

VTUX VABX Vacuum Valve with Mitsubishi iQ-R PLC through Modbus TCP using CPX-AP

This Application note details the procedure to Integrate VTUX Vacuum valves with Mitsubishi iQ-R PLC through Modbus TCP in GX Works3. Using Festo_VABX_MB Library user can Control, Teach & Parameterize VABX Valves.

VABX-A-S-VE-VBH
VABX-A-S-VE-VBL

Title VTUX VABX Vacuum Valve with Mitsubishi iQ-R PLC Modbus TCP using CPX-AP
Version 1.10
Document no. 100837
Originalen
AuthorFesto

Last saved 23.06.2026

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.

Table of Contents

1	Introduction	5
1.1	Hardware Used	6
1.2	Software Used	6
1.3	Topology Overview	7
2	Hardware Overview.....	8
2.1	VABX-A-S-EL-E12-API Overview	8
2.1.1	Connection Details.....	8
2.1.2	LED Display details of VABX-A-S-EL-...-API	9
2.2	VABX-A-S-VE-VBx Overview	9
2.2.1	LED Display details of VABX-A-S-VE-VBx displayed over LEDs on VUVX valve	10
2.2.2	LED Display Diagnostics details of VABX-A-S-VE-VBx displayed over LEDs on VUVX valve	11
3	Software Configuration.....	12
3.1	VTUX VABX Vacuum Valve Configuration in Festo Automation	12
3.1.1	CPX-AP-_-EP IP Address Configuration	12
3.1.2	Opening Device Web Page	14
3.1.3	Obtaining the Modbus Holding Register Address	15
3.1.4	Adding Device to the Project	16
3.1.5	Finding Module numbers	17
3.1.6	Connecting VABX	17
3.2	Creating New Project & Configuration of R04EN PLC in GX Works3	18
3.2.1	Creating New Project.....	18
3.2.2	Configuring PLC in GX Works3	21
3.2.3	Modbus TCP – Simple CPU Communication Configuration	21
4	Festo_VABX_MB Library Function Blocks Description	25
4.1	Connect block.....	25
4.1.1	Festo_VABX_Connect Block In-Out Variables.....	25
4.1.2	Festo_VABX_Connect Block Input Variables	25
4.1.3	Festo_VABX_Connect Block Output Variables	26
4.2	Control block	26
4.2.1	Festo_VABX_Control Block In-Out Variables	26
4.2.2	Festo_VABX_Control Block Input Variables.....	27
4.2.3	Festo_VABX_Control Block Output Variables	27
4.3	Teach Threshold Comparator block.....	28
4.3.1	Festo_VABX_Teach_Threshold Block In-Out Variables	28
4.3.2	Festo_VABX_Teach_Threshold Block Input Variables	28
4.3.3	Festo_VABX_Teach_Threshold Block Output Variables	29
4.4	Teach Window Comparator block	29
4.4.1	Festo_VABX_Teach_Window Block In-Out Variables	29
4.4.2	Festo_VABX_Teach_Window Block Input Variables	30
4.4.3	Festo_VABX_Teach_Window Block Output Variables	30

4.5	Teach Dynamic block.....	31
4.5.1	Festo_VABX_Teach_Dynamic Block In-Out Variables.....	31
4.5.2	Festo_VABX_Teach_Dynamic Block Input Variables	31
4.5.3	Festo_VABX_Teach_Dynamic Block Output Variables	32
4.6	Parameter Block.....	32
4.6.1	Festo_VABX_Parameter Block In-Out Variables	32
4.6.2	Festo_VABX_Parameter Block Input Variables.....	33
4.6.3	Festo_VABX_Parameter Block Output Variables.....	33
4.7	Parameter Copy Block	33
4.7.1	Festo_VABX_Parameter Block In-Out Variables	34
4.7.2	Festo_VABX_Parameter_Copy Block Input Variables.....	34
4.7.3	Festo_VABX_Parameter_Copy Block Output Variables	34
4.8	Structure	35
4.8.1	Structure (VABX_Ref)	35
4.8.2	Structure (Module_Parameters)	37
4.8.3	Structure (Device_Info)	38
4.8.4	Structure (Process_Data_Input).....	38
4.8.5	Structure (Process_Data_Output)	39
5	Configuration of Festo_VABX_MB Library Function Blocks in GX Works3.....	40
5.1	Importing Festo_VABX_MB Library.....	40
5.2	Global Label Declaration	43
5.3	Configuring Festo_VABX_MB Library function Blocks in GX Works3	44
5.3.1	Configuring VABX Blocks	44
6	Sample Code	49
6.1	Initial Communication	49
6.2	Connect Block.....	50
6.3	Control Block.....	50
6.3.1	Vacuum Generation	50
6.3.2	Ejector Pusle	51
6.3.3	Data Save.....	51
6.4	Threshold Comparator.....	52
6.5	Window Comparator.....	56
6.6	Dynamic TeachIn (Auto Drop Impulse Deactivated)	59
6.7	Dynamic TeachIn (Auto Drop Impulse Activated).....	63
6.8	Festo_VABX_Parameter Block	64
6.8.1	Parameter Read	64
6.8.2	Parameter Write	66
6.9	Festo_VABX_Parameter_Copy Block	67

1 Introduction

This application note outlines the procedure for integrating VTUX (serial) vacuum valves with Mitsubishi iQ-R PLC through Modbus TCP using the CPX-AP-I-EP head module. It also explains how to configure Modbus TCP for VTUX (serial) vacuum valves in GX Works3. Using the **Festo_VABX_MB** Library user can Control, Teach & Parameterize VABX Valves, it contains following function blocks.

1. **Festo_VABX_Connect** block - The Connect block provides module number, connection status, diagnostic data & CPX-AP related data to the other blocks, and it also contains the process data mapping.
2. **Festo_VABX_Control** block - The control block manages vacuum module operations by processing input commands for enabling, vacuum generation, ejector pulses, and data saving, while providing outputs for status, error reporting, pressure, process quality, and operational signals.
3. **Festo_VABX_Teach_Threshold** block - This function block facilitates the threshold comparator teach-in by handling control inputs along with Air Save function and signaling the teach-in progress, completion, status, and any errors encountered.
4. **Festo_VABX_Teach_Window** block - This function block facilitates the Window comparator teach-in by handling control inputs and signaling the teach-in progress, completion, status, and any errors encountered.
5. **Festo_VABX_Teach_Dynamic** block - This function block facilitates the Dynamic teach-in by handling control inputs and signaling the teach-in progress, completion, status, and any errors encountered.
6. **Festo_VABX_Parameter** block - This function block reads and writes module parameters and allows reading of device information.
7. **Festo_VABX_Parameter_Copy** block – This function block enables copying module parameters from a source module to multiple target modules.





1.1 Hardware Used

Device Name	Device Type	Version / Firmware
R04EN CPU	Mitsubishi PLC	1.280 & above
CPX-AP-A-EP / CPX-AP-I-EP	Modbus TCP Head module	V1.7.4 & above
VABX-A-S-EL-E12-API-SHUH-XL	Manifold sub-base	-
VABX-A-S-VE-VBL or VABX-A-S-VE-VBH	Manifold sub-base for vacuum	1.5.4 or higher
VUVX-BK10-T32CV-A1ZH-F-1T1L	Vacuum standard valve KV (2x3/2 NC)	Version Free
AP Communication Cable	NEBC-D8G4-ES-0.5-N-S-D8G4-ET	Version Free
M12 – RJ45 Ethernet Cable	NEBC-D12G4-ES-0.5-S-D12G4-ET	Version Free
Ethernet Cable	-	Version Free

Table 1.1: Hardware used

1.2 Software Used

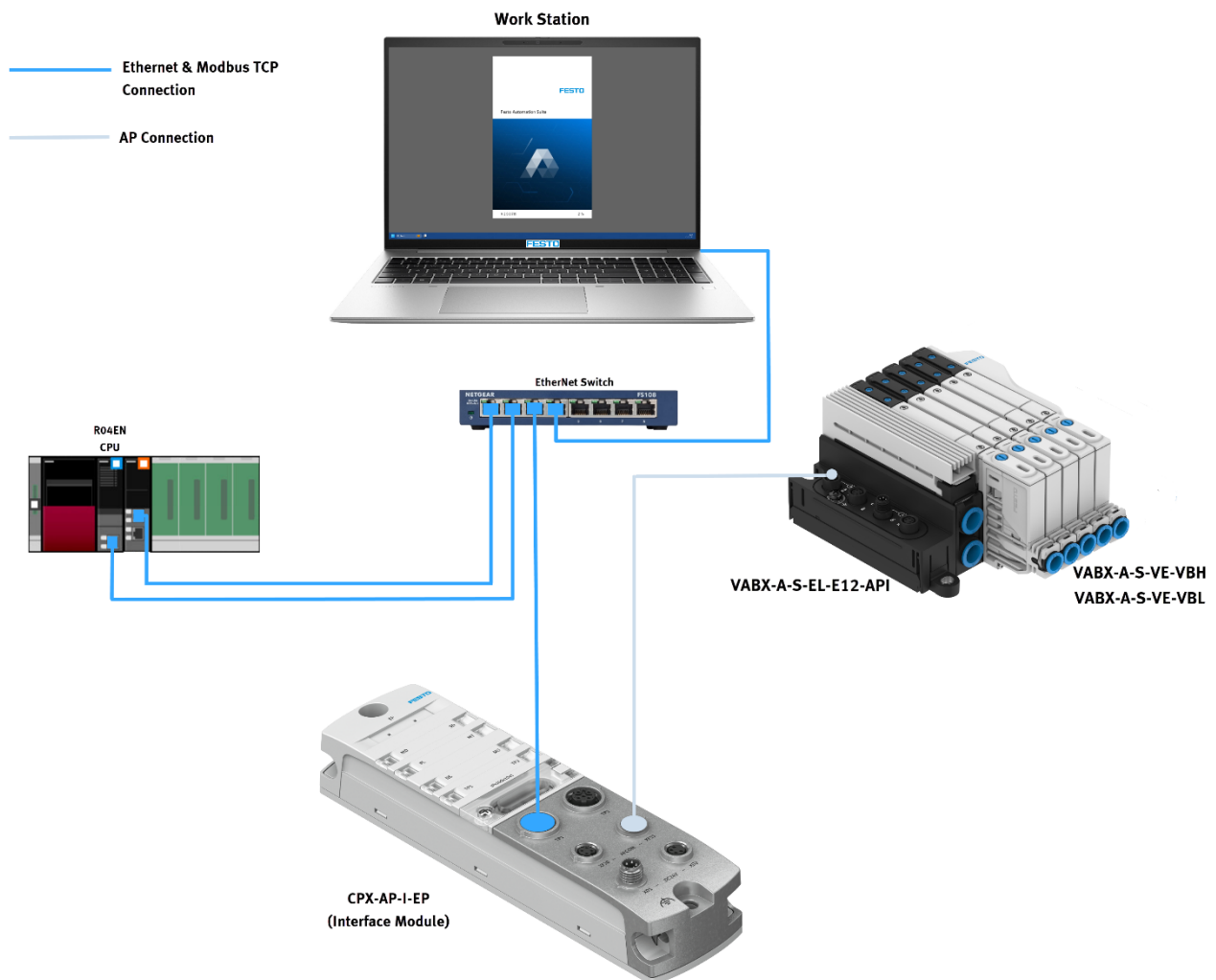
Software Name	Version Software / Firmware
GX Works3	1.123D
Festo Automation Suite	2.10.0.557 & above
Library (Festo_VABX_MB)	1.0

Table 1.2: Software used

**Note**

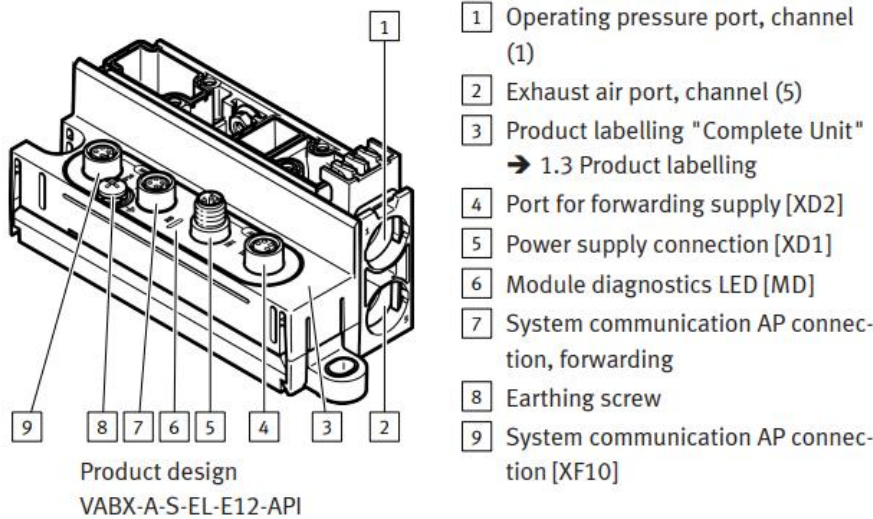
- Above mentioned Hardware is w.r.t this Application note and it may vary depends on user requirement
- This application is applicable to both CPX-AP-A-EP and CPX-AP-I-EP, and is therefore referred to collectively as CPX-AP-_-EP.
- For this Application note CPX-AP-I-EP is considered.

1.3 Topology Overview



2 Hardware Overview

2.1 VABX-A-S-EL-E12-API Overview



2.1.1 Connection Details

Power supply connection [XD1]		
M8 plug, 4-pin, A-coded		Signal
	1	+24 V DC logic supply PS
	2	0 V DC load supply PL
	3	0 V DC logic supply PS
	4	+24 V DC load supply PL

Power supply connection

Connection for voltage forwarding [XD2]		
M8 socket, 4-pin, A-coded		Signal
	1	+24 V DC logic supply PS
	2	0 V DC load supply PL
	3	0 V DC logic supply PS
	4	+24 V DC load supply PL

Connection for voltage forwarding

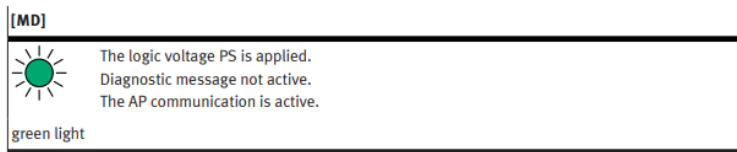
Connection for system communication [XF10]		
Socket M8, 4-pin, D-coded		Signal
	1	TX- Transmitted data -
	2	RX+ Received data +
	3	TX+ Transmitted data +
	4	RX- Received data -

Connection for system communication

Connection for system communication forwarding [XF20]		
Socket M8, 4-pin, D-coded		Signal
	1	RX- Received data -
	2	TX+ Transmitted data +
	3	RX+ Received data +
	4	TX- Transmitted data -

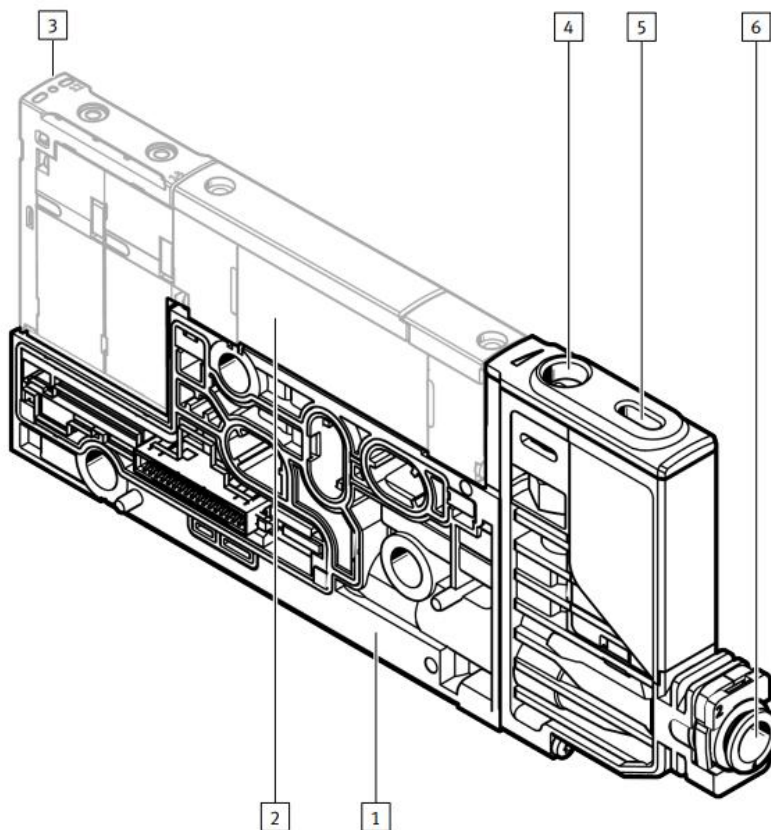
Connection for system communication forwarding

2.1.2 LED Display details of VABX-A-S-EL-...-API



Response of the LED display on the manifold sub-base VABX-A-S-EL-...-API after fault-free commissioning

2.2 VABX-A-S-VE-VBx Overview



Product design for manifold sub-base for vacuum VABX-AS-VE with valve (example)

- | | |
|---|--|
| <p>1 Manifold sub-base for vacuum</p> <p>2 Valve VUVX-BK10-...CV-..., mandatory</p> <p>3 LED display for switching positions and operating status of the manifold sub-base for vacuum and the valve</p> | <p>4 Flow control screw to adjust the intensity of the ejector pulse</p> <p>5 Silencer, integrated, open design</p> <p>6 Vacuum working port, duct (2)</p> |
|---|--|

2.2.1 LED Display details of VABX-A-S-VE-VBx displayed over LEDs on VUVX valve

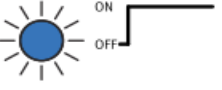
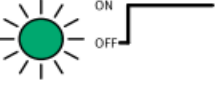
Operating status	LED 14 ¹⁾	LED 12 ²⁾	Explanation
"Generate vacuum"			Solenoid coil 14 actuation is pending. Vacuum generation is active.
"Air saving"			Solenoid coil 14 actuation is pending. The air saving function is active.
"Pressurise" (ejector pulse)			Solenoid coil 12 actuation is pending. The vacuum is reduced faster by pressurisation.
"Normal position"			Actuation of the solenoid coils 12 and 14 is not pending: status after the end of the "pressurise" signal (ejector pulse)
Two-sided actuation			Actuation of the solenoid coils 12 and 14 is pending. Vacuum generation or ejector pulse may dominate depending on the specific application and configuration.

1) vacuum generation

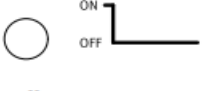
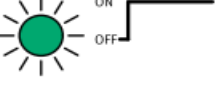
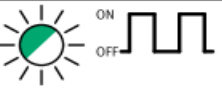
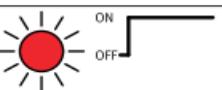
2) ejector pulse

Operating status of the manifold sub-base for vacuum

2.2.2 LED Display Diagnostics details of VABX-A-S-VE-VBx displayed over LEDs on VUVX valve

LED	Meaning	Possible error
Signal state of solenoid coils 12 and 14		
 off	A signal to switch the valve is not pending: logic 0	–
 blue light	LEDs of solenoid coils 12 and 14 light: - The signal to switch the valve is pending: logic 1 - The vacuum generation or the ejector pulse is active.	–
 green light	LED of solenoid coil 14 on: - The signal to switch the valve is pending: logic 1 - The vacuum is reached and the air saving function is active.	–

LED displays of the valves

Diagnostics LED	Meaning and possible errors
 off	- OK – diagnostics not active - power supply not active
 green light	Valve ID chip is mounted (possible depending on configuration).
 flashing green (2 Hz)	Module identification
 flashing green (0.5 Hz)	Diagnostics active, severity "Information", e.g. load voltage supply PL switched off.
 red light	Diagnostics active, severity "Error", e.g. AP communication/system communication not available/interrupted.
 fast flashing red (2 Hz)	Module ramp-up not yet completed. System communication not yet initialised.
 flashing red (1 Hz)	Diagnostics active, severity "Warning", e.g. parameterisation error

Signal state of the diagnostics LED

3 Software Configuration

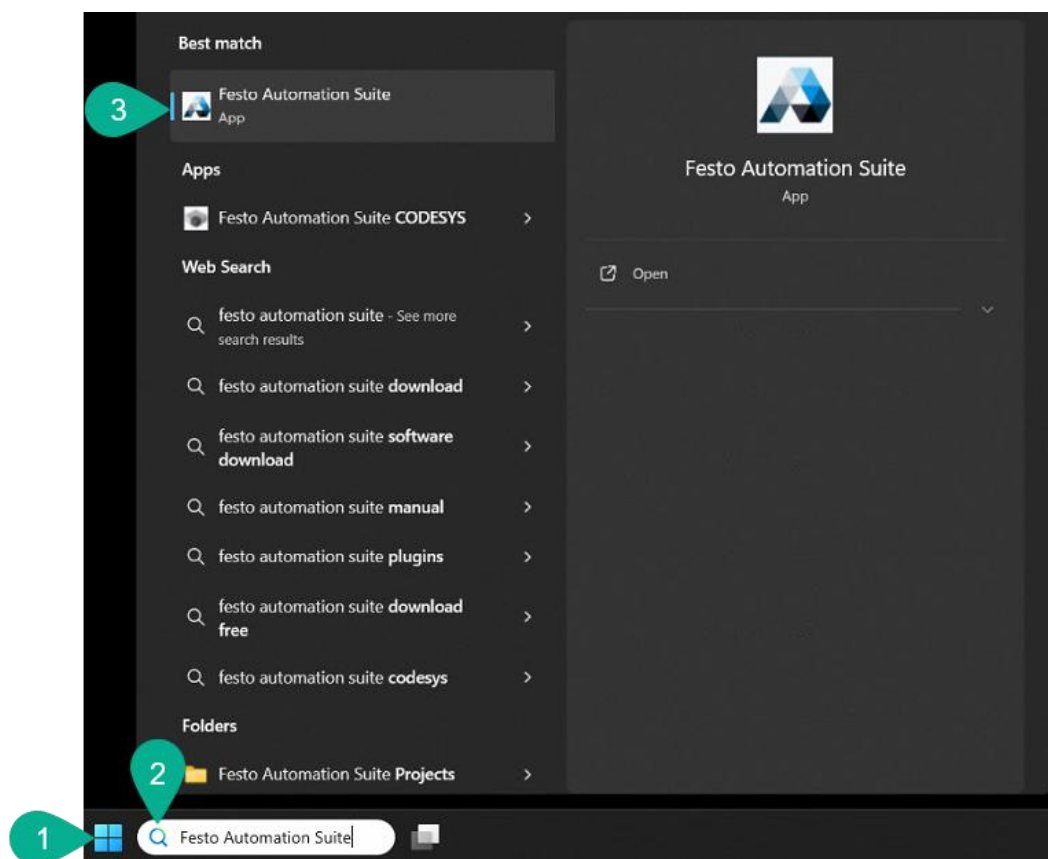
This chapter describes Software configuration of VTUX VABX valve in Festo Automation Suite and Modbus TCP configuration of CPX-AP-_-EP and VABX valves in GX Works3.

3.1 VTUX VABX Vacuum Valve Configuration in Festo Automation

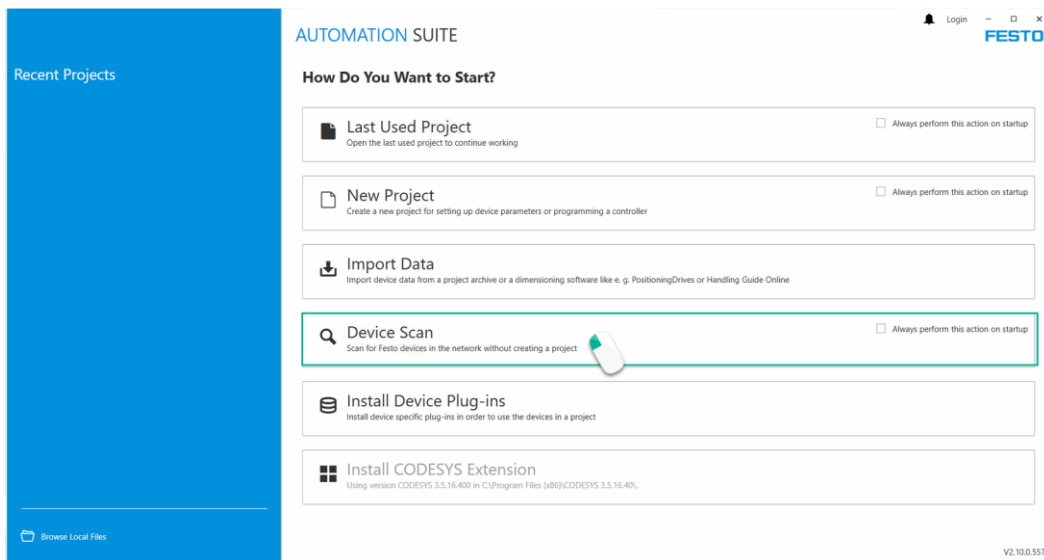
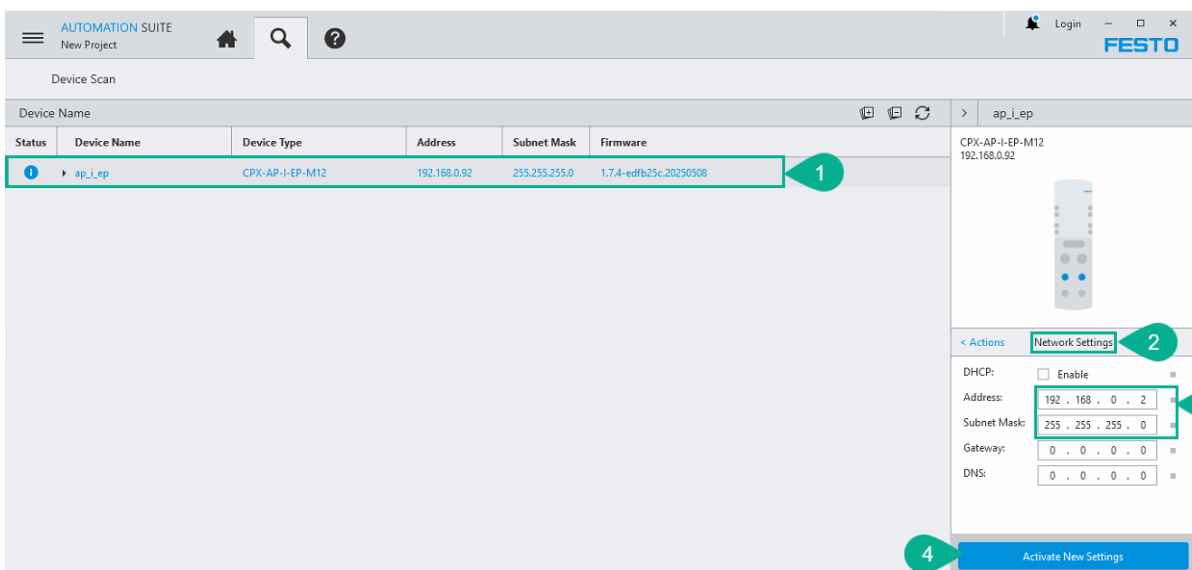
This chapter provides a detailed procedure for configuring VABX with CPX-AP-_-EP in Festo Automation Suite.

3.1.1 CPX-AP-_-EP IP Address Configuration

Step 1

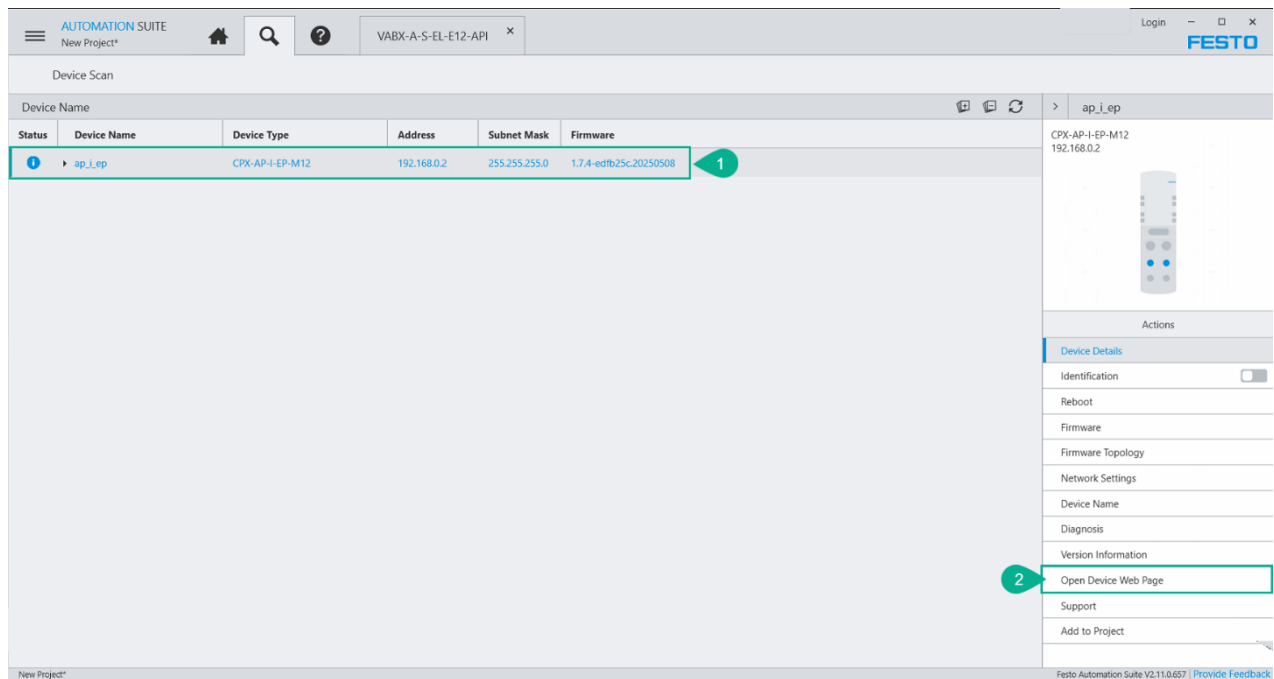


1. Click the **Search** button on Taskbar
2. Enter **Festo Automation Suite** in search box or user can navigate to **Start > Festo Software > Festo Automation Suite**
3. Click on the **Festo Automation Suite** App

Step 2Click on **Device Scan****Step 3**

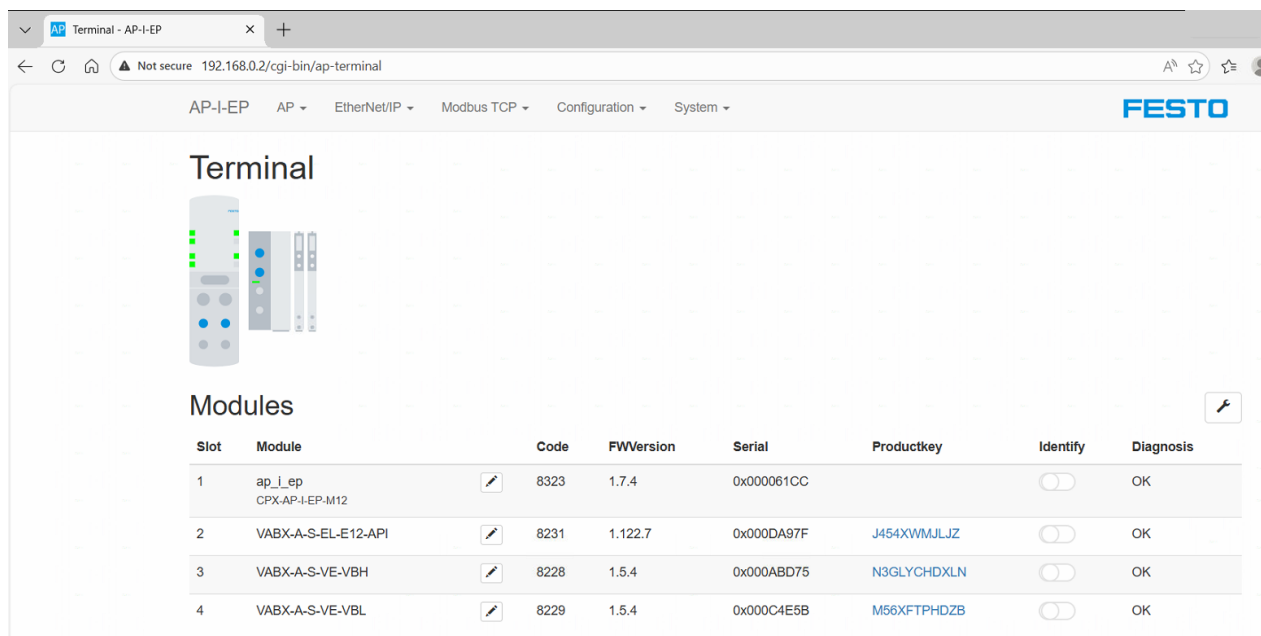
1. Select **CPX-AP-_-EP** from the list
2. Select **Network Settings** in **Actions**
3. Enter **IP Address** and **Subnet Mask**. User must make sure that IP Address of PLC and CPX-AP-_-EP are in same domain
4. Click on **Active New Settings**

3.1.2 Opening Device Web Page



1. Select **CPX-AP-_-EP** for Device Name list
2. Click on **Open Device Web Page**

Once Device Web Page opens it looks as shown in below image. It lists out the actual Hardware configuration.



3.1.3 Obtaining the Modbus Holding Register Address

Holding Register View

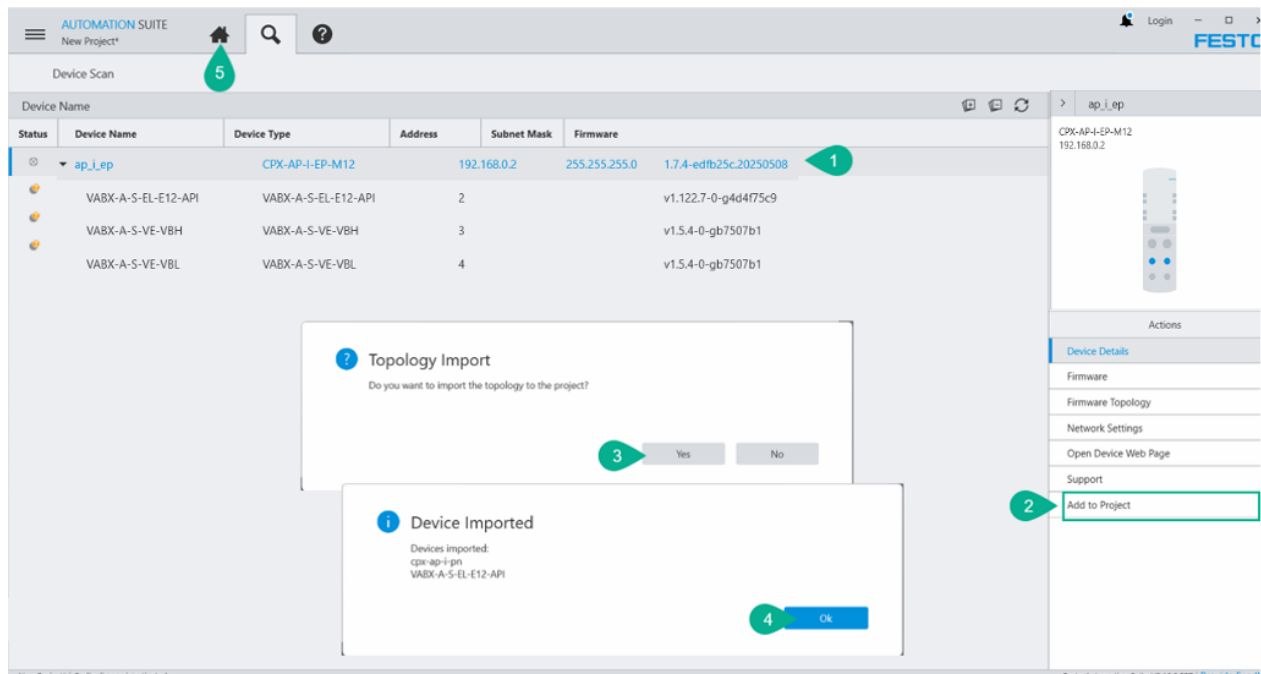
Copy CSV Search:

Register	Offset (bit)	Bit length	Module	Channel	Datatype	Name
3 Outputs						
0	0	16	3	0	USINT[2]	Module 3 - VABX-A-S-VE-VBH - Output 0
1	0	16	4	0	USINT[2]	Module 4 - VABX-A-S-VE-VBL - Output 0
4 Inputs						
5000	0	16	3	0	INT	Module 3 - VABX-A-S-VE-VBH - Channel 0
5001	0	8	3	0	USINT	Module 3 - VABX-A-S-VE-VBH - Input 0
5001	8	8	3	0	USINT	Module 3 - VABX-A-S-VE-VBH - Process quality 0
5002	0	16	4	0	INT	Module 4 - VABX-A-S-VE-VBL - Channel 0
5003	0	8	4	0	USINT	Module 4 - VABX-A-S-VE-VBL - Input 0
5003	8	8	4	0	USINT	Module 4 - VABX-A-S-VE-VBL - Process quality 0
5 Parameter Mailbox						
10000	0	16	-	-	UINT	Parameter Mailbox - Module Number
10001	0	16	-	-	UINT	Parameter Mailbox - AP Parameter ID Low Word
10002	0	16	-	-	UINT	Parameter Mailbox - AP Parameter Instance
10003	0	16	-	-	UINT	Parameter Mailbox - Execute / Command
10004	0	16	-	-	UINT	Parameter Mailbox - Data Length
10005	0	16	-	-	UINT	Parameter Mailbox - AP Parameter ID High Word
10010 - 10521	0	8192	-	-	Array Of Byte	Parameter Mailbox - Data
6 Diagnosis						
11000 - 11001	0	32	-	-	DWORD	Global Diagnosis State
11002	0	16	-	-	INT	Count of currently active diagnosis
11003	0	16	-	-	INT	Module which has latest diagnosis
11004 - 11005	0	32	-	-	DINT	Latest Diagnosis Code
11006	0	8	1	0	SINT	Diagnosis - Module 1
11006	8	8	1	0	SINT	Diagnosis - Module 1 - Submodule
11016 - 11017	0	32	2	0	DINT	Diagnosis - Module 2 - Diagnosis Code
11018	0	8	3	0	SINT	Diagnosis - Module 3
11018	8	8	3	0	SINT	Diagnosis - Module 3 - Submodule
11019	0	8	3	0	SINT	Diagnosis - Module 3 - Channel
11019	8	8	3	0	SINT	Diagnosis - Module 3 - Present State
11020 - 11021	0	32	3	0	DINT	Diagnosis - Module 3 - Module Diagnosis State
11022 - 11023	0	32	3	0	DINT	Diagnosis - Module 3 - Diagnosis Code
11024	0	8	4	0	SINT	Diagnosis - Module 4
11024	8	8	4	0	SINT	Diagnosis - Module 4 - Submodule
11025	0	8	4	0	SINT	Diagnosis - Module 4 - Channel
11025	8	8	4	0	SINT	Diagnosis - Module 4 - Present State
11026 - 11027	0	32	4	0	DINT	Diagnosis - Module 4 - Module Diagnosis State
11028 - 11029	0	32	4	0	DINT	Diagnosis - Module 4 - Diagnosis Code

1. In Device Web Page click on **Modbus TCP** drop down option
2. Click on **Holding Register View**
3. Note down the **Outputs Register** and **Bit length**
4. Note down the **Inputs Register** and **Bit length**
5. Note down the **Parameter Mailbox Register** and **Bit length**
6. Note down the **Diagnosis Register** and **Bit length**

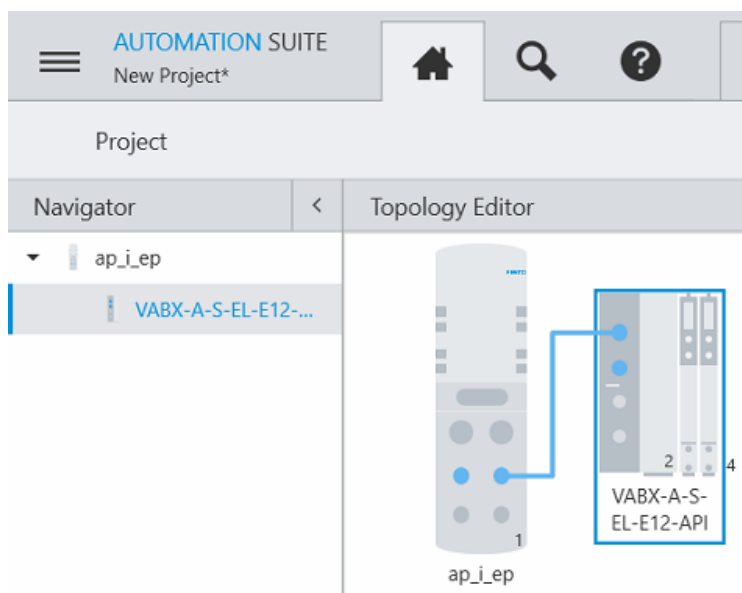
**Note**

- The bit length considered for Parameter Mailbox is 36 Registers (10000 to 10035).
- For this Application Note Module 4 is considered for Configuration

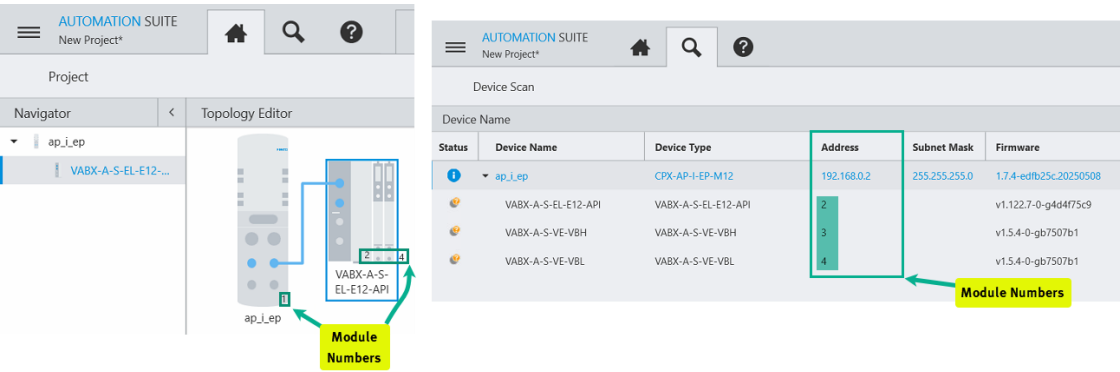
3.1.4 Adding Device to the Project

1. Select **CPX-AP-_-EP** module from **Device Name** list
2. Click on **Add to Project**
3. Click on **Yes** button of **Topology Import** pop-up window
4. Click on **Ok** button of **Device Imported** pop-up window
5. Click on **Home** button for imported topology

Once topology is imported it looks as shown in below image.



3.1.5 Finding Module numbers



User can find the Module Numbers as shown in above image.

Note

- The above imported configuration is for this Application note, and it may vary based on User Hardware configuration.

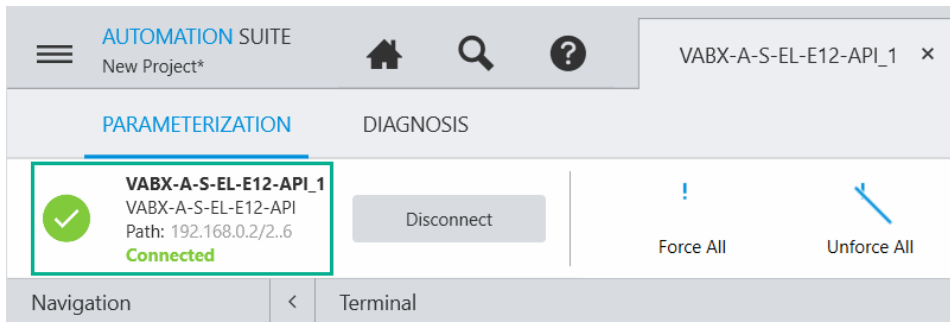
3.1.6 Connecting VABX

Step 1

#	Order Code	Information	Description
2	VABX-A-S-EL-E12-API	AP-I Interface for serial VTUX valve terminals of subbase sizes 1 and 2 to communicate with a CPX-AP System via a cable.	VABX-A-S-EL-E12-API
3	VABX-A-S-VE-VBH	Vacuum generator with integrated pressure sensor for serial VTUX valve terminals, grid 12,5 (size 2), high vacuum.	VABX-A-S-VE-VBH
4	VABX-A-S-VE-VBL	Vacuum generator with integrated pressure sensor for serial VTUX valve terminals, grid 12,5 (size 2), high suction flow rate.	VABX-A-S-VE-VBL

1. Double click on **VABX-A-S-EL-E12-API** module
2. Click on **Connect** button (Information of devices is highlighted in green box)

Once connected it looks as shown in below image.

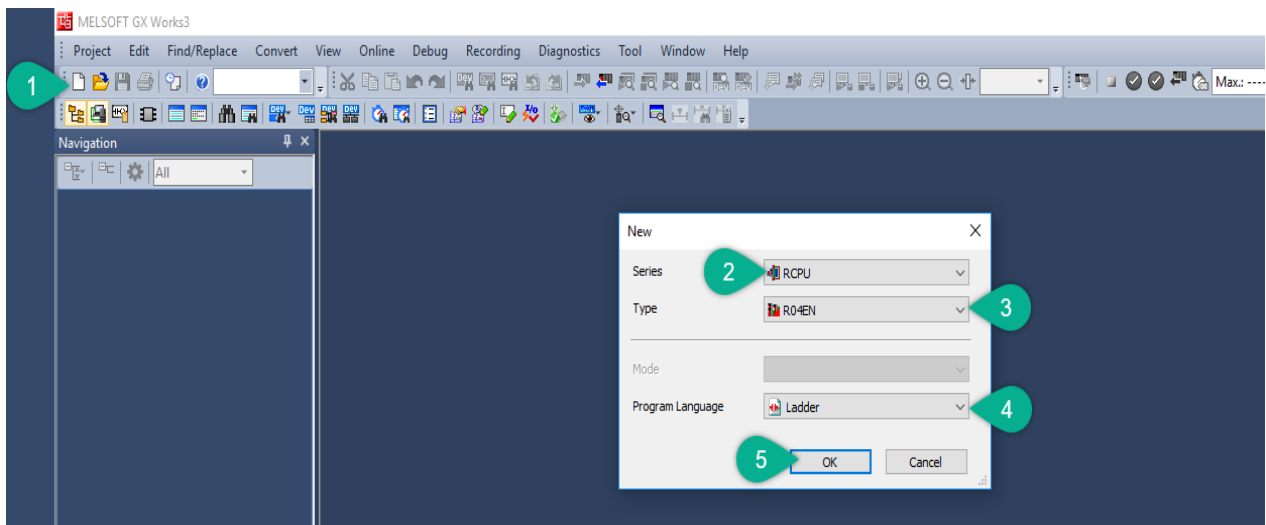


3.2 Creating New Project & Configuration of R04EN PLC in GX Works3

3.2.1 Creating New Project

Step 1

Open MELSOFT **GX Works3** software



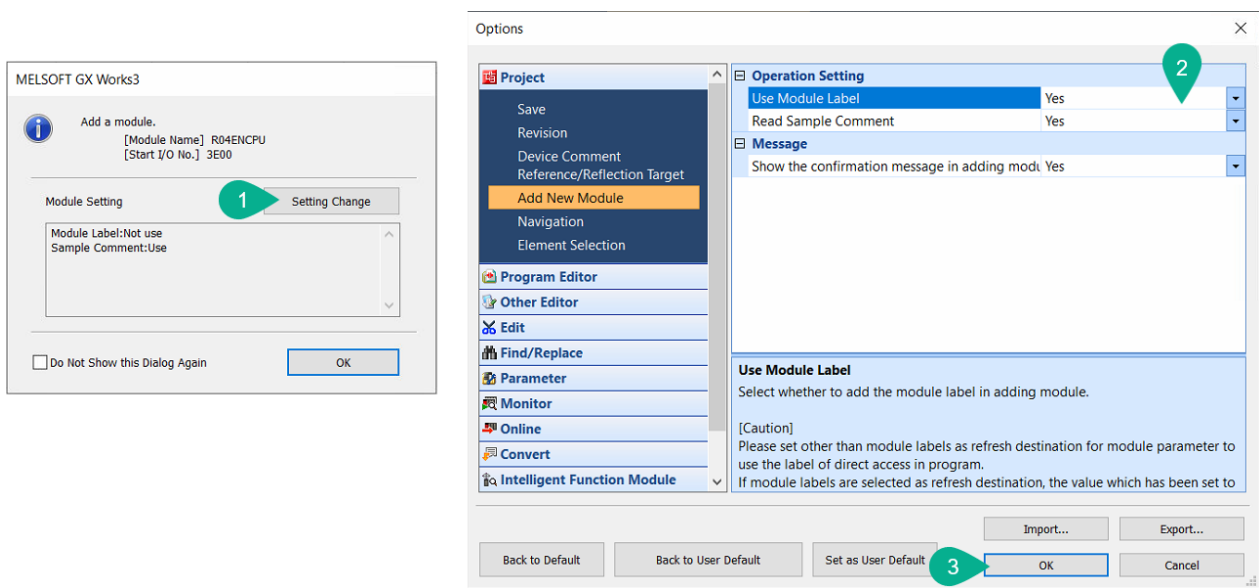
1. Select **New Project** to create a new project
2. Select **RCPU** from **Series** drop down option
3. Select your CPU model from **Type** drop down option. Example: R04EN
4. Select **Ladder** as programming language from **Programming Language** drop down option or select preferred programming language, select your settings or use the ones in the example image
5. Confirm your selection with the button **OK**



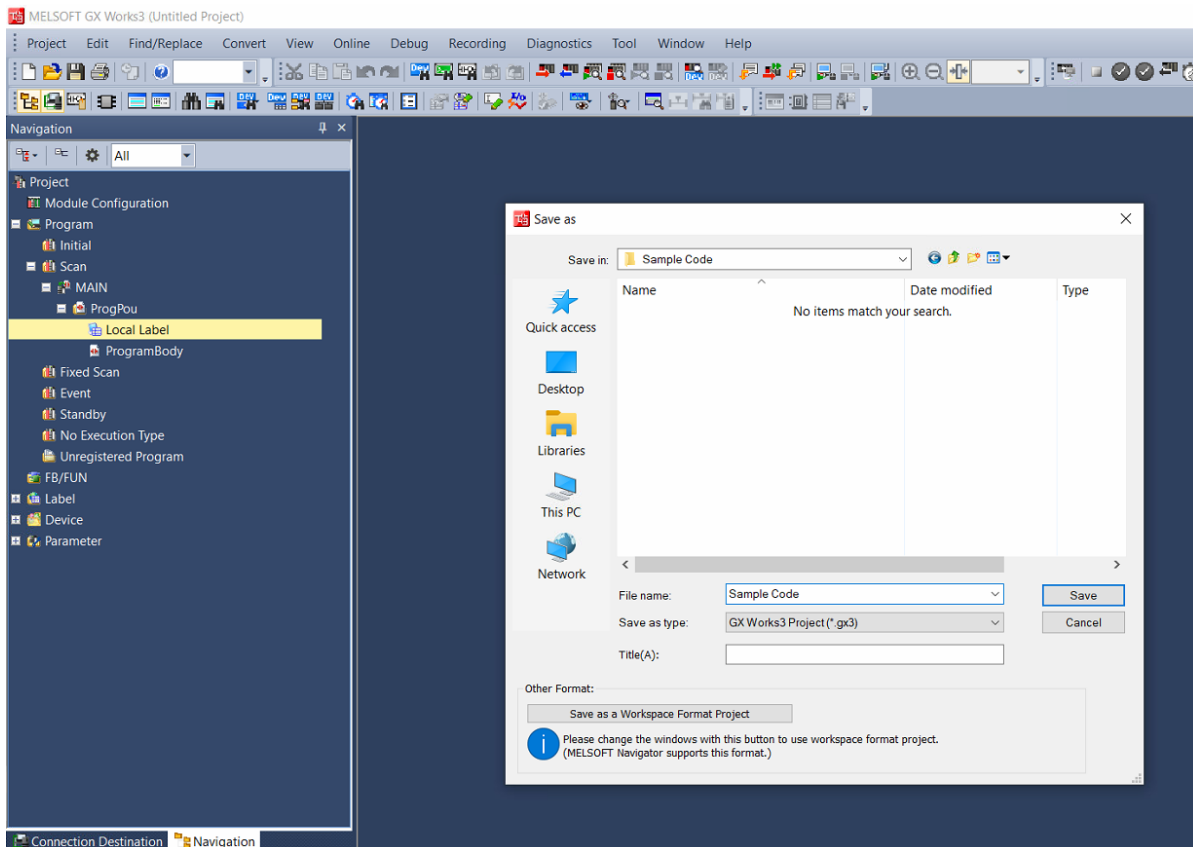
Note

- Users can also create **New Project** by clicking shortcut key **CTRL+N** or **ICON** .
- CPU model and Programming language can be select as per user requirement.



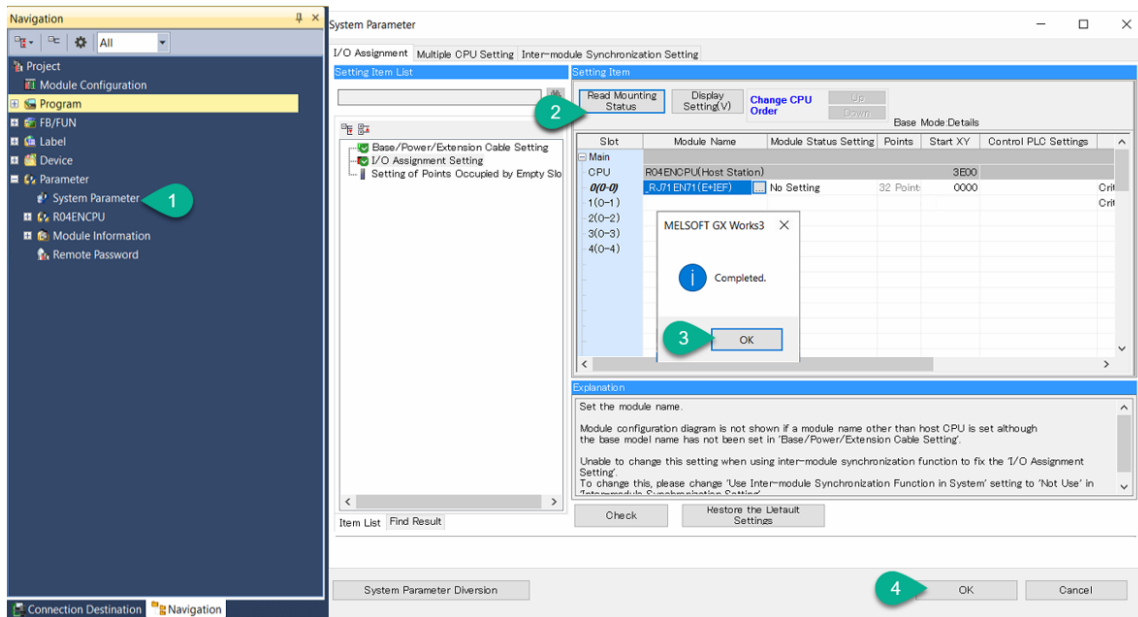
Step 2

1. Click on **Setting Change**
2. Select **Yes** in **Use Module Label**
3. Click **OK**

Step 3

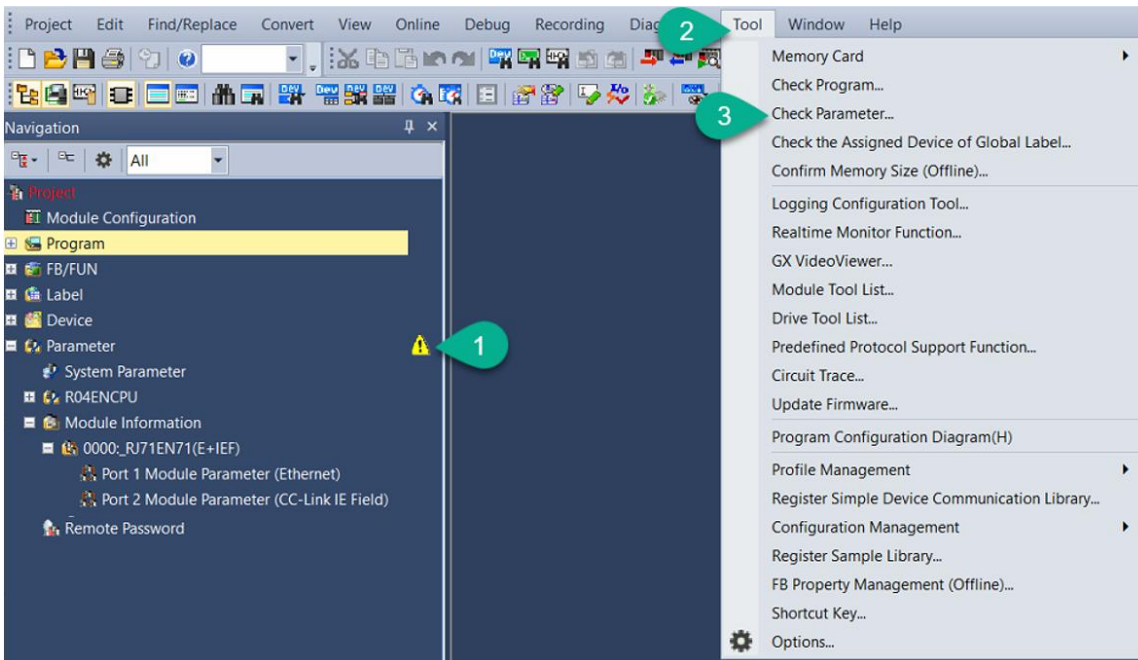
Save the project as User required

Step 4



1. Click on **System Parameter** under Parameter tree.
2. Click on **Read Mounting Status** to get the Hardware Module configuration.
3. Click on **OK** in Completed pop up.
4. Press **OK** to complete the process.

Check Parameter



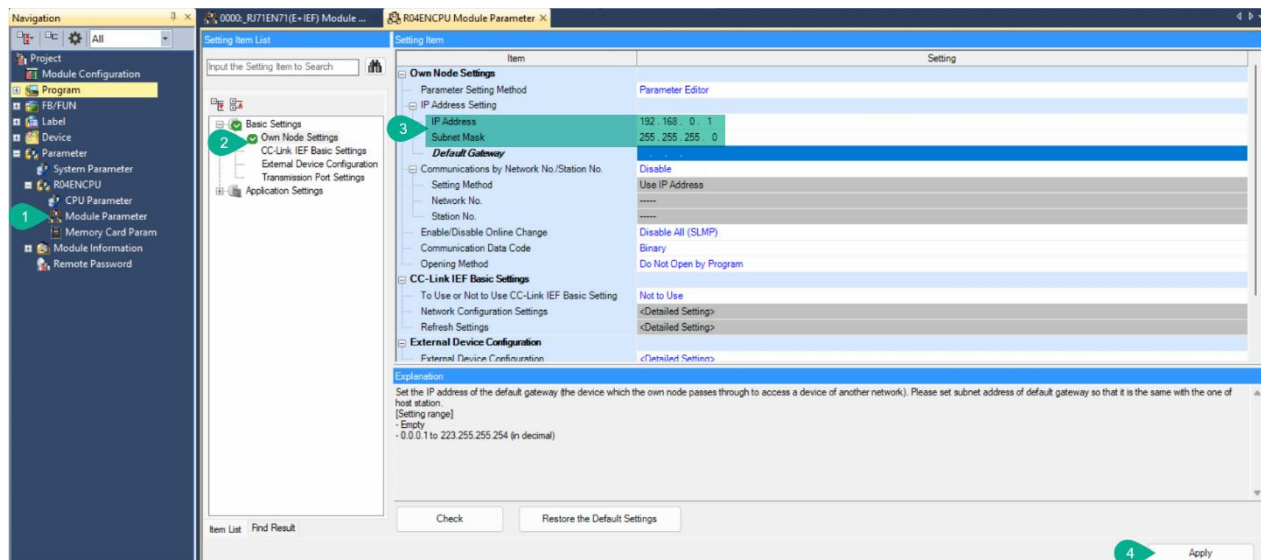
1. After reading Module Configuration, **Check parameter caution** ⚠ symbol may appear
2. Click on **Tool**
3. Click on **Check Parameters**

Make sure that Caution symbol ⚠ is cleared

3.2.2 Configuring PLC in GX Works3

Step 1

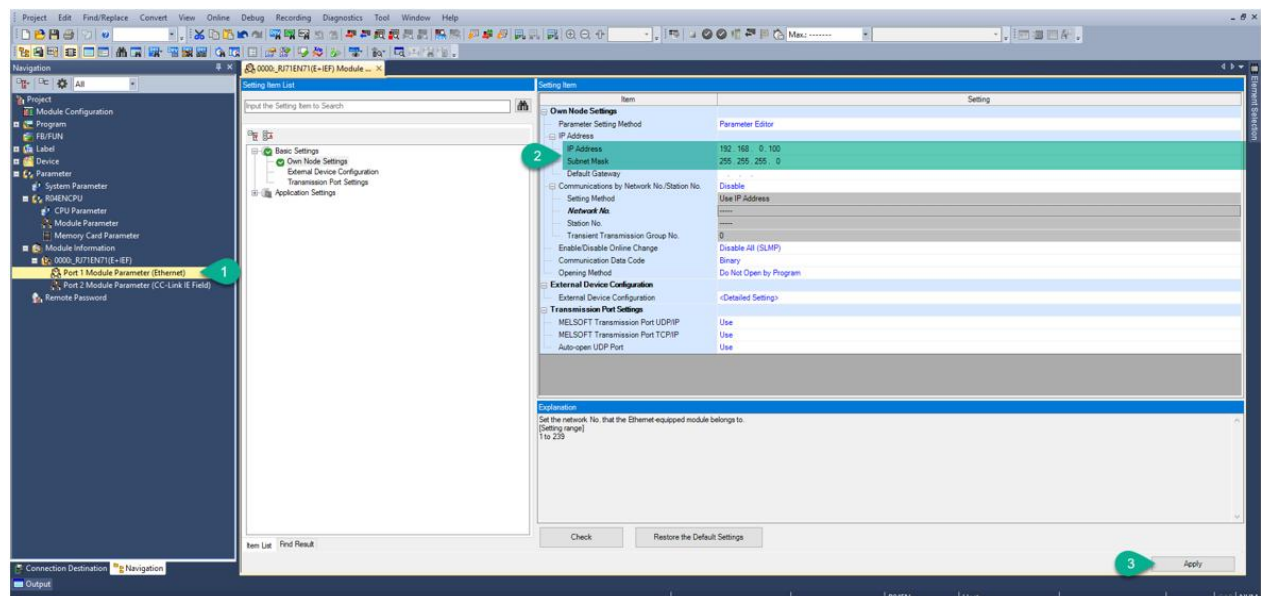
PLC IP Address setting



1. Double click on **Module Parameter** under **R04ENCPU**
2. Click on **Own Node Settings** under **Basic Settings**
3. Enter **IP Address & Subnet Mask**
4. Click **Apply** to confirm the setting

3.2.3 Modbus TCP – Simple CPU Communication Configuration

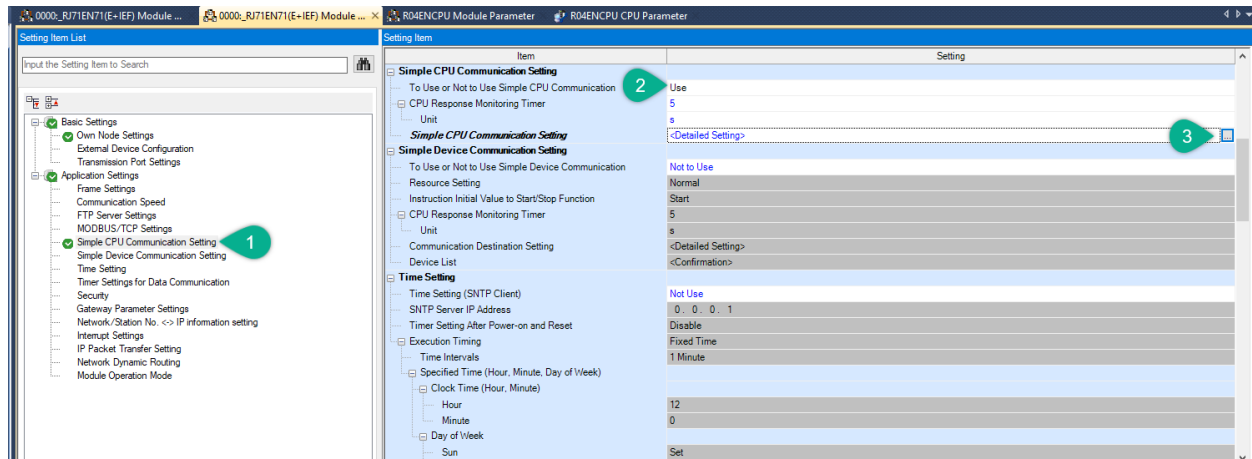
Step 1



Software Configuration

1. Under Module information expand RJ71EN71(E+IEF) and click on **Port1 Module Parameter (Ethernet)**
2. Enter the **IP Address & Subnet Mask** ; Make sure that CPX-AP-EP system IP Address is in the same domain
3. Click **Apply** button to confirm the configuration

Step 2



1. Click on **Simple Communication Setting**
2. Select **Use** in To Use or Not to Use Simple CPU Communication
3. Click on **Detail Setting** in Simple CPU Communication setting

Step 3

The screenshot shows the 'Setting Item' configuration window. At the top, there are fields for 'Latency Time' (0 s) and 'Communication Destination Filter' (Show All). Below these are 'Import/Export of Settings' buttons. The main table has columns for 'Setting No.', 'Communication Pattern', 'Communication Setting: Execution Interval(ms)', 'Communication Destination (IP Address)', and 'Word Device'. A dialog box titled 'Communication Destination Setting' is open, showing fields for 'Item', 'Setting', 'IP Address Input Format' (DEC), 'Device Type' (MODBUS/TCP-compatible device), 'IP Address' (192.168.0.2), 'TCP/UDP' (TCP), 'Port No.' (502), 'Host Station Port No.' (2000), and 'Options (Hexadecimal)' (0). Numbered callouts 1 through 9 highlight specific steps in the configuration process.

1. In **Communication Pattern**, select **Read or Write**
2. In **Communication setting**, select **Fixed Interval** and **user preferred interval (ms)**
3. Click on **Please Set** in Source column (Read Pattern)/ Destination Column (Write Pattern) of Communication Destination (IP Address)
4. In Device Type select **MODBUS TCP Compatible Device**.
5. In **IP Address** set **CPX-AP_-EP-M12 Head Module IP Address**
6. In **Host Station Port No** Set **user preferred nonstandard Port Number (Eg 2000)**
7. Click **OK**
8. Word Device (Source):
 - a. If **Communication Pattern-Read** then select the Type **Holding Register** in **Source** of **Word Device**, and assign the respective Modbus Register **Start** and **End** address referring Web configurator(Refer [Chapter 3.1.3](#))
 - b. if **Communication Pattern-Write**, then select Register type (**D/W/R/SW/SD**) in **Source** of **Word Device**, and assign **Start** and **End** device addresses where data to be write from PLC
9. Word Device (Destination):
 - a. If **Communication Pattern-Read**, then select Register type (**D/W/R/SW/SD**) in **Destination** of **Word Device** and assign **Start** and **End** device addresses where data to be Read at PLC
 - b. if **Communication Pattern-Write** then select the Type **Holding Register** in **Destination** of **Word Device** and enter the respective Modbus Register **Start** and **End** address referring Web configurator(Refer [Chapter 3.1.3](#))

Step 4

Setting Item

Latency Time: 0 s (0s to 255s)

Communication Destination Filter: Show All ☐ Hide Rows Where Setting Has Not Been Set

Import/Export of Settings: Import Export

Setting No.	Communication Pattern	Communication Setting Execution Interval (ms)	Communication Destination (IP Address)		Source		Destination		Type	Start	End	Type	Start	End	Communication Time-out Period (ms)	Communication Retry Count	Monitoring Time All Error (s)	Comment
			Source	Destination	Type	Start	End	Type										
1	Read	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	Holding Register	5002	5003	D	1000	1001	D	1000	1001	1000	3	30	PDI (VABX to PLC)
2	Write	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	D	1100	1100	Holding Register	1	1	D	1000	1001	1000	3	30	PDO (PLC to VABX)
3	Read	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	Holding Register	10000	10035	D	1005	1040	D	1000	1001	1000	3	30	Parameter Read Data
4	Write	Fixed Interval	100	MODBUS/TCP(192.168.0.1)	Host Station(192.168.0.2)	D	1105	1140	Holding Register	10000	10035	D	1000	1001	1000	3	30	Parameter Write Data
5	Read	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	Holding Register	11024	11029	D	1045	1050	D	1000	1001	1000	3	30	Drag Data
6																		
7																		
8																		

Explanation

Select the setting No. to execute Simple CPU Communication.
Please refer to an operation manual for the setting of communication destination.

Check Restore the Default Settings

Apply

1. Refer [Chapter 3.1.3](#) & follow the same procedure as in Step 1 and configure for Outputs, Parameter Mailbox and Diagnostics
2. Click on **Apply** to complete the Configuration

**Note**

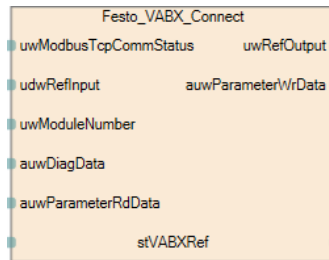
- The Parameter Mailbox uses a bit length of 36 registers (10000 to 10035).
- The configuration shown applies to a single VABX valve; the same setup must be repeated for each valve.
- The Input and Output addresses will vary depend in user configuration, for this Application Note Module 4 is considered

4 Festo_VABX_MB Library Function Blocks Description

This chapter provides the details of **Festo_VABX_MB** Library Function blocks

4.1 Connect block

The Connect block manages the module number, diagnostic information, Communication Status, and CPX parameters, shares these data with other blocks, and contains the process data mapping.



4.1.1 Festo_VABX_Connect Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1

Table 4.1: Festo_VABX_Connect Block In-Out Variables description

4.1.2 Festo_VABX_Connect Block Input Variables

Input Variable	Data Type	Description
uwModbusTcpCommSta- tus	Word [Unsigned]/Bit String [16-bit]	Modbus Communication Status
udwRefInput	Doubleword [Unsigned]/Bit String [32-bit]	VABX Process input Data
uwModuleNumber	Word [Unsigned]/Bit String [16-bit]	Module Number (Refer Chapter 3.1.5)
auwDiagData	Word [Unsigned]/Bit String [16-bit](0..5)	Diagnosis Data
auwParameterRdData	Word [Unsigned]/Bit String [16-bit](0..35)	Parameter Mailbox Read Address area

Table 4.2: Festo_VABX_Connect Block Input Variables descriptions

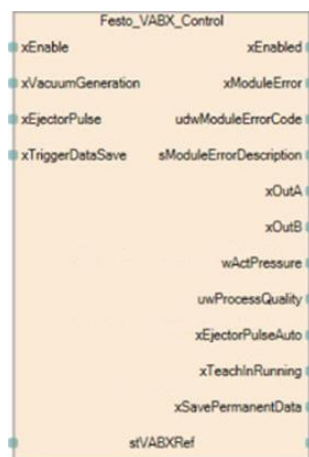
4.1.3 Festo_VABX_Connect Block Output Variables

Output Variable	Data Type	Description
uwRefOutput	Word [Unsigned]/Bit String [16-bit](0..1)	VABX Process Output Data
auwParameterWrData	Word [Unsigned]/Bit String [16-bit](0..35)	Parameter Mailbox Write Address area

Table 4.3: Festo_VABX_Connect Block Output Variables descriptions

4.2 Control block

The control block manages vacuum module operations by processing input commands for enabling, vacuum generation, ejector pulses, and data saving, while providing outputs for status, error reporting, pressure, process quality, and operational signals.



4.2.1 Festo_VABX_Control Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1

Table 4.4: Festo_VABX_Control Block In-Out Variables description

4.2.2 Festo_VABX_Control Block Input Variables

Input Variable	Data Type	Description
xEnable	Bit	Enable VTUX Vacuum control block TRUE = Enable the block FALSE = Disable the block
xVacuumGeneration	Bit	Vacuum Generation TRUE = Vacuum Generation on FALSE = Vacuum Generation off
xEjectorPulse	Bit	Ejector Pulse TRUE = Ejector Pulse on FALSE = Ejector Pulse off
xTriggerDataSave	Bit	TRUE = Save the Data

Table 4.5: Festo_VABX_Control Block Input Variables descriptions

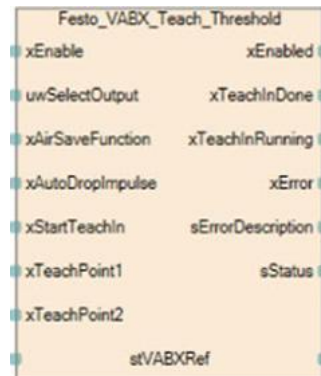
4.2.3 Festo_VABX_Control Block Output Variables

Output Variable	Data Type	Description
xEnabled	Bit	VTUX Vacuum control block Enabled feedback
xModuleError	Bit	Error in the VTUX Vacuum module
udwModuleErrorCode	Double Word [Unsigned]/Bit String [32-bit]	VTUX Vacuum module error code
sModuleErrorDescription	String(32)	Module Error Description
xOutA	Bit	TRUE = Output channel A ON
xOutB	Bit	TRUE = Output channel B ON
wActPressure	Word [Signed]	Actual pressure in the vacuum duct
uwProcessQuality	Word [Unsigned]/Bit String [16-bit]	Process quality feedback
xEjectorPulseAuto	Bit	TRUE = Automatic Ejector pulse enabled
xTeachInRunning	Bit	TRUE = Teaching in progress
xSavePermanentData	Bit	TRUE = Data saved successfully

Table 4.6: Festo_VABX_Control Block Output Variables descriptions

4.3 Teach Threshold Comparator block

This **Festo_VABX_Teach_Threshold** block facilitates the threshold comparator teach-in by handling control inputs along with Air Save function and signaling the teach-in progress, completion, status, and any errors encountered.



4.3.1 Festo_VABX_Teach_Threshold Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1

Table 4.7: Festo_VABX_Teach_Threshold Block In-Out Variables description

4.3.2 Festo_VABX_Teach_Threshold Block Input Variables

Input Variable	Data Type	Description
xEnable	Bit	Enable VTUX Vacuum Teach Threshold comparator block TRUE = Enable the block FALSE = Disable the block
uwSelectOutput	Word [Unsigned]/Bit String [16-bit]	Select the Output 0-OutA 1-OutB Input is valid before Start teaching.
xAirSaveFunction	Bit	TRUE = Enable Air save function. Input is valid before Start teaching
xAutoDropImpulse	Bit	TRUE = Enable Auto drop impulse
xStartTeachIn	Bit	TRUE = Start the Threshold comparator Teach-In
xTeachPoint1	Bit	TRUE = Start the Teach point 1
xTeachPoint2	Bit	TRUE = Start the Teach point 2

Table 4.8: Festo_VABX_Teach_Threshold Block Input Variables descriptions

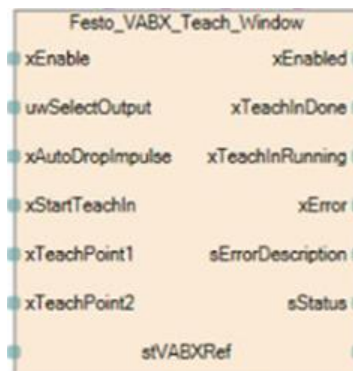
4.3.3 Festo_VABX_Teach_Threshold Block Output Variables

Output Variable	Data Type	Description
xEnabled	Bit	VTUX Vacuum Teach Threshold comparator block Enabled feedback
xTeachInDone	Bit	Teach-In done successfully
xTeachInRunning	Bit	Teach-In started and in-progress
xError	Bit	Error during Teach-In
sErrorDescription	String(32)	Teach-In error description
sStatus	String(32)	Teach-In Status

Table 4.9: Festo_VABX_Teach_Threshold Block Output Variables descriptions

4.4 Teach Window Comparator block

This **Festo_VABX_Teach_Window** block facilitates the Window comparator teach-in by handling control inputs and signaling the teach-in progress, completion, status, and any errors encountered.



4.4.1 Festo_VABX_Teach_Window Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1

Table 4.10: Festo_VABX_Teach_Window Block In-Out Variables description

4.4.2 Festo_VABX_Teach_Window Block Input Variables

Input Variable	Data Type	Description
xEnable	Bit	Enable VTUX Vacuum Teach Window comparator block TRUE = Enable the block FALSE = Disable the block
uwSelectOutput	Word [Unsigned]/Bit String [16-bit]	Select the Output 0-OutA 1-OutB Input is valid before Start teaching
xAutoDropImpulse	Bit	TRUE = Enable Auto drop impulse
xStartTeachIn	Bit	TRUE = Start the Window comparator Teach-In
xTeachPoint1	Bit	TRUE = Start the Teach point 1
xTeachPoint2	Bit	TRUE = Start the Teach point 2

Table 4.11: Festo_VABX_Teach_Window Block Input Variables descriptions

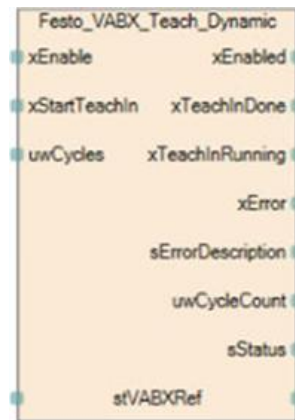
4.4.3 Festo_VABX_Teach_Window Block Output Variables

Output Variable	Data Type	Description
xEnabled	Bit	VTUX Vacuum Teach Window comparator block Enabled feedback
xTeachInDone	Bit	Teach-In done successfully
xTeachInRunning	Bit	Teach-In started and in-progress
xError	Bit	Error during Teach-In
sErrorDescription	String(32)	Teach-In error description
sStatus	String(32)	Teach-In Status

Table 4.12: Festo_VABX_Teach_Window Block Output Variables descriptions

4.5 Teach Dynamic block

This **Festo_VABX_Teach_Dynamic** block facilitates the Dynamic teach-in by handling control inputs and signaling the teach-in progress, completion, status, and any errors encountered.



4.5.1 Festo_VABX_Teach_Dynamic Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1

Table 4.13: Festo_VABX_Teach_Dynamic Block In-Out Variables description

4.5.2 Festo_VABX_Teach_Dynamic Block Input Variables

Input Variable	Data Type	Description
xEnable	Bit	Enable VTUX Vacuum Teach Dynamic block TRUE = Enable the block FALSE = Disable the block
xStartTeachIn	Bit	TRUE = Start the Dynamic Teach-In
uwCycles	Word [Unsigned]/Bit String [16-bit]	No. Of Cycles to be Teach

Table 4.14: Festo_VABX_Teach_Dynamic Block Input Variables descriptions

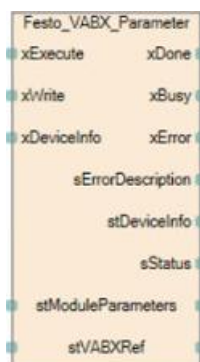
4.5.3 Festo_VABX_Teach_Dynamic Block Output Variables

Output Variable	Data Type	Description
xEnabled	Bit	VTUX Vacuum Teach Dynamic Block Enabled feedback
xTeachInDone	Bit	Teach-In done successfully
xTeachInRunning	Bit	Teach-In started and in-progress
xError	Bit	Error during Teach-In
sErrorDescription	String(32)	Teach-In error description
uwCycleCount	Word [Unsigned]/Bit String [16-bit]	Completed cycle count
sStatus	String(32)	Teach-In Status

Table 4.15: Festo_VABX_Teach_Dynamic Block Output Variables descriptions

4.6 Parameter Block

This **Festo_VABX_Parameter** block reads and writes module parameters and allows reading of device information.



4.6.1 Festo_VABX_Parameter Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1
stModuleParameters	Struct of the type Module_Parameters	Structure of module parameters Refer Chapter 4.8.2

Table 4.16: Festo_VABX_Parameter Block In-Out Variables descriptions

4.6.2 Festo_VABX_Parameter Block Input Variables

Input Variable	Data Type	Description
xExecute	Bit	TRUE = Start read or write VTUX Vacuum parameters
xWrite	Bit	TRUE = Parameter Write FALSE = Parameter Read
xDeviceInfo	Bit	TRUE = Read device information parameters

Table 4.17: Festo_VABX_Parameter Block Input Variables descriptions

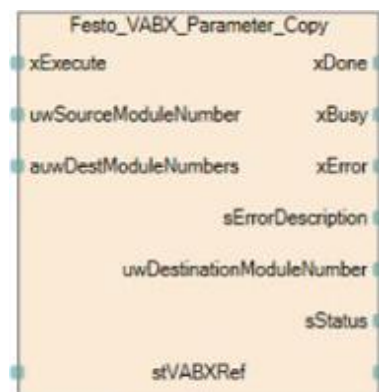
4.6.3 Festo_VABX_Parameter Block Output Variables

Output Variable	Data Type	Description
xDone	Bit	Parameter read or write is completed successfully
xBusy	Bit	Parameter read or write is under progress
xError	Bit	Error during parameter read or write
sErrorDescription	String(32)	Parameter error description
stDeviceInfo	Struct of the type Device_Info	Structure of device information parameters Refer Chapter 4.8.3
sStatus	String(32)	Parameter read or write status

Table 4.18: Festo_VABX_Parameter Block Output Variables descriptions

4.7 Parameter Copy Block

This **Festo_VABX_Parameter_Copy** block enables copying module parameters from a source module to multiple target modules.



4.7.1 Festo_VABX_Parameter Block In-Out Variables

In-Out Variable	Data Type	Description
stVABXRef	Struct of the type VABX_Ref	Data Structure to link the data with other blocks Refer Chapter 4.8.1

Table 4.19: Festo_VABX_Parameter Block In-Out Variables description

4.7.2 Festo_VABX_Parameter_Copy Block Input Variables

Input Variable	Data Type	Description
xExecute	Bit	TRUE = Start the parameter copy operation
uwSourceModuleNumber	Word [Unsigned]/Bit String [16-bit]	Set the source module number to copy the parameters from that module
auwDestModuleNumbers	Word [Unsigned]/Bit String [16-bit](0..50)	Set the destination module numbers to copy the parameters from the source module to these modules

Table 4.20: Festo_VABX_Parameter_Copy Block Input Variables descriptions

4.7.3 Festo_VABX_Parameter_Copy Block Output Variables

Output Variable	Data Type	Description
xDone	Bit	Parameter copy is completed successfully
xBusy	Bit	Parameter copy is under progress
xError	Bit	Error during parameter copy
sErrorDescription	String(32)	Parameter copy operation error description
uwDestinationModuleNumber	Word [Unsigned]/Bit String [16-bit]	Current destination module number to which parameters are being copied
sStatus	String(32)	Parameter copy status

Table 4.21: Festo_VABX_Parameter_Copy Block Output Variables descriptions

4.8 Structure

4.8.1 Structure (VABX_Ref)

Structure Variable	Data Type
Module_Number	Word [Unsigned]/Bit String [16-bit]
DataIn	Process_Data_Input Chapter 4.8.4
DataOut	Process_Data_Output Chapter 4.8.5
Module_Check_OK	Bit
Communication_Time_Out	Bit
Control_Block_Enabled	Bit
Threshold_Comparator_TeachIn_is_in_Process	Bit
Window_Comparator_TeachIn_is_in_Process	Bit
Dynamic_TeachIn_is_in_Process	Bit
Connection_OK	Bit
Connection_ErrorCode	Word [Unsigned]/Bit String [16-bit]
ComParErrorDescription	String(32)
ComParErrorCode	Word [Unsigned]/Bit String [16-bit]
DiagData	Double Word [Unsigned]/Bit String [32-bit]
ParRdData	Word [Unsigned]/Bit String [16-bit](0..35)
ParWrData	Word [Unsigned]/Bit String [16-bit](0..35)
Cntrl_CPX_ModuleNum	Word [Unsigned]/Bit String [16-bit]
Cntrl_CPX_ParID	Word [Unsigned]/Bit String [16-bit]
Cntrl_CPX_Instance	Word [Unsigned]/Bit String [16-bit]
Cntrl_CPX_Write	Bit
Cntrl_CPX_Exe	Bit
TC_CPX_ModuleNum	Word [Unsigned]/Bit String [16-bit]
TC_CPX_ParID	Word [Unsigned]/Bit String [16-bit]
TC_CPX_Instance	Word [Unsigned]/Bit String [16-bit]
TC_CPX_Write	Bit
TC_ParWrData	Word [Unsigned]/Bit String [16-bit]
TC_CPX_Exe	Bit
WC_CPX_ModuleNum	Word [Unsigned]/Bit String [16-bit]

Structure Variable	Data Type
WC_CPX_ParID	Word [Unsigned]/Bit String [16-bit]
WC_CPX_Instance	Word [Unsigned]/Bit String [16-bit]
WC_CPX_Write	Bit
WC_ParWrData	Word [Unsigned]/Bit String [16-bit]
WC_CPX_Exe	Bit
DY_CPX_ModuleNum	Word [Unsigned]/Bit String [16-bit]
DY_CPX_ParID	Word [Unsigned]/Bit String [16-bit]
DY_CPX_Instance	Word [Unsigned]/Bit String [16-bit]
DY_ParWrData	Word [Unsigned]/Bit String [16-bit]
DY_CPX_Exe	Bit
Pr_CPX_ModuleNum	Word [Unsigned]/Bit String [16-bit]
Pr_CPX_ParID	Word [Unsigned]/Bit String [16-bit]
Pr_CPX_Instance	Word [Unsigned]/Bit String [16-bit]
Pr_CPX_Write	Bit
Pr_ParWrData	Word [Unsigned]/Bit String [16-bit](0..35)
Pr_ParWrDataLen	Word [Unsigned]/Bit String [16-bit]
Pr_CPX_Exe	Bit
PrCy_CPX_ModuleNum	Word [Unsigned]/Bit String [16-bit]
PrCy_CPX_ParID	Word [Unsigned]/Bit String [16-bit]
PrCy_CPX_Instance	Word [Unsigned]/Bit String [16-bit]
PrCy_CPX_Write	Bit
PrCy_ParWrData	Word [Unsigned]/Bit String [16-bit](0..35)
PrCy_ParWrDataLen	Word [Unsigned]/Bit String [16-bit]
PrCy_CPX_Exe	Bit
CPX_Done	Bit
CPX_Busy	Bit
CPX_Error	Bit
CPX_ErrorCode	Word [Unsigned]/Bit String [16-bit]

Table 4.22: Structure (VABX_Ref) data types

4.8.2 Structure (Module_Parameters)

Structure Variable	Data Type
ProcessDiagnosis	Bit
InterlockEjectorPulse	Bit
ThersholdProcessQuality	Word [Unsigned]/Bit String [16-bit]
LimitEvacuationTime	Word [Unsigned]/Bit String [16-bit]
LimitVentingTime	Word [Unsigned]/Bit String [16-bit]
AutoDropTime	Word [Unsigned]/Bit String [16-bit]
AutomaticDropImpulse	Bit
AirSaveFunction	Bit
SwitchingPointA1	Word [Unsigned]/Bit String [16-bit]
HysteresisA	Word [Unsigned]/Bit String [16-bit]
SwitchingPointB1	Word [Unsigned]/Bit String [16-bit]
HysteresisB	Word [Unsigned]/Bit String [16-bit]
SwitchingPointA2	Word [Unsigned]/Bit String [16-bit]
SwitchingPointB2	Word [Unsigned]/Bit String [16-bit]
PressureUnit	Word [Unsigned]/Bit String [16-bit]
LockCode	Word [Unsigned]/Bit String [16-bit]
SwitchPointLogic	Bit
SwitchPointMode	Bit

Table 4.23: Structure (Module_Parameters) data types

4.8.3 Structure (Device_Info)

Structure Variable	Data Type
FieldbusSerialNumber	Double Word [Unsigned]/Bit String [32-bit]
ProductKey	String(32)
FirmwareVersion	String(32)
ModuleCode	Double Word [Unsigned]/Bit String [32-bit]
RevisionValve	Double Word [Unsigned]/Bit String [32-bit]
ValveFunction	String(32)

Table 4.24: Structure (Device_Info) data types

4.8.4 Structure (Process_Data_Input)

Structure Variable	Data Type
Actual_Pressure	Word [Signed]
OUT_A	Bit
OUT_B	Bit
Ejector_Pulse	Bit
TeachIn_Running	Bit
Save_Data	Bit
Process_Quality	Word [Unsigned]/Bit String [16-bit]

Table 4.25: Structure (Process_Data_Input) data types

4.8.5 Structure (Process_Data_Output)

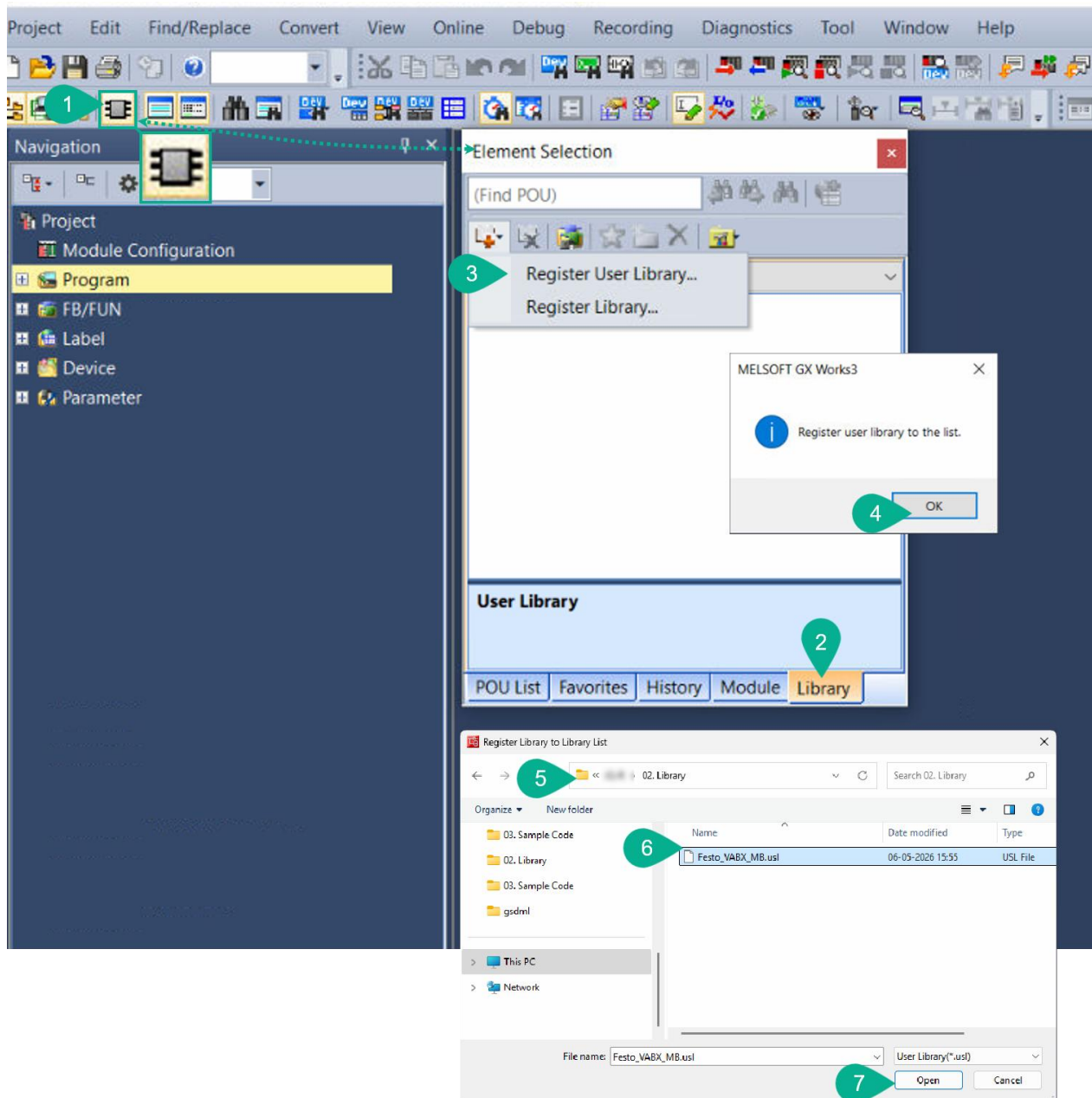
Structure Variable	Data Type
Vacuum_Generation	Bit
Ejector_Pulse	Bit
ReservedB2	Bit
ReservedB3	Bit
ReservedB4	Bit
ReservedB5	Bit
ReservedB6	Bit
Save_Data	Bit
Start_TeachIn	Bit
Select_Channel	Bit
Teach_Mode	Bit
ReservedB11	Bit
Select_Output	Bit
Teach_Point1	Bit
Teach_Point2	Bit
Stop_TeachIn	Bit

Table 4.26: Structure (Process_Data_Output) data types

5 Configuration of Festo_VABX_MB Library Function Blocks in GX Works3

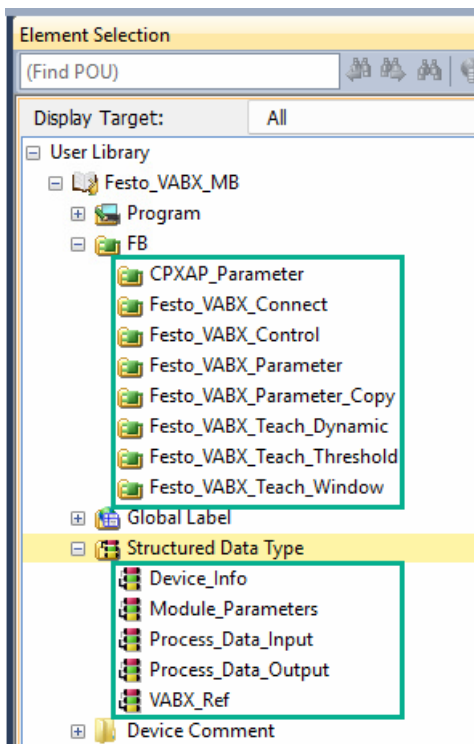
5.1 Importing Festo_VABX_MB Library

Step 1



1. Click on **Element Selection**
2. Select **Library** tab
3. Click on **Register User Library** under **Register Library List**
4. Click **OK** of **Register User Library to the list** window
5. Locate the **Festo_VABX_MB.usl** file
6. Select **Festo_VABX_MB.usl** file
7. Click **Open**

Once Library imported it looks as below



VABX Blocks :

- Festo_VABX_Connect
- Festo_VABX_Control
- Festo_VABX_Teach_Threshold
- Festo_VABX_Teach_Window
- Festo_VABX_Teach_Dynamic
- Festo_VABX_Parameter
- Festo_VABX_Parameter_Copy

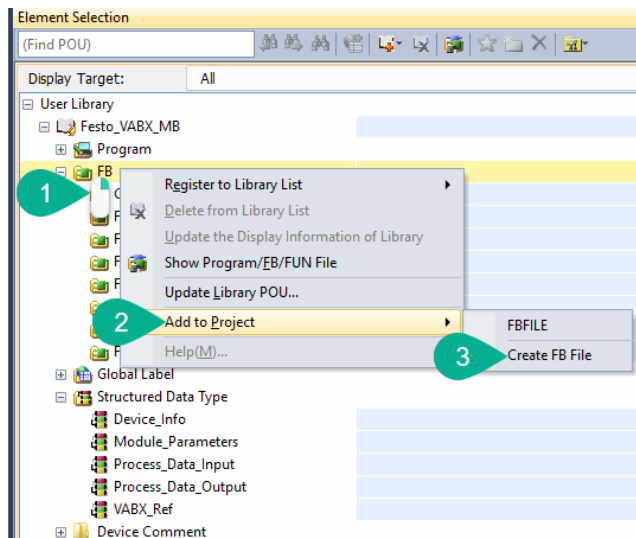
Supporting Blocks :

- CPXAP_Parameter

Structured Data Types :

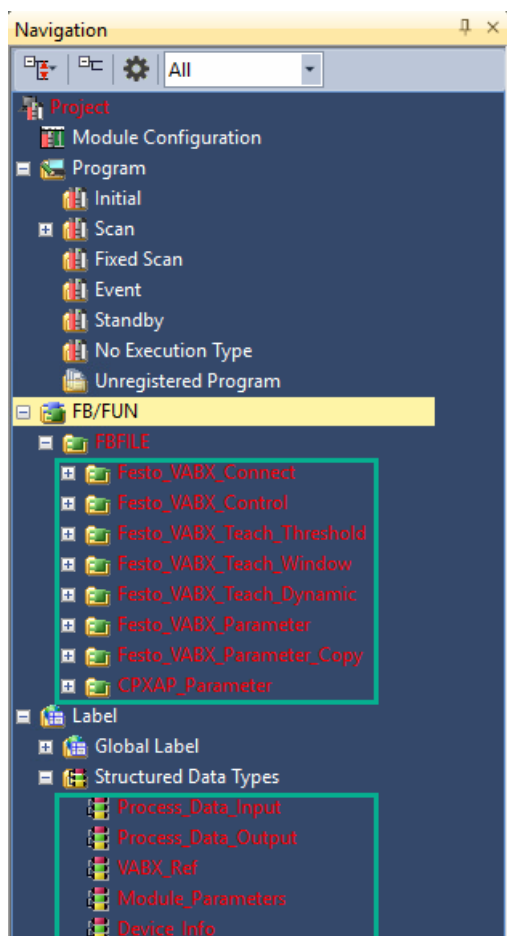
- VABX_Ref
- Device_Info
- Module_Parameters
- Process_Data_Input
- Process_Data_Output

Step 2



1. **Right Click on FB**
2. Click **Add to Project**
3. Click on **Create FB File**

Once Program added it looks as below.



5.2 Global Label Declaration

Setting No.	Communication Pattern	Communication Setting: Execution Interval(ms)	Communication Destination (IP Address)		Word Device						Comment	
			Source	Destination	Type	Start	End	Type	Start	End		
1	Read	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	Holding Register	5002	5003	D	1000	1001	PDI (VABX to PLC)
2	Write	Fixed Interval	100	MODBUS/TCP(192.168.0.1)	Host Station(192.168.0.2)	D	1100	1100	Holding Register	1	1	PDO (PLC to VABX)
3	Read	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	Holding Register	10000	10035	D	1005	1040	Parameter Read Data
4	Write	Fixed Interval	100	MODBUS/TCP(192.168.0.1)	Host Station(192.168.0.2)	D	1105	1140	Holding Register	10000	10035	Parameter Write Data
5	Read	Fixed Interval	100	MODBUS/TCP(192.168.0.2)	Host Station(192.168.0.1)	Holding Register	11024	11029	D	1045	1050	Diag Data

- Under **Global Label**, create a required Global **Data Name** (VABX_GlobalLabels is for this Application Note) and double click on it to open
- Enter required **Label Name**; create a label (Refer [Chapter 4.1](#)) for **Festo_VABX_Connect** block for udwRefInput, auwDiagData, auwParameterRdData, uwRefOutput and auwParameterWrData
- Select the respective **Data Type** (Refer [Chapter 4.1](#))
- Select **VAR_GLOBAL_RETAIN** in **Class**
- Referring Simple CPU communication setting (Refer [Chapter 3.2.3](#) Step 3) assign the respective Devices in **Assign (Device/Label)**

Note

- Make Sure the Assign Devices are defined in Latch Area
Parameter >> CPU Parameter >> Memory /Device Setting >> Device /Label Memory Area Setting >> Device Setting >> Data Register >> Latch >> Select Device >> Enter Start & End address range >> Apply

Setting Item List

Item	Symbol	Points	Device	Local Device	Latch (1)	Latch (2)
Link Relay	B	8K	0 to 1FFF		No Setting	No Setting
Link Special Relay	SB	2K	0 to 7FF		No Setting	No Setting
Annunciator	F	2K	0 to 2047		No Setting	No Setting
Edge Relay	V	2K	0 to 2047		No Setting	No Setting
Step Relay	S	0			No Setting	No Setting
Timer	T	1K	0 to 1023		No Setting	No Setting
Long Timer	LT	1K	0 to 1023		No Setting	No Setting
Retentive Timer	ST	0			No Setting	No Setting
Long Retentive Timer	LST	0			No Setting	No Setting
Counter	C	512	0 to 511		No Setting	No Setting
Long Counter	LC	512	0 to 511		No Setting	No Setting
Data Register	D	16K	0 to 18431	1000 to 2000	No Setting	No Setting
Link Register	W	8K	0 to 1FFF		No Setting	No Setting
Link Special Register	SV	2K	0 to 7FF		No Setting	No Setting
Latch Relay	L	8K	0 to 8191		No Setting	No Setting
File Register	ZR				No Setting	No Setting

Explanation
 Set the device points, local device range and latch range used in data register (D).
 To disable the device write operation from external device, set the range of write protection. (Setting unit is 1 point unit.)
 And set the settable points and range of local device and of write protection within the range which set in device points.
 The local device area capacity is decided by the setting points of local device and the number of programs which use local device.

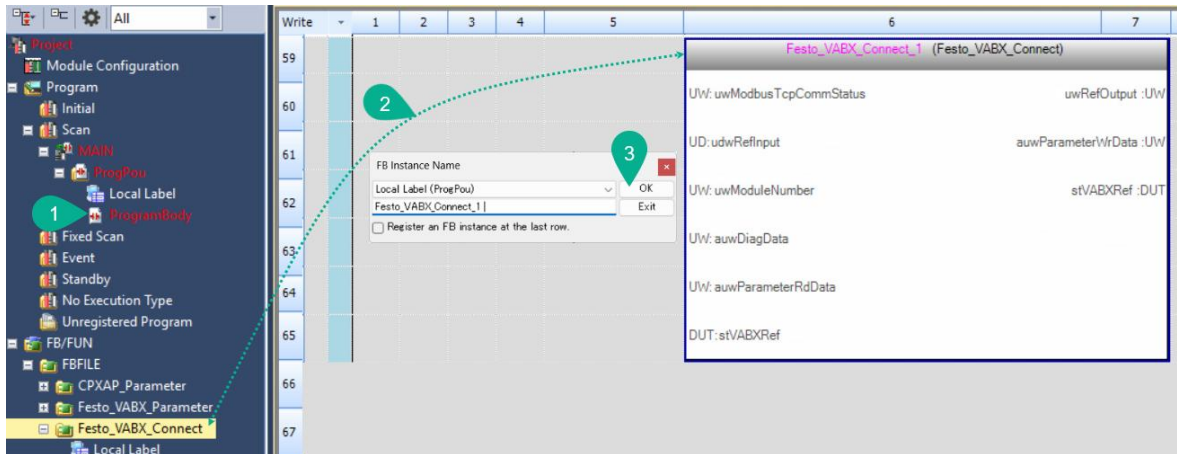
[Type] Word Device, [Notation] DEC
 [Setting range]
 <RnCPU, RnENCPU>
 Device Points: 0 to 10117632 [Points] (4 points unit)
 <RnCPU>

Check Restore the Default Settings

5.3 Configuring Festo_VABX_MB Library function Blocks in GX Works3

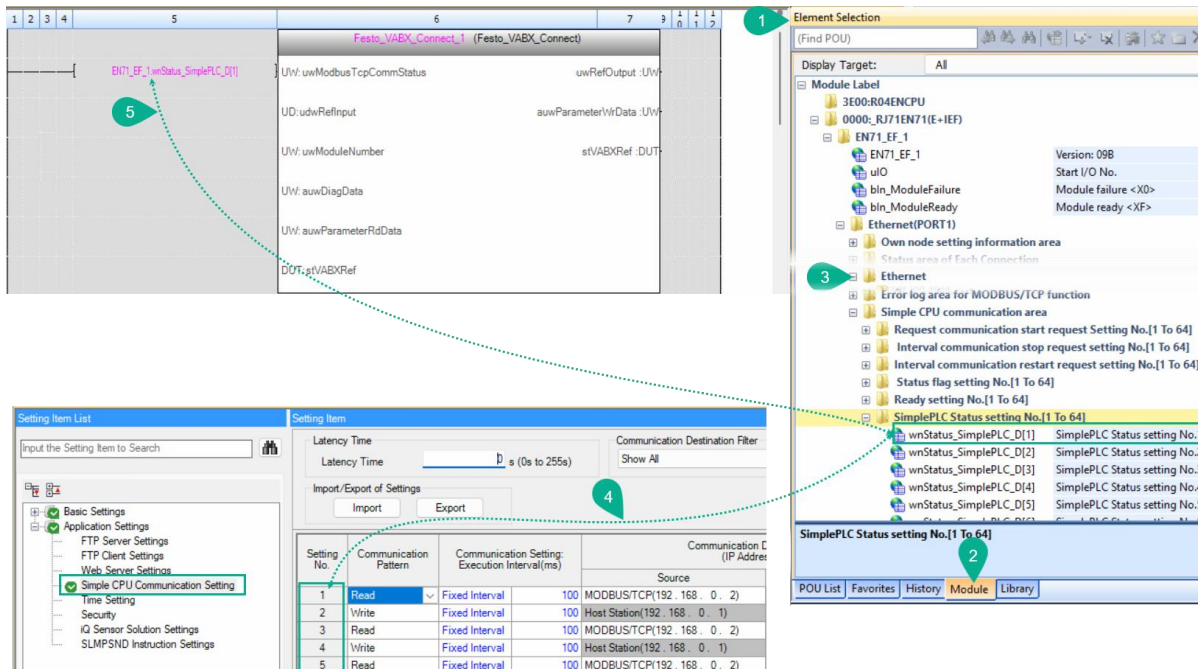
5.3.1 Configuring VABX Blocks

Step 1



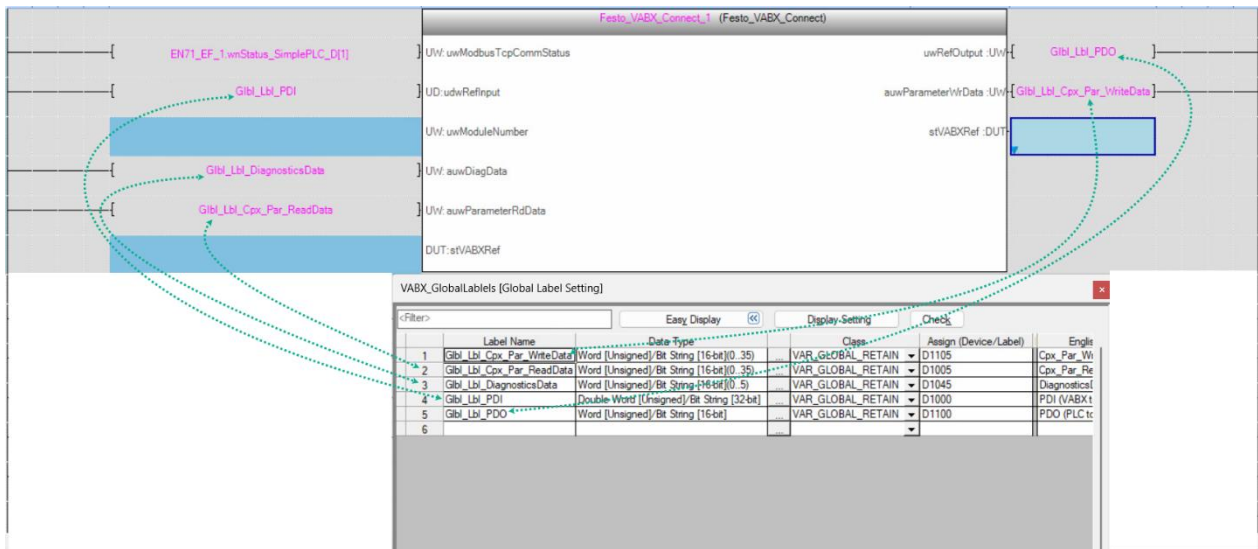
1. Double Click on **ProgramBody**
2. Drag & drop **Festo_VABX_Connect** from FBFILE of FB/FUN to Program Window
3. Click on **OK**

Step 2



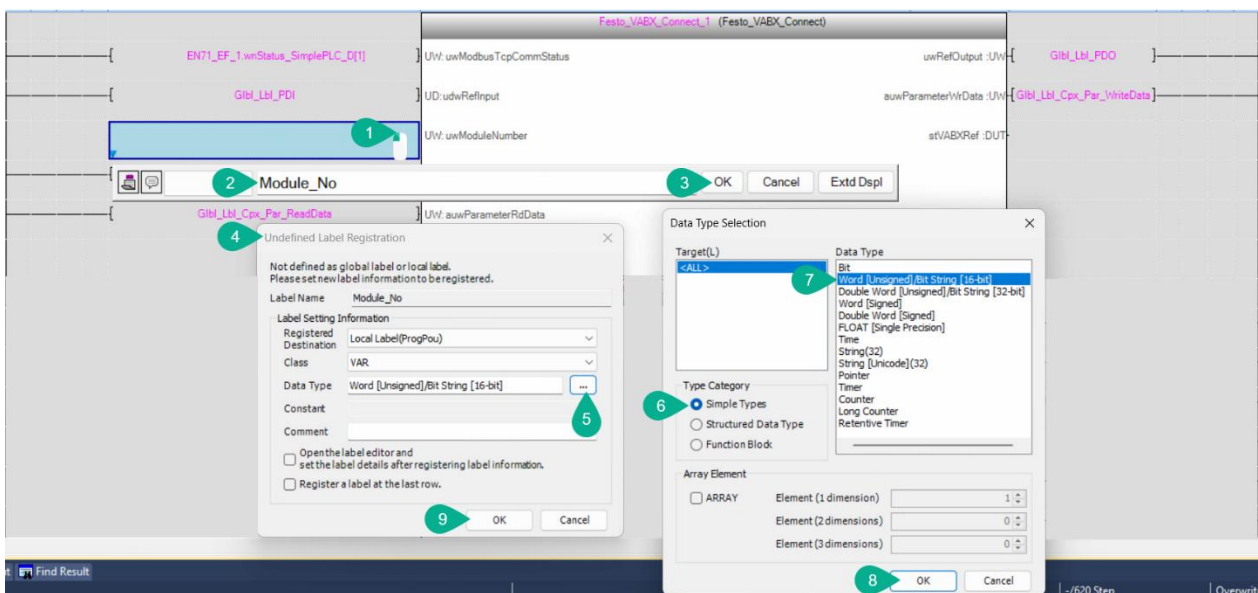
1. Open **Element Selection Window**
2. Select **Module** tab
3. Expand **Ethernet** under **RJ71EN71(E+IEF)** in **Module Label**
4. Refer **Setting No.** in **Simple CPU Communication configuration** and Select communication status word (**wnStatus_SimplePLC_D[x]**) under **Setting No.**
5. Assign the communication status word by drag & dropping (**wnStatus_SimplePLC_D[x]**) to input **uwModbusTcpCommStatus**

Step 3



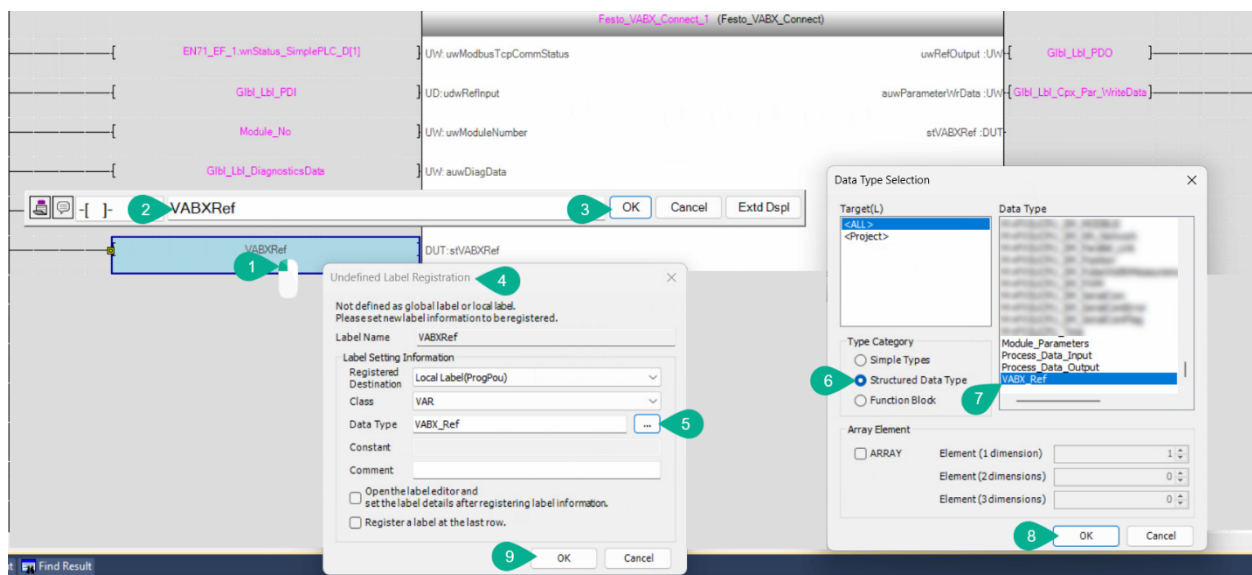
Assign the respective Global label created ([Chapter 5.2](#)) by drag and dropping the respective Global Label to the respective Inputs and Outputs as shown in above image.

Step 4



1. Double Click on next to Input / Output terminals
2. Enter required Variable Name
3. Click **OK**
4. **Undefined Label Registration** window opens
5. Click on Data Type button
6. Select **Simple Types** in **Type Category**
7. Select respective **Data Type** (Refer [Chapter 4](#))
8. Click **OK** button of **Data Type Selection** window
9. Click **OK** button of **Undefined Label Registration** window

Step 5

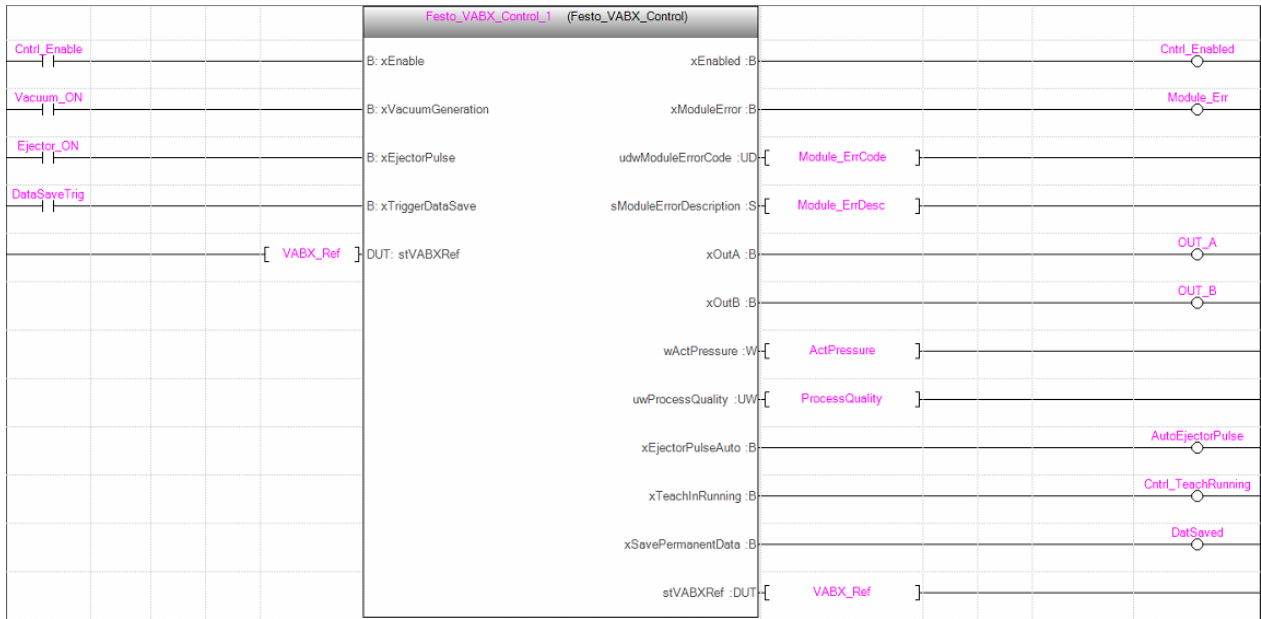


1. Double Click on next to Input / Output terminals
2. Enter required Variable Name
3. Click **OK**
4. **Undefined Label Registration** window opens
5. Click on Data Type button 
6. Select **Structured Data Type** in **Type Category**
7. Select respective **Data Type** (Refer [Chapter 4](#))
8. Click **OK** button of **Data Type Selection** window
9. Click **OK** button of **Undefined Label Registration** window

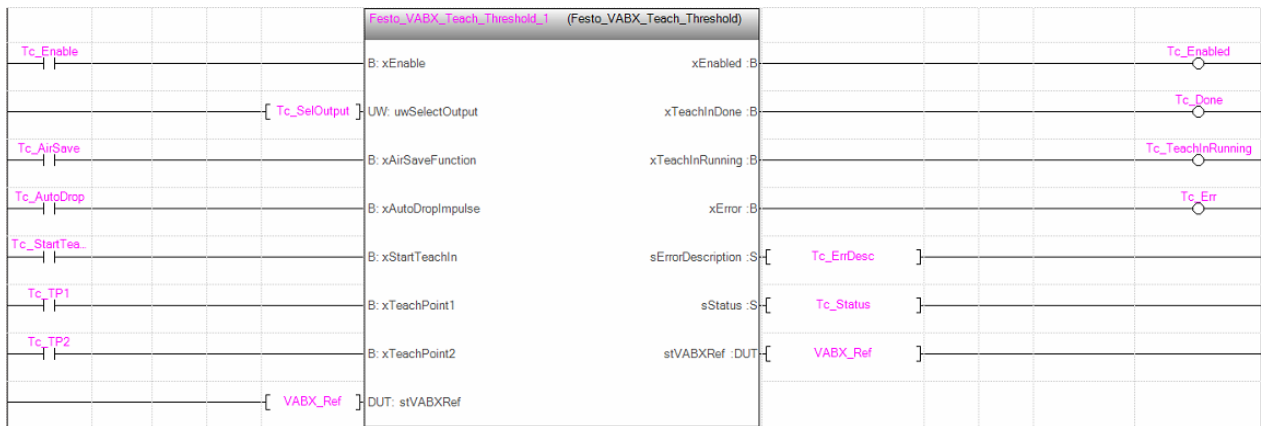
Follow the same procedure for remaining Inputs and Outputs. Once all variables are assigned it looks as below.

Festo_VABX_Connect_1 (Festo_VABX_Connect)			
EN71_EF_1.wnStatus_SimplePLC_D[1]	UW: uwModbusTcpCommStatus	uwRefOutput :UW	Gibl_Lbl_PDO
Gibl_Lbl_PDI	UD: udwRefInput	auwParameterWrData :UW	Gibl_Lbl_Cpx_Par_Write
ModuleNo	UW: uwModuleNumber	stVABXRef :DUT	VABX_Ref
Gibl_Lbl_DiagnosticsData	UW: auwDiagData		
Gibl_Lbl_Cpx_Par_ReadData	UW: auwParameterRdData		
VABX_Ref	DUT: stVABXRef		

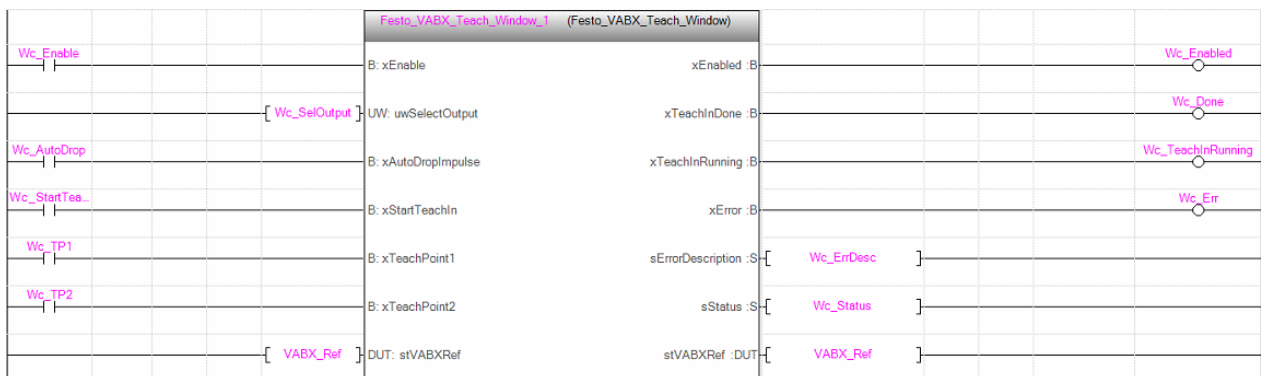
Follow the same steps for other function blocks. Once the variable assignment is completed for **Festo_VABX_Control** block, it looks as below.



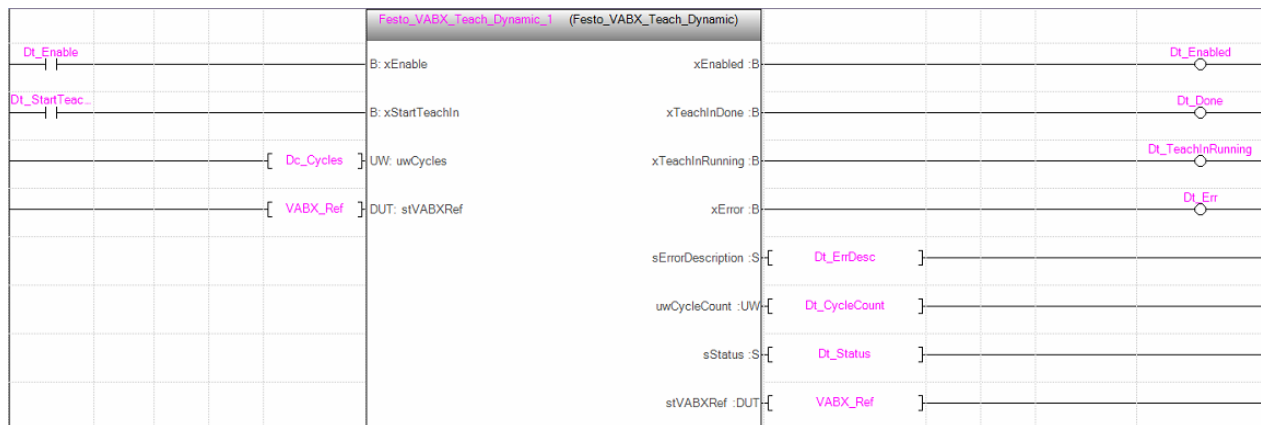
Once the variable assignment is completed for **Festo_VABX_Teach_Threshold** block, it looks as below.



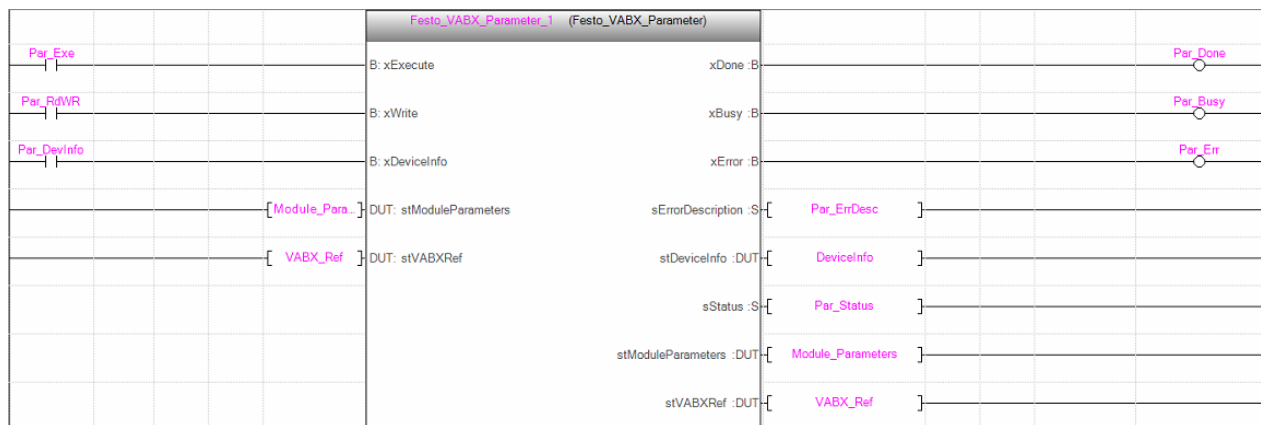
Once the variable assignment is completed for **Festo_VABX_Teach_Window** block, it looks as below.



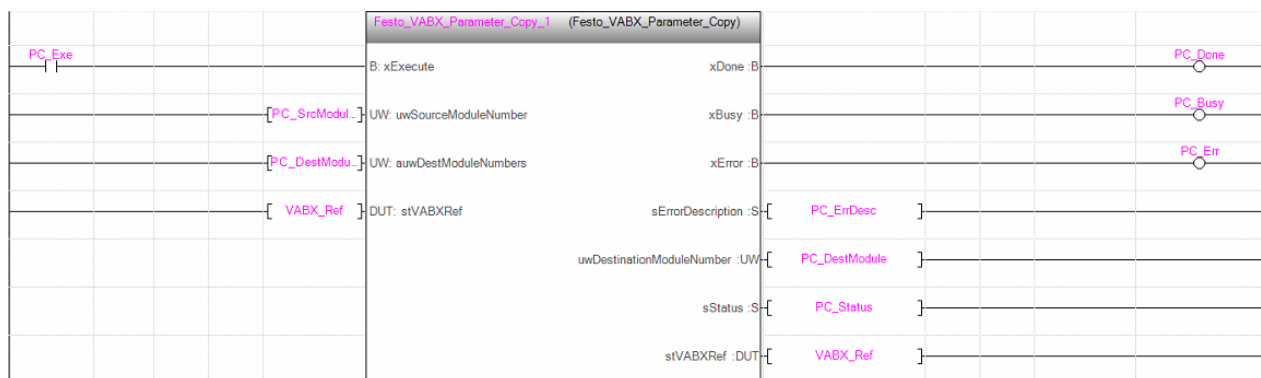
Once the variable assignment is completed for **Festo_VABX_Teach_Dynamic** block, it looks as below.



Once the variable assignment is completed for **Festo_VABX_Parameter** block, it looks as below.



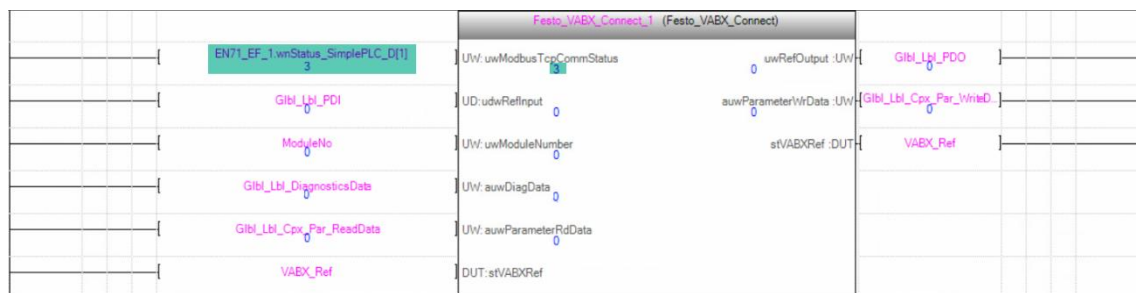
Once the variable assignment is completed for **Festo_VABX_Parameter_Copy** block, it looks as below.



6 Sample Code

This chapter explains the operation of the Control block and demonstrates teaching samples using the Threshold Comparator method, Window Comparator method, and Dynamic method. It also includes examples of reading and writing module parameters, as well as copying parameters from a Source Module to Destination Modules.

6.1 Initial Communication



The **uwModbusTcpCommStatus** must be **3** for Healthy Communication.



Note

- Simple CPU Communication Status- this image is captured from Mitsubishi Simple CPU Communication Manual

Checking the simple CPU communication status

The simple CPU communication status can be checked with the buffer memory or diagnostic functions.

Checking with the buffer memory

The simple CPU communication status can be checked with the storage status of the corresponding setting number in the following buffer memory areas.

Item	Address	Remarks
Request to start communication at request	UnIG721896 to UnIG721899 UnIG1247300 to UnIG1247327	- Setting No.1: UnIG721896.0 ... - Setting No.64: UnIG721899.F - Setting No.65: UnIG1247300.0 ... - Setting No.512: UnIG1247327.F
Ready	UnIG721912 to UnIG721915 UnIG1247412 to UnIG1247439	- Setting No.1: UnIG721912.0 ... - Setting No.64: UnIG721915.F - Setting No.65: UnIG1247412.0 ... - Setting No.512: UnIG1247439.F
Simple CPU communication status	0H: Unset	- Setting No.1: UnIG721936 ... - Setting No.64: UnIG721999 - Setting No.65: UnIG1247460 ... - Setting No.512: UnIG1247907
	1H: Preparing	
	2H: Waiting for the request	
	3H: Communicating	
	4H: Communication stop	
	5H: Retry being executed	
	6H: Monitoring at error	
	AH: Communications impossible	
Simple CPU communication error code	UnIG722000 to UnIG722063 UnIG1247908 to UnIG1248355	- Setting No.1: UnIG722000 ... - Setting No.64: UnIG722063 - Setting No.65: UnIG1247908 ... - Setting No.512: UnIG1248355

6.2 Connect Block



User must enter the Module Number in **uwModuleNumber**



Note

- Ensure the correct module number is entered; otherwise, parameter and process data can become mismatched, potentially causing incorrect operation.

6.3 Control Block

6.3.1 Vacuum Generation

Before Vacuum ON



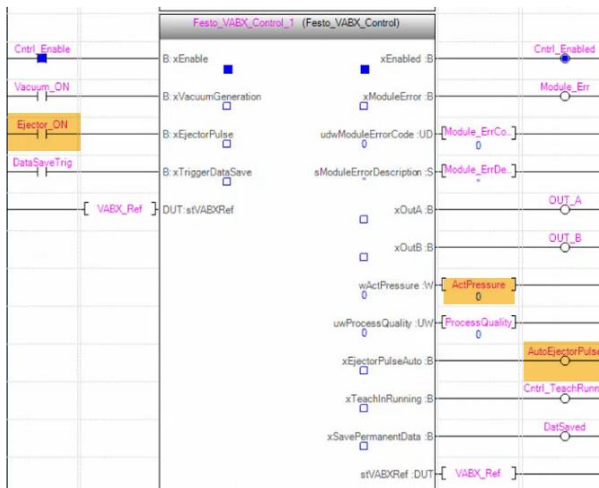
After Vacuum ON



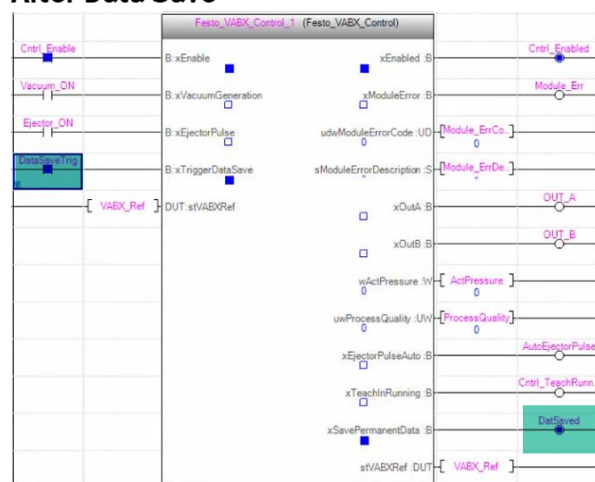
- Turn ON **xEnable** Input
- Output **xEnabled** Turns TRUE
- Turn ON **xVacuumGeneration** Input
- Monitor the Output **wActPressure** to see the changes in Actual pressure

**Note**

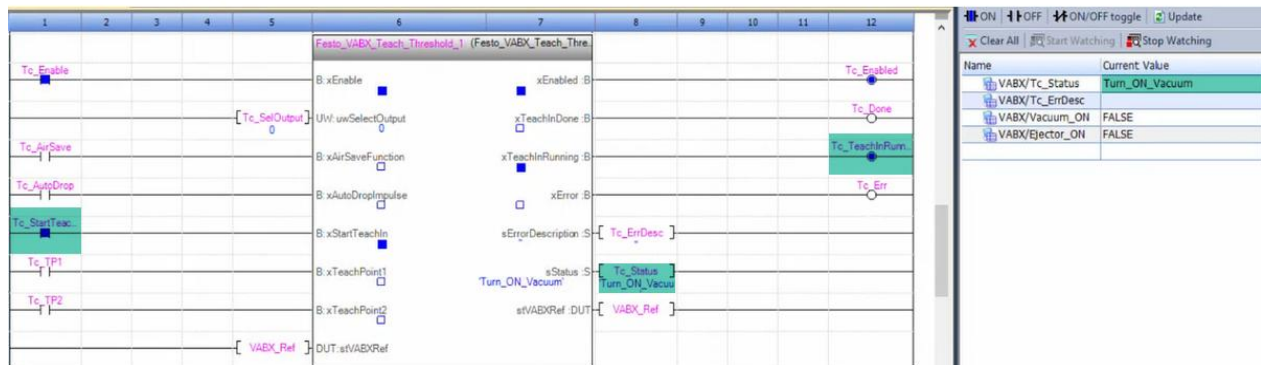
- The control block must be activated before using any TeachIn function blocks. This is because the Module Number validity will be checked, which applies to all other TeachIn blocks.

6.3.2 Ejector Pusle**Before Ejector ON****After Ejector ON**

1. Turn ON **xEnable** Input
2. Output **xEnabled** Turns TRUE
3. Turn ON **xEjectorPulse** Input
4. Output **xEjectorPulseAuto** turns TRUE
5. Monitor the Output **wActPressure** to see the changes in Actual pressure

6.3.3 Data Save**Before Data Save****After Data Save**

1. Turn ON **xEnable**
2. **xEnabled** Turns TRUE
3. Turn ON **xTriggerDataSave**
4. **xSavePermanentData** turns TRUE

Step 2**Start TeachIn**

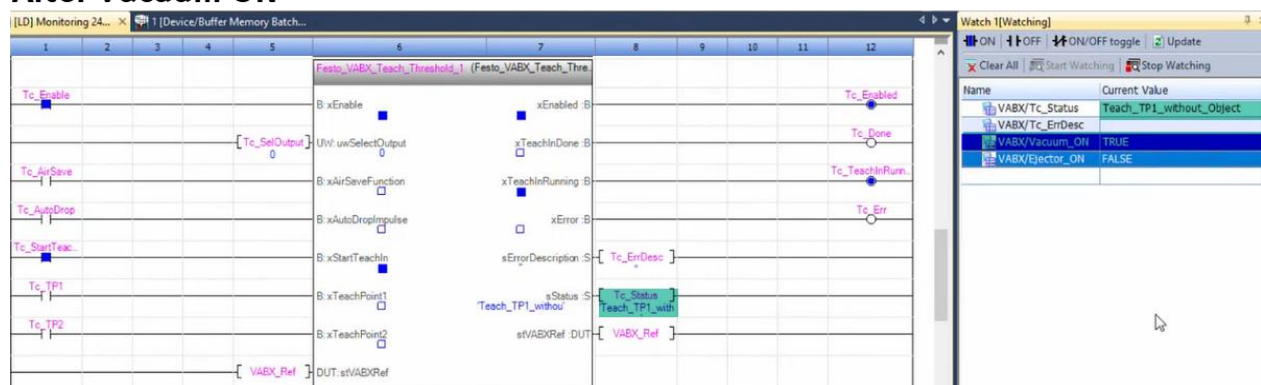
1. Turn ON/OFF **xAirSaveFunction** & **xAutoDropImpulse** input (as per user preference)
2. Select required **uwSelectOutput** Input
3. Turn ON **xStartTeachIn** input
4. Output **xTeachInRunning** turns **TRUE**
5. Output **sStatus** will update to **Turn ON Vacuum**

**Note**

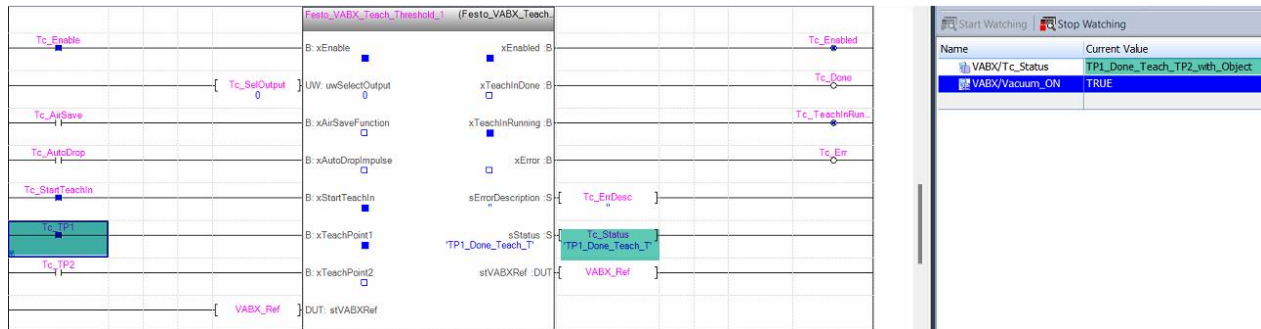
- Make sure Vacuum Input is turned OFF (Refer [Chapter 6.3.1](#))

Step 3

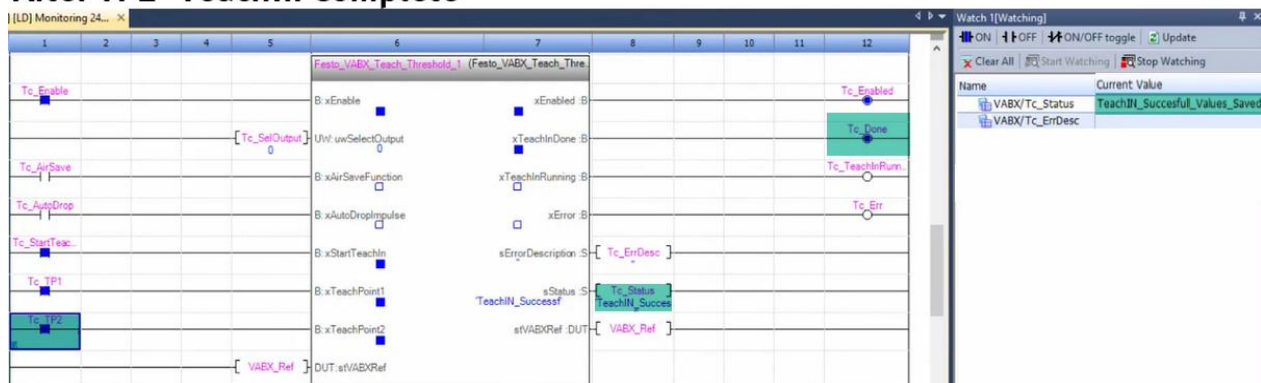
Turn ON Vacuum via **Festo_VABX_Control** block (Refer [Chapter 6.3.1](#))

After Vacuum ON

Output **sStatus** will update to **Teach TP1 without Object**

Step 4**After Teach point 1**

1. Turn ON **xTeachPoint1** Input
2. Output **sStatus** will update to **TP1 Done Teach TP2 with Object**

Step 5**After TP2 -TeachIn Complete**

1. Turn ON **xTeachPoint2** Input
2. Output **sStatus** will update to **TeachIN Completed Checking TeachIN Values**
3. Once Teach IN values validity are checked, **sStatus** will be updated to **TeachIN Successful Values Saved** and **xTeachInDone** turns TRUE

Before Teach In

Navigation	<	Parameters			
Terminal			Inresnoid process quality	30	%
▼ Modules			Limit Evacuation Time	0	s
▶ VABX-A-S-EL-E12...			Limit Venting Time	0	s
▶ VABX-A-S-VE-VBH			Auto-Drop Time	200	s
▼ VABX-A-S-VE-VBL			Automatic Drop Impulse	☐ Active	
Parameters			Air-Save Function	☐ Active	
Vacuum			Switching point A1	0	
Process Data			Hysteresis A	0	
Parameter List			Switching Point B1	0	
			Hysteresis B	0	
			Switching Point A2	0	
			Switching Point B2	0	
			Lock code	0	
			Switchpoint logic	☐ Active	
			Switchpoint mode	☐ Active	

After Teach In

Navigation	<	Parameters			
Terminal			Inresnoid process quality	30	%
▼ Modules			Limit Evacuation Time	0	s
▶ VABX-A-S-EL-E12...			Limit Venting Time	0	s
▼ VABX-A-S-VE-VBL			Auto-Drop Time	200	s
Parameters			Automatic Drop Impulse	☐ Active	
Vacuum			Air-Save Function	☐ Active	
Process Data			Switching point A1	70	
Parameter List			Hysteresis A	32	
			Switching Point B1	0	
			Hysteresis B	0	
			Switching Point A2	0	
			Switching Point B2	0	

Result:

By using OutA:

SetPoint A1, Hysteresis A and Setpoint A2 are set.

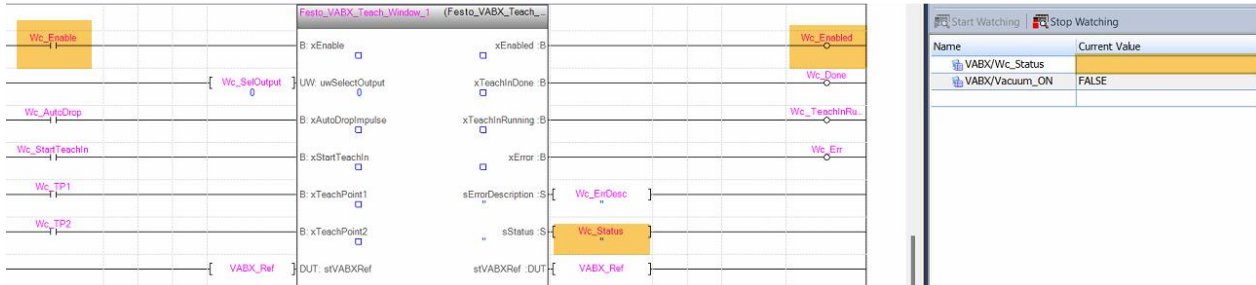
By using OutB:

SetPoint B1, Hysteresis B and Setpoint B2 are set.

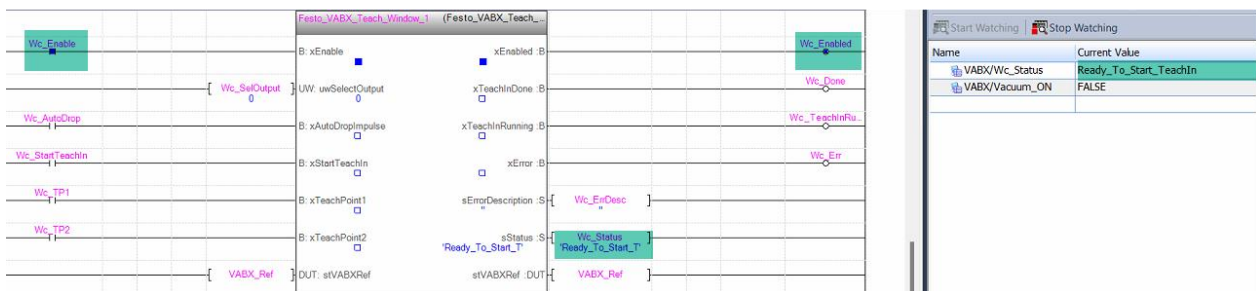
6.5 Window Comparator

Step 1

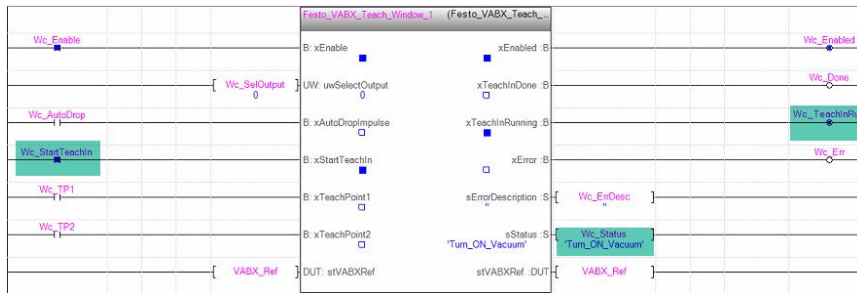
Before Enable



After Enable



1. Check all **Teach Inputs** are in **FALSE** condition
2. Select required **uwSelectOutput** Input
3. Enter 0 or 1 in **uwSelectOutput** as user preferred
4. Turn ON **xEnable** input
5. Output **xEnabled** turns **TRUE**
6. Output **sStatus** will be updated **Ready To Start TeachIn**

Step 2**After Start TeachIn**

Name	Current Value
VABX/Wc_Status	Turn_ON_Vacuum
VABX/Vacuum_ON	FALSE

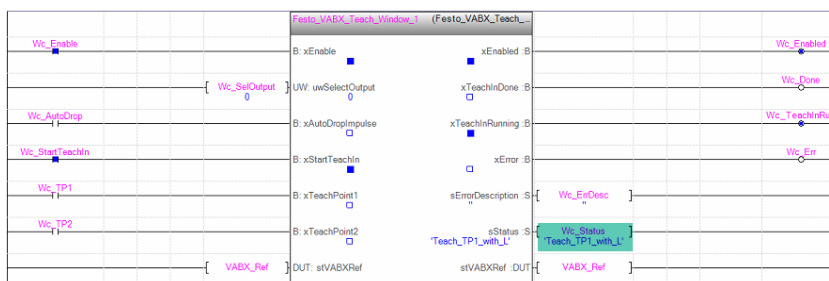
1. Turn ON **xStartTeachIn** input
2. Output **xTeachInRunning** turns **TRUE**
3. Output **sStatus** will update to **Turn ON Vacuum**

**Note**

- Make sure Vacuum Input is turned OFF (Refer [Chapter 6.3.1](#))

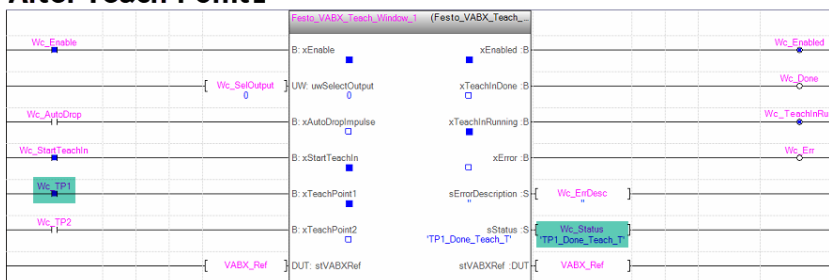
Step 3

Turn ON Vacuum via **Festo_VABX_Control** block (Refer [Chapter 6.3.1](#))

After Vacuum ON

Name	Current Value
VABX/Wc_Status	Teach_TP1_with_Low_Pressure
VABX/Vacuum_ON	TRUE

Output **sStatus** will update to **Teach TP1 with Low Pressure**

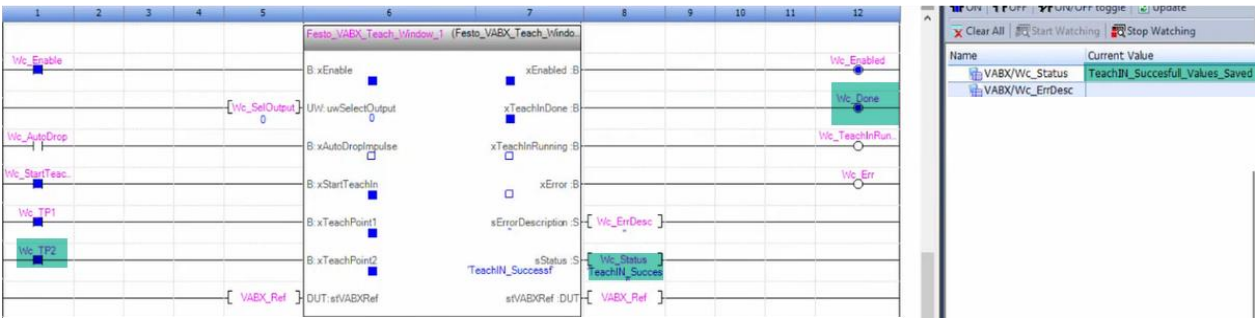
Step 4**After Teach Point1**

Name	Current Value
VABX/Wc_Status	TP1_Done_Teach_TP2_Hi_Pressure
VABX/Vacuum_ON	TRUE

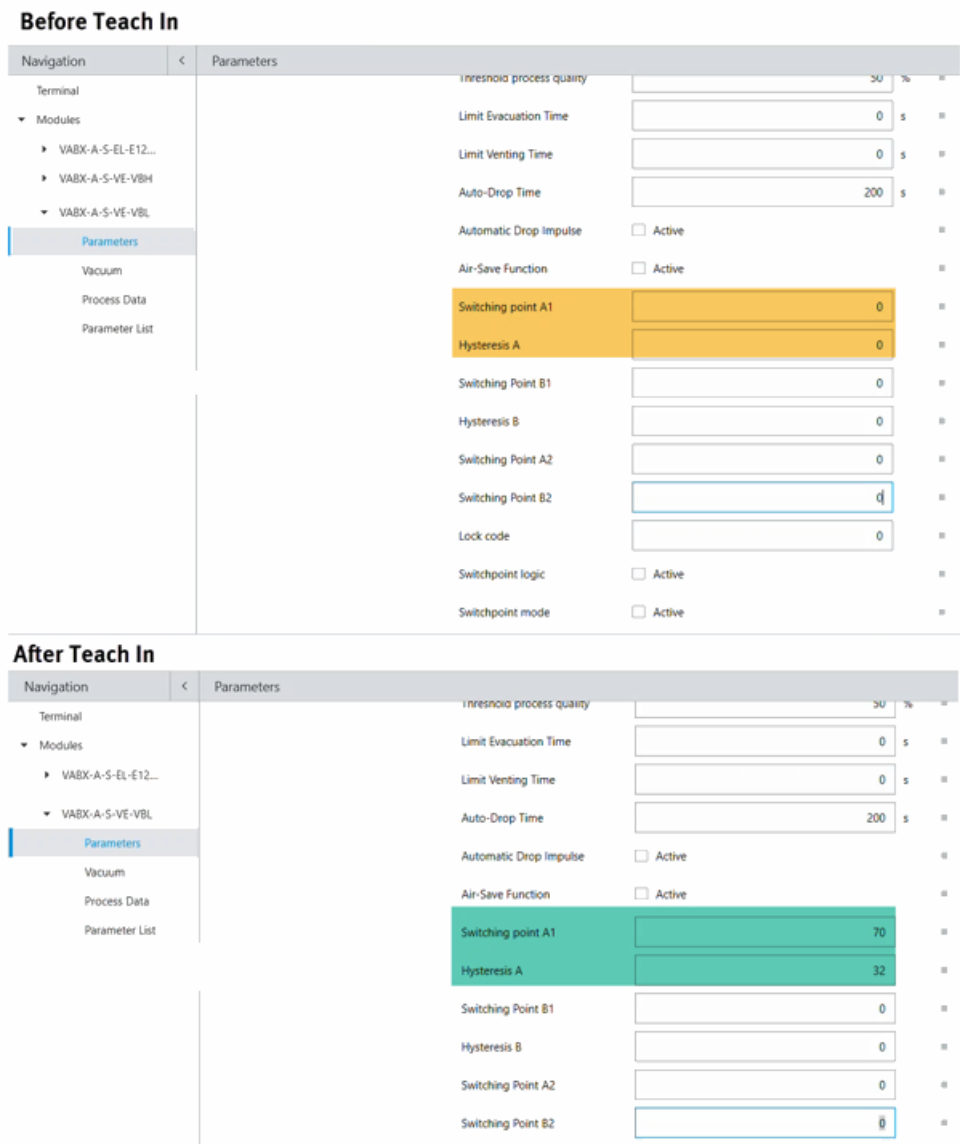
1. Turn ON **xTeachPoint1** Input
2. Output **sStatus** will update to **TP1 Done Teach TP2 with Hi Pressure**

Step 5

After TP2 - TeachIn Complete



- 1. Turn ON **xTeachPoint2** Input
- 2. Output **sStatus** will update to **TeachIn Completed Checking TeachIN Values**
- 3. Once Teach IN values validity are checked, **sStatus** will be updated to **TeachIN Successful Values Saved** and **xTeachInDone** turns TRUE



Result:

By using OutA:

SetPoint A1, Hysteresis A and Setpoint A2 are set.

By using OutB:

SetPoint B1, Hysteresis B and Setpoint B2 are set.

6.6 Dynamic TeachIn (Auto Drop Impulse Deactivated)

Navigation	
Terminal	Parameter List
▼ Modules	
▶ VABX-A-S-EL-E12...	
▶ VABX-A-S-VE-VBH	
▼ VABX-A-S-VE-VBL	
Parameters	
Vacuum	
Process Data	
Parameter List	

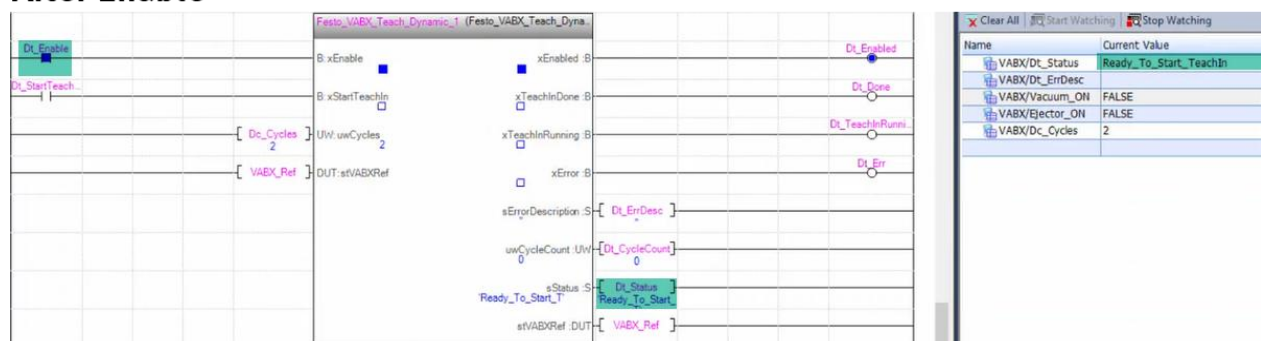
ID	Name	Value	Unit	
Standard Valve Parameters (32)				
P20240.0.0	Limit Evacuation Time	0	s	⋮
P20241.0.0	Limit Venting Time	0	s	⋮

Step 1

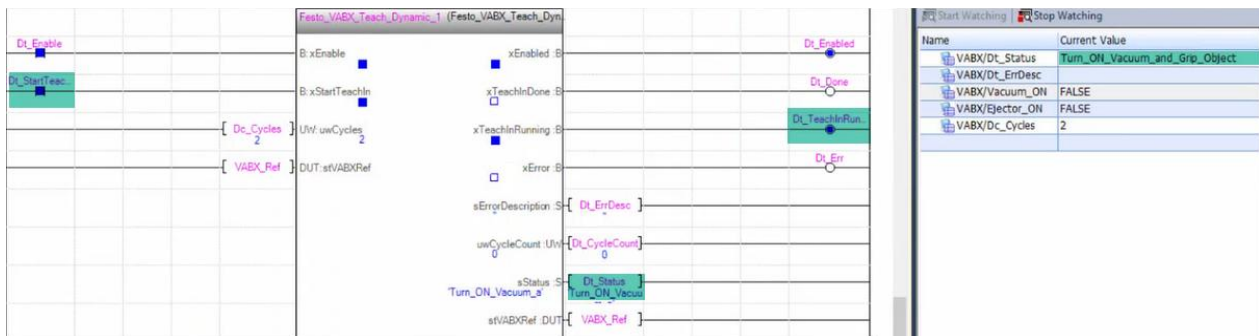
Before Enable



After Enable



1. Check **Teach Input** is in **FALSE** condition
2. Enter required No. of teach cycles in **uwCycles** Input
3. Turn ON **xEnable** input
4. Output **xEnabled** turns **TRUE**
5. Output **sStatus** will be updated **Ready To Start TeachIn**

Step 2**After Start TeachIn**

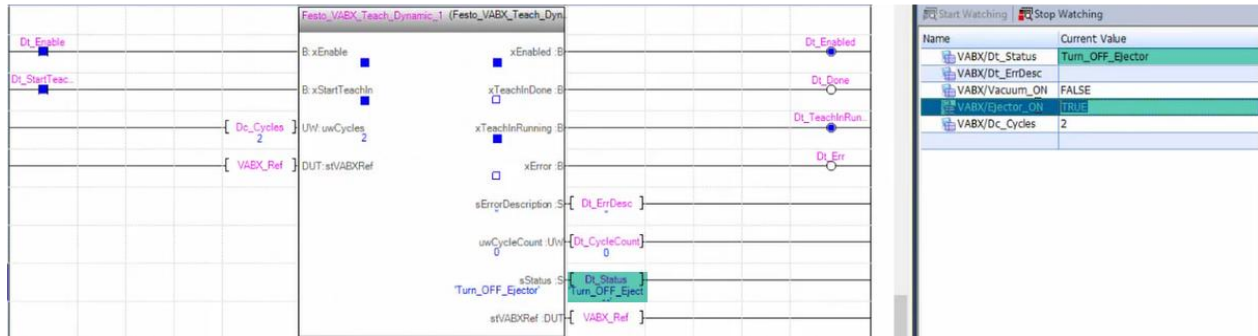
1. Turn **xStartTeachIn** Input
2. Output **xTeachInRunning** turns TRUE
3. Output **sStatus** will be updated **Turn ON Vacuum and Grip Object**

Step 3**After Vacuum ON**

1. Turn ON **Vacuum** (Output **xTeachInRunning** turns TRUE at 1st)
2. Output **sStatus** will be updated **Turn OFF Vacuum**

Step 4**After Vacuum OFF**

1. Turn OFF **Vacuum**
2. Output **sStatus** will be updated **Turn ON Ejector**

Step 5**After Ejector ON**

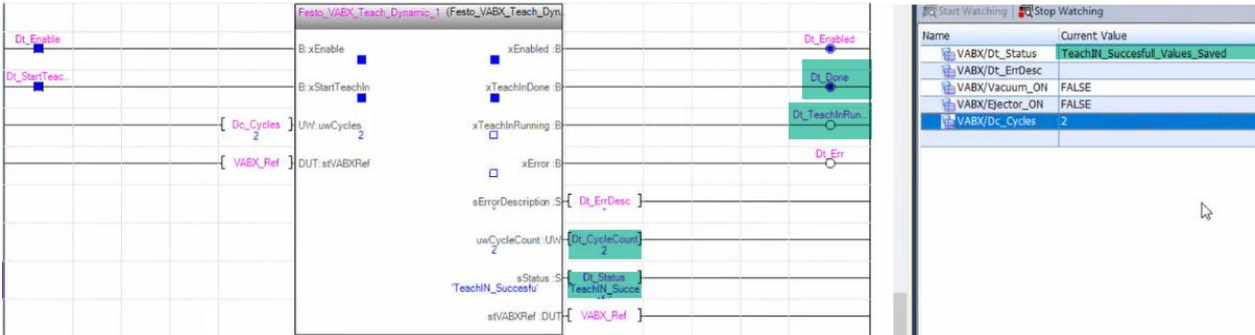
1. Turn ON **Ejector pulse**
2. Output **sStatus** will be updated **Turn OFF Ejector**

Step 6**After Ejector OFF**

1. Turn OFF **Ejector pulse**
2. Output **uwCycleCount** will be updated
3. Output **sStatus** will be updated
4. Repeat the above Steps 3 to 5 until Output **uwCycleCount** reaches **uwCycles** count entered in Input

Step 7

After Teach In Complete



1. Output **xTeachInRunning** turns FALSE
2. Output **xTeachInDone** turns TRUE
3. Output **sStatus** will be updated **TeachIN Successful Values Saved**



Note

- User must take care of Gripping Object properly; incorrect gripping leads to incorrect TeachIn.

After Teach Complete

Navigation	Parameter List
Terminal	
Modules	
VABX-A-S-EL-E12...	
VABX-A-S-VE-VBH	
VABX-A-S-VE-VBL	
Parameters	
Vacuum	
Process Data	
Parameter List	

ID	Name	Value	Unit
P20240.0.0	Limit Evacuation Time	192	s
P20241.0.0	Limit Venting Time	12	s

Result:

Limit Evacuation Time and limit venting time is set.

6.7 Dynamic TeachIn (Auto Drop Impulse Activated)

1. Check **Teach Input** is in **FALSE** condition
2. Enter required No. of teach cycles in **uwCycles** Input
3. Turn ON **xEnable** input
4. Output **xEnabled** turns **TRUE**
5. Output **sStatus** will be updated **Ready To Start TeachIn**
6. Turn ON **xStartTeachIn** Input
7. Output **sStatus** will be updated **Grip Object and Turn ON Vacuum**
8. Grip the Object
9. Turn ON **Vacuum** (Output **xTeachInRunning** turns TRUE at 1st)
10. Output **sStatus** will be updated **Turn OFF Vacuum**
11. Turn OFF **Vacuum**
12. Output **sStatus** will be updated **Turn ON Ejector**
13. The **Ejector** will turn ON (for the duration of the Auto Drop Impulse Time) and then turn OFF if the **Auto Drop Impulse** is **activated** and the **Auto Drop Impulse Time** parameter is set to a value **other than zero**.
14. Output **sStatus** will be updated Automatically with **Turn ON Ejector & Turn OFF Ejector**
15. Output **uwCycleCount** will be updated
16. Output **sStatus** will be updated
17. Repeat the above steps until Output **uwCycleCount** reaches **uwCycles** count entered in Input
18. Output **xTeachInRunning** turns FALSE
19. Output **xTeachInDone** turns TRUE

6.8 Festo_VABX_Parameter Block

6.8.1 Parameter Read

Before Execution



ID	Name	Value	Unit
P20212.0.0	Interlock ejector pulse	Inactive (0)	
P20221.0.0	Threshold process quality	50	%
P20240.0.0	Limit Evacuation Time	1038	s
P20241.0.0	Limit Venting Time	42	s
P20242.0.0	Auto-Drop Time	200	s
P20243.0.0	Automatic Drop Impulse	☐ Active	
P20244.0.0	Air-Save Function	☐ Active	
P20245.0.0	Switching point A1	72	
P20246.0.0	Hysteresis A	12	
P20247.0.0	Switching Point B1	0	
P20248.0.0	Hysteresis B	0	
P20249.0.0	Switching Point A2	0	
P20250.0.0	Switching Point B2	0	
P20252.0.0	Lock code	0	
P20253.0.0	Switchpoint logic	☐ Active	
P20254.0.0	Switchpoint mode	☐ Active	

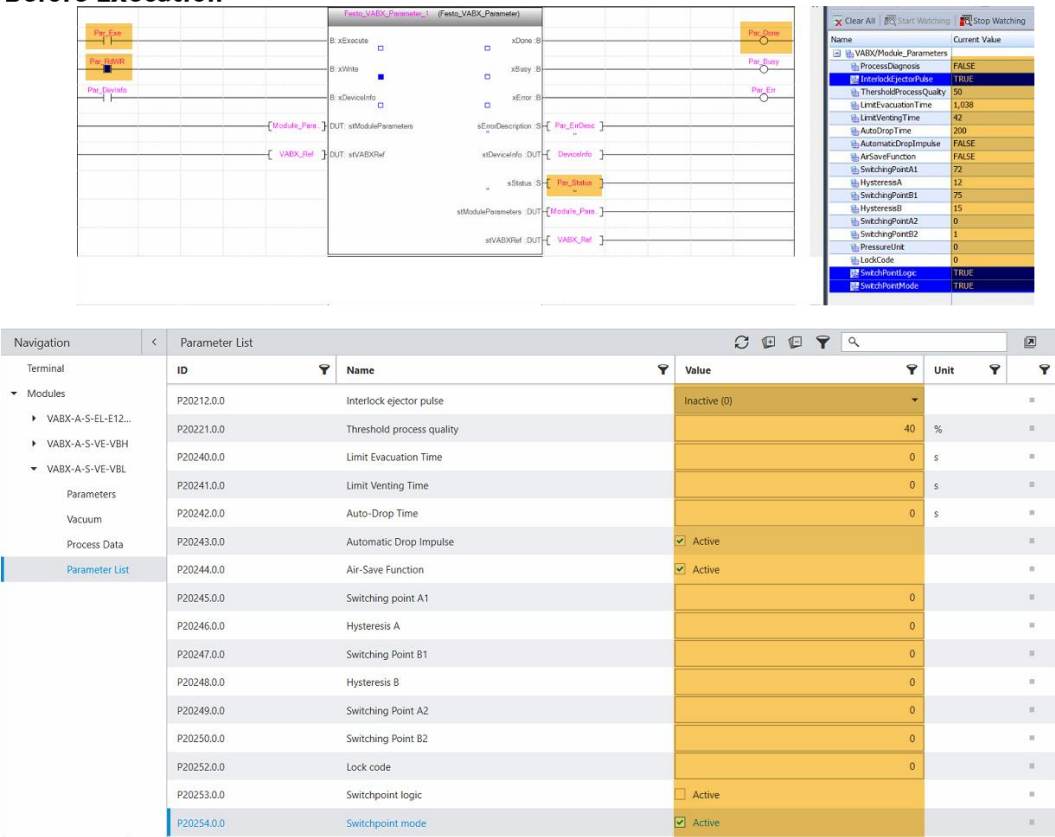
After Execution



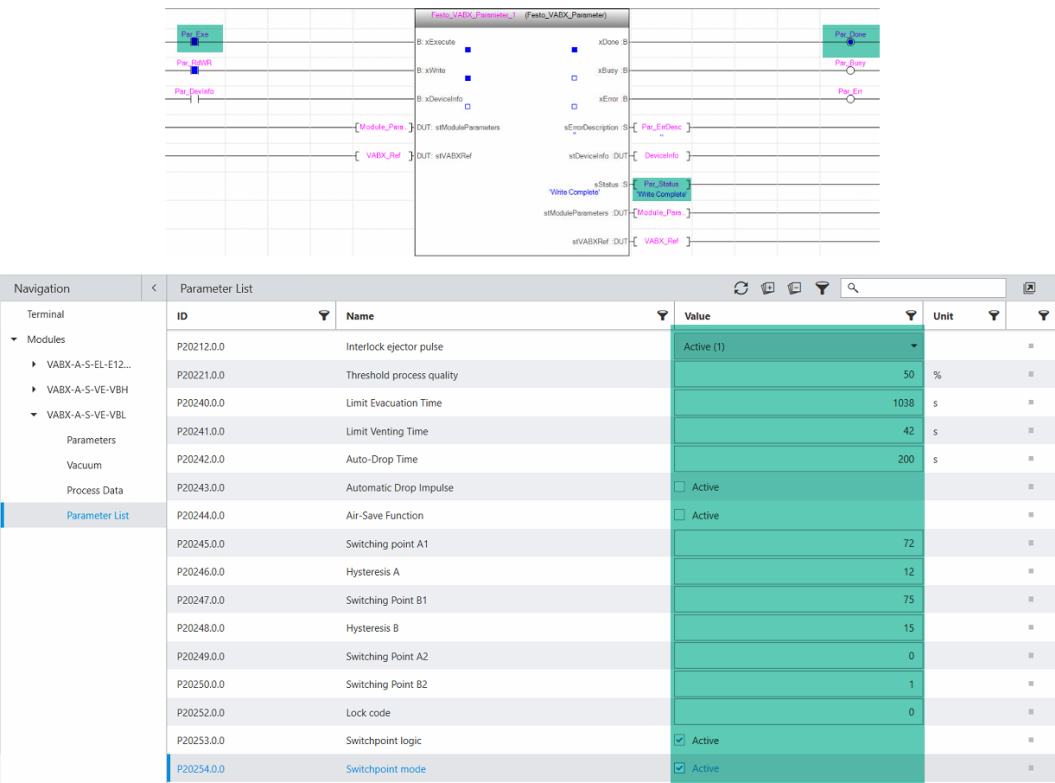
1. Turn ON **xDeviceInfo** Input (as per user preference TRUE- Reads Device Information FALSE- not Reads Device Information)
2. Turn OFF **xWrite** Input
3. Turn ON **xExecute** Input
4. Output **xBusy** turns ON while reading parameters
5. Output **sStatus** will be updated with current parameters reading
6. Output **xDone** turns ON once parameter read is completed
7. Output **sStatus** will be updated with **Read Complete**
8. Output **stDeviceInfo** will be updated with values (if **xDeviceInfo** is TRUE)
9. In-Out **stModuleParameters** will be updated with Read Data

6.8.2 Parameter Write

Before Execution



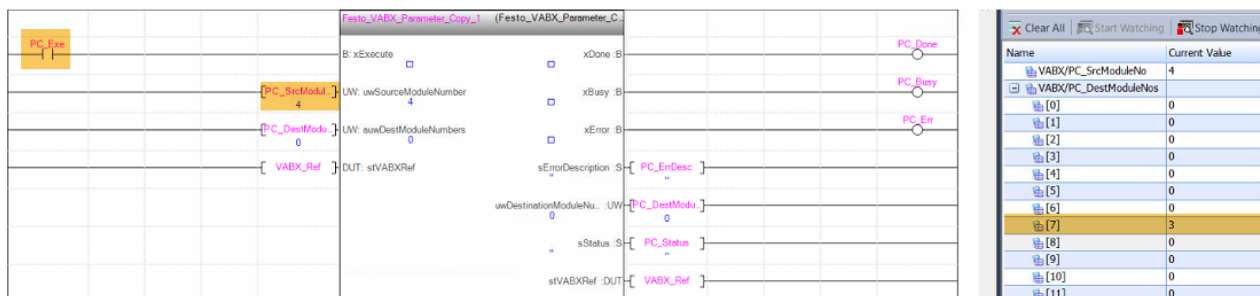
After Execution



1. Turn ON **xWrite** Input
2. Enter required Module Parameter in **stModuleParameters**
3. Turn ON **xExecute** Input
4. Output **xBusy** turns ON while writing parameters
5. Output **sStatus** will be updated with current parameters writing
6. Output **xDone** turns ON once parameter write is completed
7. Output **sStatus** will be updated with **Write Complete**

6.9 Festo_VABX_Parameter_Copy Block

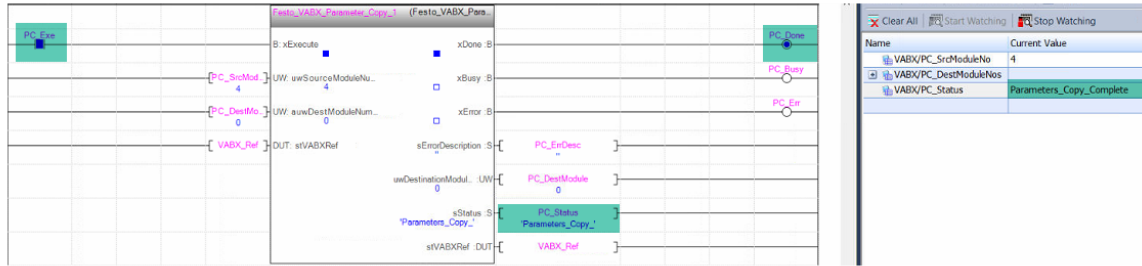
Before Execution



Source Module

Navigation	ID	Name	Value	Unit	
Terminal					
Modules					
VABX-A-S-EL-E12...	P20212.0.0	Interlock ejector pulse	Active (1)		
VABX-A-S-VE-VBH	P20221.0.0	Threshold process quality	50	%	
VABX-A-S-VE-VBL	P20240.0.0	Limit Evacuation Time	1038	s	
Parameters	P20241.0.0	Limit Venting Time	42	s	
Vacuum	P20242.0.0	Auto-Drop Time	200	s	
Process Data	P20243.0.0	Automatic Drop Impulse	☐ Active		
Parameter List	P20244.0.0	Air-Save Function	☐ Active		
	P20245.0.0	Switching point A1	72		
	P20246.0.0	Hysteresis A	12		
	P20247.0.0	Switching Point B1	75		
	P20248.0.0	Hysteresis B	15		
	P20249.0.0	Switching Point A2	0		
	P20250.0.0	Switching Point B2	1		
	P20252.0.0	Lock code	0		
	P20253.0.0	Switchpoint logic	☒ Active		
	P20254.0.0	Switchpoint mode	☒ Active		

After Execution



Destination Module

Navigation	Parameter List	ID	Name	Value	Unit	
Terminal						
▼ Modules						
▶ VABX-A-S-EL-E12...		P20212.0.0	Interlock ejector pulse	Active (1)		■
▼ VABX-A-S-VE-VBH		P20221.0.0	Threshold process quality	50	%	■
Parameters		P20240.0.0	Limit Evacuation Time	1038	s	■
Vacuum		P20241.0.0	Limit Venting Time	42	s	■
Process Data		P20242.0.0	Auto-Drop Time	200	s	■
Parameter List		P20243.0.0	Automatic Drop Impulse	☐ Active		■
▼ VABX-A-S-VE-VBL		P20244.0.0	Air-Save Function	☐ Active		■
Parameters		P20245.0.0	Switching point A1	72		■
Vacuum		P20246.0.0	Hysteresis A	12		■
Process Data		P20247.0.0	Switching Point B1	75		■
Parameter List		P20248.0.0	Hysteresis B	15		■
		P20249.0.0	Switching Point A2	0		■
		P20250.0.0	Switching Point B2	1		■
		P20252.0.0	Lock code	0		■
		P20253.0.0	Switchpoint logic	☒ Active		■
		P20254.0.0	Switchpoint mode	☒ Active		■

1. Enter required Source Module number in **uwSourceModuleNumber** Input (Refer [Chapter 3.1.5](#))
2. Enter required & valid Destination Module numbers in **auwDestModuleNumbers** Input (Make sure Source and Destination Module numbers are not same)
3. Turn ON **xExecute** Input
4. Output **xBusy** turns ON while Copying parameters
5. Output **uwDestinationModuleNumber** will be updated with Destination Module Number
6. Output **sStatus** will be updated
7. Output **xDone** turns ON once parameter copying is completed
8. Output **sStatus** will be updated with **Parameters Copy Complete**