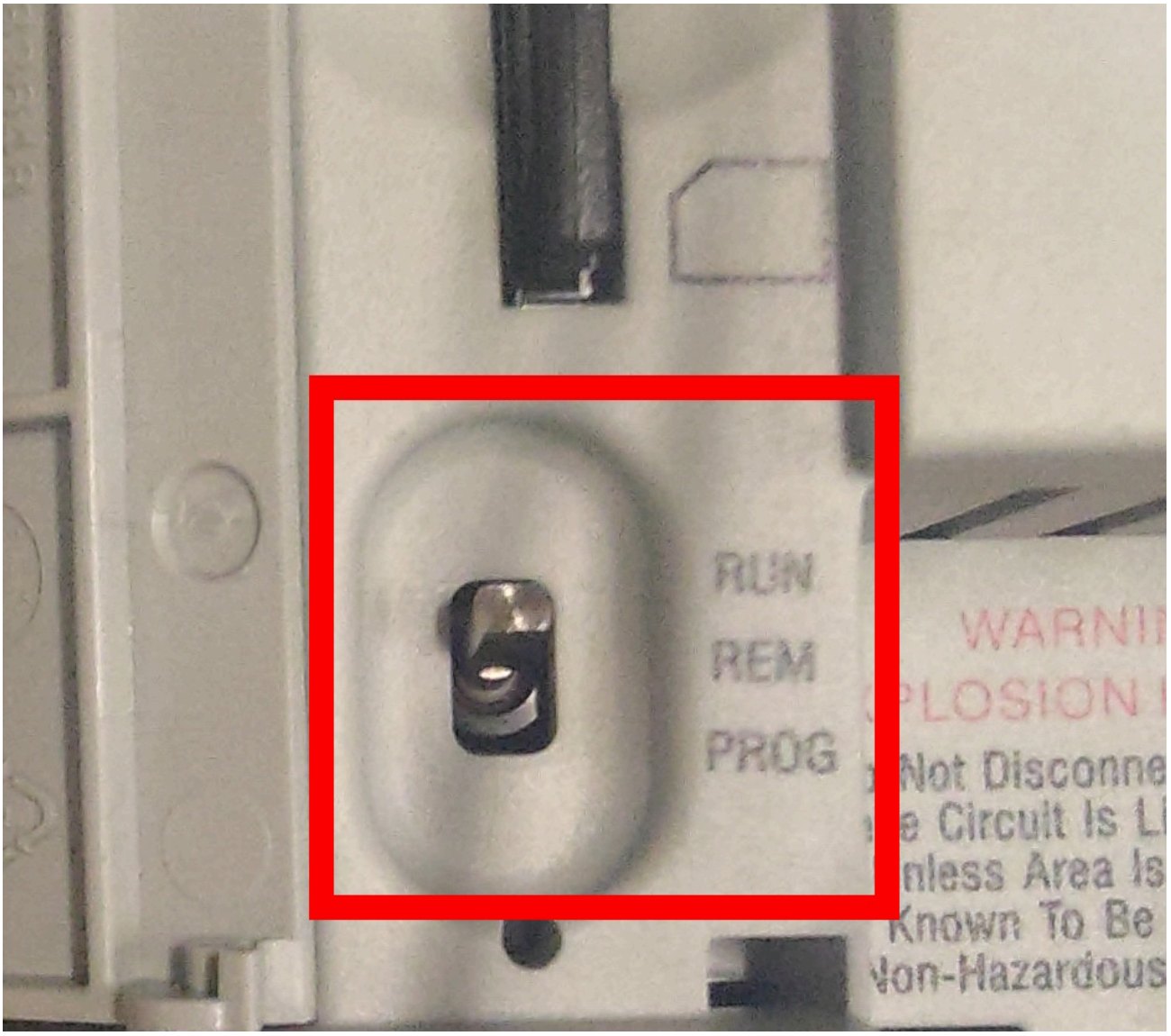

RUN
REM
PROG
WARNIN
PLOSION
Not Disconne
le Circuit Is Li
Unless Area Is
Known To Be
Non-Hazardous


Path: AB_ETHIP-1\192.168.30.50*
Offline
No Forces
No Edits
Go Online
Upload...
Download
Program Mode
Run Mode
mExampleStudio500
Tags
Fault Handler

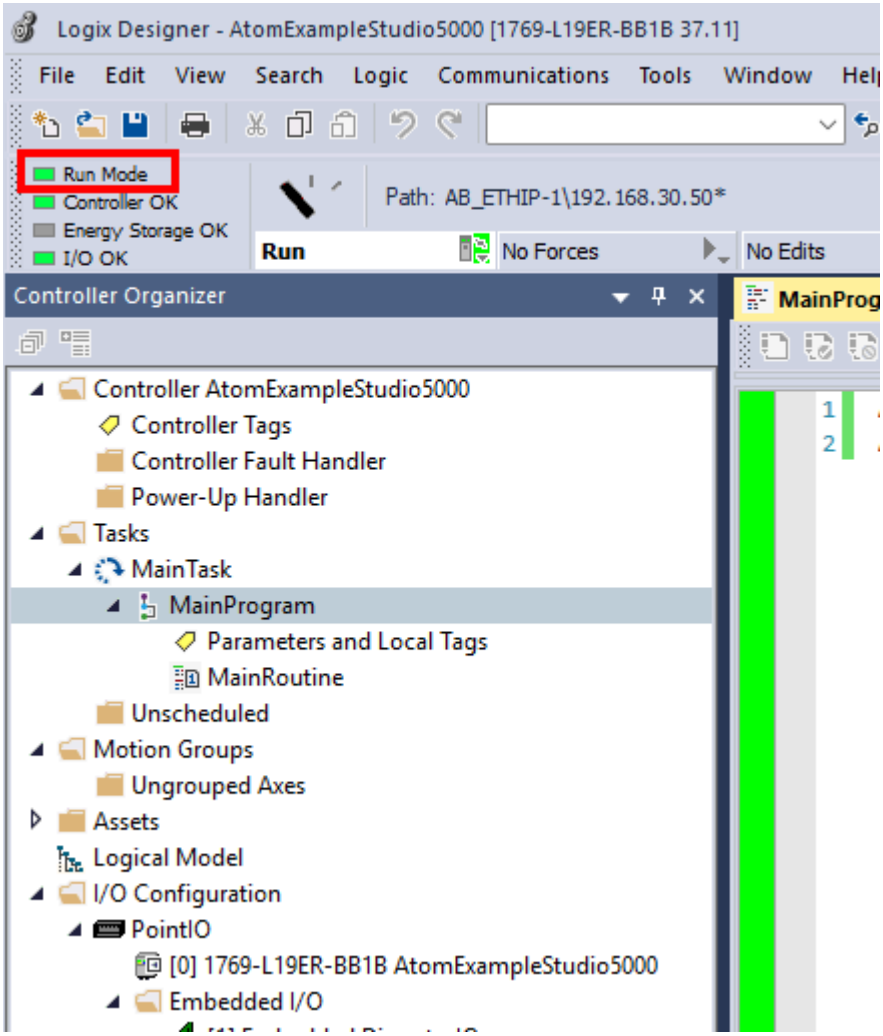



7. Flip the switch on your PLC to `RUN` mode.


Allen-Bradley
RUN
NS
FORCE
LINK 1
I/O
LINK 2
OK
SD



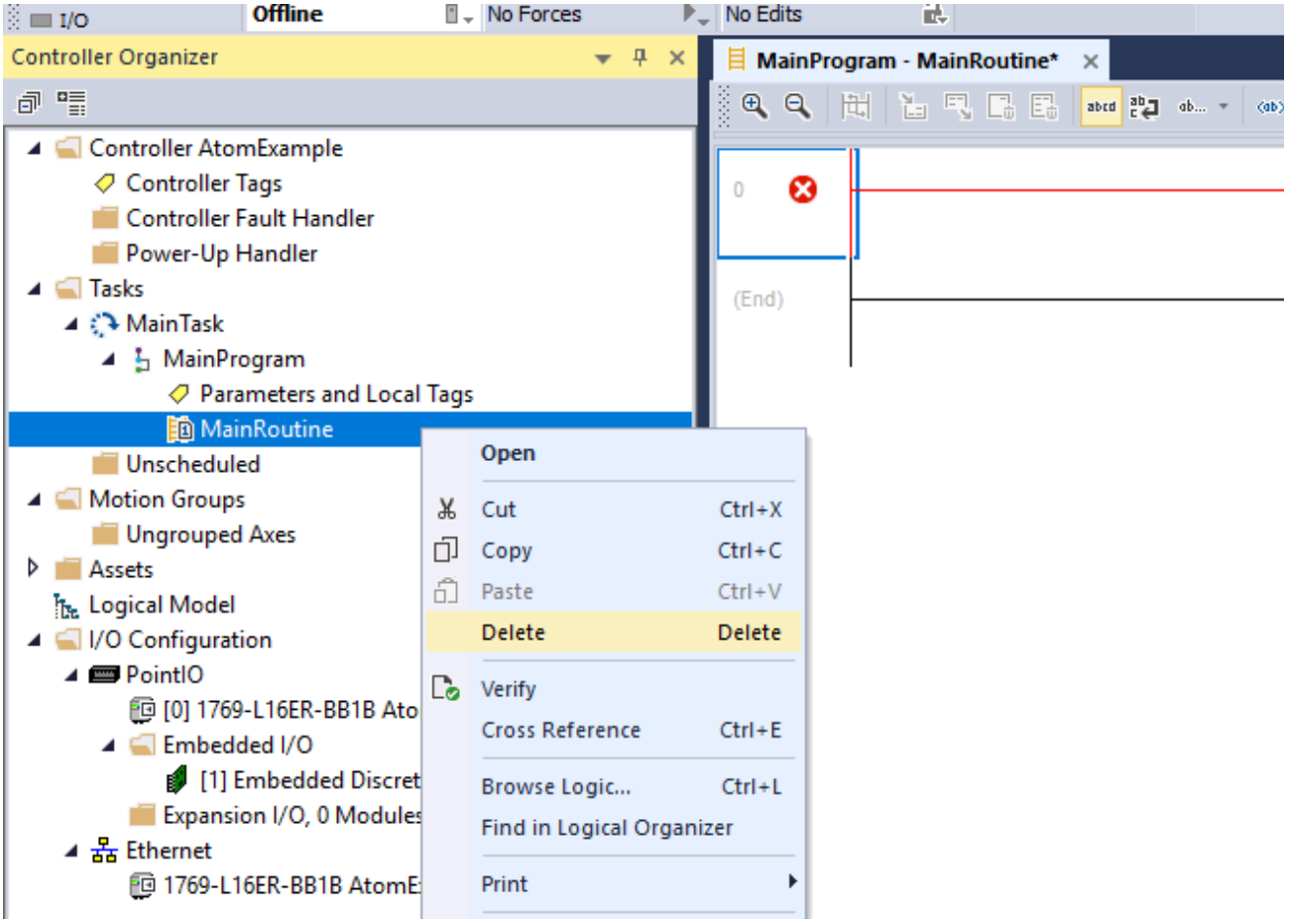

8. If everything worked properly, the controller **Run Mode** indicator light should turn green:



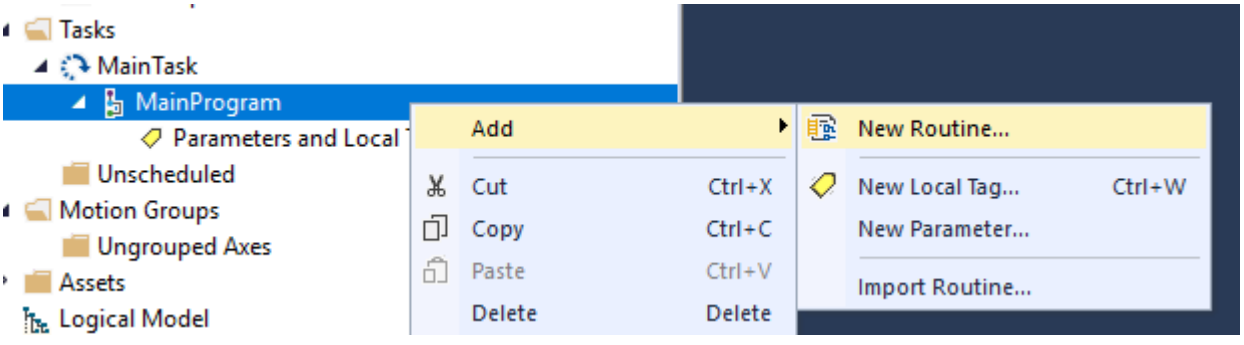

Next, jump to the Creating a user interface section.

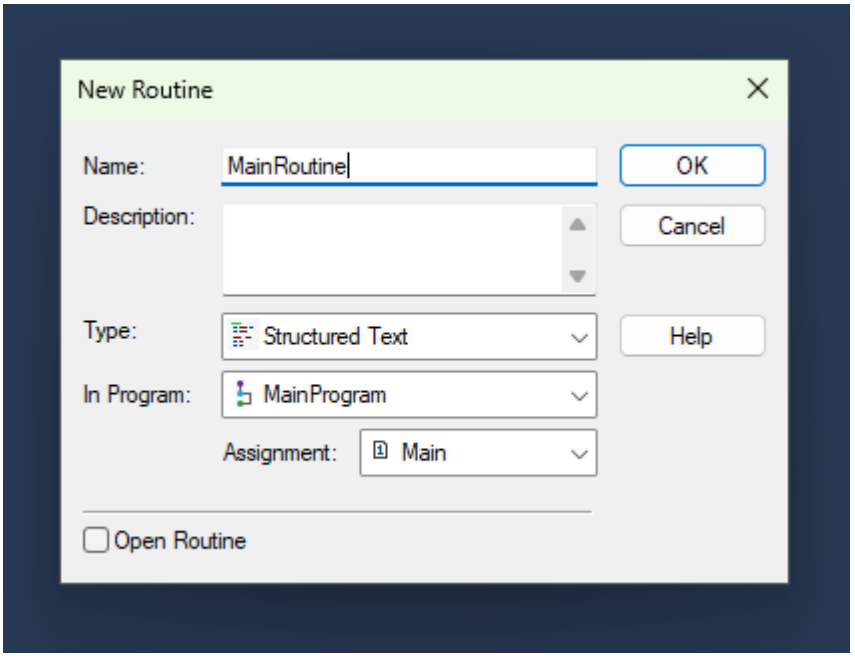
# Structured Text

1. Delete the default `MainRoutine`:

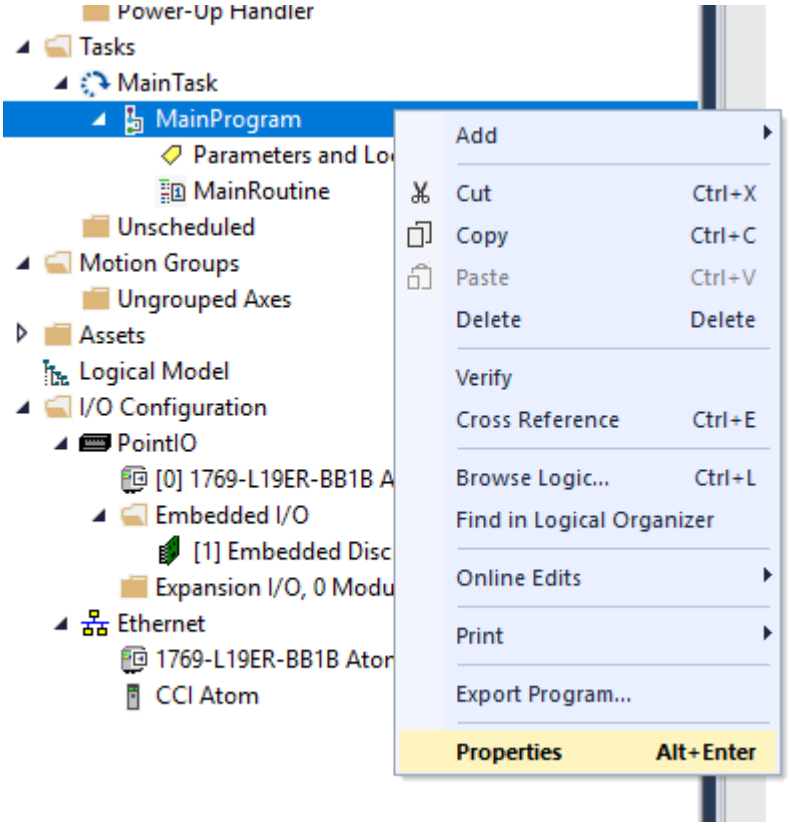


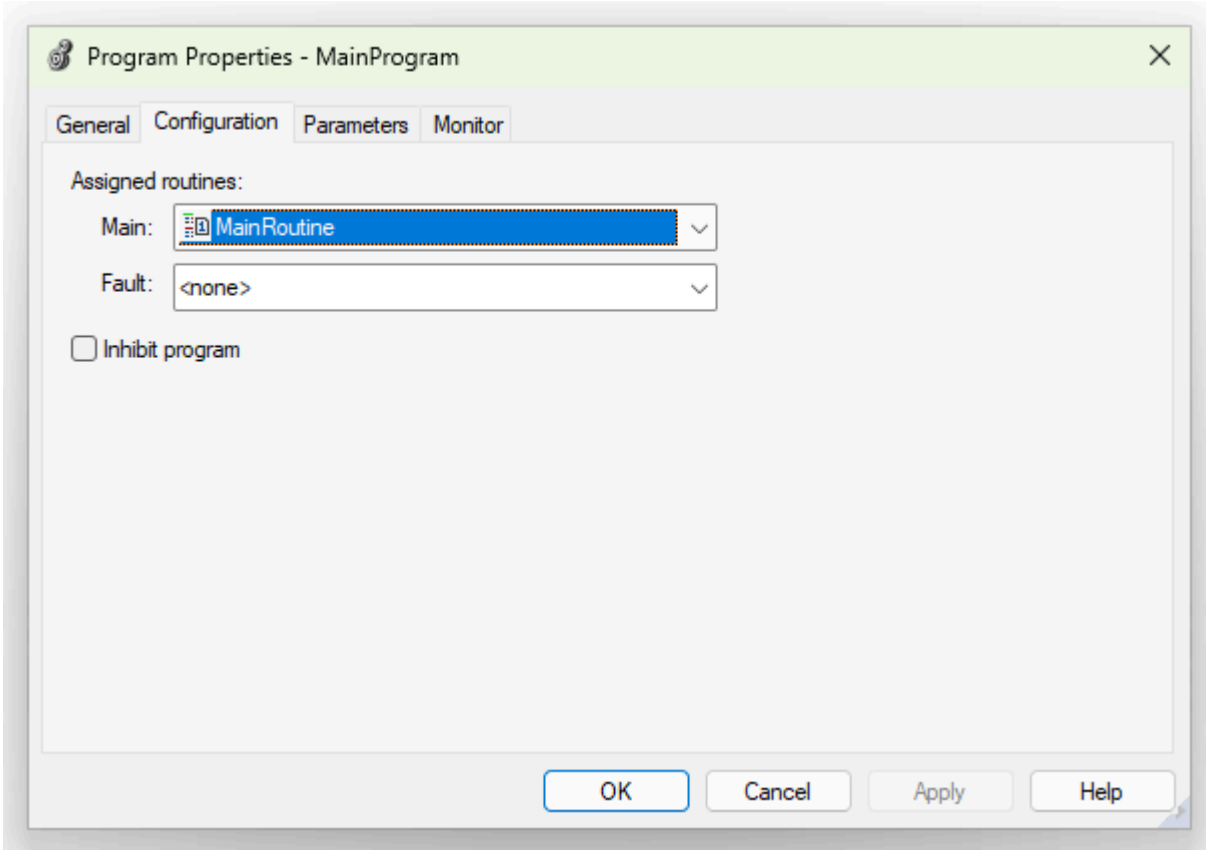

2. Right-click `MainProgram` and select **Add Routine**. Name it `MainRoutine`, set the *Type* to `Structured Text`, and click **OK**:



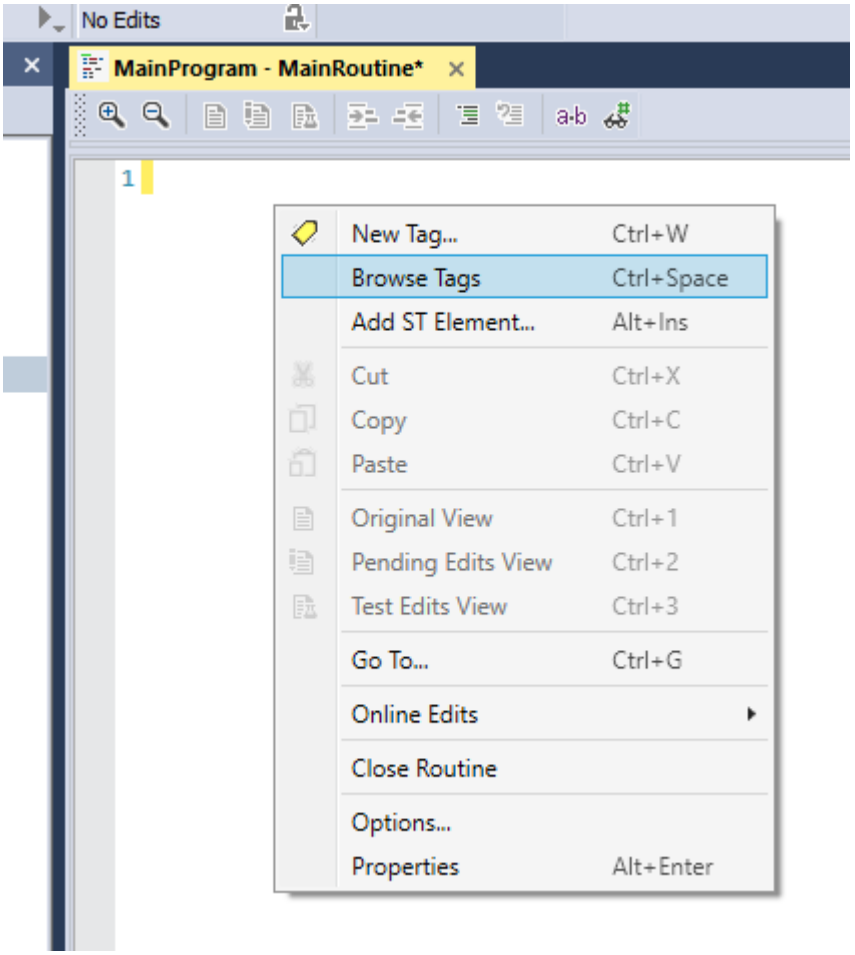



3. Right-click `MainRoutine`, select **Properties**, and ensure `MainRoutine` is set as the *Main Routine* in the **Configuration** tab:

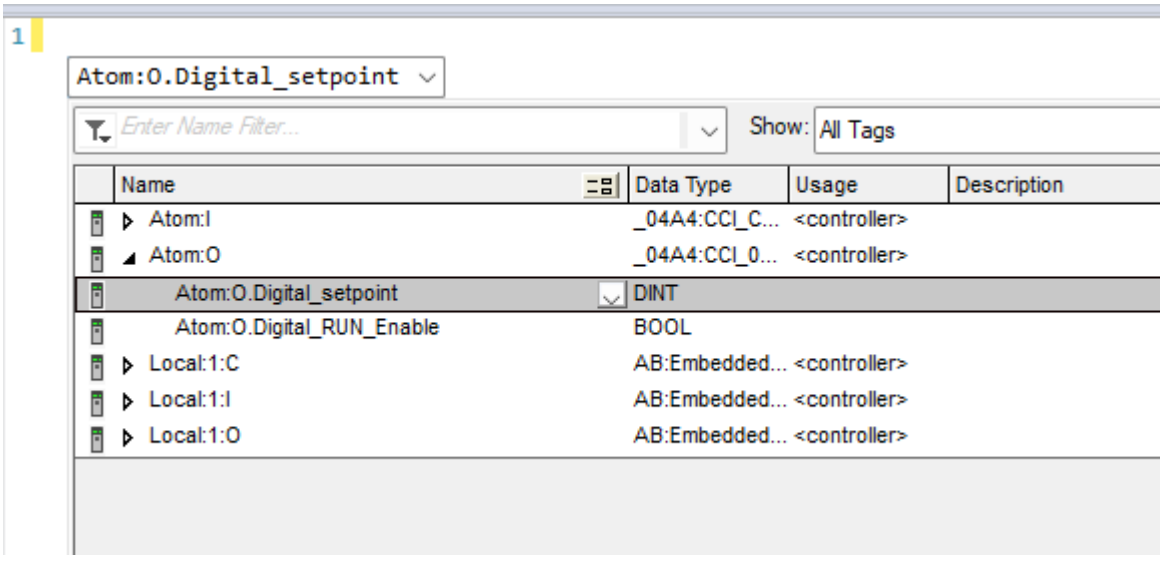





4. Insert tags by right-clicking in `MainRoutine` and selecting **Browse Tags**.

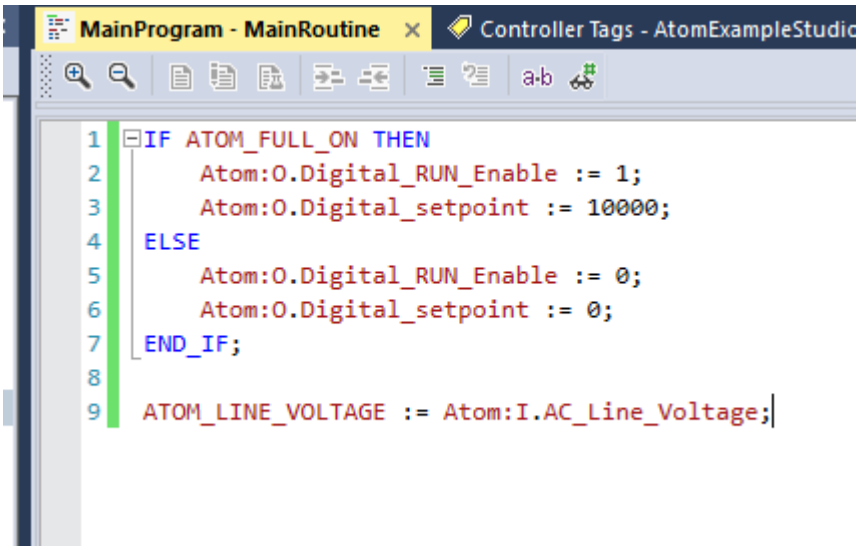



5. You can insert `Atom:I` (input) and `Atom:O` (output) tags to control ATOM:



6. Add the following code to `MainRoutine`:

```
IF ATOM_FULL_ON THEN
        Atom:O.Digital_RUN_Enable := 1;
        Atom:O.Digital_setpoint := 10000;
ELSE
        Atom:O.Digital_RUN_Enable := 0;
        Atom:O.Digital_setpoint := 0;
END_IF;

ATOM_LINE_VOLTAGE := Atom:I.AC_Line_Voltage;
```

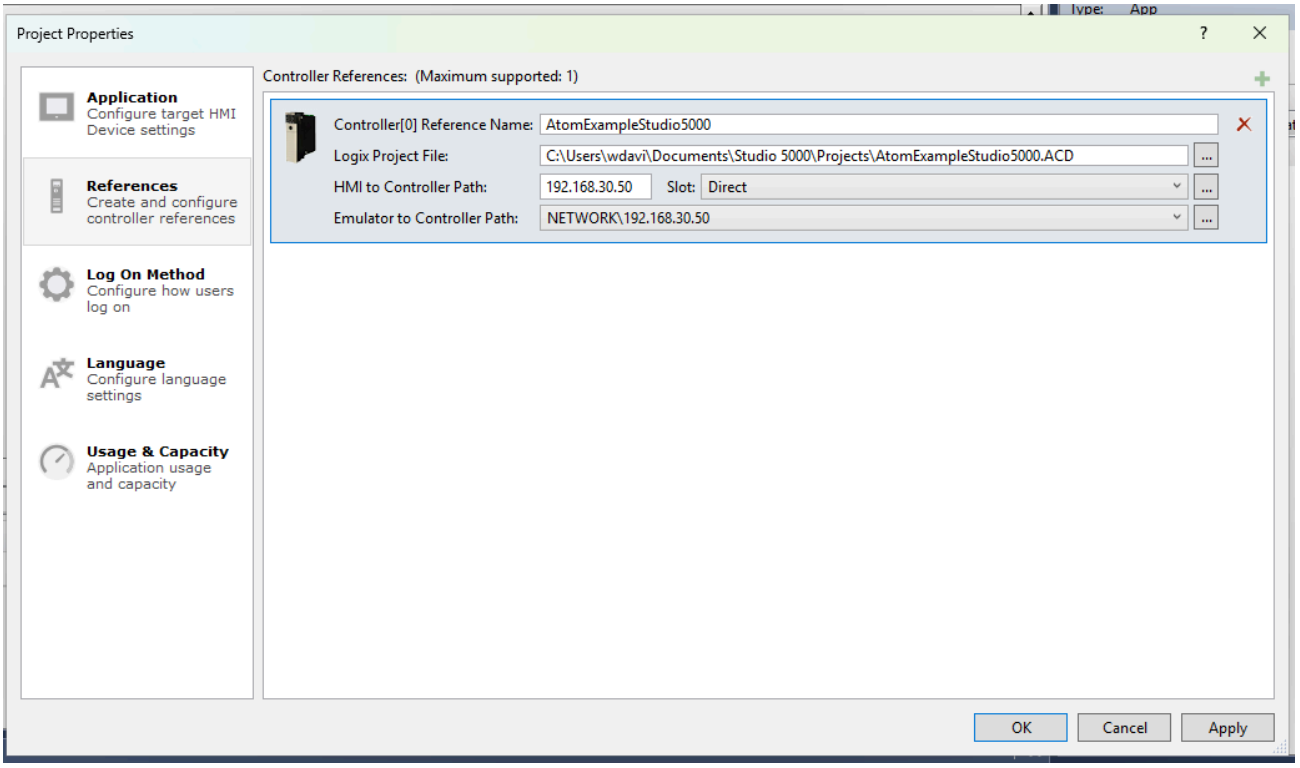



Next, jump to the Creating a user interface section.

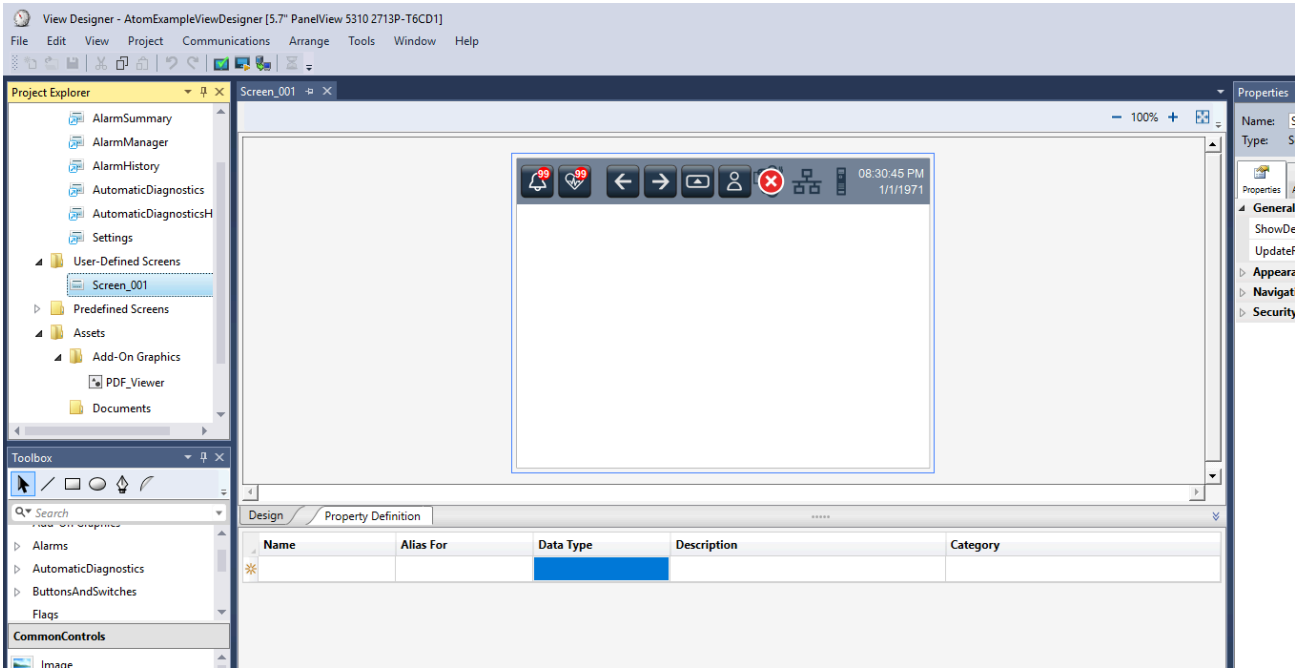
# Creating a user interface

> **INFO**
>
> Studio 5000 comes with a separate program called **View Designer** for creating user interfaces. It's usually installed at `C:\Program Files (x86)\Rockwell Software\Studio 5000\View Designer\ENU\V10\ViewDesigner.exe`

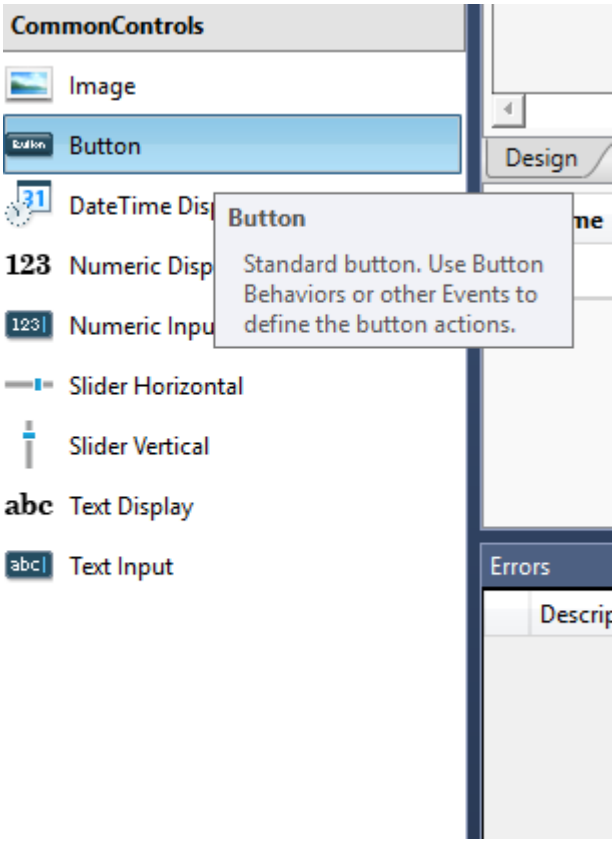
1. Launch View Designer and create a new project with the following settings:
    - **Controller[0] Reference Name**: `AtomExampleStudio5000`
    - **Logix Project File**: `path-to-your-project\AtomExampleStudio5000.ACD`
    - **HMI to Controller Path**: `192.168.30.50`
    - **Emulerator to Controller Path**: `NETWORK\192.168.30.50`

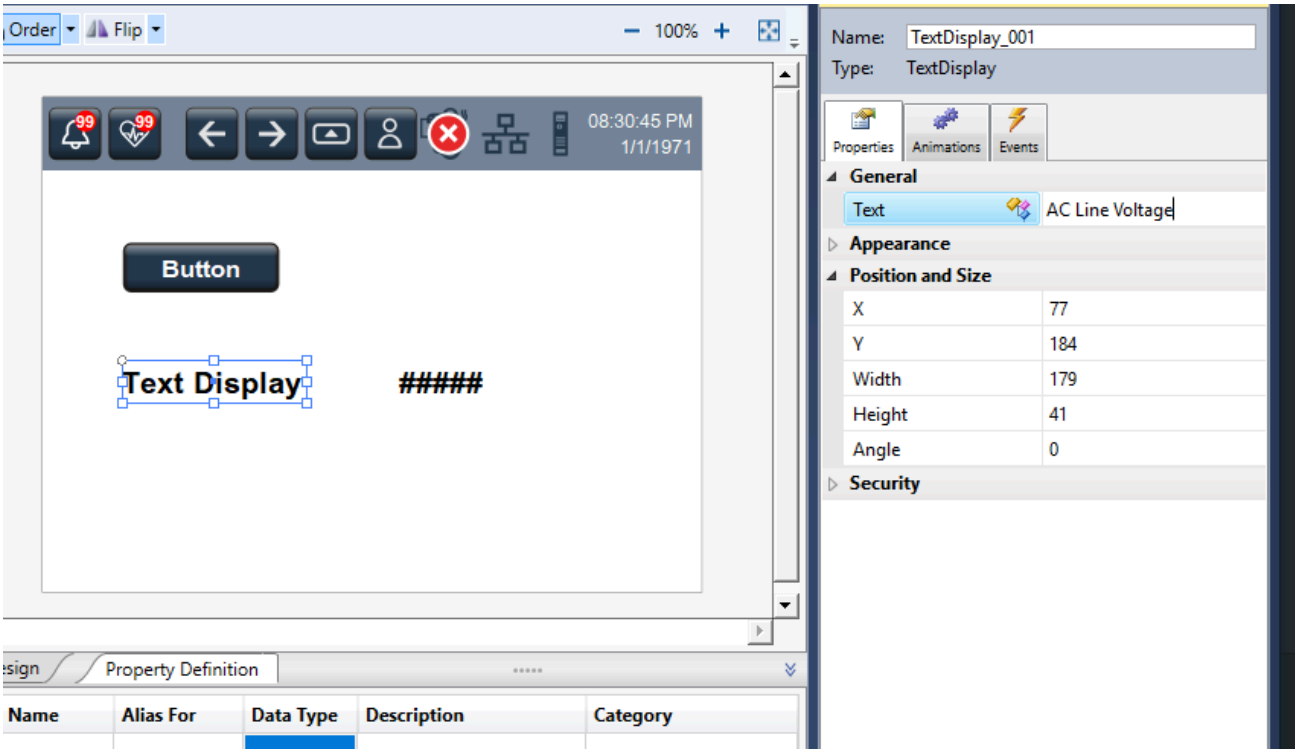



2. Open `Screen_001`:

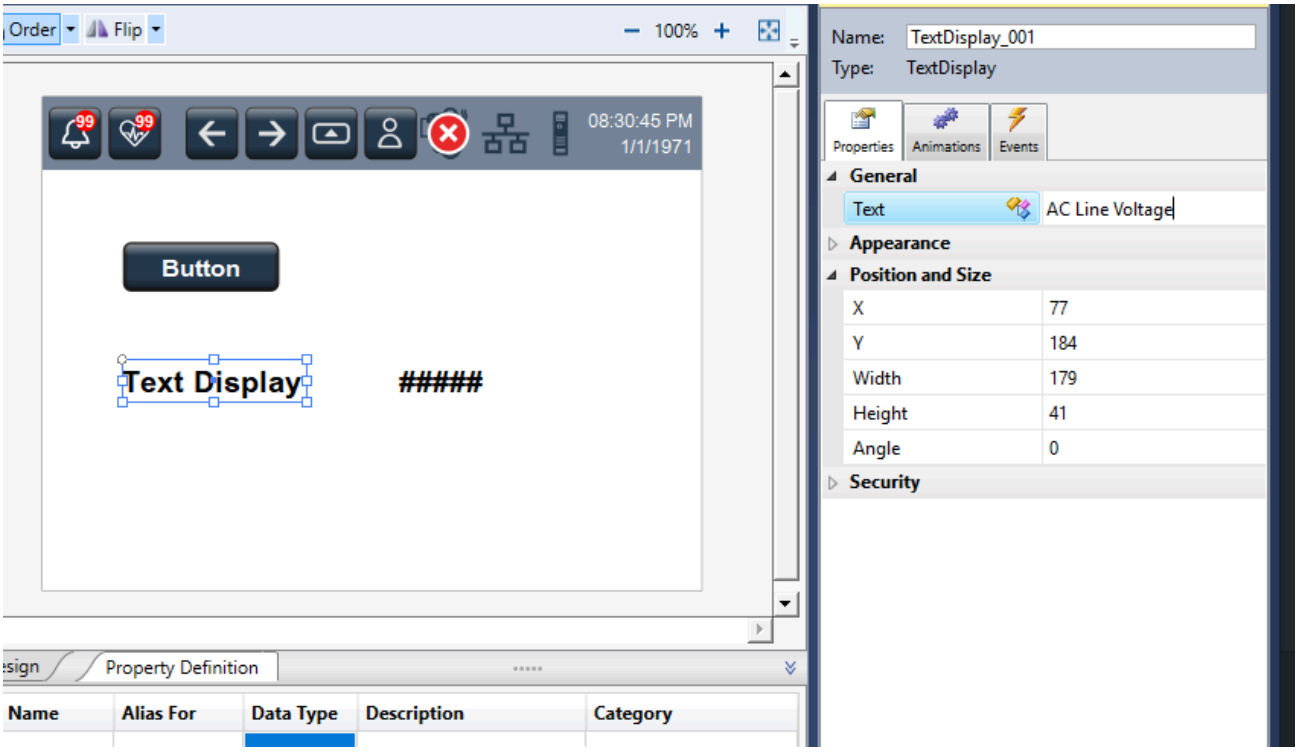



3. In the **CommonControls** toolbox, drag three components onto the screen:
    - **Button**
    - **Numeric Display**
    - **Text Display**

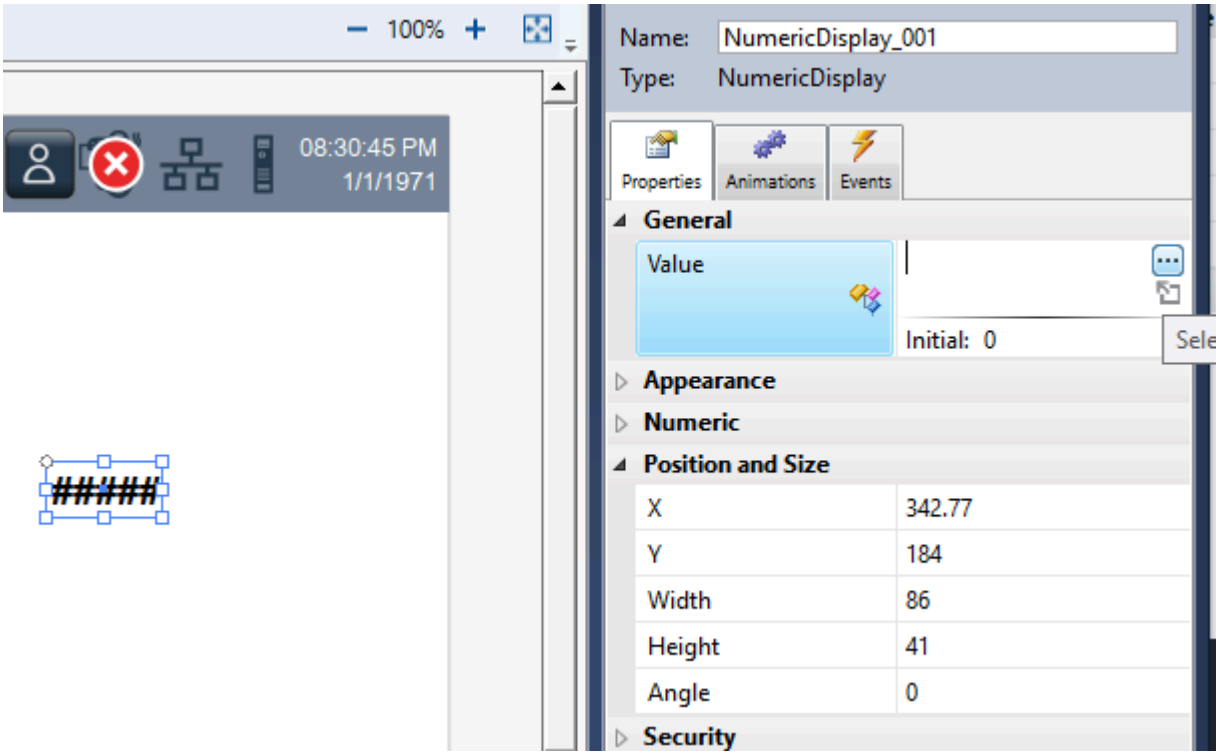

CommonControls
Image
Button
DateTime Disp
Numeric Disp
Numeric Inpu
Slider Horizontal
Slider Vertical
Text Display
Text Input
Button
Standard button. Use Button Behaviors or other Events to define the button actions.
Design
Errors
Descrip

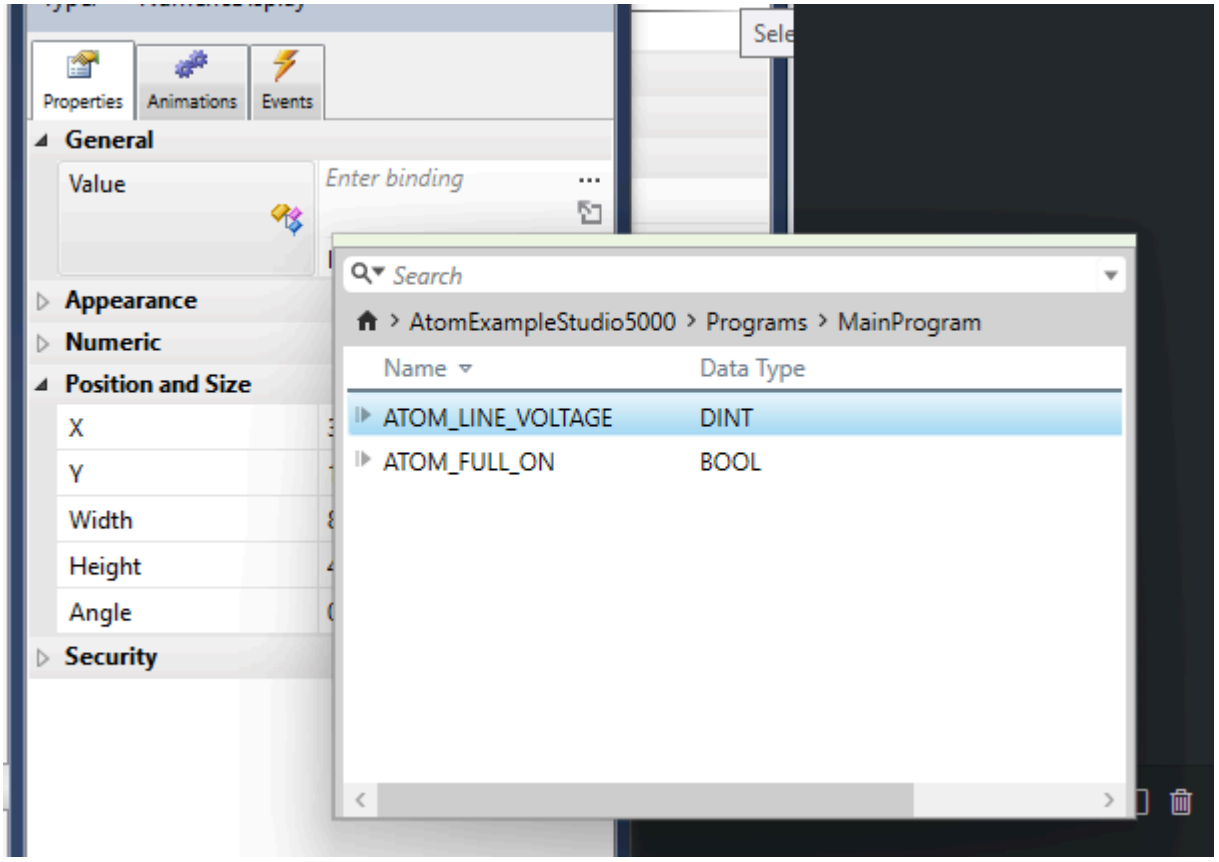

Order
Flip
100%
08:30:45 PM
1/1/1971
Button
Text Display
#####
Property Definition
Name
Alias For
Data Type
Description
Category
Name: TextDisplay_001
Type: TextDisplay
Properties
Animations
Events
General
Text
AC Line Voltage
Appearance
Position and Size
X 77
Y 184
Width 179
Height 41
Angle 0
Security

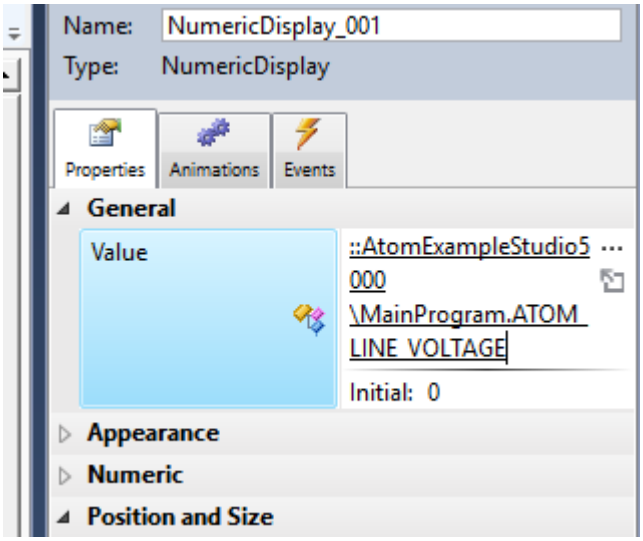
4. Select the **Text Display** component and set the text to `AC Line Voltage`:



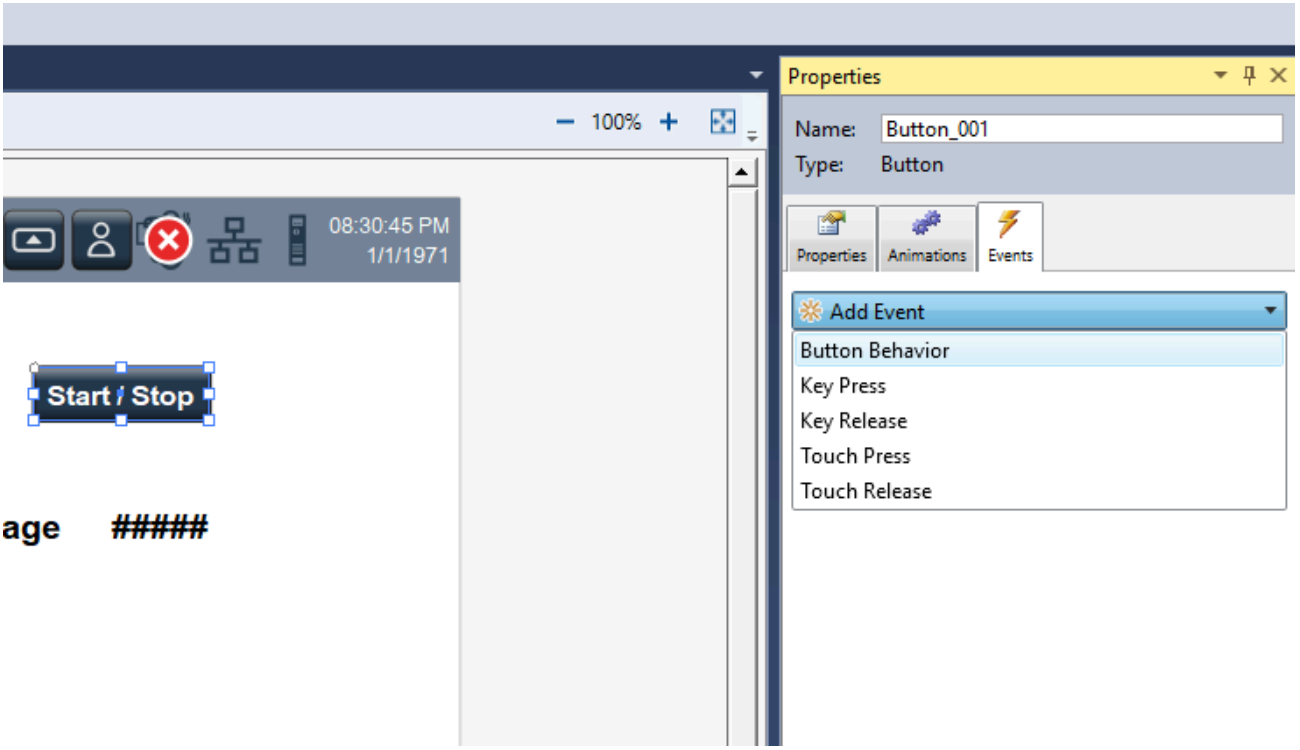

5. Select the **Numeric Display** component and set the **Value** (in the **Properties** panel) to `ATOM_LINE_VOLTAGE`:


100%
08:30:45 PM
1/1/1971
#####
Name: NumericDisplay_001
Type: NumericDisplay
Properties
Animations
Events
General
Value
Initial: 0
Appearance
Numeric
Position and Size
X 342.77
Y 184
Width 86
Height 41
Angle 0
Security

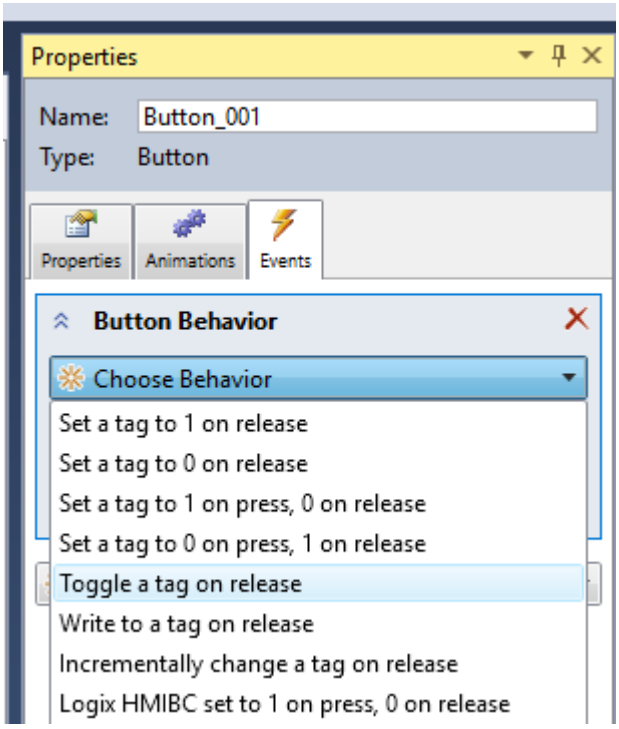

Properties
Animations
Events
General
Value
Enter binding
Appearance
Numeric
Position and Size
X
Y
Width
Height
Angle
Security
Search
AtomExampleStudio5000 › Programs › MainProgram
Name
Data Type
ATOM_LINE_VOLTAGE DINT
ATOM_FULL_ON BOOL

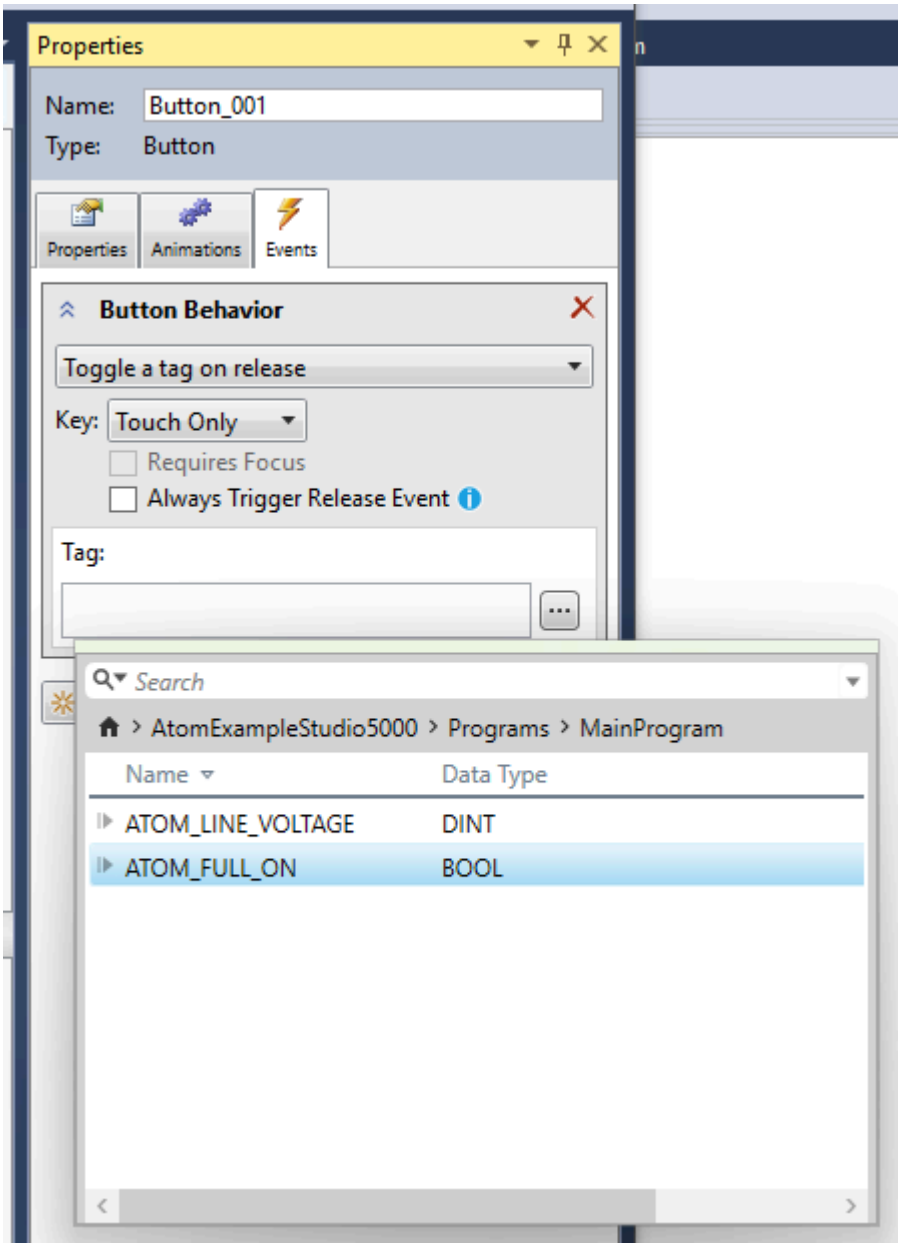


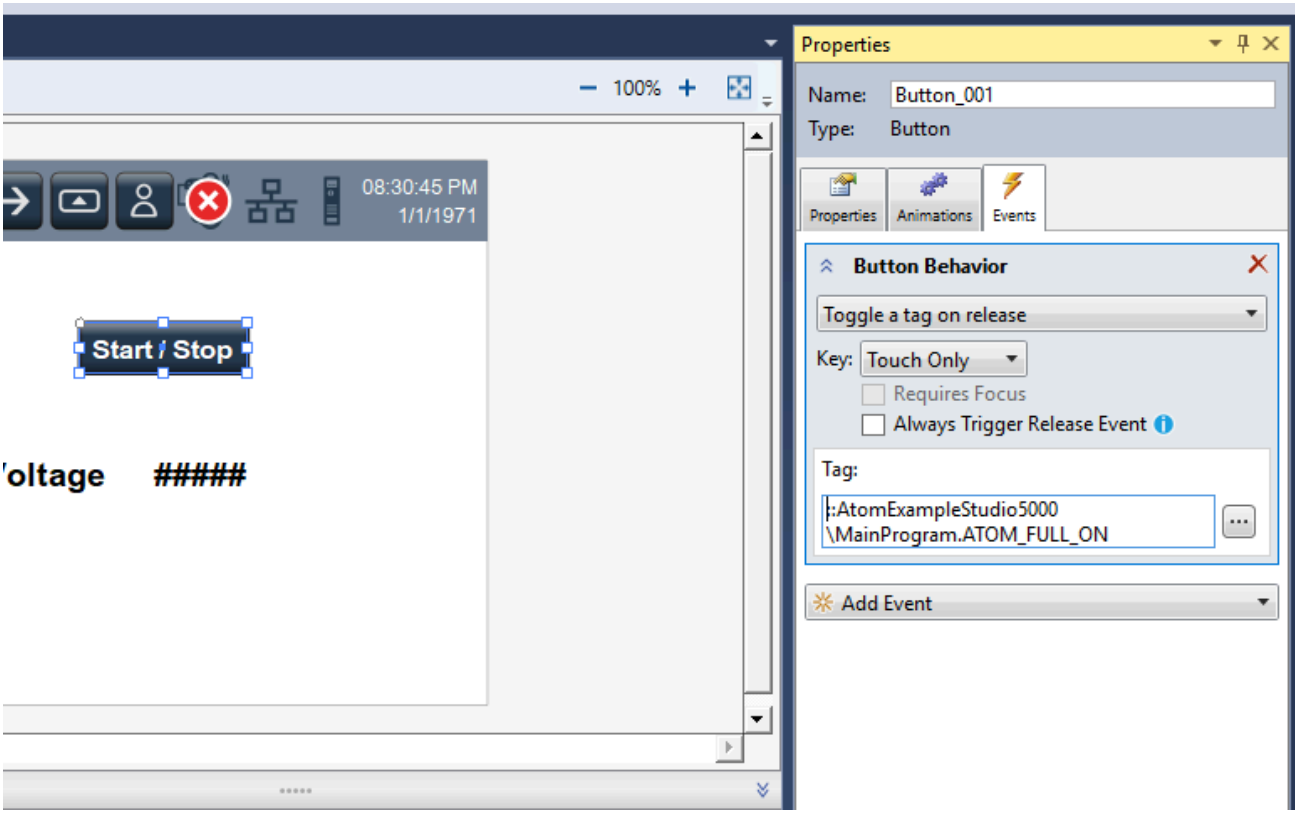

6. Select the **Button* component and set the text to `Start / Stop`. In the **Events** panel, click **Add Event**, **Button Behavior**:



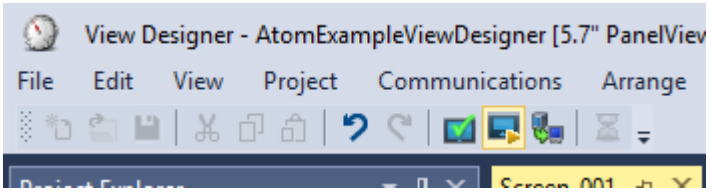

7. Select **Toggle a tag on release** and set the tag to `ATOM_FULL_ON`:


Properties
Name: Button_001
Type: Button
Properties
Animations
Events
Button Behavior
Choose Behavior
Set a tag to 1 on release
Set a tag to 0 on release
Set a tag to 1 on press, 0 on release
Set a tag to 0 on press, 1 on release
Toggle a tag on release
Write to a tag on release
Incrementally change a tag on release
Logix HMIBC set to 1 on press, 0 on release


Properties
Name: Button_001
Type: Button
Properties
Animations
Events
Button Behavior
Toggle a tag on release
Key: Touch Only
Requires Focus
Always Trigger Release Event
Tag:
Search
AtomExampleStudio5000 › Programs › MainProgram
Name
Data Type
ATOM_LINE_VOLTAGE
DINT
ATOM_FULL_ON
BOOL

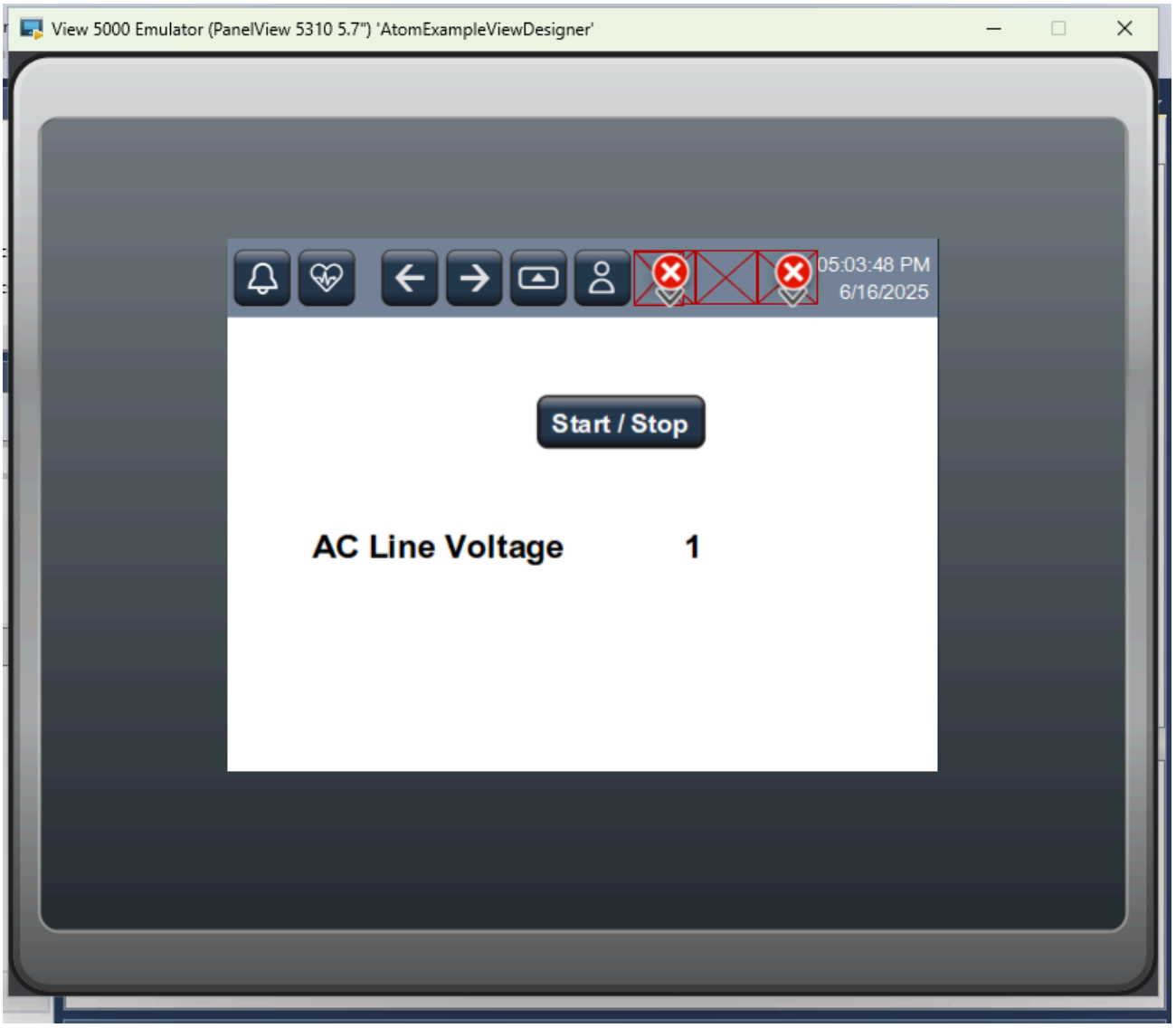



8. Select the **Emulate** button to launch the HMI emulator:

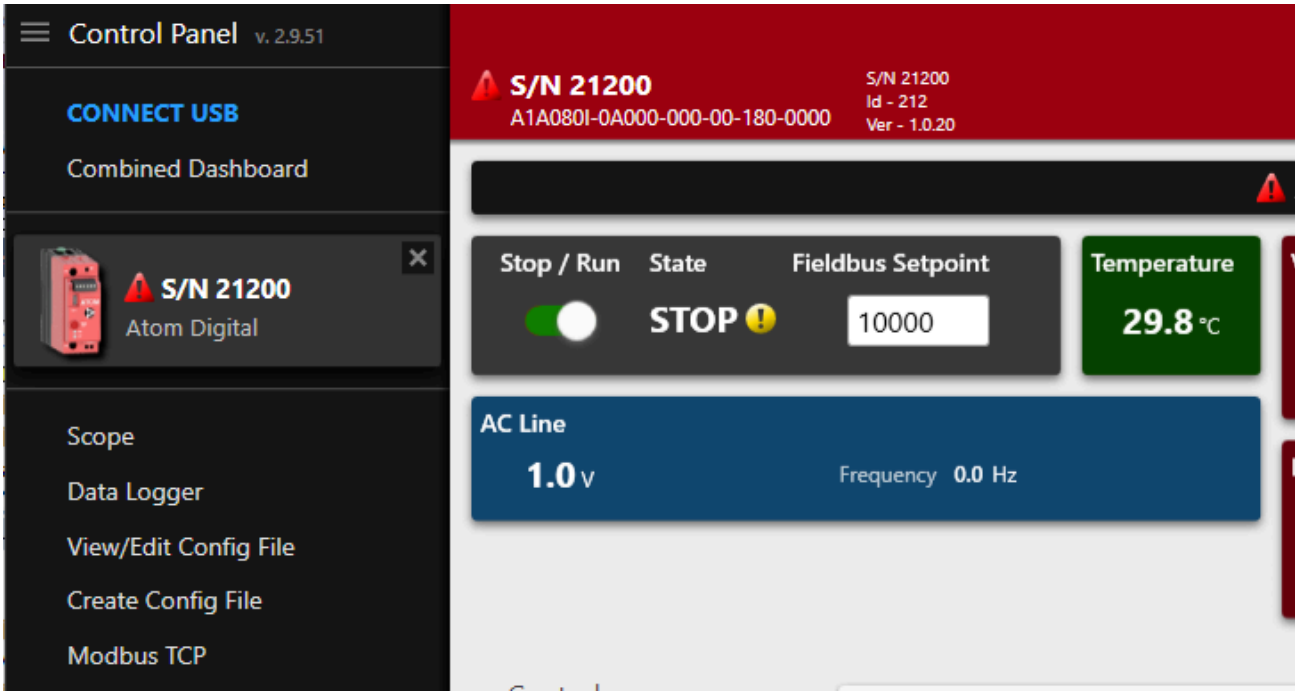



9. Ensure your PLC is in `RUN` if it is not already.

10. In the emulator, you can click `Start / Stop` to toggle ATOM's operation. The **AC Line Voltage** display should show the current line voltage (in tenths of volts (e.g., `2300` for `230.0V`)):




If you are connected to ATOM with Control Panel, you can watch the **Stop / Run** and **Fieldbus setpoint** controls change as you toggle the button in the Rockwell emulator.




# Troubleshooting

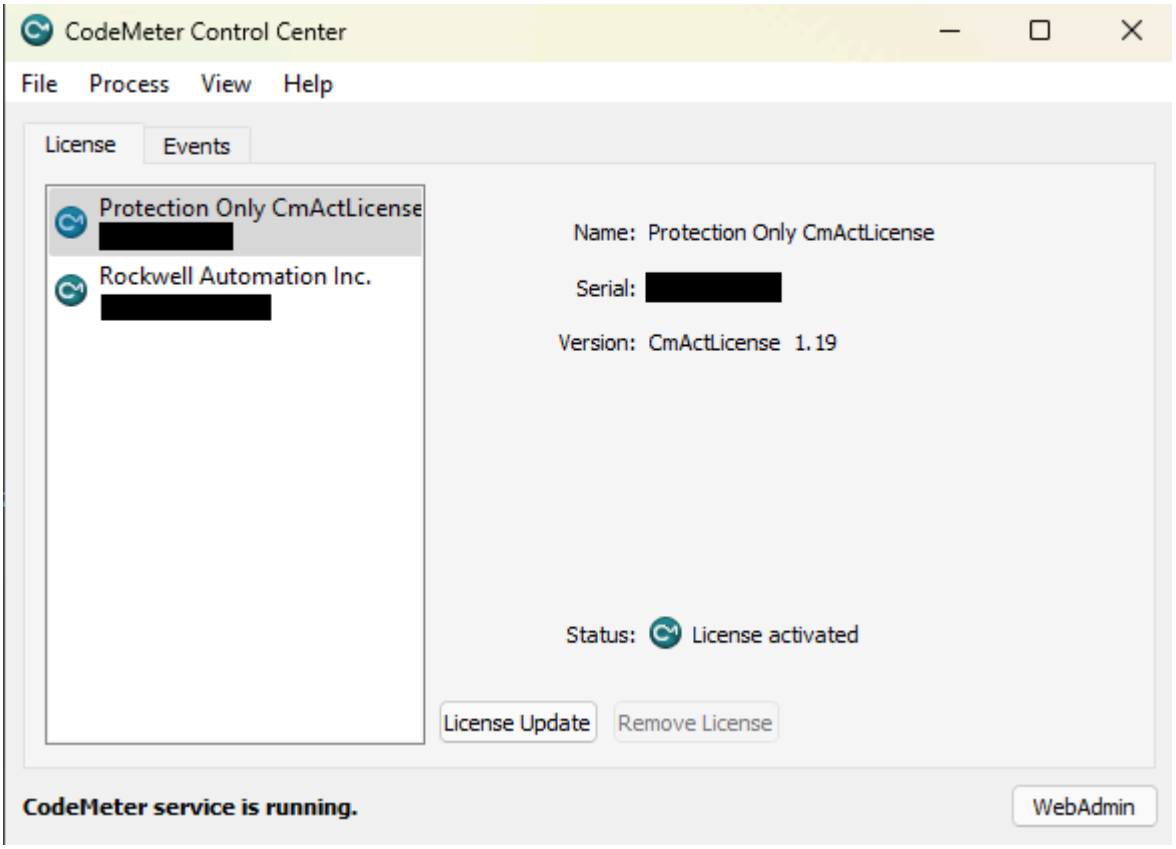
## Installation troubleshooting

### Activation issues

If you use your PC for multiple PLC environments (like Siemens TIA, Codesys, etc.) you may run into activation issues caused by CodeMeter licenses.

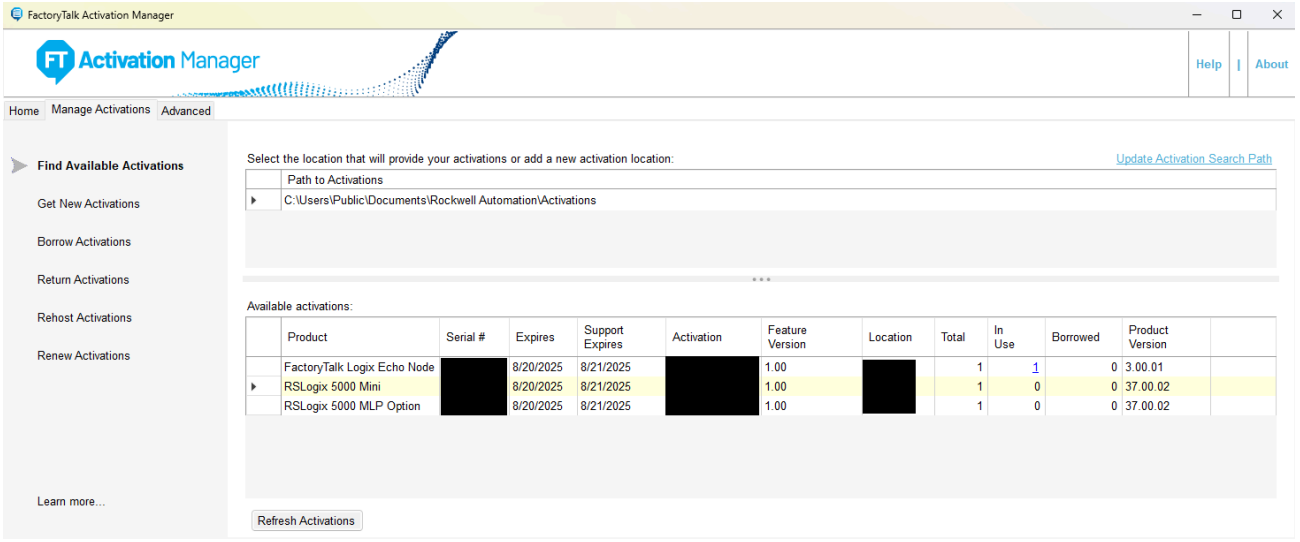
Follow this guide to delete other CodeMeter licenses as Studio 5000 requires exclusive access to the CodeMeter license manager.

Your CodeMeter should look like this:



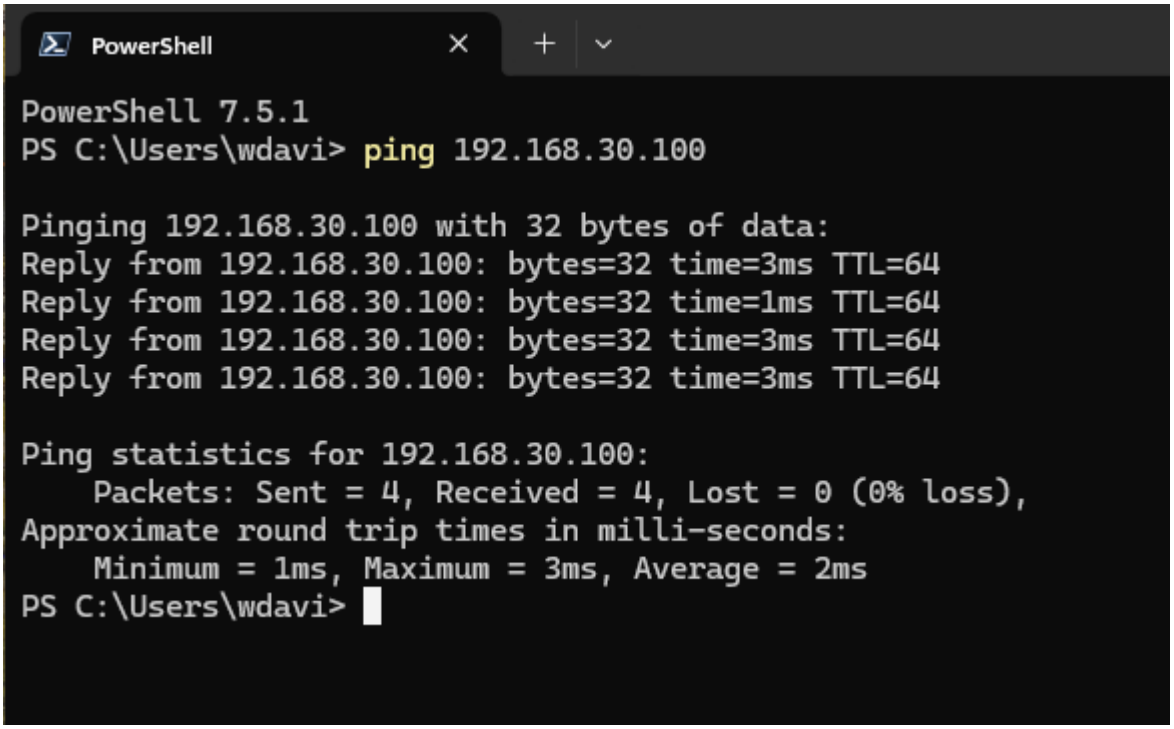

## Factory Talk activation

Use **Factory Talk Activation Manager** to ensure your have a valid Studio 5000 license.



### Can't connect to PLC or ATOM

Use the `ping` utility on Windows to check if your PC can reach the PLC/ATOM:




If:

- Ping is successful - you have a configuration problem with your PC
- Ping is unsuccessful - you have a hardware configuration, PLC configuration, or ATOM configuration problem.

# ATOM / Fieldbus / EtherNet/IP / Codesys

In this tutorial, you'll learn how to use Codesys with the SoftPLC emulator to connect to ATOM using EtherNet/IP 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 EtherNet/IP Profile.

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

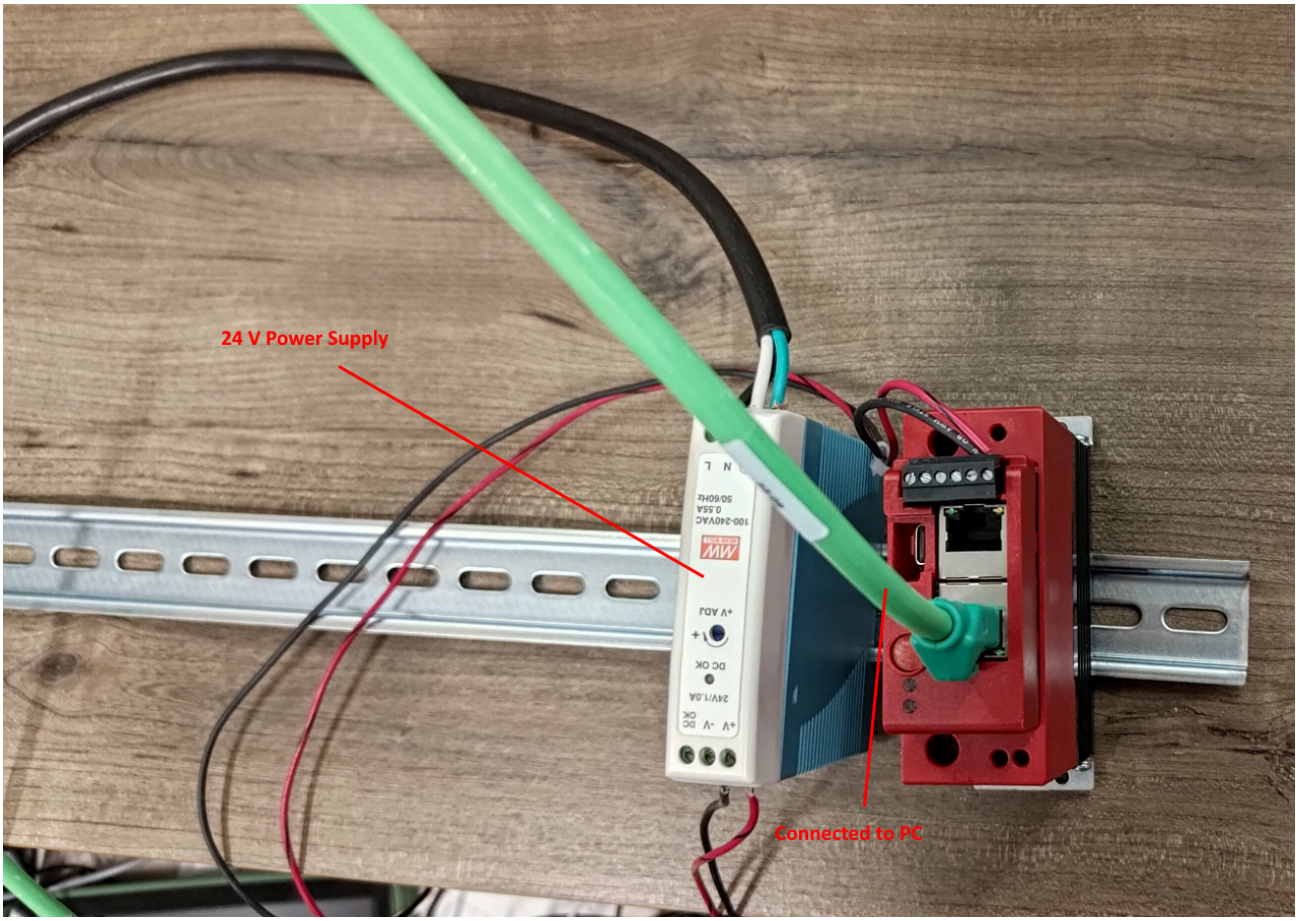
- Download ATOM_Codesys_LadderLogic_Example.zip
- Download ATOM_Codesys_StructuredText_Example.zip

## Prerequisites

1. Install Codesys
2. Download ATOM's EDS file

## Hardware setup

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



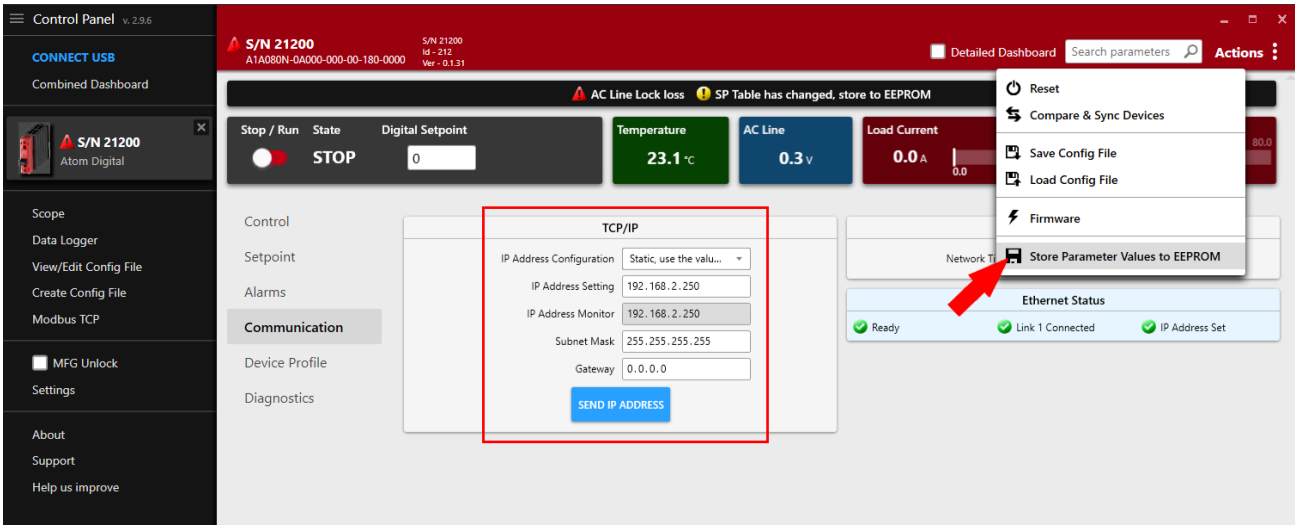

**INFO**

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.

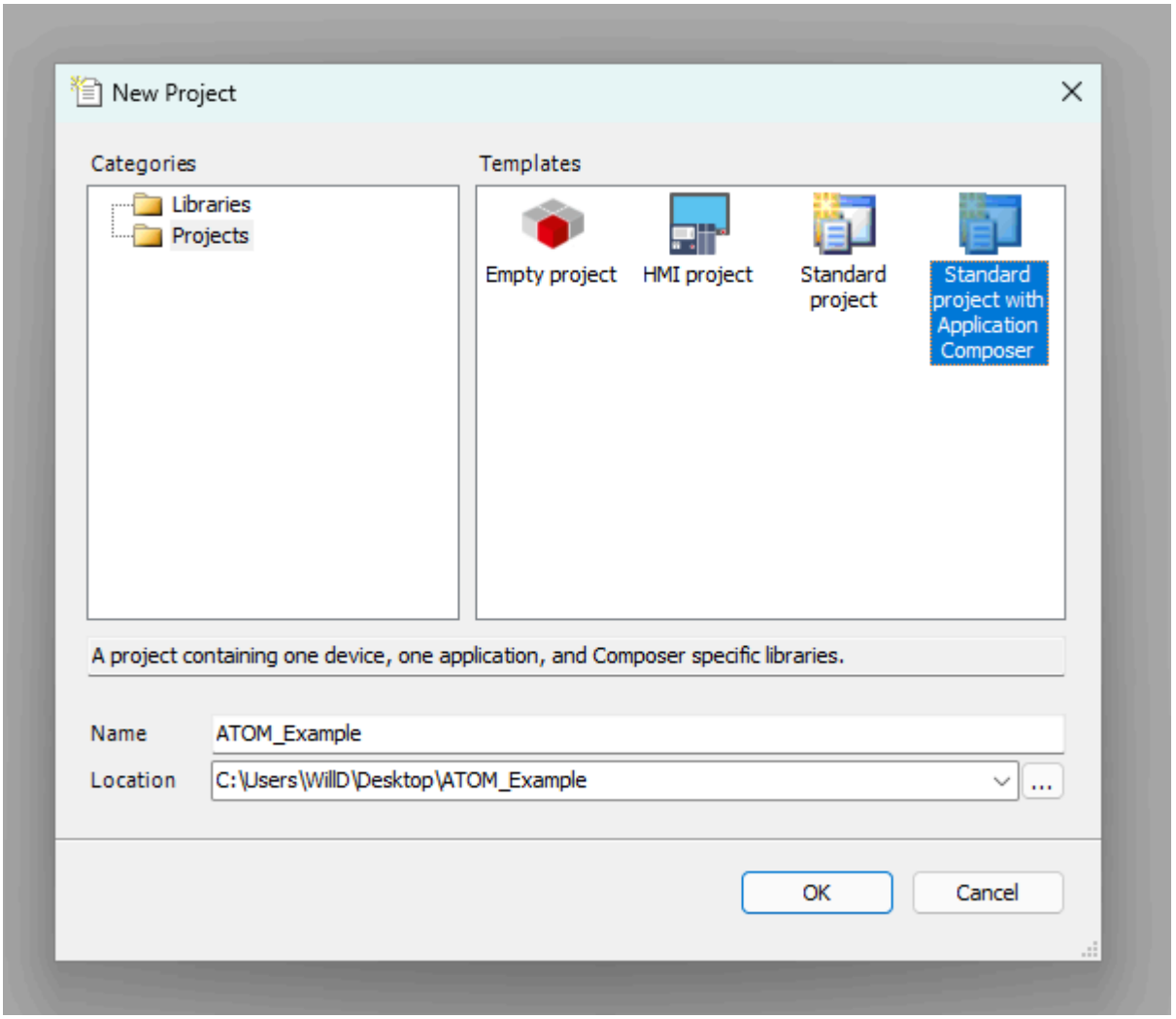
# Configuring Atom network settings

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**:

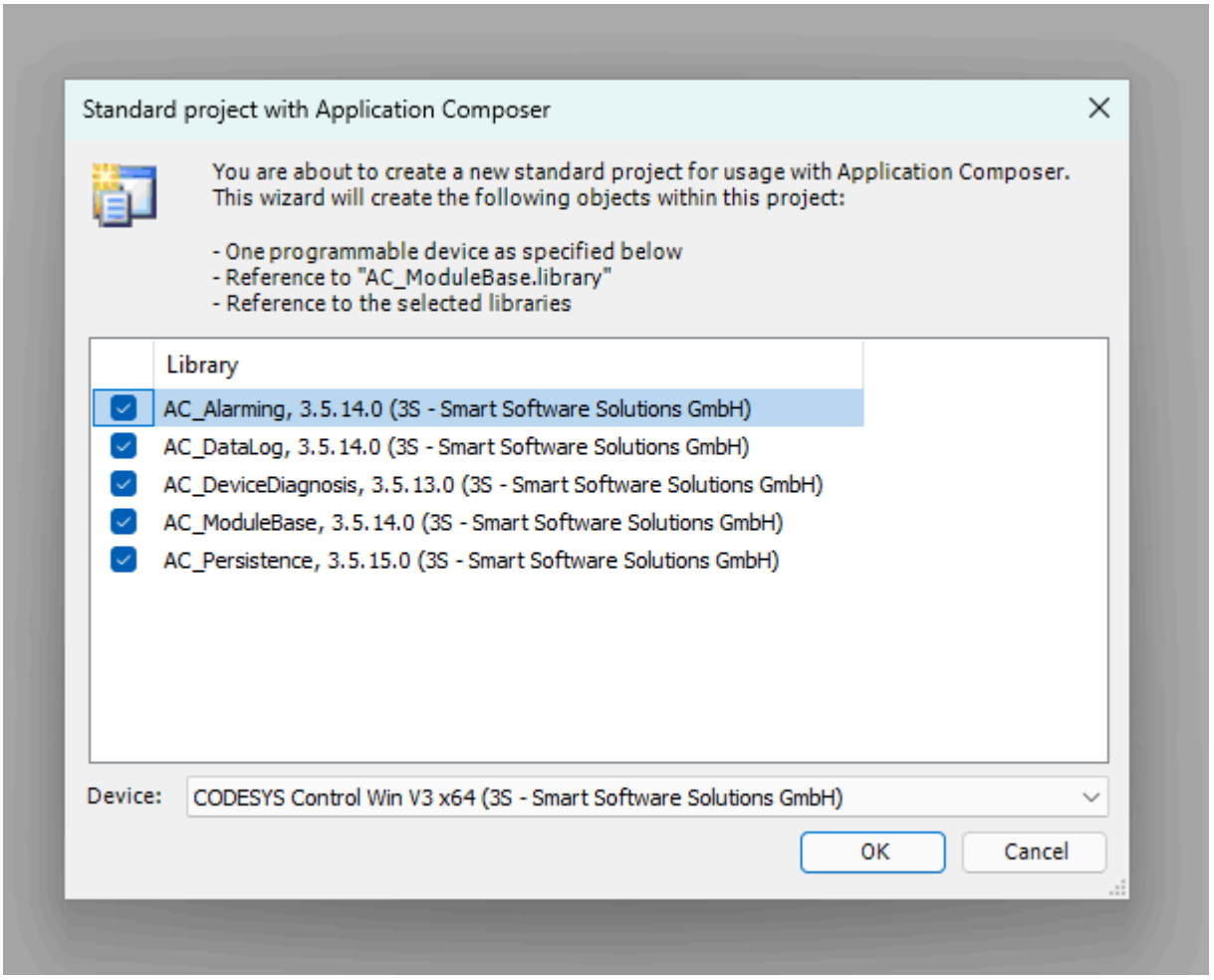



# Create a Codesys project

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



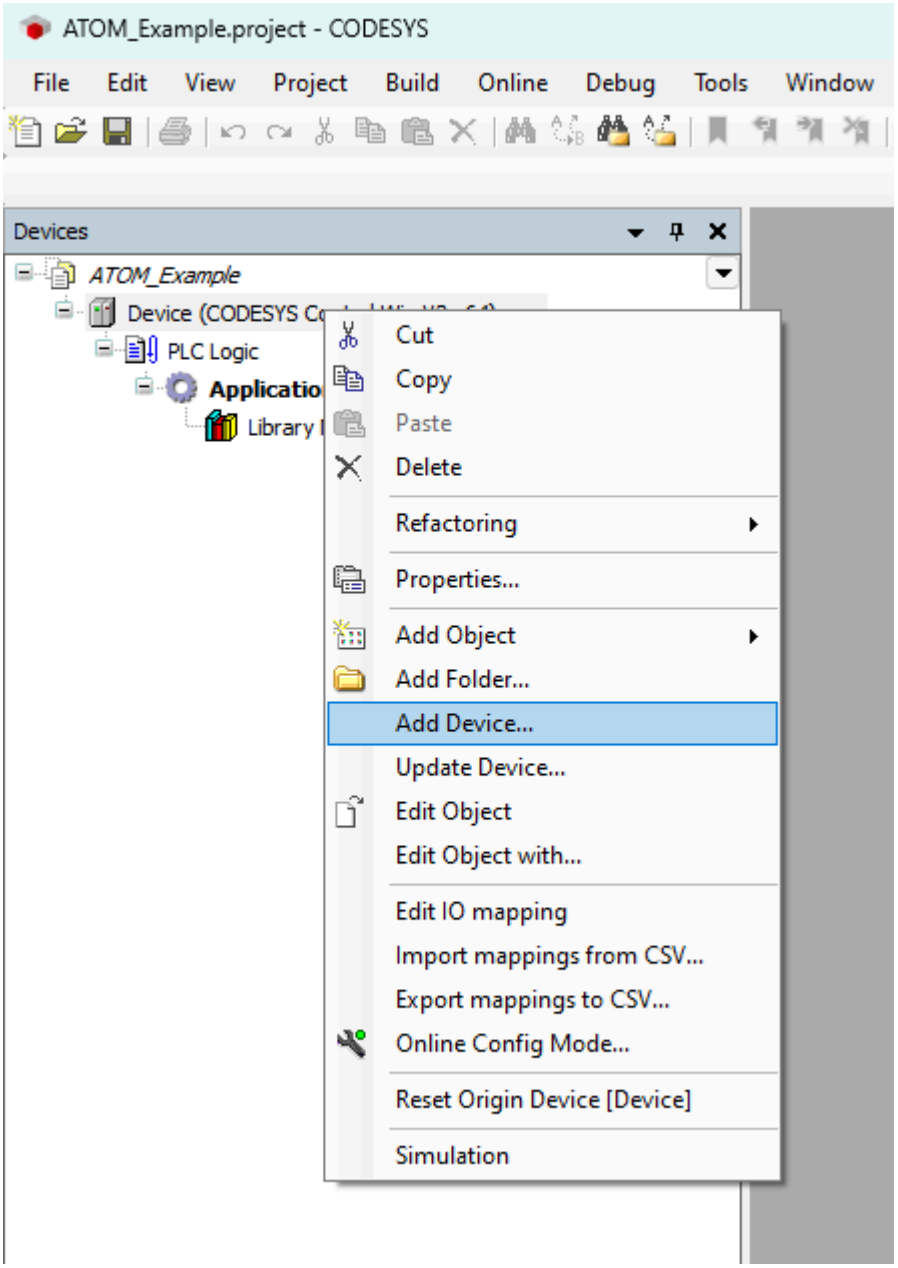

Check each library to include it in the project and select **CODESYS Control WIN V3 x64** as the device:



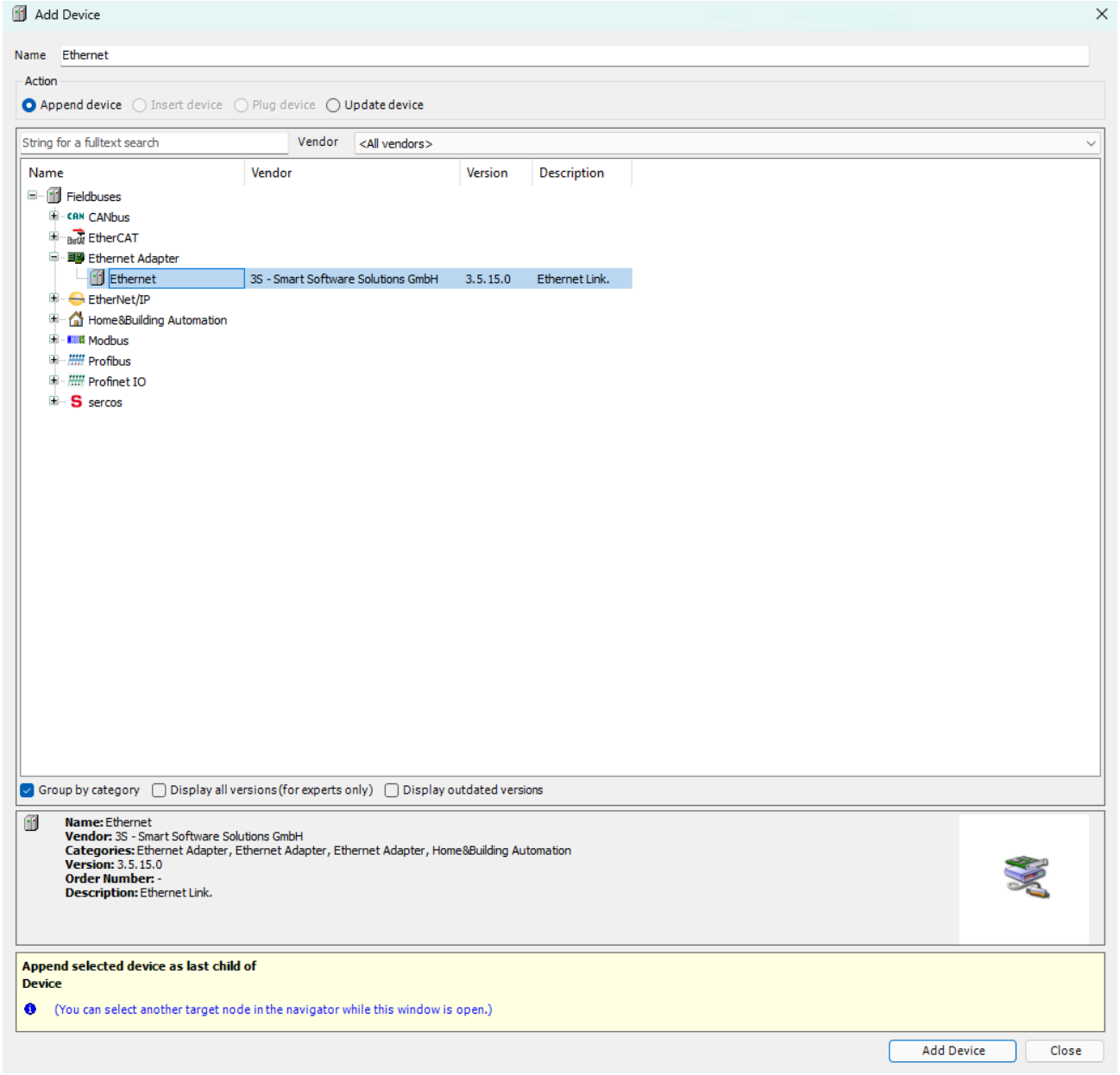

# Adding an EtherNet/IP Scanner

Next we'll add an EtherNet/IP Scanner module. This allows the PLC to discover EtherNet/IP devices on the network (in our case, ATOM) and establish a connection with them.

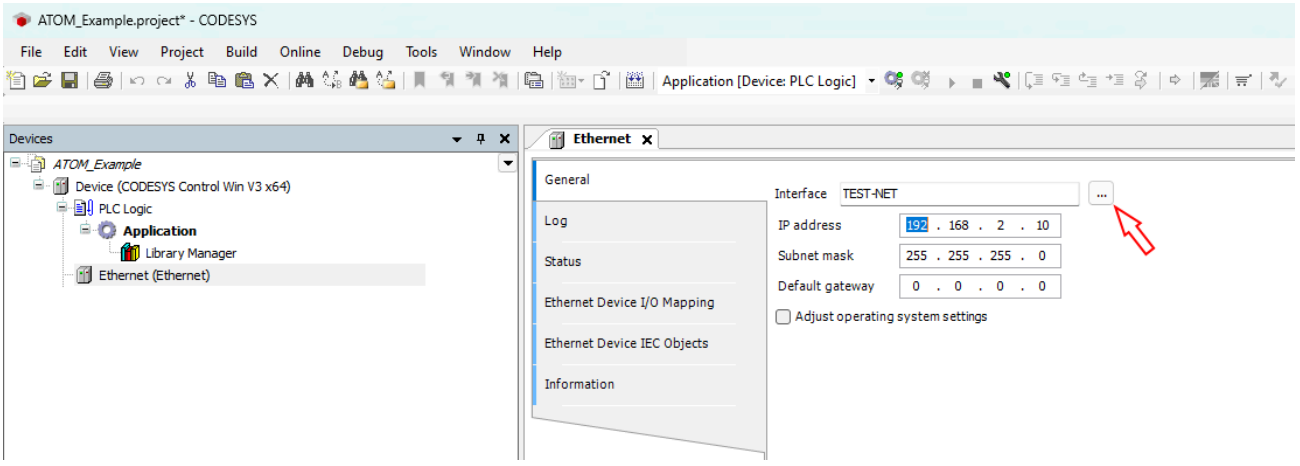
First, right click **Device** and select **Add Device**:



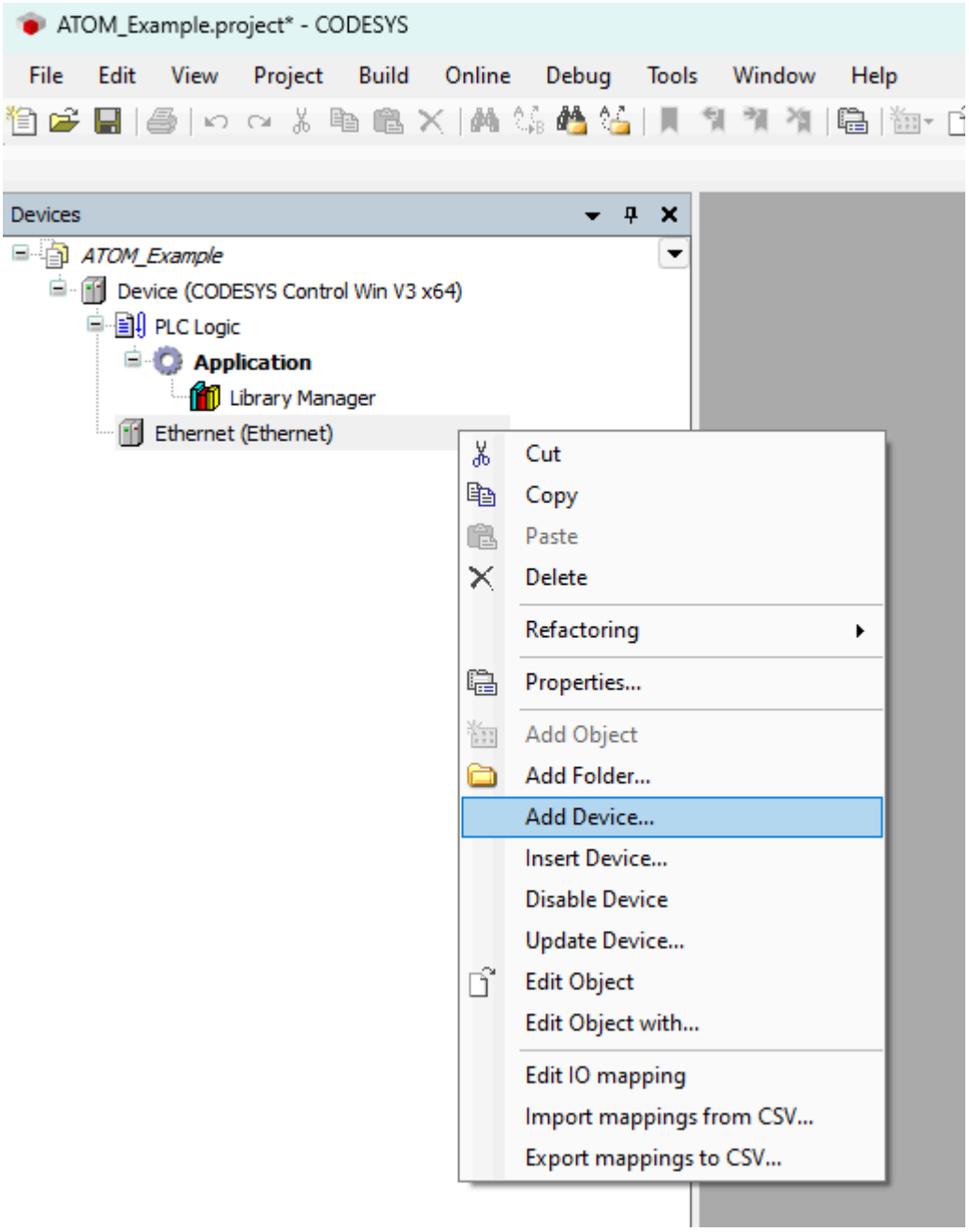

Next, expand **Ethernet Adapter** and select **Ethernet**, then click **Add Device**:



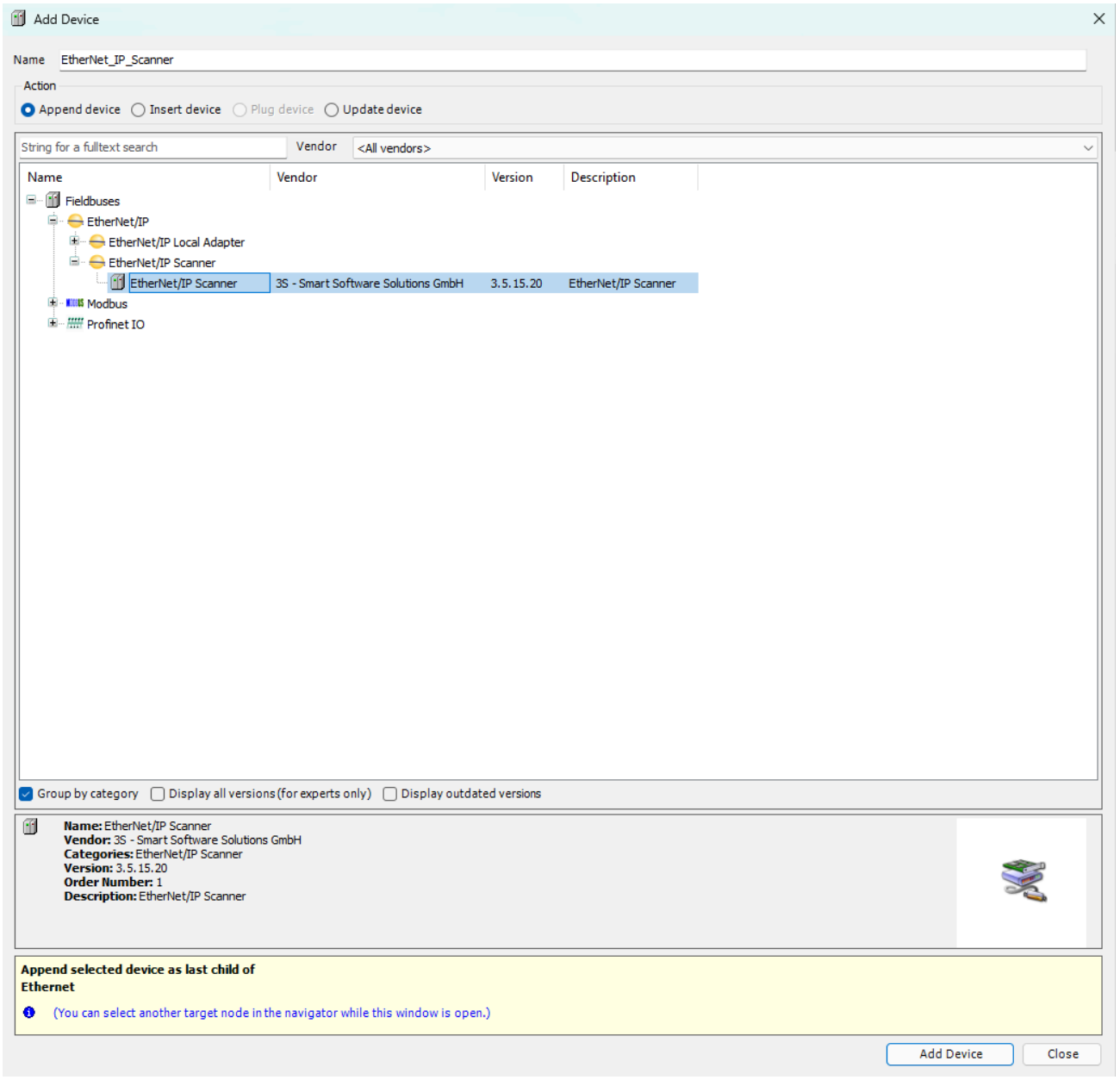

The newly added **Ethernet** device will now appear in the device tree. Double click **Ethernet (Ethernet)** to open its configuration tab. Within the **General** configuration tab, use the button indicated by the red arrow to select the network interface of the host machine that will be used to communicate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




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



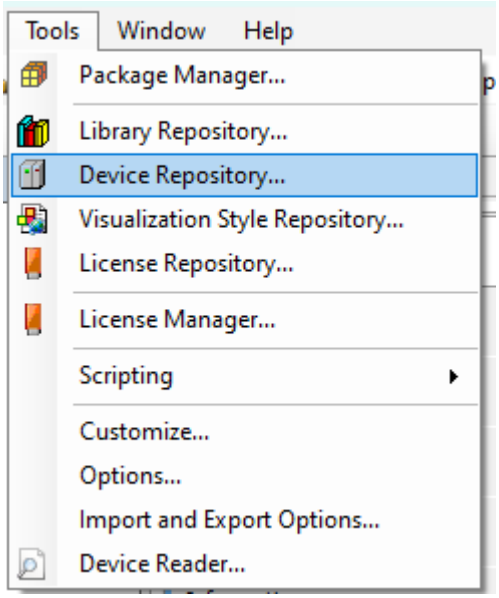

Expand **EtherNet/IP Scanner**, select **EtherNet/IP Scanner**, then click **Add Device**:



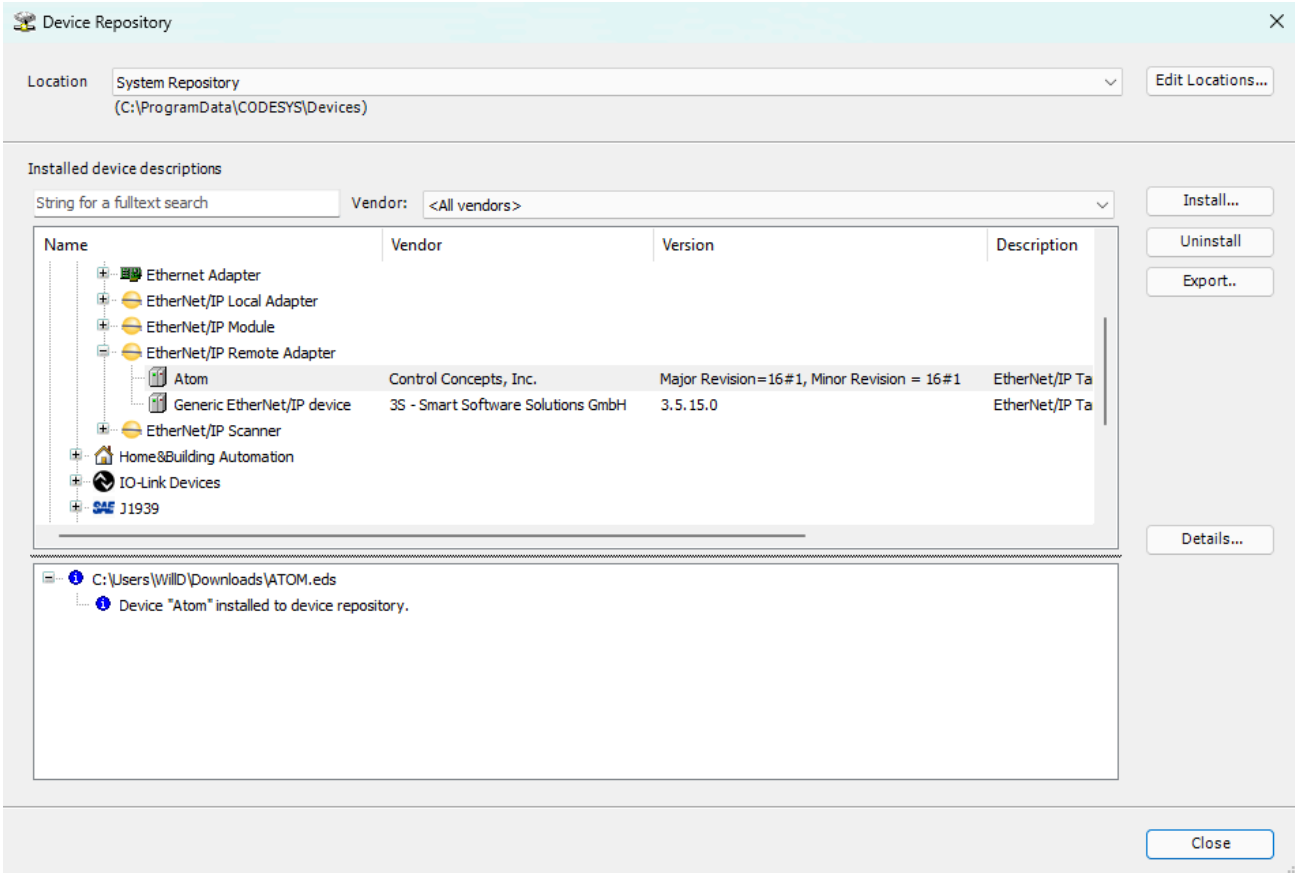

Your device tree should update to include the **EtherNet/IP Scanner** device.

# Adding ATOM to the scanner

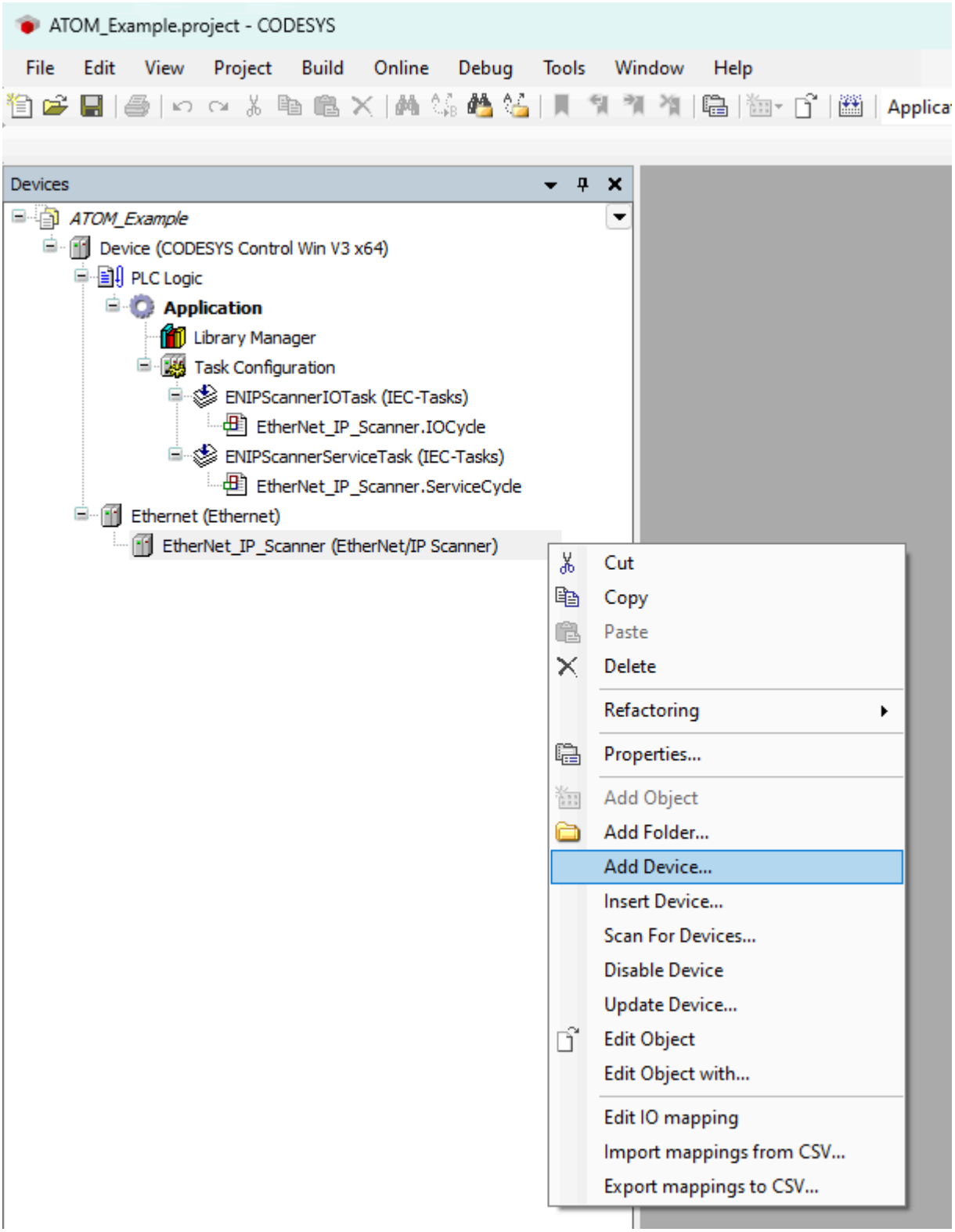
First, we'll import ATOM's EDS file you downloaded earlier into our Codesys device library. Open the tools menu and select **Device repository**:



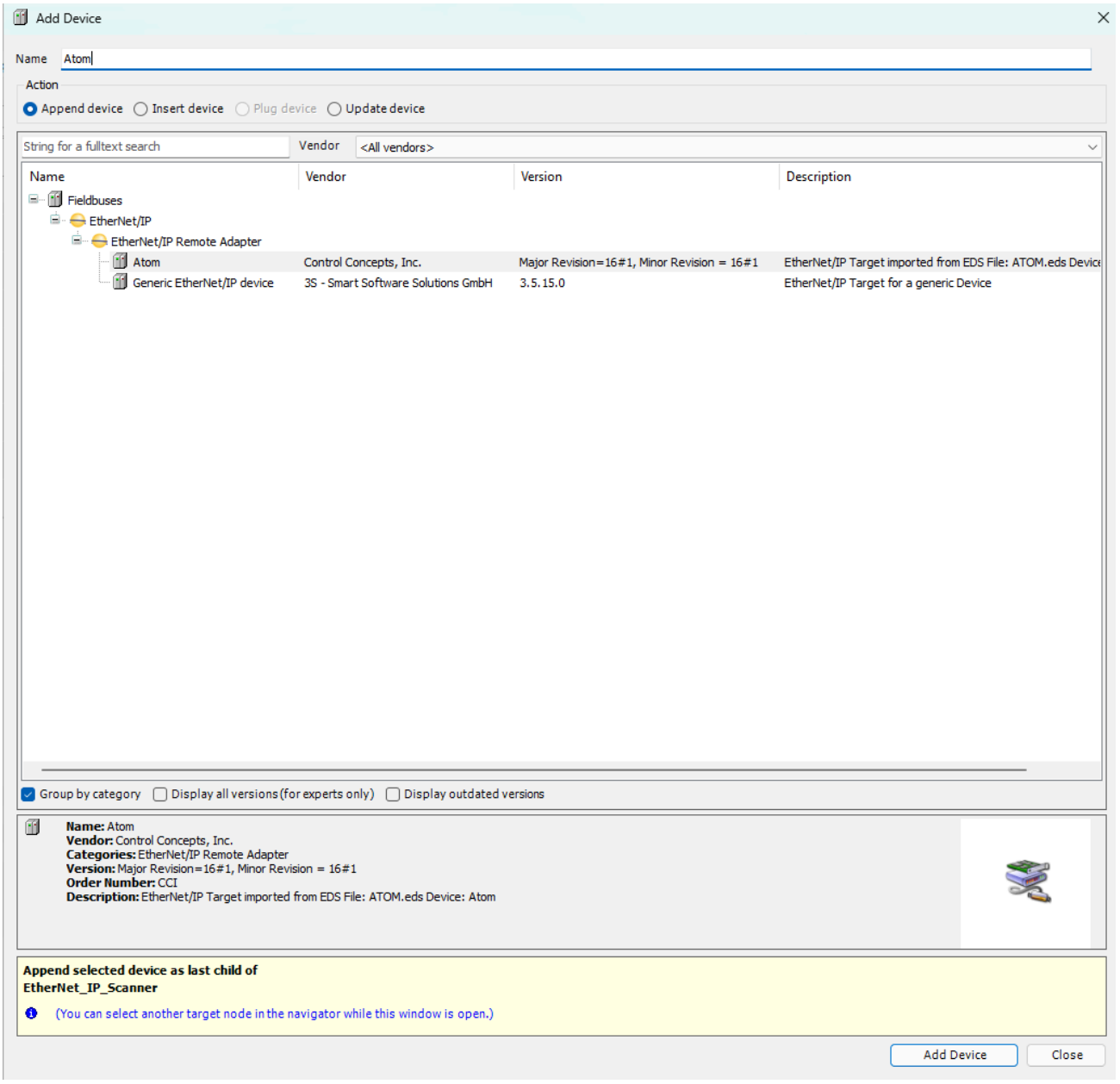

Next, click **Install** and select the `ATOM.eds` file. After you click install, **Atom** will appear under the **EtherNet/IP Remote Adapter** category. Click **Close** to dismiss the dialog:



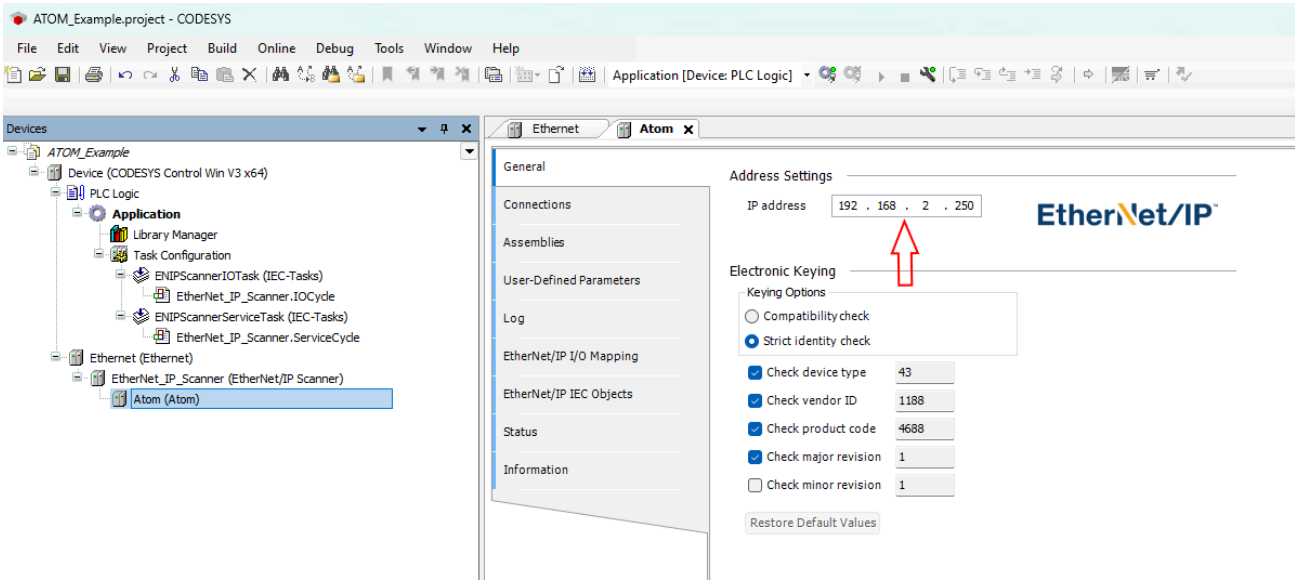
Now, we'll add ATOM to the scanner. Right click **EtherNet/IP Scanner (EtherNet/IP Scanner)** and select **Add Device**:




Expand **EtherNet/IP Remote Adapter** and select **Atom**, then click **Add Device**:




Finally, double click **Atom (Atom)** to open its configuration tab. In the **General** tab, set the **IP Address** to the IP address of your ATOM device:




# Create a program

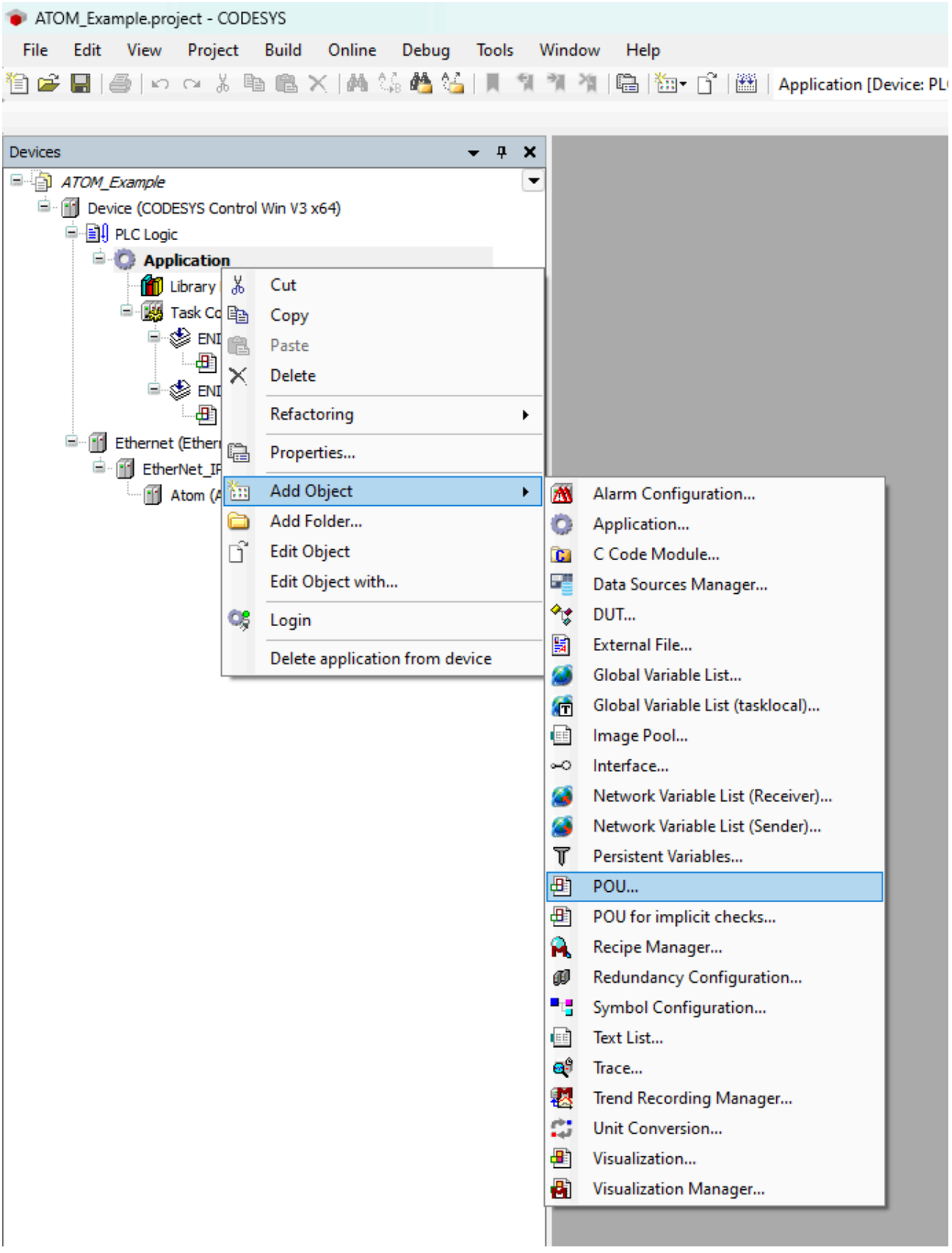
Next, we'll create a PLC program. We provide examples for both ladder logic and structured text:

- Program with ladder logic
- Program with structured text

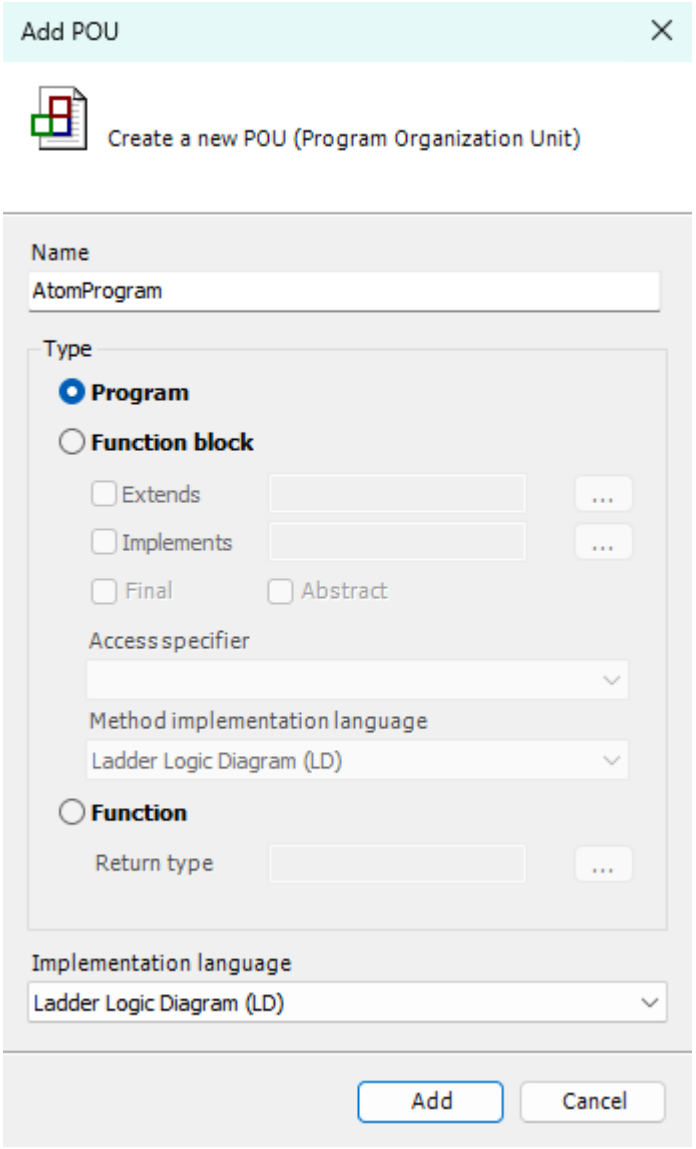
# Example: Ladder logic

## Creating the program

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


ATOM_Example.project - CODESYS
File Edit View Project Build Online Debug Tools Window Help
Application [Device: PL
Devices
ATOM_Example
Device (CODESYS Control Win V3 x64)
PLC Logic
Application
Library
Task Co
ENI
ENI
Ethernet (Ether
EtherNet_IP
Atom (A
Cut
Copy
Paste
Delete
Refactoring
Properties...
Add Object
Add Folder...
Edit Object
Edit Object with...
Login
Delete application from device
Alarm Configuration...
Application...
C Code Module...
Data Sources Manager...
DUT...
External File...
Global Variable List...
Global Variable List (tasklocal)...
Image Pool...
Interface...
Network Variable List (Receiver)...
Network Variable List (Sender)...
Persistent Variables...
POU...
POU for implicit checks...
Recipe Manager...
Redundancy Configuration...
Symbol Configuration...
Text List...
Trace...
Trend Recording Manager...
Unit Conversion...
Visualization...
Visualization Manager...

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



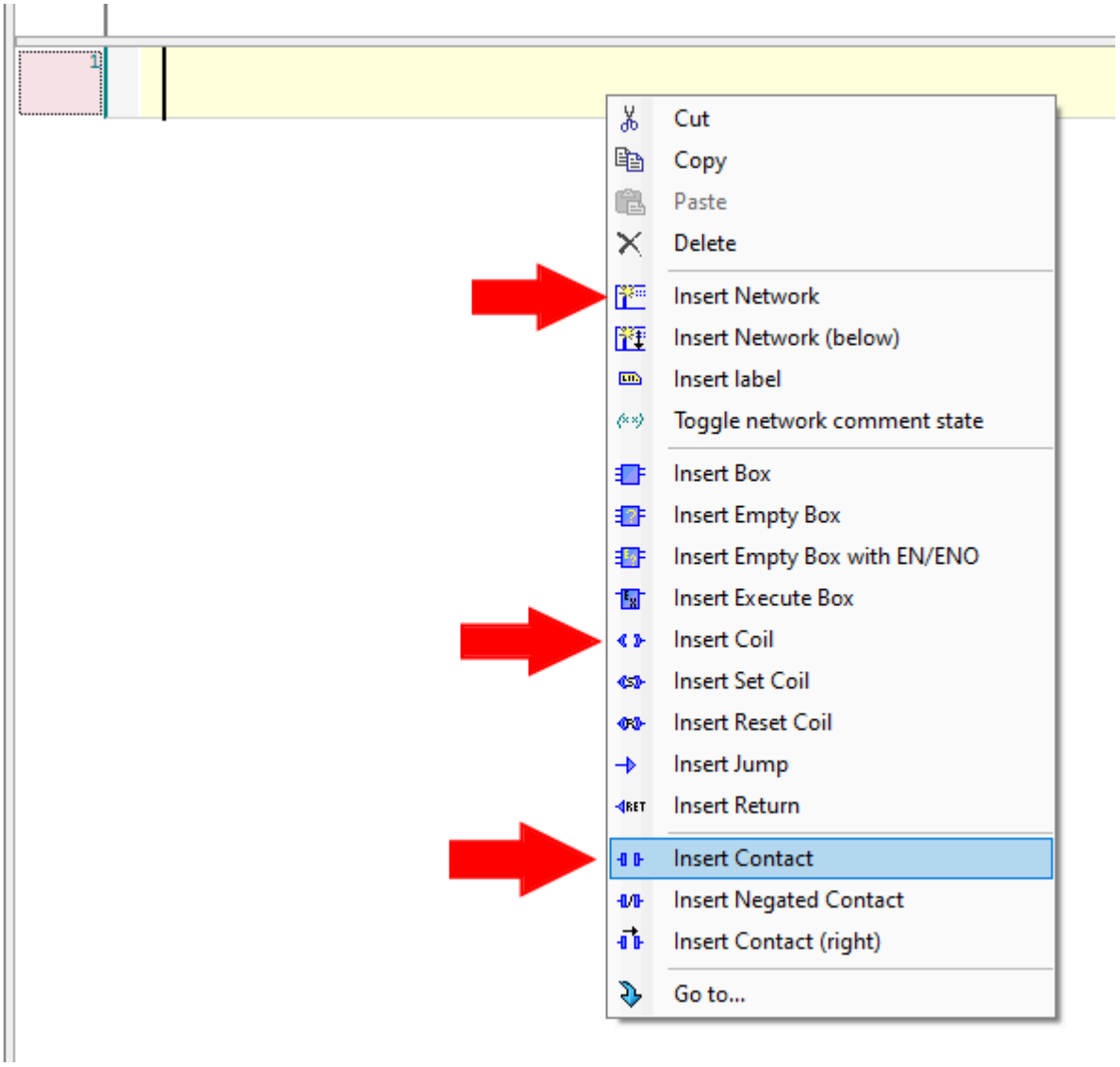

Copy the following code into the top panel of the **AtomProgram** editor:

```
PROGRAM AtomProgram
VAR

RUN_SWITCH: BOOL;
SETPOINT: DINT;
TEMP: REAL;

ATOM_OUTPUT_SETPOINT: DINT;
ATOM_OUTPUT_RUN_ENABLE: BOOL;
ATOM_INPUT_TEMP: REAL;

END_VAR
```

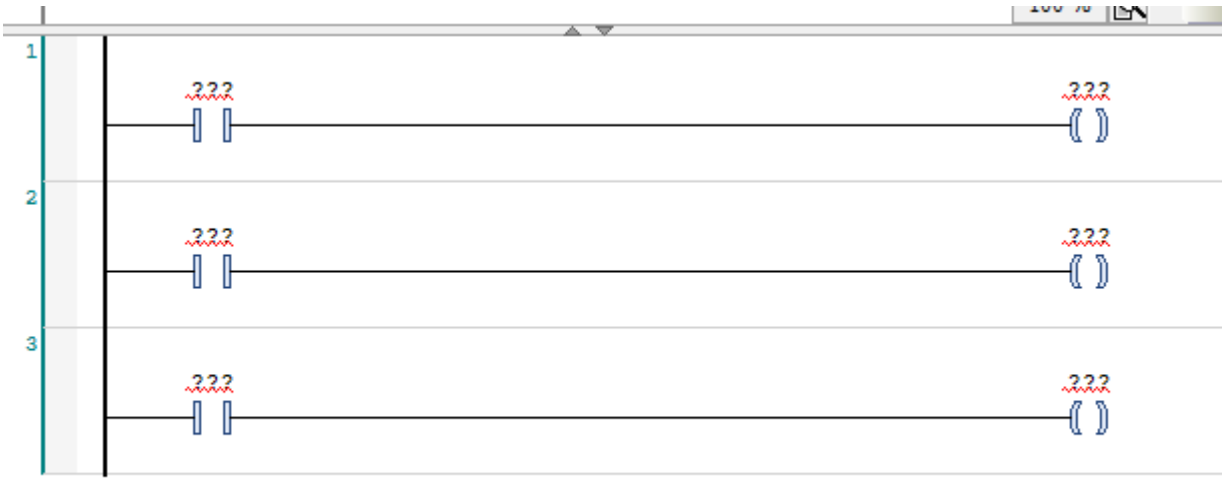
After you've copied the code over, the editor for **AtomProgram** should look like this:


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**
2. For each network, right click and insert **one** contact and **one** coil



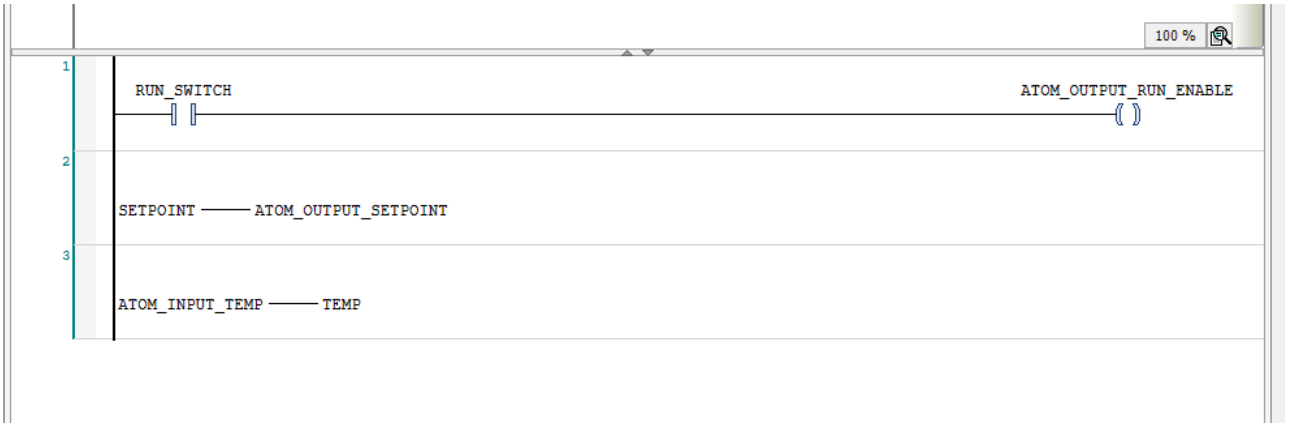

After you're finished, your ladder logic program should look like:




For each rung, replace the `???` with the corresponding variables:

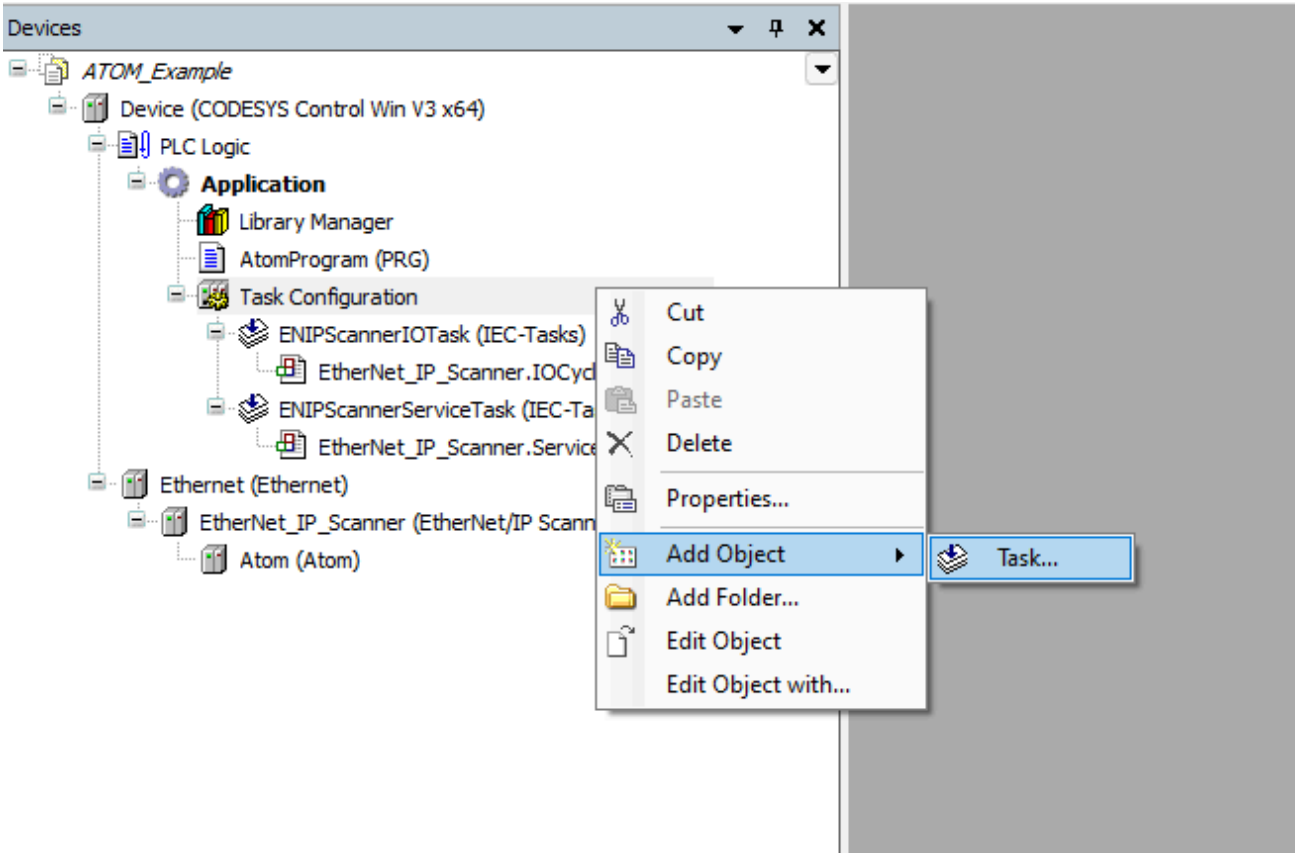
1. **Rung #1** - `RUN_SWITCH` and `ATOM_OUTPUT_RUN_ENABLE`
2. **Rung #2** - `SETPOINT` and `ATOM_OUTPUT_SETPOINT`
3. **Rung #3** - `ATOM_INPUT_TEMP` and `TEMP`

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

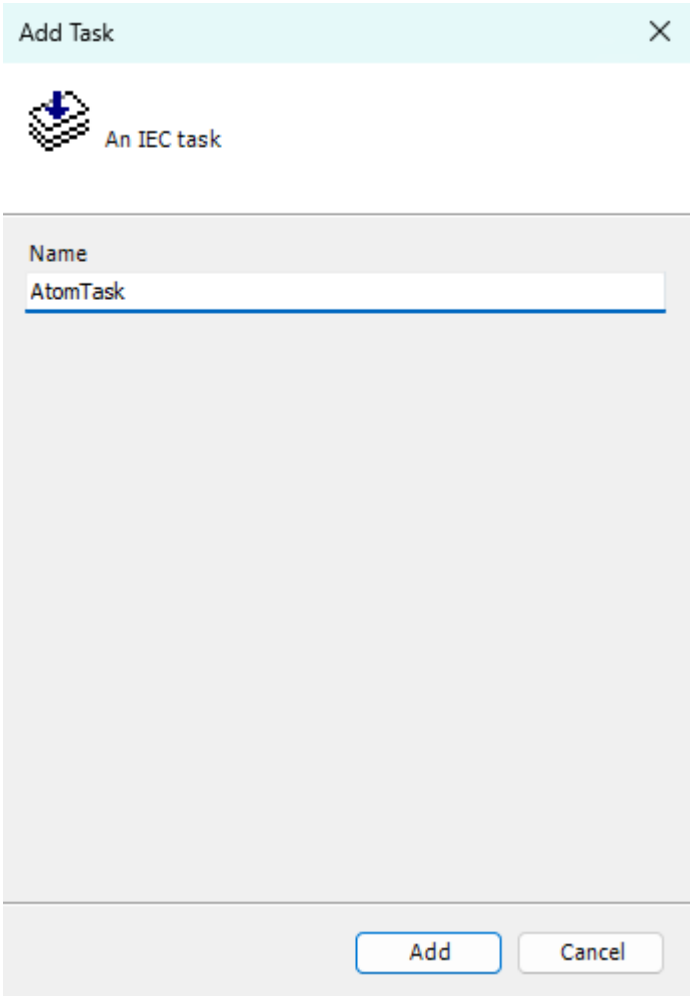



Finally, we'll add a task to call **AtomProgram** from the PLC's control loop:

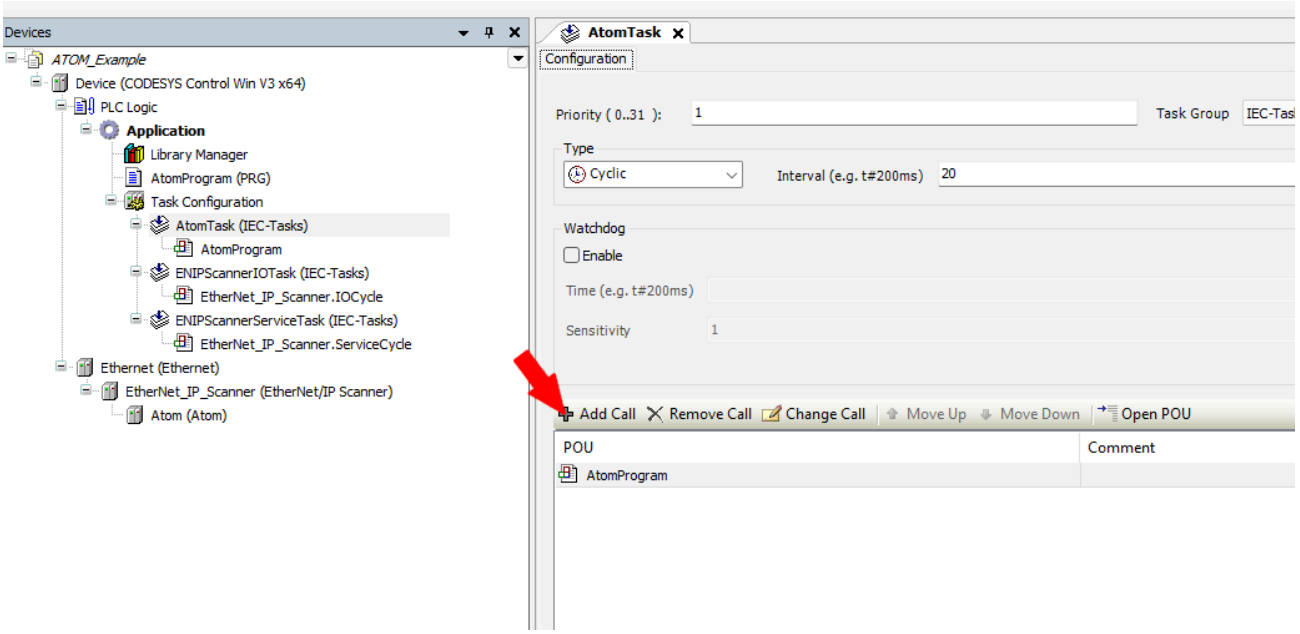
Right click **Task Configuration** and select **Add Object > Task**:




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



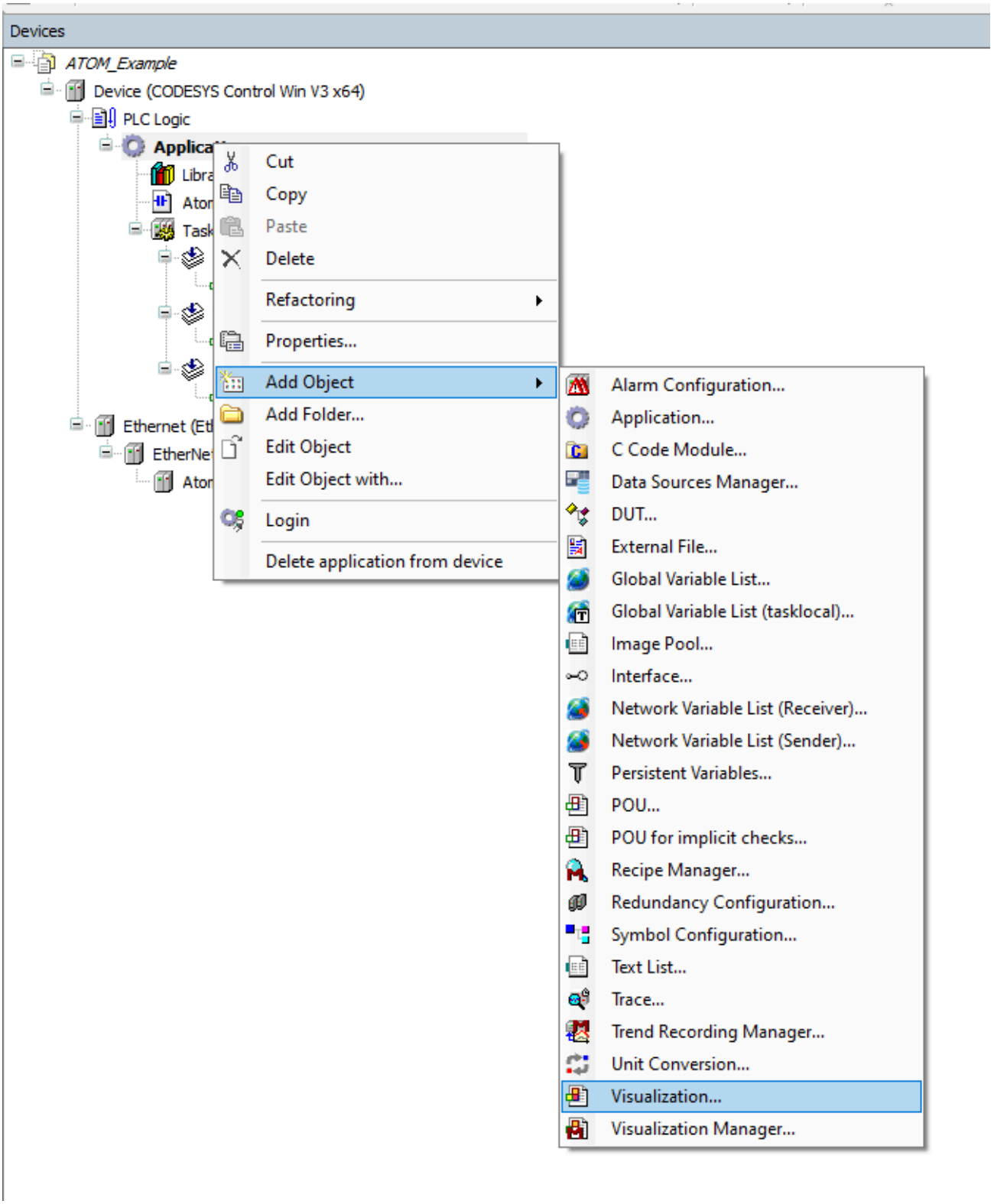

Next, double click **AtomTask (IEC-Tasks)** to open its configuration tab. Click **Add Call** and select **Application > AtomProgram**. After doing so, AtomTask's configuration should look like:



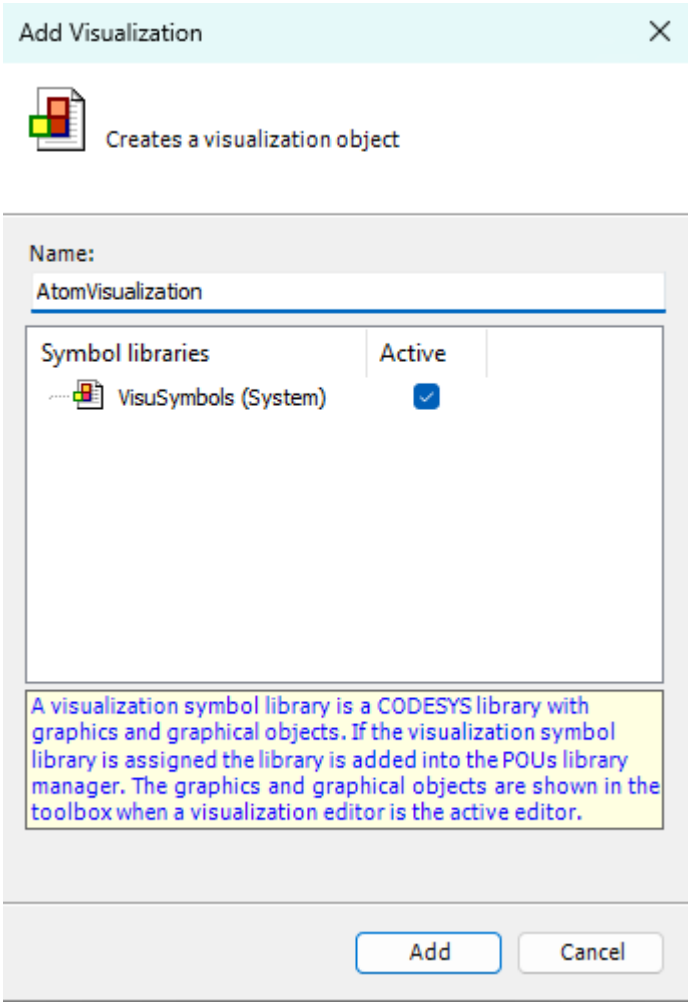

# Setting up visualization

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

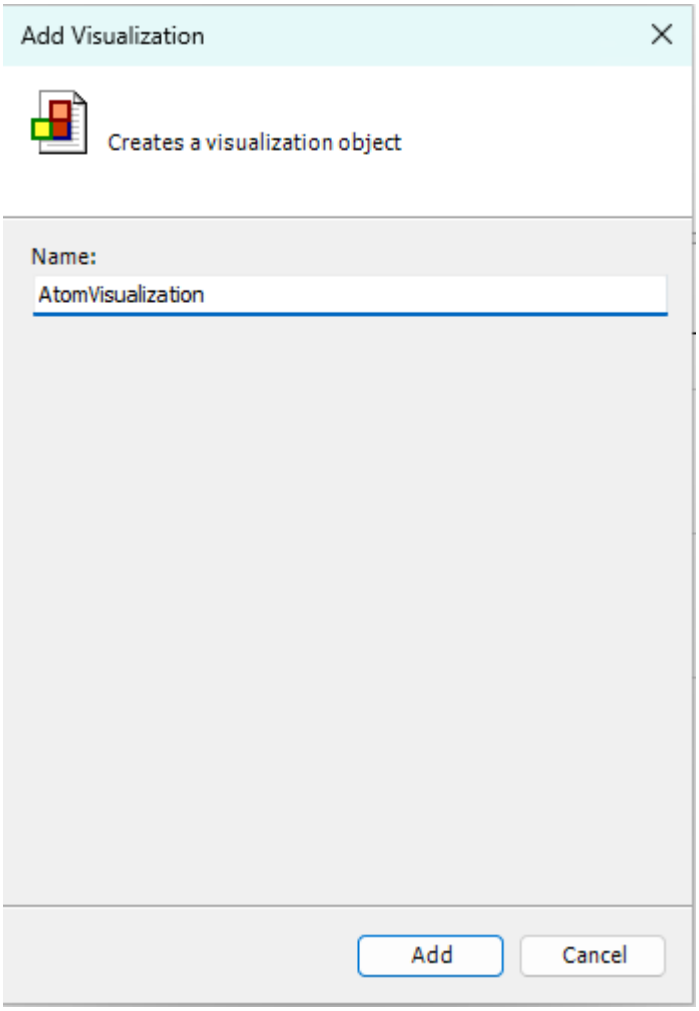
Right click **Application** and select **Add Object > Visualization**:



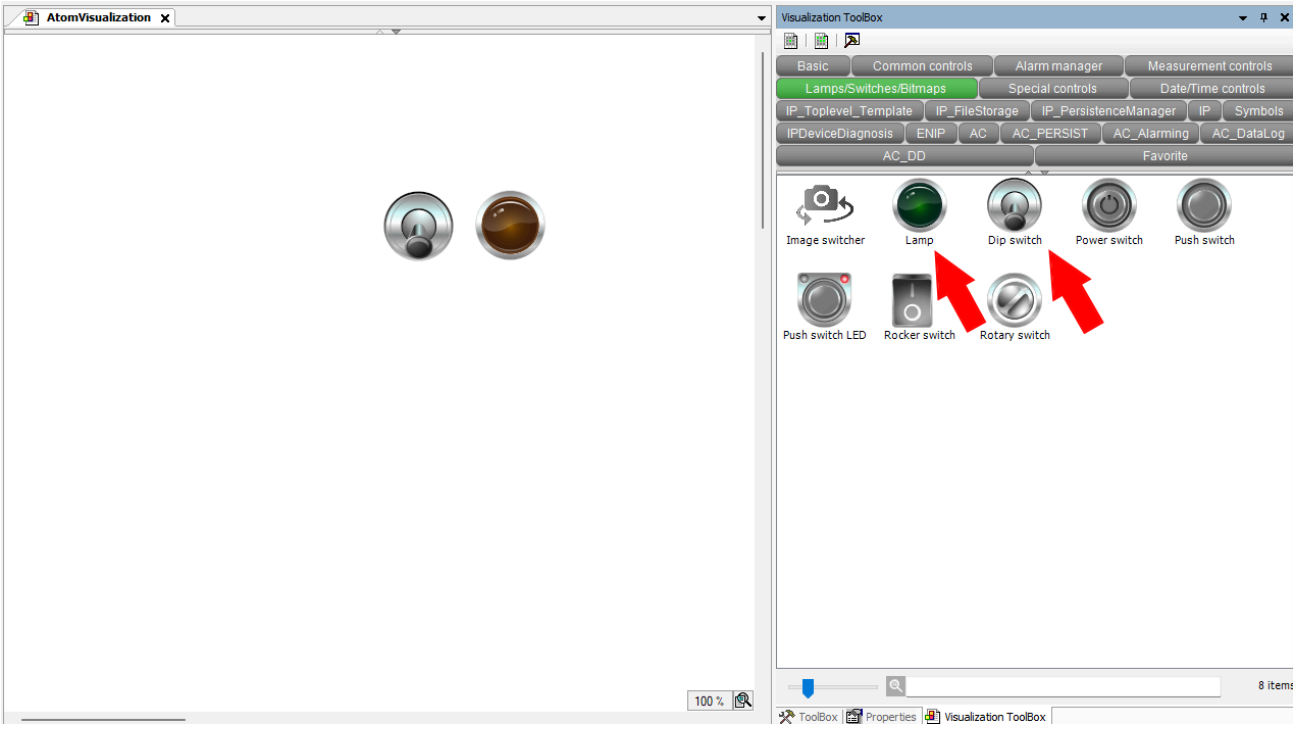

Make sure to check **Active** for **VisuSymbols (System)**, then click **Add**:



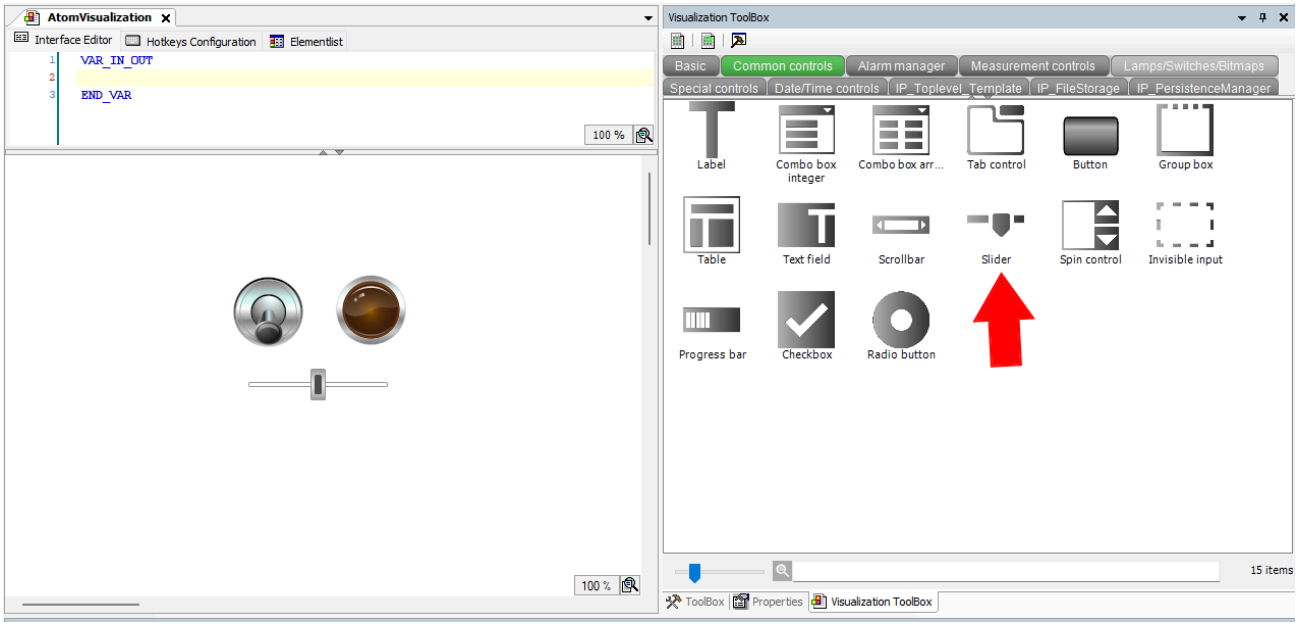

Name your visualization `AtomVisualization` and click **Add**:



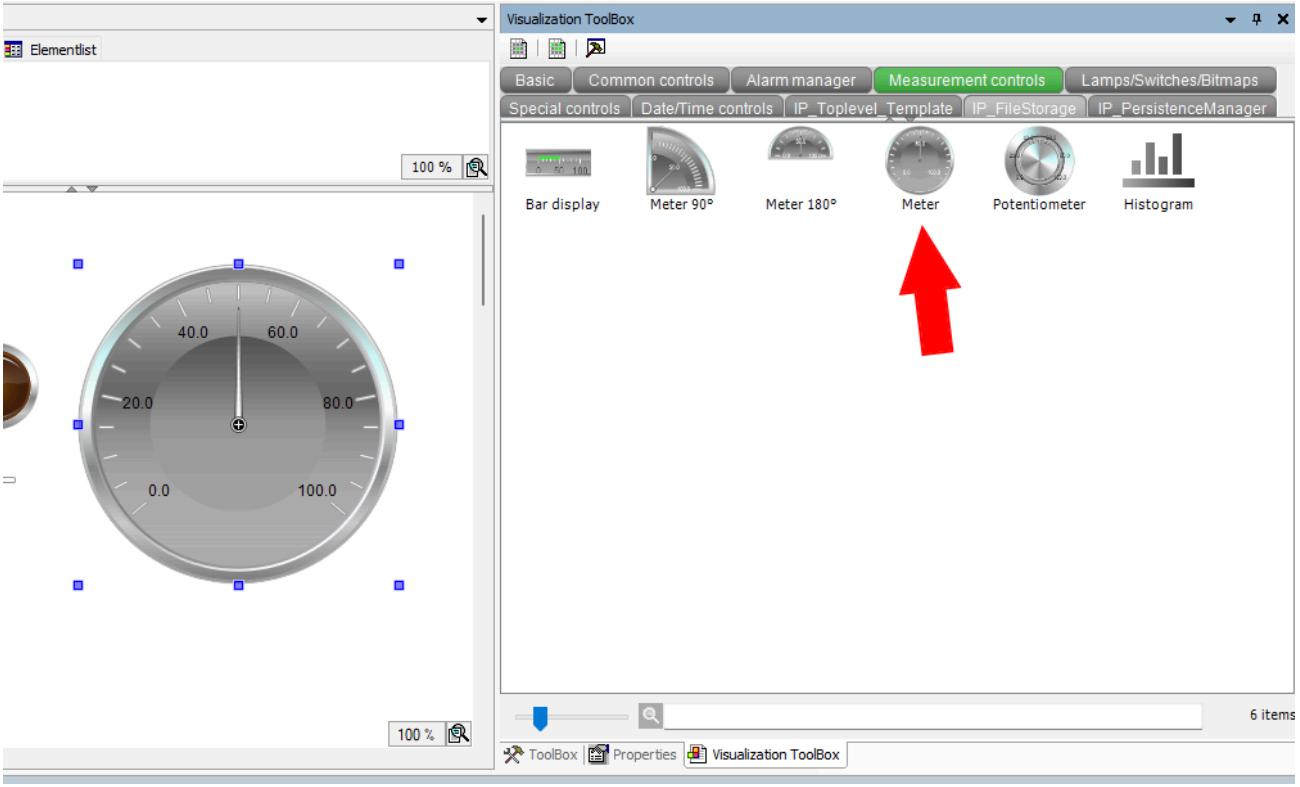

Double click **AtomVisualization** to open its configuration editor. From the **Visualization ToolBox** panel on the right, select the **Lamps/Switches/Bitmaps** category and add a lamp and a dip switch:




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

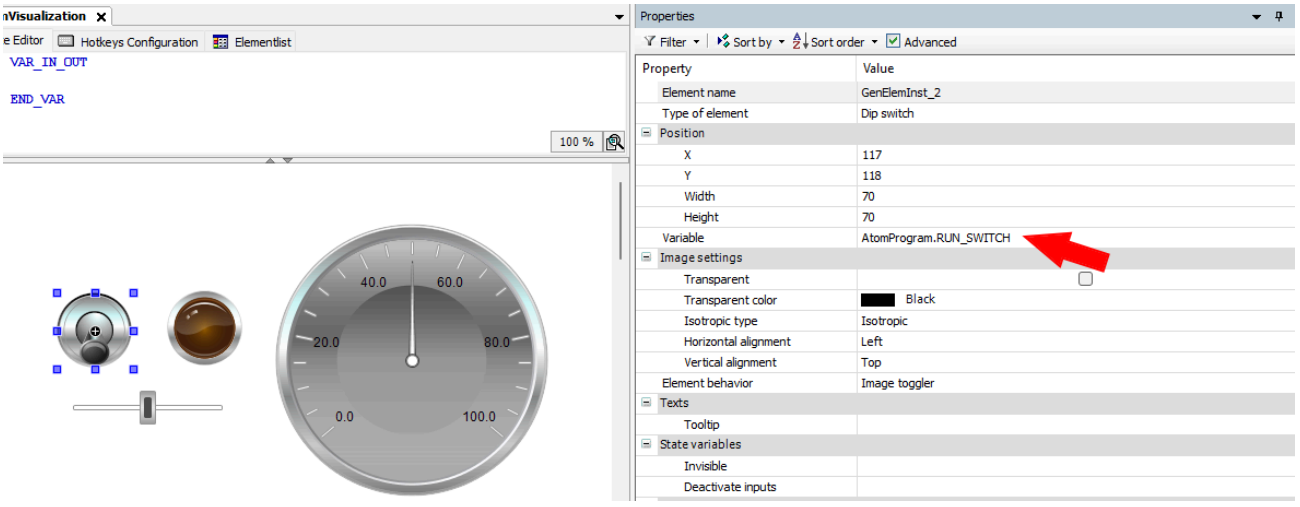



Finally, in the **Measurement controls** category, add a meter:

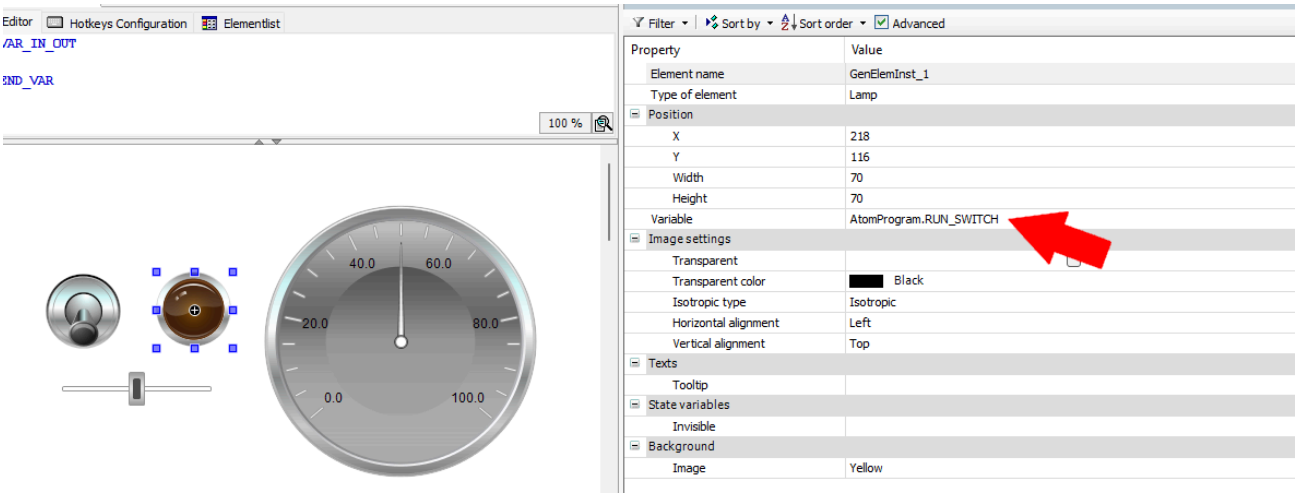



# Wiring up the controls

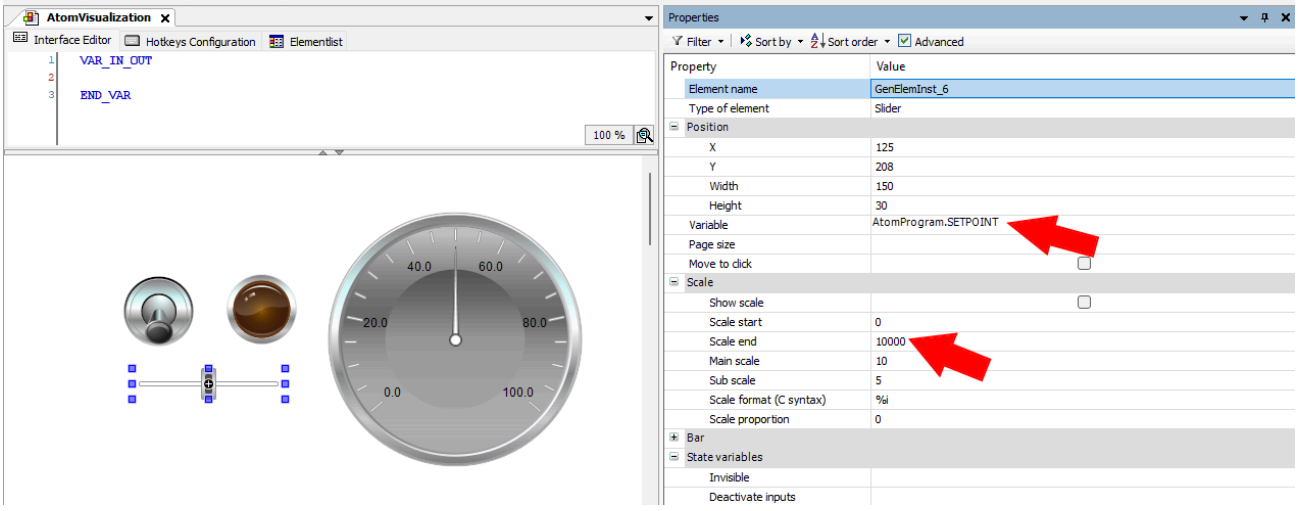
Next, we'll connect the controls to our PLC program. Select the dip switch and set the **Variable** field to `AtomProgram.RUN_SWITCH` as indicated by the red arrow:



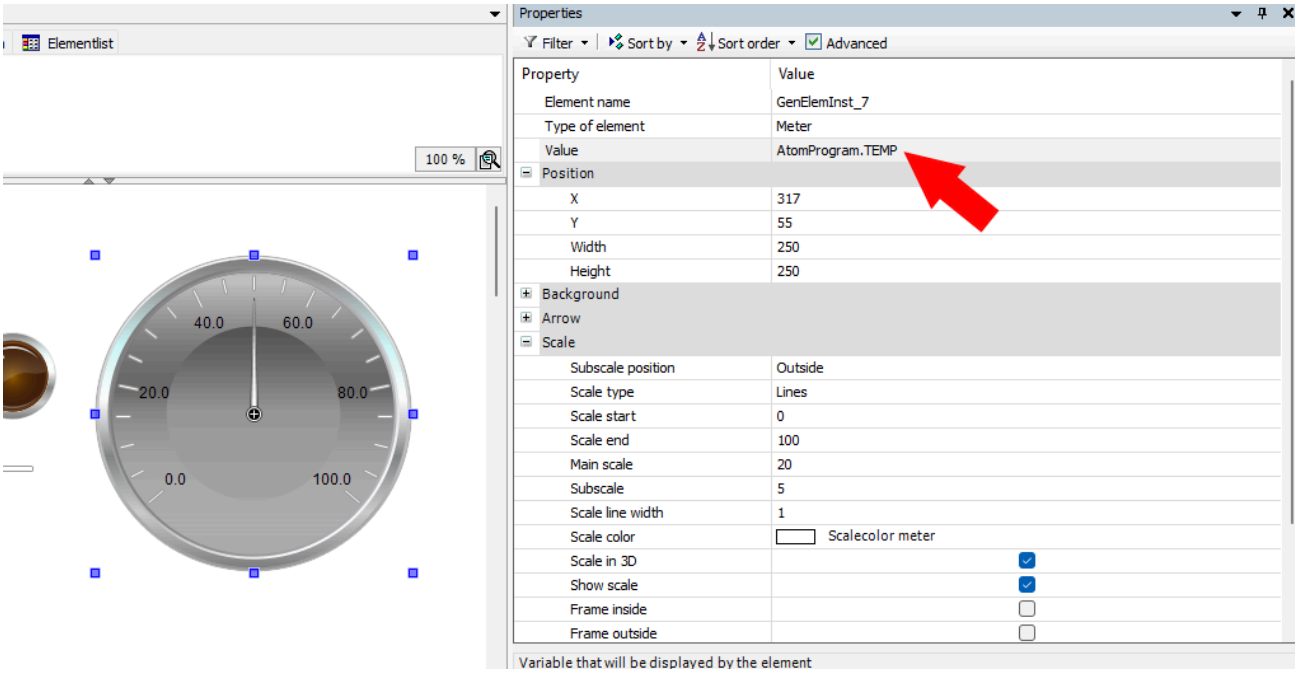
Select the lamp and set the **Variable** field to `AtomProgram.RUN_SWITCH` as indicated by the red arrow:




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

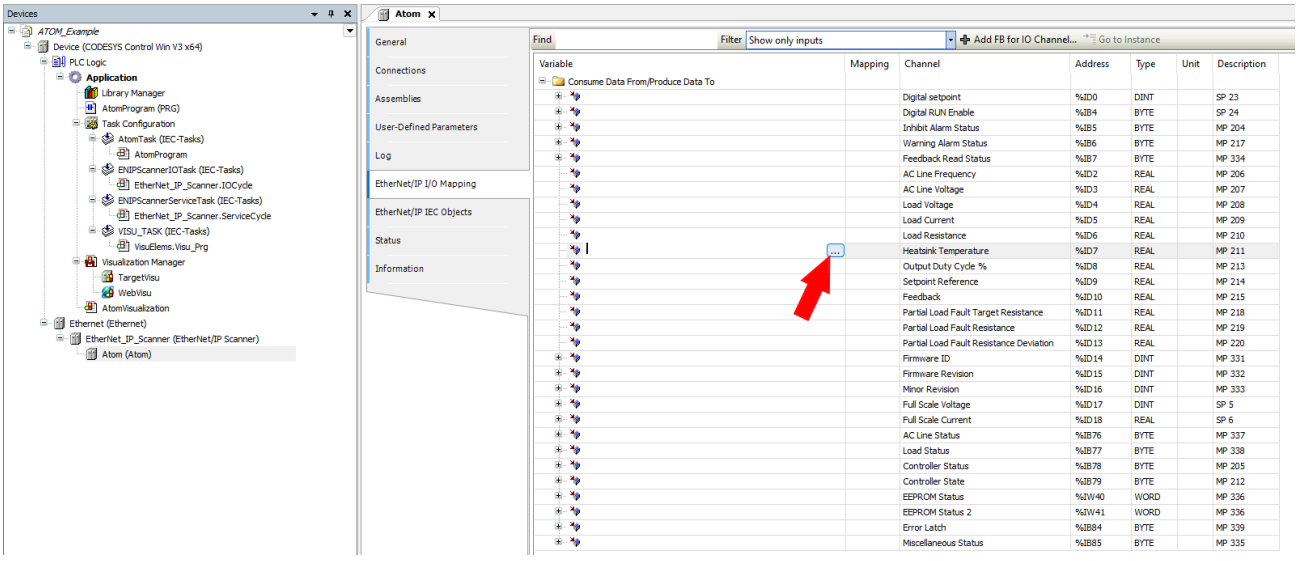



Select the meter and set the **Variable** field to `AtomProgram.TEMP`:

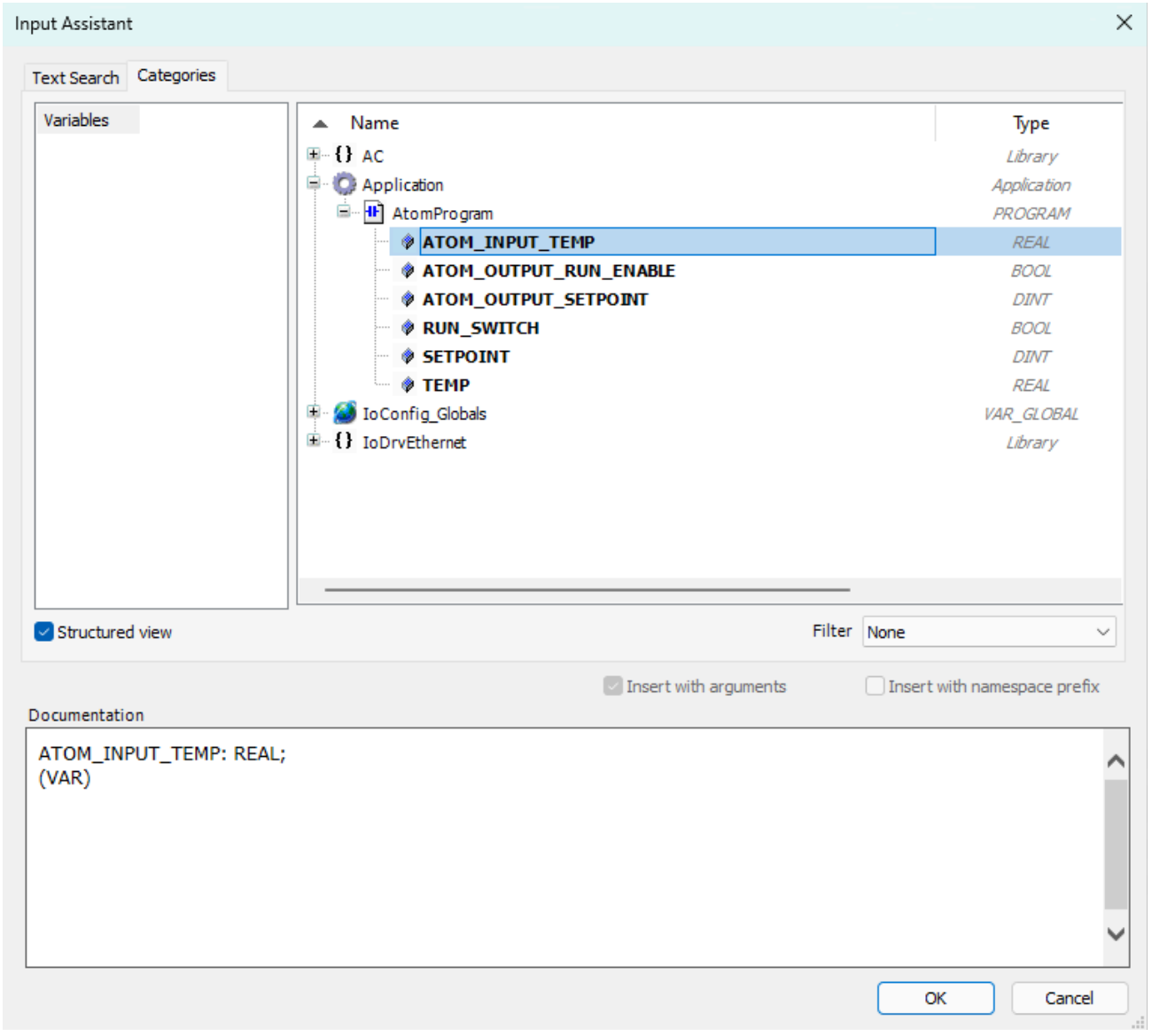



# Mapping variables

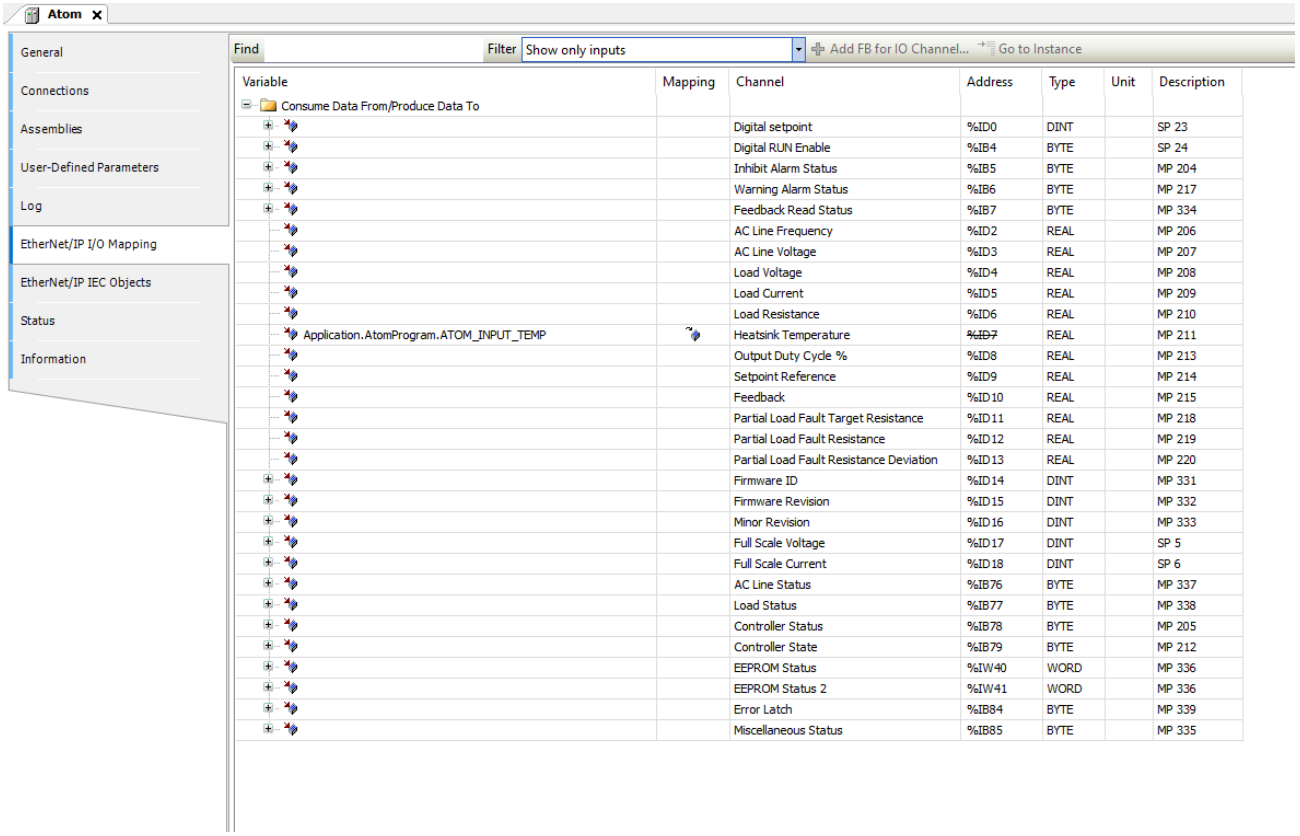
Finally, we'll map our PLC variables to ATOM. Double click **Atom** in the device tree to open its configuration window. Select the **EtherNet/IP I/O Mapping** tab and set **Filter** to **Show only inputs**:



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**:

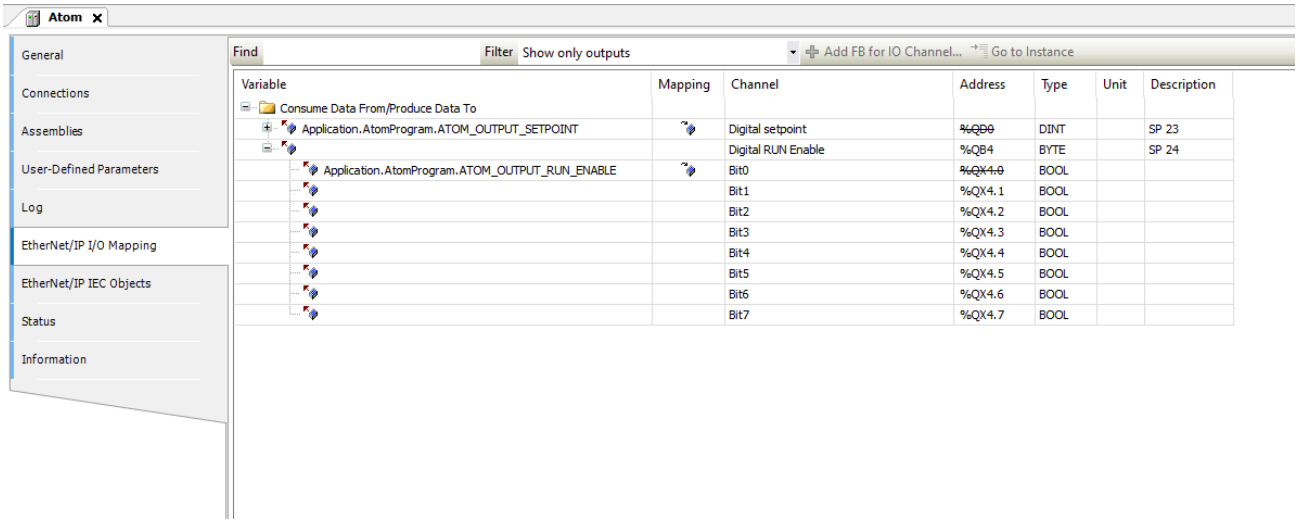



After doing so, your input I/O mappings should look like:




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`.

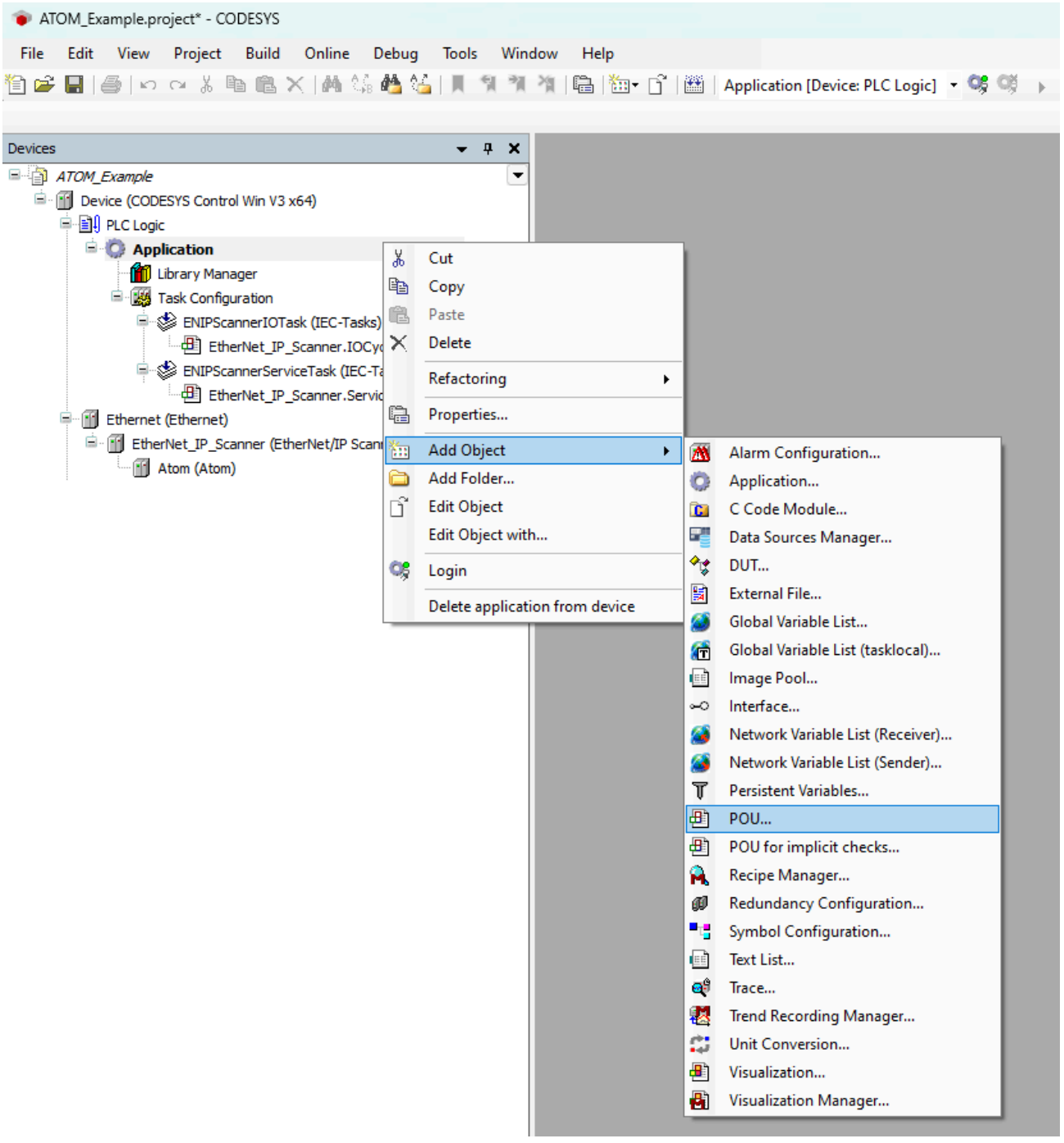
> **TAKE CARE**
>
> Make sure you map **Bit0** of **Digital RUN Enable** to **ATOM_OUTPUT_RUN_ENABLE**, NOT **Digital RUN Enable** itself.



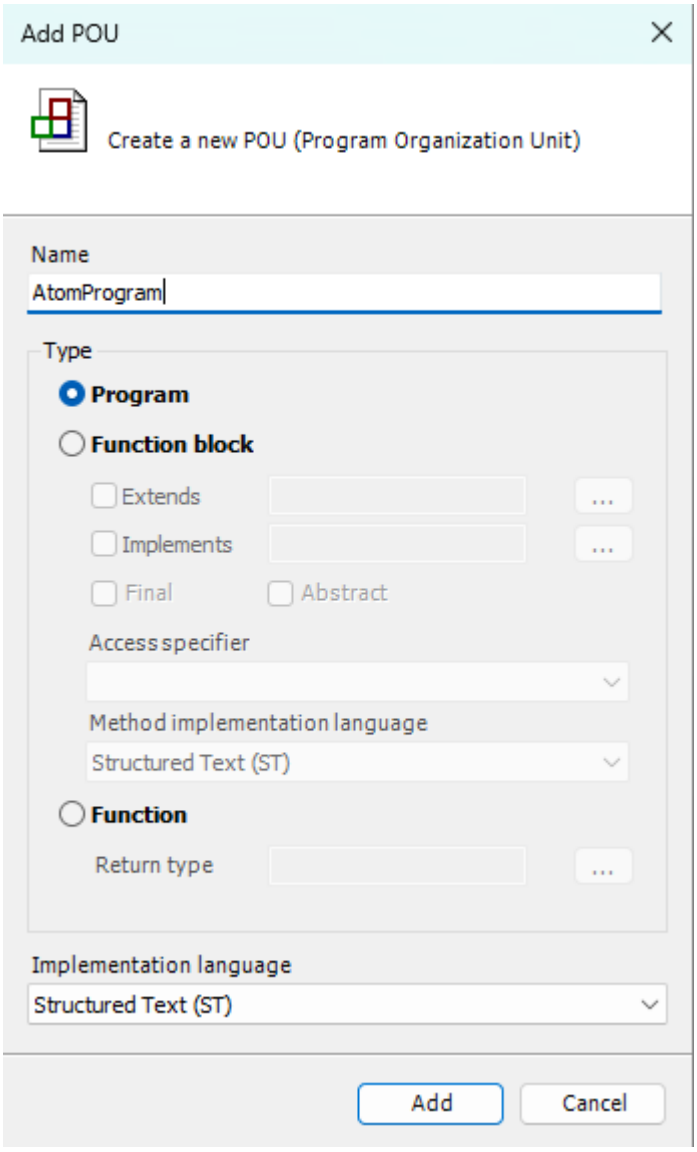

# Example: Structured text

## 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`.

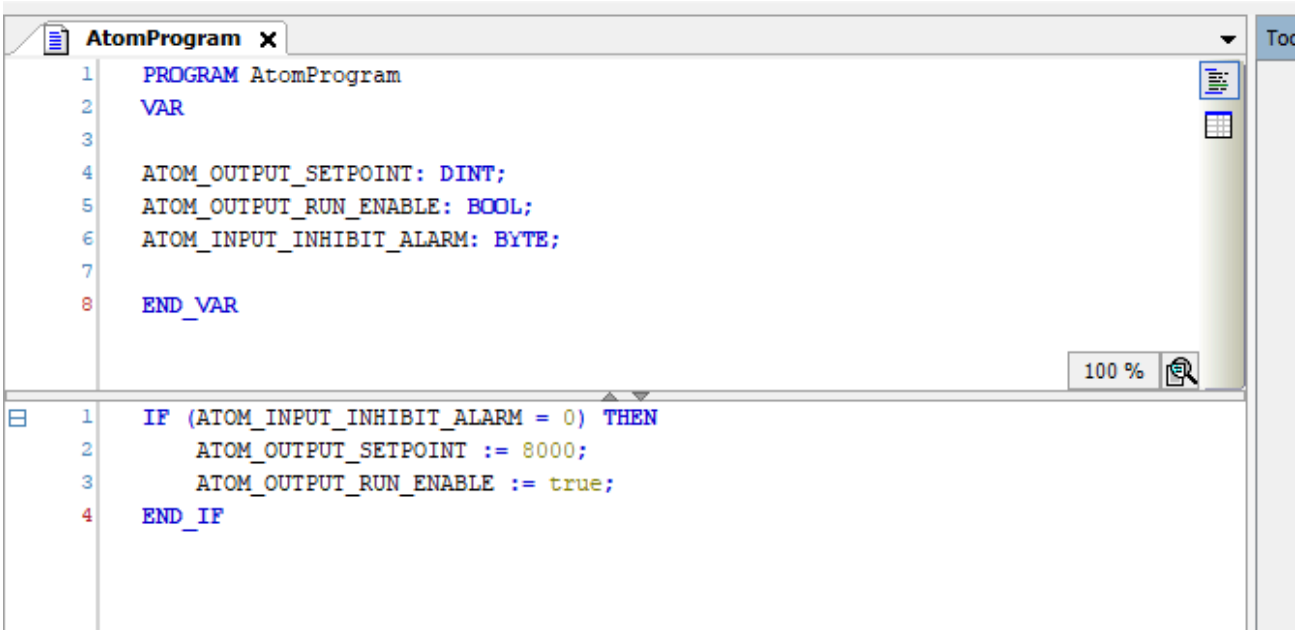
Copy the following code into the top panel of the **AtomProgram** editor:

```
PROGRAM AtomProgram
VAR

ATOM_OUTPUT_SETPOINT: BOOL;
ATOM_OUTPUT_RUN_ENABLE: BYTE;
ATOM_INPUT_INHIBIT_ALARM: BYTE;

END_VAR
```

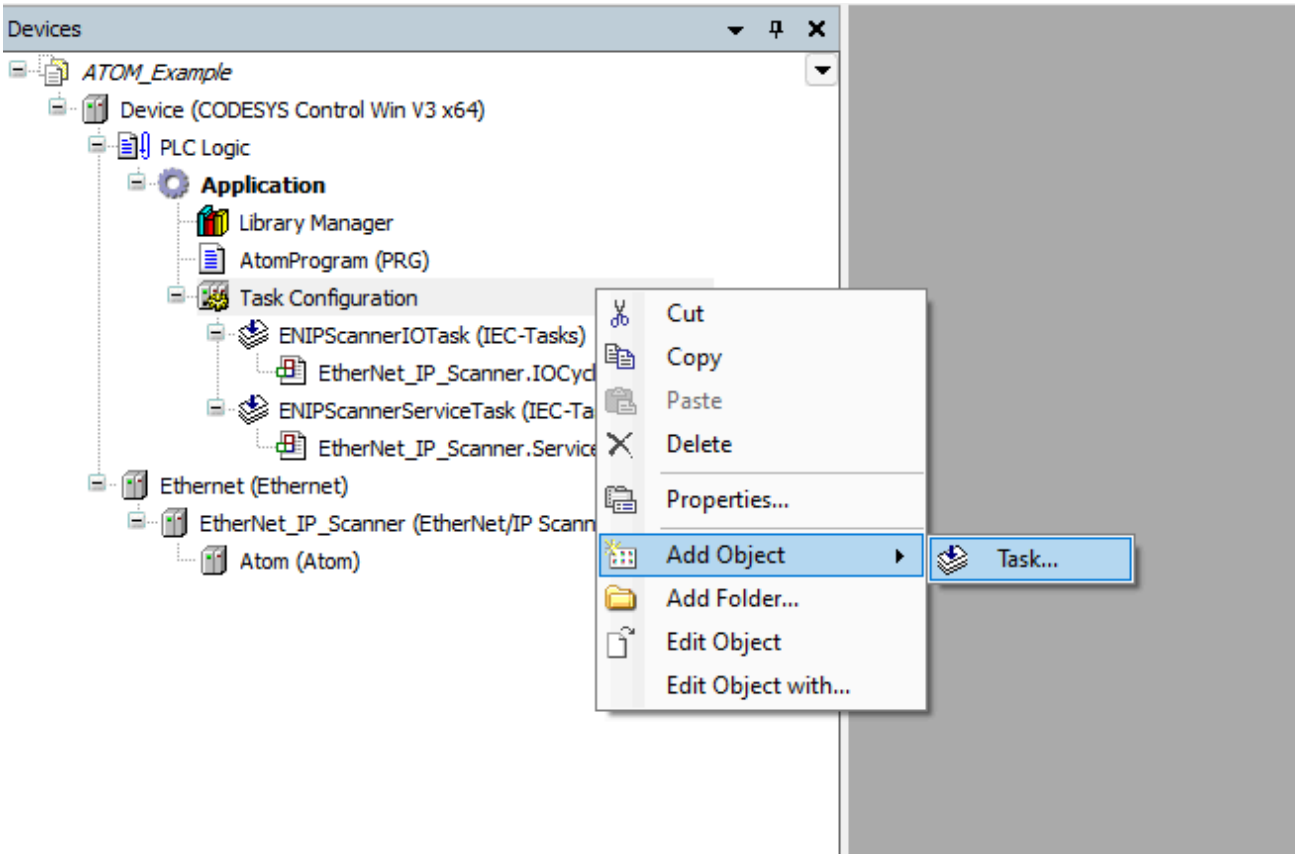
Copy the following code into the main program section:

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

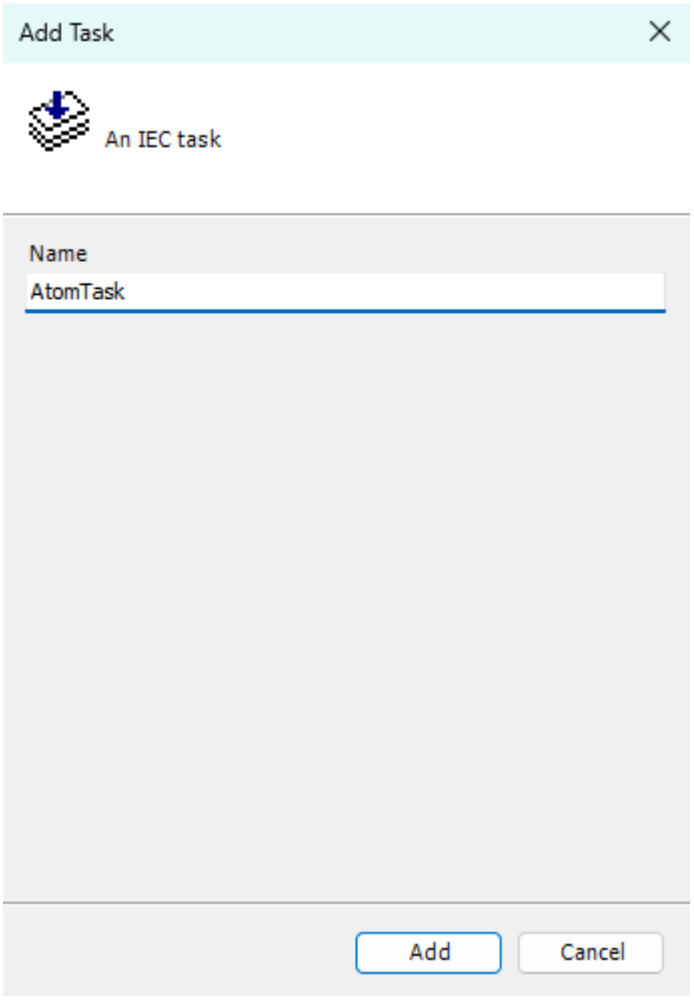
Your editor should look like:



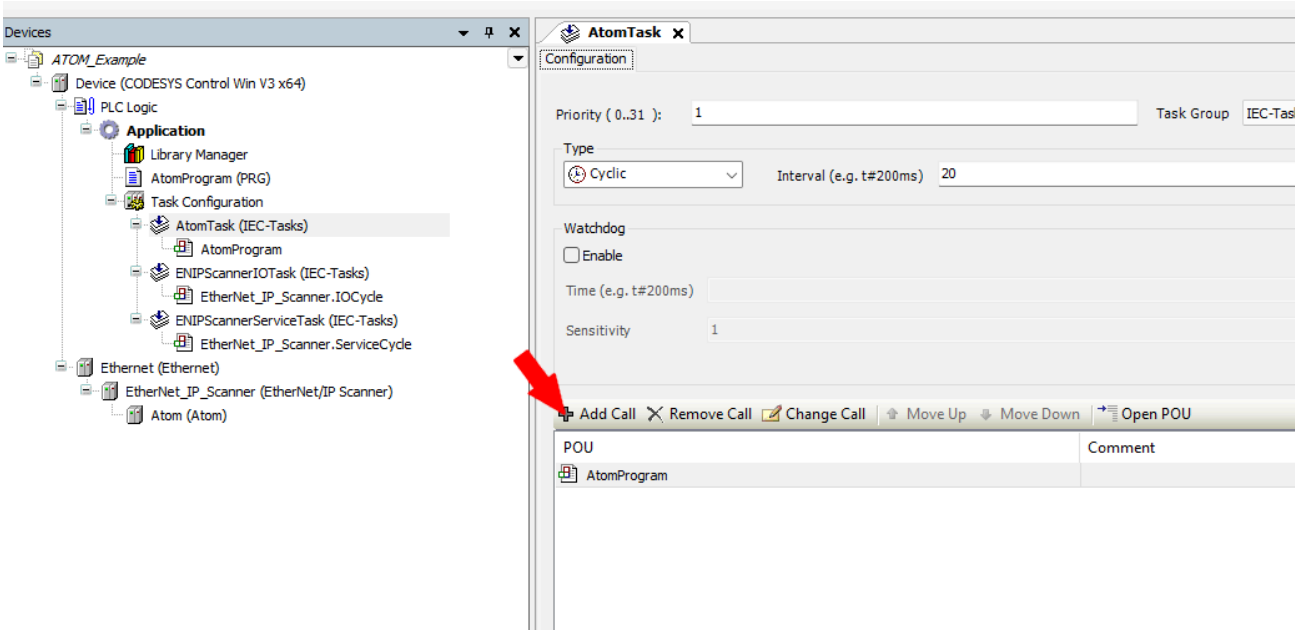
Next, we'll add a new task to call our program. Right click **Task Configuration** and Select **Add Object > Task**:




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

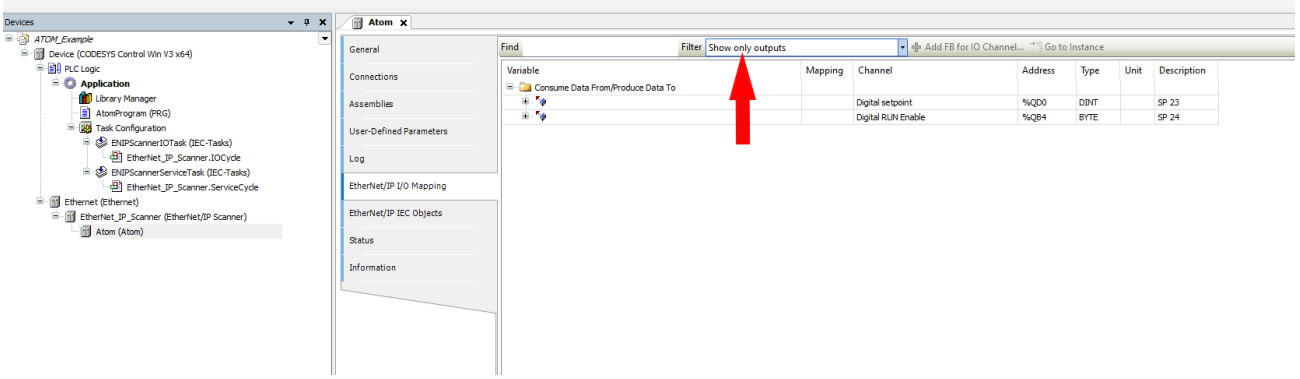



Next, double click **AtomTask (IEC-Tasks)** to open its configuration tab. Click **Add Call** and select **Application > AtomProgram**. After doing so, **AtomTask**'s configuration should look like:

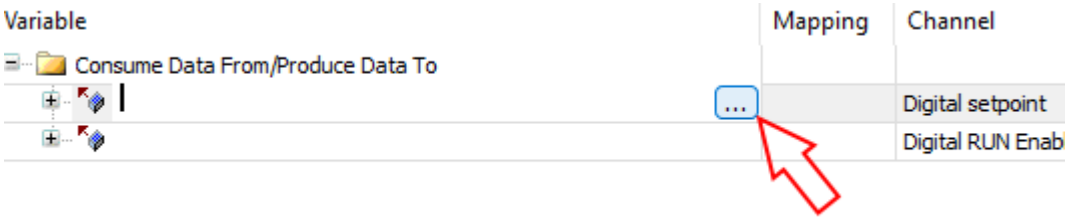



# Mapping variables

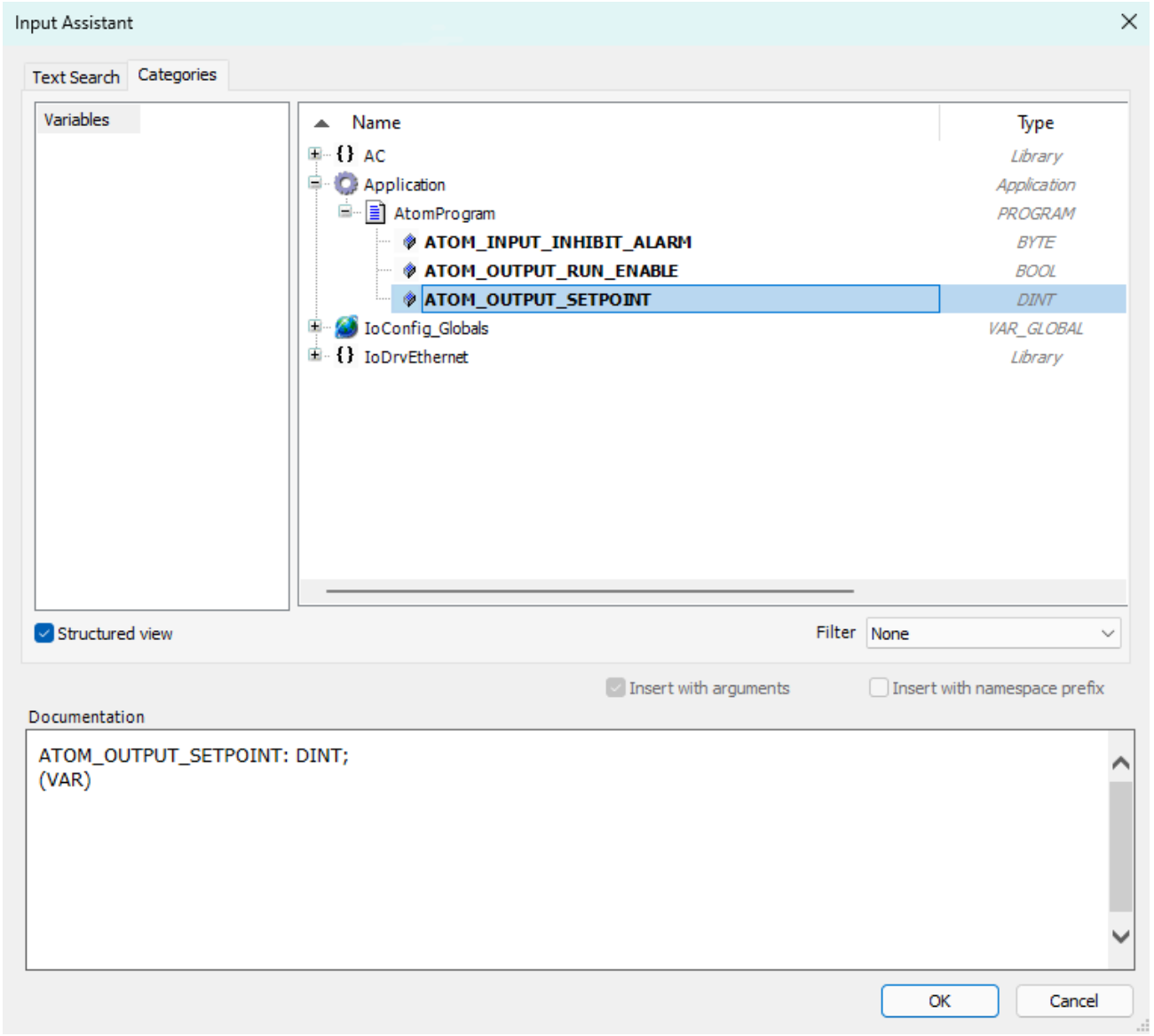
Next, we'll map our ATOM's I/O configuration to our program variables. Double click **Atom (Atom)** to open its configuration window, then select the **EtherNet/IP I/O Configuration** tab. On the **Filter** dropdown indicated by the red arrow, select **Show only outputs**:




Click the button indicated by the red arrow to map the **Digital setpoint** value:

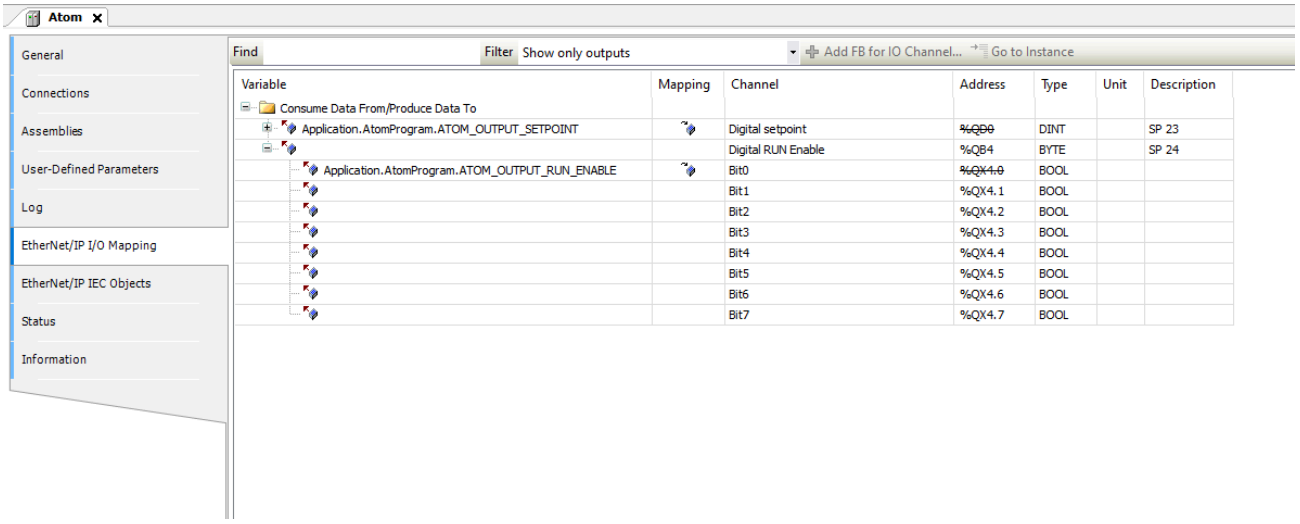



This button will open the **Input Assistant** dialog. Select the corresponding variable from your program and click **Ok**:

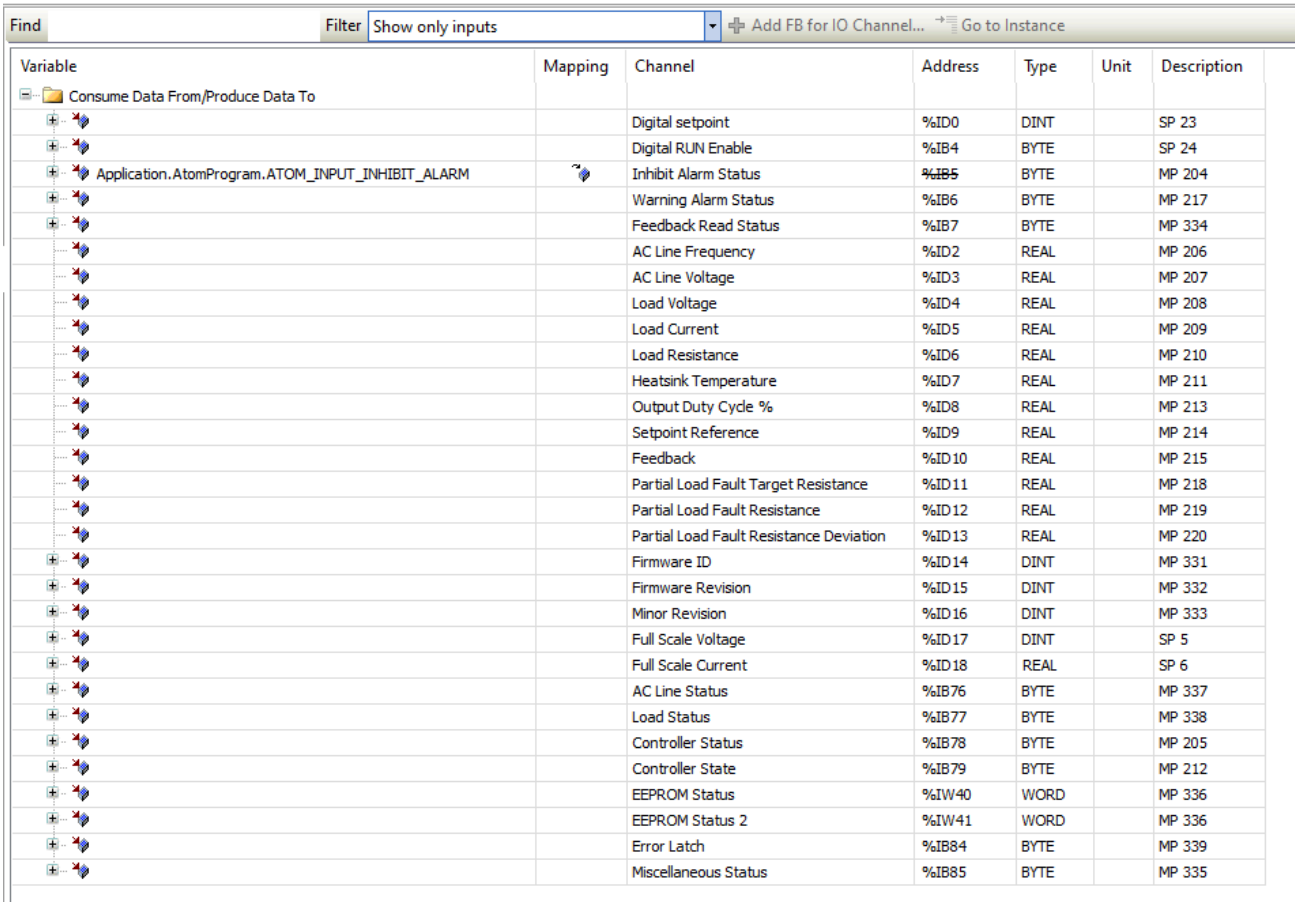


Repeat this process so that your output variables are mapped like so:

> **TAKE CARE**
>
> Make sure you map **Bit0** of **Digital RUN Enable** to **ATOM_OUTPUT_RUN_ENABLE**, NOT **Digital RUN Enable** itself.




Switch the filter to **Show only inputs** and then map the **Inhibit alarm status** variable:



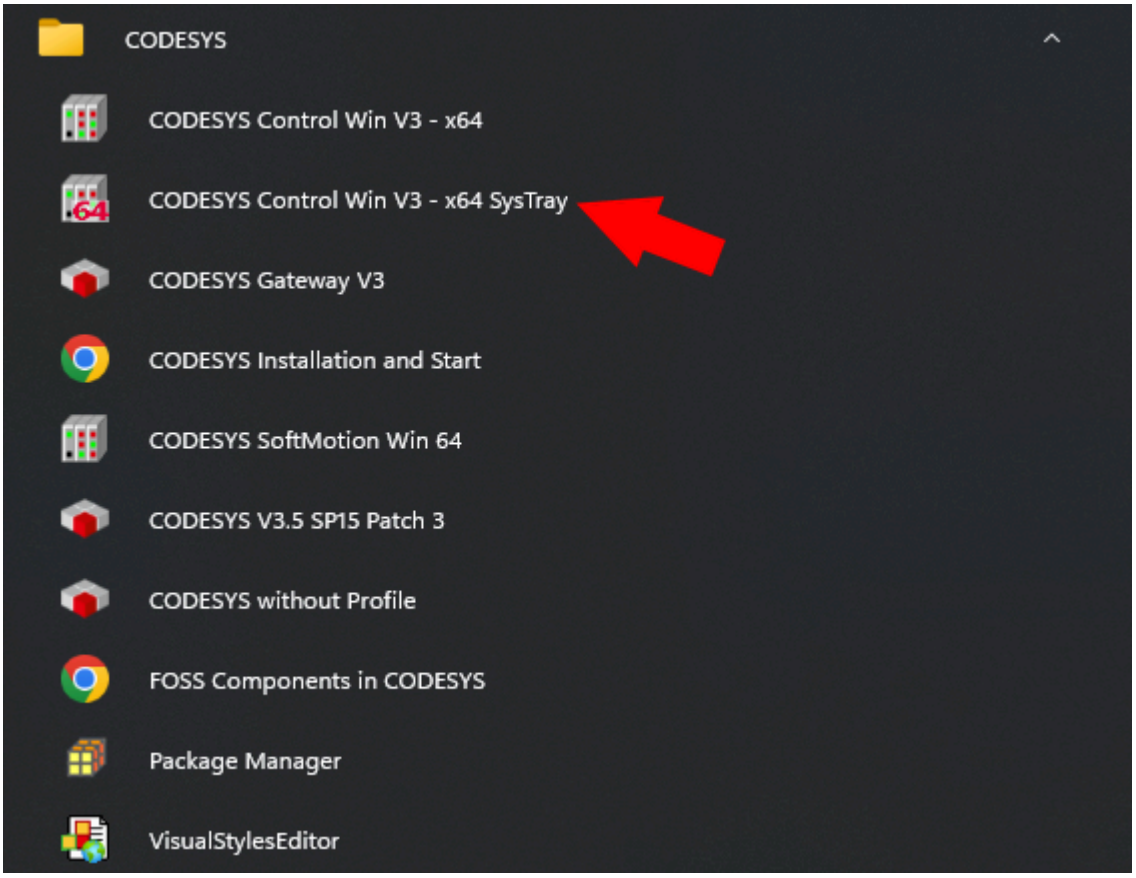
Find | Filter: Show only inputs | Add FB for IO Channel... | Go to Instance

| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| Consume Data From/Produce Data To | | | | | | |
| | | Digital setpoint | %ID0 | DINT | | SP 23 |
| | | Digital RUN Enable | %IB4 | BYTE | | SP 24 |
| Application.AtomProgram.ATOM_INPUT_INHIBIT_ALARM | | Inhibit Alarm Status | ~~%IB5~~ | BYTE | | MP 204 |
| | | Warning Alarm Status | %IB6 | BYTE | | MP 217 |
| | | Feedback Read Status | %IB7 | BYTE | | MP 334 |
| | | AC Line Frequency | %ID2 | REAL | | MP 206 |
| | | AC Line Voltage | %ID3 | REAL | | MP 207 |
| | | Load Voltage | %ID4 | REAL | | MP 208 |
| | | Load Current | %ID5 | REAL | | MP 209 |
| | | Load Resistance | %ID6 | REAL | | MP 210 |
| | | Heatsink Temperature | %ID7 | REAL | | MP 211 |
| | | Output Duty Cycle % | %ID8 | REAL | | MP 213 |
| | | Setpoint Reference | %ID9 | REAL | | MP 214 |
| | | Feedback | %ID10 | REAL | | MP 215 |
| | | Partial Load Fault Target Resistance | %ID11 | REAL | | MP 218 |
| | | Partial Load Fault Resistance | %ID12 | REAL | | MP 219 |
| | | Partial Load Fault Resistance Deviation | %ID13 | REAL | | MP 220 |
| | | Firmware ID | %ID14 | DINT | | MP 331 |
| | | Firmware Revision | %ID15 | DINT | | MP 332 |
| | | Minor Revision | %ID16 | DINT | | MP 333 |
| | | Full Scale Voltage | %ID17 | DINT | | SP 5 |
| | | Full Scale Current | %ID18 | REAL | | SP 6 |
| | | AC Line Status | %IB76 | BYTE | | MP 337 |
| | | Load Status | %IB77 | BYTE | | MP 338 |
| | | Controller Status | %IB78 | BYTE | | MP 205 |
| | | Controller State | %IB79 | BYTE | | MP 212 |
| | | EEPROM Status | %IW40 | WORD | | MP 336 |
| | | EEPROM Status 2 | %IW41 | WORD | | MP 336 |
| | | Error Latch | %IB84 | BYTE | | MP 339 |
| | | Miscellaneous Status | %IB85 | BYTE | | MP 335 |

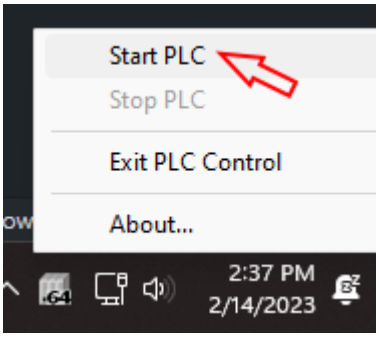
# Running the program with SoftPLC

> **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.

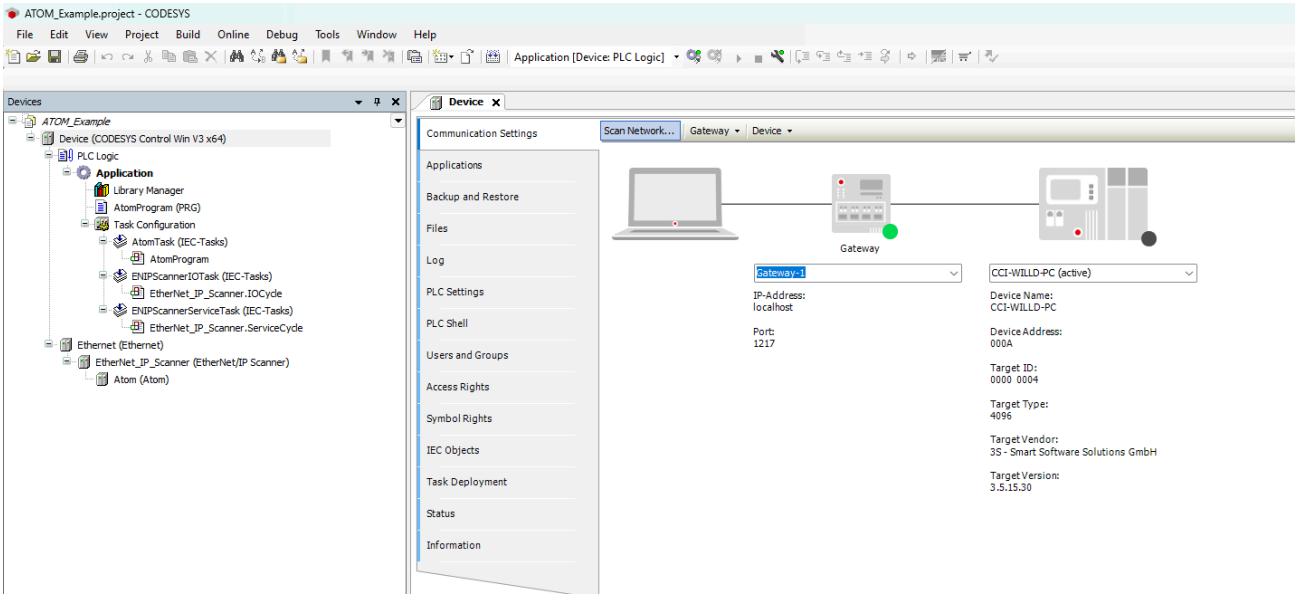
To debug the program, first make sure you start **Codesys WIN Control V3 - x64 SysTray**



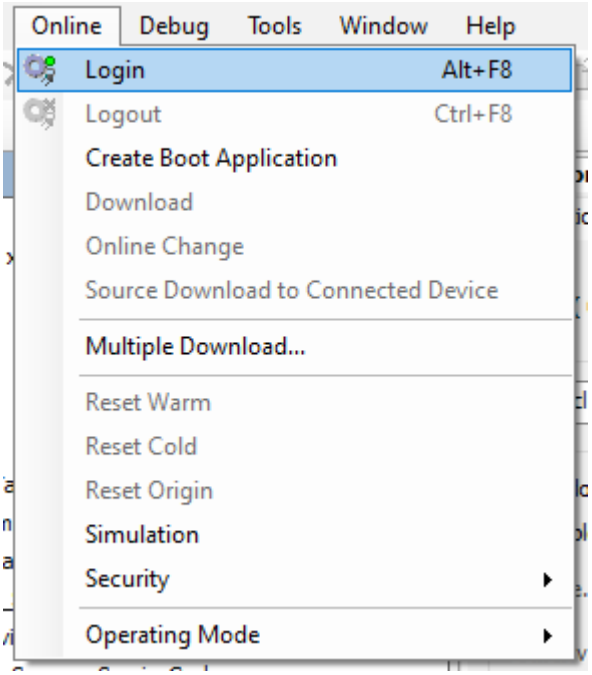

This will launch the Codesys SoftPLC. You should see an icon appear in your systray and you can right click it and select **Start PLC** to start the SoftPLC:



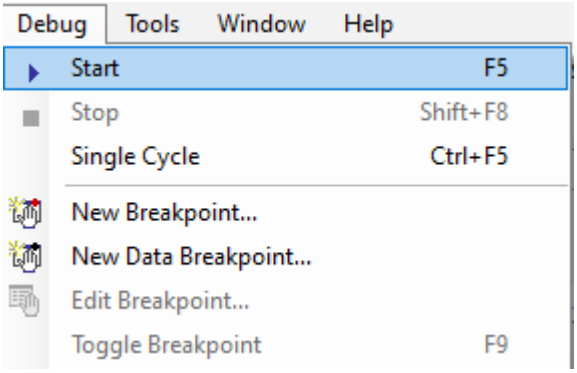

Next, in Codesys double click **Application** to open its configuration window. Here you can select **Scan Network** to discover your SoftPLC:



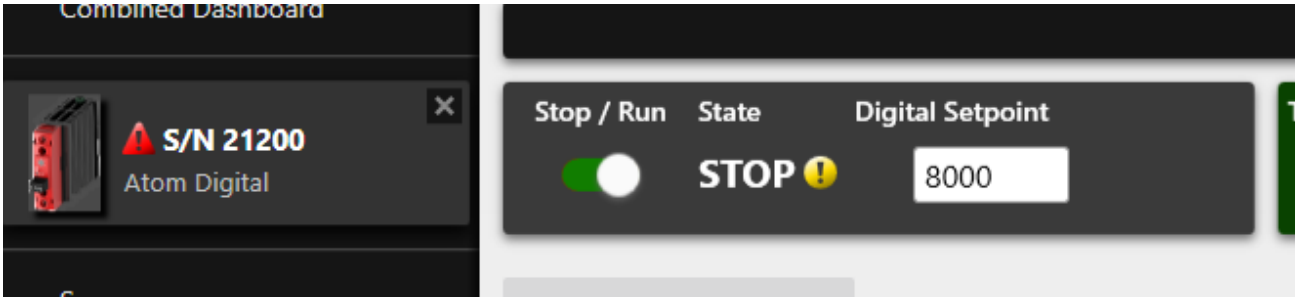

Finally, **Login** to your SoftPLC:




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:



# ATOM / Fieldbus / EtherNet/IP / Labview

> **NOTE**
>
> You do NOT need to purchase the NI-Industrial Communications for EtherNet/IP add-on.
>
> This is only used if you want your Labview application to operate as an EtherNet/IP adapter device. In our case, Labview will operate as a scanner that connects to ATOM to control it. This functionality is included in the default version of Labview.

> **NOTE**
>
> We use **explicit messaging** to connect to ATOM from Labview because Labview does not support I/O connections.

## Using our pre-built VI

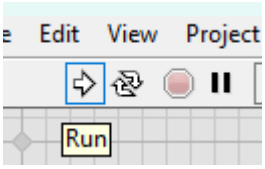
Download our ATOM control panel VI to quickly control and monitor ATOM from Labview.

Most of the parameters listed in our EtherNet/IP Profile are available through the VI controls, although you can easily extend or alter the VI to suite your needs.

Download ATOM.vi

To setup the VI, enter the IP address of your ATOM unit in the **ATOM IP Address** box. Below, we've entered `192.168.2.250` as the IP address. You can check or change your ATOM's IP address by using Control Panel.


After you've set the IP address, you can run the VI by clicking the run icon:




After the VI starts, you can adjust the Digital setpoint and RUN Enable controls. All the other indicators will update every 500 ms to reflect the current state of the ATOM unit.

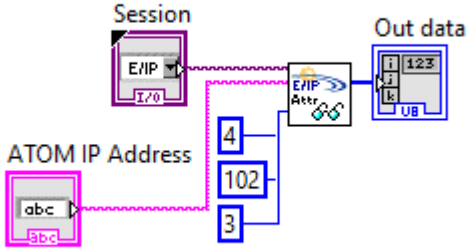
# Advanced

# Reading and parsing data from ATOM

If you need to modify or extend the VI that we provide, this section is provided to help make it easier.

The main way to pull data from ATOM is by fetching the *producing* assembly object (class 4, instance 102, attribute 3) using the Get Attribute Single service (0x0E).

Below is a simple example of fetching ATOM's assembly object. After this block diagram executes, `Out data` will be a byte array of length `86` containing the 30 parameters listed in ATOM's EtherNet/IP Profile:




To pull a parameter out of this byte array, you can calculate its index within `Out data` by finding the sum of the parameter data type sizes before it.

**EXAMPLE**

To pull out the AC Line Voltage parameter, we can calculate the sum of the parameter data type sizes before it. AC Line Voltage is parameter #7, so we sum the size of the first 6 parameters, which each have sizes of:

```
Digital setpoint = DINT = 4 bytes
Digital run enable = BOOL = 1 byte
Inhibit alarm status = BYTE = 1 byte
Warning alarm status = BYTE = 1 byte
Feedback read setatus = BOOL = 1 byte
AC Line Frequency = REAL = 4 bytes
```

Summing these up, we get:

```
4 (DINT) + 1 (BOOL) + 1 (BYTE) + 1 (BYTE) + 1 (BYTE) + 4 (REAL) = 12
```

AC Line Voltage itself is a `REAL`, so AC Line Voltage resides at index `12` within `Out data` and extends for `4` bytes.

To properly display this within your VI, you will need to reverse the order of these four bytes before interpreting them as a single precision decimal in Labview. This is because Labview is by default big-endian, while ATOM is little-endian.

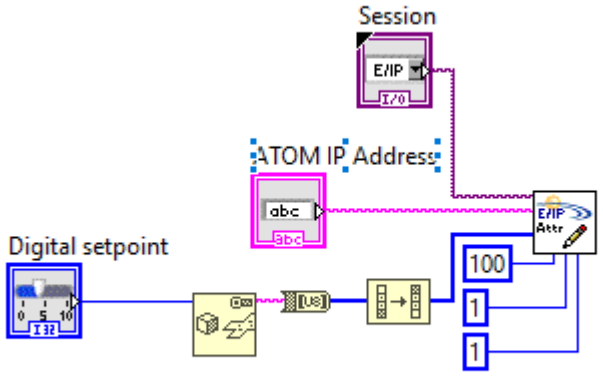
# Writing data to ATOM

Below is a simple diagram that updates the digital setpoint parameter on ATOM.

An alternative method to this one is to compile together an assembly object and use the Set Attribute Single service (0x10) to write to the consuming assembly object (class 4,

instance 101, attribute 3). The one downside to this is that it requires you to send all parameters within the consuming assembly object together.

Instead, we provide the custom ParameterLink object (class 100, instance `i`, attribute 1), where `i` is the parameter number you wish to interact with. Below, we write to ParameterLink instance `1`, attribute `1`. Consult the EtherNet/IP Profile to determine the data type of the parameter you wish to write. Here, we're writing digital setpoint, which is a `DINT`, so we'll make sure that we use an `i32` in Labview. Note that before we write the data, we have to reverse the bytes as Labview is by default big-endian, while ATOM is little-endian:



# ATOM / Fieldbus / EtherNet/IP / Other

## Unilogic PLCs

For Unilogic PLCs, ensure that:

- `Use Run Idle for T2O` = UNCHECKED
- `Use Run Idle for O2T` = CHECKED