

User Manual

CANoe

(including Notes on Installation and all Options)

Version 7.5

English

Imprint

Vector Informatik GmbH
Ingersheimer Straße 24
D-70499 Stuttgart

The information and data given in this user manual can be changed without prior notice. No part of this manual may be reproduced in any form or by any means without the written permission of the publisher, regardless of which method or which instruments, electronic or mechanical, are used. All technical information, drafts, etc. are liable to law of copyright protection.

© Copyright 2010, Vector Informatik GmbH. Printed in Germany.

All rights reserved.

80419

Table of contents

1	Introduction	7
1.1	About this user manual	8
1.1.1	Access helps and conventions	8
1.1.2	Certification	9
1.1.3	Warranty	9
1.1.4	Support	9
1.1.5	Registered trademarks	9
2	Installation	11
2.1	General	12
2.2	System requirements	12
2.3	Installation requirements	12
2.4	Installation procedure	13
2.5	Notes on activating a software based license	13
2.6	Vector USB dongle	14
2.7	MOST	15
2.7.1	MOST25: Use with Optolyzer Box	15
2.7.2	MOST150: Use with Optolyzer G2 3150o	16
2.7.3	MOST50: Use with Optolyzer G2 3050e	16
2.8	Further CANoe options	17
2.9	Switching language versions	17
2.10	Running the test	18
2.11	Troubleshooting	18
2.11.1	Software-specific error messages	18
2.11.2	Hardware-specific error messages	19
3	Basics	21
3.1	Introduction to CANoe	22
3.2	Tips for using CANoe	24
3.3	Overview of the programs	26
3.4	CANoe architecture	27
3.5	Particularities of the demo version	28
4	CANoe Tour	29
4.1	Overview	30
4.2	Preparations	30
4.3	Setting up the bus	32
4.4	Transmitting data	33
4.5	Evaluation windows	37
4.6	Working with symbolic data	40
4.7	Analysis of signal values in the Data window	41
4.8	Analysis of signal responses in the Graphics window	43
4.9	Use of the database in transmitting messages	44

4.10	Logging a measurement	45
4.11	Evaluating a log file	46
4.12	Creating a CAPL program	47
4.13	Simulation of distributed systems in CANoe	49
4.13.1	Creating the database	50
4.13.2	Creating panels	51
4.13.3	Creating network node models	53
5	Applications	55
5.1	Overview of the important elements	56
5.2	Simulation/Simulation setup	58
5.2.1	Working in the simulation setup	58
5.2.2	Simulation mode	58
5.2.3	Message attributes	59
5.2.4	System verification	60
5.3	Measurement/Measurement setup	61
5.4	Working with configurations	61
5.5	Working with databases	62
5.5.1	Use of multiple databases	64
5.5.2	Resolving ambiguities	65
5.5.3	Checking for consistency of symbolic data	65
5.6	Working with multiple channels	65
5.6.1	Channels in online mode	66
5.6.2	Channels in simulation mode	66
5.6.3	Channels in offline mode	67
5.7	Working with panels and symbols	67
5.8	Logging and evaluation of measurement files	67
5.8.1	Triggers	68
5.8.2	Data analysis	70
5.8.3	Data export and conversion	70
5.9	Test functionality in CANoe	70
5.9.1	Test Feature Set (TSF)	71
5.9.2	Test Service Library (TSL)	73
5.10	Diagnostics functionality in CANoe	74
5.10.1	Diagnostic Feature Set (DFS)	74
5.11	CANoe Realtime	75
5.12	Standalone Mode	75
5.13	Macro Recorder	76
5.14	Step Sequencer	76
5.15	COM Server	76
5.16	Troubleshooting	77
5.17	List of error messages to the CAN interface	78
6	Windows	81
6.1	Desktop concept	82
6.2	Window management	82
6.3	Simulation Setup window	83
6.4	Measurement Setup window	84
6.5	Trace window	87

6.6	Graphics window	88
6.7	State Monitor window	90
6.8	Write window	91
6.9	Data window	91
6.10	Statistics window	92
6.11	Statistics Monitor window	94
6.12	Diagnostics Console	95
6.13	Fault Memory window	96
6.14	Test Setup window	96
7	Blocks and Filter	99
7.1	Overview	100
7.2	Interactive Generator Block (IG)	101
7.3	Replay block	102
7.4	Trigger block	102
7.5	Filter and environment variable filter	103
7.6	Channel filter	103
7.7	CAPL nodes in the simulation setup	104
7.8	CAPL nodes in the measurement setup	104
8	Panel Designer	105
8.1	Overview	106
9	CAPL	109
9.1	CAPL basics	110
9.2	CAPL Browser	112
10	CAN	115
10.1	Overview	116
11	LIN	117
11.1	Preliminary note	118
11.2	How to create a LIN description file	118
11.3	How to create a CANoe.LIN configuration	118
11.4	How to simulate and analyze a LIN network	119
11.5	How to control a LIN Master's scheduler	119
11.5.1	Using the Interactive Master	119
11.5.2	Using CAPL	120
11.6	How to log and replay LIN traffic	120
11.7	How to view LIN signals	121
11.8	How to manipulate LIN signals	121
11.8.1	Using CAPL Signal API	121
11.8.2	Using the CAPL function <code>output()</code>	122
11.8.3	Using the Interactive Generator Block	122
11.8.4	Using panels	122
12	MOST	123

12.1	Preliminary note	124
12.2	MOST database: Function catalog	124
12.3	How to create a CANoe.MOST configuration	124
12.4	How to analyze a MOST network	125
12.5	How to stimulate a MOST system	126
12.6	How to log and replay MOST data traffic	126
12.7	Using CAPL	127
12.7.1	Program-controlled sending	127
12.7.2	Program-controlled receiving	128
13	FlexRay	129
13.1	Preliminary note	130
13.2	How to create a FlexRay database	130
13.3	How to create a CANoe.FlexRay configuration	131
13.4	How to simulate and analyze a FlexRay network	131
13.5	How to log and replay FlexRay traffic	132
13.6	How to view FlexRay signals	133
13.7	How to manipulate FlexRay signals	133
13.7.1	Using signals	133
13.7.2	Using CAPL functions <code>FRUpdateStatFrame</code> / <code>FRSendDynFrame</code> / <code>FRUpdatePDU</code>	134
13.7.3	Using the FlexRay Frame Panel or FlexRay PDU Panel	135
13.7.4	Using panels	135
13.8	How to implement specific behavior for a remaining bus simulation	136
14	J1939 and NMEA 2000®	137
14.1	Introduction	138
14.2	Quick start	138
14.2.1	Create a J1939 database	138
14.2.2	Create a J1939 configuration	139
14.2.3	Create communication relationships	139
14.2.4	Sample configurations	140
14.3	Use cases	140
14.3.1	Analyze J1939 networks	140
14.3.2	Diagnose J1939 networks	140
14.3.3	Simulate J1939 networks	141
14.3.4	Test J1939 networks	142
14.3.5	Log and trigger J1939 data	143
14.3.6	Trigger and filter J1939 data	143
14.3.7	Modify J1939 signals	143
14.3.8	Analyze GNSS data	143
14.3.9	Simulate a GNSS receiver	144
14.3.10	Play back GNSS protocol files	145
15	ISO11783	147
15.1	Introduction	148
15.2	Quick start	148
15.3	Use cases	148
15.3.1	Simulate Virtual Terminals	148
15.3.2	Access process data	149
15.3.3	Simulate a process data dictionary	149

16	CANopen	151
16.1	Extended features of option CANopen	152
16.1.1	Trace window	153
16.1.2	CANopen generator block	153
16.1.3	CANopen Scanner	154
16.1.4	Bus configuration	154
16.1.5	Add-ons	154
16.2	Databases	154
16.3	Generate CANopen simulations	155
16.4	Generate tests	156
16.4.1	Device test	156
16.4.2	Application test	157
16.4.2.1	Stimulate PDOS	158
16.4.2.2	Run the signal-based SDO transfer	158
16.5	Control center ProCANopen	160
16.6	Sample configurations	160
17	IP	161
17.1	Extensions of the IP option	162
17.1.1	Check installation	162
17.2	Security advice for using CANoe.IP	163
17.2.1	Exclusive use of an Ethernet interface	163
17.3	Use cases	164
17.3.1	Analyze Ethernet networks	164
17.3.2	Filter Ethernet data	164
17.3.3	Stimulate Ethernet packets	164
17.3.4	Simulate Ethernet nodes	164
17.3.5	Run Remote CAN analysis	165
17.4	Quick start	165
17.4.1	Analyze network traffic	165
17.4.2	Evaluate signals	165
17.4.3	Run Remote CAN analysis	166
17.4.4	Sample configurations	166
18	J1587	167
18.1	Introduction	168
18.2	Prerequisites	168
18.2.1	Configure J1708 channels	168
18.2.2	Define J1587 parameters in CANdb++ Editor	169
18.3	Functionality	169
18.3.1	Parameter Monitor	169
18.3.2	Diagnostics Monitor	170
18.3.3	Trace window	170
18.3.4	Data and graphic window	171
18.3.5	Interactive Generator Block	171
18.3.6	Filter	172
18.3.7	CAPL	172
19	CANaero	173
19.1	Scope of delivery	174
19.2	Basics	174
19.3	Database Concept	174

19.4	Extensions	175
19.4.1	Trace window	175
19.4.2	Data window	175
19.4.3	Interactive Generator Block	175
20	Appendix A: Support	177
21	Appendix B: Address table	178
22	Index	181

1 Introduction

In this chapter you find the following information:

1.1	About this user manual	page 8
	Access helps and conventions	
	Certification	
	Warranty	
	Support	
	Registered trademarks	

1.1 About this user manual

1.1.1 Access helps and conventions

To find information quickly

The user manual provides you the following access helps:

- at the beginning of each chapter you will find a summary of its contents,
- in the header you see the current chapter and section,
- in the footer you see to which program version the user manual replies,
- at the end of the user manual you will find an index.










Online Help: Please refer to the online help for detailed information on all topics.

Conventions

In the two following charts you will find the conventions used in the user manual regarding utilized spellings and symbols.

Style	Utilization
bold	Blocks, surface elements, window- and dialog names of the software. Accentuation of warnings and advices. [OK] Push buttons in brackets File Save Notation for menus and menu entries
CANoe	Legally protected proper names and side notes.
Source code	File name and source code.
Hyperlink	Hyperlinks and references.
<STRG>+<S>	Notation for shortcuts.

Symbol	Utilization
	Here you can obtain supplemental information.
	This symbol calls your attention to warnings.
	Here you can find additional information.
	Here is an example that has been prepared for you.
	Step-by-step instructions provide assistance at these points.
	Instructions on editing files are found at these points.
	This symbol warns you not to edit the specified file.

1.1.2 Certification

Certified Quality Management System

Vector Informatik GmbH has ISO 9001:2008 certification.
The ISO standard is a globally recognized quality standard.

1.1.3 Warranty

Restriction of warranty

We reserve the right to change the contents of the documentation and the software without notice. Vector Informatik GmbH assumes no liability for correct contents or damages which are resulted from the usage of the user manual. We are grateful for references to mistakes or for suggestions for improvement to be able to offer you even more efficient products in the future.

1.1.4 Support

You need support?

You can get through to our hotline at the phone number
+49 (711) 80670-200
or you send a problem report to the **CANoe Support**.

1.1.5 Registered trademarks

Registered trademarks

All trademarks mentioned in this user manual and if necessary third party registered are absolutely subject to the conditions of each valid label right and the rights of particular registered proprietor. All trademarks, trade names or company names are or can be trademarks or registered trademarks of their particular proprietors. All rights which are not expressly allowed are reserved. If an explicit label of trademarks, which are used in this user manual, fails, should not mean that a name is free of third party rights.

- Outlook, Windows, Windows XP, Windows 2000 and Windows NT are trademarks of the Microsoft Corporation.
- CANoe, CANalyzer, CANdb++ Editor, J1939 CAPL Generator, ProCANopen and CANeds are trademarks of Vector Informatik GmbH.
- CANopen and CiA are trademarks of the CAN in Automation e.V.
- NMEA and NMEA 2000 are trademarks of the National Marine Electronics Association.
- eclipse is subject to the Copyright of the Eclipse contributors and others.

2 Installation

This chapter contains the following information:

2.1	General	page 12
2.2	System requirements	page 12
2.3	Installation requirements	page 12
2.4	Installation procedure	page 13
2.5	Notes on activating a software based license	page 13
2.6	Vector USB dongle	page 14
2.7	MOST	page 15
	MOST25: Use with Optolyzer Box	
	MOST150: Use with Optolyzer G2 3150o	
	MOST50: Use with Optolyzer G2 3050e	
2.8	Further CANoe options	page 17
2.9	Switching language versions	page 17
2.10	Running the test	page 18
2.11	Troubleshooting	page 18
	Software-specific error messages	
	Hardware-specific error messages	

2.1 General

Overview

This manual describes the installation of the software and associated hardware. It also describes the functional test used to check whether the software and hardware are installed correctly.



Info: The hardware drivers on the **CANoe** installation CD may be newer than the ones shipped with the hardware. Please use always the latest drivers.



Note: Please note that the CAN hardware you intend to use must be enabled for use with **CANoe**. Please see the appendix for more information on enabling.

2.2 System requirements

Installation sequence Please carry out the installation in the following order:

1. Install the **hardware** as described in the hardware manual.

Once the hardware is installed, please carry out a **driver update**. For more information on this, please see the appendix.

2. Install the **software**.

The following system configuration is recommended for use with **CANoe**:

Processor

Pentium 4 / 2.6 GHz (minimum: Pentium III / 1 GHz)

Memory (RAM)

1 GB (minimum: 512 MB)

Hard disk space

200 MByte...600 MByte
(Depending on options used and operating system components required.)

Monitor resolution

1280×1024 pixels (minimum: 1024×768 pixels)

Operating system

Windows VISTA / Windows XP at least with **Service Pack 2** / **Windows 2000** with **Service Pack 4**

Other

You will need (D)COM version 1.2 or later to support the COM interface.



Note: Administrator rights are needed to install **CANoe**.

2.3 Installation requirements



Note: Please note that you cannot install **CANoe** Version 3.0 or later over an older **CANoe** version (**CANoe** Version 2.5 or older). You can, however, delete the old **CANoe** version, rename the old **CANoe** installation, or install the new **CANoe** installation in a new folder. This makes it possible to work with different **CANoe** versions.

Windows VISTA, XP, 2000 Installation of the software is identical for these operating systems.

Installing the Options Further installations steps may be required if your package includes additional options. Please refer to the installation notes in the manual for each option.

2.4 Installation procedure

How to start the installation...

Please take the following steps to install the **CANoe** software:

1. Place the **CANoe** installation CD in your CD drive.

A Start window will appear, in which you can start the software installation.

If your computer is not configured to automatically launch Start windows, you can launch the installation program **SETUP.EXE** from the **Application** folder on the CD.

2. Follow the installation program instructions.



Note: For the installation you need administrator rights.

If you are logged on as standard user (with standard user rights), you have to start the **CANoe** installation program **Setup.exe** from the Explorer directly. A dialog is opened to be logged on as a user with administrator rights. After that the installation routine can be executed successfully.

Installation types

You can run a **Standard** or **Custom** installation.

Standard

→ In a **Standard installation**, the software is installed and pre-configured in accordance with your order (hardware platform, language version and bus type).

Custom

→ With a **Custom installation**, you can install hardware connections for alternately available hardware platforms, German or English language versions and sample configurations.

2.5 Notes on activating a software based license

Products

The following products and versions support software based license protection:

- **CANoe/CANalyzer** ≥ 7.1
- **Test Automation Editor** ≥ 1.1

Licensing

The software product you are about to install requires a license.

The license protection method depends on the product:

- **Hardware based license protection**
License becomes available when USB dongle or bus interface hardware is inserted/plugged
- **Software based license protection**
An Activation ID is delivered with your product and must be activated before usage.

The respective license protection is chosen when ordering the product.

Activation ID

If the product is delivered with a software based license protection, you will find a sticker on the CD/DVD cover, which shows an **Activation ID** of one of the following formats (examples given):

- ➔ A-1A2B3C4D5F6G7-1A2B3C4D5F6G7
- ➔ ACT-0000012345-000012-123456

After installing the software you will need this **Activation ID** to activate the license on your PC.

Activating a license

Once the software has been installed, start the **Activation Wizard** from the product's start menu entry (sometimes may also be located in the **Tools** subfolder) and make your selections there.

At one point you are asked to enter the **Activation ID**. Enter the **Activation ID** exactly as printed on the sticker found on the CD/DVD cover.

After the license has been activated successfully, you can start working with your application.



Info: Some product installers may automatically start the **Activation Wizard** once the installation is complete.

Help & support

Further help on activating a license is available by pressing the **[Help]** button in the **Activation Wizard**.

If you need further assistance with activating a license (e.g. when you do not have Internet access from your machine to activate the license online) please direct your questions to: activation@vector-worldwide.com

2.6 Vector USB dongle

Application areas

The USB dongle is required...

- ➔ ...for the **MOST** option.
Exception: When using hardware from the XL product family.
- ➔ ...after upgrading from **CANoe** to **CANoe.MOST** with XL products, unless the license was not subsequently defined on the card.
- ➔ ...for Optolyzer Integration Package (OIP) for the operation for MOST150 and MOST50.



Note: The USB dongle must not be connected during the installation.

How to install the USB dongle...

Procedure:

1. Run the setup file **haspdinst.exe**.

The file is in the **drivers\VectorDriverDisk\Drivers\32_Bit\Dongle** folder on the CD.

Installing this file may take a little while (up to 4 minutes).

2. You need to reboot your computer when the installation is complete.
3. Plug the USB dongle into the USB port.



Note: You can also search for a current driver in the Vector Download Centre. Install the driver with the downloaded program **Setup.exe** afterwards.

2.7 MOST

Prerequisites

To run MOST, you will need the following:

- A MOST option license that is tied to the hardware or to a USB dongle.
- Additionally for **MOST150**, for operation with an **Optolyzer G2 3150o**:
A license for the Optolyzer Integration Package (OIP) of Vector on a USB dongle or on a simultaneously connected Vector hardware.
- Additionally for **MOST50**, for operation with an **Optolyzer G2 3050e**:
A license for the Optolyzer Integration Package (OIP) of Vector on a USB dongle or on a simultaneously connected Vector hardware.



Note: The Optolyzer Integration Package (OIP) covers operation with **MOST150** and **MOST50**.



Further Information: For help with installing the MOST hardware, please refer to the associated installation manual.

2.7.1 MOST25: Use with Optolyzer Box

Prerequisites

To work with the Optolyzer Box, you will need the following:

- An **Optolyzer Box** (Firmware Version 2.50 or later),
- The **Optolyzer ActiveX control** (incl. new **Optolyzer** license number).

Installation

1. Install the Optolyzer ActiveX control on your PC.

The installation program that corresponds to your operating system is in the **\Drivers\Optolyzer\OptoControl** folder on the CD.

CANoe was developed and tested using the Opto Control version provided on the CD. We therefore urgently recommend that you install these driver versions.

2. In the **Windows** Control Panel, set the COM port settings for all COM ports that are to be used with an Optolyzer Box to **115200,8,n,1,Hardware**.
3. Install the MOST option in a folder of your choosing.
4. Activate **General information|Settings|Synchronize Hardware** in the Vector driver configuration dialog.
5. Choose the speedgrade MOST25 and the HW type **Optolyzer** for the MOST channel in the **CANoe** hardware configuration dialog (**Configuration|Network Hardware...**).
6. Enter the Optolyzer ActiveX control license code on the **Interface** page in the hardware configuration dialog.

Optolyzer Professional

If you install Optolyzer Professional, you can enter the ActiveX control license number via this tool. It is then saved in the **Windows** registry.

You can save multiple license numbers in this way. They are all checked when a connection to the Optolyzer Box is created.

2.7.2 MOST150: Use with Optolyzer G2 3150o**Installation**

1. Install the **Optolyzer G2 3150o** or **Optolyzer G2 3150o Production** corresponding to the **Optolyzer** user manual.



Remarks: Note that a license for the **Optolyzer** Integration Package (OIP) of Vector on a Vector hardware, USB dongle, or as a license key is required.

The **Optolyzer G2** is accessed via the following ethernet port numbers. This access must not be blocked by any installed firewall:

Spy: 27998

Node: 27999

On PC side the port numbers are allocated automatically. If you need certain port numbers you have to adjust the file CAN.ini. For **Optolyzer G2 3150o** (MOST150) at channel 1 you'll have to edit the following section:

```
[OptolyzerG2_1]
MyPortNode=
MyPortSpy=
```

2. Choose the speedgrade MOST150 and the HW type **OptoLyzer OL3150o** for the MOST channel in the **CANoe** hardware configuration dialog (**Configuration|Network Hardware...**).
3. Enter the IP address of the **Optolyzer** on the **Interface** page.
(If the **Optolyzer** is already connected, you can determine its IP address at the push of a button).
4. On the **Setup** page, also select the network adapter to which the **Optolyzer** is connected.

2.7.3 MOST50: Use with Optolyzer G2 3050e**Installation**

1. Install the **Optolyzer G2 3050e** or **Optolyzer G2 3050e Production** corresponding to the **Optolyzer** user manual.



Remarks: Note that a license for the **Optolyzer** Integration Package (OIP) of Vector on a Vector hardware, USB dongle, or as a license key is required.

The **Optolyzer G2** is accessed via the following ethernet port numbers. This access must not be blocked by any installed firewall:

Spy: 27998

Node: 27999

On PC side the port numbers are allocated automatically. If you need certain port numbers you have to adjust the file CAN.ini. For **Optolyzer G2 3050e** (MOST50) at channel 1 you'll have to edit the following section:

```
[OptolyzerG2_50_1]
MyPortNode=
MyPortSpy=
```

2. Choose the speedgrade MOST50 and the HW type **OptoLyzer OL3050e** for the MOST channel in the **CANoe** hardware configuration dialog (**Configuration|Network Hardware...**).
3. Enter the IP address of the **Optolyzer** on the **Interface** page.
(If the **Optolyzer** is already connected, you can determine its IP address at the push of a button).
4. On the **Setup** page, also select the network adapter to which the **Optolyzer** is connected.

2.8 Further CANoe options

Overview

The options are designed as an add-on to the standard **CANoe**. A number of standard **CANoe** files are replaced during installation (e.g. driver) and others are added (e.g. sample configurations).

It is therefore important for the versions that they match properly. The installation program tests this compatibility and issues a warning if appropriate. If there is any incompatibility between versions, you should get in touch with Vector Support.

In addition to the sample configurations of the standard **CANoe**, option-specific examples are installed in a directory.

A number of different level 7 options can be installed simultaneously in a single directory – for example ISO11783, J1939, and CANopen.

Option CANopen, CANaerospace

Install **CANoe**, if you have not already done so. Proceed according to its installation instructions.

After installation of the standard version of **CANoe**, the installation of the option starts automatically.



Hint: During installation of option IP the message „The software ‚Vector Network Driver Miniport‘ has not passed Windows Logo Testing“ could appear several times. Please confirm the dialog with the button **[Continue installation]**.



Caution: Please do not install or uninstall the corresponding options by manually copying or deleting files. The programs use COM mechanisms of MS Windows that must be registered or deregistered by the installation program.

2.9 Switching language versions

Configuration

The German and English language versions are installed by default during the standard installation.

You can set the language for all of the program features and the Help (Menu: **Configuration|Options...|Appearance**).

You need to close and then re-start **CANoe** for the changes to become effective.

2.10 Running the test

- Prerequisite** To test the **CANoe** software installation, the CAN hardware must be successfully installed.
- Procedure**
1. Connect the two CAN ports of your CAN hardware using a cable that is terminated in a way that is appropriate to the bus type.
 2. Load the **CANSystemDemo.cfg** sample configuration in the Demo folder (\Demo_CAN_CN\CANSystemDemo) and start it.
If the installation is successful, you will be able to observe CAN messages in the Trace window.
- Result** This functional test also confirms that the CAN hardware has been installed correctly.

2.11 Troubleshooting

2.11.1 Software-specific error messages

- Overview** You will find software-specific error messages and how to correct them here.



Further Information: The **Overview** in the **System Messages** section of the online Help contains a list of additional system messages.



Nr. 0: Unable to find PC card. Error number 0. Timeout during card initialization

- Background** No response was received when accessing the CAN hardware.
- Cause** The **CANoe** version is not compatible with your CAN hardware.
- Procedure** Make sure that the installation is compatible with your CAN hardware.
To do this, open the **Help|Info** dialog. This shows the software version, e.g. **CANoe**, along with the hardware expected for this installation. These are:

CANcardXL	for CANcardXL
CANcaseXL/log	for CANcaseXL/log
CANboardXL/pxi/PCIE	for CANboardXL/pxi/PCIE
DEMO	Demo driver with virtual CAN1-CAN2 connection

- Cause** Faulty driver installation.
Check the driver installation by following the instructions provided in the appendix.



CANoe not allowed with HW version (Software protection with CAN board):0

Either your CAN hardware is not enabled for **CANoe** or it was not possible to determine whether the hardware is enabled.

Please use enabled hardware.

2.11.2 Hardware-specific error messages

Side note

Some error messages pertain to faulty settings in the CANcardXL driver configuration dialog.

You will find this information in the Windows menu under **Start|Settings|Control Panel|Vector Hardware**.

It is described as **CAN Hardware** there. For more details, please refer to the hardware manual.



No. 4000: CAN channel X is not available

Procedure

Check in **Vector Hardware** to see whether the **CANoe X** entry exists.



No. 4001: CAN channel X does not exist

Procedure

Check in **Vector Hardware** to see whether the hardware assigned to the **CANoe X** entry is active.



No. 4002: Board initialization error 3014

Cause

An error occurred when accessing the driver.

Procedure

Check in **Vector Hardware** to see whether the driver is installed correctly.

3 Basics

In this chapter you find the following information:

3.1	Introduction to CANoe	page 22
3.2	Tips for using CANoe	page 24
3.3	Overview of the programs	page 26
3.4	CANoe architecture	page 27
3.5	Particularities of the demo version	page 28

3.1 Introduction to CANoe

Purpose	<p>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.</p>
3 phase model	<p>The development process is based on a phase model which differentiates between three development stages:</p>
Phase 1 Requirements analysis and design of the network system	<p>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.</p> <p>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.</p> <p>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.</p>
Phase 2 Implementation of components with simulation of remainder of the bus	<p>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.</p>
Phase 3 Integration of the overall system	<p>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.</p>

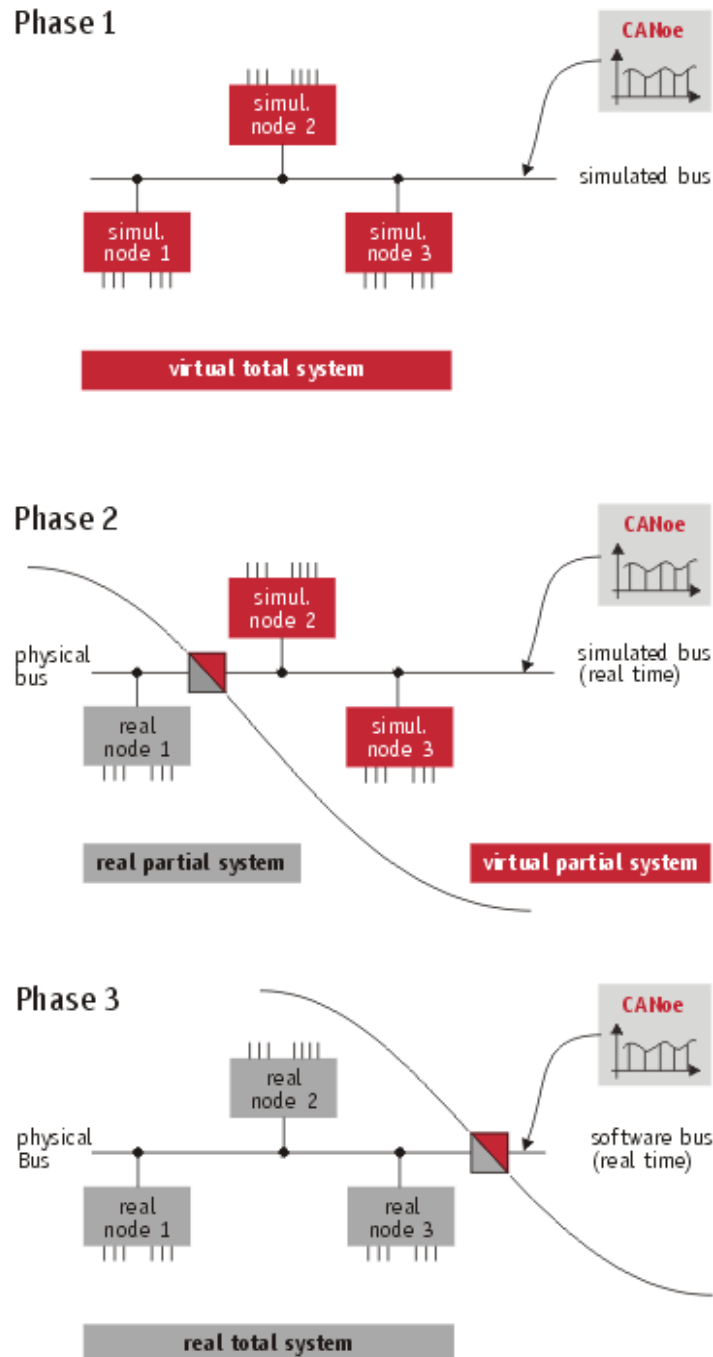


Figure 1: Phase model of the development process

Environment variables

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.

Panels

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 Designer. During the simulation values of environment variables can be displayed (lamps, counters) and interactively modified (switches, potentiometers).

Components of the simulation system

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.

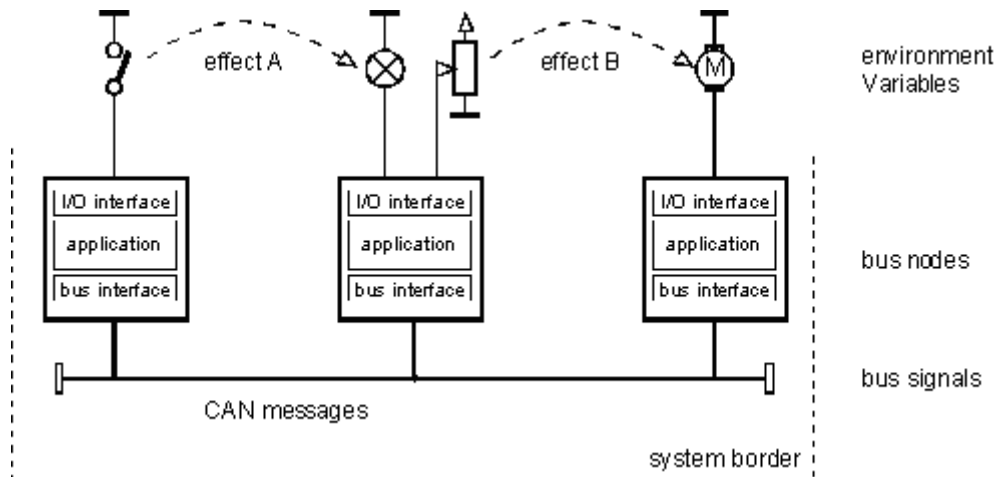


Figure 2: Components of the simulation system

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.

3.2 Tips for using CANoe

Operating CANoe...

Essentially, **CANoe** can be operated by both mouse and keyboard.

...during measurement

All of the windows of the application can be moved, enlarged, reduced, opened and closed again at any time, i.e. also during the measurement.

Main menu

CANoe is operated using the main menu. The individual menu commands are described in detail in online help.

Shortcut menu

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.

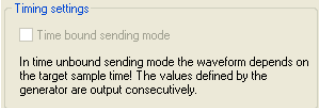
Simulation and measurement setup

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

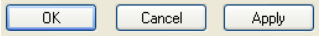
Box types in dialogs

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:

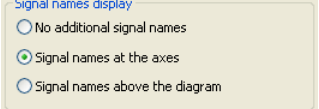
Comment box

Element	Description
	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.


Button

	Buttons serve to execute certain actions, e.g. to exit the dialog box or to open a subordinate dialog box.
---	--


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.
---	---

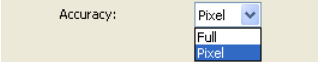
Check box

	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 „✓“.
--	--

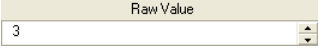
Text input box

	Alphanumeric box field, e.g. for entering file names. Numeric box field, e.g. for entering integer or floating point numbers.
---	---

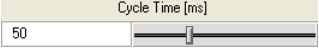
Drop down list

	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.
---	--

Spin control

	You can use the Spin control to adjust a value within a defined value range. A value may be entered directly in a text box by keyboard input, or it may be set by left-click the increment or decrement button to increment/decrement the value stepwise by the configured step width.
---	--

Slider

	With moving the slider you can edit numerical values within a specified value range.
---	--



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

3.3 Overview of the programs

Overview

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 Designer / 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. Each display and control element must be configured with a symbol from the database so that the display and control elements can be set or read-out by the CAPL models in CANoe.
- 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.

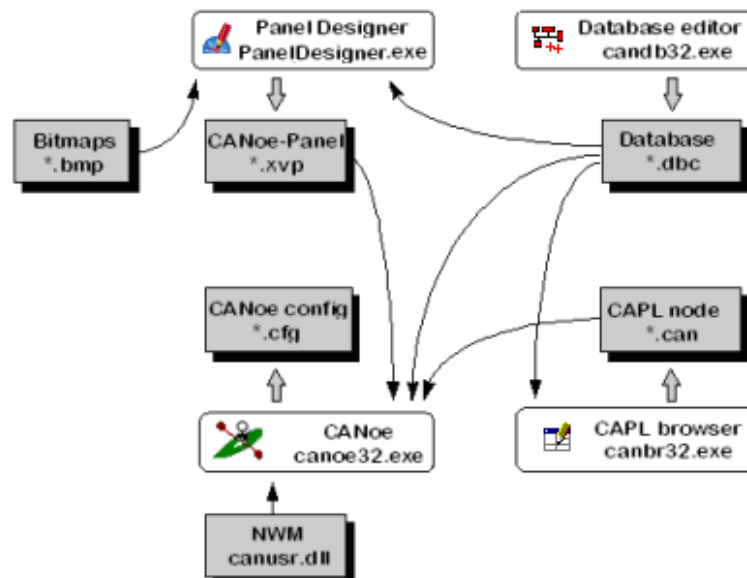


Figure 3: CANoe system overview

3.4 CANoe architecture

Overview

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.

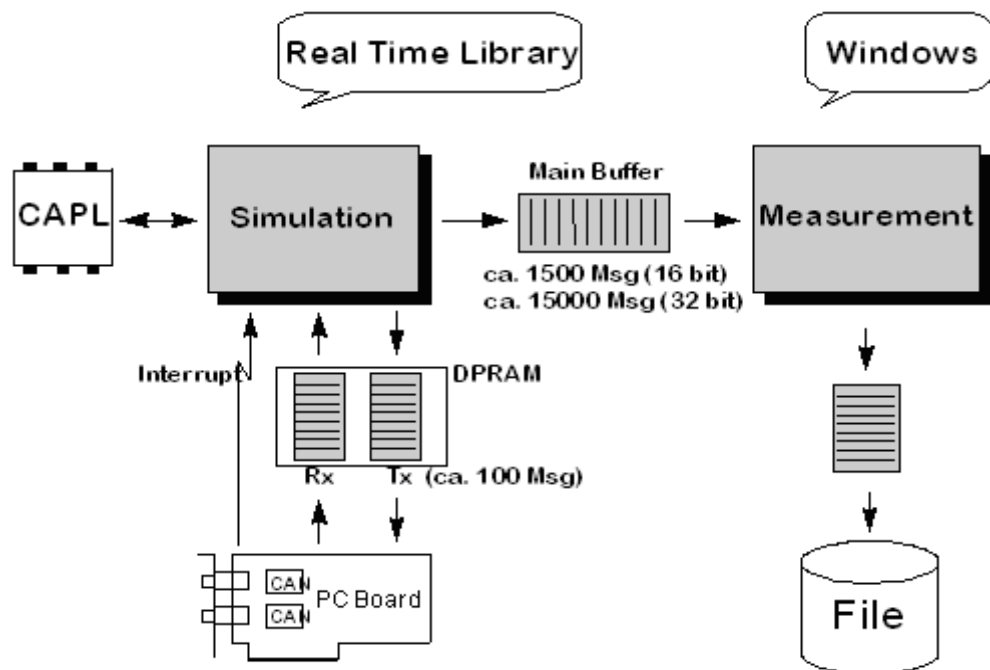


Figure 4: Internal structure of CANoe

Function blocks

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.

Blocks in real-time library

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



Info: 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.

3.5 Particularities of the demo version

Demo driver	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 a accurate time stamps.
Settings	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.
Limitations	<p>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.</p> <p>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.</p>

4 CANoe Tour

In this chapter you find the following information:

4.1	Overview	page 30
4.2	Preparations	page 30
4.3	Setting up the bus	page 32
4.4	Transmitting data	page 33
4.5	Evaluation windows	page 37
4.6	Working with symbolic data	page 40
4.7	Analysis of signal values in the Data window	page 41
4.8	Analysis of signal responses in the Graphics window	page 43
4.9	Use of the database in transmitting messages	page 44
4.10	Logging a measurement	page 45
4.11	Evaluating a log file	page 46
4.12	Creating a CAPL program	page 47
4.13	Simulation of distributed systems in CANoe	page 49
	Creating the database	
	Creating panels	
	Creating network node models	

4.1 Overview

- Operating concept** 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.
- Set up CANoe** 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.
- CAPL** 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**.

4.2 Preparations

- Windows** **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 shows you the data flow and simultaneously allows you to configure **CANoe**.

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

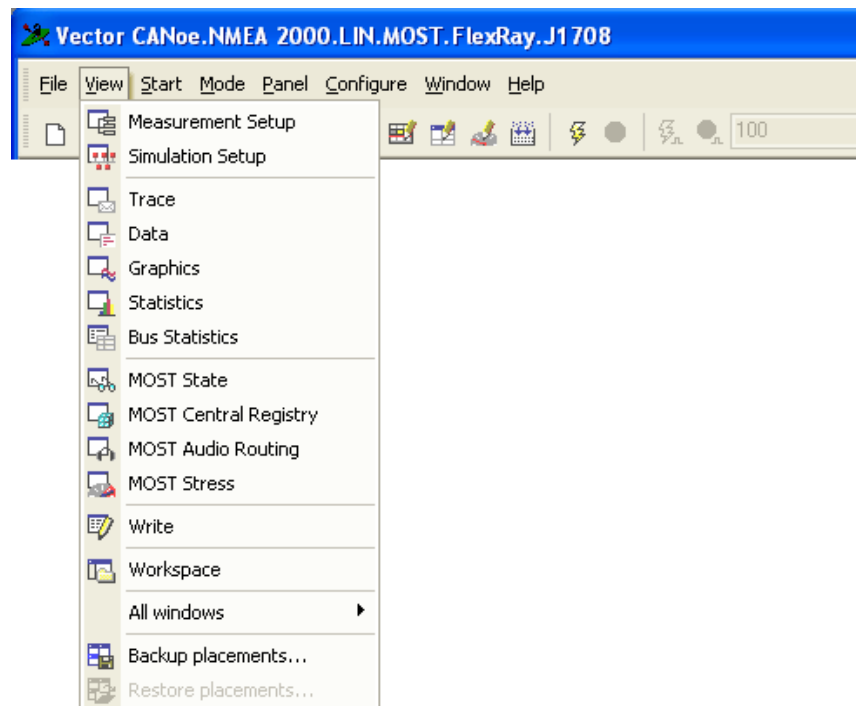



Figure 5: **View** menu on main menu bar

Simulation setup

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.

Measurement setup

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.

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

Evaluation windows

The information arriving at each evaluation block is 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.

Create a new configuration

Make sure that you begin this tour with a new configuration by selecting the menu item **File | New configuration**. The dialog for choosing a template will be opened. In this dialog choose the **CAN_83kBaudTemplate.tcn** template and close the dialog with **[OK]**. The wizard is not needed for this tour.

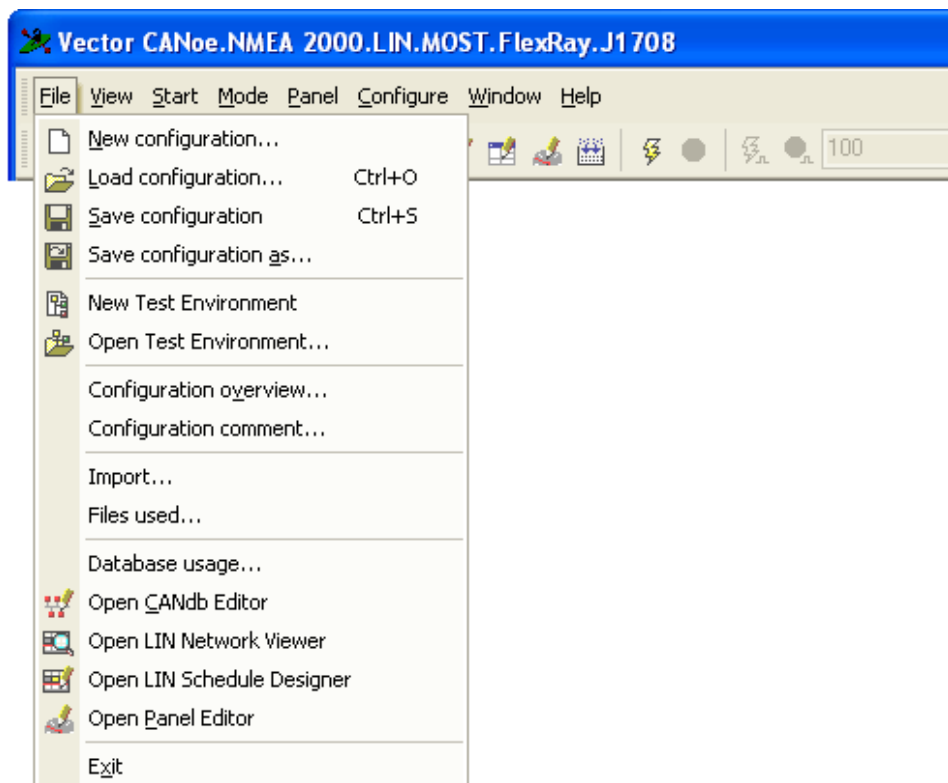


Figure 6: Menu item **File|New configuration**

4.3 Setting up the bus

Preparations

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

Connect PC card and CAN controller

First, connect the two D-Sub 9 connectors of your CAN card to one another (CANcabs). For a high-speed bus interface you need a connection cable (CANcable) with two bus termination resistors of 120 Ω each. 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.

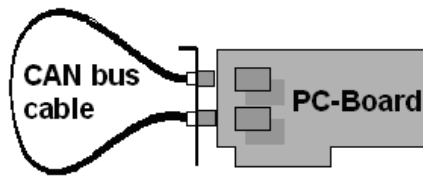


Figure 7: PC card with connection cable

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.

Define the bus parameters



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.

1. 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.

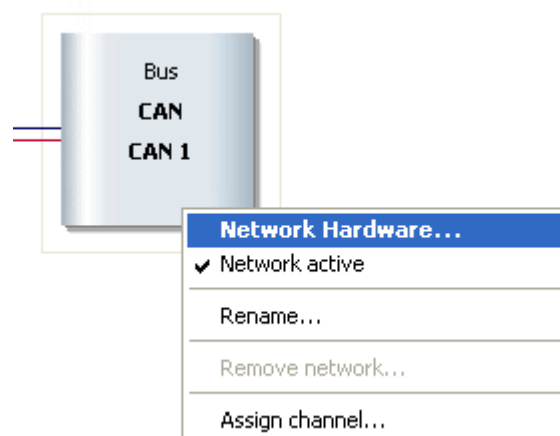


Figure 8: Shortcut menu of the bus symbol

2. Choose the shortcut menu item **Network Hardware...** and open the **Network Hardware Configuration** dialog.

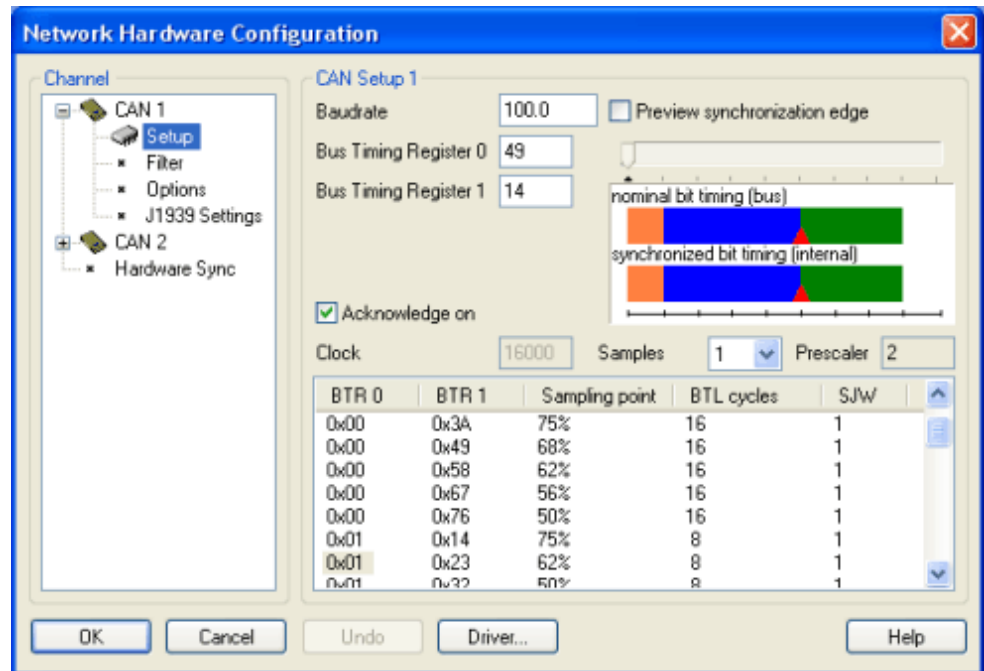


Figure 9: Network Hardware Configuration dialog

- After then select **+** and **Setup** from the configuration dialog for the first controller **CAN 1** and type in the value for the baudrate 100kBaund. This makes sense for both high speed and low speed buses. **CANoe** recommends default values for the controller registers, which you accept with **[OK]**. When you do this - besides the transmission speed of 100 kBaund - 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 **CAN 2**. When you exit the dialog, accept the values with **[OK]**.

Real channels,
application channels

With the **[Driver...]** button you can open the **Vector Hardware Config** dialog. There you can assign the application channels to the real channels.

4.4 Transmitting data

Set up a data source Your current test setup still does not have a data source. So set up a data source 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 send 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.

Insert a generator block

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 shortcut menu – inserting a generator block CAN on the bus.

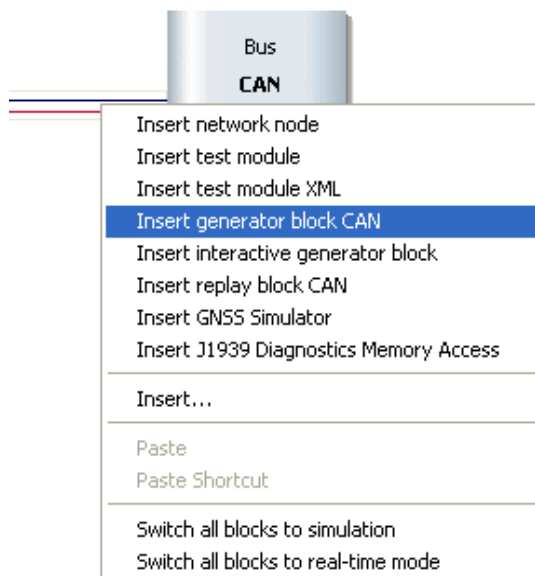


Figure 10: Bus symbol in simulation setup with shortcut menu of the bus lines

Afterwards, this appears in the simulation setup as a rectangular block that is connected to the simulated bus (red line).

2. Open the **Generator sendlist** with the generator block's shortcut menu item **Configuration of transmit list...**

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

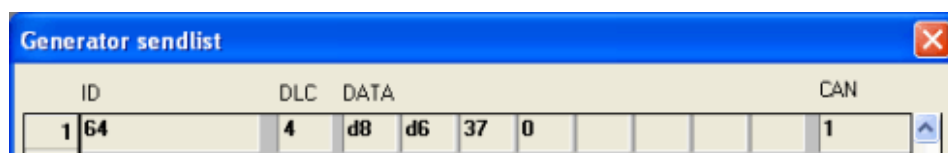


Figure 11: Transmit list of Generator block

Exit the transmit list with **[OK]** to accept the values in the configuration.

3. Open the generator block's **Generator Block Trigger Configuration** dialog via the shortcut menu **Configuration of triggering...**

In this dialog you can configure the triggering of the transmit action.

4. Activate the **With period** option and enter the value 100 in the input box to the right of this.

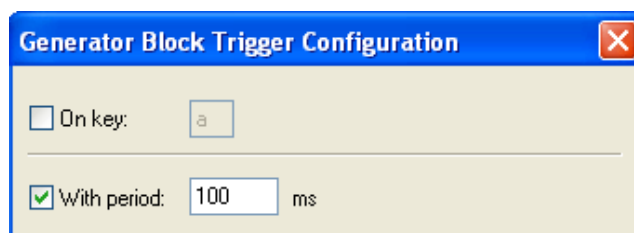


Figure 12: Triggering of Generator block

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

Associate a database

Furthermore, **CANoe** requires that you associate a database to the configuration before the start of measurement. Therefore, you should initially assign the database **POWERTRAIN.DBC** from the demo directory

DEMO_CAN_CN\CANSYSTEMDEMO\CANDB to your active **CANoe** configuration. The concrete benefits of this database will be made clear in the sections that follow.



1. 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.
2. If you go to **Databases** with the mouse pointer and click with the right mouse button, you will open the **Open** dialog with the shortcut menu item **Add...**
3. Choose in this dialog the above mentioned database.
4. If you click on the **[OK]** button, the new database is accepted for the current bus and displayed in the system view window.




Info: With the help of this symbolic information the data contents of messages can now be interpreted in **CANoe**. Please note that this is only practical if the database information describes the system that you are currently observing. So you have to ensure that the database associated to the configuration matches the real network.

Save your configuration

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

Start the measurement

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.

Display in the Trace window

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.

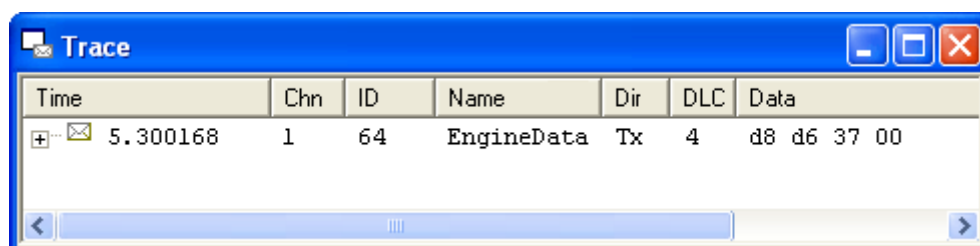


Figure 13: Trace window

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.

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**.

Configure the channel settings

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.



1. Open the configuration dialog with the bus symbol's shortcut menu **Network Hardware...** and choose in the opened dialog **CAN 2/Filter**.

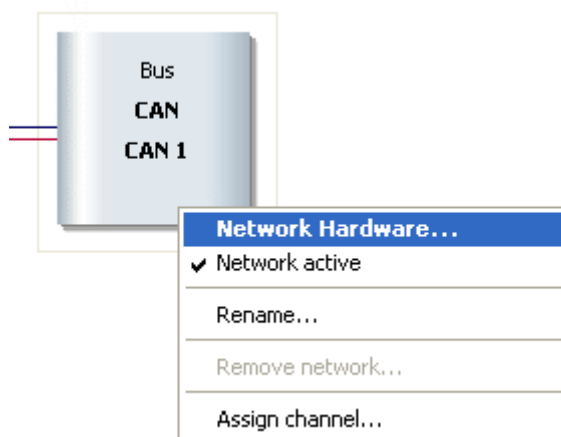


Figure 14: Shortcut menu of the bus symbol

2. 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.

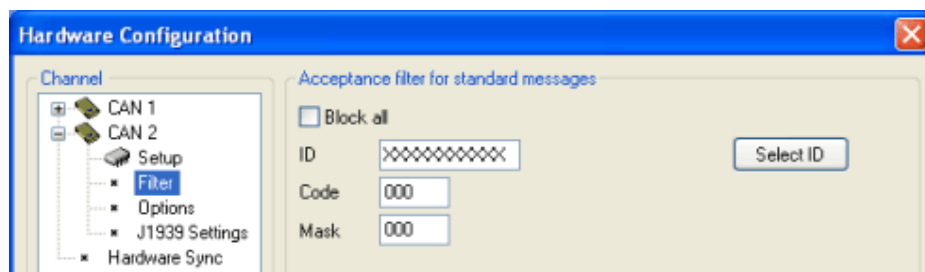


Figure 15: Configuration of acceptance filter

Display in the Trace window

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).

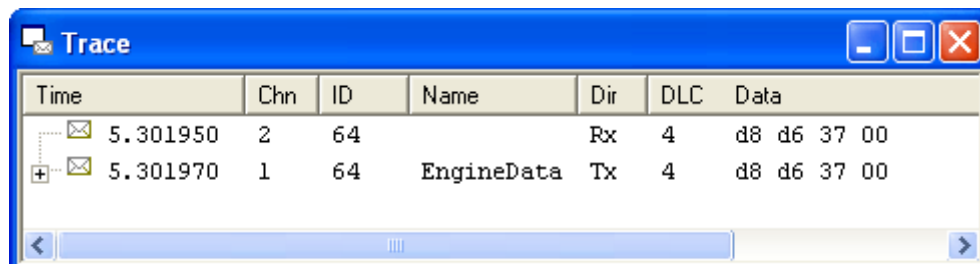


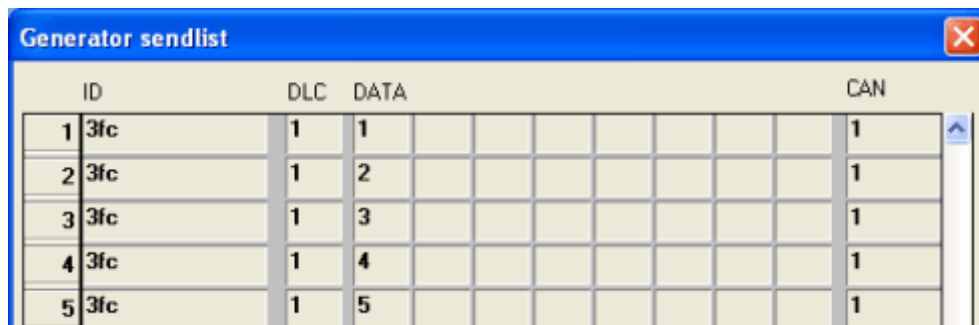
Figure 16: Trace window

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.

Insert a second generator block

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	1
2	3fc	1	2	1
3	3fc	1	3	1
4	3fc	1	4	1
5	3fc	1	5	1

Figure 17: Sendlist for generator block



Info: 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 shortcut menus appear in gray shading.

Further data sources

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.

4.5 Evaluation windows

Data analysis


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


Trace window

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.

Configuration of the Trace window

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.

Statistics window

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).

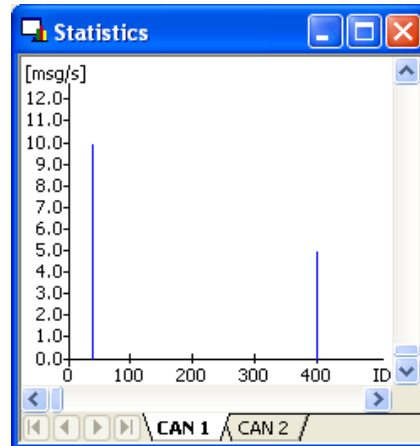


Figure 18: Statistics window

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.

Statistics report

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.

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.

Activate the statistics report

Stop the measurement and activate the statistics report in the configuration dialog of the statistics block (measurement setup).

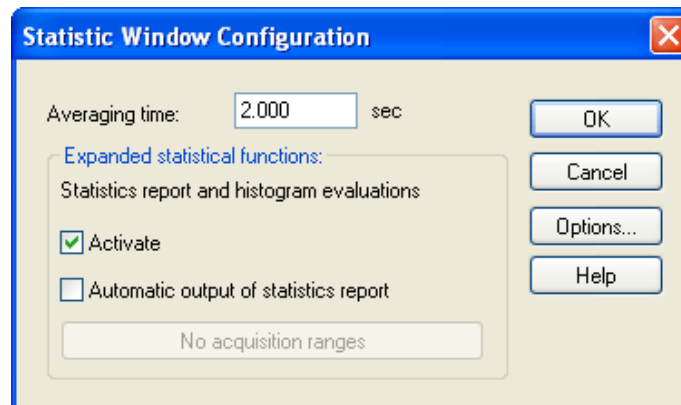
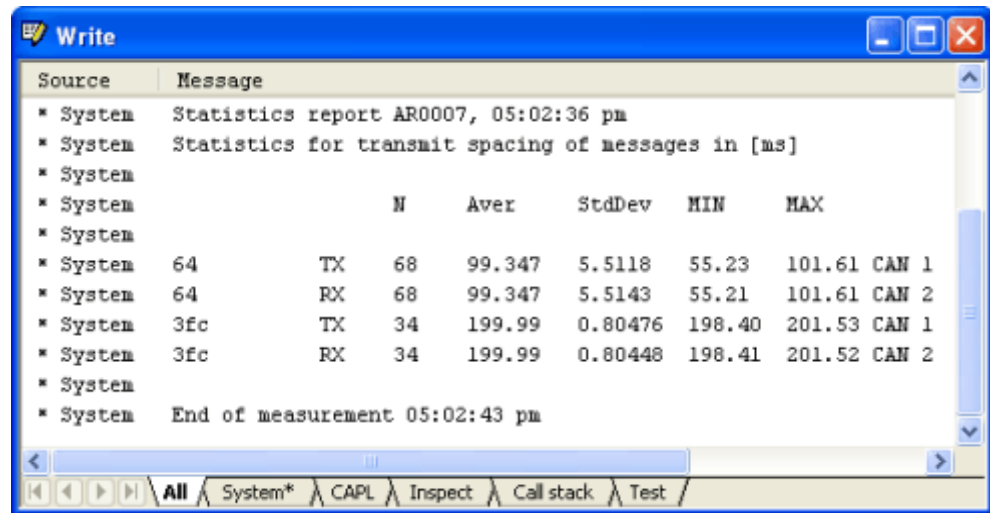


Figure 19: Activate the statistics report

Display of the statistics report in the Write window

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 shortcut menu.



The screenshot shows the 'Write' window with a statistics report. The report includes a header for 'Statistics report AR0007, 05:02:36 pm' and a table for 'Statistics for transmit spacing of messages in [ms]'. The table has columns for N, Aver, StdDev, MIN, and MAX. The report also shows the end of measurement at 05:02:43 pm.

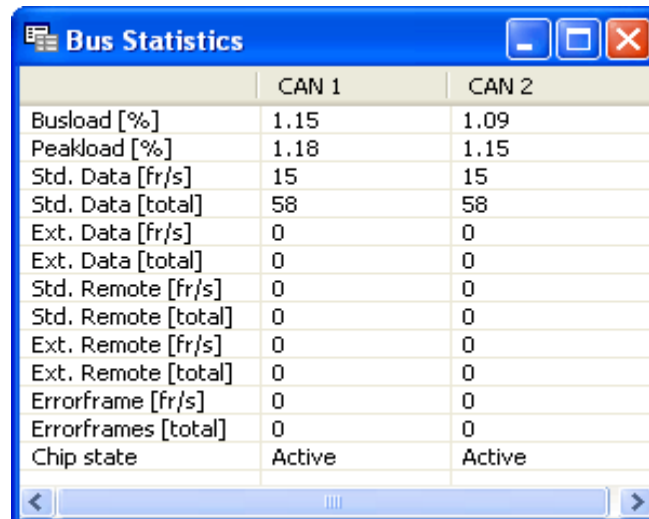
Source	Message
System	Statistics report AR0007, 05:02:36 pm
System	Statistics for transmit spacing of messages in [ms]
System	
System	N Aver StdDev MIN MAX
System	64 TX 68 99.347 5.5118 55.23 101.61 CAN 1
System	64 RX 68 99.347 5.5143 55.21 101.61 CAN 2
System	3fc TX 34 199.99 0.80476 198.40 201.53 CAN 1
System	3fc RX 34 199.99 0.80448 198.41 201.52 CAN 2
System	End of measurement 05:02:43 pm

Figure 20: Statistics report in the Write window

Bus Statistics window

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.



The screenshot shows the 'Bus Statistics' window with a table of bus data traffic statistics for CAN 1 and CAN 2.

	CAN 1	CAN 2
Busload [%]	1.15	1.09
Peakload [%]	1.18	1.15
Std. Data [fr/s]	15	15
Std. Data [total]	58	58
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

4.6 Working with symbolic data

Symbolic description of 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.



Further Information: Please refer to the CANdb++ manual and the CANdb++ online help included with the **CANoe** product for further information on the CANdb++ Editor.

Interpretation of the data bytes

At this point we would like to use the database `POWERTRAIN.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 simulation setup.

To do this, first open the database using the  button on the toolbar. The CANdb++ Editor is opened, and the contents of the database `POWERTRAIN.DBC` is shown in the Overall View window of the CANdb++ Editor.

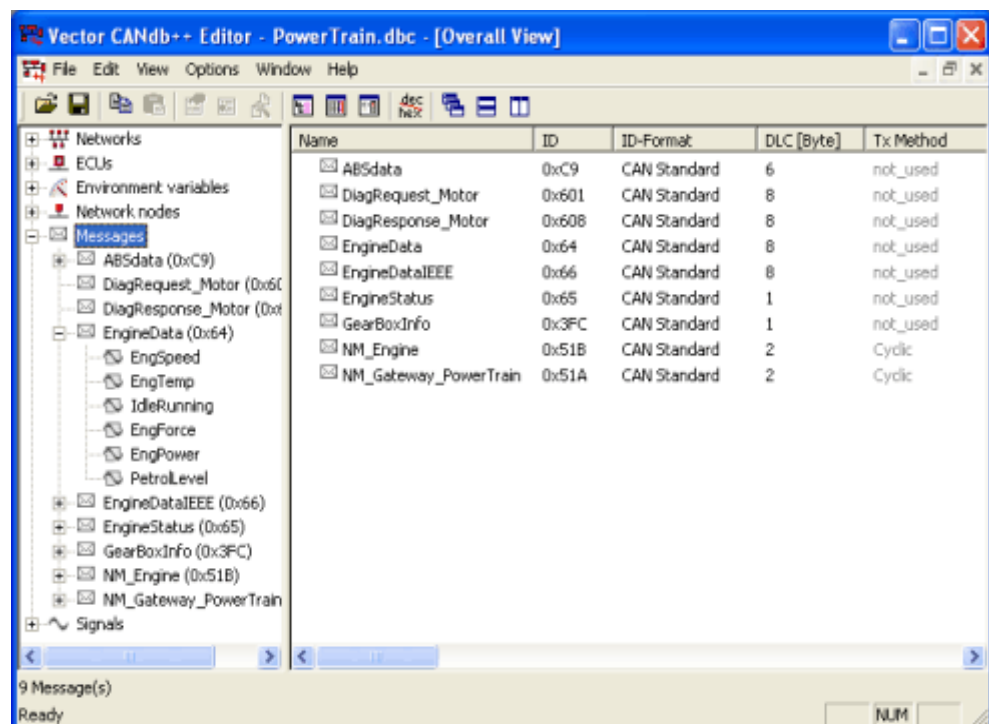


Figure 22: Overall View window of the CANdb++ Editor

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.

4.7 Analysis of signal values in the Data window

Display of momentary signal values

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 simulation setup.

Add signals in the Data window

For the display of signal values in the Data window, you have to add signals.



1. Open with the Data window's shortcut menu item **Add signals...** the Symbol Selection dialog.

The tree view of the dialog allows you to search for a specific signal. Each database gets one branch each for signals, messages and nodes.

2. Choose the **EngineData message** and select and accept all signals of this message.

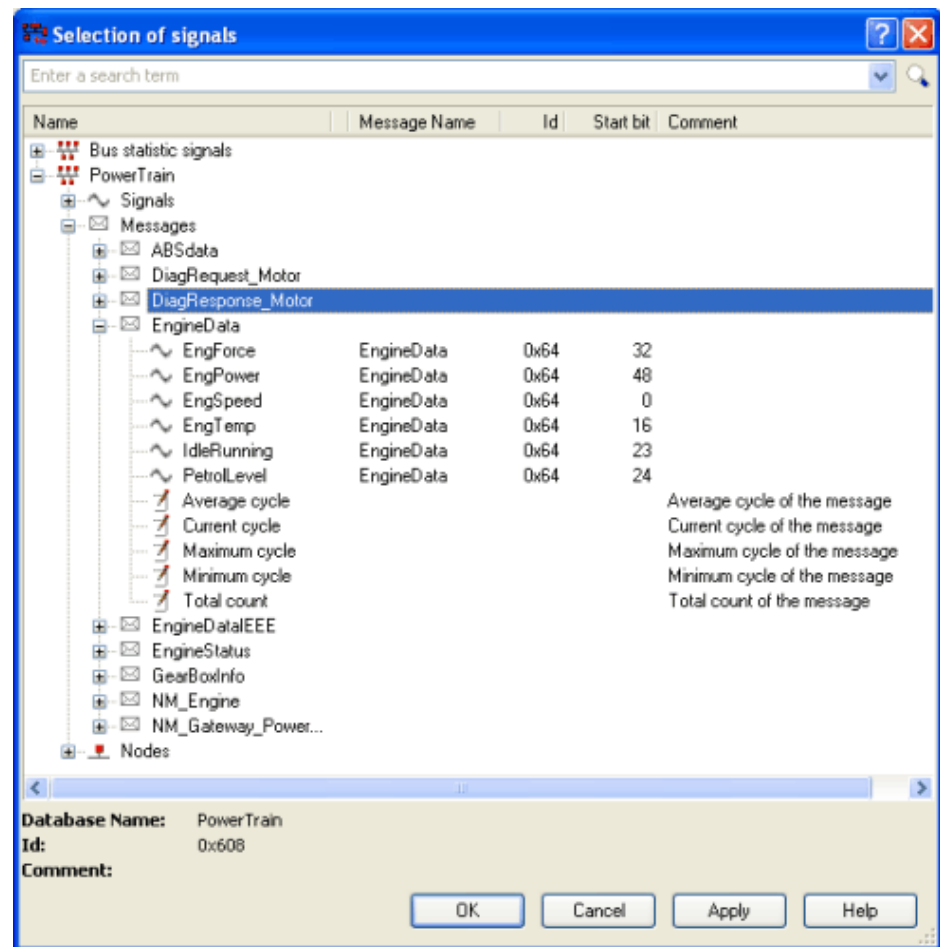


Figure 23: Selecting Signals with the Symbol Selection dialog

3. Close the dialog with **[OK]**.

Now the signal names are entered in the window.

Display in the Data 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.

Name	Value	Unit	Raw Value	Bar
EngForce	--	N	--	
EngPower	--	kW	--	
EngSpeed	55000	rpm	55000	
EngTemp	60	degC	55	
IdleRunning	0		0	

Figure 24: 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.

4.8 Analysis of signal responses in the Graphics window

Analysis of signal responses

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 simulation setup.

Add signals in the Graphics window

The second message generated in the simulation setup is also described in the associated database.



1. Open with the Graphics window's shortcut menu item **Add signals...** the Symbol Selection dialog.

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.

2. Choose the signals and confirm them with **[OK]**.

In the Graphics window you see that the signals are now entered in the legend on the left side of the window.

Display in the Graphics window

You can now observe the time responses of these signals in the Graphics 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.

For a useful display of the single gear values the **Step** connection type of the lines is suitable.



1. Open with the Graphics window's shortcut menu item **Configuration...** the **Graphic configuration** dialog.
2. Choose the item **Signal list** in the tree view on the left side.
3. Mark the **Gear** signal.
4. Open the **Lines / Markers** dialog via the **[Lines...]** button.
5. Choose the **Steps** connection type and confirm it with **[OK]**.
6. Close the **Graphic configuration** dialog with **[OK]**.
7. Start the measurement.

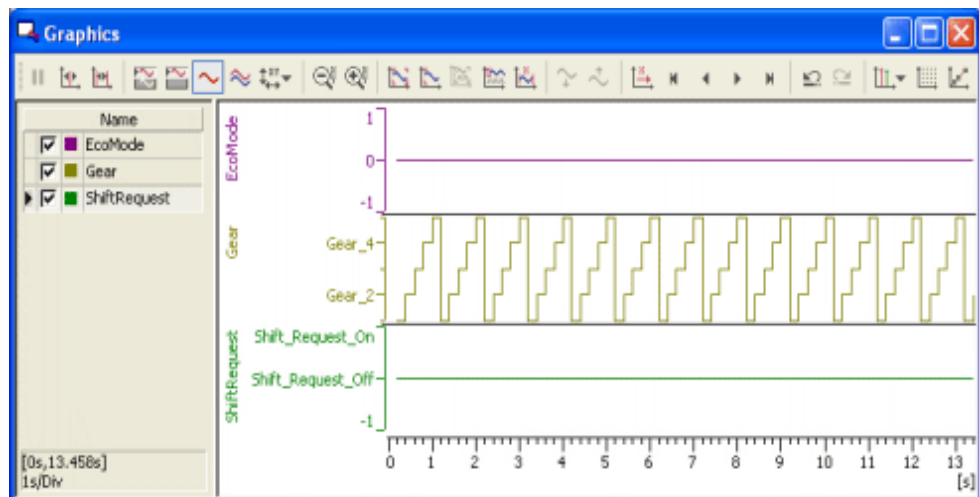


Figure 25: Graphics window

This corresponds to the five values that you entered in the generator block as part of **unit 2**. The values remain in the Graphics window after the end of the measurement.



Further Information: The measurement functions that the window provides for post-analysis are described detailed in chapter 6.6 and in the online help.

4.9 Use of the database in transmitting messages

Symbolic data from the database

Until now you have only used the symbolic database to observe signal values. However, the application capabilities reach well beyond this.

Open the transmit list of the generator block of **unit 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.

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


Figure 26: 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).

4.10 Logging a measurement

Data logging


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.

Activate the logging branch

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 configuration from the data source. You can do this by double clicking the break symbol  or with the shortcut menu item **Remove break** of the break.

Configure the logging file

With the shortcut menu item **Logging file configuration...** of the file icon located at the far right of the logging branch, you can open the configuration dialog.

Here you can enter the file name for the measurement log as well as its format. Select ASCII format here.

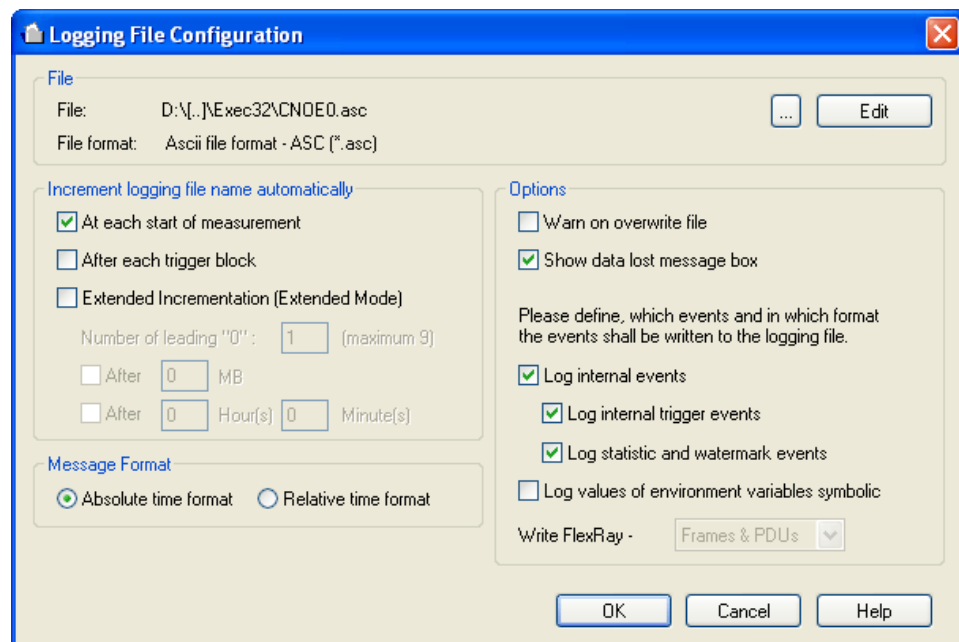


Figure 27: Configuration dialog in the Logging branch

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.

Configure the trigger conditions	<p>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.</p> <p>Open the configuration dialog with the Trigger block's shortcut menu item Configuration...</p> <p>To log the entire measurement it is sufficient to change the mode from Single Trigger to Entire Measurement in the trigger configuration dialog.</p>
Start the measurement	Start after the configuration of the log file and the trigger condition the measurement, which you stop again after 20 seconds.
Open the log file	<p>Now with a double click on the log file icon you can open the logged ASCII file.</p> <p>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.</p>

4.11 Evaluating a log file

Play back recorded data	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	<hr/> Play back the log file recorded for the last task in offline mode, and observe the signal response in the Graphics window. <hr/>
Activate the Offline mode	<p>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).</p> <p>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.</p>
Choose the data source	Select the log file of the last task via the Configuration... shortcut menu item of the file icon at the left of the measurement setup.
Deactivate the logging branch	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.
Play back the log file	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 <F>8) or in Single-Step mode (Start Step menu item or <F7>).
Analysis in Offline mode	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.

4.12 Creating a CAPL program

What is CAPL

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.



Further Information: You will find a complete description of the programming language together with numerous detailed examples in the online help.

Create a CAPL program

In the next task you will create a simple CAPL program to count messages that are generated in **CANoe's** simulation setup.

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.

Preparation

- ➔ 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.

Insert a CAPL node

First you must decide where you wish to insert your CAPL program in the data flow plan. In principle, any hot spot 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.

Configure the CAPL node

You can also access the node's configuration dialog via the shortcut menu **Configuration....** First, select a program name, e.g. COUNTER.CAN.

Start the CAPL Browser

Start the CAPL Browser either from the configuration dialog's **[Edit...]** button or directly by double clicking the program block **P** in the measurement setup.

Insert a variable

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.

Create an on message event procedure

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 shortcut 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 shortcut 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++;
    output(this); // The EngineData value is displayed in the
                  Statistics window
}
```

Create a second on message event procedure

To display all other measurement values in the Statistics window, insert the following procedure:

```
on message*
{
    output(this);
}
```



Info: Without this procedure the CAPL program would have the effect of a filter because only the message **EngineData** would be transmitted to the Statistics block.

Create a on key event procedure

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 **CAN Messages** event type again.

Now insert a Keyboard event in the CAPL program from the shortcut menu item **New**. 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()`.

Save and compile the program

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 win-dow 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.

Display in the Write window

Now start the measurement. The generator block in the simulation setup 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.

4.13 Simulation of distributed systems in CANoe

Working with environment variables

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.

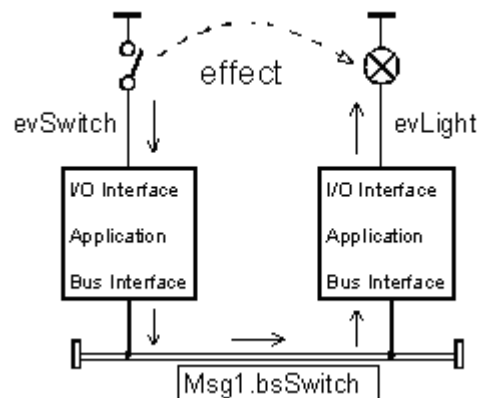
To work with environment variables in CAPL you use the event procedure type `on envVar` ("React to change in environment variable"). The CAPL functions `getValue()` and `putValue()` are used to read and write environment variables.

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

Unit 8

Create 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.


Diagram



Procedure

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

- ➔ Create the database with messages, signals and environment variables
- ➔ Create the network node periphery, i.e. the control panels
- ➔ 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.

4.13.1 Creating the database

Usage of a data base 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).

Create a database

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 peripheries. 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**.

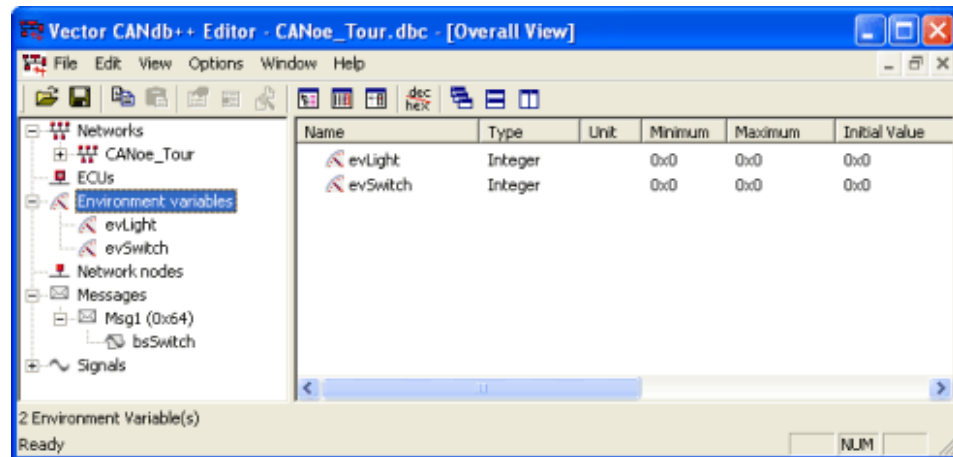


Figure 28: Environment variables in the database

Save the database and assign it to the configuration

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 shortcut menu.

4.13.2 Creating panels

Node's periphery

A separate application, the Panel Designer, is provided with **CANoe** for creating the node's periphery.

In the current configuration, one separate panel must be created for each of the two nodes.





Further Information: Please refer to the online help for a detailed introduction into the Panel Designer.

Create the first panel

The first panel has a single control, a switch.



1. You can start the Panel Designer 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 connection.
2. Open a new panel with the menu item **File|New Panel** and select the file name `SWITCH.XVP`.
3. Select a Switch  from the Toolbox of the Panel Designer and place it on the panel, e.g. via drag & drop.
4. Configure the Switch in the Properties grid under **Settings|State Count** as a control element with 2 states.
5. Assign the Switch in the Properties grid under **Settings|Image** the image file `IORGPUSHBUTTON_2.BMP` from **CANoe's** demo directory `DEMO_ADDON\BITMAP_LIBRARY\GLOBAL\Switches_2STATES`.
6. Assign the the environment variable **evSwitch** to the Switch via drag & drop from the Symbol Explorer.


7. Label the switch by selecting the Static Text **A** display element from the Toolbox and place it on the panel to the right of the Switch. Enter the label in the Properties grid of the Static Text under **Appearance|Text**.
8. 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.
9. Save the panel under the name `SWITCH.XVP`.

The panel gets the name **SWITCH** that is displayed on the opened panel's top left.

Create the second panel


The second panel has a small simple lamp as an indicating element.



1. Open a new panel with the menu item **File|New Panel** and select the file name `LIGHT.XVP`.
2. Select a Switch  from the Toolbox of the Panel Designer and place it on the panel, e.g. via drag & drop.
3. Configure the Switch in the Properties grid under **Settings|State Count** as a display element with 2 states.
4. Assign the Switch in the Properties grid under **Settings|Image** the image file `MLEDRED_2.BMP` from **CANoe's** demo directory `DEMO_ADDON\BITMAP_LIBRARY\GLOBAL\ INDICATOR_2STATES`.
5. Assign the the environment variable **evLight** to the Switch via drag & drop from the Symbol Explorer.
6. Label the switch by selecting the Static Text **A** display element from the Toolbox and place it on the panel to the right of the Switch. Enter the label in the Properties grid of the Static Text under **Appearance |Text**.
7. Save the panel under the name `LIGHT.XVP`.

The panel gets the name **LIGHT** that is displayed on the opened panel's top left.

Add the panels to the configuration

You complete this step of this unit by integrating the created panels into the **CANoe** configuration with the  button of the toolbar.

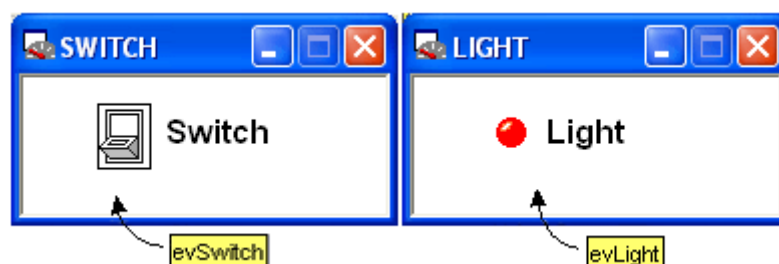



Figure 29: Panels in CANoe

Save the configuration

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

4.13.3 Creating network node models

Create 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.

Insert two network nodes

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.

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

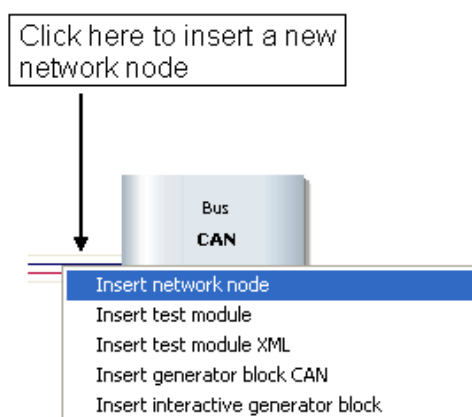


Figure 30: Inserting network nodes in the simulation setup

Configure the nodes

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.

Create the used CAPL programs

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);
}
```

Start the measurement

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**.

5 Applications

In this chapter you find the following information:

5.1	Overview of the important elements	page 56
5.2	Simulation/Simulation setup	page 58
	Working in the simulation setup	
	Simulation mode	
	Message attributes	
	System verification	
5.3	Measurement/Measurement setup	page 61
5.4	Working with configurations	page 61
5.5	Working with databases	page 62
	Use of multiple databases	
	Resolving ambiguities	
	Checking for consistency of symbolic data	
5.6	Working with multiple channels	page 65
	Channels in online mode	
	Channels in simulation mode	
	Channels in offline mode	
5.7	Working with panels and symbols	page 67
5.8	Logging and evaluation of measurement files	page 67
	Triggers	
	Data analysis	
	Data export and conversion	
5.9	Test functionality in CANoe	page 70
	Test Feature Set (TSF)	
	Test Service Library (TSL)	
5.10	Diagnostics functionality in CANoe	page 74
	Diagnostic Feature Set (DFS)	
5.11	CANoe Realtime	page 75
5.12	Standalone Mode	page 75
5.13	Macro Recorder	page 76
5.14	Step Sequencer	page 76
5.15	COM Server	page 76
5.16	Troubleshooting	page 77
5.17	List of error messages to the CAN interface	page 78

5.1 Overview of the important elements

Program start

At the program start of **CANoe** the program is called by double clicking the appropriate icon in the **CANoe** program group.

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.

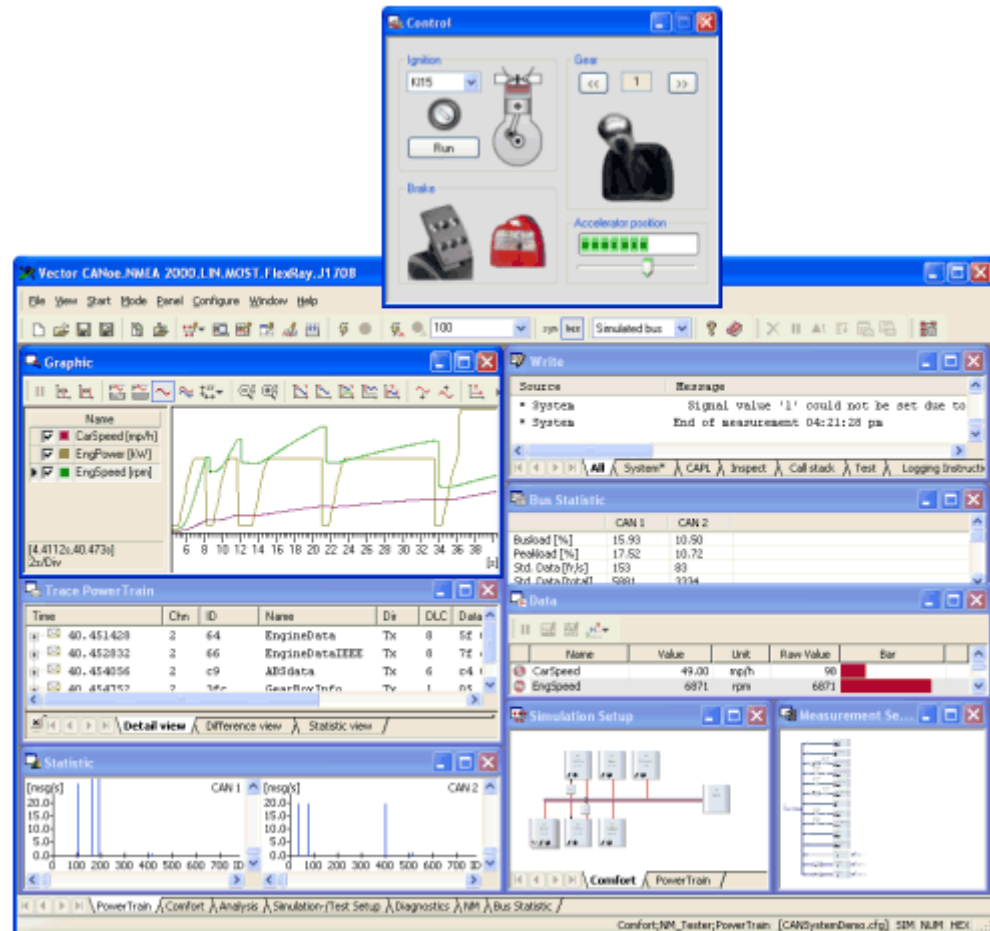
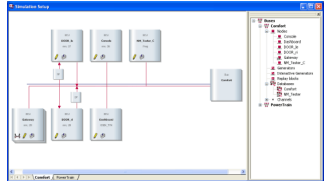

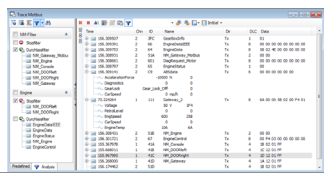
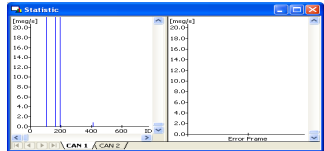
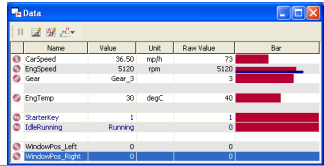
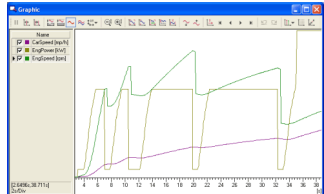
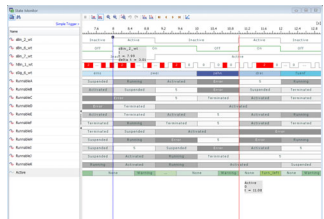


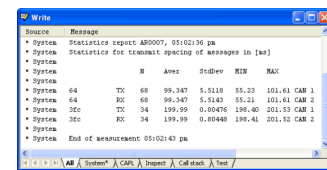
Figure 31: The CANoe screen

Central access to main features

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 are available. Particularly the data flow diagram and the function blocks in the Measurement and Simulation Setup window are directly configured with context sensitive menus. For example you can insert new function blocks such as filters or generator blocks 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 measurement and simulation setup gives you an overview of the configuration options provided by **CANoe** and shows how your actual measurement configuration appears.

Element	Description
Main menu bar	Used to select basic functions
Toolbar	Used for quick selection of important commands and also contains status indicators for the number system being used and to display of keyboard entries made during the ongoing measurement.
Status bar	The names of the active configuration file and the database being used are displayed here.
Simulation setup	In the simulation setup window the overall system is setup displayed graphically with the CAN bus and all network nodes. 
Measurement setup	The measurement setup displays the program's data flow. All options are set in this window for parameterizing a measurement or evaluation. 
Trace window	Bus activities are recorded here. The user can scroll in this window after a measurement has been completed. 
Statistics window	The mean transmit frequencies of messages are displayed as line spectra above the identifier axis in this window. As an option, the user can toggle over to mean transmit spacing. The window can be zoomed for detailed evaluations. 
Data window	Preselected data segments of messages can be displayed here. 
Graphics window	Graphic representation of signal time responses, which are displayed in a X-Y diagram above the time axis. After the end of measurement a measurement cursor and a difference cursor are provided, with which you can examine the coordinates of all measurement points or differences between two measurement points precisely. 
State Monitor Window	The State Monitor is an analysis window that shows bit values and states. It is particularly suitable for displaying digital inputs/outputs as well as status information like terminal states or network management states. 

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.

Statistics Monitor window

Statistic	Current / Last	Min	Max	Avg
Busload [%]	14.09	0.00	14.09	12.51
Min. Send Dist. [ms/s]	0.000	0.000	0.000	0.000
Engine	0.000	0.000	0.000	0.000
Gateway	2.112	0.000	9.228	2.851
NH_Tester	-	-	-	-
Unknown	-	-	-	-
Max. Bus Time [ms/s]	5.629	0.000	6.564	5.162
Engine	4.824	0.000	5.760	4.408
Gateway	0.804	0.000	1.824	0.793
NH_Tester	0.000	0.000	0.000	0.000
Unknown	0.000	0.000	0.000	0.000
Max. Frames per Burst [r/s]	5.000	0.000	6.000	4.638
Std. Data [r/s]	134	0	135	122
Std. Data [total]	7188	-	-	-
Std. Data [r/s]	0	-	0	0
Std. Data [total]	0	-	-	-
Std. Remote [r/s]	0	0	0	0
Std. Remote [total]	0	-	-	-
Std. Remote [r/s]	0	0	0	0
Std. Remote [total]	0	-	-	-
Errorframes [total]	0	0	0	0
Chip State	Active	-	-	-
Transmit Error Count	0	-	0	-
Receive Error Count	0	-	0	-
Transceiver Error	0	-	-	-

Hardware-related information such as number of data and remote frames, error frames and bus load are displayed here. Availability of this information depends on the CAN PC card being used.

5.2 Simulation/Simulation setup

5.2.1 Working in the simulation setup

Working with different buses

Via the shortcut 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 tabs at the bottom of the simulation setup window, you can easily change between the different buses of your configuration.

Bus architecture

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:

- reale bus is displayed as a black line
- simulated bus is displayed as a red line

5.2.2 Simulation mode

Simulate networks

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.

Simulation with real 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 remains set to **Real Bus**. All messages generated in the simulation setup are then placed on the real bus.

Simulation without 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 | Options...**) from **Real Bus** to **Simulated Bus**. Bus access (sending and receiving messages) is then simulated.

Working in simulation mode

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. You can resume the simulation for a defined period of time or stop a running simulation at any point in time.

Slave mode

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.



Info: 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 configure in the card's configuration dialog

5.2.3 Message attributes

Attribute Rx

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.

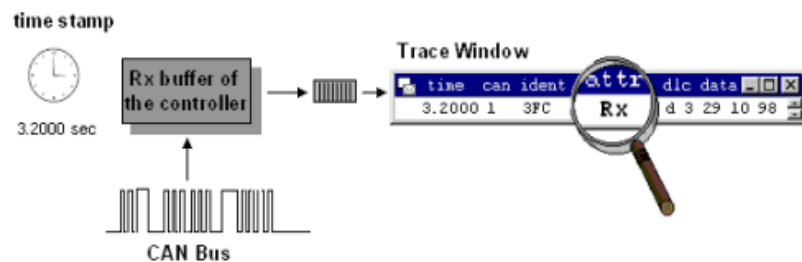


Figure 32: Receiving messages

Attribute Tx

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.

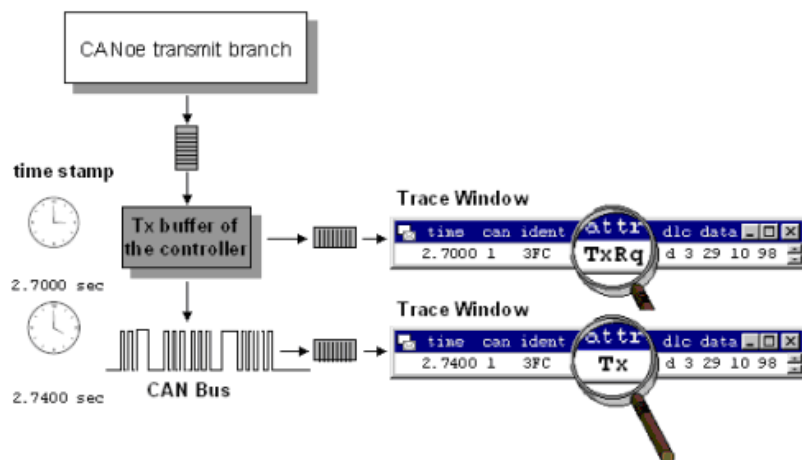


Figure 33: Transmission of messages

Attribute TxRq

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 shortcut 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.

Latency time

The TxRq display permits measurements of the difference between the time of the transmit request and the time of the 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:

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

5.2.4 System verification

Rules of the system verification

You can start a system verification by selecting the **System verification** command in the shortcut menu (right mouse button) of the system view.

The System verification checks your simulation setup according to the following rules:

Rule 1

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".

Rule 2

All gateways that are defined in the databases must be included as gateways within the simulation setup.



Example: The databases "Comfort" and "Power Train" both contain the network node "Gateway" and are assigned to the buses "Comfort" and "Power Train" 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.

Rule 3

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 gate-way – 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.

5.3 Measurement/Measurement setup

How to start a measurement

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.

Configuration during measurement

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 shortcut 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).

How to stop a measurement

You can stop the measurement

- by pressing the <ESC> key,
- selecting the main menu item **Start | Stop**,
- activating the  button on the toolbar or
- by using internal events (e.g. CAPL or trigger).



Info: 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.

5.4 Working with configurations

Save settings

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**.

Change configurations

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.

Create project directories

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.

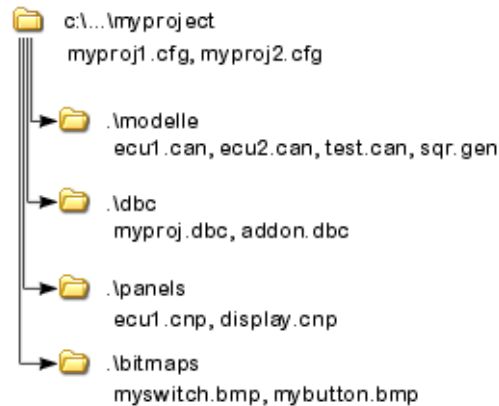


Figure 34: Example of a directory tree for a CANoe project

Reference to project files

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.

Document and archive configurations

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.

5.5 Working with databases

Working with symbolic data

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. Afterwards you can access the information in measurement windows, insertable function blocks and CAPL programs.

CANdb++ Editor

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

Messages

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**.

Signals

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.

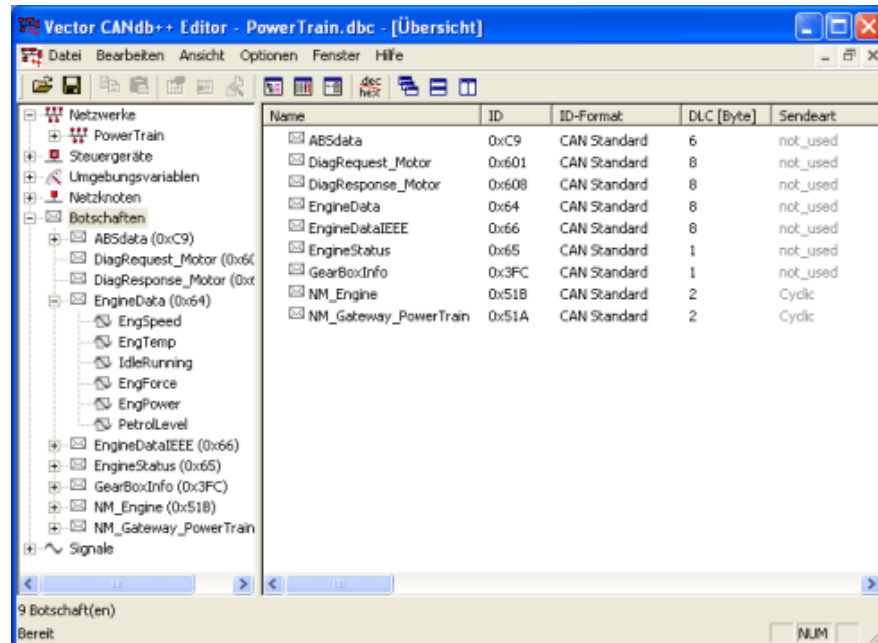


Abbildung 35: Symbolic description of the message EngineData in CANdb++

Symbol Selection dialog

The Symbol Selection dialog shows all signals and messages defined in the database. You can choose one or multiple objects there to display or manipulate them. If the database has been filled out completely, you can search in the Symbol Selection dialog for e.g. a node, a message or even a list of all signals in the database.

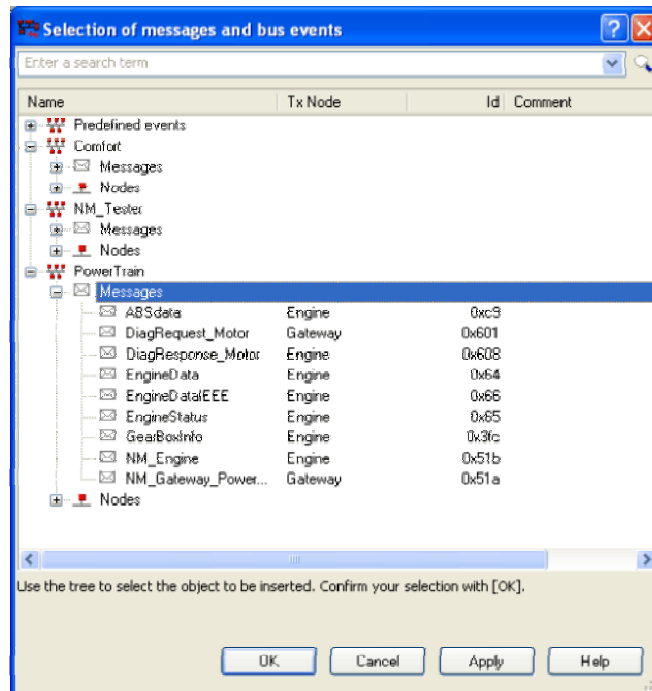


Figure 36: Symbol Selection dialog



Further Information: Please refer to online help for further instructions on working with databases.

5.5.1 Use of multiple databases

Administration of large systems

With large systems it may be sensible to distribute the descriptions of messages and signals as well as environment variables to several partial databases. Also when operating **CANoe** with two buses it makes sense to describe each system by its own database.

Assignment of databases

CANoe supports the simultaneous use of multiple databases. The assignment of databases must be made in the simulation setup. Afterwards, you can use symbolic names in all function blocks and in CAPL for the messages, signals and environment variables of all databases. To do this, enter the symbolic name in the appropriate input box. You will find a list of all symbolic names in the Symbol Selection dialogs. The list is opened by activating the small buttons located next to the appropriate input boxes. Then you can select the desired symbolic name from this list.

Ambiguities

If you are using more than one database, the messages in the databases following the first database are qualified with the database name. However, you only need these qualified names to resolve ambiguities. As long as the symbolic names are unique in all databases, you can forego qualification of symbolic names in all function blocks and when editing CAPL programs.

5.5.2 Resolving ambiguities

Resolve ambiguities	With the use of multiple databases it is essentially possible to have ambiguities in the use of symbolic names, which must be resolved by the program.
Conflict	Messages coming from the bus and are registered by the program over one of the two CAN controllers may have two different symbolic names in two databases.
Solution	Ambiguities of this type can be resolved by the order in which the databases were specified in the list of the Database dialog. You also have the option of assigning a prioritized database to each of the two CAN controllers. For messages that are received by this controller, this database will then have the highest priority for symbol assignment. Only if no symbolic name is found there will the program search through all additional databases specified in the database drop-down dialog in the order that is specified there.
Conflict	The user may wish to configure function blocks or measurement windows with different messages which have the same name in different databases.
Solution	The search sequence of the database list is also utilized to resolve name conflicts when configuring measurement windows and function blocks. In this case the database uppermost in the list is assigned to the name. However, you also have the option of resolving ambiguities of this type by qualifying the symbolic name.

5.5.3 Checking for consistency of symbolic data

Consistency check	<p>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:</p> <ul style="list-style-type: none">→ At the program start→ When a new database is associated→ When CANoe is restarted after a change to the database
Procedure	<p>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.</p>

5.6 Working with multiple channels

Channel definition	<p>The number of CAN channels you wish to use is configured in the channel definition dialog. CANoe supports up to 32 (virtual and real) channels. 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.</p> <p>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.</p>
--------------------	--



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

5.6.1 Channels in online mode

Working with real buses

In online mode with a real bus 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.

Consistency check

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.

Behavior at inconsistency

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.

5.6.2 Channels in simulation mode

Working with simulated buses

In online mode with a simulated bus buses and network nodes are fully simulated. Each channel corresponds to a simulated bus.

Behavior at inconsistency

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.

5.6.3 Channels in offline mode

Working with logged data

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.

Behavior with inconsistency

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.

5.7 Working with panels and symbols

Create your own user control interfaces

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. External events are described with the help of symbols, whose name and type are defined in the database. Therefore, symbols can also be interpreted as I/O interfaces between network nodes and their peripherals, i.e. as connection between the particular CAPL program and its input and output elements on the panels.

Control element

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

Display element

CAPL programs can change the values of associated symbols when certain events occur. This value change can then be visualized on the panel using display elements.

Window management concept

For measurements and simulation runs, usually only a small subset of all panels is needed at the same time. Therefore, **CANoe's** window management concept 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.

5.8 Logging and evaluation of measurement files

Logging file

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 block** 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.
- Trigger conditions** 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.
- Analyze logging files** 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 measurement and evaluation functions of online mode are also available to you in offline mode.

5.8.1 Triggers

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

→ **Single Trigger**

A specific event triggers the logging

→ **Toggle Trigger**

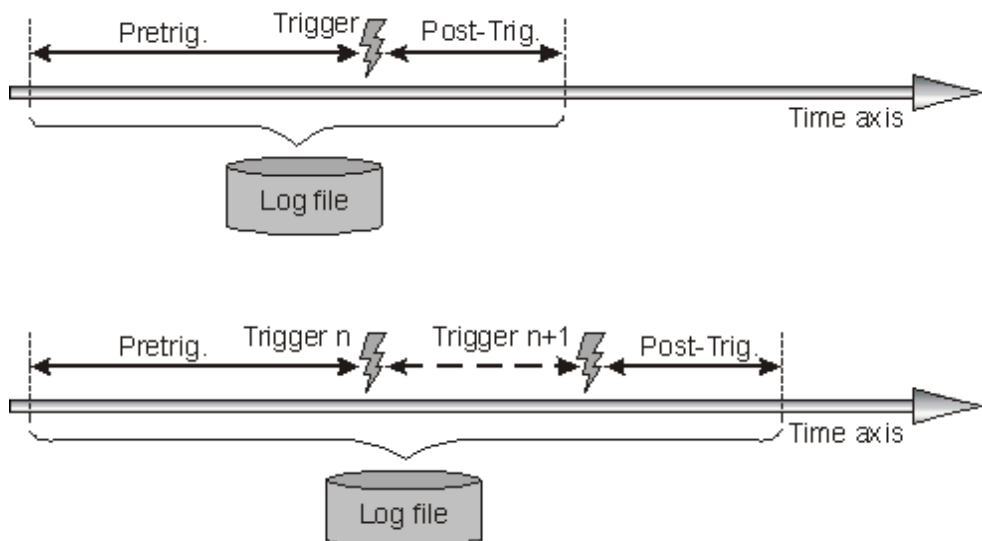
Specific events determines start and stop of the logging

→ **Entire Measurement**

The entire measurement is logged



Example: Time window for the trigger in Single Trigger mode (top) and in Toggle Trigger mode (bottom).



Single Trigger

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 pretrigger and post-trigger times, and the number of triggers you wish to log, in the **Time** section.

Toggle Trigger

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.



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

```

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 10
    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 0A
    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 0A
    9.1151 1 ABSdata Tx d 3 62 00 00
    9.1300 1 ErrorFrame Tx d 3 65 00 00
    9.1300 log trigger event
    9.1651 1 ABSdata Tx d 4 98 76 28 00
    9.1771 1 EngineData Tx d 8 73 09 00 00 00 00 72 08
    9.2004 1 WheelInfo Tx d 3 68 00 00
    9.2151 1 ABSdata
  End Triggerblock

```

Logging of an entire measurement

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.

Trigger conditions

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

→ Start

Triggering occurs at the start of measurement.

→ Stopp

Triggering occurs at the measurement stop.

→ CAPL

Triggering is by a CAPL program.

→ Benutzerdefiniert

The occurrence of a user defined condition initiates triggering.

5.8.2 Data analysis

Analyze recorded data	To study recorded log files switch CANoe to offline mode. The data source in offline mode is a file, e.g. generated by logging in online mode. Analogous to online mode, all measurement and evaluation windows are also available to you in offline mode. The only option omitted is the possibility of sending data over the bus. Furthermore, offline mode provides a powerful search and break function, with which you can intentionally stop the replay of the log file. 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.
Functions for flow control	The following functions 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.
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 slowmotion 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.
Break	In offline mode this menu item interrupts the replay of data from the source file (animate run). A restart resumes the replay at the point where it was interrupted by Break .
Step	This menu item 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.

5.8.3 Data export and conversion

Signal-based logging export	The contents of log files can be exported or converted to other file formats with the help of signal-based logging export. The export can be limited to specific signals. Additionally you can define programs that will be started after an export.
Conversion	Conversion of log files is supported in both directions, i.e. ASCII to binary and binary to ASCII.

5.9 Test functionality in CANoe

Test functionality	<p>CANoe supports testing of controllers and networks by special test functionalities in all development steps. These can be used to create tests, verify individual tests, test prototypes, or perform regression and conformity tests.</p> <ul style="list-style-type: none">➔ Design of the test functionality (system simulation)➔ Implementation of test functionality in an ECU (remaining bus simulation)➔ Specification-/Integration-/Regression tests➔ Analysis of real ECU communication➔ Perform diagnostics of ECUs with integrated tester functionality➔ Error search
--------------------	--

5.9.1 Test Feature Set (TSF)

Test procedure	CANoe supports testing of controllers and networks by special test functionalities that are called Test Feature Set. These can be used to create tests, verify individual tests, test prototypes, or perform regression and conformity tests.
CAPL	Sequential test procedures can be described in CAPL or defined in XML and may be executed at any time during a measurement. CAPL recognizes special functions used to formulate tests and can wait for specific events such as the receipt of a message.
XML	Within XML files you can build and parameterize tests with help of predefined test pattern. Parallel to test execution defined conditions can be monitored, e.g. conformance to cycle times.
Test report	The results of a performed test are captured in a test report. The report is written to a XML file but is also available in HTML format. Much of this report is user-configurable.
Perform tests	You have the opportunity to implement existing test scenarios that are usually sequential with a linear flow in CAPL or you can use predefined test sequences (test pattern) that are set up with XML files. The former purely event-oriented execution paradigm has been expanded to include a sequential execution model for test modules. The test procedures of the individual test modules are executed in parallel, while the flow within each test module is executed sequentially.
Test module	<p>The test module has a beginning (start of the test sequence) and an end (end of the test sequence). Outside of these times the test module is not running, i.e. it processes neither the test sequence nor existing event procedures. During a measurement a test module can be executed consecutive several times. The previously event procedures that inform a CAPL node about the start and end of the measurement (<code>on preStart</code>, <code>on start</code>, <code>on end</code>) are unnecessary in test modules and are also unavailable.</p> <p>At the beginning of the test module the variables are set to their initial states. The contents of a variable cannot - by definition - be transferred from one test procedure to another.</p> <p>By default, the test module is assigned to all buses (corresponds with a gateway node that has connection to all buses). If necessary, the test module can be assigned to one or more to specific buses via the shortcut menu.</p>
Test modules in XML	If XML test modules are created, the XML file contains the test parameters. The XML file consists essentially of the specification of the test parameters. Programming of the procedures is not necessary since the tests are defined by a query of parameterized and predefined test patterns.
Test parameter in CAPL test modules	<p>Examples of ways to input test parameters in CAPL test modules:</p> <ul style="list-style-type: none"> → Fixed coding of parameter in CAPL → Read-in of a file at the beginning of the test module → Parameterization by means of environment variables

Procedure of a test module

A test module may be started either automatically with the measurement start or interactively by the user. User inputs may be requested and processed during execution of a test module. User interaction is helpful if, for example

- the test sequence/SUT should be influenced by the user, or
- manually obtained observations by the user are to be incorporated into the test.

Once a test module has been started it terminates normally at the end of the test sequence or is terminated by force at the end of the measurement. The user can activate/deactivate a test module before the beginning of the measurement (recognizable with the corresponding check box in the test setup). During the measurement a deactivated test module can not be started, as well the automatic start function is out of order. Execution of the test module generates a test report. The test report is saved as a file in XML format, and if desired may also be saved as a HTML file.

Create and manage test environments

In test setup test environments can be created and managed that are independent of the **CANoe** configuration. The test setup is represented in the Test Setup window (see chapter 6.14) which is comparable to the simulation setup that describes the simulation. In **CANoe** there is exactly one Test Setup window in which several test environments can be loaded. A test environment consists of any directory structure that enables the grouping of test blocks.

The following blocks can be inserted into the test setup:

- Test block
- XML test block
- CAPL nodes (network nodes)
- Generators (Generator, IG for CAN, IG for MOST)
- Replay blocks (for all bus systems)

Edit the test environment

In each directory, the test environment can be edited using the shortcut menu. The individual nodes are parameterized and checked via their respective shortcut menus. The nodes and directories can be moved or copied into the structure at any time using drag-and-drop.

Save the test environment

Each test environment is stored in an individual file (*.tse - **T**est **S**etup **E**nvironment) and can thus be loaded or unloaded independently of the **CANoe** configuration (simulation and analysis window).

Test blocks or entire directories can be activated and deactivated individually. It is also possible to export partial structures of an existing test environment into a new file or to import an existing test environment at any point in a test environment.



Info: Even if the test environment can be used in different configurations, only one file exists in which the information is stored. A change of the test environment in a configuration thus affects all other configurations.

Insert a test module in the test setup

A test module is inserted in test setup using the shortcut menu or in simulation setup using the shortcut menu of the simulated bus. A test module in test setup is thus always assigned to all buses.

Execute a test module

The execution of test modules defined in XML files can be controlled via a dialog that can be reached using the shortcut menu of the test module. The execution of a test module is documented in the Write window (e.g. at the beginning and end of the execution of a test case, results of test cases and test modules). In some cases, this leads to a lot of information in the write window.

5.9.2 Test Service Library (TSL)

Integration of test function

With **CANoe** you can monitor cycle times of messages, response times of ECUs to the reception of a message until the transmission of the response message or the validity of signal values in messages.

Quality statements for tested ECUs can be derived from the statistic results of the tests (e.g. from the number of reported divergences in the period of time).

Checks

For these **CANoe** offers special functions (checks) that are integrated into the Test Service Library. The functions can be used in CAPL test modules and XML test modules. Details for the test specifications can be taken partially from the CANdb++ data basis in that information like target values for cycle times can be defined. With this functionality the necessary test code is very small.

Stimuli

In the Test Service Library you can find functions that make the stimulation of the ECUs easier (stimuli functions). For example you can create different signal forms with these functions.

Test design approaches

Tests can be specified either in one global test node where all the test functions are located or for every node (real or simulated) there is a node that contains the tests for this node. Therefore it is very simple to activate or deactivate the test for one node at a time.

Test targets

Users usually follow one of the following targets when using the TSL:

- The implementation of an ECU is checked against a specification every time the ECU software is changed. This is the case at the supplier that develops the software for an ECU that has to conform to an OEM specification.
- A user interacts with a predefined test evaluating the behavior of an ECU. For example the OEM receives a prototype from a supplier and tests its compliance towards parts of the specification.
- Here the results should display statistics on the conformance of the different test parts, e.g. a statistical value for the timer precision, is requested.
- The test is run without user interaction, only the results are reported.
- The result has to be displayed in a simple and clear form, e.g. in regression test of the ECU software.

User types

For the Test Service Library functions the following fundamental user types can be identified:

- Test developer

This type of user has control over the CAPL test program and is willing to analyze the write output carefully. The system should bring errors or inconsistencies of the test specification to his attention and help him to correct it.

- ECU tester

These users only apply existing tests on ECUs, without the possibility to change the test specification or the SUT (black box). The user trusts in the correctness of the test, i.e. no errors and inconsistencies are expected and if the system reported some, the user would not be able to profit from them. Also, the user expects a lenient behavior of the tests regarding user interaction, e.g. the user should not have to fear to abort a measurement or render it futile simply by querying a check state.

→ ECU developer

Users of this type have access to the SUT directly (white box) change the software of ECUs and run the test to see if the ECU conforms to the test specification. They have designed the tests themselves and may want to be informed about all errors in the test configuration at one point (like a test developer), and be informed only of the behavior of the SUT at another point (like an ECU tester).

Robustness

For the robustness of the system, two cases have to be distinguished:

→ Errors in the test specification and parameterization: these situations appear because the test is not specified properly or the data configuring it (like a database) does not contain the information expected or required.

In this case the system has to provide enough information for the test designer and user to identify and solve the problem. It must be made sure that the system does not report an unreliable result, i.e. "no error detected" may not be reported if the test itself did not work.

→ Errors produced by the SUT: these errors are exactly what the test has to detect and therefore the system must be very robust here.

5.10 Diagnostics functionality in CANoe

Diagnostics in CANoe

CANoe can be used in all steps of developing ECUs and performing diagnostics on them:

- Design of the diagnostic functionality (system simulation)
- Implementation of diagnostic functionality in an ECU (remaining bus simulation)
- Specification-/Integration-/Regression tests
- Specification of tests with XML
- Analysis of real ECU communication
- Perform diagnostics of ECUs with integrated tester functionality
- Error search

5.10.1 Diagnostic Feature Set (DFS)

Diagnostics functions Vector Informatik's Diagnostic Feature Set includes several functions that are necessary for development, test and application of ECUs with/via diagnostics.

Diagnostics console Based on the diagnostic description files (*.CDD) of **CANdela Studio** the Diagnostics console (see chapter 6.12) — part of the Diagnostic Feature Set — provides interactive access to all diagnostic services. Diagnostic requests can be selected, parametrized and displayed with their according response. The Fault Memory window (see chapter 6.13) provides quick and easy access to the fault memory of an ECU.

Diagnostics in CANape and CANdito Apart from **CANoe** the Diagnostic Features Set is also included in the Vector products **CANape** and **CANDito**. Thereby the complete development process is supported identically.

5.11 CANoe Realtime

Real-time simulation	<p>CANoe offers the opportunity to execute the real-time relevant simulation parts on an individual computer that is separated from the graphic interface.</p> <p>Thus on the one hand, the total performance of the system can be expanded more easily if necessary; on the other hand, influences above all of the graphic system on latency times and timer precisions are prevented.</p>
Advantage	<p>The configuration of the simulation and the evaluation then occur on the usual workstation computer, for example, while the simulation runs on an individual computer.</p>
Components:	<p>CANoe is divided into 2 components for the uncoupling of the real-time operation.</p>
The real-time part	<ul style="list-style-type: none"> → The real-time part executes the simulation of the model that is especially the CAPL programs. → All bus hardware is connected here. → A small "runtime kernel" runs on the real-time computer, which manages entirely without a graphic interface. → The real-time part is executed on an individual computer with the Windows operating system. <p>Support for other operating systems is under construction.</p>
The evaluation part with the graphic interface	<ul style="list-style-type: none"> → The evaluation and display of the data stream generated by the simulation or read by the bus hardware occurs on this computer. → The evaluation part is typically executed on the usual workstation computer. → The application of CANoe then occurs from the evaluation part in as much the same manner as possible as in standard operation with one computer.
Connection	<p>These two computers are connected with one another via TCP/IP – typically via Ethernet. The TCP/IP network connection is uncoupled on both sides by a data buffer so that the real-time part is not dependent on the quality of the connection.</p>
Transmission	<p>All data for the configuration of the simulation or measurement is transmitted automatically to the real-time computer at the start. During measurement, all evaluation and logging data is transmitted extremely promptly to the evaluating computer. Keyboard, panel, and any other user interactions are also transmitted directly to the real-time computer during the simulation or measurement.</p>

5.12 Standalone Mode

Standalone Mode	<p>In standalone mode the realtime-specific portions of a configuration is executed on an external device (e.g. VN8900 and CANoe RT Server) without a connection to the user PC. You can set up the device so that measurement begins as soon as it is switched on. You can also interact using a keypad attached directly to the device to stop and start measurement.</p> <p>Evaluation-specific portions of the configuration (in particular, any blocks defined in the measurement setup) are not included in measurements carried out in standalone mode.</p> <p>You can manage standalone mode either via a dialog in CANoe or via the Standalone Manager.</p>
-----------------	--

Standalone Manager The **Standalone Manager** is an accessory included with **CANoe** that you can install on any other PC. You can use it to control standalone mode on any **VN8900** devices connected to that PC.

5.13 Macro Recorder

Working with macros Macros permit reproducible and automated testing.

With the help of the macro recorder in **CANoe** macros can be made to record and replay user actions on panels and diagnostic consoles. Environment variable changes are also recorded while recording user actions on panels. If the actions of Diagnostics console and the Fault Memory window are recorded, then diagnostics data sent to the controller are included in the recording.

Recording

Once you have started to record the macro every action you execute on the panels with the mouse, and every request sent via the diagnostic consoles is recorded.



Info: Macros are being used to influence simulations (not analyses). Therefore macros can only be recorded and executed in online mode.

5.14 Step Sequencer

Graphical Editor With the Step Sequencer you can easily create simple sequences for network stimulation and application control graphically.

You can structure the steps, e.g.

- set and verify signal values and system variables,
- send messages,
- wait,
- start replay files and signal generators

in loops and conditional blocks (If, Else If, Else, End If).

Visualized Sequence Each sequence is displayed in a separate window and can be edited even during a running measurement. After starting the sequence a cursor shows the progress.

Interaction

You may set wait points where the sequence will be paused.

The network responses can be reported automatically by using commands for (textual) output to the Write window or in a file.

5.15 COM Server

Access via different applications For the communication with other applications **CANoe** offers the COM server. 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.

Controlling

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++.

5.16 Troubleshooting

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 shortcut menu of the CAN-card icon.

5.17 List of error messages to the CAN interface

Error messages with assigned error numbers	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 state of the controller, or due to the accumulation of too many higher priority messages.
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 <code>CANn . . .</code> in the CAPL programs, where n is greater than the number of existing controllers. This must be replaced by a correct number.
Incorrect checksum (1368)	The CAN controller detected a faulty CRC checksum.
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.
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.
Remedy	There are several methods for remedying this situation: <ul style="list-style-type: none"> ➔ 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.

Remedy

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.

6 Windows

In this chapter you find the following information:

6.1	Desktop concept	page 82
6.2	Window management	page 82
6.3	Simulation Setup window	page 83
6.4	Measurement Setup window	page 84
6.5	Trace window	page 87
6.6	Graphics window	page 88
6.7	State Monitor window	page 90
6.8	Write window	page 91
6.9	Data window	page 91
6.10	Statistics window	page 92
6.11	Statistics Monitor window	page 94
6.12	Diagnostics Console	page 95
6.13	Fault Memory window	page 96
6.14	Test Setup window	page 96

6.1 Desktop concept

Organize windows on different desktops

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

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

Window definitions

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

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

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

Place windows

The windows opened on different desktops may be placed inside or outside of the program window, or they may be docked in the program window. The integration of panels into **CANoe's** window management concept allows panels to be positioned in a desktop-based manner, both inside and outside of **CANoe**. This allows you to design a desktop and the spatial layout of windows/panels in an expansive framework.

6.2 Window management

Integrated windows

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

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

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

MDI windows

- Panels can be of the **MDI** or **standard** window type.
- Windows/Panels of the **MDI** type are located within the program window and may be minimized.

Docked windows

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

Floating windows

Floating windows are created by the **docking** window type:

- When you drag a docking window from the program window to the windows desktop the window automatically becomes a floating window.
- If you drag a floating window to the program window it is automatically converted to a docking window.
- You can avoid this behavior by keeping the <Shift> button pressed while dragging the window.
- Windows of the floating window type may be moved by mouse anywhere on the windows desktop independent of the program window, and they always appear in the foreground.
- When the program window is minimized the associated floating windows of the application are also minimized.
- Floating windows cannot be addressed by the <Alt>+<Tab> key combination.

Standard windows

- Panels of the **standard** window type may be moved by mouse anywhere on the windows desktop independent of the program window.
- An active program window may overlap panels that are in this state.
- Additional characteristics include the appearance of the windows desktop on the task bar, and control by the <Alt>+<Tab> key combination.
- When the program window is minimized panels of the **standard** type are not minimized.

6.3 Simulation Setup window

Graphical view of an overall system

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 chapter 3.1. 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 shortcut menu.

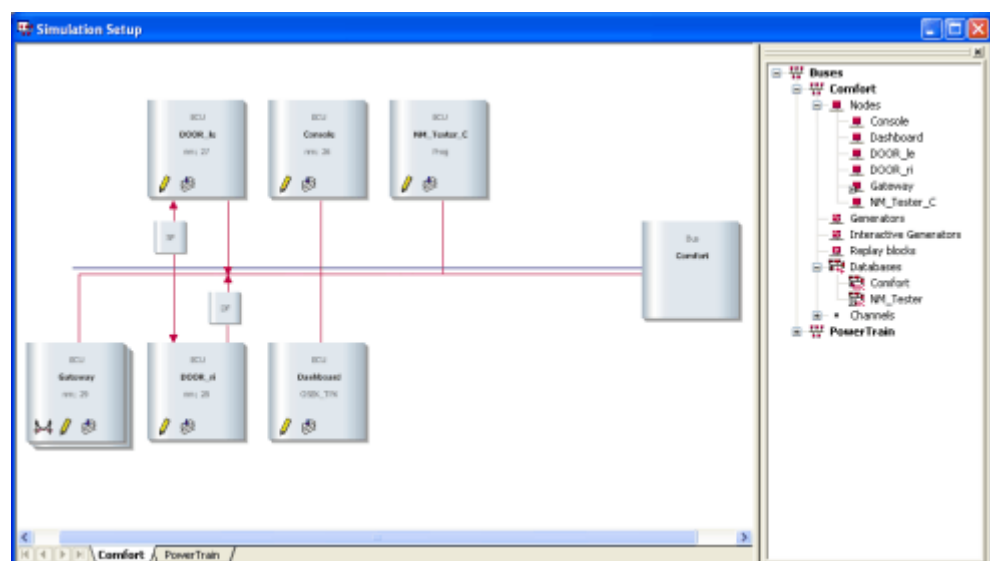


Figure 37: CANoe simulation setup

Real/simulated bus	The block for the PC card can be seen on the right side of the simulation setup. The simulated bus (displayed as a red line) and the real bus (displayed as a black line) are connected to the card block.
Insert nodes	<p>If you click the bus symbol with the right mouse button you get the popup menu for the two buses (simulated and real). By repeated selection of the menu command Insert network node in the shortcut 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.</p> <p>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... in the node's shortcut 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.</p>
Real/simulated nodes	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.
Switch over simulated/real	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 active (simulation) to inactive (real-time mode) and back again.

6.4 Measurement Setup window

Overview	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").
Graphical view of a measurement setup	The data flow diagram of the measurement setup contains data sources, basic function blocks, hot spots, inserted function blocks and data sinks. Connection lines and branches are drawn in between the individual elements to clarify the data flow.
Data source	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.
Evaluation branch	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.
Evaluation blocks	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.

Evaluation window

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".

Hot spots

Located between the function blocks are insertion points (hot spots), 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.



Further Information: You will find a description of all insertable function blocks in the online help.

Example configuration

Figure 38 shows a possible **CANoe** configuration in online mode, 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.

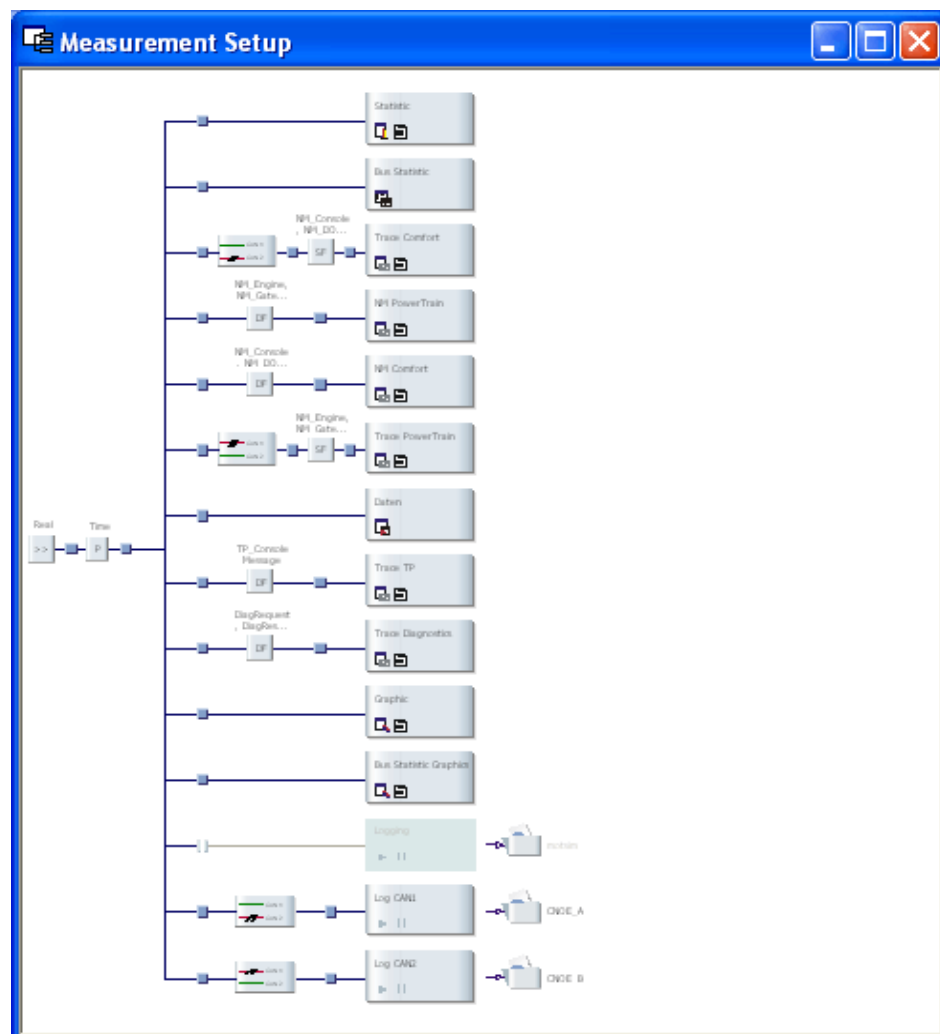


Figure 38: CANoe measurement setup



Info: 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.



Info: Data flow and functions in online and offline modes only differ in the data source and in the transmit block. Refer to chapter 5.8.2 for a description of offline mode.

Configure CANoe

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**.

Insert blocks and filter

You configure the measurement setup or its blocks with its shortcut menu. On this way new function blocks such as filters or generator blocks can be inserted at the black rectangular insertion points (hot spots) in the data flow.

Deactivate blocks and filter

If you wish to exclude a function block from the measurement, you can deactivate it before the measurement with the spacebar or with **Block active** in the shortcut 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 shortcut menu item **Block active** again.

Size of the Measurement Setup window

The measurement setup can be displayed in two different modes:

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

Arrangement of evaluation blocks

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.

Insert evaluation blocks

To insert new evaluation blocks in the measurement setup, click the branch with the right mouse button and select the new window from the shortcut 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.

Delete evaluation blocks

You can also delete the block from the measurement setup via its shortcut 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.

Open evaluation block's window

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 shortcut menu. Multiple windows of the same type are shown cascaded in the standard layout.

6.5 Trace window

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

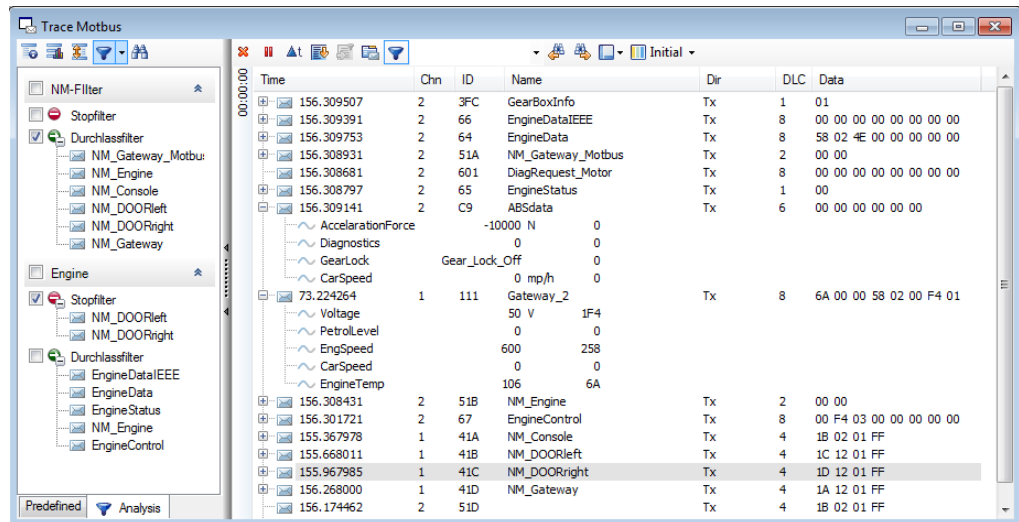


Figure 39: CAN Trace window

Display of events A number of other events are output in the Trace window, e.g.:

- ➔ Error events
- ➔ System variables and environment variables
- ➔ Transport protocol messages
- ➔ Diagnostics services

View modes The Trace window offers different view modes to observe the bus traffic:

- ➔ In **Chronological Mode** each new row is inserted below the previous row, and when the window is full it scrolls upward.
- ➔ In **Fixed Mode** each message (type) is assigned to a specific line the first time it occurs, and all further messages of the same type are written to the same line.

Analysis features The Trace window provides various features for online and offline analysis of the bus traffic, e.g.:

- ➔ Display and column filters
- ➔ Multiple search functions for strings, conditions and pattern
- ➔ Typical sort features for each column
- ➔ Import/Export of Trace data from/to log files
- ➔ Spezielle Ansichten für Busereignisse:
 - ➔ Detail view
 - ➔ Difference view (for direct comparison of different events)
 - ➔ Statistic view (several events can be selected and examined statistically)

6.6 Graphics window

Display signal responses

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.

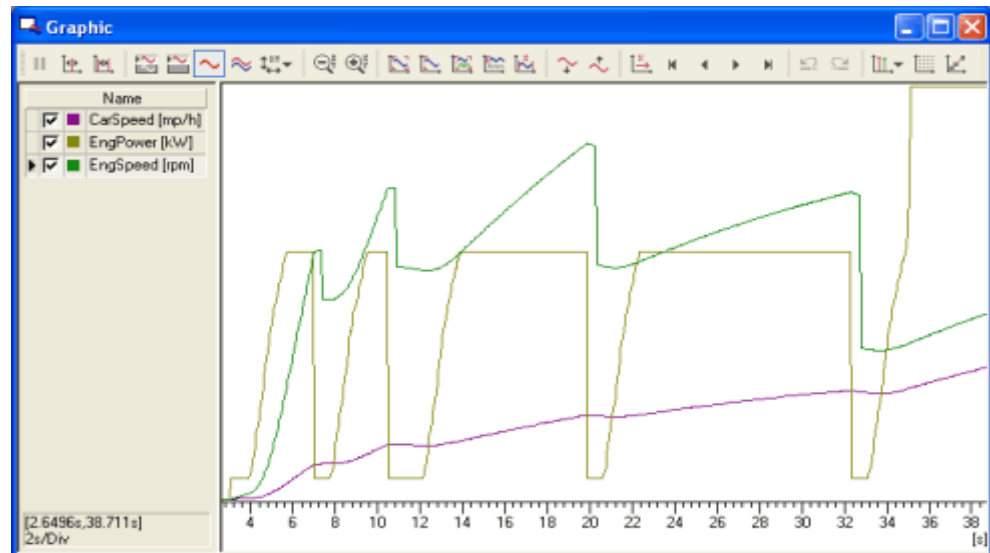


Figure 40: Graphics window

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 shortcut menu and can be enabled or disabled from there.



Info: Besides displaying signals, the Graphics window also offers the option of viewing the time responses of environment variables and diagnostics parameter. All of the statements made below regarding signals also apply, in principle, to environment variables and diagnostics parameter.

Measurement cursor

In **point measurement** mode 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 content of this rectangle is then displayed magnified when the mouse button is released (zoom). While the mouse button is held down a small square is visible which highlights the next closest measurement value. The measurement time, signal name and value are shown in the upper legend for this measurement point. In the legend with signal names, the signal values of all signals are displayed for this particular time point. The measurement cursor considers single-signal or multisignal mode. In single-signal mode the small box only jumps to measurement points of the active signal; in multisignal mode the box jumps to the next closest measurement point of all signals.

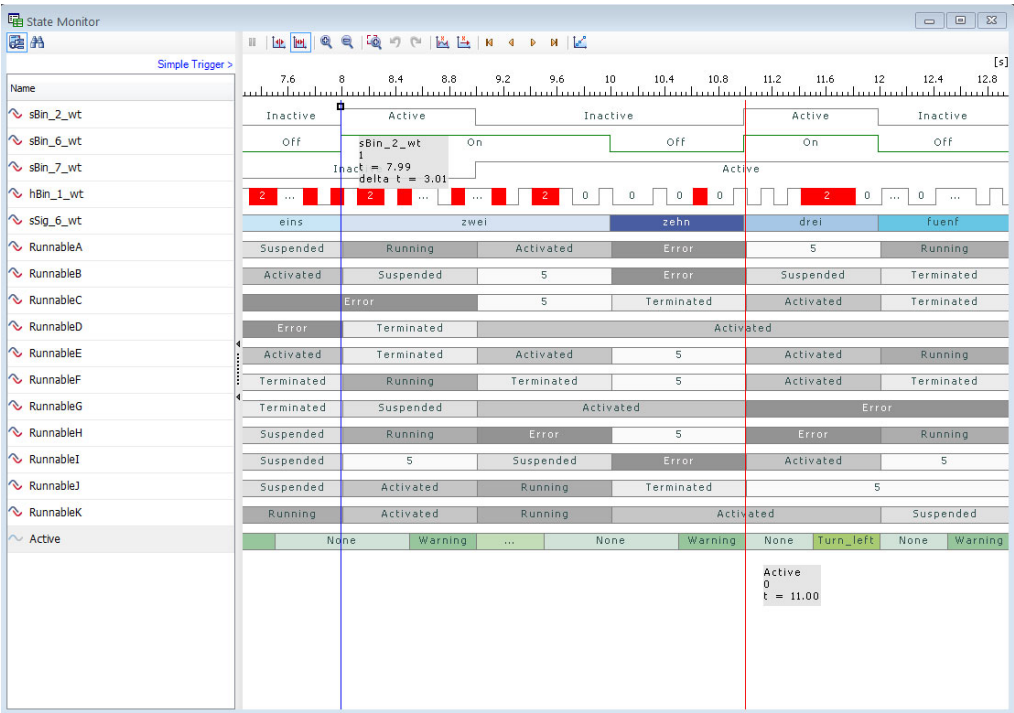
Difference markers	<p>To evaluate the in measurement value differences between two points in time, you use the difference measurement mode. The measurement cursor and a difference cursor (vertical lines in the window) are displayed. If difference mode is enabled, the cursors are shown at their active 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. 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.</p>
Window layout	<p>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:</p>
Fit all	<p>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.</p>
Zoom-in/ Zoom-out	<p>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 XY mode).</p> <p>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 Graphic configuration dialog.</p>
Fit	<p>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 XY mode, and single-signal or multisignal mode) are taken into consideration. This fits the entire graphic optimally in the window.</p>
Colors	<p>Here you select the background color for the window (white or black). Furthermore, you can open the Options dialog for configuring signal colors.</p>
Export signals	<p>With the help of the Export... 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.</p>

6.7 State Monitor window

Display states

The State Monitor is an analysis window that shows bit values and states. It is particularly suitable for displaying digital inputs/outputs as well as status information like terminal states or network management states.

Screenshot



Display and arrangement

Compared with the Graphic window states are displayed without Y-axis as rectangles or binary images. In the measurement range the numeric or symbolic value of each symbol (bus signal, environment variable or system variable) is shown. Thereby many symbols can be arranged space-saving on top of each other.

Configuration

You can configure the State Monitor in the measurement setup. You may insert multiple State Monitor windows.

Visualization

Bit signals are displayed as binary images. The high level represents the active state of the bit signal (1), the low level represents the inactive state (0).

For all other symbols the states are visualized as rectangles. Within the rectangle the numeric or symbolic value is shown. Each state may have a different fill color assigned. Colors may also be assigned to associated value ranges to color-code undershooting/overshooting of limit values. If a value table is assigned the first 11 entries of the value table will have a unique color each, all further values are initially white.

Navigation

The navigation bar is on the left of the window. You can show/hide it by clicking on the split bar. With the buttons in the upper part of the bar you can switch between different views.

6.8 Write window

Functionality

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 baudrate, 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.

You can copy the contents of the Write window to 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.

View modes

The Write window offers the following different views:

- ➔ All (shows all messages)
- ➔ System
- ➔ CAPL
- ➔ Inspect
- ➔ Call stack



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

6.9 Data window

Display of signal values

Data windows are used to display signal values (e.g. engine speed for automotive CAN buses). When a symbolic database is used the values of signals specified in the database are even displayed directly in physical units. For example, the engine speed might be viewed in RPM or temperature in degrees celsius.

	Name	Value	Unit	Raw Value	Bar
	CarSpeed	36.50	mp/h	73	<div></div>
	EngSpeed	5120	rpm	5120	<div></div>
	Gear	Gear_3		3	<div></div>
	EngTemp	30	degC	40	<div></div>
	StarterKey	1		1	<div></div>
	IdleRunning	Running		0	<div></div>
	WindowPos_Left	0		0	
	WindowPos_Right	0		0	

Figure 41: Data window

Default settings

By default the display shows the signal name, physical value, units, raw value and progress indicator (Bar) for the physical value. You can configure the displayed columns from the table header's shortcut menu.



Info: Besides displaying signals, the Graphics window also offers the option of viewing values of environment variables and diagnostics parameter. All of the statements made below regarding signals also apply, in principle, to environment variables and diagnostics parameter.

Display of signal value's changes

The signal values received in the data block of the last message remain visible in the data window until they are overwritten by new values. If a message is logged with an unchanged signal value, the activity indicator moves in the first column. If the indicator does not move, then the displayed signal value is not current, since the associated message was not received.

Min. and max. signal value

It is easy to miss minima and maxima if signal values are changing very quickly. Therefore, **Min** and **Max** columns may be added from the table header's shortcut menu. The minimum and maximum values over the entire duration of the measurement are displayed by default.

To make brief peaks easier to recognize, you can define a time interval after which the Minimum and Maximum should be reset to the momentary value. You can open the configuration dialog for setting this time interval from the **Times...** menu item of the data window's shortcut menu.

6.10 Statistics window

Functionality

The Statistics block fulfills two different functions:

- ➔ Display of 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.

The Statistics window displays the mean message rates existing at the end of the measurement. The Write window contains the statistics report (see Figure 43).

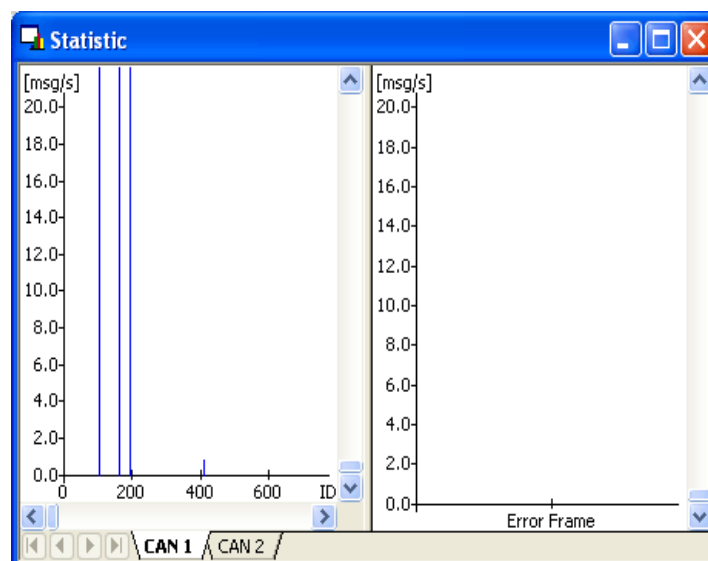


Figure 42: Statistics Window

Display of transmit interval or message rate

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

View modes

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.

Scale Statistics window

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.



Info: 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.

Statistic of bus actions

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 and maximum spacing.

Source	Message	N	Aver	StdDev	NIN	MAX
System	Statistics report AR0002, 03:22:49 pm					
System	Statistics for transmit spacing of messages in [ms]					
System						
System						
System	EngineData	TX	158	49.993	0.081572	49.00 50.06 CAN 2
System	EngineDataIEEE	TX	158	49.992	0.083078	49.00 50.06 CAN 2
System	ABSdata	TX	158	49.991	0.084216	49.00 50.06 CAN 2
System	Gateway_2	TX	145	54.503	252.31	2.13 3048.49CAN 1
System	Console_1	TX	394	19.997	0.055679	19.00 20.33 CAN 1
System	Console_2	TX	27	299.96	0.19612	299.00 300.00 CAN 1
System	DOOR_l	TX	263	29.987	1.6262	25.60 33.36 CAN 1
System	DOOR_r	TX	263	29.987	1.805	25.19 34.81 CAN 1
System	GearBoxInfo	TX	158	49.991	0.083996	49.00 50.06 CAN 2
System	NM_Console	TX	8	1116.2	233.38	586.94 1205.23CAN 1
System	NM_DOORleft	TX	7	1151.4	130.06	885.90 1205.23CAN 1

Figure 43: Statistical evaluation of a measurement in the statistics report

6.11 Statistics Monitor window

Display of statistical data

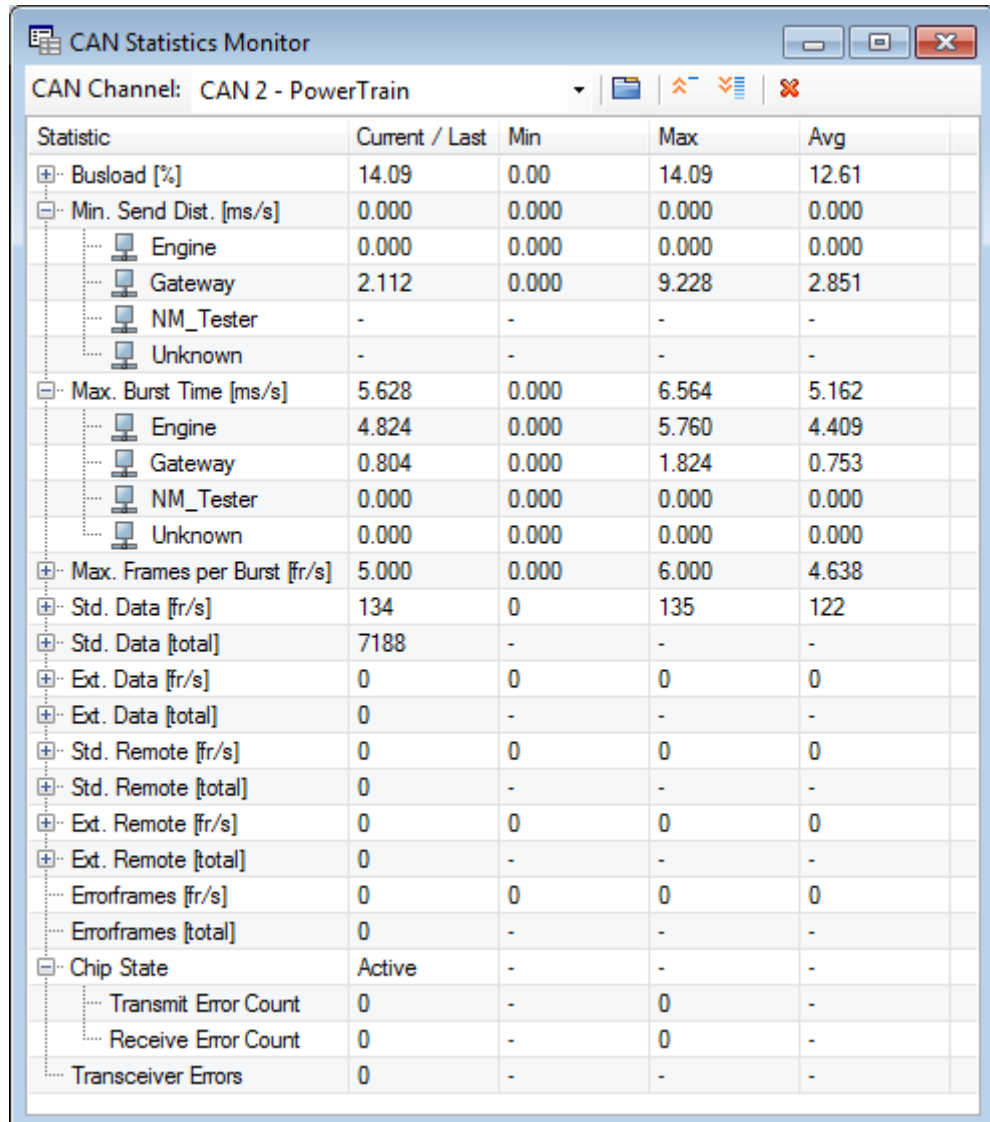
The Statistics Monitor window displays statistics about bus activities during measurement.

This window can be inserted into the measurement setup via the shortcut menus of the relevant function blocks. All bus events that arrive at the input of the measurement setup block are evaluated for this purpose. If a database is assigned to the selected channel, statistics can also be evaluated individually for each available node.

Columns

The **Current/Last** column shows the current or most recently derived statistical value. The **Min**, **Max** and **Avg** columns show the corresponding minimum, maximum and average statistical values.

Screenshot



Statistic	Current / Last	Min	Max	Avg
Busload [%]	14.09	0.00	14.09	12.61
Min. Send Dist. [ms/s]	0.000	0.000	0.000	0.000
Engine	0.000	0.000	0.000	0.000
Gateway	2.112	0.000	9.228	2.851
NM_Tester	-	-	-	-
Unknown	-	-	-	-
Max. Burst Time [ms/s]	5.628	0.000	6.564	5.162
Engine	4.824	0.000	5.760	4.409
Gateway	0.804	0.000	1.824	0.753
NM_Tester	0.000	0.000	0.000	0.000
Unknown	0.000	0.000	0.000	0.000
Max. Frames per Burst [fr/s]	5.000	0.000	6.000	4.638
Std. Data [fr/s]	134	0	135	122
Std. Data [total]	7188	-	-	-
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	-	-	-
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	-	-	-
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	-	-	-
Errorframes [fr/s]	0	0	0	0
Errorframes [total]	0	-	-	-
Chip State	Active	-	-	-
Transmit Error Count	0	-	0	-
Receive Error Count	0	-	0	-
Transceiver Errors	0	-	-	-

Figure 44: CAN Statistics Monitor window

Log bus statistics information

Bus statistics information is also recorded in logging (see chapter 5.8). 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.

6.12 Diagnostics Console

Working with diagnostics requests

The Diagnostics console makes it possible to send diagnostics requests directly to an ECU and receives and analyzes the corresponding response messages. To send diagnostics requests to an ECU the respective database in CANdela file format (*.cdd) must exist. After loading the diagnostics descriptions the Diagnostics console is opened in **CANoe** automatically, if the CDD has been assigned to a bus or a bus node.

Division of Diagnostics console

The Diagnostics console is divided up in three parts.

- In left area the diagnostics services are displayed. With the button **[Execute]** a chosen response will be sent. The response values can be displayed in symbolic or hexadecimal form.
- In upper area transmit services can be parameterized (e.g. input of a serial number).
- In the part below the results of the responses are displayed.

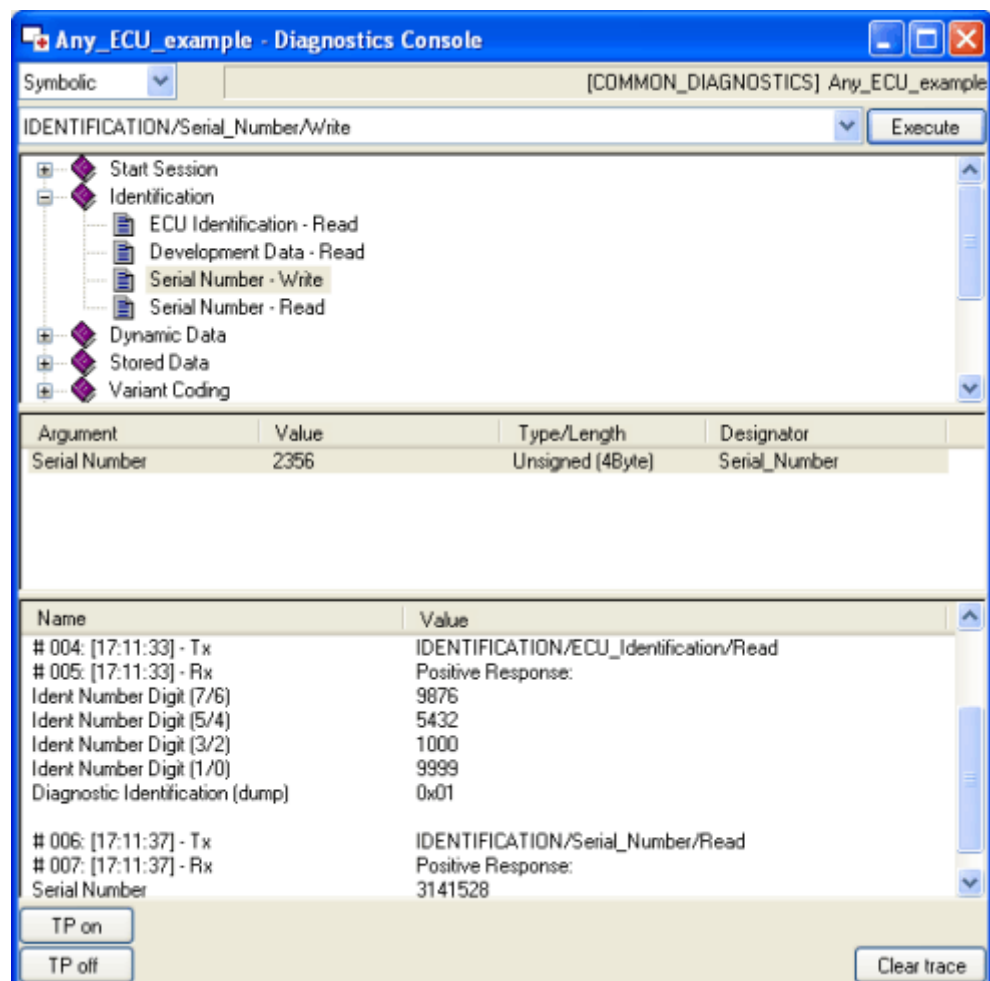


Figure 45: Diagnostics console

6.13 Fault Memory window

Working with fault memories

With the Fault Memory window you can read the fault memory list of an ECU directly. You can also delete single entries in the fault memory list (**DTC**: Diagnostic Trouble Code).

Read out fault memory list

To read out the fault memory list of an ECU, the respective database in CANdela file format (*.cdd) must exist.

After loading the diagnostics descriptions the Fault Memory window is opened in **CANoe** automatically, if the CDD has been assigned to a bus or a bus node.

After opening the Fault Memory Window the fault memory list of the ECU is read out via the **[Update]** button.

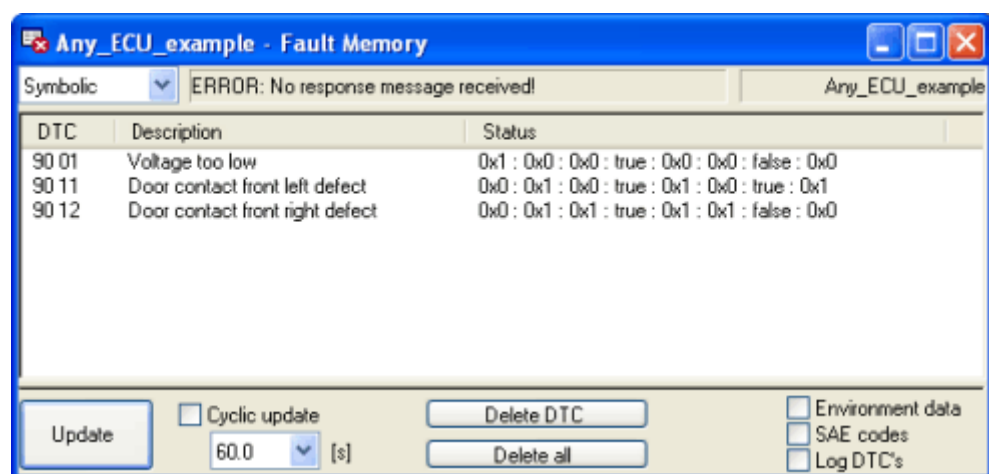


Figure 46: Fault Memory window

6.14 Test Setup window

Working with test environments

As soon as at least one test environment is opened in **CANoe**, the Test Setup window can be displayed. Here the individual test environments are displayed in a tree structure. Each root directory represents an independent test environment file.

As soon as the Test Setup window is present, it inserts itself into the normal window management and can be opened or brought into the foreground at any time using the **View** menu, the **View** icon or the **Window** menu.

With the Test Setup window, all actions such as loading and saving or recreating test environments can be executed. For this, click the right mouse button on a free area of the window and select the appropriate action from the shortcut menu.



Figure 47: Test Setup window

7 Blocks and Filter

In this chapter you find the following information:

7.1	Overview	page 100
7.2	Interactive Generator Block (IG)	page 101
7.3	Replay block	page 102
7.4	Trigger block	page 102
7.5	Filter and environment variable filter	page 103
7.6	Channel filter	page 103
7.7	CAPL nodes in the simulation setup	page 104
7.8	CAPL nodes in the measurement setup	page 104

7.1 Overview

Hot spots

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

Function blocks in measurement setup

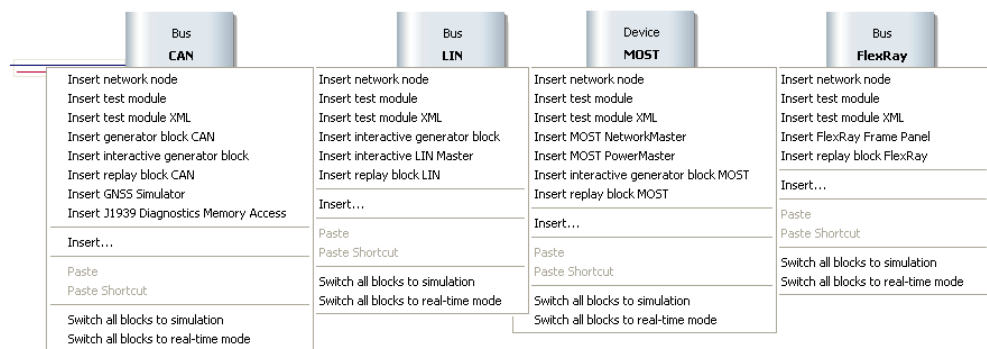
Function blocks can be recognized by their appearance or by their labels in the data flow chart.

Configure and delete function blocks

All blocks have shortcut menus which you can use to configure or delete them. By deleting the 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.

Function blocks in simulation setup

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 shortcut menu appears with the following menu commands for inserting function blocks:



Insert function blocks in the dataflow

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

Funktion block	Typ	Symbol	Reasonable place of use
Generator block	Data source	G	Simulation setup
Interaktiver Generator block	Data source	IG	Simulation 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 / Data sink	P	Simulation setup/ Measurement setup
EV Pass filter EV Stop filter	Data sink	PE SE	Measurement setup
Network node block	Data source	NK	Simulation setup

7.2 Interactive Generator Block (IG)

Application

The purpose of the Interactive Generator block is to generate and transmit messages. It appears in the data flow plan of the measurement setup as a small block 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.

Configuration during the measurement

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

Transmit list and signal list

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.

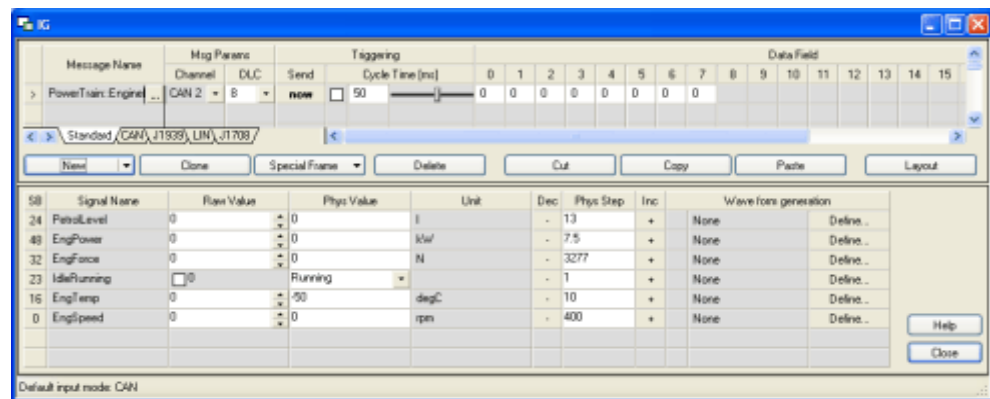


Figure 48: Configuration of the Interactive Generator block

Generate signals

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

Trigger conditions

The trigger condition is, in contrast to the traditional Generator block, entered separately for each message. You can choose between manual interactive triggering, key press and interval-driven repetition. Additionally, you can configure the number of messages to be sent at the time of triggering.

The IG 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.

7.3 Replay block

Replay measurement sequences The replay block makes it possible to replay measurement sequences which have already been recorded. The most important application is replaying a recorded data stream onto the CAN bus. Additionally they can play back environment variables, e.g. to generate test sequences. Replay blocks appear in the data flow plan as small blocks with the label **R**.

Configure replay You 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.

7.4 Trigger block

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 (see chapter 5.8).

7.5 Filter and environment variable filter

Reduce data volume The volume of data can be selectively reduced by using the filter block. Toggling between **pass filters** and **stop filters** will pass or block those identifiers and/or identifier ranges that are specified. 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.

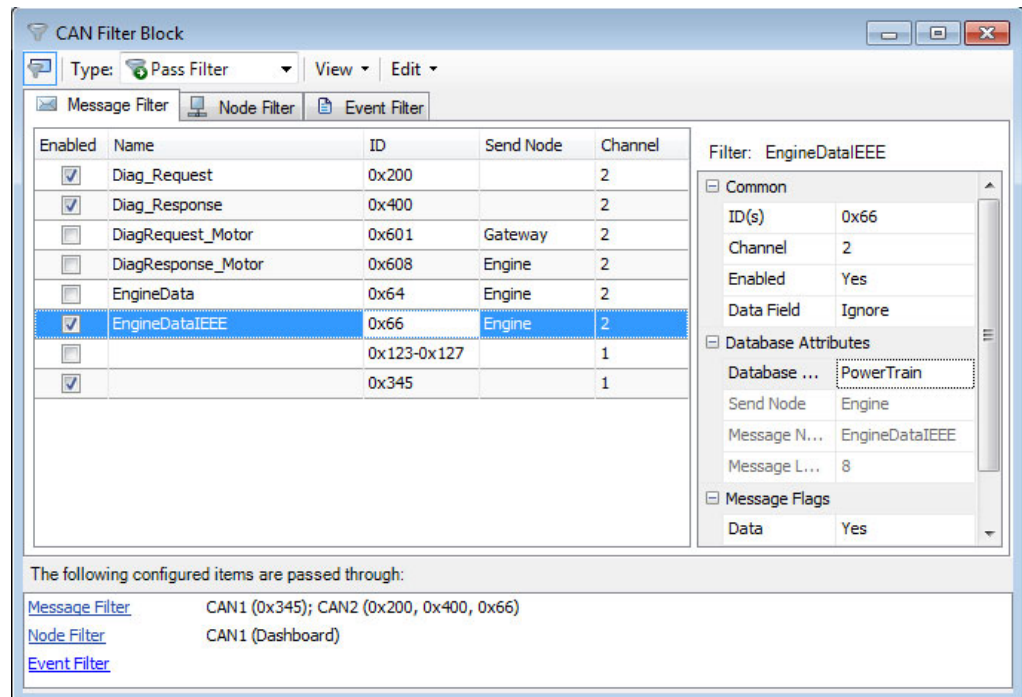


Figure 49: Filter configuration dialog for CAN filter block

Filter environment variables With **pass filters (PE)** and **stop filters (SE)** for environment variables you can selectively reduce the volume of data in the measurement setup. Only the selected environment variables will be passed through or blocked, respectively.



Info: In keeping with its function, a pass filter which is not configured (empty) does not pass any messages and so blocks all message traffic.

7.6 Channel filter

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



Figure 50: Channel filter in the data flow plan



Info: 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.

7.7 CAPL nodes in the simulation setup

Network node

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.

Start delay

The **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.

Drift/Jitter

To simulate the timer inaccuracy of real network nodes you can influence the timer of the simulated network nodes by **Drift** and **Jitter**. When this is the case, you can toggle between fixed deviation and an equally distributed fluctuation.



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

7.8 CAPL nodes in the measurement setup

Applications

Important applications of program blocks in the measurement setup include, e.g. activation of triggers or data reduction or monitoring in the measurement setup. Program blocks appear in the data flow plan as small blocks with the label **P**.



Info: 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 */
}
```



Info: 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).



Info: The CAPL source file is not deleted when a CAPL node is removed from the measurement setup.

8 Panel Designer

In this chapter you find the following information:

8.1	Overview
-----	----------

page 106

8.1 Overview


Create graphic panels

The Panel Designer 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.



Info: In this chapter signals, environment variables and system variables are called symbols.

Assign databases

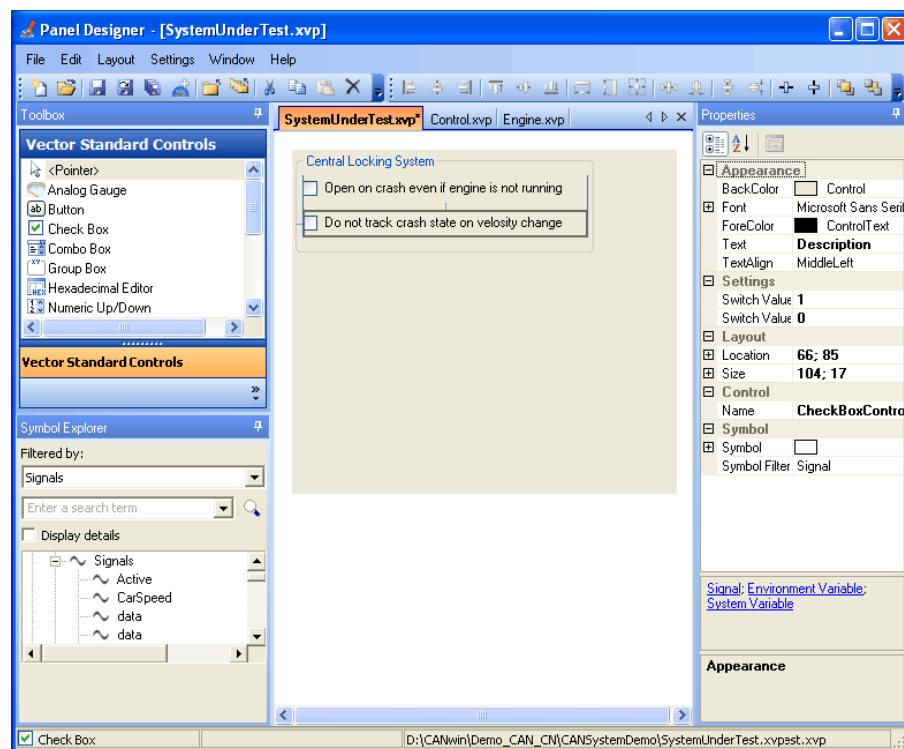
Start the Panel Designer with the  button on the **CANoe** toolbar, with the shortcut menu item **Edit** of an opened panel or – after selecting one or more panels – with the **[Edit]** button of the panel configuration dialog. This will ensure that the databases of your **CANoe** configuration will be associated automatically.

User interface

As a default setting you can find the Toolbox and the Symbol Explorer on the left hand side of the main window. From there you can place controls and symbols via drag and drop on an open panel. On the right hand side the properties of the selected objects are displayed in a table format. In the middle you can see the working area where you can create your panels.

This division of the main window makes the process to configure the panel and its controls virtually dialog-free.

Screenshot



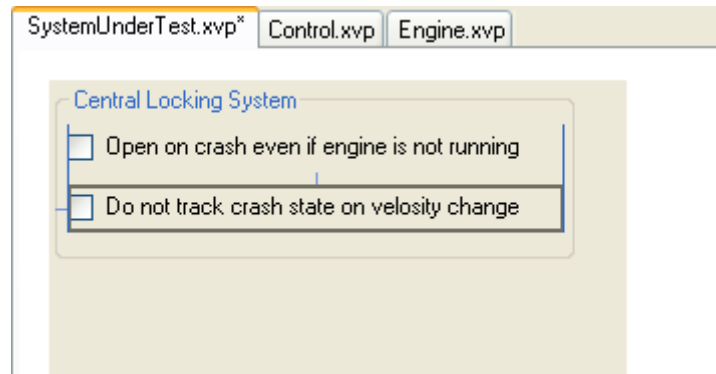
Working area

You can create and edit Panels in the working area.

You can open multiple panels at the same time in the Panel Designer. The open panels are displayed on separate tabs in the working area.

You can create a new panel via **Menu|File|New Panel**. After that you can assign controls and symbols to the panel.

To facilitate alignment of the controls, guide lines on which controls can be aligned appear when you start the assignment process. In addition to the guide lines, the **Edit** and **Layout** menus feature additional edit options, e.g. copying controls.



Toolbox

All available controls are displayed in the toolbox.

You can assign the controls from the toolbox to the panel in various ways:

- Using drag & drop
- By double-clicking
- By clicking with the left mouse button to select the required control and then clicking again with the left mouse button on the panel

Symbol Explorer

In the Symbol Explorer, you select the symbol you wish to assign to a control. All in the database available symbols are displayed.

The Symbol Explorer supports several options for accessing the same symbol. For example, you can select a symbol directly from the symbol or message list or find the corresponding symbol via a specific node.

Select the required symbol in the tree view and drag & drop to assign it to a control or the panel.

Properties window

The panel and controls are configured using the Properties window.

The window displays all settings of a selected control or panel in table format. A brief description of the active setting appears at the bottom of the Properties window.

You can select a number of elements at once to modify their common settings simultaneously in the Properties window.



Cross reference: You can find detailed information of the Panel Designer and the Panel Editor in the online help.

9 CAPL

In this chapter you find the following information:

9.1	CAPL basics	page 110
9.2	CAPL Browser	page 112

9.1 CAPL basics

Programming individual applications

The universal applicability of **CANoe** results in large measure from its user programmability. The **Communication Access Programming Language CAPL** is a C-like programming language, which allows you to program **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.

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.

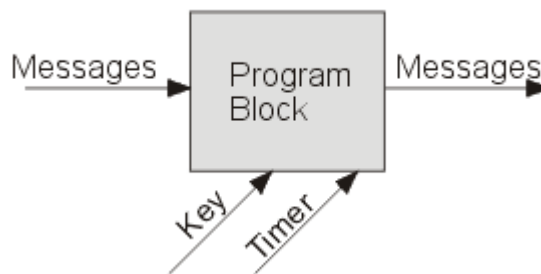
CAPL nodes are inserted in the data flow plan as function blocks. Event procedures serve as inputs in CAPL. These procedures can react to external events (e.g. the occurrence of specific messages). You send messages by calling the function `output()`. These language tools and symbolic access to the various 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.



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

Procedure

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.



Call analysis and test functions

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.

Emulate the system environment

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.

Program a gateway

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.

Trigger a Logging block

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()`.

Insert CAPL nodes

A CAPL program can be inserted in the measurement setup at all hot spots and also directly at the bus symbol in **CANoe's** simulation setup. To do this, select the menu item **Insert CAPL node** from the hot spot's shortcut 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.

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 cannot send them on the bus. Since the data flow is directed from left to right, these messages are only passed to the function blocks to the right of the CAPL program. Only messages generated by CAPL programs located in **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.

Compile CAPL programs

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**.

Symbolic names in CAPL programs

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.

The use of the symbolic database makes your program 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.

Advantage

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.

Event procedures

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**).

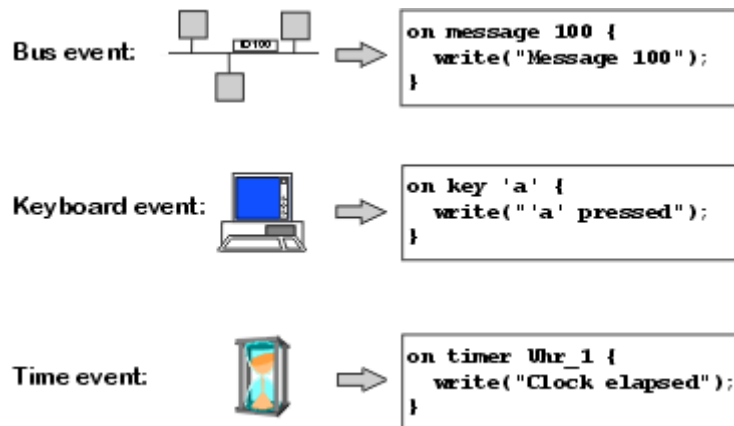


Figure 51: Examples of CAPL event procedures

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.

Program parts

A CAPL program consists of two parts:

- ➔ Declare and define global variables
- ➔ Declare and define user-defined functions and event procedures

9.2 CAPL Browser

Overview

CAPL program files are ASCII files as well as C or PASCAL program files. So you can edit them with each ASCII text editor of your choice. 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.

CAPL compiler

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.

Open CAPL Browser

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.

Browser architecture

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

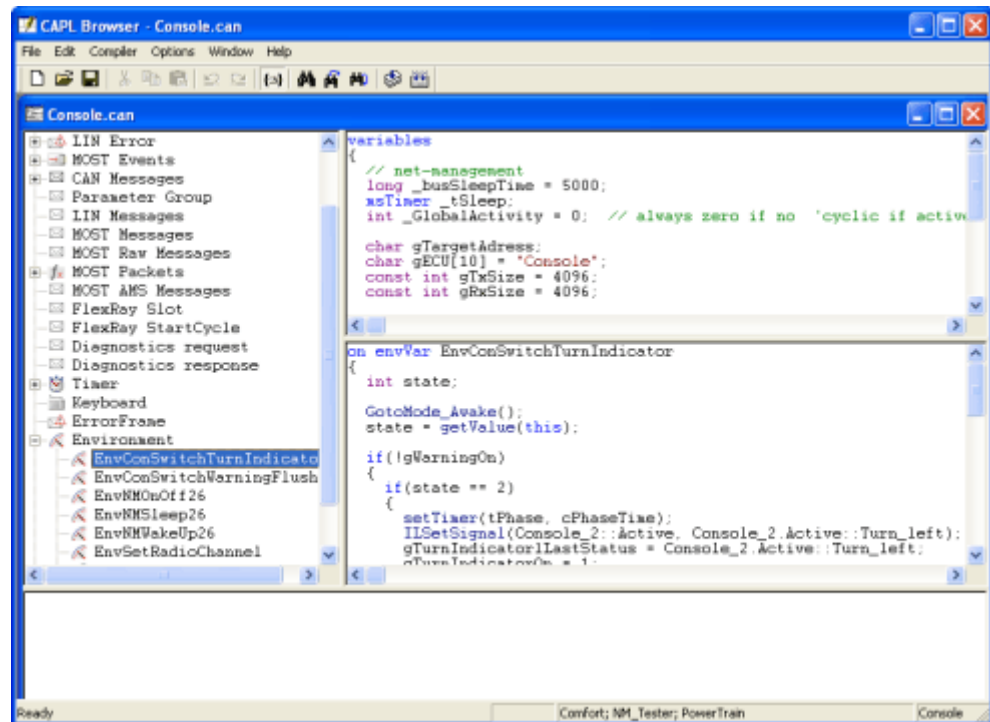


Figure 52: CAPL Browser

Browser tree

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.

Panes

You can access the most important functions for each of the panes from the shortcut menu by pressing the right mouse button. In the editor panes you have access, via the shortcut menu, to the intrinsic CAPL functions and to the objects defined in the database. Furthermore, you can copy text to the Clipboard from the shortcut menu, and from there you can paste it in your program.

Editor

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 shortcut menu.

Message window

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.

10 CAN

In this chapter you find the following information:

10.1 Overview

page 116

10.1 Overview

CAN specific features

The **CANoe** Tour and various examples in this manual are based on the CAN bus system.



Further Information: Please refer to the online help for further details.

11 LIN

In this chapter you find the following information:

11.1	Preliminary note	page 118
11.2	How to create a LIN description file	page 118
11.3	How to create a CANoe.LIN configuration	page 118
11.4	How to simulate and analyze a LIN network	page 119
11.5	How to control a LIN Master's scheduler Using the Interactive Master Using CAPL	page 119
11.6	How to log and replay LIN traffic	page 120
11.7	How to view LIN signals	page 121
11.8	How to manipulate LIN signals Using CAPL Signal API Using the CAPL function output() Using the Interactive Generator Block Using panels	page 121

11.1 Preliminary note

Introduction

LIN (**L**ocal **I**nterconnect **N**etwork) is a deterministic communication system for connecting ECUs with smart sensors, actuators and controls. Vector's software tool **CANoe.LIN** provides you with specific features for developing, analyzing and testing LIN networks according to the specifications LIN 1.x, LIN 2.0, LIN 2.1, SAE-J2602 (US-LIN) and Cooling-Bus.

This getting-started guide will help you create your first **CANoe.LIN** configuration. Through a series of how-tos you will also be guided through the most commonly needed LIN features.



Further Information: Please see the online help for further information about **CANoe.LIN** features and topics. Configuration examples for LIN demonstrating most of **CANoe.LIN**'s features can be found in the demo sub-directory: `DEMO_LIN_CN`.


11.2 How to create a LIN description file

LIN database

Although not required, it is highly recommended that you use a LIN database when developing, analyzing or testing LIN networks. The LIN database is described as a LIN description file (**LDF**) using the LIN configuration language as defined in the LIN specification.

LDF Explorer

If you do not already have an LDF, you can easily create one using the **LDF Explorer** utility provided with **CANoe.LIN**. You can start the **LDF Explorer** from the **Windows Start** menu (**Start|Vector CANwin|Tools|LDF Explorer – LIN**).

With the **LDF Explorer** you can also view and analyze your LDFs by double-clicking an LDF either in the **Windows Explorer** or in **CANoe.LIN** via **File|Open LDF Explorer** or the toolbar icon .



Further Information: Example LDFs can be found in the LIN demo subdirectories e.g. `... \Demo_LIN_CN \LINSysDemo \LINdb`.

11.3 How to create a CANoe.LIN configuration




To create a LIN configuration using a LDF, simply follow these steps:

1. Create a new configuration using **File|New Configuration**.
2. Select the LIN template and activate the checkbox **Use Wizard**.
3. Now press the button **[Database]**, open your LDF e.g. `... \Demo_LIN_CN \LINSysDemo \LINdb \door.ldf` and press **[Continue]**.
4. Select those nodes to be simulated from the **Available Nodes** list and assign them to the **Assigned Nodes** list and press **[Continue]**.
5. Select one of the LIN channels from the **Available Channels** list and assign it to the **Assigned Channels** list and press **[Continue]**.
6. To complete your configuration press **[Finish]**.

The configuration wizard has now automatically added the Master and Slave nodes to your LIN network and configured the communication according to your LDF.

11.4 How to simulate and analyze a LIN network

Start the measurement

Using the configuration created in chapter 11.3, you can now press the start button  in the main toolbar to observe the LIN traffic in the Trace window (main menu **View|Trace**) either using a simulated or real bus hardware (see main toolbar combo box). The Trace window not only displays valid LIN frames, but all types of LIN bus events and errors.



Further Information: For a full list of LIN events and errors, please see the online CAPL help for LIN.

Display in the Trace window

By expanding a LIN frame, you can view its signals as defined in the LDF associated to this LIN channel. Additional LIN-specific columns can be added to the Trace window using its configuration dialog.

LIN Statistics

To view the LIN statistics, open the **LIN Statistics Monitor** via double-click on the **LIN Statistics Monitor** in the measurement setup or via the menu **View|LIN Statistics Monitor**.



Info: For remaining bus simulations, simply connect real LIN nodes to your LIN interface and deactivate the simulated version of these nodes in the Simulation Setup (**View|Simulation Setup**) e.g. using the space bar or context menu shortcut.

11.5 How to control a LIN Master's scheduler

11.5.1 Using the Interactive Master



1. Create a LIN configuration, by following the steps described in chapter 11.3.
2. In the simulation setup (**View|Simulation Setup**) insert an Interactive Master using the context menu shortcut.
3. Open the Interactive Master block e.g. per double-click.
4. Configure per context menu, which schedule the Interactive Master should start on measurement start.
Per default the first schedule defined in the LDF will be started.
5. Start the measurement and change the schedules as desired interactively.

11.5.2 Using CAPL



1. Create a LIN configuration, by following the steps described in chapter 11.3.
2. Select the Master network node in the simulation setup (**View|Simulation Setup**) and choose **Configuration...** from the context menu.
3. Enter a file name for your CAPL program and then press **[Edit]** to start the CAPL Browser.
4. Now use the CAPL function `LINChangeSchedTable` to determine when a schedule table change should take place.

For example to change to the second schedule table in the LDF on pressing key <2> insert the following:

```
on key '2'
{
    LINChangeSchedTable(1); // Index starts with zero
}
```

5. Now compile the CAPL program e.g. by pressing <F9> and start the measurement.
6. On pressing the key <2> the Master's scheduler should change to the second schedule defined in your LDF (e.g. **Table1** in LDF example `door.ldf`).



Info: If the Interactive Master is active, it will automatically filter any schedule table commands called by CAPL.

11.6 How to log and replay LIN traffic

Log

You can log LIN traffic by activating the connection to the Logging block in the measurement setup (**View|Measurement Setup**). Double-click this block to open its configuration dialog.

Analyze

To analyze a logging file offline, use the offline mode (main menu **Mode|Offline**) and configure the block in measurement setup (using shortcut menu option **Configuration...**) to use your logging file.

Replay

Alternatively you replay a log file by inserting and configuring a LIN Replay block in the simulation setup (**View|Simulation Setup**).



Further Information: Please see the ElectricMirrorBus of the LIN system demo for an example of how to configure the LIN Replay block.

11.7 How to view LIN signals

Data window

To view frame signals numerically, either open an existing Data window (**View|Data**) or create a new one in the measurement setup (**View|Measurement Setup**). You can add signals to this window in one of the following ways:

- Via context menu shortcut **Add signals...**
- Per drag & drop from the Symbol Explorer (**View|Symbol Explorer**)
- Per drag & drop from the Trace Window (**View|Trace**).

Graphics window

To view frame signals graphically, either open an existing Graphic window (**View|Graphics**) or create a new one in the measurement setup (**View|Measurement Setup**). You can add signals in the same way as for the Data window.

Panels

Signals can also be viewed by creating your own panels using the Panel Designer utility (**File|Open Panel Designer**). For more information on how to create panels see chapter 11.8.4.

11.8 How to manipulate LIN signals

11.8.1 Using CAPL Signal API



1. Create a LIN configuration, by following the steps described in chapter 11.3.
2. Select a network node (e.g. **DWF_Left**) in the simulation setup (**View|Simulation Setup**) and choose **Configuration...** from the shortcut menu.
3. Enter a file name for your CAPL program and then press **[Edit]** to start the **CAPL Browser**.
4. Now use the appropriate signal object to manipulate a LIN signal that is defined in the assigned LDF database.

Here is a simple example assuming the LDF database (e.g. **door.ldf**) defines a signal **DWFL_WinPos**:

```
on key '+'
{
    int val;
    val=$DWFL_WinPos;    // reading signal
    $DWFL_WinPos=val+1;  // writing signal
}
```

5. Now save and compile the CAPL program e.g. by pressing <F9> and start the measurement.
6. Open the Trace window (**View|Trace**) and expand the corresponding LIN frame (e.g. **DWFL_WinPos**).
On pressing the key <+>, the signal should now increase its value by one.

11.8.2 Using the CAPL function output ()

Update multiple frame signals

You can update signals using the CAPL function `output`. Using this method, you can update multiple frame signals simultaneously. To try out this method, add a new event handler for the key `<->` to your CAPL program created in 11.5.2 with the following code:

```
on key '-'
{
    linmessage DWFL_WinPos myframe;
    int val;
    val=myframe.FWL_WinPos;
    myframe.FWL_WinPos=val-1;
    output(myframe);
}
```

Now save and recompile your CAPL program and start the measurement again. With keys `<+>` and `<->` you should be able to increment and decrement the signal's value.


11.8.3 Using the Interactive Generator Block





1. Create a LIN configuration, by following the steps described in chapter 11.3.
2. In the simulation setup (**View|Simulation Setup**) insert an Interactive Generator using the shortcut menu.
3. Open the Interactive Generator e.g. per double click.
4. To update the signals of a LIN frame, simple press **[New]** and insert a LIN frame (e.g. **DWFL_WinPos**).
5. During the measurement you can now interactive manipulate this frame's signal values (e.g. **FWL_WinPos**) using edit boxes in the bottom half of the dialog.
6. Alternatively, you can connect the signal to a Waveform generator using the button **[Define]** in the column **Waveform generation**.

11.8.4 Using panels



You can also create your own panels using the **Panel Designer** utility (**File|Open Panel Designer**), which can also be started from the main toolbar . Here is an example.

1. Drag and Drop the Trackbar  from the Toolbox to an opened panel.
2. Select a LIN signal to be manipulated via **Symbol** in the Properties window.
3. Now save your panel (**File|Save Panel**).
4. With the  button of the toolbar you can add the created panel to your **CANoe** configuration.

After that the panel will be displayed on the currently active desktop.

After measurement start it is now possible to change the signal's value using the Trackbar.

12 MOST

In this chapter you find the following information:

12.1	Preliminary note	page 124
12.2	MOST database: Function catalog	page 124
12.3	How to create a CANoe.MOST configuration	page 124
12.4	How to analyze a MOST network	page 125
12.5	How to stimulate a MOST system	page 126
12.6	How to log and replay MOST data traffic	page 126
12.7	Using CAPL	page 127
	Program-controlled sending	
	Program-controlled receiving	

12.1 Preliminary note

Introduction

MOST® (Media Oriented Systems Transport) is used to transmit audio, video, and control data via fiber optic cables. **CANoe.MOST** supports the speedgrades MOST25, MOST50 and MOST150.

This getting-started guide will help you create your first **CANoe.MOST** configuration. Through a series of how-tos you will also be guided through the most commonly needed MOST features.



Further Information: Please see the online help for further information about **CANoe.MOST** features and topics.

Configuration examples for MOST demonstrating most of **CANoe.MOST**'s features can be found in the demo sub-directory: `DEMO_MOST_CN`

12.2 MOST database: Function catalog

Function catalogs

We highly recommend, that you use a MOST function catalog for development, analysis, and testing of MOST networks. The MOST function catalog that can be used for **CANoe** is in the form of an XML file whose format is specified by the MOST Cooperation.

This function catalog is usually created by the OEM and provided to all parties involved in the project.

If you do not yet have a function catalog in XML format, contact your OEM. For members of the MOST Cooperation, a function catalog editor is available for download on the MOST Cooperation intranet. It can be used to create new function catalogs. In addition, example catalogs of standard function blocks are available on the intranet.



Further Information: Example catalogs can be found in the MOST subdirectories for demos, e.g.

`..\Program Files\CANwin\Demo_MOST_CN\MOSTSystemDemo\Database.`

12.3 How to create a CANoe.MOST configuration



To create a MOST configuration using a LDF, simply follow these steps:

1. Create a new configuration using **File|New Configuration...**
2. Select the MOST, MOST50 or MOST150 template – dependent on the speedgrades of your MOST system.
3. Select the required number of MOST channels (menu **Configuration|Options|Channel Usage**).



Info: In the case of MOST150 and MOST50, for technical reasons Vector strongly recommends placing the node and the spy of a connected **Optolyzer** on separate channels. This is already preconfigured accordingly in the MOST150 and MOST50 templates.

4. Select the hardware to be used (menu **Configuration|Network Hardware...**)



Info about MOST150 and MOST50: Enter the IP address of the **Optolyzer** on the **Interface** page. (If it is already connected, you can determine its IP address at the push of a button). In addition, select the network adapter to which the **Optolyzer** is connected.

5. Configure the hardware settings, such as hardware mode, node addresses, or additional services for each channel.
6. Add a function catalog as a database in the simulation setup.
7. Save your configuration in a directory of your choice. As a result, you can reload all your settings at any time.

12.4 How to analyze a MOST network

Trace window

Using the configuration created in chapter 12.3, you can now push the start button in the main toolbar to monitor the MOST data traffic in the Trace window (**View|Trace** menu).

The Trace window displays not only the valid MOST messages but all types of MOST bus events and errors.



Further Information: A complete list of all MOST events and errors can be found in the CAPL online help for MOST.

If the Trace window is paused, you can expand a MOST message to see how the parameters of the message are defined in the function catalog. Using the configuration dialog, you can add additional MOST-specific columns to the Trace window.

MOST analysis windows

For MOST there is a set of analysis windows that provide a quick overview of the structure and status of the connected MOST system, without significant configuration effort.

MOST Status	Displays the current status of the MOST Ring and the main settings of the connected hardware interface.
MOST System Viewer	Displays the monitored structure of the MOST ring and allows a ring scan when the bypass is open.
MOST Audio Routing	Displays the current assignment of the synchronous channels. Through selection of a connection label followed by demute , normal audio channels can be listened to with headphones at the MOST interface.
MOST Registry	Displays the content of the registry based on the monitored ring scans.
MOST FBlock Monitor	Displays all properties of the applications transmitted via the bus.



Info: If these windows do not show any communication or information regarding the connected MOST system, use the status window to check the following:

- Whether the ring is operating (💡 light on).
- Whether the ring is in a stable condition (🔒 lock).

If little is known about the system structure, you can also initiate a ring scan via the context menu of the Status window. While this has an active effect on your system, the subsequent communication usually ensures that the analysis windows will be filled.



Info: If some functions of the windows, such as **Scan** or **Get-Property** are deactivated or do not produce any results, check whether the node mode of the MOST interface is active in the status window, i.e., the bypass is open (🔓).



Further Information: Chapter 6 describes a set of analysis windows that function analogously for MOST if, for example, MOST signals are inserted into the Data window.

12.5 How to stimulate a MOST system

Interactive Generator block MOST

The Interactive Generator block MOST (IG MOST) offers the fastest way to send MOST messages or packets. The configuration user interface for entry of messages can be opened via the **View|IG MOST** menu.



Info: If the **View** menu does not offer IG MOST, this means that the configuration does not contain such a generator block. To insert a new block, the measurement must be stopped and the IG MOST must be inserted into the simulation setup as a block.

Stress window

The Stress window offers various options for stimulating the MOST system.

The configuration user interface for selecting and setting the stress mode can be opened via the **View|MOST Stress** menu.

MOST messages or packets can then be sent cyclically in order to stress the MOST system. In addition, unlock cycles or ring interruption phases can be created.

12.6 How to log and replay MOST data traffic

Logging

You can log MOST data traffic by activating the connection to the Logging block in the measurement setup (**View|Measurement Setup** menu). With a double-click on the Logging block you can open the configuration dialog.

Offline analysis

To analyze a log file offline, use offline mode (**Mode|Offline** menu) and configure the block in the measurement setup (**Configuration...** command in shortcut menu).

Replay

Alternatively, you can replay a logging file to a real MOST ring by inserting and configuring a MOST replay block in the simulation setup (**View|Simulation Setup**).

12.7 Using CAPL

CAPL programs

Alternatively, CAPL nodes can be used to send messages or packets.

As preparation a network node has to be inserted in the simulation setup and a CAPL file has to be assigned to this node.

12.7.1 Program-controlled sending



Example: Sending a message on keyboard hit:

```
on key 'm'
{
    // channel, destination address, message specification,
    // instance id
    mostAmsOutput(1, 0x100, "NetBlock.DeviceInfo.Get(0x00)",
    0x00);
}
```



Info: It would also be possible to achieve this exact behavior through suitable configuration with the IG MOST.

The MOST input assistance helps you when entering strings for message definition. To start this wizard, press <CTRL>+<M> in the CAPL Browser.



Example: Sending a MOST Package (MDP) on asynchronous channel:

```
on key 'p'
{
    BYTE pktdata[1014] = { 1, 2, 3, 4, 5, 6, 7 }
    // channel, destination address, length, packet data
    OutputMostPkt(1, 0x101, 7, pktdata);
}
```



Example: Sending a MOST Ethernet Package (MEP) – only available with MOST150:

```
on key 'e'
{
    BYTE pktdata[1506] = { 1, 2, 3, 4, 5, 6, 7 }
    // channel, destination MAC address, length, packet data
    outputEthPkt(1, 0x123456, 7, pktdata);
}
```

12.7.2 Program-controlled receiving



Example: Receiving a AMS message:

```
on mostAMSMMessage NetBlock.DeviceInfo.Get
{
    // report message variable
    mostAmsMessage NetBlock.DeviceInfo.Status msg;
    // initialize report message
    // (e.g. with source address of the sender of DeviceInfo.Get
    mostPrepareReport(this, msg);
    // user specific code to fill parameters of the message
    // ...
    // send message
    output(msg);
}
```



Example: Receiving a MOST Package (MDP):

```
OnMostPkt(long pktdatalen)
{
    BYTE buffer[1100];
    MostPktGetData(buffer, pktdatalen);
    // user specific code to analyze parameters of the packet
    // ...
}
```



Example: Receiving a MOST Ethernet Package (MEP) – only available with MOST150:

```
OnMostEthPkt(long pktDataLen)
{
    BYTE buffer[1506];
    MostEthPktGetData (buffer, pktdatalen);
    // user specific code to analyze parameters of the ethernet
    // packet
    // ...
}
```

13 FlexRay

In this chapter you find the following information:

13.1	Preliminary note	page 130
13.2	How to create a FlexRay database	page 130
13.3	How to create a CANoe.FlexRay configuration	page 131
13.4	How to simulate and analyze a FlexRay network	page 131
13.5	How to log and replay FlexRay traffic	page 132
13.6	How to view FlexRay signals	page 133
13.7	How to manipulate FlexRay signals	page 133
	Using	
	Using CAPL functions FRUpdateStatFrame/FRSendDynFrame/FRUpdatePDU	
	Using the FlexRay Frame Panel or FlexRay PDU Panel	
	Using panels	
	How to implement specific behavior for a remaining bus simulation	

13.1 Preliminary note

Introduction

FlexRay is a deterministic and optionally redundant communication system for inter-connecting ECUs. **CANoe.FlexRay/DENoe.FlexRay** supports the FlexRay standard in the version 2.1.

This getting-started guide will help you create your first **CANoe.FlexRay** configuration. Through a series of how-tos you will also be guided through the most commonly needed FlexRay features.



Further Information: Please see the online help for further information about **CANoe.FlexRay** features and topics.

Configuration examples for FlexRay demonstrating most of **CANoe.FlexRay**'s features can be found in the demo sub-directory: `DEMO_FlexRay_CN`

13.2 How to create a FlexRay database

FlexRay database

Although not required, it is highly recommended that you use a FlexRay database when developing, analyzing or testing FlexRay networks. The FlexRay database is described as a FIBEX database file using the ASAM AE MCD2 standard for FIBEX files in version 1.1.5a, 1.2.0, 1.2.0a, 2.0.0b, 2.0.0d, 2.0.1, 3.0.0 or 3.1.0.

FIBEX

If you do not already have a FIBEX file, you can easily create one using Vector's design tool **DaVinci Network Designer FlexRay**.

FIBEX database files are XML files that also can be edited – although this is not recommended – using a general XML or ASCII editor or with the **FIBEX Explorer** that is coming along with **CANoe.FlexRay**.

The **FIBEX Explorer** is also very useful in order to display all information stored in the database in a very convenient way.



Further Information: An example FIBEX file can be found in the FlexRay demo subdirectory `..\Demo_FlexRay_CN\FlexRaySystemDemo\FIBEX`.

CHI

If no FIBEX file is available, then optionally a CHI file can be imported that describes the TDMA parameters for the FlexRay communication controller. But remember that CHI files are hardware dependent. That means you need an appropriate file depending on your bus interface. For observing the network startup of a FlexRay cluster or when configuration files are not available, then the bus interfaces can be set into the so-called asynchronous mode. In this mode no frames can be sent.

13.3 How to create a CANoe.FlexRay configuration




To create a FlexRay configuration using a FIBEX file, simply follow these steps:

1. Create a new configuration using **File|New Configuration**.
2. Select the FlexRay template and activate the checkbox **Use Wizard**.
3. Now press the button **[Database]**, open your FIBEX file, e.g.
`Demo_FlexRay_CN\FlexRaySystemDemo\FIBEX\PowerTrain.xml`, select the appropriate cluster – if necessary – from within the file and press **[Continue]**.
4. Select those nodes to be simulated from the **Available Nodes** list and assign them to the **Assigned Nodes** list and press **[Continue]**.
5. Select one of the FlexRay channels from the **Available Channels** list and assign it to the **Assigned Channels** list and press **[Continue]**.
6. To complete your configuration press **[Finish]**.

The configuration wizard has now automatically added the FlexRay nodes to your FlexRay network and configured the communication according to your FIBEX file.

13.4 How to simulate and analyze a FlexRay network

Start the measurement

Using the configuration created in chapter 13.3, you can now press the start button  in the main toolbar to observe the FlexRay traffic in the Trace window (main menu **View|Trace**) when using simulated bus hardware (see main toolbar combo box).

Working with a real bus

If you want to use a real bus, then four prerequisites must be fulfilled:

Step 1

Your FlexRay bus can do a startup. This is assured when already two real startup nodes are present and are working. One or both of these startup nodes can be simulated by **CANoe** depending on your current bus interface hardware. Therefore **CANoe** must be configured in a way that it sends one or two startup frames.



Info: If **CANoe** does not send any startup frame, then you need two external startup nodes. When **CANoe** sends one startup frame, then you need one external startup node. If the bus interface is able to send two startup frames, then you need no external startup node.

Any registered frame (of the static segment) in the TX buffer list, CAPL program or Frame Panel can be defined to be a startup and/or sync frame. In the Key Slot Configuration (**Configuration|Network Hardware...|FlexRay|Key Slot Config**) you can define up to two Startup/Sync-Frames.

Step 2

The FlexRay TDMA parameters that are defined in your FIBEX or CHI file matches exactly those definitions that are used to implement the external nodes.



Info: The bus interface hardware is allowed to be used in a so-called “asynchronous mode”. This mode requires just setting the baud rate. It is restricted to receive frames only.

Step 3

Select the connected hardware interface type in a combo box of the Network Hardware Configuration dialog under **Configuration|Network Hardware...|FlexRay**.



Info: Under the tree items **Controller** or **Protocol** of the FlexRay hardware configuration dialog you can choose the database settings to be used for the TDMA parameters or a manual setting that enables the import of an appropriate CHI file. If you choose the database settings, then you can also select the database node from which the local settings will be used.

Step 4

Assure that the bus channel A is connected to the channel A of the bus hardware interface. The same applies for channel B. If both channels are exchanged, then any received frame will be interpreted with a wrong CRC and therefore it will be rejected or interpreted as an erroneous frame.



Info: When a bus interface channel is used it must be connected to the correct bus channel. But a network interface channel need not to be connected to a real bus channel. A cross-connection of the bus interface channel A to the bus interface channel B is not allowed!

If all prerequisites are fulfilled the bus interface of **CANoe** can synchronize to the real FlexRay bus when pressing measurement start.

Trace window

The Trace window not only displays valid FlexRay frames, but all types of FlexRay bus events and errors.

By expanding a FlexRay frame, you can view its signals as defined in the FIBEX file associated to this FlexRay channel. Additional FlexRay-specific columns can be added to the Trace Window using its configuration dialog.



Further Information: For a full list of FlexRay events and errors, please see the online CAPL help for FlexRay.

Bus Statistics window

To view the FlexRay bus statistics, open the Bus Statistics window via double-click on the Bus Statistics block in the measurement setup or via the menu **View|Bus Statistics**.



Info: For remaining bus simulations, simply connect real FlexRay nodes to your FlexRay interface and deactivate the simulated version of these nodes in the simulation setup e.g. per double-click or shortcut menu.

13.5 How to log and replay FlexRay traffic

Log

You can log FlexRay traffic by activating the connection to the Logging block in the measurement setup. Double-click this block to open its configuration dialog.

In the **CANoe** offline mode logging files can be analyzed in non real-time, e.g. by using single stepping.

Replay The FlexRay Replay block in the simulation setup or test setup replays the FlexRay events and values of logged environment variables.

13.6 How to view FlexRay signals

Data window To view frame signals numerically, either open an existing Data window in the measurement setup (**View|Data**) or create a new one. You can then add signals to this window in one of the following ways:

- Via shortcut menu item **Add signals...**
- Per drag & drop from the Symbol Explorer (**View|Symbol Explorer**)
- Per drag & drop from the Trace window.

Graphics window To view frame signals graphically, either open an existing Graphic window in the measurement setup (**View|Graphics**) or create a new one. You can then add signals in the same way as for the Data window.

Panels Signals can also be viewed by creating your own panels using the Panel Designer utility. See also chapter 13.7.4.

13.7 How to manipulate FlexRay signals

Send a FlexRay frame Manipulating a FlexRay signal means to send a FlexRay frame or a PDU (with valid update bit) that contains this signal. Before you can send a FlexRay frame/PDU you sometimes have to reserve a buffer inside the communication controller.



Info: You must assure, according to the FlexRay standard that a frame in a specific slot of the static segment is either sent only by **CANoe** or one other external node. A static frame of a specific slot must not be sent from **CANoe** and another external node!



Further Information: Whether your bus interface hardware requires reserving send buffers and how this can be achieved please refer to the online help for the `FRSetSendFrame/FRSetSendPDU` function in CAPL or the dialog of **Configuration|Network Hardware...|FlexRay|Send Buffer** (if not available, then it is not required for the selected hardware type). Defining a frame/PDU in the Frame Panel or PDU Panel automatically modifies the TX buffer of the FlexRay hardware.

13.7.1 Using signals



1. Create a FlexRay configuration, by following the steps described in chapter 13.3.
2. Select a network node (e.g. **GearBox**) in the simulation setup and choose **Configuration...** of the shortcut menu.
3. Enter a file name for your CAPL program and then press **[Edit]** to start the CAPL Browser.
4. Now use the appropriate signal to manipulate a FlexRay signal that is defined in the attached FIBEX database.

Here is a simple example using the event handler for the key <1> assuming the FIBEX database defines a frame **GearBoxInfo** that contains a signal **Gear**:

```

variables
{
    int gGear = 1; // 0 == reverse; 1 == neutral; 2 - 6 =
forward
}
on preStart
{
    // register FlexRay frame for sending:
    FRSetSendFrame( GearBoxInfo, 0 /*flags*/);
}
on key '1'
{
    gGear = $GearBoxInfo::Gear; // reading Signal
    if ( ((gGear > 0) && (gGear < 6)) ||
        ((gGear == 0) && ($ABSInfo::CarSpeed <= 5)) )
    {
        gGear++;
        //GearShiftText(gGear);
        $GearBoxInfo::Gear = gGear; // writing signal
    }
}

```

5. Now save and compile the CAPL program e.g. by pressing <F9> and start the measurement.
6. Open the Trace window via the main menu **View|Trace** and expand the FlexRay frame **GearBoxInfo**. On pressing the key <1>, your signal should now increase its value by one until it reaches 6.

13.7.2 Using CAPL functions

FRUpdateStatFrame/FRSendDynFrame/FRUpdatePDU

Update multiple frame signals

You can also update signals using the CAPL functions `FRUpdateStatFrame/FRSendDynFrame/FRUpdatePDU`. Using these methods, you can update multiple frame signals simultaneously. To try out this method, add a new event handler for the key <2> to your CAPL program with the following code:

```

variables
{
    int gGear = 1; // 0 == reverse; 1 == neutral; 2 - 6 = forward
    FRFrame GearBoxInfo myframe;
}

```



```

on preStart
{
    myframe.MsgChannel = %CHANNEL%;
    myframe.FR_ChannelMask = 1; // send only on A
    myframe.FR_Flags = 0x00; // flags
    FRSetPayloadLengthInByte(myframe, 16);
    // register FlexRay frame for sending:
    FRSetSendFrame(myframe);
}
on key '2'
{
    gGear = $GearBoxInfo::Gear;
    if ( ((gGear > 1) && (gGear <= 6)) ||
        ((gGear == 1) && ($ABSInfo::CarSpeed <= 5)) )
    {
        gGear--;
        myframe.Gear = gGear;
        FRUpdateStatFrame(myframe);
    }
}

```

Now save and recompile your CAPL program and start the measurement again. With keys <1> and <2> you can now increment and decrement the signal's value.



Info: If the frame that you want to send is a dynamic frame, then you have to use the function `FRSendDynFrame` instead of `FRUpdateStatFrame`! If you have to send PDUs, then use the function `FRUpdatePDU`.


13.7.3 Using the FlexRay Frame Panel or FlexRay PDU Panel





1. Insert the FlexRay Frame Panel or the FlexRay PDU Panel into the simulation setup and open it by double-click.
2. To update the signals of a FlexRay frame/PDU, simply press **[Add Row]** and insert a FlexRay frame/PDU (before measurement).
3. During the measurement you can now interactive manipulate this frame's payload area using edit boxes in the bottom half of the dialog.

13.7.4 Using panels



You can also create your own panels using the Panel Designer (**File|Open Panel Designer**), which can also be started from the main toolbar . Here is an example.

1. Drag and Drop the Trackbar  from the Toolbox to an opened panel.
2. Select a FlexRay signal to be manipulated via **Symbol** in the **Properties** window.
3. Now save your panel (**File|Save Panel**).

4. With the  button of the toolbar you can add the created panel to your **CANoe** configuration.
After that the panel will be displayed on the currently active desktop.

After measurement start it is now possible to change the signal's value using the Trackbar.

13.8 How to implement specific behavior for a remaining bus simulation

Requirements

A remaining bus simulation often requires

- communication with a Network Management protocol
- data exchange via a Transport Protocol (e.g. for diagnostics)
- automatic sending of frames cyclically with periods that possibly cannot be modeled by FlexRay's cycle multiplexing feature (e.g. frame/PDU period is 200 ms and FlexRay cycle is 5 ms)
- setting of the Update Bit of a PDU automatically with a specified period
- sending of signal values according to a global system state (e.g. if clamp 15 is off, then send invalid signal values, otherwise send current valid signal values from sensors)
- automatic update of message counters and CRC signals inside of frames or PDUs

Add-on packages

For these use cases various (sometimes OEM-specific) add-on packages for **CANoe.FlexRay** exist. Those add-on packages extend **CANoe.FlexRay** by a specific protocol API or an Interaction Layer for FlexRay that defines a specific sending behavior of the remaining bus simulation.

Tx buffer

Sometimes a complete remaining bus simulation is not possible, because the bus interface does not support enough TX buffers. In this case you can configure a partial remaining bus simulation (PRBS) in the context menu of the bus in the simulation setup. Using the PRBS configuration dialog it is possible to only send frames received by a ECU/system under test.

14 J1939 and NMEA 2000®

In this chapter you find the following information:

14.1	Introduction	page 138
14.2	Quick start	page 138
	Create a J1939 database	
	Create a J1939 configuration	
	Create communication relationships	
	Sample configurations	
14.3	Use cases	page 140
	Analyze J1939 networks	
	Diagnose J1939 networks	
	Simulate J1939 networks	
	Test J1939 networks	
	Log and trigger J1939 data	
	Trigger and filter J1939 data	
	Modify J1939 signals	
	Analyze GNSS data	
	Simulate a GNSS receiver	
	Play back GNSS protocol files	

14.1 Introduction

Introduction

The option J1939 contains specific extension for J1939 and NMEA 2000®.

J1939

SAE J1939 is a CAN-based communication protocol for data exchange between electronic control units (ECUs) in the commercial vehicle sector. It comes from the Society of Automotive Engineers (SAE) and works on the physical layer with CAN-high speed according to ISO11898-2.

Typical properties of J1939 are:

- based on 29-bit CAN identifier
- Point-to-point and broadcast communication
- no limitation of the data length to 8 bytes. With transport protocols, a transmission of up to 1785 bytes is possible.
- Network management for the management of node addresses and device names
- Definition of the parameter group (PG) as data bundle
- Definition of the parameter group number (PGN) as identification characteristic
- autonomous priority assignment regardless of the PGN although based on the CAN identifier.

NMEA 2000®

NMEA 2000® is a network specification based on the principles of the SAE J1939 definition. NMEA 2000® is used in the navigation area and defines navigation specific parameter groups. The transport protocol FastPacket is supported too.



Note: In this chapter only J1939 and NMEA 2000® specific extensions are described. The standard functionality is explained in the previous chapters of this manual.



Reference: You find more detailed information about J1939 NMEA 2000® in the online help in chapter **Option CANoe.J1939**.

14.2 Quick start

14.2.1 Create a J1939 database

J1939 database

During development, analysis and test of J1939 networks you should use a J1939 database. Different extensions, e.g. the support of J1939 transport protocols, are only available if a J1939 database is assigned to the network in the simulation setup.

If no J1939 database is available, you can create one with the **CANdb++ Editor**. You simply have to define a network attribute **ProtocolType** of type **String**. Set the value to **J1939**.

Alternatively you can use the J1939 template in the **CANdb++ Editor**. In this template, the attribute **ProtocolType** and other J1939 specific attributes are already defined.

To use the extended signal view in the trace window as well in the graphic and data window, you have to create the signal attribute **SigType** of type **Enumeration** in the database. Have a look at the online help to find out, which values are allowed and what meaning they have.

If you use the J1939 template in the CANdb++ Editor, this attribute is already defined too.



Note: The steps to create a NMEA 2000® database are identical to the steps described above. To use the protocol FastPacket the message attribute **SingleFrame** is necessary too.

14.2.2 Create a J1939 configuration



To create a new J1939 configuration, proceed the following steps:

1. Create a configuration via the **CANoe** menu **File|New Configuration....**
2. Select the J1939 template and activate the checkbox **Use Wizard**. Close the dialog with **[OK]**.
3. Press **[Database]** and load the standard database for J1939 in the directory Demo_J1939_CN\Database or select an self-created J1939 database. Next press **[Weiter]**.
4. Add nodes of the list **Available nodes** to the list **Assigned nodes**. Press **[Weiter]**.
5. Select the CAN channel from list **Available channels** and add it to the list **Assigned channel**. Press **[Weiter]**.
6. With **[Fertigstellen]** you can finish the creation of the configuration.

The configuration wizard has now added nodes to the network, assigned the database to the appropriate CAN channel and configured the communication as defined in the database.

Additionally the trace window is configured for J1939 and the measurement setup contains a J1939 Network Scanner and a J1939 Diagnostic Monitor.

14.2.3 Create communication relationships

General

To create communication relationships, you have to assign transmit and receive messages to a node. In the overview window of the **CANdb++ Editor** you can flexible define communication relationships between the nodes. Easier and pre-configured for J1939 specific demands you can do this also with the J1939 communication matrix window.

Communication matrix

In the J1939 communication matrix the relationships between J1939 parameter groups and network nodes are displayed in the table. A cross marks the connection between a transmit node and a receive node.

CANdb++ Editor view

Here you can create communication relationships between nodes easily by drag and drop of the messages to the relevant nodes or by copying the messages and pasting them into the node.

14.2.4 Sample configurations



Note: In order to simplify the initial steps of **CANoe** and to demonstrate the usage of the various functions of J1939 and NMEA 2000®, a few sample configurations are included. You can load these samples directly via the start menu of **Windows**.

14.3 Use cases

14.3.1 Analyze J1939 networks

Analysis

When analyzing communication in a J1939 network, **CANoe** supports you with the following functions:

- ➔ Representation of the transmitted parameter groups in the trace window
- ➔ Representation of signal values in the data and graphic window
- ➔ Recording of the bus traffic in log files
- ➔ Representation of nodes in the network

Trace window

The trace window displays the message traffic with the partial use of symbolic names. You can configure it so that additional J1939-specific columns are displayed, e.g. PGN, Src, Dest and Prio.

Furthermore you can highlight special events (parameter groups), use the extended signal display or filter for special J1939 categories and transport protocols. With the quick find you can easily search for certain information in the trace window.

Data and graphic window

The data window of the option J1939 has an additional column where the status is displayed with a colored indicator. For this the attribute **SigType** in the database is used and evaluated. The graphic window uses this attribute for the extended display too.

Statistic window

With the option J1939 you can change the unit of the X-axis of the statistic window. You can choose the ID range, the PN range or the range of the sender addresses.

J1939 Network Scanner

The J1939 Network Scanner displays all active nodes of the network. Network management parameter groups and others are evaluated. So the active nodes and their states are clearly displayed. For selected nodes you can get more detailed information, e.g. the J1939 device name. You can also send requests for special parameter groups.

14.3.2 Diagnose J1939 networks

Diagnostic

The option J1939 supports you with diagnostic in you network with the J1939 Diagnostic Monitor and the J1939 Memory Access window.

J1939 Diagnostic Monitor

With the J1939 Diagnostic Monitor in the measurement setup you can evaluate the diagnostic messages, specified in SAE J1939-73, of the network. Diagnostic lamps indicate the state of the entire network and the individual nodes as well. The active error codes and their course are displayed for each node.

Furthermore you can request additional diagnostic messages which help with the diagnostic of your network.

Additionally you can copy the displayed data to the the clipboard for further processing or archiving.

J1939 Memory Access window

With the J1939 Memory Access window in the simulation setup you can read out and write memory areas by using specified diagnostic messages. Besides the direct access to the memory area, you can also use an object-oriented access (manual or cyclic). You can read data blocks from files or write these block to files. Furthermore you can execute a Boot-Load process.

J1939 OBD-I/M Monitor

With the J1939 On-Board Diagnostic Inspection and Maintenance Monitor in the measurement setup you can configure diagnostic messages of type DM7 and send them to ECUs in the network. Interpreted as test commands one ore more tests aare executed and the test results are sent back with diagnostic messages of type DM8 or DM30.

14.3.3 Simulate J1939 networks

Simulation

With **CANoe** you can simulate J193 network nodes within the simulation setup.

CAPL

You can program simple nodes directly in CAPL, which is extended for J1939. There is a variable type **pg** available for parameter groups which you can use with a parameter group defined in a database too. This allows a symbolic signal access.

In the handler function **on pg** you can receive parameter groups and evaluate them.

J1939 Interaction Layer

With the J1939 Interaction Layer (IL) you can simulate the sending model of a network node using a node definition from the database. The J1939 IL assumes the sending of the Tx parameter groups, the network management, and the handling of transport protocols. An important feature of the J1939 IL is signal-oriented access, so you can set the signal values directly.



Note: The J1939 IL can manage only one address per simulated node. For simulations in which one node should use multiple addresses (virtual ECUs), you should use the J1939 node layer together with the **J1939 CAPL Generator**.

To use the functionality of the J1939 IL, you have to activate using node layers in the configuration. To do this, create a node attribute **NodeLayerModules** of type **String** in the database. Set the value to **J1939_IL.dll**. Additionally you have to create (or set) the node attribute **NmStationAddress** in the database. Set the value of this attribute to the address which is used by the J1939 IL as send address. Now you can use the extended functionality of the J1939 Interaction Layer.



Reference: In the manual and in the online help of the **CANdb++ Editor** you can find more information about nodes and attributes in databases.

The sample configuration **J1939SystemDemo** explains the use of the Interaction Layer.

J1939 CAPL Generator

The **J1939 CAPL Generator** simplifies the design of a simulation.

You can set the communication relationships between the nodes with the **CANdb++ Editor** and after that you can generate the corresponding CAPL source codes. For this the transmit messages and receive messages are evaluated and the handler functions are generated. The generated source code uses the J1939 node layer.

To use the functionality of the J1939 node layer, you have to activate using node layers in the configuration.

To do this, create a node attribute **NodeLayerModules** of type **String** in the database if it is not already available. Set the value to **J1939_NL.dll**.

Now you can use the extended functionality of the J1939 node layer in your CAPL source files.



Reference: All available configuration settings of the **J1939 CAPL Generator** are described in detail in the appropriate online help.

The sample configuration **SimpleModel** in the directory Demo_J1939_CNModeling demonstrates using the J1939 node layer.



Note: For simple models that basically work signal oriented, and use the transmission types 'cyclic' and 'on change', you can use the J1939 IL.

But if the models become more complex, you shall use the **J1939 CAPL Generator**. Therewith you have more possibilities to influence the behaviour of the simulated ECUs.

Both methods use the same database, so it doesn't matter which one you prefer. But it is recommended to start the model design with the J1939 IL and switch to the **J1939 CAPL Generator** if it is foreseeable that the functionality of the J1939 IL does not fit your requirements.

14.3.4 Test J1939 networks

Testing

For testing complete networks or individual control units, you can use Test Feature Set of **CANoe**. For J1939 it is extended by the J1939 Test Service Library and the J1939 Test Module Manager.

J1939 Test Service Library

The J1939 Test Service Library (J1939 TSL) offers several functions (test patterns and checks) to test J1939 specific mechanisms such as request parameter groups, network management and transport protocols.

You can use the J1939 TSL with CAPL and XML test modules. CAPL offers you more extensive possibilities to modify tests. So it is more useful to incorporate complex tests in a CAPL library and to specify simple tests in XML.

J1939 Test Modul Manager

The J1939 Test Module Manager makes it easier to create J1939 XML tests. You can start it from the **CANdb++ Editor** and create an XML test modules using a database. An assistant helps you getting started and creates a basic configuration (in form of a XML test module) by evaluating the transmit messages and receive messages of a node. Furthermore you can generate J1939-82 Compliance tests too.

Based on this generated test module you can create application specific tests or manually adapt and extend this test module.

14.3.5 Log and trigger J1939 data

Logging	<p>J1939 parameter groups are recorded as normal CAN messages.</p> <p>There is only a special feature when transport protocol parameter groups are recorded in an ASC logging file. The assembled parameter groups with a DLC > 8 are written as J1939 parameter groups in the logging file. You can distinguish these lines from other CAN messages by the identifier J1939.</p>
Replay	<p>CAN messages with an extended CAN identifier from the logging file are replayed as parameter groups.</p> <p>J1939 parameter groups with a DLC > 8 contained in the logging file are ignored when replaying. Only the transport protocol parameter groups are evaluated and from that a combined parameter group is generated.</p>

14.3.6 Trigger and filter J1939 data

Trigger	<p>With the trigger block you can define trigger conditions for parameter groups. These conditions you can insert as symbolic messages.</p> <p>If you select a parameter group from a J1939 database, you can set the sender and receiver address as well as the priority. As an alternative you can insert a raw message and activate the checkbox J1939.</p>
J1939 Message Filter	<p>With the extended message filter you can filter messages by their PG or nodes by their J1939 address.</p>
J1939 Node Filter	<p>With the J1939 Node Filter in the measurement setup you can filter PGs that are sent and/or received by specific nodes. Thereby you can choose between a pass-through filter and block filter.</p>

14.3.7 Modify J1939 signals

Interactive Generator Block	<p>With the interactive generator block (IG) you have J1939 specific columns in addition to the standard columns.</p> <p>You can insert a parameter groups in the interactive generator block by selecting the parameter groups in the database.</p> <p>As an alternative you can enter a 29-bit CAN ID in the column Identifier and add a x, e.g. F00100x. To identify the number as a parameter group number, add a p, e.g. F001p.</p>
CAPL, Panels, Signal generator	<p>You can change signal values with CAPL, Panels or with the signal generator of CANoe.</p> <p>To send the changed values you have to use the J1939 Interaction Layer.</p>

14.3.8 Analyze GNSS data

General	<p>With the option 1939 you can display and log GNSS data (GNSS - (Global Navigation Satellite System)) very easily and clearly. To do this the GNSS Monitor is available.</p>
---------	--

GNSS Monitor

With the GNSS Monitor you can display GNSS positions graphically and numerically. The amount of the display data you can control with a message filter.

You can also log the received positions (logging) and synchronize the data with the trace and graphic window of **CANoe**. This allows a direct assignment of the displayed data in all windows.



Note: The sample configuration **GNSS Monitor** demonstrates the usage of the GNSS Monitor.

14.3.9 Simulate a GNSS receiver**General**

Besides the possibility to analyze GNSS data, you can simulate a GNSS receiver with the option J1939. In doing so you are supported with the GNSS Simulator and the GNSS node layer.


GNSS Simulator

With the GNSS Simulator you can define a path with the help of a geometric figure (model) or a position file. This path will be traced by the simulated GNSS receiver when the simulation is started.

The generated GNSS positions are transferred cyclically with configurable messages of the NMEA 2000® protocol or the J1939 protocol.



With the following steps you can perform a simple simulation:

1. Insert a GNSS Simulator in the simulation setup via the context menu.
2. Before starting the measurement, open the dialog **Settings** with .
3. Select the model which will be traversed during the simulation and set the parameters appropriately.
4. If the simulation should start together with the measurement, set this with the checkbox **Start on measurement**.
5. Now start measurement (and simulation). The GNSS Simulators starts to calculate the positions and to transmit the selected parameter groups cyclically.
6. Watch the simulation in the graphic window.



Note: The sample configuration **GNSS Simulator** demonstrates the usage of the GNSS Simulator.

GNSS node layer

More flexibility in defining the simulated path offers the GNSS node layer. With CAPL functions you can perform a simulation with waypoints, models, a course or by replaying a protocol file.

An overview of the available CAPL functions and its descriptions are contained in the online help.



Note: The sample configuration **GNSS Monitor** demonstrates the usage of the GNSS node layer and panels.

14.3.10 Play back GNSS protocol files

General

In addition to perform a simulation with models, you can do this with protocol files. Such protocol files you can create with the GNSS Monitor by logging the data during a simulation. Due to the simple file format (ASCII) you can easily modify position data and add new positions.

If you play back protocol files with the GNSS Simulator or the GNSS node layer, you can choose between play back with a specified speed or with a user-defined speed.



Reference: More detailed information to these both methods you can find in the online help.

15 ISO11783

In this chapter you find the following information:

15.1	Introduction	page 148
15.2	Quick start	page 148
15.3	Use cases	page 148
	Simulate Virtual Terminals	
	Access process data	
	Simulate a process data dictionary	

15.1 Introduction

Introduction

ISO11783 is a network specification based on the principles of the SAE J1939 definition. ISO11783 is used in the agricultural area and defines the communication between Implements and tractor.

Several services as Virtual Terminal, Task Controller and File Server are specified.



Note: In this chapter only the ISO11783 specific extensions are described. The standard functionality is explained in the previous chapters of this manual.



Reference: In chapter **Option CANoe.ISO11783** of the online help you can find more detailed information about basic principles of ISO11783 and its functionalities.

15.2 Quick start



Note: A brief instruction how to create an ISO11783 database and a configuration is already described in chapter **Option J1939** of this manual.

The procedure is identical to ISO11783, so that you can easily orient on this.

Samples

In order to simplify the initial steps of the option ISO11783, a few sample configurations are included. These samples should demonstrate the use of various functions of the option.

You can start the samples directly via the Windows start menu.

15.3 Use cases



Note: Some typical use cases are already described in chapter **J1939** of this user manual.

The option ISO11783 offers additional extensions that are described more detailed in the further course of this chapter.

15.3.1 Simulate Virtual Terminals

Virtual Terminal

A virtual terminal (VT) is an electronic control unit, consisting of a graphical display and input functions, connected to an ISO11783 network that provides the capability for an ECU, an implement or a group of implements to interact with the user.

The layout of the graphic user interface is defined by an object pool. It defines colors, shapes, pictures, input and output fields that the VT will show on screen. Values and screen positions are also defined.

VT Window

If you don't have a real VT you can simulate one with the VT window of the option ISO11783. Insert this window via the context menu in the simulation setup of **CANoe**.

The VT window shows the Working Sets, which have sent an object pool to the VT and the data masks. With the softkeys you can control the VT. With different views you can make additional settings, watch the objects of the object pool and configure auxiliary inputs.



Note: The sample **Virtual Terminal Demo** demonstrates the use of the VT window.

VT Panel

In contrast to the VT window you can define the graphic layout of the VT with the VT panel. The panel is controlled with the Virtual Terminal DLL and the appropriate CAPL commands.



Reference: More detailed information how to use this DLL and the CAPL functions, you can find in the online help.

J1939 CAPL Generator

With the **J1939 CAPL Generator** you can generate source code for use with a Virtual Terminal. To do this only an object pool file (*.iop) is required.

15.3.2 Access process data

Process data

ISO11783 Part 10 and Part 11 specify process data and their use. These process data can be sum up in a process data dictionary, so it contains all parameters, functions and dimensions.

A process data dictionary can be described in device description file in the standardized XML format.

Interactive Task Controller

To easily access such a process data dictionary you can use the Interactive Task Controller.

At measurement start the process data dictionary is automatically sent to the Task Controller and graphically displayed. A symbolic selection of the process data is possible now and with buttons you can read out process data or write modified process data to the process data dictionary.

J1939 CAPL Generator

With the **J1939 CAPL Generator** you can generate source code for use with a Virtual Terminal. To do this only an device description file (*.xml) is required.

15.3.3 Simulate a process data dictionary

If you don't have a real ECU, you can simulate one with the option ISO11783. An extension to the node layer administrates a device description file (according to ISO11783-10) and creates a process data dictionary.

This is automatically sent to the Task Controller, as if you have a real ECU. Now you can access to the process data with several commands (read value, write value).

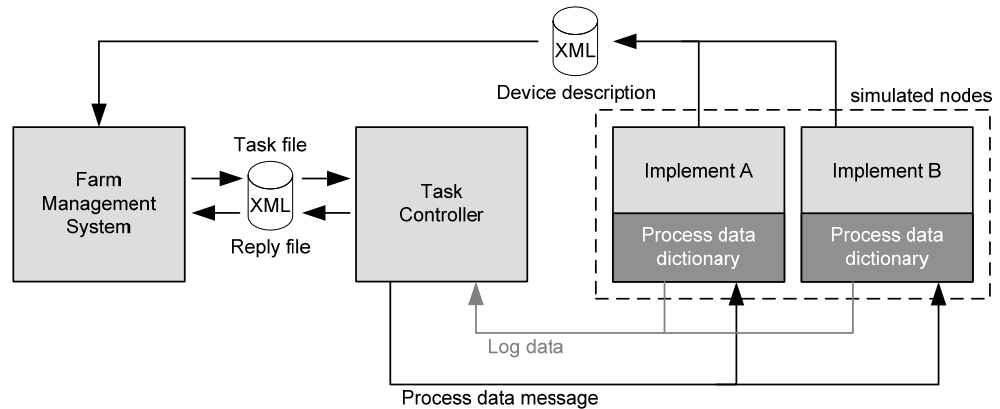


Figure 53 - Process data with ISO11783



To simulate a process data dictionary and access it, follow the next steps:

1. Add a net node to the simulation setup of CANoe which will simulate the ECU.
2. Configure this node so that it will use the node layer.
3. Create the ECU in CAPL.
4. Create a process data dictionary for this device and specify the name of the device description file.
5. Initialize the process data.
6. Connect the process data with environment variables if you want to change the values of the process data or if you want to display the data in a panel.
7. Start the measurement.
8. Use the Task Controller to read value from the process data dictionary or to change values and write them into the process data dictionary.



Reference: A detailed description of the used CAPL functions to execute the steps explained above, you can find in chapter **Option ISO11783 | ISO11783 CAPL | Process data API** of the online help.



Note: The sample configuration **System Demo** shows how you can use the process data dictionary of an ECU.

16 CANopen

In this chapter you find the following information:

16.1	Extended features of option CANopen	page 152
	Trace window	
	CANopen generator block	
	CANopen Scanner	
	Bus configuration	
	Add-ons	
16.2	Databases	page 154
16.3	Generate CANopen simulations	page 155
16.4	Generate tests	page 156
	Device test	
	Application test	
16.5	Control center ProCANopen	page 160
16.6	Sample configurations	page 160

16.1 Extended features of option CANopen

General

The option **CANoe.CANopen** lets you analyze, simulate and automatically test CANopen devices and systems. The standard variant of **CANoe** has been extended to include the necessary CANopen specific functionalities.

Product components

Supplied with the product are:

→ **CANoe** extensions (these will be explained in more detail over the course of this chapter)

→ **CANeds** EDS Editor

CANeds is a tool for creating EDS files in different formats. From a list of available objects, you can assemble the desired object dictionary. But it is also possible to read-in the object dictionary of existing devices via the Scan functionality and automatically generate an EDS.

→ CANopen configuration tool **ProCANopen**

ProCANopen gives you interactive access to CANopen devices and the ability to configure these devices. You can generate simulation models that are based on EDS files at the press of a button and execute them in **CANoe**. In addition, **ProCANopen** contains a generator for automatically creating test procedures that you can execute in **CANoe**.

Activation

You can activate the extensions for **CANoe** by loading the CANopen configuration template.



1. In **CANoe**, choose the menu command **File | New configuration**
2. Select the template **CANopenTemplate.tcn** and close the dialog with **[OK]**.

After creating a CANopen configuration with the CANopen configuration template, the connected CANopen system can be analyzed. Special interpretation of the messages for CANopen does not occur until a CANopen database has been assigned to the configuration (see **Databases** section).



Note: You will find a more detailed description of the **CANoe** extensions and the procedure for activating them (without loading the CANopen configuration template) in the following.

„Silent“ installation

For the option CANopen a silent installation is possible. For this you have to create an *.ISS response file. To create this file start the installation with the following command:

```
setup.exe -a -r -fl<Path of ISS directory\setup.iss>
```

The installation program is set to a recording mode, all settings that are done during the installation are saved in the ISS file.

Finally you can pass this file to the installation program and start the silent installation with the following command:

```
setup.exe -a -s -fl<Path of ISS directory\setup.iss> -f2<Path of logging directory\setup.log>
```

The meaning of the parameters is described here:

- **a** – administrative installation
- **r** – recording mode: saves all settings during the installation process
- **s** – silent installation
- **f1** – path and name of the ISS response file (freely selectable)
- **f2** – path and name of the log file for installation results (freely selectable)

16.1.1 Trace window

General

CANopen specific protocol information is shown to you in the trace window in text format.



Example: If an SDO read access to the object 0x1017 is executed, this is indicated in the **Interpretation** column by the entry:

[1017,00] Initiate Upload Rq.

Columns

The trace window can be extended with these columns:

- **Interpretation** (shows protocol information in text form)
- **Transfer data** (shows the net data), and
- **Error** (displays protocol errors).



To activate these columns in the trace window, perform these steps:

1. In the trace window, open the context menu by right clicking in the window, and choose the **Configuration...** menu command.
2. Select the **Columns** page in the **Trace Window Configuration** dialog.
3. Select the **CANopen** entry in the combination box: **Available fields**.
4. Press the **[Take over std. columns]** button and close the dialog with **[OK]**.

Highlight CANopen messages

You can highlight CANopen messages depending on different categories, e.g. PDO and SDO. This allows a fast overview of the current messages on the bus.

16.1.2 CANopen generator block

General

The CANopen generator block lets you create message sequences conveniently.

From a list of CANopen specific message templates, you can select and configure messages and combine them into a sequence. If the **CANoe** configuration already has a database assignment, you can select messages described there.

Now you can send the created message sequence either once or periodically.



Use the following steps to add a CANopen generator block:

1. Right-click the bus line in the simulation setup to open its context menu.
2. Choose item **Insert interactive generator block CANopen**.
3. Open the CANopen generator block by double-clicking the new **CANopen IG** symbol in the simulation setup.

16.1.3 CANopen Scanner

General

The CANopen Scanner evaluates the CAN messages and displays the active CANopen nodes in a list. Additional information is also shown such as state, device name or master properties.

Network nodes can be detected easily and quickly using the CANopen Scanner when they communicate with one another (SDO or Error Control). This does not have any effects on the bus traffic.



Follow these steps to add a CANopen Scanner to your configuration:

1. Open the context menu of a function block in the measurement setup by right-clicking it.
2. Choose the menu item **Insert CANopen Scanner**.
3. Open the CANopen Scanner by double-clicking the new **CO Scanner** symbol in the measurement setup.

16.1.4 Bus configuration

General

In the **Network Hardware Configuration** dialog you can select one of the baud rates defined under CANopen. The baud rate and associated values for the bus timing registers are then set automatically.



1. In **CANoe** choose the menu command **Configuration | Network Hardware....**
2. Select the item **CAN 1 | CANopen settings** in the tree view.
3. Set the desired baud rate and close the dialog with **[OK]**.

16.1.5 Add-ons

Additional installations

For option CANopen additional installation setups are available in the download center of the Vector homepage. The setups contain a database and if available a sample configuration as well as further modules.

For the following application profiles add-ons are available for download:

- Application profile for building door control (CiA416)
- Application profile for Lift Control Systems (CiA417)
- Device profile for Battery Modules (CiA418)
- Device profile for Battery Chargers (CiA419)
- Application profile for Special-purpose Car Add-on Devices (CiA447)
- FireCAN

16.2 Databases

General

In CANopen, standardized messages such as SDO, Heartbeat or Emergency messages are contained the standard CANopen database. This database is included with the product and is stored in the **Exec32** subdirectory of the installation directory.

The database does not contain any PDO descriptions, since they are created as part of the network creation process using **ProCANopen**.



To create a project-specific database describing all messages existing in the system, follow these steps:

1. Create the network structure using **ProCANopen**:
 - Add network nodes
 - Assign EDS files to the network nodes
2. Configure the PDOs (e.g. via graphic linking).
3. Save the project.



Note: When saving, **ProCANopen** automatically creates a database (**PRJDB.dbc**) in the project directory. A configuration file is also created for each device (may choose either DCF or XCD format).

16.3 Generate CANopen simulations

General

Simulation of network nodes plays a significant role in the development of CANopen devices and systems.

The CANopen standard specifies that device information and functions must be stored in a prescribed format in EDS files. Based on this information, CANopen simulation models may be created with **ProCANopen** and are then ready to be executed in a runtime environment (**CANoe**). The generated simulation models can be extended to include application-specific behavior to complete the total system.

Generation process

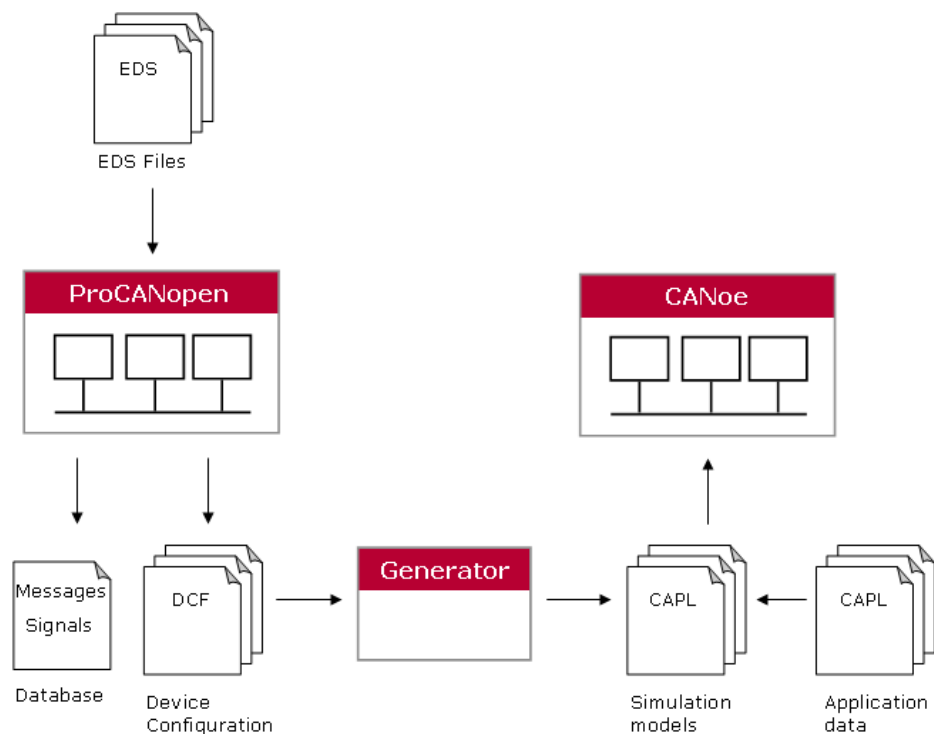


Figure 54 – Generation of a CANopen simulation



The following steps are necessary to create a simulation model:

1. Create the necessary (missing) EDS files with **CANeds**.
2. Create the network topology (linking the EDS files) with **ProCANopen**.
3. Configure the network with **ProCANopen** (e.g. set up PDOs).
4. Generate the simulation models with **ProCANopen**.
5. Simulate the total system with **CANoe**.
6. Use the **CAPL Editor** to extend the generated models to include application-specific behavior



Reference: For detailed step-by-step instructions please refer to online help for the **CANoe.CANopen** option.

16.4 Generate tests

General

EDS files describe the functional features of a CANopen device. These device descriptions form the foundation for implementing the simulation and creating test specifications.

Device tests

Device tests can be derived directly from the device descriptions.

For example, a test might be used to check the access types of all objects in the object dictionary by SDO and record the results.

Application tests

Application tests, however, cannot be created based on the EDS files, since additional knowledge of the application is needed to create these tests. But it is possible to generate a test framework.

For example, transmission of the digital input of an I/O device may be stimulated to then compare the signal value at the receiver's output.

Both tests can be applied easily to the simulated total system. As soon as the system design has been created, it is possible to subcontract development of individual components; the tests that have already been created can be applied to the real total system at later time.

16.4.1 Device test

General

You can create tests easily and conveniently using the Test configurator supplied in **ProCANopen**.

The test configuration is assembled from a number of prepared functional templates. Project specific information such as node numbers or object indices are available in relevant selection boxes, and they support you in creating test sequences.

Based on the test configuration, a generator creates a test sequence that is immediately available for execution in **CANoe**. All test results are automatically recorded in a report file.

Generation process

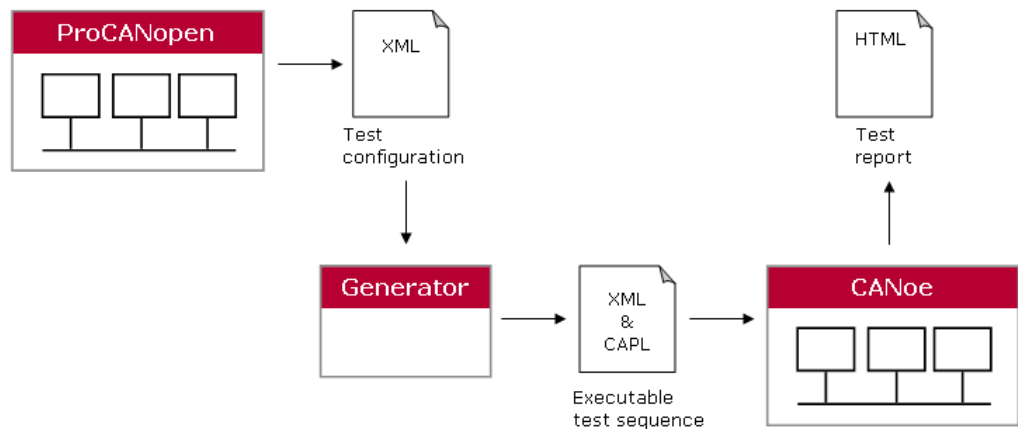


Figure 55 – Generation of a test sequence

CANopen Test Assistant

Using the CANopen Test Assistant to create test sequences is even easier and considerably more effective.

You can generate entire test sequences from a number of prepared generation templates. Individual test functions are automatically combined into a test sequence and parameterized. The EDS/DCF files for the devices form the basis for generation.

The following generation templates are available:

- ➔ Check of the object dictionary values
- ➔ Check the object dictionary access
- ➔ Check for hidden objects
- ➔ Check the device
- ➔ Check the transmit PDO (TPDO)
- ➔ Check the receive PDO (RPDO)
- ➔ Check SDO-Server download
- ➔ Check SDO-Server upload

16.4.2 Application test

General

The application behavior of the devices cannot be written to the EDS files. Therefore, the **CANoe.CANopen** option lets you generate a test framework that simplifies creation of application tests immensely.

In the test sequences, the simulated devices can be “remote controlled” by environment variables. On the one hand, PDOs can be triggered, and on the other hand reading or writing of an object value by SDO can be initiated by SDO in a simulated node.



Note: The Test Automation Editor from Vector makes it easy to create application tests (based on XML), which can be executed directly in **CANoe**. From a list of predefined templates, it is possible to put together any desired test sequences (e.g. setting and evaluating environment variables).

16.4.2.1 Stimulate PDOS

As part of the generation process, a simulation model is created for each node of the system (see [Generate CANopen simulations](#) section). Among other things, this model implements the PDO configuration that has already been described with [ProCANopen](#).

If an environment variable that symbolizes the switch of an input is set, the associated PDO transfer is triggered (`set signal (1)` triggers PDO (1)). In another environment variable, the state of the mapped output is saved for comparison purposes (`get signal 1`).

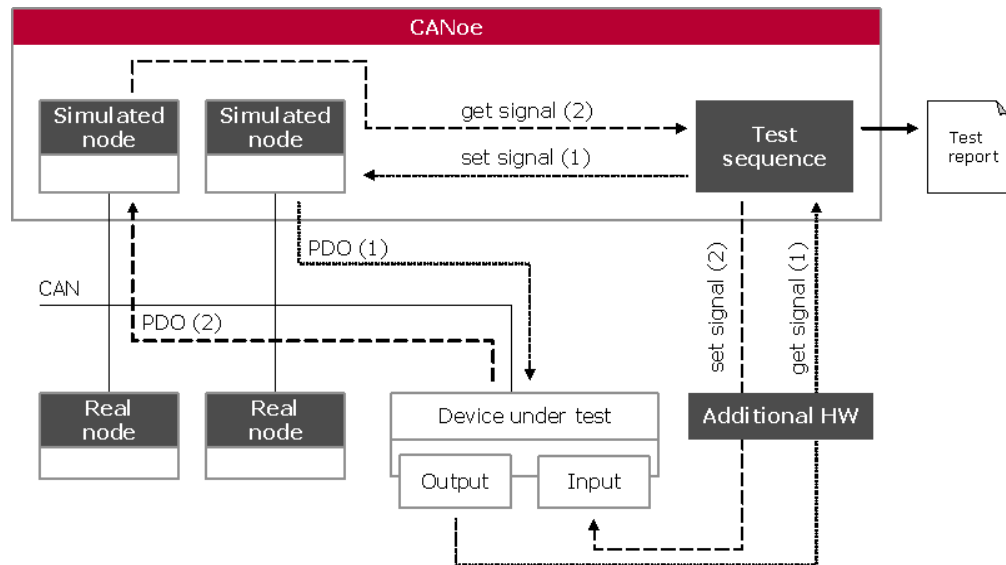


Figure 56 – Test setup

A significant advantage of this approach is that test sequences can also be applied to mixed systems (simulated subsystem plus real components) or real systems.

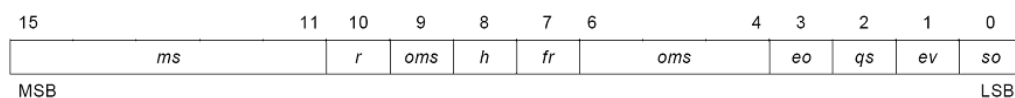
When an additional hardware device is used (e.g. I/O cab from the Vector company) environment variables can be linked to physical inputs or outputs. When the environment variable that symbolizes the input is changed, it is no longer the simulated node that is stimulated, rather the input (switch) of the real ECU via the hardware. The output value of a real ECU is mapped to another environment variable whose value can then be compared. Accordingly, test sequences that have already been developed before the creation of the individual ECUs can later be applied to the real complete system.

16.4.2.2 Run the signal-based SDO transfer

General

Signal-based SDO transfer can also be triggered via the simulated nodes.

The “Controlword” object (Index 0x6040) of the device profile “CiA402 – Drives and motion control device profile” will be used as an example to explain the method for creating a test that implements signal-based access.



LEGEND: ms = manufacturer-specific; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; eo = enable operation; qs = quick stop; ev = enable voltage; so = switch on

Figure 57 – Object description "Controlword" (Excerpt from CiA402)

Signals

Besides object attributes, the CANopen standard also defines the subdivision of the object value (signal group) into individual signals, each with a specified length and start position (in bits). In generating the simulation, a set of environment variables is created for each signal. These variables can be used to initiate a SDO transfer in a simulated node.

For example, the following environment variables are created for the signal "oms":

Variable name	Description
<code>sd_operation_mode_specific</code>	Environment variable that can be used to initiate a SDO download in a simulated node. The signal value to be written is given beforehand in the variable <code>val_operation_mode_specific</code>
<code>rd_operation_mode_specific</code>	Initiates a SDO upload in a simulated node. The read signal value is located in the following variable after a completed SDO upload: <code>val_operation_mode_specific</code> .
<code>val_operation_mode_specific</code>	Value of the signal

In creating application tests, it does not matter in which object or at which location a signal is represented. Of greater interest is information about which signals exist and which devices receive or send them. It is precisely these aspects that the user also needs to test. In creating tests, only the signal names and their values are of interest. That is why the signal name is reflected in the name of the environment variable.

In the test flow, the environment variable `val_operation_mode_specific` is initialized with just one signal value. When the variable `sd_operation_mode_specific` is set, this triggers a SDO download. Shifting of the signal value to the relevant position in the object value and execution of the SDO download to the object 0x6040 (in a target node to be indicated) is fully accepted by the simulated node. This information does not appear in the test flow!

16.5 Control center ProCANopen

General

In the development, analysis and testing of CANopen devices or CANopen systems, **ProCANopen** is the control center for creating device models and test sequences.

Functional features

This especially includes the following tasks:

- Creating a project specific database (see **Databases**)
- Generating device specific configuration files (DCF/XDC files)
- Creating executable simulation models (see **Generate CANopen simulations**)
- Generating test sequences (see **Device test**)
- Creating a test framework in which application tests can be executed (see **Application test**)

Furthermore, you can easily and quickly configure CANopen devices and systems with **ProCANopen**. For example, PDO mapping is automatically calculated in the graphic linking of process data. The configuration data are then transmitted to the ECUs in the network at the push of a button.

For error analysis, functions are available to you that implement interactive access to the object dictionary of connected ECUs.

CANeds

EDS files – which form the basis for simulation and test generation (see **Device test** section) – generally do not exist yet for the devices in development.

The EDS editor **CANeds** that is integrated in **ProCANopen** lets you create EDS files easily and quickly. From a list of available objects, you can assemble an object dictionary by drag-and-drop operation.

First, the resulting EDS file forms the basis for model generation and test creation. Second, the created EDS file can be delivered with the finished end product.



Reference: For further information on using **ProCANopen** and **CANeds** please refer to the relevant manual or online help.

16.6 Sample configurations

Samples

With the option CANopen for **CANoe**, you get several sample configurations to get used with the option CANopen and its various functions.

You can start these sample configurations directly via the Windows start menu.



Reference: You can find more detailed information how to use the sample configurations in the online help and in the configuration comments in **CANoe**.

17 IP

In this chapter you find the following information:

17.1	Extensions of the IP option	page 162
	Check installation	
17.2	Security advice for using CANoe.IP	page 163
	Exclusive use of an Ethernet interface	
17.3	Use cases	page 164
	Analyze Ethernet networks	
	Filter Ethernet data	
	Stimulate Ethernet packets	
	Simulate Ethernet nodes	
	Run Remote CAN analysis	
17.4	Quick start	page 165
	Analyze network traffic	
	Evaluate signals	
	Run Remote CAN analysis	
	Sample configurations	

17.1 Extensions of the IP option

General

The option IP for **CANoe** can be used in the field of embedded Ethernet applications. For this several extensions of the standard **CANoe** are available.

Embedded Ethernet

For embedded Ethernet networks, you can:

- use Ethernet communication
- stimulate Ethernet packets with the Ethernet Packet Builder and with CAPL
- configure several Ethernet channels and use them together with other bus systems or standalone
- decompose and analyze Ethernet packets by means of CAPL
- evaluate Ethernet packets and its protocols in the trace window
- define signals in the **CANdb++ Editor** and analyze them with self-configurable signal protocols or with panels
- use the graphic and data window as well as the message filter with signal protocols
- log Ethernet bus traffic in ASC and BLF logging format, and replay log-files in **CANoe** offline mode or with the replay block



Note: In which cases you can use the different extensions is described in detail in the further course of this user manual or in the online help system.



Note: This product includes software developed by the University of California, Berkeley and its contributors.

17.1.1 Check installation

Check installation

Check to make sure the program is functionally capable by means of one of the sample configurations in the **EthernetLoopTest** folder of your application data. These configurations utilize a loopback connection.

In case you use two Ethernet interfaces connect them directly together with a crossover cable and use the configuration **EthernetLoopTest_2Ch_CN.cfg**. If you use only one Ethernet interface, connect a loopback cable or a loopback adapter (at 10 or 100 Mbps) to it and use configuration **EthernetLoopTest_1Ch_CN.cfg**.

Configure the Ethernet interfaces in the **Network Hardware Configuration** dialog of **CANoe.IP**. Afterwards start the measurement of the sample configuration. Press **[Output Packet]** in the tester panel and check if the result indicator reports **[Pass]**.

If you use your **CANoe.IP** for Remote CAN Analysis, install the CAN-(W)LAN gateway first according to the installation instructions of the manufacturer.

Then configure the CAN-(W)LAN gateway in the **Vector Hardware Config** dialog (**Windows Control Panel | Vector Hardware**). The installation and the connection to the CAN-(W)LAN gateway is correct, if a short description is displayed for the module instead of **Remote Bus 1** or **Remote Bus 2**.

17.2 Security advice for using CANoe.IP



Caution: Please take care if your computer is connected to a network. It is possible that you send packets to the network that may cause problems.

17.2.1 Exclusive use of an Ethernet interface

You can configure your Ethernet interface for an exclusive use by **CANoe.IP**. That prevents **Windows** and other applications using this interface and influencing a connected embedded network. Vice versa an embedded network cannot influence your **Windows** system.

With the NetIsolator you can configure this setting easily and simplify the use of an Ethernet interface, which is connected e.g. to an office network sometime and to an embedded network another time.

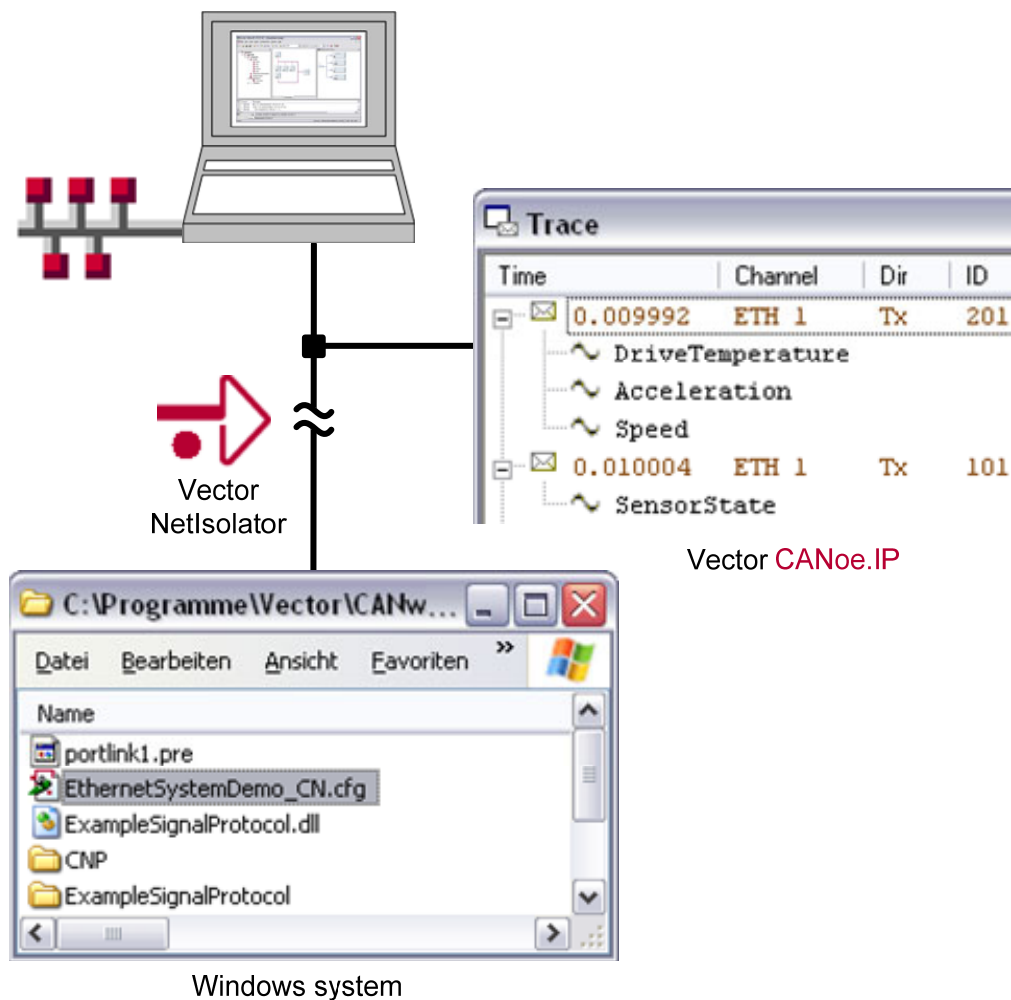


Figure 58 - Using the NetIsolator for exclusive interface use

17.3 Use cases

General In this chapter some use cases are described to illustrate how you can use the different extensions of the option IP.

17.3.1 Analyze Ethernet networks

General You can analyze Ethernet packets and messages on the bus with the trace window as well as the data and graphic window of **CANoe.IP**.

Trace window The trace window shows you the Ethernet packets which are received at the Ethernet port of your computer, independent if your computer is the receiver of the packet or not.



Note: If your network system has included a switch, the configuration of it influences how packets are directed to your computer. So it is possible that you don't see all packets in the trace window but only these which are send directly to your computer.

Data and graphic window You can analyze signals by using a configured signal protocol and a database (*.dbc). The signals are shown in the data and the graphic window of **CANoe.IP**.

Panels Via panels and environment variables you can also analyze your network, if you don't have a signal protocol DLL or don't want to use one.

17.3.2 Filter Ethernet data

IP Filter With the IP Filter in the measurement setup you can filter Ethernet packets by using addresses and protocols as filter conditions. If you have more complex use cases you can use conditions with protocol fields. You can use the IP filter as a pass-through filter and block filter.

17.3.3 Stimulate Ethernet packets

General You can build Ethernet packets with the included Ethernet Packet Builder of **CANoe.IP** and send them to your network. You can build a list with different Ethernet packets, which should be send, and configure each of them with its own data. In **CANoe.IP** you also have the possibility to send Ethernet packets from CAPL.

17.3.4 Simulate Ethernet nodes

General With the option IP you can simulate Ethernet nodes by sending and receiving Ethernet packets with CAPL. This allows you to realize Ethernet to Ethernet, Ethernet to CAN and CAN to Ethernet gateways too.

Packet API You can use the Packet API of the Ethernet Interaction Layer to create and send Ethernet packets. You can access the protocol fields of the packets and read/modify the values. Some fields, e.g. the checksum, you can automatically calculate.

Socket API

With the Socket API of the Ethernet Interaction Layer you can realize End-to-End communication. The protocols TCP and UDP are supported for the server and the client socket.

17.3.5 Run Remote CAN analysis**General**

With **CANoe.IP** you can access a CAN network with remote access via a CAN-(W)LAN gateway, using the complete functionality as if the CAN network is locally connected.

You have to create and configure a CAN network in **CANoe**, and a CAN-(W)LAN gateway which is used to access the remote CAN network in the **Vector Hardware Config** dialog (**Windows Control Panel | Vector Hardware**).

You can use several CAN-(W)LAN gateways, the configuration depends on the module itself. For details see the manual of the module.



Note: Due the WLAN connection, time delays during the packet transmission are possible. The time stamps are generated on the CAN-(W)LAN gateway directly.

17.4 Quick start**17.4.1 Analyze network traffic****Analyze network traffic**

To analyze an Ethernet network, do the following steps to create a configuration and use it:



1. Start **CANoe.IP**.
2. Create a new configuration with **File | New ...**.
3. Use the template **Ethernet**. An Ethernet channel and the trace window are pre-configured.
4. Select your Ethernet interface in the dialog **Network Hardware Configuration** that you can open via the menu **Configuration | Network Hardware ...**.
5. Start measurement and have a look at the trace window.

17.4.2 Evaluate signals**Evaluate signals**

If you want to evaluate signals with a signal protocol, you have to set up the following steps:



1. Select the signal protocol of the Ethernet channel in the dialog **Network Hardware Configuration**. There you have to set the DLL (Dynamic Link Library) that should evaluate the signals.



Note: The DLL must be created and configured by yourself. The Visual Studio 2005 project is delivered with the option and an example DLL is contained in the sample configuration **EthernetSystemDemo**.

2. Add a database and set the bus type to **Ethernet**. Alternatively use the Ethernet template in the **CANdb++ Editor**.

3. Add messages and signals to this database.
4. Add signals from the database to data and graphic window. Use the message filter for evaluating your network signals too.

17.4.3 Run Remote CAN analysis

Remote CAN analysis



To access a remote CAN network via a CAN-(W)LAN gateway, follow the next steps:

1. Configure and connect your CAN-(W)LAN gateway to the CAN network that it is accessible from the computer on which **CANoe.IP** is running. Have a look at the manual of the adapter to find more details about its configuration.
2. Create a (W)LAN connection between the CAN-(W)LAN gateway and the windows system.
3. Open the **Vector Hardware Config** dialog via the control panel of Windows (menu entry **Vector Hardware**)
4. Select the entry **Remote Bus 1** or **Remote Bus 2** at the menu item **Hardware** and insert the necessary parameters for the connection:
 - **Protocol**: select the appropriate device protocol
 - **Transport**: select the appropriate transport protocol
 - **IP Address**: set the IP address of the CAN-(W)LAN gateway
 - **Port Number**: set the used port number



Note: More information how to connect CAN channels you can find in the online help of the **Vector Hardware Config** dialog.

5. In **CANoe** create a new configuration via the menu **File | New Configuration...**
6. Open the menu **Configuration | Options** and switch to page **Channel Usage**. Set the number of CAN channels that you want to use (e.g. **1** for a single CAN channel).
7. Start measurement.



Note: Certain capabilities regarding the operation modes during the measurement may vary depending on the used device, hardware and/or firmware versions. For details have a look at the manual of the device.

17.4.4 Sample configurations

Sample configurations

With the option IP you get several sample configurations. Use these configurations to do the first steps with the option IP and to get used with the different extensions.

You can start the samples directly via the Windows start menu.



Reference: More detailed information you can find in the online help system.

18 J1587

In this chapter you find the following information:

18.1	Introduction	page 168
18.2	Prerequisites	page 168
	Configure J1708 channels	
	Define J1587 parameters in CANdb++ Editor	
18.3	Functionality	page 169
	Parameter Monitor	
	Diagnostics Monitor	
	Trace window	
	Data and graphic window	
	Interactive Generator Block	
	Filter	
	CAPL	

18.1 Introduction

Introduction

The option J1587 offers extensive capabilities for analyzing distributed real-time control systems with J1587.

With the J1587 option, you are able to

- receive,
- display,
- analyze, and
- send J1587 messages.

Hardware Interfaces

To connect to J1587 interface the following network interfaces and data loggers are available.

Network interface/Data logger	Bus transceiver
→ CANcardXL	→ 1708cab 65176opto
→ CANcardXLc	
→ CANcaseXL	→ 1708piggy 65176opto
→ CANcaseXL log	
→ CANboardXL	
→ CANboardXL PCIe	
→ CANboardXL pxi	
→ VN8900	

Sample configurations

In order to simplify the initial steps of the option J1587, a few sample configurations are included. These samples should demonstrate the use of various functions of the option. You can start them directly via the Windows start menu.

18.2 Prerequisites

18.2.1 Configure J1708 channels

Overview

To use a J1708 channel with **CANoe**, the channel must be configured using the **Vector Hardware** dialog. This application can be started from the Windows Control Panel.



1. With **Edit | Application** add an application with the name **CANoe** if it does not already exist.
2. Open the context menu with a right click and choose the entry **Application J1708 channels**.
3. Enter the number of channels you want to use.
4. Open the tree node **CANcardXL** (node **Hardware**) and select the entry **J1708cab 65176opto**.
5. Open the context menu and assign the application, you defined in step 1, to the **CANcardXL**.

If you use another network interface or a data logger for configuration of the network, the steps are analog to the steps described above.

CANoe

To use J1587 with **CANoe**, the number of J1708 channels must be configured. The setting is available under **Configuration | Options...** on page **Channel Usage**.

Configuration | Network Hardware... shows the available J1708 channels.

18.2.2 Define J1587 parameters in CANdb++ Editor

Overview

The parameter identifier (PID) in a **CANdb++** database is made up of the PID and the page of the PID. **CANoe** shows the page and the PID as one value.



Example:

PID 84 (54₁₆) – PID 84 of page 1

PID 340(154₁₆) – PID 84 of page 2

J1587 parameter

A J1587 parameter can be defined in the **CANdb++ Editor**. It is equivalent to a CAN message in the **CANdb++ Editor**, the CAN identifier is equivalent to the PID.

In the context menu of a J1587 parameter (or a CAN message) the menu item **J1587 Properties** can be selected. In this dialog the PID and the page can be edited separately.

Message list

The message list contains some additional columns for J1587:

- **J1587 Type**
This column shows the type of the parameter.
- **J1587 PID**
This column shows the PID without page.
- **J1587 Page**
This column shows the page of the PID.

18.3 Functionality

18.3.1 Parameter Monitor

Overview

The Parameter Monitor shows signal values of J1587 parameters transmitted within J1708 messages. It provides a structured view with all values mapped to the corresponding send node.

Functions

The following information is displayed:

Field	Description
Name	Level 1: name of the send node (MID)
	Level 2: parameter name (PID) / signal name (for parameters containing only one signal)
	Level 3: Name of the signal within a parameter

Field	Description
Chn	J1587 channel: number of the channel on which the message was sent/received
Time	absolute time since measurement start in seconds
PID	parameter ID (PID) of the message
Value	physical (if applicable) / raw signal value
Unit	unit of the signal value (if applicable)
Value Table Description	description of the signal value transmitted ("Value Table" entry mapped to the signal value in CANdb++ database)

18.3.2 Diagnostics Monitor

Overview The J1587 Diagnostics (DC) Monitor examines the transmitted diagnostic messages (PIDs 194-196), displays their contents, and enables requests in order to retrieve diagnostic data.

Functionality It exposes the following functionality:

- Overview of ECUs sending diagnostic messages
- Request of all diagnostic codes
- Display of active diagnostic codes of a specific ECU
- Display of inactive diagnostic codes of a specific ECU
- History View: Illustrates status transitions of diagnostic codes
- Data Request View: Enables requests of additional information ("Descriptive SCI/proprietary message") of a specific code; Deletion of an/all Occurrence Count(s); Display of response messages

18.3.3 Trace window

Overview When configuring the trace window you can choose J1587 specific fields to be displayed.

Functionality For option .J1587 the following columns are available in the trace window:

Field	Title	Function
Source	MID	MID of the message
Destination	Receiver	receiver MID for proprietary and transport protocol messages
Name	Name	Name of the parameter, if the message contains only one parameter or „...“, if the message contains more than one parameter.
DLC	DLC	Length of the data field, inclusive MID and checksum

Field	Title	Function
Data	Data	The message data in decimal or hexadecimal representation. In the symbolic view brackets separates the PID and the value of the parameter. In the numeric view raw data is show, inclusive MID and checksum. In the symbolic representation transport protocol and proprietary messages a show interpreted.
Send node	Send node	Displays the send node, which is defined in the database for the MID of the message.



Note: With the **Sym/Num**-Button in the symbol bar the display format of the **Data** column can be changed. The numeric view shows the raw data of a message, including MID and checksum. In the symbolic view the PIDs are highlighted and transport protocol/request messages are interpreted.

18.3.4 Data and graphic window

Overview

The data and graphic window shows the signals of J708 messages. The signals can be selected for a database in the configuration dialog.

Open the context menu of the window, select the **Add signals...** and select the signals, you want to add, in the dialog **Signal selection**.

18.3.5 Interactive Generator Block

Overview

The Interactive Generator can send J1587 messages.

A message can be inserted from the database with the button **[New]**. A click on the arrow of the button **[New]** opens a menu. With the menu item **J1587 Message** a new message can be defined. The user must enter a PID in the column **ID**.



Note: The interactive generator block can send only one parameter per message.

The interactive generator block contains a tab page **J1587** with three J1587 specific columns:

Column	Description
MID	The message is sent with this MID.
Receiver	MID of the receiver This column is enabled for proprietary messages only.
Priority	The message is sent with this priority.

18.3.6 Filter

Overview

With the filter it is possible to pass or block a specific PID. The default setting is to filter every message with the specified PID, not considering the MID. If the PID should be filtered only for a specific MID, the MID can be entered in the column **Source** on the tab page **J1587**.

18.3.7 CAPL

Overview

With **CAPL** J1708 messages and J1587 parameters can be received and transmitted. You can do this by using the variable type **J1587Param** and **J1587Message** in **CAPL**. Variables of this type can be transmitted with `output()`. To receive a J1587 parameter a handler named `onJ1587Param` and `onJ1587Message` must be present in the **CAPL** program.



Reference: For more information about the **CAPL** programming with J1587 is available in the online help at **Option J1587 | CAPL**.

19 CANaero

In this chapter you find the following information:

19.1	Scope of delivery	page 174
19.2	Basics	page 174
19.3	Database Concept	page 174
19.4	Extensions	page 175
	Trace window	
	Data window	
	Interactive Generator Block	

19.1 Scope of delivery

Extensions

The option **CANaero** contains the following **CANoe** extensions:

- ➔ Extensions of the standard components: trace window, data window, CAPL, interactive generator block
- ➔ Standard database
- ➔ Example configurations, demonstrating some of the provided features



Note: The possibilities that are specifically presented by these extensions and how they can be used, are described in detail in the further course of this user manual or in the online help system.

19.2 Basics

Overview

Since years aerospace technology successfully is using several bus systems for connecting avionics and supplemental electronics. One of those bus systems is CAN. Higher-layer protocols shall help getting more and more interoperability between sub-systems and for reducing the efforts of integration of electronic components.

Examples for such protocols are **CANaerospace**, ARINC 810, ARINC 812, and ARINC 825.

The option **CANaero** extends the Vector standard tool **CANoe** with support for **CAN**-based **aerospace** protocols, currently this is **CANaerospace**.

CANaerospace

The company Michael Stock Flight Systems developed the interface standard **CANaerospace** after having gained profound experiences from multiple projects in the field of aerospace technology.

The protocol considers the special demands of aerospace technology regarding security, capability of certifications, simple application and easy accessibility. It can be used with CAN 2.0A and 2.0B (11 bit and 29 bit identifier) and any bus rate data.

Sample configurations

In order to simplify the initial steps of **CANoe** and the option **CANaero** sample configurations are included to demonstrate the use of various functions of your application.

You can start the sample configurations directly via the Windows start menu.

19.3 Database Concept

General

CANoe intensively uses a database driven concept.

That means all blocks and windows are connected to a database. The delivery of the option **CANaero** includes a database for **CANaerospace** already. If you use this database and assign it to the bus in your configuration, different extensions are active.

CANaerospace database

The database entries for **CANaerospace** contains message descriptions, signal descriptions and data type definitions. All pre-defined messages, e.g. EED (Emergency Event Data), Service Requests or NOD (Normal Operation Data), from the **CANaerospace** specification V1.7 are represented in the delivered database.



Note: Whenever a new configuration is build, one of these databases or a derived one has to be assigned to that configuration.



Reference: Please refer to the appropriate chapter of this manual for a description of database usage and how to assign a database.

19.4 Extensions

19.4.1 Trace window

Overview

The trace window displays the CAN message traffic together with the symbolic names of the messages. After each message the corresponding **CANaerospace** specific fields are displayed.

Columns

In the context menu of the trace window select **Configuration | Columns**. The field **Available fields** allows to select the required protocol, here **CANaerospace**.

19.4.2 Data window

Overview

The data window shows the actual values of the transferred data. The selection of the objects is done with the help of the CANdb database. The selection dialog shows all available messages. On selection of a message, the right list shows all signals (data fields) of that message.

The data window will display a signal only, if all attributes of the message (device id etc.) are entered correctly in the database. A signal with the same value of a different device is treated as a different signal and therefore not shown!

19.4.3 Interactive Generator Block

Overview

As in standard **CANoe** the Interactive Generator Block offers the possibility of transmitting messages triggered by click on a button or cyclic with interactive changing of the data contents.

In the upper pane the message and the triggering is selected. The lower pane displays the message fields of the actually selected line. The required values have to be entered here. When selecting the data type, the corresponding signal from the database will appear, if this data type dependent signal is part of the **CANaerospace** specification.

With a click on **[Send now]** the message is transmitted. In trigger mode **cyclic** the message will be transmitted with the corresponding cycle time. Every change of the data values (here e.g. of CAL_AIRSPEED_float) will take effect on the next transmission.

20 Appendix A: Support

Need support?	<p>Our hotline can be reached</p> <ul style="list-style-type: none"> → by calling +49 (711) 80670-200 → by e-mail (support@vector-informatik.de) → or by filling out our Problem Report form online
What our support team needs to know	To answer your support questions quickly, whether by phone, e-mail, fax or mail, we require the following information:
Software	<ul style="list-style-type: none"> → Detailed description of the software, hardware model and version number, e.g., CANoe 5.2.70 (SP3), CANcardXL → Serial number <p>Note: You will find this information in CANoe under Help Info.</p>
Hardware	<ul style="list-style-type: none"> → Exact description of the hardware (e.g. CANcardXL) → Hardware serial number → Driver and firmware versions <p>You will find this information in the Windows menu under Start Settings Control Panel Vector Hardware.</p> <p>Select the CANcardXL or CANcaseXL entry in this menu and click on [Hardware-Info].</p> <p>If you have hardware problems:</p>
Computer	<ul style="list-style-type: none"> → The vcaninfo.exe log file → Detailed description (e.g. Toshiba Tecra 8000) → Laptop or desktop PC → Operating system (e.g. Windows 2000, SP 4) → Processor type and speed (e.g. Pentium III, 1 GHz) → Memory (e.g. 256 MB RAM)
PCMCIA card	<p>If you have problems with PCMCIA cards in a desktop PC:</p> <ul style="list-style-type: none"> → Detailed description of the PCMCIA drive used
Error description	<ul style="list-style-type: none"> → What problems occurred? → Which configuration did these problems occur with? → Are you getting error messages in the software, e.g. in the Write window?
Customer data	<ul style="list-style-type: none"> → Company, company address → First name, last name → Department → Telephone number, fax number, e-mail address

21 Appendix B: Address table

Vector Informatik GmbH	Vector Informatik GmbH Ingersheimer Str. 24 70499 Stuttgart Germany Phone: +49 711 80670-0 Fax: +49 711 80670-111 mailto:info@de.vector.com http://www.vector-informatik.com/
Vector CANtech, Inc.	Vector CANtech, Inc. Suite 550 39500 Orchard Hill Place Novi, Mi 48375 USA Phone: +1 248 449 9290 Fax: +1 248 449 9704 mailto:info@us.vector.com http://www.vector-cantech.com/
Vector France SAS	Vector France SAS 168, Boulevard Camélinat 92240 Malakoff France Phone: +33 1 4231 4000 Fax: +33 1 4231 4009 mailto:information@fr.vector.com http://www.vector-france.com/
Vector GB Ltd.	Vector GB Ltd. Rhodium Central Boulevard Blythe Valley Park Solihull, Birmingham West Midlands, B90 8AS United Kingdom Phone: +44 121 50 681-50 Fax: +44 121 50 681-66 mailto:info@uk.vector.com http://www.vector-gb.co.uk

Vector Informatik India Private Limited	Vector Informatik India Private Limited 4/1/1/1 Sutar Icon Sus Road Pashan Pune 411021 India Phone: +91 9673 336575 Fax: - mailto:info@in.vector.com http://www.vector-india.com
Vector Japan Co., Ltd.	Vector Japan Co., Ltd. Seafort Square Center Bld. 18F 2-3-12, Higashi-shinagawa, Shinagawa-ku 140-0002 Tokyo Japan Phone: +81 3 5769 7800 Fax: +81 3 5769 6975 mailto:info@jp.vector.com http://www.vector-japan.co.jp/
Vector Korea IT Inc.	Vector Korea IT Inc. # 1406 Mario Tower Guro-dong, Guro-gu, 222-12 Seoul, 152-848 Republic of Korea Phone: +82 2 8070 600 Fax: +82 2 8070 601 mailto:info@kr.vector.com http://www.vector-korea.com/
VecScan AB	VecScan AB Theres Svenssons Gata 9 417 55 Göteborg Sweden Phone: +46 31 7647600 Fax: +46 31 7647619 mailto:info@se.vector.com http://www.vecscan.com/

22 Index

3

3 phase model22

A

Address table178

Ambiguities65

Analysis140, 143

B

Box types in dialogs.....25

Bus configuration154

Bus Statistics window132

Button25

C

CAN115

CAN interface78

CANaerospace174

CANdb++ Editor.....169

CANeds152, 160

CANoe architecture27

CANoe Realtime75

CANoe tour30

CANopen device configuration152, 160

CANopen Generator Block153

CANopen Scanner.....154

CANopen template152

CANopen Test Assistant.....157

CAPL.....109, 144, 172

CAPL basics110

CAPL Browser112

CAPL nodes.....99, 104

Channel filter.....103

Channels in offline mode67

Channels in online mode66

Channels in simulation mode66

Check box25

CHI130

COM Server76

Comment box.....25

Communication relationships.....139

Consistency55, 65

Conventions8

Converting log files70

D

Data window91, 121, 133, 164, 171, 175

DBC databases138, 154, 174

Demo driver28

Demo version.....28

Desktop concept82

DFS55, 74

Diagnostic140, 170

Diagnostic Feature Set (DFS).....74

Diagnostics55, 74

Diagnostics Console95

Drop down list25

E

Embedded Ethernet.....162

Environment variable164

Environment variables23, 67

Error messages.....55, 78

Ethernet Interface163

Exporting log files.....70

F

FastPacket transport protocol.....138, 139

Fault Memory window96

FIBEX.....130

Filter99, 103, 143, 172

Filter block.....103

FlexRay.....	129
FlexRay database.....	130
FlexRay Frame Panel.....	135
FlexRay network.....	131
FlexRay PDU Panel.....	135
FlexRay signals	133
FlexRay traffic.....	132

G

GNSS Monitor.....	144
GNSS Simulator	144
Graphic window	164, 171
Graphics window	88, 121, 133

H

Hardware	163, 168
Hot spots.....	100

I

Interactive Generator Block	101, 122, 171, 175
Interactive Master	119
Interactive Task Controller.....	149
Introduction to CANoe	22
IP Filter	164
IP Security advice.....	163

J

J1587 Diagnostics Monitor	170
J1587 Parameter Monitor	169
J1939	138
J1939 CAPL Generator	149
J1939 Communication matrix	139
J1939 Interaction Layer	141
J1939 Node Filter	143

L

LDF	118
LDF Explorer.....	118
Licensing.....	13
Limitations.....	28
LIN	117

LIN description file	118
LIN Master's scheduler	119
LIN network.....	119
LIN signals	121
LIN traffic.....	120
Logging	143
Logging triggers	68

M

Macro Recorder	76
Main menu	24
Measurement.....	55, 61
Measurement files.....	67
Measurement setup	61
Measurement Setup window	84
Message attributes.....	59
MOST.....	123
MOST configuration	124
MOST function catalog	124
MOST network.....	125
Multiple channels	65
Multiple databases	64

N

NetIsolator.....	163
NMEA 2000®	138

O

Offline mode.....	67
Online mode.....	66
Option CANaerospace	173
Introduction	174
Sample configurations	174
Option CANopen.....	151
Add-ons.....	154
Introduction	152
Quick start.....	152
Sample configurations	160
Use cases	155, 156, 160
Option IP	161
Installation.....	162
Introduction	162
Quick start.....	165

Sample configurations	166
Use cases	164
Option ISO11783	147
Introduction	148
Quick start.....	148
Sample configurations	148
Use cases	148
Option J1587	167
Introduction	168
Prerequisites	168
Sample configurations	168
Option J1939	137
Introduction	138
Quick start.....	138
Sample configurations	140
Use cases	140
Overview of the programs	26

P

Panel Designer	105
Panels	55, 67, 121, 122, 133, 164
Phase 1	22
Phase 2.....	22
Phase 3.....	22
ProCANopen.....	152, 160
Process data dictionary	149
Program start	56

Q

Quick start	
Option IP	165
Option ISO11783.....	148
Option J1939	138

R

Radio button	25
Realtime.....	75
Registered trademarks	9
Remote access	165
Replay.....	143
Replay block	102

S

Sample configurations	
Option CANaerospace.....	174

Option CANopen.....	160
Option IP	166
Option ISO11783	148
Option J1587.....	168
Option J1939.....	140
Shortcut menu.....	24
Signals	143
Simulation	55, 58, 141, 144
Simulation generation	155
Simulation mode	58, 66
Simulation model	156
Simulation setup	58
Simulation Setup window.....	83
Single Trigger.....	68
Slider	25
Spin control	25
Standalone Manager	75
Standalone Mode.....	75
State Monitor window.....	90
Statistics Monitor window	94
Statistics window.....	92
Step Sequencer	76
Support	9
System verification.....	60

T

Test	142
Test Automation Editor	157
Test configurator	156
test environments.....	96
Test Feature Set (TSF).....	71
Test framework	157
Test functionality	55, 70
Test generation	156
Test modules	71
Test report.....	71
Test Service Library (TSL).....	73
Test Setup window.....	96
Test targets	73
Text input box	25

Tips for using CANoe.....	24
Toggle Trigger	69
Trace window.....	87, 132, 153, 164, 170, 175
Trigger block	102, 143
Troubleshooting	77
TSF	55, 71
TSL	55, 73

U

Unit 1.....	33
Unit 2.....	37
Unit 3.....	41
Unit 4.....	43
Unit 5.....	45
Unit 6.....	46

Unit 7.....	47
Unit 8.....	49

V

Virtual Terminal.....	148
-----------------------	-----

W

Warranty	9
Window management	82
WLAN to CAN	165
Working with databases.....	62
Working with multiple channels	65
Write window.....	91

X

XML.....	71
----------	----

Get more Information!

Visit our Website for:

- > News
- > Products
- > Demo Software
- > Support
- > Training Classes
- > Addresses

www.vector.com