

CMMT servo drive commissioning with MITSUBISHI IQ-R PLC By Modbus TCP Simple CPU Communication

This Application Note Gives Detailed Procedure to Integrate CMMT Servo Drive With Mitsubishi IQ-R Controller in Modbus TCP Simple CPU Communication Network. Also Described to Control the Servo Motor in Different Control Operational Modes.

CMMT-AS
CMMT-ST

Title CMMT Commissioning With MITSUBISHI IQ-R PLC By Modbus TCP Simple CPU Communication
Version 1.20
Document no. 100348
Original en
Author Festo

Last saved 25.09.2023

Copyright Notice

This documentation is the intellectual property of Festo SE & Co. KG, which also has the exclusive copyright. Any modification of the content, duplication or reprinting of this documentation as well as distribution to third parties can only be made with the express consent of Festo SE & Co. KG.

Festo SE & Co KG reserves the right to make modifications to this document in whole or in part. All brand and product names are trademarks or registered trademarks of their respective owners.

Legal Notice

Hardware, software, operating systems and drivers may only be used for the applications described and only in conjunction with components recommended by Festo SE & Co. KG.

Festo SE & Co. KG does not accept any liability for damages arising from the use of any incorrect or incomplete information contained in this documentation or any information missing therefrom.

Defects resulting from the improper handling of devices and modules are excluded from the warranty.

The data and information specified in this document should not be used for the implementation of safety functions relating to the protection of personnel and machinery.

No liability is accepted for claims for damages arising from a failure or functional defect. In other respects, the regulations with regard to liability from the terms and conditions of delivery, payment and use of software of Festo SE & Co. KG, which can be found at www.festo.com and can be supplied on request, shall apply.

All data contained in this document do not represent guaranteed specifications, particularly with regard to functionality, condition or quality, in the legal sense.

The information in this document serves only as basic information for the implementation of a specific, hypothetical application and is in no way intended as a substitute for the operating instructions of the respective manufacturers and the design and testing of the respective application by the user.

The operating instructions for Festo products can be found at www.festo.com.

Users of this document (application note) must verify that all functions described here also work correctly in the application. By reading this document and adhering to the specifications contained therein, users are also solely responsible for their own application.

Table of contents

1	Introduction	6
1.1	Hardware & Software requirements	8
1.1.1	PLC Firmware up-dation:	8
1.2	Topology Overview	9
1.3	Hardware Overview of CMMT Servo Drive	10
2	Configuration of CMMT Drive Parameterization Port using FAS	11
2.1.1	IP Address Configuration of CMMT Drive Parameterization Port X18	11
2.2	CMMT-AS Servo Drive Fundamental Configuration.....	13
2.2.1	Servo Drive Selection	13
2.2.2	Linear Axis Configuration	16
2.2.3	Rotary Axis Configuration	29
2.2.4	CMMT Drive Parameterization for Integrate with PLC.....	32
2.3	CMMT-AS Servo Drive Fieldbus Configuration	36
2.3.1	Switch Setting for Bus Protocol	37
3	Starting a Project.....	38
3.1	Create New Project File & Configurations of IQ-R Series PLC in Gx Works 3	38
3.2	Ethernet port parameters	41
3.2.1	Procedure to set IP address of PLC	41
3.3	Modbus TCP Simple CPU Communication Settings	42
3.3.1	Procedure to set IP address for RJ71EN71	42
3.3.2	Procedure to use Simple CPU Communication	42
3.3.3	Simple CPU Communication settings	43
3.3.4	Simple CPU Communication status	44
3.4	Adding Library to the project	46
3.4.1	Import PTP_Drives_Festo_MB_IQ-R in FB/Fun under program	46
4	Elements in the PTP_Drives_Festo_MB_IQ-R Library	50
4.1	PTP_Drives_Festo_MB.....	50
4.1.1	Input and Output Interface.....	52
4.1.2	ConfigEPos Bit Description	54
4.1.3	Epos ZSW1 Bit Details.....	55
4.1.4	Epos ZSW2 Bit Details.....	55
4.1.5	Diagnostic Messages	56
4.2	PTP_Drives_Parameter_MB.....	56
4.2.1	Input and Output Interface.....	57
4.2.2	Error Description	58
4.1	Word_2_DWord.....	58
4.1.1	Input Interface	59
4.1.2	Output Interface.....	59
4.2	DWord_2_Word.....	59
4.2.1	Input Interface	59
4.2.2	Output Interface.....	59

5	Library Configuration.....	60
5.1	Configuring the PTP_Drives_Festo_MB Function Block Input & Output Interface Tags in FBD	61
5.2	Configuring the PTP_Drives_Festo_MB Function Block Input & Output Interface Tags in Ladder.....	67
5.3	Configuring the PTP_Drives_Parameter_MB Function Block Input & Output Interface Tags in FBD	72
5.4	Configuring the PTP_Drives_Parameter_MB Function Block Input & Output Interface Tags in Ladder.....	75
6	PTP_Drives_Festo_MB FUNCTIONS	80
6.1	Axis Enabling & Stopping	80
6.2	Axis Software & Hardware Position Limits Enabling	80
6.3	Axis Actual Speed Monitoring.....	80
6.4	PtP_Drives_Festo_MB Modes of Operation	81
6.4.1	Relative Positioning Mode [ModePos – 1]	81
6.4.2	Absolute Positioning Mode [ModePos – 2].....	82
6.4.3	Setup Mode [ModePos – 3]	82
6.4.4	Home/Reference Point using Festo Automation Suite Configuration [ModePos – 4]	83
6.4.5	Home to Current Position [ModePos – 5].....	83
6.4.6	Traversing/Record Selection Mode [ModePos – 6]	83
6.4.7	Jog Mode [ModePos – 7].....	84
6.4.8	Inching/Incremental Jog Mode [ModePos – 8].....	84
6.4.9	Procedure to set the CMMT-AS in “Travel to fixed Stop” mode with the PtP_Drive_Festo_MB_ function block	84
7	PTP_Drives_Parameter_MB FUNCTIONS	87
7.1	Identification of Parameter Number, SubIndex, Elements Number during PTP_Drives_Parameter_MB Configuration.....	87
7.2	Read Parameter.....	88
7.3	Write Parameter	89
8	Sample code for Control CMMT-AS Drive using PTP_Drives_Festo_MB Funtion Block.....	90
8.1	Sample Code: Axis Initialization	90
8.2	Sample Code: Axis Homing.....	91
8.3	Sample Code: Axis Absolute Mode	91
8.4	Sample Code: Axis Jog Mode	93
8.5	Sample Code: Read parameter using PTP_Drives_Festo_Parameter_MB Function Block.	94
8.6	Sample code: Write parameter using PTP_Drives_Festo_Parameter_MB Function Block.	95
9	Extended Process Data	96
9.1	Set parameters in the FAS under Extended Process Data (EPD)	96
9.2	Add the EPD in PLC Logic.....	98
9.2.1	Logic to Read and Write the EPD	99
9.2.2	Changing the Record Table Value from EPD	100
10	Record table.....	103
10.1	Set the Task/Sequence in Record Table	103
10.2	Testing the Task/Sequence of control through FAS	105
11	Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.	108
11.1	Method 1 - Using PTP_Drives_Festo_MB Modes of Operation	108
11.1.1	Set EPD and Dynamics Parameters in FAS	108
11.1.2	Execute the Logic in PLC using EPD.....	110

11.1.3	Procedure to control in Torque Mode using PTP_Drive_Festo_MB function block (Refer Chapter 6.4.9)	110
11.2	Method 2 - Using FAS Record Table and PtP_Drives_Festo_MB Modes of Operation	116
11.2.1	Set the Task/Sequence in Record Table	116
11.2.2	Procedure to control in Traversing/Record Table using PTP_Drives_Festo_MB function block (Refer Chapter 6.4.9)	117
12	Additional Information	122
12.1	How to setup “Travel to fixed stop“ with CMMT-AS by using the PTP/telegram 111	122
12.1.1	Application description	122
12.1.2	Used Parameters for Torque	122
12.1.3	Travel top fixed stop (application class 3)	122
12.1.4	General parameters for “Travel to fixed stop”	126

1 Introduction

The main use of this application note is to describe the communication between CMMT Servo Drive and Mitsubishi PLC By Modbus TCP Simple CPU Communication. the following operations can be executed:

- Position mode in Relative/ Absolute mode
- Control mode-Speed/Velocity mode (Setup mode)
- Start command
- Stop command
- Record selection/Traversing Mode
- Jogging & Inching/Incremental jogging
- Axis Referencing &
- Axis homing

The application note has the description for the following:

1. Festo Automation Suite
 - Field bus communication settings between CMMT-AS and Mitsubishi PLC
 - How to set the Axis Configuration
 - How to set parameters in the Record table(with stroke limit function)
 - How to set parameters in the Extended Process Data (EPD)
2. PLC (Mitsubishi)
 - Modbus TCP Simple CPU Communication settings in GX Works3
 - Integration of PTP_Drives_Festo_MB_FX5U library into GXWorks3 project.
 - Description on the working of the PTP_Drives_Festo_MB_FX5U library.
 - Extended Process Data
 - How to control the torque control by Telegram 111 communication with PLC

PTP_Drives_Festo_MB_1 (PTP_Drives_Festo_MB)	
UW: uiModbusTCPCommStatus	DRV_Control :UW
UW: DRV_Status	AxisEnabled :B
W: ModePos	AxisPosOk :B
B: EnableAxis	AxisSpFixed :B
B: CancelTraversing	AxisAckSetpoint :B
B: IntermediateStop	ClampTorqueReached :B
B: Positive	AxisRef :B
B: Negative	AxisWarn :B
B: Jog1	AxisError :B
B: Jog2	Lockout :B
B: AckError	ActVelocity :E
B: ExecuteMode	ActPosition :D
B: ReleaseBrake	ActMode :W
B: TravelToFixStop	EPosZSW1 :UW
D: Position	EPosZSW2 :UW
D: Velocity	ActWarn :UW
W: OverV	ActFault :UW
W: OverAcc	Error :B
W: OverDec	Status :UW
UD: ConfigEPos	
E: BaseSpeedValue	

1.1 Hardware & Software requirements

Device Name	Version (or) Part Number	Device Type
CMMT-AS-...-MP Drive	V33.0.9.10 & above	Hardware
EtherNet Cable	NEBC-R3G4-ES-1-S-R3G4-ET	Hardware
R-...-ENCPU (Mitsubishi PLC)	Firmware V52 & above	Hardware
GxWorks3	V1.090U & Above	Software
PTP_Drives_Festo_MB_IQ-R	V1.3	Software
Festo Automation Suite	V2.5.0.635	Software
Festo Field Device Tool	V2.10.5.198	Software



Warning

- The **built-in Ethernet port** of the "**R-CPU**" is **incompatible** with "**Simple CPU communication-Modbus TCP**," however an **additional module, RJ71EN71(E+IEF)**, may be **used with the R CPU to enable this functionality**.
- If **RJ71EN71(E+IEF)** is added to the **CPU Extension** of the **R_EN PLC**, "**Simple CPU communication-Modbus TCP Compatible**" is then **possible** without the need for a separate RJ71EN71 Module.
- Simple CPU communication-Modbus TCP is only **supported** on **Port 1** of **RJ71EN71(E+IEF)**.
- **Simple CPU Communication (Modbus TCP Compatible)** is possible in **GX Works3** version **1.090U** and above.

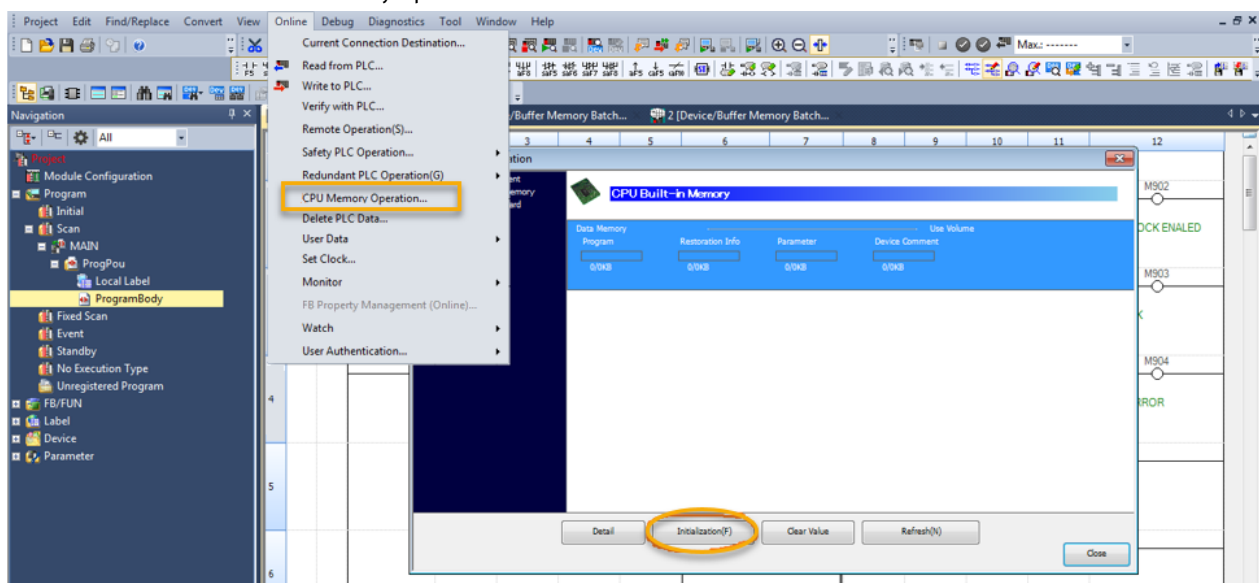
1.1.1 PLC Firmware up-dation:

- Initial Checks: (PLC year of Manufacture, PLC Firmware, Software- GX Works3 - 1.090 (Version) and above.
- Check Firmware version of PLC- should be 52 & above (GX Works 3- In Diagnostics, System monitor- Firmware version data will be available). If it less than 52, update Firm ware using SD Card.

Firmware Update Procedure- In empty SD Card, put Firm ware file.

Take Program Backup of PLC Programs, including intelligent module if available.

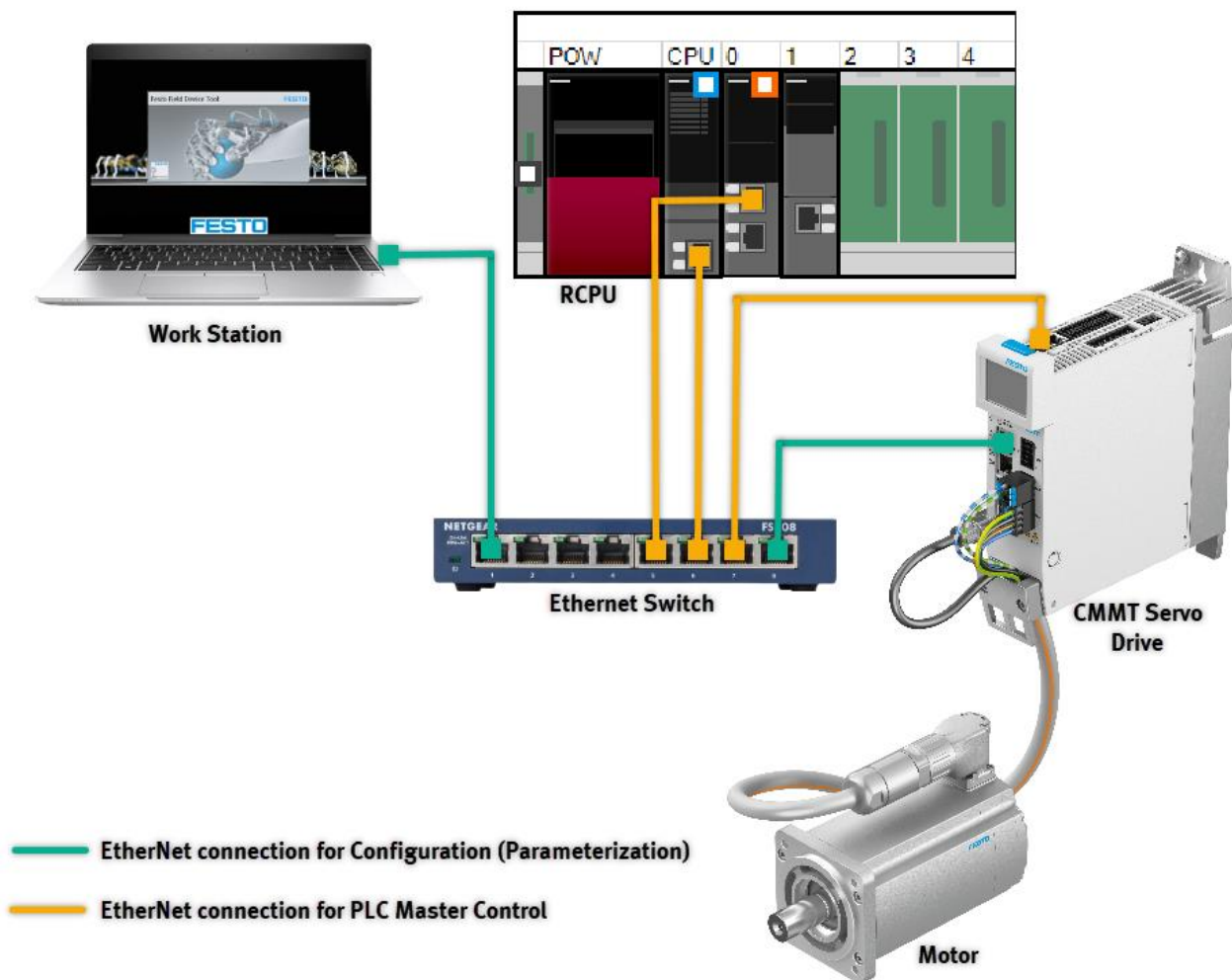
Initialize PLC- Online- CPU Memory operation-Initialize.



- Make PLC stop mode, switch off PLC , insert memory card, switch ON PLC, Card LED in PLC will get ON (denotes updating), Error LED and P. RUN LED blinks (denotes completion of Update).
- Off PLC, remove Memory card, make PLC Run mode, and switch ON PLC, cross check Firmware.

1.2 Topology Overview

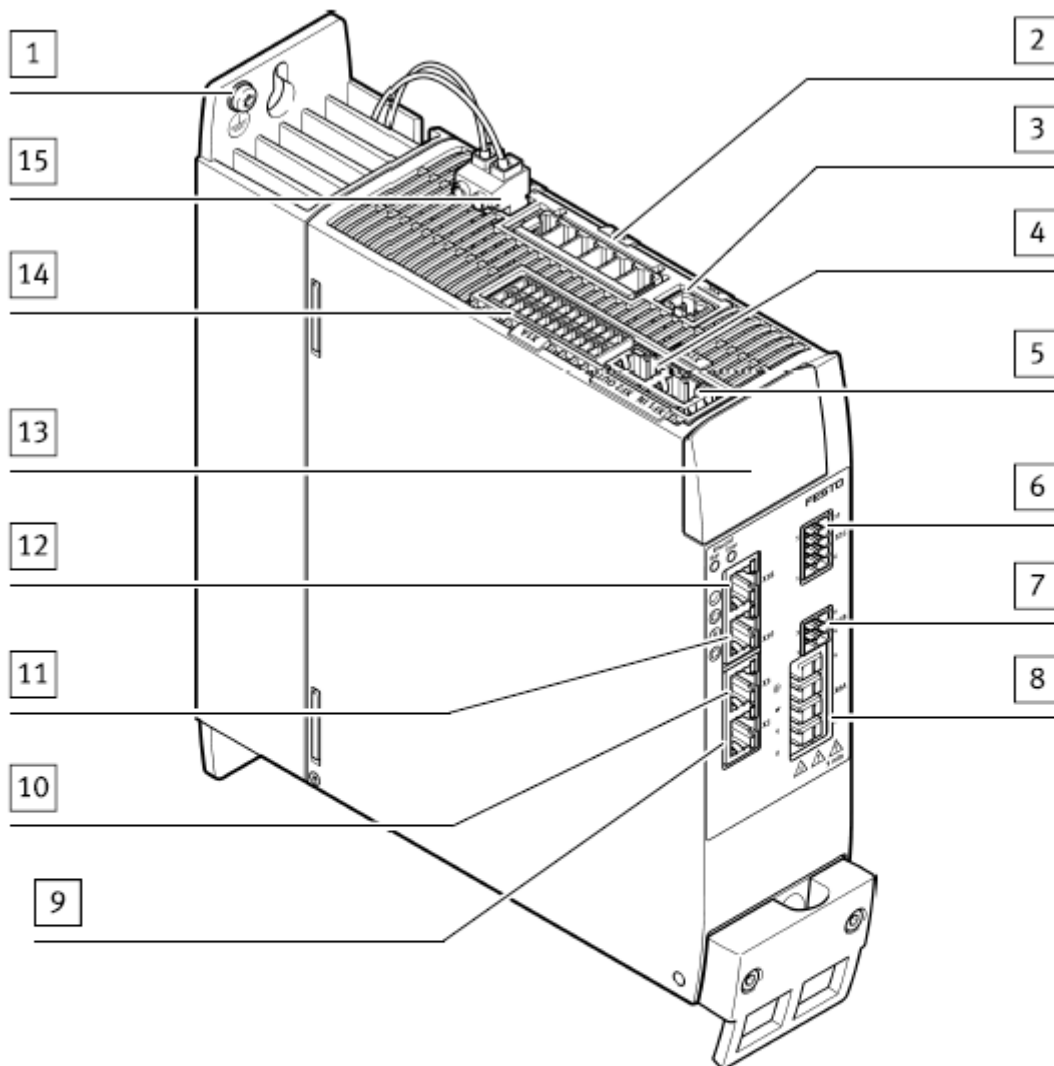
When we want to communicate between Mitsubishi PLC and CMMT motor controller via Modbus TCP Simple CPU Communication protocol the following topology must be followed.



Note

- Above network configuration is an example network for this application note but depends on users network configuration above architecture may change.

1.3 Hardware Overview of CMMT Servo Drive



- | | |
|---|--|
| 1 PE connection, housing | 9 [X2] Encoder connection 1 |
| 2 [X9A] Mains and intermediate circuit connection | 10 [X3] Encoder connection 2 |
| 3 [X9C] Logic voltage | 11 [X10] Device synchronisation |
| 4 [XF2 OUT] RTE interface port 2 | 12 [X18] Standard Ethernet |
| 5 [XF1 IN] RTE interface port 1 | 13 [X5] Connection for operating unit (behind the blind plate) |
| 6 [X1C] Inputs/outputs for the axis | 14 [X1A] I/O interface |
| 7 [X6B] Motor auxiliary connection | 15 [X9B] Connection for braking resistor |
| 8 [X6A] Motor phase connection | |

2 Configuration of CMMT Drive Parameterization Port using FAS

2.1.1 IP Address Configuration of CMMT Drive Parameterization Port X18

This chapter describes about IP address configuration of CMMT drive parametrizing port using Festo Automation Suite.



Note

- Before beginning the CMMT drive IP address setting procedure, ensure that the hardware connection between the PC and the servo drive has been made as per with the communication architecture provided in [chapter 1.2](#).
- In [chapter 1.2](#), the green colour connection is only used for parametrization; after parametrization, connect the PC to the EtherNet switch as shown in the orange colour line to establish a connection with the PLC.

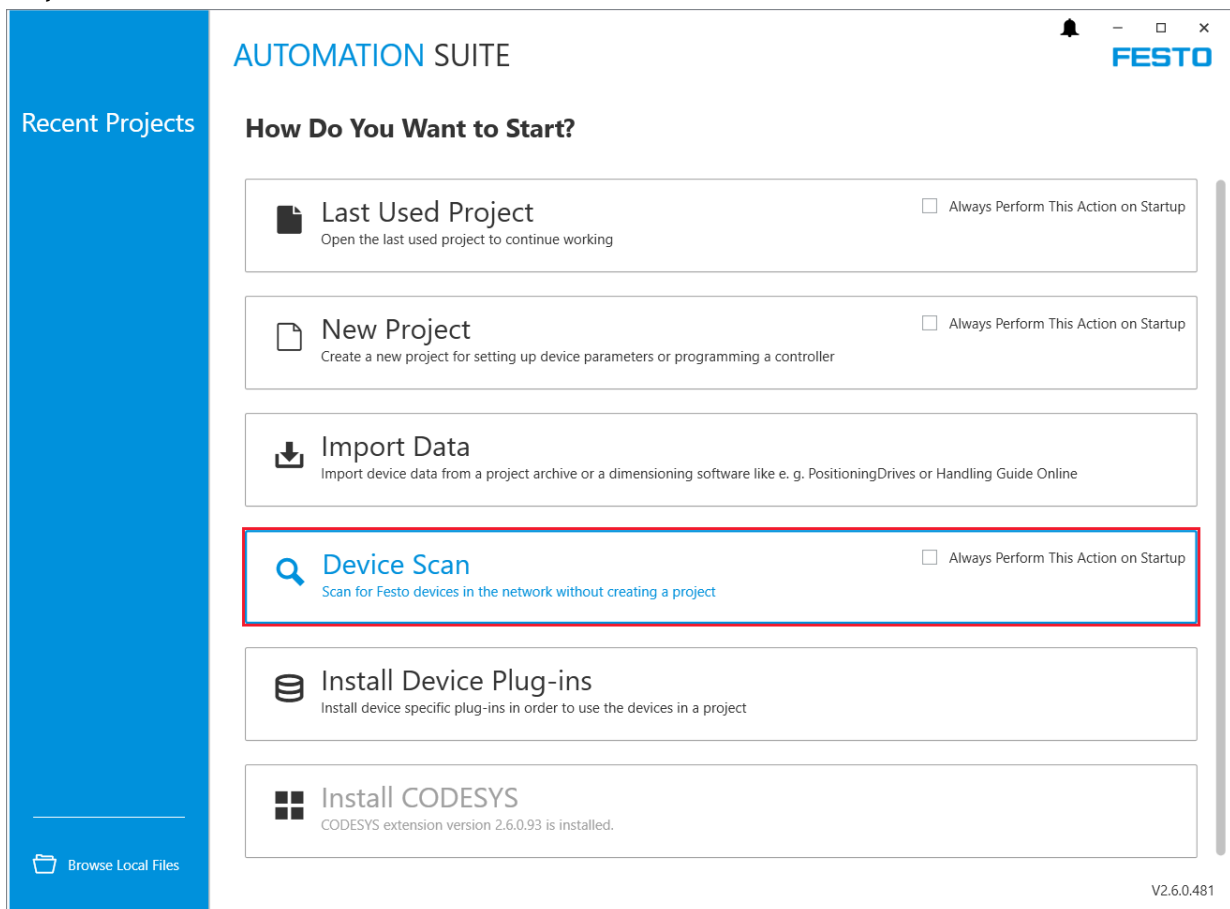
Step 1



Festo Automation Suite

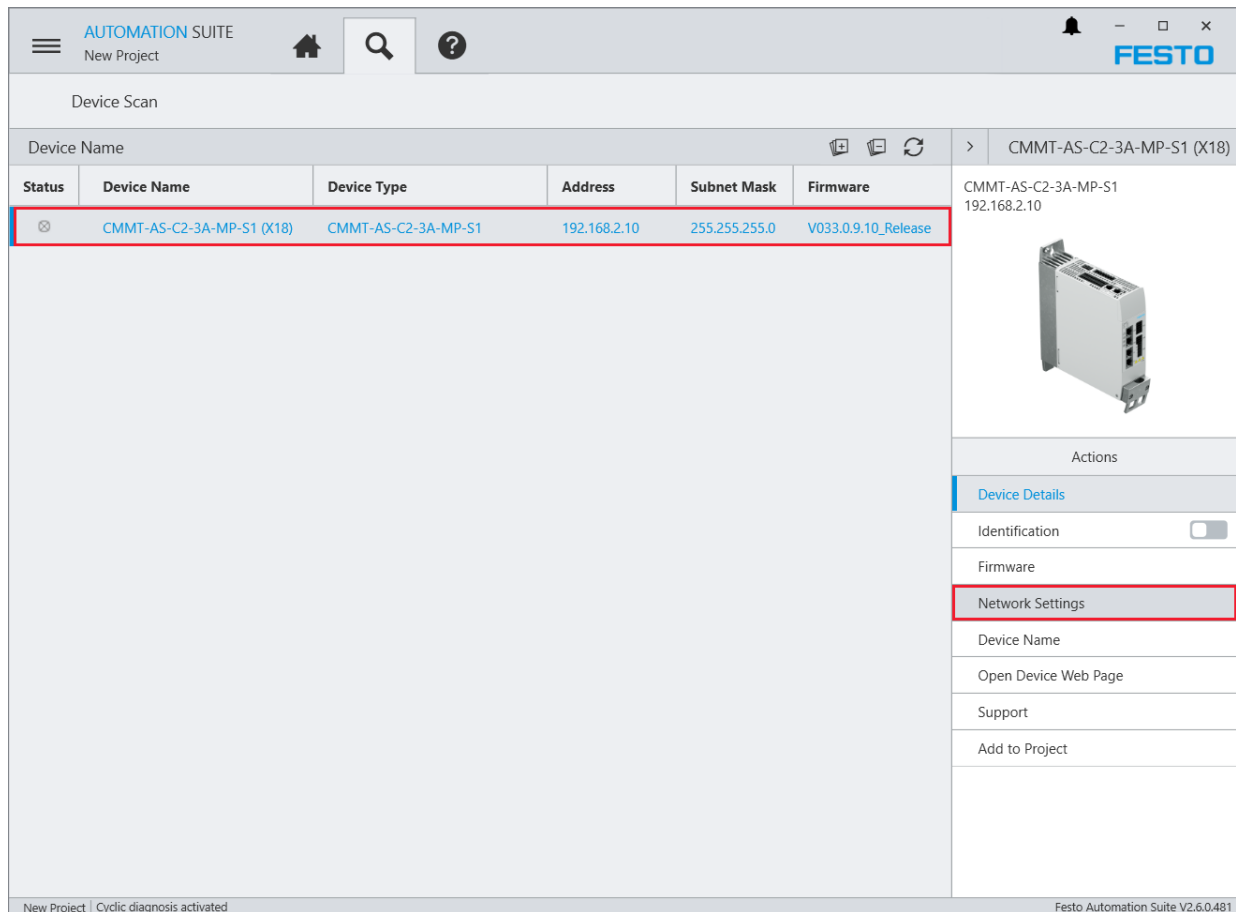
- Open the Festo Automation Suite from following path **Windows Start > Festo Software > Festo Automation Suite Tool**

Step 2



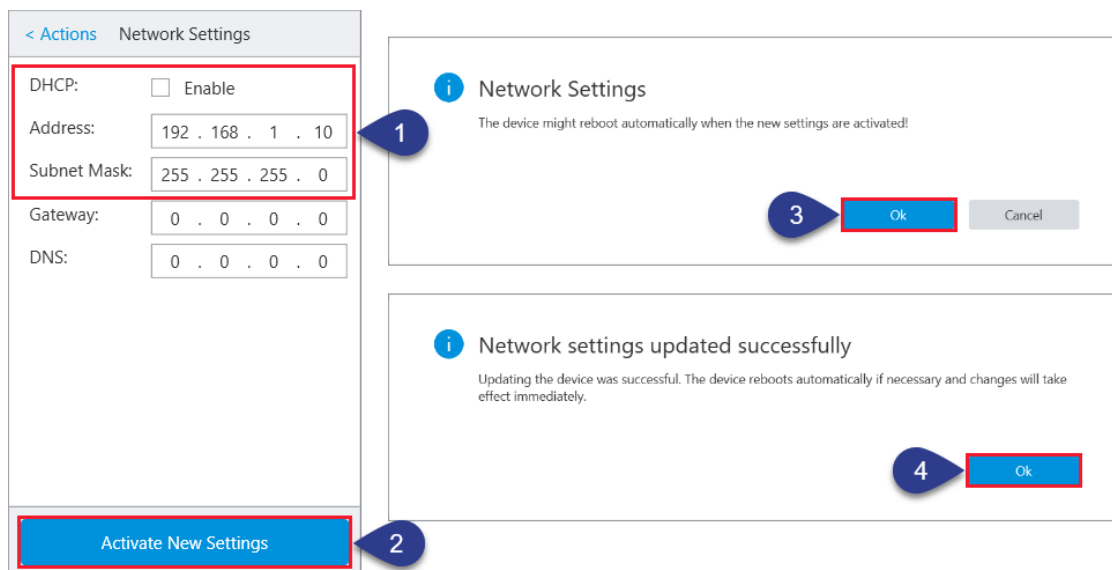
- Click **“Device Scan”** button from tool bar.

Step 3



- Select appropriate device with extension (X18) and then click on **Network Setting** from action window.

Step 4



1. **Uncheck the DHCP enable, Enter the new IP Address & Subnet Mask** as per local network.
2. Click **“Activate New Setting”** to update the changes.
3. Click **“OK”** to confirm the change of **“Network Settings”**.
4. Click **“OK”** to confirm.

2.2 CMMT-AS Servo Drive Fundamental Configuration

2.2.1 Servo Drive Selection



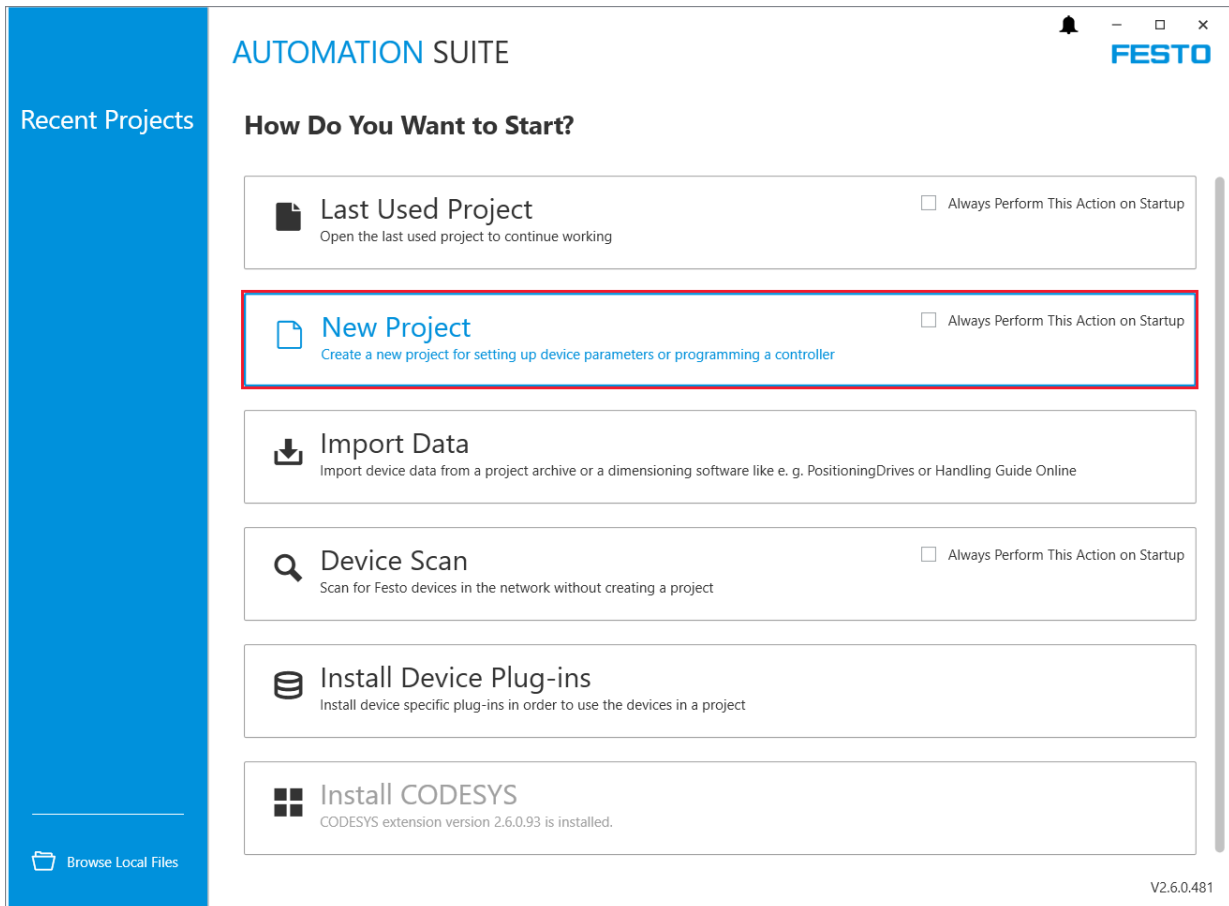
Note

- Follow the steps 1 to 4 **if drive is not in online**. If drive is connected with PC as shown in [Chapter 1.2](#), click Device scan from home as mentioned in [Chapter 2.1 Step 2](#) & jump to **Step 4**.

Step 1

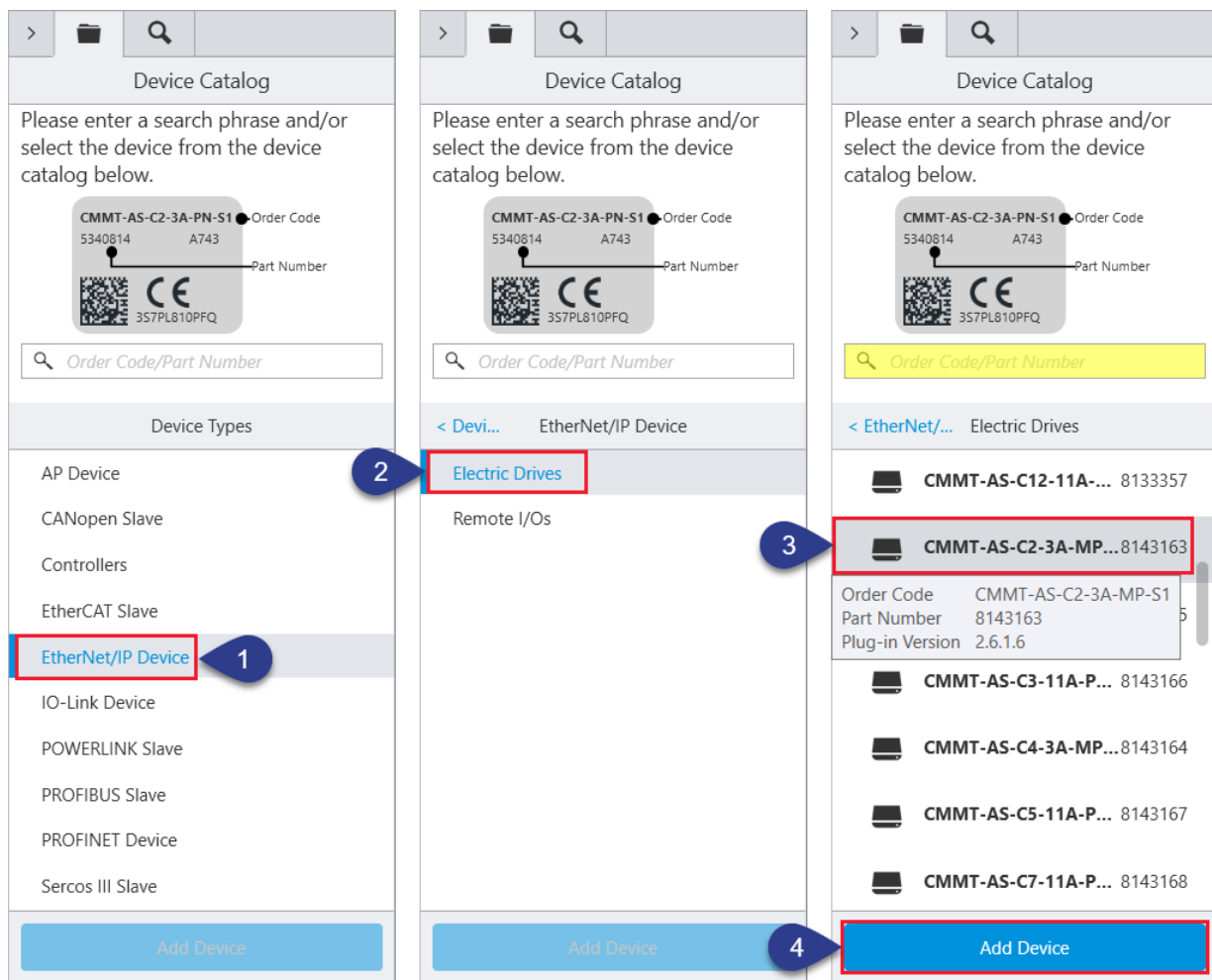
- Open **FAS** software as per [Chapter 2.1 Step 1](#).

Step 2



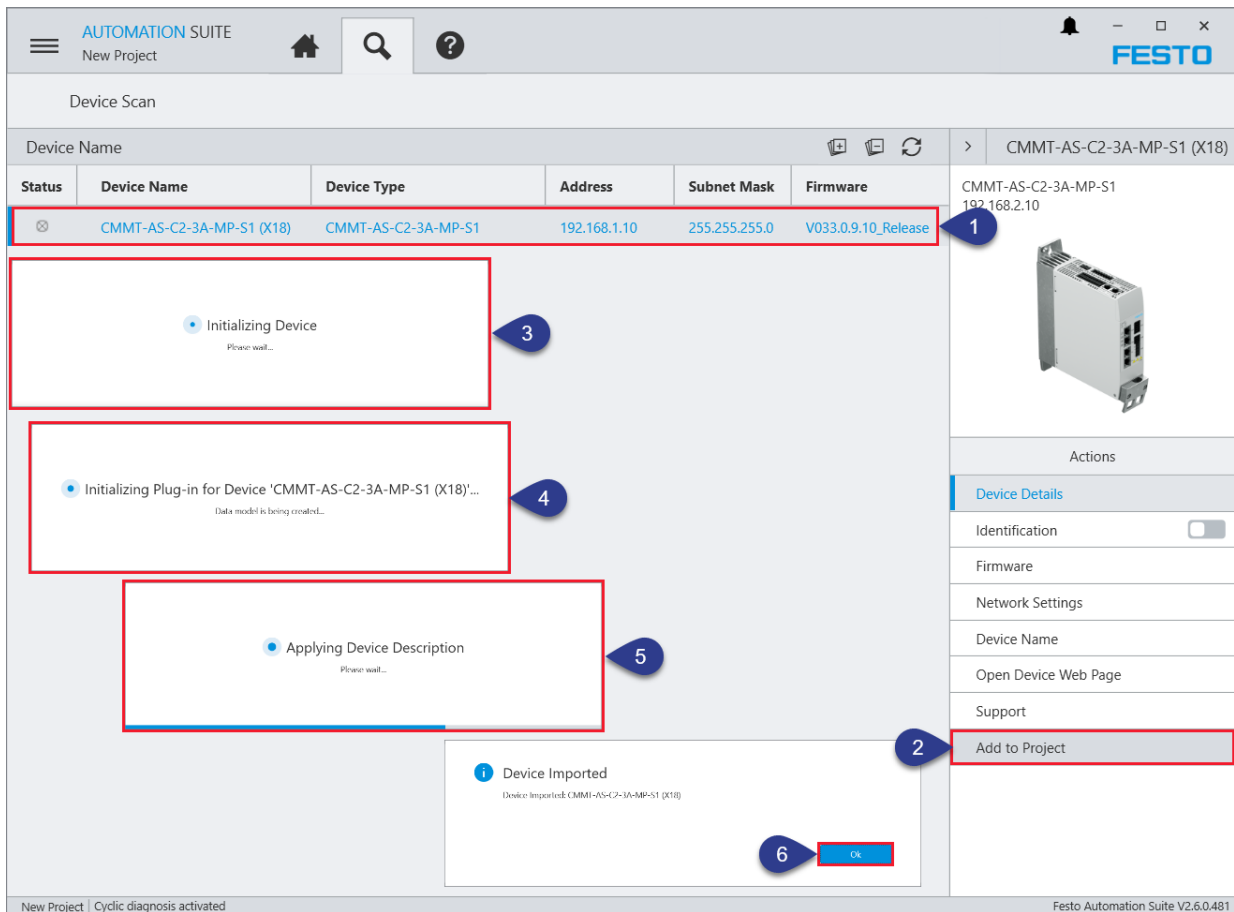
- Select **“New Project”** from start-up menu.

Step 3



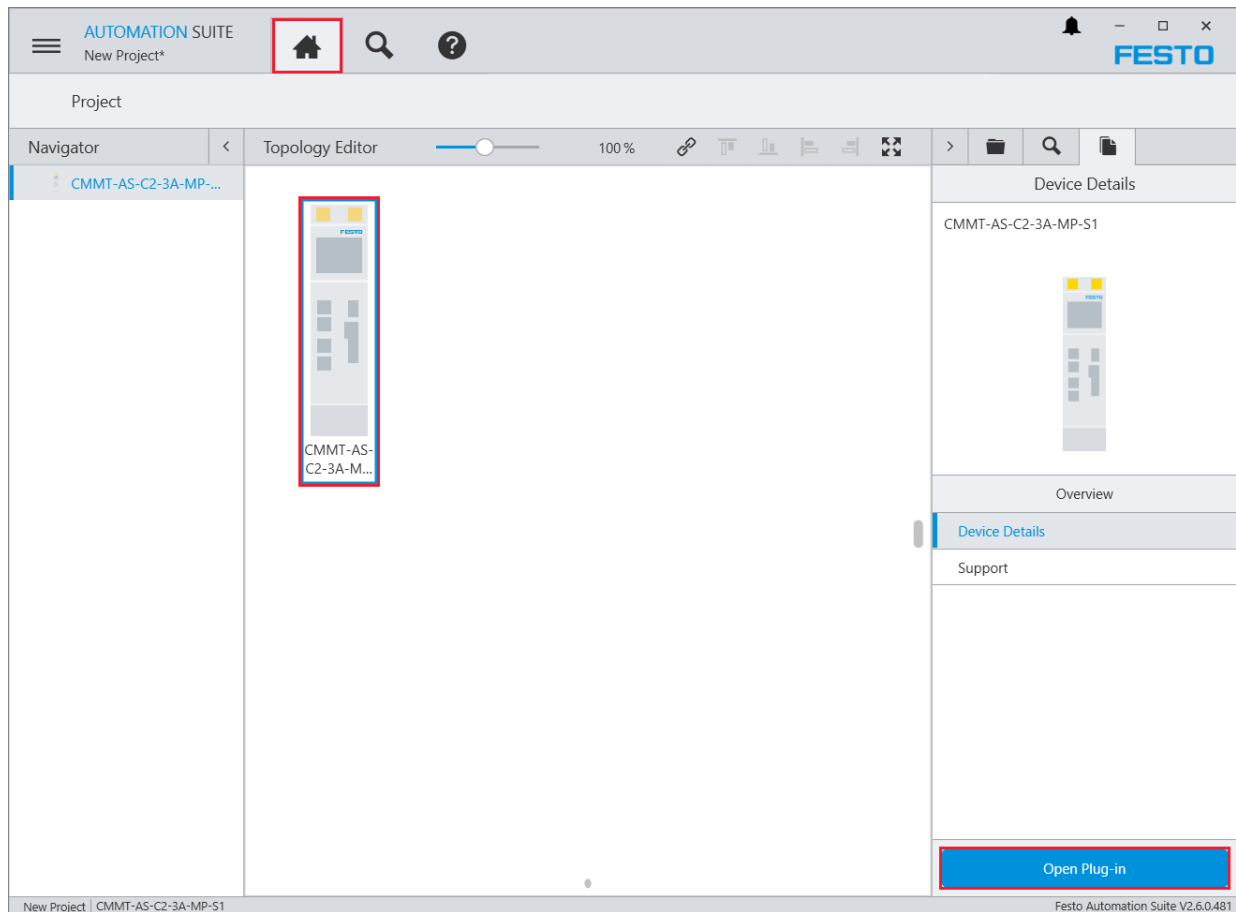
1. Click Ethernet/IP Device from the device repository.
2. Select **“Electric Drives”**.
3. Select **“Correct Drive Part Number”** from electric drives list or enter either the **“Part Number or Order Code”** as shown above image in **“Search Box”** highlighted in yellow colour rectangular box. As per this sample application **CMMT-AS-C2-3A-MP-S1** is selected.
4. After selecting the drive click on **“Add Device”**.

Step 4



1. Select the correct device based on Its configured name & IP address.
2. Click **“Add to Project”**.
3. Device initializing will start.
4. Device Plug-in initialization will start.
5. Once Plug-in initialized device will start to import the description data.
6. Once device imported click on **“OK”** to confirm the successful device import.

Step 5



- Go to **Project Section** tab & double click the newly added drive or Click **Open Plug-in** to initialize configuration parameters.

2.2.2 Linear Axis Configuration

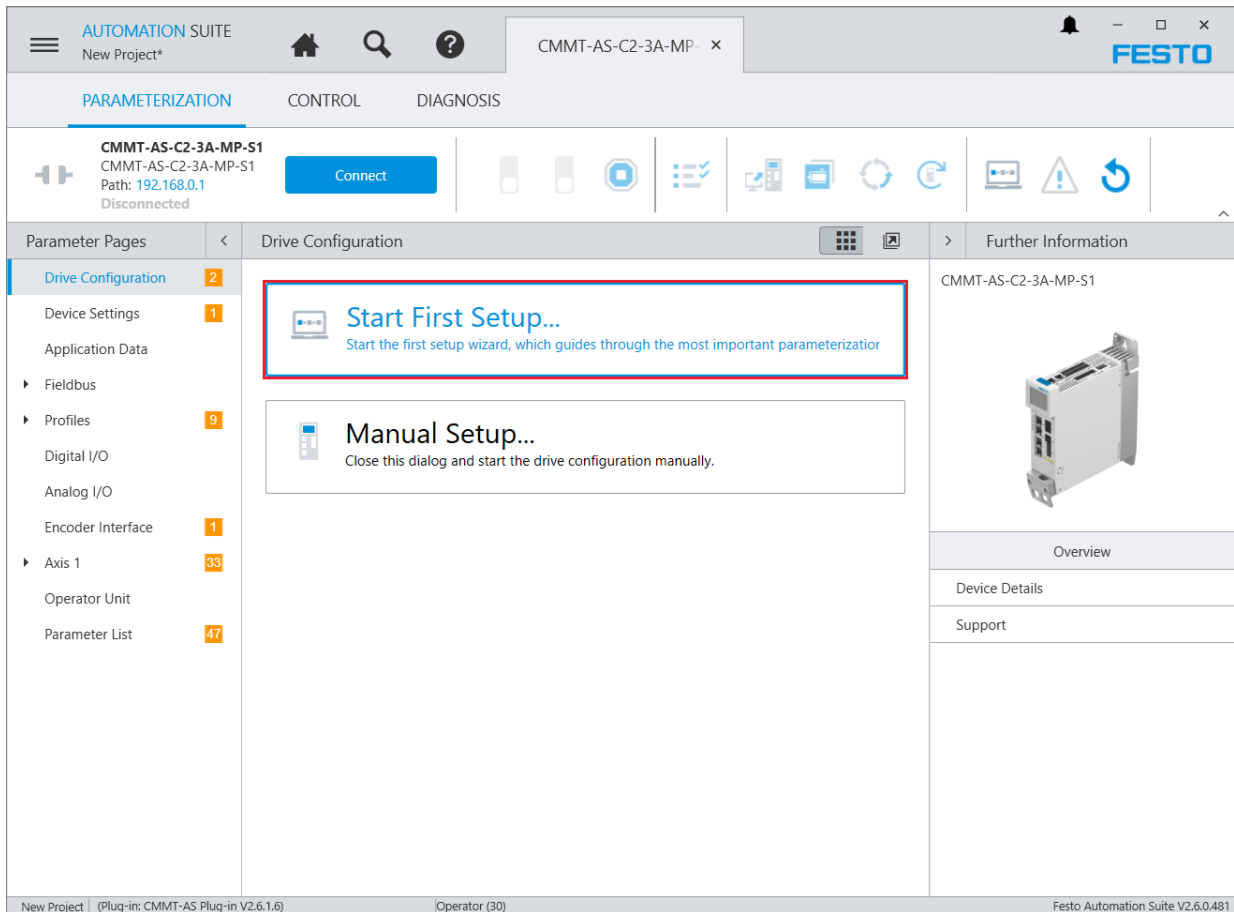
**Note**

- If the user is looking for Rotary Axis configuration skip this chapter and jump to [Chapter 2.2.3](#).

This chapter describes the parameterization of the CMMT drive for a linear axis to move 100 mm stroke length with 0.50 m/s as base speed & current position as homing mode.

Step 1 – Motor Configuration

1. Select **“Start first setup”** from drive configuration menu.



2. Once you click Start first setup following will be displayed.

Configuration of CMMT Drive Parameterization Port using FAS

First setup

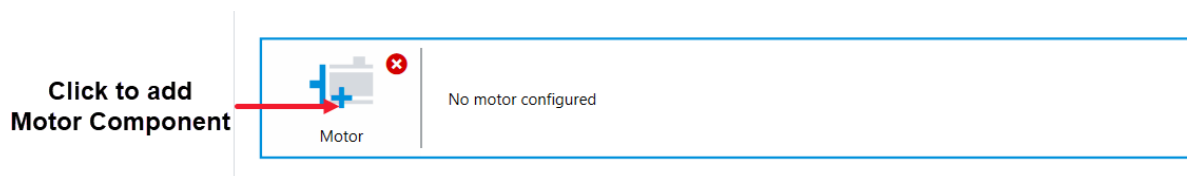
Please Select the Components of Your Drive System

Component	Details
Servo drive	CMMT-AS-C2-3A-MP-S1 8143163 Licenses Maximum Current: 6.00 A Intermediate Circuit Voltage: 320.00 V Supply Voltage: 230.00 V Ext. Brake Resistor: Deactivated
Motor	No motor configured
Axis	No axis configured
Mounting Kit	No mounting kit configured

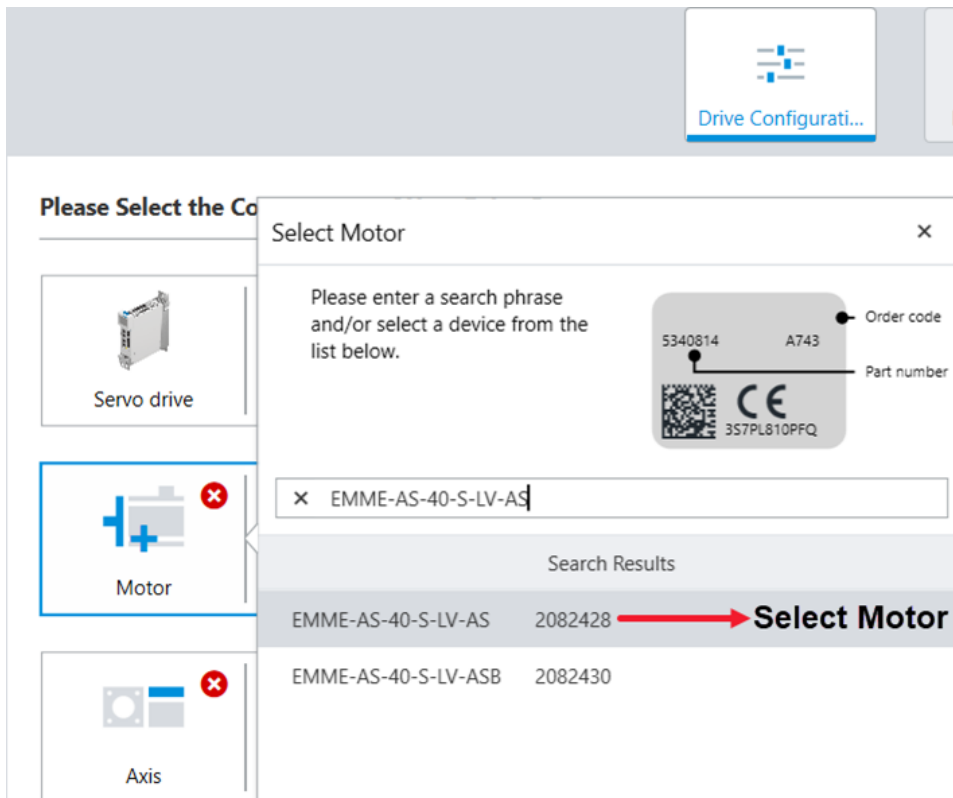
Back Next Finish

New Project | (Plug-in: CMMT-AS Plug-in V2.6.1.6) | Operator (30) | Festo Automation Suite V2.6.0.481

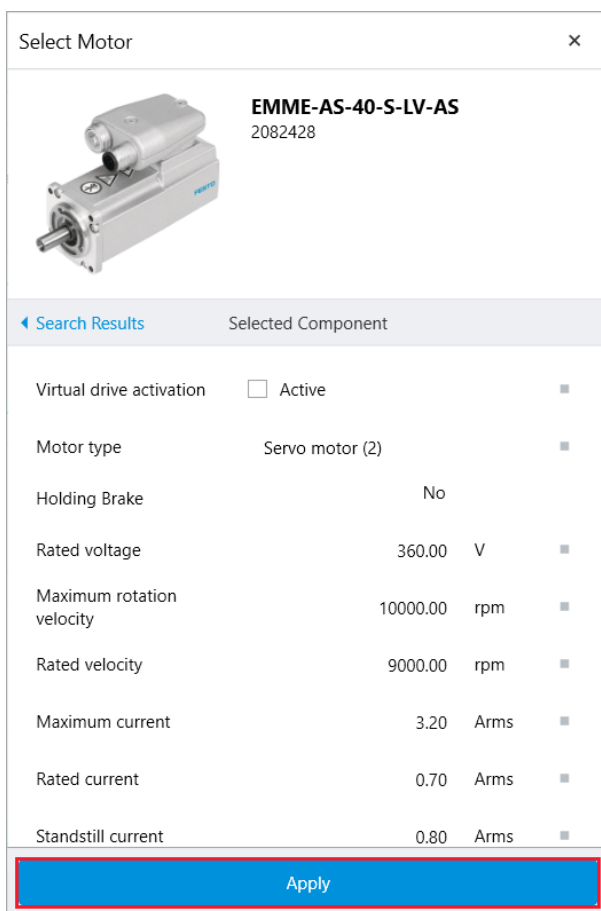
- Click **Motor** to add the motor component.



- Select the correct motor from the list of available motors. As per this sample application “**EMME-AS-40-S-LV-AS**” motor is selected.



5. After selecting the motor click **Apply** to add the motor to project.



6. The motor tab will look as shown below after configuration.

 Motor	EMME-AS-40-S-LV-AS 2082428	Type Servo motor (2)	Holding Brake No	Encoder Protocol Hiperface (0)	Encoder Type Single turn (1)	Voltage 360.00 V
--	--------------------------------------	--------------------------------	----------------------------	--	--	----------------------------

Step 2 – Axis Configuration

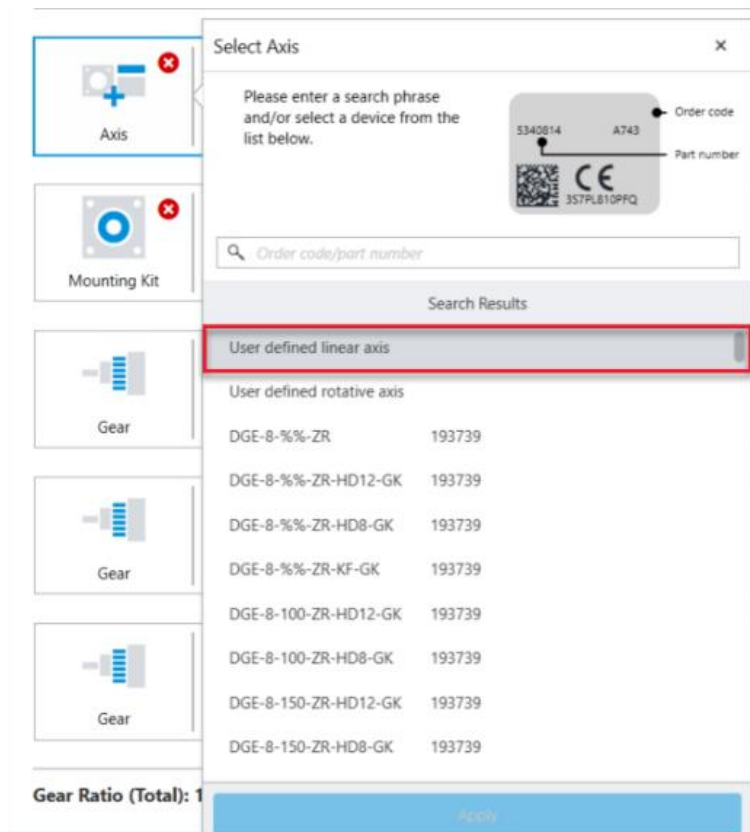
This step describes the axis type configuration. As specified initially here axis configuration is done for 100 mm in linear stock length. This configuration will change based on user application type.

1. Click **Axis** to add the required axis to project.

Click to add
Axis



2. Select the required axis from the available list. As per this sample application **User Defined Linear Axis** is selected.



3. After selecting the axis enter the needed configuration details. Click **Apply** to add the axis to project.

Select Axis



User defined linear axis

Search Results

Selected Component

Actual user unit

Metric [m, m/s, ...] (6)

Motion

Linear

Unlimited axis

☐ Active

Working Stroke

100.00 mm

Feed Constant

100.00 mm/rev

Design axis

Single axis (0)

Apply

- The Axis tab will look as shown below after configuration.



Axis

User defined linear axis

Feed Constant

100.00 mm/rev

Working Stroke

100.00 mm

Step 3 – Mounting Kit Configuration

This step illustrates the mounting kit configuration. This configuration will vary depending on user application type. At the end of this step click next button to move to next level configuration.

- Click **Mounting Kit** to add a new Mounting Kit to project.

Click to add Mounting Kit

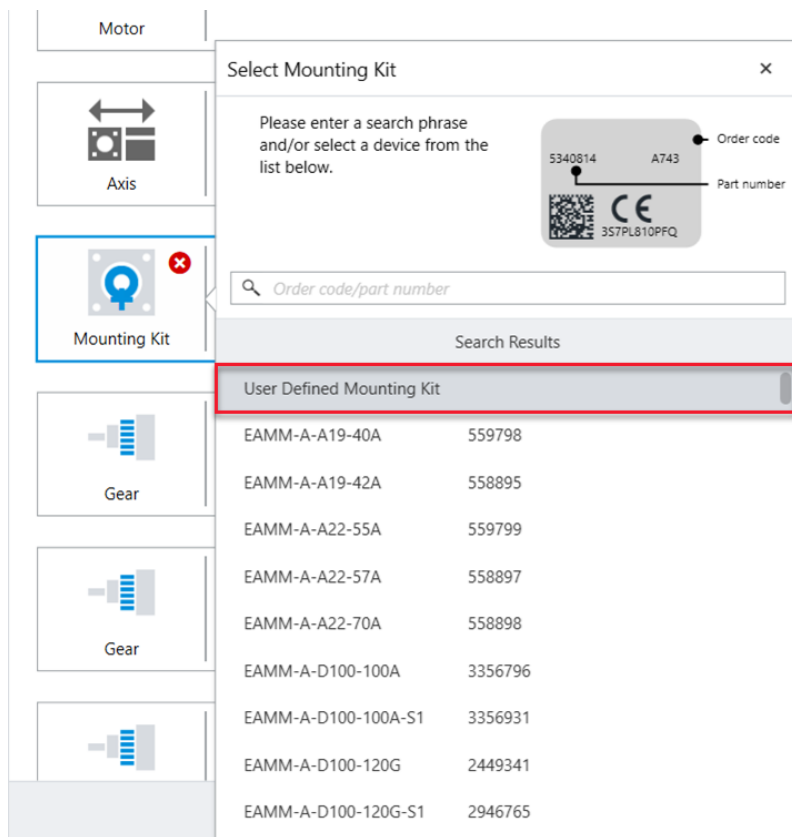


Mounting Kit

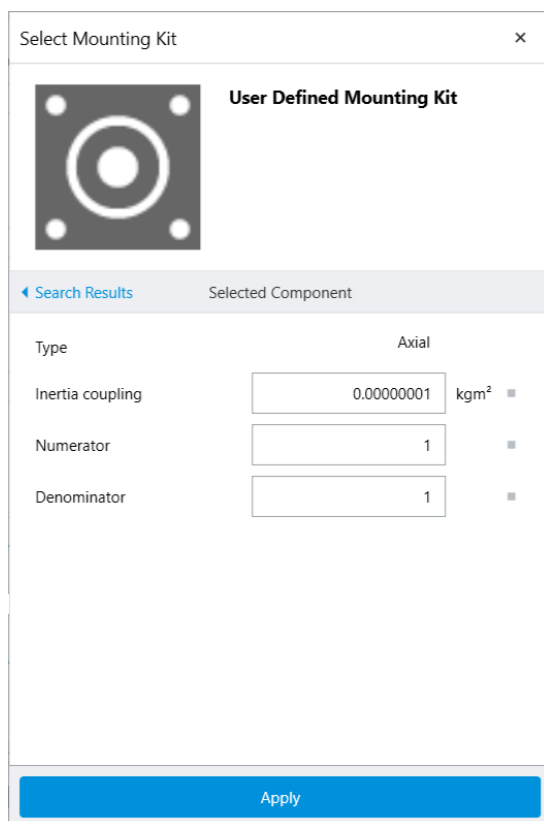
No mounting kit configured

- Select the required mounting kit. As per this sample application **User Defined Mounting Kit** is selected.

Configuration of CMMT Drive Parameterization Port using FAS



- After selecting the Mounting kit enter the required configuration details. Click **Apply** to add mounting kit to project.



- The Mounting Kit tab will look after configuration as shown below.

 Mounting Kit	User Defined Mounting Kit	Type	Gear Ratio
		Axial	1:1

Step 4 – Device Settings

1. Go to **Device Settings** to do device configuration.

←
First setup



Drive Configurati...



Device Settings



Fieldbus



Application Data



Hardware Switches



Homing Method



Software Limits

Enable Servo Drive

Activation via I/O and fieldbus (0)

Supply Voltage

Mains voltage 230.00 V

External Brake Resistor

External Brake Resistor None (0)

Resistivity 100.00 Ω

Power dissipation 21.00 W

Back
Next
Finish

2. Select **Enable Servo Drive** method as **I/O and fieldbus** from drop down option **Activation Via**. Click **Next** button for next level configuration for next level configuration.

First setup

Drive Configurati... | **Device Settings** | Fieldbus | Application Data | Hardware Switches | Homing Method | Software Limits

Enable Servo Drive

Activation via: I/O and fieldbus (0) | I/O (1) | Fieldbus (2) | I/O and Plug-in (3) | Plug-in (4)

Supply Voltage

Mains voltage: 230.00 V

External Brake Resistor

External Brake Resistor: None (0)

Resistivity: 100.00 Ω

Power dissipation: 21.00 W

Back | **Next** | Finish

Select Enable Servo Drive Method as I/O and fieldbus

Step 5 – Fieldbus Configuration

1. In **Fieldbus Configuration** select **EtherNet/IP - Modbus** from **RTE Configuration** drop-down list.

Fieldbus Configuration

RTE Configuration (user defined): PROFINET (1) | PROFINET (1) | EtherCAT (2) | EtherNet/IP - ModbusTCP (3)

Select Fieldbus as EtherNet/IP - Modbus (3)

2. In **EtherNet/IP – Modbus Configuration** select **Telegram (111)** from **Telegram Selection** drop-down list.

EtherNet/IP - ModbusTCP configuration

Telegram selection

Activate DHCP

IP address

Subnet mask

Gateway address

Select Telegram as Telegram (111)

Telegram (1)

Telegram (1)

Telegram (9)

Telegram (102)

Telegram (111)

255 . 255 . 255 . 0

0 . 0 . 0 . 0

- Enter the IP address & Subnet mask for **EtherNet/IP – ModbusTCP Configuration** (the same will be the IP address of the **EtherNet/IP interface of CMMT**) as shown in below image. This communication interface is used to establish connection with PLC to receive control command & to send signal feedbacks. Click **Next** button for next level configuration.

EtherNet/IP - ModbusTCP configuration

Telegram selection

Activate DHCP

IP address

Subnet mask

Gateway address

Telegram (111)

☐ Active

192 . 168 . 1 . 5

255 . 255 . 255 . 0

0 . 0 . 0 . 0

Enter IP Address & Subnet Mask Values

Back

Next

Finish

Step 6 – Enter Inertia Value

- Go to **Application Data** to Specify inertia value of user application. This parameter value will vary depend on application type. Click **Next** button for next level configuration.

First setup

Drive Configurati... Device Settings Fieldbus **Application Data** Hardware Switches Homing Method Software Limits

Application Data

Axis mass	0.00	kg	■
Application mass	0.00	kg	■
Total mass	0.00	kg	■

Enter Inertia Value

Load Compensation

Workpiece mass	0.00	kg	■
Offset torque	0.00	Nm	■

Mounting Position

Back **Next** Finish

Step 7 – Hardware Switches Configuration

- Specify **Hardware Switches Configuration** of user application as per requirement. Click **Next** button for next level configuration.

First setup

Drive Configurati... Device Settings Fieldbus Application Data **Hardware Switches** Homing Method Software Limits

Hardware Switches

Reference switch configuration Deactivated (0)

Limit switches configuration Not used (1)

Set Hardware Switches Configuration

Back **Next** Finish

Step 8 - Homing Method

- Select the **Homing Method**. As per this sample application “**Actual Position (37)**” mode is selected, (user can select the homing mode depending on the application requirement) click **Next** button for next level of configuration.

The screenshot shows the 'First setup' configuration window. At the top, there is a navigation bar with a back arrow and the title 'First setup'. Below this, there is a row of seven tabs: 'Drive Configurati...', 'Device Settings', 'Fieldbus', 'Application Data', 'Hardware Switches', 'Homing Method', and 'Software Limits'. The 'Homing Method' tab is selected and highlighted with a red box. Below the tabs, the 'Homing Method' configuration section is displayed. It includes a 'Method' dropdown menu with 'Actual position (37)' selected, a 'Nominal current limit value scaling factor' set to '0.80', and two checkboxes: 'Move to axis zero point after homing' (checked) and 'Activation Save zero offset' (unchecked). At the bottom of the window, there are three buttons: 'Back', 'Next', and 'Finish'. The 'Next' button is highlighted with a red box.

First setup

Drive Configurati... Device Settings Fieldbus Application Data Hardware Switches **Homing Method** Software Limits

Homing Method

Method: Actual position (37)

Nominal current limit value scaling factor: 0.80

Move to axis zero point after homing: ☒ Active

Activation Save zero offset: ☐ Active

Back Next Finish

Step 9 – Software Limits Configuration

- Here user can enable the maximum & minimum values of **Software Limits** as per application requirements. Finally click **“Finish”** button to complete the configuration.

First setup

Drive Configurati... Device Settings Fieldbus Application Data Hardware Switches Homing Method **Software Limits**

Software Limits

Axis zero point offset 3.00 mm

Software limit positions active ☒ Active

Negative software limit position -3.00 mm

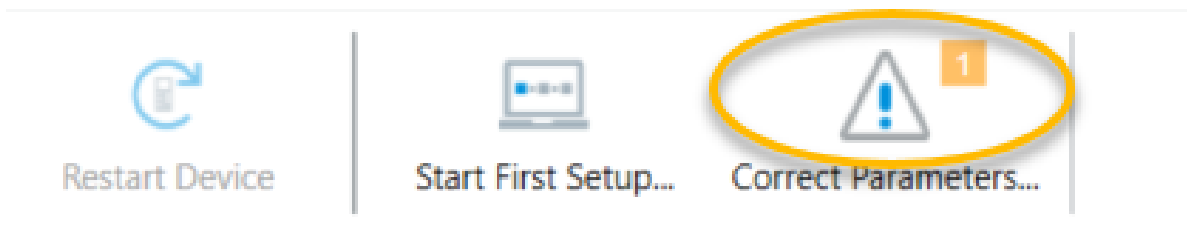
Positive software limit position 97.00 mm

Set Software Limit Configuration

Back Next **Finish**

Step 10 – Correct Parameters

- At this stage all the fundamental configurations of drive system are done. If there are any corrections in the parameter software tool will indicate number of **“Correction Parameters”** (Right most top corner in FAS tool) as shown in the below picture. So to correct parameter value click the **“Correction parameters”** button and click **Apply** button to apply system generated recommended values



Note

- This step might not be required as per user application data.

2.2.3 Rotary Axis Configuration

This chapter describes the parameterization of the CMMT drive for a rotary axis to move 1000 revolution forward and reverse with 1000 RPM base speed & current position as homing mode.



Note

- If the user is looking for Linear Axis configuration skip this chapter and jump to [Chapter 2.2.2](#)

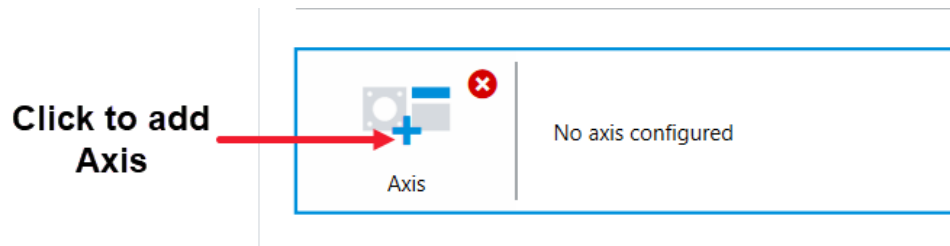
Step 1

- Please follow the **Step 1** for configuration described in [Chapter 2.2.2](#).

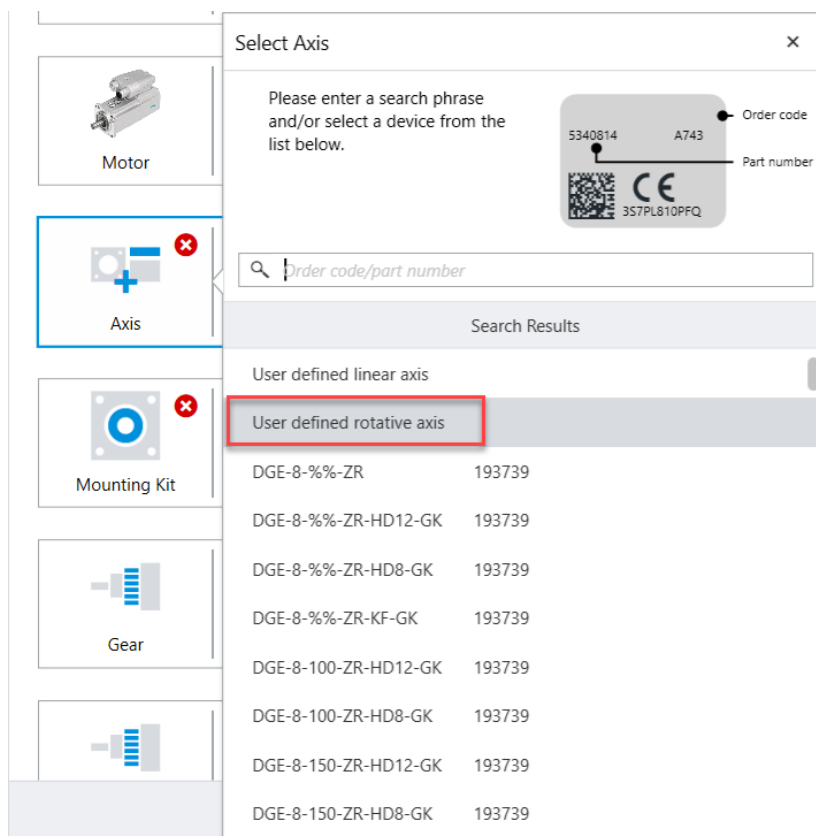
Step 2 – Axis Configuration

This step describes about the axis configuration. As specified initially here axis configuration is done for 1000 rotation in forward & reverse. This configuration will change based on user application type.

1. Click **Axis** to add the required axis to project.



1. Select the required axis from the available list. As per this sample application **User Defined Rotative Axis** is selected.



2. After selecting the axis enter the required configuration details. Click **Apply** to add the axis to project.

Select Axis



User defined rotative axis

Search Results

Selected Component

Actual user unit

Rev [r, r/min, ...] (3)

Motion

Rotative

Unlimited axis

☐ Active

Position range

1000.00 rev

Design axis

Single axis (0)

Apply

3. The Axis tab will look as shown below after configuration.



Axis

User defined rotative axis

Position Range

1000.00 rev

Step 3

- Please follow the **Step 3 to Step 8** for configuration described in [Chapter 2.2.2](#).

Step 4 – Software Limits Configuration

First setup

Drive Configurati... Device Settings Fieldbus Application Data Hardware Switches Homing Method **Software Limits**

Software Limits

Axis zero point offset 0.03 rev

Software limit positions active ☒ Active

Negative software limit position -0.03 rev

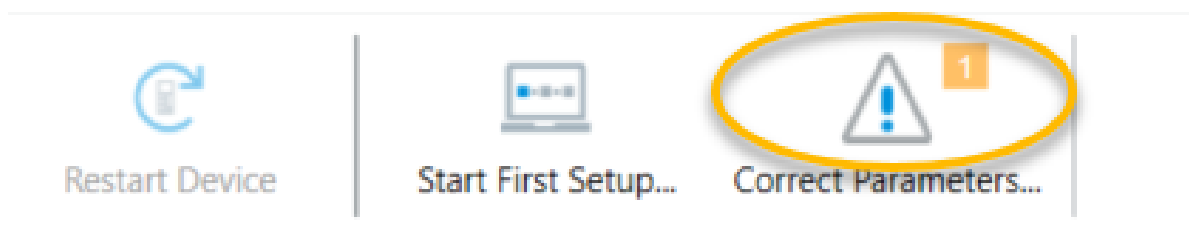
Positive software limit position 999.97 rev

Set Software Limit Configuration

Back Next Finish

Step 5 – Correct Parameters

- At this stage all the fundamental configurations of drive system are done. If there are any corrections in the parameter software tool will indicate number of **“Correction Parameters”** (Right most top corner in FAS tool) as shown in the below picture. So to correct parameter value click the **“Correction parameters”** button and click **Apply** button to apply system generated recommended values



Note

- This step might not be required as per user application data.

2.2.4 CMMT Drive Parameterization for Integrate with PLC

This chapter describes:

- Parameterization of CMMT drive to establish communication with PLC through Telegram 111.
- Also it describes some additional parameters like Factor group, Reference values, EtherNet/IP interface, User defined limits.

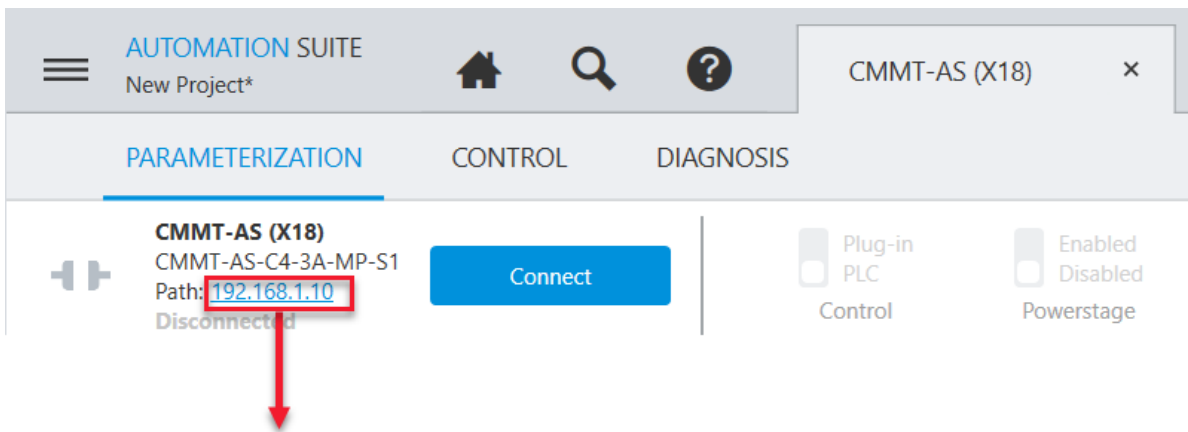
Step 1



Note

- Follow this step when drive is available for online connection. If drive is not available simply skip this step and jump to next step.

1. Click the IP Address **"Path"** from tool bar.

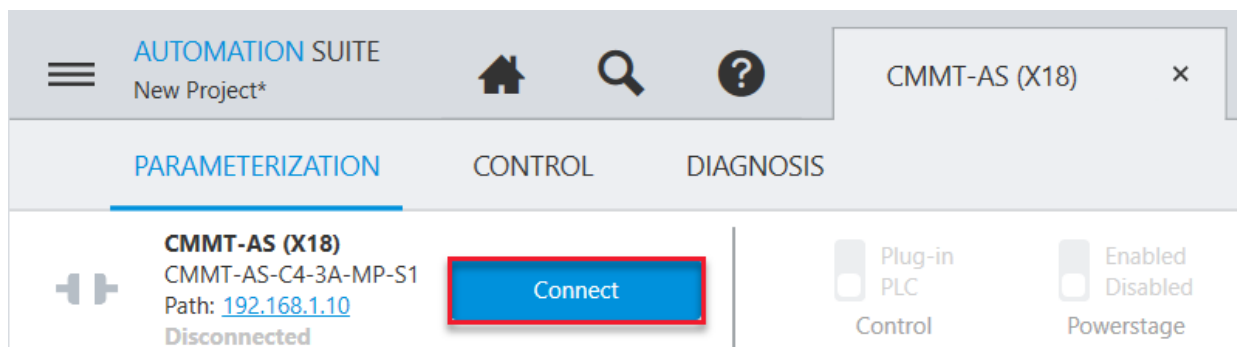


Click to Change Path

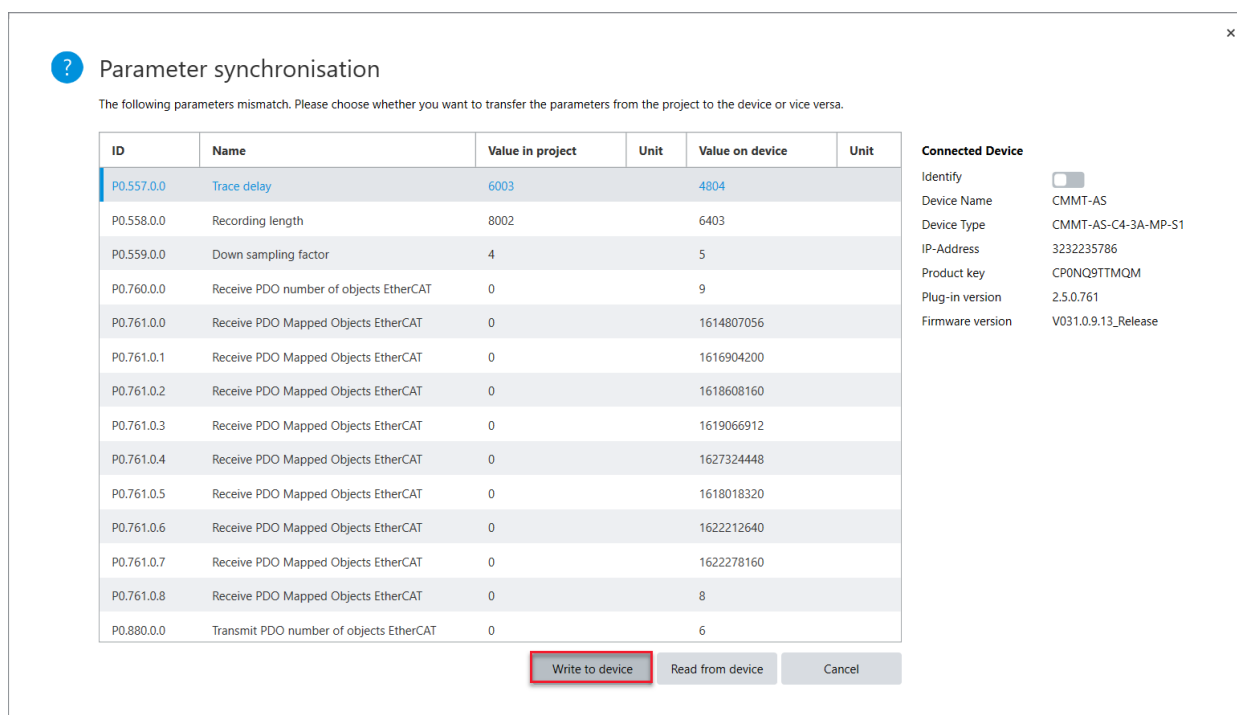
2. Select the correct drive based on IP address & Click **"Apply"** button to assign path address.



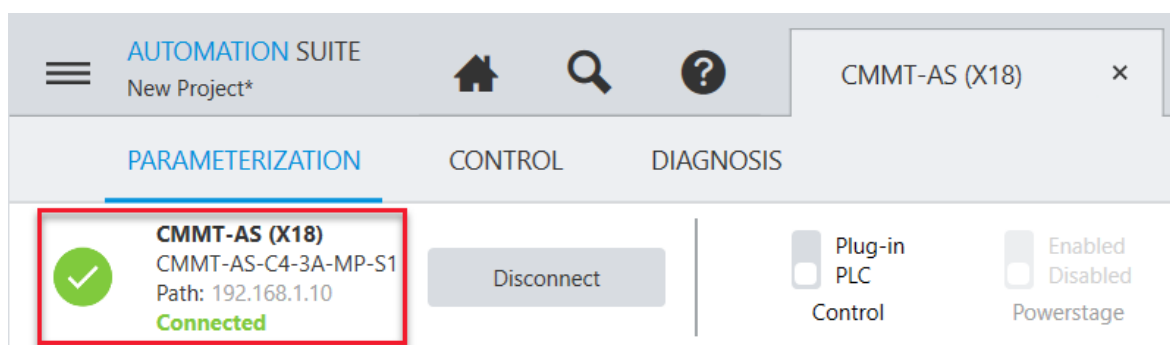
3. Click on **"Connect"** button to establish communication .



- Click **Write to device** button on **Parameter Synchronisation** window.

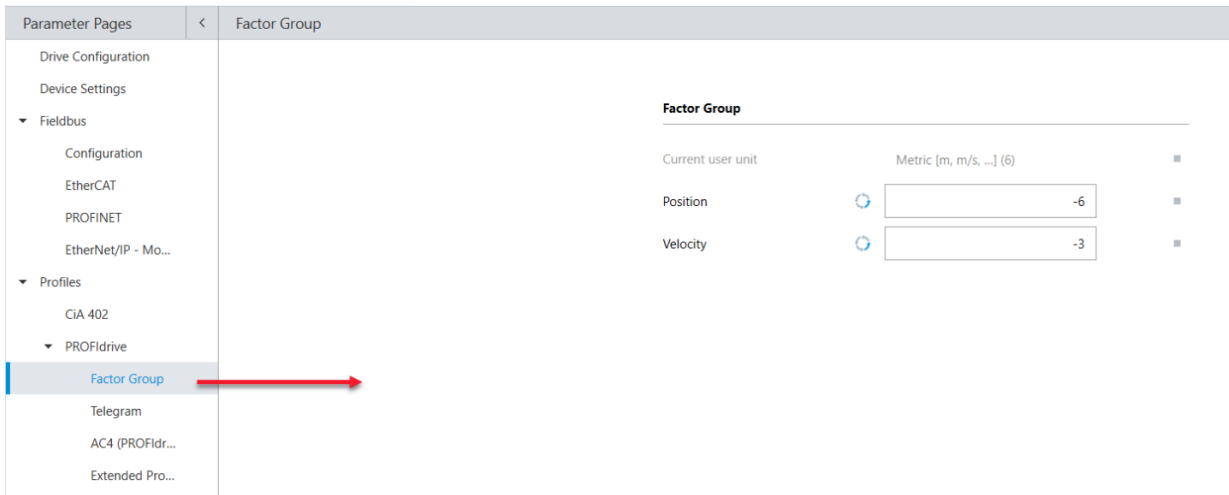


- The status of the drive changes from **Disconnected** to **Connected**.



Step 2 – Entering Factor Group Value from Position and Velocity

- Go to **Parameter Pages > Profiles > PROFIdrive > Factor Group** as shown below.



1. Example - 1 PLC scale value is calculated below for move the linear axis 10 mm in 1 m/s speed.

Position Factor (-6) = $10^{-6} = 0.000001 = 1\mu\text{m}$ Then For 10 mm = 10000 Increment value in PLC
 Velocity Factor (-3) = $10^{-3} = 0.001 = 1\text{mm/s}$ Then for 1 m/s = 1000 Increment value in PLC

2. Example - 2 PLC scale value is calculated below for move the rotary axis 5 Rev in 100 RPM speed.

Position Factor (-6) = $10^{-6} = 0.000001 \text{ Rev}$ Then For 5 Rev = 5000000 Increment value in PLC
 Velocity Factor (-3) = $10^{-3} = 0.001 \text{ RPM}$ Then for 100 RPM = 100000 Increment value in PLC

Factor Group

Current user unit		Metric [m, m/s, ...] (6)	■
Position		-6	■
Velocity		-3	■



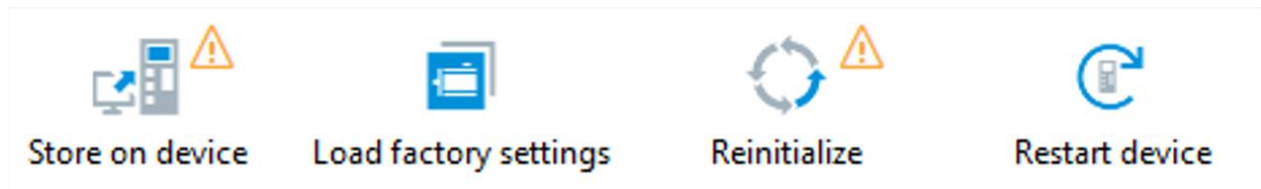
Note

For more details about fieldbus scaling refer chapter name “Scaling of internal units for the fieldbus” in user manual.

- Different user unit can’t be set for Position, Velocity, Acceleration & Jerk.

Step 3 – Store On Device

- Click the “**Store on device**” button to store all the parameters on drive memory.



Note

- The triangle warning symbol on both buttons indicates action required.
- If modified parameters are not saved on the drive, all modified parameters will automatically restore the drives with previous parameter values when the power restart.
- Its mandatory to reinitialize the drive for some parameter modification. If drive is not reinitialized the drive parameter remains same as like previous configuration.



Information

Hardware configuration of CMMT-AS controller described in this chapters are only for this application note purpose.

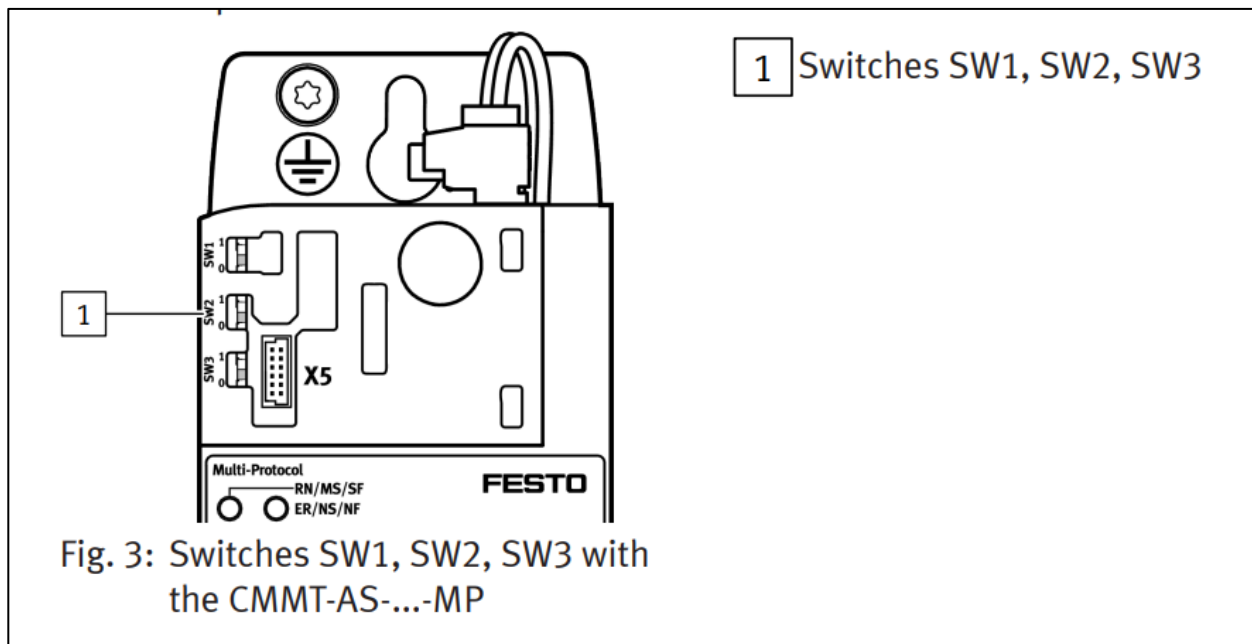
User must configure CMMT-AS controller as per their application requirement and also check Protocols supported by the firmware version. For further clarification please refer the CMMT user manual.

2.3 CMMT-AS Servo Drive Fieldbus Configuration

The CMMT-AS-...-MP product variant supports several bus protocols. The following options are available for setting the protocol:

- Automatic detection by the CMMT-AS-...-MP
- Configuration in the CMMT-AS plug-in

- Direct specification via SW1 to SW3



2.3.1 Switch Setting for Bus Protocol

Protocol	Size 3	Size 2	Size 1
Auto (detection or parameterisation)	0	0	0
PROFINET	0	0	1
EtherCAT	0	1	0
EtherNet/IP	0	1	1

The switches can be adjusted with a small slotted head screwdriver.
The switch position is evaluated once when the device is started.



Note

- For **Configuration in the CMMT-AS plug-in** all switches should be in 0 position.
- User can use either method to configure fieldbus protocol for CMMT-AS-...-MP.
- If the user select **EtherNet/IP** protocol via switch setting; must select **EtherNet/IP or ModbusTCP** as **RTE Configuration** (refer [Chapter 2.2.1 Step 5](#)).

3 Starting a Project

3.1 Create New Project File & Configurations of IQ-R Series PLC in Gx Works 3

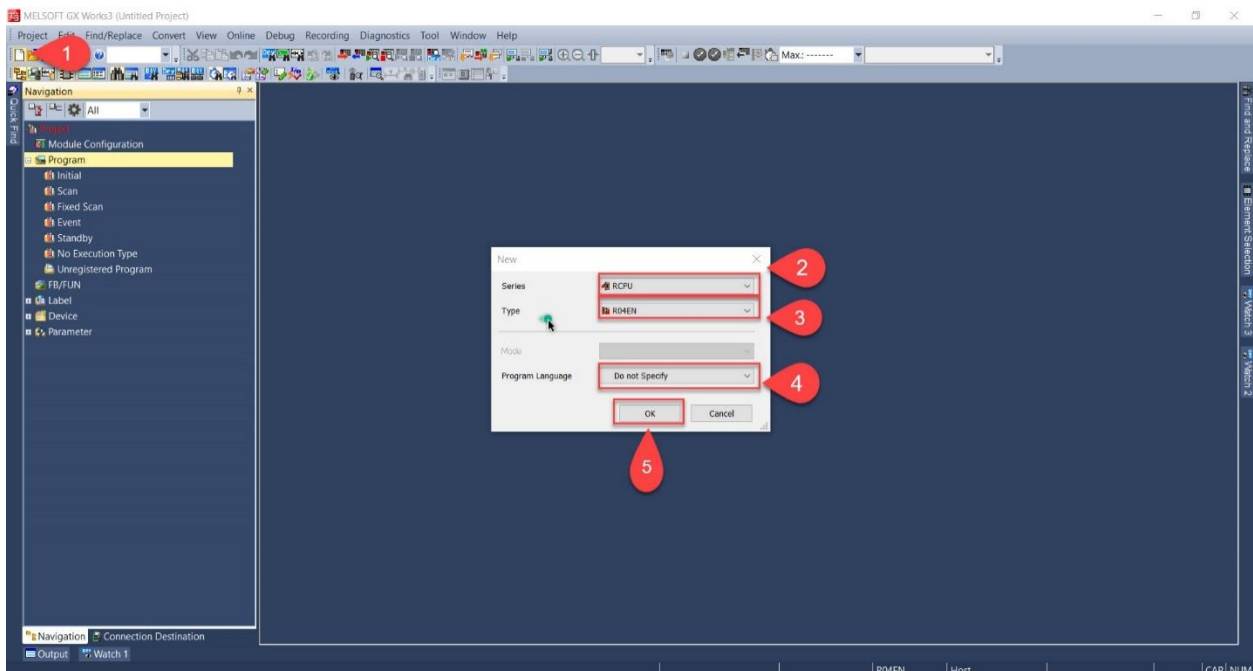
This chapter also covers how to develop projects step-by-step, how to set up communication settings, and how to configure software for things like Modbus TCP Simple CPU Communication settings, and adding libraries in the Gx Works 3 programming environment.



Note

- Ignore Chapter 3.1 in case integration of CMMT drive with existing running project file and jump to [Chapter 3.3](#) to set Simple CPU Communication.

Step – 1



Open MELSOFT **GX Works3** development software and start a new project by clicking

1. [Project] => [New]
Select **"New Project"** to create a new project.
1. Select **"RCP"** from **"Series"** drop down option.
2. Select your CPU model from **"Type"** drop down option.
3. Select **"Do not Specify"** as programming language from **"Programming Language"** drop down option.
Or user can select preferred programming language,
4. Confirm your selection with the button **"OK"**.

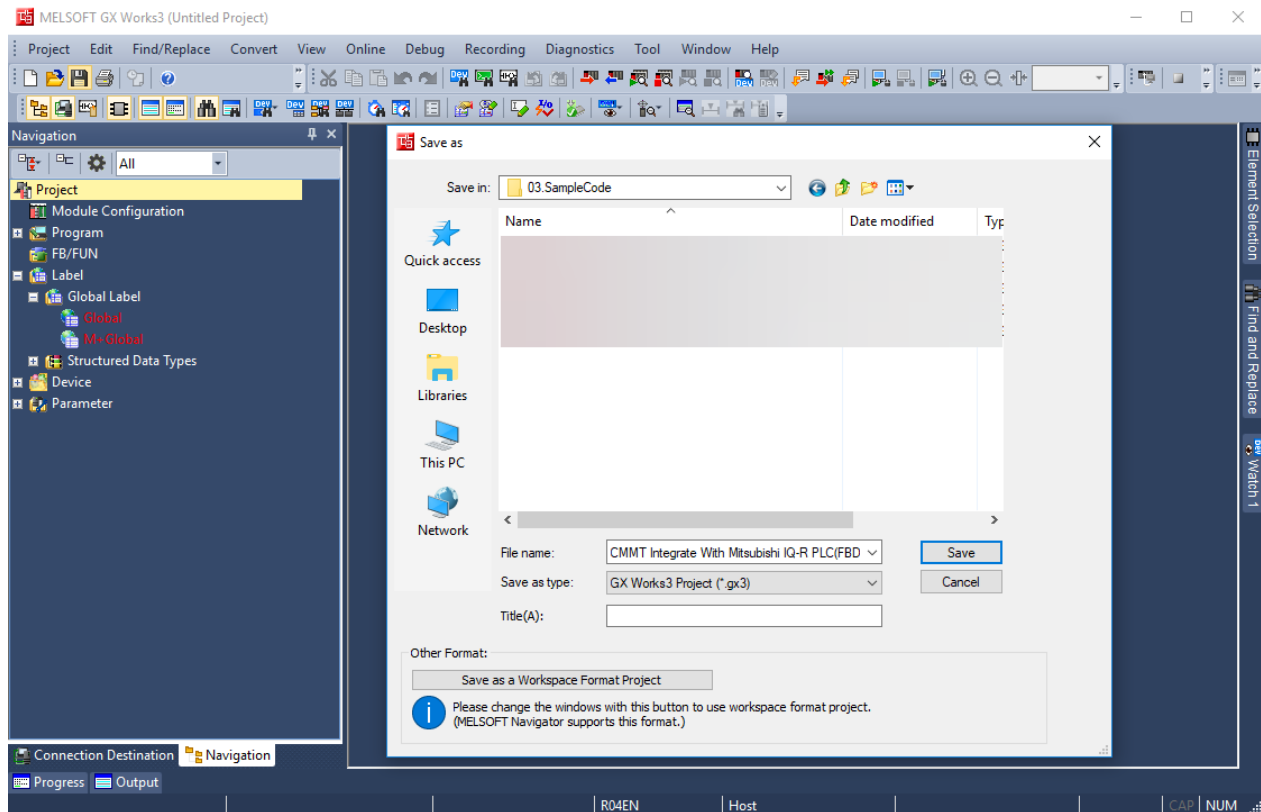


Note

- 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.
- In this App note as **example** we have considered **R04EN** CPU.

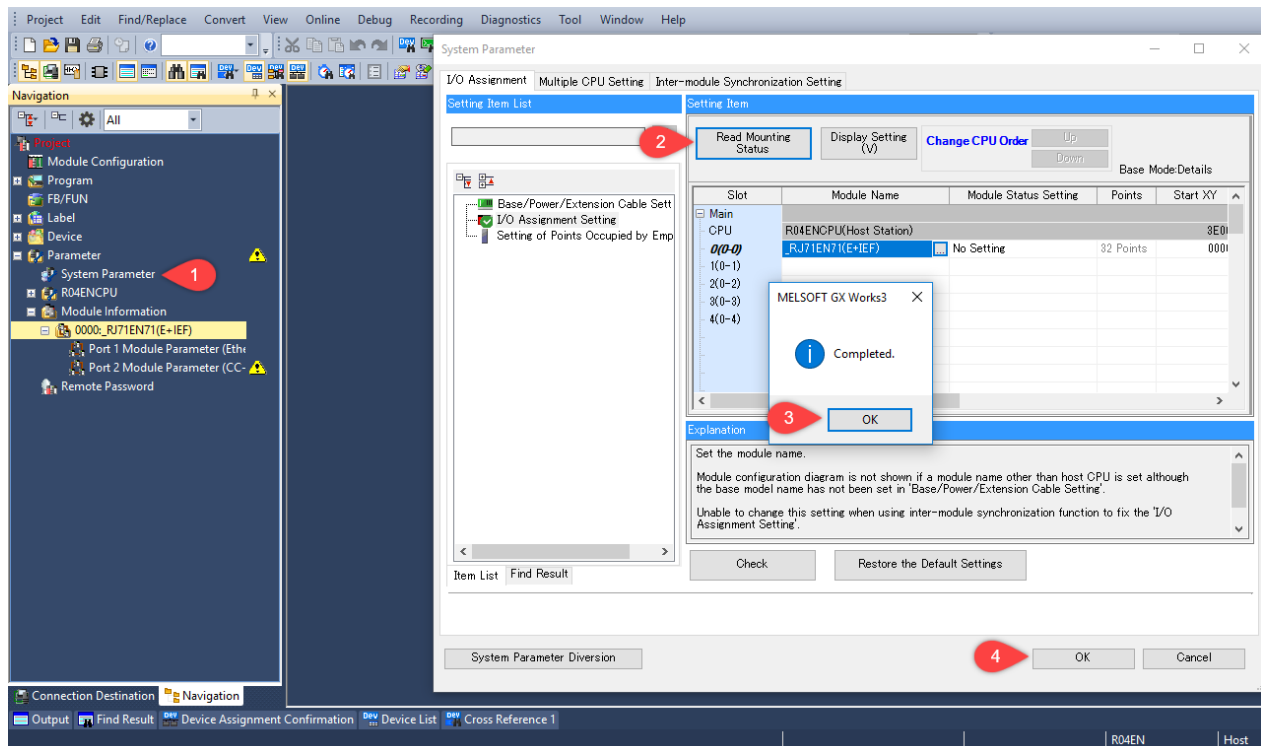
Step – 2

Save the Project



Save the project as desired.

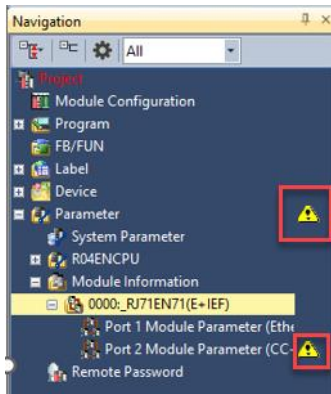
Step – 3



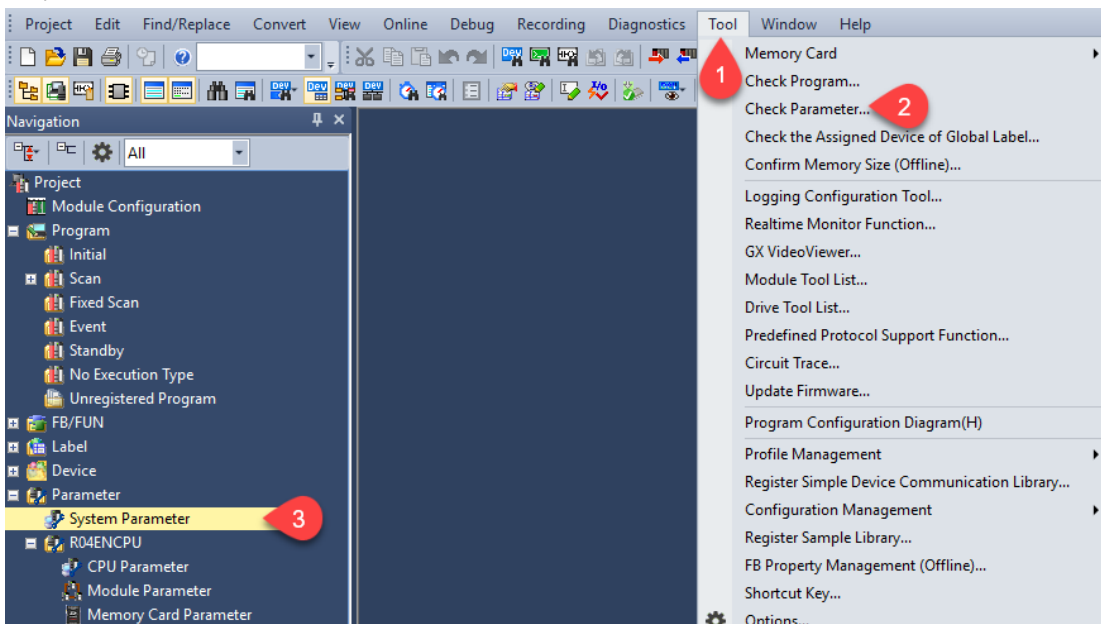
1. Click on “System Parameter” under Parameter tree.
2. Click on **“Read Mounting Status”** to get the Hardware Module configuration.
3. Click on **“OK”** in Completed pop up.
4. Press **“OK”** to complete the process.

Step – 4

Starting a Project



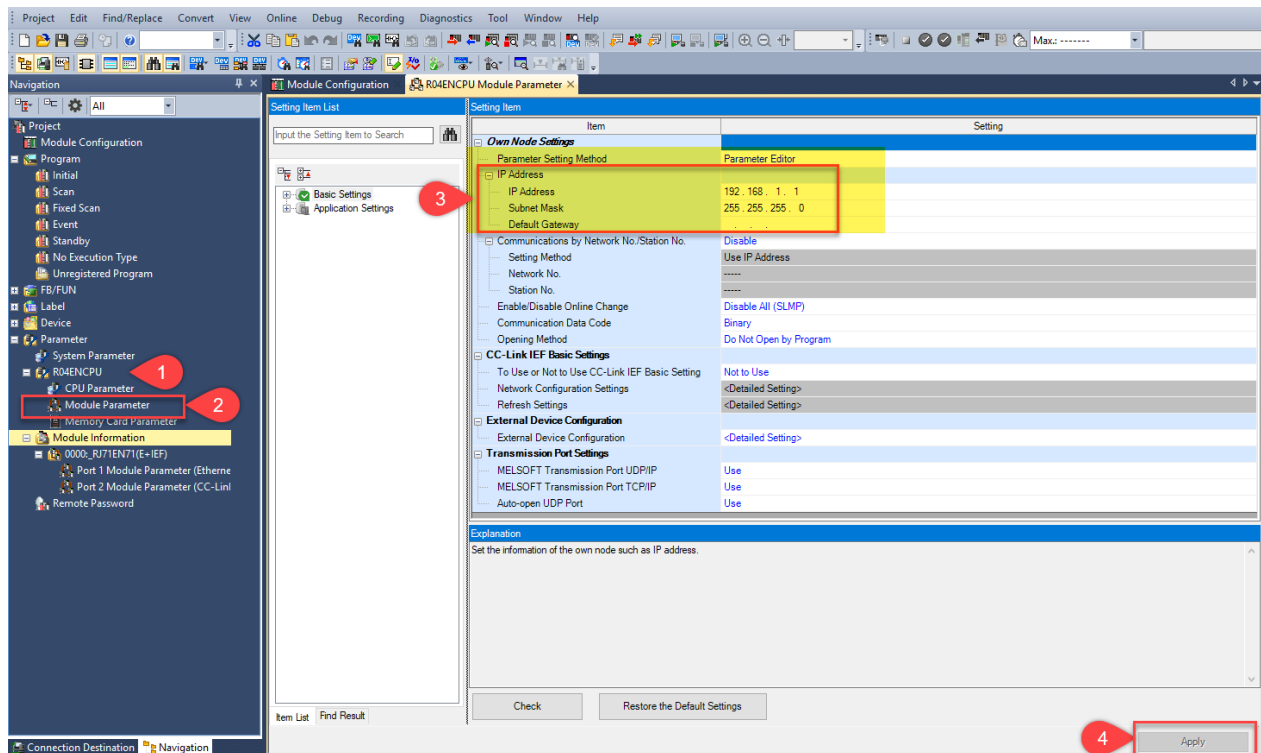
In case warning symbol appears as shown in above image, then follow the steps shown in below image to check for parameter issues.



1. Click on **Tool**.
2. Click **Check Parameter**.
3. Check whether Warning symbol cleared.

3.2 Ethernet port parameters

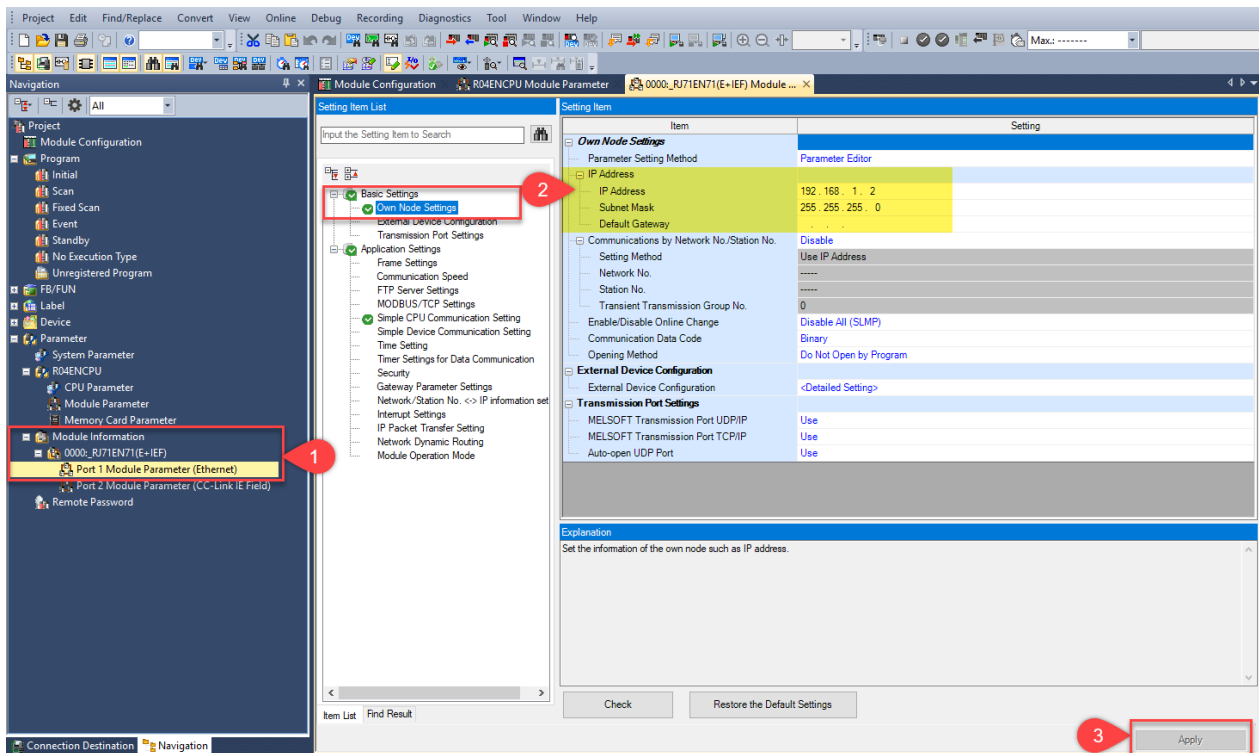
3.2.1 Procedure to set IP address of PLC



1. Expand **"R04ENCPU"** from **"Parameter"** option.
2. Double click **"Module Parameter"** option.
3. Enter the **"IP Address & Subnet Mask"** of the PLC as per requirement. as example shown in above image.
4. Click **"Apply"** button to confirm the configuration.

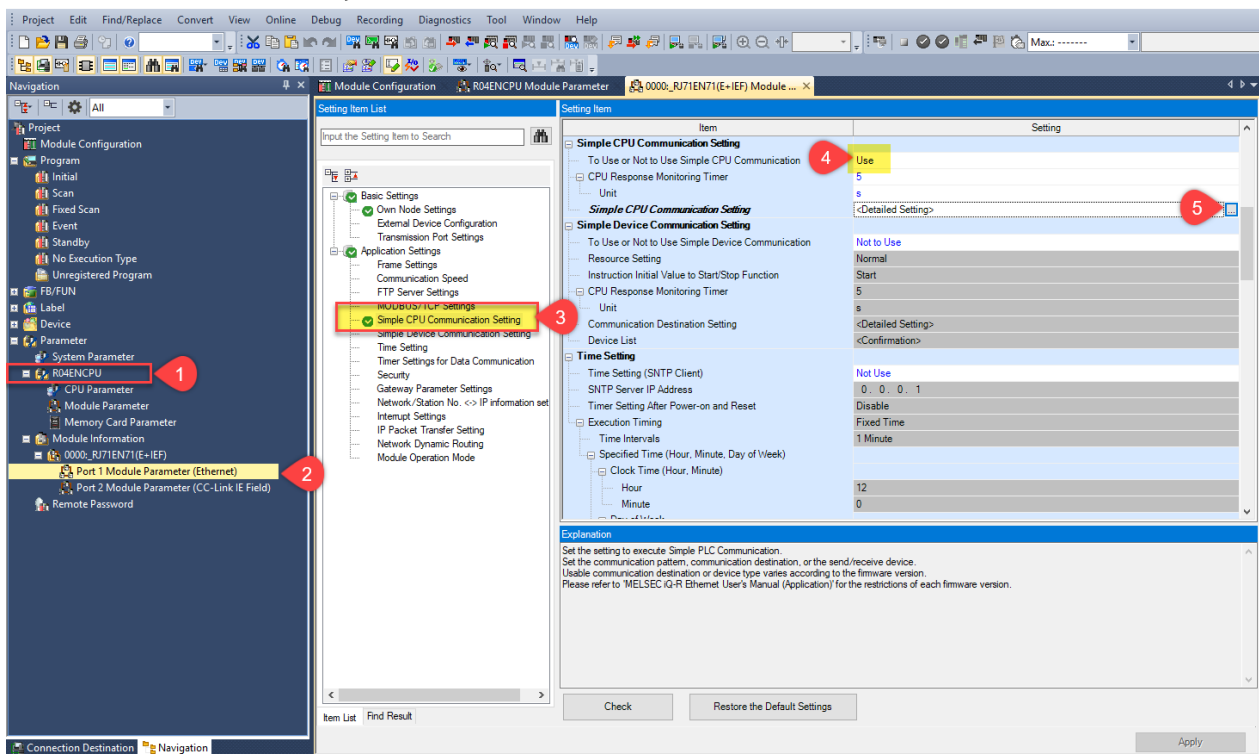
3.3 Modbus TCP Simple CPU Communication Settings

3.3.1 Procedure to set IP address for RJ71EN71



1. Under Module information expand RJ71EN71(E+IEF) and click on **“Port1 Module Parameter (Ethernet)”**.
2. Enter the **“IP Address & Subnet Mask”** as example shown in above image.
3. Click **“Apply”** button to confirm the configuration.

3.3.2 Procedure to use Simple CPU Communication



1. Click on **“Parameter”**, Expand **“Module Information”**.
2. Double click on **“Port 1 Module Parameter (Ethernet)”** of RJ71EN71(E+IEF) under Module information.
3. Expand **“Application Settings”** and click on **“Simple Communication Setting”**.

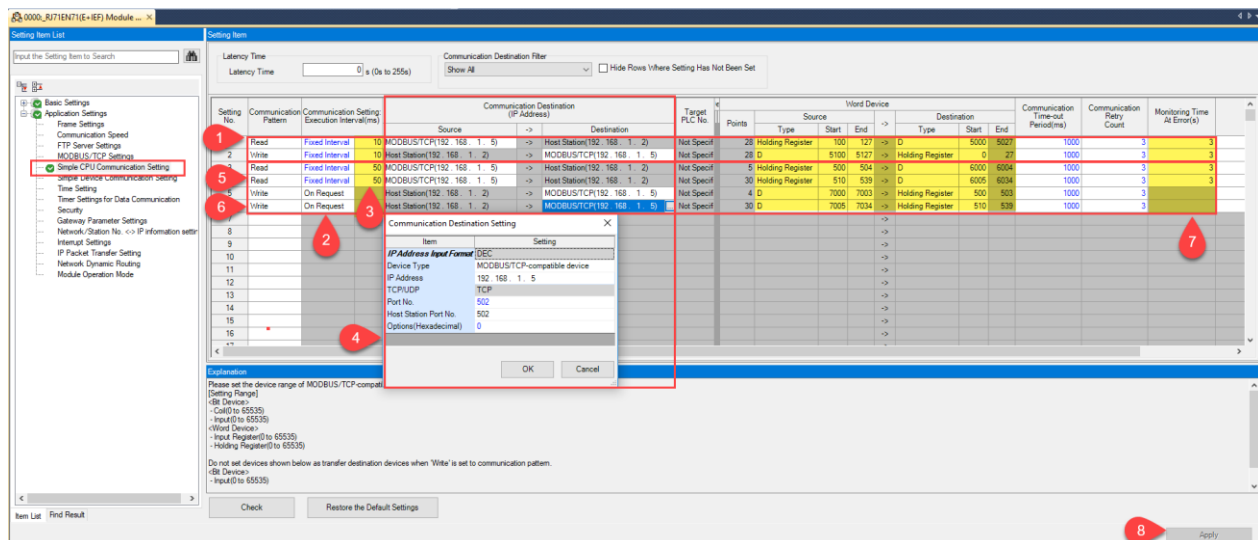
4. Select **"Use"** in To Use or Not to Use Simple CPU Communication.
5. Click on **"Detail Settings"** in Simple CPU Communication setting.



Warning

"Simple CPU communication" is possible only in **"R..EN"** CPU (**"EN"** is mandatory, only **Port 1** of **RJ71EN71** supports Simple CPU communication).

3.3.3 Simple CPU Communication settings



1. Select **"Write/Read"** in Communication Pattern.
2. To read Process Data, Single Parameter Data and Write Process Data set **"Fixed Interval"** in Communication Setting.
To Write Single Parameter Data set **"ON Request"** in Communication Setting.
3. Set **"10"** as Execution Interval(ms) for Process Data and
Set **"50"** as Execution Interval(ms) for Single Parameter Data.
4. Click on **"Set"** in Source & Destination column of Communication Destination (IP Address).
 - In Device Type select **"MODBUS TCP Compatible Device"**.
 - In **"IP Address"** set **CMMT Ethernet/IP Address**. (Refer [Chapter 2.2.2 Step – 5, Point 3](#))
 - In **"Host Station Port No"** Set **502**.
 - Click **OK**.
5. Create one Read communication pattern for - **Process Data**
 - Under **"Read"** setting
→ Word Device → Source Type → select **"Holding Register"** and set Modbus **Start Address – 100** and **End Address – 127**.
– Refer: Modbus Address table given below.
→ Word Device → Destination Type → select Register type (**D/W/R/SW/SD**) and set Modbus **Start Address – 5000** or user can define any address as required.
 - Create two Read communication pattern for - **Single Parameter Access**
 - Under **"Read"** setting
→ Word Device → Source Type → select **"Holding Register"** and set Modbus **Start Address – 500** and **End Address – 504**.
– Refer: Single Parameter Access table given below
→ Word Device → Destination Type → select Register type (**D/W/R/SW/SD**) and set Modbus **Start Address – 6000** or user can define any address as required.
→ Word Device → Source Type → select **"Holding Register"** and set Modbus **Start Address – 510** and **End Address – 539**.
→ Word Device → Destination Type → select Register type (**D/W/R/SW/SD**) and set Modbus **Start Address – 6005** or user can define any address as required.
– Refer: Single Parameter Access table given below
6. Create one Write communication pattern for - **Process Data**
 - Under **"Write"** setting
→ Word Device → Source Type → select Register type (**D/W/R/SW/SD**) and set user defined **Start Address – 5100** and **End Address – 5127** or user can define any address as required.

- > Word Device -> Destination Type -> select **"Holding Register"** and set Modbus **Start Address – 0**.
 - Refer: Modbus Address table given below.
- Create two Write communication pattern for - **Single Parameter Access**
 - > Word Device -> Source Type -> select Register type **(D/W/R/SW/SD)** and set user defined **Start Address - 7000** and **End Address – 7003** or user can define any address as required.
 - > Word Device -> Destination Type -> select **"Holding Register"** and set Modbus **Start Address – 500**.
 - > Word Device -> Source Type -> select Register type **(D/W/R/SW/SD)** and set user defined **Start Address - 7005** and **End Address – 7034** or user can define any address as required.
 - > Word Device -> Destination Type -> select **"Holding Register"** and set Modbus **Start Address – 510**.
 - Refer: Single Parameter Access table given below
- 7. Change the Monitoring Time At Error(s) from 30 to 3. (Refer Note below)
- 8. Click **"Apply"** to save setting.

Modbus Address

Modbus TCP offers the same process data as that available in the drive profile of EtherNet/IP.

Process data	Description
Byte 0 ... 23 (24 bytes)	Telegrams 1, 102 or 111
Byte 24 ... 55 (32 bytes)	Additional telegram 910 (Extended Process Data, EPD)

From the point of view of the controller:

- Input-Data: register 100 ... 127
- Output-Data: register 0 ... 27

Single Parameter Access

Single parameters can be read or written via registers 500 to 504 and 510. The PNUs with subindex are used.

Register	Length	Access	Name	Description
500	1	r/w	PNU	PNU for single parameter access
501	1	r/w	PNU Subindex	Subindex of the PNU
502	1	r/w	Number of Elements	Number of items in single parameter access
503	1	r/w	Exec	Perform single parameter access, see the following section: - 1: Read - 2: Write - 4: Error (request failed) - 16: Done (request successfully completed)
504	1	r	Data Length	Length of read or written data in bytes
510	128	r/w	Data	Written or read data



Note: Monitoring Time At Error(s)

Depending on the PLC and number of device connected to the PLC, the Simple CPU Communication will monitor the ERROR for specified time.

The PLC will establish the connection in less time if the Monitoring Error time is reduced; otherwise, it may take longer to re-establish the connection.

3.3.4 Simple CPU Communication status

Setting Item List		Setting Item													
Input the Setting Item to Search		Latency Time		Communication Destination Filter											
		Latency Time		Show All											
				Hide Rows Where Setting Has Not Been Set											
Setting No.	Communication Pattern	Communication Setting Execution Interval(ms)	Communication Destination (IP Address)		Target PLC No.	Points	Type	Source	End	Destination	Start	End	Communication Time-out Period(ms)	Communication Retry Count	Monitoring Time At Error(s)
1	Read	Fixed Interval	10	MODBUS/TCP(192.168.1.5)	Host Station(192.168.1.2)	Not Specified	28	Holding Register	100	127	D	5000	5007	1000	3
2	Write	Fixed Interval	10	Host Station(192.168.1.2)	MODBUS/TCP(192.168.1.5)	Not Specified	28	D	5100	5127	Holding Register	0	27	1000	3
3	Read	Fixed Interval	50	MODBUS/TCP(192.168.1.5)	Host Station(192.168.1.2)	Not Specified	5	Holding Register	500	504	D	6000	6004	1000	3
4	Read	Fixed Interval	50	MODBUS/TCP(192.168.1.5)	Host Station(192.168.1.2)	Not Specified	30	Holding Register	510	539	D	6005	6034	1000	3
5	Write	On Request		Host Station(192.168.1.2)	MODBUS/TCP(192.168.1.5)	Not Specified	4	D	7000	7003	Holding Register	500	503	1000	3
6	Write	On Request		Host Station(192.168.1.2)	MODBUS/TCP(192.168.1.5)	Not Specified	30	D	7005	7034	Holding Register	510	539	1000	3
7															

Device Name		F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Current Value	String
U0\G721936		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3	..
U0\G721937		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3	..
U0\G721938		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3	..
U0\G721939		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	3	..
U0\G721940		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	..
U0\G721941		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	..
U0\G721942		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	..
U0\G721943		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	..

Fig: Device Memory

Based on the images above, the following information are inferred

- The yellow highlighted communications are **fixed interval**, the current value will be 3 once communication is successfully established.
- The green highlighted communications are **on request**, the current value will remain 2 until the communication request bit is triggered, at which point communication will be established.

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

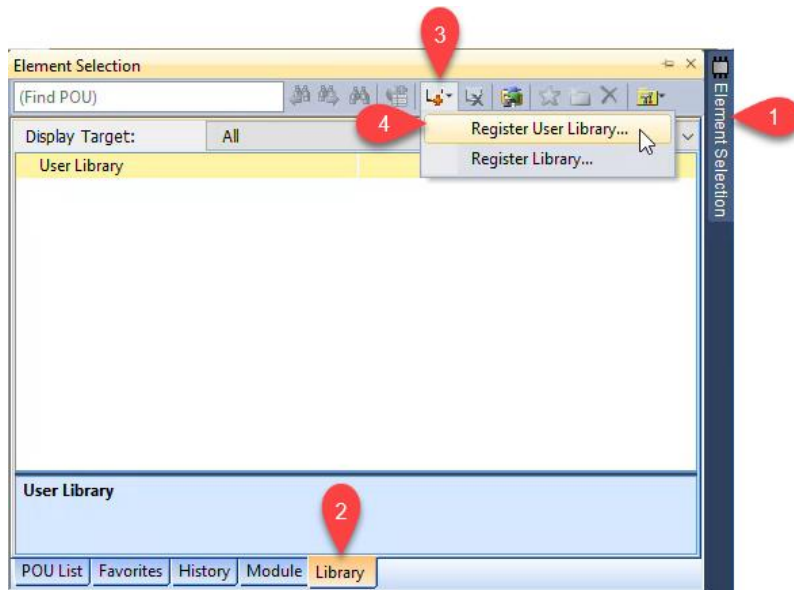
The simple CPU communication status can be checked with the storage status of the corresponding setting number in the following buffer memory areas.

Item	Address	Remarks
Request to start communication at request	Un\G721896 to Un\G721899 Un\G1247300 to Un\G1247327	- Setting No.1: Un\G721896.0 ... - Setting No.64: Un\G721899.F - Setting No.65: Un\G1247300.0 ... - Setting No.512: Un\G1247327.F
Ready	Un\G721912 to Un\G721915 Un\G1247412 to Un\G1247439	- Setting No.1: Un\G721912.0 ... - Setting No.64: Un\G721915.F - Setting No.65: Un\G1247412.0 ... - Setting No.512: Un\G1247439.F
Simple CPU communication status	0H: Unset 1H: Preparing 2H: Waiting for the request 3H: Communicating 4H: Communication stop 5H: Retry being executed 6H: Monitoring at error AH: Communications impossible	Un\G721936 to Un\G721999 Un\G1247460 to Un\G1247907
Simple CPU communication error code	Un\G722000 to Un\G722063 Un\G1247908 to Un\G1248355	- Setting No.1: Un\G722000 ... - Setting No.64: Un\G722063 - Setting No.65: Un\G1247908 ... - Setting No.512: Un\G1248355

Fig: Simple Communication Status Device No, taken from Simple CPU Communication Manual.

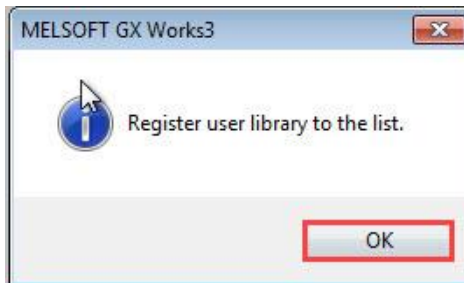
3.4 Adding Library to the project

3.4.1 Import PTP_Drives_Festo_MB_IQ-R in FB/Fun under program Step –1



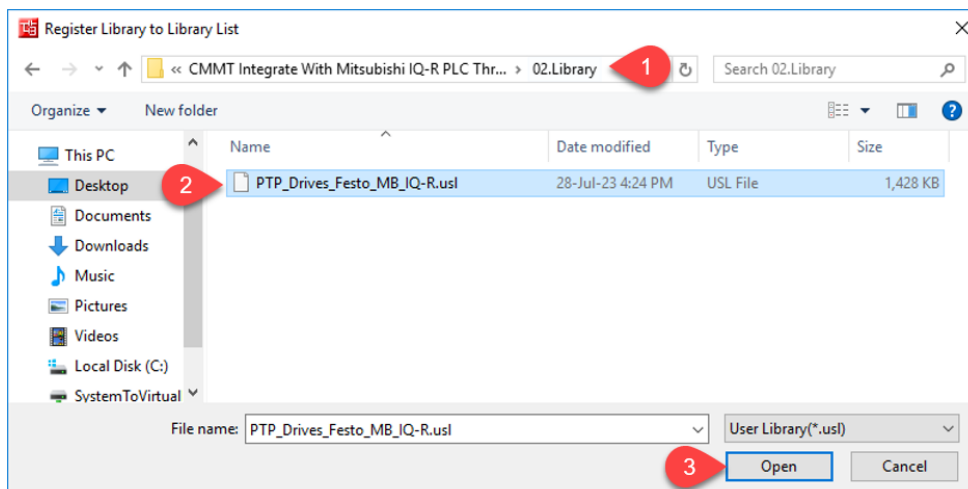
1. Open the “**Element Selection**” window.
2. Click on the “**Library**” tab.
3. Select on “**Register to Library List**”
4. Click on “**Register User Library**”

Step – 2



- Click “**OK**” button in import configuration window.

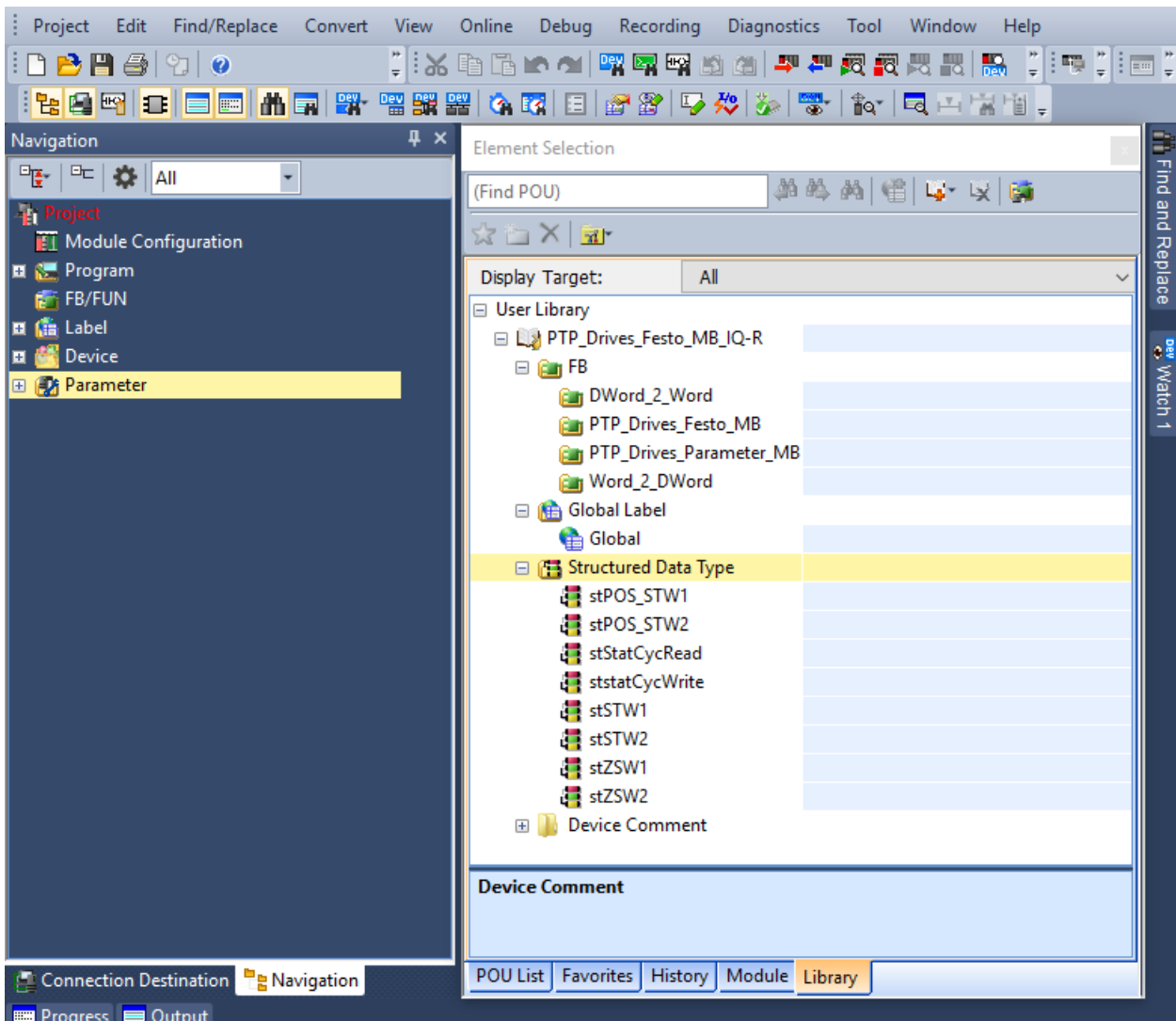
Step – 3



1. Enter the Library File location in the field.

2. Select “PTP_Drives_Festo_MB_FX5U.usl” library file.
3. Click “Open” to register the library.

After successful import of PTP_Drives_Festo_MB_FX5U will be displayed in User Library folder as shown in below picture.



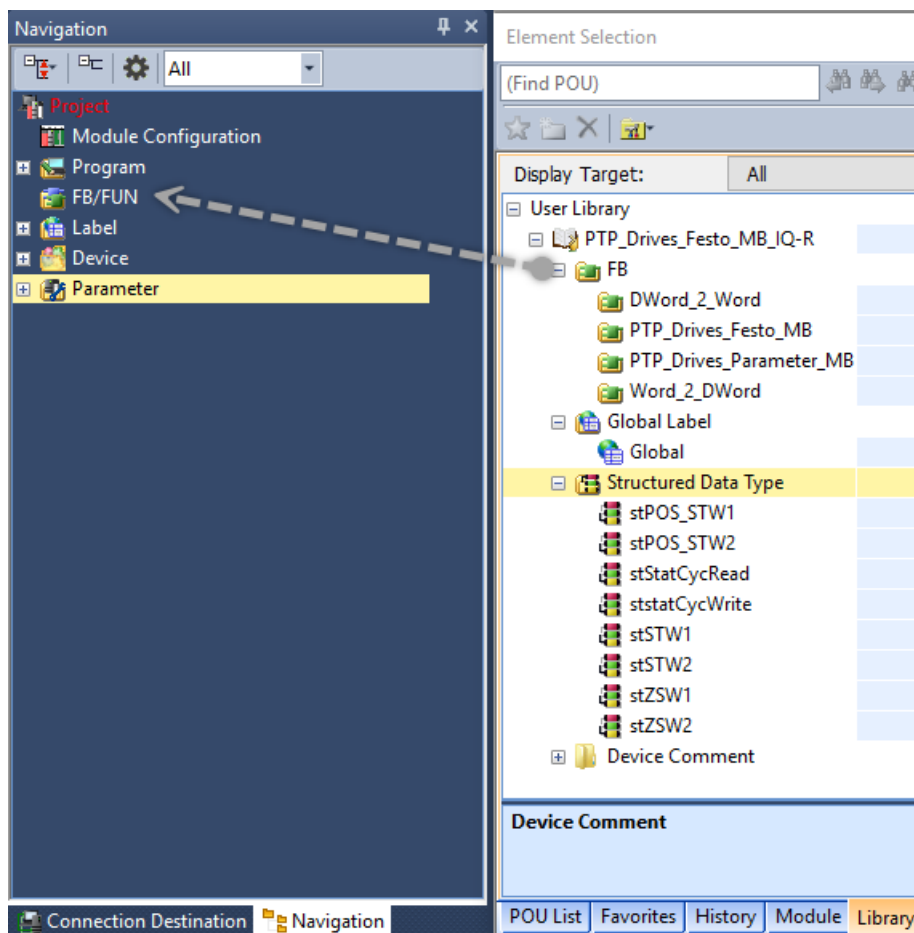
Note

- All user defined libraries will be under User Library Folder.

Step – 5 Add function block to Program.

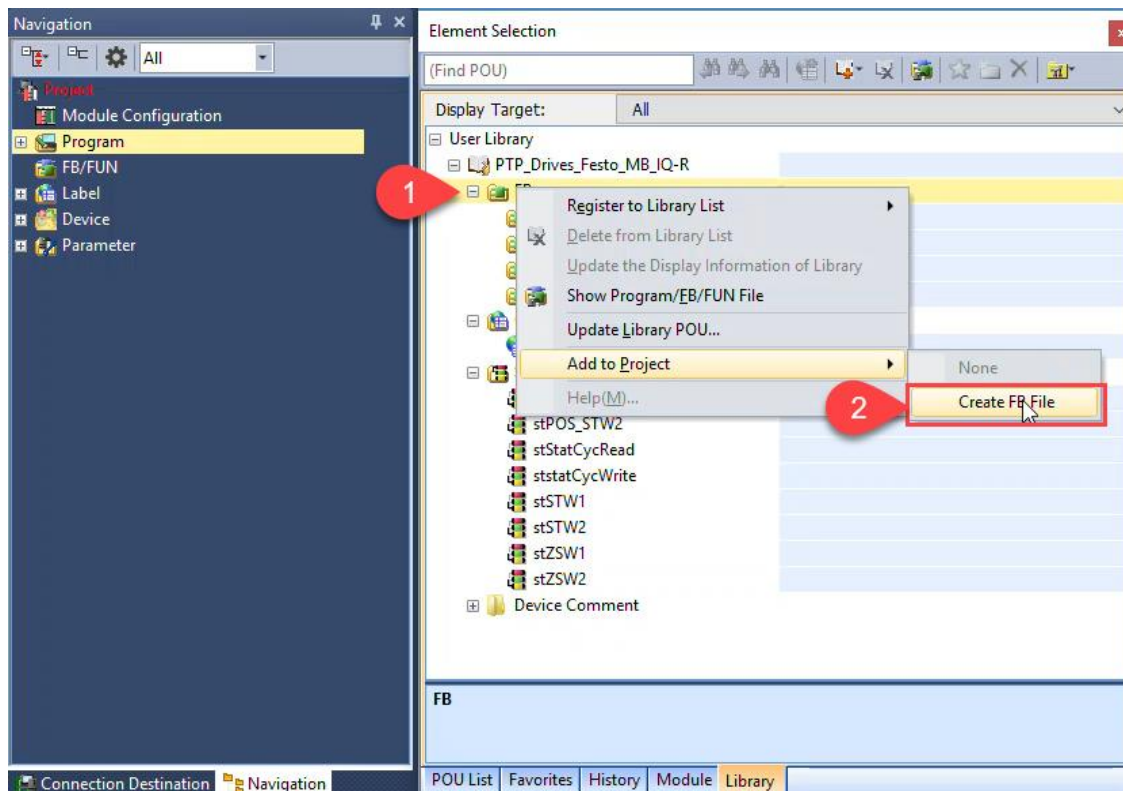
Method 1:-

Starting a Project



Drag and drop the **All FB's** from User Library folder to the **FB/FUN** as shown in below image.

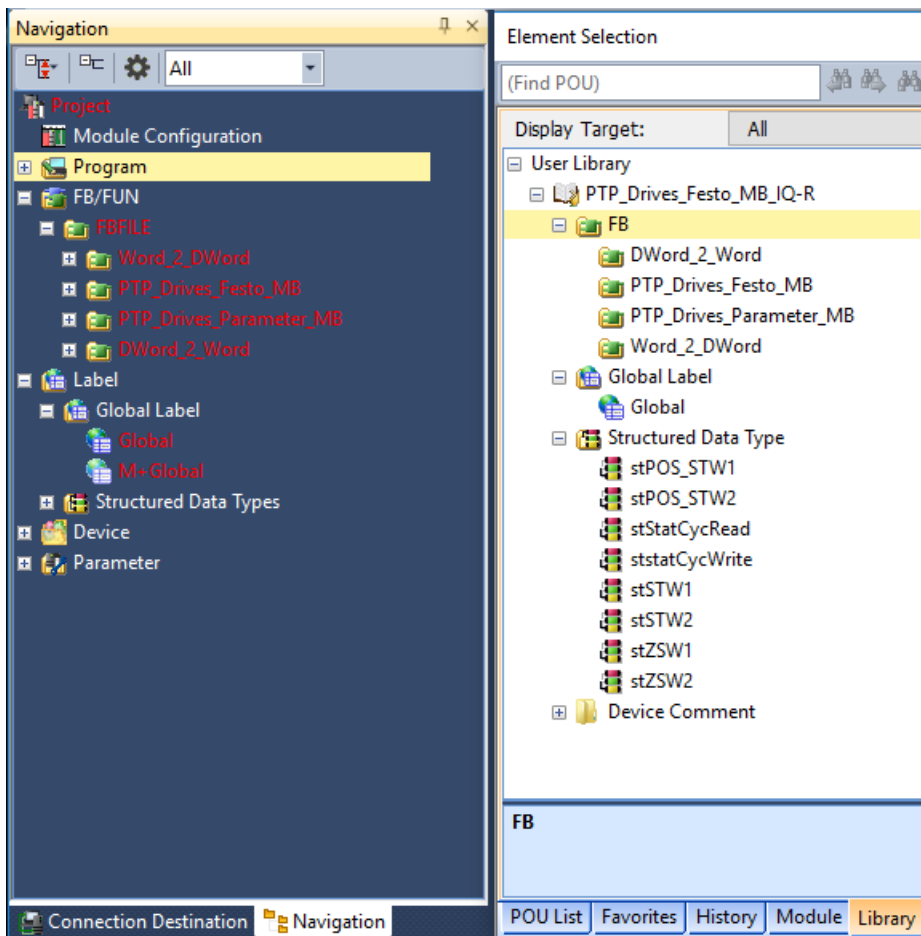
Method 2:-



1.  Right Click on **FB** under [PTP_Drives_Festo_MB_IQ-R] as shown in below image .

2. Go to **Add to Project** => Select **“Create FB File”**.

After successful addition of PTP_Drives_Festo_MB_IQ-R will be displayed in FB/FUN folder as shown in below picture.



4 Elements in the PTP_Drives_Festo_MB_IQ-R Library

Function Blocks

There are four function blocks in the library namely:

- PTP_Drives_Festo_MB
- PTP_Drives_Parameter_MB
- Word_2_DWord
- DWord_2_Word

4.1 PTP_Drives_Festo_MB

PTP_Drives_Festo_MB Function Block is used for controlling the drive in the following modes.

- **ModePos – 1:** Relative Positioning
- **ModePos – 2:** Absolute Positioning
- **ModePos – 3:** Setup Mode
- **ModePos – 4:** Home/Reference using Festo Automation Suite Configuration
- **ModePos – 5:** Home to Current Position
- **ModePos – 6:** Traversing/Record Select Mode
- **ModePos – 7:** Jog
- **ModePos – 8 :** Inching/Incremental Jog Mode

PTP_Drives_Festo_MB_1 (PTP_Drives_Festo_MB)	
UW: uiModbusTCPCommStatus	DRV_Control :UW
UW: DRV_Status	AxisEnabled :B
W: ModePos	AxisPosOk :B
B: EnableAxis	AxisSpFixed :B
B: CancelTraversing	AxisAckSetpoint :B
B: IntermediateStop	ClampTorqueReached :B
B: Positive	AxisRef :B
B: Negative	AxisWarn :B
B: Jog1	AxisError :B
B: Jog2	Lockout :B
B: AckError	ActVelocity :E
B: ExecuteMode	ActPosition :D
B: ReleaseBrake	ActMode :W
B: TravelToFixStop	EPosZSW1 :UW
D: Position	EPosZSW2 :UW
D: Velocity	ActWarn :UW
W: OverV	ActFault :UW
W: OverAcc	Error :B
W: OverDec	Status :UW
UD: ConfigEPos	
E: BaseSpeedValue	

The following tables gives a detailed explanation of the inputs and outputs data types of the function block.

4.1.1 Input and Output Interface

INPUT DATA TAGS

Tag Name	Data Type	Function Description
uiModbusTCPCommStatus	Word [Unsigned]/Bit String	Simple CPU Communication Status Refer Chapter 3.3.3
DRV_Status	Word [Unsigned]/Bit String [16-bit](0..11)	Gives the drive feedback.
ModePos	Word [Signed]	Gives the Operating mode selection command. It has the following values. 1=MDI Relative Pos. 2=MDI Absolute Pos. 3= Setup Mode 4=Home FAS Config. 5=Home Current Pos. 6=Record Select 7=Jogging 8=Inching
EnableAxis	Bit	TRUE - Enable the axis FALSE – Disable the axis.
CancelTraversing	Bit	Cancel travel task (0 – Cancel Travel) Ramp Stop @ 100% Set Deceleration Task is Rejected
IntermediateStop	Bit	Interrupt travel task (0 – Interrupt Travel) Ramp Stop @ Current Deceleration Task is NOT Rejected
Positive	Bit	TRUE -Travel positive direction.
Negative	Bit	TRUE -Travel negative direction
Jog1	Bit	TRUE – Drive jogs in Positive direction
Jog2	Bit	TRUE – Drive jogs in negative direction
AckError	Bit	Clear error command (1 – Clear error)
ExecuteMode	Bit	Execute the task (1 – Execute)
ReleaseBrake	Bit	Release Servo Motor Brake (1 – Release)
TravelToFixStop	Bit	Enable Travel to Fix Stop mode (1 – Enable)
Position	Double Word [Signed]	Position Target Setpoint or Record table selection in traversing block mode Position Target Value x FAS=P1.7841.0.0 Factor Group for Position = 10 ⁻⁶ = FAS P1.11280605.0.0 Target Position MDI Example: Position Target Setpoint= 2500000 x 10 ⁻⁶ = 2500.000 mm

Tag Name	Data Type	Function Description
Velocity	Double Word [Signed]	Velocity Target Setpoint Velocity in User Units While using MDI (Manual Data Input) Mode Velocity Target Setpoint $\times$ FAS=P1.7842.0.0 Factor Group for Velocity = 10^{-3} $\times$ OverV FAS=P1.1309.0.0 Velocity Override = FAS P1.11280605.0.0 Profile Velocity MDI Example1: Velocity Target Setpoint = $50 \times 10^{-3} = 0.05 \text{ m/s}$ Example2: Velocity Target Setpoint = $250 \times 10^{-3} = 0.25 \text{ m/s}$
OverV	Word [Signed]	Velocity override in % (0 to 199 %) OverV Value = 0-100% FAS=P1.1309.0.0 Velocity Override
OverAcc	Word [Signed]	Acceleration override in % (0 to 100 %) OverAcc Target Value = 0-100% $\times$ FAS P1.11280702.0.0 Base Value Acceleration = FAS P1.11280606.0.0 Acceleration MDI
OverDec	Word [Signed]	Deceleration override in % (0 to 100 %) OverDec Target Value = 0-100% $\times$ FAS P1.11280703.0.0 Base Value Deceleration = FAS P1.11280607.0.0 Deceleration MDI
ConfigEPos	Double Word [Unsigned]/Bit String [32-bit]	Configuration command (Bit – Granular) See Chapter 4.1.2 for further details
BaseSpeedValue	FLOAT [Single Precision]	Base speed value for actual speed display. Enter the same value given for parameter (P1.11280701.0.0)

OUTPUT DATA TAGS

Tag Name	Data Type	Function Description
DRV_Control	Word [Unsigned]/Bit String [16-bit](0..11)	Drive control hardware interface.
AxisEnabled	Bit	TRUE - Axis enabled feedback Drive Ready and Switched On/Enabled
AxisPosOk	Bit	TRUE - Axis reached target position Reached & Motion Complete
AxisSpFixed	Bit	TRUE - Setpoint is stationary feedback
AxisAckSetpoint	Bit	TRUE - Acknowledge setpoint
ClampTorqueReached	Bit	TRUE - ClampTorqueReached

Tag Name	Data Type	Function Description
AxisRef	Bit	TRUE -Referencing/Homing is Completed
AxisWarn	Bit	TRUE - Drive has a warning message (Warning Active)
AxisError	Bit	TRUE -Drive has a error message (Error Active)
Lockout	Bit	TRUE -Switching ON inhibited (ON Inhibited)
ActVelocity	FLOAT [Single Precision]	Actual velocity feedback (Feedback value scaled by base speed value)
ActPosition	Double Word [Signed]	Axis actual position feedback
ActMode	Word [Signed]	Actual mode selection feedback 1=MDI Relative Pos. 2=MDI Absolute Pos. 4=Home FAS Config. 5=Home Current Pos. 6=Record Select 7=Jogging 8=Inching
EposZSW1	Word [Unsigned]/Bit String [16-bit]	Status of Epos ZSW1 (Bit – Granular) See chapter 4.1.3 for further details
EposZSW2	Word [Unsigned]/Bit String [16-bit]	Status of Epos ZSW2 (Bit – Granular) See chapter 4.1.4 for further details
ActWarn	Word [Unsigned]/Bit String [16-bit]	Actual warning number
ActFault	Word [Unsigned]/Bit String [16-bit]	Actual fault number
Error	Bit	TRUE -AOI Fault Active
Status	Word [Unsigned]/Bit String [16-bit]	AOI Status Code: See Chapter 4.1.5 for further details

4.1.2 ConfigEPos Bit Description

ConfigEPos Bits	Function Description
Bit - 0	Drive run out, OFF2 (Pulse Enable)
Bit - 1	Quick stop, OFF3 (Pulse Enable)
Bit - 2	Software Limits Enable FAS=P1.112414140.0.0
Bit - 3	Hardware Limits Enable FAS=P1.112414150.0.0
Bit - 4	Touch probe edge (1: failing edge, 0: rising edge)
Bit - 5	Touch probe source (1: secondary encoder, 0: primary encoder)
Bit - 6	External record change via fieldbus, STW1.13
Bit – 7	Reserved
Bit - 8	Continuous Setpoint Transfer MDI, POS_STW1.12
Bit - 9	Reserved
Bit - 10	Reserved
Bit - 11	Reserved
Bit - 12	Reserved

ConfigEPos Bits	Function Description
Bit - 13	Reserved
Bit - 14	Reserved
Bit - 15	Holding Brake, 0 = set, 1 = release
Bit – 16, Bit - 17	Reserved
Bit – 18	Tracking mode, POS_STW2.0
Bit – 19 to Bit - 29	Reserved
Bit – 30	1 = activate travel to fixed stop, STW2.8
Bit – 31	Reserved

4.1.3 Epos ZSW1 Bit Details

EPos ZSW1 Bits	Function Description
Bit – 0	Position Record Selected Bit 0
Bit – 1	Position Record Selected Bit 1
Bit – 2	Position Record Selected Bit 2
Bit – 3	Position Record Selected Bit 3
Bit – 4	Position Record Selected Bit 4
Bit – 5	Position Record Selected Bit 5
Bit – 6	Position Record Selected Bit 6
Bit – 7	Reserved
Bit – 8	Negative Hardware Limit Switch is Activated
Bit – 9	Positive Hardware Limit Switch is Activated
Bit - 10	Jogging Task is Active
Bit - 11	Homing Task is Active
Bit - 12	- FW Pending - Flying Reference Active
Bit - 13	Position Record Task is Active
Bit - 14	- FW Pending - MDI Setup Active
Bit - 15	MDI Mode is Active

4.1.4 Epos ZSW2 Bit Details

EPos ZSW2 Bits	Function Description
Bit – 0	Tracking Mode Active
Bit – 1	Velocity Limit Active
Bit – 2	Ready to Accept new Setpoint
Bit - 3	- FW Pending - Print Mark Outside Outer Window
Bit - 4	Axis is Moving Positive
Bit - 5	Axis is Moving Negative
Bit - 6	Negative Software Limit Switch is Activated
Bit - 7	Positive Software Limit Switch is Activated
Bit - 8	Actual Position <= CAM Position 1

EPos ZSW2 Bits	Function Description
Bit - 9	Actual Position <= CAM Position 2
Bit - 10	Direct Output 1 Via Traversing Block
Bit - 11	Direct Output 2 Via Traversing Block
Bit - 12	Fixed STOP Reached
Bit - 13	Fixed STOP Clamping Torque Reached
Bit - 14	Travel to Fixed STOP Active
Bit - 15	Traversing Task Active

4.1.5 Diagnostic Messages

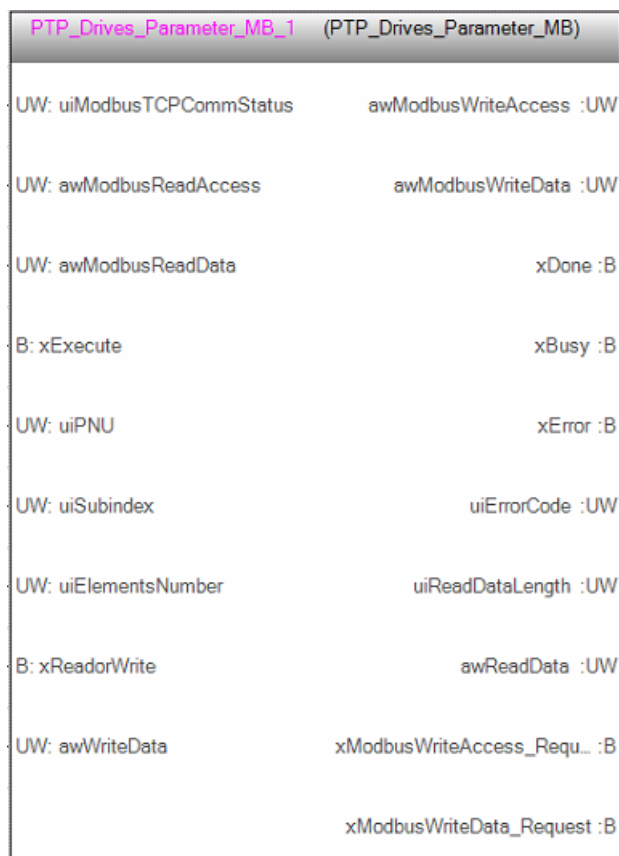
The function block gives multiple types of diagnostic messages. The following chapter will describes how to handle & solve them.

Function Block Faults

Status Error Number	Case	Remedy
7002	No Error	--
8401	Alarm message(s) in the drive	Check the error code in ActFaulted tag
8402	Switching ON is Inhibited .	Check whether axis is parked, Safety functions active & Parameter P10 <> 0
8403	- FW Pending - Flying Reference could NOT be Started	Check for pending alarms / faults in the drive
8600	Communication Disconnected	Check the communication connections/ setting between drive and PLC
8202	Incorrect Operating Mode Selected	Set ModePos tag value between 1 to 8
8203	Incorrect parameterization of the override inputs	Check the setting of override inputs
8204	Invalid traversing block number	Enter the traversing block number from 0 to 127

4.2 PTP_Drives_Parameter_MB

This Function Block allows the user to read and write a single parameter of CMMT drive from Mitsubishi PLC via Modbus TCP Simple CPU Communication.



4.2.1 Input and Output Interface

INTPUT DATA TAGS

Tag Name	Data Type	Description
uiModbusTCPCommStatus	Word [Unsigned]/Bit String [16-bit]	ModbusTCP Simple CPU Communication Status. Refer Chapter 3.3.3
awModbusReadAccess	Word [Unsigned]/Bit String [16-bit](0..4)	Array of Word through which data is Read from Drive.
awModbusReadData	Word [Unsigned]/Bit String [16-bit](0..29)	Array of Word through which data is Read from Drive.
xExecute	Bit	TRUE = FB is executed once
uiPNU	Word [Unsigned]/Bit String [16-bit]	PNU for single parameter access
uiSubindex	Word [Unsigned]/Bit String [16-bit]	Subindex of the PNU
uiElementsNumber	Word [Unsigned]/Bit String [16-bit]	Number of items in single parameter access
xReadorWrite	Bit	FALSE = Read TRUE = Write
awWriteData	Word [Unsigned]/Bit String [16-bit](0..29)	Used to feed the write parameter values during write function.

OUTPUT DATA TAGS

TagName	Data Type	Description
---------	-----------	-------------

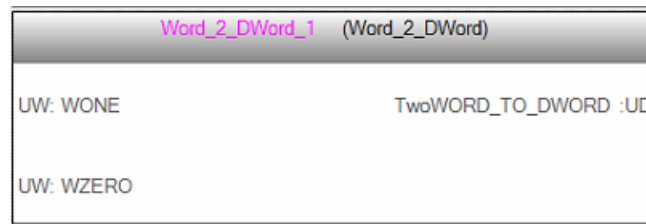
awModbusWriteAccess	Word [Unsigned]/Bit String [16-bit](0..3)	Array of Word through which data is Write to Drive.
awModbusWriteData	Word [Unsigned]/Bit String [16-bit](0..29)	Array of Word through which data is Write to Drive.
xDone	Bit	TRUE = FB successfully executed
xBusy	Bit	TRUE = FB being executed
xErro	Bit	TRUE = Error during execution
uiErrorCode	Word [Unsigned]/Bit String [16-bit]	Error code of Parameter block Refer Chapter 4.2.2
uiReadDataLength	Word [Unsigned]/Bit String [16-bit]	Length of read or written data in bytes.
awReadData	Word [Unsigned]/Bit String [16-bit](0..29)	Array of Word through which data is Read from Drive.
xModbusWriteAccess_Request	Bit	Communication Start Request for each setting No. Refer Chapter 3.3.3
xModbusWriteData_Request	Bit	Communication Start Request for each setting No. Refer Chapter 3.3.3

4.2.2 Error Description

Error Code	Name	Description
10000	Communication Failure	Simple PLC communication Status for each setting No.
10001	PNU not entered	Write the correct parameter PNU.
10002	Elements Number not entered	Write the correct Element Number. (Always Element number should be "1" . (Refer 4.2.3)
10003	Timeout	The size of the path which was sent with the Service Request is either not large enough to allow the Request to be routed to an object or too much routing data was included.(See the Description given below in Wrong Dara Entry)
10010	Wrong Data Entry	Check the following cause – PNU of parameter associated with the request was invalid. – SubIndex of parameter associated with the request was invalid. – Elements Number of subindex of the PNU of parameter associated with the request was invalid. This code is used when a parameter does not meet the requirements of this specification and/or the requirements defined in an Application Object Specification.

4.1 Word_2_DWord

This Function Block allows the user to Convert Array of word input to Double Word output. This block output TwoWORD_TO_DWORD is used to give the input data of parameter block



4.1.1 Input Interface

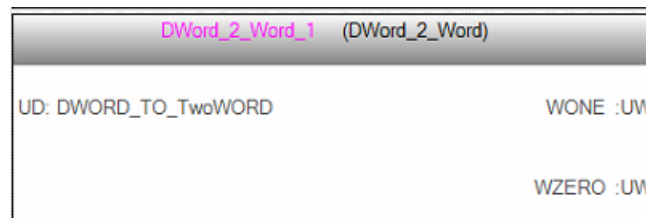
Tag Name	Data Type	Description
WONE	Word [Unsigned]/Bit String [16-bit]	Second word from Array of Two word.
WZERO	Word [Unsigned]/Bit String [16-bit]	First word from Array of Two word.

4.1.2 Output Interface

TagName	Data Type	Description
TwoWORD_TO_DWORD	Double Word [Unsigned]/Bit String [32-bit]	Two Word Parameter Data read from Parameter Block.

4.2 DWord_2_Word

This Function Block allows the user to Convert Double Word Input to Array of word Output. This block output WONE and WZERO is used to give the input data of parameter block



4.2.1 Input Interface

TagName	Data Type	Description
DWORD_TO_TwoWORD	Double Word [Unsigned]/Bit String [32-bit]	Double word Parameter Data write to Parameter Block.

4.2.2 Output Interface

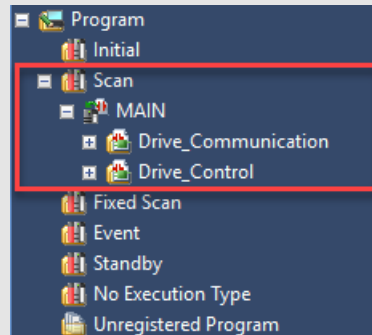
Tag Name	Data Type	Description
WONE	Word [Unsigned]/Bit String [16-bit]	Second word from Array of Two word.
WZERO	Word [Unsigned]/Bit String [16-bit]	First word from Array of Two word.

5 Library Configuration

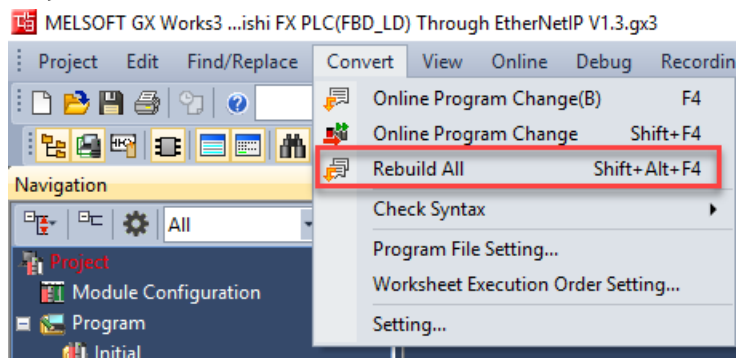


Note

The function blocks must be called cyclically with a separate instance for every CMMT Drives integrated in the user program. Simultaneous use of multiple instances of the same device is not permitted. Any Function Block that must be called in **Scan** under Program file is called cyclically. Refer Image below.



Step – 1



– [Convert] => [Rebuild All]

Step – 2

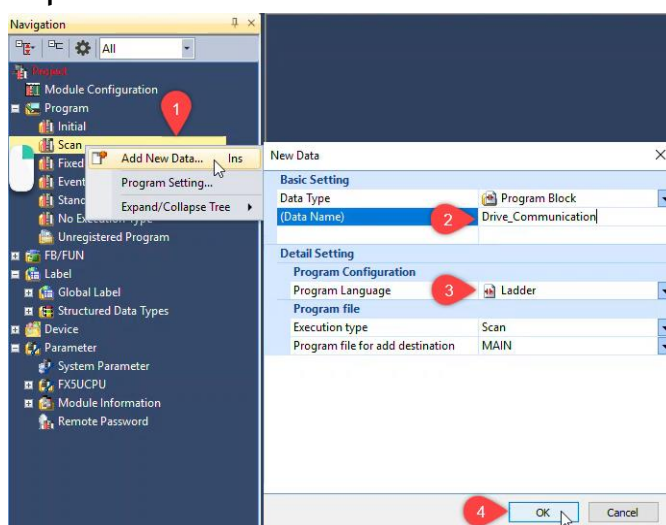


Fig – 1

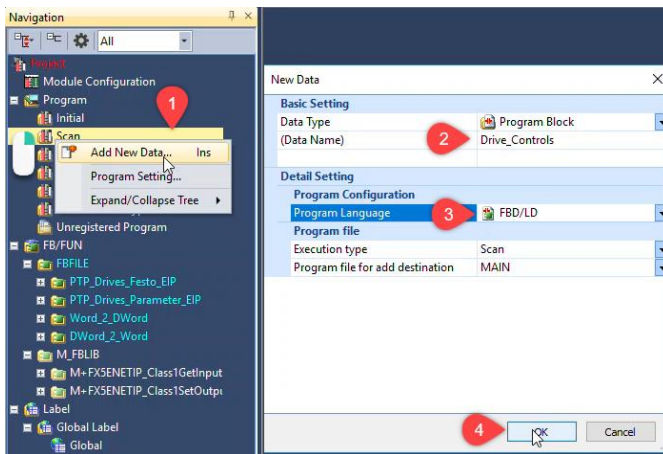


Fig – 2

1. [Program] => [Scan]
 - a. Right-Click on **“Scan”** in the **“Project”** side bar.
 - b. Left-Click on **“Add new Data.”**
 - c. A new window **“New Data”** will be opened.
2. [Data Name] > [User Defined] or use the ones in the example image.
3. [Program Language] > [Ladder] (Refer Fig – 1)
[Program Language] > [FBD/LD] (Refer Fig – 2)
4. Confirm your selection with the button **“OK”**.

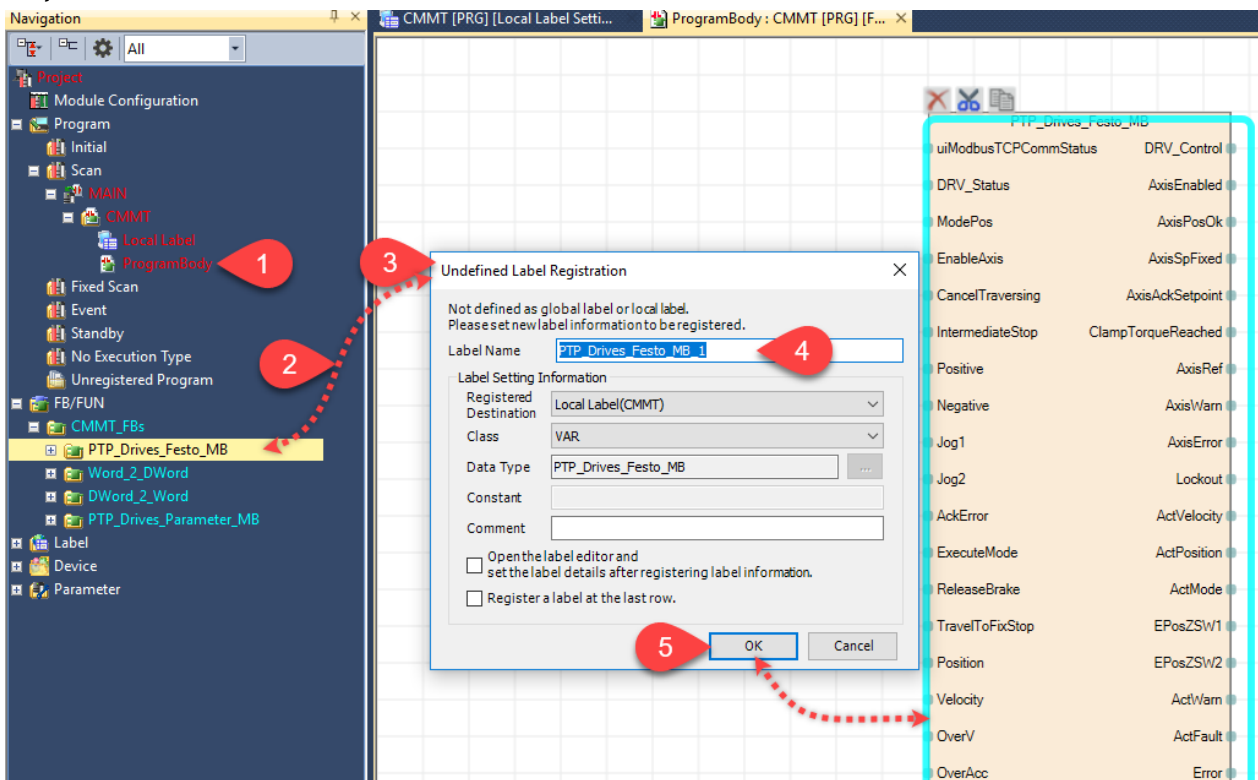


Information

PTP_Drives_Festo_MB FB is written in Structure Text language, it can be called in Ladder and FBD/LD Program Language

5.1 Configuring the PTP_Drives_Festo_MB Function Block Input & Output Interface Tags in FBD

Step – 1

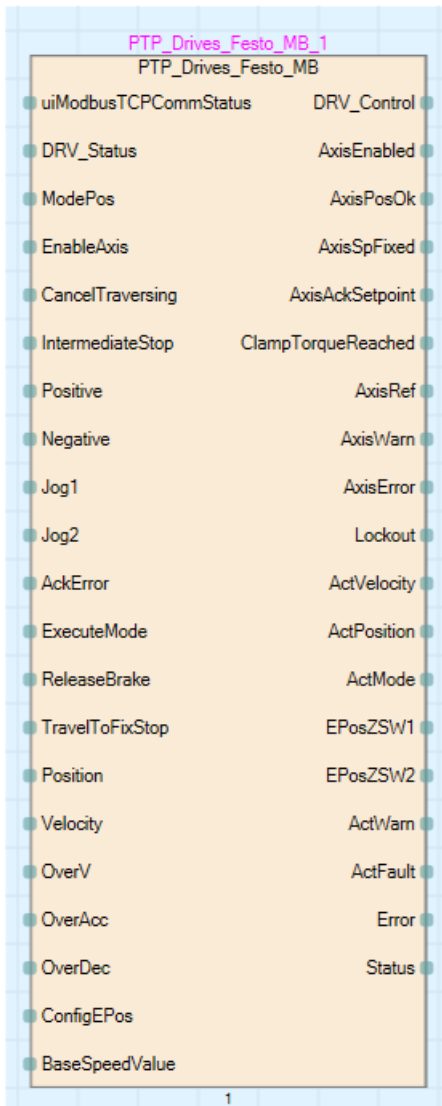


1. Open the **“Main Program”**.

2. Drag & Drop the “**PTP_Drives_Festo_MB**” block from FB/FUN to program area.
3. The “**Label Registration**” tab will open.
4. Enter the name of the FB on the Label Name field.
5. Click on **OK** to confirm the FB name.

The called function block shown in below Step-2

Step – 2



Step – 3

Assigning the Communication Status Bit to PTP_Drives_Festo_MB Function Block

Setting Item List

Input the Setting Item to Search

- Basic Settings
- Application Settings
 - Frame Settings
 - Communication Speed
 - FTP Server Settings
 - MODBUS/TCP Settings
 - Simple CPU Communication Setting
 - Simple Device Communication Setting
 - Time Setting
 - Timer Settings for Data Communication
 - Security
 - Gateway Parameter Settings

Setting Item

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

Communication Destination Filter: Show All

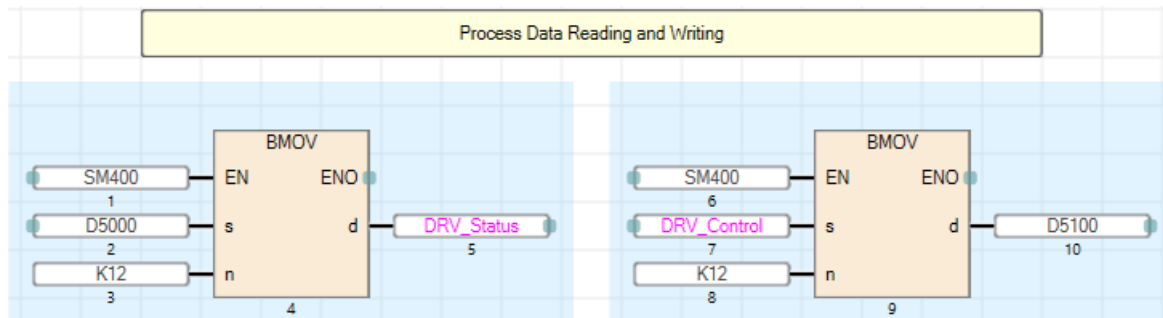
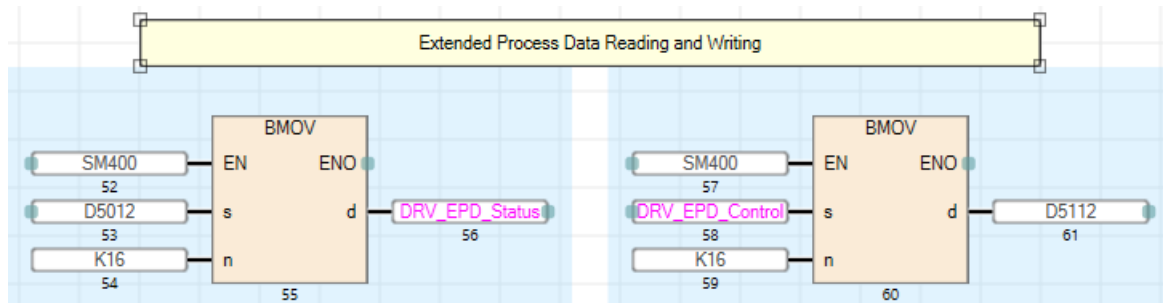
Setting No.	Communication Pattern	Communication Setting: Execution Interval(ms)	Communication Destination (IP Address)		Target PLC No.
			Source	Destination	
1	Read	Fixed Interval	10 MODBUS/TCP(192 . 168 . 1 . 5)	Host Station(192 . 168 . 1 . 2)	Not Specified
2	Write	Fixed Interval	10 Host Station(192 . 168 . 1 . 2)	MODBUS/TCP(192 . 168 . 1 . 5)	Not Specified
3	Read	Fixed Interval	50 MODBUS/TCP(192 . 168 . 1 . 5)	Host Station(192 . 168 . 1 . 2)	Not Specified
4	Read	Fixed Interval	50 MODBUS/TCP(192 . 168 . 1 . 5)	Host Station(192 . 168 . 1 . 2)	Not Specified
5	Write	On Request	Host Station(192 . 168 . 1 . 2)	MODBUS/TCP(192 . 168 . 1 . 5)	Not Specified
6	Write	On Request	Host Station(192 . 168 . 1 . 2)	MODBUS/TCP(192 . 168 . 1 . 5)	Not Specified
7					

- Under Element Selection
Select Module
- Under Module Label --> Expand - 0000:_RJ71EN71(E+IEF) --> Expand - EN71_EF_1
- Expand Ethernet(PORT1)
- Expand Simple PLC Communication area
 - Expand SimplePLC status setting no.[1 To 64]
 - Referring **simple CPU Communication setting No.**, select Simple PLC communication status (**wnStatus_SimplePLC_D[_]**) from "SimplePLC status setting no.[1 To 64]" tree.
- The yellow highlighted communication are **fixed interval** so the communication establishes on every execution interval of time.
- Drag & drop Simple PLC communication status (**wnStatus_SimplePLC_D[1]**) and link it to Variable Local tag (Status – Word [Signed]).



Note

- Point 5. - In Simple CPU Communication, if multiple Settings are assigned to read/write data with multiple CMMT drives then respective Setting No. to be used for Simple CPU Communication status.
- If communication is established successfully then value will be displayed 3.

Step – 4**Mapping Process Data:****Mapping Extended Process Data:**

To know the length of process data & extended process data, Refer [chapter 3.3.2](#) for more details.

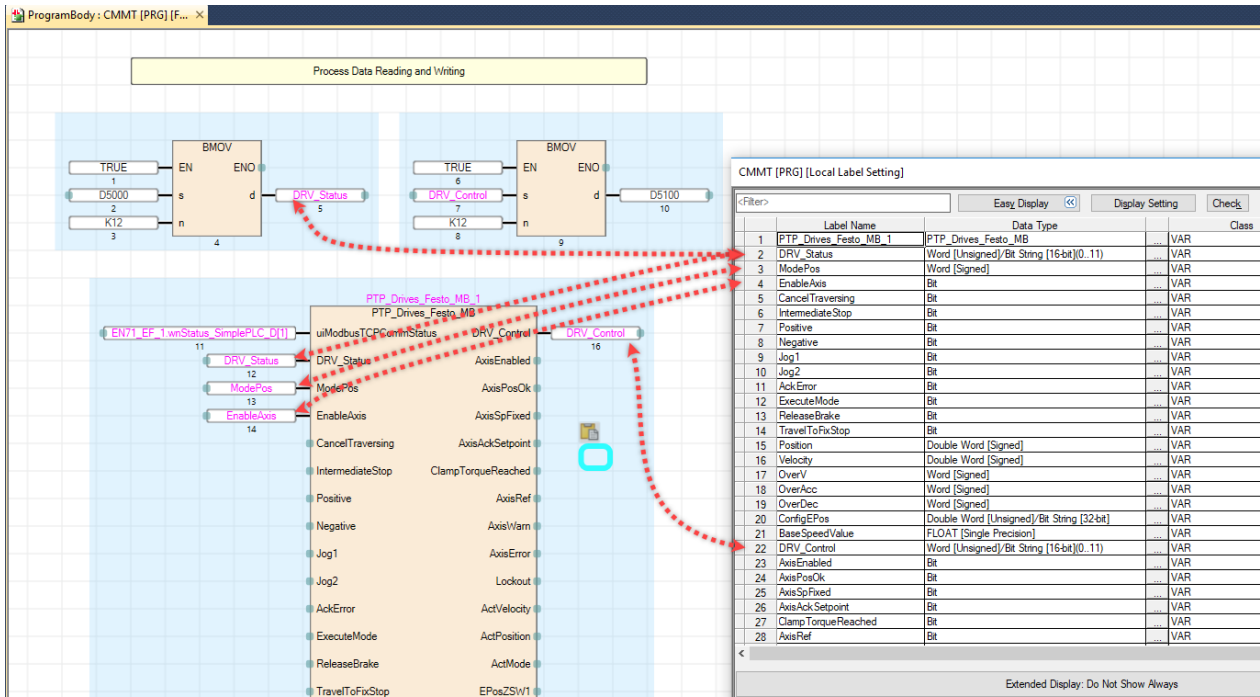
Input Mapping

1. Call the **“BMOV”** Function block to move the input data from Modbus Simple CPU Communication input data to Internal array tag.
2. Assign the enable tag to **“EN”** to enable the function block.
3. Assign the Starting address of the Modbus Simple CPU Communication input Data of specific value in the **“S”** input.
4. Assign the number of data to be moved in the **“n”** input.
5. Assign the starting address of the internal array tag in the **“d”** output.

Output Mapping

1. Call the **“BMOV”** Function block to move the Internal array tag data to Modbus Simple CPU Communication output data address.
2. Assign the enable tag to **“EN”** to enable the function block.
3. Assign the Starting address of the Internal array tag in the **“S”** input.
4. Assign the number of data to be moved in the **“n”** input.
5. Assign the starting address of the Modbus Simple CPU Communication output tag in the **“d”** output.

Step – 5



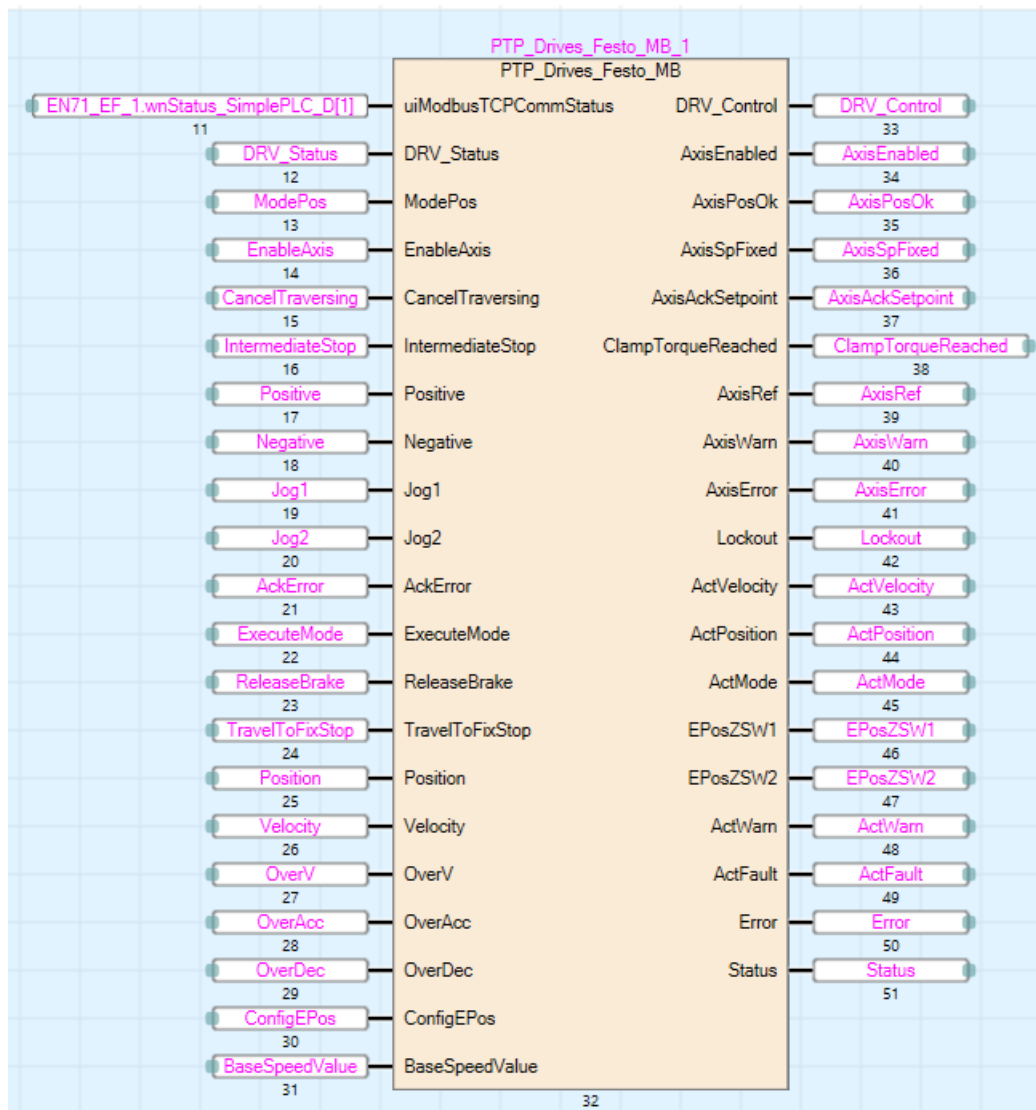
- Open the Local label to view the created local tags.
- Drag and Drop the Modepos tag from Local label to Modepos input tag in **PTP_Drives_Festo_MB** function block.



Note

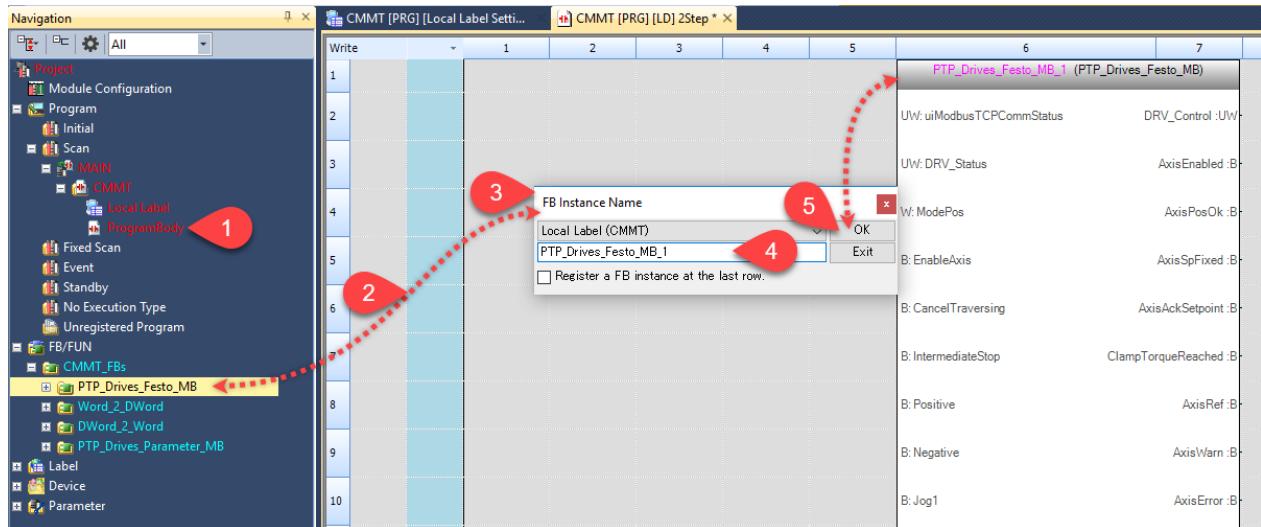
- Repeat Step-5 to assign the different input & output tags in function block.

Once all steps completed successfully the **PTP_Drives_Festo_MB** Function block will be look like below.



5.2 Configuring the PTP_Drives_Festo_MB Function Block Input & Output Interface Tags in Ladder

Step – 1



1. Open the **“Main Program”**.
2. Drag & Drop the **“PTP_Drives_Festo_MB”** block from FB/FUN to program area.
3. The **“Label Registration”** tab will open.
4. Enter the name of the FB on the Label Name field.
5. Click on **OK** to confirm the FB name.

The called function block shown in below Step-2

Step – 2

Library Configuration

PTP_Drives_Festo_MB_1 (PTP_Drives_Festo_MB)	
UW: uiModbusTCPComm...	DRV_Control :UW
UW: DRV_Status	AxisEnabled :B
W: ModePos	AxisPosOk :B
B: EnableAxis	AxisSpFixed :B
B: CancelTraversing	AxisAckSetpoint :B
B: IntermediateStop	ClampTorqueReached :B
B: Positive	AxisRef :B
B: Negative	AxisWarn :B
B: Jog1	AxisError :B
B: Jog2	Lockout :B
B: AckError	ActVelocity :E
B: ExecuteMode	ActPosition :D
B: ReleaseBrake	ActMode :W
B: TravelToFixStop	EPosZSw1 :UW
D: Position	EPosZSw2 :UW
D: Velocity	ActWarn :UW
W: OverV	ActFault :UW
W: OverAcc	Error :B
W: OverDec	Status :UW
UD: ConfigEPos	
E: BaseSpeedValue	

Step – 3

Assigning the Communication Status Bit to PTP_Drives_Festo_MB Function Block

The screenshot shows the Mitsubishi GX Developer software interface. The top window is the Variable Declaration window, showing a table with columns for Variable Name, Data Type, and Unit. The second window is the Element Selection tree, showing the hierarchy of modules and communication areas. The third window is the Setting Item List, showing the configuration of communication settings. Red numbered callouts (1-6) indicate the steps to be followed:

- Under Element Selection, Select Module
- Under Module Label --> Expand - 0000:_RJ71EN71(E+IEF) --> Expand - EN71_EF_1
- Expand Ethernet(PORT1)
- Expand Simple PLC Communication area
 - Expand SimplePLC status setting no.[1 To 64]
 - Referring **simple CPU Communication setting No.**, select Simple PLC communication status (**wnStatus_SimplePLC_D[_]**) from “SimplePLC status setting no.[1 To 64]” tree.
- The yellow highlighted communication are **fixed interval** so the communication establishes on every execution interval of time.
- Drag & drop Simple PLC communication status (**wnStatus_SimplePLC_D[1]**) and link it to Variable Local tag (Status – Word [Signed]).

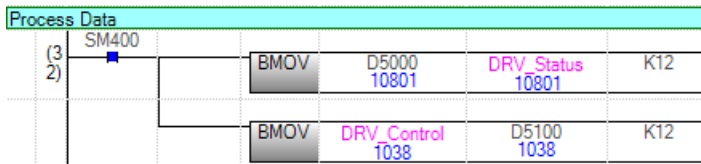


Note

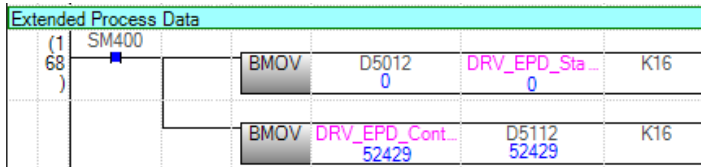
- Point 5. - In Simple CPU Communication, if multiple Settings are assigned to read/write data with multiple CMMT drives then respective Setting No. to be used for Simple CPU Communication status.
- If communication is established successfully then value will be displayed 3.

Step – 4

Mapping Process Data:



Mapping Extended Process Data:



To know the length of process data & extended process data, Refer [chapter 3.3.2](#) for more details..

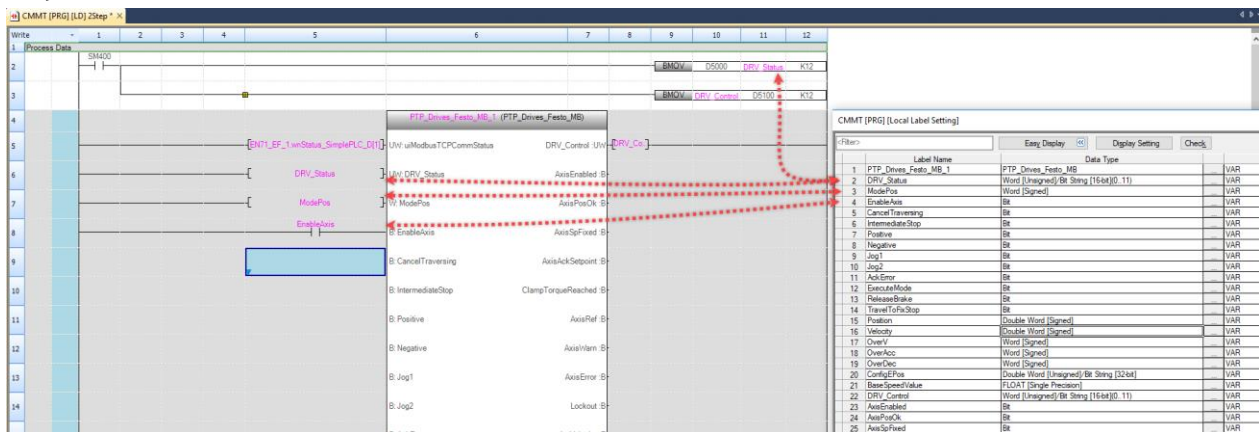
Input Mapping

1. Call the **“BMOV”** Function block to move the input data from Modbus Simple CPU Communication input data to Internal array tag.
2. Assign the enable tag to **“EN”** to enable the function block.
3. Assign the Starting address of the Modbus Simple CPU Communication input Data of specific value in the **“S”** input.
4. Assign the number of data to be moved in the **“n”** input.
5. Assign the starting address of the internal array tag in the **“d”** output.

Output Mapping

1. Call the **“BMOV”** Function block to move the Internal array tag data to Modbus Simple CPU Communication output data address.
2. Assign the enable tag to **“EN”** to enable the function block.
3. Assign the Starting address of the Internal array tag in the **“S”** input.
4. Assign the number of data to be moved in the **“n”** input.
5. Assign the starting address of the Modbus Simple CPU Communication output tag in the **“d”** output.

Step – 5



- Open the Local label to view the created local tags.
- Drag and Drop the Modepos tag from Local label to Modepos input tag in **PTP_Drives_Festo_MB** function block.



Note

- Repeat Step-5 to assign the different input & output tags in function block.

Once all steps completed successfully the **PTP_Drives_Festo_MB** Function block will be look like below.

Write	1	2	3	4	5	6	7	8	9	10	11	12
1	Process Data											
2	(0)	SM400								BMO D5000	DRV_Status	K12
3										BMO DRV_Control	D5100	K12
4	(2 4)					PTP_Drive_Festo_MB_1 (PTP_Drive_Festo_MB)						
5					[EN71_EF_1.wnStatus_SimplePLC_D[1]]	UW: uiModbusTCPCComm...	DRV_Control :UW		[DRV_Control]			
6					[DRV_Status]	UW: DRV_Status	AxisEnabled :B				AxisEnabled	
7					[ModePos]	W: ModePos	AxisPosOk :B				AxisPosOk	
8					EnableAxis	B: EnableAxis	AxisSpFixed :B				AxisSpFixed	
9					CancelTraversing	B: CancelTraversing	AxisAckSetpoint :B				AxisAckSetpoint	
10					IntermediateStop	B: IntermediateStop	ClampTorqueReached :B				ClampTorqueReached	
11					Positive	B: Positive	AxisRef :B				AxisRef	
12					Negative	B: Negative	AxisWarn :B				AxisWarn	
13					Jog1	B: Jog1	AxisError :B				AxisError	
14					Jog2	B: Jog2	Lockout :B				Lockout	
15					AckError	B: AckError	ActVelocity :E		[ActVelocity]			
16					ExecuteMode	B: ExecuteMode	ActPosition :D		[ActPosition]			
17					ReleaseBrake	B: ReleaseBrake	ActMode :W		[ActMode]			
18					TravelToFixStop	B: TravelToFixStop	EPosZSW1 :UW		[EPosZSW1]			
19					[Position]	D: Position	EPosZSW2 :UW		[EPosZSW2]			
20					[Velocity]	D: Velocity	ActWarn :UW		[ActWarn]			
21					[OverV]	W: OverV	ActFault :UW		[ActFault]			
22					[OverAcc]	W: OverAcc	Error :B				Error	
23					[OverDec]	W: OverDec	Status :UW		[Status]			
24					[ConfigEPos]	UD: ConfigEPos						
25					[BaseSpeedValue]	E: BaseSpeedValue						

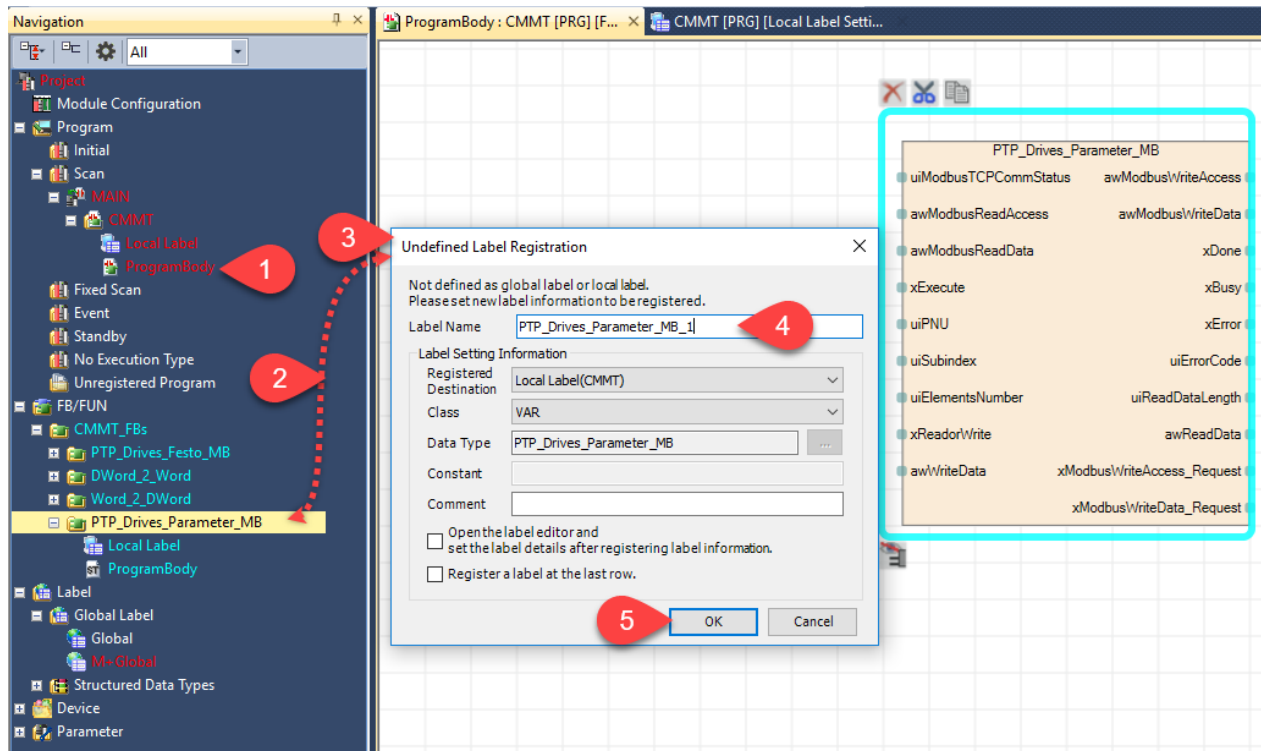


Information

PTP_Drives_Parameter_MB FB is written in Structure Text language, it can be called in Ladder and FBD/LD Program Language

5.3 Configuring the PTP_Drives_Parameter_MB Function Block Input & Output Interface Tags in FBD

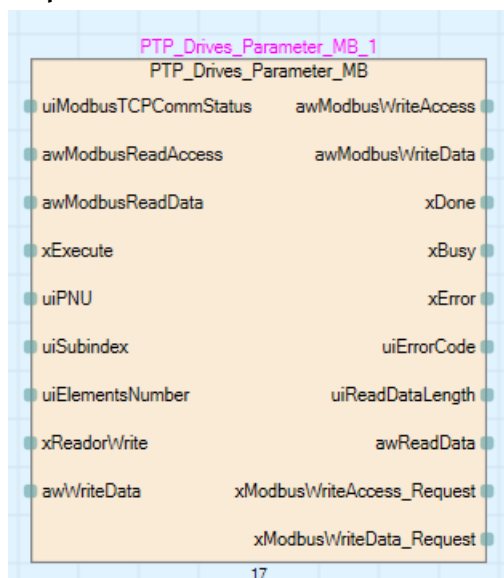
Step – 1



1. Open the “**Main Program**”.
2. Drag & Drop the “**PTP_Drives_Parameter_MB**” block from FB/FUN to program area.
3. The “**Label Registration**” tab will open.
4. Enter the name of the FB on the Label Name field.
5. Click on **OK** to confirm the FB name.

The called function block shown in below Step-2

Step – 2



Step – 3

Assigning the Communication Status Bit to PTP_Drives_Parameter_MB Function Block

Setting Item

Input the Setting Item to Search

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

Communication Destination File: Show All

Setting No.	Communication Pattern	Communication Setting Execution Interval(ms)	Source	Destination	Target PLC No.
1	Read	Fixed Interval	10 MODBUS/TCP(192.168.1.5)	Host Station(192.168.1.2)	Not Specified
2	Write	Fixed Interval	10 Host Station(192.168.1.2)	MODBUS/TCP(192.168.1.5)	Not Specified
3	Read	Fixed Interval	50 MODBUS/TCP(192.168.1.5)	Host Station(192.168.1.2)	Not Specified
4	Read	Fixed Interval	50 MODBUS/TCP(192.168.1.5)	Host Station(192.168.1.2)	Not Specified
5	Write	On Request	Host Station(192.168.1.2)	MODBUS/TCP(192.168.1.5)	Not Specified
6	Write	On Request	Host Station(192.168.1.2)	MODBUS/TCP(192.168.1.5)	Not Specified

- Under Element Selection
Select Module
- Under Module Label --> Expand - 0000:_RJ71EN71(E+IEF) --> Expand - EN71_EF_1
- Expand Ethernet(PORT1)
- Expand Simple PLC Communication area
 - Expand SimplePLC status setting no.[1 To 64]
 - Referring **simple CPU Communication setting No.**, select Simple PLC communication status (**wnStatus_SimplePLC_D[_]**) from “SimplePLC status setting no.[1 To 64]” tree.
- The yellow highlighted communication are **fixed interval** so the communication establishes on every execution interval of time.
- Drag & drop Simple PLC communication status (**wnStatus_SimplePLC_D[1]**) and link it to Variable Local tag (Status – Word [Signed]).

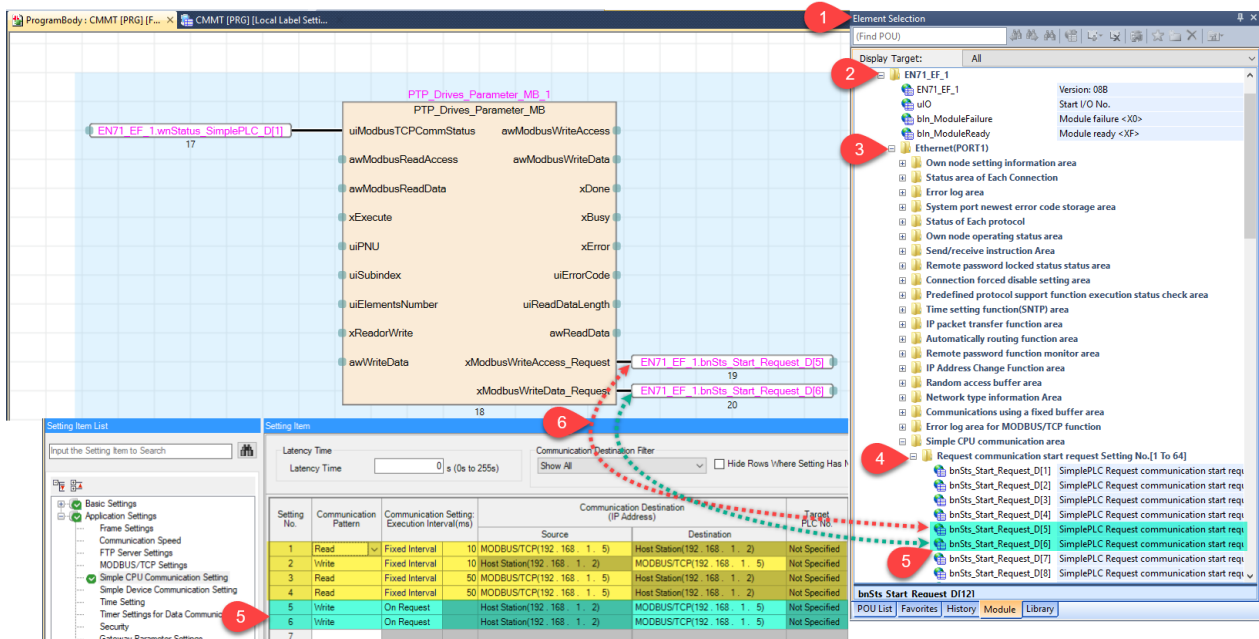


Note

- Point 5. - In Simple CPU Communication, if multiple Settings are assigned to read/write data with multiple CMMT drives then respective Setting No. to be used for Simple CPU Communication status.
- If communication is established successfully then value will be displayed 3.

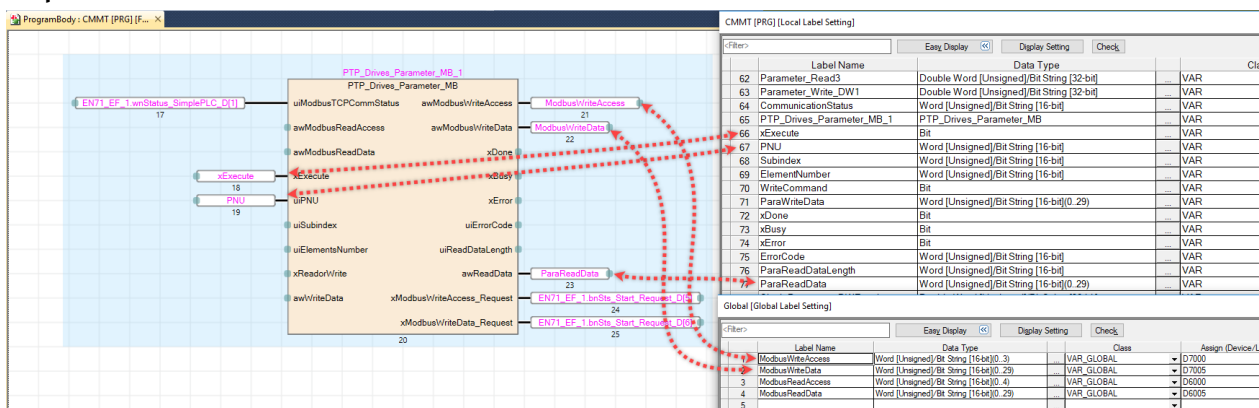
Step – 4

Assigning the Communication Start Request Bit to PTP_Drives Festo Parameter MB Function Block



- Under Element Selection Select Module
- Under Module Label --> Expand 0000:_RJ71EN71(E+IEF) --> Expand EN71_EF_1
- Expand Ethernet(PORT1)
- Expand Simple PLC Communication area
 - Expand Request Communication start request Setting no.[1 To 64]
 - Referring **simple CPU Communication setting No.**, select Request Communication start request Setting no. (**bnSts_Start_Request_D[]**) from “Request Communication start request Setting no.[1 To 64]” tree.
- The green highlighted communication are **on request** so the communication establishes only on triggering the communication request bit.
- Drag & link to xModbusWriteAccess_Request from Request Communication start request Setting no. (**bnSts_Start_Request_D[5]**).
Drag & link to xModbusWriteData_Request from Request Communication start request Setting no. (**bnSts_Start_Request_D[6]**).

Step – 5



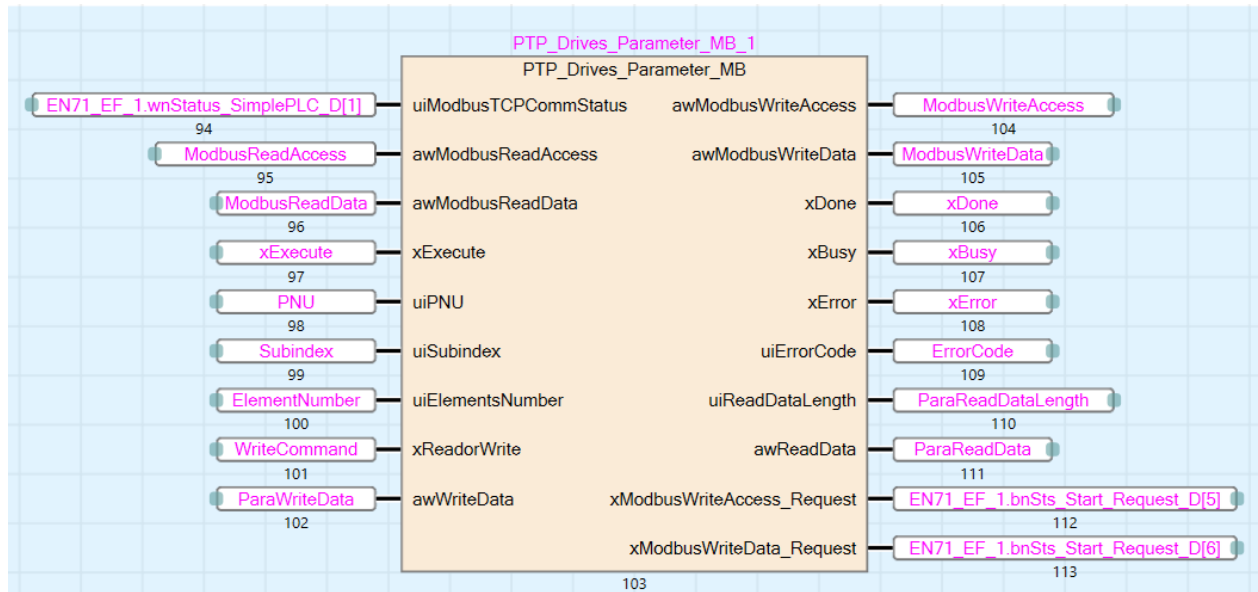
- Open the Local label to view the created local tags.
- Drag and Drop the xExecute tag from Local label to xExecute input tag in **PTP_Drives_Parameter_MB** function block.
- Drag and Drop the ModbusWriteAccess tag from Global label to awModbusWriteAccess input tag in **PTP_Drives_Parameter_MB** function block.



Note

- Repeat Step-5 to assign the different input & output tags in function block.

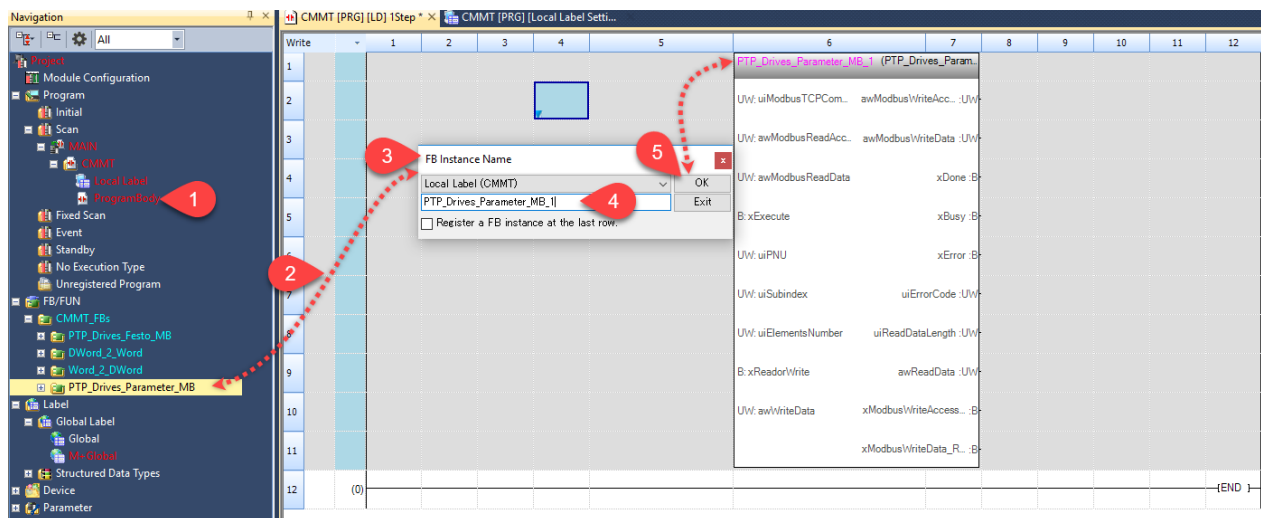
Once all steps completed successfully the **PTP_Drives_Parameter_MB** Function block will be look like below.



5.4 Configuring the PTP_Drives_Parameter_MB Function Block Input & Output Interface Tags in Ladder

Step – 1

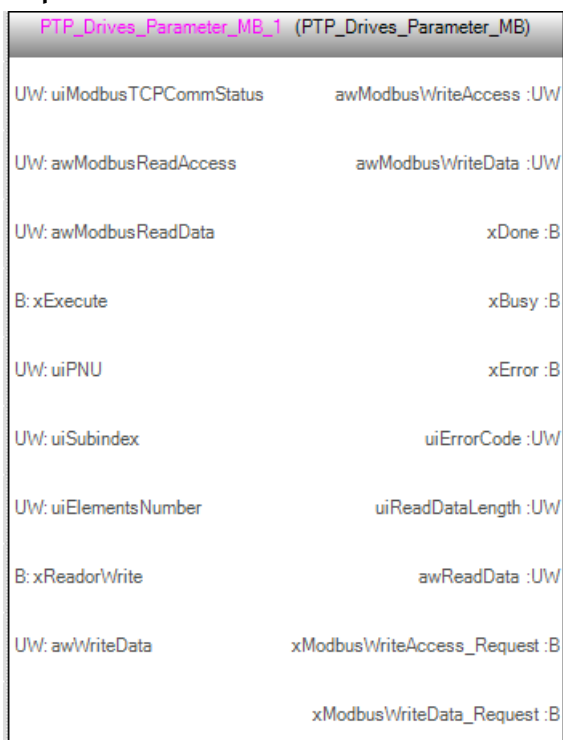
Library Configuration



1. Open the **“Main Program”**.
2. Drag & Drop the **“PTP_Drives_Parameter_MB”** block from FB/FUN to program area.
3. The **“Label Registration”** tab will open.
4. Enter the name of the FB on the Label Name field.
5. Click on **OK** to confirm the FB name.

The called function block shown in below Step-2

Step – 2



Step – 3

Assigning the Communication Status Bit to PTP_Drives_Parameter_MB Function Block

The screenshot shows the 'PTP_Drives_Parameter_MB' function block configuration. The variable declaration table lists various communication parameters. The communication settings table shows settings for communication No. 1 to 6. The tree view on the right shows the 'Simple PLC communication status' settings, with 'unStatus_SimplePLC_D[1]' highlighted in yellow. A red dotted line indicates the drag-and-drop action from the tree view to the variable declaration table.

Setting No.	Communication Pattern	Communication Setting Execution Interval(ms)	Source	Destination	Target PLC No.
1	Read	Fixed Interval	MODBUS/TCP/192.168.1.5	Host Station/192.168.1.2	Not Specified
2	Write	Fixed Interval	Host Station/192.168.1.2	MODBUS/TCP/192.168.1.5	Not Specified
3	Read	Fixed Interval	MODBUS/TCP/192.168.1.5	Host Station/192.168.1.2	Not Specified
4	Read	Fixed Interval	MODBUS/TCP/192.168.1.5	Host Station/192.168.1.2	Not Specified
5	Write	On Request	Host Station/192.168.1.2	MODBUS/TCP/192.168.1.5	Not Specified
6	Write	On Request	Host Station/192.168.1.2	MODBUS/TCP/192.168.1.5	Not Specified

- Under Element Selection
Select Module
- Under Module Label --> Expand - 0000:_RJ71EN71(E+IEF) --> Expand - EN71_EF_1
- Expand Ethernet(PORT1)
- Expand Simple PLC Communication area
 - Expand SimplePLC status setting no.[1 To 64]
 - Referring **simple CPU Communication setting No.**, select Simple PLC communication status (**wnStatus_SimplePLC_D[_]**) from "SimplePLC status setting no.[1 To 64]" tree.
- The yellow highlighted communication are **fixed interval** so the communication establishes on every execution interval of time.
- Drag & drop Simple PLC communication status (**wnStatus_SimplePLC_D[1]**) and link it to Variable Local tag (Status – Word [Signed]).

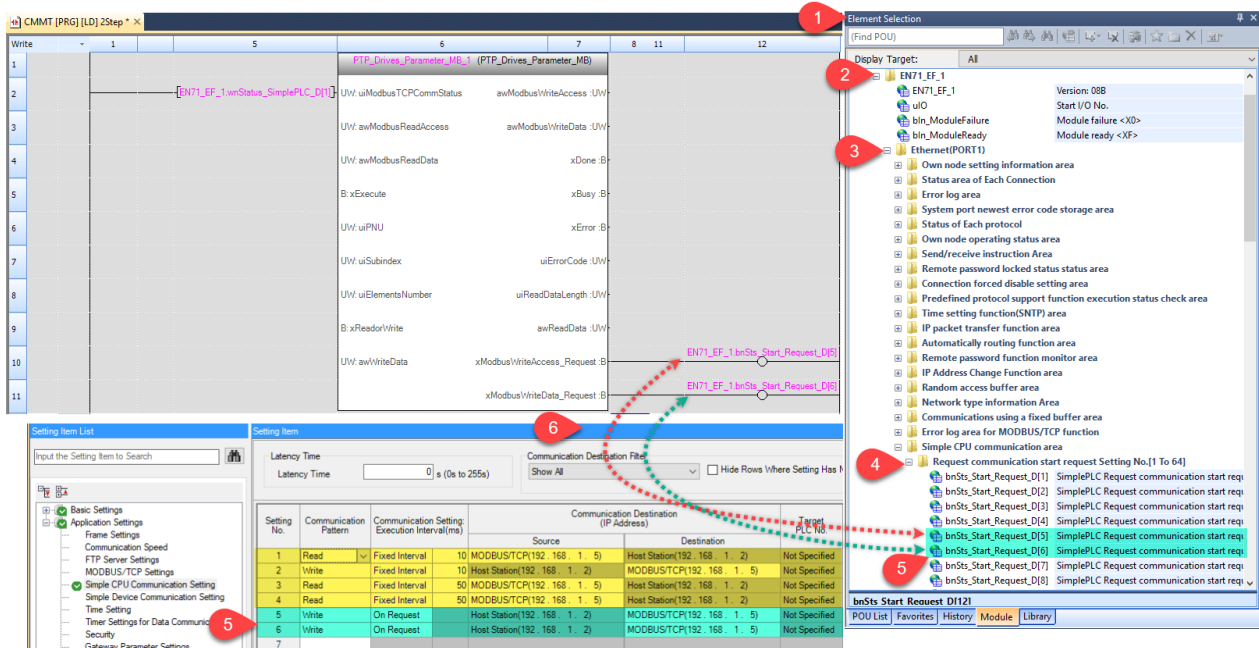


Note

- Point 5. - In Simple CPU Communication, if multiple Settings are assigned to read/write data with multiple CMMT drives then respective Setting No. to be used for Simple CPU Communication status.
- If communication is established successfully then value will be displayed 3.

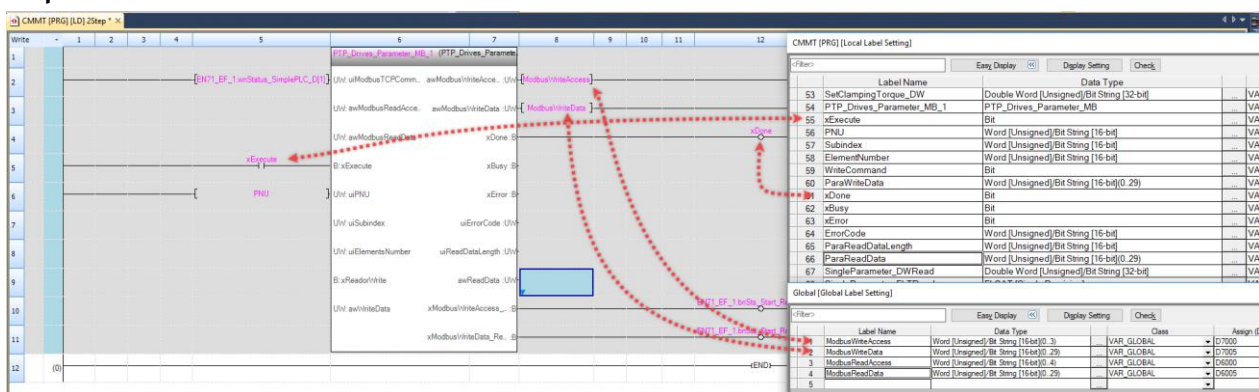
Step – 4

Assigning the Communication Start Request Bit to PTP_Drives_Festo_Parameter_MB Function Block



1. Under Element Selection
Select Module
2. Under Module Label --> Expand 0000:_R71EN71(E+IEF) --> Expand EN71_EF_1
3. Expand Ethernet(PORT1)
4. Expand Simple PLC Communication area
 - a) Expand Request Communication start request Setting no.[1 To 64]
 - b) Referring **simple CPU Communication setting No.**, select Request Communication start request Setting no. (**bnSts_Start_Request_D[]**) from “Request Communication start request Setting no.[1 To 64]” tree.
5. The green highlighted communication are **on request** so the communication establishes only on triggering the communication request bit.
6. Drag & link to xModbusWriteAccess_Request from Request Communication start request Setting no. (**bnSts_Start_Request_D[5]**).
Drag & link to xModbusWriteData_Request from Request Communication start request Setting no. (**bnSts_Start_Request_D[6]**).

Step – 5



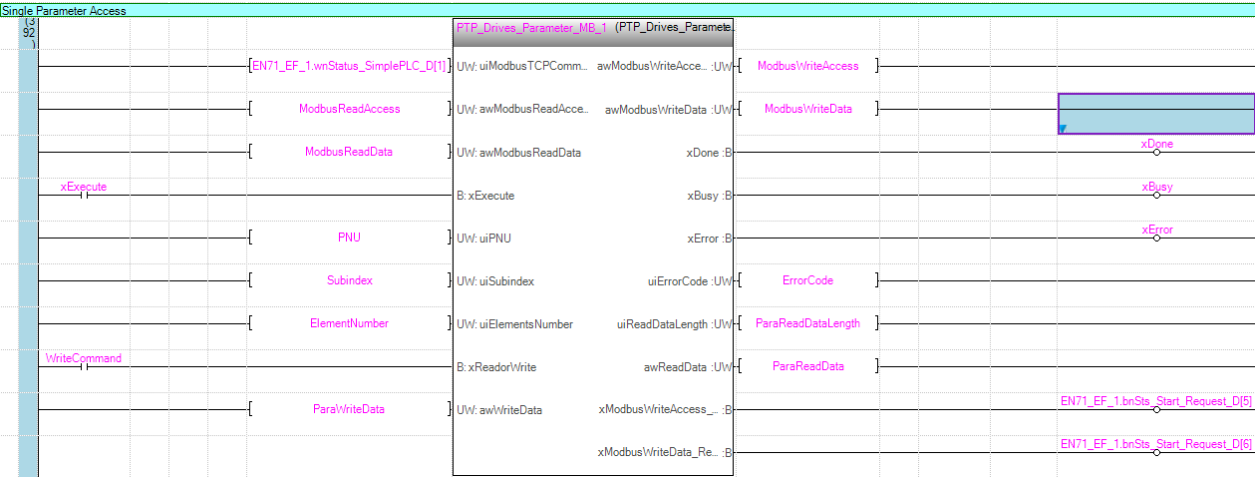
- Open the Local label to view the created local tags.
- Drag and Drop the xExecute tag from Local label to xExecute input tag in **PTP_Drives_Parameter_MB** function block.
- Drag and Drop the ModbusWriteAccess tag from Global label to awModbusWriteAccess input tag in **PTP_Drives_Parameter_MB** function block.



Note

- Repeat Step-5 to assign the different input & output tags in function block.

Once all steps completed successfully the **PTP_Drives_Parameter_MB** Function block will be look like below.



6 PTP_Drives_Festo_MB FUNCTIONS

This chapter describes about the configuration of **PtP_Drives_Festo_MB** Function Block for different operation modes, Enabling & Stopping the axis, Enabling hardware & software position limits as well as Actual speed monitoring.

6.1 Axis Enabling & Stopping

For axis enabling following tags configuration must be done.

- “**ConfigEPos**” Bit 0 & Bit1(OFF2 & OFF3) must be **TRUE**.
- “**AxisError**” & “**Lockout**” output tags must be **FALSE** to ensure drive is in ready state.
- “**ModePos**” value must be between 1 to 8.
- “**CancelTraversing**”(Reject traversing task) & “**IntermediateStop**”(intermediate stop) tags must be **TRUE**.
- “**EnableAxis**” must be **TRUE** & “**AxisEnabled**” feedback should become **TRUE** after enabling the drive.
- Toggle “**AckError**” control bit to clear any error in the Function Block.

6.2 Axis Software & Hardware Position Limits Enabling

“**ConfigEPOS tag’s Bit 2 & 3**” are used to enable the software & hardware position limits respectively

- “**ConfigEPOS.2 = TRUE**” This means software position limit is active (Festo Automation Suite Parameter **P1.112414140.0.0**)
- “**ConfigEPos.3 = TRUE**” This means hardware position limit is active (Festo Automation Suite Parameter **P1.112414150.0.0**)

When software position limit is active please pay attention if the below parameters are configured correctly as per application requirement.

The screenshot shows the 'Axis configuration' window. On the left is a sidebar with 'Axis' selected. The main area contains several settings:

- Reversing the direction of rotation:** A checkbox labeled 'Active' is currently unchecked.
- Axis zero point offset:** A text input field containing the value '0.03' with a unit 'r'.
- Software limit positions active:** A checkbox labeled 'Active' is checked.
- Negative software limit position:** A text input field containing the value '-0.03' with a unit 'r'.
- Positive software limit position:** A text input field containing the value '999.97' with a unit 'r'.

6.3 Axis Actual Speed Monitoring

User can monitor the actual speed of the axis in the variable “**ActVelocity**”. The actual speed of the axis is scaled from 0 to 100% depending on the base speed value of parameter number (**P1.11280701.0.0**). So it is mandatory to enter the same value in input tag “**BaseSpeedValue**” of the Function Block as like parameter number (**P1.11280701.0.0**).



Note

- Make sure the Function Block input tag “**BaseSpeedValue**” & Festo Automation Suite parameter number (**P1.11280701.0.0**) values are same. Function Block will display wrong velocity value if both values are not equal.

6.4 PTP_Drives_Festo_MB Modes of Operation

The PTP_Drives_Festo will control the drive in the following modes

- **ModePos – 1:** Relative Positioning
- **ModePos – 2:** Absolute Positioning
- **ModePos – 3:** Setup Mode
- **ModePos – 4:** Home/Reference using Festo Automation Suite Configuration
- **ModePos – 5:** Home to Current Position
- **ModePos – 6:** Traversing/Record Select Mode
- **ModePos – 7:** Jog
- **ModePos – 8 :** Inching/Incremental Jog Mode

6.4.1 Relative Positioning Mode [ModePos – 1]

1. Configurations for Relative Positioning Mode

- Set the “**ModePos**” value to 1.
- Check “**AxisRef**” status tag is True. If not true, the user could also use other modes like mode 4 or 5 to set the reference point for an axis.
- Enable axis by toggling “**EnableAxis**” command tag.
- “**CancelTraversing**” and “**IntermediateStop**” command tags must be in true state.
- Enter the percentage of Override acceleration, Override deceleration & Override velocity in “**OverAcc**”, “**OverDec**” & “**OverV**” respective command tags.
- The status tags “**AxisEnabled**” & “**AxisRef**” must be true and status tag “**Error**” must be false.

2. Sequence Command

- Enter the Position command value in “**Position**” tag & Velocity command value in “**Velocity**” tag. To move axis in forward direction, give position command in positive value & for moving axis in reverse direction give position command in negative value.
- Toggle the “**ExecuteMode**” command tag to start motion task for configured parameter. Rising edge of ExecuteMode tag will start the command. So, for every new position command tag want to toggle. Check the feedback tag “**ActMode**” has the same value as ModePos.
- User will receive execute command accepted handshake signal from “**AxisAckSetpoint**” status tag.
- If task is completed successfully, it is acknowledge by rising edge of “**AxisPosOk**” status tag. Also if task completed unsuccessfully with an error “**AxisError**” status variable will indicate it by going false to true.
- “**AxisSpFixed**” Status tag indicates the axis mobility status . If the axis is standby the “**Axis-SpFixed**” status tag will be high.



Note

- Refer the document “**CMMT-xx-EP PTP_Drives_Festo_MB Direct Mode 2 EtherNet/IP Timing Sequence**” document available with this download for execution command sequence.
- The existing running task command can be replaced by new task command only on **Mode 1 and 2**.

6.4.2 Absolute Positioning Mode [ModePos – 2]

1. Configuration for Absolute Positioning Mode

- Set the “**ModePos**” value to 2 .
- Check “**AxisRef**” status tag is True. If not true user could also use other modes like mode 4 or 5 to set the reference point for an axis.
- Enable axis by toggling “**EnableAxis**” command tag .
- “**CancelTraversing**” and “**IntermediateStop**” command tags must be in true state.
- Enter the percentage of Override acceleration, Override deceleration & Override velocity in “**OverAcc**”, “**OverDec**” & “**OverV**” respective command tags.
- The status tags “**AxisEnabled**” & “**AxisRef**” must be true. Also status tag “**Error**” must be false.

2. Sequence Command

- Enter the Position command value in “**Position**” tag & Velocity command value in “**Velocity**” tag. Give the absolute position value. The drive will decide forward & reverse direction based on its current position.
- Toggle the “**ExecuteMode**” command tag to start motion task for configured parameter. Rising edge of ExecuteMode tag will take for task start command. So for every new position command tag want to toggle. Check the feedback tag “**ActMode**” has the same value as ModePos.
- User will receive execute command accepted handshake signal from “**AxisAckSetpoint**” status tag.
- If commanded task is completed successfully, it is acknowledge by rising edge of “**AxisPosOk**” status tag. Also if task completed unsuccessfully with an error “**AxisError**” status tag will indicate it by going false to true.
- “**AxisSpFixed**” Status tag indicates the axis mobility status . If the axis is standby the “**Axis-SpFixed**” status tag will be high.



Note

- Refer the document “**CMMT-xx-EP PTP_Drives_Festo_MB Direct Mode 2 EtherNetIP Timing Sequence**” document available with this download for execution command sequence.
- The existing running task command can be replaced by new task command only on **Mode 1 and 2**.

6.4.3 Setup Mode [ModePos – 3]

In the setup mode, the drive is traversed at a constant speed by means of position control. There’s no need to specify a target position. It must be noted that continuous motion (e.g. conveyor belt, fan etc.) cannot be implemented in this mode.

1. Configuration for Setup Mode

- Set the “**ModePos**” value to 3.
- Check “**AxisRef**” status tag is True. If not true user could also use other modes like mode 4 or 5 to set the reference point for an axis.
- “**CancelTraversing**” and “**IntermediateStop**” command tags must be in true state.
- Enter the percentage of Override acceleration, Override deceleration & Override velocity in “**OverAcc**”, “**OverDec**” & “**OverV**” respective command tags.
- Select the axis moving direction by the inputs “**Positive**” or “**Negative**”.
- Enable axis by toggling “**EnableAxis**” command tag.

2. Sequence Command

- Motion dynamics are defined via the “**Velocity**”, “**OverV**”, “**OverAcc**” and “**OverDec**” inputs.
- Motion is started by means of a positive edge at “**ExecuteMode**”. The motion command is acknowledged via the “**AxisAckSetpoint**” output. A handshake mechanism can be programmed in this way. Check the feedback tag “**ActMode**” has the same value as ModePos.
- If a drive error has occurred during motion, this is indicated via the “**AxisError**” output. Any error occurring at the function block is indicated via the “Error output”.



Note

- To stop the axis movement. User has to use “**CancelTraversing**” or “**IntermediateStop**” command tags.

6.4.4 Home/Reference Point using Festo Automation Suite Configuration [ModePos – 4]

The reference point approach mode is used to moves the axis to home position.

1. Configuration for Reference Point Approach Mode

- Set the “**ModePos**” value to 4 . The status tags “**AxisEnabled**” must be true & “**Error**” must be in false to ensure no error.
- Axis must be in standstill state.

2. Sequence Command

- All the required parameters for homing will be considered from Automation Suite configuration in “Homing Parameters” like Homing mode, velocity, acceleration & deceleration.
- Toggle the “**ExecuteMode**” to start task. Check the feedback tag “**ActMode**” has the same value as ModePos.
- “**AxisRef**” will become true once the axis finds reference point. If an error occurs during traveling motion, The output signal “Error” will become true .

6.4.5 Home to Current Position [ModePos – 5]

This mode marks current position as reference point. This will be used when homing mode is selected as current position(37).

1. Configuration for Set Reference Point Mode

- Set the “**ModePos**” value to 5. The status tags “**AxisEnabled**” must be true & “**Error**” must be in false to ensure no error.
- Axis must be in standstill state.

2. Sequence Command

- Toggle the “**ExecuteMode**” to start task. Check the feedback tag “**ActMode**” has the same value as ModePos.
- “**AxisRef**” will become true after execute command. If an error occurs, The output signal “**Error**” will be true.

6.4.6 Traversing/Record Selection Mode [ModePos – 6]

Traversing Block mode is implemented via the “Record Table” of drive function. This mode retrieves parameters like Velocity, Position, Acceleration & Deceleration values from record table saved in the CMMT drive. It enables the creation of automatic programs, velocity mode, force mode, travel to fixed stop and outputs to be set and reset.

1. Configuration for Traversing Block Mode

- Set the “**ModePos**” value to 6.
- Enable axis by toggling “**EnableAxis**” command tag.
- Axis must be reference by toggling “**FlyRef**” command tag.
- “**CancelTraversing**” and “**IntermediateStop**” command tags must be in true state.
- Enter the percentage of Override acceleration, Override deceleration & Override velocity in “**OverAcc**”, “**OverDec**” & “**OverV**” respective command tags.
- The status tags “**AxisEnabled**” & “**AxisRef**” must be true. Also status tag “**Error**” must be false.

2. Sequence Command

- Select the record table by entering the table number in “**Position**” command tag. The value must be between 0 to 127.
- Toggle the “**ExecuteMode**” command tag to start motion task for configured parameter. Rising edge of ExecuteMode tag will start the command. So for every new position command tag want to toggle. Check the feedback tag “**ActMode**” has the same value as ModePos.
- User will receive execute command accepted handshake signal from “**AxisAckSetpoint**” status tag.
- If task is completed successfully, it is acknowledge by rising edge of “**AxisPosOk**” status tag. Also if task completed unsuccessfully with an error “**AxisError**” status tag will indicate it by going false to true.

- “**AxisSpFixed**” status tag indicates the axis mobility status . If the axis is standby the “**Axis-SpFixed**” status tag will be high.

**Note**

Refer the **Chapter – 4 from the CMMT user manual** to know the behaviour of the drive in different operation modes. And also (Refer [Chapter – 9](#))

6.4.7 Jog Mode [ModePos – 7]

1. Configuration for Jog Mode

- Set the “**ModePos**” value to 7 & check the feedback tag “**ActMode**” has the same value as ModePos.
- Enable axis by toggling “**EnableAxis**” command tag.
- Axis must be in standstill .
- Velocity, Acceleration & Jerk values are taken from Automation Suite configuration under **Jog Mode** tab.

2. Sequence Command

- “**Jog1**” & “**Jog2**” tags are used for Positive & Negative direction commands respectively.
- The “**ExecuteMode**” signal is **not required** for jog mode.

**Note**

Refer the **Chapter – 4 from the CMMT user manual** to know the behaviour of the drive in different operation modes.

6.4.8 Inching/Incremental Jog Mode [ModePos – 8]

Incremental jog mode will not work like ordinary jog mode. It will rotate the axis step by step forward or reverse based on direction selection, But ordinary jog mode rotates the axis continuously. All the configuration and command configurations are same as **ModePos – 7** but only difference is that **ModePos** value must be 8 for incremental jog mode.

**Note**

- The inching distance of positive & negative direction will be based on values set in parameters Relative Position Jog1 (P1.214530.0.0) & Relative Position Jog2 (P1.214538.0.0) respectively.

6.4.9 Procedure to set the CMMT-AS in “Travel to fixed Stop” mode with the Ptp_Drive_Festo_MB_ function block

- Using a positioning mode absolute or relative (ModePos = 1 or 2 (positioning absolute/relative))

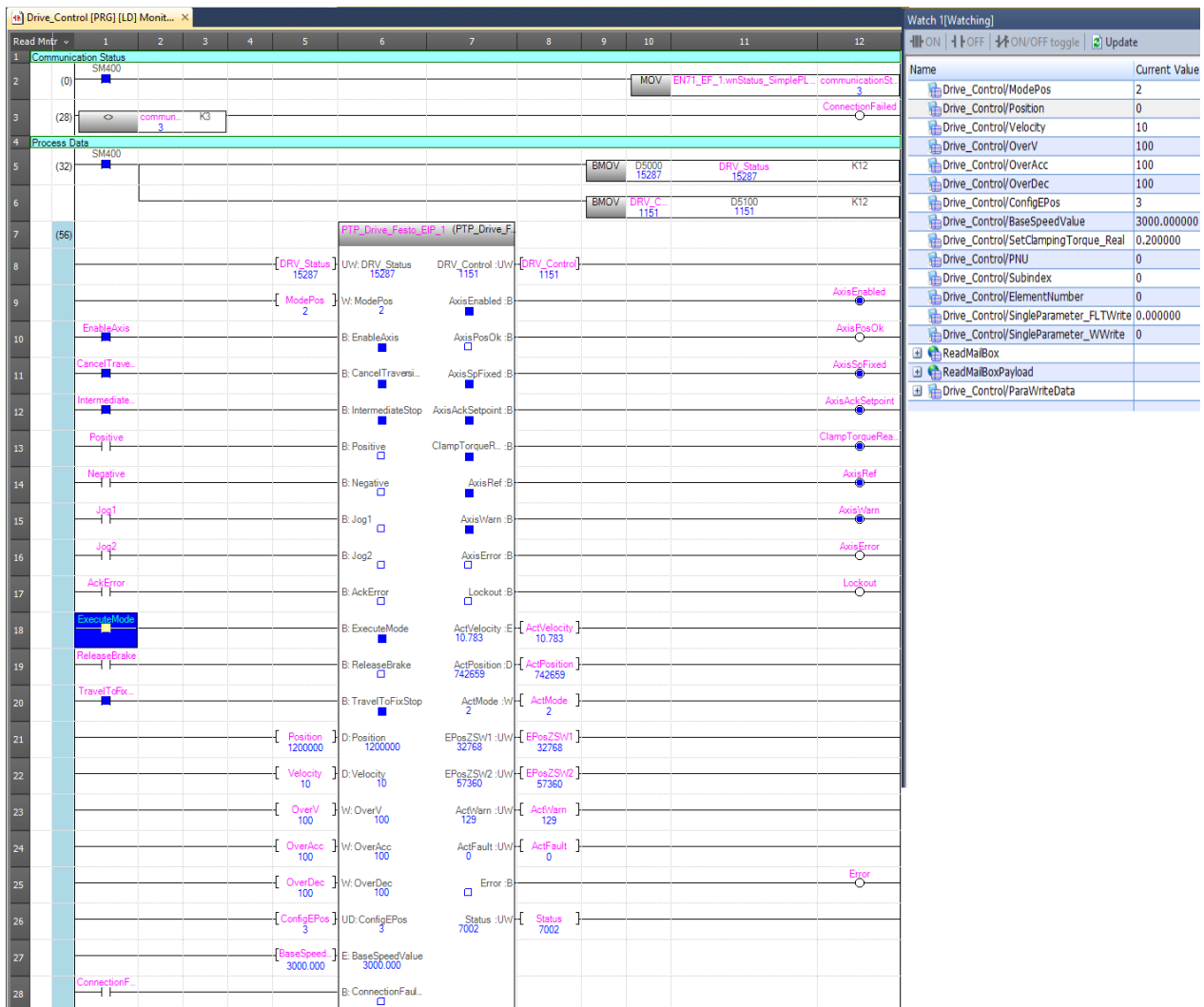
Travel to fixed Stop using Absolute Positioning Mode

1. Configuration for Travel to fixed Stop using Absolute Positioning Mode

- Set the “**ModePos**” value to 2 & check the output tag “**ActMode**” has the same value as ModePos.
- Enable the axis by toggling “**EnableAxis**” input tag.
- Check “**AxisRef**” status tag is True. If not true user could also use other modes like mode 4 or 5 to set the reference point for an axis.
- “**CancelTraversing**” and “**IntermediateStop**” input tags must be **TRUE**.
- Enable the axis by toggling “**TravelToFixStop**” input tag.
- Enter the percentage of Override acceleration, Override deceleration & Override velocity in “**Over-Acc**”, “**OverDec**” & “**OverV**” respective input tags.
- The status tags “**AxisEnabled**” & “**AxisRef**” must be **TRUE**. Also status tag “**Error**” must be **FALSE**.
- Set the “**Clamping torque**” as per user defined, this is taken from the EPD.

2. Sequence Command

- Enter the Position command value in **“Position”** tag & Velocity command value in **“Velocity”** tag. Give the absolute position value. The drive will decide forward & reverse direction based on its current position.
- Toggle the **“ExecuteMode”** command tag (Rising Edge) to start a motion task using the configured parameters (Every new position task would require a toggle of the **“ExecuteMode”** command tag).
- User will receive execute command accepted handshake signal from **“AxisAckSetpoint”** status tag.
- If the task is successfully completed, it is acknowledge by rising edge of **“AxisPosOk”** status tag. Also if task is unsuccessful **“AxisError”** output tag will be **TRUE..**
- **“AxisSpFixed”** Status tag indicates the axis mobility status . If the axis is standby the **“Axis-SpFixed”** status tag will be high.
- User will receive Clamping torque reached signal from **“ClampTorqueReached”** status tag.



Note

- If Clamping torque is reached before the position is reached then there will be no error displaced



Information

when activating the “Travel to fixed stop” mode than the following error will be disabled automatically.

The feedback can be read out on EPosZSW2 Bit14 or in ZSW2 Bit8. The PTP_Drives_Festo_MB FB has the output interface EPosZSW1 and EPosZSW2. That means the feedback can be read-out easily at Bit14 on EPosZSW2.

The PTP_Drives_Festo_MB, does not provide an input for a torque setpoint. The torque setpoint is called Clamping Torque [Nm] and can be found at P1.526801.0.0 in Festo Automation Suite.

The max. allowed clamping torque is related to the “User defined limits” P1.852.0.0 for Torque: lower limit value and P1.853.0.0 for Torque: upper limit value in tab Axis, see below:

Axis	
Homing Method	
Method	
Move to axis zero point after homing	☒ Active
Activation Save zero offset	☐ Active
Axis Configuration	
Reversing the direction of rotation	☐ Active
Axis zero point offset	3.00 mm
Software limit positions active	☐ Active
Negative software limit position	-3.00 mm
Positive software limit position	97.00 mm
User Defined Limits	
Velocity: lower limit value	-16.66667 m/s
Velocity: upper limit value	16.66667 m/s
Torque: lower limit value	-0.5472 Nm
Torque: upper limit value	0.5472 Nm
Active current: lower limit value	-3.20 Arms
Active current: upper limit value	3.20 Arms
Homing Parameters	
Crawl	Velocity [m/s]: 0.005, Accel. [m/s²]: 1.00, Jerk [m/s³]: 100.00
Search	Velocity [m/s]: 0.01, Accel. [m/s²]: 1.00, Jerk [m/s³]: 100.00
Running	Velocity [m/s]: 0.05, Accel. [m/s²]: 1.00, Jerk [m/s³]: 100.00
Homing timeout	60.00 s
Stop Deceleration	
Deceleration (stop ramp)	20.00 m/s²
Jerk (stop ramp)	1000.00 m/s³

Above image taken from FestoAutomationSuite.

User can change the clamping torque parameter value using EPD cyclically. ([Refer Chapter 9](#))

7 PTP_Drives_Parameter_MB FUNCTIONS

The PTP_Drives_Parameter_MB Function Block is used for single parameter read and write. In this chapter we explain with some examples on how to read and write CMMT parameters using this Function Block.

Call the **PTP_Drives_Parameter_MB** Function Block.



7.1 Identification of Parameter Number, SubIndex, Elements Number during PTP_Drives_Parameter_MB Configuration

- For reading or writing a CMMT drive parameter we need 3 important data namely:
 - Parameter Number (PNU)
 - SubIndex of the PNU
 - Elements Number to be read.
- The above 3 values can be obtained from the **PNUs Reference List table from CMMT drive user manual**.
- Suppose our Function Block has to read the **Support Point torque** of the CMMT drive. From the CMMT user manual we get the following information.

	PNU	Name	Data type	Access	Parameter
Parameter Number	11255.0 ...	Support point velocity [rad/s]	REAL	rw	P1.976.0.0 ... 15
	15				
Subindex and Elements Number	11256.0 ...	Support point torque [Nm]	REAL	rw	P1.977.0.0 ... 15
	15				
Subindex of the PNU	11257.0	Number of support points	UDINT	rw	P1.978.0.0

- From the below example we come to know that to read **Support Point torque** we need the following parameter.
 - Parameter Number = 11256
 - Subindex of the PNU = 0 or Specific Number from 0 to 15
 - Elements Number = 1 or For PNUs with multiple Sub Indexes, if the user wants to read as more than one Subindex of the PNU then specify the Starting Subindex number and the appropriate Elements Number.

-

- Parameters whose **Access is “RO”** can only be read and a new value to the parameter cannot be written using the PTP_Drives_Festo_Parameter.
- Parameters whose **Access is “RW”** can be read/write and a new value to the parameter can be written also using the PTP_Drives_Festo_Parameter.

1. Configuration for Reading Parameter

- ## 2. Sequence Command

- 

- For detailed information on the working of input and output tags, Refer [Chapter 4.2](#)

7.3 Write Parameter

1. Configuration for Writing Parameter

- **xReadorWrite** input tag must be False.
- Enter parameter number in “**uiPNU**”, parameter Subindex in “**uiSubindex**” and “**Elements Number**”.
 - (Refer CMMT Manual - Chapter 13.5 PNUs reference list & Chapter 7.1 for Elements Number)
- Enter the parameter value in “**awWriteData**” tag, the variable which holds the Write data, here Data is an ARRAY of Word.
 - If write Data is DWord or Float then Call the **Word_2_DWord** Function Block
 - (Refer Chapter 8.4 for knowing how to enter Float value or DWord)

2. Sequence Command

- Toggle the WriteCommand_FLT tag which enable to moves the Float or DWord value to array of awWriteData. (Every new write should require a toggle of the “WriteCommandFLT” tag).
or
- Toggle the WriteCommand_W tag which enable to moves the Word value to array of awWriteData. (Every new write should require a toggle of the “WriteCommandFLT” tag).
 - (Refer [Chapter 8.6](#) for knowing how to enter Float value, DWord and Word)
- Toggle the “**xReadorWrite**” input tag. (Every new write should require a toggle of the “**xReadorWrite**” tag).
- The “**xExecute**” command tag (Rising Edge) to start Writing parameter (Every new task would require a toggle of the “**xExecute**” tag).
- “**xBusy**” status tag indicates the task has been started.
- If the task is successfully completed, it is acknowledge by rising edge of “**xDone**” status tag. Also if task is unsuccessful “**xError**” output tag will be TRUE and “**wErrorID**” shows the error code.



Note

- For writing the Data, both configuration for writing parameter and sequence command should be followed as given in order.
- For detailed information on the working of input and output tags, [Refer Chapter 4.2](#)

8 Sample code for Control CMMT-AS Drive using PTP_Drives_Festo_MB Funtion Block

8.1 Sample Code: Axis Initialization

Before	PTP_Drive_Festo_MB.1 (PTP_Drive_Festo_MB)		After	PTP_Drive_Festo_MB.1 (PTP_Drive_Festo_MB)	
EN171_EF_1.vgStatus_Sim 3	U/v: uModbusTCPComm... 3	DRV_Control :U/v 1032	EN171_EF_1.vgStatus_Sim 3	U/v: uModbusTCPComm... 3	DRV_Control :U/v 1038
DRV_Status 8768	U/v: DRV_Status 8768	AxisEnabled :B ☐	DRV_Status 8763	U/v: DRV_Status 8763	AxisEnabled :B ☐
ModePos 0	I/v: ModePos 0	AxisPosOk :B ☐	ModePos 4	I/v: ModePos 4	AxisPosOk :B ☐
EnableAxis	B: EnableAxis ☐	AxisSpFixed :B ☒	EnableAxis	B: EnableAxis ☒	AxisSpFixed :B ☒
CancelTraversing	B: CancelTraversing ☐	AxisAckSetpoint :B ☐	CancelTraversing	B: CancelTraversing ☐	AxisAckSetpoint :B ☐
IntermediateStop	B: IntermediateStop ☐	ClampTorqueReached :B ☐	IntermediateStop	B: IntermediateStop ☐	ClampTorqueReached :B ☐
Positive	B: Positive ☐	AxisRef :B ☐	Positive	B: Positive ☐	AxisRef :B ☐
Negative	B: Negative ☐	AxisWarn :B ☐	Negative	B: Negative ☐	AxisWarn :B ☐
Jog1	B: Jog1 ☐	AxisError :B ☐	Jog1	B: Jog1 ☐	AxisError :B ☐
Jog2	B: Jog2 ☐	Lockout :B ☒	Jog2	B: Jog2 ☐	Lockout :B ☒
AckError	B: AckError ☐	ActVelocity :E 31155.000	AckError	B: AckError ☐	ActVelocity :E -0.689
ExecuteMode	B: ExecuteMode 722570	ActPosition :D 722570	ExecuteMode	B: ExecuteMode 722570	ActPosition :D 722570
ReleaseBrake	B: ReleaseBrake 0	ActMode :I/v 0	ReleaseBrake	B: ReleaseBrake 0	ActMode :I/v 0
TravelToFixStop	B: TravelToFixStop 0	EPosZSV1 :U/v 0	TravelToFixStop	B: TravelToFixStop 0	EPosZSV1 :U/v 0
Position	D: Position 0	EPosZSV2 :U/v 1	Position	D: Position 0	EPosZSV2 :U/v 1
Velocity	D: Velocity 0	ActWarn :U/v 0	Velocity	D: Velocity 0	ActWarn :U/v 0
OverV	I/v: OverV 0	ActFault :U/v 0	OverV	I/v: OverV 100	ActFault :U/v 0
OverAcc	I/v: OverAcc 0	Error :B 8402	OverAcc	I/v: OverAcc 100	Error :B 7002
OverDec	I/v: OverDec 0	Status :U/v 8402	OverDec	I/v: OverDec 100	Status :U/v 7002
ConfigEPos	UD: ConfigEPos 0		ConfigEPos	UD: ConfigEPos 3	
BaseSpeedValue	E: BaseSpeedValue 0.000		BaseSpeedValue	E: BaseSpeedValue 3000.000	

At first scan of PLC, the PTP_Drives_Festp_MB FB shows an “Error” and “Lockout”.

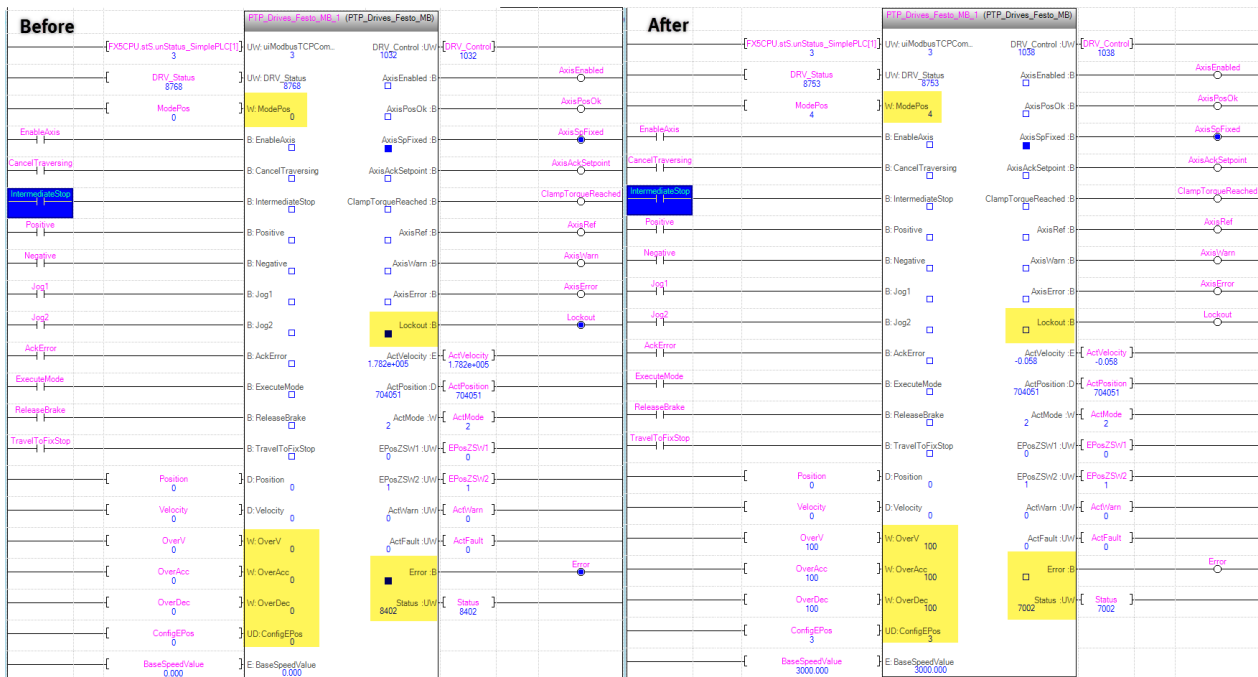
- To clear the Error and Lockout follow the steps below.
Enter the value in FB
 - ModePos := 5 [Home using FAS Configuration [ModePos – 4]]
 - CancelTraversing := TRUE
 - IntermediateStop := TRUE
 - OverV := 100
 - OverAcc := 100
 - OverDec := 100
 - ConfigEPos := 3 or 15 ([Refer ConfigEPos Bit Description](#))
 - BaseSpeedValue := 3000.0

After entering the value, PTP_Drives_Festo_MB FB does not show an “Error” and “Lockout”.

- To enable the axis, toggle **EnableAxis** control tag from false to true.
Feedback value in FB
 - AxisEnabled := TRUE
 - AxisPosOk := TRUE
 - AxisSpFixed := TRUE
 - Lockout := FALSE
 - Error := FALSE
 - Status := 7002

In case if Error Bit is True, enable “AckError” bit to clear the Error.

8.2 Sample Code: Axis Homing



- Toggle the **ExecuteMode** control tag from false to true. If axis homing done successfully then **AxisRef** status tag will update the status by turning into true.

8.3 Sample Code: Axis Absolute Mode

Sample code for Control CMMT-AS Drive using PTP_Drives_Festo_MB Funtion Block



- To run the drive in **Absolute Mode** follow the steps below.
Enter the value in FB
 - ExecuteMode := FALSE
 - ModePos := 2 [Absolute Positioning Task [ModePos – 2]]
 - Position := 10000000 (For this sample code. Refer Step 2 of [Chapter 2.2.4](#) for Position Calculation)
 - Velocity := 5000 (For this sample code. Refer Step 2 of [Chapter 2.2.4](#) for Velocity Calculation)
- Once entered the values toggle the **ExecuteMode** control tag from false to true. Axis will start to run in **Absolute Mode**.
Feedback value in FB
 - ActVelocity := Actual Velocity value feedback from drive
 - ActPosition := Actual Position value feedback from drive
 - ActMode := 2
- If axis move absolute mode done successfully then **AxisAckSetpoint** status tag will update the status by turning into true.

8.4 Sample Code: Axis Jog Mode



- To run the drive in **Jog Mode** follow the steps below.
Enter the value in FB
 - ExecuteMode := FALSE
 - ModePos := 7 [Jog Mode [ModePos – 7]]
- Once entered the values toggle the **Jog1 or Jog2** control tag from false to true. Axis will start to run in **Jog Mode**.
Feedback value in FB
 - AxisPosOk := FALSE
 - AxisSpFixed := FALSE
 - ActVelocity := Actual Velocity value feedback from drive
 - ActPosition := Actual Position value feedback from drive
 - ActMode := 7
- Jog mode will stop by toggling the **Jog1 or Jog2** control tag from true to false.



Jog Mode

The control tag **Jog1** is used to jog the axis in positive direction & the control tag **Jog2** is used to jog the axis in negative direction. For this sample code only positive direction jog is shown.

Consider we want to read/write the parameter named “ **Clamping Torque**”.

Single Parameter Access				Name				Current Value	Display F
9							Drive_Control/ModePos	7	Decimal
							Drive_Control/Positen	10,000,000	Decimal
							Drive_Control/Velocety	5,000	Decimal
							Drive_Control/OverV	100	Decimal
							Drive_Control/OverAcc	100	Decimal
							Drive_Control/OverDec	100	Decimal
							Drive_Control/ConfigEPoS	3	Decimal
							Drive_Control/BaseSpeedValue	3000.000000	--
							Drive_Control/SetClampingTorque_Real	0.200000	--
							Drive_Control/PNU	12,168	Decimal
							Drive_Control/Subindex	0	Decimal
							Drive_Control/ElementNumber	1	Decimal
							Drive_Control/SingleParameter_RTWWrite	0.000000	Decimal
							Drive_Control/SingleParameter_WWrite	0	Decimal
							Drive_Control/ParaWriteData	--	Decimal
							Drive_Control/ParaReadData	--	Decimal
							[0]	52,429	Decimal
							[1]	15,948	Decimal
							[2]	0	Decimal
							[3]	0	Decimal
							[4]	0	Decimal
							[5]	0	Decimal
							[6]	0	Decimal
							[7]	0	Decimal
							[8]	0	Decimal
							[9]	0	Decimal
							[10]	0	Decimal
							[11]	0	Decimal
							[12]	0	Decimal
							[13]	0	Decimal
							[14]	0	Decimal
							[15]	0	Decimal
							[16]	0	Decimal

Single Parameter Read Data

(51 8)

Word_2_DWord_1 (Word_2_...)

ReadMailBoxPayload[1] 15948 UW: WONE TwoWO...UD 1045220557 SingleParameter_DWordRead 1045220557

ReadMailBoxPayload[0] 52429 UW: WZERO 52429

(56 6) SM400 BMOV SingleParameter_D... 1045220557 SingleParameter_F... 0.200 K2

- Page 94 of 126 Application Note – CMMT Commissioning With MITSUBISHI IQ-R PLC PLC By MODBUS TCP-1.20

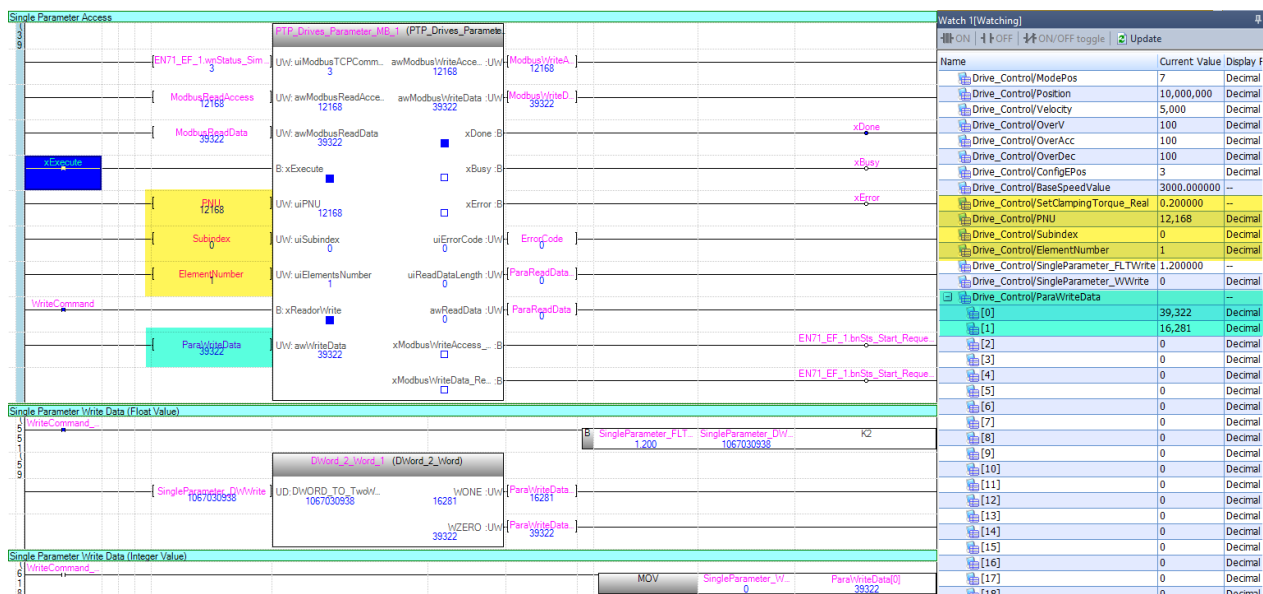
8.6 Sample code: Write parameter using PTP_Drives_Festo_Parameter_MB Function Block.

Consider we want to read/write the parameter named “Clamping Torque”.

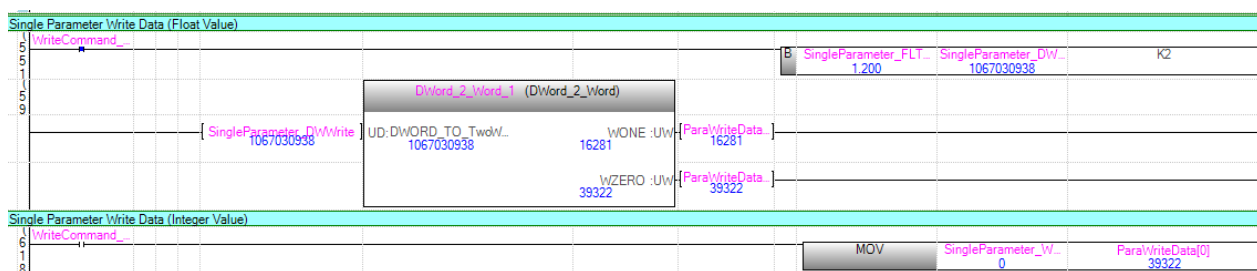
PNU	Name	Data type	Access	Parameter
12168.0	Clamping torque	REAL	rw	P1.526801.0.0

Parameter
Number
(PNU)

Parameter
Subindex



In Order to Write data type FLOAT, call the **DWord_2_Word** Function Block



- Enter the Input Real Value Output in BMOV block and get Double Word Value Output.
- Enter the DWord Output value in **DWord_2_Word** and get Array of Word Value Output.

9 Extended Process Data

9.1 Set parameters in the FAS under Extended Process Data (EPD)



Information

Maximum of 32 Number of bytes can be called in Sent Data and Received Data

- Before adding the EPD to PLC, list out the Send and Receive Data (EPD) and byte position to modify the Read Holding Registers and Write Holding Registers
- For Instance: For Torque Control Mode, user has to add following EPD

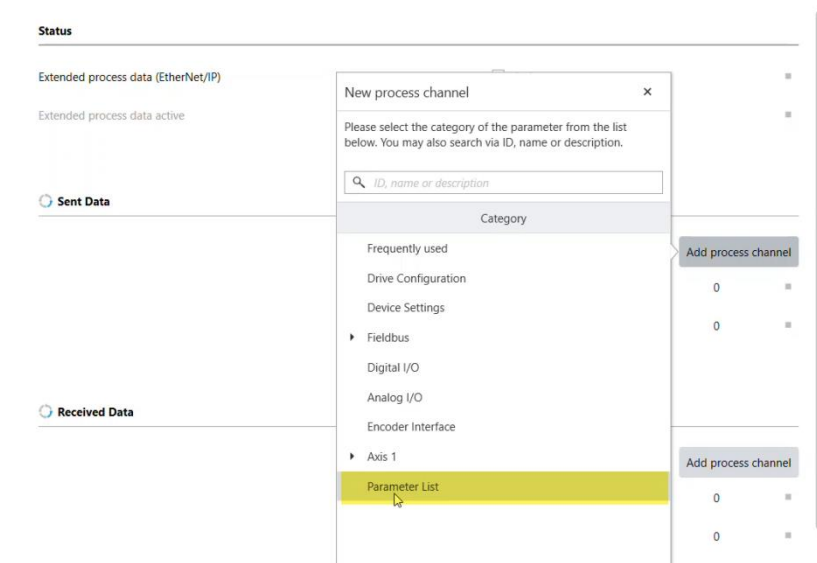
Step – 1

The screenshot shows the Mitsubishi FAS software interface. The top bar includes tabs for PARAMETERIZATION, CONTROL, and DIAGNOSIS. Below the top bar, there's a status area showing 'CMMT-AS-C2-3A-MP-S1' and a 'Connect' button. The main area is divided into a left sidebar for 'Parameter Pages' and a main content area for 'Extended Process Data'. The sidebar shows a tree structure with 'PROFIdrive' expanded under 'Fieldbus'. The main content area has sections for 'Status', 'Sent Data', and 'Received Data'. Red numbered callouts (1-5) indicate the steps: 1. Expand 'PROFIdrive' in the sidebar. 2. Click on 'Extended Process Data' in the sidebar. 3. Check the 'Active' checkbox for 'Extended process data (EtherNet/IP)'. 4. Click 'Add Process Channel' for 'Sent Data'. 5. Click 'Add Process Channel' for 'Received Data'.

1. In **Parameter pages** expand **PROFIdrive** under Profile tree.
2. Click on the **Extended Process. Data.**
3. Enable Extended Process Data (EtherNet/IP)
4. Under the **Sent Data**, Click **Add process channel**
5. Under the **Received Data**, Click **Add process channel**

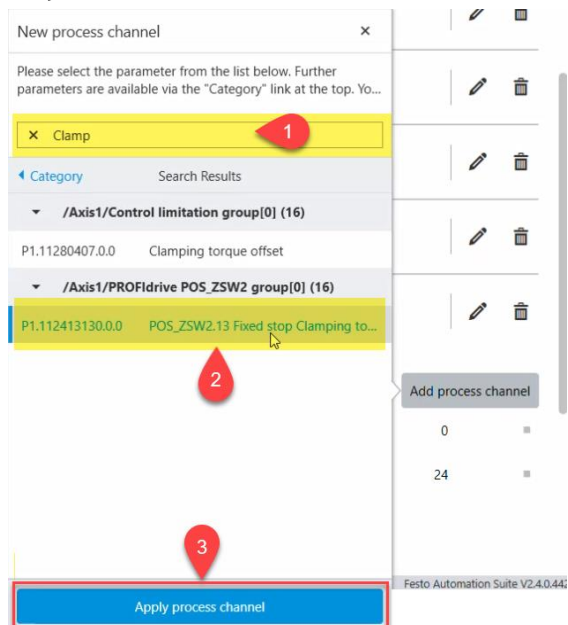
Step – 2

Sent Data – This will send the data to the PLC.



- Select **Parameter List**.

Step – 3



1. Under the search option, either enter the **P1 number** or enter the **name of the parameter**.
2. Select the P1 / Parameter which needs to be added in EPD.
3. Click on Apply process channel or Double click on the parameter that will be added to the channel.

Extended Process Data :- Sent Data from FAS (Read Data)

ID	Parameter	Type	Byte Position
P1.2220.0.0	Setpoint value torque	REAL	0
P1.150.0.0	Actual torque value motor shaft	REAL	4
P1.526801.0.0	Clamping Torque	REAL	8

Extended Process Data :- Received Data to FAS from PLC (Receive Data)

ID	Parameter	Type	Byte Position
P1.526821.0.0	Clamping Torque	REAL	0

After adding the EPD.

Extended Process Data Delete All Sent Data Delete All Received Data Active

Extended process data (EtherNet/IP) Active

Extended process data active Active

Sent Data

ID	Parameter	Type	Byte position	Function Block ID
0	P1.2220.0.0 Setpoint value torque	REAL	0	9
1	P1.150.0.0 Actual torque value motor shaft	REAL	4	9
2	P1.526801.0.0 Clamping torque	REAL	8	9

Add Process Channel

Number of bytes Tx 0

Number of bytes Tx (Offline) 12

PLC output configuration 999

Received Data

ID	Parameter	Type	Byte position	Function Block ID
0	P1.526801.0.0 Clamping torque	REAL	0	9

Add Process Channel

Number of bytes Rx 0

Number of bytes Rx (Offline) 4

PLC input configuration 9



Note

- The data type and the byte position to be noted to read and write in the PLC program.

Step – 4



Store on Device



Reinitialize

- Click the **“Store on device”** button to store all the parameters on drive memory.
- Click the **“Reinitialize”** the button to initialize the drive.



Note

- The triangle warning symbol on both buttons indicates action required.
- All the modified parameters will automatically erase from drive during power restart If modified parameters are not stored in the drive.
- Its mandatory to reinitialize the drive for some parameter modification. If drive is not reinitialized the drive performance remains same as like previous configuration.

9.2 Add the EPD in PLC Logic



Information

Maximum of 32 Number of bytes can be called in Sent Data and Received Data

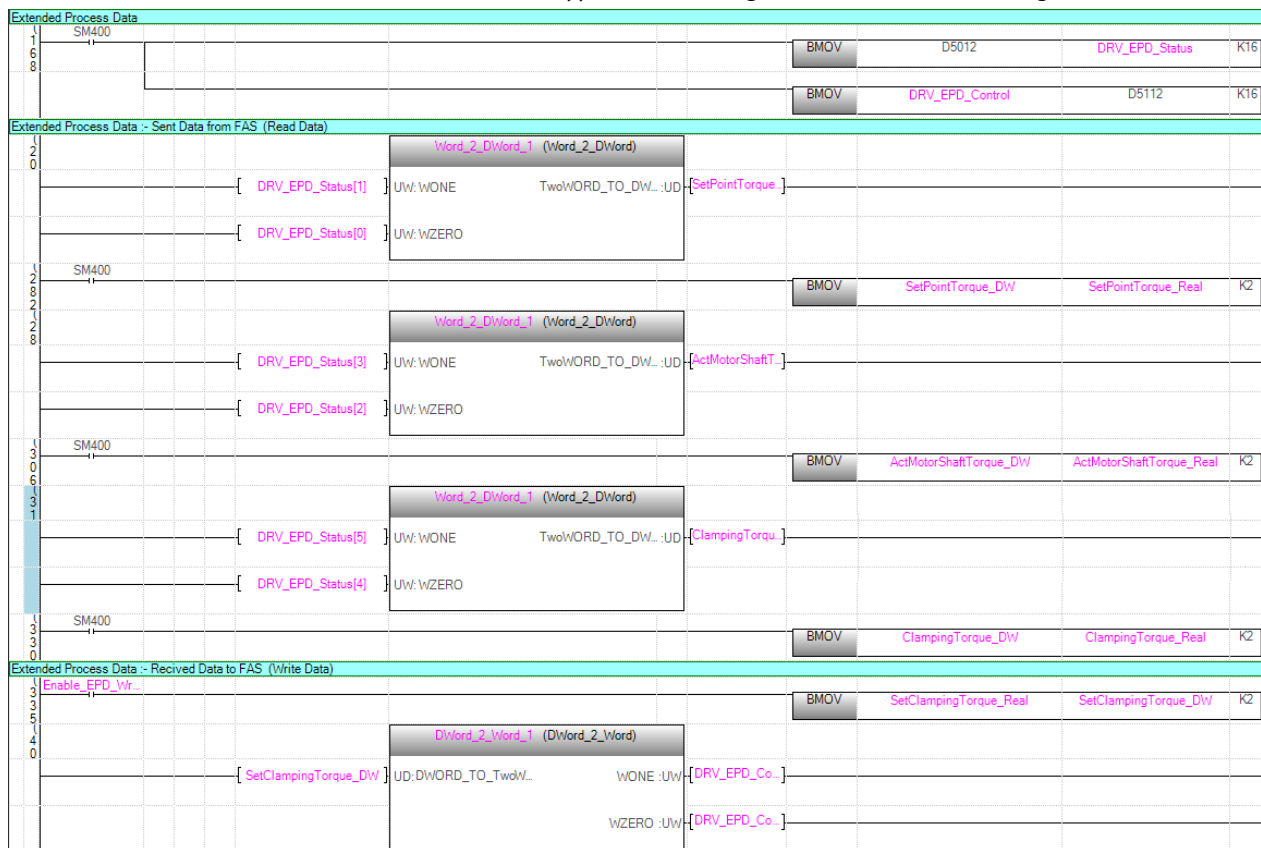


Note

- For PTP_Drives_Festo_MB FB (Position Mode) 12 Word Data Read and Data Write is used.
- User has to use from ReadData[12] & WriteData[12] to read and write the EPD. If Data is Boolean user need to extract the data from the respective word position.
- Maximum ReadData[27] & WriteData[27] can be used which includes Process Data and EPD.

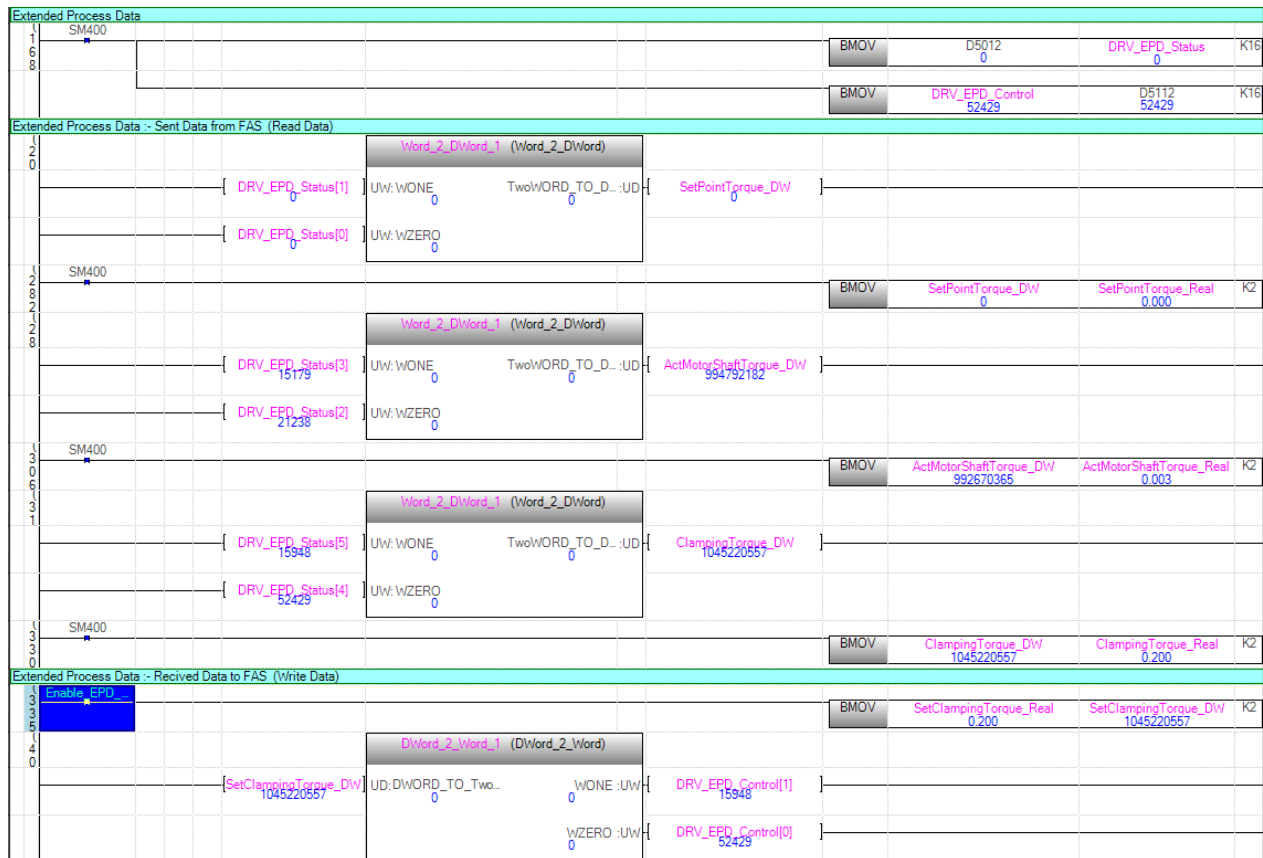
9.2.1 Logic to Read and Write the EPD

- Based on the Number of EPD & the Data type, write the logic as shown in below image.



- Read EPD value
 - Use BMOV Function Element to convert DWord to Float value.
- Write EPD value
 - Use BMOV Function Element to convert Float value to DWord.
 - Use the DWord_2_Word Function Block to convert DWord to Word, so that value can be written in WORD.

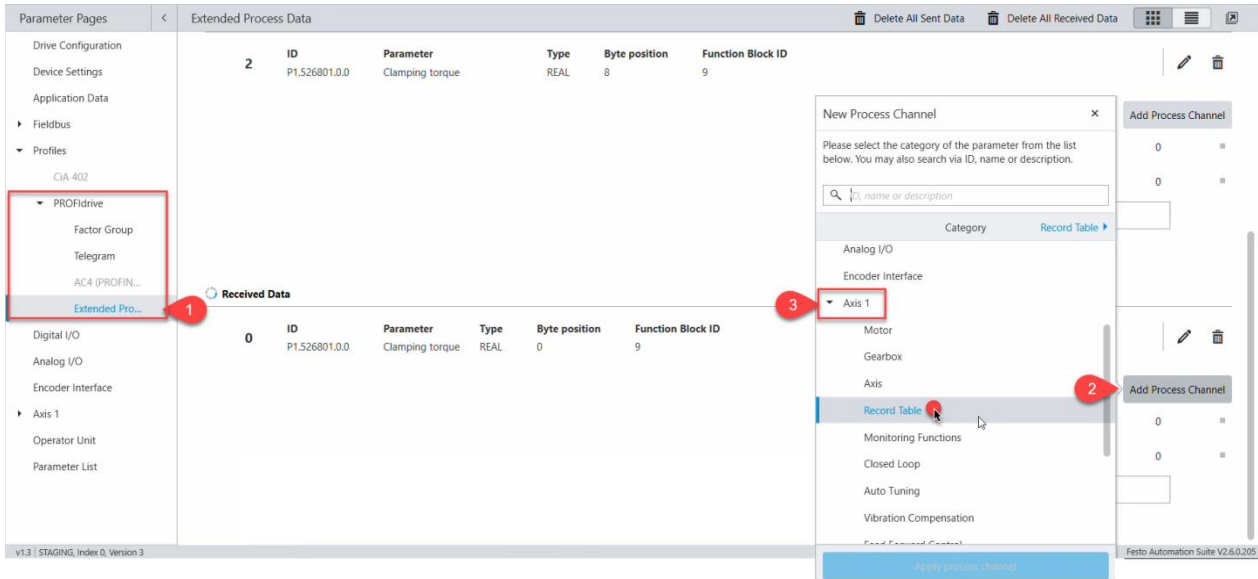
Extended Process Data



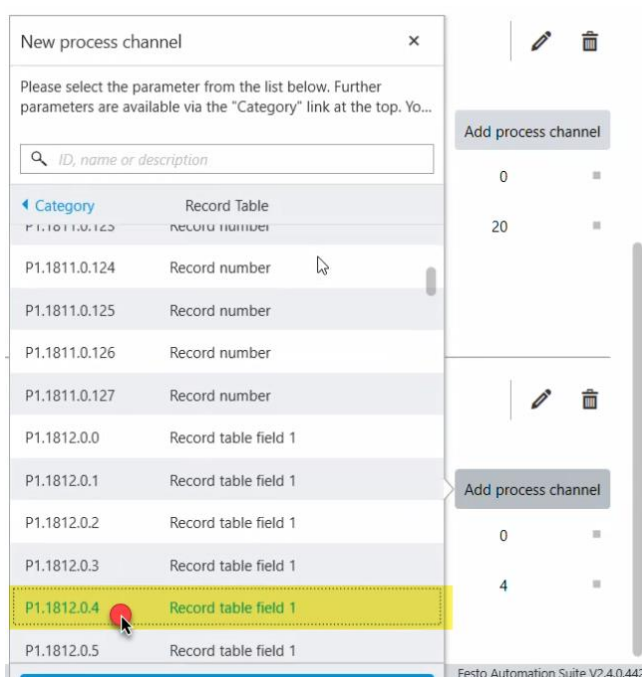
9.2.2 Changing the Record Table Value from EPD

**Caution**

1. It is not recommended to create Record Table from PLC through EPD, this may cause error in connection between CMMT and PLC.
2. When changing the value of Record Table from PLC through EPD, make sure to latch the default value of respective parameter in PLC, else at every power cycle of PLC the value of parameter will be zero. (Use Retain Memory for writing the parameter value).
3. Not all parameters of Record Table Field can be changed through EPD, and also to read data in PLC through EPD. Max of 7 Record Table Field can be called in EPD.

Step – 1

1. In **Parameter pages** expand **PROFIdrive** under Profile tree and click on the **Extended Process. Data**.
2. Under the **Received Data**, Click **Add process channel**
3. Under the **Axis 1**, Click on **Record Table**

Step – 2

- Select the right parameter to change the Record Table value. (Refer below image)

Extended Process Data

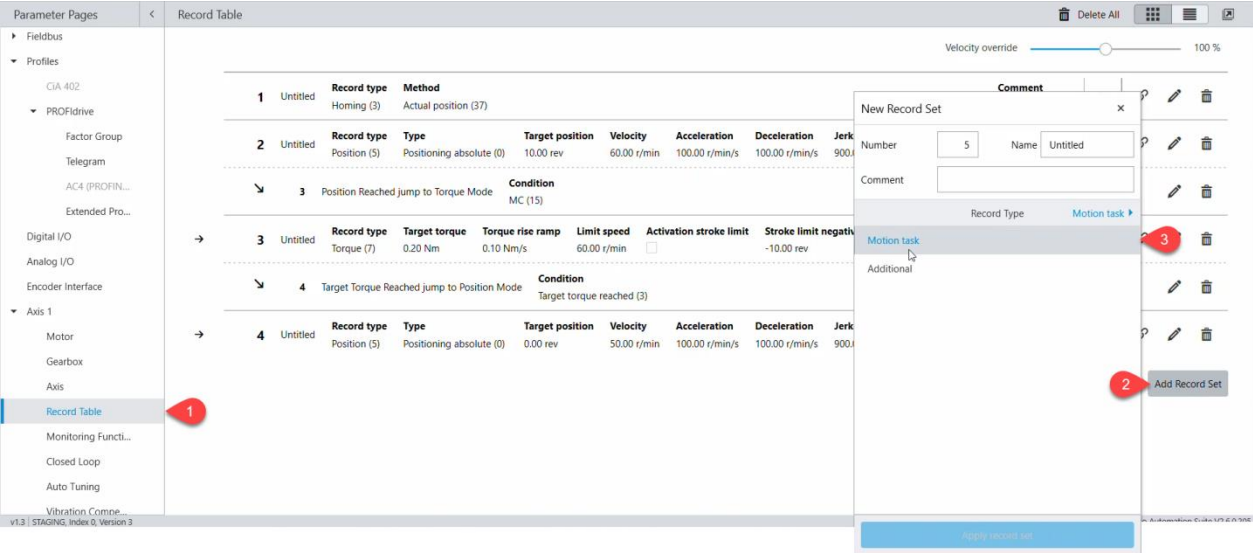
Record Table									
1	Homing	Record type Homing (3)	Method Current position (37)						
2	Position_Mode_1	Record type Position (5)	Type Positioning relative to current actual position (1)	Target position 1000.00 mm	Velocity 1.00 m/s	Acceleration 5.00 m/s ²	Deceleration 5.00 m/s ²	Jerk 150.00 m/s ³	End ve 0.00 m/s
3	Position_Mode_2	Record type Position (5)	Type Positioning relative to current actual position (1)	Target position 2000.00 mm	Velocity 10.00 m/s	Acceleration 5.00 m/s ²	Deceleration 5.00 m/s ²	Jerk 150.00 m/s ³	End v 0.00 n
4	Torque_1	Record type Torque (7)	Target torque 2.00 Nm	Torque rise ramp 1.00 Nm/s	Limit speed 0.40 m/s	Activation stroke limit ☒	Stroke limit negative -2500.00 mm	Stroke limit positive 2500.00 mm	
5	Untitled	Record type Torque (7)	Target torque 4.20 Nm	Torque rise ramp 1.00 Nm/s	Limit speed 0.40 m/s	Activation stroke limit ☒	Stroke limit negative -50.00 mm	Stroke limit positive 50.00 mm	

- User needs to note which record table value needs to change from PLC through EPD.
- The last digit of parameter refers to the Records Table in the list. (Note – Last digit starts from 1 hence user needs to count the record table from 1 onwards).

10 Record table

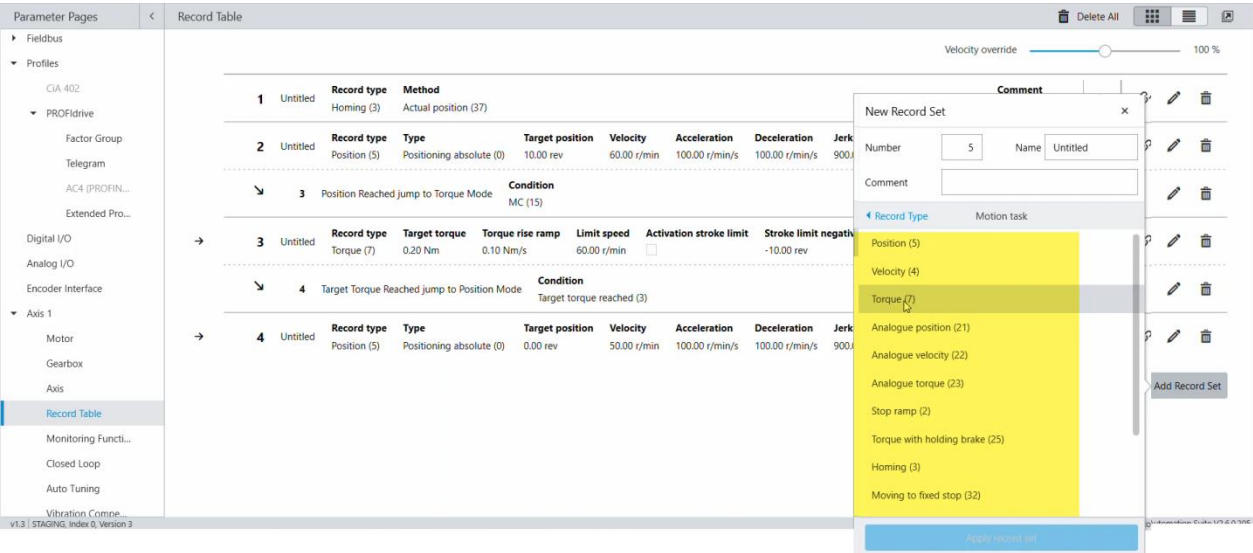
10.1 Set the Task/Sequence in Record Table

Step – 1



- 1. In **Parameter pages** => **Axis 1** => **Record Table**
- 2. Click on the **Add Record Set**
- 3. Select **Motion task**

Step – 2



- Select as per User requirement.

Step – 3

- Set the parameter value as per user requirement.
- Enable Activation Stroke limit function if required, set the stroke limit values.

Step – 4

Store on Device



Reinitialize

- Click the **“Store on device”** button to store all the parameters on drive memory.
- Click the **“Reinitialize”** the button to initialize the drive.

**Note**

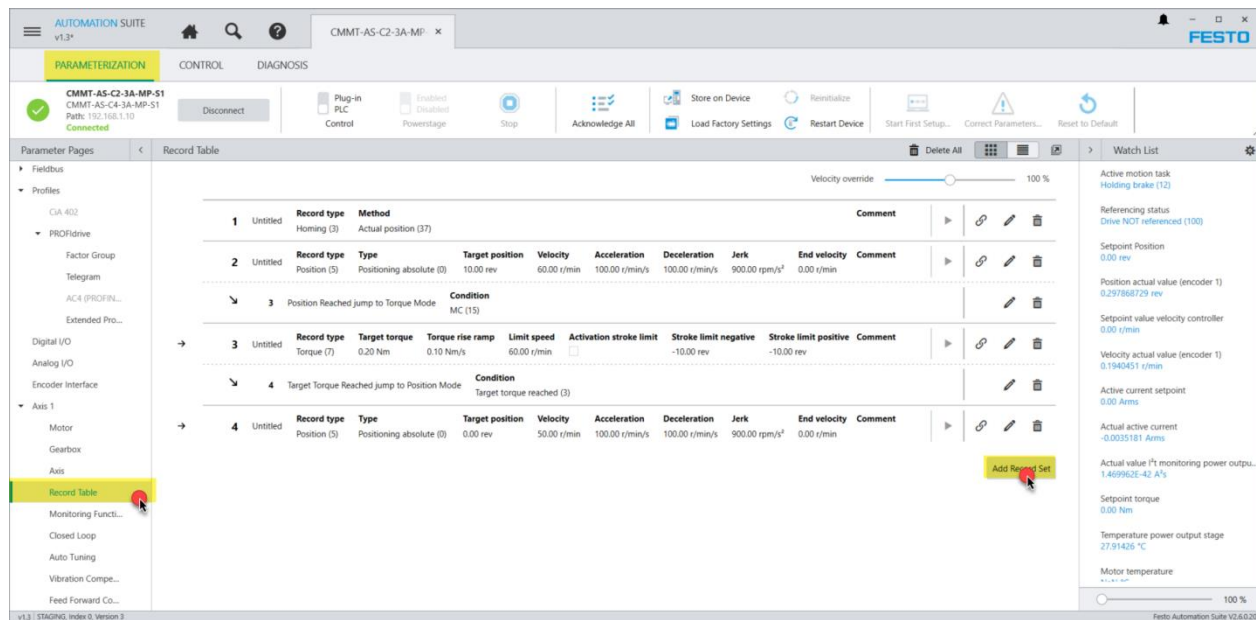
- The triangle warning symbol on both buttons indicates action required.
- All the modified parameters will automatically erase from drive during power restart If modified parameters are not stored in the drive.
- Its mandatory to reinitialize the drive for some parameter modification. If drive is not reinitialized the drive performance remains same as like previous configuration.

10.2 Testing the Task/Sequence of control through FAS

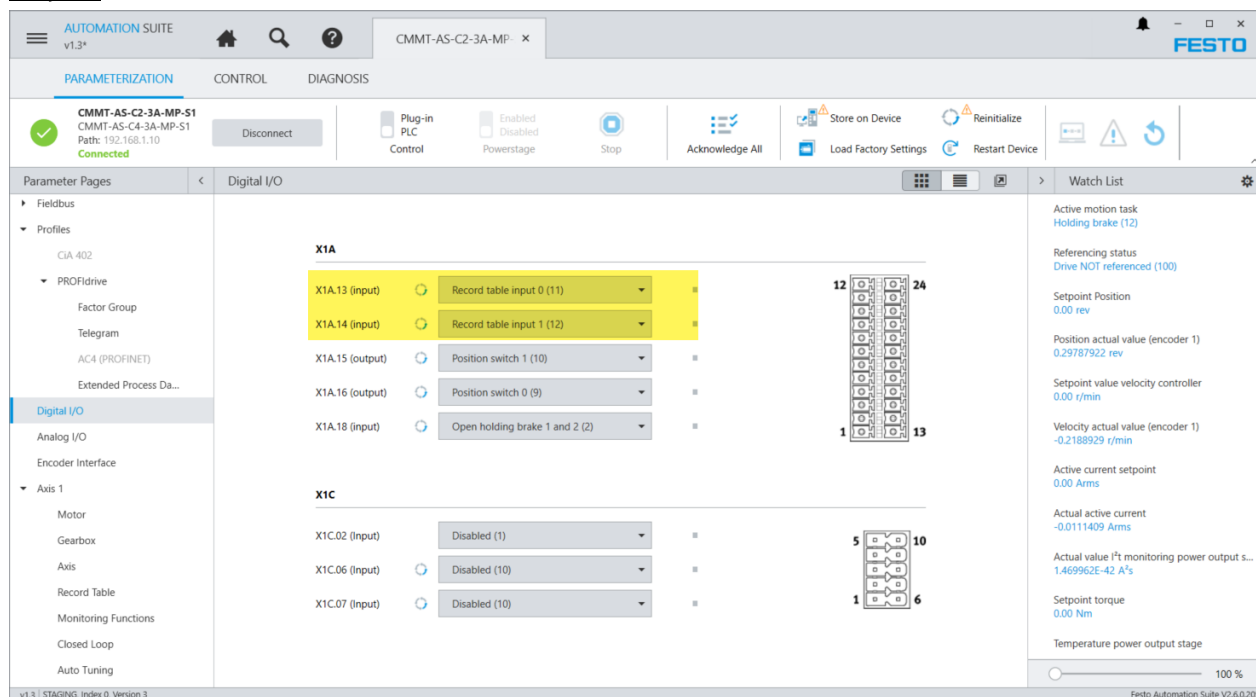
Run the CMMT-AS without PLC or Field Bus to ensure the Take/Sequence created in Record table.

Method – 1: Using I/O's running the Task/Sequence

Step – 1: Create the record list as per requirement, Example: shown in below image.



Step – 2



- assign digital input XA.13 to “record table input-0”
- assign digital input XA.14 to “record table input-1”

Step – 3

Parameter List					
ID	Name	Value	Unit		
▼ /Axis1/Controller enable group[0] (9) ↻					
P1.10231.0.0	Device status IO	Ready (1)			■
P1.10232.0.0	Controller enable selection	I/O (1)			■
P1.10234.0.0	Controller enable operating mode	Start record table (15)			■
P1.10235.0.0	Target velocity for controller enable (velocity operation)	0.00	r/min		■
P1.10236.0.0	Target torque for controller enable (torque operation)	0.00	Nm		■
P1.10237.0.0	Maximum velocity for controller enable (torque operation)	20.00	r/min		■
P1.10238.0.0	Index for controller enable	1			■
P1.112819.0.0	Error active	☐ Active			■
P1.11280018.0.0	Torque increase for controller enable	0.00	Nm/s		■
▼ /Axis1/PROFIdrive MELDW group[0] (12) ↻					
P1.11249110.0.0	MELDW.11 Controller enable	☒ Active			■
▼ /Axis1/PROFIdrive device identification group[0] (18) ↻					
P1.24126.0.0	Assignment controller enable	4			■
P1.24126.0.1	Assignment controller enable	11			■

Parameters can be found within the “Controller enable group” for axis-1

- P1.10232.0.0 – Controller enable selection (set to “ENABLE by IO” → CTRL-EN / X1A.3)
- P1.10234.0.0 – Controller enable operating mode (set to “Start record table”)
- P1.10238.0.0 – Index for controller enable (The record table number can be set to start from any desired number chosen by the user, keeping in mind that record table numbers conventionally start from 1).

Step – 4

Store on Device



Reinitialize

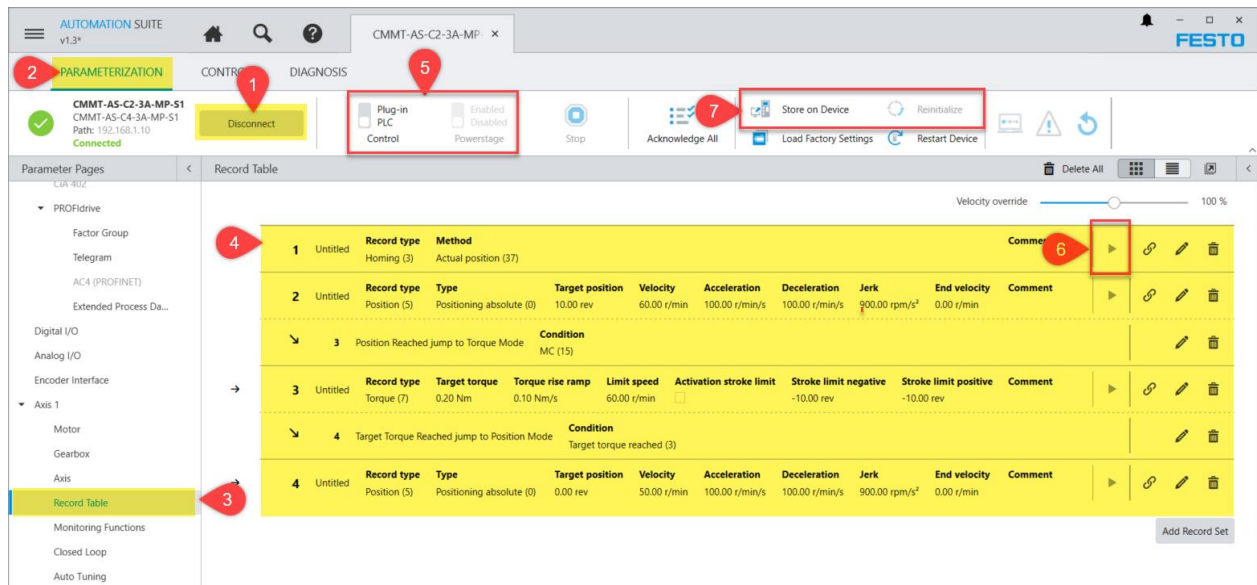
- Click the “**Store on device**” button to store all the parameters on drive memory.
- Click the “**Reinitialize**” the button to initialize the drive.

**Note**

- The triangle warning symbol on both buttons indicates action required.
- All the modified parameters will automatically erase from drive during power restart If modified parameters are not stored in the drive.
- Its mandatory to reinitialize the drive for some parameter modification. If drive is not reinitialized the drive performance remains same as like previous configuration.

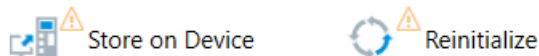
Method – 2: In FAS, Bypass the PLC through enabling Plug-in PLC Control

Step – 1



1. Click on **Connect** to establish the connection with CMMT-AS Drive.
2. Once the connection establishes, Go to **Parameterization**.
3. Expand Axis 1 => Click on Record Table
4. Create the record list, User defined.
5. Toggle **Control** and **Powerstage** button.
6. Click on the Play Button of the selected record list with user want to execute.
After checking the Task/Sequence, Store the Record Table in the Drive.

Step – 2



- Click the **“Store on device”** button to store all the parameters on drive memory.
- Click the **“Reinitialize”** the button to initialize the drive.

11 Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.



Note

- Please refer to either the "SampleCode FBD_Position&TorqueMode&EPD" or "SampleCode LD_Position&TorqueMode&EPD" program found in the "03.samplecode" folder.

For instance let's take an sequence as described below.

control sequence

- Move to a specific position from initial/start position by position control.
- Press with torque control at the position.
 - Stroke limit function applied in torque control
(For example, if the stroke limit is set to 100mm, the torque control stops when it reaches 100mm during torque control)
- Return to initial position by position control after press.

Above Sequence can be executed in two ways,

- Writing the logic in PLC using PTP_Drives_Festo_MB Modes of Operation
- Using Record Table, instead writing the entire logic in PLC and then use PTP_Drives_Festo_MB FB to execute Record Table.

11.1 Method 1 - Using PTP_Drives_Festo_MB Modes of Operation

11.1.1 Set EPD and Dynamics Parameters in FAS

Step – 1

- To execute the sequence, user has to add the EPD, given below as in FAS.

Extended process data (EtherNet/IP)
☒ Active

Extended process data active
☒ Active

Sent Data

	ID	Parameter	Type	Byte position	
0	P1.2220.0.0	Setpoint value torque	FLOAT32	0	
1	P1.150.0.0	Actual torque value motor shaft	FLOAT32	4	
2	P1.526801.0.0	Clamping torque	FLOAT32	8	

Add process channel

Number of bytes Tx20

Number of bytes Tx (Offline)12

Received Data

	ID	Parameter	Type	Byte position	
0	P1.526801.0.0	Clamping torque	FLOAT32	0	

Add process channel

Number of bytes Rx4

Number of bytes Rx (Offline)4

Extended Process Data :- Sent Data from FAS (Read Data)

ID	Parameter	Type	Byte Position	Word
P1.2220.0.0	Setpoint value torque	FLOAT32	0	2
P1.150.0.0	Actual torque value motor shaft	FLOAT32	4	2

Extended Process Data :- Received Data to FAS from PLC (Receive Data)

ID	Parameter	Type	Byte Position	Word
P1.526821.0.0	Clamping Torque	FLOAT32	0	2



Note

In this method –

- Axis won't stop even after achieving the Clamp torque. Axis continues to move till the set Position is reached.
- To stop axis when clamp torque reached is True, following parameters to be set. "CancelTraversing" and "IntermediateStop" input tags must be False. Dynamics – such as Acceleration & Deceleration must be maximum to stop the axis.

PARAMETERIZATION CONTROL DIAGNOSIS

CMMT-AS-C2-3A-MP-S1
CMMT-AS-C4-3A-MP-S1
Path: 192.168.1.10
Connected

Disconnect Plug-in PLC Control Enabled Disabled Powerstage Stop Acknowledge All Store on Device Load Factory Settings Reinitialize Restart Device

Parameter Pages < Telegram Telegram selection Telegram (111) Actual application class Application class 3 (3)

▼ PROFIdrive
Factor Group
Telegram

AC4 (PROFINET)
Extended Process Da...

Digital I/O
Analog I/O
Encoder Interface
▼ Axis 1
Motor
Gearbox
Axis
Record Table
Monitoring Functions
Closed Loop
Auto Tuning
Vibration Compensation
Feed Forward Control
Modulo Mode
► Position Trigger

Dynamic Values

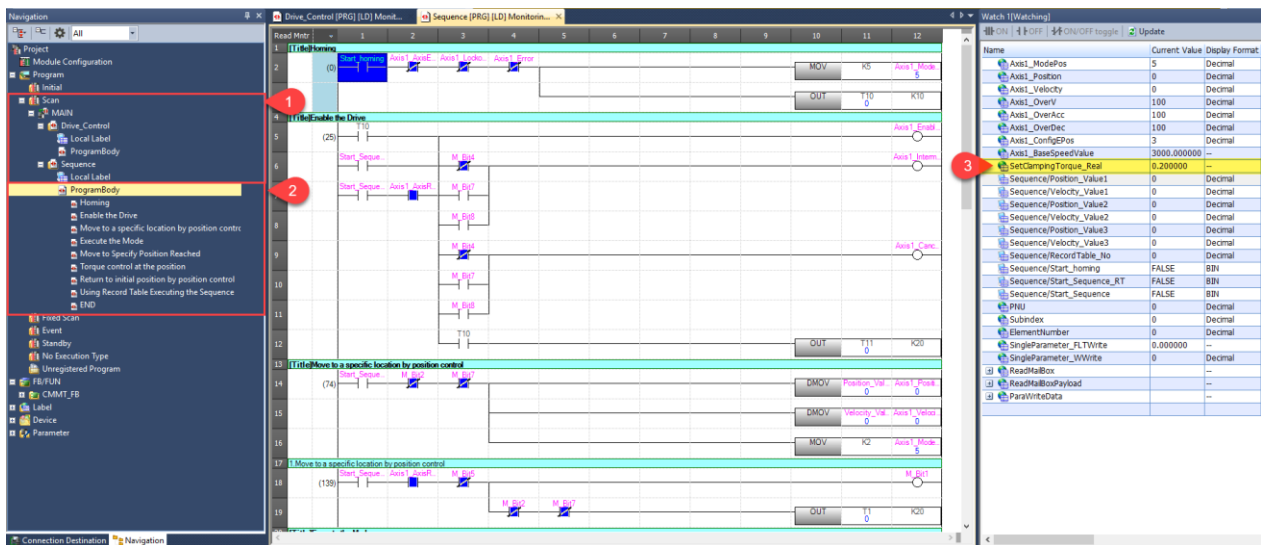
Coast stop

Parameter	Value	Unit
Acceleration	3000.00	r/min/s
Deceleration	3000.00	r/min/s
Jerk	300000.00	rpm/s ²
Deceleration (system stop AC1/AC3)	600.00	r/min/s
Jerk (system stop AC1/AC3)	600000.00	rpm/s ²
Deceleration (stop ramp)	12000.00	r/min/s
Jerk (stop ramp)	600000.00	rpm/s ²
Base value acceleration	600.00	r/min/s
Base value deceleration	600.00	r/min/s
Base value velocity	3000.00	r/min
Base value velocity (PLC, motorside)	3000.00	r/min

11.1.2 Execute the Logic in PLC using EPD

Step – 1:

Few customer uses Ladder language predominantly. (Sample Code program is written in ladder language)



Refer Sample Code program for viewing the logic.

1. Under Scan Program, [Program block] => [Customer] defined by name “Sequence”.
2. List of Device Label to modify values in Watch 1.
3. Follow the instruction given in [chapter 9](#) to add EPD

11.1.3 Procedure to control in Torque Mode using PTP_Drive_Festo_MB function block (Refer Chapter 6.4.9)

- Using a positioning mode absolute or relative (ModePos = 1 or 2 (positioning absolute/relative))



Note

- Sample Code is for your reference. It's not recommended to follow exact sequence.
- User can take this as reference to write own logic.
- Verify the Position and Velocity value based on the Liner/Rotative Axis. Correct Position and Velocity to be entered in user defined logic or even while using sample code.

Step – 1

At first scan of PLC, the PTP_Drives_Festo_MB FB shows an “Error” and “Lockout”.

- To clear the Error and Lockout follow the steps below.

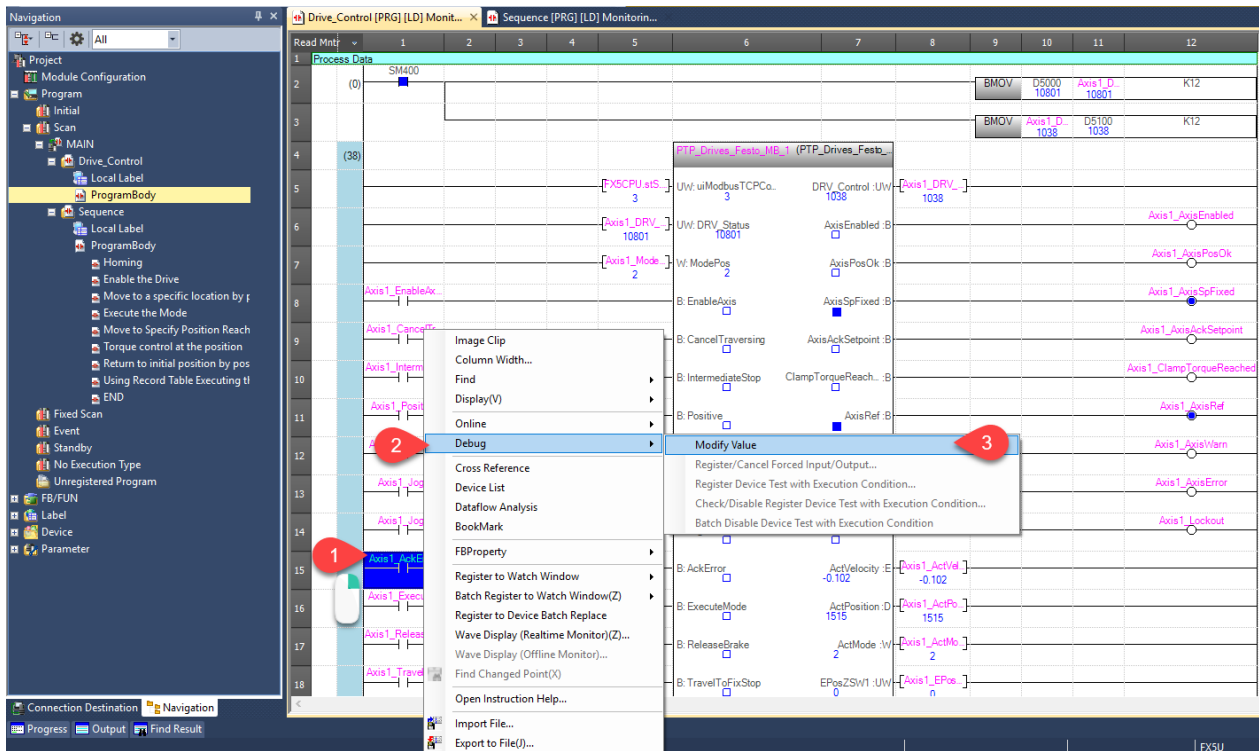
Enter the value in

- Axis1_ModePos := 5 [Home to Current Position [ModePos – 5]]
- OverV := 100
- OverAcc := 100
- OverDec := 100
- Axis1_ConfigEPoS := 3 or 15. ([Refer ConfigEPoS Bit Description](#))
- Axis1_BaseSpeedValue := 3000.0

After entering the value PTP_Drives_Festo_MB FB does not shows an “Error” and “Lockout”.

In case if Error Bit is True, see the below given image to enable “Axis1_AckError” bit.

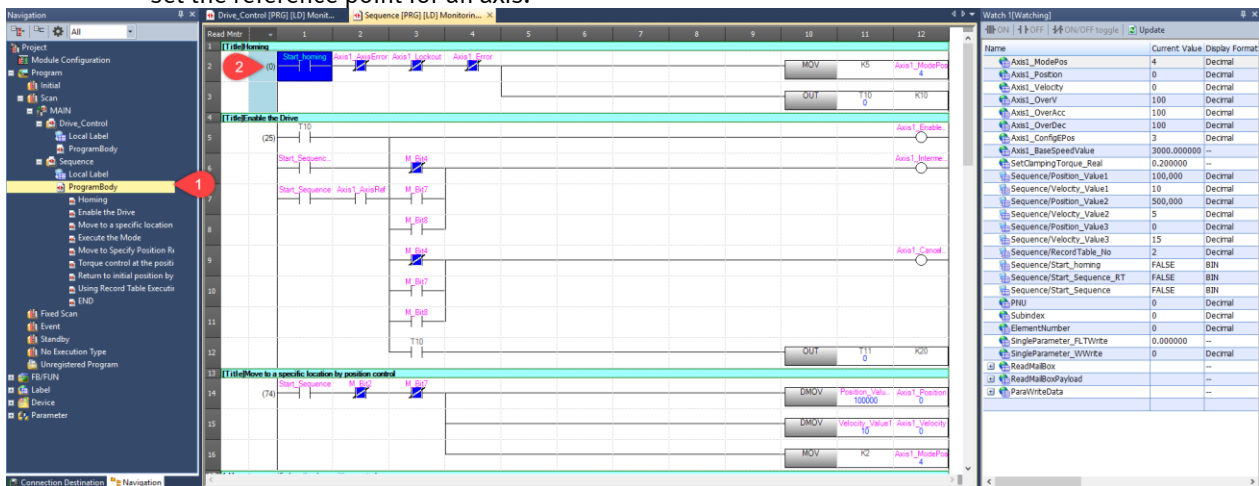
Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.



1. Right-Click [Axis1_AckError] => [Debug] => [Modify Value]
A new window “**Modify Value**” will be opened.
Or
Press “**Shift+Enter**” button to modify value.
2. Go to Debug to modify value.
3. Click “**Modify value**” button to change the Input state, or to change the Input Value.

Step – 2:

- Check “**AxisRef**” status tag is True. If not true user could also use other modes like mode 4 or 5 to set the reference point for an axis.



1. Expand [Scan Program] => [Main] => [Sequence] => [ProgramBody]
A program will be opened in new window.
2. Select the “**Start_Homing**” input tag to “**ON**” it.
 - Either use short cut “**Shift+Enter**” key will toggle the state of Start_Homing input tag or
Right-Click [Axis1_AckError] => [Debug] => [Modify Value]
 - Click “**ON**”/“**OFF**” to modify value using Modify Value pop-up window.
 - After turning ON the “**Start_Homing**” tag the axis will be referenced. Check “**AxisRef**” status tag will be True. (See the below image).

Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.

- Toggle **“Start_Homing”** tag to turn it OFF

The screenshot displays the Mitsubishi GX Developer interface. The main window shows the ladder logic for the PTP_Drives_Festo_MB_1 function block. The function block is configured with various inputs and outputs, including Axis1_Pos, Axis1_Velo, Axis1_Acc, Axis1_Dec, Axis1_EPos, Axis1_BaseSpeedValue, SetClampingTorque_Real, Sequence/Position_Value1, Sequence/Position_Value2, Sequence/Position_Value3, Sequence/Velocity_Value1, Sequence/Velocity_Value2, Sequence/Velocity_Value3, Sequence/RecordTable_No, Sequence/Start_Homing, Sequence/Start_Sequence_RT, Sequence/Start_Sequence, PNU, Subindex, ElementNumber, SingleParameter_FLTWrite, SingleParameter_WWrite, and ParaWriteData. The right-hand pane shows the Watch 1(Watching) table, which lists the current values and display formats for these variables.

Name	Current Value	Display Format
Axis1_ModePos	5	Decimal
Axis1_Position	0	Decimal
Axis1_Velocity	0	Decimal
Axis1_OverV	100	Decimal
Axis1_OverAcc	100	Decimal
Axis1_OverDec	100	Decimal
Axis1_ConfigEPos	3	Decimal
Axis1_BaseSpeedValue	3000.000000	---
SetClampingTorque_Real	0.200000	---
Sequence/Position_Value1	0	Decimal
Sequence/Position_Value2	0	Decimal
Sequence/Position_Value3	0	Decimal
Sequence/Velocity_Value1	0	Decimal
Sequence/Velocity_Value2	0	Decimal
Sequence/Velocity_Value3	0	Decimal
Sequence/RecordTable_No	0	Decimal
Sequence/Start_Homing	TRUE	BIN
Sequence/Start_Sequence_RT	FALSE	BIN
Sequence/Start_Sequence	FALSE	BIN
PNU	0	Decimal
Subindex	0	Decimal
ElementNumber	0	Decimal
SingleParameter_FLTWrite	0.000000	---
SingleParameter_WWrite	0	Decimal
ParaWriteData	---	---

Step – 3 :

- Set the **“Clamping torque”** as per user defined, this is taken from the EPD.



Note

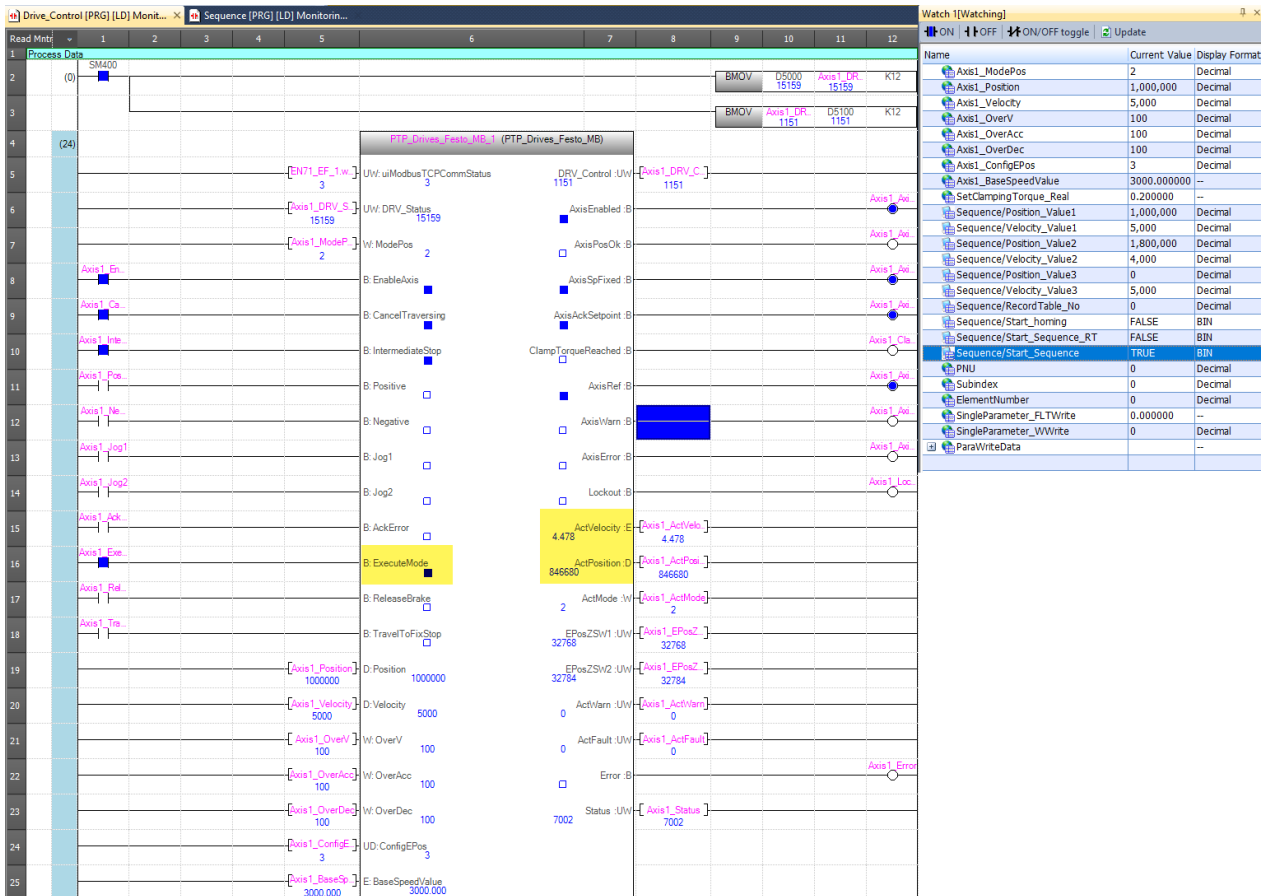
- If Clamping Torque is not set then, Travel to fixed stop won't work. As soon sequence is executed the **“ClampTorqueReached”** Bit will be True.
- Axis position won't change and Drive will go to Error mode.

The screenshot displays the Mitsubishi GX Developer interface. The main window shows the ladder logic for the Start Sequence input tag. The function block is configured with various inputs and outputs, including Axis1_Pos, Axis1_Velo, Axis1_Acc, Axis1_Dec, Axis1_EPos, Axis1_BaseSpeedValue, SetClampingTorque_Real, Sequence/Position_Value1, Sequence/Position_Value2, Sequence/Position_Value3, Sequence/Velocity_Value1, Sequence/Velocity_Value2, Sequence/Velocity_Value3, Sequence/RecordTable_No, Sequence/Start_Homing, Sequence/Start_Sequence_RT, Sequence/Start_Sequence, PNU, Subindex, ElementNumber, SingleParameter_FLTWrite, SingleParameter_WWrite, and ParaWriteData. The right-hand pane shows the Watch 1(Watching) table, which lists the current values and display formats for these variables.

Name	Current Value	Display Format
Axis1_ModePos	2	Decimal
Axis1_Position	0	Decimal
Axis1_Velocity	5	Decimal
Axis1_OverV	100	Decimal
Axis1_OverAcc	100	Decimal
Axis1_OverDec	100	Decimal
Axis1_ConfigEPos	3	Decimal
Axis1_BaseSpeedValue	3000.000000	---
SetClampingTorque_Real	0.200000	---
Sequence/Position_Value1	1,000,000	Decimal
Sequence/Position_Value2	10	Decimal
Sequence/Position_Value3	5,000,000	Decimal
Sequence/Velocity_Value1	5	Decimal
Sequence/Velocity_Value2	0	Decimal
Sequence/Velocity_Value3	5	Decimal
Sequence/RecordTable_No	2	Decimal
Sequence/Start_Homing	FALSE	BIN
Sequence/Start_Sequence_RT	FALSE	BIN
Sequence/Start_Sequence	FALSE	BIN
PNU	0	Decimal
Subindex	0	Decimal
ElementNumber	0	Decimal
SingleParameter_FLTWrite	0.000000	---
SingleParameter_WWrite	0	Decimal
ParaWriteData	---	---

1. Expand [Scan Program] => [Main] => [Sequence] => [ProgramBody]
A program will be opened in new window.
2. Select the **“Start_Sequence”** input tag to **“ON”** it.
 - Either use short cut **“Shift+Enter”** key will toggle the state of Start_sequence input tag or

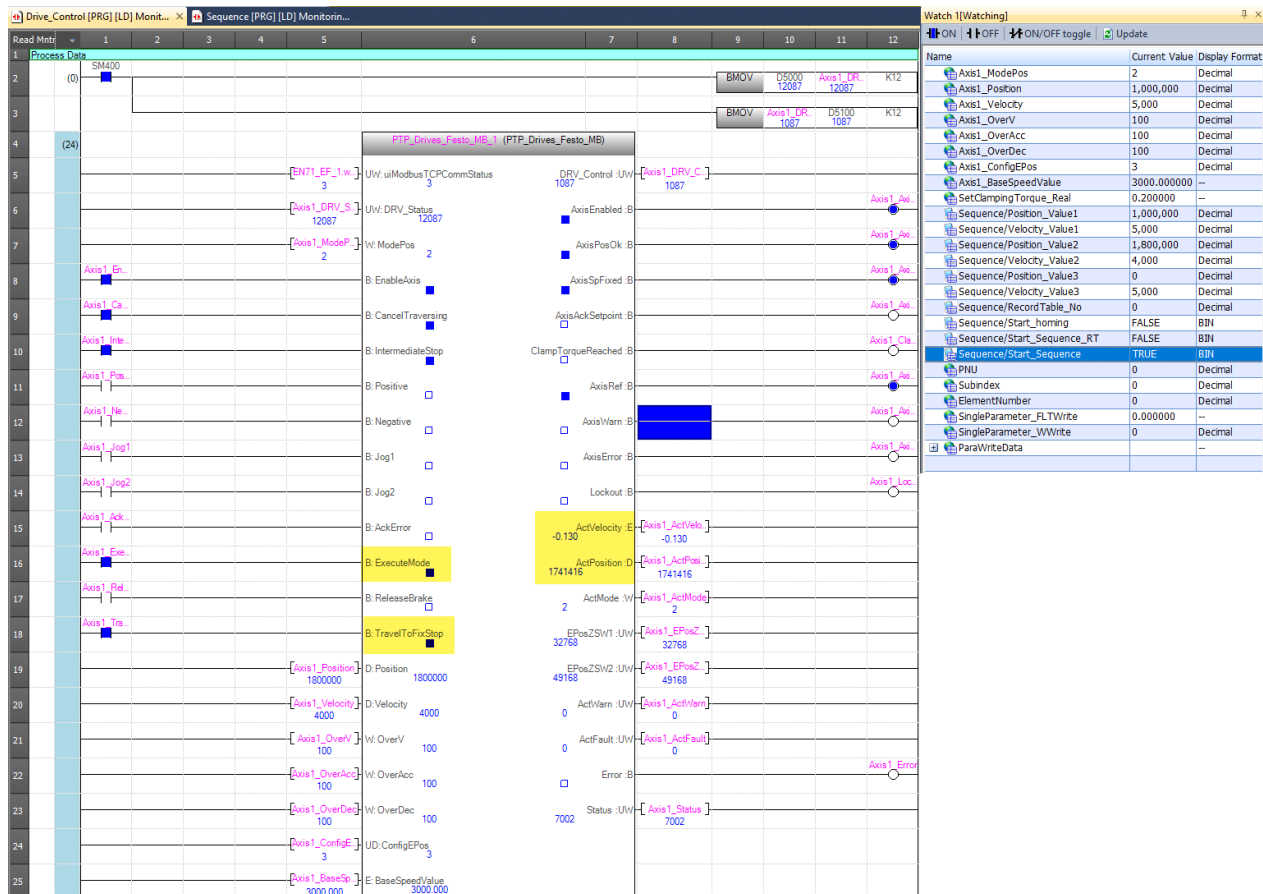
- Right-Click  [Start_Sequence] =>[Debug] =>[Modify Value]
 – Click “ON”/”OFF” to modify value using Modify Value pop-up window.



Below sequence will be executed after “Start_Sequence” button to “ON”

1. Move to a specific position from initial/start position by position control. Following configuration and command is performed.
 - Set the “**ModePos**” value to 2 & check the output tag “**ActMode**” has the same value as ModePos.
 - Enable the axis by toggling “**EnableAxis**” input tag.
 - “**CancelTraversing**” and “**IntermediateStop**” input tags must be **TRUE**.
 - Enter the Position command value in “**Position**” tag & Velocity command value in “**Velocity**” tag. Give the absolute position value. The drive will decide forward & reverse direction based on its current position.
 - Toggle the “**ExecuteMode**” command tag (Rising Edge) to start a motion task using the configured parameters (Every new position task would require a toggle of the “**ExecuteMode**” command tag).
 - User will receive execute command accepted handshake signal from “**AxisAckSetpoint**” status tag.
 - If the task is successfully completed, it is acknowledge by rising edge of “**AxisPosOk**” status variable. Also if task is unsuccessful “**AxisError**” output tag will be **TRUE**.
 - “**AxisSpFixed**” Status tag indicates the axis mobility status . If the axis is standby the “**Axis-SpFixed**” status tag will be high.

Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.



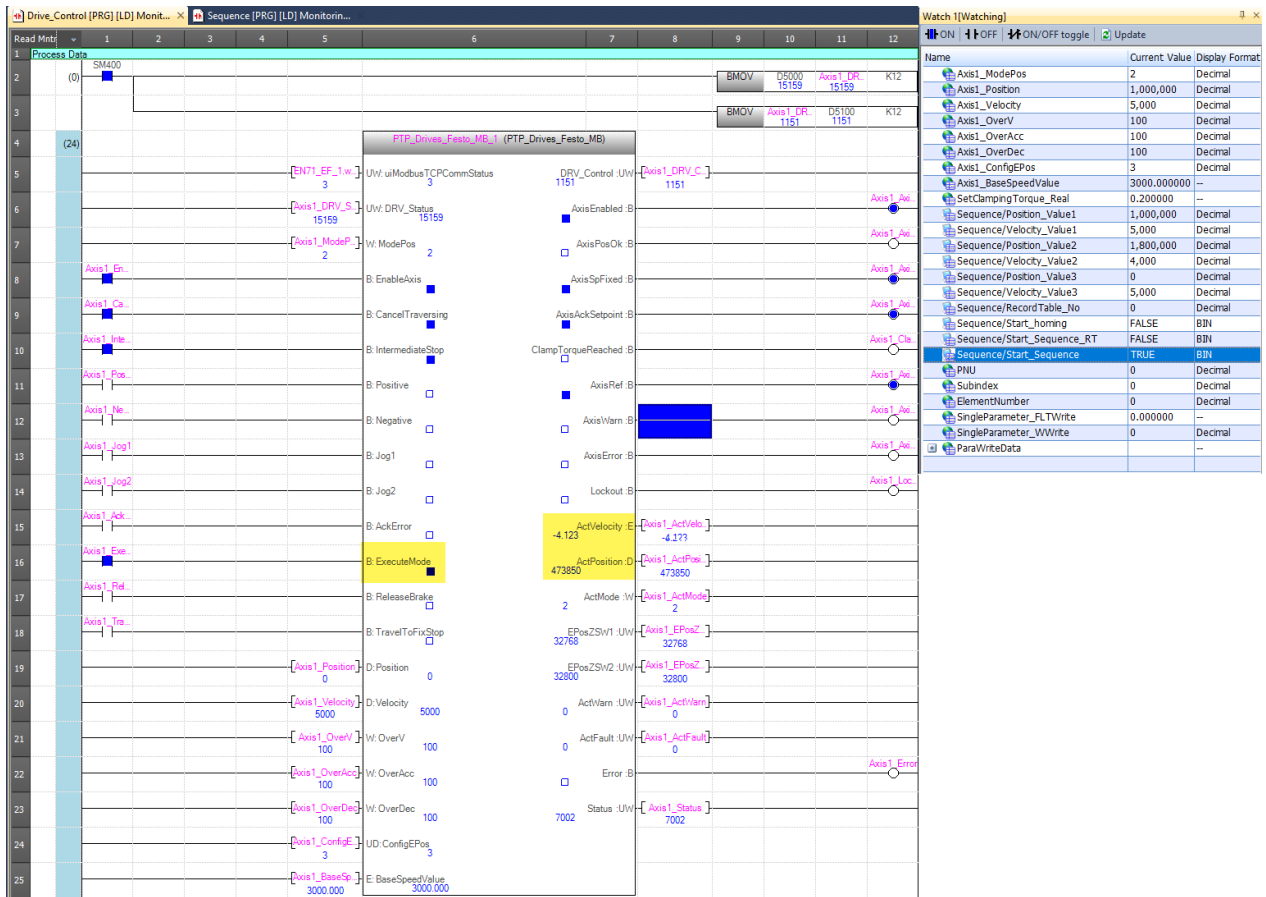
2. Move with torque control at the position.

- Stroke limit function applied in torque control - (Optional as per user requirement)
(For example, if the stroke limit is set to 10mm, the torque control stops when it reaches 10mm during torque control)
- Travel to fixed Stop using Absolute Positioning Mode

Following configuration and command is performed.

- Enable the axis by toggling **“TravelToFixStop”** input tag.
- Set the **“ModePos”** value to 2 & check the output tag **“ActMode”** has the same value as ModePos.
- Enable the axis by toggling **“EnableAxis”** input tag.
- “CancelTraversing”** and **“IntermediateStop”** input tags must be **TRUE**.
- Enter the Position command value in **“Position”** tag & Velocity command value in **“Velocity”** tag. Give the absolute position value. The drive will decide forward & reverse direction based on its current position.
- Toggle the **“ExecuteMode”** command tag (Rising Edge) to start a motion task using the configured parameters (Every new position task would require a toggle of the **“ExecuteMode”** command tag).
- User will receive execute command accepted handshake signal from **“AxisAckSetpoint”** status tag.
- If the task is successfully completed, it is acknowledge by rising edge of **“AxisPosOk”** status tag. Also if task is unsuccessful **“AxisError”** output tag will be **TRUE**.
- “AxisSpFixed”** Status tag indicates the axis mobility status . If the axis is standby the **“Axis-SpFixed”** status tag will be high.
- User will receive Clamping torque reached signal from **“ClampTorqueReached”** status tag.

Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.



3. Return to initial position by position control after press.
Following configuration and command is performed.

- Set the **“ModePos”** value to 2 & check the output tag **“ActMode”** has the same value as ModePos.
- Enable the axis by toggling **“EnableAxis”** input tag.
- **“CancelTraversing”** and **“IntermediateStop”** input tags must be **TRUE**.
- Enter the Position command value in **“Position”** tag & Velocity command value in **“Velocity”** tag. Give the absolute position value. The drive will decide forward & reverse direction based on its current position.
- Toggle the **“ExecuteMode”** command tag (Rising Edge) to start a motion task using the configured parameters (Every new position task would require a toggle of the **“ExecuteMode”** command tag).
- User will receive execute command accepted handshake signal from **“AxisAckSetpoint”** status tag.
- If the task is successfully completed, it is acknowledge by rising edge of **“AxisPosOk”** status variable. Also if task is unsuccessful **“AxisError”** output tag will be **TRUE**.
- **“AxisSpFixed”** Status tag indicates the axis mobility status . If the axis is standby the **“Axis-SpFixed”** status tag will be high.

11.2 Method 2 - Using FAS Record Table and PtP_Drives_Festo_MB Modes of Operation

11.2.1 Set the Task/Sequence in Record Table

Step – 1:

Using Record Table, instead writing the entire logic in PLC and then use PtP_Drives_Festo_MB FB to execute Record Table.

- Using a Traversing/Record Selection Mode [ModePos – 6]



Note

- Sample Code is for your reference. It's not recommended to follow exact sequence.
- User can take this as reference to write own logic.
- Verify the Position and Velocity value based on the Liner/Rotative Axis. Correct Position and Velocity to be entered in user defined FAS Record Table

Before downloading the program in PLC, first define sequence in FAS under Record Table

Step – 2

Refer [chapter 10.1](#) for setting the Record Table in FAS.

Record Table										
Delete All										
1	1	Homing	Record type Homing (3)	Method Current position (37)						
2	2	Move to Position	Record type Position (5)	Type Positioning absolute (0)	Target position 100.00 mm	Velocity 0.02 m/s	Acceleration 0.05 m/s ²	Deceleration 0.05 m/s ²	Jerk 1.50 m/s ³	End velocity 0.00 m/s
3	3	Position Reached jump to Torque Mode	Condition MC (15)							
4	3	Torque Mode	Record type Torque (7)	Target torque 0.30 Nm	Torque rise ramp 0.10 Nm/s	Limit speed 0.01 m/s	Activation stroke limit ☐	Stroke limit negative -120.00 mm	Stroke limit positive 120.00 mm	
5	4	Target Torque Reached jump to Position Mode	Condition Target torque reached (3)							
6	4	Initial Position	Record type Position (5)	Type Positioning absolute (0)	Target position 0.00 mm	Velocity 0.05 m/s	Acceleration 5.00 m/s ²	Deceleration 5.00 m/s ²	Jerk 150.00 m/s ³	End velocity 0.00 m/s
Add Record Set										

1. First sequence is to do Homing with current position(37)
2. Second Sequence is to Move to a specific location by Position Absolute (0)
Enter parameter such as
 - Target Position to be reached
 - Velocity at which position to be reached,
 - Acceleration
 - Deceleration
 - Jerk
 - End Velocity
3. Position Reached jump to Torque Mode - (Optional for user to define if required).
4. Third Sequence is to move in Torque Mode.
 - Stroke limit function applied in torque control
(For example, if the stroke limit is set to 120mm, the torque control stops when it reaches 120mm during torque control)
Enter parameter such as
 - Target Torque (User needs to enter the Torque considering Dynamic Value to achieve Target Torque)
 - Target rise ramp should be less than Target Ramp.
 - Limit Speed – Speed at which axis should move to achieve the target torque.
 - Select Activation stroke limit (User requirement - If target torque not reached under the stroke limit then “AxisError” output tag will be TRUE).
 - Stroke limit negative
 - Stroke limit positive
5. Target Torque reached move from Torque Mode to Position Mode - (Optional for user to define if required).

- User should add “Stop Ramp” record table after Target Torque reached, to stop at that sequence, else can directly jump to next sequence.
6. Fourth Sequence is to Move to a specific location by Position Absolute (0)
Enter parameter such as
- Target Position to be reached
 - Velocity at which position to be reached,
 - Acceleration
 - Deceleration
 - Jerk
 - End Velocity



Note

- In this method –
- Axis won't stop even after achieving the Target torque if Activation stroke limit is not enabled.
 - To stop axis when Target torque reached is True, following parameters to be added in record table. “Stop Ramp” task must be added in record table.

11.2.2 Procedure to control in Traversing/Record Table using PTP_Drives_Festo_MB function block (Refer Chapter 6.4.9)

Step – 1

At first scan of PLC, the PTP_Drives_Festo_MB FB shows an “Error” and “Lockout”.

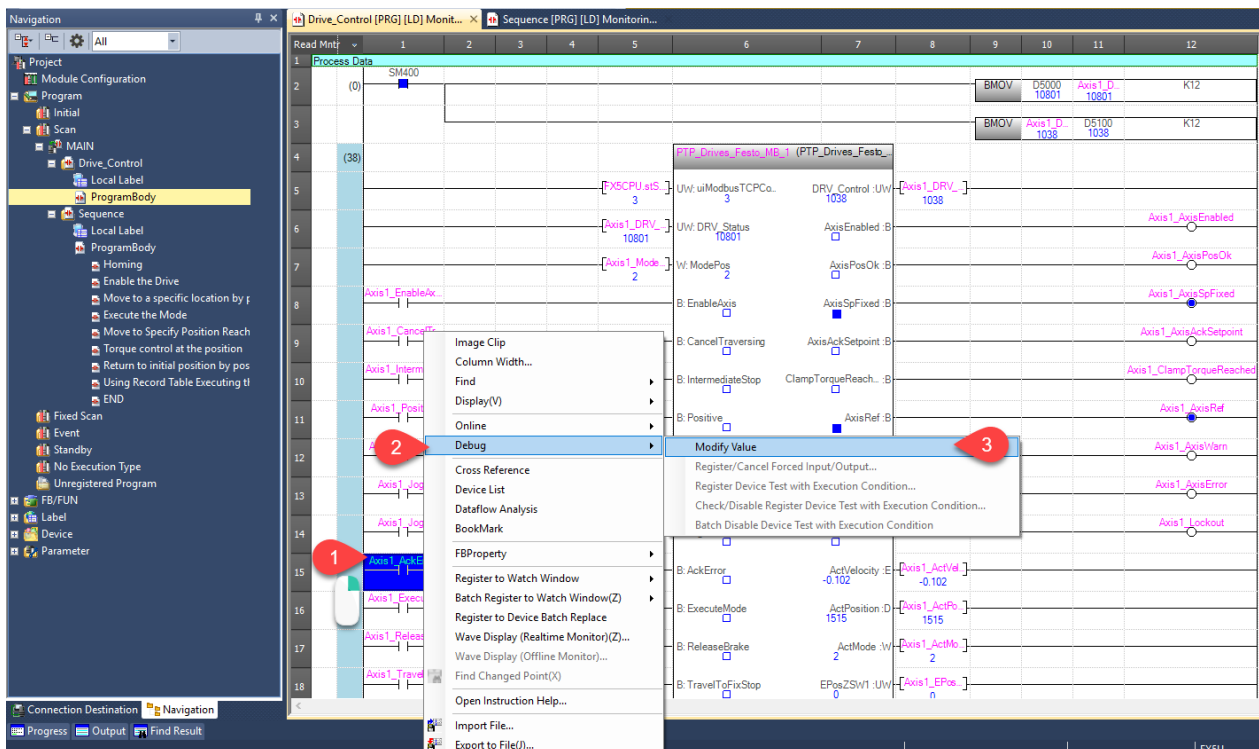
- To clear the Error and Lockout follow the steps below.

Enter the value in

- Axis1_ModePos := 5 [Home to Current Position [ModePos – 5]]
- OverV := 100
- OverAcc := 100
- OverDec := 100
- Axis1_ConfigEPoS := 3 or 15. (Refer ConfigEPoS Bit Description)
- Axis1_BaseSpeedValue := 3000.0

After entering the value PTP_Drives_Festo_MB FB does not shows an “Error” and “Lockout”.

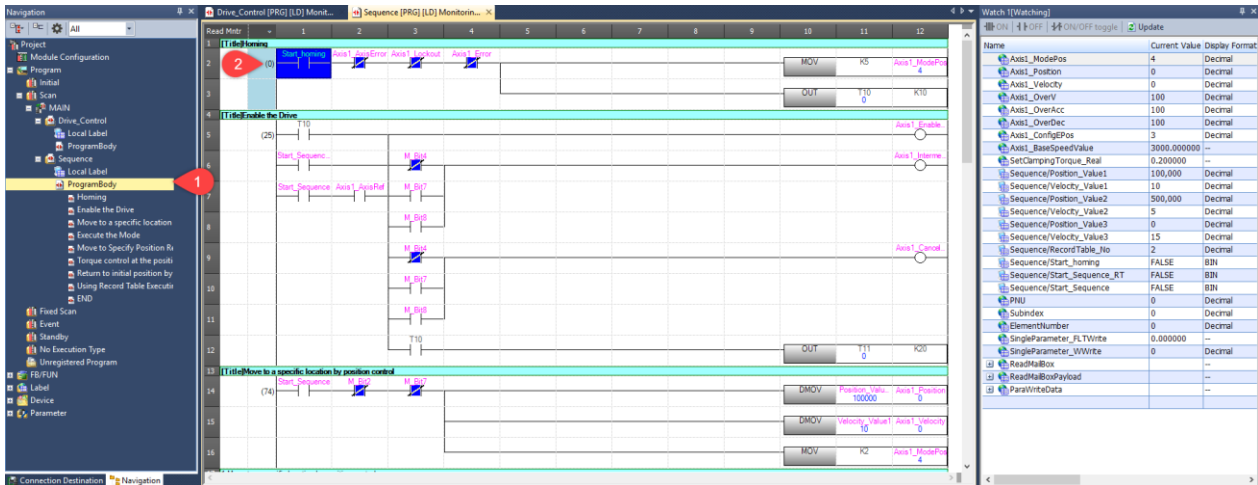
In case if Error Bit is True, see the below given image to enable “Axis1_AckError” bit.



1. Right-Click  [Axis1_AckError] => [Debug] => [Modify Value]
A new window **"Modify Value"** will be opened.
Or
Press **"Shift+Enter"** button to modify value.
2. Go to Debug to modify value.
3. Click **"Modify value"** button to change the Input state, or to change the Input Value.

Step – 2:

- Check **"AxisRef"** status tag is True. If not true user could also use other modes like mode 4 or 5 to set the reference point for an axis.



1. Expand [Scan Program] => [Main] => [Sequence] => [ProgramBody]
A program will be opened in new window.
2. Select the **"Start_Homing"** input tag to **"ON"** it.
 - Either use short cut **"Shift+Enter"** key will toggle the state of Start_Homing input tag or
Right-Click  [Axis1_AckError] => [Debug] => [Modify Value]
Click **"ON"/"OFF"** to modify value using Modify Value pop-up window.
 - After turning ON the **"Start_Homing"** tag the axis will be referenced. Check **"AxisRef"** status tag will be True. (See the below image).
 - Toggle **"Start_Homing"** tag to turn it OFF.

Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.

Name	Current Value	Display Format
Axis1_ModePos	5	Decimal
Axis1_Position	2	Decimal
Axis1_Velocity	0	Decimal
Axis1_OverV	100	Decimal
Axis1_OverAcc	100	Decimal
Axis1_OverDec	100	Decimal
Axis1_ConfigEPos	3	Decimal
Axis1_BaseSpeedValue	3000.000000	--
SetClampingTorque_Real	0.200000	--
Sequence/Position_Value1	100,000	Decimal
Sequence/Velosity_Value1	10	Decimal
Sequence/Position_Value2	500,000	Decimal
Sequence/Velosity_Value2	5	Decimal
Sequence/Position_Value3	0	Decimal
Sequence/Velosity_Value3	15	Decimal
Sequence/RecordTable_No	2	Decimal
Sequence/Start_homing	TRUE	BIN
Sequence/Start_Sequence_RT	FALSE	BIN
Sequence/Start_Sequence	FALSE	BIN
PNU	0	Decimal
Subindex	0	Decimal
ElementNumber	0	Decimal
SingleParameter_FLTWrite	0.000000	--
SingleParameter_WWrite	0	Decimal
ReadMailBox	--	--
ReadMailBoxPayload	--	--
ParaWriteData	--	--

Step – 3:

Below given logic is for executing the task in Record Table.

Name	Current Value	Display Format
Axis1_ModePos	2	Decimal
Axis1_Position	0	Decimal
Axis1_Velocity	5	Decimal
Axis1_OverV	100	Decimal
Axis1_OverAcc	100	Decimal
Axis1_OverDec	100	Decimal
Axis1_ConfigEPos	3	Decimal
Axis1_BaseSpeedValue	3000.000000	--
SetClampingTorque_Real	0.200000	--
Sequence/Position_Value1	1,000,000	Decimal
Sequence/Velosity_Value1	10	Decimal
Sequence/Position_Value2	5,000,000	Decimal
Sequence/Velosity_Value2	5	Decimal
Sequence/Position_Value3	0	Decimal
Sequence/Velosity_Value3	5	Decimal
Sequence/RecordTable_No	2	Decimal
Sequence/Start_homing	FALSE	BIN
Sequence/Start_Sequence_RT	FALSE	BIN
Sequence/Start_Sequence	FALSE	BIN
PNU	0	Decimal
Subindex	0	Decimal
ElementNumber	0	Decimal
SingleParameter_FLTWrite	0.000000	--
SingleParameter_WWrite	0	Decimal
ReadMailBox	--	--
ReadMailBoxPayload	--	--
ParaWriteData	--	--

1. Select the record table by entering the table number (highlighted in yellow color in below pic image) in “Position” command tag. The value must be between 0 to 127. (Record Table Number)

Record Table										Delete All			
1	Homing	Record type	Method										
		Homing (3)	Current position (37)										
2	Move to Position	Record type	Type	Target position	Velocity	Acceleration	Deceleration	Jerk	End velocity				
		Position (5)	Positioning absolute (0)	100.00 mm	0.02 m/s	0.05 m/s ²	0.05 m/s ²	1.50 m/s ³	0.00 m/s				
3	Position Reached jump to Torque Mode	Condition											
		MC (15)											
3	Torque Mode	Record type	Target torque	Torque rise ramp	Limit speed	Activation stroke limit	Stroke limit negative	Stroke limit positive					
		Torque (7)	0.30 Nm	0.10 Nm/s	0.01 m/s	☐	-120.00 mm	120.00 mm					
4	Target Torque Reached jump to Position Mode	Condition											
		Target torque reached (3)											
4	Initial Position	Record type	Type	Target position	Velocity	Acceleration	Deceleration	Jerk	End velocity				
		Position (5)	Positioning absolute (0)	0.00 mm	0.05 m/s	5.00 m/s ²	5.00 m/s ²	150.00 m/s ³	0.00 m/s				
Add Record Set													

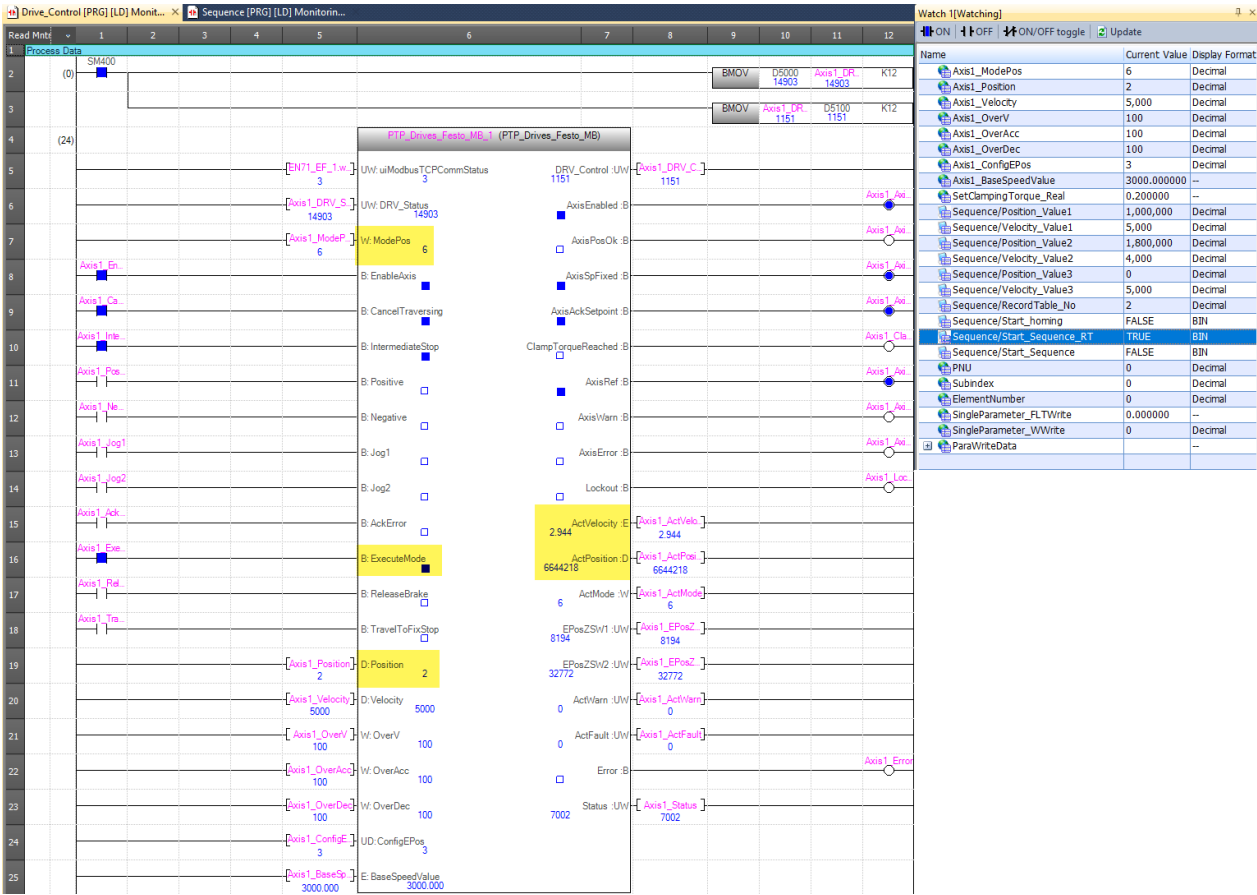
2. Select the **“Start_Sequence_RT”** button to **“ON”** it.
 - a. Either use short cut **“Shift+Enter”** key will toggle the state of Start_Sequence_RT input tag or Right-Click  [Start_Sequence_RT] => [Debug] => [Modify Value]
 - b. Click **“ON”/“OFF”** to modify value using Modify Value pop-up window.

Below sequence will be executed after **“Start_Sequence_RT”** input tag to **“ON”**

Following configuration and command is performed.

- Set the **“ModePos”** value to 6 & check the output tag **“ActMode”** has the same value as ModePos.
- Enable axis by toggling **“EnableAxis”** command tag.
- Check **“AxisRef”** status tag is True. If not true user could also use other modes like mode 4 or 5 to set the reference point for an axis.
- **“CancelTraversing”** and **“IntermediateStop”** command tags must be in **TRUE** state.
- **“CancelTraversing”** and **“IntermediateStop”** command tags must be in **TRUE** state.
- Enter the percentage of Override acceleration, Override deceleration & Override velocity in **“OverAcc”**, **“OverDec”** & **“OverV”** respective input tags.
- The status tags **“AxisEnabled”** & **“AxisRef”** must be **TRUE**. Also status tag **“Error”** must be false.
- Toggle the **“ExecuteMode”** command tag (Rising Edge) to start a motion task using the configured parameters (Every new position task would require a toggle of the **“ExecuteMode”** command tag).
- User will receive execute command accepted handshake signal from **“AxisAckSetpoint”** status tag.
- If task is successfully completed, it is acknowledge by rising edge of **“AxisPosOk”** output tag. Also if the task is unsuccessful **“AxisError”** output tag will be **TRUE**.
- **“AxisSpFixed”** Status tag indicates the axis mobility status . If the axis is standby the **“AxisSpFixed”** status tag will be high.

Sample Code Description for Torque Control using PTP_Drives_Festo_MB FB with EPD and PTP_Drives_Festo_MB FB with Record Table.



12 Additional Information

12.1 How to setup “Travel to fixed stop” with CMMT-AS by using the PTP/telegram 111

12.1.1 Application description

This document was created to be able to set-up the CMMT-AS into the “Travel to fixed stop” (application class 3) mode with using the PTP_Drives_Festo_MB function block and the telegram 111. This instruction shows the needed and useful parameters on the CMMT-AS drive.

In general, the PTP_Drives_Festo_MB function block does not offer in- or output interfaces for a “Travel to fixed stop”. To setup this mode it needs some changes on the default settings of the control words/ConfigEPos at PTP_Drives_Festo_MB function block side.

Festo named the “Torque Mode” as “Travel to fixed stop”. This mode is a positioning task with a defined clamping torque.

12.1.2 Used Parameters for Torque

Name	Parameter Number
Clamping torque [Nm]	526801
Fixed stop detection monitoring window	4694
Fixed stop detection damping time	4693
Fixed stop negative stroke limit	11280409
Fixed stop positive stroke limit	11280408
Lower limit value torque	852
Upper limit value torque	853
Target position	11280604
Profile speed	11280605
Monitoring window target torque	4611
Damping time target reached	468

12.1.3 Travel top fixed stop (application class 3)

Travel to fixed stop performs a positioning with reference to a defined max. clamping torque. During travel to fixed stop a fixed stop is approached from the current position before reaching the target position (e.g. at a work piece). Then a torque is established up to the desired clamping torque. For example, the following parameters can be set:

- Position
- Speed
- Acceleration
- Deceleration
- Clamping torque
- Clamping torque offset

A current positioning task can be switched by "STW2.8 Traverse to fixed end stop". The switching runs a positioning task with clamping torque. The closed-loop limit manager limits the motion to the clamping torque. On completion of the task the original limit is restored.

The following error monitor is not active during the task and the following status bits are set:

- ZSW2.8 Travel to fixed stop active
- POS_ZSW2.14 Move to fixed stop active

The following error monitor of the motion monitor is used during the task to detect the fixed stop.

When the fixed stop is detected, "POS_ZSW2.12 Fixed stop reached" is set and the stroke limit monitor of the motion monitor is activated based on the current position.

With pending clamping torque "POS_ZSW2.13 Fixed stop Clamping torque reached" is set.

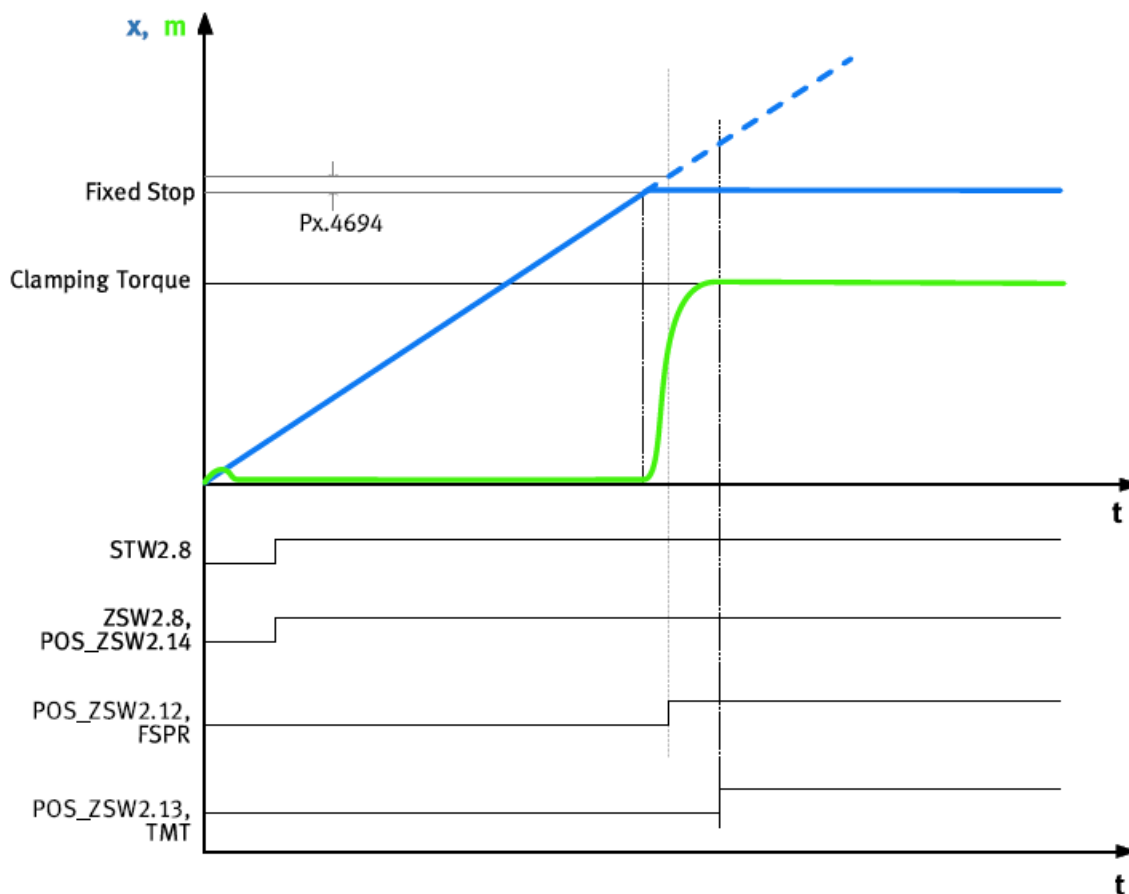
The clamping torque remains pending until a new travel command starts.

When the stroke limits for the fixed stop monitor are reached, "POS_ZSW2.12 Fixed stop reached" is

reset.

Timing

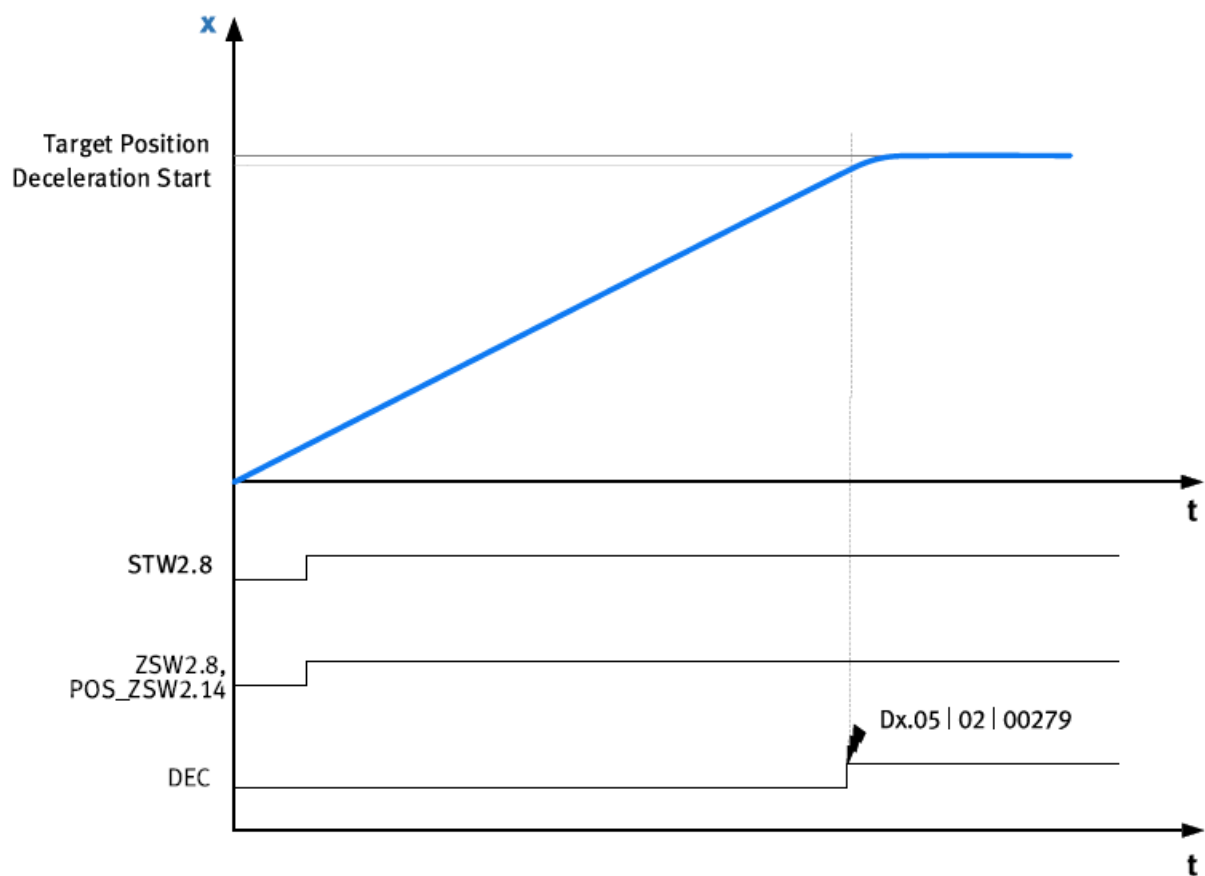
Example 1: travel to fixed stop with reaching and stopping at the fixed stop



Legend for timing diagram travel to fixed stop

Name	Description	Parameter Number
Fixed Stop	Fixed Stop	-
Clamping torque	Clamping torque	526801
FSPR	Motion monitoring function "fixed stop reached" (1 = status reached)	Px.460
TMT	Motion monitoring function "target torque range monitor" (1 = status reached)	Px.460

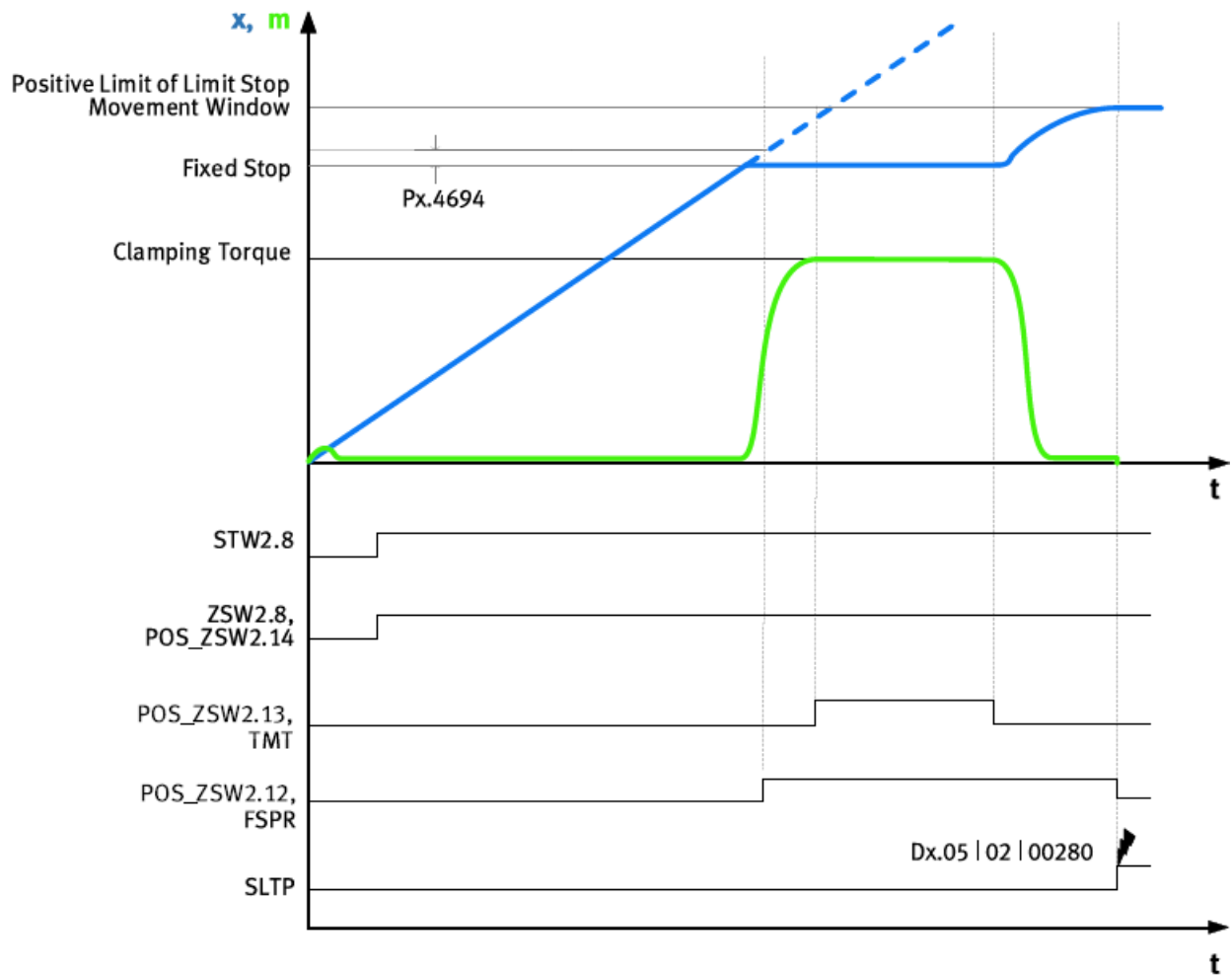
Example 2: travel to fixed stop without reaching the fixed stop



Information

The diagnostic message Dx.05.02.00279 must be activated in the Error classification within the Festo Automation Suite

Name	Description	Parameter
Target position	Target position	-
Deceleration start	Start of deceleration	-
DEC	Motion monitoring function "drive decelerated" (1 = status reached)	Px.460
Dx.05 02 000279	Diagnostic message Fixed stop not detected	-



Information

The diagnostic message Dx.05.02.00280 must be activated in the Error classification within the Festo Automation Suite

Legend for timing diagram fixed stop feeds back

Name	Description	Parameter
Positive Limit of Limit Stop Movement Window	Positive stroke limit of fixed stop monitor	11280408 11280409
Clamping torque	Clamping torque	526801
Fixed Stop	Fixed Stop	-
TMT	Motion monitoring function "target torque range monitor" (1 = status reached)	Px.460
FSPR	Motion monitoring function "fixed stop reached" (1 = status reached)	Px.460
STLP	Motion monitoring function "stroke limit reached" (1 = status reached)	Px.460
Dx.05 02 280	Diagnostic message Monitoring window of fixed stop left	Px.460

12.1.4 General parameters for “Travel to fixed stop”

Used parameters are described in the software documentation of the CMMT-AS in detail.

Below were the parameter numbers and the related chapters which are specified in the official documentation of CMMT-AS.

Details of the motion monitoring functions -> Motion monitoring functions.

The fixed stop detection acts like the following error monitor for position with critical limit and timing

-> Following error. The following error of the position and a damping time are used (Px.4694, Px.4693).

The detection of the pending clamping torque acts like the target range monitor for torque with critical limit and timing -> Target area monitoring.

The monitoring of the stroke limits after detected fixed stop acts like the stroke limit reached motion monitor -> Stroke limit reached.

The window limits can be set in the positive and negative directions (Px.11280408, Px.11280409 -> Tab. 266 Parameters).

If the motion leaves the monitoring window in the positive and negative direction it is detected and triggers the following diagnostic message:

– Monitoring window of fixed stop left: Dx.05 | 02 | 00280

The following parameter determines the braking behaviour on exit of the monitoring window:

– Activation of automatic stop ramp stroke limit: Px.4675

The clamping torque depends on the direction of motion. The set clamping torque is added with the offset. This means that the resulting clamping torque depends on the sign of the offset.

An asymmetrical clamping torque can be set with the offset for suspended axes (parameter Clamping torque offset, Px.11280407).