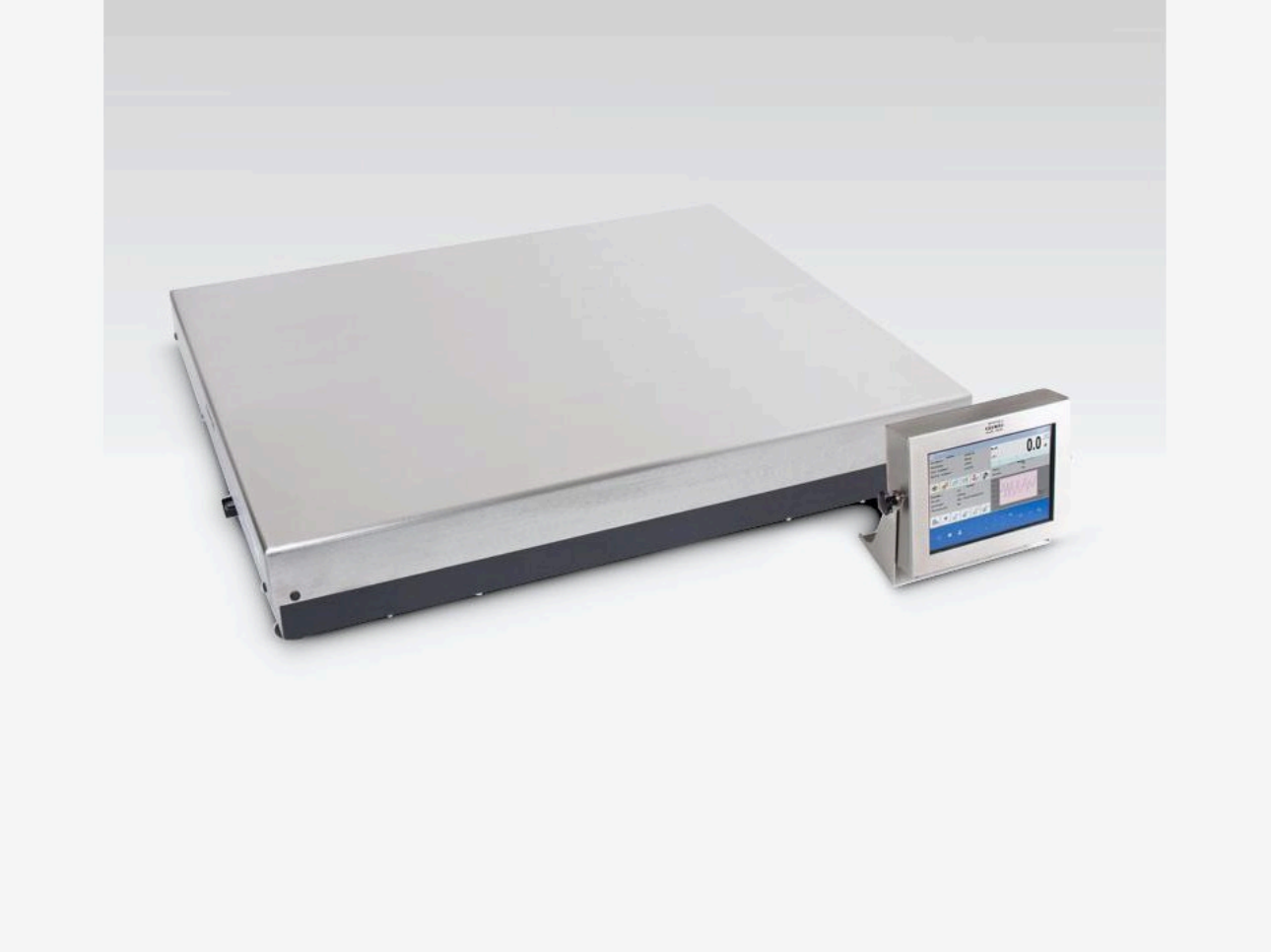
USER GUIDE

# HY10.300.HRP

Radwag

Distributed by AKRIBIS



# Software manual

**ITKP-27-01-07-20-EN**

# EtherNet/IP™

Communication Protocol of PUE HY10 Terminal

**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!

AKRIBIS www.akribis.info contacto@akribis.info

| | |
|---|---|
| ⚠ | **EtherNet/IP™ is a trademark of ODVA, Inc.** |

JULY 2020

# CONTENTS

**1. DATA STRUCTURE ..... 4**
1.1. Input Address ..... 4
1.1.1. Input Variables ..... 4
1.1.2. Input Registers ..... 5
1.2. Output Address ..... 7
1.2.1. Input Variables ..... 7
1.2.2. Output Registers ..... 7
**2. CONFIGURATION OF THE EtherNet/IP™ MODULE IN RS LOGIX ENVIRONMENT ..... 10**
2.1. RSLinxs configuration ..... 10
2.2. RSLogix Project ..... 11

# 1. DATA STRUCTURE

## 1.1. Input Address

### 1.1.1. Input Variables

| Variable | Offset | Length [WORD] | Data type |
|---|---|---|---|
| Platform 1 mass | 0 | 2 | float |
| Platform 1 tare | 4 | 2 | float |
| Platform 1 unit | 8 | 1 | integer |
| Platform 1 status | 10 | 1 | integer |
| Platform 1 LO threshold | 12 | 2 | float |
| Platform 2 mass | 16 | 2 | float |
| Platform 2 tare | 20 | 2 | float |
| Platform 2 unit | 24 | 1 | integer |
| Platform 2 status | 26 | 1 | integer |
| Platform 2 LO threshold | 28 | 2 | float |
| Platform 3 mass | 32 | 2 | float |
| Platform 3 tare | 36 | 2 | float |
| Platform 3 unit | 40 | 1 | integer |
| Platform 3 status | 42 | 1 | integer |
| Platform 3 LO threshold | 44 | 2 | float |
| Platform 4 mass | 48 | 2 | float |
| Platform 4 tare | 52 | 2 | float |
| Platform 4 unit | 56 | 1 | integer |
| Platform 4 status | 58 | 1 | integer |
| Platform 4 LO threshold | 60 | 2 | float |
| Process status (Stop, Start) | 64 | 1 | integer |
| Input state | 66 | 1 | integer |
| Min | 68 | 2 | float |
| Max | 72 | 2 | float |
| Serial number | 84 | 2 | long |
| Operator | 88 | 1 | integer |
| Product | 90 | 1 | integer |
| Customer | 92 | 1 | integer |
| Packaging | 94 | 1 | integer |
| Source warehouse | 96 | 1 | integer |
| Target warehouse | 98 | 1 | integer |
| Formulation/Dosing | 100 | 1 | integer |

### 1.1.2. Input Registers

**Platform mass** – returns platform mass in a current unit.

**Platform tare** – returns platform tare in an adjustment unit.

**Platform unit** – determines a current mass unit of a given platform.

| Unit bits | |
|---|---|
| **0** | gram [g] |
| **1** | kilogram [kg] |
| **2** | carat [ct] |
| **3** | pound [lb] |
| **4** | ounce [oz] |
| **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.

| Status bits | |
|---|---|
| **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 |

**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.

**Process status** – determines status of the dosing\formulation:

0x00 – process disabled
0x01 – process activated
0x02 – process aborted
0x03 – process completed

**Input state** – bitmask of indicator inputs. 4 first least significant bits represent weighing indicator inputs.

**Example:**

Read HEX value: 0x000B

| 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 | 1 | 0 | 1 | 1 |

Inputs number 1, 2 and 3 take HI state.

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

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

**Lot number** – returns lot number. Numerical values exclusively! Non-numerical values are neglected.

**Operator** – returns a logged in operator code.

**Product** – returns a selected product code.

**Customer** – returns a selected customer code.

**Packaging** – returns a selected packaging code.

**Source warehouse** – response: code of source warehouse.

**Target warehouse** – response: code of target warehouse.

**Formulation/Dosing process** – returns a selected formulation code.

**Dosing process** – returns the value of a selected dosing process code.

## 1.2. Output Address

### 1.2.1. Input Variables

| Variable | Offset | Length [WORD] | Data type |
|---|---|---|---|
| Command | 0 | 1 | integer |
| Command with parameter | 2 | 1 | integer |
| Platform | 4 | 1 | integer |
| Tare | 6 | 2 | float |
| LO threshold | 10 | 2 | float |
| Output state | 14 | 1 | integer |
| Min | 16 | 2 | float |
| Max | 20 | 2 | float |
| Serial number | 32 | 2 | long |
| Operator | 36 | 1 | integer |
| Product | 38 | 1 | integer |
| Customer | 40 | 1 | integer |
| Packaging | 42 | 1 | integer |
| Source warehouse | 44 | 1 | integer |
| Target warehouse | 46 | 1 | integer |
| Formulation/Dosing | 48 | 1 | integer |

### 1.2.2. Output Registers

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

| Bit No. | Operation |
|---|---|
| 0 | Zero the platform |
| 1 | Tare the platform |
| 2 | Clear statistics |
| 3 | Save/Print |
| 4 | Process start |
| 5 | Process stop |

**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 |

The above causes scale taring.

| | |
|---|---|
| ⚠ | ***Basic 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:

| Bit No. | Operation |
|---|---|
| 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 value |
| 4 | Setting MAX threshold value |

| | |
|---|---|
| ⚠ | ***Complex command requires setting a respective parameter (offset from 4 to 50 – refer to output registers table)*** |
| ⚠ | ***A command with a parameter 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 for platform 1 to the scale.

Performance of the command requires record of 3 registers:

offset 2 – command with a parameter - value 0x01 – i.e. tare setting,
offset 4 – number of a platform to which tare is to be assigned - value 0x01 for platform 1,
offset 6 – tare value in float format - 1.0.

**Platform** – complex command parameter: platform number (1 or 2).

**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: status of weighing indicator and communication module outputs.

**Example:**

Setting high state for output 1 and 3 of the indicator.

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 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

After conversion to HEX it is 0x05.

Performance of the command requires record of 2 registers:
offset 2 – command with parameter - value 0x08 – i.e. output state record,
offset 14 – output mask 0x05.

This results with HI state of outputs number 1 and 3.

**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).

**Lot number** – complex command parameter: serial number value. Numerical values exclusively! Non-numerical values are neglected.

**Operator** – complex command parameter: operator code (digits only).

**Product** – complex command parameter: product code (digits only).

**Customer** – complex command parameter: customer code (digits only).

**Packaging** – complex command parameter: packaging code (digits only)

**Source warehouse** – complex command parameter: Source warehouse code (numerical only).

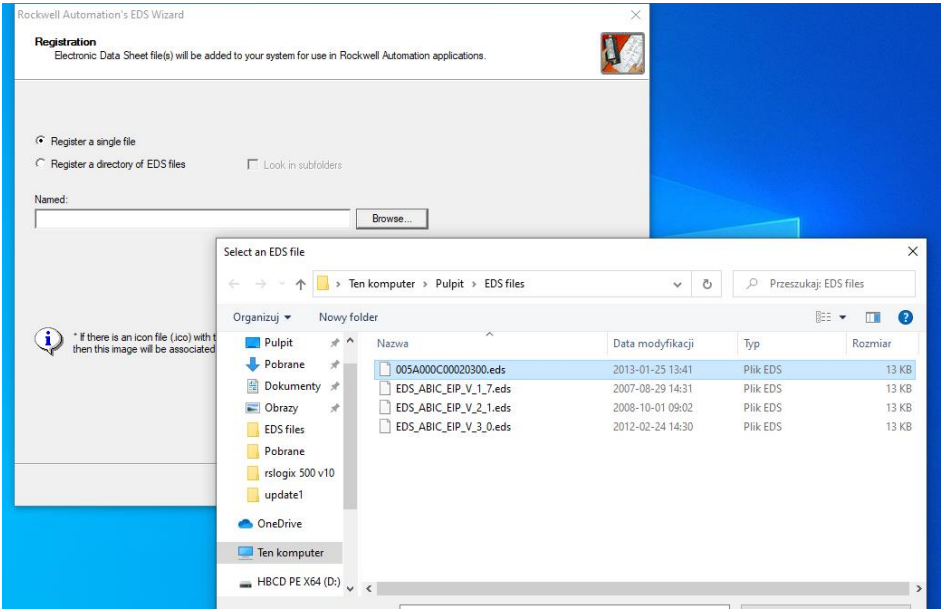
**Target warehouse** – complex command parameter: Target warehouse code (numerical only).

**Formulation/dosing process** – complex command parameter: formulation code (digits only).

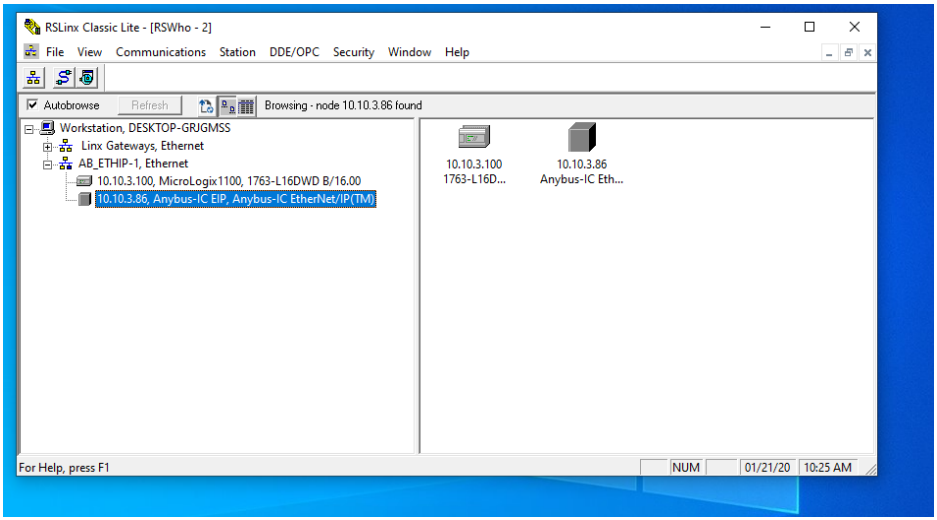
# 2. CONFIGURATION OF THE EtherNet/IP™ MODULE IN RS LOGIX ENVIRONMENT

## 2.1. RSLinxs configuration

Start the operation in the environment by configuring the devices in RSLinx software. To do this, add EtherNet/IP module of the scale using EDS file and EDS Hardware Installation Tool.

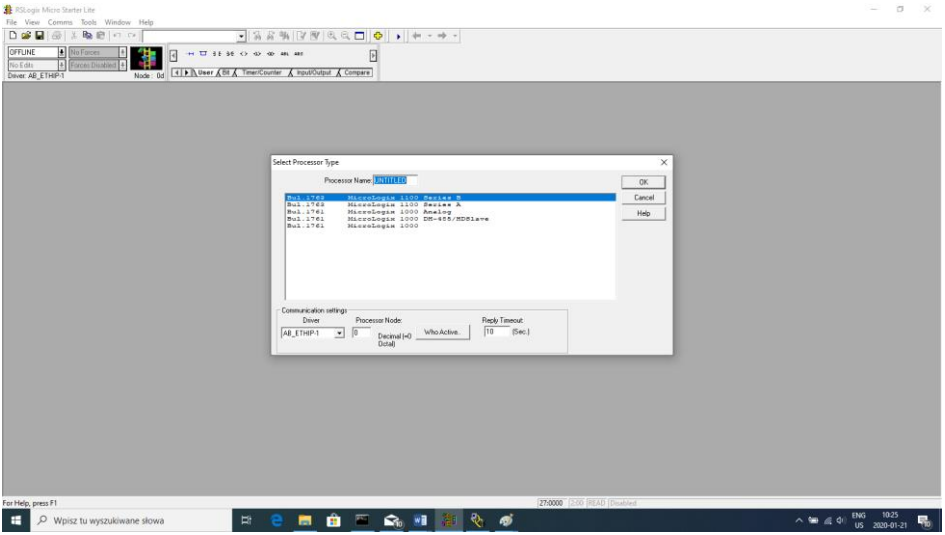



Upon connecting the scale and the Master controller to the network (make sure all devices and the PC are in the same subnet), they are visible as shown in the figure below.



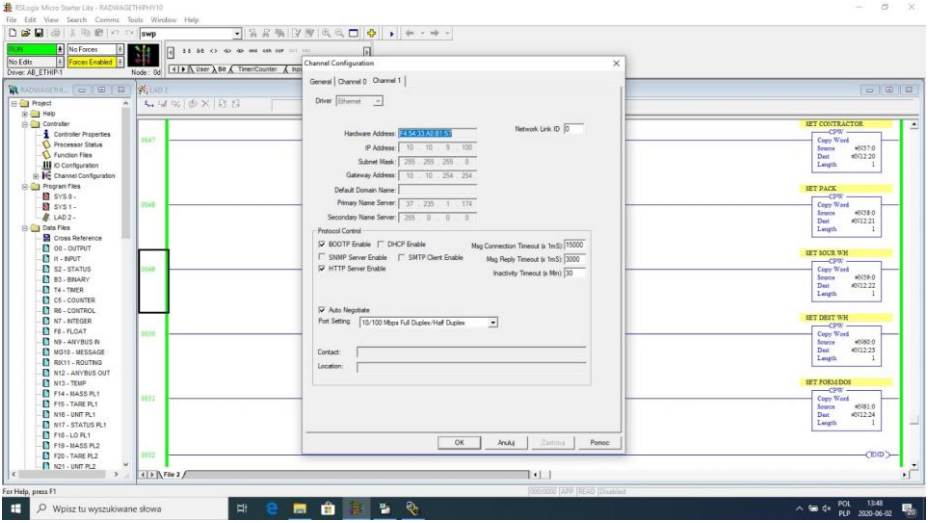

## 2.2. RSLogix Project

Start operation in the environment, to do it create a new project. In the controller window select the PLC that is to communicate with the scale.



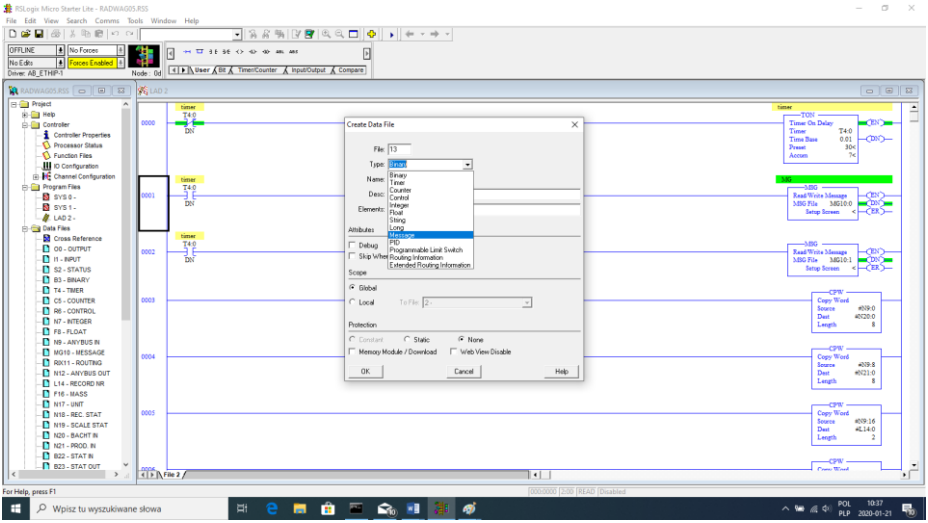

Confirm your choice and go to the project window. Next, configure the communication interface of the controller. To do that, select CHANNEL CONFIGURATION>CHANNEL 1 in the project tree.

Here, you can declare the properties of this communication channel, e.g. IP address or subnet mask.

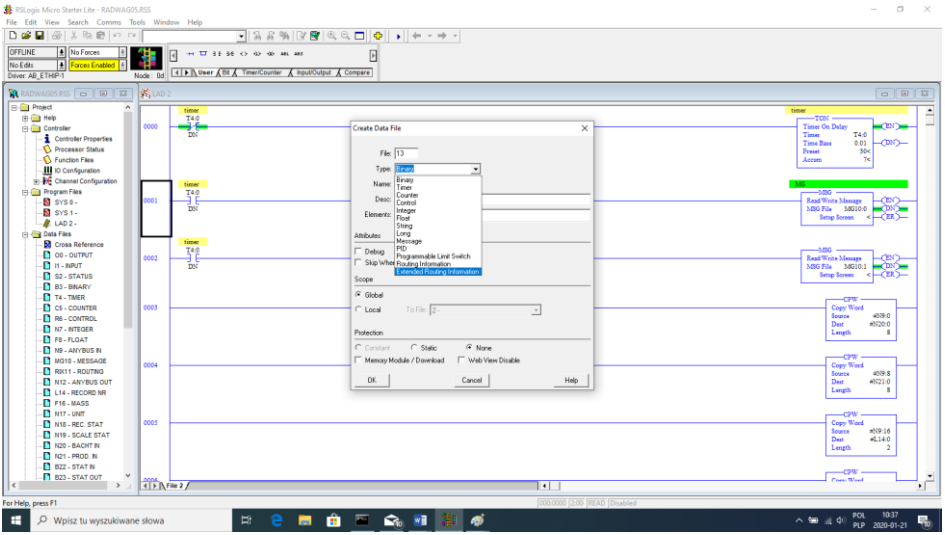


Upon configuration, check if connection with the PLC (online) is possible and download the project.

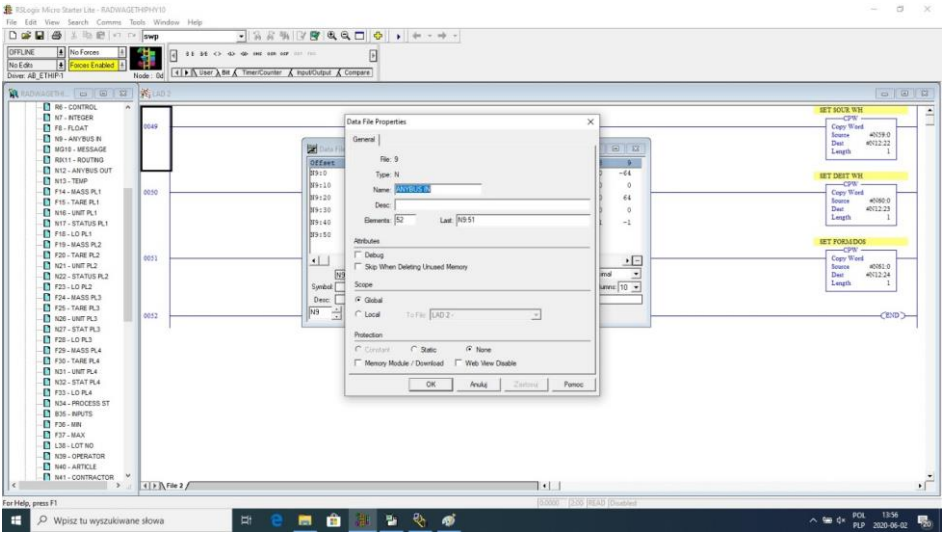
Now add a new rung to the project ladder and create a MSG function enabling readout of data from the scale. Prior to adding the function, add new data files in the project tree: two-element MG (message)

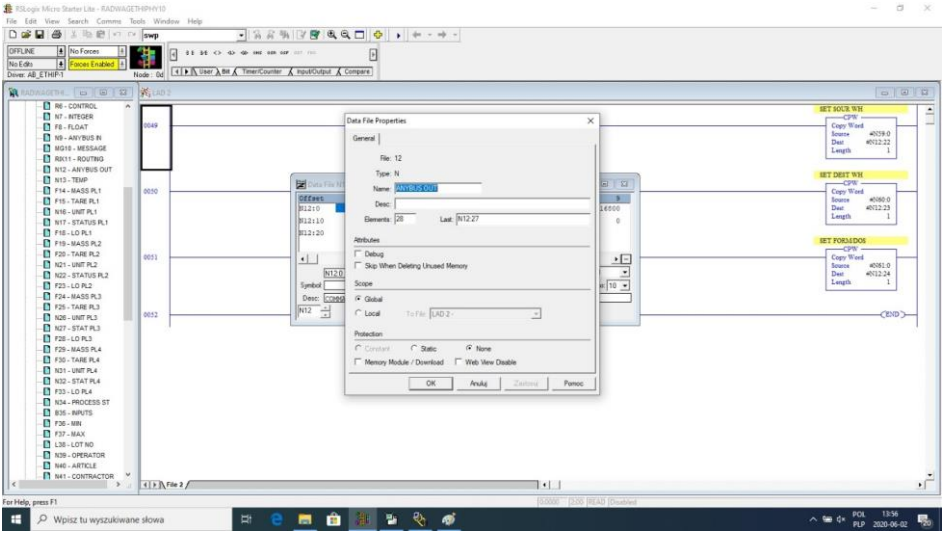


and RIX type files.

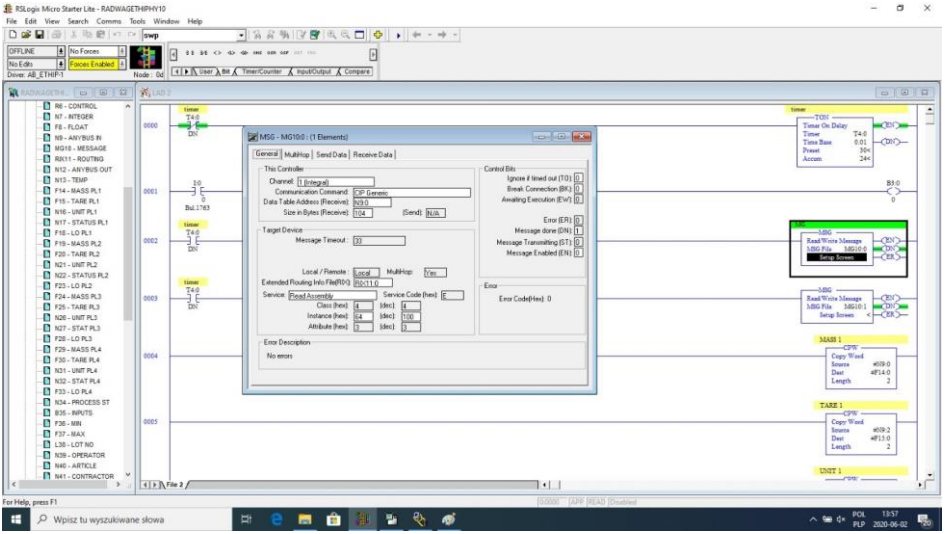



Add also two files of INTEGER type to store data read from the scale and data sent to the scale. In the example two files were created: ANYBUS IN (N9), 104 bytes, and ANYBUS OUT (N12), 56 bytes.






You can now add MSG functions, one for data readout and one for data record.

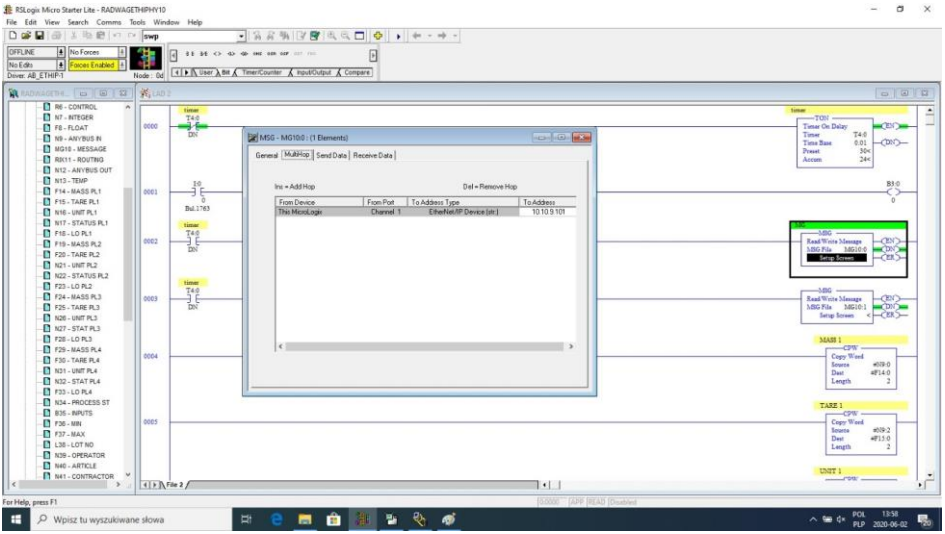



Configuration procedure:

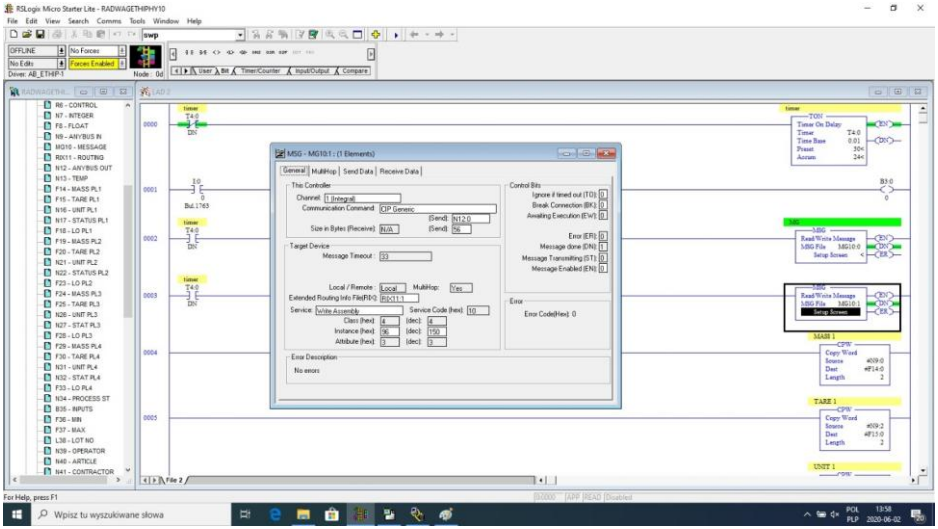
- Channel – select 1 (integral), which corresponds to EtherNet/IP
- Communication Command – CIP Generic.
- Data Table Address – N9:0 – the file for data readout.
- Size in Bytes – 104 – size of the input registers table

- Extended Routing Info File – RIX11:0 – indicate RIX file
- Service: Read assembly.
- Instance: 64.
- MulitHop: Yes.

Go to the MultiHoop tab and enter the IP address of the scale.



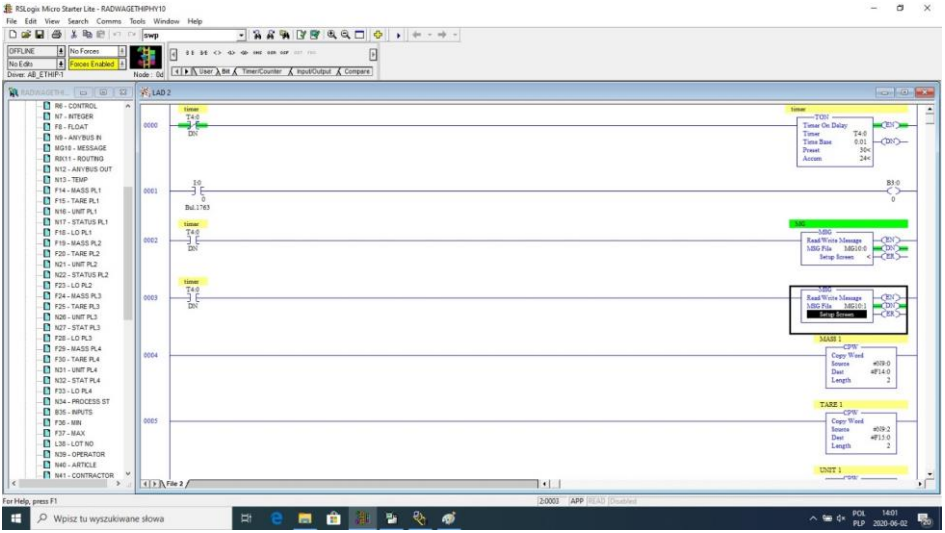

Create functions for record of data in the scale (analogous set of actions):



- Channel – select 1 (integral), which corresponds to EtherNet/IP™.
- Communication Command – CIP Generic.
- Data Table Address – N24:0 – the file for data record.
- Size in Bytes – 56 – size of the output registers table.
- Extended Routing Info File – RIX11:1 – set RIX file.
- Service: Read assembly.
- Instance: 96.
- MulitHop: Yes.

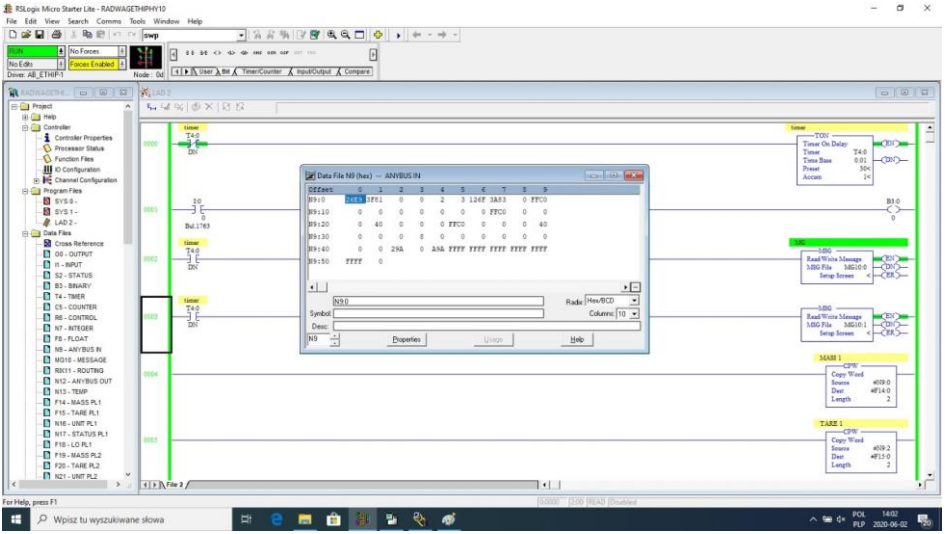
Go to the MultiHoop tab and enter the IP address of the scale.

In the example the functions are timer-triggered, this allows to control frequency of questions sent to the scale.



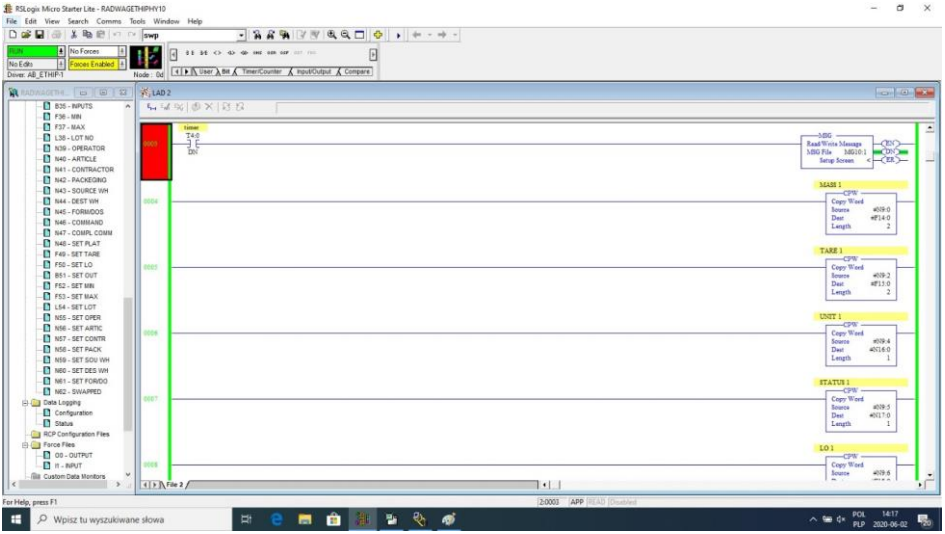
On this stage you can upload program to the controller and start it.

Upon connecting to the PLC (online) in the N9 file it is possible to carry out data readout, the MSG function should not return any errors.



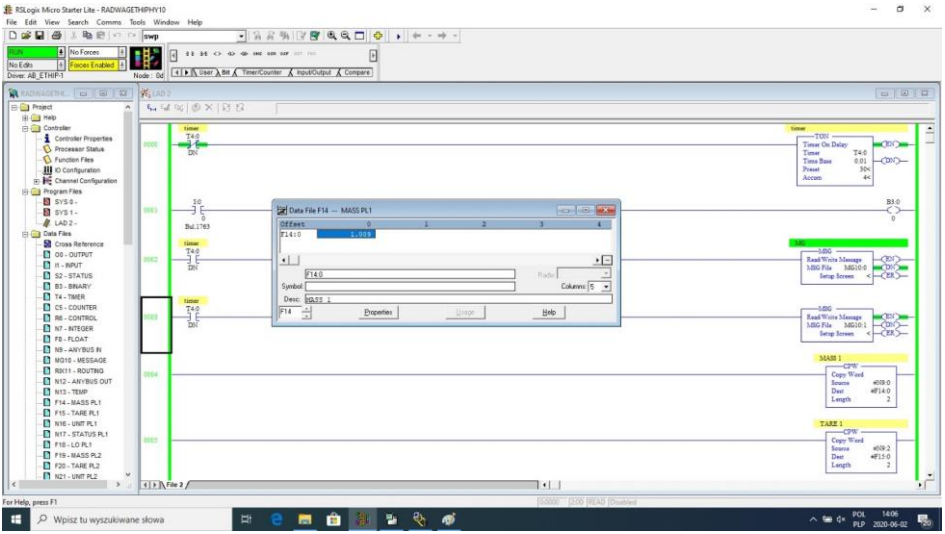

For the sake of order, you can create separate files for each scale variable.

Data between N9, N24 and variable files are written using CPW function. Mass readout function:



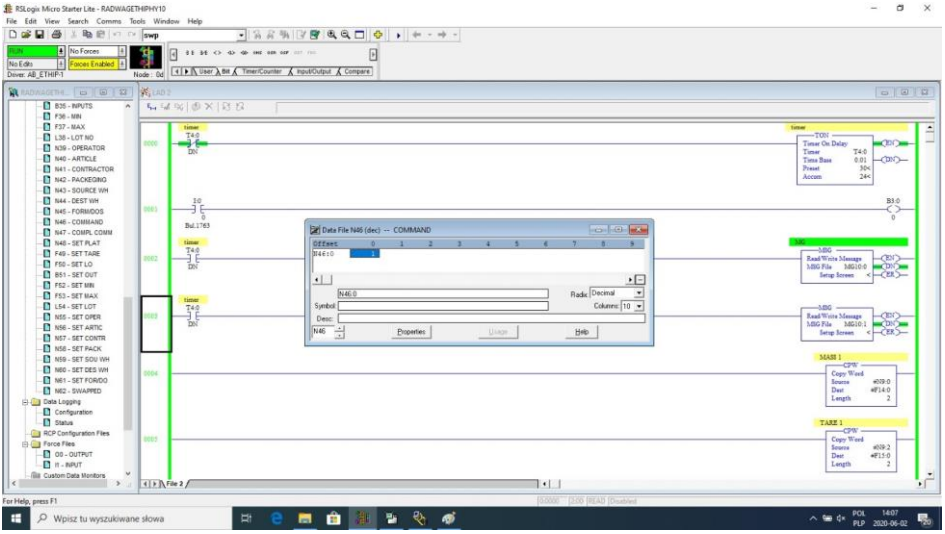

As a result, respective files contain correct data from the scale.

Mass readout example:




By record of respective values in files that correspond to output registers, particular scale functions are triggered.

Scale zeroing example:



AKRIBIS www.akribis.info contacto@akribis.info