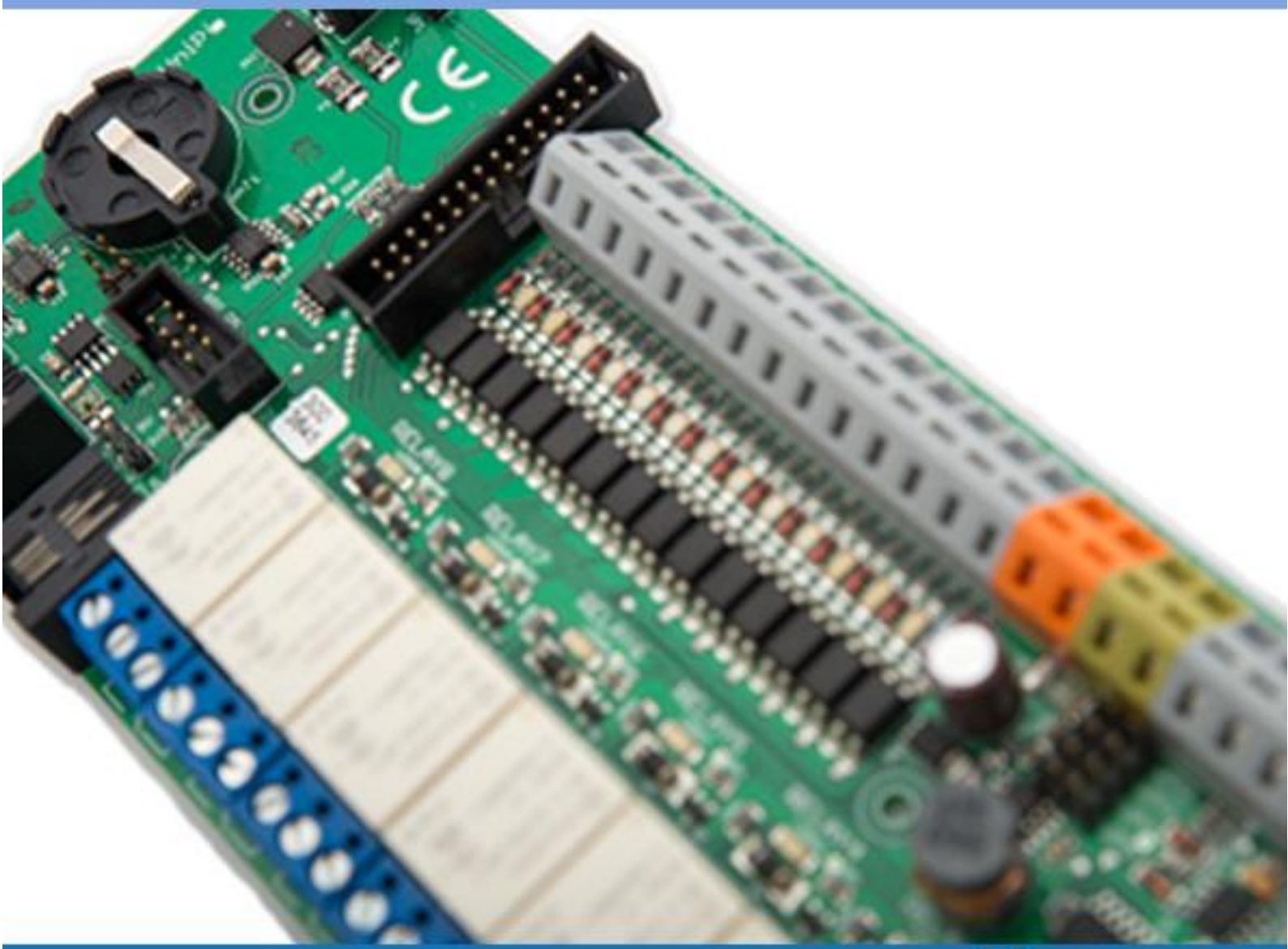
Mervis Step-by-Step Manual



UniPi .technology

Content

1 Installation

1.1 Mervis IDE

The first step is to download the installation package from http://downloads.unipi.technology

Run the installation package and make sure to include the following components:

- IDE – developing environment for control programs
- HMI –developing environment for human machine interfaces
- WindowsRT – tool for running RunTime under Windows (we will not be using that, so the installation of this package is optional)

IDE requires Microsoft .NET framework version 4+ to be installed. If the framework is not installed in your computer, download the installer from the following link:

http://www.microsoft.com/en-us/download/details.aspx?id=17851

1.2 Card

1.2.1 MicroSD card preparation

At least 4GB Micro SD card will be needed (An adapter might be necessary to connect the Micro SD card to your PC).

The SD card has to be formatted:

- The file system needs to be set to FAT32 and the allocation unit can be left to the default value
- It's recommended to use the SD Formatter, which is designed for formatting various SD cards.
- It can be downloaded from: https://www.sdcard.org/downloads/formatter_4/
- In the link, we can find a simple user manual

ATTENTION! You need to be extra careful which drive you select. Choosing wrong drive can cause losing all your data.

1.2.2 Deploying image

Prerequisites:

- Mervis OS image – http://downloads.unipi.technology/
- Win32DiskImager - http://sourceforge.net/projects/win32diskimager/
- WinRar, WinZip,…

Download the compressed image file and extract it using your favorite program (WinRar, WinZip,…)

Open the Win32DiskImager

- Set the Device representing your Micro SD card
- Image File needs to contain the path to the unpacked file
- Press the Write button. The image will be written onto the MicroSD card

Once the process is completed, close the utility, eject the microSD card, insert it into your UniPi and power it on.

1.3 Connecting UniPi

In order to be able to program the UniPi, you need to connect it with your PC. The connection should be done using an Ethernet connection. This can be done either using a router that will assign the UniPi an IP address or directly without a router.

1.3.1 Router mode

This scenario expect presence of a DHCP server (basically a router) in your network which will assign an IP address to the UniPi. This process is done automatically. But make sure that you are connected to the same physical network as your UniPi. If not, please contact your IT administrator.

1.3.2 Direct mode

If there is no DHCP server on the network (e.g. when the UniPi is connected directly to your PC using an Ethernet cable) the UniPi will assign itself an IP address from the same range as Microsoft Windows, e.g. 169.254.xxx.xxx. This supposes that your Windows is set to use DHCP as well, if not please make sure it is so.

2 Basic IDE

2.1 Simple vs Full mode

2.1.1 Simple mode

Simple Mode is designed to help beginners with the Mervis IDE enviroment. It automatically generates the “backbone” of the project. So, all the user needs to do, is to make some basic configuration.

If a device is added to a channel in simple mode (will be described later), autogen automatically generates and maps all the variables to device’s inputs and outputs.

2.1.2 Full mode

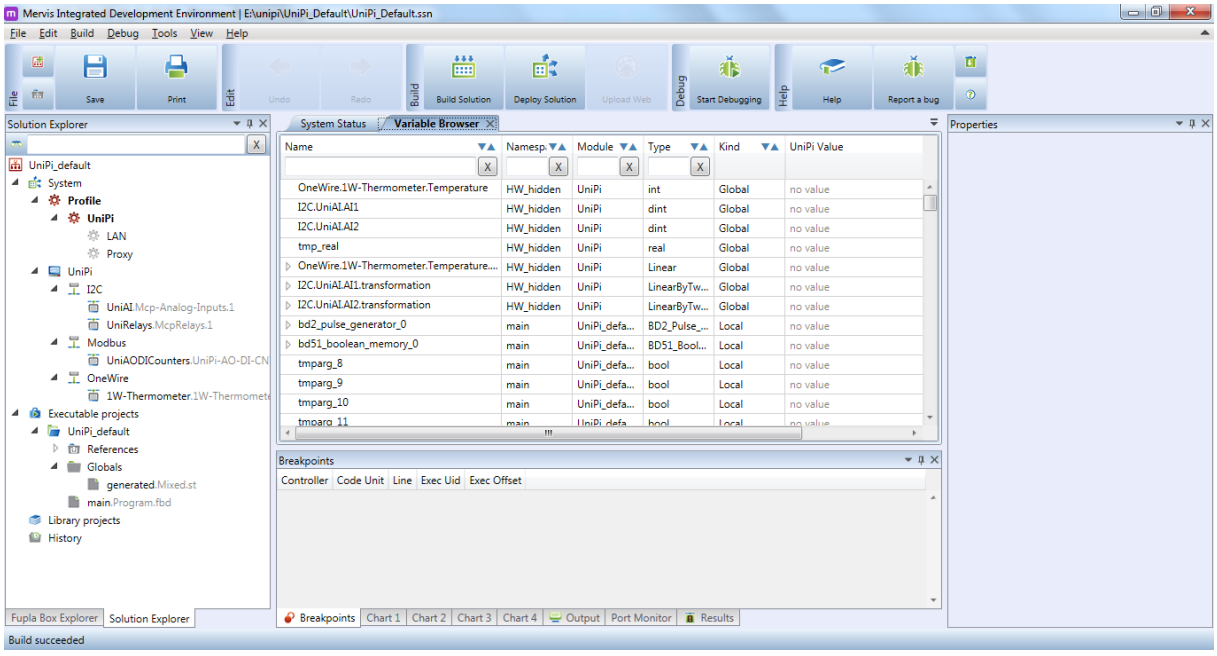
Full Mode is designed for advanced users who already know what needs to be done and are capable of creating the whole project from “scratch”, which includes adding Profiles, Devices, Executable and Library projects, History, etc.

Full Mode will be explained more extensively in Advanced IDE chapter.

2.2 New project

- Launch the Mervis IDE
- Create a new solution: File -> New Solution
- In the new window choose your Project Name and a folder where the project files will be saved
- Select your working mode – simple / full
- This scenario will be using Simple mode

2.3 Mervis IDE environment



See the upper part of the screen with buttons for file management, project build and deploy, debugging tool, help (very useful), etc.

Note the left side of the screen with Solution Explorer where the structure of whole project is shown. Structure consists of:

- Profiles
- PLCs(+ its channels and devices)
- Executable projects
- Histories (available only in full mode)

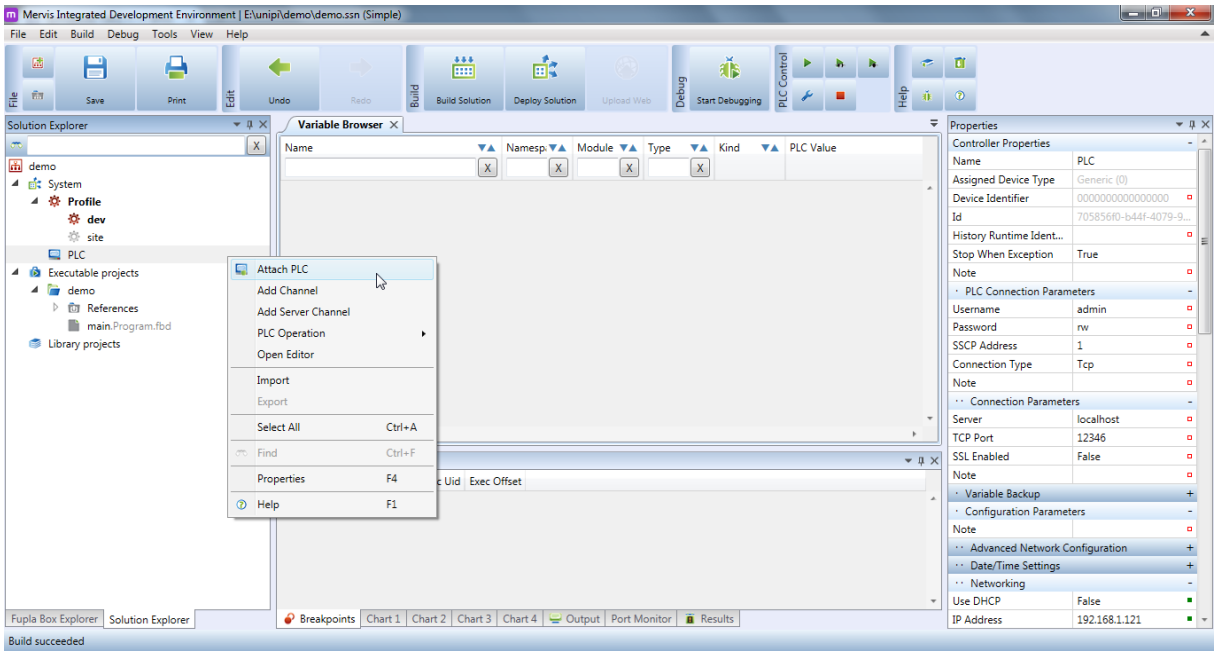
In the central area is by default used for programming behavior of your application (either FUPLA or ST), debugging, variable management, etc.

On the right side, there is a window called Properties, where properties of PLC, channels, devices, function blocks, etc., are set

2.4 Simple project walkthrough

2.4.1 Attaching PLC

In order to attach UniPi to a new or already existing solution, we must first right click on PLC in Solution Explorer and choose Attach PLC.



New window will appear with possible options of detection of the PLC - choose UDP broadcast.

Once done, you should see a list of all detected PLCs(UniPi with Mervis) in your local network. Select the one you wish to attach to this project.

- By default, the UniPi tries to request IP address from a DHCP server in your LAN
- If it fails requesting IP from DHCP, it will automatically assign a random address from the same network as Windows default IP address (e.g. 169.254.xxx.xxx)
- Make sure to have IP address from the same network as the UniPi

A new window will pop asking whether to download the configuration from the PLC or not. This is useful when attaching already configured PLC (network settings, etc.).

The UniPi is now attached to your solution.

2.4.2 Uploading/Updating RunTime

If you notice a yellow or red triangle next to Runtime Version number while selecting our UniPi from the PLC detection dialog, it means that runtime (RT) is outdated or does not match with the version of the Mervis IDE you are using. In that case, we need to update the runtime to compatible RT:

- Right click on the PLC icon -> PLC Operation -> Upload RT
- When the upload is finished, right click on the PLC icon -> PLC Operation -> Reboot PLC (explained in the next chapter)

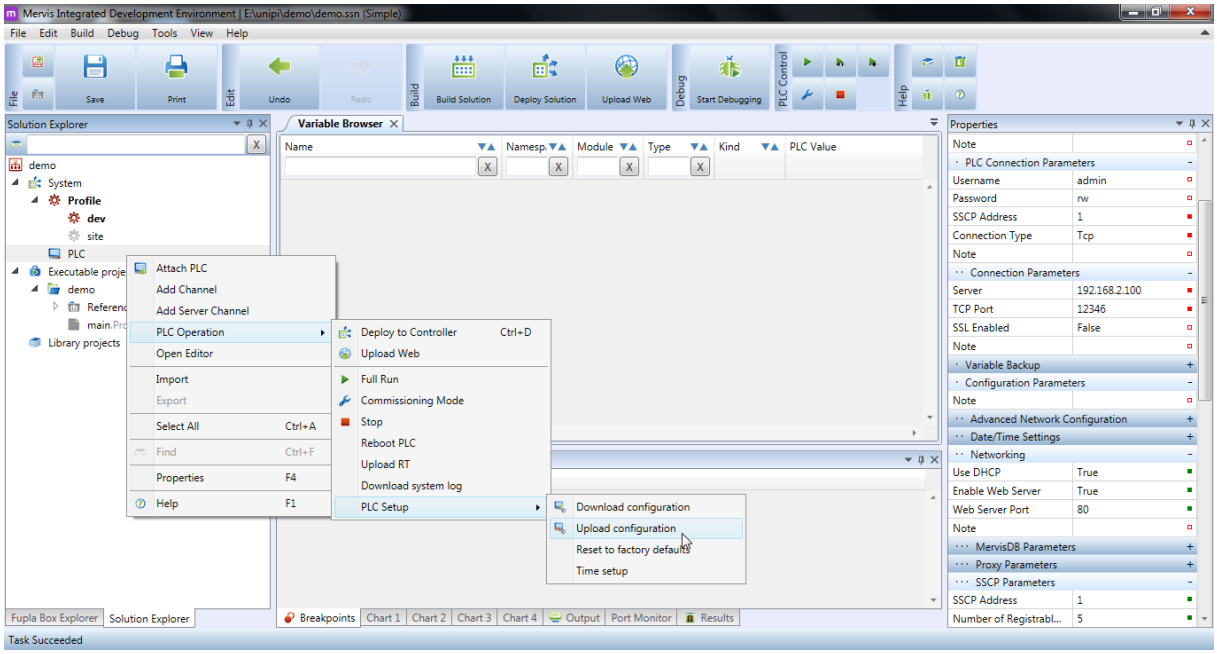
This step is also necessary when deploying project fails due to an old version of RT.

2.4.3 Controller properities

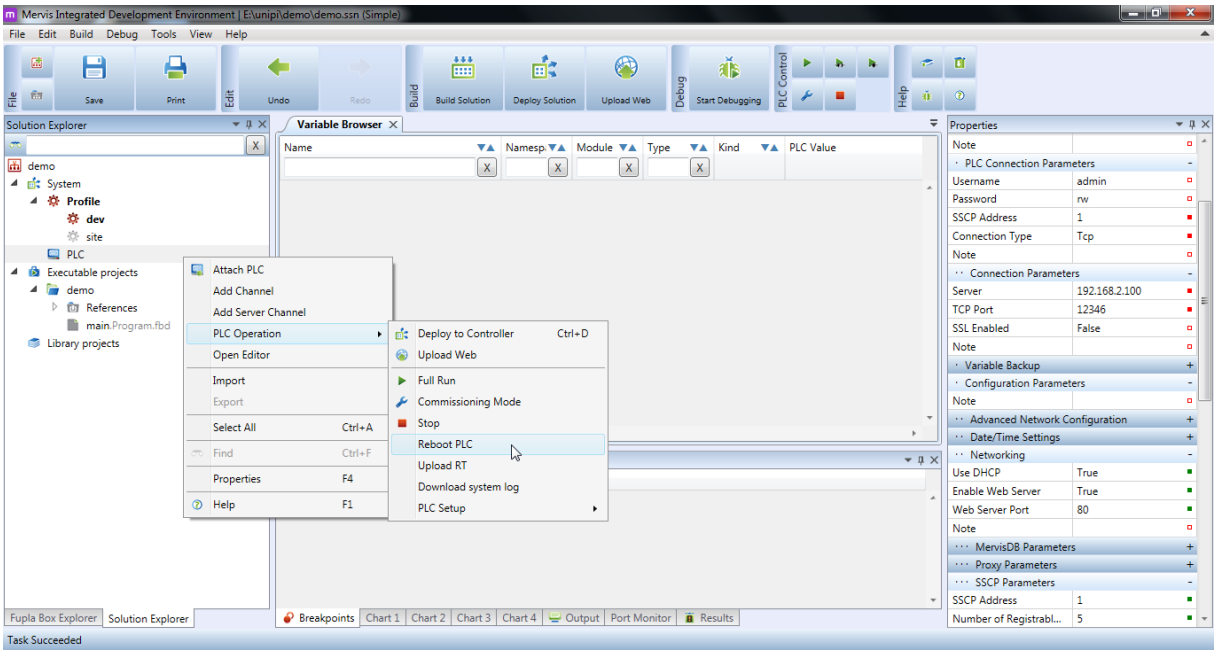
This panel is used to configure specific parameters of the UniPi (e.g. IP address, date and time or a NTP server, name of the PLC, etc.)

After any change in this section, it is required to upload the settings to the PLC which consists of two steps:

- Right click on the PLC icon (in solution explorer) and choose: PLC Operation -> PLC Setup -> Upload configuration. After a while the progress bar should display “Upload successful”.



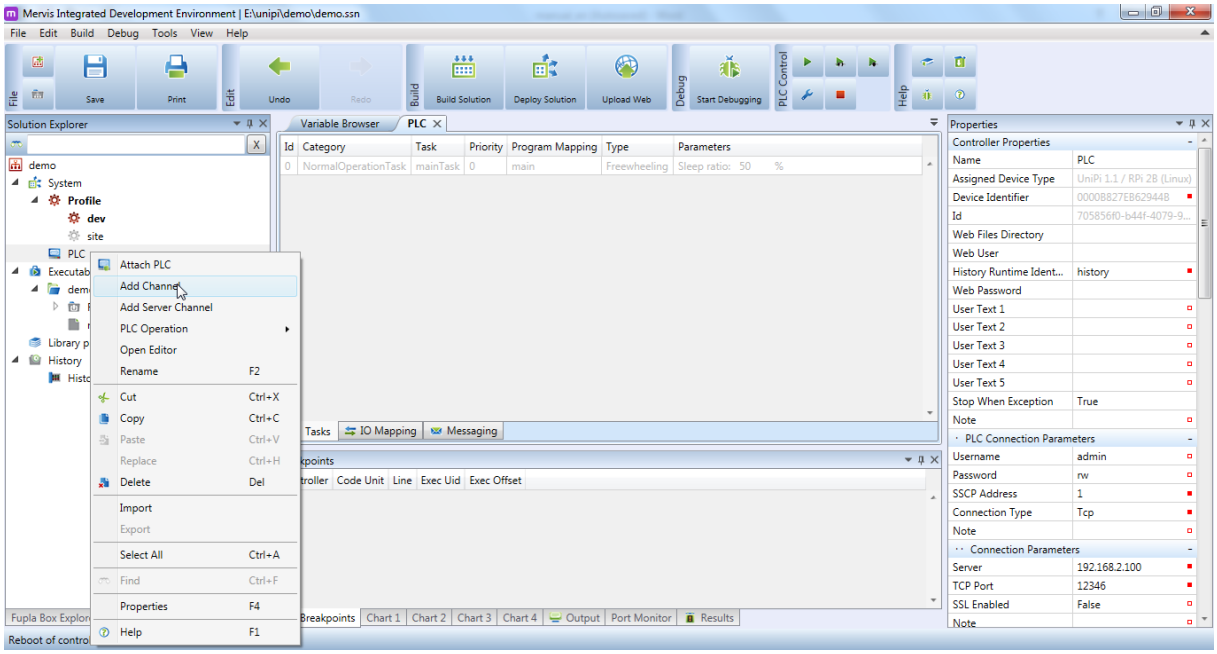
- After a successful upload, it is necessary to reboot the PLC. Right click on the PLC icon (in Solution Explorer) and choose PLC Operation -> Reboot PLC. A new menu appears which will show us two options of a reboot:
  - Warm Restart – reboot the device and use the most recent values of variables
  - Cold Restart – reboot the device and use initial values as set in the project defaults
- After the device reboots, the progress bar should display “Device reboot successful”



2.4.4 Attaching UniPi 1

2.4.4.1 Analog inputs

Right click on the PLC icon in Solution Explorer and choose Add channel



In solution explorer, under the PLC icon, you will find a newly created channel. Select that channel by left clicking on it and in the right part of the screen you will see its properties. Set is as following:

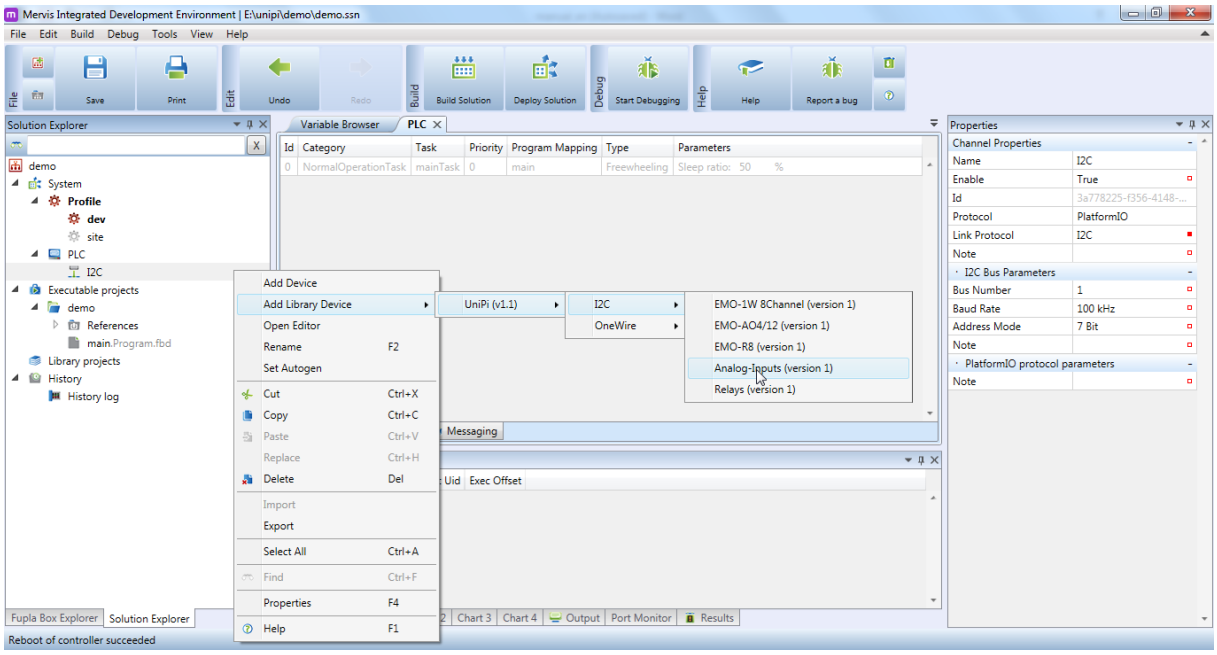
- Protocol – PlatformIO
- Link protocol – for analog inputs choose I2C
- Name – it is recommended to use a descriptive name (for example we can use the Link Protocol name - „I2C“)

Once the channel is configured proceed with adding devices.

There are two ways how to add a device:

- Manual – you must create the device and set all necessary parameters like address, reading and writing groups etc.
- From Library - you can choose a device that is already in the library

For analog inputs, you can use a predefined device. Right click on the created channel and select Add library device → UniPi (v1.1) → I2C → Analog-Inputs (version 1) from the menu. After these steps we will see a new device under our I2C channel which represents analog inputs.



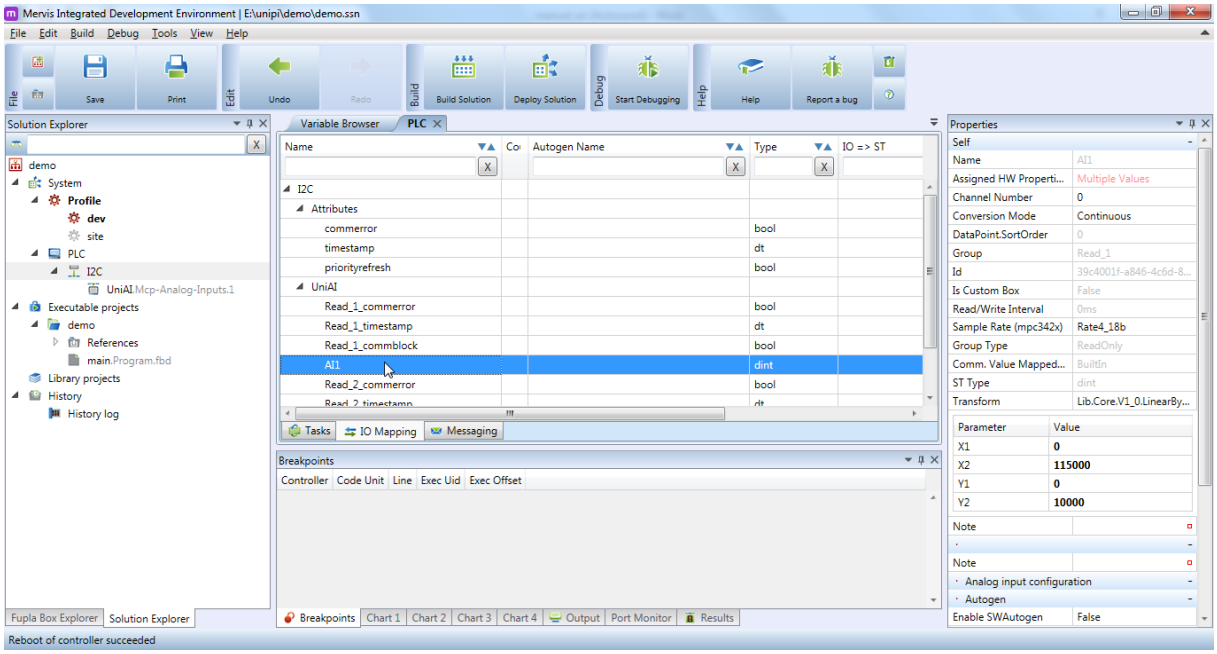
Because of being in Simple Mode, the Autogen feature automatically generates and maps variables to physical inputs and outputs.

Double left click on the PLC icon in Solution Explorer. This should show you a new PLC tab in the main window. Select IO Mapping tab (in the bottom section of the window). In this tab, you should see all input/output ports which have been added to your device.

For analog inputs, you need to locate ports with names AI1 and AI2. Choose the AI1 input by left clicking on it. This allows you to access the properties of the current port, where we can set the following parameters:

- Transform – this parameter allows to select a transformation of the physical value (linear, defined by two points, …) to a value represented in the program, e.g. that is more suitable for your purposes

- Mapping – the auto-generated and mapped variable. When in advanced IDE, you will see map your self-created variables to physical ports



Transformations

Available transformations:

- Identity – transform type 1:1 (input = output)
- ResistanceToTemperature – transforms the value to a temperature based on the value of the resistance by setting all the necessary parameters
- Linear – transforms the value based on a line defined by parameters q and k
- Threshold – transforms the value in a digital true – false value based on threshold value
- Window – a window (low and high parameters) which allows transforming discrete input value into boolean true/false value
- Negate – negates the digital input value
- LinearByTwoPoints – this type will return a value that lies on a linear line between the two predefined points
- LinearByTable – almost identical to LinearByTwoPoints transformation, but it allows us to define up to 8 points

When using transformations, make sure to set the parameters in the right order based on the fact if of using inputs or outputs or you might end up with values that won't be representing the real value.

In this case, it is predefined with LinearByTwoPoints transformation for our analog inputs. The set up values are in a range 0 – 10 000 mV.

2.4.4.2 Relays

Relays are connected via the I2C channel as well so you only need to add another device to the I2C channel. Otherwise you will need to add I2C channel, which is descripted in the beginning of Analog Inputs chapter.

Once the channel is set up then right click on the channel (in Solution Explorer) and from the menu select Add library device → UniPi (v1.1) → I2C → Relays (version 1)

See the auto-generated variables in Variable Browser.

2.4.4.3 Digital inputs and Analog Output

Digital inputs require a new channel – right click on the PLC in Solution Explorer → Add new channel.

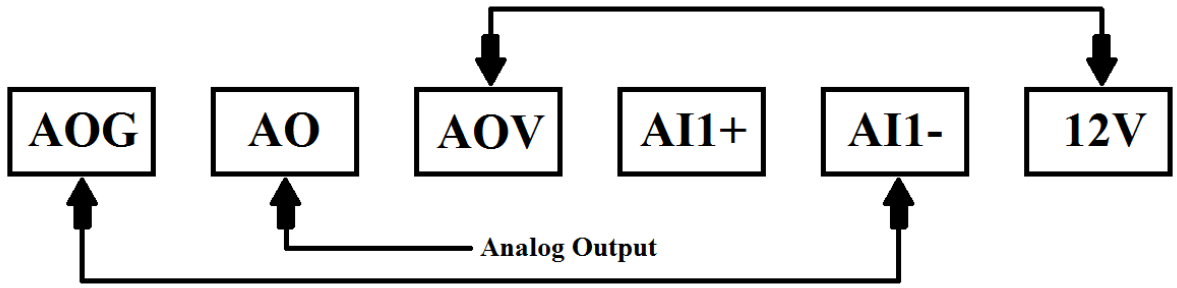
After the new channel is created under the PLC icon, left click on it and set the Properties on the right side of the screen:

- Protocol – set this to Modbus
- Link Protocol – set this to TCP
- Name – use a descriptive name (e.g. „Modbus“)

After that add a new device by right clicking on the channel and select Add library device → UniPi Technology → Unipi (1.1) AO/DI/CNT (version 1).

Digital inputs, analog output and counters has been added to the variables as (DIx, DI_CNT_x…).

The last step is to set a proper physical connection e.g.:



2.4.4.4 OneWire devices (Thermometer)

In order to set up the 1Wire devices you need to add a new channel which will represent the 1Wire channel.

Right click on the PLC in Solution Explorer → Add new channel

After the new channel is created under the PLC icon, left click on it and set the Properties on the right side of the screen:

- Protocol – set this to PlatformIO
- Link Protocol – set this to OneWire
- Name – again, use a descriptive name (e.g. „1Wire“)

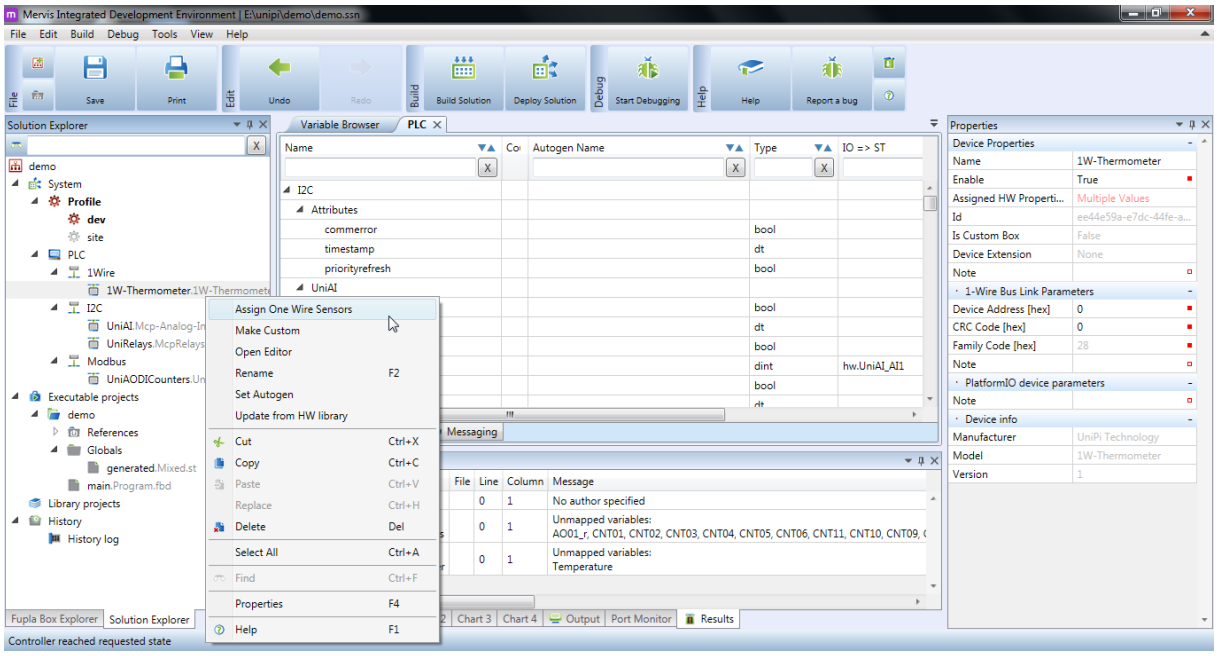
After the channel setup is completed add a new device by right clicking on the channel and choosing Add library device → UniPi (v1.1) → OneWire → 1W-Thermometer (version 1).

Because each 1Wire device uses a special address for identification, you need to set the find out the address, CRC and family code of the sensor.

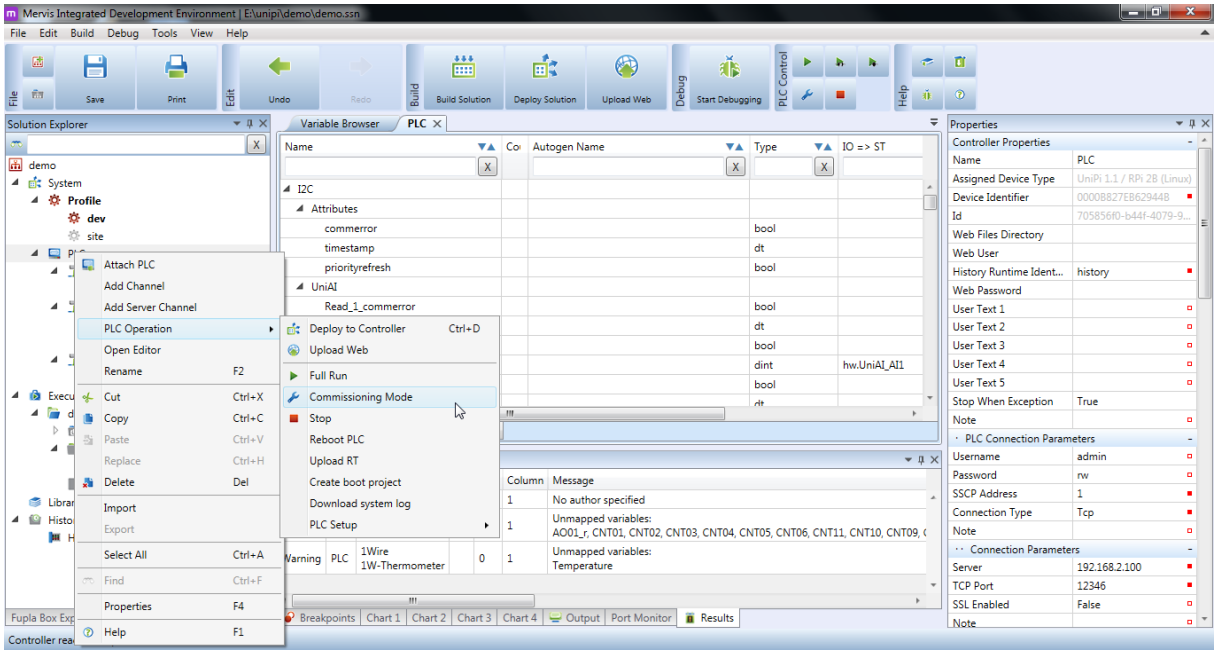
In order to do that, connect the thermometer to UniPi, compile and deploy the project to UniPi (Ctrl+Shift+D).

Now proceed with setting the address, CRC, and family code:

- Set the PLC to Commissioning mode – Right click on the PLC icon in Solution Explorer and choose PLC Operation→ Commissioning mode



- After that, right click on the added One Wire device (in Solution Explorer) and from the menu choose Assign One Wire Sensor
- After a short detection a table will show up with all detected One Wire Devices connected to UniPi. Select the device you wish to add.



- Now you have a fully functional 1Wire device which should include the assigned address, family and CRC code. As before Autogen created and mapped variables of the device.

2.4.5 Attaching Unipi Neuron

2.4.5.1 All IOs except 1-Wire

Adding most of the peripherals of all Neuron devices (except extensions) is simplified to a TCP Modbus channel. So, start by creating a TCP Modbus Channel:

Right click on the PLC in Solution Explorer → Add new channel

After the new channel is created under the PLC icon, left click on it and set the Properties on the right side of the screen:

- Protocol – set this to Modbus
- Link Protocol – set this to TCP
- Name – again, use a descriptive name (e.g. „TCP Modbus“)

After the channel setup is completed add a new device by right clicking on the channel and choosing Add library device → UniPi (v1.1) → and select the Neuron device you are using.

All the peripherals of the Neuron are now available in a Variable browser (if using Simple mode).

2.4.5.2 1-Wire devices

Adding a 1-Wire devices is done using the same process as in UniPi 1. Please see chapter 2.4.4.4.

2.4.6 Simple program

This chapter will guide you through a process of making a simple program.

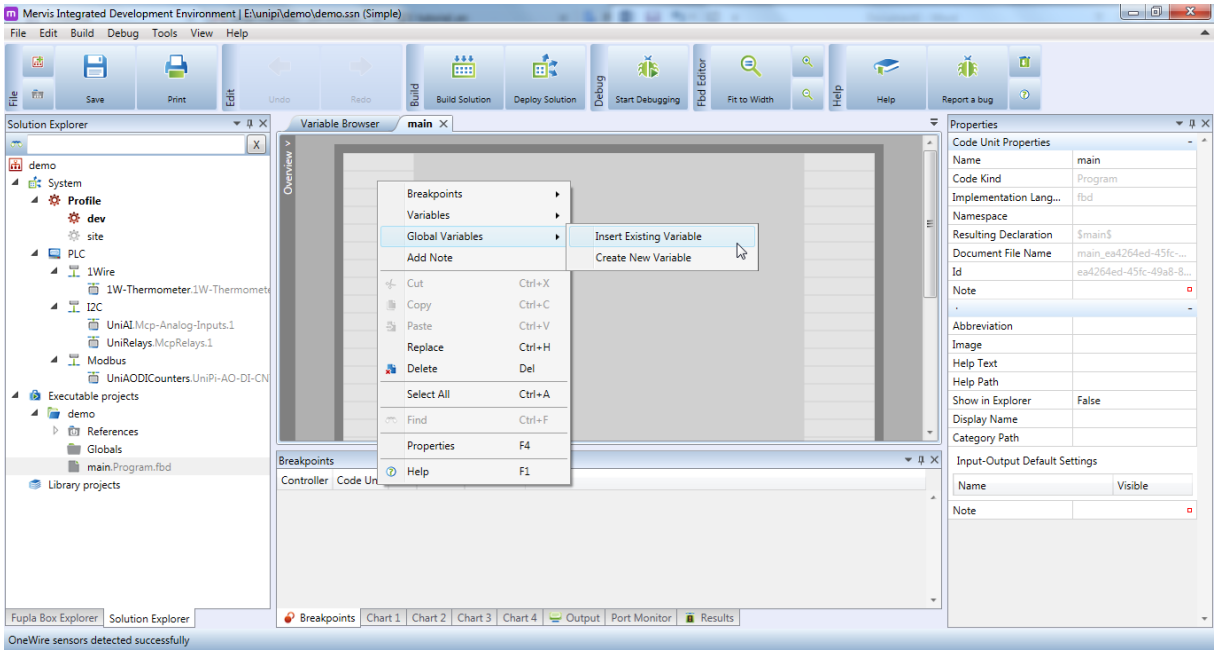
The program will switch a relay based on a status of digital inputs and results of some function blocks.

If you haven't done all the previous steps, start from the beginning of this manual – create a new solution -> attach UniPi -> configure inputs and outputs

Once done with the initial project setup, double click on main in Solution Explorer (under the tab Executable projects). In the middle main window, you will see the main control program including:

- Column showing inputs - read (left side)
- Area for placing and connecting Fupla blocks (center)
- Column showing outputs - write (right side)

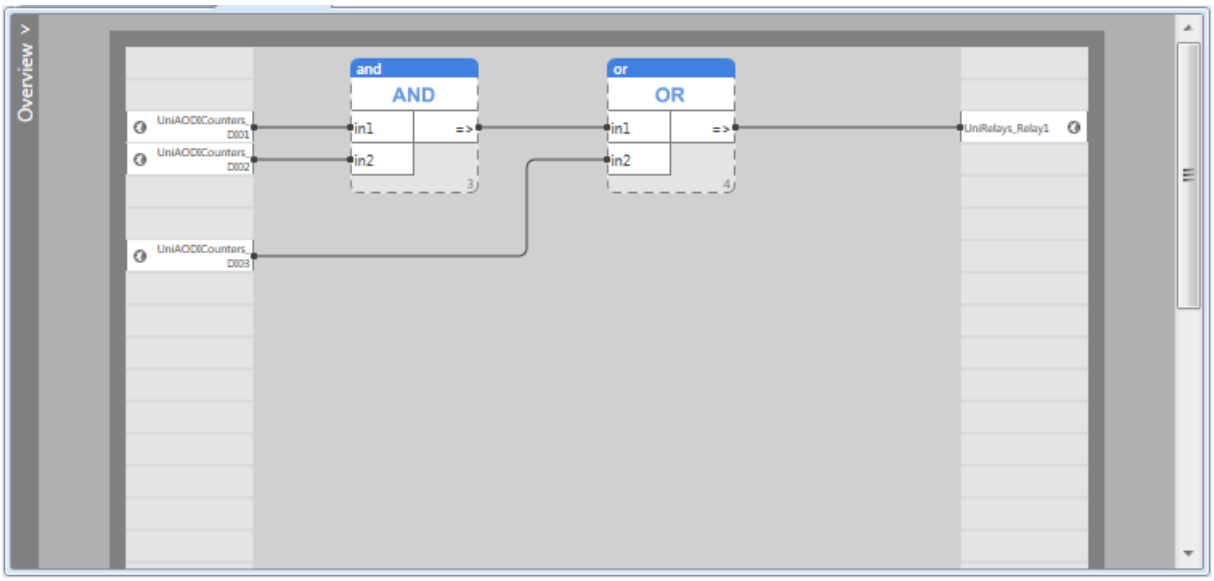
First add global variables – right click into the column for inputs and choose Global variables -> Insert Existing Variable. Gradually find and add variables UniAODICounters_DI01 - UniAODICounters_DI03



Then repeat the same process, but instead of adding a variable into the inputs column, add UniRelays_Relay1 into the outputs column

Then right click to the area for placing Fupla blocks, choose Add Library Box -> Add Library Box. Gradually find and add blocks and and or.

Make connections as in the picture below.



After finishing all the steps above, Build solution (Ctrl+Shift+B) and check for any errors or warnings, which can be seen in the Results tab at the bottom of the screen

- Error – critical mistake, which needs to be resolved, in order for the program to work
- Warning – just a reminder that something might be wrong and should be addressed, but it doesn't have any impact on the functionality

If the compilation is successful Deploy solution (Ctrl+Shift+D) to UniPi. A new window with Device run settings shows up where you can set how the UniPi runs.

Optimal settings are: Full Run + Memory location 1+2.

If the Deploy was successful, the program should be working.

Program description:

- Digital input 1 is ON AND on Digital input 2 is ON
- OR on digital input 3
- Relay 1 switches ON else is OFF

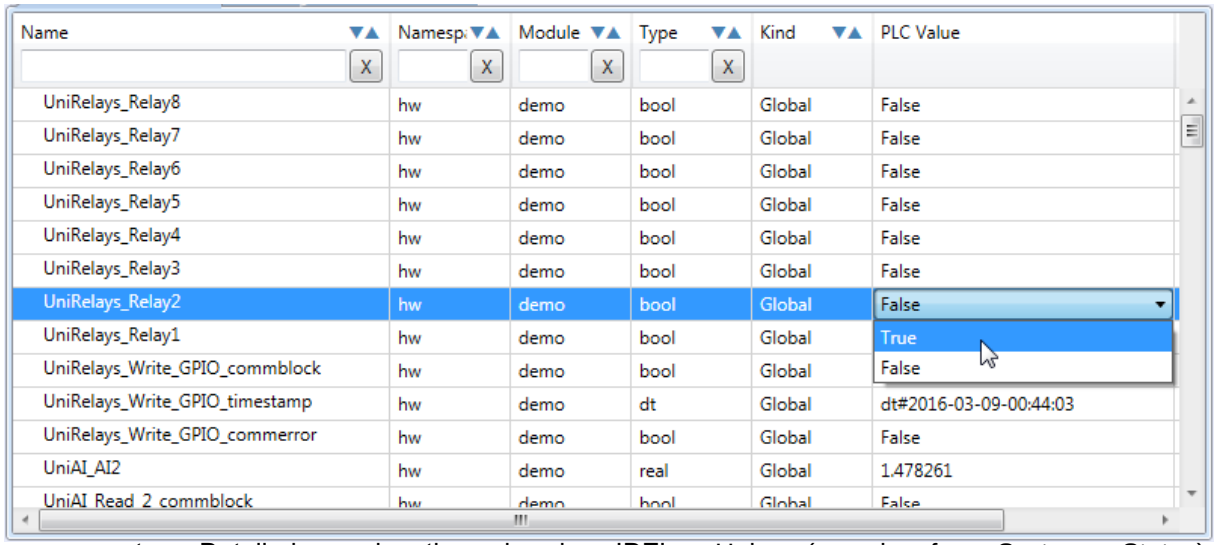
2.4.7 Debugging

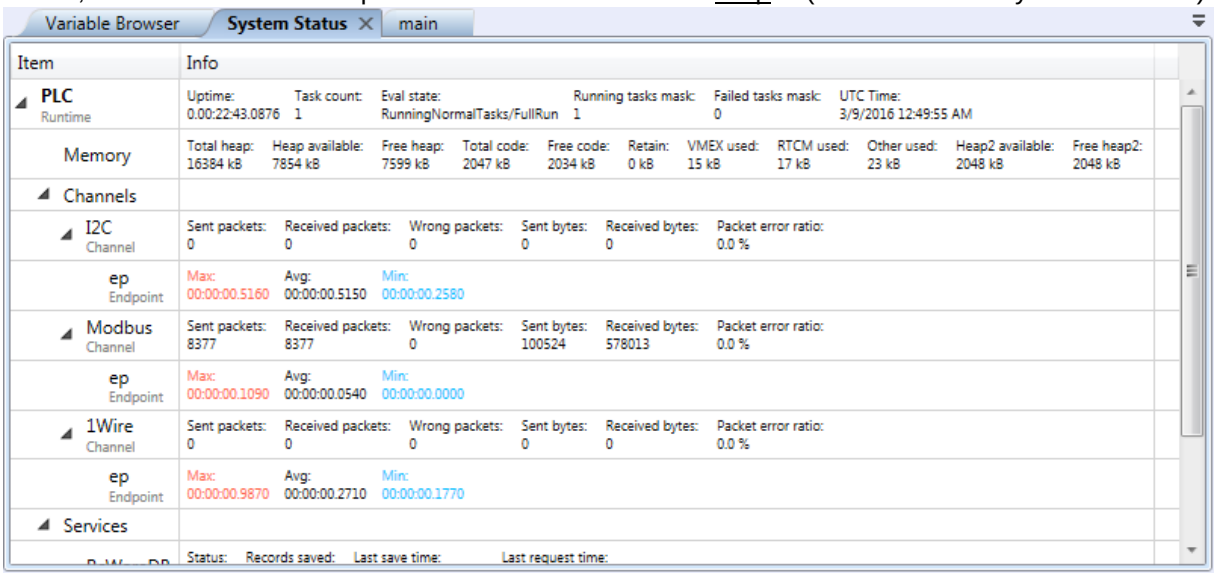
After a successful upload click the Start Debugging button.

When in debugging mode, you can see the values of all variables either in program, or in Variable Browser. You can also manually change the value of variables. Go to Variable Browser find the variable UniRelaysRelay_2 and change the value in PLC Value column to True. The relay 2 will switch on.

In debugging mode, a new tab in the central window opens – System Status.

It is useful to monitor used memory, sent or received packets, communication status with DB or Proxy server, etc. Detailed explanation is in IDE's Help (search for System Status).





3 HMI (Human Machine Interface)

For simple and user, friendly control interface of program running on UniPi, users can utilize the ability to create and deploy a web interface which can be accessed on a LAN by any device which has a compatible web browser.

First step is to make sure that project on UniP is configured and deployed properly.

To create web HMI use the Mervis HMI (see installation in the first part of this tutorial).

3.1 Creating a web

First step is to create a new project.

- On the top toolbar we need to select File -> New -> Project
- A new window will appear. Set the basic project configuration:
- AlarmsConfigFilePath – path for the file with Alarms configuration (leave it on the default path)
- Name – name of the project
- Path –path, where the whole web will be saved and stored and from there, it will be uploaded to the UniPi later on
- Size – used for changing parameters of height and width of the web panel

A new window will appear where you can change the color of the panel or set a background image

3.2 Connecting variables

Next step is to make a connection between HMI and variables from IDE project.

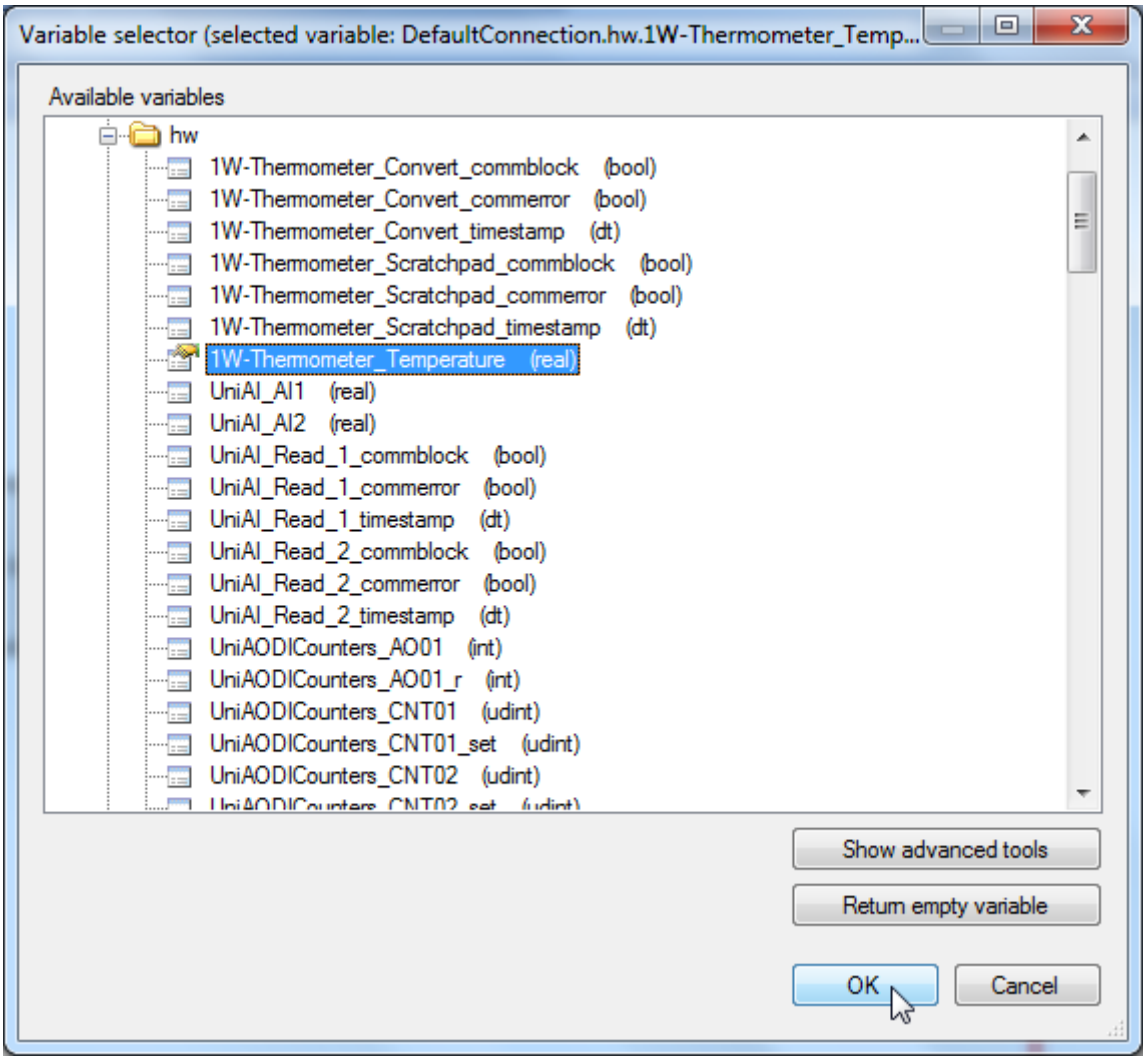
- Select File -> Variable Manager...
- Click on the Select File and go to the project folder and open the bin folder, choose the *.exs file and confirm. After that, press Retrieve which retrieves all the variables used in the project

3.3 Gadgets

On the left side of the Mervis HMI environment see all gadgets which we can be added to the HMI. Gadgets include indicators, setters, progress bar, graph, etc.

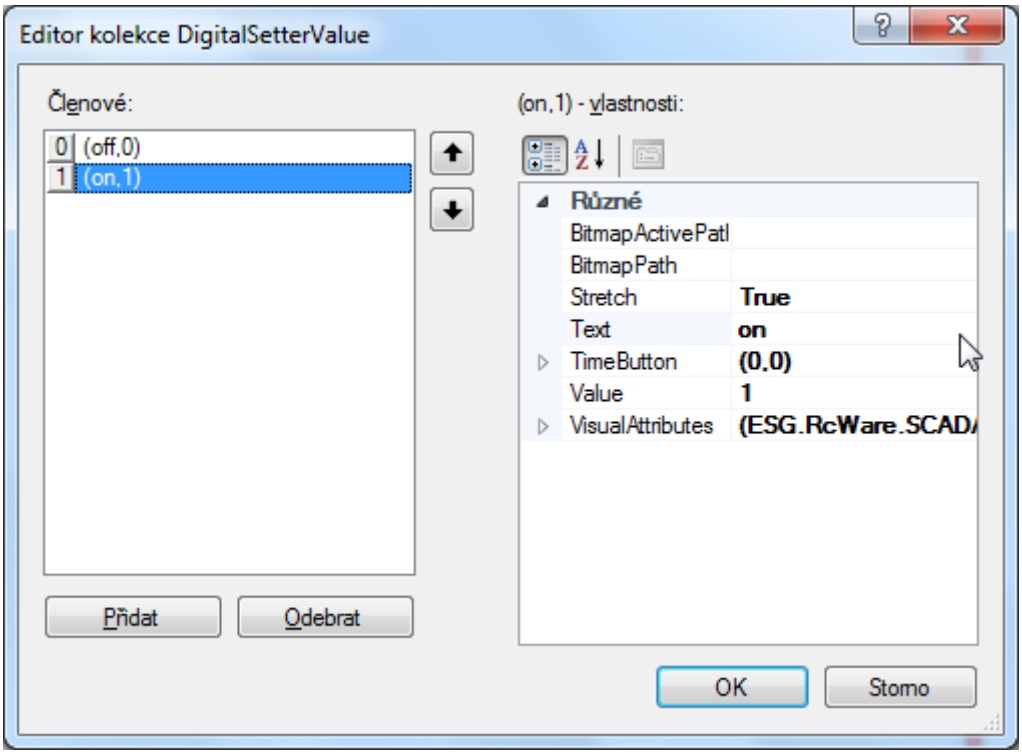
In this tutorial, will use the digital setter and analog indicator. Other gadgets can be set and utilized similarly.

- Left click on Analog indicator in Gadgets Panel, hold, drag and drop it into the HMI Panel. Properties of the gadget will show up on the right side of the screen. Attach desired variable (E.g. temperature) by left clicking on the button with three dots and choosing the required variable.



- Next, drag and drop the Digital setter. In the properties, attach a variable again, and then go to DigitalSetterValues. Here add states of the setter for setting the value by clicking on Add. Every state can be set with: text, picture, background color, or defined with value which will be inserted into our variable

These two simple gadgets can be used e.g. for indicating temperature (analog indicator) or switching relays (digital setter).



3.4 New pabel

If you need more panels in web interface, e.g. first panel for setting values (control) and second panel for displaying values (overview), just create a new panel, and then switch between them in the web page

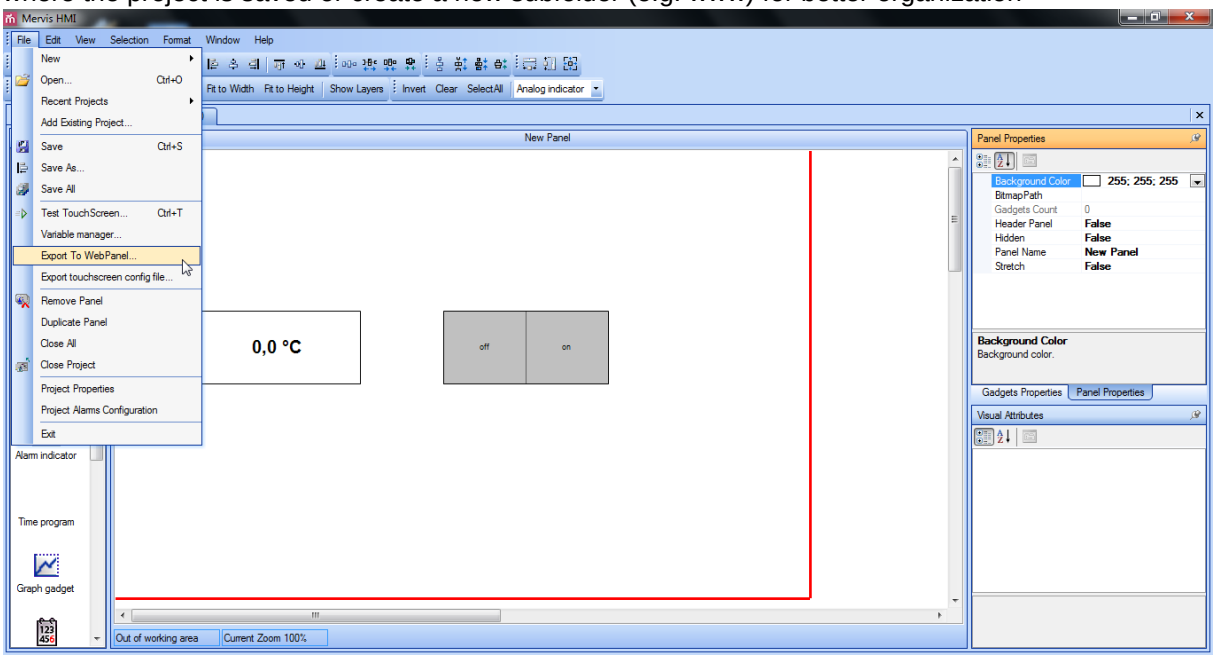
- Choose File -> New -> Panel.
- Fill in the same properties as when creating a new HMI project

3.5 Uploading web

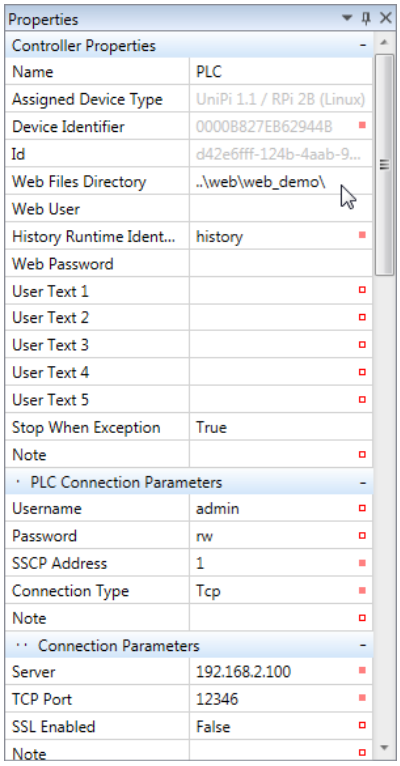
Once the design of HMI is complete, proceed with uploading the HMI to UniPi.

- First save the HMI project. File -> Save. Then export the project to a folder, from where it will be uploaded to the UniPi. File -> Export To WebPanel.... Choose the same folder as the one,

where the project is saved or create a new subfolder (e.g. www) for better organization



- Now go to Mervis IDE and open the project. You have to add the exported project to the device configuration. In Controller properties locate Web files directory column, and add the path to the exported folder. You can also set up Web user and Web password, if you needrestrict access to control of the web panel



- Now left click on the PLC in Solution Explorer, and then click on Upload web. After the upload is done, you can open web browser and type the IP address of the UniPi as an URL address. You should see the web panel. If not, try refreshing the page or rebooting the UniPi.

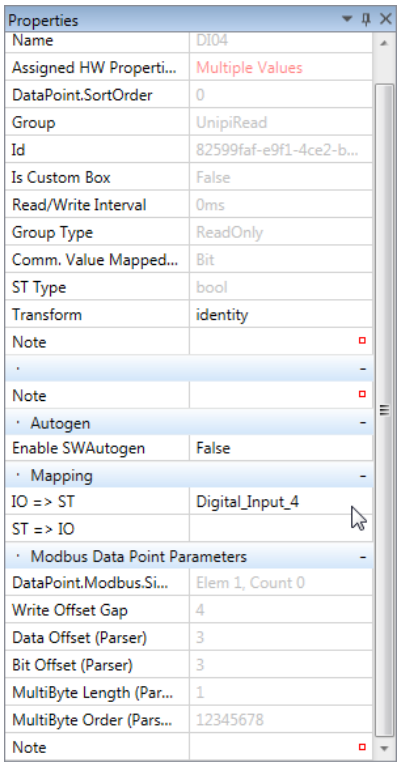
4 Advanced IDE

This chapter describes configuration and usage of advanced features of Mervis IDE, such as Mervis DB logging or remote access to the UniPi via Mervis Proxy. This features are available only in full mode of the project.

To switch to full mode right click on name of your project in the Solution Explorer and select Switch to full mode.

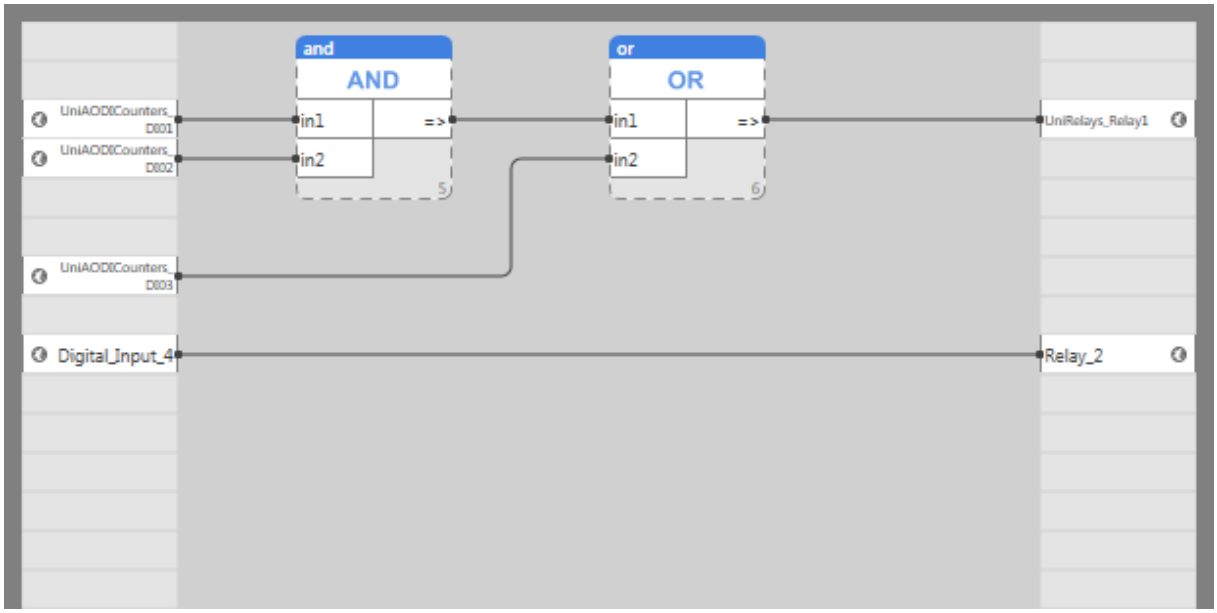
4.1 Manual mapping of HW prototypes to variables

- First create a new global variable
- Go to main and right click into the column for inputs and choose Global variables -> Create New Variable. Name it Digital_Input_4 and as a type choose bool (can also be done in ST language)
- Now create an output global variable. Name it Relay_2 with bool
- Go to IO Mapping, locate the variable DI04, and make sure the Autogen is disabled for the variable (uncheck Enable SWAutogen in Properties) and in the IO => ST column type our newly created variable Digital_Input_4



- Note: column IO => ST is for reading from inputs, column ST => IO is for writing to outputs
- Do the same for the second variable: Relay_2 to Relay2 (ST =>IO)

- Try changing the new global variables instead of the generated variables in the project program



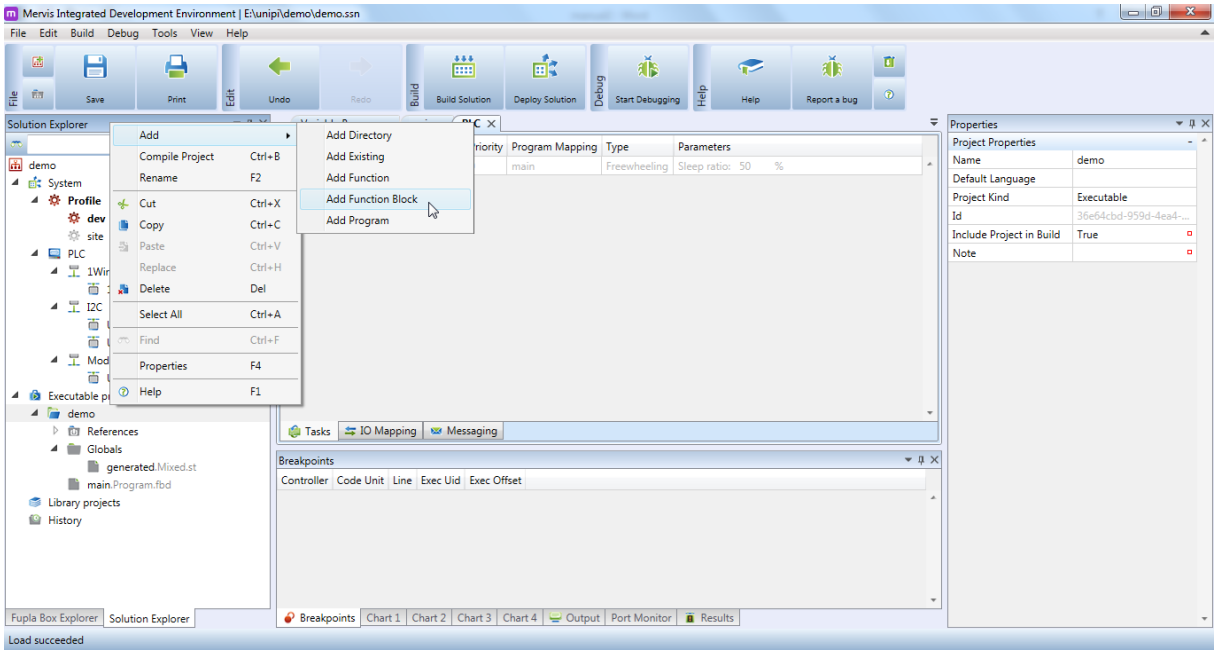
4.2 Creating function blocks

If the basic function blocks is not enough, or some of the blocks (or code) is repeating in the program, users can create their own function blocks

This is very useful in more cases, for example: If you use some operation more times in a program, create the operation in a function block and then use only the one function block. Then the program is simpler, shorter and more elegant and changes to the behavior is done only in one place.

- Before creating a function block, first switch to Full Mode. The function block will be created in ST (structured text) language, and it is only supported in full mode.
- Up to now, all examples were shown using FBD (function block diagram) language. These two program description (or languages) are equal and can be combined

- To create a new Function Block right click on project icon in Solution Explorer and choose Add -> Add Function Block. Fill in the name (e.g. conversion) and choose st as a language

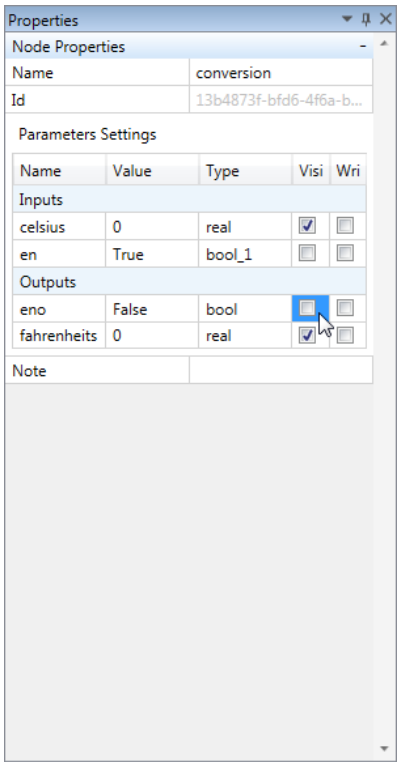


- A new program appears in the Solution Explorer, double click on it to see the function block's structure. Type the following code to the editor:

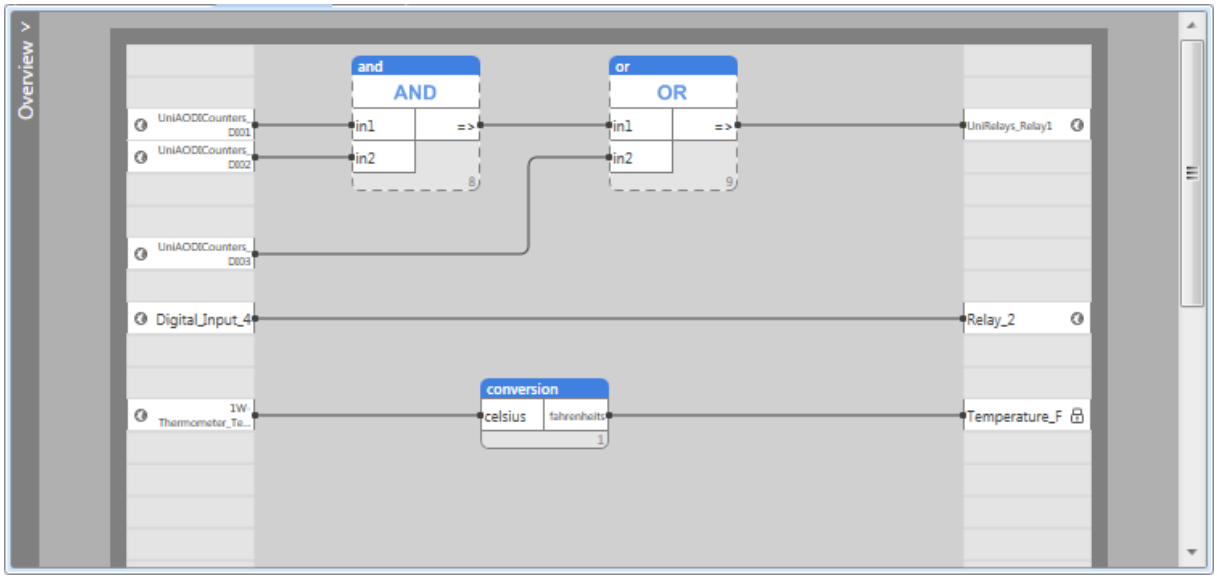
```
FUNCTION_BLOCK conversion

    VAR_INPUT
        celsius : real;
    END_VAR

    VAR_OUTPUT
        fahrenheits : real;
    END_VAR

    fahrenheits := (9*celsius)/5 + 32;

END_FUNCTION_BLOCK
```

- It should be clear, that the function block will convert temperature in Celsius degrees °C to Fahrenheit degrees °F
- Save the project and go to main. Insert an existing global variable 1W-Thermometer-Temperature to inputs column, which represents temperature from OneWire sensor. Create a new local variable Temperature_F in outputs column, which will represent temperature in °F – as a type choose real
- Now right click to the area for placing Fupla blocks, choose Add Library Box -> Add Library Box. Find our function block conversion and confirm

- Left click on the added box and uncheck en and eno in properties



- Make connections as in the picture:

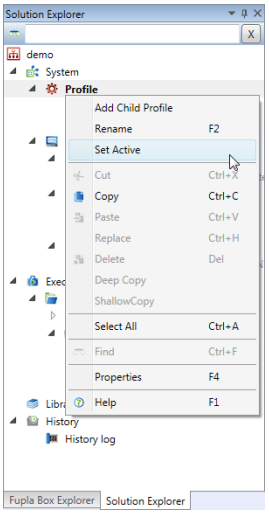


- After deploying solution to the UniPi, the data from 1W-Thermometer-Temperature (temperature in °C) will be converted to temperature in °F by function block conversion, and stored to the local variable Temperature_F.
- Check the correct working in debugging mode

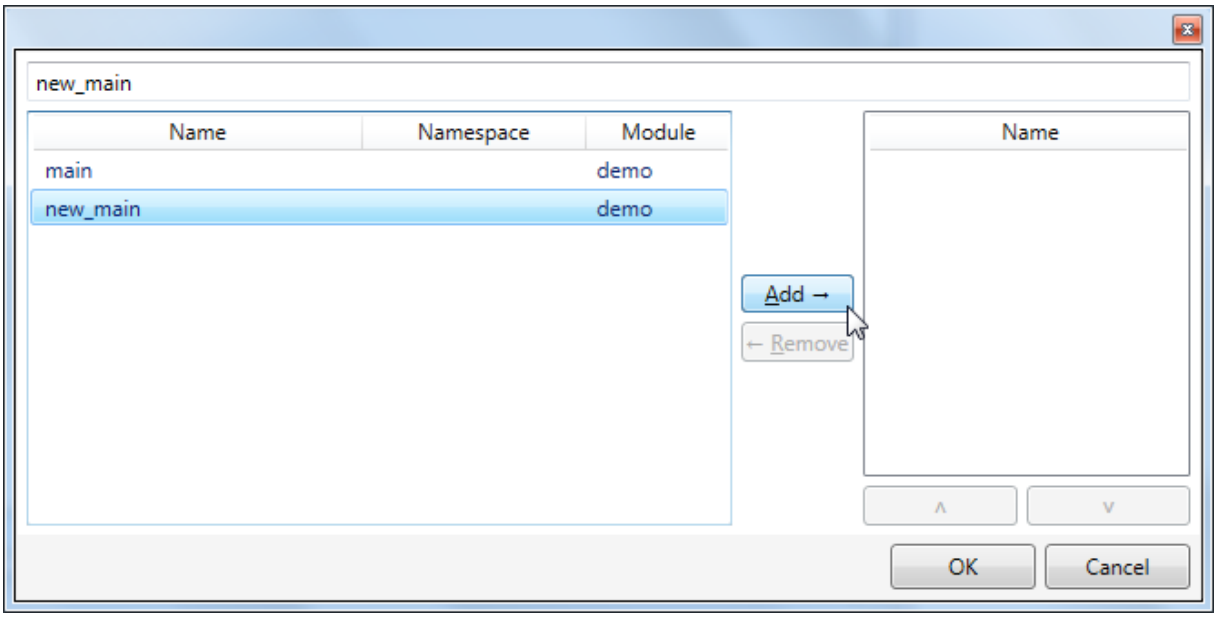
4.3 Creating a new program

- For better and clear division of behavior of the PLC, the UniPi can run multiple programs at once (e.g. one program switching lights, second for controlling room temperature, …)

- To create a Program right click on project icon in Solution Explorer, choose Add -> Add Program, fill in the name (e.g. lights) and choose type st/fdb
- In order to tell the UniPi to include also the new program, set active the base to the Profile by right click on the Profile and Set Active



- Then double click on PLC icon in Solution Explorer and go to Tasks tab. Now it is possible to change the program assigned to the task. After going to Program mapping, users can assign which programs will run and which not.
- Note: Never add more tasks! The whole program will fail.



4.4 SSCP communication

SSCP communication is used for reading/setting values from one UinPi to another or even to Proxy (e.g. for communication of two UniPis that do not reside in the same IP network).

The problem will be explained on a project SSCP_multiple_plcs, which we can download from http://downloads.unipi.technology.

- First attach a UniPi normally as we did before and set all the necessary properties. Name this UniPi e.g. Master. Then add a channel I2C and UniRelays device. Map a variable to Relay1 (use Autogen or manual mapping)

- Next add a second UniPi (right click on System -> Add new Controller) and name it e.g. Slave. Then add a channel Modbus (Tcp) and UniAODICounters device. Map a variable to DI1
- Now select the Master UniPi again and add a new channel with SSCP protocol and TCP Link protocol, name it e.g. sscp. Then right click on the new channel and select Add device
- Go to Master UniPi's properties, and remember parameters Username, Password and SSCP adress(default is admin:rw). Go back to the created device for SSCP communication and set parameters Name (e.g. to_slave_unipi), Device IP Address (same as the Slave's IP address) and Device address, User Name, Password (remembered parameters)
- Double click on the new device, and in the new opened window right click anywhere and choose Add Group. Left click on the first line (Group) and edit properties: Name (read_group), Group Type (ReadOnly)
- Go to main, and create a new global variable slave_unipi_di1 (input column)
- Double click on the new device, right click anywhere to the group and select Add Data Point. In properties of the new data point, map the variable slave_unipi_di1 to the column IO => ST. As a Source Variable, choose the variable mapped to DI1 of Slave UniPi
- The communication should be now set. As a last step, go to main, and make a connection between slave_unipi_di1 and Relay1 (output column)
- The master reads boolean value of variable DI1 from the slave, store it in slave_unipi_di and sets relay on/off depending on its value
- ATTETION: The maximum number of data points per channel is 12!

4.5 UniPi database access

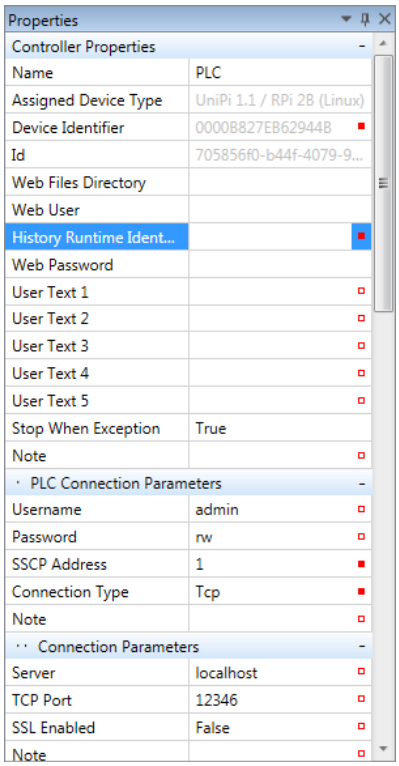
Mervis database (DB) can store historical data from various data points to the cloud database. This is extremely useful when dealing with correct behavior of the system.

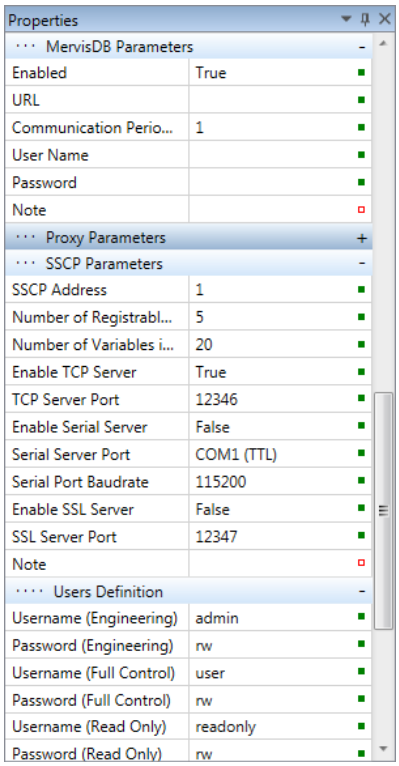
The data can be viewed from either from the online viewer Web or online Scada system.

4.5.1 Setup

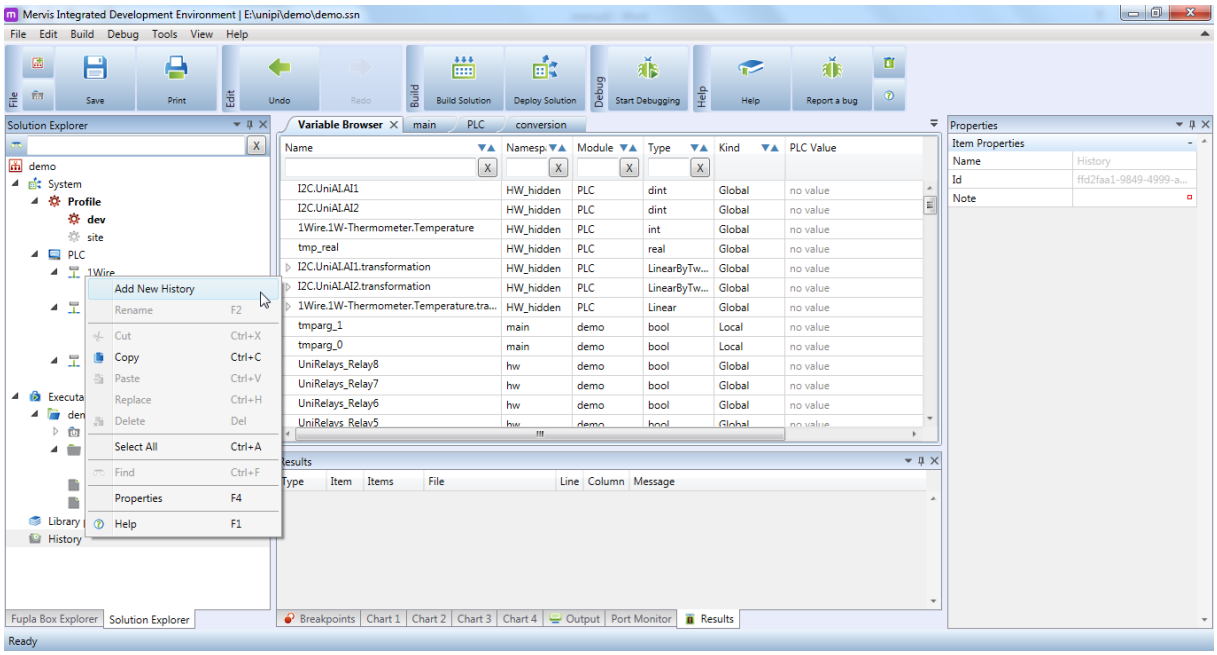
Make sure to be in advanced mode.

- Left click the PLC icon in Solution Explorer and look to the properties tab located on the right side of the screen
- First step is to set a unique history name identification in the History runtime Identification
- In order to save values fill in the necessary parameters in MervisDB Parameters
    - Enabling database – parameter Enable set on True
    - URL – URL address of database (MervisDB URL)
    - Communication Period – represents how often will the data be written into the DB
    - Access authentication – parameter User name (DB username) and Password (DB password)





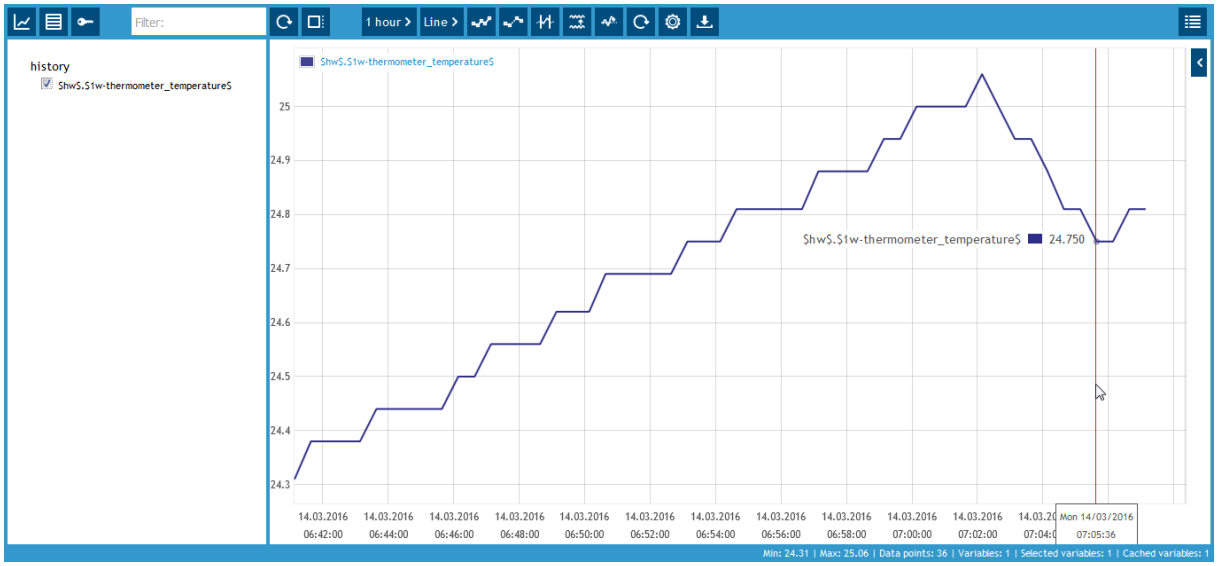
- The final step is uploading the configuration into the device and performing a device reboot
- In order to save variables from the UniPi to the database create a history, where the variables will be stored
- See a tab named History in Solution Explorer. Right click on it and choose Add new history. A new history instance will appear and configure it with the following parameters:
    - History Name
    - History Saved on PLC –> True - the data will be stored in our UniPi memory and later on they will be uploaded to our DB
    - Save period – defines how often the data will be saved in History (either synchronously or asynchronously to an event)



- First select which variables will be stored in the History
- UniPi has a part of its memory reserved for storing the history. Thanks to this, even in a case of a connection error, the values will be buffered in the memory and once the connection is restored, uploaded to the DB
- Variables can be assigned to the History from Variable browser. In browser locate the variable and right click om it and from the menu choose Add to history and select the desired history
- Deploy the project to UniPi and start debugging mode
- After the configured time you should see an increasing count and time of last request of Records saved to DB in the System status under Services

4.5.2 Online access

- If the previous steps are completed, go to https://db.unipi.technology/viewer
- Fill in your User name and Password and confirm by clicking on Set
- It is possible to choose between showing a graph or text log, change the showed time interval, change the style of drawing the graph etc.
- Note: If no data have been uploaded to DB, you will not be able to log in to the viewer



4.6 Proxy connection

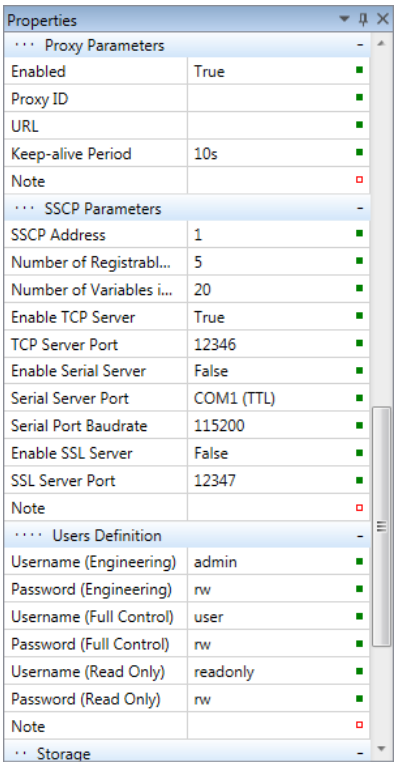
Proxy connection allows accessing UniPi remotely. This allows users to change anything in the program (e.g. value of variables) or even completely reprogram the UniPi.

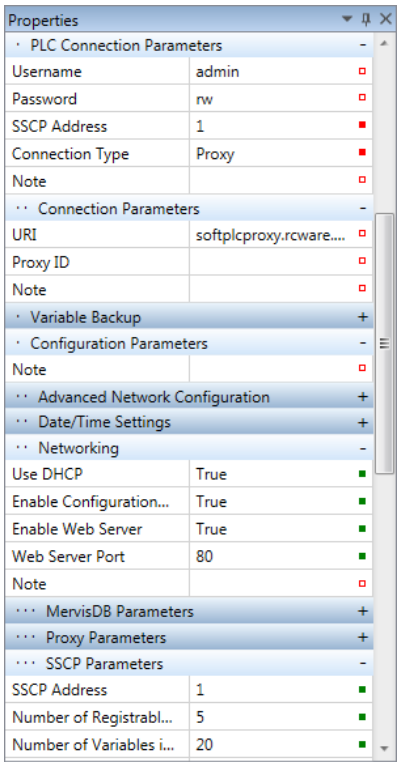
Some settings cannot be changed (such as IP address) to prevent users from losing the remote connection with UniPi.

4.6.1 Configuration

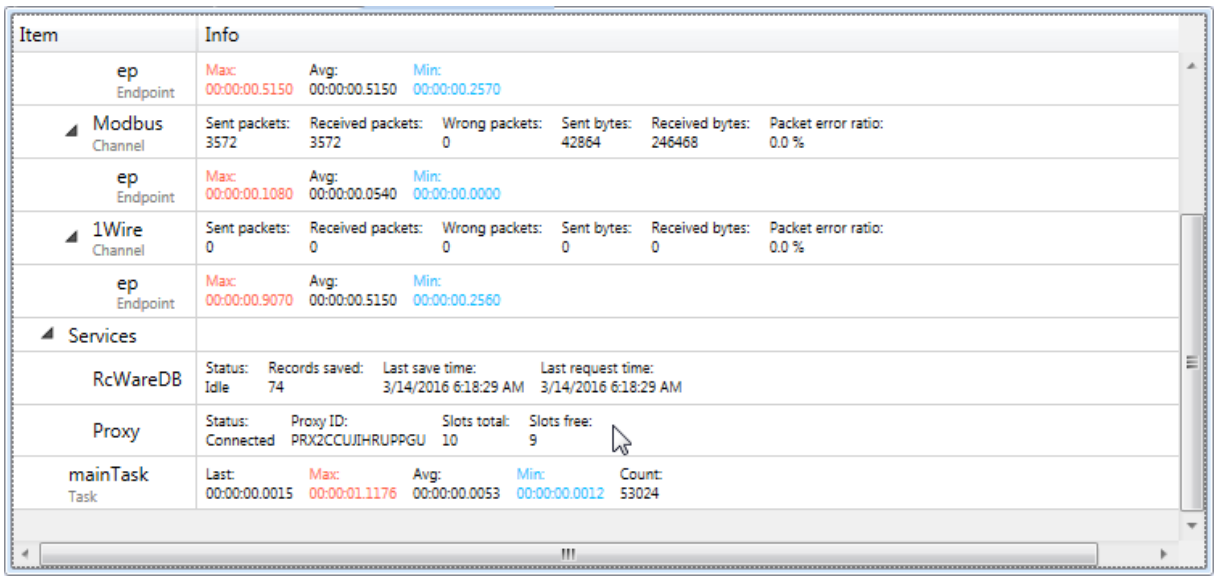
- Left click the PLC in Solution Explorer -> in properties locate the Proxy Parameters and set:
  - Enable proxy connection – parameter Enable must be set on „true“
  - Proxy ID - a unique ID string (Proxy ID)
  - Parameter URL must contain the URL + port address for the Proxy server which will be used (Proxy URL)
  - Parameter Keep-alive period can be optionally set to a different value
- Upload the configuration to UniPi and perform a device reboot
- After a successful device reboot change the connection parameters from TCP to Proxy:
  - In device properties we locate PLC Connection Parameters:
    - Parameter Connection type must be set to proxy

- In device properties we locate Connection Parameters:
  - Parameter URI must contain Controller connection URI address
- Parameter Proxy ID must be identical with the one assigned in previous step





- Try to connect to your UniPi using proxy (no other configuration uploads or reboot is required) - just go to Degugging mode
- You can also test the connection by downloading device log, rebooting the device, uploading the main control program etc. When in debug mode and look into System Status, you should see that number of free proxy slots is one lower then number of all slots



It is good idea to set up one profile for TCP connection and one profile for Proxy connection. Example:

- First do all the programming and setting the device properties in „mother's profile" (profile dev – This profile will communicate with UniPi on LAN). Then we add a new child profile proxy (right

click on profile dev and choose Add Child Profile). This profile will be used for remote management.

- Remember that the changes you make in the „child's profile" are not transferred to the upper profile and not all parameters are inherited from the upper layer profile