PES: How to manage serial line on X80 drops with M580

Communication library – PES 4.1

Marco Siena – Gunawan Adisaputra

March 2016

Table of Contents

1 INTRODUCTION AND OBJECTIVE

The GPL Communication library contains objects managing explicit messaging for Modbus TCP and Modbus RTU technologies.

From PES 4.1, Modicon M580 platform has been introduced in PES, as well as some improvements in the above mentioned communication library.

This document will summarize some of these new features, especially concerning how to manage devices on serial lines connected to BMXNOM0200 module on M580 X80 drops.

2 CASE STUDY: M580 WITH X80 DROP MANAGING BMX NOM

2.1 Case study introduction

GPL Communication library provides templates for Modbus explicit communication management.

Three use cases have already been analysed in the following articles available on vCampus:

1. **Explicit communication – Ethernet devices**
   http://www.vcampus.schneider-electric.com/psx/technical-lab/plantstruxure-knowledge-base/1066-pes-gpl-ethernet-devices-explicit-communication

2. **Explicit communication – Modbus serial devices – Direct connection (on main rack)**
   http://www.vcampus.schneider-electric.com/psx/technical-lab/plantstruxure-knowledge-base/1063-pes-gpl-modbus-serial-devices-direct-conneciton

3. **Explicit communication – Modbus serial devices through Ethernet gateway**
   http://www.vcampus.schneider-electric.com/psx/technical-lab/plantstruxure-knowledge-base/1064-pes-gpl-modbus-serial-devices-through-gateway

This document will analyze a new use case, which is the usage of BMXNOM0200 Modbus serial module on X80 drops.

Note that BMXNOM0200 firmware must be 1.4 or later.

V1.4			The version V1.4 is needed in order to use the BMXNOM0200 in a X80 drop in a Quantum Ethernet IO architecture

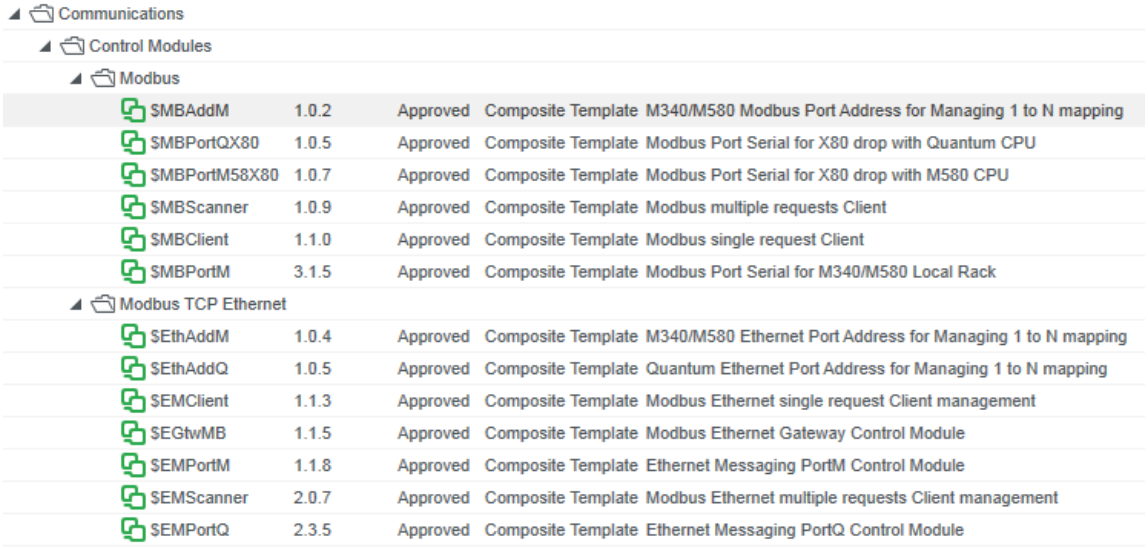
X80 RIO network can be managed either by Quantum or M580 systems (BMEP58*040 series). In the following chapter we will examine M580 option.

2.2 GPL Communication library state-of-art

The GPL library provided with PES 4.1 introduced the concept of physical port and logical port.
In previous versions, these two aspects were included in a single object, managing the complete port features.

Physical port is a simple object managing the topological address of an Ethernet or Modbus Port.
A single Physical Port object can be connected to multiple Logical Ports, in order to be able to manage independently the communications of a plant.
Logical port is a more complex object managing requests and answer to send and receive through the physical port.
PES engineers have to make sure that the sum of the amount of active requests configure for each Logical Ports is lower than the maximum amount of client requests manageable by the Physical Port per MAST cycle.

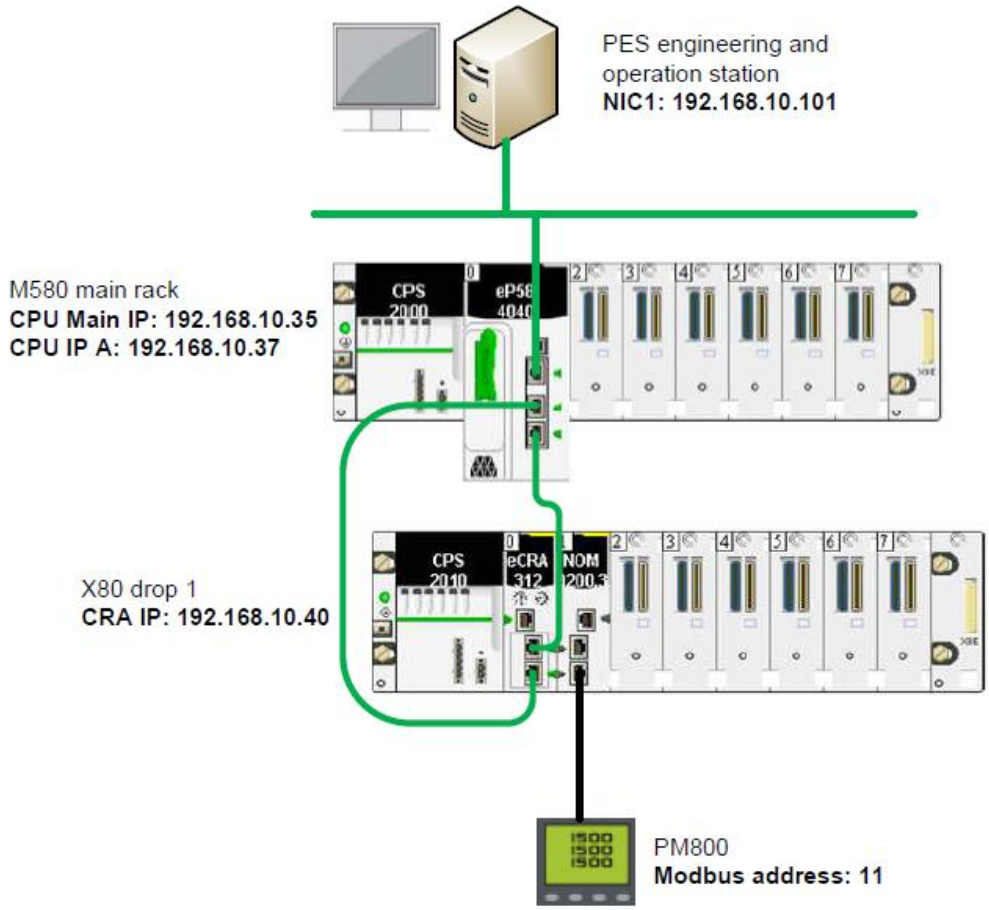
Logical port and Physical ports are platform dependent. The following image summarizes the characteristics of each object of communication library related to Modbus and Modbus TCP explicit communications.



2.3 Case Study architecture

The reference architecture for this example is represented in the following drawing.

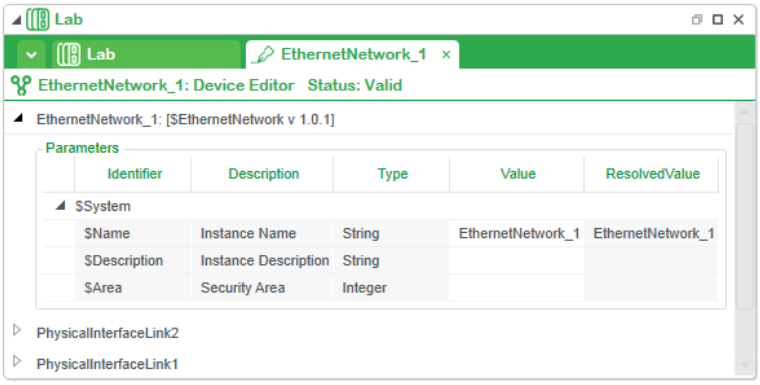
Note: this simplified example doesn't mention the drawbacks due to M580 ethernet backplane transparency: refer to M580 system planning guides in order to properly plan a consistent system architecture, especially in case of multiple M580 controllers connected to the same control network.



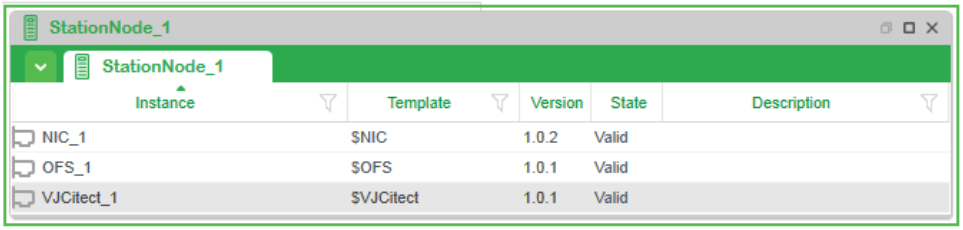
2.4 Topology manager

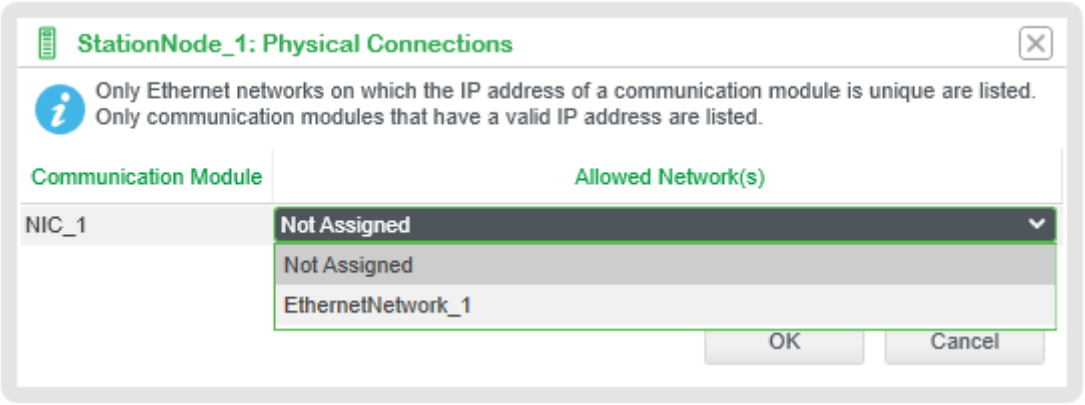
In the Topology Manager it is necessary to create the following items:

- One **Ethernet Network**

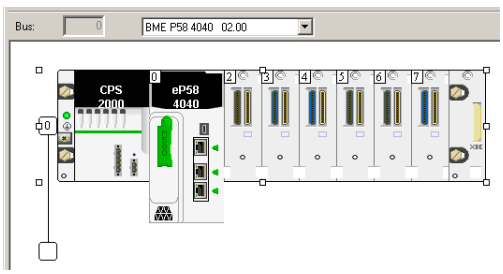


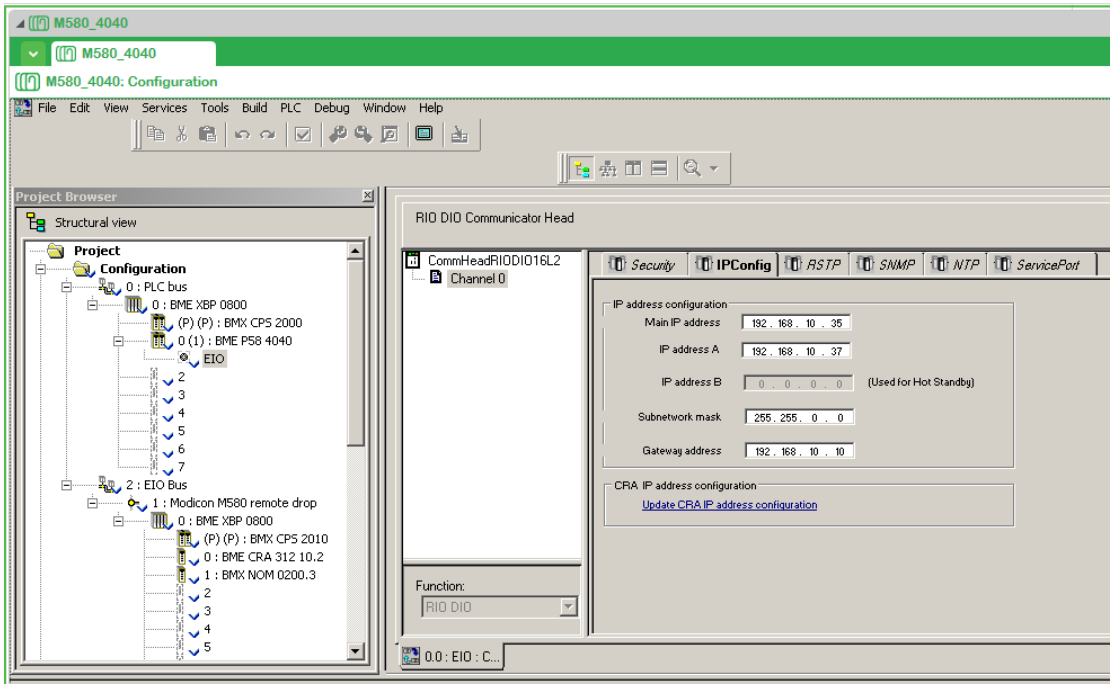
- One **Station Node** with NIC IP configuration as specified in previous chapter (or **127.0.0.1**). Connect the Station Node to the **Ethernet Network**

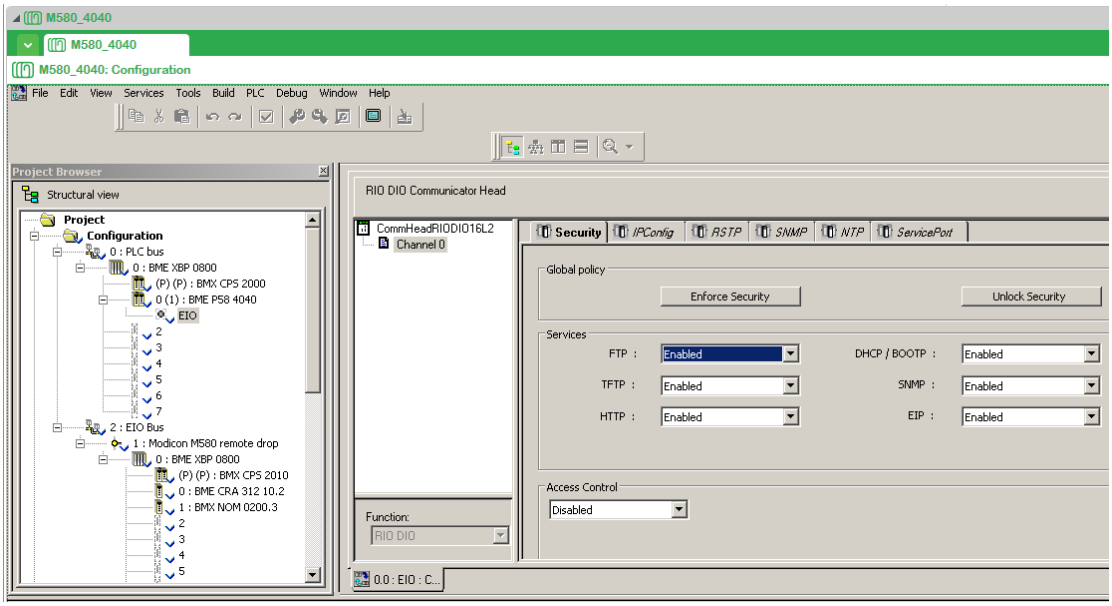


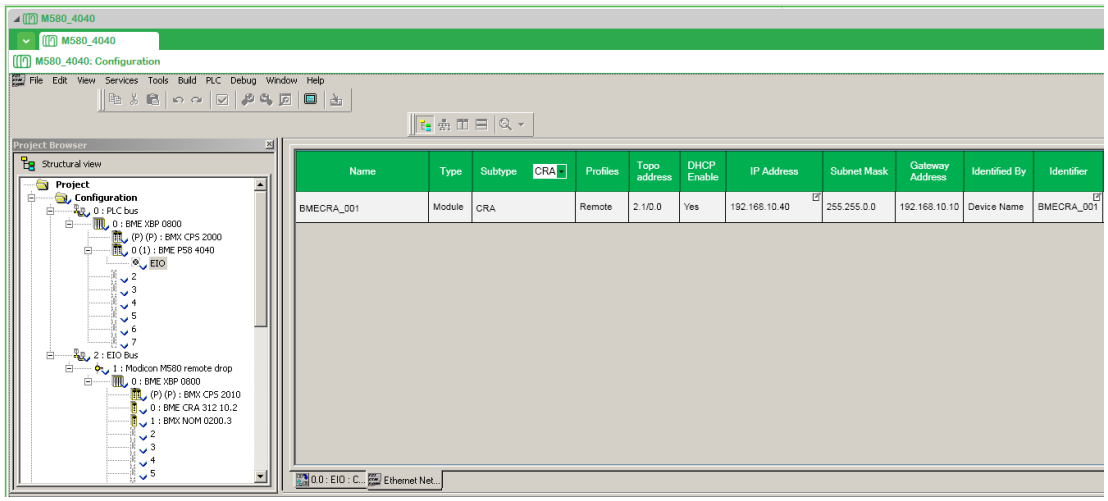


- One **M580 Controller** with IP configuration for **IP main**, **IP A** and **CRA drop 1** as specified in previous chapter. For this example it is suggested to “Unlock Security” on EIO Security tab.

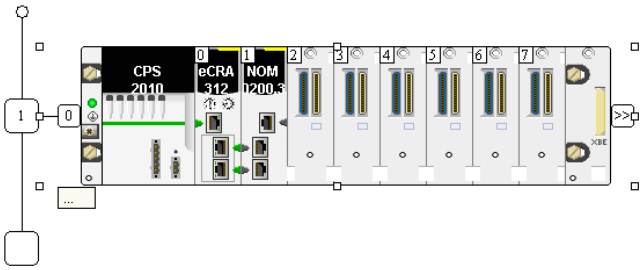


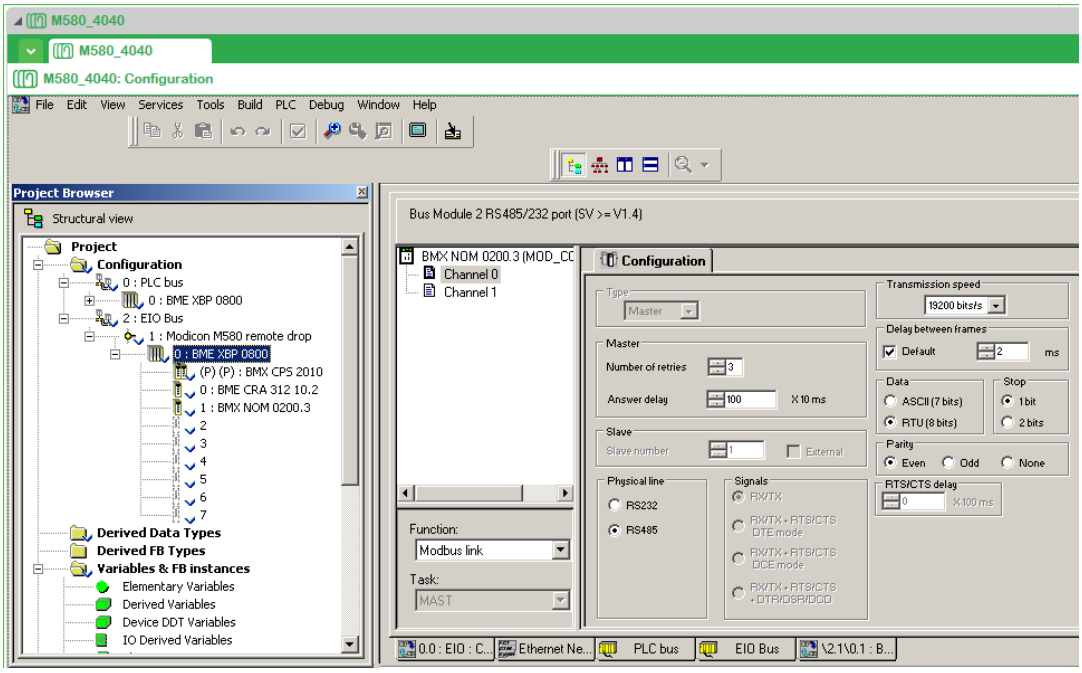

M580_4040
M580_4040
M580_4040: Configuration
File Edit View Services Tools Build PLC Debug Window Help
Project Browser
Structural view
Project
Configuration
0 : PLC bus
0 : BME XBP 0800
(P) (P) : BMX CPS 2000
0 (1) : BME P58 4040
EIO
2 : EIO Bus
1 : Modicon M580 remote drop
0 : BME XBP 0800
(P) (P) : BMX CPS 2010
0 : BME CRA 312 10.2
1 : BMX NOM 0200.3
RIO DIO Communicator Head
CommHeadRIODIO16L2
Channel 0
Security
IPConfig
RSTP
SNMP
NTP
ServicePort
IP address configuration
Main IP address 192 . 168 . 10 . 35
IP address A 192 . 168 . 10 . 37
IP address B 0 . 0 . 0 . 0 (Used for Hot Standby)
Subnetwork mask 255 . 255 . 0 . 0
Gateway address 192 . 168 . 10 . 10
CRA IP address configuration
Update CRA IP address configuration
Function:
RIO DIO
0.0 : EIO : C...


M580_4040
M580_4040
M580_4040: Configuration
File Edit View Services Tools Build PLC Debug Window Help
Project Browser
Structural view
Project
Configuration
0 : PLC bus
0 : BME XBP 0800
(P) (P) : BMX CPS 2000
0 (1) : BME P58 4040
EIO
2 : EIO Bus
1 : Modicon M580 remote drop
0 : BME XBP 0800
(P) (P) : BMX CPS 2010
0 : BME CRA 312 10.2
1 : BMX NOM 0200.3
RIO DIO Communicator Head
CommHeadRIODIO16L2
Channel 0
Security
IPConfig
RSTP
SNMP
NTP
ServicePort
Global policy
Enforce Security
Unlock Security
Services
FTP : Enabled
TFTP : Enabled
HTTP : Enabled
DHCP / BOOTP : Enabled
SNMP : Enabled
EIP : Enabled
Access Control
Disabled
Function:
RIO DIO
0.0 : EIO : C...


M580_4040
M580_4040
M580_4040: Configuration
Project Browser
Structural view
Name
Type
Subtype
CRA
Profiles
Topo address
DHCP Enable
IP Address
Subnet Mask
Gateway Address
Identified By
Identifier
BMECRA_001
Module
CRA
Remote
2.1/0.0
Yes
192.168.10.40
255.255.0.0
192.168.10.10
Device Name
BMECRA_001
0.0 : EIO : C...
Ethernet Net...

- One **X80 drop** with BMXNOM0200. Configure the Channel 0 as Modbus Link, RS485, 19200 Kbps, even parity.





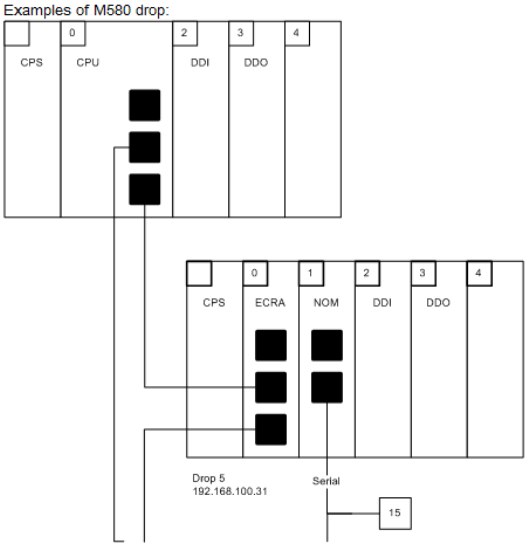
2.5 Application manager configuration

The object that will be in charge of managing Modbus explicit requests is named **`$MBPortM58X80`**.
It is specific for M580 platform and serial lines connected to BMXNOM in X80 drops.
The control facet contains a DFB, which includes explicit instructions for read and write requests management towards serial devices (PM800 in this case).
Since this object runs on the Controller, in order to have the proper routing of Modbus messages towards the BMXNOM0200, it is necessary to know the following information:

- **Rack** number, **Slot** and **Channel** used for sending ethernet requests.
- IP of the CRA **Drop** where the BMXNOM0200 is installed
- **Rack** number, BMXNOM0200 **Slot** and **Channel** used for routing request to serial line.

Normally, in Unity Pro participant, the address formatting is managed by the "**ADDMX**" EF, that is embedded inside **`$MBPortM58X80`** object.
In this case, the syntax obtained is: **ADDMX('0.0.3{192.168.10.40}\\0.1.0.11')**, see the following excerpt from Unity Pro Manual.

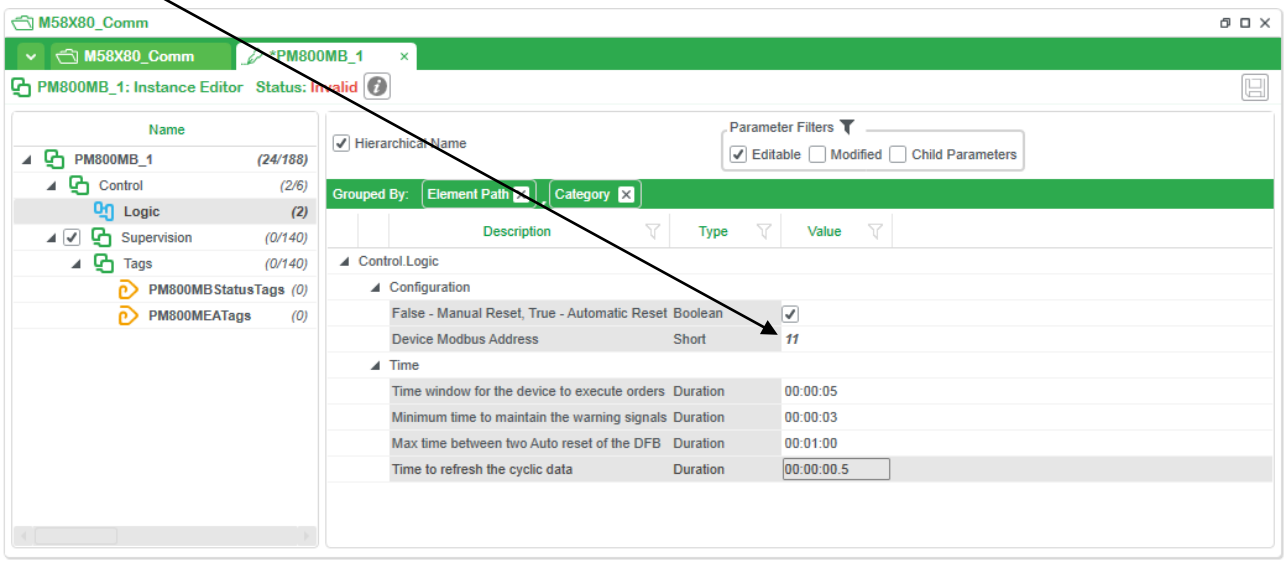
Examples of M580 drop:



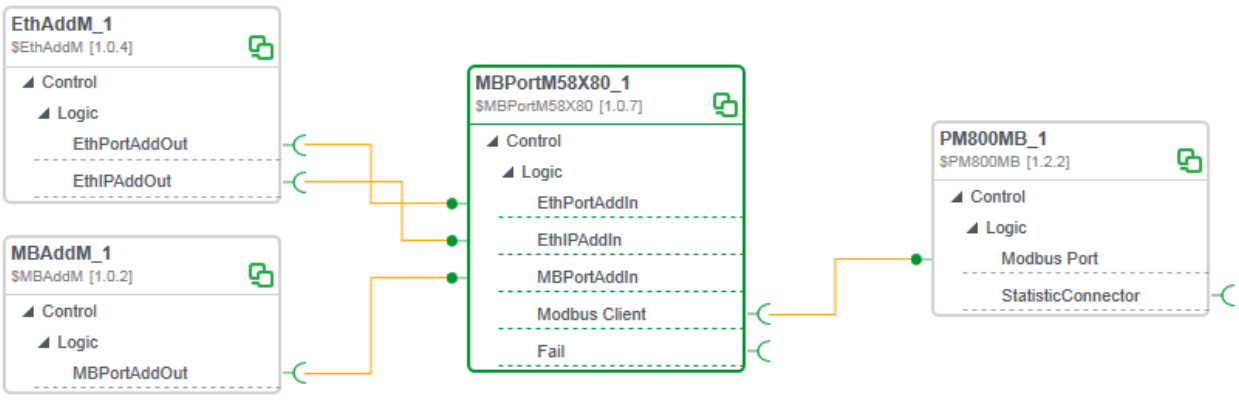
Device to Address	ADDMX Syntax
Module server of drop #5 CRA	ADDMX('0.0.3{192.168.100.31}SYS
Module server of BMX NOM in drop #5	ADDMX('0.0.3{192.168.100.31}\\0.1')
Modbus device #15 on serial link of BMX NOM in drop #5	ADDMX('0.0.3{192.168.100.31}\\0.1.1.15')

Based on these considerations, in this specific example we need one instance for the following objects:
$EthAddM ➔ M580/M340 physical ethernet port addressing
$MBAddM ➔ M580/M340 physical serial port addressing
$MBPortM58X80 ➔ X80 serial port management (BMXNOM0200)
$PM800MB ➔ PM800 modbus device object

Create all instances.
In this example, default values can be kept, except for **$PM800MB** configuration, where it is necessary to specify the Modbus slave address.

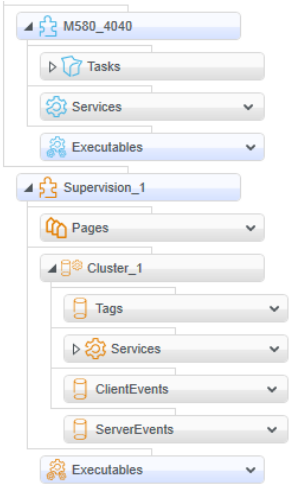


Next step is to create the proper links between instances, as shown in the following image.

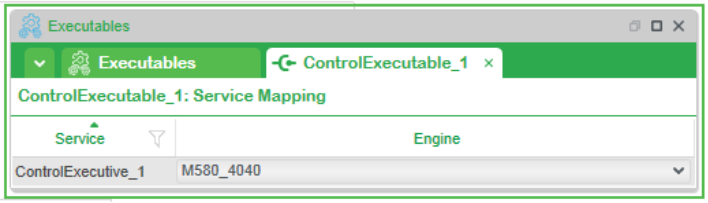


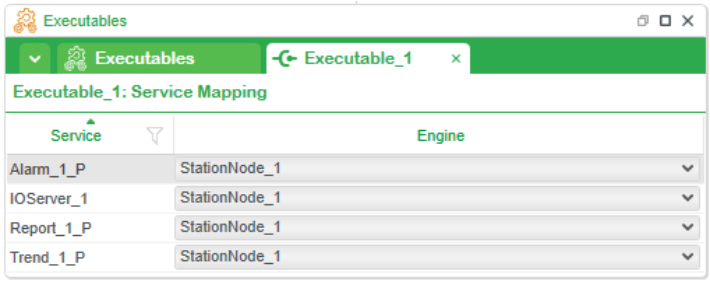
2.6 Project manager

In the PES project manager, we need to create a Control Project and a Supervision Project for the test platform.

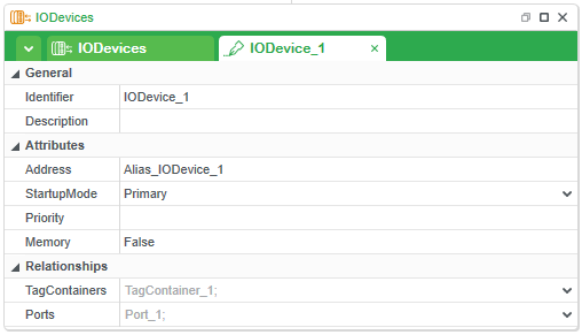


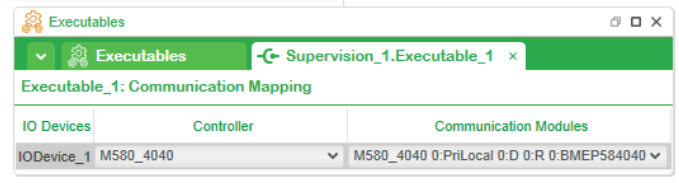
Control project has to be mapped on its counterpart in Topology, the same for Supervision project.





Create the communication mapping between Supervision and Control, as usual.



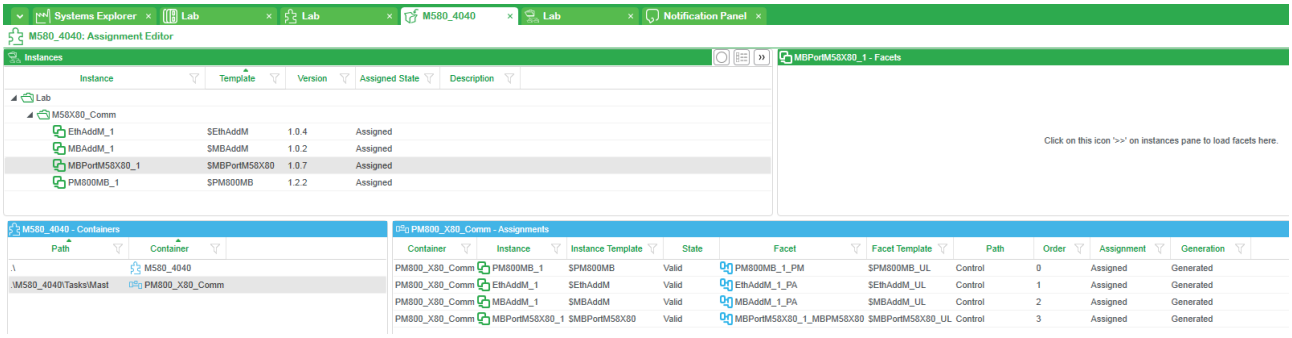


In Control project, it is necessary to assign and generate the facets of the object described in the previous chapter.

The assignment must respect the following ordering rule:

1- Serial device objects
2- Ethernet / Modbus serial addressing object
3- Port objects

Assign and generate objects.

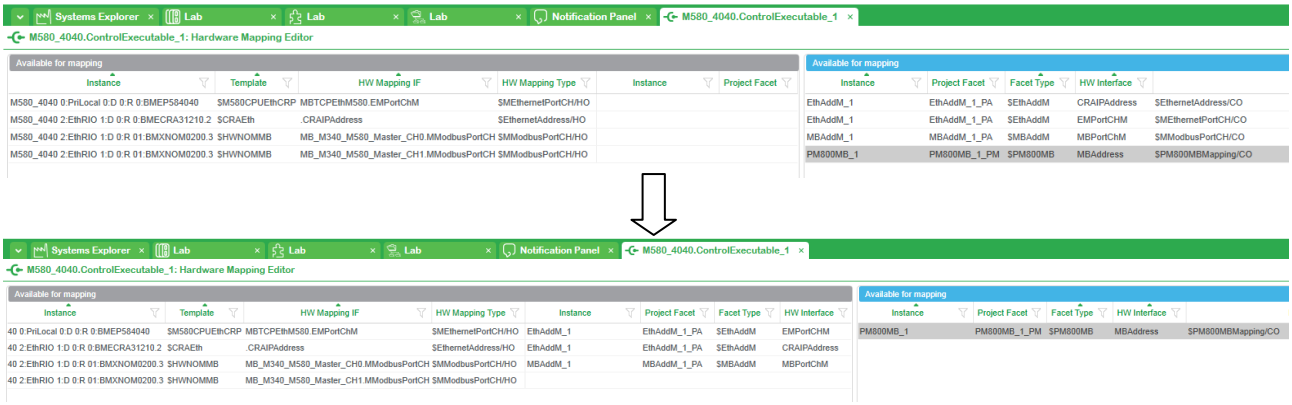


The next step is the hardware mapping.

We need to map:

1- M580 CPU port addressing ➔ **EthAddM_1** instance ➔ **`EMPortCHM`** HW interface
2- X80 CRA port addressing ➔ **EthAddM_1** instance ➔ **`CRAIPAddress`** HW interface
3- BMXNOM Modbus serial port addressing➔ **MBAddM_1** instance ➔ **`MBPortChM`** HW interface

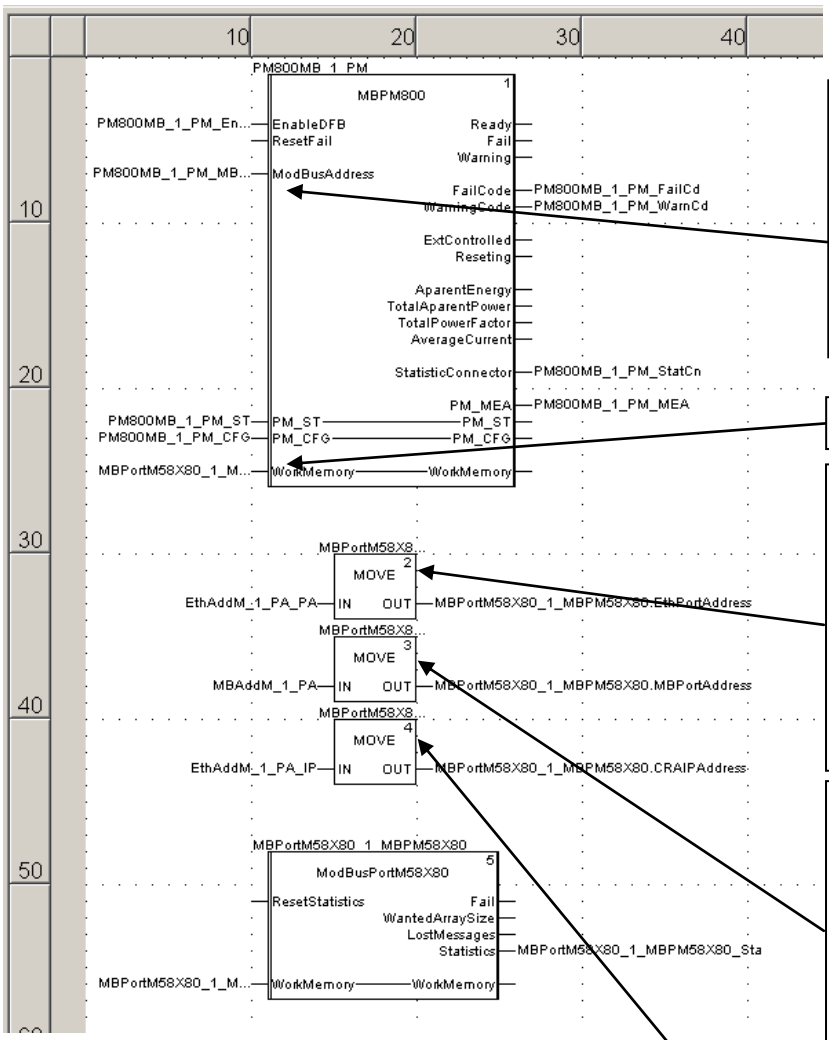
PM800MB instance doesn't need any specific mapping.



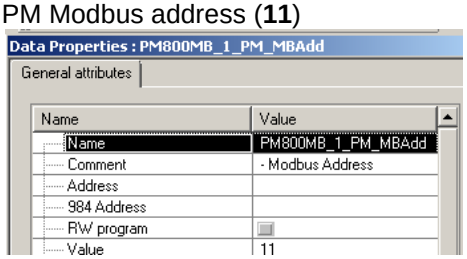
Generate and build the Control Project.

The following image shows the result of the build (Open Built Project mode).

The information related the modbus routing path are copied on the relevant public variables of **`MBPortM58X80`** object, which will define the **ADDMX** formatting inside the DFB.

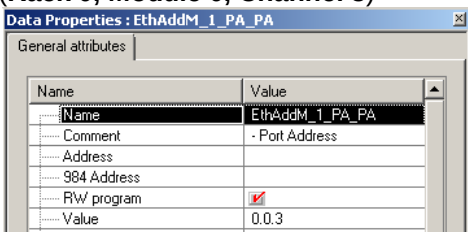


PM Modbus address (**11**)

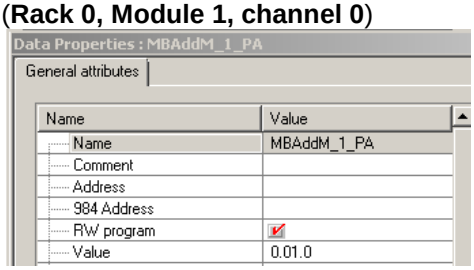


WorkMemory link

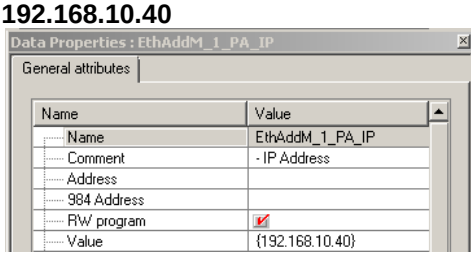
Ethernet port address assignment (**Rack 0, Module 0, Channel 3**)



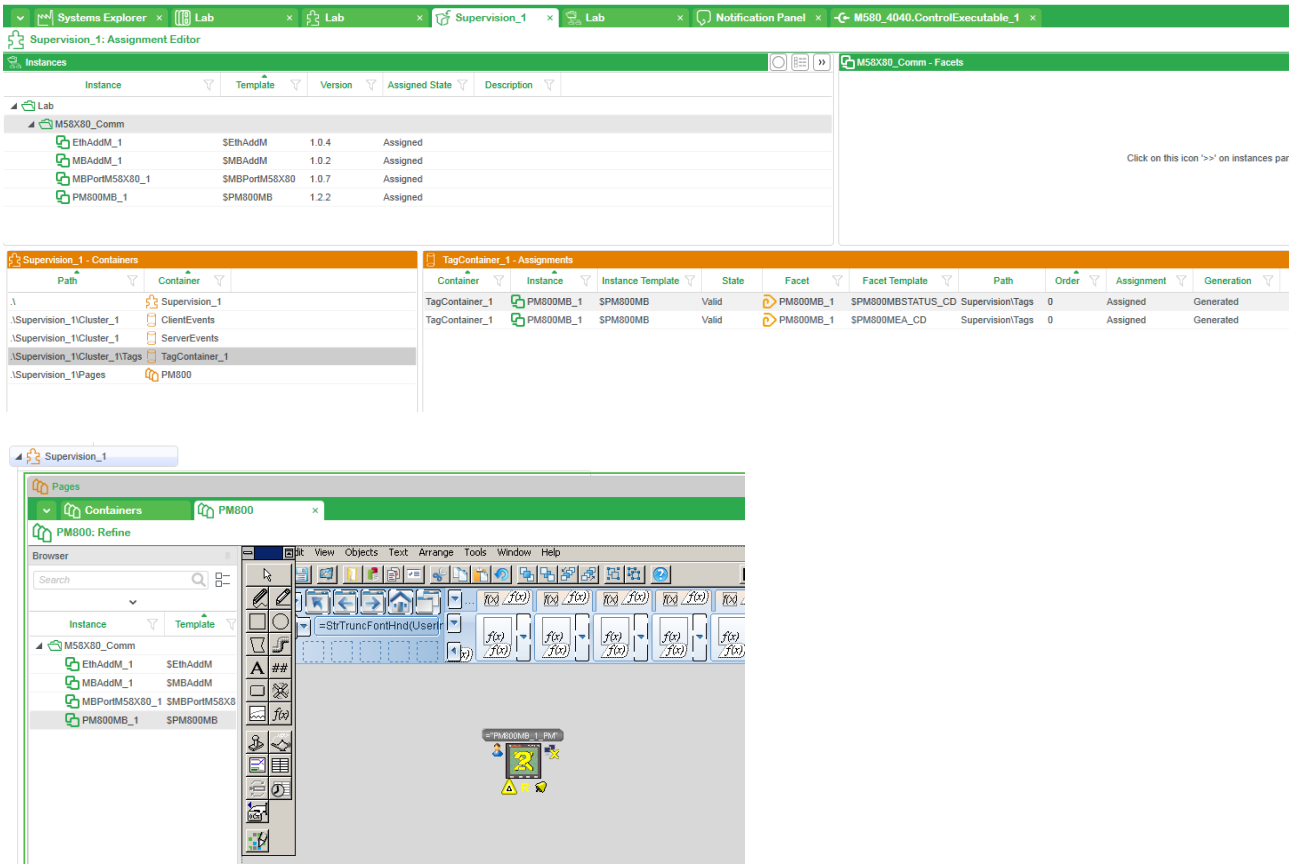
NOM modbus port address assignment (**Rack 0, Module 1, channel 0**)



CRA address assignment **192.168.10.40**



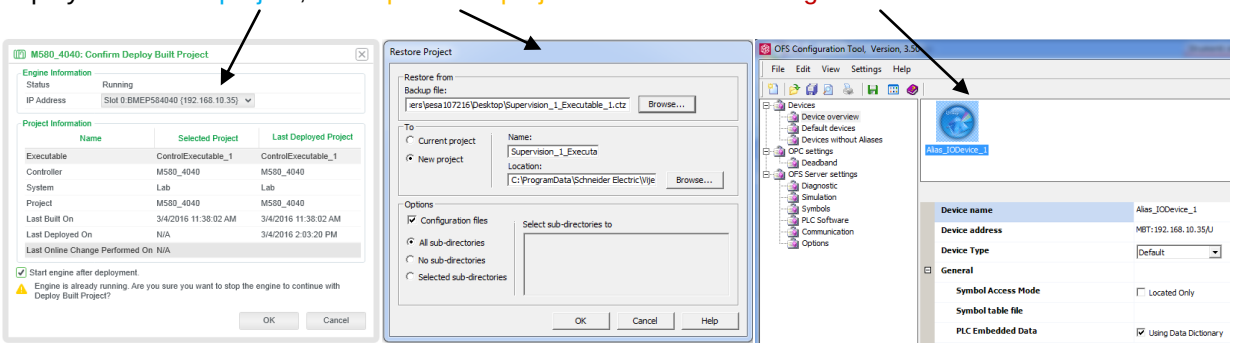
On Supervision project, assign PM800 tags to the Tag Container and create one page with PM800 genie.



Generate and build the Supervision project.

2.7 Deployment of executables

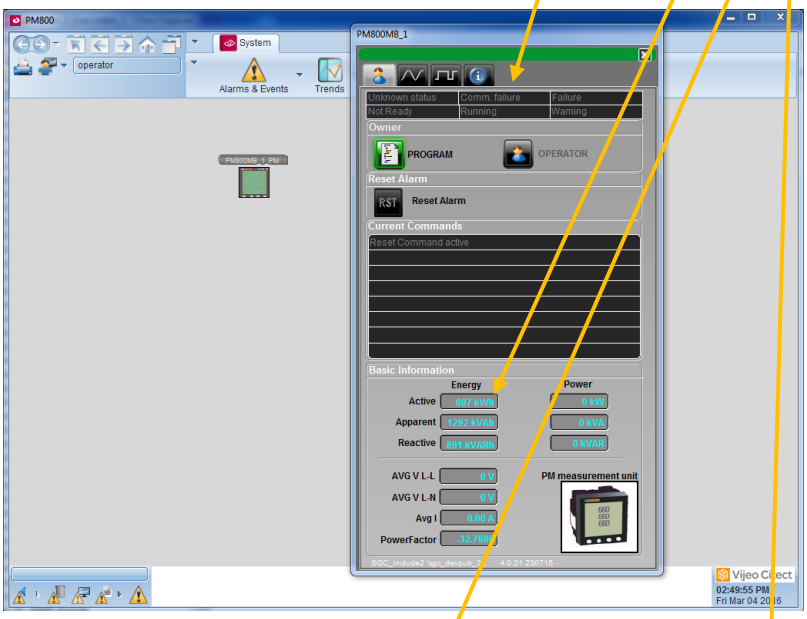
Deploy the control project, the supervision project and the OFS configuration.

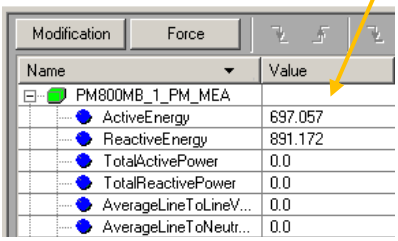


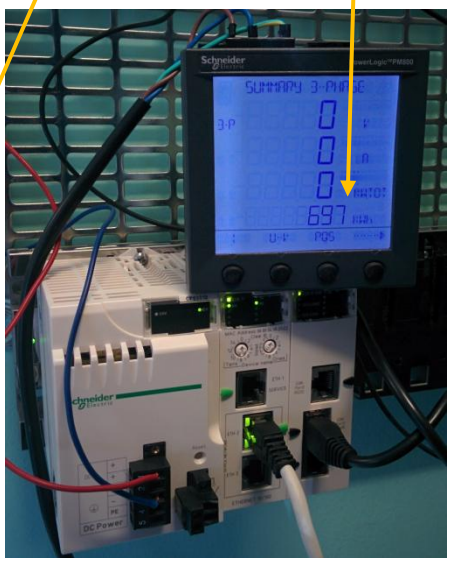
Connect devices according to the architecture shown in chapter 2.3 and make sure that the Modbus settings of PM800 are consistent with configuration defined in PES topology and application manager.

Run the controller and the supervision system.

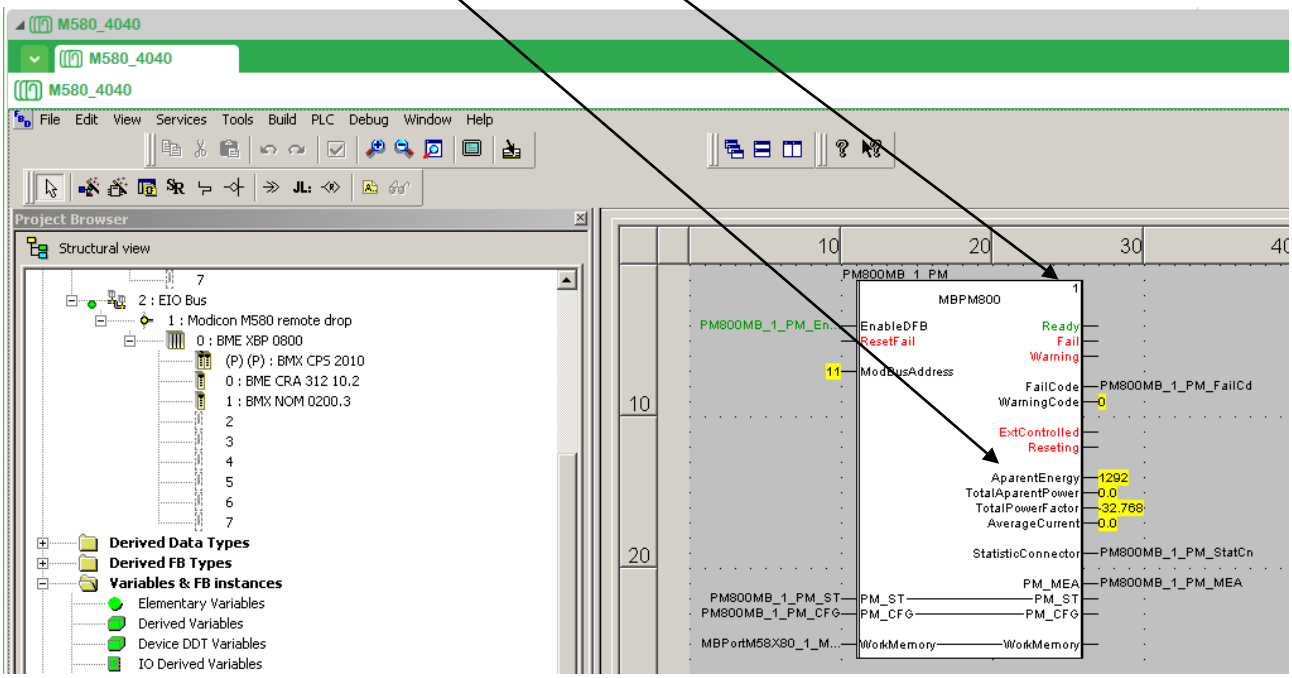
In the supervision runtime, PM is showing proper communication status and measurements according to current values.







In control participant, **PM800MB_1** instance is in "**Ready**" status, means that the communication is working properly, it displays also some measurements.



MBPortM58X80_1 DFB instance is managing requests properly (on `StatisticData` DDT output variable)