Application Note

FESTO

Integration of Motion Insights Pneumatic Function Block with Festo PLC

This Application Note explains about Motion Insights Pneumatic function block and also shows integration of function block in CODESYS software.

100854

Title .....Integration of Motion Insights Pneumatic Function Block with Festo PLC
Version 1.10
Document no. 100854
Original .....en
Author .....Festo

Last saved 23.10.2026

General information:

This document is intended for qualified, trained and instructed professionals. The data provided in this document are no guaranteed specifications, in particular with regard to functionality, condition or quality in the legal sense. The information in this document is intended only as simple indications for the implementation of a specific, hypothetical application, and in no way as a substitute for the operating instructions of the respective manufacturers or the design and testing of the respective application by the user. The respective operating instructions for Festo products can be found under www.festo.com. The user of this document must ensure that each function described herein works properly in his application. The user remains solely responsible for his or her own use of this document, even by studying this document and using the information mentioned therein. This also applies to any software (in source code and/or object code) that is made available to the user as an appendix to this document.

Rights of use of the software:

If software is made available to the user as an appendix to this document, the user is granted a simple and unlimited right of use hereto. This right also includes the processing and distribution of the software to third parties in edited or unedited form. The User is not permitted to use the name "Festo" to endorse or promote products derived from the Software without express prior written permission.

Software is provided to the user "as is". Festo does not assume any warranty or guarantee with regard to software. Festo's liability for damages of any kind arising from the use of the software is limited to intent and gross negligence.

Legal Notices:

©Festo SE & Co. KG, all rights reserved. A change in content and form is only permitted with the express written consent of Festo SE & Co. KG. Festo grants the user the right to reproduce this document in a form that remains unchanged in terms of content and form and to pass it on to third parties.

© (Festo SE & CO. KG, D-73726 Esslingen, 2026)
Internet: http://www.festo.com

Table of Contents

1 Introduction of Motion Insights Pneumatic (MIP) Function Block 5

1.1 Software Overview 6

1.2 Introduction to “FB_MIP” Function Block 6

1.2.1 FB_MIP Block Input Variables 7

1.2.2 FB_MIP Block Output Variables 8

1.3 Introduction to “FB_MIP_Single” Function Block 8

1.3.1 FB_MIP_Single Block Input Variables 9

1.3.2 FB_MIP_Single Block Output Variables 10

1.4 Examples 10

2 Adding and Configuring the MIP Function Block into CODESYS 12

2.1 Install the MIP Library into CODESYS 12

2.2 Adding the MIP Library into the PLC Project 14

2.3 Configuring the FB_MIP Function Block 16

2.4 OPC UA Communication with CODESYS 20

1 Introduction of Motion Insights Pneumatic (MIP) Function Block

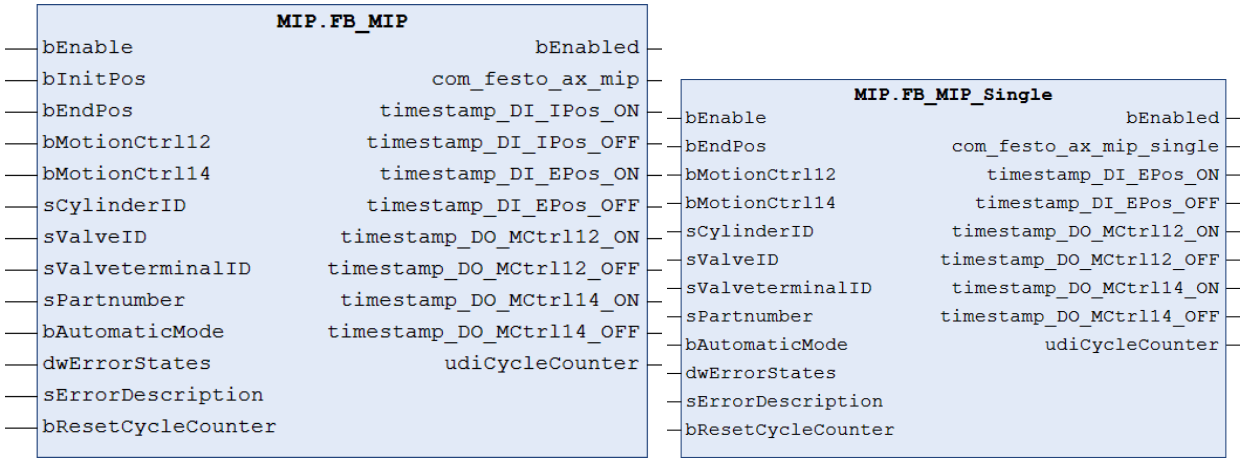
The **Motion Insights Pneumatic (MIP)** library provides standardized function blocks for monitoring and re-cording pneumatic cylinder motion data. The library records timestamps for important motion signals and stores cylinder identification and diagnostic information, including the Cylinder ID, Valve ID, Valve Terminal ID, Part Number, automatic operating mode, error states, and error descriptions.

The library consists of two function blocks:

The **FB_MIP**: Used for pneumatic cylinders equipped with two sensors: an initial-position sensor and an end-position sensor. It uses OPC UA communication.

The **FB_MIP_Single**: Used for pneumatic cylinders equipped with a single sensor. The sensor can be placed either at extracted or retracted position of the cylinder. It uses OPC UA communication.

The function blocks exchange the collected information through their input and output tags. External systems can access the pneumatic cylinder data through OPC UA communication for monitoring, diagnostics, traceability, and further analysis.



Note

- Users must select and call the appropriate function block based on the cylinder's sensor configuration used in the application.

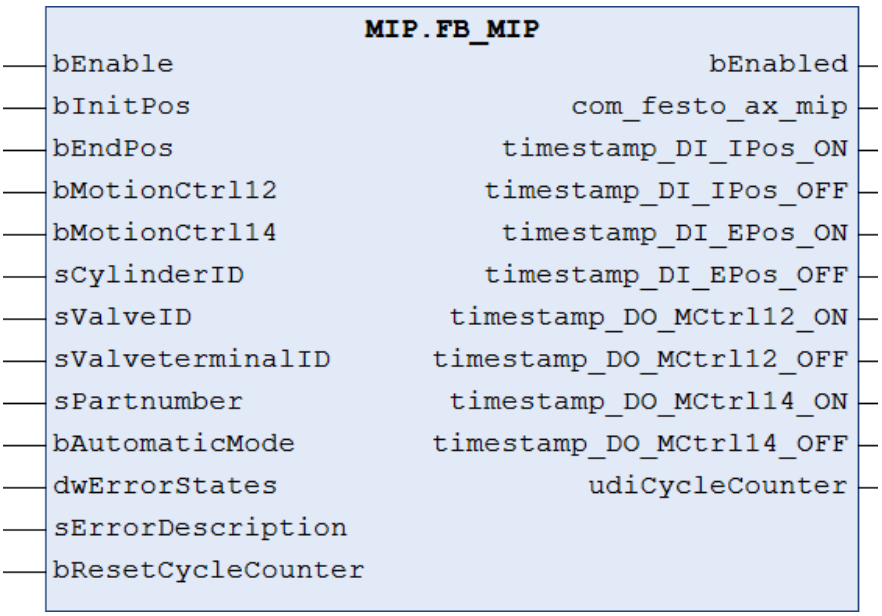
1.1 Software Overview

Software Name	Software Version
CODESYS	V3.5 SP15 Patch 4 & above

Table 1.1: Software used

1.2 Introduction to "FB_MIP" Function Block

The **FB_MIP** function block is used to monitor and record the timestamp of the input tags like bInitPos, bEndPos, bMotionCtrl12 and bMotionCtrl14 and hold the information of the Cylinder ID, Valve ID, Valve Terminal ID and other information like Part Number, Automatic mode, Error States and Error Description using Input and Output tags. This function block allows the OPC UA communication to fetch the cylinder data from the block.



1.2.1 FB_MIP Block Input Variables

Input Variables	Mandatory	Data Type	Description
bEnable	x	BOOL	**TRUE -** Enable Function Block
bInitPos	x	BOOL	**TRUE –** Updates the Timestamp of **bInitPos** Input Signal Rising Edge in **timestamp_DI_IPos_ON** Output. **FALSE –** Updates the Timestamp of **bInitPos** Input Signal Falling Edge in **timestamp_DI_IPos_OFF** Output.
bEndPos	x	BOOL	**TRUE –** Updates the Timestamp of **bEndPos** Input Signal Rising Edge in **timestamp_DI_EPos_ON** Output. **FALSE –** Updates the Timestamp of **bEndPos** Input Signal Falling Edge in **timestamp_DI_EPos_OFF** Output.
bMotionCtrl12	x (only with bistable valves)	BOOL	**TRUE –** Updates the Timestamp of **bMotionCtrl12** Input Signal Rising Edge in **timestamp_DO_MCtrl12_ON** Output. **FALSE –** Updates the Timestamp of **bMotionCtrl12** Input Signal Falling Edge in **timestamp_ DO_MCtrl12_OFF** Output.
bMotionCtrl14	x	BOOL	**TRUE –** Updates the Timestamp of **bMotionCtrl14** Input Signal Rising Edge in **timestamp_DO_MCtrl14_ON** Output. **FALSE –** Updates the Timestamp of **bMotionCtrl14** Input Signal Falling Edge in **timestamp_ DO_MCtrl14_OFF** Output.
sCylinderID	x	STRING[48]	Enter the Cylinder ID.
sValveID	x	STRING[48]	Enter the Valve ID.
sValveterminalID		STRING[48]	Enter the Valve Terminal ID.
sPartnumber		STRING[48]	Enter the Part Number.
bAutomaticMode	x	BOOL	Input the Mode Status.
dwErrorStates		DWORD	Enter the Error Status.
sErrorDescription		STRING[256]	Enter the Error Description.
bResetCycleCounter		BOOL	**TRUE –** Reset the Cycle Counter**.**

Table 1.2: FB_MIP Block Input Variables descriptions



Note

The assignment of valve signal(s) and sensor signals is defined as follows:

Monostable valve (one coil)

bMotionCtrl14 = **TRUE** moves the cylinder towards **bEndPos**.

bMotionCtrl14 = **FALSE** moves the cylinder towards **bInitPos**.

Bi-stable valve (two coils)

bMotionCtrl14 = **TRUE** moves the cylinder towards **bEndPos**.

bMotionCtrl12 = **TRUE** moves the cylinder towards **bInitPos**.

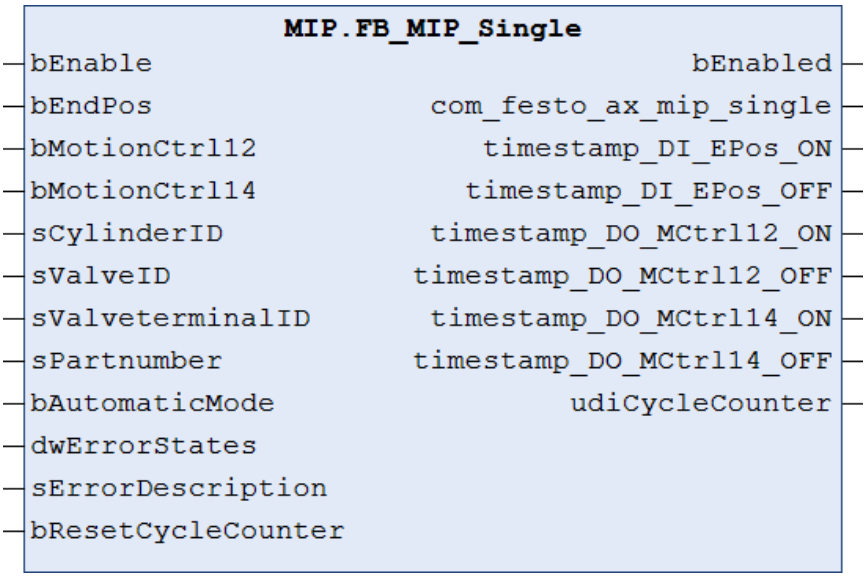
1.2.2 FB_MIP Block Output Variables

Output Variables	Data Type	Description
xEnabled	BOOL	**TRUE –** Function Block Enabled.
com_festo_ax_mip	STRING[10]	Constant String Output (1.1.0).
timestamp_DI_IPos_ON	ULINT	Timestamp of **bInitPos** Signal Rising Edge.
timestamp_DI_IPos_OFF	ULINT	Timestamp of **bInitPos** Signal Falling Edge.
timestamp_DI_EPos_ON	ULINT	Timestamp of **bEndPos** Signal Rising Edge.
timestamp_DI_EPos_OFF	ULINT	Timestamp of **bEndPos** Signal Falling Edge.
timestamp_DO_MCtrl12_ON	ULINT	Timestamp of **bMotionCtrl12** Signal Rising Edge.
timestamp_DO_MCtrl12_OFF	ULINT	Timestamp of **bMotionCtrl12** Signal Falling Edge.
timestamp_DO_MCtrl14_ON	ULINT	Timestamp of **bMotionCtrl14** Signal Rising Edge.
timestamp_DO_MCtrl14_OFF	ULINT	Timestamp of **bMotionCtrl14** Signal Falling Edge.
udiCycleCounter	UDINT	Cycle Counter of the Timestamps of the Cylinder.

Table 1.3: FB_MIP Block Output Variables descriptions

1.3 Introduction to "FB_MIP_Single" Function Block

The **FB_MIP_Single** function block is used to monitor and record the timestamp of the input tags like bEndPos, bMotionCtrl12 and bMotionCtrl14 and hold the information of the Cylinder ID, Valve ID, Valve Terminal ID and other information like Part Number, Automatic mode, Error States and Error Description using Input and Output tags. This function block allows the OPCUA communication to fetch the cylinder data from the block.



1.3.1 FB_MIP_Single Block Input Variables

Input Variables	Mandatory	Data Type	Description
bEnable	x	BOOL	**TRUE -** Enable Function Block
bEndPos	x	BOOL	**TRUE –** Updates the Timestamp of **bEndPos** Input Signal Rising Edge in **timestamp_DI_EPos_ON** Output. **FALSE –** Updates the Timestamp of **bEndPos** Input Signal Falling Edge in **timestamp_DI_EPos_OFF** Output.
bMotionCtrl12	x (only with bistable valves)	BOOL	**TRUE –** Updates the Timestamp of **bMotionCtrl12** Input Signal Rising Edge in **timestamp_DO_MCtrl12_ON** Output. **FALSE –** Updates the Timestamp of **bMotionCtrl12** Input Signal Falling Edge in **timestamp_ DO_MCtrl12_OFF** Output.
bMotionCtrl14	x	BOOL	**TRUE –** Updates the Timestamp of **bMotionCtrl14** Input Signal Rising Edge in **timestamp_DO_MCtrl14_ON** Output. **FALSE –** Updates the Timestamp of **bMotionCtrl14** Input Signal Falling Edge in **timestamp_ DO_MCtrl14_OFF** Output.
sCylinderID	x	STRING[48]	Enter the Cylinder ID.
sValveID	x	STRING[48]	Enter the Valve ID.
sValveterminalID		STRING[48]	Enter the Valve Terminal ID.
sPartnumber		STRING[48]	Enter the Part Number.
bAutomaticMode	x	BOOL	Input the Mode Status.
dwErrorStates		DWORD	Enter the Error Status.
sErrorDescription		STRING[256]	Enter the Error Description.
bResetCycleCounter		BOOL	**TRUE –** Reset the Cycle Counter**.**

Table 1.4: FB_MIP_Single Block Input Variables descriptions

➔ **Note**

The assignment of valve signal(s) and sensor signals is defined as follows:

Monostable valve (one coil)

bMotionCtrl14 = **TRUE** moves the cylinder towards **bEndPos**.

bMotionCtrl14 = **FALSE** moves the cylinder towards **bInitPos**.

Bi-stable valve (two coils)

bMotionCtrl14 = **TRUE** moves the cylinder towards **bEndPos**.

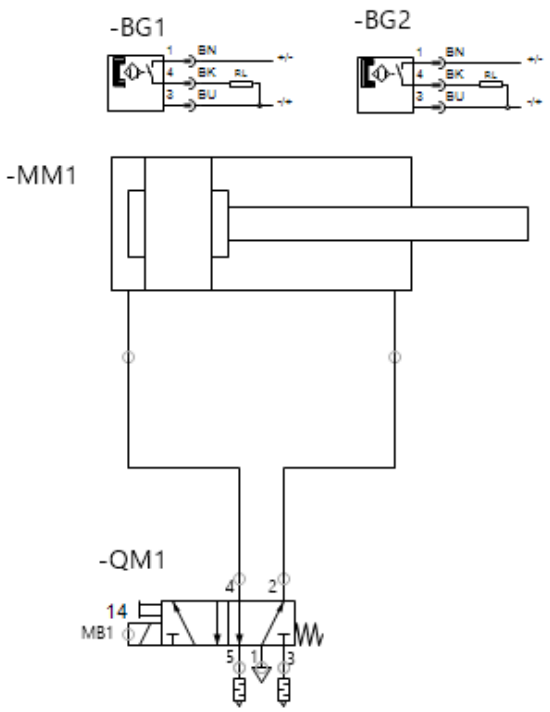
bMotionCtrl12 = **TRUE** moves the cylinder towards **bInitPos**.

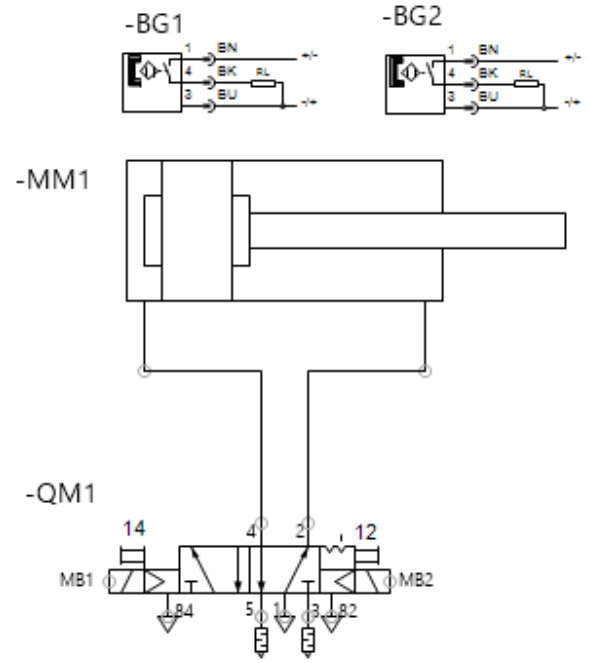
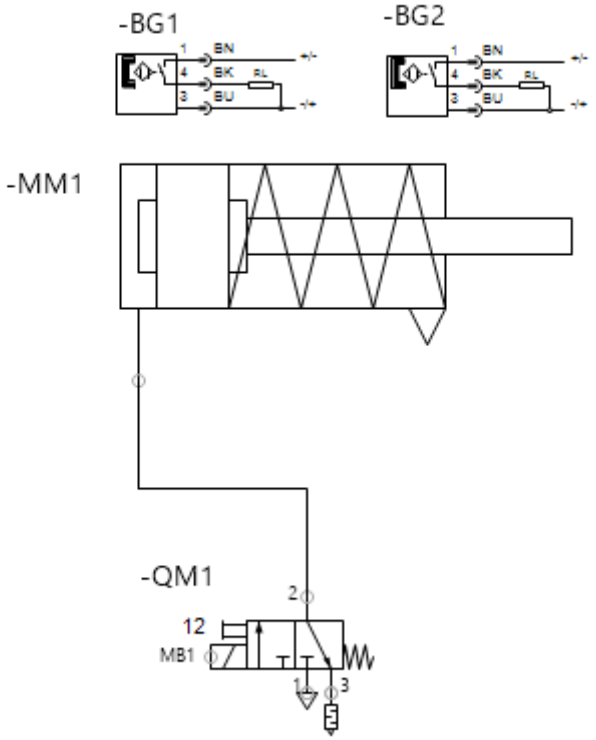
1.3.2 FB_MIP_Single Block Output Variables

Output Variables	Data Type	Description
xEnabled	BOOL	**TRUE –** Function Block Enabled.
com_festo_ax_mip_single	STRING[10]	Constant String Output (1.1.0).
timestamp_DI_EPos_ON	ULINT	Timestamp of **bEndPos** Signal Rising Edge.
timestamp_DI_EPos_OFF	ULINT	Timestamp of **bEndPos** Signal Falling Edge.
timestamp_DO_MCtrl12_ON	ULINT	Timestamp of **bMotionCtrl12** Signal Rising Edge.
timestamp_DO_MCtrl12_OFF	ULINT	Timestamp of **bMotionCtrl12** Signal Falling Edge.
timestamp_DO_MCtrl14_ON	ULINT	Timestamp of **bMotionCtrl14** Signal Rising Edge.
timestamp_DO_MCtrl14_OFF	ULINT	Timestamp of **bMotionCtrl14** Signal Falling Edge.
udiCycleCounter	UDINT	Cycle Counter of the Timestamps of the Cylinder.

Table 1.5: FB_MIP_Single Block Output Variables descriptions

1.4 Examples

Schematic	Valve Type	Signals
	5/2 monostable	-BG1 >> bInitPos -BG2 >> bEndPos -QM1-MB1 >> bMotionCtrl14

Schematic	Valve Type	Signals
	5/2 bistable	-BG1 >> bInitPos -BG2 >> bEndPos -QM1-MB1 >> bMotionCtrl14 -QM1-MB2 >> bMotionCtrl12
	3/2 monostable	-BG1 >> bInitPos -BG2 >> bEndPos -QM1-MB1 >> bMotionCtrl14

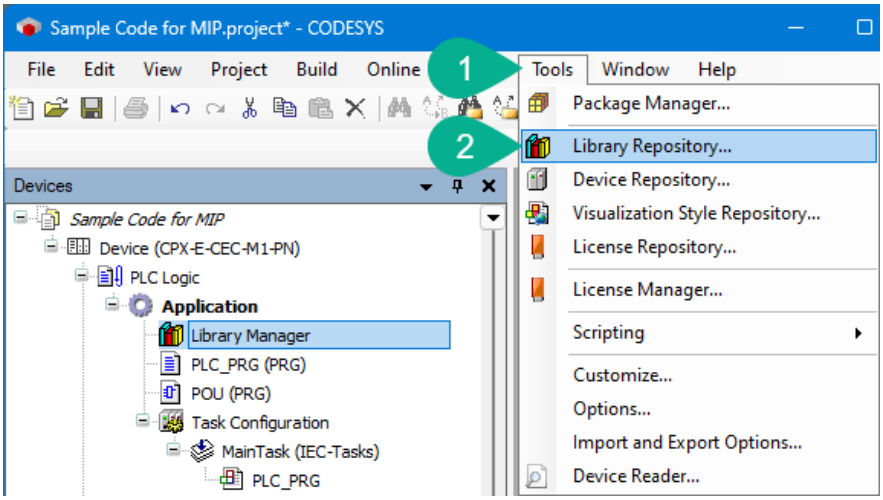
Note

- For **FB_MIP_Single**, the available signals are only BG2 and QM1–MB1/MB2.

2 Adding and Configuring the MIP Function Block into CODESYS

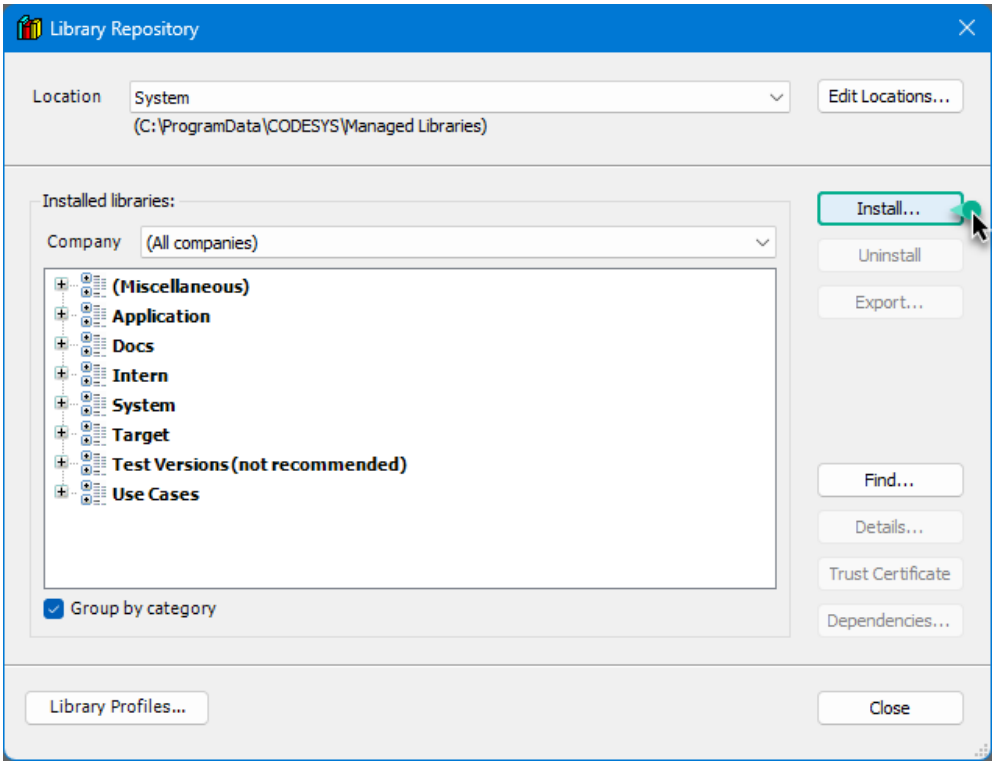
2.1 Install the MIP Library into CODESYS

Step 1



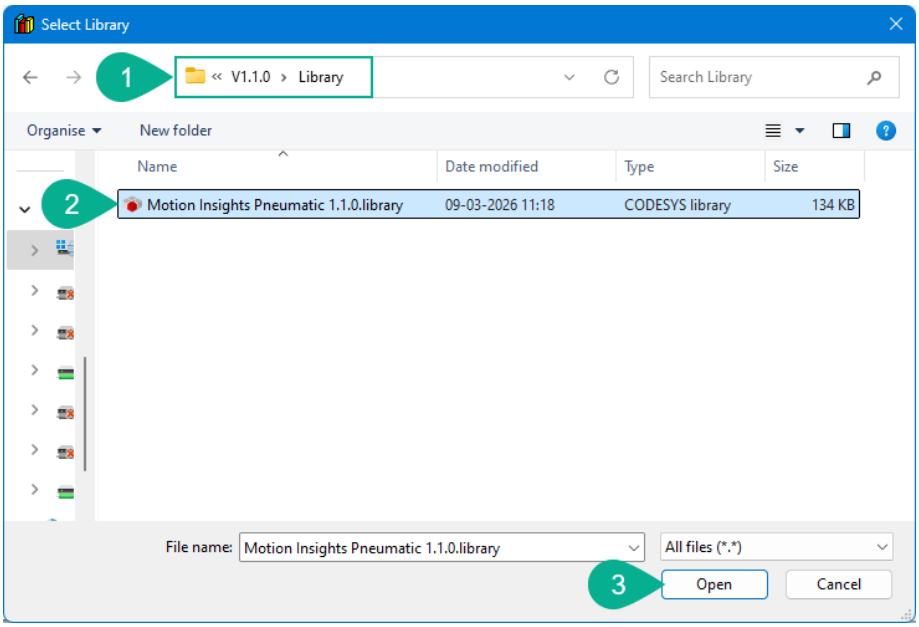
1. Select the **Tools** menu option from the toolbar.
2. Click the **Library Repository...** option from the **Tools** menu option.

Step 2



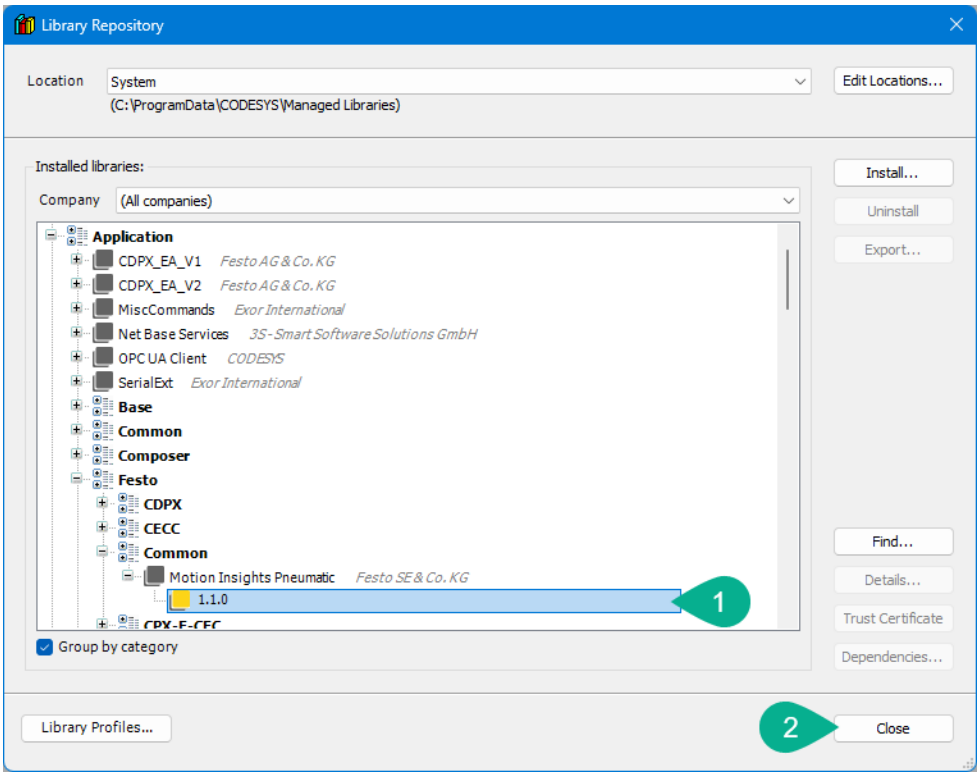
Click the **Install** button in the **Library Repository** window to open the library file.

Step 3



1. Go to the destination where library folder has been saved.
2. Select **Motion Insights Pneumatic** library from the folder.
3. Click on **Open** to install the library.

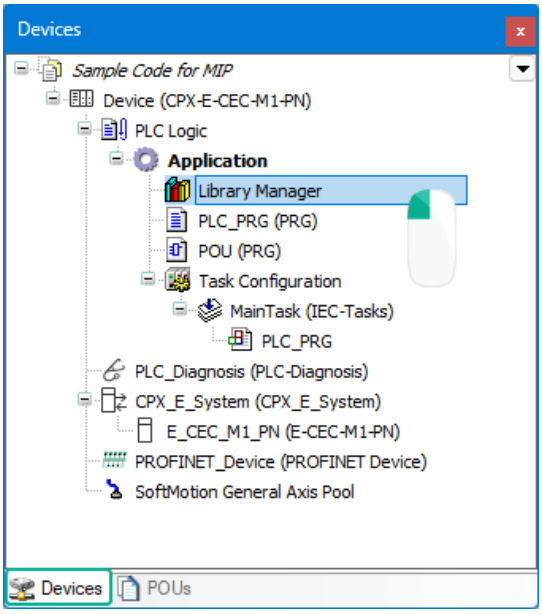
Step 4



1. The installed **Motion Insights Pneumatic** library can be found in following path **Application >> Festo >> Common >>** option in the **Library Repository** window.
2. Click the **Close** button to close the **Library Repository** window.

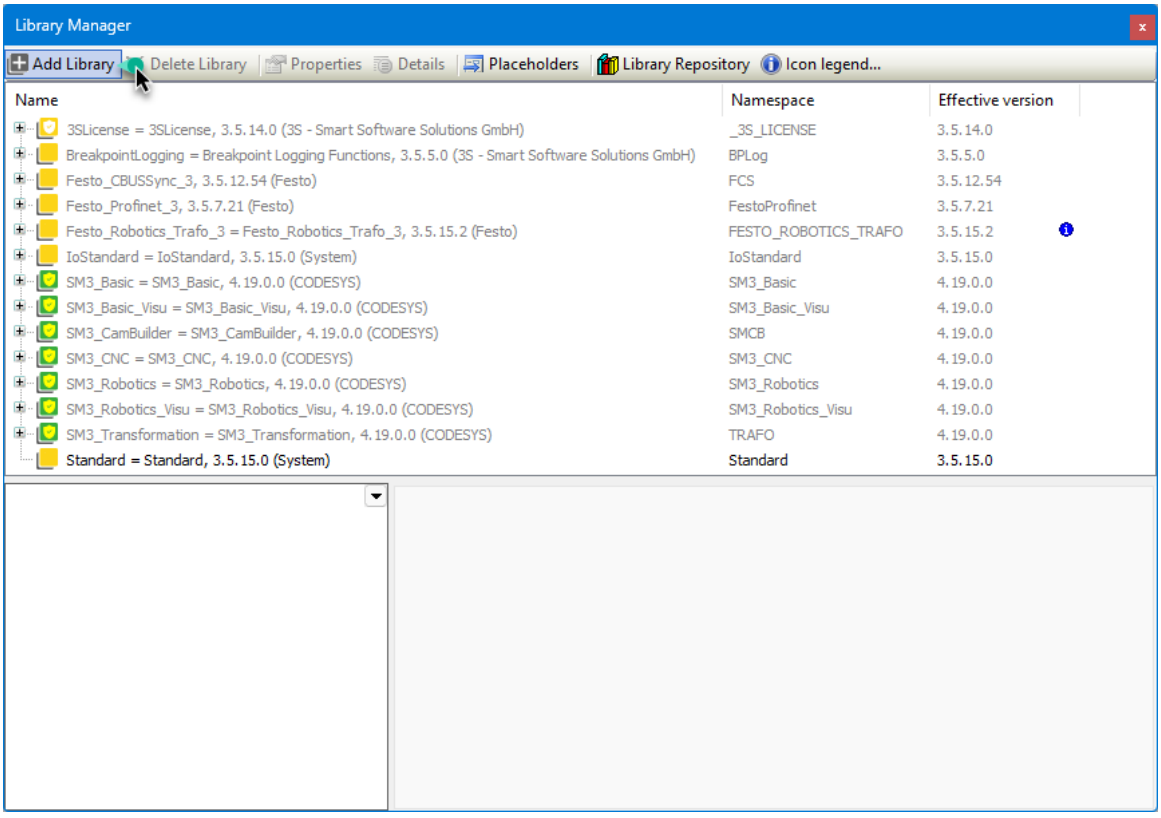
2.2 Adding the MIP Library into the PLC Project

Step 1



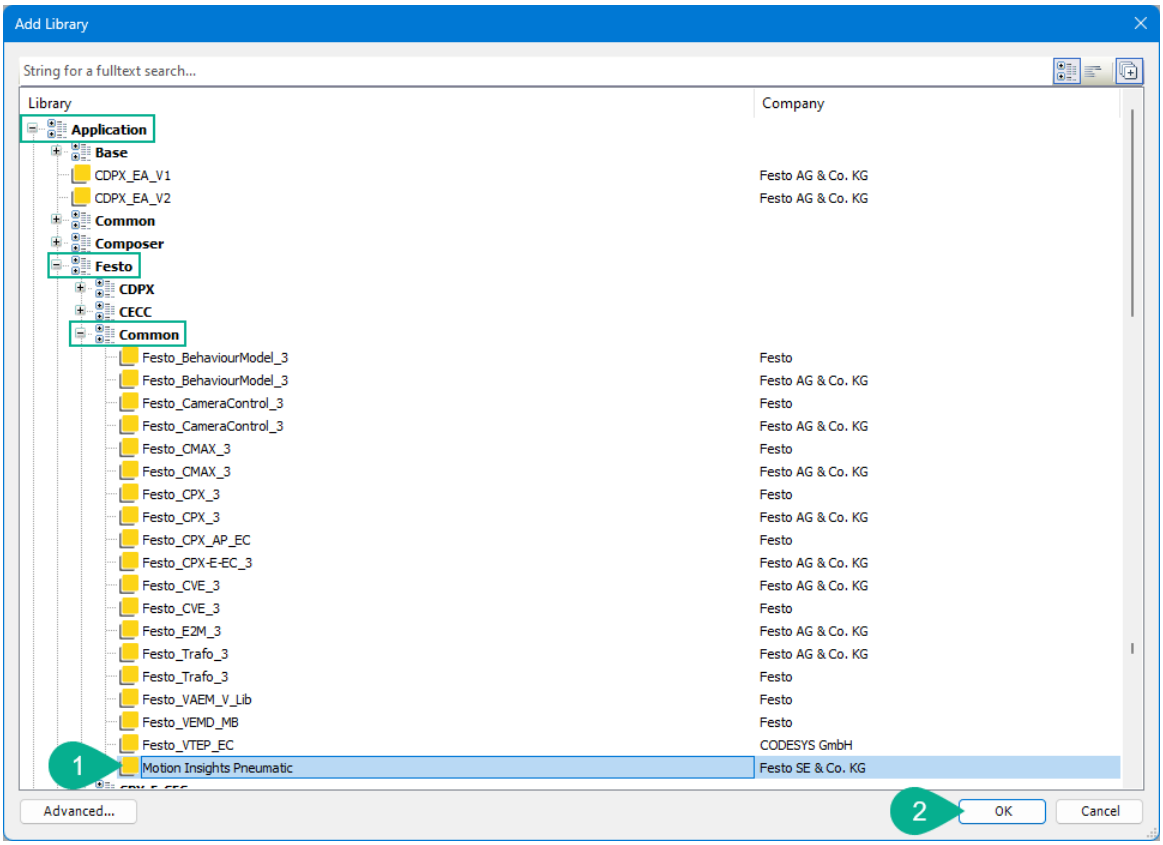
Double click the **Library Manager** option under the **Application** option from **Device Tree** in the **Devices** toolbox.

Step 2



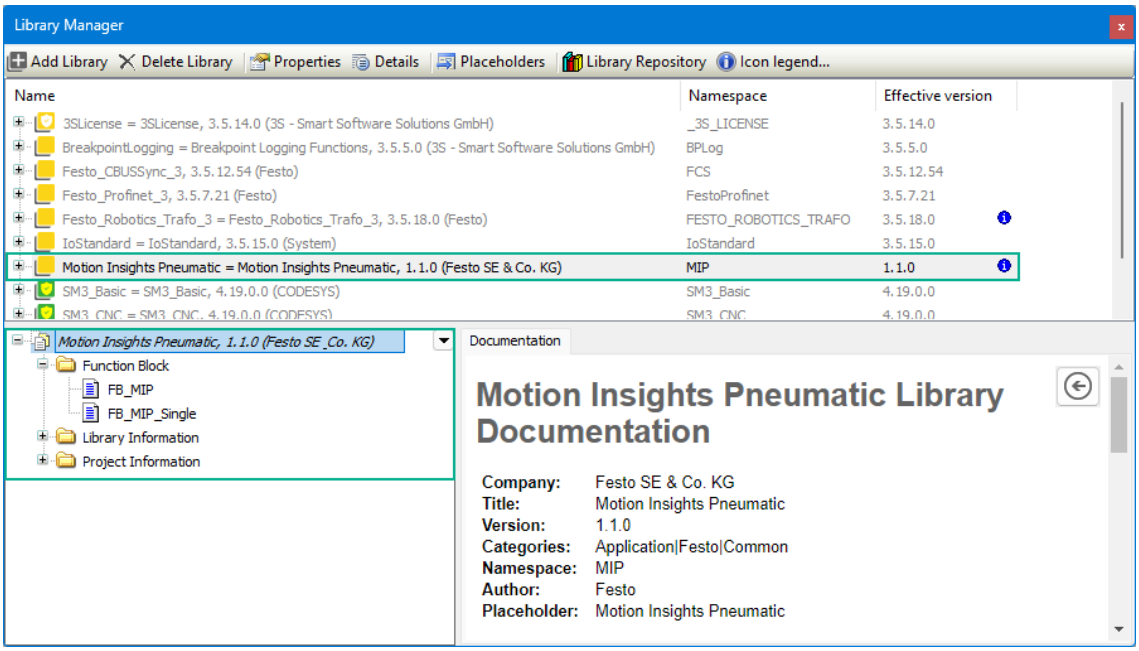
Click the **Add Library** button in the **Library Manager** window.

Step 3



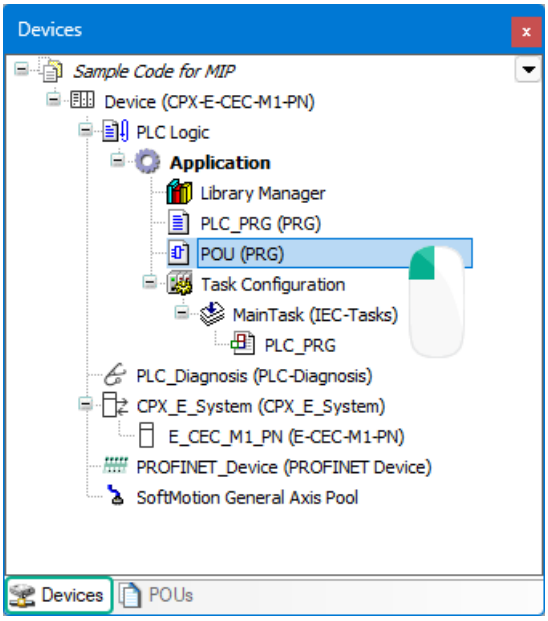
1. Select the **Motion Insights Pneumatic** library from the installed library list (**Application >> Festo >> Common**).
2. Click the **OK** button to add the library to the project.

Once the library is added successfully, the library elements will be available as shown below, highlighted in green.



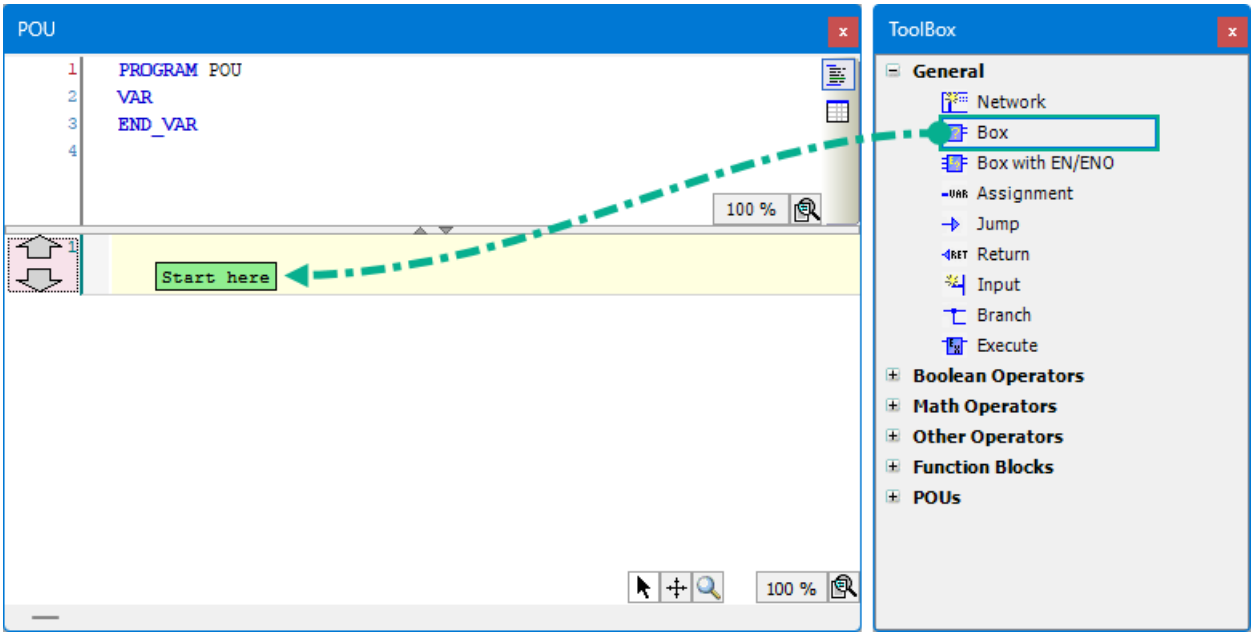
2.3 Configuring the FB_MIP Function Block

Step 1



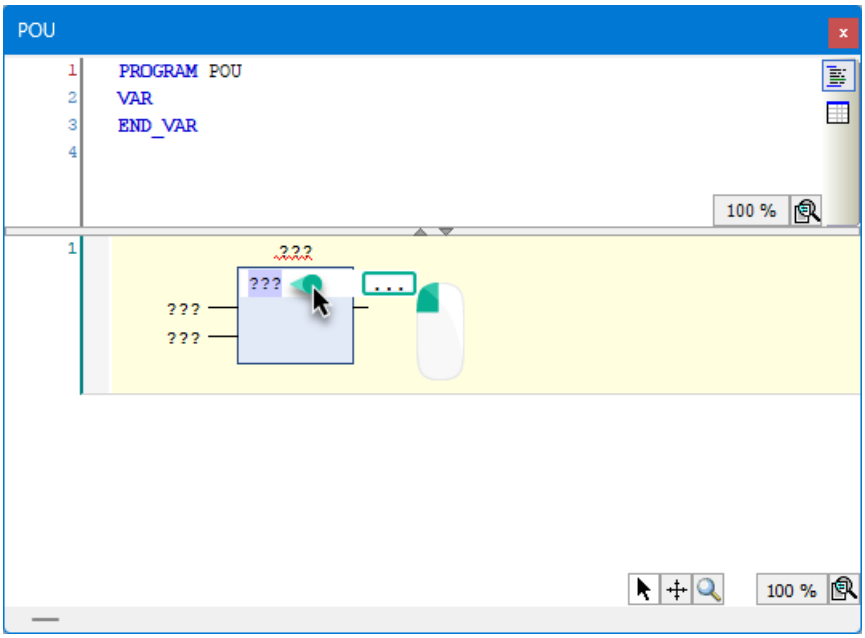
Double click the **POU (PRG)** option under the **Application** from the **Device Tree** in the **Devices** toolbox.

Step 2



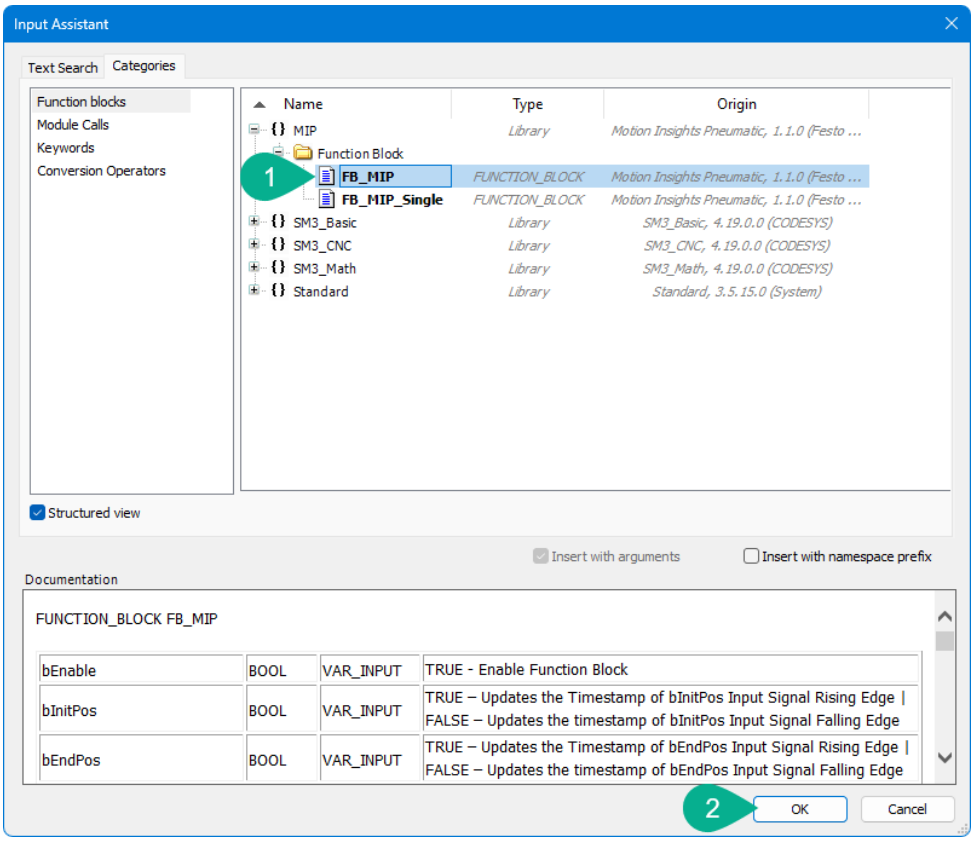
Drag and drop the **Box** element from the **Toolbox** window to an empty rung in the **POU** window.

Step 3



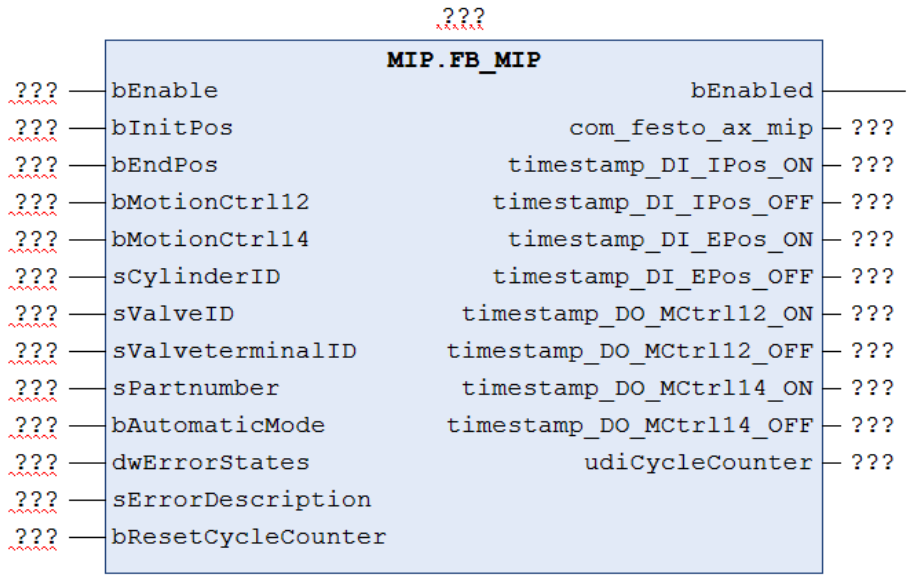
Click inside the added **Box** element and click the **… (Browse)** button to open the **Input Assistant** window.

Step 4



1. Select the **FB_MIP** function block from the **MIP** library.
2. Click the **OK** button to add the function block.

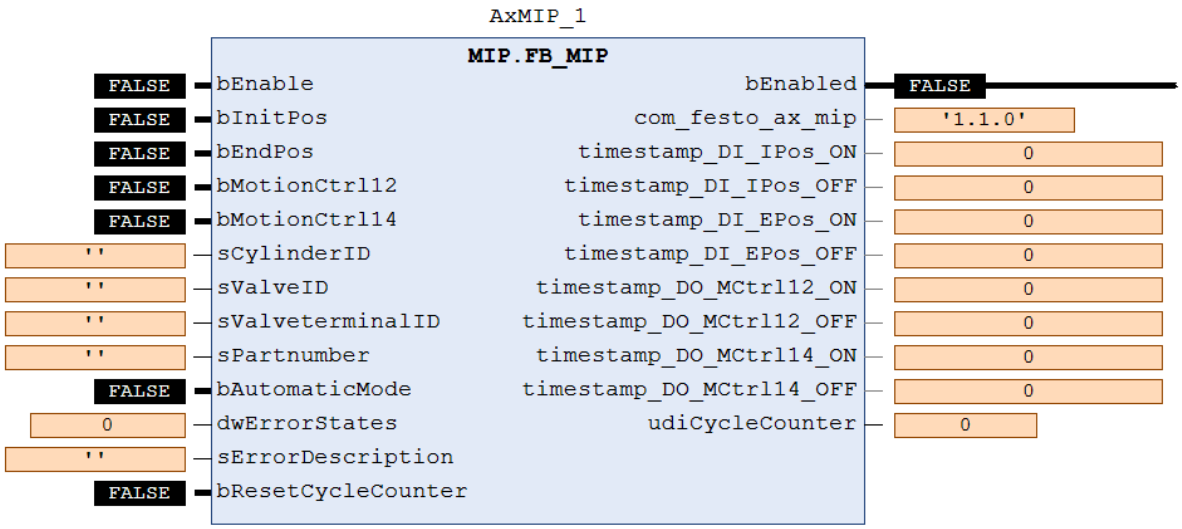
Step 5



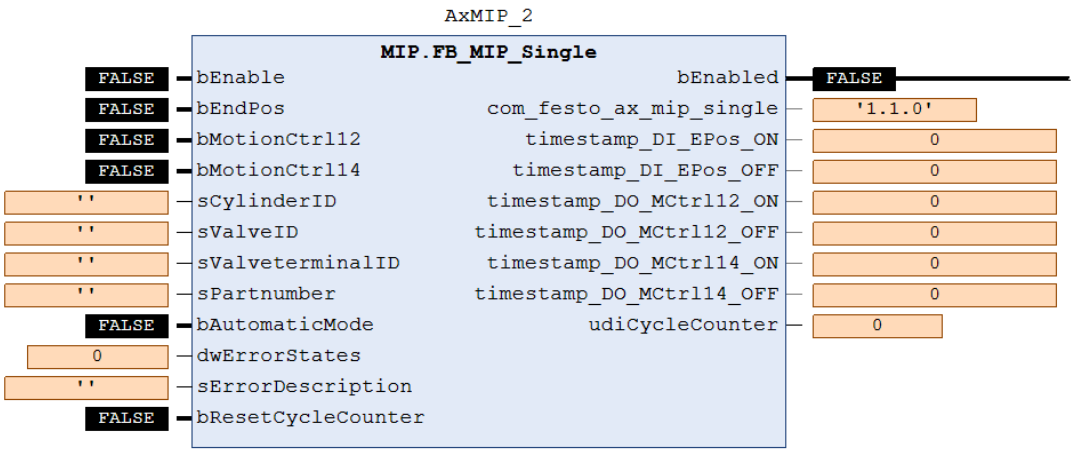
The called function block will be shown as above.

Step 6

1. Configure the instance of the function block as **AXMIP_1**.
2. Configure the I/O tags of the function block according to your application requirements.



Repeat the above steps to configure the **FB_MIP_Single** function block and configure the strings (sCylinderID, sValveID, sValveterminalID) according to your application requirements.



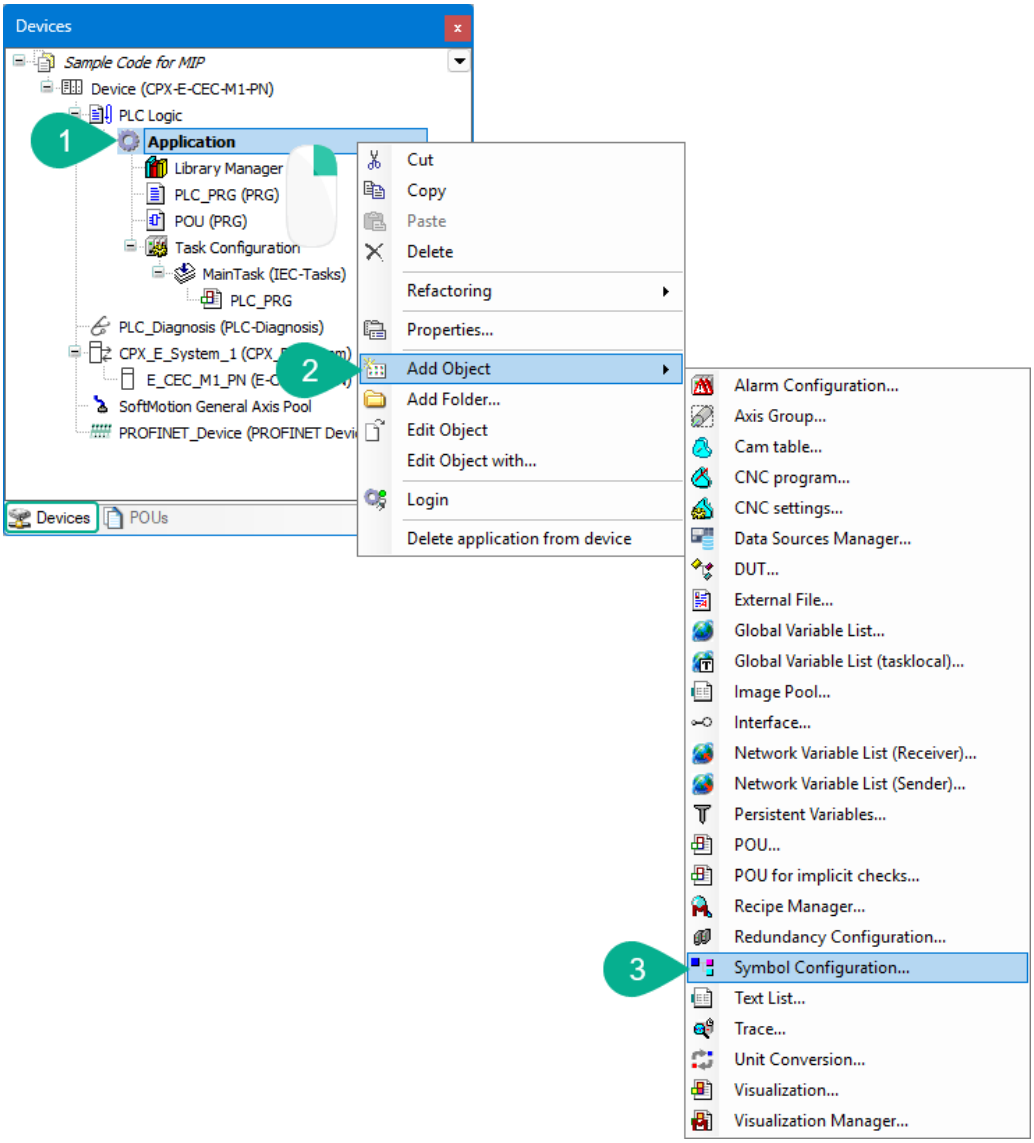
Note

- User shouldn't connect the output tag to **"com_festo_ax_mip"** and **"com_festo_ax_mip_single"** output variable. This variable will help to detect the cylinder from the connector. If we connect external tag to this output variable, the **AX MIP connector** will not detect the cylinder.

2.4 OPC UA Communication with CODESYS

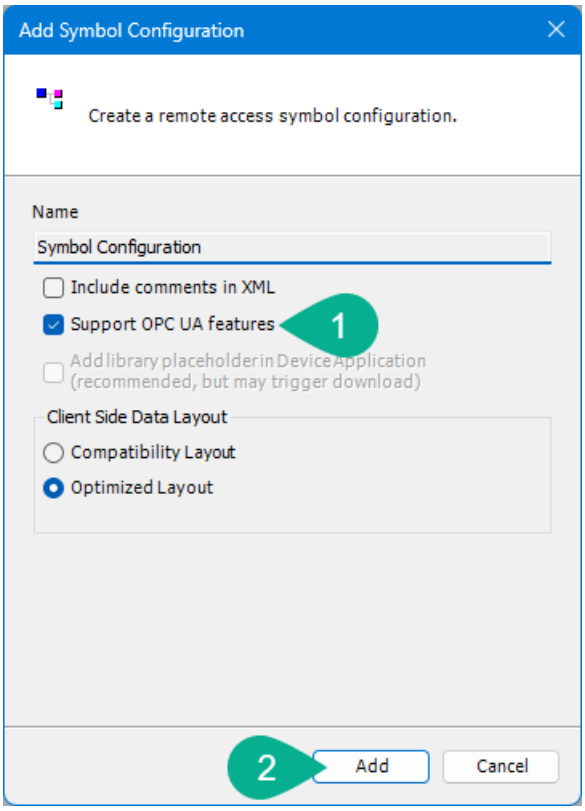
The following settings must be done when configuring Festo PLCs as OPC-UA server.

Step 1



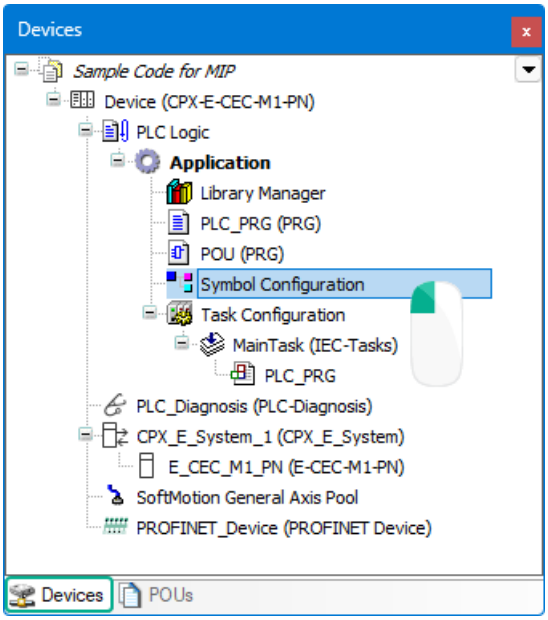
1. Right click the **Application** option under the **PLC Logic** from the **Device Tree** in the **Devices** toolbox.
2. Select the **Add Object** option from the available options.
3. Click **Symbol Configuration...** option to add the **Symbol Configuration** object to the project.

Step 2



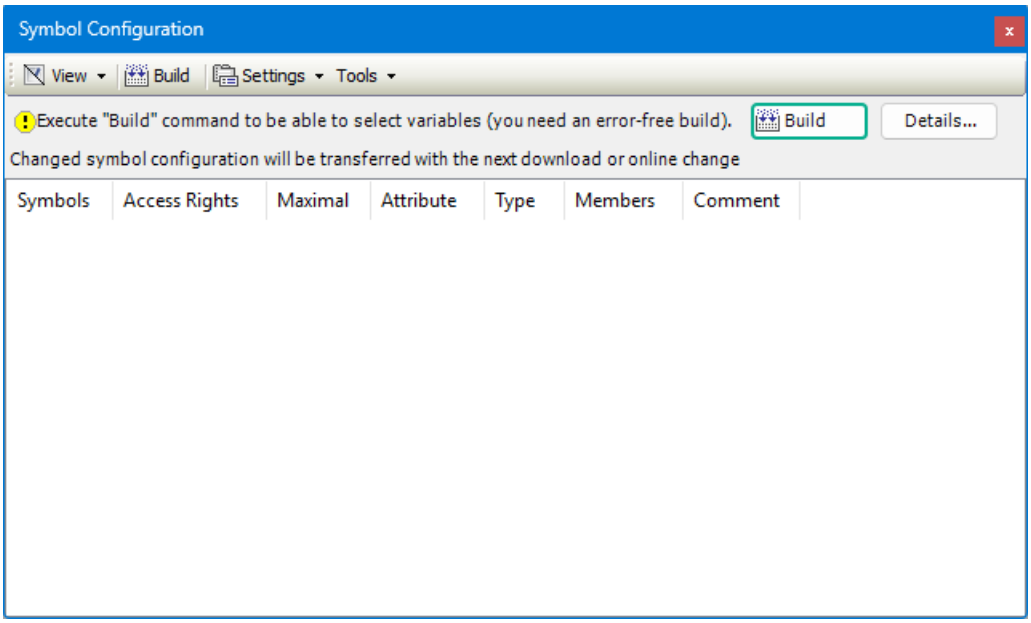
1. Select the **Support OPC UA Features** option.
2. Click the **Add** button to add the object.

Step 3



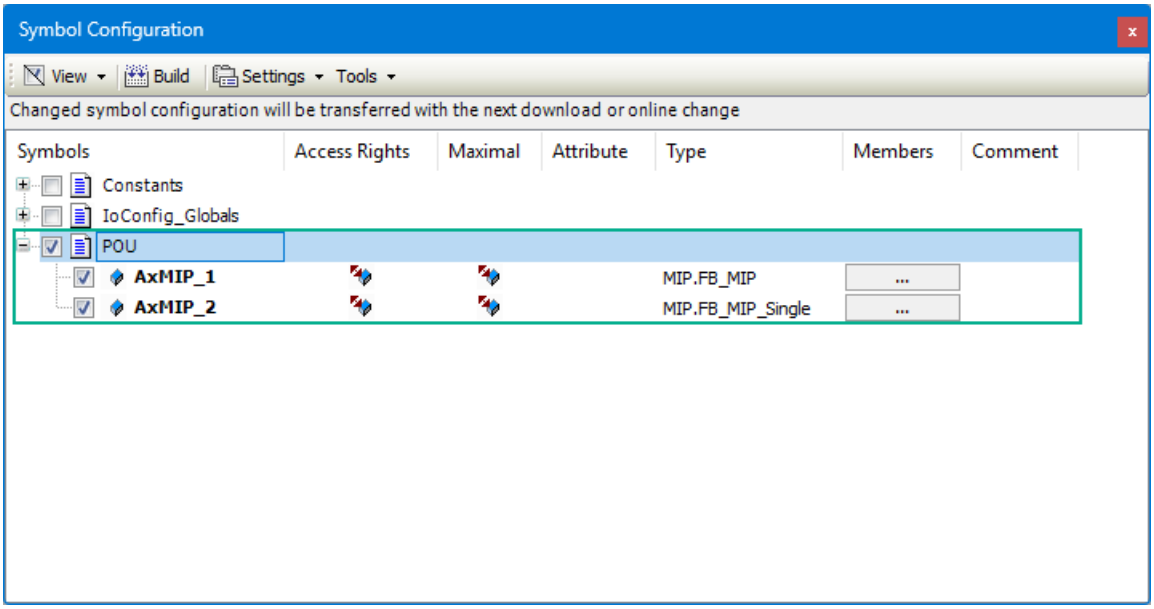
Double click the **Symbol Configuration** option under the **Application** from the **Device Tree** in the **Devices** toolbox to open the **Symbol Configuration** window.

Step 4



Click the **Build** button in the **Symbol Configuration** window, to view the variables available for the configuration.

Step 5



Select the **MIP Block Instances** that you want to change with an OPC UA client. Specify the access rights as per your application requirements and download the project to the controller.

Note

The configurations described above are for demonstration purposes only. Create and select the appropriate **MIP block** based on your application requirements, as described below:

- If the application uses **two separate sensors**, use the **FB_MIP** block.
- If the application uses **only one sensor**, use the **FB_MIP_Single** block.