



.CAN.LIN.MOST.FlexRay

Manual

Version 4.1

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

Japan

Vector Japan Co., Ltd.
Nishikawa Bldg. 2F
3-3-9 Nihonbashi, Chuo-ku
J-103-0027 Tokyo
Tel.: +81 3 3516 7850
Fax: +81 3 3516 7855
<http://www.vector-japan.co.jp>

Sweden

VecScan AB
Fabriksgatan 7
S-41250 Göteborg
Tel.: +46 031 79901 35
Fax: +46 031 79903 05
<http://www.vecscan.com/>

USA

Vector CANtech, Inc.
Suite 550
39500 Orchard Hill Place
USA- Novi, Mi 48375
Tel.: +1 248 449 9290
Fax: +1 248 449 9704
<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 Management of Vector Informatik GmbH is being certified throughout since 1998-08-19:

- 2001-11-27
according to DIN EN ISO 9001:2000-12
Certificate number: 70 100 1498
- 1998-08-19
according to DIN EN ISO 9001:1994-08
Certificate number: 70 100 F 1498 TMS

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
<Strg>+<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 new naming convention

CANoe **CAN**alyzer **.LIN**
DENoe **DEN**alyzer **.MOST** **.FlexRay**

The multi-bus functionality and die 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 CANoe.....	3
1.2.1	Menus	4
1.2.2	Dialogs	4
1.2.3	Measurement and Simulation Setup	6
1.2.4	Online Help	7
1.3	CANoe Tour.....	7
1.3.1	Preparations	8
1.3.2	Setting Up the CAN Bus.....	9
1.3.3	Transmitting Data	11
1.3.4	Evaluation Windows	15
1.3.5	Working with Symbolic Data.....	17
1.3.6	Analysis of Signal Values in the Data Window	19
1.3.7	Analysis of Signal Responses in the Graphics Window	21
1.3.8	Use of the Database in Transmitting Messages	22
1.3.9	Logging a Measurement.....	22
1.3.10	Evaluating a Log File.....	24
1.3.11	Creating a CAPL Program.....	24
1.3.12	Simulation of Distributed Systems in CANoe	27
1.3.12.1	Creating the Database	27
1.3.12.2	Creating Panels	28
1.3.12.3	Creating Network Node Models	30
1.3.13	Tips for Solving Your Own Tasks	31
1.4	Overview of the Programs.....	32
1.5	CANoe Architecture.....	33
1.6	Particularities of the Demo Version	35
2	Applications	36
2.1	Simulation/ Simulation Setup	40
2.1.1	Working in the Simulation Setup	40
2.1.2	Gateway	40
2.1.3	System View.....	41
2.1.4	Object View	41
2.1.5	System Verification.....	41

2.2	Measurement/Measurement Setup	42
2.2.1	Measurement Start	42
2.2.2	Working with Configurations.....	43
2.2.3	Representation Formats	44
2.3	Working with Databases.....	45
2.3.1	Creating and Modifying Databases	45
2.3.2	Access to Database Information.....	46
2.3.3	Associating the Database.....	47
2.3.4	Use of Multiple Databases	48
2.3.5	Resolving Ambiguities	48
2.3.6	Checking for Consistency of Symbolic Data.....	49
2.4	Working with Multiple Channels	49
2.4.1	Channel Definition	49
2.4.2	Channels in Online Mode	50
2.4.3	Channels in Simulation Mode.....	50
2.4.4	Channels in Offline Mode	51
2.5	CANoe in Load and Overload Operation.....	51
2.5.1	Behavior in Load Situations.....	51
2.5.2	Behavior with Data Loss.....	52
2.5.3	Fill Level Indicator	52
2.5.4	Optimizing Performance.....	53
2.5.5	Configuration Options at High Bus Load	53
2.6	Working with Panels and Environment Variables	55
2.6.1	Assigning and Positioning Panels	55
2.6.2	Panel Configuration Dialog.....	55
2.6.3	Initialization of Environment Variables	58
2.6.4	Panel Control	58
2.6.5	Configuring the Panel Control for Small Models.....	59
2.7	Logging and Evaluation of Measurement Files	59
2.7.1	Logging Triggers.....	60
2.7.1.1	Trigger Modes.....	60
2.7.1.2	Trigger Events.....	63
2.7.1.3	Time window	63
2.7.1.4	Configuration of the Logging Buffer	64
2.7.2	Log Files.....	65
2.7.3	Event Types in Log Files	66
2.7.4	Data Analysis in Offline Mode.....	67
2.7.4.1	Flow Control in Offline Mode.....	68

2.7.4.2	Configuration of Online and Offline Modes	68
2.7.5	Trigger and Search Conditions.....	69
2.7.5.1	Condition Primitives	69
2.7.5.2	Entry or Change of Primitives (without Database)	70
2.7.5.3	Entry or Change of Primitives (with Database)	71
2.7.6	Exporting and Converting Log Files	72
2.7.6.1	Export	72
2.7.6.2	Conversion.....	72
2.7.7	CANlog support.....	72
2.8	COM-Server	73
2.9	Troubleshooting.....	73
2.10	List of Error Messages to the CAN Interface	74
2.11	The Interface to the Hardware.....	78
2.11.1	Configuring the Hardware	79
2.11.2	Programming the Bus Parameters	80
2.11.3	Acceptance Filtering.....	82
2.11.4	Card and Driver Options.....	84
3	Windows.....	85
3.1	Simulation Setup	85
3.1.1	Configuration of the Simulation Setup.....	85
3.1.2	Layout of the Simulation Setup	87
3.2	Measurement Setup Window	87
3.2.1	Data Flow in the Measurement Setup	87
3.2.2	Configuration of the Measurement Setup.....	88
3.2.3	Working with Evaluation Blocks in the Measurement Setup.....	89
3.2.4	Message Attributes.....	90
3.2.5	Simulation Mode.....	91
3.3	Trace Window	93
3.3.1	Standard Configuration of the Trace Window.....	95
3.3.2	Configuration of Columns in the Trace Window	98
3.3.3	Trace Window Options from the Toolbar	99
3.3.4	Detail View (Trace Watch Functionality).....	99
3.3.5	Optimizing the Trace Window	99
3.4	Graphic Window.....	100
3.4.1	Selecting Signals.....	101
3.4.2	Arrangement of Signals.....	102
3.4.3	Signal Layout	102

3.4.3.1	Line Types.....	103
3.4.3.2	Display Modes	103
3.4.4	Configuration of the Measurement	103
3.4.5	Measurement and Display Functions	104
3.4.6	Signal Modes	104
3.4.7	Measurement Modes.....	105
3.4.8	Display Modes.....	106
3.4.9	Layout Functions	106
3.4.10	Export of Signals	108
3.4.11	Toolbar of the Graphics Window	108
3.4.12	Optimization of the Graphics Window	108
3.5	Write Window	110
3.6	The Data Window.....	111
3.6.1	Configuration of Signals	111
3.6.2	Display Types	112
3.6.3	Activity Indicator	114
3.6.4	Peak Indicator	114
3.6.5	Optimization of Data Display	114
3.7	Statistics Window	115
3.7.1	Direct Display in the Statistics Window.....	115
3.7.2	Statistics Report	116
3.7.3	Choosing a Histogram.....	117
3.8	Bus Statistics Window	118
4	Blocks and Filter.....	120
4.1	Generator Block	121
4.1.1	Configuration of Triggering	122
4.1.2	Configuration of Transmit List.....	122
4.1.3	Entry of Signal Values	123
4.1.4	Entry of Mode-Dependent Signals.....	124
4.1.5	Function Generator for the Transmit List.....	124
4.2	Interactive Generator Block (IG).....	126
4.2.1	Configuring the Interactive Generator Block.....	127
4.2.1.1	Transmit List	127
4.2.1.2	Value Generator.....	128
4.2.1.3	Trigger Condition.....	128
4.2.1.4	Generating a High-Load Situation.....	128
4.2.1.5	Entering Signal Values	129

4.2.1.6	Entering Mode-Dependent Signals	129
4.2.1.7	Keyboard Control	129
4.2.2	The Interactive Generator Block as a Gateway	130
4.3	Replay Block	131
4.3.1	Replay of Environment Variables	131
4.3.2	Example of an ASCII Replay File with Environment Variables	132
4.4	Trigger block.....	132
4.5	Filter block.....	133
4.6	Channel Filter	134
4.7	CAPL Nodes in the Simulation Setup.....	135
4.8	CAPL Nodes in the Measurement Setup.....	137
4.9	Environment Variable Filters in the Measurement Setup	138
4.9.1	Pass and Stop Filters for Environment Variables	138
4.9.2	Configuration of the Environment Variable Filter	138
4.9.3	Selecting Environment Variables.....	138
4.9.4	Examples of Preselection of Environment Variables	139
4.10	Break.....	139
5	The Panel Editor	140
5.1	Introduction	140
5.2	Editing a Panel	143
5.3	Display and Control Elements	143
5.3.1	Creating Elements.....	143
5.3.2	Configuration of the Elements	145
5.3.3	Arranging the Control Elements	145
5.4	The ActiveX control	147
5.4.1	The ActiveX control wizard	147
5.5	The Hexadecimal Editor	148
5.6	Working with Bitmap Controls	149
5.6.1	Bitmap File Format.....	149
5.6.2	Configuration of Bitmap Elements	150
5.6.3	Color Resolution of Bitmaps.....	151
5.7	Overlapping.....	152
5.7.1	Transparency Color.....	152
5.7.2	Background Bitmaps	152
5.8	Test Mode	152

5.9	Panel Control	153
6	CAPL Programming.....	154
6.1	Overview	154
6.1.1	Potential Applications of CAPL Programs.....	154
6.1.2	Integration of CAPL Programs.....	155
6.1.3	Use of the Symbolic Database in CAPL	156
6.1.4	Introduction to CAPL	156
6.2	CAPL Browser.....	157
6.2.1	Opening Browser.....	159
6.2.2	Browser Window	159
6.2.3	Compiling CAPL Programs.....	159
6.2.4	Searching for Run-Time Errors.....	160
6.2.5	Access to the Database	160
6.2.6	Importing and Exporting ASCII Files.....	161
6.2.7	Browser Options.....	161
7	Option .CAN	162
7.1	The Trace Window of Option .CAN	162
7.2	The Bus Statistics Window of Option .CAN.....	163
8	Option .LIN	164
8.1	Configuration of a LIN Test Environment.....	164
8.2	LIN Scheduler	164
8.3	LIN Simulator	164
8.4	LIN Specifications.....	165
8.5	The Converter Tool LDF to DBC.....	165
8.6	Trace Window for Option .LIN	165
8.7	The Bus Statistics Window for Option .LIN.....	166
9	Option .MOST.....	167
9.1	Installation Procedure.....	167
9.1.1	Prerequisites	167
9.1.2	Procedure.....	167
9.2	Profile	168
9.3	Timestamps.....	168

9.4	Synchronized timestamp	168
9.5	Original timestamp	169
9.6	Time Synchronization Accuracy	169
9.7	Database Support	169
9.7.1	Function Catalogs in XML	170
9.7.1.1	Using XML Function Catalogs in CANoe/DENoe	170
9.7.1.2	Setting disassembly mode	170
9.7.1.3	Validation	170
9.7.1.4	Import Errors and Warnings	171
9.7.1.5	Using more than one XML Function Catalog	171
9.7.1.6	Reading and Rereading XML files.....	171
9.7.2	CANdb++	172
9.7.3	CAPL	172
9.7.4	Frame	172
9.7.4.1	Lookup Key.....	173
9.7.4.2	Message Name.....	173
9.7.4.3	Message Attributes	173
9.7.4.4	Parameters	173
9.7.5	HW Configuration.....	174
9.8	Interactive Generator Block MOST	175
9.9	Trace Window	175
9.10	Busstatistic Window	177
9.11	Graphic- & Data Window.....	177
9.12	CAPL.....	177
9.12.1	Initialization of Message Variables	177
9.12.2	Selectors	178
9.12.3	Event Procedures.....	179
9.12.3.1	on mostMessage.....	179
9.12.3.2	on mostRawMessage	180
9.12.3.3	on mostLightLockError.....	180
9.12.4	Functions.....	180
9.12.4.1	MostGetFBlockID, MostGetFunctionID, MostGetOpType.....	180
9.12.4.2	MostSetFBlockID, MostSetFunctionID, MostSetOpType	180
9.12.5	Hardware API	181
9.12.5.1	Optolyzer Operation Mode.....	181
9.12.5.2	Node Mode	181

9.12.5.3 Spy Filter	182
9.12.6 Error Codes of CAPL functions	183
9.12.6.1 kMostTxQueueFull = -6.....	183
9.12.6.2 kMostWrongOptoMode = -5.....	183
9.12.6.3 kMostWrongThread = -4	183
9.12.6.4 kMostIllegalTime = -3.....	184
9.12.6.5 kMostNoConnection = -2	184
9.12.6.6 kMostInvalidChannel = -1	184
9.12.6.7 kErrDrvOK = 0	184
9.13 Demo Configurations CANoe/DENoe.....	184
9.13.1 MOSTSpy	184
9.13.2 MOSTGeneral	184
9.13.3 MOST XML Catalog	185
9.14 Known Problems & Trouble Shooting.....	185
9.14.1 Measurement Windows frequently disabled.....	185
9.14.2 CANoe doesn't receive any MOST Frames	185
9.14.3 CANoe doesn't receive some MOST Frames.....	185
9.14.4 Warning '2 Tx receipt(s) got lost'	186
9.14.5 After several minutes CANoe doesn't receive or send any MOST Frames.....	186
9.14.6 Error 'Port COM1 busy for channel 1'	186
9.14.7 Error 'No valid license for channel 1'	186
9.14.8 Weird Spurious Events from MOST bus	187
9.14.9 Transmit Acknowledgement	187
9.14.10 Timestamps of CAN and MOST constantly drift apart.....	187
9.14.11 Measurement stops after configuration of trace window	187
9.15 XML Engine.....	188
10 Option .FlexRay	189
10.1 Trace Window for Option .FlexRay.....	189
10.2 The Bus Statistics Window for Option .FlexRay	190
11 Index	191

1 Introduction

In this chapter, you get an overview about the purpose and functionality of CANoe.

A short tutorial leads you through the essential components of CANoe and roughly acquaints you with the individual functions.

1.1 Overview

CANoe is a universal development, test and analysis environment for CAN bus systems, which is made available to all project participants over the entire development process. The system producer is supported in functional distribution, functional checking and integration of the overall system. The supplier obtains an ideal test environment by simulation of the remainder of the bus and environment.

The development process is based on a phase model which differentiates between three development stages (see Figure 1).

Phase 1: Requirements analysis and design of the networked system

First, the party responsible for design distributes the overall functionality of the system among different network nodes and refines the design to the level of the network node. This includes defining messages and selecting the baud rate of the bus. Finally the bus behavior of individual network nodes must be specified, e. g. in the form of cycle times or more complex protocols. Then this information can be evaluated first by the simulation tool to provide initial estimates of bus load and the latency times to be expected at the prescribed baud rate. Afterwards, this specification can also be utilized for testing in subsequent phases.

For a more accurate study, a dynamic functional model of the overall system is created. This involves specifying the behavior of the network nodes with regard to input and output variables and the messages to be received and transmitted. Especially useful here is an event-driven model with a procedural description of behavior. For example, the model may describe how - after receiving a message (Event) - the received data are to be further processed (procedural) and how the result is to be output as a control variable.

The user must also specify the input variables to the simulation tool, so that the time behavior of network nodes and the accumulation of messages can be simulated. The results of the simulation serve to validate the design and can later be used as a reference after implementation.

Phase 2: Implementation of components with simulation of remainder of the bus

After the first phase has been completed the design and development of individual network nodes is usually performed by all participants, independently and in parallel. The models for the other network nodes can now be used to simulate the remainder of the bus for testing of a developed network node. The tool requires an interface to the real bus for this, and it must be able to conduct the simulation in real time.

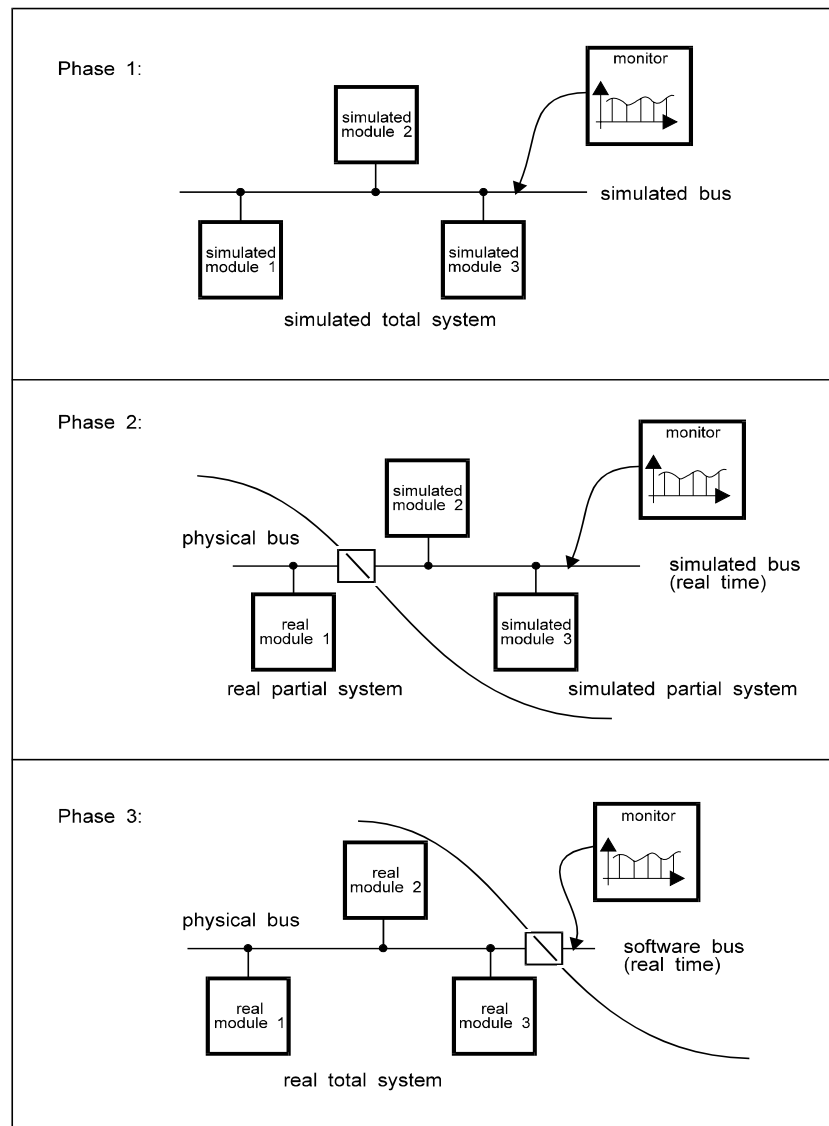


Figure 1 : Phase model of the development process

Phase 3: Integration of the overall system

In this last development phase all real network nodes are connected to the bus in a step-by-step manner. To accomplish this it must be possible to "disconnect" the models one-by-one in the simulation of the remainder of the bus. The tool serves increasingly as an intelligent analysis tool which observes the message traffic between the real network nodes on the bus and compares the results with the specified requirements.

The behavior of network nodes with regard to input and output signals is described with the help of environment variables. CANoe differentiates between discrete and continuous variables. Switch positions can be represented as discrete environment variables. With continuous environment variables, dimensions such as temperature or engine RPM can be described.

The control panels provide a user-friendly interface to the environment variables. The user can create the panels independently with the help of the Panel Editor. During the simulation values of environment variables can be displayed (lamps, counters) and interactively modified (switches, potentiometers).

The example in Figure 2 is intended to clarify the functions which CANoe provides for simulation and testing of CAN bus systems.

By pressing the pushbutton on the left control panel the discrete environment variable "Pushbutton" is set to the value 1. The bus node on the left reacts by sending out a message on the CAN bus. The bus node in the middle receives this message and sets the discrete environment variable "Light" to 1. This causes the small lamp in the middle control panel to light up.

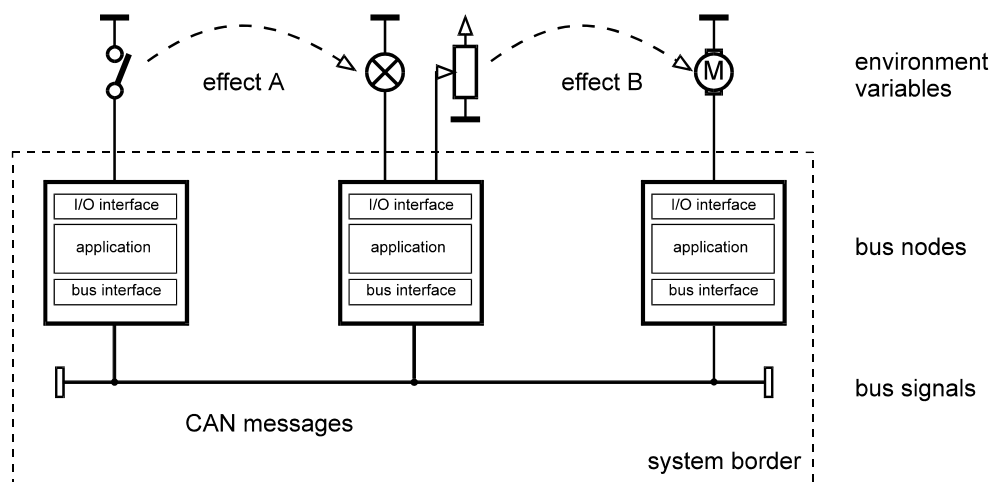


Figure 2: Components of the simulation system

Analogously, the user can also adjust the potentiometer in the middle control panel, whereby the value of the continuous environment variable "Potentiometer" is modified. This causes the middle network node to place a message on the bus with the new data. This message is received by the network node on the right. There a new value is calculated from the signal contents for the environment variable "Engine RPM". Finally, this causes the display of engine speed to be updated on the right control panel.

The behavior presented in the previous sections can be described very easily with functions available in CAPL. By this method it is possible to implement a simulation of complex systems with relatively little effort.

1.2 Tips for Using CANoe

The basic operating procedures for using CANoe 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, CANoe 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

CANoe 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:

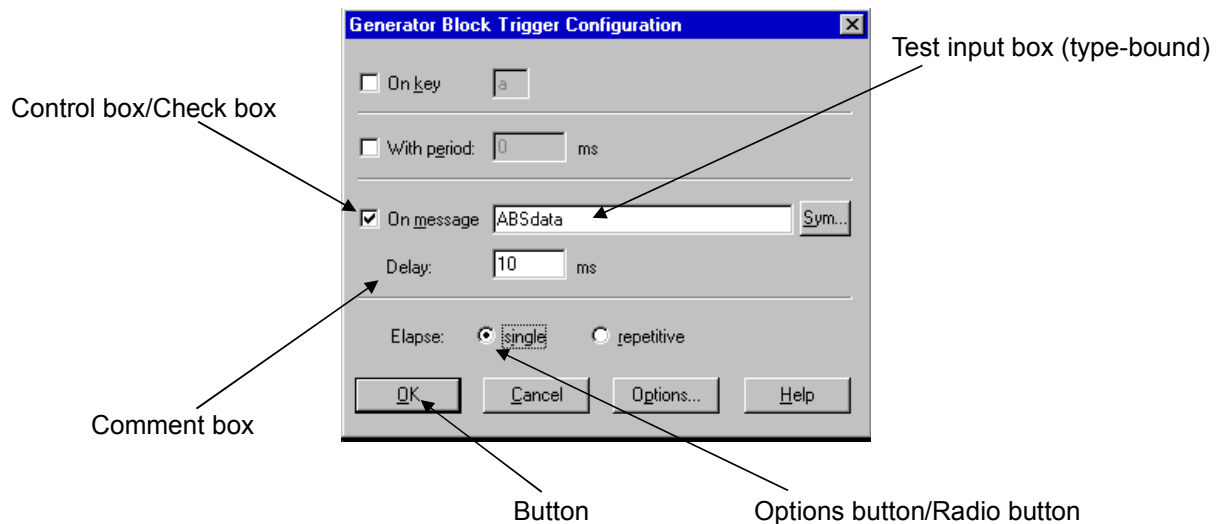


Figure 3: Box Types in Dialogs

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 CANoe dialogs also have an **[Options]** button. When this button is activated another dialog appears with which you can modify the CANoe 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 Measurement and Simulation Setup

CANoe is primarily configured in the measurement setup and simulation setup windows, in which the program's data flow plans are displayed.

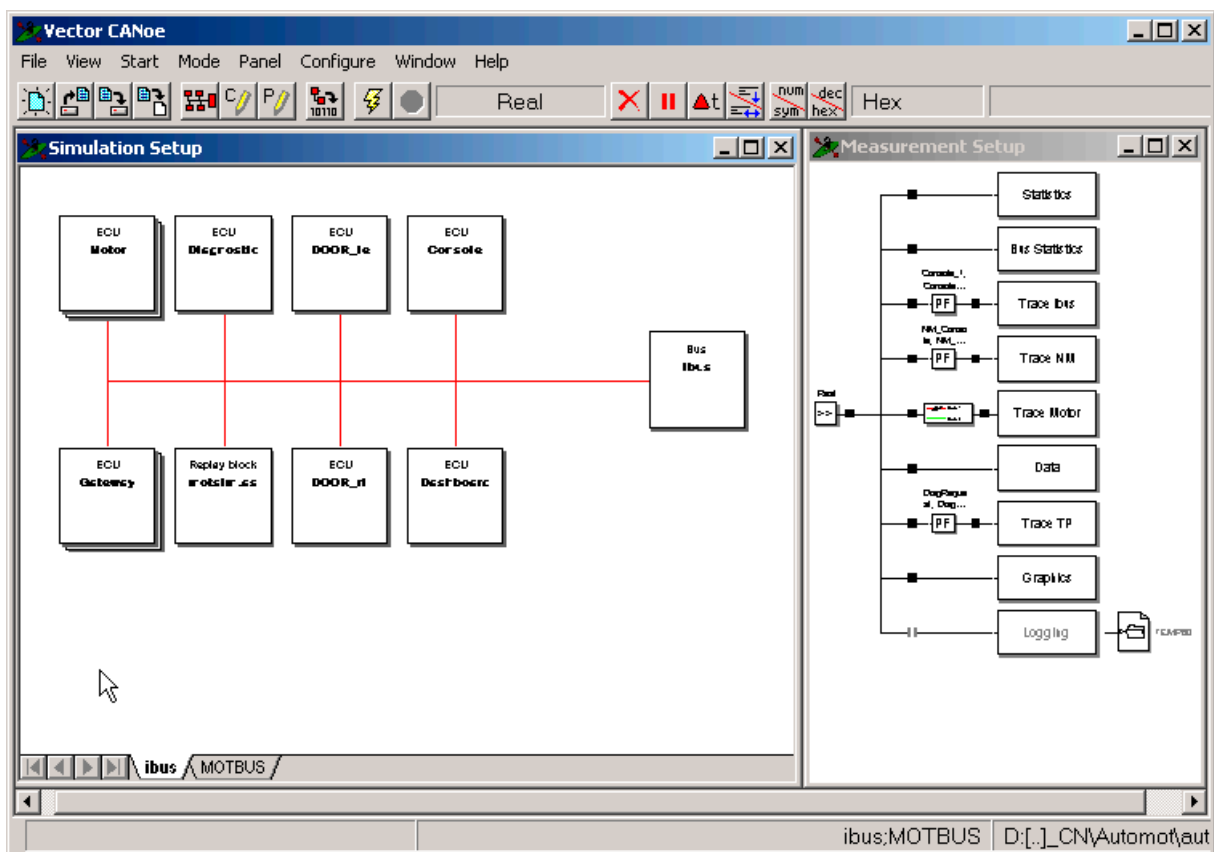


Figure 4: Measurement Setup and Simulation Setup of CANoe

Mouse Operation

All blocks and some images in the measurement setup and simulation windows are mouse sensitive. When selected by clicking the left mouse button, the element preselected in this manner is identified by a frame as the *Active Element*. When the right mouse button is clicked, a popup menu appears in which the object is configured by the methods described above. As an alternative, the con-

figuration dialog for the active block can be called directly by double clicking with the left mouse button.

Keyboard Operation

When the measurement setup or simulation setup is active and <Tab>, <Backspace> or one of the cursor keys is activated, the preselect frame around the currently active element is indexed forward. <Tab> results in forward indexing (<Backspace>: 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.

You can deactivate the preselected function block in the measurement setup and simulation setup with the spacebar, or reactivate it by pressing the spacebar again.

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

1.2.4 Online Help

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, the CANdb++ Editor, the Panel Editor, the Panel Generator and the CAPL Generator 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 CANoe Tour

If you are starting up CANoe 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 CANoe assumes the roles of both sender and receiver. In the first step CANoe is configured as a data source, i.e. as a transmitting station. You will then learn about of CANoe's analysis options by studying the generated data in the measurement windows afterwards.

In complex real systems CANoe 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 create two network nodes of a distributed system to solve a simple simulation task in CANoe.

1.3.1 Preparations

To start CANoe, call `CANOE32.EXE` by double clicking the appropriate icon in the CANoe program group.

CANoe has various evaluation windows (Trace, Data, Graphics, Statistics and Bus Statistics windows) as well as a measurement setup window and a simulation setup window which show you the data flow and simultaneously allow you to configure CANoe.

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

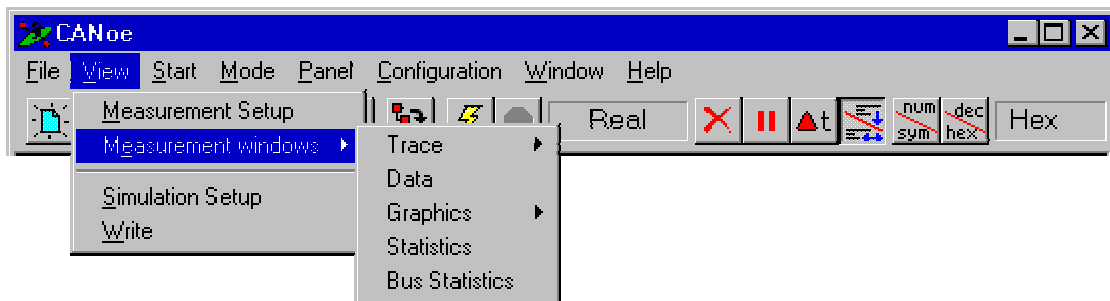


Figure 5: **View** Menu on Main Menu Bar

In the simulation setup window the overall system is shown graphically with the CAN bus and all network nodes. The simulated bus is represented by a red horizontal line. The black line beneath it symbolizes the real bus. The two buses are connected to one another via the PC-card.. To transmit data from CANoe onto the bus, insert transmit blocks in the simulation setup, which must be connected by the red line.

The data flow diagram of the CANoe measurement setup has a connection to the simulation setup on the left - symbolized by the `>>` symbol - and various evaluation blocks on the right serving as data sinks. That is, the data flow is from left to right. 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.

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**. The Simulation Setup wizard starts, which we do not need for this tour. Therefore you have to exit the wizard by selecting the **[Cancel]** button.

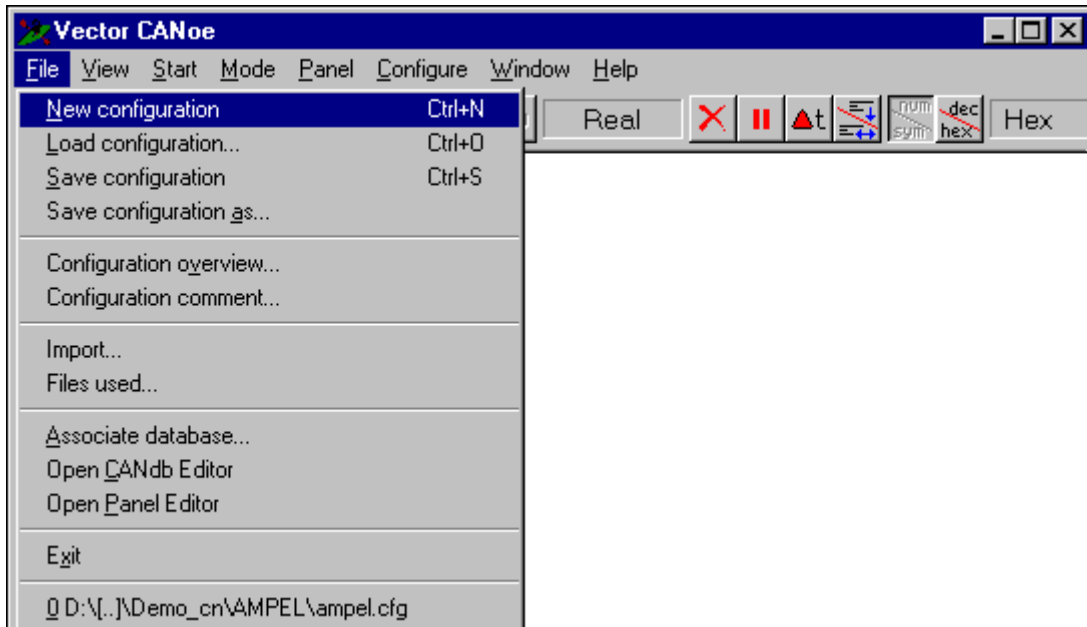


Figure 6: Menu Item **File|New configuration**

1.3.2 Setting Up the CAN Bus

To start up CANoe 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 CANoe 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.

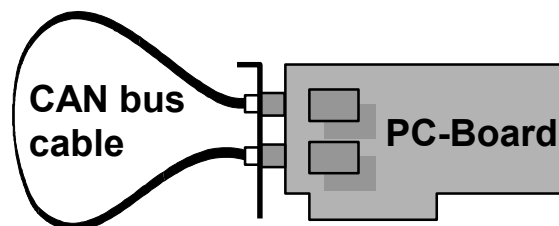


Figure 7: PC-Card with Connection Cable

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 protocol 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 simulation setup to the foreground and click the right mouse button on the square that represents the bus system.

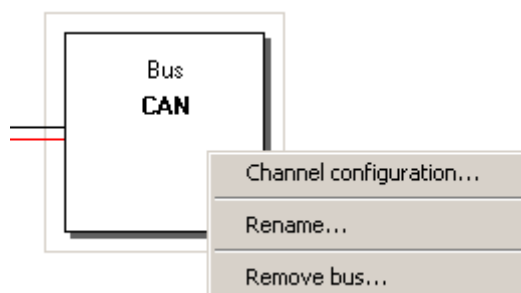


Figure 8: Popup Menu of the bus symbol

Please select **Channel configuration...** from the popup menu.

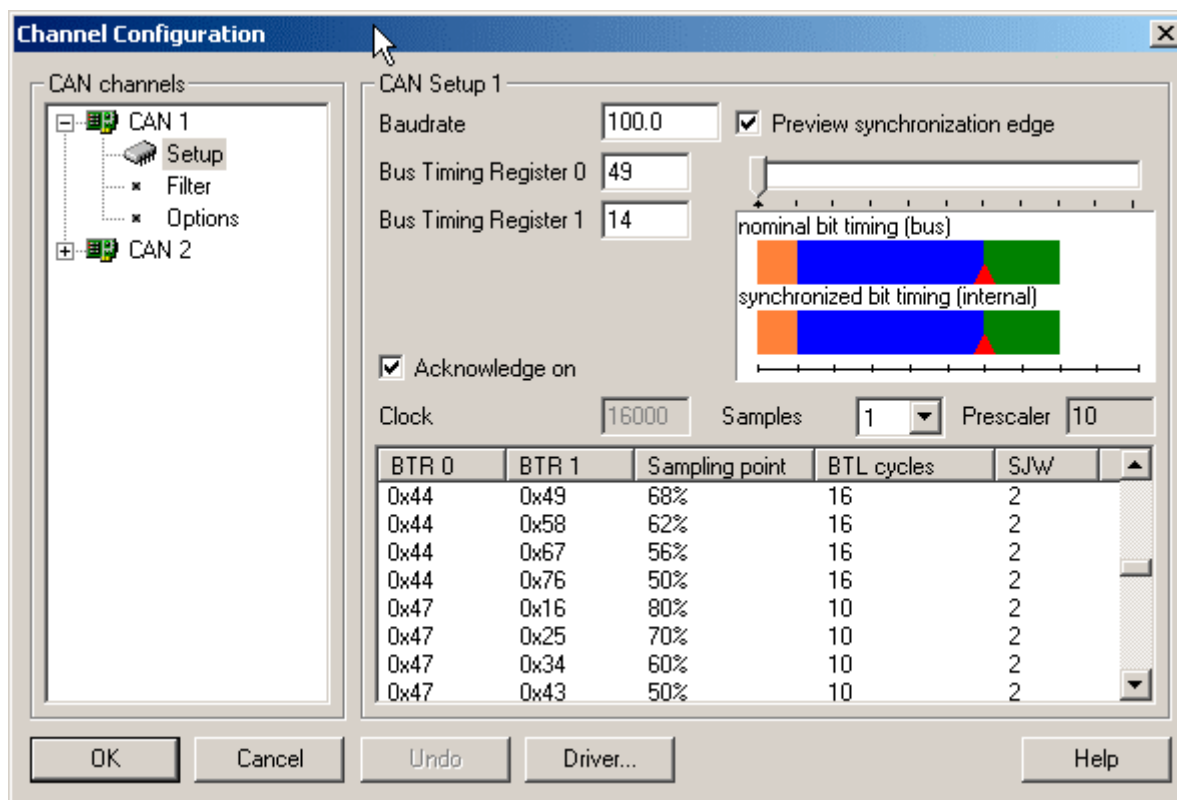


Figure 9: Configuration of Bus Parameters and Direct Baud Rate Selection

After then select **+** and **Setup** from the configuration dialog for the first controller CAN 1 and type in the value for the baudrate 100kBaud. This makes sense for both high-speed and low-speed buses. After you activate the **[Update]** button, CANoe recommends default values for the controller registers, which you accept with **[OK]**. 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. When you exit the dialog, CANoe asks you whether the parameters should be accepted. Answer with **YES**.

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 CANoe which places information on the bus cyclically.

Unit 1: Configure CANoe 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 CANoe's simulation setup which generates the message to be transmitted. This is done by clicking with the right mouse button on the bus lines in the simulation setup, and - from the popup menu - inserting a generator block on the bus.

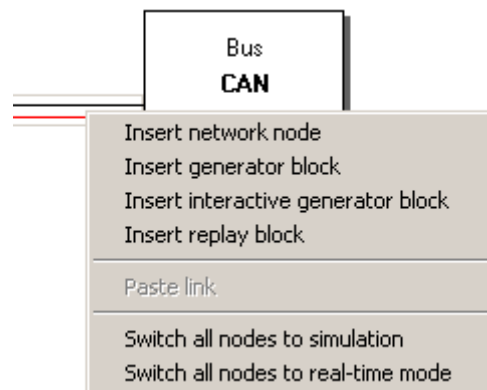
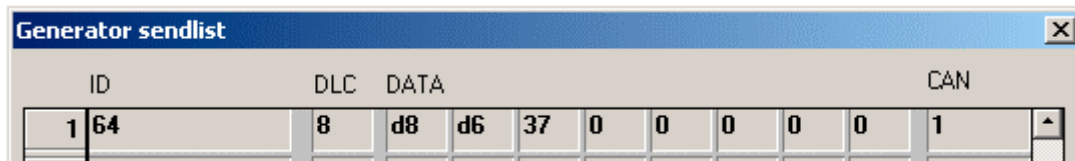


Figure 10: Bus Symbol in Simulation Setup with Popup Menu

Afterwards, this appears in the simulation setup 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.



ID	DLC	DATA	CAN
1	64	8 d8 d6 37 0 0 0 0 0	1

Figure 11: 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.

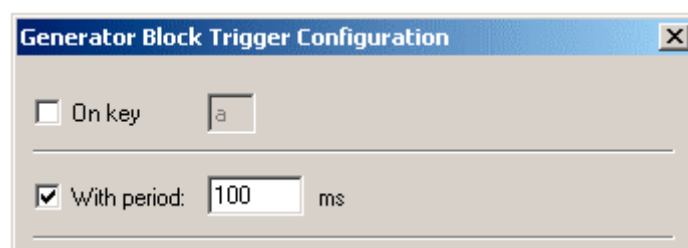


Figure 12: Triggering of Generator Block

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.

Furthermore, CANoe requires that you associate a database to the configuration before the start of measurement. Therefore, you should initially assign the database `MOTBUS.DBC` from the demo directory `DEMO_CAN_CN\AUTOMOT\CANDB` to your active CANoe configuration. The concrete benefits of this database will be made clear in the sections that follow.

You can use the simulation setup to edit databases (add, assign, delete, etc.). In the system view window of the simulation setup, you can see a tree representation of the current configuration. If you go to **Databases** with the mouse pointer and click with the right mouse button, you will bring up the **Add...** command in the context menu. You can use this command to add an (additional) database to the current bus.

After you have brought up the command, the **Open** window appears and you can select a database.

If you click on the **[OK]** button, the new database is accepted for the current bus and displayed in the system view window.

Start the measurement by pressing the start button on the toolbar. CANoe 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.

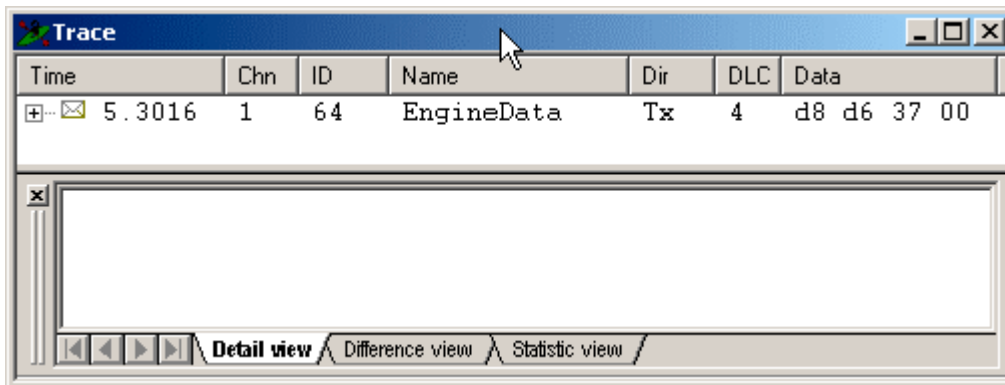


Figure 13: Trace Window

The next column shows you which of the two CAN channels was used to transmit. This value (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 **Channel configuration.../CAN 2/Filter**.

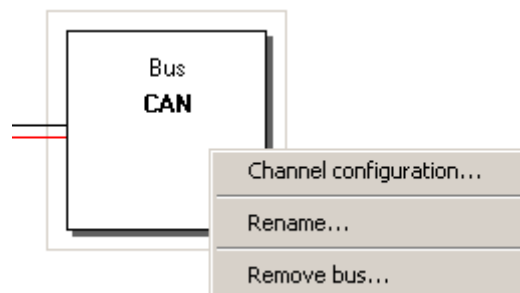


Figure 14: Popup Menu of the bus symbol

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.

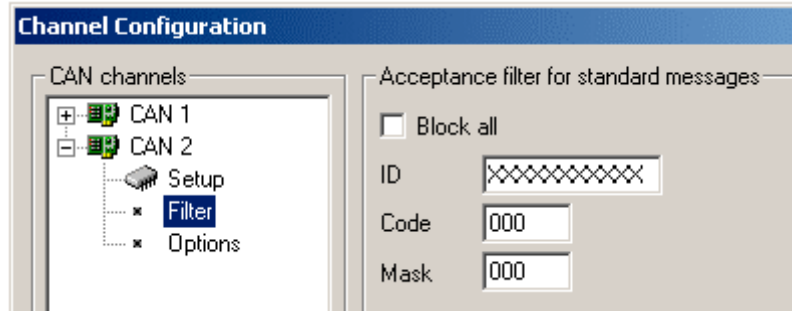


Figure 15: Configuration of Acceptance filter

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).

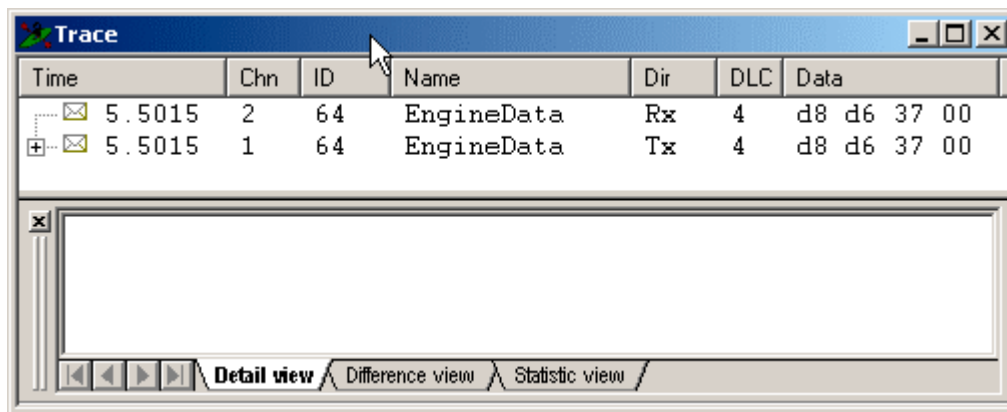
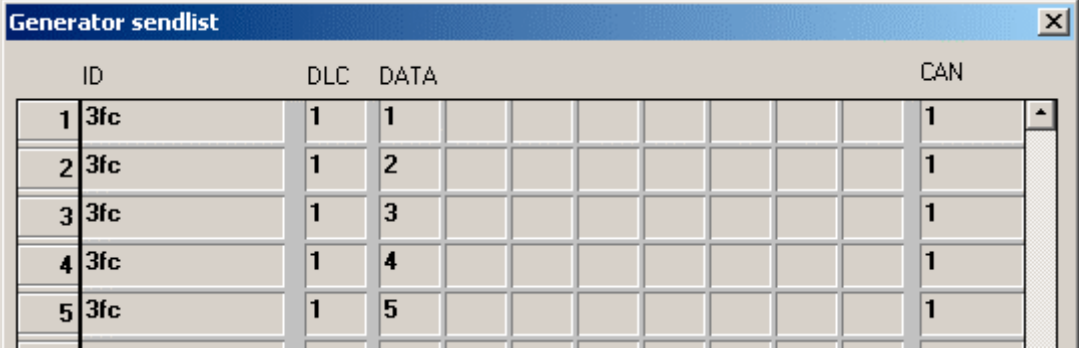


Figure 16: Result in the Trace Window

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 simulation setup. 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
2	3fc	1	2
3	3fc	1	3
4	3fc	1	4
5	3fc	1	5

Figure 17: Transmit List for Generator Block


Do not forget to stop the measurement before you reconfigure the simulation 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, CANoe 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 CANoe's logging function. A program block allows you to integrate your own transmit functionalities - which may be quite complex - into CANoe with the CAPL programming language.

1.3.4 Evaluation Windows

Evaluation windows are used to analyze data generated by the generator blocks in the simulation setup.

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 simulation setup 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).

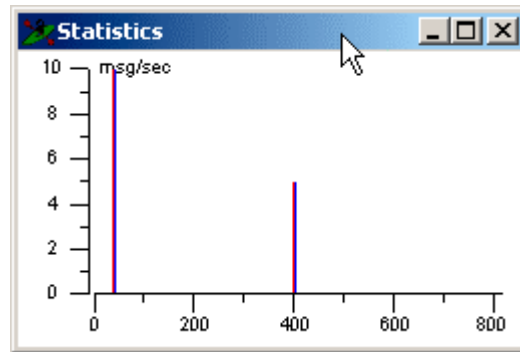


Figure 18: Statistics Window

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 configuration dialog of the Statistics block (Measurement Setup).

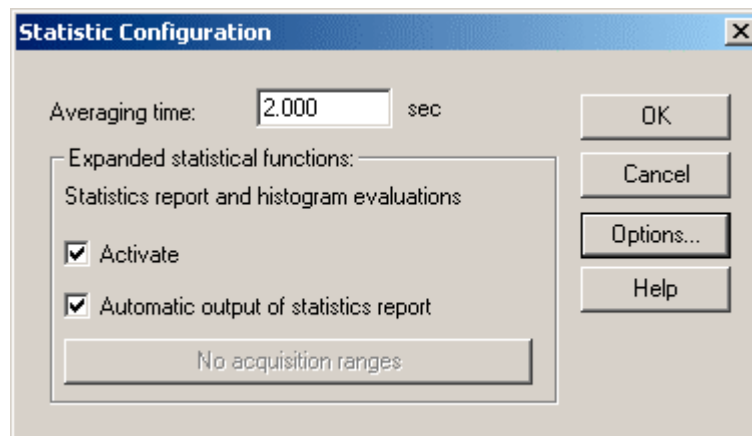
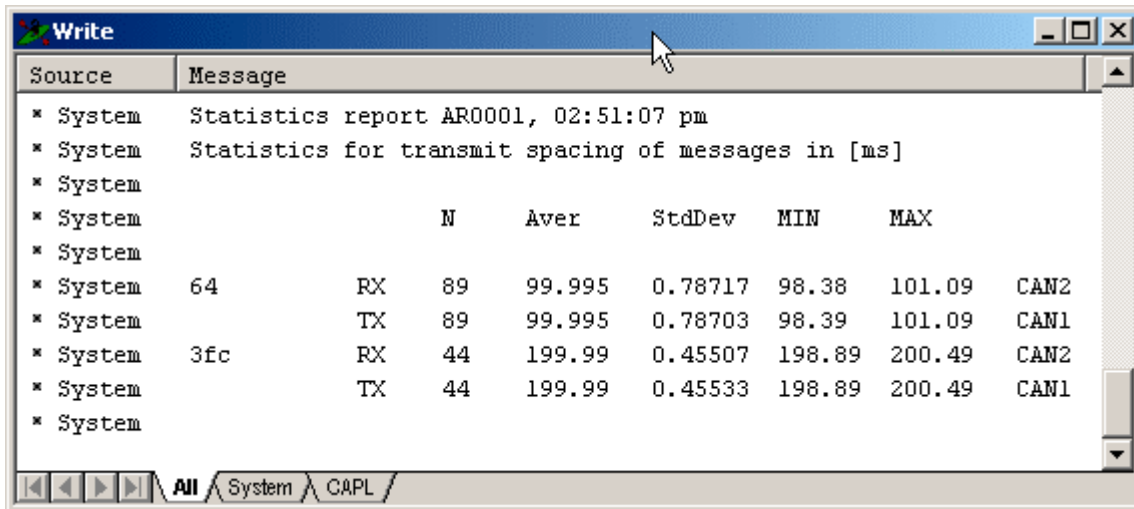


Figure 19: Activate the statistical report

If the Activate check box of the expanded statistical functions is selected, a Statistics report is generated for at least one acquisition range during the measurement. After the end of the measurement this can be output to the Write window by the **Display statistics report** command in the Statistics block's popup menu.

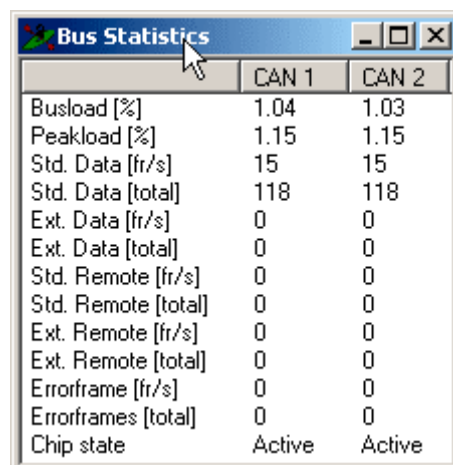


Source	Message	N	Aver	StdDev	MIN	MAX	
* System	Statistics report AR0001, 02:51:07 pm						
* System	Statistics for transmit spacing of messages in [ms]						
* System							
* System							
* System							
* System	64	RX	89	99.995	0.78717	98.38	101.09 CAN2
* System		TX	89	99.995	0.78703	98.39	101.09 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 20: 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
Busload [%]	1.04	1.03
Peakload [%]	1.15	1.15
Std. Data [fr/s]	15	15
Std. Data [total]	118	118
Ext. Data [fr/s]	0	0
Ext. Data [total]	0	0
Std. Remote [fr/s]	0	0
Std. Remote [total]	0	0
Ext. Remote [fr/s]	0	0
Ext. Remote [total]	0	0
Errorframe [fr/s]	0	0
Errorframes [total]	0	0
Chip state	Active	Active


Figure 21: 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 CANoe 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, CANoe 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 CANoe product for further information on the CANdb++ editor.

At this point we would like to use the database `MOTBUS.DBC`, which you have already associated to the active CANoe configuration. This database will be used to interpret the data bytes of the messages generated by the generator blocks in the transmit branch. To do this, first 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.

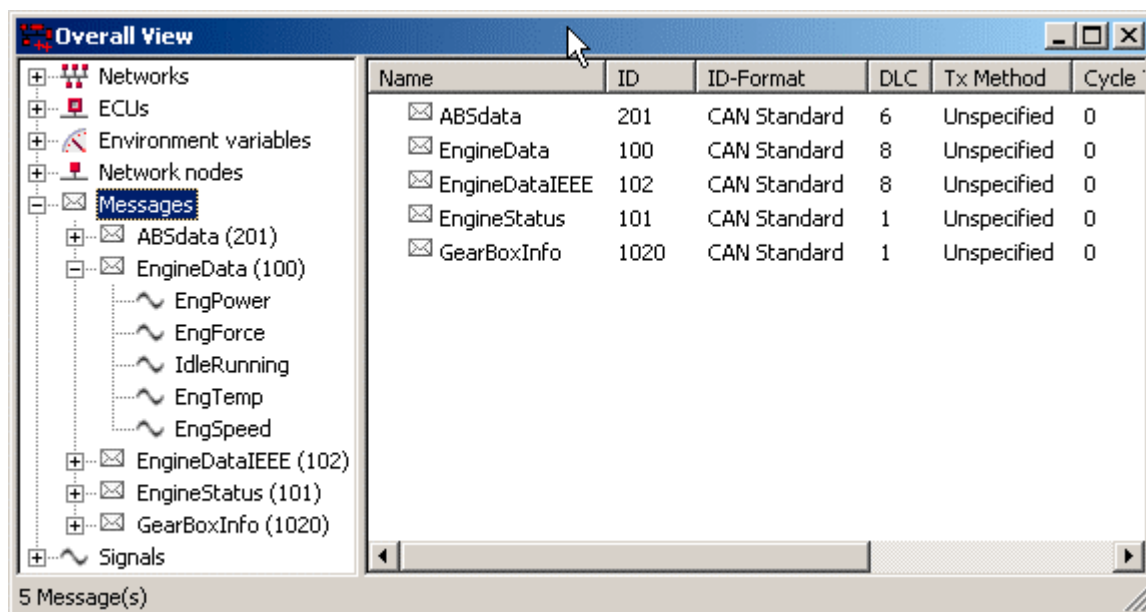


Figure 22: Overall View Window of the CANdb++ Editor


Double click the Messages object type in the area on the left side of the Overall View window. The subordinate structural level is then also shown in this area, and the area on the right shows the available messages 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 CANoe. 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 CANoe. 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  button.

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

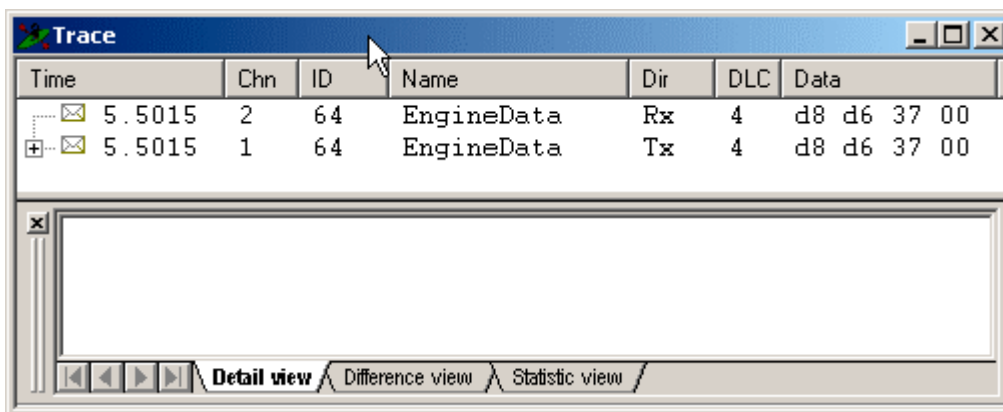


Figure 23: Trace Window

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.

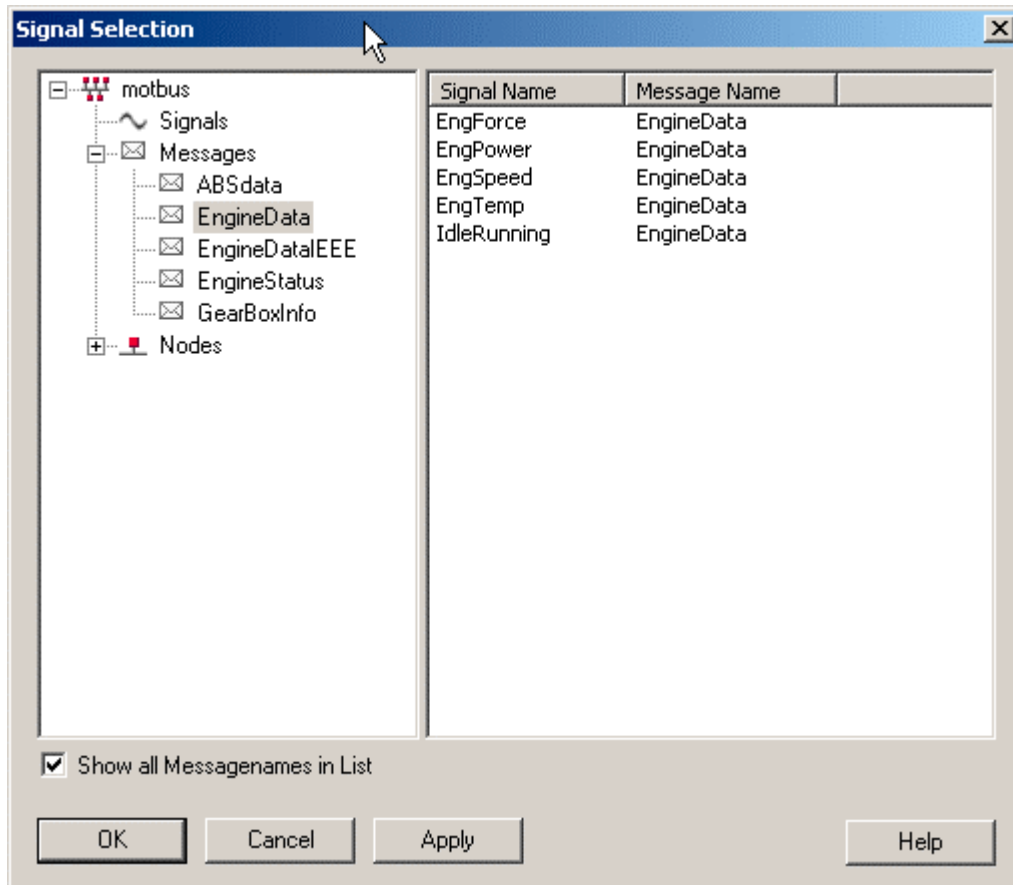


Figure 24: Selecting Signals with the Signal Explorer

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.

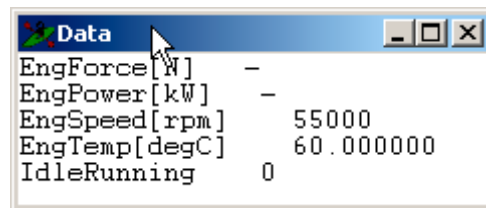


Figure 25: Data Window

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.

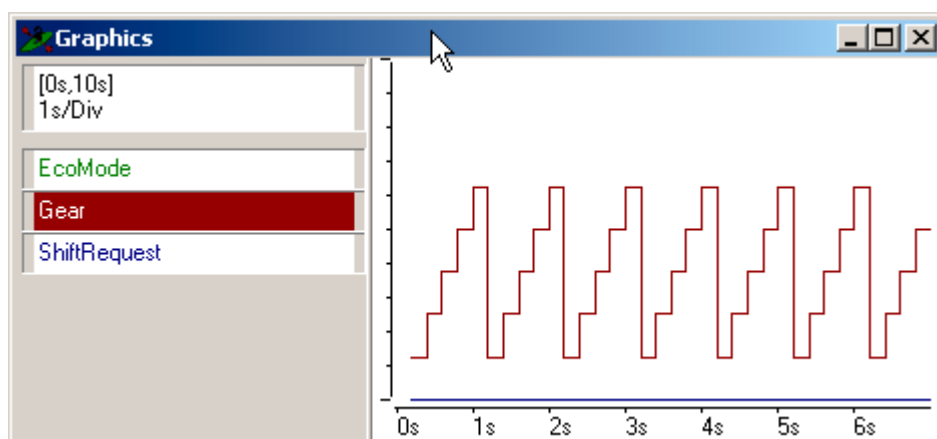


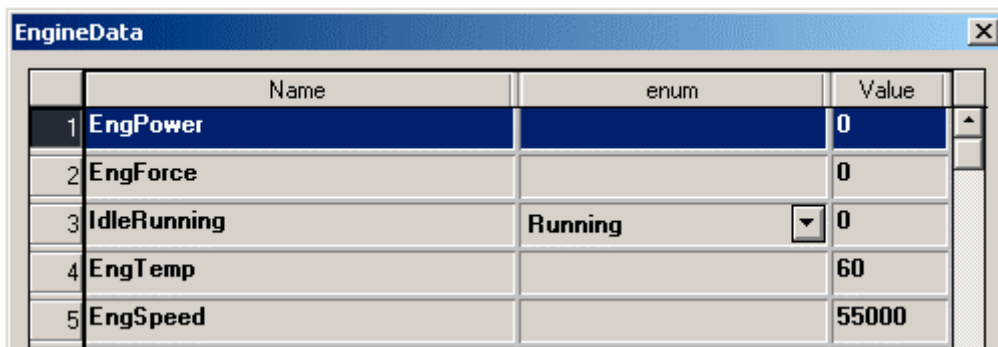
Figure 26: Graphics Window

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 3.4.5.

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, open the transmit list of the generator block of task 1. Instead of the identifier that you previously entered in the transmit list (64), 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* = 55000 rpm, *EngTemp* = 60 degrees Celsius and *IdleRunning* = 0.




	Name	enum	Value
1	EngPower		0
2	EngForce		0
3	IdleRunning	Running	0
4	EngTemp		60
5	EngSpeed		55000

Figure 27: 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 Logging a Measurement

CANoe 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 5: Log - in ASCII format - all CAN data traffic that is generated in a short measurement (approx. 20 sec.) by the generator blocks in the simulation setup.

To log the data that arrive in CANoe's measurement setup to a file, first activate the logging branch. Also remove the break that separates the logging block of a new con-

figuration from the data source. You can do this by double clicking the break symbol or with the popup menu of the break (hot spot). 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.

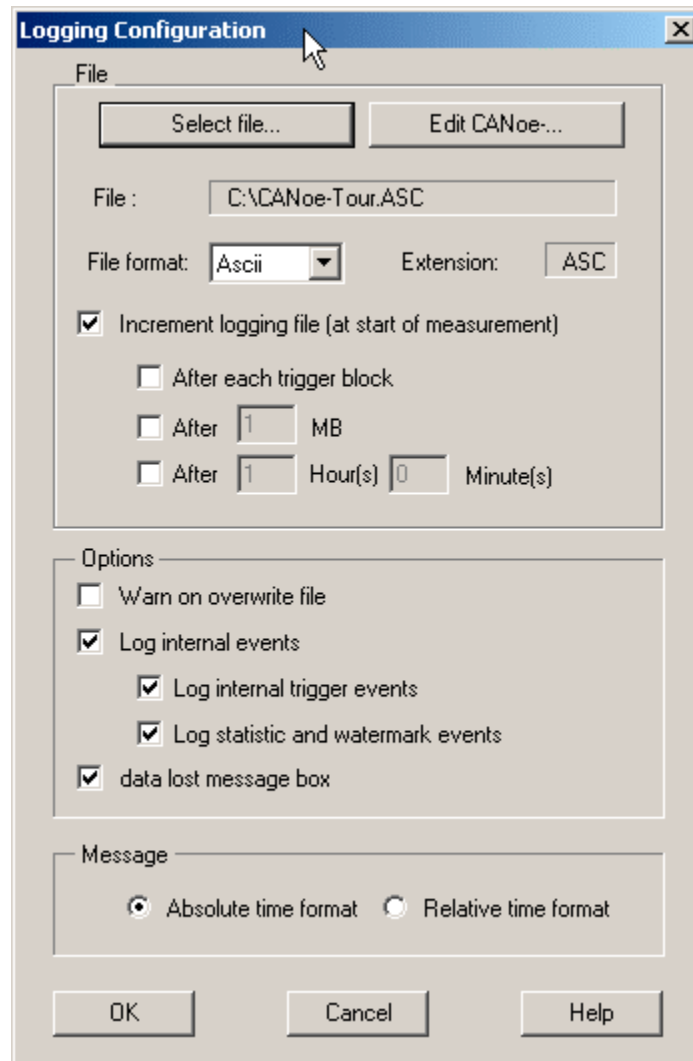


Figure 28: Configuration Dialog in the Logging Branch

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 2.7.1.1. 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.

1.3.10 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 CANoe provides for offline analysis of log files.

Unit 6: 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 CANoe 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 bus symbol - is a file icon. 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.

Also you have to separate the logging block. You can do this by double clicking the hot spot symbol in front (left) of the Logging block or with the popup menu of this hot spot.

You can now play back the measurement with the F9 key. In contrast to Online mode, here CANoe 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.

1.3.11 Creating a CAPL Program

Although CANoe provides you with a large number of transmit and analysis functions, which you can parameterize in specific configuration dialogs, sooner or later you will need to expand the functionality of CANoe with your own functions for your special task requirements. This is why CANoe offers you the C-like programming language CAPL.

In the next task you will create a simple CAPL program to count messages that are generated in CANoe's simulation setup. You will find a complete description of the programming language together with numerous detailed examples in the online help.

Unit 7: Create a CAPL program with which you can count the number of messages of the type *EngineData* (ID 64 hex) and output the counted number of messages to the Write window in response to a key press.

First, switch CANoe back to Online mode. In the simulation setup the generator block which sends *EngineData* messages cyclically onto the bus should still be the data source.

First you must decide where you wish to insert your CAPL program in the data flow plan. In principle, any hotspot in the measurement setup or in the simulation setup is available to you. However, since this program is solely for analysis purposes and does not generate any messages itself, but only counts them, it is advisable to insert the program on the right side of the measurement setup, perhaps before the Statistics block. In the hotspot's popup menu choose the function **Insert CAPL node**. A function block with the program symbol **P** now appears at the selected point in the measurement setup. You can also access the node's configuration dialog via the popup menu. First, select a program name, e.g. `COUNTER.CAN`, and then start the CAPL Browser either from the configuration dialog's **[Edit..]** button or directly by double clicking the program block in the measurement setup.

CAPL is an event-based programming language. Each CAPL program consists of event procedures, with which you can react to external events (e.g. occurrence of specific messages on the CAN bus or activation of keys on the PC keyboard). The CAPL Browser is described in detail in the online help. With its sub-windows ("Panels") it allows you to create and edit CAPL programs quickly and easily.

In principle, you can also use your own text editor to create CAPL programs. CAPL programs are normal ASCII files with the default name extension `*.CAN`, which must be compiled before the start of measurement using the compiler provided with the CANoe product.

For your program you will first need an integer variable which counts the messages. For example, you could name it `counter`. Go to the upper right Browser pane and enter this name in the variables block. The following should now appear in this pane:

```
variables {
    int counter;
}
```

Like all global variables, this variable is automatically initialized to zero at the measurement start.

In the next step, this variable should be incremented whenever an *EngineData* message is registered. Therefore, you must expand the CAPL program to include an event procedure of the type **on message** ("React to message event"). To do this, click the event type **CAN Messages** in the Browser tree using the right mouse button and insert a new event procedure of this type using the command **New** from the popup menu.

Now a procedure template appears in the Procedures Text Editor. First replace the text `<newMessage>` by the symbolic name **EngineData**, which you could also assume directly from the database via the popup menu item **CANdb Message**. During

compilation the CAPL compiler replaces the symbolic name by the corresponding identifier 0x64.

Now you only need to define which actions should be performed when the event occurs. Since the program is to count messages, the variable `counter` must be incremented whenever a message is registered. The complete procedure appears as follows:

```
on message EngineData
{
    counter++;
}
```

As a last step, the output to the Write window must still be implemented. Finally, the program should not just count messages, but also keep track of how many messages have been counted.

The output to the Write window should occur when the 'a' key is pressed. Therefore, you must define another event procedure for the event "Press key 'a'". In the Browser tree you select the type **Keyboard**. This causes the previously defined **on message** procedure to disappear, since it belongs to a different event type. Of course it still remains a component of the CAPL program and will appear again as soon as you select the **Messages** event type again.

Now insert a **Keyboard** event in the CAPL program from the popup menu. A new procedure template will appear in the Procedures Text Editor, which you fill out as follows:

```
on key 'a'
{
    write("%d EngineData messages counted",counter);
}
```

The format entry `%d` refers to the integer variable `counter`, which is entered after the comma. For the most part, this format string conforms to the C function `printf()`.

That completes the program. Save it and then start the compiler either with the F9 key, or the main menu command **Compiler | Compile** or by the lightning icon button on the toolbar. If you have made an error in creating the program, a message window will open showing you the error. Double click this error message to go to the location where the error occurred. After you have corrected it and saved the program file again, recompile the program. Once the program has compiled without errors, the message *Compiled* appears in the status bar at the bottom of Browser's main window.

Now start the measurement. The generator block in the transmit branch begins to cyclically transmit messages of the type *EngineData*, which are now counted by your program. Whenever you press the 'a' key the text "`n EngineData messages counted`" can be seen in the Write window, whereby *n* represents the number of messages counted.

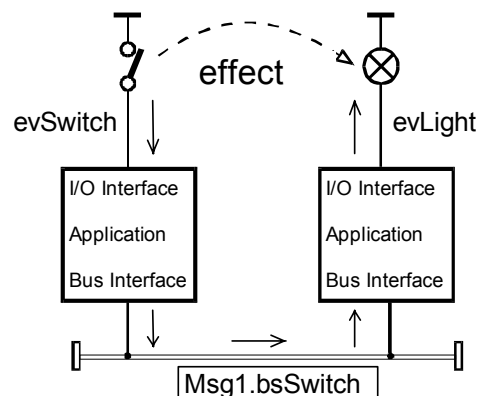
1.3.12 Simulation of Distributed Systems in CANoe

CANoe provides environment variables to model the functional bus behavior of network nodes. These environment variables are described by events and states of the system environment (external pressure, temperature, switch positions, etc.). You can observe and intentionally change these states - i.e. the values of the environment variables - on user-definable control panels.

To work with environment variables in CAPL you use the event procedure type `on envVar` (React to change in environment variable). The CAPL functions `getValue()` and `putValue()` are used to read and write environment variables.

These language tools and symbolic access to the various variables defined in the database make it possible to create simple prototypical network node models.


The following task consists of creating a complete CANoe configuration with two network node models and associated periphery, i.e. control panels. This should only involve implementation of distributed functions: After the user activates a switch, the first node informs the second node of this action. The second node then activates an indicator lamp in its periphery.



This simplest functionality was purposely selected to direct the user's attention to the creation of models and not to their functionality. More complex distributed systems can build upon the same conceptual pattern in CANoe without any difficulty.

A model for distributed systems can be created efficiently in CANoe in three steps: :

1. Create the database with messages, signals and environment variables
2. Create the network node periphery, i.e. the control panels
3. Create the network node models in CAPL

To prepare for the task you might, for example, create a new empty configuration by pressing the  button on the toolbar.

1.3.12.1 Creating the Database

The first step involves creating a database which describes the following two significant aspects of the system:

- The exchange of information between the two network nodes via the communication medium, i.e. the CAN bus; and
- The I/O interface to the periphery, i.e. the "wiring" between each node and its input and output units.

The database message and signal objects are available for describing the exchange of information over the CAN bus. The simple functionality of the example can be handled by a 1-bit signal which describes the state of the switch at the first node. This signal is packed in a CAN message and is only transmitted if the switch state changes (spontaneous transmission).

Therefore, you create a new database with the CANdb++ Editor, and in the database you create a message, e.g. with the name *Msg1* and identifier 100, which is to be transmitted by the first node. Create the signal *bsSwitch* to describe the switch position and link it to the message *Msg1*. In this case a signal length of one bit is sufficient, since only two states need to be transmitted, *On* (1) and *Off* (0):

The database provides you with environment variables for describing the I/O interface between the nodes and their peripherals. Each peripheral element (Switch, indicator lamp, slider, etc.) is "wired" to an environment variable, i.e. it is connected to the CAPL program for the network node.

In this example there are exactly two peripheral elements: A switch at the first node and an indicator lamp at the second node. Therefore, two environment variables must be created in the database, e.g. *evLight* and *evSwitch*:

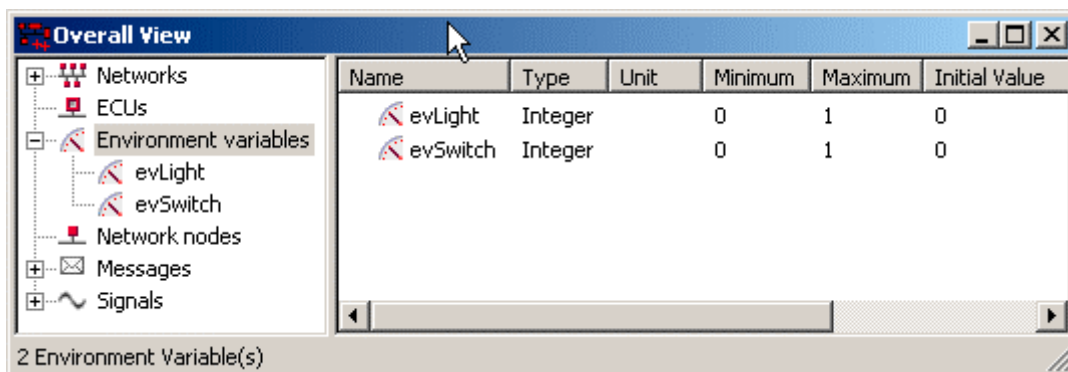


Figure 29: Environment variables in the database

Save the database, e.g. under the name `TOUR.DBC`, and associate it with your empty configuration. In the System View window of the Simulation setup you can add the database. If you go in the tree view of the current configuration to **Databases** with the mouse pointer and click on the right mouse button, you can run the command **Add...** from the context menu.

1.3.12.2 Creating Panels

A separate application, the Panel Editor, is provided in CANoe for creating the node's periphery. Please refer to chapter 5 for a detailed introduction to this editor.

In the configuration under consideration, one panel must be created for each of the two nodes. The first panel has a single control, a switch, while the second only has a small lamp as an indicating element:

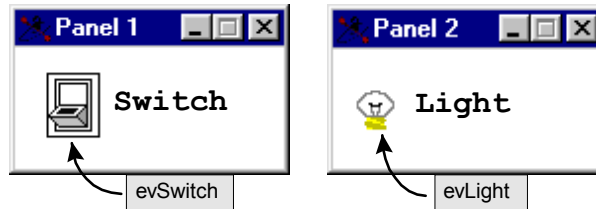






Figure 30: Panels in CANoe

You can start the Panel Editor by activating the  button on the CANoe toolbar. This ensures that the database is available with environment variables *evSwitch* and *evLight* which are necessary for the wiring.


In the Panel Editor first enter the name of the first node's panel **under Options|Panel Size, Name, Colors and Fonts**, e.g. Panel 1. On the Panel Editor's toolbar choose a switch  and place it on the panel. You can wire the switch by double clicking and then assigning the environment variable *evSwitch* to it. You can label the switch by choosing the  display element from the toolbar, placing it on the panel to the left of the switch and then configuring it with a text, e.g. "Switch". You can change the size of the panel by clicking the panel border and dragging it. Try not to size panels any larger than necessary, since available screen space is usually a very limited and hence valuable resource.

Save the panel, e.g. under the name `SWITCH.CNP`. Then create the panel for the second node in the same way. Instead of a switch, here you should insert an indicator lamp as a display element. To do this, choose the bitmap indicator  from the toolbar and then, by double clicking, configure the element as a display element with 2 states. In the configuration dialog you must also enter a bitmap to be used as the indicator. Instead of creating a new bitmap, you might use, for example, the file `LAMP_2.BMP` from CANoe's demo directory `DEMO_CAN_CN\AUTOMOT\BITMAPS\BMP_2`.

Before you save the panel, e.g. under the name `LIGHT.CNP`, you can also label the indicator lamp by inserting and configuring static text to the left of it.

You complete this step of the task by integrating the created panels into the CANoe configuration. To do this, open the panel configuration dialog and choose the menu command **Panel|Configure panels**.

Add the two panel files `SWITCH.CNP` and `LIGHT.CNP` to the list of permanently opened panels and open the panels by activating the button **[Open All Panels]**. Position the panels on the screen according to your work requirements and save these selected panel positions by the menu command **Panel|Save panel positions**.

Before creating the network node models, you should save the configuration you just created by pressing the  button on the toolbar.

1.3.12.3 Creating Network Node Models

You create the network node models in the simulation setup. At the least, the model for the first node must send a message when the switch is activated, and therefore it may not be inserted in the measurement setup.

In the simulation setup click the bus lines to insert new network node models.

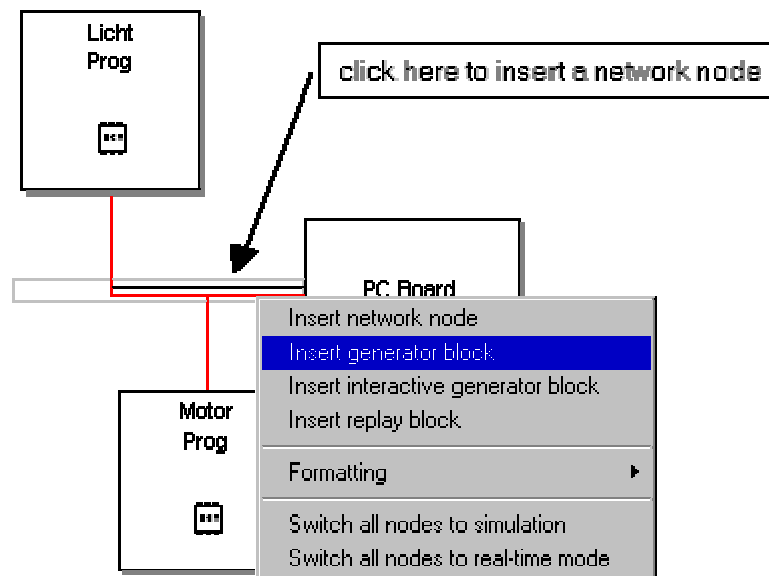


Figure 31: Inserting network nodes in the simulation setup

In this example you need two network nodes in the simulation setup: The first node supplies the switch position, and the second reacts to this by activating or deactivating a small lamp.

You can access the configuration dialog for the two nodes again by pressing the right mouse button. Here you enter the node name (e.g. *ECU 1* or *ECU 2*) and assign a file name to each of the two nodes (e.g. *ECU1.CAN* or *ECU2.CAN*). The node names are shown in the node icons; the file names refer to CAPL programs which simulate the functionalities of the two nodes. Double click on each node to open CAPL Browser for the particular CAPL program.

The first CAPL program belongs to a node at whose periphery there is a switch. When the switch position changes, the program acquires the new switch value and immediately outputs it on the bus:

```
// Reaction to change of environment var. evSwitch
on envVar evSwitch {

    // Declare a CAN message to be transmitted
    message Msg1 msg;

    // Read out the value of the light switch,
    // Assign to the bus signal bsSwitch
```

```
msg.bsSwitch = getValue(this);

// Output message on bus (spontaneous transmission)
output(msg);
}
```

The second network node reacts to this message. The CAPL program reads the value of the bus signal for the switch position and then activates or deactivates the indicator lamp at its periphery. Please note that the switch value is only acquired via the signal value on the bus. The value of the environment variable *evSwitch* is not known to this CAPL program. That is, the communication between the two nodes occurs exclusively via the CAN bus:

```
// Reaction to receipt of the CAN message M1
on message Msg1 {
    // Read out a bus signal and
    // set the environment variable
    putValue(evLight, this.bsSwitch);
}
```

Now start the measurement in CANoe. Whenever you activate the switch on Panel 1 the indicator lamp illuminates. Whenever you turn the switch off, the indicator lamp goes off. The Trace window shows you both the bus communication (Spontaneous transmission of message *Msg1* when the switch position changes) and the values of environment variables *evSwitch* and *evLight*.

Please note how easily and directly this model of a simple distributed system can be modeled in CAPL, and how the database assumes central importance.

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

The communication between two nodes via the CAN bus by means of spontaneous emission, i.e. immediately sending out a CAN message in response to state changes, is not the only capability for modeling the bus behavior of a network node. Cyclic transmission protocols can also be implemented with CAPL language tools such as timers.

1.3.13 Tips for Solving Your Own Tasks

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

Remember that CANoe's measurement setup window and simulation setup represent the data flow plans for your actual measurement or simulation task. Besides associating a database and configuring panels, you can configure all other options directly in these windows: From the data source and bus symbol with network node models in the simulation setup, to the evaluation blocks on the right side of the measurement setup window. You can always access the popup menus of all objects in the measurement and simulation setups 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 CANoe 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 behavior of a controller, you will find a brief introduction to CAPL in section 6.1.4.

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 CANoe:

- With the **CANdb++ Editor** you create or modify databases (*.DBC) which contain the symbolic information for CANoe. This includes network nodes and symbolic names for messages and signals as well as environment variables.
- In the **CAPL Browser** you create CAPL programs for the measurement and simulation setups. Instead of using message identifiers and data bytes, with the help of the database you can also work with message and signal names.
- The **CANoe main program** is used to measure and simulate CAN systems. You can associate one or more databases to any configuration from **File|Database**.
- In the **Panel Editor** you create the control panels that are later loaded in CANoe. Panels represent the I/O interface between the user and the simulated network nodes in CANoe's simulation setup. Besides standard buttons and switches, in the Panel Editor you can also use bitmaps as display and control elements. This involves configuring the bitmap element with the appropriate bitmap file that you can create with any bitmap editor. Each display and control element must be configured with an environment variable from the database so that the display and control elements can be set or read-out by the CAPL models in CANoe. ("Wiring" of the panels to the simulated network nodes)
- The **CAPL Generator** is a tool for automating the generation of network node models that can be used to simulate the remainder of a bus in a CANoe simulation. Generation is based on CAN databases. The network node models are generated as CAPL programs.

The CAPL Generator prepares the database for the generation of panels with the Panel Generator. That is, the necessary environment variables are added to the database, and they are assigned to the proper nodes by means of access rights.
- The **Panel Generator** is a tool for automating the generation of panels which are used for graphic user control and visualization of network node models. Generation is based on CAN databases.

The panels are generated as display and/or control panels in a node-based manner. Environment variables are assigned to the nodes by means of access rights.

Before a panel is generated with the Panel Generator the network node model should be created with the CAPL Generator.

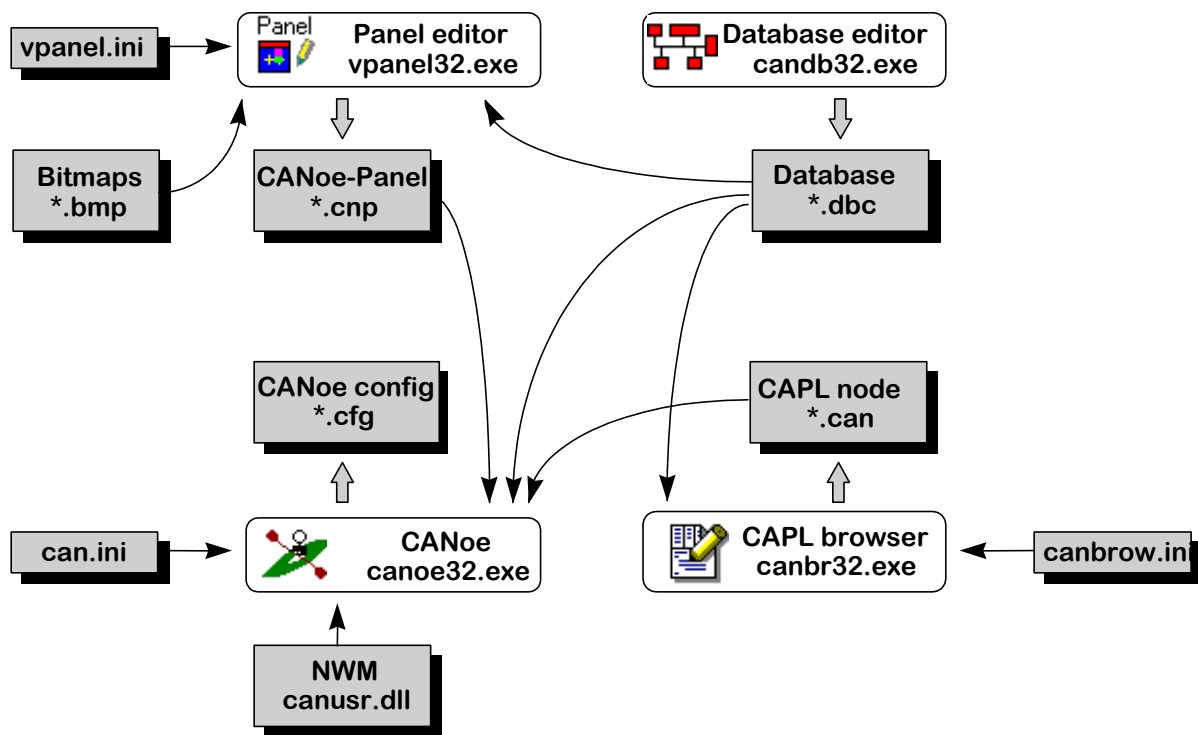


Figure 32: CANoe system overview

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

1.5 CANoe Architecture

In the course of a measurement the PC-card registers CAN messages on the bus and passes them through the simulation setup 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 CANoe 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 (`CANoe32.EXE`) and are evaluated in the function blocks on the right-hand side of the data flow plan.

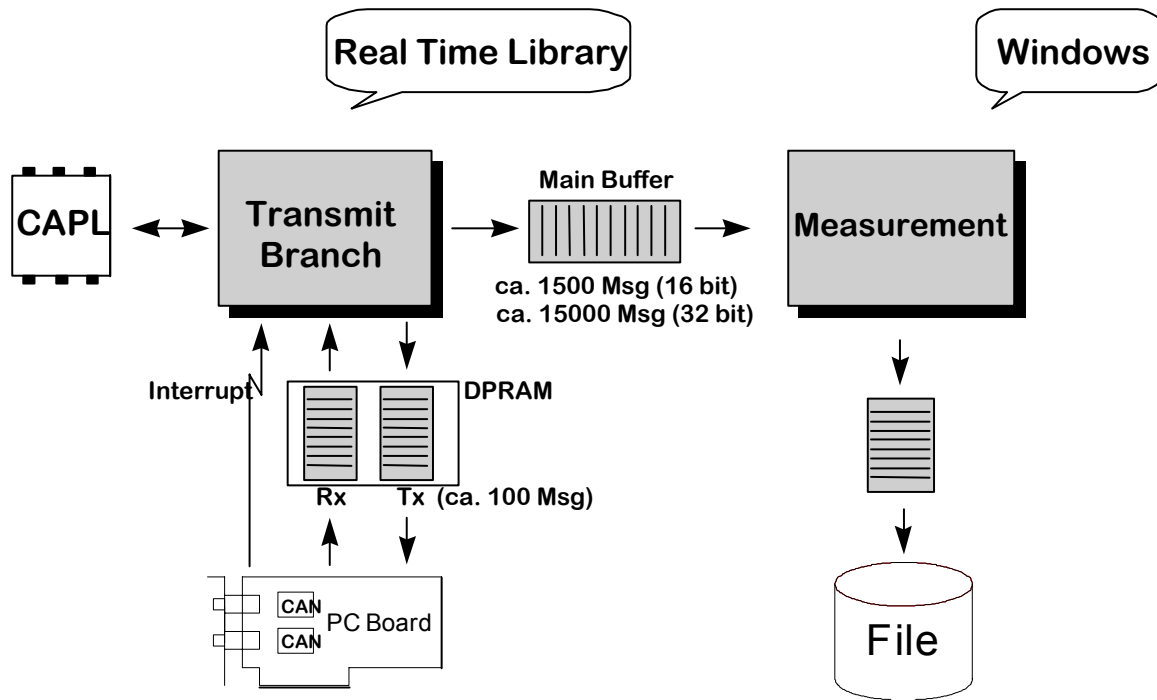


Figure 33: Internal structure of CANoe

You can influence the data flow in the two program modules by inserting function blocks in the simulation setup and/or the measurement setup. The real-time module is comprised of the PC-card block and the simulation setup. The function blocks in the measurement setup will configure the data flow in the main program, with the exception of the real-time library.

If you insert blocks in the real-time library, CANoe's simulation setup, you should 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.

Note: If you overload Windows severely by other programs during a measurement, there may be a delay in reading data out of the ring buffer. Nevertheless, the time stamp for the events, which for example is displayed in the trace window, remains accurate even in this case.

1.6 Particularities of the Demo Version

In the demo version of CANoe 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 accurate time stamps.

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

With the demo version of CANoe you can insert up to a maximum of three network node models in the simulation setup. If you load configurations with more than three simulated network nodes, you will not be able to start the configuration any longer. Aside from these limitations, the demo version is a fully functional version. In particular, messages can be evaluated and saved, and CAPL programming can be tested without limitations.

2 Applications

CANoe 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 and simulation 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 and simulation setup gives you an overview of the configuration options provided by CANoe and shows how your actual measurement configuration appears. The configuration of the simulation is made in the simulation setup window; measurements and analysis are configured in the measurement setup window.

Program Start

At the program start of CANoe the program `CANoe32.EXE` is called by double clicking the appropriate icon in the CANoe program group. CANoe 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).

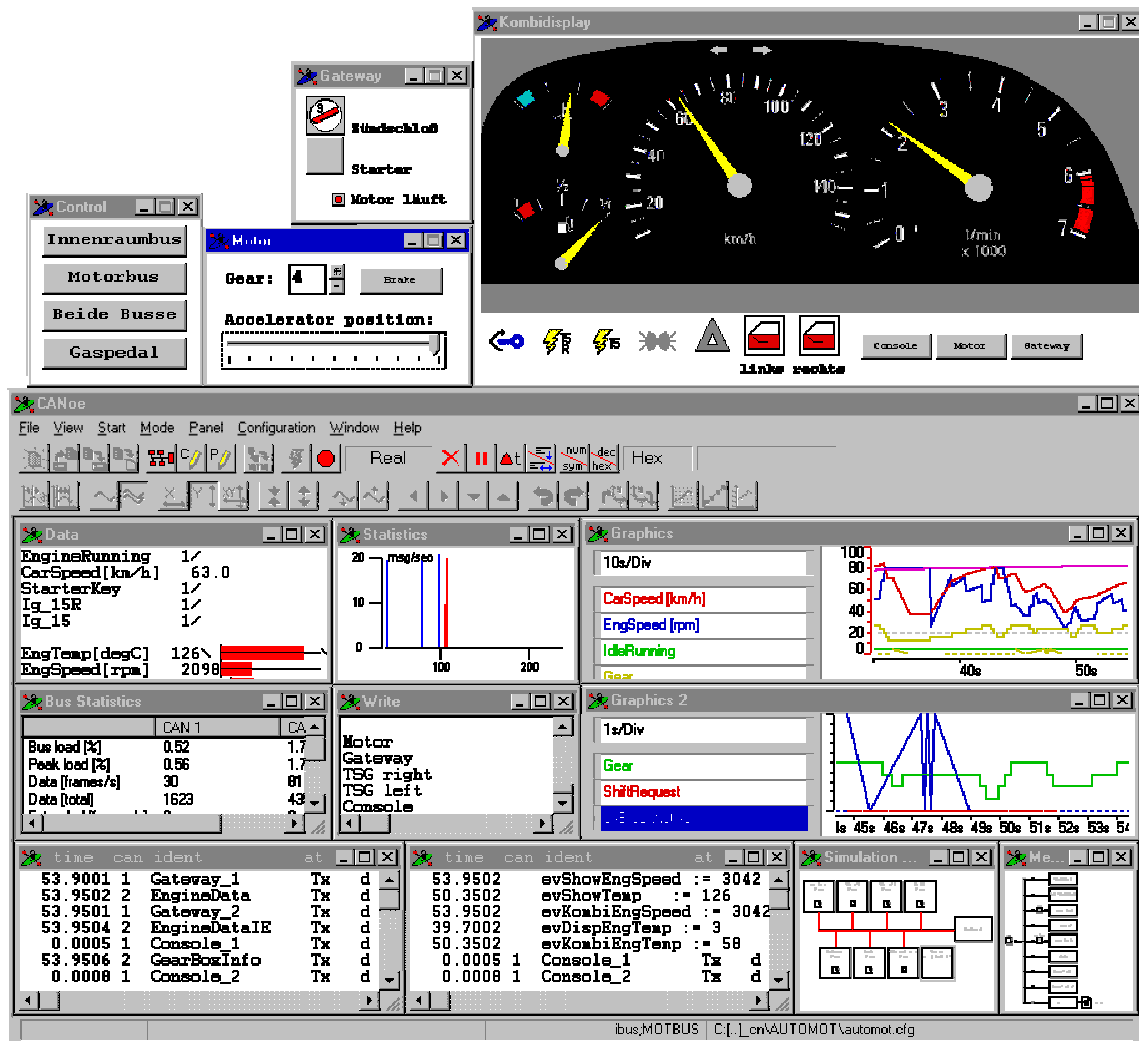


Figure 34: CANoe during a measurement run

At the program start CANoe 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 CANoe, 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 CANoe differently at the start by using multiple icons. If no entries were made in the command line, the last opened configuration is automatically loaded.

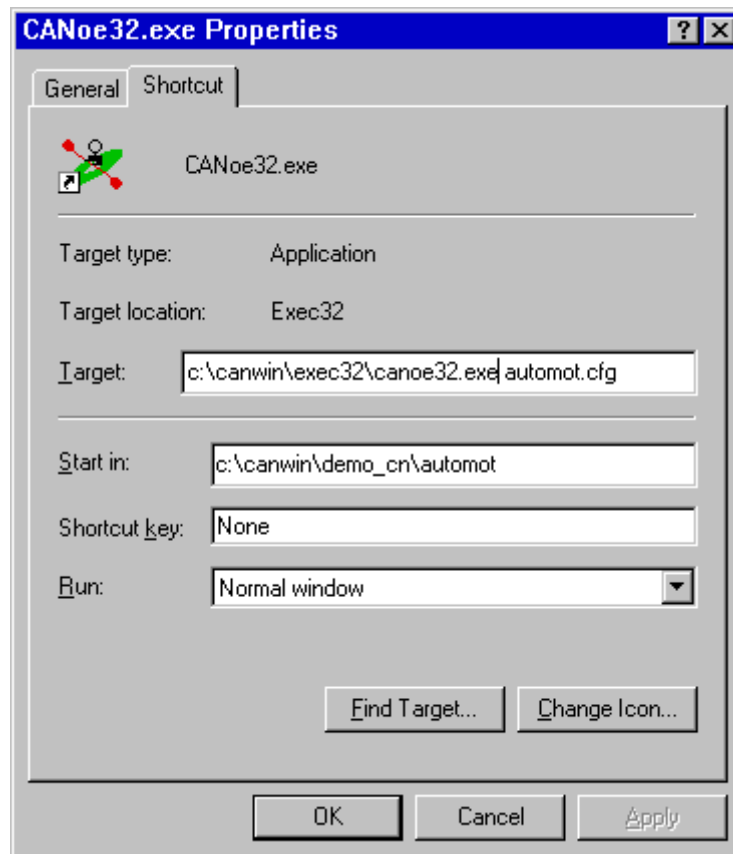


Figure 35: Automatic loading of the configuration AUTOMOT .CFG at program start

The CANoe Screen

The CANoe 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 CANoe windows by double clicking the specific evaluation block in the measurement setup or by selecting the window from the *View* menu.

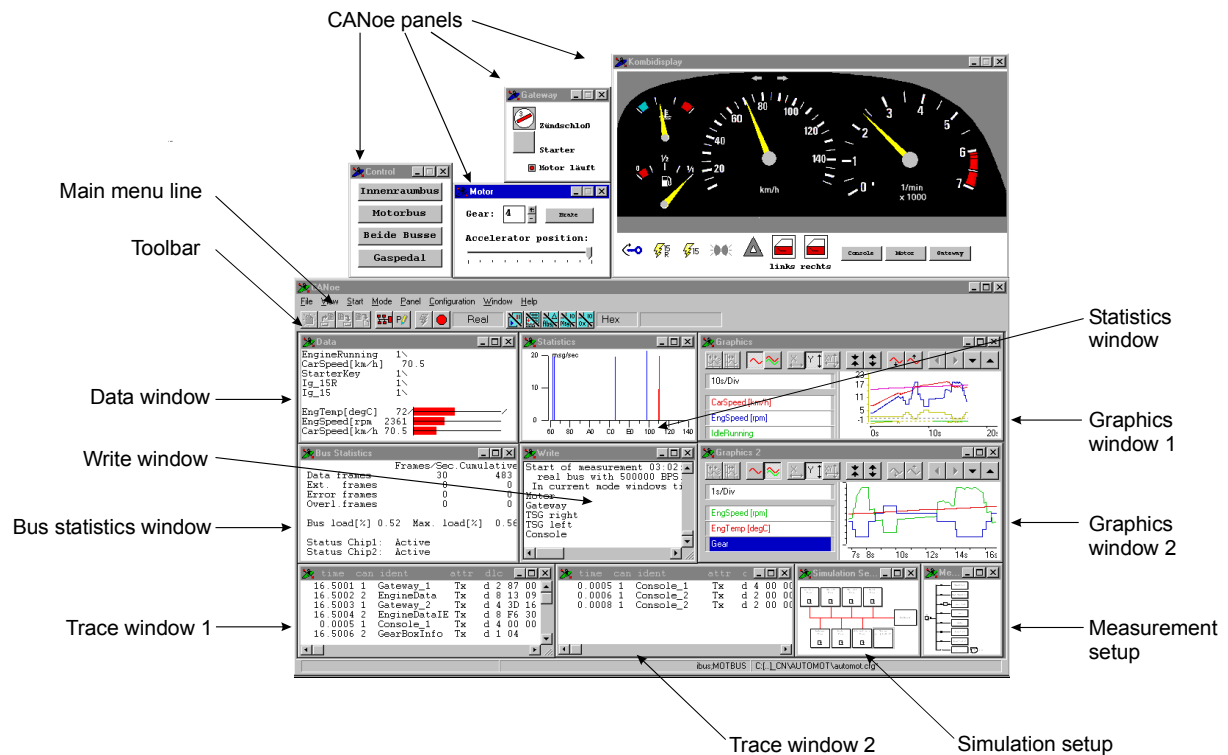


Figure 36: The CANoe screen

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.
Simulation setup and Measurement setup	In the simulation setup window the overall system is setup displayed graphically with the CAN bus and all network nodes. 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 Simulation/ Simulation Setup

2.1.1 Working in the Simulation Setup

Via the context menu (right mouse button) of the simulation setup you have access to standard operations like copy, cut, paste etc. You can also apply these functions on different bus systems. By means of the tabbed pages at the bottom of the simulation setup window, you can easily change between the different buses of your configuration.

Additionally you can easily shift objects. Just select the object with the left mouse button and drag and drop it by pressing the left mouse button (drag-and-drop).

For a simple indication and display of the real and the simulated bus the following colour code applies:

- **real Bus**
...is displayed as a black line.
- **simulated Bus**
... is displayed as a red line.

Additionally a simulation **setup assistant** is provided . This offers support in setups for both simple single-bus systems and complex hierarchical multibus network systems which can be created with relatively little effort.

2.1.2 Gateway

Several nodes can be assigned to one ECU, which makes modelling of gateways possible. A gateway is a special ECU, which is used as connection between two or several buses.

The information exchange between the buses is done via the nodes of a gateway. This can concern several buses (of the same type) as well as a system of different bus types (CAN, LIN,...).

Additionally there is the possibility to set up the interactive generator block (IG) as a gateway (see chapter 4.2.2)

2.1.3 System View

The system view can be displayed on the right in the simulation setup window. For this purpose please use the context menu (right mouse button) of the simulation setup.

The system view shows a tree structure sorted according to buses. It offers a drag-and-drop support, with which you can modify your simulation setup fast and easily. Additionally there is a context menu available for all items of the tree structure, with which you have fast access to important and frequently needed functions. You can, for example

- execute a system verification (context menu at **buses**)
The result is displayed in the write window.
- define a (new) gateway (context menu of a **node**)
- execute a simplified "Copy/Cut-and-Paste" action

2.1.4 Object View

The object view shows you all substantial information about each active object (node, gateway,...). Additionally it is displayed, whether the selected block is active (for the generator block, the interactive generator block (IG) and the replay block).

The object view can be displayed on the right of the simulation setup window. Please use the context menu (right mouse button) of the simulation setup to do so.

2.1.5 System Verification

You can start a system verification by selecting the **System verification** command in the context/popup menu (right mouse button) of the System View. The System verification checks your simulation setup according to the following rules:

- **1st Rule: All network nodes that are defined in a database must be included in the assigned bus.**

Example:

The database "ibus" defines the network "node" console and is assigned to the bus "body" in the simulation setup. To carry out this rule, there must be a CAPL block included in the simulation setup; this CAPL block has to be assigned to the database node "console".

- **2nd Rule: All gateways that are defined in the databases must be included as gateways within the simulation setup.**

Example:

The databases "ibus" and "motbus" both contain the network node "gw" and are assigned to the buses "body" and "power" in the simulation setup. These nodes

are interpreted as gateway definitions. To meet this rule, there must be a CAPL block in the simulation setup that has to be assigned to a database node and has to be included as a gateway.

It does not meet this rule, when a CAPL block is included as a node in one of the buses while no database is assigned to this bus.


- **3rd Rule: All gateways that are defined in the databases must be included as gateways in the assigned buses within the simulation setup.**

Example:

This rule is slightly different to the 2nd rule. It does not meet to include a gateway – that is defined in the databases – in the available buses of the simulation setup. The concerned CAPL block must be included as a gateway in the buses to which the databases are assigned to.


2.2 Measurement/Measurement Setup

2.2.1 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 loading 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.

2.2.2 Working with Configurations

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

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 CANoe. In the demo directory `DEMO_CAN_CN` you will find some prepared demo configurations that can serve as models when you start up CANoe and during the learning phase.

To obtain an overview of the files belonging to your project (configuration files, log files, CAPL programs, databases, Panel files, 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 subdirectories.



Figure 37: Example of a Directory Tree for a CANoe Project

References to other project files (e.g. to database files `*.DBC` or to CAPL programs `*.CAN`) are also saved in the configuration files. CANoe 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 [LastCANoeConfigurations] 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 CANoe versions cannot read the current configuration file format. However, if you still want to work with older CANoe 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), CANoe 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.

2.2.3 Representation Formats

The main menu item **Configuration|Global 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 - whether you wish to have them displayed symbolically.

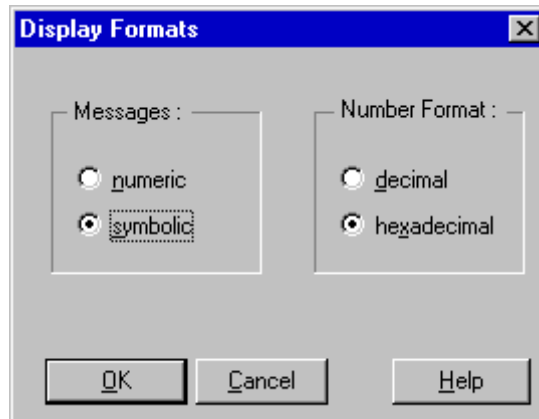


Figure 38: Dialog for configuring representation formats

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.

2.3 Working with 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.

CANoe 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.

In CANoe's symbolic mode you can address CAN messages and data contents by the symbolic names of the associated databases. The sections below offer you instructions which you should observe if you use databases when working with CANoe.

2.3.1 Creating and Modifying Databases

The CANdb++ Editor is available to you for inputting and modifying databases. It is included with the standard CANoe product.

In a database, names are assigned to CAN messages. In CANoe 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*.

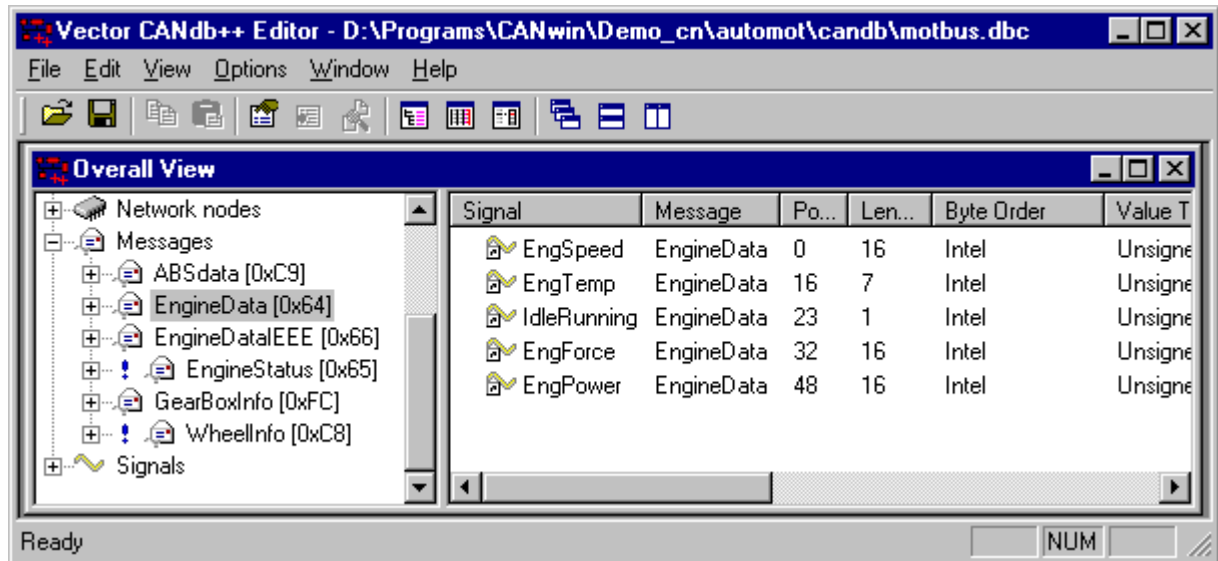


Figure 39: Symbolic description of the message *EngineData* in CANdb++

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 the 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.

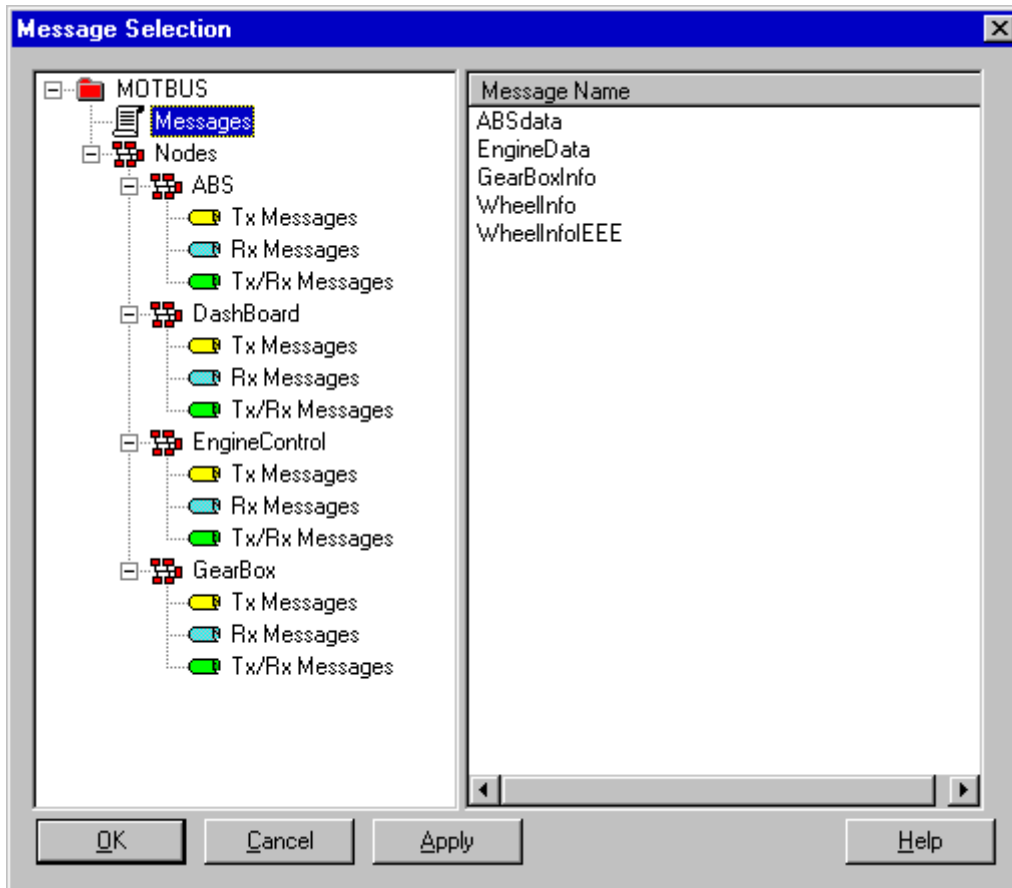


Figure 40: Message and Signal Explorer

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

Please use the simulation setup to edit databases (add, assign, delete, etc.). In the System View window of the Simulation setup you can see a tree view of the current configuration.

If you go to **Databases** with the mouse pointer and click on the right mouse button, you can run the command **Add...** from the context menu. This adds a database (or an additional database) to the current bus.

After you have run the command, the **Open** window appears and you can select a database.

If you click on the **[OK]** button, the new database is accepted for the current bus and is displayed in the System View window.

With the **Edit...** command of the context menu you start the CANdb++ Editor. There you can edit and save the selected database.

The name of the database is determined as follows. If the database includes the at-

tribute 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, as well as environment variables, to several partial databases. When operating CANoe on two buses it also makes sense to describe each system by its own database.

CANoe supports the simultaneous use of multiple databases. You can configure the databases that you would like to associate with CANoe using the menu command **File | Associate database**. Afterwards you can use the symbolic names for messages, signals and environment variables 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 CANoe's online Help function for examples of resolving ambiguities.

2.3.6 Checking for Consistency of Symbolic Data

In CANoe's symbolic mode the CAN messages are addressed using symbolic names from an associated database. Therefore, CANoe 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 CANoe 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 CANoe 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

CANoe 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 2.11.1).

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 with a real bus (menu item **Configure | Simulation...**) messages from the simulation setup 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, CANoe 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 in the measurement setup to an undefined channel, the transmit request is similarly passed through.
- An error is reported in the Write window for a sender in the simulation setup 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 Simulation Mode

In CANoe's Simulation mode (Online mode, **Simulated bus** option in the menu **Configure | Simulation...**) buses and network nodes are fully simulated.

Each channel corresponds to a simulated bus. Set up your channel definition according to how many buses you wish to simulate.

If you use channels that are not defined, CANoe behaves as follows in Simulation mode:

- The channel configuration does not cause any filtering of messages in the data flow plan.
- When transmitting from a Generator block or Replay block in the measurement setup to an undefined channel, the messages are passed through.
- When transmitting from a Generator block or Replay block in the simulation setup to an undefined channel, an error is reported in the Write window as soon as the transmit request is given to the undefined channel. Afterwards, the message is not transmitted.
- CAPL blocks do not transmit messages to which an undefined channel is assigned.

2.4.4 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 CANoe 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 in the measurement setup 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 CANoe in Load and Overload Operation

This chapter helps you to operate CANoe steady under more difficult conditions and points out the limits under high- and overload behavior.

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 CANoe cannot follow along with processing any longer, the ring buffer between the real-time library and the CANoe- application runs full. CANoe detects this when a **High water mark** limit is exceeded (cf. Figure 41), 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".

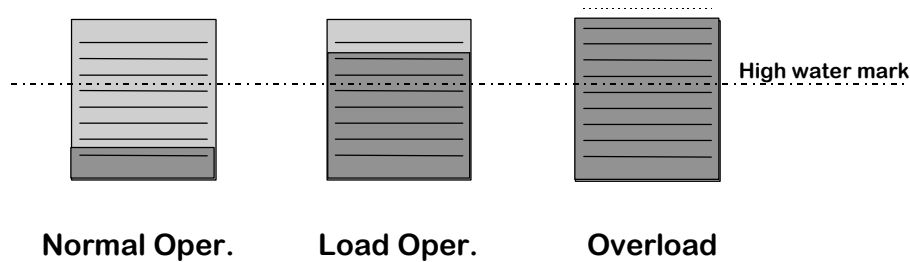


Figure 41: Fill Level of the Ring Buffer

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 CANoe for high bus loads, a performance wizard is provided under the menu items **Help | Performance | Optimization** and **Help | Performance | Optimization 2**. 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 CANoe 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. You may need to 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 CANoe's simulation setup, 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.

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 CANoe.

It is not possible to predict an optimal configuration of the measurement and simulation setups 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 Working with Panels and Environment Variables

To permit working with simulated network nodes in CANoe, the network node models in the simulation setup created in CAPL must be able to react to external events (e.g. activation of a switch). CANoe also provides you with the option of creating your own user control interfaces (Panels) and integrating them into the program.

This involves describing external events with the help of environment variables, whose names and types (discrete/continuous/character string) you define in the database.

CANoe differentiates between discrete, continuous and character string variables. Switch positions can be represented by discrete environment variables; continuous environment variables can be used to describe variables such as temperature or engine speed, while character string variables can be used to represent whole words up to a length of 24 characters.

You can interactively change the values of these environment variables during a measurement by activating the controls on the panels. The network node models react to changes in environment variable values and then execute the appropriate actions (e.g. sending out a message).

In the other direction, CAPL programs can also change the values of associated environment variables when certain events occur. This value change can then be visualized on the panel using display elements.

Therefore, environment variables can also be interpreted as I/O interfaces between network nodes and their peripherals, i.e. the "wiring" between the particular CAPL program and its input and output elements on the panels.

The tool for creating panels, the Panel Editor, is described in detail in chapter 5.

2.6.1 Assigning and Positioning Panels

You can assign panels to your active configuration from the main menu item **Panel**.

After selecting an upper entry, a dialog is opened in which you can link panels in CANoe. This dialog is described in section 2.6.2.

You would choose the menu item **Save panel positions** to permanently save - in the CANoe configuration - all of the loaded panels that you have arranged on the screen according to your working requirements. Then the panels will appear at the same positions the next time the program is started.

2.6.2 Panel Configuration Dialog

Choose the menu item **Panel | Configure Panels** to configure the control and display panels and panel control you wish to work with in CANoe. In the panel configuration

dialog you edit three lists which you select by activating one of the following three option buttons:

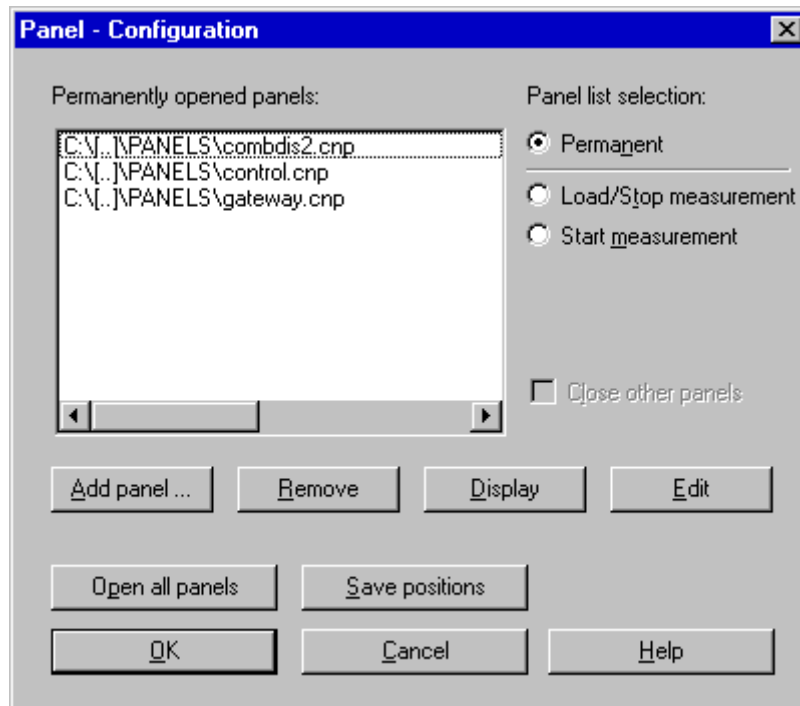


Figure 42: Panel configuration dialog

Permanently Opened Panels:

In this list you enter all panels of the configuration which should remain permanently opened, such as main panels which must always be available or panels for panel control. The panels in this list are available to you as long as CANoe is loaded. In particular, they are not affected by panel control actions.

Permanently opened panels which were closed explicitly are opened again by panel control.

Panels to be Opened After Loading/At Measurement Stop:

In this list you enter all panels which should be opened when the program is loaded or when the measurement is stopped. You will need the panels of this list if you wish to set the values of certain environment variables before and after the measurement, to prepare for the next measurement. Please note that the panels in this list are closed by activating panel control pushbuttons.

With the check box Close other panels you can set whether all those panels not entered in this list should be closed at the stop of measurement. Permanently opened panels always remain open, independent of the status of this check box.

Panels to be Opened at Measurement Start:

In this list you enter those panels which should be opened at the start of measurement. These panels are then available to you during the measurement. Please note that panels entered in this list are closed when Panel control buttons are activated.

With the check box Close other panels you can set whether all those panels not entered in this list should be closed at the start of measurement. Permanently opened panels always remain open, independent of the status of this check box.

The selected list is configured using the buttons beneath the list:

Pressing the **[Add ...]** button causes a file selection dialog to appear, in which you can select a panel. After exiting the dialog with **[OK]** the panel is inserted in the active panel list. With **[Remove]** you can remove the highlighted panel from the list again. The panel configuration file itself is not deleted by this action. To display the panel on the screen you would press the **[Display]** button.

To edit panels select one or more panels from the active list and press **[Edit]**. This opens the Panel Editor with the selected panels. If you have already edited a loaded panel with the Panel Editor, after saving it you must press the **[Display]** button to make the changes effective in CANoe.

To copy panels from one list to another, or to move a panel, open the popup menu in the list box by clicking with the right mouse button. With **Copy** you can copy all selected panels to the Clipboard. The **Cut** function removes the selected panels from the list. With **Paste** you can insert panels which you previously copied to the Clipboard into the active list.

For each of the two lists for the events **Loading/Measurement stop** or **Measurement start** you can indicate whether all those panels not in the list should be closed when the particular event occurs. To do this you would check the option box **Close other panels**. The default setting (option box not checked) leaves these panels opened. Please note that permanent panels remain open independent of the check box.

In the measurement start list you can enter those panels which you only need during the ongoing measurement. You will need the list of panels opened during loading or measurement stop if you wish to set certain environment variables before and after the measurement, in order to prepare for the next measurement. Please note that the panel groups assigned to these two events are closed by other panel control actions. Therefore, a panel which - for example - should remain open for the entire duration of the measurement must be entered in the configuration lists of all panel control push-buttons which can be operated during the measurement.

You can permanently save the panel positions. To do this, first open all panels by activating the button **[Open all panels]** and then arrange the panels on the screen according to your work requirements. When the **[Save positions]** button is pressed, the active positions of all opened panels are saved in the CANoe configuration.

Note: Simultaneously pressing the <Shift> key and moving the mouse pointer over an element in an opened panel causes a window to appear below the element, in which the name of the corresponding environment variable is shown. If you have entered a comment for the environment variable in the CAN database, the first comment line of the environment variable appears next to the name.

2.6.3 Initialization of Environment Variables

You can configure options for initialization of environment variable values in the simulation dialog by selecting the **Configuration | Simulation....** menu item.

All environment variable values are preserved after the end of measurement, and if the option **Do not change environment variable values** was set, they are assumed as start values at the start of the next measurement. With this option you can also intentionally assign start values to environment variables before the measurement start. To do this, configure the appropriate control element on the panel loaded in CANoe and then start the measurement.

If you activate the option **Reset environment variable values before measurement start to CANdb values**, all environment variables are reset to their start values at the beginning of a new measurement. To manually reset the environment variables to their start values in the database before the start of a new measurement, press the **[Reset Now]** button.

The function `CallAllOnEnvVar()` is provided in CAPL for the network node models in the simulation setup to permit reaction to the initialization of environment variables at the measurement start.

2.6.4 Panel Control

For complex models with many input and display panels, it is often impossible to arrange all of them on the screen simultaneously. On the one hand, the screen does not provide enough space; and on the other hand, system resources are inadequate to open an arbitrary number of panels simultaneously.

For measurements and simulation runs, however, usually only a small subset of all panels is needed at the same time. Therefore, CANoe panel control provides you with the option of grouping panels according to your work requirements. During the measurement you can switch back and forth between these panel groups, so that only one panel group is open at any particular time.

For this purpose, CANoe offers you two possibilities:

Creating Panel Control Buttons with the Panel Editor

With the Panel Editor you can create control panels with special elements, the panel control buttons. Each panel control pushbutton is configured with a list of the panels which should be opened when the pushbutton is activated. During a measurement you can then selectively open the control panels of particular panel groups, simply by pressing the appropriate panel control pushbutton.

Furthermore, the panel configuration dialog also offers you the option of selecting certain panels which should remain permanently open, i.e. should not be influenced by the actions of panel control. Simply enter the particular panels in the list of permanently opened panels. On the one hand, these may include the control panels themselves, which should not be closed. Of course, on the other hand you can also enter important display and control panels here which must constantly remain open.

In the panel configuration dialog you can also enter panel groups for the two predefined events **Measurement start** and **Loading/Measurement stop**. Panel control opens the particular panel when the event occurs, just as though you had pressed the particular configured panel control pushbutton. Here you can also decide whether opened panels which do not belong to these events should be closed or not.

Panel Control in CANoe

With **Panel Control** under the main menu item **Panel** you can arrange and position any panel group you want and save them under a defined name.

In this dialog **Panel Control** you can also open and close the defined panel groups and single panels.

Note, that only one panel group and/or any single panel can be opened at a time.

2.6.5 Configuring the Panel Control for Small Models

When small models are set up, it is often unnecessary to implement the mechanisms of panel control. Here it is advisable to first enter all necessary panels into the list of permanently opened panels. You should always use the list of panels to be opened after the start of measurement if there are certain panels which you only wish to use during the measurement. You can decide in the configuration dialog whether these panels should automatically be closed again at the end of a measurement.

2.7 Logging and Evaluation of Measurement Files

CANoe 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 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.

CANoe 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 measure-

ment and evaluation functions of Online mode are also available to you in Offline mode.

2.7.1 Logging Triggers

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.

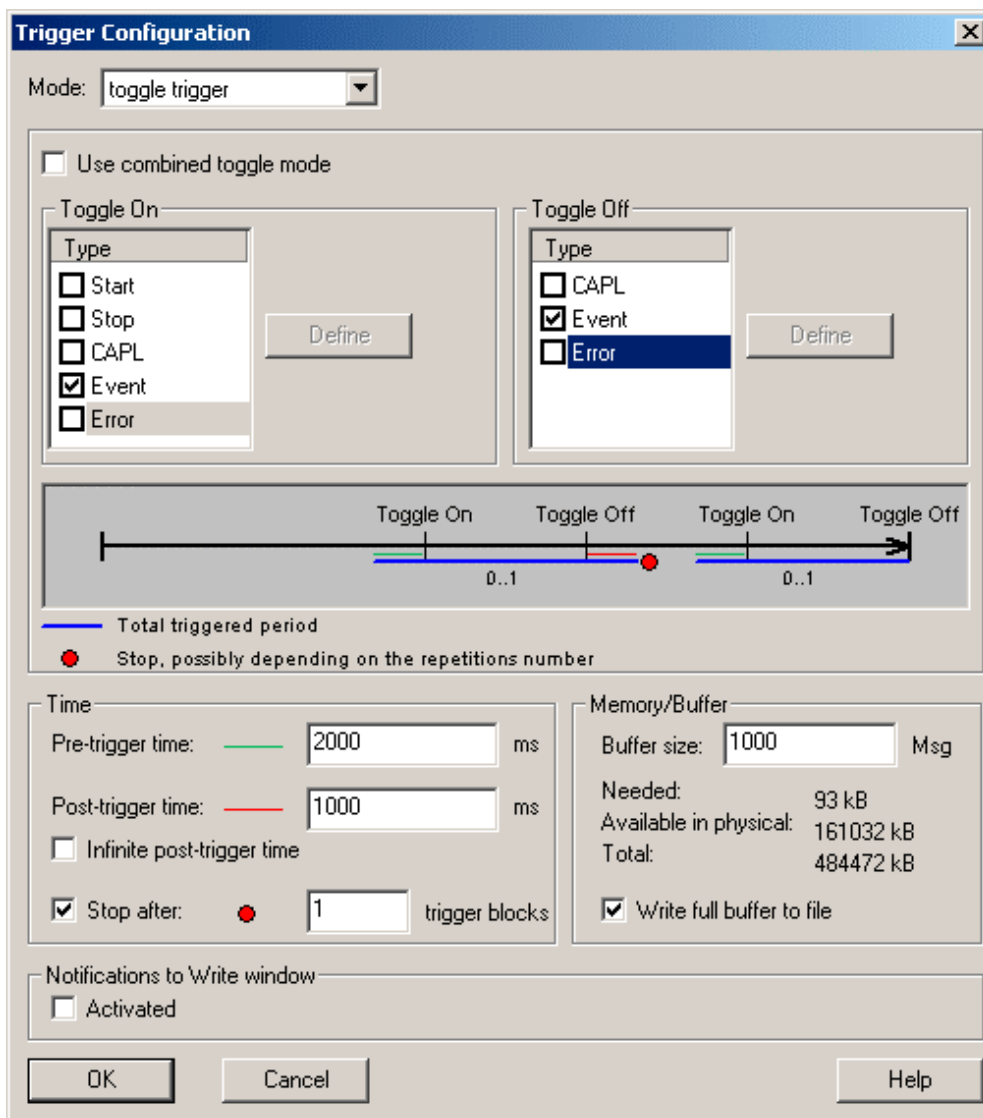


Figure 43: Trigger Configuration Dialog

2.7.1.1 Trigger Modes

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

- Entire Measurement
- Single Trigger
- Toggle Trigger

In the **Entire Measurement** trigger mode the entire measurement is logged. Pre-trigger and post-trigger times cannot be selected. Activating the button **Write full buffer to file** causes a full data buffer to be saved intermediately for long measurements. You can avoid data loss in this manner.

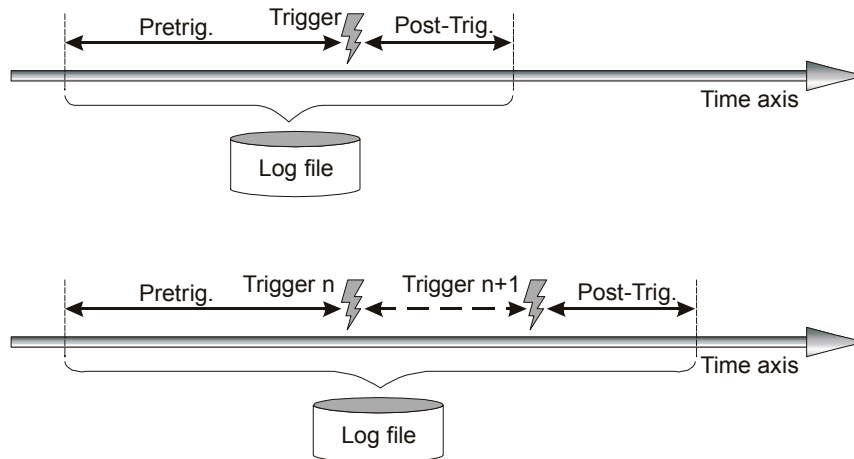


Figure 44: Time Window for the Trigger in Single Trigger Mode (top) and in Toggle-Trigger Mode (Bottom)

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** section.

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.

```

Logging Header {
  date Tue Oct 27 13:14:46 1998
  base hex
  internal events logged
  Begin Triggerblock Tue Oct 27 13:14:52 1998
    6.0004 1 WheelInfo Tx d 8 69 0F 00 00 00 00 15
    6.0151 1 ABSdata Tx d 3 A1 00 00
    6.0501 1 GearBoxInfo Tx d 1 04
    ⚡ ← 6.0501 log trigger event
    6.0651 1 ABSdata Tx d 3 4E 00 00
    6.1004 1 WheelInfo Tx d 8 EF 04 00 00 00 00 08
    6.1151 1 ABSdata Tx d 3 4E 00 00
    6.1181 1 EngineData Tx d 4 68 5B A3 00
  End Triggerblock
  Begin Triggerblock Tue Oct 27 13:14:55 1998
    9.1004 1 WheelInfo Tx d 8 B5 06 00 00 00 00 32
    9.1151 1 ABSdata Tx d 3 62 00 00
    ⚡ ← 9.1300 1 ErrorFrame
    9.1300 log trigger event
    9.1651 1 ABSdata Tx d 3 65 00 00
    9.1771 1 EngineData Tx d 4 98 76 28 00
    9.2004 1 WheelInfo Tx d 8 73 09 00 00 00 00 72
    9.2151 1 ABSdata Tx d 3 68 00 00
  End Triggerblock

```

Figure 45: 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.

2.7.1.2 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. 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.

2.7.1.3 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.

2.7.1.4 Configuration of the Logging Buffer

CANoe 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 nth 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.

2.7.2 Log Files

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

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

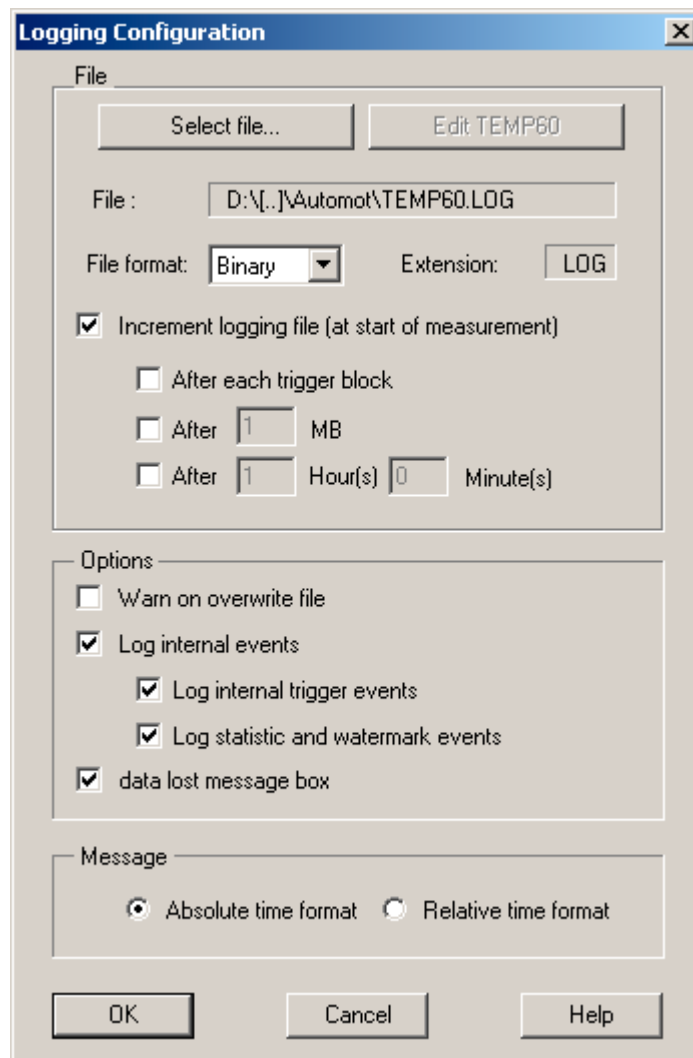


Figure 46: Configuration of the log file

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 gener-

ates 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 `CANoe.LOG` or `CANoe.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()`.

2.7.3 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 file icon's configuration dialog the internal events generated by the program (e.g. bus statistics information, triggers, changes to environment variables 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	

Event type	Format in ASCII Logging	Binary	Comments
Error Frame	<Time> <CAN> ErrorFrame	Yes	
Overload-Frame	<Time> <CAN> Overloadframe	Yes	
CAN error	<Time> CAN <CAN> Error: <Error message>	Yes	Internal event
Environment variable	<Time> <Variable name> := <Value>	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
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

2.7.4 Data Analysis in Offline Mode

To study recorded log files switch CANoe to Offline mode with the main menu item **Mode | To Offline**.

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 function, with which you can intentionally stop the replay of the log file. This is described in section 2.7. 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 file icon's popup menu on the left side of the offline measurement setup. CANoe 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 2.7.5. If the tools provided in the configuration dialog are inadequate, the CAPL language – with the function `stop()` – allows you to program a breakpoint yourself.

2.7.4.1 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**.

2.7.4.2 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 CANoe 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. The time basis in Online mode is real time, and clocks in CANoe 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.

Offline

Toggles to Offline mode. In Offline mode a data flow plan is shown in the measurement setup window, in which a file 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 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.

2.7.5 Trigger and Search Conditions

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

2.7.5.1 Condition Primitives

A condition is composed of condition primitives that can be combined with logical **or** or **and**.

Example:

1.Primitive: ID > 100

2.Primitive: D0 == 0

A logical **OR** combination results in all those messages being found whose IDs are greater than 100 or whose first data byte is 0. For a logical **AND** combination all those messages are found whose IDs are greater than 100 and whose first data byte is 0.

The **[Define]** and **[New Prim.]** buttons are provided (only when a symbolic database is used) for entering a new primitive.

The dialogs for entering and modifying primitives differ, depending on whether you are working with a symbolic database (**[New Prim.]** button) or not (**[Define]** button). The desired attributes can be input in the upper line. The identifier and data bytes are input below this. Furthermore, a time window can be specified within which the search is to take place.

2.7.5.2 Entry or Change of Primitives (without Database)

The same rules apply to both identifiers and data bytes. There is a toggle switch (*Relation*) in the column after the object name, which defines the desired relation. If it is empty, the object is not considered. If a relation is to be made, the user selects one of the following symbols:

Symbol	Meaning
==	equal to
!=	not equal to
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to

After this is a mask to be executed, and the user must choose the numbering format to be used:

Number format: Decimal

The desired value is entered here as a decimal number. To find the identifier 100 the following is entered:

ID == 100

The entry is illegal if there are still X (undefined) or \$ (invalid) characters in the input box.

Number format: Binary

A mask is entered here which specifies bits to be set, bits not to be set, and bits that are masked out, i.e. should not be considered. The characters have the following meaning:

X	Masked out
1	Bit is set
0	Bit is not set

To find all messages whose first data byte is odd, the following must be entered:

D0 == XXXXXXX1

Note: It is possible to use >, < etc. in binary mode, but this is difficult to interpret in mask conditions, and is only of interest to users who have a knowledge of precise machine-internal data representation.

Number format: Hexadecimal

Here also masks can be specified. Integers can be entered in hexadecimal format, or nibbles (half bytes) can be masked out by X. The character \$ represents invalid. For example, it can result in the following case:

XXXXXXX1 is entered in binary mode, then a switch is made to hex mode. The four X's on the left are converted to an X as a nibble. The lower values XXX1 cannot be interpreted as a completely defined nibble. Therefore, the result is X\$.

2.7.5.3 Entry or Change of Primitives (with Database)

When a symbolic database is used it is possible to trigger to messages and signals.

The **SYM** buttons **Message name** and **Signal name** after the text input boxes are used to call the symbolic selection dialog, in which you can select the message - and if applicable also the signal from the database - to which triggering should occur.

If you only enter a message name in the symbolic selection dialog, logging is triggered whenever the message occurs on the bus.

If you select a signal name in the input box below this, a decision is made based on the signal value regarding when logging should be triggered. The comparison of signal values is made based on their physical interpretations. This involves choosing a comparison operator and entering a numeric value.

Physical dimensions are stored in discrete form in the CAN messages. Consequently, the specified numeric value may not always be capable of being mapped to a discrete value. In such cases, after pressing the Enter key or the **[OK]** button the two next possible physical values are displayed in two boxes below your entry.

Example:

The signal *Enginedata.rpm* is defined as a 16-bit unsigned value with an offset of 0 and a factor of 10. If 1015 is input as the comparison value, the raw value would have to be $1015/10 = 101.5$. Since only discrete values may occur, either 101 or 102 is used, which corresponds to physical values of 1010 or 1020, respectively. It is these two values that appear in the two information boxes.

In spite of discrete saving, valid values may still have digits after the decimal point. If a factor of 10.5 were used in the above example, then 1008 and 1018.5 would be recommended as possible values.

You can also enter a time range for each primitive, within which it should be active. Outside of this time range the condition would never be satisfied.

2.7.6 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 is opened by choosing the **Export** menu item in the popup menu of the File icon to the right of the Logging block.

2.7.6.1 Export

The export can be limited to specific signals. To do this, the desired signals must be selected 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** field those programs that can be started after an export.

2.7.6.2 Conversion

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

2.7.7 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 CANoe 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 CANoe

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

2.8 COM-Server

A COM-Server is implemented in addition to the DDE-services. It 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.

2.9 Troubleshooting

See section 2.10 for a list of error messages occurring during initialization or during a measurement. In this section tips are offered for handling problems of a general nature.

CANoe 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`.

CANoe runs too slowly

Power managers, which are particularly common on notebook computers, may not be installed for CANoe 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 installation guide of your hardware.

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, CANoe cannot establish a connection to the CAN hardware. Check the installation of the CAN card. Please compare the instructions in the installation guide of your hardware.

Above all, notebook PCs frequently use a power manager. This **must** be deactivated! (please compare the instructions in the installation guide of your hardware).

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 2.11 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 CANoe. To better localize and correct system errors that occur very seldom, CANoe 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.

2.10 List of Error Messages to the CAN Interface

Error messages related to communication between CANoe 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 CANoe 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 CANoe 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 jumpered over on the card, this must be communicated to CANoe 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)

CANoe card driver and hardware are not compatible.

Remedy: The correct CANoe 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.

Faulty ID, DLC etc. (1544)

Can neither appear at the CANalyzer Window nor at CANoe.

Exception: CANIB. Description see 1128

FIFO entry>16 (114)

CANIB has reported invalid data to CANoe. 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 CANoe 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 appears 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)

CANoe 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 (1080)

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:

- CANoe 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.

Übertragungsfehler (1352)

Kann weder bei CANalyzer noch bei CANoe auftreten.

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.

2.11 The Interface to the Hardware

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

CANoe 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 CANoe 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 left side of the simulation setup, 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

¹ The Offline mode for the analysis of log files is described in section 2.7.4.

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.

2.11.1 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**.

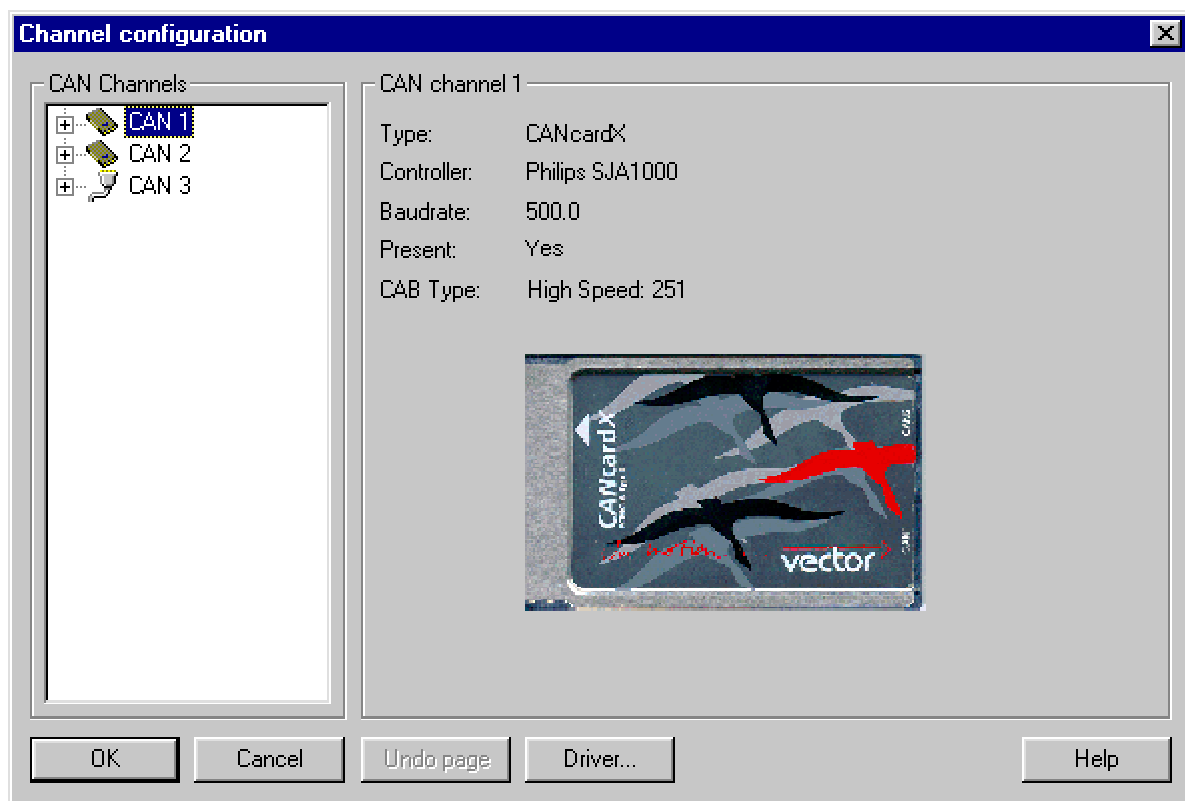


Figure 47: Configuration of Hardware

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 2.11.2). 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 2.11.3). Finally, with the **Options** item you obtain a sub-dialog for configuring driver and card options (cf. section 2.11.4).

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.

2.11.2 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:

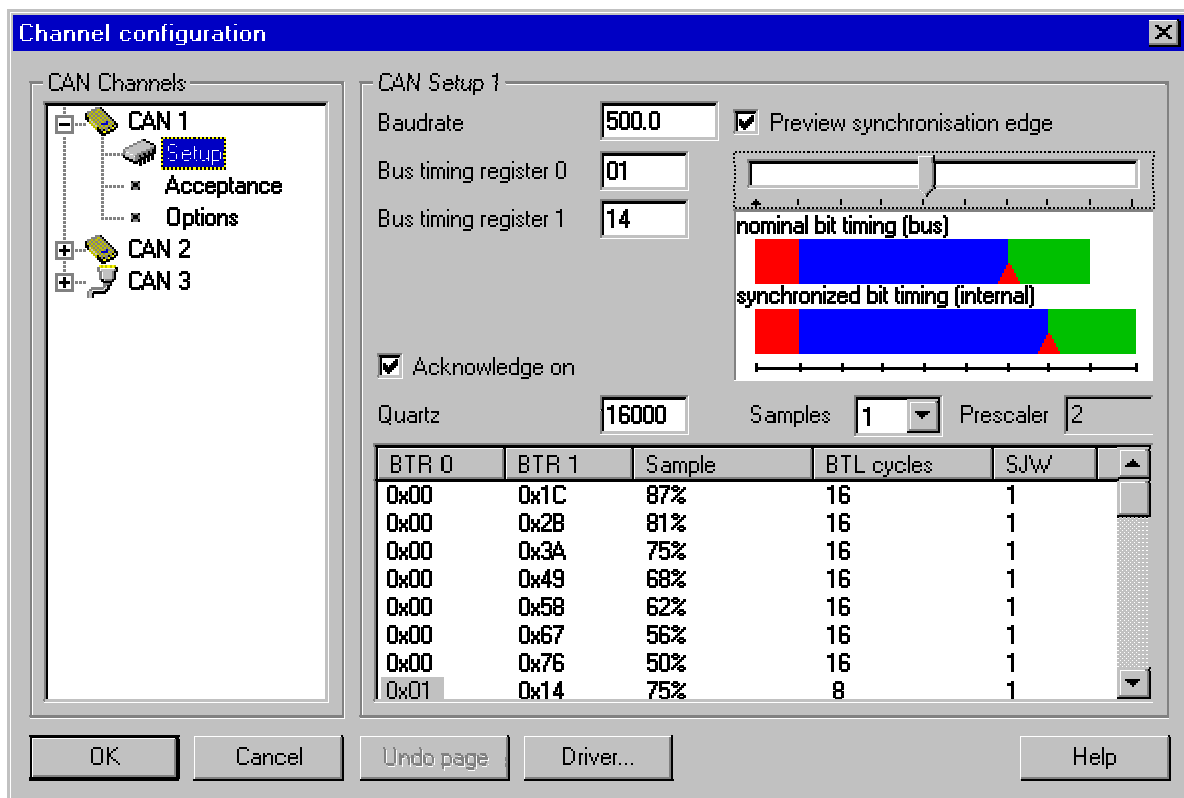


Figure 48: Bus Parameters Dialog

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 jump width (*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 output control register on 0x02 (default is FA) with 82527 controller, interrupt the TX line (jumper 9 must be removed on PB1 resp. PB2)
--------------	---

2.11.3 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.

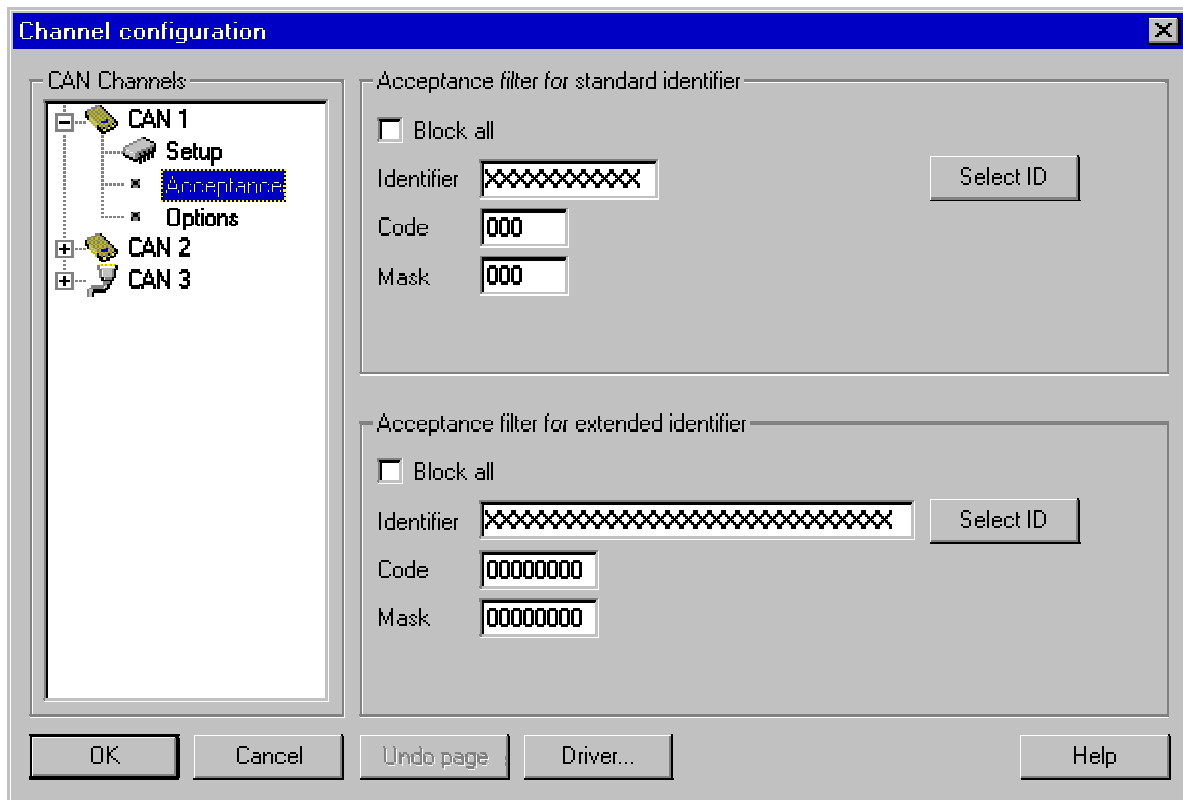


Figure 49: Acceptance Filtering with the SJA1000 Controller

Instead of entering the mask and code directly, you may program acceptance filtering in CANoe 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.

2.11.4 Card and Driver Options

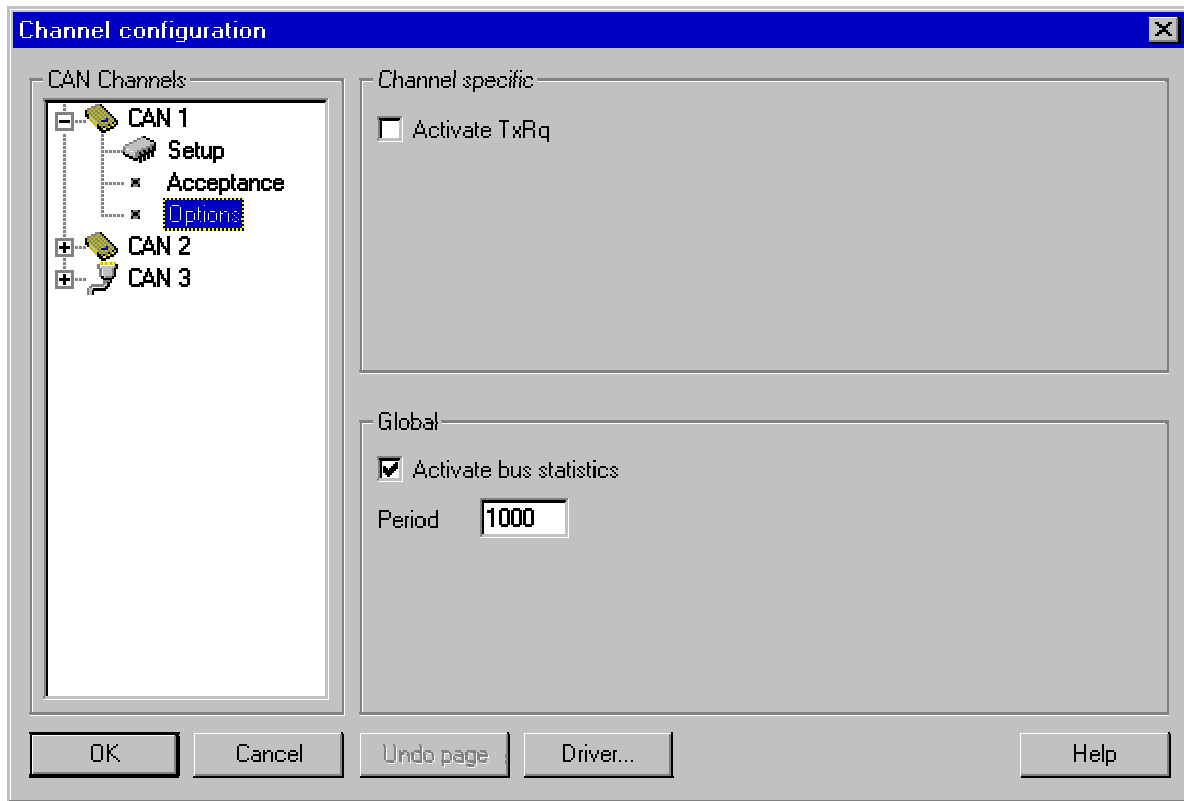


Figure 50: Card and Driver Options Dialog

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.
- Choosing whether bus statistics should be displayed, as well as their
- display refresh rate.

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

You can modify the arrangement of windows (Window layout) from the *Window* main menu:

With **Standard Layout** the standard window layout is loaded. It makes sense to call this function if you have changed the window sizes and positions and wish to quickly return to the standard configuration. In contrast to the function **File | New configuration** this function only acts on the window layout. All other configuration settings are preserved.

Select the function **Set user-defined layout** to define a window arrangement (Layout) according to your working requirements. This window layout is saved in the file `CAN.INI`. With the menu item **User-defined layout** you can finally arrange the windows of each opened configuration according to the settings made with this function.

In the lower menu area all recently opened windows of CANoe are shown. Additionally, the active window is marked with a .

For Trace windows and the Write window, the window's font can be set from the window's popup menu with the function **Font | Select**. The fonts of function blocks and function block comments can be configured in the measurement setup.

The predefined font can be set for a window with the function **Font | Reset**.

3.1 Simulation Setup

The overall system with the busses (Multibus System) and all network nodes is displayed graphically in the simulation setup window. This figure corresponds to a representation of the phases 2 and 3 described in section 1.1. The simulated bus of phase 2 is represented by a red horizontal line. The black line above it symbolizes the physical bus of phases 2 and 3. The two busses (simulated and real) are connected to one another through the PC-card. The card can be parameterized (baud rate selection, acceptance filtering, etc.) using the main menu or the card icon's popup menu. The popup menu can be accessed by clicking the card icon with the right mouse button or by selecting the card with the cursor keys and then pressing the <F10> key.

3.1.1 Configuration of the Simulation Setup

The block for the PC-card can be seen on the right side of the simulation setup. Section 2.11 describes how you can configure the card using this icon. The simulated bus (displayed as a red line) and the real bus (displayed as a black line) are connected to the card block.

If you click the bus symbol with the right mouse button you get the popup menu for the two busses (simulated and real). By repeated selection of the menu command **Insert network node** in the popup menu of the bus image you can insert as many network nodes as you wish in the simulation setup and connect them to the simulated or real bus. The status of all nodes can be switched over simultaneously using one of the two menu commands **Switch all nodes to real-time mode** or **Switch all nodes to simulation**.

Simulated network nodes are shown with red connection lines to the simulated bus in the simulation setup. During a measurement their functionality is simulated by the assigned CAPL program. In this case, the network node's bus behavior does not differ from that of a real controller with the same functionality. In contrast, real nodes are shown with black connection lines to the real bus. The network node models of real nodes do not have any effects on the measurement.

To switch over the status of the currently active node in the simulation setup simply press the spacebar. Each time the spacebar is pressed the node status changes from *simulated* to *real* to *off* and then back to *simulated*.²

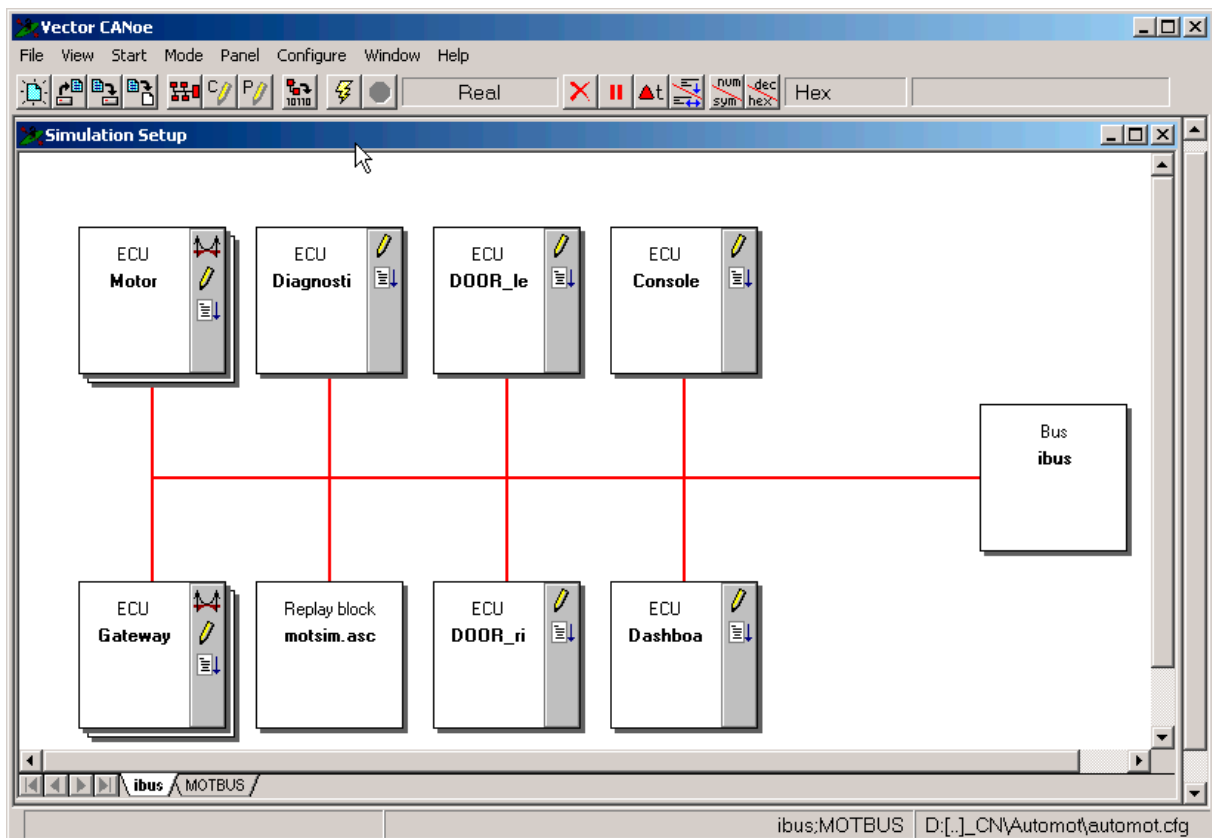


Figure 51: CANoe Simulation Setup

Besides network nodes you can also insert generator blocks and Replay blocks in the simulation setup. To remove a network node from the simulation setup choose the command **Delete this Node** in the node's popup menu, or select the node in the simulation setup using the cursor keys and then press the button. The assigned CAPL program is not deleted by this action.

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

² In this version of CANoe the option *real* has the same effects as the option *off*.

3.1.2 Layout of the Simulation Setup

With the context menu item "Group Blocks" of the simulation setup you induce CANoe to arrange the node symbols separately in different bus threads depending on their function (generators, CAPL blocks, ...)

3.2 Measurement Setup Window

3.2.1 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.

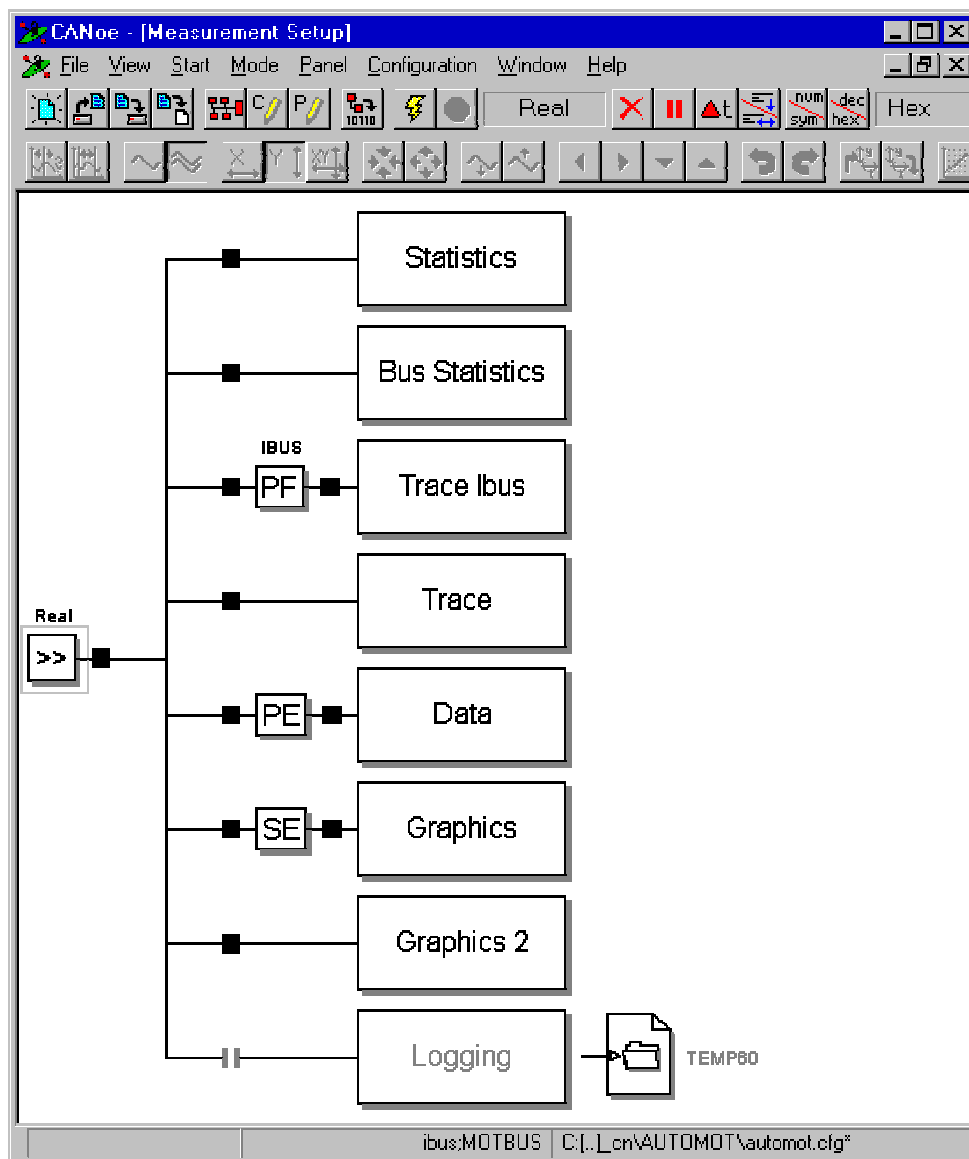


Figure 52: CANoe Measurement Setup

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

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.

The card is initialized at the start of an online measurement. You can configure the necessary parameters from the main menu under **Configuration** or from the popup menu of the simulation setup's card icon. The connection to the measurement setup is symbolized by the >> symbol on the left in the measurement setup window.

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. Instead, a log file is assigned to it for the purpose of logging bus data traffic and then studying it "offline" (cf. section 2.7.2).

Located between the function blocks are insertion points (hotspots), at which blocks can be inserted for to manipulate the data flow (Filter, replay, generator block, CAPL program block with user-definable functions). Before and after the block inserted in this manner, new hotspots appear, so that additional blocks can be inserted. The data flow can also be broken at the hotspots. You will find a description of all insertable function blocks in section 4.

Figure 52 shows a possible CANoe configuration in Online mode³, in which multiple network nodes are provided in the simulation setup. A filter is inserted in the trace branch, graphics branch and in the data branch so that only certain messages will be displayed. The statistics branch and the bus statistics branch each receive all data, while the logging branch is broken.

Note: The data flow in the measurement setup is always directional. It runs from the left, starting at the connection symbol (connection to the simulation setup) to the evaluation windows on the right.

3.2.2 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 CANoe.

³ Data flow and functions in Online and Offline modes only differ in the data source and in the transmit block. Refer to section 2.7.4 for a description of Offline mode.

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 CANoe 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.

3.2.3 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.

3.2.4 Message Attributes

Messages that were not transmitted by CANoe'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 CANoe 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.

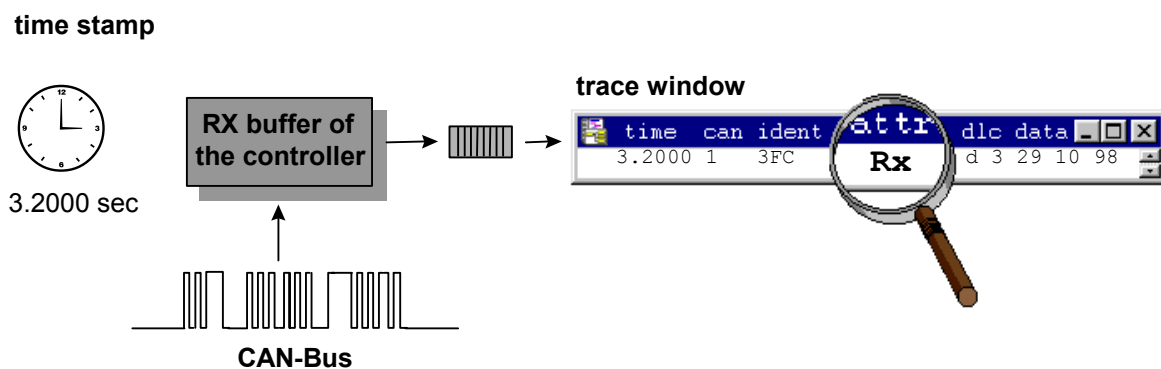


Figure 53: Receiving messages

The messages to be transmitted are passed from the simulation setup 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.

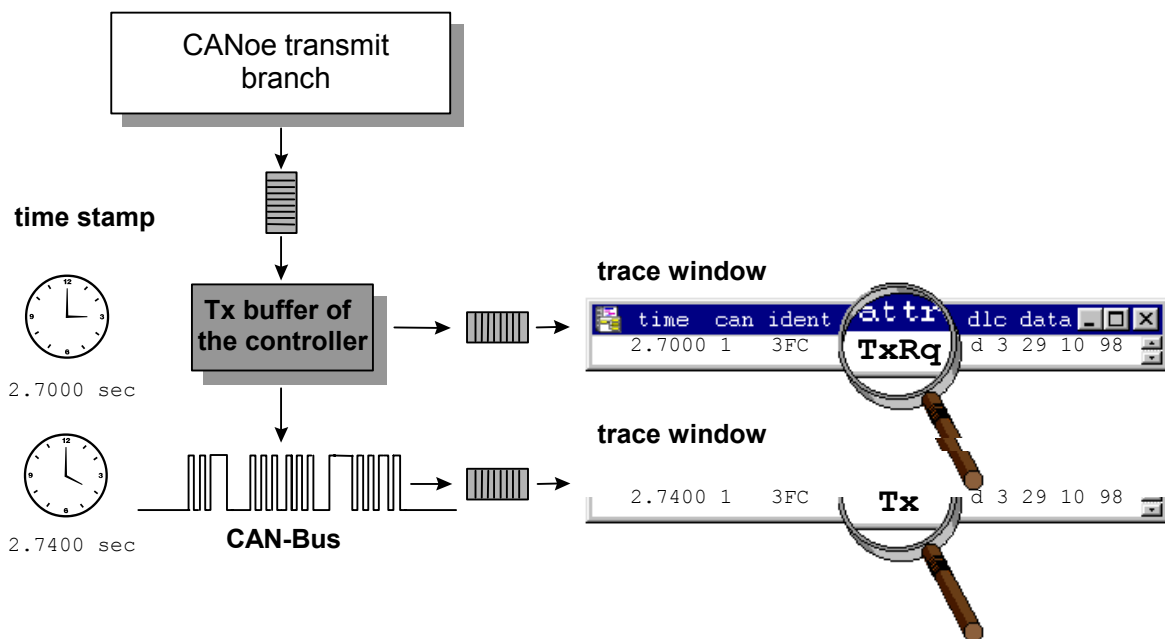


Figure 54: Transmission of messages

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.

3.2.5 Simulation Mode

At the beginning of the development process, the network is fully simulated. In this phase you can operate CANoe with or without a physical bus. In the former case it is sufficient to connect the card's two CAN controllers to the bus. In this case the operating mode in the simulation dialog (Main menu item **Configuration | Simulation**) remains set to Real Bus. All messages generated in the simulation setup are then placed on the real bus by the first chip. Here the second chip only serves as a second bus station which permits the operation of a real bus.

If you work without a real bus and real controllers, you can operate CANoe in pure simulation mode. Switch the operating mode in the simulation dialog (Menu item **Configuration | Simulation...**) from Real Bus to Simulated Bus. Bus access (sending and receiving messages) is then simulated.

In simulation mode an **animation factor** can be specified. The (simulated) measurement then appears slowed by this factor. Accordingly, the simulation is accelerated for simulation factors between zero and one. (For example, if you specify the value 0.1, the measurement would be accelerated by a factor of 10.)

The supplemental icons  and  appearing on the toolbar can be used to resume the simulation for a defined period of time or to stop a running simulation at any point in time. The length of the time step is set in the input box to the right of the icons; input is in milliseconds.

Slave mode is a special simulation mode in which the time sequencing of the measurement (time base) is controlled by an external program. A typical usage would be to control the measurement by a master program which accesses the CANoe simulation object via the COM.

CAPL Debugging enables the display of variables of CAPL nodes during a CANoe simulation as well as stopping simulations by the use of CAPL commands.

This dialog allows settings for:

- Activating/deactivating the execution of the **halt** CAPL command (default: activated)
The execution of all `halt` commands within a CANoe configuration can be enabled/disabled.
- Activating/deactivating the display of the full path in the call stack (default: deactivated)
You can select whether the call stack shows only the name of the CAPL node or the full path.
- Activating/deactivating the execution of the **inspect** CAPL command (default: activated)
The execution of all `inspect` commands within a CANoe configuration can be enabled/disabled.

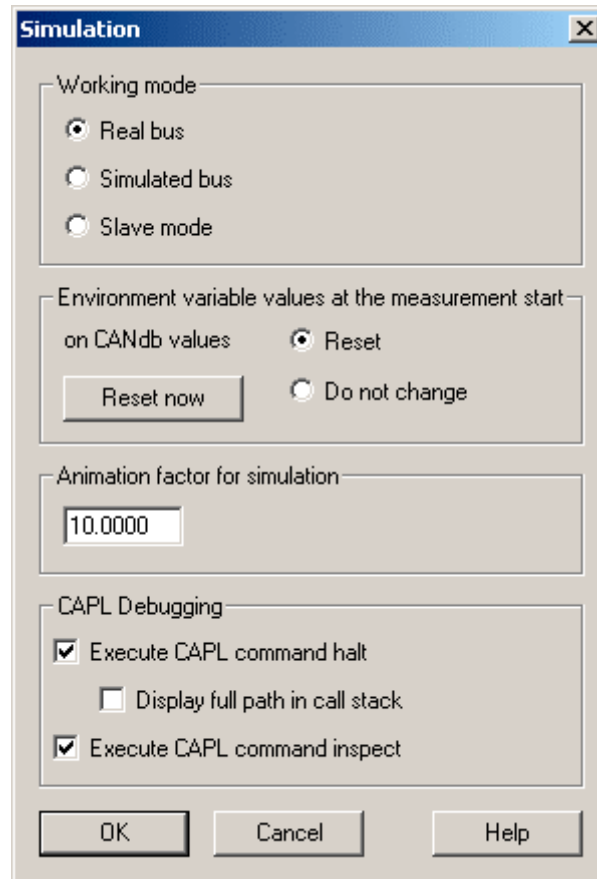


Figure 55: Simulation Dialog

Note: If you are operating CANoe in simulation mode, i.e. without a physical bus, the program emulates the functionality of the CAN chip on the interface card. In this mode, the bus baud rate is the only system parameter you would only configure in the card's configuration dialog. In this case, the values of the other card parameters do not have any effect on program execution. For small animation factors, a high baud rate might result in the time resolution of the chip emulator being insufficient. You would then get an error message, in which CANoe would inform you of the smallest feasible animation factor. To be able to continue to run a simulation in real time (Animation factor 1) you could reduce the baud rate in the card's configuration dialog. However, this will change other system properties such as the latency times of certain messages under high bus loading.

If you are operating CANoe without a card please observe the instructions in the Appendix.

3.3 Trace Window

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

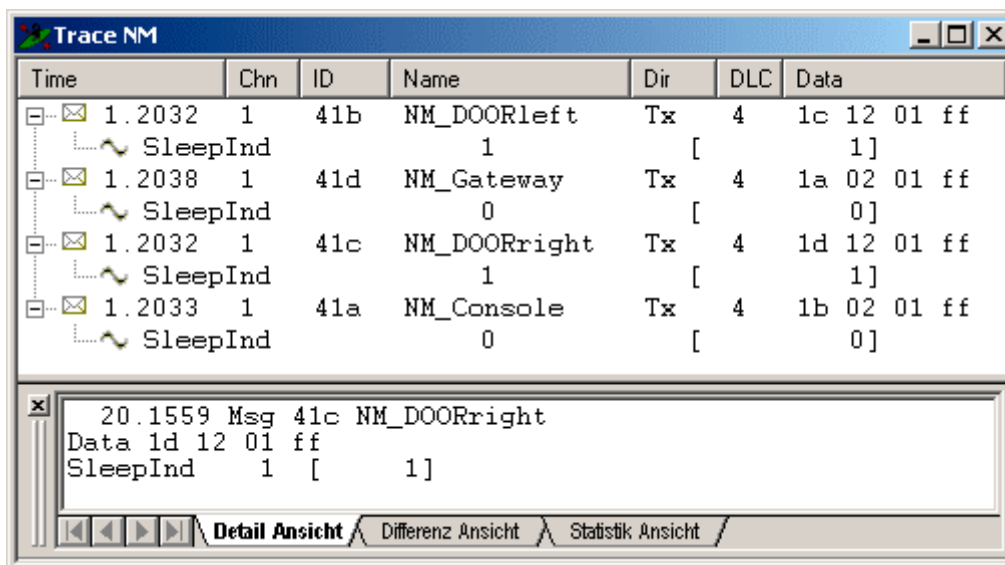


Figure 56: Trace Window of CAN

The figure shows an example of a Trace window of CAN. Depending on the specific CANoe 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:

- CANoe.LIN resp. DENoe.LIN
- CANoe.FlexRay resp. DENoe.FlexRay
- CANoe.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 C 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 in this field.

Besides these main columns, other configurable columns may exist depending on the specific CANoe option.

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.

3.3.1 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.

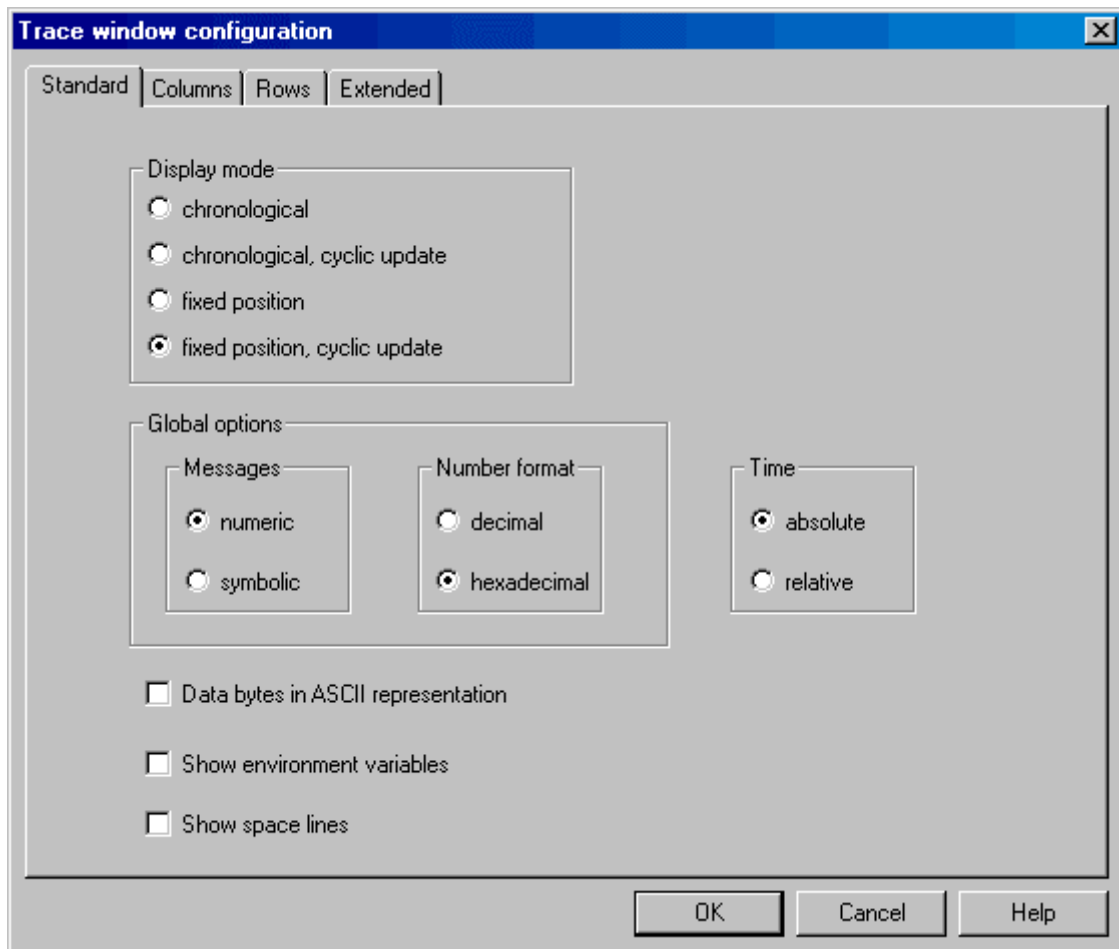


Figure 57: Configuring the Trace Block

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 that can be configured. 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.

If you also wish to see changes of environment variables in the Trace window, you must activate this by marking the relevant option.

3.3.2 Configuration of Columns in the Trace Window

Depending on the CANoe option or 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 laid out in any desired way.

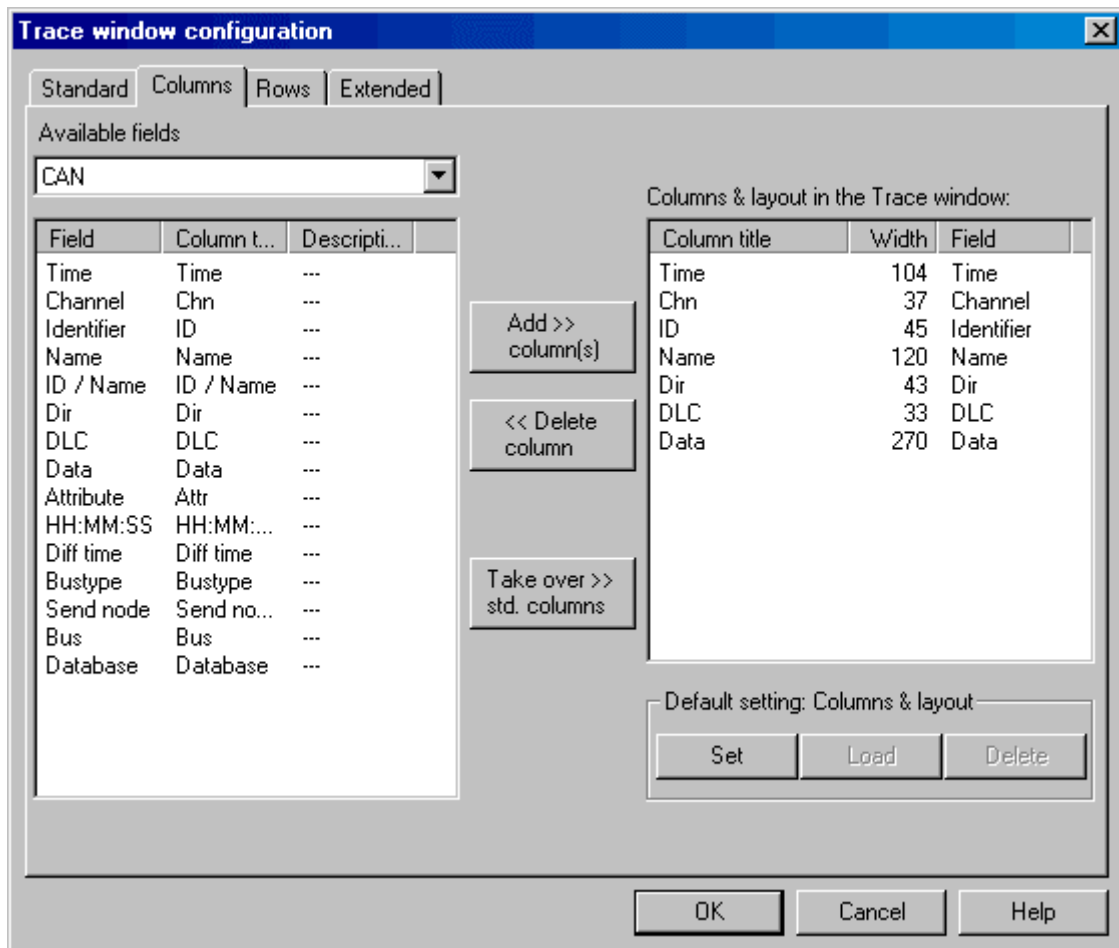








Figure 58: Columns Configuration in the Trace Window

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

Note: Not all messages or events (e.g. environment variables) 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.

3.3.3 Trace Window Options from the Toolbar

The toolbar contains five buttons with which you can directly configure the Trace window⁴:

-  Delete Trace window
-  Update/Stop Trace window
-  Toggle Trace window mode: **Chronological/Fixed position**
-  Toggle Trace window mode for time representation: **Absolute/relative**
-  Toggle Trace window mode for messages: **Symbolic/Numeric**
-  Trace window mode for numbering format: **Decimal/Hexadecimal**

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.

3.3.4 Detail View (Trace Watch Functionality)

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.

3.3.5 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 (see also 2.5 regarding this).

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.

⁴ The buttons for numbering format (hex/dec) and message format (numeric/symbolic) change the associated options globally for the entire program.

3.4 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.

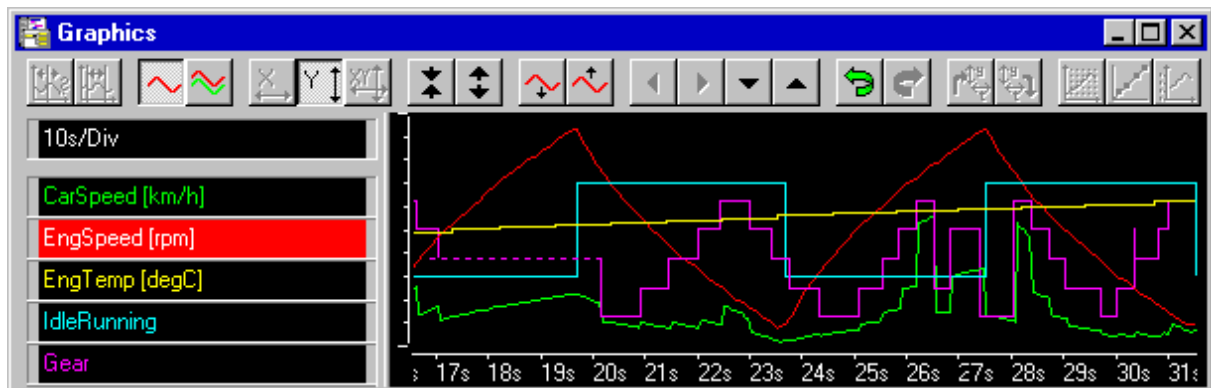


Figure 59: Graphics window

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.

Note: Besides displaying signals, the Graphics window also offers the option of viewing the time responses of environment variables. You can configure environment variables in the window's signal selection dialog by pressing the **[New EV...]** button. All of the statements made below regarding signals also apply, in principle, to environment variables.

3.4.1 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.

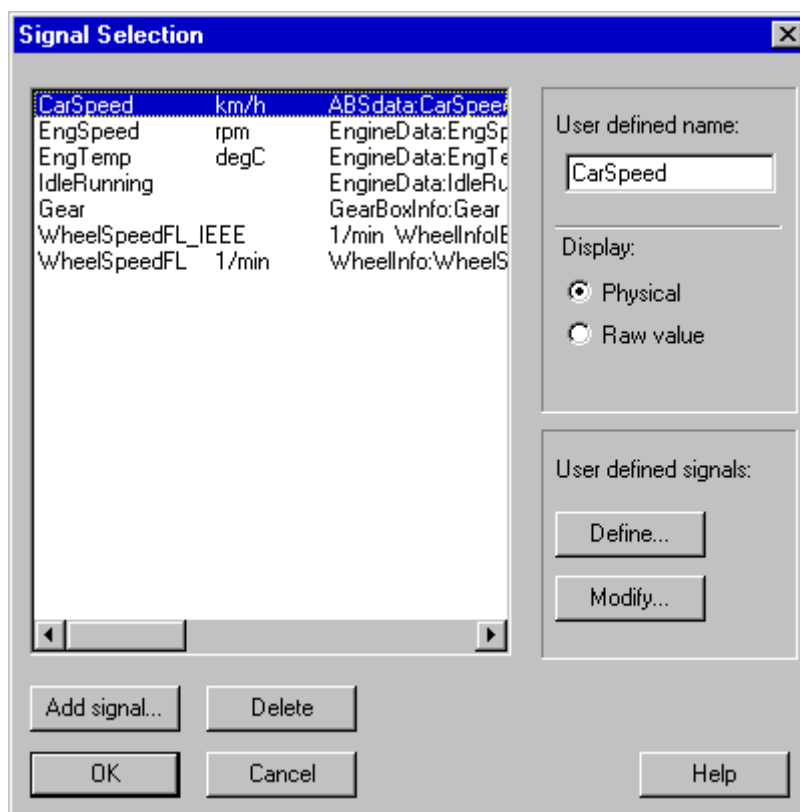


Figure 60: Signal selection dialog in the Graphics window

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.

3.4.2 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>.

3.4.3 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.

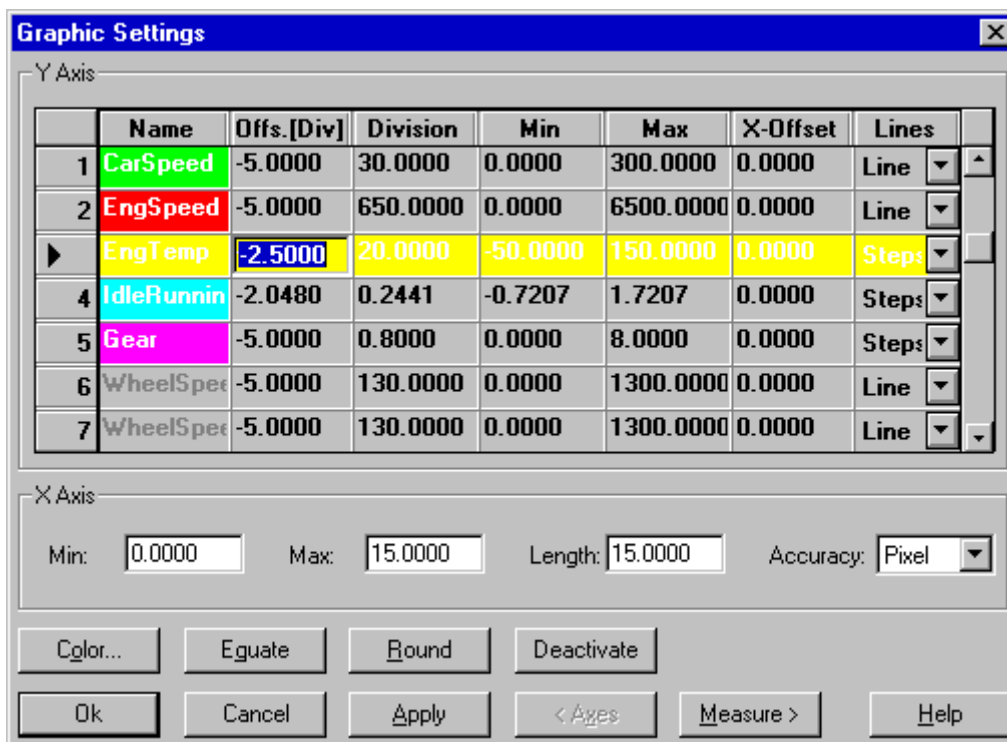


Figure 61: Configuring the axes in the Graphics window

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.

3.4.3.1 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.

3.4.3.2 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.

3.4.4 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.

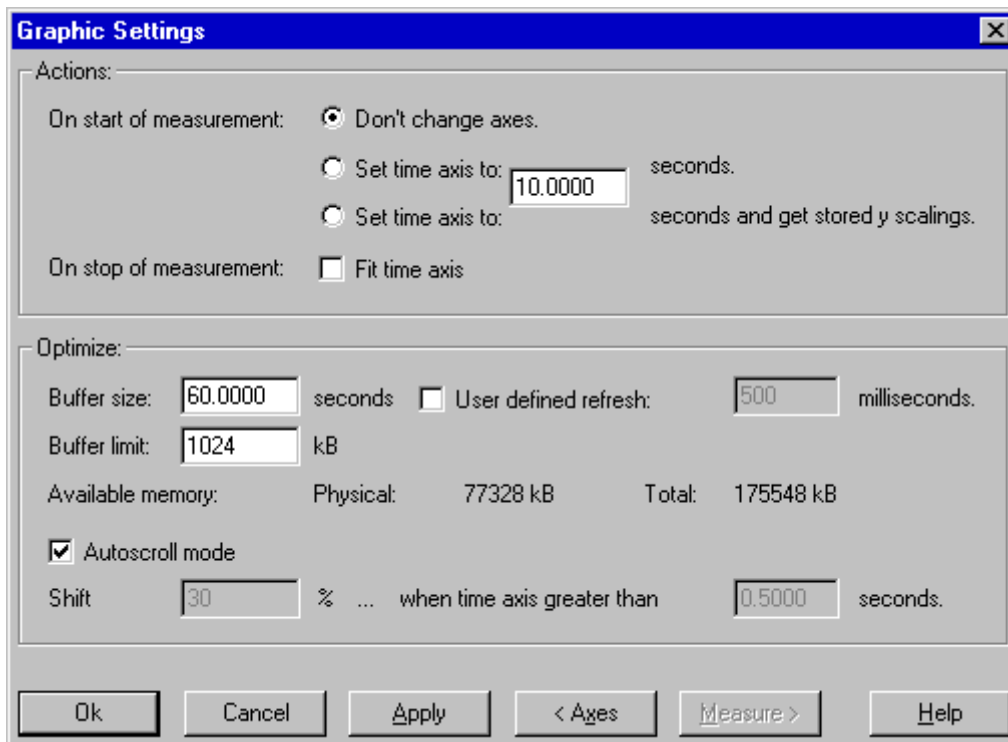


Figure 62: Measurement Options in the Graphics Window

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.

3.4.5 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.

3.4.6 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.

3.4.7 Measurement Modes

Measurement Mode "Point"

You can activate **point measurement mode** in the Graphics window with the function **Measurement cursor** 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.

Measurement Mode "Difference"

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.

3.4.8 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.

3.4.9 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 (Division = n *

10x , n = [1 to 9], x = 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 color of the measurement points matches the preset signal color.

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 color 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 color 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 color for the window (white or black). Furthermore, you can open the **Options** dialog for configuring signal colors.

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.

3.4.10 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].

3.4.11 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.

3.4.12 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 favorable 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 <code>CAN.INI</code> . Refer to documentation in section <code>[Debug]</code> of the project file regarding this.
--

3.5 Write Window

The Write window has two functions in CANoe: 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
- Inspect
- Call stack

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.

You can find a description of the most important CANoe system messages that are output to the Write window in the online help.

3.6 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.

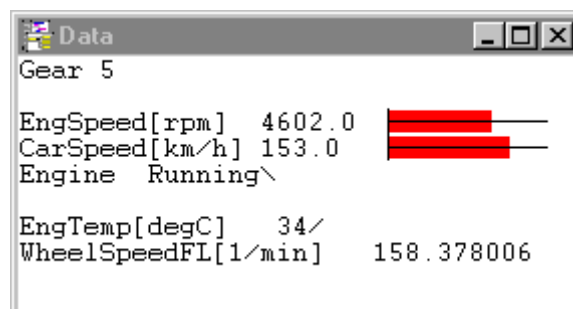


Figure 63: Data Window

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.

3.6.1 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.

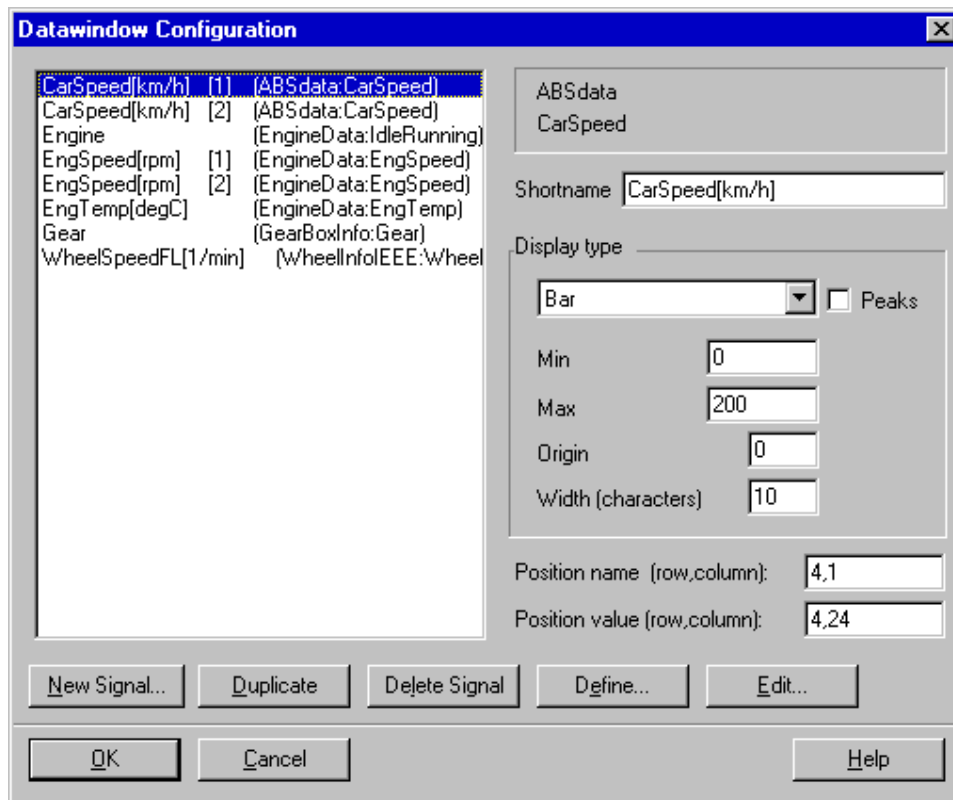


Figure 64: Data Window Configuration

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.

3.6.2 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. CANoe 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!

3.6.3 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.

3.6.4 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.

3.6.5 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.

3.7 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.

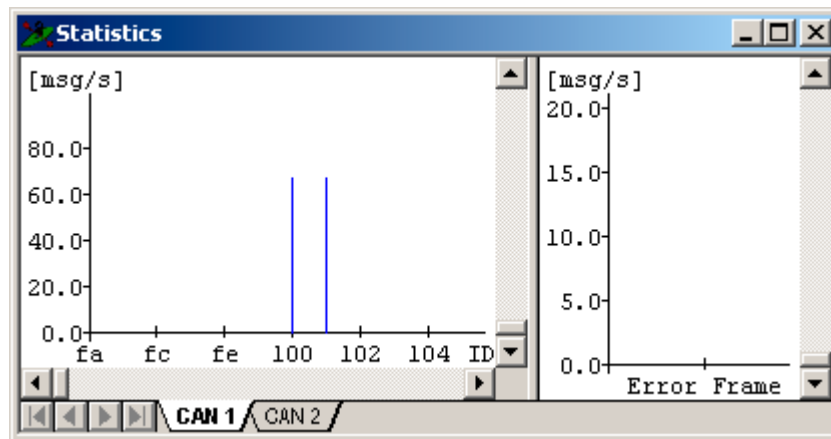


Figure 65: Statistics Window

The Statistics window displays (see Figure 65) the mean message rates existing at the end of the measurement. The Write window contains the statistics report (see Figure 67).

3.7.1 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.

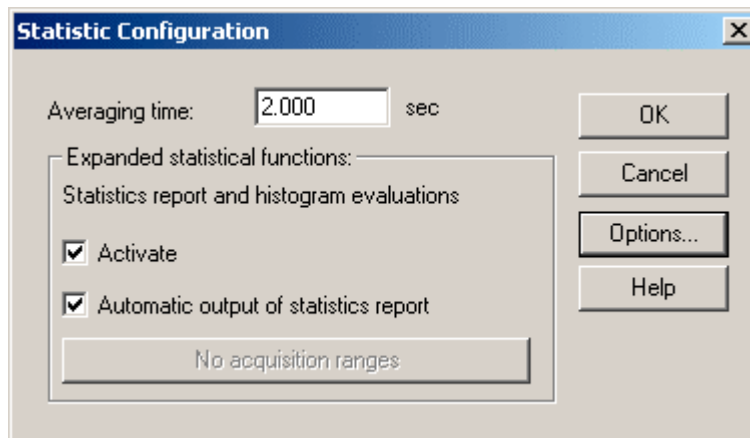


Figure 66: Configuration of the Statistics Block

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.

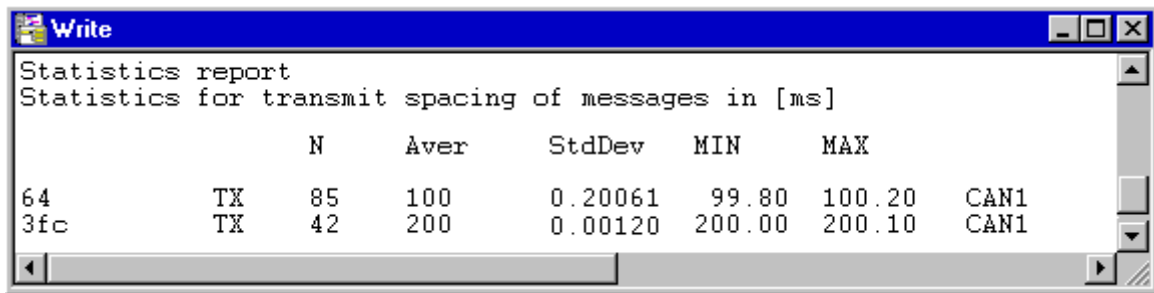
Note: If the CAN card used supports extended identifiers, the function **Basic Image** is split. The user can choose whether scaling will be over the range of standard identifiers or over the entire range.

3.7.2 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.



```
Statistics report
Statistics for transmit spacing of messages in [ms]

          N    Aver    StdDev    MIN    MAX
64      TX    85     100     0.20061  99.80  100.20  CAN1
3fc     TX    42     200     0.00120 200.00  200.10  CAN1
```

Figure 67: 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.

3.7.3 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 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.

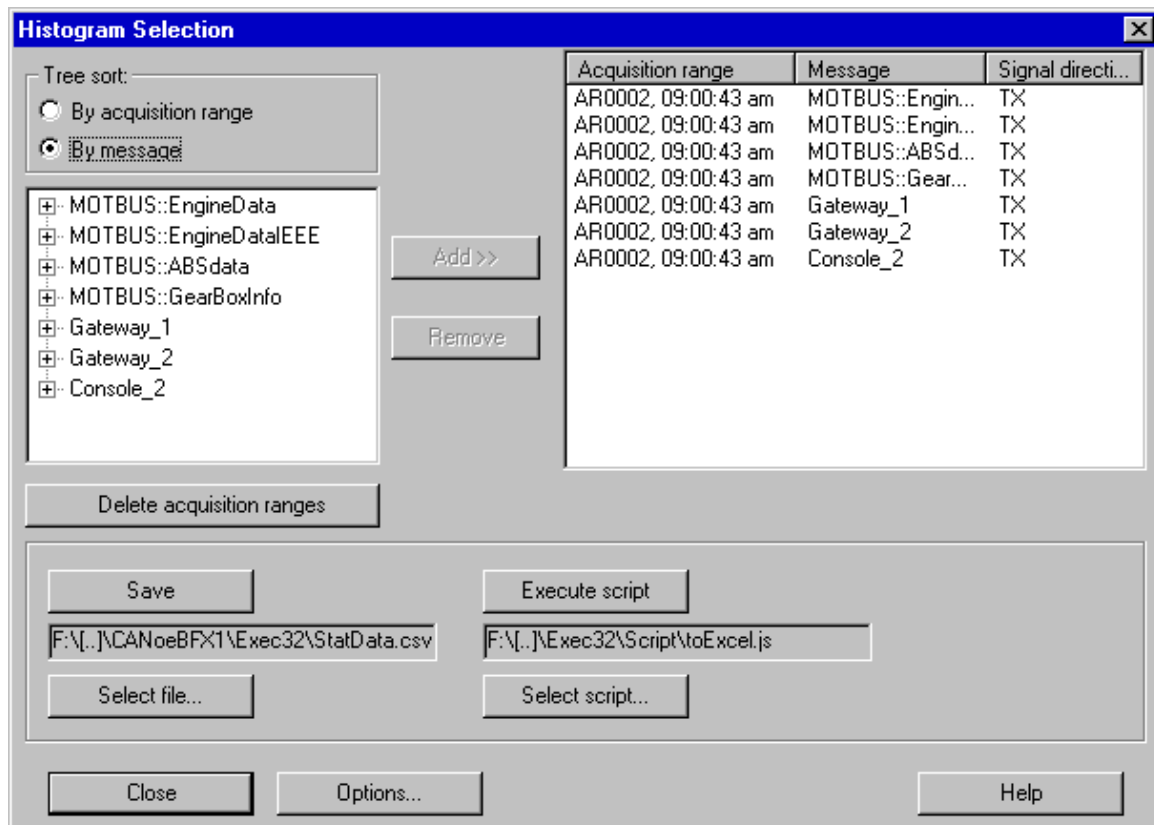


Figure 68: Histogram selection

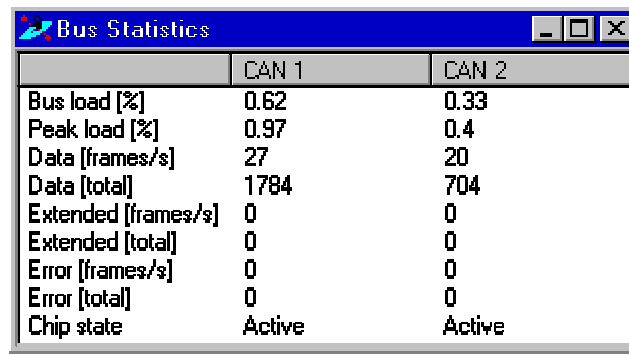
The appropriate data selected can be stored as a CSV file (values separated by commas) or transferred to Excel with the script toExcel.js. The data can also be manipulated with the user's own Java or Visual Basic script.

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

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

3.8 Bus Statistics Window

CANoe 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 69: Bus Statistics Window

Note: If you have installed CANoe.LIN you can find additional information for the shown statistic data in the manual's chapter Option LIN or in the online help.

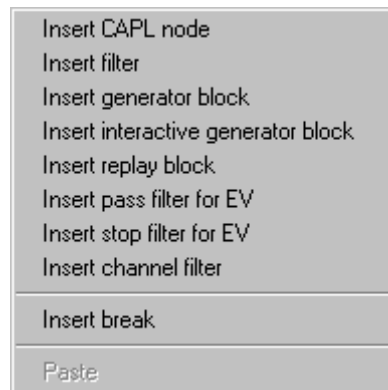
Under **Options** in the card icon's popup menu in the measurement setup window, the user can configure the time interval at which the card passes bus statistics information to CANoe. This interval defines the frequency of the bus load measurement and thereby also the averaging time. The default value 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 CANoe 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 2.7). 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 seven 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.

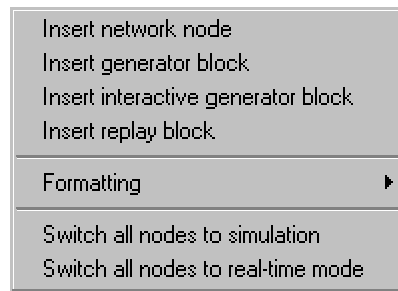
All blocks have popup menus which the user can open by either clicking with the right mouse button or by selecting the block in the data flow plan and then pressing <F10>. The first menu item opens a configuration dialog for the particular block, which only serves to parameterize the block (there are 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.

In the simulation setup the user can insert function blocks directly by means of the bus image. When you click the bus image with the right mouse button (or select it with cursor keys followed by <F10>), a popup menu appears with the following menu commands for inserting function blocks:



The following table gives you an overview where in the dataflow the function blocks should be practically inserted.

Function block	Type	Symbol	Reasonable place of use
Generator block	Data source	G	/Simulation setup
Interactive Generator block	Data source	IG	Simulation setup/Measurement setup
Replay block	Data source	R	Simulation setup
Stop filter	Data sink	SF	Measurement setup
Pass filter	Data sink	PF	Measurement setup
Channel filter	Data sink	--	Measurement setup
CAPL-Program	Data source/sink	P	Simulation setup/Measurement setup
EV Pass filter	Data sink	PE	Measurement setup
EV Stop filter	Data sink	SE	Measurement setup
Network node block*	Data source	Prog	Simulation setup

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 are displayed in the data flow diagram as little blocks with the inscription **G**.

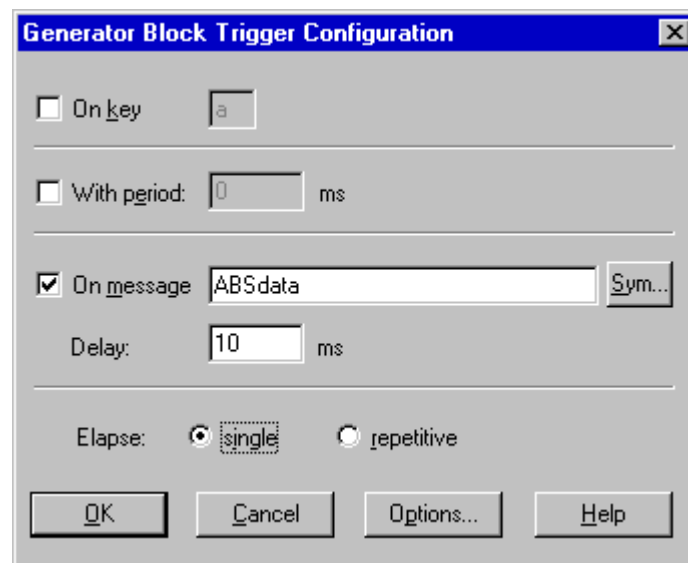


Figure 70: Generator Block Configuration - Triggering

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:

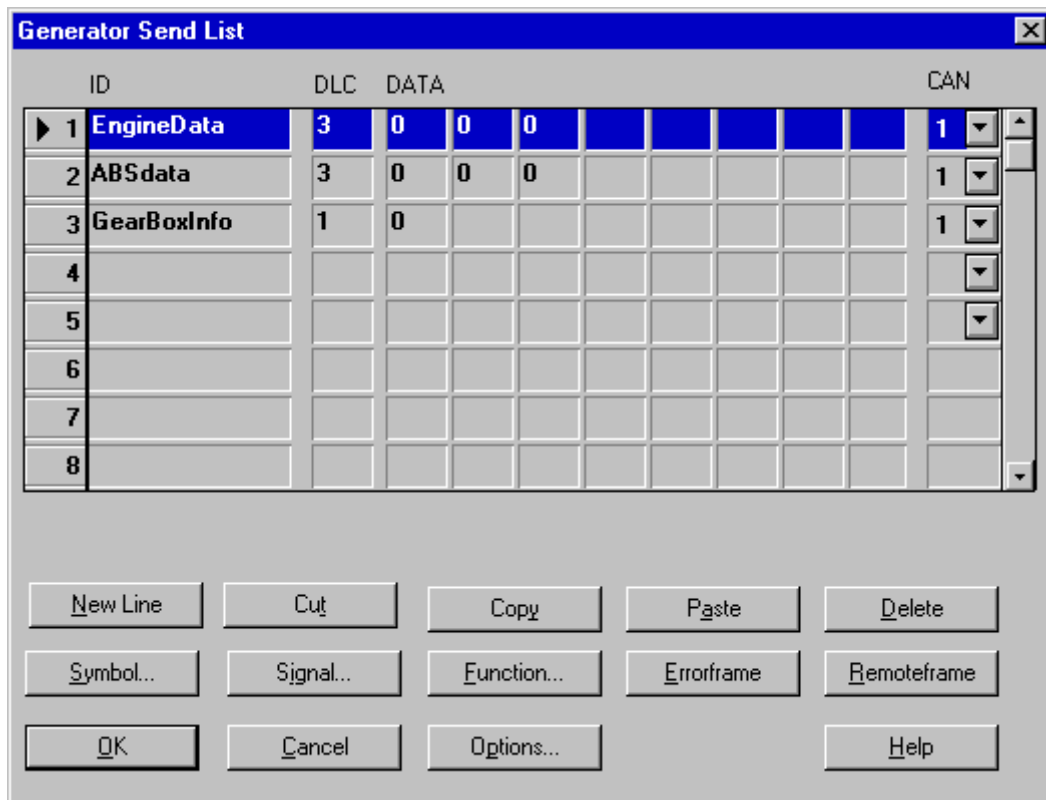


Figure 71: Generator block Configuration of Transmit list

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.

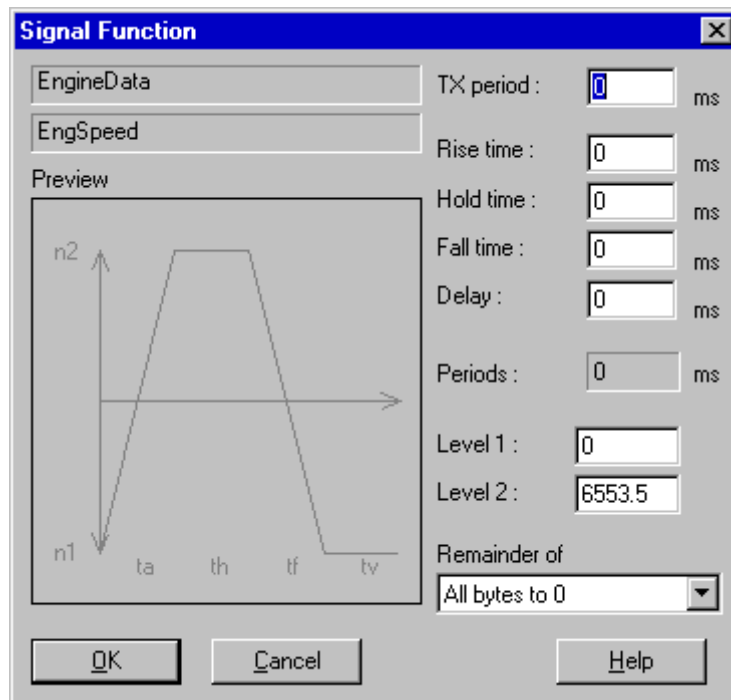


Figure 72: Signal Response Generator

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.

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

4.2.1 Configuring the Interactive Generator Block

The **configuration dialog** 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.

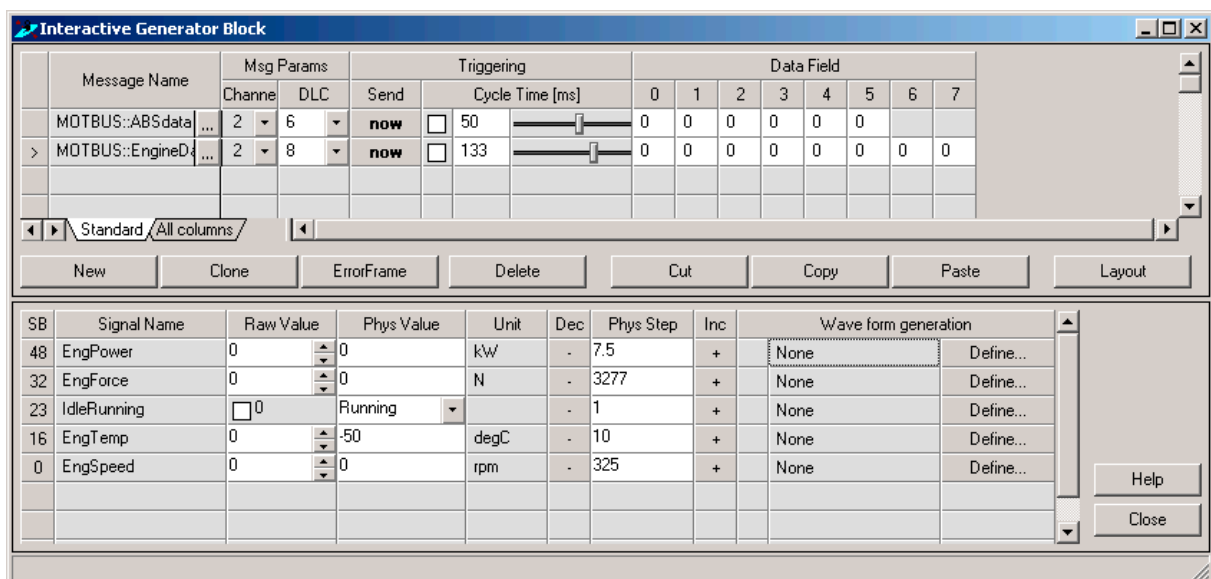


Figure 73: Configuration of the Interactive Generator Block

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 CANoe 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 sig-

nals 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
- Environment variable
- 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 re-

quest 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 CANoe 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 CANoe 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.

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.

Instructions on how to use Replay blocks to also play back environment variables, e.g. to generate test sequences, are given in section 4.3.1.

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.

4.3.1 Replay of Environment Variables

The replay block provides you with the opportunity to inject both binary and ASCII files into the simulation setup.

Thus, you can edit ASCII log files with a conventional text editor before you play them back in the replay block. In particular, you can also generate your own ASCII files with the sequences of values of certain environment variables to inject them into the measurement through the replay block.

The replay block reads an ASCII file line-by-line. Enter the number format in the first line. If the integer values in your file are to be interpreted as decimal values, you write:

```
base dec
```


If you wish to enter the numbers in hexadecimal format, you write:

```
base hex
```

You can comment out any line of the file by two forward slashes followed by a blank character (`//`). For lines which are to be fed into the measurement from a replay block, you would first enter the time of transmission in seconds (after start of measurement) in the first column. To play back an environment variable you would write the name of the environment variable in the second column followed by a space character, the relational operator `:=`, another space character and the variable value.

Besides environment variables, in the ASCII replay you can also inject messages and error frames in the ongoing measurement. Again, the first column contains the time of transmission. In the second column you would specify which of the two CAN controllers (1 or 2) you wish to have the message or error frame transmitted by. For an error frame you would write "Errorframe" in the third column. You have the option of working with either symbolic names or message identifiers.

4.3.2 Example of an ASCII Replay File with Environment Variables

```
// Number format: Decimal
base dec
// Time      Name of EV      Value
1.0000      LightSwitch    := 1
2.0000      LightSwitch    := 2
4.0000      MotorSwitch    := 1
4.1000      SpeedEntry     := 100.0
4.2000      SpeedEntry     := 150.00
4.3000      SpeedEntry     := 200.00
4.4000      SpeedEntry     := 300.00
4.5000      SpeedEntry     := 500.00
4.6000      SpeedEntry     := 800.00
4.7000      SpeedEntry     := 2000.00
4.8000      SpeedEntry     := 2200.00
```

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 2.7.1 (page 60) 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 CANoe 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.

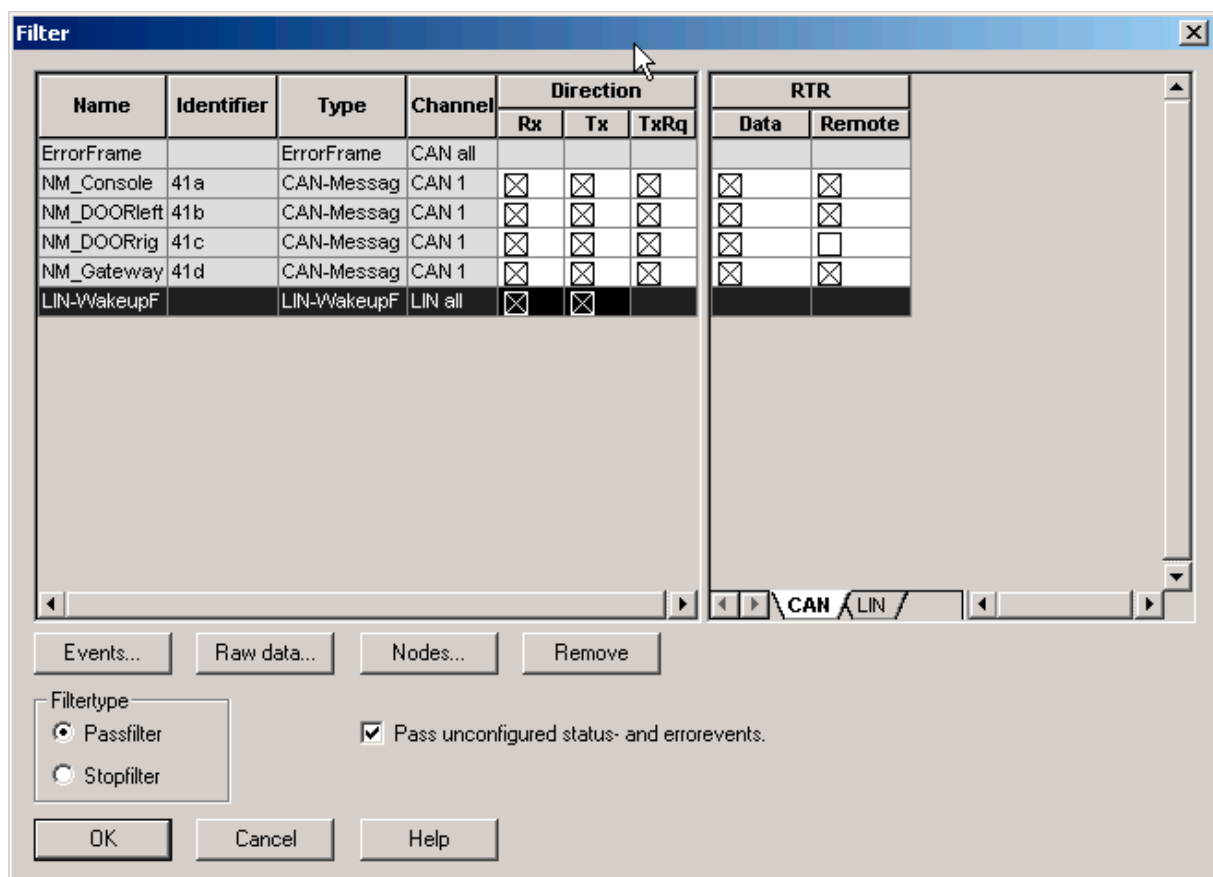


Figure 74: Filter configuration dialog

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).

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.

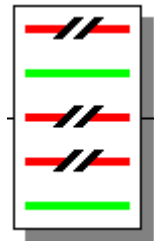


Figure 75: Channel filter in the data flow plan

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

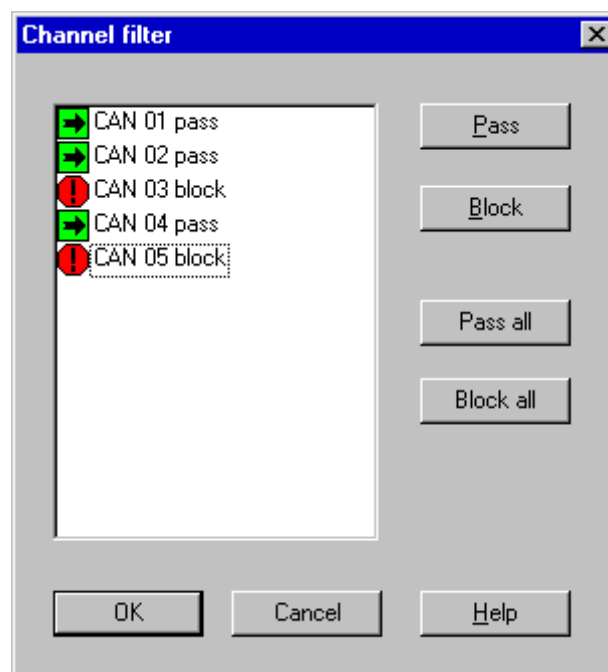


Figure 76: Configuration dialog for channel filter

The filter configuration is lost when a channel filter is removed from the measurement setup (Delete this node in the popup menu or with)

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 Simulation Setup

A CAPL node is a universal function block whose characteristics the user defines by writing a CAPL program. CAPL nodes in the simulation setup are called **Network nodes**. Together with the real nodes they define the functionality of the overall system. A functional description of a network node includes the node's behavior with regard to input and output variables as well as messages to be received and transmitted. The event-driven, procedural language CAPL is provided in CANoe for modeling network nodes.

When a new setup is created (Menu item **File | New configuration**), first the simulated bus (red line) and the real CAN bus (black line) are displayed in the simulation setup. The two buses are connected to one another via the card icon.

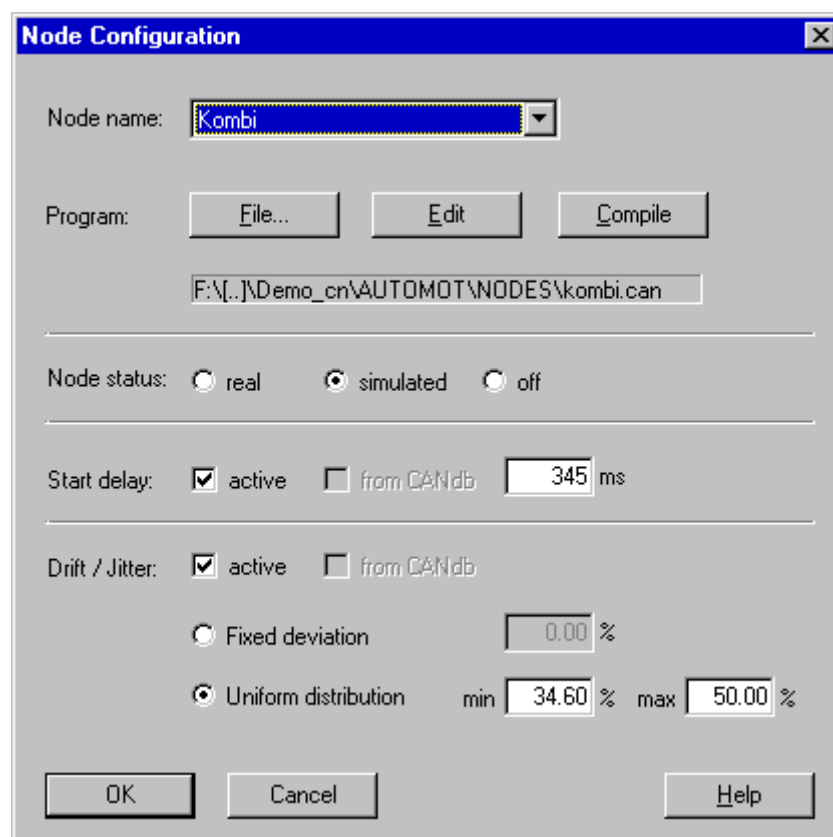


Figure 77: Configuration Dialog for Network Node

When you click the bus image with the right mouse button, the popup menu of the CAN bus appears. You can insert a network node in the simulation setup with the menu command **Insert network node**. This is displayed as a rectangular function

block which is initially connected to the simulated bus, i.e. the bus with the red line. The function block contains two text lines. The upper line shows the name of the node and is initially empty for a newly inserted node. The status bar below it provides information on node status and node type.

The configuration dialog for a network node can be opened by right clicking on the network node and choosing **Configure** from the popup menu. The corresponding node name can be selected from those defined in the database; if no name is selected the name will remain << default >>.

With the **[File...]** button you can open a file selection dialog. Here you can select the CAPL program which describes the functionality of the network node. The **[Edit]** button opens the CAPL Browser with which you can create the selected CAPL program or edit it later in a user-friendly manner. All CAPL programs must be compiled before the measurement start. This is done by activating the **[Compile]** button or by selecting the menu command **Configuration | Compile all nodes**.

The node status option buttons can influence the function of a node. The status *real* indicates a node on a real bus. It is shown in the simulation setup by a black connecting line to the actual bus and is designated as *real* in its status line. In this case the network node model has no influence on the measurement.

Simulated nodes are represented as red connection lines to the simulated bus. During a measurement its functionality is simulated by an assigned CAPL program. The bus behavior of the network node does not differ from that of a real controller with the same functionality.

Individual network nodes can be switched-off intentionally with the options button **off**. This breaks the connection to the bus.⁵

The user simply presses the spacebar to toggle the status of the currently active node in the simulation setup. The node status changes with each spacebar press from *simulated* to *real* to *off* and back to *simulated*.

The menu item **Start delay** can influence the behavior of the network node before the start of the measurement. The button switches this influence to active, which causes the node to remain inactive for the set time period after the start of the measurement. Messages are neither sent nor received during this time, nor do they react to external conditions such as environment variables or key presses.

This delay can also be defined as a supplemental attribute in the database and activated via the corresponding checkbox.

The simulated network node timers can also be influenced by **Drift** and **Jitter**. When this is the case, the user can toggle between fixed deviation and an equally distributed fluctuation. With fixed deviation the value can be given in per cent, and all timers for this network node will expire a bit faster or slower around this value. With uniform distribution, on the other hand, an interval can be entered from which the fluctuations of each timer is purely chosen.

These values can also be defined as supplemental attributes and activated from CANdb++.

⁵ In Version 1.0 this has the same effect as the option *real*.

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

4.8 CAPL Nodes in the Measurement Setup

Important applications of program blocks in the measurement setup include, e.g. activation of triggers or data reduction or monitoring in the measurement setup.

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**. When you click these blocks with the right mouse button and choose Configuration a dialog appears with the following buttons:

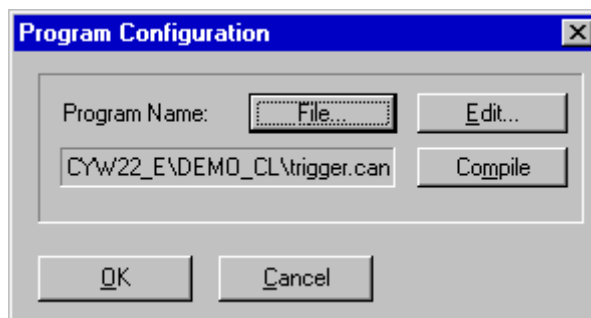


Figure 78: Configuration Dialog for CAPL Programs in the Measurement Setup

First, in the configuration dialog you assign a CAPL file name (Extension `*.CAN`) to the program block.

Press **[Edit]** to open the CAPL Browser. Browser is a user-friendly tool for creating, modifying and compiling CAPL programs.

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 same data manipulations are to be made in two different data flow branches (e.g. data reduction operations).

You can deactivate the node by pressing the spacebar or selecting the item *Node active* in the CAPL node's popup menu. Executing the same action again will reactivate the node. The menu item **Delete this Node** removes the CAPL node from the measurement setup.

Note: The CAPL source file is not deleted when a CAPL node is removed from the measurement setup (by selecting the command "Delete configuration" in the popup menu).

4.9 Environment Variable Filters in the Measurement Setup

4.9.1 Pass and Stop Filters for Environment Variables

A targeted reduction of data volume can be achieved in the measurement setup with pass and stop filters for environment variables. Only the specified environment variables are passed or blocked.

Pass filters or stop filters for environment variables appear in the data flow plan as small blocks with the label **PE** or **SE**. When the user clicks one of these blocks with the right mouse button a popup menu appears in which the filter can be configured or deleted. Configuration is performed in a dialog box, in which the environment variables can be selected from the database.

Note: In keeping with its function, a pass filter which is not configured, i.e. is empty, does not pass any environment variables.

4.9.2 Configuration of the Environment Variable Filter

The characteristics of the environment variable filter are entered by the user in the configuration dialog.

The name of the active database appears in the uppermost line. You can select the filter for the environment variables from this database.

The user activates the filter with **Mode=active**. If the mode is inactive the filter is ignored in the data flow plan.

With the **[Add]** button you can accept additional environment variables in the filter. With **[Delete]** you remove all environment variables from the filter which were previously marked in the selection list.

4.9.3 Selecting Environment Variables

In this dialog you can select environment variables in the multiple selection list on the right side of the dialog.

The left side of the dialog is used for preselection of environment variables which are displayed in the multiple selection list. In the upper left box you can check off whether all those environment variables should be displayed, to which at least one network node of the database has read or write access. In the two lower boxes you check off whether all those environment variables should be displayed, to which at least one of the selected network nodes from the network node list has read or write access. (Read and write authorizations for environment variables must be assigned in the database.)

The check boxes which you check are logically OR'd with one another. Therefore, it is not possible for example to simultaneously check off the upper and third check boxes, since both relate to read access of environment variables.

4.9.4 Examples of Preselection of Environment Variables

If you are working with databases for which read and write access to environment variables was not defined fully or not defined at all, please observe the following: In order to be able to select environment variables from such "incomplete" databases, all environment variables for which neither read nor write access was defined are treated as though all network nodes can access them.

- To display all environment variables from the database in the environment variable list, check off **All with read access** and **All with write access**.
- To display all environment variables in the environment variables list to which the network node NK1 has read and write access, first select the network node NK1 at the lower left in the network node list and then mark the check boxes **With read access** and **With write access** in the lower left area.
- To display all environment variables to which the network nodes NK1 and NK2 have write access, first select both network nodes in the network nodes list and then mark the check box **With write access** in the lower left area. Make sure that none of the other check boxes are marked.

4.10 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 The Panel Editor

The Panel Editor is used to create graphic panels. With these panels the user can change the values of discrete and continuous environment variables interactively during the simulation.

Start the Panel Editor with the  button on the CANoe toolbar, with a double click with the left mouse button on an opened panel or - after selecting one or more panels - with the **[Edit]** button of the panel configuration dialog. This will ensure that the databases of your CANoe configuration will be associated automatically.

As an alternative, together with the program call on the command line you can enter a CAN database that contains the names and types of the necessary environment variables. For example, the call `vpanel /d example.dbc` starts the Panel Editor with the database `EXAMPLE.DBC`.

You can create a new panel with the menu item **File | New**. **File | Open** opens an existing panel. If the database does not contain the environment variable names associated with the panel's indicator and control elements, you will receive an error message. In this case you should first associate the correct database or start the Panel Editor directly from CANoe.

File | Save saves changes made in the editor to the active panel file, while selecting the menu command **File | Save as** saves the active panel to a new file.

5.1 Introduction

At the start of the Panel Editor from CANoe, the environment variable lists of the databases configured there are automatically associated.

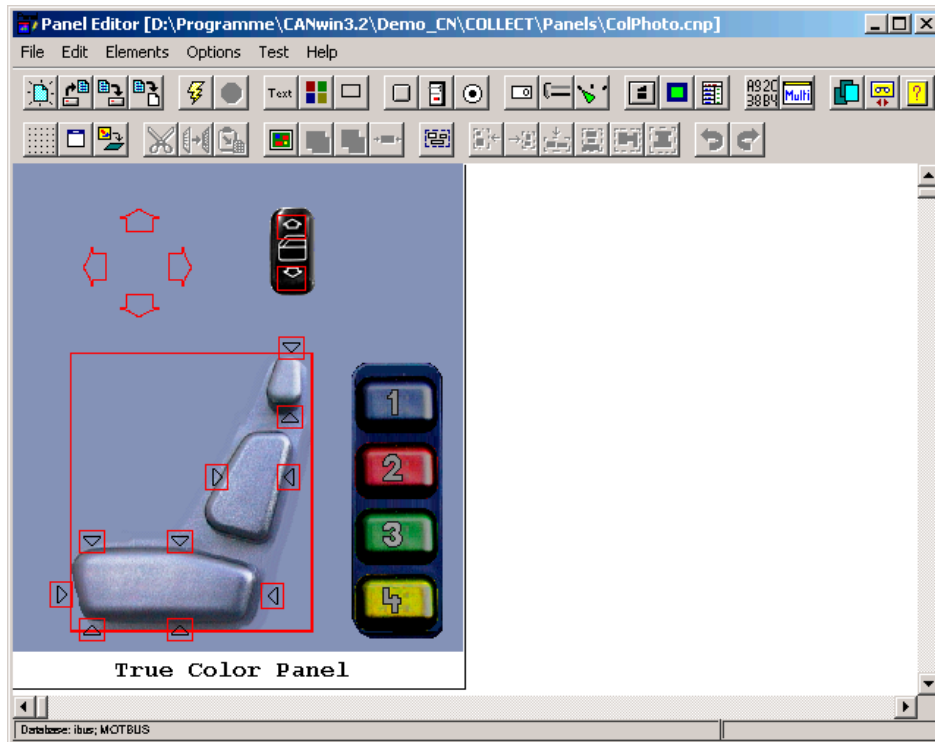


Figure 79: The Panel Editor

All elements which you can place on a panel in the Panel Editor are called controls. Differentiation is made between control elements for which the values of environment variables can be changed (e.g. switches, pushbuttons, etc.) and display elements for which values of environment variables are displayed (e.g. lamps). Some elements (e.g. sliders) can be used as both control and display elements.

Certain elements are available for each environment variable type. For example, you would use switches and lamps for discrete environment variables, sliders for continuous environment variables and text input fields for character string environment variables.

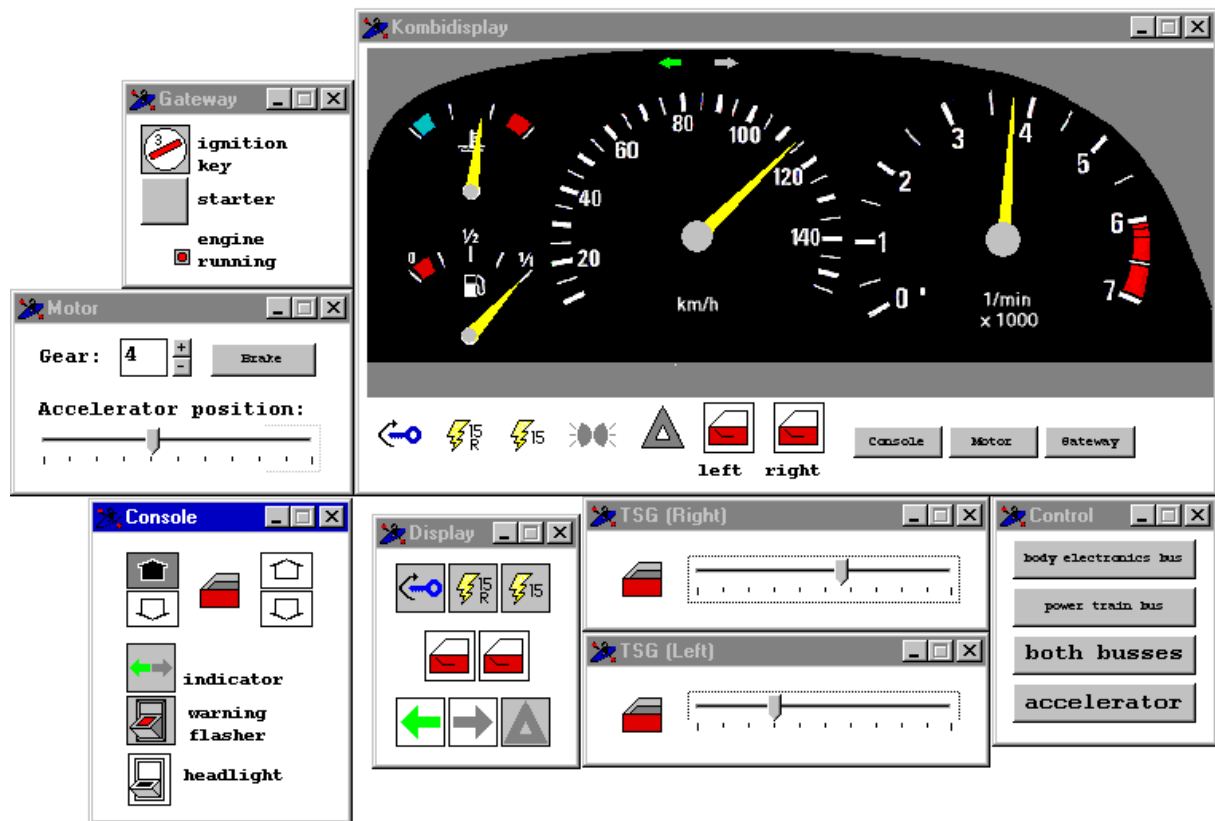


Figure 80: CANoe Panels

In the Panel Editor you assign an environment variable or a signal - whose value is to be changed or displayed - to all control and display elements. Moreover, you can place static elements (text, bitmaps) on the panel which are independent of the environment variables.

All control and display elements (switches and buttons for discrete environment variables, sliders for continuous environment variables, etc.) can be placed freely on the panel and may also overlap. New alignment functions for individual elements and group functions (e.g. center, left-justify, etc.) simplify the structuring of panels.

Elements can be configured after double clicking with the left mouse button. A special requirement is that each control must have an environment variable or a signal assigned to it from the database. Panels are saved in CANoe panel file format (default extension `CNP`) and can then be loaded in the simulation system.

Besides a variety of predefined elements, the Panel Editor lets you insert bitmap elements with user-defined bitmaps. Overlapping and transparent elements allow you to place controls in any desired manner on the panel, adjacent to one another or above one another. They also enable the display of a wide variety of bitmap element forms. There is a transparency color for the overall panel. This allows background bitmaps for bitmap elements whose transparency is active to shine through.

Note: It is now possible to automatically create the necessary panels from a correspondingly prepared and initialized database with the Panel-Generator. These panels can then be adapted to personal needs. For the detailed procedure on how to use the Panel-Generator, please read the Online-Help.

5.2 Editing a Panel



Under the menu item **Options** you will find entries for editing the panel.

Panel Size, Name, Color and Fonts... opens a dialog in which you can define the name, size, background color and transparency color of the panels.

During CANoe operation the name of the panel appears in the panel window's title bar. To make all elements you have positioned on the panel visible you must observe the minimum values for window height and width given in brackets. The transparency color applies to all panel bitmap elements whose transparency is active.

Please note that in adjusting the transparency color you also affect the elements with default bitmaps, since they are based on the default option of the transparency color.

With the menu item **Set grid** you open a dialog in which a grid can be defined for positioning the elements. All elements are always positioned exclusively on the grid points of the grid. This function makes it easier for you to attain horizontal and vertical alignment of multiple elements on the same panel.

Note: With the buttons  (Undo) and  (Redo) you can undo or repeat the last actions for placing and aligning elements.

5.3 Display and Control Elements

In this chapter creating, configuring and arranging of elements is treated.

5.3.1 Creating Elements

Under the menu item **Elements** you will find all control and display elements (Controls), which can be integrated into a panel.

To position an element in the panel area, first select an entry from the menu. As an alternative you can select an icon of the toolbar. (You will see a brief description of the icon to which the mouse cursor is pointing in the status line on the lower border of the Panel Editor.) You can select the element's position with the cursor and confirm it with the left mouse button. After they have been inserted, it is possible to move any of the elements on the panel. To do this, select the element with the mouse, drag it to the new position with the mouse button pressed, and then release the mouse button (Drag & Drop).

Control Element	Type of Environment Variable	Switch Value	Size configurable?	Type

Text	-	-	Yes	static
Bitmap	-	-	As bitmap	static
Frame	-	-	Yes	static
Switch	Discrete	1	No	Sensor
Push Button	Discrete	1	Yes	Sensor
Radio Button	Discrete	Selectable	No	Sensor/ Actor
Slider	Continuous	Interval	Yes	Sensor/ Actor
Input/Output Control	Dis- crete/Continuous/ Character string	-	Yes	Sensor/ Actor
Bitmap Push Button	Discrete	Selectable	As bitmap	Sensor/ Actor
Bitmap Switch/Indicator	Discrete	Selectable/ Multistage	As bitmap	Sensor/ Actor
Value Table Box	Discrete	-	Yes	Sensor/ Actor
Meter	Continuous	Interval	Yes	Actor
Hex Editor	Binary data/ Character string	-	Yes	Sensor/ Actor
Panel Recorder	-	-	No	special element
Panel Control Button	-	-	Yes	special element
Panel Help	-	-	No	special element
Multi Display	-	-	Yes	Actor
ActiveX Control	Control dependent	Control de- pendent	Yes	Control de- pendent

Elements of the Panel Editor

You can change the sizes of push buttons, sliders, text boxes pointer elements and all input boxes. To do this you drag the border of the particular control to the desired size while the mouse button is pressed. The marking frame sizes of bitmaps and bitmap switches are indeed also adjustable on the panel, but the actual bitmaps always retain their height and width.

With the standard functions **Cut**, **Copy**, **Paste** and **Delete** of the popup menu or the main **Edit** menu, you can copy individual elements to the clipboard and from there paste them on the same panel or another panel.

5.3.2 Configuration of the Elements

Double clicking on an element in the panel opens the configuration dialog for the particular element. For example, with push buttons you can set the label as well as the text font, while for bitmap elements you must input the name of the associated bitmap file.

When the panel is created you must assign an environment variable or a signal to each control and display element. To do this, select the desired environment variable or the desired signal from the appropriate selection list in the configuration dialog. Note, that signals indicate status only but no actions can be triggered.

Note: When the <Shift> key is pressed while the mouse pointer is simultaneously moved over an element, a window appears containing the name of the assigned environment variables or the assigned signal. If you have entered a comment for the environment variables in the CAN database, the first comment line of the environment variable is displayed.

For the elements bitmap switch/indicator, push button, switch, radio button and bitmap push button you must also enter a switch value. This value corresponds to the value of the environment variable when the element is active. For example, a three-stage switch can be constructed from three radio buttons by assigning the same environment variable to all three elements, but selecting different switch values for each of the radio buttons, e.g. 1 for the first, 2 for the second, and 3 for the third trigger.

The switch value 1 is assumed for push buttons and switches. Multi-stage switches are incremented (decremented) by 1 when the right (left) mouse button is pressed, respectively, whereby the smallest switch value is always 0. In configuring continuous elements you must specify minimum and maximum values for the assigned environment variable.

The "Alarm states" function makes it possible for the elements meter, slider, input/output box and ActiveX control to respond to a situation in which a value is above or below of a previously defined value range.

This makes it possible to change the visual representation of the panel element that displays the value when a specific limit value is reached. For example it is possible to respond to a situation in which a value is above or below of a previously defined value range of a pointer element by changing the color of the pointer.

For information on configuring elements please refer to the Panel Editor's Online Help function in the particular configuration dialog.

5.3.3 Arranging the Control Elements

Besides the standard functions for all indicator and control elements - **Cut**, **Copy**, **Paste** and **Delete** - with which you can copy individual elements to the clipboard and paste them from there to the same panel or another panel, the **Edit** menu offers you a number of functions for arranging and grouping elements. All functions relating to an element can also be activated directly via the respective popup menu. The following functions are available for arranging elements:

Set to Foreground/Set to Background

Places the selected element in the foreground/background and refreshes the panel with the active overlap sequence. After this function has been executed the element is deselected.

Center

Places the selected element in the center of the panel horizontally.

Adjust Size

Adjusts the size of the bitmap element according to the number of states entered in the configuration.

Left Alignment/Right Alignment

Places elements included in the group below one another, left/right justified, whereby the element located furthest to the left serves as reference.

Center in Group

Centers elements included in the group beneath the uppermost group element.

One Level

Places all group elements on a single horizontal level, whereby the element located furthest down serves as reference.

Same Height/ Same Width

All group elements assume the height/width of the tallest/shortest group element, provided that this is permitted.

Tile with Same Spaces Horizontally/Vertically

All group elements are tiled with the same spaces horizontally or vertically. The space between the first two elements is taken over for the following spaces between all other elements.

Tile Horizontally/Vertically

All group elements are tiled horizontally or vertically.

Group

Combines all selected elements into a group.

After activating the function with the left mouse button you select all elements to be incorporated into the group by enclosing them in a rectangle. After releasing the left mouse button, a thick frame appears around the highlighted rectangle and all elements contained in the group.

Only elements completely enclosed by the highlighted rectangle are assumed in the group.

You can now move the element group with the mouse or cursor keys, or you can use one of the alignment functions.

To deactivate the group, click the tool button again or click anywhere on the Panel Editor screen that is outside of the group.

5.4 The ActiveX control

ActiveX controls are Windows components that are basically comparable to well known controls in Windows. They do not have a fixed position in the system, however. Instead they are present in the form of .dll/.ocx files and can be integrated into a given software module if the need arises.

ActiveX controls can also be used in CANoe panels.

Precondition for data exchange using CANoe – ActiveX control:

- The control element file (.dll/.ocx) in question is known in the CANoe system.
To ensure this for the control must be created a configuration file (.ini) containing the relevant information of the control. A configuration file must be created manually or by using the ActiveX control wizard.
- The interfaces of the control are known.

Within the CANoe environment ActiveX controls can belong to two different hierarchies.

- ActiveX control elements (CANoe),
whose .ocx/.dll file and configuration file (.ini) are located in the directory {CANoe-Pfad}\exec32\ActiveX.
These control elements belong to the system and are available in each configuration.
- ActiveX control elements (panel)
whose .ocx/.dll file and configuration file (.ini) are located in the Panel directory.
These control elements belong to the corresponding configuration and are only available in it.

If ActiveX control elements are installed in both the Panel directory and in the CANoe directory with the same name, the control element in the Panel directory takes precedence, i.e. only the control element in the Panel directory will be loaded and displayed.

5.4.1 The ActiveX control wizard

The ActiveX control wizard supports you in the following tasks:

- Adding new ActiveX control elements to the CANoe environment
- Removing ActiveX control elements from the CANoe environment
- Assigning control interfaces for the data exchange with CANoe

- With the exception of entries for the alarm function, the control element configuration file can be created and changed with the aid of the wizard.

For additional information on installation, interface assignment and configuration of ActiveX control elements, please see the Online help.

5.5 The Hexadecimal Editor

The Hexadecimal Editor (Hex-Editor) serves as a display and input element for environment variables of the types *String* and *Data*. It represents environment variable values in multiple lines and allows you to edit even larger data quantities byte-by-byte.

Typical applications include editing or displaying larger data packets which occur with transport protocols on higher layers and its use as a Man-Machine Interface.

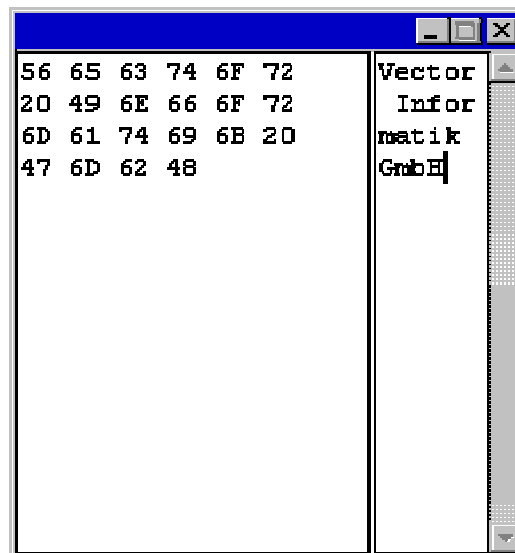



Figure 81: The Hexadecimal Editor

You can insert a Hexadecimal Editor in the panel from the menu item **Elements | Hex-Editor** or from the  button on the toolbar. After selecting the menu item the cursor changes its form to a crosshair with appended rectangle. With the left mouse button you confirm the selected position..

In **Configuration mode** the Hex-Editor can be called from the window border.

With **Type** you assign an environment variable with *data* or *string* type to the Hex-Editor. In the alphabetically sorted list you can select the desired environment variable.

In the **Layout** box you define the appearance that the Hex-Edit control should have. With **Columns per Line** you set the width of the element. Afterwards, the number of bytes corresponding to this width is shown.

With **View** you can choose among three different representation formats for the bytes of the environment variable:

- Hexadecimal and text fields (the two fields are shown next to one another)
- Only hexadecimal field
- Only text field

Hexadecimal representation consists of blocks, each with two hexadecimal digits. Each block represents one byte of the environment variable value.

Text representation consists of keyboard characters. Each byte of the environment variable is represented as a character in ASCII code. If no printable ASCII character corresponds to the byte value, a dot is shown.

You can have your entries accepted in the environment variable by pressing the <Enter> key, moving the cursor to another box, or choosing the **Accept** menu item.

Please note that in accepting environment variables of the *String* type, characters are only read up to a terminating null character (byte value zero). Characters that follow are not assumed in the variable. For environment variables of the *Data* type, on the other hand, the null character is not terminating.

In the **Text Attributes** box the font and background color can be configured. With a reset, the text attributes prescribed as default values are assumed.

Note: You can use the <Insert> key to toggle between the two input modes **Insert** and **Overwrite**.

5.6 Working with Bitmap Controls

With bitmap elements differentiation is made between two-stage and multistage elements. The first group consists of bitmap push button and two-stage bitmap switch/indicator. The second group includes multistage bitmap switch/indicator. In the configuration dialog you can assign a bitmap file to each of these elements.

You can create bitmap files with any bitmap editor (e.g. Paintbrush or Borland Resource Workshop). The formats for CANoe panel bitmaps are described in the following section.

Note: After you have edited a bitmap in the Panel Editor that is used by one or more elements, you can update it with **Reload Bitmaps**.

Examples of bitmaps that you can integrate directly into your panels or use as patterns for your own elements, are located in the directory `DEMO_CAN_CN\COLLECT`. The CANoe sample configuration `COLLECT.CFG` gives you an overview of this bitmap collection.

5.6.1 Bitmap File Format

A bitmap file for a two-way switch always consists of 3 rectangular partial bitmaps of identical height and identical width which are arranged horizontally. The inactive state is displayed in the rectangle on the left. This state appears in the panel, e.g. before

the measurement start or if no environment variable was assigned to the element. The logical state for OFF is displayed to the right of the first rectangle. It corresponds to the environment variable value 0. In the third position is the logical state for ON. This corresponds to the switch value of the environment variable.

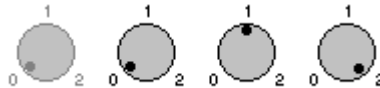


Figure 82: Bitmap for a switch with three states

A bitmap file for a n-stage switch consists of n+1 rectangular partial bitmaps of identical height and width arranged horizontally. Again, the inactive state is displayed on the left. The n switch values are located in the n partial bitmaps to the right of this.

To be able to integrate bitmap files more easily into the panels, it is advisable to make the number of stages discernible in the file names. The bitmaps supplied in the sample directories `DEMO_CAN_CN\AUTOMOT` and `DEMO_CAN_CN\COLLECT` show you a possible naming convention. The names of n-stage bitmaps here always end with "`_N`".

Note: When creating your own bitmaps the bitmap editor program should be set to save the bitmap as an **uncompressed Windows bitmap**. Special compressed bitmaps cannot be processed by some graphics drivers, which can lead to unpleasant visual effects or even system crashes.

5.6.2 Configuration of Bitmap Elements

In the configuration dialog of a bitmap element you open a file selection dialog with the **[File]** button. In this dialog you assign a bitmap to the element. Please note that your bitmap must always contain one partial bitmap more than the number of states that you entered under *States* above, according to the format described in section 5.6.1. The left partial bitmap represents the invalid state and is displayed, for example, if no valid environment variable or no valid signal was assigned to the element.

You activate or deactivate the element's transparency in the check box **Transparent Color**. For background bitmaps it is advisable not to activate the transparency, so that the drawing requirements are not increased unnecessarily.

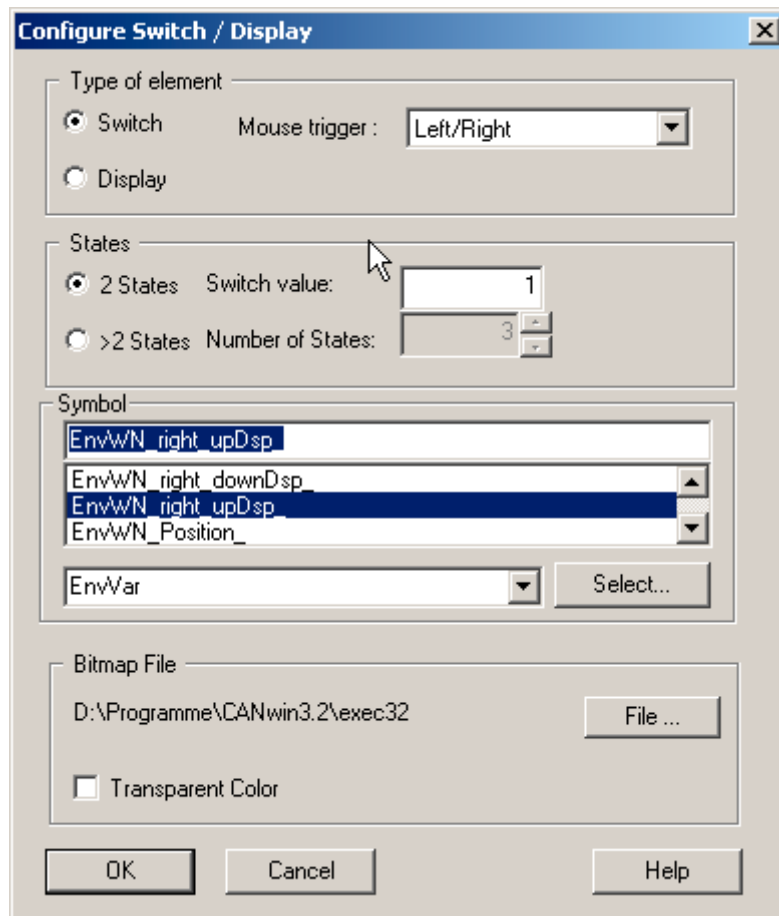


Figure 83: Configuration Dialog for Bitmap Controls

Note: To adapt the size of the control to the bitmap and number of states, select the menu item **Edit|Fit size** after you have exited the configuration with **[OK]**.

5.6.3 Color Resolution of Bitmaps

The quality of the display of bitmap elements depends on the configured monitor color resolution. If you are using bitmap elements to which high-resolution bitmap files are assigned (number of colors > 256) in your panels, the color resolution of your monitor should also be configured for the same color resolution for optimum color quality. As an alternative, in this case you could also set the `PaletteReduced=1` option in the `[PALETTE]` section of the `VPANEL.INI` file in the `EXEC32`-directory. This results in a reconfiguration of the system palette, which leads to improved display of high-resolution bitmaps. To achieve the same results in `CANoe`, set the same option in the `[VPANEL]` section of the `CAN.INI` file. Since the use of a reconfigured palette involves increased loading time, this option should only be activated if you are using high-resolution bitmaps and it is not possible to improve the monitor color resolution.

5.7 Overlapping

While creating the panels you can position the panels on top of each other. This sequence of the elements is called overlap sequence.

If you want to change the overlap sequence, you have to select an element and place it in the foreground or in the background.

Initially, when a new element is added to the panel it is placed in the foreground until the overlap sequence is modified or another element is added.

A marked element is always shown in the foreground, without influencing the actual overlap sequence of the elements.

Note: In test mode the overlap sequence is always displayed. You can update the display of overlaps based on the overlap sequence with the function **Update panel**.

5.7.1 Transparency Color

Bitmap elements whose transparencies are active allow background bitmaps to show through in areas of transparency color. Thus, the transparency color remains invisible. This allows the elements to assume any desired forms.

To draw a bitmap with transparency, fill the areas that are to be transparent with the transparency color of the panel. In the bitmap element's configuration dialog mark the check box **Transparent Color**.

The user can select the panel's transparency color in the dialog of the menu item **Panel Size, Name, Color and Fonts**. The default setting of the transparency color is pure blue (RGB(0,0,255)) and should only be changed if there is good reason to do so, e.g. to utilize created bitmaps with transparency color in multiple panels. The default bitmaps of the bitmap elements are based on the default transparency color setting.

Non-bitmap elements, such as push buttons, sliders, input boxes or text boxes, are not represented in transparency areas for bitmap elements.

5.7.2 Background Bitmaps

Background bitmaps are bitmaps, which are lying for a better layout in the background of all other elements.

5.8 Test Mode

With the menu item **Test | Start** or **Test | Stop** you can activate or deactivate test mode. In test mode you can monitor the functionality of the controls. However, all menu commands for configuring the panel are inactive.

In test mode all input and display elements react to user actions in the same way as during CANoe operation. Values of the assigned environment variables are set or value changes are displayed.

5.9 Panel Control

For complex CANoe models with many input and display panels it is often impossible to arrange them all simultaneously on the screen. On the one hand, the screen does not provide enough space. On the other hand, system resources are inadequate to simultaneously open an arbitrary number of panels.

However, usually only a small subset of all panels is needed at any given time for measurements and simulation runs. Panel control for CANoe therefore provides you with the option of grouping panels according to your work requirements.

With different **panel control buttons** you can select, which panel should be opened or closed during the measurement.

Each panel control button is configured with a list of the panels which are opened when the button is activated. Then during a measurement you can open specific panel groups from the control panel simply by pressing the appropriate panel control button.

In the **panel control configuration** dialog, under **Text** you enter the text you wish to label the button with. Under **Font Size** you can enter the size of the text characters. In the list below this you now enter the panels to be opened when the button is activated. All panels not entered in the list are closed automatically if you activate the check box **Close Other Panels**.

Note: In CANoe you can define permanent panels which are unaffected by panel control actions, i.e. they remain open even after a panel control button is activated, if they are not listed in the control list of the button.

With the **[Add]** button you can add new panels to the panel control list. Pressing **[Delete]** removes all marked panels from the list. Press **[Display]** to open the panels marked in the list for viewing.

Note: Since you may wish to use the same panel in various CANoe configurations, the panel positions are managed directly by CANoe. Therefore, they cannot be set in the Panel Editor.

6 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.

6.1 Overview

The universal applicability of CANoe results in large measure from its user programmability.

The CAN Access Programming Language CAPL is a C-like programming language, which allows you to program CANoe 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 CANoe program variations/ options support the CAPL functions. The following program variations offer additional CAPL functions:

- CANoe.LIN resp. DENoe.LIN
- CANoe.MOST resp. DENoe.MOST
- CANoe.J1939

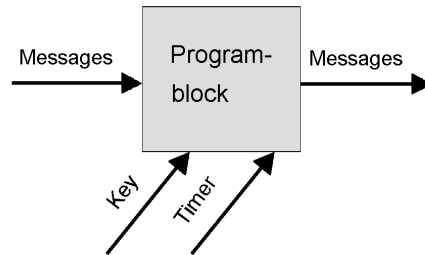
You can also write programs for problem-specific analysis of data traffic with CAPL, or you can program a gateway - a connecting element between two buses - to exchange data between different CAN buses.

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.

6.1.1 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 CANoe 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()`.

6.1.2 Integration of CAPL Programs

A CAPL program can be inserted in the measurement setup at all hotspots and also directly at the bus symbol in CANoe's simulation setup. 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 in CANoe's measurement setup can indeed generate messages, but it can-

not 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 CANoe's simulation setup 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 in 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 CANoe's simulation setup.

6.1.3 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 CANoe exclusively.

6.1.4 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`).

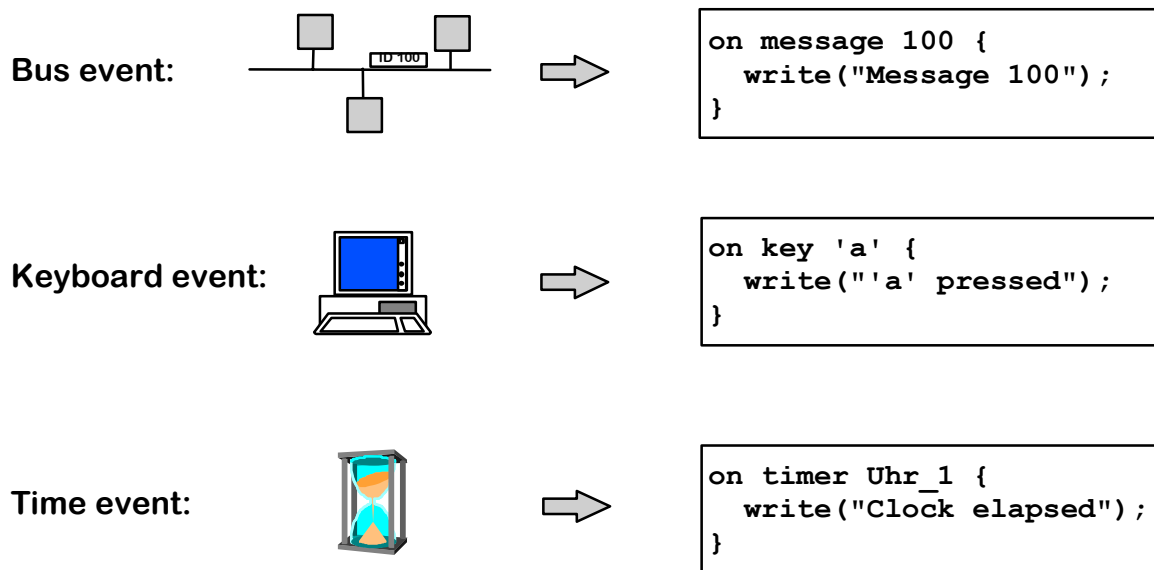


Figure 84: Examples of CAPL Event Procedures⁶

A CAPL program consists of two parts:

1. Declare and define global variables
2. Declare and define user-defined functions and event procedures

6.2 CAPL Browser

A special Browser is integrated in CANoe 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.

⁶ 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.

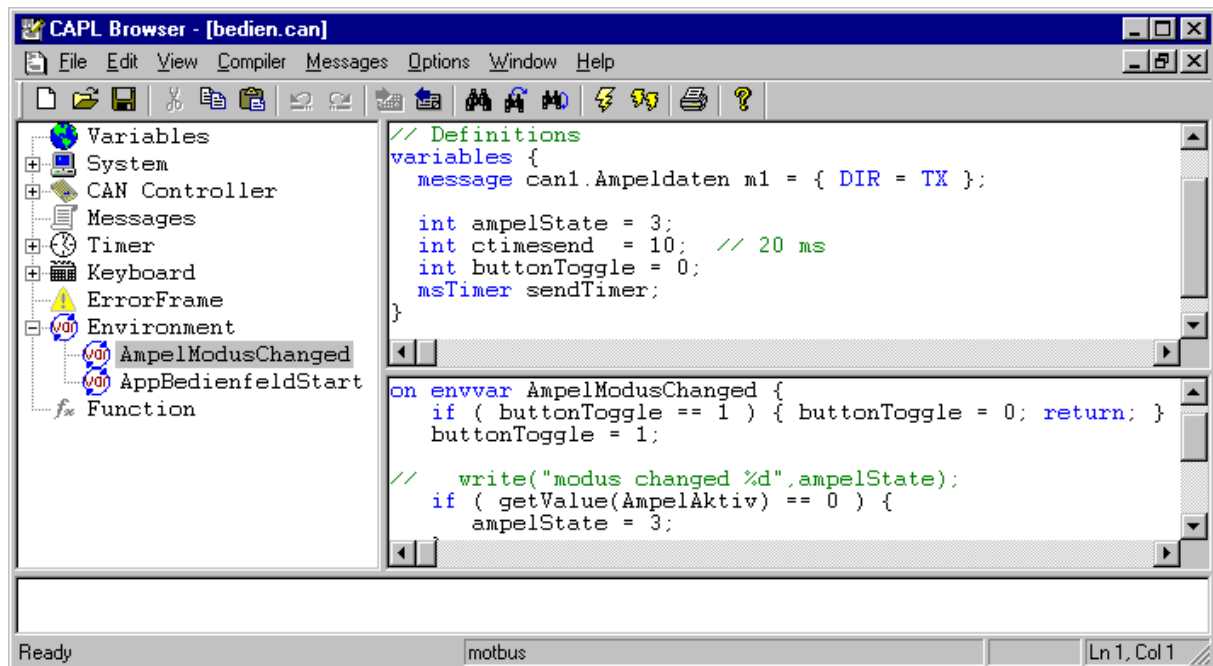


Figure 85: The Browser screen

A special Browser is integrated in CANoe 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.

6.2.1 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 CANoe, 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 or in CANoe's simulation setup, the file `PARBROW.INI` with all start parameters is automatically generated and made available to Browser. Therefore, always start Browser from CANoe, 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 (drag-and-drop). The file is then displayed in a new window.

6.2.2 Browser Window

A Browser window is subdivided into up to four sub-windows, so-called "Panes".


At the upper left is the Browser tree, which contains the CAN event types as drop-down nodes. These nodes each contain 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 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 6.2.6 you will learn how to import such CAPL programs in Browser format.

6.2.3 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.

6.2.4 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.

6.2.5 Access to the Database

Provided that you open Browser directly from the measurement setup or CANoe's simulation 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.

6.2.6 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.

6.2.7 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 or the simulation setup. That is, normally you should not change the option settings here.

7 Option .CAN

In this chapter you get a short introduction in CAN specific features of CANoe.

For the CANoe Tour and various examples the bus system CAN is underlied.

In this chapter, you get some additional information about the specific features of Option .CAN.

Note: Please refer to the online help for further details.

7.1 The Trace Window of Option .CAN

Configuring the Trace window you can (also) choose CAN specific fields to be displayed.

For Option .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

7.2 The Bus Statistics Window of Option .CAN

The Bus Statistics Window (also) displays CAN specific data and values.

For Option .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

8 Option .LIN

In this chapter you get a short introduction in CANoe.LIN resp. DENoe.LIN.

The most important and most used functions, features and applications are introduced here.

Note: Please refer to the online help for further details.

8.1 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.

8.2 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.

8.3 LIN Simulator

In addition to simulating CAN networks, CANoe can also operate LIN networks without hardware. Receiving and transmitting frames is simulated as for CAN.

8.4 LIN Specifications

CANoe.LIN supports the LIN specifications 1.1 and 1.2.

With the new database attribute "LinProtocolVersion" you can switch between the LIN specifications.

8.5 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 1.1 and 1.2.
- 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.

8.6 Trace Window for Option .LIN

Configuring the Trace window you can (also) choose LIN specific fields to be displayed.

For Option .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). 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!
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 LINda 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

8.7 The Bus Statistics Window for Option .LIN

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

9 Option .MOST

In this chapter you get a short introduction in CANoe.MOST resp. DENoe.MOST.

The most important and most used functions, features and applications are introduced here.

9.1 Installation Procedure

The CANoe/DENoe Option .MOST interfaces to the Optolyzer Box of Oasis through a standalone ActiveX control called 'Optolyzer Control' which doesn't need a license for the Optolyzer Professional Software.

9.1.1 Prerequisites

The CANoe/DENoe Option .MOST is so far tested under Windows NT 4.0 SP6, Windows 2000 and Windows XP. It works also on Windows 98 and Windows ME. It will not work on Windows 95 because the ActiveX Control doesn't work on this operating system.

You will need

- a CANoe/DENoe Option .MOST license, which is bound to some CAN HW, e.g. CANcardX,
- an Optolyzer Box (Firmware Version 2.50 or higher),
- the 'Optolyzer ActiveX Control' (incl. new license number for the Optolyzer),
- CANcardX driver version 3.4 for optimal time synchronization.

9.1.2 Procedure

- Install the Optolyzer ActiveX Control on your PC. The appropriate installation program for your operation system is provided in the folder `\Drivers\Optolyzer\OptoControl` of the CD. Since CANoe was developed and tested with the OptoControl versions from the CD, we strongly recommend to install this driver versions.
- Set the COM Port Settings for all COM Ports used with an Optolyzer Box within the Windows Control Panel, to 115200,8,n,1,Hardware
- Install Option MOST in a directory of your choice
- Enable 'Synchronize CAN hardware' on the global settings section in the configuration dialog of the CAN driver
- Enter the license code for the Optolyzer ActiveX Control in the **[Install]** tab of the MOST Hardware Configuration dialog (Menu: **Configure | MOST hardware configuration...**)

If Optolyzer Professional is installed, the license number for the ActiveX control may be entered via this tool and will then be stored in the windows registry. Multiple license numbers can be stored in this way and will all be checked while opening the connection to the Optolyzer box.

9.2 Profile

There are some configuration parameters for the Optolyzer in the profile CAN.INI in the executable directory of CANoe/DENoe.

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
```

9.3 Timestamps

CANoe 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.

9.4 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.

9.5 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 CANoe 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.

9.6 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 CANoe 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.

9.7 Database Support

CANoe.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 CANoe.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 or to assemble frames with the Interactive Generator for MOST.

9.7.1 Function Catalogs in XML

The function blocks of MOST devices are described in XML Function Catalogs. CANoe.MOST imports these function blocks from XML files. The structure of the XML Function Catalog file is described in the Document Type Definition (DTD). Look at the beginning of the XML Function Catalog file to determine the name of the DTD file. It should be part of the tag `<!DOCTYPE>`. To work properly the XML Function Catalog has to be consistent with Version 6.03 of the DTD of the MOST Cooperation. The latest XML Function Catalog as well as the DTD is available in the member area of the MOST Cooperation website (<http://www.mostcooperation.com>).

9.7.1.1 Using XML Function Catalogs in CANoe/DENoe

Similar to the usage of CANdb databases, the XML Function Catalogs can be assigned to individual busses in the System View window of the Simulation setup. Therefore, right-click on **Databases** in the System View window. Select **Add** in the context menu. The **Open File** dialog appears, where the XML file can be selected (choose file type *.xml).

CANoe/DENoe starts the import of the XML file as soon as the file is assigned to the bus.

9.7.1.2 Setting disassembly mode

With CANdb and XML there are two sources for the description of MOST messages. The key `DisassemblyMode` ([MOST] section) in the file `CAN.INI` (`Exec32` directory of your CANoe installation) controls which database should be used for message disassembly in the Trace Window. The following modes are available:

- 0 CANdb only
- 1 XML Function Catalogs only
- 2 CANdb, XML Function Catalogs
(if CANdb does not describe the message, the XML Function Catalog will be used)
- 3 XML Function Catalogs, CANdb
(if the message is not described in the XML Function Catalog, CANdb will be used)

9.7.1.3 Validation

During the import, CANoe parses the XML file and validates its structure against the DTD. In cases the XML does not match the DTD or the DTD file is unavailable the import will stop.

However, to support as many function catalogs as possible, CANoe allows the usage of XML files without DTD. Therefore, the following steps have to be performed:

- Make sure, the DTD file is not referenced within the XML catalog by removing the `<!DOCTYPE>` tag at the beginning of your XML file.
- Disable the validation by setting the entry `FCatImportValidate` to 0 in the CAN.INI file ([MOST] section):

```
[MOST]
FCatImportValidate = 0
```

Note: Using XML catalogs without any DTD or without the official DTD of the MOST Cooperation might lead to unpredictable import results!

9.7.1.4 Import Errors and Warnings

The XML import of CANoe writes logical and semantical errors and other inconsistencies in the XML Function Catalog into an ASCII file if the entry `FCatImportLog=1` is set in the [MOST] section of the profile CAN.INI. The name of the ASCII file is determined by the name of the XML file concerned (same directory, suffix `.xml` replaced by `.log`).

9.7.1.5 Using more than one XML Function Catalog

CANoe.MOST can use several XML Function Catalogs simultaneously.

If a MOST message is defined in more than one XML Function Catalog, the description with the highest precision will be used. The precision of a message description is internally defined as follows (4 = highest precision):

- 0 FBlock of message is not described
- 1 only FBlock of message is described
- 2 message is described up to 'Function'
- 3 message is described up to 'Operation'
- 4 at least one parameter of the message is described

If a message is described in XML Function Catalogs and in CANdb databases, CANoe.MOST uses the description according to the disassembly mode:

- Disassembly mode 2: If a message is described in a CANdb database, this description will be used.
- Disassembly mode 3: If the message description in the XML Function Catalog is of precision 3 or 4, this description will be used. Otherwise the description in the CANdb database will be used.

9.7.1.6 Reading and Rereading XML files

Importing a XML Function Catalog needs some time according to the size of the XML file. It needs some memory during parsing as well (approximately three to five times the size of the XML file). Therefore CANoe.MOST holds all information of the XML file in memory after parsing (the amount of memory needed after the parsing process is finished is approximately one to three times the size of the XML file).

This memory is only released when

- there is no XML Function Catalog assigned to the current configuration, or
- a XML Catalog is removed from the configuration.

The XML Function Catalog will be reread from file when

- a new configuration is loaded with another XML Function Catalog, or
- a new XML file is assigned to a bus in the Simulation Setup (CANoe/DENoe only), or
- an assigned CANdb database is saved with a changed MostXMLFile attribute (CANalyzer/DENalyzer only), or
- a CANdb database with a new MostXMLFile attribute is assigned (CANalyzer/DENalyzer only), or
- at start of measurement if the content of the XML file was modified since the last import.

During XML Function Catalog import it is not possible for CANoe to process commands therefore a spinning hour class will be displayed.

9.7.2 CANdb++

If a new database for MOST is needed, the best way to generate one is to use MostTemplate.dbc. You can choose this template by creating a new database within CANdb++ (menu File, Create Database). By starting with a copy of this template all settings needed for a MOST database are prepared correctly.

Another possibility is to generate a database from a XML file. Therefore, the import tool XML2DBC is supplied with the Option MOST. The tool imports XML descriptions of function blocks compliant to the DTD of the MOST cooperation. This utility and further documentation can be found in the `Exec32\Util_MOST` folder of your CANoe.MOST installation.

9.7.3 CAPL

CAPL only supports CANdb databases so far.

9.7.4 Frame

The database support is based on the structure of the MOST frames described in the MOST Function Catalog. According to this catalog the data section of every MOST frame has the following structure:

Byte	0	1	2 + High-Nibble(3)	Low-Nibble(3)	High-Nibble(4)	Low-Nibble(4)	5–16
Name	FBlockID	Instance-ID	FunctionID	Operation-Type	TelID	TelLen	Parameters
Numeric	31	00	201	0	0	6	...
Symbolic	Audio-DiscPlayer	00	Time-Position	Set	0	6	Pos, ...

Table 1: Layout of function catalog message

9.7.4.1 Lookup Key

The data structure in the bytes 5-16 is identified uniquely if FBlockId, FunctionId and OperationType are known. In order to associate a MOST frame with a database entry in CANdb, an identifier or lookup key is needed. By combining the values of FBlockId, FunctionId and OperationType to an identifier, this unique association between database entry and data structure is accomplished.

The ID for a MOST frame therefore is calculated as follows:

$$\text{ID} = (\text{FBlockId} \ll 16) + (\text{FunctionId} \ll 4) + \text{OperationType}$$
$$\text{ID} = (\text{msg}[0] \ll 16) + (\text{msg}[2] \ll 8) + \text{msg}[3] = 0x312010 \text{ (for the example above)}$$

9.7.4.2 Message Name

Additionally each database entry for a message can be identified by a unique name with a maximum length of 32 characters. These names can consist of any letter 0...9, a...z, A...Z and the underscore. For easy navigation between name and Identifier a concatenation of FBlockName, FunctionName and Operation is recommended, where any part shall be abbreviated so that the maximum length shall not be exceeded.

$$\text{Name} = \text{FBlockName} + \text{'_'} + \text{FunctionName} + \text{'_'} + \text{Operation}$$

9.7.4.3 Message Attributes

There are attributes assigned to every database entry for a MOST frame. The attributes FBlock, Function and OpType are used for the disassembly in the trace window. They allow to circumvent the limitation of the name length to 32 characters and should be set to the original names defined in the function catalog.

Any function in the catalog may belong to one of the function classes Method or Property. The disassembly of the OperationType depends on to which class the function belongs to. For a correct disassembly the attribute FunctionClass must be set to the appropriate value for each message. The default value is property.

For later use the attributes MostSrcAdr, MostDestAdr and InstId are provided. These attributes may be set to the typical value for the concerned message.

9.7.4.4 Parameters

For the disassembly of parameters mapped on MOST frames, signals can be placed on the corresponding database entry. For the exact procedure see the online help of CANdb++.

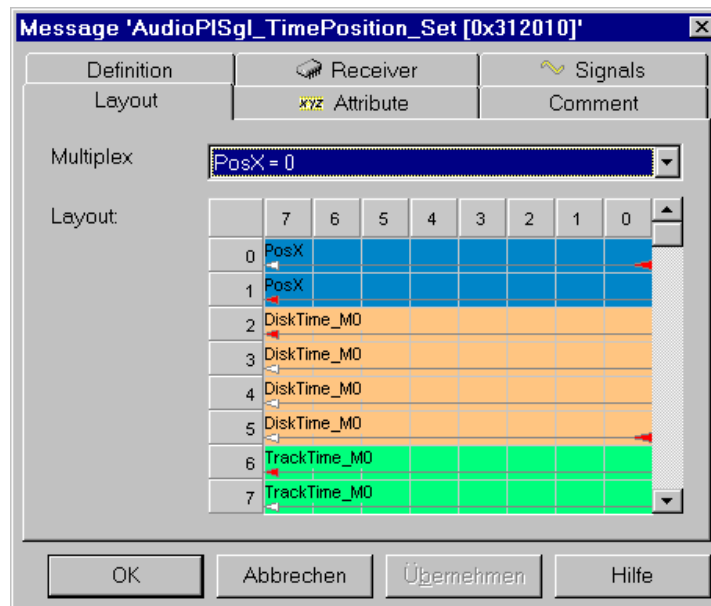


Figure 86: Signal layout of CANdb++ message

The counting of bytes for the layout view of the message starts with 0 at byte 5 of the MOST frame. This has to be taken into account during placement of signals onto a database entry for a MOST message. The first parameter 'PosX' shown in Table 1 has a start bit of 0 and is shown in byte 0 of the layout view, although it is placed in the bytes 5 and 6 of the MOST frame (see Table 1).

In this release CANdb++ supports parameters with a fixed length only. Parameters where the length is determined at runtime, e.g. with zero terminated strings, are not supported, because the position of the following parameters cannot be determined at design time.

In addition, only signals up to a length of 12 bytes and a placement within the bytes 5 to 16 can be defined. Segmented messages as through the application message service are not supported in this release.

9.7.5 HW Configuration

The data structure described above relates only to the application layer of a MOST system. It does not contain any information about the HW structure of the MOST ring. In MOST nodes the NetServices translate the addressing from the application layer addresses (FBlock, FunctionId,...) to physical addresses. In the recent release of CANoe Option MOST there's no NetServices support available.

The translation of physical addresses can be achieved by adding network nodes to the database. Each network node has the attribute 'MostNodeAdr'. CANoe.MOST translates the physical address into a symbolic device name by matching the address with the value of the node attribute and displaying the name of the corresponding network node.

The same procedure allows the translation of group addresses into group names. For each group address a network node with a suitable name could be entered into the database (attribute 'MostGroupAdr').

9.8 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.

9.9 Trace Window

For MOST frames the trace windows offers the following display options:

Name	Description
Timestamp	Synchronized timestamp
Channel	Channel information with the prefix 'M'
Identifier	Lookup key as described in chapter 9.7
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 9.7)
Dir	Received frame (Rx), transmitted frame (Tx) or requested transmission (TxRequest)
Data	17 Byte application data
RType	The message type can be Normal, RemRead, RemWrite, Alloc, Dealloc or GetSource. This column does not give any information about the addressing mode any more.
Source	Symbolic mode: Device name, if available (see chapter 9.7) Numeric mode: Source Address

Destination	Symbolic mode: Device name, if available (see chapter 9.7) Numeric mode: Destination Address
Original Time	Original timestamp generated by the hardware interface, with an offset of 0 at measurement start (see also chapter 9.3)
Type	Type of MOST event (Normal, RemRead, RemWrite, Alloc, Dealloc, GetSource), (Prepared to handle MOST packets and AMS messages)
Disassembly	Symbolic interpretation of MOST frames, according to the associated database, including FBlock, FunctionName and Operation-Type. The remaining bytes of the frame will be displayed numerically
State	State information of the received or transmitted message: bit0 (valid for Dir=Rx only): 0 - bus not active (NAc), 1 - bus active bit1 (valid for Dir=Rx only): 1 - unlock within this message (Unl) bit6 (normal and RemRead/Write Tx only): transmission result: 0 – OK, 1 – transmission failed (TxF) (meanings correspond with the description in the online help of the OptoControl)
AckNack	Acknowledgement information of the received or transmitted message: bit0 set: no response (NoResp) (Spy message only) bit1 set: valid receipt (Valid) (Spy message only) bit2 set: CRC Error (CRCError) (Spy message only) bit3 set: receive buffer full (RxBfFull) (Spy message only) bit4 set: acknowledged (Ack) (normal and RemRead/Write Tx only) bit5 set: not acknowledged (NAck) (normal and RemRead/Write Tx only) (meanings correspond with the description in the online help of the OptoControl)
CRC	CRC code (see online help of the OptoControl)
ASCII	ASCII dump of the 17 data bytes of the MOST frames
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.

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 TellId, 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.

9.10 Busstatistic Window

The busstatistic window shows some statistical information like frame/s, frames(total), number of changes of light&lock and so on. These values are calculated by the Vector MOST driver.

For a channel operated in node mode, only the frames addressed to this node are included in the calculation. However the values are calculated before any filter or other CAPL node can influence the number of the received frames.

The sampling rate with which the statistic analysis is processed may be configured in the profile CAN.INI:

```
[MOST]
// Sampling rate of bus statistic (value in ms)
// -1: Off, 100-.. valid values (default=1000)
Statistic = 1000
```

9.11 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 9.7), 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.

9.12 CAPL

For easy use, the MOST commands are integrated in CAPL.

9.12.1 Initialization of Message Variables

There are two types of message variables for MOST.

`mostMessage...` variables describe function catalog conform MOST frames.

`mostRawMessage...` variables describe MOST frames, which do not fit into the function catalog structure (e. g. RemRead, RemWrite, Alloc, Dealloc, GetSource).

Read the online help of CANoe to see how message variables are initialized, applied and send on the network.

9.12.2 Selectors

The selectors provide access to the data of message and event variables.

Keyword	Description	Range/Valid values	Access limitations
for mostMessages and mostRawMessages:			
DLC	Data Length Code (TelLen for mostMessages)	0...0xF	
DIR	Direction of transmission	Tx, Rx, TxRequest	
SA	Source address	0...0xFFFF	
DA	Destination address	0...0xFFFF	
MsgChannel	Transmission channel	1...4	
MsgCRC	CRC code (see online help of the OptoControl)	0...0xFF	read only, Spy
ACK	Acknowledgement information (see chapter 9.8)	0...0xFF	read only, Spy
Most_RType	Message type (Normal, Rem-Read, RemWrite, Alloc, Dealloc)	0...0x4	
Most_State	Status information (see chapter 9.8)	0...0xFF	read only
Most_IsSpy	Set to 1 for spy messages	0, 1	read only
BYTE(idx)	Provides access to the data bytes of the MOST frame. mostMessage: $0 \leq \text{idx} \leq 11$ mostRawMessage: $0 \leq \text{idx} \leq 16$ (see table below)	0...0xFF	
Time	Synchronized time (unit: 10µs)		read only
MsgOrigTime	Unsynchronized hardware time stamp (unit: 10µs)		read only
for mostMessages only:			
ID	Message ID (CANdb lookup key, see 9.7.4.1)	0...0xFFFFF	
Most_InstanceID	Instance ID	0...0xFF	
Most_TelID	TelID	0...0xF	
for mostLightLockError			
Most_Light	Light state of controller	0, 1	read only, mostLightLockError
Most_Lock	Lock state of controller	0, 1	read only,

			mostLightLockError
Most_Error	Error state of controller	0, 1	read only, mostLightLockError
Time	Synchronized time (unit: 10µs)		read only
MsgOrigTime	Unsynchronized hardware time stamp (unit: 10µs)		read only

Byte contents of mostRawMessages (see data sheet of MOST transceiver OS8104):

Byte	RemoteRead RType=1	RemoteWrite RType=2	Alloc RType=3	Dealloc RType=4	GetSource RType=5
0	rsvd	rsvd	rsvd	rsvd	rsvd
1	MAP	MAP	Request	CL	CL
2	rsvd	LENGTH	rsvd	rsvd	rsvd
3	D0	D0	Answer1	Answer1	rsvd
4	D1	D1	Answer2	0x0	rsvd
5	D2	D2	P0	rsvd	rsvd
6	D3	D3	P1	rsvd	NPR
7	D4	D4	P2	rsvd	rsvd
8	D5	D5	P3	rsvd	GA
9	D6	D6	P4	rsvd	NAH
10	D7	D7	P5	rsvd	NAL
11	rsvd	rsvd	P6	rsvd	rsvd
12	rsvd	rsvd	P7	rsvd	rsvd
13...16	rsvd	rsvd	rsvd	rsvd	rsvd

9.12.3 Event Procedures

9.12.3.1 on mostMessage

The event procedure `on mostMessage` is called on the receipt of a function catalog conform MOST frame (RType=Normal).

The key word `this` and the message selectors (see section 9.12.2) are available within the event procedures, to access the data of the message that has just been received. The command `output(this)` can be used for forwarding the message in a node chain.

The following examples show various modes of the event procedure `on mostMessage`:

```
on mostMessage *
```


React to all MOST messages

on `mostMessage 0x312010`

React to message `0x312010` (function block `0x31`, function `0x201`, operation `0x0`)

on `mostMessage AudioPlayer_TimePosition_Set`

React to message `AudioPlayer_TimePosition_Set` defined in a CANdb database.

on `mostMessage MsgChannel1.*`

React to all messages received by channel 1.

on `mostMessage MsgChannel1.0x312010`

React to message `0x312010` if it is received by channel 1.

on `mostMessage 0x22104C,0x312010-0x31201F`

React to messages `0x22104C` and `0x31201c` through `0x31201F`.

9.12.3.2 on `mostRawMessage`

on `mostRawMessage` is called on the receipt of other MOST frames. These are `RemoteRead`, `RemoteWrite`, `Alloc`, `Dealloc` and `GetSource`. See section 9.12.2 for the applicable selectors. The `command output(this)` can be used for forwarding the message in a node chain.

9.12.3.3 on `mostLightLockError`

On controller events on `mostLightLockError` is called. Light, Lock and Error events will be forwarded automatically.

9.12.4 Functions

9.12.4.1 `MostGetFBlockID`, `MostGetFunctionID`, `MostGetOpType`

```
long MostGetFBlockID(mostMessage msg)
```

```
long MostGetFunctionID(mostMessage msg)
```

```
long MostGetOpType(mostMessage msg)
```

These functions return the `FBlockID`, `FunctionID` or `OpType` of the frame.

Availability: Simulation Setup (any time), Measurement Setup

9.12.4.2 `MostSetFBlockID`, `MostSetFunctionID`, `MostSetOpType`

```
void MostSetFBlockID(mostMessage msg, long value)
```

```
void MostSetFunctionID(mostMessage msg, long value)
```

```
void MostSetOpType(mostMessage msg, long value)
```

Set the `FBlockID`, `FunctionID` or `OpType` of the frame.

Note: Since the `FBlockID`, `FunctionID` and `OpType` are part of the message ID this functions change the ID as well (see section 9.7.4.1).

Availability: Simulation Setup (any time), Measurement Setup

9.12.5 Hardware API

9.12.5.1 Optolyzer Operation Mode

```
long MostSetOptoMode(long channel, long optoMode)
```

Sets the operation mode of the Optolyzer to the specified value.

```
optoMode = 1: Spy Mode
```

```
optoMode = 4: Slave Mode
```

```
optoMode = 6: Master Mode
```

Refer to the OptoControl helpfile for further explanations of the values for optoMode. This function only works in the 'Prestart' section of the CAPL program.

```
long MostGetOptoMode(long channel)
```

Returns the operation mode of the Optolyzer connected to the given channel.

Availability: Simulation Setup (only in 'Prestart' section)

9.12.5.2 Node Mode

Setting Addresses of the Optolyzer

```
long MostSetOwnAdr(long channel, long grpadr, long nodeadr)
```

This function sets the node address and group address of the Optolyzer to the specified values.

Availability: Simulation Setup (at any time, not in StopMeasurement)

Retrieving Addresses of the Optolyzer

```
long MostGetNodeAdr(long channel)
```

```
long MostGetGroupAdr(long channel)
```

```
long MostGetNodePosAdr(long channel)
```

Return Node, Group or Node Position Address of the Optolyzer connected to the specified channel.

Availability: Simulation Setup (at any time, not in StopMeasurement)

Optolyzer Status

```
long MostGetLight(long channel)
```

```
long MostGetLock(long channel)
```

```
long MostSetLight(long channel, long mode)
```

Get and set the Optolyzer states.

Availability: Simulation Setup (at any time, not in StopMeasurement)

9.12.5.3 Spy Filter

Filter Mode

```
long MostSetHWFilter(long channel, long mode)
mode = 1: HW filter on
mode = 0: HW filter off
```

Effects the Optolyzer only when in spy mode.

```
long MostGetHWFilter(long channel)
```

Returns 1 if hardware filter is active, otherwise 0.

Availability: Simulation Setup (at any time, not in StopMeasurement)

AND Filter

```
long MostSetAndFilter(
    long channel,
    long arb,
    long targetAdr,
    long sourceAdr,
    long type,
    byte msg[17],
    long crc,
    long ack
)
```

Effects the Optolyzer only when in spy mode. See the online help of the OptoControl help file for information about how AND and XOR filter masks work together. The there described mask string is assembled appropriately from the parameters of the function 'MostSetAndFilter'.

```
long MostGetAndFilter(
    long channel,
    long arb,
    long targetAdr,
    long sourceAdr,
    long type,
    byte msg[17],
    long crc,
    long ack
)
```

Retrieves the status of the AND filter.

Availability: Simulation Setup (at any time, not in StopMeasurement)

XOR Filter

```
long MostSetXorFilter(  
    long    channel,  
    long    arb,  
    long    targetAdr,  
    long    sourceAdr,  
    long    type,  
    byte    msg[17],  
    long    crc,  
    long    ack  
)
```

Set the XOR filter. Effects the Optolyzer only when in spy mode.

```
long MostGetXorFilter(  
    long    channel,  
    long    arb,  
    long    targetAdr,  
    long    sourceAdr,  
    long    type,  
    byte    msg[17],  
    long    crc,  
    long    ack  
)
```

Retrieves the status of the XOR filter.

Availability: Simulation Setup (at any time, not in StopMeasurement)

9.12.6 Error Codes of CAPL functions

9.12.6.1 *kMostTxQueueFull* = -6

The transmit queue of 256 Messages is full, while trying to send another message.

9.12.6.2 *kMostWrongOptoMode* = -5

The function cannot be called in this optolyzer mode. Either trying to send messages in spy mode or configuring HW filter in node mode.

9.12.6.3 *kMostWrongThread* = -4

The function cannot be called from this part of the setup, due to the thread affinity of ActiveX.

Not all functions can be called from CAPL nodes in the analyzing setup

9.12.6.4 *kMostIllegalTime* = -3

The function was called in the wrong phase of the measurement. For example the optolyzer mode can only be changed in the 'prestart' function.

9.12.6.5 *kMostNoConnection* = -2

The Driver got no connection to the Optolyzer box or OptoControl. Check whether everything is installed properly, the Optolyzer box is connected to the PC and switched on. Check also whether the CAN.INI file configures the correct COM port and interface name for the connected Optolyzer.

9.12.6.6 *kMostInvalidChannel* = -1

The operation was requested for an invalid channel number.

9.12.6.7 *kErrDrvOK* = 0

No Error occurred.

9.13 Demo Configurations CANoe/DENoe

Demo configurations demonstrating MOST features can be found in the `De-mo_MOST_CN` folder of your CANoe/DENoe installation. Detailed descriptions of the demos are provided within the configuration comments (**File | Configuration comment...** menu).

9.13.1 MOSTSpy

This is a very simple configuration operating a single Optolyzer Box in spy mode, which allows to analyze a MOST ring very quickly.

If needed, a second channel can be activated (see 'channel usage') operated as default in master mode to provide a timing master for the MOST ring.

9.13.2 MOSTGeneral

This demo configuration has been developed and tested with the MOST starter kit:

- 2 Optolyzer Boxes
- 1 MOST MediaPlayer
- 1 MOST Speak

All devices have remained in the configuration in which they have been delivered.

If these devices are not present in the MOST ring, then the nodes 'MOST Speak' and 'MediaPlayer' in the simulation setup can be switched from 'real' mode to 'simulated'. By doing this, basic functionality of the real devices is simulated like 'start' 'stop' and the transmission of timing information.

9.13.3 MOST XML Catalog

The demo replays a logging file in offline mode to show the disassembly capabilities of the Trace window. Therefore the configuration imports two sample XML Function Catalogs. No real hardware is required for running the demo.

9.14 Known Problems & Trouble Shooting

9.14.1 Measurement Windows frequently disabled

In spite of a moderate bus load on the used CAN or MOST channels, the measurement windows frequently show a notification 'disabled' in the title bars.

Please check:

- Are the COM port settings in the windows control panel correct?
They must be 115.2 kBaud, 8 Bit, no parity, 1 Stop Bit, Hardware flow control

9.14.2 CANoe doesn't receive any MOST Frames

Please check at first:

- Does the Optolyzer have light&lock
- Are there MOST frames to receive
- Is the Optolyzer addressed (either in Spy Mode, or with correct node or group address)
The mode is configured either in the MOST hardware configuration dialog or in a CAPL program.
- In spy mode: Is the HW Filter switched off?
- Is there a message 'MOST x: Obsolete Firmware? (Version >2.41c required)' in the write window?
This indicates that the time synchronization couldn't startup. An obsolete firmware version of the Optolyzer box may be the reason.

If sending MOST frames and receiving Tx acknowledgements works well, but still no MOST frames are received, the firmware of the Optolyzer box may be obsolete. For usage of CANoe .MOST at least version 2.50 of the firmware is needed.

9.14.3 CANoe doesn't receive some MOST Frames

When running an Optolyzer box in node mode (Master, Slave), only messages sent to the node's address will be received. Furthermore, acknowledged frames will be processed, only. Not acknowledged frames can solely be detected in Spy mode.

While driving the Optolyzer box with high load (simultaneous transmission and reception of multiple messages), the receiver (Optolyzer) may need several send retries before it acknowledges a reception to the sender. Adjust the number of retries at sender site.

9.14.4 Warning '2 Tx receipt(s) got lost'

When sending MOST frames in quick succession through the Optolyzer box, the driver may lose Tx receipts for sending requests. Therefore, CANoe cannot affirm whether a frame has been sent or not. The warning 'N Tx receipt(s) got lost' will be written in the Write window as soon as an incoming Tx receipt indicates, that N former Tx receipts are missing.

In general, this warning occurs on send requests to a node address that does not exist in the MOST ring. Please check the destination address and reduce the sending frequency.

9.14.5 After several minutes CANoe doesn't receive or send any MOST Frames

After several minutes CANoe doesn't receive or send any MOST Frames. The CAN related features of CANoe still work stable.

Please check:

- Are the COM port settings in the windows control panel correct?
They must be 115.2 kBaud, 8 Bit, no parity, 1 Stop Bit, Hardware flow control
- There may be problems with the installation of the OptoControl ActiveX Control
Please try to reinstall the OptoControl. First deinstall OptoControl using Windows (Settings, Control Panel, Add/Remove Software). Ensure that there are no entries belonging to the OptoControl left in the registry (search for "*opto*"). Then install OptoControl again. Use the version provided on the CANoe CD according to the used operating system (on the CD under `Drivers\Optolyzer\OptoControl`).

9.14.6 Error 'Port COM1 busy for channel 1'

If the ActiveX can not open a connection to an Optolyzer box, this message appears at measurement start. The following reasons can cause this problem:

- No Optolyzer is connected to the COM port configured, or
- the Optolyzer is switched off, or
- the COM port is already used by another application.

If the Optolyzer was just recently connected to this computer or to CANoe Option MOST and is therefore started for the first time with CANoe, just try to start the measurement again or try switching the Optolyzer off and on again.

Connection problems to the Optolyzer may also be solved by a re-start of CANoe.

9.14.7 Error 'No valid license for channel 1'

CANoe cannot establish the connection to the Optolyzer interface box through the Optolyzer ActiveX Control because of an invalid license key. Please check:

- The license key must include the activation for the hardware and the ActiveX Control.
- The key has been entered correctly in the Install register of the MOST Hardware Configuration dialog. It must not contain white spaces or dashes.

9.14.8 Weird Spurious Events from MOST bus

In some rare cases there might be some spurious events from the MOST bus. This may include invalid timestamps or even random data. There are some plausibility checks performed on the timestamps which should detect these events. In order to avoid an impairment of the time synchronization the timestamps of these events will be set to the recent simulation time of CANoe. The events will be shown in the trace window and logging file and will be reported in the write window as 'weird time stamp'.

This problem seems to be provoked by a high overall load (CAN and MOST) or a slow PC. Please try to reduce the load by

- Disabling unused analysis functions of CANoe
- using HW filtering of the Optolyzer in spy mode
- activating acceptance masks in the CAN chip setup
- using a faster PC

This problem is difficult to reproduce and seems to be related with the ActiveX control. It will be investigated further and solved as soon as possible.

9.14.9 Transmit Acknowledgement

When using older versions of the Optolyzer ActiveX Control (versions before 2.1.1), transmit acknowledgements for the transmission of a MOST control frame do not have a hardware generated timestamp. So the 'origtimestamp' is set to zero and the synchronized timestamp is set to an estimated value depending on the time of reception of the acknowledgement.

These timestamps are shown with a trailing '?' in the trace window to indicate that these events are not exactly in chronological order with the other events.

9.14.10 Timestamps of CAN and MOST constantly drift apart

Time synchronization only works with CAN driver versions from 3.4 and later. Please install the drivers enclosed in this distribution, or visit our web site at <http://www.vector-informatik.de> to download the drivers.

Ensure that the check box 'Synchronize Hardware' is enabled in the Global Settings tab of the CAN Driver Configuration dialog.

9.14.11 Measurement stops after configuration of trace window

In some cases the measurement will be stopped immediately if the configuration dialog of a trace window is opened during measurement. This problem can be traced back to an improper installation of the ActiveX control OptoControl.

Please try to reinstall the OptoControl. First deinstall OptoControl using Windows (Settings, Control Panel, Add/Remove Software). Ensure that there are no entries belonging to the OptoControl left in the registry (search for "*opto*"). Then install OptoControl again. Use the version provided on the CANoe CD according to the used operating system (on the CD under Drivers\Optolyzer\OptoControl).

9.15 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/>.

10 Option .FlexRay

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.

10.1 Trace Window for Option .FlexRay

Configuring the Trace window you can (also) choose FlexRay specific fields to be displayed.

For Option .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

10.2 The Bus Statistics Window for Option .FlexRay

The Bus Statistics Window (also) displays FlexRay specific data and values.

For Option .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]

11 Index

!

! Load symbol · 53

<

<F10> key · 4

7

72005 · 82, 83

8

82527 · 82, 83

82C200 · 82, 83

A

Acceptance code · 83

Acceptance filtering · 56, 83, 84, 86

Acceptance filters · 81

Acceptance mask · 83, 84

Acknowledge · 83

ActiveX control · 149

ActiveX control wizard · 149

Activity indicator · 114

Alarm states · 147

Ambiguities · 113, 163

Analysis blocks · 54

Analysis branch · 55

Animate · 68, 69, 70, 99

Animation factor · 93, 94

Arranging controls · 147

Array · 162

ASCII · 66, 67

Averaging time · 55, 116, 119

Axis options · 103, 104

B

Background bitmaps · 154

Bar-type display · 115

Basic image · 117

Basic-CAN controller · 79

Baud rate · 111

Baud rate selection · 86

Baudrate · 81, 82

binary · 67

Binary · 66

Binary data · 146

Bit time · 82

Bit timing · 83

Bit timing · 82

Bitmap · 146, 151, 152

Bitmap editor · 151

Bitmap File Format · 151

Bitmap formats · 151

Bitmap switch · 147

Blocking filter · 56, 121

Branches · 8, 88

Break · 68, 69

Break condition · 68

Breakpoint · 68

Browser tree · 26, 161

BTL cycles · 81, 82

Burst · 129

Bus load · 52, 53, 79, 89, 109, 119

Bus load measurement · 119

Bus loading · 52, 55

Bus parameters · 81

Bus registers · 81

Bus statistics · 68, 85, 119

Bus statistics display · 53

Bus statistics information · 67, 119

Bus Statistics Window · 165

Bus Statistics Window · 194

Bus Timing Register · 82

Busstatistik Fenster · 168, 194

Button · 5

C

CallAllOnEnvVar · 59

CAN card · 79

CAN channel · 50

CAN.INI · 45, 53, 67, 69, 86, 157

CANcard · 79
 CANcardX · 79
 CANdb++ · 7, 33, 46
 CANoe without card · 94
 CAPL · 3, 36, 41, 59, 64, 76, 79, 87, 95, 122,
 136, 137, 138, 139, 140, 159
 CAPL Browser · 7, 33, 137
 CAPL compiler · 137
 CAPL Generator · 33
 CAPL node · 139
 CAPL program · 86, 95, 136, 157
 Card clock · 91
 Card driver · 87, 91
 Channel · 79
 Channel definition · 50, 51, 79, 80
 Check box · 5
 Chip allocation · 50, 80
 CIF files · 45
 Clipboard · 111, 161
 cnp file · 144
 Color quality · 153
 Color resolution · 153
 Colors in the Graphics window · 108
 Comment · 121, 163
 Comment box · 5
 Compilation time · 160
 Compile time · 162
 Compiler · 27, 46, 157, 160, 161, 162, 163
 COM-Server · 74
 Condition primitives · 70
 Configuration · 4, 5, 36, 50, 123, 132, 137,
 140, 141, 147, 154
 Configuration descriptions · 45
 Configuration file · 38, 44
 Configuration file formats · 45
 Configuration management · 45
 Configuration options · 37, 90, 109
 Context menus · 4
 Control · 59, 75, 145, 147, 151
 Control box · 5
 Control element · 144
 Control elements · 144, 145, 146, 154
 Control panel · 3, 33
 Control panels · 28, 56
 Controls · 145

Conversion
 Log files · 73
 Converter
 LDF to DBC · 167
 Copying blocks · 90
 Cycle time · 55, 115, 129
 Cyclic update · 55, 65

D

Data environment variables · 150
 Data flow · 8, 34, 37, 78, 89, 90, 121, 132, 139,
 140, 141, 158
 Data flow diagram · 8, 37, 43, 88, 89, 90
 Data loss · 52, 53
 Data reduction · 55, 68
 Data sink · 122
 Data source · 61, 67, 68, 70, 79, 89, 120, 122
 Data throughput · 119
 Database · 28, 41, 47, 50, 59, 140, 141, 144
 Database Editor · 33
 DBC
 LDF to DBC Converter · 167
 Deactivation function · 54
 Decrementer · 130
 Delay times · 85
 Demo version · 36
 Dialog box · 4
 Difference cursors · 106
 Difference measurement mode · 106
 Display elements · 144
 DLC · 77
 DPRAM · 76, 79
 Driver options · 67, 85
 Drop and drag · 161
 Drop down list · 5

E

End of measurement · 64, 102, 105
 Environment variables · 59
 Environment variable · 33, 56, 59, 67, 95, 100
 Environment variable filter · 140
 Environment variables · 2, 3, 27, 32, 59, 132,
 133, 140, 141, 152
 ERROR · 119

Error frame · 41, 64, 95, 119, 123, 129, 133
Error message · 50, 74, 75, 94
Error message · 65
Error Passive · 75
Error-Frame · 119
Evaluation branch · 55
Evaluation functions · 61
Export
 Log files · 73
Extended identifier · 95
Extended Identifier · 117

F

Fenster
 Busstatistik Fenster · 168, 194
File format · 161
File icon of the log file · 23
Fill level indicator · 53
Filter · 8, 37, 47, 89, 90, 122
Filters · 140
Fit · 117
Font · 86
Format entry · 27
Formats of configurations · 45
Function block · 90
Function generator · 125

G

Gateway · 156, 157
Generator block · 8, 43, 89, 121, 122, 123,
 124, 125, 126, 158
getValue() · 28, 31
Global options · 45
Grid · 108, 145
Grouping of controls · 147

H

Hardware configuration · 50
Help · 7, 84
hexadecimal · 83
Hexadecimal Editor · 150
Hexadecimal field · 151
Hexadecimal representation · 151

Hex-Edit control · 150
High-load operation · 52
Hotspots · 8, 89

I

Import of configuration descriptions · 45
Incrementer · 130
Indicator lamps · 144
Infinite loop · 91
INI file · 45, 53, 67, 69, 74, 86
INI-Datei · 157
Initialization of environment variables · 59
Input box · 5, 71, 146
Input control · 154
Integer variable · 26, 27
Interactive generator block · 127
Interface card · 8, 86
Interrupt · 77

K

Keyboard control · 131
Konfiguration
 LIN Testumgebung · 166

L

Layout · 86
LDF
 LDF to DBC Converter · 167
Line type · 103
Load operation · 52, 53
Load situation · 53
Load transition · 53
Log file · 132
 converting · 73
 exporting · 73
 file icon · 23
Logging
 time period of · 64
Logging Export · 73
Long-duration measurement · 65

M

Main memory · 65

Man-Machine Interface · 150
 Measurement configuration · 4
 Measurement cursor · 105, 106
 Measurement point · 104
 Measurement setup · 35, 89, 121, 122, 132, 138, 139, 140, 141
 Measurement start · 26, 43, 68, 79, 104
 Measurement value acquisition · 104
 Menu operation · 4
 Message attribute · 91
 Message display · 53
 Message Explorer · 47, 48, 162
 Message rate · 116
 Messages window · 161
 Mode signal · 125
 Mode-dependent signal · 125
 Moving blocks · 90
 Multiplex message · 125
 Multisignal mode · 100, 105
 Multistage switch · 146
 Multi-stage switch · 147

N

Network node · 27, 28, 86, 87
 Network nodes · 1, 2, 3, 36, 40, 86, 87, 136, 137, 138, 140, 141
 inserting · 137
 Node · 28, 87
 Node status · 137
 Node type · 137
 Nodes · 86, 87, 136, 137, 138
 Notebook · 75
 Numbering system · 45

O

Offline mode · 69, 70
 Online mode · 43, 61, 69, 70, 79, 89
 Operating instructions · 3
 Operating mode · 6, 92
 Optimization · 54, 109, 115
 Trace window · 99
 Options button · 5
 Output mode · 52, 55, 104
 output() · 139, 156

output(this) · 139
 Overflow · 76
 Overload situation · 53
 Overload symbol · 68

P

Palette · 153
 Panel · 3, 33, 59, 144, 145, 146, 147, 151, 154
 Panel configuration · 56
 panel control · 155
 Panel control · 59
 Panel editor · 3
 Panel Editor · 33, 56
 Panel Generator · 33
 Panel name · 145
 Panel size · 145
 Panels · 56
 Panes · 26, 161, 162, 163
 PARBROW.INI · 161
 PC card · 34, 44, 79, 83
 PC-card · 8, 70, 86, 90
 Peak display · 114
 Performance · 115, 119
 Performance optimization · 54
 Performance wizard · 54
 Phase model · 1, 2, 86, 92
 Point measurement mode · 105
 Pointer elements · 146
 Post-trigger time · 43, 64
 Power manager · 74
 Prescaler · 81
 Preselection · 140
 Pre-trigger time · 64
 Primitive · 70, 71
 Procedure template · 26, 27
 Procedures list · 161
 Program block · 156
 Program start · 8, 37
 Project directory · 44
 Project file · 44
 Pushbutton · 146, 147
 putValue() · 28, 32

R

Radio button · 5
Raw value · 130
Reaction times · 35, 55
Real bus · 136
Real channel · 50, 79
Real-time library · 34, 35, 52, 53, 54, 55, 56
Receive messages · 91
Receive time point · 79
Refresh · 55
Remote frame · 129
Replay block · 122, 132, 133
Representation formats · 45, 46
Representation parameters · 112
Reset · 69
Right mouse button · 4
Ring buffer · 34, 35, 52, 53, 54, 65
runError() · 162
Run-time error · 162
Rx attribute · 91, 116
Rx buffer overrun · 119
Rx error counter · 119

S

Samples · 82
Sampling point · 81
Sampling time point · 82
Scaling · 117
scalings · 107
Search condition · 70
Setup · 136
Shortcuts · 131
Signal · 47, 72
Signal curve · 103
Signal Explorer · 20, 48, 162
Signal identifier · 113
Signal response generator · 125
Signal selection dialog · 103
Signal value · 125
Signal-based Logging Export · 73
Simulated bus · 136
Simulated nodes · 138
Simulation · 1, 3, 86, 92, 93, 94

Simulation configuration · 34, 36
Simulation dialog · 93
Simulation setup · 4, 6, 7, 35, 50, 59, 86, 87, 92, 122, 132, 136, 137, 138
Simulation setup window · 86
Single-signal mode · 100, 105
Single-step mode · 69, 70
SJA 1000 · 79, 82, 83
SJW · 82
Slave mode · 93
Slider · 144, 146
Slow motion · 69
Stack overflow · 162
Standard deviation · 118
Standard Layout · 86
Start · 69
Start bit · 130
Start options · 34
Start value · 59
Statistical report · 52
Statistics report · 55, 111, 116
Statistics window · 116
Status bar · 137
Step · 69
Stop · 64
stop() · 43, 68
String environment variables · 150
Switch · 147
Switch value · 145, 147, 152
Switches · 144
Symbol selection dialog · 163
Symbolic databases · 46
Symbolic triggering · 72
Synchronization edge · 82
Synchronization jump width · 82, 83
System color palette · 153
System directory · 38
System loading · 115
System messages · 111

T

Test mode · 154
Testing · 1, 3
Testumgebung
Konfiguration · 166

Text box · 146
Text editor · 161
Text field · 151
Three-stage switch · 147
Time base · 70
Time basis · 70
Time resolution · 74, 94
Time stamp · 91, 97
Time window · 62, 64
Timeout · 75, 78
Timer · 156
Toolbar · 27, 102, 160, 162
Trace window · 94, 164, 193
 Optimization · 99
Transmission of messages · 127
Transmission time · 92
Transmit branch · 122
Transmit list · 128
Transmit protocols · 32
transparency color · 154
Transparency color · 145
Transport protocols · 150
Trapezoidal function · 126
trigger · 157
Trigger · 60, 61, 62, 64, 67, 68, 70
Trigger block · 68
Trigger condition · 60, 64, 70, 122, 123, 124,
 128, 129
Trigger time point · 129
Trigger type · 64

trigger() · 157
Triggerblock · 68
Triggering · 64, 157
Tx attribute · 91, 116
Tx error counter · 119
TXREQUEST · 85
TxRq attribute · 91

U

Undo function · 81

V

Value table · 125
Virtual channel · 50, 79

W

Window
 Trace window · 164
Window control · 43
Window layout · 86
Working directory · 44
write() · 111

Z

Zoom · 117
Zoom mode · 90