

 **CAN** *alyzer*

 **DEN** *alyzer*

.CAN.LIN.MOST.FlexRay

Manual

Version 5.1

Vector Informatik GmbH, Ingersheimer Str. 24, D-70499 Stuttgart

Tel. +49 711 80670-0, Fax +49 711 80670 111

Email can@vector-informatik.de, Internet <http://www.vector-informatik.de>

Subsidiaries

France

Vector France SAS

168, Boulevard Camélinat
F-92240 Malakoff

Tel.: +33 1 4231 4000

Fax: +33 1 4231 4009

information@vector-france.com

<http://www.vector-france.com>

Japan

Vector Japan Co., Ltd.

Seafort Square Center Bld. 18F
2-3-12, Higashi-shinagawa, Shinagawa-ku

J-140-0002 Tokyo

Tel.: +81 3 5769 6970

Fax: +81 3 5769 6975

info@vector-japan.co.jp

<http://www.vector-japan.co.jp>

Sweden

VecScan AB

Lindholmspiren 5
SE-41756 Göteborg

Tel.: +46 31 76476 00

Fax: +46 31 76476 19

info@vecscan.com

<http://www.vecscan.com/>

USA

Vector CANtech, Inc.

Suite 550
39500 Orchard Hill Place
USA-Nov, Mi 48375

Tel.: +1 248 449 9290

Fax: +1 248 449 9704

info@vector-cantech.com

<http://www.vector-cantech.com>

For Distributor Addresses please have a look on our website:

www.vector-informatik.com

International Quality Standard Certificate

The Quality/Process Management of Vector Informatik GmbH is being certified according to DIN EN ISO 9001:2000-12 (formerly DIN EN ISO 9001:1994-08) throughout since 1998-08-19.

Typographic Conventions

Note:	Identifies important notes
•	Identifies enumerations (bullet items)
➔ '1.0 Introduction'	Identifies references to further chapters of this manual
[OK]	Notation for buttons in dialogs
<TAB>	Notation for keys on the computer keyboard
<Ctrl>+<Z>	Notation for keys of the computer keyboard which should be pressed simultaneously
Add... File File open...	Notation for menu, command and dialog names
on message 0x100	Notation for MS-DOS syntax or program code

Notes on the naming convention



The multi-bus functionality and the modular configuration concept of the program variants require a new naming convention of several Vector products.

Included bus options are now indicated with a "." (DOT) followed by the name of the bus system.

Examples:

for the LIN option: **.LIN**

for the MOST option: **.MOST**

The products **CANoe** resp. **CANalyzer** always contain the CAN option; therefore ".CAN" is never listed as a bus option. All further contained options are specified as shown above.

The products **DENoe** resp. **DENalyzer** aim at users that exclusively use one or some of the bus systems LIN, MOST or FlexRay; here the CAN option is **not** included.

Examples for CANoe:

- CANoe (tool for **CAN** users)
- CANoe.LIN (tool for **CAN** and **LIN** users)
- CANoe.LIN.MOST (tool for **CAN**, **LIN** and **MOST** users)

Examples for DENoe: (DEN: Distributed Embedded Network)

- DENoe.LIN (tool for **LIN** users)
- DENoe.LIN.MOST (tool for **LIN** and **MOST** users)
- DENoe.LIN.MOST.FlexRay (tool for **LIN**, **MOST** and **FlexRay** users)

Additional Notes

- Practice parts (CANoe tour / CANalyzer tour) are currently only available for CANoe resp. CANalyzer (not for the bus options LIN, MOST, FlexRay).
- In the manual and online help basically the terms CANoe and CANalyzer are used.
- The terms DENoe resp. DENalyzer are used in manual and online help to show differences to CANoe resp. CANalyzer.

Contents

1	Introduction.....	1
1.1	Overview	1
1.2	Tips for Using CANalyzer	3
1.2.1	Menus	3
1.2.2	Dialogs	3
1.2.3	Control of the Measurement Setup	5
1.2.4	The Help System.....	6
1.3	CANalyzer Tour	6
1.3.1	Preparations	7
1.3.2	Setting Up the CAN Bus.....	8
1.3.3	Transmitting Data	10
1.3.4	Evaluation Windows	13
1.3.5	Working with Symbolic Data.....	15
1.3.6	Analysis of Signal Values in the Data Window	17
1.3.7	Analysis of Signal Responses in the Graphics Window	19
1.3.8	Use of the Database in Transmitting Messages	20
1.3.9	Analysis of an Engine Area Simulation.....	20
1.3.10	Logging a Measurement.....	21
1.3.11	Evaluating a Log File.....	23
1.3.12	Tips for Solving Your Own Tasks	23
1.4	Overview of the Programs.....	24
1.5	CANalyzer Architecture	25
1.6	Particularities of the Demo Version	25
2	Applications	27
2.1	Measurement/Measurement Setup	30
2.1.1	Measurement Start.....	30
2.1.2	Working with Configurations.....	31
2.1.3	Representation Formats.....	32
2.2	Transmitting and Receiving of Data	33
2.2.1	The Transmit Branch	33
2.2.2	The Evaluation Branches	34
2.2.3	Message Attributes.....	35
2.3	Use of Databases.....	36

2.3.1	Creating and Modifying Databases	37
2.3.2	Access to Database Information.....	38
2.3.3	Associating the Database.....	38
2.3.4	Use of Multiple Databases	40
2.3.5	Resolving Ambiguities	40
2.3.6	Checking for Consistency of Symbolic Data.....	41
2.4	Working with Multiple Channels	41
2.4.1	Channel Definition	41
2.4.2	Channels in Online Mode	42
2.4.3	Channels in Offline Mode	42
2.5	CANalyzer in Load and Overload Operation	43
2.5.1	Behavior in Load Situations.....	43
2.5.2	Behavior with Data Loss.....	43
2.5.3	Fill Level Indicator	44
2.5.4	Optimizing Performance.....	44
2.5.5	Configuration Options at High Bus Load	45
2.6	Logging and Evaluation of Measurement Files	46
2.6.1	Logging Trigger	46
2.6.1.1	Trigger Mode.....	47
2.6.1.2	Trigger Condition.....	48
2.6.1.3	Set of user defined conditions.....	48
2.6.1.4	Trigger Events.....	50
2.6.1.5	Time Window	51
2.6.1.6	Configuration of the Logging Buffer	51
2.6.2	Log Files.....	52
2.6.3	Event Types in Log Files	54
2.6.4	Data Analysis in Offline Mode.....	55
2.6.4.1	Flow Control in Offline Mode.....	56
2.6.4.2	Configuration of Online and Offline Modes	57
2.6.5	Exporting and Converting Log Files	58
2.6.5.1	Export	58
2.6.5.2	Conversion.....	58
2.6.6	CANlog support.....	58
2.7	COM-Server	58
2.8	Troubleshooting.....	59
2.9	List of Error Messages to the CAN Interface	60
2.10	Interface to the Hardware.....	63
2.10.1	Configuring the Hardware	64

2.10.2	Programming the Bus Parameters	65
2.10.3	Acceptance Filtering	68
2.10.4	Card and Driver Options	69
3	Windows	70
3.1	Desktop Concept	70
3.2	Window Management	70
3.2.1	MDI window	71
3.2.2	Docking window	71
3.2.3	Floating window	71
3.3	Measurement Setup Window	71
3.3.1	Data Flow in the Measurement Setup	71
3.3.2	Configuration of the Measurement Setup	72
3.3.3	Working with Evaluation Blocks in the Measurement Setup	73
3.4	Trace Window	74
3.4.1	Standard Configuration of the Trace Window	75
3.4.2	Configuration of the Columns in the Trace Window	77
3.4.3	Trace Window Options from the Toolbar	78
3.4.4	Trace Watch Functionality and Trace Watch Window	79
3.4.5	Optimizing the Trace Window	79
3.5	Graphic Window	79
3.5.1	Selecting Signals	80
3.5.2	Arrangement of Signals	81
3.5.3	Signal Layout	81
3.5.3.1	Line Types	82
3.5.3.2	Display Modes	83
3.5.4	Configuration of the Measurement	83
3.5.5	Measurement and Display Functions	84
3.5.6	Signal Modes	84
3.5.7	Measurement Modes	84
3.5.8	Display Modes	85
3.5.9	Layout Functions	85
3.5.10	Export of Signals	87
3.5.11	Toolbar of the Graphics Window	88
3.5.12	Optimization of the Graphics Window	88
3.6	Write Window	90
3.7	The Data Window	90
3.7.1	Configuration of Signals	91

3.7.2	Display Types	92
3.7.3	Activity Indicator	94
3.7.4	Peak Indicator	94
3.7.5	Optimization of Data Display	94
3.8	Statistics Window	95
3.8.1	Direct Display in the Statistics Window.....	95
3.8.2	Statistics Report	96
3.8.3	Choosing a Histogram.....	97
3.9	Bus Statistics Window	98
4	Blocks and Filter.....	100
4.1	Generator Block	101
4.1.1	Configuration of Triggering	102
4.1.2	Configuration of Transmit List.....	102
4.1.3	Entry of Signal Values	103
4.1.4	Entry of Mode-Dependent Signals.....	103
4.1.5	Function Generator for the Transmit List	104
4.2	Interactive Generator Block (IG).....	106
4.2.1	Configuring the Interactive Generator Block.....	106
4.2.1.1	Transmit List	107
4.2.1.2	Value Generator.....	107
4.2.1.3	Trigger Condition.....	108
4.2.1.4	Generating a High-Load Situation.....	108
4.2.1.5	Entering Signal Values	108
4.2.1.6	Entering Mode-Dependent Signals	109
4.2.1.7	Keyboard Control	109
4.2.2	The Interactive Generator Block as a Gateway	110
4.3	Replay Block	110
4.4	Trigger block.....	111
4.5	Filter block.....	111
4.6	Channel Filter	112
4.7	CAPL Nodes in the Measurement Setup.....	113
4.8	Break.....	115
5	CAPL Programming.....	116
5.1	Overview	116
5.1.1	Potential Applications of CAPL Programs.....	116

5.1.2	Integration of CAPL Programs.....	117
5.1.3	Use of the Symbolic Database in CAPL	118
5.1.4	Introduction to CAPL	118
5.2	CAPL Browser.....	119
5.2.1	Opening Browser.....	120
5.2.2	Browser Window	121
5.2.3	Compiling CAPL Programs.....	121
5.2.4	Searching for Run-Time Errors.....	121
5.2.5	Access to the Database	122
5.2.6	Importing and Exporting ASCII Files.....	122
5.2.7	Browser Options.....	123
6	.CAN option	124
6.1	The Trace Window of the .CAN option	124
6.2	The Bus Statistics Window of the .CAN option	125
7	.LIN option	126
7.1	Configuration of a LIN Test Environment.....	126
7.2	LIN Scheduler	126
7.3	LIN Specifications.....	126
7.4	The Converter Tool LDF to DBC.....	127
7.5	Trace Window of the .LIN option	127
7.6	The Bus Statistics Window of the .LIN option	128
8	.MOST option	129
8.1	Installation Procedure.....	129
8.1.1	Optolyzer.....	129
8.1.2	Tool4M-XL Installation Section	130
8.2	Configuration settings.....	130
8.3	Timestamps.....	131
8.3.1	Synchronized timestamp	131
8.3.2	Original timestamp	131
8.4	Time Synchronization Accuracy	131
8.5	Database Support	132
8.6	Interactive Generator Block MOST	132
8.7	Trace Window	133

8.8	Bus Statistic Window	134
8.9	Graphic- & Data Window	135
8.10	CAPL	135
8.11	Demo Configurations CANalyzer/DENalyzer	135
8.12	XML Engine	135
9	.FlexRay option	137
9.1	Trace Window for the .FlexRay option	137
9.2	The Bus Statistics Window for the .FlexRay option	138
10	Index	139

1 Introduction

In this chapter you get an overview about the purpose and functionality of CANalyzer. A short tutorial leads you through the essential components of CANalyzer and roughly acquaints you with the individual functions.

1.1 Overview

CANalyzer is a universal development tool for CAN bus systems, which can assist in observing, analyzing and supplementing data traffic on the bus line. Based on its powerful functions and user-programmability, all needs are covered - from initial trial runs with CAN to selective error localization in complex problems.

You can choose to work with CANalyzer on the byte level with bus-like raw data format, or on the application level with logical/physical data representation. A CAN database is used to convert raw data. This database has become a de facto standard in the motor vehicle industry. The user-friendly database management program CANdb++ is included with CANalyzer.

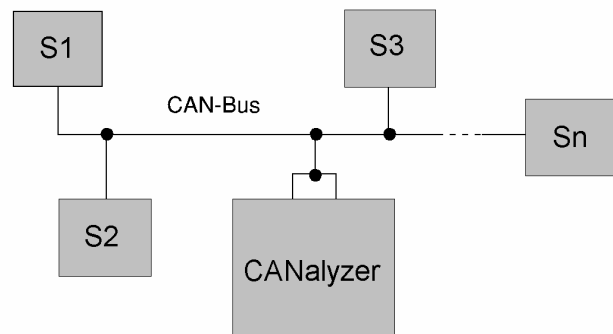
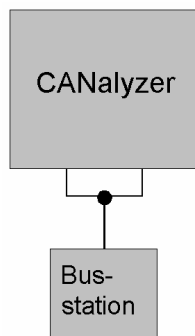
Even the basic built-in functions - which can be used without any programming knowledge - provide for an abundance of possible applications. These include listing bus data traffic (Tracing), displaying data segments of specific messages, transmitting predefined messages and replaying recorded messages, statistically evaluating messages, and acquiring statistics on bus loading and bus disturbances, as well as recording messages for replay or offline evaluation.

Furthermore, the user can expand CANalyzer's functionality as desired by means of user-programming. Program blocks can be inserted at any point in the data flow diagram. The application-oriented, C-like language CAPL (CAN Access Programming Language) serves as the programming language. A special event procedure concept and CAN-adapted language tools enable the user to develop quick solutions to specific problems. CANalyzer contains an interactive development environment that makes it easy to create, modify and compile CAPL programs.

Programmability results in numerous potential applications:

Emulation of a bus station:

CANalyzer can emulate those functions of a bus station which are relevant for data traffic, e.g. transmitting messages in response to certain events.

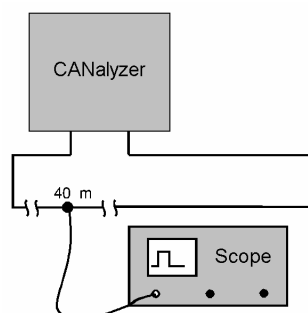
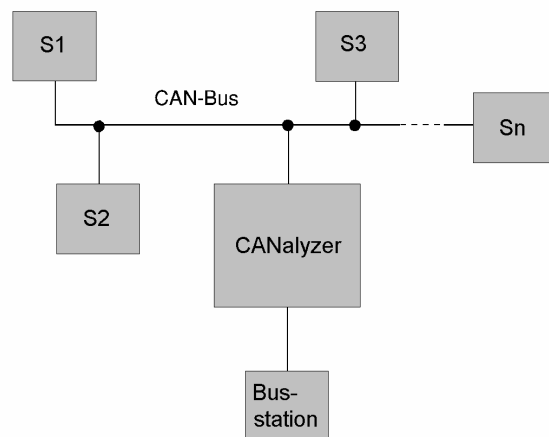


Emulation of system environment for testing a bus station:

In development equipment with bus connections the problem arises that the remaining bus participants not available yet for testing purposes. The data traffic of all remaining stations can be emulated with the help of CANalyzer to emulate the system environment.

Link between two buses:

This provides for exchange of data between CAN buses which may have different speeds. Another target application is to interpose CANalyzer between a bus station to be tested and the actual bus, in order to observe and manipulate the data transfer in a test environment.



Test generator for studying the physical layer:

It is possible to distribute messages intentionally between CANalyzer's two bus connections and to connect a real (i.e. long) bus line. This makes it possible to conduct very simple experiments involving arbitration or line reflections.

CANalyzer is controlled and configured from the data flow diagram in the measurement setup window. For further information on this please refer to section 3.3.2.

1.2 Tips for Using CANalyzer

The basic operating procedures for using CANalyzer are explained in this section. If you are working with Windows for the first time, you should first become familiar with the basics of operating Windows applications. To do this, select the Windows tutorial program under the Help menu in the Windows Program Manager.

Essentially, CANalyzer can be operated by both mouse and keyboard. For example, you can select a main menu by clicking it with the left mouse button. Then you can click again on an item in the submenu which appears, and the associated action is executed.

As an alternative, the main menu can be activated by pressing the <Alt> key. You can now select an item with the cursor keys (<Arrow right>, <Arrow left>, <Arrow up> and <Arrow down> and execute the associated action by pressing the Enter key.

You can deactivate a selected menu entry again by pressing <ESC> or by clicking outside of the menu area with the mouse button.

All of the windows described above can be moved, enlarged, reduced, opened and closed again at any time, i.e. also during the measurement. To move the window simply drag (= press the left mouse key and hold it down while the mouse is moved) the title bar of the particular window to the new position. To change the window size, drag on the sides or corners of the window.

As an alternative you can also perform these actions with the keyboard after calling the system menu (pressing <Alt>-<SPACE> or <Alt>-<->). See the Windows manuals or Windows online Help for further details.

1.2.1 Menus

CANalyzer is operated using the main menu. The individual menu commands are described in detail in online Help

Additionally, there are other context-sensitive menus in the evaluation windows described above and in the data flow plans in the simulation and measurement setup windows. These menus allow the user to specifically configure certain objects. These menus can be opened by clicking the active block in the active window or in the measurement setup window with the right mouse button. Using the keyboard this is done by pressing <F10>.

Most blocks in the measurement and simulation setups can be parameterized by selecting the first item in the context menu **Configuration**. The block's configuration dialog is opened for this purpose. You can also start this dialog directly, without going through the context menu, by double clicking on the active block or pressing the Enter key.

1.2.2 Dialogs

In addition to command inputs, which are usually made using menus, there are also parameter inputs. As a rule, parameters are entered in dialog boxes. A dialog box generally consists of six types of fields, each of which can occur more than once:

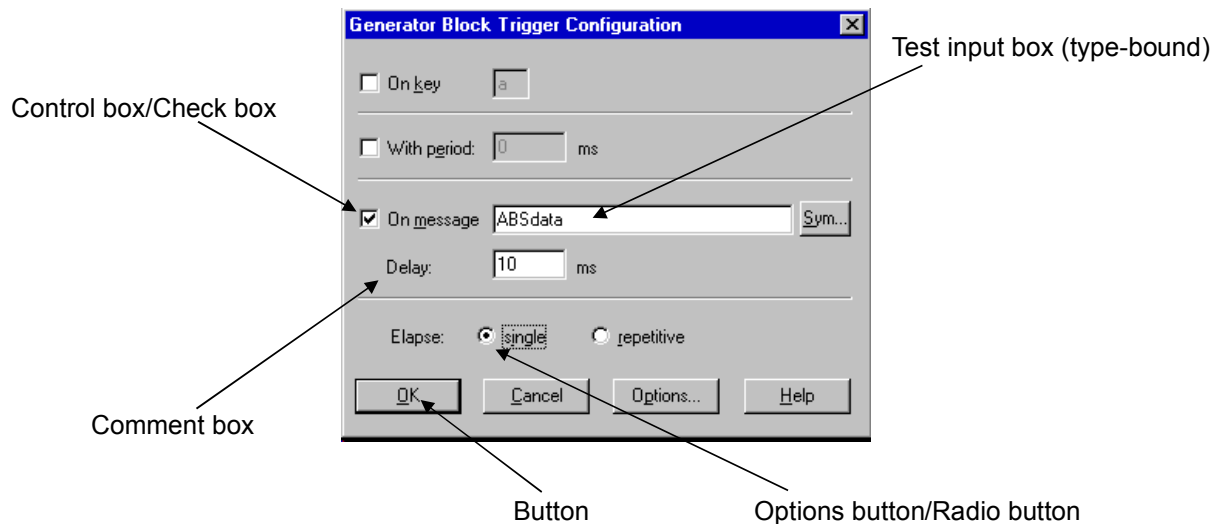


Figure 1: Box Types in Dialogs

Comment box	This tells the user what is to be input. The boxes behave passively when clicking on them with the mouse. They cannot be accessed by keyboard either.
Text input box (type-bound)	Alphanumeric boxfield, e.g. for entering file names. Numeric boxfield, e.g. for entering integer or floating point numbers.
Drop-down list	After clicking on the arrow along the right border of the box, a list drops down, from which you can select a value from a prescribed set of values.
Options button/Radio button	These buttons represent mutually exclusive options. You can only select one option at a time. If you select another option, the previous selection becomes inactive. The currently selected option button is identified by a black dot.
Control box/Check box	A check box next to an option indicates that this option can be activated or deactivated. In this case you can activate as many check boxes as desired. Activated check boxes are identified by an "x" or "✓".
Button	Buttons serve to execute certain actions, e.g. to exit the dialog box or to open a subordinate dialog box.

All dialogs have action buttons labeled **[OK]**, **[Cancel]** and **[Help]**. If you press **[OK]**, the settings you have made in the dialog are accepted into the configuration of the particular block. If you press **[Cancel]**, all settings made since the dialog box was last opened will be lost. With the **[Help]** button you can obtain a help text about the dialog box you are currently working with. After the Help window has been closed you can continue with the dialog. All settings are preserved.

Most CANalyzer dialogs also have an **[Options ...]** button. When this button is activated another dialog appears with which you can modify the CANalyzer global options (decimal/hexadecimal number representation, symbolic/numeric mode).

Note: Modification of the global options from a configuration dialog affects data representation in all system windows and dialogs.

Where there are multiple input and action boxes in a dialog box, first the desired box must be selected. Using the mouse this is done by clicking on the appropriate box. For input boxes this causes the text cursor to be placed at the mouse pointer position in the box. Check boxes change their state, and for action boxes the action is executed. With keyboard operation the particular box is selected with <Tab> or <Shift-Tab>. Check boxes can be then be toggled using the spacebar. The <Enter> key closes the dialog box and executes any actions selected in action boxes.

1.2.3 Control of the Measurement Setup

Measurements and evaluations are primarily configured in the measurement setup window which shows CANalyzer's data flow plan for the particular operating mode.

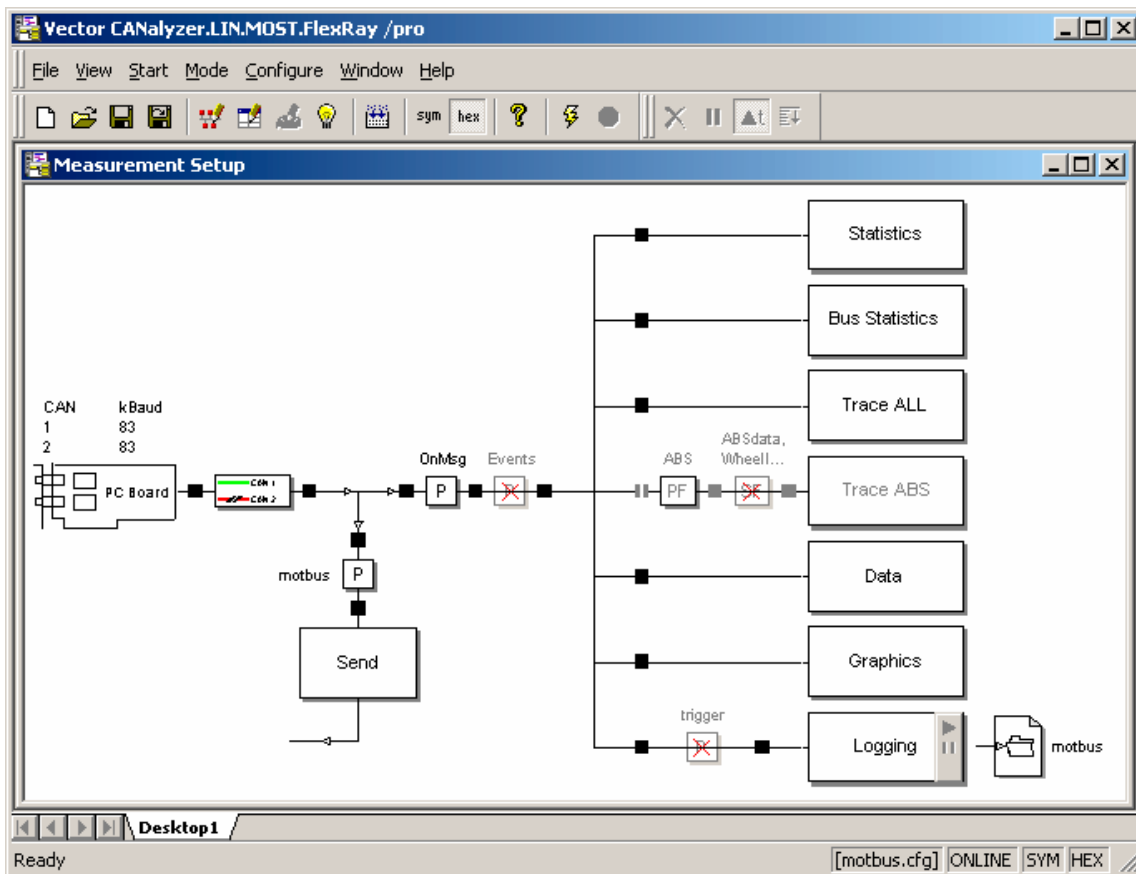


Figure 2: CANalyzer Measurement Setup

Mouse Operation

All blocks and some images in the active measurement setup window are mouse sensitive. When selected by clicking the left mouse button, the element pre-selected in this manner is identified by a frame as the *Active Element*. When the right mouse button is then clicked, a popup menu appears in which the object is

configured by the methods described above. As an alternative, the configuration dialog for the active block can be called directly by double clicking with the left mouse button.

Keyboard Operation

When the measurement setup window is active, if <Tab>, <Shift-Tab> or one of the cursor keys is activated, the preselect frame around the currently active element is indexed forward. <Tab> results in forward indexing (<Shift-Tab>: Reverse indexing) of the internal processing sequence. The cursor keys index forward to the next closest element geometrically in the direction of the arrow. When <F10> is activated the popup menu of the active element appears. As an alternative, the Enter key can be used to call the configuration dialog of the active block directly.

The spacebar can be used to deactivate the preselected function block in the measurement setup; it can be reactivated by pressing the spacebar again.

With <Ctrl-F6> and <Ctrl-Shift-F6> you can bring any opened CANalyzer window to the foreground and activate it.

1.2.4 The Help System

Selecting the main menu item **Help** opens a Help contents window, which contains basic information and references to other Help pages. You can select references by clicking with the mouse or indexing through them with <TAB> and then pressing the Enter key.

The CAPL Browser and CANdb++ Editor each have their own Help system with another main **Help** menu. Activate this from the particular program.

Activating the <F1> key causes a Help topic to appear for the element that is active or preselected at the time the key is pressed. This context-sensitive Help function is provided for all dialogs, all program window panes and for all menu items, both in the main menu and in popup menus.

1.3 CANalyzer Tour

If you are starting up CANalyzer for the first time, and its functionality and controls are still completely new to you, the following tour will help you to become familiar with its operating concept and its most important features.

For this tour you will first set up a very simple CAN bus where CANalyzer assumes the roles of both sender and receiver. In the first step CANalyzer is configured as a data source, i.e. as a transmitting station. You will then learn about CANalyzer's analysis options by studying the generated data in the measurement windows afterwards.

In complex real systems CANalyzer typically also assumes both roles. You can utilize the program as a data source to transmit data to other controllers, but you can simultaneously use it to observe, log and evaluate the data traffic on the CAN bus.

In the last part of the tour you will become familiar with the CAPL programming language and expand CANalyzer functionality by adding a simple CAPL program.

1.3.1 Preparations

To start CANalyzer, call `CANW32.EXE` by double clicking the appropriate icon in the CANalyzer program group.

CANalyzer has various evaluation windows (Trace, Data, Graphics, Statistics and Bus Statistics windows) as well as a measurement setup window that shows you the data flow in CANalyzer and simultaneously allows you to configure CANalyzer.

You can access all program windows from the **View** menu on the main menu bar.

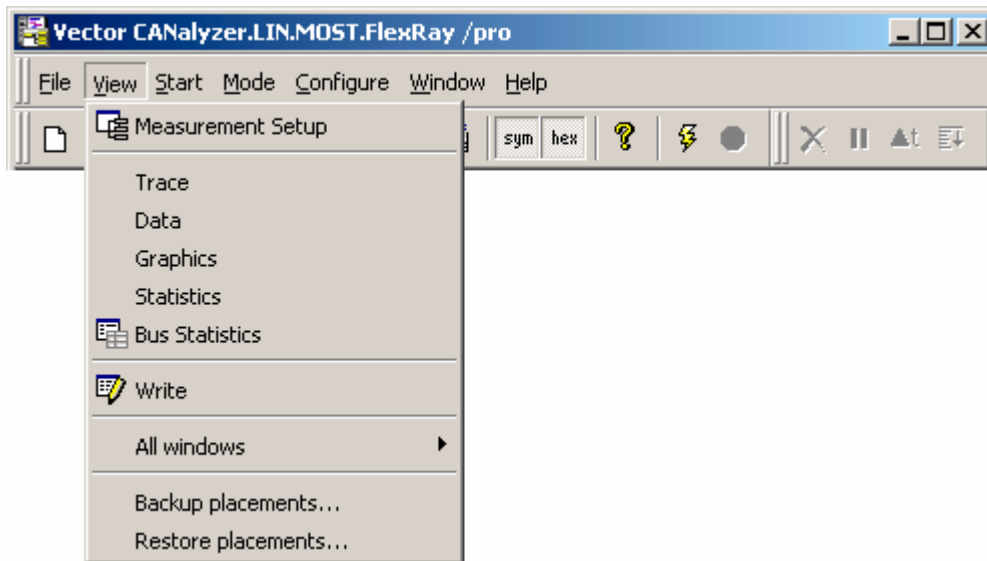



Figure 3: **View** Menu on Main Menu Bar

The data flow diagram of the CANalyzer measurement setup contains the data source on the left - symbolized by the symbol of a PC-card - and various evaluation blocks on the right serving as data sinks. That is, the data flow is from right to left. Connection lines and branches are drawn between the individual elements to clarify the data flow.

The information arriving at each evaluation block are displayed in the block's evaluation window. For example, the Trace window displays all information arriving at the trace block, while the Graphics window shows you information arriving at the graphics block.

The only exception is the logging block, which is not assigned a window but rather a file in which the data arriving at the block are logged.

On the left side of the measurement setup, CANalyzer's transmit branch branches off after the card icon. Data can be sent onto the bus from here. The data flow in the transmit branch always runs from top to bottom.

In the data flow diagram you will also recognize small black rectangles: . At these insertion points (Hotspots) you can insert additional function blocks for manipulating the data flow (Filter, replay and generator blocks, or CAPL program blocks with user-definable functions).

Make sure that you begin this tour with a new configuration by selecting the menu item **File | New configuration**.

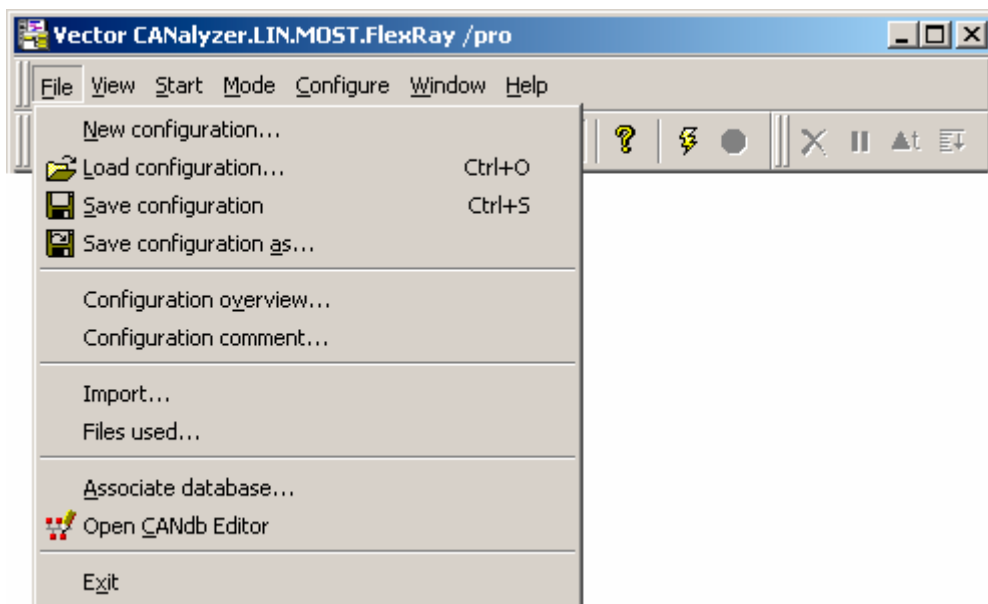


Figure 4: Menu Item **File | New configuration**

1.3.2 Setting Up the CAN Bus

To start up CANalyzers it is advisable to use a test setup with only two network nodes that is independent of existing CAN bus systems. The two CAN controllers on the supplied PC-card can serve as the network nodes.

First, connect the two D-Sub-9 connectors of your CAN card to one another. A connection cable with two bus termination resistors of 120Ω each for the high-speed bus interface is provided with the CANalyzer product. For a low-speed interface you will simply need a 3-conductor cable to interconnect the pins of the two controllers that are assigned to the bus lines CAN-High, CAN-Low and Ground.

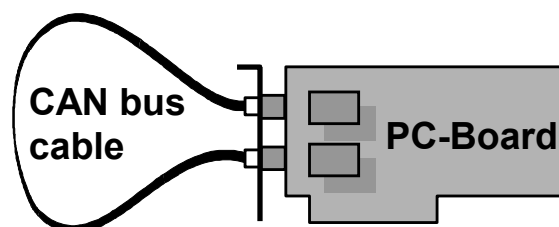


Figure 5: PC-Card with Connection Cable

Consequently, the CAN bus that you use during this tour will consist of a short 2-conductor or 3-conductor cable that connects the two CAN controllers of the CAN card to one another. This is necessary as a minimal configuration, since the CAN pro-

tol requires - in addition to a sender - at least one receiver that confirms the correct receipt of messages with an acknowledge.

Up to this point we have not considered definitions of bus parameters (Transmission speed, sampling point, etc.) which must be set for each of the two participating controllers. To do this, from the **View** menu bring the measurement setup to the foreground and click the right mouse button on the PC-card icon at the left of this window.

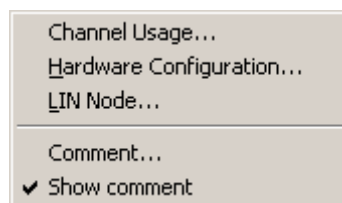


Figure 6: Popup Menu of the PC-Card Icon

In the popup menu you initially select bus parameters for the first controller CAN 1 and first set the baud rate in the configuration dialog. The Baud rate selection button takes you to the baud rate direct selection dialog, where you enter the value 100 kBaud.

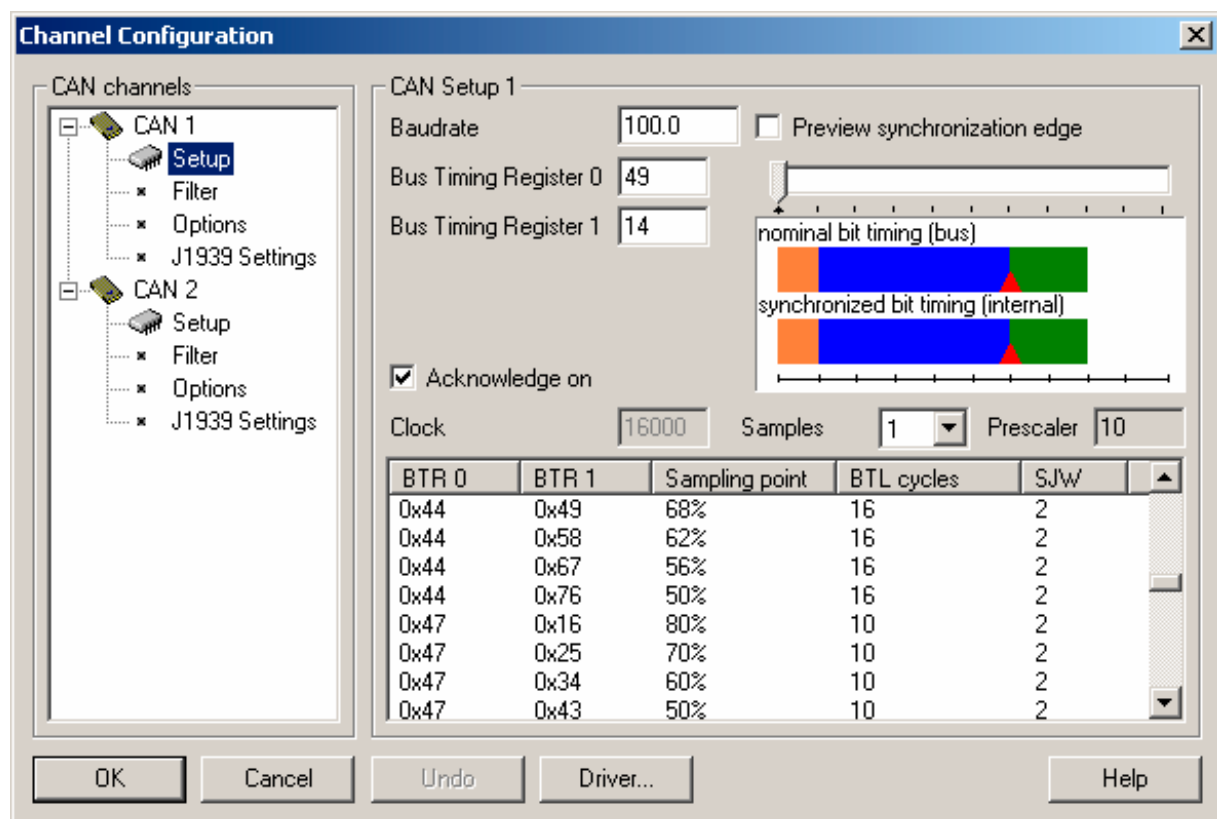


Figure 7: Configuration of Bus Parameters and Direct Baud Rate Selection

This makes sense for both high-speed and low-speed buses. The CANalyzer recommends default values for the controller registers. When you do this - besides the

transmission speed of 100 kBaud - you also implicitly define other controller parameters (Sampling point, BTL cycles, and synchronization jump width). For the overall system to function properly, the same exact values must be assumed for the second controller CAN2. Accept the parameters with **[OK]**..

1.3.3 Transmitting Data

Since your current test setup still does not have a data source, your first task is to set up a data source in CANalyzer which places information on the bus cyclically.

Unit 1: Configure CANalyzer so that - after the measurement start - a CAN message with identifier 64 (hex) is placed on the bus every 100 milliseconds. In this case the message should contain exactly four data bytes with the values D8 (hex), D6 (hex), 37 (hex) and 0.

You can solve this task by inserting a generator block in the CANalyzer transmit branch which generates the message to be transmitted. This is done by clicking with the right mouse button on the hotspot in the transmit branch directly above the block labeled **Transmit** and - from the hotspot's popup menu - inserting a generator block in the transmit branch.

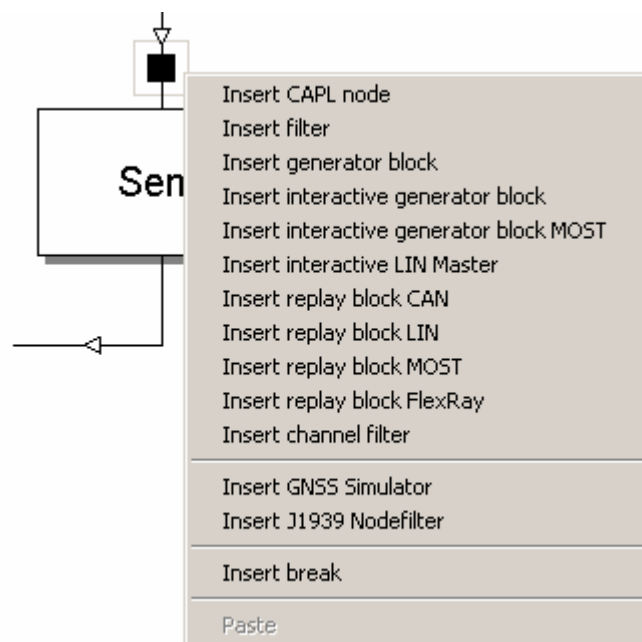


Figure 8: Hotspot in the Transmit Branch with Popup Menu

Afterwards, this appears as a rectangular block with the label G. You can then configure this block from its popup menu, which you access by pressing the right mouse button.

Afterwards, this appears as a rectangular block that is connected to the simulated bus (red line). You can then configure this block from its popup menu, which you access by pressing the right mouse button.

First, fill out the transmit list. You enter 64 as the identifier. (Check to see whether the numbering format is set to **Hex** using the **[Options]** button.) Then enter the value 4 in the DLC box as the data length entry. Finally, set the values of the data bytes in the four data boxes that follow by entering the values D8, D6, 37 and 0 there.

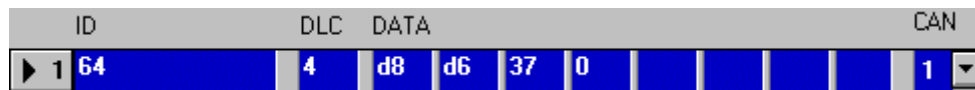


Figure 9: Transmit List of Generator Block

Exit the transmit list with **[OK]** to accept the values in the configuration. In the generator block's popup menu, you must now still configure triggering for the transmit action. On the second line check the box **With Period** and then enter the value 100 in the input box to the right of this.

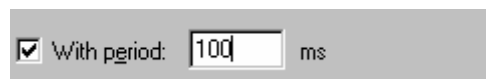



Figure 10: Triggering of Generator Block

These values are assumed into the configuration with **[OK]**.

Before you start the measurement you should save the configuration that you have prepared to this point with the menu command **File | Save configuration**. You can then reload this configuration at any time and resume your work precisely at this point.

Start the measurement by pressing the start button  on the toolbar. CANalyzer immediately begins to cyclically transmit the message you have configured in the generator block. You can recognize this in the Trace window, which automatically jumps to the foreground after the start of measurement and can now be seen at the lower right of the main program window: In the first line you see the message that is sent by the generator block, whereby the first column shows the transmit time relative to the measurement start.

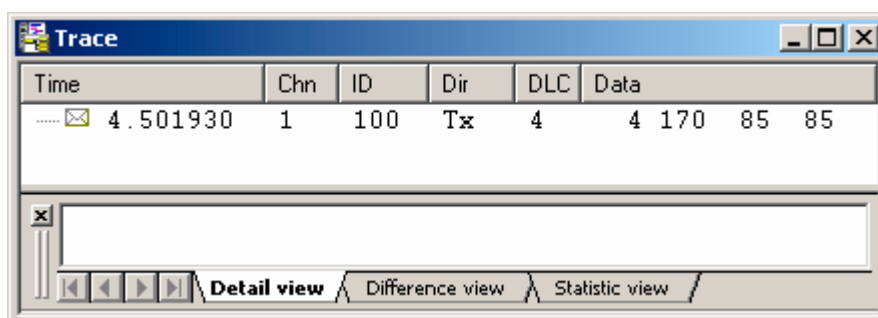


Figure 11: Trace Window

The next column shows you which of the two CAN controllers was used to transmit. This value (CAN 1) agrees with the default value assigned in the generator block's transmit list of messages to be transmitted.

Afterwards, this message is also received by the second CAN controller over the bus. The question arises: Why is this not also displayed in the Trace window? You will find the answer in the configuration dialog for the acceptance filter for the second controller. In turn, you can open this dialog from the PC-card icon's popup menu under the entry **CAN Bus parameters | CAN2 | Acceptance**.

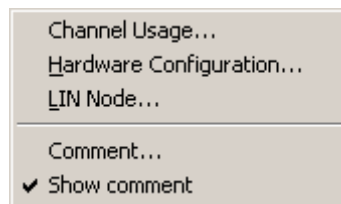


Figure 12: Popup Menu of the PC-Card Icon

The **acceptance filter** options support hardware-side filtering of messages. The default options block most message receiving. You can open the filter by entering the value **X** in the upper line.

After a new measurement start you can now also see that the message transmitted via channel 1 (Transmit attribute Tx [= Transmit] in the Trace window) was received by the second controller (Receive attribute Rx [= Receive] in the Trace window).

We will now expand the task and additionally transmit a message with modified data:

Unit 2: Expand the configuration of the last task such that, additionally, a message with identifier 3FC (hex) is transmitted every 200 milliseconds. The value of the first data byte of this message should cyclically assume values from 1 to 5.

You can solve this task by inserting another generator block in the transmit branch. For this task it does not matter whether you insert this generator block before or after the first one. Select *200 ms* as the value for cyclic triggering. The transmit list should appear as shown below:

	ID	DLC	DATA													CAN
▶ 1	3fc	1	1													1 ▼
2	3fc	1	2													1 ▼
3	3fc	1	3													1 ▼
4	3fc	1	4													1 ▼
5	3fc	1	5													1 ▼

Figure 13: Transmit List for Generator Block


Do not forget to stop the measurement before you reconfigure the measurement setup. During a running measurement it is not possible to make changes to the configuration of the data flow. The menu items of the relevant popup menus appear in gray shading.


Besides the generator block, CANalyzer also offers two additional block types as data sources. With a replay block you can play back data on the bus that were logged with CANalyzer's logging function. A program block allows you to integrate your own transmit functionalities - which may be quite complex - into CANalyzer with the CAPL programming language (cf. chapter 5).

1.3.4 Evaluation Windows

Evaluation windows are used to analyze data generated by the generator blocks in the transmit branch.

You have already learned about the Trace window. Data that reach the trace block of the measurement setup are displayed here as CAN messages in bus-oriented format. Besides the time stamp, this includes the number of the CAN controller, the identifier, an attribute for differentiating transmitted and received messages, and the data bytes of the CAN message. You can configure the Trace window - like all other analysis windows - from the popup menu that is accessed by clicking the right mouse button on the window or on the appropriate block.

Furthermore, the four buttons on the right of the toolbar can be used to configure the Trace window. For example, with  you can toggle from **stationary mode** to the **scroll mode**, in which each message arriving at the trace block is written to a new line.

With  you can toggle between absolute and relative **time representation**. In relative time representation, the time difference between two successive messages ("transmit interval") is shown in the first column. Of course, in this display format it is also easy to find the transmit interval that you entered previously in the generator block: 100 milliseconds.

The Statistics window also offers you bus-related information. Here you can observe the transmit frequencies for messages, coded by identifiers. If you have configured the transmit branch as in the two last tasks, then you should see two vertical lines in the Statistics window after the measurement start, which show the transmit frequencies of the two generated messages 64 (hex) and 3FC (hex).

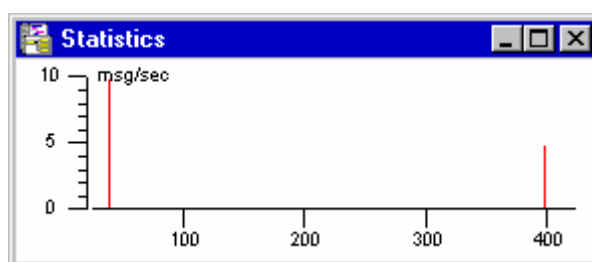
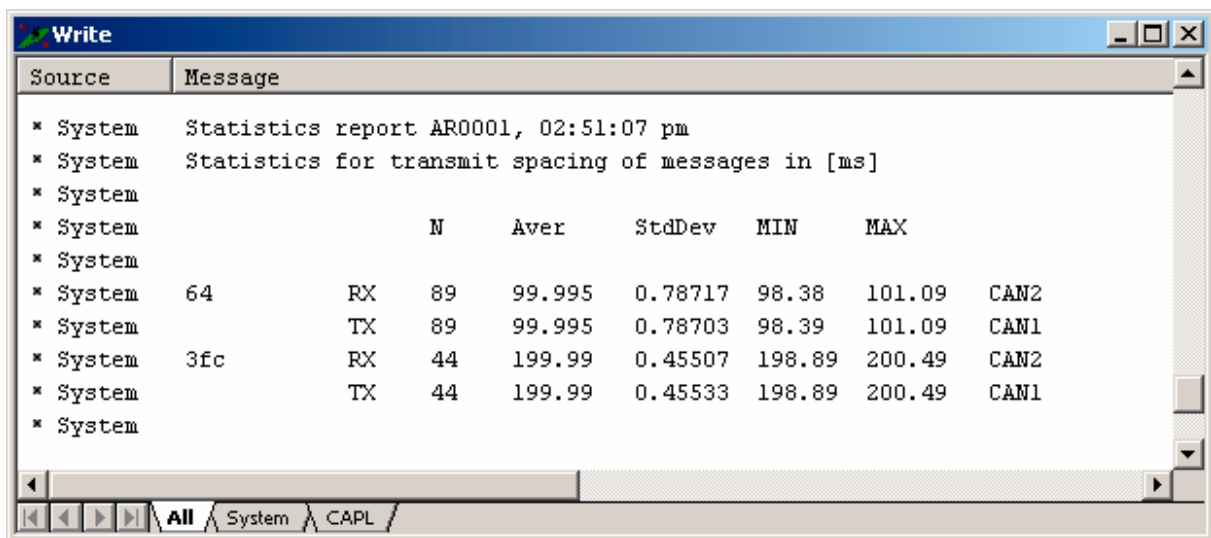


Figure 14: Statistics Window

10 messages per second were recorded for identifier 64, and half as many were recorded for identifier 3FC. This result corresponds to the cyclic periods of 100 and 200 milliseconds set in the generator blocks.

If the Graphics window display is too imprecise, the statistics block offers you a statistical report that gives you more precise information on the transmit interval for each message. Stop the measurement and activate the statistical report in the popup menu of the Statistics block (**Statistic report... Activate**).

If you now restart the measurement, statistical information is gathered in background, which you can output to the Write window after the next measurement stop using the popup menu command **Display statistics report**.

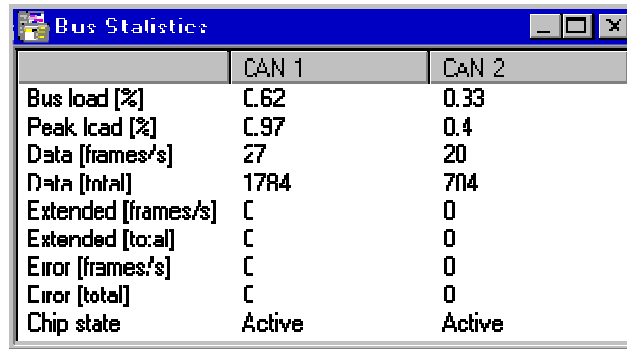


Source		Message						
* System Statistics report AR0001, 02:51:07 pm								
* System Statistics for transmit spacing of messages in [ms]								
* System								
			N	Aver	StdDev	MIN	MAX	
* System								
* System	64	RX	89	99.995	0.78717	98.38	101.09	CAN2
* System		TX	89	99.995	0.78703	98.39	101.09	CAN1
* System	3fc	RX	44	199.99	0.45507	198.89	200.49	CAN2
* System		TX	44	199.99	0.45533	198.89	200.49	CAN1
* System								

Figure 15: Statistics Report

Besides showing the total number of messages for each identifier, the statistics report also shows the mean value, standard deviation, and minimum and maximum for the recorded transmit interval.

Another bus-related window, the Bus Statistics window, provides an overview of bus data traffic. Displayed here are the total frequencies of data, remote, error and overload frames, bus loading and CAN controller status. Since in our case one message is sent every 100 ms and the second message every 200ms, the total frequency of all messages is 15 frames per second. With an average data length of about 70 bits per frame, approx. $15 * 70 \approx 1000$ bits are placed on the bus in one second. At a baud rate of 100 kBit/sec the bus load in our example would be on the order of magnitude of one percent.




	CAN 1	CAN 2
Bus load [%]	0.62	0.33
Peak load [%]	0.97	0.4
Data [frames/s]	27	20
Data [total]	1784	704
Extended [frames/s]	0	0
Extended [total]	0	0
Error [frames/s]	0	0
Error [total]	0	0
Chip state	Active	Active

Figure 16: Bus Statistics Window

1.3.5 Working with Symbolic Data

Before we discuss the remaining windows in detail, let us have a look at the capabilities offered by CANalyzer for the symbolic description of data. Of primary interest in the analysis of CAN systems - besides bus-related information such as messages, error frames and message frequencies - is information on useful data, i.e. signals such as RPM, temperature and engine load, which are provided by individual controllers, and are sent on the bus with the help of CAN messages.

To describe this information symbolically, CANalyzer provides you with the database format DBC and a database editor with which you can read, create and modify CAN databases. Please refer to the CANdb++ manual and the CANdb++ online help included with the CANalyzer product for more information on the CANdb++ editor.

At this point we would like to associate a prescribed database to the active CANalyzer configuration. This database will be used to interpret the data bytes of the messages generated by the generator blocks in the transmit branch. The database `MOTBUS.DBC` is located in CANalyzer's demo directory `DEMO_CAN_CL` which was created parallel to the system directory `EXEC32`, provided that you installed the demo configurations. You associate this database to the active CANalyzer configuration by choosing the menu command **File | Associate database**. You can now open the database using the  button on the toolbar. The CANdb++ Editor is opened, and the contents of the database `MOTBUS.DBC` are shown in the Overall View window of the CANdb++ Editor.

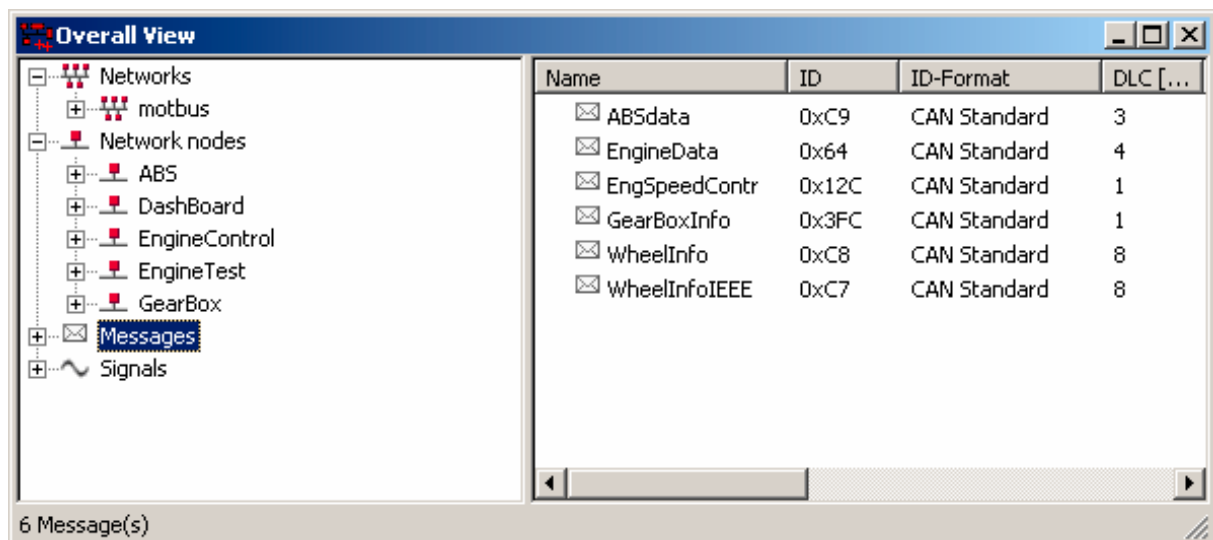


Figure 17: Overall View Window of the CANdb++ Editor

Double click the Messages object type in the area on the left side of the Overall View window. The subordinate structural level is then shown in this area. In the area on the right the available messages are shown with their system parameters (e.g. symbolic name, identifier, etc.). First, toggle the numbering format from decimal to hexadecimal in the **Options | Settings** menu item. We can deduce from the symbolic names of the messages that the system under consideration involves a description of communications in a rudimentary engine area system.

Click the message *EngineData* in the left area of the overall view window. The system parameters of signals transmitted in this message are shown in the area on the right side of the Overall View window.

The temperature *EngTemp* for example is a 7 bit signal. To obtain the physical value in degrees Celsius, the bit value must be multiplied by the factor 2, and the offset 50 must be subtracted from the result. The idle switch signal *Idle Running* in the last bit of the third data byte is a binary signal (one bit), which can assume the value 0 or 1.

With the help of this symbolic information the data contents of messages can now be interpreted in CANalyzer. Please note that this only makes sense if the database information describes the system that you are currently observing. Of course, you can also associate a different database to CANalyzer. The observed CAN data traffic is then interpreted according to the information in that database, even if it does not make any sense. You yourself are responsible for ensuring that the database associated to the configuration matches the real CAN network.

Messages that you generate in the two generator blocks can be interpreted with the database `MOTBUS.DBC`. Please note that the message you generated in the first task has the identifier 64 (hex). This agrees with the identifier of the message *EngineData* that we just examined in the database editor. If you now start the measurement, you can toggle the program to symbolic mode by activating the `sym` button.

In the Trace window you will now see the symbolic message name in addition to the identifier.

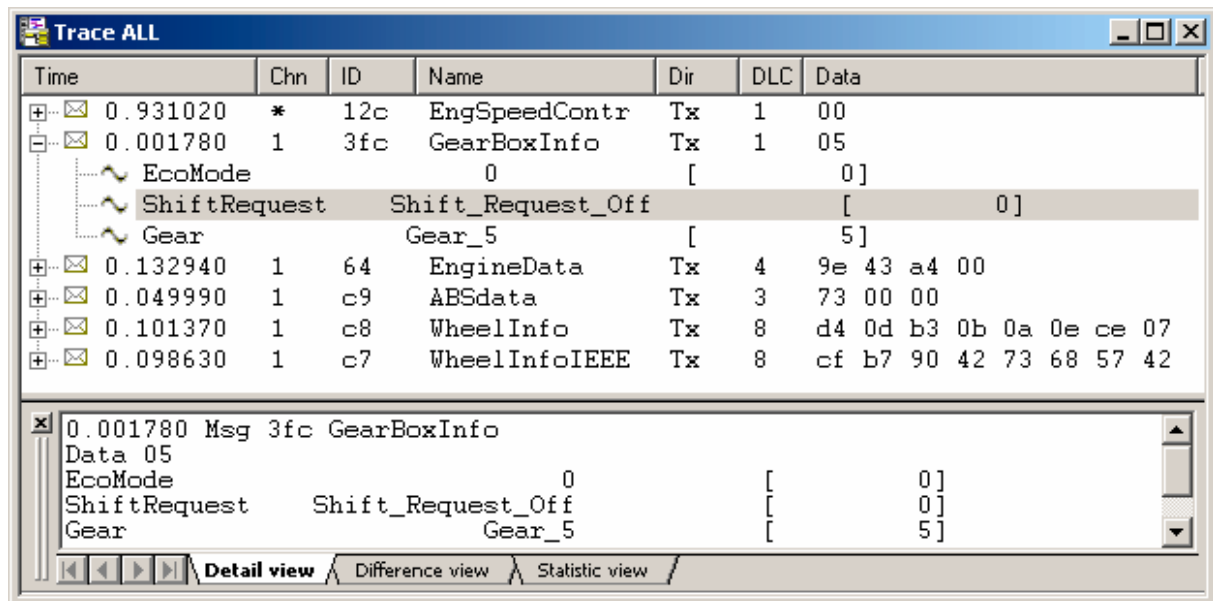


Figure 18: Trace Window

1.3.6 Analysis of Signal Values in the Data Window

Besides the use of symbolic message names, the associated database can also be used to analyze signal values. The purpose of the Data window is to assist in the study of momentary signal values.

This explains why the Data window is initially empty in a new configuration. The signal values to be displayed are exclusively dependent upon information from the database. You as the user must decide which signal values should be displayed.

Unit 3: Configure the Data window to display the signal values of the message *EngineData* (ID 64 hex) that is generated in the transmit branch.

To solve this task, first open the Data window's popup menu and then start the configuration dialog. Initially, the signal list in this dialog is still empty. With the **[New Signal]** button you start the Signal Explorer, which makes it possible for you to select a signal from the database. The object hierarchy on the left side of the dialog allows you to search for a specific signal. On the right side are the signals of the selected object.

To configure the Data window, first select *EngineData* from the list of all messages.

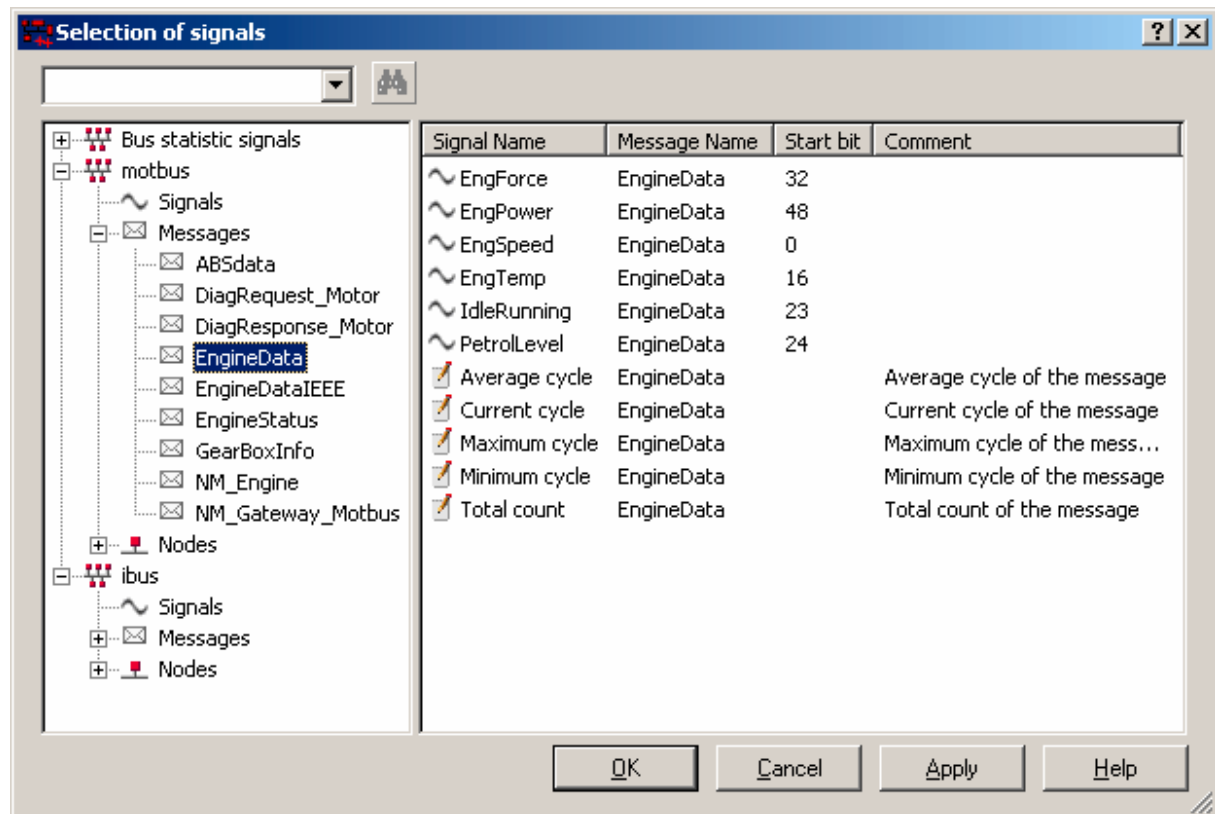


Figure 19: Selecting Signals with the Signal Explorer

Afterwards, select and accept all signals of this message from the dialog list on the right.

When you close the Data window's configuration dialog you will see that the signal names are now entered in the window. After the measurement start the generator block begins to cyclically send the message *EngineData* with data bytes D8, D6, 37 and 0 onto the bus. According to the message description in the database, the data block in the measurement setup now interprets these byte values as engine speed, temperature and idle switch and displays the appropriate signal values in the Data window in physical units.

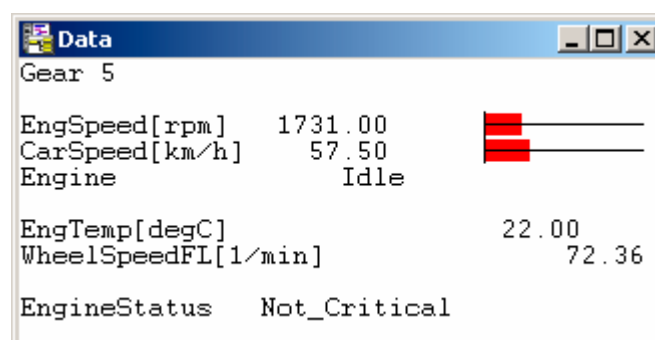


Figure 20: Data Window

With the help of the conversion formula in the database, engine speed is shown in RPM, while temperature is shown in degrees Celsius. The values of all three signals

remain constant over time, since the message is constantly transmitted with the same data bytes D8, D6, 37 and 0.

1.3.7 Analysis of Signal Responses in the Graphics Window

While the Data window displays momentary signal values, you can have the time responses of signal values displayed in the Graphics window. After the end of measurement the signal responses are available for study by user-friendly analysis functions.

Unit 4: Configure the Graphics window so that signal values are displayed for message 3FC (hex) that is generated in the transmit branch.

The second message generated in the transmit branch is also described in the associated database. In the database it will be apparent to you that the identifier 3FC is associated with the symbolic message name *GearBoxInfo* containing the signals *Gear*, *ShiftRequest* and *EcoMode*.

You can now observe the time responses of these signals in the Graphics window. The Graphics window can be configured exactly like the Data window. Here too you open the configuration dialog for signals from the window's popup menu. In the signal selection dialog you select the 3 signals of the message *GearBoxInfo*. In the Graphics window you see that the signals are now entered in the legend on the left side of the window. After the measurement start you observe that the signal *Gear* cyclically assumes values from 1 to 5, while the other two signals remain constant over time.

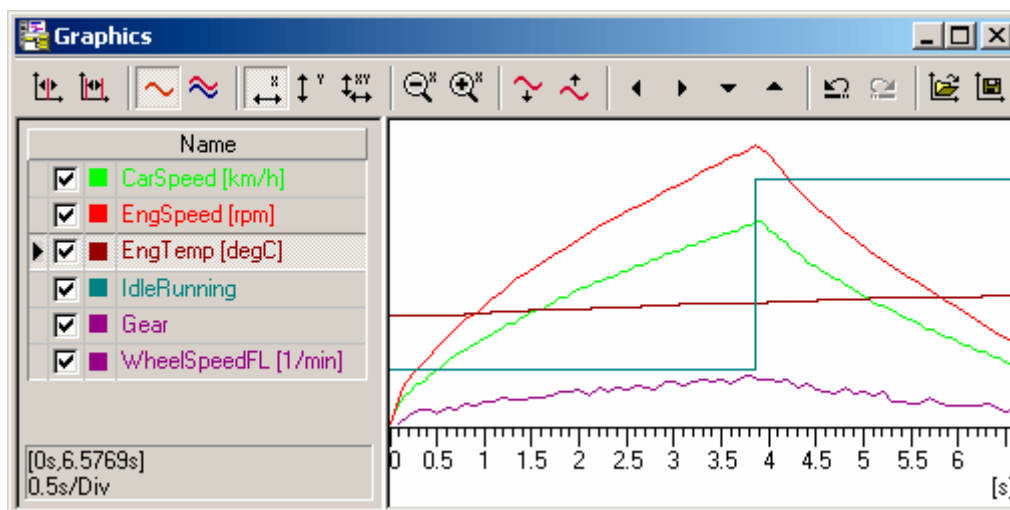


Figure 21: Graphics Window

This corresponds to the five values that you entered in the generator block as part of task 2. The values remain in the Graphics window after the end of the measurement. The measurement functions that the window provides for post-analysis are described in section 3.5.5.

1.3.8 Use of the Database in Transmitting Messages

Until now you have only used the symbolic database to observe signal values. However, the application capabilities reach well beyond this. For example, double click the generator block of task 1 to open the transmit list. Instead of the identifier that you previously entered in the transmit list, you will now recognize the associated symbolic name in the first column. In fact, you can now enter a message directly from the database using the **[Symbol...]** button, without having to work with the identifier.

Signal values can also be edited directly in the transmit list now. Select the first line of the transmit list and then activate the **[Signal...]** button. In the values dialog you can now enter the signal values directly. It will also be apparent to you, once again, that the byte values D8, D6, 37 and 0 from the first line correspond to the signal values *EngSpeed* = 5500 rpm, *EngTemp* = 60 degrees Celsius and *IdleRunning* = 0.

	name	enum	value
1	IdleRunning	Running	0
2	EngTemp		60
3	EngSpeed		5500

Figure 22: Values Dialog in the Generator Block

If you now set - for example - the value of *EngSpeed* to 1000 rpm, the generator block automatically uses the database information to compute the corresponding data bytes (10, 27, 37 and 0).

1.3.9 Analysis of an Engine Area Simulation

Included with the CANalyzer product are several sample configurations, which are provided to assist you in startup. In the directory `DEMO_CAN_CL` you will find a CAPL program block which simulates a portion of the data communication on the engine area bus of a motor vehicle. You have already added the database underlying the model, `MOTBUS.DBC`, to your configuration in Task 3.

Unit 5: The CAPL program `MOTBUS.CAN` simulates the RPM, vehicle speed, and engine temperature for a motor vehicle. Study these signals in the Data and Graphics windows while you shift the vehicle's gears during the measurement run with the '+'- and '-' keys.

To solve this task, first delete both generator blocks from CANalyzer's transmit branch and insert - in the transmit branch in their place - the CAPL program `MOTBUS.CAN` from your demo directory `DEMO_CAN_CL`: To do this, choose **Insert CAPL program** from the popup menu of the hotspot in the transmit branch. Afterwards, in the configuration dialog for the inserted program block, you would press the **[File...]** button to assign the program file `DEMO_CAN_CL\MOTBUS.CAN`. Finally, you must still compile the program (Menu command: **Configuration | Compile all nodes**). The transmit branch is now prepared.


Now, in the Data and Graphics windows you configure the signals for engine speed (*EngSpeed*) and temperature (*EngTemp*) to the message *EngineData*, the vehicle speed signal (*CarSpeed*) to the message *ABSData* and the gear signal (*Gear*) to the message *GearBoxInfo*.

Once you have started the measurement, you can view the bus traffic directly in the Trace window. The messages *EngSpeed* and *ABSData* are transmitted cyclically, while the message *GearBoxInfo* is only transmitted spontaneously and is transmitted only once at the measurement start and with each gear shifting action, i.e. when the '+'- or '-' key is activated.

You can observe the signal values in the Data and Graphics windows. After the measurement start the temperature rises slowly to a maximum value, while the vehicle speed and engine speed vary between two values. The ratio of the two signal values is determined by the selected gear.

You will find an introduction to CAPL programming and a detailed presentation of the programming language in chapter 5.

1.3.10 Logging a Measurement

CANalyzer has extensive logging functions for data logging. In the standard measurement setup the logging branch is shown at the very bottom of the screen. You can easily recognize it by the file icon  that symbolizes the log file. The log file is filled with CAN data during the measurement.

Unit 6: Log - in ASCII format - all CAN data traffic that is generated in a short measurement (approx. 20 sec.) by the generator blocks in the transmit branch.

To log the data that arrive in CANalyzer's measurement setup to a file, first activate the logging branch. Also remove the break that separates the logging block of a new configuration from the data source. From the popup menu of the file icon located at the far right of the logging branch, open the configuration dialog. Here you can enter the file name for the measurement log as well as its format. Select ASCII format here.

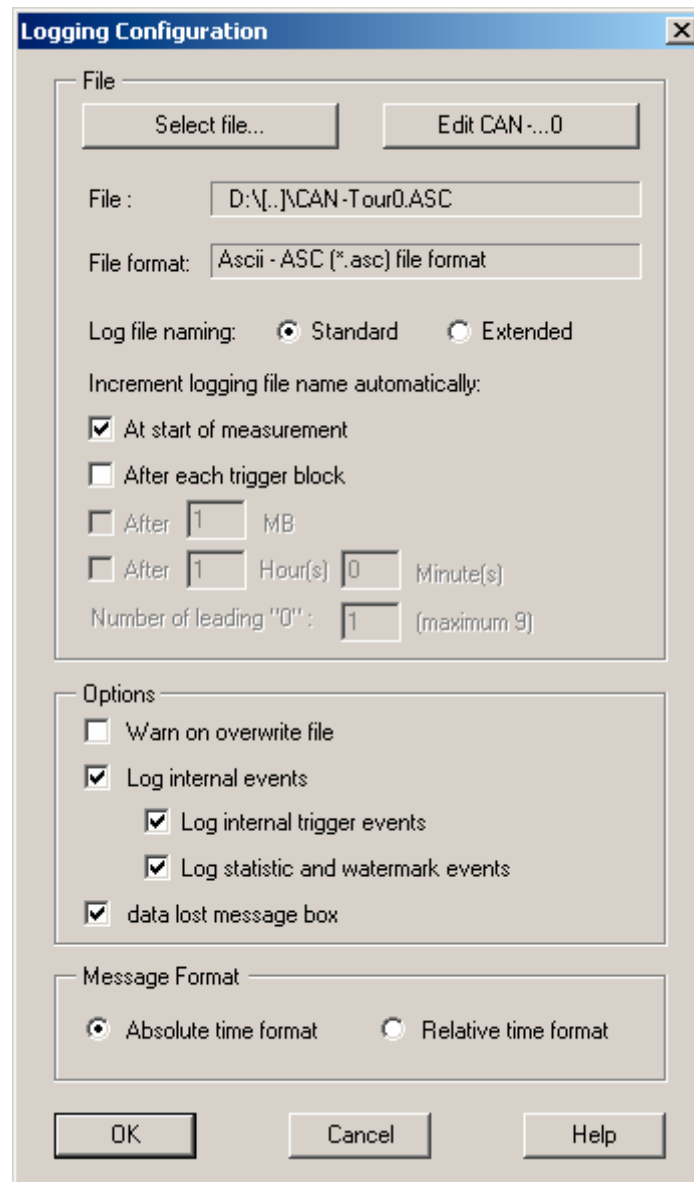


Figure 23: Configuration Dialog in the Logging Branch

Logs in binary format take up less space on your hard drive, but they cannot be read by normal text editors. The program's Offline mode offers you the same evaluation options for logs in both formats.

Besides the file icon, you can also specify trigger conditions for file logging in the logging block. This is often advisable, since frequently it is not the data traffic on the can bus over the entire measurement period that is of interest, but rather only certain time intervals, e.g. when there are implausible signal values or when error frames occur.

A description of how to define trigger conditions and time windows around these conditions is presented in section 2.6. To log the entire measurement it is sufficient to change the mode from **Single Trigger** to **Entire Measurement** in the trigger configuration dialog.

Exit the dialog with **[OK]** and then start the measurement, which you stop again after 20 seconds. Now with a double click on the log file icon you can open the logged ASCII file. Besides the logged messages you can see that statistical information was

also logged. These lines correspond exactly to the information that is displayed in the Bus Statistics window during a measurement.

1.3.11 Evaluating a Log File

Log files in ASCII format can indeed be viewed with text editors, but often it is more sensible to utilize the capabilities that CANalyzer provides for offline analysis of log files.

Unit 7: Play back the log file recorded for the last task in Offline mode, and observe the signal response in the Graphics window.

To solve this task, first switch CANalyzer to Offline mode. In the main **Mode** menu you will find two entries for this: **To Offline** and **To Offline (Copy)**. Since you can use the Graphics window configuration you prepared in Online mode here too, it is advisable to copy all configuration options of the analysis branch to Offline mode with **To Offline (Copy)**.

Now shown as the data source in the measurement setup - instead of the PC-card icon - is a file icon. Of course, the transmit branch is omitted here. Otherwise, all of measurement setup options of Online mode have been assumed. You can configure the data source by double clicking the file icon at the left of the measurement setup and entering the name of the log file selected in the last task.

You can now play back the measurement with the <F9 key>. In contrast to Online mode, here CANalyzer also offers you the option of replaying the measurement in slow motion (**Start | Animate** menu item or <F8> key) or in Single-Step mode (**Start | Step** menu item or <F7> key).

The same analysis functions are available to you in Offline mode as in Online mode. That is, the logged data are displayed in bus-related format in the Trace window, while you can observe the log's signal responses in the Graphics window.

Of course, you can also insert filters or CAPL programs in the measurement setup to further reduce the data or introduce additional user-defined analysis functions.

1.3.12 Tips for Solving Your Own Tasks

This small tour should make you aware of the fundamental control concepts and most important features of CANalyzer.

Remember that CANalyzer's measurement setup window represents the data flow plan for your actual measurement task. Besides associating a database, you can configure all other options directly in this window: From the data source on the left side, to the transmit branch, to the evaluation blocks on the right side of the window. You can always access the popup menus of all measurement setup objects by pressing the right mouse button.

All data arriving at an evaluation block are - with the exception of the logging block - displayed in the corresponding window. The evaluation windows can also be configured by pressing the right mouse button. You can save all configuration settings in a configuration file. Simply load such a prepared configuration file to prepare CANalyzer for another measurement task.

If you are using CAPL for the first time, perhaps to write your own analysis functions or to study the bus behaviour of a controller, you will find a brief introduction to CAPL in section 5.1.4.

In the CAPL manual you will find detailed explanations of the program's transmit and analysis functions, which are only discussed briefly here, and explanations of CAPL programming. The context-sensitive Help function (F1 key) describes all menu items and explains the displays and controls of all dialogs.

1.4 Overview of the Programs

The following executable programs are part of CANalyzer:

- With the **CANdb++ Editor** you create or modify databases (*.DBC) which contain the symbolic information for CANalyzer. This includes network nodes and symbolic names for messages and signals.
- In the **CAPL Browser** you create CAPL programs for the transmit and analysis branches of the measurement setup. Instead of using message identifiers and data bytes, with the help of the database you can also work with message and signal names.
- The **CANalyzer main program** is used to measure and simulate CAN systems. You can associate one or more databases to any configuration from **File | Database**.

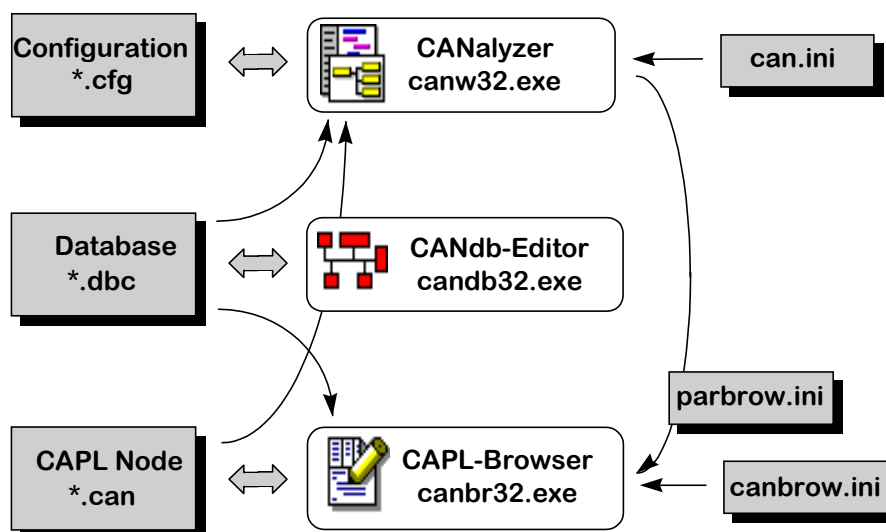


Figure 24: CANalyzer system overview

Start options for CANalyzer and Browser are provided in the linked INI files. If you are starting Browser from CANalyzer's measurement setup, a temporary file PARBROW.INI is automatically generated with the proper start options and is passed to Browser.

1.5 CANalyzer Architecture

In the course of a measurement the PC-card registers CAN messages on the bus and passes them to the measurement setup, and from there to the specified paths in the data flow plan and on to the evaluation and analysis blocks at the far right of the plan. During a measurement two program modules work closely together to this purpose: First the CANalyzer real-time library (`CANRT.DLL`) retrieves the information arriving at the card, provides them with a time stamp and shifts them to a ring buffer.

In a second step these data are read out by the actual main program (`CANW32.EXE`) and are evaluated in the function blocks on the right-hand side of the data flow plan.

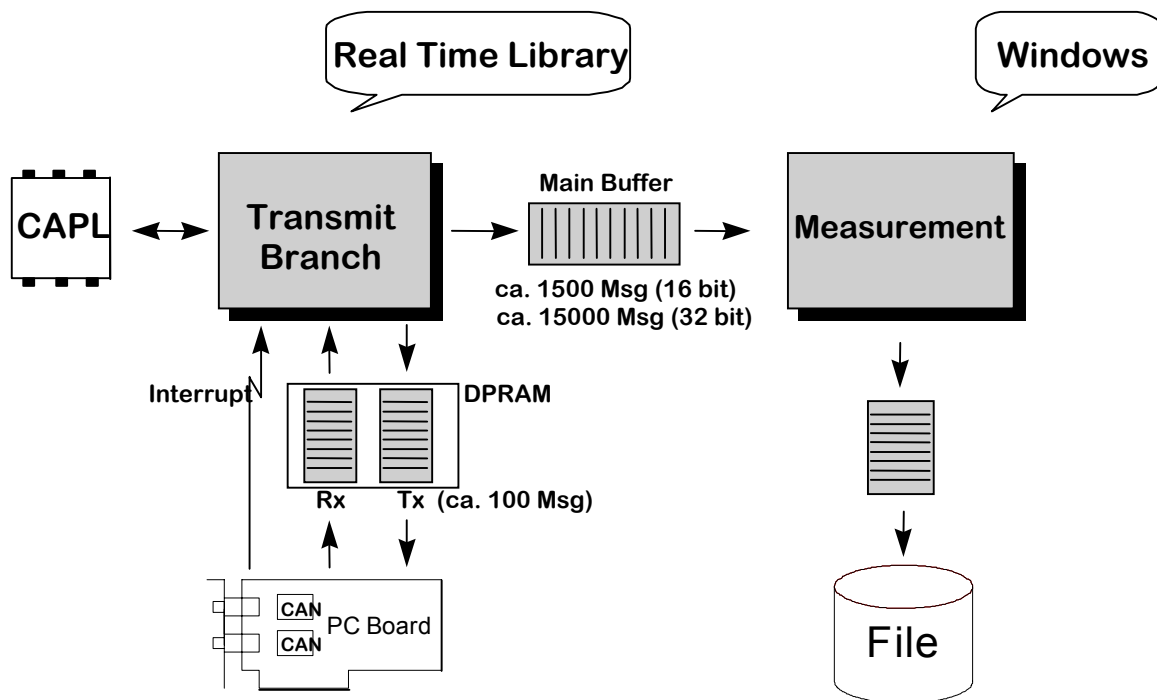


Figure 25: Internal structure of CANalyzer

You can influence the data flow in both program modules by inserting function blocks in the measurement setup. In this context the real-time module is comprised of the PC-card block, all function blocks between the card block and the transmit branch, and all blocks in the transmit branch itself. All other function blocks are used to configure the data flow in the evaluation branch.

If you insert blocks in the real-time library, i.e. CANalyzer's transmit branch/, you should be make sure that they do not demand too much computing time, so that system reaction times are not lengthened. Moreover, in CAPL programs you may only access files from here using special precautionary measures.

1.6 Particularities of the Demo Version

In the demo version of CANalyzer a demo driver which does not require a PC-card is connected to the PC instead of a regular PC-card driver. However, the functions of this driver are very limited. Primarily, it ensures that all messages which are transmitted are returned as received messages with a accurate time stamps.

Therefore, to be able to work with the demo version, a generator or program block which generates transmit messages must be inserted in the transmit branch. Messages generated in this way can then be captured, evaluated and saved.

The bus parameter options and message setup which are selected by clicking on the PC card icon in the measurement setup are irrelevant for the demo version and can be disregarded.

Aside from the PC card and the associated card driver, the demo version is a complete version. In particular, evaluation and memory storage of messages and CAPL programming can be tested without limitations.

2 Applications

CANalyzer provides you a set of significant basic functions for the work on various bus systems. Functions as loading and saving configurations, assigning databases and configuring panels, you call directly from items in the main menu. Particularly the data flow diagram and the function blocks in the measurement setup window are directly configured with context sensitive menus. Therefore you have to choose a block in the data flow diagram and click on it with the right mouse button to open the corresponding context menu. For example you can insert new function blocks such as filters or generator blocks at the black rectangular insertion points (hotspots) in the data flow or configure the PC card with the bus icon on the right of the simulation setup.

A brief look at the data flow in the measurement setup gives you an overview of the configuration options provided by CANalyzer and shows how your actual measurement configuration appears. Measurements and analysis are configured in the measurement setup window.

Program Start

At the program start of CANalyzer the program `CANW32.EXE.EXE` is called by double clicking the appropriate icon in the CANalyzer program group. CANalyzer can only operate trouble free if the system directory contains all necessary files and the hardware has been installed properly (compare enclosed instructions on hardware installation).

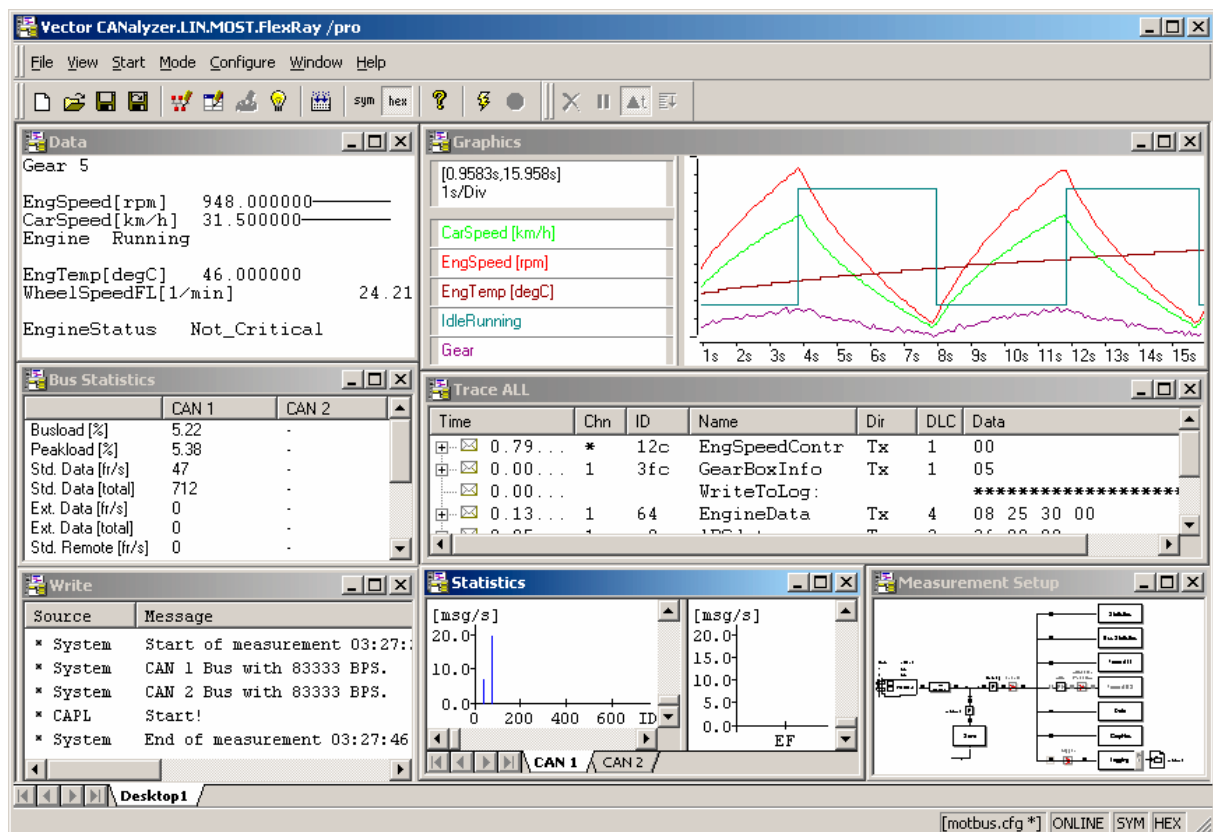


Figure 26: CANalyzer during a measurement run

At the program start CANalyzer first reads information on hardware settings, start paths, editors used, etc. from the project file `CAN.INI` in the system directory. Afterwards, a configuration file `*.CFG` is read in. This file, which contains all information on the currently active configuration of CANalyzer, is updated automatically after a prompt at each program stop.

You can specify a working directory in the program icon. To have this file loaded automatically at the start you can also enter the name of a configuration file in the command line for program names. This method can be used to configure CANalyzer differently at the start by using multiple icons. If no entries were made in the command line, the last opened configuration is automatically loaded.

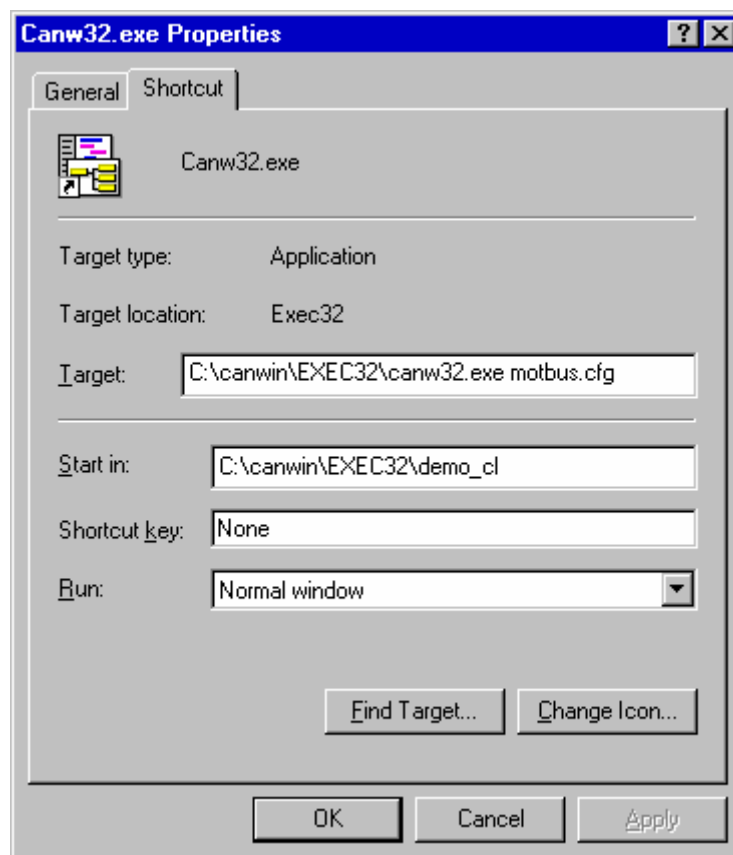


Figure 27: Automatic loading of the configuration `MOTBUS.CFG` at program start

The CANalyzer Screen

The CANalyzer screen consists of the main menu bar and the toolbar in the upper portion of the screen, the status bar at the bottom of the screen, and the data flow window and various measurement windows. You can gain access to all CANalyzer windows by double clicking the specific evaluation block in the measurement setup or by selecting the window from the **View** menu.

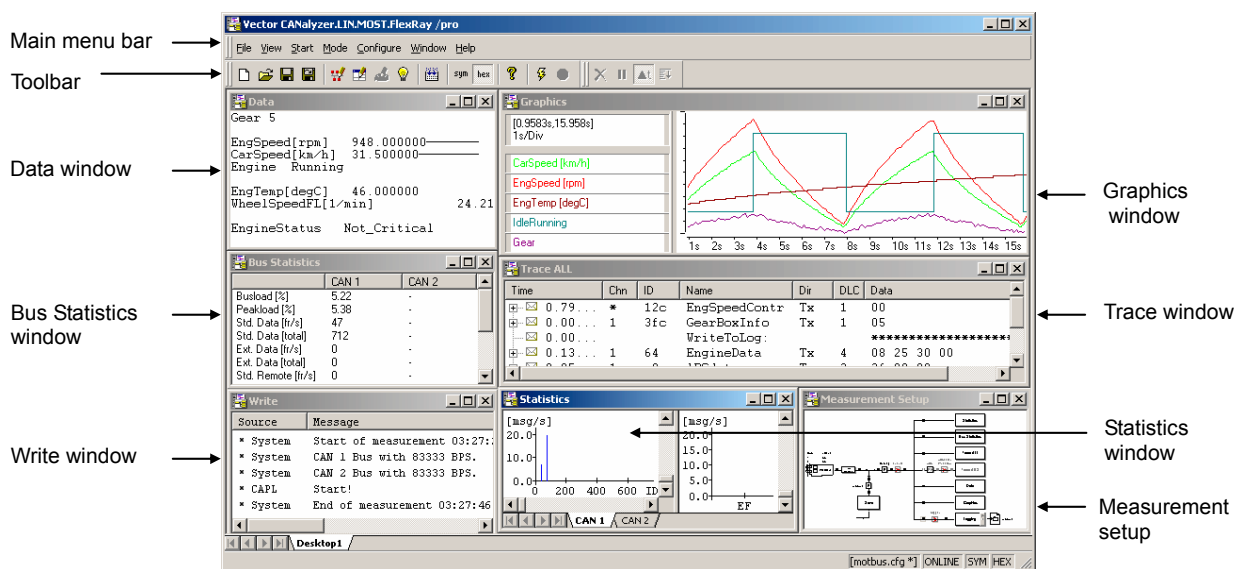


Figure 28: The CANalyzer screen


After selecting an entry the relevant window is activated and is displayed in the foreground.

Menu line	Used to select basic functions
Toolbar	Used for quick selection of important commands and also contains status indicators for the number system being used (decimal or hexadecimal) and to display of keyboard entries made during the ongoing measurement.
Measurement setup	The measurement setup displays the program's data flow. All options are set in this window for parameterizing a measurement or evaluation.
Trace window	Bus activities are recorded here. The user can scroll in this window after a measurement has been completed.
Statistics window	The mean transmit frequencies of messages are displayed as line spectra above the identifier axis in this window. As an option, the user can toggle over to mean transmit spacing. The window can be zoomed for detailed evaluations.
Data window	Preselected data segments of messages can be displayed here.
Graphics window	Graphic representation of signal time responses, which are displayed in a X-Y diagram above the time axis. After the end of measurement a measurement cursor and a difference cursor are provided, with which you can examine the coordinates of all measurement points or differences between two measurement points precisely.

Write window	Important information on the progress of the measurement can be output here (e.g. triggering of logging function). Furthermore, all outputs that the user places with the Write command in CAPL programs are written to this window.
Bus statistics window	Hardware-related information such as number of data and remote frames, error frames and bus load are displayed here. Availability of this information depends the CAN PC-card being used.
Status bar	The names of the active configuration file and the database being used are displayed here.


2.1 Measurement/Measurement Setup

2.1.1 Measurement Start

The measurement is started by pressing the F9 key, by choosing **Start | Start** in the main menu or by activating the start button  on the toolbar. In Online mode data can now be received and evaluated from the bus or can be sent out onto the bus. The data flow diagram shows you which analysis and transmit blocks are active during the particular measurement.

At the start of an Online measurement, first the CAN board is initialized. If this cannot be done, an error message is output and the measurement is terminated.

During the measurement the user can configure the Trace block, Data block and the scaling of the Statistics window and Data window. However, the menu items in the popup menus of the remaining blocks are masked out for the duration of a measurement. You cannot parameterize these blocks until after the measurement run has ended. All keyboard inputs during the measurement are passed through directly to the function blocks (CAPL programs, generator block, etc.). They are shown in the relevant status window on the toolbar. The only available program controls are the <Esc> key (terminate measurement) and all of the key combinations with the <Alt> key (Window control under Windows).

You can stop the measurement by pressing the <ESC> key, selecting the main menu item **Start | Stop** or activating the  button on the toolbar.

Internally, the measurement can be terminated by the elapse of the post-trigger time after triggering has occurred, or by the call of the `stop()` function from a CAPL program.

Note: During high system loads the stopping process may take a certain amount of time, since the system's message buffer must be emptied. A repeated stop command (double click) causes the buffered data to be ignored, and the measurement is terminated immediately, even under high system loading.

2.1.2 Working with Configurations

All options that you configure (configuration of the measurement windows, transmit branch, PC card, etc.) while working with CANalyzer can be saved to a configuration file with the extension `CFG`. Thus, you can work with different configurations to perform specific measurements and tests with CANalyzer.

To save changes of a specific configuration to a new configuration file choose the menu bar item **File|Save configuration as**. With the menu item **File|Load configuration** you can reload configuration files which you previously saved in CANalyzer. In the demo directory `DEMO_CAN_CL` you will find some prepared demo configurations that can serve as models when you start up CANalyzer and during the learning phase.

To obtain an overview of the files belonging to your project (configuration files, log files, CAPL programs, databases, , etc.) and to allow you to run them on another computer if necessary, it is advisable to create a separate project directory for each project (also called a working directory in Windows). Be sure to save all files resulting from your work in this directory. If you are working on several different CAN projects, multiple project directories are also advisable. With large projects it might even be easier to distribute the databases and configuration files of a project to different sub-directories.

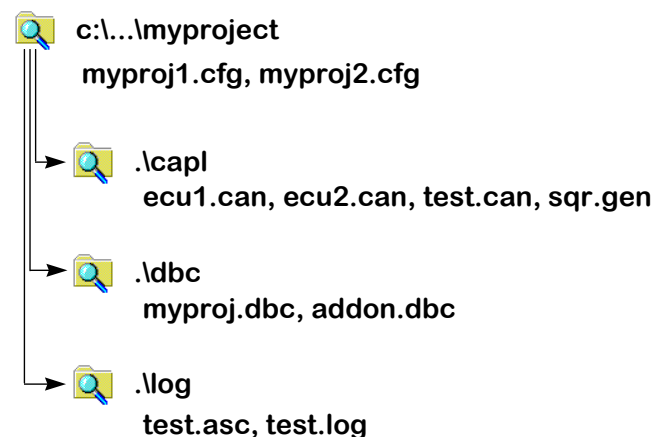


Figure 29: Example of a Directory Tree for a CANalyzer Project

References to other project files (e.g. to database files `*.DBC` or to CAPL programs `*.CAN`) are also saved in the configuration files. CANalyzer works internally with *relative paths*. That is, referenced files are searched and read-in relative to the configuration file during loading. Consequently, if you move your project directory with *all* associated files to another location in the directory tree or to another drive, the databases and CAPL programs referenced by the configuration file are found and correctly read-in at the next loading.

Note: To document or archive configurations, from the menu item **File|Files used** you can have a list generated of all files belonging to the active configuration (databases, CAPL programs, etc.) or have a ZIP archive generated.

The last configurations you worked with are saved in the [LastCANalyzerConfigurations] section of the CAN.INI file. The list of last opened configurations in the **File** menu allows you to access these configurations. If you do not specify a start configuration on the command line, the last edited configuration of this list is used at the program start.

Saving Configuration in Older Formats

Older CANalyzer versions cannot read the current configuration file format. However, if you still want to work with older CANalyzer versions you can save the configurations in formats compatible with those versions.

Select the desired version in the File Type list. However, please note that the older the selected format, the more configuration information that will be lost. Of course, the active loaded configuration remains unaffected by this.

Please refer to online Help to learn the most important differences between the versions.

Importing Configuration Descriptions

As an alternative to loading configuration files (*.CFG), CANalyzer offers you the option, via the menu item **File | Import**, of importing configuration descriptions (*.CIF). For the most part, the format of configuration descriptions corresponds to the format of INI files.

Configuration descriptions allow you to specify configurations directly in a configuration description file, and to make adaptations there, e.g. to add more Trace windows or change the paths of CAPL program files.

In Help you will find an example of the structure of a configuration description.

2.1.3 Representation Formats

The main menu item **Configuration | Options...** opens a dialog for entering the representation formats: Here you can select the numbering system (hex/dec) and decide whether you wish to have CAN messages displayed as identifiers or - provided that you have associated a database (cf. section 2.3) - whether you wish to have them displayed symbolically.

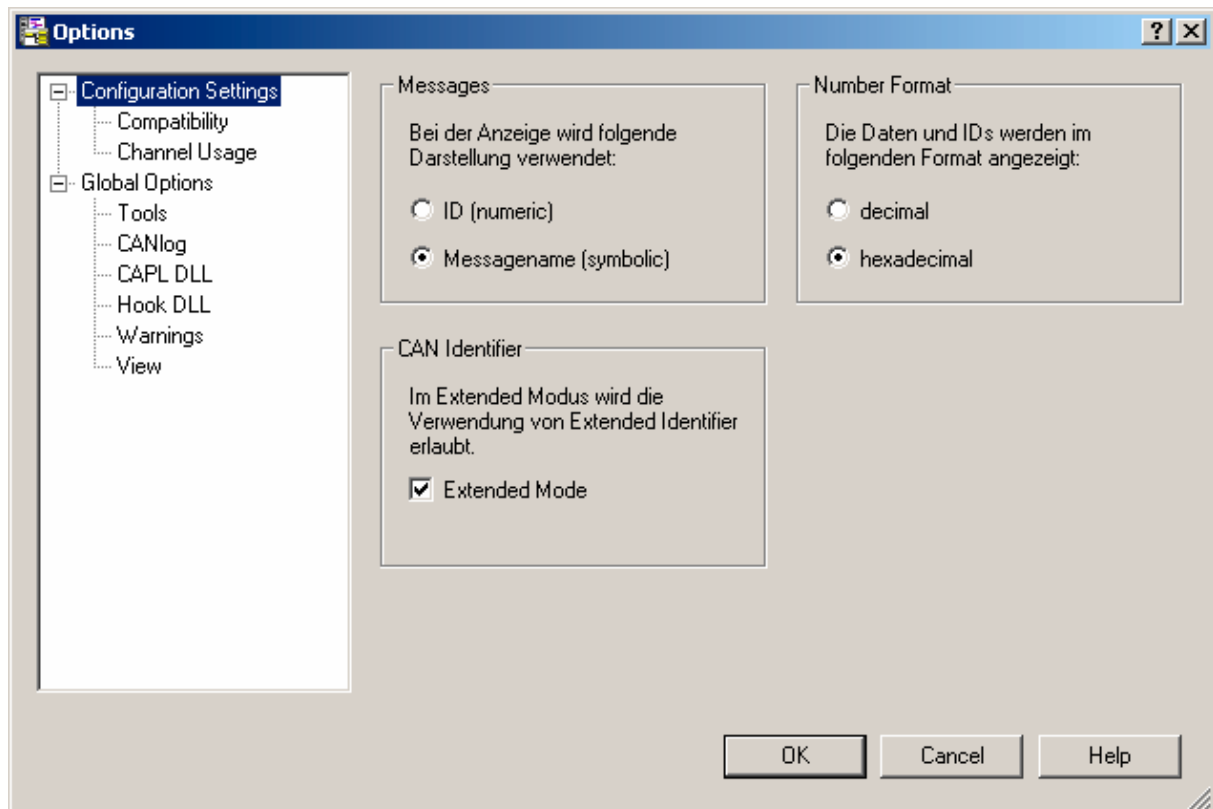


Figure 30: Dialog for configuring representation formats

These options affect the representation formats throughout the entire program.

Note: Please note that the numbering system in CAPL programs remains unaffected by these options. Identifiers with the prefix `0x` are interpreted as hexadecimal values, analogous to the C programming language. Otherwise the CAPL compiler always assumes that they are decimal numbers.

2.2 Transmitting and Receiving of Data

2.2.1 The Transmit Branch

On the left side of the measurement setup, CANalyzer's transmit branch branches off after the card icon. Here you can insert function blocks to feed data onto the bus. The data flow in the transmit branch always runs from top to bottom. The small arrows in the measurement setup indicate the direction of data flow.

The transmit branch is only accessible in Online mode, and there is has the task of forwarding messages that arrive at its input to the card driver as transmit tasks. The transmit block itself cannot be parameterized. When an attempt is made to select the block by mouse click or by keyboard, neither a menu nor a dialog box appears.

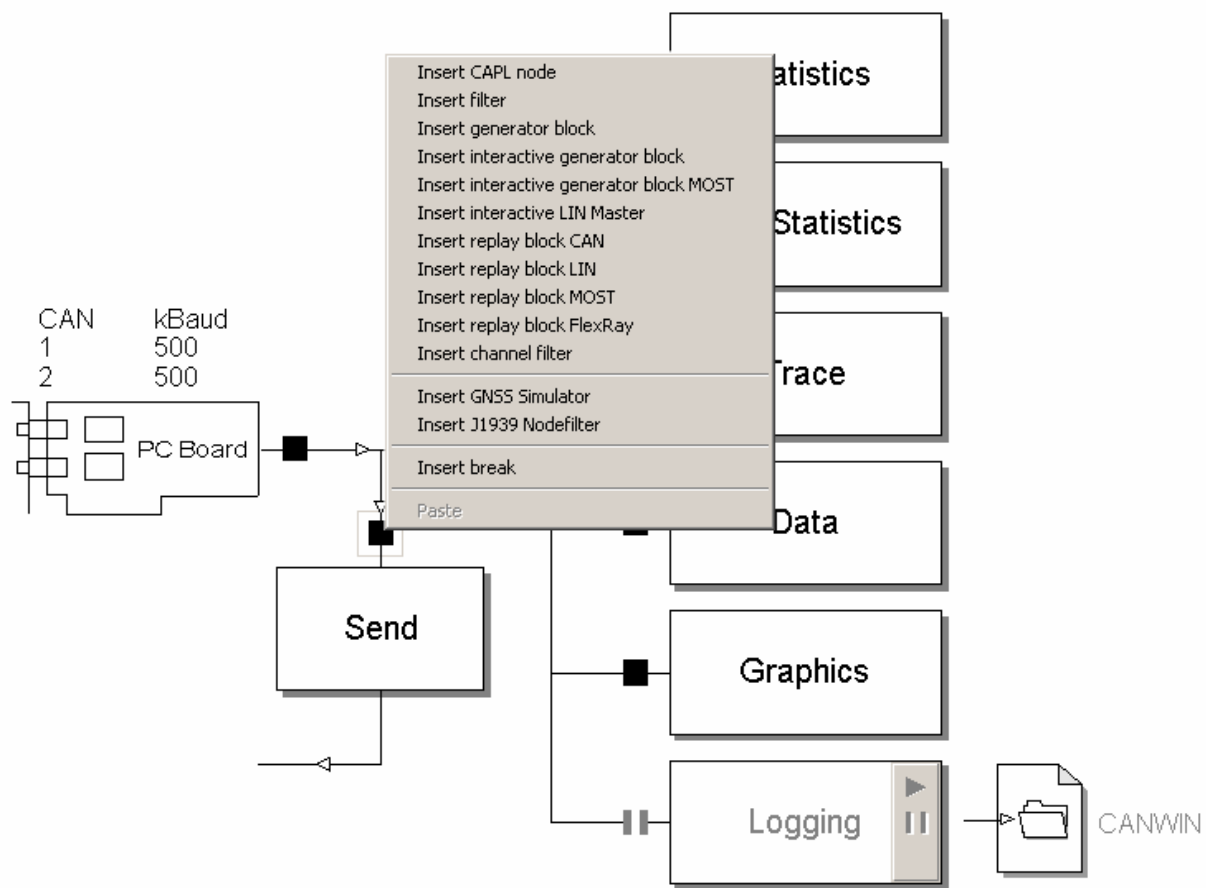


Figure 31: Inserting a generator block in the transmit branch

For typical applications, a program block, generator block or replay block is inserted at the hotspot before the transmit block. You can use this to specify what should be transmitted.

Note: When transmitting from CAPL programs, generator blocks and replay blocks, you can explicitly specify which of the two CAN controllers (bus connections) should be used to transmit the message. This is observed accordingly by the card driver.

2.2.2 The Evaluation Branches

In the evaluation branches of the measurement setup, data are passed from left to right to the measurement setup's evaluation blocks, where they can be visualized and analyzed with various functions.

Filters or user-defined analysis programs can be inserted in the data flow diagram before the evaluation blocks. As a result, the data flow can be configured in many ways to suit the particular task.

Each evaluation block has a measurement window in which the data arriving in the block are displayed. The functions of all measurement windows are described in detail in the sections below. Only the logging block is not assigned its own window. In-

stead, a log file is assigned to it for the purpose of logging bus data traffic and then studying it "offline" (cf. section 2.6.1).

2.2.3 Message Attributes

Messages that were not transmitted by CANalyzer's CAN PC-card (Receive messages), get the attribute Rx and a time stamp from the card's clock when they are received. Afterwards they are passed to CANalyzer via the card driver and, finally, they are shown in the evaluation windows. The time stamp and Rx message attribute can be seen in the Trace window.

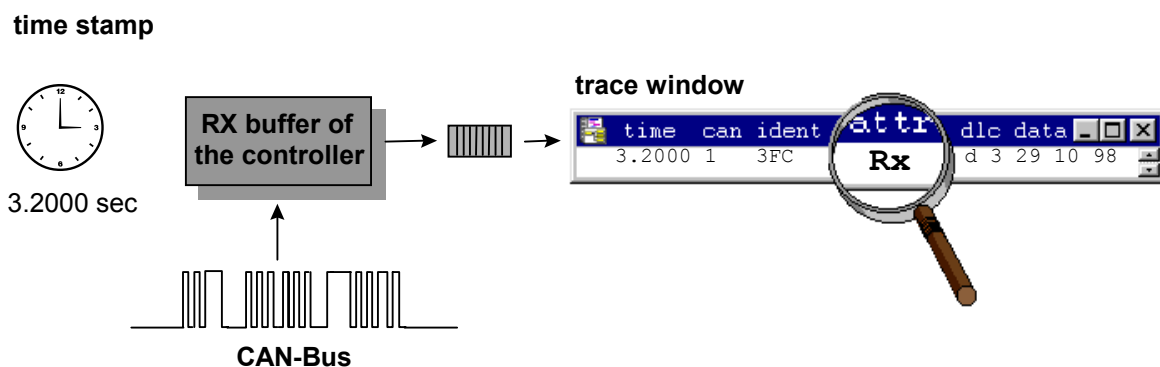


Figure 32: Receiving messages

The messages to be transmitted are passed from the transmit block via the card driver to the CAN PC-card. If your hardware supports the card and driver option **Activate TxRq** in the **Options** item of the PC-card icon's popup menu, and you have activated this, the driver returns the time of the transmit request assigned to the CAN microcontroller to you. In the Trace window, for example, you would see the message to be transmitted with the attribute *TxRq*.

After successful transmission the message is returned with the actual time of transmission and the attribute *Tx*, so that the transmit messages can be displayed and/or logged in the Trace window. If these messages reach the transmit block directly again, they are not retransmitted. This prevents unintentionally forming infinite loops which might severely load the bus under certain conditions. If an infinite loop is to be programmed intentionally (e.g. as a base load) this can be done by recopying the message in a program block.

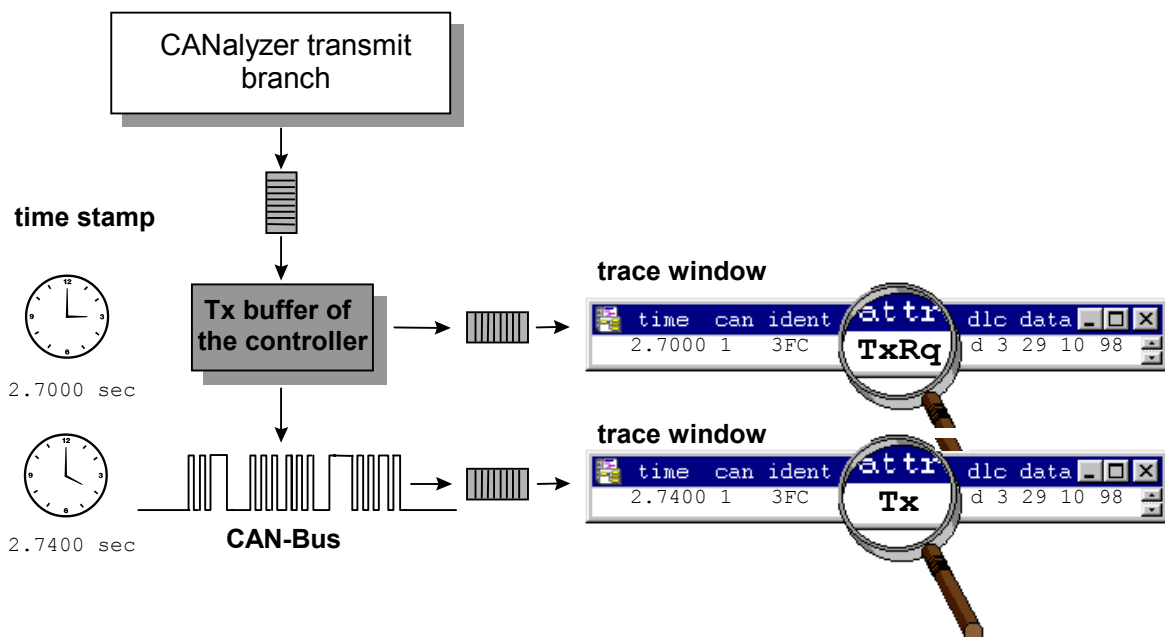


Figure 33: Transmission of messages

The *TxRq* display permits measurements of the difference between the time of transmit request and time of transmission. The time between the message with Tx attribute and TxRq attribute is essentially the transmission time of a message, i.e. the time that the CAN controller needs to place a message completely on the bus. It is a function of baud rate and message length. The transmission time also grows as a result of lost arbitration events, which can be observed more for low-priority messages at high bus loads.

Since the (very small) latency time of the card driver interrupt must be added to the transmission time, the following general formula applies:

$$t_{Tx} - t_{TxRq} = \text{Transmission time} + \text{Latency time}$$

Note: Under high load conditions the display of messages might be delayed in the evaluation windows under some circumstances. However, the time stamps for the messages remain unaffected by this, since they are already assigned to the messages when they are received on the card.

2.3 Use of Databases

When performing large-scale studies on the CAN bus, it is a great help to the user if - in addition to the bus-oriented raw data format with numerical identifiers and data contents - a symbolic interpretation of the message event is also provided.

CANalyzer supports the use of symbolic databases. You can make this information available by associating one or more databases to the active configuration (Menu item **File | Associate database**). Afterwards you can access the information in measurement windows, insertable function blocks and CAPL programs.

2.3.1 Creating and Modifying Databases

The database management program CANdb++ is available to you for inputting and modifying databases. It is included with the standard CANalyzer product.

In a database, names are assigned to CAN messages. In CANalyzer you can then address the messages using these names. For example, the clear text *EngineData* is shown in the Trace window instead of the identifier *100*.

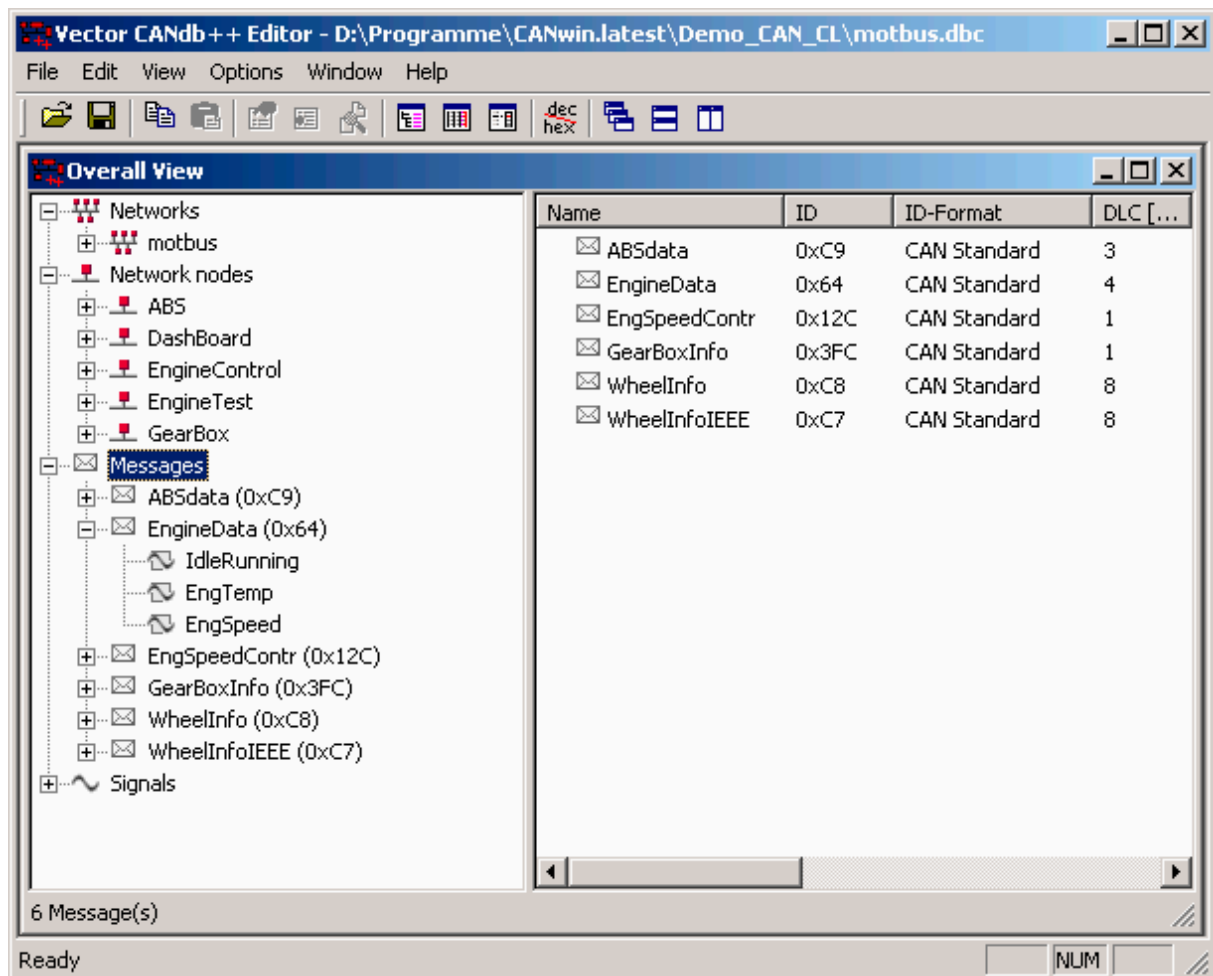


Figure 34: Symbolic description of the message *EngineData* in CANdb++

Moreover, so-called signals are defined in the database. A signal is a symbolic description of a data segment within a message. Besides identifying the data segment, the signal definition also incorporates characteristics such as machine format (Motorola/Intel), sign handling, a conversion formula and physical unit of measurement. This permits direct display of physical dimensions in the data window, such as:

"Speed = 810.4 rpm".

Please refer to CANdb++ Editor documentation or CANdb++ Editor online Help for further details on creating and modifying a database.

2.3.2 Access to Database Information

Besides the individual text input boxes, in general there are also small buttons for the purpose of entering symbolic message or signal names in function blocks. When you press one of these buttons you start the Message Explorer with all symbols defined in the database. You can choose one or (in the case of the filter configuration dialog) multiple names from the Message Explorer.

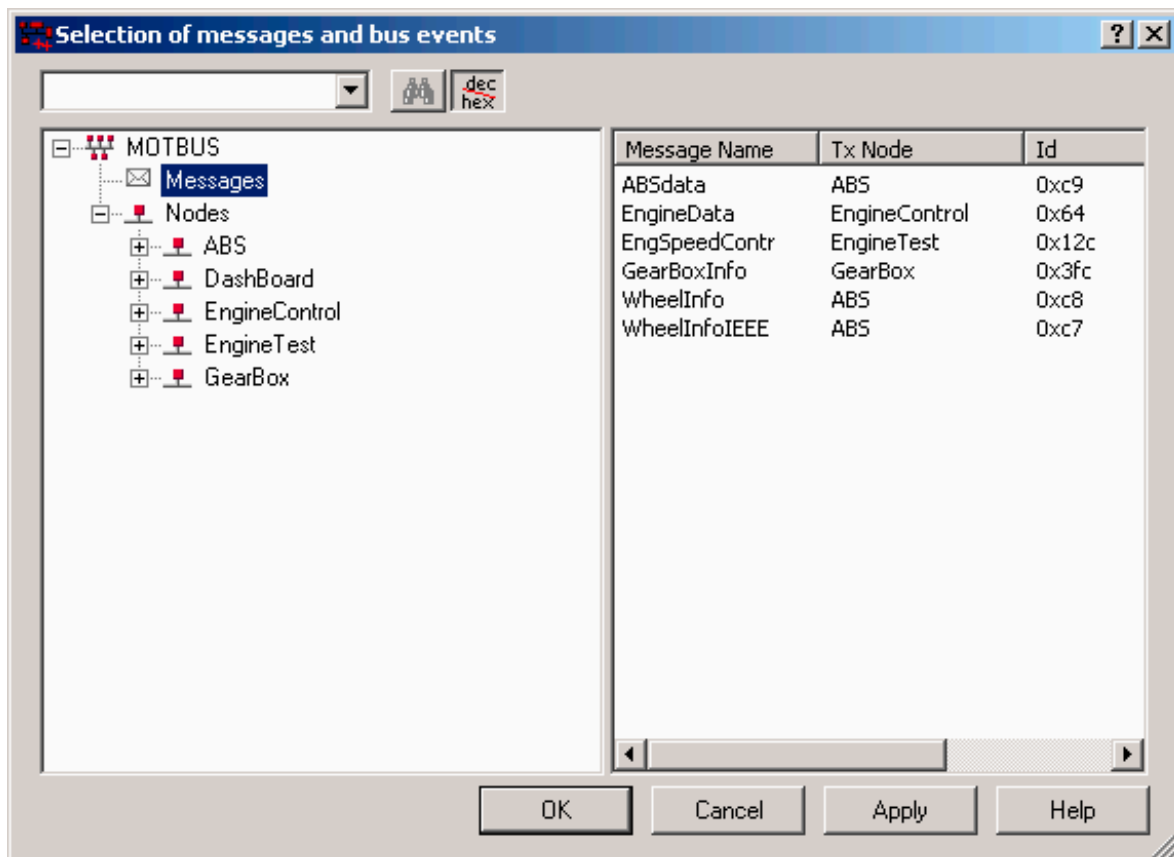


Figure 35: Message and Signal Explorer

If the database has been filled out completely, you can search in this Explorer for e.g. a node, a message or even a list of all signals in the database.

Please refer to online Help for further instructions on using the Message Explorer and Signal Explorer.

2.3.3 Associating the Database

In the database selection dialog opened by the menu command **File | Associate database**, you define the databases with which you want to work.

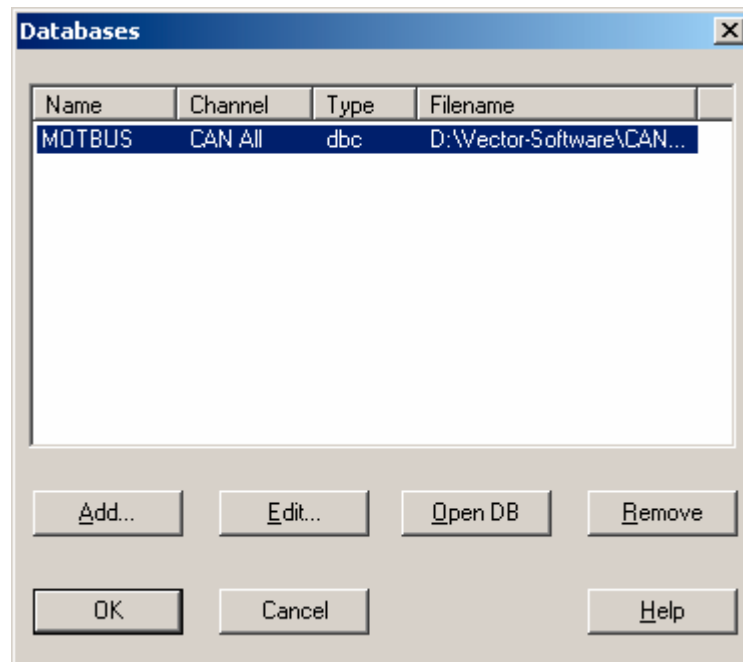


Figure 36: Database selection dialog

Under **[Add...]** you can associate new databases in a file selection dialog. If no line is highlighted in the list of databases, the new database is added at the end. Otherwise it is added above the highlighted line. You can also highlight the empty last line to have the new database inserted at the end of the list.

In the database list, the first column shows the database name which, for example, might be used to resolve ambiguities as a name qualifier. In the second column are the CAN chips assigned to the database. In the last column you will find the complete file name for the database. The sequence of databases in the list is utilized to resolve ambiguities of symbolic names and message identifiers.

By activating the **[Edit...]** button you arrive at the **Edit Database** dialog. Here you can define the database name as well as the assignment of the database to a CAN chip.

The name of the database is determined as follows. If the database includes the attribute DBName, then its value is used as the name. If the attribute does not exist, the name is derived from the file name. The user can overwrite the name. Database names must begin with a character and may not contain any blank or special characters.

The database name must be unique. If a name that is automatically determined already exists, it is assigned a unique name according to the sequence of names CANdb1, CANdb2, etc.

You can assign both CAN controllers (CAN1 and CAN2) to exactly one database. All other databases are assigned to the two controllers jointly. This assignment defines the default selector for message definitions in CAPL.

2.3.4 Use of Multiple Databases

For large systems it may be sensible to distribute the descriptions of messages and signals to several partial databases. When operating CANalyzer on two buses it also makes sense to describe each system by its own database.

CANalyzer supports the simultaneous use of multiple databases. You can configure the databases that you would like to associate with CANalyzer using the menu command **File | Associate database**. Afterwards you can use the symbolic names for messages and signals from all databases in all function blocks and in CAPL. To do this, you would simply enter the symbolic name in the appropriate input box. There is a list of all symbolic names in the signal selection dialogs which you can open by activating the small buttons next to the individual input boxes. You would select the desired symbolic names there.

If you are using more than one database, the messages in the databases that follow the first database are qualified with the database name, i.e. the message name is preceded by the database name followed by two colons. However, you will only need to use these qualified names to resolve ambiguities. As long as the symbolic names are unique among all databases, you can forego qualification of symbolic names in all function blocks and when editing CAPL programs.

2.3.5 Resolving Ambiguities

When multiple databases are used, it is essentially possible to have ambiguities in the use of symbolic names. These ambiguities must then be resolved by the program. On the one hand, messages arriving from the bus via one of the two CAN controllers and logged by the program can have different symbolic names in two databases. On the other hand, the user may wish to configure function blocks or measurement windows with different messages that have the same name in different databases.

Ambiguities of the first type are resolved by the sequence in which you have entered the databases in the list of the database selection dialog. Furthermore, you have the option of associating a prioritized database to each of the two CAN controllers. For messages received by this controller, this database then has highest priority in the symbolic association. Only if the symbolic name is not found there, does the program search all other databases indicated in the database selection dialog, and this is done in the sequence defined there.

The search sequence in the database list in the database selection dialog is also used to resolve name conflicts when configuring measurement windows and function blocks. In such a case, the name is associated with the message in the database that is highest in the list. However, you can also resolve ambiguities of this type by qualifying symbolic names.

See CANalyzer's online Help function for examples of resolving ambiguities.

2.3.6 Checking for Consistency of Symbolic Data

In CANalyzer's symbolic mode the CAN messages are addressed using symbolic names from an associated database. Therefore, CANalyzer checks the consistency of the database and the active configuration in the following situations:

- At the program start,
- When a new database is associated,
- When CANalyzer is restarted after a change to the database.

In this consistency check, the symbolic names of all CAN messages used in the measurement and simulation setups are compared to names in the database. If the message name has been changed in the database, an automatic adaptation is made to this name, and an appropriate message appears on the screen. If CANalyzer could find neither the name nor the message ID in the database, the user receives an error message. In this case the measurement cannot be started until the particular message is removed from the configuration.

2.4 Working with Multiple Channels

CANalyzer supports up to 32 (virtual and real) CAN channels. Consequently, you can also use multiple CAN cards to transmit and receive messages. This section describes what must be taken into consideration when working with multiple channels.

2.4.1 Channel Definition

The number of CAN channels you wish to use is configured in the Channel Definition Dialog.

You can access this dialog from the main menu item **Configure | CAN channels...** on the menu bar or from the PC card icon in the measurement setup. In addition to affecting the measurement itself, the channel definition also affects the inputs that are possible in the various configuration dialogs. Only defined channels are offered for selection.

The channels are allocated to the CAN chips registered in the CAN hardware configuration of your computer's Control Panel. Chip allocation is only meaningful in Online mode. In Offline mode, in which messages are replayed from a file, it is irrelevant.

To change the default chip allocation, open the **CAN Hardware** component of your computer's Control Panel. The CAN Hardware Configuration Dialog appears, in which you can modify the chip allocations of the channels.

The chip allocation is also shown in the Hardware Configuration Dialog (cf. section 2.10.1).

Note:	The number of channels is configuration-specific. It is saved in the configuration file and is restored when loading the configuration.
--------------	---

2.4.2 Channels in Online Mode

In Online mode messages from the transmit branch are transmitted on one or more real buses, and in the measurement setup they are received by one or more real buses. The defined channels correspond to these real buses with their controllers.

If you specify more than 2 CAN channels the following conditions must be satisfied to be able to start a measurement:

- Your active CAN driver must support more than 2 CAN channels. If this is not the case, you will receive a warning if you specify more than 2 CAN channels.
- A real or virtual CAN controller must be assigned to each channel. If this is not the case, you will be requested to make such an assignment.

In the Channel Definition Dialog you can choose whether or not a consistency check should be performed after configuration. The consistency check covers database assignments and the configuration of all function blocks with the exception of CAPL blocks. The check monitors whether invalid channels are referenced. If this is the case an inconsistency is reported. These reports can be output to the Write window if desired.

With CAPL blocks a determination of whether all referenced channels are valid is not made until compilation. A warning is output if any channels are invalid. Therefore, it is advisable to recompile all nodes after each new definition of channels.

If you use undefined channels, CANalyzer behaves as follows in Online mode:

- Channel configuration does *not* cause any filtering of messages in the data flow plan.
- When *receiving* on controllers not assigned to a defined channel, the received messages are passed through the measurement setup.
- When *transmitting* from a Generator block or Replay block on the right of the transmit branch to an undefined channel, the transmit request is similarly passed through.
- An error is reported in the Write window for a sender in the transmit branch as soon as the transmit request is given to an undefined channel. The message is not transmitted.
- CAPL blocks do not transmit messages to which an undefined channel is assigned.

2.4.3 Channels in Offline Mode

In Offline mode the channels correspond to those channels on which the played-in messages were logged.

Consequently, each message is played-in on the channel on which it was logged. The channels in Offline mode correspond to the channels used during logging to the log file. Therefore, you should define the number of channels such that it corresponds to the number of channels that were configured for logging to the log file.

If you use undefined channels CANalyzer behaves as follows in Offline mode:

- The channel configuration does *not* cause any filtering of messages in the data flow plan.

- If messages are played-in which are assigned to an undefined channel, these messages are passed through the measurement setup.
- When *transmitting* from a Generator block or Replay block on the right of the transmit branch to an undefined channel, the transmit request is passed through.

CAPL blocks do not send out messages to which no defined channel is allocated.

2.5 CANalyzer in Load and Overload Operation

2.5.1 Behavior in Load Situations

At high bus loading the computing power of your PC may be insufficient - under some circumstances - to simultaneously operate the more complex evaluation and display functions (Statistics window with statistical report, Data and Graphic window with many signals, Trace window in the chronological output mode, etc.). To prevent an impending data loss the program therefore has mechanisms for detecting and handling load situations.

If the rate of messages registered by the card is so high that CANalyzer cannot follow along with processing any longer, the ring buffer between the real-time library and the CANalyzer- application runs full. CANalyzer detects this when a **High water mark** limit is exceeded (cf. Figure 37), and it automatically switches over to a load mode in which display functions are reduced to provide more computing time for internal data processing.

When the **high-load limit** of the ring buffer is exceeded, the display of messages in the Trace window is interrupted briefly under load operation to provide other analysis blocks with more computing power. However, it is possible that not all messages will be displayed in the window any longer during the measurement. You can recognize this load situation during the measurement by an exclamation point (!) in the first column of the window. Although not all messages are shown, no data are lost. After you stop the measurement, the entire set of information is available to you in the Trace window, Graphic window and in logging.

2.5.2 Behavior with Data Loss

If the ring buffer overruns in spite of these measures, as a user you are immediately informed of this data loss:

An occurring data loss is registered in ASCII logging. The “*” symbol appears in the line after which the data loss occurred. In the configuration dialog for the Log file the **Lost Data Message Box** option also allows you to have a data loss shown in a separate message window at the end of measurement.

The Bus Statistics window also indicates to you a data loss during an overload situation with the “@” symbol. However, please note that the display of bus load and received messages continue to function properly, since this information is provided by the interface card and does not need to be computed by the main program first. Consequently, the Bus Statistics display allows you to estimate the extent of the data loss.

2.5.3 Fill Level Indicator

To allow you to observe the ring buffer between the real-time library and the main Windows program more precisely, the program has a fill level indicator.

You can view the fill level during the measurement in the left corner of the status bar. If the ratio between arriving and processed events is balanced, the indicator has a green color. However, if it appears in red, significantly more events are arriving than can be processed at the given time. This is a clear indication that the system is overloaded. If the queue fill level reaches an alarming limit, the system attempts to relieve the load by selectively deactivating individual evaluation windows. If desired, you can have the queue's state displayed with the help of the entry `ShowMainQueue=0` in the `[System]` section of the `CAN.INI` file.

You can also have the fill level displayed in the Write window. To do this, set a minimum value of `WriteLevel = 3` in the `[Environment]` section of the `CAN.INI` file. Then, when the ring buffer overflows the Write window shows the report "Load transition: NORMAL->DATA LOST" and informs you of the data loss. After the overload situation has ended (e.g. after a brief burst on the bus) you are similarly informed as soon as a normal situation has been restored. You can recognize this in the Write window by the report "Load transition: QUEUE HIGH -> NORMAL".

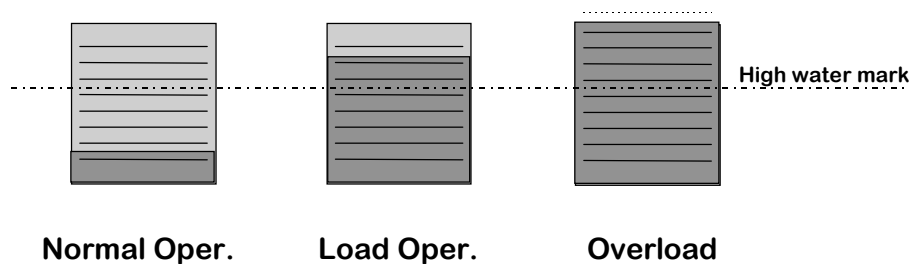


Figure 37: Fill Level of the Ring Buffer

For testing purposes you can provoke overload situations yourself by taking hold of the title bar of the main window with the mouse or by moving the window on the screen. This will interrupt the main program's data display until you release the title bar again. The work of the real-time library, however, remains unaffected by these actions. If messages are being registered on the bus, the ring buffer will be filled without the data being able to be processed by the main program, and as a result the queue will overrun. As soon as you release the title bar you can observe the effects of this overrun in the individual windows.

2.5.4 Optimizing Performance

To make it easier for you to configure CANalyzer for high bus loads, a performance wizard is provided under the menu item **Help | Performance | Optimization**. This wizard performs a two-stage performance optimization. A report is generated in the Write window listing the configuration options utilized in the optimization as well as their previous and new values. You can undo the changes made at any time.

The wizard optimizes the following options:

- Output mode and update cycle for the Trace window

- Update cycle for bus statistics
- Trigger option **Write full buffer to file** (Logging)
- Buffer size for the logging trigger
- Update mode and update cycle for the Data window
- Buffer size and drawing mode for the Graphic window.

The wizard function **Help | Performance | Undo** cancels the changes of a previous optimization of configuration options. The operations required for this are also summarized in a report.

Please note that only those values can be restored which have not been changed in the configuration dialogs since the optimization.

2.5.5 Configuration Options at High Bus Load

Besides the automatic deactivation function, which CANalyzer initiates in load situations, you can manually switch the following analysis blocks to less computing-intensive modes:

Trace Window

In the Trace window choose the output mode **Fixed position for cyclic update**. As a result, the window contents are no longer updated with each new arriving message, rather they are only updated cyclically. If necessary select a longer update time.

Data Window

If you have configured many signals in the Data window, you should choose the cyclic refresh mode here (**Configure Timer** entry in the Data window's popup menu) and enter a cycle time of maximum 500 milliseconds. The window is then only updated cyclically which saves on computing resources.

Graphic Window

If you have configured many signals in the Graphic window, you should choose a relatively large user-defined refresh (200 ms to 2 s) in the Measurement Setup Dialog (**Options** item in the popup menu) . This defines how often the graphic window display should be updated. Small values result in a fluid representation of the signal response, but they place high demands on computer resources, and with slower computers this might lead to performance problems. High values, on the other hand, reduce computing demands but lead to a more disturbed and delayed display of the signal response.

Statistics Window

Deactivate the statistics report in the Statistics window's popup menu to save on computing power. Furthermore, the averaging time can be selected in the configuration dialog of the Statistics block. Short time intervals place high demands on computer resources and might result in severely oscillating lines in the window. Very long averaging times make the display slower, but also less computing intensive.

If you insert blocks in the real-time library, i.e. in CANalyzer's transmit branch, you should ensure that they do not demand too much computing time, so that they do not increase system reaction times. Furthermore, you may only access files in CAPL programs if you observe special preventive measures.

On the other hand, it may be advisable at high bus loading to perform a data reduction as early as in the real-time library to relieve loading in the evaluation branches of CANalyzer.

It is not possible to predict an optimal configuration of the measurement setup for all situations. Cyclic updating indeed saves computing time, but also leads to poorer representation of the information. Under some circumstances it may be advisable to completely disconnect analysis branches that are not needed in the measurement setup (**Insert Break** item in the hot-spot's popup menu) or reduce the volume of data at the input to the measurement setup using filter functions. Moreover, you might also try to insert individual CAPL programs between the real-time library and the evaluation branches and observe the behavior of the program during another measurement run.

To filter out certain messages from the measurement setup, pass filters and blocking filters are provided as insertable function blocks. Furthermore, the supported PC cards with acceptance filtering (**Messages** item in the popup menu of the card icon in the measurement setup) also offer you the option of filtering out certain messages in hardware, thereby relieving both the real-time library and the main Windows program of the need to evaluate unnecessary information.

2.6 Logging and Evaluation of Measurement Files

CANalyzer offers you the option of saving the CAN data traffic in a log file, so that you can evaluate it later in Offline mode.

Logging blocks are provided to you for this purpose. The task of a logging block is to store data arriving at its input to a file. You can configure the log file in the measurement setup by the file symbol icon at the far right in the logging branch.

Each logging block is equipped with user-friendly triggering to reduce the amount of data as much as possible even before acquisition. This permits the formulation of a trigger condition, and data are only saved near the time of the trigger. During each measurement multiple triggers can be initiated for various events, whereby the user can prescribe pre-trigger and post-trigger times. The trigger condition is user-programmable. You can configure triggering in the measurement setup via the *Logging* function block.

CANalyzer has an Offline mode for analyzing log files. In contrast to Online mode the data source here is a file, e.g. a file created by logging in Online mode. All measurement and evaluation functions of Online mode are also available to you in Offline mode.

2.6.1 Logging Trigger

You can open the configuration dialog for the logging trigger by clicking the logging block in the data flow plan or by selecting **Configuration** in the context menu.

Conditions can be formulated for triggering logging and for interrupting an Offline measurement. These conditions are always input in the same dialog.

Whenever you activate the logging function the trigger block is activated as well, automatically. Additionally you can place the trigger block anywhere (hot spot) in the analysis branch of the measurement window.

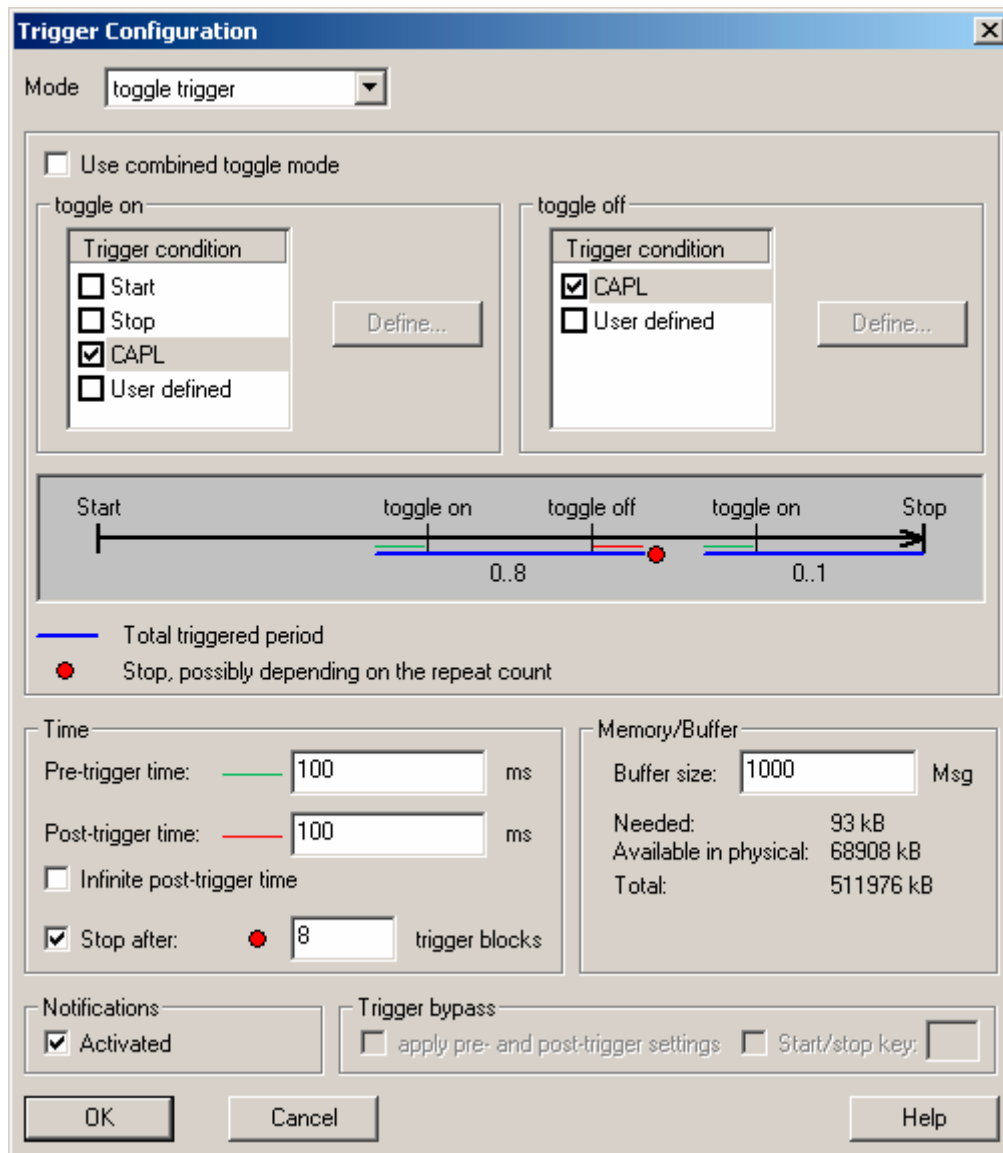


Figure 38: Trigger Configuration Dialog

2.6.1.1 Trigger Mode

The trigger mode defines the general conditions for a logging (Start point, end point, logging time period). You can select from three trigger modes:

- **Single Trigger**
a certain event triggers logging

- **Toggle Trigger**
certain events define the start and stop of logging
- **Entire Measurement**
the entire measurement is logged

2.6.1.2 Trigger Condition

If you have selected **Single** Trigger or **Toggle** Trigger as the trigger mode, you can now select from the following trigger conditions:

- **Start**
triggering occurs at the start of measurement
- **Stop**
triggering occurs at the measurement stop
- **CAPL**
triggering is by a CAPL program
- **User defined**
the occurrence of a user defined condition initiates triggering

You can select none, one or several of these trigger types.

The **[Define...]** button does not become active unless the **user defined** trigger condition is selected.

When you have chosen **Toggle** Trigger you can additionally choose whether to use the same trigger condition for block start and block end trigger or not. If you want to have the same trigger condition for both, activate the checkbox **Use combined toggle mode**.

2.6.1.3 Set of user defined conditions

Here you can input the set of conditions used to search or trigger. This set can also be used as a global breakpoint in Offline mode.

The set is organized in a tree form. Its node items are **Groups** and leaf items are **Conditions**.

- **Groups** are represented by their logical operators (**AND/OR**)
- **Conditions** are represented by a short textual description of the configured data.

The root of the tree is a predefined Group that cannot be removed.

Groups

Each group specifies a logical operator, which is applied to its immediate children items. The available operators are **AND** and **OR**.

Conditions

Each user-defined condition allows specifying

- a required run-time event (e.g. message, signal, environment variable),
- its timing (optional) and
- its required data (if applicable).

The conditions are classified by a category of the required run-time event and by a way the event can be specified (symbolic or raw form).

The available types of conditions are:

- Symbolic message condition
Specifies a message from any of the associated databases.
- Symbolic signal condition
Specifies a value of a signal. The signal should be chosen from one of the associated databases. Only signals assigned to a message (mapped signals) are allowed.
- Symbolic predefined signal condition
Specifies a value of a predefined signal. The signal should be chosen from messages that are defined in one of the associated databases.
- Predefined bus event condition
Specifies a predefined bus event (e.g. CAN Error frame)
- Bus statistic signal condition
Specifies a value of a bus statistic signal. The list of available statistic signals is the same as in the bus statistics window.
- Raw message condition
Specifies a message and its data bytes. The definition should be entered manually. As a first step the Raw condition pre-selection dialog is opened. There you can select a desired symbolic data or enter some basic data in the case of raw messages.

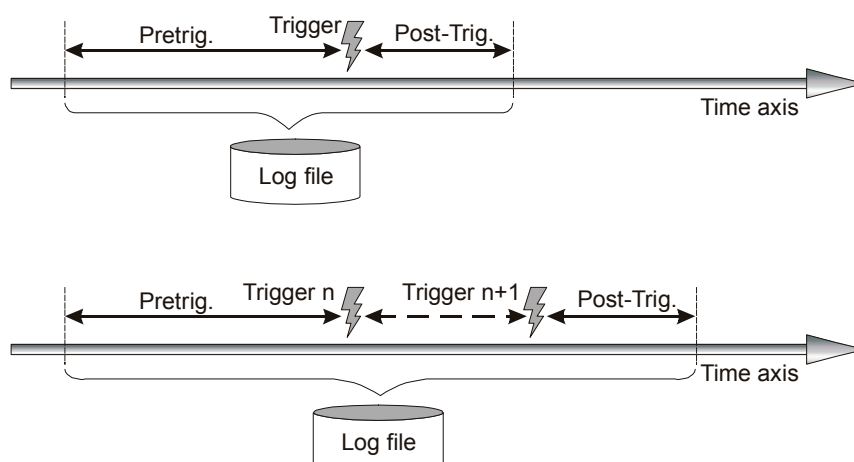


Figure 39: Time Window for the Trigger in Single Trigger Mode (top) and in Toggle Trigger Mode (Bottom)

In the **Single Trigger** trigger mode all those data occurring before and after a specific trigger is logged. You can enter the necessary settings for pre-trigger and post-trigger times, and the number of triggers you wish to log, in the **Time** area.

In **Toggle Trigger** trigger mode the time window is described by two successive triggers (start-of-block and end-of-block triggers). The first trigger activated during measurement is the start-of-block trigger, and the second is the end-of-block trigger. Afterwards, another start-of-block trigger follows, etc. The pre-trigger time in toggle trigger mode is referenced to the start-of-block trigger, and the post-trigger time is referenced to the end-of-block trigger. With the check box **Use combined toggle mode** you define, that for start-of-block and end-of-block trigger the same trigger conditions are valid.

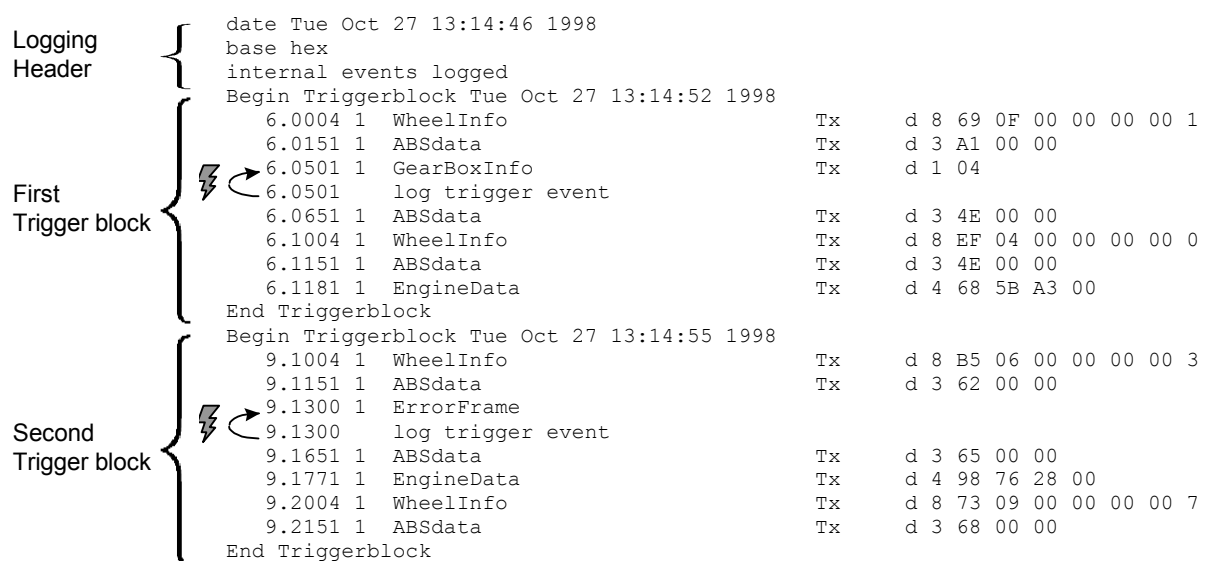


Figure 40: Log file with 2 trigger blocks. Pre-trigger: 50ms, post-trigger: 100ms, Trigger types: Message *GearBoxInfo* and Error Frames

For example, the entire measurement can be recorded in **Toggle-Trigger** mode by selecting Start and Stop as the trigger conditions. In this case, a start-of-block trigger is activated at the start of measurement, and the measurement is logged until the end-of-block trigger occurs when the measurement is stopped. Pre-trigger and post-trigger times are ignored when this option is set.

You can determine whether the trigger action should be executed once or multiple times. If the option **Measurement stop after n triggers** is selected in the trigger's configuration dialog, the measurement process is stopped after the post-trigger time of the nth trigger has elapsed.

2.6.1.4 Trigger Events

You can define trigger events using the five check boxes in the configuration dialog. The following types are available to you:

The following types are available:

- Trigger on **Start**. Triggering occurs unconditionally and immediately at the start of measurement. The logging time period is defined exclusively by the post-trigger time in this case.
- Trigger on **Stop**. Triggering is executed at the end of the measurement. Triggering can be activated by the CAPL function `stop()` or by the <Esc> key. The logging time period is defined exclusively by the pre-trigger time in this case.
- Trigger by a **CAPL** program: Using CAPL programming methods arbitrarily complex conditions can be formulated, which can depend on the occurrence of various events. Triggering is activated when the intrinsic function `trigger()` is called by a procedure. The configured pretrigger and post-trigger times define the logging time period for this trigger.
- Trigger when certain messages occur. With the **Event** trigger type, triggering occurs when specific messages with specific attributes occur. To define these attributes press the **[Condition]** button. You can find a description of the trigger conditions in chapter 2.7.5. The configured pretrigger and post-trigger times define the logging time period for this trigger.
- Trigger on **Error** frames. Triggering is executed whenever an error frame occurs. The configured pretrigger and post-trigger times define the logging time period for this trigger.

2.6.1.5 Time Window

In the configuration dialog you also define the time window (the pre-trigger and post-trigger times in milliseconds) around each trigger.

The pre-trigger time defines the time interval before activation of the trigger condition which is to be saved. If the trigger condition occurs so early that the pre-trigger time is greater than the time since the start of the measurement, only those data occurring prior to the trigger can be saved. Valid pre-trigger times must lie in the range of 0 to 180000 ms.

The post-trigger time indicates how long data continue to be acquired and saved after the trigger condition occurs. The measurement is terminated automatically after the post-trigger time has elapsed. Valid post-trigger times must lie in the range of 0 to 180000 ms.

2.6.1.6 Configuration of the Logging Buffer

CANalyzer initially saves the data arriving in the logging branch to a ring buffer in the PC's main memory.

Under **Buffer size** you can define the size of the buffer in which events are saved intermediately during a measurement. Approximately 50 bytes of memory is taken per message; however, this memory is only reserved as needed during the measurement. Consequently, for very large buffer sizes Windows may swap-out portions of the main memory to the hard drive during the measurement, and this may lead to significantly delayed program execution.

After the trigger condition occurs the ring buffer is filled with raw data until the post-trigger time has elapsed. The user can define whether the triggering process should be run once or multiple times. If the option "Measurement stop after n trigger blocks" is chosen, then the measurement process is stopped by the logging block after the

post-trigger time of the *n*th trigger has elapsed. At that time it transfers data from the ring buffer to the actual log file. Depending on the quantity of data, this memory transfer process may involve substantial waiting times.

Since the ring buffer is limited in size, data can be lost if the pre-trigger and/or post-trigger times are too long. Then only the last data are recorded up to the point where the post-trigger time elapsed, and an error message is output. If necessary the measurement can be repeated with a data reduction filter inserted upstream.

Note: If you wish to write all data to file for a long-duration measurement you should select the option **Write full buffer to file**. This option causes the buffer contents to be written to the log file during the measurement as soon as the prescribed (maximum) buffer size is reached. The buffer size is of decisive importance with this option. To prevent excessive system loading during saves to the buffer, and to prevent data losses, the buffer should not be configured to be very large. Buffers of approx. 1000 messages have proven effective in practice. This option is only accessible in **Entire Measurement** mode or in **Single Trigger** mode with the trigger type **Start**.

To prevent data losses, especially at high system loading, you should observe the following points:

- Close all applications that run in background and demand system time;
- Switch all blocks in the measurement setup over to **Cyclic Update** or disable them entirely.
- Utilize filters for data reduction.
- Do not execute any unnecessary user actions during the measurement, such as moving windows.

If a data loss occurs in spite of these measures, you have the program inform you of this by a message. To do this, activate the **Data lost message box** check box in the Logging dialog. After the end of measurement a dialog box is then displayed.

Lines in the log file marked with a '*' as a special symbol have corrupted lines around them.

The symbol for the data loss disappears when the overload situation has ended.

2.6.2 Log Files

The icon for the log file (file symbol) indicates that the data flow ends in a file.

Choosing the menu item **Log file** in the popup menu of the File icon to the right of the Logging block opens the configuration dialog for the log file.

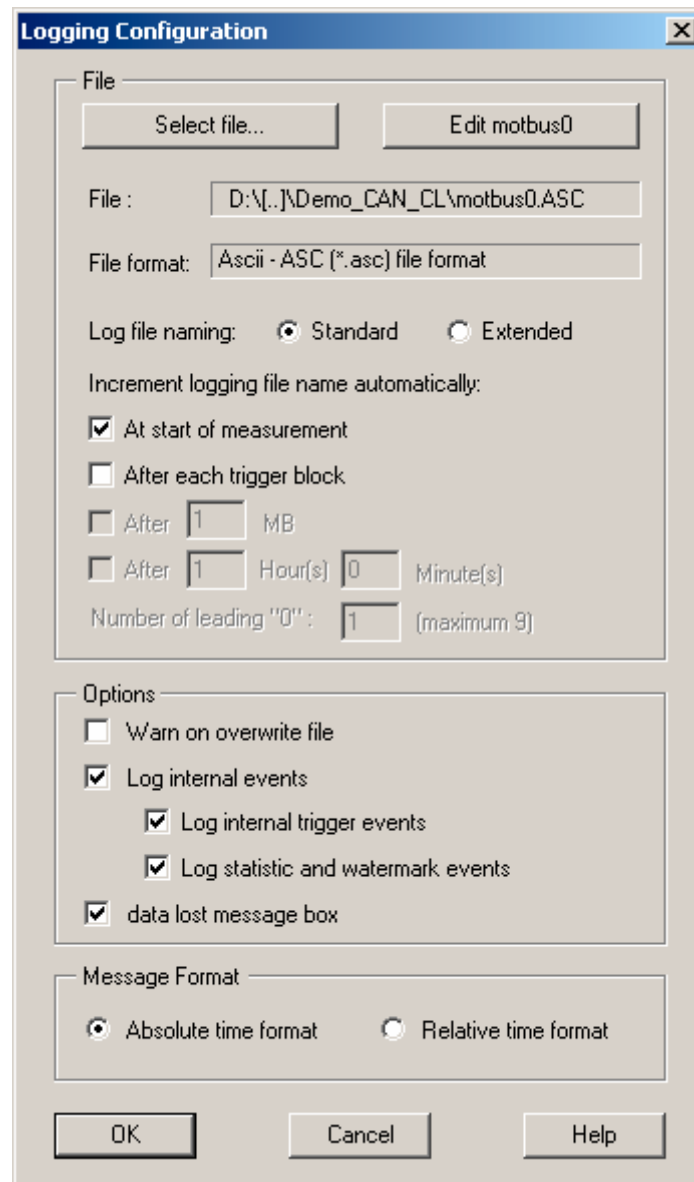


Figure 41: Configuration of the Log File

The format of the log file is defined by the File format combination box. The user can choose between **binary** and **ASCII** format. The binary format is recommended for online measurements, since this format is faster in offline evaluation and also generates smaller log files. When ASCII format is chosen, the data are saved as readable text. The setting of the global switch determines whether decimal or hexadecimal representation is used. Among other things, ASCII format can be used for data exchange with external programs or for incorporating trace data into documents.

The Offline mode data source can be configured to be either binary or ASCII files.

The automatically preset extension of the file name is `.LOG` for a binary log file or `.ASC` for an ASCII file. The recommended default name is `CANWIN.LOG` or `CANWIN.ASC`.

To view or edit an ASCII file, double click the file icon or press the **[Edit file]** button in the configuration dialog for the log file. You can use your own text editor for this. To do this, enter the following line in section `[Environment]` of the file `CAN.INI`:

```
LogEditor=MYEDITOR.EXE
```

whereby you must enter the name of your own editor for `MYEDITOR.EXE`.

With the check box **Increment logging file** you can indicate, that the name of the logging file is automatically incremented at start of measurement or after each trigger block, after reaching a defined file size or after reaching a defined duration. This prevents overwriting files that already exist.

In the configuration dialog you can indicate whether data losses in overload situations should be reported to you. In the log file, the faulty lines are marked with a '*' as a special symbol.

Note: Analogous to output in the Write window with the CAPL function `write()`, you can output text lines from CAPL programs to ASCII log files using the functions `writeToLog()` and `writeToLogEx()`.

2.6.3 Event Types in Log Files

In the following table you will find an overview of all events that are recorded in the log file. When the function *Log internal events* is active in the data icon's configuration dialog the internal events generated by the program (e.g. bus statistics information, triggers, etc.) are also logged. To have bus statistics information generated the relevant option must be activated in the **Card and Driver Options** dialog. There you also set how frequently these events should be generated.

In the first column you will find all event types that are recorded by logging. In the second column is the format of the particular event in ASCII logging. The third column provides information on whether the event can also be recorded in binary MDF log files. You can determine from the last column whether the function **Log internal event** must be activated to log the event.

Event type	Format in ASCII Logging	Binary	Comments
CAN message	<Time> <CAN> <Name or ID> <Attributes> <DLC> <Data>	Yes	
Error Frame	<Time> <CAN> ErrorFrame	Yes	
Overload-Frame	<Time> <CAN> Overloadframe	Yes	
CAN error	<Time> CAN <CAN> Error: <Error message>	Yes	Internal event
Bus statistics	<Time> <CAN> Statistic: <Data>	Yes	Internal event
Measurement start	<Time> Start of Measurement	No	
Logging Trigger	Statistic: <Time> log trigger event	Yes	Internal event

Event type	Format in ASCII Logging	Binary	Comments
Trigger block Start	Begin trigger block <Time and date>	No	
Trigger block End	End trigger block <Time and date>	No	
Overload symbol	'*' at beginning of line after the data loss occurred.	No	Internal event,

2.6.4 Data Analysis in Offline Mode

To study recorded log files switch CANalyzer to Offline mode with the main menu item **Mode | To Offline**. Figure 42 shows the corresponding data flow diagram.

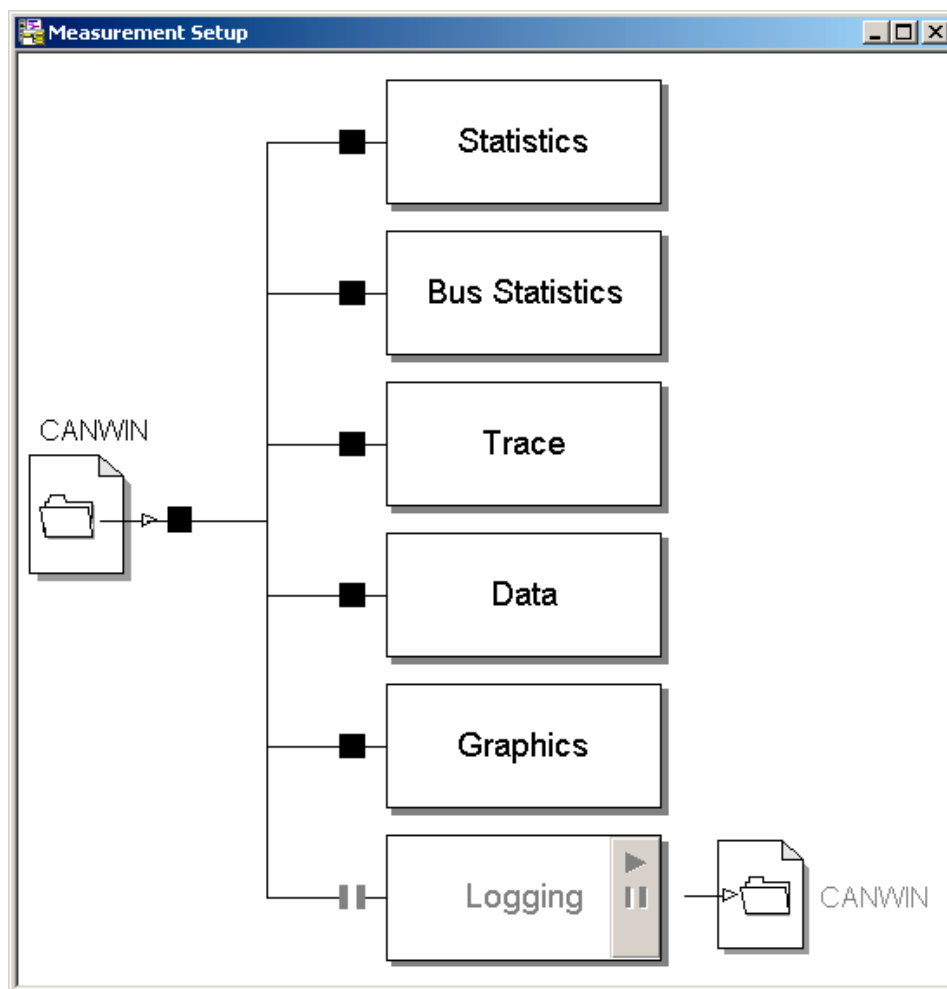


Figure 42: Data flow diagram for the Offline Mode

The data source in Offline mode is a file, e.g. generated by logging in Online mode. Analogous to Online mode, all measurement and evaluation windows are also available to you in Offline mode. The only option omitted is the possibility of sending data over the bus. Furthermore, Offline mode provides a powerful search and break func-

tion, with which you can intentionally stop the replay of the log file. This is described in section 2.6. In the logging block, which is also available in Offline mode, data can be resaved to a new file, whereby targeted data reduction can be achieved by means of insertable data manipulation blocks.

You can enter the name for this file under the **Configuration...** item in the data icon's popup menu on the left side of the offline measurement setup. CANalyzer supports both binary and ASCII logging formats for this.

The menu item below this, **Break conditions...**, opens a dialog in which you can set an interruption point – a **breakpoint**. When the condition you have specified occurs, the offline replay is interrupted until you resume replay of the file with the function **Run** (F9), **Animate** (F8) or **Step** (F7).

You can configure the break condition with the **[Conditions...]** button, which is described in more detail in section **Fehler! Verweisquelle konnte nicht gefunden werden..** If the tools provided in the configuration dialog are inadequate, the CAPL language – with the function `stop()` – allows you to program a breakpoint yourself.

2.6.4.1 Flow Control in Offline Mode

The following functions of the main *Start* menu are available to you in Offline mode to track the recorded bus proceedings on the screen in slow motion:

Start

The individual messages of the data source are read-out and are sent as quickly as possible through the components of the measurement setup. In Offline mode the measurement can be resumed after a break. *Reset* must be called for a new start.

Reset

After a measurement has been run through either partially or completely in Offline mode, it can be reset to the beginning again with *Reset* and can thereby be studied from the beginning again.

Animate

Instead of reading data from the source file as quickly as possible, in Animate mode only about 3 entries per second are read from the source file. This results in a slow-motion representation of the processes. All of the display windows may be active during this process, e.g. so that the user can easily observe the sequence of messages in the Trace window. The Animate run can be interrupted by the <Esc> key or the menu command **Start | Break**. The speed of the Animate mode can be set with the following line in section `[Offline]` of the file `CAN.INI`:

```
AnimationDelay=nnn
```

where *nnn* describes the time between the replay of two successive events in milliseconds. (The default value is 300 [ms])

Break

In Offline mode this menu item interrupts the replay of data from the source file (Animate run). The same can be achieved by the <Esc> key. A restart resumes the replay at the point where it was interrupted by Break.

Step

This menu item (or the <F7> key) causes a single step of the measurement to be run. Each time this is activated only one additional message is read from the log file and processed in the data flow plan.

Within a measurement the user can switch back and forth whenever desired between **Start**, **Animate** and **Step**.

2.6.4.2 Configuration of Online and Offline Modes

In principle, there are different measurement setups and completely separate parameter sets for Online and Offline modes, so that the two modes - including their window layouts - can be configured differently.

Nevertheless, it is often the case when switching between Online and Offline modes that the user would like to assume the configuration settings made in one mode in the other mode. Therefore CANalyzer provides the functions **Online(Copy)** and **Offline(Copy)** in the main **Mode** menu.

Online

Toggles to Online mode. In Online mode a data flow plan is shown in the measurement setup window, whereby the CAN PC-card serves as the data source. Moreover, there is also a transmit block. The time basis in Online mode is real time, and clocks in CANalyzer and on the CAN board are synchronized at the start. All time data are referenced to the start of measurement. Data acquisition cannot be resumed after a break in Online mode, rather it may only be restarted. The Animate mode is not possible.

Online (Copy)

Toggles from Offline mode to Online mode, whereby the data flow plan of Offline mode is assumed with all of its function blocks. The corresponding portion of the last Online configuration is lost in the process. Since the Offline mode does not have any transmit branch, that section remains unchanged.

Offline

Toggles to Offline mode. In Offline mode a data flow plan is shown in the measurement setup window, in which a data icon serves as the data source. Evaluation of this file is begun by **Start**, and the evaluation can be resumed after a break with <ESC>. Animate mode and Single-Step mode are also possible. The time basis in Offline mode is based on the times recorded for the data in the file.

Offline (Copy)

Toggles from Online mode to Offline mode, whereby the portion of the data flow plan of Online mode located after the branch to the transmit block is assumed. This affects all window configurations and function blocks. The last Offline configuration is lost in the process.

The mode switchover functions with copy represent a user-friendly method for assuming elaborate options such as trigger conditions, data display configurations or CAPL programs, and it allows the user to begin immediately with analysis or a new recording.

2.6.5 Exporting and Converting Log Files

The contents of log files can be exported or converted to other file formats with the help of signal-based Logging Export. The configuration dialog for Logging Export can be opened by the **Export** menu item in the popup menu of the File icon to the right of the Logging block.

2.6.5.1 Export

The user can limit the export to specific signals. This involves selecting the desired signals in the **Signals** selection list.

In the **Expanded options** dialog that is opened by pressing the **[Expanded]** button in the Logging Export Configuration dialog, the user can define in the **Actions** box those programs that can be started after an export.

2.6.5.2 Conversion

Conversion of log files is supported in both directions, i.e. ASCII->Binary and Binary->ASCII.

2.6.6 CANlog support

CANlog is a programmable data logger for CAN systems. CAN messages of different CAN devices can be received and stored. Also trigger conditions created in CANalyzer can be exported directly to CANlog.

Choose in the context menu **Export to CANlog**

- **Export to file**, to create a CANlog configuration file
- **Export to device**, to configure the CANlog hardware directly with the trigger conditions created in CANalyzer

More information about CANlog support you can find in the CANalyzer online help.

2.7 COM-Server

A COM-Server is implemented and helps the program to be gated or controlled by other applications. Besides accessing configuration-specific data it is also possible to control the measurement. You can also call CAPL functions, read signal values, and both read and write access environment variables.

Control can either be realized by COM-ready script environments (ActiveX Scripting: VBScript, JScript, Perl, ...) or by stand-alone applications, which can be written with RAD development environments (Visual Basic, Delphi, ...) or in C/C++.

Please see the Online-Help for a more detailed description of the COM-Server.

2.8 Troubleshooting

CANalyzer will not start

CFG file destroyed? Often it is helpful to delete the active configuration file `MYCONFIG.CFG`. First, the file should be backed up under a different name so that its contents are not lost. After the problem has been cleared up it can be renamed back to `MYCONFIG.CFG`.

CANalyzer runs too slowly

Power managers, which are particularly common on notebook computers, may not be installed for CANalyzer/ operation. Among other things, the power manager deprives the application of CPU for long periods of time. Therefore, transmit times are incorrect and messages can be lost when a power manager is installed. To remove the power manager from your system, please compare the instructions in the hardware installation guide.

For less powerful computers it may also be advisable to reduce the resolution of the system clock. The time stamps for messages may then be less accurate, but fewer demands are placed on computer CPU. To do this, enter the following line in section `[PCBOARD]` of the `CAN.INI` file:

```
Timerrate = 200
```

or under some circumstances even the value:

```
Timerrate = 400
```

These correspond to time resolutions of 2 or 4 milliseconds, respectively.

Card cannot be initialized

Timeout ...

With error messages of this type, CANalyzer cannot establish a connection to the CAN hardware. Check the installation of the CAN card. You will find hints for troubleshooting in the hardware installation guide.

Above all, notebook PCs frequently use a power manager. This **must be deactivated!**

Error message: "Error in transmission"

Immediate state change of CAN controller to ERROR PASSIVE

- Bus not connected? The bus connection should be checked, and possibly also the pinout of the connector being used.

- Terminating resistor present? In particular, the CAN AC2 version with 82527 controllers reacts sensitively to a missing bus termination.
- No partner on the bus? If the bus is only connected to one of the two CAN controllers, and there are no other bus nodes, the controller does not receive any acknowledge for transmission attempts.
- Baud rate and output control set? The controller register can be programmed by the popup menu of the CAN-card icon. See section 2.10 for a more detailed explanation.

Error message:

*An error has occurred in ...
For further information please report the
LINE and FILE to our Hotline*

At the current level of technology it is impossible to develop completely error-free programs that are non-trivial. Unfortunately, this is also true of CANalyzer. To better localize and correct system errors that occur very seldom, CANalyzer has diagnostic mechanisms that report these errors. Please be sure to write down the exact text of the error message. With this information our telephone Customer Support can then help you to correct the problem more quickly.

2.9 List of Error Messages to the CAN Interface

Error messages related to communication between CANalyzer and the CAN PC-card as well as errors on the CAN bus or in the CAN card firmware appear in this list. In each case, a clear text and an assigned error number are given. Some of these messages are hardware-specific and therefore do not occur with all card types.

Message was not transmitted (14)

The last transmit request was not executed by the CAN controller. This may be due to the error status of the controller, or due to the accumulation of too many higher priority messages.

DLC error (1128, 111, 112)

With the CANIB card a smaller DLC is specified in the DPRAM setup than should be transmitted. This size must be adjusted in the CAPL generator block or replay block.

This error can trigger other consequential errors. When problems are encountered the computer should be rebooted with a cold start!

Due to a firmware error in CANIB, a DLC equal to 0 cannot be processed. Therefore, it is set to at least 1 in the driver. However, an adjustment for calculating the size of the available DPRAM will not be implemented until Version 2.0.

DPRAM Overflow (5) DPRAM

In the message setup more messages are defined than the DPRAM can receive. Due to the fact that CANalyzer checks the number of messages, this report doesn't appear.

Incorrect controller no. (3,10,113)

An attempt was made to access a nonexistent CAN controller. Most CAN cards supported by CANalyzer have two controllers. But there are also cards with just one controller.

Remedy: Look for a **message CAN n** ... in the CAPL programs, where n is greater than the number of existing controllers. This must be replaced by a correct number.

Incorrect module number (109)

No CANIB was found at the address given. If the port address has been jumped over on the card, this must be communicated to CANalyzer See the manual.

Incorrect checksum (1368)

The CAN controller detected a faulty CRC checksum.

Incorrect terminating code (1432)

For CANIB cards this is the error code for general firmware problems. These occur primarily in BUSOFF and ERROR_PASSIVE states.

Incorrect card type (8)

CANalyzer card driver and hardware are not compatible.

Remedy: The correct CANalyzer version should be started, or another CAN card should be installed.

Error in statistics request ()

The CAN card or its firmware does not support bus statistics.

FIFO entry>16 (114)

CANIB has reported invalid data to CANalyzer. A potential cause for this is a RX buffer overrun. This cannot be detected directly if the CANIB was not initialized correctly. If its jumpers are transposed the user should follow the instructions in the CANalyzer manual.

Interrupt not found (108,2016)

Proper communication could not be established with the CANIB card. A check should be made to determine which interrupt is jumpered on the CANIB card, and whether there could be collisions with other hardware. It is often the case that there is an overlap conflict with the mouse interrupt. The CANIB driver itself can determine the interrupt allocated to CANIB. If this is unsuccessful the IRQ number can be entered explicitly starting with Version 2.0 (see BOARDCFG.INI).

No final code received (1384)

Can neither appear at CANalyzer nor at CANoe.

No access to IMP (12)

The firmware hasn't received access to the interface management processor of the Full CAN Chips 82526.

No reply from CAN controller (106)

The firmware could not establish a connection to the CAN controller. This is an indication of a defective CAN card.

*No messages in RX buffer (1)**No message received (7)*

No data are currently being received by the card.

Command from driver not supported (6,11,1528)

CANalyzer has sent a command to the card driver which it or the firmware does not recognize. Example: A bus statistics request to a card without the appropriate logic.

RX buffer overrun (101)

The receive buffer could not accept any more received messages.

There are several methods for remedying this situation:

- Insert breaks in branches of the data flow plan which are not needed. In an extreme case the statistics block, data block and trace block can be disconnected by breaks. The measurement is recorded with the logging block and afterwards is evaluated offline.
- For Basic-CAN controllers a reduction in the data stream can be achieved by acceptance filtering. Except in special applications the second controller in particular may be completely disconnected by this method.
- For Full-CAN controllers data reduction can be accomplished by striking messages in the message setup in conjunction with filter blocks.
- Switching-off the statistics report or other unnecessary options.

RX register overrun (105)

The Basic-CAN 82C200 controller has only two internal registers for accepting messages. At higher bus frequencies and higher message rates, newly arriving messages overwrite this buffer before the firmware can read out the register.

Correction: Use acceptance filtering.

Timeout while accessing card (4,232)

Communication problems with the firmware occur during a measurement.

Remedy: Terminate measurement and restart. If this does not help, the reset button can be pressed for some cards. Otherwise, the PC should be rebooted.

*Timeout during card initialization (0)**No reply from the card (1400)*

No connection could be established with the firmware when the CAN card was initialized.

TX buffer full, TX-REQUEST rejected (2)

The transmit buffer is still full. The new transmit request cannot be processed. There are three possible reasons for this:

- CANalyzer is transmitting data faster than the firmware can receive them and pass them to the CAN controller. This may occur, for example, if higher priority messages are being transmitted on the CAN bus.
- The number of messages transmitted one directly after another in a CAPL program is larger than the transmit buffer. This problem occurs primarily when transmission is executed in a loop in CAPL programs:

```
for (i=0;i<50;i=i+1) output(Msg);
```

Remedy: Fast transmission by setting `msTimers` and in reaction to the timer event.

- The CAN controller being addressed is in the BUSOFF state and therefore cannot accept transmit requests any longer. This can be detected in the bus statistics window.

Transmission Error (1352)

Can neither appear at CANalyzer nor at CANoe.

Unknown transmit identifier (100,1496)

A message which is to be transmitted is not entered in the message setup of the Full-CAN controller.

Invalid DPRAM address (107)

When the CAN card was initialized, an illegal identifier was entered for the DPRAM address. See card documentation.

2.10 Interface to the Hardware

In Online mode¹ the data source of CANalyzer is the CAN PC-card. It provides messages received from the CAN bus with the time of reception. Furthermore, if desired

¹ The Offline mode for the analysis of log files is described in section 2.6.4.

it can also acknowledge transmit requests with the exact time of the request and provide transmitted messages with the exact time of transmission.

Depending on the hardware used, other events might be provided by the board. These may include, e.g. detection of Error or Overload frames, measurements of bus load or the arrival of external trigger signals.

CANalyzer supports a maximum of 32 (virtual and real) channels. Consequently, you can also use multiple CAN cards to transmit and receive messages. The number of CAN channels to be used is set in the channel definition (cf. section 2.4.1).

The hardware is initialized at the start of a measurement. You must communicate the parameters required for this to CANalyzer before the measurement start. Parameters are configured from the menu bar with **Configure | Hardware** or from the popup menu of the card icon on the right side of the data flow plan, which you can obtain with the right mouse button or by <F10> on the keyboard (cf. section 1.2.1). Parameterization is hardware-dependent. It is described below for CANcardX, which contains two SJA 1000 CAN controllers. For other cards (CAN-AC2, CANcard, etc.) the description is supplied with the card. Operation of the various dialogs and the meanings of input boxes are explained in online Help.

Each channel can be parameterized independently. Consequently, applications can be supported which have independent bus systems with differing speeds.

When creating a new measurement setup, default values are configured automatically for the particular controller type.

2.10.1 Configuring the Hardware

You can obtain the dialog for hardware configuration from **Configure | CAN bus parameters...** on the menu bar.

On the left side of the dialog is an overview of the channels defined in the active configuration. The number of channels can be set in the Channel Definition Dialog (cf. section 2.4.1).

When you click on a channel symbol with the left mouse button, you will see the chip allocation for this channel on the right side of the dialog. The chip allocation is the allocation of the application channels to the CAN chips registered in the system. This can be configured in the CAN hardware configuration section of your computer's Control Panel.

After the sub-icons drop down in response to double clicking on the channel icon or clicking once on the **[+]** symbol, the chip icon appears beneath the channel symbol with the label **Setup** and two items with the labels **Acceptance filter** and **Options**.

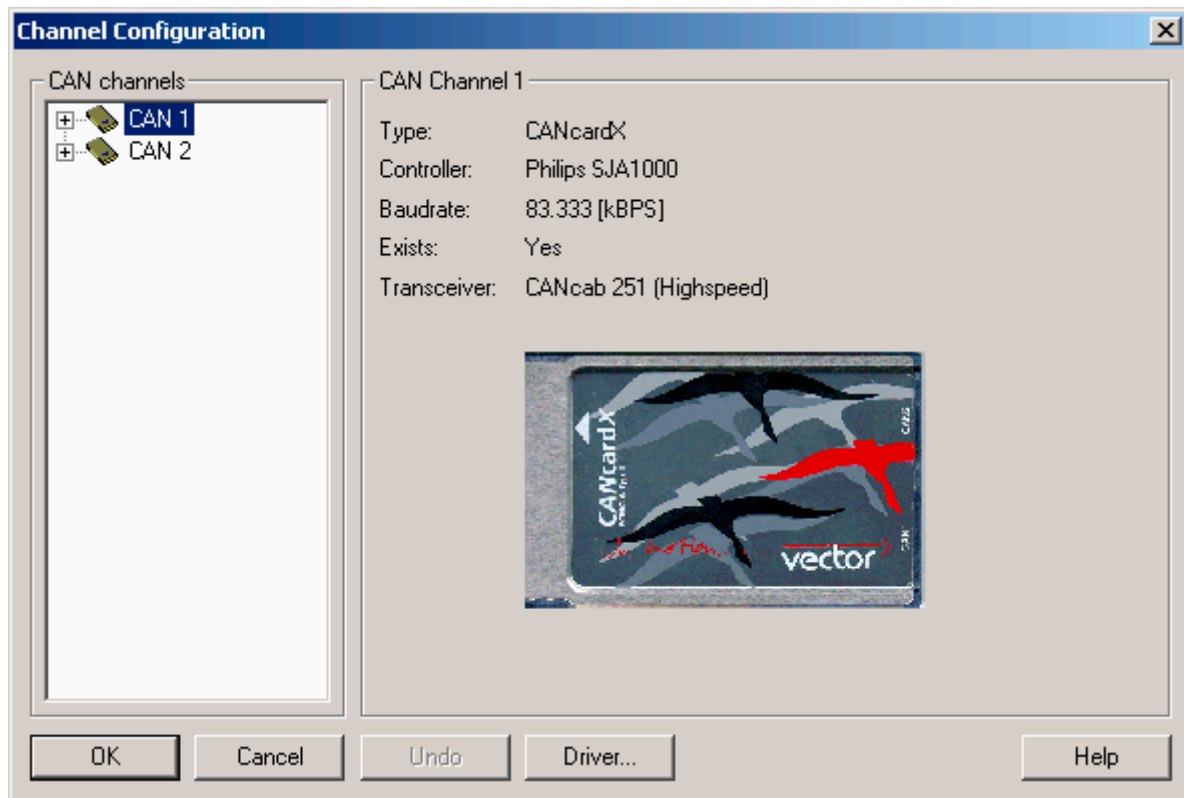


Figure 43: Configuration of Hardware

When you click the **Setup** chip icon the sub-dialog for bus parameterization appears on the right side, with which you can configure the chip's baudrate and bus registers (cf. section 2.10.2). When you click the **Acceptance Filter** item, on the right side you see the sub-dialog for configuring the chip's message acceptance filtering (cf. section 2.10.3). Finally, with the **Options** item you obtain a sub-dialog for configuring driver and card options (cf. section 2.10.4).

You can confirm all entries with the **[OK]** button or reject them with **[Cancel]**. If you only wish to undo the entries of one of the sub-dialogs mentioned above, press the **[Undo Page]** button in the particular sub-dialog. If a channel icon is selected **[Undo]** will cancel all changes (Setup / Mask / Options) made to this channel.

2.10.2 Programming the Bus Parameters

In the **Setup** sub-dialog you can directly configure the desired **Baudrate** for the CAN chip in the relevant input box.

As soon as you leave the input box (by <TAB> or mouse click in another option box) the associated register values are calculated and all relevant option boxes are updated automatically.

There may be additional input boxes - depending on the particular type of chip - for directly programming the CAN chip registers:

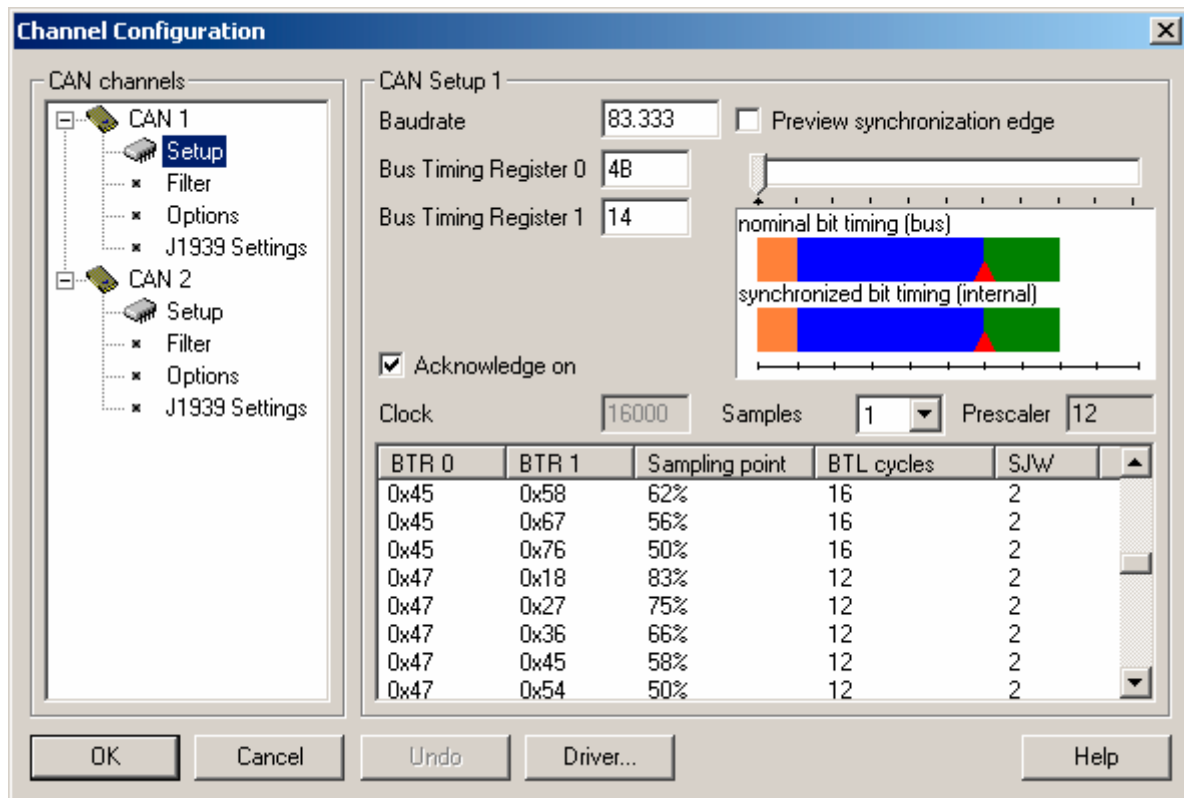


Figure 44: Bus Parameters Dialog

Above the list you see the input boxes for clock frequency and number of samples. Shown after the list is the prescaler resulting from the register values.

The CAN controllers are provided with 16 MHz clocks when delivered, and as a rule 16000 (kHz) should be entered in this box. However, it may be necessary to change out the clocks for special applications. Then the appropriate frequency must be entered here.

In the **Samples** input box you set the number of bus samples per bit time. Possible values are:

- **1 Sample.**
This setting is recommended for High Speed Buses (SAE Class C).
- **3 Samples.**
This setting is recommended for Low/Medium Speed Buses (Classes A and B) to filter out level peaks on the bus.

The two Bus Timing Registers define how an individual bit of the serial bit stream is assembled on the bus. Please refer to the data sheet for the CAN controller (SJA 1000/82C200/82527/72005) for the values that should be entered here. Input the values as hexadecimal numbers. On the right side of the dialog box you can determine whether the entered values are plausible (e.g. whether the desired baudrate has been achieved).

There are multiple Bit Timing Register pairs for a given baudrate which determine the timing of the CAN controller with regard to the sampling time point, number of BTL cycles and synchronization jump width (SJW). You can view a selection of allowable register pairs in the list of sampling options.

Displayed in this list are all Bus Timing Register values for the configured baudrate and sample count. Shown next to the register values are associated values such as the sampling time point (*Sample*) in percent of bit time after the beginning of the bit, number of BTL cycles (*BTL cycles*) and the synchronization jumpwidth (*SJW*).

Once you have changed the baudrate or sample count, click on the list box to update the list. Afterwards you can select the desired sampling option.

Representation of Bit Timing

The figure in the upper right area of the sub-dialog sketches the timing of the CAN controller resulting from the configured register values. It depicts the bit time schematically, subdivided into three regions *Sync* (orange), *Tseg1* (blue) and *Tseg2* (green).

The number of subdivisions corresponds to the value that is set for BTL cycles. The sampling time point (border between *Tseg1* and *Tseg2*) is identified by one or three small red triangle(s) according to the setting for sample count. The ratio between the bit length up to the sampling point and the overall length of the represented bit time corresponds to the percent value of the actively selected list entry.

Above the figure you see the **Preview Synchronization Edge** selection box which is deactivated by default, whereby the slider beneath it is also deactivated. Please note that this slider does not have any functionality as a control; it is only used for previewing purposes.

Click on the **Preview Synchronization Edge** selection box to activate previewing. With the help of the slider above the figure you can examine the effects of the synchronization edge and synchronization jump width on CAN controller timing. The slider is intended to represent the time point of a synchronization edge on the bus, i.e. the beginning of a bit with recessive-dominant edge. The upper area of the figure shows the nominal bit timing, i.e. a bit on the bus is depicted as it is expected by the controller. The lower area of the figure shows the internal controller timing, i.e. the bit time interval from the controller's perspective. The length of this bit interval depends on the time point of the arriving synchronization edge:

If the edge falls within the *Sync* region of nominal timing, then the chip is running synchronously. If the edge falls within the *Tseg1* region of nominal timing, then resynchronization must be performed. In this case the *Tseg1* region is lengthened by up to *SJW* (Synchronization Jump Width) BTL cycles. If the edge falls within the *Tseg2* region of nominal timing, then resynchronization must be performed. In this case the *Tseg2* region is shortened by up to *SJW* (Synchronization Jump Width) BTL cycles. If no edge falls within nominal timing, this bit time is not utilized for resynchronization.

Disconnecting the Transmit Branch

Since the CAN controller on the PC card represents the interface between the analysis software and the CAN bus, the bus is influenced by the measurement process. In particular, the CAN controller gives its acknowledge for correctly recognized messages by placing a dominant level on the bus at the relevant slot in the CAN message. To reduce the influence on the system, this functionality can be explicitly deactivated on the SJA 1000 controller. However, please note that when the acknowledge is deactivated communication can only occur over the bus if at least one other bus node sends an acknowledge.

Note: The acknowledge on the CAN-AC2 card can be deactivated as follows:
 with 82C200 controller, set the outputcontrolregister on 0x02 (default is FA)
 with 82527 controller, interrupt the TX line (jumper 9 must be removed on PB1 resp. PB2)

2.10.3 Acceptance Filtering

With all Basic-CAN controllers on the PC card (SJA 1000/82C200/82527/72005) a mask controls which messages can be transmitted and which can be received.

For example, the SJA 1000 has one acceptance filter for standard identifiers and one for extended identifiers, and it expects separate acceptance mask and acceptance code. The acceptance mask indicates which bit of the ID should be compared with the acceptance code. If the bit is 1 in the mask, then that particular bit is irrelevant for the comparison. If it is 0, that bit of the ID is compared with the corresponding bit of the acceptance code. If these two bits are identical, the message is received; otherwise it is filtered out. Both the mask and code are entered as hexadecimal numbers.

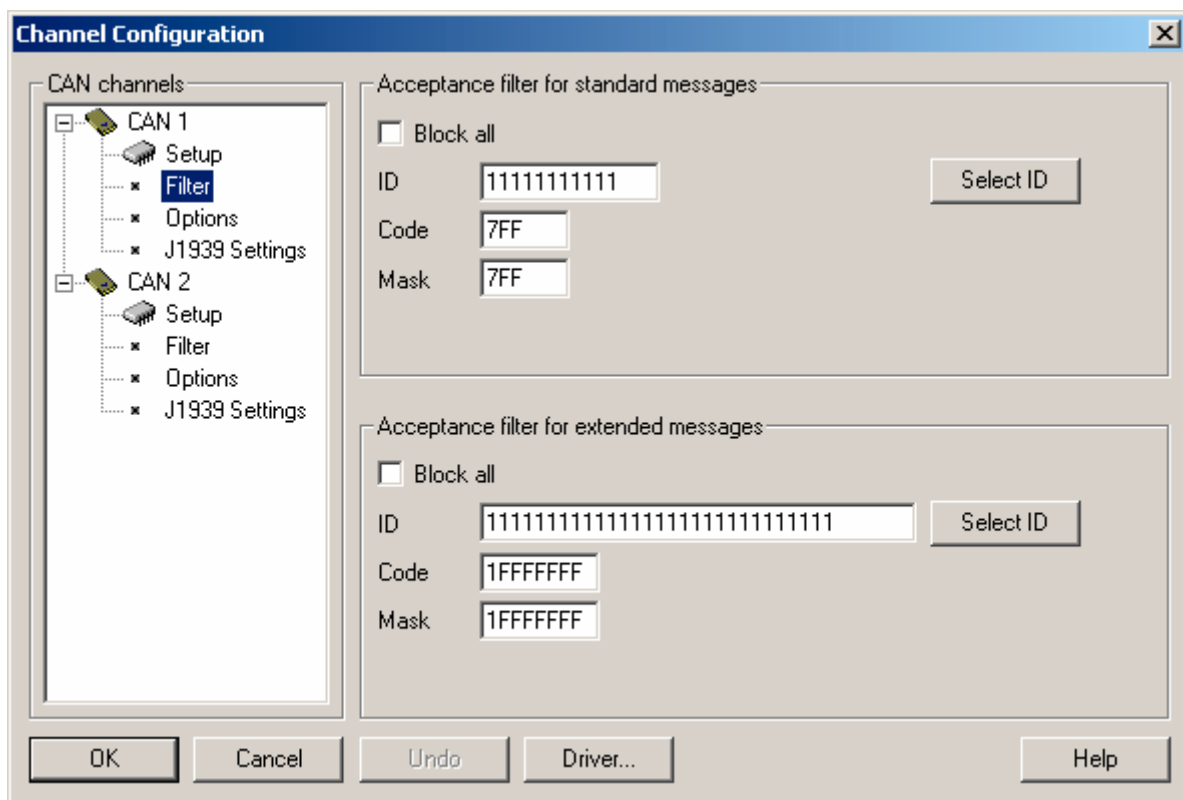


Figure 45: Acceptance Filtering with the SJA1000 Controller

Instead of entering the mask and code directly, you may program acceptance filtering in CANalyzer using a logical mask (*Acceptance filter for standard identifiers* or *Acceptance filter for extended identifiers*). You can enter one of the values 0, 1 or X for each bit in this mask. A message occurring on the bus is only be received if all mask bits given as 0 or 1 agree with the corresponding message bits. Bits shown as

X are not utilized in the comparison ("don't care"). The values of the acceptance mask and acceptance code are automatically calculated and displayed after inputting the logical mask.

You will find a detailed explanation of acceptance filtering for all supported CAN controllers, with examples, in online Help.

2.10.4 Card and Driver Options

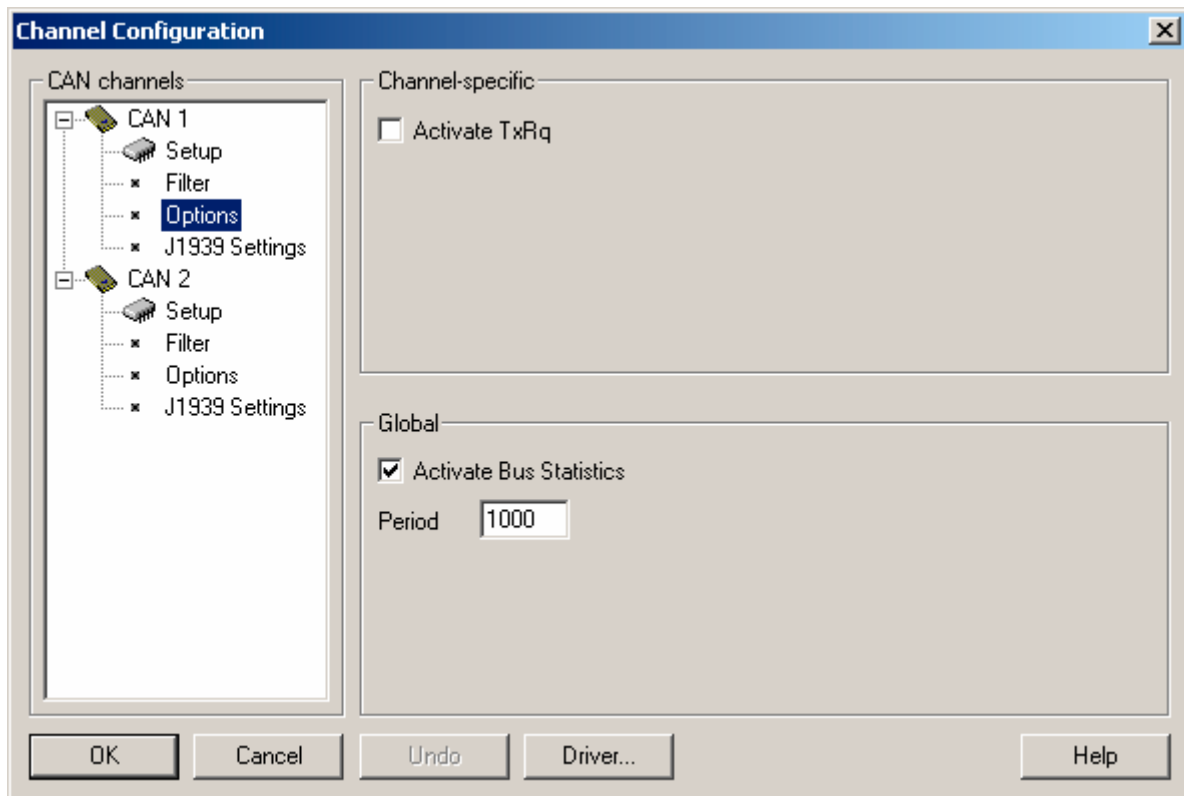


Figure 46: Card and Driver Options Dialog

Depending on the CAN PC card used, various options may be configured in this dialog. These include:

- Reporting the time point of a transmit request (TxRq) to measure delay times (cf. section 2.2).
- Choosing whether bus statistics should be displayed, as well as their
- Display refresh rate (cf. section 3.8.3).

The context-sensitive Help function offers more detailed descriptions for your particular hardware platform.

The actual display of statistical data can be deactivated separately by disconnecting the Bus Statistics window from the data flow. This will enhance performance. In contrast to a complete disabling of bus statistics support, data can still be logged and evaluated later in Offline mode.

3 Windows

It is usually the case that CANalyzer-users have a large number of windows opened, and until now these windows have been organized on a single desktop. A new desktop concept is being introduced with CANalyzer Version 5.0. Multiple virtual CANalyzer desktops provide the space to organize windows more effectively.

Under the previous window management concept opened windows were organized within the CANalyzer program window.

3.1 Desktop Concept

The purpose of desktops is to organize windows for better clarity and comprehension. You can distribute your opened windows to any desired number of desktops and sort information for work processes and other information by topic.

- Each desktop may provide any desired amount of information for viewing.
- It is possible to display identical information (identical windows) on different desktops simultaneously.

To achieve a better understanding of desktops, window management and their applications it is necessary to define certain concepts.

- There are windows that are represented in the form of blocks in the Measurement Setup. Each block defines the properties of its associated window with regard to configuration, position, size, and the desktop on which it is opened. Double clicking on a block opens its associated window.
- Each window (block) existing in the Measurement Setup may be opened on any existing desktop. If the user is utilizing n desktops, a window in the Measurement Setup may be opened n times. Each of these n windows of a block has an customized position on the n desktops.

Both of these concepts are described by the same term, **window**, and are referred to by that same term below.

The windows opened on different desktops may be placed inside or outside of the program window, or they may be docked in the program window.

The behavior of windows is determined by their types. A window may be switched from one type to another in order to assign a desired behavior to the window.

- A window of the **Floating** type is positioned outside of the program window.
- A window of the **Docked** type is permanently anchored in the program window.

3.2 Window Management

Also integrated in the window management conception are modal plug-in windows. Modal plug-in windows are only available for the measurement setup.

The **View** menu is expanded by several options if more than one window of the type (Trace, Graphic,...) exists.

There are different types of windows available that assign a specific behavior window. The following window types exist:

3.2.1 MDI window

- Windows of the **MDI** type are located within the program window and may be minimized.

3.2.2 Docking window

- Windows of the **Docking** window type may be anchored anywhere on the frame of the program window.
- They are always in the foreground.

3.2.3 Floating window

- **Floating** windows are created by the the **Docking** window type:

When you drag a **Docking** window from the program window to the Windows Desktop the window automatically becomes a **Floating** window.

If you drag a **Floating** window to the program window it is automatically converted to a **Docking** window.

You can avoid this behavior by keeping the <Shift> button pressed while dragging the window.

- Windows of the **Floating** window type may be moved by mouse anywhere on the Windows-Desktop independent of the program window, and they always appear in the foreground.
- They cannot be overlapped by the program window.
- These windows do not appear on the Task bar.
- When the program window is minimized the associated **Floating** windows of the application are also minimized.
- **Floating** windows cannot be addressed by the <Alt>+<Tab> key combination.

3.3 Measurement Setup Window

3.3.1 Data Flow in the Measurement Setup

The data flow diagram of the measurement setup contains data sources, basic function blocks, hotspots, inserted function blocks and data sinks. Connection lines and branches are drawn in between the individual elements to clarify the data flow.

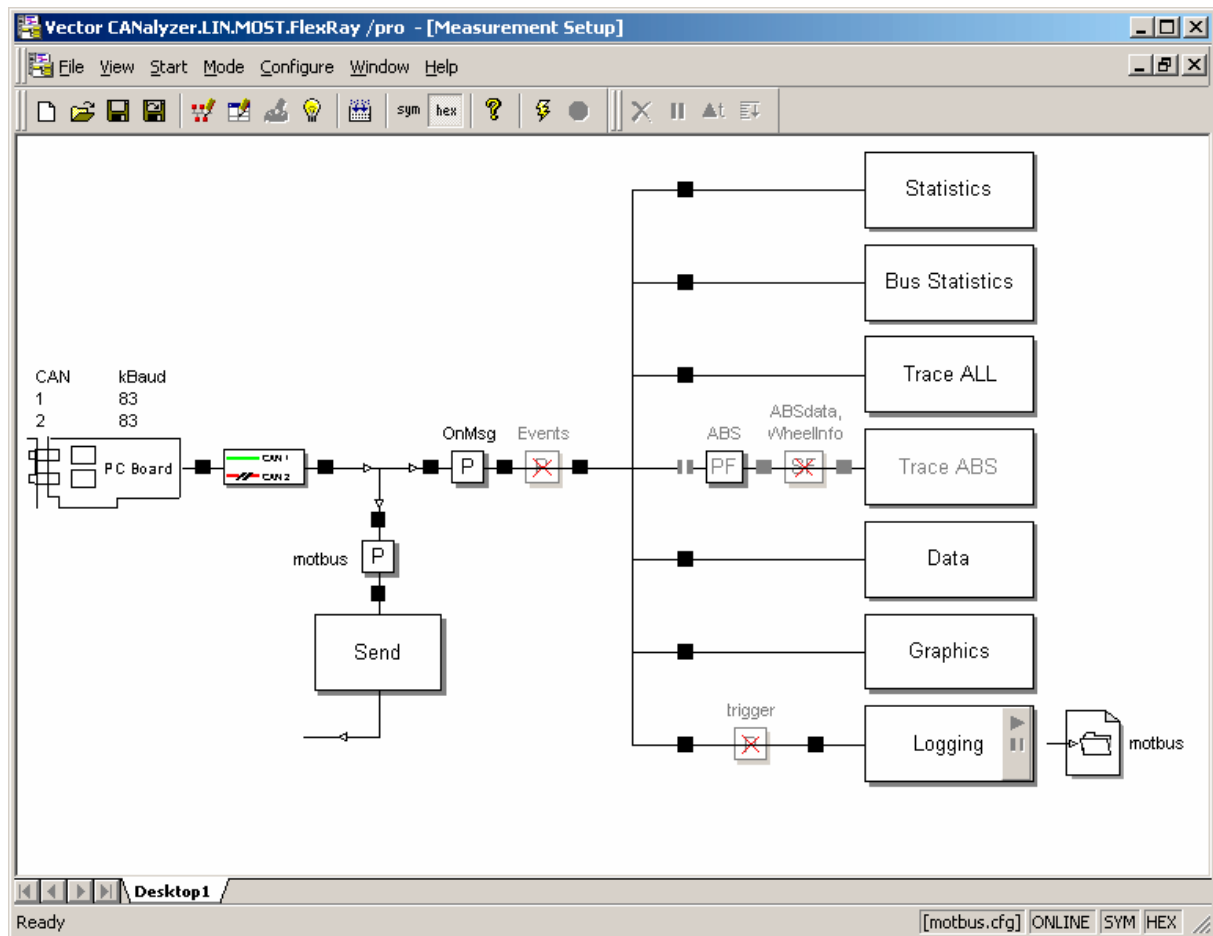


Figure 47: CANalyzer Measurement Setup

In Online mode the PC-card serves as the data source. It registers CAN messages on the bus and passes them on to CANalyzer.

Moreover, some of the supported PC-cards also provide additional information such as the detection of Error and Overload flags, the values of error counters, the bus load and external trigger signals.

3.3.2 Configuration of the Measurement Setup

Besides such functions as loading and saving configurations or associating CAN databases, which you call directly from items in the main menu, the data flow diagram and the function blocks in the measurement setup window are used primarily in the configuration of CANalyzers

You configure the measurement setup by activating a block in the data flow diagram (clicking it with the left mouse button or moving the selection frame with the <TAB> or arrow keys). Clicking on the selected hotspot with the right mouse button or pressing <F10> opens a popup menu with all configuration options.

For example, new function blocks such as filters or generator blocks can be inserted at the black rectangular insertion points (hotspots) in the data flow, and the PC-card's

CAN controller can be configured from the PC-card icon at the far left in the measurement setup.

To copy and move blocks there are the functions **Copy** and **Cut** in each block's popup menu, as well as **Paste** in the popup menu for a hotspot.

If you wish to exclude a function block from the measurement, you can deactivate it before the measurement with the spacebar or with the **Node active** line in the popup menu. This is especially helpful if you have already configured a block and only want to disable it for certain measurements without deleting it. Deactivated blocks are shown as a different shape to differentiate them from active blocks. A node can be reactivated by pressing the spacebar again or by selecting the same popup menu line again.

A brief look at the measurement setup window gives an overview of the configuration options provided by CANalyzer and shows you how your actual measurement configuration appears ("Graphic menu").

The measurement setup can be displayed in two different modes:

- Automatically fitted to the window size.
- Fixed magnification with scroll bars if necessary.

You can select these modes from the measurement setup's popup menu. In fixed magnification mode the dimensions of the measurement setup are preserved. If the window is too small to show the entire measurement setup, you can scroll to the hidden area with the help of the scroll bars.

3.3.3 Working with Evaluation Blocks in the Measurement Setup

All evaluation blocks on the right side of the measurement setup are displayed above one another. The standard evaluation blocks, Statistics and Bus Statistics, always appear exactly once each. Other evaluation blocks (Trace, Data, Graphics and Logging) appear at least once each.

To insert new evaluation blocks in the measurement setup, click the branch with the right mouse button and select the new window from the popup menu. This places the new block after the last block of the same type. It gets the standard name with a serial number. The first Trace window is called Trace, the second gets the name Trace 2, etc.

As an alternative, you can insert evaluation blocks by opening the popup menu for one of the evaluation blocks and selecting a new block there.

You can also delete the block from the measurement setup via its popup menu, provided that there is more than one evaluation block of that basic type in the measurement setup. When the block is deleted, the entire branch is always deleted, including all of the insertable evaluation blocks there.

To open the window assigned to the evaluation block, double click the block with the left mouse button or choose **Show Window** in the block's popup menu. Multiple windows of the same type are shown cascaded in the standard layout.

3.4 Trace Window

All messages arriving at the input of the Trace block are displayed as text lines in the Trace window.

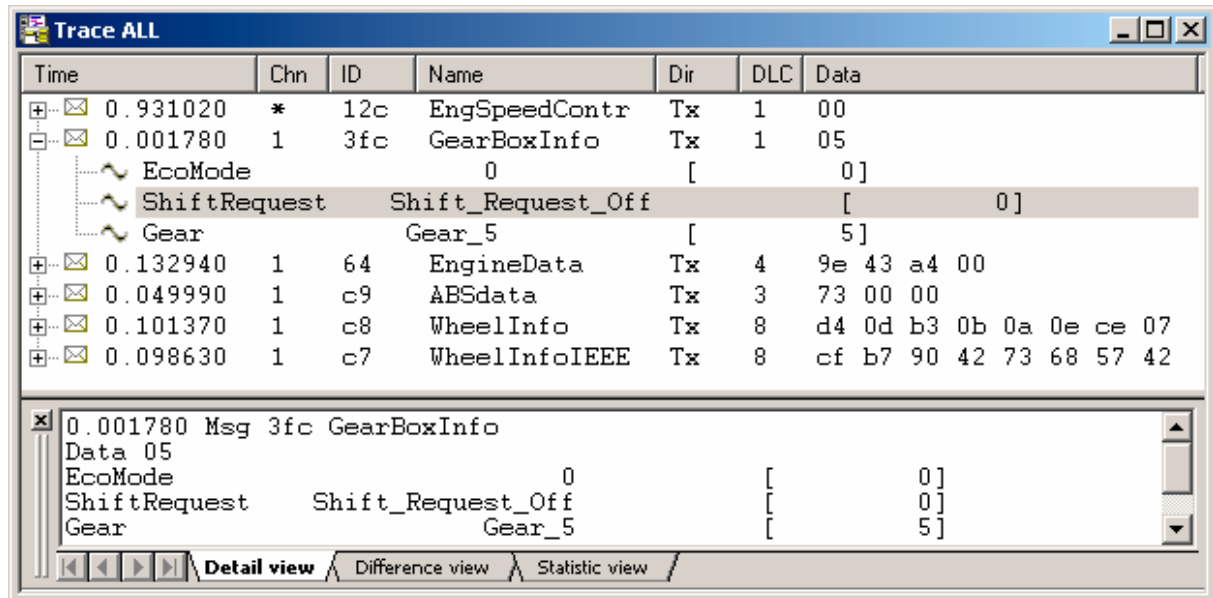


Figure 48: Trace Window

The figure shows an example of a Trace window of CAN. Depending on the specific CANalyzer option used, different columns are shown. The most important of these are described in the following table.

If you have installed one of the following options, you can find more information in the corresponding option's manual or in the online help:

- CANalyzer.LIN resp. DENalyzer.LIN
- CANalyzer.FlexRay resp. DENalyzer.FlexRay
- CANalyzer.J1939

Exemplary definitions of the CAN bus columns:

Time	Point in time when the information arrived at the CAN card (Receive, transmit or transmit request). If the message was generated by a CAPL program, the time set in the program is displayed. Output is in seconds from the start of measurement.
Chn	Number of the CAN controller chip which provided the message. Generally the number 1 or 2 is output in this column. If the message was generated by a CAPL program block, whereby the CAN number was not declared explicitly, the character * is output.
ID	Identifier for the message. Representation is decimal or hexadecimal according to the preselection. An X is appended to an extended identifier.
Name	If a symbolic database is used the message names appear. The columns

	"ID" and "Name" can also be shown as a combined column.
Dir	Possible values here are: RX = Receive, TX = Transmit or TXRQ = Transmit request
Attr	Special messages are described by add-on attributes: WU = Wake up or TE = Transceiver error
DLC	The DLC (Data length code) specifies the length of the CAN-data field in bytes.
Data	Listing of the message data bytes. Representation is decimal or hexadecimal according to the preselection. If a Remote message is involved, the text "Remote Frame" appears at this point.

A number of other events are output in the Trace window:

- Error frames:
When error frames occur a message will appear in the Trace window.
- Environment variables:
If the value of an environment variable changes, the time, name of the environment variable and new value are displayed. You can activate or deactivate the display of environment variables in the configuration dialog for the Trace window.

The Trace Window offers several views which can either be placed freely – together as one separate window – or be shown within the Trace Window (docked). There are tabs to switch between the following different views:

- Detail view
(replaces the Trace-Watch-Window)
- Difference view
(for direct comparison of different events)
- Statistic view (on signal values)
(several events can be selected and examined statistically)

The offline filter provides supplementary filtering of the recorded messages in the Trace Window. You can start this function with the **Event filtered** command in the context menu of the Trace Window.

3.4.1 Standard Configuration of the Trace Window

To configure the Trace window, select the Trace block in the data flow diagram by mouse or keyboard. Then choose the menu item **Window configuration** that appears, which opens a dialog box for configuring the window. Different display modes are available.

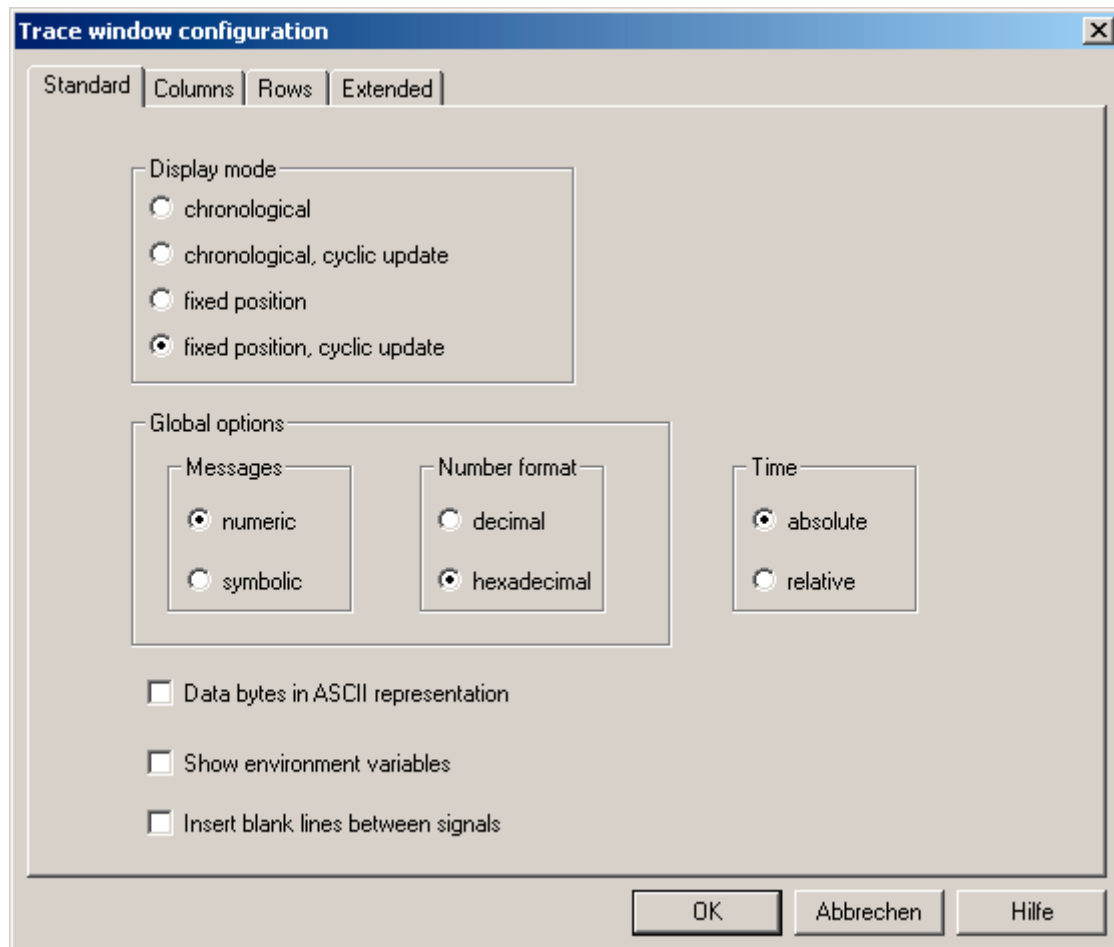


Figure 49: Configuring the Trace window

chronological *output mode*

The arriving messages are shown sorted by time. When the list reaches the window border it is scrolled, which can make it difficult to observe during fast bus traffic.

chronological, cyclic update *output mode*

Here, the arriving messages are shown sorted by time, as in the **chronological** mode. However, the Trace window is not updated with each message any longer, but rather only cyclically with a time constant. This saves on computing time during high bus loading, making bus observations possible on computers with weaker performance in this mode.


fixed position *output mode*

Here messages are assigned to fixed lines. As they arrive, new messages overwrite older messages of the same type on these lines. This mode has the advantage that even fast bus traffic can be tracked in a user-friendly manner. Furthermore, less computing time is required.

fixed position, cyclic update *output mode*

Here fixed lines are assigned to the messages, as in the **Fixed Position** mode. However, the Trace window is not updated with each message any longer, but rather only cyclically with a time constant. This saves on computing time during high bus loading, making bus observations possible on computers with weaker performance in this mode.

At any time you can switch back and forth between the two representations, even during a measurement. Thus, the user can switch over to chronological output mode after the measurement to page forward and backward through the received messages in the Trace window.

Note: The Trace window can be stopped by the toolbar icon  during a measurement run, so that you can analyze its contents in a user-friendly manner without terminating the measurement.

Furthermore, the Trace window offers you two different time formats. You can select whether the time stamp for messages should be displayed as absolute, i.e. in seconds since the start of measurement, or relative to the message preceding it. In the latter case, in fixed mode the time differences always refer to the same line. As a result, in this mode you can read off the transmission intervals between messages with the same identifiers. In **Chronological** mode the time differences are always shown referenced to the last message displayed in the window.

3.4.2 Configuration of the Columns in the Trace Window

Depending on the CANalyzer option or the purpose of the Trace window it may be advisable to show more or fewer columns. Therefore the Trace window offers a set of possible columns that can be layed out in any desired way.

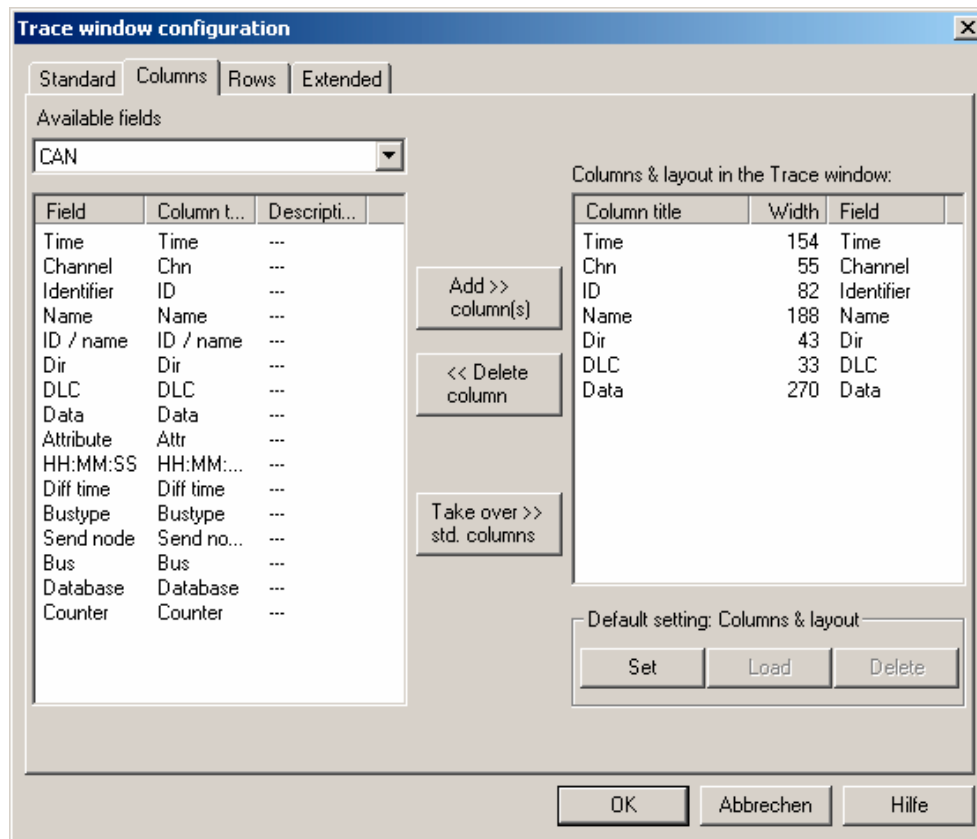






Figure 50: Columns Configuration in the Trace Window

A user-defined columns configuration is also possible as the standard. This means that all new Trace windows you create with this CANalyzer version are created with these same settings. Please refer to online Help for the exact procedure for configuring columns and selecting the individual configuration sets.

Note: Not all messages or events (e.g. Error frames) occurring in your system will fill the same columns or all columns. For example, the "Destination" column is not filled out for a CAN message. However, a J1939 message will show its "Destination address" in this column.

3.4.3 Trace Window Options from the Toolbar

The toolbar contains five buttons with which you can directly configure the Trace window²:

-  Delete Trace window
-  Update/Stop Trace window
-  Toggle Trace window mode: **Chronological/Fixed position**
-  Toggle Trace window mode for time representation: **Absolute/relative**

² The buttons for numbering format (hex/dec) and message format (numeric/symbolic) change the associated options globally for the entire program.

If you are working with multiple windows, the actions always act only on the active Trace window. If another measurement window is active, when one of these buttons is activated the window of the first Trace block is activated, and the function is executed in this window. If you are only working with one Trace block, the action is executed there directly without the window being activated.

3.4.4 Trace Watch Functionality and Trace Watch Window

The Detail View replaces the former Trace Watch window and offers user-friendly methods to further examine the contents of the Trace window after the end of a measurement or in Pause mode.

3.4.5 Optimizing the Trace Window

Since the Trace display requires a lot of computing time, at high bus loading the data might be displayed in a delayed manner. However, the messages will still have a correct time stamp that is already assigned to it by the PC-card. At high base load the message buffer might overflow and messages could be lost. A warning is output in the Write window. To prevent a buffer overflow when "Tracing along", it is recommended that all unnecessary branches (Logging, Statistics window, etc.) be disconnected. At high bus load, switch the Trace window to the fixed mode to economize on computing time.

In offline mode computer power plays a subordinate role. Here the trace branch can always be active. An **animate function** is also available, with which the entire measurement can be repeated in slow motion, and the entries can be tracked in the Trace window.

3.5 Graphic Window

The Graphics windows serve to display signal responses over time. As was the case for the Data block, if a symbolic database is used you can have the values of signals specified there displayed directly as physical variables. For example, the engine speed response could be viewed in units of RPM or the temperature pattern over time could be viewed in degrees Celsius.

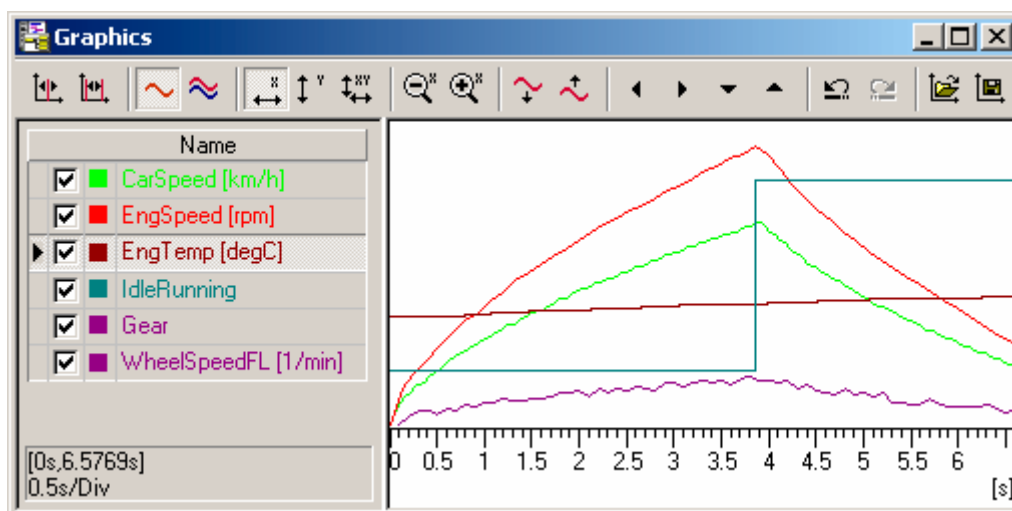


Figure 51: Graphics window

Signal-time responses are displayed graphically in the Graphics window. They are displayed in a X-Y diagram above the time axis. The measurement data remain in the Graphics window after a measurement stop and can be examined using special measurement cursors.

The Graphics window has a legend in which the selected signals are shown with their value ranges and colors. It also has a toolbar from which you can easily call the most important measurement functions. Both the legend and toolbar can be configured in the window's popup menu and can be enabled or disabled from there.

In the Graphics window there is exactly one active signal identified by inverted font in the legend. You can make a signal the active signal using the Tab key, by Page Up/Down or by clicking the signal name with the mouse.

- If **Single-signal mode** is enabled, all commands - such as Measure, Zoom and Scroll - refer to the active signal.
- In **Multisignal mode** the commands refer to all signals of the Graphics window.

3.5.1 Selecting Signals

In the Graphics window's signal selection dialog you can - just like in the dialog for the Data window - add signals, delete signals and enter display parameters for them. Open the dialog from the popup menu of the Data/Graphics block in the measurement setup, from the popup menu of the Graphics window, or by double clicking the legend at the right of the window with the mouse.

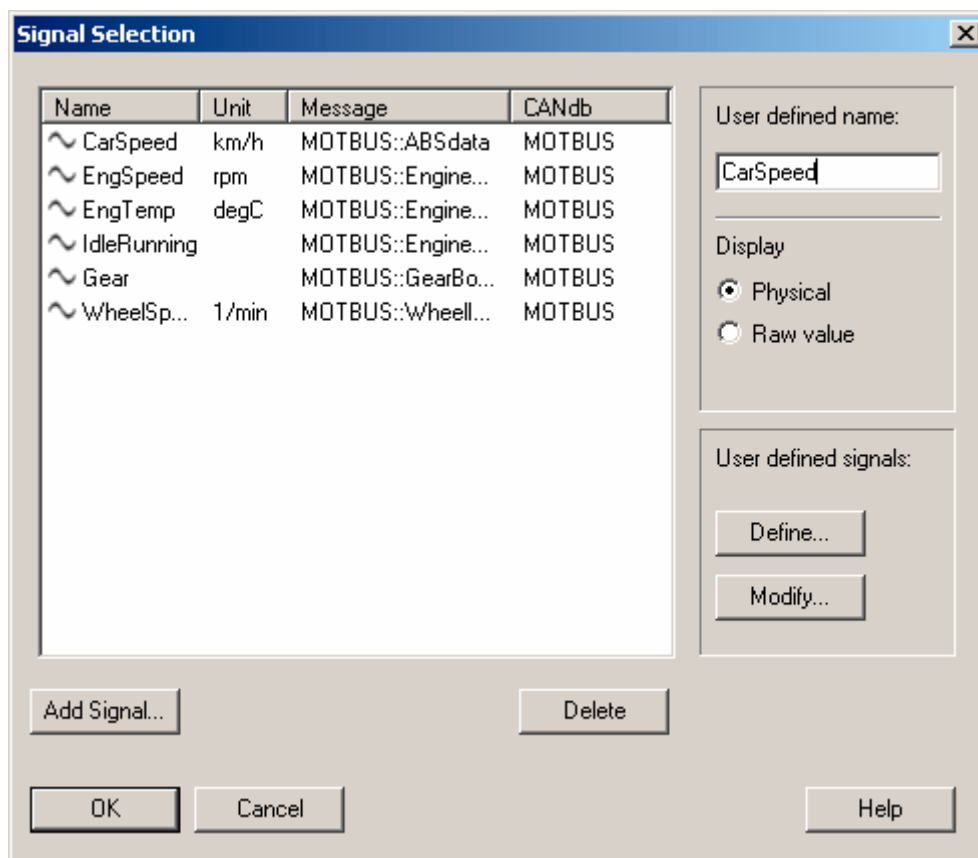


Figure 52: Signal selection dialog in the Graphics window

All Graphics window signals are displayed in the list box at the upper left of the dialog. With the **[New Signal ...]** button you first open a list for selecting CAN messages. In a second list you can then import the signals from the message you initially selected into the Graphics window.

Pressing the **[Delete]** button removes the highlighted signals from the list. With **[Define...]** you can define a signal, if you wish to display it without using the database. The **[Edit...]** button allows you to modify an existing signal description.

In the text input box **Short name** you enter a short, descriptive name for the signal, which is then output as the signal name in the Graphics window legend.

The display modes physical and decimal are available to you. In the physical display mode the raw data are extracted from the CAN message, scaled with the (linear) conversion formula, and displayed as physical values in the Graphics window. The necessary signal data are obtained from the database. In the decimal display mode, on the other hand, the raw data are only extracted from the CAN message and displayed as decimal numbers. There is no scaling by a conversion formula.

3.5.2 Arrangement of Signals

To exclude certain configured signals from the next measurement, you can deactivate them before the measurement. To do this, select the desired signal from the legend. Then in its popup menu choose the function **Deactivate**. You can also reactivate any deactivated signal from the same place.

Moreover, you can also change the sequence of signal entries in the legend of the Graphics window. This is useful if you have configured a rather large number of signals, but the window is too small to show them all in the legend. Additionally, you can move the deactivated signals to the lower border so that you can still see as many of the active signals in the legend as possible, even with relatively small window dimensions.

To move a signal entry you would select it with the mouse and choose the desired direction from the popup menu or by the key combination <Alt+Arrow up> or <Alt+Arrow down>.

3.5.3 Signal Layout

The signal layout for signals shown in the Graphics window can be controlled by a number of functions. You can set basic options in the **Graphics Window Options** dialog, which is opened from the window's popup menu or by double clicking the window with the right mouse button. All functions used to arrange the signals in the window or to study them after the end of the measurement are opened from the Graphics window's popup menu or the toolbar and are described in the next section.

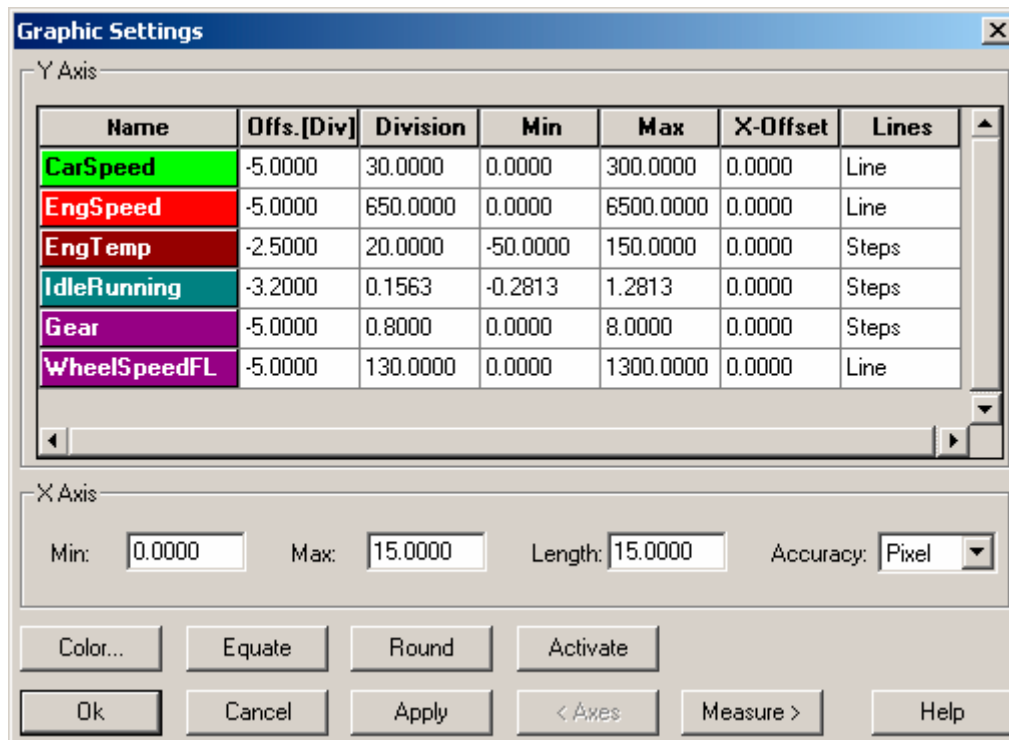


Figure 53: Configuring the axes in the Graphics window

For all signals you have entered in the Graphics window's signal configuration dialog, values are automatically assigned for Y-scaling, signal color and time axis, as well as the lines type. The values for Y-scaling and lines type are assumed from the database. In the Graphics window's **Options** dialog you can configure all of the options to satisfy your work requirements. The dialog consists of two parts, the axes options and the measurement options. You can toggle between the two parts with the action buttons **Measurement >** and **< Axes**.

Listed in the dialog's signal list are all those signals which were entered in the signal selection dialog.

3.5.3.1 Line Types

The line type in the last column identifies the display type for the display of a signal curve.

With **Line** the measurement points are connected by a line. This representation results in a continuous curve. It is the default option for displaying physical signals that are at least one byte in length.

With the **Steps** option, after a measurement point a horizontal line is output to the time of the next measurement point and from there a vertical line to the measurement point. This display type is especially suitable for digital signals. **Steps** is the default option for signals that are less than 8 bits in length.

With the **Horizontal** option, after a measurement point a horizontal line is output up to the time of the next measurement point. With **Dots** only the measurement points are marked.

3.5.3.2 Display Modes

Beneath the signal list you will find four input boxes for configuring the time axis. With **Output** you can define the display mode.

If multiple measurement points fall together within the time range of a single screen pixel, then in **Pixel** mode only measurement points at the borders of this range are displayed. This leads to faster output if there are many measurement points in a small space. Under some circumstances, however, individual peaks might not be shown in the signal response, if the measurement point with the extreme value lies within this range. In **Full** mode, all measurement points are output even if they lie within the time range of the same screen pixel at the active scaling. When there are many measurement points in a small space, this will lead to a lower output speed, but all extreme values of the signal response will be displayed.

The output mode setting has no influence whatsoever on measurement value acquisition.

3.5.4 Configuration of the Measurement

You can access the **Graphics Window Options** dialog by activating the **[Measurement]** button from the window layout dialog. Here you can configure the behavior of the Graphics window during the measurement.

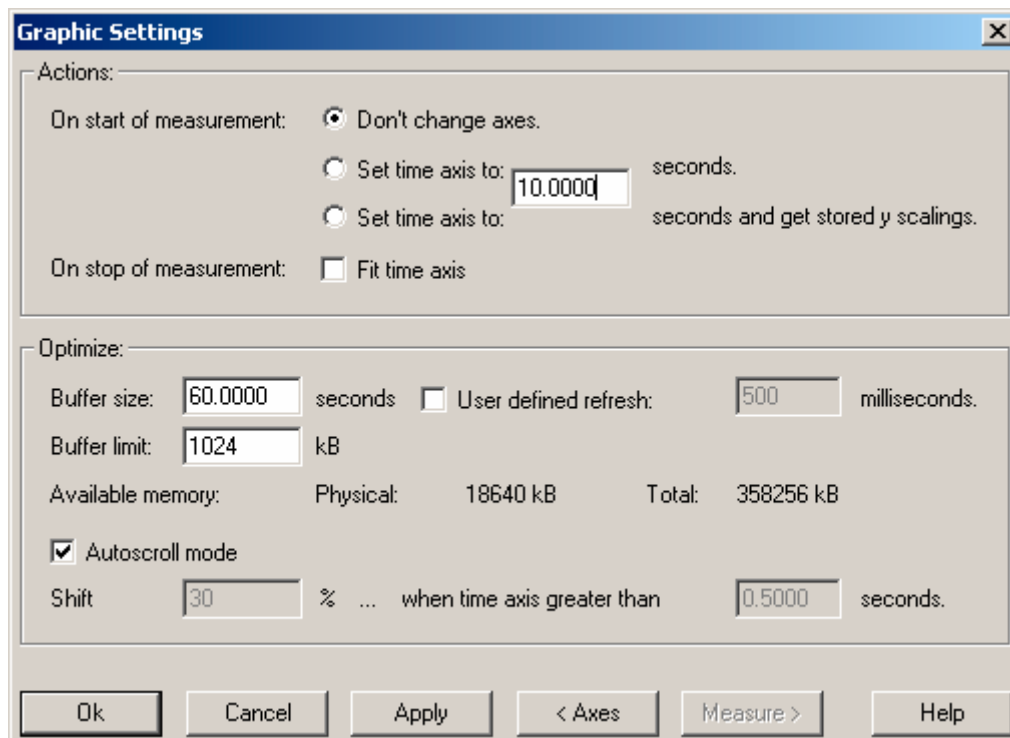


Figure 54: Measurement Options in the Graphics Window

If you select the option **Do not change axis options** under **Actions at measurement start**, then the last configured Graphics window setup is assumed at the start of the measurement. **Set with time axis** sets the Graphics window - at the measurement start - to a time range selected by you. Also, with **Set time axis and home**

configuration at the measurement start you can set the Y-axis range you had previously defined in the popup menu with **Save home configuration**.

If you check the option **Fit time axis** under **Actions at end of measurement**, the signal responses over the entire measurement duration are displayed at the end of the measurement. Otherwise, the displayed time interval remains on the screen at the end of the measurement.

3.5.5 Measurement and Display Functions

You can call measurement and display functions of the Graphics window, either from the popup menu or by activating the appropriate toolbar button. From the popup menu you can decide whether you wish to have the toolbar shown in the Graphics window or in the main window.

3.5.6 Signal Modes

You can activate single-signal mode from the popup menu with the **Single** function. In single-signal mode, functions such as **Zoom-in**, **Zoom-out**, **Home configuration** and **Fit** only act on the active signal of the Graphics window, which you select by clicking in the legend with the mouse. In multisignal mode (**All** function) the same functions act on all signals of the active Graphics window. Inverted output of the name of the active signal in the legend is disabled.

Note: It is always the case that either single-signal mode or multisignal mode is active. The operations that change time axis scaling are always executed for all signals - regardless of the setting single-signal / multisignal mode - since there is only one time axis for all signals in the Graphics window.

3.5.7 Measurement Modes

Point Measurement Mode

You can activate **point measurement mode** in the Graphics window with the function **Measurement marker** from the popup menu or by the appropriate toolbar button. If this mode is already active, it is deactivated by selecting the function again.

A measurement cursor (vertical line in the window) is displayed, which you can position by clicking and holding down the left mouse button. If the mouse pointer is located above the measurement cursor, it changes its form to a horizontal double arrow. If the mouse button is pressed at a point not located above the measurement cursor, a rectangle is dragged open when the mouse is dragged. The contents of this rectangle is then displayed magnified when the mouse button is released (Zoom function). You can also move the measurement cursor by keyboard with the key combinations <Shift-Arrow left> and <Shift-Arrow right>.

While the mouse button is held down a small square is visible which highlights the next closest measurement value. The measurement time, signal name and value are shown in the upper legend for this measurement point. In the legend with signal names, the signal values of all signals are displayed for this particular time point. The measurement cursor considers Single-Signal or Multisignal mode. In Single-Signal

mode the small box only jumps to measurement points of the active signal; in Multisignal mode the box jumps to the next closest measurement point of all signals.

Difference Measurement Mode

To evaluate the in measurement value differences between two points in time, you use the **difference measurement mode**, which you activate by the **Difference markers** item from the popup menu or by the appropriate toolbar button.

The measurement cursor and a difference cursor (vertical lines in the window) are displayed. If Difference mode is enabled, the cursors are shown at their active positions if they lie within the visible screen area. Otherwise they are moved to the viewing area. By clicking and holding the left mouse button you can position the cursors. If the mouse pointer is located above a cursor, it changes its form to a horizontal double arrow. If the mouse button is pressed at a point not located above the cursors, a rectangle is dragged open when the mouse is dragged. The contents of the rectangle are then displayed magnified when the mouse button is released (Zoom function). The cursors can only be positioned within the viewing area. However, the viewing area can be shifted by the arrow keys.

You can also move the measurement cursors by keyboard with the key combinations <Ctrl-Arrow left> and <Ctrl-Arrow right>.

While the key is pressed a small square is visible, which highlights the next closest measurement value. The measurement time, signal name and absolute value (not the difference) of this measurement point are shown in the upper legend. In the legend with signal names, the differences in signal values for all signals are shown for the time points that have been set. The two time points and the time difference are also displayed. The measurement cursor considers the option Single-Signal or Multisignal mode. In Single-Signal mode the small box only jumps to measurement points of the active signal; in Multisignal mode the box jumps to the next closest measurement point of all signals.

3.5.8 Display Modes

Three different display modes are available to you for layout functions such as **Zoom-in**, **Zoom-out**, **Home configuration** or **Fit**:

In X-mode the functions **Zoom-in**, **Zoom-out**, **Home configuration** and **Fit** only act on the time axis (X-axis) of the signals. In Y-mode the same functions act only on the values axis (Y-axis), while in XY-mode they act simultaneously on both axes.

3.5.9 Layout Functions

The Graphics window provides you with a number of functions for changing the window layout. Some of the functions available to you via the popup menu include:

Fit all

Independent of the preset mode, the signals are scaled such that they are completely visible. To do this, the program determines the actual minimum and maximum values for each signal and the time range of all signals, and scaling is adjusted accordingly.

Zoom-in/Zoom-out

This command magnifies or reduces, by a factor of 2, either the active signal (in Single-Signal mode) or all signals (in Multisignal mode). The size is changed according to the preselected axis mode, either for only one axis (in X-mode or Y-mode) or for both axes simultaneously (in X/Y-mode).

Operations that change the scaling of the time axis are always executed for all signals (independent of the option Single-Signal/Multisignal mode), since there is only one time axis for all signals in the Graphics window. Axes can also be scaled individually for each signal in the **Graphics window's** options dialog.

Fit

The signals are scaled such that they are completely visible. This involves determining the actual minimum and maximum values of each signal as well as the time range for all signals, and the scaling is set accordingly. The active modes (X-mode, Y-mode or X/Y-mode, and Single-Signal or Multisignal mode) are taken into consideration. This fits the entire graphic optimally in the window.

Round

Scaling of the displayed value range for all signals is rounded-off. This involves rounding-off the active *Division* value to a valid whole number ($\text{Division} = n * 10^x$, $n = [1 \text{ to } 9]$, $x = \text{whole number}$). The lower and upper range limits are rounded-off to the precision of the *Division* value.

Rounding always affects all signals of the Graphics window. The active mode (X-mode, Y-mode or XY-mode) is considered. Scaling of the signals is rounded-off to a whole number value.

Get scalings/Store scalings

You can save the scaling configuration of the Graphics window (i.e. the time range displayed for all signals and the value ranges of each individual signal) as a home configuration. The default home configuration has a time range from 0 to 5 seconds and the Min/Max value range of each signal taken from the database.

With the function **Store scalings** the current scaling of the active Graphics window is saved. In doing so, the active modes are considered (X-mode, Y-mode or XY-mode, and Single-Signal or Multisignal mode). Afterwards, you can reset all scalings in the Graphics window to this stored configuration at any time with the function **Get Scalings**.

Marker

This command activates or deactivates measurement point marking. The colour of the measurement points matches the preset signal colour.

Grid

This command enables/disables grid lines in the Graphics window. The grid lines match the subdivisions of the Y-axis. You can set the colour of the grid lines in the **Color Options** dialog.

Y-Axis

This command enables/disables labeling of the Y-axis in the active Graphics window.

When labeling is disabled, a Y-axis with 10 subdivisions is displayed for all signals. In the legend the following are shown for each signal: Lower and upper values of the viewing area, and the value amount between any two subdivision tick marks.

If labeling is enabled, the subdivision of the Y-axis is calculated automatically for the active signal and is displayed in the colour of the signal.

The tick marks are set to whole number values. If a signal is shown as decimal (i.e. raw signal values without conversion), then only whole numbers are displayed. In this case, if an area between two adjacent numbers is shown magnified, no subdivision tick marks will be seen any longer along the Y-axis. For the physical display type subdivisions less than 1 are also shown.

When labeling is enabled the legend also shows the following for each signal: Lower and upper values of the viewing area (i.e. not necessarily the values of the lowermost and uppermost Y-axis tick marks) and the value amount between any two tick marks.

The grid lines in the graphics display always match the tick marks on the Y-axis. Grid lines are enabled or disabled by the **Grid** command in the window's popup menu.

Colors

Here you select the background colour for the window (white or black). Furthermore, you can open the *Options* dialog for configuring signal colours.

Signal Legend

With this function you choose whether the legend for signals should be displayed on the left side of the Graphics window or not.

3.5.10 Export of Signals

With the help of this function you can save the data of one or all signals of the Graphics window to a file. Depending on the activated signal mode (i.e. single-signal or multisignal mode) the Export either applies to the currently active signal or to all signals. This function is only available if data exist for the active signal.

CSV format (Comma Separated Values Format) is supported for export. Possible delimiter symbols are comma, semicolon, tab <TAB> and space <[SPACE]>.

3.5.11 Toolbar of the Graphics Window

The graphics popup menu offers you the option of showing the graphics toolbar either in the main window or in the graphics windows. If you have the graphics toolbar shown in the main window, the functions act on the active graphics window. If no Graphics window is active, the graphics toolbar is deactivated. Click a graphics window with the mouse to activate it.

3.5.12 Optimization of the Graphics Window

The Graphics window is a very powerful but equally complex window in the measurement setup. Its performance capabilities are closely interrelated to the graphics card of your computer and the graphics driver installed. Under certain conditions effects may occur that disturb the graphics display when working with the window. Generally these unpleasant effects in the display can be eliminated if you have Windows redraw the window, for example by modifying the size of the window.

Note:	Please make sure that the Graphics window is not covered up by other measurement windows, menus or dialogs during the measurement. Under some circumstances the performance capabilities of the output may be impaired severely by such overlapping.
--------------	--

Depending on the configuration and bus load the powerful symbol algorithms used in the Graphics window may place a relatively high demand on computer resources. Since individual requirements vary widely, the Graphics window provides special configuration options for optimizing the display.

To optimize the Graphics window, open the **Options** configuration dialog from the popup menu or the window's toolbar. After pressing the **[Measurement]** button you will be presented with the measurement options, where you can optimize the following **Parameters** of the Graphics window:

Buffer size

Here you set the time interval that is to be saved (in seconds) for the signal responses of all signals configured in the Graphics window. For example, if you enter the value 10 here, the last 10 seconds of your measurement in the Graphics window are always saved and will be available to you after the end of measurement for further evaluation. Consequently, high values for this parameter will - particularly when displaying many signals - result in a large memory requirement, but will allow you to track and evaluate the signal response over a correspondingly large time period after the end of measurement.

After the measurement stop, with large signal buffers some functions may become lethargic, such as moving or fitting signals, since in this case large quantities of data must be redrawn. Therefore, you should select a value for the buffer size that is as small as possible.

Buffer limit

In addition to buffer size, you can also specify a buffer limit in kB. This defines the maximum memory usage by the Graphics window during the measurement.

Above all, this is advisable if you have specified a relatively large time span as the buffer size. Without this maximum limit, more and more memory is demanded by the operating system over the course of the measurement. This can lead to severe loading of the overall system due to swapping out of storage. Please refer to online Help for further details.

User-defined refresh

Here you define how often the Graphics window display should be updated. Small values result in continuous display of the signal response, but on the other hand they place a high demand on computing resources and may lead to performance problems with slower computers. High refresh values lessen the demand on computing resources, particular when many signals are being displayed, but they lead to more erratic display of signal responses. You should only input a refresh value if your measurement setup places special demands (high bus load, simultaneous display of many signals, etc.) on the Graphics window. If the **Refresh** check box is not checked, the Graphics window automatically determines a favourable default value. As long as no load problems are occurring during the measurement you should not modify these options.

Scrolling

If the signal curves run into the right border of the Graphics window after the start of measurement, this command will result in automatic tracking of the curves. This involves shifting the time axis to the left, to make room for the measurement signal on the right side. You can configure this behavior, referred to as scrolling, in the Graphics window. **Continuous scrolling** is activated as a default, whereby the time axis is only shifted minimally to the left to give the impression of continuously flowing signal curves.

To shift the time axis in jumps, thus maintaining the graphic diagram at a fixed location between these jumps, you would deactivate the autoscroll mode in the Graphics Window **Settings** dialog. In the line below this you can set the percentage of the displayed time interval by which the time axis should be shifted. The smaller the value, the more evenly the view is scrolled, but it is shifted more frequently. If the time interval displayed in the Graphics window is smaller than the value you specified, the entire Graphics window is reconstructed, i.e. the image is shifted by 100%. This prevents the Graphics window from demanding too much computing power due to frequent scrolling for very short time intervals.

The scrolling procedure always demands more computer resources for smaller time intervals, since scrolling must be done more quickly. Therefore, there is a minimum refresh rate for updating the window contents. If the displayed time interval is of the same order of magnitude as this rate, the signal curves are displayed again in an increasingly jumpy manner, since the window contents are not always updated. You can set the minimum refresh rate in the project file CAN.INI. This involves setting the following value in the [System] section:

```
GraphicWindowMinAutoRefreshrate= <Cycle time in ms>
```

However, please note that the smaller the value you select, the more system loading will increase. Before modifying the automatic refresh rate, it is therefore

essential that you determine an optimal refresh rate with the help of the User-Defined Refresh function.

Note: Some driver-specific problems in displaying the measurement cursors and dotted lines, such as those used to draw extrapolated signals and window grid lines, can be resolved by special options in the file `CAN.INI`. Refer to documentation in section [Debug] of the project file regarding this.

3.6 Write Window

The Write window has two functions in CANalyzer: First, important system messages on the progress of the measurement are output here (e.g. start and stop times of the measurement, preset baud rate, triggering of the logging function, statistics report after the conclusion of measurement). Secondly, all messages which you, as the user, place in CAPL programs with the function `write()` are output here.

The Write window offers several Views which can either be placed freely – together as one separate window – or be shown within the Write window (docked). There are tabs to switch between the following different views:

- All (shows all messages)
- System
- CAPL

The popup menu command **Copy contents to Clipboard** accepts the contents of the Write window in the clipboard. Write window messages serve as both a supplemental report for your measurements and – should problems occur – as a basis for error analysis by our customer service.

For a description of the most important CANalyzer system messages that are output to the Write window, please refer to the online help.

3.7 The Data Window

Data windows serve to display signal values. Signals are understood to be data segments of a message which carry specific information (e.g. engine RPM for CAN buses in motor vehicles). In the Data window you can view signals in a user-friendly manner. When a symbolic database is used the values of the signals specified there can even be displayed as physical variables. For example, engine RPM can be viewed in revolutions per minute, or temperature in degrees Celsius.

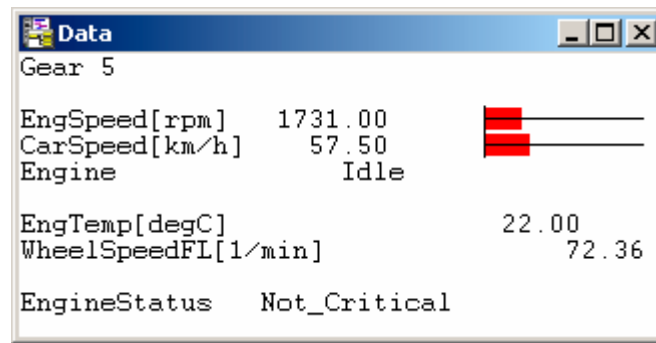


Figure 55: Data Window

Always displayed are the signal name - which can be set in the configuration dialog - and its associated value. You can also decide whether the value should be displayed as a raw datum (hexadecimal or decimal), as a physical value with accompanying unit of measurement, or as a bar chart.

The signal names and values displayed in the window are context-sensitive. When the mouse pointer is moved over them, the element below the pointer is identified by a frame. This element can be dragged to any window location with the mouse, allowing you to **group signals and values** according to your needs.

3.7.1 Configuration of Signals

From the popup menu of any data block or window, a configuration dialog can be opened in which you can enter the signals that should be displayed in the particular window, add or delete signals, or set their representation parameters.

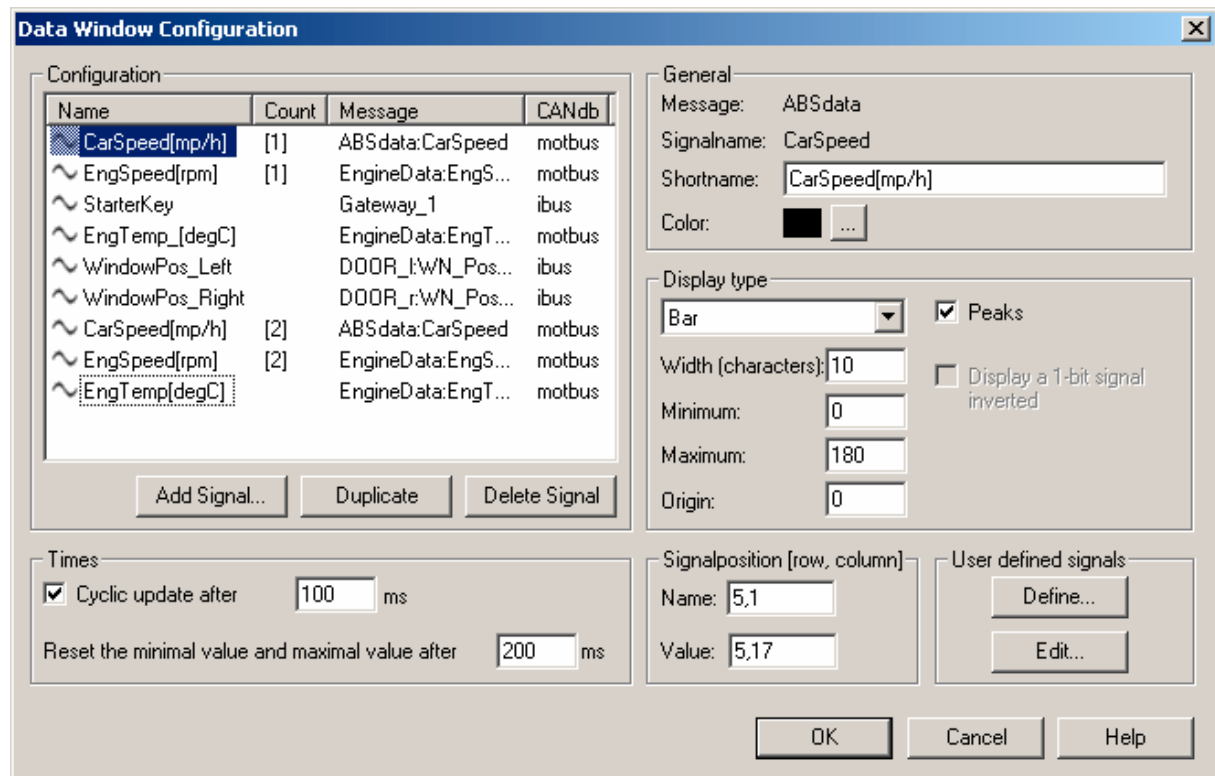


Figure 56: Data Window Configuration

All signals of the signal list are shown in the list box at the upper left of the dialog. Here you can accept new signals from the associated database. When copying, the abbreviated signal name is modified such that no ambiguities result. The abbreviated name appears during the measurement as the signal identifier in addition to the actual signal value.

With the **[Define]** and **[Edit]** buttons you can enter a new signal description - independent of the database - in the Data window or modify an existing signal description.

3.7.2 Display Types

Signal values can be displayed in the Data window in the modes shown below. The output width of the signal value is automatically calculated from the bit length and conversion formula. CANalyzer takes the necessary signal data from the database:

Physical

The raw data are extracted from the CAN message, scaled by the (linear) conversion formula, and displayed as decimal fixed point numbers.

Example: Signal T consists of 5 bits, is unsigned, has a factor of 10 and an offset of 0. Then the possible raw value range is 0-31, and the physical value range is 0-310. Therefore, the necessary output width is 3 characters.

Display type: Decimal

The raw data are extracted from the CAN message and displayed as decimal numbers.

Example: Signal T consists of 5 bits and is unsigned. Consequently, the possible raw value range is 0-31, and the necessary output width is 2 characters.

Display type: Hexadecimal

The raw data are extracted from the CAN message and are displayed as hexadecimal numbers.

Example: Signal T consists of 8 bits. The possible raw value range is then 0-255, or in hexadecimal representation 0-FF. The necessary output width is therefore 2 characters.

Display type: Bit

The raw data are extracted from the CAN message and displayed as binary numbers.

Example: Signal T consists of 8 bits. The necessary output width is also 8, since all bits are shown individually.

Display type: Bars

The physical signal value is displayed in the form of an analog bar. The following parameters must be configured for the display:

Min, Max: Minimum and maximum signal values to be displayed by the bars.

Width: Size of the bar that is displayed. The unit corresponds to the width of one character in the Data window.

Center: This defines the zero position of the bar.

With the help of parameters Min, Max and Center, it is possible to display just a section of the signal's value range. For example, this could be the working point of a signal to be monitored.

Display type: C-Style

You should only select this option if you are familiar with the programming language C or C++. In the dialog's format input box you would enter a format string that is used to display signal values in the Data window.

The signal's raw data (without conversion formula) are extracted from the CAN message, converted to a 32 bit value, and output with the help of the `sprintf()` C function:

```
sprintf(buffer, FORMATSTRING, (unsigned long)SIGNALVALUE)
```

The format string is compatible with the `printf()` format string of C. Due to the transformation to `unsigned long` an L-modifier must always be entered. Legal interpretations are "lx", "lX", "ld", "f" and "F". You can also use width and precision modifiers.

Examples of format strings: %5.2f, %8lx, %-3ld

The Data block attempts to recognize width specifiers in the format string to determine the output width for the screen. If this is not possible, the maximum width is assumed. This does not influence the output in the Data window.

Please note that invalid format strings can lead to unpredictable results as serious as system crashes. As a user you are fully responsible for the validity of the format string!

3.7.3 Activity Indicator

The signal values of the last message received in the Data block remain visible in the Data window until they are overwritten by new values. With signal values that change slowly, the user can determine whether the signal was updated with the same value or whether the particular message was not transmitted any longer and consequently an older signal value is being displayed in the Data window. This information is provided by the Data window's activity indicator: If a message is received and recorded with an unchanged signal value an alternating slash ('/' or '\') appears after the value. If the slash is not displayed, this means that the displayed constant signal value is not current, since the particular message was not received.

3.7.4 Peak Indicator

With signal values that change very quickly over time, transient minima and maxima can easily be overlooked. To make these transient peaks visible, the Data window therefore has a peak display. This allows you to recognize very short signal fluctuations as "afterglows" in the bar-type display. You configure the "afterglow duration" in the **Configure Timers** dialog, which is called from the Data window's popup menu.

With the **Peaks** option box you can configure - in the Data window - whether minimum and maximum values should be displayed in addition to the active value. If the active signal value is greater than the previous maximum, the maximum is updated immediately to the new value. It is not reset to the active value until after a specific time interval. The same principle applies to the minimum.

The minimum and maximum values are only shown in bar-type displays. With all other display types only the screen color changes when the minimum or maximum value deviates from the active value.

3.7.5 Optimization of Data Display

From the Data window's popup menu you can open a dialog for configuring the display's time behavior during the measurement.

In the lower area of the dialog are option buttons used to enhance the performance of the Data window for large quantities of data. In the normal case you should have the default option **Refresh Immediately** activated. In this case the Data window output is updated after a message is received. Additionally, the window is redrawn cyclically every 2 seconds.

To permit the processing of large quantities of data you should activate the option **Only Refresh Cyclically**. This results in the window only being updated cyclically. If there are high message rates, this can lead to a significant reduction of system load-

ing. However, to obtain reasonable outputs in the Data window you should not select a cycle time greater than 500 ms.

3.8 Statistics Window

The **Statistics block** fulfills three different functions. One of these is to display the average time between the sending of messages (secs/msg) during a measurement. It can also display the messages per second. These are done by constructing a continuously updated **line histogram** above a message identifier's axis. A sliding scale averaging method with an adjustable averaging time is used. The other function keeps statistics on all bus activities in the background; these results can be reported either as a **statistical report** in the Write window or stored via a histogram function and then processed further.

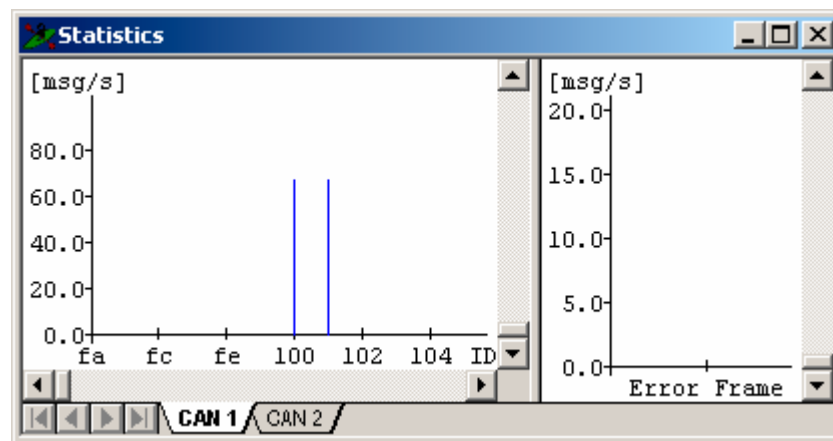


Figure 57: Statistics Window

The Statistics window displays (see Figure 57) the mean message rates existing at the end of the measurement. The Write window contains the statistics report (see Figure 59).

3.8.1 Direct Display in the Statistics Window

During the measurement either the mean transmit interval or the mean message rate is displayed in the Statistics window. Smoothed averaging is used with adjustable averaging time. The message identifiers are output on the horizontal axis, and the corresponding rates on the vertical axis. The IDs are distributed according to whether they originated from controller CAN1 or CAN2, and by message attributes Rx and Tx:

	Rx	Tx
CAN1	RED	BLUE
CAN2	RED	BLUE

In the statistics configuration dialog you can define whether the window diagram should show the mean transmit interval (sec/msg) or its inverse, the mean message rate (msg/sec). Also configurable is the **averaging time** which defines the time inter-

val at which the display is refreshed. Averaging is most precise with a low value, but this demands a lot of computing time. With high values the statistics display lags behind. An averaging time of approx. 1000 ms gives satisfactory results for standard applications.

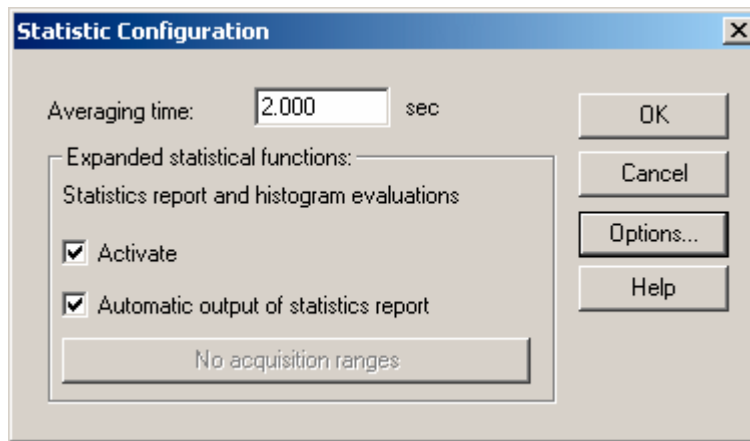


Figure 58: Configuration of the Statistics Block

In **standard** mode the messages of the channels are displayed side by side. In **tab view** mode the window is split. In the left part (standard view) the messages are shown. In the right part (**special view**) special events are shown, e.g. error frames. With the tabs on the bottom of the window you can switch between the buses. You can have up to 3 standard views but only one special view for each bus channel.

You can scale the Statistics window from the popup menu. The functions available for this, such as **Zoom**, **Fit**, **Basic Image** and **Manual Scaling** are described in detail in online Help.

Note: If the CAN card used supports extended identifiers, the function **Basic Image** is split. The user can choose whether scaling will be over the range of standard identifiers or over the entire range.

3.8.2 Statistics Report

In background, statistics are kept on all bus actions, and the results can be reported to the Write window after conclusion of the measurement. A list is constructed (sorted by message identifiers) which contains information organized separately for receive messages, transmit messages, transmit requests and transmit delays: Number of messages, mean time spacing, standard deviation, minimum spacing, maximum spacing.

When you select the option **Activate** under **Expanded statistical functions** in the statistics configuration dialog all data will be accumulated for the report. However, this of course requires computer resources and can slow down very fast bus traffic.

You can output the statistics report to the Write window either automatically or via the menu entry **Display statistics report** after the end of a measurement.

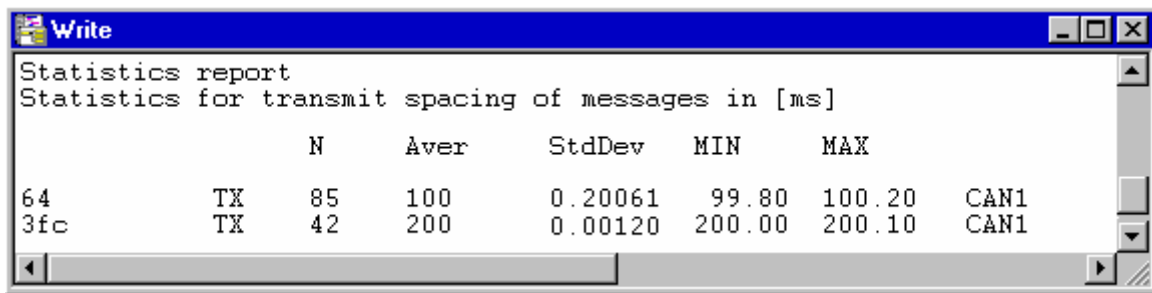


Figure 59: Statistical evaluation of a measurement in the statistics report

Afterwards, the Write window shows - for each occurring ID coded by attributes - the total number of messages, the mean transmit interval, its standard deviation, minimum and maximum.

Note: The function **Display statistics report** can only be selected if the **Statistic report... Activate** option was selected during the measurement.

3.8.3 Choosing a Histogram

When the upgraded statistics functions are activated, data is collected for interpretation during measurement runtime. Several time periods for collecting data can be recorded via the histogram function. The beginning and end of these time periods can be determined from **New evaluation** and **Stop evaluation** in the popup menu of the Statistics block or with appropriate CAPL functions.

Note: Stopping the measurement also ends data gathering for the the time range being measured at the time. Evaluations can be recorded from several measurements started one after the other in the same configuration.

The data obtained can be further evaluated with the histogram dialog box (menu item **Evaluation output**) by choosing regions of interest from the data included in the time range.

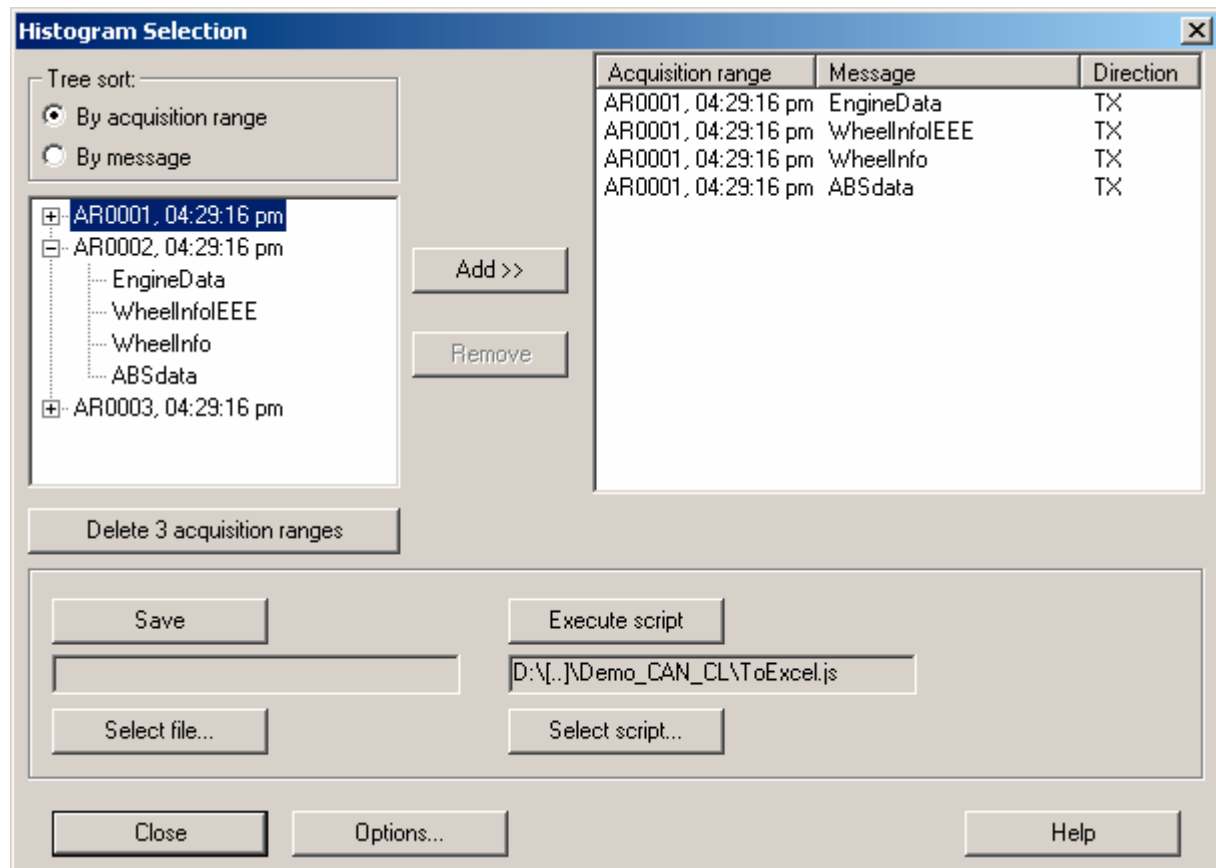


Figure 60: Histogram Selection

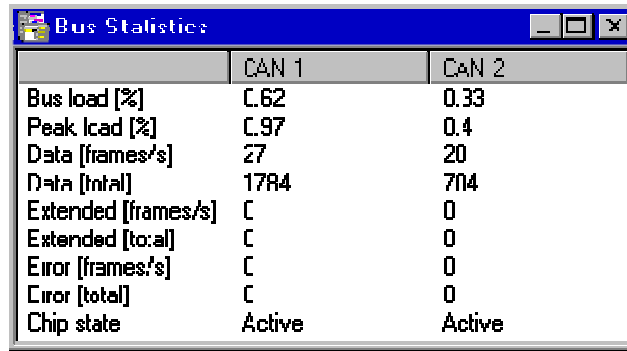
The appropriate data selected can be stored as a CSV file (values separated by commas) or transferred to Excel with the script toExcel.js. The data can also be manipulated with the user's own Java or Visual Basic script.

The collected data can be reset in the configuration dialog box of the statistics block.

Note: Changing the configuration deletes all recorded ranges of coverage.

3.9 Bus Statistics Window

CANalyzer acquires statistical data over all CAN channels used. The data acquired depend on the hardware used. The results are displayed in the Bus Statistics window. The statistical functions include the rates and number of Data and Remote frames (11 and 29 bit), Error frames and Overload frames. Also displayed are the values for momentary and maximum bus load. The state of the CAN controller is shown as ACTIVE, ERROR PASSIVE or BUS OFF. Also displayed for certain hardware platforms are the value of the CAN controller's Tx error counter and Rx error counter.



	CAN 1	CAN 2
Bus load [%]	0.62	0.33
Peak load [%]	0.97	0.4
Data [frames/s]	27	20
Data [total]	1784	704
Extended [frames/s]	0	0
Extended [total]	0	0
Error [frames/s]	0	0
Error [total]	0	0
Chip state	Active	Active

Figure 61: Bus Statistics Window

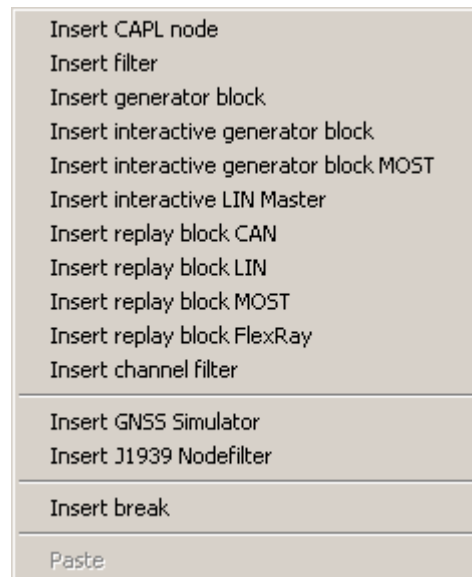
Under **Options** in the card icon's popup menu in the measurement setup window, the user can configure the time interval at which the card passes bus statistics information to CANalyzer. This interval defines the frequency of the bus load measurement and thereby also the averaging time. The default value is one second. For measurements with extreme data throughput bus statistics may be deactivated to increase performance. Above all, this affects the FIFO buffer between CANalyzer and the CAN card. The error message "Rx buffer overrun" could possibly be prevented by doing this.

Bus statistics information is also recorded in logging (cf. section 2.6). To include this information in logging activate the **Log internal events** check box in the configuration dialog for the log file. When the file is played back in Offline mode this information is then displayed again in the Bus Statistics window. The Bus Statistics window remains empty in Offline mode if the data source does not contain any bus statistics information.

4 Blocks and Filter

In the measurement setup there are square points (hotspots) between the basic function blocks, at which additional function blocks can be inserted or the data flow can be blocked. The hotspots themselves allow all data to pass unhindered.

When a hotspot is clicked on with the right mouse button (or selected with the cursor keys and then activated with <F10>), a popup menu appears with the following menu commands:



If one of the first five menu commands is clicked on, a block is inserted in the data flow plan which satisfies the particular function. New hotspots appear before and after this block, so that additional blocks can be inserted. The last menu item is a special case. If it is activated a broken hotspot appears, which is intended to show that the flow of information is blocked at this point.

Function blocks can be recognized by their appearance or by their labels in the data flow chart. A **P** stands for a CAPL node (Program block), **PF** and **SF** designate the Pass and Stop Filters, **PE** and **SE** are the corresponding filters for environment variables, a **G** refers to a Generator block, an **IG** to the Interactive Generator block and **R** stands for a Replay block. The channel filter is represented with a special icon.

All blocks have popup menus which the user can open by either clicking with the right mouse button or by selecting the block in the data flow plan and then pressing <F10>. The first menu item opens a configuration dialog for the particular block, which only serves to parameterize the block (there are two configuration dialogs for the generator block, which can be opened by the first two menu items). By selecting the last item in the popup menu **Delete this node**, you can remove the block from the data flow plan. All configuration information is lost in the process. However, the CAPL source files of the CAPL node and the log file of the replay block are not deleted.

From the popup menu you can assign a comment to each function block and analysis block. The comment is displayed by default. With the function **Show comment** in the popup menu you can enable and disable the display separately for each block.

Please note that the size of the displayed comment depends on the actual position of the function block and the space available for the display. Independent of this, it is always the case that only the first two lines of the comments are displayed. It is advisable to use only short key words in the first two lines and to enter more elaborate information further below in the text.

If you have not entered any comment, but the display is nevertheless activated, the predefined comment is displayed.

4.1 Generator Block

The purpose of the generator block is to create messages for transmission. Trigger conditions and a transmit list are defined for this purpose. A received message, a key press or a time period can be entered as a trigger condition. The trigger conditions can also be used in combination. In the transmit list, messages (and error frames) are entered in the sequence in which they are to be transmitted.

Whenever a trigger condition occurs the next message is transmitted. When the end of the list is reached, transmission may resume at the start of the list, depending on the configuration.

Since generator blocks can be configured to be very elaborate, the popup menu offers you the option of saving a generator block as a file and later reloading it, possibly from a different configuration. As a result, you can easily exchange generator blocks between different configurations.

Generator blocks appear in the data flow plan as small blocks with the label **GB**.

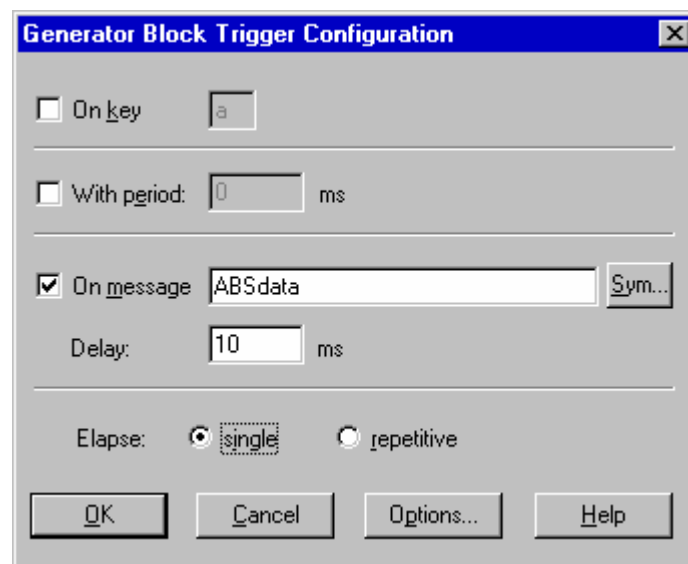


Figure 62: Generator Block Configuration - Triggering

Note: To have a generator block send data onto the bus, you must insert it in CANalyzer's transmit branch. If you insert it in the evaluation branch the data will indeed be sent to the evaluation blocks to the right and be displayed in the appropriate windows. However, they will not reach the bus due to the left-to-right directional data flow.

4.1.1 Configuration of Triggering

In the popup menu select the item **Triggering** to define a trigger condition for transmission of messages.

You can select one or more of the three possible conditions by marking the appropriate check box(es):

- with **On Message** you indicate which identifier should trigger transmission, and the delay that should occur.
- with **On Key** you indicate the key that should trigger
- with **On Period** you enter the period in milliseconds.

In the line **Flow** you define whether the transmit list should be run through exactly once or cyclically.

4.1.2 Configuration of Transmit List

In the popup menu select the item **Transmit list** or double click the generator block to create the list of messages to be transmitted:

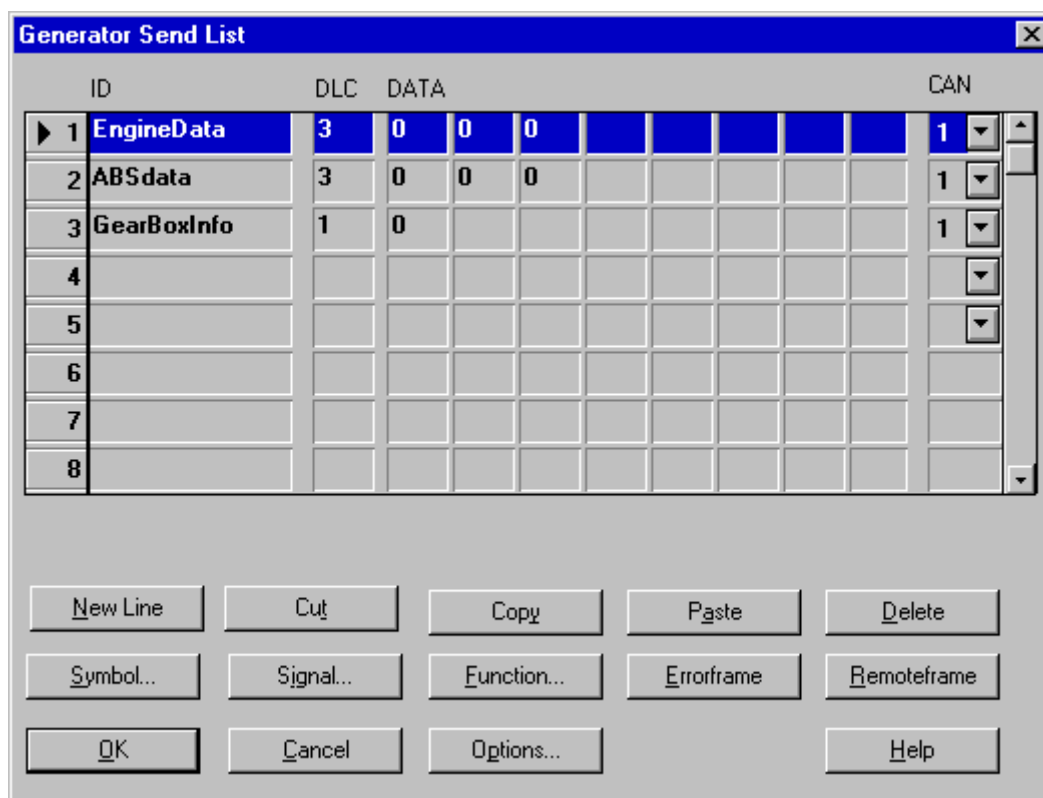


Figure 63: Generator block Configuration of Transmit list

When one of the trigger conditions occurs, the next element of this list is transmitted. When the end of the list is reached the program resumes with the first element if the run mode has been set to *cyclic*. The list may also consist of only one element.

Nine lines are displayed in the dialog box. The *active line* - to which the dialog buttons refer - is identified by the "▶" symbol at the beginning of the line. The active line is shifted automatically by activating the <TAB> key or by clicking the dialog entry boxes with the mouse.

Each line of the list consists of 11 columns. In the first column you enter the desired identifier. The DLC field defines the number of data bytes in the message. After that come the fields for data bytes. Only the number of bytes specified by the DLC are accepted. The rest are ignored. The last column is a combination box for selecting the controller by which the message is to be transmitted.

4.1.3 Entry of Signal Values

You can configure the generator block symbolically, provided that you are working with a database. Use the option button **[Symbol...]** to select a message from the database which is to be transmitted from the generator block.

Then the physical signal values of the message can be defined with the **[Signal...]** button.

Displayed in the signal value input dialog are the individual signals of the message that was on the active line of the transmit list. If the message only contains standard signals, the signal list consists of three columns. The individual signals are listed in the lines. In the last column you can prescribe a signal value. If you have assigned a value table to the signal in the database, you can use the relevant symbolic descriptor instead of a numeric value. You can select this from the signal value table in the middle column.

Physical dimensions are stored in discrete form in the CAN messages. However, it may not always be possible to represent the numeric value entered in the **Value** box as a discrete value. In such cases, when exiting the line or activating the **[OK]** button, the two next possible physical values are displayed in a dialog. Then the entered value is rounded to the next possible closest value.

Example:

The signal EngineData.RPM is defined as a 16 bit unsigned with an offset of 0 and a factor of 10. If the value to be compared is entered as 1015, the raw value would have to be $1015/10 = 101.5$. Since only discrete values may occur, either 101 or 102 must be used, which correspond to the physical values 1010 or 1020. It is these two numbers which appear in the dialog.

In spite of discrete memory storage, quantities which have digits after the decimal point can be valid. In the example above, if a factor of 10.5 is used for calculation 1008 and 1018.5 are recommended as possible values.

4.1.4 Entry of Mode-Dependent Signals

Generally there is a direct, fixed assignment of a message's data segment to a signal. However, for multiplex messages different signals are transmitted in the data segments, depending on a mode signal. Only a subset of all possible signals of the

message is defined for a mode value. In the dialog for inputting signal values, the standard signals and mode-dependent signals are shown in two separate list boxes for ease of input. Additionally, all those signals not defined in the active mode are filtered out in the mode-dependent list box.

The *Mode signal* and the *Mode* are displayed in the associated Edit boxes. If the mode value is changed, the list with mode-dependent signals is reconstructed. The mode signal itself cannot be changed here.

4.1.5 Function Generator for the Transmit List

It is often the case that the user wishes not only to place certain signal values on the bus, but rather entire signal responses. The generator block has a signal response generator for this purpose, which you configure with the **[Response...]** button.

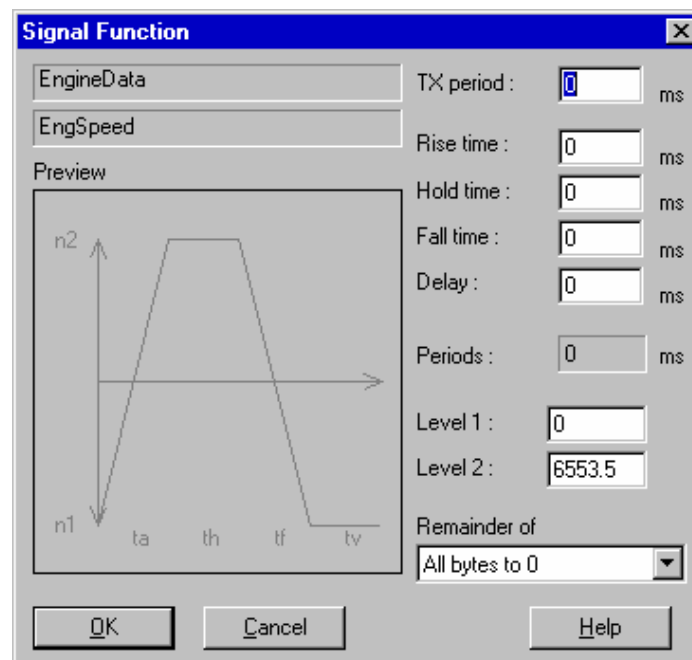


Figure 64: Signal Response Generator

The configuration dialog allows you to parameterize a trapezoidal function. When the dialog is exited with OK the corresponding lines are automatically generated in the generator block transmit list. The following signal responses can be generated with certain parameters:

Signal response	Parameter
Square	$ta = tf = 0$
Triangular	$th = tv = 0$
Saw-tooth	$th = tf = tv = 0$ or $ta = th = tv = 0$
Constant	$n1 = n2$

Displayed in the upper part of the dialog box are the message and the signal. Shown below this, in the preview box, are a trapezoid and the meanings of its parameters. By setting individual parameters to zero, the following responses can be generated:

The levels n1 and n2 must be entered physically. The relevant limitations apply here (see for example generator block signal values).

The entry for transmit interval identifies the interval between any two messages, and this corresponds to the entry in the generator block dialog **Trigger initiation period**. Since CAN is message-based and not signal-based, all signals of a message get the same transmit interval!

Since this signal characteristic generator is only a tool for the generator block and is not a block itself, the following must be observed when using different period lengths for several signals within one message:

The generator creates a list of messages with:

$$\text{No. of messages} = \text{Period} / \text{Transmit spacing}$$

This is exactly one period. If the transmit list contains more they are rejected. Using the combination box **Remainder of** the user can define how remaining signals are to be handled for supplementally generated messages:

- All bytes to 0:
All signals are set to the raw value 0.
- Continue cyclically:
During generation the previous messages are copied as often as necessary until the new period length is reached.

Example:

The original list contains 3 messages, and the new period length requires 9 messages. The new signal (byte 1) is a saw tooth.

Original Transmit List:	New Transmit List:
0 0 0 4 0	1 0 0 4 0
0 1 0 2 0	2 1 0 2 0
0 2 0 4 0	3 2 0 4 0
	4 0 0 4 0
	5 1 0 2 0
	6 2 0 4 0
	7 0 0 4 0
	8 1 0 2 0
	9 2 0 4 0

The recommended procedure for generating multiple signals is therefore as follows:

The signals are defined beginning with the shortest period length. Then only integer multiples of this period length are used.

4.2 Interactive Generator Block (IG)

The purpose of the interactive generator block is to generate and transmit messages.

Messages can also be configured and interactively transmitted during a measurement (Online). This makes the interactive generator block especially well suited for influencing a measurement in a quick and improvised way. In many cases, with the interactive generator block you can achieve your goal without the use of traditional generator blocks and without CAPL blocks.

Interactive generator blocks appear in the data flow plan of the measurement setup as small blocks with the label **IG**. Just like traditional generator blocks they are permeable to all data in the data flow diagram. That is, they do not filter the data flow like filter blocks or CAPL blocks do, rather they act in a purely additive manner.

Tip: In each of your CANalyzer configurations, insert an interactive generator block in the transmit branch of the measurement setup. This provides you with a flexible tool, even during a measurement, with which you can spontaneously influence the message traffic on the bus.

For examples of interactive generator block implementation please refer to the CANalyzer Help function.

4.2.1 Configuring the Interactive Generator Block

The **configuration dialog** of the is subdivided into a **Transmit List** (upper half of window) and a **Signal List** (lower half of window). In the Transmit List you can select individual messages and configure them. Assigned to each message is a Signal List in which signal values can be configured.

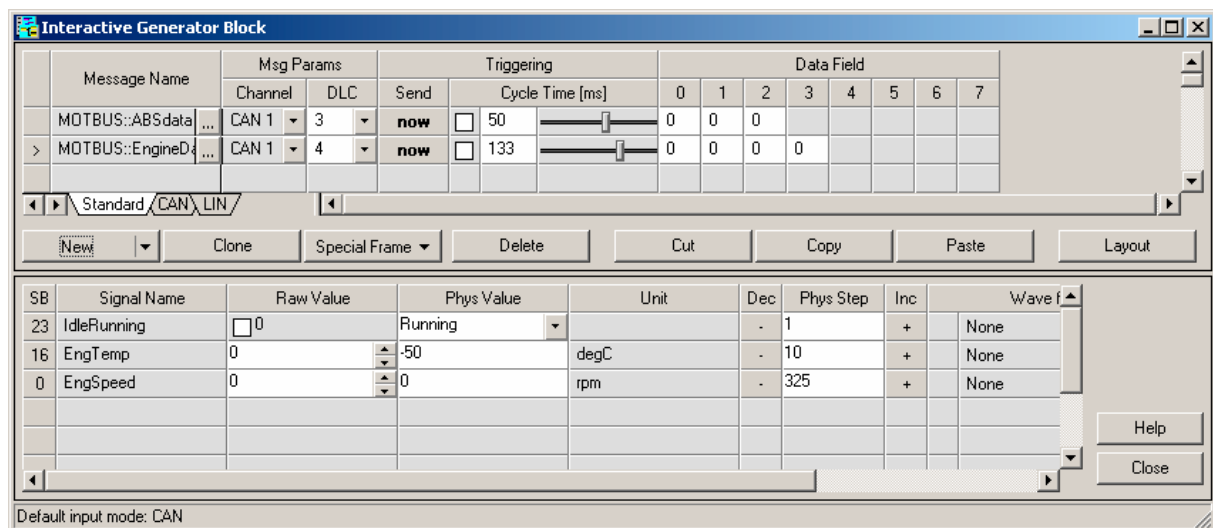


Figure 65: Configuration of the Interactive Generator Block

All input boxes are explained in the status bar at the lower border of the dialog. The explanation is always shown for the currently selected box entry. There are also keyboard shortcuts for use without the mouse; these are described in section 4.2.1.7.

4.2.1.1 Transmit List

The list of messages to be sent (Transmit list) is available in two views: **Rows** and **All Columns**. In the Rows view the most important columns of the transmit list are summarized in an easy-to-read format, while in the All Columns view even seldomly used message parameters and trigger conditions appear. The widths of the columns can be adjusted from their borders. The **[Layout]** button is used to have the columns automatically configured to the most easy-to-read layout.

The **[New]** button is used to input messages in the transmit list. The procedure differs depending on whether or not a database is associated to the CANalyzer configuration:

If there is a database, messages from the database can be added to the transmit list. Clicking the **[New]** button opens the Message Explorer with which messages can be selected symbolically. They are then assumed in the transmit list. Afterwards the signals of the messages can be configured individually in their signal lists. An alternative to activating the **[New]** button is to double click with the left mouse button in the first empty line of the message name column of the Message Explorer.

Without a database you can transmit any desired messages by manually entering the CAN identifier of the desired message in the transmit list (*Identifier* column). To generate a new message click in the first empty cell of *Identifier* column with the left mouse button or press the *New* button. After selecting the desired number of data bytes for the message (*DLC* column) you can then input the data bytes (*Data field* columns). The signal list remains empty.

Note:	The [ErrorFrame] button can be used to enter an Error frame in the transmit list. This allows Error frames to be transmitted on the bus. Remote frames can be created from normal messages in the transmit list, except that in the Frame message parameter column the description Remote is selected instead of Data .
--------------	--

4.2.1.2 Value Generator

In addition to already existing messages/signals new generated signals can be transmitted too.

The following generator types are available:

- Toggle switch
- Range of values
- Ramps and pulses
- Random
- Sine
- User defined

You can find details for using the different types in the online help.

4.2.1.3 Trigger Condition

The trigger condition (*Triggering* columns) is, in contrast to the traditional generator block, entered separately for each message.

You can choose between manual interactive triggering (**[Send]** button), key press (Key column) and interval-driven repetition (Timer; Cycle time column [ms]). Additionally, the user can configure the number of messages to be sent at the time of triggering (Burst column). The default value for Burst is 1, i.e. exactly one message is sent per triggering.

4.2.1.4 Generating a High-Load Situation

With the **High load** trigger condition you can induce a high load situation. A high bus load is reached when messages follow right after one another on the bus. To achieve this, after successful transmission of one message the next transmit request of the same message must follow immediately. The number set in "Burst" indicates the current number of messages in the transmit queue.

Note:	When high load is configured no other simultaneous triggering of the same message should be performed manually by keyboard or by timer, since the transmit queue might overflow. Furthermore, the associated receive acknowledgments (Tx or TxRq events) in front of the interactive generator block in the data flow plan must not be filtered out; otherwise the transmit queue will not be refilled.
--------------	---

4.2.1.5 Entering Signal Values

The list of message signals is taken from the database from which the particular message originates. If no database is associated, the list remains empty.

The list contains - for each signal - the signal name, raw value, physical value and units. The signal name and units are taken from the database. The raw value and physical value are adjustable. Possible value ranges are shown in the status bar at the bottom of the dialog.

You only need to enter the raw value or physical value, since the corresponding value is always also calculated by the signal's conversion formula. If a physical signal value is entered whose raw value equivalent lies between two raw values and therefore cannot be represented, it is automatically rounded up or down to the next closest value.

If the signal is of the type **Enumeration (Enum)**, the defined value descriptions are presented to you in a selection list in the input box for selection. This permits symbolic selection of the signal value. For bit signals a check box is provided, which makes it easy to toggle the bit value.

The three columns on the right represent a decrementer and incrementer, with which the physical signal value can be easily changed by an amount that is set in the Phys. Step column. Shown at the far left is the start bit **SB** of the signal within the message.

4.2.1.6 Entering Mode-Dependent Signals

As a rule, a message's data segment has a direct, fixed assignment to a signal. With multiplex messages a mode signal is used to assign different signals to a data segment, so-called mode-dependent signals. The mode value indicates those signals that are currently valid from the set of all possible message signals. That is, only a portion of all possible signals of the message are defined for a given mode value.

In the configuration dialog for the interactive generator block mode signals and mode-dependent signals of a message appear highlighted in color in the signal list. Mode signals are light green, and mode-dependent signals are shown in dark green. When the mode is changed in the mode signal, its associated signals are shown. All signals not defined in the currently set mode are filtered out.

4.2.1.7 Keyboard Control

The following keyboard shortcuts apply to work in the interactive generator block:

Spacebar	Toggles the states of check boxes and activates the Send buttons. Opens the Message Explorer to the message name fields
Arrow keys (←, →, ↑, ↓)	Switches between the various buttons and input boxes. The box entries are explained in the status bar at the dialog's lower border.
<Ctrl><↑> <Ctrl><↓>	Increments or decrements the value of the active box (by 1 for timers and data bytes of a message, and by the lowest possible raw value difference for signal values on a signal line)
<Ctrl><←> <Ctrl><→>	Logarithmic calibration of the slider in the timer columns of the transmit list
<Page ↑> <Page ↓>	Increments or decrements a signal value on a signal line by the configured step width (Phys. Step).
<Esc>	Exits the editing box, whereby any change is canceled and the previously set value is restored. Attention: <Esc> during a running measurement will terminate the measurement!
<F9>	Starts the measurement

Note: As long as the configuration dialog for the Interactive Generator block is opened and active, all keyboard entries except <F9> and <Escape> are used to edit the dialog. Therefore the defined key can only initiate transmission by the generator blocks or activation of CAPL program nodes provided that the CANalyzer main window is activated, or test mode is explicitly activated in the configuration dialog.

4.2.2 The Interactive Generator Block as a Gateway

The IG additionally offers a gateway function. With this function, you can transfer information from one bus to another. Necessary inputs you have to do in the IG configuration dialog.

Therefore CANalyzer offers two modes:

- **Transfer of chosen signals**

Activate first the register "All columns" of the dialog's upper list. Choose the desired signal with the **[New]** button afterwards.

- **Transfer of the whole bus communication**

If you choose the * sign as identifier, the whole bus communication will be transferred from one bus to the other.

If you transfer the whole bus communication, you also can build additional rules for signals. These rules have priority for the relevant signals.

You can find a detailed description in the online help.

4.3 Replay Block

The replay block makes it possible to replay measurement sequences which have already been recorded. To do this, the user specifies a log file. Messages contained in it are introduced into the data flow. The most important application is replaying a recorded data stream onto the CAN bus. To do this, the replay block must be inserted in the transmit branch.

Replay blocks appear in the data flow plan as small blocks with the label **R**. Double clicking these blocks causes a dialog box to appear, in which the replay block can be configured.

The user can specify whether RX messages or TX messages are to be transmitted or not. The user can also choose whether messages originating from CAN controller 1 should be transmitted on CAN 1 or CAN 2, or should not be transmitted at all, and similarly for messages originating from CAN 2.

The file can be transmitted once or cyclically. For cyclic transmission, transmission resumes with the first message after the end of the file is reached.

There are three possibilities to define the **start of transmission** of the first message of the file.

- **Immediate**

The first message is transmitted at the start of measurement.

- **Original**

The time of transmission is defined by the time saved with the message in the file.

- **Specified**

The user sets the time explicitly in milliseconds since the start of measurement.

In all three cases the time spacing between messages within the file is preserved. If it is less than one millisecond, transmission is delayed accordingly.

The configuration is lost when a replay block is removed from the measurement setup (by selecting the command **Delete configuration** in the popup menu). This only affects the options set in the dialog box. The replay file itself is not deleted.

Note: To make a replay block capable of sending data onto the bus, you must insert it in CANalyzer's transmit branch. If you insert it in the evaluation branch, the data are indeed sent to the evaluation blocks to the right of the replay block and are displayed in the associated windows. However, the data do not reach the bus, due to the left-to-right directional flow of data.

4.4 Trigger block

The trigger block is the same as the trigger for the logging (block) configuration. You can place this trigger not only in front of the logging block but also everywhere (Hot-Spot) in the measurement setup.

You can find more about the trigger configuration in chapter 2.6.1 (page 46) and in the online help.

4.5 Filter block

The volume of data can be selectively reduced by using the filter block. Toggling between a **pass** filter and a **stop** filter will pass or block those identifiers and/or identifier ranges that are specified. This is done with the button **Filter type**. All messages of a network node can be filtered as well.

In addition, the message type affected by the filtering function can be set for the identifier, as well as whether filtering should also apply to error frames.

Note: If you have installed CANalyzer Option LIN you can find more information in the manual's chapter Option LIN or in the online help.

Filter blocks are entered in the data flow plan by right clicking a hotspot, and appear as small blocks with the label **PF** (Pass Filter) or **SF** (Stop Filter). When these blocks are double clicked a configuration dialog appears in which the filter can be parameterized or selectively deleted by specifying messages (ranges), error frames, network nodes and attributes.

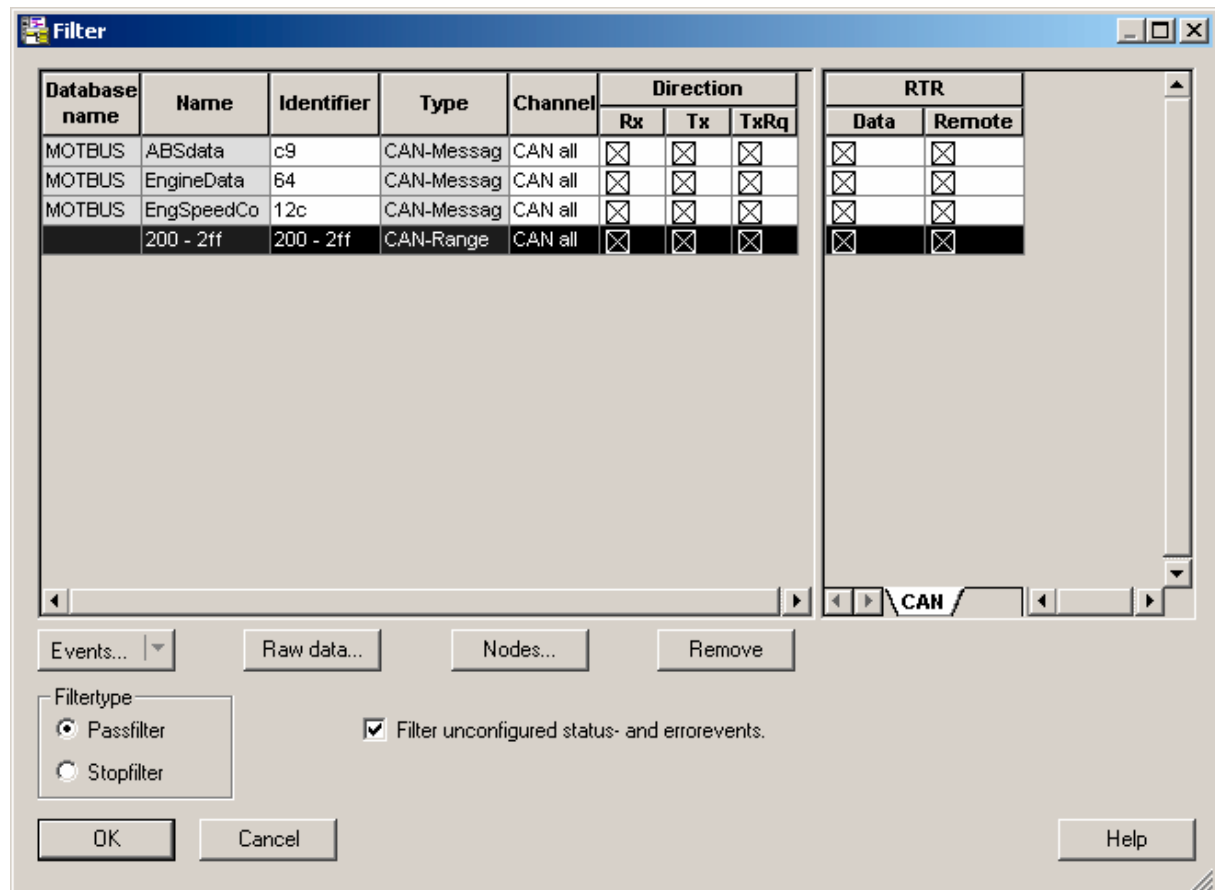


Figure 66: Filter Configuration Dialog

The filter configuration is lost when a filter block is removed from the measurement setup (by choosing the command “Delete this node” in the popup menu or “Del”).

Note: In keeping with its function, a pass filter which is not configured (empty) does not pass any messages and so blocks all message traffic. If older configurations are opened with Version 3.1, their old pass and stop filters will reappear.

4.6 Channel Filter

It is possible to completely block all messages on a channel or to let them pass with a channel filter.

The channel filter can be seen in the data flow plan as a small block in which the actual setting is graphically shown. Blocked channels are represented by broken red lines; those which are not blocked appear as green lines.

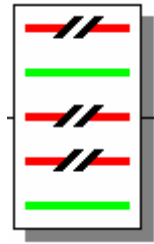


Figure 67: A Channel Filter in the Data Flow Plan

Double clicking on the filter symbol opens a configuration dialog in which the available channels are seen and can be set.

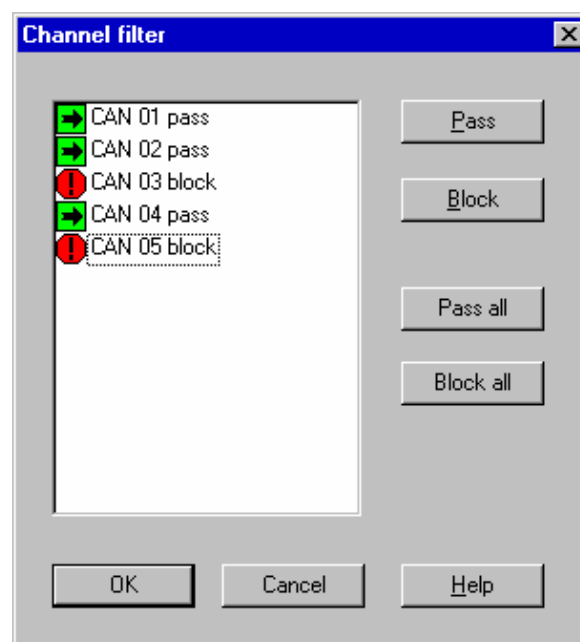


Figure 68: Configuration Dialog For Channel Filter

The filter configuration is lost when a channel filter is removed from the measurement setup (by choosing the command “Delete this node” in the popup menu “Del”)

Note: A channel filter which has not been configured can be used in the data flow plan to simply show the number of channels being used.

4.7 CAPL Nodes in the Measurement Setup

A CAPL node is a universal function block whose characteristics the user defines by writing a CAPL program. See section 5 for a detailed description of CAPL program blocks and instructions on creating CAPL programs.

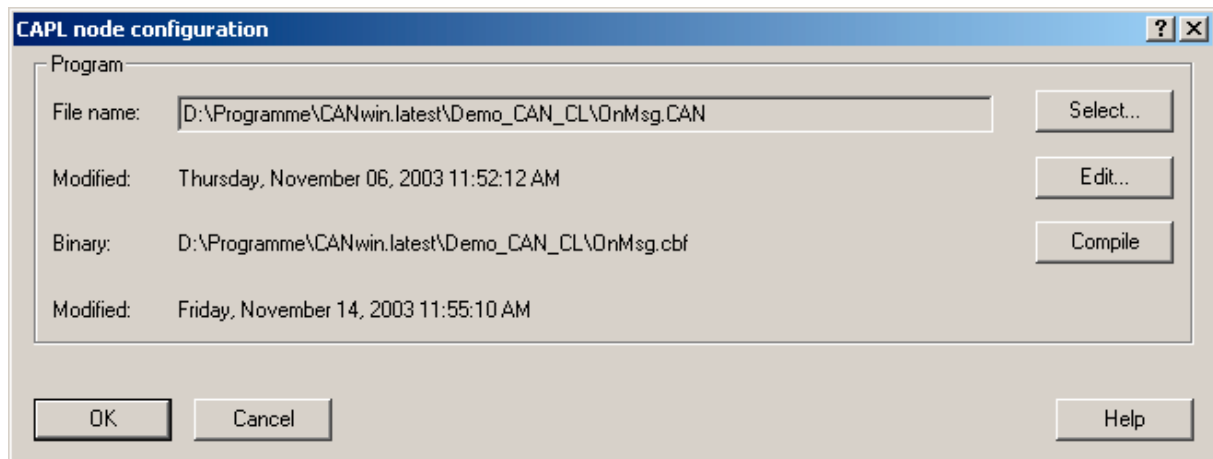


Figure 69: Configuration dialog for CAPL programs

An important use of program blocks is e.g. to preset transmit messages in the transmit branch. However, even very simple program blocks can be written for data reduction (Example: Only pass every tenth message) or for monitoring.

Note: A CAPL node in a data flow branch blocks all messages that are not explicitly output in the program with `output()`. A program that is transparent for all messages must therefore contain the following message procedure:

```
on message * {
    output(this); /* Pass all messages */
}
```

Consequently, the intentional use of `output(this)` also permits the programming of evaluation branch filters using CAPL programs whose functionality can be much more complex than that of normal pass or blocking filters.

Program blocks appear in the data flow plan as small blocks with the label **P**.

In the configuration dialog, first assign a CAPL file name (Extension `*.CAN`) to the program block.

Press **[Edit]** to open the CAPL Browser. Browser is an easy-to-use tool for creating, modifying and compiling CAPL programs and is described in detail in section 5.2.

Before you start the measurement you must compile the CAPL file. To do this, press the **[Compile]** button or choose **Configuration | Compile all nodes** in the main menu to compile all CAPL programs at once.

Note: It is permissible to reference the same CAPL programs in different program blocks. For example, this may be of interest if the identical data manipulations are to be performed in two different data flow branches (e.g. data reduction operations).

You can deactivate the node by pressing the spacebar or by selecting the line Node active in the CAPL node's popup menu. The node can be reactivated by repeating

the same action. The menu command **Delete this node** removes the CAPL node from the measurement setup.

Note: When a CAPL node is removed from the measurement setup, the CAPL source file is not deleted.

4.8 Break

If certain branches of the data flow in the measurement setup should not be run through, then a hotspot can be converted to a breakpoint. This is advisable, e.g. in online mode, if all functions (above all in the Trace window) cannot be serviced any longer without loss of data due to a high data rate.

When a break is created the configuration after the break is fully preserved, so that the old state can be reinstated after the break is deleted. Therefore, the break provides a very quick means for temporarily disconnecting certain data paths and thereby saving computer time.

At the start of a measurement the currently valid data flow plan is converted to an internal tree structure. In this conversion process breaks which are encountered in a path are propagated forward to the next branching point. Therefore, it is irrelevant to processor loading whether a break is placed at the front or back of a path. The same path is always masked out.

5 CAPL Programming

This chapter offers you a basic introduction in working with the programming language CAPL. Examples and a reference of all commands you can find in the online help.

5.1 Overview

The universal applicability of CANalyzer resp. DENalyzer results in large measure from its user programmability.

The Communication Access Programming Language CAPL is a C-like programming language, which allows you to program CANalyzer/DENalyzer for individual applications. In the development of network nodes, for example, the problem arises that the remaining bus nodes are not yet available for tests. To emulate the system environment, the data traffic of all remaining stations can be simulated with the help of CAPL.

All CANalyzer program variations/ options support the CAPL functions. The following program variations offer additional CAPL functions:

- CANalyzer.LIN resp. DENalyzer.LIN
- CANalyzer.MOST resp. DENalyzer.MOST
- CANalyzer.J1939

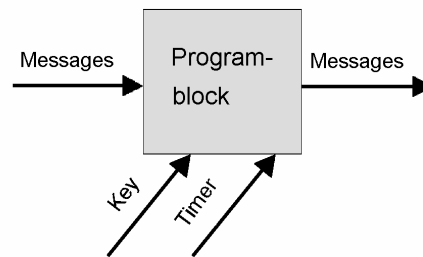
You can also write programs for problem-specific analysis of data traffic with CAPL, or you can program a gateway – a connecting element between two buses – to exchange data between different CAN buses.

CAPL nodes are inserted in the data flow plan as function blocks. Event procedures serve as inputs in CAPL. These procedures can react to external events (e.g. the occurrence of specific messages). You send messages by calling the function `output()`. These language tools and symbolic access to the various variables in the database make it possible to create simple prototypical models of nodes. The event procedures can be edited in the user-friendly Browser.

Note: You can find a detailed description of all CAPL functions in the online help.

5.1.1 Potential Applications of CAPL Programs

CAPL programs have an input through which messages pass as events into the block. Appearing at the output are all messages that either pass through the program or are generated by it. Furthermore, the program block can react to keyboard inputs (Key), time events (Timer) and – with CANoe – to changes in environment variables such as switches or slider positions.



Therefore, you can utilize a CAPL program to develop monitoring and testing for your special problem task. The CAPL program reacts to messages that CANalyzer registers on the CAN bus, and afterwards you can call your own analysis and test functions.

You can also use a CAPL program to emulate the system environment for a controller. The CAPL program reacts to both messages on the CAN bus and to your keyboard inputs, responding with certain CAN messages according to the event registered. It is entirely up to you to determine which actions are performed in response to which events.

Another possible application of CAPL is to program a gateway - that is a connecting element between two buses - to exchange data between different CAN buses and moreover to correct erroneous data occurring in this exchange.

Last but not least, the logging block can also be triggered by a CAPL program. Conditions of any desired complexity can be formulated for triggering. Triggering is initiated by a call of the intrinsic function `trigger()`.

5.1.2 Integration of CAPL Programs

A CAPL program can be inserted in the measurement setup at all hotspots. To do this, select the menu command **Insert CAPL node** from the hotspot's popup menu, and enter the name of the CAPL program file you wish to assign to this node in the configuration dialog. If you want to create a new CAPL program you can enter the name of a file that does not exist here yet. This file is then automatically created when editing.

You open the CAPL Browser by pressing the **[Edit...]** button in the configuration dialog or by double clicking the CAPL node. You can create and modify CAPL programs with this Browser.

Note: If you prefer to use your own editor to edit CAPL programs, enter it in the `[Environment]` section of the `CAN.INI` file.

Before you start the measurement you must compile all CAPL programs of the configuration. You can start the CAPL compiler from the CAPL Browser or from the configuration dialog. To compile all nodes at once, simply choose the main menu item **Configuration | Compile all nodes**.

Please note that a CAPL program may react completely differently, depending on the point at which you place it in the measurement setup. For example, a CAPL program located to the right of CANalyzer's transmit branch can indeed generate messages,

but it cannot send them on the bus. Since the data flow is directed from left to right, these messages are only passed to the function blocks to the right of the CAPL program. Only messages generated by CAPL programs located in CANalyzer's transmit branch³ can be sent out on the bus. This completely logical behavior - which may at first seem surprising - applies equally to the generator block, which - when it is located on the right side of the measurement setup - similarly generates messages without affecting the bus.

Therefore, in general those CAPL program blocks that exclusively serve analysis purposes should be inserted on the right side of the measurement setup, while program blocks for transmitting CAN messages should be inserted in CANalyzer's transmit branch.

5.1.3 Use of the Symbolic Database in CAPL

Just like other function blocks in the measurement setup, from CAPL you also have access to the symbolic information in the database. For example, instead of using the identifier 100 in your CAPL program you could use the symbolic name *EngineData* at all locations, provided that you have assigned this name to the identifier 100 in your database.

Use of the symbolic database makes your programs essentially independent of information that only relates to the CAN protocol, but has no meaning for the applications. Let us assume, for example, that during the development phase you determine that certain CAN identifiers in your system should be reassigned to change message priorities, and that in your system the message *EngineData* should now get the higher priority identifier 10 instead of the identifier 100.

In this case, let us assume that you have already developed test configurations and CAPL programs for your system which are exclusively based on the symbolic information (which do not use the identifier 100 anywhere, but rather always refer to the name *EngineData*). After modifying the identifiers in the database, you can incorporate the new information in the configuration by recompiling the CAPL programs. It is not necessary to adapt the CAPL programs to the new identifiers, since you only used symbolic names (e.g. *EngineData*), and not CAN identifiers (previously ID 100, now ID 10).

Therefore, it is advisable to manage all information relating only to the CAN bus in the database, and to use application-relevant symbolic information in CANalyzer exclusively.

5.1.4 Introduction to CAPL

CAPL is a procedural language whereby the execution of program blocks is controlled by events. These program blocks are known as event procedures.

The program code that you define in event procedures is executed when the event occurs. For example, you can send a message on the bus in response to a key press (**on key**), track the occurrence of messages on the bus (**on message**), or execute certain actions cyclically (**on timer**).

³ CAPL programs which are inserted at hotspots to the left of CANalyzer's transmit branch can also send messages on the bus. The part of the data flow between the card icon and the branch to the transmit block must, in strict terms, therefore also be included as part of the transmit branch.

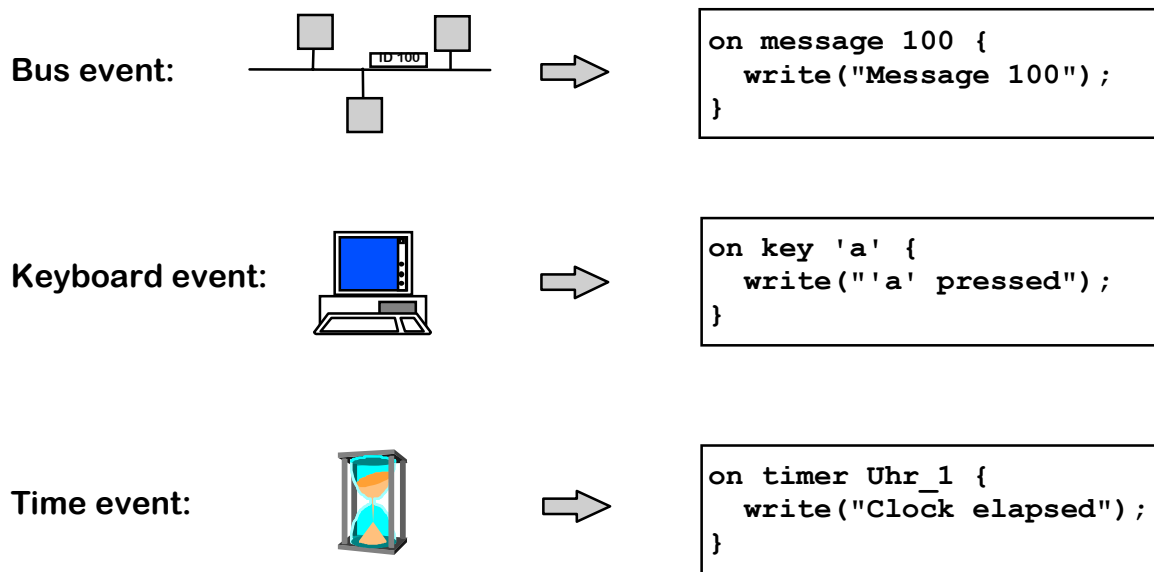


Figure 70: Examples of CAPL Event Procedures⁴

A CAPL program consists of two parts:

1. Declare and define global variables
2. Declare and define user-defined functions and event procedures

Variables in CAPL and the event procedure concept are described in further detail in the sections below.

5.2 CAPL Browser

A special Browser is integrated in CANalyzer as a user-friendly means to create, modify, and compile CAPL programs. Browser in this context is meant to signify a program for fast and targeted editing of CAPL programs.

⁴ Besides keyboard events, in CANoe you can - with event procedures of the type **on envvar** also react to actions that you perform yourself on user-defined control panels.

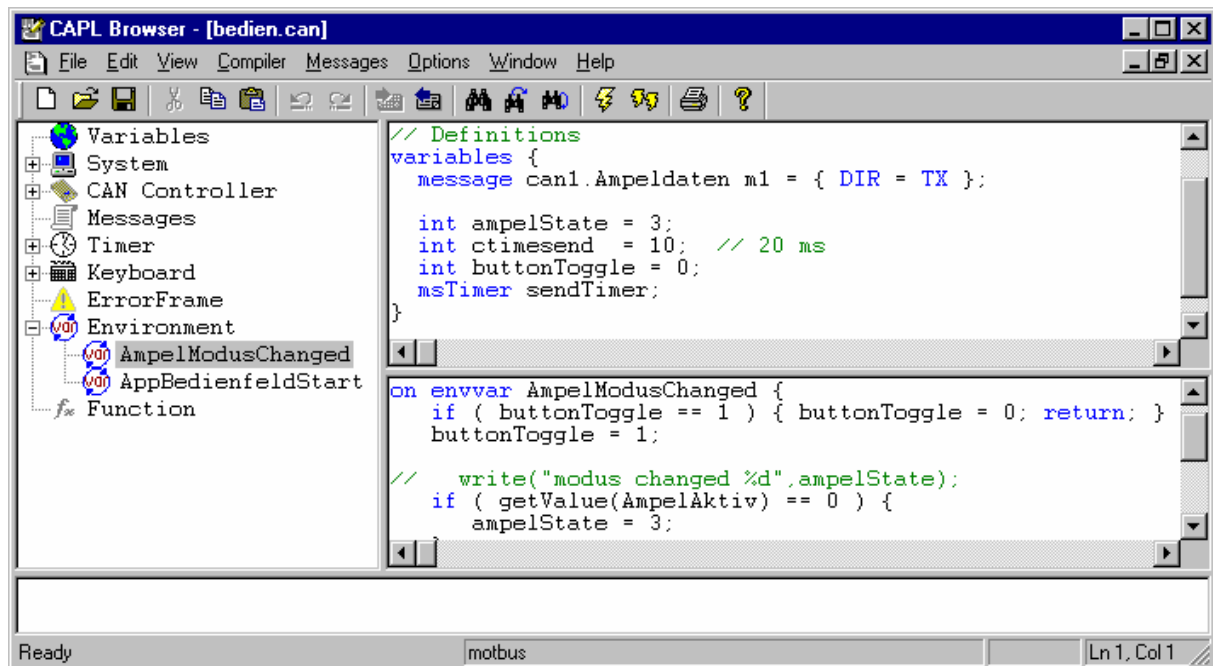


Figure 71: The Browser screen

A special Browser is integrated in CANalyzer for the user-friendly creation and modification of CAPL programs. This Browser shows you the variables, event procedures and functions of a CAPL program in structured form.

Multiple Browser windows with different CAPL programs can be opened simultaneously.

The CAPL compiler is started from Browser's main menu or toolbar. Compilation time is very short, even for larger programs. When an error is detected, the faulty program section is shown, and the cursor is positioned at the location of the error. This makes it very easy to make corrections.

5.2.1 Opening Browser

Browser is located in the system directory (canbr32.exe) and can be started as a stand-alone application like any other Windows program. However, it is recommended that you always start Browser from CANalyzer, since a number of important parameters for the program start (database name, compiler options, hardware parameters, CAPL-DLLs, etc.) must be passed.

When you start Browser by double clicking a CAPL node in the measurement setup, the file PARBROW.INI with all start parameters is automatically generated and made available to Browser. Therefore, always start Browser from CANalyzer, and furthermore make sure that the file PARBROW.INI is never given write protection, or else it might be impossible to pass important start parameters properly the next time Browser is started.

Note: A CAPL program file can be dragged from the File Manager on a CAPL Browser that has already been started ("Drop and drag"). The file is then displayed in a new window.

5.2.2 Browser Window

A Browser window is subdivided into up to four sub-windows, so-called "Panels".


At the upper left is the Browser Tree, which contains the CAN event types as drop-down nodes. In turn, each of these nodes contains the procedures that can be assigned to the CAN event types. The upper part of the text editor to the right of the Browser tree shows the global variables for the CAPL program; the lower part shows the procedure text for the procedure selected in the Procedures List. The text editor can also be configured so that global variables and procedures can be edited in a common editing window. Along the lower border is the Messages window that is used to display compiler messages for the CAPL program.

You can access the most important functions for each of the panes from the popup menu by pressing the right mouse button. In the editor panes you have access, via the popup menu, to the intrinsic CAPL functions and to the objects defined in the database. Furthermore, you can copy text to the Clipboard from the popup menu, and from there you can paste it in your program.

CAPL programs which are not available in Browser-specific file format are displayed in unstructured format in a normal text window and can be edited there. As in the editors of the Browser window, the program text can be edited via the menu command **Edit** or from the popup menu. In section 5.2.6 you will learn how to import such CAPL programs in Browser format.

5.2.3 Compiling CAPL Programs

To generate an executable program file in CBF format (CAPL Binary Format, file extension *.CBF) from a CAPL program, you must compile the program with the CAPL compiler.

Compilation of the program is started by the command **Compile** from the compiler menu, or by pressing the **Compile** icon () on the toolbar or by pressing the F9 key. If the CAPL program still does not have a file name, you are prompted to enter a name for the CBF file. If the program already has a name, the name of the CBF file is composed of the program name and the extension CBF.

All messages during the compilation process are output in the Message window.

If errors or warnings occur during the compilation process, the Message window automatically jumps to the foreground with the relevant error message. Double click the message or select the line and execute the command **Go to** from the **Messages** window to position the cursor at the location where the error occurred. After you have corrected it and saved the program file again, you recompile the program. If the program compiles without errors, the status **Compiled** appears in the status bar at the bottom of Browser's main window.

5.2.4 Searching for Run-Time Errors

Usually more difficult than the search for syntax errors that can already be found at compile time is the search for run-time errors. Nevertheless, CAPL offers you support here in two ways.

On the one hand, a number of possible run-time errors are automatically monitored in CAPL programs. These include:

- Division by zero
- Exceeding upper or lower array limits
- Exceeding upper or lower offsets in the data fields of messages
- Stack overflow in the call of CAPL subroutines

If such a run-time error is discovered, the measurement is terminated in a controlled manner. You get a message in the Write window containing the name of the CAPL program, the error type and an error index. With the help of this error index you can easily find the location in the CAPL source text that caused the error: Enter the index under the menu item **Compiler | Find run-time errors**.

On the other hand, as a user you have the option of generating run-time errors yourself with the CAPL function `runError()`, thereby making specific critical program sections fault-tolerant. In doing so, you can also give the function an error index which is then output to the Write window upon completion of the measurement.

5.2.5 Access to the Database

Provided that you open Browser directly from the measurement setup, Browser gets information regarding the databases associated to the configuration via the temporary file `PARBROW.INI`. But you can also associate one or more databases to the CAPL program directly in Browser from the **File** menu.

In Browser you have access to the database's messages and signals via the Message Explorer and Signal Explorer. You can open the selection dialog from the popup menu item **CANdb Message** or **CANdb Signal** in the editor panes on the right side. The selected object is assumed directly into the CAPL program text after exiting the dialog with **[OK]**. Of course you may also enter the signal and message names directly as clear text in the editor panes instead of using the symbolic selection dialog.

If you are using more than one database, the objects in the databases following the first database are qualified by including the database name in the symbolic selection dialogs. But you only need these qualified names to resolve ambiguities. As long as the symbolic names are unique in all databases, you can refrain from qualifying symbolic names when editing CAPL programs.

You can read about special considerations when using multiple databases in section 2.3.4.

5.2.6 Importing and Exporting ASCII Files

Browser offers an import function and an export function in the **File** menu to permit editing of your CAPL programs with text editors other than Browser.

The structural information that Browser uses to display CAPL programs to you in Browser format is contained in special comment lines. Browser automatically generates this information during compilation.

With **File | Import** a CAPL program that only exists in ASCII format can be loaded in the CAPL Browser. The procedures defined in the program are assigned to the CAPL procedure groups (e.g. Message, Timer, CAPL function, etc.). The command opens a file selection dialog in which you select the file to be imported. When importing programs make sure that the programs can be compiled error-free, since otherwise it

might not be possible to recognize function names or procedure names unambiguously.

With **File | Export** you can save CAPL programs in ASCII format. The generated file does not contain any structural information of the CAPL Browser and can only be displayed in a Browser window by importing it later.

5.2.7 Browser Options

In the menu **[Options | Editor...]** you will find commands for individual configuration of the editors in the Browser panes.

The menu item **Options | Compiler...** opens a dialog in which compiler options for the active CAPL program can be changed. These options are automatically configured for you, provided that you have opened Browser from the measurement setup. That is, normally you should not change the option settings here.

6 .CAN option

In this chapter you get a short introduction in CAN specific features of CANalyzer.

For the CANalyzer Tour and various examples the bus system CAN is underlied.

In this chapter, you get some additional information about the specific features of .CAN option.

Note: Please refer to the online help for further details.

6.1 The Trace Window of the .CAN option

Configuring the Trace window you can (also) choose CAN specific fields to be displayed.

For .CAN the following columns are available in the trace window:

Field	Title	Function
Time	Time	Absolute time from the measurement start, or relative time from the previous event
Channel	Chn	LIN channel. Number of the channel on which the message was sent/received
Identifier	ID	Identifier of the message
Name	Name	Symbolic name of the message
ID/Name	ID/Name	Depending on the global setting, either the symbolic name or the identifier of the message
Dir	Dir	TX = Transmit message, RX = Receive message, TXRQ = Transmit request
DLC	DLC	Length of the data field
Data	Data	The data in decimal or hexadecimal representation. For Remote Frames Remote Frame is shown here.
Attribute	Attr	Supplemental attributes. WU = WakeUp, NERR = Transceiver Error.
HH:MM:SS	HH:MM:SS	Absolute time from the measurement start in hh:mm:ss.000-format (24h)
Diff time	Diff time	Relative time to the previous event
Bustype	Bustype	Specification of the bustype (CAN, LIN etc). J1939-messages are specified as bustype "CAN", because J1939 is a protocol, not a bus
Send node	Send node	Specification of the send node, which is defined in the database for the corresponding message. For J1939 this column is without any content!

Bus	Bus	bus name
Database	Database	Database in which the corresponding message is defined

6.2 The Bus Statistics Window of the .CAN option

The Bus Statistics Window (also) displays CAN specific data and values.

For .CAN the following information are available in the bus statistics window:

- Busload [%]
- Peakload [%]
- Std. Data [fr/s]
- Std. Data [total]
- Ext. Data [fr/s]
- Ext. Data [total]
- Std. Remote [fr/s]
- Std. Remote [total]
- Ext. Remote [fr/s]
- Ext. Remote [total]
- Errorframe [fr/s]
- Errorframes [total]
- Chip State

7 .LIN option

In this chapter you get a short introduction in CANalyzer.LIN/DENalyzer.LIN.

The most important and most used functions, features and applications are introduced here.

Note: Please refer to the online help for further details.

7.1 Configuration of a LIN Test Environment

It is possible to configure a LIN test environment in various ways:

- Configuration by a LIN database
It is possible to configure LIN Slaves entirely from a LIN database, including reasonable initial values for their transmit messages.
- Configuration by simpler CAPL functions
If the transmitted data are to be changed during the measurement, there are simple CAPL functions for doing this. They also permit configuration of the LIN Master.
- Configuration by more complex CAPL functions
Full LINDa functionality can be achieved by the more complex functions of CAPL API.

7.2 LIN Scheduler

At the beginning of the measurement, the scheduler is automatically configured with data from the CANdb database. To do this, a master node must be defined in the simulation structure. At measurement start the scheduler begins to send automatically.

In addition to transmission requests of scheduling tables, transmission requests can also be sent directly with output.

Using the LDFtoDBC converter, included with delivery, "LIN Description Files" (LDF) can be converted directly into a CANdb database file.

To import a scheduler, follow the steps below:

- Generate a CANdb database with the LDFtoDBC converter.
- Assign the database
- Define a master node in the CANoe simulation structure or select the master node as a LIN node in the CANalyzer.

The master node must be defined in the database that is generated.

7.3 LIN Specifications

CANalyzer.LIN resp. DENalyzer.LIN supports the LIN specifications since v1.1.

With the new database attribute "LinProtocolVersion" you can switch between the LIN specifications.

7.4 The Converter Tool LDF to DBC

With the included LDF to DBC Converter the LIN Description Files (*.LDF) can be directly converted into CANdb database files (*.DBC).

Important features of the converter are:

- The converter accepts files in conformity with LDF specifications since v1.1.
- The converter performs a strict test of syntax, but does allow for a few additions/deviations, which nevertheless will generate warnings.
- The converter performs a partial semantic test as well.
Thus, references to undefined identifiers (nodes, signals, etc.) will generate an error, for example.
- Scheduler tables are stored in attributes.

7.5 Trace Window of the .LIN option

Configuring the Trace window you can (also) choose LIN specific fields to be displayed.

For .LIN the following columns are available in the trace window:

Field	Title	Function
Time	Time	Absolute time from the measurement start, or relative time from the previous event
Channel	Chn	LIN channel. Number of the channel on which the message was sent/received
Identifier	ID	Identifier of the message
Name	Name	Symbolic name of the message
ID/Name	ID/Name	Depending on the global setting, either the symbolic name or the identifier of the message
Dir	Dir	TX = Transmit message, RX = Receive message
DLC	DLC	Length of the data field
Data	Data	The data in decimal or hexadecimal representation
Protocol time	Full time	Duration of the entire bus event (for example message, transmission error, reception error) in bit times
HH:MM:SS	HH:MM:SS	Absolute time from the measurement start in hh:mm:ss.000-format (24h)
Diff time	Diff time	Relative time to the previous event
Bustype	Bustype	Specification of the bustype (CAN, LIN etc).
Send node	Send node	Specification of the send node, which is defined in the database for the corresponding message. For J1939 this column is without any content!
Typ	Typ	Type of event, e.g. transmission error

Specific 0	Checksum	Checksum over the data field
Specific 1	FSM ID	Number of the relevant finite state machine in the LIN-da hardware
Specific 2	State	State of the finite state machine
Specific 3	Followup state	Follow up state of the finite state machine
Specific 4	Offending byte	Any corrupt byte (depending on the event type)
Specific 5	Header time	Duration of the message header in bit times
Bus	Bus	bus name
Database	Database	Database in which the corresponding message is defined

7.6 The Bus Statistics Window of the .LIN option

The Bus Statistics Window (also) displays LIN specific data and values.

For Option .LIN the following information are available in the bus statistics window:

- Busload [%]
- Peakload [%]
- Data [fr/s]
- Data [total]
- TransmErr [err/s]
- TransmErr [total]
- CSErr [err/s]
- CSErr [total]
- RcvErr [err/s]
- RcvErr [total]
- SyncErr [err/s]
- SyncErr [total]
- Wakeups [fr/s]
- Wakeups [total]
- Chip State

8 .MOST option

In this chapter you get a short introduction in CANalyzer.MOST/DENalyzer.MOST.

The most important and most used functions, features and applications are introduced here.

8.1 Installation Procedure

You can configure the MOST hardware in the **Configuration | Hardware Configuration** dialog.

There is a main page for each channel and a sub-page called **Setup**. The main page for the .MOST option is labeled **MOST X** (X is the number of the channel).

8.1.1 Optolyzer

The .MOST Option establishes a connection to the Optolyzer Box from Oasis by means of an ActiveX (single station) control named the Optolyzer ActiveX control. No license is needed for the Optolyzer Professional software.

Requirements

- Optolyzer Box (Firmware Version 2.50 or higher)
- Optolyzer ActiveX Control
- License number for operating Optolyzer with the Optolyzer ActiveX Control
- Available COM port

Procedure

- Install the Optolyzer ActiveX Control on your PC.
You will find the installation program suitable for your operating system in the `\Drivers\Optolyzer\OptoControl` folder on the CD. Since CANoe was developed and tested with the OptoControl versions on the CD we highly recommend that you install these driver versions.
- From the Windows control panel set the COM port options for all COM ports used with an Optolyzer box as follows:

Baudrate:	115200
Data bits:	8
Parity:	none
Stop bits:	1
Flow control:	Hardware

- Enter the license code for the Optolyzer ActiveX Control in the Hardware Configuration Dialog.
- For users of CANoe/CANalyzer: Activate **Synchronize CAN Hardware** in the **Global settings** section of the CAN Driver Configuration dialog.

If Optolyzer Professional is installed the license number for the ActiveX Control can be input using this tool. It is then saved in the Windows Registry. This is how you can save multiple license numbers. They are all checked when the connection is made to the Optolyzer box.

8.1.2 Tool4M-XL Installation Section

The .MOST option supports the Tool4M-XL from Optitas as the hardware interface.

Requirements

- Tool4M-XL
- Available COM port
- Ethernet connection
- In the network: A fixed IP address must be permitted for the Tool4M-XL.

Procedure

- Connect the Tool4M-XL to the PC over the COM port.
This connection is only needed once for configuration purposes.
- Connect the Tool4M-XL to the PC over the Ethernet.
- Configure the Tool4M-XL in the Hardware Configuration Dialog.

8.2 Configuration settings

There are some configuration parameters for the Optolyzer in the configuration file CAN.INI in the executable directory of CANalyzer/DENalyzer.

For each channel enter a section [OptolyzerX] in CAN.INI, where X stands for the channel which should be configured by this section.

The selection of the COM port and the license key can comfortably be entered through the MOST Hardware Configuration dialog. However, the values will still be stored in CAN.INI because these are PC related settings which should not be moved to other PCs by using the configuration file on other PCs.

```
[Optolyzer1]
// Interface
// 1=COM1, 2=COM2
ComPort = 1

// Licence number for operation without OptolyzerPro
// (If Optolyzer Pro is installed with a licence for using
// the OptoControl with the connected Optolyzer Box no en-
// try is needed)
Licence=89AB0123CDEF4567
```

8.3 Timestamps

CANalyzer uses a resolution of 10 μ s for the timestamps of events. The Optolyzer box generates only timestamps with a resolution of 1 ms, thus all shown timestamps related to MOST events have at maximum this resolution, even if shown in a higher resolution.

All events received before the time synchronization could calculate a valid offset between CAN bus and MOST ring are suppressed. Therefore the first MOST frames shown in a trace window may have a timestamp several seconds after measurement start.

8.3.1 Synchronized timestamp

The unit of this timestamp is 10 μ s, with 0 at the start of measurement.

This timestamp is synchronized with the timestamps on the CAN bus by a software algorithm.

Due to the algorithm, the accuracy of this timestamp might vary slightly, but the timestamps can be used for calculations across more than one bus system or across multiple MOST channels.

Technical reasons let this algorithm work better with the spy mode of the Optolyzer box.

In node mode time synchronization may be impaired by sending MOST frames from CAPL nodes.

8.3.2 Original timestamp

This is the original timestamp generated by the MOST hardware. Only the unit is changed to 10 μ s and an offset is subtracted, so that the measurement start again has approximately a timestamp of 0.

For higher precision use this timestamp for calculations concerning periods of time only in the MOST system on one channel. The offset between original timestamps from different MOST channels may vary with each measurement start. But provided that the MOST channels are derived from a single MOST ring, the offset stays constant during the measurement, due to the synchronization of the MOST devices to the timing master in the ring.

If different MOST channels of CANalyzer are connected to different MOST rings (with different timing masters) each MOST channel has its own time base. The original timestamps will diverge in this case. Therefore the MOST channels are also synchronized which will affect only the synchronized timestamp.

8.4 Time Synchronization Accuracy

The accuracy of the synchronization between MOST channels is typically +/- 1 ms, between a MOST and a CAN channel is around +/- 2 ms.

These statements are based on measurements with a desktop PC with a Pentium III 450 MHz, Windows NT 4.0 (SP6) and a CANcardX with driver version 3.0. The Optolyzer boxes were operated in spy mode with a continuous bus load of up to 166 frames/s. (With more than 166 frames/s the buffer of the Optolyzer box starts to

fill up). The CAN bus was operated with two active channels at 1 MBit/s and a bus-load of 30 % on each channel.

Due to adaptive algorithms the full accuracy of the synchronization is achieved only after a few seconds after measurement start. If synchronization is a critical point within the measurement application, start the measurement a couple of seconds before the condition which shall be analyzed occurs.

If CANalyzer is used to trigger the interesting condition on the bus, use interactive triggering e.g. by operating a panel or an interactive generator block or use a CAPL timer started at measurement start to allow a delayed trigger condition.

8.5 Database Support

CANalyzer.MOST offers two ways of describing the data structure on the MOST frames. Known from version 0.95 is the description of messages in a CANdb++ database. This is still the only way to have database support for MOST in CAPL. In CAPL, messages can be identified and parameters can be accessed by names defined in the CANdb++ database. Additionally this way is used to show parameter values in the Data and Graphics Windows and to statically translate physical addresses into device names.

Unfortunately the CANdb++ database does not allow the description of all data formats used in MOST systems.

Therefore CANalyzer.MOST allows the import of an XML description of the function catalog. This feature is used to

- display a disassembled view of the messages in the Trace Window, to
- configure Filter Blocks or to
- assemble frames with the Interactive Generator for MOST.

8.6 Interactive Generator Block MOST

The interactive generator block MOST (IG MOST) can be used to configure and send messages while the measurement is running.

Advantages:

- quick, improvisational manipulation of the measurement
- no CAPL programming necessary

MOST messages will be included in the send list of the configuration dialog of the IG MOST. Within this dialog the messages can be configured, filled with data, and assigned to a MOST channel. By one button click a message can be send.

Within the IG MOST all informations about messages and their parameters are taken from the assigned XML function catalogs. Therefore the MOST messages can be selected from a tree representation of the function catalogs.

To send MOST messages to the MOST bus insert an IG MOST block

- in the measurement setup/send branch using the context menu of the hot spots (CANalyzer)

- in the simulation setup using the context menu of the simulated busses (CANoe)

To manipulate the analysis without sending MOST messages to the bus insert an IG MOST block in the measurement setup before the appropriate analyze block (CANalyzer and CANoe).

Please see the online help to learn more about the IG MOST.

8.7 Trace Window

For MOST frames the trace windows offers the following display options:

Nam	Description
Timestamp	Synchronized timestamp
Channel	Channel information with the prefix 'M'
Identifier	Lookup key as described in chapter 8.5
Name	Name of the frame provided by the database
ID/Name	Depending on the general display option: Symbolic mode: Name Numeric mode: Display of lookup key (see chapter 8.5)
Dir	Received frame (Rx), transmitted frame (Tx) or requested transmission (TxRequest)
Data	Transported data of the event
RType	Subtyp of the MOST Control Frame: RemoteRead, RemoteWrite, Alloc, Dealloc, GetSource
Source	<ul style="list-style-type: none"> • Symbolic Mode: Device name, if available • Numeric Mode: Source Address
Destination	<ul style="list-style-type: none"> • Symbolic Mode: Device name, if available • Numeric Mode: Destination Address
Original Time	Original timestamp generated by the hardware interface, with an offset of 0 at measurement start (see also chapter 8.3)
Type	Type of MOST event (CtrlFrame, AMSMsg, AMSMsgErr, Packet, LightLock, Register, RegData, HWMode, Trigger, NetState)
Disassembly	Symbolic interpretation of the event. For Control Frames and AMS messages the data bytes are interpreted from FBlock to OpType based on the available databases (CANdb or XML). The remaining bytes are displayed numerically.
State	State information of the received or transmitted message.
AckNack	Acknowledgement information of the received or transmitted message.
CRC	Cyclic block check code of the Control Frame or packet.
ASCII	ASCII representation of the data bytes.
DbType	Type of database the MOST frame is disassembled with. This is DBC for CANdb++ database or XML if the disassembly is based on the XML description of the MOST Function Catalog.
FBlock.InstID	Symbolic or numeric display of the functional address of messages satisfying requirements ranging from function catalog format to OpType (Control and AMS messages).

Function	Symbolic or numeric display of the function name for messages satisfying requirements ranging from function catalog format to OpType (Control and AMS messages).
OpType	Symbolic or numeric display of the OpType for messages satisfying requirements ranging from function catalog format to OpType (Control and AMS messages).

The Trace Window shows a tree view in 'overwriting mode' or when paused. Lines with MOST frames corresponding to database messages to which signals are mapped, show a small '+' at the beginning. This indicates that this part of the tree view can be extended and the sub-tree shows the names and values of the parameters mapped to the MOST frame.

Parameters are only interpreted, when the high nibble of byte 4, containing the Telld, equals to 0 or 1. Only then the concerned frame contains the bytes corresponding to the signal description in the database. In other cases the MOST frame belongs to a segmented message, for which a disassembly is not implemented yet.

8.8 Bus Statistic Window

The Bus Statistic Window can display MOST-specific data and values.

The following subjects are available in the bus statistic window of the .MOST option:

- **Frames [fr/s]**
Frames received per second over the control channel.
- **Frames [total]**
Total number of frames received over the control channel.
- **Lgt&Lck [L&L/s]**
Light-and-Lock errors per second.
L&L errors are reported by the Optolyzer, whenever it recognizes that there is no light at the input or the synchronous timing device fails. The error is only recognized and reported once, e.g. when the light is turned off. As a result, the Optolyzer cannot receive any frames but it doesn't send L&L errors continuously.
- **Lgt&Lck [total]**
Total number of Light-and-Lock errors.
L&L errors are reported by the Optolyzer, whenever it recognizes that there is no light at the input or the synchronous timing device fails. The error is only recognized and reported once, e.g. when the light is turned off. As a result, the Optolyzer cannot receive any frames but it doesn't send L&L errors continuously.
- **Buffer Lvl. [%]**
Working load of the input buffer of the Optolyzer.
- **Packets [pkt/s]**
Packets received per second over the asynchronous channel.
- **Packets [total]**
Total number of packets received.
- **State**
Light & Lock Status at the Rx pin of the MOST hardware.

- **AMSMessages [msg/s]**
AMS messages received per second.
- **AMSMessages [total]**
Total number of AMS messages received.
- **AMSTelegrams [tel/msg]**
Mean number of frames (Control frames) per received AMS message.
- **AMSDuration [ms/msg]**
Mean transmission time for received AMS messages.
- **AMSRxError [total]**
Total number of receiving transmission errors in the Application Message Service.
- **AMSTxError [total]**
Total number of sending transmission errors in the Application Message Service.

8.9 Graphic- & Data Window

The database support by CANdb++ allows the display of parameters defined in the database in the graphic- respectively in the data window. By calling the configuration dialog of either of these windows and selecting 'New Signal' a signal selection dialog will popup. This dialog shows all associated databases with the defined messages and the signals/parameters mapped onto the messages.

After the selection of a message by name (see chapter 8.5), a signal can be added to the window. The value of the signal will be shown during measurement either as graph or as numerical value. The value will be updated whenever a MOST frame with the same lookup key is received.

No XML support is implemented for these Windows so far.

8.10 CAPL

For easy use, the MOST commands are integrated in CAPL.

8.11 Demo Configurations CANalyzer/DENalyzer

Demo configurations demonstrating MOST features can be found in the `De-mo_MOST_CL` folder of your CANalyzer/DENalyzer installation. Detailed descriptions of the demos are provided within the configuration comments (**File | Configuration comment...** menu).

8.12 XML Engine

CANoe/DENoe/CANalyzer/DENalyzer uses Xerces-C++ for parsing XML files. Please notice the following license of Xerces:

"This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)."

Copyright (c) 2000 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.
- The names "Xerces" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.
- Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation, and was originally based on software copyright (c) 1999, International Business Machines, Inc., <http://www.ibm.com>. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

9 .FlexRay option

In this chapter you get a short introduction in CANoe.FlexRay bzw. DENoe.Flexray.

The most important and most used functions, features and applications are introduced here.

Note: Please refer to the online help for further details.

The FlexRay option offers extensive capabilities for analyzing distributed real-time control systems with FlexRay. The service module is available for the connection to FlexRay networks.

With the FlexRay option, you are able to

- receive,
- display,
- analyze and
- send Flexray frames.

9.1 Trace Window for the .FlexRay option

Configuring the Trace window you can (also) choose FlexRay specific fields to be displayed.

For .FlexRay the following columns are available in the trace window:

Field	Title	Function
Time	Time	This time stamp is transmitted directly by the Service Module. Currently there is still no time synchronization with other bus systems. In operation with a CAN system the time stamps of the individual bus systems run on different time references.
Channel	Chn	Defines the transmitting source. "Fr" means that the frame was received in a FlexRay network. The number indicates the channel number.
Identifier	ID	Identifier of received frame
Name	Name	Symbolic name of the message
DLC	DLC	Length of the data field
Data	Data	The data in decimal or hexadecimal representation
Typ	Typ	Type of event, e.g. transmission error
Error	Error	Symbolic representation of the FlexRay status
Specific 0	Mux	Multiplex bit
Specific 1	Frame Sate	State flag of Flexray message

Specific 2	Schedule	Assignment of the frame to the static or dynamic part of the communication cycle
Specific 3	Low Time	Addition to the time display HH:MM:SS in simulated mode
HH:MM:SS	HH:MM:SS	Absolute time from the measurement start in hh:mm:ss.000-format (24h)
Diff time	Diff time	Relative time to the previous event
Bustype	Bustype	Specification of the bus type (CAN, LIN etc). J1939-messages are specified as bus type "CAN", because J1939 is a protocol, not a bus
Send node	Send node	Specification of the send node, which is defined in the database for the corresponding message. For J1939 this column is without any content!
Bus	Bus	bus name
Database	Database	Database in which the corresponding message is defined

9.2 The Bus Statistics Window for the .FlexRay option

The Bus Statistics Window (also) displays FlexRay specific data and values.

For .FlexRay the following information are available in the bus statistics window:

- Frames [fr/s]
- Frames [total]
- Data [fr/s]
- Data [total]
- Sync [fr/s]
- Sync [total]
- Nullframes [fr/s]
- Nullframes [total]
- Errors [[fr/s]
- Errors [total]

10 Index

!

! Load symbol · 43

<

<F10> key · 3

7

72005 · 66, 68

8

82527 · 66, 68

82C200 · 66, 68

A

Acceptance code · 68

Acceptance filtering · 46, 68, 69

Acceptance filters · 65

Acceptance mask · 68, 69

Acknowledge · 67

Activity indicator · 94

Ambiguities · 92, 122

Analysis blocks · 45

Analysis branch · 46

Animate · 56, 57, 79

Animation factor · 36

Array · 122

ASCII · 53

Averaging time · 45, 95, 99

Axis options · 82, 83

B

Bar-type display · 94

Basic image · 96

Basic-CAN controller · 64

Baud rate · 90

Baudrate · 65, 66

binary · 53

Binary · 53

Bit time · 67

Bit timing · 67

Bit timing · 67

Blocking filter · 46

Branches · 7, 71

Break · 55, 56, 57

Break condition · 56

Breakpoint · 56

Browser tree · 121

BTL cycles · 65, 67

Burst · 108

Bus load · 43, 64, 72, 88, 98

Bus load measurement · 99

Bus loading · 43, 46

Bus parameters · 65

Bus registers · 65

Bus statistics · 54, 69, 98, 99

Bus statistics display · 43

Bus statistics information · 54, 99

Bus Statistics Window · 125

Bus Timing Register · 66

Busstatistik Fenster · 128, 138

Button · 4

C

CAN card · 64

CAN channel · 41

CAN.INI · 32, 44, 54, 56, 117

CANalyzer without card · 36

CANcard · 64

CANcardX · 64

CANdb++ · 6, 24, 37

CAPL · 30, 51, 60, 61, 63, 119

CAPL Browser · 24

CAPL node · 114

CAPL program · 117

CAPL Program · 113

Card clock · 35

Card driver · 33, 34, 35

Channel · 64

Channel definition · 41, 42, 64

Check box · 4

Chip allocation · 41, 64
 CIF files · 32
 Clipboard · 90, 121
 Colors in the Graphics window · 87
 Comment · 100, 122
 Comment box · 4
 Compilation time · 120
 Compile time · 121
 Compiler · 33, 117, 120, 121, 122, 123
 Configuration · 3, 4, 36, 41, 71, 101, 111, 115
 Configuration descriptions · 32
 Configuration file · 28, 31
 Configuration file formats · 32
 Configuration management · 31
 Configuration options · 27, 72, 73, 88
 Context menus · 3
 Control · 60
 Control box · 4
 Conversion
 Log files · 58
 Converter
 LDF to DBC · 127
 Copying blocks · 73
 Cycle time · 45, 95, 108
 Cyclic update · 45, 52

D

Data flow · 7, 25, 27, 33, 34, 62, 71, 72, 100, 101, 110, 115, 118
 Data flow diagram · [Z](#), 27, 30, 34, 55, 71, 72
 Data loss · 43, 44
 Data reduction · 46, 56
 Data source · 46, 53, 55, 57, 63, 72, 99
 Data throughput · 99
 Database · 30, 37, 41
 Database Editor · 24
 DBC
 LDF to DBC Converter · 127
 Deactivation function · 45
 Decrementer · 108
 Delay times · 69
 Demo version · 25, 26
 Desktop Concept · 70
 Dialog box · 3
 Difference markers · 85

Difference measurement mode · 85
 Docking window · 71
 DPRAM · 60, 63
 Driver options · 54, 69
 Drop and drag · 120
 Drop down list · 4

E

End of measurement · 50, 81, 84
 Environment variable · 75
 ERROR · 98
 Error frame · 30, 51, 75, 98, 101, 107
 Error message · 41, 59, 60
 Error message · 52
 Error Passive · 59
 Error-Frame · 98
 Evaluation branch · 46
 Evaluation functions · 46
 Export
 Logging files · 58
 Extended identifier · 74
 Extended Identifier · 96

F

Fenster
 Busstatistik Fenster · 128, 138
 File format · 121
 File icon of the log file · 21
 Fill level indicator · 44
 Filter · 7, 27, 34, 38, 72
 Fit · 96
 Floating window · 71
 Formats of configurations · 32
 Function block · 73
 Function generator · 104

G

Gateway · 116, 117
 Generator block · 7, 30, 34, 101, 102, 103, 104, 105, 118
 Global options · 32
 Grid · 87

H

Hardware configuration · 41
 Help · 6, 69
 hexadecimal · 68
 High-load operation · 43
 Hotspots · 7

I

Import of configuration descriptions · 32
 Incrementer · 108
 Infinite loop · 35
 INI file · 32, 44, 54, 56, 59
 INI-Datei · 117
 Input box · 4
 Interactive generator block · 106
 Interrupt · 61

K

Keyboard control · 109
 Konfiguration
 LIN Testumgebung · 126

L

Layout · 71
 LDF
 LDF to DBC Converter · 127
 Line type · 82
 Load operation · 43, 44
 Load situation · 43
 Load transition · 44
 Log file · 110
 converting · 58
 exporting · 58
 file icon · 21
 Logging
 time period of · 51
 Logging export · 58
 Long-duration measurement · 52

M

Main memory · 51
 MDI window · 71

Measurement configuration · 3
 Measurement marker · 84, 85
 Measurement point · 83
 Measurement setup · 25, 33, 71, 100, 110, 111, 115
 Measurement start · 30, 54, 64, 83
 Measurement value acquisition · 83
 Menu operation · 3
 Message attribute · 35
 Message display · 43
 Message Explorer · 38, 122
 Message rate · 95
 Messages window · 121
 Mode signal · 104
 Mode-dependent signal · 104
 Moving blocks · 73
 Multiplex message · 103
 Multisignal mode · 80, 84

N

Network nodes · 71
 Nodes · 71
 Notebook · 59
 Numbering system · 32

O

Offline mode · 55, 56, 57, 58
 Online mode · 30, 33, 46, 57, 63, 72
 Operating instructions · 3
 Operating mode · 5, 36
 Optimization · 44, 88, 94
 Trace window · 79
 Options button · 4
 Output mode · 43, 45, 83
 output() · 114, 116
 output(this) · 114
 Overflow · 60
 Overload situation · 43, 44
 Overload symbol · 55

P

Panes · 121, 122, 123
 PARBROW.INI · 120

PC card · 25, 31, 63, 67, 68
PC-card · 57, 73
Peak display · 94
Performance · 94, 99
Performance optimization · 44
Performance wizard · 44
Phase model · 36, 71
Point measurement mode · 84
Post-trigger time · 30, 51
Power manager · 59
Prescaler · 65
Pre-trigger time · 51
Procedures list · 121
Program block · 34, 116
Program start · 7, 27
Project directory · 31
Project file · 31

R

Radio button · 4
Raw value · 108
Reaction times · 25, 46
Real channel · 41, 64
Real-time library · 25, 43, 44, 46
Receive messages · 35
Receive time point · 63
Refresh · 45
Remote frame · 107
Replay block · 110
Representation formats · 32, 33
Representation parameters · 91
Reset · 56
Right mouse button · 3
Ring buffer · 25, 43, 44, 51, 52
runError() · 122
Run-time error · 122
Rx attribute · 35, 95
Rx buffer overrun · 99
Rx error counter · 98

S

Samples · 66
Sampling point · 65

Sampling time point · 67
Scaling · 96
scalings · 86
Search condition · 47
Set of user defined conditions · 48
Shortcuts · 109
Signal · 37
Signal curve · 82
Signal Explorer · 17, 18, 38, 122
Signal identifier · 92
Signal response generator · 104
Signal selection dialog · 82
Signal value · 103
Signal-based logging export · 58
Simulation · 36, 71
Simulation setup · 3, 25, 36, 41, 71
Simulation setup window · 71
Single-signal mode · 80, 84
Single-step mode · 57
SJA 1000 · 64, 66, 67, 68
SJW · 66
Slow motion · 56
Stack overflow · 122
Standard deviation · 97
Start · 56
Start bit · 108
Start options · 24
Statistical report · 43
Statistics report · 45, 90, 95
Statistics window · 95
Step · 57
Stop · 51
stop() · 30, 56
Symbol selection dialog · 122
Symbolic databases · 36
Synchronization edge · 67
Synchronization jump width · 66, 67
System directory · 28
System loading · 95
System messages · 90

T

Testumgebung
Konfiguration · 126

Text editor · 121
 Time base · 57
 Time basis · 57
 Time resolution · 59
 Time stamp · 35, 77
 Time window · 50, 51
 Timeout · 59, 63
 Timer · 116
 Toolbar · 81, 120, 121
 Trace window · 74, 124
 Configuration of columns · 77
 Optimization · 79
 Transmission of messages · 106
 Transmission time · 36
 Transmit block · 33
 Transmit branch · 33
 Transmit list · 107
 Trapezoidal function · 104
 trigger · 117
 Trigger · 46, 47, 49, 50, 51, 54
 Trigger block · 55
 Trigger condition · 46, 47, 51, 101, 102, 103, 107, 108
 Trigger Condition · 48
 Trigger Mode · 47
 Trigger time point · 108
 Trigger type · 50
 trigger() · 117
 Triggerblock · 55

Triggering · 51, 117
 Tx attribute · 35, 95
 Tx error counter · 98
 TXREQUEST · 69
 TxRq attribute · 35

U

Undo function · 65

V

Value table · 103
 Virtual channel · 41, 64

W

Window
 Bus Statistics Window · 125
 Trace window · 124
 Window control · 30
 Window Management · 70
 Working directory · 31
 write() · 90

Z

Zoom · 96
 Zoom mode · 73