Software manual

ITKP-20-01-04-20-EN

PROFINET

Communication Protocol:
MWMH Module
MWLH Module
MWSH Module

radwag.com

Scan the QR code to view additional scientific material that may be of interest to you.
There you will find more useful information in an accessible format!

APRIL 2020

CONTENTS

1. MODULE SETTINGS AND CONFIGURATION 4

2. DATA STRUCTURE 4
2.1. Input Registers 4
2.2. Output Registers 6

3. CONFIGURATION OF PROFINET MODULE IN TIA PORTAL V14 ENVIRONMENT 9
3.1. Import GSD 9
3.2. Module Configuration 11

4. DIAGNOSTICS APP 16

1. MODULE SETTINGS AND CONFIGURATION

Settings of MWMH, MWLH, MWSH modules for communication via PROFINET protocol are configured using MWMH MANAGER software that is to be downloaded from www.radwag.com website. Run the MWMH MANAGER program, go to COMMUNICATION –> PERIPHERALS tab, click 'Profinet protocol' entry, next enter the IP address, the subnet mask and the name of a device from the Profinet network. For more information visit www.radwag.com and read user manual of the given device.

2. DATA STRUCTURE

2.1. Input Registers

Input variables:

Variable	Offset	Length [WORD]	Data type
Mass	0	2	float
Tare	4	2	float
Unit	8	1	word
Platform status	10	1	word
LO threshold	12	2	float
Input status	66	1	word
Min	68	2	float
Max	72	2	float
Fast dosing threshold	76	2	float
Slow dosing threshold	80	2	float
Adjustment status	100	1	word
Dosing process status	102	1	word

Platform mass – returns load mass in a current unit.

Platform tare – returns platform tare in an adjustment unit.

Platform unit – determines a current mass unit of the platform.

Register bit	
0	gram [g]
1	kilogram [kg]
2	pound [lb]
3	ounce [oz]
4	carat [ct]
5	Newton [N]

Example:

Read HEX value: 0x02. Binary form:

B1/7	B1/6	B1/5	B1/4	B1/3	B1/2	B1/1	B1/0	B0/7	B0/6	B0/5	B0/4	B0/3	B0/2	B0/1	B0/0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

The unit of the weighing instrument is kilogram [kg].

Platform status – determines state of a given weighing platform.

Register bit	
0	measurement correct (the weighing instrument does not report any error)
1	measurement stable
2	weighing instrument indicates zero
3	weighing instrument tared
4	weighing instrument in II weighing range
5	weighing instrument in III weighing range
6	weighing instrument reports NULL error
7	weighing instrument reports LH error
8	weighing instrument reports FULL error
9	time_triggered_adjustment_required / temperature_triggered_adjustment_required

Example:

Read HEX value: 0x13

B1/7	B1/6	B1/5	B1/4	B1/3	B1/2	B1/1	B1/0	B0/7	B0/6	B0/5	B0/4	B0/3	B0/2	B0/1	B0/0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1

The weighing instrument does not report any error, measurement stable in II weighing range.

LO threshold – returns value of platform's **LO** threshold in an adjustment unit.

Input state – bitmask of platform inputs.

Example:

Read HEX value: 0x02

B1/7	B1/6	B1/5	B1/4	B1/3	B1/2	B1/1	B1/0	B0/7	B0/6	B0/5	B0/4	B0/3	B0/2	B0/1	B0/0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Inputs 1 and 2 take HI state

MIN – returns **MIN** threshold value (in a current unit)

MAX – returns **MAX** threshold value (in a current unit).

Fast dosing threshold – returns fast dosing threshold value (rough dosing)

Slow dosing threshold – returns slow dosing threshold value (fine dosing)

Adjustment status

0x00 – adjustment completed correctly
0x01 – process active
0x02 – range exceeded
0x03 – time exceeded
0x04 – process aborted

Dosing process status

0x00 – process disabled
0x01 – taring in progress
0x02 – process activated
0x03 – process inhibited
0x05 – process completed

2.2. Output Registers

Output variables:

Variable	Offset	Length [WORD]	Data type
Command	0	1	word
Command with parameter	2	1	word
Tare	6	2	float
LO threshold	10	2	float
Output state	14	1	word
Min	16	2	float
Max	20	2	float
Fast dosing threshold	24	2	dword
Slow dosing threshold	28	1	word
Adjustment weight mass	48	1	word

Basic command – setting a respective value triggers direct performance of a given task, see the table:

Register bit	Command
0	Zero the platform
1	Tare the platform
5	Dosing start
6	Dosing stop
7	Internal adjustment start
8	Start mass determination
9	Adjustment factor determination
10	Adjustment parameters record (start mass/adjustment factor)

Example:

Record of the register by value 0x02

B1/7	B1/6	B1/5	B1/4	B1/3	B1/2	B1/1	B1/0	B0/7	B0/6	B0/5	B0/4	B0/3	B0/2	B0/1	B0/0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Causes platform taring.

A command is executed once upon detecting that its bit has been set. If the command is to be executed more than once, it is necessary to zero the bit first, and reset it to the required value next.

Complex command – setting a respective value triggers performance of a given task, see the table:

Register bit	Command
0	Setting tare value for a given platform
1	Setting LO threshold value for a given platform
2	Setting output status
3	Setting MIN threshold
4	Setting MAX threshold
5	Setting fast dosing threshold
6	Setting slow dosing threshold
7	Setting adjustment weight value

	***Complex command requires setting the address of a respective parameter (from 2 to 24 – refer to: "Complex command parameters").***
	***A complex command is executed once upon detecting that its bit has been set. If the command is to be executed more than once, it is necessary to zero the bit first, and reset it to the required value next***

Example:

Sending tare of 1.0 value to the scale

Performance of the command requires record of 2 registers:

1. Complex command - value 0x01 – i.e. tare setting.
2. Tare **–** value 1.0 (0x3F800000).

Tare – complex command parameter: tare value (in an adjustment unit).

LO threshold – complex command parameter: LO threshold value (in an adjustment unit).

Output state – complex command parameter: determination of platform output state.

Example:

Setting high state for output 1 of the platform.

Output mask:

B1/7	B1/6	B1/5	B1/4	B1/3	B1/2	B1/1	B1/0	B0/7	B0/6	B0/5	B0/4	B0/3	B0/2	B0/1	B0/0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1

Upon conversion to HEX the result is 0x01

Performance of the command requires record of 2 registers:

1. Complex command - value 0x02 – i.e. output state setting.
2. Output mask **–** value 0x01.

MIN – complex command parameter: MIN threshold value (in the unit set for the active working mode).

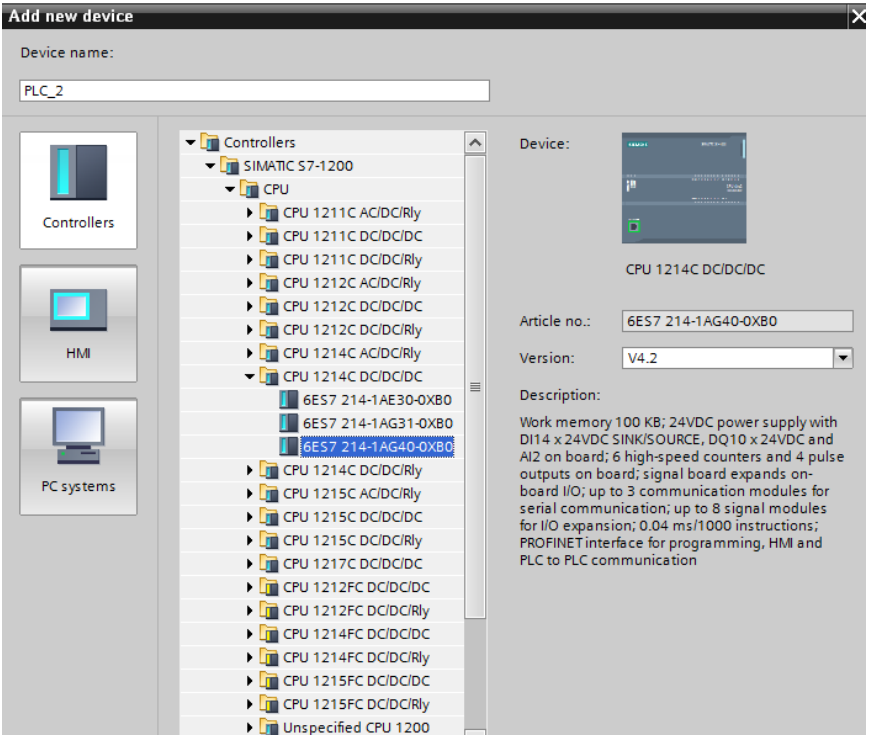
MAX – complex command parameter: MAX threshold value (in the unit set for the active working mode).

Fast dosing threshold – complex command parameter – value of fast dosing (rough dosing) threshold.

Slow dosing threshold – complex command parameter – value of slow dosing (fine dosing) threshold.

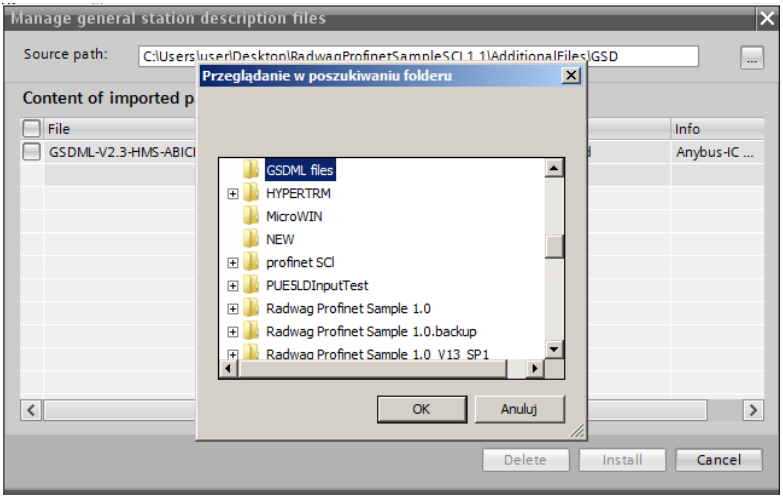
3. CONFIGURATION OF PROFINET MODULE IN TIA PORTAL V14 ENVIRONMENT

Start operation in the environment by setting up a new project, where topology of PROFINET network with MASTER controller will be specified, in this case the MASTER controller is SIEMENS controller of S7-1200 series.

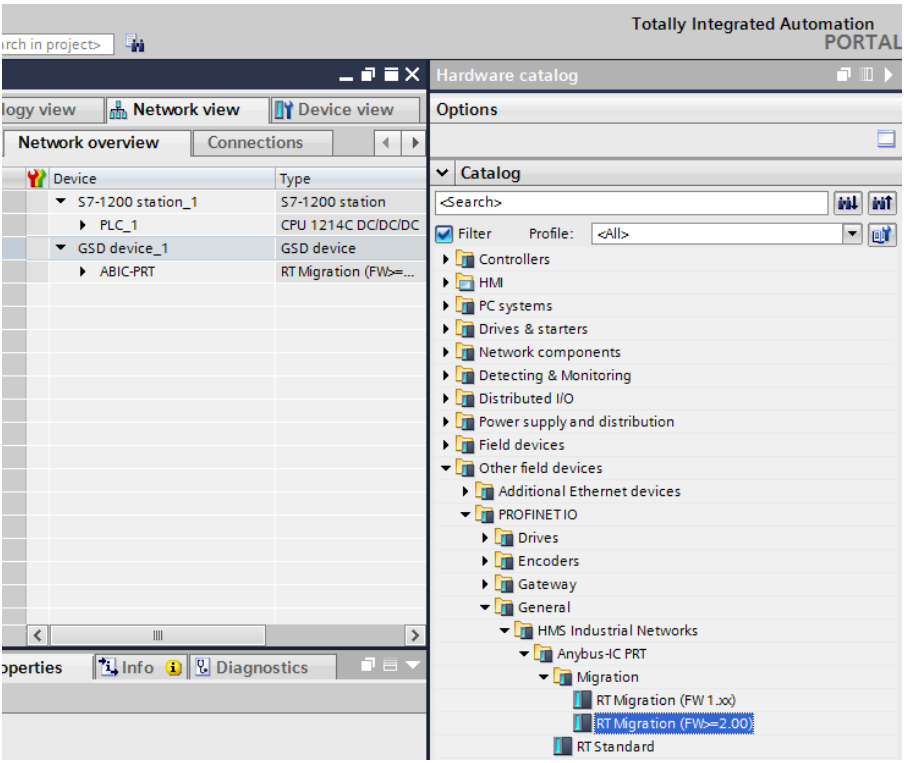


3.1. Import GSD

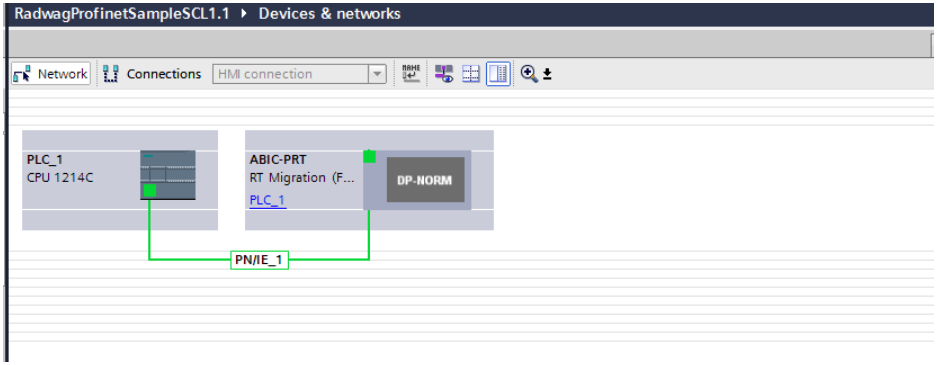
Using a delivered configuration file (GSD), add a new device to the environment. Open OPTIONS tab, next click MANAGE GENERAL STATION DESCRIPTION FILES (GSD) entry and select access path to the GSD file.



With the file successfully added, the ABIC-PRT module is displayed on the list of the devices.

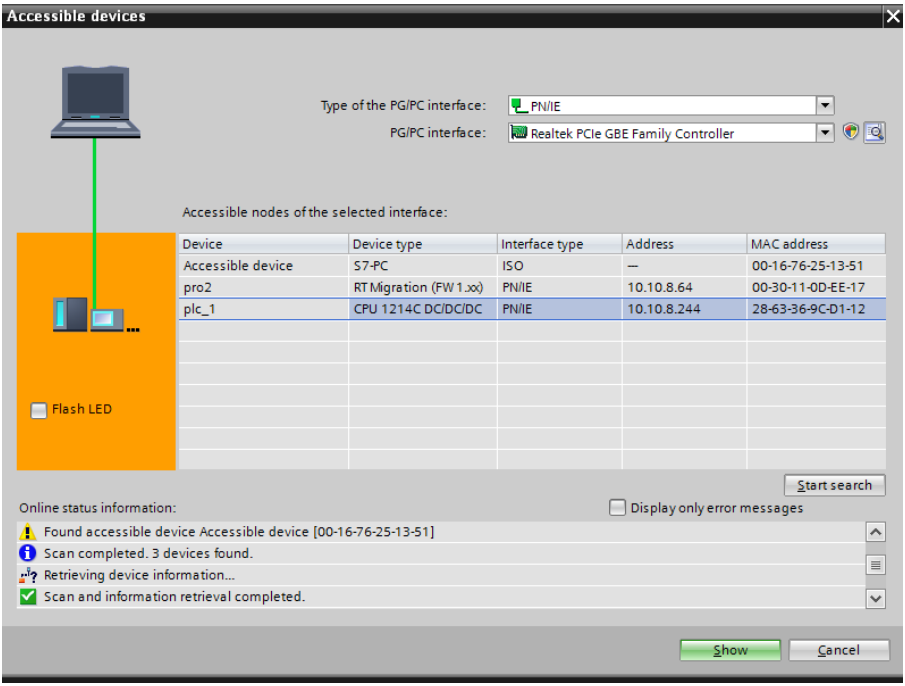


Now it is possible to make a network comprising a MASTER controller and a newly added SLAVE module.

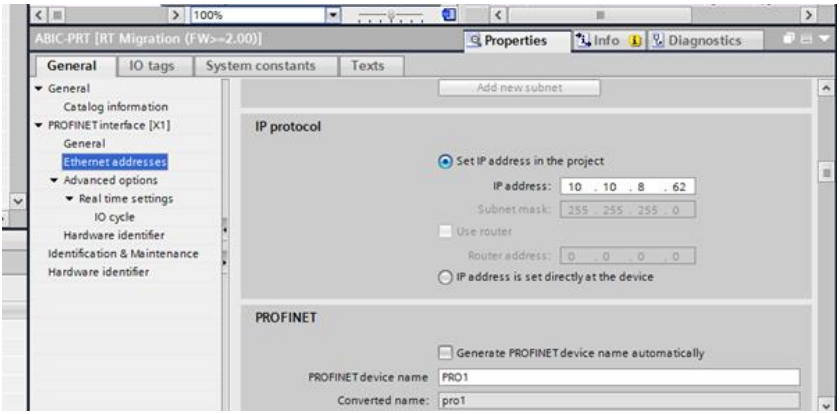


3.2. Module Configuration

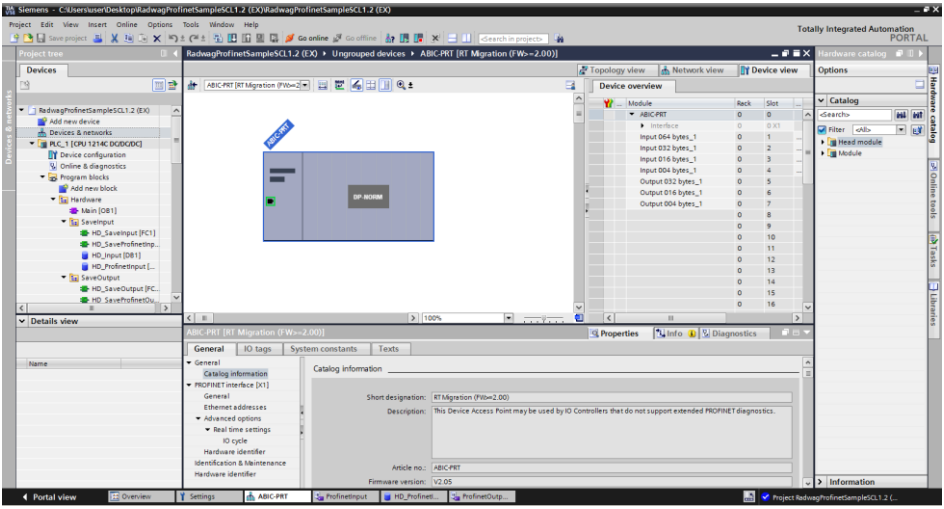
At this stage it is necessary to make network comprised of a MASTER controller and a SLAVE device (the scale). With the power supply on, you can use ACCESSIBLE DEVICES function to search for a given device in the environment. The list should display both the MASTER and the SLAVE.

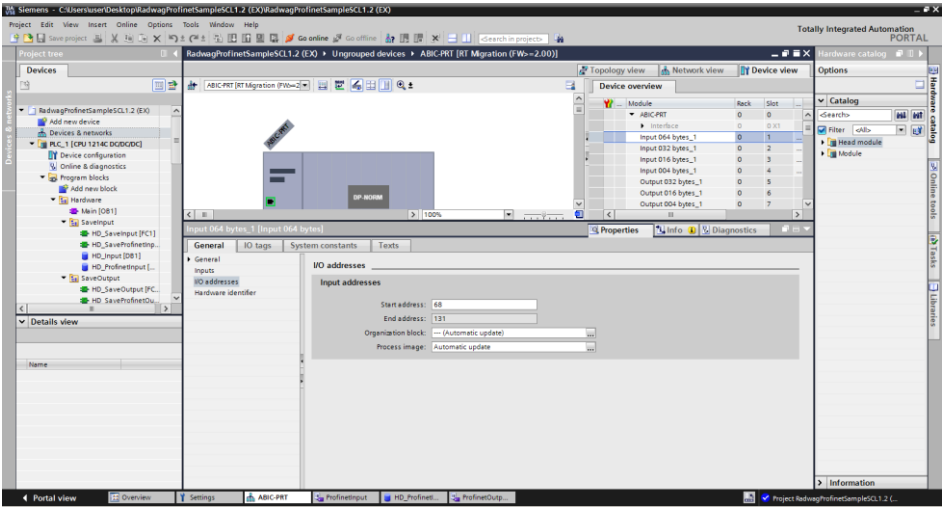


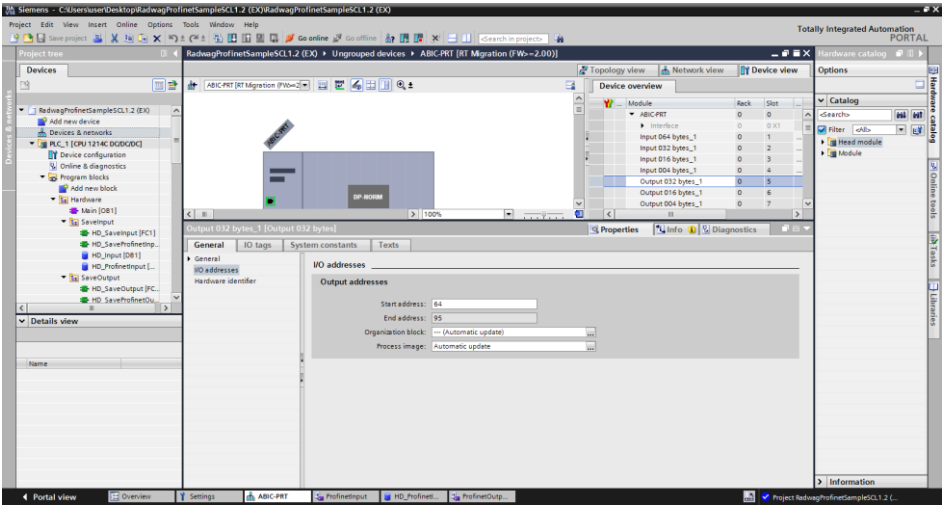
Now, specify the IP address and the name for the module in a PROFINET network. To do it, select the module, in PROPERTIES tab find PROFINET INTERFACE submenu, click it and enter the IP address and name. Make sure that these settings are accordant with parameters set in the scale menu. Remember that the SLAVE IP address and MASTER address must come from the same subnet.



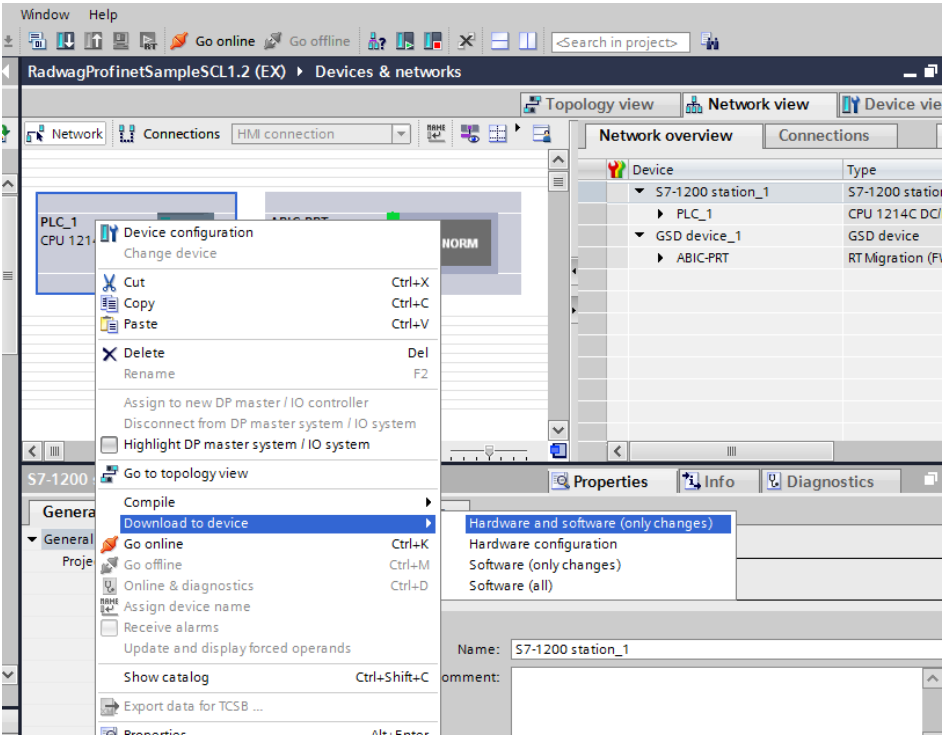
Next configure the module. First specify input and output registers size, define their start addresses. From the list of available INPUT and OUTPUT modules select such modules as presented in the picture below. Maximum size of input data is 110 bytes, and Maximum size of input data is 52 bytes. In the project, default start addresses have been used – 68 for the INPUT module and 64 for the OUTPUT module:



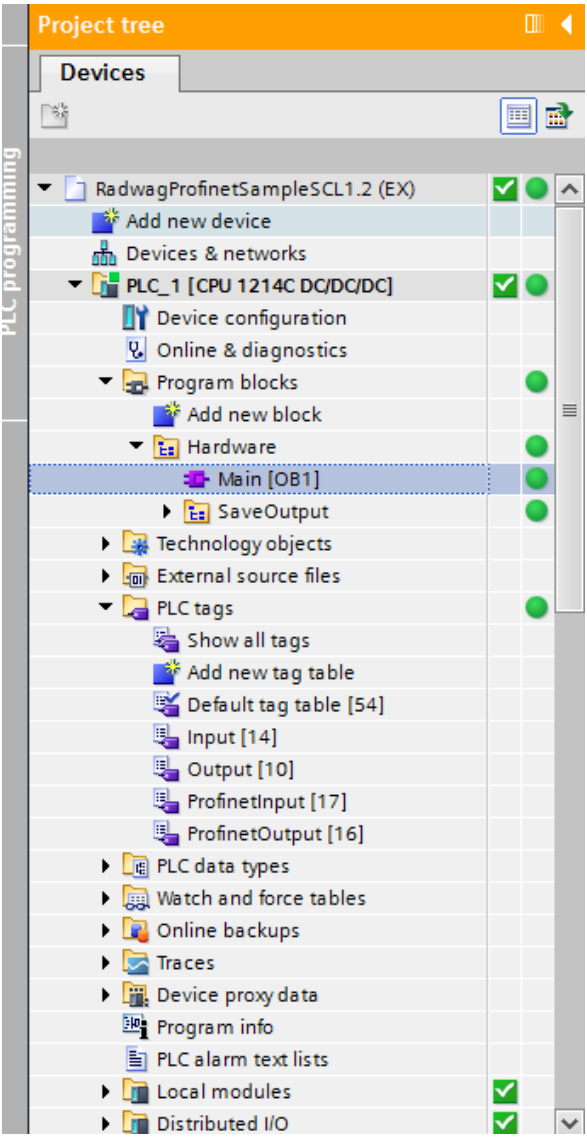

Totally Integrated Automation
PORTAL
RadwagProfinetSampleSCL1.2 (EX) ▸ Ungrouped devices ▸ ABIC-PRT [RT Migration (FW>=2.00)]
Input 064 bytes_1 [Input 064 bytes]
General
IO tags
System constants
Texts
I/O addresses
Input addresses
Start address: 68
End address: 131
Organization block: (Automatic update)
Process image: Automatic update


Totally Integrated Automation
PORTAL
RadwagProfinetSampleSCL1.2 (EX) ▸ Ungrouped devices ▸ ABIC-PRT [RT Migration (FW>=2.00)]
Output 032 bytes_1 [Output 032 bytes]
General
IO tags
System constants
Texts
I/O addresses
Output addresses
Start address: 64
End address: 95
Organization block: (Automatic update)
Process image: Automatic update

At this stage it is possible to upload the hardware configuration into the controller, and to start data upload.



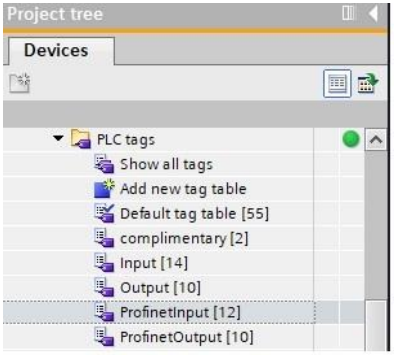
Upon successful compilation and code reading, the MASTER and SLAVE shall establish communication. To check it go online (select 'Go online' entry). The below screenshot presents what you shall see.



Now proceed to the process of program code making.

4. DIAGNOSTICS APP

Start creating the app by defining names of symbolic input and output registers. Use the PLC TAGS directory from the directory tree. For the purpose of this example, the following tables of tags have been created:



INPUT and OUTPUT tables refer to physical inputs/outputs of MASTER controller and are of no importance in case of this application. Input and output registers of PROFINET module have been specified in ProfinetInput and ProfinetOutput tables.

The below screenshots present symbolic names and addresses:

RadwagProfinetHRP1.0 ang ▸ PLC_1 [CPU 1214C DC/DC/DC] ▸ PLC tags ▸ ProfinetInput [12]

Tags | User constants

ProfinetInput

	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...	Monitor value	Comment
1	mass	Real	%ID68		☑	☑	☑	-125.3	
2	tare	Real	%ID72		☑	☑	☑	125.3	
3	unit	Word	%IW76		☑	☑	☑	16#0001	
4	status	Word	%IW78		☑	☑	☑	16#020F	
5	LO	Real	%ID80		☑	☑	☑	0.0	
6	inputs	Word	%IW134		☑	☑	☑	16#0000	
7	min	Real	%ID136		☑	☑	☑	0.0	
8	max	Real	%ID140		☑	☑	☑	0.0	
9	threshold_dose_coarse	Real	%ID144		☑	☑	☑	500.0	
10	threshold_dose_fine	Real	%ID148		☑	☑	☑	990.0	
11	calibr_stat	Word	%IW168		☑	☑	☑	16#0000	
12	dose_stat	Word	%IW170		☑	☑	☑	16#0000	
13	<Add new>				☑	☑	☑		

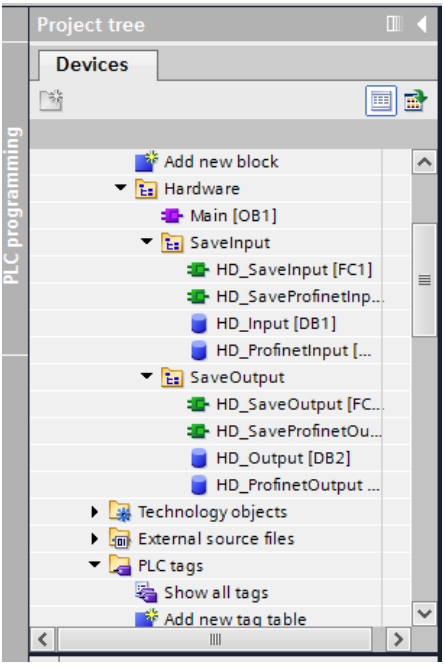
RadwagProfinetHRP1.0 ang ▸ PLC_1 [CPU 1214C DC/DC/DC] ▸ PLC tags ▸ ProfinetOutput [10]

Tags | User constants

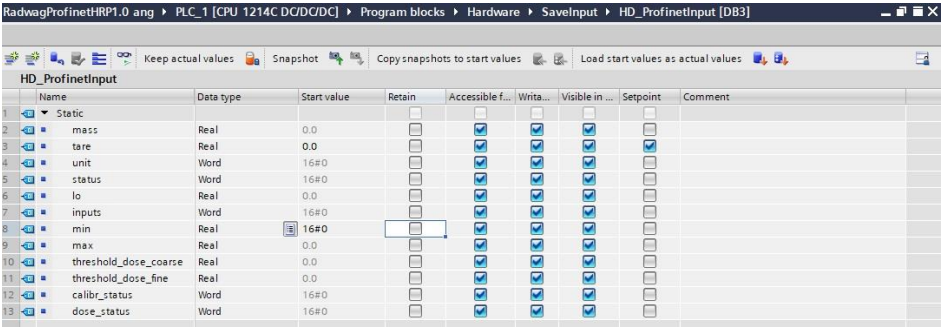
ProfinetOutput

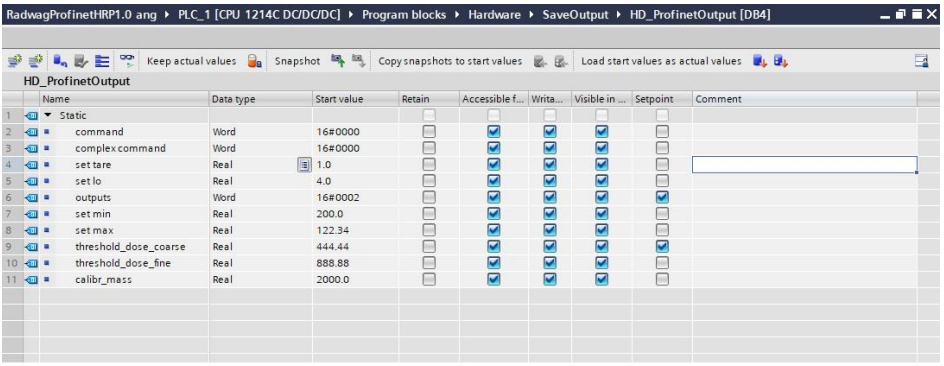
	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...	Monitor value	Comment
1	cammand	Word	%QW64		☑	☑	☑	16#0000	
2	complex command	Word	%QW66		☑	☑	☑	16#0000	
3	set tare	DWord	%QD70		☑	☑	☑	16#0000_0001	
4	set lo	Real	%QD74		☑	☑	☑	4.0	
5	outputs	Word	%QW78		☑	☑	☑	16#0002	
6	set min	Real	%QD80		☑	☑	☑	200.0	
7	set max	Real	%QD84		☑	☑	☑	122.34	
8	set threshold_dose_coarse	Real	%QD88		☑	☑	☑	444.44	
9	set threshold_dose_fine	Real	%QD92		☑	☑	☑	888.88	
10	set_calibr_mass	Real	%QD112		☑	☑	☑	2000.0	
11	<Add new>				☑	☑	☑		

So as not to work directly on the physical inputs/outputs of the module it is recommended to create data blocks comprising representations of these registers, and functions "copying" values between them. For this purpose HARDWARE group is created in the PROGRAM BLOCKS directory, the blocks are defined as below:

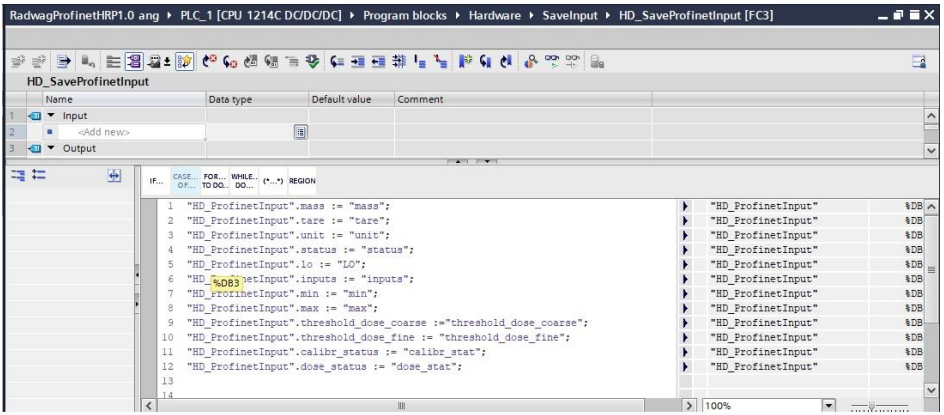


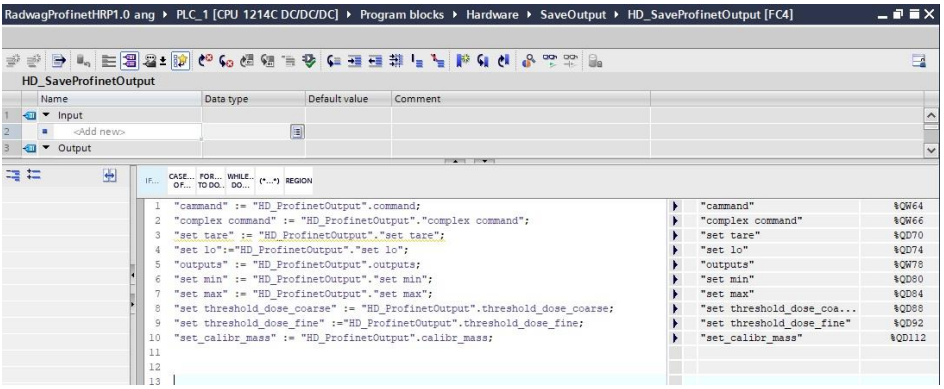
HD_OUTPUT and HD_INPUT blocks refer to physical MASTER's inputs/outputs and are of no importance in case of this project. HD_ProfinetOutput and HD_ProfinetInput blocks represent input/output registers of the scale's PROFINET module. See the screenshots below:



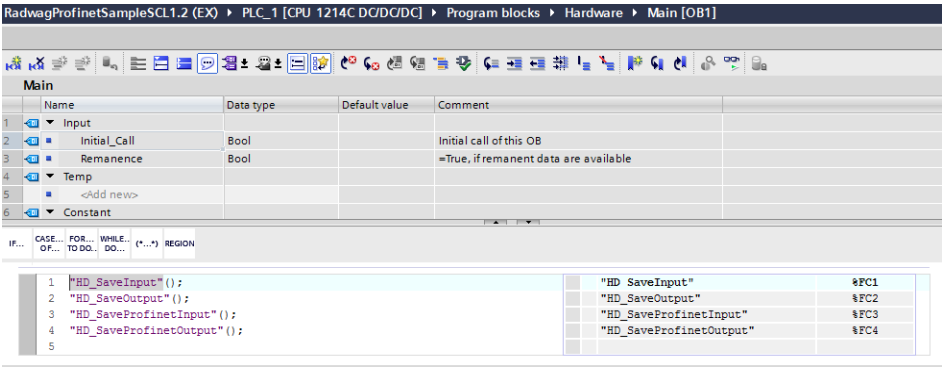


Functions copying values between the physical inputs/outputs of the module may look as presented below:





Now all that needs to be done is enforcing respective functions in the main loop of the program.



Upon compilation and upload of the program to the controller in data block it is possible to read input registers (MONITOR ALL) and to record output registers (e.g. by change of START VALUE and LOAD START VALUES AS ACTUAL) of the SLAVE module.

Example of record of the register by 0x01 value (platform zeroing):