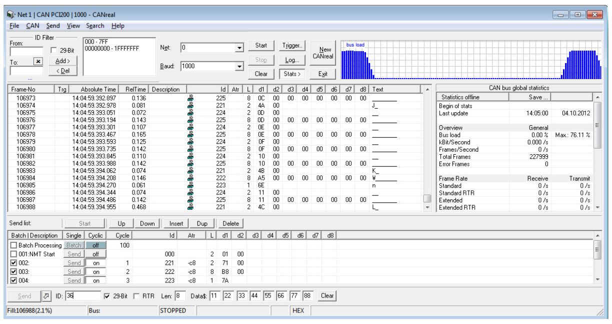
CANreal

Tool for Testing and Monitoring CAN Networks



Frame-No	Trg	Absolute Time	RelTime	Description	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
106973		14:04:59.392.897	0.136		225		8	0C	00	00	00	00	00	00	00	
106974		14:04:59.392.978	0.081		221		2	4A	00							J_
106975		14:04:59.393.051	0.072		224		2	0D	00							
106976		14:04:59.393.194	0.143		225		8	0D	00	00	00	00	00	00	00	
106977		14:04:59.393.301	0.107		224		2	0E	00							
106978		14:04:59.393.467	0.165		225		8	0E	00	00	00	00	00	00	00	
106979		14:04:59.393.593	0.125		224		2	0F	00							
106980		14:04:59.393.735	0.142		225		8	0F	00	00	00	00	00	00	00	
106981		14:04:59.393.845	0.110		224		2	10	00							
106982		14:04:59.393.988	0.142		225		8	10	00	00	00	00	00	00	00	
106983		14:04:59.394.062	0.074		221		2	4B	00							K_
106984		14:04:59.394.208	0.146		222		8	A5	00	00	00	00	00	00	00	¥
106985		14:04:59.394.270	0.061		223		1	6E								n
106986		14:04:59.394.344	0.074		224		2	11	00							
106987		14:04:59.394.486	0.142		225		8	11	00	00	00	00	00	00	00	
106988		14:04:59.394.955	0.468		221		2	4C	00							L_

CAN bus global statistics

Statistics offline	Save ...	
Begin of stats		
Last update	14:05:00	04.10.2012
Overview	General	
Bus load	0.00 %	Max.: 76.11 %
kBit/Second	0.000 /s	
Frames/Second	0 /s	
Total Frames	227999	
Error Frames	0	
Frame Rate	Receive	Transmit
Standard	0 /s	0 /s
Standard RTR	0 /s	0 /s
Extended	0 /s	0 /s
Extended RTR	0 /s	0 /s

Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch Processing	Batch	off	100											
☐ 001:NMT Start	Send	off		000		2	01	00						
☑ 002:	Send	on	1	221	-c8	2	71	00						
☑ 003:	Send	on	2	222	-c8	8	B8	00						
☑ 004:	Send	on	3	223	-c8	1	7A							

Software Manual

to Product C.1107.01

esd electronics gmbh
Vahrenwalder Str. 207 • 30165 Hannover • Germany
http://www.esd.eu
Phone: +49 (0) 511 3 72 98-0 • Fax: +49 (0) 511 3 72 98-68

esd electronics gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

Phone:	+49-511-372 98-0
Fax:	+49-511-372 98-68
E-Mail:	info@esd.eu
Internet:	www.esd.eu

Document file:	I:\Texte\Doku\MANUALS\PROGRAM\CAN\CAN-Tools\CANreal\Englisch\CANreal_Manual_en_33.odt
Date of print:	2019-08-23
Document type number:	DOC0800

Software version:	8.52

Document History

The changes in the document listed below affect changes in the hardware as well as changes in the description of the facts, only.

Rev.	Chapter	Changes versus previous version	Date
3.2	-	Classification of Messages inserted	2017.08.16
	1.2	Program call CANreal – log	
	1.3	Figures updated	
	2.1	Figure updated	
	2.3	Attributes for CAN FD inserted	
	2.3.1	Figure and description updated	
	2.4.1	Chapter supplemented: CAN FD, Configurator	
	3, 3.1, 3.2	Figures updated	
	3.2.1	New chapter „Bitrateconfigurator“	
	3.2.4	Description of CAN FD inserted	
	3.2.7.3	Context menu- Figure and description updated	
	3.2.10	New chapter “Enable custom NTCAN events”	
	3.4, 3.5	Figures updated	
	3.5.1, 3.5.2	Chapter moved, *Find* and *Go to* now under menu item *Search*	
	3.5.3	New Chapter Bookmarks	
	3.6.2	updated	
	4.3.3.2	Figure new	
	4.4	New Chapter „Internal Plugin“ for CAN FD	
	5.	Examples and descriptions updated	
3.3	7.	New Chapter: „Troubleshooting“ inserted	2019-08-23

Technical details are subject to change without further notice.

Classification of Notes

This manual contains noticeable descriptions for a safe use of the CANreal and important or useful information.

NOTICE

Notice statements are used to notify people on hazards that could result in things other than personal injury, like property damage.

NOTICE
This NOTICE statement contains the general mandatory sign and gives information that must be heeded and complied with for a safe use.

INFORMATION

INFORMATION
Notes to point out something important or useful.

Typographical Conventions

Throughout this manual the following typographical conventions are used to distinguish technical terms.

Convention	Example
File and path names	**`/dev/null`** or **`<stdio.h>`**
Function names	***open()***
Programming constants	`NULL`
Programming data types	`uint32_t`
Variable names	*Count*

Number Representation

All numbers in this document are base 10 unless designated otherwise. Hexadecimal numbers have a prefix of 0x. For example, 42 is represented as 0x2A in hexadecimal format.

Table of contents

1. Overview

CANreal is a menu-controlled program which is used for monitoring and testing CAN networks. Its self-explaining user interface offers a quick lead-in into the way the CAN network works.
The functionality of CANreal can be extended by plugins.

This manual explains in detail the individual menus and functions of CANreal. The description is followed by application examples.

1.1 System Requirements

- Windows XP or later
- 512 MB RAM
- 25 MB free hard disk space
- CAN driver from esd

1.2 Program Call

The tool CANreal is contained in the esd CAN Software Development Kit (CAN SDK), which is distributed with the esd-CAN-CD or can be downloaded from the esd-homepage (www.esd.eu).
At the installation of the SDK (Software Development Kit) the program CANreal is automatically installed.

Start the SDK-installation file `Can_sdk\setup.exe` on the esd-CAN-CD and carry out the installation. If not defined at the installation differently, after successful installation the program CANreal can be started under Windows by selecting the menu items *Start / Program files / CAN / CANreal*.

NOTICE
CANreal can run parallel to other CAN-applications and e.g. display the identifiers used there or transmit on any identifier. It is possible to run multiple CANreal instances simultaneously.

1.2.1 Command line parameters

When the program is called, parameters can be specified in the command line:

Call	Function
`CANreal --start`	CANreal is called and initialised so that the received CAN messages are immediately displayed.
`CANreal` *`profilname`*	CANreal is called and the parameter settings stored under *profilname* are adopted.
`CANreal` *`profilname`* `--start`	Combination of both calls above.

Call	**Function**
`CANreal --convert` {List with file and/or directory names}	Corresponds to the menu item *File / Convert logfiles to text* (see chapter “3.1.7 Convert Logfiles to Text ”) For the list of file and /or directory names search paths can be entered with ‘*’ or ‘?’ (e.g.: *Name*.**). With *Convert logfiles to text* an existing logfile (`*.csplog`), in which the received messages are stored in binary format, can be selected. The logfile will be converted into a readable text file (`*.txt`).

`CANreal —-id2description`
{File name}
	Corresponds to the menu item *File / Application / Text description mapping for CAN identifiers*
	(see chapter “3.1.4.1 Application”)
	In the field *Text description mapping for CAN identifiers* a file with descriptive text can be assigned to an identifier.

`CANreal —-prio <nr>`	Via menu item *File / Advanced settings / Application* (see chapter “3.1.4.1 Application”) the *Application priority* can be set, if this option is selected. <nr>=0,1,2 0 : lowest priority 1 : higher priority 2 : highest priority
`CANreal profilname --trigger`	Enables the trigger (see chapter “2.7 Trigger and Logging”)
`CANreal --log`	Enables the recording (without Trigger) (see 2.7 Trigger and Logging , page 21)

Table 1: Calling CANreal via parameters

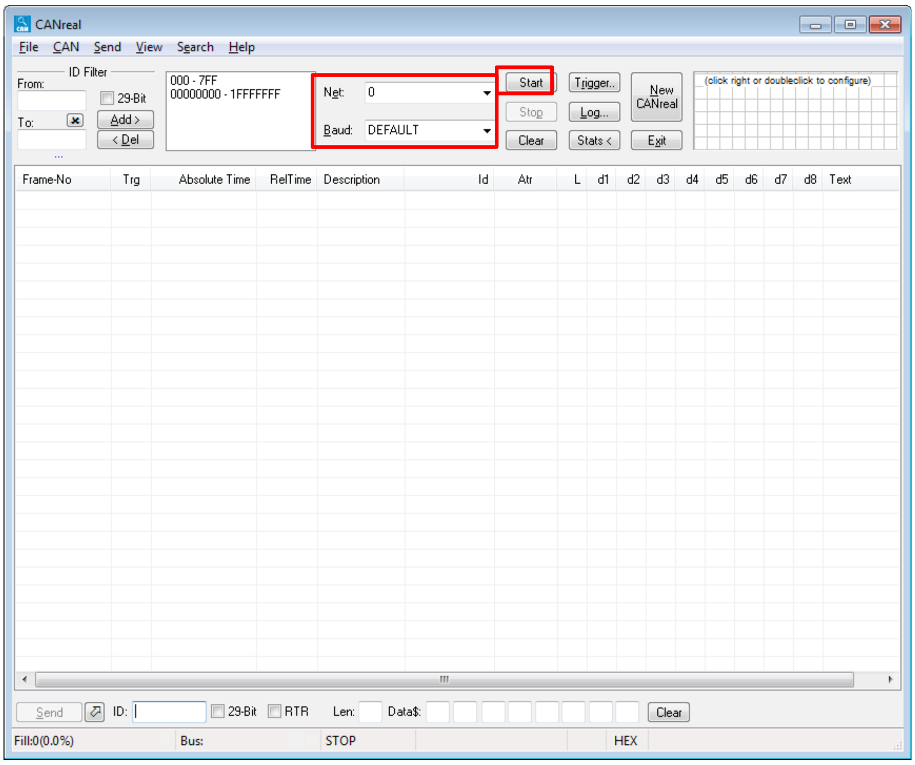
1.3 Quick Start

This chapter contains a short description for a quick start-up. For further information a detailed description of CANreal is given in this manual. After installation the CANreal window looks like this:

Please proceed the following steps:

1. Choose the desired CAN net via *Net:* and the correct baud rate setting via *Baud:*
2. Press the button labelled *Start* (the label will change to *Pause*)

Now CANreal will log the entire CAN traffic from the chosen CAN bus.

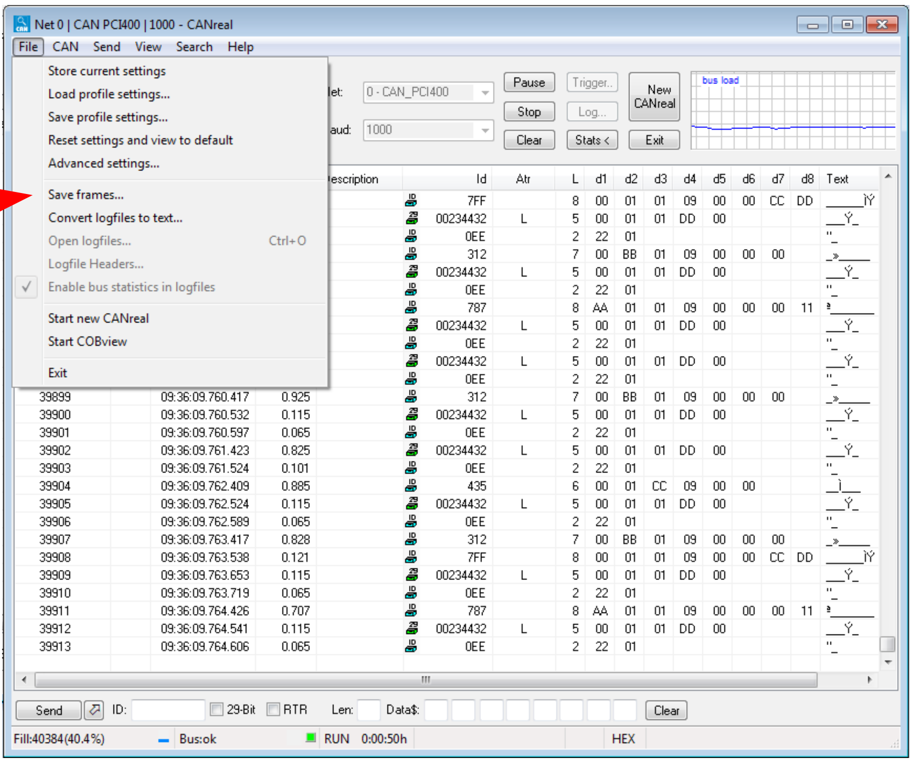

CANreal
File CAN Send View Search Help
ID Filter
From:
To:
29-Bit
Add >
< Del
000 - 7FF
00000000 - 1FFFFFFF
Start
Stop
Clear
Trigger..
Log...
Stats <
New CANreal
Exit
(click right or doubleclick to configure)
Frame-No Trg Absolute Time RelTime Description Id Atr L d1 d2 d3 d4 d5 d6 d7 d8 Text
Send
ID:
29-Bit
RTR
Len:
Data$:
Clear
HEX

Net: 0
Baud: DEFAULT

Fill:0(0.0%) Bus: STOP

Figure 1: CANreal Window

When you have proceeded the steps and all data has been acquired, you can select the menu item *File* and then *Save frames...* to save your data (see Figure 2)



Net: 0 - CAN_PCI400
Baud: 1000

Frame-No	Absolute Time	RelTime	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
			7FF		8	00	01	01	09	00	00	CC	DD	______ÌŸ
			00234432	L	5	00	01	01	DD	00				___Ý_
			0EE		2	22	01							"_
			312		7	00	BB	01	09	00	00	00		_»_____
			00234432	L	5	00	01	01	DD	00				___Ý_
			0EE		2	22	01							"_
			787		8	AA	01	01	09	00	00	00	11	ª_______
			00234432	L	5	00	01	01	DD	00				___Ý_
			0EE		2	22	01							"_
			00234432	L	5	00	01	01	DD	00				___Ý_
			0EE		2	22	01							"_
39899	09:36:09.760.417	0.925	312		7	00	BB	01	09	00	00	00		_»_____
39900	09:36:09.760.532	0.115	00234432	L	5	00	01	01	DD	00				___Ý_
39901	09:36:09.760.597	0.065	0EE		2	22	01							"_
39902	09:36:09.761.423	0.825	00234432	L	5	00	01	01	DD	00				___Ý_
39903	09:36:09.761.524	0.101	0EE		2	22	01							"_
39904	09:36:09.762.409	0.885	435		6	00	01	CC	09	00	00			__Ì___
39905	09:36:09.762.524	0.115	00234432	L	5	00	01	01	DD	00				___Ý_
39906	09:36:09.762.589	0.065	0EE		2	22	01							"_
39907	09:36:09.763.417	0.828	312		7	00	BB	01	09	00	00	00		_»_____
39908	09:36:09.763.538	0.121	7FF		8	00	01	01	09	00	00	CC	DD	______ÌŸ
39909	09:36:09.763.653	0.115	00234432	L	5	00	01	01	DD	00				___Ý_
39910	09:36:09.763.719	0.065	0EE		2	22	01							"_
39911	09:36:09.764.426	0.707	787		8	AA	01	01	09	00	00	00	11	ª_______
39912	09:36:09.764.541	0.115	00234432	L	5	00	01	01	DD	00				___Ý_
39913	09:36:09.764.606	0.065	0EE		2	22	01							"_

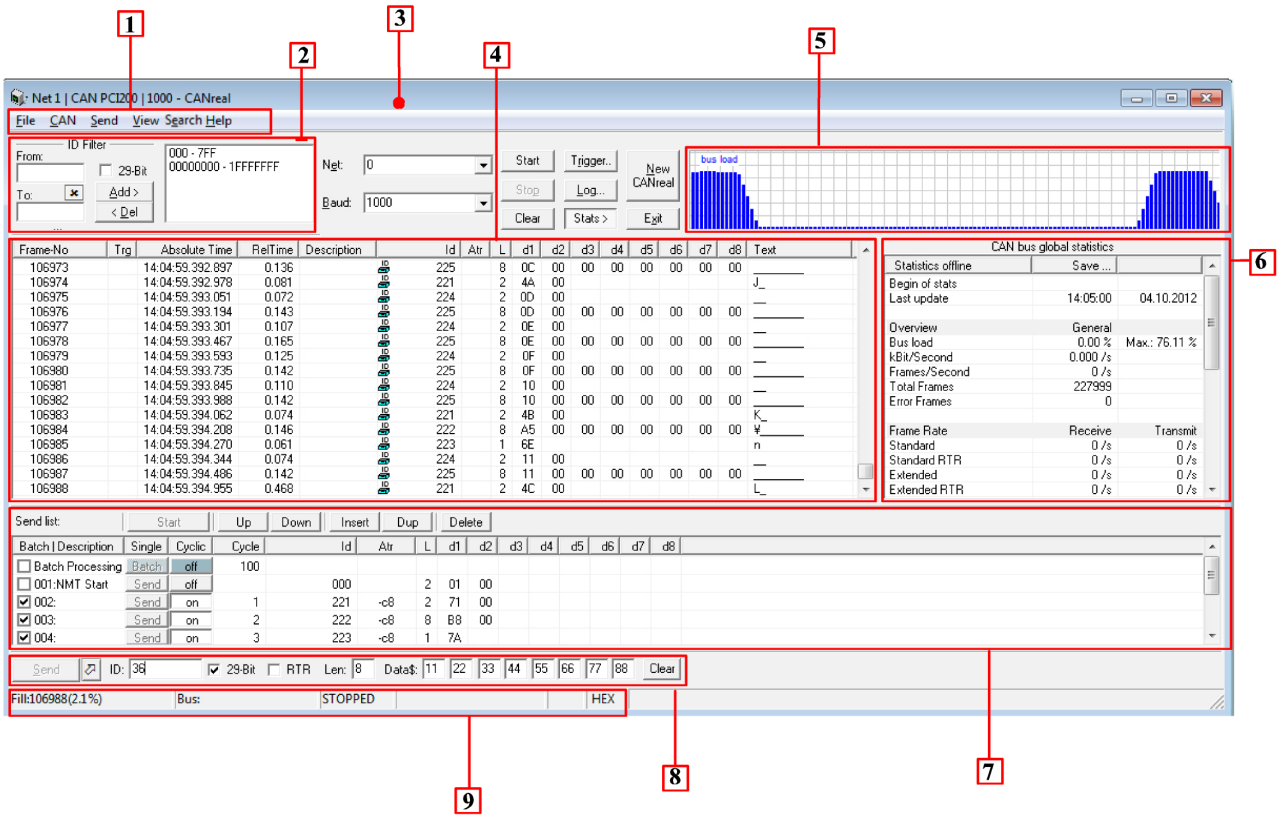
Fill:40384(40.4%) Bus:ok RUN 0:00:50h

Figure 2: CANreal menu item *File* (Example)

2. Functions of the User Interface Elements

2.1 Display of the CANreal Window

The CANreal program window is structured as shown below:



Frame-No	Trg	Absolute Time	RelTime	Description	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
106973		14:04:59.392.897	0.136		225		8	0C	00	00	00	00	00	00	00	________
106974		14:04:59.392.978	0.081		221		2	4A	00							J_
106975		14:04:59.393.051	0.072		224		2	0D	00							__
106976		14:04:59.393.194	0.143		225		8	0D	00	00	00	00	00	00	00	________
106977		14:04:59.393.301	0.107		224		2	0E	00							__
106978		14:04:59.393.467	0.165		225		8	0E	00	00	00	00	00	00	00	________
106979		14:04:59.393.593	0.125		224		2	0F	00							__
106980		14:04:59.393.735	0.142		225		8	0F	00	00	00	00	00	00	00	________
106981		14:04:59.393.845	0.110		224		2	10	00							__
106982		14:04:59.393.988	0.142		225		8	10	00	00	00	00	00	00	00	________
106983		14:04:59.394.062	0.074		221		2	4B	00							K_
106984		14:04:59.394.208	0.146		222		8	A5	00	00	00	00	00	00	00	¥_______
106985		14:04:59.394.270	0.061		223		1	6E								n
106986		14:04:59.394.344	0.074		224		2	11	00							__
106987		14:04:59.394.486	0.142		225		8	11	00	00	00	00	00	00	00	________
106988		14:04:59.394.955	0.468		221		2	4C	00							L_

CAN bus global statistics

Statistics offline	Save ...	
Begin of stats		
Last update	14:05:00	04.10.2012
Overview	General	
Bus load	0.00 %	Max.: 76.11 %
kBit/Second	0.000 /s	
Frames/Second	0 /s	
Total Frames	227999	
Error Frames	0	
Frame Rate	Receive	Transmit
Standard	0 /s	0 /s
Standard RTR	0 /s	0 /s
Extended	0 /s	0 /s
Extended RTR	0 /s	0 /s

Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch Processing	Batch	off	100											
☐ 001:NMT Start	Send	off		000		2	01	00						
☑ 002:	Send	on	1	221	-c8	2	71	00						
☑ 003:	Send	on	2	222	-c8	8	B8	00						
☑ 004:	Send	on	3	223	-c8	1	7A							

Figure 3: Structure of the CANreal window

Description

1. Menu Bar
2. Add/Delete ID Area (11-Bit Identifier)
3. Title Bar
4. Display Window for received Messages
5. Graph to display the bus load
6. *CAN bus global statistics* (see Show Statistics)
7. Send List
8. Input Bar for Send Messages
9. Status Display

2.2 Title Bar

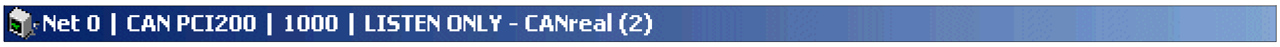

Net 0 | CAN PCI200 | 1000 | LISTEN ONLY - CANreal (2)

Figure 4: Title bar (example)

In the title bar the following details are displayed (from left to right):

Examples from the title bar	Meaning
Net 0	Net number
CAN-PCI/200	CAN device in use
1000	Baud rate (or *Auto*, as long as the automatic baud rate detection has not been completed)
STATIC	Optional information about statistics
LISTEN ONLY	Optional information about Listen-Only-Mode
CANreal(2)	Number of the CANreal program instance in brackets (here e.g.: 2)

Table 1: Description of the title bar

2.3 Display Window for received Messages

In the display window the selected CAN messages are displayed. The messages to be displayed are chosen via the CAN identifiers. For this the CAN identifier range has to be set via the *Add Area* button and the corresponding input field.

Each message received is listed in a new line (scroll down button). To optimize the column width in the title bar click left on the table separator and move it to the new position.

Column	Format	Description
Frame-No	decimal	**Message number** Each of the messages to be displayed gets a serial number Static mode: Number of the frames received with the identifier specified under *Id* and the attribute specified under *Atr*.
Trg	S, E, T, N	Trigger condition for CAN-messages (see chapter "2.7 Trigger and Logging") S... *Start Trigger,* T... *Time Period* E... *End Trigger,* N... *Number of messages*
Absolute Time	HH:MM:SS:ms	**Time flag (absolute)** In this column the time is shown in which the message displayed has been received, counted from the first message received. If HW timestamp is active (see chapter "Time", page 30), milliseconds and microseconds are displayed (HH:MM:SS.ms.µs)

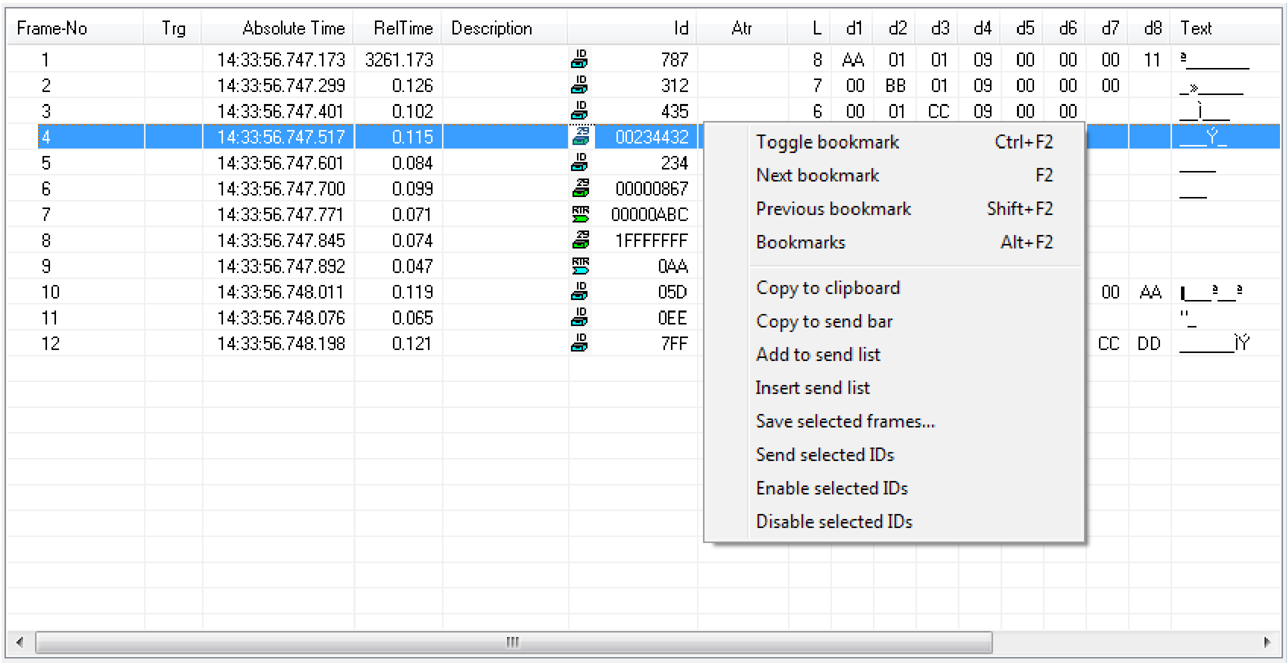
RelTime	decimal milliseconds	If HW timestamp is supported a higher accuracy can be achieved (depending on hardware, ≤ 10 µs). The HW timestamp is active, milliseconds and microseconds are displayed (ms.µs). For SW timestamp The resolution depends on the operating system (≥ 10 ms).
Description	Text	**Description of the identifier** If *Single Error Diagnostic* is active: Displayed as NTCAN-Event
Id	decimal or hexadecimal	**CAN identifier** The symbols in this column next to the CAN identifier display the attributes of the identifiers: blue - 11-bit identifier green - 29-bit identifier blue - 11-bit identifier with RTR green - 29-bit identifier with RTR ⚠ Event frame <NTCANEvent>: 0... Controller Events 1... Baud change (see chapter “2.4.1 Baud Rate”) 2... ECC-Event-ID (single error diagnostic only, see chapter “3.2.9 Single Error Diagnostic”)
Atr	L, R, E, F, F-B	**Attribute** L... 29-bit-ID R... RTR-bit E... change of status of CAN: *ok, warn* or *error* (for event frames) F... CAN FD F-B... CAN FD without baudrate switch
L	0...8	Number of valid data bytes of the message
d1 ... d8	hexadecimal	Data bytes, hexadecimal (two digit)
Text	ASCII	The data received are shown in ASCII text

Table 2: Description of the entries in display window

INFORMATION
A high accuracy of time differences (*RelTime*) can only be achieved with CAN devices and drivers, which support the hardware timestamp option (see NTCAN Part 1: Application Developers Manual [1], chapter: “Timestamps”).

2.3.1 Context Menu

To open the context menu of the window for received messages, at least one frame has to be selected. A click with the right mouse button into the display window opens the context menu.



Frame-No	Trg	Absolute Time	RelTime	Description	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
1		14:33:56.747.173	3261.173		787		8	AA	01	01	09	00	00	00	11	ª______
2		14:33:56.747.299	0.126		312		7	00	BB	01	09	00	00	00		_»_____
3		14:33:56.747.401	0.102		435		6	00	01	CC	09	00	00			__ì___
4		14:33:56.747.517	0.115		00234432											___Ý_
5		14:33:56.747.601	0.084		234											____
6		14:33:56.747.700	0.099		00000867											___
7		14:33:56.747.771	0.071		00000ABC											
8		14:33:56.747.845	0.074		1FFFFFFF											
9		14:33:56.747.892	0.047		0AA											
10		14:33:56.748.011	0.119		05D									00	AA	I__ª__ª
11		14:33:56.748.076	0.065		0EE											''_
12		14:33:56.748.198	0.121		7FF									CC	DD	______ÌY

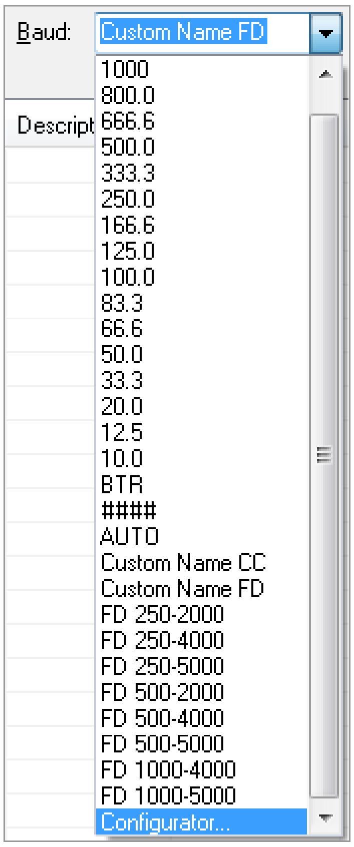
Figure 5: Context menu of the display window for received messages

Commands of the context menu	Description	
Toggle bookmark	Sets/Deletes the bookmark in the selected row (row with blue background)	[Ctrl]+ [F2]
Next bookmark	Go to the next bookmark (see page 63)	[F2]
Previous bookmark	Go to the previous bookmark	[Shift] + [F2]
Bookmarks	Open the bookmark window, see chapter „Bookmark“, page 63	[Alt] + [F2]
Copy to clipboard	Copies the selected CAN message to clipboard.	
Copy to send bar	Inserts the selected CAN message into the input bar for messages that shall be transmitted.	
Add to send list	Adds the CAN message to the end of the *Send list*.	
Insert send list	CAN messages are inserted at a selected position of the *Send list*.	
Save selected Frames	The selected CAN messages are saved in a file	
Send selected IDs	The selected CAN messages are transmitted again	
Enable selected IDs	The selected IDs are enabled	
Disable selected IDs	The selected IDs are disabled	

Table 3: Commands of the context menu

2.4 Description of the Buttons

2.4.1 *Baud Rate*



Here you can set the baud rate. You can select one of the predefined baud rate values, or:

DEF ... If the specification DEF is selected, CANreal will operate with the baud rate, set via the CAN driver. This is useful if the baud rate has already been defined by another running application.

BTR01 ... If the specification BRT01 is selected, a four-digit hexadecimal value for the CAN controller register BTR0 and BTR1 can be entered.

... Until a value is specified for BTR01, #### is shown in the list. If a value is specified, always the last specified value of BTR01 is shown instead of ####.

AUTO... An automatic baud rate detection can be selected, if the board supports this function. An overview can be taken from the table “CAN driver features” in the chapter: “Operating System Support” of “NTCAN, Part 1: Application Developers Manual” [1].
Select AUTO in the field *Baud rate:* and start the program via the button *Start*. Now the driver tries to determine a predefined baud rate on the CAN bus.

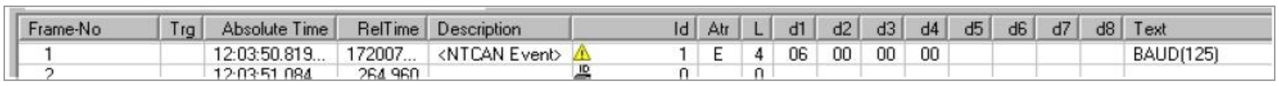
It is required that messages have already been transferred in the CAN net independent of CANreal. Otherwise the AUTO mechanism can **not** determine a baud rate!
The AUTO-baud mode will be active, until a standardized baud rate is detected.
Then the baud rate will be transferred automatically and the program CANreal will be run with this baud rate.

FD ... Pre-defined baud rates for CAN FD (2 baud rates, CAN FD only)

Configurator... The baud rate can be edited with the *baudrate configurator*. See chapter „Bitrate Configurator“, page 34.

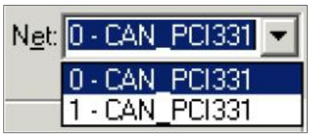
Furthermore it is possible to enter the baud rate (with decimal point, e.g. 300.0) in the field *Baud:*. If you receive a warning because of invalid entries entries, you have to check your settings.

A change of the baud rate is always reported as NTCAN event (Id: 1, for *Baud Change Event*). The new baud rate will be shown in brackets in the column *Text*. The current baud rate is displayed in the title bar of the program window.



Frame-No	Trg	Absolute Time	RelTime	Description		Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
1		12:03:50.819...	172007...	<NTCAN Event>	⚠	1	E	4	06	00	00	00					BAUD(125)
2		12:03:51.084	264.960			0		0									

2.4.2 Net Number of the CAN Device



In this input box the network number used for the board can be entered. The first board in the PC normally has the network numbers 0 and 1 (only the 0 for a one-channel board), every further board has network numbers which are accordingly rising (2 and 3, 4 and 5 ... for one-channel boards 2, 4 ...). The name of the board is shown with the available network numbers in the list *Net:*.

2.4.3 Add/Delete ID Area (11-Bit Identifier)

Here the lower and upper limit of the ID area to be activated or deactivated is specified.
It is permissible to activate or deactivate ID areas which are already active or inactive. The IDs have to be specified with hexadecimal values. If you want to enter the values in decimal format *click* on HEX in the status bar as described in chapter “Status Display” page 20.

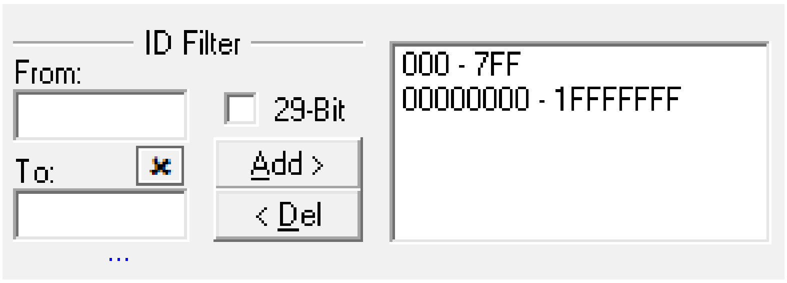


Figure 6: Add/Delete ID Area

Description of the buttons

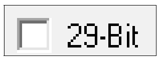
Add >
< Del

By means of these buttons the ID area which is shown in the input boxes is activated or deactivated and the corresponding CAN messages are shown in the display window.

Delete

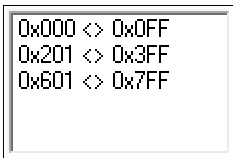
Click this button to delete the entries in the fields from and to.

29-Bit Identifier



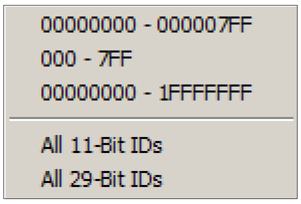
If you enable this checkbox, you can activate or deactivate messages with 29-bit CAN identifiers.
Of course, the hardware and the CAN driver have to be able to work with 29-bit CAN identifiers.

Active ID's



In this window the active ID areas are shown.
A single click on an entry with the left mouse button accepts the value range to the specification fields *from* and *to*.
A double click on an entry deletes the selected ID area.

[...]



With the [...] button you can open a pull down menu which contains a list of the recently used ID areas.
Additionally you can choose the options *All 11-Bit IDs* and *All 29-Bit IDs.*

2.4.4 Start/Stop, Trigger Buttons

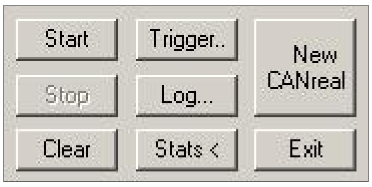


Figure 7: View of the buttons

Description of the buttons

Start/Pause/Resume

Clicking the *Start* button starts the listing of received messages in the display window.
Clicking the *Start* button again switches to *Pause*. In this status the listing of the received messages pauses without stopping the trigger- and logging function.
The messages are further being recorded in the background!
Clicking the *Resume*-button again ends the pause and the listing of the received data continues.

Stop

Clicking the *Stop*-button stops the recording and the listing of received messages in the display window and the trigger- and logging-functions are completed.

Clear

Clicking the *Clear*-button deletes all CAN message objects in the window and the message counter (*Frame-No.*) is reset to zero.

Trigger..

Clicking this icon opens the dialogue box *Trigger and Logging* and conditions can be specified, which start (*Start Trigger*) or end (*End Trigger*) the recording of the messages in files (see chapter “2.7 Trigger and Logging”). In addition properties of the file in which the messages are stored can be specified with *Output*.

Log...

Clicking the *Log*-button opens the dialogue box *Output*. The properties of the file, in which the messages are stored can be specified here.

Stats<

By clicking this button CAN bus statistics will be displayed.

Exit

Clicking this button closes the file and ends the program CANreal.

New CANreal

Clicking the *New CANreal* button starts a new instance of the program and opens a further CANreal program window. The number of the instance will be shown in the title bar.

2.5 Input Bar for Send Messages

The data of the CAN messages to be transmitted have to be specified in the input bar at the lower window frame. The same data formats as for the display window are valid. Values outside of the valid value range will not be accepted.
Data can also be taken from the display window for received messages in the display menu (see chapter "2.3.1 Context Menu").

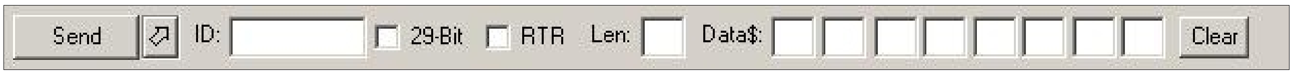


Figure 8: Input bar for messages to be transmitted

Description

Send

By clicking this button the data specified in the input bar are transmitted.

Add to *Send List*

By clicking this button the data specified in the input bar for messages to be transmitted are inserted as new entry at the end of the *Send list*.

Clear

By clicking this button once the specified transmit data in the input bar for transmit data are deleted.
By clicking this button again the entries for CAN identifier and length are deleted as well.

2.6 Status Display

This status bar shows the status of the program in operation.

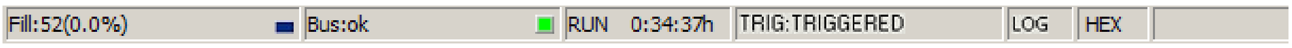


Figure 9: View of status bar

Field	Status	Description	
Display mode	Fill:XY(X.Y%),	Total number of messages in the list (fill level in %), displayed in scroll mode	
	STATIC	Static mode is enabled.	
CAN-Bus activity: (blue bar)	-	The current bus load is indicated as a blue bar. If you move the mouse over the bar a tooltip with the specification (in percent) of the bus load as numerical value is shown. If a BTR-baud rate is set, the bus activity is not available and ‘0' is indicated.	
Bus (Bus status)	ok, WARN!, OFF!	Shows the status of the CAN controller	
		Status field	Status
		green	ok
		yellow	WARN!
		red	OFF!
	...-Lost:n	Loss of n messages n... number (see chapter “5.4 Example for a Logfile with Lost frames“)	
Status	STARTED	Recording is started	
	PAUSED	Recording pauses	
	STOPPED	Recording is stopped	
TRIG	WAITING	Waiting for trigger	
	TRIGGERED	Trigger started	
	DELAYED	End trigger occurred, but messages are still recorded	
	ENDED	End trigger occurred	
LOG	-	The option *Log to File* is active	
Number format	HEX	Hexadecimal values	
	DEC	Decimal values	

In *Listen-Only-Mode* the background of the status bar is highlighted in a way that depends on the colour scheme used.

2.7 Trigger and Logging

Clicking the button *Trigger*, opens the dialogue window *Trigger and Logging*. The button *Log...* will only open the tab *Output*. Via the menu item *CAN* in the main menu the entries *Trigger* & *logging...* and *Logging...* with corresponding functions can be chosen. The calls are only available if CANreal is in *Stop* status.

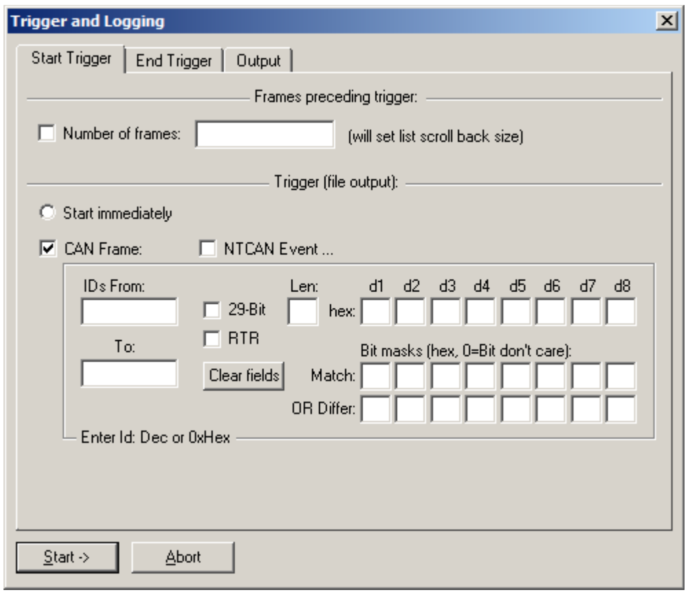
2.7.1 Start Trigger

The conditions which start (trigger) the recording of the messages in a file can be specified here.

In the field *Frames preceding Trigger* a *Number of frames* (decimal) can be specified that precede the trigger condition. The preceding CAN messages are stored in a file with the file name:
`????-pretrig.csplog`

Start immediately starts the recording of the CAN messages directly without trigger.

Activate the checkbox *CAN Frame* to define a trigger condition.
As trigger condition you can select identifier (*IDsFrom, To*), *29-Bit* identifier, active *RTR*-bit, the length *Len* of the message (*Len*: 0-7) and defined data bytes (*d1-d8*).



Trigger and Logging

Start Trigger | End Trigger | Output

Frames preceding trigger:

☐ Number of frames: (will set list scroll back size)

Trigger (file output):

○ Start immediately

☑ CAN Frame: ☐ NTCAN Event ...

IDs From: ☐ 29-Bit Len: d1 d2 d3 d4 d5 d6 d7 d8 hex:

To: ☐ RTR Bit masks (hex, 0=Bit don't care):

Clear fields Match:

OR Differ:

Enter Id: Dec or 0xHex

Start -> Abort

Figure 10: Start Trigger

Only the fields with an entry restrict the selection. If *RTR* is chosen, the entries of the length and the data are always invalid.

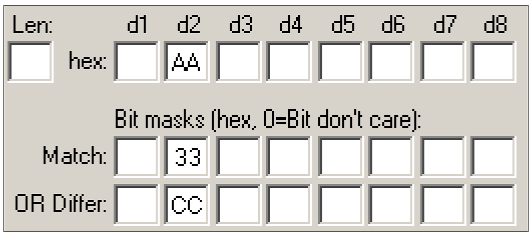
In addition the selection with the data-bit masks *Match* or *OR Differ* is possible. Only specified data bytes of *d1-d8* must match or differ from the bytes specified (see chapter “2.7.1.1 Selection via Data-Bit Mask”). The chosen data bits have the value ‘1’ in the mask. Data bytes which are not selected are greyed out.

The recording of the messages can also be triggered by an *NTCAN Event*, see chapter “2.7.1.2 NTCAN Event Trigger”.

Use the button *Clear fields* to delete the trigger conditions.

2.7.1.1 Selection via Data-Bit Mask

Additionally the selection can be done via data bit-masks, in which only specified data bits of the data bytes *d1-d8* have to match (*Match:*) or to differ (*OR Differ*).



		d1	d2	d3	d4	d5	d6	d7	d8
Len:	hex:		AA						
	Bit masks (hex, 0=Bit don't care):								
	Match:		33						
	OR Differ:		CC						

Figure 11: Data-bit mask (Example)

Match:	Trigger if the bit positions of the received CAN data masked with '1' match the bit values defined in d1..d8.
OR Differ:	Trigger if the bit positions of the received CAN data masked with '1' differ from the bit values defined in d1..d8.

Example for the selection with data-bit masks (with single data byte):

specified data byte :	d2 = AA$_h$ (1010 1010$_b$)
data byte mask *Match*:	M2= 33$_h$ (0011 0011$_b$)
data byte mask *Differ*:	D2= CC$_h$ (1100 1100$_b$)

Examples for data-bit mask *Match*:

Bit number:	7	6	5	4	3	2	1	0	
Data in trigger dialogue d2:	1	0	1	0	1	0	1	0	
Mask ***Match*** M2:	0	0	**1**	**1**	0	0	**1**	**1**	→ only bits 0,1,4,5 are compared
Received CAN data Example 1:	x	x	1	0	x	x	1	0	All four bits match with d2 → *Match* condition is fulfilled → **Trigger**
Received CAN data Example 2:	x	x	1	**1**	x	x	**0**	0	Two bits differ from d2 → *Match* condition is not fulfilled, → **No trigger**

The data bits of the CAN data marked with x are not relevant for the evaluation as defined in the mask M2. The data bits of the CAN data selected with ‘1’ in the mask M2 match the data bits of d2 only in the example 1 (trigger condition fulfilled).

Examples for data-bit mask *Differ:*

Bit number:	7	6	5	4	3	2	1	0	
Data in trigger dialogue d2:	1	0	1	0	1	0	1	0	
Mask ***Differ*** *D2*:	**1**	**1**	0	0	**1**	**1**	0	0	→ only bits 2,3,6,7 are compared
Received CAN data Example 3:	1	0	x	x	1	0	x	x	All four bits match with d2 → *Differ* condition is not fulfilled → **No trigger**
Received CAN data Example 4:	1	**1**	x	x	1	0	x	x	One bit differs from d2 → *Differ* condition is fulfilled → **Trigger**

The data bits of the CAN data marked with x are not relevant for the evaluation as defined in the mask D2. The data bits of the CAN data selected with ‘1’ in the mask D2 differ from the data bits of d2 only in the example 4 (trigger condition fulfilled).

INFORMATION
Please note that it does not make sense to mask the same bits with mask *Match* and mask *Differ*!

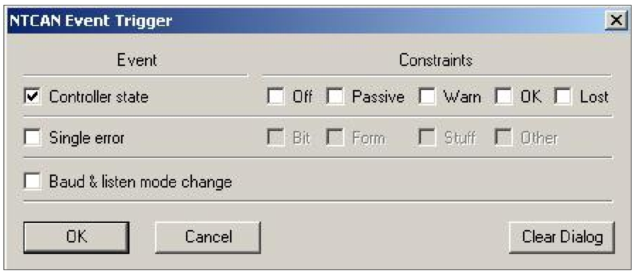
2.7.1.2 NTCAN Event Trigger

If the checkbox *NTCAN Event...* is enabled (see Figure 10), the configuration menu *NTCAN Event Trigger* is opened.

Controller state
Trigger on the error status of the CAN controller (for bus status see chapter “2.6 Status Display”).

Single Error
Trigger on single errors (see chapter “3.2.9 Single Error Diagnostic”)

Baud & listen mode changed
Trigger on *Baud rate changed* and *Listen only mode changed* (see NTCAN Part 1: Application Developers Manual [1], chapter “*Listen-Only Mode”*). Useful e.g. if a running application causes the changes.


NTCAN Event Trigger

Event	Constraints
☑ Controller state	☐ Off ☐ Passive ☐ Warn ☐ OK ☐ Lost
☐ Single error	☐ Bit ☐ Form ☐ Stuff ☐ Other
☐ Baud & listen mode change	

Figure 12: NTCAN Event Trigger

2.7.2 End Trigger

In this window the conditions can be specified which stop the recording of the messages in files if a trigger condition is met.

The recording can be automatically stopped after a specified *Number of frames* or a specified *Time period*.

Activate the checkbox *CAN Frame* and define a trigger condition.
As trigger condition you can select identifier (*IDs From, To*), *29-bit* identifier (*29-Bit*), active *RTR*-bit, the length of the messages (*Len*) and defined data bytes (*d1-d8*).
Only the fields with an entry restrict the selection.

If *RTR* is chosen, the entries of the length and the conditions data are always invalid.

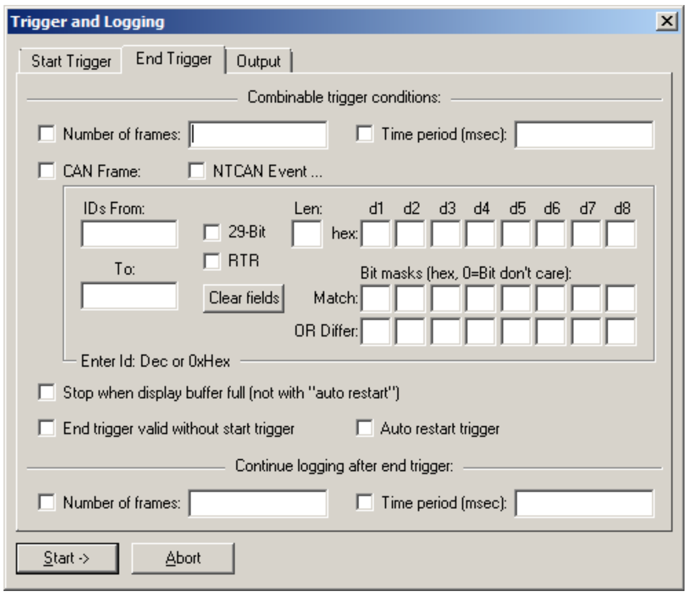


Figure 13: End trigger

In addition the selection with the data-bit masks *Match* or *OR Differ* is possible. Only specified data bytes *d1-d8* must match or differ from the bytes specified (see chapter “2.7.1.1 Selection via Data-Bit Mask“). The chosen data bits have the value ‘1’ in the mask. Data bytes which are not selected are greyed out.

The recording of the messages can also be triggered by an *NTCAN Event*, see chapter “2.7.1.2 NTCAN Event Trigger”.

Use the button *Clear fields* to delete the trigger conditions.

Stop when display buffer full stops the recording if buffer for the display window is full.
With *End trigger valid without start trigger* the end trigger condition is valid even if the start trigger condition has not already occurred.
By selecting *Auto restart trigger* the trigger cycle is automatically restarted after completion of the *End Trigger* condition.

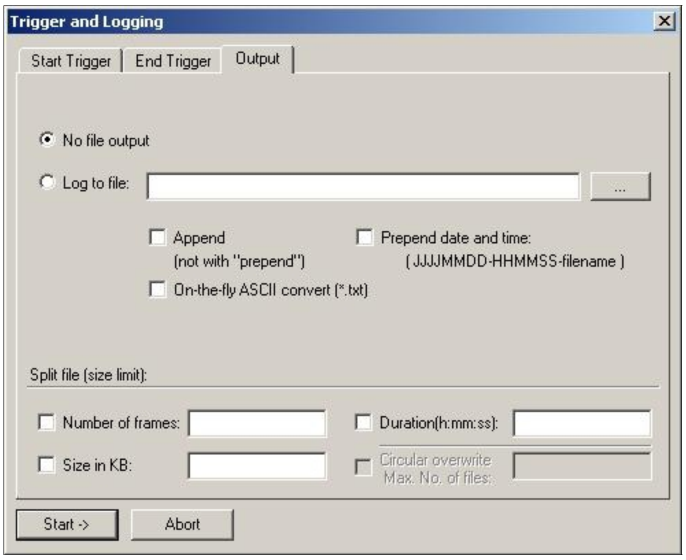
In the field *Continue Logging after end trigger* a *Number of Frames* or a *Time period* can be specified, in which the recording is continued after the end trigger condition occurred. But the logfile will not be closed after time period, if no CAN frame is received.

2.7.3 Output

In this window the properties of the message record file can be specified.
If the messages shall not be recorded into a file the button *No file output* must be chosen.

If the recording of the data in a file is selected with *Log to file*, the file name can be entered directly into the input box or selected with the button next to it.
The selected file has always the format CANreal logfiles (binary data records) with the extension: `*.csplog`.

Append does not overwrite the existing file when the recording is started again but the recording in the file is continued, i.e. a new received CAN message is appended at the end of the file.


Trigger and Logging

Start Trigger | End Trigger | Output

[✓ No file output]
Log to file: [ ] ...
[ ] Append (not with "prepend")
[ ] Prepend date and time: (JJJJMMDD-HHMMSS-filename)
[ ] On-the-fly ASCII convert (*.txt)

Split file (size limit):
[ ] Number of frames: [ ]
[ ] Duration(h:mm:ss): [ ]
[ ] Size in KB: [ ]
[ ] Circular overwrite Max. No. of files: [ ]

Start -> | Abort

Figure 14: Properties of message record file

The option *Append* can not be selected together with the options *Prepend date and time* and/or *Circular overwrite* and will be hidden if one of these selections is done.

Selecting *Prepend date and time* prepends the time to the file name in the format `JJJJMMDD-HHMMSS` (4 digits for the year, 2 digits each for month, day, hours, minutes and seconds).

Selecting *On-the-fly-ASCII convert (*.txt)* converts the received data directly into text and records it to a text file parallel to binary recording. This process needs a higher computing power.

NOTICE
If the computing capacity (processor clock rate, main memory) is insufficient, messages might get lost and the conversion will be delayed.
Solution: It is safer to record the binary messages at the On-the-fly-ASCII conversion and to convert the binary file afterwards with *Convert logfiles to text* into a readable text file (see chapter “3.1.7 Convert Logfiles to Text ”).

In the *Split file*-field the size of the generated trigger- output-files can be limited. Then a new file (file name see below) is automatically generated.
With *Number of Frames* only a defined number of messages is written in the output file.

Furthermore a maximum *Size in KB* can be defined. The duration of the file can be specified in hours, minutes and seconds. Duration does not affect the pretrigger files (*-pretrig.csplog). The pretrigger files can only be limited with the *Number of frames* or the *Size in KB*.

INFORMATION
If Duration is selected, and even if the time has expired, no new file will be started until
a CAN-fame is received.

Because of the file size limitation with *Split file* the data are not recorded in a single file but in a corresponding number of size limited files. The file names are composed of a freely selectable name which will be consecutively numbered:

```
filename_00000.csplog,
filename_00001.csplog,
...
filename_nnnnn.csplog.
```

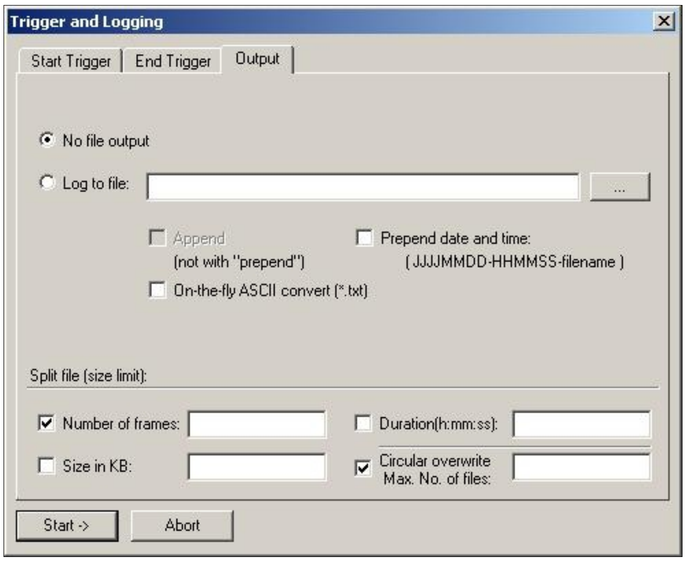
The data in the logfile header are initialized only once. Except for the file number they are static also for *Split-files*.

2.7.3.1 Description of Number of Data

If at least one option to limit the file size is selected in the field *Split file (size limit)*, the option *Circular overwrite Max. No. of files* can be selected.

With *Circular overwrite Max. No. of files* a maximum number of files can be specified.
When the last of the files is written, the recording is continued with overwriting the first and then the following files as e.g. in a ring buffer.

If *Circular overwrite Max. No. of files* is active, *Append* can not be selected.
If the max. number is set to high, CANreal limits this value automatically to the highest possible value.
After a new start incoming CAN telegrams can no longer be written at the end of the file.


Trigger and Logging

Start Trigger | End Trigger | Output

[✓ No file output]
Log to file:
Append (not with "prepend")
Prepend date and time: (JJJJMMDD-HHMMSS-filename)
On-the-fly ASCII convert (*.txt)

Split file (size limit):
[✓ Number of frames:]
Duration(h:mm:ss):
Size in KB:
[✓ Circular overwrite Max. No. of files:]

Start -> | Abort

Figure 15: Number of frames and circular overwrite

Circular overwrite does not affect pretrigger files `(*-pretrig.csplog)`. Always the entire file will be stored.

Description of the buttons

Start->

To enable the defined trigger conditions click the *Start* button.

Abort

To quit the dialogue box without starting the recording click *Abort*.
The current dialogue-settings will be kept unchanged.

3. Menu Bar

The main menu in the menu bar contains the five menu items *File, CAN, Send, View* and *Help*.

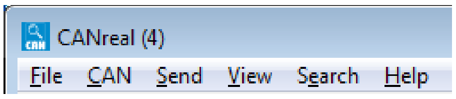


Figure 16: Menu bar of main menu

3.1 Menu Item *File*

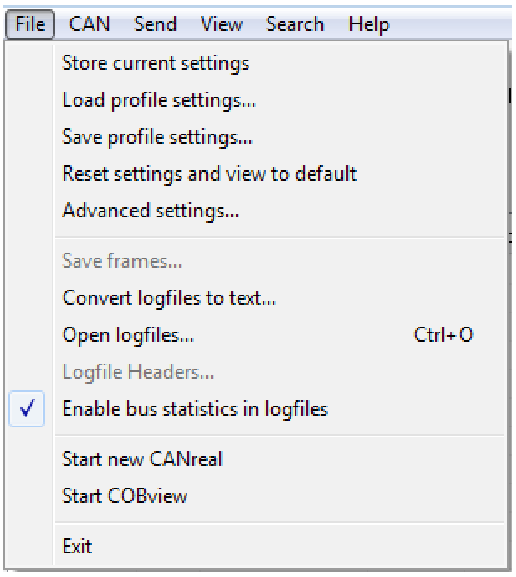


Figure 17: Menu item *File*

3.1.1 Store Current Settings

By clicking this *menu item* the current settings are stored as default settings. When CANreal is called again, all settings correspond to those which were specified at the moment *Store current settings* was called. To reset the settings click menu item *Restore settings and view to default*.

3.1.2 Load or Save CANreal Profile

With *Store profile settings* the CANreal parameters specified can be stored in profile files and can be loaded with *Load profile settings* again at a later time.

The profile files have the extension `*.cspini`.

Select *Load* or *Save* in the *Action* field to load or save the profile settings.
With the check box *start* next to *Load and...*you can select an immediate start of CANreal after loading.

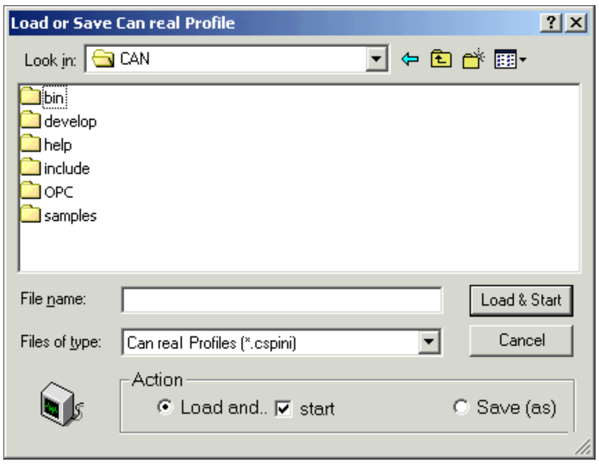


Figure 18: Load or Save CANreal profile

3.1.3 Reset Settings and View to Default

This menu item resets the settings and the view of the display window for received messages to default.

3.1.4 Advanced settings

3.1.4.1 Application

In this menu item in the field *Text description mapping for CAN identifiers* a file, which contains a description of the identifier, can be assigned to every identifier.

The file name has the extension `*.txt` and can be written in the input field, selected with the button next to the input field or generated with *Generate pattern file*. Clicking this button opens the dialogue window *Generate ID description pattern file*. Specify a new file name there.
With *Memory allocation for scroll back and trigger* the *Maximum number of CAN-frames* (1.000 - 2.000.000 messages) in the receive ring buffer of the display window can be specified.

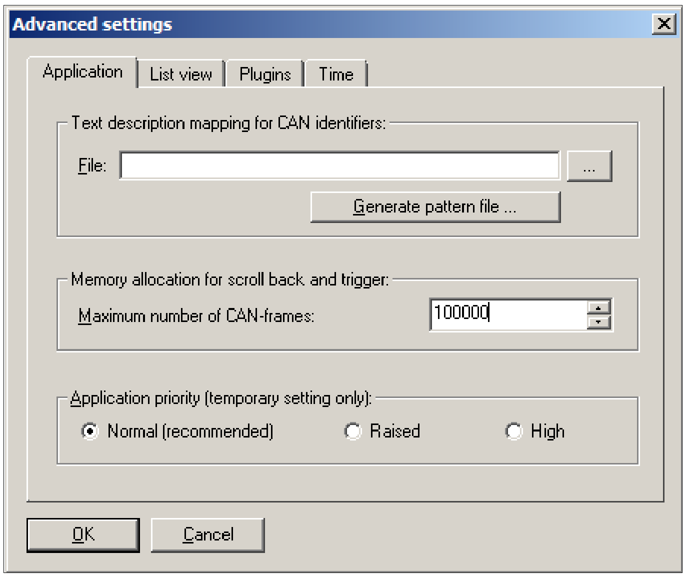

Advanced settings
Application | List view | Plugins | Time
Text description mapping for CAN identifiers:
File:
Generate pattern file ...
Memory allocation for scroll back and trigger:
Maximum number of CAN-frames: 100000
Application priority (temporary setting only):
(● Normal (recommended)) ○ Raised ○ High
OK Cancel

Figure 19: *Advanced settings - Application*

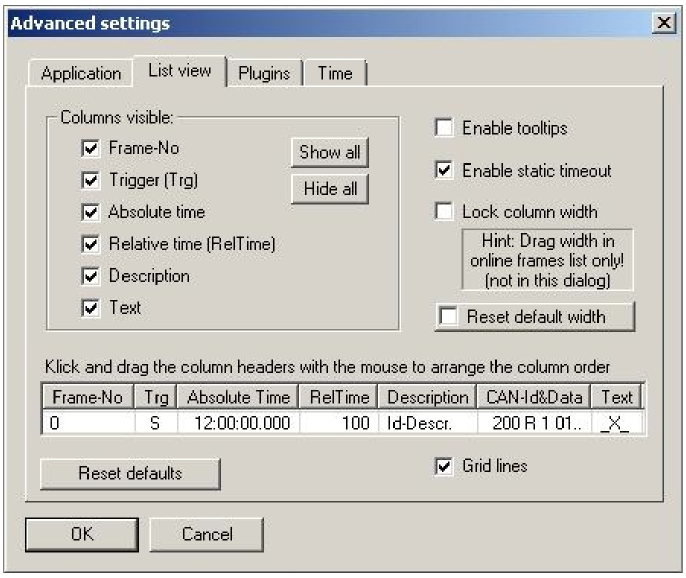
If the maximum number is reached, i.e. if the list is full, the first entry of the list will be deleted and the following entries move up.
Now a further entry can be written at the end of the list.

3.1.4.2 List View

In this menu item the display of the list can be changed.
In the box *Columns visible* single columns of the list can be selected or deselected.
The button *Show all* selects and shows all columns listed.
The button *Hide all* deselects and hides the columns listed.
CAN-Id&Data are always displayed.

The button *Enable tooltips* enables the tool tips in the display window for received messages.
The button *Enable static timeout* is associated with the static view and the plugins (see chapter “3.2.7 Static View”).

The columns can be moved in the table with the mouse. They can be arranged in the table as required.


Advanced settings
Application | List view | Plugins | Time
Columns visible:
☑ Frame-No
☑ Trigger (Trg)
☑ Absolute time
☑ Relative time (RelTime)
☑ Description
☑ Text
Show all
Hide all
☐ Enable tooltips
☑ Enable static timeout
☐ Lock column width
Hint: Drag width in online frames list only! (not in this dialog)
☐ Reset default width
Klick and drag the column headers with the mouse to arrange the column order

Frame-No	Trg	Absolute Time	RelTime	Description	CAN-Id&Data	Text
0	S	12:00:00.000	100	Id-Descr.	200 R 1 01..	_X_

Reset defaults
☑ Grid lines
OK Cancel

Figure 20: *Advanced settings – List view*

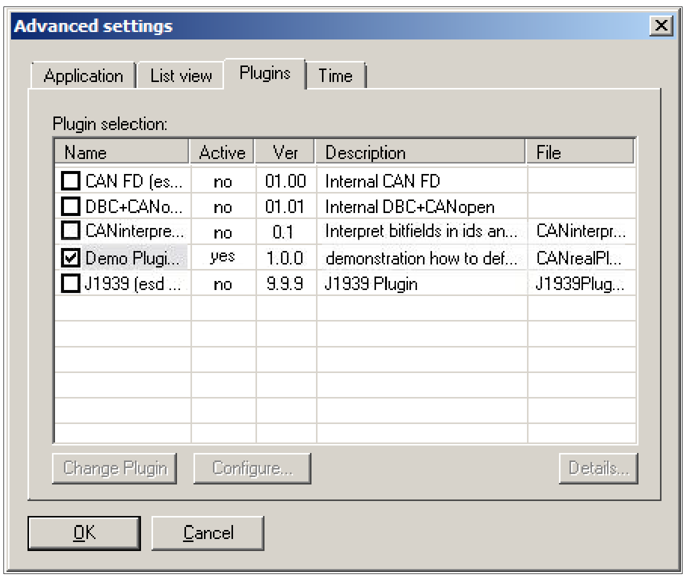
Enable the checkbox *Lock column width* or *Reset default width to* disable the arrangement of the column width or to reset the width.

3.1.4.3 Plugins

By means of plugins the display of the messages can be extended by additional columns and information, e.g. esd internal plugins for CAN-FD and CAN DBC+ CANopen or external plugins for special CAN-protocols (J1939).

In the display window all plugins contained in the plugin directory of CANreal are listed.
The plugins can be selected in the list.
Click the button *Change Plugin* to change. Always only one plugin can be enabled.
With *Configure...* the plugins can be configured.
Click the button *Details..* for additional information about the selected plugin.

For the conversion of logfiles, the Plugins are included. The additional columns are taken.



Plugin selection:

Name	Active	Ver	Description	File
☐ CAN FD (es...	no	01.00	Internal CAN FD	
☐ DBC+CANo...	no	01.01	Internal DBC+CANopen	
☐ CANinterpre...	no	0.1	Interpret bitfields in ids an...	CANinterpr...
☑ Demo Plugi...	yes	1.0.0	demonstration how to def...	CANrealPl...
☐ J1939 (esd ...	no	9.9.9	J1939 Plugin	J1939Plug...

Figure 21: *Advanced settings – Plugins*

Description of the columns

Name:	Name of the plugin
Active:	Yes or No
Ver:	Versions of the plugin
Description:	Description of the plugin
File:	File name of the plugins (The plugins are stored in the directory *canrealplugins,* which is a subdirectory of the CANreal program.)

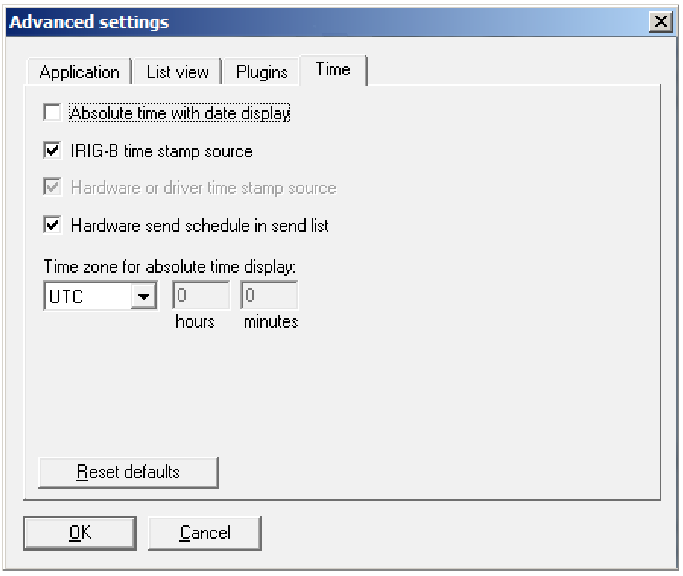
Plugins are also shown in the viewer for logfiles.
It is irrelevant whether the plugin has been active during the recording of the logfile.
When opening the file the CAN Data are evaluated and interpreted again by the plugin. One logfile can be examined with different plugins - also subsequently - for different message contents.

3.1.4.4 Time

In this menu item the time source (Windows time, hardware timestamp, IRIG-B timstamp) can be changed.
Furthermore the time display of the received messages in the context menu of the display window of the received messages can be changed.

If you activate the checkbox *Absolute time with date display*, the date will be additionally shown under *Absolute time.*

With checkbox *IRIG-B time stamp source* the external IRIG-B time stamping is enabled otherwise the Windows time is used.


Advanced settings

Application | List view | Plugins | Time

☐ Absolute time with date display
☑ IRIG-B time stamp source
☑ Hardware or driver time stamp source
☑ Hardware send schedule in send list

Time zone for absolute time display: UTC, hours: 0, minutes: 0

Figure 22: *Advanced settings – Time*

INFORMATION
Please note that the IRIG-B functionality can only be used if the hardware is equipped with an IRIG-B interface!
The following esd products support the IRIG-B functionality:
- CPCI-CAN/400-4-I-P 4xCAN,IRIG-B, PXI (esd order No.: C.2033.01)
- CAN-USB/400-IRIG-B (esd order No.: C.2069.04)
- PMC-CAN/400-4 4x CAN 1x IRIG-B (esd order No.: C.2047.01)

Also *Hardware or driver time stamp source* can be enabled. For further information see NTCAN Part 1: Application Developers Manual [1], chapter: “Timestamp”.

If the HW timestamp is enabled, the times are specified with microsecond accuracy. (*Absolute Time*: HH:MM:SS.ms.µs, *RelTime*: ms.µs).

If the hardware timestamp is not activated, the times are specified under Absolute Time and RelTime with millisecond accuracy (depending on the operating system, actual accuracy ≥ 10 ms).

Per default the *Hardware send schedule in send list* is enabled, which activates the high-precision (up to µs precise) processing of the send list. Please note that this feature can only be used with esdACC modules.

Open the drop-down menu *Time zone for absolute time display:* to choose the time zone. You can choose *Local Time* or *UTC*. You can also enter a time offset in *hours* and *minutes* if you choose *UTC++* for addition or *UTC--* for subtraction of the entered time value.

3.1.5 Save Frames

Save Frames stores all messages shown in the display window in a binary logfile (`*.csplog`).
The file type logfiles (`*.csplog`) has to be selected for saving and the text file will be generated automatically.

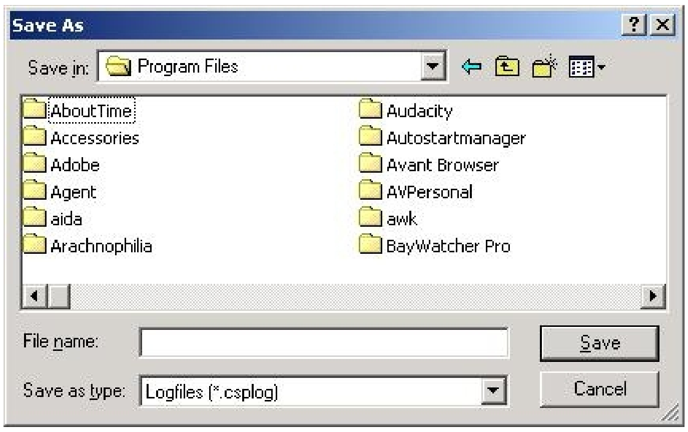


Figure 23: *Save As*

3.1.6 Enable Bus statistics in logfiles

If you enable this menu item, every second a bus statistic is inserted in the logfile.

3.1.7 Convert Logfiles to Text

With *Convert logfiles to text* an already existing logfile (`*.csplog`), in which the received data are stored binary coded, can be selected.
The selected logfile can be converted to a readable text file (`*.txt`).

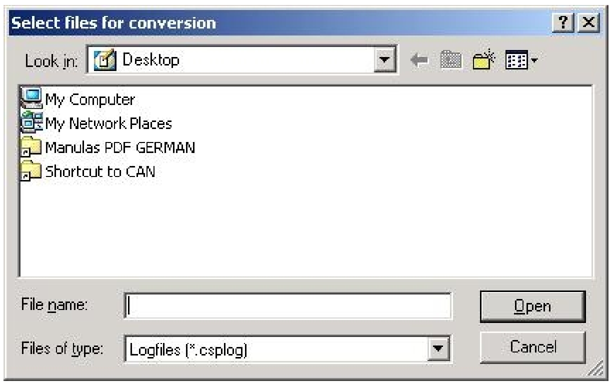
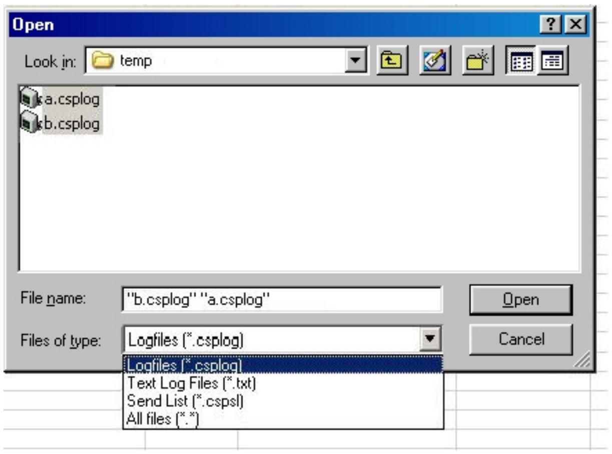


Figure 24: *Select files for conversion*

3.1.8 Open Logfiles

Via the option *Open logfiles* the file types (binary) logfile, text log file and *Send list* can be selected.

If multiple files have been selected, the files are displayed in chronological order in the *Send list*.



Open

Look in: temp

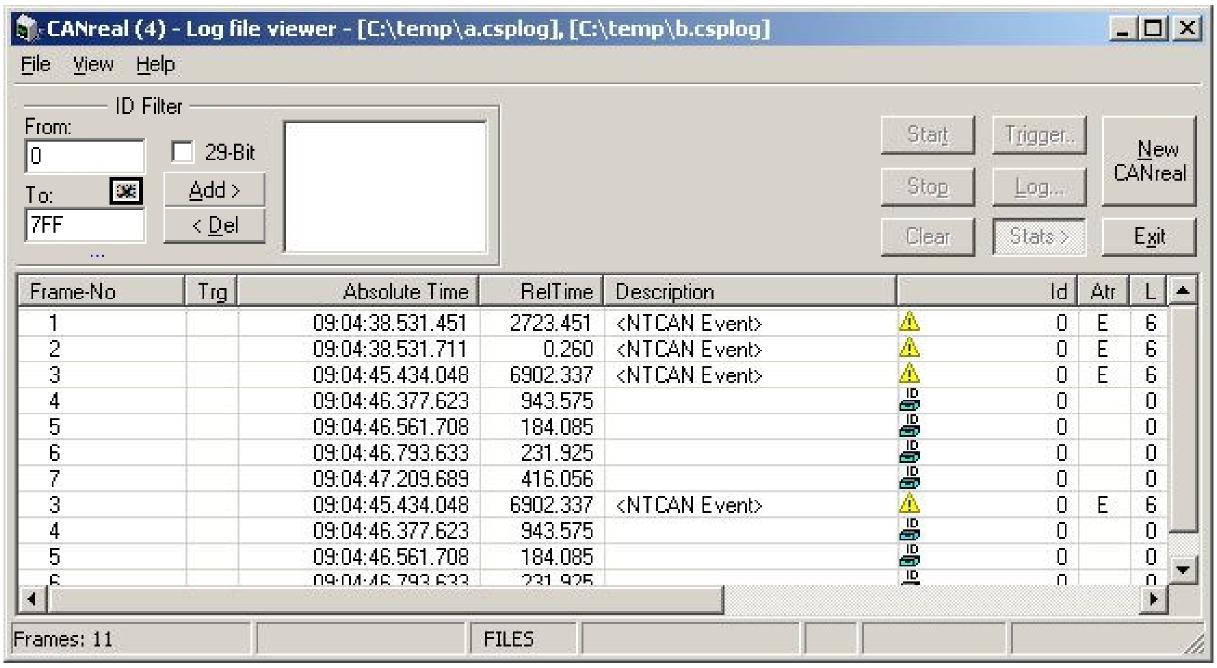
a.csplog
b.csplog

File name: "b.csplog" "a.csplog"

Files of type: Logfiles (*.csplog)

Logfiles (*.csplog)
Text Log Files (*.txt)
Send List (*.cspsl)
All files (*.*)

Figure 25: Open logfiles...



CANreal (4) - Log file viewer - [C:\temp\a.csplog], [C:\temp\b.csplog]

From: 0

To: 7FF

Frame-No	Trg	Absolute Time	RelTime	Description		Id	Atr	L
1		09:04:38.531.451	2723.451	<NTCAN Event>	⚠	0	E	6
2		09:04:38.531.711	0.260	<NTCAN Event>	⚠	0	E	6
3		09:04:45.434.048	6902.337	<NTCAN Event>	⚠	0	E	6
4		09:04:46.377.623	943.575		ID	0		0
5		09:04:46.561.708	184.085		ID	0		0
6		09:04:46.793.633	231.925		ID	0		0
7		09:04:47.209.689	416.056		ID	0		0
3		09:04:45.434.048	6902.337	<NTCAN Event>	⚠	0	E	6
4		09:04:46.377.623	943.575		ID	0		0
5		09:04:46.561.708	184.085		ID	0		0
6		09:04:46.793.633	231.925		ID	0		0

Frames: 11 FILES

Figure 26: Open logfiles viewer

3.1.9 Logfile Headers...

Click this menu item to see the headers of a recorded logfile with addition information.

3.1.10 Start new CANreal

Start new CANreal starts a new instance of the program and a new CANreal window will be opened. This can be used to monitor various properties as for example different ID ranges, CAN nets or different views as Static View or Scroll-down list.

The number of CANreal instances that can be opened depends on the CAN drivers used.

3.1.11 Start COBview

The program COBview (CANopen Object Viewer) can be called via the menu item *Start COBview*. COBview is an effective CANopen tool for the analysis/diagnostics of CANopen nodes. For further information please refer to the COBview manual. A current version of the manual can be downloaded from our esd website:
http://esd.eu/en/products/can-tools

3.1.12 Exit

Click menu item *Exit* to exit the instance of the CANreal. The program window will be closed.

3.2 Menu item *CAN*

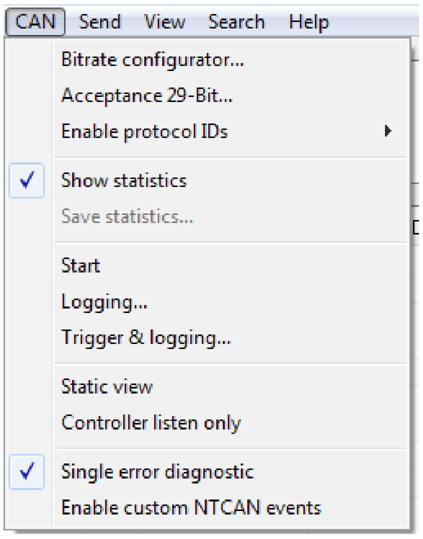


Figure 27: Menu item CAN

INFORMATION
The commands *Start, Logging* and *Trigger & Logging* can also be activated directly via the corresponding buttons in the program window (see page 18 and 19)

3.2.1 Bitrate Configurator

By means of the *Bitrate configurator* you can define your own application-specific bit rates and bit timings especially for CAN FD. The *Bitrate configurator* can be opened via the menu item *Bitrate configurator...* in the menu *CAN*, see Figure above.
It is also possible to open the *Bitrate configurator* via the drop-down list for bit rates. Click on the selection-box *Baud* and choose the last row (*Configurator...*).

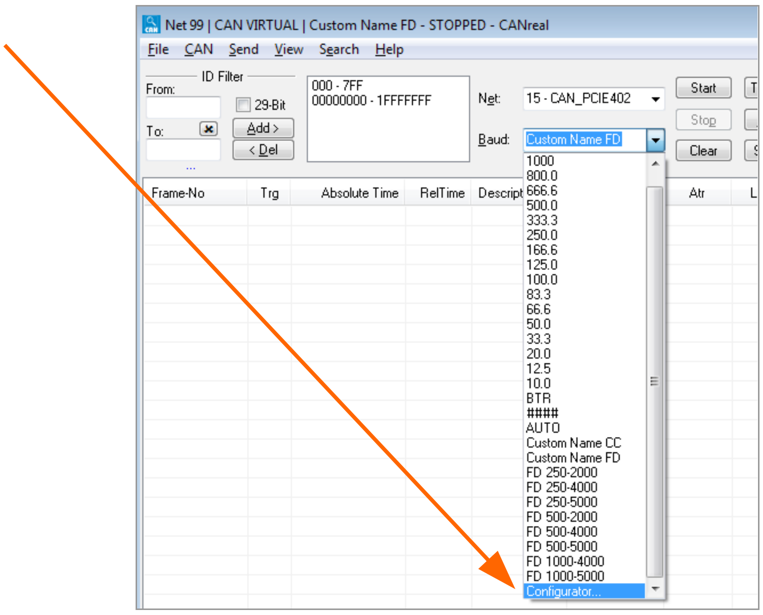
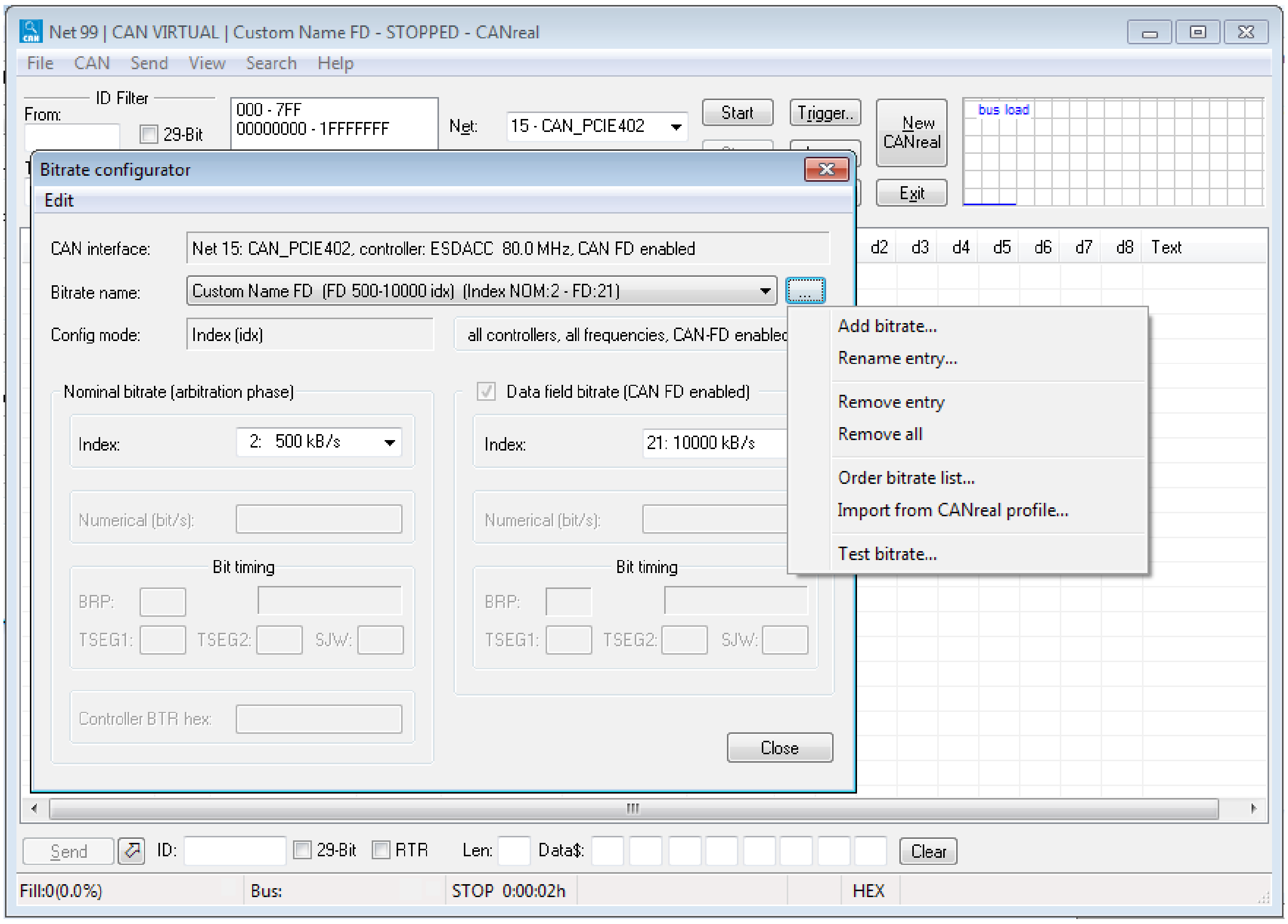


Figure 28: Open Configurator

The dialogue window of the *Bitrate configurator* opens.


Net 99 | CAN VIRTUAL | Custom Name FD - STOPPED - CANreal

Bitrate configurator

CAN interface: Net 15: CAN_PCIE402, controller: ESDACC 80.0 MHz, CAN FD enabled

Bitrate name: Custom Name FD (FD 500-10000 idx) (Index NOM:2 - FD:21)

Config mode: Index (idx) all controllers, all frequencies, CAN-FD enabled

Nominal bitrate (arbitration phase) Index: 2: 500 kB/s

Data field bitrate (CAN FD enabled) Index: 21: 10000 kB/s

Figure 29: *Bitrate configurator* with opened menu

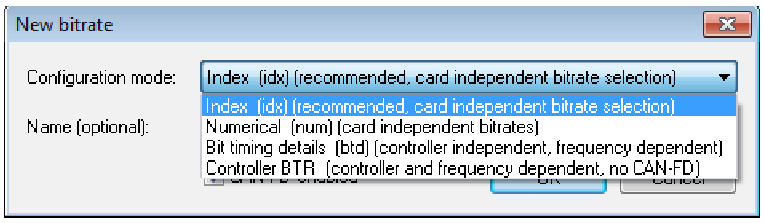
Under *Bitrate name* you can select a bit rate entry with configuration mode, or you can add a new entry, as described in the following. The configuration modes are described from page 34.

3.2.1.1 Add a New Bit Rate Entry

To define a new bit rate entry start by clicking *Add bitrate....* Then select a configuration mode and enter an optional name, which is displayed in the list of the bit rates instead of the standard name.

The menu item *Add bitrate...* and the other menu items to edit the list of the bit rates are in the main menu *Edit* of the dialogue window.

The same menu opens as shown in Figure 29 as drop-down-menu if you click on this button.


New bitrate

Configuration mode: Index (idx) (recommended, card independent bitrate selection)

Name (optional):
Index (idx) (recommended, card independent bitrate selection)
Numerical (num) (card independent bitrates)
Bit timing details (btd) (controller independent, frequency dependent)
Controller BTR (controller and frequency dependent, no CAN-FD)

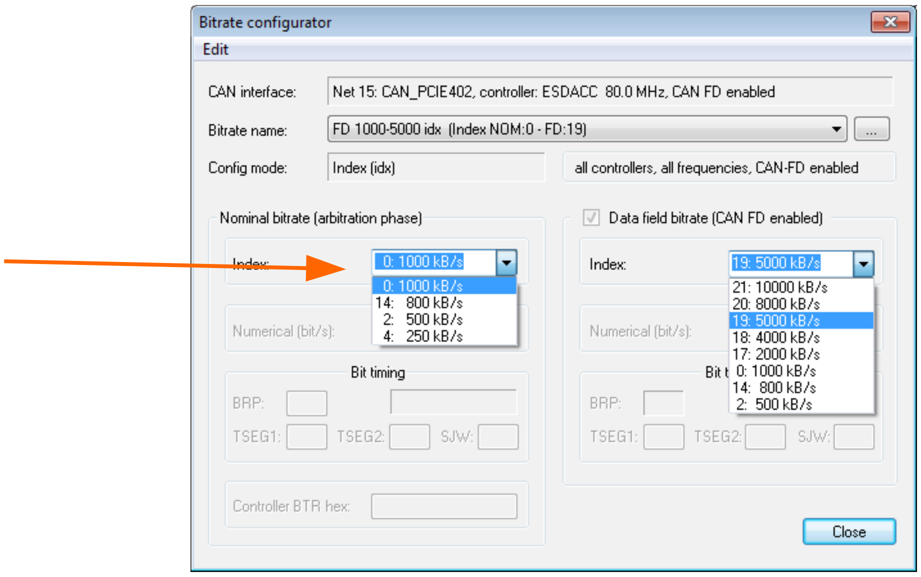
Figure 30: New bit rate

After selecting the menu item *Add bitrate...*, the dialogue window *New bitrate* is displayed.
Select the configuration mode in the drop-down menu of *Configuration mode.*
Additionally, you can enter a new name in the input field *Name (optional).*

Depending on the driver support, the bit rates can be defined via the four configuration modes: *Index, Numerical, Bit timing* and *Controller BTR*.

3.2.1.2 Configuration Modes

Index The configuration mode *Index* offers a pre-defined selection of bit rates and is independent from board, controller and controller frequency.
Available for all boards (with and without CAN FD).
Depending on the controller frequency some bit rates might be inaccurate.
In this case a warning will be shown at the beginning of the recording.


Bitrate configurator
Edit

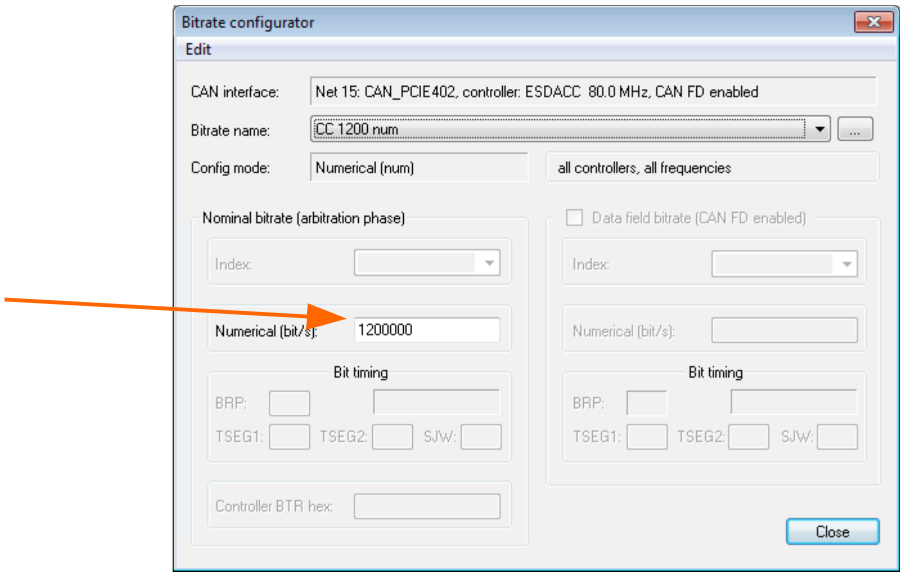
CAN interface: Net 15: CAN_PCIE402, controller: ESDACC 80.0 MHz, CAN FD enabled
Bitrate name: FD 1000-5000 idx (Index NOM:0 - FD:19)
Config mode: Index (idx) — all controllers, all frequencies, CAN-FD enabled

Nominal bitrate (arbitration phase)
Index: 0: 1000 kB/s
0: 1000 kB/s
14: 800 kB/s
2: 500 kB/s
4: 250 kB/s

Data field bitrate (CAN FD enabled)
Index: 19: 5000 kB/s
21: 10000 kB/s
20: 8000 kB/s
19: 5000 kB/s
18: 4000 kB/s
17: 2000 kB/s
0: 1000 kB/s
14: 800 kB/s
2: 500 kB/s

Figure 31: Pre-defined selection of bit rates

Numerical **(Not available for CAN FD)**
Allows a free numerical entry of the bit rate value in the input box *Numerical* (in bits/s). This configuration mode is independent from board, controller and controller frequency, but not available for CAN FD.
Depending on the controller frequency some bit rates might be not available or inaccurate. In this case a warning will be shown at the beginning of the recording.


Bitrate configurator
Edit

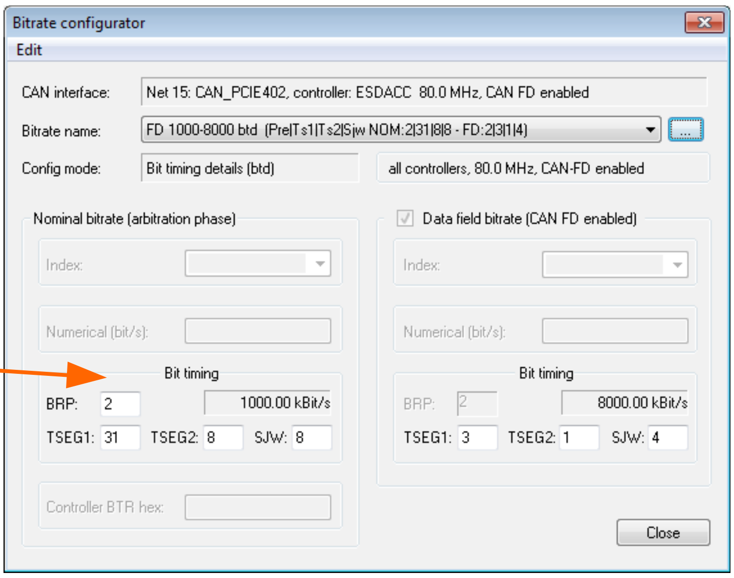
CAN interface: Net 15: CAN_PCIE402, controller: ESDACC 80.0 MHz, CAN FD enabled
Bitrate name: CC 1200 num
Config mode: Numerical (num) — all controllers, all frequencies

Nominal bitrate (arbitration phase)
Numerical (bit/s): 1200000

Figure 32: Numerical entry of the bit rate

Bit timing details

This configuration mode offers the most detailed possibility of parametrisation. The single bit timing parameters BRP, TSEG and SJW can be entered individually. This configuration mode can be used for CAN FD.



CAN interface: Net 15: CAN_PCIE402, controller: ESDACC 80.0 MHz, CAN FD enabled
Bitrate name: FD 1000-8000 btd (Pre|Ts1|Ts2|Sjw NOM:2|31|8|8 - FD:2|3|1|4)
Config mode: Bit timing details (btd) — all controllers, 80.0 MHz, CAN-FD enabled

Nominal bitrate (arbitration phase)
Index:
Numerical (bit/s):
Bit timing
BRP: 2 — 1000.00 kBit/s
TSEG1: 31 TSEG2: 8 SJW: 8
Controller BTR hex:

Data field bitrate (CAN FD enabled) ✓
Index:
Numerical (bit/s):
Bit timing
BRP: 2 — 8000.00 kBit/s
TSEG1: 3 TSEG2: 1 SJW: 4

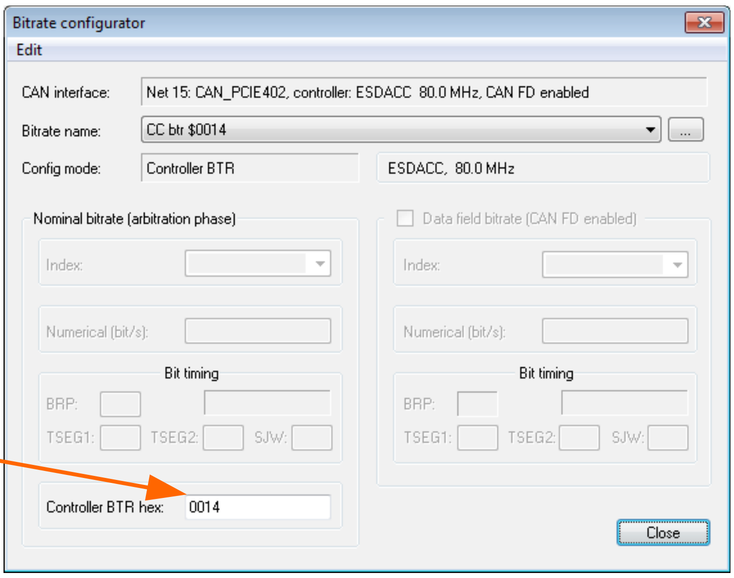
Figure 33: Bit Timing

INFORMATION
At esdACC boards the BRP (bit timing prescaler) is parametrised for both bit rates (*nominal* and *data field*) with the same value!

The parametrised bit timing can also be selected for other esdACC-boards if the CAN controllers have got the identical frequency.

Controller BTR (Not available for CAN FD)

The bit timing, parametrised via a controller-register value, is selectable for boards with identical CAN controllers and identical frequency (e.g. only SJA1000 with 16 MHz).



CAN interface: Net 15: CAN_PCIE402, controller: ESDACC 80.0 MHz, CAN FD enabled
Bitrate name: CC btr $0014
Config mode: Controller BTR — ESDACC, 80.0 MHz

Nominal bitrate (arbitration phase)
Index:
Numerical (bit/s):
Bit timing
BRP:
TSEG1: TSEG2: SJW:
Controller BTR hex: 0014

Data field bitrate (CAN FD enabled)
Index:
Numerical (bit/s):
Bit timing
BRP:
TSEG1: TSEG2: SJW:

Figure 34: Controller BTR
Not available for CAN FD -> Always use "Bit timing details" for CAN FD boards.

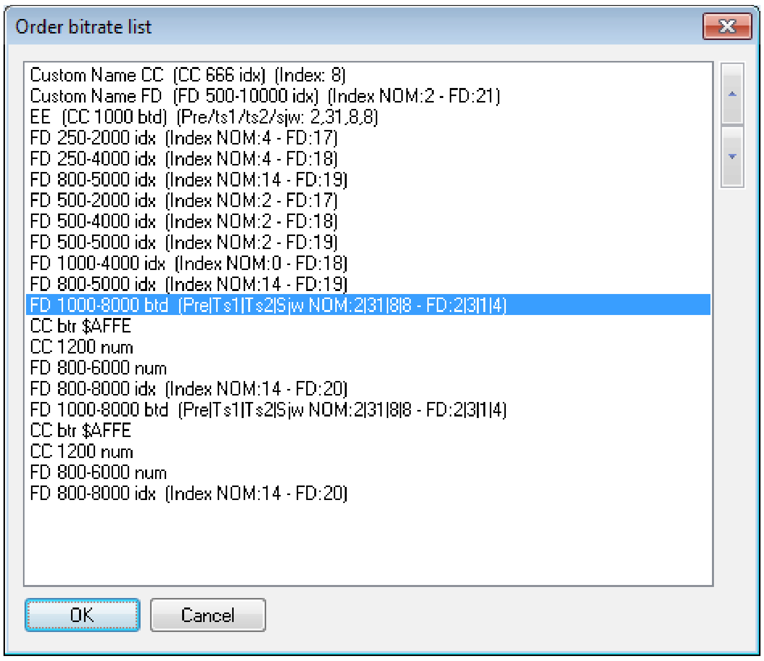
3.2.1.3 Rename / Remove Bit Rate Entries

The configuration mode can not be changed subsequently. But the name, the order of the bit rates and the parameters can be subsequently edited via the menu items under *Edit*.

Menu item	Description
Rename entry...	Rename the selected bit rate entry
Remove entry	Remove the selected bit rate entry
Remove all...	Remove all bit rate entries

3.2.1.4 Arrange the Bit Rate Entries

To arrange the bit rate entries click on the menu item *Order bitrate list...* in the *Edit* menu



```
Custom Name CC  (CC 666 idx)  (Index: 8)
Custom Name FD  (FD 500-10000 idx)  (Index NOM:2 - FD:21)
EE  (CC 1000 btd)  (Pre/ts1/ts2/sjw: 2,31,8,8)
FD 250-2000 idx  (Index NOM:4 - FD:17)
FD 250-4000 idx  (Index NOM:4 - FD:18)
FD 800-5000 idx  (Index NOM:14 - FD:19)
FD 500-2000 idx  (Index NOM:2 - FD:17)
FD 500-4000 idx  (Index NOM:2 - FD:18)
FD 500-5000 idx  (Index NOM:2 - FD:19)
FD 1000-4000 idx  (Index NOM:0 - FD:18)
FD 800-5000 idx  (Index NOM:14 - FD:19)
FD 1000-8000 btd  (Pre|Ts1|Ts2|Sjw NOM:2|31|8|8 - FD:2|3|1|4)
CC btr $AFFE
CC 1200 num
FD 800-6000 num
FD 800-8000 idx  (Index NOM:14 - FD:20)
FD 1000-8000 btd  (Pre|Ts1|Ts2|Sjw NOM:2|31|8|8 - FD:2|3|1|4)
CC btr $AFFE
CC 1200 num
FD 800-6000 num
FD 800-8000 idx  (Index NOM:14 - FD:20)
```

In the window *Order bitrate list* the bit rates are displayed in the order as they are listed in the drop-down list of *Baud*.

With the arrow keys on the right of the window you can move the selected entry up ▲ or down ▼ in the list.

Figure 35: Arrange the bit rate entries

3.2.1.5 Import the Bit Rate Entries

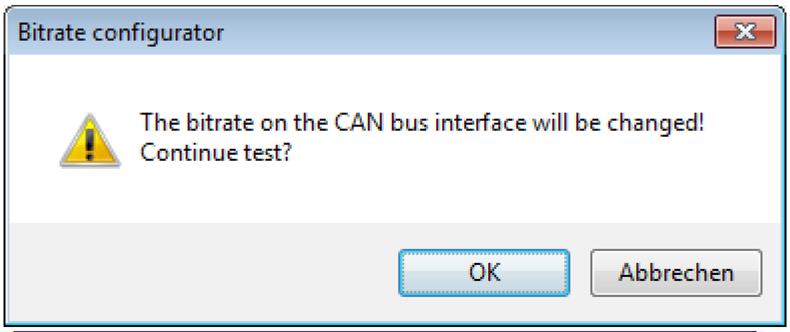
The bit rate list can be loaded from a previously stored CANreal profile. To import bit rate entries click on the menu item *Import from CANreal profile....* in the *Edit* menu.

3.2.1.6 Test the Bit Rate Entries

With the menu item *Test bitrate...* it can be tested in advance if the current CAN driver of the board supports the configuration mode and the parameters.

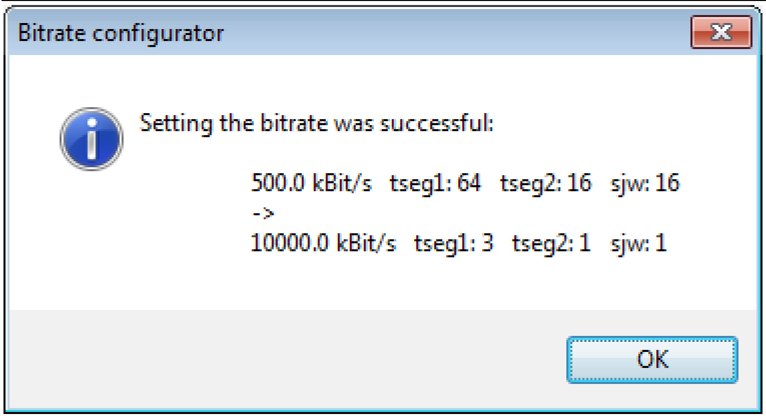
If you click on the menu item *Test bitrate...*, you will get the warning as shown on the right, that the CAN bit rate will be changed.

Only confirm this note with *OK* if you are sure that you really want to change the bit rate.



```
The bitrate on the CAN bus interface will be changed!
Continue test?
```

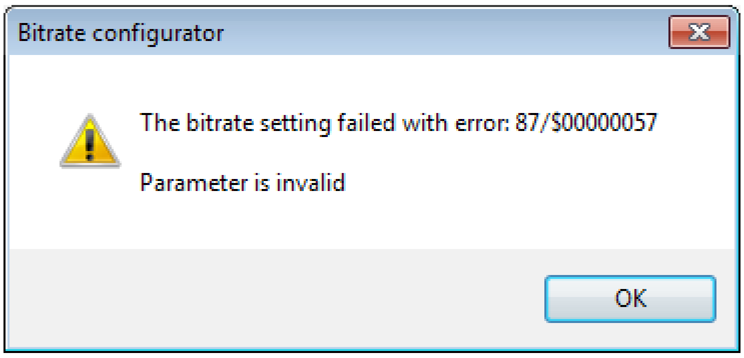
If the test has been successful, you receive the message on the right with indication of the bit timing (two bit rates at CAN FD, e.g.: 500 kBit/s, 10 000kBit/s)



```
Setting the bitrate was successful:

            500.0 kBit/s   tseg1: 64   tseg2: 16   sjw: 16
            ->
            10000.0 kBit/s   tseg1: 3   tseg2: 1   sjw: 1
```

If the test has **not** been successful, you receive the error message on the right with indication of the error number.



```
The bitrate setting failed with error: 87/$00000057

Parameter is invalid
```

Figure 36: Messages

3.2.1.7 Store Bit Rate Entries

After closing the *Bitrate configurator* dialogue the defined bit rates can be selected via the CANreal selection box *Baud* for bit rates, provided that they are supported by the board of the selected net.

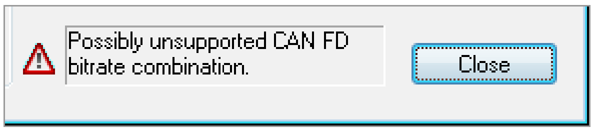
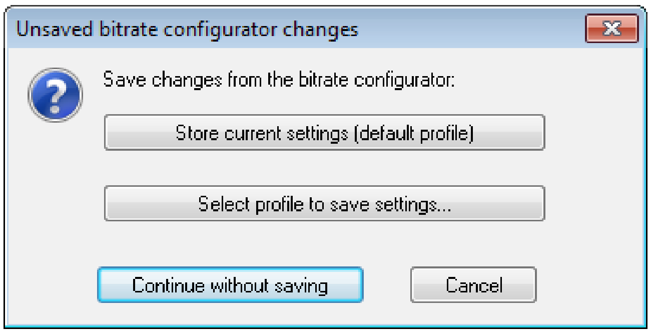

Possibly unsupported CAN FD bitrate combination.

Figure 37: Error message

In case of invalid settings you will get a warning. Check your settings before you close the Bitrate configurator.


Unsaved bitrate configurator changes

Save changes from the bitrate configurator:

Store current settings (default profile)
Select profile to save settings...
Continue without saving
Cancel

If you have changed settings and want to exit CANreal, you will get the request on the left. You are asked to store the current settings.

Select how to store your settings.

Figure 38: Request to store

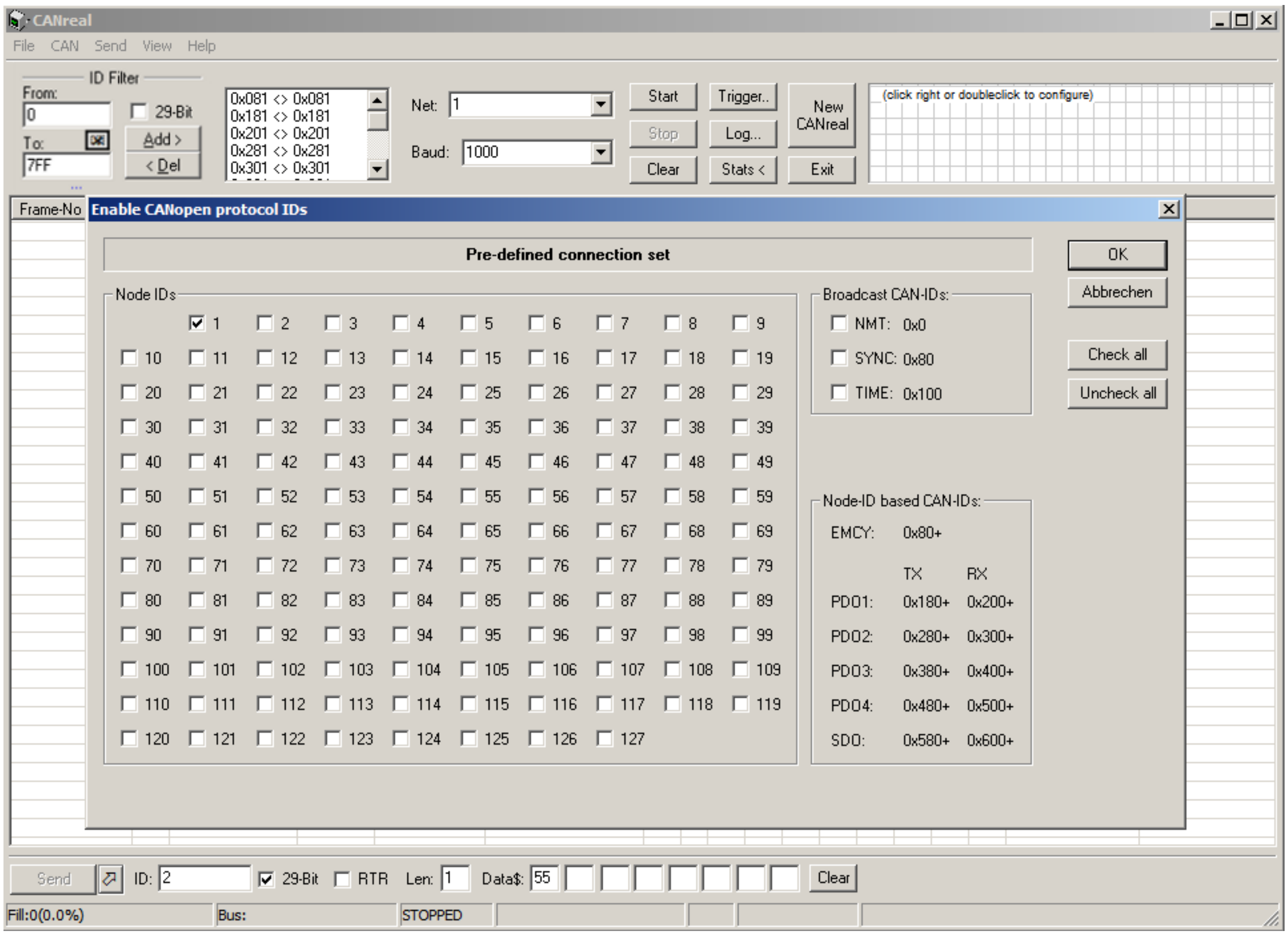
Store current settings (default profile)	If you click on this button, the changed bit rates are stored to the default profile. This command has the same function as the command *Store current settings*, see chapter “Store Current Settings” on page 27.
Select profile to save settings...	Click this button, to select the file, in which the changed CANreal settings and the bit rates are stored. This command has the same function as the command *Save profile settings*, see chapter “Load or Save CANreal Profile” on page 27. The bit rates are automatically stored in the CANreal profile and can be imported via the command *Import from CANreal profile* (see page 38) from another CANreal profile file.
Continue without saving	Exit the bit rate configurator without saving the changes.

3.2.2 Acceptance 29-Bit...

To enable the acceptance filtering for the extended frame format (29-bit CAN IDs) select this menu item. Please refer to chapter "Acceptance Filtering" of the NTCAN manual [1] for further information.

3.2.3 Enable protocol IDs

A click on this menu item opens the dialogue window *Enable CANopen protocol IDs*:.
In this dialogue window CAN IDs of CANopen nodes can be selected. The corresponding CAN-IDs of the selected CANopen nodes will be enabled and written into the field of the active IDs.


CANreal

File CAN Send View Help

ID Filter: From: 0 To: 7FF ☐ 29-Bit Add > < Del

0x081 <> 0x081
0x181 <> 0x181
0x201 <> 0x201
0x281 <> 0x281
0x301 <> 0x301

Net: 1 Baud: 1000

Start Trigger.. New CANreal Stop Log... Clear Stats < Exit

(click right or doubleclick to configure)

Frame-No

Enable CANopen protocol IDs

Pre-defined connection set

OK Abbrechen Check all Uncheck all

Node IDs: [✓ 1], ☐ 2 … ☐ 127 (all other node IDs unchecked)

Broadcast CAN-IDs:
☐ NMT: 0x0
☐ SYNC: 0x80
☐ TIME: 0x100

Node-ID based CAN-IDs:

	TX	RX
EMCY:	0x80+	
PDO1:	0x180+	0x200+
PDO2:	0x280+	0x300+
PDO3:	0x380+	0x400+
PDO4:	0x480+	0x500+
SDO:	0x580+	0x600+

Send ID: 2 [✓ 29-Bit] ☐ RTR Len: 1 Data$: 55 Clear

Fill:0(0.0%) Bus: STOPPED

Figure 39: Enable CANopen protocol IDs (Example)

If the checkbox of a CANopen node-ID is activated in the field *Node IDs*, it is added to the pre-defined base Node-IDs (see field *Node-ID based CAN-IDs*) for EMCY, Tx- and Rx-PDOs1-4 and Tx and Rx-SDO and the resulting CAN-IDs are automatically entered in the field of active IDs.

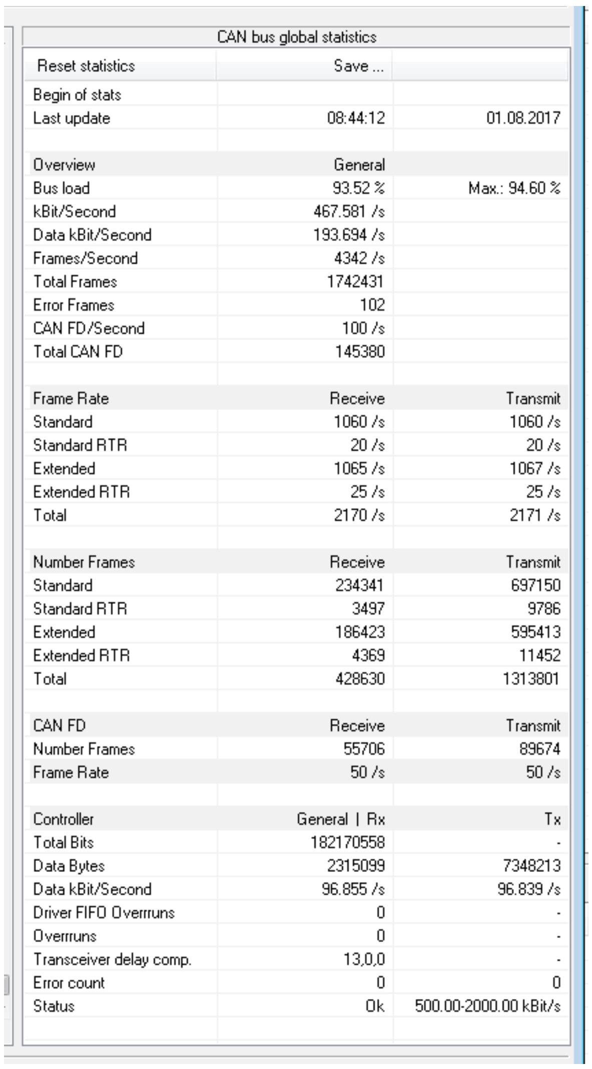
Example:

CANopen Node-ID =	1	
pre-defined base Node-IDs =	0x80+,	0x380+, 0x400+,
	0x180+, 0x200+,	0x480+, 0x500+,
	0x280+, 0x300+,	0x580+, 0x600+
=> resulting CAN-IDs =	0x81,	0x381, 0x401,
	0x181, 0x201,	0x481, 0x501,
	0x281, 0x301,	0x581, 0x601

3.2.4 Show Statistics

Choose this menu item to display the statistic window. This window is shown right of the display window for received messages. The global statistics of the CAN net is generated by the CAN driver. It is valid for all programs, that access the CAN net.

INFORMATION
The statistics is only available if the CAN hardware driver supports the statistics function. The entries for CAN FD are only displayed for detected CAN FD hardware.



CAN bus global statistics

Reset statistics	Save ...	
Begin of stats		
Last update	08:44:12	01.08.2017
Overview	General	
Bus load	93.52 %	Max.: 94.60 %
kBit/Second	467.581 /s	
Data kBit/Second	193.694 /s	
Frames/Second	4342 /s	
Total Frames	1742431	
Error Frames	102	
CAN FD/Second	100 /s	
Total CAN FD	145380	
Frame Rate	Receive	Transmit
Standard	1060 /s	1060 /s
Standard RTR	20 /s	20 /s
Extended	1065 /s	1067 /s
Extended RTR	25 /s	25 /s
Total	2170 /s	2171 /s
Number Frames	Receive	Transmit
Standard	234341	697150
Standard RTR	3497	9786
Extended	186423	595413
Extended RTR	4369	11452
Total	428630	1313801
CAN FD	Receive	Transmit
Number Frames	55706	89674
Frame Rate	50 /s	50 /s
Controller	General | Rx	Tx
Total Bits	182170558	-
Data Bytes	2315099	7348213
Data kBit/Second	96.855 /s	96.839 /s
Driver FIFO Overruns	0	-
Overruns	0	-
Transceiver delay comp.	13,0,0	-
Error count	0	0
Status	Ok	500.00-2000.00 kBit/s

Figure 40: The global CAN bus statistics

Description of the table elements

Reset statistics
Click this element to reset the statistics

Save...
Click *Save* to save the statistics in binary Log-file format. Before the statistics can be shown it must be converted to a text-file.
(see chapter “3.1.7 Convert Logfiles to Text ”)

Parameter	**Description**
Begin of stats	Time of "*Reset statistics*" [hh:mm:ss]
Last update	Date of last update
Overview	
Bus load	Percentaged bus load "Kbit/second" in relation to the bit rate of the CAN net.
kBit/Second	Total bit/s see below
Frames/Second	CAN messages per second. Specified in more detail with *Frame Rate.*
Total Frames	Absolute counter for all CAN messages. Specified in more detail with *Number Frames.*
Error Frames	Counter for all faulty CAN messages detected by the CAN controller.
Frame Rate	
Standard	Number of standard Rx- and Tx-frames per second
Standard RTR	Number of standard RTR Rx- and Tx-frames per second
Extended	Number of extended Rx- and Tx-frames per second
Extended RTR	Number of extended RTR Rx- and Tx-frames per second
Total	Total of Rx- and Tx-frames per second
Number Frames	
Standard	Number of standard Rx- and Tx-frames
Standard RTR	Number of standard RTR Rx- and Tx-frames
Extended	Number of extended Rx- and Tx-frames
Extended RTR	Number of extended RTR Rx- and Tx-frames
Total	Total of Rx- and Tx-frames
CAN FD (only with CAN FD hardware)	
Number Frames	Number of all CAN FD Rx- and Tx-frames
Frame Rate	Number of CAN FD Rx- and Tx-frames per second
Controller	
Total Bits	Counts all bits on the CAN bus (see NTCAN Part 1: Application Developers Manual [1], chapter: "*NTCAN-BUS-STATISTIC*")
Data Bytes	Number of transmitted or received data bytes
Driver FIFO Overruns	See NTCAN Part 1: Application Developers Manual [1], chapter: "EV_CAN_ERROR"
Error Count	Register of CAN controller
Status	Contains the bus state of the CAN controller (see NTCAN Part 1: Application Developers Manual [1], chapter: "*NTCAN_CTRL_STATE*").

3.2.5 Save Statistics

Via this menu item you can save the statistics. To save the data choose or enter a file in the dialogue box.

3.2.6 Start, Logging, Trigger&Logging

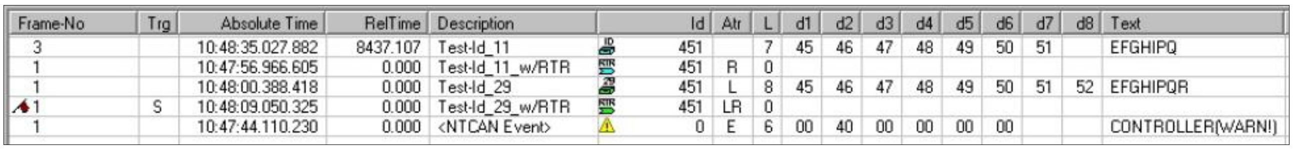
The *Start* menu item starts the listing of received messages in the display window. The menu item has the same function as the button *Start* (see page 18).

Clicking the menu item *Logging...* opens the dialogue box *Output*. The properties of the file, in which the messages are stored can be specified here (see page 25) .

Clicking this menu item opens the dialogue box *Trigger and Logging* and conditions can be specified, which start (*Start Trigger*) or end (*End Trigger*) the recording of the messages in files (see chapter “2.7 Trigger and Logging“).

3.2.7 Static View

With this menu item the static display mode can be activated. In this mode the frames are not listed in the display window according to the date of their reception, but for every received ID a new row is created in which all received frames with this ID are counted. RTR, event frames and 29-bit IDs are counted separately.
Switching into the static mode is only possible if the program run is stopped. It is not possible while the program is running. In static view for 29-bit CAN-IDs the number of rows in the list is limited to 100.000 CAN messages with and without RTR.



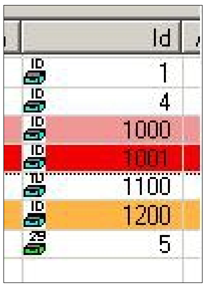
Frame-No	Trg	Absolute Time	RelTime	Description		Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
3		10:48:35.027.882	8437.107	Test-Id_11	ID	451		7	45	46	47	48	49	50	51		EFGHIPQ
1		10:47:56.966.605	0.000	Test-Id_11_w/RTR	RTR	451	R	0									
1		10:48:00.388.418	0.000	Test-Id_29	29	451	L	8	45	46	47	48	49	50	51	52	EFGHIPQR
1	S	10:48:09.050.325	0.000	Test-Id_29_w/RTR	RTR	451	LR	0									
1		10:47:44.110.230	0.000	<NTCAN Event>	⚠	0	E	6	00	40	00	00	00	00			CONTROLLER(WARN!)

Figure 41: Part of the display window in static mode

Frame-No	(in static mode) number of received messages with the identifier *Id* and the attributes listed in the column *Atr.*

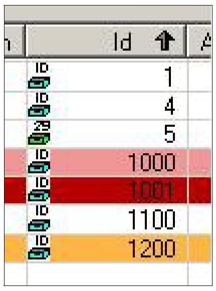
3.2.7.1 Sorting

The sorting of the *Static View* can be easily changed with a click in the particular header column.



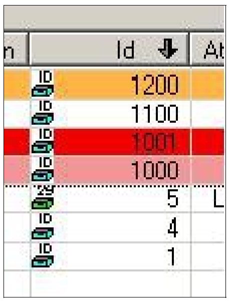
Id
1
4
1000
1001
1100
1200
5

This view shows the standard sorting of the CAN messages.



Id ⇧
1
4
5
1000
1001
1100
1200

Click on the header cell (e.g. Id) in the static view, to rearrange the CAN messages. The messages are now listed up in ascending order of the IDs.



Id ⇩	At
1200	
1100	
1001	
1000	
5	L
4	
1	

A second click on the header cell (e.g. Id), rearranges the CAN messages again. The messages are now listed up in descending order of the IDs.

After usage of the sorting function, messages with new CAN-ID are attached unsorted at the end of the list. An automatic sorting is only made in the standard sorting.
The header of the display window for received messages offers an additional context menu. In this menu the kind of data interpretation *Text*, *Number* or *Hex number* can be chosen..

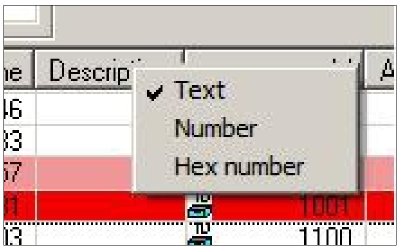
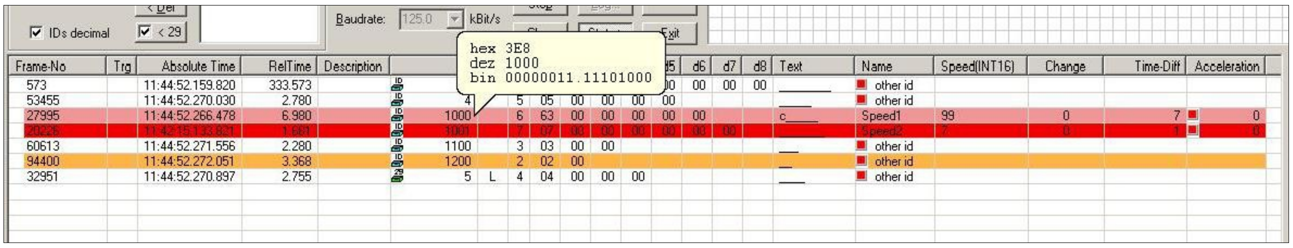


Figure 42: Context menu of the header

3.2.7.2 Static View with activated Plugin and Timeout Function



Frame-No	Trg	Absolute Time	RelTime	Description									d5	d6	d7	d8	Text	Name	Speed(INT16)	Change	Time-Diff	Acceleration
573		11:44:52.159.820	333.573										00	00	00	00	____	other id				
53455		11:44:52.270.030	2.780		4		5	05	00	00	00	00					____	other id				
27995		11:44:52.266.478	6.980		1000		6	63	00	00	00	00	00				c____	Speed1	99	0	7	0
20226		11:42:15.133.821	1.681		1001		7	07	00	00	00	00	00	00			____	Speed2	7	0	1	0
60613		11:44:52.271.556	2.280		1100		3	03	00	00							____	other id				
94400		11:44:52.272.051	3.368		1200		2	02	00								____	other id				
32951		11:44:52.270.897	2.755		5	L	4	04	00	00	00						____	other id				

Figure 43: Static View with Plugin Timeout

Background of line	Description
red	CAN messages which are not received in the predefined time are marked with red. The plugin must support static timeouts for this.
light red	CAN messages which were delayed at the beginning, but are received in the predefined time later are marked with light red.
orange	see chapter" 3.4.2 Mark frames sent"

3.2.7.3 Context Menu of Static View

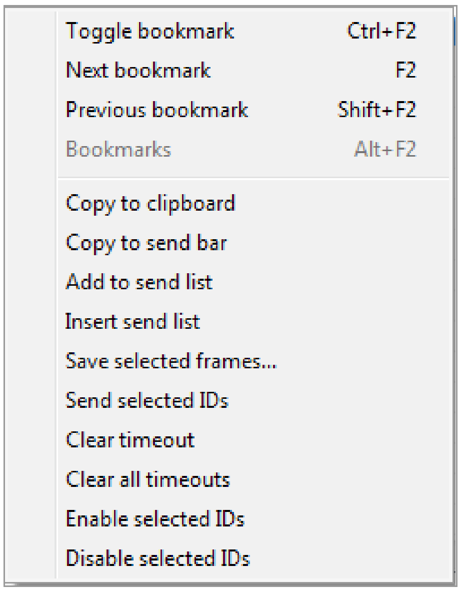


Figure 44: Context menu of the static view with enabled plugin

Commands of the context menu	Description	
Toggle bookmark	Sets/Deletes the bookmark in the selected row (row with blue background)	[Ctrl]+ [F2]
Next bookmark	Go to the next bookmark (see page 63)	[F2]
Previous bookmark	Go to the previous bookmark	[Shift] + [F2]
Bookmarks	Open the bookmark window, see chapter „Bookmark", page 63	[Alt] + [F2]

Commands of the context menu	**Description**
Copy to clipboard	Copies the selected messages to clipboard
Copy to send bar	Copies the selected CAN message into the send bar for messages that shall be transmitted.
Add to send list	Adds the CAN message to the end of the *Send list*.
Insert send list	CAN messages are inserted at a selected position of the *Send list*.
Save selected frames...	The selected CAN messages are saved in a file
Send	The selected CAN messages are transmitted again
Clear timeout	Deletes the light red marking (this menu item is only available with the particular plugin)
Clear all timeouts	Deletes all markings (this menu item is only available with the particular plugin)
Enable selected IDs	The selected IDs are enabled
Disable selected IDs	The selected IDs are disabled

3.2.8 Listen-Only Mode *(Silent mode)*

This mode can be activated with the menu item *Controller listen only* if the CAN device supports this mode. An overview about CAN devices that support this mode, is given in the NTCAN Part 1: Application Developers Manual [1], chapter: "Listen-Only Mode*"*.

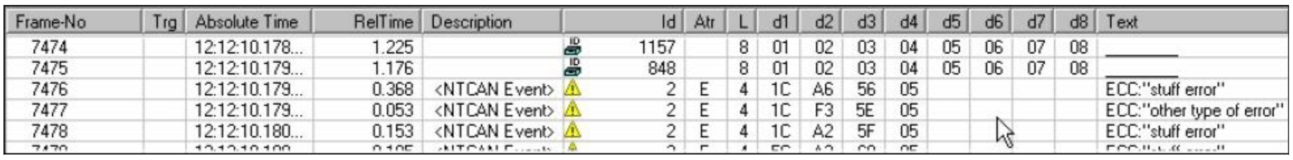
Special features of the Listen-Only Mode:

- In this mode messages can be received but **not** transmitted.
- The CAN controller is **listen-only**. The CAN controller can **neither** send **an acknowledge nor an Error-Frame**, i.e. the CAN board acts as if it is non-existent.
- There must be at least two further participants on the CANbus.

In the listen-only mode the background of the status bar is highlighted (as the tooltip) depending on the colour schema used.

3.2.9 Single Error Diagnostic

The menu item *Single error diagnostic* can only be selected for boards that support this function (for further information see NTCAN Part 1: Application Developers Manual [1], chapter: "Bus Diagnostic").
If the single error diagnostic is active, errors of the single CAN-frames are indicated as event messages.

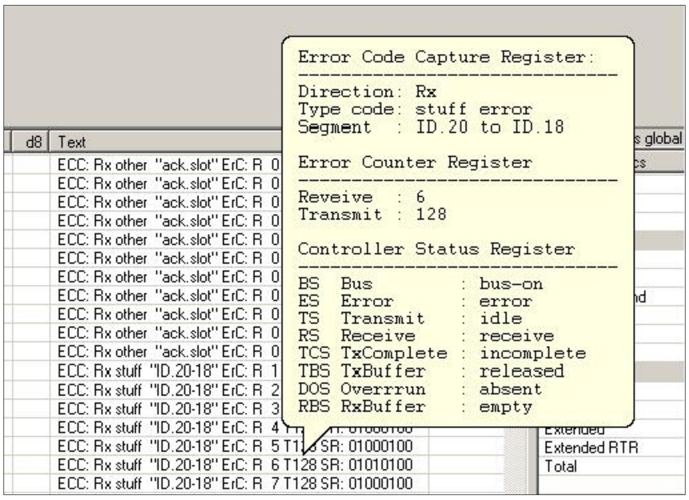


Frame-No	Trg	Absolute Time	RelTime	Description		Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
7474		12:12:10.178...	1.225			1157		8	01	02	03	04	05	06	07	08	________
7475		12:12:10.179...	1.176			848		8	01	02	03	04	05	06	07	08	________
7476		12:12:10.179...	0.368	<NTCAN Event>	⚠	2	E	4	1C	A6	56	05					ECC:"stuff error"
7477		12:12:10.179...	0.053	<NTCAN Event>	⚠	2	E	4	1C	F3	5E	05					ECC:"other type of error"
7478		12:12:10.180...	0.153	<NTCAN Event>	⚠	2	E	4	1C	A2	5F	05					ECC:"stuff error"

Figure 45: Part of the display window with listing of single errors

In the column *Description* the frame is described as <NTCAN Event>.
In the *Text* column the single errors read for the specific CAN frames from the ECC error register of the CAN controller are described. The column *Id* indicates the warning symbol for the error and the event-Id: 2, which encodes the ECC events.
The second byte (d2) contains the number of the errors in the ECC-register, which is defined in the manual of the according CAN-controller.

If the tooltip is enabled via *Advanced Settings*, a detailed description is shown as tooltip.



```
Error Code Capture Register:
--------------------------------
Direction: Rx
Type code: stuff error
Segment  : ID.20 to ID.18

Error Counter Register
--------------------------------
Reveive  : 6
Transmit : 128

Controller Status Register
--------------------------------
BS  Bus        : bus-on
ES  Error      : error
TS  Transmit   : idle
RS  Receive    : receive
TCS TxComplete : incomplete
TBS TxBuffer   : released
DOS Overrrun   : absent
RBS RxBuffer   : empty
```

```
d8  Text
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx other "ack.slot" ErC: R 0
    ECC: Rx stuff "ID.20-18" ErC: R 1
    ECC: Rx stuff "ID.20-18" ErC: R 2
    ECC: Rx stuff "ID.20-18" ErC: R 3
    ECC: Rx stuff "ID.20-18" ErC: R 4
    ECC: Rx stuff "ID.20-18" ErC: R 5 T1  SR: 01000100
    ECC: Rx stuff "ID.20-18" ErC: R 6 T128 SR: 01010100
    ECC: Rx stuff "ID.20-18" ErC: R 7 T128 SR: 01000100
```

Figure 46: Tooltip *Single error diagnostic*

3.2.10 Enable custom NTCAN events

The menu item *Enable custom NTCAN events* in the *CAN* menu is only available with special customer-specific drivers. It enables the sending of NTCAN events via the input bar for send messages.

3.3 Menu Item *Send*

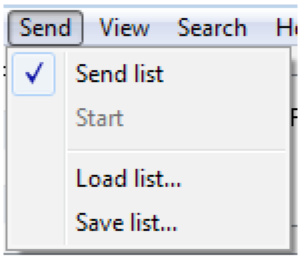
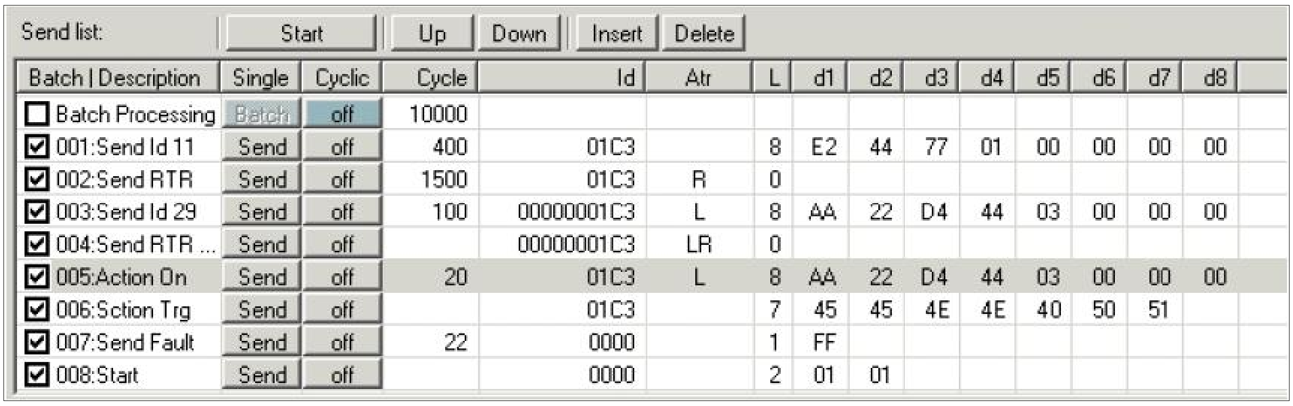


Figure 47: Menu item *Send*

3.3.1 Send List

Click menu item *Send list* to display the list of messages to be transmitted. The represented size of this list can be adjusted by moving the horizontal splitter between display window and *Send list* using the mouse. To hide the list click *Send list* again or minimize the window by using the mouse.



Send list: Start Up Down Insert Delete

Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch Processing	Batch	off	10000											
☑ 001:Send Id 11	Send	off	400	01C3		8	E2	44	77	01	00	00	00	00
☑ 002:Send RTR	Send	off	1500	01C3	R	0								
☑ 003:Send Id 29	Send	off	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☑ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☑ 005:Action On	Send	off	20	01C3	L	8	AA	22	D4	44	03	00	00	00
☑ 006:Sction Trg	Send	off		01C3		7	45	45	4E	4E	40	50	51	
☑ 007:Send Fault	Send	off	22	0000		1	FF							
☑ 008:Start	Send	off		0000		2	01	01						

Figure 48: List of messages to be transmitted

3.3.2 *Start* the Batch List

The menu item *Start* (or *Stop)* starts or stops the processing of the batch list and starts or stops the cyclic transmission. The start button has the same function (see chapter “3.3.4.2 Sending Options”).

3.3.3 Load and Save a Send List *(Load list..., Save list...)*

A *Send list* can be loaded in ASCII-text format. Lists which are generated via the data logging (see chapter “2.7 Trigger and Logging”) can be used (do not forget the conversion into *.txt-format!) or older *Send lists*, which were saved via menu point *Save list...* before.

3.3.4 Function and Handling of the Send List

The *Send list* provides the possibility to transmit up to 999 freely definable CAN messages once only or cyclically repeated. Via a counter function the data values of successive transmit messages can be incremented. Furthermore the received CAN messages can be added to the list for later transmission.
At first this chapter describes the basic functionality of the *Send list*. Then a detailed description of the single buttons and fields of the table will follow.

3.3.4.1 Structure of the Send List

The list contains a row for each CAN message. Above the list you find the buttons to insert (*Insert)*, delete (*Delete)* and arrange (*Up/Down)* the messages and the *Start* button for cyclic transmission. The first row in the *Send list* has the constant entry “Batch Processing” and a special function: It defines the cycle properties of the batch function (see chapter “3.3.4.2 Sending Options”).

Insertion of messages to be transmitted in the *Send list*.

Tx-messages can be entered as described in the following:

- The data and parameters of the Tx-message can be written directly in the fields of the *Send list*.
- The data of the Tx-message are copied to the list from the input row for transmit messages (see chapter “2.5 Input Bar for Send Messages”) by clicking the button *Copy to Send List* . They are appended at the end of the list.
- Transfer of one or more marked received CAN messages from the display window in the *Send list* via the context menu. The context menu is described in chapter “3.3.4.4 Context Menu”.
- Load a list of the messages in text format via the menu item *Load list...* . Lists can be used which are generated by data logging (see chapter “2.7 Trigger and Logging”) (do not forget conversion in *.txt-format!) or former *Send lists* that have been saved via menu item *Save list...* before.

INFORMATION
If the the last loaded *Send list* shall be automatically loaded at the start of CANreal, the name of the lists has to be saved via *File/Store current settings* or *File/Load or save profile settings...*

3.3.4.2 Sending Options

The sending function can only be used if CANreal is in global *Start* or *Pause* mode. The contents of the Tx-messages must have been entered in the list before transmission.

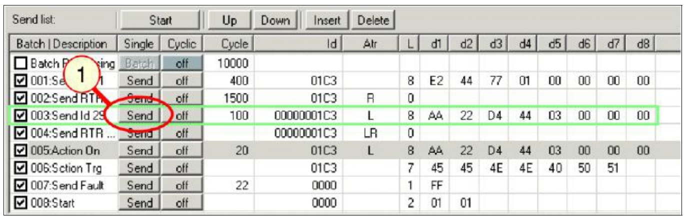
The following table shows the basic sending options. They can be freely combined.

INFORMATION
The cyclic transmission and the batch-processing can be started in parallel.
The start times of the single cyclically transmitted messages are not synchronous to the cycle of the batch transmission.

A successful transmission of a messages can be indicated in the display window if the CAN-ID has been selected correspondingly. Error messages for transmission attempts without success are not listed.

Single Send

Click on the *Send*-button ① in the line of the selected message for a single transmission of the message (marked with green in the example).

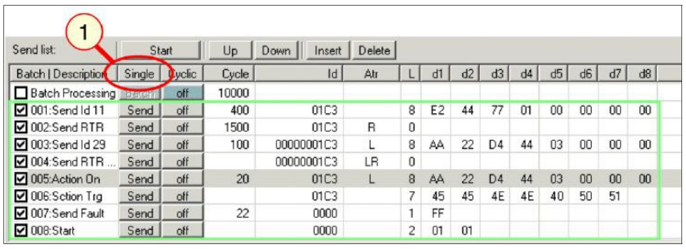


Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch P[illegible]sing	Batch	off	10000											
☑ 001:Se[illegible]1	Send	off	400	01C3		8	E2	44	77	01	00	00	00	00
☑ 002:Send RT[illegible]	Send	off	1500	01C3	R	0								
☑ 003:Send Id 29	Send	off	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☑ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☑ 005:Action On	Send	off	20	01C3	L	8	AA	22	D4	44	03	00	00	00
☑ 006:Sction Trg	Send	off		01C3		7	45	45	4E	4E	40	50	51	
☑ 007:Send Fault	Send	off	22	0000		1	FF							
☑ 008:Start	Send	off		0000		2	01	01						

Figure 49: Single Send function

Single Send All

Clicking the button *Single* ① in the table header will activate all *Send* buttons one after the other without pause. All messages in this example marked with green are transmitted.

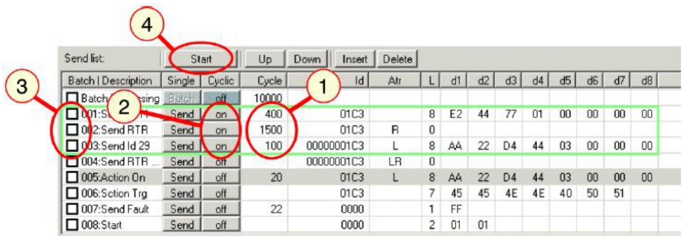


Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch Processing	Batch	off	10000											
☑ 001:Send Id 11	Send	off	400	01C3		8	E2	44	77	01	00	00	00	00
☑ 002:Send RTR	Send	off	1500	01C3	R	0								
☑ 003:Send Id 29	Send	off	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☑ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☑ 005:Action On	Send	off	20	01C3	L	8	AA	22	D4	44	03	00	00	00
☑ 006:Sction Trg	Send	off		01C3		7	45	45	4E	4E	40	50	51	
☑ 007:Send Fault	Send	off	22	0000		1	FF							
☑ 008:Start	Send	off		0000		2	01	01						

Figure 50: Single Send All function

Cycle Send

① In the column *Cycle* the cycle time of every individual message is indicated in [ms].



Send list: Start | Up | Down | Insert | Delete

Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch [illegible]sing	Batch	off	10000											
☐ 001:S[illegible]	Send	on	400	01C3		8	E2	44	77	01	00	00	00	00
☐ 002:Send RTR	Send	on	1500	01C3	R	0								
☐ 003:Send Id 29	Send	on	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☐ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☐ 005:Action On	Send	off	20	01C3	L	8	AA	22	D4	44	03	00	00	00
☐ 006:Sction Trg	Send	off		01C3		7	45	45	4E	4E	40	50	51	
☐ 007:Send Fault	Send	off	22	0000		1	FF							
☐ 008:Start	Send	off		0000		2	01	01						

Figure 51: Cycle Send function

NOTICE
Accuracy of the cycle times:
For short cycle times the base offset of ± 3 ms (accuracy of the Windows timer) prevails. For long cycle times (>1000 ms) an relative error of ±1% (accuracy of the PC's clock) has to be assumed .

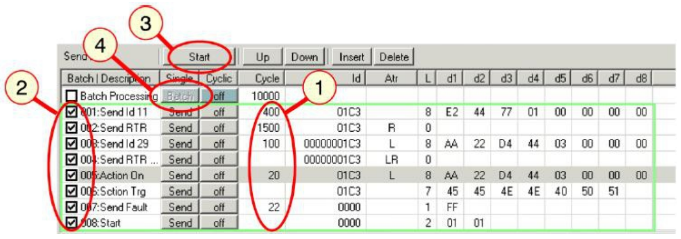
② The buttons of the messages to be transmitted in the column *Cyclic* must indicate *on*

③ The check boxes of the messages in the column *Batch/Description* must be disabled if the synchronous batch-processing is NOT wanted (recommended).

④ A click on the *Start* button starts the cyclic transmission of the configured messages (marked green in this example). To stop the transmission click the *Stop* button (former *Start button)*.

Single Batch

① In the column *Cycle* the latency (measured from the preceding message) of the transmission of the message is indicated in [ms].



Send[illegible] Start | Up | Down | Insert | Delete

Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch Processing	Batch	off	10000											
☑ 001:Send Id 11	Send	off	400	01C3		8	E2	44	77	01	00	00	00	00
☑ 002:Send RTR	Send	off	1500	01C3	R	0								
☑ 003:Send Id 29	Send	off	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☑ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☑ 005:Action On	Send	off	20	01C3	L	8	AA	22	D4	44	03	00	00	00
☑ 006:Sction Trg	Send	off		01C3		7	45	45	4E	4E	40	50	51	
☑ 007:Send Fault	Send	off	22	0000		1	FF							
☑ 008:Start	Send	off		0000		2	01	01						

Figure 52: Single Batch function

② The messages of the batch list that shall be transmitted must be activated in the corresponding checkbox in the column *Batch/Description*.
The buttons in the column *Cyclic* must be *off* (recommended).

③ Click the *Start* button afterwards.

④ To start the processing of the batch list (from top down) click the button *Batch* in the column *Single*. The messages in the example marked green are transmitted one time.

Cycle Batch

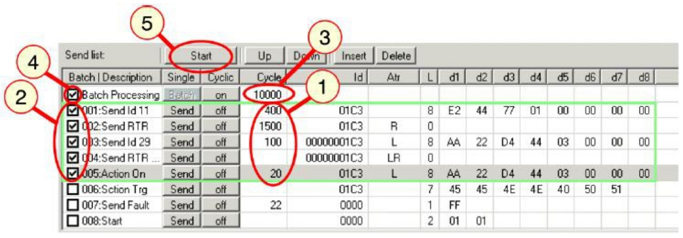
① In the column *Cycle* the latency (measured from the preceding message) of the transmission of the message is indicated in [ms].

② The messages of the batch list that shall be transmitted must be activated in the corresponding checkbox in the column *Batch/Description*.
The buttons in the column *Cyclic* must be *off* (recommended).

③ In the first line of the column *Cycle* the cycle time of the batch processing is indicated in [ms].

④ Batch processing has to be activated in the checkbox in first line of the column *Batch/Description.*

⑤ To start the processing of the batch list (from top down) click the button *Start* in the column *Single*. The messages in the example marked green are transmitted. The transmission is repeated cyclically. Click the *Stop* button to stop the cyclic transmission. If the checkbox is disabled before batch processing, the list is processed to the end and then stopped.



Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☑ Batch Processing	Batch	on	10000											
☑ 001:Send Id 11	Send	off	400	01C3		8	E2	44	77	01	00	00	00	00
☑ 002:Send RTR	Send	off	1500	01C3	R	0								
☑ 003:Send Id 29	Send	off	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☑ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☑ 005:Action On	Send	off	20	01C3	L	8	AA	22	D4	44	03	00	00	00
☐ 006:Sction Trg	Send	off		01C3		7	45	45	4E	4E	40	50	51	
☐ 007:Send Fault	Send	off	22	0000		1	FF							
☐ 008:Start	Send	off		0000		2	01	01						

Figure 53: Cycle Batch function

3.3.4.3 Buttons of the *Send List*

Description of the buttons

Start/Stop

Starts or stops the processing of the batch list and the cyclic transmission.

Up/Down

Moves a marked message or the marked range of the S*end list* one line up or down.

Insert/Dup/Delete

Insert a row below the marked message. Click *Dup* to duplicate or *Delete* to delete the marked massage

Batch/Description

- The Tx-messages are shown here in numerical order; at the right side a user-defined comment on the Tx-message can be entered. The text may contain special characters.
- With the check box the message to be send during the processing of the batch list (Single Batch, Cyclic Batch) can be enabled; A click on the button *Batch/Description* switches between the options “all messages selected”, “individual selection” and “no message selected”.

Send

Click *Send* to instantly transmit the single message defined in this line once.

Single

Clicking the button *Single* in the table header will activate all *Send* buttons one after the other automatically without pause.

Cyclic/Cycle

Cyclic = off	...	the message displayed in this row is **only** transmitted **once**
Cyclic = on	...	the message displayed in this row is transmitted **cyclically**
Cycle = xxxx		In the transmission mode “Cyclic Send” the cycle time for the transmission of the message is specified here in [ms]. In the transmission modes “Single Batch” and “Cyclic Batch” the waiting time **before** the transmission of the message is specified.

By clicking the button *Cyclic* you can switch between the options “*enable all*”, “*mixed*” and “*disable all”*.

3.3.4.4 Context Menu

To open the context menu click in the send list.

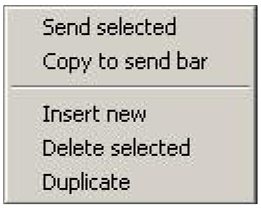
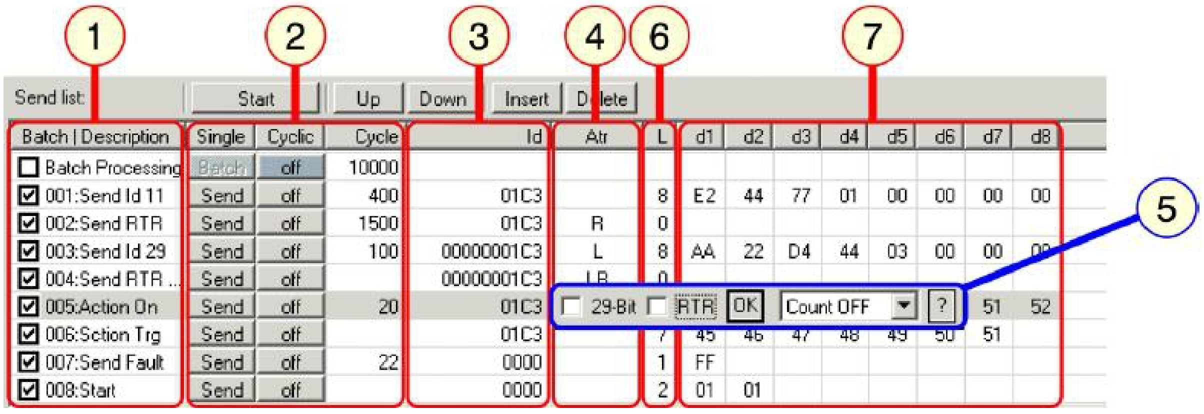


Figure 54: Context menu of the *Send list*

Commands of the context menu	Description
Send selected	Transmit the selected messages
Copy to send bar	Copies the selected CAN message into the send bar for messages that shall be transmitted.
Insert new	CAN messages are inserted at a selected position of the *Send list*.
Delete selected	Delete selected CAN messages
Duplicate	Duplicate selected CAN messages

3.3.4.5 Parameters and Data of the Tx-Messages



Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8
☐ Batch Processing	Batch	off	10000											
☑ 001:Send Id 11	Send	off	400	01C3		8	E2	44	77	01	00	00	00	00
☑ 002:Send RTR	Send	off	1500	01C3	R	0								
☑ 003:Send Id 29	Send	off	100	00000001C3	L	8	AA	22	D4	44	03	00	00	00
☑ 004:Send RTR ...	Send	off		00000001C3	LR	0								
☑ 005:Action On	Send	off	20	01C3									51	52
☑ 006:Sction Trg	Send	off		01C3		7	45	46	47	48	49	50	51	
☑ 007:Send Fault	Send	off	22	0000		1	FF							
☑ 008:Start	Send	off		0000		2	01	01						

Figure 55: Structure of the data to be send

Selecting and editing the input fields can be done by means of the keyboard (*Tab:* to the right, *Shift-Tab:* to the left, *Return:* accept, *Esc:* cancel) or the mouse. Missing entries in the numerical fields are taken as '0'.

	Parameter/Data	Description
①	*Batch/Description*	List number and comment Contains the number of the entry, which is the order of the processing of the entries in the batch list. Furthermore a comment can be assigned to the Tx-message.
②	*Single/Cyclic/Cycle*	See chapter "3.3.4.3 Buttons of the Send List"
③	*Id*	CAN-Identifier (11-bit or 29-bit presentation), in hexadecimal or decimal format, depending on global setting of *IDs decimal*
④	*Atr*	Attributes which are set are shown abbreviated in the field *Atr*: L: 29-bit message — C16: 16-bit counter R: RTR-bit — C32: 32-bit counter BE: Big Endian — C64: 64-bit counter LE: Little Endian
⑤		A click on an *Atr* (attribute) input field opens an dialogue window in which the attributes can be selected. The entries are accepted via button *OK*. *29-Bit...* Select 29-bit CAN-Identifier *RTR...* Set the RTR-bit for transmission *Count xy...* A counter can be selected for test purposes, which increments the content of the message. The following options are provided: Count OFF: No counter — 16: 16-bit counter BE: Big Endian format — 32: 32-bit counter LE: Litte Endian format — 64: 64-bit counter Bytes not used by the counter keep the previous value. *?...* opens a help window
⑥	*L*	Number of data bytes (0...8) to be transmitted
⑦	*d1...d8*	Data of the Tx-message in hexadecimal format

3.4 Menu Item *View*

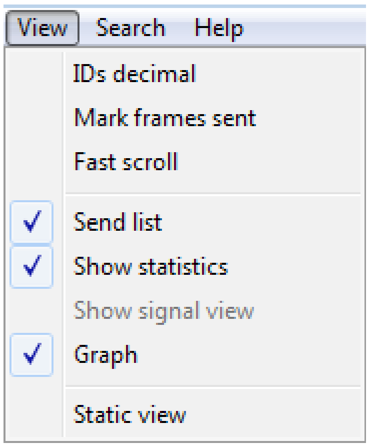


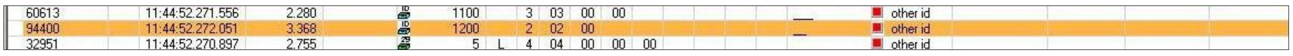
Figure 56: Menu item View

3.4.1 IDs decimal

With this menu item the IDs can be shown in decimal format. See chapter “2.4.3 Add/Delete ID Area (11-Bit Identifier)”.

3.4.2 Mark frames sent

Messages, which have been sent in the selected CAN net can be highlighted with orange by clicking *Mark frames sent*.



60613	11:44:52.271.556	2.280	1100		3	03	00	00		___	🟥	other id
94400	11:44:52.272.051	3.368	1200		2	02	00			___	🟥	other id
32951	11:44:52.270.897	2.755	5	L	4	04	00	00	00		🟥	other id

Figure 57: Mark selected frames per *Mark frames sent*

3.4.3 Fast Scroll

Choose *Fast Scroll* for faster updates of a list of the running CAN messages.

3.4.4 Send List

Click menu item *Send list* to display the list of messages to be transmitted. See chapter “3.3.1 Send List”.

3.4.5 Show Statistics

Click this menu item to open the statistics window. It is shown right of the display window for received messages.
The statistic window is described in detail in chapter “3.2.4 Show Statistics”.

3.4.6 Show Signal View

This menu item is only available if a plugin is used which supports this functionality. With this menu item you can enable or disable the signal view. For further in formation read chapter “Configure Plugins for CAN DBC” from page 70.

3.4.7 Graph

In this field of the program window the statistic data can be shown as curve chart or bar chart.

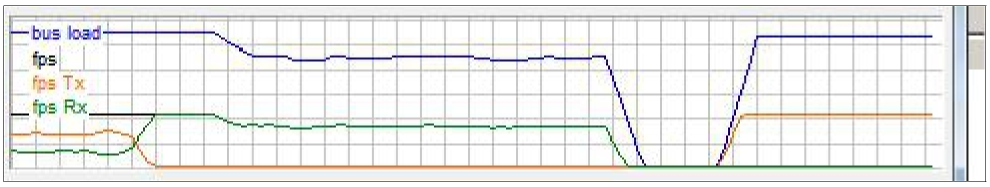
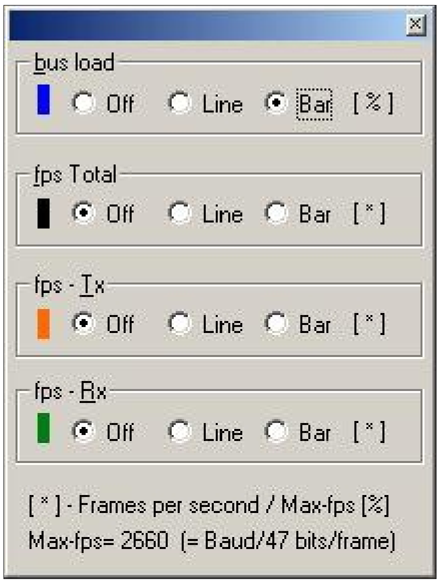


Figure 58: Graphical representation of the data

3.4.7.1 Configuration Menu

With a double click or a click with the right mouse button on the graph the configuration menu (see right) opens. The following configurations are possible:

- Bus load
- Frame rates (*frames_per_second*)



bus load: 🟦 ○ Off ○ Line [✓ Bar] [%]

fps Total: ⬛ [✓ Off] ○ Line ○ Bar [*]

fps - Tx: 🟧 [✓ Off] ○ Line ○ Bar [*]

fps - Rx: 🟩 [✓ Off] ○ Line ○ Bar [*]

[*] - Frames per second / Max-fps [%]
Max-fps= 2660 (= Baud/47 bits/frame)

Figure 59: Configuration menu of the graphs

3.4.8 Static View

See chapter “3.2.7 Static View“ for a detailed description of this menu item.

3.5 Menu item Search

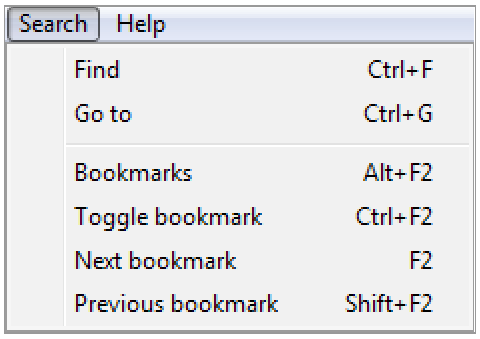


Figure 60: Menu item *Search*

3.5.1 Search for CAN Messages in Display Window (*Find*)

The input window *Find* of the search function can be opened via menu item *Find* or the key combinations [Strg]+[F] or [Ctrl]+[F].

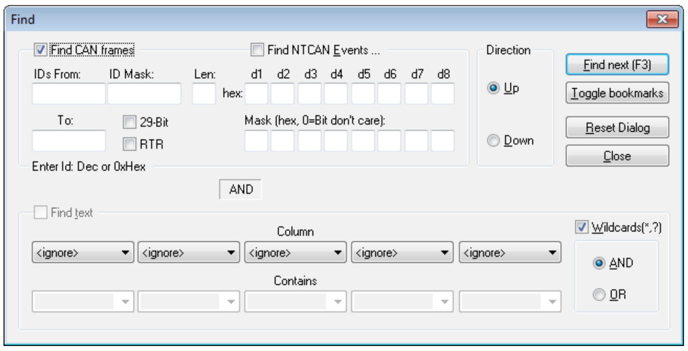


Figure 61: *Find*

Searches for received CAN messages in the display window. The following search parameters can be defined:

Find CAN frames	Searches for binary data in the CAN frames		
IDs From … To	Identifier range to be searched through in decimal or hexadecimal format		
ID Mask	The *ID Mask* masks defined bits in the IDs. The masking only works if values are entered in the fields *IDs From … To*. All bits set to '1' in mask must contain the bit value of *ID From*, to meet the search condition.		
	Example:	ID From:	0x200
		To:	0x2FF
		Mask:	0x1
	All even CAN-IDs are found. *ID From* 0x201 finds all uneven CAN-IDs in the range *ID From...To*.		
29-Bit, RTR	Limitation to the search of messages with 29-bit-CAN-ID and/or messages with enabled RTR-bit (each ON/OFF)		
Len	Limitation to the search of messages with a length of exactly x byte (with x = 0, 1, ...8)		
d1...d8, Mask	Masking of the allowed messages. See chapter “2.7.1.1 Selection via Data-Bit Mask” for an example of the selection of trigger conditions via *Mask*.		

Find NTCAN Events	Is an "OR" condition to *Find CAN frames.* See chapter "2.7.1.2 NTCAN Event Trigger" for a description of the trigger conditions.
Find text	The selection boxes in the upper line of this field contain all columns of the list view – inclusive the plug-in columns. To enable *Find text* a column has to be selected and the search text has to be entered in the line below. *Find text* is logically connected with *Find CAN frames* and *Find NTCAN Events*. For the column texts *AND/OR* operation is possible. Via the checkboxes *AND* and *OR* these operations can be enabled. * or ? can be entered as wildcards if the checkbox is enabled.
Direction	Define search direction (Up = search towards lower frame numbers, i.e. older messages / Down = search towards higher frame numbers, younger) in the list of received messages (as seen from the marked message).

Clicking the button *Find next* (F3) starts the search and the next line with a CAN message is marked. If no message has been selected, the search starts beginning with the last message.
With every further click on the button the cursor will mark the next found message. Function key [F3] has the same function as the button *Find next (F3)*.
Function key [F3] will move the cursor to the line of the next message found even if the *Find* menu is closed.
At the end or at the beginning of the list a new search cycle starts at the respective other end of the list.

Clicking on the button *Toggle Bookmarks* starts the search for received CAN messages with the defined search parameters. The bookmarks of the found messages are toggled (Set ↔ Delete).

A click on the button *Reset Dialog* deletes all dialogue settings.

3.5.2 Go to

With *Go to* function a fast navigation in the list is possible. *List index* moves the cursor to the line specified in the input field.
List per cent (0 ... 100) moves the cursor to the place in the list approximately defined by the value specified in percent in the input field. E.g. a value of 50 percent will move the cursor to the middle of the list.

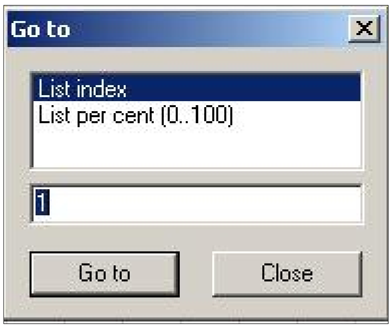
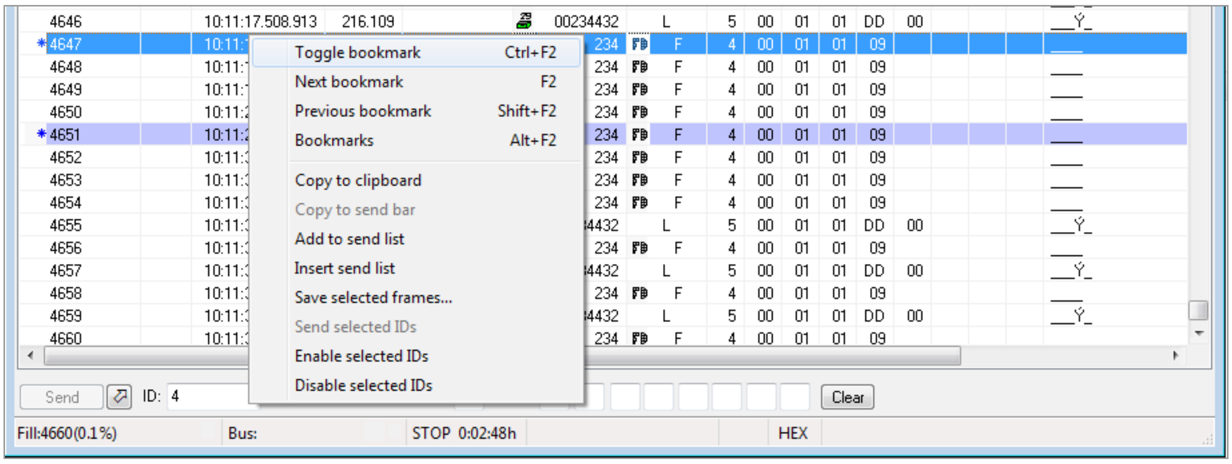


Figure 62: *Go to*

INFORMATION
To search for frame numbers use *Find next (F3)* in the dialogue *Find*.

3.5.3 Bookmark

You can bookmark selected frames, to make it easier to find them at a later time.
Frames with bookmarks are shown in the list view with an asterisk (*) as prefix and background colour (light blue in the example). The selected row is indicated by a blue background. A right mouse click opens the context menu (see also page 14).



	4646	10:11:17.508.913	216.109	00234432		L	5	00	01	01	DD	00	__Ÿ_
*	4647	10:11:1		234	FD	F	4	00	01	01	09		___
	4648	10:11:1		234	FD	F	4	00	01	01	09		___
	4649	10:11:1		234	FD	F	4	00	01	01	09		___
	4650	10:11:2		234	FD	F	4	00	01	01	09		___
*	4651	10:11:2		234	FD	F	4	00	01	01	09		___
	4652	10:11:3		234	FD	F	4	00	01	01	09		___
	4653	10:11:3		234	FD	F	4	00	01	01	09		___
	4654	10:11:3		234	FD	F	4	00	01	01	09		___
	4655	10:11:3		4432		L	5	00	01	01	DD	00	__Ÿ_
	4656	10:11:3		234	FD	F	4	00	01	01	09		___
	4657	10:11:3		4432		L	5	00	01	01	DD	00	__Ÿ_
	4658	10:11:3		234	FD	F	4	00	01	01	09		___
	4659	10:11:3		4432		L	5	00	01	01	DD	00	__Ÿ_
	4660	10:11:3		234	FD	F	4	00	01	01	09		

Figure 63: List view with bookmarks

3.5.3.1 Toggle Bookmark

Choose menu item *Toggle bookmark* to set/delete a bookmark in the selected row (row with blue background). You can also use the key combination [Ctrl.] + [F2] or [*].
The bookmarks can also be set or deleted by clicking with the left mouse button in the field on the left of the frame number.

3.5.3.2 Next Bookmark

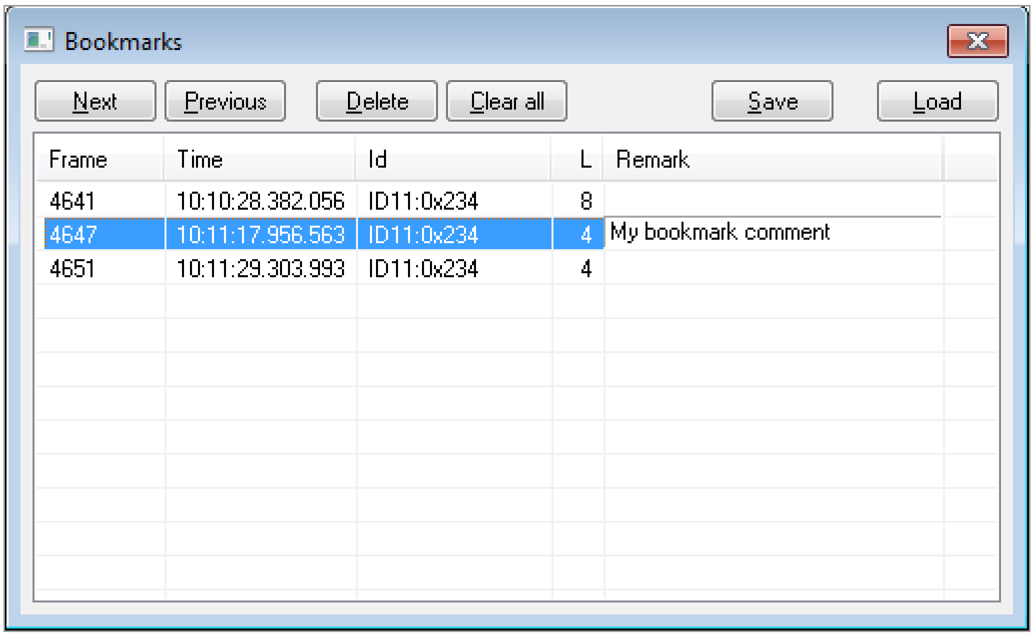
Click menu item *Next bookmark* to go to the next bookmark in the list view.
You can also use the shortcut [F2].

3.5.3.3 Previous Bookmark

Click menu item *Previous bookmark* to go to the previous bookmark in the list view.
You can also use the key combination [Shift] + [F2].

3.5.3.4 Bookmarks

Click on the menu item *Bookmarks* in the context menu to open a list with the bookmarks, that have been set. You can also use the key combination [Alt] + [F2].



Frame	Time	Id	L	Remark
4641	10:10:28.382.056	ID11:0x234	8	
4647	10:11:17.956.563	ID11:0x234	4	My bookmark comment
4651	10:11:29.303.993	ID11:0x234	4	

Figure 64: Overview of bookmarks

The bookmarked frames are listed in this window with *Frame* number, *Time*, *ID* and length (*L*). You can add a comment for each frame in the input field under *Remark.*

Button	
Next	Go to the next bookmark
Previous	Go to the previous bookmark
Delete	Deletes the bookmark at the current cursor position
Clear all	Deletes all bookmarks in the list
Save	Saves the bookmarks with position and comment in a file.
Load	Loads the bookmarks with position and comment. When you open a log-file, a bookmark file with the same name is opened automatically.

When you double-click on a frame in the in the *Bookmarks* window, the cursor of the list view jumps to the same bookmarked frame.

3.6 Menu item *Help*

For further information about CANreal click menu item *Help*.

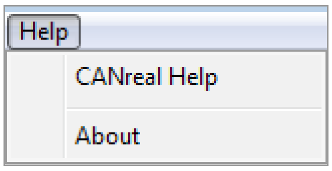


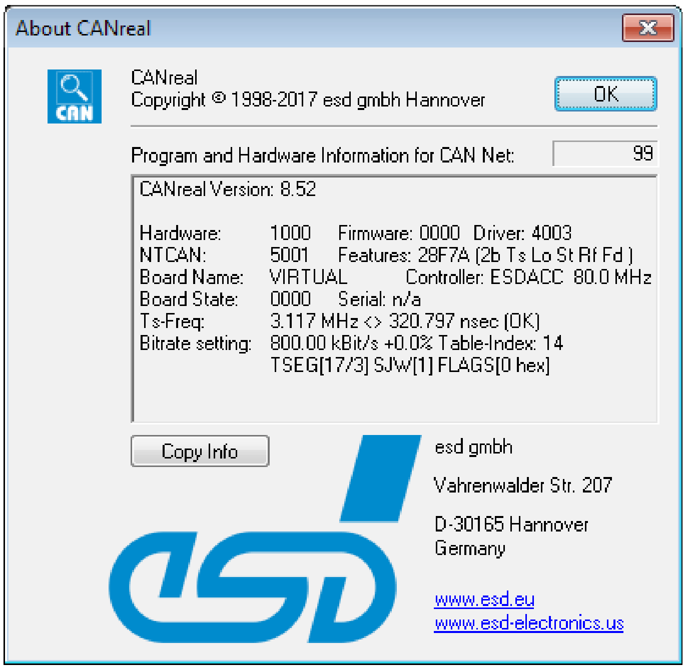
Figure 65: Menu item *Help*

3.6.1 CANreal Help

Clicking *CANreal Help* opens the CANreal Software manual.

3.6.2 About

A click on *About* in the context menu of the menu item *Help*, opens the *About CANreal* information window (see right).



```
About CANreal

CANreal
Copyright © 1998-2017 esd gmbh Hannover                OK

Program and Hardware Information for CAN Net:          99
CANreal Version: 8.52

Hardware:       1000    Firmware: 0000  Driver: 4003
NTCAN:          5001    Features: 28F7A (2b Ts Lo St Rf Fd )
Board Name:     VIRTUAL         Controller: ESDACC  80.0 MHz
Board State:    0000    Serial: n/a
Ts-Freq:        3.117 MHz <> 320.797 nsec (OK)
Bitrate setting: 800.00 kBit/s +0.0% Table-Index: 14
                TSEG[17/3] SJW[1] FLAGS[0 hex]

Copy Info

esd gmbh
Vahrenwalder Str. 207
D-30165 Hannover
Germany

www.esd.eu
www.esd-electronics.us
```

Figure 66: Program and hardware information

The window provides program and hardware information about the selected CAN net (net 0 here), the CANreal version, the hardware- and firmware- and the driver and the NTCAN versions.

Features shows functions that are supported by the CAN-device. The functions are coded as hexadecimal values. The acronyms in brackets have the following meanings:

2b...	CAN 2.0B support
Ts...	Time stamping
Sd...	Smart Disconnect
Lo...	Listen-Only-Mode is supported
St...	Statistic (CAN Bus Statistic is supported)
Rf...	Smart ID Filter (Adaptive ID Filter) Driver support for 29-bit ID filter
Fd...	CAN FD support

Board State: Shows the state of the board

0000	OK
0001	need Firmware update
0002	general HW-Error
>0002	meaning depends on HW and SW used, call esd for further information

Ts-Freq:	If the CAN device and the driver support the hardware timestamp, the row *Ts-Freq...* indicates the timestamp frequency. The timestamp frequency is shown in MHz and the resolution is shown in nano seconds.
Bitrate Setting:	Currently set bit rate with accuracy. See NTCAN-API manual for further information on this.

INFORMATION
For further information about the single functions please read the NTCAN Part 1: Application Developers Manual [1].

4. Plugins

4.1 External Plugins

External plugins are available on request for example:

- J1939
- Plugins for particular protocols (e.g. in the aeroplane)

For C-programmers in chapter “Development of Plugins for CANreal” from page 88.
External plugins are searched for in the subdirectory „canrealplugins“ of the CANreal application directory.

4.2 Internal Plugins

Internal plugins are

- CAN-DBC
- CAN-FD (Hardware is not yet available)

Internal plugins are implemented in CANreal and do not need any DLL in the plugin directory (canrealplugins)

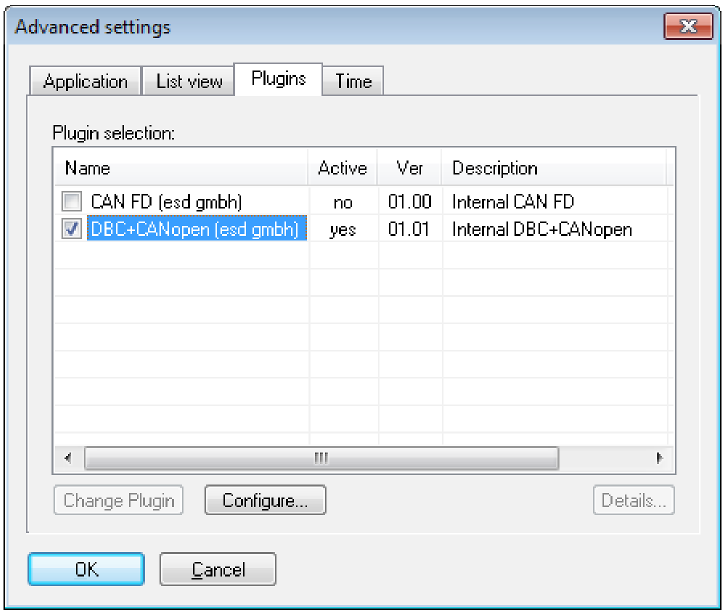
4.3 Internal Plugin for CAN-DBC (Data Base CAN) and CANopen

INFORMATION
The plugin comprises the support of CAN-DBC and CANopen messages. Both can be enabled together but both functions are described separately here (for CANopen description see chapter “Configure Plugin for CANopen” on page 73).

4.3.1 Data Base CAN / CAN-DBC

"CAN Data base files" are text files, which define rules to convert CAN raw data into physical data (real data). They contain for example: Message names (name of a CAN frame with a particular CAN-ID), names of process variables (signal names), data types and conversion factors.

Plugins



Advanced settings

Application | List view | Plugins | Time

Plugin selection:

Name	Active	Ver	Description
☐ CAN FD (esd gmbh)	no	01.00	Internal CAN FD
☑ DBC+CANopen (esd gmbh)	yes	01.01	Internal DBC+CANopen

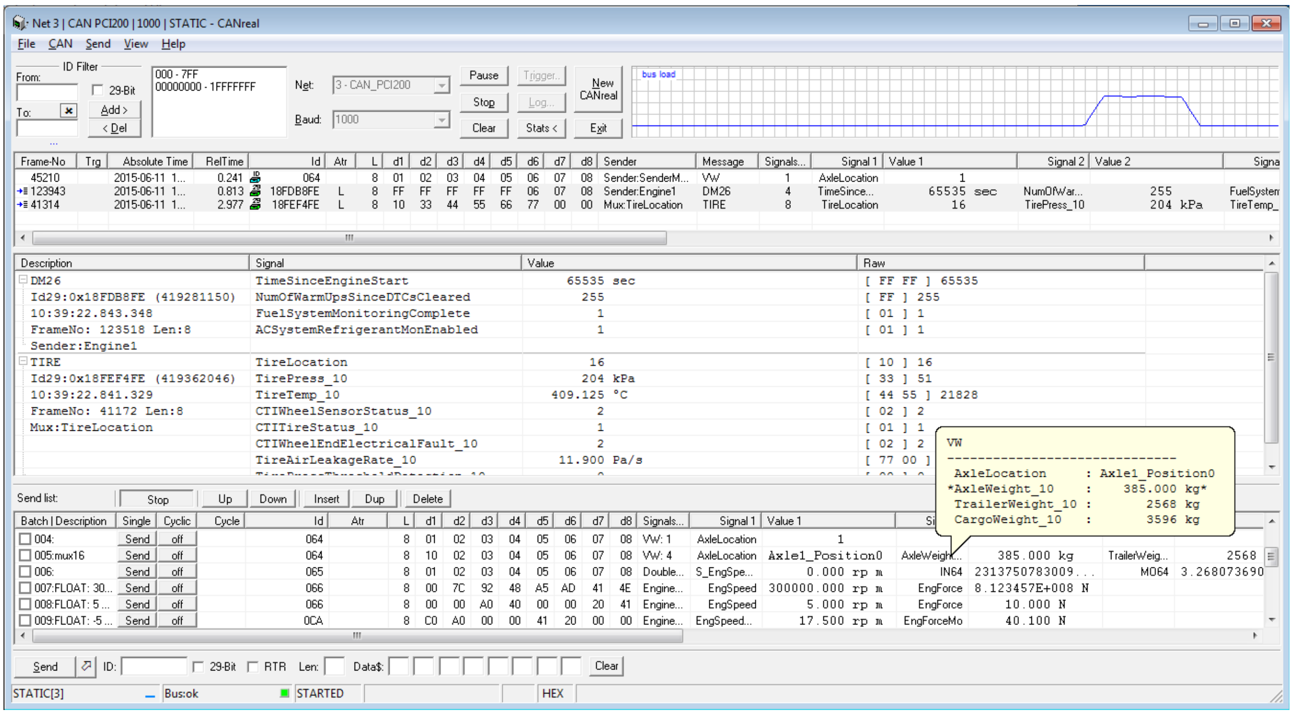
To choose the CAN-DBC plugin click on the menu item *File* in the main menu and choose *Advanced settings*.

Select the tab *Plugins* in the *Advanced settings* window.

Figure 67: Choose CAN-DBC plugin

Scope of the support of CAN-DBC in CANreal:
Message name, sender name, signal name, value name (Enum), Multiplexed signals, Val type, Motorola/Intel, 1..64-bit signals.
J1939 and CAN-FD messages with more than 8 Byte data length are not supported.

INFORMATION
For J1939 a separate plugin is available.



Net 3 | CAN PCI200 | 1000 | STATIC - CANreal

```
000 - 7FF
00000000 - 1FFFFFFF
```

Net: 3 - CAN_PCI200
Baud: 1000

Frame-No	Trg	Absolute Time	RelTime	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Sender	Message	Signals...	Signal 1	Value 1	Signal 2	Value 2	Signa
45210		2015-06-11 1...	0.241	064		8	01	02	03	04	05	06	07	08	Sender:SenderM...	VW	1	AxleLocation	1			
123943		2015-06-11 1...	0.813	18FDB8FE	L	8	FF	FF	FF	FF	FF	06	07	08	Sender:Engine1	DM26	4	TimeSince...	65535 sec	NumOfWar...	255	FuelSystem
41314		2015-06-11 1...	2.977	18FEF4FE	L	8	10	33	44	55	66	77	00	00	Mux:TireLocation	TIRE	8	TireLocation	16	TirePress_10	204 kPa	TireTemp_

Description	Signal	Value	Raw
DM26	TimeSinceEngineStart	65535 sec	[FF FF] 65535
Id29:0x18FDB8FE (419281150)	NumOfWarmUpsSinceDTCsCleared	255	[FF] 255
10:39:22.843.348	FuelSystemMonitoringComplete	1	[01] 1
FrameNo: 123518 Len:8	ACSystemRefrigerantMonEnabled	1	[01] 1
Sender:Engine1			
TIRE	TireLocation	16	[10] 16
Id29:0x18FEF4FE (419362046)	TirePress_10	204 kPa	[33] 51
10:39:22.841.329	TireTemp_10	409.125 °C	[44 55] 21828
FrameNo: 41172 Len:8	CTIWheelSensorStatus_10	2	[02] 2
Mux:TireLocation	CTITireStatus_10	1	[01] 1
	CTIWheelEndElectricalFault_10	2	[02] 2
	TireAirLeakageRate_10	11.900 Pa/s	[77 00]

Send list: Stop | Up | Down | Insert | Dup | Delete

Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Signals...	Signal 1	Value 1	Si			
☐ 004:	Send	off		064		8	01	02	03	04	05	06	07	08	VW: 1	AxleLocation	1				
☐ 005:mux16	Send	off		064		8	10	02	03	04	05	06	07	08	VW: 4	AxleLocation	Axle1_Position0	AxleWeigh...	385.000 kg	TrailerWeig...	2568
☐ 006:	Send	off		065		8	01	02	03	04	05	06	07	08	Double...	S_EngSpe...	0.000 rp m	IN64	2313750783009...	MO64	3.268073690
☐ 007:FLOAT: 30...	Send	off		066		8	00	7C	92	48	A5	AD	41	4E	Engine...	EngSpeed	300000.000 rp m	EngForce	8.123457E+008 N		
☐ 008:FLOAT: 5 ...	Send	off		066		8	00	00	A0	40	00	00	20	41	Engine...	EngSpeed	5.000 rp m	EngForce	10.000 N		
☐ 009:FLOAT: -5 ...	Send	off		0CA		8	C0	A0	00	00	41	20	00	00	Engine...	EngSpeed...	17.500 rp m	EngForceMo	40.100 N		

```
VW
--------------------------------
 AxleLocation      : Axle1_Position0
*AxleWeight_10     :    385.000 kg*
 TrailerWeight_10  :       2568 kg
 CargoWeight_10    :       3596 kg
```

STATIC[3] | Bus:ok | STARTED | HEX

Figure 68: CANreal with DBC plugin enabled

Changes in the CANreal user interface when the DBC plugin is enabled:

Depending on the configuration of the DBC plugin, in the centre of the program window between the list view for received CAN messages and the send list a signal view is displayed. Furthermore additional columns (wit tooltips) are inserted in the list view and in the send list.

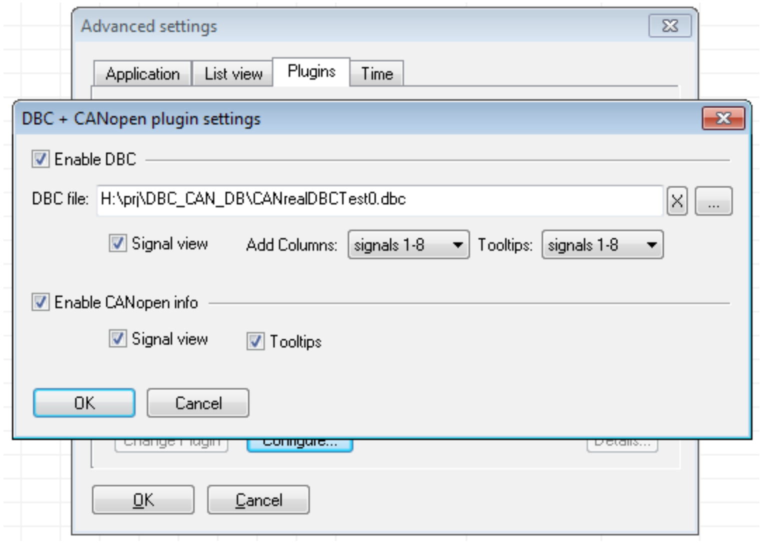
The standard columns *Text* and *Description* are redefined:
Text -> *Sender* (CAN-DBC transmitter name or name of the multiplexer message)
Description -> *Message* (DBC name of the CAN message)

INFORMATION
All standard columns (*FrameNo, TRG, ID, Len, Data, Text, Description*) can be hidden or rearranged: Click *File* -> *Advanced settings* and select tab *List View*

- In the box *Columns visible* single columns of the list can be selected or deselected.
- The column order can be arranged: Drag column headers
-> for example text- and description columns can be arranged on the right and next the columns of the DBC plugin. The plugin columns can not be rearranged and can only be hidden via the plugin configuration dialog.

4.3.2 Configure Plugins for CAN DBC

After choosing the plugin in the tab *Plugins* the dialog *DBC + CANopen plugin settings* is displayed. This dialog can also be opened in "Stop" state of CANreal via the *Configure* button in the *Plugins* dialog.
For support of CAN-DBC the *Enable DBC* box has to be activated and a DBC file has to be entered in the field *DBC File.* Or click on the […] button to select a file in the file system (*Open file - Dialog*).



DBC + CANopen plugin settings

☑ Enable DBC

DBC file: H:\prj\DBC_CAN_DB\CANrealDBCTest0.dbc

☑ Signal view Add Columns: signals 1-8 Tooltips: signals 1-8

☑ Enable CANopen info

☑ Signal view ☑ Tooltips

OK Cancel

Figure 69: DBC + CANopen plugin settings

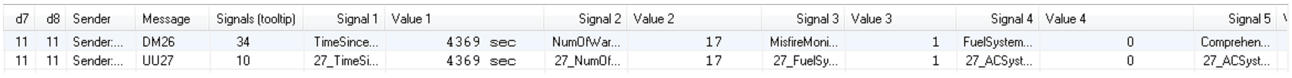
Signal view If this box is activated the *signal view* (see above) is displayed.
Signal view can also be diabled via the main menu item.
Click menu item *View* and select *Show signal view* to display the signal view.

Add columns In the CANreal list view a column *Signals* can be inserted. Additionally 16 columns for up to 8 signal value pairs can be added for the display of the signals (Signal1/Value1 .. Signal8/Value8).

Add columns	
none:	no column display
signal count:	one column *Signals*, which contains the number of the signals
signals 1-8:	one column *Signals* and 8 *Signal/Value* pairs

The order of the signals in columns corresponds to the start bit number of the DBC description.

INFORMATION
If there are more than 8 signals the first 8 non-binary signals (≥ 8-bit raw value) are shown. Only if less than 8 non-binary signals do exist, also the binary signals (< 8 bit, e.g. 1-bit boolean) are shown in the following columns.



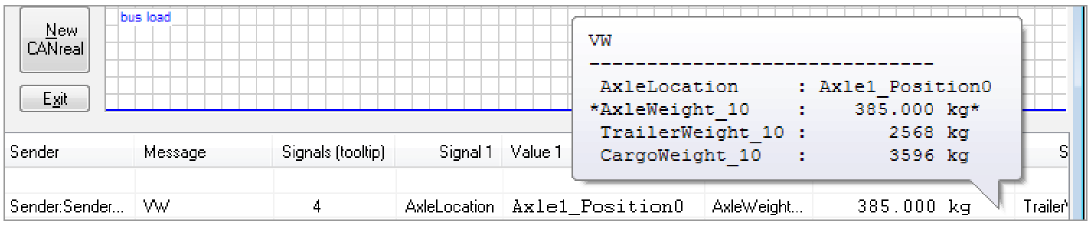
d7	d8	Sender	Message	Signals (tooltip)	Signal 1	Value 1	Signal 2	Value 2	Signal 3	Value 3	Signal 4	Value 4	Signal 5
11	11	Sender:...	DM26	34	TimeSince...	4369 sec	NumOfWar...	17	MisfireMoni...	1	FuelSystem...	0	Comprehen...
11	11	Sender:...	UU27	10	27_TimeSi...	4369 sec	27_NumOf...	17	27_FuelSy...	1	27_ACSyst...	0	27_ACSyst...

Figure 70: Additional columns for signal names and values

Tooltips	Activate the tooltip if the mouse pointer is positioned on the columns *Signals* or *Signal1..8* or *Value 1..8.*

Tooltips none:	no tooltips
signal count:	show tooltips only on column *Signals*
signals 1-8:	show tooltips on all these columns

Structure of the tooltip display: In the header the signal name is shown. Then the single signals with value and unit are shown one below the other.
The current signal column is highlighted with "*".



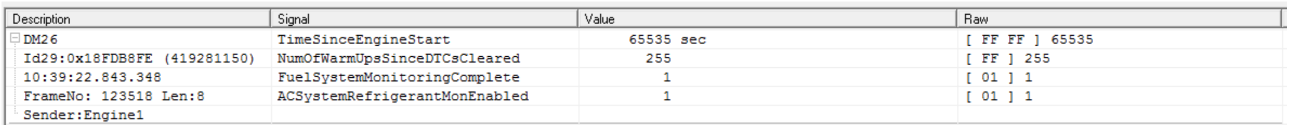
```
VW
--------------------------------
 AxleLocation       : Axle1_Position0
*AxleWeight_10      :    385.000 kg*
 TrailerWeight_10   :       2568 kg
 CargoWeight_10     :       3596 kg
```

Sender	Message	Signals (tooltip)	Signal 1	Value 1			S
Sender:Sender...	VW	4	AxleLocation	Axle1_Position0	AxleWeight...	385.000 kg	Trailer

Figure 71: Tooltip on a signal column

INFORMATION
The settings and configuration of the plugin should be saved for the next use of CANreal. For example via menu item *File → Store current settings.*

4.3.3 Structure and Functionality of the Signal View



Description	Signal	Value	Raw
DM26	TimeSinceEngineStart	65535 sec	[FF FF] 65535
Id29:0x18FDB8FE (419281150)	NumOfWarmUpsSinceDTCsCleared	255	[FF] 255
10:39:22.843.348	FuelSystemMonitoringComplete	1	[01] 1
FrameNo: 123518 Len:8	ACSystemRefrigerantMonEnabled	1	[01] 1
Sender:Engine1			

The signal view contains 4 columns:

Description:	Message name (from DBC), CAN-ID Hex (Dec in brackets), Timestamp (Hour:Minute:Second.Millisecond.Microsecond), Frame number (consecutive), CAN-data length, Transmitter or Mux (=Multiplexer Signal, from DBC)
Signal:	Signal name (from DBC)
Value:	Value + unit according to DBC description
Raw:	Raw value ([Hex] + numerical value without conversion to physical value)

The order of the signals listed one below the other correspond to the start bit number in the DBC description.
Select the CAN messages which shall be shown in the signal view.

The CAN messages are not listed automatically in the signal view. Only selected messages are shown. There are two selection modes, which depend on the CANreal list mode, the CANreal Scroll View and the CANreal Static View.

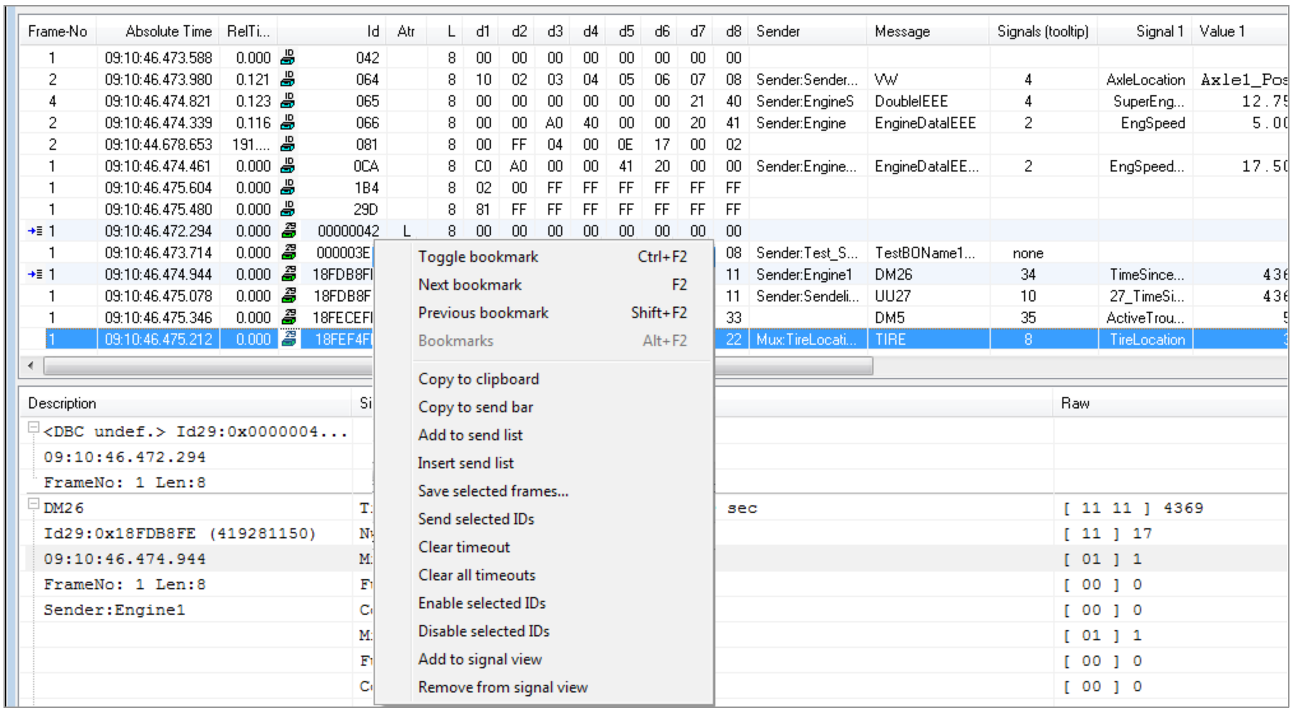
4.3.3.1 CANreal Scroll View with DBC Plugin

To select a row in list view click on it or additionally hold the Strg+/Shift-key and click on more rows in the list to select further received CAN-messages. The selected CAN messages (maximum 200) will be registered in the *signal view* and displayed itemised in signals.

INFORMATION
This kind of selection and display is also available in the viewer of recorded logfiles. (Click on menu item *File* and select *Open logfiles*)

4.3.3.2 CANreal Static View with DBC Plugin

The selection of the messages in *Static view* is as described for the selection of messages in the scroll view. Open the context menu (right mouse button) and click on *Add to signal view.* The selected messages are displayed in the signal view now.



Frame-No	Absolute Time	RelTi...	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Sender	Message	Signals (tooltip)	Signal 1	Value 1
1	09:10:46.473.588	0.000	042		8	00	00	00	00	00	00	00	00					
2	09:10:46.473.980	0.121	064		8	10	02	03	04	05	06	07	08	Sender:Sender...	VW	4	AxleLocation	Axle1_Pos
4	09:10:46.474.821	0.123	065		8	00	00	00	00	00	00	21	40	Sender:EngineS	DoubleIEEE	4	SuperEng...	12.75
2	09:10:46.474.339	0.116	066		8	00	00	A0	40	00	00	20	41	Sender:Engine	EngineDataIEEE	2	EngSpeed	5.00
2	09:10:44.678.653	191....	081		8	00	FF	04	00	0E	17	00	02					
1	09:10:46.474.461	0.000	0CA		8	C0	A0	00	00	41	20	00	00	Sender:Engine...	EngineDataIEE...	2	EngSpeed...	17.50
1	09:10:46.475.604	0.000	1B4		8	02	00	FF	FF	FF	FF	FF	FF					
1	09:10:46.475.480	0.000	29D		8	81	FF	FF	FF	FF	FF	FF	FF					
1	09:10:46.472.294	0.000	00000042	L	8	00	00	00	00	00	00	00	00					
1	09:10:46.473.714	0.000	000003E										08	Sender:Test_S...	TestBOName1...	none		
1	09:10:46.474.944	0.000	18FDB8FI										11	Sender:Engine1	DM26	34	TimeSince...	436
1	09:10:46.475.078	0.000	18FDB8F										11	Sender:Sendeli...	UU27	10	27_TimeSi...	436
1	09:10:46.475.346	0.000	18FECEFI										33		DM5	35	ActiveTrou...	5
1	09:10:46.475.212	0.000	18FEF4FI										22	Mux:TireLocati...	TIRE	8	TireLocation	3

Description	Si		Raw
<DBC undef.> Id29:0x0000004...			
09:10:46.472.294			
FrameNo: 1 Len:8			
DM26	T	sec	[11 11] 4369
Id29:0x18FDB8FE (419281150)	N		[11] 17
09:10:46.474.944	M		[01] 1
FrameNo: 1 Len:8	F		[00] 0
Sender:Engine1	C		[00] 0
	M		[01] 1
	F		[00] 0
	C		[00] 0

Figure 72: Context menu in list view using a plugin that supports signal view

4.3.3.3 Add / Remove Messages

Click on *Add to signal view* in the context menu to add a selected message to signal view.
The selected rows are light blue backgrounded. Additionally, on the left in the column *FrameNo* a symbol (arrow with list) is displayed.

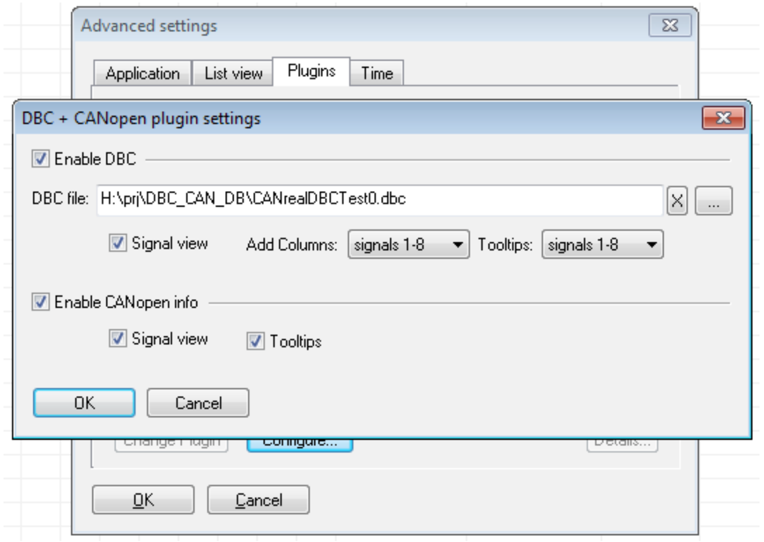
Click *Remove from signal view* in the context menu to remove the message from the signal view.

The CAN messages which are added to the list view via the static view remain selected and will be updated cyclically with the current data contents of the received CAN messages.

4.3.4 Configure Plugin for CANopen

After choosing the plugin in the tab *Plugins* the dialog *DBC + CANopen plugin settings* is displayed. This dialog can also be opened in "Stop" state of CANreal via the *Configure* button in the *Plugins* dialog.

Activate the *Enable CANopen info* box.



DBC + CANopen plugin settings

☑ Enable DBC

DBC file: H:\prj\DBC_CAN_DB\CANrealDBCTest0.dbc

☑ Signal view Add Columns: signals 1-8 Tooltips: signals 1-8

☑ Enable CANopen info

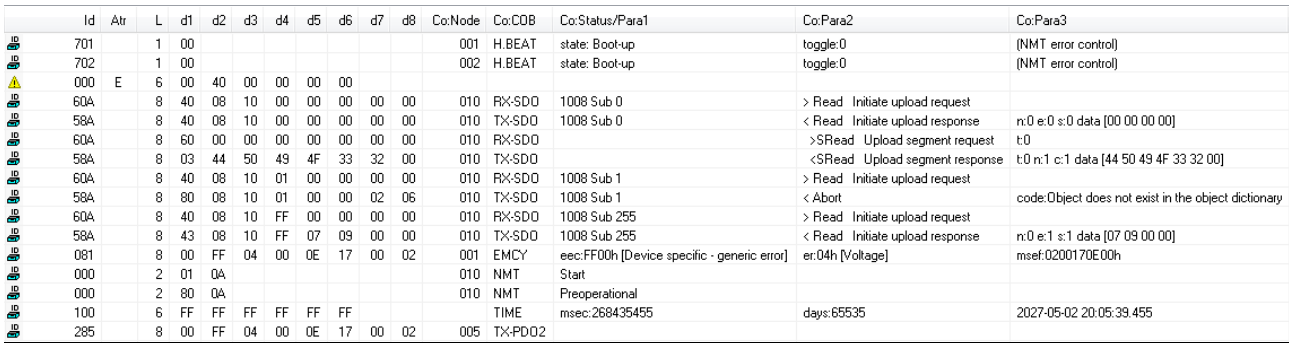
☑ Signal view ☑ Tooltips

OK Cancel

Figure 73: DBC + CANopen plugin setting

If the CANopen info support is enabled, the CANreal interface is displayed with additional features as described in the following

Depending on the configuration of the CANopen plugin an additional *Signal view* is displayed between the *list view* for received CAN messages and the *send list* (see figure 68 on page 68).
Additional columns (optional with tooltips) are displayed in the list view and in the send list.
If CAN-DBC and CANopen are both enabled, the CANopen columns are shown on the left of the CAN-DBC columns.



	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Co:Node	Co:COB	Co:Status/Para1	Co:Para2	Co:Para3
	701		1	00								001	H.BEAT	state: Boot-up	toggle:0	(NMT error control)
	702		1	00								002	H.BEAT	state: Boot-up	toggle:0	(NMT error control)
⚠	000	E	6	00	40	00	00	00	00							
	60A		8	40	08	10	00	00	00	00	00	010	RX-SDO	1008 Sub 0	> Read Initiate upload request	
	58A		8	40	08	10	00	00	00	00	00	010	TX-SDO	1008 Sub 0	< Read Initiate upload response	n:0 e:0 s:0 data [00 00 00 00]
	60A		8	60	00	00	00	00	00	00	00	010	RX-SDO		>SRead Upload segment request	t:0
	58A		8	03	44	50	49	4F	33	32	00	010	TX-SDO		<SRead Upload segment response	t:0 n:1 c:1 data [44 50 49 4F 33 32 00]
	60A		8	40	08	10	01	00	00	00	00	010	RX-SDO	1008 Sub 1	> Read Initiate upload request	
	58A		8	80	08	10	01	00	00	02	06	010	TX-SDO	1008 Sub 1	< Abort	code:Object does not exist in the object dictionary
	60A		8	40	08	10	FF	00	00	00	00	010	RX-SDO	1008 Sub 255	> Read Initiate upload request	
	58A		8	43	08	10	FF	07	09	00	00	010	TX-SDO	1008 Sub 255	< Read Initiate upload response	n:0 e:1 s:1 data [07 09 00 00]
	081		8	00	FF	04	00	0E	17	00	02	001	EMCY	eec:FF00h [Device specific - generic error]	er:04h [Voltage]	msef:0200170E00h
	000		2	01	0A							010	NMT	Start		
	000		2	80	0A							010	NMT	Preoperational		
	100		6	FF	FF	FF	FF	FF	FF				TIME	msec:268435455	days:65535	2027-05-02 20:05:39.455
	285		8	00	FF	04	00	0E	17	00	02	005	TX-PDO2			

Figure 74: CANreal list view with CANopen plugin enabled

Signal view	If this box is activated the *signal view* is displayed. Additional columns, that contain the CANopen data are shown in the *Signal view* range (see page 74).
Tooltips	With this box you can activate the tooltip, which is shown if the mouse pointer is positioned on the additional columns.

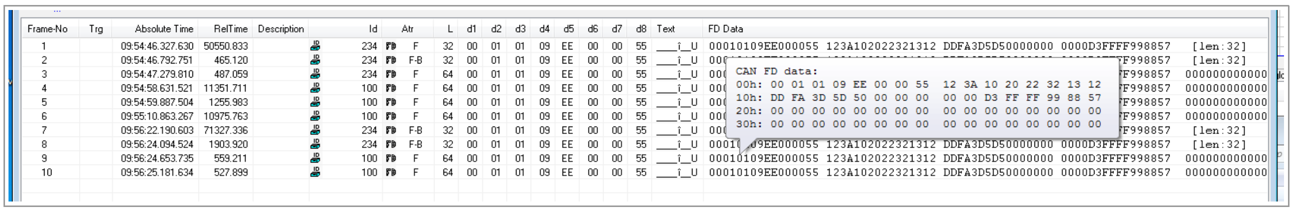
4.3.4.1 Additional CANopen Data

In the following the additional columns are described, which contain the data of the CAN messages described according to the „Predefined Connection Set“ of the CANopen Standard.

Co:Node	Number of the CANopen node (Node-ID), to which the message is related to, i.e. number of the transmitting node or number of the addressed node at NMT-messages.
Co:COB	Name of the COB (Communication Object Identifier): NMT, SYNC, TIME, EMCY, RX-PDO1..4, TX-PDO1..4, RX-SDO, TX-SDO, H.BEAT(= Heartbeat, Node Guarding, Boot-up); LSS: M-LSS, S-LSS
Co:Status/Para1	State of the CAN message, e.g. "Abort" at SDO, wrong length, or further bit coded contents of the message
Co:Para2	further bit coded contents of the message
Co:Para3	further bit coded contents of the message

4.4 Internal Plugin for CAN FD

To select the CAN FD plugin click on the menu item *File* in the main menu and then on menu item *Advanced settings*.
Select the register *Plugins* in the window *Advanced settings* and then CAN FD, as shown in Figure 67, on page 68.
The plugin inserts a column *FD Data* and a tooltip for the CAN FD data in the list view.

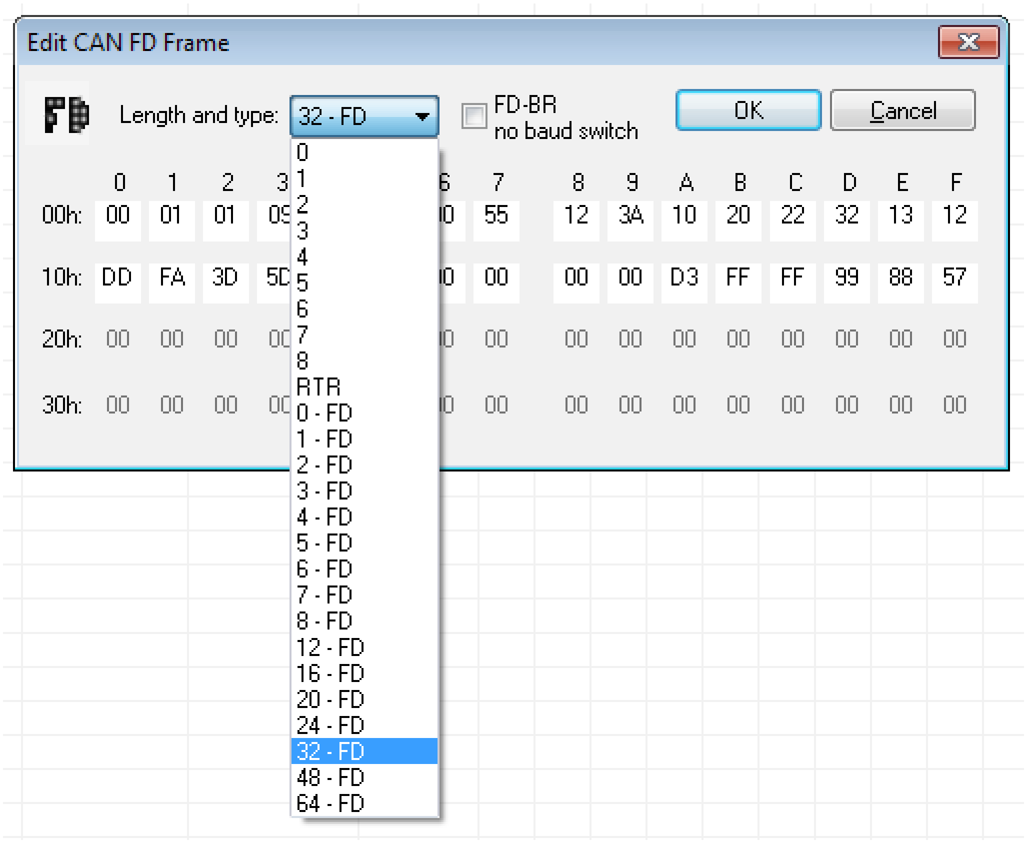


Frame-No	Trg	Absolute Time	RelTime	Description	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text	FD Data
1		09:54:46.327.630	50550.833		234	FD F	32	00	01	01	09	EE	00	00	55	____î__U	00010109EE000055 123A102022321312 DDFA3D5D50000000 0000D3FFFF998857 [len:32]
2		09:54:46.792.751	465.120		234	FD F-B	32	00	01	01	09	EE	00	00	55	____î__U	000 [illegible] F998857 [len:32]
3		09:54:47.279.810	487.059		234	FD F	64	00	01	01	09	EE	00	00	55	____î__U	000 [illegible] FF998857 000000000000
4		09:54:58.631.521	11351.711		100	FD F	64	00	01	01	09	EE	00	00	55	____î__U	000 [illegible] FF998857 000000000000
5		09:54:59.887.504	1255.983		100	FD F	64	00	01	01	09	EE	00	00	55	____î__U	000 [illegible] FF998857 000000000000
6		09:55:10.863.267	10975.763		100	FD F	64	00	01	01	09	EE	00	00	55	____î__U	000 [illegible] FF998857 000000000000
7		09:56:22.190.603	71327.336		234	FD F-B	32	00	01	01	09	EE	00	00	55	____î__U	000 [illegible] FF998857 [len:32]
8		09:56:24.094.524	1903.920		234	FD F-B	32	00	01	01	09	EE	00	00	55	____î__U	0001 [illegible] 09EE000055 123A102022321312 DDFA3D5D50000000 0000D3FFFF998857 [len:32]
9		09:56:24.653.735	559.211		100	FD F	64	00	01	01	09	EE	00	00	55	____î__U	00010109EE000055 123A102022321312 DDFA3D5D50000000 0000D3FFFF998857 000000000000
10		09:56:25.181.634	527.899		100	FD F	64	00	01	01	09	EE	00	00	55	____î__U	00010109EE000055 123A102022321312 DDFA3D5D50000000 0000D3FFFF998857 000000000000

```
CAN FD data:
00h: 00 01 01 09 EE 00 00 55  12 3A 10 20 22 32 13 12
10h: DD FA 3D 5D 50 00 00 00  00 00 D3 FF FF 99 88 57
20h: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
30h: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

Figure 75: *FD Data* and Tooltip

Click on the column FD Data in the send list to open the dialogue window *Edit CAN FD Frame.* In this dialogue window you can edit the CAN FD data.



Edit CAN FD Frame

Length and type: 32 - FD

FD-BR no baud switch

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00h:	00	01	01	09	[illegible]	[illegible]	00	55	12	3A	10	20	22	32	13	12
10h:	DD	FA	3D	5D	[illegible]	[illegible]	00	00	00	00	D3	FF	FF	99	88	57
20h:	00	00	00	00	[illegible]	[illegible]	00	00	00	00	00	00	00	00	00	00
30h:	00	00	00	00	[illegible]	[illegible]	00	00	00	00	00	00	00	00	00	00

Figure 76: *Edit CAN FD Frame* with selection of *Length and Type*

5. Examples

5.1 Example for a Log-file

Recording of a Log-file (see chapter" 2.7 Trigger and Logging")
Conversion of a Log-file in ASCII-text format (see chapter "3.1.8 Open Logfiles")

```
[--] ----------------------------------------------
[--] CANreal Logfile Header (created)
[--] ----------------------------------------------
[F1] Source:              \log.csplog
[F2] Split file series:   log
[F3] File serial number:  n/a
[F4] Id-Description file: Id2Description.txt
[F5] Header type:         0
[F6] Header length:       2093
[F7] File date:           01.08.2017
[F8] File time:           08:55:03
[S1] Hardware:            1016
[S2] Driver:              4002
[S3] Firmware:            0041
[S5] dll:                 5001
[S6] Board:               CAN_PCIE402
[S7] status:              00000000
[S8] OS:                  Windows 7 Service Pack 1 (6,1,7601,2,[Service Pack 1])
[SE] Application:         CANreal
[S9] App version:         8.51
[SA] Features:            8FFA (2b Ts Sd Lo St Rf Fd )
[SB] HW Time stamps:      resolution 80.000 MHz <> 12.500 nsec (OK)
[SC] Controller:          ESDACC,80.0 MHz
[SD] Board serial:        GB000195
[C1] Net number:          15
[C2] Baud:                CAN FD
[C3] 29-bit-Ids:          1 - enabled
[C5] Single Error Diag:   1 - enabled
[T0] Start trigger:       frames_preceding(on,1) can_frame(on) id_from(513) id_to(-1) len(-1)
       29-Bit(off) rtr(off) data(--, --, --, --, --, --, --, --)
       mask(--, --, --, --, --, --, --, --)
[T1] End   trigger:       number frames(off,2) time period(off,3000) stop full(off)
       end_w/o_start(off) post_frames(on,4) post_time(off,5000) can_frame(on) id_from(513) id_to(-
       1)_len(-1) 29-Bit(off) rtr(on)
       data(--, --, --, --, --, --, --, --)
       mask(--, --, --, --, --, --, --, --)
[C4] Id-Area:                        0000-2047 ($000-$7FF)
[C6] Baud details:                   500.00-2000.00 kBit/s CAN FD,TSEG[64/16],SJW[16],
                                     DTSEG[16/4],DSJW[16] FLAGS[0 hex]
[--] ----------------------------------------------
[--] Frame_Number     TriggerAbsolute_Time  Msec    Relative_Time  Desc    CAN_Id_dec     CAN_Id_hex
       Attributes     Length d1      d2      d3      d4      d5      d6      d7      d8      Text
[--]     Num   Trig        Abs       Msec                Rel       Desc        IdDec           IdHex
       Attr    Len    d1      d2      d3      d4      d5      d6      d7      d8      Text
[B1] Stats timestamp:                  08:55:03.186.035     01.08.2017
[B2] Stats busload/max. %:                 0.00           0.00
[BK] Stats kBit/sec:                      0.000
[BL] Stats data kBit/sec:                 0.000          0.000
[B3] Stats frm/sec:                           0
[B4] Stats frames:                      4515606
[B5] Stats error frames:                    102
[B6] Stats std frm/sec Rx,Tx:                 0              0
[B7] Stats ext frm/sec Rx,Tx:                 0              0
[B8] Stats std rtr/sec Rx,Tx:                 0              0
[B9] Stats ext rtr/sec Rx,Tx:                 0              0
[BA] Stats std Rx,Tx:                    922926        1361853
[BB] Stats ext Rx,Tx:                    879649        1264597
[BC] Stats std rtr Rx,Tx:                 16495          22333
[BD] Stats ext rtr Rx,Tx:                 20617          27136
[BN] Stats FD Rx,Tx:                      88201         121042
[BE] Stats bytes Rx,Tx:                10185081       14945245
[BF] Stats bits:                      480788808
[BG] Stats fifo overruns:                     0
[BH] Stats controller overruns:               0
[BO] Stats tdc:                              13              0              0
[BI] Stats error count Rx,Tx:                 0              0
[BJ] Stats controller:                       Ok
[--] ----------------------------------------------
```

```
[F9] State:                         Pausing
[F9] State:                         Resuming

[C3] 29-bit-Ids:                    0 - disabled
[C3] 29-bit-Ids:                    1 - enabled

[C5] Single Error Diag:             0 - disabled
[C5] Single Error Diag:             1 - enabled

Frame_Number  Trigger  Absolute_Time  Msec  Relative Time  Description  CAN_Id_dec  CAN_Id_hex
Attributes  Length  d1  d2  d3  d4  d5  d6  d7  d8  Text

Num TrigAbs      MsecRel  Desc          IdDec  IdHex Attr Len d1  d2  d3  d4  d5  d6  d7  d8 Text
17  S   09:45:51 423 851  Trig. Start F  513  0201       8    00  00  00  00  00  00  00  00
18      09:45:52 204 781  Range Control  709  02C5       1    01
19      09:45:52 214 10   Pressure       710  02C6       1    01
20      09:45:53 847 1633 Range Control  709  02C5       1    01
21      09:45:53 857 10   Pressure       710  02C6       1    01
22      09:45:55 489 1632 Range Control  709  02C5       1    01
23      09:45:55 509 20   Pressure       710  02C6       1    01
24      09:45:57 141 1632 Range Control  709  02C5       1    01
25      09:45:57 152 10   Pressure       710  02C6       1    01
26  E   09:45:58 283 1132 Trig. Stop F.  513  0201  R    0
27      09:45:58 784 501  Range Control  709  02C5       1    01
28      09:45:58 794 10   Pressure       710  02C6       1    01
29      09:46:00 426 1632 Range Control  709  02C5       1    01
30  N   09:46:00 436 10   Pressure       710  02C6       1    01
```

Trigger Start Frame starts the recording of the messages. The *Trigger Stop Frame* marks the end of the recording. Defined by the *End Trigger* condition, another four messages are recorded.

INFORMATION
If an active plugin defines additional columns, these are also converted. The active plugin can differ from the plugin used during the recording. Thus the CAN data can be newly interpreted.

5.2 Example for a Log-File with *Frames preceding Trigger*

In this example for Logfile under *Start Trigger* the option *Frames preceding Trigger* has been selected. This generates a *-pretrig.csplog-file. In this file the messages are recorded, that are preceding the message which meets the trigger condition:

```
[--] ---------------------------------------------
[--] CANreal Logfile Header (created)
[--] ---------------------------------------------
[F1] Source:              C:\TEMP\CANreal logtest\log-pretrig.csplog
[F2] Split file series:    log
[F3] File serial number:   n/a
[F4] Id-Description file:  Id2Description.txt
[F5] Header type:          0
[F6] Header length:        2093
[F7] File date:           01.08.2017
[F8] File time:           08:55:03
[S1] Hardware:            1016
[S2] Driver:              4002
[S3] Firmware:            0041
[S5] dll:                 5001
[S6] Board:               CAN_PCIE402
[S7] status:               00000000
[S8] OS:                  Windows 7 Service Pack 1 (6,1,7601,2,[Service Pack 1])
[SE] Application:         CANreal
[S9] App version:         8.51
[SA] Features:            8FFA (2b Ts Sd Lo St Rf Fd )
[SB] HW Time stamps:      resolution 80.000 MHz <> 12.500 nsec (OK)
[SC] Controller:          ESDACC,80.0 MHz
[SD] Board serial:        GB000195
[C1] Net number:          15
[C2] Baud:                CAN FD
[C3] 29-bit-Ids:          1 - enabled
[C5] Single Error Diag:   1 - enabled

[T0] Start trigger:             frames_preceding(on,1) can_frame(on) id_from(513) id_to(-1)
                                len(_1) 29_Bit(off) rtr(off)
                                data(--,--, --, --, --, --, --, --)
                                mask(--, --, --, --, --, --, --, --)
[T1] End trigger:               number_frames(off,2) time_period(off,3000) stop_full(off)
                                end_w/o_start(off) post_frames(on,4) post_time(off,5000)
                                can_frame(on) id_from(513) id_to(-1) len(-1) 29-Bit(off)
                                rtr(on)data(--, --, --, --, --, --, --, --)
                                mask(--, --, --, --, --, --, --, --)
[C4] Id_Area:                   0000-2047 ($000-$7FF)
[--] ---------------------------------------------
Frame_Number   Trigger   Absolute_Time   Msec   Relative_Time   Description   CAN_Id_dec   CAN_Id_hex
Attributes  Length  d1  d2  d3  d4  d5  d6  d7  d8  Text
Num TrigAbs       Msec  Rel    Desc          IdDec  IdHex Attr   Len  d1  d2  d3  d4  d5  d6  d7  d8

                                                                                            Text
1       09:45:39  055 11496   Range Control  709    02C5       1 01                        _
2       09:45:39  055 0       Pressure       710    02C6       1 01                        _
3       09:45:40  698 1643    Range Control  709    02C5       1 01                        _
4       09:45:40  698 0       Pressure       710    02C6       1 01                        _
5       09:45:42  340 1642    Range Control  709    02C5       1 01                        _
6       09:45:42  350 10      Pressure       710    02C6       1 01                        _
7       09:45:43  983 1632    Range Control  709    02C5       1 01                        _
8       09:45:43  993 10      Pressure       710    02C6       1 01                        _
9       09:45:45  625 1633    Range Control  709    02C5       1 01                        _
10      09:45:45  635 10      Pressure       710    02C6       1 01                        _
11      09:45:47  277 1642    Range Control  709    02C5       1 01                        _
12      09:45:47  277 0       Pressure       710    02C6       1 01                        _
13      09:45:48  920 1642    Range Control  709    02C5       1 01                        _
14      09:45:48  930 10      Pressure       710    02C6       1 01                        _
15      09:45:50  562 1633    Range Control  709    02C5       1 01                        _
16      09:45:50  572 10      Pressure       710    02C6       1 01                        _
17  S   09:45:51  423 851     Trig. Start F  513    0201       8 00   00  00  00  00  00  00  00 _
```

5.3 Description of the Header Rows [F1]...[C5]

CANreal Logfile Header (appended)		This header and the following CAN messages are appended to the logfile.	
Typ:	Example:	Description:	
[F1] Source:	`\log.csplog`	Name of the binary logfile, that has been converted.	
[F2] Split file series:	log	File name	
[F3] File serial number:	n/a	Serial number in a split-file sequence. No split-file sequence = n/a	
[F4] Id-Description file:	`Id2Description.txt`	The file containing the description text of the identifier, defined via *Text description mapping for CAN identifiers* (page 28)	
[F5] Header type:	0	(program-internal data set number)	
[F6] Header length:	2093	(program-internal data set length)	
[F7] File date:	01.08.2017	Date of the converted binary logfile.	
[F8] File time:	08:55:03	Time of the converted binary logfile.	
[F9]		*Pause / Resume Buttons*	
[S1] Hardware:	1016	Hardware version of the CAN device (Information of the CAN driver)	
[S2] Driver:	4002	CAN driver version (Information of the CAN driver)	
[S3] Firmware:	0041	Firmware version of the CAN device (Information of the CAN driver)	
[S5] dll:	5001	Version of the NTCAN.DLL (Information of the CAN driver)	
[S6] Board:	CAN_PCIE402	Board version of the CAN device (Information of the CAN driver)	
[S7] status:	00000000	Status of the CAN device (Information of the CAN driver)	
[S8] OS:	Windows XXXX - Service Pack X (xxxxx, [Service Pack X])	Windows operating system version (Information of the operating system)	
[SE] Application:	CANreal	Name of application	
[S9] App version:	X.XX	CANreal program version	
[SA] Features:	0072 (2b Ts Lo)	As described under *Help / About* (see chapter “3.6.2 About”)	
[SB] HW-Timestamp:	Resolution. ... (OK)	HW-timestamp supported and selected	
[SC] Controller:	ESDACC, 80,0 MHz	Type of the CAN controller	
[SD] Board serial:	YYXXXXXX	Serial number	
[C1] Net number:	0	CAN net number	
[C2] Baud:	500	CAN baud rate or CAN FD, see Baud details [C6]	
[C3] 29-bit-Ids:		29-bit-IDs:	
		0=	enabled
		1=	disabled

Examples

[T0] Start trigger:	None		Dialogue options to Start Trigger in written form.
[T1] End trigger:	None		Dialogue options to End Trigger in written form.
[C4] Id-Area:	2000-2047 ($000-$7FF)		Active CAN Id-area
[C5]: Single Error Diag			Corresponds to Single-Error-Diagnostic (enabled/disabled)
[C6] Baud details:	500.00-2000.00 kBit/s CAN FD,TSEG[64/16],SJW[16], DTSEG[16/4],DSJW[16] FLAGS[0 hex]		Baud rate kbit/s +/- deviation (inaccuracy) Table-Index see esd baud rate table ([1] chapter: *“canSetBaudrate”*), TSEG, SJW, FLAGS (see [1], chapter: “NTCAN_BITRATE”)
[B1] Begin of stats	08:55:03:186.053 01.08.2017		Time of “*Reset statistics*“ [hh:mm:ss]
[B2] Bus load	0.00	0.00	Percentaged bus load “Kbit/second“ in relation to the bit rate of the CAN net.
[B2] kBit/Second	0.00	0.00	Total bit/s see below
[B3] Frames/Second	0.00		CAN messages per second. Specified in more detail with *Frame Rate.*
[B4] Total Frames	4515606		Absolute counter for all CAN messages. Specified in more detail with *Number Frames.*
[B5] Error Frames	102		Counter for all faulty CAN messages detected by the CAN controller.
[B6] Standard	0	0	Number of standard Rx- and Tx-frames per second
[B7] Extended	0	0	Number of extended Rx- and Tx-frames per second
[B8] Standard RTR	0	0	Number of standard RTR Rx- and Tx-frames per second
[B9] Extended RTR	0	0	Number of extended RTR Rx- and Tx-frames per second
[BA] Standard	922926		Number of standard Rx- and Tx-frames
[BB] Extended	879649		Number of extended Rx- and Tx-frames
[BC] Standard RTR	16495		Number of standard RTR Rx- and Tx-frames
[BD] Extended RTR	20617		Number of extended RTR Rx- and Tx-frames
[BN] FD Rx, Tx	88201	121042	Number of CAN FD frames
[BE]*Data Bytes*	10185081	14945245	Number of transmitted or received data bytes
[BF]*Total Bits*	480788808		Counts all bits on the CAN bus (see NTCAN Part 1: Application Developers Manual [1], chapter: “*NTCAN-BUS-STATISTIC”*)
[BG]*Driver FIFO Overruns*	0		See NTCAN Part 1: Application Developers Manual [1], chapter: “EV_CAN_ERROR”
[BH]*Controller overruns*	0		
[BO] *tdc*	13	0	Transmitter delay compensation (see NTCAN Part 1: Application Developers Manual [1], chapter: “Transmitter Delay Compensation (TDC)”)
[BI]Error Count	0		Register of CAN controller
[BJ]Status	OK		Contains the bus state of the CAN controller (see NTCAN Part 1: Application Developers Manual [1], chapter: “*NTCAN_CTRL_STATE”*).

5.4 Example for a Logfile with *Lost frames*

Lost-frames can result if CAN messages are received faster than CANreal can read them from the FIFO-memory buffer of the driver. The CAN driver ignores some messages. These messages are indicated as “lost” behind the frame number.
The number of *Lost-Frames* lies in the range of 1..255. 255 means 255 or more. After a “lost” message the counter starts again from 0.

```
590585              11:57:41   143 =0   Frame_1_1920_CanId  1920   0780   3  71   04   41   q_A
590586 - 89 lost    11:57:41   143 =0   Frame_1_1920_CanId  1920   0780   3  71   04   9B   q_›
590587              11:57:41   153 10   Frame_1_1920_CanId  1920   0780   3  71   04   9C   q_œ
...
616053              11:57:43   176 =0   Frame_1_1920_CanId  1920   0780   3  71   04   16   q__
616054 - 255 lost   11:57:43   176 =0   Frame_1_1920_CanId  1920   0780   3  71   04   F9   q_ù
616055              11:57:43   186 10   Frame_1_1920_CanId  1920   0780   3  71   04   FA   q_ú
...
618125              11:57:43   347 =0   Frame_1_1920_CanId  1920   0780   3  71   04   10   q__
618126 - 95 lost    11:57:43   347 =0   Frame_1_1920_CanId  1920   0780   3  71   04   70   q_p
618127              11:57:43   357 10   Frame_1_1920_CanId  1920   0780   3  71   04   71   q_q
```

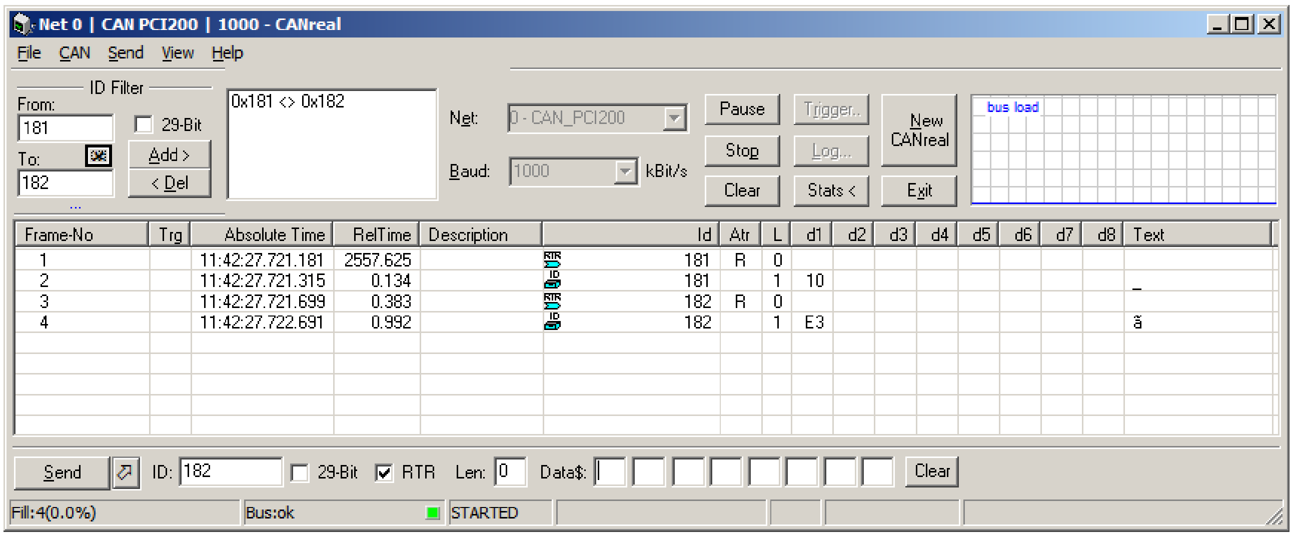
To avoid *Lost-Frames*:

- Close all other applications on your PC
- Settings under *Advanced settings* / *List View* in CANreal:
  - Disable display of the *Grid lines*
  - Hide optional columns
- Move CANreal into the foreground of the screen.
- Usage of a CAN device that relieves the Host-CPU (e.g. CAN board with integrated CPU and preprocessing of the data).
- Usage of a faster processor from 1,5 GHz or Multi-Core allow 100% bus load with 0-byte-CAN messages at 1000 kbit/s. (reference is made from “worst-case” which should normally not occur.)

NOTICE
The following step is only for experienced users!

- Improvement of the application priority via the Windows Task-Manager.

5.5 Example to Request and Reception of Messages



Net 0 | CAN PCI200 | 1000 - CANreal

From: 181
To: 182
0x181 <> 0x182
Net: 0 - CAN_PCI200
Baud: 1000 kBit/s

Frame-No	Trg	Absolute Time	RelTime	Description	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
1		11:42:27.721.181	2557.625	RTR	181	R	0									
2		11:42:27.721.315	0.134	ID	181		1	10								_
3		11:42:27.721.699	0.383	RTR	182	R	0									
4		11:42:27.722.691	0.992	ID	182		1	E3								ã

ID: 182 ☐ 29-Bit ☑ RTR Len: 0 Data$:

Fill:4(0.0%) Bus:ok 🟩 STARTED

Figure 77: Example to request and reception of data

In this example data of two digital CANopen I/O-modules (according to CiA 301) are requested and received. The data are requested via transmission of an RTR-message. The CAN identifiers of the Tx-PDOs result from:

CAN-Identifier :	0x180 + Node-ID
	CAN identifier for module number 1 (Node-ID = 1) results in 0x181.
	CAN identifier for module number 2 (Node-ID = 2) results in 0x182.

INFORMATION
The example is only functional if a CAN node with the appropriate baud rate is connected and the CAN node can respond to an remote-request on the used identifiers!

Proceeding:

1. Modify the values of the baud rate and net number if necessary. In the example the following values are chosen:
   Baud rate: 1000 kbit/s
   Net: 0
2. Click the button *<Del* to delete the list of the CAN-Identifier to be displayed. Enter ‘181’ and ‘182’ (without preceding 0x) in the fields *Add/Delete ID Area from* and *to* and confirm the entries by clicking the button *Add>*.
3. Start the process by clicking the button *Start*. Received messages with the identifiers 0x0181 or 0x0182 are displayed when they are received.
4. Enter ‘181' (without preceding 0x) in the *ID* input filed of the input bar for send messages and enable the *RTR*-checkbox.
5. After clicking the button *Send* the CAN messages are displayed, i.e. the RTR message to request the data (*Frame-No.* 1) and the message containing the requested data (Frame-No. 2), in the message window.

6. Repeat the steps 5. and 6. accordingly for the identifier ‘182'. *Frame-No.* 3 and 4 will contain data transmitted with identifier 0x182.
7. If the decimal representation is enabled, all relevant values in the input fields and in the display windows are updated and the ID’s are indicated in decimal format.

5.6 Example for using IRIG-B

INFORMATION
Please note that the IRIG-B functionality can only be used if the hardware is equipped with an IRIG-B interface!
The following esd products support the IRIG-B functionality:
- CPCI-CAN/400-4-I-P 4xCAN,IRIG-B, PXI (esd order No.: C.2033.01)
- CAN-USB/400-IRIG-B (esd order No.: C.2069.04)
- PMC-CAN/400-4 4x CAN 1x IRIG-B (esd order No.: C.2047.01)

Before using any applications with IRIG-B, the hardware should be initialized, for example with the program `irigbtest.exe.`

`irigbtest.exe` should run on windows start-up or login after driver start (batch file).

```
irigbtest.exe  {esd-Net-No} {wait}  {time-source} {year-setting}
```

esd-Net-No :	Number of esd CAN interface (“0” for first card, first interface)
wait :	Wait time for time sync (seconds), 0 = don’t wait, only configure and exit
time-source :	0 = analog, 1 = digital input
year-setting:	0 = no year (current year)
	1 = year from IRIG-B
	-- If this flag is set, the year information (which is an optional IRIG-B extension) will be used.

The **card default** after card reboot (if `irigbtest.exe` is not called) is: analog input + no year

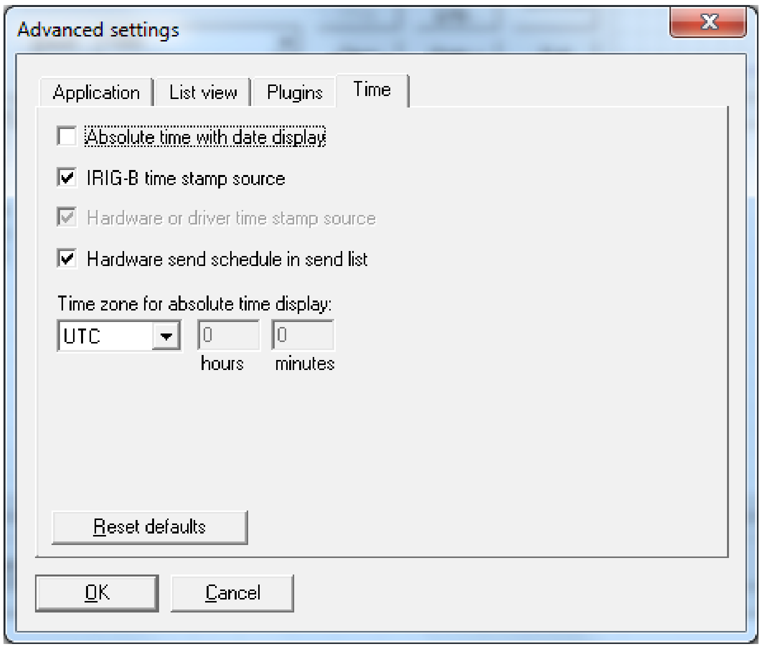
Example:

```
irigbtest.exe 0 0 1 1       # interface No. 0, no wait, digital input, with year information
```

5.6.1.1 Configure Time Settings

Click on the menu item *File* in the CANreal program window and then *Advanced settings* in the drop-down menu. Activate now the tab *Time* in the window *Advanced Settings*.

Please note that the settings cannot be changed in the CANreal “started” mode.



Advanced settings

Application | List view | Plugins | Time

☐ Absolute time with date display
☑ IRIG-B time stamp source
☑ Hardware or driver time stamp source
☑ Hardware send schedule in send list

Time zone for absolute time display:
UTC | 0 hours | 0 minutes

Reset defaults

OK | Cancel

Figure 78: *Advanced Settings* window/ *Time*

- Enable the checkbox *Absolute time with date display* to display the date together with the absolute time in the display window for the received messages. See example 1. below.
- Activate the checkbox *IRIG-B time stamp source*
  (The IRIG-B external time stamp source is enabled per default. If the checkbox is disabled, the internal Windows time is used.)
- In the drop-down menu of *Time zone for absolute time display* you can now choose the time zone:
  - *Local time* (default setting, PC time),
  - *UTC*,
  - *UTC+*,
  - *UTC-*

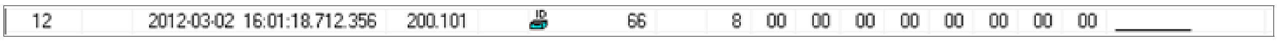
  With *UTC* the CANreal time display equals the IRIG-B time. Other time settings add or subtract an offset to the IRIG-B time for time display.

Examples:

Example 1: UTC with date (*Absolute time with date display* is enabled), the **IRIG-B time is “16:01”**

☑ Absolute time with date display

The *Absolute Time* of the received messages is displayed with date as UTC time:

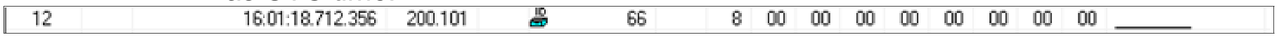


```
12    2012-03-02 16:01:18.712.356    200.101        66    8  00  00  00  00  00  00  00  00
```

Examples

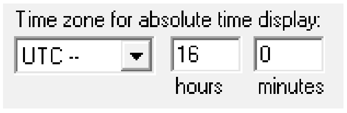
Example 2: UTC without date (*Absolute time with date display* is disabled), the **IRIG-B time is "16:01"**

The *Absolute Time* of the received messages is displayed as UTC time:



```
12    16:01:18.712.356   200.101      66   8  00  00  00  00  00  00  00  00
```

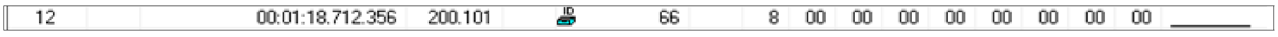
Example 3: UTC -- 16 hours, the IRIG-B time is "16:01"



Time zone for absolute time display:

UTC -- | 16 hours | 0 minutes

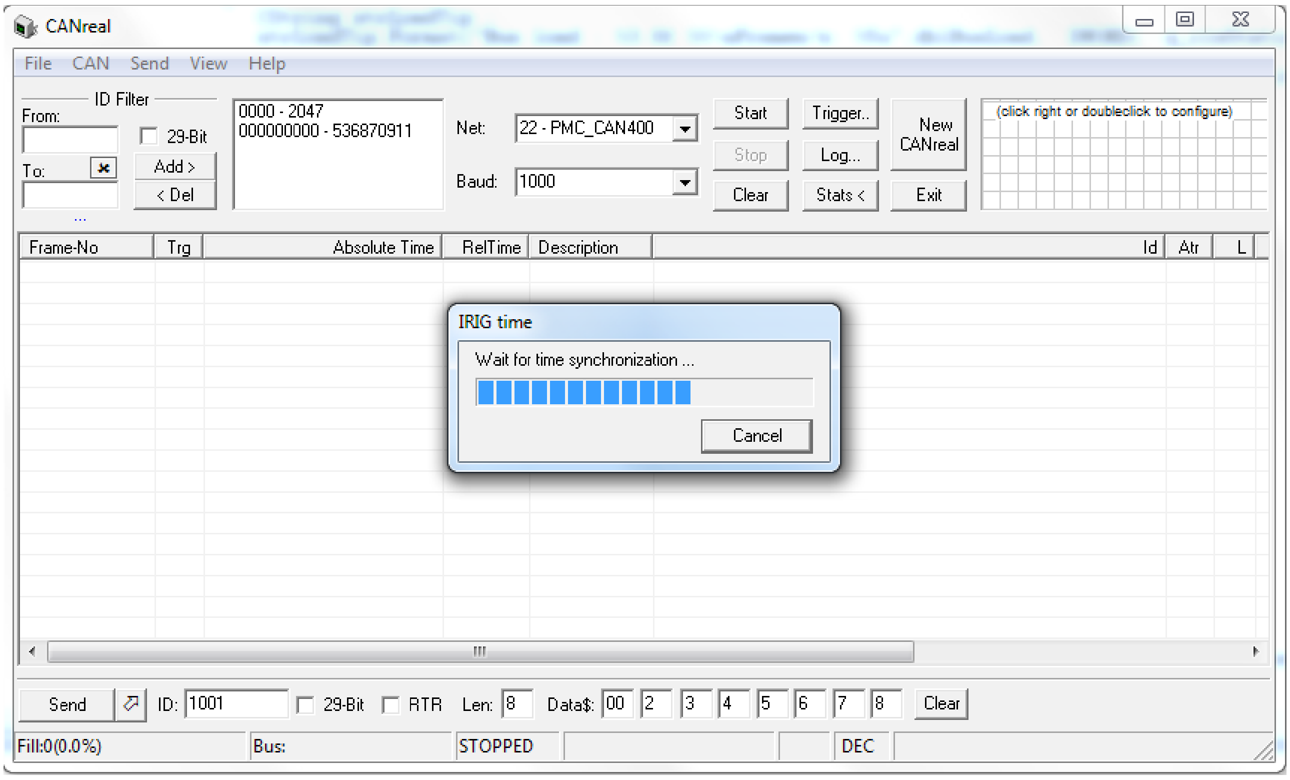
The *Absolute Time* of the received messages is 00:01:18



```
12    00:01:18.712.356   200.101      66   8  00  00  00  00  00  00  00  00
```

Save the settings in the program window via menu item *File* and *Store current settings (see Figure 17 on page 59).*

Now you can start your CANreal program by clicking the button Start in the program window.



0000 - 2047
000000000 - 536870911

Net: 22 - PMC_CAN400
Baud: 1000

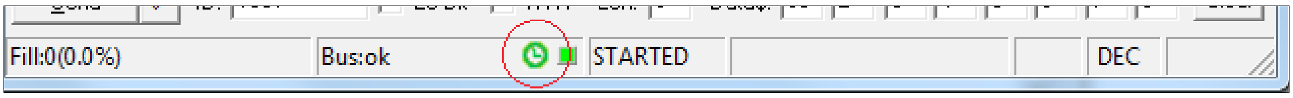
IRIG time
Wait for time synchronization ...

ID: 1001 29-Bit RTR Len: 8 Data$: 00 2 3 4 5 6 7 8

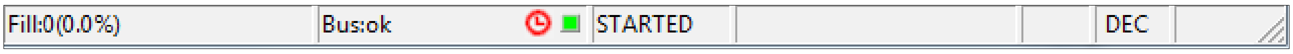
Fill:0(0.0%) Bus: STOPPED DEC

Figure 79: *Time synchronization*

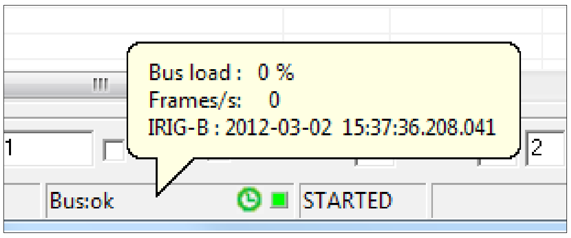
When the time synchronization is finished, the status bar displays a clock, which shows the time sync state:



Clock	**Time sync state**
Green clock :	OK
Yellow clock:	bad synchronization (current time deviates from the expected time by more than 256 ms
Red clock:	no sync, no signal
No clock at all:	The hardware or the windows driver packet does not support IRIG-B.



A tool tip on the status line displays the Bus load, the number of frames and the current IRIG-B date and time (see figure 80).



Bus load : 0 %
Frames/s: 0
IRIG-B : 2012-03-02 15:37:36.208.041

Figure 80: *Tool tip*

6. Development of Plugins for CANreal

(CANreal 4.86 and higher)

The list view and the conversion of recorded binary logfiles into text files can be upgraded via plugins. A plugin defines additional columns and provides display and tooltip texts. Further functions are timeouts for CAN messages in the static list view and triggering.

6.1 General

INFORMATION
Demo project for Visual Studio ≥ 2008 („vc90") available. Please contact our sales team (mailto:sales@esd.eu).

Plugins for CANreal are dynamic Windows libraries (DLL), that export a number of functions for the call via CANreal (callback functions). The calling standard is "WINAPI" (also known as "PASCAL").
CANreal imports via `GetProcAddress` per name, ordinal number without meaning, without underscore, without function-"decoration", without "@"-stack frame.
For this the functions have to be specified as "`__stdcall`" for Microsoft development environments and a "DEF"-file has to be generated.

In the C-header of a C++-source file "C" has to be specified externally.

Sample for C/C++-header:

```
extern "C" { int __stdcall CANRealPluginXYZ(int arg) ; }
```

Sample for DEF-file:

```
;LIBRARY CANrealPluginDev

EXPORTS
  CANRealPluginXYZ
```

INFORMATION
An example of a plugin is contained on the CAN-SDK-CD.

Please read chapter "Plugins" on page 29 for information about the selection of plugins.

6.2 Plugin Functionalities

Overview of the Functions with Type Definitions:

`typedef long          int32_t ;`	Integer 32 bit
`typedef int32_t       INT_BOOL ;`	boolean value 0 or 1 as integer
`typedef void*         HWND_HANDLE ;`	*window handle* with data type "HWND" by Windows

```
INT_BOOL CANRealPluginFnkVersion(PPluginVersion) ;
```

```
INT_BOOL CANRealPluginFnkInit(PPluginInfo) ;
```

```
void      CANRealPluginFnkNotify
             (int32_t dwNotify, int32_t scroll, int32_t *pdwRetFeatures) ;

void*     CANRealPluginFnkGetContext(int32_t id) ;
```

```
INT_BOOL    CANRealPluginFnkGetDisplay
             (PPluginArgs, int32_t *pRetNumCols, PPluginColumnData[]) ;
```

```
void      CANRealPluginFnkAnalyze(PPluginArgs) ;

int32_t   CANRealPluginFnkDoSetupDlg
             (HWND_HANDLE parent, PDlgCoords, int32_t *nReloadSize, void **) ;

void      CANRealPluginFnkReloadSettings(void *pReloadSettingsData) ;

void      CANRealPluginFnkLoadSettings(const char* szINI, const char* szKey) ;

void      CANRealPluginFnkSaveSettings(const char* szINI, const char* szKey) ;

int32_t   CANRealPluginFnkEditSendlistDlg(PPluginEditSendlist) ;
```

The stucture types `PPluginVersion`, `PPluginInfo`, `PPluginArgs`, `PPluginColumnData`, `PDlgCoords` and `PpluginEditSendlist` are described from chapter “6.6 Transfer Structures” on. `CANRealPluginFnkVersion` and `CANRealPluginFnkInit` have to be exported in every case, `CANRealPluginFnkGetDisplay` in most cases.

6.3 Sequence Chart

```
<-- CANreal starts -->
   <-- detection of plugins and gathering information -->
     |
[1]  Windows DllMain: dll-load/attach ...
     |
     CANRealPluginFnkNotify(loaded)
     |
     CANRealPluginFnkVersion()
     |
     |
    <+>.... CANRealPluginFnkNotify(unload)
            |
            Windows DllMain: DLLUnload, Detach ...
            |
            --> [1]    (multiple load/unload before actually starting possible)
     |
     |
  <-- initialization of plugin -->

     CANRealPluginFnkLoadSettings(INI-file)
     |
[2]  CANRealPluginFnkInit()
     |
     CANRealPluginFnkGetContext(online,file,...)
     |

   <-- plugin doing it's work -->

[3]  CANRealPluginFnkNotify(start,stop,pause,clear,trigger..)
     |
     +.....CANRealPluginFnkAnalyze [context*)]
     |
     | CANRealPluginFnkGetDisplay [context*)] (list, convert, reload, save_frames... )
     |
     CANRealPluginFnkNotify(start,stop,pause,clear,trigger..)
     |
     |
   <-- user opens setup dialog -->
    <+> CANRealPluginFnkDoSetupDlg(reloadsettings)
                         |
                         |   (plugin has option to be reloaded in case of
                              structural setup change, e.g. number or name of columns)
                         |
           <<<           <-- if reload -->
                         |
                         <CANRealPluginFnkNotify(stop)>
                         |
                       -- unload --
                         |
                         CANRealPluginFnkNotify(unload)
                         |
                         Windows DllMain: DLLUnload, Detach ...
                         |
                         Windows DllMain: dll-load/attach ...
                         |
                       --  (re)load --
                         |
                         CANRealPluginFnkNotify(loaded)
                         |
                         CANRealPluginFnkVersion()
                         |
                         CANRealPluginFnkReloadSettings(reloadsettings)
                         |
     |                   --> [2]
           >>>
     ...
   <-- user saves CANreal settings -->
     <+> CANRealPluginFnkSaveSettings(INI-file)
     |
   <-- user edits a send list plugin column -->
     <+> CANRealPluginFnkEditSendlistDlg
     |
     ---> [3]
     ...
     |
 <-- CANreal exit -->
```

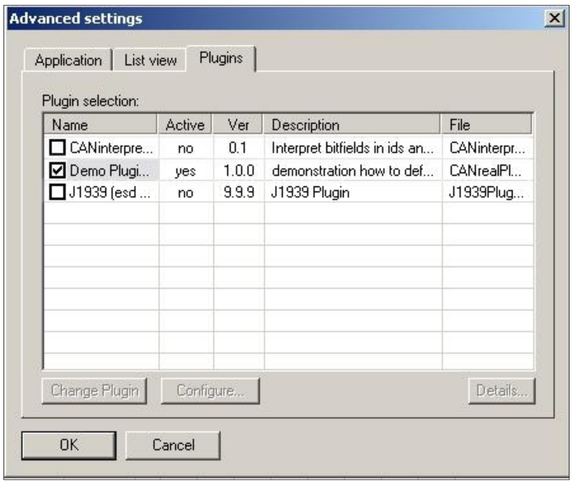
```
     |
     <CANRealPluginFnkNotify(stop)>
     |
     CANRealPluginFnkNotify(unload)
     |
     CANRealPluginFnkNotify(exit)
     |
     Windows DllMain: DLLUnload, Detach ...
```

6.4 Description of the Plugin Functionalities

`CANRealPluginFnkNotify` can be used alternatively to Windows DLLMain function and gives information to the plugin (besides further messages) about loading, unloading and program end.

The first call after loading the DLL is `CANRealPluginFnkNotify("loaded")`. Before the unloading the call `CANRealPluginFnkNotify("unload")` follows and just before the CANreal program end `CANRealPluginFnkNotify("exit")`.

Via `CANRealPluginFnkVersion` CANreal identifies version-, author- and description texts for the listing of the available plugins in the *Advanced settings / Plugins* dialogue field:



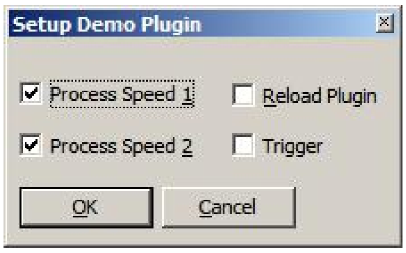
Plugin selection:

Name	Active	Ver	Description	File
☐ CANinterpre...	no	0.1	Interpret bitfields in ids an...	CANinterpr...
☑ Demo Plugi...	yes	1.0.0	demonstration how to def...	CANrealPl...
☐ J1939 (esd ...	no	9.9.9	J1939 Plugin	J1939Plug...

Figure 81: *Advanced settings / Plugins*

(Up to the final usage of the plugin the DLL can be loaded and unloaded repeatedly for listing and debugging purposes. Therefore no long initialisation should be carried out by `CANRealPluginFnkNotify` and `CANRealPluginFnkVersion`!)

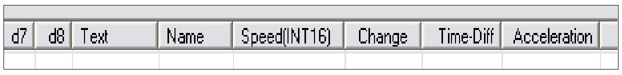
A click on the *Configure* button leads to the call of `CANRealPluginFnkDoSetupDlg`.



☑ Process Speed 1
☐ Reload Plugin
☑ Process Speed 2
☐ Trigger

Figure 82: Setup dialogue of the demo plugin

It will be continued with `CANRealPluginFnkLoadSettings`, which allows the loading of the configuration settings contained in the CANreal profile file (*.cspini).
Then `CANRealPluginFnkInit` sends structural properties of the plugin, in particular a description of the additional columns, that extend the list view:

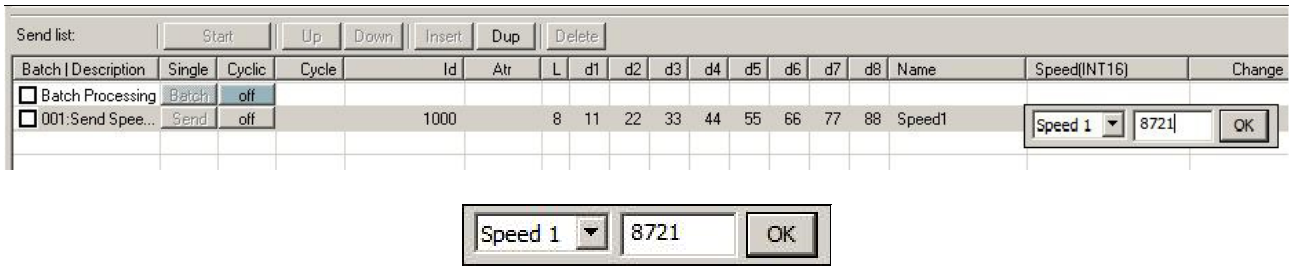


d7	d8	Text	Name	Speed(INT16)	Change	Time-Diff	Acceleration

Figure 83: Additional columns in the list view

If you save the CANreal configuration via *File / Store current settings* or *File → Load or save profile settings*, the plugin is requested via `CANRealPluginFnkSaveSettings` to store it's own configuration in the CANreal profile file (`*.cspini`). (That differs for `CANRealPluginFnkLoadSettings`: The plugin will be always unloaded before a call of `CANRealPluginFnkLoadSettings`.)

To edit one of the columns of the *Send list* belonging to the plugin, `CANRealPluginFnkEditSendlistDlg` is called (optional). Inside the function a dialogue should be opened, that enables the user to edit the entry of the *Send list* in a special "Plugin suitable" way.



Batch | Description	Single	Cyclic	Cycle	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Name	Speed(INT16)	Change
☐ Batch Processing	Batch	off															
☐ 001:Send Spee...	Send	off		1000		8	11	22	33	44	55	66	77	88	Speed1	Speed 1 ▾ 8721 OK	

Figure 84: Additional columns in the list view

`CANRealPluginFnkAnalyze` is called for every (received or transmitted) CAN message. With this older CAN messages can be analysed. The plugin can store the result of the analysis application specific in a “private memory”, attached to the last transferred CAN message. The private memory (`TpluginArgs:private_memory`) can be reserved via `CANRealPluginFnkInit` and stored in a ring buffer in the CANreal list view (with CAN data and timestamp).

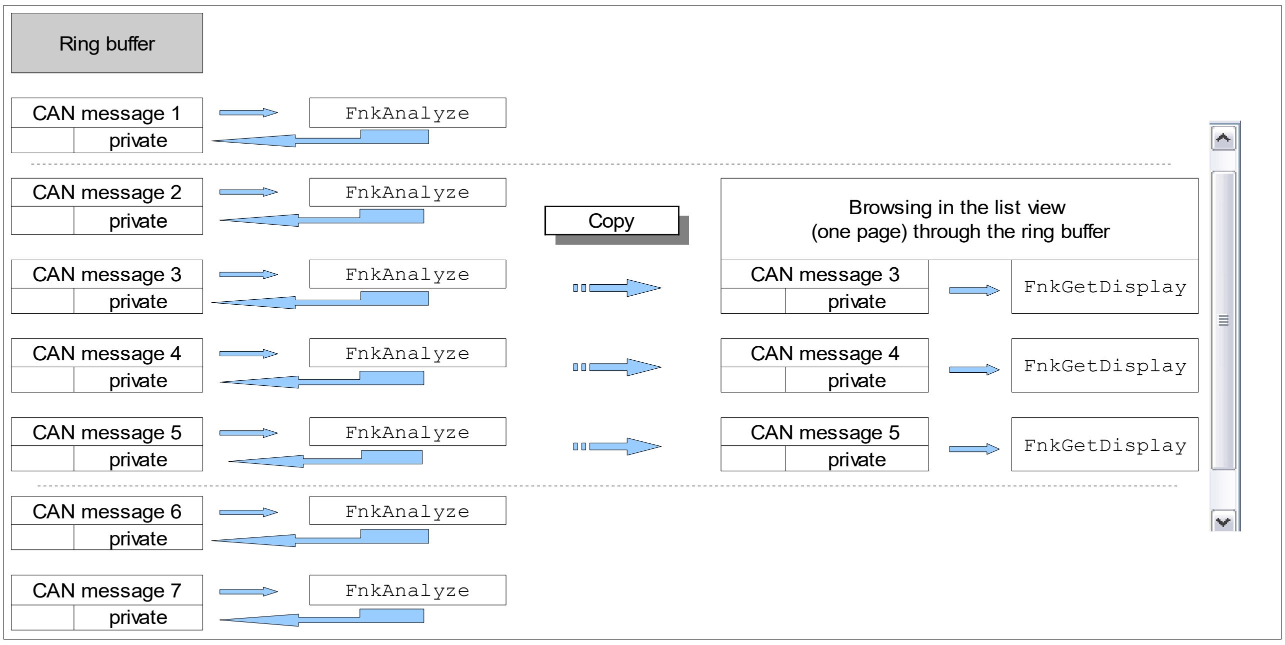


Figure 85: CANreal list view and ring buffer with private data

Note to `TpluginArgs:private_memory`:

- The private memory is limited to 128 byte per CAN message.
- The function `CANRealPluginFnkGetDisplay` receives a copy of the memory for displaying.That is the reason why the pointer `TpluginArgs:private_memory` differs at the calls of both functions.
- Do not create direct references to dynamically allocated memory areas under `TpluginArgs:private_memory` , because CANreal does not give a message if old entries of the ring buffer are deleted.

To display column texts or a tooltipp if the mouse cursor points to an entry in the list, CANreal calls `CANRealPluginFnkGetDisplay`. Furthermore `CANRealPluginFnkGetDisplay` provides additional text columns for the conversion of binary logfiles in ASCII-files. Additionally icons can be shown in the list view:

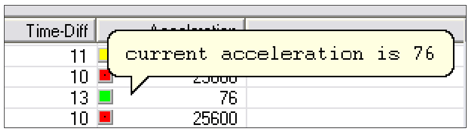


Figure 86: Tooltipp and Plugin defined icons

CANreal repeatedly calls the function within the list view for the same CAN messages (e.g. for scrolling forwards or backwards in the list) in any (message) order.
For a correct display of the list the plugin must be able to generate always the same display texts for an adequate range (review). A problem with the review might occur for the developer if the private memory is used and its capacity limited by CANreal is not sufficient. The plugin architecture might then provide e.g. a special ring buffer for defined display values, which is referenced per index or hash from the private memory.

A review range "> 1" (one) message is available in the mode scroll view and for "Open logfiles" ("Logfile viewer" = Offline list).

For static view only the last displayed message can be restored (review: one (1) message) .

At file conversion always `CANRealPluginFnkAnalyze` and `CANRealPluginFnkGetDisplay` are called alternately (review: none / (0) messages).

`CANRealPluginFnkNotify` informs about the size of the review range provided by CANreal, that should ideally completely cover the plugin.

6.5 Context Model

CANreal provides different displays and functions associated with CAN messages, which are processed quasi simultaneously and partly in different operating system threads:

- List display: Scroll view, static view, offline list (*Open logfiles*)
- Log file conversion (*Convert logfiles to text*)
- Convert during the recording (*On-the-Fly ASCII Convert*)
- *Send list*

In connection with `CANRealPluginFnkAnalyze` and `CANRealPluginFnkGetDisplay` it is required to define various display/analysis contexts.

Example:	Call the menu item *File / Convert Logfiles to text*. The recorded CAN messages have nothing in common with the messages just displayed in the list and have to be analysed separately. A plugin, that should support the parallel functionalities of CANreal, must implement the context model of CANreal - otherwise the CAN messages of a context which is not supported have to be ignored by `CANRealPluginFnkAnalyze` and no text is returned at `CANRealPluginFnkGetDisplay`!

6.5.1 Defined Analysis Contexts

Context	Function
`PLG_CONTEXT_ONLINE_LIST`	List view
`PLG_CONTEXT_FILE_CONVERT`	*Convert logfiles to text*
`PLG_CONTEXT_SENDLIST`	*Send list* (no *Analysis*)
`PLG_CONTEXT_FLY_CONVERT`	On-The-Fly ASCII Convert *)

*) Note: `CANRealPluginFnkGetDisplay` runs at `PLG_CONTEXT_FLY_CONVERT` in another thread than the application-main thread. Therefore a separate return memory for the display texts has to be provided!

Table 4: Listing of defined ASCII analysis context

6.5.2 Distribution of Calls to different Threads

The calls of `CANRealPluginFnkAnalyze` and `CANRealPluginFnkGetDisplay` are distributed to different Threads.

Context	..FnkAnalyze	..FnkGetDisplay
`PLG_CONTEXT_ONLINE_LIST`	c	m
`PLG_CONTEXT_FILE_CONVERT`	m	m
`PLG_CONTEXT_SENDLIST`	- **)	m
`PLG_CONTEXT_FLY_CONVERT`	f	f *)

**) `CANRealPluginFnkAnalyze` has not been called.

6.5.3 Threads

m	Application-main thread
c	CanRead thread, CANreal ring buffer
f	On-the-fly thread

`CANRealPluginFnkGetContext` requests user-specific context values/pointers from the plugin, which are transmitted to `CANRealPluginFnkAnalyze` and `CANRealPluginFnkGetDisplay` depending on the current context. If `CANRealPluginFnkGetContext` has not been exported, the constants „`PLG_CONTEXT_...`" are transmitted.

6.5.4 Limitations

`#define PLUG_MAX_NUMBER_COLS 40`	Maximum column number
`#define PLUG_MAX_PRIVATE_MEMORY_SIZE 128`	Maximum size of the private memory in byte
`#define MAX_PLUG_ICONS 1000`	Maximum number of icons
`#define PLG_MAX_COLUMN_SIZE 1000`	Maximum column dimensions for list view in pixel
`#define PLG_MAX_COLUMN_FIELDWIDTH 60`	Maximum field width in text files

6.5.5 Type Definitions of the Transfer Structures

`typedef unsigned char uint8_t ;`	Integer 8 bit unsigned
`typedef char int8_t ;`	Integer 8 bit
`typedef unsigned short uint16_t ;`	Integer 16 bit unsigned
`typedef short int16_t ;`	Integer 16 bit
`typedef unsigned long uint32_t ;`	Integer 32 bit unsigned
`typedef long int32_t ;`	Integer 32 bit
`typedef void* ICON_HANDLE ;`	Windows icon handle "HICON"

6.6 Transfer Structures

6.6.1 Data Structure of the CAN Message (CMSG)

CAN message with ID, length and data (see NTCAN Part 1: Application Developers Manual [1], chapter: “CMSG”).

Definition:

```
{
  int32_t       id ;
  unsigned char len ;
  unsigned char msg_lost  ;
  unsigned char reserved[2] ;
  unsigned char data[8] ;
} CMSG ;
```

Fields:

id	CAN-ID
	The higher bits of the CAN-ID are specially coded:
	`#define NTCAN_20B_BASE (0x20000000)`
	for 29-bit CAN identifier
	
	`#define NTCAN_EV_BASE (0x20000000)`
	for "Controller"-events of the NTCAN library (see [1])

len	Length of the CAN message (0..8)
	The highest bit of the length is specially coded:
	`#define NTCAN_RTR (0x10)`
	for RTR-frames

6.6.2 PluginAbsTime

Absolute timestamp of a CAN message.

Definition:

```
typedef struct STPluginAbsTime
{
    uint32_t  time ;
    uint16_t  millitm ;
    uint16_t  microseconds ;
    uint16_t  flag ;
    uint16_t  reserved ;
} TPluginAbsTime,
 *PPluginAbsTime ;
```

Fields:

time	POSIX time: Seconds from midnight of 1st January 1970, coordinated universal time (UTC)
millitm	Fractions of seconds in milliseconds
microseconds	Fractions of milliseconds in microseconds
flag	`#define PLG_ABS_MICRO (0x8000)` flag& `PLG_ABS_MICRO` !=0 → the field microseconds is assigned (driver or hardware timestamp, otherwise not assigned at software timestamp).

6.6.3 PluginFrameTime

Absolute and relative timestamp, index counter of a CAN message.

Definition:

```
typedef struct STPluginFrameTime
{
    TPluginAbsTime tAbs ;
    uint32_t       ulRel ;
    uint32_t       ulCurrentIndex ;
} TPluginFrameTime,
 *PPluginFrameTime ;
```

Fields:

tAbs	Absolute time of the CAN message
ulRel	Relative time since preceding CAN message in .. .. milliseconds : ... (software timestamp on application level) .. microseconds : ... (driver- or hardware timestamp)
ulCurrentIndex	Free running CAN message counter

6.6.4 Directions of Input / Output Parameters

IN	Input parameter: Initialised from CANreal in the plugin before the call
OUT	Output parameter: Initialised from the plugin as return value to CANreal

6.6.5 PluginVersion

This structure contains information about the plugin version, used in function(s) :
`CANRealPluginFnkVersion`

```
/* ------------------------------------------------------------------------*/
/* struct STPluginVersion; */
typedef struct STPluginVersion
{
  uint32_t      nStructSize ;            /* [IN]  */  /* equals sizeof(struct) */

  char         *pszPlgName ;             /* [OUT] */  /* name for the plugin, e.g. "esd demo plugin"   */
  char         *pszPlgPublisher ;        /* [OUT] */  /* company or organization name, "my company"    */
  char         *pszPlgVersion ;          /* [OUT] */  /* version string like "1.0.0"                   */
  char         *pszPlgDescription ;      /* [OUT] */  /* descriptive text                              */

  uint32_t      interfaceVersionPlugin ; /* [OUT] */  /* version of interface implemented by plugin,
                                                         currently unused, set to zero                 */

  uint32_t      featuresPlugin[4] ;      /* [OUT] */  /* features requested by plugin
                                                         currently only [0]|PLG_F0_PLUGIN_CONSOLE      */

  char         *pszCANRealVersion ;      /* [IN] */   /* CANreal version string                        */

  uint32_t      interfaceVersionCANreal ;/* [IN] */   /* version of interface implemented by CANreal
                                                         currently unused, set to zero                 */

  uint32_t      featuresCANreal[4] ;     /* [IN] */   /* features offered by CANreal, currently zero   */

  uint32_t      reserved[16] ;

} TPluginVersion, *PPluginVersion ;
```

6.6.6 PluginColumnInfo

This structure contains information about the column settings, used in function(s) :
`CANRealPluginFnkInit`

```
/* ------------------------------------------------------------------------*/
/* struct STPluginColumnInfo, STPluginInfo; */
typedef struct STPluginColumnInfo
{
  uint32_t      nStructSize ;   /* [OUT]  */ /* initialize with sizeof(struct), else CANreal will ignore */

  char         *pszColumnTitle ;         /* [OUT]  */ /* list view column title                      */
  char         *pszColumnShort  ;        /* [OUT]  */ /* alternative short column title for log files  */
  uint32_t      defaultColumnSize ;      /* [OUT]  */ /* list view default column with in (pixels)     */
  uint32_t      logFieldWidth ;          /* [OUT]  */ /* field with in characters for log files        */
  uint32_t      align ;                  /* [OUT]  */ /* list view column align PLG_ALIGN_xxxx         */

  uint32_t      reserved[5] ;

} TPluginColumnInfo, *PPluginColumnInfo ;
```

align

Name	Value	Meaning
`PLG_ALIGN_LEFT`	0	Left-justified text
`PLG_ALIGN_CENTER`	1	Centred text
`PLG_ALIGN_RIGHT`	2	Right-justified text

defaultColumnSize

Limit:	`#define PLG_MAX_COLUMN_SIZE (1000)`

logFieldWidth

Limit:	`#define PLG_MAX_COLUMN_FIELDWIDTH (60)`

6.6.7 PluginInfo

This structure is used in `CANRealPluginFnkInit`, contains column descriptions and icons.

```
typedef struct STPluginInfo
{
  uint32_t      nStructSize ;            /* [IN]  */  /* equals sizeof(struct)                       */

  uint32_t      dwInfFlags ;             /* [OUT] */  /* PLG_INFO_xxxx                               */

  /* request additional CAN frame marker memory  -> PluginArgs */
  uint32_t      private_memory_size ;    /* [OUT] */  /* -> TPluginArgs:private_memory*
                                                         important note: Keep it minimized !
                                                         memory will be allocted for each line in
                                                         the online list view
                                                         limit ist PLUG_MAX_PRIVATE_MEMORY_SIZE      */

  uint32_t      nNumberColumns ;         /* [OUT] */  /* number of listview columns                  */
  PPluginColumnInfo column_info ;        /* [OUT] */  /* provide an array of column descriptions     */

  uint32_t      reserved1 ;

  /* icon pool for list view columns */
  uint32_t      nNumberIcons ;           /* [OUT] */  /* number of icons in following icons-array    */
  ICON_HANDLE   *icons ;                 /* [OUT] */  /* provide an array icons "HICON",
                                                         preferred size 12x12, 16 color              */

  uint32_t      reserved[16] ;

} TPluginInfo, *PPluginInfo ;
```

6.6.8 PluginArgs

This structure `STPluginArgs` is used in functions : `CANRealPluginFnkGetDisplay`, `CANRealPluginFnkAnalyze`.

```
/* /* -------------------------------------------------------------------------*/
/* struct STPluginArgs; */
typedef struct STPluginArgs
{
  uint32_t      nStructSize ;       /* [IN] */   /* equals sizeof(struct)                         */

                                    /* ANALYZE    DISPLAY */
  void          *context ;          /* [IN]        [IN]    */
                                    /* context from CANRealPluginFnkGetContext or PLG_CONTEXT_XXXX value */


  uint32_t      dwPlgFlags ;        /* [IN+OUT]    [IN]    */  /* PLG_ROLE_XXXX, PLG_CLEAR_XXXX,
                                                                    PLG_TRIGGER_XXXX, PLG_DISP_XXXX */
  CMSG         *pcmsg ;             /* [IN+(OUT)]  [IN]    */  /* CAN frame id and data  */
  TPluginFrameTime
               *pTimestamp ;        /* [IN]        [IN]    */  /* CAN frame time of pcmsg */

                                    /* additional CAN frame marker memory "private_memory"
                                       - requested by CANRealPluginFnkInit
                                       - filled by CANRealPluginFnkAnalyze
                                       - displayed with CANRealPluginFnkGetDisplay
                         --> CANRealPluginFnkGetDisplay always receives a _copy_ of the "private_memory"
                                filled within CANRealPluginFnkAnalyze !
                         --> do not use malloc from CANRealPluginFnkAnalyze, CANreal does not tell you when
                                to free  */
  void         *private_memory ;        /* [IN+OUT]    [IN]    */  /* zero if not required or not provided */
  uint32_t      private_memory_size ;    /* [IN]       [IN]    */  /* for safety */
  uint32_t      staticTimeoutMsec ;      /* [OUT]      ./.     */  /* ---> CANRealPluginFnkAnalyze only
                                                                       ---> list view static mode only:
                                                                        next time period in milliseconds
                                                                        relative to *pTimestamp
                                                                        when this pcmsg->id shall be marked as
                                                                        timed out (red line in online list)
                                                                        0=default, not timeout  */

  uint32_t      uReserved ;
  uint32_t      reserved[16] ;
  //<<__

} TPluginArgs, *PPluginArgs ;
```

6.6.9 PluginColumnData

This structure contains the dataset, used in function(s) : `CANRealPluginFnkGetDisplay`

```
/* -------------------------------------------------------------------------*/
/* STPluginColumnData; */
typedef struct STPluginColumnData
{
  uint32_t nStructSize ;                /* [OUT]  */ /* initialize with sizeof(struct), else CANreal will
                                                        ignore */

  char * pszText ;                      /* [OUT] */  /* return text     (PLG_DISP_LISTTEXT) */
  char * pszToolText ;                  /* [OUT] */  /* return tip text (PLG_DISP_TIPTEXT ) */

  uint32_t reserved1 ;
  uint32_t reserved2 ;

  /* icon */
  uint32_t iconIndex ;                  /* [OUT] */  /* (1-based index within TPluginInfo:icons)
                                                         0=no Icon, 1..TPluginInfo:nNumberIcons  */
  uint32_t reserved[5] ;

} TPluginColumnData, *PPluginColumnData ;
```

6.6.10 DlgCoords

This structure contains information about coordinates and is used in:
`CANRealPluginFnkEditSendlistDlg`

```
/* ----------------------------------------------------------------------------*/
/* struct STDlgCoords, STPluginEditSendlist; used in function(s) : CANRealPluginFnkEditSendlistDlg */
typedef struct STDlgCoords
{
  uint32_t nStructSize ;                 /* [IN]  */  /* equals sizeof(struct)  */

  /* screen coordinates of the CAN real send list */
  int32_t xParent ;                      /* [IN] */  /* x */
  int32_t yParent ;                      /* [IN] */  /* y */
  int32_t wParent ;                      /* [IN] */  /* width  */
  int32_t hParent ;                      /* [IN] */  /* height */

  /* display hint for dialog positioning _relative_ to (xParent, yParent) */
  int32_t xHint   ;                      /* [IN] */  /* x of activated send list column      */
  int32_t yHint   ;                      /* [IN] */  /* y of activated send list row         */
  int32_t wHint   ;                      /* [IN] */  /* width of activated send list column  */
  int32_t hHint   ;                      /* [IN] */  /* height of activated send list row    */

  int32_t reserved[5] ;

} TDlgCoords, *PDlgCoords ;
```

6.6.11 PluginEditSendlist

This plugin can show an adequate dialogue, if the function `CANRealPluginFnkEditSendlistDlg` is called by CANreal, see page 116.

```
typedef struct STPluginEditSendlist
{
    uint32_t     nStructSize ;        /* [IN] */  /* equals sizeof(struct) */

    HWND_HANDLE  parent ;             /* [IN] */  /* "HWND" of send list as dialog parent window */
    TDlgCoords   coords ;             /* [IN] */  /* screen coordinates of send list with active row,
                                                 column     */

    uint32_t     dwEditFlags ;        /* [IN] */  /* PLG_EDIT_SHOW_DIALOG */

    int32_t      nRow ;               /* [IN] */  /* active send list row,    -1 = invalid  */
    int32_t      nColumn ;            /* [IN] */  /* active send list column, -1 = invalid  */

 char        *szText ;                /* [IN] */  /* currently displayed cell text (from
                                                     CANRealPluginFnkGetDisplay
                                                        No text return here!
                                                        New text is requested from
                                                        CANRealPluginFnkGetDisplay       */

    CMSG        *pcmsg ;              /* [IN+OUT] */ /* CAN frame id and data, return dialog changed
                                                    values      */
    char        *szDescription ;      /* [IN+OUT] */ /* text for description column */
    int32_t      cycleTime ;          /* [IN+OUT] */ /* cycle time in milliseconds for cycle column */

    void        *reserved1 ;
    uint32_t     reserved ;

} TPluginEditSendlist, *PPluginEditSendlist ;
```

6.7 Export Functions

6.7.1 CANRealPluginFnkVersion

Detection of plugins, versions and compatibility testing of the plugin interface and request of defined functionalities. `CANRealPluginFnkVersion` and `CANRealPluginFnkInit` must be exported.

Syntax:

```
INT_BOOL CANRealPluginFnkVersion
(
PPluginVersion pPluginVersion
) ;
```

Parameter:

pPluginVersion	(See structure "`PluginVersion`") `pszPlgName`, `pszPlgPublisher` and `pszPlgVersion` must be filled in.
PluginVersion.featuresPlugin	`featuresPlugin[0]` can be assigned with `#definePLG_F0_PLUGIN_CONSOLE(0x00000001` **`(OUT)`** for debugging purposes. CANreal generates a Windows console *(AllocConsole())* then and redirects the standard output to it. Therefore the plugin is reloaded. Switch off the bit in the *Release* version of the plugin!

Return value:

Name	Value	Meaning
`TRUE`	1	Successful execution
`FALSE`	0	No successful execution, CANreal does not use the plugin.

6.7.2 CANRealPluginFnkInit

Initialization of the display structure of the plugin. Return of column descriptions and icons. Reservation of the “private_memory” for the *Analysis* function. `CANRealPluginFnkVersion` and `CANRealPluginFnkInit` must be exported.

Syntax:

```
INT_BOOL CANRealPluginFnkInit
(
PPluginInfo pPluginInfo
) ;
```

Parameter:

pPluginVersion	See structure *PluginInfo*	
pPluginInfo.nNumberColumns	Number of columns defined by the plugin.	
	Limit:	`#define PLUG_MAX_NUMBER_COLS (40)`
pPluginInfo.column_info	Return of a field (*Array*) with column descriptions (see *PluginColumnInfo*)	

pPluginInfo.private_memory_size	Size of the *private_memory.* `#define PLUG_MAX_PRIVATE_MEMORY_SIZE (128)`	
	Limit:	`#define PLUG_MAX_PRIVATE_MEMORY_SIZE (128)`
pPluginInfo.nNumberIcons	Number of Icons in *pPluginInfo.icons.*	
	Limit:	`#define MAX_PLUG_ICONS (1000)`
pPluginInfo.icons	Field (*Array*) of icon handles for *predefined* icons. The field index, based on “1”, is specified in `PluginColumnData.iconIndex` for the display of an icon.	

PluginArgs.dwPlgFlags

Name	Bit value	Meaning
`PLG_INFO_DISABLE_SETUP_DLG *)` **(OUT)**	`0x00000002`	Block call of `CANRealPluginFnkDoSetupDlg`
`PLG_INFO_DISABLE_EDIT *)` **(OUT)**	`0x00000010`	Block call of `CANRealPluginFnk EditSendlistDlg`
`PLG_INFO_DISABLE_ANALYZE *)` **(OUT)**	`0x00000020`	Block call of `CANRealPluginFnkAnalyze`

*) can be combined with all other bit values

Return value:

Name	Value	Meaning
`TRUE`	1	Successful execution
`FALSE`	0	No successful execution, CANreal does not use the plugin.

6.7.3 CANRealPluginFnkNotify

Informs about loading/unloading the plugin, change of program status and interactions. The call of the function is made via the main thread (“m”) of CANreal. To set trigger the plugin has to export the function and to set `PLG_TRIGGER_SET_START/PLG_TRIGGER_SET_STOPT` .
If the function has been exported `PLG_NOTIFY_FEATURE_ANALYZE` has to be returned to release `CANRealPluginFnkAnalyze`!

Syntax:

```
void CANRealPluginFnkNotify
(
int32_t dwNotify,
int32_t scroll,
int32_t *pdwRetFeatures
) ;
```

Parameter:

dwNotify

Name	Bit value	Meaning
`PLG_NOTIFY_LOADED`	0x20000000	Plugin has been loaded
`PLG_NOTIFY_UNLOAD`	0	Unloading plugin
`PLG_NOTIFY_EXIT`	0	CANreal is terminated, call follows after `PLG_NOTIFY_UNLOAD`
STOP	**0 (Null)**	Stops list view
`PLG_NOTIFY_START`	0	Starts list view
`PLG_NOTIFY_PAUSE`	0	Pause list view
`PLG_NOTIFY_CLEAR`	0	Delete list view
`PLG_NOTIFY_TRIGGER *)`	0	Start via trigger function, valid together with `PLG_NOTIFY_START` *) only.

*) can be combined with all other bit values

scroll

Maximum review range for `CANRealPluginFnkGetDisplay`

Value	Meaning
>0	Review range of the list view *scroll view*
0	Static list view, review range one, the last current message

pdwRetFeatures

Return of release options, valid together with PLG_NOTIFY_START only.

Name	Bit value	Meaning
`PLG_NOTIFY_FEATURE_ANALYZE *)` **(OUT)**	0x00000001	Enable call of `CANRealPluginFnkAnalyze`.

PLG_NOTIFY_FEATURE_STARTTR *) **(OUT)**	0x00000002	The plugin enables *Start-Trigger* (--> CANRealPluginFnkAnalyze :PluginArgs.dwPlgFlags PLG_TRIGGER_SET_START) valid together with PLG_NOTIFY_TRIGGER only
PLG_NOTIFY_FEATURE_STOPTTR *) **(OUT)**	0x00000004	The plugin enables *Stopp-Trigger* (--> CANRealPluginFnkAnalyze :PluginArgs.dwPlgFlags PLG_TRIGGER_SET_STOPT) valid together with PLG_NOTIFY_TRIGGER only

*) combinable bit values

6.7.4 CANRealPluginFnkGetContext

Application-specific context for the calls of `CANRealPluginFnkAnalyze` and the return of `CANRealPluginFnkGetDisplay`.

Syntax:

```
void* CANRealPluginFnkGetContext
(
int32_t idContext
) ;
```

Parameter:

idContext

Name	Bit value	Meaning
PLG_CONTEXT_ONLINE_LIST	0	Context of the Online-*list view* or *Open Logfiles*
PLG_CONTEXT_FLY_CONVERT	1	Context for *On-The-Fly-Convert*
PLG_CONTEXT_FILE_CONVERT	2	Context for the conversion of logfiles
PLG_CONTEXT_SENDLIST	3	Context of the *Send list*
Return value	Application-specific pointer	

6.7.5 CANRealPluginFnkGetDisplay

With `CANRealPluginFnkGetDisplay` the display data of the CANreal list (text, tooltipps and icons) and text for the conversion of logfiles can be returned. `CANRealPluginFnkGetDisplay` must be exported if the plugin defines the display columns. (A plugin only used for analysis does not need this function.)

Syntax:

```
INT_BOOL CANRealPluginFnkGetDisplay
(
PPluginArgs pPluginArgs,
int32_t *pRetNumCols,
PPluginColumnData ppPluginColumnData[]
) ;
```

Parameter:

PluginArgs.context

The pointer determined via `CANRealPluginFnkGetContext` or one of the constants:

Name	Bit value	Meaning
`PLG_CONTEXT_ONLINE_LIST`	0	Context of the Online-*list view* or *Open Logfiles*
`PLG_CONTEXT_FLY_CONVERT`	1	Context for *On-The-Fly-Convert*
`PLG_CONTEXT_FILE_CONVERT`	2	Context for conversion of logfiles
`PLG_CONTEXT_SENDLIST`	3	Context of the *Send list*

PluginArgs.dwPlgFlags

`PLG_ROLE_SCROLL_VIEW`	`0`	Online *scroll view*, size of the *private_memory*.
`PLG_ROLE_STATIC_VIEW`	`0x00000002`	Online list in static view
`PLG_ROLE_FILE_CONVERT`	`0x00000004`	File conversion
`PLG_ROLE_SAVE_FRAMES`	`0x00000008`	File conversion, save list as file
`PLG_ROLE_SEND_LIST`	`0x00000010`	*Send list*
`PLG_ROLE_RELOAD_FRAMES`	`0x00000020`	Restore messages (*Open Logfiles*)
`PLG_DISP_LISTTEXT` `*)`	`0`	Request display text
`PLG_DISP_TIPTEXT` `*)`	`0x00200000`	Request tooltip text

*) can be combined with all other bit values

PluginArgs.pcmsg	CAN message: ID and data
PluginArgs.pTimestamp	Timestamp of the CAN message
PluginArgs.private_memory/private_memory_size	Copy of `CANRealPluginFnkAnalyze` (s. next chapter)
pretNumCols	Number of columns, for which displayed data shall be returned.
ppPluginColumnData	(See Structure *PluginColumnData*) return a pointer, that points to a field (*ARRAY*) of *PluginColumnData* structures. Each field index describes a display column. After return out of the function the content of the field must not be changed. (Please note *On-The-Fly-Context* in the own thread!)
pPluginArgs	(See structure *PluginArgs*)

Return value:

Name	**Value**	**Meaning**
`TRUE`	1	Returned columns are displayed
`FALSE`	0	No display, ignore *ppPluginColumnData*

6.7.6 CANRealPluginFnkAnalyze

According to the executed context, every transmitted/received or recorded CAN message is transferred to the function for analysis.

Syntax:

```
void CANRealPluginFnkAnalyze
(
PPluginArgs pPluginArgs
) ;
```

Parameter:

PluginArgs.context

The pointer determined via `CANRealPluginFnkGetContext` or a constant:

Name	Bit value	Meaning
`PLG_CONTEXT_ONLINE_LIST`	0	Context of the Online-*list view* or *Open Logfiles*
`PLG_CONTEXT_FLY_CONVERT`	1	Context for *On-The-Fly-Convert*
`PLG_CONTEXT_FILE_CONVERT`	2	Context for conversion of logfiles

PluginArgs.dwPlgFlags

`PLG_ROLE_SCROLL_VIEW`	0	Online *scroll view*
`PLG_ROLE_STATIC_VIEW`	0x00000002	Online list in static view
`PLG_ROLE_FILE_CONVERT`	0x00000004	File conversion
`PLG_ROLE_SAVE_FRAMES`	0x00000008	File conversion, save list to file
`PLG_ROLE_RELOAD_FRAMES`	0x00000020	Reload messages (*Open Logfiles*)
`PLG_CLEAR_CONTEXT *)`	0	With this call the context can be newly set up. Saved counters, histories, etc. are reset and deleted.
`PLG_TRIGGER_SET_START *)` **`(OUT)`**	0	Set *Start Trigger* (only valid with CANRealPluginFnkNotify:dwNotifyFlags PLG_NOTIFY_FEATURE_STARTTR)
`PLG_TRIGGER_SET_STOPT *)` **`(OUT)`**	0x00020000	Set *Stopp Trigger* (only valid with CANRealPluginFnkNotify:dwNotifyFlags PLG_NOTIFY_FEATURE_STOPTTR)

*) can be combined with all other bit values

PluginArgs.pcmsg	CAN message: ID and data
PluginArgs.pTimestamp	Timestamp of the CAN message
PluginArgs.private_memory/private_memory_size	Memory to store the result of the analysis, a copy will later be transferred to `CANRealPluginFnkGetDisplay`. Reserved size via `CANRealPluginFnkInit`.
PluginArgs.staticTimeoutMsec	In the context of the static list view only: Timeout for the next reception of this CAN-ID.

6.7.7 CANRealPluginFnkDoSetupDlg

Is a request to the plugin to show the user a modal configuration dialogue.

Syntax:

```
int32_t CANRealPluginFnkDoSetupDlg
(
HWND_HANDLE parent,
PDlgCoords pDlgCoords,
int32_t *nReloadSize,
void ** ppReloadSettings
) ;
```

Parameter:

parent	The window handle to which the dialogue shall be displayed as sub window.
pDlgCoords	(See structure *DlgCoords)* DlgCoords.*Parent: window coordinates of the "parent" window DlgCoords.*Hint are *not* valid (=0)
nReloadSize	Size of the structure returned in *ppReloadSettings*.
ppReloadSettings	Is set to an address of a structure, which can be read by `CANRealPluginFnkReloadSettings`.

Return value:

Name	Value	Meaning
`PLG_SETUP_RETURN_OK`	0	OK, successful
`PLG_SETUP_RELOAD`	1	Successful, unload and reload plugin
`PLG_SETUP_RETURN_ERROR`	-1	Error occurred

6.7.8 CANRealPluginFnkReloadSettings

Load plugin settings from a memory structure.

Syntax:

```
void CANRealPluginFnkReloadSettings
(
void *pReloadSettingsData
) ;
```

Parameter:

pReloadSettingsData	Pointer to a data structure freely defined by the developer, that has been returned before as a copy via `CANRealPluginFnkDoSetupDlg`(ppReloadSettings).

6.7.9 CANRealPluginFnkLoadSettings

Read plugin settings from the CANreal profile file. The file is an INI-file. Corresponds to the call of `CANRealPluginFnkSaveSettings`.

Syntax:

```
void CANRealPluginFnkLoadSettings
(
const char* szINI,
const char* szKey
) ;
```

Parameter:

pReloadSettingsData	Pointer to a data structure freely defined by the developer, that has been returned before as a copy via `CANRealPluginFnkDoSetupDlg`(ppReloadSettings).
szINI	Full path and name of the INI-file.
szKey	Main key (INI-section), under which the data are stored. The key has the form: `[DLL::plugin-name::publisher::dll-file-name]`

6.7.10 CANRealPluginFnkSaveSettings

Store the plugin settings into the CANreal profile file. The file is an INI-file.

Syntax:

```
void CANRealPluginFnkSaveSettings
(
const char* szINI,
const char* szKey
) ;
```

Parameter:

szINI	Full path and name of the INI-file.
szKey	Main key (INI-section), under which the data are stored. The key has the form: `[DLL::plugin-name::publisher::dll-file-name]`

6.7.11 CANRealPluginFnkEditSendlistDlg

This function is called by CANreal if the user makes a click on entries of the *Send list*. The plugin can show an adequate dialogue. After return from the function the data in the transmitted *PluginEditSendlist* structure must not be changed by the plugin. Therefore the dialogue has to be carried out modal to the window of the *Send list*. If the dialogue shall not be displayed, PLG_EDIT_RETURN_CANCEL must be returned.

Syntax:

```
int32_t CANRealPluginFnkEditSendlistDlg
         (
           PPluginEditSendlist pPluginEditSendlist
         ) ;
```

Parameter:

pPluginEditSendlist

(see structure "PluginEditSendlist") *PluginEditSendlist.dwEditFlags*

Name	Bit value	Meaning
PLG_EDIT_SHOW_DIALOG	1	Request to display the dialogue
The CAN data, the description and the entry *cycle* can be set. At the call the following information are transferred: • row and column of the selected cell of the *Send list* • current text of the cell • the position of the control element of the *Send list* in coordinates of the screen (coords.*Parent) • relative to it the position of the selected cell (coords.*Hint)		

Return value:

Name	Value	Meaning
PLG_EDIT_RETURN_OK	0	Change *Send list*
PLG_EDIT_RETURN_CANCEL	1	Do not change the *Send list*.

7. Troubleshooting

7.1 Troubleshooting at Program Call

Problem	Proposed solution
The program CANreal can not be started (e.g. after a crash of the computer)	→ Restart your computer and start CANreal.
CANreal can not be started after a restart of the computer	→ Reinstall CAN SDK on your computer, CANreal will be installed automatically (see chapter „Program Call“, page 8)
CANreal can still not be started after reinstallation	→ CANreal saves the window arrangement and the profile data in the AppData directory: `C:\Users\<Username>\AppData\Roaming\CANreal.ini` and in the registry: `HKEY_CURRENT_USER\Software\esd\CANreal` In case of problems with the program start, you can delete the entries.

7.2 Troubleshooting in CAN Wiring

Problem	Proposed solution
Errors in CAN wiring	→ Read and follow the wiring notes in the hardware manual of the esd CAN module used. Chapter: “Correct Wiring of Electrically Isolated CAN Networks“ → If necessary also read chapter „CAN Bus Troubleshooting Guide“ in the hardware manual of the esd CAN module used. The “CAN-Bus Troubleshooting Guide” is a guide to find and eliminate the most frequent hardware-error causes in the wiring of CAN networks.

8. References

[1] NTCAN Part 1: Application Developers Manual Rev.5.3, Order No.: C.2001.21, 2019-07-25, esd electronics gmbh

9. Order Information

CANreal is contained in the scope of delivery of the CAN-SDK and can be downloaded from our website www.esd.eu.

PDF Manuals
Manuals are available in English and usually in German as well. For availability of the manuals see table below.
Please download the manuals as PDF documents from our esd website www.esd.eu for free.

Manuals		**Order No.**
CANreal-ME	Software manual in English	C.1107.21
CANreal-MD	Software manual in German	C.1107.20

Table 5: Available manuals

Printed Manuals
If you need a printout of the manual additionally, please contact our sales team: sales@esd.eu for a quotation. Printed manuals may be ordered for a fee.