

CPX-FB36 in Modbus/TCP mode

The application node contains a step by step explanation how to configure and handle a CPX-FB36 in Modbus/TCP mode with a Schneider M580 PLC and Unit Pro L V11.0

CPX-FB36

Title CPX-FB36 in Modbus/TCP mode
Version 1.10
Document no. 100080
Originalen
AuthorFesto
Last saved 20.10.2016

Copyright Notice

This documentation is the intellectual property of Festo AG & 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 AG & Co. KG.

Festo AG & 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 AG & Co. KG.

Festo AG & 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 AG & 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 Components/Software/ IP address used 5**
 - 1.1 Recommended manuals as reference 5
 - 1.2 Topology 5
 - 1.3 DIL settings of the CPX-FB36 6
- 2 Commissioning in Unity Pro L V11.0 8**
 - 2.1 Key requirements 8
 - 2.2 Establish the Modbus/TCP communication via IO-Scanning 9
- 3 How to parameterize the CPX-FB36 device in Modbus/TCP 19**
 - 3.1 Work with CPX saved parameters 19

1 Components/Software/ IP address used

Type/Name	Version Software/Firmware	IP address
CPX-FB36	REV 13	192.168.10.10
Schneider M580 BME P58 3040 PLC	OS-Version 2.10	192.168.10.1
FMT Software	4.21.203	--
Unit PRO L	V11.0	--

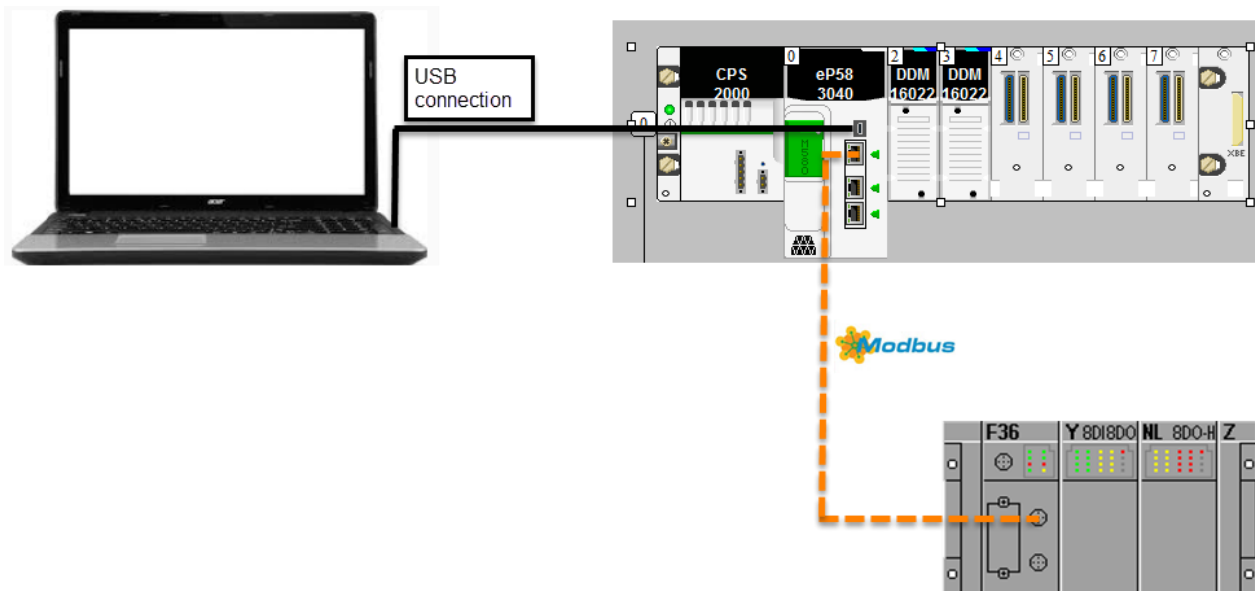
Table 1.1: 1 Components/Software used

1.1 Recommended manuals as reference

- CPX System manual
https://www.festo.com/net/SupportPortal/Files/407638/CPX-SYS_2009-02e_526446g1.pdf
- CPX-FB36 manual
https://www.festo.com/net/SupportPortal/Files/326813/CPX-FB36_2013-09_8024075g1.pdf

1.2 Topology

The CPX-FB36 is connected via M12-RJ45 Ethernet cable to the first port of the M580 PLC.



Note

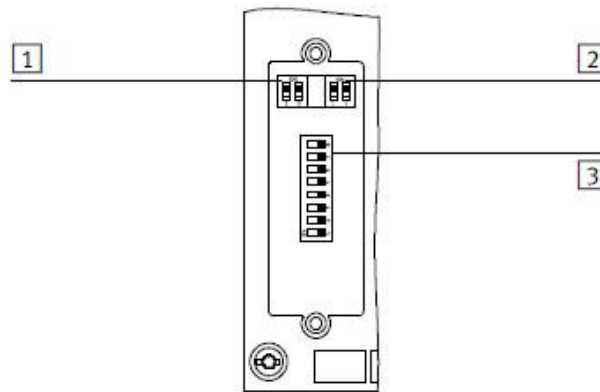
Festo offers M12-RJ45 and RJ45-RJ45 cable in SAP.

<i>Typecode: NEBC-D12G4-ES-1-S-R3G4-ET</i>	<i>pn: 8040451 (M12-RJ45 -> 1m length)</i>
<i>NEBC-D12G4-ES-3-S-R3G4-ET</i>	<i>pn: 8040452 (M12-RJ45 -> 3m length)</i>
<i>NEBC-D12G4-ES-5-S-R3G4-ET</i>	<i>pn: 8040453 (M12-RJ45 -> 5m length)</i>
<i>NEBC-R3G4-ES-1-S-R3G4-ET</i>	<i>pn: 8040455 (RJ45-RJ45 -> only 1m length available)</i>

1.3 DIL settings of the CPX-FB36

A) Overview of the DIL's:





- 1 DIL switch group 1:
Operating mode and protocol
- 2 DIL switch group 2:
Diagnostics mode for remote I/O or number of I/O bytes for Remote Controller
- 3 DIL switch group 3:
IP addressing



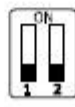
Procedure

1. Switch off the power supply.
2. Remove the DIL switch cover.
3. Change the DIL switch settings
4. Mount the cover.

B) Setting for Modbus/TCP Mode:

Operating mode and protocol	Setting of DIL switch group 1	
Remote I/O operating mode All functions of the CPX terminal are controlled directly via EtherNet/IP or Modbus TCP. A CPX-FEC or CPX-CEC that may be integrated into the CPX terminal works as a passive function module without controller.		DIL 1.1: OFF (factory setting)
Operating mode Remote Controller A CPX-FEC or CPX-CEC integrated into the CPX terminal takes over I/O control.		DIL 1.1: ON
EtherNet/IP protocol The CPX terminal uses the EtherNet/IP protocol.		DIL 1.2: OFF (factory setting)
Modbus TCP protocol The CPX terminal uses the Modbus/TCP protocol.		DIL 1.2: ON

Only Valid for Ethernet/IP mode.
In Modbus/TCP mode it is
always possible to use the I/O
Interface

Diagnostics mode for the Remote I/O operating mode	Setting of DIL switch group 2	
The I/O diagnostic interface and the status bits are switched off (+ 0 I/O bits)		2.1: OFF 2.2: OFF (factory setting)



C) IP address DIL settings:

1.2.4 Setting IP addressing

- Use DIL switch group 3 (→ Fig. 1/1 [3]).

By using DIL switch group 3 you can set the type of addressing or the IP address of the bus node.

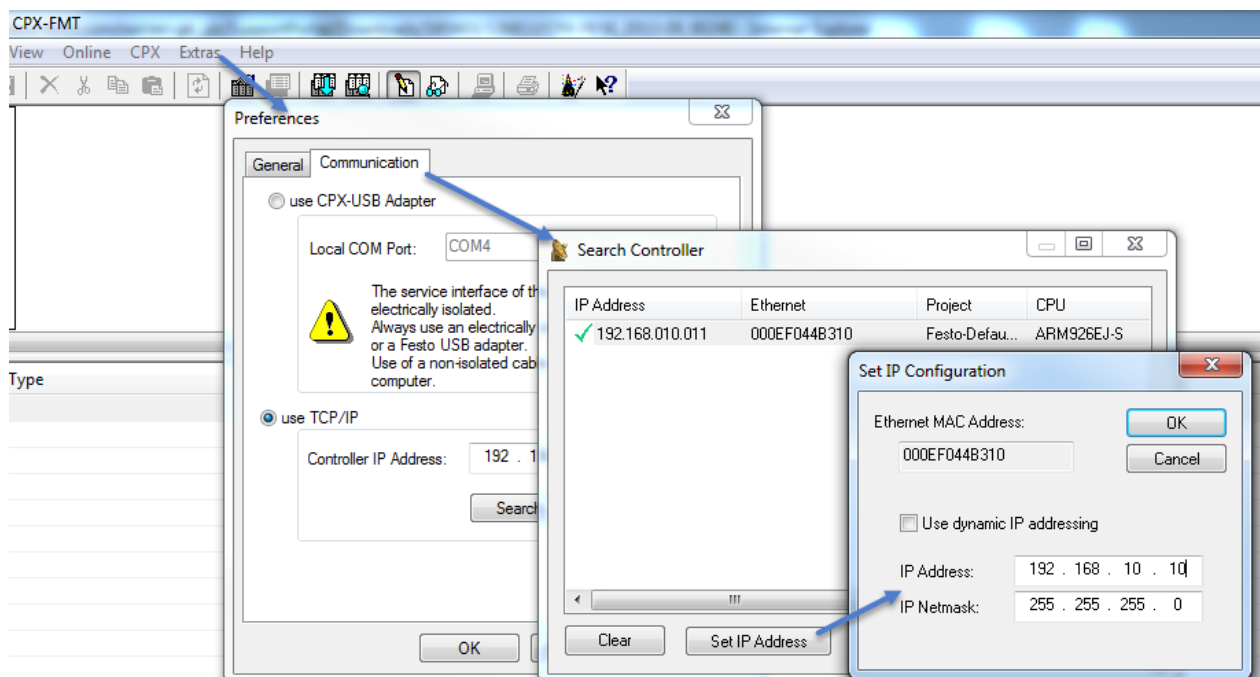
- Set all slide switches to "OFF", so that when the bus node is switched on it receives a dynamic or saved IP address (→ section 1.3.4).
- Use DIL switches 1 ... 8 to define a binary number other than 0 and 255.
This number is used as part of the IP address when turning on the bus node.

Dynamic/saved addressing	Fixed addressing
 <p>Setting: All switches OFF (factory setting)</p>	 <p>Setting: Host ID of the IP address</p>

Tab. 1/4: Settings for addressing type or IP address

**Note**

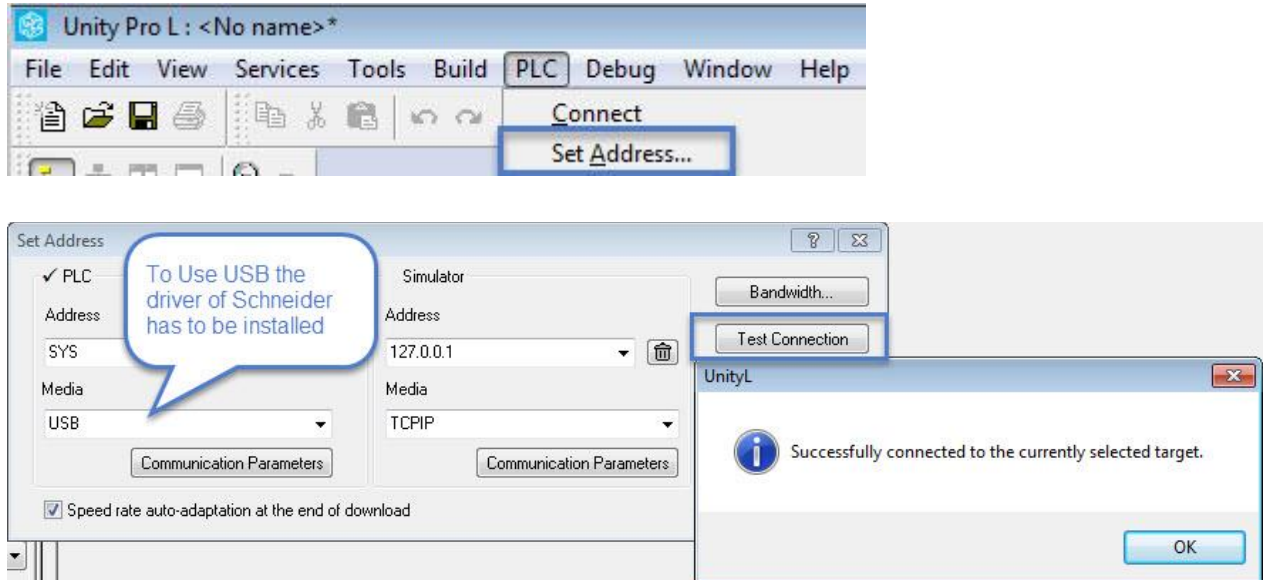
If you are using the factory settings then it is e.g. possible to use FMT Software for setting the IP address



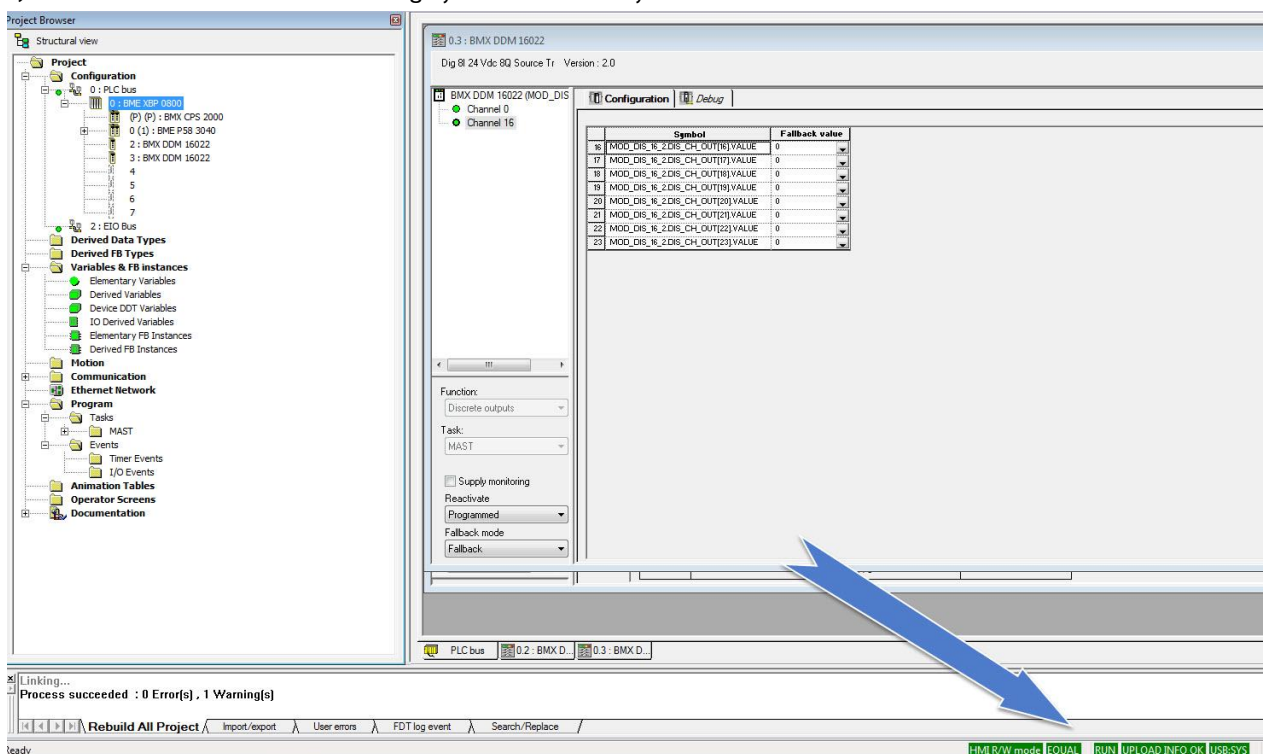
2 Commissioning in Unity Pro L V11.0

2.1 Key requirements

A) You have tested the Online connection to the PLC

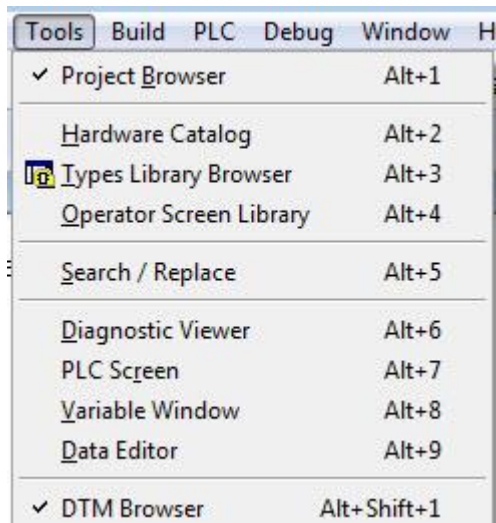


B) You have in Online mode a running system without any errors

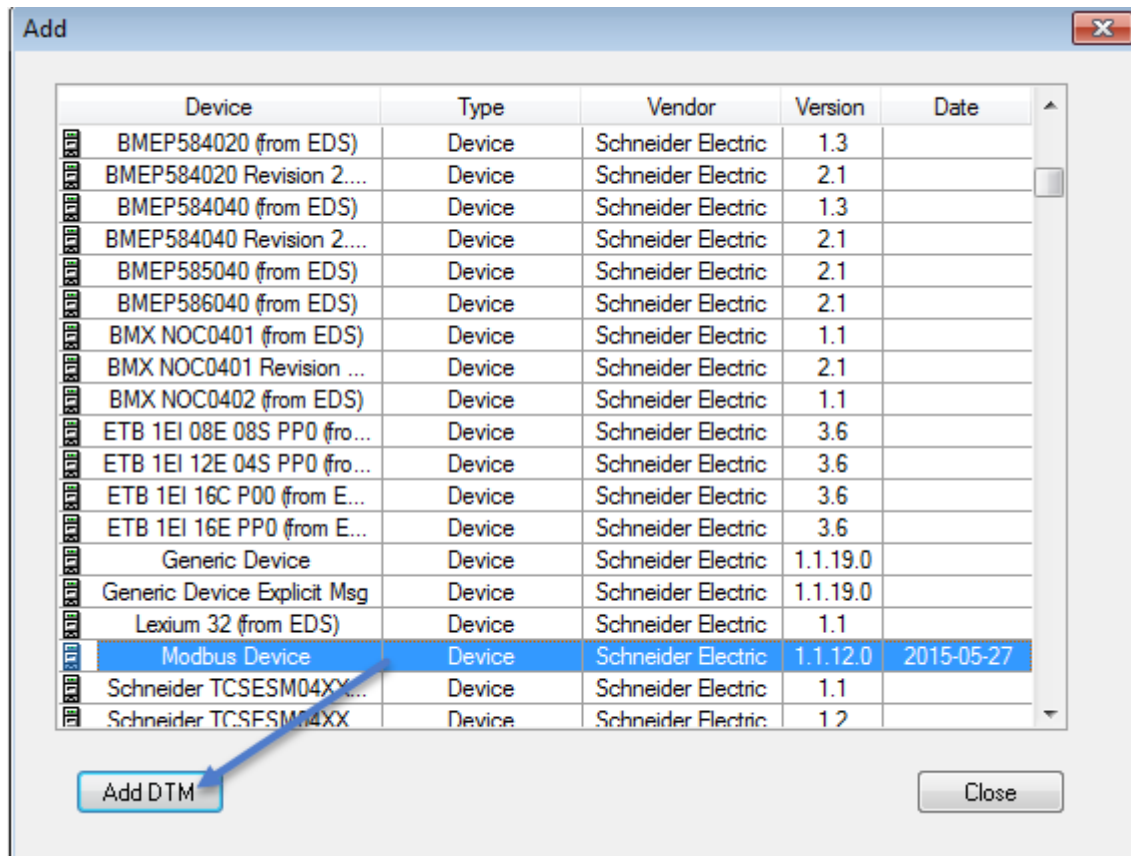
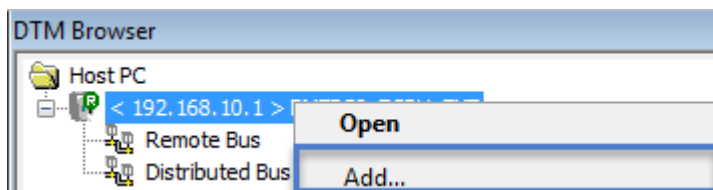


2.2 Establish the Modbus/TCP communication via IO-Scanning

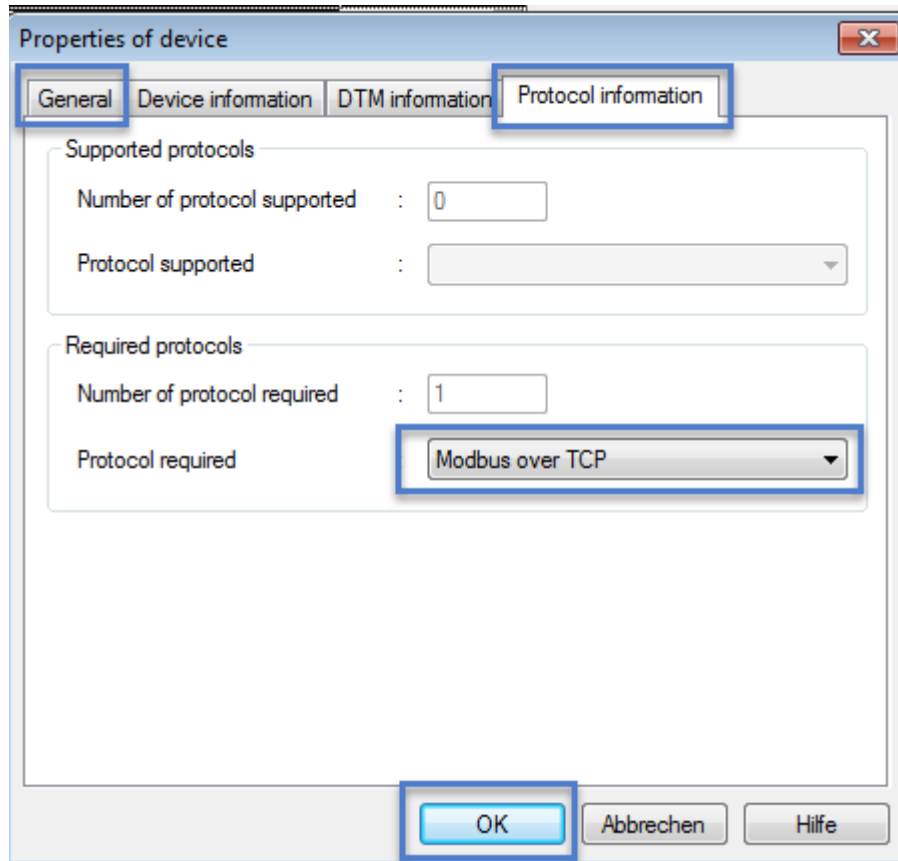
A) Start the DTM Browser



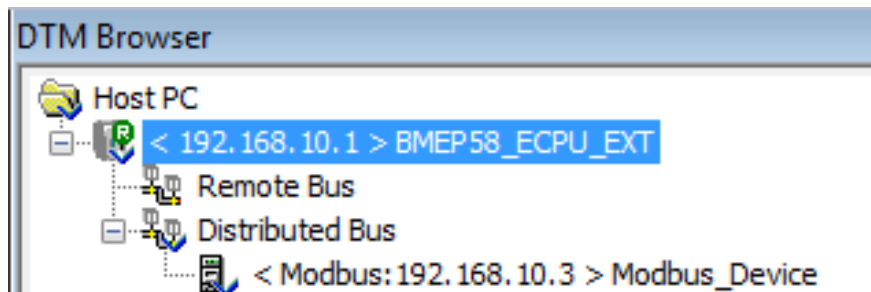
B) Add device



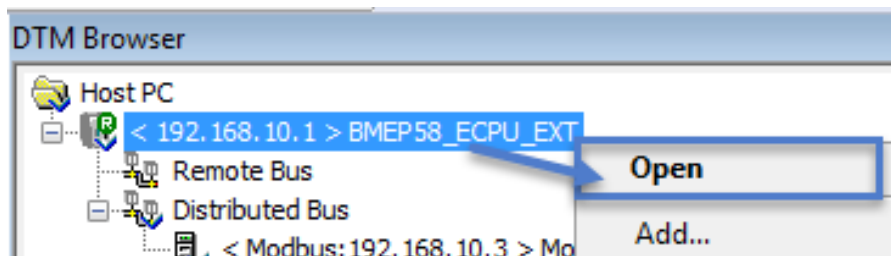
--> A Pop-up appears where you can change the name, check the protocol settings etc...



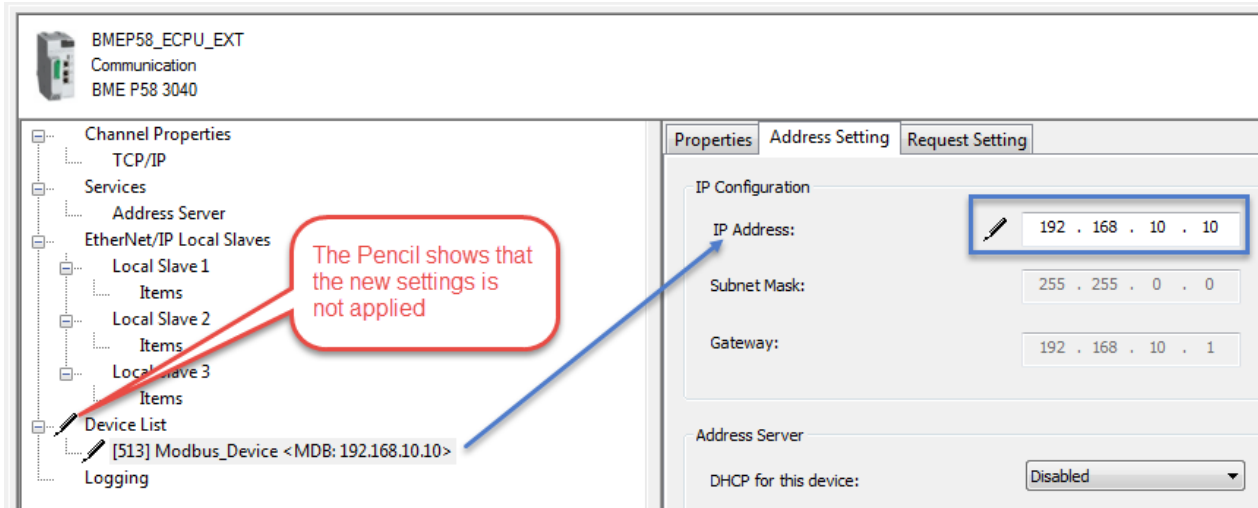
Result: The new Modbus/TCP device appears in the List



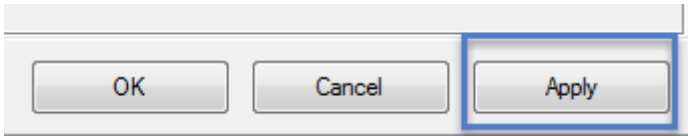
C) Change the Modbus/TCP device settings:



Part 1: Change the IP settings of the Modbus/TCP device



Apply the new settings!



➔ **Note**
It is a must to apply any change otherwise it is not valid!

Part2: Define The Modbus/TCP message



A new “Request” appears which includes changeable parameters

Optional Parameters (There is no “must” to change them):

Connection Bit	Unit ID	Health Time Out(ms)	Repetitive Rate(ms)
1	255	1500	60

Must parameters which you have to change:

Remove				
RD Address	RD Length	Last Value	WR Address	WR Length
0	1	Hold Value	0	0

Background information:**1. The CPX-FB36 can handle Modbus/TCP following codes:**

- FC3 means: Read multiple Holding register
(Standard, which the most Modbus/TCP master support)
- FC6 means: Write single Holding register
(Commonly, but not all Modbus/TCP master support it)
- FC16 means: Write multiple Holding register
(Standard, which the most Modbus/TCP master support)
- FC23 means: Read & Write multiple Holding register
(Commonly, but not all Modbus/TCP master support it -> The M580 PLC support it if you are using one request)
- FC43 means: Read device identification
(Seldom, but some Modbus/TCP master support it)

A register contains always 16 Bit information and the Function code defines “how you want” to get this Information.

With the Schneider M580 PLC the recommended function code is 23. That means with one telegram/request you can read and write all process data of the CPX system.

2. The Modbus address for read and write are starting not from 0

See CPX-FB36 manual information:

Modbus command	Function code	Modbus address	Meaning	Remote I/O 16-bit access	group
read 4x registers	3	45357...45391 45392...45647 45648...45655 45656...46055	CPX status information Processing data inputs Diagnostic memory parameters Diagnostic memory data	Read Read Read Read	A B C C
write 4x registers	6, 16	40001...40256 40257...40264	Processing data outputs Diagnostic memory parameters	Write Write	D E
read/write 4x registers	23	45357...45391 45392...45647 45648...45655 45656...46055 40001...40256 40257...40264	CPX status information Processing data inputs Diagnostic memory parameters Diagnostic memory data Processing data outputs Diagnostic memory parameters	Read Read Read Read Write Write	A B C C D E
read device identification	43	Objects	objects ID0, 1, 2, 3, 4, 5	Read	F

Tab. D/1: Overview of the Modbus function codes for the CPX-FB36 in the Remote I/O operating mode

3. The process data amount depends on the CPX system and the mapping rules are based on chapter D.4 of the CPX-FB36 manual

For example following valve terminal:



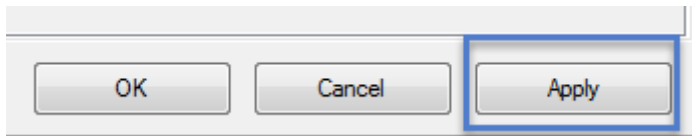
The mapping would be:

A	B	C	D
Modbus_Input_Address	Description	Modbus_output address	Description
45392	Feedback_I/O Interface	40001	Write_Access to I/O Interface
45393	Read access I/O data	40002	Write_data for I/O Interface
45394	diagnostic_info_FB36		
45395	CPX_8DI/8DO input info	40003	CPX_8DI/8DO output info
45396	Echo CPX_8DI/8DO output info		
45397	diagnostic_info_CPX_8DI/8DO		
45398	Echo CPX_8DO_H output info	40004	CPX_8DO_H output info
45399	diagnostic_info_CPX_8DO_H		
Total: 8 Input words = 16 Byte 4 Output words = 8 Byte			

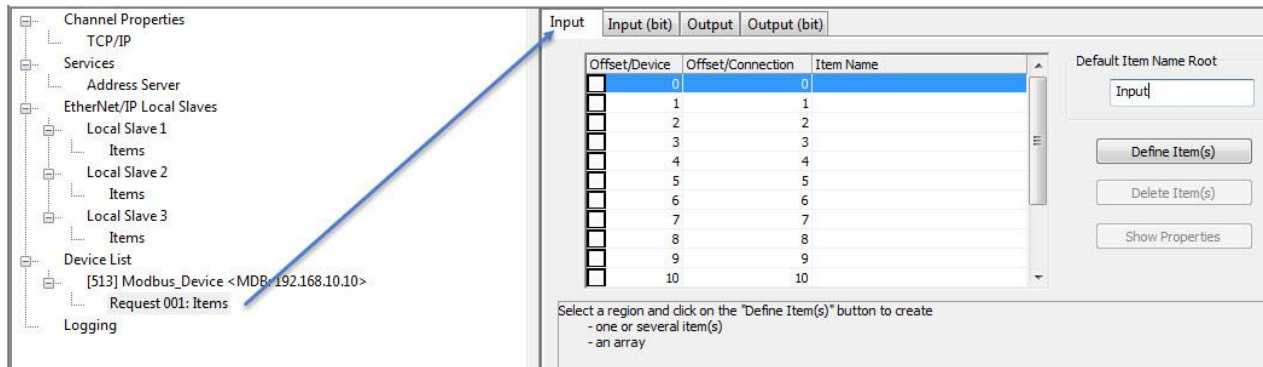
Because of the used CPX system following “Must” parameters have be done:

RD Address	RD Length	Last Value	WR Address	WR Length
45392	8	Hold Value	40001	4

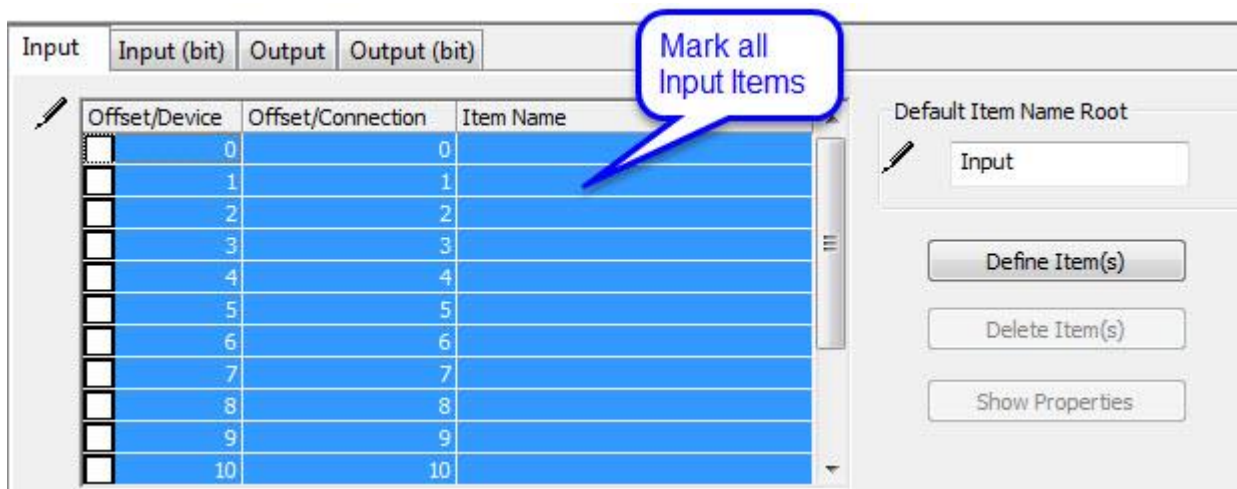
Apply the settings:



D) Create the Item(s) for the process data input and output

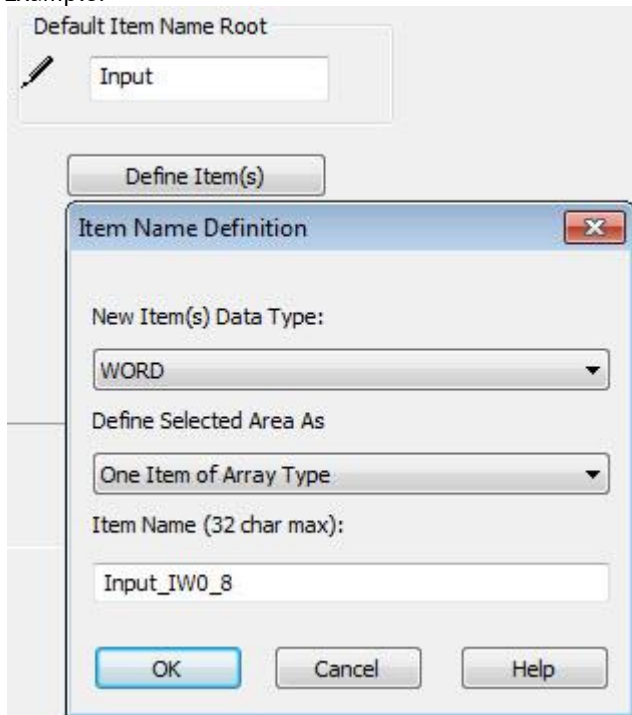


Part1: To do for the Inputs



Define the name, type etc...

Example:



Result:

Input

Input (bit)

Output

Output (bit)

16

16

16

16

16

16

Offset/Device	Offset/Connection	Item Name
0	0	[0] Input_IW0_8
1	1	1
2	2	2 [1]
3	3	3
4	4	4 [2]
5	5	5
6	6	6 [3]
7	7	7
8	8	8 [4]
9	9	9
10	10	10 [5]

Select a region and click on the "Define Item(s)" button to create

- one or several item(s)

- an array

Part2: To do for the Outputs

Input	Input (bit)	Output	Output (bit)																																			
<table><tr><th>Offset/Device</th><th>Offset/Connection</th><th>Item Name</th></tr><tr><td><input type="checkbox"/></td><td>0</td><td>0</td><td></td></tr><tr><td><input type="checkbox"/></td><td>1</td><td>1</td><td></td></tr><tr><td><input type="checkbox"/></td><td>2</td><td>2</td><td></td></tr><tr><td><input type="checkbox"/></td><td>3</td><td>3</td><td></td></tr><tr><td><input type="checkbox"/></td><td>4</td><td>4</td><td></td></tr><tr><td><input type="checkbox"/></td><td>5</td><td>5</td><td></td></tr><tr><td><input type="checkbox"/></td><td>6</td><td>6</td><td></td></tr><tr><td><input type="checkbox"/></td><td>7</td><td>7</td><td></td></tr></table>				Offset/Device	Offset/Connection	Item Name	<input type="checkbox"/>	0	0		<input type="checkbox"/>	1	1		<input type="checkbox"/>	2	2		<input type="checkbox"/>	3	3		<input type="checkbox"/>	4	4		<input type="checkbox"/>	5	5		<input type="checkbox"/>	6	6		<input type="checkbox"/>	7	7	
Offset/Device	Offset/Connection	Item Name																																				
<input type="checkbox"/>	0	0																																				
<input type="checkbox"/>	1	1																																				
<input type="checkbox"/>	2	2																																				
<input type="checkbox"/>	3	3																																				
<input type="checkbox"/>	4	4																																				
<input type="checkbox"/>	5	5																																				
<input type="checkbox"/>	6	6																																				
<input type="checkbox"/>	7	7																																				

Define the name, type etc...
Example:

Default Item Name Root

Output

Define Item(s)

Item Name Definition

New Item(s) Data Type:
WORD

Define Selected Area As
One Item of Array Type


























Item Name (32 char max):
Output_QW0_4

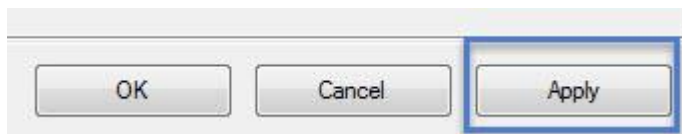
OK

Cancel

Help

Result:

Input	Input (bit)	Output	Output (bit)																											
<div><table><thead><tr><th>Offset/Device</th><th>Offset/Connection</th><th>Item Name</th></tr></thead><tbody><tr><td> 0</td><td>0</td><td>0 [0] Output_QW0_4</td></tr><tr><td> 16</td><td>1</td><td>1</td></tr><tr><td> 2</td><td>2</td><td>2 [1]</td></tr><tr><td> 16</td><td>3</td><td>3</td></tr><tr><td> 4</td><td>4</td><td>4 [2]</td></tr><tr><td> 16</td><td>5</td><td>5</td></tr><tr><td> 6</td><td>6</td><td>6 [3]</td></tr><tr><td> 16</td><td>7</td><td>7</td></tr></tbody></table></div>				Offset/Device	Offset/Connection	Item Name	 0	0	0 [0] Output_QW0_4	 16	1	1	 2	2	2 [1]	 16	3	3	 4	4	4 [2]	 16	5	5	 6	6	6 [3]	 16	7	7
Offset/Device	Offset/Connection	Item Name																												
 0	0	0 [0] Output_QW0_4																												
 16	1	1																												
 2	2	2 [1]																												
 16	3	3																												
 4	4	4 [2]																												
 16	5	5																												
 6	6	6 [3]																												
 16	7	7																												

Part3: Apply

D) Check if the defined data are inside your variable table

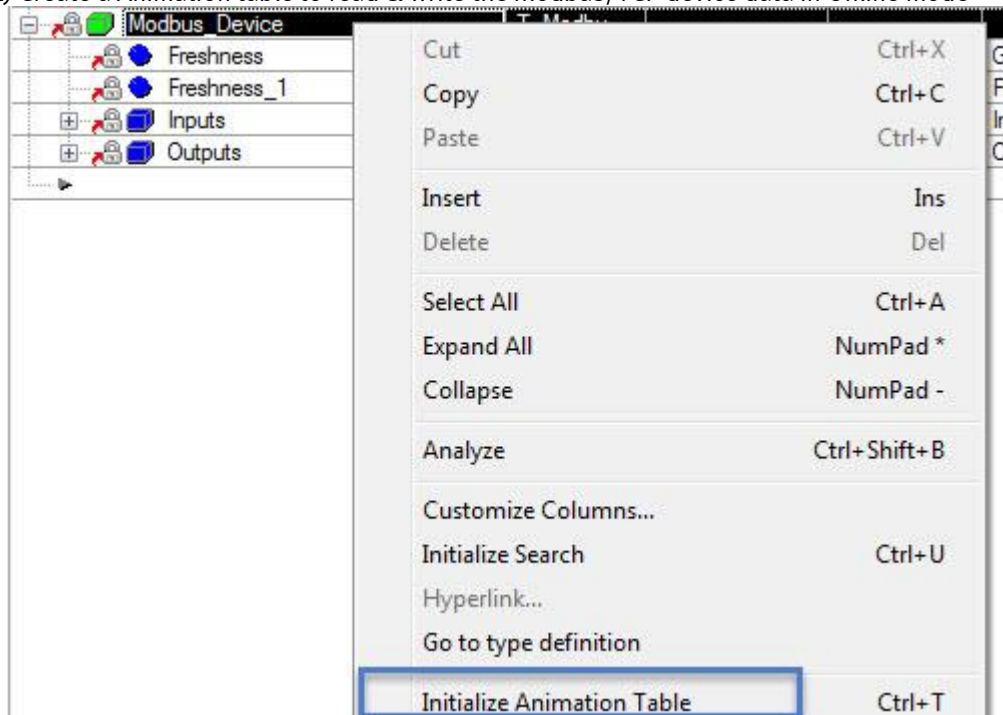
The Project Browser shows the following structure:

- Project
 - Configuration
 - Derived Data Types
 - Derived FB Types
 - Variables & FB instances
 - Elementary Variables
 - Derived Variables
 - Device DDT Variables
 - IO Derived Variables
 - Elementary FB Instances
 - Derived FB Instances
 - Motion
 - Communication
 - Ethernet Network
 - Program
 - Tasks
 - MAST
 - Events
 - Timer Events
 - I/O Events
 - Animation Tables
 - Operator Screens
 - Documentation

The Variable table shows the following variables:

Name	Type	Address
BMEP58_ECPU_EXT	T_BMEP5...	
MOD_DIS_16_1	T_U_DIS...	
MOD_DIS_16_2	T_U_DIS...	
Modbus_Device	T_Modbu...	
Freshness	BOOL	
Freshness_1	BOOL	
Inputs	T_Modbu...	
Input_IW0_8	ARRAY0...	
Input_IW0_8[0]	WORD	
Input_IW0_8[1]	WORD	
Input_IW0_8[2]	WORD	
Input_IW0_8[3]	WORD	
Input_IW0_8[4]	WORD	
Input_IW0_8[5]	WORD	
Input_IW0_8[6]	WORD	
Input_IW0_8[7]	WORD	
Outputs	T_Modbu...	
Output_QW0_4	ARRAY0...	
Output_QW0_4[0]	WORD	
Output_QW0_4[1]	WORD	
Output_QW0_4[2]	WORD	
Output_QW0_4[3]	WORD	

E) Create a Animation table to read & write the Modbus/TCP device data in Online mode



F) Rebuild project, go Online, download and start the Project

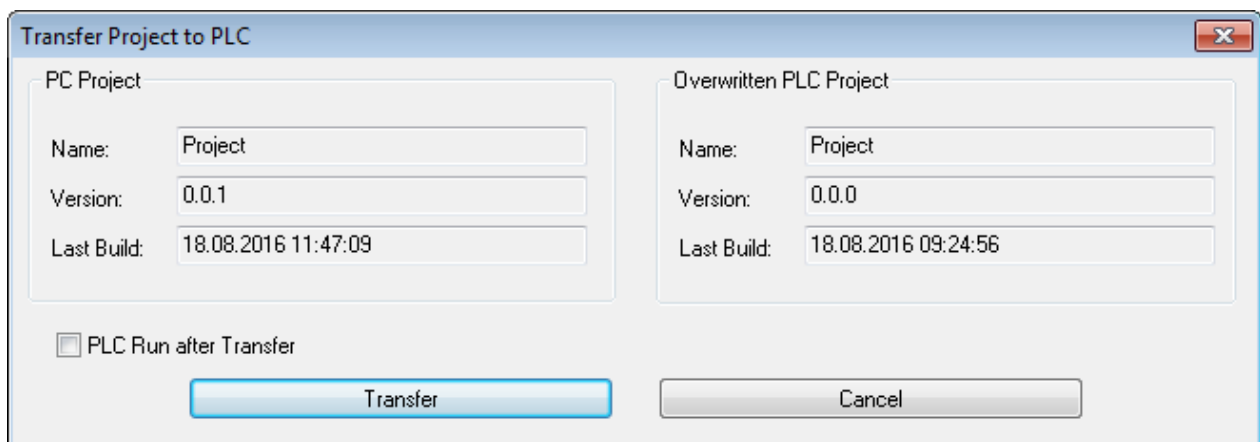
- Rebuild



- Connect



- Download



- Start



G) Test it with the Animation table in the Online mode

Modification Table:

Name	Value	Type	Comment
Modbus_Device		T_Modbus_Dev...	
Freshness	1	BOOL	Global Freshness
Freshness_1	1	BOOL	Freshness of Object
Inputs		T_Modbus_Dev...	Input Variables
Input_IW0_8		ARRAY[0..7] OF...	
Input_IW0_8[0]	0	WORD	
Input_IW0_8[1]	0	WORD	
Input_IW0_8[2]	0	WORD	
Input_IW0_8[3]	0	WORD	
Input_IW0_8[4]	255	WORD	
Input_IW0_8[5]	0	WORD	
Input_IW0_8[6]	255	WORD	
Input_IW0_8[7]	0	WORD	
Outputs		T_Modbus_Dev...	Output Variables
Output_QW0_4		ARRAY[0..3] OF...	
Output_QW0_4[0]	0	WORD	
Output_QW0_4[1]	0	WORD	
Output_QW0_4[2]	255	WORD	
Output_QW0_4[3]	255	WORD	

Annotations in the image:

- Blue arrow from **40004 CPX-8DO-H** points to the **Inputs** section of the table.
- Blue arrow from **40003 CPX-8DI/DO** points to the **Outputs** section of the table.
- Blue arrow from **Echosignal** points to the **Input_IW0_8[4]** and **Input_IW0_8[6]** rows.

Result on the valve terminal:



3 How to parameterize the CPX-FB36 device in Modbus/TCP

3.1 Work with CPX saved parameters

A) Download the FMT Software from the Support Portal

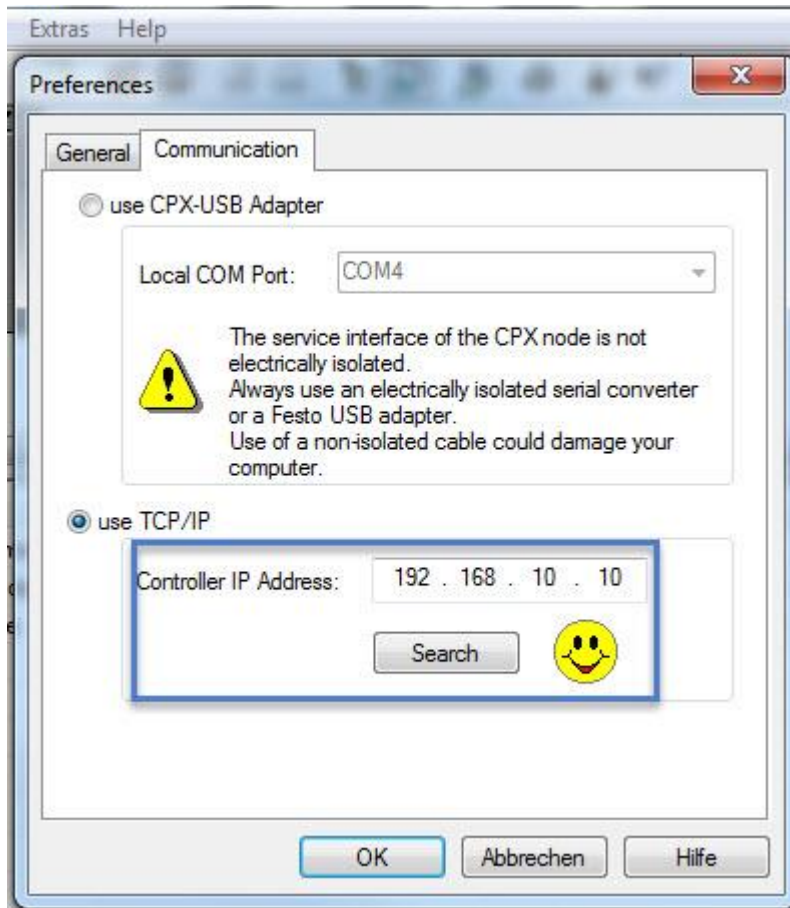
Support Portal

Please select a category on the left or use the search.

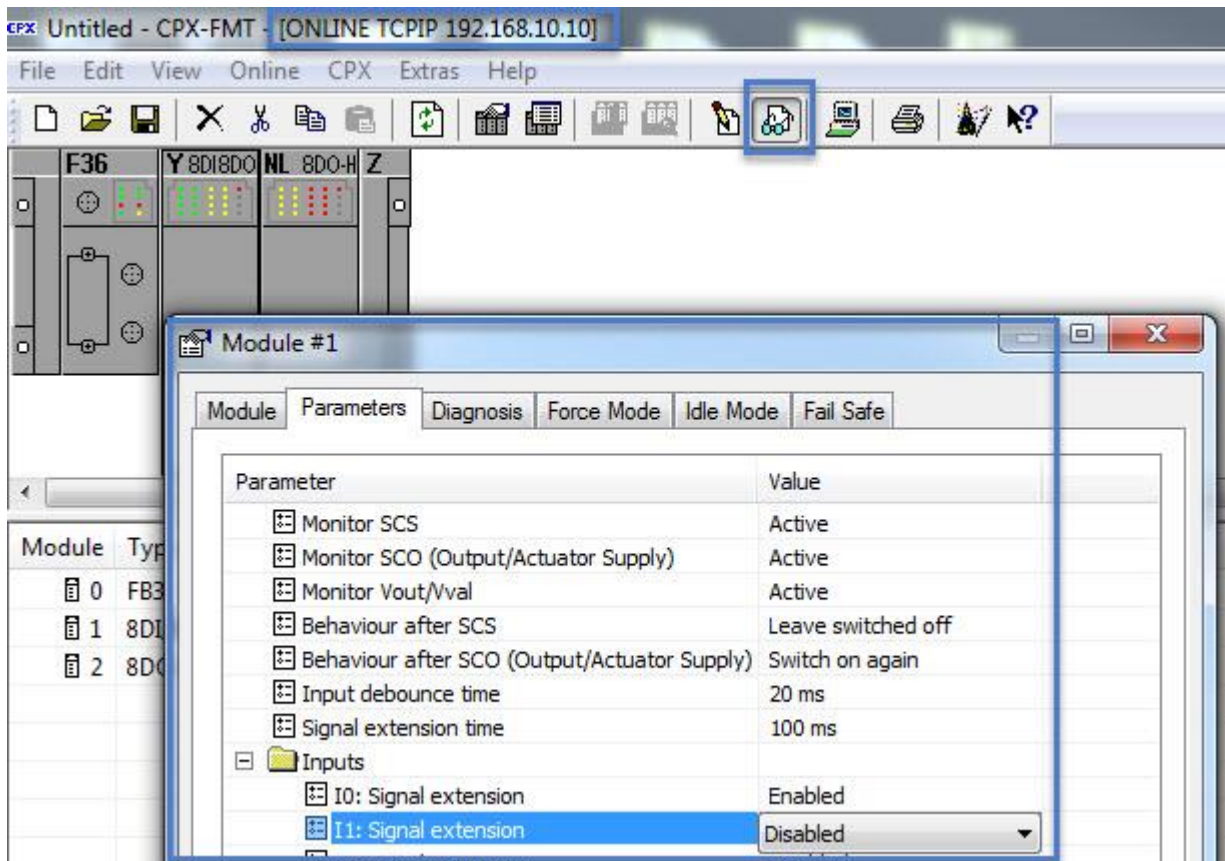
The screenshot shows the Festo Support Portal interface. At the top, there is a search bar with 'FMT' entered and a 'Find' button. To the right of the search bar is a Festo logo and a product code breakdown for 'DNC-125-100-PPV-A' (163501 R408 pmax. 12 bar), with labels for Part number, Series, and Order code. Further right are links for Contact, Product conformity, Terms and conditions of use for electronic documentation, and Support Community (new!). Below the search bar is a navigation menu with tabs: Top 3, Product information [16], Technical documentation [27], Certificates [0], Software [4] (highlighted), Expert knowledge [20], and Training [0]. The main content area displays search results for 'FMT'. The first result is 'FMT – Festo Maintenance Tool' with a description: 'CPX module catalogue update for FST 4.x and CPX-FMT'. It includes a checkbox icon and text: 'This update imports newer CPX modules into the module catalog of FST4.x and CPX-FMT.' The version is 'Update 14' dated '10/06/2015'. To the right of this result is a red callout box with the text 'Install Software and Update' and arrows pointing to the version and description. The second result is also 'FMT – Festo Maintenance Tool' with a description: 'The CPX-FMT is a helpful tool for commissioning, configuration and extended diagnosis of a CPX valve terminal. If Industrial Ethernet fieldbus nodes like'. The version is '4.21.203' dated '02/05/2014'. To the right of this result are links for Commissioning, File and language versions, and a star rating of 4.5 (48 reviews).

Description	Version	Filter result
FMT – Festo Maintenance Tool CPX module catalogue update for FST 4.x and CPX-FMT <input type="checkbox"/> This update imports newer CPX modules into the module catalog of FST4.x and CPX-FMT.	Update 14 10/06/2015	→ Com → File → Install Software and Update
FMT – Festo Maintenance Tool <input type="checkbox"/> The CPX-FMT is a helpful tool for commissioning, configuration and extended diagnosis of a CPX valve terminal. If Industrial Ethernet fieldbus nodes like	4.21.203 02/05/2014	→ Commissioning → File and language versions ★★★★★ (48)

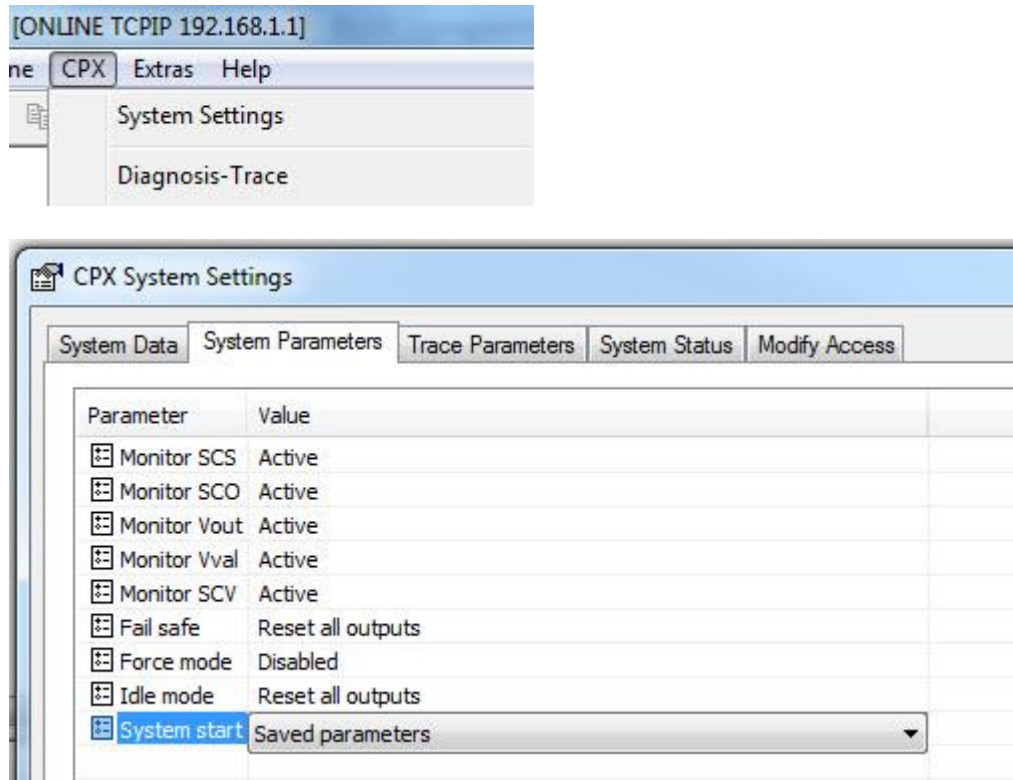
B) Establish an Ethernet communication to CPX-FB36



C) Go online and change the parameters of the modules



D) Activate Saved parameters



Result: If you are working with saved parameters the M-Led will be on



Advantage / Disadvantage

Positive: Easy to change parameters

All parameters are available after power off/on

Disadvantage: - Extra Software necessary

- If CPX-FB36 is damaged all set parameters are lost