GM Support in CANoe 7.2

Version 2.1
2010-02-23

Application Note AN-IND-8-003

Author(s)	Siegfried Beeh, Friederike Gengenbach
Restrictions	Customer Confidential: GM worldwide and GM suppliers

Abstract	This application note explains the current stage of GM support in CANoe 7.2 and its component Test Feature Set.

1.0 Overview

1.1 Introduction

As a simulation, analysis and testing tool for electronic control units, CANoe supports the GMLAN 3.0 protocol in a special way. This document briefly describes the support provided by CANoe specifically for GMLAN and explains the possible use cases of the “Test Feature Set” for GMLAN in detail. This document is based on CANoe version 7.2. Because CANoe support for GMLAN is extended continuously, we recommend that the reader checks whether a newer version of this document is available.

The general Test Feature Set concept is explained in a separate document; see chapter 6.0. Familiarity with the Test Feature Set is assumed in this document.

1.2 Terms and Abbreviations

Term	Meaning
CANoe	Vector product
SUT	System under Test
TFS	Test Feature Set →CANoe component
TSL	Test Service Library →TFS component
CAPL	CANoe Access Programming Language →Language used for programming simulation nodes, test modules etc. in CANoe.
hsCAN	High speed CAN Bus (GM-specific terminology)
swCAN	Single wire CAN Bus (GM-specific terminology)
GM extended message	Message with a 29-bit identifier
Standard message	Message with an 11-bit identifier
Panel Designer	Vector product for editing panels
Panel Editor	Vector product for editing panels
Test Automation Editor	Vector product for editing XML test modules

Table 1: Terms and Abbreviations

2.0 Basic Support in CANoe

Basic support for GMLAN is activated automatically as soon as a GMLAN network file is identified. The existence and contents of the “UseGMParameterIds” network attribute are used as identification criteria.

Details about basic support, which are available in the Online Help for CANoe, are referenced briefly in the following chapters.

2.1 Trace Window

Separate columns display GM-specific “SourceID”, “Prio” and “ParameterID” information. The “GMLAN” base configuration can be selected. This contains basic information along with additional GM-relevant default columns such as “Prio”, “PID”, “SourceID” and “CAN-ID”.

2.2 Logging

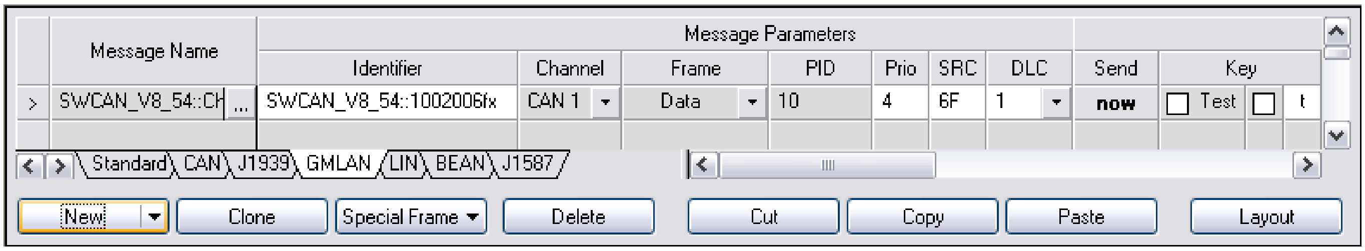
GM-extended messages are logged along with standard messages. The CAN-ID is decoded in the comment field and documented separately as “ParameterID”, “SourceID” and “Prio”.

2.3 Export

Signal histories for standard messages can be exported from the log block, but this is not possible for signals on GM-extended messages.

2.4 Interactive Generator Block

The Interactive Generator Block (IG) allows GM-extended messages to be sent. A special “GMLAN” tab is provided for this purpose, as well as the ability to modify the message priority and SourceID.



	Message Name	Message Parameters							Send	Key		
		Identifier	Channel	Frame	PID	Prio	SRC	DLC				
>	SWCAN_V8_54::CH ...	SWCAN_V8_54::1002006fx	CAN 1	Data	10	4	6F	1	now	☐ Test	☐	t

Figure 1: Defining a message in the Interactive Generator Block

2.5 Generator Block

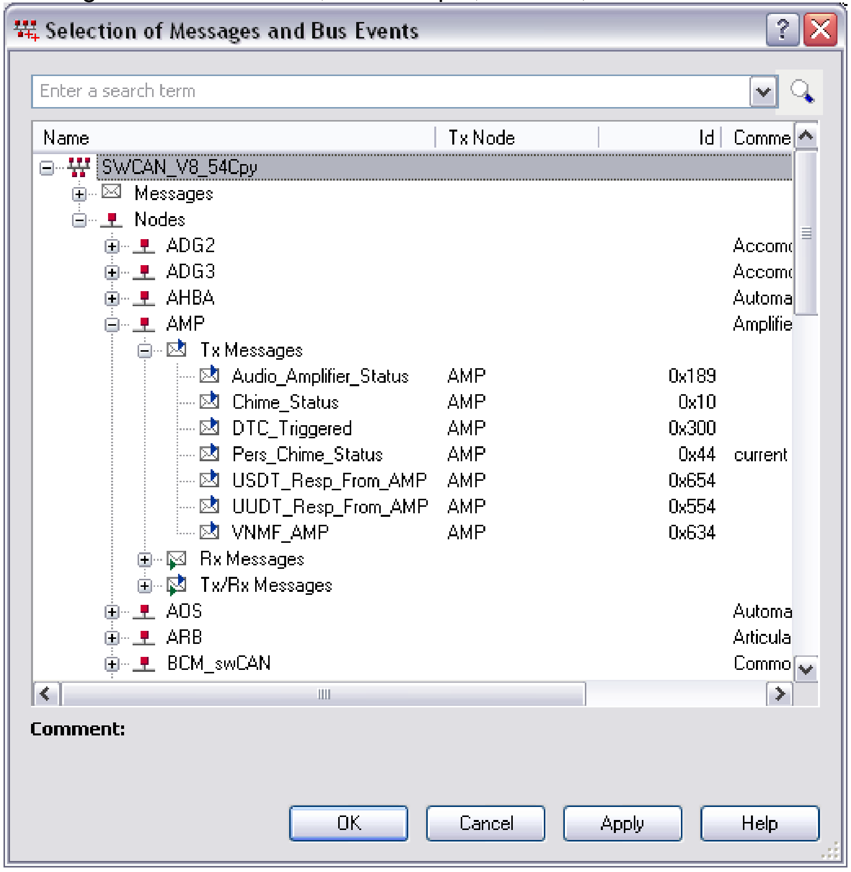
The Generator Block makes it possible to send GM-extended messages in symbolic entry mode.

2.6 Symbol Selection Dialog

The symbolic selection dialog makes it possible to select symbols from the network database (*.dbc).

2.6.1 Selecting a Message

Messages can be selected, for example, in filters, in the Interactive Generator Block and in the CAPL browser.



Name	Tx Node	Id	Comme
SWCAN_V8_54Cpy			
Messages			
Nodes			
ADG2			Accom
ADG3			Accom
AHBA			Automa
AMP			Amplifie
Tx Messages			
Audio_Amplifier_Status	AMP	0x189	
Chime_Status	AMP	0x10	
DTC_Triggered	AMP	0x300	
Pers_Chime_Status	AMP	0x44	current
USDT_Resp_From_AMP	AMP	0x654	
UUDT_Resp_From_AMP	AMP	0x554	
VNMF_AMP	AMP	0x634	
Rx Messages			
Tx/Rx Messages			
AOS			Automa
ARB			Articula
BCM_swCAN			Commo

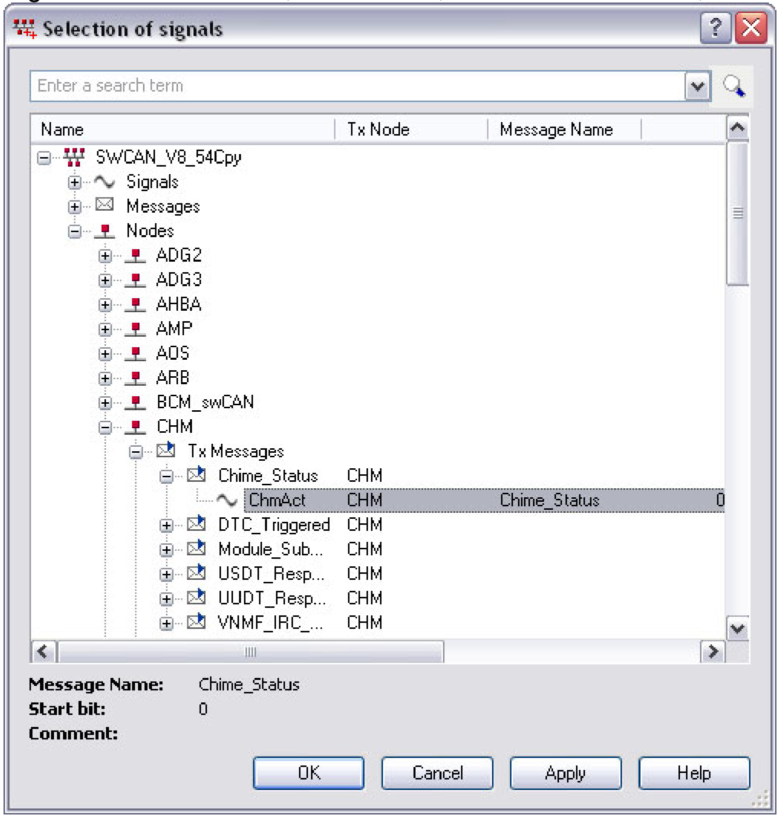
Comment:

Figure 2: Message selection in the symbolic selection dialog

Only the message name is included in the **CAPL** browser, regardless of the path followed in the selection dialog.

2.6.2 Selecting a signal

Signals can be selected, for instance, in the CAPL browser, the Data Window and the Graphics Window.



Name	Tx Node	Message Name	
SWCAN_V8_54Cpy			
Signals			
Messages			
Nodes			
ADG2			
ADG3			
AHBA			
AMP			
AOS			
ARB			
BCM_swCAN			
CHM			
Tx Messages			
Chime_Status	CHM		
ChmAct	CHM	Chime_Status	0
DTC_Triggered	CHM		
Module_Sub...	CHM		
USDT_Resp...	CHM		
UUDT_Resp...	CHM		
VNMF_IRC_...	CHM		

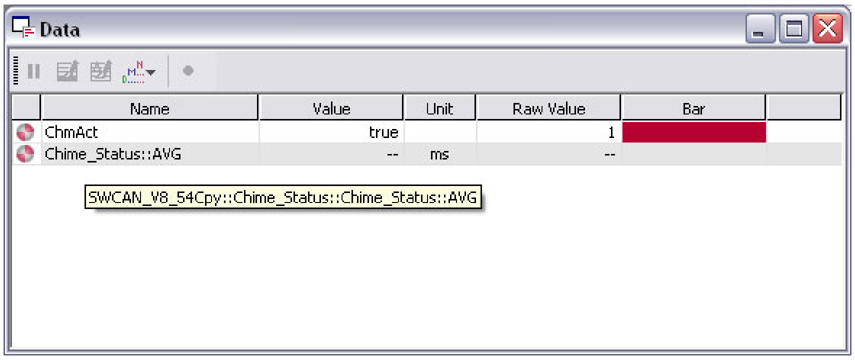
Message Name: Chime_Status
Start bit: 0
Comment:

Figure 3: Signal selection in the symbolic selection dialog

Only the signal name is included in the **CAPL browser**, regardless of how the selection dialog is navigated. If additional qualifiers are required, these can be inserted in the CAPL browser by choosing “Insert message from CANdb++…”.

2.7 Data Window

Application signals are fully supported in the Data Window.



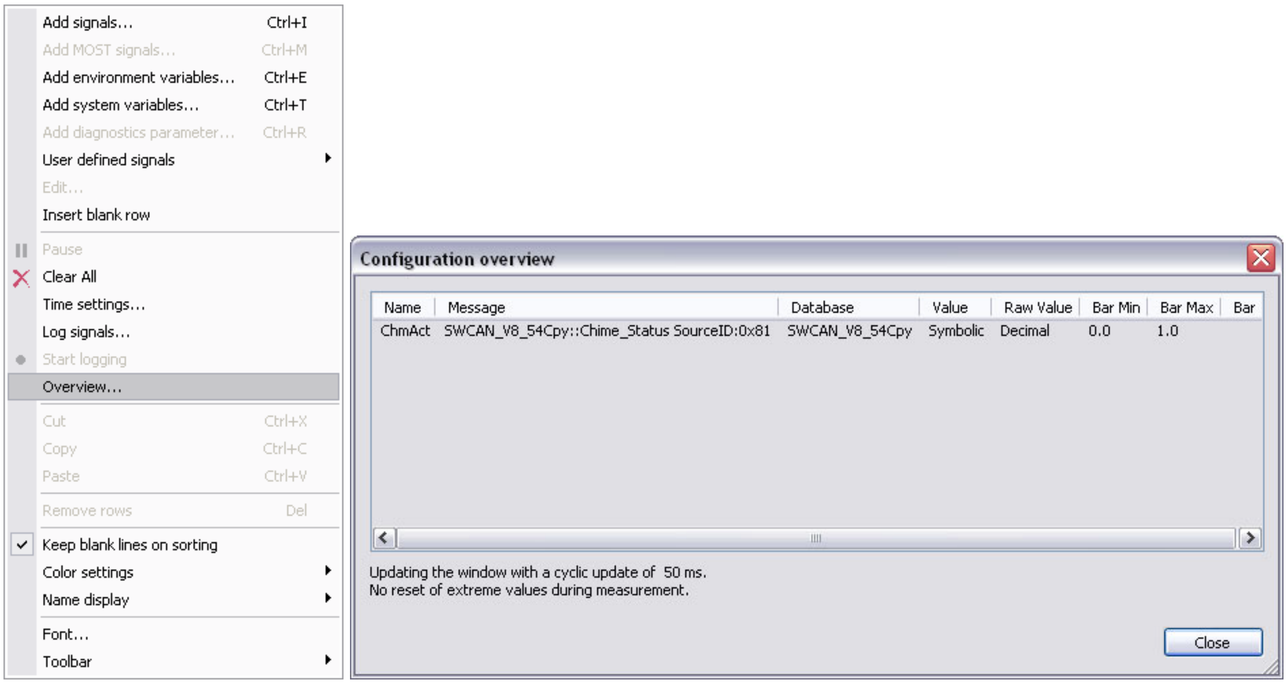
Name	Value	Unit	Raw Value	Bar
ChmAct	true		1	🟥
Chime_Status::AVG	--	ms	--	

SWCAN_V8_54Cpy::Chime_Status::Chime_Status::AVG

.

Figure 4: Data Window

The send node of the signals can be checked in the special dialog “configuration overview” that is accessible from the context menu of the Data Window:



Name	Message	Database	Value	Raw Value	Bar Min	Bar Max	Bar
ChmAct	SWCAN_V8_54Cpy::Chime_Status SourceID:0x81	SWCAN_V8_54Cpy	Symbolic	Decimal	0.0	1.0	

Updating the window with a cyclic update of 50 ms.
No reset of extreme values during measurement.

Figure 5: Configuration overview

2.7.1 Special signals in the Data window

Message-specific statistical signals can be selected in the Data and Graphics windows. These are available for standard messages but not for GM-extended messages.

2.8 Graphics Window

The Graphics Window allows the display of signals on a timeline. The notes that apply to the Data Window apply equally to the selection of signals and special signals in the Graphics Window (see 2.7).

2.9 Panels

A panel may contain elements used to visualize and control signal values. The notes that apply to the Data Window apply similar to the selection of signals here (see 2.7).

The Panel Editor selects a signal via a symbol path that contains the database name, the tx node, the tx message and the signal. For older configurations (created with older versions than CANoe 7.0), it must be ensured that the tx node is part of the symbol path (see area highlighted in red).

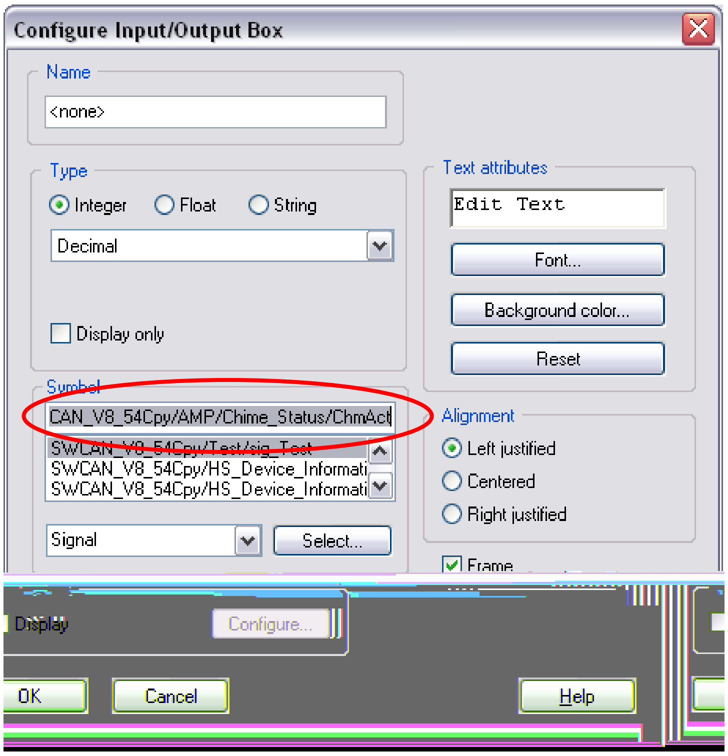


Figure 6: Configuration dialog for an “Input/Output Box” panel control

The Panel Designer fully supports GM by automatically inserting the tx node.

2.10 CAPL

A special message type – “gmlanMessage” – is available for GM-extended messages in CAPL, the CANoe programming language. With this message type, the Prio and SourceID parameters present in addition to the ParameterID can be queried using operators and set using special functions.

Relevant CAPL code fragments are available e.g. in chapter 4.3.

3.0 “GM Package” Modeling Package

The “GM Package” modeling package contains the following special GM-specific components:

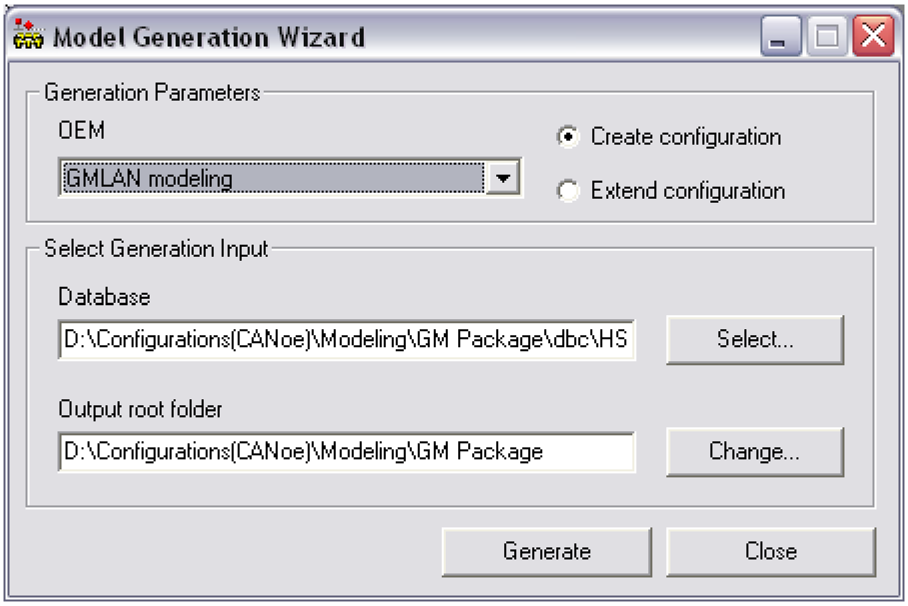
- Interaction layer
- Network management
- Model generation
- Message panels for visualizing and controlling signal values.

Since the supplied manual of the GM Package provides details on the modeling package and the components included in it, the following chapter discusses the functionality of the package only briefly.

The modeling package is not included in the default CANoe installation, but can be requested from Vector support mailto:support@de.vector.com as a separate package free of charge.

3.1 Model Generation

The model generation wizard included in CANoe has been extended to include a GM option. swCAN and hsCAN networks can be generated.



OEM: GMLAN modeling

[✓ Create configuration] Extend configuration

Database: D:\Configurations(CANoe)\Modeling\GM Package\dbc\HS

Output root folder: D:\Configurations(CANoe)\Modeling\GM Package

Figure 7: Model Generation Wizard

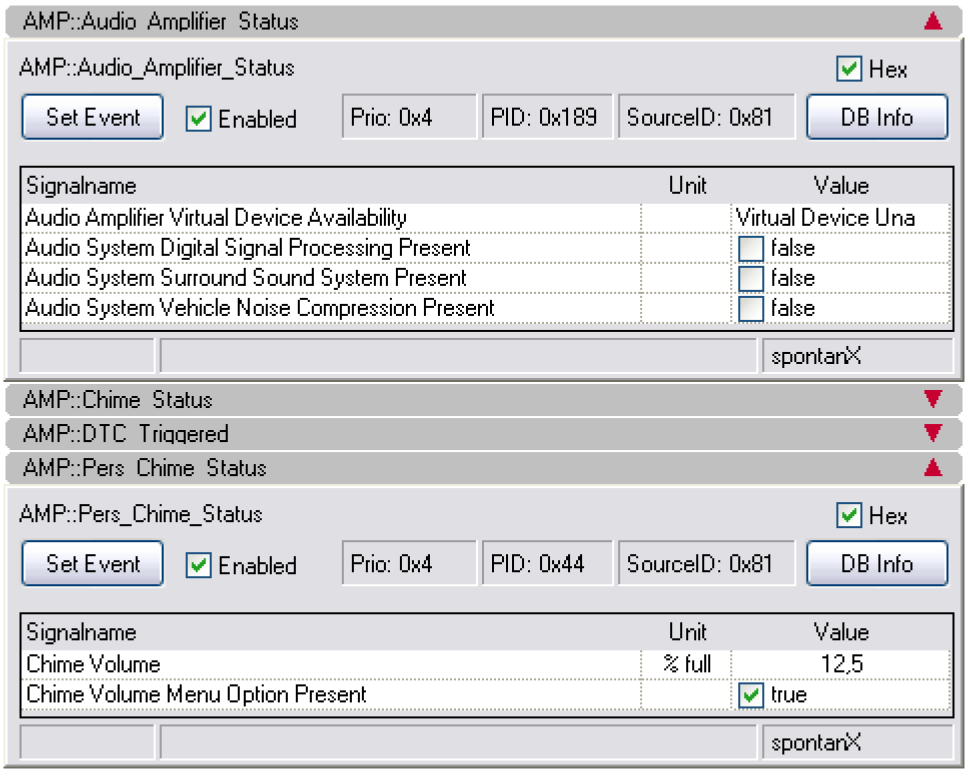
It is also possible to generate a multi-bus configuration.

Generation produces a complete CANoe configuration; the only action required is to select a node from the alternative nodes in the GM network file and to deactivate or delete unused nodes in the simulation setup.

3.2 Node Panels, Message Panels

The configuration generated by the model generation wizard uses message controls developed especially for GM. These message controls make it possible, for example, to modify the signal values that are transmitted by the simulated nodes. The panels can be used to visualize signal values for concretely existing nodes.

These controls can be created/changed in the “Panel Editor”.



AMP::Audio_Amplifier_Status

AMP::Audio_Amplifier_Status ✓ Hex

Set Event | ✓ Enabled | Prio: 0x4 | PID: 0x189 | SourceID: 0x81 | DB Info

Signalname	Unit	Value
Audio Amplifier Virtual Device Availability		Virtual Device Una
Audio System Digital Signal Processing Present		☐ false
Audio System Surround Sound System Present		☐ false
Audio System Vehicle Noise Compression Present		☐ false

spontanX

AMP::Chime_Status

AMP::DTC_Triggered

AMP::Pers_Chime_Status

AMP::Pers_Chime_Status ✓ Hex

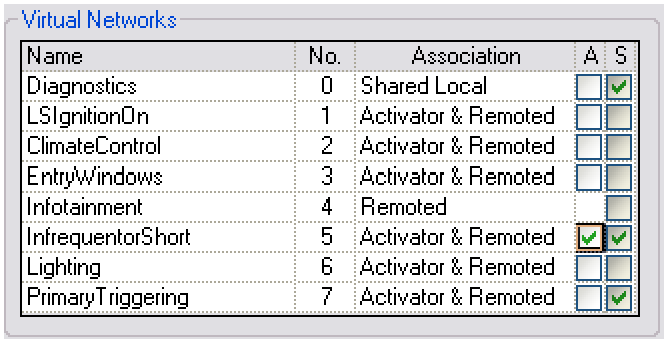
Set Event | ✓ Enabled | Prio: 0x4 | PID: 0x44 | SourceID: 0x81 | DB Info

Signalname	Unit	Value
Chime Volume	% full	12,5
Chime Volume Menu Option Present		✓ true

spontanX

Figure 8: Panel for changing and visualizing all send signals of a node in their messages

In addition, a network management panel is generated for each node with which the virtual network status for existing simulated nodes can be visualized and modified.



Virtual Networks

Name	No.	Association	A	S
Diagnostics	0	Shared Local	☐	✓
LSIgnitionOn	1	Activator & Remoted	☐	☐
ClimateControl	2	Activator & Remoted	☐	☐
EntryWindows	3	Activator & Remoted	☐	☐
Infotainment	4	Remoted		☐
InfrequentorShort	5	Activator & Remoted	✓	✓
Lighting	6	Activator & Remoted	☐	☐
PrimaryTriggering	7	Activator & Remoted	☐	✓

Figure 9: Panel for controlling and visualizing the network status for the current node

3.3 Interaction Layer

The interaction layer can be integrated into individual simulation nodes as a modeling DLL. It is used automatically when a model is generated using the model generation wizard. The interaction layer configures itself on the basis of the network database. Changes to the network database are therefore integrated automatically. The tasks of the individual components are:

- To provide a signal-oriented interface, i.e. individual signal values can be set as signal stimulants from within in CAPL, panels and the Test Feature Set. When using the COM server, then ensure to use the signal objects that are “outputs” or “inputs” of nodes.
- To utilize the network database’s transmission model. Cyclical transmissions, spontaneous transmissions in reaction to changes, etc. are sent to the bus in accordance with the transmission model.
- For some test cases, it is also necessary to be able to send messages on request. This is supported as well.
- Linkage to the network management: The “virtual networks” observed by the network management are known to the interaction layer. This affects whether a transmission should actually be sent or not.
- To provide a standard fault injection interface which is suitable to control the sending of messages from within CAPL. For more information about these functions refer to the CANoe Online Help. These functionalities can also be used comfortable in CANoe Test Modules.

3.4 Network Management

Network management is also available as a special modeling DLL. This component also configures itself based on information contained in the network database.

The individual tasks of the components are:

- To control bus communications when waking up virtual networks and keeping them awake, insofar as the node‘s role is that of an activator.
- To observe bus communications in order to detect changes in the status of the virtual networks that control the node.
- Depending on the status of the virtual networks, the interaction layer is informed and its transmissions are suppressed completely. The virtual networks can be activated or deactivated manually in CAPL or on the panel.
- To provide a CAPL interface that makes it possible to control virtual networks or remain informed of virtual network status changes from within the application.

4.0 Modeling Control Units (ECUs) in CAPL

In principle, the following message types are available with GMLAN:

- Standard messages (with 11-bit message IDs)
- GM-extended messages (with 29-bit message IDs)

Messages of both types can be sent as well as received:

4.1 Signal Access

To access to signal values (set and get signal value) there is a tight way available by using the $-operator:

```
dword sigValue;
$ECC::A2RAlrmTime = 10;           // set signal value
sigValue = $ECC::A2RAlrmTime;     // get signal value
```

4.2 Receiving Standard Messages

Standard messages can be received and evaluated in the usual way by defining “on message” handlers:

```
on message Dynamic_Normal_Force
{
  long val;
  val = this.DynNrmlFrcFrLfWhl;  // signal access

  // Application code …
}
```

4.3 Receiving GM-Extended Messages

GM-extended messages can be received and evaluated by defining “on gmlanmessage” handlers: The message handler can then be programmed specifically for one or more SourceIDs:

```
on gmlanmessage Chime_Status
{
  long val;
  if(gmLanGetSourceId(this) == AMP.SourceId)
  {
    val = this.ChmAct;
    // Application code …
  }
}
```

4.4 Receiving Signals Directly

Signals can be received directly in CANoe by defining an “on signal” handler in CAPL. This is supported for signals on standard messages but not currently for signals on GM-extended messages.

4.5 Sending Standard Messages

Standard messages are sent according to the following general principle:

```
message Dynamic_Normal_Force msg;
msg.DynNrmlFrcFrLfWhl = 1;  // example signal value
output(msg);
```

4.6 Sending GM-Extended Messages

Before a GM-extended message is sent, the SourceID must be specified; whereas the Prio is added automatically:

```
gmlanmessage ChimeStatus msg;
gmLanSetSourceId(msg, AMP.SourceId);
output(msg)
```

Hint: When working signal oriented (4.1), the message is sent by the Interaction layer; no need to care about SourceIDs.

5.0 Test Feature Set

5.1 Test Feature Set in CAPL

A typical test sequence consists of

- Stimulation of signals or messages (SUT input vector)
- Allowance for a SUT settling time until the expected reaction is to occur, and
- Evaluation of signal states or message events (SUT output vector) in response to the stimulation.

The allowed settling time can be shortened by specifying a concrete event in the test sequence, e.g. receipt of a message with a specific message ID.

Test sequences can be structured into test cases, test groups and test modules.

The CANoe test sequence and test structure apply equally to GMLAN, regardless of whether the test modules are formulated in CAPL, XML or .NET. There are, however, a number of particularities, which are explained in the following chapters.

5.1.1 Configuring the Restbus Simulation

The restbus simulation is used directly by the Test Feature Set. Based on the interaction layer configured there, it takes charge of sending messages and thus of signal stimulation. Consequently, the prerequisite for the stimulation of signal values is that the restbus simulation node is present in the simulation setup and is assigned to the database node, and that the “CANoe GM Interaction Layer” modeling DLL is assigned to the simulation node.

Note:	The restbus simulation will be correctly and fully configured if it is created using the “model generation wizard” that comes with the modeling package. Use of this generation method is therefore recommended.

5.1.2 Signal Abstraction – Evaluation

Note:	For signals that are mapped onto GM extended messages the signal symbol has to be prefixed by the send node, e.g. checkSignalInRange(ECC::A2RAlrmTime, 10, 20).

A signal value can be compared directly to a specified interval using the following functions:

```
long checkSignalInRange(dbSignal aSignal,
                        float aLowLimit,
                        float aHighLimit);
```

Combined signal value queries, immediate comparison with a specified interval and automatic recording of the actual and target values for the comparison results are also possible by means of special functions:

```
long testValidateSignalInRange (char aTestStep[],
                                dbSignal aSignal,
                                float aLowLimit,
                                float aHighLimit)

long testValidateSignalOutsideRange (char aTestStep[],
                                     dbSignal aSignal,
                                     float aLowLimit,
                                     float aHighLimit)
```

Please refer to the CANoe Online Help for further details about these functions.

5.1.3 Signal Abstraction – Stimulation

Signal values can be stimulated with the $-operator (4.1). This requires (at least for the specified Tx node) a configured restbus simulation (5.1.1).

```
$ECC::A2RAlrmTime = 10;
```

If a signal that is already being sent by a real ECU is to be stimulated as part of a test sequence, then the real ECU should be stimulated, thereby indirectly causing the changed signal value to be sent.

Using the signal stimulation within the Test Feature Set the stimulation is reported in the test report.

5.1.4 Message Abstraction – Evaluation

There are also **wait functions** available for testing message events, with which the test sequence pauses until the event condition is fulfilled. Simulation node techniques (see 4.1 and 4.3) are available likewise. These wait functions can be used either symbolically by specifying a message symbol from the network file or numerically by specifying a message ID.

For **standard messages** these are:

```
long TestWaitForMessage(dbMessage aMsg, dword aTimeout)
long TestWaitForMessage(long aMsgId, dword aTimeout)
```

For **GM-extended messages**, only the numeric function for passing the message ID is available; alternative ways of using the wait function are described below:

a) Symbolic setup via instantiation of a message buffer

The “aMsgId” expected in the function is created by means of a series of calls. The example below is constructed for the send node “AMP” that transmits the message “Chime_Status”, with a maximum wait time of 2000ms:

```
// Example signal evaluation on message level
testcase a ()
{
  gmlanmessage Chime_Status msg;

  if(testWaitForMessage(gmLanId(Chime_Status, AMP), 2000) == 1)
  {
    testGetWaitEventMsgData(msg);

    if(msg.ChmAct == 1)
      TestStepPass("A", "Message received, signal matches");
    else
      TestStepFail("A", "Message received, signal does not match");
  }
  else
    TestStepFail("A", "Message missing");
}
```

Commonly, application level tests are done by using signal access and signal test functions. Message level checks are done mostly by utilizing the TSL checks (refer 5.3 for details).

b) Numerical setup via modification of a message buffer

If the name of the message or send node is not known in CAPL and the message ID is to be set up in a strictly numerical manner, the following steps are necessary:

```
// initialise local variables
gmlanmessage * msg1;

// do something, e.g. stimulate the SUT

// Set up and wait for response
msg1.id = (0x10);
gmLanSetSourceId(msg1, 0x81);
gmLanSetPrio(msg1, 4);
TestWaitForMessage(msg1.id, 2000);
```

5.1.5 Message Abstraction – Stimulation

GM messages are sent in the same way as in simulation nodes. Standard GM messages are handled as described in the CANoe Online Help. Only the GM-extended messages require special treatment: The “Prio” and “SourceID” parameters can be added to a CAN-ID in the following ways:

Numerically, within the variable definition:

```
testcase numericOutput()
{
  gmlanmessage 0x10x msg1 = {SA =0x81, PRIO= 0x4};
  output(msg1);
}
```

Specifying PRIO (for the message priority) is optional, as this is taken from the message definition. If it is not specified, the SA selector (for the SourceID) is taken to be 0. For this reason, it is absolutely necessary to add the SourceID by using, for example, the following process (see also 4.6):

```
testcase symbolicOutputProgrammatic()
{
  gmlanmessage Chime_Status msg2;
  gmLanSetSourceId(msg2, AMP.SourceId);
  output(msg2);
}
```

5.1.6 Interaction Layer Fault Injection Functions

There are also **Fault Injection** functions available which are suitable to control the sending of messages, e.g. to change the cycle time, to suppress messages completely or to change the DLC of messages. For more information about these functions refer to the CANoe Online Help.

These fault injection functions can also be used for messages with extended IDs. Therefore the ID-based functions have to be used:

```
id_DDM = gmLanId(Driver_Door_Status, DDM);
TestDisableMsg(id_DDM);
```

This example disables the sending of the message “Driver_Door_Status” of node “DDM”.

5.2 Test Feature Set in XML

GMLAN requires no new constructs with the Test Feature Set in XML test modules. The existing qualification of signals using send nodes can be used directly here and must be used for GM-extended messages and their signals.

The table below shows the existing test functions and their GM support:

Test function name	Usage in test modules **for messages with standard or GM-extended IDs**
Await Value Match	✓
CANstress Configuration	✓ Trigger/disturbance definition for standard IDs ⃠ Trigger/disturbance definition for GM-extended IDs
CANstress State	✓
Diagnostics Service	✓
Diagnostics Service Check (replaced by Diagnostics Service)	✓
Initialize	✓ for signal abstraction ✓ for message abstraction
Replay	✓
Request Response	✓
Set	✓ for signal abstraction ✓ for message abstraction
State Change	✓
State Check	✓
Stimulate Ramp	✓
System Call	✓
Tester Confirmation	✓
Until End	✓
VT System Configuration	✓
Wait	✓
Window Capture	✓

Table 2: XML Test Functions

Syntax example for GM-extended:

```
<testcase title="my GM test case">
  <statechange wait="200ms">
    <in>
      <cansignal id="stim_sig" node="sender">
       10
      </cansignal>
    </in>
    <expected>
      <cansignal id="expected_sig" node="sender">
        <eq>10</eq>
      </cansignal>
    </expected>
  </statechange>
</testcase>
```

5.3 Test Service Library in CAPL and XML Test Modules

The table below shows the existing check objects (“TSL checks”) and their usage for different message types:

Check base name	Usage in test modules **for messages with standard IDs**		Usage in test modules **for messages with GM-extended IDs**	
	CAPL	XML	CAPL	XML
Cycle Time (Abs)				
For message	✓	✓	✓ (ID-based)	✓
Cycle Time (Rel)				
For message	✓	✓	✓ (ID-based)	✓
For node	✓	✓	✓	✓
DLC Monitoring				
For single message	✓	✓	✓ (ID-based)	✓
For all tx messages of node	✓	✓	✓	✓
For all rx messages of node	✓	✓	✓	✓
Error Frame Check	✓	✓	✓	✓
Fallback Observation	--	✓	--	✓
Message Distance	✓	✓	✓ (ID-based)	✓
Unknown Message Received	✓	✓	✓	✓
Signal Value Constancy	✓	✓	✓	✓
Node Active				
For message list	--	✓	--	✓
For single node	✓	✓	✓	✓
For all nodes	✓	--	✓	--
Node Inactive				
For message list	--	✓	--	✓
For single node	✓	✓	✓	✓
For all nodes	✓	--	✓	--
Absence of Defined Message	✓	✓	✓ (ID-based)	✓
Occurrence of a Message				
For single message	✓	✓	✓ (ID-based)	✓
For node	✓	--	✓	--
Request Response Check	--	✓	--	✓
Value Invariant	--	✓	--	✓
Signal Value	✓	✓	✓	✓
Timeout Check	✓	✓	✓	✓

Table 3: Test Service Library Checks

Symbols	**Meaning**
✓	Fully supported
--	Generally not available (specific feature of XML test modules or CAPL test modules)
⃠	Not supported for GM-extended messages

5.4 Test Automation Editor

The Test Automation Editor 1.1 supports fully the GMLAN networks and the editing of XML test modules.
It provides a setting to fill in always the send node of an added symbol.

For further information about the separate Vector product Test Automation Editor, see http://www.vector.com/tae.

5.5 CANdb++ Test Module Generator

The CANdb++ test module generator is available as standard component within the CANoe installation. It can be accessed from the context menu of node objects, or within the "File→PlugIns" menu.

The CANdb++ Test Module Generator uses the check objects in the Test Service Library and fully supports GM. If a configuration contains multiple buses, it is important to specify the bus name for the generation.

Please refer to the CANoe Online Help for more detailed information about the test module generator.

6.0 Additional Resources

VECTOR APPLICATION NOTE
AN-IND-1-002 Testing with CANoe

7.0 Contact

Germany
and all countries not named below:
Vector Informatik GmbH
Ingersheimer Str. 24
70499 Stuttgart
GERMANY
Phone: +49 711-80670-0
Fax: +49 711-80670-111
E-mail: info@de.vector.com

France, Belgium, Luxemburg:

Vector France SAS
168 Boulevard Camélinat
92240 Malakoff
FRANCE
Phone: +33 1 42 31 40 00
Fax: +33 1 42 31 40 09
E-mail: information@fr.vector.com

Sweden, Denmark, Norway,
Finland, Iceland:
VecScan AB
Theres Svenssons Gata 9
41755 Göteborg
SWEDEN
Phone: +46 31 764 76 00
Fax: +46 31 764 76 19
E-mail: info@se.vector.com

United Kingdom, Ireland:
Vector GB Ltd.
Rhodium
Central Boulevard
Blythe Valley Park
Solihull, Birmingham
West Midlands B90 8AS
UNITED KINGDOM
Phone: +44 121 50681-50
E-mail: info@uk.vector.com

China:
Vector Informatik GmbH
Shanghai Representative Office
Suite 605, Tower C,
Everbright Convention Center
No. 70 Caobao Road
Xuhui District
Shanghai 200235
P.R. CHINA
Phone: +86 21 - 6432 5353 ext. 0
Fax: +86 21 - 6432 5308
E-mail: info@vector-china.com

India:
Vector Informatik India Private Ltd.
4/1/1/1 Sutar Icon
Sus Road
Pashan
Pune 411021
INDIA

Phone: +91 9673 336575
E-mail: info@vector-india.com

USA, Canada, Mexico:
Vector CANtech, Inc.
39500 Orchard Hill Pl., Ste 550
Novi, MI 48375
USA
Phone: +1 248 449 9290
Fax: +1 248 449 9704
E-mail: info@us.vector.com

Japan:
Vector Japan Co. Ltd.
Seafort Square Center Bld. 18F
2-3-12, Higashi-shinagawa,
Shinagawa-ku
Tokyo 140-0002
JAPAN
Phone: +81 3 5769 7800
Fax: +81 3 5769 6975
E-mail: info@jp.vector.com

Korea:
Vector Korea IT Inc.
#1406, Mario Tower,
222-12 Guro-dong, Guro-gu
Seoul, 152-848
REPUBLIC OF KOREA
Phone: +82 2 807 0600
Fax: +82 2 807 0601
E-mail: info@kr.vector.com