TECHBASE
AUTOMATYKA ELEKTRONIKA INFORMATYKA PRZEMYSŁOWA

The
iMOD Series

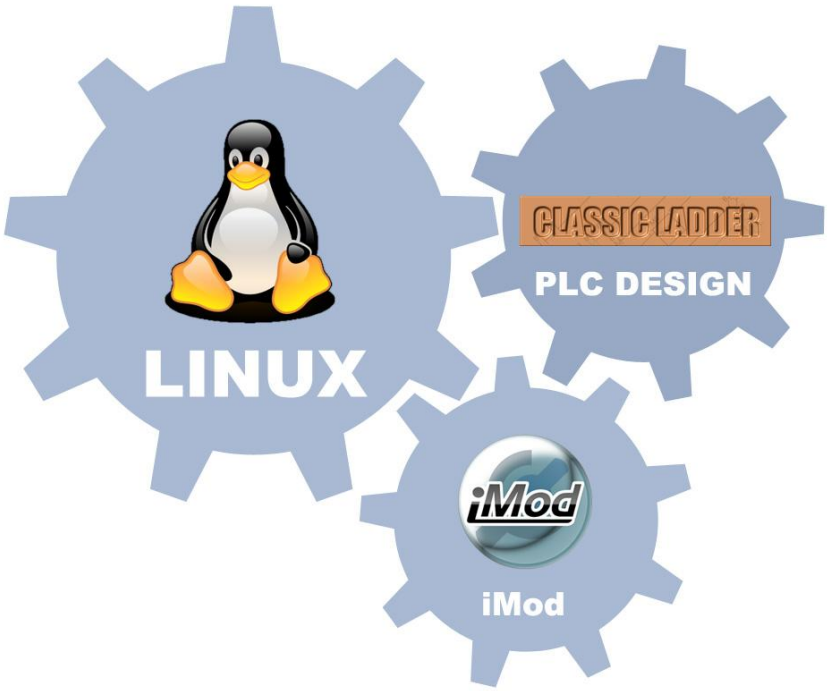
iMod + PLC Manual

Table of contents:

Chapter 1. Introduction

The iMod operating on the NPE platform can work as a PLC. In order to do that you need to use the ClassicLadder application for converting ladder diagram into C programming language.

Chapter 2. Preparation of working environment



In order to get the PLC functionality on the iMod you need:

1. Linux System or a program for operating virtual machines e.g., Oracle VirtualBox together with the Linux image
2. ClassicLadderDesign Software
3. The latest version of the iMod software engine on the NPE platform (further information can be found in the chapter on preparation of the iMod application)

The files of the projects used in this manual can be found on the FTP server:
Address: ftp.a2s.pl
login: npe_iMod@ftp.a2s.pl
password: npe_1m0d

Linux on any system (VirtualBox + image)

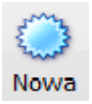
In order to build a PLC application you need Linux with the GTK+ graphical user interface.

You can run Linux directly from Windows with a virtual machine.

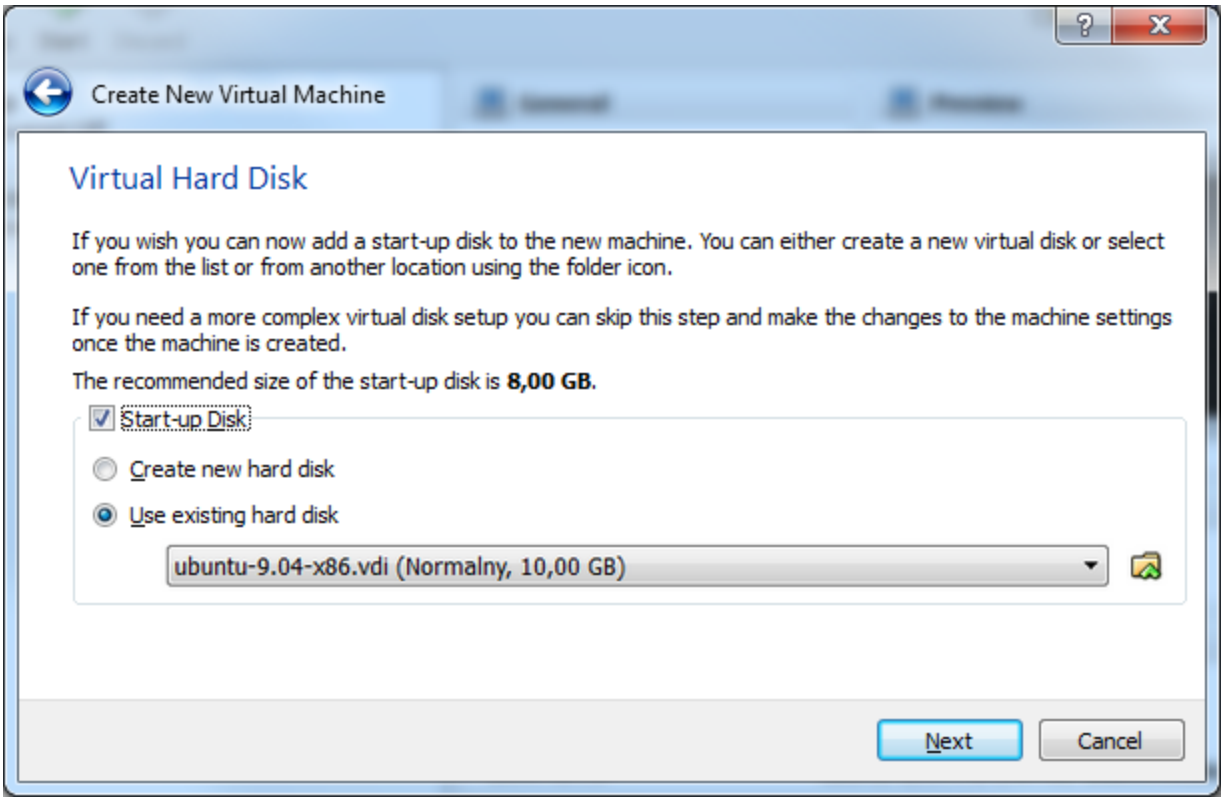
Download the VirtualBox software from this website:
https://www.virtualbox.org/

Download the Linux image from this website:
http://virtualboxes.org/images/ubuntu/

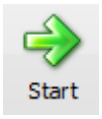
1. Run the VirtualBox program.



2. Create a new virtual machine by clicking the New button.
3. Name the virtual machine, choose Linux as an Operating System and choose a proper version of the system.
4. Assign the minimum amount of RAM for the virtual machine (256 MB).
5. Choose the 'existing hard disc' option and select a file with the Ubuntu system image.

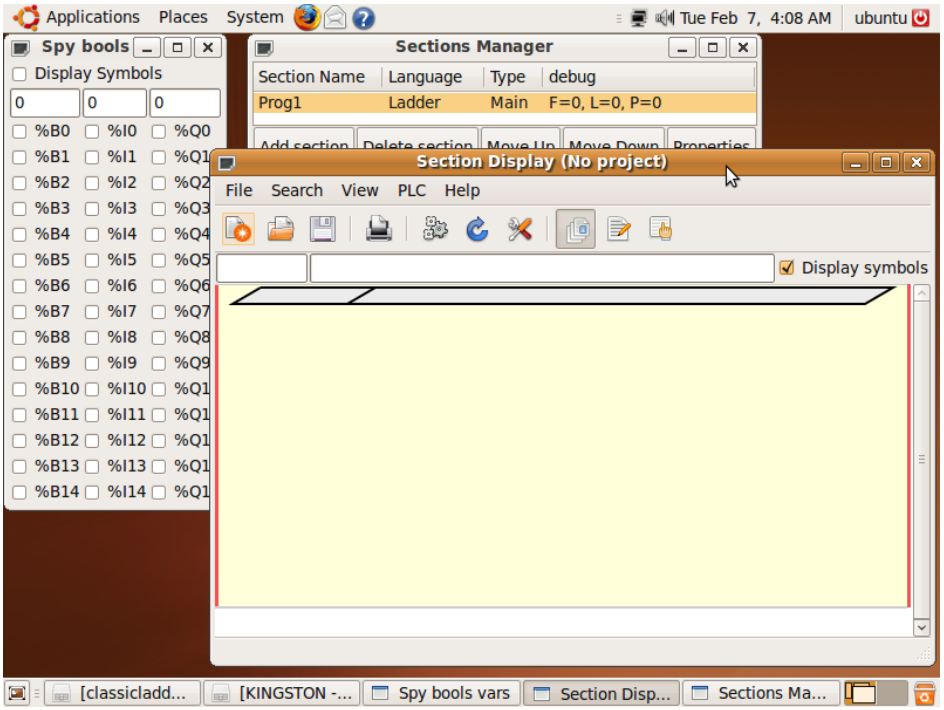


6. After a correct image upload, a window with parameters of the virtual machine appears. Click on the added virtual machine with the right mouse button and choose the "Start" option.



Linux password and login should be entered on a website where the system image is downloaded from.
For the Ubuntu Linux 9.04 version:
login: ubuntu
password: reverse

Ladder Design application



The ClassicLadder program is used for the purpose of this manual.

The Classic Ladder software can be downloaded from this website:

http://sourceforge.net/projects/classicladder/

or from the ftp.a2s.pl

In order to add new elements in the ladder diagram in the application, run the editor window:

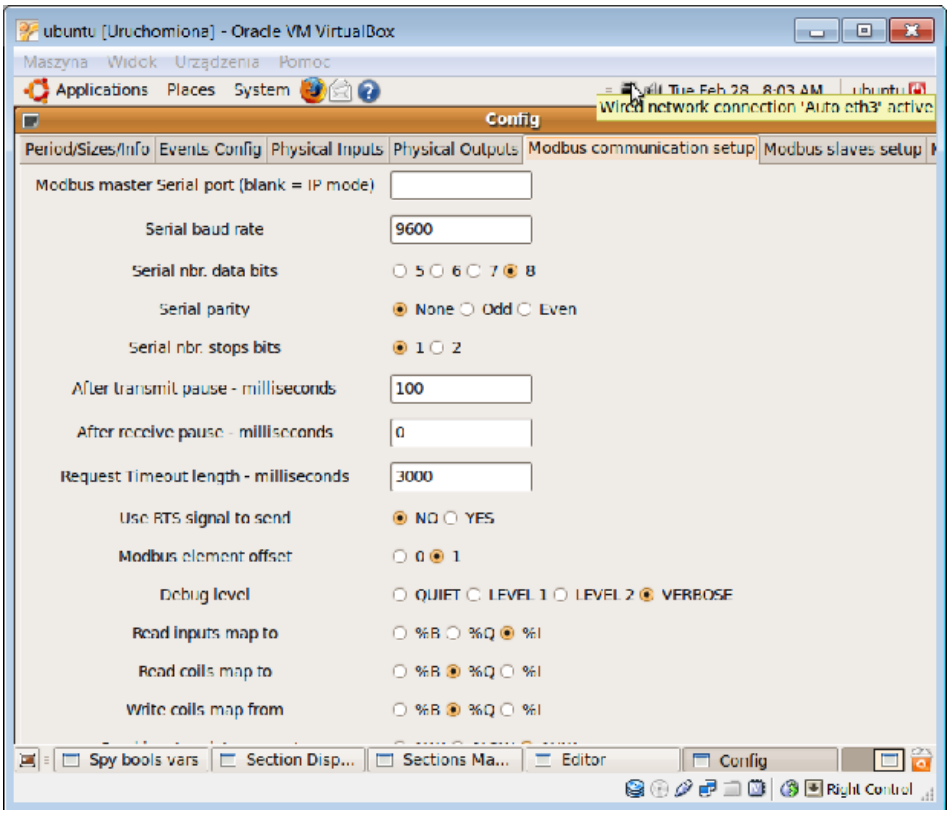
View-> Editor Window -> Modify

You can assign an address to a variable and in some cases the remaining parameters of a function block in the *Properties* window.

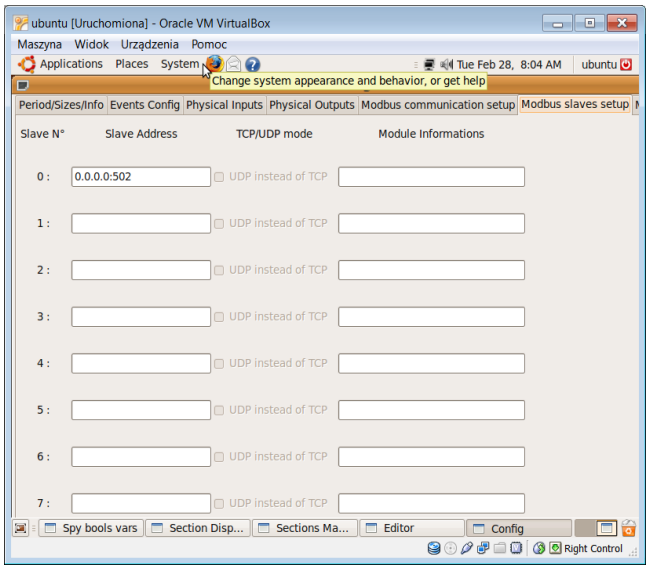
Remember that before saving a project you need to close the edit mode in the Editor window by clicking the OK button.

Program configuration

In order to access the configuration settings you need to go the *PLC → Configuration*.

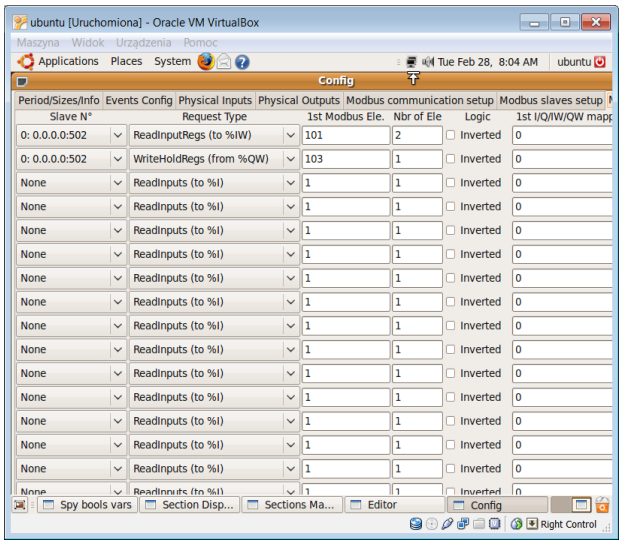


Set the communication parameters with the device in the *Modbus communication setup* tab. When you leave an empty *Modbus master Serial port* field, communication is carried out via the Ethernet interface.



Enter the IP and communication port for proper slave modbus devices in the *Modbus slaves setup* tab. Note the id number of a slave device.

The IP 0.0.0.0 address means a local host.



In the *Modbus I/O Setup* tab enter all parameters, with modbus addresses and the type of a request. Each parameter should be assigned to a proper slave device.

In order to save changes in the configuration parameters, you need to close the *Config* window. Changes are saved automatically.

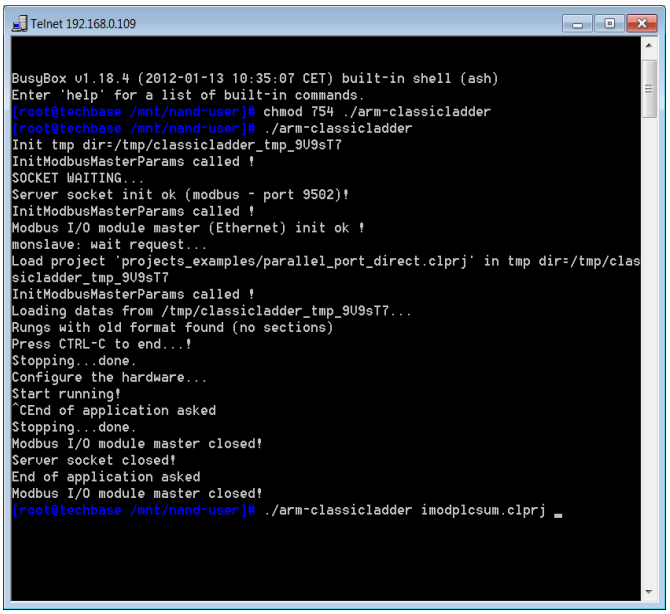
iMod

Before performing next steps, make sure that you have the latest version of the software with the *softmgr* application.

Free technical support applies only to the latest stable packages. You can update the iMod engine with the following command:

softmgr update imod

In order to change into the PLC mode, run the ClassicLadder project on the iMod with the *arm-classicladder* application. The application is available on the ftp.a2s.pl server (login and password are at the beginning of chapter 2).



1. Copy the *arm-classicladder* and *projekt.clprj* files (project of the previously created PLC application) in the binary mode into the following directory:

/mnt/nand-user/

2. Give the appropriate permissions for the arm-classicladder file with the command:

   chmod 754 /mnt/nand-user/arm-classicladder

3. Run the *arm-classicladder* application:

   /mnt/nand-user/arm-classicladder

4. Run the project with the *arm-classicladder* application:

   /mnt/nand-user/arm-classicladder projekt.clprj

You can close the PLC application by interrupting the program with the combination of CTRL+C or during the iMod reboot.

The PLC application is interrupted at the reboot of the iMod device. This is why you should add a startup script into the */mnt/mtd/rcs* file which results in auto running of the script after a reboot of the device.

In order to run local applications, you need to go to the PLC *menu-> RUN*. With the *Spy bools var* window(*View -> Bools var window*) and the Spy free vars window (*View-> Free var window*) you can control states and values of parameters.

Chapter 3. Example configuration

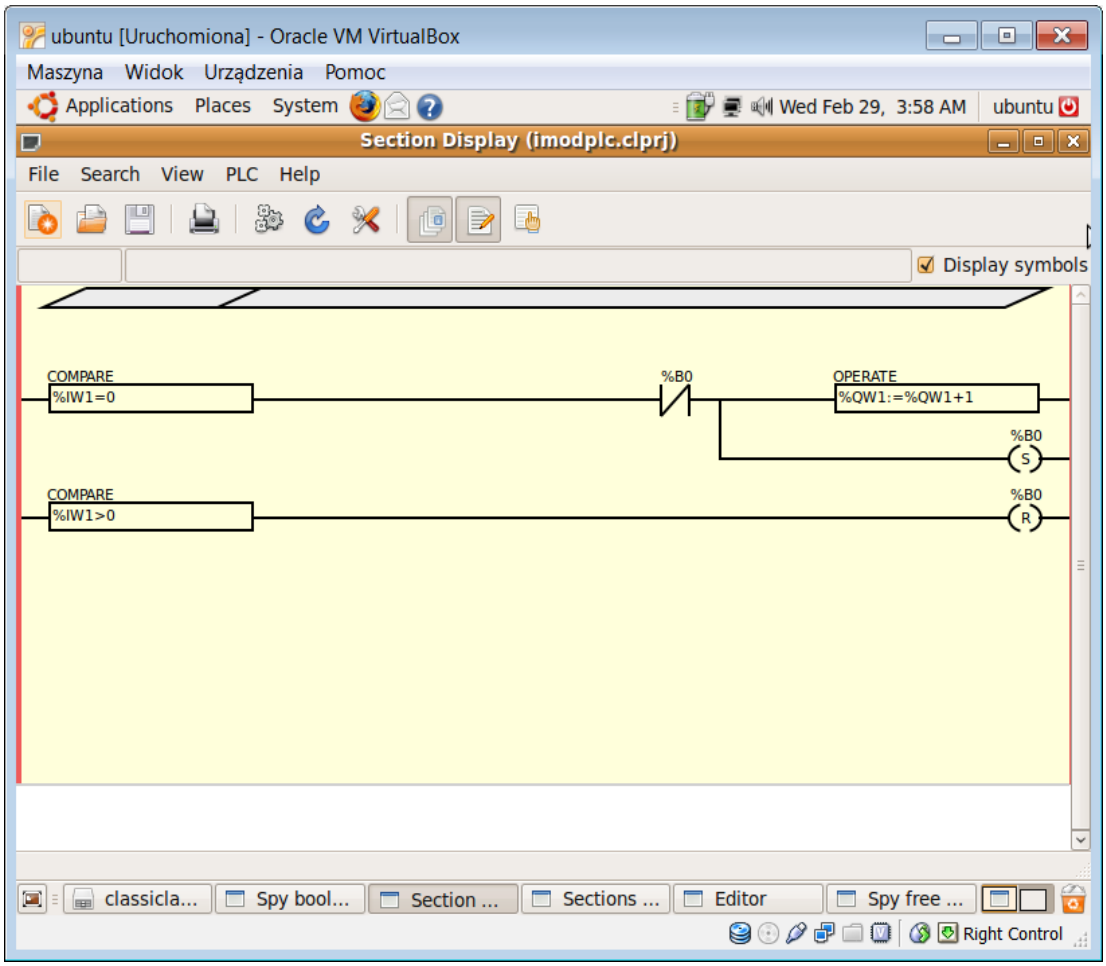
Two examples are described in the example configuration of the iMod telemetry module in the PLC controller working mode:

1. counting the USER_LED changes into high status, and
2. sending an e-mail after 10 USER_LED status changes.

1. Counting the USER_LED status changes

In this example the application counts the USER_LED changes and saves the amount of the changes in the iMod register.

LadderClassic structure and configuration



Two main variables are used in this application:

- IW1 – USER_LED readout
- QW1 – saving number of changes

and a single auxiliary variable:

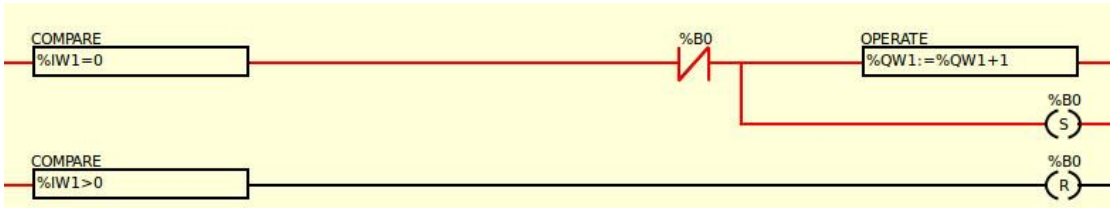
- B0

Function blocks are also used:

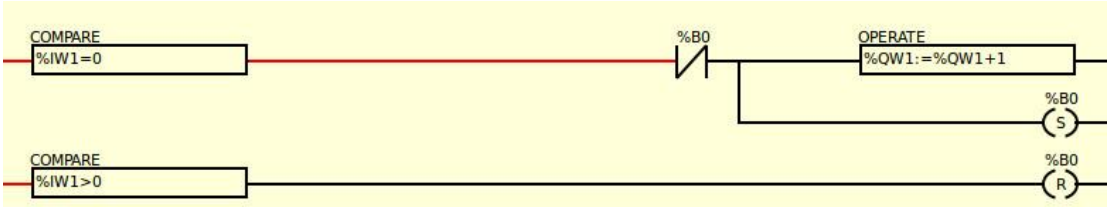
- Variable Comparison
- Variable Assignment
- Set Output
- Reset Output
- N.C. Input

Program Description

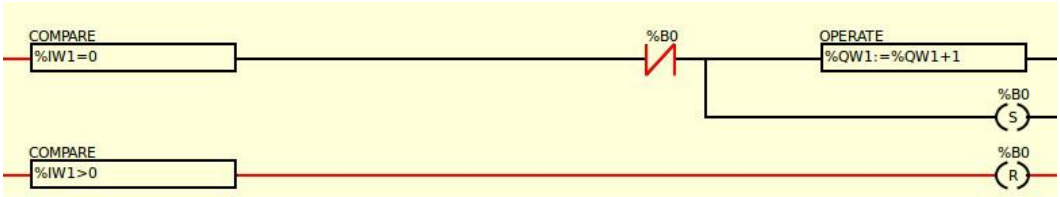
The iMod is set for storing the I/O states by default, this is why the application is set to count the USER_LED changes, when the LED takes the value of 0 (LED is off).



- At the moment of USER_LED state change into 0, the signal triggers with the COMPARE %IW1=0 function block, through the normally closed contact with the B0 variable and reaches the OPERATE %QW1:=%QW1+1 function block in the first place and then the SET coil with the B0 variable.



- Because of the OPERATE function block, the QW1 variable increases by 1. The SET coil changes the B0 variable state, which results in disconnection of the normally closed coil with the variable, so the signal does not reach the OPERATE function block. This is a protection, so the QW1 parameter is increased only by 1 point.



- At the moment of the USER_LED change into 1, the B0 variable is reset with the %IW1>0 function block, as a result the normally closed coil is closed

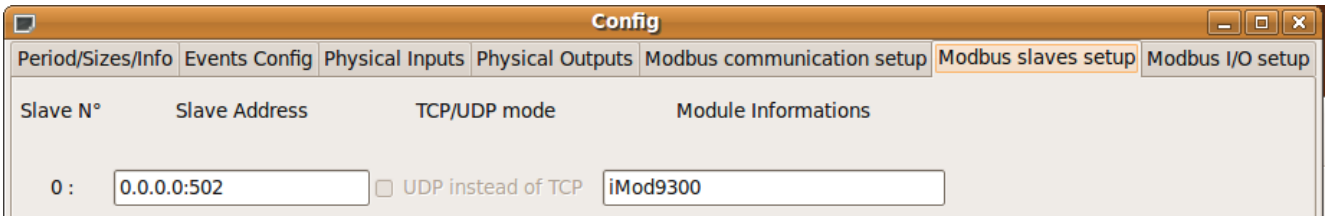
again and at the following change of the USER_LED into 0 it passes the signal.

Configuration

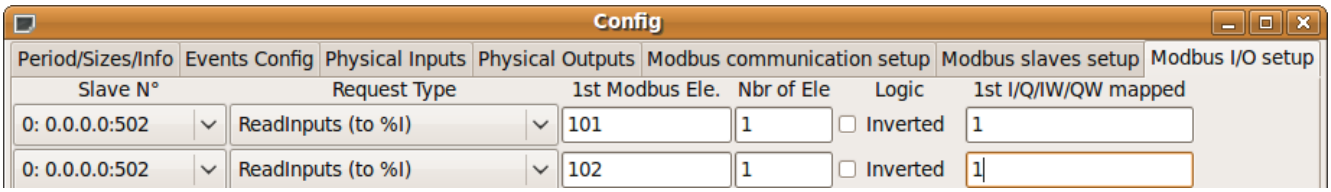
Modbus communication configuration
(*PLC → Configuration → Modbus communication setup*):

For the purpose of this example we connect with the device through Ethernet, this is why we leave the Modbus Master Serial Port empty, rest of the settings is default.

Modbus addressing configuration
(*PLC → Configuration → Modbus slaves setup*):



Modbus parameter configuration
(*PLC → Configuration → Modbus I/O setup*)



Pay attention to the modbus addressing. It often happens, that the address is shifted 1 place up or down. In this case, the addressing of parameters in ClassicLadder should be shifted by 1, according to the iMod.

iMod Structure and Configuration

Configuration of the iMod is created only through the *MainConfig.xml* file. First, you need to properly configure the access channels and parameter addressing.

Channel Definition

According to the modbus communication in ClassicLadder, the access-channel is created under the 0 modbus address of the device, through Ethernet on the 502 port. The iMod device is a source of parameters.

```
<?xml version="1.0" encoding="UTF-8"?>

<imod version="1.1.0">

<group name="Channel definitions">

        <access-channel name="Modbus_S1">
                <protocol name="MODBUS" />
                <port>"ET-502-TCP"</port>
                <property name="device-id" value="0"/>
        </access-channel>

        <source-channel name="NPE_io">
                <protocol name="HARDWARE"/>
                <gap>0</gap>
                <cycle>2</cycle>
        </source-channel>

</group>

</imod>
```

Parameter definition

Create two parameters: one for reading the USER_LED state values (id 100 parameter), second for saving the number of status changes (id 101 parameter).

```
<imod version="1.1.0">
 <group name="Channel definitions">

        <parameter>
                <id>"100"</id>
                  <description>"USER_LED"</description>
                <source-channel channel-name="NPE_io" parameter-id="USER_LED"/>
                  <access-channel channel-name="Modbus_S1" parameter-id="100">
                        <property name="varspace" value="DISCRETE"/>
                </access-channel>
        </parameter>

        <parameter>
                 <id>"counter"</id>
                 <access-channel channel-name="Modbus_S1" parameter-id="101"/>
        </parameter>

</group>

</imod>
```

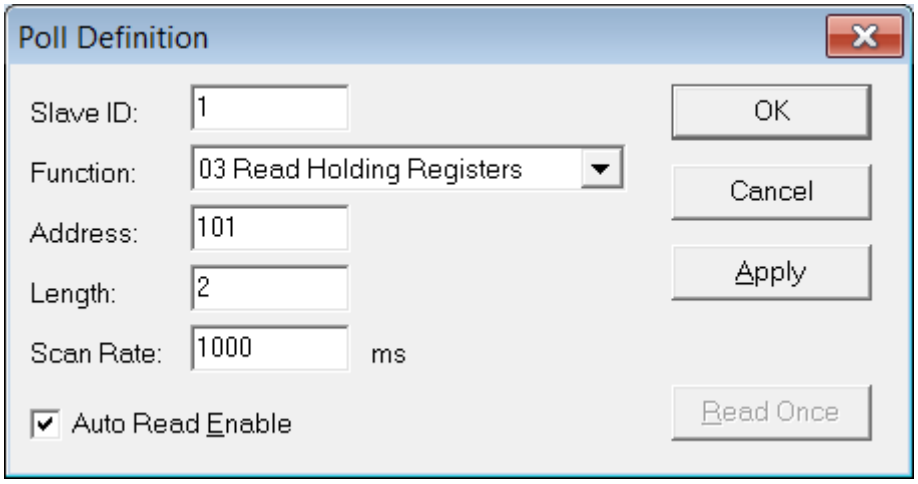
Verification of an example

In order to verify the example properly a modbus master application - Modbus Poll is used.

The Modbus Poll application is available at the address:
http://www.modbustools.com/modbus_poll.asp

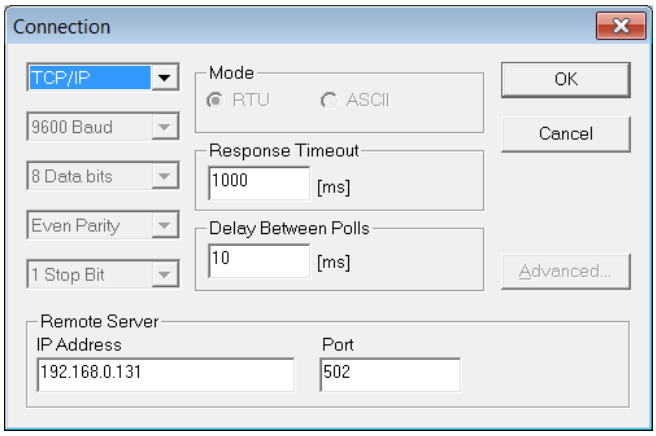
In addition, you need to upload the already mentioned configuration file and run the *arm-classicladder* application together with a project generated by ClassicLadder.

Establishing the connection



First, you need to set the parameters for the Modbus Poll application. In order to do so, go to the Poll Definition (*Setup -> Poll Definition*) window.

In this example we read the Slave device - iMod. iMod modbus address is 1. We use the 03 modbus function code, read the registers with the addresses: 100, 101 and 102, this is why we start to read from 101 and 3 following addresses (there is a modbus shift).

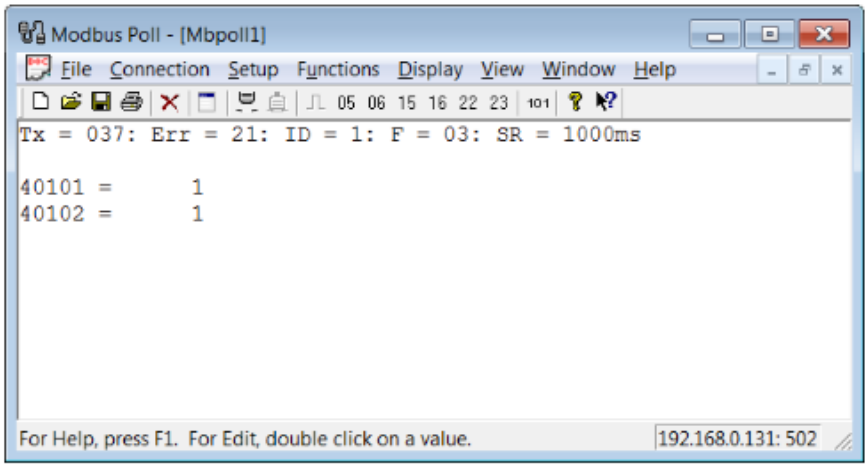


Next, connect with the iMod (*Connection -> Connect*).

In accordance with the configuration file, connect through the TCP/IP and the 502 port.

The default IP of the iMod telemetry module is:
192.168.0.101

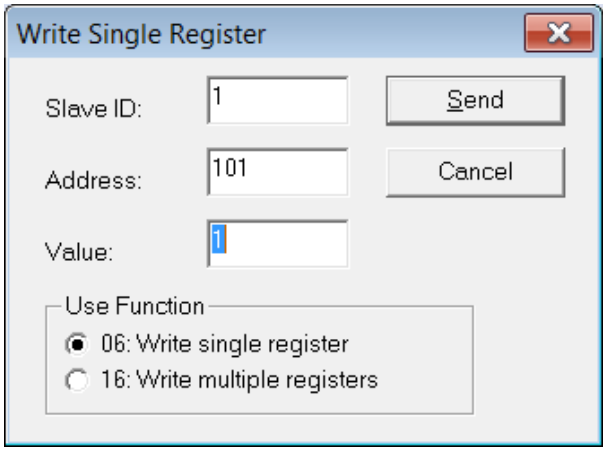
Verification of an example



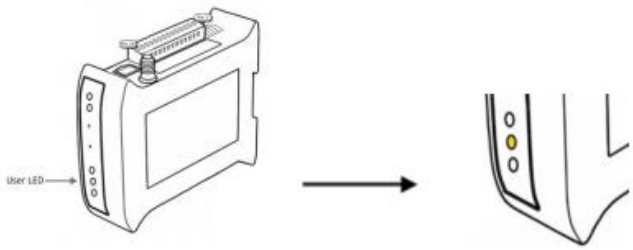
After a correct connection a following window should appear (the window may differ, depending on the Modbus Poll version).

Parameter description in the Modbus Poll window:

Address	Description
101	USER_LED
102	Counter - indicates the amount of the LED status changes

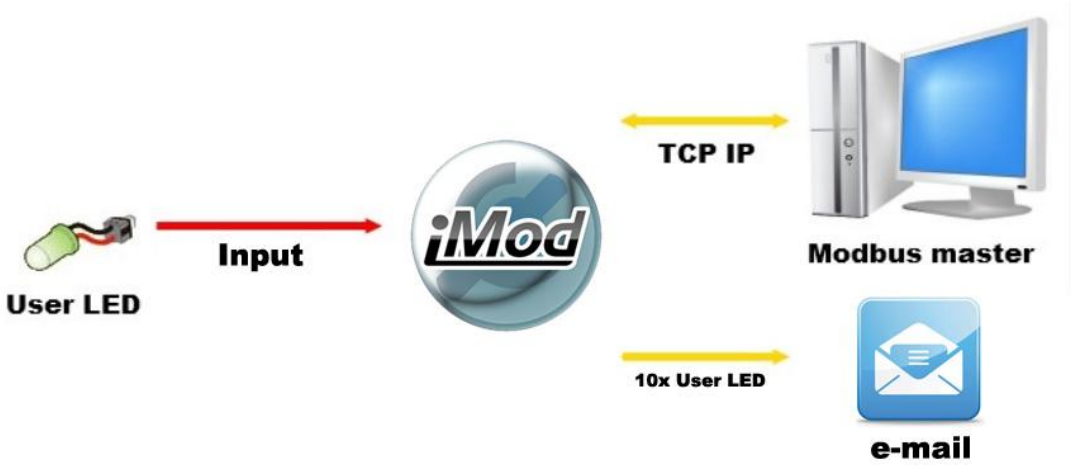


In order to change the value of a selected parameter, click twice the address or the presented value.



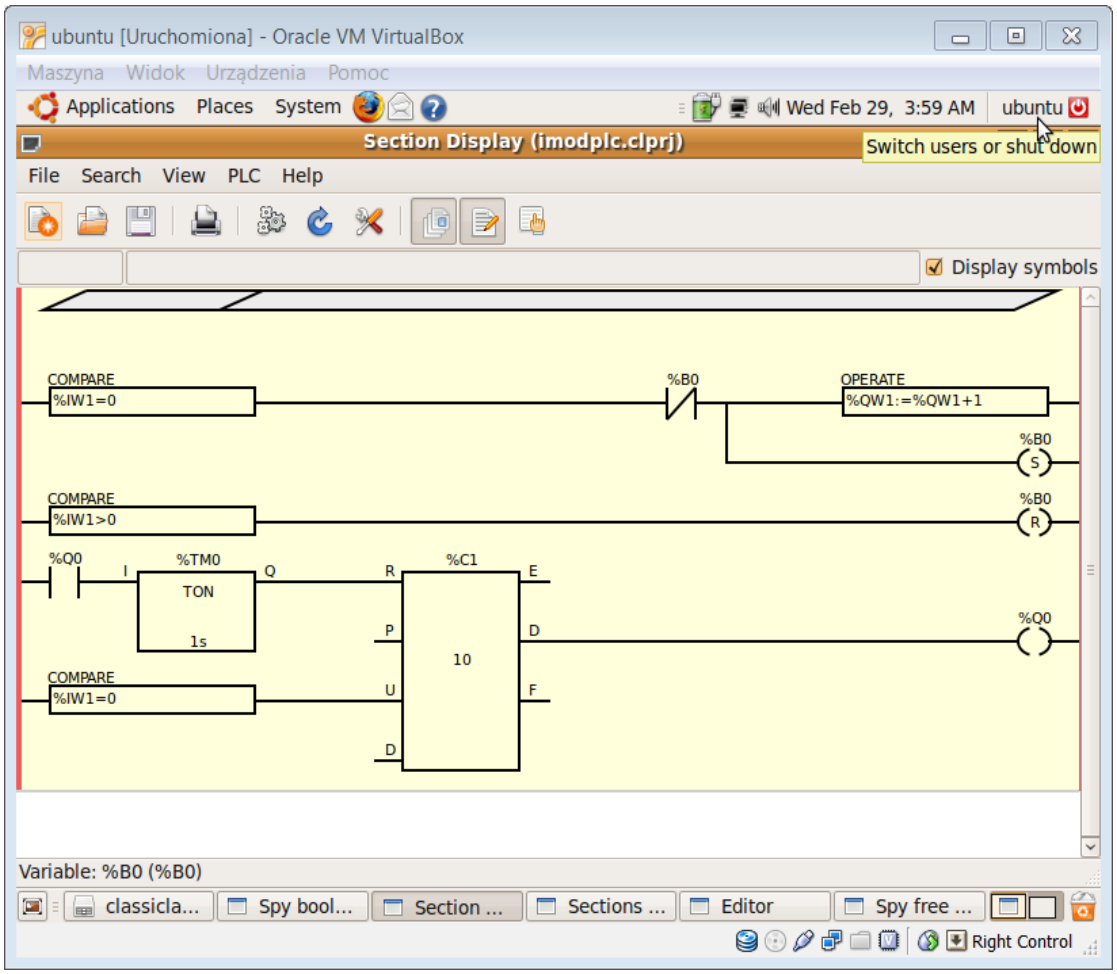
In order to change a parameter value, enter the value in the Value window and send it to the device with the Send button. The USER_LED status on the device should change and the 102 value should increase by one point.

2. E-mail message whenever the USER_LED status changes 10 times.



The application sends e-mail messages to the address entered in the configuration file, when the USER_LED changes 10 times. At the moment of sending the e-mail, the number of USER_LED changes is reset and the count starts one more time.

Classic Ladder Structure and Configuration

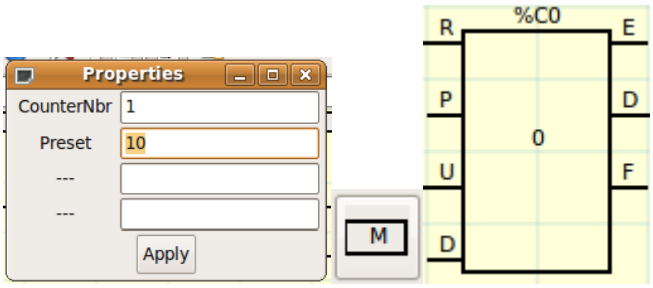


This application is an extended version of the application from the previous example, this is why all the parameters are the same. In addition to the variables presented in the previous example there is an additional one:

- Q0 – a variable responsible for sending an e-mail.

Program Description

This program counts the USER_LED changes. An e-mail is send to a previously defined recipient after 10 changes of the USER_LED status.

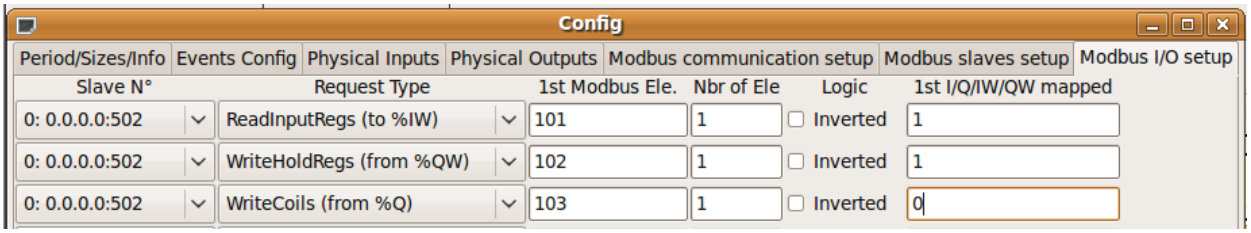


The Counter Block and the Timer IEC Block functions blocks are used. The counter block is set to start and send a signal to the %Q0 coil, when the counted number is more than 10.

1. When the USER_LED status is zero, the signal goes from the counter to the U connector, which increases the number by 1 (this is a one-time change and you do not need to use protection as before).
2. When the number of changes is 10, a signal is sent to the D output which launches the %Q0 coil. The Q0 parameter is responsible for sending an e-mail.
3. Launching of a high status resets the counter and the counting process starts one more time. However, before the reset, the high status goes to the timer which is set for one second.
4. Only after one second is the counter reset. If the counter reset is too fast, iMod might have difficulties with reading the Q0 parameter value and in consequence it might not send an e-mail.

Configuration

The modbus communication configuration and modbus addressing configuration is the same.
For the new modbus parameter configuration you need to add a new variable:
(*PLC → Configuration → Modbus I/O setup*)



Please look at the 1st mapped column - in this case 0 is set, because the Q0 variable starts from the "0", not "1" as before.

iMod Configuration Structure

The iMod configuration file is an extension of a file from the previous example. Add proper code lines to channels and parameters.

Channel Definition

Add the message channel to the channel definition. It is responsible for e-mail configuration and e-mail message content channel.

```
        <message-channel name="Email_sender">
                    <protocol name="EMAIL">
                            <property name="user" value="testnpe"/>
                            <property name="password" value="123npe"/>
                    </protocol>
                    <port>"poczta.o2.pl"</port>
                    <recipient>"info@a2s.pl"</recipient>
            </message-channel>

         <message id="Mess_1">
```

```
                <![CDATA[
                "USER_LED change state 10 times"
                ]]>
        </message>
```

Parameter definition

In order get the functionality to send an e-mail after every 10 e-mails sent, you need to add a parameter responsible for sending an e-mail after a status change.

```
<parameter>
        <id>"10"</id>
        <access-channel channel-name="Modbus_S1" parameter-id="102">
                <property name="varspace" value="COIL"/>
        </access-channel>
        <event type="OnChange">
                <message-channel channel-name="Email_sender"/>
                <message-id>"Mess_1"</message-id>
        </event>
</parameter>
```

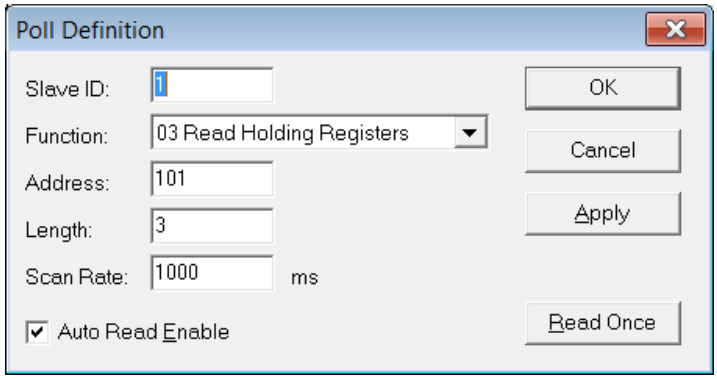
Verification of an example

In order to verify the example properly a modbus master application - Modbus Poll is used.

The Modbus Poll application is available at the address:
http://www.modbustools.com/modbus_poll.asp

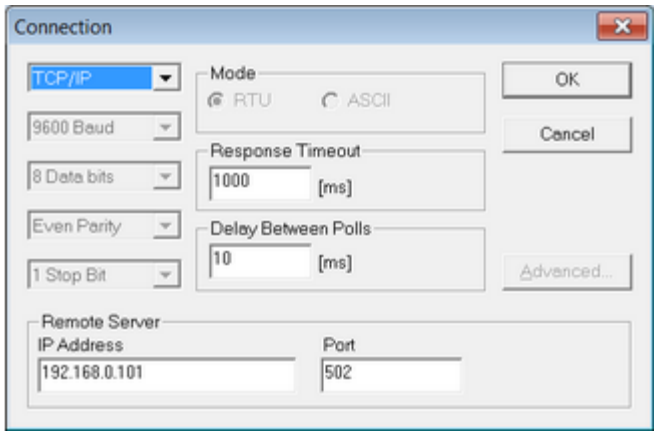
In addition, you need to upload the already mentioned configuration file and run the *arm-classicladder* application together with a project generated by ClassicLadder.

Establishing a connection



First, you need to set the parameters for the Modbus Poll application. In order to do that, go to the Poll Definition (*Setup -> Poll Definition*) window.

In this example we read the Slave device - iMod. Its modbus address is 0. Use the 03 modbus function code. The addresses for readout are registers with the addresses of 100, 101 and 102, so start reading the addresses from 101 - 103 (there is a modbus shift).

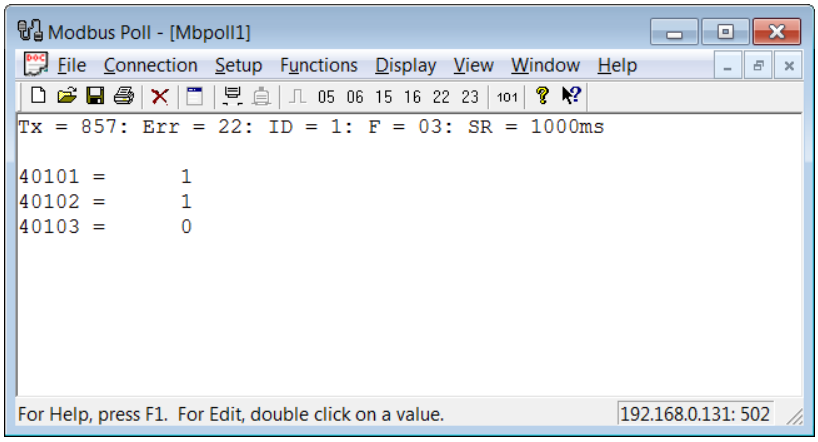


Next, connect with the iMod (*Connection -> Connect*).

In accordance with the configuration file, connect through the TCP/IP and through the 502 port.

The default IP of the iMod telemetry module is:
192.168.0.101

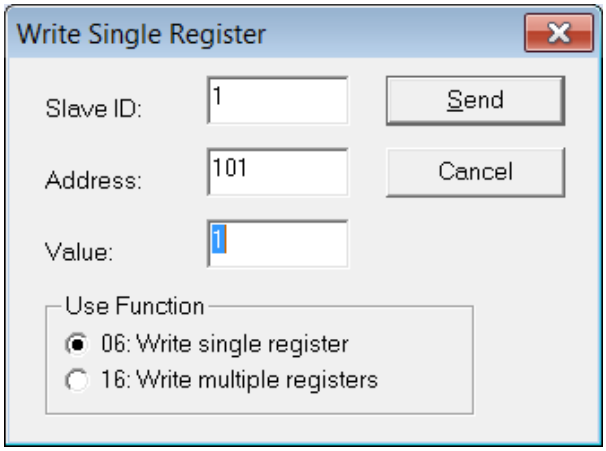
Verification of an example



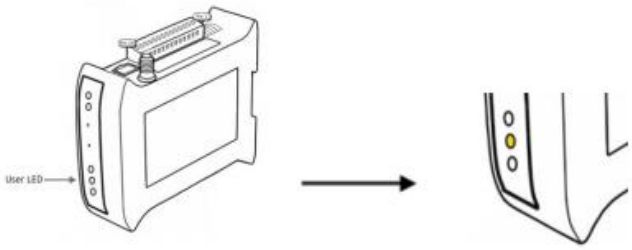
After a correct connection a following window should appear (the window may differ, depending on the Modbus Poll version).

Parameter description in the Modbus Poll window:

Address	Description
101	USER_LED
102	Counter - indicates the amount of the LED status changes
103	The counter - when the parameter reaches 10, an e-mail message is send and the value is reset

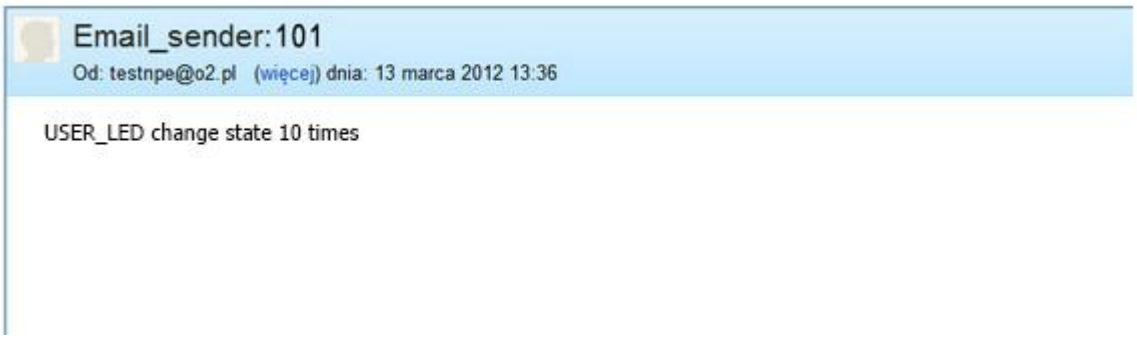


In order to change the value of a selected parameter, click twice the address or the presented value.



In order to change a parameter value, enter the value in the Value window and send it to the device with the Send button. The USER_LED status on the device should change and the 102 and 103 parameter value should increase by one point.

When the USER_LED status changes 10 times the iMod application sends and e-mail to a predefined address.


Email_sender:101
Od: testnpe@o2.pl (więcej) dnia: 13 marca 2012 13:36
USER_LED change state 10 times

Appendix A: Description of Classic Ladder function blocks

Symbol	Description	Function
	Normally Open contact	It conducts signal (closes contacts), when the logical value of an assigned variable is "1".
	Normally closed contact	Normally closed contact. It conducts signal (closes contacts), when the logical value of an assigned variable is "0".
	Rising edge contact	This contact sends an impulse when the status changes from open to close.
	Falling edge contact	This contact sends an impulse when the status changes from close to open.
	Horizontal connection line	Horizontal line connecting elements.
	Vertical connection line	Vertical line connecting elements.
	Horizontal connection line till the end	Creates a horizontal line till the end of the line.
	Timer function block	Timer, you can choose a name, value, constant and mode: • ON • OFF
	Meter function block	Meter R- Meter reset P- sets a set value so that there is 1 at the output U- counting up D on the left - counting down D on the right – when the counted value is equal or greater than the set value, the output = 1
	Comparison function block	You can use the mathematical comparison function with this block.

	Timer function block	Timer, counts a set time.
	Monostable block	This block sets the output value to “1” for a fixed time, and after that the output changes to “0”.
	Normally open coil	A relay sets the value of an assigned variable to “1”, when a signal is given. This is an open contact relay.
	Normally closed coil	A relay sets the value of an assigned variable to “0”, when a signal is given. It is a closed contact relay.
	Adjustable coil	The "SET” adjustable relay. The relay sets a variable to “1”.
	Reset coil	The "RESET" adjustable relay is suitable for operation with the “SET” relay. When you give a signal, the value of the assigned variable will be set to “0”.
	Jump coil	Jump on a selected line.
	Call coil	Call of a selected line.
	Operation	Possibility to perform mathematical operations.
	Indicator	Indicator