



**The GE Fanuc Automation PLC plugin
PRINTED MANUAL**

GE Fanuc Automation PLC plugin

© 1999-2024 AGG Software

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Printed: 11/2/2024

Publisher

AGG Software

Production

© 1999-2024 AGG Software

<http://www.aggsoft.com>

Table of Contents

| | |
|--|----------|
| Part 1 Introduction | 1 |
| Part 2 System requirements | 2 |
| Part 3 Installing GE Fanuc Automation PLC | 2 |
| Part 4 Glossary | 3 |
| Part 5 User Manual | 4 |
| 1 Data query | 4 |
| 2 Request method | 6 |
| 3 Data parser | 6 |
| Part 6 Troubles? | 7 |
| 1 Possible problems | 7 |

1 Introduction

Our plugin allows to read data from the wide range of GE Fanuc Automation PLCs. It allows to read data from all memory types you can access in your PLC programs. The plugin supports CMM Master-Slave, CMM Peer-to-Peer, SNP and SNP-X protocols.

The SNP protocol is a proprietary communications protocol developed by GE Fanuc Automation. It is the native communications protocol for all models of the Series 90 PLC product line.

The SNP-X protocol is a highly optimized extension of SNP. While it offers fewer functions than SNP, SNP-X is simpler to use and provides a significant performance improvement over SNP. It does not support PLC programming or configuration operations.

CCM protocol is included in the EPROM firmware for both the Series 90-70 and Series 90-30 CMM modules. The CCM protocol was originally developed for the Series Six Communications Control Module (CCM) and is available on most GE Fanuc PLCs.

Supported devices:

- Series GE Micro
- Series 90-30 311
- Series 90-30 313
- Series 90-30 331
- Series 90-30 341
- Series 90-30 350
- Series 90-30 360
- Series 90-70 731
- Series 90-70 732
- Series 90-70 771
- Series 90-70 772
- Series 90-70 781
- Series 90-70 782
- GE OPEN - Wide range model support
- Series Five
- Series Six CCM2 - including Expanded I/O

This module has the following features:

- Can send valid data request to any compatible device;
- CRC for each data packet will be calculated and verified automatically;
- Can poll PLC data by a custom interval;
- Can flexibly parse all received data packets and extract register's values.
- Memory Types Supported: I, Q, G, M, S, SA, SB, SC, R, AI, AQ (depends on PLC)

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 "GE Fanuc Automation PLC" to be installed:

Operating system: Windows 2000 SP4 and above, including both x86 and x64 workstations and servers. The 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.

3 Installing GE Fanuc Automation PLC

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.

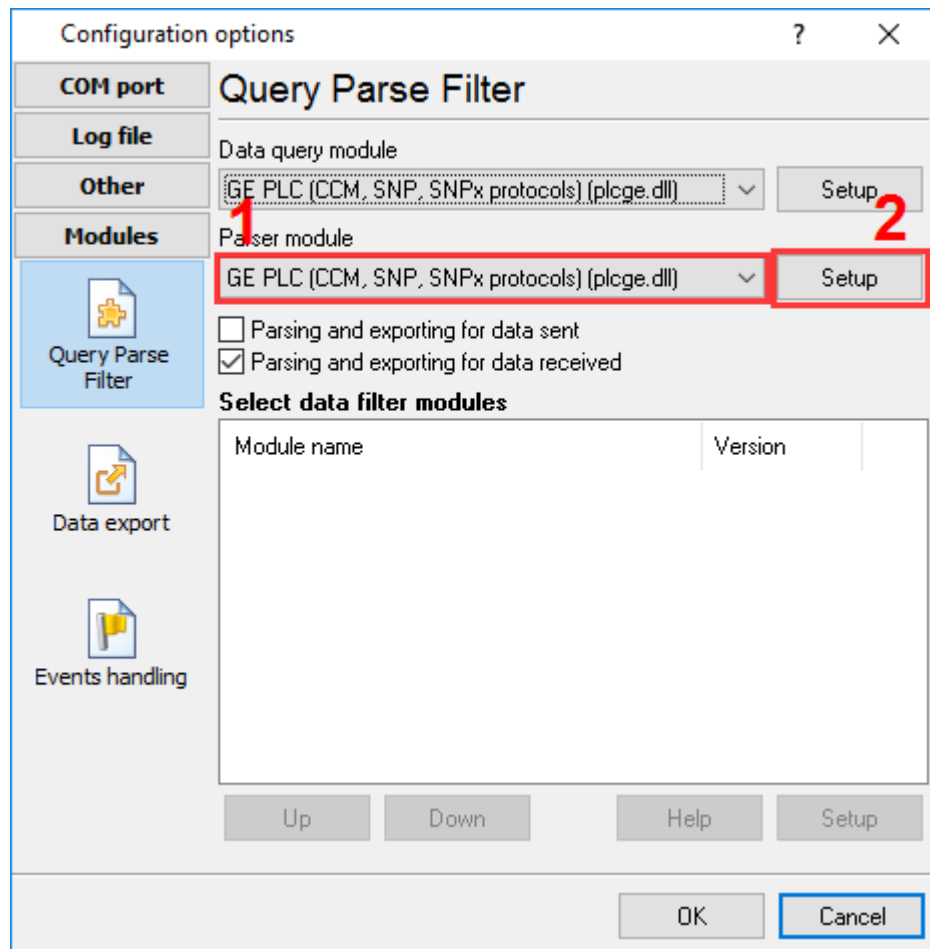


Fig. 1. Example of installed module

4 Glossary

Main program - it is the main executable of the application, for example, Advanced Serial Data Logger and asdlog.exe. It allows you to create several configurations with different settings and use different plugins.

Plugin - it is the additional plugin module for the main program. The plugin module extends the functionality of the main program.

Parser - it is the plugin module that processes the data flow, singling out data packets from it, and then 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.

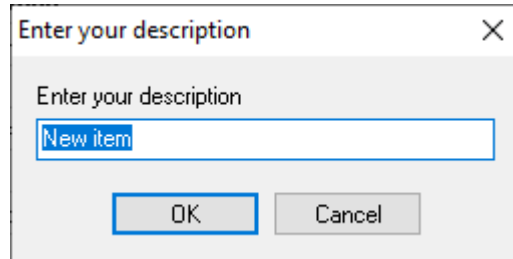


Fig.1. Name dialog

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

- **Protocol** - the protocol type that the plugin will use to interact with your PLC;
- **Source address** - the address of the master computer in a network. It is used in some protocols;
- **Target address** - the address of a slave PLC in a network. For the CCM protocol it should be a number from 0 to 72. For SNP and SNP-X it should be a SNP ID: up to seven characters (0-9, A-F);
- **Memory** - a memory type in the PLC;
- **Data address** - the address (offset) of the first byte in the memory. This value is zero-based. For example, the %R1 register has the offset equal to 0;
- **Read values** - the number of values to read (number of registers, number of discrete inputs/outputs). A number of bytes depends on the memory type;
- **Request timeout** - this is the time interval for which the program is sending request to a 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.

Note: the plugin does not check memory ranges and number of bytes to read. You need to configure requests according to this note.

GE PLC 4.0.20 build 1115

Protocol: SNP-X

Source address: 1

Requests queue

| Property | Value |
|--|----------------|
| Request #1 | |
| <input checked="" type="checkbox"/> Send requests, otherwise parse response only | |
| Target address | 2 |
| Memory | Registers(%R) |
| Data address | 0 |
| Read values | 4 |
| Request timeout (ms) | 10000 |
| Request method | |
| <input type="radio"/> Once, on program startup | |
| <input checked="" type="radio"/> Polling | |
| Interval (ms) | 1000 |
| Interval units | Millisecond |
| Response items | |
| Item #1 | |
| Name | ITEM1 |
| Offset | -1 |
| Count | 1 |
| <input checked="" type="checkbox"/> Append counter to name | |
| <input type="checkbox"/> Zero based counter | |
| Data type | Decimal, 8 bit |
| <input type="checkbox"/> Little endian, otherwise Big endian (numbers only) | |
| <input checked="" type="checkbox"/> Unsigned, otherwise Signed (decimal numbers only) | |
| <input checked="" type="checkbox"/> Swapped (most significant register first) (32 and 64 bit numbers only) | |
| Default value | 0 |

Action: ▾

Minimal interval between data packets (ms): 0

OK Cancel

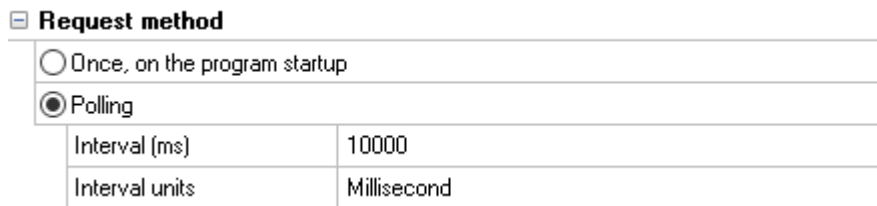
Fig.2. Request

5.2 Request method

The plugin can send requests in two modes:

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.



| Request method | |
|--|-------------|
| <input type="radio"/> Once, on the program startup | |
| <input checked="" type="radio"/> Polling | |
| Interval (ms) | 10000 |
| Interval units | Millisecond |

Fig. 2. 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.

5.3 Data parser

All data export modules use parser variables containing parsed (decoded) values. Our GE Fanuc Automation PLC picks out significant data blocks (data packets) from the common data flow, analyzes the extracted data packet, and checks its integrity using CRC (cyclical redundancy check).

All parser items are assigned with a corresponding request in the queue. You can define one or more parser items (variables) in one request.

You can add a new parser item (variable) to the request by clicking "Actions - Add response item." Before, you should select a caption of the corresponding request. A new parser item (variable) will appear in the "Response items" group (fig. 3).

| | |
|--|--------------------------------|
| ☐ Response items | |
| ☐ Voltage Va-n (V) | |
| Name | VOLTAGE_VA_N |
| Offset | 0 |
| Count | 1 |
| <input type="checkbox"/> Append counter to name | |
| Data type | Single precision float, 32 bit |
| <input type="checkbox"/> Little endian, otherwise Big endian (numbers only) | |
| <input type="checkbox"/> Unsigned, otherwise Signed (decimal numbers only) | |
| <input checked="" type="checkbox"/> Swapped (most significant register first) (32 and 64 bit numbers only) | |

Fig. 3. Data parser items.

Each response item has a few important options:

- **Name** - the of the parser variable. This name you'll bind with fields in data publication modules.
- **Offset** - the device can respond 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 the first byte of your value is located at the beginning of the data block, then this value should be 0. You can specify -1 here. Then the program will automatically calculate the value offset. Please, note, the data block does not include a data packet header (address and data size).
- **Count** - is the number of values (nor bytes) with the same parameters (data type and default value), 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. Please note that it is a count of values, not bytes or registers (one value can allocate one or more bytes or registers).
- **Data type** - is the data type of the value. Each value can utilize one (for the "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 export. Solution: configure the parser and make sure that one or more variables are defined 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 match source data.

Unable to disconnect from the database [%s] and **Unable to connect to a database [%s]** - it is impossible to connect or disconnect 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 the "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 execution of 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.