The MODBUS RTU/ASCII, MODBUS/TCP plugin
PRINTED MANUAL

MODBUS RTU/ASCII, MODBUS/TCP plugin

Printed: 8/3/2017

Publisher

AGG Software

Production

© 1999-2017 AGG Software
http://www.aggsoft.com

Table of Contents

1 Introduction

MODBUS is a serial communications protocol for use with its programmable logic controllers (PLCs). Simple and robust, it has since become a de facto standard communication protocol, and it is now a commonly available means of connecting industrial electronic devices.

Our MODBUS RTU/ASCII and MODBUS TCP plugin can work in two modes:

1. Data query and parser - the plugin sends MODBUS requests and process the responses. In this mode the program operates in the Master mode;
2. Data parser only - in this mode the plugin waits for responses only. For example, it can work in parallel with other MODBUS master or in the "Spy" mode.

This module has the following features:

- Can send valid data request to any MODBUS-compatible device;
- Can send request with any MODBUS function;
- CRC for each data packet will be calculated and verified automatically;
- Can request bytes, word, double words and single registers;
- Can poll MODBUS data by a custom interval;
- Can flexibly parse all received data packets and extract register's values.

2 System requirements

The following requirements must be met for "MODBUS RTU/ASCII, MODBUS/TCP" to be installed:

Operating system: Windows 2000 SP4 and above, including both x86 and x64 workstations and servers. A latest service pack for the corresponding OS is required.
Free disk space: Not less than 5 MB of free disk space is recommended.
Special access requirements: You should log on as a user with Administrator rights in order to install this module.

The main application (core) must be installed, for example, Advanced Serial Data Logger.

Notes for Microsoft Vista and above:

Since our software saves data to the registry and installs to the Program Files folder, the following requirements must be met:

1. You need Administrator rights to run and install our software
2. The shortcut icon of our software will be located on the desktop;
3. Windows Vista will ask for your confirmation to continue the installation.

NOTE: You can configure the user account only once in order not to see the above dialog box any more. Search Google for the solution of this problem.

3 Installing MODBUS RTU/ASCII, MODBUS/TCP

1. Close the main application (for example, Advanced Serial Data Logger) if it is running;
2. Copy the program to your hard drive;
3. Run the module installation file with a double click on the file name in Windows Explorer;
4. Follow the instructions of the installation software. Usually, it is enough just to click the "Next" button several times;
5. Start the main application. The name of the module will appear on the "Modules" tab of the "Settings" window if it is successfully installed.

If the module is compatible with the program, its name and version will be displayed in the module list. You can see examples of installed modules on fig.1-2. Some types of modules require additional configuration. To do it, just select a module from the list and click the "Setup" button next to the list. The configuration of the module is described below.

You can see some types of modules on the "Log file" tab. To configure such a module, you should select it from the "File type" list and click the "Advanced" button.

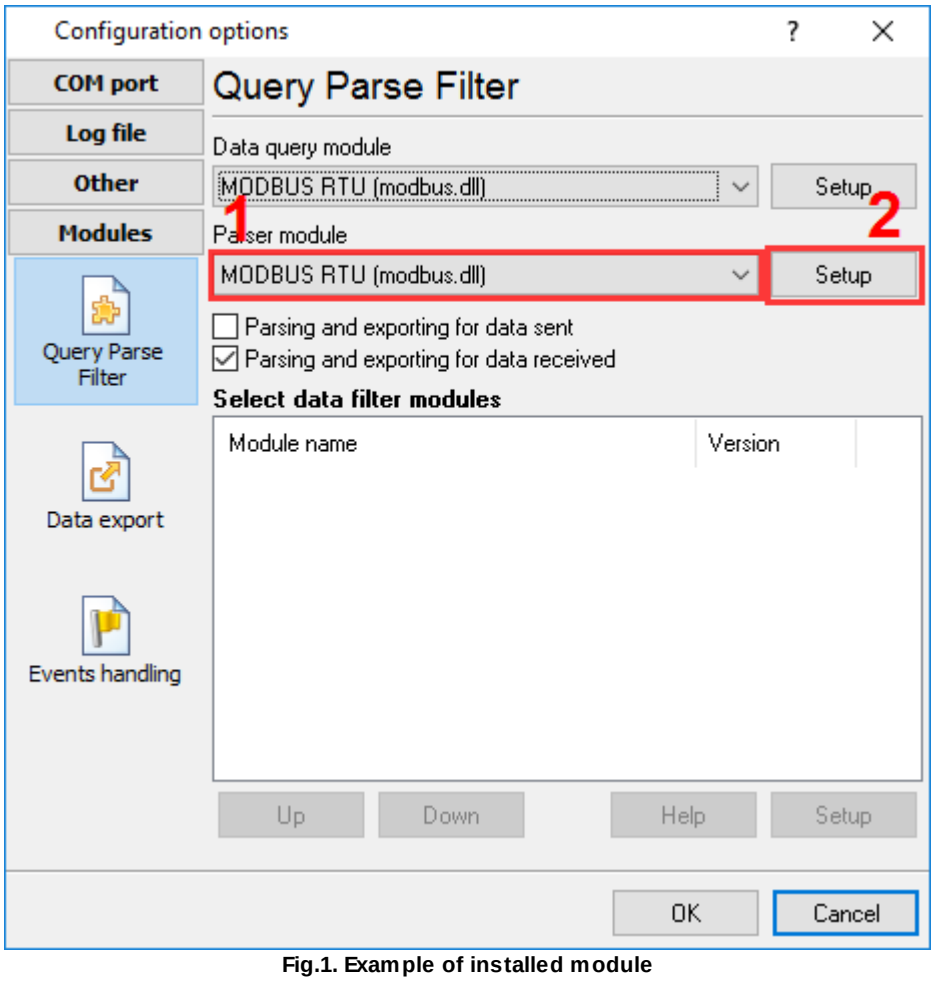

Fig.1. Example of installed module

4 Glossary

Plug-in - module

Main program – the program shell that uses this module. For example: Advanced Serial Data Logger

Parser – the module that processes the data flow singling out data packets from it and variables from data packets. These variables are used in data export modules after that.

Core - see "Main program".

5 User Manual

5.1 Data query

To add new item click "Actions->Add new request". The dialog window will be shown (fig.1). Enter a request description, that can contain any characters and click the "OK" button.

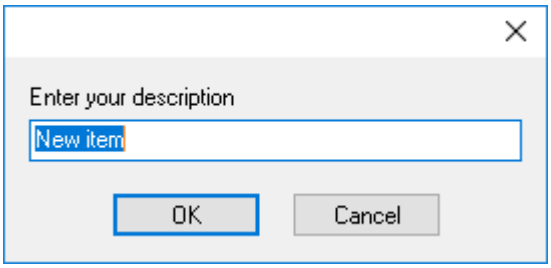

Fig.1. Name dialog

New MODBUS request will appear in the requests tree (fig.2). Each MODBUS request has few important options:

- **Device address** - your hardware device address in the RS232 or RS485 network. By MODBUS protocol specification this address can be from 0 to 255. If you'll specify 0 as a device address, then all devices in the network should answer to this request;
- **Function** - the MODBUS protocol function number. Usually, this value is 3 for reading holding registers or 4 for reading input registers;
- **First register** - it's register address. This value is zero-based (by MODBUS protocol specification). If you want to read a register with number 100, then specify 99 here;
- **Registers to read** - please, specify number of consistent registers in the device memory;
- **Request timeout** - It is the time interval for which the program is sending request to a MODBUS device. After reaching the timeout limit the program will automatically cancel current request and execute next request in the queue. The timeout value depends on the network on which master (program) and slave (device) is running. If the network is slow then timeout value should be larger and if network is fast then timeout value can be small.

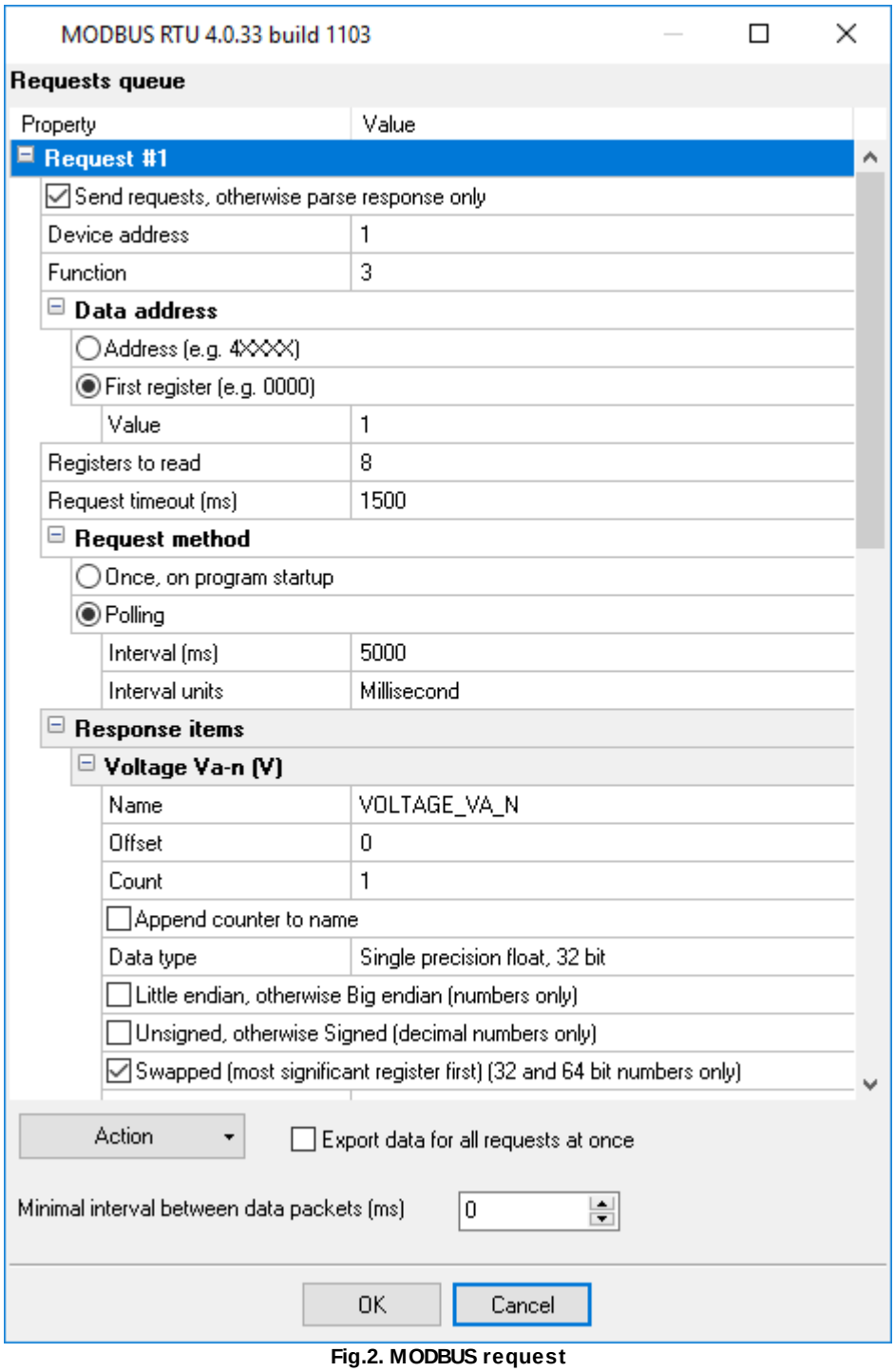

Fig.2. MODBUS request

5.2 Request method

The plugin can send requests in the following mode:

Once, on program startup - the program will send request once, when the program starts.

Polling - the program will be sending request periodically based on an interval specified. The interval between requests depends on the network on which master (program) and slave (device) is running. If the network is slow then time for each request will be larger and vice versa. Because, the program are executing all requests in the queue one by one, then time between requests depends on the number of requests in the queue.

At specified time - the time of the day using the 24hr format (e.g. 18:00:00). You may specify several time points separated by a semicolon (e.g. 11:00:00;11:20:00;11:40:00).

Time using Unix Cron schedule - a flexible schedule format that allows to send requests periodically or at the specified time. You can find the detailed information about this format and see examples in the "Cron time format" section. The default is 0 0 12 * * *, which means "every week, every day at 12:00:00".

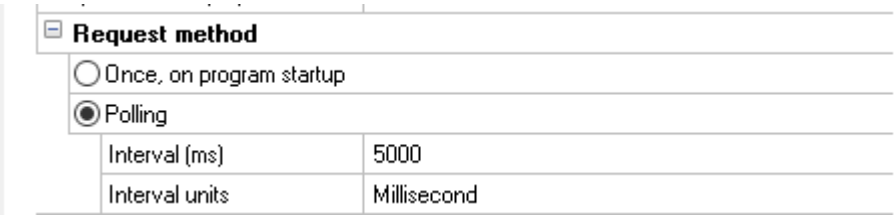

Fig.3. Request methods

If you added few requests to the queue, then you can move it up and down. Select a request title and execute a corresponding menu item by clicking the "Actions" button.
With help of this button you can change an item description and delete requests.

You can access all actions through the popup menu in the request tree.

5.3 "Cron" time format

The cron format is a simple yet powerful way to describe time and operation periodicity. The traditional (inherited from the Unix world) cron format consists of five fields separated with spaces:

<Second> <Minutes> <Hours> <Month days> <Months> <Weekdays>

Any of the five fields can contain the * (asterisk) character as its value. It stands for the entire range of possible values. For example, every minute, every hours and so on. You can also use the "nonstandard" ? character in the first four fields. See its description below.

Any field can contain a list of comma-separated values (for example, 1,3,7) or an interval (subrange) of values defined by a hyphen (for example, 1-5).

You can use the / character after the asterisk (*) character or after an interval to specify the value increment. For example, you can use 0-23/2 in the "Hours" field to specify that the operation should be carried out every two hours (old version analog: 0,2,4,6,8,10,12,14,16,18,20,22). The value */4 in the "Minutes" field means that the operations must be carried out every four minutes. 1-30/3 is the same as 1,4,7,10,13,16,19,22,25,28.

You can use three-word abbreviations in the "Months" (Jan, Feb, ... , Dec) and "Weekdays" (Mon, Tue, ... , Sun) fields instead of numbers.

Examples

Note: the <Second> field equal 0 in all examples

Format	Description
* * * * *	every minute
59 23 31 12 5	one minute before the end of the year if the last day in the year is Friday
59 23 31 Dec Fri	one minute before the end of the year if the last day in the year is Friday (one more variant)
45 17 7 6 *	every year on the 7th of June at 17:45
0,15,30,45 0,6,12,18 1,15,31 * 1-5 *	00:00, 00:15, 00:30, 00:45, 06:00, 06:15, 06:30, 06:45, 12:00, 12:15, 12:30, 12:45, 18:00, 18:15, 18:30, 18:45, if it is the 1st, 15th or 31st of any month and only on workdays
*/15 */6 1,15,31 * 1-5	00:00, 00:15, 00:30, 00:45, 06:00, 06:15, 06:30, 06:45, 12:00, 12:15, 12:30, 12:45, 18:00, 18:15, 18:30, 18:45, if it is the 1st, 15th or 31st of any month and only on workdays (one more variant)
0 12 * * 1-5 (0 12 * * Mon-Fri)	at noon on workdays
* * * 1,3,5,7,9,11 *	every minute in January, March, May, July, September and November
1,2,3,5,20-25,30-35,59 23 31 12 *	on the last day in the year at 23:01, 23:02, 23:03, 23:05, 23:20, 23:21, 23:22, 23:23, 23:24, 23:25, 23:30, 23:31, 23:32, 23:33, 23:34, 23:35, 23:59
0 9 1-7 * 1	on the first Monday of every month at 9 in the morning
0 0 1 * *	at midnight on the 1st of every month
* 0-11 * *	every minute till noon
* * * 1,2,3 *	every minute in January, February and March
* * * Jan,Feb,Mar *	every minute in January, February and March
0 0 * * *	every day at midnight
0 0 * * 3	every Wednesday at midnight

You can use the nonstandard "?" character in the first four fields of the cron format. It stands for the start time, i.e. the question mark will be replaced with the start time during the field processing: minute for the minute field, hour for the hours field, month day for the month day field and month for the month field.

For example, if you specify:

? ? * * *

the task will be run at the moment of startup and will continue being run at the same time (if the user does not restart the program again, of course) – the question marks are replaced with the time the program was started at. For example, if you start the program at 8:25, the questions marks will be replaced like this:

25 8 * * * *

Here are some more examples:

- ? ? ? ? * - run _only_ at startup;
- ? * * * * - run at startup (for example, at 10:15) and continue being run in exactly one hour: at 11:15, 12:15, 13:15 and so on;
- * ? * * * - run every minute during the startup hour;
- */5 ? * * * - run on the next day (if the cron is not restarted) at the same hour every minute and so on every day, once in five minutes, during the startup hour.

5.4 Data parser

All data publication modules uses variables parsed. The parser should pick out significant data blocks (data packets) from the common data flow. Our MODBUS RTU/ASCII, MODBUS/TCP module can do it. This module analyze data flow and control data packets integrity by CRC (cyclical redundancy check). All parser items assigned with a corresponding request in the queue. You can assign one or more parser items (variables) to one request. Typically, each request has one parser item.

You can add new parser item (variable) to the request by clicking "Actions -> Add response item". Before, you should select a caption of the corresponding request. New parser item (variable) will appear in the "Response items" group (fig.4).

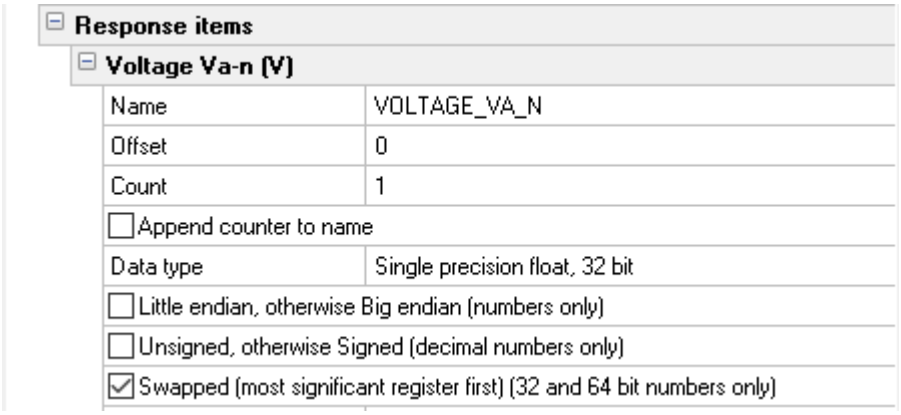

Fig.4. Data parser items.

Each response item has few important options:

- **Name** - the of the parser variable. This name you'll bind with fields in data publication modules.
- **Offset** - the device can response few data bytes, but you need only some of them. The "Offset" field contains a byte offset of the data from the beginning of the data block. This value is zero-based. If first byte of your value located at the begin of data block, then this value should be 0. You can specify -1 here, then the program will automatically calculate the value offset;
- **Count** - the number of values (nor bytes) with same parameters (data type and default value), that located one after another since the offset. If you specify more than one here, then a value index (1, 2, 3 etc) will be added to the parser item name;
- **Data type** - data type of the value. Each value can utilize one (for byte data type) or more bytes;
- **Default value** - this value will be used if the parser can't parser data block for this parser item. For example, if the data block has a small size or offset is too large.

6 Troubles?

6.1 Possible problems

No data for publication/exporting – no data is passed for exporting. Solution: configure the parser, make sure that one or more variables are declared in the parser.

Error on binding variable with name %s [%s] – the error usually occurs if data does not correspond to the specified format. For example, the date and time format does not correspond to the data.

Unable to disconnect from the database [%s] and **Unable to connect to a database [%s]** – it is impossible to connect/disconnect to/form the database. You should check the parameters of the database connection. The analysis of the additional information will help you locate the error.

Database access error [%s]. Stop operations with the database? – the message appears if an error occurs during an attempt to execute an SQL query if the second variant of reacting to errors is selected. The message implies a "Yes" or "No" answer. The analysis of the additional information will help you locate the error.

Unable to verify your SQL script [%s] – the message appears when an attempt to analyze your SQL query fails. Check if the syntax of your SQL query is correct.

Tested successfully – the message appears if your database connection is successfully tested. It requires no additional actions.

Database isn't used – the message appears if the module is temporarily disabled (the "Temporarily disabled" check box is selected) or the database name field is empty. Check the connection parameters.

Database isn't selected - the message appears if the database type is not selected. Check the connection parameters.

Database: %s – %s contains the database name. The message appears if the database connection is successful. Usually, you see it when you call the module for the first time. It requires no additional actions.

Invalid data block length (columns=%d,length=%d) – an internal application error. It means that the data sent by the parser is in an invalid format. Perhaps, you are using the module incompatible with the version of the Advanced Serial Data Logger kernel. Update the versions of both the kernel and the module.

The time of connection is not due yet (%d,%d) – the message appears during an attempt to connect to the database after the connection to it has been lost and the "Reconnect after" option is enabled. No additional actions are required.

Invalid procedure call. Bad arguments –an attempt to call the module using invalid parameters. Perhaps, you are using the module incompatible with the version of the Advanced Serial Data Logger kernel. Update the versions of both the kernel and the module.

Writing to the database is complete - the message appears if your queue of SQL queries is successfully executed. It requires no additional actions.

Writing to the database is complete with errors – the message appears if the executing your queue of SQL queries was interrupted by an error. It requires no additional actions.

Your SQL is empty. Please, specify some SQL text first – the message appears if you do not enter the text for your SQL query. Check if the options on the "SQL queue" tab are configured correctly.

Invalid temporary path – the path to the temporary file specified by you does not exist. Enter a new path in the "Temporary folder" field on the "Errors handling" tab.

%s, %d – will be replaced by additional information.