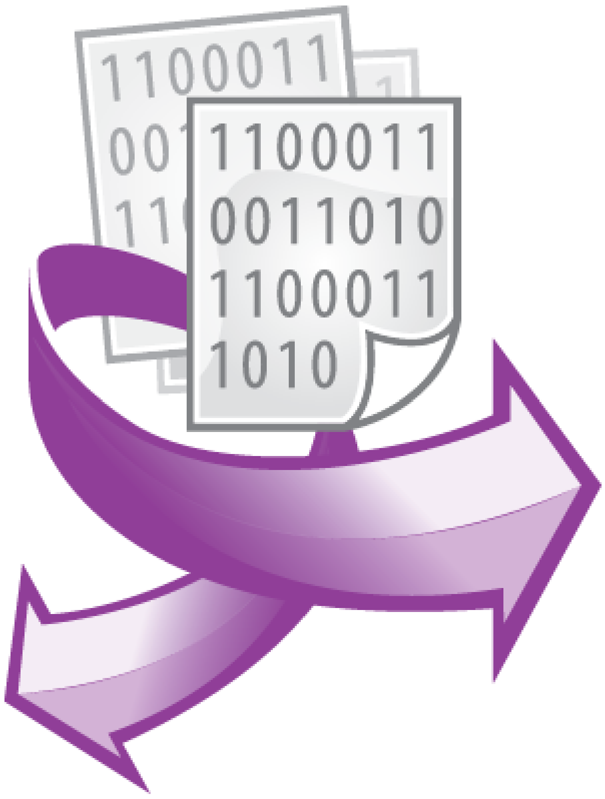


The M-Bus plugin
PRINTED MANUAL

M-Bus plugin

Printed: 11/2/2024

Publisher

AGG Software

Production

© 1999-<%YEAR% AGG Software
http://www.aggsoft.com

Table of Contents

1 About

The M-Bus plugin allows you to read and log data from the corresponding devices. M-Bus ("Meter-Bus", do not mess with MODBUS) is the European standard for building distributed data collection systems and commercial energy metering (heat, water, gas, electricity, etc.).

One of the main advantages of this data exchange protocol is the possibility to retrieve the full list of all metering points and their values, including units of measurement. If allows you to connect different equipment and does not require developing specialized software only for a single device type. Another advantage is the ability to connect many devices on one M-Bus communication line (more than 250 devices, unlike MODBUS)

Our M-Bus plugin has the following features and capabilities:

- You can configure a list of requests to devices to read certain values at a specified time.
- You can read both instant and archive data, if any.
- It is possible to read all available data, as well as individual values with their units of measurement.
- The module can automatically search for all devices on the network by primary and secondary address.
- You can poll multiple devices at the same time.
- The plugin automatically calculates and verifies a checksum for all data packets to control data integrity.

2 Data request

This plugin allows you to define one or more requests, and specify individual parameters for each request. This way, you can implement a request queue to read multiple values or poll multiple devices.

To add a new request, click the "Action - Add Request" button. The following dialog box will be displayed (fig. 1). Enter a description for the request, which can contain any characters, and click the "OK" button.

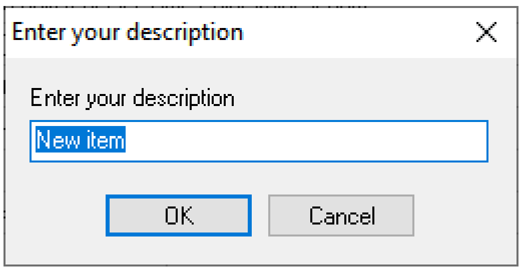


Fig. 1. Request name dialog.

A new M-Bus request will appear in the request list (fig. 2). Each request has several important options that you should configure:

Send requests - if this checkbox is disabled, the corresponding request is temporarily suspended. Usually, this option is always on.

Device address - this field specifies a network address of your device in the M-Bus network. You can specify as a short primary (primary) address (from 0 to 252), or a long secondary address, which is usually the same as the device's serial number. If you specify 254 as the device address, all devices on the network will respond to this request. This allows you to get the secondary address of an individually connected device. If you poll devices of the same type, you can specify several addresses separated by a comma in this field. For example: 1,2,3,4. It helps minimizing the number of identical requests in the queue.

Process storage numbers - in this field you can specify identifiers of memory storage, separated by commas, that the module will process. Data from other data storage will be ignored. If the field is empty, all available storage numbers are processed. Usually, the number "0" is reserved for instant data, while other storage numbers can store archived data.

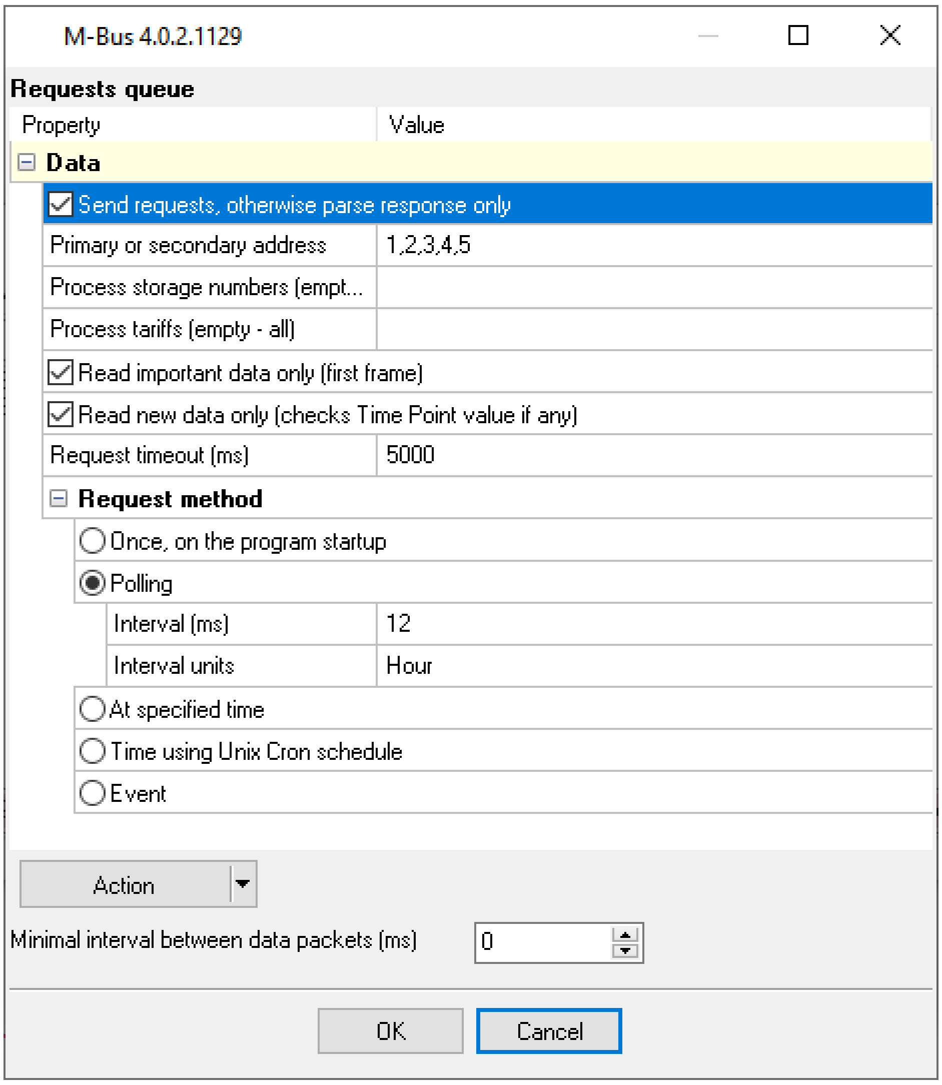
Process tariffs - in this field you can specify tariff numbers, separated by comma, that the module will process. The tariff number can be from 0 to 16. If the field is empty, the plugin processes data for all tariffs. Typically, tariff 0 is used in devices where no tariff system is used (no tariffs).

Read important data only - if this option is enabled, the plugin processes data only from the first data packet that contains instant values. Sometimes, this allows you to speed up the polling of devices if you only need instant values, and do not need historical data.

Read new data only - the plugin can check a time stamp in data packets with help of a value with the "TimePoint" data type. Presence of a timestamp characterizes archive values. Using it, the plugin may process newer data.

Response timeout - it is an interval to wait for a response from an M-Bus device after sending a request. After this timeout, the plugin automatically cancels the current request and execute the next request in the queue. All data arrived after this interval will be ignored. The timeout value depends on a network, devices performance, and the amount of requested data.

Minimal interval between data packets - it is the minimum interval between requests sent over the M-Bus network. You can use this delay with old and slow M-Bus hubs because it cannot process many fast requests, and you have multiple requests in the queue with different schedules. The most modern hubs don't need this delay.



Requests queue

Property	Value
⊟ Data	
✓ Send requests, otherwise parse response only	
Primary or secondary address	1,2,3,4,5
Process storage numbers (empt...	
Process tariffs (empty - all)	
✓ Read important data only (first frame)	
✓ Read new data only (checks Time Point value if any)	
Request timeout (ms)	5000
⊟ Request method	
○ Once, on the program startup	
● Polling	
Interval (ms)	12
Interval units	Hour
○ At specified time	
○ Time using Unix Cron schedule	
○ Event	

Minimal interval between data packets (ms): 0

Fig. 2. Requests queue.

3 Device search

You can start the device search process in the M-Bus network by selecting the corresponding command in the "Action" menu (see fig. 2).

Primary address range

In this mode, the program prompts you for a start and end address in the range from 0 to 252, and searches for devices on the network. When searching for a device, the program sends a special request, which is answered by all M-Bus compatible devices, if a device address in the request is valid.

Please note that if there are devices on the network with the same primary address, then the responses of the devices will overlap and such devices will not be detected.

Secondary address

The secondary address is usually identical to a device serial number and is set at the factory. In combination with a different manufacturer, it may guarantee that the device address is unique in your local M-Bus network.

To search for devices on the network, a special wildcard-based procedure was developed for the rapid and automatic determination of already installed slaves.

The duration of the search depends on the number of devices connected to the M-Bus network.

During the search, the program shows the number of devices found and remaining search time.

Secondary addresses range

This search mode is similar to the previous one, but allows you to set a special mask for a secondary address that the program uses to filter-out unwanted devices. This can significantly reduce the search time if an approximate range of device serial numbers is known.

The mask contains a set of eight digits (0-9) and one special "F" character. The "F" character means that any digit in this position may appear in a device secondary address.

If a device secondary address (serial number) is shorter than eight digits, you must add leading zeros.

Search mask examples:

FFFFFFFF - the program searches for all devices on a network.

0021FFFF - the program searches for all addresses that start with 0021. The mask "FFFF" may include up to 9999 variants of a secondary address.

0021FFF9 - the program searches for all addresses that start with 0021 and ends with 9.

4 Request method

The plugin can send requests in the following mode:

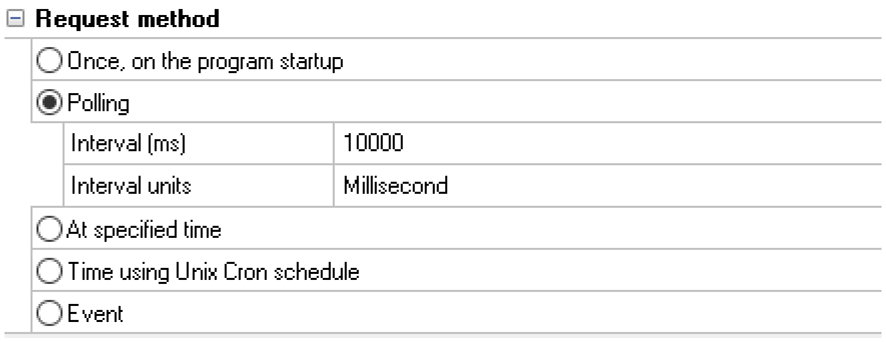
Once, on program startup - the program will send a request once when the program starts.

Polling - the program will send a 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 the time for each request will be larger and vice versa. Because the program executes all requests in the queue one by one, the time between requests depends on the number of requests in the queue.

At the 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 sending requests periodically or at the specified time. You can find 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".

Event - the program executes the corresponding request when the plugin receives an external event. These events can be generated by our other plugins, like "Event generator," "Script execute," "Expressions,"



Request method

○ Once, on the program startup	
● Polling	
Interval (ms)	10000
Interval units	Millisecond
○ At specified time	
○ Time using Unix Cron schedule	
○ Event	

Fig. 3. Request methods

If you have added several requests to the queue, you can move them up or down. To do it, select a request, click the "Action" button, and select an action ("Move up" or "Move down").

You can also click this button to change a request's description or remove a request from the queue.

You can also perform the same actions by using the context menu that pops up when you right-click items in the request tree.

5 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 hour and so on. In the first four fields, you can also use the proprietary "?" (w/o quotes) character. 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 (*) 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 of 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 proprietary "?" 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 simultaneously (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 CRON is not restarted) at the same hour every minute and so on every day, once in five minutes, during the startup hour.