PiiGAB
PROCESSINFORMATION

Getting started MBus2Modbus

PiiGAB M-Bus Explorer & PiiGAB M-Bus 900

www.piigab.com

PiiGAB Processinformation i Göteborg AB • Anders Carlssons gata 7 • S-417 55 Göteborg • Sweden
Tel: +46(0)31 559977 • Fax: +46(0)31 227071 • email: info@piigab.se

Contents

1. Document Information

This document will describe how to prepare the files used for Modbus in PiiGAB Explorer for PiiGAB 900. The document will use the internal M-Bus meter inside the PiiGAB 900 and two external M-Bus meters to show how to convert M-Bus meters to Modbus. This document is only for a site where there is a Modbus client requesting data from the M-Bus meters. Not the other way around.

If you see something that is not correct in this document, that misleads you or if you are missing something please contact us so we can improve this document continuously. See contact information at the end of the document.

1.1 Versions

Version	Modified by	Detail
1.00.00	Stefan Eriksson	Initial version
1.00.01	Stefan Eriksson	Edited distributor contact information

2. Preconditions

Object	Detail/Other
One PiiGAB 900	IP-address set to 192.168.10.123
Connection with two M-Bus meters set to primary address 1 and 11 with PiiGAB M-Bus Setup Wizard	M-Bus meters supports EN13757
PiiGAB Explorer / M-Bus OPC-server	Version 2.00.00.000 or later
Getting started PiiGAB Explorer M-Bus	

3. Requirements

- PiiGAB Explorer

3.1 Optional requirements

- PiiGAB M-Bus Setup wizard version 3.1.0 or later

4. Steps to setup Modbus communication

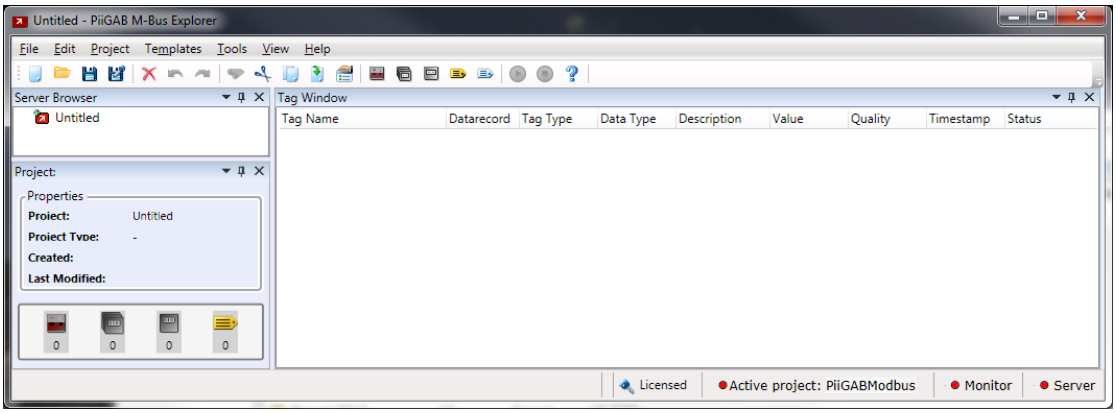
There are many steps to configure a PiiGAB 900 for Modbus communication. Many of these steps require pure M-Bus communication and configuration in PiiGAB Explorer. There are some tips and tricks to make the configuration easier.

Here is a list of points to have in mind while configuring for Modbus communication with a PiiGAB 900.

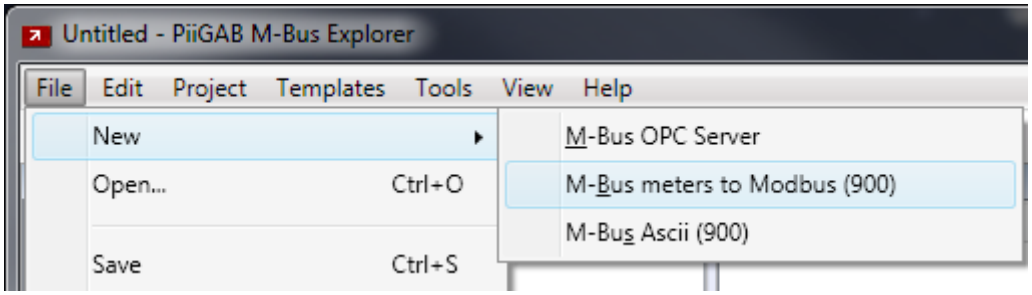
1. Make sure you have connection to the PiiGAB 900 and the M-Bus meters with PiiGAB Wizard. Please see *Getting started PiiGAB 900*
2. Your PiiGAB 900 must have Modbus present in its license
3. There is a great advantage if your PiiGAB 900 has two slave ports available:
   - Slave port 1 for M-Bus communication with PiiGAB Wizard and PiiGAB Explorer
   - Slave port 2 for Modbus communication with the Modbus client
4. Two projects opened in PiiGAB Explorer is a good way when configuring:
   - One M-Bus project to monitor the M-Bus meters through slave port 1
   - One Modbus project for Modbus configuration
5. The *Getting started PiiGAB Explorer M-Bus* is a great help to configure M-Bus
6. You should use the *Browse* template in PiiGAB Explorer to explore an M-Bus meter
7. Knowing how many M-Bus telegrams maximum you have to read from each M-Bus meter is mandatory
8. Knowing how long it will take to read the M-Bus meters is also mandatory to trim the timeout settings in the PiiGAB 900 and the Modbus client
9. Trimming the polling time in the Modbus client to handle how quick/slow the M-Bus meter can response back
10. Avoid long read-outs of M-Bus meters by reading them separately
11. The Modbus client inside the PiiGAB M-Bus Setup Wizard may be used to test the Modbus configuration

5. Create a Modbus project

1. Start PiiGAB Explorer

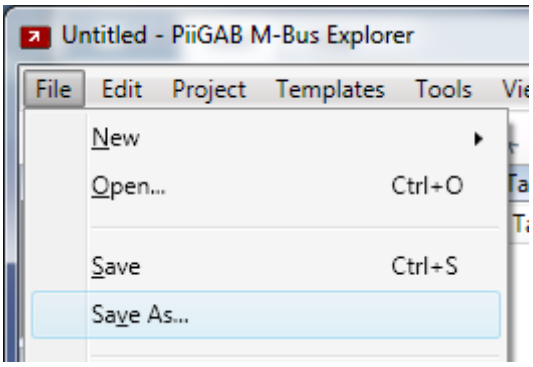


2. Go to *File* menu, select *New* and click on *M-Bus meters to Modbus (900)*



A Modbus project is created in PiiGAB Explorer.

3. Go to *File* menu, click on *Save As…*

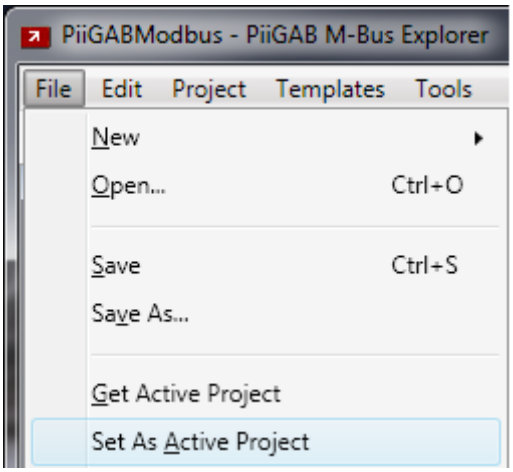


4. Save your project. This example saved the project as *PiiGABModbus*.

5.1 Set project as active project for the OPC-server - Optional

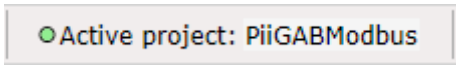
If you want to monitor the M-Bus meters in your Modbus project then you have to set the project as the active OPC-server project. Ignore this section if you use the monitor in another project, for example an M-Bus project.

1. Go to *File* menu and click on *Set As Active Project*



By setting the project as the active, the OPC-server will know which configuration file it will load when starting. You must do this if you want to monitor the OPC-tags in PiiGAB Explorer.

2. Make sure your project is the active project in the bottom right corner of PiiGAB Explorer

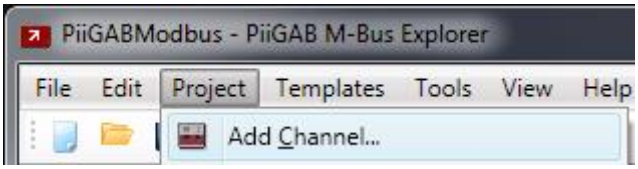


Note:
If you don't wish to monitor in your Modbus project, this section is optional.

5.2 Create a channel (M-Bus master)

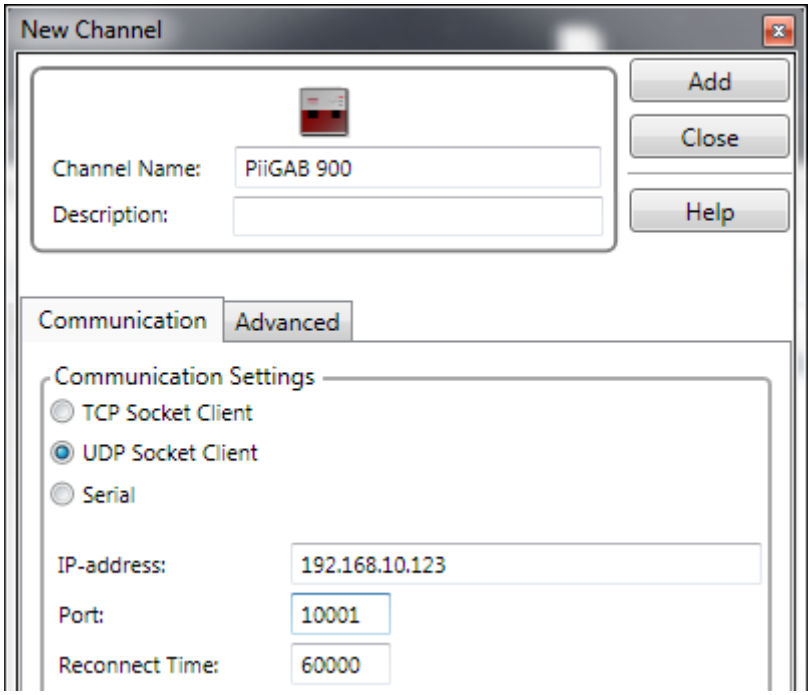
A channel in PiiGAB Explorer represents an M-Bus master. The channel needs the M-Bus master's communication parameters.

1. Go to *Project* menu and click on *Add Channel…*



You will see a window to configure the channel.

2. Configure the channel as specified in the picture below



3. Press *Add* to create the channel and add it into the project's tree view

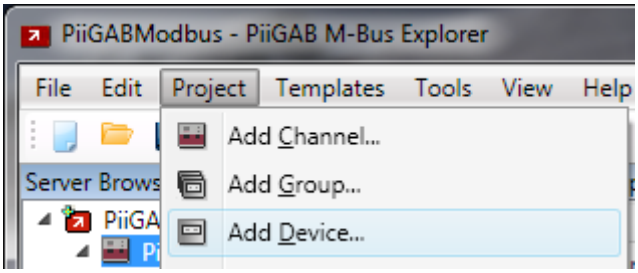
Note:
The channel's settings are usually the same as used in the PiiGAB M-Bus Setup Wizard. Your gateway may have another IP-address; change the configuration for your setup. If your gateway communicates is serial, choose serial settings instead.

This configuration expects that Slave port 1 in the PiiGAB 900 is configured for M-Bus communication on port 10001 with UDP protocol. Slave port 1 will be used to monitor tags in PiiGAB Explorer.

If you don't want to use the monitor function then the channel parameters are unnecessary.

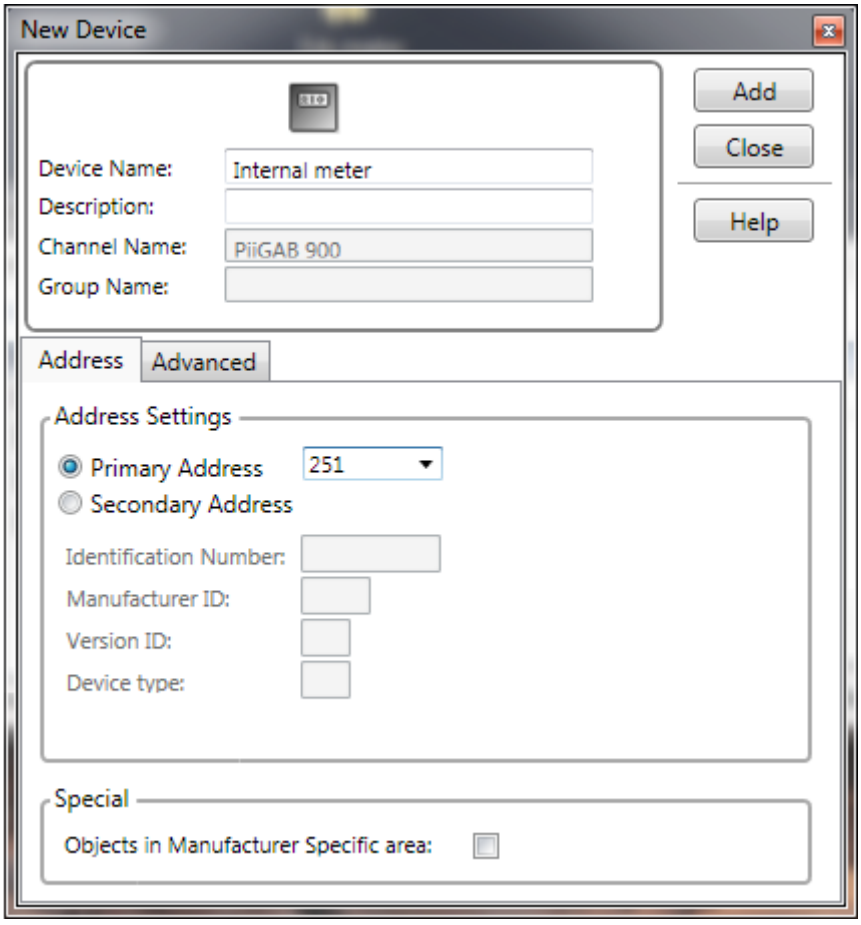
5.3 Create a meter for PiiGAB 900's internal meter

1. Go to the *Project* menu and click on *Add Device…*



Your will see a window to configure the device.

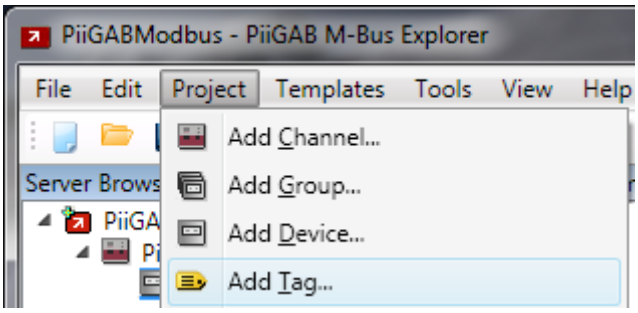
2. Configure the device as specified in the picture below



3. Press *Add* to create the device and add it into the project's tree view

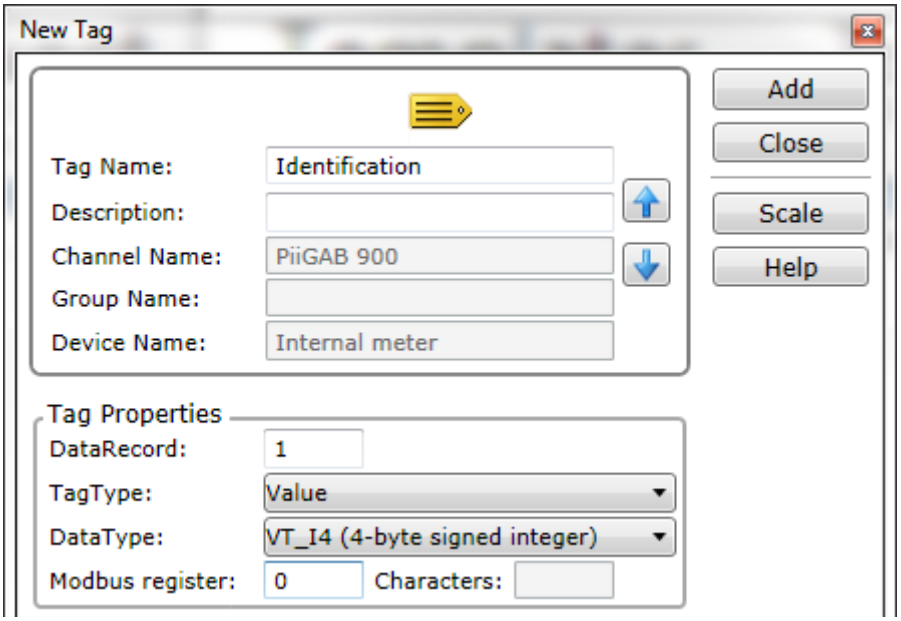
5.4 Create a tag for the internal meter with a Modbus register

1. Go to the *Project* menu and click on *Add Tag…*



Your will see a window to configure the tag.

2. Configure the tag as specified in the picture below



3. Press add to create the tag and add it into the meter's view

   This tag will read the PiiGAB 900's identification (serial number) as a 32-bit signed integer. The Modbus client will read the tag on Modbus register 0.

Repeat the steps in section 6.3 and 6.4 until you have created a configuration that suits your site. You may want to use *Getting started PiiGAB Explorer M-Bus* to find out what you can acquire from an M-Bus meter.

6. Setting Modbus registers to tags

Section 6 specifies how to set a Modbus register to a tag. Not knowing how to specify a Modbus register to a specific tag can be troublesome and cause problems. To know what Modbus register you shall specify to a tag depends on what previous Modbus register you recently specified and on what data type the recent tag had.

6.1 Find out how many Modbus registers a tag requires

The tag's data type specifies how many Modbus registers a tag requires. Below is a list of data types which explains how many Modbus registers are needed.

Data type M-Bus	Data type OPC-server	Modbus registers
INT8	VT_I2	1
INT16	VT_I2	1
INT24	VT_I4	2
INT32	VT_I4	2
REAL4	VT_R4	2
INT48	VT_R8	Not supported *
INT64	VT_BSTR	Not supported *
BCD2	VT_I2	1
BCD4	VT_I2	1
BCD6	VT_I4	2
BCD8	VT_I4	2
BCD12	VT_R8	Not supported *
	VT_BSTR	Depends on size **

* Please see section 8 to make a "work-around" for this with the *ValueSplit* method.

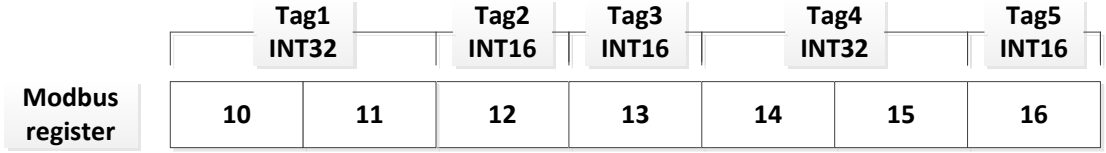
** There are only a few Modbus clients who support strings. The number of Modbus registers depends on the size of the string. Please see section 10 about strings.

6.2 Addressing Modbus registers to tags

To start addressing Modbus registers, please select the first Modbus 16 bit register.

1. Decide a start register for your first tag. In the examples below we are using register 10 as start register.
2. Calculate the next Modbus register based on the data type of the tag
3. Address the next tag with the next Modbus register
4. Continue with step 3 and 4 until all tags you want to address have an unique Modbus register

Here is a picture showing an example of different tags and what Modbus registers they start on and occupy.

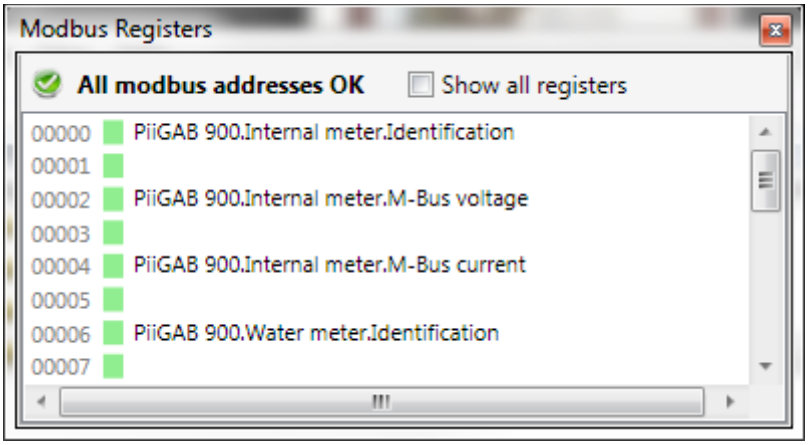


6.3 Avoiding overlapping Modbus registers

When addressing Modbus registers it's very important that Modbus registers don't overlap each other. Please use the Modbus register list window in PiiGAB Explorer to determine if Modbus registers overlap each other.

1. Go to the *Tools* menu and click on *Modbus Registers List…*

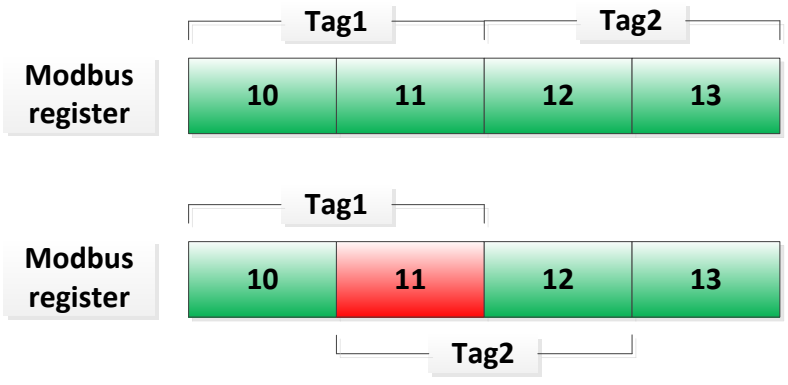
In the window there are red boxes if there is overlapping Modbus registers. If all boxes are green then there are no overlapping Modbus registers in the project.



6.3.1 Theoretic example

If a tag has INT32 specified as data type and starts on Modbus register 10, the next tag's Modbus register must start on register 12.

Data type INT32 occupies two Modbus register and therefore Modbus registers 10 and 11 are occupied by the first tag. Addressing the second tag to start on Modbus register 11 will overlap the first tag. Both the first and second tag will then share the same data at Modbus register 11 and may cause invalid values for the first tag and second tag.

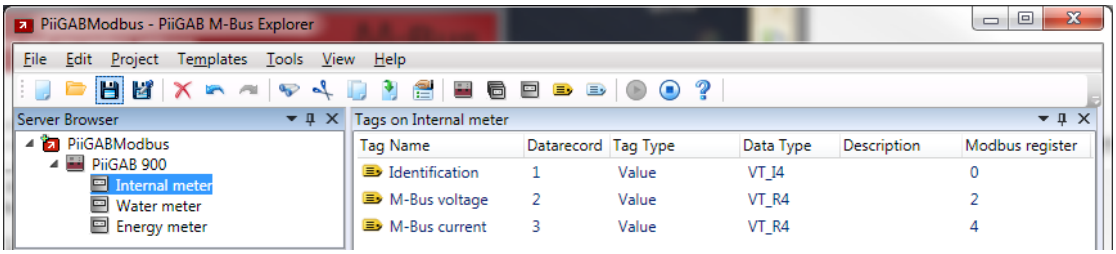


7. Simple Modbus configuration of M-Bus meters

This section will show a simple configuration of three M-Bus meters. Please see the *Getting started PiiGAB Explorer M-Bus* for more details on how to setup such a configuration as these three.

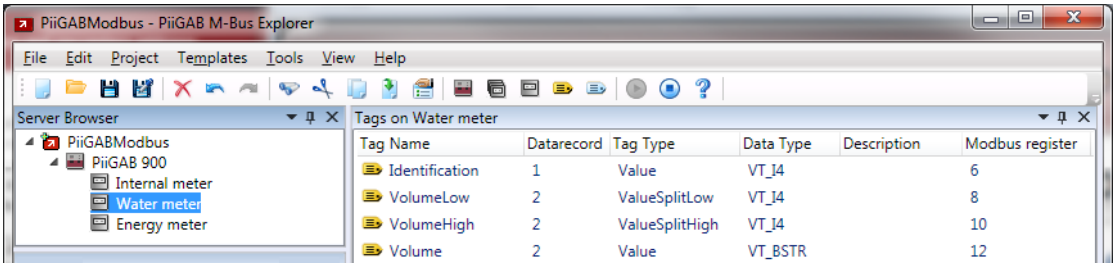
7.1 PiiGAB 900's internal meter

Object	Value
Primary address	251
Telegrams	1



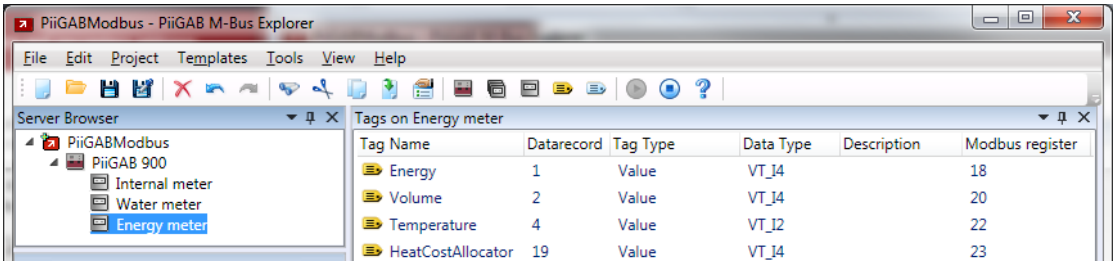
7.2 Water meter

Object	Value
Primary address	11
Telegrams	1



7.3 Energy meter

Object	Value
Primary address	1
Telegrams	3



8. Understanding ValueSplit for large data types

The Modbus protocol is normally limited to data types not greater than 32-bits. In M-Bus there are several data types which are greater than 32-bits. On occasions there will be objects with such data types that you wish to read over at your Modbus client. But since Modbus by default don't have any support for data types greater than 32-bit, there is no optimal solution. Instead there is "work-around" solution in PiiGAB Explorer and PiiGAB 900.

8.1 Find what objects data types are greater than 32-bits

Use the *Browse* template in PiiGAB Explorer to find out if there are any objects in an M-Bus meter which are greater than 32-bits. Please see *Getting started PiiGAB Explorer M-Bus* which has a detailed description of how to do this.

8.1.1 List of data types in M-Bus which is greater than 32-bits

Here is a list of the most common data types greater than 32-bits and are common in M-Bus.

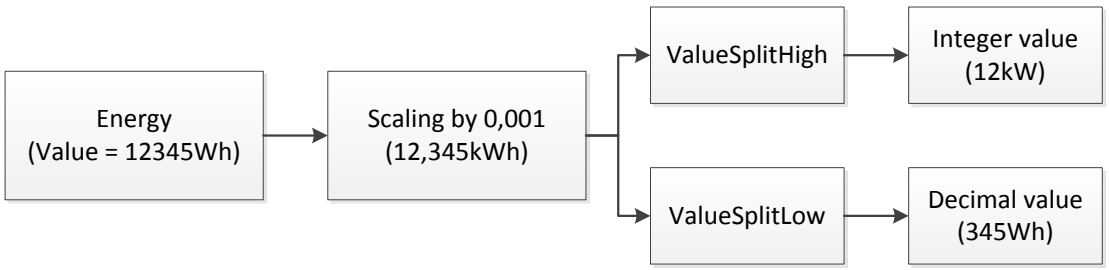
Data type	Bits	Bytes	Min value	Max value
BCD12	48 bits	6 bytes	0	999 999 999 999
INT48	48 bits	6 bytes	-140737488355328	140 737 488 355 327
INT64	64 bits	8 bytes	-9223372036854775808	9 223 372 036 854 775 807

8.2 Work-around data types greater than 32-bits

If there is an M-Bus meter with an object defined as *BCD12*, the maximum value for this data type is *999 999 999 999*. This value cannot fit inside a 32-bit data type. But if it's scaled by a factor of 0,001 the value will be *999 999 999,999.* The integer part of the value (999 999 999) is stored in one tag and the decimal part of the value (999) is stored in another tag. Both tags can now be handled as 32-bit data types. This can be handled with tags specified with the tag types *ValueSplitHigh* and *ValueSplitLow*.

8.2.1 Theoretic example

Imagine an M-Bus meter with an energy object defined as a BCD12 data type. The value of the object is 12345Wh. Scale the value with a ratio of 0,001 to get 12,345kWh. Create two tags in PiiGAB Explorer and scale them both with factor 0,001 and call them *EnergyHigh* and *EnergyLow*. Set *EnergyHigh's* tag type to *ValueSplitHigh* and *EnergyLow's* tag type to *ValueSplitLow*. The tags will then contain the values EnergyHigh 12kWh and EnergyLow 345Wh.

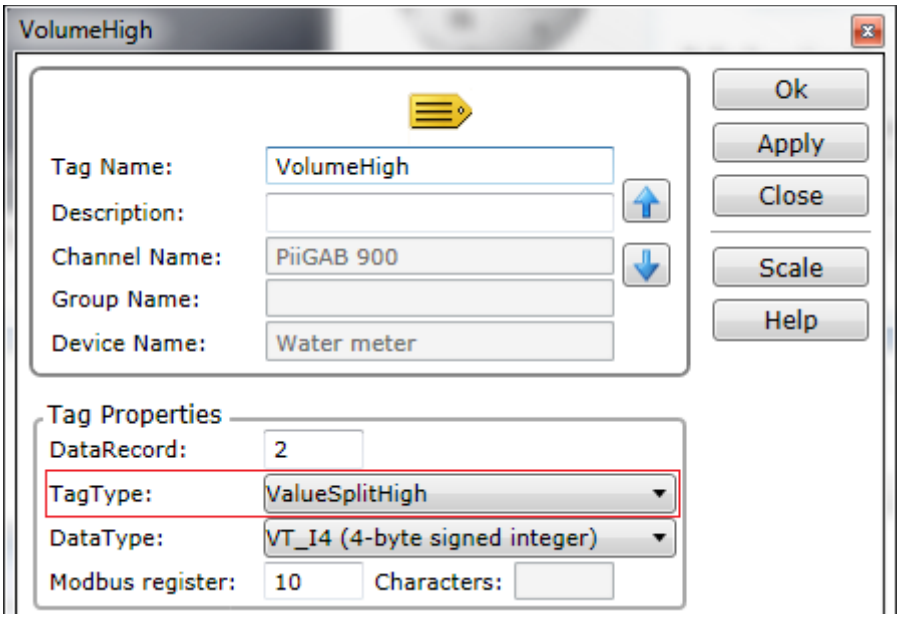


8.3 Assigning ValueSplitHigh and ValueSplitLow to a tag

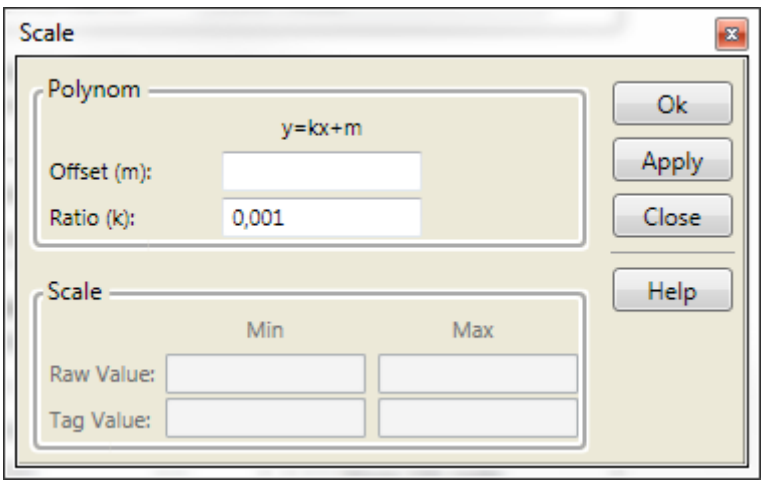
In section *7.2 Water meter* there are two tags which are set to *ValueSplitHigh* and *ValueSplitLow*.

Notice:
ValueSplit tag types can only be used if the PiiGAB Explorer is selected as a Modbus project.

1. Select or create any tag you want for ValueSplit
2. In the *TagType* field specify either *ValuesplitHigh* or *ValueSplitLow*



3. Click the *Add* or *OK* button to create or alter the tag
4. Optional - Click the *Scale* button



5. Specify the scaling you want to the tag
6. Press *OK*

9. Why M-Bus multi telegrams may cause problems

Those M-Bus meters that are multi telegram meters you must take in extra consideration. It is mandatory for you to specify how many telegrams you want to read from an M-Bus meters. Please see *Getting started PiiGAB Explorer M-Bus* to find out if a meter is a single or multi telegram meter.

Multi telegram meters may contain two or more telegrams. In some cases there can be up to 40+ telegrams. If you don't specify how many telegrams you want to read from an M-Bus meter, PiiGAB 900 will follow the M-Bus standard and read all telegrams in the M-Bus meter. This can, in worst case, take several seconds or tens of seconds to complete the reading. As a result this may cause the Modbus client to timeout.

9.1 Theoretic example

Imagine an M-Bus meter with 40 telegrams, each telegram takes about 750ms to read.

Telegram	**Time to read**
Telegram1	750ms
Telegram2	750ms
Telegram3	750ms
Telegram4	750ms
	…
Telegram15	750ms
Telegram16	750ms
Telegram17	750ms
Telegram18	750ms
Telegram19	750ms
	…
Telegram37	750ms
Telegram38	750ms
Telegram39	750ms
Telegram40	750ms
Total	**30000ms or 30s**

The master port in PiiGAB 900 will therefore be occupied for 30000ms reading just this M-Bus meter. The Modbus client must wait at least 30000ms for the response.

It's very unlikely that you need to read all 40+ telegrams from the M-Bus meters. You can specify how many telegrams you need to read for all M-Bus meters. Please see the *Getting started PiiGAB Explorer M-Bus* which explains how to do this.

It's more likely you only need to read the first, second and third telegrams - but properly only the first telegram. The time to read will then only be 750ms. This will cause the timeout for the Modbus client to decrease dramatically.

9.2 Find telegrams in an M-Bus meter and specify how many to read

The *Getting started PiiGAB Explorer M-Bus* describes the process of finding out how many telegrams it's necessary to read and how to specify that for an M-Bus meter. Section 7.*3 Energy meter* represents an M-Bus meter which contains many telegrams. Only three are useful to read. All other telegrams are useless and will only waste time and band width.

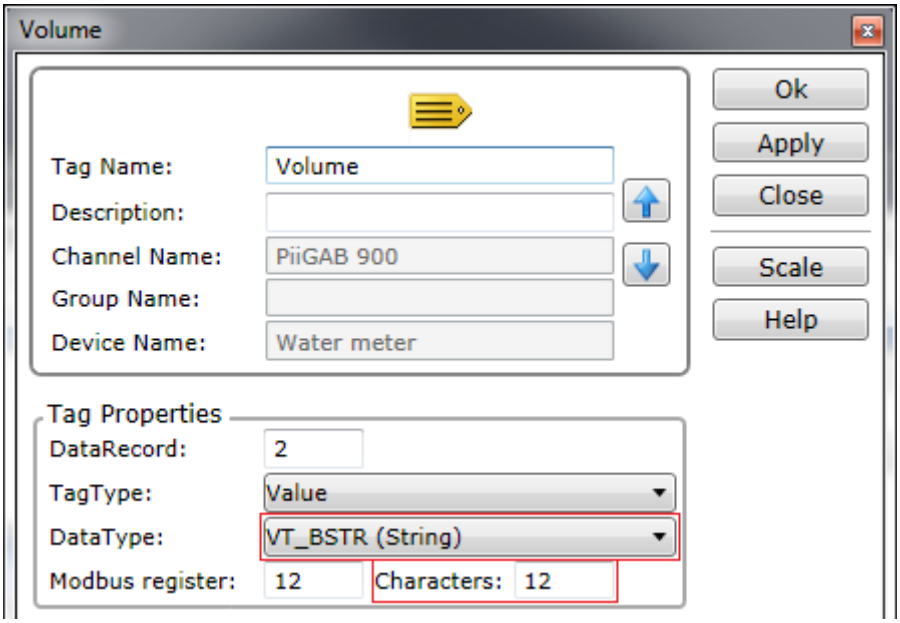
10. Using tags with string as data type

Few Modbus client support tags with string as data type. If your Modbus client supports strings and you don't want to use the *"ValueSplit"* solution on large data type such as *INT48*, *INT64* or *BCD12*. Then you can configure a tag as string and specify how many characters the tag contains. String data types may occupy more than two or four Modbus registers and therefore you have to watch out not to overlap Modbus registers.

10.1 Specify a tag as string

In the section *7.2 Water meter* there is the *Volume* tag which is defined as string. In the M-Bus meter this object is a BCD12 and will contain a maximum of 12 characters.

1. Select or create any tag you want for set as string
2. In the *DataType* field specify *VT_BSTR*
3. In the *Characters* field specify how many characters the tag can contain



4. Click the *Add* or *OK* button to create or alter the tag

This tag will occupy six Modbus registers. If it starts on Modbus register 12, then the next tag must start on Modbus register 18 to avoid overlapping Modbus registers. Please use the *Modbus Register List* window in PiiGAB Explorer to make sure you don't overlap Modbus registers.

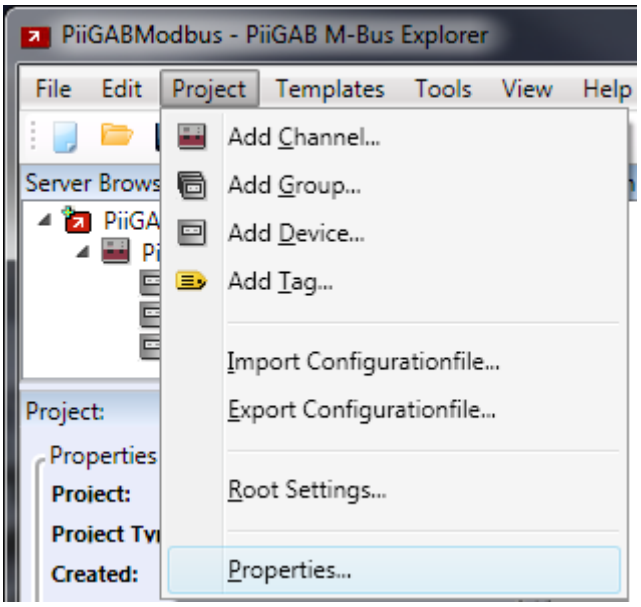
	Tag1 String[12]						Tag2 INT16
Modbus register	**12**	**13**	**14**	**15**	**16**	**17**	**18**

11. Configure the PiiGAB 900 for Modbus

When you have completed the configuring of your Modbus project you are ready to try the configuration in your PiiGAB 900. Hopefully you have checked with the Monitor function that all tags have a value that you expect. This is optional but it's well worth if you can use the Monitor function.

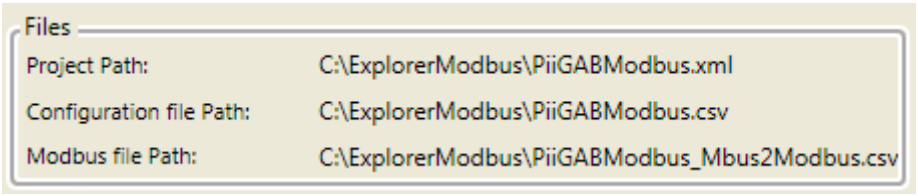
11.1 Finding the folders with the Modbus configuration files

1. Go to the *Project* menu and click on *Properties…*



   A new window will appear which will show some project data.

2. Look in the *Files* box which will display where the configuration files are located



   The location of the CSV-files specified by *Configuration file Path* and *Modbus file Path* are the two files that contain the Modbus configuration. These files must be uploaded into your PiiGAB 900 to allow Modbus communication with the Modbus client.

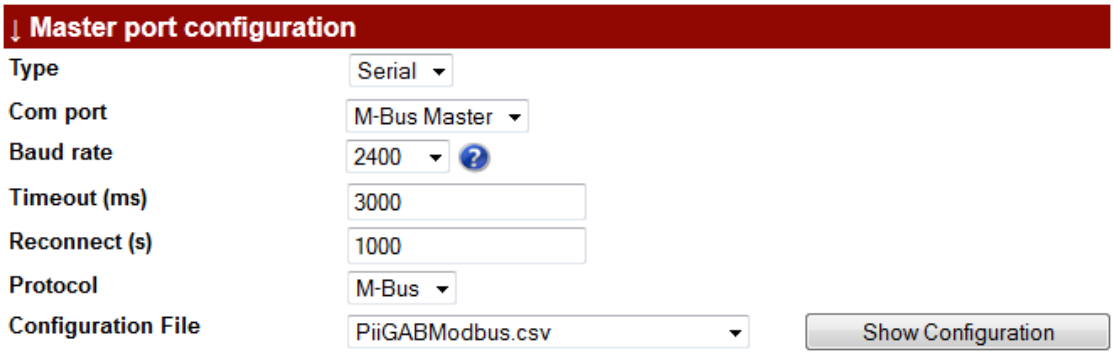
11.2 Upload the Modbus configuration into the PiiGAB 900

1. Open PiiGAB 900's web interface
2. Click on *Configuration*
3. Find the *Upload CSV-file* section in the configuration page

4. Press the browse button *(Bläddra…)* and browse to the one of the CSV-files at the location specified in section *11.1 step 2*
5. Press the *Upload* button to upload the file
6. Upload the other CSV-file as well

11.3 Configure the Master port

1. Click on the *Master port* tab in the configuration page
2. In the *Configuration File* field, specify the CSV-file for the Matser port. (Usually that file which doesn't have "Mbus2Modbus" in its name)



3. Press the *Save Settings* button

11.4 Configure the slave port for the Modbus client

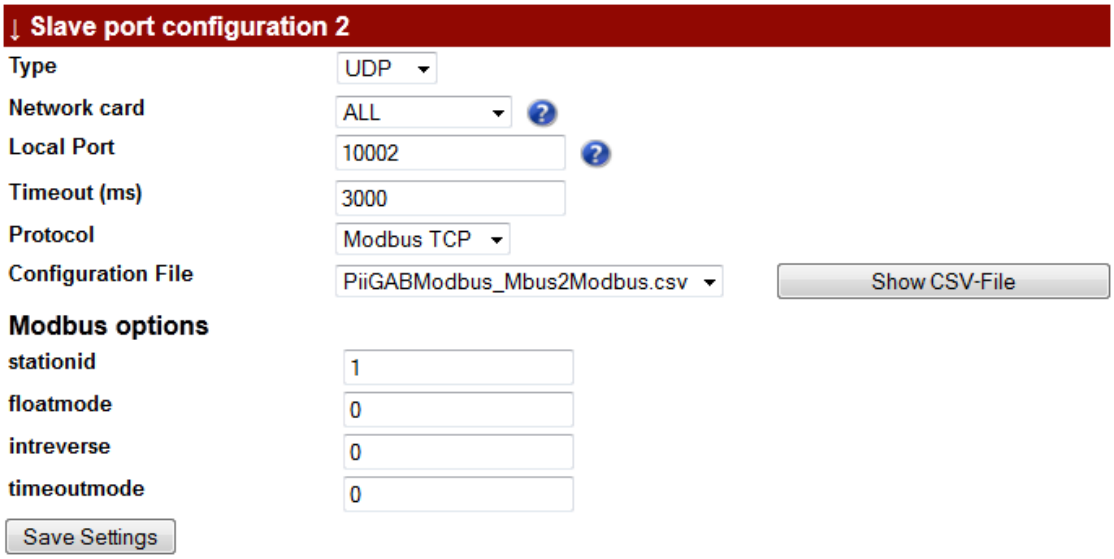
You are maybe already using one slave port for M-Bus communication with PiiGAB Explorer and PiiGAB M-Bus Setup Wizard. It's recommended that you leave that slave port and don't re-configure it. That slave port can be used with PiiGAB Explorer and PiiGAB Wizard for debugging and testing. If you have a slave port which is unused then please use it for the Modbus client.

1. Click on the *Slave port* tab in the configuration page, for example *Slave port 2*
2. Configure the slave port's parameters to communicate with the Modbus client
3. In the *Configuration File* field, specify the CSV-file for the Slave port. (Usually that file which has "Mbus2Modbus" in its name)
4. Press the *Save Settings* button

Both Master port and Slave port are now configured to handle Modbus requests from the Modbus client.

11.5 Configure the slave port for PiiGAB M-Bus Setup Wizard's Modbus client - Optional

1. Click on the *Slave port* tab in the configuration page, for example *Slave port 2*
2. Configure the slave port as the picture below



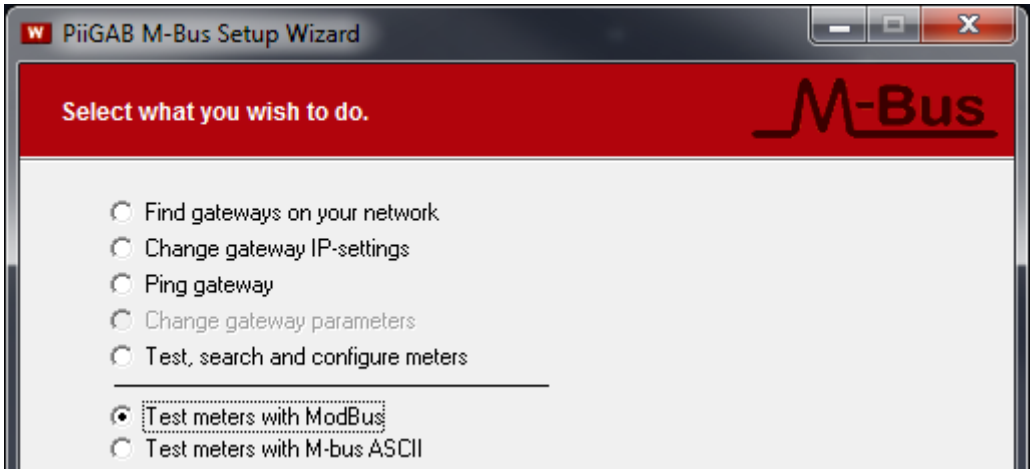
3. Press the *Save Settings* button

12. Test Modbus configuration with PiiGAB M-Bus Setup Wizard – Optional

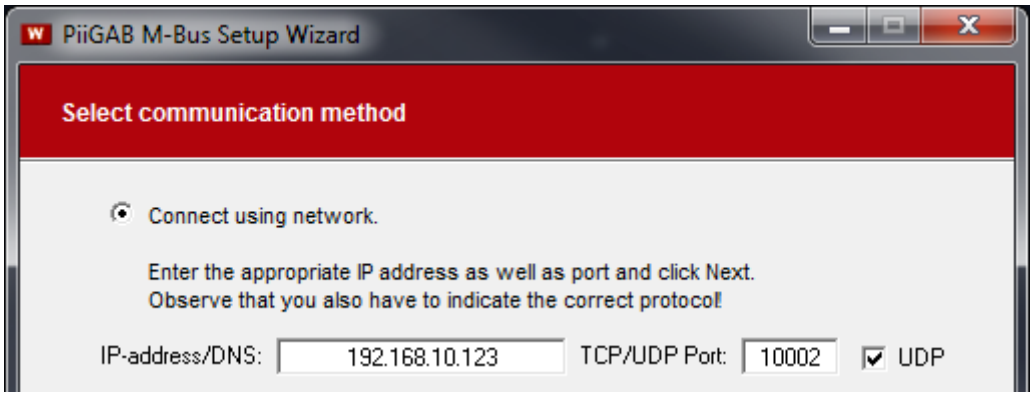
Before you test with the actual Modbus client, you may test the Modbus configuration with the built-in Modbus client in PiiGAB M-Bus Setup wizard. The Modbus client in PiiGAB Wizard is free and is a generic Modbus client not in any way bound to PiiGAB's hardware.

Note:
Please make sure you have configured a slave port for Modbus communication according to section 12.5.

1. Download and install the latest version of *PiiGAB M-Bus Setup Wizard* on PiiGAB's home page: www.piigab.com
2. Start PiiGAB Wizard and make sure you have at least version 3.1.0
3. In the main menu, select *Test meters with Modbus*



4. Press *Next* to continue
5. Select *Connect using network* and configure the connection as shown in the picture below



Note:
Your PiiGAB 900's IP-address may not be 192.168.10.123. Change to your PiiGAB 900's IP-address.

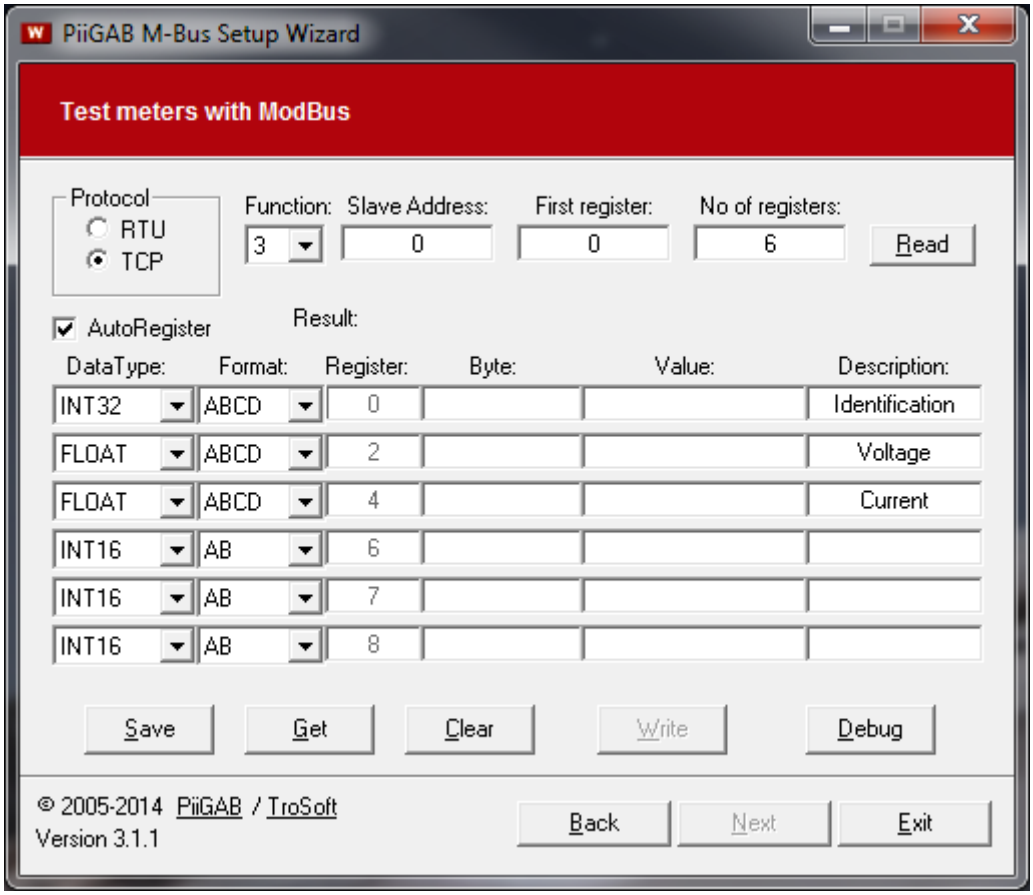
6. Press *Next* to continue

Note:
The following configurations will follow the configuration made in PiiGAB Explorer at section *8 Simple Modbus configuration of M-Bus meters*. All three M-Bus meters will be tested. You may have PiiGAB Explorer running with the Monitor function active to verify that you receive the same values both in PiiGAB Explorer and PiiGAB Wizard.

12.1 Test Modbus configuration with the PiiGAB 900's internal meter

This configuration of PiiGAB Wizard will read the internal meter inside the PiiGAB 900.

1. Configure PiiGAB Wizard as the picture below



2. Press the *Read* button to read the internal meter

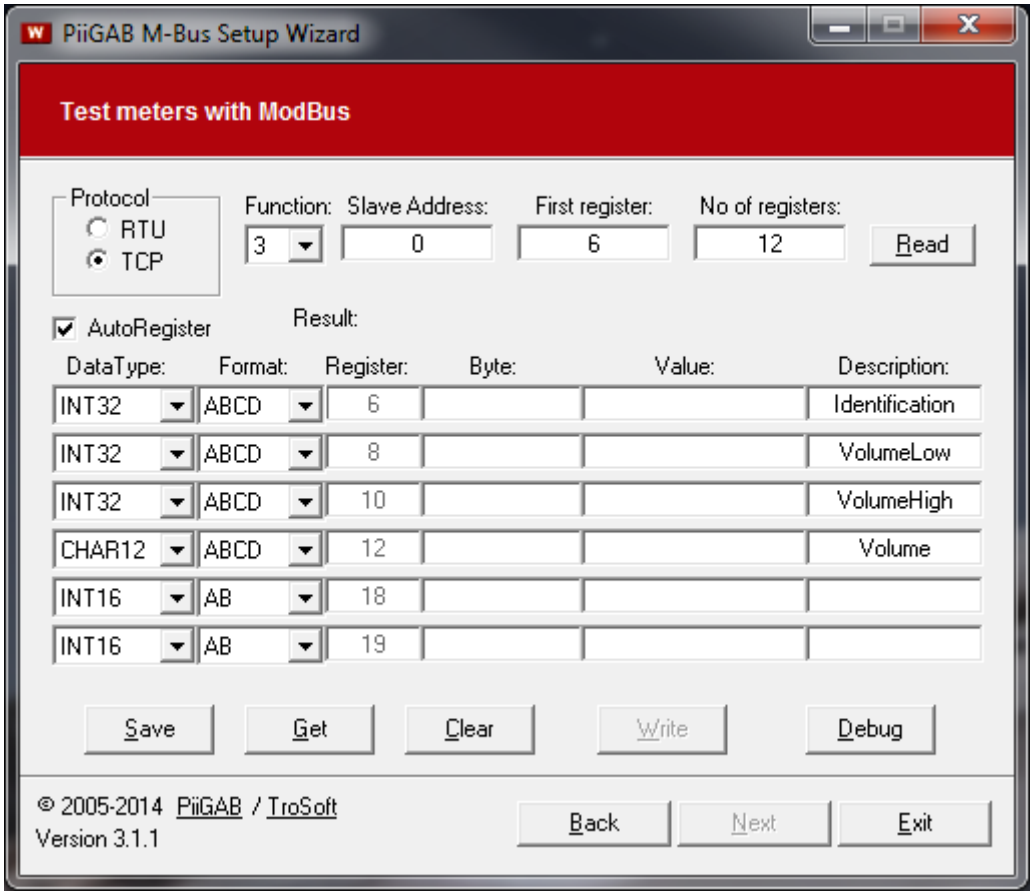
Here is the result of PiiGAB Wizard reading the internal meter over Modbus.

DataType:	Format:	Register:	Byte:	Value:	Description:
INT32	ABCD	0	0100 0090	16777360	Identification
FLOAT	ABCD	2	421E CCCD	39.700001	Voltage
FLOAT	ABCD	4	40AC CCCD	5.400000	Current

12.2 Test Modbus configuration with the water meter

This configuration of PiiGAB Wizard will read the external water meter connected to the PiiGAB 900.

1. Configure PiiGAB Wizard as the picture below



2. Press the *Read* button to read the internal meter

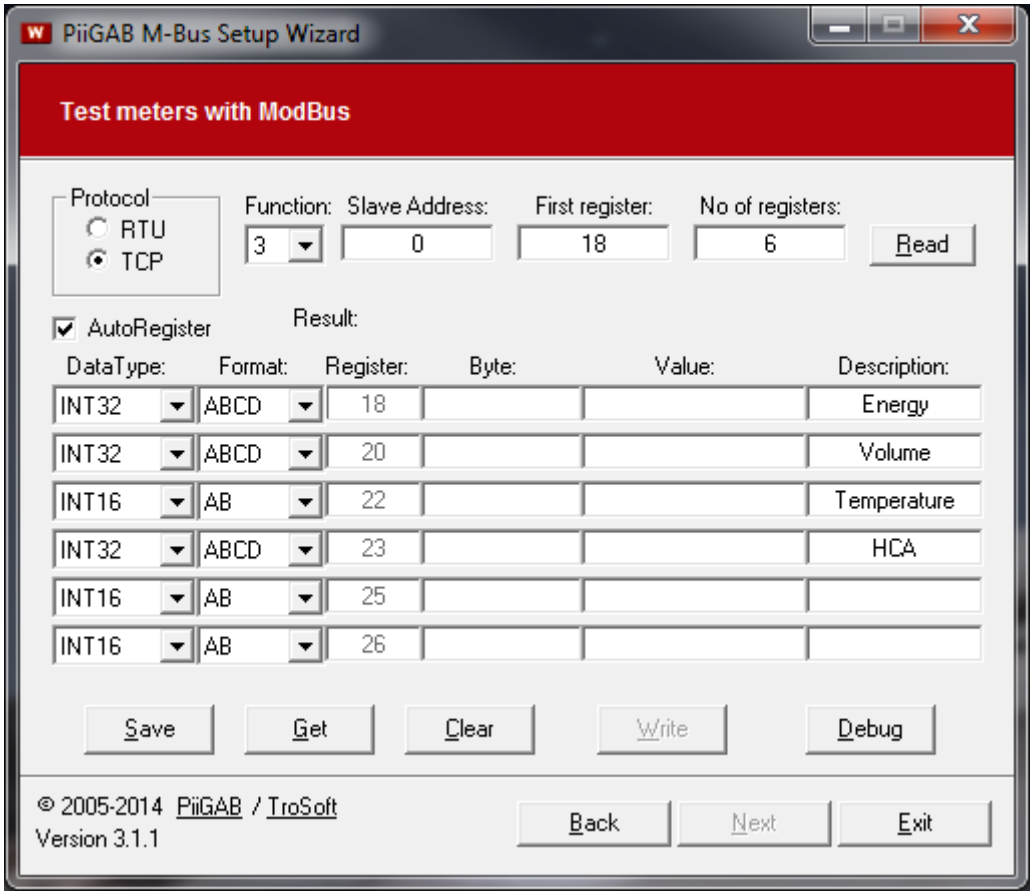
Here is the result of PiiGAB Wizard reading the water meter over Modbus. Notice that both tags with ValueSplit are read as INT32 and the entire volume object is also read as a 12 character string.

DataType:	Format:	Register:	Byte:	Value:	Description:
INT32	ABCD	6	03A0 0B5D	60820317	Identification
INT32	ABCD	8	0000 00C8	200	VolumeLow
INT32	ABCD	10	0000 0001	1	VolumeHigh
CHAR12	ABCD	12	2020 2020 202(	1200	Volume

12.3 Test Modbus configuration with the energy meter

This configuration of PiiGAB Wizard will read the external energy meter connected to the PiiGAB 900.

1. Configure PiiGAB Wizard as the picture below



2. Press the *Read* button to read the internal meter

Here is the result of PiiGAB Wizard reading the energy meter over Modbus.

DataType:	Format:	Register:	Byte:	Value:	Description:
INT32	ABCD	18	0000 0260	608	Energy
INT32	ABCD	20	0000 146E	5230	Volume
INT16	AB	22	00F3	243	Temperature
INT32	ABCD	23	0000	0	HCA

13. Timeout, polling time and M-Bus meters readout

To get a stable Modbus communication there are several parameters which might cause problems if they are incorrect or ignored.

13.1 Timeouts

The example with the *Energy meter* in section 8.3 describes an M-Bus meter with three telegrams which takes about 2.6 seconds in total to read. For this configuration this is the longest time where the PiiGAB 900 is occupied before it will return with a response to the Modbus client. The Modbus client's timeout must take this time into consideration.

13.1.1 Slave port timeout

The slave port's timeout is really simple to set. Since the slave port awaits a response from the Master port the slave port's timeout can be specified to 3000ms.

13.1.2 Modbus client's timeout

Since the Modbus client awaits a response from the slave port the Modbus client must await a minimum of 3000ms. But the Modbus client's must also take in consideration how long it will take to transport the response from the slave port back to the Modbus client. This time, which might in some cases can be ignored, but with serial communication this might cause some extra time to elapse before the response arrives. The Modbus client's timeout must handle this possible extra time. Try with 4000ms to start and adjust if necessary.

13.2 Polling time

Some Modbus clients are used to read slaves with extremely fast polling frequency. When you combine M-Bus meters to a Modbus site through a PiiGAB 900 you have to take in consideration how long it will take to read the M-Bus meters and what value the timeout is specified for the Modbus client. The *Energy meter* in section 8.3 has a total read-out time of 2.6 seconds. Requesting data from this M-Bus meter faster than 2.6 seconds is meaningless and will only cause problems. Also if your Modbus client's timeout is set to 4000ms (maybe to handle the Energy meter) then there is no need to request data faster than that timeout. You can also consider how often you shall acquire data from the M-Bus meters. Maybe you only need to read the M-bus meters once each minute, each hour or once each day. Avoiding extremely fast requesting time will make a more stable site. If you cannot control the polling time then your Modbus client might not be suited for your site.

13.3 Read M-Bus meters separately in your Modbus client

This document's example has three M-Bus meters. Requesting all three M-Bus meters from the Modbus client at the same time will cause higher response time then reading meter per meter.

M-Bus meter	Response time	Round up time
Internal	< 1000ms	1000ms
Water	< 1000ms	1000ms
Energy	About 2600ms	3000ms
Total	**4600ms**	**5000ms**

Requesting all M-Bus meter will take about 4600ms but can be rounded up to 5000ms. The Modbus client's timeout must be higher, for example 6000ms. Therefore you are better to request each M-Bus meter separately to avoid long read-out times.

14. Appendix

14.1 Contacts

PiiGAB Processinformation
Anders Carlssons gata 7
417 55 Göteborg
Sweden

Phone + 46 31 55 99 77
www.piigab.com

Distributors

Please contact our distributors in respective countries:

Germany

Relay GmbH
Stettiner Str. 38
33106 Paderborn
Germany

Phone +49 5251 17670
www.relay.de

Norway

Autic Systems AS
Stoltenbergs gate 48
3110 Tønsberg
Norway

Phone +47 33 30 09 50
www.autic.no

Czech Republic

Papouch store s.r.o.
Strasnicka 3165/1b
102 00 PRAGUE 10
Czech Republic

Phone +420 267 314 267-8
www.papouch.com