

VTEM with Keyence KV-X PLC through EtherCAT

This Application Note explains how to integrate the VTEM Motion Terminal and the function block library for the Festo VTEM Motion Terminal through EtherCAT with Keyence KV-X PLC in KV STUDIO programming environment.

VTEM

Title VTEM with Keyence KV-X PLC through EtherCAT
Version 1.10
Document no. 100827
Original en
Author Festo

Last saved 08.10.2026

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.

Table of Contents

1	Introduction	5
1.1	Software Used	6
1.2	Hardware Used	6
1.3	Topology Overview	7
2	Hardware Configuration	8
2.1	VTEM Motion Terminal Overview	8
2.2	CPX-FB37 Bus Node Overview	8
2.3	Overview of DIL Switch Setting of CPX-FB37	9
2.3.1	Setting Operating Mode and EtherCAT interface	9
2.3.2	Setting diagnostics mode, I/O bytes or starting of the bootloader	10
2.3.3	Setting the EtherCAT address	10
2.3.4	4-pin M12 socket (D-coded) Connection Details	10
3	Software Configuration	11
3.1	Creating a New Project and Adding Controller in KV STUDIO	11
3.2	Setting Up the Controller	12
3.2.1	Built-in Ethernet Port Configuration	12
3.2.2	PLC Communication Setup	13
3.3	Installing CPX-FB37 ESI File in KV STUDIO	15
3.4	CPX-FB37 Module Configuration	17
3.4.1	Scanning the CPX-FB37 Device Connected to the EtherCAT Master	17
3.4.2	CPX-FB37 Node Settings	18
3.4.3	Creating Device Variables for CPX-FB37 Modules	20
3.5	Adding Festo_VTEM_EC Library to a KV STUDIO Project	22
4	VTEM Process Data details	25
4.1	VTEM Motion Terminal Communication Protocol	25
4.1.1	Motion Terminal VTEM Function and Parameterization	25
4.2	Connecting I/O Data to the Function Blocks	28
5	Festo_VTEM_EC Library Function Block Descriptions	29
5.1	FB_ValveControl	29
5.1.1	FB_ValveControl Block Input Variable	30
5.1.2	FB_ValveControl Block Output Variable	30
5.1.3	Response to Valve Mode Set	31
5.2	FB_Upload_Single_Parameter	32
5.2.1	FB_Upload_Single_Parameter Block Input Variable	32
5.2.2	FB_Upload_Single_Parameter Block Output Variable	32

5.3	FB_Download_Single_Parameter	33
5.3.1	FB_Download_Single_Parameter Block Input Variable	33
5.3.2	FB_Download_Single_Parameter Block Output Variable	33
5.3.3	Response to Transfer Mode	34
5.4	FB_Save_Configuration	35
5.4.1	FB_Save_Configuration Block Input Variable	35
5.4.2	FB_Save_Configuration Block Output Variable	35
5.4.3	Response to Save Persistent.....	36
5.5	FB_ReadMalfunctionList.....	36
5.5.1	FB_ReadMalfunctionList Block Input Variable	37
5.5.2	FB_ReadMalfunctionList Block Output Variable.....	37
5.5.3	Response to Read Malfunction List.....	38
5.5.4	Structure stMalfunction	39
5.5.5	Structure stTimeStamp	39
6	Configuring of Festo_VTEM_EC Function Blocks in KV STUDIO	40
7	Sample Code for Festo_VTEM_EC Library Function Blocks	44
7.1	Sample Code for Linking VTEM Motion Terminal Process Data of Valve in Slot 0 to Function Block Interface Tags.....	44
7.2	Sample Code for Controlling the VTEM Motion Terminal Valve using FB_ValveControl	45
7.3	Sample Code for Reading the Valve Parameter-Set Value from the VTEM Motion Terminal using FB_Upload_Single_Parameter.....	46
7.4	Sample Code for Uploading Parameter from the VTEM Motion Terminal Valve using FB_Upload_Single_Parameter.....	47
7.5	Sample Code for Downloading Parameter to the VTEM Motion Terminal Valve using FB_Download_Single_Parameter	48
7.6	Sample Code for Saving the Parameters in the VTEM Motion Terminal Valve using FB_Save_Configuration	49
7.7	Sample code for Reading Malfunction list of the VTEM Motion Terminal Valve using FB_ReadMalfunctionList.....	50

1 Introduction

This application note demonstrates integration and controlling of the Festo VTEM Motion Terminal with KEYENCE KV-X PLC through EtherCAT in KV STUDIO.

This library contains five function blocks used to control, download, upload, save parameters to VTEM Motion Terminal & diagnose the error details of the Motion Terminal.

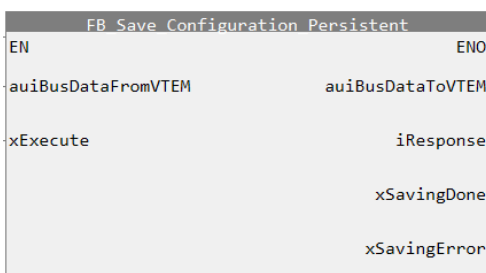
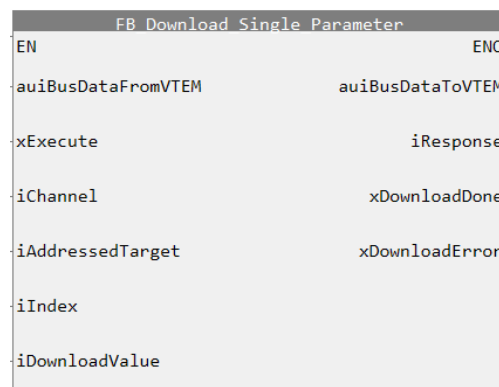
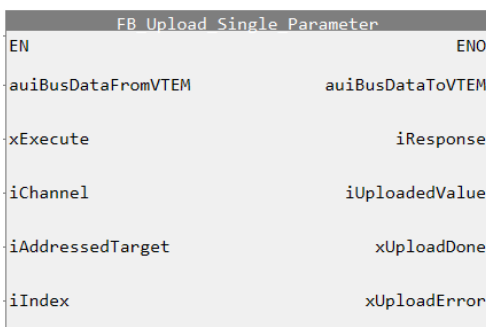
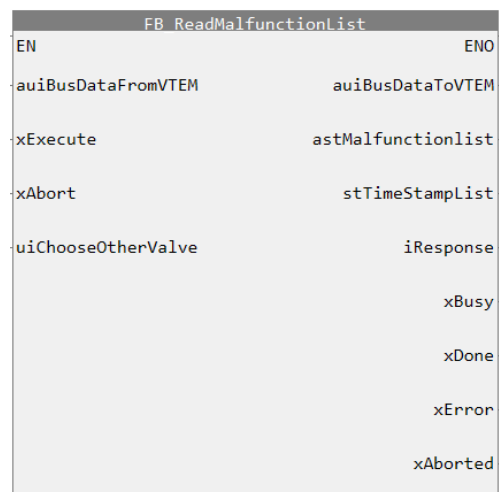
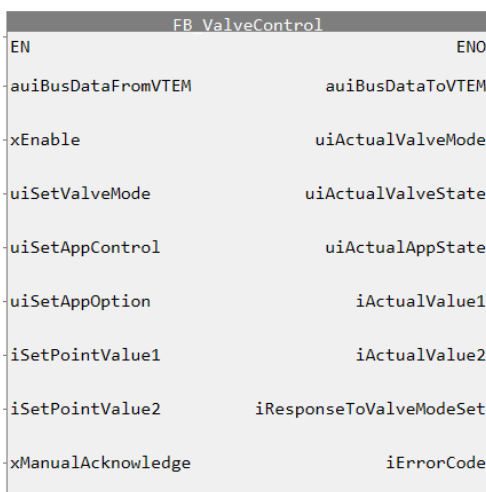
FB_ValveControl block is used to control a single valve in the Motion Terminal.

FB_Upload_Single_Parameter block is used to read the Motion Terminal valve parameter.

FB_Download_Single_Parameter block is used to write the Motion Terminal valve parameter.

FB_Save_Configuration block is used to save the Motion Terminal valve parameter.

FB_ReadMalfunctionList block is used to read the malfunction list of the VTEM Motion Terminal.



1.1 Software Used

Software Name	Software Version
KEYENCE KV STUDIO	V12.41
Web Browser	Version Free
CPX-FB37 ESI File	V2211
Library (Festo_VTEM_EC)	V12.41.10

Table 1.1: Software used

1.2 Hardware Used

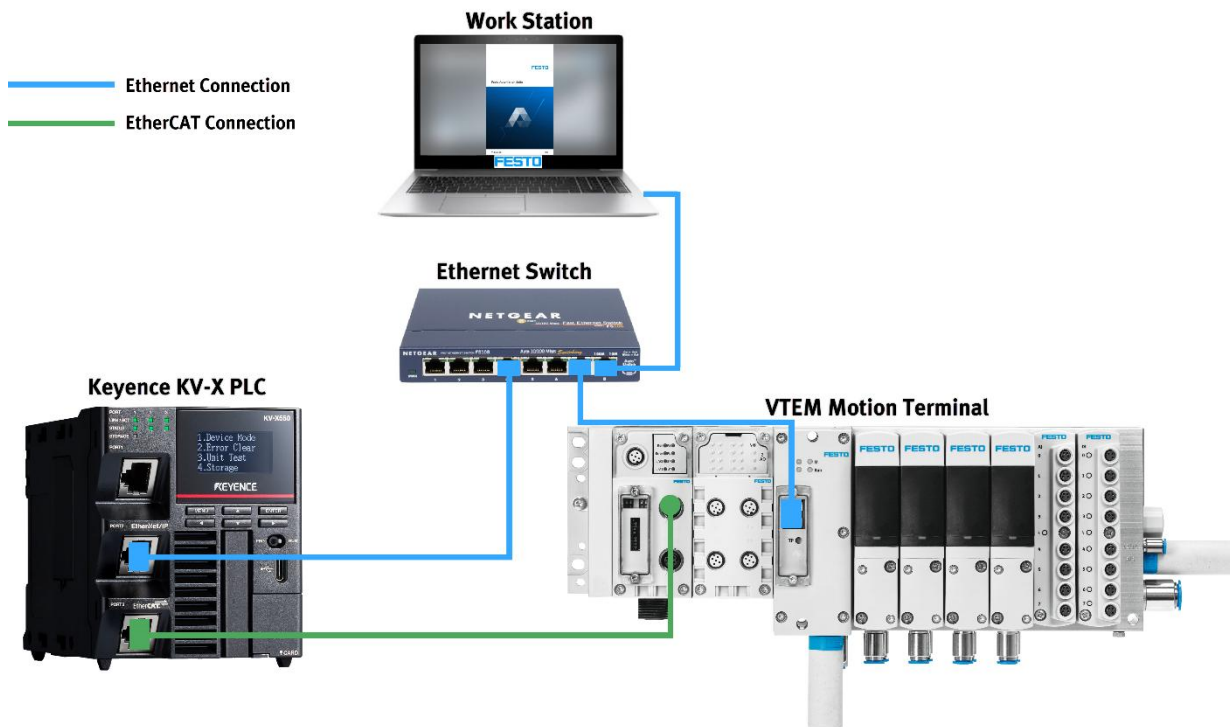
Device Name	Device Type	Version / Firmware
KEYENCE KV-X PLC	KV-X520	V1.502
Mini-USB Cable with NEFC Electrical Adapter	547432	Version Free
Festo Motion Terminal	VTEM	V4.26.6 (Rev17)
Festo Bus Node	CPX-FB37	Version R09
M12 – RJ45 Ethernet Cable	Version Free	Version Free

Table 1.2: Hardware used

1.3 Topology Overview

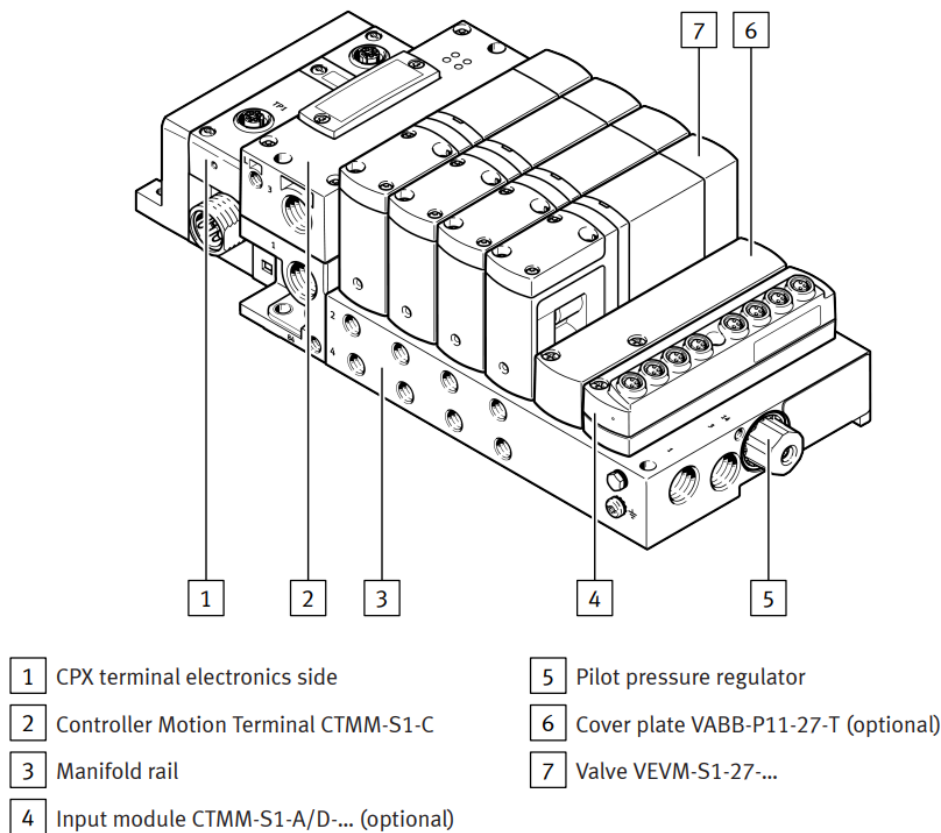
**Note**

- The following Topology Overview is only an example for this application note. It may vary depending on user's network configuration.

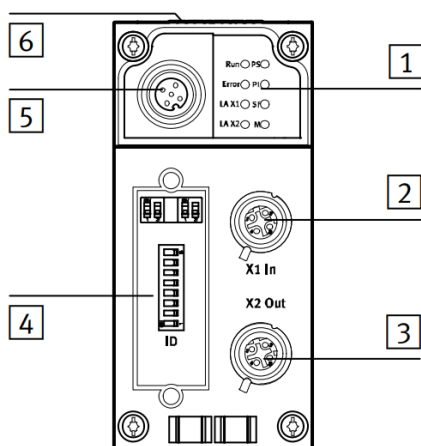


2 Hardware Configuration

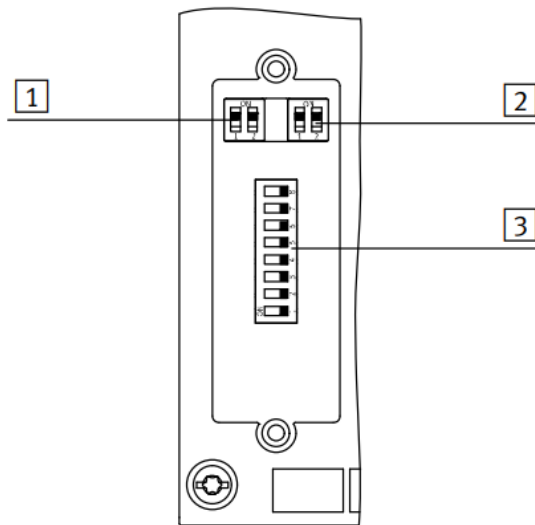
2.1 VTEM Motion Terminal Overview



2.2 CPX-FB37 Bus Node Overview



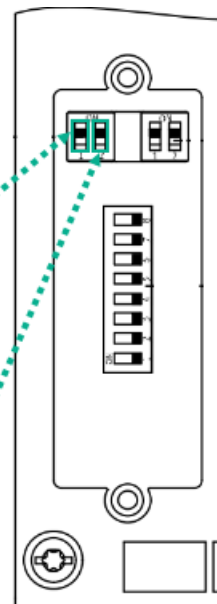
2.3 Overview of DIL Switch Setting of CPX-FB37



- 1 DIL switch 1: Operating mode, EtherCAT interface
- 2 DIL switch 2: Diagnostics mode, data field size, bootloader
- 3 DIL switch 3: EtherCAT address

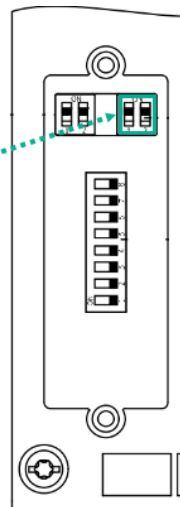
2.3.1 Setting Operating Mode and EtherCAT interface

DIL switch 1	Setting	Operating mode
	DIL 1.1: OFF (Factory setting)	Remote I/O All functions of the CPX terminal are controlled directly by the EtherCAT I/O controller or a higher-level PLC.
	DIL 1.1: ON	Remote controller A CPX-FEC or CPX-CEC integrated into the terminal controls all functions
DIL switch 1	Setting	Function
	DIL 1.2: OFF (Factory setting)	Modular device profile (MDP)
	DIL 1.2: ON	Fixes I/O size (64 byte I/A), compatible with the bus node CPX-FB38



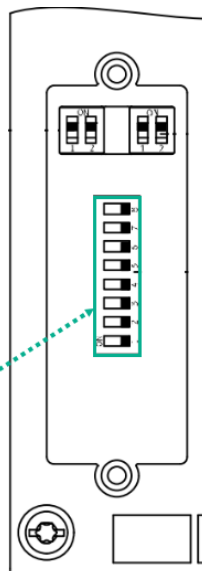
2.3.2 Setting diagnostics mode, I/O bytes or starting of the bootloader

DIL switch 2	Setting	Remote I/O	Remote controller
	DIL 2.1: OFF DIL 2.2: OFF (Factory setting)	I/O diagnostics interface and status bits switched off	8 byte I/8 byte O for communication of the bus node with the CPX-FEC or CPX-CEC
	DIL 2.1: OFF DIL 2.2: ON	Status bits are switched on + 16 I bits (only 8 bits used)	Reserved
	DIL 2.1: ON DIL 2.2: OFF	I/O diagnostics interface is switched on + 16 I/O bits	16 byte I/16 byte O for communication of the bus node with the CPX-FEC or CPX-CEC
	DIL 2.1: ON DIL 2.2: ON	Bootloader (Restore the firmware in case of loss of function)	Bootloader (Restore the firmware in case of loss of function)



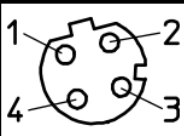
2.3.3 Setting the EtherCAT address

DIL switch 3	Function
	Binary-coded input of the EtherCAT address ¹⁾ Example on left in figure: EtherCAT address = $2^1 + 2^2 + 2^5$ = $2 + 4 + 32 = 38$
	Binary-coded input of the EtherCAT address Example on left in figure (all switch elements are OFF): EtherCAT address = 0, the programmed EtherCAT address in the EEPROM is used (factory setting).



To avoid setting the CPX-FB37 module's IP address via DIL switches, ensure all switches from DIL 3.1 to 3.8 are off.

2.3.4 4-pin M12 socket (D-coded) Connection Details

M12 socket, D-coded	Pin	Signal	Explanation
	1	TD+	Transmitted data (Transmit Data) +
	2	RD+	Received data (Receive Data) +
	3	TD-	Transmitted data -
	4	RD-	Received data -
	Housing	FE	Screening/functional earth

3 Software Configuration

This chapter describes KEYENCE KV STUDIO software configuration such as creating a new project and adding the controller, installing CPX-FB37 ESI file, configuring CPX-FB37 & adding **Festo_VTEM_EC** library to the project.

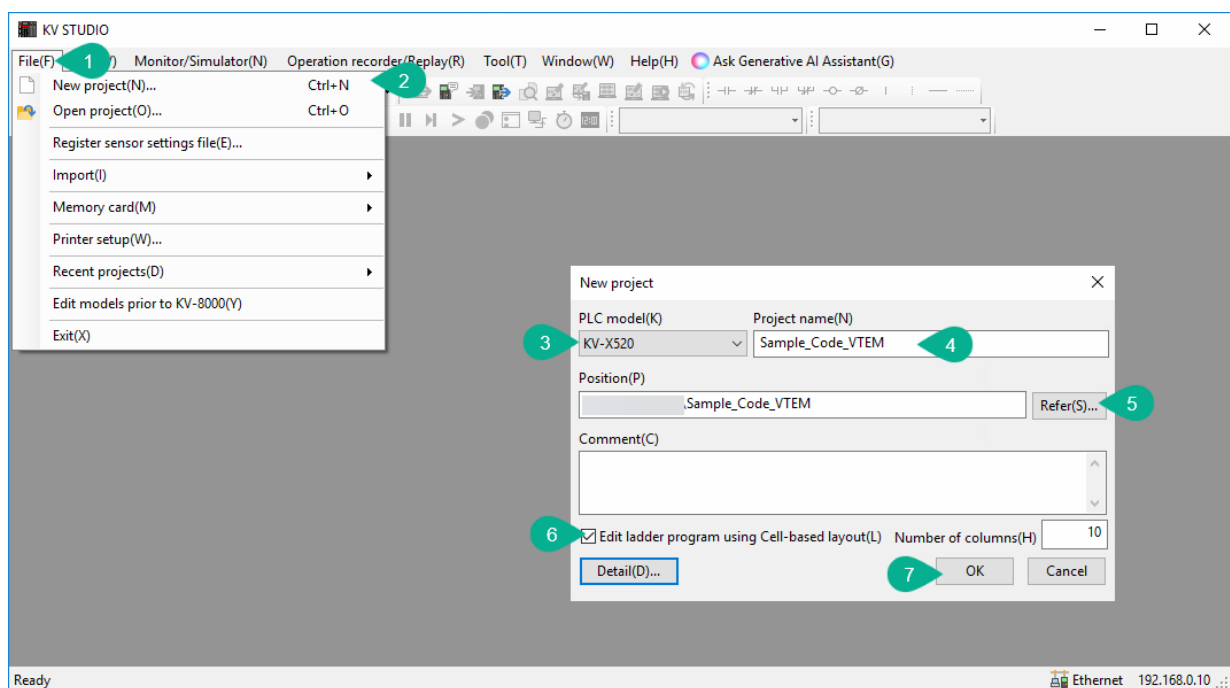
3.1 Creating a New Project and Adding Controller in KV STUDIO

Step 1



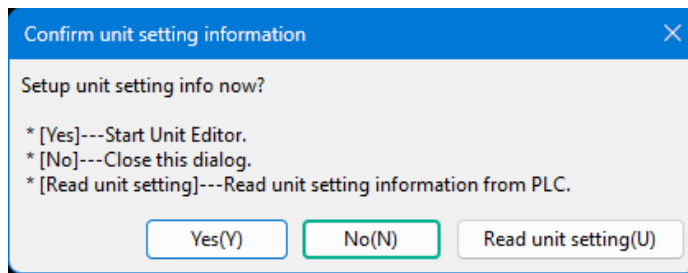
Open the **KV STUDIO Ver.12G** software (**Windows Start >> KEYENCE KV STUDIO Ver.12G >> KV STUDIO Ver.12G**).

Step 2



1. Click the **File(F)** menu option from toolbar.
2. Click the **New project(N)...** option from **File** menu.
3. Select the **KV-X520** from the **PLC model(K)** drop-down list. The user can select the PLC as required.
4. Enter the required name for the project in the **Project name(N)** field.
5. Click the **Refer(S)...** button to select the folder in which to save the project.
6. Select the **Edit ladder program using Cell-based layout(L)** option checkbox, as required.
7. Click the **OK** button to create the project.

Step 3

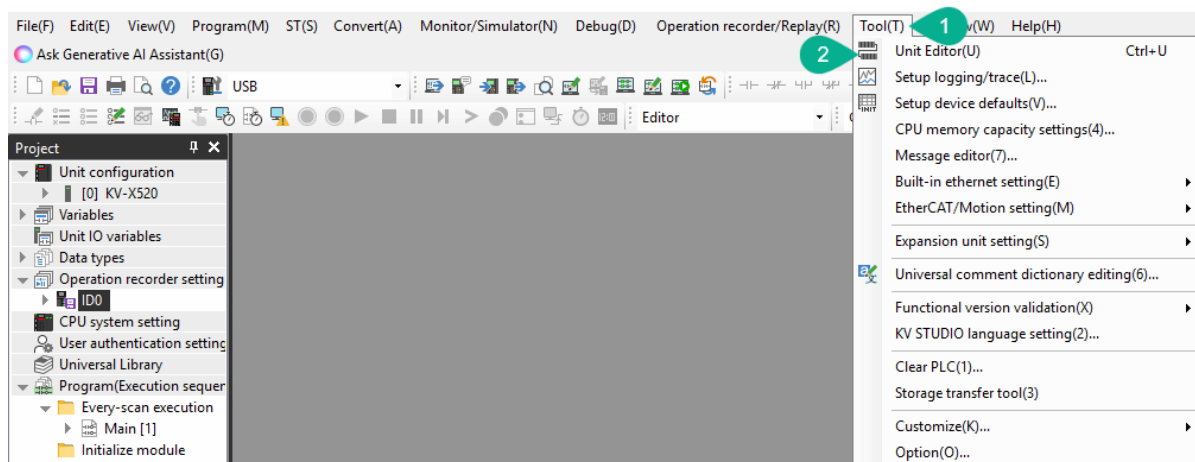


Click the **No** button to close the **Confirm unit setting information** window.

3.2 Setting Up the Controller

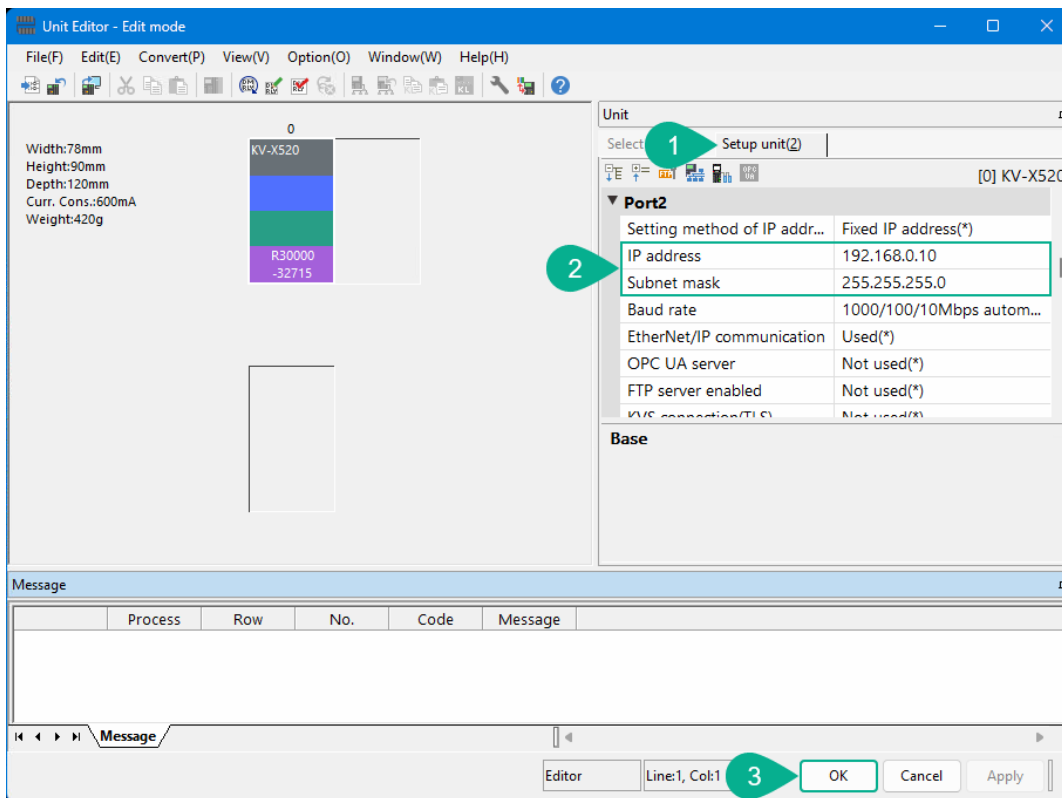
3.2.1 Built-in Ethernet Port Configuration

Step 1



1. Click the **Tool(T)** menu option from toolbar.
2. Click the **Unit Editor(U)...** option from **Tool(T)** menu.

Step 2



1. Click the **Setup unit(2)** tab in the **Unit Editor** window.
2. Enter the required **IP Address** and **Subnet Mask** value for the PLC in the **Port 2** settings. (For this sample code, 192.168.0.10 is used as the PLC IP address, and 255.255.255.0 is used as the subnet mask).
3. Click the **OK** button to apply the changes and close the window.



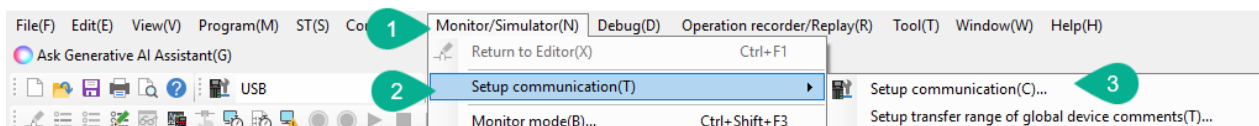
Note

Port Configuration Selection.

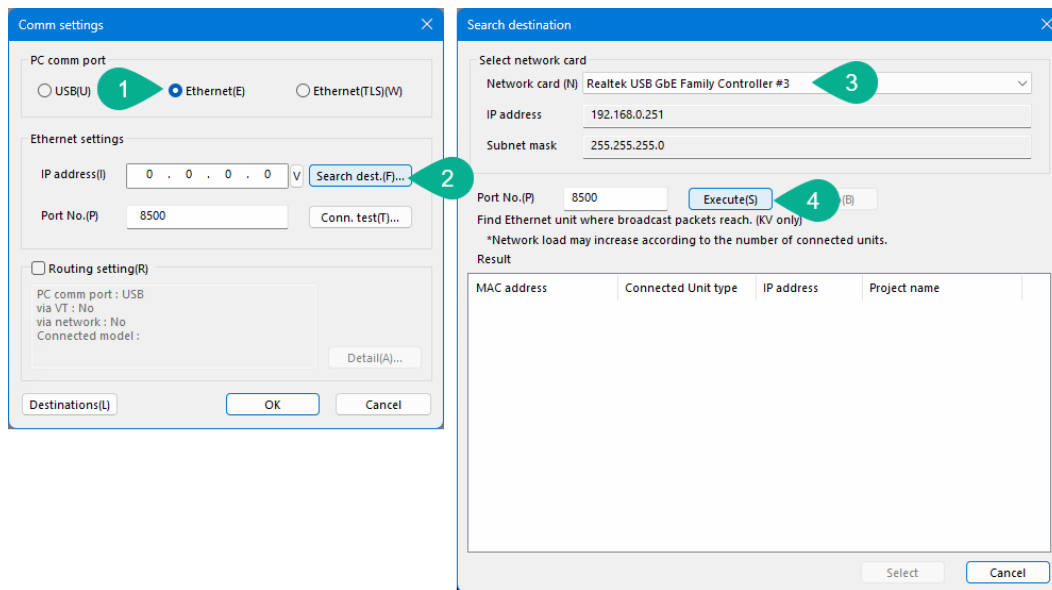
- The KEYENCE KV-X520 PLC has **two Ethernet ports** that serve as programming interfaces. In this example, the PLC is connected to the workstation via **Ethernet Port 2**. [Refer Chapter 1.3](#).

3.2.2 PLC Communication Setup

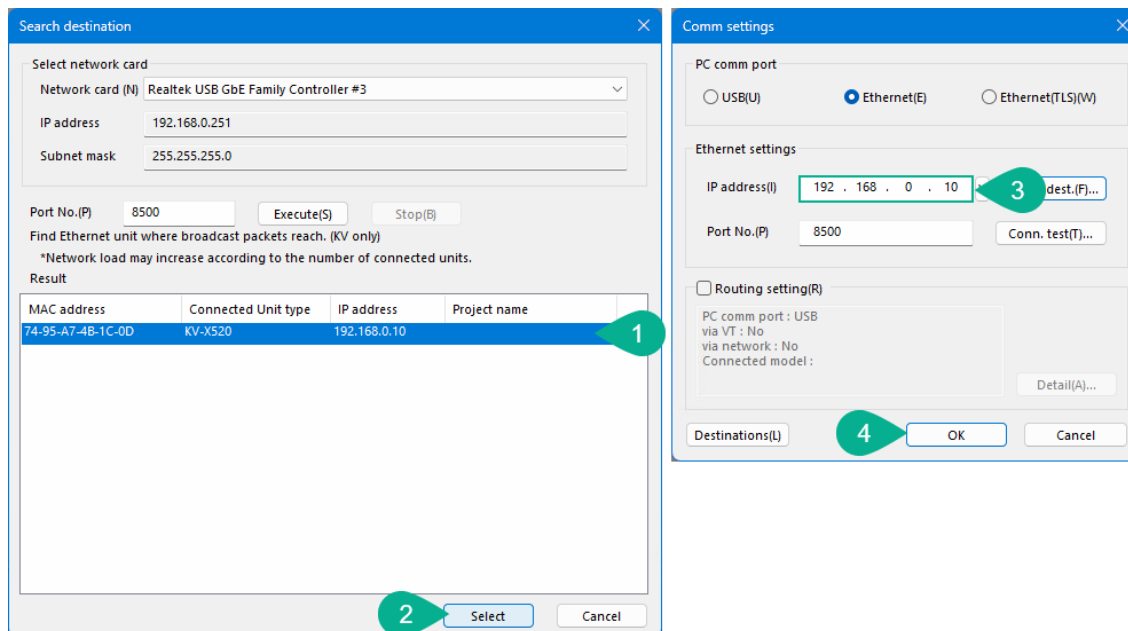
Step 1



1. Click the **Monitor/Simulator(N)** menu option from toolbar.
2. Select the **Setup communication(T)** option.
3. Click the **Setup communication(C)...** option to start the communication setup.

Step 2

1. Select the **Ethernet(E)** option under the **PC comm port** settings in **Comm settings** window.
2. Click the **Search dest.(F)...** button under the **Ethernet settings** option.
3. Select the required **Network Card** from the **Network card (N)** drop-down list in the **Search destination** window.
4. Click the **Execute(S)** button to search for the connected controller.

Step 3

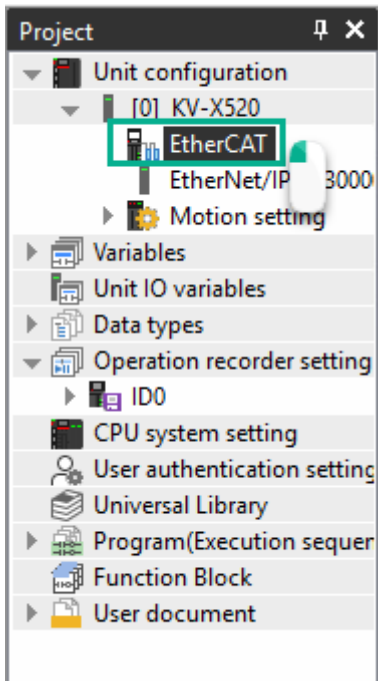
1. Once the connected controller is detected, it will appear in the **Result** area as shown.
2. Select the detected controller and click the **Select** button to apply the settings.
3. Selected controller's **IP Address** will be updated in the **IP address(I)** field under the **Ethernet settings** option.
4. Click the **OK** button to update the connection settings and **Close** the **Comm Settings** window.

3.3 Installing CPX-FB37 ESI File in KV STUDIO

Step 1

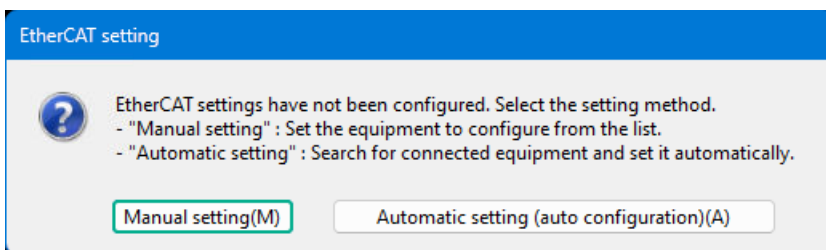
Download the **ESI** file of **CPX-FB37** module from **Festo Support Portal** and save it in an appropriate folder in the workstation.

Step 2

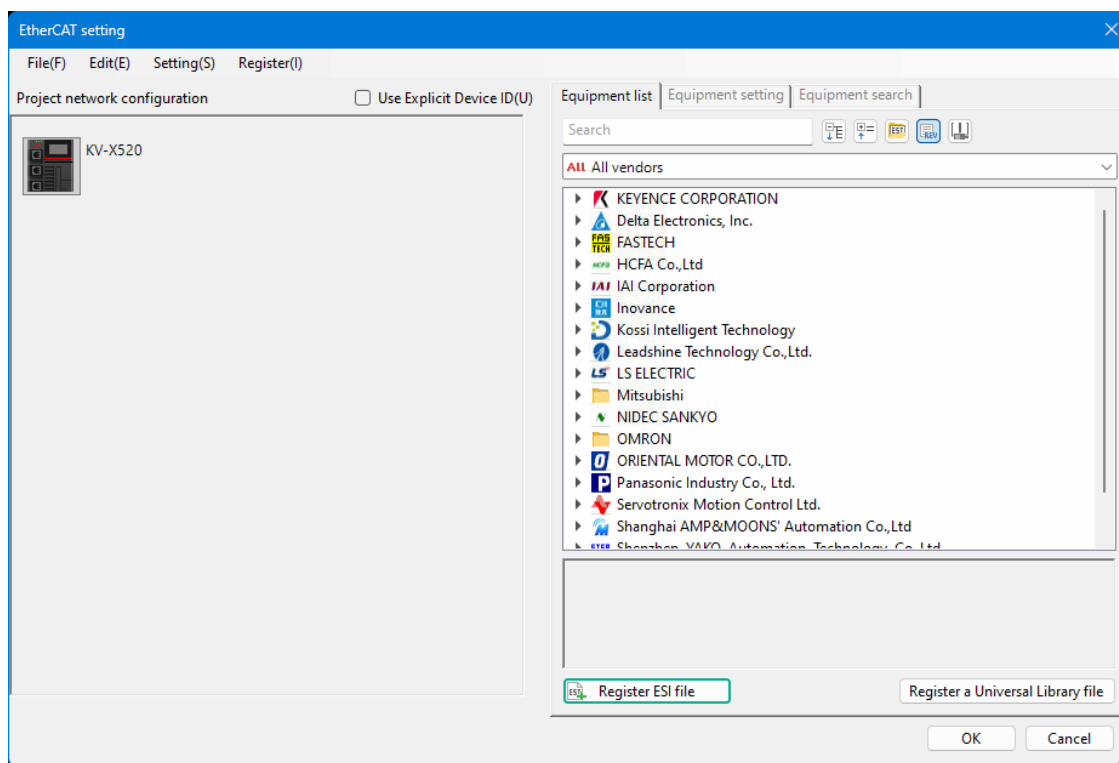


Double-click the **EtherCAT** option under **Unit Configuration** settings of the **[0] KV-X520** PLC as shown to open the **EtherCAT Setting** window.

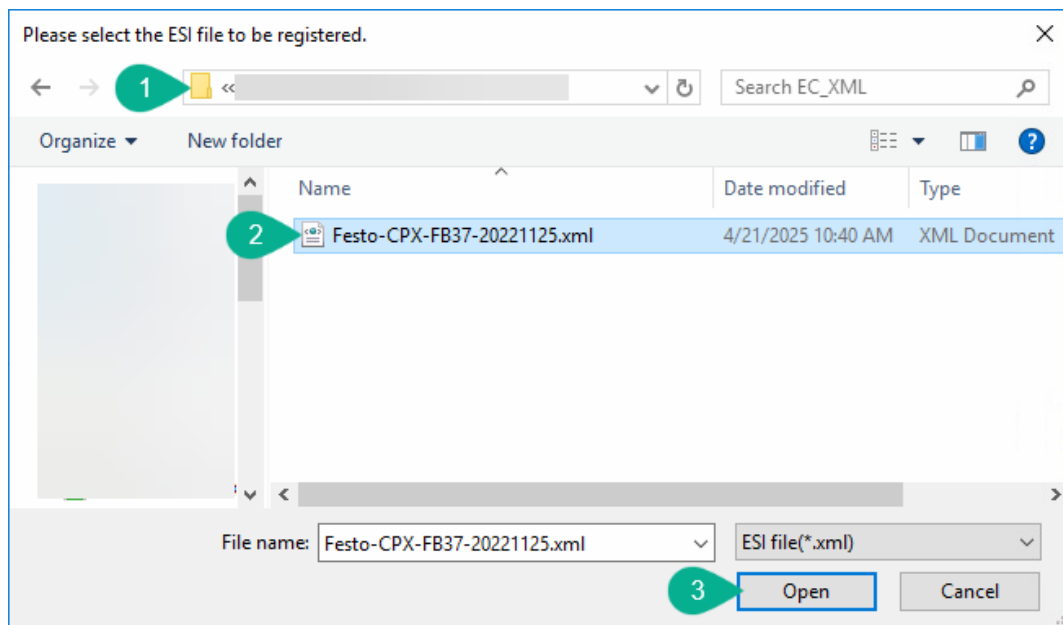
Step 3



Click the **Manual setting(M)** button to start the **CPX-FB37** ESI file installation.

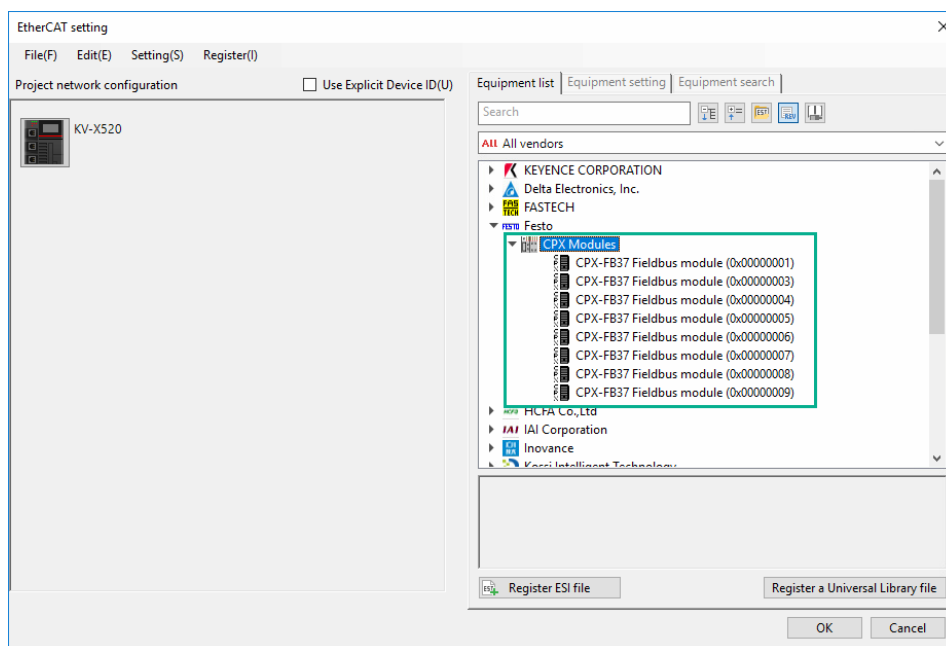
Step 4

Click the **Register ESI File** option from the **EtherCAT setting** window.

Step 5

1. Locate the downloaded **CPX-FB37 ESI** file in the workstation.
2. Select the **ESI** file from the location.
3. Click the **Open** button to install the selected ESI file.

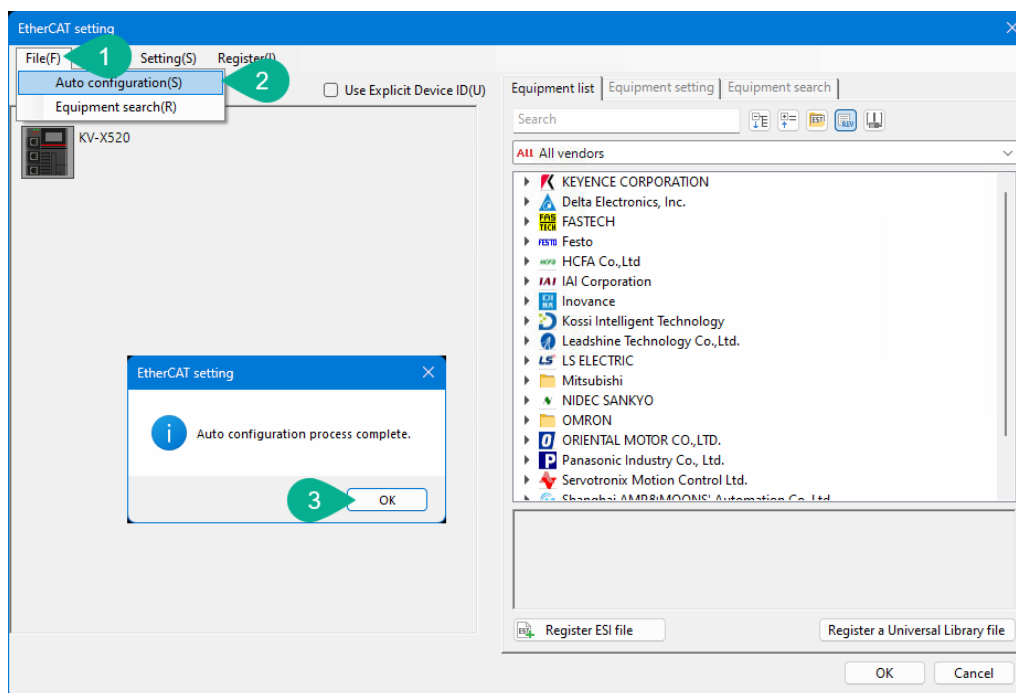
Step 6



The installed **CPX-FB37 ESI** file will be added to **Equipment list** as highlighted above.

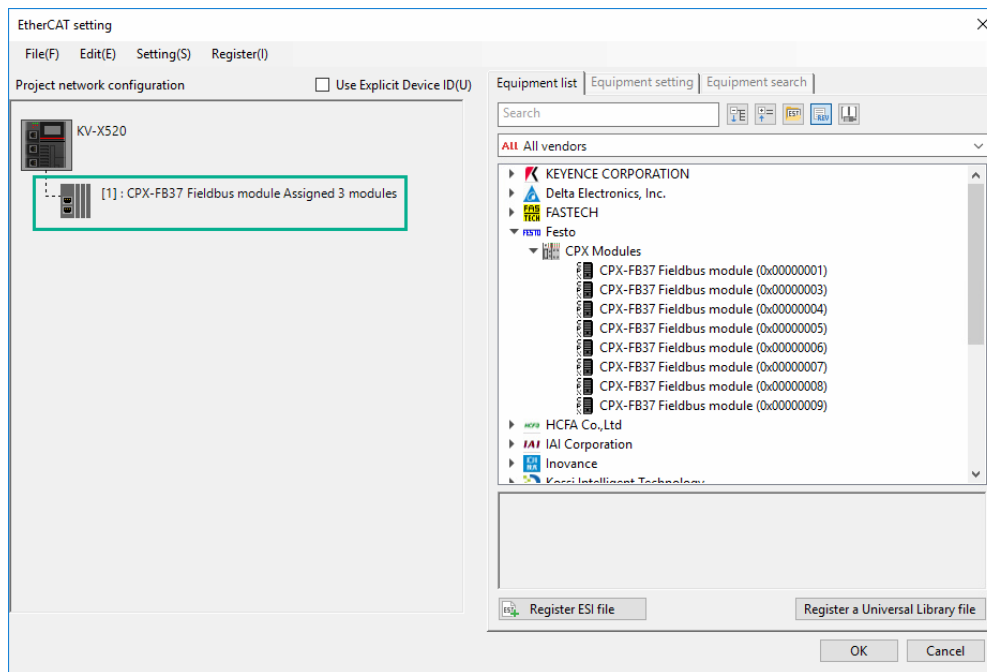
3.4 CPX-FB37 Module Configuration

3.4.1 Scanning the CPX-FB37 Device Connected to the EtherCAT Master



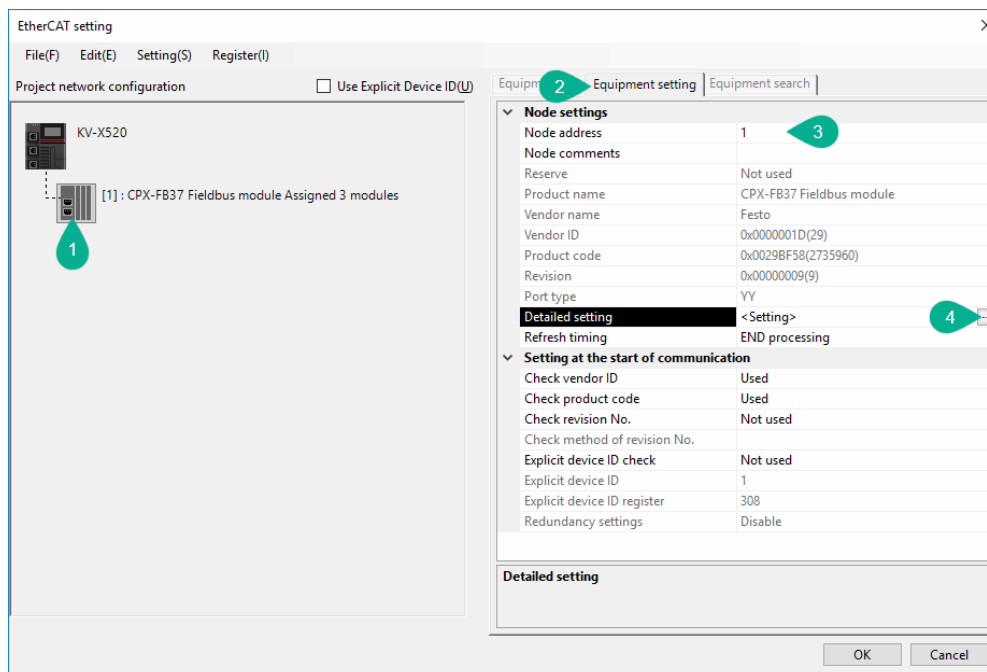
1. Click the **File(F)** menu option on the toolbar in the **EtherCAT Settings** window.
2. Click the **Auto configuration(S)** option from **File** menu.
3. Once scanning is complete, click the **OK** button.

The **CPX-FB37** device will be added to the **EtherCAT Master** as shown below.



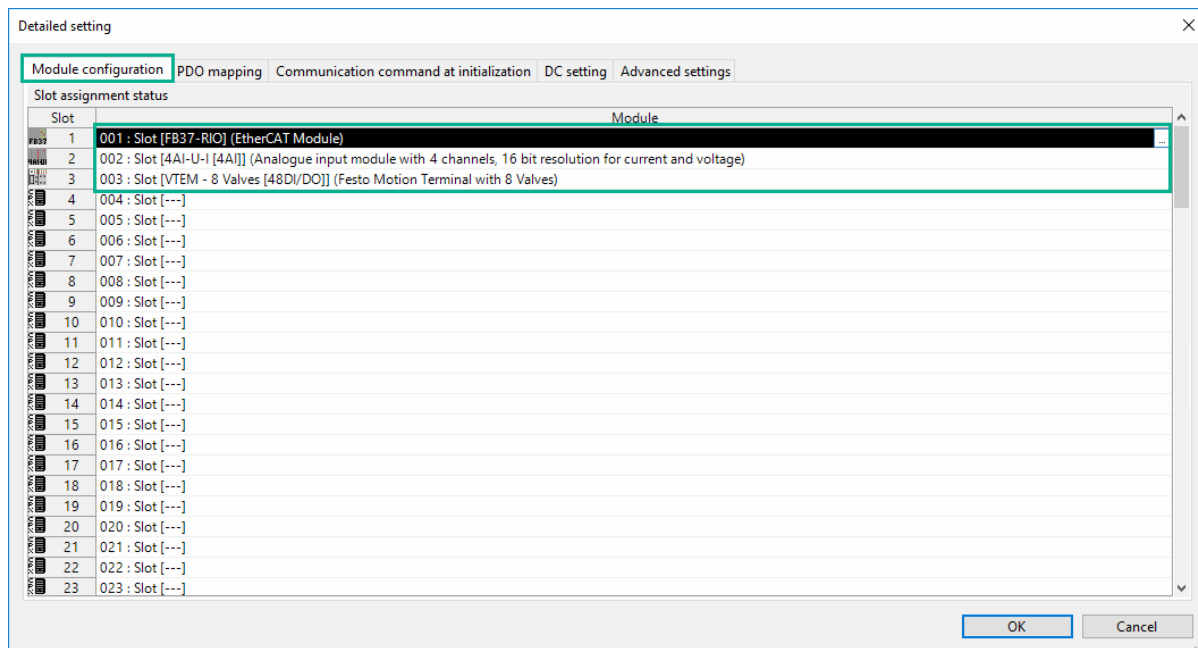
3.4.2 CPX-FB37 Node Settings

Step 1



1. Select the added **CPX-FB37** device in the **EtherCAT setting** window.
2. Click the **Equipment setting** tab for node settings.
3. Enter the required **Node Address** for the slave device as required. For this sample code, node address 1 is used.
4. Click the ... (**Browse**) button in the **Detailed setting** option from **Node settings** to open the **Detailed setting** window.

Step 2



The connected **CPX-FB37 Modules** will be shown in the **Module Configuration** tab of the **Detailed setting** window.

3.4.3 Creating Device Variables for CPX-FB37 Modules

Step 1

Detailed setting

Module configuration | PDO mapping | Communication command at initialization | DC setting | Advanced settings

Direction	PDO mapping name (index)	PDO entry name	Index	Data size (bit)	Data type	Buffer	Variable name
		(All areas)			(48 words)	Q0.0	CPX_Output
		▶ Valve 0 - Output 0	0x7002:01	8	UINT	Q0.0	
		▶ Valve 0 - Output 1	0x7002:02	8	UINT	Q1.0	
		▶ Valve 0 - Output 2	0x7002:03	8	UINT	Q2.0	
		▶ Valve 0 - Output 3	0x7002:04	8	UINT	Q3.0	
		▶ Valve 0 - Output 4	0x7002:05	8	UINT	Q4.0	
		▶ Valve 0 - Output 5	0x7002:06	8	UINT	Q5.0	
		▶ Valve 1 - Output 0	0x7002:07	8	UINT	Q6.0	
		▶ Valve 1 - Output 1	0x7002:08	8	UINT	Q7.0	
		▶ Valve 1 - Output 2	0x7002:09	8	UINT	Q8.0	
		▶ Valve 1 - Output 3	0x7002:0A	8	UINT	Q9.0	
		▶ Valve 1 - Output 4	0x7002:0B	8	UINT	Q10.0	
		▶ Valve 1 - Output 5	0x7002:0C	8	UINT	Q11.0	
		▶ Valve 2 - Output 0	0x7002:0D	8	UINT	Q12.0	
		▶ Valve 2 - Output 1	0x7002:0E	8	UINT	Q13.0	

Module 3 (VTEM - 8 Valves [48DI/DO]), Outputs (0x1602)

Detailed setting

Module configuration | PDO mapping | Communication command at initialization | DC setting | Advanced settings

Direction	PDO mapping name (index)	PDO entry name	Index	Data size (bit)	Data type	Buffer	Variable name
		▶ Valve 7 - Output 3	0x7002:2E	8	UINT	Q45.0	
		▶ Valve 7 - Output 4	0x7002:2F	8	UINT	Q46.0	
		▶ Valve 7 - Output 5	0x7002:30	8	UINT	Q47.0	
	(Add)						
		(All areas)			(52 words)	I0.0	CPX_Input
		▶ Input 0	0x6001:01	16	INT	I0.0	
		▶ Input 1	0x6001:02	16	INT	I1.0	
		▶ Input 2	0x6001:03	16	INT	I2.0	
		▶ Input 3	0x6001:04	16	INT	I3.0	
		▶ Valve 0 - Input 0	0x6002:01	8	UINT	I4.0	
		▶ Valve 0 - Input 1	0x6002:02	8	UINT	I5.0	
		▶ Valve 0 - Input 2	0x6002:03	8	UINT	I6.0	
		▶ Valve 0 - Input 3	0x6002:04	8	UINT	I7.0	
		▶ Valve 0 - Input 4	0x6002:05	8	UINT	I8.0	
		▶ Valve 0 - Input 5	0x6002:06	8	UINT	I9.0	
		▶ Valve 1 - Input 0	0x6002:07	8	UINT	I10.0	
		▶ Valve 1 - Input 1	0x6002:08	8	UINT	I11.0	
		▶ Valve 1 - Input 2	0x6002:09	8	UINT	I12.0	
		▶ Valve 1 - Input 3	0x6002:0A	8	UINT	I13.0	
		▶ Valve 1 - Input 4	0x6002:0B	8	UINT	I14.0	
		▶ Valve 1 - Input 5	0x6002:0C	8	UINT	I15.0	
		▶ Valve 2 - Input 0	0x6002:0D	8	UINT	I16.0	
		▶ Valve 2 - Input 1	0x6002:0E	8	UINT	I17.0	

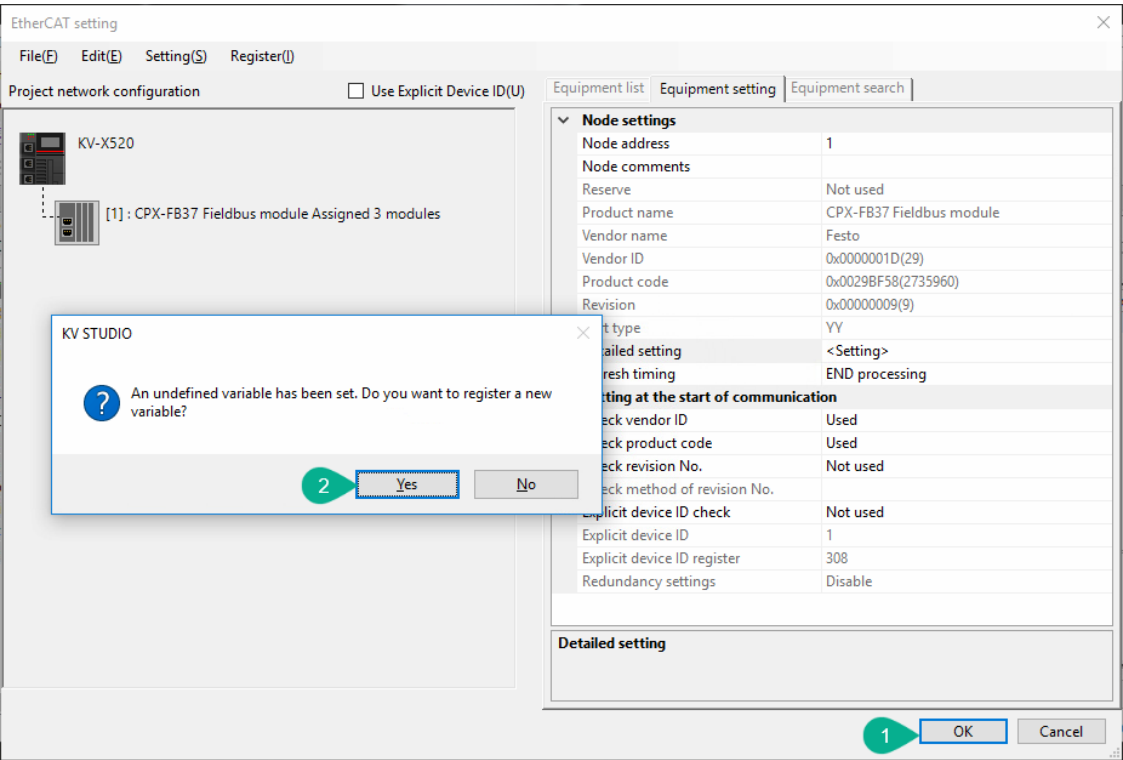
Module 2 (4AI-U-I [4AI]), Inputs (0x1A01)

Module 3 (VTEM - 8 Valves [48DI/DO]), Inputs (0x1A02)

OK Cancel

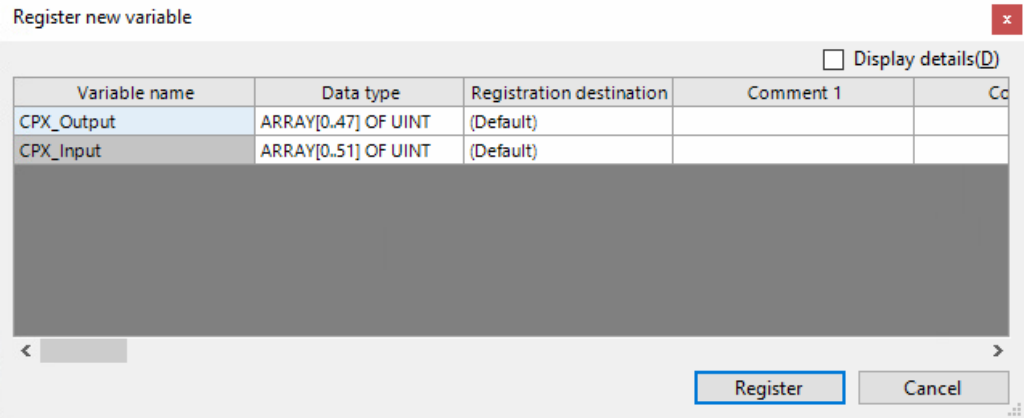
1. Enter the required **Variable Name** for the **Process Data Output** in the **All Areas** PDO entry, as shown.
2. Enter the required **Variable Name** for the **Process Data Input** in the **All Areas** PDO entry, as shown.
3. Click the **OK** button to add the variables for **CPX-FB37 Modules** and close the window.

Step 2



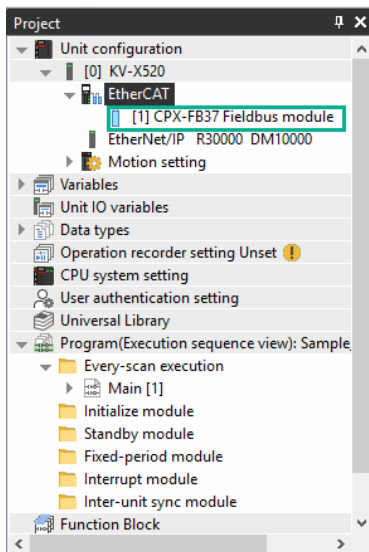
- 1. Click the **OK** button to close the **EtherCAT** window.
- 2. Click the **Yes** button to register the newly created, undefined device variable.

Step 3



Click the **Register** button in the **Register new variable** window to register the variables.

The added **CPX-FB37** module will be shown in the **Project** workspace as shown below.

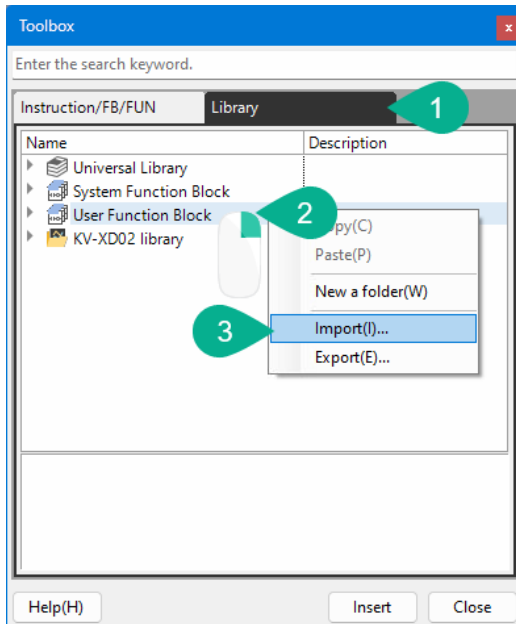


3.5 Adding Festo_VTEM_EC Library to a KV STUDIO Project

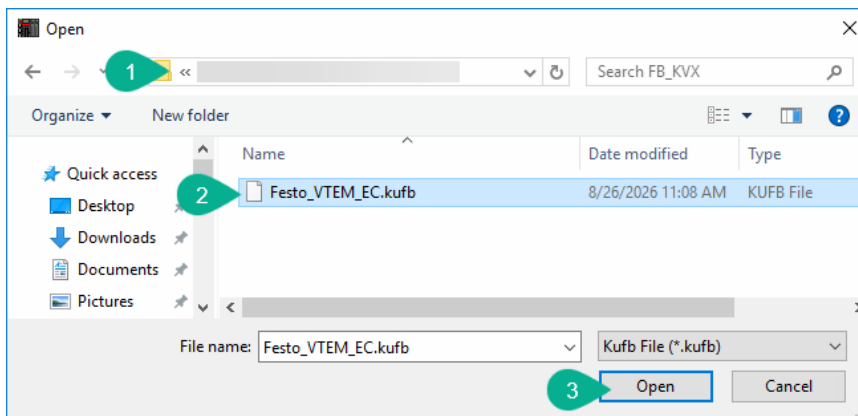
Step 1

Download the **Festo_VTEM_EC** library from the Festo Support Portal and save it in an appropriate folder on workstation.

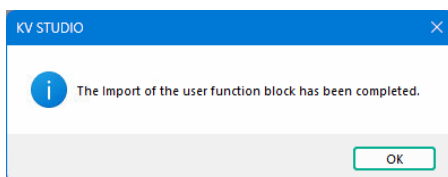
Step 2



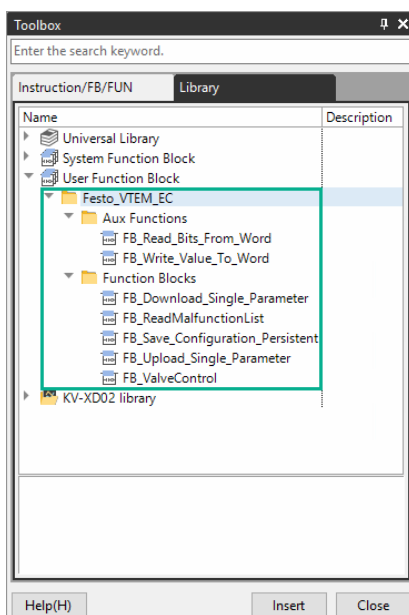
1. Select the **Library** tab in the **Toolbox** window.
2. Right-click the **User Function Block** option from **Library** toolbox.
3. Click the **Import(I)** option to import the library to the project.

Step 3

1. Locate the downloaded **Festo_VTEM_EC** library.
2. Select the **Library (Festo_VTEM_EC.kufb)** file from the location.
3. Click the **Open** button to install the selected library file.

Step 4

Click the **OK** button to finish the library import.

Step 5

After a successful import, **Festo_VTEM_EC** the library contents are displayed, as shown in the image above.

The **Festo_VTEM_EC** library contains several structures and function blocks for different purposes.

Function Blocks

- **FB_ValveControl** block is used to control a single valve in the Motion Terminal.
- **FB_Upload_Single_Parameter** block is used to read the Motion Terminal valve parameter.
- **FB_Download_Single_Parameter** block is used to write the Motion Terminal valve parameter.
- **FB_Save_Configuration** block is used to save the Motion Terminal valve parameter.
- **FB_ReadMalfunctionList** block is used to read the malfunction list of the VTEM Motion Terminal.

Structured Data Types

- stMalfunction
- stTimeStamp

The structured data types listed above are used in **FB_ReadMalfunctionList** block.



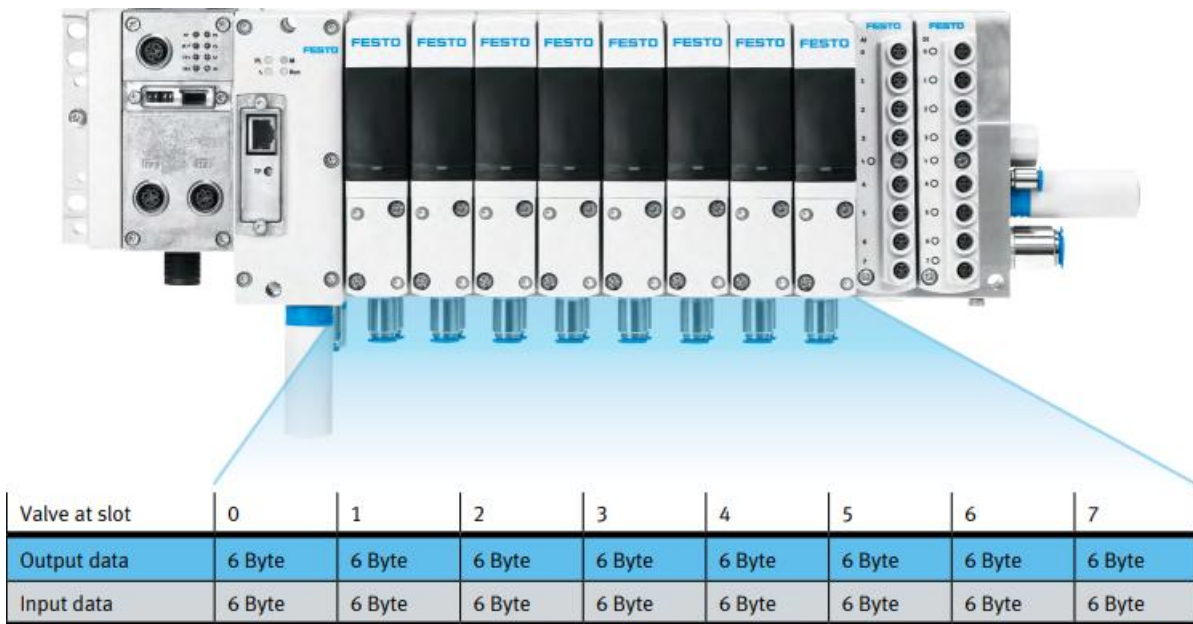
Note

- All user-defined libraries will be under User Function Block section.

4 VTEM Process Data details

4.1 VTEM Motion Terminal Communication Protocol

The communication between the higher-order controller (PLC) and the Motion Terminal controller uses 8×6 bytes each of input and output data from the CPX terminal. Each of the maximum 8 valve slots on a Motion Terminal are assigned 6 bytes of input data (Process Data Input, PDI) and 6 bytes of output data (Process Data Output, PDO), regardless of the actual number of valves present.



Information

The exact structure of the input and output data is described in detail in “Motion Terminal VTEM function and parameterization”.

The information in this description is assumed to be known.

4.1.1 Motion Terminal VTEM Function and Parameterization

The byte-level details of the input and output control data are provided below.

Output Data

Output data, valve at slot 2				Output data, valve at slot 3						Output data, valve at slot 4					
Byte 3	Byte 2	Byte 1	Byte 0	Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0	Byte 5	Byte 4	Byte 3	Byte 2		
Setpoint Value 1		Command		Setpoint Value 2		Setpoint Value 1		Command		Setpoint Value 2		Setpoint Value 1			
Byte 1								Byte 0							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
App Option								App Control		Valve Mode					

The operating mode of the valve is specified via the Valve Mode command:

Valve Mode	Meaning	IF	THEN
0	Reserved	–	–
1 ... 59	Start / run Motion App	Required parametrization has been completed.	Valve is inactive and configurable (Actual Valve Mode = 61, Valve State = 1 (configurable)). Setpoint/control values are valid. The requested Motion App has started.
1 ... 59	Start / run Motion App	Setpoint/control values are invalid.	See "Invalid output data" on page 21 of VTEM Quick Reference Application Note (Festo Motion Terminal VTEM - Quick Reference PLC programming).
1 ... 59	Start / run Motion App	Another Motion App or the Transfer Mode is still running.	Running Motion App continues to run, as indicated by the actual Valve Mode.
1 ... 59	Start / run Motion App	The device startup process has been completed, at least one inactive error but no active error is present (Valve Mode = 61, Valve State = 0 (not ready)).	Motion App cannot be started because of an inactive error that must first be acknowledged (send Valve Mode "62").
1 ... 59	Start / run Motion App	The device startup process has been completed, at least one active error is present (Valve Mode = 61, Valve State = 3 (failure)).	Motion App cannot start because of an active error that must first be corrected. After the error has been fixed, Valve State should change to 0 (not ready).
60	Start Teach-in run	See Valve Mode 1 ... 59	–
61	End running Motion App / Teach-in run	Motion App / Teach-in run is running	Motion App is stopped (Current Valve Mode = 61, Valve State = 1 (configurable)). Transfer Mode is running.
62	Acknowledge inactive errors and warnings	Valve State = 0 (not ready) because of an inactive error	Valve State changes to "configurable", malfunction list is cleared. Valve State = 1 (configurable). Warnings in malfunction list are deleted.
63	Transfer Mode	Valve is inactive (Actual Valve Mode = 61)	Transfer Mode command is executed.

Table 4.1: Valve Mode commands

Input Data

Input data, valve at slot 2				Input data, valve at slot 3						Input data, valve at slot 4					
Byte 3	Byte 2	Byte 1	Byte 0	Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0	Byte 5	Byte 4	Byte 3	Byte 2		
Actual Value 1		Status		Actual Value 2		Actual Value 1		Status		Actual Value 2		Actual Value 1			
Byte 1								Byte 0							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
App State								Valve State		Actual Valve Mode					

The current operating mode and state of the valve are shown in the Actual Valve Mode and Valve State fields of the input data:

Actual Valve Mode	Meaning
0	Reserved
1 ... 59	Running Motion App
60	Executing Teach-in run
61	Valve is inactive or performs self-calibration
63	Transfer Mode is active

Table 4.2: Valve mode meanings

Valve State		Meaning
0	Not ready	The Motion Terminal startup process is not complete or a detected and corrected error has not yet been acknowledged.
1	Configurable	The valve is inactive. A Motion App can be run or the valve can be switched to Transfer Mode.
2	Running	A Motion App is currently being executed.
3	Failure	An error has been detected but not yet eliminated. Starting/Running a Motion App is not possible.

Table 4.3: Valve state meanings

4.2 Connecting I/O Data to the Function Blocks

In the program that uses the function blocks of this library, the 6 bytes of input data and 6 bytes of output data must be maintained per valve as ARRAY [0..5] OF UINT. For EtherCAT communication with the KV-X, although the datatype is UINT, only the first eight bits are used to store each data byte. It is also recommended to create a separate instance of the function blocks used for each valve.

Module configuration PDO mapping Communication command at initialization DC setting Advanced settings									
Direction	PDO mapping name (index)	PDO entry name	Index	Data size (bit)	Data type	Buffer	Variable name	Variable name	Assignment destination
Module 2 (4AI-U-I [4AI]) Inputs (0x1A01)		Input 0	0x6001-01	16	INT	I0.0	Data from Analog Module	CPX_Input[0]	ETHERCAT[1]CPX-FB37 Fieldbus module 10.0
		Input 1	0x6001-02	16	INT	I1.0		CPX_Input[1]	ETHERCAT[1]CPX-FB37 Fieldbus module 11.0
		Input 2	0x6001-03	16	INT	I2.0		CPX_Input[2]	ETHERCAT[1]CPX-FB37 Fieldbus module 12.0
		Input 3	0x6001-04	16	INT	I3.0		CPX_Input[3]	ETHERCAT[1]CPX-FB37 Fieldbus module 13.0
		Valve 0 - Input 0	0x6002-01	8	UINT	I4.0		CPX_Input[4]	ETHERCAT[1]CPX-FB37 Fieldbus module 14.0
		Valve 0 - Input 1	0x6002-02	8	UINT	I5.0		CPX_Input[5]	ETHERCAT[1]CPX-FB37 Fieldbus module 15.0
		Valve 0 - Input 2	0x6002-03	8	UINT	I6.0		CPX_Input[6]	ETHERCAT[1]CPX-FB37 Fieldbus module 16.0
		Valve 0 - Input 3	0x6002-04	8	UINT	I7.0		CPX_Input[7]	ETHERCAT[1]CPX-FB37 Fieldbus module 17.0
		Valve 0 - Input 4	0x6002-05	8	UINT	I8.0		CPX_Input[8]	ETHERCAT[1]CPX-FB37 Fieldbus module 18.0
		Valve 0 - Input 5	0x6002-06	8	UINT	I9.0		CPX_Input[9]	ETHERCAT[1]CPX-FB37 Fieldbus module 19.0
		Valve 1 - Input 0	0x6002-07	8	UINT	I10.0		CPX_Input[10]	ETHERCAT[1]CPX-FB37 Fieldbus module 110.0
		Valve 1 - Input 1	0x6002-08	8	UINT	I11.0		CPX_Input[11]	ETHERCAT[1]CPX-FB37 Fieldbus module 111.0
		Valve 1 - Input 2	0x6002-09	8	UINT	I12.0		CPX_Input[12]	ETHERCAT[1]CPX-FB37 Fieldbus module 112.0
		Valve 1 - Input 3	0x6002-0A	8	UINT	I13.0		CPX_Input[13]	ETHERCAT[1]CPX-FB37 Fieldbus module 113.0
		Valve 1 - Input 4	0x6002-0B	8	UINT	I14.0		CPX_Input[14]	ETHERCAT[1]CPX-FB37 Fieldbus module 114.0
		Valve 1 - Input 5	0x6002-0C	8	UINT	I15.0		CPX_Input[15]	ETHERCAT[1]CPX-FB37 Fieldbus module 115.0
		Valve 2 - Input 0	0x6002-0D	8	UINT	I16.0		CPX_Input[16]	ETHERCAT[1]CPX-FB37 Fieldbus module 116.0
		Valve 2 - Input 1	0x6002-0E	8	UINT	I17.0		CPX_Input[17]	ETHERCAT[1]CPX-FB37 Fieldbus module 117.0
		Valve 2 - Input 2	0x6002-0F	8	UINT	I18.0		CPX_Input[18]	ETHERCAT[1]CPX-FB37 Fieldbus module 118.0
		Valve 2 - Input 3	0x6002-10	8	UINT	I19.0		CPX_Input[19]	ETHERCAT[1]CPX-FB37 Fieldbus module 119.0
		Valve 2 - Input 4	0x6002-11	8	UINT	I20.0		CPX_Input[20]	ETHERCAT[1]CPX-FB37 Fieldbus module 120.0
		Valve 2 - Input 5	0x6002-12	8	UINT	I21.0		CPX_Input[21]	ETHERCAT[1]CPX-FB37 Fieldbus module 121.0
		Valve 3 - Input 0	0x6002-13	8	UINT	I22.0		CPX_Input[22]	ETHERCAT[1]CPX-FB37 Fieldbus module 122.0
		Valve 3 - Input 1	0x6002-14	8	UINT	I23.0		CPX_Input[23]	ETHERCAT[1]CPX-FB37 Fieldbus module 123.0
		Valve 3 - Input 2	0x6002-15	8	UINT	I24.0		CPX_Input[24]	ETHERCAT[1]CPX-FB37 Fieldbus module 124.0
		Valve 3 - Input 3	0x6002-16	8	UINT	I25.0		CPX_Input[25]	ETHERCAT[1]CPX-FB37 Fieldbus module 125.0
		Valve 3 - Input 4	0x6002-17	8	UINT	I26.0		CPX_Input[26]	ETHERCAT[1]CPX-FB37 Fieldbus module 126.0
		Valve 3 - Input 5	0x6002-18	8	UINT	I27.0		CPX_Input[27]	ETHERCAT[1]CPX-FB37 Fieldbus module 127.0
		Valve 4 - Input 0	0x6002-19	8	UINT	I28.0		CPX_Input[28]	ETHERCAT[1]CPX-FB37 Fieldbus module 128.0
		Valve 4 - Input 1	0x6002-1A	8	UINT	I29.0		CPX_Input[29]	ETHERCAT[1]CPX-FB37 Fieldbus module 129.0
		Valve 4 - Input 2	0x6002-1B	8	UINT	I30.0		CPX_Input[30]	ETHERCAT[1]CPX-FB37 Fieldbus module 130.0
		Valve 4 - Input 3	0x6002-1C	8	UINT	I31.0		CPX_Input[31]	ETHERCAT[1]CPX-FB37 Fieldbus module 131.0
		Valve 4 - Input 4	0x6002-1D	8	UINT	I32.0		CPX_Input[32]	ETHERCAT[1]CPX-FB37 Fieldbus module 132.0
		Valve 4 - Input 5	0x6002-1E	8	UINT	I33.0		CPX_Input[33]	ETHERCAT[1]CPX-FB37 Fieldbus module 133.0
		Valve 5 - Input 0	0x6002-1F	8	UINT	I34.0		CPX_Input[34]	ETHERCAT[1]CPX-FB37 Fieldbus module 134.0
Module 3 (VTEM - 8 Valves [48D/DO]) Inputs (0x1A02)		Valve 5 - Input 1	0x6002-20	8	UINT	I35.0	Data from Valve 0	CPX_Input[35]	ETHERCAT[1]CPX-FB37 Fieldbus module 135.0
		Valve 5 - Input 2	0x6002-21	8	UINT	I36.0		CPX_Input[36]	ETHERCAT[1]CPX-FB37 Fieldbus module 136.0
		Valve 5 - Input 3	0x6002-22	8	UINT	I37.0		CPX_Input[37]	ETHERCAT[1]CPX-FB37 Fieldbus module 137.0
		Valve 5 - Input 4	0x6002-23	8	UINT	I38.0		CPX_Input[38]	ETHERCAT[1]CPX-FB37 Fieldbus module 138.0
		Valve 5 - Input 5	0x6002-24	8	UINT	I39.0		CPX_Input[39]	ETHERCAT[1]CPX-FB37 Fieldbus module 139.0
		Valve 6 - Input 0	0x6002-25	8	UINT	I40.0		CPX_Input[40]	ETHERCAT[1]CPX-FB37 Fieldbus module 140.0
		Valve 6 - Input 1	0x6002-26	8	UINT	I41.0		CPX_Input[41]	ETHERCAT[1]CPX-FB37 Fieldbus module 141.0
		Valve 6 - Input 2	0x6002-27	8	UINT	I42.0		CPX_Input[42]	ETHERCAT[1]CPX-FB37 Fieldbus module 142.0
		Valve 6 - Input 3	0x6002-28	8	UINT	I43.0		CPX_Input[43]	ETHERCAT[1]CPX-FB37 Fieldbus module 143.0
		Valve 6 - Input 4	0x6002-29	8	UINT	I44.0		CPX_Input[44]	ETHERCAT[1]CPX-FB37 Fieldbus module 144.0
		Valve 6 - Input 5	0x6002-2A	8	UINT	I45.0		CPX_Input[45]	ETHERCAT[1]CPX-FB37 Fieldbus module 145.0
		Valve 7 - Input 0	0x6002-2B	8	UINT	I46.0		CPX_Input[46]	ETHERCAT[1]CPX-FB37 Fieldbus module 146.0
		Valve 7 - Input 1	0x6002-2C	8	UINT	I47.0		CPX_Input[47]	ETHERCAT[1]CPX-FB37 Fieldbus module 147.0
		Valve 7 - Input 2	0x6002-2D	8	UINT	I48.0		CPX_Input[48]	ETHERCAT[1]CPX-FB37 Fieldbus module 148.0
		Valve 7 - Input 3	0x6002-2E	8	UINT	I49.0		CPX_Input[49]	ETHERCAT[1]CPX-FB37 Fieldbus module 149.0
		Valve 7 - Input 4	0x6002-2F	8	UINT	I50.0		CPX_Input[50]	ETHERCAT[1]CPX-FB37 Fieldbus module 150.0
		Valve 7 - Input 5	0x6002-30	8	UINT	I51.0		CPX_Input[51]	ETHERCAT[1]CPX-FB37 Fieldbus module 151.0
		(Add)							
Module 3 (VTEM - 8 Valves [48D/DO]) Outputs (0x1A02)		(All areas)			(48 words)	Q0.0	Data to Valve 0	CPX_Output	ETHERCAT[1]CPX-FB37 Fieldbus module Q0.0
		Valve 0 - Output 0	0x7002-01	8	UINT	Q0.0		CPX_Output[0]	ETHERCAT[1]CPX-FB37 Fieldbus module Q0.0
		Valve 0 - Output 1	0x7002-02	8	UINT	Q1.0		CPX_Output[1]	ETHERCAT[1]CPX-FB37 Fieldbus module Q1.0
		Valve 0 - Output 2	0x7002-03	8	UINT	Q2.0		CPX_Output[2]	ETHERCAT[1]CPX-FB37 Fieldbus module Q2.0
		Valve 0 - Output 3	0x7002-04	8	UINT	Q3.0		CPX_Output[3]	ETHERCAT[1]CPX-FB37 Fieldbus module Q3.0
		Valve 0 - Output 4	0x7002-05	8	UINT	Q4.0		CPX_Output[4]	ETHERCAT[1]CPX-FB37 Fieldbus module Q4.0
		Valve 0 - Output 5	0x7002-06	8	UINT	Q5.0		CPX_Output[5]	ETHERCAT[1]CPX-FB37 Fieldbus module Q5.0
		Valve 1 - Output 0	0x7002-07	8	UINT	Q6.0		CPX_Output[6]	ETHERCAT[1]CPX-FB37 Fieldbus module Q6.0
		Valve 1 - Output 1	0x7002-08	8	UINT	Q7.0		CPX_Output[7]	ETHERCAT[1]CPX-FB37 Fieldbus module Q7.0
		Valve 1 - Output 2	0x7002-09	8	UINT	Q8.0		CPX_Output[8]	ETHERCAT[1]CPX-FB37 Fieldbus module Q8.0
		Valve 1 - Output 3	0x7002-0A	8	UINT	Q9.0		CPX_Output[9]	ETHERCAT[1]CPX-FB37 Fieldbus module Q9.0
		Valve 1 - Output 4	0x7002-0B	8	UINT	Q10.0		CPX_Output[10]	ETHERCAT[1]CPX-FB37 Fieldbus module Q10.0
		Valve 1 - Output 5	0x7002-0C	8	UINT	Q11.0		CPX_Output[11]	ETHERCAT[1]CPX-FB37 Fieldbus module Q11.0
		Valve 2 - Output 0	0x7002-0D	8	UINT	Q12.0		CPX_Output[12]	ETHERCAT[1]CPX-FB37 Fieldbus module Q12.0
		Valve 2 - Output 1	0x7002-0E	8	UINT	Q13.0		CPX_Output[13]	ETHERCAT[1]CPX-FB37 Fieldbus module Q13.0
		Valve 2 - Output 2	0x7002-0F	8	UINT	Q14.0		CPX_Output[14]	ETHERCAT[1]CPX-FB37 Fieldbus module Q14.0
		Valve 2 - Output 3	0x7002-10	8	UINT	Q15.0		CPX_Output[15]	ETHERCAT[1]CPX-FB37 Fieldbus module Q15.0
		Valve 2 - Output 4	0x7002-11	8	UINT	Q16.0		CPX_Output[16]	ETHERCAT[1]CPX-FB37 Fieldbus module Q16.0
		Valve 2 - Output 5	0x7002-12	8	UINT	Q17.0		CPX_Output[17]	ETHERCAT[1]CPX-FB37 Fieldbus module Q17.0
		Valve 3 - Output 0	0x7002-13	8	UINT	Q18.0		CPX_Output[18]	ETHERCAT[1]CPX-FB37 Fieldbus module Q18.0
		Valve 3 - Output 1	0x7002-14	8	UINT	Q19.0		CPX_Output[19]	ETHERCAT[1]CPX-FB37 Fieldbus module Q19.0
		Valve 3 - Output 2	0x7002-15	8	UINT	Q20.0		CPX_Output[20]	ETHERCAT[1]CPX-FB37 Fieldbus module Q20.0
		Valve 3 - Output 3	0x7002-16	8	UINT	Q21.0		CPX_Output[21]	ETHERCAT[1]CPX-FB37 Fieldbus module Q21.0
		Valve 3 - Output 4	0x7002-17	8	UINT	Q22.0		CPX_Output[22]	ETHERCAT[1]CPX-FB37 Fieldbus module Q22.0
		Valve 3 - Output 5	0x7002-18	8	UINT	Q23.0		CPX_Output[23]	ETHERCAT[1]CPX-FB37 Fieldbus module Q23.0
		Valve 4 - Output 0	0x7002-19	8	UINT	Q24.0		CPX_Output[24]	ETHERCAT[1]CPX-FB37 Fieldbus module Q24.0
		Valve 4 - Output 1	0x7002-1A	8	UINT	Q25.0		CPX_Output[25]	ETHERCAT[1]CPX-FB37 Fieldbus module Q25.0
		Valve 4 - Output 2	0x7002-1B	8	UINT	Q26.0		CPX_Output[26]	ETHERCAT[1]CPX-FB37 Fieldbus module Q26.0
		Valve 4 - Output 3	0x7002-1C	8	UINT	Q27.0		CPX_Output[27]	ETHERCAT[1]CPX-FB37 Fieldbus module Q27.0
		Valve 4 - Output 4	0x7002-1D	8	UINT	Q28.0		CPX_Output[28]	ETHERCAT[1]CPX-FB37 Fieldbus module Q28.0
		Valve 4 - Output 5	0x7002-1E	8	UINT	Q29.0		CPX_Output[29]	ETHERCAT[1]CPX-FB37 Fieldbus module Q29.0
		Valve 5 - Output 0	0x7002-1F	8	UINT	Q30.0		CPX_Output[30]	ETHERCAT[1]CPX-FB37 Fieldbus module Q30.0
		Valve 5 - Output 1	0x7002-20	8	UINT	Q31.0		CPX_Output[31]	ETHERCAT[1]CPX-FB37 Fieldbus module Q31.0
		Valve 5 - Output 2	0x7002-21	8	UINT	Q32.0		CPX_Output[32]	ETHERCAT[1]CPX-FB37 Fieldbus module Q32.0
		Valve 5 - Output 3	0x7002-22	8	UINT	Q33.0		CPX_Output[33]	ETHERCAT[1]CPX-FB37 Fieldbus module Q33.0
		Valve 5 - Output 4	0x7002-23	8	UINT	Q34.0		CPX_Output[34]	ETHERCAT[1]CPX-FB37 Fieldbus module Q34.0
		Valve 5 - Output 5	0x7002-24	8	UINT	Q35.0		CPX_Output[35]	ETHERCAT[1]CPX-FB37 Fieldbus module Q35.0
		Valve 6 - Output 0	0x7002-25	8	UINT	Q36.0		CPX_Output[36]	ETHERCAT[1]CPX-FB37 Fieldbus module Q36.0
		Valve 6 - Output 1	0x7002-26	8	UINT	Q37.0		CPX_Output[37]	ETHERCAT[1]CPX-FB37 Fieldbus module Q37.0
		Valve 6 - Output 2	0x7002-27	8	UINT	Q38.0		CPX_Output[38]	ETHERCAT[1]CPX-FB37 Fieldbus module Q38.0
		Valve 6 - Output 3	0x7002-28	8	UINT	Q39.0		CPX_Output[39]	ETHERCAT[1]CPX-FB37 Fieldbus module Q39.0
		Valve 6 - Output 4	0x7002-29	8	UINT	Q40.0		CPX_Output[40]	ETHERCAT[1]CPX-FB37 Fieldbus module Q40.0
		Valve 6 - Output 5	0x7002-2A	8	UINT	Q41.0		CPX_Output[41]	ETHERCAT[1]CPX-FB37 Fieldbus module Q41.0
		Valve 7 - Output 0	0x7002-2B	8	UINT	Q42.0		CPX_Output[42]	ETHERCAT[1]CPX-FB37 Fieldbus module Q42.0
		Valve 7 - Output 1	0x7002-2C	8	UINT	Q43.0		CPX_Output[43]	ETHERCAT[1]CPX-FB37 Fieldbus module Q43.0
		Valve 7 - Output 2	0x7002-2D	8	UINT	Q44.0		CPX_Output[44]	ETHERCAT[1]CPX-FB37 Fieldbus module Q44.0
		Valve 7 - Output 3	0x7002-2E	8	UINT	Q45.0		CPX_Output[45]	ETHERCAT[1]CPX-FB37 Fieldbus module Q45.0
		Valve 7 - Output 4	0x7002-2F	8	UINT	Q46.0		CPX_Output[46]	ETHERCAT[1]CPX-FB37 Fieldbus module Q46.0
		Valve 7 - Output 5	0x7002-30	8	UINT	Q47.0		CPX_Output[47]	ETHERCAT[1]CPX-FB37 Fieldbus module Q47.0
		(Add)							

The input and output data of a valve are transferred to the function blocks as ARRAY[0..5] OF UINT.

5 Festo_VTEM_EC Library Function Block Descriptions

5.1 FB_ValveControl

Function block **FB_ValveControl** is used to operate and control Motion Apps. For this purpose, the function block interprets the values at the inputs and generates the output data that is sent to the Motion Terminal.

At the same time, the function block interprets the input data of a valve received from the Motion Terminal, classifies the information and provides it to the appropriate outputs.

This means that the function block outputs are updated at least one cycle after the input values change.

FB_ValveControl	
EN	ENO
auiBusDataFromVTEM	auiBusDataToVTEM
xEnable	uiActualValveMode
uiSetValveMode	uiActualValveState
uiSetAppControl	uiActualAppState
uiSetAppOption	iActualValue1
iSetPointValue1	iActualValue2
iSetPointValue2	iResponseToValveModeSet
xManualAcknowledge	iErrorCode

5.1.1 FB_ValveControl Block Input Variable

Input Variables	Data Type	Description
aiBusDataFromVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of input data received from the Motion Terminal must be applied to this input.
xEnable	BOOL	Input must be TRUE to be able to use the function block. When this input changes to FALSE, the function block stops any operation running on the valve (Valve Mode = “valve inactive”). The output iResponseToValveModeSet shows the value 1051 after the function block has stopped successfully.
uiSetValveMode	UINT	Input for setting the valve mode. Permissible values are: 1 ... 59 (ID Motion App), 60 (Teach-in run), 61 (End Motion App), 62 (Acknowledge inactive error), alternatively, use input “xManualAcknowledge”
uiSetAppControl	UINT	Input for setting the Motion App control. The meaning is specific to the Motion App.
uiSetAppOption	UINT	Input for setting the Motion App option. The meaning is specific to the Motion App.
iSetPointValue1	INT	Input for setpoint value 1: The meaning is specific to the Motion App.
iSetPointValue2	INT	Input for setpoint value 2: The meaning is specific to the Motion App.
xManualAcknowledge	BOOL	Inactive errors when the valve status is ‘not ready’ can be acknowledged with a rising edge at this input (corresponds to setting “uiSetValveMode := 62” --> Valve status becomes “configurable”).

Table 5.1: FB_ValveControl Block Input Variable descriptions

5.1.2 FB_ValveControl Block Output Variable

Output Variables	Data Type	Description
aiBusDataToVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of output data that must be sent to the Motion Terminal are available at this output.
uiActualValveMode	UINT	Returned mode of the valve
uiActualValveState	UINT	Returned status of the valve
uiActualAppState	UINT	Returned status of the Motion App
iActualValue1	INT	Returned actual value 1
iActualValue2	INT	Returned actual value 2
iResponseToValveModeSet	INT	Response to the sent Valve Mode (Input uiSetValveMode). Refer Chapter 5.1.3.
iErrorCode	UINT	If an error occurs, this output contains the most recent error code.

Table 5.2: FB_ValveControl Block Output Variable descriptions

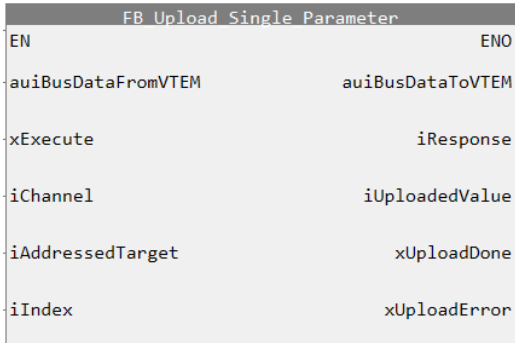
5.1.3 Response to Valve Mode Set

Status Value	Status Description
0	Reserved
1	Invalid Set Valve Mode
2	Invalid Set App Control
3	Invalid Set App Option
4	Invalid Set Point Value 1
5	Invalid Set Point Value 2
6	Invalid Configuration
10	Access Denied Web Configuration
11	Access Denied Saving Parametrization
12	No License for this Motion App
13	All Licenses in use
14	Invalid License file
15	No valve detected
16	Valve self-calibration running
1020	No Transfer Mode with this FB
1031	Preparing for start
1032	Starting
1033	Motion App Running
1034	Transfer Mode Running
1035	Stopping
1036	All Motion Apps stopped
1037	Acknowledge needed
1038	Failure
1040	Preparing for acknowledge
1041	Acknowledging
1042	All errors acknowledged
1043	Failure still active
1050	No communication
1051	FB not enabled

Table 5.3: Response to Valve Mode Set descriptions

5.2 FB_Upload_Single_Parameter

The function block transmits a value via the transfer mode from the Motion Terminal to the PLC.



5.2.1 FB_Upload_Single_Parameter Block Input Variable

Input Variables	Data Type	Description
auibusDataFromVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of input data received from the Motion Terminal must be applied to this input.
xExecute	BOOL	Rising edge at the input starts the transmission. If the “TRUE” signal is removed before transmission is complete (xUploadDone = TRUE), the transfer mode is terminated.
iChannel	INT	Transfer mode channel
iAddressedTarget	INT	Channel-specific address of the target from which the value is to be read.
iIndex	INT	Index of the stored value to be read.

Table 5.4: FB_Upload_Single_Parameter Block Input Variable descriptions

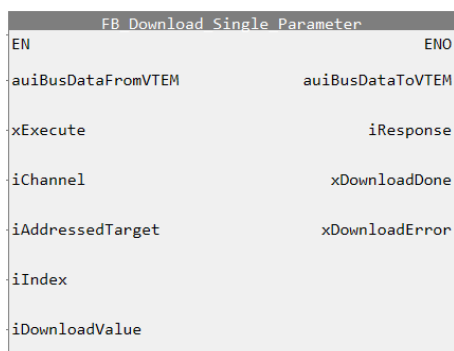
5.2.2 FB_Upload_Single_Parameter Block Output Variable

Output Variables	Data Type	Description
auibusDataToVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of output data that must be sent to the Motion Terminal are available at this output.
iResponse	INT	Response to the status of the transmission. Refer Chapter 5.3.3.
iUploadedValue	INT	Value read from the Motion Terminal.
xUploadDone	BOOL	Output becomes TRUE when the transfer has completed without an error.
xUploadError	BOOL	Output becomes TRUE when errors occur during the transfer.

Table 5.5: FB_Upload_Single_Parameter Block Output Variable descriptions

5.3 FB_Download_Single_Parameter

The function block transmits a value via the transfer mode from the PLC to the Motion Terminal. It can only be used for writable values.



5.3.1 FB_Download_Single_Parameter Block Input Variable

Input Variables	Data Type	Description
auibusDataFromVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of input data received from the Motion Terminal must be applied to this input.
xExecute	BOOL	Rising edge at the input starts the transmission. If the “TRUE” signal is removed before transmission is completed (xDownloadDone = TRUE), the transfer mode is terminated.
iChannel	INT	Transfer mode channel
iAddressedTarget	INT	Channel-specific address of the target to which the value to be transmitted is to be written.
iIndex	INT	Index of the parameter to which the transmitted value is to be written.
iDownloadValue	INT	Value to be written to the parameter.

Table 5.6: FB_Download_Single_Parameter Block Input Variable descriptions

5.3.2 FB_Download_Single_Parameter Block Output Variable

Output Variables	Data Type	Description
auibusDataToVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of output data that must be sent to the Motion Terminal are available at this output.
iResponse	INT	Response to the status of the transmission. Refer Chapter 5.3.3.
xDownloadDone	BOOL	Output becomes TRUE when the transfer has completed without an error.
xDownloadError	BOOL	Output becomes TRUE when errors occur during the transfer.

Table 5.7: FB_Download_Single_Parameter Block Output Variable descriptions

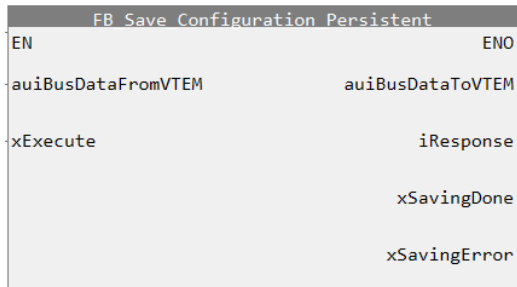
5.3.3 Response to Transfer Mode

Status Value	Status Description
0	Reserved
1	Invalid Channel
2	Invalid Transfer Control
3	Invalid Addressed Target
4	Invalid Index
5	Invalid Transfer Value
6	Invalid Combination
7	Access Denied: Data in use
8	No Data, Empty slot
9	Invalid Parameter Set
10	Access Denied: Web Configuration
11	Access Denied: Saving Parametrization
15	No valve detected
20	Reading Transfer Mode
21	Ready for new transfer
22	Preparing transfer
23	Transferring
24	Transfer Successful
25	FB needs Rising edge of xExecute
26	Valve mode is not 61; Inactive
27	Invalid Firmware version
28	Communication failed
29	Invalid Module Number
30	Resetting
31	Aborting

Table 5.8: Response to Transfer Mode descriptions

5.4 FB_Save_Configuration

The function block provides a simple way to call the function “Save configuration persistent”. The configuration data for the valve currently being parameterized is saved permanently on the controller of the Motion Terminal.



5.4.1 FB_Save_Configuration Block Input Variable

Input Variables	Data Type	Description
auiBusDataFromVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of input data received from the Motion Terminal must be applied to this input.
xExecute	BOOL	Execute saving of configuration in VTEM.

Table 5.9: FB_Save_Configuration Block Input Variable descriptions

5.4.2 FB_Save_Configuration Block Output Variable

Output Variables	Data Type	Description
auiBusDataToVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of output data that must be sent to the Motion Terminal are available at this output.
iResponse	INT	Response to the status of saving process. Refer Chapter 5.4.3.
xSavingDone	BOOL	The output becomes “TRUE” when the saving process is completed.
xSavingError	BOOL	The output becomes “TRUE” when an error occurs during the saving process.

Table 5.10: FB_Save_Configuration Block Output Variable descriptions

5.4.3 Response to Save Persistent

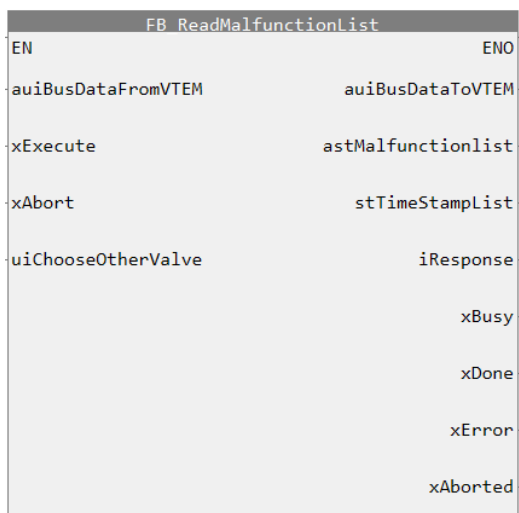
Status Value	Status Description
0	Reserved
1	Saving in progress
2	Saving Successful
3	Saving not possible
4	Saving failed
10	Access Denied: Web Configuration
11	Access Denied: Saving Parametrization
15	No valve detected
20	Preparing to save
21	Ready to save
25	FB needs Rising edge of xExecute

Table 5.11: Response to Save Persistent descriptions

5.5 FB_ReadMalfunctionList

The function block “**FB_ReadMalfunctionList**” automatically reads the contents of the malfunction list of a valve and passes it to the output “**stMalfunctionlist**”. Refer Malfunction Codes document for the details.

The input “**uiChooseOtherValve**” specifies the valve from which the malfunction list is read. In this way, the process data of any other slot can be used for this purpose.



5.5.1 FB_ReadMalfunctionList Block Input Variable

Input Variables	Data Type	Description
auiBusDataFromVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of input data received from the Motion Terminal must be applied to this input.
xExecute	BOOL	TRUE signal at the input starts the transmission. If the TRUE signal is removed, the transfer mode is terminated. The function block is reset once the transfer process has been completed successfully (iResponse = “4” or “5”) or with an error (xError = TRUE).
xAbort	BOOL	TRUE signal at the input terminates the process in the function block (iResponse = “7”). The transfer mode is terminated if necessary and the valve is set to the “inactive” state (valve mode = 61).
uiChooseOtherValve	UINT	Select the valve whose malfunction list is to be read. 0: Read the malfunction list of the valve whose process data the function block uses. 100 ... 107: Read the malfunction list of the valve in slot 0 (100) ... 7 (107).

Table 5.12: FB_ReadMalfunctionList Block Input Variable descriptions

5.5.2 FB_ReadMalfunctionList Block Output Variable

Output Variables	Data Type	Description
auiBusDataToVTEM	ARRAY[0..5] OF UINT	The 6 Bytes of output data that must be sent to the Motion Terminal are available at this output.
astMalfunctionlist	ARRAY[1..40] OF stMalfunction	Contents of the malfunction list (The array contents are complete as soon as xDone = TRUE. The output is reset as soon as xAbort = FALSE and xExecute= FALSE). Refer Chapter 5.5.4.
stTimeStampList	STRUCT of the type stTimeStamp	The output shows the current timestamp of the VTEM Motion Terminal. Refer Chapter 5.5.5.
iResponse	INT	Response to the status of the transmission. Refer Chapter 5.5.3.
xBusy	BOOL	Output is TRUE as long as the function block executes an action.
xDone	BOOL	Output is TRUE when the transfer has completed without an error (even if the malfunction list is empty).
xError	BOOL	Output is TRUE if an error occurred during the transmission process.
xAborted	BOOL	Output is TRUE if the transmission was aborted using the “xAbort” input. Output is reset if xExecute and xAbort are FALSE at the same time.

Table 5.13: FB_ReadMalfunctionList Block Output Variable descriptions

5.5.3 Response to Read Malfunction List

Status Value	Status Description
0	Idle
1	Setting valve to inactive
2	Getting malfunction count
3	Transferring
4	Done
5	No entries in malfunction list
6	Error while transmitting data to VTEM
7	Aborting
8	Resetting
10	FB needs rising edge xExecute
100	Reserved
101	Invalid Set Valve Mode
102	Invalid Set App Control
103	Invalid Set App Option
104	Invalid Set Point Value 1
105	Invalid Set Point Value 2
106	Invalid configuration
110	Access denied Web Config
111	Access denied saving parameterization
112	No license for this MA
113	All licenses in use
114	Invalid license file
115	No valve detected
201	Valve self-calibration running
202	Invalid Channel
203	Invalid Transfer Control
204	Invalid iChoose Other Valve
205	Invalid Index
206	Invalid Transfer Value
207	No valve detected at selected slot
208	Access denied Data In Use

Table 5.14: Response to Read Malfunction List descriptions

5.5.4 Structure stMalfunction

Structure Variables	Data Type	Description
uiMalfunctionCode	UINT	Malfunction Code
uiMalfunctionSubcode	UINT	Malfunction Subcode
uiMalfunctionClassification	UINT	Malfunction Classification
uiHours	UINT	Timestamp (hours) of the malfunction message
uiMinutes	UINT	Timestamp (minutes) of the malfunction message
uiSeconds	UINT	Timestamp (seconds) of the malfunction message
uiMilliseconds	UINT	Timestamp (milliseconds) of the malfunction message

Table 5.15: Structure stMalfunction descriptions

5.5.5 Structure stTimeStamp

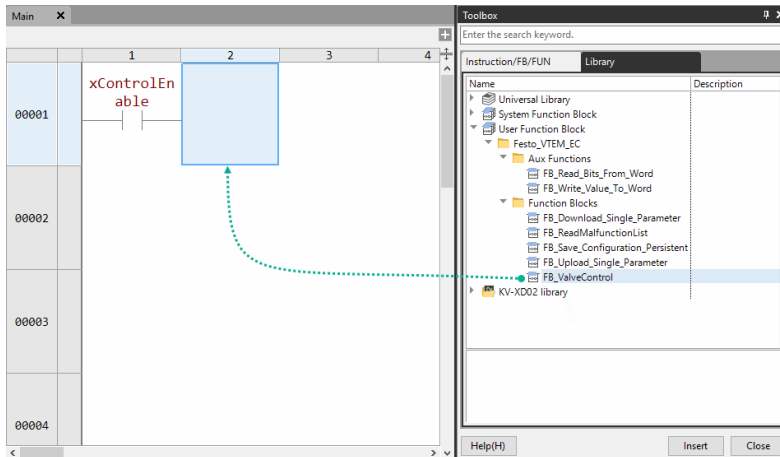
Structure Variables	Data Type	Description
uiHours	UINT	Timestamp (hours) of the malfunction message
uiMinutes	UINT	Timestamp (minutes) of the malfunction message
uiSeconds	UINT	Timestamp (seconds) of the malfunction message
uiMilliseconds	UINT	Timestamp (milliseconds) of the malfunction message

Table 5.16: Structure stTimeStamp descriptions

6 Configuring of Festo_VTEM_EC Function Blocks in KV STUDIO

This chapter describes how to add **Festo_VTEM_EC** library function blocks to a program & assign input and output tags.

Step 1



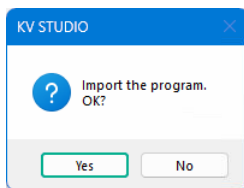
Drag and drop the **FB_ValveControl** function block from **Function Blocks** folder under **Festo_VTEM_EC** library.



Note

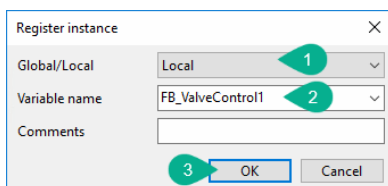
- Make sure to have separate variables for block **EN** bit for each of function block, to not have function block process data outputs from interfering with each other.

Step 2



Click the **Yes** button to import the function block to the project.

Step 3



1. Select the **Variable Scope** as required from the **Global/Local** drop-down list in the **Register instance** window.
2. Enter the required **Function Block** instance name in the **Variable name** field.
3. Click the **OK** button to register the variable.

Step 4

FB_ValveControl1		
FB_ValveControl		
EN		ENO
???-auiBusDataFromVTEM		auiBusDataToVTEM-???
???-xEnable		uiActualValveMode-???
???-uiSetValveMode		uiActualValveState-???
???-uiSetAppControl		uiActualAppState-???
???-uiSetAppOption		iActualValue1-???
???-iSetPointValue1		iActualValue2-???
???-iSetPointValue2		iResponseToValveModeSet-???
???-xManualAcknowledge		iErrorCode-???

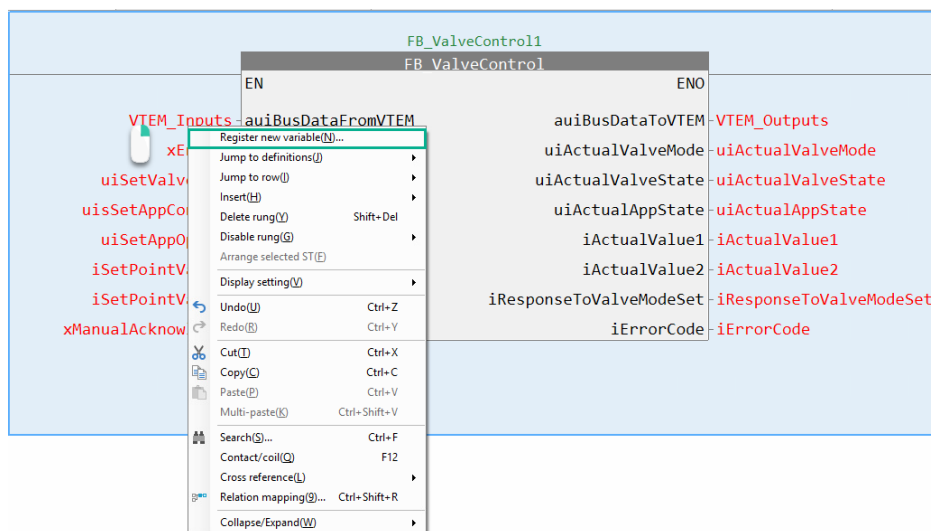
The added function block will be shown with **Instance Name** as shown above.

Step 5

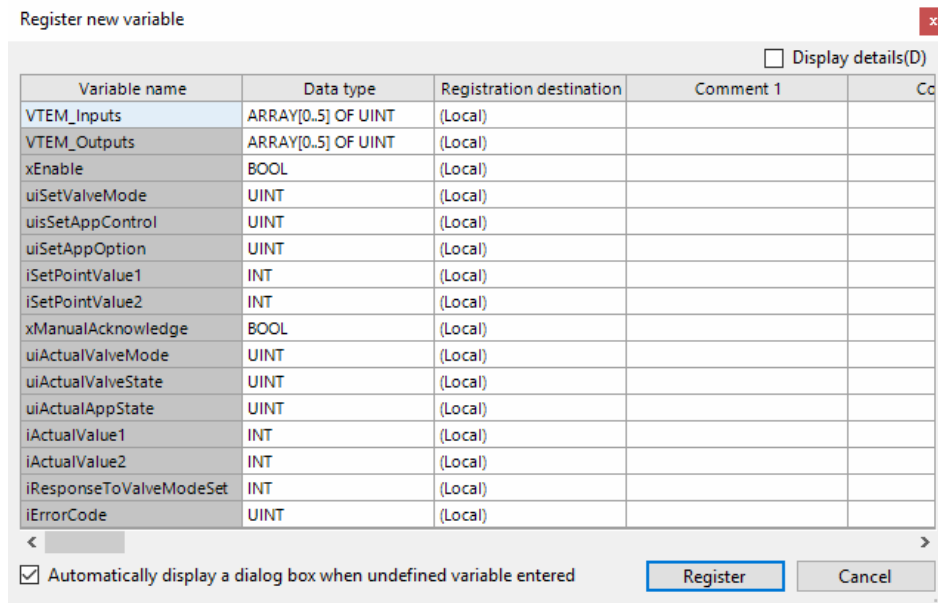
FB_ValveControl1		
FB_ValveControl		
EN		ENO
VTEM_Inputs-auiBusDataFromVTEM		auiBusDataToVTEM-VTEM_Outputs
xEnable-xEnable		uiActualValveMode-uiActualValveMode
uiSetValveMode-uiSetValveMode		uiActualValveState-uiActualValveState
uiSetAppControl-uiSetAppControl		uiActualAppState-uiActualAppState
uiSetAppOption-uiSetAppOption		iActualValue1-iActualValue1
iSetPointValue1-iSetPointValue1		iActualValue2-iActualValue2
iSetPointValue2-iSetPointValue2		iResponseToValveModeSet-iResponseToValveModeSet
xManualAcknowledge-xManualAcknowledge		iErrorCode-iErrorCode

Enter the required names for function block interface tags as shown above.

Step 6



Select and Right-click the function block, then click the **Register new variable(N)...** option to register all function block interface variables.

Step 7

Click the **Register** button to register all the created variables.

Step 8

FB_ValveControl1				
FB_ValveControl				
	EN		ENO	
VTEM_Inputs	auibusDataFromVTEM	auibusDataToVTEM	VTEM_Outputs	
xEnable	xEnable	uiActualValveMode	uiActualValveMode	
uiSetValveMode	uiSetValveMode	uiActualValveState	uiActualValveState	
uisSetAppControl	uisSetAppControl	uiActualAppState	uiActualAppState	
uiSetAppOption	uiSetAppOption	iActualValue1	iActualValue1	
iSetPointValue1	iSetPointValue1	iActualValue2	iActualValue2	
iSetPointValue2	iSetPointValue2	iResponseToValveModeSet	iResponseToValveModeSet	
xManualAcknowledge	xManualAcknowledge	iErrorCode	iErrorCode	

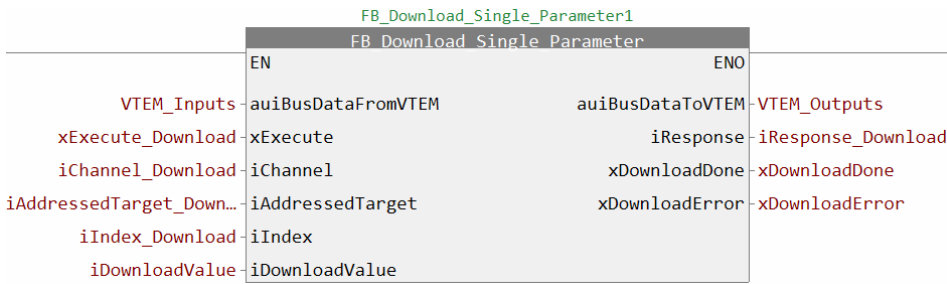
Once the function block interface variables have been registered successfully, **FB_ValveControl** function block will appear as shown above.

Follow the steps above to configure all the remaining function blocks of **Festo_VTEM_EC** library.

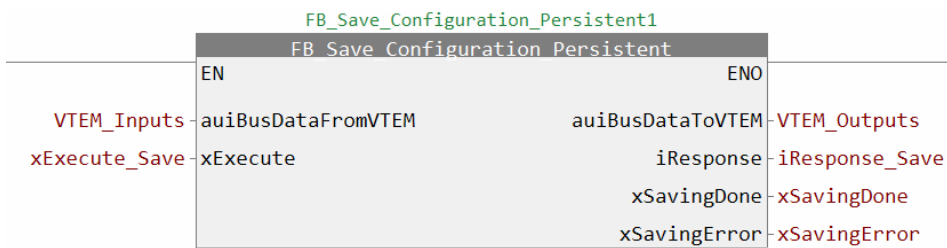
Once all the IO assignments are complete, the **FB_Upload_Single_Parameter** block will appear as shown below.

FB_Upload_Single_Parameter1				
FB Upload Single Parameter				
	EN		ENO	
VTEM_Inputs	auibusDataFromVTEM	auibusDataToVTEM	VTEM_Outputs	
xExecute_Upload	xExecute	iResponse	iResponse_Upload	
iChannel_Upload	iChannel	iUploadedValue	iUploadedValue	
iAddressedTarget_Upload	iAddressedTarget	xUploadDone	xUploadDone	
iIndex_Upload	iIndex	xUploadError	xUploadError	

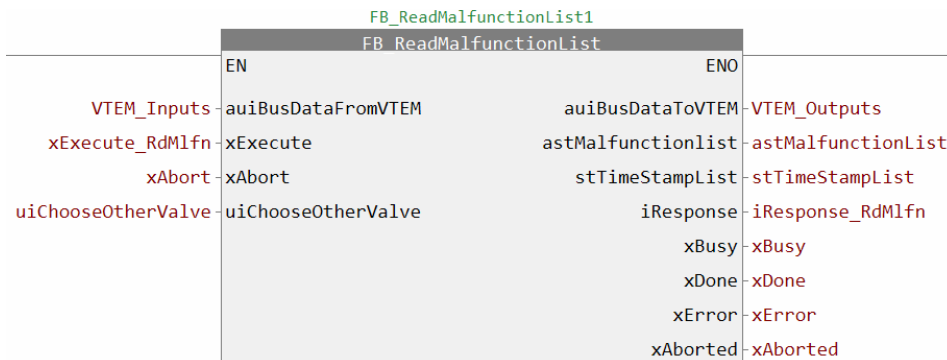
Once all the IO assignments are complete, the **FB_Download_Single_Parameter** block will appear as shown below.



Once all the IO assignments are complete, the **FB_Save_Configuration** block will appear as shown below.



Make sure to correctly set the structured data types for **astMalfunctionlist** and **stTimeStampList**. Once all the IO assignments are complete, the **FB_ReadMalfunctionList** block will appear as shown below.



Note

Unsafe system states when function blocks are called in parallel

Simultaneous access by multiple function blocks to the output data of a valve (**auibusDataToVTEM**).

- Ensure that no more than one function block instance can access the output data of a valve at the same time (**auibusDataToVTEM**) by activating only one block at a time.



Information

It is recommended to use a separate instance of the function blocks for each valve.

7 Sample Code for Festo_VTEM_EC Library Function Blocks

This sample code demonstrates how to control the VTEM Motion Terminal; read, write, and save parameters; and retrieve diagnostic data from VTEM Motion Terminal connected to KEYENCE KV-X PLC through EtherCAT using “**Festo_VTEM_EC**” library FBs.



Note

- In the configuration used in this application note, all FBs share same **auibusDataToVTEM**, so for each function block to operate correctly, ensure that only one function block is active at a time by enabling and disabling the **EN** bit of each FB accordingly.
- Therefore, before using each FB, make sure the **EN** bit is active for that FB, and **EN** bit is False for all other FBs.

7.1 Sample Code for Linking VTEM Motion Terminal Process Data of Valve in Slot 0 to Function Block Interface Tags

This sample code is used to copy process data input of Valve in Slot 0 from **VTEM Process Data** to the interface tags of the **Festo_VTEM_EC** library function blocks.

Input Mapping

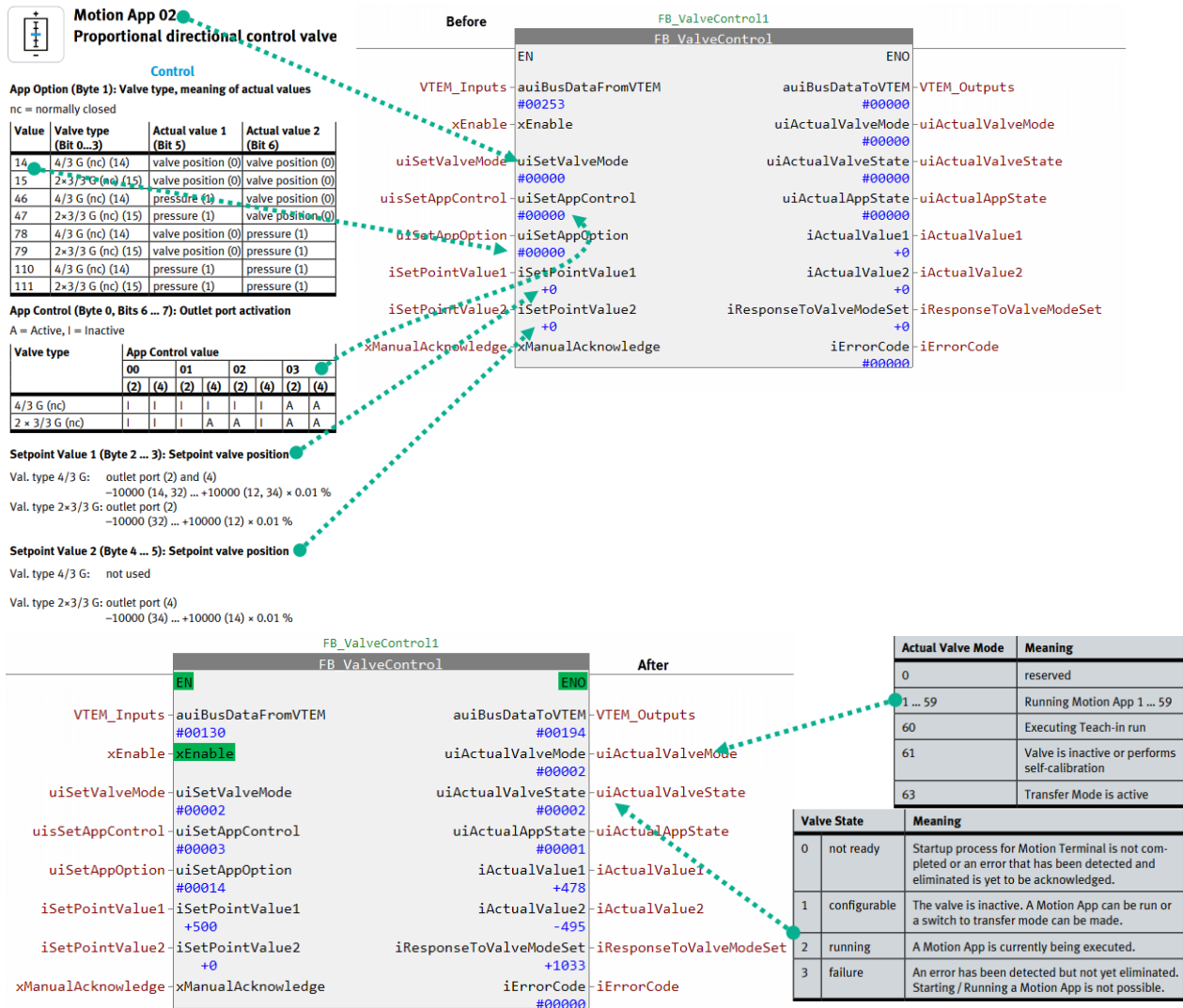


Output Mapping



The VTEM Motion Terminal process data contains information for up to eight valves, the maximum number supported by a single VTEM. However, the **auibusDataFromVTEM** and **auibusDataToVTEM** array interfaces of the function blocks are configured to handle data for only one valve. Therefore, the process data for the valve in the required VTEM slot must be copied to the function-block array interfaces. In this sample code, the process data for the valve in slot 0 is mapped to these interfaces.

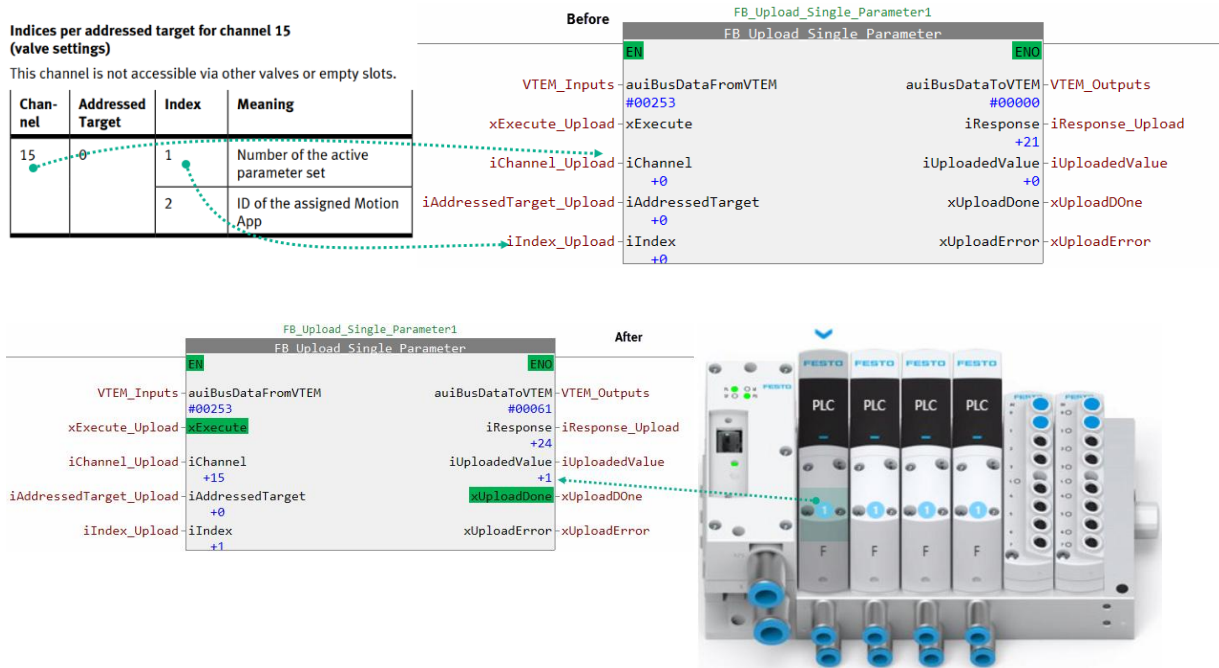
7.2 Sample Code for Controlling the VTEM Motion Terminal Valve using FB_ValveControl



1. Set **EN** to True by making **xControlEnable** True.
2. Enter the motion App number in the **uiSetValveMode** input tag. For this sample code, **Motion App 02** is used.
3. Enter the App control value in the **uiSetAppControl** input tag. For this sample code, **App Control 03** is used.
4. Enter the App option value in the **uiSetAppOption** input tag. For this sample code, **App Option 14 (16#0E)** is used.
5. Enter the setpoint value 1 in the **iSetPointValue1** input tag. For this sample code, **500** is set.
6. Enter the setpoint value 2 in the **iSetPointValue2** input tag. For this sample code, **0** is used because the App option selected for the selected Motion App does not require any value in Setpoint 2.
7. After all parameters have been entered in the input tags then toggle the **xEnable** tag to start valve control.
8. If Error shows in the **iErrorCode** monitor tag then user must execute the Acknowledge function using **xManualAcknowledge** control tag. Then user must enable the block once again.
9. Once the function block is enabled, user receives feedback from the valve in tags like **uiActualValveMode**, **uiActualValveState**, **uiActualAppState**, **iActualValue1** and **iActualValue2**.

10. And user will get the response to valve mode set value in **iResponseToValveModeSet** monitor tag in Integer format. (Refer Chapter 5.1.3).
11. If error occurs in the block, user can get the error code data in **iErrorCode** monitor tag in Integer format.

7.3 Sample Code for Reading the Valve Parameter-Set Value from the VTEM Motion Terminal using FB_Upload_Single_Parameter



1. Enter the channel number as **15** in the **iChannel** input tag.
2. Enter **0** in the **iAddressedTarget** input tag.
3. Enter index number as **1** in the **iIndex** input tag.
4. After all parameters have been entered in the input tags then toggle the **xExecute** tag to start the upload process.
5. When the parameter value has been uploaded successfully, then the monitor tag **xUploadDone** will be **True**.
6. The uploaded value is available in the **iUploadedValue** monitor tag in integer format. This value will be Parameter Set value, which is **1** as shown in the image above for this sample code.
7. User will get the response from the upload process in **iResponse** monitor tag in Integer format. (Refer Chapter 5.3.3).
8. If error occurs in the block user can get the error feedback in **xUploadError** monitor tag.



Note

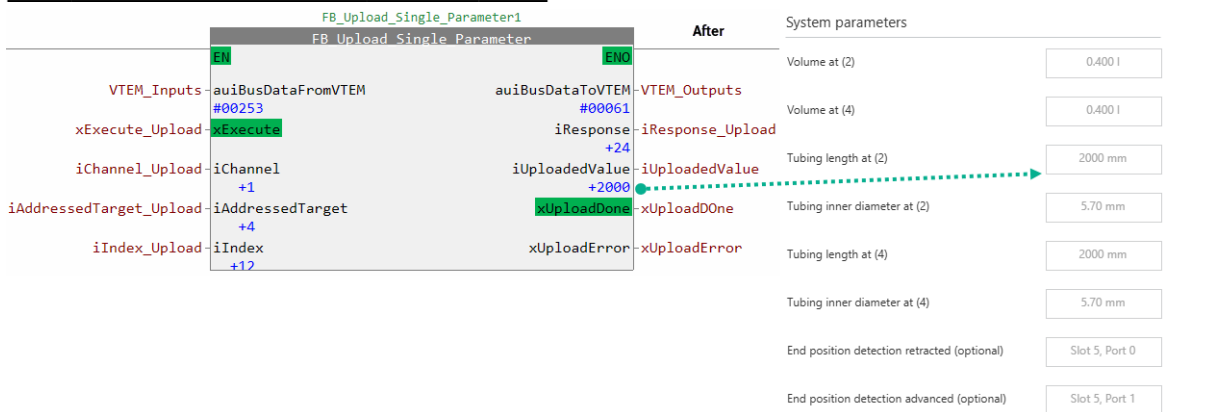
- Users can change the program set value for the valve using **FB_Download_Single_Parameter** follow steps like those shown in this chapter.

7.4 Sample Code for Uploading Parameter from the VTEM Motion Terminal Valve using FB_Upload_Single_Parameter



Motion App 04 Model-based proportional pressure regulation

Parameters						
Type	Index	Parameters	Range	Unit		
System	12	Tubing length at (2)	2000 ... 5000	1 mm		
	13	Tubing length at (4)	2000 ... 5000	1 mm		
	14	Tubing inner diameter at (2)	400 ... 1100	0.01 mm		
	15	Tubing inner diameter at (4)	400 ... 1100	0.01 mm		
	40	Volume at (2)	100 ... 5000	1 ml		
	41	Volume at (4)	100 ... 5000	1 ml		
	70	End position detection retracted (optional) (Slot number + 10 + port number)	20 ... 27 40 ... 47, 50 ... 57 80 ... 87, 90 ... 97	–		
	71	End position detection advanced (optional) (Slot number + 10 + port number)	20 ... 27 40 ... 47, 50 ... 57 80 ... 87, 90 ... 97	–		
	Application	182	Maximum pressure rise at (2)	10 ... 20000	0.01 bar/s	
		183	Maximum pressure rise at (4)	10 ... 20000	0.01 bar/s	



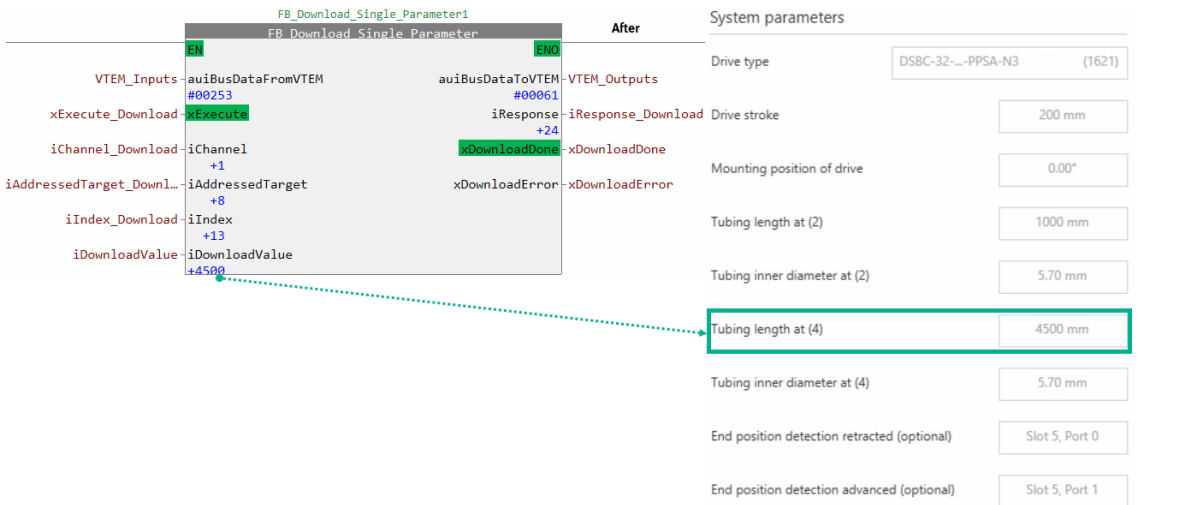
1. Enter the channel number, which corresponds to the parameter-set number. In this sample code, enter **1** (Refer Chapter 7.3) in the **iChannel** input tag.
2. Enter the Motion App number in the **iAddressedTarget** input tag. For this sample code, **Motion App 04** is used.
3. Enter the index number in the **iIndex** input tag. For this sample code, **12** is used.
4. After all parameters have been entered in the input tags, toggle the **xExecute** tag to start the upload process.
5. When the parameter value has been uploaded successfully, the monitor tag **xUploadDone** will turn to **True**.
6. The uploaded value is available in the **iUploadedValue** monitor tag in integer format.
7. Users will get the response from the upload process in **iResponse** monitor tag in Integer format. (Refer Chapter 5.3.3).
8. If an error occurs in the function block, the user can view the error feedback in **xUploadError** monitor tag.

7.5 Sample Code for Downloading Parameter to the VTEM Motion Terminal Valve using FB_Download_Single_Parameter



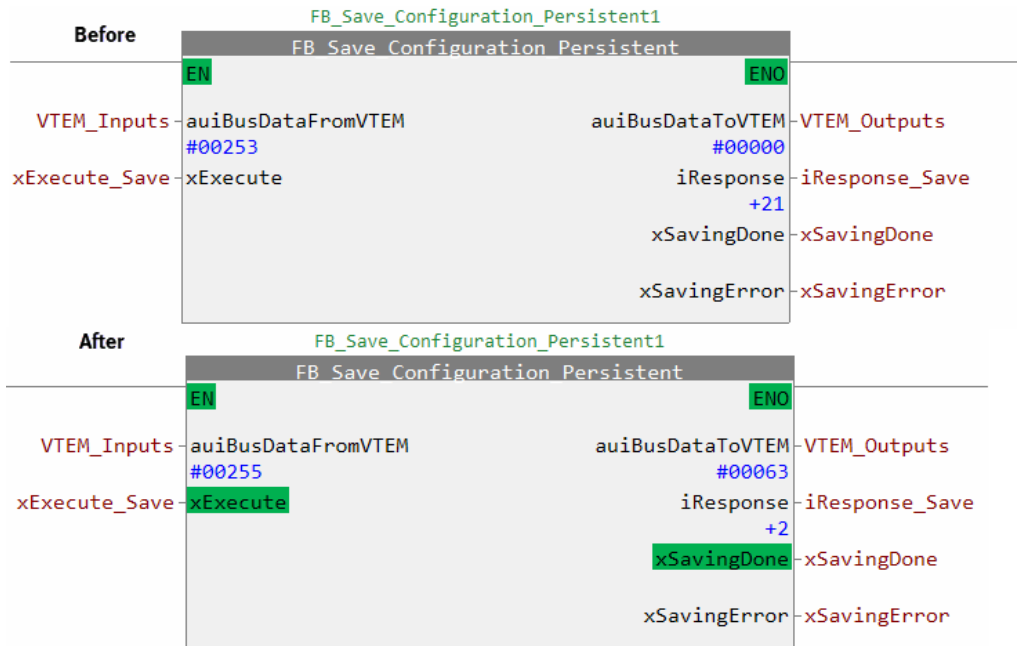
Motion App 08 Selectable pressure level (ECO)

Parameters					
Type	Index	Parameter	Range	Unit	
System	12	Tubing length at (2)	0 ... 20000	1 mm	
	13	Tubing length at (4)	0 ... 20000	1 mm	
	14	Tubing inner diameter at (2)	200 ... 1100	0.01 mm	
	15	Tubing inner diameter at (4)	200 ... 1100	0.01 mm	
	20	Drive type (→ List of supported drives at Support Portal)	–	–	
	21	Drive stroke	10 ... 5000	1 mm	
	60	Installation position of drive	–18000 ... +18000	0.01 °	
	70	End position detection retracted (optional) (Slot number × 10 + port number)	20 ... 27 40 ... 47, 50 ... 57 80 ... 87, 90 ... 97	–	
	71	End position detection advanced (optional) (Slot number × 10 + port number)	20 ... 27 40 ... 47, 50 ... 57 80 ... 87, 90 ... 97	–	



1. Enter the channel number, which corresponds to the parameter-set number. In this example, enter **1** (Refer Chapter 7.3) in the **iChannel** input tag.
2. Enter the Motion App number in the **iAddressedTarget** input tag. For this sample code, **Motion App 08** is used.
3. Enter the index number in the **iIndex** input tag. For this sample code, **13** is used.
4. Enter the parameter data value in the **iDownloadValue** input tag. For this sample code, **4500** is used.
5. After all parameters have been entered in the input tags, toggle the **xExecute** tag to start the Download process.
6. When the parameter value has been downloaded successfully, the monitor tag **xDownloadDone** will turn to **True**.
7. And users receive the response from the download process in **iResponse** monitor tag in Integer format. (Refer Chapter 5.3.3).
8. If an error occurs in the function block, the user can view the error feedback in **xDownloadError** monitor tag.

7.6 Sample Code for Saving the Parameters in the VTEM Motion Terminal Valve using FB_Save_Configuration



1. Toggle the **xExecute** tag to start the save process.
2. When the parameters have been saved successfully, the monitor tag **xSavingDone** will turn to **True**.
3. User will get the response from the save process in **iResponse** monitor tag in Integer format. ([Refer Chapter 5.4.3](#)).
4. If an error occurs in the function block, the user can view the error feedback in **xSavingError** monitor tag.

7.7 Sample code for Reading Malfunction list of the VTEM Motion Terminal Valve using FB_ReadMalfunctionList



1. In the **uiChooseOtherValve** input tag, select the valve whose process data is used or select a specific valve.
2. Toggle the **xExecute** control tag from false to true to read the malfunction list.
3. If an error appears in the **xError** monitor tag, the user must execute the abort function using the **xAbort** control tag.
4. Once the abort operation is complete, the monitor tag **xAborted** will be in true state. The user must then execute the block once again.
5. When the malfunction list has been read successfully, the monitor tag **xDone** will turn to **True**.
6. The malfunction list is available in the **astMalfunctionList** structure data type tag as shown in the image above.
7. The current time stamp is available in the **stTimeStampList** structure data type tag as shown in the image above.