
# The ICP-CON and ADAM parser plugin
## PRINTED MANUAL

# ICP-CON and ADAM parser plugin



Printed: 29.11.2016

**Publisher**

*AGG Software*

**Production**

*© 1999-2016 AGG Software*
*http://www.aggsoft.com*

# Table of Contents

# 1 Introduction

Several developers offer modules for controlling and collecting data. These modules implement the functions of analog-and-digital, digital-and-analog, digital input/output, timers/counters, etc. Modules can be controlled remotely with the help of commands that are called the DCON protocol. Data exchange between the module and the host is maintained in the ASCII format via a bidirectional communication line of the RS-485 standard.

It is the data query and parser module makes it possible to read and process the current values from various DAQ modules. The data parser module can check the checksum of data packets and convert data packets into variables. You can later use these variables in data export modules.

The module supports the following types of DAQ modules:

## ICP-CON I-7000 from the ICP DAS

ICPCON I-7005 (8-channel thermistor input and 6-channel digital output module)
ICPCON I-7011 (Single channel analog input module with DOx2, DIx1)
ICPCON I-7012 (Single channel analog input module with DOx2, DIx1)
ICPCON I-7013(D) (Single channel RTD input module)
ICPCON I-7014 (Single channel analog input module with linear mapping, DOx2, DIx1)
ICPCON I-7015(D) (6-channel RTD input module)
ICPCON I-7016 (D) (ADC/DAC module with DOx4,DIx1)
ICPCON I-7017 (8-channel voltage and current input module)
ICPCON I-7018 (8-channel voltage, current and thermocouple input module)
ICPCON I-7019 (8-channel voltage, current, and thermocouple input module, with various types of inputs)
ICPCON I-7033(D) (3-channel RTD input module)
ICPCON I-7041 (DIO module, DIx14)
ICPCON I-7042 (DIO module, DOx13)
ICPCON I-7043 (DIO module, DOx16)
ICPCON I-7044 (DIO module, DIx4, DOx8)
ICPCON I-7045 (DIO module, DOx16)
ICPCON I-7050 (DIO module, DIx7, DOx8)
ICPCON I-7051 (DIO module, DIx16)
ICPCON I-7052 (DIO module, DIx8)
ICPCON I-7053 (DIO module, DIx16)
ICPCON I-7055 (DIO module, DIx8, DOx8)
ICPCON I-7058 (DIO module, DIx8)
ICPCON I-7059 (DIO module, DIx8)
ICPCON I-7060 (DIO module, DIx4, DOx4)
ICPCON I-7063 (DIO module, DIx8, DOx3)
ICPCON I-7065 (DIO module, DIx4, DOx5)
ICPCON I-7066 (DIO module, DOx7)
ICPCON I-7067 (DIO module, DOx7)
ICPCON I-7080 (2 channel counter with DOx2)
ICPCON I-7083 (3 axis, 32-bit encoder counter)

## ADAM-4000 series from the Advantech

ADAM 4013 (Single channel RTD input module)

ADAM 4015 (6-channel RTD input module)
ADAM 4016 (ADC/DAC module with DOx4,DIx1)
ADAM 4017 (8-channel analog input module)
ADAM 4018 (8-channel analog input module)
ADAM 4050 (DIO module, DIx7, DOx8)
ADAM 4051 (DIO module, DIx16)
ADAM 4052 (DIO module, DIx8)
ADAM 4053 (DIO module, DIx16)
ADAM 4055 (DIO module, DIx8, DOx8)
ADAM 4080 (Counter/Frequency input module)

### NuDAM-6000 series from the ADLINK Technology Inc.

NuDAM 6013 (Single channel RTD input module)
NuDAM 6015 (6-channel RTD input module)
NuDAM 6016 (ADC/DAC module with DOx4,DIx1)
NuDAM 6017 (8-channel analog input module)
NuDAM 6018 (8-channel analog input module)
NuDAM 6050 (DIO module, DIx7, DOx8)
NuDAM 6051 (DIO module, DIx16)
NuDAM 6052 (DIO module, DIx8)
NuDAM 6053 (DIO module, DIx16)
NuDAM 6055 (DIO module, DIx8, DOx8)
NuDAM 6080 (Counter/Frequency input module)

Note: Products and companies mentioned here are used only for definition and identification purposes and can be trademarks and/or registered trademarks of the respective companies.

# 2 System requirements

The following requirements must be met for "ICP-CON and ADAM parser" 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 ICP-CON and ADAM parser

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.

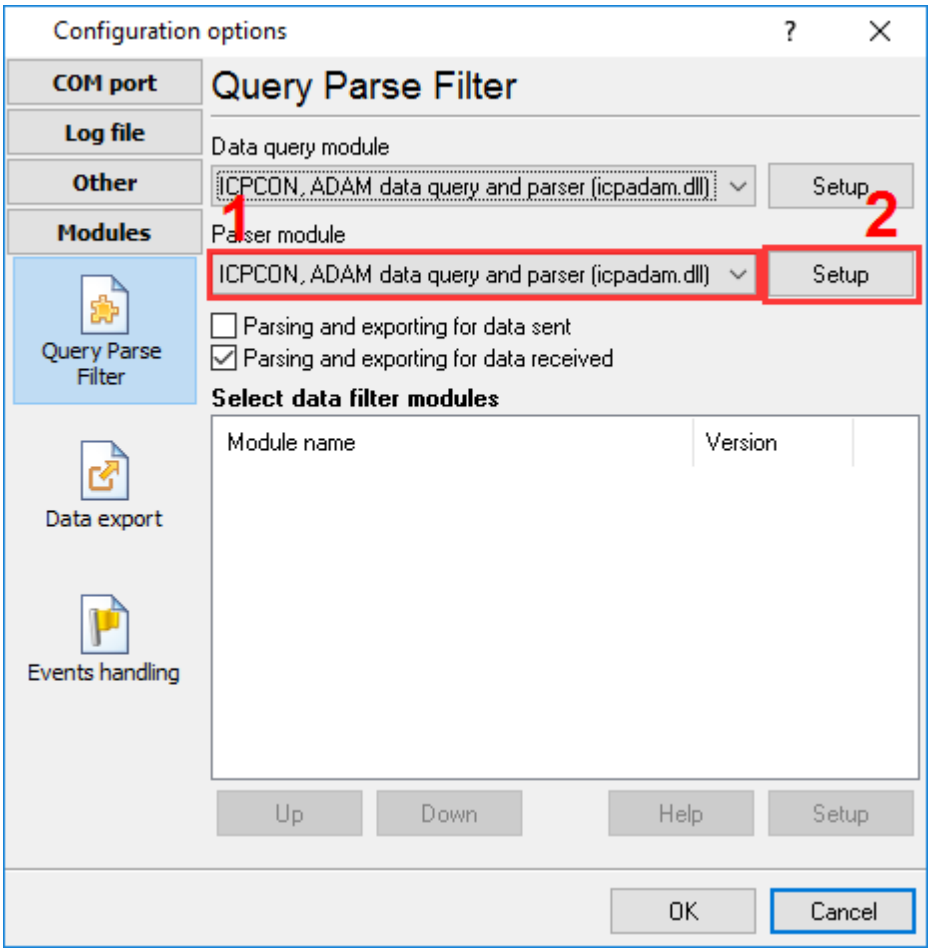


**Fig.1. Example of installed module**

# 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 Configuration

This module can simultaneously run in two modes:

- **Data Query mode** – in this mode, the module sends queries to read data to the external device.
- **Parser mode** – the module parses data received from external devices in this mode and tries to single out information data packets from the flow, after that the parser extracts data from these data packets and saves them as variables. Then you can use these variables for data export.

To activate both modes at the same time, you should select the "Universal data query and parser" module in the "Data query module" and "Parser module" lists (fig.1).

If you select the module in one of the lists only, only the corresponding mode will be activated. For example, you may need it when you monitor data exchange between another program and an external device and want to export the obtained data somewhere.

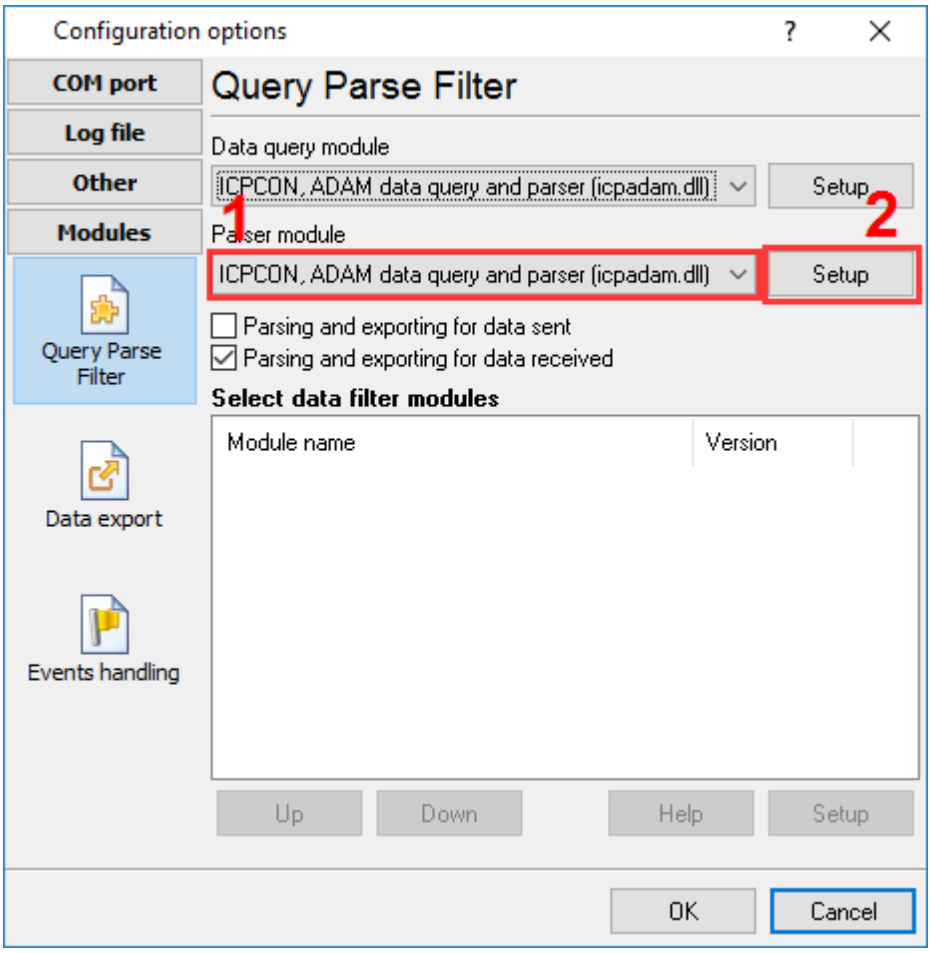


**Fig.1 Selecting the mode**

After you select the module in the list, you have to configure it. To do it, click the "Setup" button. You should use the new dialog box to create a request queue that the module will use.

## Request queue

You can create a queue of requests consisting of one or several items. To add a new request, you should click the "Actions -> Add new request" button. After that you will see the dialog box (fig.2) where you should enter the description of the request. The description can consist of any set of characters. After you enter the description, click the "OK" button and the new request will be added to the queue (fig.3). You can also use the "Action" button or the context menu to change the description, move or remove the request. To do it, Select your request in the queue and right-click it.

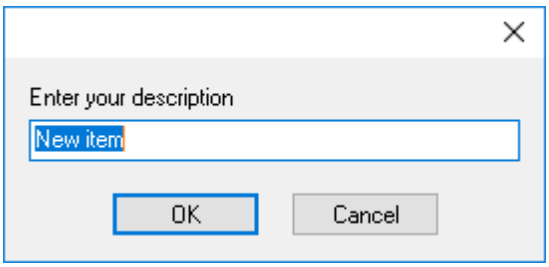


**Fig.2. Request description dialog box.**

When you add a new request, it is added with the default parameters:

- **Device** – you should select the type of a supported device in this field;
- **Request** – after you select the device type, you should use this field to select the command that will be sent;
- **Processing type** – you should select the processing type in this list;
    1. **Send and receive** – this request will be sent to the external device and after that data from it will be processed. For this mode to run normally, you should select the module in both lists on fig.1;
    2. **Send** – the module only sends the request to the external device. The module does not wait for any data to be received in response. For this mode to run normally, you should select the module in the "Data query" list on fig.1;
    3. **Receive** – the module only processes incoming data. No command is sent to the external device. For this mode to run normally, you should select the module in the "Data parser" list on fig.1;
    4. **Disabled** – this request is temporarily disabled. Data is neither sent nor processed.
- **Request properties** – this group of parameters appears when the first or second processing type is selected and it allows you to specify the parameters of querying the external device.

## Request properties

You can specify the following parameters:

**Request timeout** – the module will wait for a response from the external device for the specified time interval in milliseconds. If the module does not receive a response within the specified interval, the next request becomes active in the queue and data that will be received after this interval is over will be ignored. If data is received before the interval expires, the module will switch to the next request right after it receives data.

The following group of parameters allows you to select when to send the request:

- **Once, on program startup** – the module will send the request only once, either when it is started or when its configuration is changed;
- **Polling** – the module will send the request all the time. The polling interval is specified in the "Interval" and "Units" parameters. These parameters specify the minimum interval between requests. The actual interval may depend on the timeout and lags in the data transfer interface. Since the module sends all requests in turns, the actual polling interval also depends on the number of requests in the queue and their parameters.
- **At specified time** – you can specify the exact time when the request should be sent. You can specify several values separating them with a semicolon. Time is specified in the following format: HH:MM:SS (hours, minutes, seconds). The actual time of sending the request depends on the same conditions as the previous parameter.

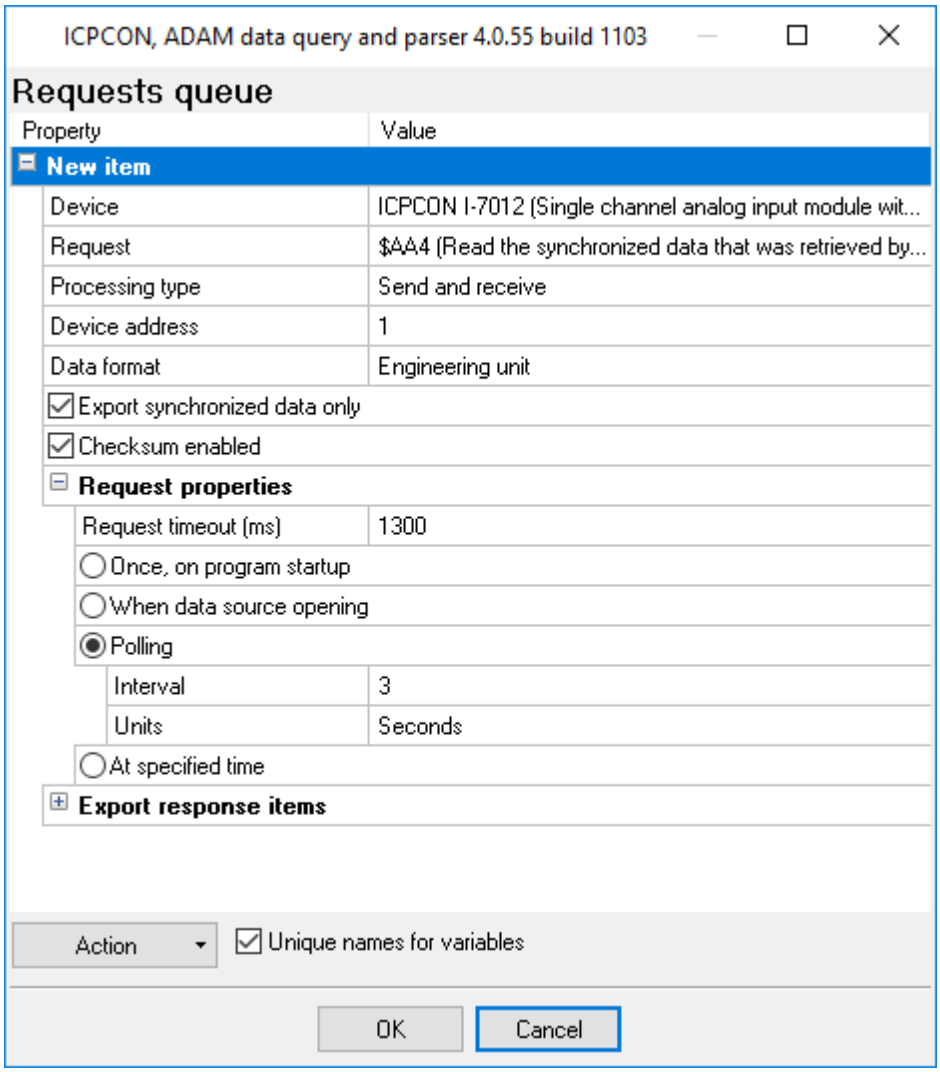


**Fig.3. Request queue**

## Advanced parameters

Each device type or command may have advanced parameters. These include:

- **Checksum enabled** – the checksum is enabled or disabled. If the checksum is enabled, it is added to the request before it is sent and checked when the response is received;
- **Data format** – data format in the response. It depends on your device settings;
- **Export synchronized data only** – the module exports only synchronized data (only in some commands).

Refer to the documentation for the particular device for a more detailed description of these and other advanced parameters.

After you configure the request queue, click the "OK" button, the requests being sent will be displayed in the main window of the program. By default, they are highlighted in yellow. You can set the main program to stamp the date and time when it shows data on the screen.

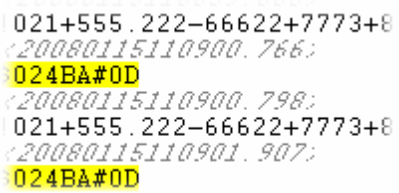


**Fig.4. Sample data being sent and received**

## Parser

For the module to be able to process responses, you should select the first or third processing type. Each command has its own set of exported variables that are programmed in the module. You can find more details about data that some command returns in the documentation for the corresponding device.

After the module receives a data flow, it does the following:
1. Singles out data packets from the data flow, i.e. the data that the device returns;
2. Checks the device response for signs of the data packet beginning and/or ending and also its length
3. If checksum is enabled, it checks it
4. Checks the response timeout. If the timeout has expired, it ignores the data packet
5. If the data packet passes the check, the module parses it into variables.

The list of available variables is specified in the **Export response items** group (fig.3). If the device returns no data, this list will be empty. You can select what variables you want to export. To do it, select the corresponding variables. By default, all data is exported.

The "**Unique names for variables**" checkbox allows you to export data in two modes:

1. Option enabled. In this mode, each exported variable will have a unique name that will contain the device type, its address, the variables identifier and its ordinal number. Then you can use the Aggregator module to combine various variables into one data string. Thus, you can collect and export data from several devices of different types into one data table. Sample variable names: I7015_1_VAL1, I7012_2_VAL5
2. Option disabled. In this mode, the name of an exported variable will consist of the variable identifier and its number. It allows you to export data into one table for several devices of one type without using the Aggregator module and without more complicated configuration. Later, you can identify data by the port number and device address in the table (if you export it). Sample variable names: VAL1, VAL2, SYNC.

## Additional variables

If a data packet passes the check and there is some data to be exported, the module automatically adds the following additional variables:

**FULL_DATA_PACKET** – this variable will contain the entire data packet, including the signs of packet beginning and end.

**DATA_PACKET** – this variable will contain the data packet, excluding the signs of packet beginning and end.

**Add date/time stamp to each response parsed** – if this option is enabled, the parser will add the new "date stamp" variable named **DATE_TIME_STAMP** to each data packet that will be parsed into variables. The module uses the computer clock to get the date and time. This variable will contain the date and time when the data packet was received;

**Add port number to each response parsed** – if this option is enabled, the parser will add the new "port number" variable named **SERIAL_PORT_NUMBER** to each data packet that will be parsed into variables. It will allow you to identify data during export if you simultaneously collect data from several devices.

# 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.