User Manual
CANinterpreter

Version history

Version	Changes	Date	Editor
V1.0	First version	2013/07/03	ged
V1.0.3	Minor bugfixes	2013/09/20	ged
V1.1	Additional PlugIns added	2014/02/28	ged
V1.2	CAN message format changed	2016/04/01	ri

Disclaimer

Even if software from emotas embedded communication GmbH is produced on high reliability levels and tested on different environments there is no proof on absolute reliability. Therefor emotas embedded communication GmbH cannot give any liability on even software or documentation and cannot guarantee there functionality in use cases of our customers. Especially description and technical data are no granted properties of our products. As a consequence emotas embedded communication GmbH will not be responsible for any damages and losses in consequence of using our products.

emotas embedded communication GmbH may apply changes on products and documentation to improve reliability, stability and serve technical advantages. This changes may not be announced.

emotas embedded communication GmbH holds all rights on their products and documentation. Propagation to third parties even from parts of products or documentation may be permitted by emotas embedded communication GmbH. Copies of products or documentation may by established only on backup purposes. It is in the responsibility of our customers not to pass these copies to third parties. We appreciate very much all considerations of our customers on errors ore other aspects of improvement.

Copyright

Fritz-Haber-Str. 9
D-06217 Merseburg
Germany
Tel. +49 3461/79416-0
Fax. +49 3461/79416-10
service@emotas.de
http://www.emotas.de

Table of Contents

1 Introduction

Thank your for using the CANinterpreter. The CANinterpreter is a versatile tool to test, configure and monitor CANopen devices. The following manual explains the installation and use of the program.

2 Installation

2.1 Windows

To install the tool on Windows start the setup setup_caninterpreter.exe and follow the instructions of the setup. The setup creates a shortcut to start the program. At the first connection to CAN the program requires a license file, which is copied to the program and enables the licensed features.

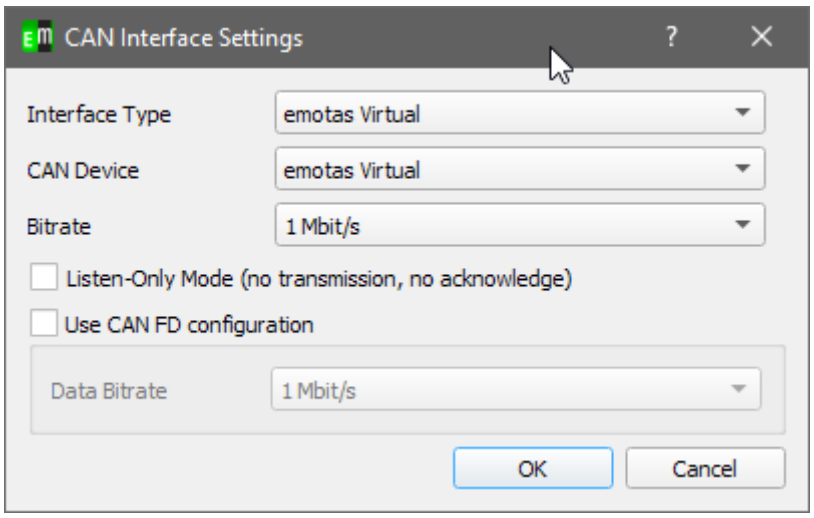
It is possible to use the tool without license file for one hour with a fixed bit rate of 125 kbit/s for evaluation purposes.

2.2 Linux

To install the tool in Linux just unzip the ZIP file setup_caninterpreter.zip into a directory. To start the program run the script CANinterpreter.sh in this directory. At the first connection to CAN the program requires a license file, which is copied to the program and enables the licensed features

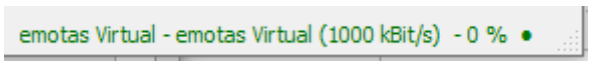
3 First steps

The first step at the very first start of the tool is the configuration of the CAN interface. Open CAN interface settings at the menu entry "Connection → CAN Interface Settings". The following mask appears.



Choose the type of the CAN interface, the name of the CAN device and the bit rate in the CANopen network and confirm the settings with OK.

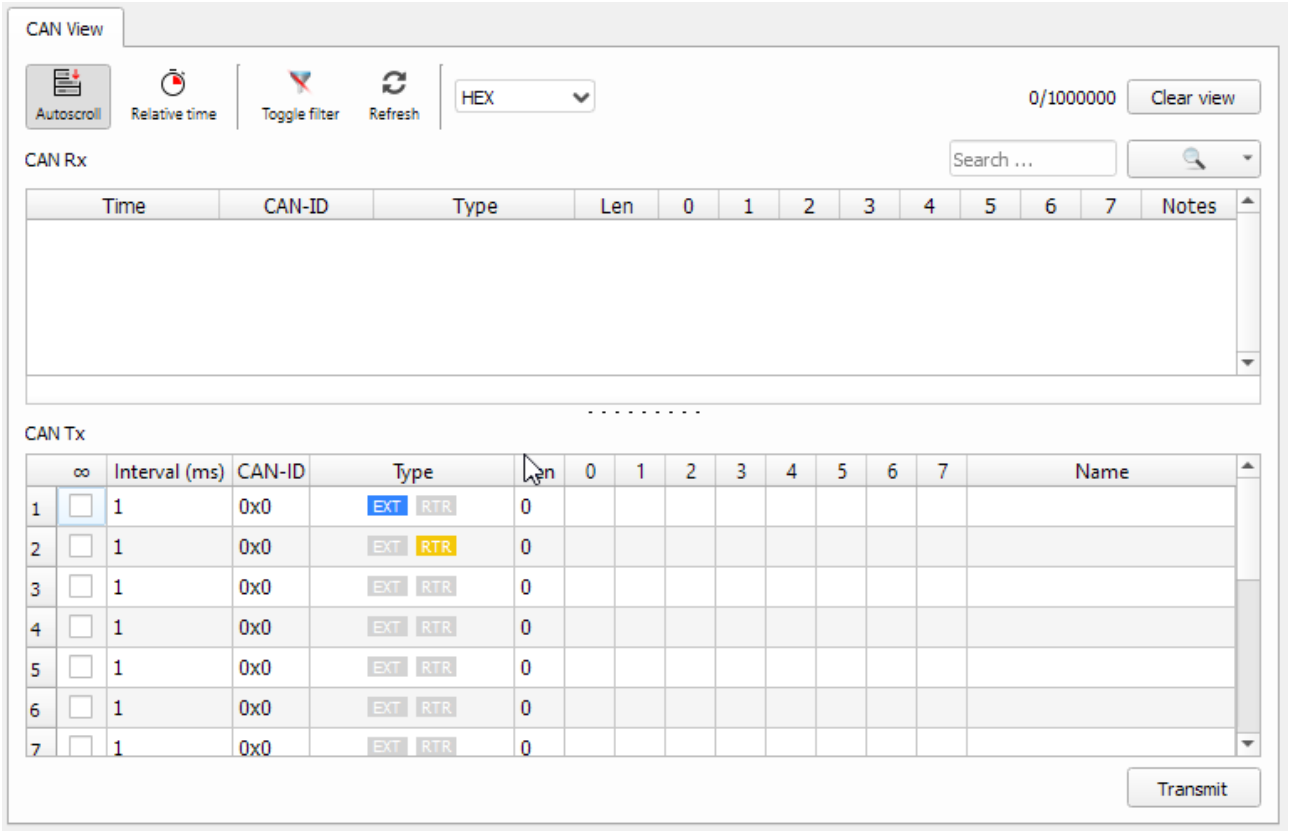
Connect now the CANinterpreter with the CAN interface via "Connection → Connect". In the status bar you can see now "Connected to " and the name of the CAN device and the current bitrate.



4 Program components

4.1 CAN View

The CAN View shows received and transmitted CAN messages. To send a CAN message the Transmit table below can be used. The values for CAN-IDs, DLC and data can be specified as decimal values or as hexadecimal values with leading 0x.



The type of the CAN message can be set via the checkboxes in the “ext” and “rtr” columns. The following overview shows how each CAN message type could be set:

CAN message type	**ext**	**rtr**
CAN base data message		
CAN extended data message	x	
CAN base RTR message		x
CAN extended RTR message	x	x

Up to 16 transmit messages can be defined. The selected one is sent by clicking on the Transmit button. Cyclic messages can be sent automatically by the tool, if the value in the column 'Interval(ms)' is larger than 0.

In the menu of the CAN View window a filter for distinct CAN IDs can be defined. A list of CAN IDs can be defined, which either can be displayed or ignored. The list can include single values like (100,200,0x400,0x500) or a range (0x100-0x200) or a combination of both e.g. 1,2,0x300-0x400,720.

The filter type defines the behavior of the filter:

- PASS – only the CAN IDs in the filter list are displayed
- REJCECT – the CAN IDs in the filter list are ignored, all other CAN IDs are displayed

The time stamp of the CAN message can be absolute or relative values and the accuracy depends on the used CAN interfaces and the operating system. For most CAN interfaces no TX time stamp is available.

The recorded CAN messages can be exported into text files by CAN View → Export CAN-Logging.

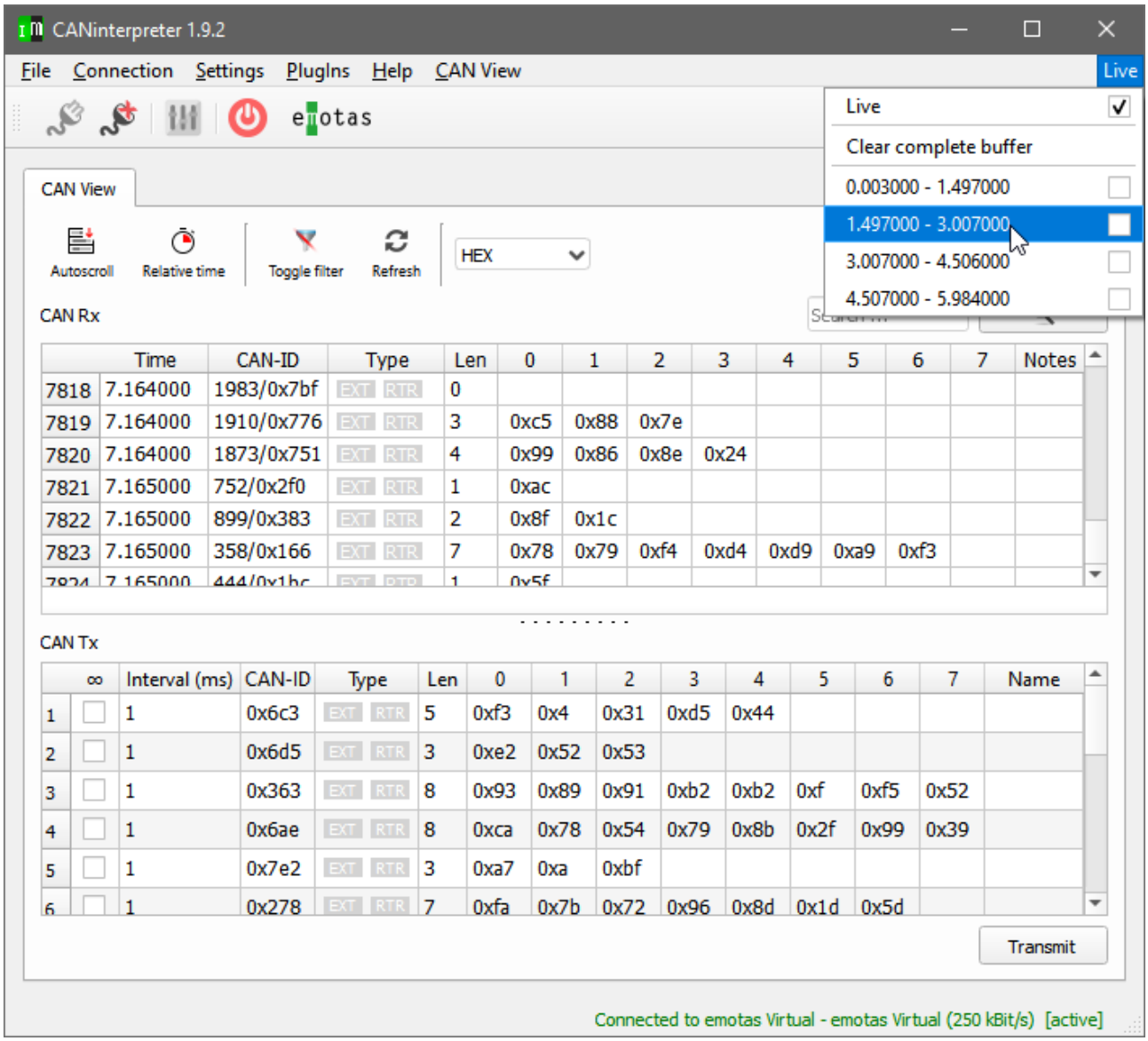
The format of the save text is explained below:

```
3.653302 0x5a0/1440 (8): 43 18 10 03 00 00 00 00
time stamp
           CAN-ID
                    DLC
                         Data in hexadecimal notation
```

4.2 CAN message handling

The received messages are stored as message package to the hard disc drive if the "Maximal message count" (configurable in the options) is reached and removed from the internal buffer.

The saved messages could be viewed via the menu in the upper right corner of the window.



Each message package holds the timestamp of the first and of the last message as name.

If a message package is selected newly received message are still being processed in the background and could be viewed by switching back to "Live".

5 Menu

The menu provides access to various functions and settings of the CANinterpreter.

File

- **Export CAN Logging**

  Export of CAN messages as text file. Import and interpretation in the PlugIns at a later point of time is possible.

- **Quit**

  Quit the application.

Connection

- **CAN Interface Settings**

  Dialog to configure CAN interface and bit rate

- **Connect**

  Connect to CAN using the configured interface

- **Disconnect**

  Disconnect from CAN

Settings

- **Filter Settings**

  Filter configuration for CAN messages. Only valid for the main part of the program. There are separate filters for all PlugIns.

- **Options**

  Opens the options dialog to configure various settings of the program.

- **Save**

  Saves the current settings. Enabled "Settings → Options → Save settings automatically at exit" saves the settings automatically when the program is quit.

- **Export Settings**

  Export of the current settings into a configuration file. Can be used to store various settings of different use cases.

- **Import Settings**

  Import of settings from a configuration file.

- **Update Licence File**

  Dialog to select a new license file. The content of the license file can be viewed before the

import.

- **Check for Updates**

  Query the web server for updates of the tool. Beside the IP address no additional data is transmitted.

PlugIns

Menu to activate various extensions of the CANinterpreter. The availability of PlugIns depends on the license.

Help

- **Manual**

  Shows the complete manual as PDF file.

- **About**

  Shows 'about' dialog including license information.

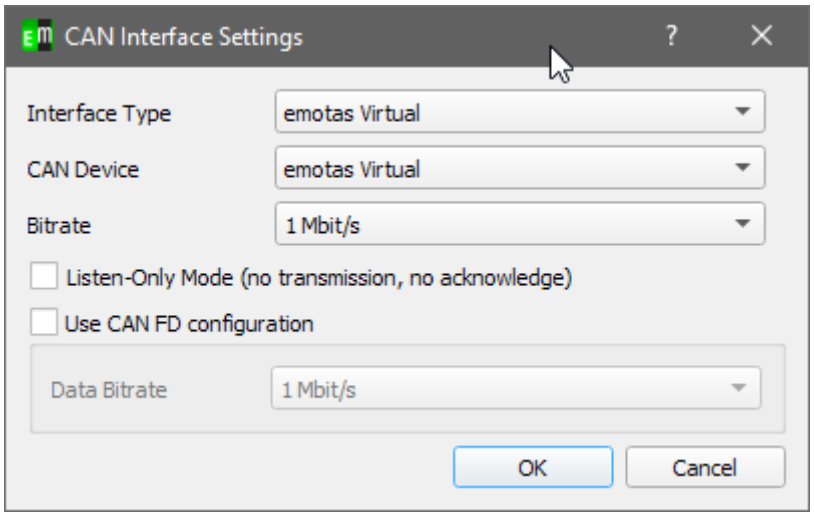
- **About Qt**

  Information about the Qt framework and license information for the used Qt components.

6 Settings

6.1 CAN Settings

The CAN interface can be configured using the CAN settings dialog.



- **Interface Type**

  Selection of the CAN interface. Currently different CAN interface manufacturers are supported on Windows and on Linux.

- **CAN Device**

Depending on the Interface type the available CAN devices are listed here to choose from. E.g. can0, can1, ...van0 for SocketCAN or usb1...usb8 for PCANBasic.

- **Bit Rate**

  Configuration of the bit rate of the CANinterpreter. The configured bit rate must match with the bit rate of the CAN network. For SocketCAN it is necessary to set the bit rate before starting the program

- **Advanced Settings**

  Some CAN Interfaces support some additional settings that can be configured there. These might be:

  - Listen-Only-Mode
  - User specific bit rates
  - Activation of terminator resistors

6.2 Program Settings

The option dialog provides access to various settings of the CANinterpreter.

General Settings

- Warning when exit with active CAN connection
- Save Settings automatically at exit
- Auto Connect to CAN after startup

  Automatic setup of a connection to CAN interface with program start

Scripting

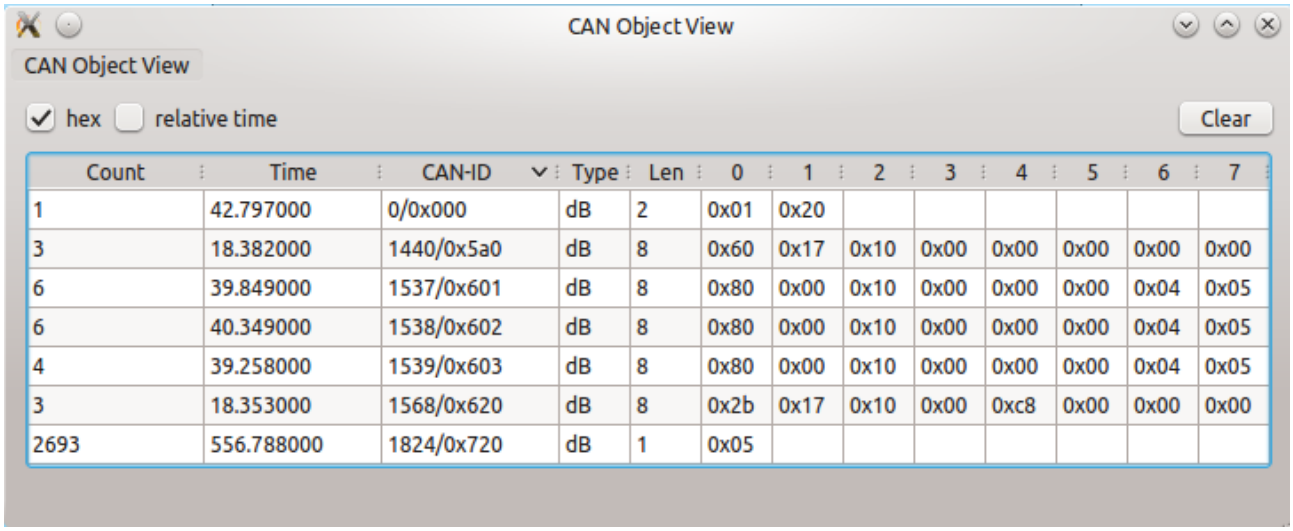
- Autostart Script at connect

  Autostart Script starts automatically with active Sripting Interpreter and connection to CAN. In combination with the "Auto Connect to Can after startup" the script startst automatically after starting the program

7 PlugIns

7.1 CAN Object View

The CAN Object View shows all received CAN messages in the so called 'Object View'. That means that all received CAN IDs are shown in a table with the last data and the number of receptions.



Count	Time	CAN-ID	Type	Len	0	1	2	3	4	5	6	7
1	42.797000	0/0x000	dB	2	0x01	0x20						
3	18.382000	1440/0x5a0	dB	8	0x60	0x17	0x10	0x00	0x00	0x00	0x00	0x00
6	39.849000	1537/0x601	dB	8	0x80	0x00	0x10	0x00	0x00	0x00	0x04	0x05
6	40.349000	1538/0x602	dB	8	0x80	0x00	0x10	0x00	0x00	0x00	0x04	0x05
4	39.258000	1539/0x603	dB	8	0x80	0x00	0x10	0x00	0x00	0x00	0x04	0x05
3	18.353000	1568/0x620	dB	8	0x2b	0x17	0x10	0x00	0xc8	0x00	0x00	0x00
2693	556.788000	1824/0x720	dB	1	0x05							

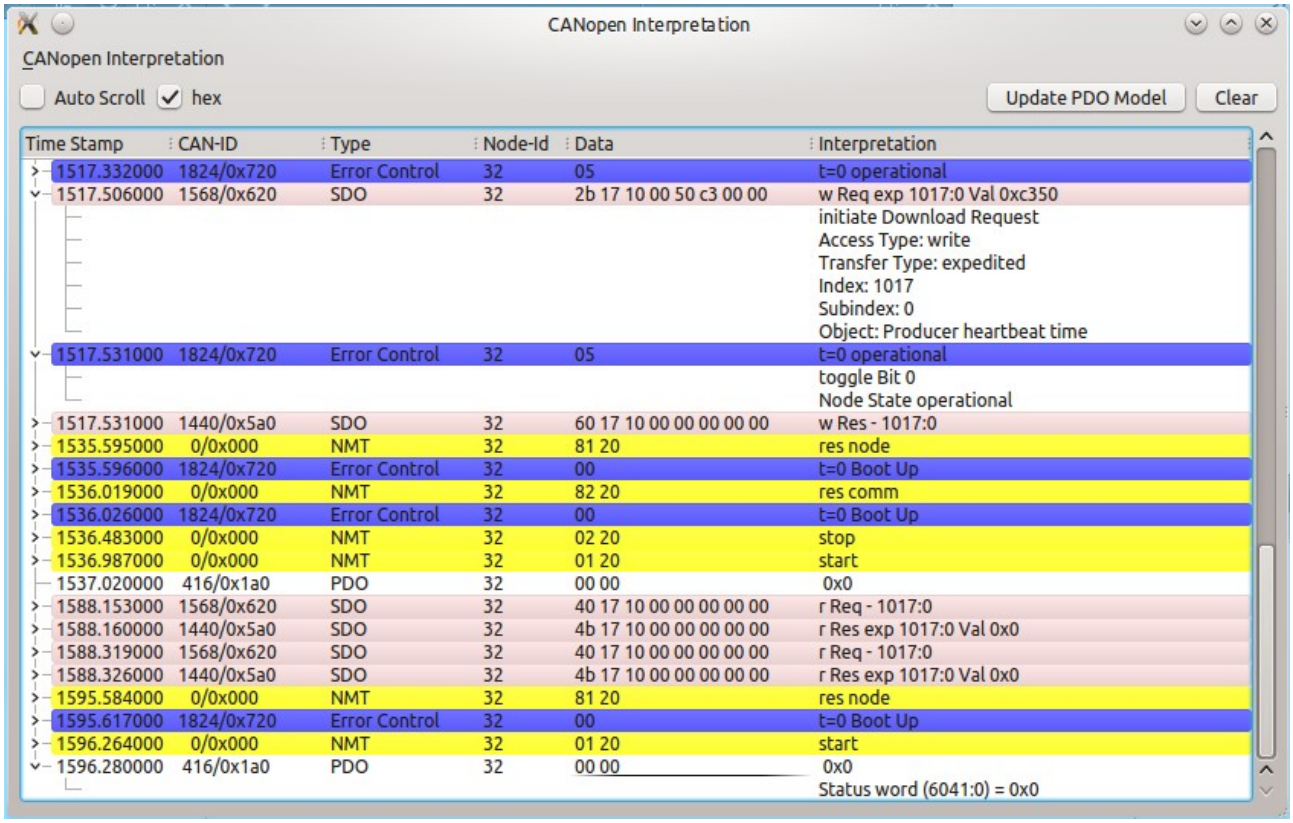
The table can be sorted by the number, the time stamp or the CAN ID. A filter can be configured in the same way as in the CAN View PlugIn. The CAN Object View is included in the standard scope of delivery of the CANinterpreter.

7.2 User-defined Interpretation

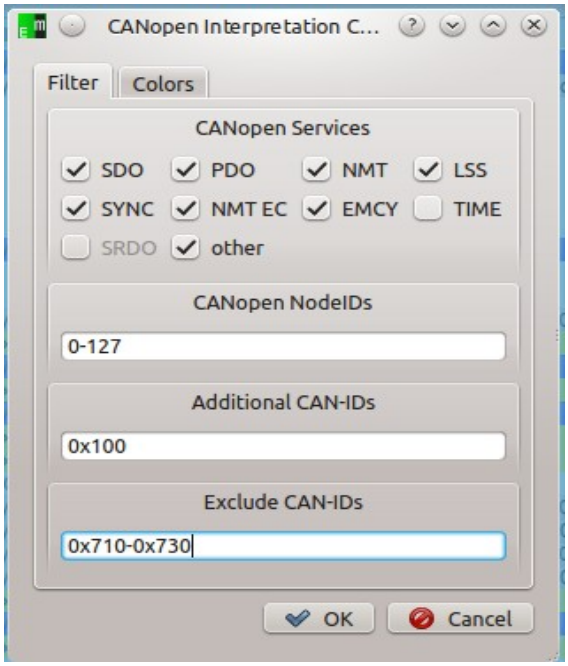
This PlugIn allows user specific interpretation of CAN data.

7.3 CANopen Interpretation (optional)

The CANopen Interpreter interprets all received CAN messages according to the CANopen protocol and displays the CANopen service of the message, the source or target node ID and a service-specific interpretation in a human-readable format. Additionally the CANopen Interpreter can interpret the content of PDO messages according to the PDO mapping as defined in EDS or DCF files.



A filtering of CAN messages is possible by different criteria:



- CANopen Service – only the enabled services are displayed
- CANopen NodeID – only the selected node IDs are displayed. A definition of ranges like e.g (1,2-30,40) is possible.
- Additional CAN-IDs: CAN IDs, which are rejected by the previous filters can be enabled again.
- Exclude CAN-IDs: CAN IDs, which have passed the previous filters can be filtered out selectively. A definition of ranges like e.g. 100,0x710-0x730 is possible.

Interpreted CANopen messages can be exported and imported as text files. Raw CAN loggings can be imported and interpreted as well.

7.4 CANopen Object View (optional)

The CANopen ObjectView combines the CANopen interpretation with an object view of the CAN messages. All received CAN-IDs are shown with the last interpreted values.

7.5 EnergyBus Interpretation (optional)

The optional EnergyBus Interpretation displays the state of EnergyBus devices in an EnergyBus (CiA-454) network.

7.6 EnergyBus Object View (optional)

The EnergyBus ObjectView combines the EnergyBus interpretation with an object view of the CAN messages. All received CAN-IDs are shown with the last interpreted values.

7.7 File Logger

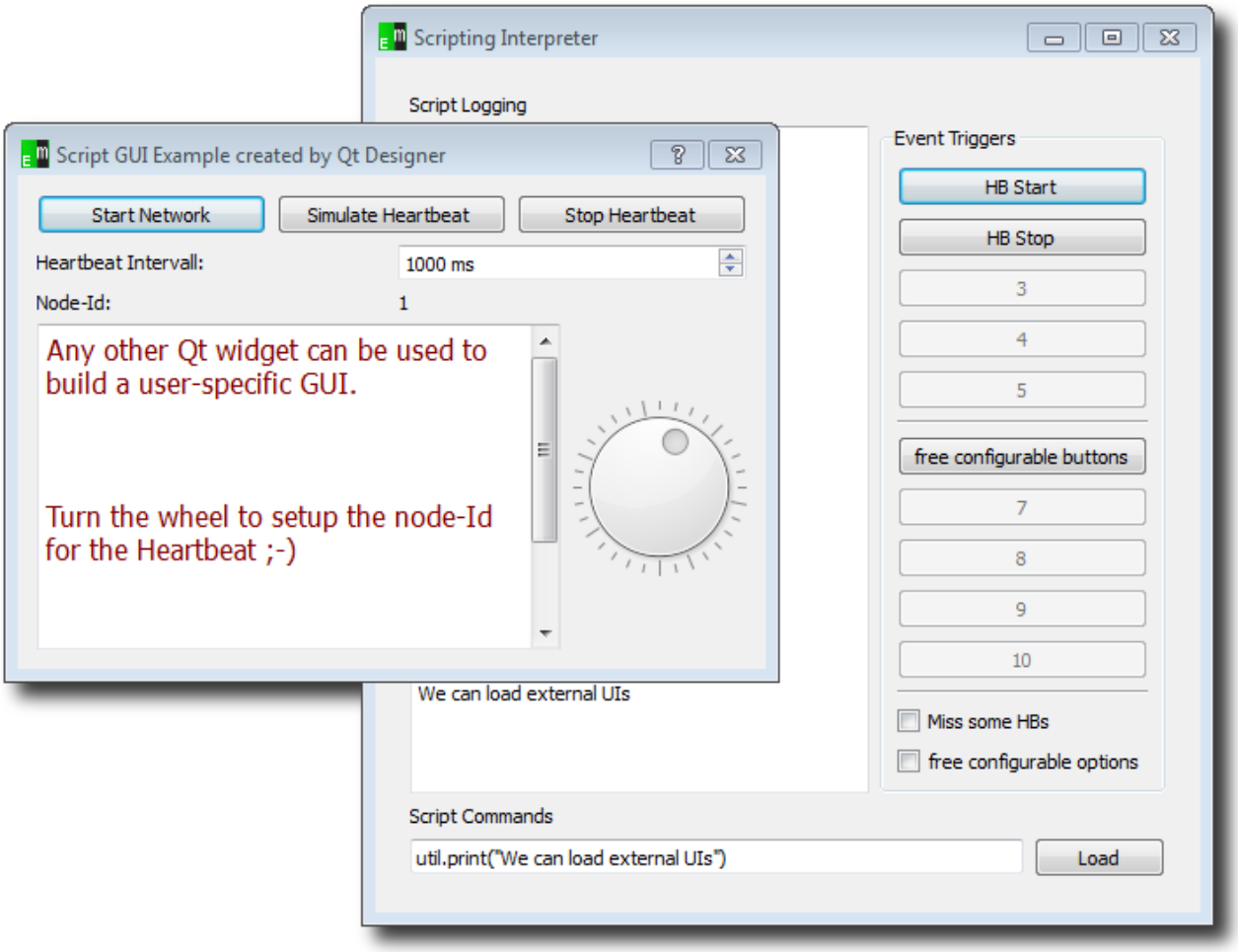
The File Logger is able to record CAN messages directly into log files according to certain trigger conditions.

7.8 Scripting Interpreter (optional)

The Scripting Interpreter provides the possibility to run QtScript(JavaScript) programs with special extensions for CAN.

Even tailored user interfaces can be generated by the QtDesigner and used in scripts.

All additional CAN/CANopen-specific commands are explained in the separate document "CANopen Scripting Interpreter – API Reference" (cde_script_api.pdf).



The following example shows the usage of a QtScript program:

```
// print something to console and set device to operational
util.print("Test of simple device");
nmt.preopNetwork();
nmt.startNode(32);
i = 0;
util.print("We are in " + util.pwd());

// set node id for SDO access
sdo.setNodeId(32);

// loop over objects 0x4000 to 0x04010
for (object = 0x4000; object < 0x4005; object ++) {
    str = "Test object ";
    str = str + object;
    util.print(str);

    // write value to object 0x4000..
    result = sdo.write(object, 0x0, UNSIGNED32, i);
    if (result == "SDO_OK") {
        util.print(" Write OK");
    } else {
        util.print(" Write NOT OK");
    }
    // wait a bit to allow device update its internal values
```

```
    util.msleep(10);

    // read from 0x4100.. and expect same value
    result = sdo.read(object+0x100, 0x0, 0x07);
    if (result == i) {
        util.print("Read OK");
    } else {
        util.print(" Read NOT OK");
    }
    i++;
}

// user defined function that can be called from Scripting tab
function urk(count) {
    for (i = 0; i < count; i++) {
        nmt.startNetwork();
        nmt.stopNetwork();
        var dlc = 4;
        var canId = 0x100 + i;
        can.sendBaseFrame(canId, dlc, 1 , 2 , 3 ,4 , 0, 0, 0 ,0);
    }
}
// call user defined function
urk(4);
// setup cyclic timer every 2 seconds
timerId = util.every(2000, “urk(10)“ );
```

This example is installed with the CDE as example1.js.

Script commands can be entered into the command line at the bottom of the window. Complete script files can be loaded as well if a path to a script file is specified in the command line. The command line stores the history and using the cursor buttons up and down older commands can be selected again. The command line history is stored when the program is closed.

8 Support & Contact

On all questions and upcoming problems on CANinterpreter you may contact us via email (support@emotas.de) or by phone +49(0)3461/794160. If a CANopen device does not react as expected, a logging of the CAN communication is useful for the analysis. Please send us your current CAN logging by email, ideally also before you contact us by phone.